aws

Panduan Pengguna

Amazon EKS

Amazon EKS: Panduan Pengguna

Table of Contents

Apa itu Amazon EKS?

Amazon Elastic Kubernetes Service (Amazon EKS) adalah layanan terkelola yang menghilangkan kebutuhan untuk menginstal, mengoperasikan, dan memelihara bidang kontrol Kubernetes Anda sendiri di Amazon Web Services ().AWSKubernetesadalah sistem sumber terbuka yang mengotomatiskan manajemen, penskalaan, dan penyebaran aplikasi kontainer.

Fitur Amazon EKS

Berikut ini adalah fitur utama Amazon EKS:

Jaringan dan otentikasi yang aman

Amazon EKS mengintegrasikan Kubernetes beban kerja Anda dengan layanan AWS jaringan dan keamanan. Ini juga terintegrasi dengan AWS Identity and Access Management (IAM) untuk memberikan otentikasi untuk cluster Anda. Kubernetes

Penskalaan cluster yang mudah

Amazon EKS memungkinkan Anda untuk meningkatkan skala Kubernetes cluster Anda dengan mudah berdasarkan permintaan beban kerja Anda. Amazon EKS mendukung Podpenskalaan otomatis horizontal berdasarkan CPU atau metrik khusus, dan penskalaan otomatis cluster berdasarkan permintaan seluruh beban kerja.

KubernetesPengalaman terkelola

Anda dapat membuat perubahan pada Kubernetes cluster Anda menggunakan`eksctl`, AWS Management Console, AWS Command Line Interface (AWS CLI), API `kubectl`, dan Terraform.

Ketersediaan tinggi

Amazon EKS menyediakan ketersediaan tinggi untuk pesawat kontrol Anda di beberapa Availability Zone.

Integrasi dengan AWS layanan

Amazon EKS terintegrasi dengan AWS layanan lain, menyediakan platform komprehensif untuk menerapkan dan mengelola aplikasi kontainer Anda. Anda juga dapat lebih mudah memecahkan masalah Kubernetes beban kerja Anda dengan berbagai alat observabilitas.

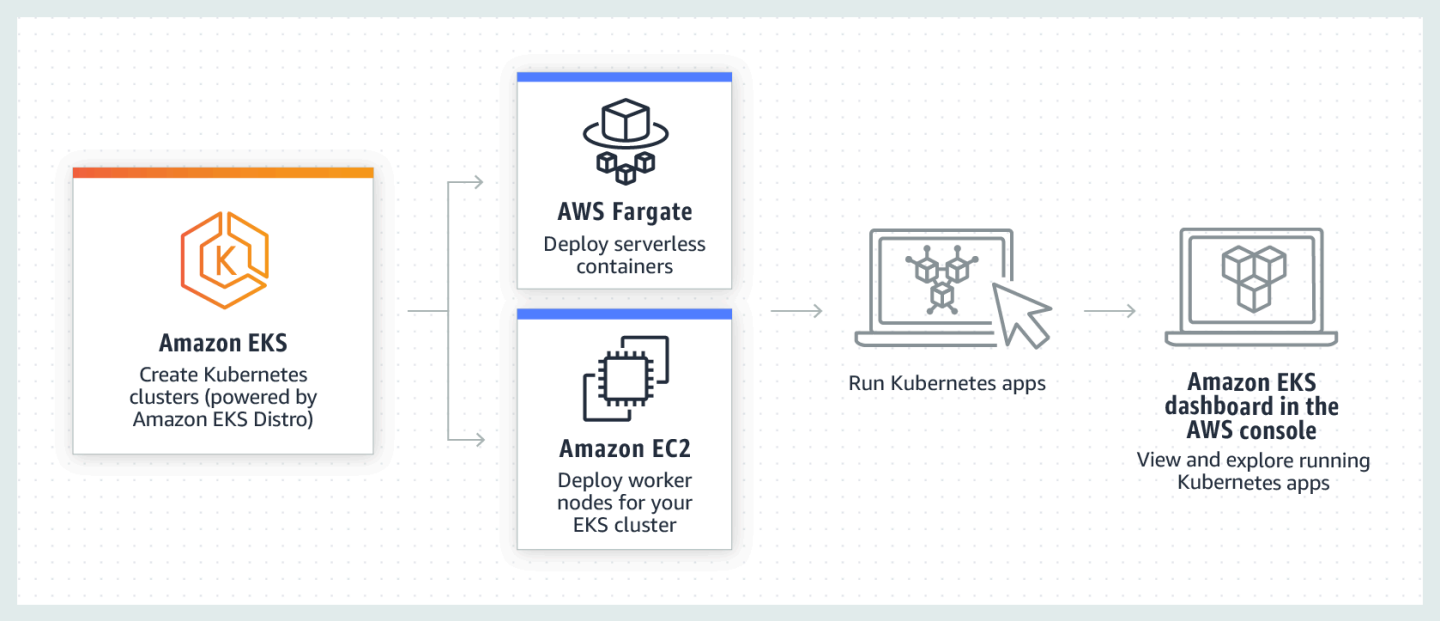
Untuk detail tentang fitur Amazon EKS lainnya, lihat fitur Amazon EKS.

Memulai dengan Amazon EKS

Untuk membuat klaster pertama Anda dan sumber daya yang terkait, lihat Memulai dengan Amazon EKS. Secara umum, memulai dengan Amazon EKS melibatkan langkah-langkah berikut.

1. Buat klaster — Mulailah dengan membuat klaster Anda menggunakan `eksctl` AWS Management Console, AWS CLI,, atau salah satu AWS SDK.
2. Pilih pendekatan Anda untuk menghitung sumber daya — Putuskan antara AWS Fargate,Karpenter, grup node terkelola, dan node yang dikelola sendiri.
3. Pengaturan - Siapkan pengontrol, driver, dan layanan yang diperlukan.
4. Menyebarkan beban kerja — Sesuaikan Kubernetes beban kerja Anda untuk memanfaatkan sumber daya dan kemampuan tipe node pilihan Anda dengan sebaik-baiknya.
5. Manajemen — Mengawasi beban kerja Anda, mengintegrasikan AWS layanan untuk merampingkan operasi dan meningkatkan kinerja beban kerja. Anda dapat melihat informasi tentang beban kerja Anda menggunakan. AWS Management Console

Diagram berikut menunjukkan aliran dasar menjalankan Amazon EKS di cloud. Untuk mempelajari tentang opsi Kubernetes penerapan lainnya, lihatOpsi deployment.



Harga untuk Amazon EKS

Cluster Amazon EKS terdiri dari bidang kontrol dan komputasi Amazon Elastic Compute Cloud (Amazon EC2) atau Fargate yang Anda jalankan. Pods Untuk informasi selengkapnya tentang harga bidang kontrol, lihat harga Amazon EKS. Baik Amazon EC2 dan Fargate menyediakan:

Instans Sesuai Permintaan

Bayar untuk instans yang Anda gunakan pada saat yang kedua, tanpa komitmen jangka panjang atau pembayaran di muka. Untuk informasi selengkapnya, lihat Harga dan Harga Sesuai Permintaan Amazon EC2.AWS Fargate

, Savings Plans

Anda dapat mengurangi biaya Anda dengan membuat komitmen untuk jumlah penggunaan yang konsisten, dalam USD per jam, untuk jangka waktu satu atau tiga tahun. Untuk informasi selengkapnya, lihat Harga dengan Savings Plans.

Kasus penggunaan umum di Amazon EKS

Amazon EKS menawarkan pengelolaan yang kuatKuberneteslayanan padaAWS, dirancang untuk mengoptimalkan aplikasi kontainer. Berikut ini adalah beberapa kasus penggunaan Amazon EKS yang paling umum, membantu Anda memanfaatkan kekuatannya untuk kebutuhan spesifik Anda.

Menyebarkan aplikasi ketersediaan tinggi

MenggunakanPenyeimbangan Beban Elastis, Anda dapat memastikan bahwa aplikasi Anda sangat tersedia di beberapaZona Ketersediaan.

Membangun arsitektur microservices

GunakanKubernetesfitur penemuan layanan denganAWS Cloud MapatauAmazon VPC Kisiuntuk membangun sistem yang tangguh.

Mengotomatiskan proses rilis perangkat lunak

Kelola pipeline continuous integration and continuous deployment (CICD) yang menyederhanakan proses pembuatan, pengujian, dan penyebaran aplikasi secara otomatis.

Menjalankan aplikasi tanpa server

GunakanAWS Fargatedengan Amazon EKS untuk menjalankan aplikasi tanpa server. Ini berarti Anda hanya dapat fokus pada pengembangan aplikasi, sementara Amazon EKS dan Fargate menangani infrastruktur yang mendasarinya.

Menjalankan beban kerja pembelajaran mesin

Amazon EKS kompatibel dengan kerangka kerja pembelajaran mesin populer seperti TensorFlow, MXNet, dan PyTorch. Dengan dukungan GPU, Anda dapat menangani tugas pembelajaran mesin yang kompleks secara efektif.

Menerapkan secara konsisten di tempat dan di cloud

GunakanAmazon EKS Di Mana Sajauntuk beroperasiKubernetescluster di infrastruktur Anda sendiri menggunakan alat yang konsisten dengan Amazon EKS di cloud.

Menjalankan pemrosesan batch yang hemat biaya dan beban kerja data besar

MemanfaatkanContoh Spotuntuk menjalankan pemrosesan batch dan beban kerja data besar seperti Apache HadoopdanSpark, pada sebagian kecil dari biaya. Ini memungkinkan Anda memanfaatkan kapasitas Amazon EC2 yang tidak terpakai dengan harga diskon.

Mengamankan aplikasi dan memastikan kepatuhan

Menerapkan praktik keamanan yang kuat dan menjaga kepatuhan terhadap Amazon EKS, yang terintegrasi denganAWSLayanan keamanan seperti AWS Identity and Access Management(SAYA), Awan Pribadi Virtual Amazon(Amazon VPC), dan AWS Key Management Service(AWS KMS). Ini memastikan privasi dan perlindungan data sesuai standar industri.

Arsitektur Amazon EKS

Amazon EKS selaras dengan arsitektur cluster umum. Kubernetes Untuk informasi selengkapnya, lihat KubernetesKomponen dalam Kubernetes dokumentasi. Bagian berikut merangkum beberapa detail arsitektur tambahan untuk Amazon EKS.

Bidang kontrol

Amazon EKS memastikan setiap cluster memiliki bidang Kubernetes kontrol uniknya sendiri. Desain ini membuat infrastruktur masing-masing cluster terpisah, tanpa tumpang tindih antara cluster atau akun. AWS Pengaturan meliputi:

Komponen terdistribusi

Bidang kontrol memposisikan setidaknya dua instance server API dan tiga `etcd`instans di tiga AWS Availability Zone dalam sebuah. Wilayah AWS

Kinerja optimal

Amazon EKS secara aktif memonitor dan menyesuaikan instans bidang kontrol untuk mempertahankan kinerja puncak.

Ketahanan

Jika instans bidang kontrol terputus-putus, Amazon EKS dengan cepat menggantikannya, menggunakan Availability Zone yang berbeda jika diperlukan.

Waktu aktif yang konsisten

Dengan menjalankan klaster di beberapa Availability Zone, ketersediaan titik akhir server API yang andal Service Level Agreement (SLA) tercapai.

Amazon EKS menggunakan Amazon Virtual Private Cloud (Amazon VPC) untuk membatasi lalu lintas antara komponen pesawat kontrol dalam satu cluster. Komponen klaster tidak dapat melihat atau menerima komunikasi dari kluster atau AWS akun lain, kecuali jika diizinkan oleh kebijakan kontrol akses Kubernetes berbasis peran (RBAC).

Hitung

Selain bidang kontrol, cluster Amazon EKS memiliki satu set mesin pekerja yang disebut node. Memilih jenis node cluster Amazon EKS yang sesuai sangat penting untuk memenuhi persyaratan spesifik Anda dan mengoptimalkan pemanfaatan sumber daya. Amazon EKS menawarkan tipe simpul utama berikut:

AWS Fargate

Fargate adalah mesin komputasi tanpa server untuk kontainer yang menghilangkan kebutuhan untuk mengelola instance yang mendasarinya. Dengan Fargate, Anda menentukan kebutuhan sumber daya aplikasi Anda, dan AWS secara otomatis menyediakan, menskalakan, dan memelihara infrastruktur. Opsi ini sangat ideal untuk pengguna yang memprioritaskan ease-of-use dan ingin berkonsentrasi pada pengembangan dan penyebaran aplikasi daripada mengelola infrastruktur.

Karpenter

Karpenteradalah autoscaler Kubernetes cluster yang fleksibel dan berkinerja tinggi yang membantu meningkatkan ketersediaan aplikasi dan efisiensi cluster. Karpentermeluncurkan sumber daya komputasi berukuran tepat sebagai respons terhadap perubahan beban aplikasi. Opsi ini dapat menyediakan sumber daya just-in-time komputasi yang memenuhi persyaratan beban kerja Anda.

Grup simpul terkelola

Grup node terkelola adalah perpaduan antara otomatisasi dan kustomisasi untuk mengelola kumpulan instans Amazon EC2 dalam klaster Amazon EKS. AWSmenangani tugas-tugas seperti menambal, memperbarui, dan menskalakan node, mengurangi aspek operasional. Secara paralel, `kubelet` argumen kustom didukung, membuka kemungkinan untuk kebijakan manajemen CPU dan memori tingkat lanjut. Selain itu, mereka meningkatkan peran keamanan melalui AWS Identity and Access Management (IAM) untuk akun layanan, sambil membatasi kebutuhan akan izin terpisah per cluster.

Simpul yang dikelola sendiri

Node yang dikelola sendiri menawarkan kontrol penuh atas instans Amazon EC2 Anda dalam klaster Amazon EKS. Anda bertanggung jawab untuk mengelola, menskalakan, dan memelihara node, memberi Anda kendali penuh atas infrastruktur yang mendasarinya. Opsi ini cocok untuk pengguna yang membutuhkan kontrol granular dan penyesuaian node mereka dan siap menginvestasikan waktu dalam mengelola dan memelihara infrastruktur mereka.

Konsep Kubernetes

Amazon Elastic Kubernetes Service (Amazon EKS) AWS adalah layanan terkelola berdasarkan proyek open source. Kubernetes Meskipun ada hal-hal yang perlu Anda ketahui tentang bagaimana layanan Amazon EKS terintegrasi dengan AWS Cloud (terutama ketika Anda pertama kali membuat kluster Amazon EKS), setelah aktif dan berjalan, Anda menggunakan cluster Amazon EKS Anda dengan cara yang sama seperti yang Anda lakukan pada Kubernetes cluster lainnya. Jadi untuk mulai mengelola Kubernetes cluster dan menyebarkan beban kerja, Anda memerlukan setidaknya pemahaman dasar tentang konsep. Kubernetes

Halaman ini membagi Kubernetes konsep menjadi tiga bagian: MengapaKubernetes, Cluster, dan Beban Kerja. Bagian pertama menjelaskan nilai menjalankan Kubernetes layanan, khususnya sebagai layanan terkelola seperti Amazon EKS. Bagian Beban Kerja mencakup bagaimana

Kubernetes aplikasi dibangun, disimpan, dijalankan, dan dikelola. Bagian Cluster menjabarkan berbagai komponen yang membentuk Kubernetes cluster dan apa tanggung jawab Anda untuk membuat dan memelihara cluster. Kubernetes

Topik

Saat Anda membaca konten ini, tautan akan mengarahkan Anda ke deskripsi Kubernetes konsep lebih lanjut di Amazon EKS dan Kubernetes dokumentasi, jika Anda ingin menyelami topik apa pun yang kami bahas di sini. Untuk detail tentang cara Amazon EKS mengimplementasikan bidang Kubernetes kontrol dan fitur komputasi, lihat arsitektur Amazon EKS.

MengapaKubernetes?

Kubernetesdirancang untuk meningkatkan ketersediaan dan skalabilitas saat menjalankan aplikasi kontainer berkualitas produksi yang kritis terhadap misi. Daripada hanya berjalan Kubernetes pada satu mesin (meskipun itu mungkin), Kubernetes mencapai tujuan tersebut dengan memungkinkan Anda menjalankan aplikasi di seluruh set komputer yang dapat memperluas atau kontrak untuk memenuhi permintaan. Kubernetestermasuk fitur yang memudahkan Anda untuk:

- Menerapkan aplikasi pada beberapa mesin (menggunakan kontainer yang di-deploy di Pod)
- Pantau kesehatan kontainer dan mulai ulang kontainer yang gagal
- Skala kontainer naik dan turun berdasarkan beban
- Perbarui wadah dengan versi baru
- Alokasikan sumber daya antar kontainer
- Menyeimbangkan lalu lintas di seluruh mesin

Setelah Kubernetes mengotomatiskan jenis tugas kompleks ini memungkinkan pengembang aplikasi untuk fokus membangun dan meningkatkan beban kerja aplikasi mereka, daripada mengkhawatirkan infrastruktur. Pengembang biasanya membuat file konfigurasi, diformat sebagai file YAMM, yang menggambarkan keadaan aplikasi yang diinginkan. Ini dapat mencakup kontainer mana yang akan dijalankan, batas sumber daya, jumlah replika Pod, alokasi CPU/memori, aturan afinitas, dan banyak lagi.

Atribut dari Kubernetes

Untuk mencapai tujuannya, Kubernetes memiliki atribut berikut:

- Containerized - Kubernetes adalah alat orkestrasi kontainer. Untuk menggunakannyaKubernetes, Anda harus terlebih dahulu memiliki aplikasi Anda dalam wadah. Tergantung pada jenis aplikasi, ini bisa sebagai satu set layanan mikro, sebagai pekerjaan batch atau dalam bentuk lain. Kemudian, aplikasi Anda dapat memanfaatkan Kubernetes alur kerja yang mencakup ekosistem alat yang sangat besar, di mana kontainer dapat disimpan sebagai gambar dalam registri kontainer, disebarkan ke Kubernetescluster, dan dijalankan pada node yang tersedia. Anda dapat membuat dan menguji kontainer individual di komputer lokal Anda dengan Docker atau runtime kontainer lain, sebelum menerapkannya ke cluster AndaKubernetes.
- Scalable - Jika permintaan untuk aplikasi Anda melebihi kapasitas instance yang berjalan dari aplikasi tersebut, Kubernetes dapat ditingkatkan. Sesuai kebutuhan, Kubernetes dapat mengetahui apakah aplikasi memerlukan lebih banyak CPU atau memori dan merespons dengan memperluas kapasitas yang tersedia secara otomatis atau menggunakan lebih banyak kapasitas yang ada. Penskalaan dapat dilakukan pada level Pod, jika ada cukup komputasi yang tersedia untuk menjalankan lebih banyak instance aplikasi (penskalaan otomatis Pod horizontal), atau pada tingkat node, jika lebih banyak node perlu dimunculkan untuk menangani peningkatan kapasitas (Cluster Autoscaler atau Karpenter). Karena kapasitas tidak lagi diperlukan, layanan ini dapat menghapus Pod yang tidak perlu dan mematikan node yang tidak dibutuhkan.
- Tersedia - Jika aplikasi atau node menjadi tidak sehat atau tidak tersedia, Kubernetes dapat memindahkan beban kerja yang berjalan ke node lain yang tersedia. Anda dapat memaksakan masalah hanya dengan menghapus instance yang sedang berjalan dari beban kerja atau node yang menjalankan beban kerja Anda. Intinya di sini adalah bahwa beban kerja dapat diangkat di lokasi lain jika mereka tidak dapat lagi berjalan di tempat mereka berada.
- Deklaratif - Kubernetes menggunakan rekonsiliasi aktif untuk terus-menerus memeriksa apakah status yang Anda deklarasikan untuk klaster Anda cocok dengan status sebenarnya. Dengan menerapkan Kubernetesobjek ke kluster, biasanya melalui file konfigurasi berformat YAML, Anda dapat, misalnya, meminta untuk memulai beban kerja yang ingin Anda jalankan di cluster Anda. Anda nantinya dapat mengubah konfigurasi untuk melakukan sesuatu seperti menggunakan versi kontainer yang lebih baru atau mengalokasikan lebih banyak memori. Kubernetesakan melakukan apa yang perlu dilakukan untuk menetapkan keadaan yang diinginkan. Ini dapat mencakup membawa node ke atas atau ke bawah, menghentikan dan memulai ulang beban kerja, atau menarik kontainer yang diperbarui.
- Composable - Karena aplikasi biasanya terdiri dari beberapa komponen, Anda ingin dapat mengelola satu set komponen ini (sering diwakili oleh beberapa kontainer) bersama-sama.

Meskipun Docker Compose menawarkan cara untuk melakukan ini secara langsungDocker, perintah Kubernetes Kompose dapat membantu Anda melakukannya. Kubernetes Lihat Menerjemahkan File Docker Tulis ke Kubernetes Sumber Daya untuk contoh cara melakukannya.

- Extensible - Tidak seperti perangkat lunak berpemilik, Kubernetes proyek open source dirancang untuk terbuka bagi Anda memperluas cara Kubernetes apa pun yang Anda inginkan untuk memenuhi kebutuhan Anda. API dan file konfigurasi terbuka untuk modifikasi langsung. Pihak ketiga didorong untuk menulis Controller mereka sendiri, untuk memperluas infrastruktur dan prestasi pengguna akhirKubernetes. Webhook memungkinkan Anda menyiapkan aturan klaster untuk menegakkan kebijakan dan beradaptasi dengan perubahan kondisi. Untuk ide lebih lanjut tentang cara memperluas Kubernetes cluster, lihat Memperluas Kubernetes.
- Portable - Banyak organisasi telah menstandarisasi operasi mereka Kubernetes karena memungkinkan mereka untuk mengelola semua kebutuhan aplikasi mereka dengan cara yang sama. Pengembang dapat menggunakan pipeline yang sama untuk membangun dan menyimpan aplikasi kontainer. Aplikasi tersebut kemudian dapat digunakan ke Kubernetes kluster yang berjalan di tempat, di awan, di point-of-sales terminal di restoran, atau di perangkat IOT yang tersebar di seluruh situs jarak jauh perusahaan. Sifat open source-nya memungkinkan orang untuk mengembangkan Kubernetes distribusi khusus ini, bersama dengan alat yang diperlukan untuk mengelolanya.

Mengelola Kubernetes

Kuberneteskode sumber tersedia secara gratis, jadi dengan peralatan Anda sendiri, Anda dapat menginstal dan mengelola Kubernetes sendiri. Namun, pengelolaan diri Kubernetes membutuhkan keahlian operasional yang mendalam dan membutuhkan waktu dan upaya untuk mempertahankannya. Karena alasan tersebut, kebanyakan orang yang menerapkan beban kerja produksi memilih penyedia cloud (seperti Amazon EKS) atau penyedia lokal (seperti Amazon EKS Anywhere) dengan Kubernetes distribusi dan dukungan ahli yang telah diuji sendiri. Kubernetes Ini memungkinkan Anda untuk menurunkan banyak angkat berat yang tidak berdiferensiasi yang diperlukan untuk mempertahankan cluster Anda, termasuk:

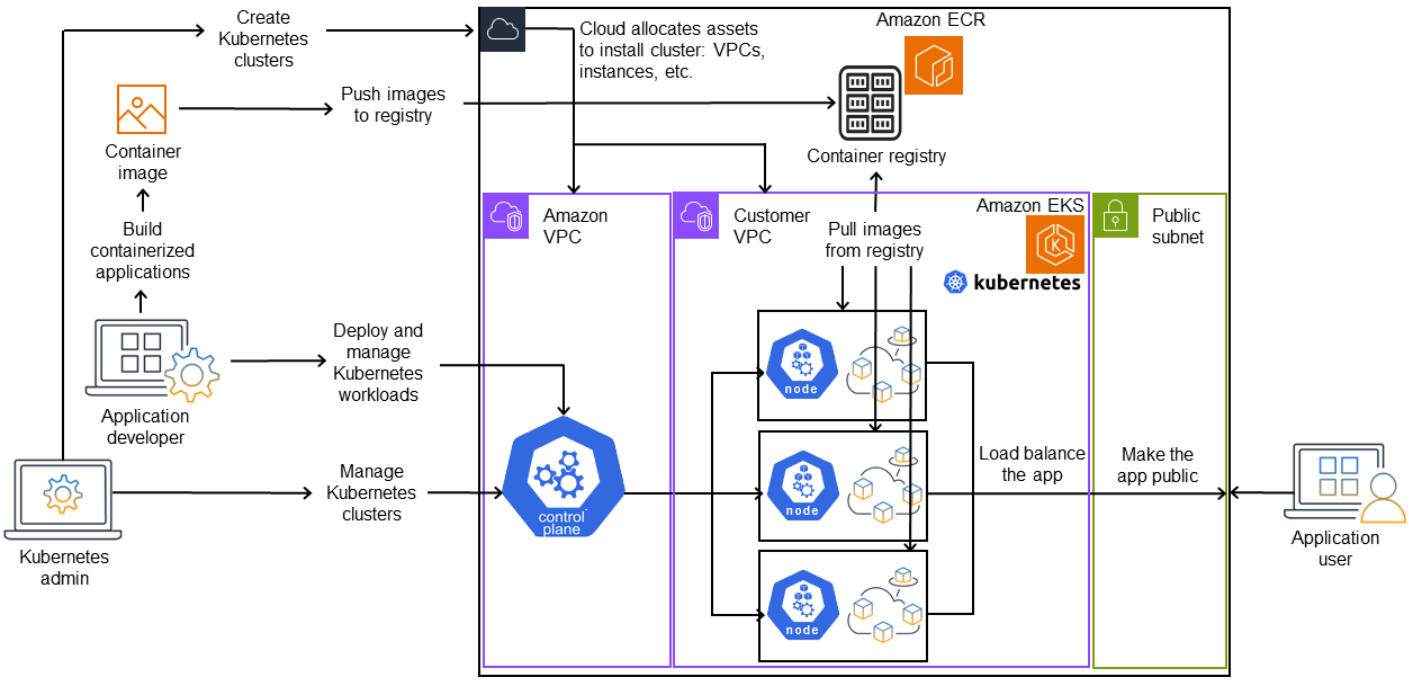
- Perangkat Keras - Jika Anda tidak memiliki perangkat keras yang tersedia untuk dijalankan Kubernetes sesuai kebutuhan Anda, penyedia cloud seperti AWS Amazon EKS dapat menghemat biaya di muka. Dengan Amazon EKS, ini berarti Anda dapat menggunakan sumber daya cloud terbaik yang ditawarkan oleh AWS, termasuk instans komputer (Amazon Elastic Compute Cloud), lingkungan pribadi Anda sendiri (Amazon VPC), identitas pusat dan manajemen izin (IAM), dan penyimpanan (Amazon EBS). AWS mengelola komputer, jaringan, pusat data, dan semua komponen fisik lainnya yang diperlukan untuk menjalankannyaKubernetes. Demikian juga, Anda

tidak perlu merencanakan pusat data Anda untuk menangani kapasitas maksimum pada hari-hari permintaan tertinggi Anda. Untuk Amazon EKS Kubernetes Anywhere, atau Kubernetes cluster lokal lainnya, Anda bertanggung jawab untuk mengelola infrastruktur yang digunakan dalam penerapan Anda, tetapi Anda masih dapat mengandalkan AWS untuk membantu Anda tetap Kubernetes up to date.

- Manajemen bidang kontrol - Amazon EKS mengelola keamanan dan ketersediaan bidang Kubernetes kontrol yang AWS di-host, yang bertanggung jawab untuk menjadwalkan kontainer, mengelola ketersediaan aplikasi, dan tugas-tugas utama lainnya, sehingga Anda dapat fokus pada beban kerja aplikasi Anda. Jika cluster Anda rusak, AWS harus memiliki sarana untuk memulihkan cluster Anda ke status berjalan. Untuk Amazon EKS Anywhere, Anda akan mengelola sendiri pesawat kontrol.
- Upgrade yang diuji - Saat Anda memutakhirkan kluster, Anda dapat mengandalkan Amazon EKS atau Amazon EKS Anywhere untuk menyediakan versi distribusi yang diuji. Kubernetes
- Add-on - Ada ratusan proyek yang dibangun untuk memperluas dan bekerja dengan Kubernetes yang dapat Anda tambahkan ke infrastruktur klaster Anda atau gunakan untuk membantu menjalankan beban kerja Anda. Alih-alih membangun dan mengelola sendiri add-on tersebut, AWS sediakan Add-on Amazon EKS yang dapat Anda gunakan dengan cluster Anda. Amazon EKS Anywhere menyediakan Paket Terkurasi yang mencakup pembuatan banyak proyek open source populer. Jadi Anda tidak perlu membangun perangkat lunak sendiri atau mengelola patch keamanan kritis, perbaikan bug, atau peningkatan. Demikian juga, jika default memenuhi kebutuhan Anda, biasanya konfigurasi yang sangat sedikit dari add-on tersebut diperlukan. Lihat Memperluas Cluster untuk detail tentang memperluas klaster Anda dengan add-on.

Kubernetesdalam tindakan

Diagram berikut menunjukkan aktivitas utama yang akan Anda lakukan sebagai Kubernetes Admin atau Pengembang Aplikasi untuk membuat dan menggunakan Kubernetes cluster. Dalam prosesnya, ini menggambarkan bagaimana Kubernetes komponen berinteraksi satu sama lain, menggunakan AWS cloud sebagai contoh penyedia cloud yang mendasarinya.



KubernetesAdmin membuat Kubernetes cluster menggunakan alat khusus untuk jenis penyedia tempat cluster akan dibangun. Contoh ini menggunakan AWS cloud sebagai penyedia, yang menawarkan Kubernetes layanan terkelola yang disebut Amazon EKS. Layanan terkelola secara otomatis mengalokasikan sumber daya yang diperlukan untuk membuat cluster, termasuk membuat dua Virtual Private Clouds (Amazon VPC) baru untuk cluster, menyiapkan jaringan, memetakan Kubernetes izin ke mereka untuk mengelola aset di cloud, melihat bahwa layanan bidang kontrol memiliki tempat untuk dijalankan, dan mengalokasikan nol atau lebih instans Amazon EC2 sebagai node untuk menjalankan beban kerja. Kubernetes AWS mengelola satu VPC Amazon itu sendiri untuk bidang kontrol, sedangkan VPC Amazon lainnya berisi node pelanggan yang menjalankan beban kerja.

Banyak tugas Kubernetes Admin ke depan dilakukan dengan menggunakan Kubernetes alat seperti kubectl. Alat itu membuat permintaan layanan langsung ke bidang kontrol cluster. Cara kueri dan perubahan dibuat ke cluster kemudian sangat mirip dengan cara Anda melakukannya di Kubernetes cluster mana pun.

Pengembang aplikasi yang ingin menyebarkan beban kerja ke cluster ini dapat melakukan beberapa tugas. Pengembang perlu membangun aplikasi menjadi satu atau lebih gambar kontainer, lalu

mendorong gambar tersebut ke registri kontainer yang dapat diakses oleh Kubernetes cluster. AWS menawarkan Amazon Elastic Container Registry (Amazon ECR) Registry ECR) untuk tujuan itu.

Untuk menjalankan aplikasi, pengembang dapat membuat file konfigurasi berformat YAML yang memberi tahu cluster cara menjalankan aplikasi, termasuk kontainer mana yang akan ditarik dari registri dan cara membungkus kontainer tersebut dalam Pod. Bidang kontrol (scheduler) menjadwalkan kontainer ke satu atau lebih node dan runtime kontainer pada setiap node benar-benar menarik dan menjalankan kontainer yang diperlukan. Pengembang juga dapat mengatur penyeimbang beban aplikasi untuk menyeimbangkan lalu lintas ke kontainer yang tersedia yang berjalan pada setiap node dan mengekspos aplikasi sehingga tersedia di jaringan publik ke dunia luar. Dengan semua itu dilakukan, seseorang yang ingin menggunakan aplikasi dapat terhubung ke titik akhir aplikasi untuk mengaksesnya.

Bagian berikut membahas detail masing-masing fitur ini, dari perspektif Kubernetes Cluster dan Beban Kerja.

Klaster

Jika tugas Anda adalah memulai dan mengelola Kubernetes cluster, Anda harus tahu bagaimana Kubernetes cluster dibuat, ditingkatkan, dikelola, dan dihapus. Anda juga harus tahu komponen apa yang membentuk cluster dan apa yang perlu Anda lakukan untuk mempertahankan komponen tersebut.

Alat untuk mengelola cluster menangani tumpang tindih antara Kubernetes layanan dan penyedia perangkat keras yang mendasarinya. Untuk alasan itu, otomatisasi tugas-tugas ini cenderung dilakukan oleh Kubernetes penyedia (seperti Amazon EKS atau Amazon EKS Anywhere) menggunakan alat yang khusus untuk penyedia. Misalnya, untuk memulai cluster Amazon EKS Anda dapat menggunakan`eksctl create cluster`, sedangkan untuk Amazon EKS Anywhere Anda dapat menggunakan`eksctl anywhere create cluster`. Perhatikan bahwa sementara perintah ini membuat Kubernetes cluster, mereka khusus untuk penyedia dan bukan bagian dari Kubernetes proyek itu sendiri.

Alat pembuatan dan manajemen cluster

KubernetesProyek ini menawarkan alat untuk membuat Kubernetes cluster secara manual. Jadi jika Anda ingin menginstal Kubernetes pada satu mesin, atau menjalankan bidang kontrol pada mesin dan menambahkan node secara manual, Anda dapat menggunakan alat CLI seperti kind, minikube, atau kubeadm yang tercantum di bawah Install Tools. Kubernetes Untuk menyederhanakan dan

mengotomatiskan siklus hidup penuh pembuatan dan pengelolaan klaster, jauh lebih mudah untuk menggunakan alat yang didukung oleh Kubernetes penyedia yang sudah mapan, seperti Amazon EKS atau Amazon EKS Anywhere.

Di AWS Cloud, Anda dapat membuat kluster Amazon EKS menggunakan alat CLI, seperti eksctl, atau alat deklaratif lainnya, seperti Terraform (lihat Amazon EKS Blueprints for Terraform). Anda juga dapat membuat cluster dari AWS Management Console. Lihat fitur Amazon EKS untuk daftar apa yang Anda dapatkan dengan Amazon EKS. Kubernetestanggung jawab yang diambil Amazon EKS untuk Anda meliputi:

- Bidang kontrol terkelola - AWS memastikan bahwa kluster Amazon EKS tersedia dan dapat diskalakan karena mengelola bidang kontrol untuk Anda dan membuatnya tersedia di seluruh AWS Availability Zone.
- Manajemen node - Alih-alih menambahkan node secara manual, Anda dapat meminta Amazon EKS membuat node secara otomatis sesuai kebutuhan, menggunakan Grup Node Terkelola atau Karpenter. Grup Node Terkelola memiliki integrasi dengan Kubernetes Cluster Autoscaling. Menggunakan alat manajemen node, Anda dapat memanfaatkan penghematan biaya, dengan hal-hal seperti Instans Spot dan konsolidasi node, dan ketersediaan, menggunakan fitur Penjadwalan untuk mengatur bagaimana beban kerja diterapkan dan node dipilih.
- Jaringan cluster - Menggunakan CloudFormation template, `eksctl` mengatur jaringan antara bidang kontrol dan komponen bidang data (node) di Kubernetes cluster. Ini juga mengatur titik akhir di mana komunikasi internal dan eksternal dapat berlangsung. Lihat De-mystifying jaringan cluster untuk node pekerja Amazon EKS untuk detailnya. Komunikasi antar Pod di Amazon EKS dilakukan dengan menggunakan Amazon EKS Pod Identities, yang menyediakan sarana untuk membiarkan Pod memanfaatkan metode AWS cloud untuk mengelola kredensyal dan izin.
- Add-Ons - Amazon EKS menyelamatkan Anda dari keharusan membangun dan menambahkan komponen perangkat lunak yang biasa digunakan untuk mendukung Kubernetes cluster. Misalnya, saat Anda membuat klaster Amazon EKS dari konsol AWS Manajemen, secara otomatis menambahkan kube-proxy Amazon EKS, plugin Amazon VPC CNI untuk,dan add-on CoreDNS. Kubernetes Lihat add-on Amazon EKS untuk informasi lebih lanjut tentang add-on ini, termasuk daftar yang tersedia.

Untuk menjalankan cluster Anda di komputer dan jaringan lokal Anda sendiri, Amazon menawarkan Amazon EKS Anywhere. Alih-alih AWS Cloud menjadi penyedia, Anda memiliki pilihan untuk menjalankan Amazon EKS Anywhere di VMware vSphere, bare metal (penyedia Tinkerbell), Snow CloudStack, atau platform Nutanix menggunakan peralatan Anda sendiri.

Amazon EKS Anywhere didasarkan pada perangkat lunak Amazon EKS Distro yang sama yang digunakan oleh Amazon EKS. Namun, Amazon EKS Anywhere bergantung pada implementasi antarmuka KubernetesCluster API (CAPI) yang berbeda untuk mengelola siklus hidup penuh mesin di cluster Amazon EKS Anywhere (seperti CAPV untuk vSphere dan CAPC for). CloudStack Karena seluruh cluster berjalan pada peralatan Anda, Anda mengambil tanggung jawab tambahan untuk mengelola bidang kontrol dan mencadangkan datanya (lihat etcd nanti di dokumen ini).

Komponen cluster

Kuberneteskomponen cluster dibagi menjadi dua bidang utama: bidang kontrol dan node pekerja. Control Plane Components mengelola cluster dan menyediakan akses ke API-nya. Node pekerja (kadang-kadang hanya disebut sebagai Nodes) menyediakan tempat di mana beban kerja yang sebenarnya dijalankan. Komponen Node terdiri dari layanan yang berjalan pada setiap node untuk berkomunikasi dengan bidang kontrol dan menjalankan kontainer. Kumpulan node pekerja untuk cluster Anda disebut sebagai Data Plane.

Bidang kontrol

Bidang kontrol terdiri dari satu set layanan yang mengelola cluster. Layanan ini semua mungkin berjalan pada satu komputer atau dapat tersebar di beberapa komputer. Secara internal, ini disebut sebagai Control Plane Instances (CPI). Bagaimana CPI dijalankan tergantung pada ukuran cluster dan persyaratan untuk ketersediaan tinggi. Seiring meningkatnya permintaan di klaster, layanan pesawat kontrol dapat menskalakan untuk menyediakan lebih banyak contoh layanan tersebut, dengan permintaan diseimbangkan beban di antara instans.

Tugas yang dilakukan komponen bidang Kubernetes kontrol meliputi:

- Berkomunikasi dengan komponen cluster (server API) - Server API (kube-apiserver) mengekspos Kubernetes API sehingga permintaan ke cluster dapat dibuat baik dari dalam maupun luar cluster. Dengan kata lain, permintaan untuk menambah atau mengubah objek klaster (Pod, Services, Nodes, dan sebagainya) dapat berasal dari perintah luar, seperti permintaan dari `kubectl` untuk menjalankan Pod. Demikian juga, permintaan dapat dibuat dari server API ke komponen dalam cluster, seperti query ke `kubelet` layanan untuk status Pod.
- Menyimpan data tentang cluster (etcd key value store) - Layanan etcd menyediakan peran penting untuk melacak keadaan cluster saat ini. Jika layanan etcd menjadi tidak dapat diakses, Anda tidak akan dapat memperbarui atau menanyakan status cluster, meskipun beban kerja akan terus berjalan untuk sementara waktu. Untuk alasan itu, cluster kritis biasanya memiliki beberapa instance yang seimbang beban dari layanan etcd yang berjalan pada satu waktu dan melakukan

pencadangan berkala dari penyimpanan nilai kunci etcd jika terjadi kehilangan atau kerusakan data. Ingatlah bahwa, di Amazon EKS, ini semua ditangani untuk Anda secara otomatis secara default. Amazon EKS Anywhere menyediakan instruksi untuk pencadangan dan pemulihan etcd. Lihat Model Data etcd untuk mempelajari cara etcd mengelola data.

- Schedule Pods to Nodes (Scheduler) - Permintaan untuk memulai atau menghentikan Pod di Kubernetes diarahkan ke KubernetesScheduler (kube-scheduler). Karena sebuah cluster dapat memiliki beberapa node yang mampu menjalankan Pod, maka terserah Scheduler untuk memilih node (atau node, dalam kasus replika) Pod mana yang harus dijalankan. Jika tidak ada kapasitas yang cukup untuk menjalankan Pod yang diminta pada node yang ada, permintaan akan gagal, kecuali Anda telah membuat ketentuan lain. Ketentuan tersebut dapat mencakup mengaktifkan layanan seperti Grup Node Terkelola atau Karpenter yang dapat secara otomatis memulai node baru untuk menangani beban kerja.
- Simpan komponen dalam keadaan yang diinginkan (Controller Manager) - Kubernetes Controller Manager berjalan sebagai proses daemon (kube-controller-manager) untuk mengawasi status cluster dan membuat perubahan pada cluster untuk membangun kembali status yang diharapkan. Secara khusus, ada beberapa pengontrol yang mengawasi Kubernetes objek yang berbeda, yang meliputi, statefulset-controller, endpoint-controller node-lifecycle-controller, cronjob-controller, dan lain-lain.
- Kelola sumber daya cloud (Cloud Controller Manager) - Interaksi antara Kubernetes dan penyedia cloud yang melakukan permintaan sumber daya pusat data yang mendasarinya ditangani oleh Cloud Controller Manager (cloud-controller-manager). Pengontrol yang dikelola oleh Cloud Controller Manager dapat mencakup pengontrol rute (untuk menyiapkan rute jaringan cloud), pengontrol layanan (untuk menggunakan layanan penyeimbangan beban cloud), dan pengontrol node (untuk menggunakan API cloud agar Kubernetes node tetap sinkron dengan node cloud).

Node Pekerja (bidang data)

Untuk Kubernetes cluster simpul tunggal, beban kerja berjalan pada mesin yang sama dengan bidang kontrol. Namun, konfigurasi yang lebih normal adalah memiliki satu atau lebih sistem komputer terpisah (Node) yang didedikasikan untuk menjalankan Kubernetes beban kerja.

Saat pertama kali membuat Kubernetes cluster, beberapa alat pembuatan cluster memungkinkan Anda mengonfigurasi sejumlah node tertentu yang akan ditambahkan ke cluster (baik dengan mengidentifikasi sistem komputer yang ada atau dengan meminta penyedia membuat yang baru). Sebelum beban kerja ditambahkan ke sistem tersebut, layanan ditambahkan ke setiap node untuk mengimplementasikan fitur-fitur ini:

- Manage each node (kubelet) - Server API berkomunikasi dengan layanan kubelet yang berjalan pada setiap node untuk memastikan bahwa node terdaftar dengan benar dan Pod yang diminta oleh Scheduler sedang berjalan. Kubelet dapat membaca manifes Pod dan mengatur volume penyimpanan atau fitur lain yang dibutuhkan oleh Pod pada sistem lokal. Ini juga dapat memeriksa kesehatan wadah yang berjalan secara lokal.
- Jalankan kontainer pada sebuah node (container runtime) - Container Runtime pada setiap node mengelola kontainer yang diminta untuk setiap Pod yang ditetapkan ke node. Itu berarti ia dapat menarik gambar kontainer dari registri yang sesuai, menjalankan penampung, menghentikannya, dan menanggapi pertanyaan tentang wadah. Runtime kontainer default adalah containerd. Pada Kubernetes 1,24, integrasi khusus Docker (Dockershim) yang dapat digunakan sebagai runtime kontainer dibatalkan dari. Kubernetes Meskipun Anda masih dapat menggunakan Docker untuk menguji dan menjalankan kontainer di sistem lokal Kubernetes Anda, untuk menggunakannya Docker sekarang Anda harus Menginstal Docker Mesin pada setiap node untuk menggunakannyaKubernetes.
- Mengelola jaringan antar kontainer (kube-proxy) - Untuk dapat mendukung komunikasi antar Pod menggunakan Layanan, Kubernetes diperlukan cara untuk menyiapkan jaringan Pod untuk melacak alamat IP dan port yang terkait dengan Pod tersebut. Layanan kube-proxy berjalan pada setiap node untuk memungkinkan komunikasi antar Pod berlangsung.

Perluas Cluster

Ada beberapa layanan yang dapat Anda tambahkan Kubernetes untuk mendukung cluster, tetapi tidak dijalankan di bidang kontrol. Layanan ini sering berjalan langsung pada node di namespace sistem kube atau di namespace sendiri (seperti yang sering dilakukan dengan penyedia layanan pihak ketiga). Contoh umum adalah layanan CoreDNS, yang menyediakan layanan DNS ke cluster. Lihat Menemukan layanan bawaan untuk informasi tentang cara melihat layanan klaster mana yang berjalan di kube-system di klaster Anda.

Ada berbagai jenis add-on yang dapat Anda pertimbangkan untuk ditambahkan ke cluster Anda. Agar klaster tetap sehat, Anda dapat menambahkan fitur observabilitas yang memungkinkan Anda melakukan hal-hal seperti pencatatan, audit, dan metrik. Dengan informasi ini, Anda dapat memecahkan masalah yang terjadi, seringkali melalui antarmuka observabilitas yang sama. Contoh dari jenis layanan ini termasuk Amazon GuardDuty,, AWS Distro untuk CloudWatch, Amazon VPC CNI plugin untuk OpenTelemetryKubernetes, dan Grafana Monitoring. Kubernetes Untuk penyimpanan, add-on ke Amazon EKS termasuk Amazon Elastic Block Store CSI Driver (untuk menambahkan perangkat penyimpanan blok), Amazon Elastic File System CSI Driver (untuk

menambahkan penyimpanan sistem file), dan beberapa add-on penyimpanan pihak ketiga (seperti Amazon FSx untuk driver ONTAP CSI). NetApp

Untuk daftar pengaya Amazon EKS yang lebih lengkap, lihat add-on Amazon EKS.

Beban kerja

Kubernetesmendefinisikan Beban Kerja sebagai “aplikasi yang berjalan diKubernetes.” Aplikasi tersebut dapat terdiri dari serangkaian layanan mikro yang dijalankan sebagai Container di Pod, atau dapat dijalankan sebagai pekerjaan batch atau jenis aplikasi lainnya. KubernetesTugasnya adalah memastikan bahwa permintaan yang Anda buat untuk objek yang akan disiapkan atau dikerahkan dilakukan. Sebagai seseorang yang menerapkan aplikasi, Anda harus mempelajari tentang bagaimana kontainer dibuat, bagaimana Pod didefinisikan, dan metode apa yang dapat Anda gunakan untuk menerapkannya.

Kontainer

Elemen paling dasar dari beban kerja aplikasi yang Anda gunakan dan kelola Kubernetes adalah Pod. Sebuah Pod mewakili cara memegang komponen aplikasi serta mendefinisikan spesifikasi yang menggambarkan atribut Pod. Bandingkan ini dengan sesuatu seperti paket RPM atau Deb, yang mengemas perangkat lunak bersama untuk sistem Linux, tetapi tidak berjalan sendiri sebagai entitas.

Karena Pod adalah unit deployable terkecil, biasanya memiliki satu kontainer. Namun, beberapa kontainer dapat berada di dalam Pod jika kontainer digabungkan dengan erat. Misalnya, sebuah kontainer server web mungkin dikemas dalam Pod dengan jenis kontainer sidecar yang dapat menyediakan logging, monitoring, atau layanan lain yang terkait erat dengan kontainer server web. Dalam hal ini, berada di Pod yang sama memastikan bahwa untuk setiap instance Pod yang sedang berjalan, kedua container selalu berjalan pada node yang sama. Demikian juga, semua kontainer dalam sebuah Pod berbagi lingkungan yang sama, dengan kontainer dalam sebuah Pod berjalan seolah-olah mereka berada di host terisolasi yang sama. Efek dari hal ini adalah bahwa kontainer berbagi satu alamat IP yang menyediakan akses ke Pod dan kontainer dapat berkomunikasi satu sama lain seolah-olah mereka berjalan di localhost mereka sendiri.

Spesifikasi Pod (PodSpec) menentukan status Pod yang diinginkan. Anda dapat menerapkan Pod individual atau beberapa Pod dengan menggunakan sumber daya beban kerja untuk mengelola Template Pod. Sumber daya beban kerja mencakup Deployments (untuk mengelola beberapa Replika Pod), StatefulSets(untuk menyebarkan Pod yang perlu unik, seperti Pod database), dan DaemonSets(di mana sebuah Pod perlu dijalankan secara terus menerus di setiap node). Lebih lanjut tentang itu nanti.

Sementara Pod adalah unit terkecil yang Anda gunakan, sebuah kontainer adalah unit terkecil yang Anda buat dan kelola.

Kontainer Bangunan

Pod sebenarnya hanya sebuah struktur di sekitar satu atau lebih kontainer, dengan setiap kontainer itu sendiri memegang sistem file, executable, file konfigurasi, pustaka, dan komponen lainnya untuk benar-benar menjalankan aplikasi. Karena sebuah perusahaan bernama Docker Inc. pertama kali mempopulerkan kontainer, beberapa orang menyebut kontainer sebagai Kontainer. Docker Namun, Open Container Initiative telah menetapkan runtime kontainer, gambar, dan metode distribusi untuk industri. Ditambah fakta bahwa kontainer dibuat dari banyak fitur Linux yang ada, yang lain sering menyebut kontainer sebagai Kontainer OCI, Kontainer Linux, atau hanya Kontainer.

Saat Anda membangun wadah, Anda biasanya memulai dengan Docker file (secara harfiah dinamai itu). Di dalam Dockerfile itu, Anda mengidentifikasi:

- Gambar dasar - Gambar wadah dasar adalah wadah yang biasanya dibangun dari versi minimal sistem file sistem operasi (seperti Red Hat Enterprise Linux atau Ubuntu) atau sistem minimal yang ditingkatkan untuk menyediakan perangkat lunak untuk menjalankan jenis aplikasi tertentu (seperti aplikasi nodejs atau python).
- Perangkat lunak aplikasi - Anda dapat menambahkan perangkat lunak aplikasi Anda ke wadah Anda dengan cara yang sama seperti Anda menambahkannya ke sistem Linux. Misalnya, di Dockerfile Anda, Anda dapat menjalankan `npm` dan `yarn` menginstal aplikasi Java atau `yum` dan `dnf` untuk menginstal paket RPM. Dengan kata lain, menggunakan perintah RUN di Dockerfile, Anda dapat menjalankan perintah apa pun yang tersedia di sistem file gambar dasar Anda untuk menginstal perangkat lunak atau mengkonfigurasi perangkat lunak di dalam gambar kontainer yang dihasilkan.
- Instruksi - Referensi Dockerfile menjelaskan instruksi yang dapat Anda tambahkan ke Dockerfile saat Anda mengonfigurasinya. Ini termasuk instruksi yang digunakan untuk membangun apa yang ada di wadah itu sendiri (`ADD`atau `COPY` file dari sistem lokal), mengidentifikasi perintah untuk dijalankan ketika wadah dijalankan (`CMD`atau`ENTRYPOINT`), dan menghubungkan wadah ke sistem yang dijalankannya (dengan mengidentifikasi `USER` untuk dijalankan sebagai, lokal `VOLUME` untuk dipasang, atau port ke`EXPOSE`).

Sementara `docker` perintah dan layanan secara tradisional telah digunakan untuk membangun container (`docker build`), alat lain yang tersedia untuk membangun gambar kontainer termasuk podman dan nerdctl. Lihat Membangun Gambar Kontainer yang Lebih Baik atau Bangun dengan Docker untuk mempelajari tentang membangun kontainer.

Menyimpan Wadah

Setelah Anda membangun gambar kontainer Anda, Anda dapat menyimpannya dalam registri distribusi kontainer di workstation Anda atau di registri kontainer publik. Menjalankan registri kontainer pribadi di workstation Anda memungkinkan Anda menyimpan gambar kontainer secara lokal, membuatnya tersedia untuk Anda.

Untuk menyimpan gambar kontainer dengan cara yang lebih umum, Anda dapat mendorongnya ke registri kontainer publik. Registries kontainer publik menyediakan lokasi sentral untuk menyimpan dan mendistribusikan gambar kontainer. Contoh pendaftar kontainer publik termasuk Amazon Elastic Container Registry, registriRed Hat Quay, dan Docker registri Hub.

Saat menjalankan beban kerja kontainer di Amazon Elastic Kubernetes Service (Amazon EKS), kami sarankan untuk Docker menarik salinan Gambar Resmi yang disimpan di Amazon Elastic Container Registry. AWS Amazon ECR telah menyimpan gambar-gambar ini sejak 2021. Anda dapat mencari gambar kontainer populer di Galeri Publik Amazon ECR, dan khusus untuk gambar Docker Hub, Anda dapat mencari Galeri Amazon ECR Docker.

Menjalankan wadah

Karena kontainer dibangun dalam format standar, kontainer dapat berjalan pada mesin apa pun yang dapat menjalankan runtime kontainer (sepertiDocker) dan yang isinya cocok dengan arsitektur mesin lokal (seperti `x86_64` atau`arm`). Untuk menguji kontainer atau menjalankannya di desktop lokal Anda, Anda dapat menggunakan `docker run` atau `podman run` perintah untuk memulai penampung di localhost. NamunKubernetes, untuk setiap node pekerja memiliki runtime kontainer yang diterapkan dan terserah Kubernetes untuk meminta node menjalankan wadah.

Setelah wadah ditugaskan untuk berjalan pada node, node akan melihat apakah versi yang diminta dari gambar kontainer sudah ada di node. Jika tidak, Kubernetes beri tahu runtime kontainer untuk menarik kontainer itu dari registri kontainer yang sesuai, lalu jalankan kontainer itu secara lokal. Perlu diingat bahwa gambar kontainer mengacu pada paket perangkat lunak yang dipindahkan antara laptop Anda, registri kontainer, dan Kubernetes node. Sebuah wadah mengacu pada instance yang sedang berjalan dari gambar itu.

Pod

Setelah kontainer Anda siap, bekerja dengan Pod termasuk mengonfigurasi, menerapkan, dan membuat Pod dapat diakses.

Mengkonfigurasi Pod

Ketika Anda mendefinisikan sebuah Pod, Anda menetapkan satu set atribut untuk itu. Atribut tersebut harus menyertakan setidaknya nama Pod dan image container yang akan dijalankan. Namun, ada banyak hal lain yang ingin Anda konfigurasikan dengan definisi Pod Anda juga (lihat PodSpechalaman untuk detail tentang apa yang bisa masuk ke Pod). Ini termasuk:

- Penyimpanan - Ketika kontainer yang sedang berjalan dihentikan dan dihapus, penyimpanan data dalam wadah itu akan hilang, kecuali jika Anda mengatur penyimpanan yang lebih permanen. Kubernetesmendukung banyak jenis penyimpanan yang berbeda dan mengabstraksinya di bawah payung Volume. Jenis penyimpanan termasuk CephFS, NFS, iSCSI, dan lainnya. Anda bahkan dapat menggunakan perangkat blok lokal dari komputer lokal. Dengan salah satu jenis penyimpanan yang tersedia dari cluster Anda, Anda dapat memasang volume penyimpanan ke titik pemasangan yang dipilih di sistem file container Anda. Volume Persisten adalah salah satu yang terus ada setelah Pod dihapus, sementara Volume Ephemeral dihapus ketika Pod dihapus. Jika administrator klaster Anda membuat yang berbeda StorageClassesuntuk klaster Anda, Anda mungkin memiliki opsi untuk memilih atribut penyimpanan yang Anda gunakan, seperti apakah volume dihapus atau direklamasi setelah digunakan, apakah itu akan diperluas jika lebih banyak ruang diperlukan, dan bahkan apakah itu memenuhi persyaratan kinerja tertentu.
- Secrets - Dengan membuat Secrets tersedia untuk container dalam spesifikasi Pod, Anda dapat memberikan izin yang dibutuhkan container tersebut untuk mengakses sistem file, basis data, atau aset lain yang dilindungi. Kunci, kata sandi, dan token adalah salah satu item yang dapat disimpan sebagai rahasia. Menggunakan rahasia membuatnya sehingga Anda tidak perlu menyimpan informasi ini dalam gambar kontainer, tetapi hanya perlu membuat rahasia tersedia untuk menjalankan wadah. Mirip dengan Rahasia adalah ConfigMaps. A ConfigMap cenderung menyimpan informasi yang kurang penting, seperti pasangan nilai kunci untuk mengonfigurasi layanan.
- Sumber daya kontainer - Objek untuk kontainer konfigurasi lebih lanjut dapat mengambil bentuk konfigurasi sumber daya. Untuk setiap kontainer, Anda dapat meminta jumlah memori dan CPU yang dapat digunakan, serta menempatkan batas dari jumlah total sumber daya yang dapat digunakan wadah. Lihat Manajemen Sumber Daya untuk Pod dan Container untuk contoh.
- Gangguan - Pod dapat terganggu secara tidak sengaja (node turun) atau secara sukarela (upgrade diinginkan). Dengan mengonfigurasi anggaran gangguan Pod, Anda dapat mengontrol seberapa tersedia aplikasi Anda saat terjadi gangguan. Lihat Menentukan Anggaran Gangguan untuk aplikasi Anda untuk contoh.
- Ruang nama - Kubernetes menyediakan berbagai cara untuk mengisolasi Kubernetes komponen dan beban kerja satu sama lain. Menjalankan semua Pod untuk aplikasi tertentu di Namespace

yang sama adalah cara umum untuk mengamankan dan mengelola Pod tersebut bersama-sama. Anda dapat membuat ruang nama Anda sendiri untuk digunakan atau memilih untuk tidak menunjukkan namespace (yang menyebabkan Kubernetes penggunaan namespace). `default` Kuberneteskomponen control plane biasanya berjalan di namespace kube-system.

Konfigurasi yang baru saja dijelaskan biasanya dikumpulkan bersama dalam file YAMM untuk diterapkan ke Kubernetes cluster. Untuk Kubernetes kluster pribadi, Anda mungkin hanya menyimpan file YAMAL ini di sistem lokal Anda. Namun, dengan klaster dan beban kerja yang lebih kritis, GitOpsadalah cara populer untuk mengotomatiskan penyimpanan dan pembaruan untuk sumber daya beban kerja dan infrastruktur. Kubernetes

Objek yang digunakan untuk mengumpulkan dan menyebarkan informasi Pod ditentukan oleh salah satu metode penerapan berikut.

Menerapkan Pod

Metode yang akan Anda pilih untuk menerapkan Pod tergantung pada jenis aplikasi yang Anda rencanakan untuk dijalankan dengan Pod tersebut. Berikut adalah beberapa pilihan Anda:

- Aplikasi stateless - Aplikasi stateless tidak menyimpan data sesi klien, jadi sesi lain tidak perlu merujuk kembali ke apa yang terjadi pada sesi sebelumnya. Ini membuat lebih mudah untuk mengganti Pod dengan yang baru jika menjadi tidak sehat atau memindahkannya tanpa menyimpan status. Jika Anda menjalankan aplikasi stateless (seperti server web), Anda dapat menggunakan Deployment untuk menyebarkan Pod dan. ReplicaSets A ReplicaSet mendefinisikan berapa banyak instance dari sebuah Pod yang ingin Anda jalankan secara bersamaan. Meskipun Anda dapat menjalankan ReplicaSet secara langsung, adalah umum untuk menjalankan replika secara langsung di dalam Deployment, untuk menentukan berapa banyak replika Pod yang harus dijalankan pada satu waktu.
- Aplikasi stateful - Aplikasi stateful adalah aplikasi di mana identitas Pod dan urutan peluncuran Pod adalah penting. Aplikasi ini membutuhkan penyimpanan persisten yang stabil dan perlu digunakan dan diskalakan secara konsisten. Untuk menerapkan aplikasi stateful diKubernetes, Anda dapat menggunakan. StatefulSets Contoh aplikasi yang biasanya StatefulSet dijalankan sebagai database. Dalam a StatefulSet, Anda dapat menentukan replika, Pod dan kontainernya, volume penyimpanan yang akan dipasang, dan lokasi dalam wadah tempat data disimpan. Lihat Menjalankan Aplikasi Stateful yang Direplikasi untuk contoh database yang digunakan sebagai file. ReplicaSet
- Aplikasi per-node - Ada kalanya Anda ingin menjalankan aplikasi pada setiap node di cluster AndaKubernetes. Misalnya, pusat data Anda mungkin mengharuskan setiap komputer

menjalankan aplikasi pemantauan atau layanan akses jarak jauh tertentu. UntukKubernetes, Anda dapat menggunakan a DaemonSetuntuk memastikan bahwa aplikasi yang dipilih berjalan pada setiap node di cluster Anda.

- Aplikasi berjalan hingga selesai - Ada beberapa aplikasi yang ingin Anda jalankan untuk menyelesaikan tugas tertentu. Ini bisa termasuk yang menjalankan laporan status bulanan atau membersihkan data lama. Objek Job dapat digunakan untuk menyiapkan aplikasi untuk memulai dan menjalankan, lalu keluar ketika tugas selesai. CronJobObjek memungkinkan Anda mengatur aplikasi untuk berjalan pada jam, menit, hari tertentu dalam sebulan, bulan, atau hari dalam seminggu, menggunakan struktur yang ditentukan oleh format crontab Linux.

Membuat aplikasi dapat diakses dari jaringan

Dengan aplikasi yang sering digunakan sebagai satu set layanan mikro yang berpindah ke tempat yang berbeda, Kubernetes diperlukan cara bagi layanan mikro tersebut untuk dapat menemukan satu sama lain. Selain itu, bagi orang lain untuk mengakses aplikasi di luar Kubernetes cluster, Kubernetes diperlukan cara untuk mengekspos aplikasi itu di alamat dan port luar. Fitur terkait jaringan ini dilakukan dengan objek Service dan Ingress, masing-masing:

- Services - Karena Pod dapat berpindah ke node dan alamat yang berbeda, Pod lain yang diperlukan untuk berkomunikasi dengan Pod pertama dapat merasa sulit untuk menemukan di mana Pod itu berada. Untuk mengatasi masalah ini, Kubernetes memungkinkan Anda mewakili aplikasi sebagai Layanan. Dengan Service, Anda dapat mengidentifikasi Pod atau kumpulan Pod dengan nama tertentu, kemudian menunjukkan port apa yang mengekspos layanan aplikasi tersebut dari Pod dan port apa yang dapat digunakan aplikasi lain untuk menghubungi layanan tersebut. Pod lain di dalam klaster dapat dengan mudah meminta Service dengan nama dan Kubernetes akan mengarahkan permintaan tersebut ke port yang tepat untuk sebuah instance dari Pod yang menjalankan layanan tersebut.
- Ingress - Ingress adalah apa yang dapat membuat aplikasi diwakili oleh Kubernetes Layanan tersedia untuk klien yang berada di luar cluster. Fitur dasar Ingress termasuk penyeimbang beban (dikelola oleh Ingress), pengontrol Ingress, dan aturan untuk permintaan perutean dari pengontrol ke Layanan. Ada beberapa Ingress Controller yang dapat Anda pilih. Kubernetes

Langkah selanjutnya

Memahami Kubernetes konsep dasar dan bagaimana kaitannya dengan Amazon EKS akan membantu Anda menavigasi dokumentasi dan Kubernetesdokumentasi Amazon EKS untuk

menemukan informasi yang Anda perlukan untuk mengelola kluster Amazon EKS dan menyebarkan beban kerja ke klaster tersebut. Untuk mulai menggunakan Amazon EKS, pilih dari yang berikut ini:

- Buat cluster sederhana
- Buat cluster yang lebih kompleks
- Menyebarkan aplikasi sampel
- Jelajahi cara mengelola klaster Anda

Opsi deployment

Anda dapat menerapkan Amazon EKS menggunakan salah satu opsi berikut:

Amazon EKS di cloud

Anda dapat berjalan Kubernetes di AWS cloud tanpa perlu menginstal, mengoperasikan, dan memelihara bidang Kubernetes kontrol atau node Anda sendiri. Opsi ini adalah apa yang tercakup dalam panduan ini.

Amazon EKS di Outposts

AWS Outpostsmengaktifkan model asliLayanan AWS, infrastruktur, dan operasi di fasilitas lokal Anda. Dengan Amazon EKS di Outposts, Anda dapat memilih untuk menjalankan cluster yang diperluas atau lokal. Dengan cluster yang diperluas, bidang Kubernetes kontrol berjalan dalamWilayah AWS, dan node berjalan di Outposts. Dengan cluster lokal, seluruh Kubernetes cluster berjalan secara lokal di Outposts, termasuk bidang Kubernetes kontrol dan node. Untuk informasi selengkapnya, lihat Amazon EKS pada AWS Outposts.

Amazon EKS Anywhere

Amazon EKS Anywhere adalah opsi penerapan untuk Amazon EKS yang memungkinkan Anda membuat dan mengoperasikan Kubernetes cluster lokal dengan mudah. Baik Amazon EKS dan Amazon EKS Anywhere dibangun di Amazon EKS Distro. Untuk mempelajari lebih lanjut tentang Amazon EKS Anywhere, dan perbedaannya dengan Amazon EKS, lihat Ikhtisar dan Membandingkan Amazon EKS Anywhere dengan Amazon EKS di dokumentasi Amazon EKS Anywhere. Untuk jawaban atas beberapa pertanyaan umum, lihat FAQ Amazon EKS Anywhere.

Distro Amazon EKS

Amazon EKS Distro adalah distribusi Kubernetes perangkat lunak sumber terbuka dan dependensi yang sama yang digunakan oleh Amazon EKS di cloud. Amazon EKS Distro mengikuti siklus rilis Kubernetes versi yang sama dengan Amazon EKS dan disediakan sebagai

proyek sumber terbuka. Untuk mempelajari lebih lanjut, lihat Amazon EKS Distro. Anda juga dapat melihat dan mengunduh kode sumber untuk Amazon EKS Distro diGitHub.

Saat memilih opsi penerapan mana yang akan digunakan untuk Kubernetes klaster Anda, pertimbangkan hal berikut:

Fitur	Amazon EKS	Amazon EKS di Outposts	Amazon EKS Anywhere	Distro Amazon EKS
Perangkat keras	AWS-disediakan	AWS-disediakan	Disediakan oleh Anda	Disediakan oleh Anda
Lokasi penyebaran	AWSawan	Pusat data Anda	Pusat data Anda	Pusat data Anda
Kuberneteslokasi pesawat kontrol	AWSawan	AWScloud atau pusat data Anda	Pusat data Anda	Pusat data Anda
Kuberneteslokasi pesawat data	AWSawan	Pusat data Anda	Pusat data Anda	Pusat data Anda
Dukungan	AWS Support	AWS Support	AWS Support	Dukungan komunitas OSS

Menyiapkan untuk menggunakan Amazon EKS

AWSsumber daya biasanya memiliki batasan akses yang membatasi akses ke AWS entitas yang membuatnya. Oleh karena itu, sangat penting untuk menetapkan konfigurasi pengguna yang AWS Command Line Interface tepat sejak awal. Selain itu, Anda perlu melengkapi mesin lokal Anda dengan alat penting untuk manajemen baris perintah yang efisien dari cluster Amazon EKS Anda. Topik ini akan membantu Anda mempersiapkan manajemen baris perintah cluster Anda.

Langkah 1: Siapkan AWS CLI

AWS CLIIni adalah alat baris perintah untuk bekerja dengan AWS layanan, termasuk Amazon EKS. Ini juga digunakan untuk mengautentikasi pengguna IAM atau peran untuk akses ke kluster Amazon EKS dan AWS sumber daya lainnya dari mesin lokal Anda. Untuk menyediakan sumber daya AWS dari baris perintah, Anda perlu mendapatkan ID kunci AWS akses dan kunci rahasia untuk digunakan di baris perintah. Maka Anda perlu mengkonfigurasi kredensional ini di. AWS CLI Jika Anda belum menginstalAWS CLI, lihat Menginstal atau memperbarui versi terbaru dari AWS CLI Panduan AWS Command Line Interface Pengguna.

Untuk membuat kunci akses

1. Masuk ke AWS Management Console.
2. Di kanan atas, pilih nama AWS pengguna Anda untuk membuka menu navigasi. Misalnya, pilih **`webadmin`**. Kemudian pilih Security credentials.
3. Di bawah tombol Access, pilih Create Access Key.
4. Pilih Command Line Interface (CLI), lalu pilih Berikutnya.
5. Pilih Buat access key.
6. Pilih Unduh file.csv.

Untuk mengkonfigurasi AWS CLI

Setelah menginstalAWS CLI, lakukan langkah-langkah berikut untuk mengkonfigurasinya. Untuk informasi selengkapnya, lihat Mengkonfigurasi AWS CLI dalam Panduan AWS Command Line Interface Pengguna.

1. Di jendela terminal, masukkan perintah berikut:

```
aws configure
```

Secara opsional, Anda dapat mengonfigurasi profil bernama, seperti**`--profile cluster-admin`**. Jika Anda mengonfigurasi profil bernama diAWS CLI, Anda harus selalu meneruskan bendera ini di perintah berikutnya.

2. Masukkan AWS kredensi Anda. Sebagai contoh:

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

Untuk mendapatkan token keamanan

Jika diperlukan, jalankan perintah berikut untuk mendapatkan token keamanan baru untukAWS CLI. Untuk informasi selengkapnya, lihat `get-session-token` dalam AWS CLI Referensi Perintah.

Secara default, token berlaku selama 15 menit. Untuk mengubah batas waktu sesi default, berikan **`--duration-seconds`** tanda. Sebagai contoh:

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

Perintah ini mengembalikan kredensi keamanan sementara untuk sesi. AWS CLI Anda akan melihat output respons berikut:

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

Untuk memverifikasi identitas pengguna

Jika diperlukan, jalankan perintah berikut untuk memverifikasi AWS kredensional identitas pengguna IAM Anda (seperti*`ClusterAdmin`*) untuk sesi terminal.

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

Perintah ini mengembalikan Amazon Resource Name (ARN) dari entitas IAM yang dikonfigurasi untuk. AWS CLI Anda akan melihat contoh keluaran respons berikut:

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

Langkah 2: Instal Kubernetes alat

Untuk berkomunikasi dengan Kubernetes cluster, Anda memerlukan alat untuk berinteraksi dengan Kubernetes API. Selain itu, Anda memerlukan beberapa alat lain, seperti alat untuk mengelola Kubernetes lingkungan di mesin lokal Anda.

Untuk membuat sumber daya AWS

- Sumber daya kluster Amazon EKS — Jika Anda baruAWS, sebaiknya instal `eksctl`. `eksctl`adalah utilitas infrastruktur sebagai kode (IAc) yang digunakan AWS CloudFormation untuk membuat cluster Amazon EKS Anda dengan mudah. Ini juga menciptakan Kubernetes sumber daya tambahan, seperti akun layanan. Untuk petunjuk tentang cara menginstal`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.
- AWSsumber daya - Jika Anda terbiasa mengotomatiskan penyediaan dan penyebaran AWS infrastruktur Anda, sebaiknya instal. Terraform Terraformadalah alat infrastruktur sumber terbuka sebagai kode (IAc) yang dikembangkan oleh. HashiCorp Ini memungkinkan Anda untuk menentukan dan menyediakan infrastruktur menggunakan bahasa konfigurasi tingkat tinggi seperti HashiCorp Configuration Language (HCL) atau JSON. Untuk petunjuk tentang cara menginstalTerraform, lihat Menginstal Terraform di Terraform dokumentasi.

Untuk menginstal **`kubectl`**

`kubectl`adalah alat baris perintah open source yang digunakan untuk berkomunikasi dengan server Kubernetes API di cluster Amazon EKS Anda. Jika Anda belum menginstalnya di mesin lokal Anda, pilih dari opsi berikut.

- AWSversi — Untuk menginstal versi Amazon EKS yang didukung, `kubectl` lihat. Menginstal atau memperbarui `kubectl`
- Versi komunitas — Untuk menginstal versi komunitas terbaru`kubectl`, lihat halaman Instal alat dalam Kubernetes dokumentasi.

Untuk mengatur lingkungan pengembangan

- Alat penyebaran lokal - Jika Anda baruKubernetes, pertimbangkan untuk menginstal alat penyebaran lokal seperti `minikube`atau. `kind` Alat-alat ini memungkinkan Anda mengelola klaster Amazon EKS di mesin lokal Anda.
- Package manager — Helmadalah manajer paket populer Kubernetes yang menyederhanakan instalasi dan pengelolaan paket yang kompleks. DenganHelm, lebih mudah untuk menginstal dan mengelola paket seperti AWS Load Balancer Controller di cluster Amazon EKS Anda.

Langkah selanjutnya

- Memulai dengan Amazon EKS

Menginstal atau memperbarui **`kubectl`**

`Kubectl`adalah alat baris perintah yang Anda gunakan untuk berkomunikasi dengan server Kubernetes API. `kubectl`Biner tersedia di banyak manajer paket sistem operasi. Menggunakan manajer paket untuk instalasi Anda seringkali lebih mudah daripada proses pengunduhan dan penginstalan manual.

Topik ini membantu Anda mengunduh dan menginstal, atau memperbarui, `kubectl` biner di perangkat Anda. Biner identik dengan versi komunitas hulu. Biner ini tidak unik untuk Amazon EKS atauAWS.

> **Note**
>
> Anda harus menggunakan versi `kubectl` yang tidak lebih dari satu perbedaan kecil dari bidang kendali Amazon EKS klaster Anda. Misalnya, `1.28` `kubectl` klien bekerja dengan Kubernetes`1.27`,`1.28`, dan `1.29` cluster.

Untuk menginstal atau memperbarui**kubectl**

1. Tentukan apakah Anda sudah kubectl menginstal pada perangkat Anda.

   ```
   kubectl version --client
   ```

   Jika Anda telah kubectl menginstal di jalur perangkat Anda, contoh output mencakup informasi yang mirip dengan berikut ini. Jika Anda ingin memperbarui versi yang saat ini telah Anda instal dengan versi yang lebih baru, selesaikan langkah berikutnya, pastikan untuk menginstal versi baru di lokasi yang sama dengan versi Anda saat ini.

   ```
   Client Version: v1.29.X-eks-1234567
   ```

   Jika Anda tidak menerima output, maka Anda belum kubectl menginstal, atau tidak diinstal di lokasi yang ada di jalur perangkat Anda.

2. Instal atau kubectl perbaruimacOS,Linux, dan sistem Windows operasi.

   macOS

   Untuk menginstal atau memperbarui **kubectlmacOS**

   1. Unduh biner untuk Kubernetes versi cluster Anda dari Amazon S3.

      - Kubernetes 1.29

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.0/2024-01-04/bin/darwin/amd64/kubectl
        ```

      - Kubernetes 1.28

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.5/2024-01-04/bin/darwin/amd64/kubectl
        ```

      - Kubernetes 1.27

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.9/2024-01-04/bin/darwin/amd64/kubectl
        ```

      - Kubernetes 1.26

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.12/2024-01-04/bin/darwin/amd64/kubectl
```

- Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-01-04/bin/darwin/amd64/kubectl
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-01-04/bin/darwin/amd64/kubectl
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-01-04/bin/darwin/amd64/kubectl
```

2. (Opsional) Verifikasi biner yang diunduh dengan `SHA-256` checksum untuk biner Anda.

   a. Unduh `SHA-256` checksum untuk Kubernetes versi cluster Anda.

   - Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.0/2024-01-04/bin/darwin/amd64/kubectl.sha256
```

   - Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.5/2024-01-04/bin/darwin/amd64/kubectl.sha256
```

   - Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.9/2024-01-04/bin/darwin/amd64/kubectl.sha256
```

   - Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.12/2024-01-04/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-01-04/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-01-04/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-01-04/bin/darwin/amd64/kubectl.sha256
```

b. Periksa `SHA-256` checksum untuk biner yang Anda unduh.

```
openssl sha1 -sha256 kubectl
```

c. Pastikan checksum yang dihasilkan dalam output cocok dengan checksum di file yang diunduh`kubectl.sha256`.

3. Menerapkan izin eksekusi ke biner.

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

4. Salin biner ke folder `PATH` Anda. Jika Anda telah menginstal versi `kubectl`, maka kami merekomendasikan untuk membuat `$HOME/bin/kubectl` dan pastikan `$HOME/bin` terinstal lebih dulu di `$PATH` Anda.

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

5. (Opsional) Tambahkan path `$HOME/bin` ke file inisialisasi shell Anda sehingga terkonfigurasi ketika Anda membuka shell.

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

Linux (amd64)

Untuk menginstal atau memperbarui **`kubectl`** pada Linux (**`amd64`**)

1. Unduh `kubectl` biner untuk Kubernetes versi cluster Anda dari Amazon S3.
   - Kubernetes `1.29`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.0/2024-01-04/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.28`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.5/2024-01-04/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.27`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.9/2024-01-04/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.26`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.12/2024-01-04/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.25`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-01-04/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.24`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-01-04/bin/linux/amd64/kubectl
     ```

   - Kubernetes `1.23`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-01-04/bin/linux/amd64/kubectl
     ```

2. (Opsional) Verifikasi biner yang diunduh dengan SHA-256 checksum untuk biner Anda.

   a. menggunakan perintah untuk platform perangkat keras perangkat Anda. Tautan pertama untuk setiap versi adalah untuk amd64 dan tautan kedua untukarm64.

      - Kubernetes 1.29

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.0/2024-01-04/bin/linux/amd64/kubectl.sha256
        ```

      - Kubernetes 1.28

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.5/2024-01-04/bin/linux/amd64/kubectl.sha256
        ```

      - Kubernetes 1.27

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.9/2024-01-04/bin/linux/amd64/kubectl.sha256
        ```

      - Kubernetes 1.26

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.12/2024-01-04/bin/linux/amd64/kubectl.sha256
        ```

      - Kubernetes 1.25

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-01-04/bin/linux/amd64/kubectl.sha256
        ```

      - Kubernetes 1.24

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-01-04/bin/linux/amd64/kubectl.sha256
        ```

      - Kubernetes 1.23

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-01-04/bin/linux/amd64/kubectl.sha256
        ```

   b. Periksa SHA-256 checksum untuk biner yang Anda unduh dengan salah satu perintah berikut.

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

  Saat menggunakan perintah ini, pastikan Anda melihat output berikut:

  ```
  kubectl: OK
  ```

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

  Saat menggunakan perintah ini, pastikan checksum yang dihasilkan dalam output cocok dengan checksum di file yang diunduh`kubectl.sha256`.

3. Menerapkan izin eksekusi ke biner.

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

4. Salin biner ke folder `PATH` Anda. Jika Anda telah menginstal versi `kubectl`, maka kami merekomendasikan untuk membuat `$HOME/bin/kubectl` dan pastikan `$HOME/bin` terinstal lebih dulu di `$PATH` Anda.

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

5. (Opsional) Tambahkan path `$HOME/bin` ke file inisialisasi shell Anda sehingga terkonfigurasi ketika Anda membuka shell.

   > **Note**
   >
   > Langkah ini mengasumsikan Anda menggunakan shell Bash; jika Anda menggunakan shell lain, ubah perintah untuk menggunakan file inisialisasi shell spesifik Anda.

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

Linux (arm64)

Untuk menginstal atau memperbarui **`kubectl`** pada Linux (**`arm64`**)

1. Unduh `kubectl` biner untuk Kubernetes versi cluster Anda dari Amazon S3.
   - Kubernetes `1.29`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.0/2024-01-04/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.28`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.5/2024-01-04/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.27`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.9/2024-01-04/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.26`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.12/2024-01-04/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.25`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-01-04/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.24`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-01-04/bin/linux/arm64/kubectl
     ```

   - Kubernetes `1.23`

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-01-04/bin/linux/arm64/kubectl
     ```

2. (Opsional) Verifikasi biner yang diunduh dengan SHA-256 checksum untuk biner Anda.

   a. menggunakan perintah untuk platform perangkat keras perangkat Anda. Tautan pertama untuk setiap versi adalah untuk amd64 dan tautan kedua untukarm64.

   - Kubernetes 1.29

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.0/2024-01-04/bin/linux/arm64/kubectl.sha256
     ```

   - Kubernetes 1.28

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.5/2024-01-04/bin/linux/arm64/kubectl.sha256
     ```

   - Kubernetes 1.27

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.9/2024-01-04/bin/linux/arm64/kubectl.sha256
     ```

   - Kubernetes 1.26

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.12/2024-01-04/bin/linux/arm64/kubectl.sha256
     ```

   - Kubernetes 1.25

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-01-04/bin/linux/arm64/kubectl.sha256
     ```

   - Kubernetes 1.24

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-01-04/bin/linux/arm64/kubectl.sha256
     ```

   - Kubernetes 1.23

     ```
     curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-01-04/bin/linux/arm64/kubectl.sha256
     ```

   b. Periksa SHA-256 checksum untuk biner yang Anda unduh dengan salah satu perintah berikut.

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

  Saat menggunakan perintah ini, pastikan Anda melihat output berikut:

  ```
  kubectl: OK
  ```

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

  Saat menggunakan perintah ini, pastikan checksum yang dihasilkan dalam output cocok dengan checksum di file yang diunduh`kubectl.sha256`.

3. Menerapkan izin eksekusi ke biner.

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

4. Salin biner ke folder `PATH` Anda. Jika Anda telah menginstal versi `kubectl`, maka kami merekomendasikan untuk membuat `$HOME/bin/kubectl` dan pastikan `$HOME/bin` terinstal lebih dulu di `$PATH` Anda.

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

5. (Opsional) Tambahkan path `$HOME/bin` ke file inisialisasi shell Anda sehingga terkonfigurasi ketika Anda membuka shell.

   > **Note**
   >
   > Langkah ini mengasumsikan Anda menggunakan shell Bash; jika Anda menggunakan shell lain, ubah perintah untuk menggunakan file inisialisasi shell spesifik Anda.

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

Windows

Untuk menginstal atau memperbarui **kubectl**Windows

1. Buka PowerShell terminal.
2. Unduh `kubectl` biner untuk Kubernetes versi cluster Anda dari Amazon S3.
   - Kubernetes `1.29`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.0/2024-01-04/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.28`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.5/2024-01-04/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.27`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.9/2024-01-04/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.26`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.12/2024-01-04/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.25`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-01-04/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.24`

     ```
     curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-01-04/bin/windows/amd64/kubectl.exe
     ```

   - Kubernetes `1.23`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.23.17/2024-01-04/bin/windows/amd64/kubectl.exe
```

3. (Opsional) Verifikasi biner yang diunduh dengan SHA-256 checksum untuk biner Anda.
   a. Unduh SHA-256 checksum untuk Kubernetes versi cluster Anda untukWindows.
      - Kubernetes 1.29

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.29.0/2024-01-04/bin/windows/amd64/kubectl.exe.sha256
```

      - Kubernetes 1.28

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.28.5/2024-01-04/bin/windows/amd64/kubectl.exe.sha256
```

      - Kubernetes 1.27

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.27.9/2024-01-04/bin/windows/amd64/kubectl.exe.sha256
```

      - Kubernetes 1.26

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.26.12/2024-01-04/bin/windows/amd64/kubectl.exe.sha256
```

      - Kubernetes 1.25

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.25.16/2024-01-04/bin/windows/amd64/kubectl.exe.sha256
```

      - Kubernetes 1.24

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.24.17/2024-01-04/bin/windows/amd64/kubectl.exe.sha256
```

      - Kubernetes 1.23

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.23.17/2024-01-04/bin/windows/amd64/kubectl.exe.sha256
```

b. Periksa SHA-256 checksum untuk biner yang Anda unduh.

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

c. Pastikan checksum yang dihasilkan dalam output cocok dengan checksum di file yang diunduh`kubectl.sha256`. PowerShellOutputnya harus berupa string karakter yang setara huruf besar.

4. Salin biner ke folder PATH Anda. Jika Anda memiliki direktori yang ada di dalam Anda PATH yang Anda gunakan untuk utilitas baris perintah, salin biner ke direktori itu. Jika tidak, selesaikan langkah-langkah berikut.

   a. Membuat direktori baru untuk binari baris perintah Anda, seperti `C:\bin`.

   b. Salin biner `kubectl.exe` ke direktori baru Anda.

   c. Edit variabel PATH lingkungan pengguna atau sistem Anda untuk menambahkan direktori baru ke direktori AndaPATH.

   d. Tutup PowerShell terminal Anda dan buka yang baru untuk mengambil PATH variabel baru.

3. Setelah Anda menginstal`kubectl`, Anda dapat memverifikasi versinya.

```
kubectl version --client
```

Saat pertama kali menginstal`kubectl`, itu belum dikonfigurasi untuk berkomunikasi dengan server apa pun. Kami akan membahas konfigurasi ini sesuai kebutuhan dalam prosedur lain. Jika Anda perlu memperbarui konfigurasi untuk berkomunikasi dengan cluster tertentu, Anda dapat menjalankan perintah berikut. Ganti *`region-code`* dengan tempat Wilayah AWS cluster Anda berada. Ganti *`my-cluster`* dengan nama klaster Anda.

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

Memulai dengan Amazon EKS

Pastikan Anda siap untuk menggunakan Amazon EKS sebelum melalui panduan memulai. Untuk informasi selengkapnya, lihat Menyiapkan untuk menggunakan Amazon EKS.

Ada dua panduan memulai yang tersedia untuk membuat Kubernetes cluster baru dengan node di Amazon EKS:

- Memulai dengan Amazon EKS – `eksctl`— Panduan memulai ini membantu Anda menginstal semua sumber daya yang diperlukan untuk memulai menggunakan Amazon EKS`eksctl`, utilitas baris perintah sederhana untuk membuat dan mengelola Kubernetes cluster di Amazon EKS. Di akhir tutorial, Anda akan memiliki klaster Amazon EKS yang sudah berjalan yang dapat digunakan sebagai tempat men-deploy aplikasi. Ini adalah cara tercepat dan paling sederhana untuk memulai dengan Amazon EKS.
- Memulai dengan Amazon EKS — AWS Management Console dan AWS CLI— Panduan memulai ini membantu Anda membuat semua sumber daya yang diperlukan untuk memulai Amazon EKS menggunakan AWS Management Console dan AWS CLI. Di akhir tutorial, Anda akan memiliki klaster Amazon EKS yang sudah berjalan yang dapat digunakan sebagai tempat men-deploy aplikasi. Dalam panduan ini, Anda membuat setiap sumber daya yang diperlukan untuk klaster Amazon EKS secara manual. Prosedur ini memberikan visibilitas tentang bagaimana setiap sumber daya dibuat dan bagaimana mereka berinteraksi satu sama lain.

Kami juga menawarkan koleksi tutorial langsung yang dikuratori. Untuk informasi selengkapnya, lihat Menavigasi Amazon EKS di AWS Komunitas.

Memulai dengan Amazon EKS – **`eksctl`**

Panduan ini membantu Anda membuat semua sumber daya yang diperlukan untuk memulai dengan Amazon Elastic Kubernetes Service (Amazon EKS`eksctl`) menggunakan utilitas baris perintah sederhana untuk membuat dan Kubernetes mengelola cluster di Amazon EKS. Di akhir tutorial, Anda akan memiliki klaster Amazon EKS yang sudah berjalan yang dapat digunakan sebagai tempat men-deploy aplikasi.

Prosedur dalam panduan ini membuatkan beberapa sumber daya untuk Anda secara otomatis, yang seharusnya secara manual, ketika Anda membuat klaster menggunakan AWS Management Console. Jika Anda lebih suka membuat sebagian besar sumber daya secara manual untuk lebih memahami

bagaimana mereka berinteraksi satu sama lain, gunakan AWS Management Console untuk membuat cluster dan komputasi Anda. Untuk informasi selengkapnya, lihat Memulai dengan Amazon EKS — AWS Management Console dan AWS CLI.

Prasyarat

Sebelum memulai tutorial ini, Anda harus menginstal dan mengonfigurasi alat-alat dan sumber daya yang Anda butuhkan berikut untuk membuat dan mengelola sebuah klaster Amazon EKS.

- **`kubectl`**— Alat baris perintah untuk bekerja dengan Kubernetes cluster. Untuk informasi selengkapnya, lihat Menginstal atau memperbarui `kubectl`.
- **`eksctl`** — Alat baris perintah agar bisa bekerja dengan klaster EKS yang mengotomatisasi banyak tugas individu. Untuk informasi selengkapnya, lihat Instalasi di `eksctl` dokumentasi.
- Izin IAM yang diperlukan - Prinsip keamanan IAM yang Anda gunakan harus memiliki izin untuk bekerja dengan peran Amazon EKS IAM, peran terkait layanan,, VPC, dan sumber AWS CloudFormation daya terkait. Untuk informasi selengkapnya, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon Elastic Container Service untuk Kubernetes dan Menggunakan peran terkait layanan di Panduan Pengguna IAM. Anda harus menyelesaikan semua langkah dalam panduan ini sebagai pengguna yang sama. Untuk memeriksa pengguna saat ini, jalankan perintah berikut:

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

Langkah 1: Buat klaster dan simpul Amazon EKS Anda

> **Important**
>
> Untuk memulai sesederhana dan secepat mungkin, topik ini mencakup langkah-langkah untuk membuat klaster dan simpul dengan pengaturan default. Sebelum membuat klaster dan simpul untuk produksi, kami merekomendasikan supaya Anda membiasakan diri dengan semua pengaturan, dan men-deploy klaster serta simpul dengan pengaturan yang sesuai dengan kebutuhan Anda. Untuk informasi selengkapnya, lihat Membuat klaster Amazon EKS dan Simpul Amazon EKS. Beberapa pengaturan hanya dapat diaktifkan saat membuat cluster dan node Anda.

Anda dapat membuat sebuah klaster dengan salah satu jenis simpul berikut. Untuk mempelajari selengkapnya tentang setiap jenis, lihat Simpul Amazon EKS. Setelah klaster Anda di-deploy, Anda dapat menambahkan jenis simpul lainnya.

- Fargate — Linux — Pilih jenis node ini jika Anda ingin menjalankan Linux aplikasi. AWS Fargate Fargate adalah mesin komputasi tanpa server yang memungkinkan Anda menerapkan tanpa Kubernetes Pods mengelola instans Amazon EC2.
- Node terkelola Linux — — Pilih jenis node ini jika Anda ingin menjalankan aplikasi Amazon Linux di instans Amazon EC2. Meskipun tidak tercakup dalam panduan ini, Anda juga dapat menambahkan Windowsself-managed dan Bottlerocketnode ke cluster Anda.

Buat cluster Amazon EKS Anda dengan perintah berikut. Anda dapat mengganti *my-cluster* dengan nilai Anda sendiri. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter. Ganti *region-code* dengan apa pun Wilayah AWS yang didukung oleh Amazon EKS. Untuk daftar Wilayah AWS, lihat titik akhir dan kuota Amazon EKS di panduan Referensi AWS Umum.

Fargate – Linux

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

Managed nodes – Linux

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

Pembuatan cluster membutuhkan waktu beberapa menit. Selama pembuatan Anda akan melihat beberapa baris output. Baris terakhir output mirip dengan baris contoh berikut.

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

`eksctl`membuat `kubectl config` file di `~/.kube` atau menambahkan konfigurasi cluster baru dalam `config` file yang ada `~/.kube` di komputer Anda.

Setelah pembuatan cluster selesai, lihat AWS CloudFormation tumpukan bernama `eksctl-`*my-cluster*`-cluster` di AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation untuk melihat semua sumber daya yang dibuat.

Langkah 2: Lihat Kubernetes sumber daya

1. Lihat simpul klaster Anda.

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

Contoh output adalah sebagai berikut.

Fargate – Linux

```
NAME                                                  STATUS   ROLES    AGE
   VERSION              INTERNAL-IP   EXTERNAL-IP   OS-IMAGE         KERNEL-
VERSION                   CONTAINER-RUNTIME
fargate-ip-192-0-2-0.region-code.compute.internal   Ready    <none>
  8m3s    v1.2.3-eks-1234567   192.0.2.0     <none>        Amazon Linux 2
  1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
fargate-ip-192-0-2-1.region-code.compute.internal   Ready    <none>
  7m30s   v1.2.3-eks-1234567   192-0-2-1     <none>        Amazon Linux 2
  1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
```

Managed nodes – Linux

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

Untuk informasi lebih lanjut tentang apa yang Anda lihat di output, lihatLihat Kubernetes sumber daya.

2. Melihat beban kerja yang berjalan di klaster Anda.

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

Contoh output adalah sebagai berikut.

Fargate – Linux

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

Managed nodes – Linux

```
NAMESPACE     NAME                          READY   STATUS    RESTARTS   AGE     IP
          NODE                                        NOMINATED NODE   READINESS
 GATES
kube-system   aws-node-12345             1/1     Running   0          7m43s
   192.0.2.1   ip-192-0-2-1.region-code.compute.internal   <none>
 <none>
kube-system   aws-node-67890             1/1     Running   0          7m46s
   192.0.2.0   ip-192-0-2-0.region-code.compute.internal   <none>
 <none>
kube-system   coredns-1234567890-abcde   1/1     Running   0          14m
   192.0.2.3   ip-192-0-2-3.region-code.compute.internal   <none>
 <none>
kube-system   coredns-1234567890-12345   1/1     Running   0          14m
   192.0.2.4   ip-192-0-2-4.region-code.compute.internal   <none>
 <none>
kube-system   kube-proxy-12345           1/1     Running   0          7m46s
   192.0.2.0   ip-192-0-2-0.region-code.compute.internal   <none>
 <none>
kube-system   kube-proxy-67890           1/1     Running   0          7m43s
   192.0.2.1   ip-192-0-2-1.region-code.compute.internal   <none>
 <none>
```

Untuk informasi lebih lanjut tentang apa yang Anda lihat di output, lihatLihat Kubernetes sumber daya.

Langkah 3: Hapus klaster dan simpul Anda

Setelah Anda selesai dengan cluster dan node yang Anda buat untuk tutorial ini, Anda harus membersihkan dengan menghapus cluster dan node dengan perintah berikut. Jika Anda ingin melakukan lebih banyak dengan klaster ini sebelum Anda membersihkannya, lihat Langkah selanjutnya.

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

Langkah selanjutnya

Topik dokumentasi berikut membantu Anda untuk memperluas fungsionalitas klaster Anda.

- Menerapkan aplikasi sampel ke cluster Anda.
- Prinsipal IAM yang membuat cluster adalah satu-satunya prinsipal yang dapat melakukan panggilan ke server Kubernetes API dengan `kubectl` atau. AWS Management Console Jika Anda ingin prinsipal IAM lainnya memiliki akses ke cluster Anda, maka Anda perlu menambahkannya. Lihat informasi yang lebih lengkap di Berikan akses ke Kubernetes API dan Izin yang diperlukan.
- Sebelum menerapkan cluster untuk penggunaan produksi, sebaiknya Anda membiasakan diri dengan semua pengaturan untuk cluster dan node. Beberapa pengaturan (seperti mengaktifkan akses SSH ke node Amazon EC2) harus dilakukan saat cluster dibuat.
- Untuk meningkatkan keamanan klaster Anda, konfigurasikan plugin Amazon VPC Container Networking Interface untuk menggunakan peran IAM untuk akun layanan.

Memulai dengan Amazon EKS — AWS Management Console dan AWS CLI

Panduan ini membantu Anda membuat semua sumber daya yang diperlukan untuk memulai Amazon Elastic Kubernetes Service (Amazon EKS) menggunakan dan. AWS Management Console AWS CLI Dalam panduan ini, Anda membuat setiap sumber daya secara manual. Di akhir tutorial, Anda akan memiliki klaster Amazon EKS yang sudah berjalan yang dapat digunakan sebagai tempat men-deploy aplikasi.

Prosedur dalam panduan ini memberikan visibilitas lengkap tentang bagaimana setiap sumber daya dibuat dan bagaimana mereka berinteraksi satu sama lain. Jika Anda lebih menyukai jika sebagian

besar sumber daya dibuat untuk Anda secara otomatis, gunakan `eksctl` CLI untuk membuat klaster dan simpul Anda. Untuk informasi selengkapnya, lihat Memulai dengan Amazon EKS – `eksctl`.

Prasyarat

Sebelum memulai tutorial ini, Anda harus menginstal dan mengonfigurasi alat-alat dan sumber daya yang Anda butuhkan berikut untuk membuat dan mengelola sebuah klaster Amazon EKS.

- AWS CLI— Alat baris perintah untuk bekerja dengan AWS layanan, termasuk Amazon EKS. Untuk informasi selengkapnya, lihat Menginstal, memperbarui, dan mencopot instalasi AWS CLI di Panduan Pengguna AWS Command Line Interface . Setelah menginstal AWS CLI, kami sarankan Anda juga mengkonfigurasinya. Untuk informasi selengkapnya, lihat Konfigurasi cepat dengan `aws configure` dalam Panduan Pengguna AWS Command Line Interface .
- **`kubectl`**— Alat baris perintah untuk bekerja dengan Kubernetes cluster. Untuk informasi selengkapnya, lihat Menginstal atau memperbarui `kubectl`.
- Izin IAM yang diperlukan - Prinsip keamanan IAM yang Anda gunakan harus memiliki izin untuk bekerja dengan peran Amazon EKS IAM, peran terkait layanan,, VPC, dan sumber AWS CloudFormation daya terkait. Untuk informasi selengkapnya, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon Elastic Kubernetes Service dan Menggunakan peran terkait layanan di Panduan Pengguna IAM. Anda harus menyelesaikan semua langkah dalam panduan ini sebagai pengguna yang sama. Untuk memeriksa pengguna saat ini, jalankan perintah berikut:

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

- Kami menyarankan Anda menyelesaikan langkah-langkah dalam topik ini di shell Bash. Jika Anda tidak menggunakan shell Bash, beberapa perintah skrip seperti karakter kelanjutan baris dan cara variabel diatur dan digunakan memerlukan penyesuaian untuk shell Anda. Selain itu, aturan mengutip dan melarikan diri untuk shell Anda mungkin berbeda. Untuk informasi selengkapnya, lihat Menggunakan tanda kutip dengan string AWS CLI di AWS Command Line Interface Panduan Pengguna.

Langkah 1: Buat klaster dan Amazon EKS Anda

> **Important**
>
> Untuk memulai sesederhana dan secepat mungkin, topik ini mencakup langkah-langkah untuk membuat cluster dengan pengaturan default. Sebelum membuat cluster untuk

penggunaan produksi, kami sarankan Anda membiasakan diri dengan semua pengaturan dan menyebarkan cluster dengan pengaturan yang memenuhi persyaratan Anda. Untuk informasi selengkapnya, lihat Membuat klaster Amazon EKS. Beberapa pengaturan hanya dapat diaktifkan saat membuat cluster Anda.

Untuk membuat klaster Anda

1. Buat Amazon VPC dengan subnet publik dan privat yang memenuhi persyaratan Amazon EKS. Ganti *region-code* dengan apa pun Wilayah AWS yang didukung oleh Amazon EKS. Untuk daftar Wilayah AWS, lihat titik akhir dan kuota Amazon EKS di panduan Referensi AWS Umum.Anda dapat mengganti *my-eks-vpc-stack* dengan nama apa pun yang Anda pilih.

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

> **Tip**
>
> Untuk daftar semua sumber daya yang dibuat oleh perintah sebelumnya, buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation. Pilih *my-eks-vpc-stack* tumpukan dan kemudian pilih tab Sumber Daya.

2. Buat peran IAM cluster dan lampirkan kebijakan terkelola Amazon EKS IAM yang diperlukan ke dalamnya. Kubernetescluster yang dikelola oleh Amazon EKS melakukan panggilan ke AWS layanan lain atas nama Anda untuk mengelola sumber daya yang Anda gunakan dengan layanan ini.

   a. Salin isi berikut ke file bernama *eks-cluster-role-trust-policy.json*.

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

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

b. Buat peran.

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

c. Lampirkan kebijakan terkelola IAM Amazon EKS yang diperlukan untuk peran tersebut.

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

3. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.

   Pastikan bahwa yang Wilayah AWS ditampilkan di kanan atas konsol Anda Wilayah AWS adalah tempat Anda ingin membuat cluster Anda. Jika tidak, pilih dropdown di sebelah Wilayah AWS nama dan pilih Wilayah AWS yang ingin Anda gunakan.
4. Pilih Add cluster, lalu pilih Create. Jika Anda tidak melihat opsi ini, pilih Cluster di panel navigasi kiri terlebih dahulu.
5. Pada halaman Configure cluster, lakukan hal berikut:

   a. Masukkan Nama untuk klaster Anda, seperti**`my-cluster`**. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter.

   b. Untuk Peran Layanan Cluster, pilih *`MyAmazoneks ClusterRole`*.

   c. Biarkan pengaturan yang tersisa pada nilai defaultnya dan pilih Berikutnya.
6. Pada halaman Tentukan jaringan, lakukan hal berikut:

   a. Pilih ID VPC yang Anda buat pada langkah sebelumnya dari daftar dropdown VPC. *`Ini adalah sesuatu seperti vpc-00x0000x000x0x000 | -VPC. my-eks-vpc-stack`*

   b. Biarkan pengaturan yang tersisa pada nilai defaultnya dan pilih Berikutnya.
7. Pada halaman Konfigurasi observabilitas, pilih Berikutnya.

8. Pada halaman Pilih add-on, pilih Berikutnya.

   Untuk informasi lebih lanjut tentang add-on, lihatAdd-on Amazon EKS.

9. Pada halaman Konfigurasi pengaturan add-on yang dipilih, pilih Berikutnya.

10. Pada halaman Tinjau dan buat, pilih Buat.

    Di sebelah kanan nama klaster, status klaster adalah Membuat selama beberapa menit sampai proses penyediaan klaster selesai. Jangan lanjutkan ke langkah berikutnya sampai status berubah menjadi Aktif.

    > **Note**
    >
    > Mungkin error akan terjadi karena salah satu Availability Zone dalam permintaan Anda tidak memiliki kapasitas yang cukup untuk membuat klaster Amazon EKS. Jika hal ini terjadi, output galat berisi Availability Zones yang dapat mendukung klaster baru. Cobalah untuk kembali membuat klaster dengan setidaknya dua subnet yang terletak di Availability Zones yang didukung untuk akun Anda. Untuk informasi selengkapnya, lihat Kapasitas tidak mencukupi.

Langkah 2: Konfigurasikan komputer Anda agar bisa berkomunikasi dengan klaster Anda

Di bagian ini, Anda akan membuat file `kubeconfig` untuk klaster Anda. Pengaturan dalam file ini mengaktifkan `kubectl` CLI agar dapat berkomunikasi dengan klaster Anda.

Untuk mengonfigurasi komputer Anda agar bisa berkomunikasi dengan klaster Anda

1. Buat atau perbarui file `kubeconfig` untuk klaster Anda. Ganti *`region-code`* dengan tempat Wilayah AWS Anda membuat cluster Anda. Ganti *`my-cluster`* dengan nama klaster Anda.

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

   Secara default, file `config` dibuat di `~/.kube` atau konfigurasi klaster baru ditambahkan ke file `config` yang sudah ada di `~/.kube`.

2. Uji konfigurasi Anda.

```
kubectl get svc
```

> **Note**
>
> Jika Anda menerima kesalahan otorisasi atau jenis sumber daya, lihat Tidak sah atau akses ditolak (kubectl) di topik pemecahan masalah.

Contoh output adalah sebagai berikut.

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

Langkah 3: Buat node

> **Important**
>
> Untuk memulai sesederhana dan secepat mungkin, topik ini mencakup langkah-langkah untuk membuat node dengan pengaturan default. Sebelum membuat node untuk penggunaan produksi, kami sarankan Anda membiasakan diri dengan semua pengaturan dan menyebarkan node dengan pengaturan yang memenuhi persyaratan Anda. Untuk informasi selengkapnya, lihat Simpul Amazon EKS. Beberapa pengaturan hanya dapat diaktifkan saat membuat node Anda.

Anda dapat membuat sebuah klaster dengan salah satu jenis simpul berikut. Untuk mempelajari selengkapnya tentang setiap jenis, lihat Simpul Amazon EKS. Setelah klaster Anda di-deploy, Anda dapat menambahkan jenis simpul lainnya.

- Fargate — Linux — Pilih jenis node ini jika Anda ingin menjalankan Linux aplikasi. AWS Fargate Fargate adalah mesin komputasi tanpa server yang memungkinkan Anda menerapkan tanpa Kubernetes Pods mengelola instans Amazon EC2.
- Node terkelola Linux — — Pilih jenis node ini jika Anda ingin menjalankan aplikasi Amazon Linux di instans Amazon EC2. Meskipun tidak tercakup dalam panduan ini, Anda juga dapat menambahkan Windowsself-managed dan Bottlerocketnode ke cluster Anda.

Fargate – Linux

Buat profil Fargate. Ketika Kubernetes Pods diterapkan dengan kriteria yang cocok dengan kriteria yang ditentukan dalam profil, akan Pods diterapkan ke Fargate.

Untuk membuat profil Fargate

1. Buat IAM role dan lampirkan kebijakan terkelola Amazon EKS IAM ke dalam peran itu. Saat klaster Anda membuat Pods infrastruktur Fargate, komponen yang berjalan di infrastruktur Fargate harus melakukan panggilan ke AWS API atas nama Anda. Ini agar mereka dapat melakukan tindakan seperti menarik gambar kontainer dari Amazon ECR atau merutekan log ke AWS layanan lain. Peran Pod eksekusi Amazon EKS memberikan izin IAM untuk melakukan ini.

   a. Salin isi berikut ke file bernama *pod-execution-role-trust-policy*.json. Ganti *region-code* dengan tempat Wilayah AWS cluster Anda berada. Jika Anda ingin menggunakan peran yang sama Wilayah AWS di semua akun Anda, ganti *region-code* dengan*. Ganti *111122223333* dengan ID akun Anda dan *my-cluster* dengan nama cluster Anda. Jika Anda ingin menggunakan peran yang sama untuk semua cluster di akun Anda, ganti *my-cluster* dengan*.

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

   b. Buat peran IAM Pod eksekusi.

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

c. Lampirkan kebijakan terkelola IAM Amazon EKS yang diperlukan untuk peran tersebut.

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

2. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
3. Pada halaman Clusters, pilih *my-cluster* cluster.
4. Pada halaman ***my-cluster***, lakukan hal berikut:
   a. Pilih tab Compute.
   b. Di bawah Profil Fargate, pilih Tambahkan Profil Fargate.
5. Pada halaman Konfigurasi Profil Fargate, lakukan hal berikut:
   a. Untuk Nama, masukkan nama unik untuk profil Fargate Anda, seperti ***my-profile***.
   b. Untuk peran eksekusi Pod, pilih FargatePodExecutionRoleAmazonEks yang Anda buat pada langkah sebelumnya.
   c. Pilih dropdown Subnet dan batalkan pilihan subnet apa pun dengan namanya. `Public` Hanya subnet pribadi yang didukung untuk Pods yang berjalan di Fargate.
   d. Pilih Berikutnya.
6. Pada halaman Podpilihan Konfigurasi, lakukan hal berikut:
   a. Untuk Namespace, masukkan **`default`**.
   b. Pilih Berikutnya.
7. Pada halaman Periksa dan buat, tinjau informasi untuk profil Fargate Anda dan pilih Buat.
8. Setelah beberapa menit, Status di bagian konfigurasi Profil Fargate akan berubah dari Membuat menjadi Aktif. Jangan lanjutkan ke langkah berikutnya sampai status berubah menjadi Aktif.

9. Jika Anda berencana untuk menerapkan semua Pods ke Fargate (tidak ada ke node Amazon EC2), lakukan hal berikut untuk membuat profil Fargate lain dan jalankan resolver nama default () di Fargate. CoreDNS

> **Note**
>
> Jika Anda tidak melakukan ini, Anda tidak akan memiliki node saat ini.

a. *Pada halaman **Profil Fargate, pilih profil** saya.*

b. Di bawah profil Fargate, pilih Tambahkan Profil Fargate.

c. Untuk Nama, masukkan ***CoreDNS***.

d. Untuk peran eksekusi Pod, pilih FargatePodExecutionRoleAmazonEks yang Anda buat pada langkah sebelumnya.

e. Pilih dropdown Subnet dan batalkan pilihan subnet apa pun dengan namanya. `Public` Hanya subnet pribadi yang didukung untuk Pods berjalan di Fargate.

f. Pilih Berikutnya.

g. Untuk Namespace, masukkan **`kube-system`**.

h. Pilih Match labels, lalu pilih Add label.

i. Masukkan **`k8s-app`** untuk Kunci dan **`kube-dns`** nilai. Ini diperlukan untuk nama default resolver (CoreDNS) untuk menyebarkan ke Fargate.

j. Pilih Berikutnya.

k. Pada halaman Periksa dan buat, tinjau informasi untuk profil Fargate Anda dan pilih Buat.

l. Jalankan perintah berikut untuk menghapus `eks.amazonaws.com/compute-type : ec2` anotasi default dari. CoreDNS Pods

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

> **Note**
>
> Sistem membuat dan menyebarkan dua node berdasarkan label profil Fargate yang Anda tambahkan. Anda tidak akan melihat apa pun yang tercantum dalam grup Node karena tidak berlaku untuk node Fargate, tetapi Anda akan melihat node baru yang tercantum di tab Ikhtisar.

Managed nodes – Linux

Membuat grup simpul terkelola, menentukan subnet dan IAM role simpul yang Anda buat di langkah-langkah sebelumnya.

Untuk membuat grup node Linux terkelola Amazon EC2

1. Buat IAM role simpul dan lampirkan kebijakan terkelola Amazon EKS IAM ke dalam peran itu. `kubelet`Daemon node Amazon EKS melakukan panggilan ke AWS API atas nama Anda. Simpul menerima izin untuk panggilan API ini melalui profil instans IAM dan kebijakan terkait.

   a. Salin konten berikut ke file bernama *`node-role-trust-policy.json`*.

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

   b. Buat IAM role simpul.

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

c. Lampirkan kebijakan IAM terkelola yang diperlukan ke peran.

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

2. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
3. Pilih nama klaster yang Anda buat di Langkah 1: Buat klaster dan Amazon EKS Anda, misalnya *my-cluster*.
4. Pada halaman ***my-cluster***, lakukan hal berikut:
   a. Pilih tab Compute.
   b. Pilih Tambahkan Grup Node.
5. Pada halaman Configure Node Group, lakukan hal berikut:
   a. Untuk Nama, masukkan nama unik untuk grup node terkelola Anda, seperti***my-nodegroup***. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa.
   b. Untuk nama peran Node IAM, pilih NodeRole peran *MyAmazoneks* yang Anda buat di langkah sebelumnya. Kami merekomendasikan bahwa setiap grup node menggunakan peran IAM uniknya sendiri.
   c. Pilih Berikutnya.
6. Pada halaman Setel konfigurasi komputasi dan penskalaan, terima nilai default dan pilih Berikutnya.
7. Pada halaman Tentukan jaringan, terima nilai default dan pilih Berikutnya.
8. Pada halaman Tinjauan dan buat, tinjau konfigurasi grup simpul terkelola dan pilih Buat.
9. Setelah beberapa menit, Status di bagian Konfigurasi Grup simpul akan berubah dari Sedang membuat ke Aktif. Jangan lanjutkan ke langkah berikutnya sampai status berubah menjadi Aktif.

Langkah 4: Lihat sumber daya

Anda dapat melihat node dan Kubernetes beban kerja Anda.

Untuk melihat node dan beban kerja

1. Pada panel navigasi sebelah kiri, pilih Klaster. Dalam daftar Cluster, pilih nama cluster yang Anda buat, seperti *my-cluster*.
2. Pada halaman ***my-cluster***, pilih yang berikut ini:
   a. Tab komputasi — Anda melihat daftar Node yang digunakan untuk cluster. Anda dapat memilih nama node untuk melihat informasi lebih lanjut tentangnya.
   b. Tab Sumber Daya — Anda melihat semua Kubernetes sumber daya yang digunakan secara default ke kluster Amazon EKS. Pilih jenis sumber daya apa pun di konsol untuk mempelajarinya lebih lanjut.

Langkah 5: Hapus sumber daya

Setelah Anda selesai dengan cluster dan node yang Anda buat untuk tutorial ini, Anda harus menghapus sumber daya yang Anda buat. Jika Anda ingin melakukan lebih banyak hal dengan cluster ini sebelum menghapus sumber daya, lihatLangkah selanjutnya.

Untuk menghapus sumber daya yang Anda buat dalam panduan ini

1. Hapus grup node atau profil Fargate yang Anda buat.
   a. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
   b. Pada panel navigasi sebelah kiri, pilih Klaster. Dalam daftar cluster, pilih *my-cluster*.
   c. Pilih tab Compute.
   d. Jika Anda membuat grup node, c hoose grup *my-nodegroup* node dan kemudian pilih Delete. Masukkan ***my-nodegroup***, lalu pilih Hapus.
   e. Untuk setiap profil Fargate yang Anda buat, pilih dan kemudian pilih Hapus. Masukkan nama profil, lalu pilih Hapus.

> **Note**
>
> Saat menghapus profil Fargate kedua, Anda mungkin harus menunggu yang pertama selesai dihapus.

f. Jangan lanjutkan sampai grup node atau profil Fargate .

2. Hapus klaster .
   a. Pada panel navigasi sebelah kiri, pilih Klaster. Dalam daftar cluster, pilih *my-cluster*.
   b. Pilih Hapus klaster.
   c. Masuk ***my-cluster*** dan kemudian pilih Hapus. Jangan lanjutkan sampai cluster dihapus.
3. Hapus AWS CloudFormation tumpukan VPC yang Anda buat.
   a. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
   b. Pilih *my-eks-vpc-stack* tumpukan, lalu pilih Hapus.
   c. Di kotak dialog Hapus ***my-eks-vpc-stack***konfirmasi, pilih Hapus tumpukan.
4. Hapus IAM role yang Anda buat.
   a. Buka konsol IAM di https://console.aws.amazon.com/iam/.
   b. Di panel navigasi sebelah kiri, pilih Peran.
   c. ***Pilih setiap peran yang Anda buat dari daftar (MyAmazoneksClusterRole, serta Amazoneks atau MyAmazoneks). FargatePodExecutionRole NodeRole*** Pilih Hapus, masukkan teks konfirmasi yang diminta, lalu pilih Hapus.

Langkah selanjutnya

Topik dokumentasi berikut membantu Anda untuk memperluas fungsionalitas klaster Anda.

- Prinsipal IAM yang membuat cluster adalah satu-satunya prinsipal yang dapat melakukan panggilan ke server Kubernetes API dengan `kubectl` atau. AWS Management Console Jika Anda ingin prinsipal IAM lainnya memiliki akses ke cluster Anda, maka Anda perlu menambahkannya. Lihat informasi yang lebih lengkap di Berikan akses ke Kubernetes API dan Izin yang diperlukan.
- Menerapkan aplikasi sampel ke cluster Anda.

- Sebelum menerapkan cluster untuk penggunaan produksi, sebaiknya Anda membiasakan diri dengan semua pengaturan untuk cluster dan node. Beberapa pengaturan (seperti mengaktifkan akses SSH ke node Amazon EC2) harus dilakukan saat cluster dibuat.
- Untuk meningkatkan keamanan klaster Anda, konfigurasikan plugin Amazon VPC Container Networking Interface untuk menggunakan peran IAM untuk akun layanan.

Klaster-klaster Amazon EKS

Sebuah klaster Amazon EKS terdiri dari dua komponen utama:

- Bidang pengendali Amazon EKS
- Simpul Amazon EKS yang terdaftar dengan bidang kontrol

Bidang kontrol Amazon EKS terdiri dari node bidang kontrol yang menjalankan Kubernetes perangkat lunak, seperti `etcd` dan server Kubernetes API. Bidang kontrol berjalan di akun yang dikelola oleh AWS, dan Kubernetes API diekspos melalui titik akhir Amazon EKS yang terkait dengan cluster Anda. Setiap bidang kendali klaster Amazon EKS adalah penyewa tunggal dan unik, serta berjalan pada pengaturan instans Amazon EC2 miliknya sendiri.

Semua data yang disimpan oleh `etcd` node dan volume Amazon EBS terkait dienkripsi menggunakan. AWS KMS Control plane klaster disediakan di beberapa Availability Zone dan dihadapkan oleh Penyeimbang Beban Jaringan Elastic Load Balancing. Amazon EKS juga menyediakan antarmuka jaringan elastis di subnet VPC Anda untuk menyediakan konektivitas dari instance bidang kontrol ke node (misalnya, untuk `kubectl exec logs proxy` mendukung aliran data).

> **Important**
>
> Di lingkungan Amazon EKS, `etcd` penyimpanan dibatasi hingga 8 GiB sesuai panduan hulu. Anda dapat memantau metrik untuk ukuran database saat ini dengan menjalankan perintah berikut. Jika klaster Anda memiliki Kubernetes versi di bawah ini`1.28`, ganti *`apiserver_storage_size_bytes`* dengan yang berikut:
>
> - Kubernetesversi `1.27` dan `1.26` - **`apiserver_storage_db_total_size_in_bytes`**
> - Kubernetesversi `1.25` dan di bawah ini - **`etcd_db_total_size_in_bytes`**
>
> ```
> kubectl get --raw=/metrics | grep "apiserver_storage_size_bytes"
> ```

Node Amazon EKS berjalan di AWS akun Anda dan terhubung ke bidang kontrol klaster Anda melalui titik akhir server API dan file sertifikat yang dibuat untuk klaster Anda.

> **Note**
>
> - Anda dapat mengetahui bagaimana berbagai komponen Amazon EKS yang berbeda bekerja di Jaringan Amazon EKS.
> - Untuk cluster yang terhubung, lihatKonektor Amazon EKS.

Topik

Membuat klaster Amazon EKS

Topik ini memberikan ikhtisar opsi yang tersedia dan menjelaskan apa yang harus dipertimbangkan saat Anda membuat klaster Amazon EKS. Jika Anda perlu membuat cluster di AWS Outpost, lihatCluster lokal untuk Amazon EKS di AWS Outposts. Jika ini adalah pertama kalinya Anda membuat cluster Amazon EKS, kami sarankan Anda mengikuti salah satu Memulai dengan Amazon EKS panduan kami. Panduan ini membantu Anda membuat cluster default yang sederhana tanpa memperluas ke semua opsi yang tersedia.

Prasyarat

- VPC dan subnet yang ada yang memenuhi persyaratan Amazon EKS. Sebelum Anda menerapkan klaster untuk penggunaan produksi, kami sarankan Anda memiliki pemahaman menyeluruh

tentang persyaratan VPC dan subnet. Jika Anda tidak memiliki VPC dan subnet, Anda dapat membuatnya menggunakan template yang disediakan Amazon EKS. AWS CloudFormation

- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl` versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.
- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`,`apt-get`, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru dari file AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Prinsipal IAM dengan izin `create` dan kluster `describe` Amazon EKS. Lihat informasi yang lebih lengkap di Buat Kubernetes cluster lokal di Outpost dan Buat daftar atau deskripsikan semua klaster.

Untuk membuat cluster Amazon EKS

1. Jika Anda sudah memiliki peran IAM cluster, atau Anda akan membuat cluster Anda dengan`eksctl`, maka Anda dapat melewati langkah ini. Secara default, `eksctl` buat peran untuk Anda.

   Untuk membuat peran IAM cluster Amazon EKS

   1. Jalankan perintah berikut untuk membuat file JSON kebijakan kepercayaan IAM.

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

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

2. Buat peran IAM cluster Amazon EKS. Jika perlu, kata pengantar *eks-cluster-role-trust-policy.json* dengan jalur di komputer Anda tempat Anda menulis file di langkah sebelumnya. Perintah tersebut mengaitkan kebijakan kepercayaan yang Anda buat pada langkah sebelumnya dengan peran tersebut. Untuk membuat peran IAM, prinsipal IAM yang membuat peran harus diberi `iam:CreateRole` tindakan (izin).

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

3. Anda dapat menetapkan kebijakan terkelola Amazon EKS atau membuat kebijakan kustom Anda sendiri. Untuk izin minimum yang harus Anda gunakan dalam kebijakan kustom, lihatIAM role klaster Amazon EKS.

   Lampirkan kebijakan terkelola Amazon EKS yang diberi nama `AmazonEKSClusterPolicy`ke peran tersebut. Untuk melampirkan kebijakan IAM ke kepala sekolah IAM, prinsipal yang melampirkan kebijakan harus diberikan salah satu tindakan IAM berikut (izin): atau. `iam:AttachUserPolicy` `iam:AttachRolePolicy`

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

2. Buat klaster Amazon EKS.

   Anda dapat membuat cluster dengan menggunakan`eksctl`, AWS Management Console, atau AWS CLI.

   eksctl

   Prasyarat

   Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Untuk membuat klaster Anda

Buat IPv4 klaster Amazon EKS dengan Kubernetes versi default Amazon EKS di default Anda Wilayah AWS. Sebelum menjalankan perintah, buat penggantian berikut:

- Ganti *region-code* dengan Wilayah AWS yang Anda inginkan untuk membuat cluster Anda.
- Ganti *my-cluster* dengan nama untuk cluster Anda. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter. Nama harus unik di dalam Wilayah AWS dan Akun AWS tempat Anda membuat cluster.
- Ganti *1.28* dengan versi yang didukung Amazon EKS.
- Ubah nilai `vpc-private-subnets` untuk memenuhi kebutuhan Anda. Anda juga dapat menambahkan ID tambahan. Anda harus menentukan setidaknya dua ID subnet. Jika Anda lebih suka menentukan subnet publik, Anda dapat mengubah `--vpc-private-subnets` ke`--vpc-public-subnets`. Subnet publik memiliki tabel rute terkait dengan rute ke gateway internet, tetapi subnet pribadi tidak memiliki tabel rute terkait. Sebaiknya gunakan subnet pribadi bila memungkinkan.

  Subnet yang Anda pilih harus memenuhi persyaratan subnet Amazon EKS. Sebelum memilih subnet, sebaiknya Anda terbiasa dengan semua persyaratan dan pertimbangan VPC Amazon EKS dan subnet.

```
eksctl create cluster --name my-cluster --region region-code --version 1.28 --
vpc-private-subnets subnet-ExampleID1,subnet-ExampleID2 --without-nodegroup
```

Penyediaan klaster memerlukan waktu beberapa menit. Saat cluster sedang dibuat, beberapa baris output muncul. Baris terakhir output mirip dengan baris contoh berikut.

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

> **Tip**
>
> Untuk melihat sebagian besar opsi yang dapat Anda tentukan saat membuat cluster`eksctl`, gunakan **`eksctl create cluster --help`** perintah. Untuk melihat semua opsi yang tersedia, Anda dapat menggunakan `config` file. Untuk

informasi selengkapnya, lihat Menggunakan file config dan skema file config di dokumentasi `eksctl`. Anda dapat menemukan contoh file konfigurasi diGitHub.

Pengaturan opsional

Berikut ini adalah pengaturan opsional yang, jika diperlukan, harus ditambahkan ke perintah sebelumnya. Anda hanya dapat mengaktifkan opsi ini saat membuat cluster, bukan setelahnya. Jika Anda perlu menentukan opsi ini, Anda harus membuat cluster dengan file `eksctl` konfigurasi dan menentukan pengaturan, daripada menggunakan perintah sebelumnya.

- Jika Anda ingin menentukan satu atau beberapa grup keamanan yang ditetapkan Amazon EKS ke antarmuka jaringan yang dibuatnya, tentukan opsi. `securityGroup`

  Apakah Anda memilih grup keamanan atau tidak, Amazon EKS membuat grup keamanan yang memungkinkan komunikasi antara cluster dan VPC Anda. Amazon EKS mengaitkan grup keamanan ini, dan apa pun yang Anda pilih, ke antarmuka jaringan yang dibuatnya. Untuk informasi selengkapnya tentang grup keamanan klaster yang dibuat Amazon EKS, lihatthe section called “Persyaratan grup keamanan”. Anda dapat mengubah aturan di grup keamanan klaster yang dibuat Amazon EKS.
- Jika Anda ingin menentukan blok `IPv4` Classless Inter-domain Routing (CIDR) mana yang Kubernetes memberikan alamat IP layanan, tentukan opsinya. `serviceIPv4CIDR`

  Menentukan jangkauan Anda sendiri dapat membantu mencegah konflik antara Kubernetes layanan dan jaringan lain yang diintip atau terhubung ke VPC Anda. Masukkan rentang dalam notasi CIDR. Misalnya: `10.2.0.0/16`.

  Blok CIDR harus memenuhi persyaratan berikut:

  - Berada dalam salah satu rentang berikut:`10.0.0.0/8`,`172.16.0.0/12`, atau`192.168.0.0/16`.
  - Memiliki ukuran minimum `/24` dan ukuran maksimum`/12`.
  - Tidak tumpang tindih dengan kisaran VPC untuk sumber daya Amazon EKS Anda.

  Anda hanya dapat menentukan opsi ini saat menggunakan keluarga `IPv4` alamat dan hanya pada pembuatan cluster. Jika Anda tidak menentukan ini, maka Kubernetes tetapkan alamat IP layanan dari blok `10.100.0.0/16` atau `172.20.0.0/16` CIDR.

- Jika Anda membuat klaster dan ingin klaster menetapkan `IPv6` alamat Pods dan layanan, bukan `IPv4` alamat, tentukan `ipFamily`opsi.

  Kubernetesmemberikan `IPv4` alamat ke Pods dan layanan, secara default. Sebelum memutuskan untuk menggunakan `IPv6` keluarga, pastikan bahwa Anda sudah terbiasa dengan semua pertimbangan dan persyaratan dalamthe section called “Persyaratan dan pertimbangan VPC”, the section called “Persyaratan dan pertimbangan subnet”the section called “Persyaratan grup keamanan”, dan the section called “IPv6” topik. Jika Anda memilih `IPv6` keluarga, Anda tidak dapat menentukan rentang alamat Kubernetes untuk menetapkan alamat `IPv6` layanan dari seperti yang Anda bisa untuk `IPv4` keluarga. Kubernetesmenetapkan alamat layanan dari rentang alamat lokal yang unik (`fc00::/7`).

AWS Management Console

Untuk membuat klaster Anda

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih Add cluster dan kemudian pilih Create.
3. Pada halaman Configure cluster, masukkan bidang berikut:
   - Nama — Nama untuk klaster Anda. Itu harus unik dalam diri Anda Akun AWS. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter. Nama harus unik di dalam Wilayah AWS dan Akun AWS tempat Anda membuat cluster.
   - Kubernetesversi — Versi yang Kubernetes akan digunakan untuk cluster Anda. Sebaiknya pilih versi terbaru, kecuali jika Anda memerlukan versi sebelumnya.
   - Peran layanan klaster — Pilih peran IAM klaster Amazon EKS yang Anda buat untuk memungkinkan bidang Kubernetes kontrol mengelola AWS sumber daya atas nama Anda.
   - Enkripsi rahasia — (Opsional) Pilih untuk mengaktifkan enkripsi Kubernetes rahasia rahasia menggunakan kunci KMS. Anda juga dapat mengaktifkan ini setelah Anda membuat cluster Anda. Sebelum Anda mengaktifkan kemampuan ini, pastikan Anda sudah terbiasa dengan informasi di dalamnyaMengaktifkan enkripsi rahasia pada cluster yang ada.
   - Tanda — (Opsional) Tambahkan tanda apapun ke klaster Anda. Untuk informasi selengkapnya, lihat Menandai sumber daya Amazon EKS Anda.

Setelah selesai dengan halaman ini, pilih Berikutnya.

4. Pada halaman Tentukan jaringan, pilih nilai untuk kolom berikut:

   - VPC — Pilih VPC yang sudah ada yang memenuhi persyaratan Amazon EKS VPC untuk membuat klaster Anda. Sebelum memilih VPC, kami sarankan Anda memahami semua persyaratan dan pertimbangan di dalamnya. Persyaratan dan pertimbangan Amazon EKS VPC dan subnet Anda tidak dapat mengubah VPC mana yang ingin Anda gunakan setelah pembuatan cluster. Jika tidak ada VPC yang terdaftar, maka Anda harus membuatnya terlebih dahulu. Untuk informasi selengkapnya, lihat Membuat VPC untuk klaster Amazon EKS Anda.
   - Subnet — Secara default, semua subnet yang tersedia di VPC yang ditentukan di bidang sebelumnya telah dipilih sebelumnya. Anda harus memilih setidaknya dua.

     Subnet yang Anda pilih harus memenuhi persyaratan subnet Amazon EKS. Sebelum memilih subnet, sebaiknya Anda terbiasa dengan semua persyaratan dan pertimbangan VPC Amazon EKS dan subnet.

     Grup keamanan — (Opsional) Tentukan satu atau beberapa grup keamanan yang ingin Anda kaitkan Amazon EKS ke antarmuka jaringan yang dibuatnya.

     Apakah Anda memilih grup keamanan atau tidak, Amazon EKS membuat grup keamanan yang memungkinkan komunikasi antara cluster dan VPC Anda. Amazon EKS mengaitkan grup keamanan ini, dan apa pun yang Anda pilih, ke antarmuka jaringan yang dibuatnya. Untuk informasi selengkapnya tentang grup keamanan klaster yang dibuat Amazon EKS, lihatthe section called “Persyaratan grup keamanan”. Anda dapat mengubah aturan di grup keamanan klaster yang dibuat Amazon EKS.
   - Pilih keluarga alamat IP cluster — Anda dapat memilih IPv4 dan IPv6.

     Kubernetesmemberikan `IPv4` alamat ke Pods dan layanan, secara default. Sebelum memutuskan untuk menggunakan `IPv6` keluarga, pastikan bahwa Anda sudah terbiasa dengan semua pertimbangan dan persyaratan dalamthe section called “Persyaratan dan pertimbangan VPC”, the section called “Persyaratan dan pertimbangan subnet”the section called “Persyaratan grup keamanan”, dan the section called “IPv6” topik. Jika Anda memilih `IPv6` keluarga, Anda tidak dapat menentukan rentang alamat Kubernetes untuk menetapkan alamat `IPv6` layanan dari seperti yang Anda bisa untuk `IPv4` keluarga. Kubernetesmenetapkan alamat layanan dari rentang alamat lokal yang unik (`fc00::/7`).

- (Opsional) Pilih Konfigurasi rentang alamat IP Kubernetes Layanan dan tentukan **IPv4**rentang Layanan.

  Menentukan jangkauan Anda sendiri dapat membantu mencegah konflik antara Kubernetes layanan dan jaringan lain yang diintip atau terhubung ke VPC Anda. Masukkan rentang dalam notasi CIDR. Misalnya: `10.2.0.0/16`.

  Blok CIDR harus memenuhi persyaratan berikut:

  - Berada dalam salah satu rentang berikut:`10.0.0.0/8`,`172.16.0.0/12`, atau`192.168.0.0/16`.
  - Memiliki ukuran minimum `/24` dan ukuran maksimum`/12`.
  - Tidak tumpang tindih dengan kisaran VPC untuk sumber daya Amazon EKS Anda.

  Anda hanya dapat menentukan opsi ini saat menggunakan keluarga `IPv4` alamat dan hanya pada pembuatan cluster. Jika Anda tidak menentukan ini, maka Kubernetes tetapkan alamat IP layanan dari blok `10.100.0.0/16` atau `172.20.0.0/16` CIDR.

- Untuk akses endpoint Cluster, pilih opsi. Setelah cluster Anda dibuat, Anda dapat mengubah opsi ini. Sebelum memilih opsi non-default, pastikan untuk membiasakan diri dengan opsi dan implikasinya. Untuk informasi selengkapnya, lihat Kendali akses titik akhir klaster Amazon EKS.

  Setelah selesai dengan halaman ini, pilih Berikutnya.

5. (Opsional) Pada halaman Konfigurasi observabilitas, pilih opsi pencatatan bidang Metrik dan Kontrol mana yang akan diaktifkan. Secara default, setiap jenis log dimatikan.

   - Untuk informasi selengkapnya tentang opsi Prometheus metrik, lihatAktifkan Prometheus metrik saat membuat klaster.
   - Untuk informasi selengkapnya tentang opsi Pencatatan bidang kontrol, lihatPencatatan bidang kendali Amazon EKS.

   Setelah selesai dengan halaman ini, pilih Berikutnya.

6. Pada halaman Pilih add-on, pilih add-on yang ingin Anda tambahkan ke cluster Anda. Anda dapat memilih add-on dan AWS Marketplace add-on Amazon EKS sebanyak yang Anda butuhkan. Jika AWS Marketplace add-on yang ingin Anda instal tidak terdaftar, Anda dapat mencari AWS Marketplace add-on yang tersedia dengan memasukkan teks di kotak pencarian. Anda juga dapat mencari berdasarkan kategori, vendor, atau model harga dan

kemudian memilih add-on dari hasil pencarian. Setelah selesai dengan halaman ini, pilih Berikutnya.

7. Pada halaman Konfigurasi pengaturan add-on yang dipilih, pilih versi yang ingin Anda instal. Anda selalu dapat memperbarui ke versi yang lebih baru setelah pembuatan cluster. Anda dapat memperbarui konfigurasi setiap add-on setelah pembuatan cluster. Untuk informasi selengkapnya tentang mengonfigurasi add-on, lihat. Memperbarui add-on Setelah selesai dengan halaman ini, pilih Berikutnya.
8. Pada halaman Tinjau dan buat, tinjau informasi yang Anda masukkan atau pilih pada halaman sebelumnya. Jika Anda perlu melakukan perubahan, pilih Edit. Saat Anda puas, pilih Buat. Bidang Status menunjukkan CREATING saat cluster disediakan.

> **Note**
>
> Mungkin error akan terjadi karena salah satu Availability Zone dalam permintaan Anda tidak memiliki kapasitas yang cukup untuk membuat klaster Amazon EKS. Jika hal ini terjadi, output galat berisi Availability Zones yang dapat mendukung klaster baru. Cobalah untuk kembali membuat klaster dengan setidaknya dua subnet yang terletak di Availability Zones yang didukung untuk akun Anda. Untuk informasi selengkapnya, lihat Kapasitas tidak mencukupi.

Penyediaan klaster memerlukan waktu beberapa menit.

AWS CLI

Untuk membuat klaster Anda

1. Buat cluster Anda dengan perintah berikut. Sebelum menjalankan perintah, buat penggantian berikut:
   - Ganti *region-code* dengan Wilayah AWS yang Anda inginkan untuk membuat cluster Anda.
   - Ganti *my-cluster* dengan nama untuk cluster Anda. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter. Nama harus unik di dalam Wilayah AWS dan Akun AWS tempat Anda membuat cluster.
   - Ganti *1.29* dengan versi yang didukung Amazon EKS.

- Ganti *111122223333* dengan ID akun Anda dan *myAmazonEKSClusterRole* dengan nama peran IAM cluster Anda.
- Ganti nilai untuk `subnetIds` Anda sendiri. Anda juga dapat menambahkan ID tambahan. Anda harus menentukan setidaknya dua ID subnet.

  Subnet yang Anda pilih harus memenuhi persyaratan subnet Amazon EKS. Sebelum memilih subnet, sebaiknya Anda terbiasa dengan semua persyaratan dan pertimbangan VPC Amazon EKS dan subnet.
- Jika Anda tidak ingin menentukan ID grup keamanan, hapus `,securityGroupIds=sg-`*ExampleID1* dari perintah. Jika Anda ingin menentukan satu atau beberapa ID grup keamanan, ganti nilainya `securityGroupIds` dengan milik Anda sendiri. Anda juga dapat menambahkan ID tambahan.

  Apakah Anda memilih grup keamanan atau tidak, Amazon EKS membuat grup keamanan yang memungkinkan komunikasi antara cluster dan VPC Anda. Amazon EKS mengaitkan grup keamanan ini, dan apa pun yang Anda pilih, ke antarmuka jaringan yang dibuatnya. Untuk informasi selengkapnya tentang grup keamanan klaster yang dibuat Amazon EKS, lihatthe section called “Persyaratan grup keamanan”. Anda dapat mengubah aturan di grup keamanan klaster yang dibuat Amazon EKS.

```
aws eks create-cluster --region region-code --name my-cluster --kubernetes-
version 1.29 \
   --role-arn arn:aws:iam::111122223333:role/myAmazonEKSClusterRole \
   --resources-vpc-config
 subnetIds=subnet-ExampleID1,subnet-ExampleID2,securityGroupIds=sg-ExampleID1
```

> **Note**
>
> Mungkin error akan terjadi karena salah satu Availability Zone dalam permintaan Anda tidak memiliki kapasitas yang cukup untuk membuat klaster Amazon EKS. Jika hal ini terjadi, output galat berisi Availability Zones yang dapat mendukung klaster baru. Cobalah untuk kembali membuat klaster dengan setidaknya dua subnet yang terletak di Availability Zones yang didukung untuk akun Anda. Untuk informasi selengkapnya, lihat Kapasitas tidak mencukupi.

Pengaturan opsional

Berikut ini adalah pengaturan opsional yang, jika diperlukan, harus ditambahkan ke perintah sebelumnya. Anda hanya dapat mengaktifkan opsi ini saat membuat cluster, bukan setelahnya.

- Jika Anda ingin menentukan blok IPv4 Classless Inter-domain Routing (CIDR) mana yang Kubernetes memberikan alamat IP layanan, Anda harus menentukannya dengan menambahkan ke perintah berikut. **--kubernetes-network-config serviceIpv4Cidr=*CIDR block***

  Menentukan jangkauan Anda sendiri dapat membantu mencegah konflik antara Kubernetes layanan dan jaringan lain yang diintip atau terhubung ke VPC Anda. Masukkan rentang dalam notasi CIDR. Misalnya: 10.2.0.0/16.

  Blok CIDR harus memenuhi persyaratan berikut:

  - Berada dalam salah satu rentang berikut:10.0.0.0/8,172.16.0.0/12, atau192.168.0.0/16.
  - Memiliki ukuran minimum /24 dan ukuran maksimum/12.
  - Tidak tumpang tindih dengan kisaran VPC untuk sumber daya Amazon EKS Anda.

  Anda hanya dapat menentukan opsi ini saat menggunakan keluarga IPv4 alamat dan hanya pada pembuatan cluster. Jika Anda tidak menentukan ini, maka Kubernetes tetapkan alamat IP layanan dari blok 10.100.0.0/16 atau 172.20.0.0/16 CIDR.

- Jika Anda membuat klaster dan ingin klaster menetapkan IPv6 alamat Pods dan layanan, bukan IPv4 alamat, tambahkan **--kubernetes-network-config ipFamily=ipv6** ke perintah berikut.

  Kubernetesmemberikan IPv4 alamat ke Pods dan layanan, secara default. Sebelum memutuskan untuk menggunakan IPv6 keluarga, pastikan bahwa Anda sudah terbiasa dengan semua pertimbangan dan persyaratan dalamthe section called “Persyaratan dan pertimbangan VPC”, the section called “Persyaratan dan pertimbangan subnet”the section called “Persyaratan grup keamanan”, dan the section called “IPv6” topik. Jika Anda memilih IPv6 keluarga, Anda tidak dapat menentukan rentang alamat Kubernetes untuk menetapkan alamat IPv6 layanan dari seperti yang Anda bisa untuk IPv4 keluarga. Kubernetesmenetapkan alamat layanan dari rentang alamat lokal yang unik (fc00::/7).

2. Dibutuhkan beberapa menit untuk menyediakan cluster. Anda dapat melakukan kueri status klaster Anda dengan perintah berikut.

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

Jangan lanjutkan ke langkah berikutnya sampai output yang dikembalikan`ACTIVE`.

3. Jika Anda membuat cluster Anda menggunakan`eksctl`, maka Anda dapat melewati langkah ini. Ini karena `eksctl` sudah menyelesaikan langkah ini untuk Anda. Aktifkan `kubectl` untuk berkomunikasi dengan cluster Anda dengan menambahkan konteks baru ke `kubectl config` file. Untuk informasi selengkapnya tentang cara membuat dan memperbarui file, lihatMembuat atau memperbarui `kubeconfig` file untuk klaster Amazon EKS.

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

Contoh output adalah sebagai berikut.

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

4. Konfirmasikan komunikasi dengan cluster Anda dengan menjalankan perintah berikut.

```
kubectl get svc
```

Contoh output adalah sebagai berikut.

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

5. (Disarankan) Untuk menggunakan beberapa add-on Amazon EKS, atau untuk mengaktifkan Kubernetes beban kerja individual agar memiliki izin tertentu AWS Identity and Access Management (IAM), buat penyedia IAM OpenID Connect (OIDC) untuk klaster Anda. Anda hanya perlu membuat OIDC penyedia IAM untuk cluster Anda sekali. Untuk mempelajari selengkapnya tentang add-on Amazon EKS, lihat Add-on Amazon EKS. Untuk mempelajari selengkapnya tentang penugasa izin IAM tertentu untuk beban kerja Anda, lihat IAM role untuk akun layanan.

6. (Disarankan) Konfigurasikan klaster Anda untuk Amazon VPC CNI plugin for Kubernetes plugin sebelum menerapkan node Amazon EC2 ke cluster Anda. Secara default, plugin diinstal dengan cluster Anda. Saat Anda menambahkan node Amazon EC2 ke cluster Anda, plugin secara otomatis diterapkan ke setiap node Amazon EC2 yang Anda tambahkan. Plugin mengharuskan Anda untuk melampirkan salah satu kebijakan IAM berikut ke peran IAM:

   `AmazonEKS_CNI_Policy`kebijakan IAM terkelola

   Jika cluster Anda menggunakan `IPv4` keluarga

   Kebijakan IAM yang Anda buat

   Jika cluster Anda menggunakan `IPv6` keluarga

   Peran IAM yang Anda lampirkan kebijakan dapat menjadi peran IAM node, atau peran khusus yang hanya digunakan untuk plugin. Kami merekomendasikan untuk melampirkan kebijakan untuk peran ini. Untuk informasi selengkapnya tentang membuat peran, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA) atauIAM role simpul Amazon EKS.

7. Jika Anda menerapkan cluster Anda menggunakan AWS Management Console, Anda dapat melewati langkah ini. Ini AWS Management Console menyebarkan add-on Amazon VPC CNI plugin for Kubernetes,CoreDNS, dan `kube-proxy` Amazon EKS, secara default.

   Jika Anda menerapkan klaster menggunakan salah satu `eksctl` atau AWS CLI, maka add-on Amazon VPC CNI plugin for Kubernetes,CoreDNS, dan yang `kube-proxy` dikelola sendiri akan diterapkan. Anda dapat memigrasikan add-on Amazon VPC CNI plugin for Kubernetes,CoreDNS, dan `kube-proxy` dikelola sendiri yang diterapkan dengan klaster Anda ke add-on Amazon EKS. Untuk informasi selengkapnya, lihat Add-on Amazon EKS.

8. (Opsional) Jika Anda belum melakukannya, Anda dapat mengaktifkan Prometheus metrik untuk klaster Anda. Untuk informasi selengkapnya, lihat Membuat scraper di Amazon Managed Service for Prometheus User Guide.

9. Jika Anda mengaktifkan Prometheus metrik, Anda harus mengaturnya `aws-auth ConfigMap` untuk memberikan izin dalam cluster scraper. Untuk informasi selengkapnya, lihat Mengonfigurasi klaster Amazon EKS Anda di Panduan Pengguna Layanan Terkelola Amazon untuk Prometheus.

10. Jika Anda berencana untuk menerapkan beban kerja ke klaster yang menggunakan volume Amazon EBS, dan Anda membuat klaster `1.23` atau yang lebih baru, Anda harus menginstal Driver CSI Amazon EBS ke klaster sebelum menerapkan beban kerja.

Langkah selanjutnya yang disarankan:

- Prinsipal IAM yang menciptakan cluster adalah satu-satunya prinsipal yang memiliki akses ke cluster. Berikan izin kepada prinsipal IAM lainnya sehingga mereka dapat mengakses klaster Anda.
- Jika prinsipal IAM yang membuat klaster hanya memiliki izin IAM minimum yang direferensikan dalam prasyarat, maka Anda mungkin ingin menambahkan izin Amazon EKS tambahan untuk prinsipal tersebut. Untuk informasi selengkapnya tentang pemberian izin Amazon EKS kepada prinsipal IAM, lihat. Identity and access management untuk Amazon EKS
- Jika Anda ingin prinsipal IAM yang membuat cluster, atau prinsipal lainnya untuk melihat sumber daya Kubernetes di konsol Amazon EKS, berikan kepada entitas. Izin yang diperlukan
- Jika Anda ingin node dan prinsipal IAM mengakses cluster Anda dari dalam VPC Anda, aktifkan titik akhir pribadi untuk cluster Anda. Titik akhir publik diaktifkan secara default. Anda dapat menonaktifkan titik akhir publik setelah Anda mengaktifkan titik akhir pribadi, jika diinginkan. Untuk informasi selengkapnya, lihat Kendali akses titik akhir klaster Amazon EKS.
- Aktifkan enkripsi rahasia untuk cluster Anda.
- Konfigurasikan logging untuk klaster Anda.
- Tambahkan node ke cluster Anda.

Wawasan cluster

Wawasan klaster Amazon EKS memberikan rekomendasi untuk membantu Anda mengikuti praktik terbaik Amazon EKS dan Kubernetes. Setiap klaster Amazon EKS menjalani pemeriksaan otomatis dan berulang terhadap daftar wawasan yang dikuratori Amazon EKS. Pemeriksaan wawasan ini dikelola sepenuhnya oleh Amazon EKS dan menawarkan rekomendasi tentang cara mengatasi temuan apa pun.

Penggunaan wawasan cluster yang disarankan:

- Sebelum memperbarui Kubernetes versi cluster Anda, periksa wawasan cluster di konsol EKS.
- Jika klaster Anda telah mengidentifikasi masalah, tinjau dan buat perbaikan yang sesuai. Masalahnya termasuk tautan ke Amazon EKS danKubernetes.

- Setelah memperbaiki masalah, tunggu hingga wawasan cluster disegarkan. Jika semua masalah telah diselesaikan, perbarui klaster Anda.

Saat ini, Amazon EKS hanya mengembalikan wawasan yang terkait dengan kesiapan peningkatan Kubernetes versi.

Wawasan pemutakhiran mengidentifikasi kemungkinan masalah yang dapat memengaruhi peningkatan Kubernetes klaster. Ini meminimalkan upaya yang dihabiskan administrator untuk mempersiapkan peningkatan dan meningkatkan keandalan aplikasi pada versi yang lebih baru. Kubernetes Cluster secara otomatis dipindai oleh Amazon EKS terhadap daftar kemungkinan peningkatan Kubernetes versi yang memengaruhi masalah. Amazon EKS sering memperbarui daftar pemeriksaan wawasan berdasarkan ulasan perubahan yang dibuat di setiap rilis Kubernetes versi.

Wawasan peningkatan Amazon EKS mempercepat proses pengujian dan verifikasi untuk versi baru. Mereka juga memungkinkan administrator cluster dan pengembang aplikasi untuk memanfaatkan Kubernetes kemampuan terbaru dengan menyoroti masalah dan menawarkan saran remediasi. Untuk melihat daftar pemeriksaan wawasan yang dilakukan dan masalah relevan yang diidentifikasi Amazon EKS, Anda dapat menghubungi operasi Amazon EKS `ListInsights` API atau melihat di konsol Amazon EKS.

Wawasan cluster diperbarui secara berkala. Anda tidak dapat menyegarkan wawasan cluster secara manual. Jika Anda memperbaiki masalah klaster, perlu beberapa waktu untuk memperbarui wawasan cluster. Untuk menentukan apakah perbaikan berhasil, bandingkan waktu perubahan diterapkan dengan “waktu penyegaran terakhir” dari wawasan cluster.

Lihat wawasan cluster (Konsol)

Untuk melihat wawasan kluster Amazon EKS:

a. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.

b. Dari daftar cluster, pilih nama cluster Amazon EKS yang ingin Anda lihat wawasannya.

c. Pilih tab Upgrade Insights.

d. Pada halaman Upgrade Insights Anda akan melihat bidang-bidang berikut:

- Nama — Pemeriksaan yang dilakukan oleh Amazon EKS terhadap cluster.
- Status wawasan — Wawasan dengan status “Kesalahan” biasanya berarti versi yang terkena dampak adalah N+1 dari Kubernetes versi cluster saat ini, sedangkan status “Peringatan” berarti wawasan berlaku untuk Kubernetes versi N+2 yang akan datang atau lebih. Wawasan dengan

status “Passing” berarti Amazon EKS tidak menemukan masalah apa pun yang terkait dengan pemeriksaan wawasan ini di klaster Anda. Status wawasan “Tidak Diketahui” berarti Amazon EKS tidak dapat menentukan apakah klaster Anda terpengaruh oleh pemeriksaan wawasan ini.

- Versi — Kubernetes Versi yang diperiksa wawasan untuk kemungkinan masalah.
- Waktu penyegaran terakhir (UTC- 5:00) — Waktu status wawasan terakhir disegarkan untuk cluster ini.
- Waktu transisi terakhir (UTC- 5:00) - Waktu status wawasan ini terakhir berubah.
- Deskripsi — Informasi dari pemeriksaan wawasan, yang mencakup peringatan dan tindakan yang direkomendasikan untuk perbaikan.

Lihat wawasan cluster ()AWS CLI

Untuk melihat wawasan kluster Amazon EKS:

a. Tentukan cluster mana yang ingin Anda periksa untuk wawasan. Perintah berikut mencantumkan wawasan untuk cluster tertentu. Buat modifikasi berikut pada perintah sesuai kebutuhan dan kemudian jalankan perintah yang dimodifikasi:

   - Ganti *region-code* dengan kode untuk Anda Wilayah AWS.
   - Ganti *my-cluster* dengan nama klaster Anda.

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

   Contoh output adalah sebagai berikut.

```
{
    "insights": [
        {
            "category": "UPGRADE_READINESS",
            "name": "Deprecated APIs removed in Kubernetes v1.29",
            "insightStatus": {
                "status": "PASSING",
                "reason": "No deprecated API usage detected within the last 30 days."
            },
            "kubernetesVersion": "1.29",
            "lastTransitionTime": 1698774710.0,
            "lastRefreshTime": 1700157422.0,
            "id": "123e4567-e89b-42d3-a456-579642341238",
```

```
            "description": "Checks for usage of deprecated APIs that are scheduled
 for removal in Kubernetes v1.29. Upgrading your cluster before migrating to the
 updated APIs supported by v1.29 could cause application impact."
        }
    ]
}
```

b. Untuk informasi deskriptif tentang wawasan, jalankan perintah berikut. Buat modifikasi berikut pada perintah sesuai kebutuhan dan kemudian jalankan perintah yang dimodifikasi:

- Ganti *region-code* dengan kode untuk Anda Wilayah AWS.
- Ganti *123e4567-e89b-42d3-a456-579642341238* dengan ID wawasan yang diambil dari daftar wawasan cluster.
- Ganti *my-cluster* dengan nama klaster Anda.

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

Contoh output adalah sebagai berikut.

```
{
    "insight": {
        "category": "UPGRADE_READINESS",
        "additionalInfo": {
            "EKS update cluster documentation": "https://docs.aws.amazon.com/eks/
latest/userguide/update-cluster.html",
            "Kubernetes v1.29 deprecation guide": "https://kubernetes.io/docs/
reference/using-api/deprecation-guide/#v1-29"
        },
        "name": "Deprecated APIs removed in Kubernetes v1.29",
        "insightStatus": {
            "status": "PASSING",
            "reason": "No deprecated API usage detected within the last 30 days."
        },
        "kubernetesVersion": "1.29",
        "recommendation": "Update manifests and API clients to use newer Kubernetes
 APIs if applicable before upgrading to Kubernetes v1.29.",
        "lastTransitionTime": 1698774710.0,
        "lastRefreshTime": 1700157422.0,
        "categorySpecificSummary": {
            "deprecationDetails": [
                {
```

```
                    "usage": "/apis/flowcontrol.apiserver.k8s.io/v1beta2/
flowschemas",
                    "replacedWith": "/apis/flowcontrol.apiserver.k8s.io/v1beta3/
flowschemas",
                    "stopServingVersion": "1.29",
                    "clientStats": [],
                    "startServingReplacementVersion": "1.26"
                },
                {
                    "usage": "/apis/flowcontrol.apiserver.k8s.io/v1beta2/
prioritylevelconfigurations",
                    "replacedWith": "/apis/flowcontrol.apiserver.k8s.io/v1beta3/
prioritylevelconfigurations",
                    "stopServingVersion": "1.29",
                    "clientStats": [],
                    "startServingReplacementVersion": "1.26"
                }
            ]
        },
        "id": "f6a11fe4-77f7-48c6-8326-9a13f022ecb3",
        "resources": [],
        "description": "Checks for usage of deprecated APIs that are scheduled for
 removal in Kubernetes v1.29. Upgrading your cluster before migrating to the updated
 APIs supported by v1.29 could cause application impact."
    }
}
```

Memperbarui Kubernetes versi cluster Amazon EKS

Ketika Kubernetes versi baru tersedia di Amazon EKS, Anda dapat memperbarui cluster Amazon EKS Anda ke versi terbaru.

> **Important**
>
> Setelah Anda memutakhirkan cluster, Anda tidak dapat menurunkan versi ke versi sebelumnya. Kami menyarankan agar, sebelum Anda memperbarui ke Kubernetes versi baru, Anda meninjau informasi di KubernetesVersi Amazon EKS dan juga meninjau langkah-langkah pembaruan dalam topik ini.

KubernetesVersi baru terkadang memperkenalkan perubahan signifikan. Oleh karena itu, kami menyarankan Anda menguji perilaku aplikasi Anda terhadap Kubernetes versi baru sebelum memperbarui cluster produksi Anda. Anda dapat melakukannya dengan membuat alur kerja integrasi berkelanjutan untuk menguji perilaku aplikasi Anda sebelum pindah ke Kubernetes versi baru.

Proses pembaruan terdiri dari Amazon EKS meluncurkan node server API baru dengan Kubernetes versi yang diperbarui untuk menggantikan yang sudah ada. Amazon EKS melakukan pemeriksaan kondisi infrastruktur dan kesiapan standar untuk lalu lintas jaringan pada simpul baru untuk memverifikasi bahwa hal tersebut bekerja seperti yang diharapkan. Namun, setelah Anda memulai upgrade cluster, Anda tidak dapat menjeda atau menghentikannya. Jika salah satu pemeriksaan ini gagal, Amazon EKS mengembalikan penerapan infrastruktur, dan klaster Anda tetap pada versi sebelumnyaKubernetes. Aplikasi yang berjalan tidak terpengaruh, dan klaster Anda tidak pernah dibiarkan dalam keadaan non-deterministik atau tidak dapat dipulihkan. Amazon EKS secara teratur mendukung semua klaster terkelola, dan mekanisme yang ada untuk memulihkan klaster jika diperlukan. Kami terus mengevaluasi dan meningkatkan proses manajemen Kubernetes infrastruktur kami.

Untuk memperbarui cluster, Amazon EKS memerlukan hingga lima alamat IP yang tersedia dari subnet yang Anda tentukan saat membuat klaster. Amazon EKS membuat antarmuka jaringan elastis klaster baru (antarmuka jaringan) di salah satu subnet yang Anda tentukan. Antarmuka jaringan dapat dibuat dalam subnet yang berbeda dari antarmuka jaringan yang ada, jadi pastikan bahwa aturan grup keamanan Anda mengizinkan komunikasi cluster yang diperlukan untuk setiap subnet yang Anda tentukan saat Anda membuat cluster Anda. Jika salah satu subnet yang Anda tentukan saat membuat klaster tidak ada, tidak memiliki cukup alamat IP yang tersedia, atau tidak memiliki aturan grup keamanan yang memungkinkan komunikasi cluster yang diperlukan, maka pembaruan dapat gagal.

> **Note**
>
> Untuk memastikan bahwa titik akhir server API untuk klaster Anda selalu dapat diakses, Amazon EKS menyediakan bidang Kubernetes kontrol yang sangat tersedia dan melakukan pembaruan bergulir instance server API selama operasi pembaruan. Untuk memperhitungkan perubahan alamat IP instance server API yang mendukung titik akhir server Kubernetes API Anda, Anda harus memastikan bahwa klien server API Anda mengelola koneksi ulang secara efektif. Versi terbaru `kubectl` dan pustaka Kubernetes klien yang didukung secara resmi, lakukan proses penyambungan kembali ini secara transparan.

Perbarui Kubernetes versi untuk kluster Amazon EKS Anda

Untuk memperbarui Kubernetes versi untuk klaster Anda

1. Bandingkan Kubernetes versi bidang kontrol cluster Anda dengan Kubernetes versi node Anda.

   - Dapatkan Kubernetes versi pesawat kontrol cluster Anda.

     ```
     kubectl version
     ```

   - Dapatkan Kubernetes versi node Anda. Perintah ini mengembalikan semua simpul Amazon EC2 dan Fargate swakelola dan terkelola. Setiap Fargate terdaftar sebagai Pod simpulnya sendiri.

     ```
     kubectl get nodes
     ```

   Sebelum memperbarui bidang kontrol Anda ke Kubernetes versi baru, pastikan bahwa versi Kubernetes minor dari node terkelola dan node Fargate di cluster Anda sama dengan versi bidang kontrol Anda. Misalnya, jika bidang kontrol Anda menjalankan versi `1.28` dan salah satu node Anda menjalankan versi`1.27`, maka Anda harus memperbarui node Anda ke versi `1.28` sebelum memperbarui bidang kontrol Anda ke 1,29. Kami juga merekomendasikan agar Anda memperbarui simpul swakelola ke versi yang sama dengan versi bidang pengendalian Anda sebelum memperbarui bidang pengendalian itu. Lihat informasi yang lebih lengkap di Memperbarui grup simpul terkelola dan Pembaruan simpul yang dikelola sendiri. Jika Anda memiliki node Fargate dengan versi minor lebih rendah dari versi bidang kontrol, pertama-tama hapus Pod yang diwakili oleh node. Kemudian perbarui pesawat kontrol Anda. Yang tersisa Pods akan diperbarui ke versi baru setelah Anda menerapkannya kembali.

2. Jika Kubernetes versi yang awalnya Anda gunakan untuk menggunakan cluster Anda adalah Kubernetes `1.25` atau yang lebih baru, lewati langkah ini.

   Secara default, pengontrol penerimaan kebijakan Pod keamanan diaktifkan di kluster Amazon EKS. Sebelum memperbarui klaster Anda, pastikan bahwa kebijakan Pod keamanan yang tepat sudah ada. Ini untuk menghindari potensi masalah keamanan. Anda dapat memeriksa kebijakan default dengan **`kubectl get psp eks.privileged`** perintah.

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

Jika Anda menerima kesalahan berikut, lihat Kebijakan Pod keamanan default Amazon EKS sebelum melanjutkan.

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

3. Jika Kubernetes versi yang awalnya Anda gunakan untuk menggunakan cluster Anda adalah Kubernetes `1.18` atau yang lebih baru, lewati langkah ini.

   Anda mungkin perlu menghapus istilah yang dihentikan dari CoreDNS manifes Anda.

   a. Periksa untuk melihat apakah CoreDNS manifes Anda memiliki garis yang hanya memiliki kata`upstream`.

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

   Jika tidak ada output yang dikembalikan, ini berarti manifes Anda tidak memiliki baris. Jika ini masalahnya, lompat ke langkah berikutnya. Jika kata `upstream` dikembalikan, hapus baris.

   b. Hapus baris di dekat bagian atas file yang hanya memiliki kata `upstream` dalam file configmap. Jangan mengubah apa pun dalam file. Setelah baris dihapus, simpan perubahannya.

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

4. Perbarui cluster Anda menggunakan `eksctl` AWS Management Console, the, atau AWS CLI.

> **Important**
>
> - Jika Anda memperbarui ke versi `1.23` dan menggunakan volume Amazon EBS di klaster, Anda harus menginstal driver Amazon EBS CSI di klaster sebelum memperbarui klaster ke versi `1.23` untuk menghindari gangguan beban kerja. Lihat informasi yang lebih lengkap di Kubernetes1.23 dan Driver CSI Amazon EBS.
> - Kubernetes `1.24` dan kemudian digunakan `containerd` sebagai runtime container default. Jika Anda beralih ke `containerd` runtime dan sudah Fluentd mengonfigurasinyaContainer Insights, Anda harus bermigrasi Fluentd ke Fluent

Bit sebelum memperbarui klaster Anda. FluentdParser dikonfigurasi untuk hanya mengurai pesan log dalam format JSON. Tidak sepertidockerd, runtime containerd kontainer memiliki pesan log yang tidak dalam format JSON. Jika Anda tidak bermigrasi keFluent Bit, beberapa Fluentd's parser yang dikonfigurasi akan menghasilkan sejumlah besar kesalahan di dalam wadah. Fluentd Untuk informasi selengkapnya tentang migrasi, lihat Mengatur Fluent Bit sebagai DaemonSet untuk mengirim log ke CloudWatch Log.

- Karena Amazon EKS menjalankan bidang pengendali yang banyak tersedia, Anda dapat memperbarui hanya satu versi minor pada satu waktu. Untuk informasi selengkapnya tentang persyaratan ini, lihat KubernetesKebijakan Dukungan Versi dan Versi Skew. Asumsikan bahwa versi cluster Anda saat ini adalah versi 1.27 dan Anda ingin memperbaruinya ke versi1.29. Anda harus terlebih dahulu memperbarui 1.27 klaster versi Anda ke versi 1.28 dan kemudian memperbarui 1.28 klaster versi Anda ke versi1.29.
- Tinjau versi miring antara Kubernetes kube-apiserver dan kubelet pada node Anda.
  - Mulai dari Kubernetes versi1.28, kubelet mungkin hingga tiga versi minor yang lebih tua darikube-apiserver. Lihat Kebijakan miring versi upstream Kubernetes.
  - Jika kubelet pada node terkelola dan Fargate Anda ada di Kubernetes versi 1.25 atau yang lebih baru, Anda dapat memperbarui cluster Anda hingga tiga versi ke depan tanpa memperbarui versi. kubelet Misalnya, jika versi aktif kubelet1.25, Anda dapat memperbarui versi kluster Amazon EKS Anda dari 1.25 ke1.26, ke1.27, dan ke 1.28 sementara versi kubelet tetap ada1.25.
  - Jika kubelet pada node terkelola dan Fargate Anda ada di Kubernetes versi 1.24 atau lebih lama, mungkin hanya hingga dua versi minor yang lebih lama dari versi. kube-apiserver Dengan kata lain, jika versi kubelet is 1.24 atau lebih lama, Anda hanya dapat memperbarui cluster Anda hingga dua versi ke depan. Misalnya, jika versi on1.21, Anda dapat memperbarui versi kluster Amazon EKS dari 1.21 ke 1.221.23, dan ke, tetapi Anda tidak akan dapat memperbarui cluster 1.24 saat kubelet masih aktif1.21. kubelet
- Sebagai praktik terbaik sebelum memulai pembaruan, pastikan bahwa kubelet pada node Anda berada pada Kubernetes versi yang sama dengan bidang kontrol Anda.
- Jika cluster Anda dikonfigurasi dengan versi Amazon VPC CNI plugin for Kubernetes yang lebih awal dari1.8.0, maka kami sarankan Anda memperbarui plugin ke versi

terbaru sebelum memperbarui cluster Anda. Untuk memperbarui plugin, lihatBekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS.

- Jika Anda memperbarui klaster ke versi `1.25` atau yang lebih baru dan AWS Load Balancer Controller menerapkan klaster Anda, perbarui pengontrol ke versi `2.4.7` atau yang lebih baru sebelum memperbarui versi cluster Anda ke`1.25`. Untuk informasi selengkapnya, lihat catatan Kubernetes1,25 rilis.

eksctl

Prosedur ini membutuhkan `eksctl` versi `0.175.0` atau yang lebih baru. Anda dapat memeriksa versi Anda dengan perintah berikut:

```
eksctl version
```

Untuk petunjuk tentang cara menginstal dan memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Perbarui Kubernetes versi pesawat kontrol Amazon EKS Anda. Ganti *`my-cluster`* dengan nama klaster Anda. Ganti *`1.29`* dengan nomor versi yang didukung Amazon EKS yang ingin Anda perbarui klaster Anda. Untuk daftar nomor versi yang didukung, lihatKubernetesVersi Amazon EKS.

```
eksctl upgrade cluster --name my-cluster --version 1.29 --approve
```

Pembaruan memerlukan waktu beberapa menit.

AWS Management Console

a. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
b. Pilih nama kluster Amazon EKS untuk diperbarui dan pilih Perbarui versi cluster.
c. Untuk Kubernetesversi, pilih versi untuk memperbarui klaster Anda dan pilih Perbarui.
d. Untuk nama Cluster, masukkan nama klaster Anda dan pilih Konfirmasi.

   Pembaruan memerlukan waktu beberapa menit.

AWS CLI

a. Perbarui cluster Amazon EKS Anda dengan AWS CLI perintah berikut. Ganti *example values* dengan milik Anda sendiri. Ganti *1.29* dengan nomor versi yang didukung Amazon EKS yang ingin Anda perbarui klaster Anda. Untuk daftar nomor versi yang didukung, lihatKubernetesVersi Amazon EKS.

```
aws eks update-cluster-version --region region-code --name my-cluster --
kubernetes-version 1.29
```

Contoh output adalah sebagai berikut.

```
{
    "update": {
        "id": "b5f0ba18-9a87-4450-b5a0-825e6e84496f",
        "status": "InProgress",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "1.29"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.1"
            }
        ],
[...]
        "errors": []
    }
}
```

b. Pantau status pembaruan klaster Anda dengan perintah berikut. Gunakan nama klaster dan perbarui ID yang telah dikembalikan dengan perintah sebelumnya. Ketika `Successful` status ditampilkan, pembaruan selesai. Pembaruan memerlukan waktu beberapa menit.

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

Contoh output adalah sebagai berikut.

```
{
    "update": {
        "id": "b5f0ba18-9a87-4450-b5a0-825e6e84496f",
        "status": "Successful",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "1.29"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.1"
            }
        ],
[...]
        "errors": []
    }
}
```

5. Setelah pembaruan klaster Anda selesai, perbarui node Anda ke versi Kubernetes minor yang sama dengan kluster Anda yang diperbarui. Lihat informasi yang lebih lengkap di Pembaruan simpul yang dikelola sendiri dan Memperbarui grup simpul terkelola. Setiap baru Pods yang diluncurkan di Fargate memiliki `kubelet` versi yang cocok dengan versi cluster Anda. Fargate yang ada Pods tidak berubah.
6. (Opsional) Jika Anda menerapkan Kubernetes Cluster Autoscaler ke klaster Anda sebelum memperbarui klaster, perbarui Cluster Autoscaler ke versi terbaru yang cocok dengan versi Kubernetes mayor dan minor yang Anda perbarui.

   a. Buka halaman rilis Cluster Autoscaler di browser web dan temukan versi Cluster Autoscaler terbaru yang cocok dengan versi mayor dan minor klaster Anda. Kubernetes Misalnya, jika Kubernetes versi klaster Anda `1.29` menemukan rilis Cluster Autoscaler terbaru yang dimulai dengan. `1.29` Catat nomor versi semantik (`1.29.n`, misalnya) untuk rilis yang akan digunakan pada langkah berikutnya.

   b. Atur citra label Klaster Autoscaler ke versioning yang Anda simpan pada langkah sebelumnya menggunakan perintah berikut. Jika perlu, ganti *`1.29.n`* dengan nilai Anda sendiri.

```
kubectl -n kube-system set image deployment.apps/cluster-autoscaler cluster-
autoscaler=registry.k8s.io/autoscaling/cluster-autoscaler:v1.29.n
```

7. (Cluster dengan node GPU saja) Jika cluster Anda memiliki grup node dengan dukungan GPU (misalnya,`p3.2xlarge`), Anda harus memperbarui plugin perangkat NVIDIA Kubernetes DaemonSet di cluster Anda. Ganti *vX.X.X* dengan s-device-plugin versi NVIDIA/K8 yang Anda inginkan sebelum menjalankan perintah berikut.

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

8. Perbarui Amazon VPC CNI plugin for Kubernetes,CoreDNS, dan `kube-proxy` add-on. Kami menyarankan untuk memperbarui add-on ke versi minimum yang tercantum dalam token akun Layanan.

   - Jika Anda menggunakan add-on Amazon EKS, pilih Cluster di konsol Amazon EKS, lalu pilih nama cluster yang Anda perbarui di panel navigasi kiri. Pemberitahuan muncul di konsol. Mereka memberi tahu Anda bahwa versi baru tersedia untuk setiap add-on yang memiliki pembaruan yang tersedia. Untuk memperbarui add-on, pilih tab Add-ons. Di salah satu kotak untuk add-on yang memiliki pembaruan yang tersedia, pilih Perbarui sekarang, pilih versi yang tersedia, lalu pilih Perbarui.
   - Sebagai alternatif, Anda dapat menggunakan AWS CLI atau `eksctl` untuk memperbarui add-on. Untuk informasi selengkapnya, lihat Memperbarui add-on.

9. Jika perlu, perbarui versi Anda`kubectl`. Anda harus menggunakan versi `kubectl` yang tidak lebih dari satu perbedaan kecil dari bidang kendali Amazon EKS klaster Anda. Misalnya, `1.28` `kubectl` klien bekerja dengan Kubernetes`1.27`,`1.28`, dan `1.29` cluster. Anda dapat memeriksa versi yang saat ini diinstal dengan perintah berikut.

```
kubectl version --client
```

Menghapus kluster Amazon EKS

Usai menggunakan klaster Amazon EKS, Anda harus menghapus sumber daya yang terkait dengannya sehingga Anda tidak perlu mengeluarkan biaya yang tidak semestinya.

Untuk menghapus klaster yang terhubung, lihat Menderegistrasi sebuah cluster

Important

- Jika Anda memiliki layanan aktif di klaster yang terkait dengan penyeimbang beban, Anda harus menghapus layanan tersebut sebelum menghapus klaster agar penyeimbang beban terhapus dengan benar. Jika tidak, Anda dapat memiliki sumber daya orphaned di VPC yang mencegah Anda untuk dapat menghapus VPC.
- Jika Anda menerima kesalahan karena pembuat klaster telah dihapus, lihat artikel ini untuk menyelesaikannya.
- Layanan Terkelola Amazon untuk sumber daya Prometheus berada di luar siklus hidup klaster dan perlu dipertahankan secara independen dari klaster. Saat Anda menghapus klaster Anda, pastikan juga untuk menghapus pencakar yang berlaku untuk menghentikan biaya yang berlaku. Untuk informasi selengkapnya, lihat Menemukan dan menghapus pencakar di Amazon Managed Service for Prometheus User Guide.

Anda dapat menghapus cluster dengan`eksctl`, file AWS Management Console, atau file AWS CLI.

eksctl

Untuk menghapus klaster dan simpul Amazon EKS dengan **`eksctl`**

Prosedur ini membutuhkan `eksctl` versi `0.175.0` atau yang lebih baru. Anda dapat memeriksa versi Anda dengan perintah berikut:

```
eksctl version
```

Untuk petunjuk tentang cara menginstal atau meningkatkan`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

1. Buat daftar semua layanan yang berjalan di klaster Anda.

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

2. Hapus layanan apa pun yang memiliki nilai `EXTERNAL-IP` yang terkait. Layanan ini digawangi oleh penyeimbang beban Elastic Load Balancing, dan Anda harus menghapusnya Kubernetes agar penyeimbang beban dan sumber daya terkait dilepaskan dengan benar.

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

3. Hapus cluster dan node yang terkait dengan perintah berikut, ganti *prod* dengan nama cluster Anda.

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

Output:

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

AWS Management Console

Untuk menghapus cluster Amazon EKS dengan AWS Management Console

1. Buat daftar semua layanan yang berjalan di klaster Anda.

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

2. Hapus layanan apa pun yang memiliki nilai `EXTERNAL-IP` yang terkait. Layanan ini digawangi oleh penyeimbang beban Elastic Load Balancing, dan Anda harus menghapusnya Kubernetes agar penyeimbang beban dan sumber daya terkait dilepaskan dengan benar.

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

3. Hapus semua grup simpul dan profil Fargate.

   a. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.

   b. Di panel navigasi kiri, pilih Amazon EKS Clusters, lalu di daftar tab cluster, pilih nama cluster yang ingin Anda hapus.

   c. Pilih tab Compute dan pilih grup node untuk dihapus. Pilih Hapus, masukkan nama grup simpul, lalu pilih Hapus. Hapus semua grup simpul dalam klaster.

> **Note**
>
> Grup-grup simpul yang terdaftar hanyalah grup simpul dikelola.

d. Pilih Profil Fargate untuk dihapus, pilih Hapus, masukkan nama profil, lalu pilih Hapus. Hapus semua profil Fargate di klaster.

4. Hapus semua AWS CloudFormation tumpukan node yang dikelola sendiri.

   a. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.

   b. Pilih tumpukan simpul yang akan dihapus, lalu pilih Hapus.

   c. Di kotak dialog Hapus tumpukan konfirmasi, pilih Hapus tumpukan. Hapus semua tumpukan simpul swakelola di dalam klaster.

5. Hapus klaster.

   a. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.

   b. pilih cluster yang akan dihapus dan pilih Hapus.

   c. Pada layar konfirmasi hapus klaster, pilih Hapus.

6. (Opsional) Hapus tumpukan VPC AWS CloudFormation .

   a. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.

   b. Pilih tumpukan VPC yang akan dihapus, lalu pilih Hapus.

   c. Di kotak dialog Hapus tumpukan konfirmasi, pilih Hapus tumpukan.

AWS CLI

Untuk menghapus cluster Amazon EKS dengan AWS CLI

1. Buat daftar semua layanan yang berjalan di klaster Anda.

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

2. Hapus layanan apa pun yang memiliki nilai `EXTERNAL-IP` yang terkait. Layanan ini digawangi oleh penyeimbang beban Elastic Load Balancing, dan Anda harus menghapusnya Kubernetes agar penyeimbang beban dan sumber daya terkait dilepaskan dengan benar.

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

3. Hapus semua grup simpul dan profil Fargate.

   a. Buat daftar grup simpul di klaster Anda dengan perintah berikut.

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

   > **Note**
   >
   > Grup-grup simpul yang terdaftar adalah grup simpul dikelola saja.

   b. Hapus setiap grup simpul dengan perintah berikut. Hapus semua grup simpul dalam klaster.

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

   c. Buat daftar profil Fargate di klaster Anda dengan perintah berikut.

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

   d. Hapus setiap profil Fargate dengan perintah berikut. Hapus semua profil Fargate di dalam klaster.

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

4. Hapus semua AWS CloudFormation tumpukan node yang dikelola sendiri.

   a. Buat daftar AWS CloudFormation tumpukan yang tersedia dengan perintah berikut. Temukan nama templat simpul dalam output yang dihasilkan.

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

   b. Hapus setiap tumpukan node dengan perintah berikut, ganti *node-stack* dengan nama tumpukan node Anda. Hapus semua tumpukan simpul swakelola di dalam klaster.

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

5. Hapus cluster dengan perintah berikut, ganti *my-cluster* dengan nama cluster Anda.

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

6. (Opsional) Hapus tumpukan VPC AWS CloudFormation .

   a. Buat daftar AWS CloudFormation tumpukan yang tersedia dengan perintah berikut. Temukan nama templat VPC dalam output yang dihasilkan.

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

   b. Hapus tumpukan VPC dengan perintah berikut, ganti *my-vpc-stack* dengan nama tumpukan VPC Anda.

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

Kendali akses titik akhir klaster Amazon EKS

Topik ini membantu Anda mengaktifkan akses pribadi untuk titik akhir server Kubernetes API klaster Amazon EKS Anda dan membatasi, atau sepenuhnya menonaktifkan, akses publik dari internet.

Saat Anda membuat klaster baru, Amazon EKS membuat titik akhir untuk server Kubernetes API terkelola yang Anda gunakan untuk berkomunikasi dengan klaster Anda (menggunakan alat Kubernetes manajemen seperti`kubectl`). Secara default, endpoint server API ini bersifat publik ke internet, dan akses ke server API diamankan menggunakan kombinasi AWS Identity and Access Management (IAM) dan Kubernetes Role Based Access Control (RBAC) asli.

Anda dapat mengaktifkan akses pribadi ke server Kubernetes API sehingga semua komunikasi antara node dan server API tetap berada dalam VPC Anda. Anda dapat membatasi alamat IP yang dapat mengakses server API Anda dari internet, atau sepenuhnya menonaktifkan akses internet ke server API.

> **Note**
>
> Karena titik akhir ini untuk server Kubernetes API dan bukan AWS PrivateLink titik akhir tradisional untuk berkomunikasi dengan AWS API, titik akhir ini tidak muncul sebagai titik akhir di konsol VPC Amazon.

Saat Anda mengaktifkan akses privat titik akhir untuk klaster Anda, Amazon EKS membuat zona privat Route 53 yang di-hosting atas nama Anda, dan mengaitkannya dengan VPC klaster Anda. Zona privat yang di-hosting ini dikelola oleh Amazon EKS, dan tidak muncul di sumber daya Route 53 akun Anda. Agar zona privat yang di-hosting dapat merutekan lalu lintas ke server API Anda dengan benar, VPC Anda harus memiliki `enableDnsHostnames` dan `enableDnsSupport` yang diatur ke `true`, dan opsi DHCP yang diatur untuk VPC Anda harus menyertakan `AmazonProvidedDNS` dalam daftar server nama domainnya. Untuk informasi selengkapnya, lihat Memperbarui DNS dukungan untuk VPC Anda dalam Panduan Pengguna Amazon VPC.

Anda dapat menentukan persyaratan akses titik akhir server API ketika membuat klaster baru, dan dapat memperbarui akses tersebut untuk klaster kapanpun.

Memodifikasi akses titik akhir klaster

Gunakan prosedur di bagian ini untuk memodifikasi akses titik akhir untuk klaster yang sudah ada. Tabel berikut menunjukkan kombinasi akses titik akhir server API yang didukung dan perilaku terkaitnya.

Opsi akses titik akhir server API

Akses publik titik akhir	Akses privat titik akhir	Perilaku
Diaktifkan	Dinonaktifkan	• Ini adalah perilaku default untuk klaster Amazon EKS yang baru. • KubernetesPermintaan API yang berasal dari dalam VPC klaster Anda (seperti node untuk mengontro l komunikasi pesawat) meninggalkan VPC tetapi bukan jaringan Amazon. • Server API klaster Anda dapat diakses dari internet. Secara opsional, Anda dapat membatasi blok CIDR yang dapat mengakses titik akhir publik. Jika

Akses publik titik akhir	Akses privat titik akhir	Perilaku
		Anda membatasi akses ke blok CIDR tertentu, maka disarankan agar Anda juga mengaktifkan titik akhir pribadi, atau memastikan bahwa blok CIDR yang Anda tentukan menyertakan alamat tempat node dan Fargate Pods (jika Anda menggunakannya) mengakses titik akhir publik.
Diaktifkan	Diaktifkan	• KubernetesPermintaan API dalam VPC klaster Anda (seperti node untuk mengontrol komunikasi pesawat) menggunakan titik akhir VPC pribadi. • Server API klaster Anda dapat diakses dari internet. Secara opsional, Anda dapat membatasi blok CIDR yang dapat mengakses titik akhir publik.

Akses publik titik akhir	Akses privat titik akhir	Perilaku
Dinonaktifkan	Diaktifkan	• Semua lalu lintas ke server API klaster Anda harus berasal dari dalam VPC klaster Anda atau jaringan terkoneksi. • Tidak ada akses publik ke server API Anda dari internet. Perintah `kubectl` apa pun harus berasal dari dalam VPC atau jaringan yang terhubung. Untuk opsi koneksi lainnya, lihat Mengakses server API privat saja. • Titik akhir server API klaster diselesaikan oleh server DNS publik ke alamat IP privat dari VPC. Di masa lalu, titik akhir hanya dapat diselesaikan dari dalam VPC. Jika titik akhir Anda tidak menyelesaikan ke alamat IP privat dalam VPC untuk klaster yang sudah ada, Anda dapat: • Aktifkan akses publik dan kemudian nonaktifk an kembali. Anda hanya perlu melakukannya sekali untuk sebuah klaster dan titik akhir akan

Akses publik titik akhir	Akses privat titik akhir	Perilaku
		diselesaikan ke alamat IP privat sejak saat itu. • Perbarui klaster Anda.

Anda dapat memodifikasi akses endpoint server API cluster menggunakan AWS Management Console atau AWS CLI.

AWS Management Console

Untuk memodifikasi akses endpoint server API cluster Anda menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih nama klaster untuk menampilkan informasi klaster Anda.
3. Pilih tab Jaringan dan pilih Perbarui.
4. Untuk akses Pribadi, pilih apakah akan mengaktifkan atau menonaktifkan akses pribadi untuk titik akhir server Kubernetes API klaster Anda. Jika Anda mengaktifkan akses pribadi, permintaan Kubernetes API yang berasal dari dalam VPC klaster Anda menggunakan titik akhir VPC pribadi. Anda harus mengaktifkan akses privat untuk menonaktifkan akses publik.
5. Untuk akses Publik, pilih apakah akan mengaktifkan atau menonaktifkan akses publik untuk titik akhir server Kubernetes API klaster Anda. Jika Anda menonaktifkan akses publik, server Kubernetes API klaster Anda hanya dapat menerima permintaan dari dalam VPC klaster.
6. (Opsional) Jika Anda telah mengaktifkan Akses publik, Anda dapat menentukan alamat mana dari internet yang dapat berkomunikasi ke titik akhir publik. Pilih Pengaturan lanjutan. Masukkan blok CIDR, seperti`203.0.113.5/32`. Blok tidak dapat mencakup reserved addresses. Anda dapat memasukkan blok tambahan dengan memilih Tambahkan sumber. Ada jumlah maksimum blok CIDR yang dapat Anda tentukan. Untuk informasi selengkapnya, lihat Amazon EKS service quotas. Jika Anda menetapkan tidak ada blok, maka titik akhir server API publik menerima permintaan dari semua (`0.0.0.0/0`) alamat IP. Jika Anda membatasi akses ke titik akhir publik Anda menggunakan blok CIDR, disarankan agar Anda juga mengaktifkan akses endpoint pribadi sehingga node dan Fargate Pods (jika Anda menggunakannya) dapat berkomunikasi dengan cluster. Tanpa mengaktifkan titik akhir privat, sumber CIDR titik akhir akses publik Anda harus menyertakan sumber keluar dari VPC Anda. Misalnya, jika Anda memiliki simpul di subnet privat yang berkomunikasi dengan

internet melalui NAT Gateway, Anda perlu menambahkan alamat IP keluar dari NAT Gateway sebagai bagian dari blok CIDR yang diizinkan di titik akhir publik Anda.

7. Pilih Perbarui untuk menyelesaikan.

AWS CLI

Untuk memodifikasi akses titik akhir server API klaster Anda menggunakan AWS CLI

Selesaikan langkah-langkah berikut menggunakan AWS CLI versi `1.27.160` atau yang lebih baru. Anda dapat memeriksa versi saat ini dengan `aws --version`. Untuk menginstal atau memutakhirkan AWS CLI, lihat Menginstal AWS CLI.

1. Perbarui akses titik akhir server API klaster Anda dengan perintah AWS CLI berikut. Gantikan nama klaster Anda dan nilai akses titik akhir yang diinginkan. Jika Anda mengatur `endpointPublicAccess=true`, maka Anda dapat (opsional) memasukkan satu blok CIDR, atau daftar blok CIDR yang dipisahkan koma untuk `publicAccessCidrs`. Blok tidak dapat mencakup alamat yang disimpan. Jika Anda menentukan blok CIDR, maka titik akhir server API publik hanya akan menerima permintaan dari blok yang terdaftar. Ada jumlah maksimum blok CIDR yang dapat Anda tentukan. Untuk informasi selengkapnya, lihat Amazon EKS service quotas. Jika Anda membatasi akses ke titik akhir publik Anda menggunakan blok CIDR, disarankan agar Anda juga mengaktifkan akses endpoint pribadi sehingga node dan Fargate Pods (jika Anda menggunakannya) dapat berkomunikasi dengan cluster. Tanpa mengaktifkan titik akhir privat, sumber CIDR titik akhir akses publik Anda harus menyertakan sumber keluar dari VPC Anda. Misalnya, jika Anda memiliki simpul di subnet privat yang berkomunikasi dengan internet melalui NAT Gateway, Anda perlu menambahkan alamat IP keluar dari NAT Gateway sebagai bagian dari blok CIDR yang diizinkan di titik akhir publik Anda. Jika Anda tidak menentukan blok CIDR, maka titik akhir server API publik menerima permintaan dari semua alamat IP (0.0.0.0/0).

> **Note**
>
> Perintah berikut memungkinkan akses privat dan akses publik dari satu alamat IP untuk titik akhir server API. Ganti *`203.0.113.5/32`* dengan satu blok CIDR, atau daftar blok CIDR yang dipisahkan koma yang ingin Anda batasi akses jaringan.

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

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

Contoh output adalah sebagai berikut.

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

2. Pantau status pembaruan akses titik akhir Anda dengan perintah berikut, menggunakan nama klaster dan ID pembaruan yang dikembalikan oleh perintah sebelumnya. Pembaruan Anda selesai saat status ditampilkan sebagai `Successful`.

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

Contoh output adalah sebagai berikut.

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

Mengakses server API privat saja

Jika Anda telah menonaktifkan akses publik untuk titik akhir server Kubernetes API klaster, Anda hanya dapat mengakses server API dari dalam VPC atau jaringan yang terhubung. Berikut adalah beberapa cara yang mungkin untuk mengakses endpoint server Kubernetes API:

Jaringan yang terhubung

Hubungkan jaringan Anda ke VPC dengan gateway AWS transit atau opsi konektivitas lainnya dan kemudian gunakan komputer di jaringan yang terhubung. Anda harus memastikan bahwa grup keamanan pesawat kendali Amazon EKS Anda berisi aturan yang mengizinkan lalu lintas masuk di port 443 dari jaringan terhubung milik Anda.

Tuan rumah benteng Amazon EC2

Anda dapat meluncurkan instans Amazon EC2 ke subnet publik di VPC klaster Anda dan kemudian masuk melalui SSH ke instance tersebut untuk menjalankan perintah. `kubectl` Untuk

informasi lebih lanjut, lihat host Linux benteng di AWS. Anda harus memastikan bahwa grup keamanan pesawat kendali Amazon EKS Anda berisi aturan yang mengizinkan lalu lintas masuk di port 443 dari host bastion milik Anda. Untuk informasi selengkapnya, lihat Persyaratan dan pertimbangan grup keamanan Amazon EKS.

Saat Anda mengonfigurasi `kubectl` untuk host bastion Anda, pastikan untuk menggunakan AWS kredenal yang sudah dipetakan ke konfigurasi RBAC cluster Anda, atau tambahkan prinsipal IAM yang akan digunakan bastion Anda ke konfigurasi RBAC sebelum Anda menghapus akses publik titik akhir. Lihat informasi yang lebih lengkap di the section called “Berikan akses ke API Kubernetes” dan Tidak sah atau akses ditolak (`kubectl`).

AWS Cloud9 IDE

AWS Cloud9 adalah lingkungan pengembangan terintegrasi berbasis cloud (IDE) yang memungkinkan Anda menulis, menjalankan, dan men-debug kode Anda hanya dengan browser. Anda dapat membuat AWS Cloud9 IDE di VPC klaster Anda dan menggunakan IDE untuk berkomunikasi dengan cluster Anda. Untuk informasi selengkapnya, lihat Membuat lingkungan di AWS Cloud9. Anda harus memastikan bahwa grup keamanan pesawat kendali Amazon EKS Anda berisi aturan yang mengizinkan lalu lintas masuk di port 443 dari grup keamanan IDE milik Anda. Untuk informasi selengkapnya, lihat Persyaratan dan pertimbangan grup keamanan Amazon EKS.

Saat Anda mengonfigurasi AWS Cloud9 IDE Anda, pastikan `kubectl` untuk menggunakan AWS kredensional yang sudah dipetakan ke konfigurasi RBAC klaster Anda, atau tambahkan prinsipal IAM yang akan digunakan IDE Anda ke konfigurasi RBAC sebelum Anda menghapus akses publik titik akhir. Lihat informasi yang lebih lengkap di Berikan akses ke Kubernetes API dan Tidak sah atau akses ditolak (`kubectl`).

Mengaktifkan enkripsi rahasia pada cluster yang ada

Jika Anda mengaktifkan enkripsi rahasia, Kubernetes rahasia dienkripsi menggunakan AWS KMS key yang Anda pilih. Kunci KMS harus memenuhi ketentuan berikut:

- Simetris
- Dapat mengenkripsi dan mendekripsi data
- Dibuat Wilayah AWS sama dengan cluster
- Jika kunci KMS dibuat di akun yang berbeda, kepala sekolah IAM harus memiliki akses ke kunci KMS.

Untuk informasi selengkapnya, lihat Mengizinkan prinsipal IAM di akun lain menggunakan kunci KMS di Panduan Pengembang. AWS Key Management Service

> **Warning**
>
> Anda tidak dapat menonaktifkan enkripsi rahasia setelah mengaktifkannya. Tindakan ini tidak dapat diubah.

eksctl

Anda dapat mengaktifkan enkripsi dengan dua cara:

- Tambahkan enkripsi ke klaster Anda dengan satu perintah.

  Untuk mengenkripsi ulang rahasia Anda secara otomatis, jalankan perintah berikut.

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

  Untuk memilih keluar dari mengenkripsi ulang rahasia Anda secara otomatis, jalankan perintah berikut.

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

- Tambahkan enkripsi ke cluster Anda dengan `kms-cluster.yaml` file.

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

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

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

Agar rahasia Anda dienkripsi ulang secara otomatis, jalankan perintah berikut.

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

Untuk memilih keluar dari enkripsi ulang rahasia Anda secara otomatis, jalankan perintah berikut.

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

AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih cluster yang ingin Anda tambahkan enkripsi KMS.
3. Pilih tab Ikhtisar (ini dipilih secara default).
4. Gulir ke bawah ke bagian enkripsi Rahasia dan pilih Aktifkan.
5. Pilih kunci dari daftar dropdown dan pilih tombol Enable. Jika tidak ada kunci yang tercantum, Anda harus membuatnya dulu. Untuk informasi selengkapnya, lihat Membuat kunci
6. Pilih tombol Konfirmasi untuk menggunakan tombol yang dipilih.

AWS CLI

1. Kaitkan konfigurasi enkripsi rahasia dengan cluster Anda menggunakan AWS CLI perintah berikut. Ganti *example values* dengan milik Anda sendiri.

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

Contoh output adalah sebagai berikut.

```
{
  "update": {
    "id": "3141b835-8103-423a-8e68-12c2521ffa4d",
    "status": "InProgress",
```

```
    "type": "AssociateEncryptionConfig",
    "params": [
      {
        "type": "EncryptionConfig",
        "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\"arn:aws:kms:region-code:account:key/key\"}}]"
      }
    ],
    "createdAt": 1613754188.734,
    "errors": []
  }
}
```

2. Anda dapat memantau status pembaruan enkripsi Anda dengan perintah berikut. Gunakan spesifik `cluster name` dan `update ID` yang dikembalikan pada output sebelumnya. Ketika `Successful` status ditampilkan, pembaruan selesai.

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

Contoh output adalah sebagai berikut.

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

3. Untuk memverifikasi bahwa enkripsi diaktifkan di klaster Anda, jalankan perintah `describe-cluster`. Responsnya berisi `EncryptionConfig` string.

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

Setelah Anda mengaktifkan enkripsi pada cluster Anda, Anda harus mengenkripsi semua rahasia yang ada dengan kunci baru:

> **Note**
>
> Jika Anda menggunakan`eksctl`, menjalankan perintah berikut hanya diperlukan jika Anda memilih untuk tidak mengenkripsi ulang rahasia Anda secara otomatis.

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

> **Warning**
>
> Jika Anda mengaktifkan enkripsi rahasia untuk klaster yang ada dan kunci KMS yang Anda gunakan pernah dihapus, maka tidak ada cara untuk memulihkan cluster. Jika Anda menghapus kunci KMS, Anda secara permanen menempatkan cluster dalam keadaan terdegradasi. Untuk informasi selengkapnya, lihat Menghapus kunci AWS KMS.

> **Note**
>
> Secara default, `create-key` perintah membuat kunci KMS enkripsi simetris dengan kebijakan kunci yang memberikan akses admin root akun pada AWS KMS tindakan dan sumber daya. Jika Anda ingin mengurangi izin, pastikan bahwa `kms:CreateGrant` tindakan `kms:DescribeKey` dan diizinkan pada kebijakan untuk prinsipal yang memanggil `create-cluster` API.
>
> Untuk cluster yang menggunakan Enkripsi Amplop KMS, `kms:CreateGrant` izin diperlukan. Kondisi `kms:GrantIsForAWSResource` ini tidak didukung untuk CreateCluster tindakan, dan tidak boleh digunakan dalam kebijakan KMS untuk mengontrol `kms:CreateGrant` izin bagi pengguna yang melakukan. CreateCluster

Mengaktifkan Windows dukungan untuk klaster Amazon EKS Anda

Sebelum menerapkan Windows node, perhatikan pertimbangan berikut.

Pertimbangan

- Anda dapat menggunakan jaringan host pada node Windows menggunakan `HostProcess` Pod. Untuk informasi selengkapnya, lihat Membuat Windows `HostProcess`Pod di Kubernetes dokumentasi.
- Cluster Amazon EKS harus berisi satu Linux atau lebih node Fargate untuk menjalankan Pods sistem inti yang hanya Linux berjalan, seperti. CoreDNS
- Log `kube-proxy` peristiwa `kubelet` dan diarahkan ke Log EKS Windows Peristiwa dan diatur ke batas 200 MB.
- Anda tidak dapat menggunakan Kelompok keamanan untuk Pods dengan Pods berjalan di Windows node.
- Anda tidak dapat menggunakan jaringan khusus dengan Windows node.
- Anda tidak dapat menggunakan `IPv6` dengan Windows node.
- Windowsnode mendukung satu elastic network interface per node. Secara default, jumlah Pods yang dapat Anda jalankan per Windows node sama dengan jumlah alamat IP yang tersedia per elastic network interface untuk jenis instance node, minus satu. Untuk informasi selengkapnya, lihat alamat IP per antarmuka jaringan per jenis instans di Panduan Pengguna Amazon EC2 untuk Instans Windows.
- Di kluster Amazon EKS, satu layanan dengan penyeimbang beban dapat mendukung hingga 1024 back-end. Pods Masing-masing Pod memiliki alamat IP uniknya sendiri. Batas sebelumnya 64 Pods tidak lagi terjadi, setelah pembaruan Windows Server dimulai dengan OS Build 17763.2746.
- Kontainer Windows tidak didukung untuk Amazon EKS Pods di Fargate.
- Anda tidak dapat mengambil log dari `vpc-resource-controller` Pod. Anda sebelumnya bisa ketika Anda menyebarkan pengontrol ke bidang data.
- Ada periode pendinginan sebelum `IPv4` alamat ditetapkan ke Pod baru. Hal ini mencegah lalu lintas mengalir ke Pod lama dengan `IPv4` alamat yang sama karena `kube-proxy` aturan basi.
- Sumber untuk pengontrol dikelola padaGitHub. Untuk berkontribusi, atau mengajukan masalah terhadap pengontrol, kunjungi proyek diGitHub.
- Saat menentukan ID AMI khusus untuk grup node Windows terkelola, tambahkan `eks:kube-proxy-windows` ke peta konfigurasi AWS IAM Authenticator Anda. Untuk informasi selengkapnya, lihat Batas dan ketentuan saat menentukan ID AMI.

Prasyarat

- Sebuah klaster yang sudah ada. Cluster harus menjalankan salah satu Kubernetes versi dan versi platform yang tercantum dalam tabel berikut. Versi apa pun Kubernetes dan platform yang lebih lambat dari yang terdaftar juga didukung. Jika versi klaster atau platform Anda lebih awal dari salah satu versi berikut, Anda perlu mengaktifkan Windows dukungan lama di bidang data klaster Anda. Setelah klaster Anda berada di salah satu versi berikut Kubernetes dan platform, atau yang lebih baru, Anda dapat menghapus Windows dukungan lama dan mengaktifkan Windows dukungan di bidang kontrol Anda.

Versi Kubernetes	Versi platform
1.29	eks.1
1.28	eks.1
1.27	eks.1
1.26	eks.1
1,25	eks.1
1.24	eks.2

- Cluster Anda harus memiliki setidaknya satu (kami sarankan setidaknya dua) Linux node atau Fargate Pod untuk dijalankan. CoreDNS Jika Anda mengaktifkan Windows dukungan lama, Anda harus menggunakan Linux node (Anda tidak dapat menggunakan Pod Fargate) untuk menjalankannya. CoreDNS
- Yang adaIAM role klaster Amazon EKS.

Mengaktifkan dukungan Windows

Jika klaster Anda tidak berada di, atau lebih baru, dari salah satu versi Kubernetes dan platform yang tercantum dalam Prasyarat, Anda harus mengaktifkan dukungan lama sebagai gantinya. Windows Untuk informasi selengkapnya, lihat Mengaktifkan dukungan lama Windows.

Jika Anda belum pernah mengaktifkan Windows dukungan di klaster Anda, lanjutkan ke langkah berikutnya.

Jika Anda mengaktifkan Windows dukungan pada klaster yang lebih awal dari versi Kubernetes atau platform yang tercantum dalam Prasyarat, maka Anda harus terlebih dahulu menghapus `vpc-resource-controller` dan `vpc-admission-webhook` dari bidang data Anda. Mereka sudah usang dan tidak lagi dibutuhkan.

Untuk mengaktifkan Windows dukungan untuk klaster Anda

1. Jika Anda tidak memiliki node Amazon Linux di cluster Anda dan menggunakan grup keamanan untukPods, lewati ke langkah berikutnya. Jika tidak, konfirmasikan bahwa kebijakan `AmazonEKSVPCResourceController` terkelola dilampirkan ke peran klaster Anda. Ganti *`eksClusterRole`* dengan nama peran cluster Anda.

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

   Contoh output adalah sebagai berikut.

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

   Jika kebijakan dilampirkan, seperti pada output sebelumnya, lewati langkah berikutnya.

2. Lampirkan kebijakan ResourceController terkelola AmazonekSvPC ke Anda. IAM role klaster Amazon EKS Ganti *`eksClusterRole`* dengan nama peran cluster Anda.

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

3. Buat file bernama *`vpc-resource-controller-configmap.yaml`* dengan isi berikut ini.

   ```
   apiVersion: v1
   ```

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

4. Terapkan `ConfigMap` ke cluster Anda.

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

5. Verifikasi bahwa Anda `aws-auth` `ConfigMap` berisi pemetaan untuk peran instance Windows node untuk menyertakan grup izin `eks:kube-proxy-windows` RBAC. Anda dapat memverifikasi dengan menjalankan perintah berikut.

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

Contoh output adalah sebagai berikut.

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

Anda harus melihat `eks:kube-proxy-windows` terdaftar di bawah grup. Jika grup tidak ditentukan, Anda perlu memperbarui `ConfigMap` atau membuatnya untuk menyertakan grup yang diperlukan. Untuk informasi lebih lanjut tentang `aws-auth``ConfigMap`, lihatTerapkan aws-authConfigMap ke cluster Anda.

Menghapus Windows dukungan lama dari pesawat data Anda

Jika Anda mengaktifkan Windows dukungan pada klaster yang lebih awal dari versi Kubernetes atau platform yang tercantum dalam Prasyarat, maka Anda harus terlebih dahulu menghapus `vpc-resource-controller` dan `vpc-admission-webhook` dari bidang data Anda. Mereka sudah usang dan tidak lagi diperlukan karena fungsionalitas yang mereka sediakan sekarang diaktifkan di bidang kontrol.

1. Copot pemasangan `vpc-resource-controller` dengan perintah berikut. Gunakan perintah ini terlepas dari alat mana Anda awalnya menginstalnya. Ganti *`region-code`* (hanya instance teks itu setelahnya`/manifests/`) dengan tempat Wilayah AWS cluster Anda berada.

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

2. Copot pemasangan `vpc-admission-webhook` menggunakan instruksi untuk alat yang Anda instal.

   eksctl

   Jalankan perintah berikut.

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

   kubectl on macOS or Windows

   Jalankan perintah berikut. Ganti *`region-code`* (hanya instance teks itu setelahnya`/manifests/`) dengan tempat Wilayah AWS cluster Anda berada.

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

3. Aktifkan Windows dukungan untuk cluster Anda di bidang kontrol.

Menonaktifkan dukungan Windows

Untuk menonaktifkan Windows dukungan pada klaster Anda

1. Jika klaster Anda berisi node Amazon Linux dan Anda menggunakan grup keamanan untuk Pods bersamanya, lewati langkah ini.

   Hapus kebijakan IAM `AmazonVPCResourceController` terkelola dari peran klaster Anda. Ganti *`eksClusterRole`* dengan nama peran klaster Anda dan *`111122223333`* dengan ID akun Anda.

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

2. Nonaktifkan Windows IPAM di. `amazon-vpc-cni` ConfigMap

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

Menerapkan Pod

Saat Anda menerapkan Pod ke klaster Anda, Anda perlu menentukan sistem operasi yang mereka gunakan jika Anda menjalankan campuran tipe node.

Untuk LinuxPods, gunakan teks pemilih simpul berikut dalam manifes Anda.

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

Untuk WindowsPods, gunakan teks pemilih simpul berikut dalam manifes Anda.

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

Anda dapat menerapkan aplikasi sampel untuk melihat pemilih node yang digunakan.

Mengaktifkan dukungan lama Windows

Jika klaster Anda berada di, atau lebih baru, dari salah satu versi Kubernetes dan platform yang tercantum dalam Prasyarat, maka sebaiknya Anda mengaktifkan Windows dukungan pada bidang kontrol Anda. Untuk informasi selengkapnya, lihat Mengaktifkan dukungan Windows.

Langkah-langkah berikut membantu Anda mengaktifkan Windows dukungan lama untuk bidang data kluster Amazon EKS jika versi klaster atau platform Anda lebih awal dari versi yang tercantum dalam Prasyarat. Setelah klaster dan versi platform Anda berada di, atau lebih lambat dari versi yang tercantum dalam Prasyarat, kami sarankan Anda menghapus Windows dukungan lama dan mengaktifkannya untuk bidang kontrol Anda.

Anda dapat menggunakan`eksctl`, Windows klien, atau macOS atau Linux klien untuk mengaktifkan Windows dukungan lama untuk klaster Anda.

eksctl

Untuk mengaktifkan Windows dukungan lama untuk klaster Anda dengan **`eksctl`**

Prasyarat

Prosedur ini membutuhkan `eksctl` versi `0.175.0` atau yang lebih baru. Anda dapat memeriksa versi Amazon Linux Anda dengan perintah berikut.

```
eksctl version
```

Untuk informasi selengkapnya tentang menginstal atau meningkatkan`eksctl`, lihat Instalasi di `eksctl` dokumentasi.

1. Aktifkan Windows dukungan untuk klaster Amazon EKS Anda dengan `eksctl` perintah berikut. Ganti *`my-cluster`* dengan nama klaster Anda. Perintah ini menerapkan pengontrol sumber daya VPC dan webhook pengontrol penerimaan VPC yang diperlukan pada kluster Amazon EKS untuk menjalankan beban kerja. Windows

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

   > **Important**
   >
   > Webhook pengendali izin masuk VPC ditandatangani dengan sertifikat yang berakhir satu tahun setelah tanggal penerbitan. Untuk menghindari waktu henti proses,

pastikan untuk memperpanjang sertifikat sebelum kedaluwarsa. Untuk informasi selengkapnya, lihat Memperbarui sertifikat webhook izin masuk VPC.

2. Setelah Anda mengaktifkan Windows dukungan, Anda dapat meluncurkan grup Windows node ke dalam cluster Anda. Untuk informasi selengkapnya, lihat Meluncurkan node yang dikelola sendiri Windows.

Windows

Untuk mengaktifkan Windows dukungan lama untuk klaster Anda dengan klien Windows

Pada langkah-langkah berikut, ganti *region-code* dengan tempat Wilayah AWS klaster Anda berada.

1. Deploy pengendali sumber daya VPC ke klaster Anda.

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

2. Deploy webhook pengendali izin masuk VPC ke klaster Anda.

   a. Unduh skrip dan file deployment yang diperlukan.

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

   b. Pasang OpenSSL dan jq.

   c. Siapkan dan deploy webhook izin masuk VPC.

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

> **Important**
>
> Webhook pengendali izin masuk VPC ditandatangani dengan sertifikat yang berakhir satu tahun setelah tanggal penerbitan. Untuk menghindari waktu henti proses, pastikan untuk memperpanjang sertifikat sebelum kedaluwarsa. Untuk informasi selengkapnya, lihat Memperbarui sertifikat webhook izin masuk VPC.

3. Tentukan apakah klaster Anda memiliki pengikatan peran klaster yang diperlukan.

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

Jika output yang mirip dengan output contoh berikut dikembalikan, maka klaster memiliki pengikatan peran yang diperlukan.

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

Jika output mencakup `Error from server (NotFound)`, maka klaster tidak memiliki pengikatan peran klaster yang diperlukan. Tambahkan pengikatan dengan membuat file yang diberi nama *`eks-kube-proxy-windows-crb.yaml`* dengan konten berikut.

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

Menerapkan konfigurasi pada klaster.

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

4. Setelah Anda mengaktifkan Windows dukungan, Anda dapat meluncurkan grup Windows node ke dalam cluster Anda. Untuk informasi selengkapnya, lihat Meluncurkan node yang dikelola sendiri Windows.

macOS and Linux

Untuk mengaktifkan Windows dukungan lama untuk klaster Anda dengan Linux macOS atau klien

Prosedur ini mensyaratkan bahwa `openssl` perpustakaan dan `jq` Prosesor JSON telah diinstal pada sistem klien Anda.

Pada langkah-langkah berikut, ganti *`region-code`* dengan tempat Wilayah AWS klaster Anda berada.

1. Men-deploy pengendali sumber daya VPC untuk klaster Anda.

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

2. Buat manifes webhook pengendali izin masuk VPC untuk klaster Anda.

   a. Unduh skrip penulisan dan file deployment yang diperlukan.

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

   b. Mengamenambahkan izin pada penulisan shell sehingga penulisan dapat dijalankan.

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

   c. Cipatakan rahasia untuk komunikasi komunitas pengguna yang aman.

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

   d. Verifikasi rahasia.

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

e. Mengonfigurasi webhook dan membuat file deployment.

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

3. Deploy webhook izin masuk VPC.

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

> **Important**
>
> Webhook pengendali izin masuk VPC ditandatangani dengan sertifikat yang berakhir satu tahun setelah tanggal penerbitan. Untuk menghindari waktu henti proses, pastikan untuk memperpanjang sertifikat sebelum kedaluwarsa. Untuk informasi selengkapnya, lihat Memperbarui sertifikat webhook izin masuk VPC.

4. Tentukan apakah klaster Anda memiliki pengikatan peran klaster yang diperlukan.

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

Jika output yang mirip dengan output contoh berikut dikembalikan, maka klaster memiliki pengikatan peran yang diperlukan.

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

Jika output mencakup `Error from server (NotFound)`, maka klaster tidak memiliki pengikatan peran klaster yang diperlukan. Tambahkan pengikatan dengan membuat file yang diberi nama *`eks-kube-proxy-windows-crb.yaml`* dengan konten berikut.

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: eks:kube-proxy-windows
  labels:
    k8s-app: kube-proxy
```

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

Menerapkan konfigurasi pada klaster.

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

5. Setelah Anda mengaktifkan Windows dukungan, Anda dapat meluncurkan grup Windows node ke dalam cluster Anda. Untuk informasi selengkapnya, lihat Meluncurkan node yang dikelola sendiri Windows.

Memperbarui sertifikat webhook izin masuk VPC

Sertifikat yang digunakan oleh webhook izin masuk VPC berakhir satu tahun setelah diterbitkan. Untuk menghindari waktu henti, penting bagi Anda untuk memperbarui sertifikat sebelum kedaluwarsa. Anda dapat memeriksa tanggal kedaluwarsa sertifikat Anda saat ini dengan perintah berikut.

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

Contoh output adalah sebagai berikut.

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

Anda dapat memperbarui sertifikat menggunakan `eksctl` atau komputer Windows Linux/macOS. Ikuti petunjuk untuk alat yang awalnya Anda gunakan untuk memasang webhook izin masuk VPC.

Misalnya, jika Anda awalnya memasang webhook izin masuk VPC menggunakan `eksctl`, maka Anda harus memperbarui sertifikat menggunakan petunjuk pada tab `eksctl`.

eksctl

1. Pasang ulang sertifikat. Ganti *`my-cluster`* dengan nama klaster Anda.

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

2. Verifikasi bahwa Anda penerima dari hasil berikut.

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

3. Mulai ulang webhook deployment.

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

4. Jika sertifikat yang Anda perpanjang telah kedaluwarsa, dan Anda Windows Pods terjebak dalam `Container creating` status, maka Anda harus menghapus dan menerapkannya kembali. Pods

Windows

1. Dapatkan penulisan untuk menghasilkan sertifikat baru.

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

2. Siapkan parameter untuk penulisan.

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

3. Mulai ulang webhook deployment.

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

4. Jika sertifikat yang Anda perpanjang telah kedaluwarsa, dan Anda Windows Pods terjebak dalam `Container creating` status, maka Anda harus menghapus dan menerapkannya kembali. Pods

Linux and macOS

Prasyarat

Anda harus memiliki OpenSSL `jq` dan diinstal pada komputer Anda.

1. Dapatkan penulisan untuk menghasilkan sertifikat baru.

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

2. Ubah izin.

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

3. Jalankan penulisan.

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

4. Mulai ulang webhook.

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

5. Jika sertifikat yang Anda perpanjang telah kedaluwarsa, dan Anda Windows Pods terjebak dalam `Container creating` status, maka Anda harus menghapus dan menerapkannya kembali. Pods

Mendukung Pod kepadatan yang lebih tinggi pada node Windows

Di Amazon EKS, masing-masing Pod dialokasikan `IPv4` alamat dari VPC Anda. Karena ini, jumlah Pods yang dapat Anda terapkan ke node dibatasi oleh alamat IP yang tersedia, bahkan jika ada sumber daya yang cukup untuk menjalankan lebih banyak Pods pada node. Karena hanya satu elastic network interface yang didukung oleh node Windows, secara default, jumlah maksimum alamat IP yang tersedia pada node Windows sama dengan:

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

Satu alamat IP digunakan sebagai alamat IP utama dari antarmuka jaringan, sehingga tidak dapat dialokasikanPods.

Anda dapat mengaktifkan Pod kepadatan yang lebih tinggi pada node Windows dengan mengaktifkan delegasi awalan IP. Fitur ini memungkinkan Anda untuk menetapkan `/28` `IPv4` awalan ke antarmuka jaringan utama, bukan menetapkan alamat sekunder. `IPv4` Menetapkan awalan IP meningkatkan `IPv4` alamat maksimum yang tersedia pada node menjadi:

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

Dengan jumlah alamat IP yang tersedia secara signifikan lebih besar ini, alamat IP yang tersedia seharusnya tidak membatasi kemampuan Anda untuk menskalakan jumlah Pods pada node Anda. Untuk informasi selengkapnya, lihat Tingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda.

Persyaratan klaster pribadi

Topik ini menjelaskan cara menerapkan kluster Amazon EKS yang digunakan di AWS Cloud, tetapi tidak memiliki akses internet keluar. Jika Anda memiliki kluster lokal AWS Outposts, lihatMeluncurkan node Amazon Linux yang dikelola sendiri di Outpost, alih-alih topik ini.

Jika Anda tidak familier dengan jaringan Amazon EKS, lihat De-mistifikasi jaringan klaster untuk simpul pekerja Amazon EKS. Jika klaster Anda tidak memiliki akses internet keluar, maka klaster harus memenuhi persyaratan berikut:

- Cluster Anda harus menarik gambar dari registri kontainer yang ada di VPC Anda. Anda dapat membuat Amazon Elastic Container Registry di VPC Anda dan menyalin gambar kontainer ke sana untuk diambil node Anda. Untuk informasi selengkapnya, lihat Salin gambar kontainer dari satu repositori ke repositori lain.
- Cluster Anda harus mengaktifkan akses pribadi endpoint. Hal ini diperlukan untuk node untuk mendaftar dengan endpoint cluster. titik akhir akses publik adalah opsional. Untuk informasi selengkapnya, lihat Kendali akses titik akhir klaster Amazon EKS.
- Dikelola sendiri Linux dan Windows node harus menyertakan argumen bootstrap berikut sebelum diluncurkan. Argumen ini melewati introspeksi Amazon EKS dan tidak memerlukan akses ke Amazon EKS API dari dalam VPC.

1. Tentukan nilai endpoint cluster Anda dengan perintah berikut. Ganti *my-cluster* dengan nama klaster Anda.

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

   Contoh output adalah sebagai berikut.

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

2. Tentukan nilai otoritas sertifikat klaster Anda dengan perintah berikut. Ganti *my-cluster* dengan nama klaster Anda.

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

   Output yang dikembalikan adalah string panjang.

3. Ganti *cluster-endpoint* dan *certificate-authority* dalam perintah berikut dengan nilai yang dikembalikan dalam output dari perintah sebelumnya. Untuk informasi selengkapnya tentang menentukan argumen bootstrap saat meluncurkan node yang dikelola sendiri, lihat Meluncurkan simpul Amazon Linux yang dikelola sendiri dan. Meluncurkan node yang dikelola sendiri Windows

- Untuk Linux node:

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

  Untuk argumen tambahan, lihat skrip bootstrap padaGitHub.

- Untuk Windows node:

> **Note**
>
> Jika Anda menggunakan layanan khusus CIDR, maka Anda perlu menentukannya menggunakan `-ServiceCIDR` parameter. Jika tidak, resolusi DNS untuk Pods di cluster akan gagal.

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

Untuk argumen tambahan, lihatParameter konfigurasi skrip bootstrap.

- Cluster Anda `aws-auth` `ConfigMap` harus dibuat dari dalam VPC Anda. Untuk informasi selengkapnya tentang membuat dan menambahkan entri ke `aws-auth``ConfigMap`, masukkan **`eksctl create iamidentitymapping --help`** di terminal Anda. Jika `ConfigMap` tidak ada di server Anda, `eksctl` akan membuatnya ketika Anda menggunakan perintah untuk menambahkan pemetaan identitas.
- Podsdikonfigurasi dengan peran IAM untuk akun layanan memperoleh kredensil dari panggilan API AWS Security Token Service (AWS STS). Jika tidak ada akses internet keluar, Anda harus membuat dan menggunakan titik akhir AWS STS VPC di VPC Anda. Sebagian besar AWS `v1` SDK menggunakan AWS STS titik akhir global secara default (`sts.amazonaws.com`), yang tidak menggunakan titik akhir VPC AWS STS . Untuk menggunakan titik akhir AWS STS VPC, Anda mungkin perlu mengonfigurasi SDK untuk menggunakan endpoint regional AWS STS (). `sts.`*`region-code`*`.amazonaws.com` Untuk informasi selengkapnya, lihat Konfigurasikan AWS Security Token Service titik akhir untuk akun layanan.

- Subnet VPC klaster Anda harus memiliki titik akhir antarmuka VPC untuk akses apa pun Layanan AWS yang Anda butuhkan. Pods Untuk informasi selengkapnya, silakan lihat Mengakses sebuah layanan AWS menggunakan sebuah titik akhir VPC antarmuka. Beberapa layanan dan titik akhir yang umum digunakan tercantum dalam tabel berikut. Untuk daftar lengkap titik akhir, lihat AWS layanan yang terintegrasi dengan AWS PrivateLink dalam AWS PrivateLink Panduan.

Layanan	Titik Akhir
Amazon EC2	com.amazonaws. *`kode wilayah.ec2`*
Amazon Elastic Container Registry (untuk menarik gambar kontainer)	com.amazonaws. *`kode wilayah .ecr.api`*, com.amazonaws. *`kode wilayah .ecr.dkr`*, dan com.amazonaws. *`kode wilayah .s3`*
Aplikasi Load Balancer dan Network Load Balancer	com.amazonaws. *`kode wilayah .elasticloadbalancing`*
AWS X-Ray	com.amazonaws. *`kode wilayah .xray`*
CloudWatch Log Amazon	com.amazonaws. *`kode wilayah.logs`*

Layanan	Titik Akhir
AWS Security Token Service (diperlukan saat menggunakan peran IAM untuk akun layanan)	com.amazonaws. *kode wilayah* .sts

Pertimbangan

- Setiap node yang dikelola sendiri harus disebarkan ke subnet yang memiliki titik akhir antarmuka VPC yang Anda butuhkan. Jika Anda membuat grup node terkelola, grup keamanan titik akhir antarmuka VPC harus mengizinkan CIDR untuk subnet, atau Anda harus menambahkan grup keamanan node yang dibuat ke grup keamanan titik akhir antarmuka VPC.
- Jika Anda Pods menggunakan volume Amazon EFS, maka sebelum menerapkan file kustomization.yaml driver harus diubah untuk menyetel image container agar digunakan sama dengan cluster Amazon EKS. Driver Amazon EFS CSI Wilayah AWS
- Anda dapat menggunakan AWS Load Balancer Controlleruntuk menyebarkan AWS Application Load Balancers (ALB) dan Network Load Balancers ke cluster pribadi Anda. Saat menerapkannya, Anda harus menggunakan flag baris perintah untuk mengaturenable-shield,enable-waf, dan enable-wafv2 ke false. Penemuan sertifikat dengan nama host dari objek Ingress tidak didukung. Ini karena pengontrol perlu menjangkau AWS Certificate Manager, yang tidak memiliki titik akhir antarmuka VPC.

  Pengendali yang didukung pada penyeimbang beban jaringan dengan target IP, yang diperlukan untuk penggunaan pada Fargate. Untuk informasi selengkapnya, lihat Penyeimbangan beban aplikasi pada Amazon EKS dan Buat penyeimbang beban jaringan.
- Cluster Autoscaler didukung. Saat menerapkan Cluster AutoscalerPods, pastikan bahwa baris perintah termasuk. --aws-use-static-instance-list=true Untuk informasi selengkapnya, lihat Menggunakan Daftar Instans Statis padaGitHub. VPC node pekerja juga harus menyertakan titik akhir VPC dan titik akhir AWS STS VPC penskalaan otomatis.
- Beberapa produk perangkat lunak kontainer menggunakan panggilan API yang mengakses AWS Marketplace Metering Service untuk memantau penggunaan. Cluster pribadi tidak mengizinkan panggilan ini, jadi Anda tidak dapat menggunakan jenis penampung ini di kluster pribadi.

KubernetesVersi Amazon EKS

Kubernetesberkembang pesat dengan fitur baru, pembaruan desain, dan perbaikan bug. Komunitas merilis versi Kubernetes minor baru (seperti`1.29`) rata-rata setiap empat bulan sekali. Amazon EKS mengikuti siklus rilis dan penghentian upstream untuk versi minor. Saat Kubernetes versi baru tersedia di Amazon EKS, kami menyarankan Anda memperbarui cluster Anda secara proaktif untuk menggunakan versi terbaru yang tersedia.

Versi minor berada di bawah dukungan standar di Amazon EKS selama 14 bulan pertama setelah dirilis. Setelah versi melewati akhir tanggal dukungan standar, secara otomatis memasuki dukungan diperpanjang untuk 12 bulan ke depan. Dukungan yang diperluas memungkinkan Anda untuk tinggal di Kubernetes versi tertentu lebih lama dengan biaya tambahan per jam cluster. Jika Anda belum memperbarui klaster sebelum periode dukungan yang diperpanjang berakhir, klaster Anda akan ditingkatkan secara otomatis ke versi perpanjangan tertua yang saat ini didukung.

Kami menyarankan Anda membuat cluster Anda dengan Kubernetes versi terbaru yang tersedia yang didukung oleh Amazon EKS. Jika aplikasi Anda memerlukan versi tertentuKubernetes, Anda dapat memilih versi yang lebih lama. Anda dapat membuat kluster Amazon EKS baru pada versi apa pun yang ditawarkan dalam dukungan standar atau diperpanjang.

Versi yang tersedia pada dukungan standar

KubernetesVersi berikut saat ini tersedia dalam dukungan standar Amazon EKS:

- `1.29`
- `1.28`
- `1.27`
- `1.26`
- `1.25`

Untuk perubahan penting yang harus diperhatikan untuk setiap versi dalam dukungan standar, lihatCatatan rilis untuk versi dukungan standar.

Versi yang tersedia pada dukungan diperpanjang

KubernetesVersi berikut saat ini tersedia di Amazon EKS dukungan diperpanjang:

- `1.24`

- 1.23

Untuk perubahan penting yang harus diperhatikan untuk setiap versi dalam dukungan yang diperluas, lihatCatatan rilis untuk versi dukungan yang diperluas.

KubernetesVersi berikut saat ini tersedia di Amazon EKS dukungan diperpanjang, dengan persyaratan tambahan bahwa Anda tidak dapat membuat cluster baru dengan versi ini:

- 1.22
- 1.21

  Untuk informasi tentang versi ini, lihat Catatan rilis untuk versi 1.21 dan 1.22

Kalender Kubernetes rilis Amazon EKS

Tabel berikut menunjukkan tanggal rilis dan dukungan penting untuk dipertimbangkan untuk setiap Kubernetes versi.

> **Note**
>
> Tanggal yang hanya berisi bulan dan tahun merupakan perkiraan, dan akan diperbarui dengan tanggal yang tepat saat diketahui.

Versi Kubernetes	Rilis hulu	Rilis Amazon EKS	Akhir dari dukungan standar	Akhir dari dukungan yang diperpanjang
1.29	13 Desember 2023	23 Januari 2024	Maret 23, 2025	Maret 23, 2026
1.28	15 Agustus 2023	26 September 2023	November 26, 2024	November 26, 2025
1.27	11 April 2023	24 Mei 2023	Juli 24, 2024	Juli 24, 2025
1.26	Desember 9, 2022	11 April 2023	Juni 11, 2024	Juni 11, 2025

Versi Kubernetes	Rilis hulu	Rilis Amazon EKS	Akhir dari dukungan standar	Akhir dari dukungan yang diperpanjang
1.25	23 Agustus 2022	22 Februari 2023	1 Mei 2024	1 Mei 2025
1.24	Mei 3, 2022	15 November 2022	Januari 31, 2024	Januari 31, 2025
1.23	Desember 7, 2021	Agustus 11, 2022	11 Oktober 2023	Oktober 11, 2024
1.22	4 Agustus 2021	4 April 2022	Juni 4, 2023	September 1, 2024
1.21	8 April 2021	19 Juli 2021	16 Februari 2023	Juli 15, 2024

FAQ versi Amazon EKS

Berapa banyak Kubernetes versi yang tersedia dalam dukungan standar?

> Sejalan dengan dukungan Kubernetes komunitas untuk Kubernetes versi, Amazon EKS berkomitmen untuk menawarkan dukungan standar untuk setidaknya empat versi siap produksi Kubernetes pada waktu tertentu. Kami akan mengumumkan akhir tanggal dukungan standar dari versi Kubernetes minor yang diberikan setidaknya 60 hari sebelumnya. Karena kualifikasi Amazon EKS dan proses rilis untuk Kubernetes versi baru, akhir tanggal dukungan standar Kubernetes versi di Amazon EKS akan aktif atau setelah tanggal Kubernetes proyek berhenti mendukung versi upstream.

Berapa lama Kubernetes menerima dukungan standar oleh Amazon EKS?

> Sebuah Kubernetes versi menerima dukungan standar selama 14 bulan setelah pertama kali tersedia di Amazon EKS. Ini benar bahkan jika upstream Kubernetes tidak lagi mendukung versi yang tersedia di Amazon EKS. Kami mendukung patch keamanan yang berlaku untuk Kubernetes versi yang didukung di Amazon EKS.

Apakah saya diberi tahu ketika dukungan standar berakhir untuk Kubernetes versi di Amazon EKS?

Ya. Jika ada cluster di akun Anda yang menjalankan versi mendekati akhir dukungan, Amazon EKS mengirimkan pemberitahuan AWS Health Dashboard sekitar 12 bulan setelah Kubernetes versi dirilis di Amazon EKS. Pemberitahuan tersebut mencakup akhir tanggal dukungan. Ini setidaknya 60 hari sejak tanggal pemberitahuan.

KubernetesFitur apa saja yang didukung oleh Amazon EKS?

Amazon EKS mendukung semua fitur Kubernetes API (GA) yang tersedia secara umum. Dimulai dengan Kubernetes versi`1.24`, API beta baru tidak diaktifkan di cluster secara default. Namun, API beta yang sudah ada sebelumnya dan versi baru dari API beta yang ada terus diaktifkan secara default. Fitur Alpha tidak didukung.

Apakah grup node terkelola Amazon EKS secara otomatis diperbarui bersama dengan versi bidang kontrol cluster?

Tidak. Grup node terkelola membuat instans Amazon EC2 di akun Anda. Instans-instans ini tidak secara otomatis dimutakhirkan saat Anda atau Amazon EKS memperbarui bidang kendali. Untuk informasi selengkapnya, lihat Memperbarui grup simpul terkelola. Sebaiknya pertahankan Kubernetes versi yang sama di bidang kontrol dan node Anda.

Apakah grup node yang dikelola sendiri secara otomatis diperbarui bersama dengan versi bidang kontrol cluster?

Tidak. Grup node yang dikelola sendiri menyertakan instans Amazon EC2 di akun Anda. Instans ini tidak ditingkatkan secara otomatis saat Anda atau Amazon EKS memperbarui versi pesawat kontrol atas nama Anda. Grup simpul swakelola sama sekali tidak mengindikasikan bahwa ia perlu diperbarui di dalam konsol. Anda dapat melihat versi `kubelet` yang diinstal pada sebuah simpul dengan memilih simpul di dalam daftar Simpul pada tab Gambaran Umum klaster Anda untuk menentukan simpul-simpul yang perlu diperbarui. Anda harus memperbarui simpul secara manual. Untuk informasi selengkapnya, lihat Pembaruan simpul yang dikelola sendiri.

KubernetesProyek ini menguji kompatibilitas antara bidang kontrol dan node hingga tiga versi minor. Misalnya, `1.26` node terus beroperasi ketika diatur oleh bidang kontrol. `1.29` Namun, menjalankan cluster dengan node yang terus-menerus tiga versi minor di belakang bidang kontrol tidak disarankan. Untuk informasi selengkapnya, lihat kebijakan dukungan Kubernetes versi dan versi miring dalam dokumentasi. Kubernetes Sebaiknya pertahankan Kubernetes versi yang sama di bidang kontrol dan node Anda.

Apakah Pods berjalan di Fargate secara otomatis ditingkatkan dengan peningkatan versi pesawat kontrol cluster otomatis?

Tidak. Kami sangat menyarankan menjalankan Fargate Pods sebagai bagian dari pengontrol replikasi, seperti penerapan. Kubernetes Kemudian lakukan restart bergulir dari semua FargatePods. Versi baru Fargate digunakan dengan versi Pod yang `kubelet` versi yang sama dengan versi bidang kontrol cluster Anda yang diperbarui. Untuk informasi selengkapnya, lihat Penerapan dalam dokumentasi. Kubernetes

> **Important**
>
> Jika Anda memperbarui bidang kontrol, Anda masih harus memperbarui node Fargate sendiri. Untuk memperbarui node Fargate, hapus Fargate yang Pod diwakili oleh node dan gunakan ulang. Pod Yang baru Pod diterapkan dengan `kubelet` versi yang versi yang sama dengan cluster Anda.

FAQ dukungan Amazon diperpanjang

Dukungan standar dan terminologi dukungan yang diperluas adalah hal baru bagi saya. Apa arti istilah-istilah itu?

Dukungan standar untuk Kubernetes versi di Amazon EKS dimulai ketika Kubernetes versi dirilis di Amazon EKS, dan akan berakhir 14 bulan setelah tanggal rilis. Dukungan diperpanjang untuk Kubernetes versi akan dimulai segera setelah akhir dukungan standar, dan akan berakhir setelah 12 bulan ke depan. Misalnya, dukungan standar untuk versi `1.23` di Amazon EKS berakhir pada 11 Oktober 2023. Dukungan diperpanjang untuk versi `1.23` dimulai pada 12 Oktober 2023 dan akan berakhir pada 11 Oktober 2024.

Apa yang harus saya lakukan untuk mendapatkan dukungan tambahan untuk cluster Amazon EKS?

Anda tidak perlu mengambil tindakan apa pun untuk mendapatkan dukungan tambahan untuk kluster Amazon EKS Anda. Dukungan standar akan dimulai ketika Kubernetes versi dirilis di Amazon EKS, dan akan berakhir 14 bulan setelah tanggal rilis. Dukungan diperpanjang untuk Kubernetes versi akan dimulai segera setelah akhir dukungan standar, dan akan berakhir setelah 12 bulan ke depan. Cluster yang berjalan pada Kubernetes versi yang melewati akhir dukungan standar akan secara otomatis dionboard ke dukungan yang diperluas.

Untuk Kubernetes versi mana saya bisa mendapatkan dukungan tambahan?

Dukungan yang diperluas tersedia untuk Kubernetes versi `1.23` dan lebih tinggi. Anda dapat menjalankan cluster pada versi apa pun hingga 12 bulan setelah akhir dukungan standar untuk versi itu. Ini berarti bahwa setiap versi akan didukung selama 26 bulan di Amazon EKS (14 bulan dukungan standar ditambah 12 bulan dukungan diperpanjang).

Bagaimana jika saya tidak ingin menggunakan dukungan yang diperluas?

Jika Anda tidak ingin terdaftar secara otomatis dalam dukungan yang diperluas, Anda dapat meningkatkan klaster Anda ke Kubernetes versi yang ada dalam dukungan Amazon EKS standar. Cluster yang tidak ditingkatkan ke Kubernetes versi dalam dukungan standar akan secara otomatis memasukkan dukungan yang diperluas.

Apa yang akan terjadi pada akhir 12 bulan dukungan diperpanjang?

Cluster yang berjalan pada Kubernetes versi yang telah menyelesaikan siklus hidup 26 bulan (dukungan standar 14 bulan ditambah dukungan diperpanjang 12 bulan) akan ditingkatkan secara otomatis ke versi berikutnya.

Di akhir tanggal dukungan yang diperpanjang, Anda tidak dapat lagi membuat kluster Amazon EKS baru dengan versi yang tidak didukung. Pesawat kontrol yang ada diperbarui secara otomatis oleh Amazon EKS ke versi paling awal yang didukung melalui proses penerapan bertahap setelah tanggal dukungan berakhir. Setelah pembaruan bidang kontrol otomatis, pastikan untuk memperbarui add-on cluster dan node Amazon EC2 secara manual. Untuk informasi selengkapnya, lihat Perbarui Kubernetes versi untuk kluster Amazon EKS Anda.

Kapan tepatnya pesawat kontrol saya diperbarui secara otomatis setelah akhir tanggal dukungan yang diperpanjang?

Amazon EKS tidak dapat memberikan kerangka waktu tertentu. Pembaruan otomatis dapat terjadi kapan saja setelah akhir tanggal dukungan yang diperpanjang. Anda tidak akan menerima pemberitahuan apa pun sebelum pembaruan. Kami menyarankan Anda secara proaktif memperbarui pesawat kontrol Anda tanpa bergantung pada proses pembaruan otomatis Amazon EKS. Untuk informasi selengkapnya, lihat Memperbarui Kubernetes versi cluster Amazon EKS.

Bisakah saya meninggalkan pesawat kontrol saya pada Kubernetes versi tanpa batas waktu?

Tidak. Keamanan cloud di AWS adalah prioritas tertinggi. Melewati titik tertentu (biasanya satu tahun), Kubernetes komunitas berhenti merilis kerentanan umum dan eksposur (CVE) patch dan mencegah pengiriman CVE untuk versi yang tidak didukung. Ini berarti bahwa kerentanan khusus untuk versi yang lebih lama bahkan Kubernetes mungkin tidak dilaporkan. Ini membuat cluster

terbuka tanpa pemberitahuan dan tidak ada opsi perbaikan jika terjadi kerentanan. Mengingat hal ini, Amazon EKS tidak mengizinkan pesawat kontrol untuk tetap menggunakan versi yang mencapai akhir dukungan yang diperpanjang.

Apakah ada biaya tambahan untuk mendapatkan dukungan tambahan?

Ya, ada biaya tambahan untuk kluster Amazon EKS yang berjalan dalam dukungan yang diperpanjang. Untuk detail harga, lihat dukungan tambahan Amazon EKS untuk harga Kubernetes versi di AWS blog.

Apa yang termasuk dalam dukungan diperpanjang?

Cluster Amazon EKS di Extended Support menerima patch keamanan yang sedang berlangsung untuk pesawat Kubernetes kontrol. Selain itu, Amazon EKS akan merilis patch untuk Amazon VPC CNI`kube-proxy`,, CoreDNS dan add-on untuk versi Extended Support. Amazon EKS juga akan merilis tambalan untuk AWS AMI Amazon EKS yang dioptimalkan untuk Amazon Linux,Bottlerocket, dan Windows, serta node Amazon EKS Fargate untuk versi tersebut. Semua cluster di Extended Support akan terus mendapatkan akses ke dukungan teknis dari AWS.

> **Note**
>
> Extended Support untuk Amazon EKS Windows AMI yang dioptimalkan yang diterbitkan oleh AWS tidak tersedia untuk Kubernetes versi `1.23` tetapi tersedia untuk Kubernetes versi `1.24` dan lebih tinggi.

Apakah ada batasan tambalan untuk Kubernetes non-komponen dalam dukungan yang diperluas?

Sementara Extended Support mencakup semua komponen Kubernetes spesifik dari AWS, itu hanya akan memberikan dukungan untuk AMI AWS yang dioptimalkan Amazon EKS yang dipublikasikan untuk Amazon LinuxBottlerocket,, dan Windows setiap saat. Ini berarti, Anda berpotensi memiliki komponen yang lebih baru (seperti OS atau kernel) di AMI yang dioptimalkan Amazon EKS saat menggunakan Extended Support. Misalnya, setelah Amazon Linux 2 mencapai akhir siklus hidupnya pada tahun 2025, Amazon EKS yang dioptimalkan Amazon Linux AMI akan dibangun menggunakan OS Amazon Linux yang lebih baru. Amazon EKS akan mengumumkan dan mendokumentasikan perbedaan siklus hidup dukungan penting seperti ini untuk setiap versi. Kubernetes

Dapatkah saya membuat cluster baru menggunakan versi pada dukungan yang diperluas?

Ya, dengan pengecualian `1.22` dan`1.21`. Misalnya, Anda dapat membuat `1.23` cluster, tetapi bukan `1.22` cluster.

Catatan rilis untuk versi dukungan standar

Topik ini memberikan perubahan penting yang harus diperhatikan untuk setiap Kubernetes versi dalam dukungan standar. Saat memutakhirkan, tinjau dengan cermat perubahan yang terjadi antara versi lama dan baru untuk klaster Anda.

> **Note**
>
> Untuk `1.24` dan kluster yang lebih baru, Amazon EKS AMI yang diterbitkan secara resmi termasuk `containerd` sebagai satu-satunya runtime. Kubernetesversi lebih awal dari `1.24` digunakan Docker sebagai runtime default. Versi ini memiliki opsi flag bootstrap yang dapat Anda gunakan untuk menguji beban kerja Anda pada klaster yang `containerd` didukung. Untuk informasi selengkapnya, lihat Amazon EKS mengakhiri dukungan untuk`Dockershim`.

Kubernetes1.29

Kubernetes`1.29`sekarang tersedia di Amazon EKS. Untuk informasi selengkapnya Kubernetes`1.29`, lihat pengumuman rilis resmi.

> **Important**
>
> - Versi `flowcontrol.apiserver.k8s.io/v1beta2` API yang tidak digunakan lagi dari `FlowSchema` dan tidak lagi `PriorityLevelConfiguration` disajikan. Kubernetes `v1.29` Jika Anda memiliki manifes atau perangkat lunak klien yang menggunakan grup API beta yang tidak digunakan lagi, Anda harus mengubahnya sebelum memutakhirkan ke. `v1.29`

- `.status.kubeProxyVersion`Bidang untuk objek Node sekarang tidak digunakan lagi, dan Kubernetes proyek mengusulkan untuk menghapus bidang itu dalam rilis future. Bidang usang tidak akurat dan secara historis telah dikelola oleh `kubelet` - yang sebenarnya tidak mengetahui `kube-proxy` versinya, atau bahkan apakah sedang berjalan. `kube-proxy` Jika Anda telah

menggunakan bidang ini dalam perangkat lunak klien, hentikan - informasinya tidak dapat diandalkan dan bidang tersebut sekarang tidak digunakan lagi.

- Kubernetes`1.29`Untuk mengurangi potensi permukaan serangan, `LegacyServiceAccountTokenCleanUp` fitur tersebut memberi label token berbasis rahasia yang dihasilkan secara otomatis sebagai tidak valid jika tidak digunakan untuk waktu yang lama (1 tahun secara default), dan secara otomatis menghapusnya jika penggunaan tidak dicoba untuk waktu yang lama setelah ditandai sebagai tidak valid (1 tahun tambahan secara default). Untuk mengidentifikasi token tersebut, Anda dapat menjalankan:

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

Untuk Kubernetes `1.29` changelog lengkap, lihat. https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.29.md#changelog-since-v1280

Kubernetes1.28

Kubernetes`1.28`sekarang tersedia di Amazon EKS. Untuk informasi selengkapnya Kubernetes`1.28`, lihat pengumuman rilis resmi.

- Kubernetes`v1.28`memperluas kemiringan yang didukung antara node inti dan komponen bidang kontrol dengan satu versi minor, dari `n-2` `ken-3`, sehingga komponen node (`kubelet`dan`kube-proxy`) untuk versi minor tertua yang didukung dapat bekerja dengan komponen bidang kontrol (`kube-apiserver`,, `kube-scheduler`kube-controller-manager,`cloud-controller-manager`) untuk versi minor terbaru yang didukung.
- Metrik `force_delete_pods_total` dan `force_delete_pod_errors_total` di `Pod GC Controller` dalamnya ditingkatkan untuk memperhitungkan semua penghapusan Pod yang kuat. Alasan ditambahkan ke metrik untuk menunjukkan apakah pod dihapus secara paksa karena dihentikan, yatim piatu, diakhiri dengan out-of-service taint, atau dihentikan dan tidak terjadwal.
- `PersistentVolume (PV)`Pengontrol telah dimodifikasi untuk secara otomatis menetapkan default `StorageClass` ke unbound apa pun `PersistentVolumeClaim` dengan `storageClassName` tidak disetel. Selain itu, mekanisme validasi `PersistentVolumeClaim` penerimaan dalam server API telah disesuaikan untuk memungkinkan perubahan nilai dari status yang tidak disetel ke nama sebenarnya`StorageClass` .

Untuk Kubernetes `1.28` changelog lengkap, lihat. https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.28.md#changelog-since-v1270

Kubernetes1.27

Kubernetes`1.27`sekarang tersedia di Amazon EKS. Untuk informasi selengkapnya Kubernetes`1.27`, lihat pengumuman rilis resmi.

Important

- Dukungan untuk anotasi `seccomp.security.alpha.kubernetes.io/pod` dan `seccomp` anotasi alfa telah `container.seccomp.security.alpha.kubernetes.io` dihapus. `seccomp`Anotasi alfa tidak digunakan lagi`1.19`, dan dengan penghapusannya`1.27`, `seccomp` bidang tidak akan lagi terisi otomatis dengan anotasi. `Pods` `seccomp` Sebagai gantinya, gunakan `securityContext.seccompProfile` bidang untuk `Pods` atau wadah untuk mengonfigurasi `seccomp` profil. Untuk memeriksa apakah Anda menggunakan `seccomp` anotasi alfa usang di klaster Anda, jalankan perintah berikut:

  ```
  kubectl get pods --all-namespaces -o json | grep
   -E 'seccomp.security.alpha.kubernetes.io/pod|
  container.seccomp.security.alpha.kubernetes.io'
  ```

- Argumen baris `--container-runtime` perintah untuk `kubelet` telah dihapus. Runtime container default untuk Amazon EKS telah ada `containerd` sejak saat itu`1.24`, yang menghilangkan kebutuhan untuk menentukan runtime container. Dari `1.27` dan seterusnya, Amazon EKS akan mengabaikan `--container-runtime` argumen yang diteruskan ke skrip bootstrap apa pun. Penting agar Anda tidak meneruskan argumen ini untuk `--kubelet-extra-args` mencegah kesalahan selama proses bootstrap node. Anda harus menghapus `--container-runtime` argumen dari semua alur kerja pembuatan node dan membangun skrip.

- `kubelet`Dalam Kubernetes `1.27` meningkatkan default `kubeAPIQPS` ke `50` dan `kubeAPIBurst` ke`100`. Penyempurnaan ini memungkinkan `kubelet` untuk menangani volume kueri API yang lebih tinggi, meningkatkan waktu respons dan kinerja. Ketika tuntutan untuk `Pods` meningkat, karena persyaratan penskalaan, default yang direvisi memastikan bahwa secara efisien `kubelet` dapat mengelola beban kerja yang meningkat. Akibatnya, `Pod` peluncuran lebih cepat dan operasi cluster lebih efektif.

- Anda dapat menggunakan Pod topologi berbutir lebih halus untuk menyebarkan kebijakan seperti. `minDomain` Parameter ini memberi Anda kemampuan untuk menentukan jumlah minimum domain yang `Pods` harus Anda sebarkan. `nodeAffinityPolicy`dan `nodeTaintPolicy` memungkinkan tingkat granularitas ekstra dalam mengatur Pod distribusi. Ini sesuai dengan afinitas simpul, noda, dan `matchLabelKeys` bidang dalam spesifikasi Anda`topologySpreadConstraints`. `Pod's` Hal ini memungkinkan pemilihan `Pods` untuk menyebarkan perhitungan setelah upgrade bergulir.
- Kubernetes`1.27`dipromosikan ke beta mekanisme kebijakan baru untuk `StatefulSets` yang mengontrol masa pakai `PersistentVolumeClaims` (PVCs) mereka. Kebijakan PVC retensi baru memungkinkan Anda menentukan apakah `PVCs` yang dihasilkan dari templat `StatefulSet` spesifikasi akan dihapus atau dipertahankan secara otomatis saat dihapus atau replika di dalamnya `StatefulSet` diperkecil. `StatefulSet`
- `goaway-chance`Opsi di server Kubernetes API membantu mencegah koneksi `HTTP/2` klien macet pada satu instance server API, dengan menutup koneksi secara acak. Ketika koneksi ditutup, klien akan mencoba untuk menyambung kembali, dan kemungkinan akan mendarat di server API yang berbeda sebagai akibat dari load balancing. Versi Amazon EKS `1.27` telah mengaktifkan `goaway-chance` bendera. Jika beban kerja Anda yang berjalan di klaster Amazon EKS menggunakan klien yang tidak kompatibel dengannya `HTTP GOAWAY`, sebaiknya Anda memperbarui klien agar ditangani `GOAWAY` dengan menyambungkan kembali saat penghentian koneksi.

Untuk Kubernetes `1.27` changelog lengkap, lihat. https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.27.md#changelog-since-v1260

Kubernetes1.26

Kubernetes`1.26`sekarang tersedia di Amazon EKS. Untuk informasi selengkapnya Kubernetes`1.26`, lihat pengumuman rilis resmi.

> **Important**
>
> Kubernetes`1.26`tidak lagi mendukung CRI`v1alpha2`. Hal ini mengakibatkan node `kubelet` tidak lagi mendaftarkan jika runtime container tidak mendukung CRI`v1`. Ini juga berarti bahwa Kubernetes `1.26` tidak mendukung versi minor `1.5` containerd dan versi sebelumnya. Jika Anda menggunakan containerd, Anda perlu memutakhirkan ke `1.6.0` versi containerd atau yang lebih baru sebelum memutakhirkan node apa pun ke. Kubernetes `1.26` Anda juga perlu memutakhirkan runtime kontainer lain yang hanya mendukung. `v1alpha2` Untuk

informasi lebih lanjut, tunda ke vendor runtime kontainer. Secara default, Amazon Linux dan Bottlerocket AMI menyertakan versi containerd. `1.6.6`

- Sebelum Anda meningkatkan ke Kubernetes`1.26`, tingkatkan Amazon VPC CNI plugin for Kubernetes ke versi Anda `1.12` atau yang lebih baru. Jika Anda tidak memutakhirkan ke Amazon VPC CNI plugin for Kubernetes versi `1.12` atau yang lebih baru, Amazon VPC CNI plugin for Kubernetes akan macet. Untuk informasi selengkapnya, lihat Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS.
- `goaway-chance`Opsi di server Kubernetes API membantu mencegah koneksi `HTTP/2` klien macet pada satu instance server API, dengan menutup koneksi secara acak. Ketika koneksi ditutup, klien akan mencoba untuk menyambung kembali, dan kemungkinan akan mendarat di server API yang berbeda sebagai akibat dari load balancing. Versi Amazon EKS `1.26` telah mengaktifkan `goaway-chance` bendera. Jika beban kerja Anda yang berjalan di klaster Amazon EKS menggunakan klien yang tidak kompatibel dengannya `HTTP GOAWAY`, sebaiknya Anda memperbarui klien agar ditangani `GOAWAY` dengan menyambungkan kembali saat penghentian koneksi.

Untuk Kubernetes `1.26` changelog lengkap, lihat. https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.26.md#changelog-since-v1250

Kubernetes1,25

Kubernetes`1.25`sekarang tersedia di Amazon EKS. Untuk informasi selengkapnya Kubernetes`1.25`, lihat pengumuman rilis resmi.

Important

- Dimulai dengan Kubernetes versi`1.25`, Anda tidak akan lagi dapat menggunakan `P2` instans Amazon EC2 dengan Amazon EKS yang dioptimalkan Amazon Linux AMI yang dipercepat di luar kotak. AMI ini untuk Kubernetes versi `1.25` atau yang lebih baru akan mendukung driver `NVIDIA 525` seri atau yang lebih baru, yang tidak kompatibel dengan `P2` instance. Namun, driver `NVIDIA 525` seri atau yang lebih baru kompatibel dengan`P3`,`P4`, dan `P5` instance, sehingga Anda dapat menggunakan instance tersebut dengan AMI untuk Kubernetes versi `1.25` atau yang lebih baru. Sebelum kluster Amazon EKS Anda ditingkatkan ke versi`1.25`, migrasikan `P2` instans apa pun ke,`P3`, `P4` dan instans. `P5` Anda juga harus secara proaktif meningkatkan aplikasi Anda untuk bekerja

dengan `NVIDIA 525` seri atau yang lebih baru. Kami berencana untuk mendukung port `NVIDIA 525` seri yang lebih baru atau driver yang lebih baru ke Kubernetes versi `1.23` dan `1.24` pada akhir Januari 2024.

- `PodSecurityPolicy`(PSP) dihapus di Kubernetes`1.25`. PSPsdiganti dengan Pod Security Admission (PSA) dan Pod Security Standards(PSS). PSAadalah pengontrol masuk bawaan yang mengimplementasikan kontrol keamanan yang diuraikan dalam PSS. PSAdan PSS lulus ke stable in Kubernetes `1.25` dan diaktifkan di Amazon EKS secara default. Jika ada PSPs di klaster, pastikan untuk bermigrasi dari PSP ke built-in Kubernetes PSS atau ke policy-as-code solusi sebelum memutakhirkan klaster ke versi. `1.25` Jika Anda tidak bermigrasi dari PSP, Anda mungkin mengalami gangguan pada beban kerja Anda. Lihat informasi yang lebih lengkap di FAQ (PSP) penghapusan kebijakan keamanan Pod.
- Kubernetesversi `1.25` berisi perubahan yang mengubah perilaku fitur yang ada yang dikenal sebagai API Priority and Fairness (APF). APF berfungsi untuk melindungi server API dari potensi kelebihan beban selama periode volume permintaan yang meningkat. Hal ini dilakukan dengan menempatkan pembatasan pada jumlah permintaan bersamaan yang dapat diproses pada waktu tertentu. Hal ini dicapai melalui penerapan tingkat prioritas yang berbeda dan batasan permintaan yang berasal dari berbagai beban kerja atau pengguna. Pendekatan ini memastikan bahwa aplikasi kritis atau permintaan prioritas tinggi menerima perlakuan istimewa, sekaligus mencegah permintaan prioritas yang lebih rendah dari membanjiri server API. Untuk informasi selengkapnya, lihat Prioritas dan Keadilan API dalam Kubernetes dokumentasi atau Prioritas dan Keadilan API dalam Panduan Praktik Terbaik EKS.

  Pembaruan ini diperkenalkan di PR #10352 dan PR #118601. Sebelumnya, APF memperlakukan semua jenis permintaan secara seragam, dengan setiap permintaan menggunakan satu unit dari batas permintaan bersamaan. Perubahan perilaku APF menetapkan unit konkurensi yang lebih tinggi ke `LIST` permintaan karena beban yang sangat berat yang diberikan pada server API oleh permintaan ini. Server API memperkirakan jumlah objek yang akan dikembalikan oleh `LIST` permintaan. Ini menetapkan unit konkurensi yang sebanding dengan jumlah objek yang dikembalikan.

  Setelah memutakhirkan ke versi Amazon EKS `1.25` atau yang lebih tinggi, perilaku yang diperbarui ini dapat menyebabkan beban kerja dengan `LIST` permintaan berat (yang sebelumnya berfungsi tanpa masalah) mengalami pembatasan tarif. Ini akan ditunjukkan oleh kode respons HTTP 429. Untuk menghindari potensi gangguan beban kerja karena tingkat `LIST` permintaan terbatas, kami sangat menyarankan Anda untuk merestrukturisasi

beban kerja Anda untuk mengurangi tingkat permintaan ini. Atau, Anda dapat mengatasi masalah ini dengan menyesuaikan pengaturan APF untuk mengalokasikan lebih banyak kapasitas untuk permintaan penting sekaligus mengurangi kapasitas yang dialokasikan ke yang tidak penting. Untuk informasi selengkapnya tentang teknik mitigasi ini, lihat Mencegah Permintaan yang Diturunkan di Panduan Praktik Terbaik EKS.

- Amazon EKS `1.25` menyertakan penyempurnaan untuk otentikasi klaster yang berisi pustaka yang diperbarui. YAML Jika YAML nilai yang `aws-auth` `ConfigMap` ditemukan di `kube-system` namespace dimulai dengan makro, di mana karakter pertama adalah kurung kurawal, Anda harus menambahkan tanda kutip (“ ”) sebelum dan sesudah kurawal kurawal (). `{ }` Ini diperlukan untuk memastikan bahwa `aws-iam-authenticator` versi `v0.6.3` secara akurat mem-parsing `aws-auth` `ConfigMap` di Amazon EKS`1.25`.
- Versi beta API (`discovery.k8s.io/v1beta1`) dari `EndpointSlice` sudah tidak digunakan lagi Kubernetes `1.21` dan tidak lagi disajikan pada. Kubernetes `1.25` API ini telah diperbarui ke`discovery.k8s.io/v1`. Untuk informasi lebih lanjut, lihat `EndpointSlice`di Kubernetes dokumentasi. The AWS Load Balancer Controller `v2.4.6` dan sebelumnya menggunakan `v1beta1` endpoint untuk berkomunikasi dengan`EndpointSlices`. Jika Anda menggunakan `EndpointSlices` konfigurasi untukAWS Load Balancer Controller, Anda harus memutakhirkan ke AWS Load Balancer Controller `v2.4.7` sebelum memutakhirkan kluster Amazon EKS Anda ke`1.25`. Jika Anda memutakhirkan ke `1.25` saat menggunakan `EndpointSlices` konfigurasi untukAWS Load Balancer Controller, pengontrol akan macet dan mengakibatkan gangguan pada beban kerja Anda. Untuk meng-upgrade controller, lihatApa itu AWS Load Balancer Controller?.

- `SeccompDefault`dipromosikan ke beta di Kubernetes`1.25`. Dengan menyetel `--seccomp-default` flag saat Anda mengonfigurasi`kubelet`, runtime container menggunakan `RuntimeDefault` `seccomp` profilnya, bukan mode unconfined (). `seccomp disabled` Profil default menyediakan serangkaian default keamanan yang kuat, sambil mempertahankan fungsionalitas beban kerja. Meskipun flag ini tersedia, Amazon EKS tidak mengaktifkan flag ini secara default, sehingga perilaku Amazon EKS secara efektif tidak berubah. Jika mau, Anda dapat mulai mengaktifkan ini di node Anda. Untuk lebih jelasnya, lihat tutorial Membatasi Syscalls Container dengan seccomp dalam dokumentasi. Kubernetes
- Support untuk Container Runtime Interface (CRI) untuk Docker (juga dikenal sebagai`Dockershim`) telah dihapus dari Kubernetes `1.24` dan kemudian. Satu-satunya runtime kontainer di Amazon

EKS resmi AMIs untuk Kubernetes `1.24` dan kluster yang lebih baru adalah. `containerd` Sebelum memutakhirkan ke Amazon EKS `1.24` atau yang lebih baru, hapus referensi apa pun ke flag skrip bootstrap yang tidak didukung lagi. Untuk informasi selengkapnya, lihat Amazon EKS mengakhiri dukungan untuk`Dockershim`.

- Dukungan untuk kueri wildcard tidak digunakan lagi dan dihapus di. CoreDNS `1.8.7` CoreDNS `1.9` Ini dilakukan sebagai langkah pengamanan. Kueri wildcard tidak lagi berfungsi dan kembali `NXDOMAIN` alih-alih alamat IP.
- `goaway-chance`Opsi di server Kubernetes API membantu mencegah koneksi `HTTP/2` klien macet pada satu instance server API, dengan menutup koneksi secara acak. Ketika koneksi ditutup, klien akan mencoba untuk menyambung kembali, dan kemungkinan akan mendarat di server API yang berbeda sebagai akibat dari load balancing. Versi Amazon EKS `1.25` telah mengaktifkan `goaway-chance` bendera. Jika beban kerja Anda yang berjalan di klaster Amazon EKS menggunakan klien yang tidak kompatibel dengannya `HTTP GOAWAY`, sebaiknya Anda memperbarui klien agar ditangani `GOAWAY` dengan menyambungkan kembali saat penghentian koneksi.

Untuk Kubernetes `1.25` changelog lengkap, lihat. https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.25.md#changelog-since-v1240

Catatan rilis untuk versi dukungan yang diperluas

Topik ini memberikan perubahan penting yang harus diperhatikan untuk setiap Kubernetes versi dalam dukungan yang diperluas. Saat memutakhirkan, tinjau dengan cermat perubahan yang terjadi antara versi lama dan baru untuk klaster Anda.

Kubernetes1.24

Kubernetes`1.24`sekarang tersedia di Amazon EKS. Untuk informasi selengkapnya Kubernetes`1.24`, lihat pengumuman rilis resmi.

> **Important**
>
> - Dimulai dengan Kubernetes`1.24`, API beta baru tidak diaktifkan di cluster secara default. Secara default, API beta yang ada dan versi baru dari API beta yang ada terus diaktifkan. Amazon EKS mengikuti perilaku yang sama seperti hulu Kubernetes`1.24`. Gerbang fitur yang mengontrol fitur baru untuk operasi API baru dan yang sudah ada diaktifkan secara

default. Ini sejalan dengan huluKubernetes. Untuk informasi selengkapnya, lihat KEP-3136: API Beta Nonaktif secara Default. GitHub

- Support untuk Container Runtime Interface (CRI) untuk Docker (juga dikenal sebagaiDockershim) dihapus dari. Kubernetes `1.24` AMI resmi Amazon containerd EKS memiliki satu-satunya runtime. Sebelum pindah ke Amazon EKS `1.24` atau yang lebih tinggi, Anda harus menghapus referensi apa pun ke flag skrip bootstrap yang tidak didukung lagi. Anda juga harus memastikan bahwa penerusan IP diaktifkan untuk node pekerja Anda. Untuk informasi selengkapnya, lihat Amazon EKS mengakhiri dukungan untuk`Dockershim`.
- Jika Anda sudah Fluentd mengonfigurasi untukContainer Insights, maka Anda harus bermigrasi Fluentd ke Fluent Bit sebelum memperbarui klaster Anda. FluentdParser dikonfigurasi untuk hanya mengurai pesan log dalam format JSON. Tidak seperti`dockerd`, runtime `containerd` kontainer memiliki pesan log yang tidak dalam format JSON. Jika Anda tidak bermigrasi keFluent Bit, beberapa Fluentd's parser yang dikonfigurasi akan menghasilkan sejumlah besar kesalahan di dalam wadah. Fluentd Untuk informasi selengkapnya tentang migrasi, lihat Mengatur Fluent Bit sebagai DaemonSet untuk mengirim log ke CloudWatch Log.
- Di dalam Kubernetes `1.23` dan sebelumnya, sertifikat `kubelet` penyajian dengan IP yang tidak dapat diverifikasi dan Nama Alternatif Subjek DNS (SAN) secara otomatis dikeluarkan dengan SAN yang tidak dapat diverifikasi. SAN yang tidak dapat diverifikasi ini dihilangkan dari sertifikat yang disediakan. Di klaster versi `1.24` dan yang lebih baru, sertifikat `kubelet` penyajian tidak dikeluarkan jika SAN tidak dapat diverifikasi. Ini mencegah perintah `kubectl` exec dan `kubectl` log berfungsi. Untuk informasi selengkapnya, lihat Pertimbangan penandatanganan sertifikat sebelum meningkatkan klaster Anda ke 1,24 Kubernetes.
- Saat memutakhirkan `1.23` kluster Amazon EKS yang menggunakanFluent Bit, Anda harus memastikan bahwa klaster tersebut berjalan `k8s/1.3.12` atau lebih baru. Anda dapat melakukan ini dengan menerapkan kembali file Fluent Bit YAMG terbaru yang berlaku dari. GitHub Untuk informasi selengkapnya, lihat Menyiapkan Fluent Bit di Panduan CloudWatch Pengguna Amazon.

- Anda dapat menggunakan Petunjuk Sadar Topologi untuk menunjukkan preferensi Anda untuk menjaga lalu lintas di zona saat node pekerja klaster digunakan di beberapa zona ketersediaan. Routing lalu lintas dalam zona dapat membantu mengurangi biaya dan meningkatkan kinerja

jaringan. Secara default, Petunjuk Sadar Topologi diaktifkan di Amazon EKS. `1.24` Untuk informasi selengkapnya, lihat Petunjuk Sadar Topologi dalam dokumentasi. Kubernetes

- The `PodSecurityPolicy` (PSP) dijadwalkan untuk dihapus di Kubernetes`1.25`. PSPssedang diganti dengan Pod Security Admission (PSA). PSA adalah pengontrol masuk bawaan yang menggunakan kontrol keamanan yang diuraikan dalam Standar Keamanan Pod (PSS). PSA dan PSS keduanya fitur beta dan diaktifkan di Amazon EKS secara default. Untuk mengatasi penghapusan PSP dalam versi`1.25`, kami sarankan Anda menerapkan PSS di Amazon EKS. Untuk informasi selengkapnya, lihat Menerapkan Standar Keamanan Pod di Amazon EKS di AWS blog.
- `client.authentication.k8s.io/v1alpha1` ExecCredential Itu dihapus di Kubernetes`1.24`. ExecCredential API umumnya tersedia di Kubernetes`1.22`. Jika Anda menggunakan plugin kredensi client-go yang mengandalkan `v1alpha1` API, hubungi distributor plugin Anda tentang cara bermigrasi ke API. `v1`
- Untuk Kubernetes`1.24`, kami menyumbangkan fitur ke proyek Autoscaler Cluster hulu yang menyederhanakan penskalaan grup node terkelola Amazon EKS ke dan dari nol node. Sebelumnya, agar Cluster Autoscaler memahami sumber daya, label, dan noda grup node terkelola yang diskalakan ke nol node, Anda perlu menandai grup Auto Scaling Amazon EC2 yang mendasarinya dengan detail node yang menjadi tanggung jawabnya. Sekarang, ketika tidak ada node yang berjalan di grup node terkelola, Cluster Autoscaler memanggil operasi Amazon EKS `DescribeNodegroup` API. Operasi API ini memberikan informasi yang dibutuhkan Cluster Autoscaler dari resource, label, dan taint grup node terkelola. Fitur ini mengharuskan Anda menambahkan `eks:DescribeNodegroup` izin ke kebijakan IAM akun layanan Cluster Autoscaler. Ketika nilai tag Cluster Autoscaler pada grup Auto Scaling yang menyalakan grup node terkelola Amazon EKS bertentangan dengan grup node itu sendiri, Cluster Autoscaler lebih memilih nilai tag grup Auto Scaling. Ini agar Anda dapat mengganti nilai sesuai kebutuhan. Untuk informasi selengkapnya, lihat Penskalaan otomatis.
- Jika Anda bermaksud menggunakan Inferentia atau jenis Trainium instance dengan Amazon EKS`1.24`, Anda harus meningkatkan ke plugin AWS Neuron perangkat versi 1.9.3.0 atau yang lebih baru. Untuk informasi lebih lanjut, lihat rilis Neuron K8 [1.9.3.0] di Dokumentasi. AWS Neuron
- `Containerd`telah `IPv6` diaktifkan untuk`Pods`, secara default. Ini menerapkan pengaturan kernel node ke ruang nama Pod jaringan. Karena itu, kontainer dalam Pod mengikat ke alamat `IPv4` loopback (`127.0.0.1`) dan `IPv6` (`::1`). `IPv6`adalah protokol default untuk komunikasi. Sebelum memperbarui klaster Anda ke versi`1.24`, kami sarankan Anda menguji multi-container Pods Anda. Ubah aplikasi sehingga mereka dapat mengikat ke semua alamat IP pada antarmuka loopback.

Mayoritas pustaka mengaktifkan IPv6 pengikatan, yang kompatibel dengan versi sebelumnya. IPv4 Jika tidak memungkinkan untuk memodifikasi kode aplikasi Anda, Anda memiliki dua opsi:

- Jalankan `init` wadah dan atur `disable ipv6` ke `true` (`sysctl -w net.ipv6.conf.all.disable_ipv6=1`).
- Konfigurasikan webhook penerimaan yang bermutasi untuk menyuntikkan `init` wadah di samping aplikasi Anda. Pods

Jika Anda perlu memblokir IPv6 Pods semua node, Anda mungkin harus IPv6 menonaktifkan instance Anda.

- `goaway-chance`Opsi di server Kubernetes API membantu mencegah koneksi HTTP/2 klien macet pada satu instance server API, dengan menutup koneksi secara acak. Ketika koneksi ditutup, klien akan mencoba untuk menyambung kembali, dan kemungkinan akan mendarat di server API yang berbeda sebagai akibat dari load balancing. Versi Amazon EKS `1.24` telah mengaktifkan `goaway-chance` bendera. Jika beban kerja Anda yang berjalan di klaster Amazon EKS menggunakan klien yang tidak kompatibel dengannya `HTTP GOAWAY`, sebaiknya Anda memperbarui klien agar ditangani `GOAWAY` dengan menyambungkan kembali saat penghentian koneksi.

Untuk Kubernetes `1.24` changelog lengkap, lihat. https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.24.md#changelog-since-v1230

Kubernetes1.23

Kubernetes`1.23`sekarang tersedia di Amazon EKS. Untuk informasi selengkapnya Kubernetes`1.23`, lihat pengumuman rilis resmi.

> **Important**
>
> - Fitur migrasi volume Kubernetes in-tree to container storage interface (CSI) diaktifkan. Fitur ini memungkinkan penggantian plugin penyimpanan Kubernetes in-tree yang ada untuk Amazon EBS dengan driver Amazon EBS CSI yang sesuai. Untuk informasi selengkapnya, lihat Kubernetes1.17 Fitur: Migrasi Volume Kubernetes In-Tree ke CSI Berpindah ke Beta di blog. Kubernetes
>
>   Fitur ini menerjemahkan API in-tree ke API CSI yang setara dan mendelegasikan operasi ke driver CSI pengganti. Dengan fitur ini, jika Anda menggunakan yang ada `StorageClassPersistentVolume`,, dan `PersistentVolumeClaim` objek yang

termasuk dalam beban kerja ini, kemungkinan tidak akan ada perubahan nyata. Fitur ini memungkinkan Kubernetes untuk mendelegasikan semua operasi manajemen penyimpanan dari plugin in-tree ke driver CSI. Jika Anda menggunakan volume Amazon EBS di klaster yang ada, instal driver Amazon EBS CSI di klaster sebelum memperbarui klaster ke versi. `1.23` Jika Anda tidak menginstal driver sebelum memperbarui cluster yang ada, interupsi pada beban kerja Anda mungkin terjadi. Jika Anda berencana untuk menerapkan beban kerja yang menggunakan volume Amazon EBS di `1.23` klaster baru, instal driver Amazon EBS CSI di klaster Anda sebelum menerapkan beban kerja klaster Anda. Untuk petunjuk tentang cara menginstal driver Amazon EBS CSI di cluster Anda, lihat. Driver CSI Amazon EBS Untuk pertanyaan umum tentang fitur migrasi, lihatMigrasi Amazon EBS CSI pertanyaan yang sering diajukan.

- Extended Support untuk Amazon EKS Windows AMI yang dioptimalkan yang diterbitkan oleh AWS tidak tersedia untuk Kubernetes versi `1.23` tetapi tersedia untuk Kubernetes versi `1.24` dan lebih tinggi.

- Kubernetesberhenti mendukung `dockershim` dalam versi `1.20` dan dihapus `dockershim` dalam versi`1.24`. Untuk informasi lebih lanjut, lihat KubernetesMoving on FromDockershim: Commitments and Next Steps di Kubernetes blog. Amazon EKS akan mengakhiri dukungan untuk `dockershim` memulai dalam versi Amazon EKS`1.24`. Dimulai dengan versi Amazon EKS`1.24`, AMI resmi Amazon `containerd` EKS akan menjadi satu-satunya runtime.

  Meskipun versi Amazon EKS `1.23` terus mendukung`dockershim`, kami sarankan Anda mulai menguji aplikasi Anda sekarang untuk mengidentifikasi dan menghapus Docker dependensi apa pun. Dengan cara ini, Anda siap untuk memperbarui cluster Anda ke versi`1.24`. Untuk informasi selengkapnya tentang `dockershim` penghapusan, lihatAmazon EKS mengakhiri dukungan untuk`Dockershim`.
- Kubernetesjaringan `IPv4` `IPv6` lulusan/dual-stack untukPods, layanan, dan node untuk ketersediaan umum. Namun, Amazon EKS dan Amazon VPC CNI plugin for Kubernetes tidak mendukung jaringan dual-stack. Cluster Anda dapat menetapkan `IPv4` atau `IPv6` alamat ke Pods dan layanan, tetapi tidak dapat menetapkan kedua jenis alamat.
- KubernetesMenyelesaikan fitur Pod Security Admission (PSA) ke versi beta. Fitur ini diaktifkan secara default. Untuk informasi selengkapnya, lihat Penerimaan Keamanan Pod di Kubernetes dokumentasi. PSA menggantikan pengontrol penerimaan Pod Security Policy (PSP). Pengontrol penerimaan PSP tidak didukung dan dijadwalkan untuk dihapus dalam Kubernetes versi`1.25`.

Pengontrol PSP penerimaan memberlakukan standar Pod keamanan di Pods dalam namespace berdasarkan label namespace tertentu yang menetapkan tingkat penegakan. Untuk informasi selengkapnya, lihat Pod Security Standards (PSS) dan Pod Security Admission (PSA) di panduan praktik terbaik Amazon EKS.

- `kube-proxy`Gambar yang digunakan dengan cluster sekarang menjadi gambar dasar minimal yang dikelola oleh Amazon EKS Distro (EKS-D). Gambar berisi paket minimal dan tidak memiliki shell atau manajer paket.
- Kuberneteslulus wadah fana ke beta. Kontainer ephemeral adalah kontainer sementara yang berjalan di namespace yang sama dengan yang sudah ada. Pod Anda dapat menggunakannya untuk mengamati status Pods dan wadah untuk tujuan pemecahan masalah dan debugging. Ini sangat berguna untuk pemecahan masalah interaktif ketika tidak `kubectl exec` cukup karena wadah telah macet atau gambar kontainer tidak menyertakan utilitas debugging. Contoh wadah yang menyertakan utilitas debugging adalah gambar distroless. Untuk informasi selengkapnya, lihat Debugging dengan wadah debug singkat dalam dokumentasi. Kubernetes
- Kubernetesmeluluskan API `HorizontalPodAutoscaler` `autoscaling/v2` stabil ke ketersediaan umum. `HorizontalPodAutoscaler`API `autoscaling/v2beta2` tidak digunakan lagi. Ini tidak akan tersedia di`1.26`.
- `goaway-chance`Opsi di server Kubernetes API membantu mencegah koneksi `HTTP/2` klien macet pada satu instance server API, dengan menutup koneksi secara acak. Ketika koneksi ditutup, klien akan mencoba untuk menyambung kembali, dan kemungkinan akan mendarat di server API yang berbeda sebagai akibat dari load balancing. Versi Amazon EKS `1.23` telah mengaktifkan `goaway-chance` bendera. Jika beban kerja Anda yang berjalan di klaster Amazon EKS menggunakan klien yang tidak kompatibel dengannya `HTTP GOAWAY`, sebaiknya Anda memperbarui klien agar ditangani `GOAWAY` dengan menyambungkan kembali saat penghentian koneksi.

Untuk Kubernetes `1.23` changelog lengkap, lihat. https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.23.md#changelog-since-v1220

Catatan rilis untuk versi 1.21 dan 1.22

> **Important**
>
> Anda tidak dapat membuat cluster baru dengan versi ini.

Topik ini memberikan perubahan penting yang harus diperhatikan untuk versi `1.22` dan`1.21`. Saat memutakhirkan, tinjau dengan cermat perubahan yang terjadi antara versi lama dan baru untuk klaster Anda.

Kubernetesversi **`1.22`**

Pengontrol penerimaan berikut diaktifkan untuk semua versi `1.22` platform:`DefaultStorageClass`,,,`DefaultTolerationSeconds`,`LimitRanger`,`MutatingAdmissionW` `CertificateSubjectRestrictionRuntimeClass`, dan`DefaultIngressClass`.

Versi Kubernetes	Amazon EKS versi platform	Catatan rilis	Tanggal rilis
`1.22.17`	`eks.26`	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	April 1, 2024
`1.22.17`	`eks.14`	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juni 30, 2023
`1.22.17`	`eks.13`	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	9 Juni 2023
`1.22.17`	`eks.12`	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	5 Mei 2023
`1.22.17`	`eks.11`	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	24 Maret 2023
`1.22.16`	`eks.10`	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	27 Januari 2023

Versi Kubernetes	Amazon EKS versi platform	Catatan rilis	Tanggal rilis
1.22.15	eks.9	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Desember 5, 2022
1.22.15	eks.8	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	18 November 2022
1.22.15	eks.7	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	7 November 2022
1.22.13	eks.6	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	21 September 2022
1.22.10	eks.5	Versi platform baru dengan etcd ketahanan yang ditingkatkan.	Agustus 15, 2022
1.22.10	eks.4	Versi platform baru dengan perbaikan dan penyempurnaan keamanan. Versi platform ini juga memperkenalkan pengontrol penandaan baru yang menandai semua node pekerja aws:eks:cluster-name untuk memudahkan mengalokasikan biaya untuk node pekerja ini. Untuk informasi selengkapnya, lihat Menandai sumber daya Anda untuk penagihan.	21 Juli 2022
1.22.10	eks.3	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juli 7, 2022

Versi Kubernetes	Amazon EKS versi platform	Catatan rilis	Tanggal rilis
1.22.9	eks.2	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	31 Mei 2022
1.22.6	eks.1	Rilis awal Kubernetes versi 1.22 untuk Amazon EKS.	4 April 2022

Kubernetesversi **1.21**

Pengontrol penerimaan berikut diaktifkan untuk semua versi 1.21 platform:DefaultStorageClass,,,DefaultTolerationSeconds,LimitRanger,MutatingAdmissionW CertificateSubjectRestrictionRuntimeClass, danDefaultIngressClass.

Versi Kubernetes	Amazon EKS versi platform	Catatan rilis	Tanggal rilis
1.21.14	eks.31	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	April 1, 2024
1.21.14	eks.18	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	9 Juni 2023
1.21.14	eks.17	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	5 Mei 2023
1.21.14	eks.16	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	24 Maret 2023

Versi Kubernetes	Amazon EKS versi platform	Catatan rilis	Tanggal rilis
1.21.14	eks.15	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	27 Januari 2023
1.21.14	eks.14	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Desember 5, 2022
1.21.14	eks.13	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	18 November 2022
1.21.14	eks.12	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	7 November 2022
1.21.13	eks.11	Versi platform baru dengan `etcd` ketahanan yang ditingkatkan.	10 Oktober 2022
1.21.13	eks.10	Versi platform baru dengan `etcd` ketahanan yang ditingkatkan.	Agustus 15, 2022

Versi Kubernetes	Amazon EKS versi platform	Catatan rilis	Tanggal rilis
1.21.13	eks.9	Versi platform baru dengan perbaikan dan penyempurnaan keamanan. Versi platform ini juga memperkenalkan pengontrol penandaan baru yang menandai semua node pekerja aws:eks:cluster-name untuk memudahkan mengalokasikan biaya untuk node pekerja ini. Untuk informasi selengkapnya, lihat Menandai sumber daya Anda untuk penagihan.	21 Juli 2022
1.21.13	eks.8	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juli 7, 2022
1.21.12	eks.7	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	31 Mei 2022

Versi Kubernetes	Amazon EKS versi platform	Catatan rilis	Tanggal rilis
1.21.9	eks.6	AWS Security Token Service Titik akhir dikembalikan kembali ke titik akhir global dari versi platform sebelumnya. Jika Anda ingin menggunakan endpoint Regional saat menggunakan peran IAM untuk akun layanan, maka Anda harus mengaktif kannya. Untuk petunjuk tentang cara mengaktifkan titik akhir regional, lihatKonfigurasikan AWS Security Token Service titik akhir untuk akun layanan.	8 April 2022
1.21.5	eks.5	Saat menggunakanIAM role untuk akun layanan, titik akhir AWS Security Token Service Regional sekarang digunakan secara default, bukan titik akhir global. Namun, perubahan ini dikembalikan ke titik akhir global. eks.6 Penjadwal Fargate yang diperbarui menyediakan node pada tingkat yang jauh lebih tinggi selama penerapan besar.	10 Maret 2022
1.21.5	eks.4	Versi 1.10.1-eksbuild.1 Amazon VPC CNI yang dikelola sendiri dan add-on Amazon EKS sekarang menjadi versi default yang digunakan.	13 Desember 2021

Versi Kubernetes	Amazon EKS versi platform	Catatan rilis	Tanggal rilis
1.21.2	eks.3	Versi platform baru dengan dukungan untuk manajemen Windows IPv4 alamat pada VPC Resource Controller yang berjalan di bidang Kubernetes kontrol. Menambahkan arahan Kubernetes filter untuk logging Fargate Fluent Bit.	November 8, 2021
1.21.2	eks.2	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	September 17, 2021
1.21.2	eks.1	Rilis awal Kubernetes versi 1.21 untuk Amazon EKS.	19 Juli 2021

Amazon EKS versi platform

Versi platform Amazon EKS mewakili kemampuan bidang kontrol cluster Amazon EKS, seperti flag server Kubernetes API yang diaktifkan, serta versi Kubernetes patch saat ini. Setiap versi Kubernetes minor memiliki satu atau lebih versi platform Amazon EKS terkait. Versi platform untuk versi Kubernetes minor yang berbeda bersifat independen. Anda dapat mengambil versi platform cluster Anda saat ini menggunakan AWS CLI atau AWS Management Console. Jika Anda memiliki cluster lokal aktif AWS Outposts, lihat Amazon EKS versi platform cluster lokal alih-alih topik ini.

Ketika versi Kubernetes minor baru tersedia di Amazon EKS, seperti 1.29, versi platform Amazon EKS awal untuk versi Kubernetes minor tersebut dimulai pada`eks.1`. Namun, Amazon EKS merilis versi platform baru secara berkala untuk mengaktifkan pengaturan bidang Kubernetes kontrol baru dan untuk memberikan perbaikan keamanan.

Saat versi platform Amazon EKS baru tersedia untuk versi minor:

- Nomor versi platform Amazon EKS bertambah (`eks.`*n+1*).

- Amazon EKS secara otomatis memutakhirkan semua cluster yang ada ke versi platform Amazon EKS terbaru untuk versi Kubernetes minor yang sesuai. Pemutakhiran otomatis versi platform Amazon EKS yang sudah ada diluncurkan secara bertahap. Proses peluncuran mungkin memakan waktu lama. Jika Anda memerlukan fitur versi platform Amazon EKS terbaru segera, Anda harus membuat klaster Amazon EKS baru.

  Jika klaster Anda memiliki lebih dari dua versi platform di belakang versi platform saat ini, kemungkinan Amazon EKS tidak dapat memperbarui cluster Anda secara otomatis. Untuk detail tentang apa yang dapat menyebabkan ini, lihat Versi platform Amazon EKS lebih dari dua versi di belakang versi platform saat ini.
- Amazon EKS boleh menerbitkan AMI simpul baru dengan versi patch yang sesuai. Namun, semua versi tambalan kompatibel antara bidang kontrol EKS dan AMI simpul untuk versi Kubernetes minor tertentu.

Versi platform Amazon EKS baru tidak memperlihatkan perubahan yang dapat merusak atau tidak menyebabkan gangguan layanan.

Cluster selalu dibuat dengan versi platform Amazon EKS terbaru yang tersedia (`eks.`*n*) untuk Kubernetes versi yang ditentukan. Jika Anda memperbarui klaster ke versi Kubernetes minor baru, klaster Anda akan menerima versi platform Amazon EKS saat ini untuk versi Kubernetes minor yang Anda perbarui.

Versi platform Amazon EKS saat ini dan terbaru dideskripsikan dalam tabel berikut.

Kubernetesversi `1.29`

Pengontrol penerimaan berikut diaktifkan untuk semua versi `1.29` platform:`NodeRestriction`,,`ExtendedResourceToleration`, `NamespaceLifecycle`,`LimitRanger`,`ServiceAccount`,`TaintNodesByCondition`,`PodSecurity`,`Pri` `MutatingAdmissionWebhook`,`ValidatingAdmissionWebhook`,`ResourceQuota`.

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
`1.29.1`	`eks.5`	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 29, 2024

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.29.1	eks.4	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 20, 2024
1.29.1	eks.3	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 12, 2024
1.29.0	eks.1	Rilis awal versi 1.29 Kubernetes untuk EKS. Untuk informasi selengkapnya, lihat Kubernetes1.29.	23 Januari 2024

Kubernetesversi **1.28**

Pengontrol penerimaan berikut diaktifkan untuk semua versi 1.28 platform:NodeRestriction,,,ExtendedResourceToleration, NamespaceLifecycleLimitRanger,ServiceAccount,TaintNodesByCondition,PodSecurity,Pri MutatingAdmissionWebhookValidatingAdmissionWebhook,ResourceQuota.

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.28.7	eks.11	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 29, 2024
1.28.7	eks.10	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 20, 2024
1.28.6	eks.9	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 12, 2024

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.28.5	eks.7	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Januari 17, 2024
1.28.4	eks.6	Versi platform baru dengan entri akses, perbaikan keamanan, dan penyempurnaan.	14 Desember 2023
1.28.4	eks.5	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Desember 12, 2023
1.28.3	eks.4	Versi platform baru dengan Identitas EKS Pod, perbaikan keamanan dan penyempurnaan.	10 November 2023
1.28.3	eks.3	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	3 November 2023
1.28.2	eks.2	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	16 Oktober 2023
1.28.1	eks.1	Rilis awal versi 1.28 Kubernetes untuk EKS. Untuk informasi selengkapnya, lihat Kubernetes1.28.	26 September 2023

Kubernetesversi **1.27**

Pengontrol penerimaan berikut diaktifkan untuk semua versi 1.27 platform:NodeRestriction,,,ExtendedResourceToleration, NamespaceLifecycleLimitRanger,ServiceAccount,TaintNodesByCondition,PodSecurity,Pri MutatingAdmissionWebhookValidatingAdmissionWebhook,ResourceQuota.

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.27.11	eks.15	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 29, 2024
1.27.11	eks.14	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 20, 2024
1.27.10	eks.13	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 12, 2024
1.27.9	eks.11	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Januari 17, 2024
1.27.8	eks.10	Versi platform baru dengan entri akses, perbaikan keamanan, dan penyempurnaan.	14 Desember 2023
1.27.8	eks.9	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Desember 12, 2023
1.27.7	eks.8	Versi platform baru dengan Identitas EKS Pod, perbaikan keamanan dan penyempurnaan.	10 November 2023
1.27.7	eks.7	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	3 November 2023
1.27.6	eks.6	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	16 Oktober 2023

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.27.4	eks.5	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Agustus 30, 2023
1.27.4	eks.4	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juli 30, 2023
1.27.3	eks.3	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juni 30, 2023
1.27.2	eks.2	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	9 Juni 2023
1.27.1	eks.1	Rilis awal versi 1.27 Kubernetes untuk EKS. Untuk informasi selengkapnya, lihat Kubernetes1.27.	24 Mei 2023

Kubernetesversi **1.26**

Pengontrol penerimaan berikut diaktifkan untuk semua versi 1.26 platform:NodeRestriction,,,ExtendedResourceToleration, NamespaceLifecycleLimitRanger,ServiceAccount,TaintNodesByCondition,PodSecurity,Pri
MutatingAdmissionWebhookValidatingAdmissionWebhook,ResourceQuota.

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.26.14	eks.16	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 29, 2024

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.26.14	eks.15	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 20, 2024
1.26.13	eks.14	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 12, 2024
1.26.12	eks.12	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Januari 17, 2024
1.26.11	eks.11	Versi platform baru dengan entri akses, perbaikan keamanan, dan penyempurnaan.	14 Desember 2023
1.26.11	eks.10	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Desember 12, 2023
1.26.10	eks.9	Versi platform baru dengan Identitas EKS Pod, perbaikan keamanan dan penyempurnaan.	10 November 2023
1.26.10	eks.8	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	3 November 2023
1.26.9	eks.7	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	16 Oktober 2023
1.26.7	eks.6	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Agustus 30, 2023

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.26.7	eks.5	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juli 30, 2023
1.26.6	eks.4	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juni 30, 2023
1.26.5	eks.3	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	9 Juni 2023
1.26.4	eks.2	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	5 Mei 2023
1.26.2	eks.1	Rilis awal versi 1.26 Kubernetes untuk EKS. Untuk informasi selengkapnya, lihat Kubernetes1.26.	11 April 2023

Kubernetesversi **1.25**

Pengontrol penerimaan berikut diaktifkan untuk semua versi 1.25 platform:NodeRestriction,,,ExtendedResourceToleration, NamespaceLifecycleLimitRanger,ServiceAccount,TaintNodesByCondition,PodSecurity,Pri MutatingAdmissionWebhookValidatingAdmissionWebhook,ResourceQuota.

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.25.16	eks.17	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 29, 2024

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.25.16	eks.16	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 20, 2024
1.25.16	eks.15	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 12, 2024
1.25.16	eks.13	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Januari 17, 2024
1.25.16	eks.12	Versi platform baru dengan entri akses, perbaikan keamanan, dan penyempurnaan.	14 Desember 2023
1.25.16	eks.11	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Desember 12, 2023
1.25.15	eks.10	Versi platform baru dengan Identitas EKS Pod, perbaikan keamanan dan penyempurnaan.	10 November 2023
1.25.15	eks.9	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	3 November 2023
1.25.14	eks.8	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	16 Oktober 2023
1.25.12	eks.7	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Agustus 30, 2023

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.25.12	eks.6	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juli 30, 2023
1.25.11	eks.5	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juni 30, 2023
1.25.10	eks.4	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	9 Juni 2023
1.25.9	eks.3	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	5 Mei 2023
1.25.8	eks.2	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	24 Maret 2023
1.25.6	eks.1	Rilis awal versi 1.25 Kubernetes untuk EKS. Untuk informasi selengkapnya, lihat Kubernetes1,25.	21 Februari 2023

Kubernetesversi **1.24**

Pengontrol penerimaan berikut diaktifkan untuk semua versi 1.24 platform:CertificateApproval,,,CertificateSigning,CertificateSubjectRestriction,Defau
StorageObjectInUseProtectionTaintNodesByCondition, danValidatingAdmissionWebhook.

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.24.17	eks.20	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 29, 2024
1.24.17	eks.19	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 20, 2024
1.24.17	eks.18	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 12, 2024
1.24.17	eks.16	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Januari 17, 2024
1.24.17	eks.15	Versi platform baru dengan entri akses, perbaikan keamanan, dan penyempurnaan.	14 Desember 2023
1.24.17	eks.14	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Desember 12, 2023
1.24.17	eks.13	Versi platform baru dengan Identitas EKS Pod, perbaikan keamanan dan penyempurnaan.	10 November 2023
1.24.17	eks.12	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	3 November 2023
1.24.17	eks.11	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	16 Oktober 2023

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.24.16	eks.10	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Agustus 30, 2023
1.24.16	eks.9	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juli 30, 2023
1.24.15	eks.8	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juni 30, 2023
1.24.14	eks.7	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	9 Juni 2023
1.24.13	eks.6	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	5 Mei 2023
1.24.12	eks.5	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	24 Maret 2023
1.24.8	eks.4	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	27 Januari 2023
1.24.7	eks.3	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Desember 5, 2022
1.24.7	eks.2	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	18 November 2022

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.24.7	eks.1	Rilis awal versi 1.24 Kubernetes untuk EKS. Untuk informasi selengkapnya, lihat Kubernetes1.24.	15 November 2022

Kubernetesversi **1.23**

Pengontrol penerimaan berikut diaktifkan untuk semua versi 1.23 platform:CertificateApproval,,,CertificateSigning,CertificateSubjectRestriction,DefaultStorageClass,DefaultTolerationSeconds,ExtendedResourceToleration,LimitRanger,MutatingAdmissionWebhook,NamespaceLifecycle,NodeRestriction,PersistentVolumeClaimResize,Priority,PodSecurity,ResourceQuota,RuntimeClass,ServiceAccount,StorageObjectInUseProtection,TaintNodesByCondition, danValidatingAdmissionWebhook.

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.23.17	eks.22	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 29, 2024
1.23.17	eks.21	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 20, 2024
1.23.17	eks.20	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Maret 12, 2024
1.23.17	eks.18	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Januari 17, 2024
1.23.17	eks.17	Versi platform baru dengan entri akses, perbaikan keamanan, dan penyempurnaan.	14 Desember 2023

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.23.17	eks.16	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Desember 12, 2023
1.23.17	eks.15	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	10 November 2023
1.23.17	eks.14	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	3 November 2023
1.23.17	eks.13	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	16 Oktober 2023
1.23.17	eks.12	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Agustus 30, 2023
1.23.17	eks.11	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juli 30, 2023
1.23.17	eks.10	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Juni 30, 2023
1.23.17	eks.9	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	9 Juni 2023
1.23.17	eks.8	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	5 Mei 2023

Versi Kubernetes	Versi platform EKS	Catatan rilis	Tanggal rilis
1.23.17	eks.7	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	24 Maret 2023
1.23.14	eks.6	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	27 Januari 2023
1.23.13	eks.5	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	Desember 5, 2022
1.23.13	eks.4	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	18 November 2022
1.23.12	eks.3	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	7 November 2022
1.23.10	eks.2	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	21 September 2022
1.23.7	eks.1	Rilis awal versi 1.23 Kubernetes untuk EKS. Untuk informasi selengkapnya, lihat Kubernetes1.23.	Agustus 11, 2022

Dapatkan versi platform saat ini

Untuk mendapatkan versi platform saat ini untuk cluster Anda (konsol)

1. Buka konsol Amazon EKS.
2. Pada panel navigasi, silakan pilih Klaster.
3. Dalam daftar cluster, pilih Nama Cluster untuk memeriksa versi platform.

4. Pilih tab Gambaran Umum.
5. Versi Platform tersedia di bagian Detail.

Untuk mendapatkan versi platform saat ini untuk cluster Anda (AWS CLI)

1. Tentukan Nama cluster yang ingin Anda periksa versi platform.
2. Jalankan perintah berikut:

```
aws eks describe-cluster --name my-cluster --query cluster.platformVersion
```

Contoh output adalah sebagai berikut.

```
"eks.10"
```

Penskalaan otomatis

Autoscaling adalah fungsi yang secara otomatis mengukur sumber daya Anda keluar dan masuk untuk memenuhi tuntutan yang berubah. Ini adalah mayorKubernetesfungsi yang seharusnya membutuhkan sumber daya manusia yang luas untuk bekerja secara manual.

Amazon EKS mendukung dua produk penskalaan otomatis:

Karpenter

Karpenteradalah fleksibel, berkinerja tinggiKubernetescluster autoscaler yang membantu meningkatkan ketersediaan aplikasi dan efisiensi cluster.Karpentermeluncurkan sumber daya komputasi berukuran tepat (misalnya, instans Amazon EC2) sebagai respons terhadap perubahan beban aplikasi dalam waktu kurang dari satu menit. Melalui integrasiKubernetesbersamaAWS,Karpenterdapat ketentuan just-in-time menghitung sumber daya yang secara tepat memenuhi persyaratan beban kerja Anda.Karpentersecara otomatis menyediakan sumber daya komputasi baru berdasarkan persyaratan spesifik beban kerja cluster. Ini termasuk persyaratan komputasi, penyimpanan, akselerasi, dan penjadwalan. Amazon EKS mendukung cluster menggunakanKarpenter, meskipunKarpenterbekerja dengan kesesuaian apa punKubernetesgugus. Untuk informasi lebih lanjut, lihatKarpenterdokumentasi.

Autoscaler klaster

TheKubernetesCluster Autoscaler secara otomatis menyesuaikan jumlah node di klaster Anda ketika pod gagal atau dijadwal ulang ke node lain. Cluster Autoscaler menggunakan grup Auto Scaling. Untuk informasi selengkapnya, lihat Autoscaler Klaster di AWS.

Kelola akses

Pelajari cara mengelola akses ke klaster Amazon EKS Anda. Menggunakan Amazon EKS membutuhkan pengetahuan tentang bagaimana keduanya Kubernetes dan AWS Identity and Access Management (AWS IAM) menangani kontrol akses.

Bagian ini meliputi:

the section called “Berikan akses ke API Kubernetes”— Pelajari cara mengaktifkan aplikasi atau pengguna untuk mengautentikasi ke Kubernetes API. Anda dapat menggunakan entri akses, aws-auth ConfigMap, atau penyedia OIDC eksternal.

the section called “Akses cluster saya dengan kubectl”— Pelajari cara mengonfigurasi kubectl untuk berkomunikasi dengan cluster Amazon EKS Anda. Gunakan AWS CLI untuk membuat file kubeconfig.

the section called “Berikan akses beban kerja ke AWS”— Pelajari cara mengaitkan akun Kubernetes layanan dengan Peran AWS IAM. Anda dapat menggunakan Pod Identity atau IAM Roles for Service Accounts (IRSA).

Tugas Umum:

- Berikan pengembang akses ke Kubernetes API. Lihat Kubernetes sumber daya di AWS Management Console.
  - Solusi: Gunakan Entri Akses untuk mengaitkan izin Kubernetes RBAC dengan Pengguna atau Peran AWS IAM.
- Konfigurasikan kubectl untuk berbicara dengan klaster Amazon EKS menggunakan Credentials. AWS
  - Solusi: Gunakan AWS CLI untuk membuat file kubeconfig.
- Gunakan penyedia identitas eksternal, seperti Ping Identity, untuk mengautentikasi pengguna ke Kubernetes API.
  - Solusi: Tautkan penyedia OIDC eksternal.
- Berikan beban kerja pada Kubernetes klaster Anda kemampuan untuk memanggil AWS API.
  - Solusi: Gunakan Pod Identity untuk mengaitkan Peran AWS IAM ke Akun Kubernetes Layanan.

Latar Belakang:

- Pelajari cara kerja Akun Layanan Kubernetes.
- Tinjau Model Kubernetes Role Based Access Control (RBAC)
- Untuk informasi selengkapnya tentang mengelola akses ke AWS sumber daya, lihat Panduan Pengguna AWS IAM. Atau, ikuti pelatihan pengantar gratis tentang penggunaan AWS IAM.

Berikan akses ke Kubernetes API

Cluster Anda memiliki titik akhir Kubernetes API. Kubectl menggunakan API ini. Anda dapat mengautentikasi ke API ini menggunakan dua jenis identitas:

- Prinsipal AWS Identity and Access Management (IAM) (peran atau pengguna) - Jenis ini memerlukan otentikasi ke IAM. Pengguna dapat masuk AWS sebagai pengguna IAM atau dengan identitas federasi dengan menggunakan kredensil yang disediakan melalui sumber identitas. Pengguna hanya dapat masuk dengan identitas federasi jika administrator Anda sebelumnya menyiapkan federasi identitas menggunakan peran IAM. Ketika pengguna mengakses AWS dengan menggunakan federasi, mereka secara tidak langsung mengambil peran. Saat pengguna menggunakan jenis identitas ini, Anda:
  - Dapat menetapkan mereka Kubernetes izin sehingga mereka dapat bekerja dengan Kubernetes objek di cluster Anda. Untuk informasi selengkapnya tentang cara menetapkan izin ke prinsipal IAM Anda sehingga mereka dapat mengakses Kubernetes objek di klaster Anda, lihat. Kelola entri akses
  - Dapat menetapkan mereka izin IAM sehingga mereka dapat bekerja dengan kluster Amazon EKS Anda dan sumber dayanya menggunakan Amazon EKS API,,, AWS CLI AWS CloudFormation, AWS Management Console atau. `eksctl` Untuk informasi selengkapnya, lihat Tindakan yang ditentukan oleh Amazon Elastic Kubernetes Service di Referensi Otorisasi Layanan.
  - Node bergabung dengan cluster Anda dengan mengasumsikan peran IAM. Kemampuan untuk mengakses klaster Anda menggunakan prinsipal IAM disediakan oleh AWS IAM Authenticator for, Kubernetes yang berjalan pada bidang kontrol Amazon EKS.
- Pengguna di penyedia OpenID Connect (OIDC) Anda sendiri — Jenis ini memerlukan otentikasi ke OIDCpenyedia Anda. Untuk informasi selengkapnya tentang menyiapkan OIDC penyedia Anda sendiri dengan kluster Amazon EKS Anda, lihatMengautentikasi pengguna untuk klaster Anda dari penyedia OpenID Connect identitas. Saat pengguna menggunakan jenis identitas ini, Anda:

- Dapat menetapkan mereka Kubernetes izin sehingga mereka dapat bekerja dengan Kubernetes objek di cluster Anda.
- Tidak dapat menetapkan mereka izin IAM sehingga mereka dapat bekerja dengan kluster Amazon EKS Anda dan sumber dayanya menggunakan Amazon EKS API,,, AWS CLI AWS CloudFormation, AWS Management Console atau. `eksctl`

Anda dapat menggunakan kedua jenis identitas dengan cluster Anda. Metode otentikasi IAM tidak dapat dinonaktifkan. Metode otentikasi OIDC adalah opsional.

Kaitkan Identitas IAM dengan Izin Kubernetes

AWS IAM Authenticator for Kubernetes diinstal pada bidang kontrol cluster Anda. Ini memungkinkan AWS Identity and Access Management(IAM) prinsipal (peran dan pengguna) yang Anda izinkan untuk mengakses Kubernetes sumber daya di cluster Anda. Anda dapat mengizinkan prinsipal IAM untuk mengakses Kubernetes objek di cluster Anda menggunakan salah satu metode berikut:

- Membuat entri akses — Jika klaster Anda berada pada atau lebih lambat dari versi platform yang tercantum di bagian Prasyarat untuk Kubernetes versi klaster Anda, kami sarankan Anda menggunakan opsi ini.

  Gunakan entri akses untuk mengelola Kubernetes izin prinsipal IAM dari luar cluster. Anda dapat menambahkan dan mengelola akses ke cluster dengan menggunakan EKS API, AWS Command Line Interface, AWS SDK AWS CloudFormation, dan AWS Management Console. Ini berarti Anda dapat mengelola pengguna dengan alat yang sama dengan yang Anda buat dengan cluster.

  Untuk memulai, ikutiMenyiapkan entri akses, laluMigrasi entri yang ada untuk `aws-auth` `ConfigMap` mengakses entri.
- Menambahkan entri ke **`aws-auth` `ConfigMap`** — Jika versi platform cluster Anda lebih awal dari versi yang tercantum di bagian Prasyarat, maka Anda harus menggunakan opsi ini. Jika versi platform cluster Anda pada atau lebih lambat dari versi platform yang tercantum di bagian Prasyarat untuk Kubernetes versi klaster Anda, dan Anda telah menambahkan entri ke`ConfigMap`, maka sebaiknya Anda memigrasikan entri tersebut untuk mengakses entri. `ConfigMap`Namun, Anda tidak dapat memigrasikan entri yang ditambahkan Amazon EKS, seperti entri untuk peran IAM yang digunakan dengan grup node terkelola atau profil Fargate. Untuk informasi selengkapnya, lihat the section called “Berikan akses ke API Kubernetes”.

- Jika Anda harus menggunakan `aws-auth ConfigMap` opsi, Anda dapat menambahkan entri ke `ConfigMap` menggunakan **`eksctl create iamidentitymapping`** perintah. Untuk informasi selengkapnya, lihat Mengelola pengguna dan peran IAM dalam `eksctl` dokumentasi.

Atur Mode Otentikasi Cluster

Setiap cluster memiliki mode otentikasi. Mode otentikasi menentukan metode mana yang dapat Anda gunakan untuk memungkinkan prinsipal IAM mengakses Kubernetes objek di cluster Anda. Ada tiga mode otentikasi.

> **Important**
>
> Setelah metode entri akses diaktifkan, itu tidak dapat dinonaktifkan.
> Jika `ConfigMap` metode ini tidak diaktifkan selama pembuatan cluster, itu tidak dapat diaktifkan nanti. Semua cluster yang dibuat sebelum pengenalan entri akses memiliki `ConfigMap` metode yang diaktifkan.

Bagian `aws-auth ConfigMap` dalam cluster

Ini adalah mode otentikasi asli untuk cluster Amazon EKS. Prinsipal IAM yang menciptakan cluster adalah pengguna awal yang dapat mengakses cluster dengan menggunakan`kubectl`. Pengguna awal harus menambahkan pengguna lain ke daftar di `aws-auth ConfigMap` dan menetapkan izin yang memengaruhi pengguna lain dalam cluster. Pengguna lain ini tidak dapat mengelola atau menghapus pengguna awal, karena tidak ada entri di `ConfigMap` to manage.

Baik entri `ConfigMap` dan akses

Dengan mode otentikasi ini, Anda dapat menggunakan kedua metode untuk menambahkan prinsip IAM ke cluster. Perhatikan bahwa setiap metode menyimpan entri terpisah; misalnya, jika Anda menambahkan entri akses dari AWS CLI, `aws-auth ConfigMap` tidak diperbarui.

Akses entri saja

Dengan mode otentikasi ini, Anda dapat menggunakan EKS API, AWS Command Line Interface, AWS SDK AWS CloudFormation, dan AWS Management Console untuk mengelola akses ke cluster untuk prinsipal IAM.

Setiap entri akses memiliki tipe dan Anda dapat menggunakan kombinasi cakupan akses untuk membatasi prinsipal ke namespace tertentu dan kebijakan akses untuk menetapkan kebijakan

izin yang dapat digunakan kembali yang telah dikonfigurasi sebelumnya. Atau, Anda dapat menggunakan STANDARD jenis dan Kubernetes RBAC grup untuk menetapkan izin khusus.

Mode autentikasi	Metode
`ConfigMap` hanya (`CONFIG_MAP`)	`aws-auth` `ConfigMap`
EKS API dan `ConfigMap` (`API_AND_CONFIG_MAP`)	mengakses entri di EKS API, AWS Command Line Interface, AWS SDK AWS CloudFormation, dan dan AWS Management Console `aws-auth` `ConfigMap`
EKS API saja (`API`)	mengakses entri di EKS API, AWS Command Line Interface, AWS SDK, dan AWS CloudFormation AWS Management Console

Kelola entri akses

Prasyarat

- Keakraban dengan opsi akses klaster untuk klaster Amazon EKS Anda. Untuk informasi selengkapnya, lihat Berikan akses ke Kubernetes API .
- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS. Untuk menggunakan entri akses dan mengubah mode otentikasi cluster, cluster harus memiliki versi platform yang sama atau lebih lambat dari versi yang tercantum dalam tabel berikut, atau Kubernetes versi yang lebih lambat dari versi yang tercantum dalam tabel.

Versi Kubernetes	Versi platform
`1.28`	`eks.6`
`1.27`	`eks.10`
`1.26`	`eks.11`
`1.25`	`eks.12`

Versi Kubernetes	Versi platform
1.24	eks.15
1.23	eks.17

Anda dapat memeriksa versi Anda saat ini Kubernetes dan platform dengan mengganti *my-cluster* di perintah berikut dengan nama cluster Anda dan kemudian menjalankan perintah yang dimodifikasi:. **aws eks describe-cluster --name *my-cluster* --query 'cluster.{"Kubernetes Version": version, "Platform Version": platformVersion}'**

> **Important**
>
> Setelah Amazon EKS memperbarui klaster Anda ke versi platform yang tercantum dalam tabel, Amazon EKS membuat entri akses dengan izin administrator ke klaster untuk prinsipal IAM yang awalnya membuat cluster. Jika Anda tidak ingin prinsipal IAM tersebut memiliki izin administrator ke cluster, hapus entri akses yang dibuat Amazon EKS. Untuk cluster dengan versi platform yang lebih awal dari yang tercantum di tabel sebelumnya, pembuat klaster selalu merupakan administrator cluster. Tidak mungkin menghapus izin administrator cluster dari pengguna IAM atau peran yang membuat cluster.

- Prinsipal IAM dengan izin berikut untuk klaster Anda:`CreateAccessEntry`,,, `ListAccessEntries` `DescribeAccessEntry`, dan. `DeleteAccessEntry` `UpdateAccessEntry` Untuk informasi selengkapnya tentang izin Amazon EKS, lihat Tindakan yang ditentukan oleh Amazon Elastic Kubernetes Service di Referensi Otorisasi Layanan.
- Prinsipal IAM yang ada untuk membuat entri akses untuk, atau entri akses yang ada untuk memperbarui atau menghapus.

Menyiapkan entri akses

Untuk mulai menggunakan entri akses, Anda harus mengubah mode otentikasi cluster ke mode `API_AND_CONFIG_MAP` atau`API`. Ini menambahkan API untuk entri akses.

AWS Management Console

Untuk membuat entri akses

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih nama cluster tempat Anda ingin membuat entri akses.
3. Pilih tab Access.
4. Mode otentikasi menunjukkan mode otentikasi cluster saat ini. Jika mode mengatakanEKS API, Anda sudah dapat menambahkan entri akses dan Anda dapat melewati langkah-langkah yang tersisa.
5. Pilih Kelola akses.
6. Untuk modus otentikasi Cluster, pilih mode dengan. EKS API Perhatikan bahwa Anda tidak dapat mengubah mode autentikasi kembali ke mode yang menghapus entri EKS API dan akses.
7. Pilih Simpan perubahan. Amazon EKS mulai memperbarui cluster, status klaster berubah menjadiUpdating, dan perubahan dicatat di tab Riwayat Perbarui.
8. Tunggu status cluster kembali keActive. Saat clusterActive, Anda dapat mengikuti langkah-langkah Membuat entri akses untuk menambahkan akses ke cluster untuk prinsipal IAM.

AWS CLI

Prasyarat

Versi terbaru dari AWS CLI v1 diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. AWS CLI v2 tidak mendukung fitur baru selama beberapa hari. Anda dapat memeriksa versi saat ini dengan `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Package manager seperti `yum`apt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru dari file AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan `aws configure` di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal di AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.

1.

2. Jalankan perintah berikut. Ganti *my-cluster* dengan nama klaster Anda. Jika Anda ingin menonaktifkan `ConfigMap` metode secara permanen, ganti `API_AND_CONFIG_MAP` dengan`API`.

   Amazon EKS mulai memperbarui cluster, status klaster berubah menjadi`UPDATING`, dan perubahan dicatat dalamaws eks list-updates.

   ```
   aws eks update-cluster-config --name my-cluster --access-config
    authenticationMode=API_AND_CONFIG_MAP
   ```

3. Tunggu status cluster kembali ke`Active`. Saat clusterActive, Anda dapat mengikuti langkah-langkah Membuat entri akses untuk menambahkan akses ke cluster untuk prinsipal IAM.

Membuat entri akses

Pertimbangan

Sebelum membuat entri akses, pertimbangkan hal berikut:

- Entri akses mencakup Nama Sumber Daya Amazon (ARN) dari satu, dan hanya satu, prinsipal IAM yang ada. Prinsipal IAM tidak dapat dimasukkan dalam lebih dari satu entri akses. Pertimbangan tambahan untuk ARN yang Anda tentukan:
  - Praktik terbaik IAM merekomendasikan untuk mengakses klaster Anda menggunakan peran IAM yang memiliki kredensi jangka pendek, daripada pengguna IAM yang memiliki kredensi jangka panjang. Untuk informasi selengkapnya, lihat Mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara di Panduan Pengguna IAM.
  - Jika ARN adalah untuk peran IAM, itu dapat mencakup jalur. ARN dalam `aws-auth` `ConfigMap` entri, tidak dapat menyertakan jalur. Misalnya, ARN Anda bisa `arn:aws:iam::`*`111122223333`*`:role/`*`development/apps/my-role`* atau. `arn:aws:iam::`*`111122223333`*`:role/`*`my-role`*
  - Jika jenis entri akses adalah apa pun selain `STANDARD` (lihat pertimbangan selanjutnya tentang jenis), ARN harus sama dengan Akun AWS cluster Anda. Jika jenisnya`STANDARD`, ARN bisa sama, atau berbeda, Akun AWS dari akun tempat cluster Anda berada.
  - Anda tidak dapat mengubah prinsipal IAM setelah entri akses dibuat.
  - Jika Anda pernah menghapus prinsipal IAM dengan ARN ini, entri akses tidak dihapus secara otomatis. Kami menyarankan Anda menghapus entri akses dengan ARN untuk prinsipal IAM

yang Anda hapus. Jika Anda tidak menghapus entri akses dan membuat ulang prinsipal IAM, meskipun memiliki ARN yang sama, entri akses tidak akan berfungsi. Ini karena meskipun ARN sama untuk prinsipal IAM yang dibuat ulang, `roleID` atau `userID` (Anda dapat melihat ini dengan `aws sts get-caller-identity` AWS CLI perintah) berbeda untuk prinsip IAM yang dibuat ulang daripada untuk prinsipal IAM asli. Meskipun Anda tidak melihat kepala sekolah IAM `roleID` atau `userID` untuk entri akses, Amazon EKS menyimpannya dengan entri akses.

- Setiap entri akses memiliki tipe. Anda dapat menentukan `EC2 Linux` (untuk peran IAM yang digunakan dengan Linux atau Bottlerocket node yang dikelola sendiri), `EC2 Windows` (untuk peran IAM yang digunakan dengan node yang dikelola sendiri Windows), `FARGATE_LINUX` (untuk peran IAM yang digunakan dengan), atau sebagai tipe. AWS Fargate (Fargate)`STANDARD` Jika Anda tidak menentukan jenis, Amazon EKS secara otomatis menyetel jenisnya`STANDARD`. Tidak perlu membuat entri akses untuk peran IAM yang digunakan untuk grup node terkelola atau profil Fargate, karena Amazon EKS menambahkan entri untuk peran ini ke, terlepas dari versi platform mana cluster Anda `aws-auth` `ConfigMap` berada.

  Anda tidak dapat mengubah jenis setelah entri akses dibuat.

- Jika jenis entri aksesnya`STANDARD`, Anda dapat menentukan nama pengguna untuk entri akses. Jika Anda tidak menentukan nilai untuk nama pengguna, Amazon EKS menetapkan salah satu nilai berikut untuk Anda, tergantung pada jenis entri akses dan apakah prinsipal IAM yang Anda tentukan adalah peran IAM atau pengguna IAM. Kecuali Anda memiliki alasan khusus untuk menentukan nama pengguna Anda sendiri, sebaiknya jangan tentukan nama pengguna dan biarkan Amazon EKS membuatnya secara otomatis untuk Anda. Jika Anda menentukan nama pengguna Anda sendiri:
  - Itu tidak bisa dimulai dengan`system:`,`eks:`,`aws:`,`amazon:`, atau`iam:`.
  - Jika nama pengguna untuk peran IAM, kami sarankan Anda menambahkan `{{SessionName}}` ke akhir nama pengguna Anda. Jika Anda `{{SessionName}}` menambahkan nama pengguna Anda, nama pengguna harus menyertakan titik dua sebelum {{SessionName}}. Ketika peran ini diasumsikan, nama sesi yang ditentukan saat mengasumsikan peran secara otomatis diteruskan ke cluster dan akan muncul di CloudTrail log. Misalnya, Anda tidak dapat memiliki nama penggunajohn`{{SessionName}}`. Nama pengguna harus `:john{{SessionName}}` ataujo:hn`{{SessionName}}`. Usus besar hanya harus sebelumnya`{{SessionName}}`. Nama pengguna yang dihasilkan oleh Amazon EKS dalam tabel berikut mencakup ARN. Karena ARN termasuk titik dua, ia memenuhi persyaratan ini. Titik dua tidak diperlukan jika Anda tidak menyertakan `{{SessionName}}` nama pengguna Anda.

Tipe utama IAM	Tipe	Nilai nama pengguna yang ditetapkan secara otomatis oleh Amazon EKS
Pengguna	STANDARD	ARN pengguna. Contoh: arn:aws:iam:: *111122223333* :user/*my-user*
Peran	STANDARD	STS ARN dari peran ketika diasumsikan. Amazon EKS {{SessionName}} menambahkan peran tersebut. Contoh: arn:aws:sts:: *111122223333* :assumed-role/ *my-role*/{{SessionName}} Jika ARN peran yang Anda tentukan berisi jalur, Amazon EKS menghapusnya di nama pengguna yang dihasilkan.
Peran	EC2 Linux atau EC2 Windows	system:node:{{EC2PrivateDNSName}}
Peran	FARGATE_LINUX	system:node:{{SessionName}}

Anda dapat mengubah nama pengguna setelah entri akses dibuat.

- Jika jenis entri akses adalahSTANDARD, dan Anda ingin menggunakan otorisasi Kubernetes RBAC, Anda dapat menambahkan satu atau beberapa nama grup ke entri akses. Setelah Anda membuat entri akses, Anda dapat menambahkan dan menghapus nama grup. Agar prinsipal IAM memiliki akses ke Kubernetes objek di cluster Anda, Anda harus membuat dan mengelola objek otorisasi

Kubernetes berbasis peran (RBAC). Buat Kubernetes `RoleBinding` atau `ClusterRoleBinding` objek pada cluster Anda yang menentukan nama grup sebagai `subject` for`kind: Group`. Kubernetesmengotorisasi akses utama IAM ke objek cluster apa pun yang telah Anda tentukan dalam `ClusterRole` objek Kubernetes `Role` atau yang juga Anda tentukan dalam pengikatan Anda. `roleRef` Jika Anda menentukan nama grup, sebaiknya Anda terbiasa dengan objek otorisasi Kubernetes berbasis peran (RBAC). Untuk informasi selengkapnya, lihat Menggunakan Otorisasi RBAC dalam dokumentasi. Kubernetes

> **Important**
>
> Amazon EKS tidak mengonfirmasi bahwa objek Kubernetes RBAC apa pun yang ada di cluster Anda menyertakan nama grup apa pun yang Anda tentukan.

Alih-alih, atau sebagai tambahan, Kubernetes mengotorisasi akses utama IAM ke Kubernetes objek di klaster Anda, Anda dapat mengaitkan kebijakan akses Amazon EKS ke entri akses. Amazon EKS memberi otorisasi kepada prinsipal IAM untuk mengakses Kubernetes objek di klaster Anda dengan izin dalam kebijakan akses. Anda dapat membuat cakupan izin kebijakan akses ke Kubernetes ruang nama yang Anda tentukan. Penggunaan kebijakan akses tidak mengharuskan Anda mengelola objek Kubernetes RBAC. Untuk informasi selengkapnya, lihat Mengaitkan dan memisahkan kebijakan akses ke dan dari entri akses.

- Jika Anda membuat entri akses dengan tipe `EC2 Linux` atau`EC2 Windows`, prinsipal IAM yang membuat entri akses harus memiliki `iam:PassRole` izin. Untuk informasi selengkapnya, lihat Memberikan izin pengguna untuk meneruskan peran ke Layanan AWS dalam Panduan Pengguna IAM.
- Mirip dengan perilaku IAM standar, pembuatan dan pembaruan entri akses pada akhirnya konsisten, dan mungkin memerlukan beberapa detik untuk menjadi efektif setelah panggilan API awal berhasil dikembalikan. Anda harus merancang aplikasi Anda untuk memperhitungkan potensi penundaan ini. Kami menyarankan agar Anda tidak menyertakan entri akses yang dibuat atau diperbarui di jalur kode ketersediaan tinggi yang penting dari aplikasi Anda. Sebaliknya, buat perubahan dalam inisialisasi terpisah atau rutinitas pengaturan yang lebih jarang Anda lakukan. Selain itu, pastikan untuk memverifikasi bahwa perubahan telah dibuat merata sebelum alur kerja produksi bergantung padanya.
- Entri akses tidak mendukung peran terkait layanan. Anda tidak dapat membuat entri akses di mana ARN utama adalah peran terkait layanan. Anda dapat mengidentifikasi peran terkait layanan dengan ARN mereka, yang ada dalam format. `arn:aws:iam::*:role/aws-service-role/*`

Anda dapat membuat entri akses menggunakan AWS Management Console atau AWS CLI.

AWS Management Console

Untuk membuat entri akses

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih nama cluster tempat Anda ingin membuat entri akses.
3. Pilih tab Access.
4. Pilih Buat entri akses.
5. Untuk prinsipal IAM, pilih peran atau pengguna IAM yang ada. Praktik terbaik IAM merekomendasikan untuk mengakses klaster Anda menggunakan peran IAM yang memiliki kredensi jangka pendek, daripada pengguna IAM yang memiliki kredensi jangka panjang. Untuk informasi selengkapnya, lihat Mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara di Panduan Pengguna IAM.
6. Untuk Type, jika entri akses adalah untuk peran node yang digunakan untuk node Amazon EC2 yang dikelola sendiri, pilih EC2 Linux atau EC2 Windows. Jika tidak, terima default (Standar).
7. Jika Jenis yang Anda pilih adalah Standar dan Anda ingin menentukan Nama Pengguna, masukkan nama pengguna.
8. Jika Jenis yang Anda pilih adalah Standar dan Anda ingin menggunakan otorisasi Kubernetes RBAC untuk prinsipal IAM, tentukan satu atau beberapa nama untuk Grup. Jika Anda tidak menentukan nama grup apa pun dan ingin menggunakan otorisasi Amazon EKS, Anda dapat mengaitkan kebijakan akses di langkah selanjutnya, atau setelah entri akses dibuat.
9. (Opsional) Untuk Tag, tetapkan label ke entri akses. Misalnya, untuk membuatnya lebih mudah untuk menemukan semua sumber daya dengan tag yang sama.
10. Pilih Berikutnya.
11. Pada halaman Tambahkan kebijakan akses, jika jenis yang Anda pilih adalah Standar dan Anda ingin Amazon EKS mengotorisasi prinsipal IAM untuk memiliki izin ke Kubernetes objek di klaster Anda, selesaikan langkah-langkah berikut. Jika tidak, pilih Selanjutnya.

    a. Untuk nama Kebijakan, pilih kebijakan akses. Anda tidak dapat melihat izin kebijakan akses, tetapi mereka menyertakan izin serupa dengan yang ada di objek yang dihadapi

Kubernetes penggunaClusterRole. Untuk informasi selengkapnya, lihat Peran yang dihadapi pengguna dalam dokumentasi. Kubernetes

b. Pilih salah satu opsi berikut:

- Cluster — Pilih opsi ini jika Anda ingin Amazon EKS mengotorisasi prinsipal IAM agar memiliki izin dalam kebijakan akses untuk semua Kubernetes objek di klaster Anda.
- Kubernetesnamespace — Pilih opsi ini jika Anda ingin Amazon EKS mengotorisasi prinsipal IAM agar memiliki izin dalam kebijakan akses untuk semua Kubernetes objek di namespace tertentu Kubernetes di klaster Anda. Untuk Namespace, masukkan nama Kubernetes namespace di cluster Anda. Jika Anda ingin menambahkan ruang nama tambahan, pilih Tambahkan namespace baru dan masukkan nama namespace.

c. Jika Anda ingin menambahkan kebijakan tambahan, pilih Tambah kebijakan. Anda dapat membuat cakupan setiap kebijakan secara berbeda, tetapi Anda dapat menambahkan setiap kebijakan hanya sekali.

d. Pilih Berikutnya.

12. Tinjau konfigurasi untuk entri akses Anda. Jika ada yang terlihat salah, pilih Sebelumnya untuk kembali melalui langkah-langkah dan memperbaiki kesalahan. Jika konfigurasi sudah benar, pilih Buat.

AWS CLI

Prasyarat

Versi terbaru dari AWS CLI v1 diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. AWS CLI v2 tidak mendukung fitur baru selama beberapa hari. Anda dapat memeriksa versi saat ini dengan `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Package manager seperti `yumapt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru dari file AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan `aws configure` di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal di AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.

Untuk membuat entri akses

Anda dapat menggunakan salah satu contoh berikut untuk membuat entri akses:

- Buat entri akses untuk grup node Amazon EC2 Linux yang dikelola sendiri. Ganti *my-cluster* dengan nama cluster Anda, *111122223333* dengan Akun AWS ID Anda, dan *EKS- my-cluster-self-managed -ng-1* dengan nama peran IAM node Anda. Jika grup node Anda adalah grup node Windows, maka ganti *EC2_Linux* dengan. **EC2_Windows**

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/EKS-my-cluster-self-managed-ng-1 --type EC2_Linux
```

  Anda tidak dapat menggunakan `--kubernetes-groups` opsi ketika Anda menentukan jenis selain`STANDARD`. Anda tidak dapat mengaitkan kebijakan akses ke entri akses ini, karena jenisnya adalah nilai selain`STANDARD`.

- Buat entri akses yang memungkinkan peran IAM yang tidak digunakan untuk grup node yang dikelola sendiri Amazon EC2, yang Kubernetes ingin Anda otorisasi akses ke klaster. *Ganti my-cluster dengan nama cluster Anda, 111122223333 dengan Akun AWS ID Anda, dan peran saya dengan nama peran IAM Anda.* Ganti *Viewers* dengan nama grup yang telah Anda tentukan di `ClusterRoleBinding` objek Kubernetes `RoleBinding` atau di klaster Anda.

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role --type STANDARD --user Viewers --
kubernetes-groups Viewers
```

- Buat entri akses yang memungkinkan pengguna IAM untuk mengautentikasi ke cluster Anda. Contoh ini diberikan karena hal ini dimungkinkan, meskipun praktik terbaik IAM merekomendasikan untuk mengakses klaster Anda menggunakan peran IAM yang memiliki kredensi jangka pendek, daripada pengguna IAM yang memiliki kredensil jangka panjang. Untuk informasi selengkapnya, lihat Mewajibkan pengguna manusia untuk menggunakan federasi dengan penyedia identitas untuk mengakses AWS menggunakan kredensi sementara di Panduan Pengguna IAM.

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:user/my-user --type STANDARD --username my-user
```

  Jika Anda ingin pengguna ini memiliki lebih banyak akses ke klaster Anda daripada izin dalam peran penemuan Kubernetes API, Anda perlu mengaitkan kebijakan akses ke entri akses, karena `--kubernetes-groups` opsi tersebut tidak digunakan. Untuk informasi

selengkapnya, lihat Mengaitkan dan memisahkan kebijakan akses ke dan dari entri akses dan peran penemuan API dalam Kubernetes dokumentasi.

Memperbarui entri akses

Anda dapat memperbarui entri akses menggunakan AWS Management Console atau AWS CLI.

AWS Management Console

Untuk memperbarui entri akses

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih nama cluster tempat Anda ingin membuat entri akses.
3. Pilih tab Access.
4. Pilih entri akses yang ingin Anda perbarui.
5. Pilih Edit.
6. Untuk Username, Anda dapat mengubah nilai yang ada.
7. Untuk Grup, Anda dapat menghapus nama grup yang ada atau menambahkan nama grup baru. Jika nama grup berikut ada, jangan hapus mereka: system:nodes atau system:bootstrappers. Menghapus grup ini dapat menyebabkan cluster Anda berfungsi dengan tidak benar. Jika Anda tidak menentukan nama grup apa pun dan ingin menggunakan otorisasi Amazon EKS, kaitkan kebijakan akses di langkah selanjutnya.
8. Untuk Tag, Anda dapat menetapkan label ke entri akses. Misalnya, untuk membuatnya lebih mudah untuk menemukan semua sumber daya dengan tag yang sama. Anda juga dapat menghapus tag yang ada.
9. Pilih Simpan perubahan.
10. Jika Anda ingin mengaitkan kebijakan akses ke entri, lihatMengaitkan dan memisahkan kebijakan akses ke dan dari entri akses.

AWS CLI

Prasyarat

Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2`**

`| cut -d ' ' -f1`. Package manager seperti `yum` `apt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru dari file AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.

Untuk memperbarui entri akses

Ganti *`my-cluster`* dengan nama cluster Anda, *`111122223333`* dengan Akun AWS ID Anda, dan *`EKS- my-cluster-my-namespace -Viewers`* dengan nama peran IAM.

```
aws eks update-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers --kubernetes-
groups Viewers
```

Anda tidak dapat menggunakan `--kubernetes-groups` opsi jika jenis entri akses adalah nilai selain`STANDARD`. Anda juga tidak dapat mengaitkan kebijakan akses ke entri akses dengan jenis selain`STANDARD`.

Menghapus entri akses

Jika Anda menemukan bahwa Anda menghapus entri akses karena kesalahan, Anda selalu dapat membuatnya kembali. Jika entri akses yang Anda hapus dikaitkan dengan kebijakan akses apa pun, asosiasi akan dihapus secara otomatis. Anda tidak perlu memisahkan kebijakan akses dari entri akses sebelum menghapus entri akses.

Anda dapat menghapus entri akses menggunakan AWS Management Console atau AWS CLI.

AWS Management Console

Untuk menghapus entri akses

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih nama klaster tempat Anda ingin menghapus entri akses.
3. Pilih tab Access.
4. Dalam daftar entri Access, pilih entri akses yang ingin Anda hapus.
5. Pilih Hapus.

6. Di kotak dialog konfirmasi, pilih Hapus.

AWS CLI

Prasyarat

Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get`,`, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru dari file AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.

Untuk menghapus entri akses

Ganti *`my-cluster`* dengan nama cluster Anda, *`111122223333`* dengan Akun AWS ID Anda, dan peran saya dengan nama *`peran`* IAM yang tidak lagi ingin Anda akses ke cluster Anda.

```
aws eks delete-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role
```

Mengaitkan dan memisahkan kebijakan akses ke dan dari entri akses

Anda dapat menetapkan satu atau beberapa kebijakan akses untuk mengakses jenis entri. `STANDARD` Amazon EKS secara otomatis memberikan jenis entri akses lainnya izin yang diperlukan agar berfungsi dengan baik di klaster Anda. Kebijakan akses Amazon EKS mencakup Kubernetes izin, bukan izin IAM. Sebelum mengaitkan kebijakan akses ke entri akses, pastikan Anda sudah terbiasa dengan Kubernetes izin yang disertakan dalam setiap kebijakan akses. Untuk informasi selengkapnya, lihat Izin kebijakan akses. Jika tidak ada kebijakan akses yang memenuhi persyaratan Anda, maka jangan kaitkan kebijakan akses ke entri akses. Sebagai gantinya, tentukan satu atau beberapa nama grup untuk entri akses dan buat serta kelola Kubernetes objek kontrol akses berbasis peran. Untuk informasi selengkapnya, lihat Membuat entri akses.

Prasyarat

- Entri akses yang ada. Untuk membuatnya, lihat Membuat entri akses.

- AWS Identity and Access Management Peran atau pengguna dengan izin berikut:`ListAccessEntries`,,`DescribeAccessEntry`,`UpdateAccessEntry`, `ListAccessPoliciesAssociateAccessPolicy`, dan`DisassociateAccessPolicy`. Untuk informasi selengkapnya, lihat Tindakan yang ditentukan oleh Amazon Elastic Kubernetes Service di Referensi Otorisasi Layanan.

Sebelum mengaitkan kebijakan akses dengan entri akses, pertimbangkan persyaratan berikut:

- Anda dapat mengaitkan beberapa kebijakan akses ke setiap entri akses, tetapi Anda hanya dapat mengaitkan setiap kebijakan ke entri akses satu kali. Jika Anda mengaitkan beberapa kebijakan akses, prinsipal IAM entri akses memiliki semua izin yang disertakan dalam semua kebijakan akses terkait.
- Anda dapat membuat cakupan kebijakan akses ke semua sumber daya di klaster atau dengan menentukan nama satu atau beberapa ruang Kubernetes nama. Anda dapat menggunakan karakter wildcard untuk nama namespace. Misalnya, jika Anda ingin membuat cakupan kebijakan akses ke semua ruang nama yang dimulai`dev-`, Anda dapat menentukan `dev-*` sebagai nama namespace. Pastikan bahwa namespace ada di cluster Anda dan ejaan Anda cocok dengan nama namespace yang sebenarnya di cluster. Amazon EKS tidak mengonfirmasi ejaan atau keberadaan ruang nama di cluster Anda.
- Anda dapat mengubah cakupan akses untuk kebijakan akses setelah Anda mengaitkannya ke entri akses. Jika Anda telah mencakup kebijakan akses ke Kubernetes ruang nama, Anda dapat menambahkan dan menghapus ruang nama untuk asosiasi, jika diperlukan.
- Jika Anda mengaitkan kebijakan akses ke entri akses yang juga memiliki nama grup yang ditentukan, maka prinsipal IAM memiliki semua izin di semua kebijakan akses terkait. Ini juga memiliki semua izin dalam setiap Kubernetes `Role` atau `ClusterRole` objek yang ditentukan dalam setiap Kubernetes `Role` dan `RoleBinding` objek yang menentukan nama grup.
- Jika menjalankan `kubectl auth can-i --list` perintah, Anda tidak akan melihat Kubernetes izin apa pun yang ditetapkan oleh kebijakan akses yang terkait dengan entri akses untuk prinsipal IAM yang Anda gunakan saat menjalankan perintah. Perintah hanya menampilkan Kubernetes izin jika Anda telah memberikannya Kubernetes `Role` atau `ClusterRole` objek yang telah Anda ikat ke nama grup atau nama pengguna yang Anda tentukan untuk entri akses.
- Jika Anda menyamar sebagai Kubernetes pengguna atau grup saat berinteraksi dengan Kubernetes objek di cluster Anda, seperti menggunakan `kubectl` perintah dengan **`--as`** ***`username`*** atau**`--as-group`** ***`group-name`***, Anda memaksa penggunaan otorisasi RBAC. Kubernetes Akibatnya, prinsipal IAM tidak memiliki izin yang ditetapkan oleh kebijakan akses

apa pun yang terkait dengan entri akses. Satu-satunya Kubernetes izin yang dimiliki pengguna atau grup yang ditiru oleh kepala sekolah IAM adalah Kubernetes izin yang Anda berikan kepada mereka Kubernetes `Role` atau `ClusterRole` objek yang Anda ikat ke nama grup atau nama pengguna. Agar prinsipal IAM Anda memiliki izin dalam kebijakan akses terkait, jangan menyamar sebagai pengguna atau Kubernetes grup. Prinsipal IAM juga akan tetap memiliki izin apa pun yang Anda berikan kepada mereka di Kubernetes `Role` atau `ClusterRole` objek yang Anda ikat ke nama grup atau nama pengguna yang Anda tentukan untuk entri akses. Untuk informasi selengkapnya, lihat Peniruan identitas pengguna dalam dokumentasi. Kubernetes

Anda dapat mengaitkan kebijakan akses ke entri akses menggunakan AWS Management Console atau AWS CLI.

AWS Management Console

Untuk mengaitkan kebijakan akses ke entri akses menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih nama klaster yang memiliki entri akses yang ingin Anda kaitkan dengan kebijakan akses.
3. Pilih tab Access.
4. Jika jenis entri akses adalah Standar, Anda dapat mengaitkan atau memisahkan kebijakan akses Amazon EKS. Jika jenis entri akses Anda adalah apa pun selain Standar, maka opsi ini tidak tersedia.
5. Pilih Kebijakan akses asosiasi.
6. Untuk nama Kebijakan, pilih kebijakan dengan izin yang Anda inginkan untuk memiliki kepala sekolah IAM. Untuk melihat izin yang disertakan dalam setiap kebijakan, lihatIzin kebijakan akses.
7. Untuk cakupan Access, pilih cakupan akses. Jika Anda memilih Cluster, izin dalam kebijakan akses diberikan kepada prinsipal IAM untuk sumber daya di semua Kubernetes ruang nama. Jika Anda memilih Kubernetesnamespace, Anda kemudian dapat memilih Tambahkan namespace baru. Di bidang Namespace yang muncul, Anda dapat memasukkan nama Kubernetes namespace di cluster Anda. Jika Anda ingin prinsipal IAM memiliki izin di beberapa ruang nama, maka Anda dapat memasukkan beberapa ruang nama.
8. Pilih Tambahkan kebijakan akses.

AWS CLI

Prasyarat

Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru dari file AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.

Untuk mengaitkan kebijakan akses ke entri akses

1. Lihat kebijakan akses yang tersedia.

```
aws eks list-access-policies --output table
```

Contoh output adalah sebagai berikut.

```
-------------------------------------------------------------------------------------------
|                                        ListAccessPolicies
                      |
+-----------------------------------------------------------------------------------------
+
||                                          accessPolicies
                     ||
|+-------------------------------------------------------------------
+-------------------------------+|
||                                arn                                |
     name              ||
|+-------------------------------------------------------------------
+-------------------------------+|
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy       |
 AmazonEKSAdminPolicy         ||
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy |
 AmazonEKSClusterAdminPolicy  ||
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy        |
 AmazonEKSEditPolicy          ||
```

```
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy         |
 AmazonEKSViewPolicy          ||
|+-------------------------------------------------------------------
+------------------------------+|
```

Untuk melihat izin yang disertakan dalam setiap kebijakan, lihatIzin kebijakan akses.

2. Lihat entri akses yang ada. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks list-access-entries --cluster-name my-cluster
```

Contoh output adalah sebagai berikut.

```
{
    "accessEntries": [
        "arn:aws:iam::111122223333:role/my-role",
        "arn:aws:iam::111122223333:user/my-user"
    ]
}
```

3. Kaitkan kebijakan akses ke entri akses. Contoh berikut mengaitkan kebijakan AmazonEKSViewPolicy akses ke entri akses. *Setiap kali peran IAM peran saya mencoba mengakses Kubernetes objek di cluster, Amazon EKS akan mengotorisasi peran tersebut untuk menggunakan izin dalam kebijakan untuk mengakses Kubernetes objek di ruang nama my-namespace1 dan my-namespace2 saja.* Kubernetes Ganti *my-cluster* dengan nama cluster Anda, *111122223333* dengan Akun AWS ID Anda, dan peran saya dengan nama *peran* IAM yang Anda inginkan Amazon EKS untuk mengotorisasi akses ke objek cluster. Kubernetes

```
aws eks associate-access-policy --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role \
    --access-scope type=namespace,namespaces=my-namespace1,my-namespace2 --
policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy
```

Jika Anda ingin kepala sekolah IAM memiliki izin di seluruh klaster, ganti dengan. **type=namespace,namespaces=*my-namespace1*,*my-namespace2*** **type=cluster** Jika Anda ingin mengaitkan beberapa kebijakan akses ke entri akses, jalankan perintah beberapa kali, masing-masing dengan kebijakan akses unik. Setiap kebijakan akses terkait memiliki ruang lingkupnya sendiri.

> **Note**
>
> Jika nanti Anda ingin mengubah cakupan kebijakan akses terkait, jalankan perintah sebelumnya lagi dengan cakupan baru. Misalnya, jika Anda ingin menghapus *my-namespace2*, Anda akan menjalankan perintah lagi hanya dengan menggunakan. **type=namespace,namespaces=*my-namespace1*** Jika Anda ingin mengubah cakupan dari **namespace** ke**cluster**, Anda akan menjalankan perintah lagi menggunakan**type=cluster**, menghapus**type=namespace,namespaces=*my-namespace1*,*my-namespace2***.

Untuk memisahkan kebijakan akses dari entri akses

1. Tentukan kebijakan akses mana yang terkait dengan entri akses.

```
aws eks list-associated-access-policies --cluster-name my-cluster --principal-
arn arn:aws:iam::111122223333:role/my-role
```

Contoh output adalah sebagai berikut.

```
{
    "clusterName": "my-cluster",
    "principalArn": "arn:aws:iam::111122223333",
    "associatedAccessPolicies": [
        {
            "policyArn": "arn:aws:eks::aws:cluster-access-
policy/AmazonEKSViewPolicy",
            "accessScope": {
                "type": "cluster",
                "namespaces": []
            },
            "associatedAt": "2023-04-17T15:25:21.675000-04:00",
            "modifiedAt": "2023-04-17T15:25:21.675000-04:00"
        },
        {
            "policyArn": "arn:aws:eks::aws:cluster-access-
policy/AmazonEKSAdminPolicy",
            "accessScope": {
                "type": "namespace",
                "namespaces": [
```

```
                    "my-namespace1",
                    "my-namespace2"
                ]
            },
            "associatedAt": "2023-04-17T15:02:06.511000-04:00",
            "modifiedAt": "2023-04-17T15:02:06.511000-04:00"
        }
    ]
}
```

Pada contoh sebelumnya, prinsipal IAM untuk entri akses ini memiliki izin tampilan di semua ruang nama di cluster, dan izin administrator ke dua ruang nama. Kubernetes

2. Memutuskan kebijakan akses dari entri akses. Dalam contoh ini, `AmazonEKSAdminPolicy` kebijakan dipisahkan dari entri akses. Prinsipal IAM mempertahankan izin dalam kebijakan `AmazonEKSViewPolicy` akses untuk objek di ruang nama *my-namespace1 dan my-namespace2*, karena kebijakan akses tersebut tidak terlepas dari entri akses.

```
aws eks disassociate-access-policy --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role \
    --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy
```

Izin kebijakan akses

Kebijakan akses termasuk `rules` yang berisi Kubernetes `verbs` (izin) dan`resources`. Kebijakan akses tidak menyertakan izin atau sumber daya IAM. Mirip dengan Kubernetes `Role` dan `ClusterRole` objek, kebijakan akses hanya mencakup `allow`rules. Anda tidak dapat mengubah konten kebijakan akses. Anda tidak dapat membuat kebijakan akses sendiri. Jika izin dalam kebijakan akses tidak memenuhi kebutuhan Anda, buat objek Kubernetes RBAC dan tentukan nama grup untuk entri akses Anda. Untuk informasi selengkapnya, lihat Membuat entri akses. Izin yang terkandung dalam kebijakan akses mirip dengan izin dalam peran klaster yang dihadapi Kubernetes pengguna. Untuk informasi selengkapnya, lihat Peran yang dihadapi pengguna dalam dokumentasi. Kubernetes

Pilih kebijakan akses apa pun untuk melihat isinya. Setiap baris dari setiap tabel di setiap kebijakan akses adalah aturan terpisah.

AmazonEks AdminPolicy

Kebijakan akses ini mencakup izin yang memberikan izin utama IAM untuk sumber daya. Ketika dikaitkan dengan entri akses, cakupan aksesnya biasanya satu atau lebih Kubernetes ruang nama. Jika Anda ingin prinsipal IAM memiliki akses administrator ke semua sumber daya di klaster Anda, kaitkan kebijakan AmazonEks ClusterAdminPolicy akses ke entri akses Anda.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy`

KubernetesGrup API	Sumber daya Kubernetes	Kuberneteskata kerja (izin)
apps	daemonsets , deployments , deployments/rollback , deployments/scale , replicasets , replicasets/scale, statefulsets , statefulsets/scale	create, delete, deletecollection , patch, update
apps	controllerrevisions , daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefulsets , statefulsets/scale , statefulsets/status	get, list, watch
authorization.k8s.io	localsubjectaccessreviews	create
autoscaling	horizontalpodautoscalers	create, delete, deletecollection , patch, update
autoscaling	horizontalpodautoscalers , horizonta	get, list, watch

KubernetesGrup API	Sumber daya Kubernetes	Kuberneteskata kerja (izin)
	lpodautoscalers/status	
batch	cronjobs, jobs	create, delete, deletecollection , patch, update
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets , deployments , deployments/rollback , deployments/scale , ingresses , networkpolicies , replicasets , replicasets/scale , replicationcontrollers/scale	create, delete, deletecollection , patch, update
extensions	daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , ingresses , ingresses/status , networkpolicies , replicasets , replicasets/scale , replicasets/status , replicationcontrollers/scale	get, list, watch
networking.k8s.io	ingresses , ingresses/status , networkpolicies	get, list, watch

KubernetesGrup API	Sumber daya Kubernetes	Kuberneteskata kerja (izin)
networking.k8s.io	ingresses , networkpolicies	create, delete, deletecollection , patch, update
policy	poddisruptionbudgets	create, delete, deletecollection , patch, update
policy	poddisruptionbudgets , poddisruptionbudgets/status	get, list, watch
rbac.authorization.k8s.io	rolebindings , roles	create, delete, deletecollection , get, list, patch, update, watch
	configmaps , endpoints , persistentvolumeclaims , persistentvolumeclaims/status , pods, replicationcontrollers , replicationcontrollers/scale , serviceaccounts , services, services/status	get,list, watch
	pods/attach , pods/exec , pods/portforward , pods/proxy , secrets, services/proxy	get, list, watch

KubernetesGrup API	Sumber daya Kubernetes	Kuberneteskata kerja (izin)
	configmaps , events, persistentvolumeclaims , replicationcontrollers , replicationcontrollers/scale , secrets, serviceaccounts , services, services/proxy	create, delete, deletecollection , patch, update
	pods, pods/attach , pods/exec , pods/portforward , pods/proxy	create, delete, deletecollection , patch, update
	serviceaccounts	impersonate
	bindings, events, limitranges , namespaces/status , pods/log, pods/status , replicationcontrollers/status , resourcequotas , resourcequotas/status	get, list, watch
	namespaces	get,list, watch

AmazonEks ClusterAdminPolicy

Kebijakan akses ini mencakup izin yang memberikan akses administrator utama IAM ke klaster. Ketika dikaitkan dengan entri akses, cakupan aksesnya biasanya cluster, bukan Kubernetes namespace. Jika Anda ingin kepala sekolah IAM memiliki ruang lingkup administratif yang lebih terbatas, pertimbangkan untuk mengaitkan kebijakan AmazonEks AdminPolicy akses ke entri akses Anda.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy`

KubernetesGrup API	KubernetesNonResourceURL	Sumber daya Kubernetes	Kuberneteskata kerja (izin)
*		*	*
	*		*

AmazonEks EditPolicy

Kebijakan akses ini mencakup izin yang memungkinkan prinsipal IAM untuk mengedit sebagian besar Kubernetes sumber daya.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy`

KubernetesGrup API	Sumber daya Kubernetes	Kuberneteskata kerja (izin)
`apps`	`daemonsets` , `deployments` , `deployments/rollback` , `deployments/scale` , `replicasets` , `replicasets/scale`, `statefulsets` , `statefulsets/scale`	`create`, `delete`, `deletecollection` , `patch`, `update`
`apps`	`controllerrevisions` , `daemonsets` , `daemonsets/status` , `deployments` , `deployments/scale` , `deployments/status` , `replicasets` , `replicasets/scale` , `replicasets/status` , `statefulsets` , `statefulsets/scale` , `statefulsets/status`	`get`, `list`, `watch`

KubernetesGrup API	Sumber daya Kubernetes	Kuberneteskata kerja (izin)
autoscaling	horizontalpodautoscalers , horizontalpodautoscalers/status	get, list, watch
autoscaling	horizontalpodautoscalers	create, delete, deletecollection , patch, update
batch	cronjobs, jobs	create, delete, deletecollection , patch, update
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets , deployments , deployments/rollback , deployments/scale , ingresses , networkpolicies , replicasets , replicasets/scale , replicationcontrollers/scale	create, delete, deletecollection , patch, update

KubernetesGrup API	Sumber daya Kubernetes	Kuberneteskata kerja (izin)
extensions	daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , ingresses , ingresses/status , networkpolicies , replicasets , replicasets/scale , replicasets/status , replicationcontrollers/scale	get, list, watch
networking.k8s.io	ingresses , networkpolicies	create, delete, deletecollection , patch, update
networking.k8s.io	ingresses , ingresses/status , networkpolicies	get, list, watch
policy	poddisruptionbudgets	create, delete, deletecollection , patch, update
policy	poddisruptionbudgets , poddisruptionbudgets/status	get, list, watch
	namespaces	get, list, watch
	pods/attach , pods/exec , pods/portforward , pods/proxy , secrets, services/proxy	get, list, watch
	serviceaccounts	impersonate

KubernetesGrup API	Sumber daya Kubernetes	Kuberneteskata kerja (izin)
	pods, pods/attach , pods/exec , pods/port forward , pods/proxy	create, delete, deletecollection , patch, update
	configmaps , events, persistentvolumeclaims , replicationcontrollers , replicationcontrollers/scale , secrets, serviceaccounts , services, services/proxy	create, delete, deletecollection , patch, update
	configmaps , endpoints , persistentvolumeclaims , persistentvolumeclaims/status , pods, replicationcontrollers , replicationcontrollers/scale , serviceaccounts , services, services/status	get, list, watch
	bindings, events, limitranges , namespaces/status , pods/log, pods/status , replicationcontrollers/status , resourcequotas , resourcequotas/status	get, list, watch

AmazonEks ViewPolicy

Kebijakan akses ini mencakup izin yang memungkinkan prinsipal IAM untuk melihat sebagian besar Kubernetes sumber daya.

ARN – `arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy`

KubernetesGrup API	Sumber daya Kubernetes	Kubernetes kata kerja (izin)
apps	controllerrevisions , daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefulsets , statefulsets/scale , statefulsets/status	get, list, watch
autoscaling	horizontalpodautoscalers , horizontalpodautoscalers/status	get, list, watch
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets ,daemonsets/status ,deployments ,deployments/scale,deployments/status ,ingresses ,,ingresses/status,networkpo	get, list, watch

KubernetesGrup API	Sumber daya Kubernetes	Kuberneteskata kerja (izin)
	licies ,replicase ts , replicasets/ scale,replicasets/ status , replicati oncontrollers/scale	
networking.k8s.io	ingresses , ingresses /status , networkpo licies	get, list, watch
policy	poddisruptionbudgets , poddisruptionbudge ts/status	get, list, watch
	configmaps , endpoints , persistentvolumecl aims , persisten tvolumeclaims/stat us , pods, replicati oncontrollers , replicationcontrol lers/scale , serviceac counts , services, services/status	get, list, watch
	bindings,events,limitrang es ,namespaces/ status,pods/log,pods/ status , replicatio ncontrollers/statu s , rresourcequotas , r esourcequotas/status	get, list, watch
	namespaces	get, list, watch

Akses pembaruan kebijakan

Lihat detail tentang pembaruan untuk mengakses kebijakan, sejak diperkenalkan. Untuk peringatan otomatis tentang perubahan pada halaman ini, berlangganan umpan RSS di halaman riwayat Dokumen Amazon EKS.

Perubahan	Deskripsi	Tanggal
Kebijakan akses diperkenalkan.	Amazon EKS memperkenalkan kebijakan akses.	29 Mei 2023

Migrasi entri yang ada untuk **`aws-auth ConfigMap`** mengakses entri

Jika Anda telah menambahkan entri ke klaster Anda, kami sarankan Anda membuat entri akses untuk entri yang ada di Anda. `aws-auth ConfigMap` `aws-auth ConfigMap` Setelah membuat entri akses, Anda dapat menghapus entri dari entri Anda. `ConfigMap` Anda tidak dapat mengaitkan kebijakan akses ke entri di. `aws-auth ConfigMap` Jika Anda ingin mengaitkan kebijakan akses ke prinsipal IAM Anda, buat entri akses.

> **Important**
>
> Jangan hapus `aws-auth ConfigMap` entri yang ada yang dibuat oleh Amazon EKS saat Anda menambahkan grup node terkelola atau profil Fargate ke klaster Anda. Jika Anda menghapus entri yang dibuat Amazon EKS`ConfigMap`, klaster Anda tidak akan berfungsi dengan baik. Namun, Anda dapat menghapus entri apa pun untuk grup node yang dikelola sendiri setelah Anda membuat entri akses untuk mereka.

Prasyarat

- Keakraban dengan entri akses dan kebijakan akses. Lihat informasi yang lebih lengkap di Kelola entri akses dan Mengaitkan dan memisahkan kebijakan akses ke dan dari entri akses.
- Cluster yang ada dengan versi platform yang pada atau lebih lambat dari versi yang tercantum dalam Prasyarat Memungkinkan peran IAM atau pengguna mengakses objek Kubernetes pada topik klaster Amazon EKS Anda.

- Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbaruieksctl, lihat Instalasi dalam `eksctl` dokumentasi.
- Kubernetesizin untuk memodifikasi `aws-auth` `ConfigMap` di `kube-system` namespace.
- AWS Identity and Access Management Peran atau pengguna dengan izin berikut: `CreateAccessEntry` dan`ListAccessEntries`. Untuk informasi selengkapnya, lihat Tindakan yang ditentukan oleh Amazon Elastic Kubernetes Service di Referensi Otorisasi Layanan.

Untuk memigrasikan entri dari Anda **`aws-auth ConfigMap`** ke entri akses

1. Lihat entri yang ada di Andaaws-auth `ConfigMap`. Ganti *`my-cluster`* dengan nama klaster Anda.

   ```
   eksctl get iamidentitymapping --cluster my-cluster
   ```

   Contoh output adalah sebagai berikut.

   ```
   ARN
                USERNAME                                GROUPS
                            ACCOUNT
   arn:aws:iam::111122223333:role/EKS-my-cluster-Admins
                Admins                                  system:masters
   arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers
                my-namespace-Viewers                    Viewers
   arn:aws:iam::111122223333:role/EKS-my-cluster-self-managed-ng-1
                               system:node:{{EC2PrivateDNSName}}
    system:bootstrappers,system:nodes
   arn:aws:iam::111122223333:user/my-user
                my-user
   arn:aws:iam::111122223333:role/EKS-my-cluster-fargateprofile1
                               system:node:{{SessionName}}
    system:bootstrappers,system:nodes,system:node-proxier
   arn:aws:iam::111122223333:role/EKS-my-cluster-managed-ng
                               system:node:{{EC2PrivateDNSName}}
    system:bootstrappers,system:nodes
   ```

2. Buat entri akses untuk setiap `ConfigMap` entri yang Anda buat kembali di output sebelumnya. Saat membuat entri akses, pastikan untuk menentukan nilai yang sama untukARN,, `USERNAME`GROUPS, dan `ACCOUNT` dikembalikan dalam output Anda. Dalam contoh keluaran,

Anda akan membuat entri akses untuk semua entri kecuali dua entri terakhir, karena entri tersebut dibuat oleh Amazon EKS untuk profil Fargate dan grup node terkelola.

3. Hapus entri dari entri akses apa pun yang Anda `ConfigMap` buat. Jika Anda tidak menghapus entri dari`ConfigMap`, pengaturan untuk entri akses untuk ARN utama IAM akan menggantikan `ConfigMap` entri. Ganti *111122223333* dengan Akun AWS ID Anda dan *EKS- my-cluster-my-namespace -Viewers* dengan nama peran dalam entri di Anda. `ConfigMap` Jika entri yang Anda hapus adalah untuk pengguna IAM, bukan peran IAM, ganti **`role`** dengan **`user`** dan *EKS- my-cluster-my-namespace -Viewers dengan nama* pengguna.

```
eksctl delete iamidentitymapping --arn arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers --cluster my-cluster
```

Mengaktifkan akses utama IAM ke klaster Anda

> **Important**
>
> `aws-auth`ConfigMapItu sudah usang. Metode yang direkomendasikan untuk mengelola akses ke Kubernetes API adalah Entri Akses.

Akses ke klaster Anda menggunakan prinsip IAM diaktifkan oleh AWS IAM Authenticator for, Kubernetes yang berjalan di bidang kontrol Amazon EKS. Authenticator mendapatkan informasi konfigurasinya dari file. `aws-auth` `ConfigMap` Untuk semua `aws-auth` `ConfigMap` pengaturan, lihat Format Konfigurasi Lengkap aktifGitHub.

Tambahkan prinsipal IAM ke kluster Amazon EKS Anda

Saat Anda membuat klaster Amazon EKS, prinsipal IAM yang membuat klaster secara otomatis diberikan `system:masters` izin dalam konfigurasi kontrol akses berbasis peran (RBAC) klaster di bidang kontrol Amazon EKS. Prinsipal ini tidak muncul dalam konfigurasi yang terlihat, jadi pastikan untuk melacak prinsipal mana yang awalnya membuat cluster. Untuk memberikan kepala sekolah IAM tambahan kemampuan untuk berinteraksi dengan cluster Anda, edit bagian `aws-auth` `ConfigMap` dalam Kubernetes dan buat Kubernetes `rolebinding` atau `clusterrolebinding` dengan nama `group` yang Anda tentukan di. `aws-auth` `ConfigMap`

> **Note**
>
> Untuk informasi selengkapnya tentang konfigurasi kontrol akses Kubernetes berbasis peran (RBAC), lihat Menggunakan Otorisasi RBAC dalam dokumentasi. Kubernetes

Untuk menambahkan prinsipal IAM ke cluster Amazon EKS

1. Tentukan kredensil mana yang `kubectl` digunakan untuk mengakses klaster Anda. Di komputer Anda, Anda dapat melihat kredenal mana yang `kubectl` digunakan dengan perintah berikut. Ganti ***~/.kube/config*** dengan jalur ke `kubeconfig` file Anda jika Anda tidak menggunakan jalur default.

   ```
   cat ~/.kube/config
   ```

   Contoh output adalah sebagai berikut.

   ```
   [...]
   contexts:
   - context:
       cluster: my-cluster.region-code.eksctl.io
       user: admin@my-cluster.region-code.eksctl.io
     name: admin@my-cluster.region-code.eksctl.io
   current-context: admin@my-cluster.region-code.eksctl.io
   [...]
   ```

   Dalam contoh keluaran sebelumnya, kredensial untuk pengguna bernama *`admin`* dikonfigurasi untuk klaster bernama. *`my-cluster`* Jika ini adalah pengguna yang membuat cluster, maka ia sudah memiliki akses ke cluster Anda. Jika bukan pengguna yang membuat cluster, maka Anda perlu menyelesaikan langkah-langkah yang tersisa untuk mengaktifkan akses cluster untuk prinsipal IAM lainnya. Praktik terbaik IAM menyarankan agar Anda memberikan izin untuk peran, bukan pengguna. Anda dapat melihat prinsipal lain mana yang saat ini memiliki akses ke cluster Anda dengan perintah berikut:

   ```
   kubectl describe -n kube-system configmap/aws-auth
   ```

   Contoh output adalah sebagai berikut.

```
Name:         aws-auth
Namespace:    kube-system
Labels:       <none>
Annotations:  <none>

Data
====
mapRoles:
----
- groups:
  - system:bootstrappers
  - system:nodes
  rolearn: arn:aws:iam::111122223333:role/my-node-role
  username: system:node:{{EC2PrivateDNSName}}


BinaryData
====

Events:  <none>
```

Contoh sebelumnya adalah default `aws-auth` `ConfigMap`. Hanya peran instance node yang memiliki akses ke cluster.

2. Pastikan bahwa Anda sudah ada Kubernetes `roles` dan `rolebindings` atau `clusterroles` dan `clusterrolebindings` bahwa Anda dapat memetakan prinsipal IAM ke. Untuk informasi selengkapnya tentang sumber daya ini, lihat Menggunakan Otorisasi RBAC dalam dokumentasi. Kubernetes

   1. Lihat yang ada Kubernetes `roles` atau`clusterroles`. `Roles`dicakup ke a`namespace`, tetapi `clusterroles` dicakup ke cluster.

      ```
      kubectl get roles -A
      ```

      ```
      kubectl get clusterroles
      ```

   2. Lihat detail apa pun `role` atau yang `clusterrole` dikembalikan di keluaran sebelumnya dan konfirmasikan bahwa ia memiliki izin (`rules`) yang Anda inginkan untuk dimiliki oleh prinsipal IAM Anda di cluster Anda.

Ganti *role-name* dengan `role` nama yang dikembalikan dalam output dari perintah sebelumnya. Ganti *kube-system* dengan namespace dari file. `role`

```
kubectl describe role role-name -n kube-system
```

Ganti *cluster-role-name* dengan `clusterrole` nama yang dikembalikan dalam output dari perintah sebelumnya.

```
kubectl describe clusterrole cluster-role-name
```

3. Lihat yang ada Kubernetes `rolebindings` atau`clusterrolebindings`. `Rolebindings`dicakup ke a`namespace`, tetapi `clusterrolebindings` dicakup ke cluster.

```
kubectl get rolebindings -A
```

```
kubectl get clusterrolebindings
```

4. Lihat detail apa pun `rolebinding` atau `clusterrolebinding` dan konfirmasikan bahwa ia memiliki `role` atau `clusterrole` dari langkah sebelumnya yang terdaftar sebagai `roleRef` dan nama grup yang terdaftar untuk`subjects`.

   Ganti *role-binding-name* dengan `rolebinding` nama yang dikembalikan dalam output dari perintah sebelumnya. Ganti *kube-system* dengan `namespace` dari`rolebinding`.

```
kubectl describe rolebinding role-binding-name -n kube-system
```

   Contoh output adalah sebagai berikut.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: eks-console-dashboard-restricted-access-role-binding
  namespace: default
subjects:
- kind: Group
  name: eks-console-dashboard-restricted-access-group
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
```

```
  name: eks-console-dashboard-restricted-access-role
  apiGroup: rbac.authorization.k8s.io
```

Ganti *cluster-role-binding-name* dengan `clusterrolebinding` nama yang dikembalikan dalam output dari perintah sebelumnya.

```
kubectl describe clusterrolebinding cluster-role-binding-name
```

Contoh output adalah sebagai berikut.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks-console-dashboard-full-access-binding
subjects:
- kind: Group
  name: eks-console-dashboard-full-access-group
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: eks-console-dashboard-full-access-clusterrole
  apiGroup: rbac.authorization.k8s.io
```

3. Edit `aws-authConfigMap`. Anda dapat menggunakan alat seperti `eksctl` untuk memperbarui `ConfigMap` atau Anda dapat memperbaruinya secara manual dengan mengeditnya.

> **Important**
>
> Kami merekomendasikan menggunakan`eksctl`, atau alat lain, untuk mengedit`ConfigMap`. Untuk informasi tentang alat lain yang dapat Anda gunakan, lihat Menggunakan alat untuk membuat perubahan `aws-authConfigMap` pada panduan praktik terbaik Amazon EKS. Format yang tidak benar `aws-auth ConfigMap` dapat menyebabkan Anda kehilangan akses ke cluster Anda.

eksctl

Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

1. Lihat pemetaan saat ini di. `ConfigMap` Ganti *`my-cluster`* dengan nama klaster Anda. Ganti *`region-code`* dengan tempat Wilayah AWS cluster Anda berada.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

   Contoh output adalah sebagai berikut.

```
ARN
                    USERNAME                                GROUPS
             ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

2. Tambahkan pemetaan untuk peran. Ganti *`my-role`* dengan nama peran Anda. Ganti *`eks-console-dashboard-full-access-group`* dengan nama grup yang ditentukan dalam `ClusterRoleBinding` objek Kubernetes `RoleBinding` atau Anda. Ganti *`111122223333`* dengan ID akun Anda. Anda dapat mengganti *`admin`* dengan nama apa pun yang Anda pilih.

```
eksctl create iamidentitymapping --cluster my-cluster --region=region-code \
    --arn arn:aws:iam::111122223333:role/my-role --username admin --group eks-
console-dashboard-full-access-group \
    --no-duplicate-arns
```

   > **Important**
   >
   > Peran ARN tidak dapat menyertakan jalur seperti. `role/my-team/developers/my-role` Format ARN harus. `arn:aws:iam::111122223333:role/my-role` Dalam contoh ini, `my-team/developers/` perlu dihapus.

   Contoh output adalah sebagai berikut.

```
[...]
2022-05-09 14:51:20 [#]  adding identity "arn:aws:iam::111122223333:role/my-
role" to auth ConfigMap
```

3. Tambahkan pemetaan untuk pengguna. Praktik terbaik IAM menyarankan agar Anda memberikan izin untuk peran, bukan pengguna. Ganti *my-user* dengan nama pengguna Anda. Ganti *eks-console-dashboard-restricted-access-group* dengan nama grup yang ditentukan dalam `ClusterRoleBinding` objek Kubernetes `RoleBinding` atau Anda. Ganti *111122223333* dengan ID akun Anda. Anda dapat mengganti *pengguna saya* dengan nama apa pun yang Anda pilih.

```
eksctl create iamidentitymapping --cluster my-cluster --region=region-code \
    --arn arn:aws:iam::111122223333:user/my-user --username my-user --
group eks-console-dashboard-restricted-access-group \
    --no-duplicate-arns
```

Contoh output adalah sebagai berikut.

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws:iam::111122223333:user/my-
user" to auth ConfigMap
```

4. Lihat pemetaan di lagi. `ConfigMap`

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

Contoh output adalah sebagai berikut.

```
ARN
                    USERNAME                                GROUPS
                      ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
arn:aws:iam::111122223333:role/admin
                        my-role                                 eks-console-
dashboard-full-access-group
```

```
arn:aws:iam::111122223333:user/my-user
                        my-user                                    eks-console-
dashboard-restricted-access-group
```

Edit ConfigMap manually

1. Buka `ConfigMap` untuk mengedit.

```
kubectl edit -n kube-system configmap/aws-auth
```

> **Note**
>
> Jika Anda menerima kesalahan yang menyatakan "`Error from server (NotFound): configmaps "aws-auth" not found`“, maka gunakan prosedur Terapkan `aws-auth`ConfigMap ke cluster Anda untuk menerapkan stok`ConfigMap`.

2. Tambahkan prinsip IAM Anda ke. `ConfigMap` Grup IAM bukanlah prinsipal IAM, sehingga tidak dapat ditambahkan ke. `ConfigMap`
   - Untuk menambahkan peran IAM (misalnya, untuk pengguna federasi): Tambahkan detail peran ke `mapRoles` bagian`ConfigMap`, di bawah. `data` Tambahkan bagian ini jika belum ada di dalam file. Setiap masuk mendukung parameter berikut:
     - rolearn: ARN IAM role untuk menambahkan. Nilai ini tidak dapat menyertakan jalur. Misalnya, Anda tidak dapat menentukan ARN seperti. `arn:aws:iam::111122223333:role/my-team/developers/role-name` ARN harus sebagai gantinya. `arn:aws:iam::111122223333:role/role-name`
     - username: Nama pengguna dalam Kubernetes untuk memetakan ke peran IAM.
     - grup: Grup atau daftar Kubernetes grup untuk memetakan peran. Grup dapat berupa grup default, atau grup yang ditentukan dalam `clusterrolebinding` atau`rolebinding`. Untuk informasi selengkapnya, lihat Peran default dan binding peran dalam dokumentasi. Kubernetes
   - Untuk menambahkan pengguna IAM: Praktik terbaik IAM menyarankan Anda memberikan izin ke peran, bukan pengguna. Tambahkan detail pengguna ke `mapUsers` bagian`ConfigMap`, di bawah`data`. Tambahkan bagian ini jika belum ada di dalam file. Setiap entri mendukung parameter berikut:

- userarn: ARN pengguna IAM untuk ditambahkan.
- username: Nama pengguna dalam Kubernetes untuk memetakan ke pengguna IAM.
- grup: Grup, atau daftar Kubernetes grup untuk memetakan pengguna. Grup dapat berupa grup default, atau grup yang ditentukan dalam `clusterrolebinding` atau`rolebinding`. Untuk informasi selengkapnya, lihat Peran default dan binding peran dalam dokumentasi. Kubernetes

Misalnya, blok YAMAL berikut berisi:

- `mapRoles`Bagian yang memetakan instance node IAM ke Kubernetes grup sehingga node dapat mendaftarkan diri dengan cluster dan peran `my-console-viewer-role` IAM yang dipetakan ke Kubernetes grup yang dapat melihat semua Kubernetes sumber daya untuk semua cluster. Untuk daftar izin IAM dan Kubernetes grup yang diperlukan untuk peran `my-console-viewer-role` IAM, lihat. Izin yang diperlukan
- `mapUsers`Bagian yang memetakan pengguna `admin` IAM dari AWS akun default ke `system:masters` Kubernetes grup dan `my-user` pengguna dari AWS akun lain yang dipetakan ke Kubernetes grup yang dapat melihat Kubernetes sumber daya untuk namespace tertentu. Untuk daftar izin IAM dan Kubernetes grup yang diperlukan untuk pengguna `my-user` IAM, lihat. Izin yang diperlukan

Tambahkan atau hapus baris seperlunya dan ganti semua *`example values`*dengan nilai Anda sendiri.

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this
 file will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
      rolearn: arn:aws:iam::111122223333:role/my-role
      username: system:node:{{EC2PrivateDNSName}}
    - groups:
      - eks-console-dashboard-full-access-group
      rolearn: arn:aws:iam::111122223333:role/my-console-viewer-role
      username: my-console-viewer-role
```

```
  mapUsers: |
    - groups:
      - system:masters
      userarn: arn:aws:iam::111122223333:user/admin
      username: admin
    - groups:
      - eks-console-dashboard-restricted-access-group
      userarn: arn:aws:iam::444455556666:user/my-user
      username: my-user
```

3. Simpan file dan keluar dari editor teks Anda.

Terapkan **aws-authConfigMap** ke cluster Anda

Secara otomatis `aws-auth` `ConfigMap` dibuat dan diterapkan ke cluster Anda ketika Anda membuat grup node terkelola atau ketika Anda membuat grup node menggunakan`eksctl`. Ini awalnya dibuat untuk memungkinkan node bergabung dengan cluster Anda, tetapi Anda juga menggunakan ini `ConfigMap` untuk menambahkan akses kontrol akses berbasis peran (RBAC) ke prinsip-prinsip IAM. Jika Anda telah meluncurkan node yang dikelola sendiri dan belum menerapkannya `aws-auth` `ConfigMap` ke cluster Anda, Anda dapat melakukannya dengan prosedur berikut.

Untuk menerapkan **aws-authConfigMap** ke cluster Anda

1. Periksa untuk melihat apakah Anda sudah menerapkan `aws-authConfigMap`.

   ```
   kubectl describe configmap -n kube-system aws-auth
   ```

   Jika Anda menerima kesalahan yang menyatakan "`Error from server (NotFound): configmaps "aws-auth" not found`“, maka lanjutkan dengan langkah-langkah berikut untuk menerapkan stok`ConfigMap`.
2. Unduh, edit, dan terapkan peta konfigurasi AWS autentikator.

   a. Unduh peta konfigurasi.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/aws-auth-cm.yaml
   ```

   b. Dalam `aws-auth-cm.yaml` file, atur `rolearn` ke Amazon Resource Name (ARN) dari peran IAM yang terkait dengan node Anda. Anda dapat melakukan ini dengan editor teks, atau dengan mengganti *`my-node-instance-role`* dan menjalankan perintah berikut:

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-role|' aws-auth-cm.yaml
```

Jangan memodifikasi baris lain dalam file ini.

> **Important**
>
> Peran ARN tidak dapat menyertakan jalur seperti. `role/my-team/developers/my-role` Format ARN harus. `arn:aws:iam::111122223333:role/my-role` Dalam contoh ini, `my-team/developers/` perlu dihapus.

Anda dapat memeriksa output AWS CloudFormation tumpukan untuk grup node Anda dan mencari nilai berikut:

- InstanceRoleARN — Untuk grup simpul yang dibuat dengan `eksctl`
- NodeInstanceRole— Untuk grup simpul yang dibuat dengan AWS CloudFormation templat penjual Amazon EKS di AWS Management Console

c. Terapkan konfigurasi. Perintah ini mungkin memerlukan waktu beberapa menit untuk diselesaikan.

```
kubectl apply -f aws-auth-cm.yaml
```

> **Note**
>
> Jika Anda menerima kesalahan otorisasi atau jenis sumber daya, lihat Tidak sah atau akses ditolak (`kubectl`) di topik pemecahan masalah.

3. Perhatikan status simpul Anda dan tunggu sampai simpul mencapai Status `Ready`.

```
kubectl get nodes --watch
```

Masukkan `Ctrl+C` untuk kembali ke prompt shell.

Mengautentikasi pengguna untuk klaster Anda dari penyedia OpenID Connect identitas

Amazon EKS mendukung penggunaan OpenID Connect (OIDC) penyedia identitas sebagai metode untuk mengautentikasi pengguna ke klaster Anda. OIDCpenyedia identitas dapat digunakan dengan, atau sebagai alternatif untuk AWS Identity and Access Management (IAM). Untuk informasi selengkapnya tentang menggunakan IAM, lihat the section called “Berikan akses ke API Kubernetes”. Setelah mengonfigurasi otentikasi ke klaster, Anda dapat membuat Kubernetes `roles` dan `clusterroles` menetapkan izin ke peran, lalu mengikat peran ke identitas menggunakan dan. Kubernetes `rolebindings` `clusterrolebindings` Untuk informasi selengkapnya, lihat Menggunakan Otorisasi RBAC dalam dokumentasi. Kubernetes

Pertimbangan

- Anda dapat mengaitkan satu penyedia OIDC identitas ke klaster Anda.
- Kubernetestidak menyediakan penyedia OIDC identitas. Anda dapat menggunakan penyedia OIDC identitas publik yang ada, atau Anda dapat menjalankan penyedia identitas Anda sendiri. Untuk daftar penyedia tersertifikasi, lihat OpenID Certification di situs OpenID.
- URL penerbit penyedia OIDC identitas harus dapat diakses publik, sehingga Amazon EKS dapat menemukan kunci penandatanganan. Amazon EKS tidak mendukung penyedia OIDC identitas dengan sertifikat yang ditandatangani sendiri.
- Anda tidak dapat menonaktifkan autentikasi IAM ke klaster Anda, karena masih diperlukan untuk menggabungkan node ke cluster.
- Cluster Amazon EKS masih harus dibuat oleh prinsipal AWS IAM, bukan pengguna penyedia OIDC identitas. Ini karena pembuat cluster berinteraksi dengan Amazon EKS API, bukan Kubernetes API.
- OIDCpengguna yang diautentikasi oleh penyedia identitas tercantum dalam log audit klaster jika CloudWatch log dihidupkan untuk bidang kontrol. Untuk informasi selengkapnya, lihat Mengaktifkan dan menonaktifkan log bidang kendali.
- Anda tidak dapat masuk ke akun AWS Management Console dengan akun dari OIDC penyedia. Anda hanya dapat melihat Kubernetes sumber daya di konsol dengan masuk ke AWS Identity and Access Management akun AWS Management Console dengan.

Kaitkan penyedia OIDC identitas

Sebelum Anda dapat mengaitkan penyedia OIDC identitas dengan klaster Anda, Anda memerlukan informasi berikut dari penyedia Anda:

URL Penerbit

URL penyedia identitas OIDC yang memungkinkan server API menemukan kunci penandatanganan publik untuk memverifikasi token. URL harus dimulai dengan `https://` dan harus sesuai dengan klaim `iss` dalam token ID OIDC penyedia. Sesuai dengan standar OIDC, komponen jalur diperbolehkan tetapi parameter kueri tidak. Biasanya URL hanya terdiri dari nama host, seperti `https://server.example.org` atau `https://example.com`. URL ini harus mengarah ke level di bawah `.well-known/openid-configuration` dan harus dapat diakses oleh publik melalui internet.

ID Klien (juga dikenal sebagai audiens)

ID untuk aplikasi klien yang membuat permintaan otentikasi ke penyedia identitas OIDC.

Anda dapat mengaitkan penyedia identitas menggunakan `eksctl` atau AWS Management Console.

eksctl

Untuk mengaitkan penyedia OIDC identitas ke klaster Anda menggunakan **`eksctl`**

1. Buat file bernama *`associate-identity-provider.yaml`* dengan isi berikut ini. Ganti *`example values`* dengan milik Anda sendiri. Nilai-nilai di `identityProviders` bagian ini diperoleh dari penyedia OIDC identitas Anda. Nilai hanya diperlukan untuk `name`, `type`, `issuerUrl`, dan pengaturan `clientId` di bawah `identityProviders`.

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

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

identityProviders:
  - name: my-provider
    type: oidc
```

```
    issuerUrl: https://example.com
    clientId: kubernetes
    usernameClaim: email
    usernamePrefix: my-username-prefix
    groupsClaim: my-claim
    groupsPrefix: my-groups-prefix
    requiredClaims:
      string: string
    tags:
      env: dev
```

> **Important**
>
> Jangan tentukan `system:`, atau bagian dari string tersebut, untuk `groupsPrefix` atau `usernamePrefix`.

2. Buat penyedia.

```
eksctl associate identityprovider -f associate-identity-provider.yaml
```

3. Untuk digunakan `kubectl` untuk bekerja dengan klaster dan penyedia OIDC identitas Anda, lihat Menggunakan `kubectl` dalam Kubernetes dokumentasi.

AWS Management Console

Untuk mengaitkan penyedia OIDC identitas ke klaster Anda menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih klaster Anda, lalu pilih tab Access.
3. Di bagian Penyedia OIDC Identitas, pilih Penyedia Identitas Rekanan.
4. Pada halaman Penyedia OIDC Identitas Rekanan, masukkan atau pilih opsi berikut, lalu pilih Rekanan.
   - Untuk Nama, masukkan nama unik untuk penyedia.
   - Untuk Penerbit URL, masukkan URL untuk penyedia Anda. URL ini harus dapat diakses melalui internet.
   - Untuk ID Klien, masukkan ID klien penyedia OIDC identitas (juga dikenal sebagai audiens).

- Untuk Klaim nama pengguna, masukkan klaim untuk digunakan sebagai nama pengguna.
- Untuk Klaim grup, masukkan klaim yang akan digunakan sebagai grup pengguna.
- (Opsional) Pilih Opsi lanjutan, masukkan atau pilih informasi berikut.
  - Prefiks nama pengguna – Masukkan prefiks untuk ditambahkan ke klaim nama pengguna. Prefiks diawali dengan nama pengguna klaim untuk mencegah bentrokan dengan nama yang sudah ada. Jika Anda tidak memberikan nilai, dan nama pengguna adalah nilai selain `email`, prefiks default ke dalam nilai untuk Penerbit URL. Anda dapat menggunakan `-`nilai untuk menonaktifkan semua prefiks. Jangan tentukan `system:` atau bagian dari string tersebut.
  - Prefiks grup – Masukkan prefiks untuk ditambahkan ke klaim grup. Prefiks ditambahkan ke klaim grup untuk mencegah bentrokan dengan nama yang sudah ada (seperti `system: groups`). Misalnya, `oidc:` nilai membuat nama grup seperti `oidc:engineering` dan `oidc:infra`. Jangan tentukan `system:` atau bagian dari string tersebut.
  - Klaim yang diperlukan – Pilih Tambahkan klaim dan masukkan satu atau beberapa pasangan nilai kunci yang menjelaskan klaim yang diperlukan dalam token ID klien. Pasangan menggambarkan klaim yang diperlukan dalam ID Token. Jika diatur, setiap klaim diverifikasi untuk ada dalam token ID dengan nilai yang cocok.

5. Untuk digunakan `kubectl` untuk bekerja dengan klaster dan penyedia OIDC identitas Anda, lihat Menggunakan `kubectl` dalam Kubernetes dokumentasi.

Putuskan hubungan penyedia OIDC identitas dari klaster Anda

Jika Anda memisahkan penyedia OIDC identitas dari klaster Anda, pengguna yang termasuk dalam penyedia tidak dapat lagi mengakses klaster. Namun, Anda masih dapat mengakses cluster dengan prinsipal IAM.

Untuk memisahkan penyedia OIDC identitas dari klaster Anda menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di bagian Penyedia OIDC Identitas, pilih Disassociate, masukkan nama penyedia identitas, lalu pilih`Disassociate`.

Contoh kebijakan IAM

Jika Anda ingin mencegah penyedia OIDC identitas dikaitkan dengan klaster, buat dan kaitkan kebijakan IAM berikut ke akun IAM administrator Amazon EKS Anda. Untuk informasi selengkapnya, lihat Membuat kebijakan IAM dan Menambahkan izin identitas IAM di Panduan Pengguna IAM dan Tindakan, sumber daya, dan kunci kondisi untuk Amazon Elastic Kubernetes Service dalam Referensi Otorisasi Layanan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "denyOIDC",
            "Effect": "Deny",
            "Action": [
                "eks:AssociateIdentityProviderConfig"
            ],
            "Resource": "arn:aws:eks:us-west-2.amazonaws.com:111122223333:cluster/*"

        },
        {
            "Sid": "eksAdmin",
            "Effect": "Allow",
            "Action": [
                "eks:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Contoh kebijakan berikut memungkinkan asosiasi penyedia OIDC identitas jika `clientID` is `kubernetes` dan `issuerUrl` is`https://cognito-idp.us-west-2amazonaws.com/*`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCognitoOnly",
            "Effect": "Deny",
            "Action": "eks:AssociateIdentityProviderConfig",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-instance",
```

```
            "Condition": {
                "StringNotLikeIfExists": {
                    "eks:issuerUrl": "https://cognito-idp.us-west-2.amazonaws.com/*"
                }
            }
        },
        {
            "Sid": "DenyOtherClients",
            "Effect": "Deny",
            "Action": "eks:AssociateIdentityProviderConfig",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-instance",
            "Condition": {
                "StringNotEquals": {
                    "eks:clientId": "kubernetes"
                }
            }
        },
        {
            "Sid": "AllowOthers",
            "Effect": "Allow",
            "Action": "eks:*",
            "Resource": "*"
        }
    ]
}
```

Mitra penyedia OIDC identitas yang divalidasi

Amazon EKS memelihara hubungan dengan jaringan mitra yang menawarkan dukungan untuk penyedia OIDC identitas yang kompatibel. Lihat dokumentasi mitra berikut untuk detail tentang cara mengintegrasikan penyedia identitas dengan Amazon EKS.

Mitra	Produk	Dokumentasi
PingIdentity	PingOne untuk Enterprise	Instruksi instalasi

Amazon EKS bertujuan untuk memberikan berbagai pilihan untuk mencakup semua kasus penggunaan. Jika Anda mengembangkan penyedia identitas OIDC kompatibel yang didukung secara komersial yang tidak tercantum di sini, hubungi tim mitra kami di aws-container-partners@amazon .com untuk informasi lebih lanjut.

Membuat atau memperbarui **kubeconfig** file untuk klaster Amazon EKS

Dalam topik ini, Anda membuat `kubeconfig` file untuk klaster Anda (atau memperbarui yang sudah ada).

Alat `kubectl` baris perintah menggunakan informasi konfigurasi dalam `kubeconfig` file untuk berkomunikasi dengan server API cluster. Untuk informasi selengkapnya, lihat Mengatur Akses Cluster Menggunakan File kubeconfig dalam dokumentasi. Kubernetes

Amazon EKS menggunakan `aws eks get-token` perintah `kubectl` untuk otentikasi cluster. Secara default, AWS CLI menggunakan kredensil yang sama yang dikembalikan dengan perintah berikut:

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

Prasyarat

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS.
- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl` versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.
- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get`,`, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Pengguna IAM atau peran dengan izin untuk menggunakan tindakan `eks:DescribeCluster` API untuk klaster yang Anda tentukan. Untuk informasi selengkapnya, lihat Contoh kebijakan berbasis identitas Amazon EKS. Jika Anda menggunakan identitas dari OpenID Connect penyedia

Anda sendiri untuk mengakses klaster Anda, lihat Menggunakan `kubectl` dalam Kubernetes dokumentasi untuk membuat atau memperbarui `kube config` file Anda.

Buat **`kubeconfig`** file secara otomatis

Prasyarat

- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Izin untuk menggunakan tindakan `eks:DescribeCluster` API untuk klaster yang Anda tentukan. Untuk informasi selengkapnya, lihat Contoh kebijakan berbasis identitas Amazon EKS.

Untuk membuat **`kubeconfig`** file Anda dengan AWS CLI

1. Buat atau perbarui file `kubeconfig` untuk klaster Anda. Ganti *`kode wilayah`* dengan cluster Anda dan ganti *`my-cluster dengan nama cluster`* Anda. Wilayah AWS

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

   Secara default, file konfigurasi yang dihasilkan dibuat di `kubeconfig` jalur default (`.kube`) di direktori home Anda atau digabungkan dengan `config` file yang ada di lokasi tersebut. Anda dapat menentukan jalur lain dengan opsi **`--kubeconfig`**.

   Anda dapat menentukan ARN IAM role dengan pilihan **`--role-arn`** untuk digunakan dalam autentikasi ketika Anda mengeluarkan perintah `kubectl`. Jika tidak, prinsipal IAM dalam rantai kredensi default AWS CLI atau SDK Anda akan digunakan. Anda dapat melihat identitas default AWS CLI atau SDK Anda dengan menjalankan `aws sts get-caller-identity` perintah.

   Untuk semua opsi yang tersedia, jalankan `aws eks update-kubeconfig help` perintah atau lihat `update-kubeconfig`di Referensi AWS CLI Perintah.

2. Uji konfigurasi Anda.

```
kubectl get svc
```

Contoh output adalah sebagai berikut.

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

Jika Anda menerima kesalahan otorisasi atau jenis sumber daya, lihat Tidak sah atau akses ditolak (`kubectl`) di topik pemecahan masalah.

Berikan akses beban kerja Kubernetes untuk menggunakan Akun Layanan AWSKubernetes

Akun Kubernetes layanan memberikan identitas untuk proses yang berjalan di filePod. Untuk informasi selengkapnya, lihat Mengelola Akun Layanan di Kubernetes dokumentasi. Jika Anda Pod membutuhkan akses ke AWS layanan, Anda dapat memetakan akun layanan ke AWS Identity and Access Management identitas untuk memberikan akses tersebut. Untuk informasi selengkapnya, lihat IAM role untuk akun layanan.

Token akun layanan

`BoundServiceAccountTokenVolume`Fitur ini diaktifkan secara default dalam Kubernetes versi. Fitur ini meningkatkan keamanan token akun layanan dengan memungkinkan beban kerja berjalan Kubernetes untuk meminta token web JSON yang merupakan pemirsa, waktu, dan kunci terikat. Token akun layanan memiliki masa kedaluwarsa satu jam. Di Kubernetes versi sebelumnya, token tidak memiliki kedaluwarsa. Ini berarti bahwa klien yang mengandalkan token ini harus menyegarkan token dalam waktu satu jam. SDK Kubernetes klien berikut menyegarkan token secara otomatis dalam jangka waktu yang diperlukan:

- Versi Go `0.15.7` dan yang lebih baru
- Versi `12.0.0` Python dan yang lebih baru
- Versi Java `9.0.0` dan yang lebih baru
- JavaScript versi `0.10.3` dan yang lebih baru
- Cabang Ruby `master`

- Versi Haskell `0.3.0.0`
- C#versi `7.0.5` dan yang lebih baru

Jika beban kerja Anda menggunakan versi klien sebelumnya, maka Anda harus memperbaruinya. Untuk mengaktifkan kelancaran migrasi klien ke token akun layanan terikat waktu yang lebih baru, Kubernetes tambahkan periode kedaluwarsa yang diperpanjang ke token akun layanan selama satu jam default. Untuk cluster Amazon EKS, periode kedaluwarsa yang diperpanjang adalah 90 hari. Server Kubernetes API klaster Amazon EKS Anda menolak permintaan dengan token yang berusia lebih dari 90 hari. Kami menyarankan Anda memeriksa aplikasi Anda dan dependensinya untuk memastikan bahwa SDK klien Kubernetes sama atau lebih lambat dari versi yang tercantum sebelumnya.

Saat server API menerima permintaan dengan token yang berumur lebih dari satu jam, server API akan menganotasi peristiwa log audit API. `annotations.authentication.k8s.io/stale-token` Nilai anotasi terlihat seperti contoh berikut:

```
subject: system:serviceaccount:common:fluent-bit, seconds after warning threshold:
 4185802.
```

Jika klaster Anda mengaktifkan pencatatan bidang kontrol, maka anotasi ada di log audit. Anda dapat menggunakan kueri Wawasan CloudWatch Log berikut untuk mengidentifikasi semua Pods di klaster Amazon EKS Anda yang menggunakan token basi:

```
fields @timestamp
| filter @logStream like /kube-apiserver-audit/
| filter @message like /seconds after warning threshold/
| parse @message "subject: *, seconds after warning threshold:*\"" as subject,
 elapsedtime
```

`subject`Ini mengacu pada akun layanan yang Pod digunakan. `elapsedtime`Ini menunjukkan waktu yang telah berlalu (dalam detik) setelah membaca token terbaru. Permintaan ke server API ditolak ketika `elapsedtime` melebihi 90 hari (7.776.000 detik). Anda harus secara proaktif memperbarui SDK Kubernetes klien aplikasi Anda untuk menggunakan salah satu versi yang tercantum sebelumnya yang secara otomatis menyegarkan token. Jika token akun layanan yang digunakan mendekati 90 hari dan Anda tidak memiliki cukup waktu untuk memperbarui versi SDK klien Anda sebelum token kedaluwarsa, maka Anda dapat menghentikan yang sudah ada Pods dan

membuat yang baru. Ini menghasilkan refetching token akun layanan, memberi Anda tambahan 90 hari untuk memperbarui SDK versi klien Anda.

Jika Pod merupakan bagian dari penerapan, cara yang disarankan untuk menghentikan Pods sambil menjaga ketersediaan tinggi adalah dengan melakukan peluncuran dengan perintah berikut. Ganti *my-deployment* dengan nama penerapan Anda.

```
kubectl rollout restart deployment/my-deployment
```

Pengaya klaster

Add-on klaster berikut telah diperbarui untuk menggunakan SDK Kubernetes klien yang secara otomatis mengisi ulang token akun layanan. Sebaiknya pastikan bahwa versi yang terdaftar, atau versi yang lebih baru, diinstal pada cluster Anda.

- Amazon VPC CNI plugin for Kubernetesdan plugin pembantu metrik versi `1.8.0` dan yang lebih baru. Untuk memeriksa versi Anda saat ini atau memperbaruinya, lihat Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS dan cni-metrics-helper.
- CoreDNSversi `1.8.4` dan yang lebih baru. Untuk memeriksa versi Anda saat ini atau memperbaruinya, lihatBekerja dengan add-on CoreDNS Amazon EKS.
- AWS Load Balancer Controllerversi `2.0.0` dan yang lebih baru. Untuk memeriksa versi Anda saat ini atau memperbaruinya, lihatApa itu AWS Load Balancer Controller?.
- `kube-proxy`Versi saat ini. Untuk memeriksa versi Anda saat ini atau memperbaruinya, lihatBekerja dengan add-on Kubernetes `kube-proxy`.
- AWS untuk versi Fluent Bit `2.25.0` atau yang lebih baru. Untuk memperbarui versi Anda saat ini, lihat Rilis diGitHub.
- Fluentd image versi 1.14.6-1.2 atau yang lebih baru dan plugin filter Fluentd untuk metadata Kubernetes versi 2.11.1 atau yang lebih baru.

Memberikan AWS Identity and Access Management izin untuk beban kerja di klaster Amazon Elastic Kubernetes Service

Amazon EKS menyediakan dua cara untuk memberikan AWS Identity and Access Management izin ke beban kerja yang berjalan di klaster Amazon EKS: peran IAM untuk akun layanan, dan Identitas Pod EKS.

IAM role untuk akun layanan

Peran IAM untuk akun layanan (IRSA) mengonfigurasi aplikasi Kubernetes yang berjalan AWS dengan izin IAM berbutir halus untuk mengakses berbagai sumber daya lain seperti bucket AWS Amazon S3, tabel Amazon DynamoDB, dan banyak lagi. Anda dapat menjalankan beberapa aplikasi bersama-sama dalam kluster Amazon EKS yang sama, dan memastikan setiap aplikasi hanya memiliki set izin minimum yang diperlukan. IRSA dibangun untuk mendukung berbagai opsi Kubernetes penerapan yang didukung oleh AWS seperti Amazon EKS, Amazon EKS Anywhere, Layanan OpenShift Red Hat di AWS, dan Kubernetes cluster yang dikelola sendiri di instans Amazon EC2. Dengan demikian, IRSA dibangun menggunakan AWS layanan dasar seperti IAM, dan tidak mengambil ketergantungan langsung pada layanan Amazon EKS dan EKS API. Untuk informasi selengkapnya, lihat IAM role untuk akun layanan.

Identitas EKS Pod

EKS Pod Identity menawarkan kepada administrator klaster alur kerja yang disederhanakan untuk mengautentikasi aplikasi untuk mengakses berbagai AWS sumber daya lain seperti bucket Amazon S3, tabel Amazon DynamoDB, dan banyak lagi. EKS Pod Identity hanya untuk EKS, dan sebagai hasilnya, ini menyederhanakan bagaimana administrator klaster dapat mengkonfigurasi aplikasi Kubernetes untuk mendapatkan izin IAM. Izin ini sekarang dapat dengan mudah dikonfigurasi dengan lebih sedikit langkah langsung melalui AWS Management Console, EKS API, dan AWS CLI, dan tidak ada tindakan apa pun untuk diambil di dalam cluster di Kubernetes objek apa pun. Administrator klaster tidak perlu beralih antara layanan EKS dan IAM, atau menggunakan operasi IAM istimewa untuk mengonfigurasi izin yang diperlukan oleh aplikasi Anda. Peran IAM sekarang dapat digunakan di beberapa cluster tanpa perlu memperbarui kebijakan kepercayaan peran saat membuat cluster baru. Kredensi IAM yang disediakan oleh EKS Pod Identity mencakup tag sesi peran, dengan atribut seperti nama cluster, namespace, nama akun layanan. Tag sesi peran memungkinkan administrator untuk membuat peran tunggal yang dapat bekerja di seluruh akun layanan dengan mengizinkan akses ke AWS sumber daya berdasarkan tag yang cocok. Untuk informasi selengkapnya, lihat Identitas EKS Pod.

Membandingkan Identitas Pod EKS dan IRSA

Pada tingkat tinggi, baik EKS Pod Identity dan IRSA memungkinkan Anda untuk memberikan izin IAM untuk aplikasi yang berjalan pada klaster Kubernetes. Tetapi mereka pada dasarnya berbeda dalam cara Anda mengonfigurasinya, batas yang didukung, dan fitur yang diaktifkan. Di bawah ini, kami membandingkan beberapa aspek kunci dari kedua solusi.

	Identitas Pod EKS	IRSA
Ekstensibilitas peran	Anda harus mengatur setiap peran sekali untuk membangun kepercayaan dengan prinsipal layanan Amazon EKS yang baru diperkenalkan. `pods.eks.amazonaws.com` Setelah langkah satu kali ini, Anda tidak perlu memperbarui kebijakan kepercayaan peran setiap kali digunakan di klaster baru.	Anda harus memperbarui kebijakan kepercayaan peran IAM dengan titik akhir OIDC penyedia kluster EKS baru setiap kali Anda ingin menggunakan peran dalam klaster baru.
Skalabilitas cluster	EKS Pod Identity tidak mengharuskan pengguna untuk menyiapkan penyedia IAM OIDC, jadi batasan ini tidak berlaku.	Setiap kluster EKS memiliki URL penerbit OpenID Connect (OIDC) yang terkait dengannya. Untuk menggunakan IRSA, OpenID Connect penyedia unik perlu dibuat untuk setiap cluster EKS di IAM. IAM memiliki batas global default 100 OIDC penyedia untuk masing-masing Akun AWS. Jika Anda berencana untuk memiliki lebih dari 100 kluster EKS untuk masing-masing Akun AWS dengan IRSA, maka Anda akan mencapai batas penyedia IAMOIDC.
Skalabilitas peran	EKS Pod Identity tidak mengharuskan pengguna untuk menentukan hubungan kepercayaan antara peran IAM dan akun layanan dalam	Dalam IRSA, Anda mendefinisikan hubungan kepercayaan antara peran IAM dan akun layanan dalam kebijakan kepercayaan peran. Secara

	Identitas Pod EKS	IRSA
	kebijakan kepercayaan, sehingga batasan ini tidak berlaku.	default, panjang ukuran kebijakan kepercayaan adalah2048. Ini berarti bahwa Anda biasanya dapat mendefinisikan 4 hubungan kepercayaan dalam satu kebijakan kepercaya an. Meskipun Anda bisa mendapatkan batas panjang kebijakan kepercayaan yang meningkat, Anda biasanya terbatas pada maksimal 8 hubungan kepercayaan dalam satu kebijakan kepercayaan.

	Identitas Pod EKS	IRSA
Peran dapat digunakan kembali	AWS STS Kredensi sementara yang disediaka n oleh EKS Pod Identity mencakup tag sesi peran, seperti nama klaster, namespace, nama akun layanan. Tag sesi peran memungkinkan administr ator untuk membuat peran IAM tunggal yang dapat digunakan dengan beberapa akun layanan, dengan izin efektif yang berbeda, dengan mengizinkan akses ke AWS sumber daya berdasarkan tag yang dilampirkan padanya. Ini juga disebut Attribute-based Access Control (ABAC). Untuk informasi selengkap nya, lihat Menentukan izin untuk Identitas Pod EKS untuk mengambil peran berdasarkan tag.	AWS STS tag sesi tidak didukung. Anda dapat menggunakan kembali peran antar klaster, tetapi setiap pod menerima semua izin peran tersebut.
Lingkungan yang didukung	EKS Pod Identity hanya tersedia di Amazon EKS.	IRSA dapat digunakan seperti Amazon EKS, Amazon EKS Anywhere Layanan OpenShift Red Hat di AWS, dan Kubernetes cluster yang dikelola sendiri di instans Amazon EC2.

	Identitas Pod EKS	IRSA
Versi EKS didukung	KubernetesVersi EKS `1.24` atau yang lebih baru. Untuk versi platform tertentu, lihatEKS Pod Identity versi cluster.	Semua versi cluster EKS yang didukung.

Identitas EKS Pod

Aplikasi dalam Pod kontainer dapat menggunakan AWS SDK atau AWS CLI untuk membuat permintaan API untuk Layanan AWS menggunakan izin AWS Identity and Access Management (IAM). Aplikasi harus menandatangani permintaan AWS API mereka dengan AWS kredensialnya.

EKS Pod Identities menyediakan kemampuan untuk mengelola kredensyal untuk aplikasi Anda, mirip dengan cara profil instans Amazon EC2 memberikan kredensyal ke instans Amazon EC2. Alih-alih membuat dan mendistribusikan AWS kredensyal Anda ke container atau menggunakan peran instans Amazon EC2, Anda mengaitkan peran IAM dengan akun layanan dan mengonfigurasi Anda Pods untuk menggunakan akun Kubernetes layanan.

Setiap asosiasi EKS Pod Identity memetakan peran ke akun layanan di namespace di klaster yang ditentukan. Jika Anda memiliki aplikasi yang sama di beberapa cluster, Anda dapat membuat asosiasi identik di setiap cluster tanpa mengubah kebijakan kepercayaan peran tersebut.

Jika sebuah pod menggunakan akun layanan yang memiliki asosiasi, Amazon EKS menetapkan variabel lingkungan dalam kontainer pod. Variabel lingkungan mengonfigurasi AWS SDK, termasuk AWS CLI, untuk menggunakan kredenal Identitas Pod EKS.

Manfaat Identitas EKS Pod

EKS Pod Identities memberikan manfaat sebagai berikut:

- Keistimewaan paling sedikit — Anda dapat mencakup izin IAM ke akun layanan, dan hanya akun layanan Pods yang menggunakan akun layanan tersebut yang memiliki akses ke izin tersebut. Fitur ini juga mengurangi kebutuhan akan solusi dari pihak ketiga seperti `kiam` atau `kube2iam`.
- Isolasi kredenal — Pod's Kontainer hanya dapat mengambil kredensional untuk peran IAM yang terkait dengan akun layanan yang digunakan penampung. Kontainer tidak pernah memiliki akses ke kredensional yang digunakan oleh kontainer lain di wadah lain. Pods Saat menggunakan

Identitas Pod, Pod's container juga memiliki izin yang ditetapkan ke peran IAM node Amazon EKS, kecuali jika Anda memblokir Pod akses ke Amazon EC2 Instance Metadata Service (IMDS). Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

- Auditabilitas — Akses dan pencatatan peristiwa tersedia AWS CloudTrail untuk membantu memfasilitasi audit retrospektif.

EKS Pod Identity adalah metode yang lebih sederhana daripadaIAM role untuk akun layanan, karena metode ini tidak menggunakan penyedia OIDC identitas. EKS Pod Identity memiliki penyempurnaan sebagai berikut:

- Operasi independen — Di banyak organisasi, menciptakan penyedia OIDC identitas adalah tanggung jawab tim yang berbeda daripada mengelola Kubernetes cluster. EKS Pod Identity memiliki pemisahan tugas yang bersih, di mana semua konfigurasi asosiasi Identitas Pod EKS dilakukan di Amazon EKS dan semua konfigurasi izin IAM dilakukan di IAM.
- Reusability — EKS Pod Identity menggunakan satu prinsipal IAM, bukan prinsipal terpisah untuk setiap klaster yang digunakan peran IAM untuk akun layanan. Administrator IAM Anda menambahkan prinsipal berikut ke kebijakan kepercayaan dari peran apa pun agar dapat digunakan oleh EKS Pod Identities.

```
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            }
```

- Skalabilitas — Setiap set kredensial sementara diasumsikan oleh EKS Auth layanan di EKS Pod Identity, bukan setiap AWS SDK yang Anda jalankan di setiap pod. Kemudian, Amazon EKS Pod Identity Agent yang berjalan di setiap node mengeluarkan kredensialnya ke SDK. Dengan demikian beban dikurangi menjadi satu kali untuk setiap node dan tidak diduplikasi di setiap pod. Untuk detail lebih lanjut tentang prosesnya, lihatCara kerja EKS Pod Identity.

Untuk informasi lebih lanjut untuk membandingkan dua alternatif, lihatBerikan akses beban kerja Kubernetes untuk menggunakan Akun Layanan AWSKubernetes.

Ikhtisar pengaturan EKS Pod Identities

Aktifkan Identitas Pod EKS dengan menyelesaikan prosedur berikut:

1. Siapkan Agen Identitas Amazon EKS Pod— Anda hanya menyelesaikan prosedur ini sekali untuk setiap cluster.

2. Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM dengan EKS Pod Identity— Selesaikan prosedur ini untuk setiap set izin unik yang Anda inginkan untuk dimiliki aplikasi.

3. PodsKonfigurasikan untuk menggunakan akun Kubernetes layanan— Selesaikan prosedur ini untuk masing-masing Pod yang membutuhkan akses Layanan AWS.
4. Menggunakan AWS SDK yang didukung— Konfirmasikan bahwa beban kerja menggunakan AWS SDK versi yang didukung dan bahwa beban kerja menggunakan rantai kredenal default.

Pertimbangan Identitas EKS Pod

- Anda dapat mengaitkan satu peran IAM ke setiap akun Kubernetes layanan di setiap cluster. Anda dapat mengubah peran mana yang dipetakan ke akun layanan dengan mengedit asosiasi EKS Pod Identity.
- Anda hanya dapat mengaitkan peran yang Akun AWS sama dengan cluster. Anda dapat mendelegasikan akses dari akun lain ke peran di akun ini yang Anda konfigurasikan untuk Identitas Pod EKS untuk digunakan. Untuk tutorial tentang mendelegasikan akses dan`AssumeRole`, lihat Mendelegasikan akses di seluruh AWS akun menggunakan peran IAM dalam Panduan Pengguna IAM.
- Agen Identitas Pod EKS diperlukan. Ini berjalan sebagai a Kubernetes `DaemonSet` pada node Anda dan hanya menyediakan kredensional untuk pod pada node yang dijalankan. Untuk informasi selengkapnya tentang kompatibilitas Agen Identitas Pod EKS, lihat bagian berikutPembatasan Identitas Pod EKS.
- EKS Pod Identity Agent menggunakan node dan menggunakan port `80` dan port `2703` pada alamat link-lokal pada node. `hostNetwork` Alamat ini `169.254.170.23` untuk IPv4 dan `[fd00:ec2::23]` untuk IPv6 cluster.

EKS Pod Identity versi cluster

Untuk menggunakan EKS Pod Identities, cluster harus memiliki versi platform yang sama atau lebih lambat dari versi yang tercantum dalam tabel berikut, atau Kubernetes versi yang lebih lambat dari versi yang tercantum dalam tabel.

Versi Kubernetes	Versi platform
`1.28`	`eks.4`

Versi Kubernetes	Versi platform
1.27	eks.8
1.26	eks.9
1.25	eks.10
1.24	eks.13

Pembatasan Identitas Pod EKS

Identitas Pod EKS tersedia pada hal-hal berikut:

- Amazon EKS versi cluster tercantum dalam topik sebelumnyaEKS Pod Identity versi cluster.
- Node pekerja di cluster yang merupakan instans Linux Amazon EC2.

EKS Pod Identities tidak tersedia di berikut ini:

- Wilayah China.
- AWS GovCloud (US).
- AWS Outposts.
- Amazon EKS Anywhere.
- Kubernetescluster yang Anda buat dan jalankan di Amazon EC2. Komponen EKS Pod Identity hanya tersedia di Amazon EKS.

Anda tidak dapat menggunakan EKS Pod Identities dengan:

- Pod yang berjalan di mana saja kecuali instans Linux Amazon EC2. Pod Linux dan Windows yang berjalan AWS Fargate (Fargate) tidak didukung. Pod yang berjalan di instans Windows Amazon EC2 tidak didukung.
- Add-on Amazon EKS yang membutuhkan kredensi IAM. Pengaya EKS hanya dapat menggunakan peran IAM untuk akun layanan sebagai gantinya. Daftar add-on EKS yang menggunakan kredensi IAM meliputi:
  - Amazon VPC CNI plugin for Kubernetes
  - AWS Load Balancer Controller

- Driver CSI penyimpanan: EBS CSI, EFS CSI, Amazon FSx for Lustre CSI driver, Amazon FSx untuk driver ONTAP CSI, Amazon NetApp FSx untuk driver OpenZFS CSI, driver CSI Cache File Amazon

> **Note**
>
> Jika pengontrol, driver, dan plugin ini diinstal sebagai add-on yang dikelola sendiri, bukan add-on EKS, mereka mendukung EKS Pod Identities selama mereka diperbarui untuk menggunakan SDK terbaru. AWS

Cara kerja EKS Pod Identity

Asosiasi Amazon EKS Pod Identity menyediakan kemampuan untuk mengelola kredensional untuk aplikasi Anda, mirip dengan cara profil instans Amazon EC2 memberikan kredensional ke instans Amazon EC2.

Amazon EKS Pod Identity memberikan kredensi ke beban kerja Anda dengan API Auth EKS tambahan dan pod agen yang berjalan di setiap node.

Di add-on Anda, seperti add-on Amazon EKS dan pengontrol yang dikelola sendiri, operator, dan add-on lainnya, penulis perlu memperbarui perangkat lunak mereka untuk menggunakan SDK terbaru. AWS Untuk daftar kompatibilitas antara EKS Pod Identity dan add-on yang diproduksi oleh Amazon EKS, lihat bagian Pembatasan Identitas Pod EKS sebelumnya.

Menggunakan Identitas Pod EKS dalam kode Anda

Dalam kode Anda, Anda dapat menggunakan AWS SDK untuk mengakses AWS layanan. Anda menulis kode untuk membuat klien untuk AWS layanan dengan SDK, dan secara default SDK mencari di rantai lokasi untuk menggunakan AWS Identity and Access Management kredensil. Setelah kredensi yang valid ditemukan, pencarian dihentikan. Untuk informasi selengkapnya tentang lokasi default yang digunakan, lihat rantai penyedia kredenal di Panduan Referensi AWS SDK dan Alat.

EKS Pod Identities telah ditambahkan ke penyedia kredensi Container yang dicari dalam satu langkah dalam rantai kredensi default. Jika beban kerja Anda saat ini menggunakan kredensial yang sebelumnya ada dalam rantai kredensil, kredensial-kredensialnya akan terus digunakan meskipun Anda mengonfigurasi asosiasi Identitas Pod EKS untuk beban kerja yang sama. Dengan cara ini

Anda dapat dengan aman bermigrasi dari jenis kredensil lain dengan membuat asosiasi terlebih dahulu, sebelum menghapus kredensil lama.

Penyedia kredensial kontainer menyediakan kredensi sementara dari agen yang berjalan di setiap node. Di Amazon EKS, agennya adalah Agen Identitas Pod Amazon EKS dan di Amazon Elastic Container Service agennya adalah`amazon-ecs-agent`. SDK menggunakan variabel lingkungan untuk menemukan agen untuk terhubung.

Sebaliknya, peran IAM untuk akun layanan menyediakan token identitas web yang harus ditukar dengan AWS SDK AWS Security Token Service dengan menggunakan. `AssumeRoleWithWebIdentity`

Bagaimana Agen Identitas EKS Pod bekerja dengan Pod

1. Saat Amazon EKS memulai pod baru yang menggunakan akun layanan dengan asosiasi Identitas Pod EKS, klaster akan menambahkan konten berikut ke Pod manifes:

```
    env:
    - name: AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE
      value: "/var/run/secrets/pods.eks.amazonaws.com/serviceaccount/eks-pod-
identity-token"
    - name: AWS_CONTAINER_CREDENTIALS_FULL_URI
      value: "http://169.254.170.23/v1/credentials"
    volumeMounts:
    - mountPath: "/var/run/secrets/pods.eks.amazonaws.com/serviceaccount/"
      name: eks-pod-identity-token
  volumes:
  - name: eks-pod-identity-token
    projected:
      defaultMode: 420
      sources:
      - serviceAccountToken:
          audience: pods.eks.amazonaws.com
          expirationSeconds: 86400 # 24 hours
          path: eks-pod-identity-token
```

2. Kubernetesmemilih node mana untuk menjalankan pod. Kemudian, Amazon EKS Pod Identity Agent pada node menggunakan AssumeRoleForPodIdentityaction untuk mengambil kredensial sementara dari EKS Auth API.
3. Agen Identitas Pod EKS membuat kredensil ini tersedia untuk AWS SDK yang Anda jalankan di dalam container Anda.

4. Anda menggunakan SDK dalam aplikasi Anda tanpa menentukan penyedia kredensi untuk menggunakan rantai kredensi default. Atau, Anda menentukan penyedia kredensi kontainer. Untuk informasi selengkapnya tentang lokasi default yang digunakan, lihat rantai penyedia kredenal di Panduan Referensi AWS SDK dan Alat.
5. SDK menggunakan variabel lingkungan untuk terhubung ke Agen Identitas Pod EKS dan mengambil kredensialnya.

> **Note**
>
> Jika beban kerja Anda saat ini menggunakan kredensial yang sebelumnya ada dalam rantai kredensil, kredensial-kredensialnya akan terus digunakan meskipun Anda mengonfigurasi asosiasi Identitas Pod EKS untuk beban kerja yang sama.

Siapkan Agen Identitas Amazon EKS Pod

Asosiasi Amazon EKS Pod Identity menyediakan kemampuan untuk mengelola kredensional untuk aplikasi Anda, mirip dengan cara profil instans Amazon EC2 memberikan kredensional ke instans Amazon EC2.

Amazon EKS Pod Identity memberikan kredensi ke beban kerja Anda dengan API Auth EKS tambahan dan pod agen yang berjalan di setiap node.

Membuat Agen Identitas Amazon EKS Pod

Prasyarat agen

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS. Versi cluster dan versi platform harus sama atau lebih lambat dari versi yang tercantum diEKS Pod Identity versi cluster.
- Peran node memiliki izin bagi agen untuk melakukan `AssumeRoleForPodIdentity` tindakan di EKS Auth API. Anda dapat menggunakan AWS kebijakan terkelola: AmazonEks WorkerNodePolicy atau menambahkan kebijakan khusus yang serupa dengan berikut ini:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "eks-auth:AssumeRoleForPodIdentity",
            ],
            "Resource": "*"
        }
    ]
}
```

Tindakan ini dapat dibatasi oleh tag untuk membatasi peran mana yang dapat diasumsikan oleh pod yang menggunakan agen.

- Node dapat menjangkau dan mengunduh gambar dari Amazon ECR. Gambar kontainer untuk add-on ada di registri yang tercantum di. Pendaftar gambar kontainer Amazon

  Perhatikan bahwa Anda dapat mengubah lokasi gambar dan `imagePullSecrets` menyediakan add-on EKS di pengaturan konfigurasi Opsional di AWS Management Console, dan di `--configuration-values` AWS CLI dalam.
- Node dapat mencapai Amazon EKS Auth API. Untuk cluster pribadi, `eks-auth` titik akhir di AWS PrivateLink diperlukan.

AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel navigasi sebelah kiri, pilih Clusters, lalu pilih nama cluster yang ingin Anda konfigurasikan untuk add-on EKS Pod Identity Agent.
3. Pilih tab Add-ons.
4. Pilih Get more add-ons
5. Pilih kotak di kanan atas kotak add-on untuk EKS Pod Identity Agent dan kemudian pilih Berikutnya.
6. Pada halaman Konfigurasi pengaturan add-on yang dipilih, pilih versi apa pun di daftar dropdown Versi.
7. (Opsional) Perluas pengaturan konfigurasi opsional untuk memasukkan konfigurasi tambahan. Misalnya, Anda dapat memberikan lokasi gambar kontainer alternatif dan`ImagePullSecrets`. Kunci JSON Schema dengan diterima ditampilkan dalam skema konfigurasi Add-on.

   Masukkan tombol konfigurasi dan nilai dalam nilai Konfigurasi.

8. Pilih Berikutnya.
9. Konfirmasikan bahwa pod EKS Pod Identity Agent berjalan di klaster Anda.

```
kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
```

Contoh output adalah sebagai berikut.

```
eks-pod-identity-agent-gmqp7                                          1/1
 Running   1 (24h ago)   24h
eks-pod-identity-agent-prnsh                                          1/1
 Running   1 (24h ago)   24h
```

Anda sekarang dapat menggunakan asosiasi EKS Pod Identity di klaster Anda. Untuk informasi selengkapnya, lihat Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM dengan EKS Pod Identity.

AWS CLI

1. Jalankan AWS CLI perintah berikut. Ganti `my-cluster` dengan nama klaster Anda.

```
aws eks create-addon --cluster-name my-cluster --addon-name eks-pod-identity-
agent --addon-version v1.0.0-eksbuild.1
```

> **Note**
>
> Agen Identitas Pod EKS tidak menggunakan peran *service-account-role-arn* untuk IAM untuk akun layanan. Anda harus memberikan izin kepada Agen Identitas Pod EKS dalam peran node.

2. Konfirmasikan bahwa pod EKS Pod Identity Agent berjalan di klaster Anda.

```
kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
```

Contoh output adalah sebagai berikut.

```
eks-pod-identity-agent-gmqp7                                          1/1
 Running   1 (24h ago)   24h
```

```
eks-pod-identity-agent-prnsh                                          1/1
 Running   1 (24h ago)   24h
```

Anda sekarang dapat menggunakan asosiasi EKS Pod Identity di klaster Anda. Untuk informasi selengkapnya, lihat Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM dengan EKS Pod Identity.

Memperbarui Agen Identitas Pod Amazon EKS

Perbarui jenis add-on Amazon EKS. Jika Anda belum menambahkan jenis add-on Amazon EKS ke cluster Anda, lihatMembuat Agen Identitas Amazon EKS Pod.

AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel navigasi sebelah kiri, pilih Clusters, lalu pilih nama cluster yang ingin Anda konfigurasikan untuk add-on EKS Pod Identity Agent.
3. Pilih tab Add-ons.
4. Jika versi baru dari add-on tersedia, Agen Identitas Pod EKS memiliki tombol versi Perbarui. Pilih Perbarui versi.
5. Pada halaman Configure Amazon EKS Pod Identity Agent, pilih versi baru di daftar dropdown Versi.
6. Pilih Simpan perubahan.

   Mungkin perlu beberapa detik untuk pembaruan selesai. Kemudian, konfirmasikan bahwa versi add-on telah diperbarui dengan memeriksa Status.

AWS CLI

1. Lihat versi add-on mana yang diinstal pada cluster Anda. Ganti *my-cluster* dengan nama klaster Anda.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent --query "addon.addonVersion" --output text
   ```

   Contoh output adalah sebagai berikut.

```
v1.0.0-eksbuild.1
```

Anda perlu membuat add-on sebelum Anda dapat memperbaruinya dengan prosedur ini.

2. Perbarui add-on Anda menggunakan. AWS CLI Jika Anda ingin menggunakan AWS Management Console atau `eksctl` memperbarui add-on, lihatMemperbarui add-on. Salin perintah yang mengikuti ke perangkat Anda. Buat modifikasi berikut pada perintah, sesuai kebutuhan, dan kemudian jalankan perintah yang dimodifikasi.

   - Ganti *my-cluster* dengan nama klaster Anda.
   - Ganti *v1.0.0-eksbuild.1*dengan versi yang Anda inginkan.
   - Ganti *111122223333* dengan ID akun Anda.
   - Jalankan perintah berikut:

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent --addon-version v1.0.0-eksbuild.1'
   ```

   Mungkin perlu beberapa detik untuk pembaruan selesai.

3. Konfirmasikan bahwa versi add-on telah diperbarui. Ganti *my-cluster* dengan nama klaster Anda.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent
   ```

   Mungkin perlu beberapa detik untuk pembaruan selesai.

   Contoh output adalah sebagai berikut.

   ```
   {
       "addon": {
           "addonName": "eks-pod-identity-agent",
           "clusterName": "my-cluster",
           "status": "ACTIVE",
           "addonVersion": "v1.0.0-eksbuild.1",
           "health": {
               "issues": []
           },
   ```

```
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/eks-pod-
identity-agent/74c33d2f-b4dc-8718-56e7-9fdfa65d14a9",
        "createdAt": "2023-04-12T18:25:19.319000+00:00",
        "modifiedAt": "2023-04-12T18:40:28.683000+00:00",
        "tags": {}
    }
}
```

Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM dengan EKS Pod Identity

Topik ini mencakup cara mengonfigurasi akun Kubernetes layanan untuk mengambil peran AWS Identity and Access Management (IAM) dengan EKS Pod Identity. Apa pun Pods yang dikonfigurasi untuk menggunakan akun layanan kemudian dapat mengakses apa pun Layanan AWS yang peran memiliki izin untuk diakses.

Untuk membuat asosiasi EKS Pod Identity, hanya ada satu langkah; Anda membuat asosiasi di EKS melalui AWS Management Console, AWS CLI, AWS SDK, AWS CloudFormation dan alat lainnya. Tidak ada data atau metadata tentang asosiasi di dalam cluster di Kubernetes objek apa pun dan Anda tidak menambahkan anotasi apa pun ke akun layanan.

Prasyarat

- Sebuah klaster yang sudah ada. Jika Anda tidak memilikinya, Anda dapat membuatnya dengan mengikuti salah satu Memulai dengan Amazon EKS panduan.
- Prinsip IAM yang menciptakan asosiasi harus memiliki`iam:PassRole`.
- Versi terbaru dari yang AWS CLI diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Anda dapat memeriksa versi saat ini dengan `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Package manager seperti `yum`apt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru dari file AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan `aws configure` di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal di AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl`

versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.

- `kubectl` `config`File yang sudah ada yang berisi konfigurasi cluster Anda. Untuk membuat `kubectl config` file, lihatMembuat atau memperbarui `kubeconfig` file untuk klaster Amazon EKS.

Membuat asosiasi Identitas Pod EKS

AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel navigasi sebelah kiri, pilih Clusters, lalu pilih nama cluster yang ingin Anda konfigurasikan untuk add-on EKS Pod Identity Agent.
3. Pilih tab Access.
4. Dalam asosiasi Pod Identity, pilih Create.
5. Untuk peran IAM, pilih peran IAM dengan izin yang Anda inginkan untuk memiliki beban kerja.

> **Note**
>
> Daftar ini hanya berisi peran yang memiliki kebijakan kepercayaan berikut yang memungkinkan EKS Pod Identity untuk menggunakannya.

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

```
    ]
}
```

`sts:AssumeRole`

EKS Pod Identity menggunakan `TagSession` untuk mengambil peran IAM sebelum meneruskan kredensialnya sementara ke pod Anda.

`sts:TagSession`

EKS Pod Identity menggunakan `TagSession` untuk menyertakan tag sesi dalam permintaan untuk AWS STS.

Anda dapat menggunakan tag ini dalam kebijakan kepercayaan untuk membatasi akun layanan, ruang nama, dan klaster mana yang dapat menggunakan peran ini. condition keys

Untuk daftar kunci kondisi Amazon EKS, lihat Ketentuan yang ditentukan oleh Amazon Elastic Kubernetes Service di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat digunakan untuk menggunakan kunci kondisi, lihat Tindakan yang ditentukan oleh Amazon Elastic Kubernetes Service.

6. Untuk Kubernetesnamespace, pilih Kubernetes namespace yang berisi akun layanan dan beban kerja. Secara opsional, Anda dapat menentukan namespace dengan nama yang tidak ada di cluster.
7. Untuk akun Kubernetes layanan, pilih akun Kubernetes layanan yang akan digunakan. Manifes untuk Kubernetes beban kerja Anda harus menentukan akun layanan ini. Secara opsional, Anda dapat menentukan akun layanan berdasarkan nama yang tidak ada di cluster.
8. (Opsional) Untuk Tag, pilih Tambahkan tag untuk menambahkan metadata dalam pasangan kunci dan nilai. Tag ini diterapkan pada asosiasi dan dapat digunakan dalam kebijakan IAM.

   Anda dapat mengulangi langkah ini untuk menambahkan beberapa tag.
9. Pilih Buat.

AWS CLI

1. Jika Anda ingin mengaitkan kebijakan IAM yang ada dengan peran IAM Anda, lewati ke langkah berikutnya.

Buat kebijakan IAM. Anda dapat membuat kebijakan sendiri, atau menyalin kebijakan AWS terkelola yang telah memberikan beberapa izin yang Anda perlukan dan menyesuaikannya dengan persyaratan spesifik Anda. Untuk informasi selengkapnya, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

a. Buat file yang menyertakan izin untuk Layanan AWS yang ingin Anda Pods akses. Untuk daftar semua tindakan untuk semua Layanan AWS, lihat Referensi Otorisasi Layanan.

   Anda dapat menjalankan perintah berikut untuk membuat contoh file kebijakan yang memungkinkan akses hanya-baca ke bucket Amazon S3. Anda dapat secara opsional menyimpan informasi konfigurasi atau skrip bootstrap di bucket ini, dan kontainer di dalam Anda Pod dapat membaca file dari bucket dan memuatnya ke dalam aplikasi Anda. Jika Anda ingin membuat kebijakan contoh ini, salin konten berikut ke perangkat Anda. Ganti *my-pod-secrets-bucket* dengan nama bucket Anda dan jalankan perintah.

```
cat >my-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
EOF
```

b. Buat kebijakan IAM.

```
aws iam create-policy --policy-name my-policy --policy-document file://my-policy.json
```

2. Buat peran IAM dan kaitkan dengan akun Kubernetes layanan.

   1. Jika Anda memiliki akun Kubernetes layanan yang sudah ada yang ingin Anda ambil peran IAM, maka Anda dapat melewati langkah ini.

Buat akun Kubernetes layanan. Salin konten berikut ke perangkat Anda. Ganti *my-service-account*dengan nama yang Anda inginkan dan *default* dengan namespace yang berbeda, jika perlu. Jika Anda mengubah *default*, namespace harus sudah ada.

```
cat >my-service-account.yaml <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  name: my-service-account
  namespace: default
EOF
kubectl apply -f my-service-account.yaml
```

Jalankan perintah berikut.

```
kubectl apply -f my-service-account.yaml
```

2. Jalankan perintah berikut untuk membuat file kebijakan kepercayaan untuk peran IAM.

```
cat >trust-relationship.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
EOF
```

3. Buat peran. Ganti *my-role* dengan nama untuk peran IAM Anda, dan *my-role-description* dengan deskripsi untuk peran Anda.

```
aws iam create-role --role-name my-role --assume-role-policy-document
 file://trust-relationship.json --description "my-role-description"
```

4. Lampirkan kebijakan IAM ke peran Anda. Ganti *my-role* dengan nama peran IAM Anda dan *my-policy* dengan nama kebijakan yang sudah ada yang Anda buat.

```
aws iam attach-role-policy --role-name my-role --policy-
arn=arn:aws:iam::111122223333:policy/my-policy
```

> **Note**
>
> Tidak seperti peran IAM untuk akun layanan, EKS Pod Identity tidak menggunakan anotasi pada akun layanan.

5. Jalankan perintah berikut untuk membuat asosiasi. Ganti `my-cluster` dengan nama cluster, ganti *my-service-account*dengan nama yang Anda inginkan dan *default* dengan namespace yang berbeda, jika perlu.

```
aws eks create-pod-identity-association --cluster-name my-cluster --role-
arn arn:aws:iam::111122223333:role/my-role --namespace default --service-
account my-service-account
```

Contoh output adalah sebagai berikut.

```
{
    "association": {
        "clusterName": "my-cluster",
        "namespace": "default",
        "serviceAccount": "my-service-account",
        "roleArn": "arn:aws:iam::111122223333:role/my-role",
        "associationArn": "arn:aws::111122223333:podidentityassociation/my-
cluster/a-abcdefghijklmnop1",
        "associationId": "a-abcdefghijklmnop1",
        "tags": {},
        "createdAt": 1700862734.922,
        "modifiedAt": 1700862734.922
    }
}
```

> **Note**
>
> Anda dapat menentukan namespace dan akun layanan berdasarkan nama yang tidak ada di cluster. Anda harus membuat namespace, akun layanan, dan beban kerja yang menggunakan akun layanan agar asosiasi Identitas Pod EKS berfungsi.

3. Konfirmasikan bahwa akun peran dan layanan dikonfigurasi dengan benar.

   a. Konfirmasikan bahwa kebijakan kepercayaan peran IAM telah dikonfigurasi dengan benar.

   ```
   aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
   ```

   Contoh output adalah sebagai berikut.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Sid": "Allow EKS Auth service to assume this role for Pod
    Identities",
               "Effect": "Allow",
               "Principal": {
                   "Service": "pods.eks.amazonaws.com"
               },
               "Action": [
                   "sts:AssumeRole",
                   "sts:TagSession"
               ]
           }
       ]
   }
   ```

   b. Konfirmasikan bahwa kebijakan yang Anda lampirkan pada peran Anda di langkah sebelumnya dilampirkan pada peran tersebut.

   ```
   aws iam list-attached-role-policies --role-name my-role --query
    AttachedPolicies[].PolicyArn --output text
   ```

   Contoh output adalah sebagai berikut.

```
arn:aws:iam::111122223333:policy/my-policy
```

c. Tetapkan variabel untuk menyimpan Nama Sumber Daya Amazon (ARN) kebijakan yang ingin Anda gunakan. *Ganti kebijakan* saya dengan nama kebijakan yang ingin Anda konfirmasi izin.

```
export policy_arn=arn:aws:iam::111122223333:policy/my-policy
```

d. Lihat versi default kebijakan.

```
aws iam get-policy --policy-arn $policy_arn
```

Contoh output adalah sebagai berikut.

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws:iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
    }
}
```

e. Lihat konten kebijakan untuk memastikan bahwa kebijakan tersebut mencakup semua izin yang Anda Pod butuhkan. Jika perlu, ganti *1* dalam perintah berikut dengan versi yang dikembalikan pada output sebelumnya.

```
aws iam get-policy-version --policy-arn $policy_arn --version-id v1
```

Contoh output adalah sebagai berikut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
```

```
        }
    ]
}
```

Jika Anda membuat kebijakan contoh di langkah sebelumnya, maka output Anda sama. Jika Anda membuat kebijakan yang berbeda, maka *contoh* konten berbeda.

Langkah selanjutnya

PodsKonfigurasikan untuk menggunakan akun Kubernetes layanan

PodsKonfigurasikan untuk menggunakan akun Kubernetes layanan

Jika Pod perlu mengakses Layanan AWS, maka Anda harus mengkonfigurasinya untuk menggunakan akun Kubernetes layanan. Akun layanan harus dikaitkan dengan peran AWS Identity and Access Management (IAM) yang memiliki izin untuk mengakses. Layanan AWS

Prasyarat

- Sebuah klaster yang sudah ada. Jika Anda tidak memilikinya, Anda dapat membuatnya menggunakan salah satu panduan Memulai dengan Amazon EKS.
- Akun Kubernetes layanan yang ada dan asosiasi Identitas Pod EKS yang mengaitkan akun layanan dengan peran IAM. Peran harus memiliki kebijakan IAM terkait yang berisi izin yang Pods ingin Anda gunakan. Layanan AWS Untuk informasi selengkapnya tentang cara membuat akun dan peran layanan, dan mengonfigurasinya, lihatMengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM dengan EKS Pod Identity.
- Versi terbaru dari yang AWS CLI diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Anda dapat memeriksa versi saat ini dengan `aws --version | cut -d / -f2 | cut -d ' ' -f1`. Package manager seperti `yum`apt-get`,`, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan `aws configure` di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal di AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl`

versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.

- `kubectl` `config`File yang ada yang berisi konfigurasi cluster Anda. Untuk membuat `kubectl` `config` file, lihatMembuat atau memperbarui `kubeconfig` file untuk klaster Amazon EKS.

Untuk mengkonfigurasi Pod untuk menggunakan akun layanan

1. Gunakan perintah berikut untuk membuat manifes penerapan yang dapat digunakan Pod untuk mengonfirmasi konfigurasi. Ganti *example values* dengan nilai-nilai milik Anda sendiri.

```
cat >my-deployment.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      serviceAccountName: my-service-account
      containers:
      - name: my-app
        image: public.ecr.aws/nginx/nginx:X.XX
EOF
```

2. Terapkan manifes ke cluster Anda.

```
kubectl apply -f my-deployment.yaml
```

3. Konfirmasikan bahwa variabel lingkungan yang diperlukan ada untuk AndaPod.

   a. Lihat Pods yang digunakan dengan penerapan pada langkah sebelumnya.

```
kubectl get pods | grep my-app
```

   Contoh output adalah sebagai berikut.

```
my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
```

b. Konfirmasikan bahwa Pod memiliki file token akun layanan mount.

```
kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep
 AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:
```

Contoh output adalah sebagai berikut.

```
AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:  /var/run/secrets/
pods.eks.amazonaws.com/serviceaccount/eks-pod-identity-token
```

4. Konfirmasikan bahwa Anda Pods dapat berinteraksi dengan Layanan AWS menggunakan izin yang ditetapkan dalam kebijakan IAM yang dilampirkan pada peran Anda.

> **Note**
>
> Saat Pod menggunakan AWS kredensi dari peran IAM yang terkait dengan akun layanan, SDK AWS CLI atau lainnya dalam container untuk itu Pod menggunakan kredenal yang disediakan oleh peran tersebut. Jika Anda tidak membatasi akses ke kredensial yang diberikan ke peran IAM node Amazon EKS, Pod masih memiliki akses ke kredensi ini. Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

Jika Anda tidak Pods dapat berinteraksi dengan layanan seperti yang Anda harapkan, selesaikan langkah-langkah berikut untuk mengonfirmasi bahwa semuanya telah dikonfigurasi dengan benar.

a. Konfirmasikan bahwa Anda Pods menggunakan versi AWS SDK yang mendukung asumsi peran IAM melalui asosiasi Identitas Pod EKS. Untuk informasi selengkapnya, lihat Menggunakan AWS SDK yang didukung.

b. Konfirmasikan bahwa penyebaran menggunakan akun layanan.

```
kubectl describe deployment my-app | grep "Service Account"
```

Contoh output adalah sebagai berikut.

```
Service Account:  my-service-account
```

Menentukan izin untuk Identitas Pod EKS untuk mengambil peran berdasarkan tag

EKS Pod Identity melampirkan tag ke kredensial sementara untuk setiap pod dengan atribut seperti nama cluster, namespace, nama akun layanan. Tag sesi peran ini memungkinkan administrator untuk membuat peran tunggal yang dapat bekerja di seluruh akun layanan dengan mengizinkan akses ke AWS sumber daya berdasarkan tag yang cocok. Dengan menambahkan dukungan untuk tag sesi peran, pelanggan dapat menegakkan batasan keamanan yang lebih ketat antara cluster, dan beban kerja dalam cluster, sambil menggunakan kembali peran IAM dan kebijakan IAM yang sama.

Misalnya, kebijakan berikut memungkinkan `s3:GetObject` tindakan jika objek ditandai dengan nama kluster EKS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectTagging"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:ExistingObjectTag/eks-cluster-name": "${aws:PrincipalTag/eks-
cluster-name}"
                }
            }
        }
    ]
}
```

Daftar tag sesi yang ditambahkan oleh EKS Pod Identity

Daftar berikut berisi semua kunci untuk tag yang ditambahkan ke `AssumeRole` permintaan yang dibuat oleh Amazon EKS. Untuk menggunakan tag ini dalam kebijakan, gunakan `${aws:PrincipalTag/` diikuti oleh kunci, misalnya`${aws:PrincipalTag/kubernetes-namespace}`.

- `eks-cluster-arn`
- `eks-cluster-name`
- `kubernetes-namespace`
- `kubernetes-service-account`
- `kubernetes-pod-name`
- `kubernetes-pod-uid`

Tag lintas-akun

Semua tag sesi yang ditambahkan oleh EKS Pod Identity bersifat transitif; kunci dan nilai tag diteruskan ke `AssumeRole` tindakan apa pun yang digunakan beban kerja Anda untuk beralih peran ke akun lain. Anda dapat menggunakan tag ini dalam kebijakan di akun lain untuk membatasi akses dalam skenario lintas akun. Untuk informasi selengkapnya, lihat Merantai peran dengan tag sesi di Panduan Pengguna IAM.

Tag kustom

EKS Pod Identity tidak dapat menambahkan tag kustom tambahan ke `AssumeRole` tindakan yang dilakukannya. Namun, tag yang Anda terapkan ke peran IAM selalu tersedia meskipun format yang sama: `${aws:PrincipalTag/` diikuti oleh kunci, misalnya`${aws:PrincipalTag/MyCustomTag}`.

> **Note**
>
> Tag yang ditambahkan ke sesi melalui `sts:AssumeRole` permintaan diutamakan dalam kasus konflik. Misalnya, asumsikan bahwa Amazon EKS menambahkan kunci `eks-cluster-name` dan nilai `my-cluster` ke sesi ketika EKS mengambil peran pelanggan. Anda juga telah menambahkan `eks-cluster-name` tag ke peran IAM dengan nilai`my-own-cluster`. Dalam hal ini, yang pertama diutamakan dan nilai untuk `eks-cluster-name` tag akan menjadi. `my-cluster`

Menggunakan AWS SDK yang didukung

> **Important**
>
> Versi dokumentasi sebelumnya tidak benar. AWS SDK for Java v1 tidak mendukung EKS Pod Identity.

Saat menggunakanIdentitas EKS Pod, kontainer di dalam Anda Pods harus menggunakan versi AWS SDK yang mendukung asumsi peran IAM dari Agen Identitas Pod EKS. Pastikan Anda menggunakan versi berikut, atau yang lebih baru, untuk AWS SDK Anda:

- Java (Versi 2) - 2.21.30
- Pergi v1 - v1.47.11
- Go v2 - rilis-2023-11-14
- Python (Boto3) — 1.34.41
- Python (botocore) - 1.32.0
- AWS CLI — 1.30.0

  AWS CLI — 2.15.0
- JavaScript v2 — 2.1550.0
- JavaScript v3 — 3.458.0
- Ruby — 3.188.0
- C++ — 1.11.263
- .NET — 3.7.734.0 —
- PHP — 3.287.1

Untuk memastikan bahwa Anda menggunakan SDK yang didukung, ikuti petunjuk penginstalan untuk SDK pilihan Anda di Tools to Build on AWS saat Anda membuat container.

Menggunakan kredensial Identitas Pod EKS

Untuk menggunakan kredensi dari asosiasi Identitas Pod EKS, kode Anda dapat menggunakan AWS SDK apa pun untuk membuat klien untuk AWS layanan dengan SDK, dan secara default SDK akan mencari di rantai lokasi untuk kredensi yang akan digunakan. AWS Identity and Access Management

Kredensial Identitas Pod EKS akan digunakan jika Anda tidak menentukan penyedia kredensi saat Anda membuat klien atau menginisialisasi SDK.

Ini berfungsi karena EKS Pod Identities telah ditambahkan ke penyedia kredensi Container yang dicari dalam satu langkah dalam rantai kredensi default. Jika beban kerja Anda saat ini menggunakan kredensial yang sebelumnya berada dalam rantai kredensial, kredensial-kredensialnya akan terus digunakan meskipun Anda mengonfigurasi asosiasi Identitas Pod EKS untuk beban kerja yang sama.

Untuk informasi selengkapnya tentang cara kerja Identitas Pod EKS, lihatCara kerja EKS Pod Identity.

Peran Identitas EKS Pod

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

`sts:AssumeRole`

EKS Pod Identity menggunakan `TagSession` untuk mengambil peran IAM sebelum meneruskan kredensialnya sementara ke pod Anda.

`sts:TagSession`

EKS Pod Identity menggunakan `TagSession` untuk menyertakan tag sesi dalam permintaan untukAWS STS.

Anda dapat menggunakan tag ini dalam kebijakan kepercayaan untuk membatasi akun layanan, ruang nama, dan klaster mana yang dapat menggunakan peran ini. condition keys

Untuk daftar kunci kondisi Amazon EKS, lihat Ketentuan yang ditentukan oleh Amazon Elastic Kubernetes Service di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat digunakan untuk menggunakan kunci kondisi, lihat Tindakan yang ditentukan oleh Amazon Elastic Kubernetes Service.

IAM role untuk akun layanan

Aplikasi dalam Pod kontainer dapat menggunakan AWS SDK atau AWS CLI untuk membuat permintaan API untuk Layanan AWS menggunakan izin AWS Identity and Access Management (IAM). Aplikasi harus menandatangani permintaan AWS API mereka dengan AWS kredensialnya. Peran IAM untuk akun layanan menyediakan kemampuan untuk mengelola kredensional untuk aplikasi Anda, mirip dengan cara profil instans Amazon EC2 memberikan kredensil ke instans Amazon EC2. Alih-alih membuat dan mendistribusikan AWS kredensil Anda ke container atau menggunakan peran instans Amazon EC2, Anda mengaitkan peran IAM dengan akun layanan dan mengonfigurasi Anda Pods untuk menggunakan akun Kubernetes layanan. Anda tidak dapat menggunakan peran IAM untuk akun layanan dengan kluster lokal untuk Amazon EKS aktif. AWS Outposts

Peran IAM untuk akun layanan memberikan manfaat berikut:

- Keistimewaan paling sedikit — Anda dapat mencakup izin IAM ke akun layanan, dan hanya akun layanan Pods yang menggunakan akun layanan tersebut yang memiliki akses ke izin tersebut. Fitur ini juga mengurangi kebutuhan akan solusi dari pihak ketiga seperti `kiam` atau `kube2iam`.
- Isolasi kredenal — Pod's Kontainer hanya dapat mengambil kredensil untuk peran IAM yang terkait dengan akun layanan yang digunakan penampung. Kontainer tidak pernah memiliki akses ke kredensil yang digunakan oleh kontainer lain di wadah lain. Pods Saat menggunakan peran IAM untuk akun layanan, Pod's container juga memiliki izin yang ditetapkan ke peran IAM node Amazon EKS, kecuali Anda memblokir Pod akses ke Layanan Metadata Instans Amazon EC2 (IMDS). Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.
- Auditabilitas — Akses dan pencatatan peristiwa tersedia AWS CloudTrail untuk membantu memastikan audit retrospektif.

Aktifkan peran IAM untuk akun layanan dengan menyelesaikan prosedur berikut:

1. Buat OIDC penyedia IAM untuk klaster Anda— Anda hanya menyelesaikan prosedur ini sekali untuk setiap cluster.

> **Note**
>
> Jika Anda mengaktifkan titik akhir EKS VPC, titik akhir layanan EKS OIDC tidak dapat diakses dari dalam VPC itu. Akibatnya, operasi Anda seperti membuat penyedia OIDC dengan `eksctl` VPC tidak akan berfungsi dan akan menghasilkan batas waktu saat mencoba meminta. `https://oidc.eks.region.amazonaws.com` Contoh pesan kesalahan berikut:
>
> ```
> ** server can't find oidc.eks.region.amazonaws.com: NXDOMAIN
> ```
>
> Untuk menyelesaikan langkah ini, Anda dapat menjalankan perintah di luar VPC, misalnya di dalam AWS CloudShell atau di komputer yang terhubung ke internet.

2. Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM— Selesaikan prosedur ini untuk setiap set izin unik yang Anda inginkan untuk dimiliki aplikasi.
3. PodsKonfigurasikan untuk menggunakan akun Kubernetes layanan— Selesaikan prosedur ini untuk masing-masing Pod yang membutuhkan akses Layanan AWS.
4. Menggunakan AWS SDK yang didukung— Konfirmasikan bahwa beban kerja menggunakan AWS SDK versi yang didukung dan bahwa beban kerja menggunakan rantai kredenal default.

IAM,Kubernetes, dan OpenID Connect (OIDC) informasi latar belakang

Pada tahun 2014, AWS Identity and Access Management menambahkan dukungan untuk identitas federasi menggunakan OpenID Connect ()OIDC. Fitur ini memungkinkan Anda untuk mengautentikasi panggilan AWS API dengan penyedia identitas yang didukung dan menerima token OIDC JSON web yang valid (JWT). Anda dapat meneruskan token ini ke operasi AWS STS `AssumeRoleWithWebIdentity` API dan menerima kredenal peran sementara IAM. Anda dapat menggunakan kredensil ini untuk berinteraksi dengan manapun Layanan AWS, termasuk Amazon S3 dan DynamoDB.

Setiap token JWT ditandatangani oleh tanda tangan key pair. Kunci disajikan pada penyedia OIDC yang dikelola oleh Amazon EKS dan kunci pribadi berputar setiap 7 hari. Amazon EKS menyimpan kunci publik sampai kedaluwarsa. Jika Anda menghubungkan klien OIDC eksternal, ketahuilah bahwa Anda perlu menyegarkan kunci penandatanganan sebelum kunci publik kedaluwarsa. Pelajari cara the section called “Ambil kunci penandatanganan”.

Kubernetestelah lama menggunakan akun layanan sebagai sistem identitas internalnya sendiri. Podsdapat mengautentikasi dengan server Kubernetes API menggunakan token yang dipasang secara otomatis (yang bukan- OIDCJWT) yang hanya dapat divalidasi oleh server Kubernetes API. Token akun layanan lama ini tidak kedaluwarsa, dan memutar kunci penandatanganan adalah proses yang sulit. Dalam Kubernetes versi`1.12`, dukungan ditambahkan untuk `ProjectedServiceAccountToken` fitur baru. Fitur ini adalah token OIDC JSON web yang juga berisi identitas akun layanan dan mendukung audiens yang dapat dikonfigurasi.

Amazon EKS menghosting titik akhir OIDC penemuan publik untuk setiap cluster yang berisi kunci penandatanganan untuk token `ProjectedServiceAccountToken` JSON web sehingga sistem eksternal, seperti IAM, dapat memvalidasi dan menerima OIDC token yang dikeluarkan oleh. Kubernetes

Buat OIDC penyedia IAM untuk klaster Anda

Cluster Anda memiliki URL penerbit OpenID Connect(OIDC) yang terkait dengannya. Untuk menggunakan peran AWS Identity and Access Management (IAM) untuk akun layanan, OIDC penyedia IAM harus ada untuk URL OIDC penerbit klaster Anda.

Prasyarat

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS.
- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`,`apt-get`, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru dari file AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl` versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.

- `kubectlconfig`File yang sudah ada yang berisi konfigurasi cluster Anda. Untuk membuat `kubectl config` file, lihatMembuat atau memperbarui `kubeconfig` file untuk klaster Amazon EKS.

Anda dapat membuat OIDC penyedia IAM untuk cluster Anda menggunakan `eksctl` atau. AWS Management Console

eksctl

Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbaruui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Untuk membuat penyedia OIDC identitas IAM untuk cluster Anda dengan **`eksctl`**

1. Tentukan ID OIDC penerbit untuk klaster Anda.

   Ambil ID OIDC penerbit klaster Anda dan simpan dalam variabel. Ganti *`my-cluster`* dengan nilai Anda sendiri.

   ```
   cluster_name=my-cluster
   ```

   ```
   oidc_id=$(aws eks describe-cluster --name $cluster_name --query
    "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
   ```

   ```
   echo $oidc_id
   ```

2. Tentukan apakah OIDC penyedia IAM dengan ID penerbit klaster Anda sudah ada di akun Anda.

   ```
   aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
   ```

   Jika output dikembalikan, maka Anda sudah memiliki OIDC penyedia IAM untuk cluster Anda dan Anda dapat melewati langkah berikutnya. Jika tidak ada output yang dikembalikan, maka Anda harus membuat OIDC penyedia IAM untuk cluster Anda.

3. Buat penyedia OIDC identitas IAM untuk klaster Anda dengan perintah berikut.

```
eksctl utils associate-iam-oidc-provider --cluster $cluster_name --approve
```

> **Note**
>
> Jika Anda mengaktifkan titik akhir EKS VPC, titik akhir layanan EKS OIDC tidak dapat diakses dari dalam VPC itu. Akibatnya, operasi Anda seperti membuat penyedia OIDC dengan `eksctl` VPC tidak akan berfungsi dan akan menghasilkan batas waktu saat mencoba meminta. `https://oidc.eks.region.amazonaws.com` Contoh pesan kesalahan berikut:
>
> ```
> ** server can't find oidc.eks.region.amazonaws.com: NXDOMAIN
> ```
>
> Untuk menyelesaikan langkah ini, Anda dapat menjalankan perintah di luar VPC, misalnya di dalam AWS CloudShell atau di komputer yang terhubung ke internet.

AWS Management Console

Untuk membuat penyedia OIDC identitas IAM untuk cluster Anda dengan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel kiri, pilih Cluster, lalu pilih nama cluster Anda di halaman Clusters.
3. Di bagian Detail pada tab Ikhtisar, perhatikan nilai URL penyedia OpenID Connect.
4. Buka konsol IAM di https://console.aws.amazon.com/iam/.
5. Di panel navigasi kiri, pilih Penyedia Identitas di bawah Manajemen akses. Jika Penyedia tercantum dalam daftar yang cocok dengan URL untuk klaster Anda, maka Anda sudah memiliki penyedia untuk klaster Anda. Jika penyedia tidak tercantum dalam daftar yang cocok dengan URL untuk klaster Anda, maka Anda harus membuatnya.
6. Untuk membuat penyedia, pilih Tambah penyedia.
7. Untuk jenis Penyedia, pilih OpenID Connect.
8. Untuk URL Penyedia, masukkan URL OIDC penyedia untuk klaster Anda, lalu pilih Dapatkan cap jempol.
9. Untuk Audiens, masukkan **`sts.amazonaws.com`** dan pilih Tambahkan penyedia.

Langkah selanjutnya

Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM

Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM

Topik ini mencakup cara mengonfigurasi akun Kubernetes layanan untuk mengambil peran AWS Identity and Access Management (IAM). Apa pun Pods yang dikonfigurasi untuk menggunakan akun layanan kemudian dapat mengakses apa pun Layanan AWS yang peran memiliki izin untuk diakses.

Prasyarat

- Sebuah klaster yang sudah ada. Jika Anda tidak memilikinya, Anda dapat membuatnya dengan mengikuti salah satu Memulai dengan Amazon EKS panduan.
- Penyedia IAM OpenID Connect (OIDC) yang ada untuk cluster Anda. Untuk mengetahui apakah Anda sudah memilikinya atau cara membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda.
- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru dari file AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl` versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.
- `kubectlconfig`File yang sudah ada yang berisi konfigurasi cluster Anda. Untuk membuat `kubectl config` file, lihatMembuat atau memperbarui `kubeconfig` file untuk klaster Amazon EKS.

Untuk mengaitkan peran IAM dengan akun Kubernetes layanan

1. Jika Anda ingin mengaitkan kebijakan IAM yang ada dengan peran IAM Anda, lewati ke langkah berikutnya.

Buat kebijakan IAM. Anda dapat membuat kebijakan sendiri, atau menyalin kebijakan AWS terkelola yang telah memberikan beberapa izin yang Anda perlukan dan menyesuaikannya dengan persyaratan spesifik Anda. Untuk informasi selengkapnya, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

a. Buat file yang menyertakan izin untuk Layanan AWS yang ingin Anda Pods akses. Untuk daftar semua tindakan untuk semua Layanan AWS, lihat Referensi Otorisasi Layanan.

   Anda dapat menjalankan perintah berikut untuk membuat contoh file kebijakan yang memungkinkan akses hanya-baca ke bucket Amazon S3. Anda dapat secara opsional menyimpan informasi konfigurasi atau skrip bootstrap di bucket ini, dan kontainer di dalam Anda Pod dapat membaca file dari bucket dan memuatnya ke dalam aplikasi Anda. Jika Anda ingin membuat kebijakan contoh ini, salin konten berikut ke perangkat Anda. Ganti *my-pod-secrets-bucket* dengan nama bucket Anda dan jalankan perintah.

```
cat >my-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
EOF
```

b. Buat kebijakan IAM.

```
aws iam create-policy --policy-name my-policy --policy-document file://my-policy.json
```

2. Buat peran IAM dan kaitkan dengan akun Kubernetes layanan. Anda dapat menggunakan `eksctl` atau AWS CLI.

   eksctl

   Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Ganti *`my-service-account`*dengan nama akun Kubernetes layanan yang `eksctl` ingin Anda buat dan kaitkan dengan peran IAM. Ganti *`default`* dengan namespace tempat Anda `eksctl` ingin membuat akun layanan. Ganti *`my-cluster`* dengan nama klaster Anda. *`Ganti peran`* saya dengan nama peran yang ingin Anda kaitkan dengan akun layanan. Jika belum ada, `eksctl` buatlah untuk Anda. Ganti *`111122223333`* dengan ID akun dan kebijakan saya dengan nama *`kebijakan yang`* ada.

```
eksctl create iamserviceaccount --name my-service-account --namespace default --
cluster my-cluster --role-name my-role \
    --attach-policy-arn arn:aws:iam::111122223333:policy/my-policy --approve
```

> **Important**
>
> Jika akun peran atau layanan sudah ada, perintah sebelumnya mungkin gagal. `eksctl`memiliki opsi berbeda yang dapat Anda berikan dalam situasi tersebut. Untuk informasi lebih lanjut jalankan**`eksctl create iamserviceaccount --help`**.

AWS CLI

1. Jika Anda memiliki akun Kubernetes layanan yang sudah ada yang ingin Anda ambil peran IAM, maka Anda dapat melewati langkah ini.

   Buat akun Kubernetes layanan. Salin konten berikut ke perangkat Anda. Ganti *`my-service-account`*dengan nama yang Anda inginkan dan *`default`* dengan namespace yang berbeda, jika perlu. Jika Anda mengubah *`default`*, namespace harus sudah ada.

   ```
   cat >my-service-account.yaml <<EOF
   apiVersion: v1
   kind: ServiceAccount
   metadata:
     name: my-service-account
     namespace: default
   EOF
   ```

```
kubectl apply -f my-service-account.yaml
```

2. Atur Akun AWS ID Anda ke variabel lingkungan dengan perintah berikut.

```
account_id=$(aws sts get-caller-identity --query "Account" --output text)
```

3. Tetapkan penyedia OIDC identitas cluster Anda ke variabel lingkungan dengan perintah berikut. Ganti *my-cluster* dengan nama klaster Anda.

```
oidc_provider=$(aws eks describe-cluster --name my-cluster --region
 $AWS_REGION --query "cluster.identity.oidc.issuer" --output text | sed -e "s/
^https:\/\///")
```

4. Tetapkan variabel untuk namespace dan nama akun layanan. Ganti *my-service-account* dengan akun Kubernetes layanan yang ingin Anda ambil peran. Ganti *default* dengan namespace akun layanan.

```
export namespace=default
export service_account=my-service-account
```

5. Jalankan perintah berikut untuk membuat file kebijakan kepercayaan untuk peran IAM. Jika Anda ingin mengizinkan semua akun layanan dalam namespace untuk menggunakan peran tersebut, salin konten berikut ke perangkat Anda. Ganti *StringEquals*dengan **StringLike** dan ganti *$service_account* dengan. * Anda dapat menambahkan beberapa entri dalam `StringLike` kondisi `StringEquals` atau untuk memungkinkan beberapa akun layanan atau ruang nama mengambil peran tersebut. Untuk mengizinkan peran dari akun yang berbeda Akun AWS dari akun yang digunakan klaster Anda untuk mengambil peran, lihat Izin IAM lintas akun untuk informasi selengkapnya.

```
cat >trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::$account_id:oidc-provider/$oidc_provider"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
```

```
          "$oidc_provider:aud": "sts.amazonaws.com",
          "$oidc_provider:sub": "system:serviceaccount:
$namespace:$service_account"
        }
      }
    }
  ]
}
EOF
```

6. Buat peran. Ganti *my-role* dengan nama untuk peran IAM Anda, dan *my-role-description* dengan deskripsi untuk peran Anda.

```
aws iam create-role --role-name my-role --assume-role-policy-document
 file://trust-relationship.json --description "my-role-description"
```

7. Lampirkan kebijakan IAM ke peran Anda. Ganti *my-role* dengan nama peran IAM Anda dan *my-policy* dengan nama kebijakan yang sudah ada yang Anda buat.

```
aws iam attach-role-policy --role-name my-role --policy-arn=arn:aws:iam::
$account_id:policy/my-policy
```

8. Beri anotasi akun layanan Anda dengan Nama Sumber Daya Amazon (ARN) dari peran IAM yang Anda inginkan untuk diasumsikan oleh akun layanan. Ganti *my-role* dengan nama peran IAM Anda yang ada. Misalkan Anda mengizinkan peran dari akun yang berbeda Akun AWS dari akun yang digunakan klaster Anda untuk mengambil peran pada langkah sebelumnya. Kemudian, pastikan untuk menentukan Akun AWS dan peran dari akun lain. Untuk informasi selengkapnya, lihat Izin IAM lintas akun.

```
kubectl annotate serviceaccount -n $namespace $service_account
 eks.amazonaws.com/role-arn=arn:aws:iam::$account_id:role/my-role
```

3. Konfirmasikan bahwa akun peran dan layanan dikonfigurasi dengan benar.

   a. Konfirmasikan bahwa kebijakan kepercayaan peran IAM telah dikonfigurasi dengan benar.

```
aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
```

   Contoh output adalah sebagai berikut.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:default:my-
service-account",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
                }
            }
        }
    ]
}
```

b. Konfirmasikan bahwa kebijakan yang Anda lampirkan pada peran Anda di langkah sebelumnya dilampirkan pada peran tersebut.

```
aws iam list-attached-role-policies --role-name my-role --query
 AttachedPolicies[].PolicyArn --output text
```

Contoh output adalah sebagai berikut.

```
arn:aws:iam::111122223333:policy/my-policy
```

c. Tetapkan variabel untuk menyimpan Nama Sumber Daya Amazon (ARN) kebijakan yang ingin Anda gunakan. *Ganti kebijakan* saya dengan nama kebijakan yang ingin Anda konfirmasi izin.

```
export policy_arn=arn:aws:iam::111122223333:policy/my-policy
```

d. Lihat versi default kebijakan.

```
aws iam get-policy --policy-arn $policy_arn
```

Contoh output adalah sebagai berikut.

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws:iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
    }
}
```

e. Lihat konten kebijakan untuk memastikan bahwa kebijakan tersebut mencakup semua izin yang Anda Pod butuhkan. Jika perlu, ganti *1* dalam perintah berikut dengan versi yang dikembalikan pada output sebelumnya.

```
aws iam get-policy-version --policy-arn $policy_arn --version-id v1
```

Contoh output adalah sebagai berikut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

Jika Anda membuat kebijakan contoh di langkah sebelumnya, maka output Anda sama. Jika Anda membuat kebijakan yang berbeda, maka *contoh* konten berbeda.

f. Konfirmasikan bahwa akun Kubernetes layanan dianotasi dengan peran tersebut.

```
kubectl describe serviceaccount my-service-account -n default
```

Contoh output adalah sebagai berikut.

```
Name:                my-service-account
Namespace:           default
Annotations:         eks.amazonaws.com/role-arn:
 arn:aws:iam::111122223333:role/my-role
Image pull secrets:  <none>
Mountable secrets:   my-service-account-token-qqjfl
Tokens:              my-service-account-token-qqjfl
[...]
```

4. (Opsional)Konfigurasikan AWS Security Token Service titik akhir untuk akun layanan. AWS merekomendasikan menggunakan titik AWS STS akhir regional alih-alih titik akhir global. Ini mengurangi latensi, menyediakan redundansi bawaan, dan meningkatkan validitas token sesi.

Langkah selanjutnya

PodsKonfigurasikan untuk menggunakan akun Kubernetes layanan

PodsKonfigurasikan untuk menggunakan akun Kubernetes layanan

Jika Pod perlu mengakses Layanan AWS, maka Anda harus mengkonfigurasinya untuk menggunakan akun Kubernetes layanan. Akun layanan harus dikaitkan dengan peran AWS Identity and Access Management (IAM) yang memiliki izin untuk mengakses. Layanan AWS

Prasyarat

- Sebuah klaster yang sudah ada. Jika Anda tidak memilikinya, Anda dapat membuatnya menggunakan salah satu panduan Memulai dengan Amazon EKS.
- Penyedia IAM OpenID Connect (OIDC) yang ada untuk cluster Anda. Untuk mengetahui apakah Anda sudah memilikinya atau cara membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda.
- Akun Kubernetes layanan yang ada yang terkait dengan peran IAM. Akun layanan harus dianotasi dengan Nama Sumber Daya Amazon (ARN) dari peran IAM. Peran harus memiliki kebijakan IAM terkait yang berisi izin yang Pods ingin Anda gunakan. Layanan AWS Untuk informasi selengkapnya tentang cara membuat akun dan peran layanan, dan mengonfigurasinya, lihatMengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM.
- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get`,, atau Homebrew untuk macOS sering

beberapa versi di belakang versi terbaru dari file AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.

- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl` versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.
- `kubectl``config`File yang sudah ada yang berisi konfigurasi cluster Anda. Untuk membuat `kubectl config` file, lihatMembuat atau memperbarui `kubeconfig` file untuk klaster Amazon EKS.

Untuk mengkonfigurasi Pod untuk menggunakan akun layanan

1. Gunakan perintah berikut untuk membuat manifes penerapan yang dapat digunakan Pod untuk mengonfirmasi konfigurasi. Ganti *`example values`* dengan nilai-nilai milik Anda sendiri.

```
cat >my-deployment.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      serviceAccountName: my-service-account
      containers:
      - name: my-app
        image: public.ecr.aws/nginx/nginx:X.XX
EOF
```

2. Terapkan manifes ke cluster Anda.

```
kubectl apply -f my-deployment.yaml
```

3. Konfirmasikan bahwa variabel lingkungan yang diperlukan ada untuk AndaPod.

   a. Lihat Pods yang digunakan dengan penerapan pada langkah sebelumnya.

   ```
   kubectl get pods | grep my-app
   ```

   Contoh output adalah sebagai berikut.

   ```
   my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
   ```

   b. Lihat ARN dari peran IAM yang digunakan. Pod

   ```
   kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep AWS_ROLE_ARN:
   ```

   Contoh output adalah sebagai berikut.

   ```
   AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/my-role
   ```

   Peran ARN harus cocok dengan ARN peran yang Anda anotasi dengan akun layanan yang ada. Untuk informasi selengkapnya tentang membuat anotasi akun layanan, lihat. Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM

   c. Konfirmasikan bahwa Pod memiliki mount file token identitas web.

   ```
   kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep
    AWS_WEB_IDENTITY_TOKEN_FILE:
   ```

   Contoh output adalah sebagai berikut.

   ```
   AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
   serviceaccount/token
   ```

   `kubelet`Meminta dan menyimpan token atas namaPod. Secara default, token akan `kubelet` menyegarkan jika token lebih tua dari 80 persen dari total waktu untuk hidup atau lebih dari 24 jam. Anda dapat mengubah durasi kedaluwarsa untuk akun apa pun selain akun layanan default dengan menggunakan pengaturan dalam spesifikasi AndaPod. Untuk

informasi selengkapnya, lihat Proyeksi Volume Token Akun Layanan dalam Kubernetes dokumentasi.

Amazon EKS Pod Identity Webhook pada kluster Pods mengawasinya menggunakan akun layanan dengan anotasi berikut:

```
eks.amazonaws.com/role-arn: arn:aws:iam::111122223333:role/my-role
```

Webhook menerapkan variabel lingkungan sebelumnya ke variabel tersebutPods. Cluster Anda tidak perlu menggunakan webhook untuk mengonfigurasi variabel lingkungan dan pemasangan file token. Anda dapat mengonfigurasi secara manual Pods untuk memiliki variabel lingkungan ini. Versi AWS SDK yang didukung mencari variabel lingkungan ini terlebih dahulu di penyedia rantai kredenal. Kredensi peran digunakan untuk Pods yang memenuhi kriteria ini.

4. Konfirmasikan bahwa Anda Pods dapat berinteraksi dengan Layanan AWS menggunakan izin yang ditetapkan dalam kebijakan IAM yang dilampirkan pada peran Anda.

> **Note**
>
> Saat Pod menggunakan AWS kredensi dari peran IAM yang terkait dengan akun layanan, SDK AWS CLI atau lainnya dalam container untuk itu Pod menggunakan kredenal yang disediakan oleh peran tersebut. Jika Anda tidak membatasi akses ke kredensial yang diberikan ke peran IAM node Amazon EKS, Pod masih memiliki akses ke kredenal ini. Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

Jika Anda tidak Pods dapat berinteraksi dengan layanan seperti yang Anda harapkan, selesaikan langkah-langkah berikut untuk mengonfirmasi bahwa semuanya telah dikonfigurasi dengan benar.

a. Konfirmasikan bahwa Anda Pods menggunakan versi AWS SDK yang mendukung asumsi peran IAM melalui file token identitas OpenID Connect web. Untuk informasi selengkapnya, lihat Menggunakan AWS SDK yang didukung.

b. Konfirmasikan bahwa penyebaran menggunakan akun layanan.

```
kubectl describe deployment my-app | grep "Service Account"
```

Contoh output adalah sebagai berikut.

```
Service Account:  my-service-account
```

c. Jika Anda Pods masih tidak dapat mengakses layanan, tinjau langkah-langkah yang dijelaskan Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM untuk mengonfirmasi bahwa akun peran dan layanan Anda telah dikonfigurasi dengan benar.

Konfigurasikan AWS Security Token Service titik akhir untuk akun layanan

Jika Anda menggunakan akun Kubernetes layananIAM role untuk akun layanan, maka Anda dapat mengonfigurasi jenis AWS Security Token Service titik akhir yang digunakan oleh akun layanan jika versi cluster dan platform Anda sama atau lebih lambat dari yang tercantum dalam tabel berikut. Jika versi Anda Kubernetes atau platform lebih awal dari yang tercantum dalam tabel, maka akun layanan Anda hanya dapat menggunakan titik akhir global.

Versi Kubernetes	Versi platform	Jenis titik akhir default
1.29	eks.1	Regional
1.28	eks.1	Regional
1.27	eks.1	Regional
1.26	eks.1	Regional
1.25	eks.1	Regional
1.24	eks.2	Regional
1.23	eks.1	Regional

AWS merekomendasikan penggunaan titik AWS STS akhir regional alih-alih titik akhir global. Ini mengurangi latensi, menyediakan redundansi bawaan, dan meningkatkan validitas token sesi. AWS Security Token Service Harus aktif di Wilayah AWS tempat yang Pod sedang berjalan. Selain itu, aplikasi Anda harus memiliki redundansi bawaan untuk yang berbeda Wilayah AWS jika terjadi kegagalan layanan di. Wilayah AWS Untuk informasi selengkapnya, lihat Mengelola AWS STSWilayah AWS dalam Panduan Pengguna IAM.

Prasyarat

- Sebuah klaster yang sudah ada. Jika Anda sama sekali tidak memilikinya, Anda dapat membuatnya menggunakan salah satu panduan Memulai dengan Amazon EKS.
- Penyedia IAM OIDC yang sudah ada untuk klaster Anda. Untuk informasi selengkapnya, lihat Buat OIDC penyedia IAM untuk klaster Anda.
- Akun Kubernetes layanan yang ada dikonfigurasi untuk digunakan dengan Amazon EKS IAM untuk fitur akun layanan.

Untuk mengonfigurasi tipe titik akhir yang digunakan oleh akun Kubernetes layanan

Contoh berikut semua menggunakan akun `aws-node` Kubernetes layanan yang digunakan oleh plugin Amazon VPC CNI. Anda dapat mengganti *`example values`* dengan akun layanan Anda sendiri, ruang namaPods, dan sumber daya lainnya.

1. Pilih Pod yang menggunakan akun layanan yang ingin Anda ubah endpoint. Tentukan mana Wilayah AWS yang Pod berjalan. Ganti *`aws-node-6mfgv`* dengan Pod nama Anda dan *`kube-system`* dengan Pod namespace Anda.

   ```
   kubectl describe pod aws-node-6mfgv -n kube-system |grep Node:
   ```

   Contoh output adalah sebagai berikut.

   ```
   ip-192-168-79-166.us-west-2/192.168.79.166
   ```

   Wilayah AWS

2. Tentukan jenis titik akhir yang digunakan akun Pod's layanan.

   ```
   kubectl describe pod aws-node-6mfgv -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
   ```

   Contoh output adalah sebagai berikut.

   ```
   AWS_STS_REGIONAL_ENDPOINTS: regional
   ```

   Jika titik akhir saat ini bersifat global, `global` maka dikembalikan dalam output. Jika tidak ada output yang dikembalikan, maka tipe endpoint default sedang digunakan dan belum diganti.

3. Jika versi cluster atau platform Anda sama atau lebih lambat dari yang tercantum dalam tabel, maka Anda dapat mengubah jenis titik akhir yang digunakan oleh akun layanan Anda dari tipe default ke tipe yang berbeda dengan salah satu perintah berikut. Ganti *aws-node* dengan nama akun layanan Anda dan *kube-system* dengan namespace untuk akun layanan Anda.

   - Jika tipe titik akhir default atau saat ini bersifat global dan Anda ingin mengubahnya menjadi regional:

     ```
     kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=true
     ```

     Jika Anda menggunakan IAM role untuk akun layanan untuk membuat URL S3 yang telah ditandatangani sebelumnya dalam aplikasi Anda yang berjalan di Pods 'container, format URL untuk titik akhir regional mirip dengan contoh berikut:

     ```
     https://bucket.s3.us-west-2.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
     ```

   - Jika tipe titik akhir default atau saat ini bersifat regional dan Anda ingin mengubahnya menjadi global:

     ```
     kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=false
     ```

     Jika aplikasi Anda secara eksplisit membuat permintaan ke titik akhir AWS STS global dan Anda tidak mengganti perilaku default menggunakan titik akhir regional di kluster Amazon EKS, permintaan akan gagal dengan kesalahan. Untuk informasi selengkapnya, lihat Kontainer pod menerima kesalahan berikut: `An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region`.

     Jika Anda menggunakan IAM role untuk akun layanan untuk menghasilkan URL S3 yang telah ditandatangani sebelumnya dalam aplikasi Anda yang berjalan di Pods 'container, format URL untuk titik akhir global mirip dengan contoh berikut:

     ```
     https://bucket.s3.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
     ```

Jika Anda memiliki otomatisasi yang mengharapkan URL yang telah ditandatangani sebelumnya dalam format tertentu atau jika aplikasi atau dependensi hilir yang menggunakan URL yang telah ditandatangani sebelumnya memiliki harapan untuk Wilayah AWS target, maka buat perubahan yang diperlukan untuk menggunakan titik akhir yang sesuai. AWS STS

4. Hapus dan buat ulang semua Pods yang ada yang terkait dengan akun layanan untuk menerapkan variabel lingkungan kredensyal. Kait web yang bermutasi tidak menerapkannya pada Pods yang sudah berjalan. Anda dapat mengganti*Pods*,*kube-system*, dan *-l k8s-app=aws-node* dengan informasi untuk Pods yang Anda tetapkan anotasi Anda.

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

5. Konfirmasikan bahwa semua Pods dimulai ulang.

```
kubectl get Pods -n kube-system -l k8s-app=aws-node
```

6. Lihat variabel lingkungan untuk salah satuPods. Verifikasi bahwa `AWS_STS_REGIONAL_ENDPOINTS` nilainya adalah apa yang Anda atur pada langkah sebelumnya.

```
kubectl describe pod aws-node-kzbtr -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
```

Contoh output adalah sebagai berikut.

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

Izin IAM lintas akun

Anda dapat mengonfigurasi izin IAM lintas akun baik dengan membuat penyedia identitas dari klaster akun lain atau dengan menggunakan operasi berantai. `AssumeRole` Dalam contoh berikut, Akun A memiliki klaster Amazon EKS yang mendukung peran IAM untuk akun layanan. Podsyang berjalan di cluster itu harus mengasumsikan izin IAM dari Akun B.

Example Buat penyedia identitas dari klaster akun lain

Example

Dalam contoh ini, Akun A menyediakan Akun B dengan URL penerbit OpenID Connect (OIDC) dari klaster mereka. Akun B mengikuti petunjuk di Buat OIDC penyedia IAM untuk klaster Anda dan Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM menggunakan URL penerbit OIDC dari klaster Akun A. Kemudian, administrator klaster membuat anotasi akun layanan di klaster Akun A untuk menggunakan peran dari Akun B (*444455556666*).

```
apiVersion: v1
kind: ServiceAccount
metadata:
  annotations:
    eks.amazonaws.com/role-arn: arn:aws:iam::444455556666:role/account-b-role
```

Example Gunakan operasi berantai **`AssumeRole`**

Example

Dalam contoh ini, Akun B membuat kebijakan IAM dengan izin untuk diberikan Pods di klaster Akun A. *Akun B (444455556666) melampirkan kebijakan tersebut ke peran IAM dengan hubungan kepercayaan yang memungkinkan AssumeRole izin ke Akun A (111122223333).*

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

Akun A membuat peran dengan kebijakan kepercayaan yang mendapatkan kredensi dari penyedia identitas yang dibuat dengan alamat penerbit OIDC klaster.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.region-
code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity"
    }
  ]
}
```

Akun A melampirkan kebijakan ke peran tersebut untuk mengambil peran yang dibuat oleh Akun B dengan izin berikut.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::444455556666:role/account-b-role"
        }
    ]
}
```

Kode aplikasi Pods untuk mengambil peran Akun B menggunakan dua profil: `account_b_role` dan`account_a_role`. Profil `account_b_role` menggunakan profil `account_a_role` sebagai sumbernya. Untuk AWS CLI, `~/.aws/config` file ini mirip dengan yang berikut ini.

```
[profile account_b_role]
source_profile = account_a_role
role_arn=arn:aws:iam::444455556666:role/account-b-role

[profile account_a_role]
web_identity_token_file = /var/run/secrets/eks.amazonaws.com/serviceaccount/token
role_arn=arn:aws:iam::111122223333:role/account-a-role
```

Untuk menentukan profil berantai untuk AWS SDK lain, lihat dokumentasi SDK yang Anda gunakan. Untuk informasi selengkapnya, lihat Alat untuk Dibangun AWS.

Menggunakan AWS SDK yang didukung

Saat menggunakanIAM role untuk akun layanan, kontainer di dalam Anda Pods harus menggunakan versi AWS SDK yang mendukung asumsi peran IAM melalui file token identitas OpenID Connect web. Pastikan Anda menggunakan versi berikut, atau yang lebih baru, untuk AWS SDK Anda:

- Java (Versi 2) - 2.10.11
- Jawa — 1.11.704
- Pergi — 1.23.13
- Python (Boto3) - 1.9.220
- Python (botocore) - 1.12.200
- AWS CLI — 1.16.232
- Node — 2.525.0 dan 3.27.0
- Ruby — 3.58.0
- C++ — 1.7.174
- .NET — 3.3.659.1 — Anda juga harus menyertakan. `AWSSDK.SecurityToken`
- PHP — 3.110.7

Banyak Kubernetes add-on populer, seperti Cluster Autoscaler, theApa itu AWS Load Balancer Controller?, dan Amazon VPC CNI plugin for Kubernetesmendukung peran IAM untuk akun layanan.

Untuk memastikan bahwa Anda menggunakan SDK yang didukung, ikuti petunjuk penginstalan SDK pilihan Anda di Tools to Build on AWS saat Anda membuat container.

Menggunakan kredensialnya

Untuk menggunakan kredensional dari peran IAM untuk akun layanan, kode Anda dapat menggunakan AWS SDK apa pun untuk membuat klien untuk AWS layanan dengan SDK, dan secara default SDK mencari di rantai lokasi untuk kredensional yang akan digunakan. AWS Identity and Access Management Peran IAM untuk kredenal akun layanan akan digunakan jika Anda tidak menentukan penyedia kredensi saat membuat klien atau menginisialisasi SDK.

Ini berfungsi karena peran IAM untuk akun layanan telah ditambahkan sebagai langkah dalam rantai kredensi default. Jika beban kerja Anda saat ini menggunakan kredenal yang sebelumnya dalam

rantai kredensional, kredensial tersebut akan terus digunakan meskipun Anda mengonfigurasi peran IAM untuk akun layanan untuk beban kerja yang sama.

SDK secara otomatis menukar OIDC token akun layanan untuk kredensi sementara dari AWS Security Token Service dengan menggunakan tindakan. `AssumeRoleWithWebIdentity` Amazon EKS dan tindakan SDK ini terus memutar kredensi sementara dengan memperbaruinya sebelum kedaluwarsa.

Ambil kunci penandatanganan

Kubernetesmasalah a `ProjectedServiceAccountToken` untuk masing-masing KubernetesService Account. Token ini adalah OIDC token, yang selanjutnya merupakan jenisJSON web token (JWT). Amazon EKS menghosting OIDC titik akhir publik untuk setiap cluster yang berisi kunci penandatanganan untuk token sehingga sistem eksternal dapat memvalidasinya.

Untuk memvalidasi`ProjectedServiceAccountToken`, Anda perlu mengambil kunci penandatanganan OIDC publik, juga disebut. JSON Web Key Set (JWKS) Gunakan kunci ini dalam aplikasi Anda untuk memvalidasi token. Misalnya, Anda dapat menggunakan pustaka PyjWT Python untuk memvalidasi token menggunakan kunci ini. Untuk informasi lebih lanjut tentang`ProjectedServiceAccountToken`, lihatthe section called “IAM,Kubernetes, dan OpenID Connect (OIDC) informasi latar belakang”.

Prasyarat

- Penyedia AWS Identity and Access Management (IAM) OpenID Connect (OIDC) yang sudah ada untuk cluster Anda. Untuk menentukan apakah Anda sudah memiliki satu, atau harus membuat satu, lihat Buat OIDC penyedia IAM untuk klaster Anda.
- AWS CLI— Alat baris perintah untuk bekerja dengan AWS layanan, termasuk Amazon EKS. Untuk informasi selengkapnya, lihat Menginstal, memperbarui, dan mencopot instalasi AWS CLI di Panduan Pengguna AWS Command Line Interface . Setelah menginstal AWS CLI, kami sarankan Anda juga mengkonfigurasinya. Untuk informasi selengkapnya, lihat Konfigurasi cepat dengan `aws configure` dalam Panduan Pengguna AWS Command Line Interface .

Ambil Kunci Penandatanganan OIDC Publik ()AWS CLI

1. Ambil OIDC URL untuk klaster Amazon EKS Anda menggunakan file. AWS CLI

```
$ aws eks describe-cluster --name my-cluster --query 'cluster.identity.oidc.issuer'
"https://oidc.eks.us-east-1.amazonaws.com/id/8EBDXXXX00BAE"
```

2. Ambil kunci penandatanganan publik menggunakancurl, atau alat serupa. Hasilnya adalah a JSON Web Key Set (JWKS).

> **Important**
>
> Amazon EKS membatasi panggilan ke titik akhir. OIDC Anda harus men-cache kunci penandatanganan publik. Hormati `cache-control` header yang termasuk dalam respons.

> **Important**
>
> Amazon EKS memutar kunci OIDC penandatanganan setiap tujuh hari.

```
$ curl https://oidc.eks.us-east-1.amazonaws.com/id/8EBDXXXX00BAE/keys
{"keys":
[{"kty":"RSA","kid":"2284XXXX4a40","use":"sig","alg":"RS256","n":"wklbXXXXMVfQ","e":"AQAB"}
```

Simpul Amazon EKS

KubernetesNode adalah mesin yang menjalankan aplikasi kontainer. Setiap node memiliki komponen-komponen berikut:

- Container runtime — Perangkat lunak yang bertanggung jawab untuk menjalankan kontainer.
- **`kubelet`**— Pastikan wadah sehat dan berjalan sesuai dengan yang terkaitPod.
- **`kube-proxy`**— Mempertahankan aturan jaringan yang memungkinkan komunikasi ke AndaPods.

Untuk informasi selengkapnya, lihat Node dalam Kubernetes dokumentasi.

Cluster Amazon EKS Anda dapat menjadwalkan Pods kombinasi node yang dikelola sendiri, grup node terkelola Amazon EKS, dan AWS Fargate. Untuk mempelajari lebih lanjut tentang node yang digunakan di klaster Anda, lihatLihat Kubernetes sumber daya.

> **Important**
>
> AWS Fargatedengan Amazon EKS tidak tersedia di AWS GovCloud (AS-Timur) dan AWS GovCloud (AS-Barat).

> **Note**
>
> Node harus berada di VPC yang sama dengan subnet yang Anda pilih saat Anda membuat cluster. Namun, node tidak harus berada di subnet yang sama.

Tabel berikut memberikan beberapa kriteria untuk dievaluasi saat memutuskan opsi mana yang paling sesuai dengan kebutuhan Anda. Tabel ini tidak menyertakan node terhubung yang dibuat di luar Amazon EKS, yang hanya dapat dilihat.

> **Note**
>
> Bottlerocketmemiliki beberapa perbedaan spesifik dari informasi umum dalam tabel ini. Untuk informasi lebih lanjut, lihat Bottlerocket dokumentasi diGitHub.

Kriteria	Grup simpul terkelola EKS	Node yang dikelola sendiri	AWS Fargate
Dapat di-deploy ke AWS Outposts	Tidak	Ya	Tidak
Dapat dikerahkan ke Zona AWSLokal	Tidak	Ya — Untuk informasi lebih lanjut, lihat Amazon EKS danAWSZona Lokal.	Tidak
Dapat menjalankan kontainer yang membutuhkan Windows	Ya	Ya — Cluster Anda masih membutuhkan setidaknya satu (dua yang direkomendasikan untuk ketersediaan) Linux node.	Tidak
Dapat menjalankan kontainer yang membutuhkan Linux	Ya	Ya	Ya
Dapat menjalankan beban kerja yang memerlukan Chip Inferentia	Ya — Hanya untuk simpul Amazon Linux	Ya Hanya untuk Amazon Linux	Tidak
Dapat menjalankan beban kerja yang memerlukan GPU	Ya — Hanya untuk simpul Amazon Linux	Ya Hanya untuk Amazon Linux	Tidak
Dapat menjalankan beban kerja yang memerlukan prosesor Arm	Ya	Ya	Tidak
Dapat menjalankan AWS Bottlerocket	Ya	Ya	Tidak

Kriteria	Grup simpul terkelola EKS	Node yang dikelola sendiri	AWS Fargate
Pod berbagi lingkungan runtime kernel dengan yang lain Pods	Ya - Semua Anda Pods di setiap node Anda	Ya - Semua Anda Pods di setiap node Anda	Tidak - Masing-masing Pod memiliki kernel khusus
Pod berbagi CPU, memori, penyimpanan, dan sumber daya jaringan dengan yang lainPods.	Yes - Dapat mengakiba tkan sumber daya yang tidak terpakai pada setiap simpul	Yes - Dapat mengakiba tkan sumber daya yang tidak terpakai pada setiap simpul	Tidak - Masing-masing Pod memiliki sumber daya khusus dan dapat diukur secara independe n untuk memaksimalkan pemanfaatan sumber daya.
Pod dapat menggunakan lebih banyak perangkat keras dan memori daripada yang diminta dalam Pod spesifikasi	Ya — Jika Pod membutuhkan lebih banyak sumber daya daripada yang diminta, dan sumber daya tersedia di node, Pod dapat menggunakan sumber daya tambahan.	Ya — Jika Pod membutuhkan lebih banyak sumber daya daripada yang diminta, dan sumber daya tersedia di node, Pod dapat menggunakan sumber daya tambahan.	Tidak - Ini Pod dapat digunakan kembali menggunak an vCPU dan konfigurasi memori yang lebih besar.

Kriteria	Grup simpul terkelola EKS	Node yang dikelola sendiri	AWS Fargate
Harus men-deploy dan mengelola instans Amazon EC2	Ya — otomatis melalui Amazon EKS jika Anda men-deploy Amazon EKS yang dioptimalkan AMI. Jika Anda men-deploy AMI kustom, maka Anda harus memperbarui instans secara manual.	Ya - Konfigurasi manual atau menggunakan AWS CloudFormation template yang disediakan Amazon EKS untuk menerapkan Linux(x86), Linux(Arm), atau Windowsnode.	Tidak
Harus mengamankan, memelihara, dan patch sistem operasi dari instans Amazon EC2	Ya	Ya	Tidak
Dapat memberikan argumen bootstrap pada penerapan node, seperti `kubelet`argumen tambahan.	Ya - Menggunakan `eksctl` atau meluncurkan template dengan AMI kustom	Ya - Untuk informasi lebih lanjut, lihat informasi penggunaan skrip bootstrap diGitHub.	Tidak

Kriteria	Grup simpul terkelola EKS	Node yang dikelola sendiri	AWS Fargate
Dapat menetapkan alamat IP Pods dari blok CIDR yang berbeda dari alamat IP yang ditetapkan ke node.	Ya - Menggunakan template peluncuran dengan AMI kustom. Untuk informasi selengkapnya, lihat Menyesuaikan node terkelola dengan template peluncuran.	Ya — Untuk informasi lebih lanjut, lihat Jaringan khusus untuk pod.	Tidak
Dapat SSH ke simpul	Ya	Ya	Tidak - Tidak ada sistem operasi host node untuk SSH.
Dapat men-deploy AMI kustom Anda sendiri ke simpul	Ya - Menggunakan template peluncuran	Ya	Tidak
Dapat men-deploy CNI kustom Anda sendiri ke simpul	Ya — Menggunakan templat peluncuran dengan AMI kustom	Ya	Tidak

Kriteria	Grup simpul terkelola EKS	Node yang dikelola sendiri	AWS Fargate
Harus memperbarui node AMI sendiri	Ya - Jika Anda menerapkan AMI yang dioptimalkan Amazon EKS, Anda akan diberi tahu di konsol Amazon EKS saat pembaruan tersedia. Anda dapat melakukan pembaruan dengan satu klik di konsol. Jika Anda menerapkan AMI kustom, Anda tidak akan diberi tahu di konsol Amazon EKS saat pembaruan tersedia. Anda harus melakukan pembaruan sendiri.	Ya — Menggunakan alat selain konsol Amazon EKS. Ini karena node yang dikelola sendiri tidak dapat dikelola dengan konsol Amazon EKS.	Tidak

Kriteria	Grup simpul terkelola EKS	Node yang dikelola sendiri	AWS Fargate
Harus memperbarui Kubernetes versi node sendiri	Ya - Jika Anda menerapkan AMI yang dioptimalkan Amazon EKS, Anda akan diberi tahu di konsol Amazon EKS saat pembaruan tersedia. Anda dapat melakukan pembaruan dengan satu klik di konsol. Jika Anda menerapkan AMI kustom, Anda tidak akan diberi tahu di konsol Amazon EKS saat pembaruan tersedia. Anda harus melakukan pembaruan sendiri.	Ya — Menggunakan alat selain konsol Amazon EKS. Ini karena node yang dikelola sendiri tidak dapat dikelola dengan konsol Amazon EKS.	Tidak - Anda tidak mengelola simpul.
Dapat menggunakan penyimpanan Amazon EBS dengan Pods	Ya	Ya	Tidak
Dapat menggunakan penyimpanan Amazon EFS dengan Pods	Ya	Ya	Ya
Dapat menggunakan Amazon FSx for Lustre storage dengan Pods	Ya	Ya	Tidak

Kriteria	Grup simpul terkelola EKS	Node yang dikelola sendiri	AWS Fargate
Dapat menggunakan Network Load Balancer untuk layanan	Ya	Ya	Ya, saat menggunakan Buat penyeimbang beban jaringan
Pods dapat berjalan dalam subnet publik	Ya	Ya	Tidak
Dapat menetapkan grup keamanan VPC yang berbeda ke individu Pods	Ya — hanya Linux node	Ya — hanya Linux node	Ya
Bisa lari Kubernetes DaemonSets	Ya	Ya	Tidak
Support `HostPort` dan `HostNetwork` dalam Pod manifes	Ya	Ya	Tidak
Wilayah AWSketersediaan	Semua wilayah yang didukung Amazon EKS	Semua wilayah yang didukung Amazon EKS	Beberapa wilayah yang didukung Amazon EKS
Dapat menjalankan kontainer di host khusus Amazon EC2	Ya	Ya	Tidak

Kriteria	Grup simpul terkelola EKS	Node yang dikelola sendiri	AWS Fargate
Harga	Biaya instans Amazon EC2 yang berjalan berlipat ganda. Pods Untuk informasi lebih lanjut, lihat Harga Amazon EC2.	Biaya instans Amazon EC2 yang berjalan berlipat ganda. Pods Untuk informasi lebih lanjut, lihat harga Amazon EC2.	Biaya memori Fargate secara individu dan konfigurasi CPU. Masing-masing Pod memiliki biaya sendiri. Untuk informasi lebih lanjut, lihat Harga AWS Fargate.

Grup simpul terkelola

Grup node terkelola Amazon EKS mengotomatiskan penyediaan dan pengelolaan siklus hidup node (instans Amazon EC2) untuk kluster Amazon EKS. Kubernetes

Dengan grup node terkelola Amazon EKS, Anda tidak perlu menyediakan atau mendaftarkan instans Amazon EC2 secara terpisah yang menyediakan kapasitas komputasi untuk menjalankan aplikasi Anda. Kubernetes Anda dapat membuat, memperbarui secara otomatis, atau mengakhiri simpul untuk klaster Anda dengan satu pengoperasian. Pembaruan dan penghentian node secara otomatis menguras node untuk memastikan bahwa aplikasi Anda tetap tersedia.

Setiap node terkelola disediakan sebagai bagian dari grup Auto Scaling Amazon EC2 yang dikelola untuk Anda oleh Amazon EKS. Setiap sumber daya termasuk instans dan grup Auto Scaling berjalan di dalam AWS akun Anda. Setiap grup node berjalan di beberapa Availability Zone yang Anda tentukan.

Anda dapat menambahkan grup node terkelola ke cluster baru atau yang sudah ada menggunakan konsol Amazon EKS,`eksctl`, AWS CLI; AWS API, atau infrastruktur sebagai alat kode termasuk AWS CloudFormation. Node yang diluncurkan sebagai bagian dari grup node terkelola secara otomatis ditandai untuk penemuan otomatis oleh autoscaler Kubernetes cluster. Anda dapat menggunakan grup node untuk menerapkan Kubernetes label ke node dan memperbaruinya kapan saja.

Tidak ada biaya tambahan untuk menggunakan grup simpul terkelola Amazon EKS, Anda hanya membayar untuk sumber daya AWS yang Anda sediakan. Ini termasuk instans Amazon EC2, volume Amazon EBS, jam cluster Amazon EKS, dan infrastruktur lainnya. AWS Tidak ada biaya minimum dan tidak ada komitmen di muka.

Untuk memulai dengan klaster Amazon EKS baru dan grup simpul terkelola, lihat Memulai dengan Amazon EKS — AWS Management Console dan AWS CLI.

Untuk menambahkan grup simpul terkelola ke klaster yang ada, lihat Membuat grup simpul terkelola.

Konsep grup simpul terkelola

- Grup simpul terkelola Amazon EKS membuat dan mengelola instans Amazon EC2 untuk Anda.
- Setiap node terkelola disediakan sebagai bagian dari grup Auto Scaling Amazon EC2 yang dikelola untuk Anda oleh Amazon EKS. Selain itu, setiap sumber daya termasuk instans Amazon EC2 dan grup Auto Scaling berjalan dalam akun Anda. AWS
- Grup Auto Scaling dari grup node terkelola mencakup setiap subnet yang Anda tentukan saat membuat grup.
- Amazon EKS menandai sumber daya grup node yang dikelola sehingga dikonfigurasi untuk menggunakan Kubernetes Cluster Autoscaler.

> **Important**
>
> Jika Anda menjalankan aplikasi stateful di beberapa Availability Zone yang didukung oleh volume Amazon EBS dan menggunakan KubernetesPenskalaan otomatis, Anda harus mengonfigurasi beberapa grup node, masing-masing dicakup ke satu Availability Zone. Selain itu, Anda harus mengaktifkan opsi fitur `--balance-similar-node-groups`.

- Anda dapat menggunakan templat peluncuran khusus untuk tingkat fleksibilitas dan penyesuaian yang lebih besar saat menerapkan node terkelola. Misalnya, Anda dapat menentukan `kubelet` argumen tambahan dan menggunakan AMI kustom. Untuk informasi selengkapnya, lihat Menyesuaikan node terkelola dengan template peluncuran. Jika Anda tidak menggunakan templat peluncuran khusus saat pertama kali membuat grup node terkelola, ada templat peluncuran yang dibuat secara otomatis. Jangan memodifikasi templat yang dibuat secara otomatis ini atau terjadi kesalahan secara manual.
- Amazon EKS mengikuti model tanggung jawab bersama untuk CVE dan patch keamanan pada grup simpul terkelola. Ketika simpul terkelola menjalankan Amazon EKS yang dioptimalkan AMI,

Amazon EKS bertanggung jawab untuk membangun versi patch AMI ketika bug atau masalah dilaporkan. Kita bisa memublikasikan perbaikan. Namun, Anda bertanggung jawab untuk men-deploy versi patch AMI ini grup simpul terkelola Anda. Ketika simpul terkelola menjalankan AMI kustom, Anda bertanggung jawab untuk membangun versi patch AMI ketika bug atau masalah dilaporkan dan kemudian men-deploy AMI. Untuk informasi selengkapnya, lihat Memperbarui grup simpul terkelola.

- Grup simpul terkelola Amazon EKS dapat diluncurkan di subnet publik maupun privat. Jika Anda meluncurkan grup simpul terkelola di subnet publik pada atau setelah 22 April 2020, subnet tersebut harus mengatur `MapPublicIpOnLaunch` ke BETUL agar instans berhasil bergabung dengan klaster. Jika subnet publik dibuat menggunakan `eksctl` atau AWS CloudFormation templat penjual Amazon EKS pada atau setelah 26 Maret 2020, maka pengaturan ini sudah disetel ke true. Jika subnet publik dibuat sebelum 26 Maret 2020, Anda harus mengubah pengaturan secara manual. Untuk informasi selengkapnya, lihat Memodifikasi atribut `IPv4` pengalamatan publik untuk subnet Anda.
- Saat menerapkan grup node terkelola di subnet pribadi, Anda harus memastikan bahwa grup tersebut dapat mengakses Amazon ECR untuk menarik gambar kontainer. Anda dapat melakukan ini dengan menghubungkan gateway NAT ke tabel rute subnet atau dengan menambahkan titik akhir VPC berikut AWS PrivateLink :
  - Antarmuka titik akhir API Amazon ECR - `com.amazonaws.`*`region-code`*`.ecr.api`
  - Antarmuka titik akhir API registri Amazon ECR Docker - `com.amazonaws.`*`region-code`*`.ecr.dkr`
  - Titik akhir gerbang Amazon S3 - `com.amazonaws.`*`region-code`*`.s3`

  Untuk layanan dan titik akhir lainnya yang umum digunakan, lihat. Persyaratan klaster pribadi
- Grup node terkelola tidak dapat digunakan pada AWS Outpostsatau di AWS Wavelength atau AWS Local Zones.
- Anda dapat membuat beberapa grup simpul terkelola dalam satu klaster. Misalnya, Anda dapat membuat satu grup node dengan Amazon EKS standar yang dioptimalkan Amazon Linux AMI untuk beberapa beban kerja dan lainnya dengan varian GPU untuk beban kerja yang memerlukan dukungan GPU.
- Jika grup node terkelola mengalami kegagalan pemeriksaan status instans Amazon EC2, Amazon EKS mengembalikan kode kesalahan untuk membantu Anda mendiagnosis masalah tersebut. Untuk informasi selengkapnya, lihat Kode kesalahan grup node terkelola.
- Amazon EKS menambahkan Kubernetes label ke instance grup node terkelola. Amazon EKS yang diberi label ini diawali dengan `eks.amazonaws.com`.

- Amazon EKS secara otomatis menguras node menggunakan Kubernetes API selama penghentian atau pembaruan.
- Anggaran gangguan pod tidak dihormati saat mengakhiri sebuah node dengan `AZRebalance` atau mengurangi jumlah node yang diinginkan. Tindakan ini mencoba untuk mengusir Pods pada node. Tetapi jika dibutuhkan lebih dari 15 menit, node dihentikan terlepas dari apakah semua Pods pada node dihentikan. Untuk memperpanjang periode hingga node dihentikan, tambahkan hook siklus hidup ke grup Auto Scaling. Untuk informasi selengkapnya, lihat Menambahkan kait siklus hidup di Panduan Pengguna Auto Scaling Amazon EC2.
- Untuk menjalankan proses drain dengan benar setelah menerima pemberitahuan gangguan Spot atau pemberitahuan penyeimbangan kapasitas, `CapacityRebalance` harus diatur ke. `true`
- Memperbarui grup node terkelola menghormati anggaran Pod gangguan yang Anda tetapkan untuk Anda. Pods Untuk informasi selengkapnya, lihat Perilaku pembaruan simpul terkelola.
- Tidak ada biaya tambahan untuk menggunakan grup simpul terkelola Amazon EKS. Anda hanya membayar AWS sumber daya yang Anda berikan.
- Jika Anda ingin mengenkripsi volume Amazon EBS untuk simpul Anda, Anda dapat men-deploy simpul menggunakan templat peluncuran. Untuk menerapkan node terkelola dengan volume Amazon EBS terenkripsi tanpa menggunakan templat peluncuran, enkripsi semua volume Amazon EBS baru yang dibuat di akun Anda. Untuk informasi selengkapnya, lihat Enkripsi secara default di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Tipe kapasitas grup simpul terkelola

Saat membuat grup simpul terkelola, Anda dapat memilih tipe kapasitas Sesuai Permintaan atau Spot. Amazon EKS menerapkan grup node terkelola dengan grup Auto Scaling Amazon EC2 yang hanya berisi Instans Spot On-Demand atau hanya Amazon EC2 Spot. Anda dapat Pods menjadwalkan aplikasi toleran kesalahan ke grup node terkelola Spot, dan aplikasi intoleran kesalahan ke grup node On-Demand dalam satu cluster. Kubernetes Secara default, grup simpul terkelola men-deploy instans Amazon EC2 Sesuai-Permintaan.

Sesuai Permintaan

Dengan Instans Sesuai Permintaan, Anda membayar kapasitas komputasi pada jam kedua tanpa komitmen jangka panjang.

Cara kerjanya

Secara default, jika Anda tidak menentukan Jenis Kapasitas, grup node terkelola disediakan dengan Instans Sesuai Permintaan. Grup simpul terkelola mengonfigurasi grup Amazon EC2 Auto Scaling atas nama Anda dengan menerapkan pengaturan berikut:

- Strategi alokasi untuk penyediaan kapasitas Sesuai Permintaan diatur ke `prioritized`. Grup simpul terkelola menggunakan urutan tipe instans yang dilewatkan dalam API untuk menentukan tipe instans mana yang akan digunakan pertama kali ketika memenuhi kapasitas Sesuai Permintaan. Misalnya, Anda dapat menentukan tiga tipe instans dalam urutan berikut: `c5.large`, `c4.large`, dan `c3.large`. Ketika Instans Sesuai Permintaan Anda diluncurkan, grup simpul terkelola memenuhi kapasitas Sesuai Permintaan dengan memulai `c5.large`, lalu `c4.large`, dan kemudian `c3.large`. Untuk informasi selengkapnya, lihat grup Auto Scaling Amazon EC2 di Panduan Pengguna Auto Scaling Amazon EC2.
- Amazon EKS menambahkan Kubernetes label berikut ke semua node dalam grup node terkelola yang menentukan tipe kapasitas:`eks.amazonaws.com/capacityType: ON_DEMAND`. Anda dapat menggunakan label ini untuk menjadwalkan aplikasi yang tidak menoleransi stateful atau kesalahan pada simpul Sesuai Permintaan.

Spot

Instans Spot Amazon EC2 adalah kapasitas Amazon EC2 cadangan yang menawarkan diskon dari harga Sesuai Permintaan. Instans Spot Amazon EC2 dapat terganggu dengan pemberitahuan gangguan dua menit ketika EC2 kembali membutuhkan kapasitas. Untuk informasi selengkapnya, lihat Instans Spot dalam Panduan Pengguna Amazon EC2 untuk Instans Linux. Anda dapat mengonfigurasi grup simpul terkelola dengan Instans Spot Amazon EC2 untuk mengoptimalkan biaya untuk simpul komputasi yang berjalan di klaster Amazon EKS Anda.

Cara kerjanya

Untuk menggunakan Instans Spot di dalam grup node terkelola, buat grup node terkelola dengan menyetel tipe kapasitas sebagai`spot`. Grup simpul terkelola mengonfigurasi grup Amazon EC2 Auto Scaling atas nama Anda dengan menerapkan praktik terbaik Spot berikut:

- Untuk memastikan bahwa node Spot Anda disediakan dalam kumpulan kapasitas Spot yang optimal, strategi alokasi diatur ke salah satu dari berikut ini:
  - `price-capacity-optimized`(PCO) — Saat membuat grup node baru di cluster dengan Kubernetes versi `1.28` atau lebih tinggi, strategi alokasi diatur ke. `price-capacity-`

`optimized` Namun, strategi alokasi tidak akan diubah untuk grup node yang sudah dibuat `capacity-optimized` sebelum grup node terkelola Amazon EKS mulai mendukung PCO.

- `capacity-optimized`(CO) - Saat membuat grup node baru dalam cluster dengan Kubernetes versi `1.27` atau lebih rendah, strategi alokasi diatur ke`capacity-optimized`.

Untuk meningkatkan jumlah kumpulan kapasitas Spot yang tersedia untuk mengalokasikan kapasitas, konfigurasikan grup node terkelola untuk menggunakan beberapa jenis instance.

- Amazon EC2 Spot Capacity Rebalancing diaktifkan sehingga Amazon EKS dapat membuang dan menyeimbangkan simpul Spot untuk meminimalkan gangguan aplikasi ketika simpul Spot berada pada risiko gangguan tinggi. Untuk informasi selengkapnya, lihat Penyeimbangan Kembali Kapasitas Auto Scaling Amazon EC2 di Panduan Pengguna Auto Scaling Amazon EC2.
  - Saat node Spot menerima rekomendasi penyeimbangan ulang, Amazon EKS secara otomatis mencoba meluncurkan node Spot pengganti baru.
  - Jika pemberitahuan gangguan Spot dua menit tiba sebelum pengganti simpul Spot berada dalam status `Ready`, Amazon EKS mulai membuang simpul Spot yang menerima rekomendasi penyeimbangan kembali. Amazon EKS menguras node dengan upaya terbaik. Akibatnya, tidak ada jaminan bahwa Amazon EKS akan menunggu node pengganti bergabung dengan cluster sebelum menguras node yang ada.
  - Saat node Spot pengganti di-bootstrap dan dalam `Ready` status aktif, Kubernetes Amazon EKS mengikat dan menguras node Spot yang menerima rekomendasi penyeimbangan kembali. Menjaga simpul Spot memastikan bahwa pengendali layanan tidak mengirim permintaan baru ke simpul Spot ini. Hal ini juga menghapusnya dari daftar simpul Spot yang sehat, aktif. Menguras node Spot memastikan bahwa berjalan Pods diusir dengan anggun.
- Amazon EKS menambahkan Kubernetes label berikut ke semua node dalam grup node terkelola yang menentukan tipe kapasitas:`eks.amazonaws.com/capacityType: SPOT`. Anda dapat menggunakan label ini untuk menjadwalkan aplikasi toleran kesalahan pada simpul Spot.

Pertimbangan untuk memilih tipe kapasitas

Ketika ingin memutuskan apakah akan men-deploy grup simpul dengan kapasitas Sesuai Permintaan atau Spot, Anda harus mempertimbangkan kondisi berikut:

- Instans Spot cocok untuk aplikasi tanpa kewarganegaraan, toleran kesalahan, dan fleksibel. Ini termasuk beban kerja pelatihan pembelajaran batch dan mesin, ETL data besar seperti Apache Spark, aplikasi pemrosesan antrian, dan titik akhir API stateless. Karena Spot adalah kapasitas Amazon EC2 cadangan, yang dapat berubah seiring waktu, kami menyarankan Anda

menggunakan kapasitas Spot untuk beban kerja yang toleran interupsi. Lebih khusus lagi, kapasitas Spot cocok untuk beban kerja yang dapat mentolerir periode di mana kapasitas yang diperlukan tidak tersedia.

- Kami menyarankan Anda menggunakan On-Demand untuk aplikasi yang tidak toleran terhadap kesalahan. Ini termasuk alat manajemen cluster seperti alat pemantauan dan operasional, penerapan yang memerlukan`StatefulSets`, dan aplikasi stateful, seperti database.
- Untuk memaksimalkan ketersediaan aplikasi Anda saat menggunakan Instans Spot, sebaiknya konfigurasikan grup simpul terkelola Spot untuk menggunakan beberapa tipe instans. Kami merekomendasikan agar Anda menerapkan aturan berikut ketika menggunakan beberapa tipe instans:
  - Dalam grup node terkelola, jika Anda menggunakan Cluster Autoscaler, sebaiknya gunakan kumpulan tipe instans yang fleksibel dengan jumlah vCPU dan sumber daya memori yang sama. Ini untuk memastikan bahwa node dalam skala cluster Anda seperti yang diharapkan. Misalnya, jika Anda memerlukan empat vCPU dan delapan memori GiB, `c3.xlarge` gunakan,,,,, `c4.xlarge` `c5.xlarge` `c5d.xlarge` `c5a.xlargec5n.xlarge`, atau jenis instance serupa lainnya.
  - Untuk meningkatkan ketersediaan aplikasi, sebaiknya gunakan beberapa grup node terkelola Spot. Untuk ini, setiap grup harus menggunakan satu set tipe instance fleksibel yang memiliki vCPU dan sumber daya memori yang sama. Sebagai contoh, jika Anda membutuhkan memori 4 vCPUs dan 8 GiB, kami sarankan Anda membuat satu grup simpul terkelola dengan `c3.xlarge`, `c4.xlarge`, `c5.xlarge`, `c5d.xlarge`, `c5a.xlarge`, `c5n.xlarge`, atau tipe instans serupa lainnya, dan grup simpul terkelola kedua dengan `m3.xlarge`, `m4.xlarge`, `m5.xlarge`, `m5d.xlarge`, `m5a.xlarge`, `m5n.xlarge` atau tipe instans serupa lainnya.
  - Saat menerapkan grup node Anda dengan tipe kapasitas Spot yang menggunakan templat peluncuran khusus, gunakan API untuk meneruskan beberapa jenis instance. Jangan melewatkan satu jenis instance melalui template peluncuran. Untuk informasi selengkapnya tentang men-deploy grup simpul menggunakan templat peluncuran, lihat Menyesuaikan node terkelola dengan template peluncuran.

Membuat grup simpul terkelola

Topik ini menjelaskan bagaimana Anda dapat meluncurkan grup node terkelola Amazon EKS dari node yang mendaftar dengan kluster Amazon EKS Anda. Setelah node bergabung dengan cluster, Anda dapat menyebarkan Kubernetes aplikasi ke mereka.

Jika ini adalah pertama kalinya Anda meluncurkan grup simpul terkelola Amazon EKS, kami sarankan Anda mengikuti salah satu panduan Memulai dengan Amazon EKS kami sebagai gantinya. Panduan tersebut memberi petunjuk untuk membuat klaster Amazon EKS dengan simpul.

Important

- Simpul Amazon EKS adalah instans Amazon EC2 standar. Anda dikenakan biaya berdasarkan harga normal Amazon EC2. Untuk informasi selengkapnya, lihat Harga Amazon EC2.
- Anda tidak dapat membuat node terkelola di Wilayah AWS mana Anda memiliki AWS Outposts, AWS Wavelength, atau AWS Local Zones diaktifkan. Anda dapat membuat node yang dikelola sendiri di Wilayah AWS tempat Anda mengaktifkan AWS Outposts, atau AWS Wavelength, Local Zones. Untuk informasi lebih lanjut, lihat Meluncurkan simpul Amazon Linux yang dikelola sendiri, , dan Meluncurkan node yang dikelola sendiri Bottlerocket. Anda juga dapat membuat grup node Amazon Linux yang dikelola sendiri di Outpost. Untuk informasi selengkapnya, lihat Meluncurkan node Amazon Linux yang dikelola sendiri di Outpost.
- Jika Anda tidak menentukan ID AMI untuk `bootstrap.sh` file yang disertakan dengan Linux atau Bottlerocket yang dioptimalkan Amazon EKS, grup node terkelola menerapkan angka maksimum pada nilai. `maxPods` Untuk contoh dengan kurang dari 30 vCPU, jumlah maksimumnya adalah. `110` Untuk contoh dengan lebih dari 30 vCPU, jumlah maksimum melompat ke. `250` Angka-angka ini didasarkan pada ambang Kubernetes skalabilitas dan pengaturan yang direkomendasikan oleh pengujian tim skalabilitas Amazon EKS internal. Untuk informasi selengkapnya, lihat plugin Amazon VPC CNI meningkatkan pod per node membatasi posting blog.

Prasyarat

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMembuat klaster Amazon EKS.
- Peran IAM yang ada untuk digunakan node. Untuk membuatnya, lihat IAM role simpul Amazon EKS. Jika peran ini tidak memiliki salah satu kebijakan untuk VPC CNI, peran terpisah yang mengikuti diperlukan untuk pod VPC CNI.
- (Opsional, tetapi disarankan) Amazon VPC CNI plugin for Kubernetes Add-on dikonfigurasi dengan peran IAM sendiri yang memiliki kebijakan IAM yang diperlukan yang melekat padanya. Untuk

informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).

- Keakraban dengan pertimbangan yang tercantum dalam. Memilih jenis instans Amazon EC2 Bergantung pada jenis instans yang Anda pilih, mungkin ada prasyarat tambahan untuk cluster dan VPC Anda.
- Untuk menambahkan grup node Windows terkelola, Anda harus terlebih dahulu mengaktifkan Windows dukungan untuk klaster Anda. Untuk informasi selengkapnya, lihat Mengaktifkan Windows dukungan untuk klaster Amazon EKS Anda.

Anda dapat membuat grup simpul terkelola dengan `eksctl` atau AWS Management Console.

eksctl

Untuk membuat grup simpul terkelola dengan **`eksctl`**

Prosedur ini membutuhkan `eksctl` versi `0.175.0` atau yang lebih baru. Anda dapat memeriksa versi Anda dengan perintah berikut:

```
eksctl version
```

Untuk petunjuk tentang cara menginstal atau meningkatkan`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

1. (Opsional) Jika kebijakan IAM terkelola Amazoneks_CNI_Policy dilampirkan pada kebijakan IAM Anda, kami sarankan untuk menetapkannya ke peran IAM yang IAM role simpul Amazon EKS Anda kaitkan ke akun layanan sebagai gantinya. Kubernetes `aws-node` Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).
2. Buat grup node terkelola dengan atau tanpa menggunakan templat peluncuran kustom. Menentukan template peluncuran secara manual memungkinkan kustomisasi grup node yang lebih besar. Misalnya, ini dapat memungkinkan penerapan AMI khusus atau memberikan argumen ke `boostrap.sh` skrip di AMI Amazon EKS yang dioptimalkan. Untuk daftar lengkap setiap opsi yang tersedia dan default, masukkan perintah berikut.

   ```
   eksctl create nodegroup --help
   ```

Dalam perintah berikut, ganti *my-cluster* dengan nama cluster Anda dan ganti *my-mng* dengan nama grup node Anda. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa.

> **Important**
>
> Jika Anda tidak menggunakan templat peluncuran kustom saat pertama kali membuat grup node terkelola, jangan gunakan satu di lain waktu untuk grup node. Jika Anda tidak menentukan templat peluncuran kustom, sistem akan secara otomatis membuat templat peluncuran yang tidak kami sarankan agar Anda memodifikasi secara manual. Memodifikasi templat peluncuran yang dibuat secara otomatis ini secara manual dapat menyebabkan kesalahan.

Tanpa template peluncuran

`eksctl`membuat template peluncuran Amazon EC2 default di akun Anda dan menyebarkan grup node menggunakan templat peluncuran yang dibuat berdasarkan opsi yang Anda tentukan. Sebelum menentukan nilai untuk`--node-type`, lihatMemilih jenis instans Amazon EC2.

Ganti *ami-family* dengan kata kunci yang diizinkan. Untuk informasi selengkapnya, lihat Menyetel simpul AMI Family dalam `eksctl` dokumentasi. Ganti *my-key* dengan nama pasangan kunci atau kunci publik Amazon EC2 Anda. Kunci ini digunakan untuk SSH ke simpul Anda setelah diluncurkan.

> **Note**
>
> UntukWindows, perintah ini tidak mengaktifkan SSH. Sebagai gantinya, ini mengaitkan key pair Amazon EC2 Anda dengan instans dan memungkinkan Anda untuk RDP ke instans.

Jika Anda belum memiliki pasangan kunci Amazon EC2, maka Anda dapat membuatnya di AWS Management Console. Untuk Linux selengkapnya, lihat pasangan kunci dan Linux instans Amazon EC2 di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Untuk Windows selengkapnya, lihat pasangan kunci Amazon EC2 dan Windows instans di Panduan Pengguna Amazon EC2 untuk Instans Windows.

Kami merekomendasikan memblokir Pod akses ke IMDS jika kondisi berikut benar:

- Anda berencana untuk menetapkan peran IAM ke semua akun Kubernetes layanan Anda sehingga Pods hanya memiliki izin minimum yang mereka butuhkan.
- Tidak ada Pods di klaster yang memerlukan akses ke layanan metadata instans Amazon EC2 (IMDS) karena alasan lain, seperti mengambil arus. Wilayah AWS

Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

Jika Anda ingin memblokir Pod akses ke IMDS, tambahkan **`--disable-pod-imds`** opsi ke perintah berikut.

```
eksctl create nodegroup \
  --cluster my-cluster \
  --region region-code \
  --name my-mng \
  --node-ami-family ami-family \
  --node-type m5.large \
  --nodes 3 \
  --nodes-min 2 \
  --nodes-max 4 \
  --ssh-access \
  --ssh-public-key my-key
```

Instance Anda secara opsional dapat menetapkan jumlah alamat IP yang jauh lebih tinggiPods, menetapkan alamat IP Pods dari blok CIDR yang berbeda dari instans, dan digunakan ke cluster tanpa akses internet. Untuk informasi selengkapnyaTingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda, lihatJaringan khusus untuk pod,, dan Persyaratan klaster pribadi untuk opsi tambahan untuk ditambahkan ke perintah sebelumnya.

Grup node terkelola menghitung dan menerapkan satu nilai untuk jumlah maksimum Pods yang dapat dijalankan pada setiap node grup node Anda, berdasarkan jenis instance. Jika Anda membuat grup node dengan tipe instance yang berbeda, nilai terkecil yang dihitung di semua jenis instance diterapkan sebagai jumlah maksimum Pods yang dapat dijalankan pada setiap jenis instance dalam grup node. Grup node terkelola menghitung

nilai menggunakan skrip yang direferensikan di. Amazon EKS merekomendasikan maksimum Pods untuk setiap jenis instans Amazon EC2

Dengan template peluncuran

Template peluncuran harus sudah ada dan harus memenuhi persyaratan yang ditentukan dalamLuncurkan dasar-dasar konfigurasi templat.

Kami merekomendasikan memblokir Pod akses ke IMDS jika kondisi berikut benar:

- Anda berencana untuk menetapkan peran IAM ke semua akun Kubernetes layanan Anda sehingga Pods hanya memiliki izin minimum yang mereka butuhkan.
- Tidak ada Pods di klaster yang memerlukan akses ke layanan metadata instans Amazon EC2 (IMDS) karena alasan lain, seperti mengambil arus. Wilayah AWS

Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

Jika Anda ingin memblokir Pod akses ke IMDS, maka tentukan pengaturan yang diperlukan dalam template peluncuran.

a. Salin konten berikut ke perangkat Anda. Ganti *example values* dan kemudian jalankan perintah yang dimodifikasi untuk membuat `eks-nodegroup.yaml` file. Beberapa pengaturan yang Anda tentukan saat men-deploy tanpa templat peluncuran dipindahkan ke templat peluncuran. Jika Anda tidak menentukan versi `version`, maka versi default templat akan digunakan.

```
cat >eks-nodegroup.yaml <<EOF
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code
managedNodeGroups:
- name: my-mng
  launchTemplate:
    id: lt-id
    version: "1"
EOF
```

Untuk daftar selengkapnya tentang `eksctl` pengaturan file konfigurasi, lihat Skema file Config di dokumentasi `eksctl`. Instance Anda secara opsional dapat menetapkan

jumlah alamat IP yang jauh lebih tinggiPods, menetapkan alamat IP Pods dari blok CIDR yang berbeda dari instans, menggunakan `containerd` runtime, dan digunakan ke cluster tanpa akses internet keluar. Untuk informasi selengkapnyaTingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda, lihat Jaringan khusus untuk podUji migrasi dari Docker ke `containerd`,,, dan Persyaratan klaster pribadi untuk opsi tambahan untuk ditambahkan ke file konfigurasi.

Jika Anda tidak menentukan ID AMI dalam template peluncuran, grup node terkelola menghitung dan menerapkan satu nilai untuk jumlah maksimum Pods yang dapat dijalankan pada setiap node grup node Anda, berdasarkan jenis instance. Jika Anda membuat grup node dengan tipe instance yang berbeda, nilai terkecil yang dihitung di semua jenis instance diterapkan sebagai jumlah maksimum Pods yang dapat dijalankan pada setiap jenis instance dalam grup node. Grup node terkelola menghitung nilai menggunakan skrip yang direferensikan di. Amazon EKS merekomendasikan maksimum Pods untuk setiap jenis instans Amazon EC2

Jika Anda menetapkan ID AMI di template peluncuran, tentukan jumlah maksimum Pods yang dapat dijalankan pada setiap node grup node jika Anda menggunakan jaringan khusus atau ingin menambah jumlah alamat IP yang ditetapkan ke instans Anda. Untuk informasi selengkapnya, lihat Amazon EKS merekomendasikan maksimum Pods untuk setiap jenis instans Amazon EC2.

b. Men-deploy grup simpul dengan perintah berikut.

```
eksctl create nodegroup --config-file eks-nodegroup.yaml
```

AWS Management Console

Untuk membuat grup node terkelola menggunakan AWS Management Console

1. Tunggu status klaster Anda ditampilkan sebagai `ACTIVE`. Anda tidak dapat membuat grup node terkelola untuk klaster yang belum ada`ACTIVE`.
2. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
3. Pilih nama cluster tempat Anda ingin membuat grup node terkelola.
4. Pilih tab Compute.
5. Pilih Tambahkan grup simpul.

6. Pada halaman Konfigurasi grup simpul, isi parameter yang sesuai, dan kemudian pilih Selanjutnya.

   - Nama — Masukkan nama unik untuk grup simpul terkelola Anda. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa.
   - Peran IAM Node — Pilih peran instance node yang akan digunakan dengan grup node Anda. Untuk informasi selengkapnya, lihat IAM role simpul Amazon EKS.

     > ⚠ Important
     >
     > - Anda tidak dapat menggunakan peran yang sama yang digunakan untuk membuat cluster apa pun.
     > - Sebaiknya gunakan peran yang saat ini tidak digunakan oleh grup node yang dikelola sendiri. Jika tidak, Anda berencana untuk menggunakan dengan grup node yang dikelola sendiri yang baru. Untuk informasi selengkapnya, lihat Menghapus grup simpul terkelola.

   - Gunakan template peluncuran - (Opsional) Pilih jika Anda ingin menggunakan template peluncuran yang ada. Pilih Nama Template Luncurkan. Kemudian, pilih versi Template Luncurkan. Jika Anda tidak memilih versi, maka Amazon EKS menggunakan versi default templat. Template peluncuran memungkinkan lebih banyak penyesuaian grup node Anda, seperti memungkinkan Anda untuk menerapkan AMI kustom, menetapkan jumlah alamat IP yang jauh lebih tinggiPods, menetapkan alamat IP dari blok CIDR yang berbeda Pods dari instans, mengaktifkan `containerd` runtime untuk instans Anda, dan menyebarkan node ke cluster tanpa akses internet keluar. Untuk informasi lebih lanjut, lihat Tingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda, Jaringan khusus untuk pod, Uji migrasi dari Docker ke `containerd`, dan Persyaratan klaster pribadi .

     Templat peluncuran harus memenuhi persyaratan di Menyesuaikan node terkelola dengan template peluncuran. Jika Anda tidak menggunakan templat peluncuran Anda sendiri, API Amazon EKS membuat templat peluncuran Amazon EC2 default di akun Anda dan men-deploy grup simpul menggunakan templat peluncuran default.

     Jika Anda menerapkan peran IAM untuk akun layanan, menetapkan izin yang diperlukan secara langsung ke setiap Pod yang memerlukan akses ke AWS layanan, dan tidak ada Pods di klaster Anda yang memerlukan akses ke IMDS karena alasan lain, seperti mengambil saat ini Wilayah AWS, maka Anda juga dapat menonaktifkan akses ke IMDS

untuk Pods itu jangan gunakan jaringan host dalam template peluncuran. Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

- Kubernetes label — (Opsional) Anda dapat memilih untuk menerapkan Kubernetes label ke node di grup node terkelola Anda.
- Kubernetes taints — (Opsional) Anda dapat memilih untuk menerapkan Kubernetes taints ke node di grup node terkelola Anda. Pilihan yang tersedia di menu Effect adalah**`NoSchedule`**,**`NoExecute`**, dan**`PreferNoSchedule`**. Untuk informasi selengkapnya, lihat Noda simpul pada grup simpul terkelola.
- Tanda – (Opsional) Anda dapat memilih untuk menandai grup simpul terkelola Amazon EKS Anda. Tag ini tidak menyebar ke sumber daya lain di grup node, seperti grup Auto Scaling atau instance. Untuk informasi selengkapnya, lihat Menandai sumber daya Amazon EKS Anda.

7. Pada halaman Atur konfigurasi komputasi dan penskalaan, isi parameter yang sesuai, dan kemudian pilih Selanjutnya.

   - Jenis AMI - Pilih tipe AMI. Jika Anda menerapkan instance Arm, pastikan untuk meninjau pertimbangan sebelum menerapkan. Amazon EKS mengoptimalkan AMI Arm Amazon Linux

     Jika Anda menentukan template peluncuran di halaman sebelumnya, dan menetapkan AMI di template peluncuran, maka Anda tidak dapat memilih nilai. Nilai dari templat ditampilkan. AMI yang ditentukan dalam template harus memenuhi persyaratan diMenentukan AMI.
   - Tipe kapasitas – Pilih tipe kapasitas. Untuk informasi selengkapnya tentang memilih tipe kapasitas, lihat Tipe kapasitas grup simpul terkelola. Anda tidak dapat mencampur tipe kapasitas yang berbeda dalam grup node yang sama. Jika Anda ingin menggunakan kedua tipe kapasitas, buatlah grup simpul terpisah, masing-masing dengan tipe kapasitas dan instans mereka sendiri.
   - Jenis instans - Secara default, satu atau lebih jenis instance ditentukan. Untuk menghapus tipe instans default, pilih X di sisi kanan tipe instans. Pilih tipe instans yang akan digunakan dalam grup simpul terkelola. Untuk informasi selengkapnya, lihat Memilih jenis instans Amazon EC2.

     Konsol menampilkan satu set tipe instans yang umum digunakan. Jika Anda perlu membuat grup node terkelola dengan tipe instans yang tidak ditampilkan`eksctl`, gunakan AWS CLI, AWS CloudFormation, atau SDK untuk membuat grup node. Jika Anda menentukan template peluncuran pada halaman sebelumnya, maka Anda tidak dapat

memilih nilai karena jenis instance harus ditentukan dalam template peluncuran. Nilai dari templat peluncuran ditampilkan. Jika Anda memilih jenis Spot untuk Kapasitas, sebaiknya tentukan beberapa jenis instans untuk meningkatkan ketersediaan.

- Ukuran disk — Masukkan ukuran disk (dalam GiB) untuk digunakan untuk volume root simpul Anda.

  Jika Anda menentukan template peluncuran di halaman sebelumnya, maka Anda tidak dapat memilih nilai karena harus ditentukan dalam template peluncuran.

- Ukuran yang diinginkan – Tentukan jumlah simpul yang harus dipertahankan oleh grup simpul terkelola saat peluncuran.

  > **Note**
  >
  > Amazon EKS tidak secara otomatis menskalakan grup node Anda masuk atau keluar. Namun, Anda dapat mengonfigurasi Kubernetes Cluster Autoscaler untuk melakukan ini untuk Anda.

- Ukuran minimum – Tentukan jumlah simpul minimum yang dapat diskalakan kedalam oleh grup simpul terkelola.
- Ukuran maksimum – Tentukan jumlah simpul maksimum yang dapat diskalakan keluar oleh grup simpul terkelola.
- Konfigurasi pembaruan grup node — (Opsional) Anda dapat memilih jumlah atau persentase node yang akan diperbarui secara paralel. Node ini tidak akan tersedia selama pembaruan. Untuk Maksimum tidak tersedia, pilih salah satu opsi berikut dan tentukan Nilai:
  - Angka — Pilih dan tentukan jumlah node dalam grup node Anda yang dapat diperbarui secara paralel.
  - Persentase — Pilih dan tentukan persentase node dalam grup node Anda yang dapat diperbarui secara paralel. Ini berguna jika Anda memiliki sejumlah besar node di grup node Anda.

8. Di halaman Tentukan jaringan, isi parameter yang sesuai, dan pilih Selanjutnya.

   - Subnet — Pilih subnet untuk meluncurkan simpul terkelola Anda.

> **Important**
>
> Jika Anda menjalankan aplikasi stateful di beberapa Availability Zone yang didukung oleh volume Amazon EBS dan menggunakan KubernetesPenskalaan otomatis, Anda harus mengonfigurasi beberapa grup node, masing-masing dicakup ke satu Availability Zone. Selain itu, Anda harus mengaktifkan fitur `--balance-similar-node-groups`.

> **Important**
>
> - Jika Anda memilih subnet publik, dan klaster Anda hanya memiliki titik akhir server API publik yang diaktifkan, maka subnet harus mengatur `MapPublicIPOnLaunch` ke `true` agar instans dapat bergabung dengan klaster. Jika subnet dibuat menggunakan `eksctl` atau AWS CloudFormation templat yang dijual Amazon EKS pada atau setelah 26 Maret 2020, maka pengaturan ini sudah diatur ke `true`. Jika subnet dibuat dengan `eksctl` atau AWS CloudFormation templat sebelum 26 Maret 2020, maka Anda perlu mengubah pengaturan secara manual. Untuk informasi selengkapnya, lihat Memodifikasi atribut `IPv4` pengalamatan publik untuk subnet Anda.
> - Jika Anda menggunakan templat peluncuran dan menentukan beberapa antarmuka jaringan, Amazon EC2 tidak akan menetapkan alamat `IPv4` publik secara otomatis, meskipun `MapPublicIpOnLaunch` disetel ke. `true` Untuk simpul yang akan bergabung dengan klaster dalam skenario ini, maka Anda harus mengaktifkan titik akhir server API privat klaster Anda atau meluncurkan simpul dalam subnet privat dengan akses internet keluar yang disediakan melalui metode alternatif, seperti Gateway NAT. Untuk informasi selengkapnya, lihat Pengalamatan IP instans Amazon EC2 di Panduan Pengguna Amazon EC2 untuk Instans Linux.

- Konfigurasikan akses SSH ke node (Opsional). Mengaktifkan SSH mengizinkan Anda untuk tekoneksi ke instans dan mengumpulkan informasi diagnostik jika ada masalah. Kami sangat menyarankan untuk mengaktifkan akses jarak jauh saat Anda membuat grup node. Anda tidak dapat mengaktifkan akses jarak jauh setelah grup node dibuat.

Jika Anda memilih untuk menggunakan templat peluncuran, maka opsi ini tidak ditampilkan. Untuk mengaktifkan akses jarak jauh ke simpul Anda, tentukan pasangan kunci di templat peluncuran dan pastikan bahwa port yang tepat terbuka untuk simpul dalam grup keamanan yang Anda tentukan di templat peluncuran. Untuk informasi selengkapnya, lihat Menggunakan grup keamanan kustom.

> **Note**
>
> UntukWindows, perintah ini tidak mengaktifkan SSH. Sebagai gantinya, ini mengaitkan key pair Amazon EC2 Anda dengan instans dan memungkinkan Anda untuk RDP ke instans.

- Untuk pasangan kunci SSH (Opsional), pilih kunci SSH Amazon EC2 untuk digunakan. Untuk Linux selengkapnya, lihat pasangan kunci dan Linux instans Amazon EC2 di Panduan Pengguna Amazon EC2 untuk Instans Linux. Untuk Windows selengkapnya, lihat pasangan kunci Amazon EC2 dan Windows instans di Panduan Pengguna Amazon EC2 untuk Instans Windows. Jika Anda memilih untuk menggunakan templat peluncuran, maka Anda tidak dapat memilih templat lain. Ketika kunci SSH Amazon EC2 disediakan untuk grup node yang menggunakan Bottlerocket AMI, penampung administratif juga diaktifkan. Untuk informasi selengkapnya, lihat Kontainer admin diGitHub.
- Untuk Izinkan akses jarak jauh SSH dari, jika Anda ingin membatasi akses ke instance tertentu, pilih grup keamanan yang terkait dengan instance tersebut. Jika Anda tidak memilih grup keamanan tertentu, maka akses SSH diizinkan dari mana saja di internet (`0.0.0.0/0`).

9. Pada halaman Tinjau dan buat, tinjau konfigurasi grup simpul terkelola dan pilih Buat.

   Jika simpul gagal untuk bergabung dengan klaster, maka lihat Simpul gagal untuk bergabung dengan klaster di panduan Pemecahan Masalah.

10. Perhatikan status simpul Anda dan tunggu sampai simpul mencapai Status `Ready`.

    ```
    kubectl get nodes --watch
    ```

11. (Hanya node GPU) Jika Anda memilih jenis instans GPU dan AMI akselerasi Amazon EKS yang dioptimalkan, maka Anda harus menerapkan plugin perangkat NVIDIA untuk Kubernetes sebagai DaemonSet di cluster Anda. Ganti *vX.X.X* dengan s-device-plugin versi NVIDIA/K8 yang Anda inginkan sebelum menjalankan perintah berikut.

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

Sekarang setelah Anda memiliki kluster Amazon EKS yang berfungsi dengan node, Anda siap untuk mulai menginstal Kubernetes add-on dan menerapkan aplikasi ke cluster Anda. Topik dokumentasi berikut membantu Anda untuk memperluas fungsionalitas klaster Anda.

- Prinsipal IAM yang membuat cluster adalah satu-satunya prinsipal yang dapat melakukan panggilan ke server Kubernetes API dengan `kubectl` atau. AWS Management Console Jika Anda ingin prinsipal IAM lainnya memiliki akses ke cluster Anda, maka Anda perlu menambahkannya. Lihat informasi yang lebih lengkap di Berikan akses ke Kubernetes API dan Izin yang diperlukan.
- Kami merekomendasikan memblokir Pod akses ke IMDS jika kondisi berikut benar:
  - Anda berencana untuk menetapkan peran IAM ke semua akun Kubernetes layanan Anda sehingga Pods hanya memiliki izin minimum yang mereka butuhkan.
  - Tidak ada Pods di klaster yang memerlukan akses ke layanan metadata instans Amazon EC2 (IMDS) karena alasan lain, seperti mengambil arus. Wilayah AWS

  Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.
- Penskalaan otomatis— Konfigurasikan Kubernetes Cluster Autoscaler untuk secara otomatis menyesuaikan jumlah node dalam grup node Anda.
- Menerapkan aplikasi sampel ke cluster Anda.
- Manajemen klaster – Pelajari cara menggunakan alat penting untuk mengelola klaster Anda.

Memperbarui grup simpul terkelola

Saat Anda memulai pembaruan grup node terkelola, Amazon EKS secara otomatis memperbarui node untuk Anda, menyelesaikan langkah-langkah yang tercantum di dalamnyaPerilaku pembaruan simpul terkelola. Jika Anda menggunakan AMI yang dioptimasi dengan Amazon EKS, Amazon EKS secara otomatis menerapkan patch keamanan terbaru dan pembaruan sistem operasi ke simpul Anda sebagai bagian dari versi perilisan AMI terbaru.

Ada beberapa skenario di mana pembaruan versi atau konfigurasi grup simpul terkelola Amazon EKS sangat berguna untuk dilakukan:

- Anda telah memperbarui Kubernetes versi untuk cluster Amazon EKS Anda dan ingin memperbarui node Anda untuk menggunakan Kubernetes versi yang sama.
- Versi perilisan AMI yang baru tersedia untuk grup simpul terkelola Anda. Untuk informasi selengkapnya tentang versi AMI, lihat bagian ini:
  - Versi Amazon Linux AMI yang dioptimalkan oleh Amazon EKS
  - Amazon EKS mengoptimalkan Bottlerocket AMI
  - Amazon EKS dioptimalkan versi Windows AMI
- Anda ingin menyesuaikan jumlah instans minimum, maksimum, atau diinginkan di grup simpul terkelola Anda.
- Anda ingin menambahkan atau menghapus Kubernetes label dari instance di grup node terkelola Anda.
- Anda ingin menambahkan atau menghapus tanda AWS dari grup simpul terkelola Anda.
- Anda harus men-deploy versi templat peluncuran yang baru dengan perubahan konfigurasi, seperti AMI kustom yang diperbarui.
- Anda telah menerapkan versi `1.9.0` atau yang lebih baru dari add-on Amazon VPC CNI, mengaktifkan add-on untuk delegasi awalan, dan ingin instance AWS Nitro System baru dalam grup node mendukung peningkatan jumlah yang signifikan. Pods Untuk informasi selengkapnya, lihat Tingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda.
- Anda telah mengaktifkan delegasi awalan IP untuk node Windows dan ingin instance Sistem AWS Nitro baru dalam grup node untuk mendukung peningkatan jumlah yang signifikan. Pods Untuk informasi selengkapnya, lihat Tingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda.

Jika ada versi rilis AMI yang lebih baru untuk versi grup node terkelola, Anda dapat memperbarui Kubernetes versi grup node untuk menggunakan versi AMI yang lebih baru. Demikian pula, jika klaster menjalankan Kubernetes versi yang lebih baru dari grup node, Anda dapat memperbarui grup node untuk menggunakan versi rilis AMI terbaru agar sesuai dengan versi klaster Anda. Kubernetes

Ketika sebuah node dalam grup node terkelola dihentikan karena operasi penskalaan atau pembaruan, node tersebut terkuras Pods terlebih dahulu. Untuk informasi selengkapnya, lihat Perilaku pembaruan simpul terkelola.

Memperbarui versi grup simpul

Anda dapat memperbarui versi grup simpul dengan `eksctl` atau AWS Management Console. Versi yang Anda perbarui tidak boleh lebih besar dari versi pesawat kontrol.

eksctl

Untuk memperbarui versi grup simpul dengan **`eksctl`**

- Perbarui grup node terkelola ke rilis AMI terbaru dari Kubernetes versi yang sama yang saat ini digunakan di node dengan perintah berikut. Ganti setiap *`example value`*dengan nilai-nilai Anda sendiri.

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code
```

> **Note**
>
> Jika Anda memutakhirkan grup simpul yang di-deploy dengan templat peluncuran ke versi templat peluncuran baru, tambahkan `--launch-template-version` *`version-number`* ke perintah sebelumnya. Templat peluncuran harus memenuhi persyaratan yang dijelaskan di Menyesuaikan node terkelola dengan template peluncuran. Jika templat peluncuran mencakup AMI kustom, AMI harus memenuhi persyaratan di Menentukan AMI. Saat Anda memutakhirkan grup node ke versi template peluncuran yang lebih baru, setiap node didaur ulang agar sesuai dengan konfigurasi baru dari versi template peluncuran yang ditentukan.
> Anda tidak dapat langsung memutakhirkan grup simpul yang di-deploy tanpa templat peluncuran ke versi templat peluncuran baru. Sebaliknya, Anda harus men-deploy grup simpul baru dengan menggunakan templat peluncuran untuk memperbarui grup simpul ke versi templat peluncuran baru.

Anda dapat memutakhirkan grup node ke versi yang sama dengan Kubernetes versi bidang kontrol. Misalnya, jika Anda menjalankan klaster Kubernetes`1.28`, Anda dapat memutakhirkan node yang sedang berjalan Kubernetes `1.27` ke versi `1.28` dengan perintah berikut.

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code \
  --kubernetes-version=1.28
```

AWS Management Console

Untuk memperbarui versi grup simpul dengan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih klaster yang berisi grup simpul yang akan diperbarui.
3. Jika setidaknya tersedia satu grup simpul yang memiliki pembaruan, maka akan ada kotak yang muncul di bagian atas halaman yang memberitahukan Anda tentang pembaruan yang tersedia. Jika Anda memilih tab Compute, Anda akan melihat Perbarui sekarang di kolom versi rilis AMI di tabel grup Node untuk grup node yang memiliki pembaruan yang tersedia. Untuk memperbarui grup node, pilih Perbarui sekarang.

   Anda tidak akan melihat notifikasi untuk grup simpul yang di-deploy dengan AMI kustom. Jika simpul di-deploy dengan AMI kustom, selesaikan langkah-langkah berikut untuk men-deploy AMI kustom yang baru diperbarui.

   a. Buat versi baru AMI Anda.

   b. Buat versi templat peluncuran baru dengan ID AMI baru.

   c. Mutakhirkan simpul ke versi templat peluncuran baru.
4. Pada kotak dialog Perbarui versi grup node, aktifkan atau nonaktifkan opsi berikut:

   - Perbarui versi grup node - Opsi ini tidak tersedia jika Anda menerapkan AMI khusus atau AMI Amazon EKS yang dioptimalkan saat ini ada di versi terbaru untuk klaster Anda.
   - Ubah versi template peluncuran - Opsi ini tidak tersedia jika grup simpul dikerahkan tanpa templat peluncuran khusus. Anda hanya dapat memperbarui versi templat peluncuran untuk grup simpul yang telah di-deploy dengan templat peluncuran kustom. Pilih versi Template Luncurkan yang ingin Anda perbarui grup node. Jika grup simpul dikonfigurasi dengan AMI kustom, maka versi yang Anda pilih juga harus menentukan AMI. Saat Anda memutakhirkan ke versi template peluncuran yang lebih baru, setiap node didaur ulang agar sesuai dengan konfigurasi baru dari versi template peluncuran yang ditentukan.

5. Untuk strategi Update, pilih salah satu opsi berikut:

   - Pembaruan bergulir - Opsi ini menghormati anggaran Pod gangguan untuk klaster Anda. Pembaruan gagal jika ada masalah anggaran Pod gangguan yang menyebabkan Amazon EKS tidak dapat menguras yang berjalan di grup node Pods ini dengan anggun.
   - Pembaruan paksa - Opsi ini tidak menghormati anggaran Pod gangguan. Pembaruan terjadi terlepas dari masalah anggaran Pod gangguan dengan memaksa restart node terjadi.

6. Pilih Perbarui.

Edit konfigurasi grup simpul

Anda dapat mengubah beberapa konfigurasi dari grup simpul terkelola.

Untuk mengedit konfigurasi grup simpul

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih klaster yang berisi grup simpul untuk mengedit.
3. Pilih tab Compute.
4. Pilih grup node yang akan diedit, lalu pilih Edit.
5. (Opsional) Pada halaman grup Edit node, lakukan hal berikut:

   a. Edit konfigurasi penskalaan grup Node.

      - Ukuran yang diinginkan – Tentukan jumlah simpul saat ini yang harus dipertahankan oleh grup simpul terkelola.
      - Ukuran minimum – Tentukan jumlah simpul minimum yang dapat diskalakan kedalam oleh grup simpul terkelola.
      - Ukuran maksimum – Tentukan jumlah maksimum simpul yang dapat diskalakan keluar oleh grup simpul terkelola. Untuk jumlah maksimal simpul yang didukung dalam grup simpul, lihat Amazon EKS service quotas.

   b. (Opsional) Tambahkan atau hapus Kuberneteslabel ke node di grup node Anda. Label yang ditampilkan di sini hanya label yang telah Anda terapkan dengan Amazon EKS. Label lain mungkin ada di simpul Anda namun tidak ditampilkan di sini.

   c. (Opsional) Tambahkan atau hapus Kubernetesnoda ke node di grup node Anda. Kecacatan yang ditambahkan dapat memiliki efek, baik **`NoSchedule`**, **`NoExecute`**, atau

`PreferNoSchedule`. Untuk informasi selengkapnya, lihat Noda simpul pada grup simpul terkelola.

d. (Opsional) Tambahkan atau hapus Tag dari sumber daya grup node Anda. Tanda ini hanya diterapkan pada grup simpul Amazon EKS. Mereka tidak menyebar ke sumber daya lain, seperti subnet atau instans Amazon EC2 di grup node.

e. (Opsional) Edit konfigurasi pembaruan Grup Node. Pilih Angka atau Persentase.

- Angka — Pilih dan tentukan jumlah node dalam grup node Anda yang dapat diperbarui secara paralel. Node ini tidak akan tersedia selama pembaruan.
- Persentase — Pilih dan tentukan persentase node dalam grup node Anda yang dapat diperbarui secara paralel. Node ini tidak akan tersedia selama pembaruan. Ini berguna jika Anda memiliki banyak node di grup node Anda.

f. Setelah selesai mengedit, pilih Simpan perubahan.

Perilaku pembaruan simpul terkelola

Strategi upgrade node pekerja terkelola Amazon EKS memiliki empat fase berbeda yang dijelaskan di bagian berikut.

Fase pengaturan

Fase pengaturan memiliki langkah-langkah berikut:

1. Ini membuat versi template peluncuran Amazon EC2 baru untuk grup Auto Scaling yang terkait dengan grup node Anda. Versi template peluncuran baru menggunakan AMI target atau versi template peluncuran khusus untuk pembaruan.
2. Ini memperbarui grup Auto Scaling untuk menggunakan versi template peluncuran terbaru.
3. Ini menentukan jumlah maksimum node untuk meng-upgrade secara paralel menggunakan `updateConfig` properti untuk kelompok node. Maksimum yang tidak tersedia memiliki kuota 100 node. Nilai default adalah satu node. Untuk informasi selengkapnya, lihat `updateConfig`properti di Referensi API Amazon EKS.

Tingkatkan fase

Saat memutakhirkan node dalam grup node terkelola, node yang ditingkatkan diluncurkan di Availability Zone yang sama dengan yang sedang ditingkatkan. Untuk menjamin penempatan ini, kami menggunakan Penyeimbangan Zona Ketersediaan Amazon EC2. Untuk informasi

selengkapnya, lihat Penyeimbangan Kembali Zona Ketersediaan di Panduan Pengguna Auto Scaling Amazon EC2. Untuk memenuhi persyaratan ini, ada kemungkinan bahwa kami akan meluncurkan hingga dua instance per Availability Zone di grup node terkelola Anda.

Fase peningkatan skala memiliki langkah-langkah berikut:

1. Ini meningkatkan ukuran maksimum Grup Auto Scaling dan ukuran yang diinginkan dengan ukuran yang lebih besar:
   - Hingga dua kali jumlah Availability Zone tempat grup Auto Scaling digunakan.
   - Upgrade maksimum yang tidak tersedia.

     Misalnya, jika grup node Anda memiliki lima Availability Zones dan `maxUnavailable` sebagai satu, proses upgrade dapat meluncurkan maksimal 10 node. Namun ketika `maxUnavailable` 20 (atau apa pun yang lebih tinggi dari 10), proses akan meluncurkan 20 node baru.
2. Setelah menskalakan grup Auto Scaling, ia memeriksa apakah node yang menggunakan konfigurasi terbaru ada di grup node. Langkah ini hanya berhasil jika memenuhi kriteria ini:
   - Setidaknya satu node baru diluncurkan di setiap Availability Zone di mana node ada.
   - Setiap node baru harus dalam `Ready` keadaan.
   - Node baru harus memiliki label yang diterapkan Amazon EKS.

     Ini adalah label yang diterapkan Amazon EKS pada node pekerja dalam grup node biasa:
     - `eks.amazonaws.com/nodegroup-image=`*`$amiName`*
     - `eks.amazonaws.com/nodegroup=`*`$nodeGroupName`*

     Ini adalah label yang diterapkan Amazon EKS pada node pekerja dalam template peluncuran khusus atau grup node AMI:
     - `eks.amazonaws.com/nodegroup-image=`*`$amiName`*
     - `eks.amazonaws.com/nodegroup=`*`$nodeGroupName`*
     - `eks.amazonaws.com/sourceLaunchTemplateId=`*`$launchTemplateId`*
     - `eks.amazonaws.com/sourceLaunchTemplateVersion=`*`$launchTemplateVersion`*
3. Ini menandai node sebagai tidak dapat dijadwalkan untuk menghindari penjadwalan baru. Pods Ini juga memberi label node `node.kubernetes.io/exclude-from-external-load-balancers=true` untuk menghapus node dari penyeimbang beban sebelum mengakhiri node.

Berikut ini adalah alasan yang diketahui yang menyebabkan `NodeCreationFailure` kesalahan dalam fase ini:

Kapasitas tidak mencukupi di Availability Zone

Ada kemungkinan bahwa Availability Zone mungkin tidak memiliki kapasitas tipe instans yang diminta. Disarankan untuk mengonfigurasi beberapa jenis instance saat membuat grup node terkelola.

Batas instans EC2 di akun Anda

Anda mungkin perlu menambah jumlah instans Amazon EC2 yang dapat dijalankan akun Anda secara bersamaan menggunakan Service Quotas. Untuk informasi selengkapnya, lihat Service Quotas EC2 di Panduan Pengguna Amazon Elastic Compute Cloud untuk Instans. Linux

Data pengguna kustom

Data pengguna khusus terkadang dapat merusak proses bootstrap. Skenario ini dapat menyebabkan `kubelet` tidak dimulai pada node atau node tidak mendapatkan label Amazon EKS yang diharapkan pada mereka. Untuk informasi selengkapnya, lihat Menentukan AMI.

Setiap perubahan yang membuat node tidak sehat atau tidak siap

Tekanan disk node, tekanan memori, dan kondisi serupa dapat menyebabkan node tidak akan `Ready` berstatus.

Fase upgrade

Fase upgrade memiliki langkah-langkah berikut:

1. Ini secara acak memilih node yang perlu ditingkatkan, hingga maksimum yang tidak tersedia yang dikonfigurasi untuk grup node.
2. Ini menguras Pods dari simpul. Jika Pods tidak meninggalkan node dalam waktu 15 menit dan tidak ada tanda gaya, fase pemutakhiran gagal dengan `PodEvictionFailure` kesalahan. Untuk skenario ini, Anda dapat menerapkan bendera gaya dengan `update-nodegroup-version` permintaan untuk menghapusPods.
3. Ini mengikat simpul setelah setiap Pod diusir dan menunggu selama 60 detik. Hal ini dilakukan agar pengontrol layanan tidak mengirim permintaan baru ke node ini dan menghapus node ini dari daftar node aktifnya.
4. Ini mengirimkan permintaan penghentian ke Grup Auto Scaling untuk node yang diapit.

5. Ini mengulangi langkah-langkah upgrade sebelumnya sampai tidak ada node dalam grup node yang digunakan dengan versi sebelumnya dari template peluncuran.

Berikut ini adalah alasan yang diketahui yang menyebabkan `PodEvictionFailure` kesalahan dalam fase ini:

PDB agresif

PDB agresif didefinisikan pada Pod atau ada beberapa PDB yang menunjuk ke yang sama. Pod

Penerapan yang menoleransi semua noda

Setelah setiap Pod diusir, diharapkan node kosong karena node tercemar pada langkah-langkah sebelumnya. Namun, jika penerapan mentolerir setiap noda, maka node lebih cenderung tidak kosong, yang menyebabkan kegagalan penggusuran. Pod

Skala turun fase

Fase penurunan skala mengurangi ukuran maksimum grup Auto Scaling dan ukuran yang diinginkan satu per satu untuk kembali ke nilai sebelum pembaruan dimulai.

Jika alur kerja Upgrade menentukan bahwa Cluster Autoscaler meningkatkan grup node selama fase penurunan skala alur kerja, ia segera keluar tanpa membawa grup node kembali ke ukuran aslinya.

Noda simpul pada grup simpul terkelola

Amazon EKS mendukung konfigurasi Kubernetes taint melalui grup node terkelola. Taints dan toleransi bekerja sama untuk memastikan bahwa Pods tidak dijadwalkan ke node yang tidak pantas. Satu atau lebih cacat dapat diterapkan pada sebuah simpul. Ini menandai bahwa node tidak boleh menerima apa pun Pods yang tidak mentolerir noda. Toleransi diterapkan Pods dan memungkinkan, tetapi tidak memerlukan, Pods untuk menjadwalkan ke node dengan noda yang cocok. Untuk informasi selengkapnya, lihat Taints and Tolerations dalam dokumentasi. Kubernetes

Kubernetesnode taints dapat diterapkan ke grup node terkelola baru dan yang sudah ada menggunakan AWS Management Console atau melalui Amazon EKS API.

- Untuk informasi tentang membuat grup node dengan taint menggunakanAWS Management Console, lihatMembuat grup simpul terkelola.
- Berikut ini adalah contoh pembuatan sebuah grup simpul dengan suatu noda menggunakan perintah AWS CLI:

```
aws eks create-nodegroup \
 --cli-input-json '
{
  "clusterName": "my-cluster",
  "nodegroupName": "node-taints-example",
  "subnets": [
     "subnet-1234567890abcdef0",
     "subnet-abcdef01234567890",
     "subnet-021345abcdef67890"
   ],
  "nodeRole": "arn:aws:iam::111122223333:role/AmazonEKSNodeRole",
  "taints": [
     {
         "key": "dedicated",
         "value": "gpuGroup",
         "effect": "NO_SCHEDULE"
     }
   ]
}'
```

Untuk informasi selengkapnya dan contoh penggunaan, lihat taint dalam dokumentasi Kubernetes referensi.

Note

- Taints dapat diperbarui setelah Anda membuat grup node menggunakan `UpdateNodegroupConfig` API.
- Kunci cacat harus diawali dengan huruf atau angka. Ini dapat berisi huruf, angka, tanda hubung (`-`), periode (`.`), dan garis bawah (). `_` Panjangnya bisa mencapai 63 karakter.
- Secara opsional, kunci cacat dapat dimulai dengan prefiks subdomain DNS dan satu `/`. Jika dimulai dengan prefiks subdomain DNS, panjangnya bisa mencapai 253 karakter.
- Nilai adalah opsional dan harus diawali dengan huruf atau angka. Ini dapat berisi huruf, angka, tanda hubung (`-`), periode (`.`), dan garis bawah (). `_` Panjangnya bisa mencapai 63 karakter.
- Saat menggunakan Kubernetes langsung atauAWS Management Console, efek noda harus`NoSchedule`,`PreferNoSchedule`, atau`NoExecute`. Namun, saat

menggunakan API AWS CLI atau, efek taint harus**NO_SCHEDULE**,**PREFER_NO_SCHEDULE**, atau**NO_EXECUTE**.

- Maksimal 50 taint diperbolehkan per grup node.
- Jika taint yang dibuat menggunakan grup node terkelola dihapus secara manual dari node, maka Amazon EKS tidak menambahkan taints kembali ke node. Ini benar bahkan jika taints ditentukan dalam konfigurasi grup node terkelola.

Anda dapat menggunakan `aws eks update-nodegroup-config`AWS CLIperintah untuk menambah, menghapus, atau mengganti taints untuk grup node terkelola.

Menyesuaikan node terkelola dengan template peluncuran

Untuk kustomisasi tingkat tertinggi, Anda dapat menerapkan node terkelola menggunakan template peluncuran Anda sendiri. Menggunakan template peluncuran memungkinkan kemampuan seperti berikut:

- Berikan argumen bootstrap pada penerapan node, seperti `kubelet`argumen tambahan.
- Tetapkan alamat IP Pods dari blok CIDR yang berbeda dari alamat IP yang ditetapkan ke node.
- Terapkan AMI kustom Anda sendiri ke node.
- Terapkan CNI kustom Anda sendiri ke node.

Saat Anda memberikan template peluncuran Anda sendiri saat pertama kali membuat grup node terkelola, Anda juga akan memiliki fleksibilitas yang lebih besar nanti. Selama Anda menerapkan grup node terkelola dengan templat peluncuran Anda sendiri, Anda dapat memperbaruinya secara berulang dengan versi berbeda dari template peluncuran yang sama. Saat Anda memperbarui grup node ke versi template peluncuran yang berbeda, semua node dalam grup didaur ulang agar sesuai dengan konfigurasi baru dari versi template peluncuran yang ditentukan.

Grup node terkelola selalu digunakan dengan template peluncuran untuk digunakan dengan grup Auto Scaling Amazon EC2. Jika Anda tidak menyediakan template peluncuran, Amazon EKS API membuatnya secara otomatis dengan nilai default di akun Anda. Namun, kami tidak menyarankan Anda memodifikasi templat peluncuran yang dibuat secara otomatis. Selain itu, grup node yang ada yang tidak menggunakan templat peluncuran kustom tidak dapat diperbarui secara langsung. Sebagai gantinya, Anda harus membuat grup node baru dengan template peluncuran khusus untuk melakukannya.

Luncurkan dasar-dasar konfigurasi templat

Anda dapat membuat template peluncuran Auto Scaling Amazon EC2 dengan AWS CLI,, atau AWS Management Console SDK. AWS Untuk informasi selengkapnya, lihat Membuat Template Peluncuran untuk grup Auto Scaling di Panduan Pengguna Auto Scaling Amazon EC2. Beberapa pengaturan dalam templat peluncuran mirip dengan pengaturan yang digunakan untuk konfigurasi simpul terkelola. Saat menerapkan atau memperbarui grup node dengan template peluncuran, beberapa pengaturan harus ditentukan baik dalam konfigurasi grup node atau template peluncuran. Jangan tentukan pengaturan di kedua tempat. Jika terdapat pengaturan yang seharusnya tidak ada, maka operasi seperti membuat atau memperbarui grup simpul akan gagal.

Tabel berikut mencantumkan pengaturan yang dilarang dalam template peluncuran. Ini juga mencantumkan pengaturan serupa, jika ada yang tersedia, yang diperlukan dalam konfigurasi grup node terkelola. Pengaturan yang tercantum adalah pengaturan yang muncul di konsol. Mereka mungkin memiliki nama yang mirip tetapi berbeda di AWS CLI dan SDK.

Peluncuran templat — Dilarang	Konfigurasi grup simpul Amazon EKS
Subnet dalam Antarmuka Jaringan (Tambahkan antarmuka jaringan)	Subnet di bawah konfigurasi jaringan grup Node pada halaman Tentukan jaringan
Profil instans IAM dalam Detail lanjutan	Peran IAM node di bawah konfigurasi grup Node pada halaman grup Configure Node
Perilaku Shutdown dan Perilaku Berhenti - Hibernasi dalam Detail lanjutan. Pertahankan default Jangan sertakan dalam pengaturan templat peluncuran dalam templat peluncuran untuk kedua pengaturan.	Tidak setara. Amazon EKS harus mengontrol siklus hidup instans, bukan grup Auto Scaling.

Tabel berikut mencantumkan pengaturan yang dilarang dalam konfigurasi grup node terkelola. Ini juga mencantumkan pengaturan serupa, jika ada yang tersedia, yang diperlukan dalam template peluncuran. Pengaturan yang tercantum adalah pengaturan yang muncul di konsol. Mereka mungkin memiliki nama yang mirip di AWS CLI dan SDK.

Konfigurasi grup simpul Amazon EKS - Dilarang	Luncurkan templat

(Hanya jika Anda menetapkan AMI kustom dalam template peluncuran) Jenis AMI di bawah konfigurasi komputasi grup Node pada halaman konfigurasi Set komputasi dan penskalaan — Tampilan konsol Ditentukan dalam templat peluncuran dan ID AMI yang ditentukan.
 	
Jika Gambar Aplikasi dan OS (Amazon Machine Image) tidak ditentukan dalam template peluncuran, Anda dapat memilih AMI dalam konfigurasi grup node.
	Gambar Aplikasi dan OS (Gambar Mesin Amazon) di bawah Meluncurkan konten template - Anda harus menentukan ID jika Anda memiliki salah satu dari persyaratan berikut:
	
	• Menggunakan AMI khusus. Jika Anda menentukan AMI yang tidak memenuhi persyaratan yang tercantum di Menentukan AMI, deployment grup simpul akan gagal.
	• Ingin memberikan data pengguna untuk memberikan argumen ke `bootstrap .sh` file yang disertakan dengan AMI Amazon EKS yang dioptimalkan. Anda dapat mengaktifkan instans Anda untuk menetapkan jumlah alamat IP yang jauh lebih tinggiPods, menetapkan alamat IP Pods dari blok CIDR yang berbeda dari instans, atau menyebarkan cluster pribadi tanpa akses internet keluar. Untuk informasi selengkapnya, lihat topik berikut:
	  • Tingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda
	  • Jaringan khusus untuk pod
	  • Persyaratan klaster pribadi
	  • Menentukan AMI

Ukuran disk di bawah konfigurasi komputasi grup Node pada Mengatur halaman konfigurasi komputasi dan penskalaan — Tampilan konsol Ditentukan dalam templat peluncuran.	Ukuran dalam Penyimpanan (Volume) (Tambahkan volume baru). Anda harus menentukan ini dalam templat peluncuran.

Konfigurasi grup simpul Amazon EKS - Dilarang	Luncurkan templat
SSH key pair di bawah konfigurasi grup Node pada halaman Specify Networking - Konsol menampilkan kunci yang ditentukan dalam template peluncuran atau menampilkan Tidak ditentukan dalam template peluncuran.	Nama pasangan kunci dalam Pasangan kunci (login).
Anda tidak dapat menentukan grup keamanan sumber yang diizinkan akses jarak jauh saat menggunakan templat peluncuran.	Grup keamanan dalam Pengaturan jaringan untuk instans atau Grup keamanan di bawah Antarmuka jaringan (Tambahkan antarmuka jaringan), tapi tidak keduanya. Untuk informasi selengkapnya, lihat Menggunakan grup keamanan kustom.

Note

- Jika Anda menerapkan grup node menggunakan template peluncuran, tentukan nol atau satu jenis Instance di bawah Meluncurkan konten template dalam template peluncuran. Atau, Anda dapat menentukan 0—20 jenis instans untuk tipe Instance di halaman Setel konfigurasi komputasi dan penskalaan di konsol. Atau, Anda dapat melakukannya menggunakan alat lain yang menggunakan Amazon EKS API. Jika Anda menentukan tipe instans dalam templat peluncuran, dan menggunakan templat peluncuran tersebut untuk men-deploy grup simpul Anda, maka Anda tidak dapat menentukan tipe instans apa pun di konsol atau menggunakan alat lain yang menggunakan Amazon EKS API. Jika Anda tidak menentukan jenis instance dalam template peluncuran, di konsol, atau menggunakan alat lain yang menggunakan Amazon EKS API, jenis `t3.medium` instance akan digunakan. Jika grup simpul Anda menggunakan tipe kapasitas Spot, sebaiknya tentukan beberapa tipe instans menggunakan konsol. Untuk informasi selengkapnya, lihat Tipe kapasitas grup simpul terkelola.
- Jika kontainer apa pun yang Anda terapkan ke grup node menggunakan Layanan Metadata Instans Versi 2, pastikan untuk menetapkan batas hop respons Metadata ke 2 dalam template peluncuran Anda. Untuk informasi lebih lanjut, lihat metadata instans dan data pengguna di Panduan Pengguna Amazon EC2 untuk Instans Linux. Jika Anda

menerapkan grup simpul terkelola tanpa menggunakan templat peluncuran kustom, nilai ini secara otomatis ditetapkan untuk grup simpul dalam templat peluncuran default.

Penandaan instans Amazon EC2

Anda dapat menggunakan parameter `TagSpecification` dari templat peluncuran untuk menentukan tanda mana yang akan diterapkan ke instans Amazon EC2 di grup simpul Anda. Entitas IAM yang memanggil `CreateNodegroup` atau `UpdateNodegroupVersion` API harus memiliki izin untuk `ec2:RunInstances` dan `ec2:CreateTags`, dan tanda harus ditambahkan ke templat peluncuran.

Menggunakan grup keamanan kustom

Anda dapat menggunakan templat peluncuran untuk menentukan Amazon EC2 kustom grup keamanan untuk diterapkan ke instans dalam grup simpul Anda. Ini dapat berupa parameter grup keamanan tingkat instans atau sebagai bagian dari parameter konfigurasi antarmuka jaringan. Namun, Anda tidak dapat membuat templat peluncuran yang menentukan tingkat instans dan grup keamanan antarmuka jaringan. Pertimbangkan kondisi yang berlaku berikut untuk menggunakan grup keamanan kustom dengan grup simpul terkelola:

- Amazon EKS hanya mengizinkan templat peluncuran dengan spesifikasi antarmuka jaringan tunggal.
- Secara default, Amazon EKS menerapkan grup keamanan klaster untuk instans dalam grup simpul Anda guna memfasilitasi komunikasi antara simpul dan pesawat kontrol. Jika Anda menentukan grup keamanan kustom di templat peluncuran menggunakan salah satu opsi yang disebutkan sebelumnya, Amazon EKS tidak menambahkan grup keamanan klaster. Jadi, Anda harus memastikan bahwa aturan masuk dan keluar dari grup keamanan Anda memungkinkan komunikasi dengan titik akhir klaster Anda. Jika aturan grup keamanan Anda salah, node pekerja tidak dapat bergabung dengan cluster. Untuk informasi selengkapnya tentang aturan grup keamanan, lihatPersyaratan dan pertimbangan grup keamanan Amazon EKS.
- Jika Anda memerlukan akses SSH ke instance di grup node Anda, sertakan grup keamanan yang memungkinkan akses tersebut.

Data pengguna Amazon EC2

Template peluncuran mencakup bagian untuk data pengguna khusus. Anda dapat menentukan pengaturan konfigurasi untuk grup node Anda di bagian ini tanpa membuat AMI kustom individual secara manual. Untuk informasi selengkapnya tentang setelan yang tersediaBottlerocket, lihat Menggunakan data pengguna diGitHub.

Anda dapat menyediakan data pengguna Amazon EC2 di templat peluncuran menggunakan `cloud-init` saat meluncurkan instans Anda. Untuk informasi selengkapnya, lihat dokumentasi cloud-init. Data pengguna Anda dapat digunakan untuk melakukan operasi konfigurasi umum. Ini termasuk operasi berikut:

- Termasuk pengguna atau grup
- Menginstal paket

Data pengguna Amazon EC2 dalam template peluncuran yang digunakan dengan grup node terkelola harus dalam format arsip multi-bagian MIME untuk AMI Amazon Linux dan format TOMM untuk AMI. Bottlerocket Ini karena data pengguna Anda digabungkan dengan data pengguna Amazon EKS yang diperlukan simpul untuk bergabung dengan klaster. Jangan tentukan perintah apa pun dalam data pengguna Anda yang memulai atau memodifikasi`kubelet`. Ini dilakukan sebagai bagian dari data pengguna yang digabungkan oleh Amazon EKS. Parameter `kubelet` tertentu, seperti pengaturan label pada simpul, dapat dikonfigurasi secara langsung melalui API grup simpul terkelola.

> **Note**
>
> Untuk informasi selengkapnya tentang `kubelet` kustomisasi lanjutan, termasuk memulainya secara manual atau meneruskan parameter konfigurasi kustom, lihatMenentukan AMI. Jika ID AMI kustom ditentukan dalam template peluncuran, Amazon EKS tidak menggabungkan data pengguna.

Rincian berikut memberikan informasi lebih lanjut tentang bagian data pengguna.

Amazon Linux 2 user data

Anda dapat menggabungkan beberapa blok data pengguna menjadi satu file multi-bagian MIME. Misalnya, Anda dapat menggabungkan boothook cloud yang mengonfigurasi Docker daemon

dengan skrip shell data pengguna yang menginstal paket khusus. File multi-bagian MIME terdiri dari komponen berikut:

- Tipe konten dan deklarasi batas bagian – `Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="`
- Deklarasi versi MIME – `MIME-Version: 1.0`
- Satu atau lebih blok data pengguna, yang berisi komponen berikut:
  - Batas pembukaan, yang menandakan awal dari blok data pengguna – `--==MYBOUNDARY==`
  - Deklarasi tipe konten untuk blok: `Content-Type: text/cloud-config; charset="us-ascii"`. Untuk informasi selengkapnya, lihat dokumentasi cloud-init.
  - Isi data pengguna (misalnya, daftar perintah atau `cloud-init` arahan shell).
  - Batas penutupan, yang menandakan akhir dari file multi-bagian MIME: `--==MYBOUNDARY==--`

Berikut ini adalah contoh file multi-bagian MIME yang dapat Anda gunakan untuk membuat milik Anda sendiri.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
echo "Running custom user data script"

--==MYBOUNDARY==--
```

Amazon Linux 2023 user data

Amazon Linux 2023 (AL2023) memperkenalkan proses inisialisasi node baru `nodeadm` yang menggunakan skema konfigurasi YAMB. Jika Anda menggunakan grup node yang dikelola sendiri atau AMI dengan template peluncuran, Anda sekarang harus menyediakan metadata klaster tambahan secara eksplisit saat membuat grup node baru. Contoh parameter minimum yang diperlukan adalah sebagai berikut, di mana`apiServerEndpoint`,`certificateAuthority`, dan layanan sekarang `cidr` diperlukan:

```
---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: my-cluster
    apiServerEndpoint: https://example.com
    certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
    cidr: 10.100.0.0/16
```

Anda biasanya akan mengatur konfigurasi ini dalam data pengguna Anda, baik apa adanya atau disematkan dalam dokumen multi-bagian MIME:

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="BOUNDARY"

--BOUNDARY
Content-Type: application/node.eks.aws

---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig spec: [...]

--BOUNDARY--
```

Di AL2, metadata dari parameter ini ditemukan dari panggilan Amazon EKS `DescribeCluster` API. Dengan AL2023, perilaku ini telah berubah karena panggilan API tambahan berisiko melambat selama peningkatan skala node besar. Perubahan ini tidak memengaruhi Anda jika Anda menggunakan grup node terkelola tanpa templat peluncuran atau jika Anda menggunakanKarpenter. Untuk informasi selengkapnya tentang `certificateAuthority` dan layanan`cidr`, lihat `DescribeCluster` di Referensi API Amazon EKS.

Bottlerocket user data

Bottlerocketstruktur data pengguna dalam format TOMB. Anda dapat memberikan data pengguna untuk digabungkan dengan data pengguna yang disediakan oleh Amazon EKS. Misalnya, Anda dapat memberikan `kubelet` pengaturan tambahan.

```
[settings.kubernetes.system-reserved]
cpu = "10m"
memory = "100Mi"
```

```
ephemeral-storage= "1Gi"
```

Untuk informasi selengkapnya tentang pengaturan yang didukung, lihat Bottlerocketdokumentasi. Anda dapat mengonfigurasi label node dan taints dalam data pengguna Anda. Namun, kami menyarankan Anda mengonfigurasi ini dalam grup node Anda sebagai gantinya. Amazon EKS menerapkan konfigurasi ini saat Anda melakukannya.

Saat data pengguna digabungkan, pemformatan tidak dipertahankan, tetapi kontennya tetap sama. Konfigurasi yang Anda berikan dalam data pengguna akan mengesampingkan pengaturan apa pun yang dikonfigurasi oleh Amazon EKS. Jadi, jika Anda menetapkan `settings.kubernetes.max-pods` atau`settings.kubernetes.cluster-dns-ip`, nilai dalam data pengguna Anda diterapkan ke node.

Amazon EKS tidak mendukung semua TOMB yang valid. Berikut ini adalah daftar format yang tidak didukung yang diketahui:

- Kutipan dalam kunci yang dikutip: `'quoted "value"' = "value"`
- Kutipan lolos dalam nilai: `str = "I'm a string. \"You can quote me\""`
- Campuran pelampung dan bilangan bulat: `numbers = [ 0.1, 0.2, 0.5, 1, 2, 5 ]`
- Jenis campuran dalam array: `contributors = ["foo@example.com", { name = "Baz", email = "baz@example.com" }]`
- Header bertanda kurung dengan tombol yang dikutip: `[foo."bar.baz"]`

Windows user data

Data pengguna Windows menggunakan PowerShell perintah. Saat membuat grup node terkelola, data pengguna kustom Anda digabungkan dengan data pengguna terkelola Amazon EKS. PowerShellPerintah Anda didahulukan, diikuti oleh perintah data pengguna terkelola, semuanya dalam satu **<powershell></powershell>** tag.

> **Note**
>
> Ketika tidak ada ID AMI yang ditentukan dalam template peluncuran, jangan gunakan skrip Bootstrap Windows Amazon EKS dalam data pengguna untuk mengonfigurasi Amazon EKS.

Contoh data pengguna adalah sebagai berikut.

```
<powershell>
Write-Host "Running custom user data script"
</powershell>
```

Menentukan AMI

Jika Anda memiliki salah satu dari persyaratan berikut, maka tentukan ID AMI di `ImageId` bidang template peluncuran Anda. Pilih persyaratan yang Anda miliki untuk informasi tambahan.

Berikan data pengguna untuk meneruskan argumen ke **`bootstrap.sh`** file yang disertakan dengan Amazon EKS yang dioptimalkanLinux/BottlerocketAMI

Bootstrapping adalah istilah yang digunakan untuk menggambarkan penambahan perintah yang dapat dijalankan ketika sebuah instance dimulai. Misalnya, bootstrap memungkinkan penggunaan argumen tambahan. `kubelet` Anda dapat meneruskan argumen ke `bootstrap.sh` skrip dengan menggunakan `eksctl` tanpa menentukan template peluncuran. Atau Anda dapat melakukannya dengan menentukan informasi di bagian data pengguna dari template peluncuran.

eksctl without specifying a launch template

Buat file bernama *`my-nodegroup.yaml`* dengan isi berikut ini. Ganti setiap *`example value`* dengan nilai-nilai Anda sendiri. `--dns-cluster-ip`Argumen `--apiserver-endpoint`,`--b64-cluster-ca`, dan bersifat opsional. Namun, mendefinisikannya memungkinkan `bootstrap.sh` skrip untuk menghindari `describeCluster` panggilan. Ini berguna dalam pengaturan cluster pribadi atau cluster tempat Anda sering menskalakan node masuk dan keluar. Untuk informasi lebih lanjut tentang `bootstrap.sh` skrip, lihat `bootstrap.sh`file diGitHub.

- Satu-satunya argumen yang diperlukan adalah nama cluster (*`my-cluster`*).
- Untuk mengambil ID AMI yang dioptimalkan`ami-`*`1234567890abcdef0`*, Anda dapat menggunakan tabel di bagian berikut:
  - Mendapatkan ID Amazon Linux AMI yang dioptimalkan Amazon EKS
  - Mengambil ID Bottlerocket AMI Amazon EKS yang dioptimalkan
  - Mengambil ID Windows AMI Amazon EKS yang dioptimalkan
- Untuk mengambil *`certificate-authority`* untuk cluster Anda, jalankan perintah berikut.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
 --name my-cluster --region region-code
```

- Untuk mengambil *api-server-endpoint* untuk cluster Anda, jalankan perintah berikut.

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

- Nilai untuk `--dns-cluster-ip` adalah CIDR layanan Anda dengan `.10` di akhir. Untuk mengambil *service-cidr* untuk cluster Anda, jalankan perintah berikut. Misalnya, jika nilai yang dikembalikan adalah`ipv4 10.100.0.0/16`, maka nilai Anda adalah*10.100.0.10*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
 --output text --name my-cluster --region region-code
```

- Contoh ini memberikan `kubelet` argumen untuk menetapkan `max-pods` nilai kustom menggunakan `bootstrap.sh` skrip yang disertakan dengan AMI Amazon EKS yang dioptimalkan. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa. Untuk bantuan dalam memilih*my-max-pods-value*, lihatAmazon EKS merekomendasikan maksimum Pods untuk setiap jenis instans Amazon EC2.

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

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

managedNodeGroups:
  - name: my-nodegroup
    ami: ami-1234567890abcdef0
    instanceType: m5.large
    privateNetworking: true
    disableIMDSv1: true
    labels: { x86-al2-specified-mng }
    overrideBootstrapCommand: |
      #!/bin/bash
      /etc/eks/bootstrap.sh my-cluster \
        --b64-cluster-ca certificate-authority \
        --apiserver-endpoint api-server-endpoint \
        --dns-cluster-ip service-cidr.10 \
        --kubelet-extra-args '--max-pods=my-max-pods-value' \
```

```
        --use-max-pods false
```

Untuk setiap opsi `eksctl config` file yang tersedia, lihat Skema file Config dalam dokumentasi. `eksctl` `eksctl`Utilitas masih membuat template peluncuran untuk Anda dan mengisi data penggunanya dengan data yang Anda berikan dalam `config` file.

Buat grup node dengan perintah berikut.

```
eksctl create nodegroup --config-file=my-nodegroup.yaml
```

User data in a launch template

Tentukan informasi berikut di bagian data pengguna dari template peluncuran Anda. Ganti setiap *`example value`* dengan nilai-nilai Anda sendiri. `--dns-cluster-ip`Argumen `--apiserver-endpoint`, `--b64-cluster-ca`, dan bersifat opsional. Namun, mendefinisikannya memungkinkan `bootstrap.sh` skrip untuk menghindari `describeCluster` panggilan. Ini berguna dalam pengaturan cluster pribadi atau cluster tempat Anda sering menskalakan node masuk dan keluar. Untuk informasi lebih lanjut tentang `bootstrap.sh` skrip, lihat `bootstrap.sh`file diGitHub.

- Satu-satunya argumen yang diperlukan adalah nama cluster (*`my-cluster`*).
- Untuk mengambil *`certificate-authority`* untuk cluster Anda, jalankan perintah berikut.

  ```
  aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
   --name my-cluster --region region-code
  ```

- Untuk mengambil *`api-server-endpoint`* untuk cluster Anda, jalankan perintah berikut.

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

- Nilai untuk `--dns-cluster-ip` adalah CIDR layanan Anda dengan `.10` di akhir. Untuk mengambil *`service-cidr`* untuk cluster Anda, jalankan perintah berikut. Misalnya, jika nilai yang dikembalikan adalah`ipv4 10.100.0.0/16`, maka nilai Anda adalah*`10.100.0.10`*.

  ```
  aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
   --output text --name my-cluster --region region-code
  ```

- Contoh ini memberikan `kubelet` argumen untuk menetapkan `max-pods` nilai kustom menggunakan `bootstrap.sh` skrip yang disertakan dengan AMI Amazon EKS yang

dioptimalkan. Untuk bantuan dalam memilih*my-max-pods-value*, lihatAmazon EKS merekomendasikan maksimum Pods untuk setiap jenis instans Amazon EC2.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
set -ex
/etc/eks/bootstrap.sh my-cluster \
  --b64-cluster-ca certificate-authority \
  --apiserver-endpoint api-server-endpoint \
  --dns-cluster-ip service-cidr.10 \
  --kubelet-extra-args '--max-pods=my-max-pods-value' \
  --use-max-pods false

--==MYBOUNDARY==--
```

Berikan data pengguna untuk meneruskan argumen ke **`Start-EKSBootstrap.ps1`** file yang disertakan dengan Windows AMI Amazon EKS yang dioptimalkan

Bootstrapping adalah istilah yang digunakan untuk menggambarkan penambahan perintah yang dapat dijalankan ketika sebuah instance dimulai. Anda dapat meneruskan argumen ke `Start-EKSBootstrap.ps1` skrip dengan menggunakan `eksctl` tanpa menentukan template peluncuran. Atau Anda dapat melakukannya dengan menentukan informasi di bagian data pengguna dari template peluncuran.

Jika Anda ingin menentukan ID Windows AMI kustom, ingatlah pertimbangan berikut:

- Anda harus menggunakan template peluncuran dan memberikan perintah bootstrap yang diperlukan di bagian data pengguna. Untuk mengambil Windows ID yang Anda inginkan, Anda dapat menggunakan tabel diAmazon EKS dioptimalkan Windows AMI.
- Ada beberapa batasan dan kondisi. Misalnya, Anda harus menambahkan `eks:kube-proxy-windows` ke peta konfigurasi AWS IAM Authenticator Anda. Untuk informasi selengkapnya, lihat Batas dan ketentuan saat menentukan ID AMI.

Tentukan informasi berikut di bagian data pengguna dari template peluncuran Anda. Ganti setiap *example value* dengan nilai-nilai Anda sendiri. `-DNSClusterIP`Argumen `-APIServerEndpoint-Base64ClusterCA`,, dan bersifat opsional. Namun, mendefinisikannya memungkinkan `Start-EKSBootstrap.ps1` skrip untuk menghindari `describeCluster` panggilan.

- Satu-satunya argumen yang diperlukan adalah nama cluster (*my-cluster*).
- Untuk mengambil *certificate-authority* untuk cluster Anda, jalankan perintah berikut.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text --
name my-cluster --region region-code
```

- Untuk mengambil *api-server-endpoint* untuk cluster Anda, jalankan perintah berikut.

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

- Nilai untuk `--dns-cluster-ip` adalah CIDR layanan Anda dengan `.10` di akhir. Untuk mengambil *service-cidr* untuk cluster Anda, jalankan perintah berikut. Misalnya, jika nilai yang dikembalikan adalah`ipv4 10.100.0.0/16`, maka nilai Anda adalah*10.100.0.10*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr" --
output text --name my-cluster --region region-code
```

- Untuk argumen tambahan, lihatParameter konfigurasi skrip bootstrap.

> **Note**
>
> Jika Anda menggunakan layanan khusus CIDR, maka Anda perlu menentukannya menggunakan `-ServiceCIDR` parameter. Jika tidak, resolusi DNS untuk Pods di cluster akan gagal.

```
<powershell>
[string]$EKSBootstrapScriptFile = "$env:ProgramFiles\Amazon\EKS\Start-EKSBootstrap.ps1"
& $EKSBootstrapScriptFile -EKSClusterName my-cluster `
  -Base64ClusterCA certificate-authority `
  -APIServerEndpoint api-server-endpoint `
  -DNSClusterIP service-cidr.10
```

```
</powershell>
```

Jalankan AMI khusus karena persyaratan keamanan, kepatuhan, atau kebijakan internal tertentu

Untuk informasi selengkapnya, lihat Amazon Machine Images (AMI) di Panduan Pengguna Amazon EC2 untuk Instans Linux. Spesifikasi build Amazon EKS AMI berisi sumber daya dan skrip konfigurasi untuk membuat Amazon EKS AMI khusus berdasarkan Amazon Linux. Untuk informasi selengkapnya, lihat Amazon EKS AMI Build Specification onGitHub. Untuk membuat AMI kustom yang diinstal dengan sistem operasi lain, lihat Amazon EKS Contoh AMI Kustom aktifGitHub.

> **Important**
>
> Saat menentukan AMI, Amazon EKS tidak menggabungkan data pengguna apa pun. Sebaliknya, Anda bertanggung jawab untuk menyediakan `bootstrap` perintah yang diperlukan untuk node untuk bergabung dengan cluster. Jika simpul Anda gagal untuk bergabung dengan klaster, tindakan Amazon EKS `CreateNodegroup` dan `UpdateNodegroupVersion` juga gagal.

Batas dan ketentuan saat menentukan ID AMI

Berikut ini adalah batasan dan kondisi yang terkait dengan menentukan ID AMI dengan grup node terkelola:

- Anda harus membuat grup node baru untuk beralih antara menentukan ID AMI dalam template peluncuran dan tidak menentukan ID AMI.
- Anda tidak diberi tahu di konsol saat versi AMI yang lebih baru tersedia. Untuk memperbarui grup node ke versi AMI yang lebih baru, Anda perlu membuat versi baru template peluncuran dengan ID AMI yang diperbarui. Kemudian, Anda perlu memperbarui grup node dengan versi template peluncuran baru.
- Bidang berikut tidak dapat disetel di API jika Anda menentukan ID AMI:
  - `amiType`
  - `releaseVersion`
  - `version`
- Setiap `taints` set dalam API diterapkan secara asinkron jika Anda menentukan ID AMI. Untuk menerapkan taints sebelum node bergabung dengan cluster, Anda harus meneruskan taints ke

`kubelet` dalam data pengguna Anda menggunakan flag baris `--register-with-taints` perintah. Untuk informasi lebih lanjut, lihat `kubelet`di Kubernetes dokumentasi.

- Saat menentukan ID AMI khusus untuk grup node Windows terkelola, tambahkan `eks:kube-proxy-windows` ke peta konfigurasi AWS IAM Authenticator Anda. Ini diperlukan agar DNS berfungsi dengan baik.

  1. Buka peta konfigurasi AWS IAM Authenticator untuk mengedit.

     ```
     kubectl edit -n kube-system cm aws-auth
     ```

  2. Tambahkan entri ini ke `groups` daftar di bawah masing-masing `rolearn` yang terkait dengan Windows node. Peta konfigurasi Anda akan terlihat mirip dengan `aws-auth-cm-windows.yaml`.

     ```
     - eks:kube-proxy-windows
     ```

  3. Simpan file, dan tutup editor teks Anda.

Menghapus grup simpul terkelola

Topik ini menjelaskan cara menghapus grup simpul terkelola Amazon EKS. Saat Anda menghapus grup node terkelola, Amazon EKS pertama-tama menetapkan ukuran minimum, maksimum, dan yang diinginkan dari grup Auto Scaling Anda ke nol. Ini kemudian menyebabkan grup node Anda menurunkan skala.

Sebelum setiap instance dihentikan, Amazon EKS mengirimkan sinyal untuk menguras Pods dari node itu. Jika Pods tidak terkuras setelah beberapa menit, Amazon EKS memungkinkan Auto Scaling melanjutkan penghentian instance. Setelah setiap instance dihentikan, grup Auto Scaling akan dihapus.

> **Important**
>
> Jika Anda menghapus grup node terkelola yang menggunakan peran IAM node yang tidak digunakan oleh grup node terkelola lainnya di cluster, peran tersebut akan dihapus dari. `aws-auth ConfigMap` Jika salah satu grup node yang dikelola sendiri di cluster menggunakan peran IAM node yang sama, node yang dikelola sendiri pindah ke status. `NotReady` Selain itu, operasi cluster juga terganggu. Untuk menambahkan pemetaan untuk peran yang Anda gunakan hanya untuk grup node yang dikelola sendiri, lihatMembuat

entri akses, apakah versi platform klaster Anda setidaknya merupakan versi minimum yang tercantum di bagian prasyarat. Kelola entri akses Jika versi platform Anda lebih awal dari versi minimum yang diperlukan untuk entri akses, Anda dapat menambahkan entri kembali ke `aws-authConfigMap`. Untuk informasi lebih lanjut, masukkan `eksctl create iamidentitymapping --help` di terminal Anda.

Anda dapat menghapus grup node terkelola dengan `eksctl` atau AWS Management Console.

eksctl

Untuk menghapus grup node terkelola dengan **`eksctl`**

Masukkan perintah berikut. Ganti setiap *`example value`* dengan nilai-nilai Anda sendiri.

```
eksctl delete nodegroup \
  --cluster my-cluster \
  --name my-mng \
  --region region-code
```

Untuk opsi lainnya, lihat Menghapus dan menguras nodegroup dalam dokumentasi. `eksctl`

AWS Management Console

Untuk menghapus grup node terkelola Anda dengan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pada halaman Clusters, pilih cluster yang berisi grup node untuk dihapus.
3. Pada halaman cluster yang dipilih, pilih tab Compute.
4. Di bagian Node groups, pilih grup node yang akan dihapus. Lalu pilih Hapus.
5. Dalam kotak dialog Hapus konfirmasi grup simpul, masukkan nama grup simpul. Lalu pilih Hapus.

AWS CLI

Untuk menghapus grup node terkelola Anda dengan AWS CLI

1. Masukkan perintah berikut. Ganti setiap *`example value`* dengan nilai-nilai Anda sendiri.

```
aws eks delete-nodegroup \
```

```
--cluster-name my-cluster \
--nodegroup-name my-mng \
--region region-code
```

2. Gunakan tombol panah pada keyboard Anda untuk menggulir output respons. Tekan **q** tombol ketika Anda selesai.

Untuk opsi lainnya, lihat `delete-nodegroup` perintah di Referensi AWS CLI Perintah.

Simpul yang dikelola sendiri

Cluster berisi satu atau lebih node Amazon EC2 yang dijadwalkan Pods aktif. Node Amazon EKS berjalan di AWS akun Anda dan terhubung ke bidang kontrol klaster Anda melalui titik akhir server API cluster. Anda ditagih untuk mereka berdasarkan harga Amazon EC2. Untuk informasi lebih lanjut, lihat Harga Amazon EC2.

Sebuah klaster dapat berisi beberapa grup simpul. Setiap grup node berisi satu atau lebih node yang digunakan dalam Amazon EC2 Auto Scalinggrup. Jenis instance node dalam grup dapat bervariasi, seperti saat menggunakan pemilihan tipe instance berbasis atribut dengan. Karpenter Semua instance dalam grup node harus menggunakan peran IAM node Amazon EKS.

Amazon EKS menyediakan Amazon Machine Images (AMI) khusus yang disebut AMI yang dioptimalkan Amazon EKS. AMI dikonfigurasi untuk bekerja dengan Amazon EKS. Komponen mereka termasuk`containerd`,`kubelet`, dan AWS IAM Authenticator. AMI juga berisi bootstrap script khusus yang memungkinkannya untuk menemukan dan terhubung ke bidang kendali klaster Anda secara otomatis.

Jika Anda membatasi akses ke titik akhir publik klaster Anda menggunakan blok CIDR, sebaiknya Anda juga mengaktifkan akses titik akhir pribadi. Ini agar node dapat berkomunikasi dengan cluster. Jika titik akhir privat diaktifkan, blok CIDR yang Anda tentukan untuk akses publik harus menyertakan sumber jalan keluar dari VPC Anda. Untuk informasi selengkapnya, lihat Kendali akses titik akhir klaster Amazon EKS.

Untuk menambahkan simpul-simpul yang dikelola sendiri untuk klaster Amazon EKS Anda, lihat topik-topik berikutnya. Jika Anda meluncurkan node yang dikelola sendiri secara manual, tambahkan tag berikut ke setiap node. Untuk informasi selengkapnya, lihat Penambahan dan penghapusan tanda pada sumber daya individu. Jika Anda mengikuti langkah-langkah dalam panduan berikut, tag yang diperlukan secara otomatis ditambahkan ke node untuk Anda.

Kunci	Nilai
`kubernetes.io/cluster/ my-cluster`	`owned`

Untuk informasi selengkapnya tentang node dari Kubernetes perspektif umum, lihat Node dalam Kubernetes dokumentasi.

Topik

- Meluncurkan simpul Amazon Linux yang dikelola sendiri
- Meluncurkan node yang dikelola sendiri Bottlerocket
- Meluncurkan node yang dikelola sendiri Windows
- Pembaruan simpul yang dikelola sendiri

Meluncurkan simpul Amazon Linux yang dikelola sendiri

Topik ini menjelaskan bagaimana Anda dapat meluncurkan grup Linux node Auto Scaling yang mendaftar dengan kluster Amazon EKS Anda. Setelah node bergabung dengan cluster, Anda dapat menyebarkan Kubernetes aplikasi ke mereka. Anda juga dapat meluncurkan node Amazon Linux yang dikelola sendiri dengan `eksctl` atau. AWS Management Console Jika Anda perlu meluncurkan node AWS Outposts, lihatMeluncurkan node Amazon Linux yang dikelola sendiri di Outpost.

Prasyarat

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMembuat klaster Amazon EKS. Jika Anda memiliki subnet di Wilayah AWS mana Anda memiliki AWS Outposts, AWS Wavelength, atau AWS Local Zones diaktifkan, subnet tersebut harus tidak diteruskan saat Anda membuat klaster Anda.
- Peran IAM yang ada untuk digunakan node. Untuk membuatnya, lihat IAM role simpul Amazon EKS. Jika peran ini tidak memiliki salah satu kebijakan untuk VPC CNI, peran terpisah yang mengikuti diperlukan untuk pod VPC CNI.
- (Opsional, tetapi disarankan) Amazon VPC CNI plugin for Kubernetes Add-on dikonfigurasi dengan peran IAM sendiri yang memiliki kebijakan IAM yang diperlukan yang melekat padanya. Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).

- Keakraban dengan pertimbangan yang tercantum dalam. Memilih jenis instans Amazon EC2 Bergantung pada jenis instans yang Anda pilih, mungkin ada prasyarat tambahan untuk cluster dan VPC Anda.

eksctl

> **Note**
>
> `eksctl`tidak mendukung Amazon Linux 2023 saat ini.

Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Untuk meluncurkan Linux node yang dikelola sendiri menggunakan **`eksctl`**

1. (Opsional) Jika kebijakan IAM terkelola Amazoneks_CNI_Policy dilampirkan pada kebijakan IAM Anda, kami sarankan untuk menetapkannya ke peran IAM yang IAM role simpul Amazon EKS Anda kaitkan ke akun layanan sebagai gantinya. Kubernetes `aws-node` Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).
2. Perintah berikutnya membuat grup simpul dalam klaster yang ada. Ganti *`al-nodes`* dengan nama untuk grup node Anda. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa. Ganti *`my-cluster`* dengan nama klaster Anda. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter. Ganti sisanya *`example value`* dengan nilai Anda sendiri. Node dibuat dengan Kubernetes versi yang sama dengan bidang kontrol, secara default.

   Sebelum memilih nilai untuk`--node-type`, tinjauMemilih jenis instans Amazon EC2.

   Ganti *`my-key`* dengan nama pasangan kunci atau kunci publik Amazon EC2 Anda. Kunci ini digunakan untuk SSH ke simpul Anda setelah diluncurkan. Jika Anda belum memiliki pasangan kunci Amazon EC2, Anda dapat membuatnya di AWS Management Console.

Untuk informasi selengkapnya, lihat Kunci Pasangan Amazon EC2 dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

Buat grup simpul Anda dengan perintah berikut.

> **Important**
>
> Jika Anda ingin menyebarkan grup node ke, Wavelength AWS Outposts, atau subnet Zona Lokal, ada pertimbangan tambahan:
>
> - Subnet tidak boleh diteruskan saat Anda membuat cluster.
> - Anda harus membuat grup node dengan file konfigurasi yang menentukan subnet dan. `volumeType: gp2` Untuk informasi selengkapnya, lihat Membuat nodegroup dari file konfigurasi dan skema file Config dalam dokumentasi. `eksctl`

```
eksctl create nodegroup \
  --cluster my-cluster \
  --name al-nodes \
  --node-type t3.medium \
  --nodes 3 \
  --nodes-min 1 \
  --nodes-max 4 \
  --ssh-access \
  --managed=false \
  --ssh-public-key my-key
```

Untuk menyebarkan grup simpul yang:

- dapat menetapkan jumlah alamat IP yang jauh lebih tinggi Pods daripada konfigurasi default, lihatTingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda.
- dapat menetapkan `IPv4` alamat Pods dari CIDR blok yang berbeda dari contoh, lihatJaringan khusus untuk pod.
- dapat menetapkan `IPv6` alamat Pods dan layanan, lihat`IPv6`alamat untuk cluster,Pods, dan services.
- menggunakan `containerd` runtime, Anda harus menyebarkan grup node menggunakan file. `config` Untuk informasi selengkapnya, lihat Uji migrasi dari Docker ke `containerd`.
- tidak memiliki akses internet outbound, lihatPersyaratan klaster pribadi.

Untuk daftar lengkap terkait semua opsi dan default yang tersedia, masukkan perintah berikutnya.

```
eksctl create nodegroup --help
```

Jika simpul gagal bergabung dengan klaster, lihat Simpul gagal untuk bergabung dengan klaster dalam Panduan pemecahan masalah.

Contoh output adalah sebagai berikut. Beberapa baris adalah output sementara node dibuat. Salah satu baris terakhir dari output adalah baris contoh berikutnya.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Opsional) Menyebarkan aplikasi sampel untuk menguji cluster dan Linux node Anda.
4. Kami merekomendasikan memblokir Pod akses ke IMDS jika kondisi berikut benar:

   - Anda berencana untuk menetapkan peran IAM ke semua akun Kubernetes layanan Anda sehingga Pods hanya memiliki izin minimum yang mereka butuhkan.
   - Tidak ada Pods di klaster yang memerlukan akses ke layanan metadata instans Amazon EC2 (IMDS) karena alasan lain, seperti mengambil arus. Wilayah AWS

   Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

AWS Management Console

Langkah 1: Untuk meluncurkan Linux node yang dikelola sendiri menggunakan AWS Management Console

1. Unduh versi terbaru dari AWS CloudFormation template.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/amazon-eks-nodegroup.yaml
   ```

2. Tunggu status klaster Anda ditampilkan sebagai `ACTIVE`. Jika Anda meluncurkan node Anda sebelum cluster aktif, node gagal mendaftar dengan cluster dan Anda harus meluncurkannya kembali.

3. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
4. Pilih Buat tumpukan dan kemudian pilih Dengan sumber daya baru (standar).
5. Untuk Menentukan templat, pilih Unggah sebuah file templat dan kemudian pilih Pilih file.
6. Pilih `amazon-eks-nodegroup.yaml` file yang Anda unduh.
7. Pilih Selanjutnya.
8. Pada halaman Tentukan detail tumpukan, masukkan parameter berikut yang sesuai, lalu pilih Berikutnya:

   - Nama tumpukan: Pilih nama tumpukan untuk tumpukan AWS CloudFormation Anda. Misalnya, Anda bisa menyebutnya***`my-cluster-nodes`***. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter.
   - ClusterName: Masukkan nama yang Anda gunakan saat membuat cluster Amazon EKS Anda. Nama ini harus sama dengan nama cluster atau node Anda tidak dapat bergabung dengan cluster.
   - ClusterControlPlaneSecurityGroup: Pilih SecurityGroupsnilai dari AWS CloudFormation output yang Anda hasilkan saat Anda membuat VPC Anda.

     Langkah-langkah berikut menunjukkan satu operasi untuk mengambil grup yang berlaku.

     1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
     2. Pilih nama cluster.
     3. Pilih tab Jaringan.
     4. Gunakan nilai grup keamanan tambahan sebagai referensi saat memilih dari daftar ClusterControlPlaneSecurityGrouptarik-turun.
   - NodeGroupName: Masukkan nama untuk grup node Anda. Nama ini dapat digunakan nanti untuk mengidentifikasi grup node Auto Scaling yang dibuat untuk node Anda. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa.
   - NodeAutoScalingGroupMinSize: Masukkan jumlah minimum node yang dapat diskalakan oleh grup Auto Scaling node Anda.
   - NodeAutoScalingGroupDesiredCapacity: Masukkan jumlah node yang diinginkan untuk diskalakan saat tumpukan Anda dibuat.
   - NodeAutoScalingGroupMaxSize: Masukkan jumlah maksimum node yang dapat diskalakan oleh grup Auto Scaling node Anda.

- NodeInstanceType: Pilih jenis instance untuk node Anda. Untuk informasi selengkapnya, lihat Memilih jenis instans Amazon EC2.
- NodeImageIdSSMParam: Diisi sebelumnya dengan parameter Amazon EC2 Systems Manager dari AMI Amazon EKS yang dioptimalkan baru-baru ini untuk versi variabel. Kubernetes Untuk menggunakan versi Kubernetes minor berbeda yang didukung dengan Amazon EKS, ganti *1.XX* dengan versi lain yang didukung. Sebaiknya tentukan Kubernetes versi yang sama dengan cluster Anda.

  Anda juga dapat mengganti *amazon-linux-2* dengan tipe AMI yang berbeda. Untuk informasi selengkapnya, lihat Mendapatkan ID Amazon Linux AMI yang dioptimalkan Amazon EKS.

  > **Note**
  >
  > Node Amazon EKS AMI didasarkan pada Amazon Linux. Anda dapat melacak peristiwa keamanan atau privasi untuk Amazon Linux 2 di Pusat Keamanan Amazon Linux atau berlangganan ke Umpan RSS yang terkait. Kejadian keamanan dan privasi mencakup gambaran umum mengenai masalah, paket apa yang terpengaruh, dan cara memperbarui instans Anda untuk memperbaiki masalah tersebut.

- NodeImageId: (Opsional) Jika Anda menggunakan AMI kustom Anda sendiri (bukan AMI yang dioptimalkan Amazon EKS), masukkan ID AMI node untuk Anda Wilayah AWS. Jika Anda menentukan nilai di sini, itu akan mengganti nilai apa pun di bidang NodeImageIdSSMParam.
- NodeVolumeSize: Tentukan ukuran volume root untuk node Anda, di GiB.
- NodeVolumeType: Tentukan jenis volume root untuk node Anda.
- KeyName: Masukkan nama key pair Amazon EC2 SSH yang dapat Anda gunakan untuk terhubung menggunakan SSH ke node Anda setelah diluncurkan. Jika Anda belum memiliki pasangan kunci Amazon EC2, maka Anda dapat membuatnya di AWS Management Console. Untuk informasi selengkapnya, lihat Pasangan kunci Amazon EC2 dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

> **Note**
>
> Jika Anda tidak menyediakan key pair di sini, pembuatan AWS CloudFormation stack gagal.

- BootstrapArguments: Tentukan argumen opsional apa pun untuk diteruskan ke skrip bootstrap node, seperti `kubelet` argumen tambahan. Untuk informasi lebih lanjut, lihat informasi penggunaan skrip bootstrap diGitHub.

  Untuk menyebarkan grup simpul yang:

  - dapat menetapkan jumlah alamat IP yang jauh lebih tinggi Pods daripada konfigurasi default, lihatTingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda.
  - dapat menetapkan `IPv4` alamat Pods dari CIDR blok yang berbeda dari contoh, lihatJaringan khusus untuk pod.
  - dapat menetapkan `IPv6` alamat Pods dan layanan, lihat`IPv6`alamat untuk cluster,Pods, dan services.
  - menggunakan `containerd` runtime, Anda harus menyebarkan grup node menggunakan file. `config` Untuk informasi selengkapnya, lihat Uji migrasi dari Docker ke `containerd`.
  - tidak memiliki akses internet outbound, lihatPersyaratan klaster pribadi.
- DisableImDSv1: Secara default, setiap node mendukung Instance Metadata Service Version 1 (IMDSv1) dan IMDSv2. Anda dapat menonaktifkan IMDSv1. Untuk mencegah node future dan Pods dalam grup node menggunakan mDSv1, atur disableImDSv1 ke true. Untuk informasi selengkapnya tentang IMDS, lihat Mengonfigurasi layanan metadata instans. Untuk informasi selengkapnya tentang membatasi akses ke node Anda, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.
- VpcId: Masukkan ID untuk VPC yang Anda buat.
- Subnet: Pilih subnet yang sudah Anda buat untuk VPC Anda. Jika Anda membuat VPC menggunakan langkah-langkah yang dijelaskan dalamMembuat VPC untuk klaster Amazon EKS Anda, tentukan hanya subnet pribadi dalam VPC untuk node Anda untuk diluncurkan. Anda dapat melihat subnet mana yang bersifat pribadi dengan membuka setiap subnet link dari tab Networking cluster Anda.

> **Important**
>
> - Jika salah satu subnet adalah subnet publik, maka mereka harus mengaktifkan pengaturan tugas alamat IP publik otomatis. Jika pengaturan tidak diaktifkan untuk subnet publik, maka node apa pun yang Anda terapkan ke subnet publik tersebut tidak akan diberi alamat IP publik dan tidak akan dapat berkomunikasi dengan cluster atau layanan lainnya. AWS Jika subnet digunakan sebelum 26 Maret 2020 menggunakan salah satu templat AWS CloudFormation VPC Amazon EKS, atau dengan menggunakan`eksctl`, maka penetapan alamat IP publik otomatis dinonaktifkan untuk subnet publik. Untuk informasi tentang cara mengaktifkan penetapan alamat IP publik untuk subnet, lihat Memodifikasi atribut `IPv4` pengalamatan publik untuk subnet Anda. Jika node dikerahkan ke subnet pribadi, maka node dapat berkomunikasi dengan cluster dan AWS layanan lainnya melalui gateway NAT.
> - Jika subnet tidak memiliki akses internet, pastikan Anda mengetahui pertimbangan dan langkah-langkah tambahan. Persyaratan klaster pribadi
> - Jika Anda memilih AWS Outposts subnet Wavelength, atau Local Zone, subnet tidak boleh diteruskan saat Anda membuat cluster.

9. Pilih pilihan yang Anda inginkan di halaman Configure stack options, lalu pilih Next.
10. Pilih kotak centang di sebelah kiri Saya mengakui yang AWS CloudFormation mungkin membuat sumber daya IAM. , dan kemudian pilih Buat tumpukan.
11. Setelah tumpukan Anda selesai dibuat, pilih tumpukan di konsol dan pilih Outputs.
12. Rekam NodeInstanceRoleuntuk grup node yang telah dibuat. Anda memerlukan ini saat mengonfigurasi simpul Amazon EKS Anda.

Langkah 2: Untuk mengaktifkan node untuk bergabung dengan cluster Anda

> **Note**
>
> Jika Anda meluncurkan node di dalam VPC pribadi tanpa akses internet keluar, pastikan untuk mengaktifkan node untuk bergabung dengan cluster Anda dari dalam VPC.

1. Periksa untuk melihat apakah Anda sudah memiliki `aws-authConfigMap`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Jika Anda ditampilkan `aws-auth`ConfigMap, maka perbarui sesuai kebutuhan.

   a. Buka `ConfigMap` untuk mengedit.

   ```
   kubectl edit -n kube-system configmap/aws-auth
   ```

   b. Tambahkan `mapRoles` entri baru sesuai kebutuhan. Tetapkan `rolearn` nilai ke NodeInstanceRolenilai yang Anda rekam dalam prosedur sebelumnya.

   ```
   [...]
   data:
     mapRoles: |
       - rolearn: <ARN of instance role (not instance profile)>
         username: system:node:{{EC2PrivateDNSName}}
         groups:
           - system:bootstrappers
           - system:nodes
   [...]
   ```

   c. Simpan file dan keluar dari editor teks Anda.

3. Jika Anda menerima kesalahan yang menyatakan "`Error from server (NotFound): configmaps "aws-auth" not found`, maka terapkan stok`ConfigMap`.

   a. Unduh peta konfigurasi.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/aws-auth-cm.yaml
   ```

   b. Dalam `aws-auth-cm.yaml` file, atur `rolearn` nilai ke NodeInstanceRolenilai yang Anda rekam dalam prosedur sebelumnya. Anda dapat melakukan ini dengan editor teks, atau dengan mengganti *`my-node-instance-role`* dan menjalankan perintah berikut:

   ```
   sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-
   instance-role|' aws-auth-cm.yaml
   ```

   c. Terapkan konfigurasi. Perintah ini mungkin memerlukan waktu beberapa menit untuk diselesaikan.

```
kubectl apply -f aws-auth-cm.yaml
```

4. Perhatikan status simpul Anda dan tunggu sampai simpul mencapai Status `Ready`.

   ```
   kubectl get nodes --watch
   ```

   Masukkan `Ctrl+C` untuk kembali ke prompt shell.

   > **Note**
   >
   > Jika Anda menerima kesalahan otorisasi atau jenis sumber daya, lihat Tidak sah atau akses ditolak (`kubectl`) di topik pemecahan masalah.

   Jika simpul gagal bergabung dengan klaster, lihat Simpul gagal untuk bergabung dengan klaster dalam Panduan pemecahan masalah.

5. (Hanya node GPU) Jika Anda memilih jenis instans GPU dan AMI akselerasi Amazon EKS yang dioptimalkan, Anda harus menerapkan plugin perangkat NVIDIA untuk Kubernetes sebagai DaemonSet di cluster Anda. Ganti *`vX.X.X`* dengan s-device-plugin versi NVIDIA/K8 yang Anda inginkan sebelum menjalankan perintah berikut.

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

Langkah 3: Tindakan tambahan

1. (Opsional) Menyebarkan aplikasi sampel untuk menguji cluster dan Linux node Anda.
2. (Opsional) Jika kebijakan IAM terkelola Amazoneks_CNI_Policy (jika Anda memiliki klaster) atau *`AmazonEKS_CNI_IPv6_Policy`* (yang Anda buat sendiri jika Anda `IPv4` memiliki klaster) dilampirkan ke Anda, kami sarankan untuk menetapkannya ke peran IAM `IPv6` yang Anda kaitkan ke the section called “IAM role simpul” akun layanan sebagai gantinya. Kubernetes `aws-node` Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).
3. Kami merekomendasikan memblokir Pod akses ke IMDS jika kondisi berikut benar:

- Anda berencana untuk menetapkan peran IAM ke semua akun Kubernetes layanan Anda sehingga Pods hanya memiliki izin minimum yang mereka butuhkan.
- Tidak ada Pods di klaster yang memerlukan akses ke layanan metadata instans Amazon EC2 (IMDS) karena alasan lain, seperti mengambil arus. Wilayah AWS

Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

Blok Kapasitas untuk ML

> **Important**
>
> Fitur ini saat ini hanya tersedia untuk instans P5 di AS Timur (Ohio) dan AS Timur (Virginia N.) dan P4d di AS Timur (Ohio) Wilayah AWS dan AS Barat (Oregon). Wilayah AWS

Blok Kapasitas untuk pembelajaran mesin (ML) memungkinkan Anda untuk memesan instans GPU di masa mendatang untuk mendukung beban kerja MS berdurasi pendek Anda. Instans yang berjalan di dalam Blok Kapasitas secara otomatis ditempatkan berdekatan di dalam Amazon UltraClusters EC2, jadi tidak perlu menggunakan grup penempatan klaster. Untuk informasi selengkapnya, lihat Blok Kapasitas untuk ML di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Anda dapat menggunakan Blok Kapasitas dengan Amazon EKS untuk menyediakan dan menskalakan node yang dikelola sendiri. Langkah-langkah berikut memberikan gambaran umum contoh.

1. Buat template peluncuran di file AWS Management Console. Untuk informasi selengkapnya, lihat Menggunakan Blok Kapasitas untuk beban kerja pembelajaran mesin di Panduan Pengguna Auto Scaling Amazon EC2.

   Pastikan untuk menyertakan konfigurasi jenis instans dan Amazon Machine Image (AMI).
2. Tautkan Blok Kapasitas ke templat peluncuran menggunakan ID reservasi kapasitas.

   Berikut ini adalah contoh AWS CloudFormation template untuk membuat template peluncuran yang menargetkan Blok Kapasitas:

   ```
   NodeLaunchTemplate:
   ```

```
    Type: "AWS::EC2::LaunchTemplate"
    Properties:
      LaunchTemplateData:
        InstanceMarketOptions:
          MarketType: "capacity-block"
        CapacityReservationSpecification:
          CapacityReservationTarget:
            CapacityReservationId: "cr-02168da1478b509e0"
        IamInstanceProfile:
          Arn: iam-instance-profile-arn
        ImageId: image-id
        InstanceType: p5.48xlarge
        KeyName: key-name
        SecurityGroupIds:
        - sg-05b1d815d1EXAMPLE
        UserData: user-data
```

Anda harus melewati subnet di Availability Zone di mana reservasi dilakukan karena Blok Kapasitas bersifat zonal.

3. Jika Anda membuat grup node yang dikelola sendiri sebelum reservasi kapasitas menjadi aktif, maka atur kapasitas yang diinginkan`0`. Saat membuat grup node, pastikan bahwa Anda hanya menentukan subnet masing-masing untuk Availability Zone di mana kapasitas dicadangkan.

   Berikut ini adalah contoh CloudFormation template yang dapat digunakan.
   Contoh ini mendapatkan `LaunchTemplateId` dan `Version` dari `AWS::Amazon EC2::LaunchTemplate` sumber daya yang ditunjukkan pada contoh sebelumnya. Itu juga mendapatkan nilai untuk`DesiredCapacity`,, `MaxSizeMinSize`, dan `VPCZoneIdentifier` yang dideklarasikan di tempat lain dalam template yang sama.

```
NodeGroup:
  Type: "AWS::AutoScaling::AutoScalingGroup"
  Properties:
    DesiredCapacity: !Ref NodeAutoScalingGroupDesiredCapacity
    LaunchTemplate:
      LaunchTemplateId: !Ref NodeLaunchTemplate
      Version: !GetAtt NodeLaunchTemplate.LatestVersionNumber
    MaxSize: !Ref NodeAutoScalingGroupMaxSize
    MinSize: !Ref NodeAutoScalingGroupMinSize
    VPCZoneIdentifier: !Ref Subnets
    Tags:
      - Key: Name
```

```
        PropagateAtLaunch: true
        Value: !Sub ${ClusterName}-${NodeGroupName}-Node
      - Key: !Sub kubernetes.io/cluster/${ClusterName}
        PropagateAtLaunch: true
        Value: owned
```

4. Setelah grup node berhasil dibuat, pastikan untuk merekam `NodeInstanceRole` untuk grup node yang dibuat. Anda memerlukan ini untuk memastikan bahwa ketika grup node diskalakan, node baru bergabung dengan cluster dan Kubernetes mampu mengenali node. Untuk informasi lebih lanjut, lihat AWS Management Console instruksi diMeluncurkan simpul Amazon Linux yang dikelola sendiri.
5. Kami menyarankan Anda membuat kebijakan penskalaan terjadwal untuk grup Auto Scaling yang sejajar dengan waktu reservasi Blok Kapasitas. Untuk informasi selengkapnya, lihat Penskalaan terjadwal untuk Auto Scaling Amazon EC2 di Panduan Pengguna Auto Scaling Amazon EC2.

   Anda dapat menggunakan semua instans yang Anda pesan hingga 30 menit sebelum waktu akhir Blok Kapasitas. Contoh yang masih berjalan pada saat itu akan mulai berakhir. Untuk memberikan waktu yang cukup untuk menguras node dengan baik, kami sarankan Anda menjadwalkan penskalaan ke skala nol lebih dari 30 menit sebelum waktu akhir reservasi Blok Kapasitas.

   Jika Anda ingin meningkatkan secara manual setiap kali reservasi kapasitas terjadi`Active`, maka Anda perlu memperbarui kapasitas yang diinginkan grup Auto Scaling pada waktu mulai reservasi Blok Kapasitas. Kemudian Anda juga perlu menurunkan skala secara manual lebih dari 30 menit sebelum waktu akhir reservasi Blok Kapasitas.
6. Grup node sekarang siap untuk beban kerja dan Pods dijadwalkan.
7. Agar Anda Pods terkuras dengan baik, kami sarankan Anda mengatur AWS Node Termination Handler. Handler ini akan dapat melihat peristiwa siklus hidup “ASG Scale-in” dari Amazon EC2 Auto EventBridge Scaling menggunakan dan Kubernetes mengizinkan bidang kontrol untuk mengambil tindakan yang diperlukan sebelum instance menjadi tidak tersedia. Jika tidak, Anda Pods dan Kubernetes objek akan terjebak dalam keadaan tertunda. Untuk informasi selengkapnya, lihat AWS Node Termination Handler di GitHub.

   Jika Anda tidak menyiapkan Node Termination Handler, kami sarankan Anda mulai menguras Pods secara manual sebelum menekan jendela 30 menit sehingga mereka memiliki cukup waktu untuk dikeringkan dengan anggun.

Meluncurkan node yang dikelola sendiri Bottlerocket

> **Note**
>
> Grup node terkelola mungkin menawarkan beberapa keuntungan untuk kasus penggunaan Anda. Untuk informasi selengkapnya, lihat Grup simpul terkelola.

Topik ini menjelaskan cara meluncurkan grup Auto Scaling dari node Bottlerocket yang mendaftar dengan cluster Amazon EKS Anda. Bottlerocketadalah sistem operasi open-source Linux berbasis AWS yang dapat Anda gunakan untuk menjalankan kontainer pada mesin virtual atau host bare metal. Setelah node bergabung dengan cluster, Anda dapat menyebarkan Kubernetes aplikasi ke mereka. Untuk informasi selengkapnyaBottlerocket, lihat Menggunakan Bottlerocket AMI dengan Amazon EKS GitHub aktif dan dukungan AMI khusus dalam `eksctl` dokumentasi.

Untuk informasi tentang peningkatan di tempat, lihat BottlerocketMemperbarui Operator di. GitHub

> **Important**
>
> - Simpul Amazon EKS adalah instans Amazon EC2 standar, dan Anda dikenakan biaya berdasarkan harga instans Amazon EC2 normal. Untuk informasi selengkapnya, lihat Harga Amazon EC2.
> - Anda dapat meluncurkan node Bottlerocket di Amazon EKS cluster yang diperluas di AWS Outposts, tetapi Anda tidak dapat meluncurkannya di cluster lokal di Outposts. AWS Untuk informasi selengkapnya, lihat Amazon EKS pada AWS Outposts.
> - Anda dapat menerapkan instans x86 Amazon EC2 dengan atau prosesor. Arm Namun, Anda tidak dapat menyebarkan ke instance yang memiliki Inferentia chip.
> - Bottlerocketkompatibel dengan AWS CloudFormation. Namun, tidak ada CloudFormation template resmi yang dapat disalin untuk menyebarkan Bottlerocket node untuk Amazon EKS.
> - Bottlerocketgambar tidak datang dengan SSH server atau shell. Anda dapat menggunakan metode out-of-band akses untuk SSH memungkinkan mengaktifkan wadah admin dan meneruskan beberapa langkah konfigurasi bootstrap dengan data pengguna. Untuk informasi lebih lanjut, lihat bagian ini di bottlerocket README.md di: GitHub
>   - Eksplorasi
>   - Kontainer admin

- Kubernetespengaturan

Untuk meluncurkan Bottlerocket node menggunakan **`eksctl`**

Prosedur ini membutuhkan `eksctl` versi `0.175.0` atau yang lebih baru. Anda dapat memeriksa versi Anda dengan perintah berikut:

```
eksctl version
```

Untuk petunjuk tentang cara menginstal atau meningkatkan`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

> **Note**
>
> Prosedur ini hanya bekerja untuk klaster yang dibuat dengan `eksctl`.

1. Salin konten berikut ke perangkat Anda. Ganti *`my-cluster`* dengan nama klaster Anda. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter. Ganti *`ng-bottlerocket`* dengan nama untuk grup node Anda. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa. Untuk menerapkan pada instance Arm, ganti *`m5.large`* dengan tipe instance Arm. Ganti *`my-ec2-keypair-name`* dengan nama key pair Amazon EC2 SSH yang dapat Anda gunakan untuk terhubung menggunakan SSH ke node Anda setelah diluncurkan. Jika Anda belum memiliki pasangan kunci Amazon EC2, maka Anda dapat membuatnya di AWS Management Console. Untuk informasi selengkapnya, lihat Pasangan kunci Amazon EC2 dalam Panduan Pengguna Amazon EC2 untuk Instans Linux. Ganti semua yang tersisa *`example values`*dengan nilai Anda sendiri. Setelah Anda membuat penggantian, jalankan perintah yang dimodifikasi untuk membuat `bottlerocket.yaml` file.

   Jika menentukan tipe instans Arm Amazon EC2, maka lanjutkan dengan meninjau pertimbangan di Amazon EKS mengoptimalkan AMI Arm Amazon Linux sebelum men-deploy. Untuk petunjuk tentang cara menerapkan menggunakan AMI kustom, lihat BottlerocketMembangun GitHub dan mendukung AMI khusus dalam `eksctl` dokumentasi. Untuk menerapkan grup node terkelola, terapkan AMI kustom menggunakan template peluncuran. Untuk informasi selengkapnya, lihat Menyesuaikan node terkelola dengan template peluncuran.

> **Important**
>
> Untuk menyebarkan grup node ke AWS Outposts, AWS Wavelength, atau subnet Zona AWS Lokal, jangan lulus AWS Outposts AWS Wavelength, atau subnet Zona AWS Lokal saat Anda membuat klaster. Anda harus menentukan subnet dalam contoh berikut. Untuk informasi selengkapnya lihat Buat nodegroup dari file config dan Skema file config di dalam dokumentasi `eksctl`. Ganti *`region-code`* dengan tempat Wilayah AWS cluster Anda berada.

```
cat >bottlerocket.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: '1.29'

iam:
  withOIDC: true

nodeGroups:
  - name: ng-bottlerocket
    instanceType: m5.large
    desiredCapacity: 3
    amiFamily: Bottlerocket
    ami: auto-ssm
    iam:
       attachPolicyARNs:
          - arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
          - arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
          - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
          - arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
    ssh:
        allow: true
        publicKeyName: my-ec2-keypair-name
EOF
```

2. Deploy simpul Anda dengan perintah berikut.

```
eksctl create nodegroup --config-file=bottlerocket.yaml
```

Contoh output adalah sebagai berikut.

Beberapa baris adalah output sementara node dibuat. Salah satu baris terakhir dari output adalah baris contoh berikutnya.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Opsional) Buat volume Kubernetes persisten pada Bottlerocket node menggunakan Amazon EBS CSI Plugin. Driver Amazon EBS default bergantung pada alat sistem file yang tidak disertakan. Bottlerocket Untuk informasi selengkapnya tentang cara membuat kelas penyimpanan menggunakan driver, lihat Driver CSI Amazon EBS.
4. (Opsional) Secara default, `kube-proxy` menetapkan parameter `nf_conntrack_max` kernel ke nilai default yang mungkin berbeda dari apa yang Bottlerocket awalnya ditetapkan saat boot. Untuk menjaga Bottlerocket pengaturan default, edit `kube-proxy` konfigurasi dengan perintah berikut.

   ```
   kubectl edit -n kube-system daemonset kube-proxy
   ```

   Tambahkan `--conntrack-max-per-core` dan `--conntrack-min` ke `kube-proxy` argumen yang ada dalam contoh berikut. Pengaturan `0` menyiratkan bahwa tidak ada perubahan.

   ```
   containers:
   - command:
     - kube-proxy
     - --v=2
     - --config=/var/lib/kube-proxy-config/config
     - --conntrack-max-per-core=0
     - --conntrack-min=0
   ```
5. (Opsional) Menyebarkan aplikasi sampel untuk menguji Bottlerocket node Anda.
6. Kami merekomendasikan memblokir Pod akses ke IMDS jika kondisi berikut benar:

   - Anda berencana untuk menetapkan peran IAM ke semua akun Kubernetes layanan Anda sehingga Pods hanya memiliki izin minimum yang mereka butuhkan.

- Tidak ada Pods di klaster yang memerlukan akses ke layanan metadata instans Amazon EC2 (IMDS) karena alasan lain, seperti mengambil arus. Wilayah AWS

Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

Meluncurkan node yang dikelola sendiri Windows

Topik ini menjelaskan cara meluncurkan grup Windows node Auto Scaling yang mendaftar dengan kluster Amazon EKS Anda. Setelah node bergabung dengan cluster, Anda dapat menyebarkan Kubernetes aplikasi ke mereka.

> **Important**
>
> - Simpul Amazon EKS adalah instans Amazon EC2 standar, dan Anda dikenakan biaya berdasarkan harga instans Amazon EC2 normal. Untuk informasi selengkapnya, lihat Harga Amazon EC2.
> - Anda dapat meluncurkan node Windows di Amazon EKS cluster yang diperluas di AWS Outposts, tetapi Anda tidak dapat meluncurkannya di cluster AWS lokal di Outposts. Untuk informasi selengkapnya, lihat Amazon EKS pada AWS Outposts.

Aktifkan Windows dukungan untuk cluster Anda. Kami menyarankan Anda meninjau pertimbangan penting sebelum meluncurkan grup Windows node. Untuk informasi selengkapnya, lihat Mengaktifkan dukungan Windows.

Anda dapat meluncurkan Windows node yang dikelola sendiri dengan `eksctl` atau. AWS Management Console

eksctl

Untuk meluncurkan Windows node yang dikelola sendiri menggunakan **`eksctl`**

Prosedur ini mengharuskan Anda untuk sudah menginstal `eksctl`, dan bahwa `eksctl` versi yang Anda miliki setidaknya `0.175.0`. Anda dapat memeriksa versi Amazon Linux Anda dengan perintah berikut.

```
eksctl version
```

Untuk petunjuk tentang cara menginstal atau meningkatkan`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

> **Note**
>
> Prosedur ini hanya bekerja untuk klaster yang dibuat dengan `eksctl`.

1. (Opsional) Jika kebijakan IAM terkelola Amazoneks_CNI_Policy (jika Anda memiliki klaster) atau *`AmazonEKS_CNI_IPv6_Policy`* (yang Anda buat sendiri jika Anda `IPv4` memiliki klaster) dilampirkan ke Anda, kami sarankan untuk menetapkannya ke peran IAM `IPv6` yang Anda kaitkan ke the section called “IAM role simpul” akun layanan sebagai gantinya. Kubernetes `aws-node` Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).
2. Prosedur ini mengasumsikan bahwa Anda memiliki cluster yang ada. Jika Anda belum memiliki kluster Amazon EKS dan grup node Amazon Linux untuk menambahkan grup Windows node, kami sarankan Anda mengikuti Memulai dengan Amazon EKS – `eksctl` panduan ini. Panduan ini memberikan panduan lengkap tentang cara membuat cluster Amazon EKS dengan node Amazon Linux.

   Buat grup simpul Anda dengan perintah berikut. Ganti *`region-code`* dengan tempat Wilayah AWS cluster Anda berada. Ganti *`my-cluster`* dengan nama klaster Anda. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter. Ganti *`ng-windows`* dengan nama untuk grup node Anda. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa. Untuk Kubernetes versi `1.24` atau yang lebih baru, Anda dapat `2022` mengganti *`2019`* dengan menggunakan Windows Server 2022. Ganti sisanya *`example values`* dengan nilai-nilai Anda sendiri.

   > **Important**
   >
   > Untuk menyebarkan grup node ke AWS Outposts,, AWS Wavelength atau subnet Zona AWS Lokal, jangan lewatkan subnet, AWS Outposts Wavelength, atau Local Zone saat Anda membuat cluster. Buat grup node dengan file konfigurasi,

tentukan subnet, AWS Outposts Wavelength, atau Local Zone. Untuk informasi selengkapnya, lihat Membuat nodegroup dari file konfigurasi dan skema file Config dalam dokumentasi. `eksctl`

```
eksctl create nodegroup \
    --region region-code \
    --cluster my-cluster \
    --name ng-windows \
    --node-type t2.large \
    --nodes 3 \
    --nodes-min 1 \
    --nodes-max 4 \
    --managed=false \
    --node-ami-family WindowsServer2019FullContainer
```

Note

- Jika simpul gagal bergabung dengan klaster, lihat Simpul gagal untuk bergabung dengan klaster dalam panduan Pemecahan Masalah.
- Untuk melihat opsi yang tersedia untuk `eksctl` perintah, masukkan perintah berikut.

  ```
  eksctl command -help
  ```

Contoh output adalah sebagai berikut. Beberapa baris adalah output sementara node dibuat. Salah satu baris terakhir dari output adalah baris contoh berikutnya.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Opsional) Menyebarkan aplikasi sampel untuk menguji cluster dan Windows node Anda.
4. Kami merekomendasikan memblokir Pod akses ke IMDS jika kondisi berikut benar:

   - Anda berencana untuk menetapkan peran IAM ke semua akun Kubernetes layanan Anda sehingga Pods hanya memiliki izin minimum yang mereka butuhkan.

- Tidak ada Pods di klaster yang memerlukan akses ke layanan metadata instans Amazon EC2 (IMDS) karena alasan lain, seperti mengambil arus. Wilayah AWS

Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

AWS Management Console

Prasyarat

- Cluster Amazon EKS yang ada dan grup Linux node. Jika Anda tidak memiliki sumber daya ini, kami rekomendasikan supaya Anda mengikuti salah satu panduan Memulai dengan Amazon EKS untuk membuatnya. Panduan menjelaskan cara membuat cluster Amazon EKS dengan Linux node.
- Grup VPC dan keamanan yang ada yang memenuhi persyaratan untuk klaster Amazon EKS. Untuk informasi selengkapnya, lihat Persyaratan dan pertimbangan Amazon EKS VPC dan subnet dan Persyaratan dan pertimbangan grup keamanan Amazon EKS. Memulai dengan Amazon EKSPanduan ini membuat VPC yang memenuhi persyaratan. Atau, Anda juga dapat mengikuti Membuat VPC untuk klaster Amazon EKS Anda untuk membuatnya secara manual.
- Cluster Amazon EKS yang ada yang menggunakan VPC dan grup keamanan yang memenuhi persyaratan cluster Amazon EKS. Untuk informasi selengkapnya, lihat Membuat klaster Amazon EKS. Jika Anda memiliki subnet di Wilayah AWS mana Anda memiliki AWS Outposts, AWS Wavelength, atau AWS Local Zones diaktifkan, subnet tersebut harus tidak diteruskan saat Anda membuat cluster.

Langkah 1: Untuk meluncurkan Windows node yang dikelola sendiri menggunakan AWS Management Console

1. Tunggu status klaster Anda ditampilkan sebagai `ACTIVE`. Jika Anda meluncurkan node sebelum cluster aktif, node gagal mendaftar dengan cluster dan Anda perlu meluncurkannya kembali.
2. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation
3. Pilih Buat tumpukan.
4. Untuk Tentukan templat, pilih Amazon S3 URL.
5. Salin URL berikut dan tempel ke URL Amazon S3.

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2023-02-09/amazon-eks-windows-nodegroup.yaml
```

6. Pilih Berikutnya dua kali.
7. Pada halaman Quick create stack, masukkan parameter berikut yang sesuai:

   - Nama tumpukan: Pilih nama tumpukan untuk tumpukan AWS CloudFormation Anda. Misalnya, Anda bisa menyebutnya***my-cluster*-nodes**.
   - ClusterName: Masukkan nama yang Anda gunakan saat membuat cluster Amazon EKS Anda.

     > **Important**
     >
     > Nama ini harus sama persis dengan nama yang Anda gunakanLangkah 1: Buat klaster dan Amazon EKS Anda. Jika tidak, node Anda tidak dapat bergabung dengan cluster.

   - ClusterControlPlaneSecurityGroup: Pilih grup keamanan dari AWS CloudFormation output yang Anda hasilkan saat membuat VPC.

     Langkah-langkah berikut menunjukkan satu metode untuk mengambil grup yang berlaku.

     1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
     2. Pilih nama cluster.
     3. Pilih tab Jaringan.
     4. Gunakan nilai grup keamanan tambahan sebagai referensi saat memilih dari daftar ClusterControlPlaneSecurityGrouptarik-turun.

   - NodeGroupName: Masukkan nama untuk grup node Anda. Nama ini dapat digunakan nanti untuk mengidentifikasi grup node Auto Scaling yang dibuat untuk node Anda. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa.
   - NodeAutoScalingGroupMinSize: Masukkan jumlah minimum node yang dapat diskalakan oleh grup Auto Scaling node Anda.
   - NodeAutoScalingGroupDesiredCapacity: Masukkan jumlah node yang diinginkan untuk diskalakan saat tumpukan Anda dibuat.

- NodeAutoScalingGroupMaxSize: Masukkan jumlah maksimum node yang dapat diskalakan oleh grup Auto Scaling node Anda.
- NodeInstanceType: Pilih jenis instance untuk node Anda. Untuk informasi selengkapnya, lihat Memilih jenis instans Amazon EC2.

> **Note**
>
> Jenis instans yang didukung untuk versi terbaru tercantum di Amazon VPC CNI plugin for Kubernetesvpc_ip_resource_limit.go on. GitHub Anda mungkin perlu memperbarui versi CNI untuk menggunakan jenis instans terbaru yang didukung. Untuk informasi selengkapnya, lihat Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS.

- NodeImageIdSSMParam: Diisi sebelumnya dengan parameter Amazon EC2 Systems Manager dari ID AMI Inti yang dioptimalkan Amazon EKS saat ini. Windows Untuk menggunakan versi lengkapWindows, ganti `Core` dengan`Full`.
- NodeImageId: (Opsional) Jika Anda menggunakan AMI kustom Anda sendiri (bukan AMI yang dioptimalkan Amazon EKS), masukkan ID AMI node untuk Anda Wilayah AWS. Jika Anda menentukan nilai untuk bidang ini, itu akan mengganti nilai apa pun di bidang NodeImageIdSSMParam.
- NodeVolumeSize: Tentukan ukuran volume root untuk node Anda, di GiB.
- KeyName: Masukkan nama key pair Amazon EC2 SSH yang dapat Anda gunakan untuk terhubung menggunakan SSH ke node Anda setelah diluncurkan. Jika Anda belum memiliki pasangan kunci Amazon EC2, Anda dapat membuatnya di Konsol Manajemen AWS Management Console. Untuk informasi selengkapnya, lihat Pasangan Kunci Amazon EC2 dalam Panduan Pengguna Amazon EC2 untuk Instans Windows.

> **Note**
>
> Jika Anda tidak menyediakan key pair di sini, AWS CloudFormation tumpukan gagal dibuat.

- BootstrapArguments: Tentukan argumen opsional apa pun untuk diteruskan ke skrip bootstrap node, seperti `kubelet` argumen tambahan yang digunakan-`KubeletExtraArgs`.

- DisableImDSv1: Secara default, setiap node mendukung Instance Metadata Service Version 1 (IMDSv1) dan IMDSv2. Anda dapat menonaktifkan IMDSv1. Untuk mencegah node future dan Pods dalam grup node menggunakan mDSv1, atur disableImDSv1 ke true. Untuk informasi selengkapnya tentang IMDS, lihat Mengonfigurasi layanan metadata instans.
- VpcId: Pilih ID untuk VPC yang Anda buat.
- NodeSecurityGroups: Pilih grup keamanan yang dibuat untuk grup Linux node Anda saat Anda membuat VPC. Jika Linux node Anda memiliki lebih dari satu grup keamanan yang melekat padanya, tentukan semuanya. Ini untuk, misalnya, jika grup Linux node dibuat dengan`eksctl`.
- Subnet: Pilih subnet yang Anda buat. Jika Anda membuat VPC Anda menggunakan langkah-langkah diMembuat VPC untuk klaster Amazon EKS Anda, maka tentukan hanya subnet pribadi dalam VPC untuk node Anda untuk diluncurkan.

  > **Important**
  >
  > - Jika salah satu subnet adalah subnet publik, maka mereka harus mengaktifkan pengaturan tugas alamat IP publik otomatis. Jika pengaturan tidak diaktifkan untuk subnet publik, maka node apa pun yang Anda terapkan ke subnet publik tersebut tidak akan diberi alamat IP publik dan tidak akan dapat berkomunikasi dengan cluster atau layanan lainnya. AWS Jika subnet digunakan sebelum 26 Maret 2020 menggunakan salah satu templat AWS CloudFormation VPC Amazon EKS, atau dengan menggunakan`eksctl`, maka penetapan alamat IP publik otomatis dinonaktifkan untuk subnet publik. Untuk informasi tentang cara mengaktifkan penetapan alamat IP publik untuk subnet, lihat Memodifikasi atribut `IPv4` pengalamatan publik untuk subnet Anda. Jika node dikerahkan ke subnet pribadi, maka node dapat berkomunikasi dengan cluster dan AWS layanan lainnya melalui gateway NAT.
  > - Jika subnet tidak memiliki akses internet, pastikan Anda mengetahui pertimbangan dan langkah-langkah tambahan. Persyaratan klaster pribadi
  > - Jika Anda memilih AWS Outposts, Wavelength, atau subnet Local Zone, maka subnet tidak boleh diteruskan saat Anda membuat cluster.

8. Nyatakan bahwa tumpukan dapat membuat sumber daya IAM, kemudian pilih Buat tumpukan.
9. Setelah tumpukan Anda selesai dibuat, pilih tumpukan di konsol dan pilih Outputs.

10. Rekam NodeInstanceRoleuntuk grup node yang telah dibuat. Anda memerlukan ini saat mengonfigurasi Windows node Amazon EKS Anda.

Langkah 2: Untuk mengaktifkan node untuk bergabung dengan cluster Anda

1. Periksa untuk melihat apakah Anda sudah memiliki `aws-authConfigMap`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Jika Anda ditampilkan `aws-authConfigMap`, maka perbarui sesuai kebutuhan.

   a. Buka `ConfigMap` untuk mengedit.

```
kubectl edit -n kube-system configmap/aws-auth
```

   b. Tambahkan `mapRoles` entri baru sesuai kebutuhan. Tetapkan `rolearn` nilai ke NodeInstanceRolenilai yang Anda rekam dalam prosedur sebelumnya.

```
[...]
data:
  mapRoles: |
- rolearn: <ARN of linux instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
    - rolearn: <ARN of windows instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
        - eks:kube-proxy-windows
[...]
```

   c. Simpan file dan keluar dari editor teks Anda.

3. Jika Anda menerima kesalahan yang menyatakan "`Error from server (NotFound): configmaps "aws-auth" not found`, maka terapkan stok`ConfigMap`.

   a. Unduh peta konfigurasi.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/aws-auth-cm-windows.yaml
```

b. Dalam `aws-auth-cm-windows.yaml` file, atur `rolearn` nilai ke nilai yang berlaku NodeInstanceRoleyang Anda rekam dalam prosedur sebelumnya. Anda dapat melakukan ini dengan editor teks, atau dengan mengganti *`example values`* dan menjalankan perintah berikut:

```
sed -i.bak -e 's|<ARN of linux instance role (not instance profile)>|my-
node-linux-instance-role|' \
    -e 's|<ARN of windows instance role (not instance profile)>|my-node-
windows-instance-role|' aws-auth-cm-windows.yaml
```

> **Important**
>
> - Jangan memodifikasi baris lain dalam file ini.
> - Jangan gunakan peran IAM yang sama untuk keduanya Windows dan Linux node.

c. Terapkan konfigurasi. Perintah ini mungkin memakan waktu beberapa menit untuk menyelesaikannya.

```
kubectl apply -f aws-auth-cm-windows.yaml
```

4. Perhatikan status simpul Anda dan tunggu sampai simpul mencapai Status `Ready`.

```
kubectl get nodes --watch
```

Masukkan `Ctrl+C` untuk kembali ke prompt shell.

> **Note**
>
> Jika Anda menerima kesalahan otorisasi atau jenis sumber daya, lihat Tidak sah atau akses ditolak (`kubectl`) di topik pemecahan masalah.

Jika simpul gagal bergabung dengan klaster, lihat Simpul gagal untuk bergabung dengan klaster dalam Panduan pemecahan masalah.

Langkah 3: Tindakan tambahan

1. (Opsional) Menyebarkan aplikasi sampel untuk menguji cluster dan Windows node Anda.
2. (Opsional) Jika kebijakan IAM terkelola Amazoneks_CNI_Policy (jika Anda memiliki klaster) atau *AmazonEKS_CNI_IPv6_Policy* (yang Anda buat sendiri jika Anda IPv4 memiliki klaster) dilampirkan ke Anda, kami sarankan untuk menetapkannya ke peran IAM IPv6 yang Anda kaitkan ke the section called “IAM role simpul” akun layanan sebagai gantinya. Kubernetes aws-node Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).
3. Kami merekomendasikan memblokir Pod akses ke IMDS jika kondisi berikut benar:

   - Anda berencana untuk menetapkan peran IAM ke semua akun Kubernetes layanan Anda sehingga Pods hanya memiliki izin minimum yang mereka butuhkan.
   - Tidak ada Pods di klaster yang memerlukan akses ke layanan metadata instans Amazon EC2 (IMDS) karena alasan lain, seperti mengambil arus. Wilayah AWS

   Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

Pembaruan simpul yang dikelola sendiri

Saat AMI baru yang dioptimalkan Amazon EKS dirilis, pertimbangkan untuk mengganti node di grup node yang dikelola sendiri dengan AMI baru. Demikian juga, jika Anda telah memperbaruiKubernetesversi untuk cluster Amazon EKS Anda, perbarui node untuk menggunakan node dengan yang samaKubernetesversi.

> **Important**
>
> Topik ini mencakup pembaruan simpul untuk simpul yang dikelola sendiri. Jika Anda menggunakan Grup simpul terkelola, lihat Memperbarui grup simpul terkelola.

Ada dua cara dasar untuk memperbarui grup simpul yang dikelola sendiri di klaster Anda untuk menggunakan AMI baru:

Bermigrasi ke grup simpul baru

Buat grup node baru dan migrasikanPodsuntuk kelompok tersebut. Migrasi ke grup node baru lebih anggun daripada sekadar memperbarui ID AMI di yang sudah adaAWS CloudFormationtumpukan. Hal ini karena proses migrasinodakelompok node lama sebagaiNoScheduledan menguras node setelah tumpukan baru siap menerima yang adaPodbeban kerja.

Memperbarui grup simpul yang sudah ada yang dikelola sendiri

PerbaruiAWS CloudFormationtumpukan untuk grup node yang ada untuk menggunakan AMI baru. Metode ini tidak didukung untuk grup simpul yang dibuat denganeksctl.

Bermigrasi ke grup simpul baru

Topik ini menjelaskan bagaimana Anda dapat membuat grup node baru, dengan anggun memigrasikan aplikasi yang ada ke grup baru, dan menghapus grup node lama dari cluster Anda. Anda dapat bermigrasi ke grup simpul baru menggunakan `eksctl` atau AWS Management Console.

eksctl

Untuk memigrasi aplikasi Anda ke grup simpul baru dengan **`eksctl`**

Untuk informasi selengkapnya tentang penggunaan eksctl untuk migrasi, lihat Nodegroup tidak dikelola dalam dokumentasi. `eksctl`

Prosedur ini membutuhkan `eksctl` versi `0.175.0` atau yang lebih baru. Anda dapat memeriksa versi Anda dengan perintah berikut:

```
eksctl version
```

Untuk petunjuk tentang cara menginstal atau meningkatkan`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

> **Note**
>
> Prosedur ini hanya bekerja untuk grup klaster dan simpul yang dibuat dengan `eksctl`.

1. Ambil nama grup node yang ada, ganti *my-cluster* dengan nama cluster Anda.

```
eksctl get nodegroups --cluster=my-cluster
```

Contoh output adalah sebagai berikut.

```
CLUSTER      NODEGROUP          CREATED               MIN SIZE      MAX SIZE
 DESIRED CAPACITY     INSTANCE TYPE     IMAGE ID
default      standard-nodes   2019-05-01T22:26:58Z  1             4             3
                    t3.medium         ami-05a71d034119ffc12
```

2. Luncurkan grup node baru `eksctl` dengan perintah berikut. Dalam perintah, ganti setiap *example value*dengan nilai Anda sendiri. Nomor versi tidak boleh lebih lambat dari Kubernetes versi untuk pesawat kontrol Anda. Selain itu, tidak boleh lebih dari dua versi minor lebih awal dari Kubernetes versi untuk pesawat kontrol Anda. Kami menyarankan Anda menggunakan versi yang sama dengan pesawat kontrol Anda.

Kami merekomendasikan memblokir Pod akses ke IMDS jika kondisi berikut benar:

- Anda berencana untuk menetapkan peran IAM ke semua akun Kubernetes layanan Anda sehingga Pods hanya memiliki izin minimum yang mereka butuhkan.
- Tidak ada Pods di klaster yang memerlukan akses ke layanan metadata instans Amazon EC2 (IMDS) karena alasan lain, seperti mengambil arus. Wilayah AWS

Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

Untuk memblokir Pod akses ke IMDS, tambahkan `--disable-pod-imds` opsi ke perintah berikut.

> **Note**
>
> Untuk lebih banyak flag yang tersedia dan deskripsinya, lihat https://eksctl.io/.

```
eksctl create nodegroup \
  --cluster my-cluster \
  --version 1.29 \
```

```
  --name standard-nodes-new \
  --node-type t3.medium \
  --nodes 3 \
  --nodes-min 1 \
  --nodes-max 4 \
  --managed=false
```

3. Setelah perintah sebelumnya selesai, verifikasi bahwa semua simpul Anda telah mencapai status `Ready` dengan perintah berikut:

```
kubectl get nodes
```

4. Hapus grup node asli dengan perintah berikut. Dalam perintah, ganti setiap *example value* dengan nama grup cluster dan node Anda:

```
eksctl delete nodegroup --cluster my-cluster --name standard-nodes-old
```

AWS Management Console and AWS CLI

Untuk memigrasikan aplikasi Anda ke grup node baru dengan dan AWS Management ConsoleAWS CLI

1. Luncurkan grup node baru dengan mengikuti langkah-langkah yang diuraikan dalamMeluncurkan simpul Amazon Linux yang dikelola sendiri.
2. Setelah tumpukan Anda selesai dibuat, pilih tumpukan di konsol dan pilih Outputs.
3. Rekam NodeInstanceRoleuntuk grup node yang telah dibuat. Anda memerlukan ini untuk menambahkan simpul Amazon EKS baru ke klaster Anda.

> **Note**
>
> Jika Anda melampirkan kebijakan IAM tambahan ke peran IAM grup node lama Anda, lampirkan kebijakan yang sama tersebut ke peran IAM grup node baru Anda untuk mempertahankan fungsionalitas tersebut pada grup baru. Ini berlaku untuk Anda jika Anda menambahkan izin untuk Kubernetes Cluster Autoscaler, misalnya.

4. Perbarui grup keamanan untuk kedua grup simpul sehingga mereka dapat berkomunikasi satu sama lain. Untuk informasi selengkapnya, lihat Persyaratan dan pertimbangan grup keamanan Amazon EKS.

a. Catat ID grup keamanan untuk kedua grup simpul. Ini ditampilkan sebagai NodeSecurityGroupnilai dalam output AWS CloudFormation stack.

   Anda dapat menggunakan AWS CLI perintah berikut untuk mendapatkan ID grup keamanan dari nama tumpukan. Dalam perintah ini, `oldNodes` adalah nama AWS CloudFormation tumpukan untuk tumpukan node lama Anda, dan `newNodes` merupakan nama tumpukan tempat Anda bermigrasi. Ganti setiap *`example value`* dengan nilai-nilai Anda sendiri.

   ```
   oldNodes="old_node_CFN_stack_name"
   newNodes="new_node_CFN_stack_name"

   oldSecGroup=$(aws cloudformation describe-stack-resources --stack-name
    $oldNodes \
   --query 'StackResources[?
   ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
   --output text)
   newSecGroup=$(aws cloudformation describe-stack-resources --stack-name
    $newNodes \
   --query 'StackResources[?
   ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
   --output text)
   ```

b. Tambahkan aturan masuk ke setiap grup keamanan simpul sehingga mereka menerima lalu lintas dari satu sama lain.

   AWS CLI Perintah berikut menambahkan aturan masuk ke setiap grup keamanan yang memungkinkan semua lalu lintas pada semua protokol dari grup keamanan lainnya. Konfigurasi ini memungkinkan Pods di setiap grup node untuk berkomunikasi satu sama lain saat Anda memigrasikan beban kerja Anda ke grup baru.

   ```
   aws ec2 authorize-security-group-ingress --group-id $oldSecGroup \
   --source-group $newSecGroup --protocol -1
   aws ec2 authorize-security-group-ingress --group-id $newSecGroup \
   --source-group $oldSecGroup --protocol -1
   ```

5. Edit configmap `aws-auth` untuk memetakan peran instans simpul baru di RBAC.

   ```
   kubectl edit configmap -n kube-system aws-auth
   ```

Tambahkan entri `mapRoles` baru untuk grup simpul baru. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
apiVersion: v1
data:
  mapRoles: |
    - rolearn: ARN of instance role (not instance profile)
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes>
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-16-NodeInstanceRole-
U11V27W93CX5
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
```

Ganti *ARN of instance role (not instance profile)* cuplikan dengan NodeInstanceRolenilai yang Anda rekam pada langkah sebelumnya. Kemudian, simpan dan tutup file untuk menerapkan configmap yang diperbarui.

6. Perhatikan status simpul Anda dan tunggu simpul baru Anda bergabung dengan klaster dan mencapai status `Ready` tersebut.

```
kubectl get nodes --watch
```

7. (Opsional) Jika Anda menggunakan Kubernetes Cluster Autoscaler, skala penerapan ke nol (0) replika untuk menghindari tindakan penskalaan yang bertentangan.

```
kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
```

8. Gunakan perintah berikut untuk mencemari setiap node yang ingin Anda hapus. `NoSchedule` Ini agar yang baru Pods tidak dijadwalkan atau dijadwal ulang pada node yang Anda ganti. Untuk informasi selengkapnya, lihat Taints and Tolerations dalam dokumentasi. Kubernetes

```
kubectl taint nodes node_name key=value:NoSchedule
```

Jika Anda memutakhirkan node ke Kubernetes versi baru, Anda dapat mengidentifikasi dan mencemari semua node dari Kubernetes versi tertentu (dalam hal ini,`1.27`) dengan cuplikan kode berikut. Nomor versi tidak boleh lebih lambat dari Kubernetes versi pesawat kontrol Anda. Ini juga tidak boleh lebih dari dua versi minor lebih awal dari Kubernetes versi pesawat kontrol Anda. Kami menyarankan Anda menggunakan versi yang sama dengan pesawat kontrol Anda.

```
K8S_VERSION=1.27
nodes=$(kubectl get nodes -o jsonpath="{.items[?
(@.status.nodeInfo.kubeletVersion==\"v$K8S_VERSION\")].metadata.name}")
for node in ${nodes[@]}
do
    echo "Tainting $node"
    kubectl taint nodes $node key=value:NoSchedule
done
```

9. Tentukan penyedia DNS klaster Anda.

```
kubectl get deployments -l k8s-app=kube-dns -n kube-system
```

Contoh output adalah sebagai berikut. Cluster ini digunakan CoreDNS untuk resolusi DNS, tetapi klaster Anda dapat kembali `kube-dns` sebagai gantinya):

```
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
coredns   1         1         1            1           31m
```

10. Jika deployment Anda saat ini berjalan kurang dari dua replika, skalakan deployment menjadi dua replika. Ganti *`coredns`* dengan **kubedns** jika output perintah Anda sebelumnya mengembalikannya.

```
kubectl scale deployments/coredns --replicas=2 -n kube-system
```

11. Keluarkan setiap simpul yang Anda ingin hapus dari klaster Anda dengan perintah berikut:

```
kubectl drain node_name --ignore-daemonsets --delete-local-data
```

Jika Anda memutakhirkan node ke Kubernetes versi baru, identifikasi dan tiriskan semua node dari Kubernetes versi tertentu (dalam hal ini,*`1.27`*) dengan cuplikan kode berikut.

```
K8S_VERSION=1.27
nodes=$(kubectl get nodes -o jsonpath="{.items[?
(@.status.nodeInfo.kubeletVersion==\"v$K8S_VERSION\")].metadata.name}")
for node in ${nodes[@]}
do
    echo "Draining $node"
    kubectl drain $node --ignore-daemonsets --delete-local-data
done
```

12. Setelah node lama Anda selesai menguras, cabut aturan masuk grup keamanan yang Anda otorisasi sebelumnya. Kemudian, hapus AWS CloudFormation tumpukan untuk mengakhiri instance.

> **Note**
>
> Jika Anda melampirkan kebijakan IAM tambahan ke peran IAM grup node lama Anda, seperti menambahkan izin untuk Kubernetes Cluster Autoscaler), lepaskan kebijakan tambahan tersebut dari peran tersebut sebelum Anda dapat menghapus tumpukan. AWS CloudFormation

a. Cabut aturan masuk yang Anda buat untuk grup keamanan node Anda sebelumnya. Dalam perintah ini, `oldNodes` adalah nama AWS CloudFormation tumpukan untuk tumpukan node lama Anda, dan `newNodes` merupakan nama tumpukan tempat Anda bermigrasi.

```
oldNodes="old_node_CFN_stack_name"
newNodes="new_node_CFN_stack_name"

oldSecGroup=$(aws cloudformation describe-stack-resources --stack-name
 $oldNodes \
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
newSecGroup=$(aws cloudformation describe-stack-resources --stack-name
 $newNodes \
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
aws ec2 revoke-security-group-ingress --group-id $oldSecGroup \
```

```
--source-group $newSecGroup --protocol -1
aws ec2 revoke-security-group-ingress --group-id $newSecGroup \
--source-group $oldSecGroup --protocol -1
```

b. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.

c. Pilih tumpukan simpul lama Anda.

d. Pilih Hapus.

e. Di kotak dialog Hapus tumpukan konfirmasi, pilih Hapus tumpukan.

13. Edit `aws-auth` configmap untuk menghapus peran instans simpul lama dari RBAC.

```
kubectl edit configmap -n kube-system aws-auth
```

Hapus `mapRoles` entri untuk grup simpul lama. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
apiVersion: v1
data:
  mapRoles: |
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-16-NodeInstanceRole-
W70725MZQFF8
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-15-NodeInstanceRole-
U11V27W93CX5
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes>
```

Simpan dan tutup file untuk menerapkan configmap yang diperbarui.

14. (Opsional) Jika Anda menggunakan Kubernetes Cluster Autoscaler, skala penerapan kembali ke satu replika.

> **Note**
>
> Anda juga harus menandai grup Auto Scaling baru Anda dengan tepat (misalnya,`k8s.io/cluster-autoscaler/enabled,k8s.io/cluster-autoscaler/`*`my-cluster`*) dan memperbarui perintah untuk penerapan Cluster Autoscaler Anda untuk menunjuk ke grup Auto Scaling yang baru ditandai. Untuk informasi selengkapnya, lihat Cluster Autoscaler di. AWS

```
kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
```

15. (Opsional) Verifikasi bahwa Anda menggunakan versi terbaru dari plugin Amazon VPC CNI untuk Kubernetes. Anda mungkin perlu memperbarui versi CNI untuk menggunakan jenis instans terbaru yang didukung. Untuk informasi selengkapnya, lihat Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS.
16. Jika klaster Anda menggunakan `kube-dns` untuk resolusi DNS (lihat langkah sebelumnya), skalakan ke dalam deployment `kube-dns` hingga satu replika.

```
kubectl scale deployments/kube-dns --replicas=1 -n kube-system
```

Memperbarui grup simpul yang sudah ada yang dikelola sendiri

Topik ini menjelaskan bagaimana Anda dapat memperbarui tumpukan node yang AWS CloudFormation dikelola sendiri dengan AMI baru. Anda dapat menggunakan prosedur ini untuk memperbarui node Anda ke versi baru Kubernetes mengikuti pembaruan cluster. Jika tidak, Anda dapat memperbarui ke AMI Amazon EKS terbaru yang dioptimalkan untuk Kubernetes versi yang ada.

> **Important**
>
> Topik ini mencakup pembaruan simpul untuk simpul yang dikelola sendiri. Untuk informasi tentang penggunaanGrup simpul terkelola, lihatMemperbarui grup simpul terkelola.

AWS CloudFormation Template node Amazon EKS default terbaru dikonfigurasi untuk meluncurkan instance dengan AMI baru ke dalam klaster Anda sebelum menghapus yang lama, satu per satu.

Konfigurasi ini memastikan bahwa Anda selalu memiliki jumlah instans aktif yang diinginkan oleh grup Auto Scaling di klaster Anda selama pembaruan masih berlanjut.

> **Note**
>
> Metode ini tidak didukung untuk grup node yang dibuat dengan`eksctl`. Jika Anda membuat grup klaster atau simpul dengan `eksctl`, lihat Bermigrasi ke grup simpul baru.

Untuk memperbarui grup simpul yang ada

1. Tentukan penyedia DNS untuk klaster Anda.

   ```
   kubectl get deployments -l k8s-app=kube-dns -n kube-system
   ```

   Contoh output adalah sebagai berikut. Cluster ini digunakan CoreDNS untuk resolusi DNS, tetapi klaster Anda mungkin kembali `kube-dns` sebagai gantinya. Output Anda mungkin terlihat berbeda tergantung pada versi `kubectl` yang Anda gunakan.

   ```
   NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
   coredns   1         1         1            1           31m
   ```

2. Jika deployment Anda saat ini berjalan kurang dari dua replika, skalakan deployment menjadi dua replika. Ganti *coredns* dengan **kube-dns** jika output perintah Anda sebelumnya mengembalikannya.

   ```
   kubectl scale deployments/coredns --replicas=2 -n kube-system
   ```

3. (Opsional) Jika Anda menggunakan Kubernetes Cluster Autoscaler, skala penerapan ke nol (0) replika untuk menghindari tindakan penskalaan yang bertentangan.

   ```
   kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
   ```

4. Tentukan tipe instans dan jumlah instans yang diinginkan dari grup simpul Anda saat ini. Anda memasukkan nilai-nilai ini nanti ketika Anda memperbarui AWS CloudFormation template untuk grup.

   a. Buka konsol Amazon EC2 di https://console.aws.amazon.com/ec2/.

b. Di panel navigasi kiri, pilih Luncurkan Konfigurasi, dan catat jenis instans untuk konfigurasi peluncuran node yang ada.

c. Di panel navigasi kiri, pilih Grup Auto Scaling, dan catat jumlah instans yang diinginkan untuk grup Auto Scaling node yang ada.

5. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.

6. Pilih tumpukan grup simpul Anda, dan kemudian pilih Perbarui.

7. Pilih Ganti template saat ini dan pilih URL Amazon S3.

8. Untuk URL Amazon S3, rekatkan URL berikut ke area teks untuk memastikan bahwa Anda menggunakan template node AWS CloudFormation versi terbaru. Kemudian, pilih Berikutnya:

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/amazon-eks-nodegroup.yaml
```

9. Pada halaman Menentukan detail tumpukan, isilah parameter berikut dan pilih Selanjutnya:

- NodeAutoScalingGroupDesiredCapacity— Masukkan jumlah instans yang diinginkan yang Anda rekam pada langkah sebelumnya. Atau, masukkan jumlah node baru yang Anda inginkan untuk diskalakan saat tumpukan Anda diperbarui.
- NodeAutoScalingGroupMaxSize— Masukkan jumlah maksimum node yang dapat diskalakan oleh grup Auto Scaling node Anda. Nilai ini harus setidaknya satu node lebih dari kapasitas yang Anda inginkan. Ini agar Anda dapat melakukan pembaruan bergulir dari node Anda tanpa mengurangi jumlah node Anda selama pembaruan.
- NodeInstanceType— Pilih jenis instans yang Anda rekam di langkah sebelumnya. Atau, pilih jenis instance yang berbeda untuk node Anda. Sebelum memilih jenis instans yang berbeda, tinjauMemilih jenis instans Amazon EC2. Setiap jenis instans Amazon EC2 mendukung jumlah maksimum antarmuka jaringan elastis (antarmuka jaringan) dan setiap antarmuka jaringan mendukung jumlah maksimum alamat IP. Karena setiap node pekerja danPod, diberi alamat IP sendiri, penting untuk memilih jenis instance yang akan mendukung jumlah maksimum Pods yang ingin Anda jalankan di setiap node Amazon EC2. Untuk daftar jumlah antarmuka jaringan dan alamat IP yang didukung oleh jenis instans, lihat alamat IP per antarmuka jaringan per jenis instans. Misalnya, tipe `m5.large` instance mendukung maksimum 30 alamat IP untuk node pekerja danPods.

> **Note**
>
> Jenis instans yang didukung untuk versi terbaru ditampilkan di Amazon VPC CNI plugin for Kubernetesvpc_ip_resource_limit.go on. GitHub Anda mungkin perlu memperbarui Amazon VPC CNI plugin for Kubernetes versi untuk menggunakan jenis instans terbaru yang didukung. Untuk informasi selengkapnya, lihat Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS.

> **Important**
>
> Beberapa jenis instance mungkin tidak tersedia di semua Wilayah AWS.

- NodeImageIdSSMParam - Parameter Amazon EC2 Systems Manager dari ID AMI yang ingin Anda perbarui. Nilai berikut menggunakan AMI Amazon EKS terbaru yang dioptimalkan untuk Kubernetes versi`1.29`.

```
/aws/service/eks/optimized-ami/1.29/amazon-linux-2/recommended/image_id
```

Anda dapat mengganti *`1.29`* dengan Kubernetesversi yang didukung yang sama. Atau, itu harus sampai satu versi lebih awal dari Kubernetes versi yang berjalan di bidang kontrol Anda. Kami rekomendasikan supaya Anda menyimpan simpul pada versi yang sama dengan bidang kendali Anda. Anda juga dapat mengganti *`amazon-linux-2`* dengan tipe AMI yang berbeda. Untuk informasi selengkapnya, lihat Mendapatkan ID Amazon Linux AMI yang dioptimalkan Amazon EKS.

> **Note**
>
> Menggunakan parameter Amazon EC2 Systems Manager memungkinkan Anda memperbarui node di masa mendatang tanpa harus mencari dan menentukan ID AMI. Jika AWS CloudFormation tumpukan Anda menggunakan nilai ini, pembaruan tumpukan apa pun selalu meluncurkan AMI terbaru yang dioptimalkan Amazon EKS yang dioptimalkan untuk Kubernetes versi yang Anda tentukan. Ini bahkan terjadi bahkan jika Anda tidak mengubah nilai apa pun di template.

- NodeImageId— Untuk menggunakan AMI kustom Anda sendiri, masukkan ID untuk AMI yang akan digunakan.

  > **Important**
  >
  > Nilai ini mengesampingkan nilai apa pun yang ditentukan untuk NodeImageId SSMParam. Jika Anda ingin menggunakan nilai NodeImageIdSSMParam, pastikan nilai untuk NodeImageIdkosong.

- DisableIMDSv1 — Secara default, setiap node mendukung Instance Metadata Service Version 1 (IMDSv1) dan IMDSv2. Namun, Anda dapat menonaktifkan IMDSv1. Pilih true jika Anda tidak ingin node atau Pods terjadwal dalam grup node untuk menggunakan IMDSv1. Untuk informasi selengkapnya tentang IMDS, lihat Mengonfigurasi layanan metadata instans. Jika Anda telah menerapkan peran IAM untuk akun layanan, tetapkan izin yang diperlukan langsung ke semua Pods yang memerlukan akses ke layanan. AWS Dengan cara ini, tidak ada Pods di cluster Anda yang memerlukan akses ke IMDS karena alasan lain, seperti mengambil arus. Wilayah AWS Kemudian, Anda juga dapat menonaktifkan akses ke IMDSv2 untuk Pods itu jangan gunakan jaringan host. Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

10. (Opsional) Di halaman Opsi, tandai sumber daya tumpukan Anda. Pilih Berikutnya.
11. Pada halaman Tinjauan, tinjau informasi Anda, nyatakan bahwa tumpukan dapat membuat sumber daya IAM, kemudian pilih Perbarui tumpukan.

    > **Note**
    >
    > Pembaruan setiap simpul dalam klaster membutuhkan waktu beberapa menit. Tunggu pembaruan semua simpul selesai sebelum melakukan langkah berikutnya.

12. Jika penyedia DNS klaster Anda adalah `kube-dns`, skalakan ke dalam deployment `kube-dns` hingga satu replika.

    ```
    kubectl scale deployments/kube-dns --replicas=1 -n kube-system
    ```

13. (Opsional) Jika Anda menggunakan Kubernetes Cluster Autoscaler, skala penerapan kembali ke jumlah replika yang Anda inginkan.

    ```
    kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
    ```

14. (Opsional) Verifikasi bahwa Anda menggunakan versi terbaru Amazon VPC CNI plugin for Kubernetes. Anda mungkin perlu memperbarui Amazon VPC CNI plugin for Kubernetes versi untuk menggunakan jenis instans terbaru yang didukung. Untuk informasi selengkapnya, lihat Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS.

AWS Fargate

> **Important**
>
> AWS Fargate dengan Amazon EKS tidak tersedia di AWS GovCloud (AS-Timur) dan AWS GovCloud (AS-Barat).

Topik ini membahas penggunaan Amazon EKS untuk dijalankan KubernetesPods. AWS Fargate Fargate adalah teknologi yang menyediakan kapasitas komputasi sesuai permintaan dan berukuran tepat untuk kontainer. Dengan Fargate, Anda tidak perlu menyediakan, mengonfigurasi, atau menskalakan grup mesin virtual sendiri untuk menjalankan kontainer. Anda juga tidak perlu memilih jenis server, memutuskan kapan harus menskalakan grup node Anda, atau mengoptimalkan pengemasan cluster.

Anda dapat mengontrol mana yang Pods dimulai di Fargate dan bagaimana mereka berjalan dengan profil Fargate. Profil Fargate didefinisikan sebagai bagian dari cluster Amazon EKS Anda. Amazon EKS terintegrasi Kubernetes dengan Fargate dengan menggunakan pengontrol yang dibuat dengan menggunakan model upstream yang dapat diperluas yang disediakan AWS oleh. Kubernetes Pengontrol ini berjalan sebagai bagian dari pesawat Kubernetes kontrol terkelola Amazon EKS dan bertanggung jawab untuk menjadwalkan asli ke Kubernetes Pods Fargate. Pengontrol Fargate menyertakan penjadwal baru yang berjalan bersama penjadwal default selain beberapa Kubernetes pengontrol penerimaan yang bermutasi dan memvalidasi. Saat Anda memulai sebuah Pod yang memenuhi kriteria untuk berjalan di Fargate, pengontrol Fargate yang berjalan di cluster mengenali, memperbarui, dan menjadwalkan ke Fargate. Pod

Topik ini menjelaskan berbagai komponen Pods yang berjalan di Fargate, dan menyerukan pertimbangan khusus untuk menggunakan Fargate dengan Amazon EKS.

AWS Fargate pertimbangan

Berikut adalah beberapa hal yang perlu dipertimbangkan tentang menggunakan Fargate di Amazon EKS.

- Masing-masing Pod yang berjalan di Fargate memiliki batas isolasi sendiri. Mereka tidak berbagi kernel yang mendasarinya, sumber daya CPU, sumber daya memori, atau elastic network interface dengan yang lainPod.
- Network Load Balancers dan Application Load Balancers (ALB) dapat digunakan dengan Fargate dengan target IP saja. Lihat informasi yang lebih lengkap di Buat penyeimbang beban jaringan dan Penyeimbangan beban aplikasi pada Amazon EKS.
- Layanan terbuka Fargate hanya berjalan pada mode IP tipe target, dan bukan pada mode IP node. Cara yang disarankan untuk memeriksa konektivitas dari layanan yang berjalan pada node terkelola dan layanan yang berjalan di Fargate adalah dengan menghubungkan melalui nama layanan.
- Pod harus cocok dengan profil Fargate pada saat mereka dijadwalkan untuk berjalan di Fargate. Pod yang tidak cocok dengan profil Fargate mungkin macet sebagai `Pending` Jika profil Fargate yang cocok ada, Anda dapat menghapus pending Pods yang telah Anda buat untuk menjadwal ulang mereka ke Fargate.
- Daemonset tidak didukung di Fargate. Jika aplikasi Anda memerlukan daemon, konfigurasikan ulang daemon tersebut untuk dijalankan sebagai wadah sespan di Anda. Pods
- Kontainer istimewa tidak didukung di Fargate.
- Pod yang berjalan di Fargate tidak dapat menentukan `HostPort` atau `HostNetwork` dalam manifes. Pod
- Batas default `nofile` dan `nproc` lunak adalah 1024 dan batas kerasnya adalah 65535 untuk Fargate. Pods
- GPU saat ini tidak tersedia di Fargate.
- Pod yang berjalan di Fargate hanya didukung pada subnet pribadi (dengan akses gateway NAT ke AWS layanan, tetapi bukan rute langsung ke Internet Gateway), jadi VPC klaster Anda harus memiliki subnet pribadi yang tersedia. Untuk klaster tanpa akses internet luar, lihat Persyaratan klaster pribadi.
- Anda dapat menggunakan Vertical Pod Autoscaler untuk mengatur ukuran CPU dan memori awal yang benar untuk Fargate AndaPods, dan kemudian menggunakan Penskala Otomatis Pod Horizontal untuk menskalakannya. Pods Jika Anda ingin Vertical Pod Autoscaler untuk secara otomatis di-deploy ulang Pods ke Fargate dengan kombinasi CPU dan memori yang lebih besar, atur mode untuk Vertical Pod Autoscaler ke salah satu atau untuk memastikan fungsionalitas yang benar. `Auto` `Recreate` Untuk informasi selengkapnya, lihat dokumentasi Vertical Pod Autoscaler pada. GitHub

- Resolusi DNS dan nama host DNS harus diaktifkan untuk VPC Anda. Untuk informasi selengkapnya, lihat Melihat dan memperbarui dukungan DNS untuk VPC Anda.
- Amazon EKS Fargate menambahkan defense-in-depth Kubernetes aplikasi dengan mengisolasi setiap Pod dalam Mesin Virtual (VM). Batas VM ini mencegah akses ke sumber daya berbasis host yang digunakan oleh Pod lain jika terjadi pelarian kontainer, yang merupakan metode umum untuk menyerang aplikasi kontainer dan mendapatkan akses ke sumber daya di luar container.

  Menggunakan Amazon EKS tidak mengubah tanggung jawab Anda berdasarkan model tanggung jawab bersama. Anda harus hati-hati mempertimbangkan konfigurasi keamanan klaster dan kontrol tata kelola. Cara teraman untuk mengisolasi aplikasi adalah selalu menjalankannya di cluster terpisah.
- Profil Fargate mendukung penentuan subnet dari blok CIDR sekunder VPC. Anda mungkin ingin menentukan blok CIDR sekunder. Ini karena ada sejumlah alamat IP yang tersedia di subnet. Akibatnya, ada juga sejumlah terbatas Pods yang dapat dibuat di cluster. Dengan menggunakan subnet yang berbeda untukPods, Anda dapat meningkatkan jumlah alamat IP yang tersedia. Untuk informasi selengkapnya, lihat Menambahkan blok `IPv4` CIDR ke VPC.
- Layanan metadata instans Amazon EC2 (IMDS) tidak tersedia untuk yang Pods digunakan ke node Fargate. Jika Anda memilikinya Pods yang disebarkan ke Fargate yang membutuhkan kredensyal IAM, tetapkan ke penggunaan Anda. Pods IAM role untuk akun layanan Jika Anda Pods memerlukan akses ke informasi lain yang tersedia melalui IMDS, maka Anda harus membuat kode keras informasi ini ke dalam Pod spesifikasi Anda. Ini termasuk Wilayah AWS atau Availability Zone Pod yang digunakan.
- Anda tidak dapat menerapkan Pods Fargate AWS Outposts ke AWS Wavelength,, atau Local AWS Zones.
- Amazon EKS harus menambal Fargate secara berkala Pods agar tetap aman. Kami mencoba pembaruan dengan cara yang mengurangi dampak, tetapi ada kalanya Pods harus dihapus jika tidak berhasil diusir. Ada beberapa tindakan yang dapat Anda lakukan untuk meminimalkan gangguan. Untuk informasi selengkapnya, lihat Penambalan OS Fargate.
- Plugin Amazon VPC CNI untuk Amazon EKS diinstal pada node Fargate. Anda tidak dapat menggunakan Plugin CNI alternatif yang kompatibel dengan node Fargate.
- Sebuah Pod berjalan di Fargate secara otomatis memasang sistem file Amazon EFS. Anda tidak dapat menggunakan penyediaan volume persisten dinamis dengan simpul Fargate, tetapi Anda dapat menggunakan penyediaan statis.
- Anda tidak dapat memasang volume Amazon EBS ke FargatePods.

- Anda dapat menjalankan pengontrol Amazon EBS CSI di node Fargate, tetapi node Amazon EBS CSI hanya dapat DaemonSet berjalan di instans Amazon EC2.
- Setelah Kubernetes Job ditandai `Completed` atau`Failed`, Pods yang biasanya Job dibuat terus ada. Perilaku ini memungkinkan Anda untuk melihat log dan hasil Anda, tetapi dengan Fargate Anda akan dikenakan biaya jika Anda tidak membersihkannya setelahnya. Job

  Untuk secara otomatis menghapus terkait Pods setelah Job selesai atau gagal, Anda dapat menentukan periode waktu menggunakan pengontrol time-to-live (TTL). Contoh berikut menunjukkan menentukan `.spec.ttlSecondsAfterFinished` dalam Job manifes Anda.

```
apiVersion: batch/v1
kind: Job
metadata:
  name: busybox
spec:
  template:
    spec:
      containers:
      - name: busybox
        image: busybox
        command: ["/bin/sh", "-c", "sleep 10"]
      restartPolicy: Never
  ttlSecondsAfterFinished: 60 # <-- TTL controller
```

Memulai dengan AWS Fargate menggunakan Amazon EKS

> **Important**
>
> AWS Fargate dengan Amazon EKS tidak tersedia di AWS GovCloud (AS-Timur) dan AWS GovCloud (AS-Barat).

Topik ini menjelaskan cara memulai menjalankan Pods kluster Amazon EKS Anda. AWS Fargate

Jika Anda membatasi akses ke titik akhir publik klaster Anda menggunakan blok CIDR, sebaiknya Anda juga mengaktifkan akses titik akhir pribadi. Dengan cara ini, Fargate Pods dapat berkomunikasi dengan cluster. Tanpa mengaktifkan titik akhir pribadi, blok CIDR yang Anda tentukan untuk akses publik harus menyertakan sumber keluar dari VPC Anda. Untuk informasi selengkapnya, lihat Kendali akses titik akhir klaster Amazon EKS.

Prasyarat

Sebuah klaster yang sudah ada. Jika Anda belum memiliki cluster Amazon EKS, lihatMemulai dengan Amazon EKS.

Pastikan node yang ada dapat berkomunikasi dengan Fargate Pods

Jika Anda bekerja dengan cluster baru tanpa node, atau cluster dengan hanya grup node terkelola, Anda dapat melompat keBuat peran eksekusi Fargate Pod.

Asumsikan bahwa Anda bekerja dengan cluster yang sudah ada yang sudah memiliki node yang terkait dengannya. Pastikan bahwa Pods pada node ini dapat berkomunikasi secara bebas dengan Pods yang berjalan di Fargate. Podsyang berjalan di Fargate secara otomatis dikonfigurasi untuk menggunakan grup keamanan cluster untuk cluster yang terkait dengannya. Pastikan bahwa setiap node yang ada di cluster Anda dapat mengirim dan menerima lalu lintas ke dan dari grup keamanan klaster. Grup simpul terkelolasecara otomatis dikonfigurasi untuk menggunakan grup keamanan cluster juga, jadi Anda tidak perlu memodifikasi atau memeriksa kompatibilitas ini.

Untuk grup node yang sudah ada yang dibuat dengan `eksctl` atau AWS CloudFormation templat terkelola Amazon EKS, Anda dapat menambahkan grup keamanan klaster ke node secara manual. Atau, sebagai alternatif, Anda dapat memodifikasi templat peluncuran grup Auto Scaling untuk grup node untuk melampirkan grup keamanan klaster ke instance. Untuk informasi selengkapnya, lihat Mengubah grup keamanan dari instans dalam Panduan Pengguna Amazon VPC.

Anda dapat memeriksa grup keamanan untuk klaster Anda AWS Management Console di bagian bawah Jaringan untuk cluster. Atau, Anda dapat melakukan ini menggunakan AWS CLI perintah berikut. Saat menggunakan perintah ini, ganti *`my-cluster`* dengan nama cluster Anda.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

Buat peran eksekusi Fargate Pod

Saat kluster Pods Anda AWS Fargate aktif, komponen yang berjalan di infrastruktur Fargate harus melakukan panggilan ke AWS API atas nama Anda. Peran Pod eksekusi Amazon EKS memberikan izin IAM untuk melakukan ini. Untuk membuat peran AWS Fargate Pod eksekusi, lihatPeran IAM Pod eksekusi Amazon EKS.

> **Note**
>
> Jika Anda membuat klaster dengan `eksctl` menggunakan `--fargate` opsi, klaster Anda sudah memiliki peran Pod eksekusi yang dapat Anda temukan di konsol IAM dengan pola `eksctl-`*`my-cluster`*`-FargatePodExecutionRole-`*`ABCDEFGHIJKL`* tersebut. Demikian pula, jika Anda menggunakan `eksctl` untuk membuat profil Fargate Anda, `eksctl` buat peran Pod eksekusi Anda jika belum dibuat.

Buat profil Fargate untuk klaster Anda

Sebelum Anda dapat menjadwalkan Pods yang berjalan di Fargate di cluster Anda, Anda harus menentukan profil Fargate yang menentukan mana yang Pods menggunakan Fargate saat diluncurkan. Untuk informasi selengkapnya, lihat AWS Fargate profil.

> **Note**
>
> Jika Anda membuat cluster Anda dengan `eksctl` menggunakan `--fargate` opsi, maka profil Fargate sudah dibuat untuk klaster Anda dengan pemilih untuk semua Pods di dan ruang nama. `kube-system` `default` Gunakan prosedur berikut untuk membuat profil Fargate untuk namespaces lain yang ingin Anda gunakan dengan Fargate.

Anda dapat membuat profil Fargate menggunakan `eksctl` atau AWS Management Console.

eksctl

Prosedur ini membutuhkan `eksctl` versi `0.175.0` atau yang lebih baru. Anda dapat memeriksa versi Anda dengan perintah berikut:

```
eksctl version
```

Untuk petunjuk tentang cara menginstal atau meningkatkan`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Untuk membuat profil Fargate dengan **`eksctl`**

Buat profil Fargate Anda dengan `eksctl` perintah berikut, ganti masing-masing *`example value`* dengan nilai Anda sendiri. Anda diminta untuk menentukan namespace. Namun, `--labels` opsi tidak diperlukan.

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --name my-fargate-profile \
    --namespace my-kubernetes-namespace \
    --labels key=value
```

Anda dapat menggunakan wildcard tertentu untuk *`my-kubernetes-namespace`* dan *`key=value`* label. Untuk informasi selengkapnya, lihat Wildcard profil Fargate.

AWS Management Console

Untuk membuat profil Fargate untuk cluster dengan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih klaster untuk membuat profil Fargate untuk.
3. Pilih tab Compute.
4. Di bawah Profil Fargate, pilih Tambahkan profil Fargate.
5. Pada halaman profil Konfigurasi Fargate, lakukan hal berikut:
   a. Untuk Nama, masukkan nama untuk profil Fargate Anda. Nama harus unik.
   b. Untuk peran eksekusi Pod, pilih peran Pod eksekusi yang akan digunakan dengan profil Fargate Anda. Hanya peran IAM dengan kepala `eks-fargate-pods.amazonaws.com` layanan yang ditampilkan. Jika Anda tidak melihat peran apa pun yang terdaftar, Anda harus membuatnya. Untuk informasi selengkapnya, lihat Peran IAM Pod eksekusi Amazon EKS.
   c. Ubah Subnet yang dipilih sesuai kebutuhan.

      > **Note**
      >
      > Hanya subnet pribadi yang didukung untuk Pods yang berjalan di Fargate.

   d. Untuk Tandai, Anda dapat menandai profil Fargate Anda secara opsional. Tag ini tidak menyebar ke sumber daya lain yang terkait dengan profil sepertiPods.
   e. Pilih Berikutnya.

6. Pada halaman Podpilihan Konfigurasi, lakukan hal berikut:

   a. Untuk Namespace, masukkan namespace yang cocok. Pods

      - Anda dapat menggunakan ruang nama tertentu untuk mencocokkan, seperti **`kube-system`** atau. **`default`**
      - Anda dapat menggunakan wildcard tertentu (misalnya,**`prod-*`**) untuk mencocokkan beberapa ruang nama (misalnya, `prod-deployment` dan). `prod-test` Untuk informasi selengkapnya, lihat Wildcard profil Fargate.

   b. (Opsional) Tambahkan Kubernetes label ke pemilih. Secara khusus menambahkannya ke salah satu yang perlu Pods dicocokkan dengan namespace yang ditentukan.

      - Anda dapat menambahkan label **`infrastructure: fargate`** ke pemilih sehingga hanya Pods di namespace yang ditentukan yang juga memiliki `infrastructure: fargate` Kubernetes label cocok dengan pemilih.
      - Anda dapat menggunakan wildcard tertentu (misalnya,**`key?: value?`**) untuk mencocokkan beberapa ruang nama (misalnya, `keya: valuea` dan). `keyb: valueb` Untuk informasi selengkapnya, lihat Wildcard profil Fargate.

   c. Pilih Berikutnya.

7. Pada halaman Periksa dan buat, tinjau informasi untuk profil Fargate Anda dan pilih Buat.

Perbarui CoreDNS

Secara default, CoreDNS dikonfigurasi untuk berjalan di infrastruktur Amazon EC2 di kluster Amazon EKS. Jika Anda hanya ingin menjalankan Pods di Fargate di cluster Anda, selesaikan langkah-langkah berikut.

> **Note**
>
> Jika Anda membuat cluster Anda dengan `eksctl` menggunakan `--fargate` opsi, maka Anda dapat melompat keLangkah selanjutnya.

1. Buat profil Fargate untuk CoreDNS dengan perintah berikut. Ganti *`my-cluster`* dengan nama cluster Anda, *`111122223333`* dengan ID akun Anda, *`AmazonEKSFargatePodExecutionRole`* dengan nama peran Pod eksekusi Anda, dan*`0000000000000001`*,*`0000000000000002`*, dan *`0000000000000003`* dengan ID subnet

pribadi Anda. Jika Anda tidak memiliki peran Pod eksekusi, Anda harus membuatnya terlebih dahulu.

> **Important**
>
> Peran ARN tidak dapat menyertakan jalur selain. / Misalnya, jika nama peran Andadevelopment/apps/my-role, Anda perlu mengubahnya menjadi my-role saat menentukan ARN untuk peran tersebut. Format ARN peran harus arn:aws:iam::*111122223333*:role/*role-name*.

```
aws eks create-fargate-profile \
    --fargate-profile-name coredns \
    --cluster-name my-cluster \
    --pod-execution-role-arn
 arn:aws:iam::111122223333:role/AmazonEKSFargatePodExecutionRole \
    --selectors namespace=kube-system,labels={k8s-app=kube-dns} \
    --subnets subnet-0000000000000001 subnet-0000000000000002
 subnet-0000000000000003
```

2. Jalankan perintah berikut untuk menghapus eks.amazonaws.com/compute-type : ec2 anotasi dari. CoreDNS Pods

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

Langkah selanjutnya

- Anda dapat mulai memigrasi aplikasi yang ada untuk berjalan di Fargate dengan alur kerja berikut.
  1. Buat profil Fargate yang cocok dengan Kubernetes namespace dan label aplikasi Anda. Kubernetes
  2. Hapus dan buat ulang yang ada Pods sehingga dijadwalkan di Fargate. Misalnya, perintah berikut memicu peluncuran penerapan. coredns Anda dapat memodifikasi namespace dan jenis penyebaran untuk memperbarui spesifik Anda. Pods

```
kubectl rollout restart -n kube-system deployment coredns
```

- Terapkan Penyeimbangan beban aplikasi pada Amazon EKS untuk mengizinkan objek Ingress untuk Anda Pods berjalan di Fargate.
- Anda dapat menggunakan Vertical Pod Autoscaler untuk mengatur ukuran CPU dan memori awal yang benar untuk Fargate AndaPods, dan kemudian menggunakan Penskala Otomatis Pod Horizontal untuk menskalakannya. Pods Jika Anda ingin Vertical Pod Autoscaler untuk secara otomatis di-deploy ulang Pods ke Fargate dengan kombinasi CPU dan memori yang lebih tinggi, atur mode Vertical Pod Autoscaler ke salah satu atau. `Auto` `Recreate` Ini untuk memastikan fungsionalitas yang benar. Untuk informasi selengkapnya, lihat dokumentasi Vertical Pod Autoscaler pada. GitHub
- Anda dapat mengatur kolektor AWS Distro for OpenTelemetry (ADOT) untuk pemantauan aplikasi dengan mengikuti petunjuk ini.

AWS Fargate profil

> **Important**
>
> AWS Fargate dengan Amazon EKS tidak tersedia di AWS GovCloud (AS-Timur) dan AWS GovCloud (AS-Barat).

Sebelum Anda menjadwalkan Pods Fargate di cluster Anda, Anda harus menentukan setidaknya satu profil Fargate yang menentukan yang Pods menggunakan Fargate saat diluncurkan.

Sebagai administrator, Anda dapat menggunakan profil Fargate untuk mendeklarasikan yang berjalan Pods di Fargate. Anda dapat melakukan ini melalui pemilih profil. Anda dapat menambahkan hingga lima pemilih ke setiap profil. Setiap pemilih harus berisi namespace. Pemilih juga dapat menyertakan label. Bidang label terdiri dari beberapa pasangan nilai kunci opsional. Pod yang cocok dengan pemilih dijadwalkan di Fargate. Pod dicocokkan menggunakan namespace dan label yang ditentukan dalam pemilih. Jika pemilih namespace didefinisikan tanpa label, Amazon EKS mencoba menjadwalkan semua Pods yang berjalan di namespace itu ke Fargate menggunakan profil. Jika a to-be-scheduled Pod cocok dengan salah satu pemilih di profil Fargate, maka Pod itu dijadwalkan di Fargate.

Jika Pod cocok dengan beberapa profil Fargate, Anda dapat menentukan profil mana yang Pod digunakan dengan menambahkan Kubernetes label berikut ke Pod spesifikasi:. `eks.amazonaws.com/fargate-profile: my-fargate-profile` PodHarus cocok dengan pemilih di profil itu untuk dijadwalkan ke Fargate. KubernetesAturan afinitas/anti-afinitas tidak berlaku dan tidak diperlukan dengan Amazon EKS Fargate. Pods

Saat Anda membuat profil Fargate, Anda harus menentukan peran Pod eksekusi. Peran eksekusi ini untuk komponen Amazon EKS yang berjalan di infrastruktur Fargate menggunakan profil. Ini ditambahkan ke Kubernetes Role Based Access Control (RBAC) cluster untuk otorisasi. Dengan begitu, `kubelet` yang berjalan pada infrastruktur Fargate dapat mendaftar dengan cluster Amazon EKS Anda dan muncul di cluster Anda sebagai node. Peran Pod eksekusi juga memberikan izin IAM ke infrastruktur Fargate untuk memungkinkan akses baca ke repositori gambar Amazon ECR. Untuk informasi selengkapnya, lihat Peran IAM Pod eksekusi Amazon EKS.

Profil Fargate tidak dapat diubah. Namun, Anda dapat membuat profil baru yang diperbarui untuk mengganti profil yang ada, dan kemudian menghapus yang asli.

> **Note**
>
> Semua Pods yang berjalan menggunakan profil Fargate dihentikan dan dimasukkan ke dalam status tertunda ketika profil dihapus.

Jika ada profil Fargate dalam klaster yang berada dalam `DELETING` status, Anda harus menunggu sampai setelah profil Fargate dihapus sebelum Anda membuat profil lain di cluster itu.

Amazon EKS dan Fargate Pods tersebar di setiap subnet yang ditentukan dalam profil Fargate. Namun, Anda mungkin berakhir dengan penyebaran yang tidak merata. Jika Anda harus memiliki spread yang merata, gunakan dua profil Fargate. Bahkan spread penting dalam skenario di mana Anda ingin menerapkan dua replika dan tidak ingin downtime. Kami merekomendasikan bahwa setiap profil hanya memiliki satu subnet.

Komponen profil Fargate

Komponen-komponen berikut yang terkandung dalam profil Fargate.

Peran eksekusi pod

Saat kluster Pods Anda aktif AWS Fargate, `kubelet` yang berjalan di infrastruktur Fargate harus melakukan panggilan ke AWS API atas nama Anda. Misalnya, perlu melakukan panggilan untuk

menarik gambar kontainer dari Amazon ECR. Peran Pod eksekusi Amazon EKS memberikan izin IAM untuk melakukan ini.

Ketika Anda membuat profil Fargate, Anda harus menentukan peran Pod eksekusi untuk digunakan dengan Anda. Pods Peran ini ditambahkan ke kontrol akses Kubernetes berbasis peran (RBAC) cluster untuk otorisasi. Ini agar yang berjalan di infrastruktur Fargate dapat mendaftar dengan cluster Amazon EKS Anda dan muncul di cluster Anda sebagai simpul. `kubelet` Untuk informasi selengkapnya, lihat Peran IAM Pod eksekusi Amazon EKS.

Subnet

ID subnet yang akan diluncurkan Pods menggunakan profil ini. Saat ini, Pods yang berjalan di Fargate tidak diberi alamat IP publik. Oleh karena itu, hanya subnet pribadi tanpa rute langsung ke Internet Gateway yang diterima untuk parameter ini.

Selektor

Penyeleksi yang cocok Pods untuk menggunakan profil Fargate ini. Anda dapat menentukan hingga lima penyeleksi dalam profil Fargate. Penyeleksi memiliki komponen-komponen berikut:

- Namespace – Anda harus menentukan namespace untuk pemilih. Pemilih hanya cocok dengan Pods yang dibuat di namespace ini. Namun, Anda dapat membuat beberapa penyeleksi untuk menargetkan beberapa ruang nama.
- Label - Anda dapat secara opsional menentukan Kubernetes label yang cocok untuk pemilih. Pemilih hanya cocok Pods yang memiliki semua label yang ditentukan dalam pemilih.

Wildcard profil Fargate

Selain karakter yang diizinkan olehKubernetes, Anda diizinkan untuk menggunakan **`*`** dan **`?`** dalam kriteria pemilih untuk ruang nama, kunci label, dan nilai label:

- **`*`**mewakili tidak ada, satu, atau beberapa karakter. Misalnya, **`prod*`** dapat mewakili `prod` dan`prod-metrics`.
- **`?`**mewakili satu karakter (misalnya, **`value?`** dapat mewakili`valuea`). Namun, itu tidak dapat mewakili `value` dan`value-a`, karena hanya `?` dapat mewakili tepat satu karakter.

Karakter wildcard ini dapat digunakan dalam posisi apa pun dan dalam kombinasi (misalnya,, **`prod*dev`**, dan**`frontend*?`**). Wildcard lain dan bentuk pencocokan pola, seperti ekspresi reguler, tidak didukung.

Jika ada beberapa profil yang cocok untuk namespace dan label dalam Pod spesifikasi, Fargate mengambil profil berdasarkan pengurutan alfanumerik berdasarkan nama profil. Misalnya, jika profil A (dengan nama`beta-workload`) dan profil B (dengan nama`prod-workload`) memiliki pemilih yang cocok Pods untuk diluncurkan, Fargate memilih profil A `beta-workload` () untuk file. Pods PodsMemiliki label dengan profil A pada Pods (misalnya,`eks.amazonaws.com/fargate-profile=beta-workload`).

Jika Anda ingin memigrasikan Pods Fargate yang ada ke profil baru yang menggunakan wildcard, ada dua cara untuk melakukannya:

- Buat profil baru dengan penyeleksi yang cocok, lalu hapus profil lama. Pod yang diberi label dengan profil lama dijadwal ulang ke profil baru yang cocok.
- Jika Anda ingin memigrasikan beban kerja tetapi tidak yakin label Fargate apa yang ada di setiap FargatePod, Anda dapat menggunakan metode berikut. Buat profil baru dengan nama yang mengurutkan secara alfanumerik terlebih dahulu di antara profil di cluster yang sama. Kemudian, daur ulang Pods Fargate yang perlu dimigrasikan ke profil baru.

Membuat profil Fargate

Topik ini menjelaskan cara membuat profil Fargate. Anda juga harus telah membuat peran Pod eksekusi untuk digunakan untuk profil Fargate Anda. Untuk informasi lebih lanjut, lihatPeran IAM Pod eksekusi Amazon EKS. Podsyang berjalan di Fargate hanya didukung pada subnet pribadi dengan akses gateway NAT Layanan AWS, tetapi bukan rute langsung ke Internet Gateway. Ini agar VPC cluster Anda harus memiliki subnet pribadi yang tersedia. Anda dapat membuat profil dengan `eksctl` atau AWS Management Console.

Prosedur ini membutuhkan `eksctl` versi `0.175.0` atau yang lebih baru. Anda dapat memeriksa versi Anda dengan perintah berikut:

```
eksctl version
```

Untuk petunjuk tentang cara menginstal atau meningkatkan`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

eksctl

Untuk membuat profil Fargate dengan **`eksctl`**

Buat profil Fargate Anda dengan `eksctl` perintah berikut, ganti masing-masing *`example value`* dengan nilai Anda sendiri. Anda diminta untuk menentukan namespace. Namun, `--labels` opsi tidak diperlukan.

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --name my-fargate-profile \
    --namespace my-kubernetes-namespace \
    --labels key=value
```

Anda dapat menggunakan wildcard tertentu untuk *`my-kubernetes-namespace`* dan *`key=value`* label. Untuk informasi selengkapnya, lihat Wildcard profil Fargate.

AWS Management Console

Untuk membuat profil Fargate untuk cluster dengan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih klaster untuk membuat profil Fargate untuk.
3. Pilih tab Compute.
4. Di bawah Profil Fargate, pilih Tambahkan profil Fargate.
5. Pada halaman profil Konfigurasi Fargate, lakukan hal berikut:
   a. Untuk Nama, masukkan nama unik untuk profil Fargate Anda, seperti ***`my-profile`***.
   b. Untuk peran eksekusi Pod, pilih peran Pod eksekusi yang akan digunakan dengan profil Fargate Anda. Hanya peran IAM dengan kepala `eks-fargate-pods.amazonaws.com` layanan yang ditampilkan. Jika Anda tidak melihat peran apa pun yang terdaftar, Anda harus membuatnya. Untuk informasi selengkapnya, lihat Peran IAM Pod eksekusi Amazon EKS.
   c. Ubah Subnet yang dipilih sesuai kebutuhan.

      > **Note**
      >
      > Hanya subnet pribadi yang didukung untuk Pods yang berjalan di Fargate.

   d. Untuk Tandai, Anda dapat menandai profil Fargate Anda secara opsional. Tag ini tidak menyebar ke sumber daya lain yang terkait dengan profil, sepertiPods.
   e. Pilih Berikutnya.

6. Pada halaman Podpilihan Konfigurasi, lakukan hal berikut:

   a. Untuk Namespace, masukkan namespace yang cocok. Pods

      - Anda dapat menggunakan ruang nama tertentu untuk mencocokkan, seperti **`kube-system`** atau. **`default`**
      - Anda dapat menggunakan wildcard tertentu (misalnya,**`prod-*`**) untuk mencocokkan beberapa ruang nama (misalnya, `prod-deployment` dan). `prod-test` Untuk informasi selengkapnya, lihat Wildcard profil Fargate.

   b. (Opsional) Tambahkan Kubernetes label ke pemilih. Secara khusus, tambahkan ke salah satu yang perlu Pods dicocokkan dengan namespace yang ditentukan.

      - Anda dapat menambahkan label **`infrastructure: fargate`** ke pemilih sehingga hanya Pods di namespace yang ditentukan yang juga memiliki `infrastructure: fargate` Kubernetes label cocok dengan pemilih.
      - Anda dapat menggunakan wildcard tertentu (misalnya,**`key?: value?`**) untuk mencocokkan beberapa ruang nama (misalnya, `keya: valuea` dan). `keyb: valueb` Untuk informasi selengkapnya, lihat Wildcard profil Fargate.

   c. Pilih Berikutnya.

7. Pada halaman Tinjau dan buat, tinjau informasi untuk profil Fargate Anda dan pilih Buat.

Menghapus profil Fargate

Topik ini menjelaskan cara menghapus profil Fargate.

Saat Anda menghapus profil Fargate, semua Pods yang dijadwalkan ke Fargate dengan profil akan dihapus. Jika itu Pods cocok dengan profil Fargate lain, maka mereka dijadwalkan di Fargate dengan profil itu. Jika mereka tidak lagi cocok dengan profil Fargate, maka mereka tidak dijadwalkan ke Fargate dan mungkin tetap tertunda.

Hanya satu profil Fargate dalam sebuah klaster yang dapat memiliki status `DELETING` pada satu waktu. Tunggu profil Fargate selesai dihapus sebelum Anda dapat menghapus profil lain di cluster itu.

Anda dapat menghapus profil dengan`eksctl`, file AWS Management Console, atau file AWS CLI. Pilih tab dengan nama alat yang ingin Anda gunakan untuk menghapus profil Anda.

eksctl

Untuk menghapus profil Fargate dengan **`eksctl`**

Gunakan perintah berikut untuk menghapus profil dari klaster. Ganti setiap *example value* dengan nilai-nilai Anda sendiri.

```
eksctl delete fargateprofile  --name my-profile --cluster my-cluster
```

AWS Management Console

Untuk menghapus profil Fargate dari cluster dengan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pada panel navigasi sebelah kiri, pilih Klaster. Dalam daftar cluster, pilih cluster tempat Anda ingin menghapus profil Fargate.
3. Pilih tab Compute.
4. Pilih profil Fargate yang akan dihapus, lalu pilih Hapus.
5. Pada halaman Hapus profil Fargate, masukkan nama profil, lalu pilih Hapus.

AWS CLI

Untuk menghapus profil Fargate dengan AWS CLI

Gunakan perintah berikut untuk menghapus profil dari klaster. Ganti setiap *example value* dengan nilai-nilai Anda sendiri.

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

Konfigurasi Fargate Pod

> **Important**
>
> AWS Fargatedengan Amazon EKS tidak tersedia di AWS GovCloud (AS-Timur) dan AWS GovCloud (AS-Barat).

Bagian ini menjelaskan beberapa detail Pod konfigurasi unik untuk dijalankan Kubernetes PodsAWS Fargate.

PodCPU dan memori

DenganKubernetes, Anda dapat menentukan permintaan, jumlah vCPU minimum, dan sumber daya memori yang dialokasikan ke setiap kontainer dalam file. Pod Podsdijadwalkan oleh Kubernetes untuk memastikan bahwa setidaknya sumber daya yang diminta untuk masing-masing Pod tersedia pada sumber daya komputasi. Untuk informasi selengkapnya, lihat Mengelola sumber daya komputasi untuk kontainer dalam Kubernetes dokumentasi.

> **Note**
>
> Karena Amazon EKS Fargate hanya menjalankan satu Pod per node, skenario pengusiran Pods jika sumber daya lebih sedikit tidak terjadi. Semua Amazon EKS Fargate Pods berjalan dengan prioritas terjamin, sehingga CPU dan memori yang diminta harus sama dengan batas untuk semua kontainer. Untuk informasi selengkapnya, lihat Mengkonfigurasi Kualitas Layanan untuk Pods Kubernetes dokumentasi.

Ketika Pods dijadwalkan di Fargate, vCPU dan reservasi memori dalam Pod spesifikasi menentukan berapa banyak CPU dan memori untuk disediakan untuk. Pod

- Permintaan maksimum dari setiap kontainer Init digunakan untuk menentukan permintaan Init vCPU dan memori persyaratan.
- Permintaan untuk semua kontainer berjalan lama ditambahkan untuk menentukan lama berjalan permintaan vCPU dan memori persyaratan.
- Yang lebih besar dari dua nilai sebelumnya dipilih untuk vCPU dan permintaan memori untuk digunakan untuk Anda. Pod
- Fargate menambahkan 256 MB ke reservasi Pod memori masing-masing untuk Kubernetes komponen yang diperlukan (`kubelet`,`kube-proxy`, dan`containerd`).

Fargate membulatkan ke konfigurasi komputasi berikut yang paling cocok dengan jumlah permintaan vCPU dan memori untuk memastikan Pods selalu memiliki sumber daya yang mereka butuhkan untuk dijalankan.

Jika Anda tidak menentukan kombinasi vCPU dan memori, maka kombinasi terkecil yang tersedia digunakan (0,25 vCPU dan memori 0,5 GB).

Tabel berikut menunjukkan kombinasi vCPU dan memori yang tersedia untuk Pods berjalan di Fargate.

Nilai vCPU	Nilai memori
0,25 vCPU	0,5 GB, 1 GB, 2 GB
0,5 vCPU	1 GB, 2 GB, 3 GB, 4 GB
1 vCPU	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB
2 vCPU	Antara 4 GB dan 16 GB dalam kenaikan 1-GB
4 vCPU	Antara 8 GB dan 30 GB dalam tambahan 1-GB
8 vCPU	Antara 16 GB dan 60 GB dengan peningkatan 4-GB
16 vCPU	Antara 32 GB dan 120 GB dengan peningkatan 8-GB

Memori tambahan yang disediakan untuk Kubernetes komponen dapat menyebabkan tugas Fargate dengan lebih banyak vCPU daripada yang diminta untuk disediakan. Misalnya, permintaan untuk 1 vCPU dan 8 GB memori akan membuat memori sebesar 256 MB ditambahkan ke permintaan memori, dan akan menyediakan tugas Fargate dengan 2 vCPUs dan 9 GB memori, karena tidak ada tugas dengan 1 vCPU dan 9 GB memori.

Tidak ada korelasi antara ukuran Pod berjalan di Fargate dan ukuran node yang dilaporkan Kubernetes oleh with. `kubectl get nodes` Ukuran node yang dilaporkan seringkali lebih besar dari Pod kapasitas. Anda dapat memverifikasi Pod kapasitas dengan perintah berikut. Ganti *`default`* dengan Pod namespace Anda dan *`pod-name`* dengan nama Anda. Pod

```
kubectl describe pod --namespace default pod-name
```

Contoh output adalah sebagai berikut.

```
[...]
annotations:
    CapacityProvisioned: 0.25vCPU 0.5GB
[...]
```

`CapacityProvisioned`Anotasi mewakili Pod kapasitas yang dipaksakan dan menentukan biaya Pod pengoperasian Anda di Fargate. Untuk informasi harga untuk konfigurasi komputasi, lihat AWS Fargate Harga.

Penyimpanan Fargate

Sebuah Pod berjalan di Fargate secara otomatis memasang sistem file Amazon EFS. Anda tidak dapat menggunakan penyediaan volume persisten dinamis dengan simpul Fargate, tetapi Anda dapat menggunakan penyediaan statis. Untuk informasi selengkapnya, lihat Driver Amazon EFS CSI di GitHub.

Saat disediakan, masing-masing yang Pod berjalan di Fargate menerima penyimpanan sementara 20 GiB default. Jenis penyimpanan ini dihapus setelah Pod berhenti. Baru Pods diluncurkan ke Fargate memiliki enkripsi volume penyimpanan sementara yang diaktifkan secara default. PodPenyimpanan sementara dienkripsi dengan algoritma enkripsi AES-256 menggunakan kunci terkelola. AWS Fargate

> **Note**
>
> Penyimpanan default yang dapat digunakan untuk Amazon EKS Pods yang berjalan di Fargate kurang dari 20 GiB. Ini karena beberapa ruang digunakan oleh `kubelet` dan Kubernetes modul lain yang dimuat di dalamPod.

Anda dapat meningkatkan jumlah total penyimpanan sementara hingga maksimum 175 GiB. Untuk mengonfigurasi ukuran denganKubernetes, tentukan permintaan `ephemeral-storage` sumber daya untuk setiap kontainer dalam filePod. Ketika Kubernetes jadwalPods, ini memastikan bahwa jumlah permintaan sumber daya untuk masing-masing Pod kurang dari kapasitas tugas Fargate. Untuk informasi selengkapnya, lihat Manajemen Sumber Daya untuk Pods dan Kontainer dalam Kubernetes dokumentasi.

Amazon EKS Fargate menyediakan lebih banyak penyimpanan sementara daripada yang diminta untuk tujuan penggunaan sistem. Misalnya, permintaan 100 GiB akan menyediakan tugas Fargate dengan penyimpanan sementara 115 GiB.

Penambalan OS Fargate

> **Important**
>
> AWS Fargatedengan Amazon EKS tidak tersedia di AWS GovCloud (AS-Timur) dan AWS GovCloud (AS-Barat).

Amazon EKS secara berkala menambal OS untuk AWS Fargate node agar tetap aman. Sebagai bagian dari proses patching, kami mendaur ulang node untuk menginstal patch OS. Pembaruan dicoba dengan cara yang menciptakan dampak paling kecil pada layanan Anda. Namun, jika Pods tidak berhasil diusir, ada kalanya mereka harus dihapus. Berikut ini adalah tindakan yang dapat Anda lakukan untuk meminimalkan potensi gangguan:

- Tetapkan anggaran Pod gangguan yang sesuai (PDB) untuk mengontrol jumlah yang turun secara Pods bersamaan.
- Buat EventBridge aturan Amazon untuk menangani penggusuran yang gagal sebelum Pods dihapus.
- Buat konfigurasi notifikasi di Pemberitahuan AWS Pengguna.

Amazon EKS bekerja sama dengan Kubernetes komunitas untuk membuat perbaikan bug dan patch keamanan tersedia secepat mungkin. Semua Fargate Pods dimulai pada versi Kubernetes patch terbaru, yang tersedia dari Amazon EKS untuk Kubernetes versi cluster Anda. Jika Anda memiliki versi patch Pod yang lebih lama, Amazon EKS mungkin mendaur ulangnya untuk memperbaruinya ke versi terbaru. Ini memastikan bahwa Anda Pods dilengkapi dengan pembaruan keamanan terbaru. Dengan begitu, jika ada masalah Common Vulnerabilities and Exposures (CVE) yang kritis, Anda tetap up to date untuk mengurangi risiko keamanan.

Untuk membatasi jumlah Pods yang turun pada satu waktu saat Pods didaur ulang, Anda dapat mengatur anggaran Pod gangguan (PDB). Anda dapat menggunakan PDB untuk menentukan ketersediaan minimum berdasarkan persyaratan masing-masing aplikasi Anda sambil tetap mengizinkan pembaruan terjadi. Untuk informasi selengkapnya, lihat Menentukan Anggaran Gangguan untuk Aplikasi Anda di Dokumentasi. Kubernetes

Amazon EKS menggunakan Eviction API untuk menguras dengan aman Pod sambil menghormati PDB yang Anda tetapkan untuk aplikasi. Pod diusir oleh Availability Zone untuk meminimalkan

dampak. Jika penggusuran berhasil, yang baru Pod mendapatkan tambalan terbaru dan tidak diperlukan tindakan lebih lanjut.

Ketika penggusuran Pod gagal, Amazon EKS mengirimkan acara ke akun Anda dengan rincian tentang penggusuran Pods yang gagal itu. Anda dapat menindaklanjuti pesan sebelum waktu penghentian yang dijadwalkan. Waktu spesifik bervariasi berdasarkan urgensi tambalan. Ketika tiba waktunya, Amazon EKS mencoba untuk mengusir lagi. Pods Namun, kali ini acara baru tidak dikirim jika penggusuran gagal. Jika penggusuran gagal lagi, yang sudah Pods ada dihapus secara berkala sehingga yang baru Pods dapat memiliki tambalan terbaru.

Berikut ini adalah contoh peristiwa yang diterima ketika Pod penggusuran gagal. Ini berisi rincian tentang cluster, Pod nama, Pod namespace, profil Fargate, dan waktu penghentian yang dijadwalkan.

```
{
    "version": "0",
    "id": "12345678-90ab-cdef-0123-4567890abcde",
    "detail-type": "EKS Fargate Pod Scheduled Termination",
    "source": "aws.eks",
    "account": "111122223333",
    "time": "2021-06-27T12:52:44Z",
    "region": "region-code",
    "resources": [
        "default/my-database-deployment"
    ],
    "detail": {
        "clusterName": "my-cluster",
        "fargateProfileName": "my-fargate-profile",
        "podName": "my-pod-name",
        "podNamespace": "default",
        "evictErrorMessage": "Cannot evict pod as it would violate the pod's disruption
 budget",
        "scheduledTerminationTime": "2021-06-30T12:52:44.832Z[UTC]"
    }
}
```

Selain itu, memiliki beberapa PDB yang terkait dengan a Pod dapat menyebabkan peristiwa kegagalan penggusuran. Acara ini mengembalikan pesan galat berikut.

```
"evictErrorMessage": "This pod has multiple PodDisruptionBudget, which the eviction
 subresource does not support",
```

Anda dapat membuat tindakan yang diinginkan berdasarkan acara ini. Misalnya, Anda dapat menyesuaikan anggaran Pod gangguan (PDB) Anda untuk mengontrol bagaimana penggusuranPods. Lebih khusus lagi, misalkan Anda memulai dengan PDB yang menentukan persentase target Pods yang tersedia. Sebelum paksa Pods Anda dihentikan selama peningkatan, Anda dapat menyesuaikan PDB ke persentase yang berbeda dari. Pods Untuk menerima acara ini, Anda harus membuat EventBridge aturan Amazon di Akun AWS dan milik cluster Wilayah AWS tersebut. Aturan harus menggunakan pola Kustom berikut. Untuk informasi selengkapnya, lihat Membuat EventBridge aturan Amazon yang bereaksi terhadap peristiwa di Panduan EventBridge Pengguna Amazon.

```
{
  "source": ["aws.eks"],
  "detail-type": ["EKS Fargate Pod Scheduled Termination"]
}
```

Target yang sesuai dapat ditetapkan untuk acara untuk menangkapnya. Untuk daftar lengkap target yang tersedia, lihat EventBridge target Amazon di Panduan EventBridge Pengguna Amazon. Anda juga dapat membuat konfigurasi notifikasi di Pemberitahuan AWS Pengguna. Saat menggunakan AWS Management Console untuk membuat notifikasi, di bawah Aturan Acara, pilih Elastic Kubernetes Service (EKS) untuk Layanan AWSnama dan EKS Fargate Pod Terjadwal Terminasi untuk tipe Event. Untuk informasi selengkapnya, lihat Memulai Pemberitahuan AWS Pengguna di Panduan AWS Pengguna Pemberitahuan Pengguna.

Metrik Fargate

> **Important**
>
> AWS Fargatedengan Amazon EKS tidak tersedia di AWS GovCloud (AS-Timur) dan AWS GovCloud (AS-Barat).

Anda dapat mengumpulkan metrik sistem dan metrik CloudWatch penggunaan untuk. AWS Fargate

Metrik aplikasi

Untuk aplikasi yang berjalan di Amazon EKS danAWS Fargate, Anda dapat menggunakan AWS Distro for OpenTelemetry (ADOT). ADOT memungkinkan Anda mengumpulkan metrik sistem dan mengirimkannya ke dasbor CloudWatch Container Insights. Untuk memulai ADOT untuk aplikasi

yang berjalan di Fargate, lihat CloudWatch Menggunakan Wawasan Kontainer AWS dengan Distro OpenTelemetry untuk dokumentasi ADOT.

Metrik penggunaan

Anda dapat menggunakan metrik CloudWatch penggunaan untuk memberikan visibilitas ke dalam penggunaan sumber daya akun Anda. Gunakan metrik ini untuk memvisualisasikan penggunaan layanan Anda saat ini pada CloudWatch grafik dan dasbor.

Metrik penggunaan AWS Fargate sesuai dengan service quotas AWS. Anda dapat mengonfigurasi alarm yang memperingatkan ketika penggunaan Anda mendekati kuota layanan. Untuk informasi lebih lanjut tentang kuota layanan Fargate, lihat Amazon EKS service quotas.

AWS Fargate memuat metrik berikut ini di namespace `AWS/Usage`.

Metrik	Deskripsi
`ResourceCount`	Jumlah sumber daya tertentu yang berjalan di akun Anda. Sumber daya ditentukan oleh dimensi yang terkait dengan metrik.

Dimensi berikut ini digunakan untuk memilah penggunaan metrik yang dipublikasikan oleh AWS Fargate.

Dimensi	Deskripsi
`Service`	Nama dari layanan AWS yang berisi sumber daya. Untuk metrik penggunaan AWS Fargate, nilai untuk dimensi ini adalah `Fargate`.
`Type`	Jenis Entitas yang Dilaporkan Saat ini, satu-satunya nilai yang valid untuk penggunaan metrik AWS Fargate adalah `Resource`.
`Resource`	Jenis sumber daya yang sedang berjalan. Saat ini, AWS Fargate mengembalikan informasi tentang penggunaan Sesuai Permintaan Fargate Anda. Nilai sumber daya untuk penggunaan Sesuai Permintaan Fargate adalah `OnDemand`.

Dimensi	Deskripsi
	Note Penggunaan Fargate On-Demand menggabungkan Amazon EKS menggunakan Pods Fargate, tugas Amazon ECS menggunakan jenis peluncuran Fargate dan tugas Amazon ECS menggunakan penyedia kapasitas. `FARGATE`
`Class`	Kelas sumber daya yang ditelusuri. Saat ini, AWS Fargate tidak menggunakan dimensi kelas.

Membuat CloudWatch alarm untuk memantau metrik penggunaan sumber daya Fargate

AWS Fargatemenyediakan metrik CloudWatch penggunaan yang sesuai dengan kuota AWS layanan untuk penggunaan sumber daya Fargate On-Demand. Di konsol Service Quotas, Anda dapat memvisualisasikan penggunaan Anda pada grafik. Anda juga dapat mengonfigurasi alarm yang memperingatkan Anda ketika penggunaan mendekati kuota layanan. Untuk informasi selengkapnya, lihat Metrik Fargate.

Gunakan langkah-langkah berikut untuk membuat CloudWatch alarm berdasarkan metrik penggunaan sumber daya Fargate.

Untuk membuat alarm berdasarkan pada kuota penggunaan Fargate Anda (AWS Management Console)

1. Buka konsol Service Quotas di https://console.aws.amazon.com/servicequotas/.
2. Di panel navigasi kiri, pilih AWSlayanan.
3. Dari daftar Layanan AWS, cari dan pilih AWS Fargate.
4. Dalam daftar Kuota Layanan, pilih kuota penggunaan Fargate yang ingin Anda buat alarm.
5. Di bagian CloudWatch alarm Amazon, pilih Buat.
6. Untuk Ambang batas Alarm, pilih persentase nilai kuota yang ingin Anda tetapkan sebagai nilai alarm.
7. Untuk Nama alarm, masukkan nama untuk alarm, lalu pilih Buat.

Pencatatan log Fargate

> **Important**
>
> AWS Fargate dengan Amazon EKS tidak tersedia di AWS GovCloud (AS-Timur) dan AWS GovCloud (AS-Barat).

Amazon EKS di Fargate menawarkan router log bawaan berdasarkan. Fluent Bit Ini berarti Anda tidak secara eksplisit menjalankan Fluent Bit wadah sebagai sespan, tetapi Amazon menjalankannya untuk Anda. Yang harus Anda lakukan adalah mengkonfigurasi router log. Konfigurasi terjadi melalui dedicated `ConfigMap` yang harus memenuhi kriteria berikut:

- Bernama `aws-logging`
- Dibuat di namespace khusus yang disebut `aws-observability`
- Tidak dapat melebihi 5300 karakter.

Setelah Anda membuat`ConfigMap`, Amazon EKS di Fargate secara otomatis mendeteksi dan mengonfigurasi router log dengannya. Fargate menggunakan versi AWS forFluent Bit, distribusi yang sesuai dengan hulu yang dikelola oleh. Fluent Bit AWS Untuk informasi lebih lanjut, lihat AWS untuk Fluent Bit di GitHub.

Router log memungkinkan Anda untuk menggunakan luasnya layanan di AWS untuk analisis log dan penyimpanan. Anda dapat melakukan streaming log dari Fargate langsung ke Amazon, CloudWatch Amazon OpenSearch Service. Anda juga dapat melakukan streaming log ke tujuan seperti Amazon S3, Amazon Kinesis Data Streams, dan alat mitra melalui Amazon Data Firehose.

Prasyarat

- Profil Fargate yang ada yang menentukan Kubernetes namespace yang ada yang Anda gunakan Fargate. Pods Untuk informasi selengkapnya, lihat Buat profil Fargate untuk klaster Anda.
- Peran Pod eksekusi Fargate yang ada. Untuk informasi selengkapnya, lihat Buat peran eksekusi Fargate Pod.

Konfigurasi router log

Untuk mengkonfigurasi router log

Pada langkah-langkah berikut, ganti masing-masing *example value* dengan nilai Anda sendiri.

1. Buat Kubernetes namespace khusus bernama. `aws-observability`

   a. Simpan konten berikut ini ke file bernama *aws-observability-namespace*`.yaml` pada komputer Anda. Nilai untuk `name` harus `aws-observability` dan `aws-observability: enabled` label diperlukan.

   ```
   kind: Namespace
   apiVersion: v1
   metadata:
     name: aws-observability
     labels:
       aws-observability: enabled
   ```

   b. Buat namespace.

   ```
   kubectl apply -f aws-observability-namespace.yaml
   ```

2. Buat `ConfigMap` dengan nilai `Fluent Conf` data untuk mengirimkan log kontainer ke tujuan. Fluent Conf adalahFluent Bit, yang merupakan bahasa konfigurasi prosesor log yang cepat dan ringan yang digunakan untuk merutekan log kontainer ke tujuan log pilihan Anda. Untuk informasi selengkapnya, lihat File Konfigurasi dalam Fluent Bit dokumentasi.

   > **Important**
   >
   > Bagian utama yang termasuk dalam tipikal `Fluent Conf` adalah`Service`,`Input`,`Filter`, dan`Output`. Namun, router log Fargate hanya menerima:
   >
   > - `Output`Bagian `Filter` dan.
   > - Sebuah `Parser` bagian.
   >
   > Jika Anda memberikan bagian lain, mereka akan ditolak.

Router log Fargate mengelola bagian `Service` dan`Input`. Ini memiliki `Input` bagian berikut, yang tidak dapat dimodifikasi dan tidak diperlukan di Anda`ConfigMap`. Namun, Anda bisa mendapatkan wawasan darinya, seperti batas buffer memori dan tag yang diterapkan untuk log.

```
[INPUT]
    Name tail
    Buffer_Max_Size 66KB
    DB /var/log/flb_kube.db
    Mem_Buf_Limit 45MB
    Path /var/log/containers/*.log
    Read_From_Head On
    Refresh_Interval 10
    Rotate_Wait 30
    Skip_Long_Lines On
    Tag kube.*
```

Saat membuat`ConfigMap`, pertimbangkan aturan berikut yang digunakan Fargate untuk memvalidasi bidang:

- `[FILTER]`,`[OUTPUT]`, dan seharusnya `[PARSER]` ditentukan di bawah setiap kunci yang sesuai. Contohnya, `[FILTER]` harus berada dalam `filters.conf`. Anda dapat memiliki satu atau lebih `[FILTER]` s di bawah`filters.conf`. `[PARSER]`Bagian `[OUTPUT]` dan juga harus berada di bawah kunci yang sesuai. Dengan menentukan beberapa `[OUTPUT]` bagian, Anda dapat merutekan log Anda ke tujuan yang berbeda secara bersamaan.
- Fargate memvalidasi kunci yang diperlukan untuk setiap bagian. `Name` dan `match` diperlukan untuk setiap `[FILTER]` dan `[OUTPUT]`. `Name` dan `format` diperlukan untuk setiap `[PARSER]`. Kunci peka terhadap huruf besar dan kecil.
- Variabel lingkungan seperti `${ENV_VAR}` tidak diizinkan di`ConfigMap`.
- Indentasi harus sama dengan arahan atau pasangan nilai kunci dalam setiap `filters.conf`, `output.conf`, dan `parsers.conf`. Pasangan nilai kunci yang harus menjorok lebih dari arahan.
- Fargate memvalidasi terhadap filter yang didukung berikut:`grep`,,`parser`,,`record_modifier`, `rewrite_tag` `throttlenest`, `modify` dan. `kubernetes`
- Fargate memvalidasi output yang didukung berikut: `es`, `firehose`, `kinesis_firehose`, `cloudwatch`, `cloudwatch_logs`, dan `kinesis`.

- Setidaknya satu `Output` plugin yang didukung harus disediakan di `ConfigMap` untuk mengaktifkan logging. `Filter`dan `Parser` tidak diperlukan untuk mengaktifkan logging.

Anda juga dapat menjalankan Fluent Bit di Amazon EC2 menggunakan konfigurasi yang diinginkan untuk memecahkan masalah apa pun yang timbul dari validasi. Buat Anda `ConfigMap` menggunakan salah satu contoh berikut.

> **Important**
>
> Amazon EKS Fargate logging tidak mendukung konfigurasi dinamis. `ConfigMaps` Setiap perubahan `ConfigMaps` diterapkan Pods hanya untuk yang baru. Perubahan tidak diterapkan pada yang adaPods.

Buat `ConfigMap` menggunakan contoh untuk tujuan log yang Anda inginkan.

> **Note**
>
> Anda juga dapat menggunakan Amazon Kinesis Data Streams untuk tujuan log Anda. Jika Anda menggunakan Kinesis Data Streams, pastikan bahwa peran eksekusi pod telah `kinesis:PutRecords` diberikan izin. Untuk informasi selengkapnya, lihat Izin Amazon Kinesis Data Streams Fluent Bitdi: Manual Resmi.

CloudWatch

Untuk membuat **`ConfigMap`** untuk CloudWatch

Anda memiliki dua opsi output saat menggunakan CloudWatch:

- Plugin keluaran yang ditulis dalam C
- Plugin keluaran yang ditulis dalam Golang

Contoh berikut menunjukkan cara menggunakan `cloudwatch_logs` plugin untuk mengirim log ke CloudWatch.

1. Simpan konten berikut ini ke file bernama *aws-logging-cloudwatch-configmap*.yaml. Ganti *region-code* dengan tempat Wilayah AWS cluster Anda berada. Parameter di bawah [OUTPUT] ini diperlukan.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  flb_log_cw: "false"  # Set to true to ship Fluent Bit process logs to
 CloudWatch.
  filters.conf: |
    [FILTER]
        Name parser
        Match *
        Key_name log
        Parser crio
    [FILTER]
        Name kubernetes
        Match kube.*
        Merge_Log On
        Keep_Log Off
        Buffer_Size 0
        Kube_Meta_Cache_TTL 300s
  output.conf: |
    [OUTPUT]
        Name cloudwatch_logs
        Match   kube.*
        region region-code
        log_group_name my-logs
        log_stream_prefix from-fluent-bit-
        log_retention_days 60
        auto_create_group true
  parsers.conf: |
    [PARSER]
        Name crio
        Format Regex
        Regex ^(?<time>[^ ]+) (?<stream>stdout|stderr) (?<logtag>P|F) (?
<log>.*)$
        Time_Key    time
        Time_Format %Y-%m-%dT%H:%M:%S.%L%z
```

2. Menerapkan manifes ke klaster Anda.

```
kubectl apply -f aws-logging-cloudwatch-configmap.yaml
```

3. Unduh kebijakan CloudWatch IAM ke komputer Anda. Anda juga dapat melihat kebijakan di GitHub.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-logging-examples/mainline/examples/fargate/cloudwatchlogs/permissions.json
```

Amazon OpenSearch Service

Untuk membuat OpenSearch Layanan **ConfigMap** untuk Amazon

Jika Anda ingin mengirim log ke Amazon OpenSearch Service, Anda dapat menggunakan es output, yang merupakan plugin yang ditulisC. Contoh berikut menunjukkan cara menggunakan plugin untuk mengirim log ke OpenSearch.

1. Simpan konten berikut ini ke file bernama *aws-logging-opensearch-configmap*.yaml. Ganti setiap *example value* dengan nilai-nilai Anda sendiri.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  output.conf: |
    [OUTPUT]
      Name  es
      Match *
      Host  search-example-gjxdcilagiprbglqn42jsty66y.region-code.es.amazonaws.com
      Port  443
      Index example
      Type  example_type
      AWS_Auth On
      AWS_Region region-code
      tls   On
```

2. Menerapkan manifes ke klaster Anda.

```
kubectl apply -f aws-logging-opensearch-configmap.yaml
```

3. Unduh kebijakan OpenSearch IAM ke komputer Anda. Anda juga dapat melihat kebijakan di GitHub.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-
fluent-logging-examples/mainline/examples/fargate/amazon-elasticsearch/
permissions.json
```

Pastikan kontrol akses OpenSearch Dashboard dikonfigurasi dengan benar. OpenSearch Dasbor `all_access role` in harus memiliki peran eksekusi Pod Fargate dan peran IAM yang dipetakan. Pemetaan yang sama harus dilakukan untuk `security_manager` peran tersebut. Anda dapat menambahkan pemetaan sebelumnya dengan memilihMenu, kemudian, lalu `SecurityRoles`, dan kemudian memilih peran masing-masing. Untuk informasi selengkapnya, lihat Bagaimana cara memecahkan masalah CloudWatch Log sehingga dialirkan ke domain Amazon ES saya? .

Firehose

Untuk membuat **`ConfigMap`** untuk Firehose

Anda memiliki dua opsi output saat mengirim log ke Firehose:

- `kinesis_firehose` — Plugin output ditulis dalam C.
- `firehose` — Plugin output ditulis dalam Golang.

Contoh berikut menunjukkan cara menggunakan `kinesis_firehose` plugin untuk mengirim log ke Firehose.

1. Simpan konten berikut ini ke file bernama *`aws-logging-firehose-configmap`*`.yaml`. Ganti *`region-code`* dengan tempat Wilayah AWS cluster Anda berada.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
```

```
  output.conf: |
    [OUTPUT]
     Name  kinesis_firehose
     Match *
     region region-code
     delivery_stream my-stream-firehose
```

2. Menerapkan manifes ke klaster Anda.

```
kubectl apply -f aws-logging-firehose-configmap.yaml
```

3. Unduh kebijakan Firehose IAM ke komputer Anda. Anda juga dapat melihat kebijakan di GitHub.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-
logging-examples/mainline/examples/fargate/kinesis-firehose/permissions.json
```

3. Buat kebijakan IAM dari file kebijakan yang Anda unduh di langkah sebelumnya.

```
aws iam create-policy --policy-name eks-fargate-logging-policy --policy-document
 file://permissions.json
```

4. Lampirkan kebijakan IAM ke peran eksekusi pod yang ditentukan untuk profil Fargate Anda dengan perintah berikut. Ganti *111122223333* dengan ID akun Anda. Ganti *AmazonEKSFargatePodExecutionRole* dengan peran Pod eksekusi Anda (untuk informasi selengkapnya, lihatBuat peran eksekusi Fargate Pod).

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/eks-fargate-logging-policy \
  --role-name AmazonEKSFargatePodExecutionRole
```

Kubernetesdukungan filter

Fitur ini membutuhkan Kubernetes versi minimum dan level platform berikut, atau yang lebih baru.

Versi Kubernetes	Tingkat platform
1.23 dan kemudian	eks.1

Fluent BitKubernetesFilter memungkinkan Anda untuk menambahkan Kubernetes metadata ke file log Anda. Untuk informasi selengkapnya tentang filter, lihat Kubernetesdi Fluent Bit dokumentasi. Anda dapat menerapkan filter menggunakan titik akhir server API.

```
filters.conf: |
    [FILTER]
        Name                kubernetes
        Match               kube.*
        Merge_Log           On
        Buffer_Size         0
        Kube_Meta_Cache_TTL 300s
```

> **Important**
>
> - `Kube_URL`,`Kube_CA_File`,`Kube_Token_Command`, dan `Kube_Token_File` merupakan parameter konfigurasi yang dimiliki layanan dan tidak boleh ditentukan. Amazon EKS Fargate mengisi nilai-nilai ini.
> - `Kube_Meta_Cache_TTL`adalah waktu Fluent Bit menunggu hingga berkomunikasi dengan server API untuk metadata terbaru. Jika `Kube_Meta_Cache_TTL` tidak ditentukan, Amazon EKS Fargate menambahkan nilai default 30 menit untuk mengurangi beban pada server API.

Untuk mengirimkan log Fluent Bit proses ke akun Anda

Anda dapat secara opsional mengirimkan log Fluent Bit proses ke Amazon CloudWatch menggunakan yang berikut `ConfigMap` ini. Pengiriman log proses Fluent Bit ke CloudWatch membutuhkan biaya konsumsi dan penyimpanan log tambahan. Ganti *`region-code`* dengan tempat Wilayah AWS cluster Anda berada.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
  labels:
data:
  # Configuration files: server, input, filters and output
  # ======================================================
```

```
  flb_log_cw: "true"  # Ships Fluent Bit process logs to CloudWatch.

  output.conf: |
    [OUTPUT]
        Name cloudwatch
        Match kube.*
        region region-code
        log_group_name fluent-bit-cloudwatch
        log_stream_prefix from-fluent-bit-
        auto_create_group true
```

Log berada di Wilayah AWS tempat cluster berada di bawah CloudWatch. Nama grup log adalah *my-cluster*-fluent-bit-logs dan nama Fluent Bit logstream adalahfluent-bit-*podname*-*pod-namespace*.

> **Note**
>
> - Log proses dikirim hanya ketika Fluent Bit proses berhasil dimulai. Jika ada kegagalan saat memulaiFluent Bit, log proses terlewatkan. Anda hanya dapat mengirimkan log proses ke CloudWatch.
> - Untuk men-debug log proses pengiriman ke akun Anda, Anda dapat menerapkan sebelumnya `ConfigMap` untuk mendapatkan log proses. Fluent Bitgagal untuk memulai biasanya karena Anda `ConfigMap` tidak diurai atau diterima Fluent Bit saat memulai.

Untuk menghentikan log Fluent Bit proses pengiriman

Log Fluent Bit proses pengiriman ke CloudWatch membutuhkan biaya konsumsi dan penyimpanan log tambahan. Untuk mengecualikan log proses dalam `ConfigMap` pengaturan yang ada, lakukan langkah-langkah berikut.

1. Temukan grup CloudWatch log yang dibuat secara otomatis untuk log Fluent Bit proses klaster Amazon EKS Anda setelah mengaktifkan pencatatan Fargate. Ini mengikuti formatnya`{cluster_name}-fluent-bit-logs`.
2. Hapus aliran CloudWatch log yang ada yang dibuat untuk setiap log Pod's proses di grup CloudWatch log.
3. Edit `ConfigMap` dan aturflb_log_cw: "false".
4. Mulai ulang semua yang ada Pods di cluster.

Aplikasi uji

1. Menyebarkan sampelPod.

   a. Simpan konten berikut ini ke file bernama *sample-app*`.yaml` pada komputer Anda.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: sample-app
  namespace: same-namespace-as-your-fargate-profile
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:latest
          ports:
            - name: http
              containerPort: 80
```

   b. Menerapkan manifes ke klaster.

```
kubectl apply -f sample-app.yaml
```

2. Lihat log NGINX menggunakan tujuan yang Anda konfigurasikan di file. `ConfigMap`

Pertimbangan ukuran

Kami menyarankan Anda merencanakan hingga 50 MB memori untuk router log. Jika Anda mengharapkan aplikasi Anda untuk menghasilkan catatan pada throughput yang sangat tinggi, maka Anda harus menyediakan memori hingga 100 MB.

Memecahkan masalah

Untuk mengonfirmasi apakah fitur logging diaktifkan atau dinonaktifkan karena alasan tertentu, seperti tidak validConfigMap, dan mengapa fitur tersebut tidak valid, periksa acara Anda dengan. Pod **`kubectl describe pod`** ***`pod_name`*** Output mungkin termasuk Pod peristiwa yang menjelaskan apakah logging diaktifkan atau tidak, seperti contoh output berikut.

```
[...]
Annotations:          CapacityProvisioned: 0.25vCPU 0.5GB
                      Logging: LoggingDisabled: LOGGING_CONFIGMAP_NOT_FOUND
                      kubernetes.io/psp: eks.privileged
[...]
Events:
  Type     Reason           Age        From
               Message
  ----     ------           ----       ----
               -------
  Warning  LoggingDisabled  <unknown>  fargate-scheduler
               Disabled logging because aws-logging configmap was not found. configmap
 "aws-logging" not found
```

PodPeristiwa bersifat fana dengan periode waktu tergantung pada pengaturannya. Anda juga dapat melihat Pod's anotasi menggunakan**`kubectl describe pod`** ***`pod-name`***. Dalam Pod anotasi, ada informasi tentang apakah fitur logging diaktifkan atau dinonaktifkan dan alasannya.

Memilih jenis instans Amazon EC2

Amazon EC2 menyediakan berbagai pilihan jenis instans untuk node pekerja. Setiap jenis instans menawarkan kemampuan komputasi, memori, penyimpanan, dan jaringan yang berbeda. Setiap instance juga dikelompokkan dalam keluarga instance berdasarkan kemampuan ini. Untuk daftar, lihat Jenis instans yang tersedia di Panduan Pengguna Amazon EC2 untuk Instans Linux dan jenis instans yang tersedia di Panduan Pengguna Amazon EC2 untuk Instans Windows. Amazon EKS merilis beberapa variasi AMI Amazon EC2 untuk mengaktifkan dukungan. Untuk memastikan bahwa jenis instans yang Anda pilih kompatibel dengan Amazon EKS, pertimbangkan kriteria berikut.

- Semua AMI Amazon EKS saat ini tidak mendukung `g5g` dan `mac` keluarga.
- Armdan AMI Amazon EKS yang tidak dipercepat tidak mendukung`g3`,`g4`,`inf`, dan `p` keluarga.
- AMI Amazon EKS yang dipercepat tidak mendukung `ac`,`hpc`,,`m`, dan `t` keluarga.

- Untuk instans berbasis ARM, Amazon Linux 2023 (AL2023) hanya mendukung jenis instans yang menggunakan prosesor atau yang lebih baru. Graviton2 AL2023 tidak mendukung `A1` instance.

Saat memilih di antara jenis instans yang didukung oleh Amazon EKS, pertimbangkan kemampuan masing-masing jenis berikut.

Jumlah instance dalam grup node

Secara umum, lebih sedikit, contoh yang lebih besar lebih baik, terutama jika Anda memiliki banyak. Daemonsets Setiap instance memerlukan panggilan API ke server API, jadi semakin banyak instance yang Anda miliki, semakin banyak beban di server API.

Sistem operasi

Tinjau jenis instans yang didukung untuk Linux, Windows, dan Bottlerocket. Sebelum membuat Windows instance, tinjauMengaktifkan Windows dukungan untuk klaster Amazon EKS Anda.

Arsitektur perangkat keras

Apakah Anda membutuhkan x86 atauArm? Anda hanya dapat menerapkan Linux padaArm. Sebelum menerapkan Arm instance, tinjau. Amazon EKS mengoptimalkan AMI Arm Amazon Linux Apakah Anda memerlukan instance yang dibangun di atas Nitro System (Linux atau Windows) atau yang memiliki kemampuan Dipercepat? Jika Anda membutuhkan kemampuan yang dipercepat, Anda hanya dapat menggunakannya Linux dengan Amazon EKS.

Jumlah maksimum Pods

Karena masing-masing Pod diberi alamat IP sendiri, jumlah alamat IP yang didukung oleh jenis instance merupakan faktor dalam menentukan jumlah Pods yang dapat dijalankan pada instance. Untuk menentukan secara manual berapa banyak jenis Pods instance yang mendukung, lihatAmazon EKS merekomendasikan maksimum Pods untuk setiap jenis instans Amazon EC2.

> **Note**
>
> Jika Anda menggunakan Amazon EKS yang dioptimalkan Amazon Linux 2 AMI `v20220406` atau yang lebih baru, Anda dapat menggunakan jenis instans baru tanpa memutakhirkan ke AMI terbaru. Untuk AMI ini, AMI secara otomatis menghitung `max-pods` nilai yang diperlukan jika tidak tercantum dalam file. `eni-max-pods.txt` Jenis instans yang saat ini dalam pratinjau mungkin tidak didukung oleh Amazon EKS secara

default. Nilai `max-pods` untuk tipe seperti itu masih perlu ditambahkan `eni-max-pods.txt` di AMI kami.

AWS Jenis instans Sistem Nitro secara opsional mendukung lebih banyak alamat IP daripada jenis instans Sistem non-Nitro. Namun, tidak semua alamat IP yang ditetapkan untuk sebuah instance tersedia untukPods. Untuk menetapkan jumlah alamat IP yang jauh lebih besar ke instans Anda, Anda harus memiliki versi `1.9.0` atau yang lebih baru dari add-on Amazon VPC CNI yang diinstal di cluster Anda dan dikonfigurasi dengan tepat. Untuk informasi selengkapnya, lihat Tingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda. Untuk menetapkan jumlah alamat IP terbesar ke instans Anda, Anda harus memiliki versi `1.10.1` atau yang lebih baru dari add-on Amazon VPC CNI yang diinstal di cluster Anda dan menyebarkan cluster dengan keluarga. `IPv6`

Keluarga IP

Anda dapat menggunakan jenis instans apa pun yang didukung saat menggunakan `IPv4` keluarga untuk klaster, yang memungkinkan klaster Anda menetapkan `IPv4` alamat pribadi ke Layanan Pods dan Anda. Tetapi jika Anda ingin menggunakan `IPv6` keluarga untuk cluster Anda, maka Anda harus menggunakan jenis instance AWS Nitro System atau tipe instance bare metal. Hanya `IPv4` didukung untuk Windows instance. Cluster Anda harus menjalankan versi `1.10.1` atau yang lebih baru dari add-on Amazon VPC CNI. Untuk informasi selengkapnya tentang penggunaan `IPv6`, lihat `IPv6`alamat untuk cluster,Pods, dan services.

Versi add-on Amazon VPC CNI yang Anda jalankan

Versi terbaru plugin Amazon VPC CNI untuk Kubernetes mendukung jenis instans ini. Anda mungkin perlu memperbarui versi add-on Amazon VPC CNI untuk memanfaatkan jenis instans terbaru yang didukung. Untuk informasi selengkapnya, lihat Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS. Versi terbaru mendukung fitur terbaru untuk digunakan dengan Amazon EKS. Versi sebelumnya tidak mendukung semua fitur. Anda dapat melihat fitur yang didukung oleh versi yang berbeda di Changelog aktif. GitHub

Wilayah AWS bahwa Anda membuat node Anda di

Tidak semua jenis instance tersedia di semua Wilayah AWS.

Apakah Anda menggunakan grup keamanan untuk Pods

Jika Anda menggunakan grup keamanan untukPods, hanya jenis instans tertentu yang didukung. Untuk informasi selengkapnya, lihat Kelompok keamanan untuk Pods.

Amazon EKS merekomendasikan maksimum Pods untuk setiap jenis instans Amazon EC2

Karena masing-masing Pod diberi alamat IP sendiri, jumlah alamat IP yang didukung oleh jenis instance merupakan faktor dalam menentukan jumlah Pods yang dapat dijalankan pada instance. Amazon EKS menyediakan skrip yang dapat Anda unduh dan jalankan untuk menentukan jumlah maksimum yang disarankan Amazon EKS Pods untuk dijalankan pada setiap jenis instans. Skrip menggunakan atribut perangkat keras dari setiap instance, dan opsi konfigurasi, untuk menentukan Pods jumlah maksimum. Anda dapat menggunakan nomor yang dikembalikan dalam langkah-langkah ini untuk mengaktifkan kemampuan seperti menetapkan alamat IP Pods dari subnet yang berbeda dari instans dan secara signifikan meningkatkan jumlah alamat IP untuk instans Anda. Jika Anda menggunakan grup node terkelola dengan beberapa tipe instance, gunakan nilai yang akan berfungsi untuk semua jenis instance.

1. Unduh skrip yang dapat Anda gunakan untuk menghitung jumlah maksimum Pods untuk setiap jenis instans.

   ```
   curl -O https://raw.githubusercontent.com/awslabs/amazon-eks-ami/master/files/max-pods-calculator.sh
   ```

2. Tandai skrip sebagai dapat dieksekusi di komputer Anda.

   ```
   chmod +x max-pods-calculator.sh
   ```

3. Jalankan skrip, ganti *m5.large*dengan jenis instans yang Anda rencanakan untuk diterapkan dan *1.9.0-eksbuild.1* dengan versi add-on Amazon VPC CNI Anda. Untuk menentukan versi add-on Anda, lihat prosedur pembaruan diBekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS.

   ```
   ./max-pods-calculator.sh --instance-type m5.large --cni-version 1.9.0-eksbuild.1
   ```

   Contoh output adalah sebagai berikut.

   ```
   29
   ```

   Anda dapat menambahkan opsi berikut ke skrip untuk melihat maksimum yang Pods didukung saat menggunakan kemampuan opsional.

- `--cni-custom-networking-enabled`— Gunakan opsi ini ketika Anda ingin menetapkan alamat IP dari subnet yang berbeda dari instans Anda. Untuk informasi selengkapnya, lihat Jaringan khusus untuk pod. Menambahkan opsi ini ke skrip sebelumnya dengan nilai contoh yang sama menghasilkan`20`.
- `--cni-prefix-delegation-enabled`— Gunakan opsi ini ketika Anda ingin menetapkan lebih banyak alamat IP secara signifikan ke setiap elastic network interface. Kemampuan ini memerlukan instans Amazon Linux yang berjalan pada Sistem Nitro dan versi `1.9.0` atau yang lebih baru dari add-on Amazon VPC CNI. Untuk informasi selengkapnya, lihat Tingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda. Menambahkan opsi ini ke skrip sebelumnya dengan nilai contoh yang sama menghasilkan`110`.

Anda juga dapat menjalankan skrip dengan `--help` opsi untuk melihat semua opsi yang tersedia.

> **Note**
>
> Skrip Pods kalkulator maks membatasi nilai pengembalian `110` berdasarkan ambang Kubernetes skalabilitas dan pengaturan yang disarankan. Jika jenis instans Anda memiliki lebih dari 30 vCPU, batas ini akan menjadi angka berdasarkan pengujian `250` tim skalabilitas Amazon EKS internal. Untuk informasi selengkapnya, lihat plugin Amazon VPC CNI meningkatkan pod per node membatasi posting blog.

AMI yang dioptimalkan Amazon EKS

Anda dapat men-deploy simpul dengan Amazon EKS yang sudah dibangun sebelumnya dan dioptimalkan oleh Amazon Machine Images (AMI) Anda sendiri. Untuk informasi tentang setiap jenis Amazon EKS yang dioptimalkan oleh AMI, lihat salah satu topik berikut. Untuk petunjuk tentang cara membuat AMI kustom Anda sendiri, lihatBuild script Amazon Linux AMI yang dioptimalkan oleh Amazon EKS.

Topik

- Amazon EKS mengakhiri dukungan untukDockershim
- Amazon Linux AMI yang dioptimalkan oleh Amazon EKS
- Amazon EKS mengoptimalkan Bottlerocket AMI
- Ubuntu Linux AMI yang dioptimalkan oleh Amazon EKS

- Amazon EKS dioptimalkan Windows AMI

Amazon EKS mengakhiri dukungan untuk`Dockershim`

Kubernetestidak lagi mendukung`Dockershim`. YangKubernetestim menghapus runtime diKubernetesversi`1.24`. Untuk informasi lebih lanjut, lihatKubernetesBergerak dari Dockershim: Komitmen dan Langkah BerikutnyapadaKubernetesBlog.

Amazon EKS juga mengakhiri dukungan untuk`Dockershim`dimulai denganKubernetesversi`1.24`rilis. Amazon EKS AMI yang secara resmi diterbitkan memiliki`containerd`sebagai satu-satunya runtime dimulai dengan versi`1.24`. Topik ini mencakup beberapa detail, tetapi informasi lebih lanjut tersedia diYang perlu Anda ketahui tentang pindah ke containerd di Amazon EKS.

Ada`kubectl`plugin yang dapat Anda gunakan untuk melihat mana dari AndaKubernetesbeban kerja me-mountDockerVolume soket. Untuk informasi lebih lanjut, lihatDetektor untukDockerSoket (DDS)di atasGitHub. AMI Amazon EKS yang berjalanKubernetesversi yang lebih awal dari`1.24`menggunakanDockersebagai runtime default. Namun, AMI Amazon EKS ini memiliki opsi bendera bootstrap yang dapat Anda gunakan untuk menguji beban kerja Anda pada klaster yang didukung menggunakan`containerd`. Untuk informasi selengkapnya, lihat Uji migrasi dari Docker ke `containerd`.

Kami akan terus mempublikasikan AMI untuk yang adaKubernetesversi sampai akhir tanggal dukungan mereka. Untuk informasi selengkapnya, lihat Kalender Kubernetes rilis Amazon EKS. Jika Anda membutuhkan lebih banyak waktu untuk menguji beban kerja`containerd`, gunakan versi yang didukung sebelumnya`1.24`. Tapi, saat Anda ingin meningkatkan AMI Amazon EKS resmi ke versi`1.24`atau yang lebih baru, pastikan untuk memvalidasi bahwa beban kerja Anda berjalan`containerd`.

Yang`containerd`runtime memberikan kinerja dan keamanan yang lebih andal.`containerd`adalah runtime yang sedang distandarisasi di seluruh Amazon EKS. Fargate danBottlerocketsudah menggunakan`containerd`hanya.`containerd`membantu meminimalkan jumlah rilis Amazon EKS AMI yang diperlukan untuk mengatasi`Dockershim` Kerentanan Umum dan Eksposur(CVE). Karena`Dockershim`sudah menggunakan`containerd`secara internal, Anda mungkin tidak perlu melakukan perubahan apa pun. Namun, ada beberapa situasi di mana perubahan mungkin atau harus diperlukan:

- Anda harus membuat perubahan pada aplikasi yang me-mountDockersoket. Misalnya, gambar kontainer yang dibuat dengan wadah akan terkena dampak. Banyak alat pemantauan juga me-

mountDockersoket. Anda mungkin perlu menunggu pembaruan atau menyebarkan kembali beban kerja untuk pemantauan waktu proses.

- Anda mungkin perlu membuat perubahan untuk aplikasi yang bergantung pada spesifikDockerpengaturan. Misalnya,`HTTPS_PROXY`protokol tidak lagi didukung. Anda harus memperbarui aplikasi yang menggunakan protokol ini. Untuk informasi lebih lanjut, lihat`dockerd`di dalamDockerDokumen.
- Jika Anda menggunakan Amazon ECR credential helper untuk menarik gambar, Anda harus beralih ke`kubelet`penyedia kredensi gambar. Untuk informasi lebih lanjut, lihatKonfigurasikan`kubelet`penyedia kredensi gambardi dalamKubernetesdokumentasi.
- Karena Amazon EKS`1.24`tidak lagi mendukungDocker, beberapa bendera yangSkrip bootstrap Amazon EKSsebelumnya didukung tidak lagi didukung. Sebelum pindah ke Amazon EKS`1.24`atau yang lebih baru, Anda harus menghapus referensi apa pun ke bendera yang sekarang tidak didukung:
  - `--container-runtime dockerd`(`containerd`adalah satu-satunya nilai yang didukung)
  - `--enable-docker-bridge`
  - `--docker-config-json`
- Jika Anda sudah memilikiFluentddikonfigurasi untukContainer Insights, maka Anda harus bermigrasiFluentdkepadaFluent Bitsebelum berganti`containerd`. YangFluentdparser dikonfigurasi untuk hanya mengurai pesan log dalam format JSON. Tidak seperti`dockerd`, yang`containerd`runtime kontainer memiliki pesan log yang tidak dalam format JSON. Jika Anda tidak bermigrasi keFluent Bit, beberapa dikonfigurasiFluentd'sparser akan menghasilkan sejumlah besar kesalahan di dalamFluentdwadah. Untuk informasi lebih lanjut tentang migrasi, lihatMengaturFluent BitsebagaiDaemonSetuntuk mengirim log keCloudWatchLog.
- Jika Anda menggunakan AMI kustom dan Anda meningkatkan ke Amazon EKS`1.24`, maka Anda harus memastikan bahwa penerusan IP diaktifkan untuk node pekerja Anda. Pengaturan ini tidak diperlukan denganDockertetapi diperlukan untuk`containerd`. Diperlukan untuk memecahkan masalahPod-untuk-Pod,Pod-ke-eksternal, atauPod-untuk-apiserverkonektivitas jaringan.

  Untuk memverifikasi pengaturan ini pada node pekerja, jalankan salah satu perintah berikut:

  - `sysctl net.ipv4.ip_forward`
  - `cat /proc/sys/net/ipv4/ip_forward`

  Jika outputnya`0`, kemudian jalankan salah satu perintah berikut untuk mengaktifkan`net.ipv4.ip_forward`variabel kernel:

  - `sysctl -w net.ipv4.ip_forward=1`

- `echo 1 > /proc/sys/net/ipv4/ip_forward`

Untuk aktivasi pengaturan di Amazon EKS AMI di`containerd`runtime, lihat`install-worker.sh`di atasGitHub.

Amazon Linux AMI yang dioptimalkan oleh Amazon EKS

Amazon EKS yang dioptimalkan Amazon Linux AMI dibangun di atas Amazon Linux 2 (AL2) dan Amazon Linux 2023 (AL2023). Ini dikonfigurasi untuk berfungsi sebagai gambar dasar untuk node Amazon EKS. AMI dikonfigurasi untuk bekerja dengan Amazon EKS dan mencakup komponen-komponen berikut:

- `kubelet`
- AWS Autentikator IAM
- Docker(Versi Amazon EKS `1.23` dan sebelumnya)
- `containerd`

> **Note**
>
> - Anda dapat melacak peristiwa keamanan atau privasi untuk AL2 di pusat keamanan Amazon Linux atau berlangganan umpan RSS terkait. Kejadian keamanan dan privasi mencakup gambaran umum mengenai masalah, paket apa yang terpengaruh, dan cara memperbarui instans Anda untuk memperbaiki masalah tersebut.
> - Sebelum menerapkan akselerasi atau Arm AMI, tinjau informasi di Amazon Amazon Linux AMI yang dioptimalkan oleh Amazon EKS danAmazon EKS mengoptimalkan AMI Arm Amazon Linux.
> - Untuk Kubernetes versi`1.23`, Anda dapat menggunakan bendera bootstrap opsional untuk menguji migrasi dari Docker ke`containerd`. Untuk informasi selengkapnya, lihat Uji migrasi dari Docker ke `containerd`.
> - Dimulai dengan Kubernetes versi`1.25`, Anda tidak akan lagi dapat menggunakan P2 instans Amazon EC2 dengan Amazon EKS yang dioptimalkan Amazon Linux AMI yang dipercepat di luar kotak. AMI ini untuk Kubernetes versi `1.25` atau yang lebih baru akan mendukung driver `NVIDIA 525` seri atau yang lebih baru, yang tidak kompatibel dengan P2 instance. Namun, driver `NVIDIA 525` seri atau yang lebih baru kompatibel

denganP3,P4, dan P5 instance, sehingga Anda dapat menggunakan instance tersebut dengan AMI untuk Kubernetes versi 1.25 atau yang lebih baru. Sebelum kluster Amazon EKS Anda ditingkatkan ke versi1.25, migrasikan P2 instans apa pun ke,P3, P4 dan instans. P5 Anda juga harus secara proaktif meningkatkan aplikasi Anda untuk bekerja dengan NVIDIA 525 seri atau yang lebih baru. Kami berencana untuk mendukung port NVIDIA 525 seri yang lebih baru atau driver yang lebih baru ke Kubernetes versi 1.23 dan 1.24 pada akhir Januari 2024.

- Dimulai dengan versi Amazon EKS 1.30 atau yang lebih baru, grup node terkelola yang baru dibuat akan secara otomatis menggunakan AL2023 sebagai sistem operasi node. Sebelumnya, grup node baru akan default ke AL2. Anda dapat terus menggunakan AL2 dengan memilihnya sebagai tipe AMI saat membuat grup node baru.
- Support untuk AL2 akan berakhir pada 30 Juni 2025. Untuk informasi selengkapnya, lihat FAQ Amazon Linux 2.

Tingkatkan dari AL2 ke AL2023

AMI Amazon EKS yang dioptimalkan tersedia dalam dua keluarga berdasarkan AL2 dan AL2023. AL2023 adalah sistem operasi berbasis Linux baru yang dirancang untuk menyediakan lingkungan yang aman, stabil, dan berkinerja tinggi untuk aplikasi cloud Anda. Ini adalah generasi berikutnya dari Amazon Linux dari Amazon Web Services dan tersedia di semua versi Amazon EKS yang didukung, termasuk versi 1.23 dan 1.24 dukungan yang diperluas. AMI akselerasi Amazon EKS berdasarkan AL2023 akan tersedia di kemudian hari. Jika Anda memiliki beban kerja yang dipercepat, Anda harus terus menggunakan AMI atau Bottlerocket yang dipercepat AL2.

AL2023 menawarkan beberapa peningkatan dibandingkan AL2. Untuk perbandingan selengkapnya, lihat Membandingkan AL2 dan Amazon Linux 2023 di Panduan Pengguna Amazon Linux 2023. Beberapa paket telah ditambahkan, ditingkatkan, dan dihapus dari AL2. Sangat disarankan untuk menguji aplikasi Anda dengan AL2023 sebelum memutakhirkan. Untuk daftar semua perubahan paket di AL2023, lihat Perubahan paket di Amazon Linux 2023 di Catatan Rilis Amazon Linux 2023.

Selain perubahan ini, Anda harus mengetahui hal-hal berikut:

- AL2023 memperkenalkan proses inisialisasi node baru nodeadm yang menggunakan skema konfigurasi YAMAL. Jika Anda menggunakan grup node yang dikelola sendiri atau AMI dengan template peluncuran, Anda sekarang harus menyediakan metadata klaster tambahan secara eksplisit saat membuat grup node baru. Contoh parameter minimum yang diperlukan adalah

sebagai berikut, di manaapiServerEndpoint,certificateAuthority, dan layanan sekarang cidr diperlukan:

```
---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: my-cluster
    apiServerEndpoint: https://example.com
    certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
    cidr: 10.100.0.0/16
```

Di AL2, metadata dari parameter ini ditemukan dari panggilan Amazon EKS DescribeCluster API. Dengan AL2023, perilaku ini telah berubah karena panggilan API tambahan berisiko melambat selama peningkatan skala node besar. Perubahan ini tidak memengaruhi Anda jika Anda menggunakan grup node terkelola tanpa templat peluncuran atau jika Anda menggunakanKarpenter. Untuk informasi selengkapnya tentang certificateAuthority dan layanancidr, lihat DescribeCluster di Referensi API Amazon EKS.

- Dockertidak didukung di AL2023 untuk semua versi Amazon EKS yang didukung. Support for Docker telah berakhir dan telah dihapus dengan versi Amazon EKS 1.24 atau lebih tinggi di AL2. Untuk informasi selengkapnya tentang penghentian, lihat Amazon EKS mengakhiri dukungan untuk. Dockershim
- Versi Amazon VPC CNI 1.16.2 atau yang lebih besar diperlukan untuk AL2023.
- AL2023 membutuhkan secara IMDSv2 default. IMDSv2memiliki beberapa manfaat yang membantu meningkatkan postur keamanan. Ini menggunakan metode otentikasi berorientasi sesi yang memerlukan pembuatan token rahasia dalam permintaan HTTP PUT sederhana untuk memulai sesi. Token sesi dapat berlaku di mana saja antara 1 detik dan 6 jam. Untuk informasi selengkapnya tentang cara transisi dari IMDSv1 keIMDSv2, lihat Transisi ke menggunakan Layanan Metadata Instans Versi 2 dan Dapatkan manfaat penuh dari IMDSv2 dan nonaktifkan IMDSv1 di seluruh infrastruktur Anda. AWS Jika Anda ingin menggunakannyaIMDSv1, Anda masih dapat melakukannya dengan mengganti pengaturan secara manual menggunakan properti peluncuran opsi metadata instance.

> **Note**
>
> UntukIMDSv2, jumlah hop default untuk grup node terkelola diatur ke 1. Ini berarti bahwa kontainer tidak akan memiliki akses ke kredenal node menggunakan IMDS.

Jika Anda memerlukan akses kontainer ke kredensional node, Anda masih dapat melakukannya dengan mengganti templat peluncuran Amazon EC2 kustom secara manual, meningkatkannya menjadi 2. `HttpPutResponseHopLimit` Atau, Anda dapat menggunakan Amazon EKS Pod Identity untuk memberikan kredensi alih-alih. `IMDSv2`

- AL2023 menampilkan generasi berikutnya dari hierarki grup kontrol terpadu (). `cgroupv2` `cgroupv2`digunakan untuk mengimplementasikan runtime kontainer, dan oleh`systemd`. Meskipun AL2023 masih menyertakan kode yang dapat membuat sistem berjalan menggunakan`cgroupv1`, ini bukan konfigurasi yang direkomendasikan atau didukung. Konfigurasi ini akan sepenuhnya dihapus dalam rilis besar masa depan Amazon Linux.

Untuk grup node terkelola yang sudah ada sebelumnya, Anda dapat melakukan peningkatan di tempat atau peningkatan biru/hijau tergantung pada cara Anda menggunakan templat peluncuran:

- Jika Anda menggunakan AMI kustom dengan grup node terkelola, Anda dapat melakukan pemutakhiran di tempat dengan menukar ID AMI di template peluncuran. Anda harus memastikan bahwa aplikasi Anda dan data pengguna apa pun mentransfer ke AL2023 terlebih dahulu sebelum melakukan strategi peningkatan ini.
- Jika Anda menggunakan grup node terkelola dengan templat peluncuran standar atau dengan templat peluncuran khusus yang tidak menentukan ID AMI, Anda harus meningkatkan menggunakan strategi biru/hijau. Upgrade biru/hijau biasanya lebih kompleks dan melibatkan pembuatan grup node yang sama sekali baru di mana Anda akan menentukan AL2023 sebagai tipe AMI. Grup node baru perlu dikonfigurasi dengan hati-hati untuk memastikan bahwa semua data kustom dari grup node AL2 kompatibel dengan OS baru. Setelah grup node baru telah diuji dan divalidasi dengan aplikasi Anda, Pods dapat dimigrasikan dari grup node lama ke grup node baru. Setelah migrasi selesai, Anda dapat menghapus grup node lama.

Jika Anda menggunakan Karpenter dan ingin menggunakan AL2023, Anda harus memodifikasi `EC2NodeClass` `amiFamily` bidang dengan AL2023. Secara default, Drift diaktifkan diKarpenter. Ini berarti bahwa setelah `amiFamily` bidang telah diubah, secara otomatis Karpenter akan memperbarui node pekerja Anda ke AMI terbaru bila tersedia.

Amazon Amazon Linux AMI yang dioptimalkan oleh Amazon EKS

> **Note**
>
> AMI akselerasi Amazon EKS berdasarkan AL2023 akan tersedia di kemudian hari. Jika Anda memiliki beban kerja yang dipercepat, Anda harus terus menggunakan AMI yang dipercepat AL2 atau. Bottlerocket

Amazon EKS yang dioptimalkan Amazon Linux AMI yang dipercepat dibangun di atas standar Amazon EKS yang dioptimalkan Amazon Linux AMI. Ini dikonfigurasi untuk berfungsi sebagai gambar opsional untuk node Amazon EKS untuk mendukung beban kerja berbasis GPU, Inferentia, dan Trainium.

Selain konfigurasi AMI standar yang dioptimalkan oleh Amazon EKS, AMI yang terakselerasi mencakup hal-hal berikut:

- NVIDIAdriver
- `nvidia-container-runtime` (sebagai waktu aktif default)
- AWS Neuronruntime kontainer

Untuk daftar komponen terbaru yang disertakan dalam AMI yang dipercepat, lihat `amazon-eks-ami` Rilis diGitHub.

> **Note**
>
> - AMI akselerasi Amazon EKS yang dioptimalkan hanya mendukung GPU dan tipe instans Inferentia berbasis. Pastikan untuk menentukan jenis instance ini di AWS CloudFormation template node Anda. Dengan menggunakan AMI terakselerasi yang dioptimalkan oleh Amazon EKS, Anda setuju dengan NVIDIA's user license agreement (EULA).
> - AMI terakselerasi yang dioptimalkan oleh Amazon EKS sebelumnya disebut sebagai AMI yang dioptimalkan oleh Amazon EKS dengan dukungan GPU.
> - Versi sebelumnya dari Amazon EKS yang dioptimalkan AMI yang dipercepat menginstal `nvidia-docker` repositori. Repositori tidak lagi disertakan dalam Amazon EKS AMI versi `v20200529` dan yang lebih baru.

Untuk mengaktifkan beban kerja berbasis GPU

Prosedur berikut menjelaskan cara menjalankan beban kerja pada instans berbasis GPU dengan AMI terakselerasi yang dioptimalkan oleh Amazon EKS. Untuk opsi lain, lihat referensi berikut:

- Untuk informasi selengkapnya tentang menggunakan beban kerja Inferentia berbasis, lihatInferensi machine learning menggunakan AWS Inferentia.
- Untuk informasi lebih lanjut tentang penggunaanNeuron, lihat Container - Kubernetes - Memulai di Dokumentasi Neuron.AWS

1. Setelah node GPU Anda bergabung dengan cluster Anda, Anda harus menerapkan plugin perangkat NVIDIA Kubernetes sebagai a DaemonSet di cluster Anda. Ganti *vX.X.X* dengan s-device-plugin versi NVIDIA/K8 yang Anda inginkan sebelum menjalankan perintah berikut.

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

2. Anda dapat memverifikasi bahwa node Anda memiliki GPU yang dapat dialokasikan dengan perintah berikut.

   ```
   kubectl get nodes "-o=custom-
   columns=NAME:.metadata.name,GPU:.status.allocatable.nvidia\.com/gpu"
   ```

Untuk menerapkan Pod untuk menguji apakah node GPU Anda dikonfigurasi dengan benar

1. Buat file bernama `nvidia-smi.yaml` dengan isi berikut ini. Ganti *tag* dengan tag yang Anda inginkan untuk `nvidia/cuda`. Manifes ini meluncurkan sebuah NVIDIA CUDAwadah yang berjalan `nvidia-smi` pada sebuah node.

   ```
   apiVersion: v1
   kind: Pod
   metadata:
     name: nvidia-smi
   spec:
     restartPolicy: OnFailure
     containers:
     - name: nvidia-smi
       image: nvidia/cuda:tag
       args:
   ```

```
    - "nvidia-smi"
    resources:
      limits:
        nvidia.com/gpu: 1
```

2. Terapkan manifes dengan perintah berikut.

```
kubectl apply -f nvidia-smi.yaml
```

3. Setelah Pod selesai berjalan, lihat lognya dengan perintah berikut.

```
kubectl logs nvidia-smi
```

Contoh output adalah sebagai berikut.

```
Mon Aug  6 20:23:31 20XX
+-----------------------------------------------------------------------------+
| NVIDIA-SMI XXX.XX                 Driver Version: XXX.XX                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  On   | 00000000:00:1C.0 Off |                    0 |
| N/A   46C    P0    47W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

Amazon EKS mengoptimalkan AMI Arm Amazon Linux

Instans Arm memberikan penghematan biaya yang signifikan untuk aplikasi scale-out dan Arm berbasis seperti server web, layanan mikro kontainer, armada caching, dan penyimpanan data terdistribusi. Saat menambahkan Arm node ke cluster Anda, tinjau pertimbangan berikut.

Pertimbangan

- Jika klaster Anda diterapkan sebelum 17 Agustus 2020, Anda harus melakukan peningkatan satu kali dari manifes add-on cluster kritis. Ini agar Kubernetes dapat menarik gambar yang benar untuk setiap arsitektur perangkat keras yang digunakan di cluster Anda. Untuk informasi selengkapnya terkait cara memperbarui klaster add-on, lihat Perbarui Kubernetes versi untuk kluster Amazon EKS Anda. Jika Anda menerapkan cluster Anda pada atau setelah 17 Agustus 2020, maka, CoreDNS`kube-proxy`, dan Amazon VPC CNI plugin for Kubernetes add-on Anda sudah memiliki kemampuan multi-arsitektur.
- Aplikasi yang digunakan ke Arm node harus dikompilasi untuk Arm.
- Jika Anda memiliki DaemonSets yang disebarkan di cluster yang ada, atau Anda ingin menerapkannya ke cluster baru yang juga ingin Anda gunakan Arm node, maka verifikasi bahwa Anda DaemonSet dapat berjalan di semua arsitektur perangkat keras di cluster Anda.
- Anda dapat menjalankan grup Arm node dan grup node x86 di cluster yang sama. Jika ya, pertimbangkan untuk menerapkan gambar kontainer multi-arsitektur ke repositori kontainer seperti Amazon Elastic Container Registry dan kemudian menambahkan pemilih node ke manifes Anda sehingga Kubernetes mengetahui arsitektur perangkat keras apa yang dapat digunakan. Pod Untuk informasi selengkapnya, lihat Mendorong gambar multi-arsitektur di Panduan Pengguna Amazon ECR dan Memperkenalkan gambar wadah multi-arsitektur untuk posting blog Amazon ECR.

Uji migrasi dari Docker ke **`containerd`**

Amazon EKS mengakhiri dukungan untuk Docker memulai dengan `1.24` peluncuran Kubernetes versi. Untuk informasi selengkapnya, lihat Amazon EKS mengakhiri dukungan untuk`Dockershim`.

Untuk Kubernetes versi`1.23`, Anda dapat menggunakan flag bootstrap opsional untuk mengaktifkan `containerd` runtime untuk AMI AL2 Amazon EKS yang dioptimalkan. Fitur ini memberi Anda jalur yang jelas untuk bermigrasi `containerd` saat memperbarui ke versi `1.24` atau yang lebih baru. Amazon EKS mengakhiri dukungan untuk Docker memulai dengan `1.24` peluncuran Kubernetes versi. `containerd`Runtime diadopsi secara luas di Kubernetes masyarakat dan merupakan proyek lulus dengan CNCF. Anda dapat mengujinya dengan menambahkan grup node ke cluster baru atau yang sudah ada.

Anda dapat mengaktifkan flag boostrap dengan membuat salah satu jenis grup node berikut.

swakelola

Buat grup node menggunakan instruksi diMeluncurkan simpul Amazon Linux yang dikelola sendiri. Tentukan AMI Amazon EKS yang dioptimalkan dan teks berikut untuk `BootstrapArguments` parameter.

```
--container-runtime containerd
```

Dikelola

Jika Anda menggunakan`eksctl`, buat file bernama *`my-nodegroup`*`.yaml` dengan konten berikut. Ganti setiap *`example value`* dengan nilai-nilai Anda sendiri. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa. Untuk mengambil ID AMI yang dioptimalkan`ami-`*`1234567890abcdef0`*, lihatMendapatkan ID Amazon Linux AMI yang dioptimalkan Amazon EKS.

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code
  version: 1.23
managedNodeGroups:
  - name: my-nodegroup
    ami: ami-1234567890abcdef0
    overrideBootstrapCommand: |
      #!/bin/bash
      /etc/eks/bootstrap.sh my-cluster --container-runtime containerd
```

> **Note**
>
> Jika Anda meluncurkan banyak node secara bersamaan, Anda mungkin juga ingin menentukan nilai untuk argumen `--apiserver-endpoint--b64-cluster-ca`,, dan `--dns-cluster-ip` bootstrap untuk menghindari kesalahan. Untuk informasi selengkapnya, lihat Menentukan AMI.

Jalankan perintah berikut untuk membuat grup node.

```
eksctl create nodegroup -f my-nodegroup.yaml
```

Jika Anda lebih suka menggunakan alat yang berbeda untuk membuat grup node terkelola, Anda harus menerapkan grup node menggunakan template peluncuran. Di template peluncuran Anda, tentukan ID AMI Amazon EKS yang dioptimalkan, lalu terapkan grup node menggunakan template peluncuran dan berikan data pengguna berikut. Data pengguna ini meneruskan argumen ke dalam `bootstrap.sh` file. Untuk informasi lebih lanjut tentang file bootstrap, lihat bootstrap.sh diGitHub.

```
/etc/eks/bootstrap.sh my-cluster --container-runtime containerd
```

Informasi lain

Untuk informasi selengkapnya tentang penggunaan AMI Amazon Linux Amazon yang dioptimalkan Amazon EKS, lihat bagian berikut:

- Untuk menggunakan Amazon Linux dengan grup node terkelola, lihatGrup simpul terkelola.
- Untuk meluncurkan node Amazon Linux yang dikelola sendiri, lihatMendapatkan ID Amazon Linux AMI yang dioptimalkan Amazon EKS.
- Untuk informasi versi, lihat Versi Amazon Linux AMI yang dioptimalkan oleh Amazon EKS.
- Untuk mengambil ID terbaru dari Amazon EKS Amazon Linux AMI yang dioptimalkan, lihatMendapatkan ID Amazon Linux AMI yang dioptimalkan Amazon EKS.
- Untuk skrip sumber terbuka yang digunakan untuk membangun AMI Amazon EKS yang dioptimalkan, lihat. Build script Amazon Linux AMI yang dioptimalkan oleh Amazon EKS

Versi Amazon Linux AMI yang dioptimalkan oleh Amazon EKS

Amazon EKS dioptimalkan Amazon Linux AMI Kubernetes diversi berdasarkan versi dan tanggal rilis AMI dalam format berikut:

```
k8s_major_version.k8s_minor_version.k8s_patch_version-release_date
```

Setiap rilis AMI mencakup berbagai versi `kubelet`Docker,, Linux kernel, dan`containerd`. AMI yang dipercepat juga mencakup berbagai versi NVIDIA driver. Anda dapat menemukan informasi versi ini di Changelog di. GitHub

Mendapatkan ID Amazon Linux AMI yang dioptimalkan Amazon EKS

Anda dapat mengambil ID Amazon Machine Image (AMI) secara terprogram untuk AMI yang dioptimalkan Amazon EKS dengan menanyakan API Parameter Store. AWS Systems Manager Parameter ini mengeliminasi kebutuhan Anda untuk mencari ID AMI yang dioptimalkan oleh Amazon EKS secara manual. Untuk informasi selengkapnya tentang Systems Manager Parameter Store API, lihat GetParameter.

Untuk mengambil ID AMI untuk AMI yang dioptimalkan Amazon EKS menggunakan AWS CLI

1. Tentukan wilayah instance node Anda akan digunakan, sepertius-east-1.
2. Tentukan jenis AMI yang Anda butuhkan. Untuk informasi tentang jenis instans Amazon EC2, lihat Jenis Instans.
   - amazon-linux-2adalah untuk instance x86 berbasis Amazon Linux 2 (AL2).
   - amazon-linux-2-arm64adalah untuk instance AL2 ARM, seperti instance berbasis AWS Graviton.
   - amazon-linux-2-gpuadalah untuk instans akselerasi GPU AL2.
   - amazon-linux-2023/x86_64/standardadalah untuk instance berbasis Amazon Linux 2023 (AL2023)x86.
   - amazon-linux-2023/arm64/standardadalah untuk instans AL2023 ARM.
3. Tentukan Kubernetes versi cluster yang akan dilampirkan node Anda, seperti 1.29.
4. Jalankan AWS CLI perintah berikut untuk mengambil ID AMI yang sesuai. Ganti Wilayah AWS, Kubernetes versi, dan platform yang sesuai. Anda harus masuk ke AWS CLI menggunakan prinsipal IAM yang memiliki izin IAM untuk mengambil ssm:GetParameter metadata AMI Amazon EKS yang dioptimalkan.

```
aws ssm get-parameter --name /aws/service/eks/optimized-ami/1.29/amazon-linux-2/
recommended/image_id \
                --region region-code --query "Parameter.Value" --output text
```

Contoh output adalah sebagai berikut.

```
ami-1234567890abcdef0
```

Build script Amazon Linux AMI yang dioptimalkan oleh Amazon EKS

Amazon Elastic Kubernetes Service (Amazon EKS) memiliki skrip open-source yang digunakan untuk membangun AMI Amazon EKS yang dioptimalkan. Skrip build ini tersedia di GitHub.

Amazon EKS yang dioptimalkan Amazon Linux AMI dibangun di atas Amazon Linux 2 (AL2) dan Amazon Linux 2023 (AL2023), khusus untuk digunakan sebagai simpul di cluster Amazon EKS. Anda dapat menggunakan repositori ini untuk melihat secara spesifik bagaimana tim Amazon EKS mengonfigurasi`kubelet`, Autentikator AWS IAM untukDocker, Kubernetes dan membangun AMI berbasis Amazon Linux Anda sendiri dari awal.

Repositori skrip build menyertakan template HashiCorppacker dan skrip build untuk menghasilkan AMI. Skrip ini adalah sumber kebenaran untuk build AMI yang dioptimalkan Amazon EKS, sehingga Anda dapat mengikuti GitHub repositori untuk memantau perubahan pada AMI kami. Misalnya, mungkin Anda ingin AMI Anda sendiri menggunakan versi yang sama dengan Docker yang digunakan tim Amazon EKS untuk AMI resmi.

GitHubRepositori juga berisi skrip bootstrap khusus dan skrip nodeadm yang berjalan pada saat boot untuk mengonfigurasi data sertifikat instance Anda, titik akhir bidang kontrol, nama cluster, dan banyak lagi.

Selain itu, GitHub repositori berisi template node AWS CloudFormation Amazon EKS kami. Templat ini memudahkan dalam menjalankan instans yang menjalankan AMI yang dioptimalkan oleh Amazon EKS dan mendaftarkannya dengan sebuah klaster.

Untuk informasi lebih lanjut, lihat repositori GitHub di https://github.com/awslabs/. amazon-eks-ami

Amazon EKS dioptimalkan AL2 berisi flag bootstrap opsional untuk mengaktifkan `containerd` runtime.

Mengonfigurasi VT1 untuk AMI Amazon Linux kustom Anda

AMI Amazon Linux khusus di Amazon EKS dapat mendukung keluarga instans transcoding video VT1 untuk Amazon Linux 2 (AL2), Ubuntu 18, dan 20. Ubuntu VT1 mendukung kartu transcoding media Xilinx U30 dengan codec H.264/AVC dan H.265/HEVC yang dipercepat. Untuk mendapatkan manfaat dari instans yang dipercepat ini, Anda harus mengikuti langkah-langkah ini:

1. Buat dan luncurkan AMI dasar dari AL2, Ubuntu 18, atau Ubuntu 20.
2. Setelah AMI berbasis diluncurkan, Instal driver XRT dan runtime pada node.

3. Membuat klaster Amazon EKS.
4. Instal plugin Kubernetes FPGA di cluster Anda.

```
kubectl apply -f fpga-device-plugin.yml
```

Plugin sekarang akan mengiklankan perangkat Xilinx U30 per node di cluster Amazon EKS Anda. Anda dapat menggunakan image docker FFMPEG untuk menjalankan contoh beban kerja transcoding video di cluster Amazon EKS Anda.

Mengkonfigurasi DL1 untuk Amazon Linux 2 AMI kustom Anda

AMI Amazon Linux 2 (AL2) khusus di Amazon EKS dapat mendukung beban kerja pembelajaran mendalam dalam skala besar melalui konfigurasi dan Kubernetes add-on tambahan. Dokumen ini menjelaskan komponen yang diperlukan untuk menyiapkan Kubernetes solusi generik untuk penyiapan di lokasi atau sebagai dasar dalam konfigurasi cloud yang lebih besar. Untuk mendukung fungsi ini, Anda harus melakukan langkah-langkah berikut di lingkungan kustom Anda:

- Driver perangkat lunak SynaPaseAI® dimuat pada sistem — Ini termasuk dalam AMI yang tersedia di Github.

  Plugin perangkat Habana - Daemonset yang memungkinkan Anda mengaktifkan pendaftaran perangkat Habana secara otomatis di Kubernetes cluster Anda dan melacak kesehatan perangkat.
- Helm 3.x
- Bagan helm untuk menginstal Operator MPI.
- Operator MPI

1. Buat dan luncurkan AMI dasar dari AL2, Ubuntu 18, atau Ubuntu 20.
2. Ikuti petunjuk ini untuk mengatur lingkungan untuk DL1.

Amazon EKS mengoptimalkan Bottlerocket AMI

Bottlerocketadalah Linux distribusi open source yang disponsori dan didukung oleh. AWS Bottlerocketdibuat khusus untuk menampung beban kerja kontainer. DenganBottlerocket, Anda dapat meningkatkan ketersediaan penerapan kontainer dan mengurangi biaya operasional dengan mengotomatiskan pembaruan pada infrastruktur kontainer Anda. Bottlerockethanya mencakup perangkat lunak penting untuk menjalankan kontainer, yang meningkatkan penggunaan sumber

daya, mengurangi ancaman keamanan, dan menurunkan overhead manajemen. BottlerocketAMI termasuk`containerd`,`kubelet`, dan AWS IAM Authenticator. Selain grup node terkelola dan node yang dikelola sendiri, Bottlerocket juga didukung oleh Karpenter.

Keuntungan

Menggunakan Bottlerocket dengan cluster Amazon EKS Anda memiliki keuntungan sebagai berikut:

- Uptime yang lebih tinggi dengan biaya operasional yang lebih rendah dan kompleksitas manajemen yang lebih rendah — Bottlerocket memiliki jejak sumber daya yang lebih kecil, waktu boot yang lebih pendek, dan kurang rentan terhadap ancaman keamanan daripada distribusi lainnyaLinux. Bottlerocket'sfootprint yang lebih kecil membantu mengurangi biaya dengan menggunakan lebih sedikit penyimpanan, komputasi, dan sumber daya jaringan.
- Peningkatan keamanan dari pembaruan OS otomatis - Pembaruan Bottlerocket diterapkan sebagai satu unit yang dapat diputar kembali, jika perlu. Ini menghilangkan risiko pembaruan rusak atau gagal yang dapat membuat sistem dalam keadaan tidak dapat digunakan. DenganBottlerocket, pembaruan keamanan dapat diterapkan secara otomatis segera setelah tersedia dengan cara yang minimal mengganggu dan dibatalkan jika terjadi kegagalan.
- Dukungan premium - AWS disediakan build Bottlerocket di Amazon EC2 tercakup dalam paket AWS Support yang sama yang juga AWS mencakup layanan seperti Amazon EC2, Amazon EKS, dan Amazon ECR.

Pertimbangan-pertimbangan

Pertimbangkan hal berikut saat menggunakan Bottlerocket untuk tipe AMI Anda:

- Bottlerocketmendukung instans Amazon EC2 dengan `x86_64` dan prosesor. `arm64` BottlerocketAMI tidak direkomendasikan untuk digunakan dengan instans Amazon EC2 dengan chip Inferentia.
- Saat ini, tidak ada AWS CloudFormation template yang dapat Anda gunakan untuk menyebarkan Bottlerocket node.
- Bottlerocketgambar tidak menyertakan server SSH atau shell. Anda dapat menggunakan metode out-of-band akses untuk memungkinkan SSH. Pendekatan ini memungkinkan wadah admin dan meneruskan beberapa langkah konfigurasi bootstrap dengan data pengguna. Untuk informasi selengkapnya, lihat bagian berikut di BottlerocketOS tentangGitHub:
  - Eksplorasi
  - Kontainer admin

  - Kubernetespengaturan
- Bottlerocketmenggunakan berbagai jenis wadah:
  - Secara default, wadah kontrol diaktifkan. Container ini menjalankan AWS Systems Manageragen yang dapat Anda gunakan untuk menjalankan perintah atau memulai sesi shell di instans Amazon EC2Bottlerocket. Untuk informasi selengkapnya, lihat Menyiapkan Pengelola Sesi di Panduan AWS Systems Manager Pengguna.
  - Jika kunci SSH diberikan saat membuat grup node, wadah admin diaktifkan. Sebaiknya gunakan wadah admin hanya untuk skenario pengembangan dan pengujian. Kami tidak menyarankan menggunakannya untuk lingkungan produksi. Untuk informasi selengkapnya, lihat Kontainer admin diGitHub.

Informasi lain

Untuk informasi selengkapnya tentang penggunaan Bottlerocket AMI yang dioptimalkan Amazon EKS, lihat bagian berikut:

- Untuk detailnyaBottlerocket, lihat dokumentasi dan rilis diGitHub.
- Untuk menggunakan Bottlerocket dengan grup node terkelola, lihatGrup simpul terkelola.
- Untuk meluncurkan Bottlerocket node yang dikelola sendiri, lihatMeluncurkan node yang dikelola sendiri Bottlerocket.
- Untuk mengambil ID terbaru dari Bottlerocket AMI yang dioptimalkan Amazon EKS, lihatMengambil ID Bottlerocket AMI Amazon EKS yang dioptimalkan.
- Untuk detail tentang dukungan kepatuhan, lihatBottlerocketDukungan kepatuhan.

Mengambil ID Bottlerocket AMI Amazon EKS yang dioptimalkan

Anda dapat mengambil ID Amazon Machine Image (AMI) untuk AMI yang dioptimalkan Amazon EKS dengan menanyakan API AWS Systems Manager Parameter Store. Dengan menggunakan parameter ini, Anda tidak perlu mencari ID AMI yang dioptimalkan Amazon EKS secara manual. Untuk informasi selengkapnya tentang Systems Manager Parameter Store API, lihat GetParameter. Prinsip IAM yang Anda gunakan harus memiliki izin `ssm:GetParameter` IAM untuk mengambil metadata AMI Amazon EKS yang dioptimalkan.

Anda dapat mengambil ID gambar dari Bottlerocket AMI Amazon EKS terbaru yang dioptimalkan dengan AWS CLI perintah berikut dengan menggunakan `image_id` sub-parameter. Ganti *`1.29`*

dengan versi yang didukung dan *region-code* dengan Wilayah yang didukung Amazon EKS yang Anda inginkan ID AMI.

```
aws ssm get-parameter --name /aws/service/bottlerocket/aws-k8s-1.29/x86_64/latest/
image_id --region region-code --query "Parameter.Value" --output text
```

Contoh output adalah sebagai berikut.

```
ami-1234567890abcdef0
```

BottlerocketDukungan kepatuhan

Bottlerocketsesuai dengan rekomendasi yang didefinisikan oleh berbagai organisasi:

- Ada Benchmark CIS didefinisikan untukBottlerocket. Dalam konfigurasi default,Bottlerocket gambar memiliki sebagian besar kontrol yang diperlukan oleh profil konfigurasi CIS Level 1. Anda dapat menerapkan kontrol yang diperlukan untuk profil konfigurasi CIS Level 2. Untuk informasi selengkapnya, lihat MemvalidasiBottlerocket AMI yang dioptimalkan Amazon EKS terhadap Tolok Ukur CIS diAWS blog.
- Set fitur yang dioptimalkan dan permukaan serangan yang dikurangi berarti bahwaBottlerocket instance memerlukan lebih sedikit konfigurasi untuk memenuhi persyaratan PCI DSS. Benchmark CIS untukBottlerocket adalah sumber daya yang sangat baik untuk panduan pengerasan, dan mendukung kebutuhan Anda untuk standar konfigurasi yang aman di bawah persyaratan PCI DSS 2.2. Anda juga dapat memanfaatkan Fluent Bit untuk mendukung kebutuhan Anda untuk pencatatan audit tingkat sistem operasi berdasarkan persyaratan PCI DSS 10.2. AWSmenerbitkanBottlerocket instans baru (ditambal) secara berkala untuk membantu Anda memenuhi persyaratan PCI DSS 6.2 (untuk v3.2.1) dan persyaratan 6.3.3 (untuk v4.0).
- Bottlerocketadalah fitur yang memenuhi syarat HIPAA yang diotorisasi untuk digunakan dengan beban kerja teregulasi untuk Amazon EC2 dan Amazon EKS. Untuk informasi selengkapnya, lihat Merancang untuk Keamanan dan Kepatuhan HIPAA pada Amazon EKS.

Ubuntu Linux AMI yang dioptimalkan oleh Amazon EKS

Canonical telah bermitra dengan Amazon EKS untuk membuat simpul AMI yang dapat Anda gunakan dalam klaster Anda.

Canonical memberikan gambar built-for-purposeKubernetes Node OS. UbuntuCitra yang diminimumkan ini dioptimalkan untuk Amazon EKS dan mencakupAWS kernel kustom yang

dikembangkan bersama denganAWS. Untuk informasi selengkapnya, lihat Ubuntudi Amazon Elastic Kubernetes Service (EKS). Untuk informasi tentang Support, lihat bagian Perangkat lunak pihak ketiga dari FAQ DukunganAWS Premium.

Amazon EKS dioptimalkan Windows AMI

WindowsAMI yang dioptimalkan Amazon EKS dibangun di atas Windows Server 2019 dan Windows Server 2022. Mereka dikonfigurasi untuk berfungsi sebagai gambar dasar untuk node Amazon EKS. Secara default, AMI menyertakan komponen-komponen berikut:

- `kubelet`
- `kube-proxy`
- AWS Authenticator IAM untuk Kubernetes
- `csi-proxy`
- `containerd`

> **Note**
>
> Anda dapat melacak peristiwa keamanan atau privasi untuk Windows Server dengan panduan pembaruan Microsoft keamanan.

Amazon EKS menawarkan AMI yang dioptimalkan untuk Windows wadah dalam varian berikut:

- WindowsServer AMI Inti 2019 yang Dioptimalkan Amazon EKS
- Amazon EKS Windows Server 2019 AMI Penuh yang Dioptimalkan
- WindowsServer Amazon EKS yang Dioptimalkan 2022 Core AMI
- WindowsServer Amazon EKS yang dioptimalkan 2022 AMI Lengkap

> **Important**
>
> - 20H2AMI Inti Windows Server Amazon yang dioptimalkan EKS tidak digunakan lagi. Tidak ada versi baru dari AMI ini yang akan dirilis.
> - Untuk memastikan bahwa Anda memiliki pembaruan keamanan terbaru secara default, Amazon EKS mempertahankan Windows AMI yang dioptimalkan selama 4 bulan terakhir.

Setiap AMI baru akan tersedia selama 4 bulan sejak rilis awal. Setelah periode ini, AMI yang lebih tua dibuat pribadi dan tidak lagi dapat diakses. Kami mendorong penggunaan AMI terbaru untuk menghindari kerentanan keamanan dan kehilangan akses ke AMI lama yang telah mencapai akhir masa pakai yang didukung. Meskipun kami tidak dapat menjamin bahwa kami dapat menyediakan akses ke AMI yang telah dibuat pribadi, Anda dapat meminta akses dengan AWS Support mengajukan tiket.

Kalender rilis

Tabel berikut mencantumkan tanggal rilis dan akhir dukungan untuk Windows versi di Amazon EKS. Jika tanggal akhir kosong, itu tandanya versi masih didukung.

Versi Windows	Rilis Amazon EKS	Akhir dukungan Amazon EKS
WindowsServer 2022 Inti	10/17/2022	
WindowsServer 2022 Lengkap	10/17/2022	
Windows20H2Inti Server	8/12/2021	8/9/2022
WindowsServer 2004 Inti	8/19/2020	12/14/2021
WindowsServer 2019 Inti	10/7/2019	
WindowsServer 2019 Penuh	10/7/2019	
WindowsServer 1909 Inti	10/7/2019	12/8/2020

Parameter konfigurasi skrip bootstrap

Saat Anda membuat Windows node, ada skrip pada node yang memungkinkan untuk mengkonfigurasi parameter yang berbeda. Bergantung pada pengaturan Anda, skrip ini dapat ditemukan di node di lokasi yang mirip dengan:`C:\Program Files\Amazon\EKS\Start-EKSBootstrap.ps1`. Anda dapat menentukan nilai parameter kustom dengan menentukannya sebagai argumen untuk skrip bootstrap. Misalnya, Anda dapat memperbarui data pengguna di template peluncuran. Untuk informasi selengkapnya, lihat Data pengguna Amazon EC2.

Skrip ini mencakup parameter baris perintah berikut:

- `-EKSClusterName`— Menentukan nama cluster Amazon EKS untuk node pekerja ini untuk bergabung.
- `-KubeletExtraArgs`- Menentukan argumen tambahan untuk `kubelet` (opsional).
- `-KubeProxyExtraArgs`- Menentukan argumen tambahan untuk `kube-proxy` (opsional).
- `-APIServerEndpoint`— Menentukan titik akhir server API cluster Amazon EKS (opsional). Hanya berlaku bila digunakan dengan`-Base64ClusterCA`. Melewati panggilan`Get-EKSCluster`.
- `-Base64ClusterCA`- Menentukan base64 dikodekan konten CA cluster (opsional). Hanya berlaku bila digunakan dengan`-APIServerEndpoint`. Melewati panggilan`Get-EKSCluster`.
- `-DNSClusterIP`— Mengganti alamat IP yang akan digunakan untuk kueri DNS dalam cluster (opsional). Default ke `10.100.0.10` atau `172.20.0.10` berdasarkan alamat IP antarmuka utama.
- `-ServiceCIDR`— Mengganti rentang alamat IP Kubernetes layanan dari mana layanan cluster ditangani. Default ke `172.20.0.0/16` atau `10.100.0.0/16` berdasarkan alamat IP antarmuka utama.
- `-ExcludedSnatCIDRs`— Daftar `IPv4` CIDR untuk dikecualikan dari Source Network Address Translation (SNAT). Ini berarti bahwa IP pribadi pod yang dapat dialamatkan VPC tidak akan diterjemahkan ke alamat IP dari `IPv4` alamat utama ENI instance untuk lalu lintas keluar. Secara default, `IPv4` CIDR VPC untuk node Amazon Windows EKS ditambahkan. Menentukan CIDR ke parameter ini juga mengecualikan CIDR yang ditentukan. Untuk informasi selengkapnya, lihat SNAT untuk Pods.

Selain parameter baris perintah, Anda juga dapat menentukan beberapa parameter variabel lingkungan. Saat menentukan parameter baris perintah, itu diutamakan daripada variabel lingkungan masing-masing. Variabel lingkungan harus didefinisikan sebagai cakupan mesin (atau sistem) karena skrip bootstrap hanya akan membaca variabel cakupan mesin.

Skrip memperhitungkan variabel lingkungan berikut:

- `SERVICE_IPV4_CIDR`— Lihat parameter baris `ServiceCIDR` perintah untuk definisi.
- `EXCLUDED_SNAT_CIDRS`— Harus berupa string yang dipisahkan koma. Lihat parameter baris `ExcludedSnatCIDRs` perintah untuk definisi.

Luncurkan node Windows Server 2022 yang dikelola sendiri dengan **`eksctl`**

Anda dapat menggunakan berikut ini **`test-windows-2022.`**`yaml` sebagai referensi untuk menjalankan Windows Server 2022 sebagai node yang dikelola sendiri. Ganti setiap *`example value`* dengan nilai-nilai Anda sendiri.

> **Note**
>
> Anda harus menggunakan `eksctl` versi `0.116.0`atau yang lebih baru untuk menjalankan node Windows Server 2022 yang dikelola sendiri.

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

metadata:
  name: windows-2022-cluster
  region: region-code
  version: '1.29'

nodeGroups:
  - name: windows-ng
    instanceType: m5.2xlarge
    amiFamily: WindowsServer2022FullContainer
    volumeSize: 100
    minSize: 2
    maxSize: 3
  - name: linux-ng
    amiFamily: AmazonLinux2
    minSize: 2
    maxSize: 3
```

Grup node kemudian dapat dibuat menggunakan perintah berikut.

```
eksctl create cluster -f test-windows-2022.yaml
```

gMSAdukungan otentikasi

Amazon EKS Windows Pods memungkinkan berbagai jenis grup Managed Service Account (gMSA) otentikasi.

- Amazon EKS mendukung identitas Active Directory domain untuk otentikasi. Untuk informasi selengkapnya tentang bergabung dengan domaingMSA, lihat WindowsOtentikasi di Amazon Windowspods EKS di blog. AWS
- Amazon EKS menawarkan plugin yang memungkinkan non-domain-joined Windows node untuk mengambil gMSA kredensi dengan identitas pengguna portabel. Untuk informasi selengkapnya tentang domainlessgMSA, lihat Otentikasi Tanpa Domain Windows untuk Amazon EKS di blog. Windows pods AWS

Gambar kontainer yang di-cache

Amazon EKS Windows AMI yang dioptimalkan memiliki gambar kontainer tertentu yang di-cache untuk `containerd` runtime. Gambar kontainer di-cache saat membuat AMI kustom menggunakan komponen build yang dikelola Amazon. Untuk informasi selengkapnya, lihat Menggunakan komponen build yang dikelola Amazon.

Gambar kontainer cache berikut adalah untuk `containerd` runtime:

- `amazonaws.com/eks/pause-windows`
- `mcr.microsoft.com/windows/nanoserver`
- `mcr.microsoft.com/windows/servercore`

Informasi lain

Untuk informasi selengkapnya tentang penggunaan Windows AMI yang dioptimalkan Amazon EKS, lihat bagian berikut:

- Untuk menggunakan Windows dengan grup node terkelola, lihatGrup simpul terkelola.
- Untuk meluncurkan Windows node yang dikelola sendiri, lihatMeluncurkan node yang dikelola sendiri Windows.
- Untuk informasi versi, lihat Amazon EKS dioptimalkan versi Windows AMI.
- Untuk mengambil ID terbaru dari Windows AMI Amazon EKS yang dioptimalkan, lihatMengambil ID Windows AMI Amazon EKS yang dioptimalkan.
- Untuk menggunakan Amazon EC2 Image Builder untuk membuat Windows AMI yang dioptimalkan Amazon EKS khusus, lihat. Membuat Windows AMI Amazon EKS kustom yang dioptimalkan
- Untuk praktik terbaik, lihat Manajemen Windows AMI Amazon EKS yang dioptimalkan di Panduan Praktik Terbaik EKS.

Amazon EKS dioptimalkan versi Windows AMI

> **Important**
>
> Extended Support untuk Amazon EKS Windows AMI yang dioptimalkan yang diterbitkan oleh AWS tidak tersedia untuk Kubernetes versi `1.23` tetapi tersedia untuk Kubernetes versi `1.24` dan lebih tinggi.

Topik ini mencantumkan versi Windows AMI Amazon EKS yang dioptimalkan dan versi yang sesuai dari `kubelet`,`containerd`,, dan `csi-proxy`.

Metadata AMI yang dioptimalkan oleh Amazon EKS, termasuk ID AMI, untuk setiap varian bisa didapatkan secara terprogram. Untuk informasi selengkapnya, lihat Mengambil ID Windows AMI Amazon EKS yang dioptimalkan.

AMI diversi berdasarkan Kubernetes versi dan tanggal rilis AMI dalam format berikut:

```
k8s_major_version.k8s_minor_version-release_date
```

> **Note**
>
> Grup simpul terkelola Amazon EKS mendukung rilis Windows AMI November 2022 dan selanjutnya.

Amazon EKS Windows Server yang dioptimalkan 2022 Core AMI

Tabel berikut mencantumkan versi saat ini dan sebelumnya dari Amazon EKS Windows Server 2022 Core AMI yang dioptimalkan.

Kubernetes version 1.29

Kubernetesversi **`1.29`**

Versi AMI	Versi **`kubelet`**	Versi **`containerd`**	Versi **`csi-proxy`**	Catatan rilis
`1.29-2024.04.09`	`1.29.0`	`1.6.28`	`1.1.2`	Ditingkatkan `containerd` ke`1.6.28`. Membangun kembali CNI dan `csi-proxy` menggunakan. `golang 1.22.1`
`1.29-2024.03.12`	`1.29.0`	`1.6.25`	`1.1.2`	
`1.29-2024.02.13`	`1.29.0`	`1.6.25`	`1.1.2`	
`1.29-2024.02.06`	`1.29.0`	`1.6.25`	`1.1.2`	Memperbaiki bug di mana gambar jeda salah dihapus oleh proses pengumpulan `kubelet` sampah.
`1.29-2024.01.11`	`1.29.0`	`1.6.18`	`1.1.2`	WindowsPembaruan Mandiri yang Dikecualikan KB5034439 di Server 2022 Core AMI. Windows KB hanya berlaku untuk Windows instalasi dengan WinRE partisi terpisah, yang tidak disertakan dengan Windows AMI Amazon EKS Optimized kami.

Kubernetes version 1.28

Kubernetesversi **1.28**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.28-2024.03.12	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.11	1.28.5	1.6.18	1.1.2	WindowsPembaruan Mandiri yang Dikecualikan KB5034439 di Server 2022 Core AMI. Windows KB hanya berlaku untuk Windows instalasi dengan WinRE partisi terpisah, yang tidak disertakan dengan Windows AMI Amazon EKS Optimized kami.
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_I

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				`PV4_CIDR` dan`EXCLUDED_SNAT_CIDRS`).
`1.28-2023-09.27`	`1.28.2`	`1.6.6`	`1.1.2`	Memperbaiki penasihat keamanan dikubelet.
`1.28-2023.09.12`	`1.28.1`	`1.6.6`	`1.1.2`	

Kubernetes version 1.27

Kubernetesversi **1.27**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
`1.27-2024.04.09`	`1.27.9`	`1.6.25`	`1.1.2`	Ditingkatkan `containerd` ke`1.6.25`. Membangun kembali CNI dan `csi-proxy` menggunakan. `golang 1.22.1`
`1.27-2024.03.12`	`1.27.9`	`1.6.18`	`1.1.2`	
`1.27-2024.02.13`	`1.27.9`	`1.6.18`	`1.1.2`	
`1.27-2024.01.11`	`1.27.9`	`1.6.18`	`1.1.2`	WindowsPembaruan Mandiri yang Dikecualikan KB5034439 di Server 2022 Core AMI. Windows KB hanya berlaku untuk Windows instalasi dengan WinRE partisi terpisah, yang tidak disertakan

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				dengan Windows AMI Amazon EKS Optimized kami.
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Memperbaiki penasihat keamanan dikubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod dari Kubernetes server API. Permintaan tarik gabungan #100.
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	Memperbaiki containers-roadmap masalah #2042, yang menyebabkan node gagal menarik gambar ECR Amazon pribadi.
1.27-2023.05.17	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetesversi **1.26**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.26-2024.03.12	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.11	1.26.12	1.6.18	1.1.2	WindowsPembaruan Mandiri yang Dikecualikan KB5034439 di Server 2022 Core AMI. Windows KB hanya berlaku untuk Windows instalasi dengan WinRE partisi terpisah, yang tidak disertakan dengan Windows AMI Amazon EKS Optimized kami.
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditingkatkan kubelet ke1.26.9. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod dari Kubernetes server API. Permintaan tarik gabungan #100.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Ditingkatkan Kubernetes ke1.26.4. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

Kubernetesversi **1.25**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.25-2024.03.12	1.25.16	1.6.18	1.1.2	

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.11	1.25.16	1.6.18	1.1.2	WindowsPembaruan Mandiri yang Dikecualikan KB5034439 di Server 2022 Core AMI. Windows KB hanya berlaku untuk Windows instalasi dengan WinRE partisi terpisah, yang tidak disertakan dengan Windows AMI Amazon EKS Optimized kami.
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditingkatkan kubelet ke1.25.14. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod dari Kubernetes server API. Permintaan tarik gabungan #100.

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Termasuk tambalan untuk CVE-2023-3676 , CVE-2023-3893 , dan CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Ditingkatkan Kubernetes ke 1.25.9. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	Menginstal gMSAplugin tanpa domain untuk memfasilitasi gMSA otentikasi kontainer Windows di Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetesversi **1.24**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.24-2024.03.12	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.11	1.24.17	1.6.18	1.1.2	WindowsPembaruan Mandiri yang Dikecualikan KB5034439 di Server 2022 Core AMI. Windows KB hanya berlaku untuk Windows

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				instalasi dengan WinRE partisi terpisah, yang tidak disertakan dengan Windows AMI Amazon EKS Optimized kami.
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditingkatkan kubelet ke1.24.17. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod dari Kubernetes server API. Permintaan tarik gabungan #100.
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Ditingkatkan Kubernetes ke1.24.13. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	Menginstal gMSAplugin tanpa domain untuk memfasilitasi otentikasi GMSA untuk kontainer di Amazon EKS. Windows

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Kubernetesversi diturunkan menjadi 1.24.7 karena 1.24.10 memiliki masalah yang dilaporkan di. kube-proxy
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	
1.24-2022.12.13	1.24.7	1.6.6	1.1.1	
1.24-2022.10.11	1.24.7	1.6.6	1.1.1	

Amazon EKS Windows Server yang dioptimalkan 2022 AMI Penuh

Tabel berikut mencantumkan versi saat ini dan sebelumnya dari Amazon EKS Windows Server 2022 Full AMI yang dioptimalkan.

Kubernetes version 1.29

Kubernetesversi **1.29**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	Ditingkatkan containerd ke1.6.28. Membangun

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				kembali CNI dan `csi-proxy` menggunakan. `golang 1.22.1`
`1.29-2024.03.12`	`1.29.0`	`1.6.25`	`1.1.2`	
`1.29-2024.02.13`	`1.29.0`	`1.6.25`	`1.1.2`	
`1.29-2024.02.06`	`1.29.0`	`1.6.25`	`1.1.2`	Memperbaiki bug di mana gambar jeda salah dihapus oleh proses pengumpulan `kubelet` sampah.
`1.29-2024.01.09`	`1.29.0`	`1.6.18`	`1.1.2`	

Kubernetes version 1.28

Kubernetesversi **`1.28`**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
`1.28-2024.04.09`	`1.28.5`	`1.6.25`	`1.1.2`	Ditingkatkan `containerd` ke`1.6.25`. Membangun kembali CNI dan `csi-proxy` menggunakan. `golang 1.22.1`
`1.28-2024.03.12`	`1.28.5`	`1.6.18`	`1.1.2`	
`1.28-2024.02.13`	`1.28.5`	`1.6.18`	`1.1.2`	

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Memperbaiki penasihat keamanan dikubelet.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1.27

Kubernetesversi **1.27**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.27-2024.03.12	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Memperbaiki penasihat keamanan dikubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod dari Kubernetes server API. Permintaan tarik gabungan #100.

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	Memperbaiki containers-roadmap masalah #2042, yang menyebabkan node gagal menarik gambar ECR Amazon pribadi.
1.27-2023.05.18	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetesversi **1.26**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.26-2024.03.12	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditingkatkan kubelet ke1.26.9. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				dari Kubernetes server API. Permintaan tarik gabungan #100.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Ditingkatkan Kubernetes ke1.26.4. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

Kubernetesversi **1.25**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.25-2024.03.12	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.09	1.25.16	1.6.18	1.1.2	
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	Termasuk tambalan untuk CVE-2023-5528 .
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	Ditingkatkan containerd ke 1.6.18. Ditingkatkan kubelet ke 1.25.14. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR dan EXCLUDED_SNAT_CIDRS).
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod dari Kubernetes server API. Permintaan tarik gabungan #100.
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Termasuk tambalan untuk CVE-2023-3676 , CVE-2023-3893 , dan CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Ditingkatkan Kubernetes ke1.25.9. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	Menginstal gMSAplugin tanpa domain untuk memfasilitasi gMSA otentikasi kontainer Windows di Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetesversi **1.24**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.24-2024.03.12	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditingkatkan kubelet ke1.24.17. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				dari Kubernetes server API. Permintaan tarik gabungan #100.
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Ditingkatkan Kubernetes ke1.24.13. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	Menginstal gMSAplugin tanpa domain untuk memfasilitasi gMSA otentikasi kontainer Windows di Amazon EKS.
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Kubernetesversi diturunkan menjadi 1.24.7 karena 1.24.10 memiliki masalah yang dilaporkan di. kube-proxy
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	
1.24-2022.12.14	1.24.7	1.6.6	1.1.1	
1.24-2022.10.11	1.24.7	1.6.6	1.1.1	

Amazon EKS Windows Server 2019 Core AMI yang dioptimalkan

Tabel berikut mencantumkan versi saat ini dan sebelumnya dari Amazon EKS Windows Server 2019 Core AMI yang dioptimalkan.

Kubernetes version 1.29

Kubernetesversi **1.29**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	Ditingkatkan containerd ke1.6.28. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.29-2024.03.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Memperbaiki bug di mana gambar jeda salah dihapus oleh proses pengumpulan kubelet sampah.
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes version 1.28

Kubernetesversi **1.28**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.28-2024.03.13	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Memperbaiki penasihat keamanan dikubelet.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1.27

Kubernetesversi **1.27**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.27-2024.03.13	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Memperbaiki penasihat keamanan dikubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				Kubernetes biner konektor, yang mendapatkan alamat IP Pod dari Kubernetes server API. Permintaan tarik gabungan #100.
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	Memperbaiki containers-roadmap masalah #2042, yang menyebabkan node gagal menarik gambar ECR Amazon pribadi.
11.27-2023.05.18	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetesversi **1.26**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.26-2024.03.13	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditingkatkan kubelet ke1.26.9. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				dari Kubernetes server API. Permintaan tarik gabungan #100.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Ditingkatkan Kubernetes ke1.26.4. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

Kubernetesversi **1.25**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.25-2024.03.13	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.09	1.25.16	1.6.18	1.1.2	
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	Termasuk tambalan untuk CVE-2023-5528 .
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	Ditingkatkan containerd ke 1.6.18. Ditingkatkan kubelet ke 1.25.14. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR dan EXCLUDED_SNAT_CIDRS).
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod dari Kubernetes server API. Permintaan tarik gabungan #100.
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Termasuk tambalan untuk CVE-2023-3676 , CVE-2023-3893 , dan CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Ditingkatkan Kubernetes ke1.25.9. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	Menginstal gMSAplugin tanpa domain untuk memfasilitasi gMSA otentikasi kontainer Windows di Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetesversi **1.24**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.24-2024.03.13	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditingkatkan kubelet ke1.24.17. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				dari Kubernetes server API. Permintaan tarik gabungan #100.
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Ditingkatkan Kubernetes ke1.24.13. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	Menginstal gMSAplugin tanpa domain untuk memfasilitasi gMSA otentikasi kontainer Windows di Amazon EKS.
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Kubernetesversi diturunkan menjadi 1.24.7 karena 1.24.10 memiliki masalah yang dilaporkan di. kube-proxy
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	
1.24-2022.12.13	1.24.7	1.6.6	1.1.1	
1.24-2022.11.08	1.24.7	1.6.6	1.1.1	

Amazon EKS Windows Server 2019 AMI Penuh yang dioptimalkan

Tabel berikut mencantumkan versi saat ini dan sebelumnya dari Amazon EKS Windows Server 2019 Full AMI yang dioptimalkan.

Kubernetes version 1.29

Kubernetesversi **1.29**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	Ditingkatkan containerd ke1.6.28. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.29-2024.03.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Memperbaiki bug di mana gambar jeda salah dihapus oleh proses pengumpulan kubelet sampah.
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes version 1.28

Kubernetesversi **1.28**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.28-2024.03.13	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.09	1.28.5	1.6.18	1.1.2	
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Memperbaiki penasihat keamanan dikubelet.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1.27

Kubernetesversi **1.27**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.27-2024.03.13	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Memperbaiki penasihat keamanan dikubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				Kubernetes biner konektor, yang mendapatkan alamat IP Pod dari Kubernetes server API. Permintaan tarik gabungan #100.
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	Memperbaiki containers-roadmap masalah #2042, yang menyebabkan node gagal menarik gambar ECR Amazon pribadi.
1.27-2023.05.17	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetesversi **1.26**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.26-2024.03.13	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditingkatkan kubelet ke1.26.9. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				dari Kubernetes server API. Permintaan tarik gabungan #100.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Ditingkatkan Kubernetes ke1.26.4. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

Kubernetesversi **1.25**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.25-2024.03.13	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.09	1.25.16	1.6.18	1.1.2	
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	Termasuk tambalan untuk CVE-2023-5528 .
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	Ditingkatkan containerd ke 1.6.18. Ditingkatkan kubelet ke 1.25.14. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR dan EXCLUDED_SNAT_CIDRS).
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod dari Kubernetes server API. Permintaan tarik gabungan #100.
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Termasuk tambalan untuk CVE-2023-3676 , CVE-2023-3893 , dan CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Ditingkatkan Kubernetes ke1.25.9. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	Menginstal gMSAplugin tanpa domain untuk memfasilitasi gMSA otentikasi kontainer Windows di Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetesversi **1.24**

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	Ditingkatkan containerd ke1.6.25. Membangun kembali CNI dan csi-proxy menggunakan. golang 1.22.1
1.24-2024.03.13	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Termasuk tambalan untukCVE-2023-5528 .
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	Ditingkatkan containerd ke1.6.18. Ditingkatkan kubelet ke1.24.17. Ditambahkan variabel lingkungan skrip bootstrap baru (SERVICE_IPV4_CIDR danEXCLUDED_SNAT_CIDRS).
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	Upgrade plugin Amazon VPC CNI untuk menggunakan Kubernetes biner konektor, yang mendapatkan alamat IP Pod

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
				dari Kubernetes server API. Permintaan tarik gabungan #100.
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Termasuk tambalan untukCVE-2023-3676 ,CVE-2023-3893 , danCVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.21	1.24.13	1.6.6	1.1.1	Masalah teratasi yang menyebabkan daftar pencarian akhiran DNS salah diisi.
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Ditingkatkan Kubernetes ke1.24.13. Menambahkan dukungan untuk pemetaan port host di CNI. Permintaan tarik gabungan #93.
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	Memperbaiki bug yang menyebabkan masalah konektivitas jaringan #1126 pada pod setelah node restart. Memperkenalkan parameter konfigurasi skrip bootstrap baru (ExcludedSnatCIDRs).

Versi AMI	Versi **kubelet**	Versi **containerd**	Versi **csi-proxy**	Catatan rilis
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	Menambahkan mekanisme pemulihan untuk kubelet dan kube-proxy pada kerusakan layanan.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	Menginstal gMSAplugin tanpa domain untuk memfasilitasi gMSA otentikasi kontainer Windows di Amazon EKS.
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Kubernetesversi diturunkan menjadi 1.24.7 karena 1.24.10 memiliki masalah yang dilaporkan di. kube-proxy
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	
1.24-2022.12.14	1.24.7	1.6.6	1.1.1	
1.24-2022.10.12	1.24.7	1.6.6	1.1.1	

Mengambil ID Windows AMI Amazon EKS yang dioptimalkan

Anda dapat mengambil ID Amazon Machine Image (AMI) secara terprogram untuk AMI yang dioptimalkan Amazon EKS dengan menanyakan API Parameter Store. AWS Systems Manager

Parameter ini mengeliminasi kebutuhan Anda untuk mencari ID AMI yang dioptimalkan oleh Amazon EKS secara manual. Untuk informasi selengkapnya tentang Systems Manager Parameter Store API, lihat GetParameter. Prinsip IAM yang Anda gunakan harus memiliki izin `ssm:GetParameter` IAM untuk mengambil metadata AMI Amazon EKS yang dioptimalkan.

Anda dapat mengambil ID gambar dari Windows AMI Amazon EKS terbaru yang dioptimalkan dengan perintah berikut dengan menggunakan `image_id` sub-parameter. Anda dapat mengganti *`1.29`* dengan versi Amazon EKS yang didukung dan dapat menggantinya *`region-code`* dengan Wilayah yang didukung Amazon EKS yang Anda inginkan ID AMI. Ganti *`Core`* dengan `Full` untuk melihat ID AMI lengkap Windows Server. Untuk Kubernetes versi `1.24` atau yang lebih baru, Anda dapat mengganti *`2019`* dengan `2022` untuk melihat ID AMI Windows Server 2022.

```
aws ssm get-parameter --name /aws/service/ami-windows-latest/Windows_Server-2019-
English-Core-EKS_Optimized-1.29/image_id --region region-code --query "Parameter.Value"
 --output text
```

Contoh output adalah sebagai berikut.

```
ami-1234567890abcdef0
```

Membuat Windows AMI Amazon EKS kustom yang dioptimalkan

Anda dapat menggunakan EC2 Image Builder untuk membuat AMI yang Windows dioptimalkan Amazon EKS khusus dengan salah satu opsi berikut:

- Menggunakan Amazon EKS yang dioptimalkan Windows AMI sebagai basis
- Menggunakan komponen build yang dikelola Amazon

Dengan kedua metode tersebut, Anda harus membuat resep Image Builder Anda sendiri. Untuk informasi selengkapnya, lihat Membuat versi baru resep gambar di Panduan Pengguna Image Builder.

> **Important**
>
> Komponen yang dikelola Amazon berikut untuk `eks` menyertakan tambalan untuk. `CVE-2023-5528`
>
> - `1.24.3`dan lebih tinggi

- `1.25.2`dan lebih tinggi
- `1.26.2`dan lebih tinggi
- `1.27.0`dan lebih tinggi
- `1.28.0`dan lebih tinggi

Menggunakan Amazon EKS yang dioptimalkan Windows AMI sebagai basis

Opsi ini adalah cara yang disarankan untuk membangun Windows AMI kustom Anda. WindowsAMI yang dioptimalkan Amazon EKS yang kami sediakan lebih sering diperbarui daripada komponen build yang dikelola Amazon.

1. Mulai resep Image Builder baru.
   a. Buka konsol EC2 Image Builder di https://console.aws.amazon.com/imagebuilder.
   b. Di panel navigasi kiri, pilih Resep gambar.
   c. Pilih Buat resep gambar.
2. Di bagian Rincian resep, masukkan Nama dan Versi.
3. Tentukan ID Windows AMI Amazon EKS yang dioptimalkan di bagian Gambar dasar.
   a. Pilih Masukkan ID AMI kustom.
   b. Ambil ID AMI untuk versi Windows OS yang Anda butuhkan. Untuk informasi selengkapnya, lihat Mengambil ID Windows AMI Amazon EKS yang dioptimalkan.
   c. Masukkan ID AMI kustom. Jika ID AMI tidak ditemukan, pastikan ID Wilayah AWS untuk AMI cocok dengan yang Wilayah AWS ditampilkan di kanan atas konsol Anda.
4. (Opsional) Untuk mendapatkan pembaruan keamanan terbaru, tambahkan `update-windows` komponen di bagian Build components -.
   a. Dari daftar dropdown di sebelah kanan kotak pencarian Cari komponen berdasarkan nama, pilih dikelola Amazon.
   b. Di kotak Cari komponen dengan nama pencarian, masukkan**`update-windows`**.
   c. Pilih kotak centang hasil **`update-windows`**pencarian. Komponen ini mencakup Windows tambalan terbaru untuk sistem operasi.

5. Lengkapi masukan resep gambar yang tersisa dengan konfigurasi yang Anda butuhkan. Untuk informasi selengkapnya, lihat Membuat versi resep gambar baru (konsol) di Panduan Pengguna Image Builder.
6. Pilih Buat resep.
7. Gunakan resep gambar baru di pipeline gambar baru atau yang sudah ada. Setelah pipeline gambar Anda berjalan dengan sukses, AMI kustom Anda akan terdaftar sebagai gambar keluaran dan siap digunakan. Untuk informasi selengkapnya, lihat Membuat pipeline gambar menggunakan wizard konsol EC2 Image Builder.

Menggunakan komponen build yang dikelola Amazon

Saat menggunakan Windows AMI Amazon EKS yang dioptimalkan sebagai basis tidak layak, Anda dapat menggunakan komponen build yang dikelola Amazon sebagai gantinya. Opsi ini mungkin tertinggal dari Kubernetes versi terbaru yang didukung.

1. Mulai resep Image Builder baru.
   a. Buka konsol EC2 Image Builder di https://console.aws.amazon.com/imagebuilder.
   b. Di panel navigasi kiri, pilih Resep gambar.
   c. Pilih Buat resep gambar.
2. Di bagian Rincian resep, masukkan Nama dan Versi.
3. Tentukan opsi mana yang akan Anda gunakan untuk membuat AMI kustom Anda di bagian Gambar dasar:
   - Pilih gambar terkelola - Pilih Windows untuk Sistem Operasi Gambar (OS) Anda. Kemudian pilih salah satu opsi berikut untuk Asal gambar.
     - Mulai cepat (dikelola Amazon) - Di dropdown nama Gambar, pilih versi Server yang didukung Amazon EKS. Windows Untuk informasi selengkapnya, lihat Amazon EKS dioptimalkan Windows AMI.
     - Gambar yang dimiliki oleh saya - Untuk nama Gambar, pilih ARN gambar Anda sendiri dengan lisensi Anda sendiri. Citra yang Anda berikan belum memiliki komponen-komponen Amazon EKS yang diinstal.
   - Masukkan ID AMI khusus — Untuk ID AMI, masukkan ID untuk AMI Anda dengan lisensi Anda sendiri. Citra yang Anda berikan belum memiliki komponen-komponen Amazon EKS yang diinstal.
4. Di bagian Build components - Windows, lakukan hal berikut:

a. Dari daftar dropdown di sebelah kanan kotak pencarian Cari komponen berdasarkan nama, pilih dikelola Amazon.

b. Di kotak Cari komponen dengan nama pencarian, masukkan**eks**.

c. Pilih kotak centang hasil **eks-optimized-ami-windows**pencarian, meskipun hasil yang dikembalikan mungkin bukan versi yang Anda inginkan.

d. Di kotak Cari komponen dengan nama pencarian, masukkan**update-windows**.

e. Pilih kotak centang hasil pencarian pembaruan-jendela. Komponen ini mencakup Windows tambalan terbaru untuk sistem operasi.

5. Di bagian Komponen yang dipilih, lakukan hal berikut:

a. Pilih opsi Versioning untuk. **eks-optimized-ami-windows**

b. Pilih Tentukan versi komponen.

c. Di bidang Versi Komponen, masukkan ***version.x*** , ganti *version*dengan Kubernetes versi yang didukung. Memasukkan bagian *x*untuk nomor versi menunjukkan untuk menggunakan versi komponen terbaru yang juga sejajar dengan bagian versi yang Anda tentukan secara eksplisit. Perhatikan output konsol karena akan memberi tahu Anda apakah versi yang Anda inginkan tersedia sebagai komponen terkelola. Perlu diingat bahwa Kubernetes versi terbaru mungkin tidak tersedia untuk komponen build. Untuk informasi selengkapnya tentang versi yang tersedia, lihatMengambil informasi tentang versi `eks-optimized-ami-windows` komponen.

> **Note**
>
> Versi komponen `eks-optimized-ami-windows` build berikut memerlukan `eksctl` versi `0.129` atau yang lebih rendah:
>
> - `1.24.0`

6. Lengkapi masukan resep gambar yang tersisa dengan konfigurasi yang Anda butuhkan. Untuk informasi selengkapnya, lihat Membuat versi resep gambar baru (konsol) di Panduan Pengguna Image Builder.

7. Pilih Buat resep.

8. Gunakan resep gambar baru di pipeline gambar baru atau yang sudah ada. Setelah pipeline gambar Anda berjalan dengan sukses, AMI kustom Anda akan terdaftar sebagai gambar

keluaran dan siap digunakan. Untuk informasi selengkapnya, lihat Membuat pipeline gambar menggunakan wizard konsol EC2 Image Builder.

Mengambil informasi tentang versi **`eks-optimized-ami-windows`** komponen

Anda dapat mengambil informasi spesifik mengenai apa yang diinstal dengan masing-masing komponen. Misalnya, Anda dapat memverifikasi `kubelet` versi apa yang diinstal. Komponen melalui pengujian fungsional pada versi sistem Windows operasi yang didukung Amazon EKS. Untuk informasi selengkapnya, lihat Kalender rilis. Versi Windows OS lain yang tidak terdaftar sebagai didukung atau telah mencapai akhir dukungan mungkin tidak kompatibel dengan komponen.

1. Buka konsol EC2 Image Builder di https://console.aws.amazon.com/imagebuilder.
2. Di panel navigasi kiri, pilih Komponen.
3. Dari daftar dropdown di sebelah kanan kotak pencarian Cari komponen berdasarkan nama, ubah Dimiliki oleh saya menjadi Mulai cepat (dikelola Amazon).
4. Dalam kotak Temukan komponen berdasarkan nama, masukkan **`eks`**.
5. (Opsional) Jika Anda menggunakan versi terbaru, urutkan kolom Versi dalam urutan menurun dengan memilihnya dua kali.
6. Pilih **`eks-optimized-ami-windows`**tautan dengan versi yang diinginkan.

Deskripsi di halaman yang dihasilkan menunjukkan informasi spesifik.

Penyimpanan

Bab ini mencakup pilihan penyimpanan untuk klaster Amazon EKS.

Topik

Driver CSI Amazon EBS

Driver Amazon Elastic Block Store (Amazon EBS) Container Storage Interface (CSI) mengelola siklus hidup volume Amazon EBS sebagai penyimpanan untuk Volume yang Anda buat. Kubernetes Driver Amazon EBS CSI membuat volume Amazon EBS untuk jenis volume ini: Kubernetes volume fana generik dan volume persisten.

Berikut adalah beberapa hal yang perlu dipertimbangkan saat menggunakan driver Amazon EBS CSI.

- Plugin Amazon EBS CSI memerlukan izin IAM untuk melakukan panggilan ke AWS API atas nama Anda. Untuk informasi selengkapnya, lihat Membuat peran IAM driver Amazon EBS CSI.
- Anda tidak dapat memasang volume Amazon EBS ke FargatePods.
- Anda dapat menjalankan pengontrol Amazon EBS CSI di node Fargate, tetapi node Amazon EBS CSI hanya dapat DaemonSet berjalan di instans Amazon EC2.

Driver Amazon EBS CSI tidak diinstal saat Anda pertama kali membuat cluster. Untuk menggunakan driver, Anda harus menambahkannya sebagai add-on Amazon EKS atau sebagai add-on yang dikelola sendiri.

- Untuk petunjuk tentang cara menambahkannya sebagai add-on Amazon EKS, lihatMengelola driver Amazon EBS CSI sebagai add-on Amazon EKS.
- Untuk petunjuk tentang cara menambahkannya sebagai instalasi yang dikelola sendiri, lihat proyek driver Amazon EBS Container Storage Interface (CSI) aktif. GitHub

Setelah Anda menginstal driver CSI dengan salah satu metode, Anda dapat menguji fungsionalitas dengan aplikasi sampel. Untuk informasi selengkapnya, lihat Menyebarkan aplikasi sampel dan memverifikasi bahwa driver CSI berfungsi.

Membuat peran IAM driver Amazon EBS CSI

Plugin Amazon EBS CSI memerlukan izin IAM untuk melakukan panggilan ke AWS API atas nama Anda. Untuk informasi selengkapnya, lihat Mengatur izin pengemudi diGitHub.

> **Note**
>
> Podsakan memiliki akses ke izin yang ditetapkan ke peran IAM kecuali Anda memblokir akses ke IMDS. Untuk informasi selengkapnya, lihat Praktik terbaik keamanan untuk Amazon EKS.

Prasyarat

- Sebuah klaster yang sudah ada.
- Penyedia AWS Identity and Access Management (IAM) OpenID Connect (OIDC) yang sudah ada untuk cluster Anda. Untuk menentukan apakah Anda sudah memiliki satu, atau harus membuat satu, lihat Buat OIDC penyedia IAM untuk klaster Anda.

Prosedur berikut menunjukkan cara membuat peran IAM dan melampirkan kebijakan AWS terkelola padanya. Anda dapat menggunakan`eksctl`, AWS Management Console, atau AWS CLI.

> **Note**
>
> Langkah-langkah spesifik dalam prosedur ini ditulis untuk menggunakan driver sebagai add-on Amazon EKS. Langkah-langkah yang berbeda diperlukan untuk menggunakan driver sebagai add-on yang dikelola sendiri. Untuk informasi selengkapnya, lihat Mengatur izin driver padaGitHub.

eksctl

Untuk membuat peran IAM plugin Amazon EBS CSI Anda dengan **eksctl**

1. Buat peran IAM dan lampirkan kebijakan. AWS mempertahankan kebijakan AWS terkelola atau Anda dapat membuat kebijakan kustom Anda sendiri. Anda dapat membuat peran IAM dan melampirkan kebijakan AWS terkelola dengan perintah berikut. Ganti *my-cluster* dengan nama klaster Anda. Perintah menyebarkan AWS CloudFormation tumpukan yang membuat peran IAM dan melampirkan kebijakan IAM padanya. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

   ```
   eksctl create iamserviceaccount \
       --name ebs-csi-controller-sa \
       --namespace kube-system \
       --cluster my-cluster \
       --role-name AmazonEKS_EBS_CSI_DriverRole \
       --role-only \
       --attach-policy-arn arn:aws:iam::aws:policy/service-role/
   AmazonEBSCSIDriverPolicy \
       --approve
   ```

2. Jika Anda menggunakan kunci KMS khusus untuk enkripsi pada volume Amazon EBS Anda, sesuaikan peran IAM sesuai kebutuhan. Sebagai contoh, lakukan hal berikut:

   a. Salin dan tempel kode berikut ke *kms-key-for-encryption-on-ebs*`.json` file baru. Ganti *custom-key-arn* dengan ARN kunci KMS kustom.

      ```
      {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": [
              "kms:CreateGrant",
              "kms:ListGrants",
              "kms:RevokeGrant"
            ],
            "Resource": ["custom-key-arn"],
            "Condition": {
              "Bool": {
                "kms:GrantIsForAWSResource": "true"
      ```

```
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": ["custom-key-arn"]
    }
  ]
}
```

b. Buat kebijakan. Anda dapat mengubah *KMS_Key_For_Encryption_On_EBS_Policy* ke nama yang berbeda. Namun, jika Anda melakukannya, pastikan untuk mengubahnya di langkah selanjutnya juga.

```
aws iam create-policy \
  --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
  --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. Lampirkan kebijakan IAM ke peran dengan perintah berikut. Ganti *111122223333* dengan ID akun Anda. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/KMS_Key_For_Encryption_On_EBS_Policy \
  --role-name AmazonEKS_EBS_CSI_DriverRole
```

AWS Management Console

Untuk membuat peran IAM plugin Amazon EBS CSI Anda dengan AWS Management Console

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.

3. Pada halaman Peran, pilih Buat peran.
4. Pada halaman Pilih entitas tepercaya, lakukan hal berikut:
   a. Di bagian Jenis entitas tepercaya, pilih Identitas web.
   b. Untuk penyedia Identitas, pilih URL OpenID Connect penyedia untuk klaster Anda (seperti yang ditunjukkan di bawah Ikhtisar di Amazon EKS).
   c. Untuk Audiens, pilih `sts.amazonaws.com`.
   d. Pilih Berikutnya.
5. Pada halaman Tambahkan izin, lakukan hal berikut:
   a. Di dalam kotak Filter kebijakan, masukkan `AmazonEBSCSIDriverPolicy`.
   b. Pilih kotak centang di sebelah kiri yang `AmazonEBSCSIDriverPolicy` dikembalikan dalam pencarian.
   c. Pilih Berikutnya.
6. Pada halaman Nama, tinjau, dan buat, lakukan hal berikut:
   a. Untuk nama Peran, masukkan nama unik untuk peran Anda, seperti*`AmazonEKS_EBS_CSI_DriverRole`*.
   b. Di bawah Tambahkan tag (Opsional), tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tanda di IAM, lihat Menandai sumber daya IAM di Panduan Pengguna IAM.
   c. Pilih Buat peran.
7. Setelah peran dibuat, pilih peran di konsol untuk dibuka, dan kemudian diedit.
8. Pilih tab Trust relationship, lalu pilih Edit trust policy.
9. Temukan garis yang terlihat mirip dengan baris berikut:

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

Tambahkan koma ke akhir baris sebelumnya, lalu tambahkan baris berikut setelah baris sebelumnya. Ganti *`region-code`* dengan tempat Wilayah AWS cluster Anda berada. Ganti *`EXAMPLED539D4633E53DE1B71EXAMPLE`* dengan ID penyedia OIDC cluster Anda.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:ebs-csi-controller-sa"
```

10. Pilih Perbarui kebijakan untuk menyelesaikan.
11. Jika Anda menggunakan kunci KMS khusus untuk enkripsi pada volume Amazon EBS Anda, sesuaikan peran IAM sesuai kebutuhan. Sebagai contoh, lakukan hal berikut:

    a. Di panel navigasi di sebelah kiri, pilih Kebijakan.

    b. Pada halaman Kebijakan, pilih Buat Kebijakan.

    c. Di halaman Buat kebijakan, pilih tab JSON.

    d. Salin dan tempel kode berikut ke editor, ganti *custom-key-arn* dengan ARN kunci KMS khusus.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": ["custom-key-arn"],
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": ["custom-key-arn"]
    }
  ]
}
```

e. Pilih Berikutnya: Tanda.

f. Pada halaman Tambahkan tag (Opsional), pilih Berikutnya: Ulasan.

g. Untuk Nama, masukkan nama unik untuk kebijakan Anda (misalnya,***KMS_Key_For_Encryption_On_EBS_Policy***).

h. Pilih Buat kebijakan.

i. Di panel navigasi sebelah kiri, pilih Peran.

j. Pilih ***Amazoneks_EBS_CSI_ DriverRole*** di konsol untuk membukanya untuk diedit.

k. Dari daftar tarik-turun Tambahkan izin, pilih Lampirkan kebijakan.

l. Di dalam kotak Filter kebijakan, masukkan *KMS_Key_For_Encryption_On_EBS_Policy*.

m. Pilih kotak centang di sebelah kiri *KMS_Key_For_Encryption_On_EBS_Policy* yang dikembalikan dalam pencarian.

n. Pilih Lampirkan kebijakan.

AWS CLI

Untuk membuat peran IAM plugin Amazon EBS CSI Anda dengan AWS CLI

1. Lihat URL penyedia OIDC klaster Anda. Ganti *my-cluster* dengan nama klaster Anda. Jika output dari perintah adalah None, tinjau Prasyarat.

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

Contoh output adalah sebagai berikut.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Buat peran IAM, berikan tindakan. `AssumeRoleWithWebIdentity`

a. Salin konten berikut ke file yang diberi nama*aws-ebs-csi-driver-trust-policy*`.json`. Ganti *111122223333* dengan ID akun Anda. Ganti *EXAMPLED539D4633E53DE1B71EXAMPLE* dan *region-code* dengan nilai yang dikembalikan pada langkah sebelumnya. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:ebs-csi-controller-sa"
        }
      }
    }
  ]
}
```

b. Buat peran. Anda dapat mengubah *AmazonEKS_EBS_CSI_DriverRole* ke nama yang berbeda. Jika Anda mengubahnya, pastikan untuk mengubahnya di langkah selanjutnya.

```
aws iam create-role \
  --role-name AmazonEKS_EBS_CSI_DriverRole \
  --assume-role-policy-document file://"aws-ebs-csi-driver-trust-
policy.json"
```

3. Lampirkan kebijakan. AWS mempertahankan kebijakan AWS terkelola atau Anda dapat membuat kebijakan kustom Anda sendiri. Lampirkan kebijakan AWS terkelola ke peran dengan perintah berikut. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy \
  --role-name AmazonEKS_EBS_CSI_DriverRole
```

4. Jika Anda menggunakan kunci KMS khusus untuk enkripsi pada volume Amazon EBS Anda, sesuaikan peran IAM sesuai kebutuhan. Sebagai contoh, lakukan hal berikut:

   a. Salin dan tempel kode berikut ke *kms-key-for-encryption-on-ebs*`.json` file baru. Ganti *custom-key-arn* dengan ARN kunci KMS kustom.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": ["custom-key-arn"],
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": ["custom-key-arn"]
    }
  ]
}
```

   b. Buat kebijakan. Anda dapat mengubah *KMS_Key_For_Encryption_On_EBS_Policy* ke nama yang berbeda. Namun, jika Anda melakukannya, pastikan untuk mengubahnya di langkah selanjutnya juga.

```
aws iam create-policy \
```

```
  --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
  --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. Lampirkan kebijakan IAM ke peran dengan perintah berikut. Ganti *111122223333* dengan ID akun Anda. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/KMS_Key_For_Encryption_On_EBS_Policy \
  --role-name AmazonEKS_EBS_CSI_DriverRole
```

Sekarang setelah Anda membuat peran IAM driver Amazon EBS CSI, Anda dapat melanjutkan. Menambahkan add-on driver Amazon EBS CSI Ketika Anda menerapkan plugin dalam prosedur itu, itu membuat dan dikonfigurasi untuk menggunakan akun layanan yang diberi `ebs-csi-controller-sa` nama. Akun layanan terikat pada Kubernetes `clusterrole` yang diberi Kubernetes izin yang diperlukan.

Mengelola driver Amazon EBS CSI sebagai add-on Amazon EKS

Untuk meningkatkan keamanan dan mengurangi jumlah pekerjaan, Anda dapat mengelola driver Amazon EBS CSI sebagai add-on Amazon EKS. Untuk informasi tentang add-on Amazon EKS, lihatAdd-on Amazon EKS. Anda dapat menambahkan add-on Amazon EBS CSI dengan mengikuti langkah-langkah di. Menambahkan add-on driver Amazon EBS CSI

Jika Anda menambahkan add-on Amazon EBS CSI, Anda dapat mengelolanya dengan mengikuti langkah-langkah di bagian danMemperbarui driver Amazon EBS CSI sebagai add-on Amazon EKS. Menghapus add-on Amazon EBS CSI

Prasyarat

- Sebuah klaster yang sudah ada. Untuk melihat versi platform yang diperlukan, jalankan perintah berikut.

```
aws eks describe-addon-versions --addon-name aws-ebs-csi-driver
```

- Penyedia AWS Identity and Access Management (IAM) OpenID Connect (OIDC) yang sudah ada untuk cluster Anda. Untuk menentukan apakah Anda sudah memiliki satu, atau harus membuat satu, lihat Buat OIDC penyedia IAM untuk klaster Anda.
- Peran IAM driver Amazon EBS CSI. Jika Anda tidak memenuhi prasyarat ini, mencoba menginstal add-on dan menjalankan `kubectl describe pvc` akan muncul `failed to provision volume with StorageClass` bersama dengan kesalahan. `could not create volume in EC2: UnauthorizedOperation` Untuk informasi selengkapnya, lihat Membuat peran IAM driver Amazon EBS CSI.
- Jika Anda menggunakan kluster yang dibatasi secara luas `PodSecurityPolicy`, pastikan add-on diberikan izin yang cukup untuk diterapkan. Untuk izin yang diperlukan oleh setiap add-onPod, lihat definisi manifes add-on yang relevan di. GitHub

> **Important**
>
> Untuk menggunakan fungsionalitas snapshot dari driver Amazon EBS CSI, Anda harus menginstal snapshotter eksternal sebelum instalasi add-on. Komponen snapshotter eksternal harus dipasang dengan urutan sebagai berikut:
>
> - `CustomResourceDefinition`(CRD) untuk`volumesnapshotclasses`,`volumesnapshots`, dan `volumesnapshotcontents`
> - RBAC (`ClusterRole`,`ClusterRoleBinding`, dan sebagainya)
> - penyebaran pengontrol
>
> Untuk informasi selengkapnya, lihat CSI Snapshotter di. GitHub

Menambahkan add-on driver Amazon EBS CSI

> **Important**
>
> Sebelum menambahkan driver Amazon EBS sebagai add-on Amazon EKS, konfirmasikan bahwa Anda tidak memiliki versi driver yang dikelola sendiri yang diinstal pada cluster Anda. Jika demikian, lihat Menghapus instalasi driver Amazon EBS CSI yang dikelola sendiri. GitHub

Anda dapat menggunakan`eksctl`,AWS Management Console, atau AWS CLI untuk menambahkan add-on Amazon EBS CSI ke cluster Anda.

eksctl

Untuk menambahkan add-on Amazon EBS CSI menggunakan **`eksctl`**

Jalankan perintah berikut. Ganti *`my-cluster`* dengan nama klaster Anda, *`111122223333`* dengan ID akun Anda, dan *`AmazonEKS_EBS_CSI_DriverRole`* dengan nama peran IAM yang dibuat sebelumnya. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat)Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
eksctl create addon --name aws-ebs-csi-driver --cluster my-cluster --service-
account-role-arn arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --force
```

Jika Anda menghapus **`--force`** opsi dan salah satu pengaturan add-on Amazon EKS bertentangan dengan pengaturan yang ada, pembaruan add-on Amazon EKS gagal, dan Anda menerima pesan kesalahan untuk membantu Anda menyelesaikan konflik. Sebelum menentukan opsi ini, pastikan add-on Amazon EKS tidak mengelola pengaturan yang perlu Anda kelola, karena pengaturan tersebut ditimpa dengan opsi ini. Untuk informasi selengkapnya tentang opsi lain untuk setelan ini, lihat Addons dalam `eksctl` dokumentasi. Untuk informasi selengkapnya tentang manajemen Kubernetes lapangan Amazon EKS, lihat Manajemen lapangan Kubernetes.

AWS Management Console

Untuk menambahkan add-on Amazon EBS CSI menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pada panel navigasi sebelah kiri, pilih Klaster.
3. Pilih nama cluster yang ingin Anda konfigurasikan add-on Amazon EBS CSI.
4. Pilih tab Add-ons.
5. Pilih Get more add-ons
6. Pada halaman Pilih add-on, lakukan hal berikut:
   a. Di bagian Amazon EKS-Addons, pilih kotak centang Amazon EBS CSI Driver.
   b. Pilih Berikutnya.
7. Pada halaman Configure selected add-ons settings, lakukan hal berikut:
   a. Pilih Versi yang ingin Anda gunakan.

b. Untuk Pilih peran IAM, pilih nama peran IAM yang Anda lampirkan pada kebijakan IAM driver Amazon EBS CSI.

c. (Opsional) Anda dapat memperluas Pengaturan konfigurasi opsional. Jika Anda memilih Override untuk metode resolusi Konflik, satu atau beberapa pengaturan untuk add-on yang sudah ada dapat ditimpa dengan pengaturan add-on Amazon EKS. Jika Anda tidak mengaktifkan opsi ini dan ada konflik dengan pengaturan yang ada, operasi gagal. Anda dapat menggunakan pesan kesalahan yang dihasilkan untuk memecahkan masalah konflik. Sebelum memilih opsi ini, pastikan add-on Amazon EKS tidak mengelola pengaturan yang perlu Anda kelola sendiri.

d. Pilih Berikutnya.

8. Pada halaman Review dan add, pilih Create. Setelah penginstalan add-on selesai, Anda melihat add-on yang diinstal.

AWS CLI

Untuk menambahkan add-on Amazon EBS CSI menggunakan AWS CLI

Jalankan perintah berikut. Ganti *my-cluster* dengan nama klaster Anda, *111122223333* dengan ID akun Anda, dan *AmazonEKS_EBS_CSI_DriverRole* dengan nama peran yang dibuat sebelumnya. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat)Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
aws eks create-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole
```

Sekarang setelah Anda menambahkan driver Amazon EBS CSI sebagai add-on Amazon EKS, Anda dapat melanjutkan. Menyebarkan aplikasi sampel dan memverifikasi bahwa driver CSI berfungsi Prosedur itu termasuk pengaturan kelas penyimpanan.

Memperbarui driver Amazon EBS CSI sebagai add-on Amazon EKS

Amazon EKS tidak secara otomatis memperbarui Amazon EBS CSI untuk klaster Anda saat versi baru dirilis atau setelah Anda memperbarui klaster ke versi Kubernetes minor baru. Untuk memperbarui Amazon EBS CSI pada cluster yang ada, Anda harus memulai pembaruan dan kemudian Amazon EKS memperbarui add-on untuk Anda.

eksctl

Untuk memperbarui add-on Amazon EBS CSI menggunakan **eksctl**

1. Periksa versi add-on Amazon EBS CSI Anda saat ini. Ganti *my-cluster* dengan nama klaster Anda.

```
eksctl get addon --name aws-ebs-csi-driver --cluster my-cluster
```

Contoh output adalah sebagai berikut.

```
NAME                    VERSION                     STATUS  ISSUES  IAMROLE
 UPDATE AVAILABLE
aws-ebs-csi-driver      v1.11.2-eksbuild.1          ACTIVE  0
  v1.11.4-eksbuild.1
```

2. Perbarui add-on ke versi yang dikembalikan `UPDATE AVAILABLE` di bawah output dari langkah sebelumnya.

```
eksctl update addon --name aws-ebs-csi-driver --version v1.11.4-eksbuild.1 --
cluster my-cluster \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --force
```

Jika Anda menghapus **--force** opsi dan salah satu pengaturan add-on Amazon EKS bertentangan dengan pengaturan yang ada, pembaruan add-on Amazon EKS gagal, dan Anda menerima pesan kesalahan untuk membantu Anda menyelesaikan konflik. Sebelum menentukan opsi ini, pastikan add-on Amazon EKS tidak mengelola pengaturan yang perlu Anda kelola, karena pengaturan tersebut ditimpa dengan opsi ini. Untuk informasi selengkapnya tentang opsi lain untuk setelan ini, lihat Addons dalam `eksctl` dokumentasi. Untuk informasi selengkapnya tentang manajemen Kubernetes lapangan Amazon EKS, lihat Manajemen lapangan Kubernetes.

AWS Management Console

Untuk memperbarui add-on Amazon EBS CSI menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pada panel navigasi sebelah kiri, pilih Klaster.

3. Pilih nama cluster yang ingin Anda perbarui add-on Amazon EBS CSI.
4. Pilih tab Add-ons.
5. Pilih Driver Amazon EBS CSI.
6. Pilih Edit.
7. Pada halaman Konfigurasi Amazon EBS CSI Driver, lakukan hal berikut:
   a. Pilih Versi yang ingin Anda gunakan.
   b. Untuk Pilih peran IAM, pilih nama peran IAM yang Anda lampirkan pada kebijakan IAM driver Amazon EBS CSI.
   c. (Opsional) Anda dapat memperluas Pengaturan konfigurasi opsional dan memodifikasi sesuai kebutuhan.
   d. Pilih Simpan perubahan.

AWS CLI

Untuk memperbarui add-on Amazon EBS CSI menggunakan AWS CLI

1. Periksa versi add-on Amazon EBS CSI Anda saat ini. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver
 --query "addon.addonVersion" --output text
```

   Contoh output adalah sebagai berikut.

```
v1.11.2-eksbuild.1
```

2. Tentukan versi add-on Amazon EBS CSI mana yang tersedia untuk versi cluster Anda.

```
aws eks describe-addon-versions --addon-name aws-ebs-csi-driver --kubernetes-
version 1.23 \
  --query "addons[].addonVersions[].[addonVersion,
 compatibilities[].defaultVersion]" --output text
```

   Contoh output adalah sebagai berikut.

```
v1.11.4-eksbuild.1
```

```
True
v1.11.2-eksbuild.1
False
```

Versi dengan `True` di bawahnya adalah versi default yang digunakan saat add-on dibuat. Versi yang digunakan saat add-on dibuat mungkin bukan versi terbaru yang tersedia. Pada output sebelumnya, versi terbaru digunakan saat add-on dibuat.

3. Perbarui add-on ke versi dengan `True` yang dikembalikan dalam output dari langkah sebelumnya. Jika dikembalikan dalam output, Anda juga dapat memperbarui ke versi yang lebih baru.

```
aws eks update-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver
 --addon-version v1.11.4-eksbuild.1 \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --resolve-
conflicts PRESERVE
```

Opsi *`PRESERVE mempertahankan`* pengaturan kustom apa pun yang telah Anda tetapkan untuk add-on. Untuk informasi selengkapnya tentang opsi lain untuk setelan ini, lihat update-addon di Referensi Baris Perintah Amazon EKS. Untuk informasi selengkapnya tentang manajemen konfigurasi add-on Amazon EKS, lihat Manajemen lapangan Kubernetes.

Menghapus add-on Amazon EBS CSI

Anda memiliki dua opsi untuk menghapus add-on Amazon EKS.

- Pertahankan perangkat lunak add-on di cluster Anda — Opsi ini menghapus pengelolaan Amazon EKS dari pengaturan apa pun. Ini juga menghapus kemampuan Amazon EKS untuk memberi tahu Anda tentang pembaruan dan secara otomatis memperbarui add-on Amazon EKS setelah Anda memulai pembaruan. Namun, ini mempertahankan perangkat lunak add-on di cluster Anda. Opsi ini menjadikan add-on instalasi yang dikelola sendiri, bukan add-on Amazon EKS. Dengan opsi ini, tidak ada downtime untuk add-on. Perintah dalam prosedur ini menggunakan opsi ini.
- Hapus perangkat lunak add-on sepenuhnya dari klaster Anda — Kami menyarankan Anda menghapus add-on Amazon EKS dari klaster Anda hanya jika tidak ada sumber daya di klaster Anda yang bergantung padanya. Untuk melakukan opsi ini, hapus `--preserve` dari perintah yang Anda gunakan dalam prosedur ini.

Jika add-on memiliki akun IAM yang terkait dengannya, akun IAM tidak akan dihapus.

Anda dapat menggunakan`eksctl`,AWS Management Console, atau AWS CLI untuk menghapus add-on Amazon EBS CSI.

eksctl

Untuk menghapus add-on Amazon EBS CSI menggunakan **`eksctl`**

Ganti *`my-cluster`* dengan nama cluster Anda, lalu jalankan perintah berikut.

```
eksctl delete addon --cluster my-cluster --name aws-ebs-csi-driver --preserve
```

AWS Management Console

Untuk menghapus add-on Amazon EBS CSI menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pada panel navigasi sebelah kiri, pilih Klaster.
3. Pilih nama cluster yang ingin Anda hapus add-on Amazon EBS CSI.
4. Pilih tab Add-ons.
5. Pilih Driver Amazon EBS CSI.
6. Pilih Hapus.
7. Dalam Hapus: aws-ebs-csi-driver konfirmasi kotak dialog, lakukan hal berikut:
   a. Jika Anda ingin Amazon EKS berhenti mengelola pengaturan untuk add-on, pilih Pertahankan di klaster. Lakukan ini jika Anda ingin mempertahankan perangkat lunak add-on di cluster Anda. Ini agar Anda dapat mengelola semua pengaturan add-on sendiri.
   b. Masukkan **`aws-ebs-csi-driver`**.
   c. Pilih Hapus.

AWS CLI

Untuk menghapus add-on Amazon EBS CSI menggunakan AWS CLI

Ganti *`my-cluster`* dengan nama cluster Anda, lalu jalankan perintah berikut.

```
aws eks delete-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver --
preserve
```

Menyebarkan aplikasi sampel dan memverifikasi bahwa driver CSI berfungsi

Anda dapat menguji fungsionalitas driver CSI dengan aplikasi sampel. Topik ini menunjukkan satu contoh, tetapi Anda juga dapat melakukan hal berikut:

- Terapkan aplikasi sampel yang menggunakan snapshotter eksternal untuk membuat snapshot volume. Untuk informasi selengkapnya, lihat Snapshot Volume diGitHub.
- Menerapkan aplikasi sampel yang menggunakan pengubahan ukuran volume. Untuk informasi selengkapnya, lihat Mengubah Ukuran Volume diGitHub.

Prosedur ini menggunakan contoh penyediaan volume Dinamis dari GitHub repositori driver Amazon EBS Container Storage Interface (CSI) untuk menggunakan volume Amazon EBS yang disediakan secara dinamis.

1. Kloning GitHub repositori driver Amazon EBS Container Storage Interface (CSI) ke sistem lokal Anda.

   ```
   git clone https://github.com/kubernetes-sigs/aws-ebs-csi-driver.git
   ```

2. Navigasikan ke direktori contoh `dynamic-provisioning`.

   ```
   cd aws-ebs-csi-driver/examples/kubernetes/dynamic-provisioning/
   ```

3. (Opsional) `manifests/storageclass.yaml` File ini menyediakan volume gp2 Amazon EBS secara default. Untuk menggunakan gp3 volume sebagai gantinya, tambahkan `type: gp3` kemanifests/storageclass.yaml.

   ```
   echo "parameters:
     type: gp3" >> manifests/storageclass.yaml
   ```

4. Deploy kelas penyimpanan `ebs-sc`, klaim volume `ebs-claim` yang persisten, dan contoh aplikasi app dari direktori `manifests`.

```
kubectl apply -f manifests/
```

5. Jelaskan tentang kelas penyimpanan `ebs-sc`.

```
kubectl describe storageclass ebs-sc
```

Contoh output adalah sebagai berikut.

```
Name:            ebs-sc
IsDefaultClass:  No
Annotations:     kubectl.kubernetes.io/last-applied-
configuration={"apiVersion":"storage.k8s.io/v1","kind":"StorageClass","metadata":
{"annotations":{},"name":"ebs-
sc"},"provisioner":"ebs.csi.aws.com","volumeBindingMode":"WaitForFirstConsumer"}

Provisioner:           ebs.csi.aws.com
Parameters:            <none>
AllowVolumeExpansion:  <unset>
MountOptions:          <none>
ReclaimPolicy:         Delete
VolumeBindingMode:     WaitForFirstConsumer
Events:                <none>
```

> **Note**
>
> Kelas penyimpanan menggunakan mode pengikatan `WaitForFirstConsumer` volume. Ini berarti bahwa volume tidak disediakan secara dinamis sampai Pod membuat klaim volume persisten. Untuk informasi selengkapnya, lihat Mode Pengikatan Volume dalam Kubernetes dokumentasi.

6. Tonton Pods di namespace default. Setelah beberapa menit, `app` Pod status berubah menjadi`Running`.

```
kubectl get pods --watch
```

Masukkan `Ctrl+C` untuk kembali ke prompt shell.

7. Buat daftar volume yang persisten dalam namespace default. Cari volume persisten dengan klaim `default/ebs-claim`.

```
kubectl get pv
```

Contoh output adalah sebagai berikut.

```
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY
   STATUS   CLAIM               STORAGECLASS   REASON   AGE
pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a   4Gi        RWO            Delete
   Bound    default/ebs-claim   ebs-sc                  30s
```

8. Jelaskan volume persisten. Ganti pvc-*37717cd6-d0dc-11e9-b17f-06fad4858a5a* dengan nilai dari output pada langkah sebelumnya.

```
kubectl describe pv pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a
```

Contoh output adalah sebagai berikut.

```
Name:              pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a
Labels:            <none>
Annotations:       pv.kubernetes.io/provisioned-by: ebs.csi.aws.com
Finalizers:        [kubernetes.io/pv-protection external-attacher/ebs-csi-aws-com]
StorageClass:      ebs-sc
Status:            Bound
Claim:             default/ebs-claim
Reclaim Policy:    Delete
Access Modes:      RWO
VolumeMode:        Filesystem
Capacity:          4Gi
Node Affinity:
  Required Terms:
    Term 0:        topology.ebs.csi.aws.com/zone in [region-code]
Message:
Source:
    Type:              CSI (a Container Storage Interface (CSI) volume source)
    Driver:            ebs.csi.aws.com
    VolumeHandle:      vol-0d651e157c6d93445
    ReadOnly:          false
    VolumeAttributes:      storage.kubernetes.io/
csiProvisionerIdentity=1567792483192-8081-ebs.csi.aws.com
Events:                <none>
```

ID Volume Amazon EBS adalah nilai untuk `VolumeHandle` pada output sebelumnya.

9. Verifikasi bahwa Pod sedang menulis data ke volume.

```
kubectl exec -it app -- cat /data/out.txt
```

Contoh output adalah sebagai berikut.

```
Wed May 5 16:17:03 UTC 2021
Wed May 5 16:17:08 UTC 2021
Wed May 5 16:17:13 UTC 2021
Wed May 5 16:17:18 UTC 2021
[...]
```

10. Setelah selesai, hapus sumber daya untuk aplikasi contoh ini.

```
kubectl delete -f manifests/
```

Migrasi Amazon EBS CSI pertanyaan yang sering diajukan

> **Important**
>
> Jika Anda telah Pods menjalankan versi `1.22` atau klaster sebelumnya, maka Anda harus menginstal driver Amazon EBS CSI sebelum memperbarui cluster Anda ke versi `1.23` untuk menghindari gangguan layanan.

Fitur migrasi antarmuka penyimpanan kontainer (CSI) Amazon EBS memindahkan tanggung jawab untuk menangani operasi penyimpanan dari penyedia penyimpanan EBS in-tree Amazon EBS ke driver Amazon EBS CSI.

Apa itu driver CSI?

Driver CSI:

- ganti driver penyimpanan Kubernetes “in-tree” yang ada di kode sumber Kubernetes proyek.
- bekerja dengan penyedia penyimpanan, seperti Amazon EBS.

- menyediakan model plugin yang disederhanakan yang memudahkan penyedia penyimpanan seperti AWS merilis fitur dan mempertahankan dukungan tanpa bergantung pada siklus Kubernetes rilis.

Untuk informasi selengkapnya, lihat Pendahuluan dalam dokumentasi Kubernetes CSI.

Apa itu migrasi CSI?

Fitur Migrasi Kubernetes CSI memindahkan tanggung jawab untuk menangani operasi penyimpanan dari plugin penyimpanan in-tree yang ada, sepertikubernetes.io/aws-ebs, ke driver CSI yang sesuai. Objek yang adaStorageClass, PersistentVolume dan PersistentVolumeClaim (PVC) terus bekerja, selama driver CSI yang sesuai diinstal. Saat fitur diaktifkan:

- Beban kerja yang ada yang memanfaatkan PVC terus berfungsi seperti biasanya.
- Kubernetesmelewati kontrol semua operasi manajemen penyimpanan ke driver CSI.

Untuk informasi selengkapnya, lihat Kubernetes1.23: Pembaruan Status Migrasi Volume Kubernetes In-Tree ke CSI di blog. Kubernetes

Untuk membantu Anda bermigrasi dari plugin in-tree ke driver CSI, CSIMigrationAWS flag CSIMigration dan flag diaktifkan secara default pada versi 1.23 Amazon EKS dan kluster yang lebih baru. Bendera ini memungkinkan klaster Anda menerjemahkan API in-tree ke API CSI yang setara. Bendera ini disetel pada bidang Kubernetes kontrol yang dikelola oleh Amazon EKS dan dalam kubelet pengaturan yang dikonfigurasi di Amazon EKS yang dioptimalkan AMI. Jika Anda telah Pods menggunakan volume Amazon EBS di cluster Anda, Anda harus menginstal driver Amazon EBS CSI sebelum memperbarui cluster Anda ke versi. **1.23** Jika tidak, operasi volume seperti penyediaan dan pemasangan mungkin tidak berfungsi seperti yang diharapkan. Untuk informasi selengkapnya, lihat Driver CSI Amazon EBS.

> **Note**
>
> StorageClassPenyedia di dalam pohon diberi nama. kubernetes.io/aws-ebs StorageClassPenyedia Amazon EBS CSI diberi nama. ebs.csi.aws.com

Bisakah saya memasang **kubernetes.io/aws-ebs StorageClass** volume di versi **1.23** dan kluster yang lebih baru?

Ya, selama driver Amazon EBS CSI diinstal. Untuk versi yang baru dibuat `1.23` dan kluster yang lebih baru, sebaiknya instal driver Amazon EBS CSI sebagai bagian dari proses pembuatan klaster Anda. Kami juga merekomendasikan hanya menggunakan `StorageClasses` berdasarkan `ebs.csi.aws.com` penyedia.

Jika Anda telah memperbarui bidang kontrol cluster ke versi `1.23` dan belum memperbarui node Anda`1.23`, maka `CSIMigrationAWS` `kubelet` flag `CSIMigration` dan tidak diaktifkan. Dalam hal ini, driver in-tree digunakan untuk me-mount volume `kubernetes.io/aws-ebs` berbasis. Driver Amazon EBS CSI masih harus diinstal, untuk memastikan bahwa Pods menggunakan volume `kubernetes.io/aws-ebs` berbasis dapat dijadwalkan. Pengemudi juga diperlukan agar operasi volume lainnya berhasil.

Bisakah saya menyediakan **kubernetes.io/aws-ebs StorageClass** volume di Amazon EKS **1.23** dan kluster yang lebih baru?

Ya, selama driver Amazon EBS CSI diinstal.

Apakah **kubernetes.io/aws-ebs StorageClass** penyedia akan dihapus dari Amazon EKS?

`kubernetes.io/aws-ebs``StorageClass`Penyedia dan tipe `awsElasticBlockStore` volume tidak lagi didukung, tetapi tidak ada rencana untuk menghapusnya. Sumber daya ini diperlakukan sebagai bagian dari Kubernetes API.

Bagaimana cara menginstal driver Amazon EBS CSI?

Kami merekomendasikan menginstal add-on Amazon EBS CSI driver Amazon EKS. Ketika pembaruan diperlukan untuk add-on Amazon EKS, Anda memulai pembaruan dan Amazon EKS memperbarui add-on untuk Anda. Jika Anda ingin mengelola driver sendiri, Anda dapat menginstalnya menggunakan bagan Helm open source.

> **Important**
>
> Driver Amazon EBS Kubernetes in-tree berjalan di pesawat Kubernetes kontrol. Ini menggunakan izin IAM yang ditetapkan untuk menyediakan volume IAM role klaster Amazon

EKS Amazon EBS. Driver Amazon EBS CSI berjalan pada node. Pengemudi membutuhkan izin IAM untuk menyediakan volume. Untuk informasi selengkapnya, lihat Membuat peran IAM driver Amazon EBS CSI.

Bagaimana saya bisa memeriksa apakah driver Amazon EBS CSI diinstal di cluster saya?

Untuk menentukan apakah driver diinstal pada cluster Anda, jalankan perintah berikut:

```
kubectl get csidriver ebs.csi.aws.com
```

Untuk memeriksa apakah instalasi tersebut dikelola oleh Amazon EKS, jalankan perintah berikut:

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

Akankah Amazon EKS mencegah pembaruan cluster ke versi **1.23** jika saya belum menginstal driver Amazon EBS CSI?

Tidak.

Bagaimana jika saya lupa menginstal driver Amazon EBS CSI sebelum saya memperbarui cluster saya ke versi 1.23? Bisakah saya menginstal driver setelah memperbarui cluster saya?

Ya, tetapi operasi volume yang memerlukan driver Amazon EBS CSI akan gagal setelah pembaruan cluster Anda hingga driver diinstal.

Apa default yang **StorageClass** diterapkan dalam versi Amazon EKS yang baru dibuat **1.23** dan cluster yang lebih baru?

`StorageClass`Perilaku default tetap tidak berubah. Dengan setiap cluster baru, Amazon EKS menerapkan `StorageClass` nama `kubernetes.io/aws-ebs` berbasis`gp2`. Kami tidak berencana untuk menghapus ini `StorageClass` dari cluster yang baru dibuat. Pisahkan dari default cluster`StorageClass`, jika Anda membuat `ebs.csi.aws.com` basis `StorageClass` tanpa menentukan jenis volume, driver Amazon EBS CSI akan digunakan secara default. `gp3`

Akankah Amazon EKS membuat perubahan apa pun yang **StorageClasses** sudah ada di cluster saya yang ada saat saya memperbarui cluster saya ke versi**1.23**?

Tidak.

Bagaimana cara memigrasikan volume persisten dari **kubernetes.io/aws-ebsStorageClass** ke **ebs.csi.aws.com** menggunakan snapshot?

Untuk memigrasikan volume persisten, lihat Memigrasi kluster Amazon EKS dari gp2 ke gp3 EBS volume di blog. AWS

Bagaimana cara memodifikasi volume Amazon EBS menggunakan anotasi?

Dimulai dengan `aws-ebs-csi-driverv1.19.0-eksbuild.2`, Anda dapat memodifikasi volume Amazon EBS menggunakan anotasi dalam `PersistentVolumeClaim` s (PVC) mereka. Fitur modifikasi volume baru diimplementasikan sebagai sespan tambahan, yang disebut. `volumemodifier` Untuk informasi selengkapnya, lihat Menyederhanakan migrasi volume Amazon EBS dan modifikasi saat Kubernetes menggunakan Driver EBS CSI di blog. AWS

Apakah migrasi didukung untuk beban kerja Windows?

Ya. Jika Anda menginstal driver Amazon EBS CSI menggunakan bagan Helm open source, setel ke. `node.enableWindows true` Ini diatur secara default jika menginstal driver Amazon EBS CSI sebagai add-on Amazon EKS. Saat membuat`StorageClasses`, atur `fsType` ke sistem file Windows, sepertintfs. Operasi volume untuk beban kerja Windows kemudian dimigrasikan ke driver Amazon EBS CSI sama seperti untuk beban kerja Linux.

Driver Amazon EFS CSI

Amazon Elastic File System (Amazon EFS) menyediakan penyimpanan file tanpa server dan sepenuhnya elastis sehingga Anda dapat berbagi data file tanpa menyediakan atau mengelola kapasitas dan kinerja penyimpanan. Driver Amazon EFS Container Storage Interface (CSI) menyediakan antarmuka CSI yang memungkinkan Kubernetes cluster berjalan AWS untuk mengelola siklus hidup sistem file Amazon EFS. Topik ini menunjukkan cara menerapkan driver Amazon EFS CSI ke cluster Amazon EKS Anda.

Pertimbangan

- Driver Amazon EFS CSI tidak kompatibel dengan gambar kontainer berbasis Windows.

- Anda tidak dapat menggunakan penyediaan dinamis untuk volume persisten dengan node Fargate, tetapi Anda dapat menggunakan penyediaan statis.
- Penyediaan dinamis membutuhkan `1.2` atau lebih lambat dari pengemudi. Anda dapat menggunakan penyediaan statis untuk volume persisten menggunakan versi driver pada `1.1` versi kluster Amazon EKS yang didukung.
- Versi `1.3.2` atau yang lebih baru dari driver ini mendukung arsitektur Arm64, termasuk instans berbasis Amazon EC2 Graviton.
- Versi `1.4.2` atau yang lebih baru dari driver ini mendukung penggunaan FIPS untuk memasang sistem file.
- Catat kuota sumber daya untuk Amazon EFS. Misalnya, ada kuota 1000 titik akses yang dapat dibuat untuk setiap sistem file Amazon EFS. Untuk informasi selengkapnya, lihat Kuota sumber daya Amazon EFS yang tidak dapat Anda ubah.

Prasyarat

- Penyedia AWS Identity and Access Management (IAM) OpenID Connect (OIDC) yang sudah ada untuk cluster Anda. Untuk menentukan apakah Anda sudah memiliki satu, atau harus membuat satu, lihat Buat OIDC penyedia IAM untuk klaster Anda.
- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`,`apt-get`, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl` versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.

> **Note**
>
> PodRunning on AWS Fargate secara otomatis memasang sistem file Amazon EFS.

Membuat peran IAM

Driver Amazon EFS CSI memerlukan izin IAM untuk berinteraksi dengan sistem file Anda. Buat peran IAM dan lampirkan kebijakan AWS terkelola yang diperlukan padanya. Anda dapat menggunakan`eksctl`, AWS Management Console, atau AWS CLI.

> **Note**
>
> Langkah-langkah spesifik dalam prosedur ini ditulis untuk menggunakan driver sebagai add-on Amazon EKS. Untuk detail tentang instalasi yang dikelola sendiri, lihat Mengatur izin driver pada. GitHub

eksctl

Untuk membuat peran IAM driver Amazon EFS CSI Anda dengan **`eksctl`**

Jalankan perintah berikut untuk membuat peran IAM. Ganti *`my-cluster`* dengan nama cluster Anda dan *`AmazonEKS_EFS_CSI_DriverRole`* dengan nama untuk peran Anda.

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
eksctl create iamserviceaccount \
    --name efs-csi-controller-sa \
    --namespace kube-system \
    --cluster $cluster_name \
    --role-name $role_name \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEFSCSIDriverPolicy \
    --approve
TRUST_POLICY=$(aws iam get-role --role-name $role_name --query
 'Role.AssumeRolePolicyDocument' | \
    sed -e 's/efs-csi-controller-sa/efs-csi-*/' -e 's/StringEquals/StringLike/')
aws iam update-assume-role-policy --role-name $role_name --policy-document
 "$TRUST_POLICY"
```

AWS Management Console

Untuk membuat peran IAM driver Amazon EFS CSI Anda dengan AWS Management Console

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Pada halaman Peran, pilih Buat peran.
4. Pada halaman Pilih entitas tepercaya, lakukan hal berikut:
   a. Di bagian Jenis entitas tepercaya, pilih Identitas web.
   b. Untuk penyedia Identitas, pilih URL OpenID Connect penyedia untuk klaster Anda (seperti yang ditunjukkan di bawah Ikhtisar di Amazon EKS).
   c. Untuk Audiens, pilih `sts.amazonaws.com`.
   d. Pilih Berikutnya.
5. Pada halaman Tambahkan izin, lakukan hal berikut:
   a. Di dalam kotak Filter kebijakan, masukkan *AmazonEFSCSIDriverPolicy*.
   b. Pilih kotak centang di sebelah kiri yang *AmazonEFSCSIDriverPolicy* dikembalikan dalam pencarian.
   c. Pilih Berikutnya.
6. Pada halaman Nama, tinjau, dan buat, lakukan hal berikut:
   a. Untuk nama Peran, masukkan nama unik untuk peran Anda, seperti***AmazonEKS_EFS_CSI_DriverRole***.
   b. Di bawah Tambahkan tag (Opsional), tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tanda di IAM, lihat Menandai sumber daya IAM di Panduan Pengguna IAM.
   c. Pilih Buat peran.
7. Setelah peran dibuat, pilih peran di konsol untuk dibuka, dan kemudian diedit.
8. Pilih tab Trust relationship, lalu pilih Edit trust policy.
9. Temukan garis yang terlihat mirip dengan baris berikut:

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

Tambahkan baris berikut di atas baris sebelumnya. Ganti *region-code* dengan tempat Wilayah AWS cluster Anda berada. Ganti *EXAMPLED539D4633E53DE1B71EXAMPLE* dengan ID penyedia OIDC cluster Anda.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:efs-csi-*",
```

10. Ubah `Condition` operator dari `"StringEquals"` ke`"StringLike"`.
11. Pilih Perbarui kebijakan untuk menyelesaikan.

AWS CLI

Untuk membuat peran IAM driver Amazon EFS CSI Anda dengan AWS CLI

1. Lihat URL penyedia OIDC klaster Anda. Ganti *my-cluster* dengan nama klaster Anda. Jika output dari perintah adalah `None`, tinjau Prasyarat.

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

Contoh output adalah sebagai berikut.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Buat peran IAM yang memberikan tindakan. `AssumeRoleWithWebIdentity`

   a. Salin isi berikut ke file bernama *aws-efs-csi-driver-trust-policy*`.json`. Ganti *111122223333* dengan ID akun Anda. Ganti *EXAMPLED539D4633E53DE1B71EXAMPLE* dan *region-code* dengan nilai yang dikembalikan pada langkah sebelumnya. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringLike": {
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:efs-csi-*",
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

b. Buat peran. Anda dapat mengubah *AmazonEKS_EFS_CSI_DriverRole* ke nama yang berbeda, tetapi jika Anda melakukannya, pastikan untuk mengubahnya juga dalam langkah berikutnya.

```
aws iam create-role \
  --role-name AmazonEKS_EFS_CSI_DriverRole \
  --assume-role-policy-document file://"aws-efs-csi-driver-trust-
policy.json"
```

3. Lampirkan kebijakan AWS terkelola yang diperlukan ke peran dengan perintah berikut. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy \
  --role-name AmazonEKS_EFS_CSI_DriverRole
```

Menginstal driver Amazon EFS CSI

Kami menyarankan Anda menginstal driver Amazon EFS CSI melalui add-on Amazon EKS. Untuk menambahkan add-on Amazon EKS ke cluster Anda, lihatMembuat add-on. Untuk informasi selengkapnya tentang add-on, lihatAdd-on Amazon EKS. Jika Anda tidak dapat menggunakan add-

on Amazon EKS, kami mendorong Anda untuk mengirimkan masalah tentang mengapa Anda tidak dapat ke repositori peta jalan GitHub Containers.

Atau, jika Anda ingin instalasi yang dikelola sendiri dari driver Amazon EFS CSI, lihat Instalasi aktif. GitHub

Membuat sistem file Amazon EFS

Untuk membuat sistem file Amazon EFS, lihat Membuat sistem file Amazon EFS untuk Amazon EKSGitHub.

Menerapkan aplikasi sampel

Anda dapat menerapkan berbagai contoh aplikasi dan memodifikasinya sesuai kebutuhan. Untuk informasi selengkapnya, lihat Contoh diGitHub.

Driver CSI Amazon FSx for Lustre

Driver FSx for Lustre Container Storage Interface (CSI) menyediakan antarmuka CSI yang memungkinkan kluster Amazon EKS mengelola siklus hidup sistem file FSx for Lustre. Untuk informasi selengkapnya, lihat Panduan Pengguna FSx for Lustre.

Topik ini menunjukkan cara menerapkan driver FSx for Lustre CSI ke cluster Amazon EKS Anda dan memverifikasi bahwa itu berfungsi. Kami merekomendasikan menggunakan versi driver terbaru. Untuk versi yang tersedia, lihat Matriks Kompatibilitas Spesifikasi CSI pada GitHub.

> **Note**
>
> Pengemudi tidak didukung di Fargate.

Untuk deskripsi rinci tentang parameter yang tersedia dan contoh lengkap yang mendemonstrasikan fitur driver, lihat proyek driver FSx for Lustre Container Storage Interface (CSI) di. GitHub

Prasyarat

Anda harus memiliki:

- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut`**

`-d ' ' -f1`. Package manager seperti `yum` `apt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru dari file AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.

- Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.
- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl` versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.

Prosedur berikut membantu Anda membuat cluster pengujian sederhana dengan driver FSx for Lustre CSI sehingga Anda dapat melihat cara kerjanya. Kami tidak menyarankan menggunakan cluster pengujian untuk beban kerja produksi. Untuk tutorial ini, kami sarankan menggunakan*`example values`*, kecuali jika dicatat untuk menggantikannya. Anda dapat mengganti apa pun *`example value`* saat menyelesaikan langkah-langkah untuk klaster produksi Anda. Kami merekomendasikan untuk menyelesaikan semua langkah di terminal yang sama karena variabel diatur dan digunakan di seluruh langkah dan tidak akan ada di terminal yang berbeda.

Untuk menyebarkan driver FSx for Lustre CSI ke cluster Amazon EKS

1. Tetapkan beberapa variabel untuk digunakan dalam langkah-langkah yang tersisa. Ganti *`my-csi-fsx-cluster`* dengan nama cluster pengujian yang ingin Anda buat dan *`region-code`* dengan Wilayah AWS yang Anda inginkan untuk membuat cluster pengujian Anda.

   ```
   export cluster_name=my-csi-fsx-cluster
   export region_code=region-code
   ```

2. Buat cluster uji.

   ```
   eksctl create cluster \
     --name $cluster_name \
     --region $region_code \
     --with-oidc \
   ```

```
  --ssh-access \
  --ssh-public-key my-key
```

Penyediaan klaster memerlukan waktu beberapa menit. Selama pembuatan klaster, Anda akan melihat beberapa baris output. Baris terakhir output mirip dengan baris contoh berikut.

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

3. Buat akun Kubernetes layanan untuk driver dan lampirkan kebijakan `AmazonFSxFullAccess` AWS-managed ke akun layanan dengan perintah berikut. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
eksctl create iamserviceaccount \
  --name fsx-csi-controller-sa \
  --namespace kube-system \
  --cluster $cluster_name \
  --attach-policy-arn arn:aws:iam::aws:policy/AmazonFSxFullAccess \
  --approve \
  --role-name AmazonEKSFSxLustreCSIDriverFullAccess \
  --region $region_code
```

Anda akan melihat beberapa baris output saat akun layanan dibuat. Baris output terakhir mirip dengan yang berikut ini.

```
[#]  1 task: {
    2 sequential sub-tasks: {
        create IAM role for serviceaccount "kube-system/fsx-csi-controller-sa",
        create serviceaccount "kube-system/fsx-csi-controller-sa",
    } }
[#]  building iamserviceaccount stack "eksctl-my-csi-fsx-cluster-addon-
iamserviceaccount-kube-system-fsx-csi-controller-sa"
[#]  deploying stack "eksctl-my-csi-fsx-cluster-addon-iamserviceaccount-kube-
system-fsx-csi-controller-sa"
[#]  waiting for CloudFormation stack "eksctl-my-csi-fsx-cluster-addon-
iamserviceaccount-kube-system-fsx-csi-controller-sa"
[#]  created serviceaccount "kube-system/fsx-csi-controller-sa"
```

Perhatikan nama AWS CloudFormation tumpukan yang digunakan. Dalam contoh output sebelumnya, tumpukan diberi nama`eksctl-`*`my-csi-fsx-cluster`*`-addon-iamserviceaccount-kube-system-`*`fsx-csi-controller-sa`*.

4. Deploy driver dengan perintah berikut. Ganti *`release-X.XX`* dengan cabang yang Anda inginkan. Cabang master tidak didukung karena mungkin berisi fitur mendatang yang tidak kompatibel dengan versi stabil driver yang saat ini dirilis. Kami merekomendasikan menggunakan versi terbaru yang dirilis. Untuk daftar cabang aktif, lihat [aws-fsx-csi-driver](https://github.com/kubernetes-sigs/aws-fsx-csi-driver)di GitHub.

   > **Note**
   >
   > Anda dapat melihat konten yang diterapkan [aws-fsx-csi-driver](https://github.com/kubernetes-sigs/aws-fsx-csi-driver)diGitHub.

   ```
   kubectl apply -k "github.com/kubernetes-sigs/aws-fsx-csi-driver/deploy/kubernetes/
   overlays/stable/?ref=release-X.XX"
   ```

   Contoh output adalah sebagai berikut.

   ```
   serviceaccount/fsx-csi-controller-sa created
   serviceaccount/fsx-csi-node-sa created
   clusterrole.rbac.authorization.k8s.io/fsx-csi-external-provisioner-role created
   clusterrole.rbac.authorization.k8s.io/fsx-external-resizer-role created
   clusterrolebinding.rbac.authorization.k8s.io/fsx-csi-external-provisioner-binding
    created
   clusterrolebinding.rbac.authorization.k8s.io/fsx-csi-resizer-binding created
   deployment.apps/fsx-csi-controller created
   daemonset.apps/fsx-csi-node created
   csidriver.storage.k8s.io/fsx.csi.aws.com created
   ```

5. Perhatikan ARN untuk peran yang dibuat. Jika Anda tidak mencatatnya sebelumnya dan tidak memilikinya tersedia lagi di AWS CLI output, Anda dapat melakukan hal berikut untuk melihatnya di AWS Management Console.

   a. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.

   b. Pastikan konsol diatur ke peran IAM Wilayah AWS yang Anda buat, lalu pilih Tumpukan.

   c. Pilih tumpukan bernama `eksctl-`*`my-csi-fsx-cluster`*`-addon-iamserviceaccount-kube-system-`*`fsx-csi-controller-sa`*.

d. Pilih tab Outputs. Role1 ARN tercantum di halaman Output (1).

6. Patch penyebaran driver untuk menambahkan akun layanan yang Anda buat sebelumnya dengan perintah berikut. Ganti ARN dengan ARN yang Anda catat. Ganti *111122223333* dengan ID akun Anda. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
kubectl annotate serviceaccount -n kube-system fsx-csi-controller-sa \
  eks.amazonaws.com/role-
arn=arn:aws:iam::111122223333:role/AmazonEKSFSxLustreCSIDriverFullAccess --
overwrite=true
```

Contoh output adalah sebagai berikut.

```
serviceaccount/fsx-csi-controller-sa annotated
```

Untuk menerapkan kelas Kubernetes penyimpanan, klaim volume persisten, dan aplikasi sampel untuk memverifikasi bahwa driver CSI berfungsi

Prosedur ini menggunakan repositori driver FSx for Lustre Container Storage Interface (CSIGitHub) untuk menggunakan FSx for Lustre volume yang disediakan secara dinamis.

1. Perhatikan grup keamanan untuk klaster Anda. Anda dapat melihatnya di bagian AWS Management Console bawah Jaringan atau dengan menggunakan AWS CLI perintah berikut.

```
aws eks describe-cluster --name $cluster_name --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

2. Buat grup keamanan untuk sistem file Amazon FSx Anda sesuai dengan kriteria yang ditampilkan di Grup Keamanan VPC Amazon di Panduan Pengguna Amazon FSx for Lustre. Untuk VPC, pilih VPC cluster Anda seperti yang ditunjukkan di bawah bagian Networking. Untuk “grup keamanan yang terkait dengan klien Lustre Anda”, gunakan grup keamanan klaster Anda. Anda dapat meninggalkan aturan keluar sendiri untuk memungkinkan Semua lalu lintas.

3. Unduh manifes kelas penyimpanan dengan perintah berikut.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/
master/examples/kubernetes/dynamic_provisioning/specs/storageclass.yaml
```

4. Edit bagian parameter `storageclass.yaml` file. Ganti setiap *`example value`* dengan nilai-nilai Anda sendiri.

```
parameters:
  subnetId: subnet-0eabfaa81fb22bcaf
  securityGroupIds: sg-068000ccf82dfba88
  deploymentType: PERSISTENT_1
  automaticBackupRetentionDays: "1"
  dailyAutomaticBackupStartTime: "00:00"
  copyTagsToBackups: "true"
  perUnitStorageThroughput: "200"
  dataCompressionType: "NONE"
  weeklyMaintenanceStartTime: "7:09:00"
  fileSystemTypeVersion: "2.12"
```

   - **`subnetId`**— ID subnet tempat sistem file Amazon FSx for Lustre harus dibuat. Amazon FSx for Lustre tidak didukung di semua Availability Zone. Buka konsol Amazon FSx for Lustre di https://console.aws.amazon.com/fsx/ untuk mengonfirmasi bahwa subnet yang ingin Anda gunakan adalah di Availability Zone yang didukung. Subnet dapat menyertakan node Anda, atau dapat berupa subnet atau VPC yang berbeda:
     - Anda dapat memeriksa subnet node di AWS Management Console dengan memilih grup node di bawah bagian Compute.
     - Jika subnet yang Anda tentukan bukan subnet yang sama dengan yang Anda miliki node, maka VPC Anda harus terhubung, dan Anda harus memastikan bahwa Anda memiliki port yang diperlukan terbuka di grup keamanan Anda.
   - **`securityGroupIds`**— ID grup keamanan yang Anda buat untuk sistem file.
   - **`deploymentType`**(opsional) — Jenis penyebaran sistem file. Nilai yang valid adalah `SCRATCH_1`, `SCRATCH_2`, `PERSISTENT_1`, dan `PERSISTENT_2`. Untuk informasi selengkapnya tentang jenis deployment, lihat Buat sistem file Amazon FSx for Lustre.
   - parameter lain (opsional) — Untuk informasi tentang parameter lainnya, lihat StorageClassMengeditGitHub.

5. Buat manifes kelas penyimpanan.

```
kubectl apply -f storageclass.yaml
```

   Contoh output adalah sebagai berikut.

```
storageclass.storage.k8s.io/fsx-sc created
```

6. Unduh manifes klaim volume persisten.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/
master/examples/kubernetes/dynamic_provisioning/specs/claim.yaml
```

7. (Opsional) Edit file `claim.yaml`. Ubah *1200Gi* ke salah satu nilai kenaikan berikut, berdasarkan kebutuhan penyimpanan Anda dan `deploymentType` yang Anda pilih pada langkah sebelumnya.

```
storage: 1200Gi
```

   - SCRATCH_2dan PERSISTENT —**1.2 TiB**,**2.4 TiB**, atau kenaikan 2,4 TiB lebih dari 2,4 TiB.
   - SCRATCH_1—**1.2 TiB**,, **2.4 TiB3.6 TiB**, atau kenaikan 3,6 TiB lebih dari 3,6 TiB.

8. Buat klaim volume persisten.

```
kubectl apply -f claim.yaml
```

   Contoh output adalah sebagai berikut.

```
persistentvolumeclaim/fsx-claim created
```

9. Konfirmasikan bahwa sistem file disediakan.

```
kubectl describe pvc
```

   Contoh output adalah sebagai berikut.

```
Name:          fsx-claim
Namespace:     default
StorageClass:  fsx-sc
Status:        Bound
[...]
```

> **Note**
>
> `Status` mungkin ditampilkan sebagai `Pending` selama 5-10 menit, sebelum berubah ke `Bound`. Jangan lanjutkan ke langkah berikutnya sampai `Status` adalah `Bound`. Jika `Status` ditampilkan `Pending` selama lebih dari 10 menit, gunakan pesan peringatan di `Events` sebagai referensi untuk mengatasi masalah apa pun.

10. Deploy aplikasi sampel.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/master/examples/kubernetes/dynamic_provisioning/specs/pod.yaml
```

11. Verifikasi bahwa aplikasi sampel sedang dijalankan.

```
kubectl get pods
```

Contoh output adalah sebagai berikut.

```
NAME      READY   STATUS              RESTARTS   AGE
fsx-app   1/1     Running             0          8s
```

12. Verifikasi bahwa sistem file dipasang dengan benar oleh aplikasi.

```
kubectl exec -ti fsx-app -- df -h
```

Contoh output adalah sebagai berikut.

```
Filesystem                  Size  Used Avail Use% Mounted on
overlay                      80G  4.0G   77G   5% /
tmpfs                        64M     0   64M   0% /dev
tmpfs                       3.8G     0  3.8G   0% /sys/fs/cgroup
192.0.2.0@tcp:/abcdef01     1.1T  7.8M  1.1T   1% /data
/dev/nvme0n1p1               80G  4.0G   77G   5% /etc/hosts
shm                          64M     0   64M   0% /dev/shm
tmpfs                       6.9G   12K  6.9G   1% /run/secrets/kubernetes.io/
serviceaccount
tmpfs                       3.8G     0  3.8G   0% /proc/acpi
tmpfs                       3.8G     0  3.8G   0% /sys/firmware
```

13. Verifikasi bahwa data telah ditulis ke sistem file FSx for Lustre oleh aplikasi sampel.

```
kubectl exec -it fsx-app -- ls /data
```

Contoh output adalah sebagai berikut.

```
out.txt
```

Contoh keluaran ini menunjukkan bahwa aplikasi sampel berhasil menulis `out.txt` file ke sistem file.

> **Note**
>
> Sebelum menghapus cluster, pastikan untuk menghapus sistem file FSx for Lustre. Untuk informasi selengkapnya, lihat Membersihkan sumber daya di Panduan Pengguna FSx for Lustre.

Amazon FSx untuk driver NetApp ONTAP CSI

NetApp'sAstra Tridentmenyediakan orkestrasi penyimpanan dinamis menggunakan driver yang sesuai dengan Container Storage Interface (CSI). Hal ini memungkinkan kluster Amazon EKS untuk mengelola siklus hidup volume persisten (PV) yang didukung oleh Amazon FSx untuk sistem file ONTAP. NetApp Untuk memulai, lihat Menggunakan Astra Trident dengan Amazon FSx untuk NetApp ONTAP dalam dokumentasi. Astra Trident

Amazon FSx untuk NetApp ONTAP adalah layanan penyimpanan yang memungkinkan Anda meluncurkan dan menjalankan sistem ONTAP file yang dikelola sepenuhnya di cloud. ONTAPadalah teknologi sistem NetApp's file yang menyediakan serangkaian akses data dan kemampuan manajemen data yang diadopsi secara luas. FSx untuk ONTAP menyediakan fitur, kinerja, dan API sistem NetApp file lokal dengan kelincahan, skalabilitas, dan kesederhanaan layanan yang dikelola sepenuhnya. AWS Untuk informasi selengkapnya, lihat Panduan Pengguna FSx for ONTAP.

Amazon FSx untuk driver OpenZFS CSI

Amazon FSx for OpenZFS adalah layanan penyimpanan file yang dikelola sepenuhnya yang memudahkan pemindahan data AWS dari ZFS lokal atau server file berbasis Linux lainnya.

Anda dapat melakukan ini tanpa mengubah kode aplikasi Anda atau bagaimana Anda mengelola data. Ini menawarkan penyimpanan file yang sangat andal, terukur, efisien, dan kaya fitur yang dibangun di atas sistem file OpenZFS open-source. Ini menggabungkan kemampuan ini dengan kelincahan, skalabilitas, dan kesederhanaan layanan yang dikelola AWS sepenuhnya. Untuk informasi selengkapnya, lihat Panduan Pengguna Amazon FSx for OpenZFS.

Driver Amazon FSx for OpenZFS Container Storage Interface (CSI) menyediakan antarmuka CSI yang memungkinkan klaster Amazon EKS mengelola siklus hidup Amazon FSx untuk volume OpenZFS. Untuk menerapkan driver Amazon FSx untuk OpenZFS CSI ke klaster Amazon EKS Anda, lihat terus. `aws-fsx-openzfs-csi-driver`GitHub

Driver CSI CSI Cache File Amazon

Amazon File Cache adalah cache berkecepatan tinggiAWS yang dikelola sepenuhnya yang digunakan untuk memproses data file, di mana pun data disimpan. Amazon File Cache secara otomatis memuat data ke dalam cache saat diakses untuk pertama kalinya dan merilis data saat tidak digunakan. Untuk informasi selengkapnya, lihat Panduan Pengguna Amazon File Cache.

Driver Amazon File Cache Container Storage Interface (CSI) menyediakan antarmuka CSI yang memungkinkan klaster Amazon EKS untuk mengelola siklus hidup cache file Amazon EKS untuk mengelola siklus hidup cache Amazon EKS. Untuk menerapkan driver CSI CSI Amazon File Cache ke klaster Amazon EKS Anda, lihat `aws-file-cache-csi-driver`di GitHub.

Mountpointuntuk driver Amazon S3 CSI

Dengan driver Mountpoint untuk Amazon S3 Container Storage Interface (CSI), Kubernetes aplikasi Anda dapat mengakses objek Amazon S3 melalui antarmuka sistem file, mencapai throughput agregat tinggi tanpa mengubah kode aplikasi apa pun. Dibangun Mountpointuntuk Amazon S3, driver CSI menghadirkan bucket Amazon S3 sebagai volume yang dapat diakses oleh kontainer di Amazon EKS dan cluster yang dikelola sendiri. Kubernetes Topik ini menunjukkan cara menerapkan driver CSI Amazon S3 Mountpoint untuk kluster Amazon EKS Anda.

Pertimbangan

- Driver Mountpoint untuk Amazon S3 CSI saat ini tidak kompatibel dengan gambar kontainer berbasis Windows.
- Driver Mountpoint untuk Amazon S3 CSI tidak mendukung. AWS Fargate Namun, kontainer yang berjalan di Amazon EC2 (baik dengan Amazon EKS atau Kubernetes instalasi khusus) didukung.

- Driver Mountpoint untuk Amazon S3 CSI hanya mendukung penyediaan statis. Penyediaan dinamis, atau pembuatan bucket baru, tidak didukung.

> **Note**
>
> Penyediaan statis mengacu pada penggunaan bucket Amazon S3 yang sudah ada yang ditentukan sebagai `bucketName` dalam objek`volumeHandle`. `PersistentVolume` Untuk informasi selengkapnya, lihat Penyediaan Statis aktif. GitHub

- Volume yang dipasang dengan driver CSI Amazon S3 Mountpoint untuk Amazon tidak mendukung semua fitur sistem file POSIX. Untuk detail tentang perilaku sistem file, lihat Mountpoint perilaku sistem file Amazon S3 aktif. GitHub

Prasyarat

- Penyedia AWS Identity and Access Management (IAM) OpenID Connect (OIDC) yang sudah ada untuk cluster Anda. Untuk menentukan apakah Anda sudah memiliki satu, atau harus membuat satu, lihat Buat OIDC penyedia IAM untuk klaster Anda.
- Versi 2.12.3 atau yang lebih baru AWS CLI diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell
- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl` versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.

Membuat kebijakan IAM

Driver Mountpoint untuk Amazon S3 CSI memerlukan izin Amazon S3 untuk berinteraksi dengan sistem file Anda. Bagian ini menunjukkan cara membuat kebijakan IAM yang memberikan izin yang diperlukan.

Contoh kebijakan berikut mengikuti rekomendasi izin IAM untukMountpoint. Atau, Anda dapat menggunakan kebijakan AWS terkelola AmazonS3FullAccess, tetapi kebijakan terkelola ini memberikan lebih banyak izin daripada yang diperlukan. Mountpoint

Untuk informasi selengkapnya tentang izin yang disarankanMountpoint, lihat MountpointIzin IAM di. GitHub

Membuat kebijakan IAM dengan konsol IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi di sebelah kiri, pilih Kebijakan.
3. Pada halaman Kebijakan, pilih Buat kebijakan.
4. Untuk editor Kebijakan, pilih JSON.
5. Di bawah Editor kebijakan, salin dan tempel yang berikut ini:

> **Important**
>
> Ganti *DOC-EXAMPLE-BUCKET1* dengan nama bucket Amazon S3 Anda sendiri.

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Sid": "MountpointFullBucketAccess",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
            ]
        },
        {
            "Sid": "MountpointFullObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:AbortMultipartUpload",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
```

```
            ]
        }
    ]
}
```

Bucket direktori, diperkenalkan dengan kelas penyimpanan Amazon S3 Express One Zone, menggunakan mekanisme otentikasi yang berbeda dari bucket tujuan umum. Alih-alih menggunakan `s3:*` tindakan, Anda harus menggunakan `s3express:CreateSession` tindakan. Untuk informasi tentang bucket direktori, lihat Bucket direktori di Panduan Pengguna Amazon S3.

Di bawah ini adalah contoh kebijakan hak istimewa paling tidak yang akan Anda gunakan untuk bucket direktori.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3express:CreateSession",
            "Resource": "arn:aws:s3express:aws-region:111122223333:bucket/DOC-
EXAMPLE-BUCKET1--az_id--x-s3"
        }
    ]
}
```

6. Pilih Berikutnya.
7. Pada halaman Tinjau dan buat, beri nama kebijakan Anda. Contoh panduan ini menggunakan nama. `AmazonS3CSIDriverPolicy`
8. Pilih Buat kebijakan.

Membuat peran IAM

Driver Mountpoint untuk Amazon S3 CSI memerlukan izin Amazon S3 untuk berinteraksi dengan sistem file Anda. Bagian ini menunjukkan cara membuat peran IAM untuk mendelegasikan izin ini. Untuk membuat peran ini, Anda dapat menggunakan`eksctl`, konsol IAM, atau. AWS CLI

> **Note**
>
> Kebijakan IAM `AmazonS3CSIDriverPolicy` dibuat di bagian sebelumnya.

eksctl

MountpointUntuk membuat peran IAM driver Amazon S3 CSI Anda dengan **`eksctl`**

Untuk membuat peran IAM dan akun Kubernetes layanan, jalankan perintah berikut. Perintah ini juga melampirkan kebijakan `AmazonS3CSIDriverPolicy` IAM ke peran, membubuhi keterangan akun Kubernetes layanan (`s3-csi-controller-sa`) dengan Nama Sumber Daya Amazon (ARN) peran IAM, dan menambahkan nama akun Kubernetes layanan ke kebijakan kepercayaan untuk peran IAM.

```
CLUSTER_NAME=my-cluster
REGION=region-code
ROLE_NAME=AmazonEKS_S3_CSI_DriverRole
POLICY_ARN=AmazonEKS_S3_CSI_DriverRole_ARN
eksctl create iamserviceaccount \
    --name s3-csi-driver-sa \
    --namespace kube-system \
    --cluster $CLUSTER_NAME \
    --attach-policy-arn $POLICY_ARN \
    --approve \
    --role-name $ROLE_NAME \
    --region $REGION \
    --role-only
```

IAM console

MountpointUntuk membuat peran IAM driver Amazon S3 CSI Anda dengan AWS Management Console

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Pada halaman Peran, pilih Buat peran.
4. Pada halaman Pilih entitas tepercaya, lakukan hal berikut:

    a. Di bagian Jenis entitas tepercaya, pilih Identitas web.

b. Untuk penyedia Identitas, pilih URL OpenID Connect penyedia untuk klaster Anda (seperti yang ditunjukkan di bawah Ikhtisar di Amazon EKS).

   Jika tidak ada URL yang ditampilkan, tinjau bagian Prasyarat.

c. Untuk Audiens, pilih `sts.amazonaws.com`.

d. Pilih Berikutnya.

5. Pada halaman Tambahkan izin, lakukan hal berikut:

   a. Di dalam kotak Filter kebijakan, masukkan **`AmazonS3CSIDriverPolicy`**.

   > **Note**
   >
   > Kebijakan ini dibuat di bagian sebelumnya.

   b. Pilih kotak centang di sebelah kiri `AmazonS3CSIDriverPolicy` hasil yang dikembalikan dalam pencarian.

   c. Pilih Berikutnya.

6. Pada halaman Nama, tinjau, dan buat, lakukan hal berikut:

   a. Untuk nama Peran, masukkan nama unik untuk peran Anda, seperti**`AmazonEKS_S3_CSI_DriverRole`**.

   b. Di bawah Tambahkan tag (Opsional), tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tanda di IAM, lihat Menandai sumber daya IAM di Panduan Pengguna IAM.

   c. Pilih Buat peran.

7. Setelah peran dibuat, pilih peran di konsol untuk dibuka, dan kemudian diedit.

8. Pilih tab Trust relationship, lalu pilih Edit trust policy.

9. Temukan baris yang serupa dengan yang berikut ini:

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

   Tambahkan koma ke akhir baris sebelumnya, lalu tambahkan baris berikut setelahnya. Ganti *`region-code`* dengan tempat Wilayah AWS cluster Anda berada. Ganti *`EXAMPLED539D4633E53DE1B71EXAMPLE`* dengan ID penyedia OIDC cluster Anda.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:s3-csi-*"
```

10. Ubah `Condition` operator dari `"StringEquals"` ke`"StringLike"`.
11. Pilih Perbarui kebijakan untuk menyelesaikan.

AWS CLI

MountpointUntuk membuat peran IAM driver Amazon S3 CSI Anda dengan AWS CLI

1. Lihat URL penyedia OIDC untuk klaster Anda. Ganti *my-cluster* dengan nama klaster Anda. Jika output dari perintah adalah None, tinjau Prasyarat.

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

Contoh output adalah sebagai berikut.

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Buat peran IAM, berikan tindakan kepada akun Kubernetes layanan. `AssumeRoleWithWebIdentity`

   a. Salin isi berikut ke file bernama *aws-s3-csi-driver-trust-policy*`.json`. Ganti *111122223333* dengan ID akun Anda. Ganti *EXAMPLED539D4633E53DE1B71EXAMPLE* dan *region-code* dengan nilai yang dikembalikan pada langkah sebelumnya.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
```

```
        "StringLike": {
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:s3-csi-*",
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

b. Buat peran. Anda dapat mengubah *AmazonEKS_S3_CSI_DriverRole* ke nama yang berbeda, tetapi jika Anda melakukannya, pastikan untuk mengubahnya juga dalam langkah berikutnya.

```
aws iam create-role \
  --role-name AmazonEKS_S3_CSI_DriverRole \
  --assume-role-policy-document file://"aws-s3-csi-driver-trust-policy.json"
```

3. Lampirkan kebijakan IAM yang dibuat sebelumnya ke peran dengan perintah berikut.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonS3CSIDriverPolicy \
  --role-name AmazonEKS_S3_CSI_DriverRole
```

> **Note**
>
> Kebijakan IAM `AmazonS3CSIDriverPolicy` dibuat di bagian sebelumnya.

4. Lewati langkah ini jika Anda menginstal driver sebagai add-on Amazon EKS. Untuk instalasi driver yang dikelola sendiri, buat akun Kubernetes layanan yang dianotasi dengan ARN dari peran IAM yang Anda buat.

a. Simpan konten berikut ini ke file bernama *mountpoint-s3-service-account*`.yaml`. Ganti *111122223333* dengan ID akun Anda.

```
---
apiVersion: v1
kind: ServiceAccount
metadata:
```

```
  labels:
    app.kubernetes.io/name: aws-mountpoint-s3-csi-driver
  name: mountpoint-s3-csi-controller-sa
  namespace: kube-system
  annotations:
    eks.amazonaws.com/role-arn:
arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
```

b. Buat akun Kubernetes layanan di cluster Anda. Akun Kubernetes layanan (`mountpoint-s3-csi-controller-sa`) dianotasi dengan peran IAM yang Anda buat bernama. *AmazonEKS_S3_CSI_DriverRole*

```
kubectl apply -f mountpoint-s3-service-account.yaml
```

> **Note**
>
> Ketika Anda menyebarkan plugin dalam prosedur ini, itu membuat dan dikonfigurasi untuk menggunakan akun layanan bernama`s3-csi-driver-sa`.

Menginstal driver Mountpoint untuk Amazon S3 CSI

Anda dapat menginstal driver Mountpoint untuk Amazon S3 CSI melalui add-on Amazon EKS. Anda dapat menggunakan`eksctl`, AWS Management Console, atau AWS CLI untuk menambahkan add-on ke cluster Anda.

Anda dapat menginstal driver Mountpoint Amazon S3 CSI secara opsional sebagai instalasi yang dikelola sendiri. Untuk petunjuk cara melakukan instalasi yang dikelola sendiri, lihat Instalasi aktif. GitHub

eksctl

Untuk menambahkan add-on Amazon S3 CSI menggunakan **`eksctl`**

Jalankan perintah berikut. Ganti *my-cluster* dengan nama klaster Anda, *111122223333* dengan ID akun Anda, dan *AmazonEKS_S3_CSI_DriverRole* dengan nama peran IAM yang dibuat sebelumnya.

```
eksctl create addon --name aws-mountpoint-s3-csi-driver --cluster my-cluster --
service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
 --force
```

Jika Anda menghapus `--force` opsi dan salah satu pengaturan add-on Amazon EKS bertentangan dengan pengaturan yang ada, pembaruan add-on Amazon EKS gagal, dan Anda menerima pesan kesalahan untuk membantu Anda menyelesaikan konflik. Sebelum menentukan opsi ini, pastikan add-on Amazon EKS tidak mengelola pengaturan yang perlu Anda kelola, karena pengaturan tersebut ditimpa dengan opsi ini. Untuk informasi selengkapnya tentang opsi lain untuk setelan ini, lihat Addons dalam `eksctl` dokumentasi. Untuk informasi selengkapnya tentang manajemen Kubernetes lapangan Amazon EKS, lihat Manajemen lapangan Kubernetes.

AWS Management Console

Untuk menambahkan add-on Mountpoint untuk Amazon S3 CSI menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pada panel navigasi sebelah kiri, pilih Klaster.
3. Pilih nama cluster yang ingin Anda konfigurasikan untuk Mountpoint add-on Amazon S3 CSI.
4. Pilih tab Add-ons.
5. Pilih Get more add-ons
6. Pada halaman Pilih add-on, lakukan hal berikut:
   a. Di bagian Amazon EKS-Addons, pilih kotak centang untuk Amazon Mountpoint S3 CSI Driver.
   b. Pilih Berikutnya.
7. Pada halaman Configure selected add-ons settings, lakukan hal berikut:
   a. Pilih Versi yang ingin Anda gunakan.
   b. Untuk peran Select IAM, pilih nama peran IAM yang Anda lampirkan Mountpoint untuk kebijakan IAM driver Amazon S3 CSI.
   c. (Opsional) Anda dapat memperluas Pengaturan konfigurasi opsional. Jika Anda memilih Override untuk metode resolusi Konflik, satu atau beberapa pengaturan untuk add-on yang sudah ada dapat ditimpa dengan pengaturan add-on Amazon EKS. Jika Anda tidak mengaktifkan opsi ini dan ada konflik dengan pengaturan yang ada, operasi gagal. Anda dapat menggunakan pesan kesalahan yang dihasilkan untuk memecahkan masalah

konflik. Sebelum memilih opsi ini, pastikan add-on Amazon EKS tidak mengelola pengaturan yang perlu Anda kelola sendiri.

d. Pilih Berikutnya.

8. Pada halaman Review dan add, pilih Create. Setelah penginstalan add-on selesai, Anda melihat add-on yang diinstal.

AWS CLI

Untuk menambahkan add-on Mountpoint untuk Amazon S3 CSI menggunakan AWS CLI

Jalankan perintah berikut. Ganti *my-cluster* dengan nama klaster Anda, *111122223333* dengan ID akun Anda, dan *AmazonEKS_S3_CSI_DriverRole* dengan nama peran yang dibuat sebelumnya.

```
aws eks create-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-driver \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
```

Mengkonfigurasi Mountpoint untuk Amazon S3

Dalam kebanyakan kasus, Anda dapat mengonfigurasi Mountpoint untuk Amazon S3 hanya dengan nama bucket. Untuk petunjuk tentang mengonfigurasi Mountpoint Amazon S3, lihat Mountpoint Mengonfigurasi Amazon S3 aktif. GitHub

Menerapkan aplikasi sampel

Anda dapat menerapkan penyediaan statis ke driver di bucket Amazon S3 yang ada. Untuk informasi selengkapnya, lihat Penyediaan statis aktif. GitHub

Menghapus Mountpoint untuk Driver Amazon S3 CSI

Anda memiliki dua opsi untuk menghapus add-on Amazon EKS.

- Pertahankan perangkat lunak add-on di cluster Anda — Opsi ini menghapus pengelolaan Amazon EKS dari pengaturan apa pun. Ini juga menghapus kemampuan Amazon EKS untuk memberi tahu Anda tentang pembaruan dan secara otomatis memperbarui add-on Amazon EKS setelah Anda memulai pembaruan. Namun, ini mempertahankan perangkat lunak add-on di cluster Anda. Opsi

ini menjadikan add-on instalasi yang dikelola sendiri, bukan add-on Amazon EKS. Dengan opsi ini, tidak ada downtime untuk add-on. Perintah dalam prosedur ini menggunakan opsi ini.

- Hapus perangkat lunak add-on sepenuhnya dari klaster Anda — Kami menyarankan Anda menghapus add-on Amazon EKS dari klaster Anda hanya jika tidak ada sumber daya di klaster Anda yang bergantung padanya. Untuk melakukan opsi ini, hapus `--preserve` dari perintah yang Anda gunakan dalam prosedur ini.

Jika add-on memiliki akun IAM yang terkait dengannya, akun IAM tidak akan dihapus.

Anda dapat menggunakan`eksctl`, AWS Management Console, atau AWS CLI untuk menghapus add-on Amazon S3 CSI.

eksctl

Untuk menghapus add-on Amazon S3 CSI menggunakan **`eksctl`**

Ganti *`my-cluster`* dengan nama cluster Anda, lalu jalankan perintah berikut.

```
eksctl delete addon --cluster my-cluster --name aws-mountpoint-s3-csi-driver --
preserve
```

AWS Management Console

Untuk menghapus add-on Amazon S3 CSI menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pada panel navigasi sebelah kiri, pilih Klaster.
3. Pilih nama cluster yang ingin Anda hapus add-on Amazon EBS CSI.
4. Pilih tab Add-ons.
5. Pilih Mountpointuntuk Driver Amazon S3 CSI.
6. Pilih Hapus.
7. Dalam Hapus: Konfirmasi aws-mountpoint-s 3-csi-driver kotak dialog, lakukan hal berikut:

    a. Jika Anda ingin Amazon EKS berhenti mengelola pengaturan untuk add-on, pilih Pertahankan di klaster. Lakukan ini jika Anda ingin mempertahankan perangkat lunak add-on di cluster Anda. Ini agar Anda dapat mengelola semua pengaturan add-on sendiri.

b. Masukkan **aws-mountpoint-s3-csi-driver**.

c. Pilih Hapus.

AWS CLI

Untuk menghapus add-on Amazon S3 CSI menggunakan AWS CLI

Ganti *my-cluster* dengan nama cluster Anda, lalu jalankan perintah berikut.

```
aws eks delete-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-driver --preserve
```

Pengontrol snapshot CSI

Pengontrol snapshot Container Storage Interface (CSI) memungkinkan penggunaan fungsionalitas snapshotting pada driver CSI yang kompatibel, seperti driver Amazon EBS CSI.

Berikut adalah beberapa hal yang perlu dipertimbangkan saat menggunakan pengontrol snapshot CSI.

- Pengontrol snapshot harus diinstal bersama driver CSI dengan fungsionalitas snapshotting. Driver Amazon EBS CSI mendukung pembuatan snapshot Amazon EBS dari volume terkelola Amazon EBS CSI. Untuk instruksi instalasi, lihat Driver CSI Amazon EBS.
- Kubernetestidak mendukung snapshot volume yang disajikan melalui migrasi CSI, seperti volume Amazon EBS menggunakan penyedia with. `StorageClass` `kubernetes.io/aws-ebs` Volume harus dibuat dengan `StorageClass` yang mereferensikan penyedia driver CSI,. `ebs.csi.aws.com` Untuk informasi selengkapnya tentang migrasi CSI, lihatMigrasi Amazon EBS CSI pertanyaan yang sering diajukan.

Kami menyarankan Anda menginstal pengontrol snapshot CSI melalui add-on terkelola Amazon EKS. Untuk menambahkan add-on Amazon EKS ke cluster Anda, lihatMembuat add-on. Untuk informasi selengkapnya tentang add-on, lihatAdd-on Amazon EKS.

Atau, jika Anda menginginkan penginstalan yang dikelola sendiri dari pengontrol snapshot Amazon EBS CSI, lihat Penggunaan di upstream aktif. Kubernetes `external-snapshotter` GitHub

Jaringan Amazon EKS

Cluster Amazon EKS Anda dibuat dalam VPC. Jaringan pod disediakan oleh plugin Amazon VPC Container Network Interface (CNI). Bab ini mencakup topik-topik berikut untuk mempelajari lebih lanjut tentang jaringan untuk cluster Anda.

Topik

Persyaratan dan pertimbangan Amazon EKS VPC dan subnet

Saat membuat cluster, Anda menentukan VPC dan setidaknya dua subnet yang berada di Availability Zone yang berbeda. Topik ini memberikan gambaran umum tentang persyaratan dan pertimbangan khusus Amazon EKS untuk VPC dan subnet yang Anda gunakan dengan klaster Anda. Jika Anda tidak memiliki VPC untuk digunakan dengan Amazon EKS, Anda dapat membuatnya menggunakan template yang disediakan AWS CloudFormation Amazon EKS. Jika Anda membuat kluster lokal atau diperluas AWS Outposts, lihat Persyaratan dan pertimbangan VPC klaster lokal dan subnet Amazon EKS alih-alih topik ini.

Persyaratan dan pertimbangan VPC

Saat Anda membuat klaster, VPC yang Anda tentukan harus memenuhi persyaratan dan pertimbangan berikut:

- VPC harus memiliki cukup banyak alamat IP yang tersedia untuk cluster, node apa pun, dan Kubernetes sumber daya lain yang ingin Anda buat. Jika VPC yang ingin Anda gunakan tidak memiliki jumlah alamat IP yang cukup, cobalah untuk menambah jumlah alamat IP yang tersedia.

  Anda dapat melakukan ini dengan memperbarui konfigurasi cluster untuk mengubah subnet dan grup keamanan yang digunakan cluster. Anda dapat memperbarui dari AWS Management Console, versi terbaru dari AWS CLI, AWS CloudFormation, dan `eksctl` versi `v0.164.0-rc.0`

atau yang lebih baru. Anda mungkin perlu melakukan ini untuk menyediakan subnet dengan lebih banyak alamat IP yang tersedia untuk berhasil meningkatkan versi cluster.

> **Important**
>
> Semua subnet yang Anda tambahkan harus berada dalam kumpulan AZ yang sama seperti yang disediakan semula saat Anda membuat cluster. Subnet baru harus memenuhi semua persyaratan lainnya, misalnya mereka harus memiliki alamat IP yang memadai. Misalnya, asumsikan bahwa Anda membuat cluster dan menentukan empat subnet. Dalam urutan yang Anda tentukan, subnet pertama berada di `us-west-2a` Availability Zone, subnet kedua dan ketiga berada di `us-west-2b` Availability Zone, dan subnet keempat berada di Availability Zone. `us-west-2c` Jika Anda ingin mengubah subnet, Anda harus menyediakan setidaknya satu subnet di masing-masing dari tiga Availability Zone, dan subnet harus dalam VPC yang sama dengan subnet asli.

Jika Anda membutuhkan lebih banyak alamat IP daripada blok CIDR di VPC, Anda dapat menambahkan blok CIDR tambahan dengan mengaitkan blok Classless Inter-Domain Routing (CIDR) tambahan dengan VPC Anda. Anda dapat mengaitkan blok CIDR privat (RFC 1918) dan publik (non-RFC 1918) ke VPC Anda baik sebelum atau setelah Anda membuat klaster Anda. Diperlukan waktu cluster hingga lima jam agar blok CIDR yang Anda kaitkan dengan VPC dikenali.

Anda dapat menghemat penggunaan alamat IP dengan menggunakan gateway transit dengan VPC layanan bersama. Untuk informasi selengkapnya, lihat VPC terisolasi dengan layanan bersama dan Pola konservasi alamat IP yang dapat dirutekan VPC Amazon EKS dalam jaringan hibrida.

- Jika Anda Kubernetes ingin menetapkan `IPv6` alamat Pods dan layanan, kaitkan blok `IPv6` CIDR dengan VPC Anda. Untuk informasi selengkapnya, lihat Mengaitkan blok `IPv6` CIDR dengan VPC Anda di Panduan Pengguna Amazon VPC.
- VPC harus memiliki `DNS` nama host dan `DNS` dukungan resolusi. Jika tidak, node tidak dapat mendaftar ke cluster Anda. Untuk informasi selengkapnya, lihat atribut DNS untuk VPC Anda di Panduan Pengguna Amazon VPC.
- VPC mungkin memerlukan titik akhir VPC menggunakan. AWS PrivateLink Untuk informasi selengkapnya, lihat Persyaratan dan pertimbangan subnet.

Jika Anda membuat cluster dengan Kubernetes `1.14` atau sebelumnya, Amazon EKS menambahkan tag berikut ke VPC Anda:

Kunci	Nilai
`kubernetes.io/cluster/` *`my-cluster`*	`owned`

Tag ini hanya digunakan oleh Amazon EKS. Anda dapat menghapus tag tanpa memengaruhi layanan Anda. Ini tidak digunakan dengan cluster yang versi `1.15` atau lebih baru.

Persyaratan dan pertimbangan subnet

Saat Anda membuat klaster, Amazon EKS membuat 2-4 antarmuka jaringan elastis di subnet yang Anda tentukan. Antarmuka jaringan ini memungkinkan komunikasi antara cluster Anda dan VPC Anda. Antarmuka jaringan ini juga memungkinkan Kubernetes fitur seperti `kubectl exec` dan`kubectl logs`. Setiap antarmuka jaringan Amazon EKS yang dibuat memiliki teks `Amazon EKS` *`cluster-name`* dalam deskripsinya.

Amazon EKS dapat membuat antarmuka jaringannya di subnet apa pun yang Anda tentukan saat membuat cluster. Anda dapat mengubah subnet Amazon EKS yang membuat antarmuka jaringannya setelah cluster Anda dibuat. Saat Anda memperbarui Kubernetes versi cluster, Amazon EKS menghapus antarmuka jaringan asli yang dibuatnya, dan membuat antarmuka jaringan baru. Antarmuka jaringan ini dapat dibuat dalam subnet yang sama dengan antarmuka jaringan asli atau dalam subnet yang berbeda dari antarmuka jaringan asli. Untuk mengontrol antarmuka jaringan subnet mana yang dibuat, Anda dapat membatasi jumlah subnet yang Anda tentukan hanya dua saat Anda membuat klaster atau memperbarui subnet setelah membuat cluster.

Persyaratan subnet untuk cluster

Subnet yang Anda tentukan saat membuat atau memperbarui klaster harus memenuhi persyaratan berikut:

- Subnet masing-masing harus memiliki setidaknya enam alamat IP untuk digunakan oleh Amazon EKS. Namun, kami merekomendasikan setidaknya 16 alamat IP.
- Subnet tidak dapat berada di AWS Outposts, AWS Wavelength, atau Zona AWS Lokal. Namun, jika Anda memilikinya di VPC, Anda dapat menyebarkan node dan Kubernetes sumber daya yang dikelola sendiri ke jenis subnet ini.

- Subnet dapat bersifat publik atau pribadi. Namun, kami menyarankan Anda menentukan subnet pribadi, jika memungkinkan. Subnet publik adalah subnet dengan tabel rute yang mencakup rute ke gateway internet, sedangkan subnet pribadi adalah subnet dengan tabel rute yang tidak menyertakan rute ke gateway internet.
- Subnet tidak dapat berada di Availability Zone berikut:

Wilayah AWS	Nama Wilayah	ID Zona Ketersediaan Dilarang
`us-east-1`	AS Timur (Virginia Utara)	`use1-az3`
`us-west-1`	AS Barat (California Utara)	`usw1-az2`
`ca-central-1`	Kanada (Pusat)	`cac1-az3`

Penggunaan keluarga alamat IP berdasarkan komponen

Tabel berikut berisi keluarga alamat IP yang digunakan oleh setiap komponen Amazon EKS. Anda dapat menggunakan terjemahan alamat jaringan (NAT) atau sistem kompatibilitas lainnya untuk terhubung ke komponen ini dari alamat IP sumber dalam keluarga dengan "No" nilai untuk entri tabel.

Fungsionalitas dapat berbeda tergantung pada pengaturan IP family (`ipFamily`) cluster. Pengaturan ini mengubah jenis alamat IP yang digunakan untuk CIDR blok Kubernetes yang ditetapkan. Services Sebuah cluster dengan nilai pengaturan IPv4 disebut sebagaiIPv4 cluster, dan sebuah cluster dengan nilai pengaturan IPv6 disebut sebagaiIPv6 cluster.

Komponen	**`IPv4`**alamat saja	**`IPv6`**alamat saja	Alamat tumpukan ganda
Titik akhir publik EKS API	Ya	Tidak	Tidak
EKS API VPC titik akhir	Ya	Tidak	Tidak
Titik akhir publik EKS Auth API	Ya[1]	Ya[1]	Ya[1]

Komponen	**IPv4**alamat saja	**IPv6**alamat saja	Alamat tumpukan ganda
Titik akhir VPC EKS Auth API	Ya[1]	Ya[1]	Ya[1]
Titik akhir publik kluster EKS	Ya	Tidak	Tidak
Titik akhir pribadi kluster EKS	Ya[2]	Ya[2]	Tidak
Subnet kluster EKS	Ya[2]	Tidak	Ya[2]
Node Alamat IP Primer	Ya[2]	Tidak	Ya[2]
CIDRRentang cluster untuk alamat Service IP	Ya[2]	Ya[2]	Tidak
PodAlamat IP dari VPC CNI	Ya[2]	Ya[2]	Tidak

Note

[1] Titik akhir adalah tumpukan ganda dengan keduanya `IPv4` dan `IPv6` alamat. Aplikasi Anda di luar AWS, node Anda untuk cluster, dan pod Anda di dalam cluster dapat mencapai titik akhir ini dengan salah satu `IPv4` atau`IPv6`.

[2] Anda memilih antara `IPv4` cluster dan `IPv6` cluster dalam pengaturan IP family (`ipFamily`) cluster saat Anda membuat cluster dan ini tidak dapat diubah. Sebagai gantinya, Anda harus memilih pengaturan yang berbeda saat membuat klaster lain dan memigrasikan beban kerja Anda.

Persyaratan subnet untuk node

Anda dapat menyebarkan node dan Kubernetes sumber daya ke subnet yang sama dengan yang Anda tentukan saat membuat klaster. Namun, ini tidak perlu. Ini karena Anda juga dapat menyebarkan node dan Kubernetes sumber daya ke subnet yang tidak Anda tentukan saat membuat cluster. Jika Anda menerapkan node ke subnet yang berbeda, Amazon EKS tidak membuat antarmuka jaringan cluster di subnet tersebut. Subnet apa pun yang Anda gunakan node dan Kubernetes sumber daya harus memenuhi persyaratan berikut:

- Subnet harus memiliki cukup alamat IP yang tersedia untuk menyebarkan semua node dan Kubernetes sumber daya Anda.
- Jika Anda Kubernetes ingin menetapkan `IPv6` alamat Pods dan layanan, maka Anda harus memiliki satu blok `IPv6` CIDR dan satu blok `IPv4` CIDR yang terkait dengan subnet Anda. Untuk informasi selengkapnya, lihat Mengaitkan blok `IPv6` CIDR dengan subnet Anda di Panduan Pengguna Amazon VPC. Tabel rute yang terkait dengan subnet harus menyertakan rute ke `IPv4` dan `IPv6` alamat. Untuk informasi selengkapnya, lihat Rute di Panduan Pengguna Amazon VPC. Pod hanya diberi `IPv6` alamat. Namun antarmuka jaringan yang dibuat Amazon EKS untuk klaster Anda dan node Anda diberi alamat `IPv4` dan `IPv6` alamat.
- Jika Anda memerlukan akses masuk dari internet ke AndaPods, pastikan untuk memiliki setidaknya satu subnet publik dengan alamat IP yang cukup tersedia untuk menyebarkan penyeimbang beban dan pemasukan. Anda dapat menyebarkan penyeimbang beban ke subnet publik. Load balancer dapat memuat keseimbangan ke Pods subnet pribadi atau publik. Sebaiknya gunakan node Anda ke subnet pribadi, jika memungkinkan.
- Jika Anda berencana untuk menyebarkan node ke subnet publik, subnet harus menetapkan alamat atau alamat publik secara otomatis`IPv4`. `IPv6` Jika Anda menyebarkan node ke subnet pribadi yang memiliki blok `IPv6` CIDR terkait, subnet pribadi juga harus menetapkan alamat secara otomatis. `IPv6` Jika Anda menggunakan AWS CloudFormation template Amazon EKS untuk menerapkan VPC Anda setelah 26 Maret 2020, pengaturan ini diaktifkan. Jika Anda menggunakan template untuk menyebarkan VPC Anda sebelum tanggal ini atau Anda menggunakan VPC Anda sendiri, Anda harus mengaktifkan pengaturan ini secara manual. Untuk informasi selengkapnya, lihat Memodifikasi atribut `IPv4` pengalamatan publik untuk subnet Anda dan Memodifikasi atribut `IPv6` pengalamatan untuk subnet Anda di Panduan Pengguna Amazon VPC.
- Jika subnet tempat Anda menerapkan node adalah subnet pribadi dan tabel rutenya tidak menyertakan rute ke perangkat terjemahan alamat jaringan (NAT) () atau gateway khusus egres (`IPv4`)`IPv6`, tambahkan titik akhir VPC menggunakan ke VPC Anda. AWS PrivateLink Titik akhir VPC diperlukan untuk semua node Anda dan Pods perlu berkomunikasi dengannya. Layanan AWS

Contohnya termasuk Amazon ECR, Elastic Load Balancing, Amazon AWS Security Token Service, dan CloudWatch Amazon Simple Storage Service (Amazon S3). Titik akhir harus menyertakan subnet tempat node berada. Tidak semua Layanan AWS mendukung titik akhir VPC. Untuk informasi lebih lanjut, lihat Apa itu AWS PrivateLink? dan AWS layanan yang terintegrasi dengan AWS PrivateLink. Untuk daftar persyaratan Amazon EKS lainnya, lihatPersyaratan klaster pribadi.

- Jika Anda ingin menerapkan penyeimbang beban ke subnet, subnet harus memiliki tag berikut:
  - Subnet privat

Kunci	Nilai
`kubernetes.io/role/internal-elb`	`1`

  - Subnet publik

Kunci	Nilai
`kubernetes.io/role/elb`	`1`

Ketika Kubernetes cluster yang versi `1.18` dan sebelumnya dibuat, Amazon EKS menambahkan tag berikut ke semua subnet yang ditentukan.

Kunci	Nilai
`kubernetes.io/cluster/` *`my-cluster`*	`shared`

Saat Anda membuat Kubernetes cluster baru sekarang, Amazon EKS tidak menambahkan tag ke subnet Anda. Jika tag berada di subnet yang digunakan oleh cluster yang sebelumnya merupakan versi sebelumnya`1.19`, tag tidak secara otomatis dihapus dari subnet saat cluster diperbarui ke versi yang lebih baru. Versi `2.1.1` atau sebelumnya AWS Load Balancer Controllermembutuhkan tag ini. Jika Anda menggunakan versi yang lebih baru dari Load Balancer Controller, Anda dapat menghapus tag tanpa mengganggu layanan Anda.

Jika Anda menerapkan VPC dengan `eksctl`menggunakan atau salah satu templat AWS CloudFormation VPC Amazon EKS, berikut ini berlaku:

- Pada atau setelah 26 Maret 2020 - `IPv4` Alamat publik secara otomatis ditetapkan oleh subnet publik ke node baru yang digunakan untuk subnet publik.
- Sebelum 26 Maret 2020 - `IPv4` Alamat publik tidak secara otomatis ditetapkan oleh subnet publik ke node baru yang digunakan ke subnet publik.

Perubahan ini berdampak pada grup node baru yang diterapkan ke subnet publik dengan cara berikut:

- Grup node terkelola - Jika grup node dikerahkan ke subnet publik pada atau setelah 22 April 2020, penetapan otomatis alamat IP publik harus diaktifkan untuk subnet publik. Untuk informasi selengkapnya, lihat Memodifikasi atribut `IPv4` pengalamatan publik untuk subnet Anda.
- Linux, Windows, atau grup node yang dikelola sendiri Arm — Jika grup node diterapkan ke subnet publik pada atau setelah 26 Maret 2020, penetapan otomatis alamat IP publik harus diaktifkan untuk subnet publik. Jika tidak, node harus diluncurkan dengan alamat IP publik sebagai gantinya. Untuk informasi selengkapnya, lihat Memodifikasi atribut `IPv4` pengalamatan publik untuk subnet Anda atau Menetapkan `IPv4` alamat publik selama peluncuran instance.

Persyaratan dan pertimbangan subnet bersama

Anda dapat menggunakan berbagi VPC untuk berbagi subnet dengan AWS akun lain dalam hal yang sama. AWS Organizations Anda dapat membuat klaster Amazon EKS di subnet bersama, dengan pertimbangan berikut:

- Pemilik subnet VPC harus berbagi subnet dengan akun peserta sebelum akun tersebut dapat membuat cluster Amazon EKS di dalamnya.
- Anda tidak dapat meluncurkan sumber daya menggunakan grup keamanan default untuk VPC karena milik pemilik. Selain itu, peserta tidak dapat meluncurkan sumber daya menggunakan grup keamanan yang dimiliki oleh peserta lain atau pemilik.
- Dalam subnet bersama, peserta dan pemilik secara terpisah mengontrol grup keamanan dalam setiap akun masing-masing. Pemilik subnet dapat melihat grup keamanan yang dibuat oleh peserta tetapi tidak dapat melakukan tindakan apa pun pada mereka. Jika pemilik subnet ingin menghapus atau memodifikasi grup keamanan ini, peserta yang membuat grup keamanan harus mengambil tindakan.
- Jika cluster dibuat oleh peserta, pertimbangan berikut berlaku:

  - Peran IAM cluster dan peran IAM Node harus dibuat di akun itu. Lihat informasi yang lebih lengkap di IAM role klaster Amazon EKS dan IAM role simpul Amazon EKS.
  - Semua node harus dibuat oleh peserta yang sama, termasuk grup node terkelola.
- Pemilik VPC bersama tidak dapat melihat, memperbarui, atau menghapus klaster yang dibuat peserta di subnet bersama. Ini merupakan tambahan dari sumber daya VPC yang setiap akun memiliki akses berbeda. Untuk informasi selengkapnya, lihat Tanggung jawab dan izin untuk pemilik dan peserta di Panduan Pengguna Amazon VPC.
- Jika Anda menggunakan fitur jaringan kustom Amazon VPC CNI plugin for Kubernetes, Anda perlu menggunakan pemetaan ID Availability Zone yang tercantum di akun pemilik untuk membuat masing-masing. `ENIConfig` Untuk informasi selengkapnya, lihat Jaringan khusus untuk pod.

Untuk informasi selengkapnya tentang berbagi subnet VPC, lihat Bagikan VPC Anda dengan akun lain di Panduan Pengguna Amazon VPC.

Membuat VPC untuk klaster Amazon EKS Anda

Anda dapat menggunakan Amazon Virtual Private Cloud (Amazon VPC) untuk meluncurkan AWS sumber daya ke jaringan virtual yang telah Anda tentukan. Jaringan virtual ini sangat mirip dengan jaringan tradisional yang mungkin Anda operasikan di pusat data Anda sendiri. Namun, ia datang dengan manfaat menggunakan infrastruktur yang dapat diskalakan dari Amazon Web Services. Kami menyarankan Anda untuk memiliki pemahaman menyeluruh tentang layanan VPC Amazon sebelum menerapkan cluster Amazon EKS produksi. Untuk informasi selengkapnya, silakan lihat ACL Jaringan di Panduan Pengguna Amazon VPC.

Kluster, node, dan Kubernetes sumber daya Amazon EKS disebarkan ke VPC. Jika Anda ingin menggunakan VPC yang ada dengan Amazon EKS, VPC itu harus memenuhi persyaratan yang dijelaskan di. Persyaratan dan pertimbangan Amazon EKS VPC dan subnet Topik ini menjelaskan cara membuat VPC yang memenuhi persyaratan Amazon EKS menggunakan templat yang disediakan AWS CloudFormation Amazon EKS. Setelah menerapkan template, Anda dapat melihat sumber daya yang dibuat oleh template untuk mengetahui dengan tepat sumber daya apa yang dibuatnya, dan konfigurasi sumber daya tersebut.

Prasyarat

Untuk membuat VPC untuk Amazon EKS, Anda harus memiliki izin IAM yang diperlukan untuk membuat sumber daya Amazon VPC. Sumber daya ini adalah VPC, subnet, grup keamanan, tabel dan rute rute, dan gateway internet dan NAT. Untuk informasi selengkapnya, lihat Membuat VPC

dengan kebijakan contoh subnet publik di Panduan Pengguna Amazon VPC dan daftar lengkap Tindakan, sumber daya, dan kunci kondisi untuk Amazon EC2 di Referensi Otorisasi Layanan.

Anda dapat membuat VPC dengan subnet publik dan privat, subnet publik saja, atau subnet privat saja.

Public and private subnets

VPC ini memiliki dua subnet publik dan dua subnet pribadi. Tabel rute terkait subnet publik memiliki rute ke gateway internet. Namun, tabel rute subnet pribadi tidak memiliki rute ke gateway internet. Satu subnet publik dan satu subnet privat di-deploy ke Availability Zone yang sama. Subnet publik dan pribadi lainnya dikerahkan ke Availability Zone kedua dalam hal yang sama. Wilayah AWS Kami merekomendasikan opsi ini untuk sebagian besar penerapan.

Dengan opsi ini, Anda dapat menyebarkan node Anda ke subnet pribadi. Opsi ini memungkinkan Kubernetes untuk menyebarkan penyeimbang beban ke subnet publik yang dapat memuat lalu lintas keseimbangan ke Pods node di subnet pribadi. `IPv4`Alamat publik secara otomatis ditetapkan ke node yang disebarkan ke subnet publik, tetapi `IPv4` alamat publik tidak ditetapkan ke node yang disebarkan ke subnet pribadi.

Anda juga dapat menetapkan `IPv6` alamat ke node di subnet publik dan pribadi. Node dalam subnet pribadi dapat berkomunikasi dengan cluster dan lainnyaLayanan AWS. Podsdapat berkomunikasi ke internet melalui gateway NAT menggunakan `IPv4` alamat atau gateway Internet outbound-only menggunakan alamat yang digunakan di setiap Availability `IPv6` Zone. Grup keamanan dikerahkan yang memiliki aturan yang menolak semua lalu lintas masuk dari sumber selain cluster atau node tetapi memungkinkan semua lalu lintas keluar. Subnet diberi tag sehingga Kubernetes dapat menyebarkan penyeimbang beban ke mereka.

Untuk membuat VPC Anda

1. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
2. Dari bilah navigasi, pilih Wilayah AWS yang mendukung Amazon EKS.
3. Pilih Buat tumpukan, Dengan sumber daya baru (standar).
4. Di bawah Prasyarat - Siapkan templat, pastikan Template sudah siap dipilih dan kemudian di bawah Tentukan templat, pilih URL Amazon S3.
5. Anda dapat membuat VPC yang hanya mendukung `IPv4`, atau VPC yang mendukung dan. `IPv4` `IPv6` Tempelkan salah satu URL berikut ke area teks di bawah URL Amazon S3 dan pilih Berikutnya:

- IPv4

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
amazon-eks-vpc-private-subnets.yaml
```

- IPv4 dan IPv6

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
amazon-eks-ipv6-vpc-public-private-subnets.yaml
```

6. Pada halaman Tentukan detail tumpukan, masukkan parameter, lalu pilih Berikutnya.

- Nama tumpukan: Pilih nama tumpukan untuk tumpukan AWS CloudFormation Anda. Misalnya, Anda dapat menggunakan nama template yang Anda gunakan pada langkah sebelumnya. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter.
- VpcBlock: Pilih rentang IPv4 CIDR untuk VPC Anda. Setiap node,Pod, dan penyeimbang beban yang Anda terapkan diberi IPv4 alamat dari blok ini. IPv4Nilai default menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, maka Anda dapat mengubahnya. Untuk informasi selengkapnya, lihat Pengukuran VPC and subnet dalam Panduan Pengguna Amazon VPC. Anda juga dapat menambahkan blok CIDR tambahan ke VPC setelah dibuat. Jika Anda membuat IPv6 VPC, rentang IPv6 CIDR secara otomatis ditetapkan untuk Anda dari ruang Alamat Unicast Global Amazon.
- PublicSubnet01Block: Tentukan blok IPv4 CIDR untuk subnet publik 1. Nilai default Menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, Anda dapat mengubahnya. Jika Anda membuat IPv6 VPC, blok ini ditentukan untuk Anda dalam template.
- PublicSubnet02Block: Tentukan blok IPv4 CIDR untuk subnet publik 2. Nilai default Menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, Anda dapat mengubahnya. Jika Anda membuat IPv6 VPC, blok ini ditentukan untuk Anda dalam template.
- PrivateSubnet01Block: Tentukan blok IPv4 CIDR untuk subnet pribadi 1. Nilai default Menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, Anda dapat mengubahnya. Jika Anda membuat IPv6 VPC, blok ini ditentukan untuk Anda dalam template.

- PrivateSubnet02Block: Tentukan blok `IPv4` CIDR untuk subnet pribadi 2. Nilai default Menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, Anda dapat mengubahnya. Jika Anda membuat `IPv6` VPC, blok ini ditentukan untuk Anda dalam template.

7. (Opsional) Pada halaman Configure stack options, beri tag sumber daya tumpukan Anda dan kemudian pilih Berikutnya.
8. Di halaman Tinjau, pilih Buat tumpukan.
9. Ketika tumpukan Anda dibuat, pilih tumpukan tersebut di dalam konsol dan pilih Outputs.
10. Rekam VpcIduntuk VPC yang dibuat. Anda membutuhkan ini ketika Anda membuat cluster dan node Anda.
11. Rekam subnet yang dibuat dan apakah Anda membuatnya sebagai subnet publik atau pribadi. SubnetIds Anda memerlukan setidaknya dua dari ini ketika Anda membuat cluster dan node Anda.
12. Jika Anda membuat `IPv4` VPC, lewati langkah ini. Jika Anda membuat `IPv6` VPC, Anda harus mengaktifkan opsi `IPv6` alamat penetapan otomatis untuk subnet publik yang dibuat oleh templat. Pengaturan itu sudah diaktifkan untuk subnet pribadi. Untuk mengaktifkan pengaturan, selesaikan langkah-langkah berikut:

    a. Buka konsol Amazon VPC di https://console.aws.amazon.com/vpc/.

    b. Di panel navigasi kiri, pilih Subnet

    c. Pilih salah satu subnet publik Anda (***stack-name/SubnetPublic01 atau stack-name***/SubnetPublic02 berisi kata public) dan pilih Actions, Edit pengaturan subnet.

    d. Pilih kotak centang Aktifkan **`IPv6`** alamat penetapan otomatis dan kemudian pilih Simpan.

    e. Selesaikan langkah-langkah sebelumnya lagi untuk subnet publik Anda yang lain.

Only public subnets

VPC ini memiliki tiga subnet publik yang disebarkan ke Availability Zone yang berbeda dalam file. Wilayah AWS Semua node secara otomatis diberi `IPv4` alamat publik dan dapat mengirim dan menerima lalu lintas internet melalui gateway internet. Sebuah grup subnet di-deploy, dimana ia menolak semua lalu lintas masuk dan mengizinkan semua lalu lintas keluar. Subnet diberi tag sehingga Kubernetes dapat menyebarkan penyeimbang beban ke mereka.

Untuk membuat VPC Anda

1. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
2. Dari bilah navigasi, pilih Wilayah AWS yang mendukung Amazon EKS.
3. Pilih Buat tumpukan, Dengan sumber daya baru (standar).
4. Di bawah Siapkan template, pastikan Template sudah siap dipilih dan kemudian di bawah Sumber template, pilih URL Amazon S3.
5. Rekatkan URL berikut ke area teks di bawah URL Amazon S3 dan pilih Berikutnya:

   ```
   https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-sample.yaml
   ```

6. Pada halaman Tentukan Detail, masukkan parameter, lalu pilih Berikutnya.

   - Nama tumpukan: Pilih nama tumpukan untuk tumpukan AWS CloudFormation Anda. Misalnya, Anda bisa menyebutnya***amazon-eks-vpc-sample***. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter.
   - VpcBlock: Pilih blok CIDR untuk VPC Anda. Setiap node,Pod, dan penyeimbang beban yang Anda terapkan diberi `IPv4` alamat dari blok ini. `IPv4`Nilai default menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, maka Anda dapat mengubahnya. Untuk informasi selengkapnya, lihat Pengukuran VPC and subnet dalam Panduan Pengguna Amazon VPC. Anda juga dapat menambahkan blok CIDR tambahan ke VPC setelah dibuat.
   - Subnet01Block: Tentukan blok CIDR untuk subnet 1. Nilai default Menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, Anda dapat mengubahnya.
   - Subnet02Block: Tentukan blok CIDR untuk subnet 2. Nilai default Menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, Anda dapat mengubahnya.
   - Subnet03Block: Tentukan blok CIDR untuk subnet 3. Nilai default Menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, Anda dapat mengubahnya.
7. (Opsional) Di halaman Opsi, tandai sumber daya tumpukan Anda. Pilih Berikutnya.
8. Pada halaman Tinjauan, pilih Buat.
9. Ketika tumpukan Anda dibuat, pilih tumpukan tersebut di konsol dan pilih Outputs.

10. Rekam VpcIduntuk VPC yang dibuat. Anda membutuhkan ini ketika Anda membuat cluster dan node Anda.
11. Rekam SubnetIdsuntuk subnet yang dibuat. Anda memerlukan setidaknya dua dari ini ketika Anda membuat cluster dan node Anda.
12. (Opsional) Kluster apa pun yang Anda terapkan ke VPC ini dapat menetapkan alamat `IPv4` pribadi ke alamat Anda dan. Pods services Jika Anda ingin menerapkan cluster ke VPC ini untuk menetapkan `IPv6` alamat pribadi ke Pods danservices, buat pembaruan ke VPC, subnet, tabel rute, dan grup keamanan Anda. Untuk informasi selengkapnya, lihat Memigrasi VPC yang ada dari `IPv4` ke `IPv6` dalam Panduan Pengguna Amazon VPC. Amazon EKS mengharuskan subnet Anda mengaktifkan opsi `Auto-assign IPv6` alamat. Secara default, ini dinonaktifkan.

Only private subnets

VPC ini memiliki tiga subnet pribadi yang digunakan ke Availability Zone yang berbeda di. Wilayah AWS Sumber daya yang dikerahkan ke subnet tidak dapat mengakses internet, juga tidak dapat mengakses sumber daya internet di subnet. Template membuat titik akhir VPC menggunakan AWS PrivateLink beberapa node Layanan AWS yang biasanya perlu diakses. Jika node Anda memerlukan akses internet keluar, Anda dapat menambahkan gateway NAT publik di Availability Zone setiap subnet setelah VPC dibuat. Grup keamanan dibuat yang menyangkal semua lalu lintas masuk, kecuali dari sumber daya yang dikerahkan ke subnet. Grup keamanan juga memungkinkan semua lalu lintas keluar. Subnet diberi tag sehingga Kubernetes dapat menyebarkan penyeimbang beban internal kepada mereka. Jika Anda membuat VPC dengan konfigurasi ini, lihat Persyaratan klaster pribadi persyaratan dan pertimbangan tambahan.

Untuk membuat VPC Anda

1. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
2. Dari bilah navigasi, pilih Wilayah AWS yang mendukung Amazon EKS.
3. Pilih Buat tumpukan, Dengan sumber daya baru (standar).
4. Di bawah Siapkan template, pastikan Template sudah siap dipilih dan kemudian di bawah Sumber template, pilih URL Amazon S3.
5. Rekatkan URL berikut ke area teks di bawah URL Amazon S3 dan pilih Berikutnya:

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-fully-private-vpc.yaml
```

6. Pada halaman Tentukan Detail, masukkan parameter dan kemudian pilih Berikutnya.

   - Nama tumpukan: Pilih nama tumpukan untuk tumpukan AWS CloudFormation Anda. Misalnya, Anda bisa menyebutnya***amazon-eks-fully-private-vpc***. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter.
   - VpcBlock: Pilih blok CIDR untuk VPC Anda. Setiap node,Pod, dan penyeimbang beban yang Anda terapkan diberi `IPv4` alamat dari blok ini. `IPv4`Nilai default menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, maka Anda dapat mengubahnya. Untuk informasi selengkapnya, lihat Pengukuran VPC and subnet dalam Panduan Pengguna Amazon VPC. Anda juga dapat menambahkan blok CIDR tambahan ke VPC setelah dibuat.
   - PrivateSubnet01Block: Tentukan blok CIDR untuk subnet 1. Nilai default Menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, Anda dapat mengubahnya.
   - PrivateSubnet02Block: Tentukan blok CIDR untuk subnet 2. Nilai default Menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, Anda dapat mengubahnya.
   - PrivateSubnet03Block: Tentukan blok CIDR untuk subnet 3. Nilai default Menyediakan alamat IP yang cukup untuk sebagian besar implementasi, tetapi jika tidak, Anda dapat mengubahnya.
7. (Opsional) Di halaman Opsi, tandai sumber daya tumpukan Anda. Pilih Berikutnya.
8. Pada halaman Tinjauan, pilih Buat.
9. Ketika tumpukan Anda dibuat, pilih tumpukan tersebut di konsol dan pilih Outputs.
10. Rekam VpcIduntuk VPC yang dibuat. Anda membutuhkan ini ketika Anda membuat cluster dan node Anda.
11. Rekam SubnetIdsuntuk subnet yang dibuat. Anda memerlukan setidaknya dua dari ini ketika Anda membuat cluster dan node Anda.
12. (Opsional) Kluster apa pun yang Anda terapkan ke VPC ini dapat menetapkan alamat `IPv4` pribadi ke alamat Anda dan. Pods services Jika Anda ingin menyebarkan cluster ke VPC ini untuk menetapkan `IPv6` alamat pribadi ke Pods danservices, buat pembaruan ke VPC, subnet, tabel rute, dan grup keamanan Anda. Untuk informasi selengkapnya, lihat Memigrasi VPC yang ada dari `IPv4` ke `IPv6` dalam Panduan Pengguna Amazon VPC. Amazon EKS mengharuskan subnet Anda mengaktifkan opsi `Auto-assign` `IPv6` alamat (dinonaktifkan secara default).

Persyaratan dan pertimbangan grup keamanan Amazon EKS

Topik ini menjelaskan persyaratan grup keamanan klaster Amazon EKS.

Saat Anda membuat klaster, Amazon EKS membuat grup keamanan yang diberi nama`eks-cluster-sg-`*`my-cluster`*`-`*`uniqueID`*. Grup keamanan ini memiliki aturan default berikut:

Jenis aturan	Protokol	Port	Sumber	Tujuan
Ke dalam	Semua	Semua	Diri Sendiri	
Ke luar	Semua	Semua		0.0.0.0/0 (`IPv4`) atau: :/0 () `IPv6`

> **Important**
>
> Jika klaster Anda tidak memerlukan aturan keluar, Anda dapat menghapusnya. Jika Anda menghapusnya, Anda masih harus memiliki aturan minimum yang tercantum dalam Membatasi lalu lintas cluster. Jika Anda menghapus aturan masuk, Amazon EKS akan membuatnya ulang setiap kali cluster diperbarui.

Amazon EKS menambahkan tag berikut ke grup keamanan. Jika Anda menghapus tag, Amazon EKS menambahkannya kembali ke grup keamanan setiap kali klaster Anda diperbarui.

Kunci	Nilai
`kubernetes.io/cluster/` *`my-cluster`*	`owned`
`aws:eks:cluster-name`	*`my-cluster`*
`Name`	`eks-cluster-sg-` *`my-cluster`* *`-uniqueid`*

Amazon EKS secara otomatis mengaitkan grup keamanan ini ke sumber daya berikut yang juga dibuatnya:

- 2—4 antarmuka jaringan elastis (disebut untuk sisa dokumen ini sebagai antarmuka jaringan) yang dibuat saat Anda membuat cluster Anda.
- Antarmuka jaringan node dalam grup node terkelola apa pun yang Anda buat.

Aturan default memungkinkan semua lalu lintas mengalir bebas antara cluster dan node Anda, dan memungkinkan semua lalu lintas keluar ke tujuan mana pun. Saat membuat klaster, Anda dapat (opsional) menentukan grup keamanan Anda sendiri. Jika ya, Amazon EKS juga mengaitkan grup keamanan yang Anda tentukan ke antarmuka jaringan yang dibuatnya untuk cluster Anda. Namun, itu tidak mengaitkannya dengan grup simpul apa pun yang Anda buat.

Anda dapat menentukan ID grup keamanan klaster Anda di AWS Management Console bawah bagian Jaringan kluster. Atau, Anda dapat melakukannya dengan menjalankan AWS CLI perintah berikut.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

Membatasi lalu lintas klaster

Jika Anda perlu membatasi port terbuka antara cluster dan node, Anda dapat menghapus aturan keluar default dan menambahkan aturan minimum berikut yang diperlukan untuk cluster. Jika Anda menghapus aturan masuk default, Amazon EKS akan membuatnya ulang setiap kali cluster diperbarui.

Jenis aturan	Protokol	Port	Tujuan
Ke luar	TCP	443	Grup keamanan klaster
Ke luar	TCP	10250	Grup keamanan klaster
Keluar (DNS)	TCP dan UDP	53	Grup keamanan klaster

Anda juga harus menambahkan aturan untuk lalu lintas berikut:

- Protokol dan port apa pun yang Anda harapkan untuk digunakan node untuk komunikasi antar simpul.
- Akses internet keluar sehingga node dapat mengakses Amazon EKS API untuk introspeksi cluster dan pendaftaran node pada waktu peluncuran. Jika node Anda tidak memiliki akses internet, tinjau Persyaratan klaster pribadi untuk pertimbangan tambahan.
- Akses node untuk menarik gambar kontainer dari Amazon ECR atau API pendaftar kontainer lainnya yang mereka butuhkan untuk menarik gambar, seperti. DockerHub Untuk informasi selengkapnya, lihat Rentang alamat IP AWS di Referensi Umum AWS.
- Akses node ke Amazon S3.
- Aturan terpisah diperlukan untuk `IPv4` dan `IPv6` alamat.

Jika Anda mempertimbangkan untuk membatasi aturan, kami sarankan Anda menguji semua aturan Anda secara menyeluruh Pods sebelum menerapkan aturan yang diubah ke klaster produksi.

Jika Anda awalnya menggunakan cluster dengan Kubernetes `1.14` dan versi platform `eks.3` atau sebelumnya, pertimbangkan hal berikut:

- Anda mungkin juga memiliki control plane dan grup keamanan node. Ketika grup ini dibuat, mereka menyertakan aturan terbatas yang tercantum dalam tabel sebelumnya. Grup keamanan ini tidak lagi diperlukan dan dapat dihapus. Namun, Anda perlu memastikan grup keamanan klaster Anda berisi aturan yang dikandung grup tersebut.
- Jika Anda menerapkan cluster menggunakan API secara langsung atau Anda menggunakan alat seperti AWS CLI atau AWS CloudFormation untuk membuat cluster dan Anda tidak menentukan grup keamanan pada pembuatan klaster, maka grup keamanan default untuk VPC diterapkan ke antarmuka jaringan cluster yang dibuat Amazon EKS.

Pengaya jaringan Amazon EKS

Beberapa add-on jaringan tersedia untuk kluster Amazon EKS Anda.

Pengaya bawaan

> **Note**
>
> Jika Anda membuat cluster dengan cara apa pun kecuali dengan menggunakan konsol, setiap cluster dilengkapi dengan versi add-on bawaan yang dikelola sendiri. Versi yang

dikelola sendiri tidak dapat dikelola dari AWS Management Console, AWS Command Line Interface, atau SDK. Anda mengelola konfigurasi dan upgrade add-on yang dikelola sendiri. Sebaiknya tambahkan jenis add-on Amazon EKS ke klaster Anda alih-alih menggunakan jenis add-on yang dikelola sendiri. Jika Anda membuat cluster di konsol, jenis Amazon EKS dari add-on ini diinstal.

Amazon VPC CNI plugin for Kubernetes

Add-on CNI ini menciptakan antarmuka jaringan elastis dan menempelkannya ke node Amazon EC2 Anda. Add-on ini juga memberikan alamat pribadi `IPv4` atau `IPv6` alamat dari VPC Anda ke masing-masing dan layanan. Pod Add-on ini diinstal, secara default, di cluster Anda. Untuk informasi selengkapnya, lihat Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS.

CoreDNS

CoreDNSadalah server DNS yang fleksibel dan dapat diperluas yang dapat berfungsi sebagai DNS Kubernetes cluster. CoreDNSmemberikan resolusi nama untuk semua Pods di cluster. Add-on ini diinstal, secara default, di cluster Anda. Untuk informasi selengkapnya, lihat Bekerja dengan add-on CoreDNS Amazon EKS.

`kube-proxy`

Add-on ini mempertahankan aturan jaringan pada node Amazon EC2 Anda dan memungkinkan komunikasi jaringan ke Anda. Pods Add-on ini diinstal, secara default, di cluster Anda. Untuk informasi selengkapnya, lihat Bekerja dengan add-on Kubernetes `kube-proxy`.

Pengaya AWS jaringan opsional

AWS Load Balancer Controller

Saat Anda menyebarkan objek Kubernetes layanan dari jenis`loadbalancer`, pengontrol membuat AWS Network Load Balancers. Saat Anda membuat objek Kubernetes ingress, pengontrol membuat AWS Application Load Balancers. Sebaiknya gunakan pengontrol ini untuk menyediakan Network Load Balancers, daripada menggunakan pengontrol Cloud Provider lama yang ada di dalamnya. Kubernetes Untuk informasi lebih lanjut, lihat AWS Load Balancer Controllerdokumentasi.

AWS Pengontrol API Gerbang

Pengontrol ini memungkinkan Anda menghubungkan layanan di beberapa Kubernetes cluster menggunakan API Kubernetes gateway. Pengontrol menghubungkan Kubernetes layanan yang berjalan di instans Amazon EC2, container, dan fungsi tanpa server dengan menggunakan layanan Amazon VPC Lattice. Untuk informasi selengkapnya, lihat dokumentasi AWS Gateway API Controller.

Untuk informasi selengkapnya tentang add-on, lihatAdd-on Amazon EKS.

Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS

Amazon VPC CNI plugin for KubernetesAdd-on ini diterapkan pada setiap node Amazon EC2 di cluster Amazon EKS Anda. Add-on membuat antarmuka jaringan elastis dan menempelkannya ke node Amazon EC2 Anda. Add-on ini juga memberikan alamat pribadi `IPv4` atau `IPv6` alamat dari VPC Anda ke masing-masing dan layanan. Pod

Versi add-on digunakan dengan setiap node Fargate di cluster Anda, tetapi Anda tidak memperbaruinya di node Fargate. Plugin CNI lain yang kompatibel tersedia untuk digunakan pada kluster Amazon EKS, tetapi ini adalah satu-satunya plugin CNI yang didukung oleh Amazon EKS.

Tabel berikut mencantumkan versi terbaru yang tersedia dari jenis add-on Amazon EKS untuk setiap Kubernetes versi.

Versi Kubernetes	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**	**1.24**	**1.23**
Amazon EKS jenis versi VPC CNI	`v1.18.0-eksbuild`	`v1.18.0-eksbuild`	`v1.18.0-eksbuild`	`v1.18.0-eksbuild`	`v1.18.0-eksbuild`	`v1.18.0-eksbuild`	`v1.18.0-eksbuild.1`

> **Important**
>
> Jika Anda mengelola sendiri add-on ini, versi dalam tabel mungkin tidak sama dengan versi yang dikelola sendiri yang tersedia. Untuk informasi selengkapnya tentang memperbarui jenis pengaya yang dikelola sendiri, lihat. Memperbarui add-on yang dikelola sendiri

Prasyarat

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS.
- Penyedia AWS Identity and Access Management (IAM) OpenID Connect (OIDC) yang sudah ada untuk cluster Anda. Untuk menentukan apakah Anda sudah memiliki satu, atau harus membuat satu, lihat Buat OIDC penyedia IAM untuk klaster Anda.
- Peran IAM dengan kebijakan IAM Amazoneks_CNI_Policy (jika klaster Anda menggunakan keluarga) atau kebijakan IPv6 (jika klaster Anda menggunakan keluarga) `IPv4` yang dilampirkan padanya. `IPv6` Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).
- Jika Anda menggunakan versi `1.7.0` atau yang lebih baru Amazon VPC CNI plugin for Kubernetes dan Anda menggunakan kebijakan Pod keamanan khusus, lihat Hapus kebijakan Pod keamanan Amazon EKS defaultKebijakan keamanan pod.
-

  > **Important**
  >
  > Amazon VPC CNI plugin for Kubernetesversi `v1.16.0` untuk `v1.16.1` menghapus kompatibilitas dengan Kubernetes versi `1.23` dan sebelumnya. Versi VPC CNI `v1.16.2` mengembalikan kompatibilitas dengan Kubernetes versi `1.23` dan sebelumnya dan spesifikasi CNI. `v0.4.0`

  Amazon VPC CNI plugin for Kubernetesversi `v1.16.0` untuk `v1.16.1` mengimplementasikan versi `v1.0.0` spesifikasi CNI. Spesifikasi `v1.0.0` CNI didukung pada kluster EKS yang menjalankan Kubernetes versi `v1.24` atau yang lebih baru. Versi VPC CNI `v1.16.0` ke `v1.16.1` dan spesifikasi CNI `v1.0.0` tidak didukung pada versi atau versi sebelumnya. Kubernetes `v1.23` Untuk informasi lebih lanjut `v1.0.0` tentang spesifikasi CNI, lihat Spesifikasi Container Network Interface (CNI) di

Pertimbangan

- Versi ditentukan sebagai`major-version.minor-version.patch-version-eksbuild.build-number`.
- Periksa kompatibilitas versi untuk setiap fitur

Beberapa fitur dari setiap rilis Amazon VPC CNI plugin for Kubernetes memerlukan Kubernetes versi certian. Saat menggunakan fitur Amazon EKS yang berbeda, jika versi add-on tertentu diperlukan, maka itu dicatat dalam dokumentasi fitur. Kecuali Anda memiliki alasan khusus untuk menjalankan versi sebelumnya, kami sarankan untuk menjalankan versi terbaru.

Membuat add-on Amazon EKS

Buat jenis add-on Amazon EKS.

1. Lihat versi add-on mana yang diinstal pada cluster Anda.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
 | cut -d : -f 3
```

Contoh output adalah sebagai berikut.

```
v1.12.6-eksbuild.2
```

2. Lihat jenis add-on yang diinstal pada cluster Anda. Bergantung pada alat yang digunakan untuk membuat klaster, saat ini Anda mungkin tidak menginstal jenis add-on Amazon EKS di cluster Anda. Ganti *my-cluster* dengan nama klaster Anda.

```
$ aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

Jika nomor versi dikembalikan, Anda memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda dan tidak perlu menyelesaikan langkah-langkah yang tersisa dalam prosedur ini. Jika kesalahan dikembalikan, Anda tidak memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda. Selesaikan langkah-langkah yang tersisa dari prosedur ini untuk menginstalnya.

3. Simpan konfigurasi add-on yang Anda instal saat ini.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

4. Buat add-on menggunakan. AWS CLI Jika Anda ingin menggunakan AWS Management Console atau `eksctl` untuk membuat add-on, lihat Membuat add-on dan tentukan nama `vpc-cni` add-on. Salin perintah yang mengikuti ke perangkat Anda. Buat modifikasi berikut pada perintah, sesuai kebutuhan, dan kemudian jalankan perintah yang dimodifikasi.

- Ganti *my-cluster* dengan nama klaster Anda.
- Ganti *v1.18.0-eksbuild.1*dengan versi terbaru yang tercantum dalam tabel versi terbaru untuk versi cluster Anda.
- Ganti *111122223333* dengan ID akun Anda dan *AmazonKsvPCCnirole dengan nama peran* IAM yang ada yang telah Anda buat. Menentukan peran mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk klaster Anda. Untuk menentukan apakah Anda memiliki satu untuk cluster Anda, atau untuk membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda.

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.18.0-eksbuild.1 \
    --service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
```

Jika Anda telah menerapkan pengaturan khusus ke add-on saat ini yang bertentangan dengan pengaturan default add-on Amazon EKS, pembuatan mungkin gagal. Jika pembuatan gagal, Anda menerima kesalahan yang dapat membantu Anda menyelesaikan masalah. Atau, Anda dapat menambahkan **--resolve-conflicts OVERWRITE** ke perintah sebelumnya. Hal ini memungkinkan add-on untuk menimpa pengaturan kustom yang ada. Setelah Anda membuat add-on, Anda dapat memperbaruinya dengan pengaturan khusus Anda.

5. Konfirmasikan bahwa versi terbaru add-on untuk Kubernetes versi klaster Anda telah ditambahkan ke klaster Anda. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

Mungkin perlu beberapa detik untuk menyelesaikan pembuatan add-on.

Contoh output adalah sebagai berikut.

```
v1.18.0-eksbuild.1
```

6. Jika Anda membuat pengaturan khusus untuk add-on asli Anda, sebelum Anda membuat add-on Amazon EKS, gunakan konfigurasi yang Anda simpan di langkah sebelumnya untuk memperbarui add-on Amazon EKS dengan pengaturan khusus Anda.

7. (Opsional) Instal `cni-metrics-helper` ke cluster Anda. Ini mengikis elastic network interface dan informasi alamat IP, menggabungkannya pada tingkat cluster, dan menerbitkan metrik ke Amazon. CloudWatch Untuk informasi lebih lanjut, lihat cni-metrics-helperdi GitHub.

Memperbarui add-on Amazon EKS

Perbarui jenis add-on Amazon EKS. Jika Anda belum menambahkan jenis add-on Amazon EKS ke klaster Anda, tambahkan atau lihatMemperbarui add-on yang dikelola sendiri, alih-alih menyelesaikan prosedur ini.

1. Lihat versi add-on mana yang diinstal pada cluster Anda. Ganti *my-cluster* dengan nama klaster Anda.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
    "addon.addonVersion" --output text
   ```

   Contoh output adalah sebagai berikut.

   ```
   v1.12.6-eksbuild.2
   ```

   Jika versi yang dikembalikan sama dengan versi untuk versi cluster Anda di tabel versi terbaru, maka Anda sudah menginstal versi terbaru di cluster Anda dan tidak perlu menyelesaikan sisa prosedur ini. Kubernetes Jika Anda menerima kesalahan, alih-alih nomor versi dalam output Anda, maka Anda tidak memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda. Anda perlu membuat add-on sebelum Anda dapat memperbaruinya dengan prosedur ini.
2. Simpan konfigurasi add-on yang Anda instal saat ini.

   ```
   kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
   ```
3. Perbarui add-on Anda menggunakan. AWS CLI Jika Anda ingin menggunakan AWS Management Console atau `eksctl` memperbarui add-on, lihatMemperbarui add-on. Salin perintah yang mengikuti ke perangkat Anda. Buat modifikasi berikut pada perintah, sesuai kebutuhan, dan kemudian jalankan perintah yang dimodifikasi.

   - Ganti *my-cluster* dengan nama klaster Anda.
   - Ganti *v1.18.0-eksbuild.1*dengan versi terbaru yang tercantum dalam tabel versi terbaru untuk versi cluster Anda.

- Ganti *111122223333* dengan ID akun Anda dan *AmazonKsvPCCnirole dengan nama peran* IAM yang ada yang telah Anda buat. Menentukan peran mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk klaster Anda. Untuk menentukan apakah Anda memiliki satu untuk cluster Anda, atau untuk membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda.
- Opsi **--resolve-conflicts** *PRESERVE mempertahankan* nilai konfigurasi yang ada untuk add-on. Jika Anda telah menetapkan nilai kustom untuk pengaturan add-on, dan Anda tidak menggunakan opsi ini, Amazon EKS menimpa nilai Anda dengan nilai defaultnya. Jika Anda menggunakan opsi ini, kami sarankan untuk menguji setiap bidang dan perubahan nilai pada klaster non-produksi sebelum memperbarui add-on pada cluster produksi Anda. Jika Anda mengubah nilai ini menjadi`OVERWRITE`, semua pengaturan diubah ke nilai default Amazon EKS. Jika Anda telah menetapkan nilai kustom untuk setelan apa pun, nilai tersebut mungkin akan ditimpa dengan nilai default Amazon EKS. Jika Anda mengubah nilai ininone, Amazon EKS tidak mengubah nilai pengaturan apa pun, tetapi pembaruan mungkin gagal. Jika pembaruan gagal, Anda menerima pesan galat untuk membantu menyelesaikan konflik.
- Jika Anda tidak memperbarui pengaturan konfigurasi, hapus **--configuration-values '{*"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}*}'**dari perintah. Jika Anda memperbarui pengaturan konfigurasi, ganti *“env”: {"AWS_VPC_K8S_CNI_EXTERNALSNAT*” :"true "} dengan pengaturan yang ingin Anda atur. Dalam contoh ini, variabel `AWS_VPC_K8S_CNI_EXTERNALSNAT` lingkungan diatur ke`true`. Nilai yang Anda tentukan harus valid untuk skema konfigurasi. Jika Anda tidak tahu skema konfigurasi, jalankan**aws eks describe-addon-configuration --addon-name vpc-cni --addon-version *v1.18.0-eksbuild.1***, ganti *v1.18.0-eksbuild.1* dengan nomor versi add-on yang ingin Anda lihat konfigurasinya. Skema dikembalikan dalam output. Jika Anda memiliki konfigurasi kustom yang ada, ingin menghapus semuanya, dan mengatur nilai untuk semua pengaturan kembali ke default Amazon EKS, hapus *“env”: {"AWS_VPC_K8S_CNI_EXTERNALSNAT” :"true "} dari perintah, sehingga Anda kosong*. **{}** Untuk penjelasan tentang setiap pengaturan, lihat Variabel Konfigurasi CNI pada GitHub.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.18.0-eksbuild.1 \
    --service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
 \
    --resolve-conflicts PRESERVE --configuration-values '{"env":
{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}}'
```

Mungkin perlu beberapa detik untuk pembaruan selesai.

4. Konfirmasikan bahwa versi add-on telah diperbarui. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

Mungkin perlu beberapa detik untuk pembaruan selesai.

Contoh output adalah sebagai berikut.

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.18.0-eksbuild.1",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/vpc-
cni/74c33d2f-b4dc-8718-56e7-9fdfa65d14a9",
        "createdAt": "2023-04-12T18:25:19.319000+00:00",
        "modifiedAt": "2023-04-12T18:40:28.683000+00:00",
        "serviceAccountRoleArn":
 "arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole",
        "tags": {},
        "configurationValues": "{\"env\":{\"AWS_VPC_K8S_CNI_EXTERNALSNAT\":\"true
\"}}"
    }
}
```

Memperbarui add-on yang dikelola sendiri

Important

Sebaiknya tambahkan jenis add-on Amazon EKS ke klaster Anda alih-alih menggunakan jenis add-on yang dikelola sendiri. Jika Anda tidak terbiasa dengan perbedaan antara jenis, lihatthe section called “Add-on Amazon EKS”. Untuk informasi selengkapnya tentang menambahkan add-on Amazon EKS ke klaster Anda, lihatthe section called “Membuat add-

on”. Jika Anda tidak dapat menggunakan add-on Amazon EKS, kami mendorong Anda untuk mengirimkan masalah tentang mengapa Anda tidak dapat ke repositori peta jalan GitHub Containers.

1. Konfirmasikan bahwa Anda tidak memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

   Jika pesan kesalahan dikembalikan, Anda tidak memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda. Untuk mengelola sendiri add-on, selesaikan langkah-langkah yang tersisa dalam prosedur ini untuk memperbarui add-on. Jika nomor versi dikembalikan, Anda memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda. Untuk memperbaruinya, gunakan prosedur diMemperbarui add-on, daripada menggunakan prosedur ini. Jika Anda tidak terbiasa dengan perbedaan antara jenis add-on, lihatAdd-on Amazon EKS.

2. Lihat versi gambar kontainer mana yang saat ini diinstal di cluster Anda.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
 | cut -d : -f 3
```

   Contoh output adalah sebagai berikut.

```
v1.12.6-eksbuild.2
```

   Output Anda mungkin tidak menyertakan nomor build.

3. Backup pengaturan Anda saat ini sehingga Anda dapat mengonfigurasi pengaturan yang sama setelah Anda memperbarui versi Anda.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

4. Untuk meninjau versi yang tersedia dan membiasakan diri dengan perubahan dalam versi yang ingin Anda perbarui, lihat `releases` diGitHub. Perhatikan bahwa kami sarankan memperbarui ke yang sama`major`. `minor`. `patch`versi yang tercantum dalam tabel versi terbaru yang tersedia, bahkan jika versi yang lebih baru tersedia di GitHub.. Versi build yang tercantum dalam

tabel tidak ditentukan dalam versi yang dikelola sendiri yang tercantum. GitHub Perbarui versi Anda dengan menyelesaikan tugas di salah satu opsi berikut:

- Jika Anda tidak memiliki pengaturan khusus untuk add-on, jalankan perintah di bawah To `apply this release:` judul GitHub untuk rilis yang Anda perbarui.
- Jika Anda memiliki pengaturan khusus, unduh file manifes dengan perintah berikut. Ubah *`https://raw.githubusercontent.com/aws/ amazon-vpc-cni-k 8s/v1.18.0/config/master/aws-k8s-cni.yaml`* menjadi URL untuk rilis yang Anda perbarui. GitHub

```
curl -O https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/v1.18.0/config/master/aws-k8s-cni.yaml
```

Jika perlu, ubah manifes dengan pengaturan kustom dari cadangan yang Anda buat pada langkah sebelumnya, lalu terapkan manifes yang dimodifikasi ke klaster Anda. Jika node Anda tidak memiliki akses ke repositori Amazon EKS Amazon ECR pribadi tempat gambar ditarik (lihat baris yang dimulai dengan `image:` manifes), maka Anda harus mengunduh gambar, menyalinnya ke repositori Anda sendiri, dan memodifikasi manifes untuk menarik gambar dari repositori Anda. Untuk informasi selengkapnya, lihat Salin gambar kontainer dari satu repositori ke repositori lain.

```
kubectl apply -f aws-k8s-cni.yaml
```

5. Konfirmasikan bahwa versi baru sekarang diinstal pada cluster Anda.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
 | cut -d : -f 3
```

Contoh output adalah sebagai berikut.

```
v1.18.0
```

6. (Opsional) Instal `cni-metrics-helper` ke cluster Anda. Ini mengikis elastic network interface dan informasi alamat IP, menggabungkannya pada tingkat cluster, dan menerbitkan metrik ke Amazon. CloudWatch Untuk informasi lebih lanjut, lihat cni-metrics-helperdi GitHub.

Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA)

Amazon VPC CNI plugin for KubernetesIni adalah plugin jaringan untuk Pod jaringan di kluster Amazon EKS. Plugin ini bertanggung jawab untuk mengalokasikan alamat IP VPC ke Kubernetes node dan mengkonfigurasi jaringan yang diperlukan untuk setiap node. Pods Plugin:

- Memerlukan izin AWS Identity and Access Management (IAM). Jika klaster Anda menggunakan keluarga IPv4, izin akan ditentukan dalam kebijakan terkelola. `AmazonEKS_CNI_Policy` AWS Jika klaster Anda menggunakan keluarga IPv6, izin harus ditambahkan ke kebijakan IAM yang Anda buat.Anda dapat melampirkan kebijakan ke peran IAM node Amazon EKS, atau ke peran IAM terpisah. Kami menyarankan Anda menetapkannya ke peran terpisah, seperti yang dijelaskan dalam topik ini.
- Membuat dan dikonfigurasi untuk menggunakan akun Kubernetes layanan bernama `aws-node` saat digunakan. Akun layanan terikat pada Kubernetes `clusterrole` namaaws-node, yang diberi Kubernetes izin yang diperlukan.

> **Note**
>
> PodsUntuk Amazon VPC CNI plugin for Kubernetes memiliki akses ke izin yang ditetapkan ke peran IAM node Amazon EKS, kecuali Anda memblokir akses ke IMDS. Untuk informasi selengkapnya, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.

Prasyarat

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS.
- Penyedia AWS Identity and Access Management (IAM) OpenID Connect (OIDC) yang sudah ada untuk cluster Anda. Untuk menentukan apakah Anda sudah memiliki satu, atau harus membuat satu, lihat Buat OIDC penyedia IAM untuk klaster Anda.

Langkah 1: Buat peran Amazon VPC CNI plugin for Kubernetes IAM

Untuk membuat peran IAM

1. Tentukan keluarga IP cluster Anda.

```
aws eks describe-cluster --name my-cluster | grep ipFamily
```

Contoh output adalah sebagai berikut.

```
"ipFamily": "ipv4"
```

Output dapat kembali `ipv6` sebagai gantinya.

2. Buat peran IAM. Anda dapat menggunakan `eksctl` atau `kubectl` dan AWS CLI untuk membuat peran IAM Anda.

   eksctl

   Buat peran IAM dan lampirkan kebijakan IAM ke peran dengan perintah yang cocok dengan keluarga IP cluster Anda. Perintah membuat dan menerapkan AWS CloudFormation tumpukan yang membuat peran IAM, melampirkan kebijakan yang Anda tentukan padanya, dan membuat anotasi `aws-node` Kubernetes akun layanan yang ada dengan ARN peran IAM yang dibuat.

   - `IPv4`

     Ganti *my-cluster* dengan nilai Anda sendiri.

     ```
     eksctl create iamserviceaccount \
         --name aws-node \
         --namespace kube-system \
         --cluster my-cluster \
         --role-name AmazonEKSVPCCNIRole \
         --attach-policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
         --override-existing-serviceaccounts \
         --approve
     ```

   - `IPv6`

     Ganti *my-cluster* dengan nilai Anda sendiri. Ganti *111122223333* dengan ID akun Anda dan ganti *AmazonEKS_CNI_IPv6_Policy* dengan nama `IPv6` polis Anda. Jika Anda tidak memiliki `IPv6` kebijakan, lihat Buat kebijakan IAM untuk cluster yang menggunakan keluarga `IPv6` untuk membuatnya. Untuk digunakan `IPv6` dengan cluster Anda, itu harus memenuhi beberapa persyaratan. Untuk informasi selengkapnya, lihat `IPv6`alamat untuk cluster,Pods, dan services.

```
eksctl create iamserviceaccount \
    --name aws-node \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKSVPCCNIRole \
    --attach-policy-arn
 arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
    --override-existing-serviceaccounts \
    --approve
```

kubectl and the AWS CLI

1. Lihat URL penyedia OIDC klaster Anda.

   ```
   aws eks describe-cluster --name my-cluster --query
    "cluster.identity.oidc.issuer" --output text
   ```

   Contoh output adalah sebagai berikut.

   ```
   https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
   ```

   Jika tidak ada output yang dikembalikan, maka Anda harus membuat penyedia IAM OIDC untuk cluster Anda.

2. Salin isi berikut ke file bernama *vpc-cni-trust-policy.json*. Ganti *111122223333* dengan ID akun Anda dan *EXAMPLED539D4633E53DE1B71EXAMPLE* dengan output yang dikembalikan pada langkah sebelumnya. Ganti *region-code* dengan tempat Wilayah AWS cluster Anda berada.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Principal": {
                   "Federated": "arn:aws:iam::111122223333:oidc-provider/
   oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
               },
               "Action": "sts:AssumeRoleWithWebIdentity",
   ```

```
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:aws-node"
                }
            }
        }
    ]
}
```

3. Buat peran. Anda dapat mengganti *AmazonEKSVPCCNIRole* dengan nama apa pun yang Anda pilih.

```
aws iam create-role \
  --role-name AmazonEKSVPCCNIRole \
  --assume-role-policy-document file://"vpc-cni-trust-policy.json"
```

4. Lampirkan kebijakan IAM yang diperlukan ke peran tersebut. Jalankan perintah yang cocok dengan keluarga IP cluster Anda.

   - IPv4

     ```
     aws iam attach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
       --role-name AmazonEKSVPCCNIRole
     ```

   - IPv6

     Ganti *111122223333* dengan ID akun Anda dan *AmazonEKS_CNI_IPv6_Policy* dengan nama IPv6 polis Anda. Jika Anda tidak memiliki IPv6 kebijakan, lihat Buat kebijakan IAM untuk cluster yang menggunakan keluarga IPv6 untuk membuatnya. Untuk digunakan IPv6 dengan cluster Anda, itu harus memenuhi beberapa persyaratan. Untuk informasi selengkapnya, lihat IPv6alamat untuk cluster,Pods, dan services.

     ```
     aws iam attach-role-policy \
       --policy-arn arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
       --role-name AmazonEKSVPCCNIRole
     ```

5. Jalankan perintah berikut untuk membuat anotasi akun `aws-node` layanan dengan ARN dari peran IAM yang Anda buat sebelumnya. Ganti *`example values`* dengan nilai-nilai milik Anda sendiri.

```
kubectl annotate serviceaccount \
    -n kube-system aws-node \
    eks.amazonaws.com/role-
arn=arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
```

3. (Opsional) Konfigurasikan jenis AWS Security Token Service titik akhir yang digunakan oleh akun Kubernetes layanan Anda. Untuk informasi selengkapnya, lihat Konfigurasikan AWS Security Token Service titik akhir untuk akun layanan.

Langkah 2: Menyebarkan kembali Amazon VPC CNI plugin for KubernetesPods

1. Hapus dan buat ulang semua Pods yang ada yang terkait dengan akun layanan untuk menerapkan variabel lingkungan kredensi. Anotasi tidak diterapkan pada Pods yang sedang berjalan tanpa anotasi. Perintah berikut menghapus yang ada `aws-node` DaemonSet Pods dan menerapkannya dengan anotasi akun layanan.

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

2. Konfirmasikan bahwa Pods semua dimulai ulang.

```
kubectl get pods -n kube-system -l k8s-app=aws-node
```

3. Jelaskan salah satu Pods dan verifikasi bahwa variabel `AWS_WEB_IDENTITY_TOKEN_FILE` dan `AWS_ROLE_ARN` lingkungan ada. Ganti *`cpjw7`* dengan nama salah satu yang Anda Pods kembalikan dalam output dari langkah sebelumnya.

```
kubectl describe pod -n kube-system aws-node-cpjw7 | grep 'AWS_ROLE_ARN:\|
AWS_WEB_IDENTITY_TOKEN_FILE:'
```

Contoh output adalah sebagai berikut.

```
AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
      AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
```

```
      AWS_ROLE_ARN:
 arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
      AWS_WEB_IDENTITY_TOKEN_FILE:           /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
```

Dua set hasil duplikat dikembalikan karena Pod berisi dua kontainer. Kedua kontainer memiliki nilai yang sama.

Jika Anda Pod menggunakan titik akhir Wilayah AWS al, maka baris berikut juga dikembalikan pada output sebelumnya.

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

Langkah 3: Hapus kebijakan CNI dari peran IAM node

Jika peran IAM node Amazon EKS saat ini memiliki kebijakan `AmazonEKS_CNI_Policy` IAM (`IPv4`) atau `IPv6`kebijakan yang dilampirkan padanya, dan Anda telah membuat peran IAM terpisah, melampirkan kebijakan ke sana, dan menetapkannya ke akun `aws-node` Kubernetes layanan, maka sebaiknya Anda menghapus kebijakan tersebut dari peran node dengan AWS CLI perintah yang cocok dengan keluarga IP klaster Anda. Ganti *`AmazonEKSNodeRole`* dengan nama peran node Anda.

- `IPv4`

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn
 arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
```

- `IPv6`

  Ganti *`111122223333`* dengan ID akun Anda dan *`AmazonEKS_CNI_IPv6_Policy`* dengan nama `IPv6` polis Anda.

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn
 arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy
```

Buat kebijakan IAM untuk cluster yang menggunakan keluarga **`IPv6`**

Jika Anda membuat klaster yang menggunakan `IPv6` keluarga dan klaster memiliki versi `1.10.1` atau yang lebih baru dari Amazon VPC CNI plugin for Kubernetes add-on yang dikonfigurasi, maka Anda perlu membuat kebijakan IAM yang dapat Anda tetapkan ke peran IAM. Jika Anda memiliki klaster yang sudah ada yang tidak Anda konfigurasikan dengan `IPv6` keluarga saat Anda membuatnya, maka untuk menggunakannya`IPv6`, Anda harus membuat cluster baru. Untuk informasi selengkapnya tentang penggunaan `IPv6` dengan klaster Anda, lihat[IPv6alamat untuk cluster,Pods, dan services](#).

1. Salin teks berikut dan simpan ke file bernama *`vpc-cni-ipv6-policy`*`.json`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
```

2. Buat kebijakan IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-document
 file://vpc-cni-ipv6-policy.json
```

Memilih kasus penggunaan Pod jaringan

Amazon VPC CNI plugin for KubernetesMenyediakan jaringan untukPods. Tabel berikut membantu Anda memahami kasus penggunaan jaringan mana yang dapat Anda gunakan bersama serta kemampuan serta Amazon VPC CNI plugin for Kubernetes pengaturan yang dapat Anda gunakan dengan berbagai jenis simpul Amazon EKS. Semua informasi dalam tabel hanya berlaku untuk Linux `IPv4` node.

Jenis simpul Amazon EKS	Amazon EC2			Fargate
Kasus penggunaan	Alamat IP individual yang ditetapkan untuk antarmuka jaringan	Awalan IP ditetapkan untuk antarmuka jaringan	Kelompok keamanan untuk Pods	
Jaringan khusus untuk pod—Tetapkan alamat IP dari subnet yang berbeda dari subnet node	Ya	Ya	Ya	Ya (subnet dikendalikan melalui profil Fargate)
SNAT untuk Pods	Ya (defaultnya adalah`false`)	Ya (defaultnya adalah`false`)	Ya (`true`hanya)	Ya (`true`hanya)
Kemampuan				
Ruang lingkup kelompok keamanan	Simpul	Simpul	Pod (Jika Anda telah menetapkan `POD_SECURITY_GROUP_ENFORCING_MODE = standard` dan `AWS_VPC_K`	Pod

Jenis simpul Amazon EKS	Amazon EC2			Fargate
Kasus penggunaan	Alamat IP individual yang ditetapkan untuk antarmuka jaringan	Awalan IP ditetapkan untuk antarmuka jaringan	Kelompok keamanan untuk Pods	
			`8S_CNI_EXTERNALSNAT =false`, lalu lintas yang ditujukan untuk titik akhir di luar VPC menggunakan grup keamanan node, bukan grup keamanan) Pod's	
Jenis subnet Amazon VPC	Swasta dan publik	Swasta dan publik	Hanya pribadi	Hanya pribadi
Kebijakan jaringan (VPC CNI)	Kompatibel	Kompatibel	Kompatibel Hanya dengan versi `1.14.0` atau yang lebih baru dari plugin Amazon VPC CNI	Tidak didukung
Kepadatan pod per node	Sedang	Tinggi	Rendah	Satu

Jenis simpul Amazon EKS	Amazon EC2			Fargate
Kasus penggunaan	Alamat IP individual yang ditetapkan untuk antarmuka jaringan	Awalan IP ditetapkan untuk antarmuka jaringan	Kelompok keamanan untuk Pods	
Waktu peluncuran Pod	Lebih baik	Terbaik	Baik	Sedang
Pengaturan plugin Amazon VPC CNI (untuk informasi selengkapnya tentang setiap setelan, lihat amazon-vpc-cni-k 8s aktif) GitHub				
`WARM_ENI_TARGET`	Ya	Tidak berlaku	Tidak berlaku	Tidak berlaku
`WARM_IP_TARGET`	Ya	Ya	Tidak berlaku	Tidak berlaku
`MINIMUM_IP_TARGET`	Ya	Ya	Tidak berlaku	Tidak berlaku
`WARM_PREFIX_TARGET`	Tidak berlaku	Ya	Tidak berlaku	Tidak berlaku

Note

- Anda tidak dapat menggunakan `IPv6` dengan jaringan khusus.
- `IPv6`alamat tidak diterjemahkan, jadi SNAT tidak berlaku.
- Arus lalu lintas ke dan dari Pods dengan grup keamanan terkait tidak dikenakan penegakan kebijakan Calico jaringan dan terbatas pada penegakan kelompok keamanan Amazon VPC saja.
- Jika Anda menggunakan penegakan kebijakan Calico jaringan, kami sarankan Anda menyetel variabel lingkungan `ANNOTATE_POD_IP` `true` untuk menghindari masalah yang

diketahuiKubernetes. Untuk menggunakan fitur ini, Anda harus menambahkan `patch` izin untuk pod ke file `aws-node`ClusterRole. Perhatikan bahwa menambahkan izin tambalan ke `aws-node` DaemonSet meningkatkan cakupan keamanan untuk plugin. Untuk informasi selengkapnya, lihat ANNOTATE_POD_IP di repo VPC CNI aktif. GitHub

- Awalan IP dan alamat IP dikaitkan dengan antarmuka jaringan elastis Amazon EC2 standar. Pod yang membutuhkan grup keamanan tertentu diberi alamat IP utama dari antarmuka jaringan cabang. Anda dapat mencampur Pods mendapatkan alamat IP, atau alamat IP dari awalan IP dengan Pods mendapatkan antarmuka jaringan cabang pada node yang sama.

Simpul Windows

Setiap node hanya mendukung satu antarmuka jaringan. Anda dapat menggunakan `IPv4` alamat dan `IPv4` awalan sekunder. Secara default, jumlah `IPv4` alamat yang tersedia pada node sama dengan jumlah `IPv4` alamat sekunder yang dapat Anda tetapkan ke setiap elastic network interface, dikurangi satu. Namun, Anda dapat meningkatkan `IPv4` alamat dan Pod kepadatan yang tersedia pada node dengan mengaktifkan awalan IP. Untuk informasi selengkapnya, lihat Tingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda.

Calicokebijakan jaringan didukung padaWindows. Anda tidak dapat menggunakan grup keamanan untuk Pods atau jaringan khusus diWindows.

`IPv6`alamat untuk cluster,Pods, dan services

Secara default, Kubernetes berikan `IPv4` alamat ke alamat Anda Pods danservices. Alih-alih menetapkan `IPv4` alamat ke Pods danservices, Anda dapat mengonfigurasi klaster Anda untuk menetapkan `IPv6` alamat kepada mereka. Amazon EKS tidak mendukung dual-stacked Pods atauservices, meskipun Kubernetes dalam versi `1.23` dan yang lebih baru. Akibatnya, Anda tidak dapat menetapkan keduanya `IPv4` dan `IPv6` alamat ke Pods danservices.

Anda memilih keluarga IP mana yang ingin Anda gunakan untuk cluster Anda saat Anda membuatnya. Anda tidak dapat mengubah keluarga setelah Anda membuat cluster.

Pertimbangan untuk menggunakan **`IPv6`** keluarga untuk cluster Anda

- Anda harus membuat cluster baru dan menentukan bahwa Anda ingin menggunakan `IPv6` keluarga untuk cluster itu. Anda tidak dapat mengaktifkan `IPv6` keluarga untuk klaster yang Anda

perbarui dari versi sebelumnya. Untuk petunjuk tentang cara membuat klaster baru, lihat Membuat klaster Amazon EKS.

- Versi add-on Amazon VPC CNI yang Anda terapkan ke cluster Anda harus versi atau yang lebih baru. `1.10.1` Versi ini atau yang lebih baru digunakan secara default. Setelah menerapkan add-on, Anda tidak dapat menurunkan versi add-on Amazon VPC CNI ke versi yang lebih rendah daripada `1.10.1` tanpa terlebih dahulu menghapus semua node di semua grup node di cluster Anda.
- WindowsPodsdan services tidak didukung.
- Jika Anda menggunakan node Amazon EC2, Anda harus mengonfigurasi add-on Amazon VPC CNI dengan delegasi awalan IP dan. `IPv6` Jika Anda memilih `IPv6` keluarga saat membuat cluster Anda, `1.10.1` versi add-on default ke konfigurasi ini. Ini adalah kasus untuk add-on Amazon EKS yang dikelola sendiri atau Amazon. Untuk informasi selengkapnya tentang delegasi awalan IP, lihat. Tingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda
- Saat Anda membuat klaster, VPC dan subnet yang Anda tentukan harus memiliki blok `IPv6` CIDR yang ditetapkan ke VPC dan subnet yang Anda tentukan. Mereka juga harus memiliki blok `IPv4` CIDR yang ditugaskan untuk mereka. Ini karena, bahkan jika Anda hanya ingin menggunakan`IPv6`, VPC masih memerlukan blok `IPv4` CIDR untuk berfungsi. Untuk informasi selengkapnya, lihat Mengaitkan blok `IPv6` CIDR dengan VPC Anda di Panduan Pengguna Amazon VPC.
- Saat Anda membuat cluster dan node, Anda harus menentukan subnet yang dikonfigurasi untuk menetapkan alamat secara otomatis`IPv6`. Jika tidak, Anda tidak dapat menerapkan cluster dan node Anda. Secara default, konfigurasi ini dinonaktifkan. Untuk informasi selengkapnya, lihat Memodifikasi atribut `IPv6` pengalamatan untuk subnet Anda di Panduan Pengguna Amazon VPC.
- Tabel rute yang ditetapkan ke subnet Anda harus memiliki rute untuk `IPv6` alamat. Untuk informasi selengkapnya, lihat Memigrasi ke `IPv6` dalam Panduan Pengguna Amazon VPC.
- Grup keamanan Anda harus mengizinkan `IPv6` alamat. Untuk informasi selengkapnya, lihat Memigrasi ke `IPv6` dalam Panduan Pengguna Amazon VPC.
- Anda hanya dapat menggunakan `IPv6` dengan node Amazon EC2 atau Fargate AWS berbasis Nitro.
- Anda tidak dapat menggunakan `IPv6` Kelompok keamanan untuk Pods dengan node Amazon EC2. Namun, Anda dapat menggunakannya dengan node Fargate. Jika Anda memerlukan grup keamanan terpisah untuk individuPods, lanjutkan menggunakan `IPv4` keluarga dengan node Amazon EC2, atau gunakan node Fargate sebagai gantinya.

- Jika sebelumnya Anda menggunakan jaringan khusus untuk membantu mengurangi kelelahan alamat IP, Anda dapat menggunakannya sebagai gantinya. IPv6 Anda tidak dapat menggunakan jaringan khusus denganIPv6. Jika Anda menggunakan jaringan khusus untuk isolasi jaringan, maka Anda mungkin perlu terus menggunakan jaringan khusus dan IPv4 keluarga untuk cluster Anda.
- Anda tidak dapat menggunakan IPv6 dengan AWS Outposts.
- Podsdan hanya services diberi IPv6 alamat. Mereka tidak diberi IPv4 alamat. Karena Pods dapat berkomunikasi ke IPv4 titik akhir melalui NAT pada instance itu sendiri, DNS64 dan NAT64 tidak diperlukan. Jika lalu lintas membutuhkan alamat IP publik, lalu lintas kemudian alamat jaringan sumber diterjemahkan ke IP publik.
- IPv6Alamat sumber dari alamat jaringan Pod bukan sumber yang diterjemahkan ke IPv6 alamat node saat berkomunikasi di luar VPC. Ini dirutekan menggunakan gateway internet atau gateway internet khusus egress.
- Semua node diberi IPv6 alamat IPv4 dan.
- Driver CSI Amazon FSx for Lustre tidak didukung.
- Anda dapat menggunakan versi 2.3.1 atau yang lebih baru dari AWS Load Balancer Controller untuk memuat aplikasi keseimbangan atau lalu lintas jaringan ke IPv6 Pods dalam mode IP, tetapi bukan mode instance. Untuk informasi selengkapnya, lihat Apa itu AWS Load Balancer Controller?.
- Anda harus melampirkan kebijakan IPv6 IAM ke peran IAM node atau CNI IAM Anda. Di antara keduanya, kami sarankan Anda melampirkannya ke peran IAM CNI. Lihat informasi yang lebih lengkap di Buat kebijakan IAM untuk cluster yang menggunakan keluarga IPv6 dan Langkah 1: Buat peran Amazon VPC CNI plugin for Kubernetes IAM.
- Setiap Fargate Pod menerima IPv6 alamat dari CIDR yang ditentukan untuk subnet tempat ia digunakan. Unit perangkat keras yang mendasari yang menjalankan Fargate Pods mendapatkan IPv6 alamat unik IPv4 dan dari CIDR yang ditugaskan ke subnet tempat unit perangkat keras digunakan.
- Kami menyarankan Anda melakukan evaluasi menyeluruh terhadap aplikasi, add-on Amazon EKS, dan AWS layanan yang Anda integrasikan sebelum menerapkan clusterIPv6. Ini untuk memastikan bahwa semuanya berfungsi seperti yang diharapkanIPv6.
- Penggunaan IPv6 titik akhir Layanan Metadata Instans Amazon EC2 tidak didukung dengan Amazon EKS.
- Saat membuat grup node yang dikelola sendiri dalam cluster yang menggunakan IPv6 keluarga, data pengguna harus menyertakan yang berikut BootstrapArguments untuk bootstrap.shfile

yang berjalan saat node start up. *Ganti cidr* Anda dengan IPv6 CIDR rentang VPC cluster Anda.

```
--ip-family ipv6 --service-ipv6-cidr your-cidr
```

Jika Anda tidak tahu IPv6 CIDR rentang untuk cluster Anda, Anda dapat melihatnya dengan perintah berikut (memerlukan AWS CLI versi 2.4.9 atau yang lebih baru).

```
aws eks describe-cluster --name my-cluster --query
 cluster.kubernetesNetworkConfig.serviceIpv6Cidr --output text
```

Menerapkan **IPv6** cluster dan mengelola node Amazon Linux

Dalam tutorial ini, Anda menerapkan VPC IPv6 Amazon, kluster Amazon EKS bersama IPv6 keluarga, dan grup node terkelola dengan node Amazon EC2 Amazon Linux. Anda tidak dapat menerapkan node Amazon Windows EC2 di IPv6 klaster. Anda juga dapat menerapkan node Fargate ke cluster Anda, meskipun instruksi tersebut tidak disediakan dalam topik ini untuk kesederhanaan.

Sebelum membuat cluster untuk penggunaan produksi, kami sarankan Anda membiasakan diri dengan semua pengaturan dan menyebarkan cluster dengan pengaturan yang memenuhi persyaratan Anda. Untuk informasi lebih lanjut, lihatMembuat klaster Amazon EKS, Grup simpul terkelola dan pertimbangan untuk topik ini. Anda hanya dapat mengaktifkan beberapa pengaturan saat membuat cluster Anda.

Prasyarat

Sebelum memulai tutorial ini, Anda harus menginstal dan mengonfigurasi alat-alat dan sumber daya yang Anda butuhkan berikut untuk membuat dan mengelola sebuah klaster Amazon EKS.

- Alat baris kubectl perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda1.28, Anda dapat menggunakan kubectl versi1.27,1.28, atau 1.29 dengan itu. Untuk menginstal atau memutakhirkan kubectl, lihat Menginstal atau memperbarui kubectl.
- Prinsip keamanan IAM yang Anda gunakan harus memiliki izin untuk bekerja dengan peran Amazon EKS IAM, peran terkait layanan,, VPC, AWS CloudFormation dan sumber daya terkait.

Untuk informasi selengkapnya, lihat Kunci tindakan, sumber daya, dan kondisi untuk Amazon Elastic Kubernetes Service dan Menggunakan peran terkait layanan di Panduan Pengguna IAM.

Prosedur disediakan untuk membuat sumber daya dengan salah satu `eksctl` atau AWS CLI. Anda juga dapat menerapkan sumber daya menggunakan AWS Management Console, tetapi instruksi tersebut tidak disediakan dalam topik ini untuk kesederhanaan.

eksctl

Prasyarat

`eksctl`versi `0.175.0` atau yang lebih baru diinstal pada komputer Anda. Untuk menginstal atau memperbaruinya, lihat Instalasi di `eksctl` dokumentasi.

Untuk menyebarkan **IPv6** cluster dengan **`eksctl`**

1. Buat file `ipv6-cluster.yaml`. Salin perintah yang mengikuti ke perangkat Anda. Buat modifikasi berikut pada perintah sesuai kebutuhan dan kemudian jalankan perintah yang dimodifikasi:
   - Ganti *`my-cluster`* dengan nama untuk cluster Anda. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter.
   - Ganti *`region-code`* dengan apa pun Wilayah AWS yang didukung oleh Amazon EKS. Untuk daftar Wilayah AWS, lihat titik akhir dan kuota Amazon EKS di panduan Referensi AWS Umum.
   - Nilai untuk `version` dengan versi cluster Anda. Untuk informasi selengkapnya, lihat Kubernetesversi Amazon EKS yang didukung.
   - Ganti *`my-nodegroup`* dengan nama untuk grup node Anda. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa.
   - Ganti *`t3.medium`* dengan jenis instans Sistem AWS Nitro apa pun.

```
cat >ipv6-cluster.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
```

```
metadata:
  name: my-cluster
  region: region-code
  version: "X.XX"

kubernetesNetworkConfig:
  ipFamily: IPv6

addons:
  - name: vpc-cni
    version: latest
  - name: coredns
    version: latest
  - name: kube-proxy
    version: latest

iam:
  withOIDC: true

managedNodeGroups:
  - name: my-nodegroup
    instanceType: t3.medium
EOF
```

2. Buat cluster Anda.

```
eksctl create cluster -f ipv6-cluster.yaml
```

Pembuatan cluster membutuhkan waktu beberapa menit. Jangan lanjutkan sampai Anda melihat baris output terakhir, yang terlihat mirip dengan output berikut.

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

3. Konfirmasikan bahwa default Pods adalah `IPv6` alamat yang ditetapkan.

```
kubectl get pods -n kube-system -o wide
```

Contoh output adalah sebagai berikut.

```
NAME                       READY   STATUS    RESTARTS   AGE     IP
                             NODE
 NOMINATED NODE   READINESS GATES
aws-node-rslts             1/1     Running   1          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
aws-node-t74jh             1/1     Running   0          5m32s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-cw7w2   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::                ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-tx6n8   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::1               ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
kube-proxy-btpbk           1/1     Running   0          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
kube-proxy-jjk2g           1/1     Running   0          5m33s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

4. Konfirmasikan bahwa layanan default adalah IPv6 alamat yang ditetapkan.

```
kubectl get services -n kube-system -o wide
```

Contoh output adalah sebagai berikut.

```
NAME       TYPE        CLUSTER-IP          EXTERNAL-IP   PORT(S)         AGE
 SELECTOR
kube-dns   ClusterIP   fd30:3087:b6c2::a   <none>        53/UDP,53/TCP   57m
 k8s-app=kube-dns
```

5. (Opsional) Menyebarkan aplikasi sampel atau menyebarkan AWS Load Balancer Controllerdan aplikasi sampel untuk memuat aplikasi keseimbangan atau lalu lintas jaringan ke. IPv6 Pods

6. Setelah Anda selesai dengan cluster dan node yang Anda buat untuk tutorial ini, Anda harus membersihkan sumber daya yang Anda buat dengan perintah berikut.

```
eksctl delete cluster my-cluster
```

AWS CLI

Prasyarat

Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna. Jika Anda menggunakan AWS CloudShell, Anda mungkin perlu menginstal versi `2.12.3` atau yang lebih baru `1.27.160` atau atau yang lebih baru AWS CLI, karena AWS CLI versi default yang diinstal di AWS CloudShell mungkin versi sebelumnya.

Important

- Anda harus menyelesaikan semua langkah dalam prosedur ini sebagai pengguna yang sama. Untuk memeriksa pengguna saat ini, jalankan perintah berikut:

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

- Anda harus menyelesaikan semua langkah dalam prosedur ini di shell yang sama. Beberapa langkah menggunakan variabel yang diatur dalam langkah sebelumnya. Langkah-langkah yang menggunakan variabel tidak akan berfungsi dengan baik jika nilai variabel diatur dalam shell yang berbeda. Jika Anda menggunakan AWS CloudShelluntuk menyelesaikan prosedur berikut, ingatlah bahwa jika Anda tidak berinteraksi dengannya menggunakan keyboard atau pointer selama kurang lebih 20-30 menit, sesi shell Anda berakhir. Proses yang berjalan tidak dihitung sebagai interaksi.
- Instruksi ditulis untuk cangkang Bash, dan mungkin perlu disesuaikan dengan cangkang lainnya.

Untuk membuat cluster Anda dengan AWS CLI

Ganti semua *example values* langkah prosedur ini dengan nilai Anda sendiri.

1. Jalankan perintah berikut untuk mengatur beberapa variabel yang digunakan dalam langkah-langkah selanjutnya. Ganti *region-code* dengan tempat Wilayah AWS Anda ingin menyebarkan sumber daya Anda. Nilainya bisa berupa apa saja Wilayah AWS yang didukung oleh Amazon EKS. Untuk daftar Wilayah AWS, lihat titik akhir dan kuota Amazon EKS di panduan Referensi AWS Umum. Ganti *my-cluster* dengan nama untuk cluster Anda. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter. Ganti *my-nodegroup* dengan nama untuk grup node Anda. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa. Ganti *111122223333* dengan ID akun Anda.

```
export region_code=region-code
export cluster_name=my-cluster
export nodegroup_name=my-nodegroup
export account_id=111122223333
```

2. Buat VPC Amazon dengan subnet publik dan pribadi yang memenuhi Amazon EKS dan persyaratan. IPv6

   a. Jalankan perintah berikut untuk mengatur variabel untuk nama AWS CloudFormation tumpukan Anda. Anda dapat mengganti *my-eks-ipv6-vpc* dengan nama apa pun yang Anda pilih.

   ```
   export vpc_stack_name=my-eks-ipv6-vpc
   ```

   b. Buat IPv6 VPC menggunakan template. AWS CloudFormation

   ```
   aws cloudformation create-stack --region $region_code --stack-name
    $vpc_stack_name \
     --template-url https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/amazon-eks-ipv6-vpc-public-private-
   subnets.yaml
   ```

Tumpukan membutuhkan waktu beberapa menit untuk membuatnya. Jalankan perintah berikut. Jangan melanjutkan ke langkah berikutnya sampai output dari perintah tersebut`CREATE_COMPLETE`.

```
aws cloudformation describe-stacks --region $region_code --stack-name
 $vpc_stack_name --query Stacks[].StackStatus --output text
```

c. Ambil ID subnet publik yang dibuat.

```
aws cloudformation describe-stacks --region $region_code --stack-name
 $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPublic`].OutputValue' --
output text
```

Contoh output adalah sebagai berikut.

```
subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE
```

d. Aktifkan opsi `IPv6` alamat tetapkan otomatis untuk subnet publik yang dibuat.

```
aws ec2 modify-subnet-attribute --region $region_code --
subnet-id subnet-0a1a56c486EXAMPLE --assign-ipv6-address-on-
creation
aws ec2 modify-subnet-attribute --region $region_code --subnet-id
 subnet-099e6ca77aEXAMPLE --assign-ipv6-address-on-creation
```

e. Ambil nama subnet dan grup keamanan yang dibuat oleh template dari AWS CloudFormation tumpukan yang digunakan dan simpan dalam variabel untuk digunakan di langkah selanjutnya.

```
security_groups=$(aws cloudformation describe-stacks --region $region_code
 --stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SecurityGroups`].OutputValue' --
output text)

public_subnets=$(aws cloudformation describe-stacks --region $region_code --
stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPublic`].OutputValue' --
output text)
```

```
private_subnets=$(aws cloudformation describe-stacks --region $region_code
 --stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPrivate`].OutputValue' --
output text)

subnets=${public_subnets},${private_subnets}
```

3. Buat peran IAM cluster dan lampirkan kebijakan terkelola Amazon EKS IAM yang diperlukan ke dalamnya. Kubernetescluster yang dikelola oleh Amazon EKS melakukan panggilan ke AWS layanan lain atas nama Anda untuk mengelola sumber daya yang Anda gunakan dengan layanan.

   a. Jalankan perintah berikut untuk membuat `eks-cluster-role-trust-policy.json` file.

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

   b. Jalankan perintah berikut untuk menetapkan variabel untuk nama peran Anda. Anda dapat mengganti *myAmazonEKSClusterRole* dengan nama apa pun yang Anda pilih.

   ```
   export cluster_role_name=myAmazonEKSClusterRole
   ```

   c. Buat peran.

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

   d. Ambil ARN dari peran IAM dan simpan dalam variabel untuk langkah selanjutnya.

```
cluster_iam_role=$(aws iam get-role --role-name $cluster_role_name --
query="Role.Arn" --output text)
```

e. Lampirkan kebijakan terkelola IAM Amazon EKS yang diperlukan untuk peran tersebut.

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

4. Buat cluster Anda.

```
aws eks create-cluster --region $region_code --name $cluster_name --kubernetes-
version 1.XX \
   --role-arn $cluster_iam_role --resources-vpc-config subnetIds=
$subnets,securityGroupIds=$security_groups \
   --kubernetes-network-config ipFamily=ipv6
```

-

> **Note**
>
> Mungkin error akan terjadi karena salah satu Availability Zone dalam permintaan Anda tidak memiliki kapasitas yang cukup untuk membuat klaster Amazon EKS. Jika hal ini terjadi, output galat berisi Availability Zones yang dapat mendukung klaster baru. Cobalah untuk kembali membuat klaster dengan setidaknya dua subnet yang terletak di Availability Zones yang didukung untuk akun Anda. Untuk informasi selengkapnya, lihat Kapasitas tidak mencukupi.

Cluster membutuhkan waktu beberapa menit untuk membuatnya. Jalankan perintah berikut. Jangan lanjutkan ke langkah berikutnya sampai output dari perintah tersebut`ACTIVE`.

```
aws eks describe-cluster --region $region_code --name $cluster_name --query
 cluster.status
```

5. Buat atau perbarui `kubeconfig` file untuk klaster Anda sehingga Anda dapat berkomunikasi dengan cluster Anda.

```
aws eks update-kubeconfig --region $region_code --name $cluster_name
```

Secara default, file `config` dibuat di `~/.kube` atau konfigurasi klaster baru ditambahkan ke file `config` yang sudah ada di `~/.kube`.

6. Buat peran IAM node.

   a. Jalankan perintah berikut untuk membuat `vpc-cni-ipv6-policy.json` file.

```
cat >vpc-cni-ipv6-policy <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
EOF
```

   b. Buat kebijakan IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-
document file://vpc-cni-ipv6-policy.json
```

   c. Jalankan perintah berikut untuk membuat `node-role-trust-relationship.json` file.

```
cat >node-role-trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

d. Jalankan perintah berikut untuk menetapkan variabel untuk nama peran Anda. Anda dapat mengganti *AmazonEKSNodeRole* dengan nama apa pun yang Anda pilih.

```
export node_role_name=AmazonEKSNodeRole
```

e. Buat peran IAM.

```
aws iam create-role --role-name $node_role_name --assume-role-policy-
document file://"node-role-trust-relationship.json"
```

f. Lampirkan kebijakan IAM ke peran IAM.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::
$account_id:policy/AmazonEKS_CNI_IPv6_Policy \
    --role-name $node_role_name
```

> **Important**
>
> Untuk kesederhanaan dalam tutorial ini, kebijakan dilampirkan ke peran IAM ini. Namun, dalam klaster produksi, kami merekomendasikan untuk melampirkan kebijakan ke peran IAM yang terpisah. Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).

g. Lampirkan dua kebijakan terkelola IAM yang diperlukan ke peran IAM.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy \
  --role-name $node_role_name
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly \
  --role-name $node_role_name
```

h. Ambil ARN dari peran IAM dan simpan dalam variabel untuk langkah selanjutnya.

```
node_iam_role=$(aws iam get-role --role-name $node_role_name --
query="Role.Arn" --output text)
```

7. Buat grup node terkelola.

a. Lihat ID subnet yang Anda buat pada langkah sebelumnya.

```
echo $subnets
```

Contoh output adalah sebagai berikut.

```
subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE,subnet-
0377963d69EXAMPLE,subnet-0c05f819d5EXAMPLE
```

b. Buat grup simpul. Ganti *0a1a56c486EXAMPLE099e6ca77aEXAMPLE*,*0377963d69EXAMPLE*,, dan *0c05f819d5EXAMPLE* dengan nilai yang dikembalikan dalam output dari langkah sebelumnya. Pastikan untuk menghapus koma antara ID subnet dari output sebelumnya dalam perintah berikut. Anda dapat mengganti *t3.medium* dengan jenis instans Sistem AWS Nitro apa pun.

```
aws eks create-nodegroup --region $region_code --cluster-name $cluster_name
 --nodegroup-name $nodegroup_name \
    --subnets subnet-0a1a56c486EXAMPLE subnet-099e6ca77aEXAMPLE
 subnet-0377963d69EXAMPLE subnet-0c05f819d5EXAMPLE \
    --instance-types t3.medium --node-role $node_iam_role
```

Grup simpul membutuhkan waktu beberapa menit untuk membuatnya. Jalankan perintah berikut. Jangan lanjutkan ke langkah berikutnya sampai output yang dikembalikan`ACTIVE`.

```
aws eks describe-nodegroup --region $region_code --cluster-name
 $cluster_name --nodegroup-name $nodegroup_name \
    --query nodegroup.status --output text
```

8. Konfirmasikan bahwa default Pods adalah IPv6 alamat yang ditetapkan di IP kolom.

```
kubectl get pods -n kube-system -o wide
```

Contoh output adalah sebagai berikut.

```
NAME                       READY   STATUS    RESTARTS   AGE     IP
                             NODE
 NOMINATED NODE   READINESS GATES
aws-node-rslts             1/1     Running   1          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
aws-node-t74jh             1/1     Running   0          5m32s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-cw7w2   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::                ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-tx6n8   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::1               ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
kube-proxy-btpbk           1/1     Running   0          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
kube-proxy-jjk2g           1/1     Running   0          5m33s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

9. Konfirmasikan bahwa layanan default diberikan IPv6 alamat di IP kolom.

```
kubectl get services -n kube-system -o wide
```

Contoh output adalah sebagai berikut.

```
NAME       TYPE        CLUSTER-IP          EXTERNAL-IP   PORT(S)         AGE
 SELECTOR
```

```
kube-dns   ClusterIP   fd30:3087:b6c2::a   <none>        53/UDP,53/TCP   57m
 k8s-app=kube-dns
```

10. (Opsional) Menyebarkan aplikasi sampel atau menyebarkan AWS Load Balancer Controllerdan aplikasi sampel untuk memuat aplikasi keseimbangan atau lalu lintas jaringan ke. IPv6 Pods
11. Setelah Anda selesai dengan cluster dan node yang Anda buat untuk tutorial ini, Anda harus membersihkan sumber daya yang Anda buat dengan perintah berikut. Pastikan Anda tidak menggunakan sumber daya apa pun di luar tutorial ini sebelum menghapusnya.

    a. Jika Anda menyelesaikan langkah ini di shell yang berbeda dari saat Anda menyelesaikan langkah sebelumnya, tetapkan nilai semua variabel yang digunakan pada langkah sebelumnya, ganti *example values* dengan nilai yang Anda tentukan saat Anda menyelesaikan langkah sebelumnya. Jika Anda menyelesaikan langkah ini di shell yang sama dengan tempat Anda menyelesaikan langkah sebelumnya, lewati ke langkah berikutnya.

    ```
    export region_code=region-code
    export vpc_stack_name=my-eks-ipv6-vpc
    export cluster_name=my-cluster
    export nodegroup_name=my-nodegroup
    export account_id=111122223333
    export node_role_name=AmazonEKSNodeRole
    export cluster_role_name=myAmazonEKSClusterRole
    ```

    b. Hapus grup node Anda.

    ```
    aws eks delete-nodegroup --region $region_code --cluster-name $cluster_name
     --nodegroup-name $nodegroup_name
    ```

    Penghapusan membutuhkan waktu beberapa menit. Jalankan perintah berikut. Jangan lanjutkan ke langkah berikutnya jika ada output yang dikembalikan.

    ```
    aws eks list-nodegroups --region $region_code --cluster-name $cluster_name
     --query nodegroups --output text
    ```

    c. Hapus klaster .

    ```
    aws eks delete-cluster --region $region_code --name $cluster_name
    ```

Cluster membutuhkan beberapa menit untuk menghapus. Sebelum melanjutkan pastikan bahwa cluster dihapus dengan perintah berikut.

```
aws eks describe-cluster --region $region_code --name $cluster_name
```

Jangan lanjutkan ke langkah berikutnya sampai output Anda mirip dengan output berikut.

```
An error occurred (ResourceNotFoundException) when calling the
 DescribeCluster operation: No cluster found for name: my-cluster.
```

d. Hapus sumber daya IAM yang Anda buat. Ganti *AmazonEKS_CNI_IPv6_Policy* dengan nama yang Anda pilih, jika Anda memilih nama yang berbeda dari yang digunakan pada langkah sebelumnya.

```
aws iam detach-role-policy --role-name $cluster_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEKSClusterPolicy
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::$account_id:policy/AmazonEKS_CNI_IPv6_Policy
aws iam delete-policy --policy-arn arn:aws:iam::
$account_id:policy/AmazonEKS_CNI_IPv6_Policy
aws iam delete-role --role-name $cluster_role_name
aws iam delete-role --role-name $node_role_name
```

e. Hapus AWS CloudFormation tumpukan yang membuat VPC.

```
aws cloudformation delete-stack --region $region_code --stack-name
 $vpc_stack_name
```

SNAT untuk Pods

Jika Anda menerapkan klaster menggunakan `IPv6` keluarga, maka informasi dalam topik ini tidak berlaku untuk klaster Anda, karena `IPv6` alamat tidak diterjemahkan jaringan. Untuk informasi selengkapnya tentang penggunaan `IPv6` dengan cluster Anda, lihat[IPv6alamat untuk cluster,Pods, dan services](#).

Secara default, masing-masing Pod di klaster Anda diberi `IPv4` alamat pribadi dari blok routing antar-domain (CIDR) tanpa kelas yang terkait dengan VPC tempat digunakan. Pod Podsdalam VPC yang sama berkomunikasi satu sama lain menggunakan alamat IP pribadi ini sebagai titik akhir. Saat Pod berkomunikasi ke `IPv4` alamat apa pun yang tidak berada dalam blok CIDR yang terkait dengan VPC Anda, plugin Amazon VPC CNI (untuk keduanya Linuxatau Windows) menerjemahkan Pod's`IPv4` alamat ke alamat pribadi utama dari antarmuka elastic network primer dari node yang sedang berjalan, secara default*. `IPv4`Pod

> **Note**
>
> Untuk Windows node, ada detail tambahan yang perlu dipertimbangkan. Secara default, plugin VPC CNI untuk Windows didefinisikan dengan konfigurasi jaringan di mana lalu lintas ke tujuan dalam VPC yang sama dikecualikan untuk SNAT. Ini berarti bahwa komunikasi VPC internal telah dinonaktifkan SNAT dan alamat IP yang dialokasikan ke a dapat Pod dirutekan di dalam VPC. Tetapi lalu lintas ke tujuan di luar VPC memiliki Pod IP sumber yang diambil ke alamat IP utama ENI misalnya. Konfigurasi default ini untuk Windows memastikan bahwa pod dapat mengakses jaringan di luar VPC Anda dengan cara yang sama seperti instance host.

Karena perilaku ini:

- Anda Pods dapat berkomunikasi dengan sumber daya internet hanya jika node yang mereka jalankan memiliki alamat IP publik atau elastis yang ditetapkan untuk itu dan berada di subnet publik. Tabel rute terkait subnet publik memiliki rute ke gateway internet. Kami merekomendasikan untuk menyebarkan node ke subnet pribadi, bila memungkinkan.
- Untuk versi plugin sebelumnya`1.8.0`, sumber daya yang ada di jaringan atau VPC yang terhubung ke VPC cluster Anda menggunakan VPC peering, VPC transit AWS Direct Connect, atau tidak dapat memulai komunikasi ke antarmuka jaringan elastis sekunder Anda di belakang. Pods Anda Pods dapat memulai komunikasi ke sumber daya tersebut dan menerima tanggapan dari mereka.

Jika salah satu dari pernyataan berikut benar di lingkungan Anda, maka ubah konfigurasi default dengan perintah berikut.

- Anda memiliki sumber daya dalam jaringan atau VPC yang terhubung ke VPC cluster Anda menggunakan VPC peering, VPC transit AWS Direct Connect, atau yang perlu memulai

komunikasi Pods dengan Anda `IPv4` menggunakan alamat dan versi plugin Anda lebih awal dari. `1.8.0`

- Anda Pods berada di subnet pribadi dan perlu berkomunikasi keluar ke internet. Subnet memiliki rute ke gateway NAT.

```
kubectl set env daemonset -n kube-system aws-node AWS_VPC_K8S_CNI_EXTERNALSNAT=true
```

> **Note**
>
> Variabel konfigurasi `AWS_VPC_K8S_CNI_EXTERNALSNAT` dan `AWS_VPC_K8S_CNI_EXCLUDE_SNAT_CIDRS` CNI tidak berlaku untuk Windows node. Menonaktifkan SNAT tidak didukung untuk. Windows Sedangkan untuk mengecualikan daftar `IPv4` CIDR dari SNAT, Anda dapat menentukan ini dengan menentukan `ExcludedSnatCIDRs` parameter dalam skrip bootstrap. Windows Untuk informasi selengkapnya tentang penggunaan parameter ini, lihatParameter konfigurasi skrip bootstrap.

* Jika Pod's spesifikasi berisi `hostNetwork=true` (default adalah`false`), maka alamat IP-nya tidak diterjemahkan ke alamat yang berbeda. Ini adalah kasus untuk `kube-proxy` dan Amazon VPC CNI plugin for Kubernetes Pods yang berjalan di cluster Anda, secara default. Untuk iniPods, alamat IP sama dengan alamat IP utama node, sehingga alamat Pod's IP tidak diterjemahkan. Untuk informasi selengkapnya tentang Pod's `hostNetwork` setelan, lihat inti PodSpec v1 di referensi Kubernetes API.

Konfigurasikan klaster Anda untuk kebijakan Kubernetes jaringan

Secara default, tidak ada batasan Kubernetes untuk alamat IP, port, atau koneksi antara apa pun Pods di cluster Anda atau antara Anda Pods dan sumber daya di jaringan lain. Anda dapat menggunakan kebijakan Kubernetes jaringan untuk membatasi lalu lintas jaringan ke dan dari AndaPods. Untuk informasi selengkapnya, lihat Kebijakan Jaringan dalam Kubernetes dokumentasi.

Jika Anda memiliki versi `1.13` atau versi sebelumnya Amazon VPC CNI plugin for Kubernetes di klaster Anda, Anda perlu menerapkan solusi pihak ketiga untuk menerapkan kebijakan Kubernetes jaringan ke klaster Anda. Versi `1.14` atau versi plugin yang lebih baru dapat menerapkan kebijakan jaringan, jadi Anda tidak perlu menggunakan solusi pihak ketiga. Dalam topik ini, Anda mempelajari cara mengonfigurasi klaster agar menggunakan kebijakan Kubernetes jaringan di klaster tanpa menggunakan add-on pihak ketiga.

Kebijakan jaringan di Amazon VPC CNI plugin for Kubernetes didukung dalam konfigurasi berikut.

- Amazon EKS cluster versi `1.25` dan yang lebih baru.
- Versi 1.14 atau yang lebih baru Amazon VPC CNI plugin for Kubernetes di cluster Anda.
- Cluster dikonfigurasi untuk `IPv4` atau `IPv6` alamat.
- Anda dapat menggunakan kebijakan jaringan dengan grup keamanan untuk Pods. Dengan kebijakan jaringan, Anda dapat mengontrol semua komunikasi dalam cluster. Dengan grup keamanan untukPods, Anda dapat mengontrol akses ke Layanan AWS dari aplikasi dalam filePod.
- Anda dapat menggunakan kebijakan jaringan dengan jaringan kustom dan delegasi awalan.

Pertimbangan

- Saat menerapkan kebijakan Amazon VPC CNI plugin for Kubernetes jaringan ke cluster Anda dengan Amazon VPC CNI plugin for Kubernetes, Anda dapat menerapkan kebijakan ke node Amazon EC2 Linux saja. Anda tidak dapat menerapkan kebijakan ke Fargate atau Windows node.
- Jika klaster Anda saat ini menggunakan solusi pihak ketiga untuk mengelola kebijakan Kubernetes jaringan, Anda dapat menggunakan kebijakan yang sama dengan Amazon VPC CNI plugin for Kubernetes. Namun Anda harus menghapus solusi yang ada sehingga tidak mengelola kebijakan yang sama.
- Anda dapat menerapkan beberapa kebijakan jaringan untuk hal yang samaPod. Ketika dua atau beberapa kebijakan yang memilih yang sama Pod dikonfigurasi, semua kebijakan diterapkan ke kebijakanPod.
- Jumlah maksimum kombinasi unik port untuk setiap protokol di masing-masing `ingress:` atau `egress:` pemilih dalam kebijakan jaringan adalah 24.
- Untuk setiap Kubernetes layanan Anda, port layanan harus sama dengan port kontainer. Jika Anda menggunakan port bernama, gunakan nama yang sama dalam spesifikasi layanan juga.
- Penegakan kebijakan saat Pod startup

  Amazon VPC CNI plugin for KubernetesKonfigurasi kebijakan jaringan untuk Pod secara paralel dengan penyediaan pod. Sampai semua kebijakan dikonfigurasi untuk pod baru, container di pod baru akan dimulai dengan kebijakan allow default. Ini disebut mode standar. Kebijakan allow default berarti bahwa semua lalu lintas masuk dan keluar diizinkan ke dan dari pod baru.

  Anda dapat mengubah kebijakan jaringan default ini dengan menyetel variabel lingkungan `NETWORK_POLICY_ENFORCING_MODE` ke `strict` dalam `aws-node` wadah CNI VPC. `DaemonSet`

```
env:
  - name: NETWORK_POLICY_ENFORCING_MODE
    value: "strict"
```

Dengan `NETWORK_POLICY_ENFORCING_MODE` variabel disetel `kestrict`, pod yang menggunakan VPC CNI dimulai dengan kebijakan penolakan default, kemudian kebijakan dikonfigurasi. Ini disebut mode ketat. Dalam mode ketat, Anda harus memiliki kebijakan jaringan untuk setiap titik akhir yang perlu diakses pod di klaster Anda. Perhatikan bahwa persyaratan ini berlaku untuk CoreDNS pod. Kebijakan penolakan default tidak dikonfigurasi untuk pod dengan jaringan Host.

- Fitur kebijakan jaringan membuat dan memerlukan Definisi Sumber Daya `PolicyEndpoint` Kustom (CRD) yang disebut`policyendpoints.networking.k8s.aws`. `PolicyEndpoint`objek Sumber Daya Kustom dikelola oleh Amazon EKS. Anda tidak boleh memodifikasi atau menghapus sumber daya ini.
- Jika Anda menjalankan pod yang menggunakan kredensial IAM peran instance atau terhubung ke IMDS EC2, berhati-hatilah untuk memeriksa kebijakan jaringan yang akan memblokir akses ke IMDS EC2. Anda mungkin perlu menambahkan kebijakan jaringan untuk mengizinkan akses ke EC2 IMDS. Untuk informasi selengkapnya, lihat Metadata instans dan data pengguna di Panduan Pengguna Amazon EC2 untuk Instans Linux.

  Pod yang menggunakan peran IAM untuk akun layanan tidak mengakses IMDS EC2.
- Amazon VPC CNI plugin for KubernetesItu tidak menerapkan kebijakan jaringan ke antarmuka jaringan tambahan untuk setiap pod, hanya antarmuka utama untuk setiap pod (`eth0`). Ini mempengaruhi arsitektur berikut:
  - `IPv6`pod dengan `ENABLE_V4_EGRESS` variabel disetel `ketrue`. Variabel ini memungkinkan fitur `IPv4` keluar untuk menghubungkan pod IPv6 ke `IPv4` titik akhir seperti yang berada di luar cluster. Fitur `IPv4` jalan keluar bekerja dengan membuat antarmuka jaringan tambahan dengan alamat IPv4 loopback lokal.
  - Saat menggunakan plugin jaringan berantai seperti. Multus Karena plugin ini menambahkan antarmuka jaringan ke setiap pod, kebijakan jaringan tidak diterapkan ke plugin jaringan yang dirantai.
- Fitur kebijakan jaringan menggunakan port `8162` pada node untuk metrik secara default. Juga, fitur yang digunakan port `8163` untuk probe kesehatan. Jika Anda menjalankan aplikasi lain di node atau di dalam pod yang perlu menggunakan port ini, aplikasi gagal dijalankan. Dalam versi VPC CNI `v1.14.1` atau yang lebih baru, Anda dapat mengubah port port ini di tempat-tempat berikut:

AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel navigasi kiri, pilih Cluster, lalu pilih nama cluster yang ingin Anda konfigurasikan untuk add-on Amazon VPC CNI.
3. Pilih tab Add-ons.
4. Pilih kotak di kanan atas kotak add-on dan kemudian pilih Edit.
5. Pada halaman Konfigurasi ***nama addon***:

   a. Pilih versi `v1.14.0-eksbuild.3` atau yang lebih baru dalam daftar dropdown Versi.

   b. Perluas pengaturan konfigurasi opsional.

   c. Masukkan kunci `"enableNetworkPolicy":` dan nilai JSON `"true"` dalam nilai Konfigurasi. Teks yang dihasilkan harus berupa objek JSON yang valid. Jika kunci dan nilai ini adalah satu-satunya data dalam kotak teks, kelilingi kunci dan nilai dengan kurawal kurawal. `{}`

      Contoh berikut mengaktifkan fitur kebijakan jaringan, log kebijakan jaringan diaktifkan, log kebijakan jaringan yang dikirim ke Amazon CloudWatch Logs, dan metrik serta probe kesehatan disetel ke nomor port default:

```
{
    "enableNetworkPolicy": "true",
    "nodeAgent": {
        "enablePolicyEventLogs": "true",
        "enableCloudWatchLogs": "true",
        "healthProbeBindAddr": "8163",
        "metricsBindAddr": "8162"
    }
}
```

Helm

Jika Anda telah menginstal Amazon VPC CNI plugin for Kubernetes melaluihelm, Anda dapat memperbarui konfigurasi untuk mengubah port.

- Jalankan perintah berikut untuk mengubah port. Tetapkan nomor port dalam nilai untuk kunci `nodeAgent.metricsBindAddr` atau kuncinodeAgent.healthProbeBindAddr, masing-masing.

```
helm upgrade --set nodeAgent.metricsBindAddr=8162 --set
 nodeAgent.healthProbeBindAddr=8163 aws-vpc-cni --namespace kube-system eks/
aws-vpc-cni
```

kubectl

1. Buka `aws-node DaemonSet` di editor Anda.

```
kubectl edit daemonset -n kube-system aws-node
```

2. Ganti nomor port dalam argumen perintah berikut di dalam `aws-network-policy-agent` wadah dalam `args:` manifes daemonset VPC CNI`aws-node`.

```
    - args:
            - --metrics-bind-addr=:8162
            - --health-probe-bind-addr=:8163
```

Prasyarat

- Versi cluster minimum

Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS. Cluster harus Kubernetes versi `1.25` atau yang lebih baru. Cluster harus menjalankan salah satu Kubernetes versi dan versi platform yang tercantum dalam tabel berikut. Perhatikan bahwa versi apa pun Kubernetes dan platform yang lebih lambat dari yang terdaftar juga didukung. Anda dapat memeriksa Kubernetes versi Anda saat ini dengan mengganti *`my-cluster`* dalam perintah berikut dengan nama cluster Anda dan kemudian menjalankan perintah yang dimodifikasi:

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

Versi Kubernetes	Versi platform
1.27.4	eks.5
1.26.7	eks.6
1.25.12	eks.7

- Versi VPC CNI minimum

  Versi 1.14 atau yang lebih baru Amazon VPC CNI plugin for Kubernetes di cluster Anda. Anda dapat melihat versi mana yang saat ini Anda miliki dengan perintah berikut.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
 cut -d : -f 3
```

  Jika versi Anda lebih awal dari1.14, lihat Memperbarui add-on Amazon EKS untuk meningkatkan ke versi 1.14 atau yang lebih baru.

- Versi kernel Linux minimum

  Node Anda harus memiliki versi kernel Linux 5.10 atau yang lebih baru. Anda dapat memeriksa versi kernel Anda denganuname -r. Jika Anda menggunakan versi terbaru Amazon EKS yang dioptimalkan Amazon Linux, Amazon EKS mengoptimalkan AMI Amazon Linux yang dipercepat, dan AMI Bottlerocket, mereka sudah memiliki versi kernel yang diperlukan.

  Amazon EKS mengoptimalkan versi AMI Amazon Linux yang dipercepat v20231116 atau yang lebih baru memiliki versi kernel5.10.

Untuk mengonfigurasi klaster Anda untuk menggunakan kebijakan Kubernetes jaringan

1. Pasang sistem file BPF

> **Note**
>
> Jika klaster Anda versi 1.27 atau lebih baru, Anda dapat melewati langkah ini karena semua Amazon EKS mengoptimalkan Amazon Linux dan Bottlerocket AMI untuk 1.27 atau yang lebih baru sudah memiliki fitur ini.

Untuk semua versi cluster lainnya, jika Anda memutakhirkan Amazon EKS yang dioptimalkan Amazon Linux ke versi `v20230703` atau yang lebih baru atau Anda memutakhirkan AMI Bottlerocket ke `v1.0.2` versi atau yang lebih baru, Anda dapat melewati langkah ini.

a. Pasang sistem file Berkeley Packet Filter (BPF) pada setiap node Anda.

```
sudo mount -t bpf bpffs /sys/fs/bpf
```

b. Kemudian, tambahkan perintah yang sama ke data pengguna Anda di template peluncuran untuk Grup Auto Scaling Amazon EC2 Anda.

2. Aktifkan kebijakan jaringan di VPC CNI

a. Lihat jenis add-on yang diinstal pada cluster Anda. Bergantung pada alat yang digunakan untuk membuat klaster, saat ini Anda mungkin tidak menginstal jenis add-on Amazon EKS di cluster Anda. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

Jika nomor versi dikembalikan, Anda memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda dan tidak perlu menyelesaikan langkah-langkah yang tersisa dalam prosedur ini. Jika kesalahan dikembalikan, Anda tidak memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda.

b. • Pengaya Amazon EKS

AWS Management Console

a. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.

b. Di panel navigasi kiri, pilih Cluster, lalu pilih nama cluster yang ingin Anda konfigurasikan untuk add-on Amazon VPC CNI.

c. Pilih tab Add-ons.

d. Pilih kotak di kanan atas kotak add-on dan kemudian pilih Edit.

e. Pada halaman Konfigurasi ***nama addon***:

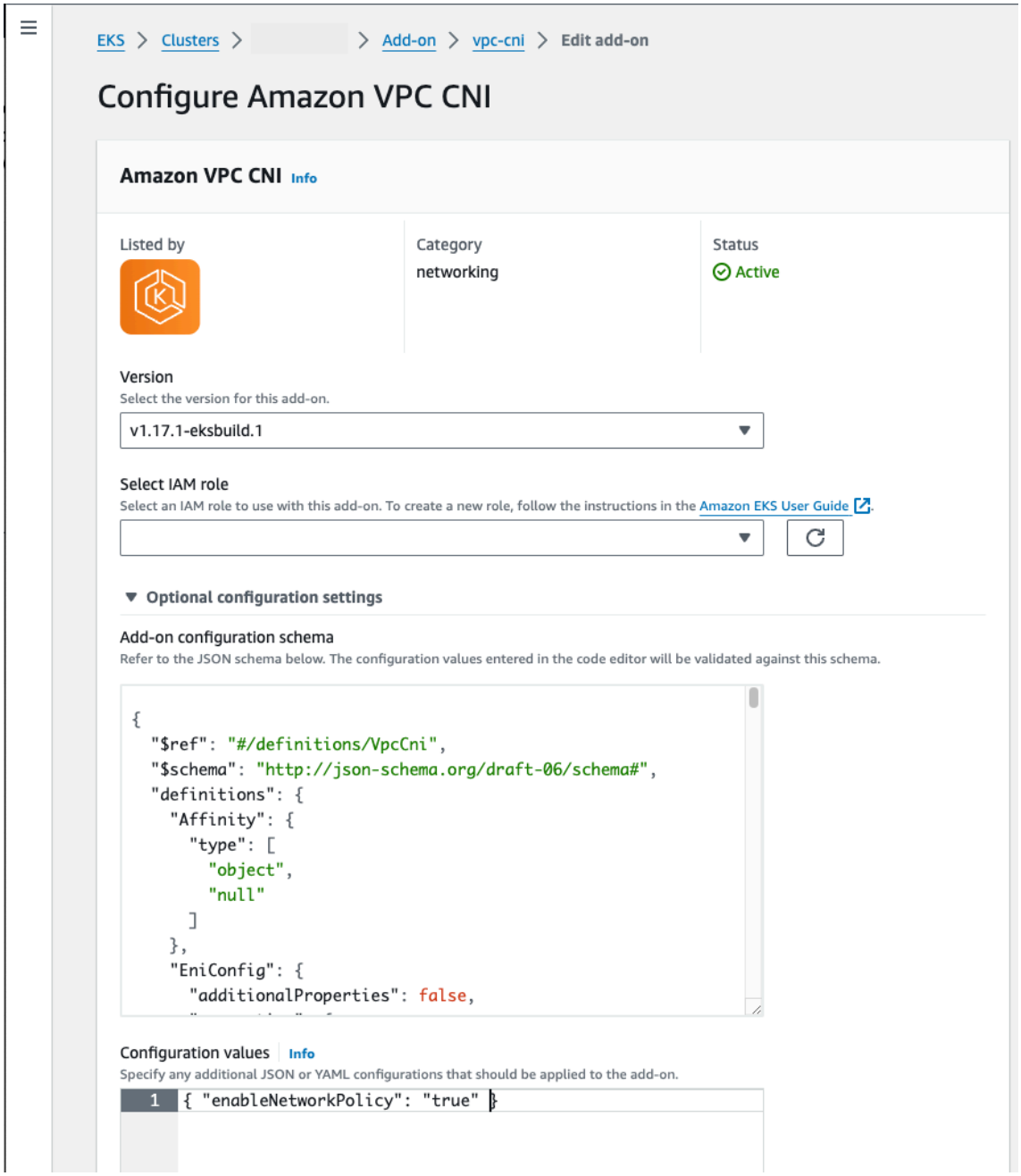
i. Pilih versi `v1.14.0-eksbuild.3` atau yang lebih baru dalam daftar dropdown Versi.

ii. Perluas pengaturan konfigurasi opsional.

iii. Masukkan kunci `"enableNetworkPolicy":` dan nilai JSON `"true"` dalam nilai Konfigurasi. Teks yang dihasilkan harus berupa objek JSON yang valid. Jika kunci dan nilai ini adalah satu-satunya data dalam kotak teks, kelilingi kunci dan nilai dengan kurawal kurawal. `{}` Contoh berikut menunjukkan kebijakan jaringan diaktifkan:

```
{ "enableNetworkPolicy": "true" }
```

Screenshot berikut menunjukkan contoh skenario ini.



AWS CLI

- Jalankan AWS CLI perintah berikut. Ganti `my-cluster` dengan nama cluster Anda dan peran IAM ARN dengan peran yang Anda gunakan.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni
 --addon-version v1.14.0-eksbuild.3 \
```

```
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values
 '{"enableNetworkPolicy": "true"}'
```

- Add-on yang dikelola sendiri

  Helm

  Jika Anda telah menginstal Amazon VPC CNI plugin for Kubernetes melaluihelm, Anda dapat memperbarui konfigurasi untuk mengaktifkan kebijakan jaringan.

  - Jalankan perintah berikut untuk mengaktifkan kebijakan jaringan.

    ```
    helm upgrade --set enableNetworkPolicy=true aws-vpc-cni --namespace
     kube-system eks/aws-vpc-cni
    ```

  kubectl

  a. Buka amazon-vpc-cni ConfigMap di editor Anda.

     ```
     kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
     ```

  b. Tambahkan baris berikut ke data dalamConfigMap.

     ```
     enable-network-policy-controller: "true"
     ```

     Setelah Anda menambahkan baris, Anda ConfigMap akan terlihat seperti contoh berikut.

     ```
     apiVersion: v1
      kind: ConfigMap
      metadata:
       name: amazon-vpc-cni
       namespace: kube-system
      data:
       enable-network-policy-controller: "true"
     ```

  c. Buka aws-node DaemonSet di editor Anda.

```
kubectl edit daemonset -n kube-system aws-node
```

d. Ganti `false` dengan `true` dalam argumen perintah di `--enable-network-policy=false` dalam `aws-network-policy-agent` wadah `args:` dalam manifes daemonset VPC CNI`aws-node`.

```
    - args:
       - --enable-network-policy=true
```

3. Konfirmasikan bahwa `aws-node` pod berjalan di klaster Anda.

```
kubectl get pods -n kube-system | grep 'aws-node\|amazon'
```

Contoh output adalah sebagai berikut.

```
aws-node-gmqp7                                          2/2     Running   1 (24h
 ago)   24h
aws-node-prnsh                                          2/2     Running   1 (24h
 ago)   24h
```

Jika kebijakan jaringan diaktifkan, ada 2 kontainer dalam `aws-node` pod. Di versi sebelumnya dan jika kebijakan jaringan dinonaktifkan, hanya ada satu kontainer di `aws-node` pod.

Sekarang Anda dapat menerapkan kebijakan Kubernetes jaringan ke klaster Anda. Untuk informasi selengkapnya, lihat Kuberneteskebijakan jaringan.

Bintang demo kebijakan jaringan

Demo ini membuat layanan front-end, back-end, dan klien di cluster Amazon EKS Anda. Demo ini juga menciptakan antarmuka pengguna grafis manajemen yang menunjukkan jalur masuk dan keluar yang tersedia di antara setiap layanan. Kami menyarankan Anda menyelesaikan demo di klaster tempat Anda tidak menjalankan beban kerja produksi.

Sebelum Anda membuat kebijakan jaringan apa pun, semua layanan dapat berkomunikasi dua arah. Setelah Anda menerapkan kebijakan jaringan, Anda dapat melihat bahwa klien hanya dapat berkomunikasi dengan layanan front-end, dan back-end hanya menerima lalu lintas dari front-end.

Untuk menjalankan demo kebijakan Stars

1. Menerapkan layanan antarmuka pengguna front-end, back-end, klien, dan manajemen:

```
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/namespace.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/management-ui.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/backend.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/frontend.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/client.yaml
```

2. Lihat semua Pods di cluster.

```
kubectl get pods -A
```

Contoh output adalah sebagai berikut.

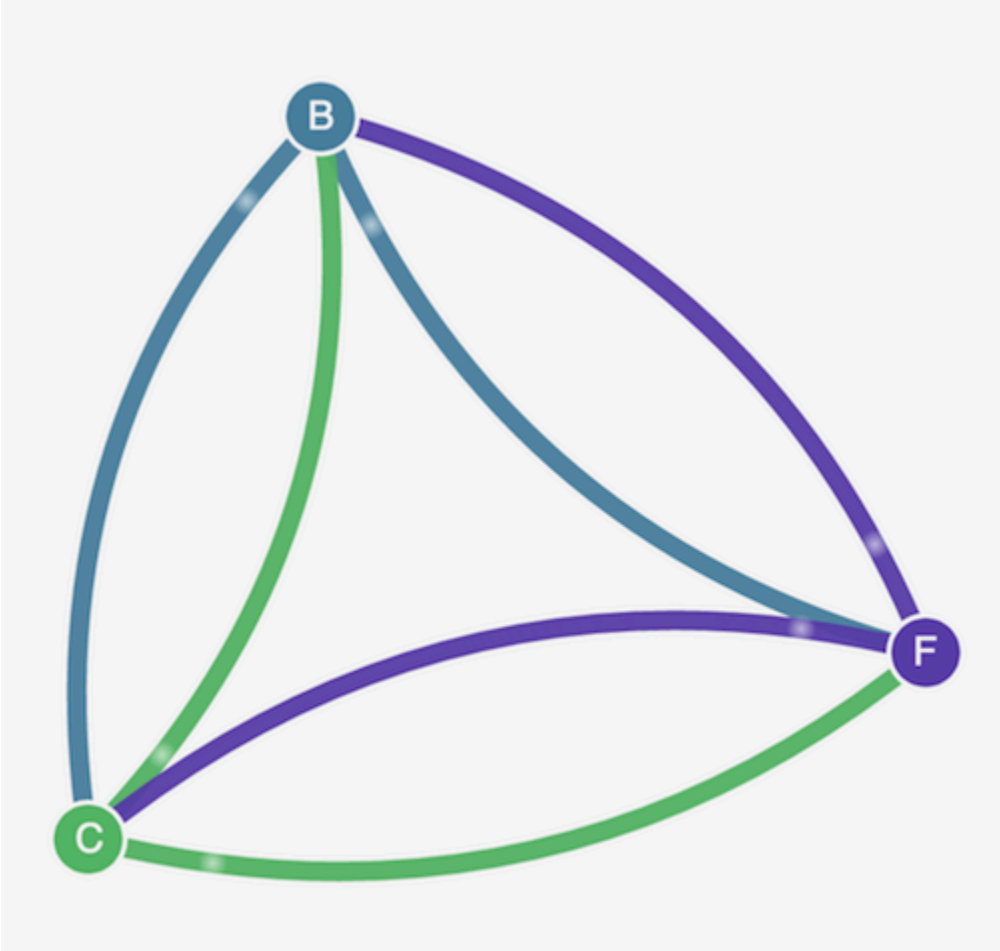
Dalam output Anda, Anda akan melihat pod di namespace yang ditunjukkan pada output berikut. *NAMA-NAMA* Pod Anda dan jumlah pod di READY kolom berbeda dari yang ada di output berikut. Jangan lanjutkan sampai Anda melihat pod dengan nama yang mirip dan semuanya ada Running di STATUS kolom.

```
NAMESPACE         NAME                                       READY   STATUS
 RESTARTS   AGE
[...]
client            client-xlffc                               1/1     Running   0
       5m19s
[...]
management-ui     management-ui-qrb2g                        1/1     Running   0
       5m24s
stars             backend-sz87q                              1/1     Running   0
       5m23s
stars             frontend-cscnf                             1/1     Running   0
       5m21s
[...]
```

3. Untuk terhubung ke antarmuka pengguna manajemen, sambungkan ke layanan EXTERNAL-IP yang berjalan di klaster Anda:

```
kubectl get service/management-ui -n management-ui
```

4. Buka browser ke lokasi dari langkah sebelumnya. Anda harus melihat antarmuka pengguna manajemen. Node C adalah layanan klien, simpul F adalah layanan front-end, dan simpul B adalah layanan back-end. Setiap node memiliki akses komunikasi penuh ke semua node lain, seperti yang ditunjukkan oleh garis tebal berwarna.



5. Terapkan kebijakan jaringan berikut di ruang nama `stars` dan `client` ruang nama untuk mengisolasi layanan satu sama lain:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: default-deny
spec:
  podSelector:
    matchLabels: {}
```

Anda dapat menggunakan perintah berikut untuk menerapkan kebijakan ke kedua ruang nama:

```
kubectl apply -n stars -f https://eksworkshop.com/beginner/120_network-policies/
calico/stars_policy_demo/apply_network_policies.files/default-deny.yaml
kubectl apply -n client -f https://eksworkshop.com/beginner/120_network-policies/
calico/stars_policy_demo/apply_network_policies.files/default-deny.yaml
```

6. Segarkan peramban Anda. Anda melihat bahwa antarmuka pengguna manajemen tidak dapat lagi menjangkau salah satu node, sehingga tidak muncul di antarmuka pengguna.
7. Terapkan kebijakan jaringan yang berbeda berikut untuk memungkinkan antarmuka pengguna manajemen mengakses layanan. Terapkan kebijakan ini untuk mengizinkan UI:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

Terapkan kebijakan ini untuk mengizinkan klien:

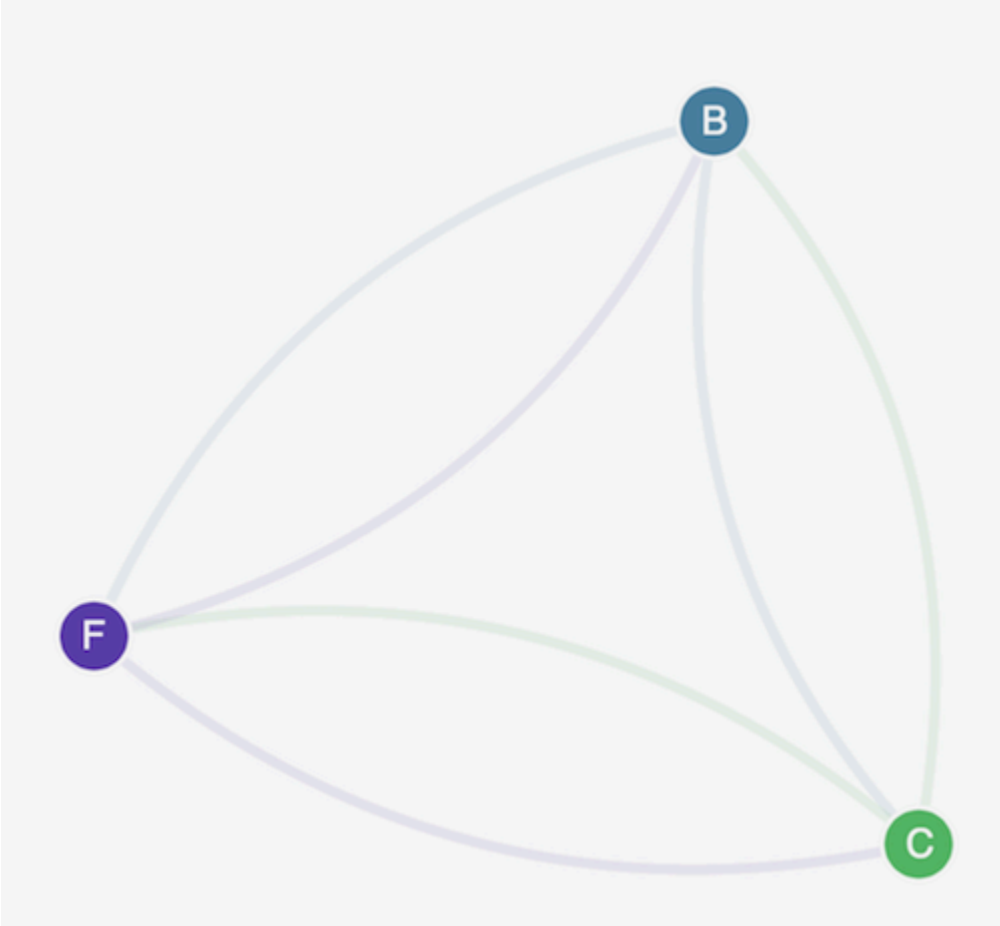
```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: client
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
```

```
            role: management-ui
```

Anda dapat menggunakan perintah berikut untuk menerapkan kedua kebijakan:

```
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/apply_network_policies.files/allow-ui.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/apply_network_policies.files/allow-ui-client.yaml
```

8. Segarkan peramban Anda. Anda melihat bahwa antarmuka pengguna manajemen dapat mencapai node lagi, tetapi node tidak dapat berkomunikasi satu sama lain.

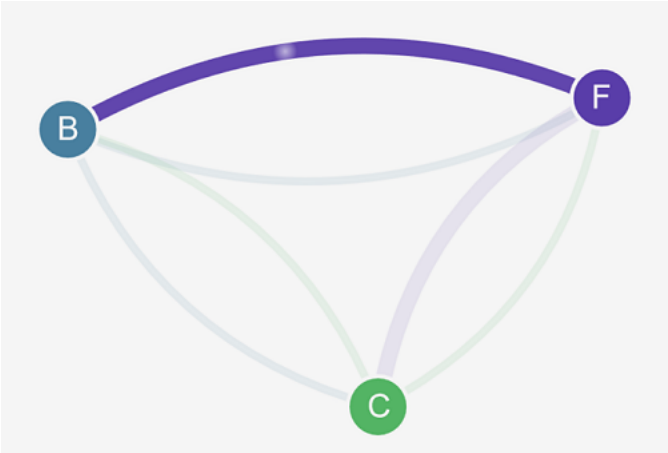


9. Terapkan kebijakan jaringan berikut untuk mengizinkan lalu lintas dari layanan front-end ke layanan back-end:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
```

```
  name: backend-policy
spec:
  podSelector:
    matchLabels:
      role: backend
  ingress:
    - from:
        - podSelector:
            matchLabels:
              role: frontend
      ports:
        - protocol: TCP
          port: 6379
```

10. Segarkan peramban Anda. Anda melihat bahwa front-end dapat berkomunikasi dengan back-end.

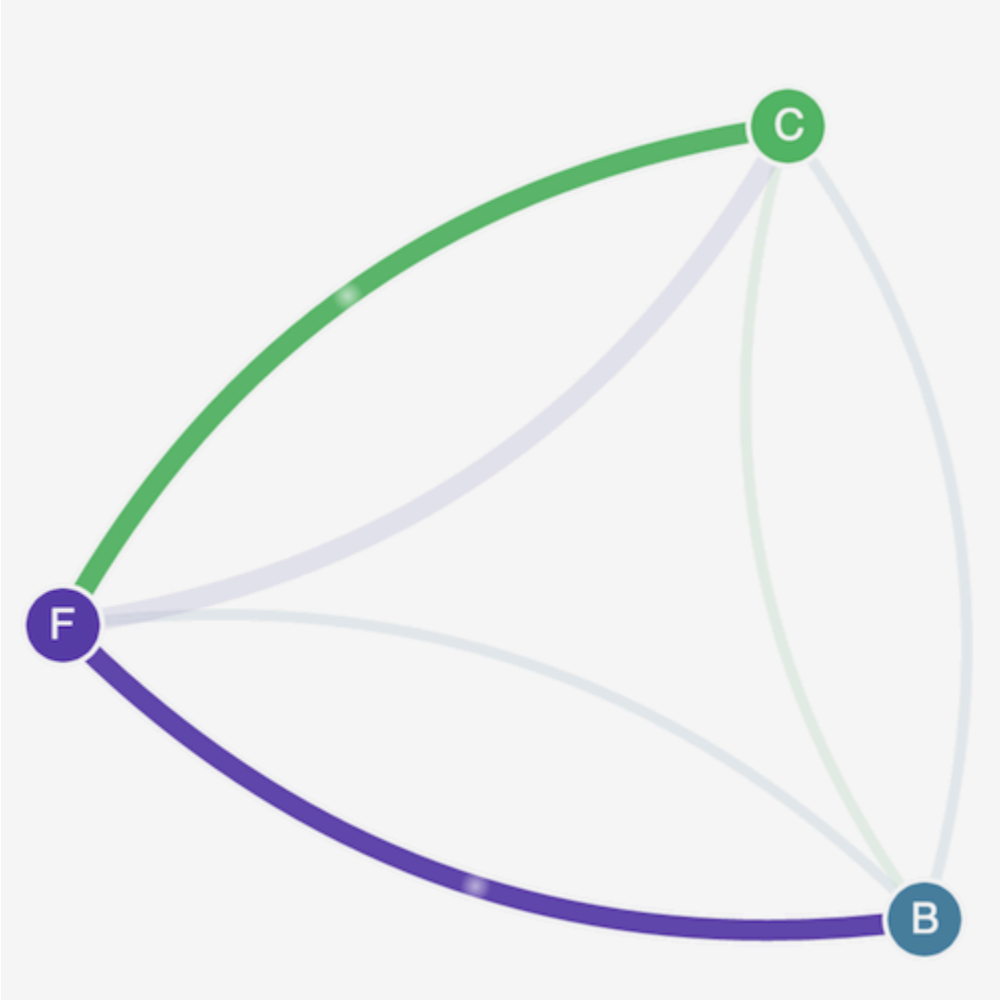


11. Terapkan kebijakan jaringan berikut untuk mengizinkan lalu lintas dari klien ke layanan front-end:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: frontend-policy
spec:
  podSelector:
    matchLabels:
      role: frontend
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
```

```
            role: client
    ports:
      - protocol: TCP
        port: 80
```

12. Segarkan peramban Anda. Anda melihat bahwa klien dapat berkomunikasi dengan layanan front-end. Layanan front-end masih dapat berkomunikasi dengan layanan back-end.



13. (Opsional) Setelah selesai dengan demo, Anda dapat menghapus sumber dayanya.

```
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/client.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/frontend.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/backend.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/management-ui.yaml
```

```
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/namespace.yaml
```

Bahkan setelah menghapus sumber daya, masih ada titik akhir kebijakan jaringan pada node yang mungkin mengganggu jaringan di cluster Anda dengan cara yang tidak terduga. Satu-satunya cara pasti untuk menghapus aturan ini adalah dengan me-reboot node atau menghentikan semua node dan mendaur ulangnya. Untuk mengakhiri semua simpul, atur jumlah Grup Auto Scaling yang diinginkan ke 0, lalu buat cadangan ke jumlah yang diinginkan, atau cukup hentikan simpul.

Pemecahan masalah kebijakan jaringan

Anda dapat memecahkan masalah dan menyelidiki koneksi jaringan yang menggunakan kebijakan jaringan dengan membaca Log kebijakan jaringan dan dengan menjalankan alat dari. eBPF SDK

Log kebijakan jaringan

Apakah koneksi diizinkan atau ditolak oleh kebijakan jaringan dicatat dalam log aliran. Log kebijakan jaringan pada setiap node menyertakan log aliran untuk setiap pod yang memiliki kebijakan jaringan. Log kebijakan jaringan disimpan di`/var/log/aws-routed-eni/network-policy-agent.log`. Contoh berikut adalah dari sebuah `network-policy-agent.log` file:

```
{"level":"info","timestamp":"2023-05-30T16:05:32.573Z","logger":"ebpf-
client","msg":"Flow Info: ","Src
IP":"192.168.87.155","Src Port":38971,"Dest IP":"64.6.160","Dest
Port":53,"Proto":"UDP","Verdict":"ACCEPT"}
```

Log kebijakan jaringan dinonaktifkan secara default. Untuk mengaktifkan log kebijakan jaringan, ikuti langkah-langkah berikut:

> **Note**
>
> Log kebijakan jaringan memerlukan 1 vCPU tambahan untuk `aws-network-policy-agent` wadah dalam manifes daemonset VPC CNI. `aws-node`

Pengaya Amazon EKS

AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel navigasi kiri, pilih Cluster, lalu pilih nama cluster yang ingin Anda konfigurasikan untuk add-on Amazon VPC CNI.
3. Pilih tab Add-ons.
4. Pilih kotak di kanan atas kotak add-on dan kemudian pilih Edit.
5. Pada halaman Konfigurasi ***nama addon***:

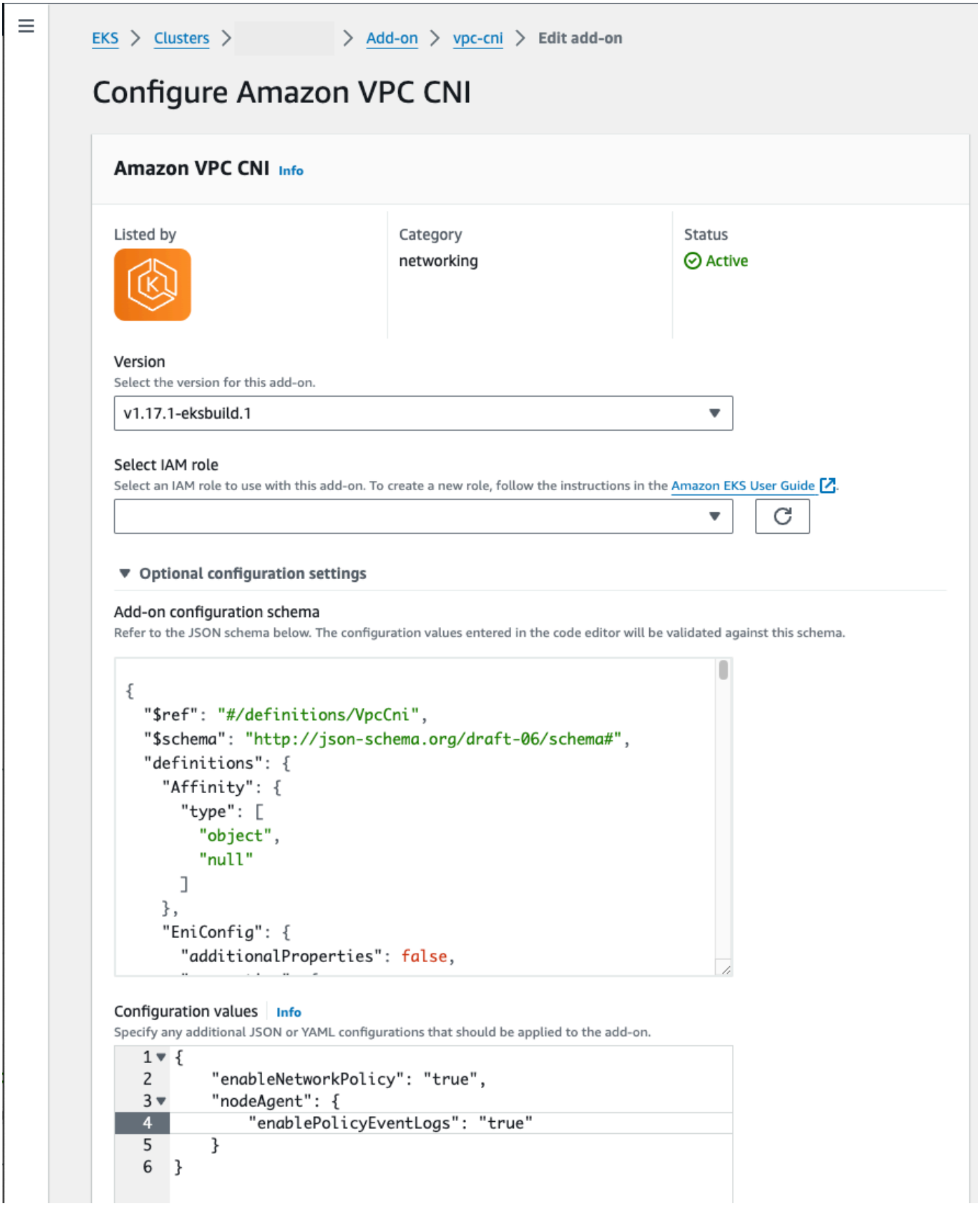
    a. Pilih versi `v1.14.0-eksbuild.3` atau yang lebih baru dalam daftar dropdown Versi.

    b. Perluas pengaturan konfigurasi opsional.

    c. Masukkan kunci JSON tingkat atas `"nodeAgent":` dan nilai adalah objek dengan kunci `"enablePolicyEventLogs":` dan nilai `"true"` dalam nilai Konfigurasi. Teks yang dihasilkan harus berupa objek JSON yang valid. Contoh berikut menunjukkan kebijakan jaringan dan log kebijakan jaringan diaktifkan, dan log kebijakan jaringan dikirim ke CloudWatch Log:

    ```
    {
        "enableNetworkPolicy": "true",
        "nodeAgent": {
            "enablePolicyEventLogs": "true"
        }
    }
    ```

Screenshot berikut menunjukkan contoh skenario ini.


EKS > Clusters > > Add-on > vpc-cni > Edit add-on
Configure Amazon VPC CNI
Amazon VPC CNI Info
Listed by
Category
networking
Status
Active
Version
Select the version for this add-on.
v1.17.1-eksbuild.1
Select IAM role
Select an IAM role to use with this add-on. To create a new role, follow the instructions in the Amazon EKS User Guide.
Optional configuration settings
Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.
{
"$ref": "#/definitions/VpcCni",
"$schema": "http://json-schema.org/draft-06/schema#",
"definitions": {
"Affinity": {
"type": [
"object",
"null"
]
},
"EniConfig": {
"additionalProperties": false,
Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.
1 {
2 "enableNetworkPolicy": "true",
3 "nodeAgent": {
4 "enablePolicyEventLogs": "true"
5 }
6 }

AWS CLI

- Jalankan AWS CLI perintah berikut. Ganti `my-cluster` dengan nama cluster Anda dan ganti peran IAM ARN dengan peran yang Anda gunakan.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values '{"nodeAgent":
 {"enablePolicyEventLogs": "true"}}'
```

Add-on yang dikelola sendiri

Helm

Jika Anda telah menginstal Amazon VPC CNI plugin for Kubernetes melaluihelm, Anda dapat memperbarui konfigurasi untuk menulis log kebijakan jaringan.

- Jalankan perintah berikut untuk mengaktifkan kebijakan jaringan.

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true aws-vpc-cni --namespace
 kube-system eks/aws-vpc-cni
```

kubectl

Jika Anda telah menginstal Amazon VPC CNI plugin for Kubernetes melaluikubectl, Anda dapat memperbarui konfigurasi untuk menulis log kebijakan jaringan.

1. Buka `aws-node` `DaemonSet` di editor Anda.

```
kubectl edit daemonset -n kube-system aws-node
```

2. Ganti `false` dengan `true` dalam argumen perintah di `--enable-policy-event-logs=false` dalam `aws-network-policy-agent` wadah `args:` dalam manifes daemonset VPC CNI`aws-node`.

```
     - args:
```

```
        - --enable-policy-event-logs=true
```

Kirim log kebijakan jaringan ke Amazon CloudWatch Logs

Anda dapat memantau log kebijakan jaringan menggunakan layanan seperti Amazon CloudWatch Logs. Anda dapat menggunakan metode berikut untuk mengirim log kebijakan jaringan ke CloudWatch Log.

Untuk kluster EKS, log kebijakan akan ditempatkan di bawah `/aws/eks/`*`cluster-name`*`/cluster/` dan untuk klaster K8S yang dikelola sendiri, log akan ditempatkan di bawah /. `/aws/k8s-cluster/cluster`

Kirim log kebijakan jaringan dengan Amazon VPC CNI plugin for Kubernetes

Jika Anda mengaktifkan kebijakan jaringan, kontainer kedua ditambahkan ke `aws-node` pod untuk agen node. Agen node ini dapat mengirim log kebijakan jaringan ke CloudWatch Log.

> **Note**
>
> Hanya log kebijakan jaringan yang dikirim oleh agen node. Log lain yang dibuat oleh VPC CNI tidak disertakan.

Prasyarat

- Tambahkan izin berikut sebagai bait atau kebijakan terpisah ke peran IAM yang Anda gunakan untuk CNI VPC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

Pengaya Amazon EKS

AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel navigasi kiri, pilih Cluster, lalu pilih nama cluster yang ingin Anda konfigurasikan untuk add-on Amazon VPC CNI.
3. Pilih tab Add-ons.
4. Pilih kotak di kanan atas kotak add-on dan kemudian pilih Edit.
5. Pada halaman Konfigurasi ***nama addon***:

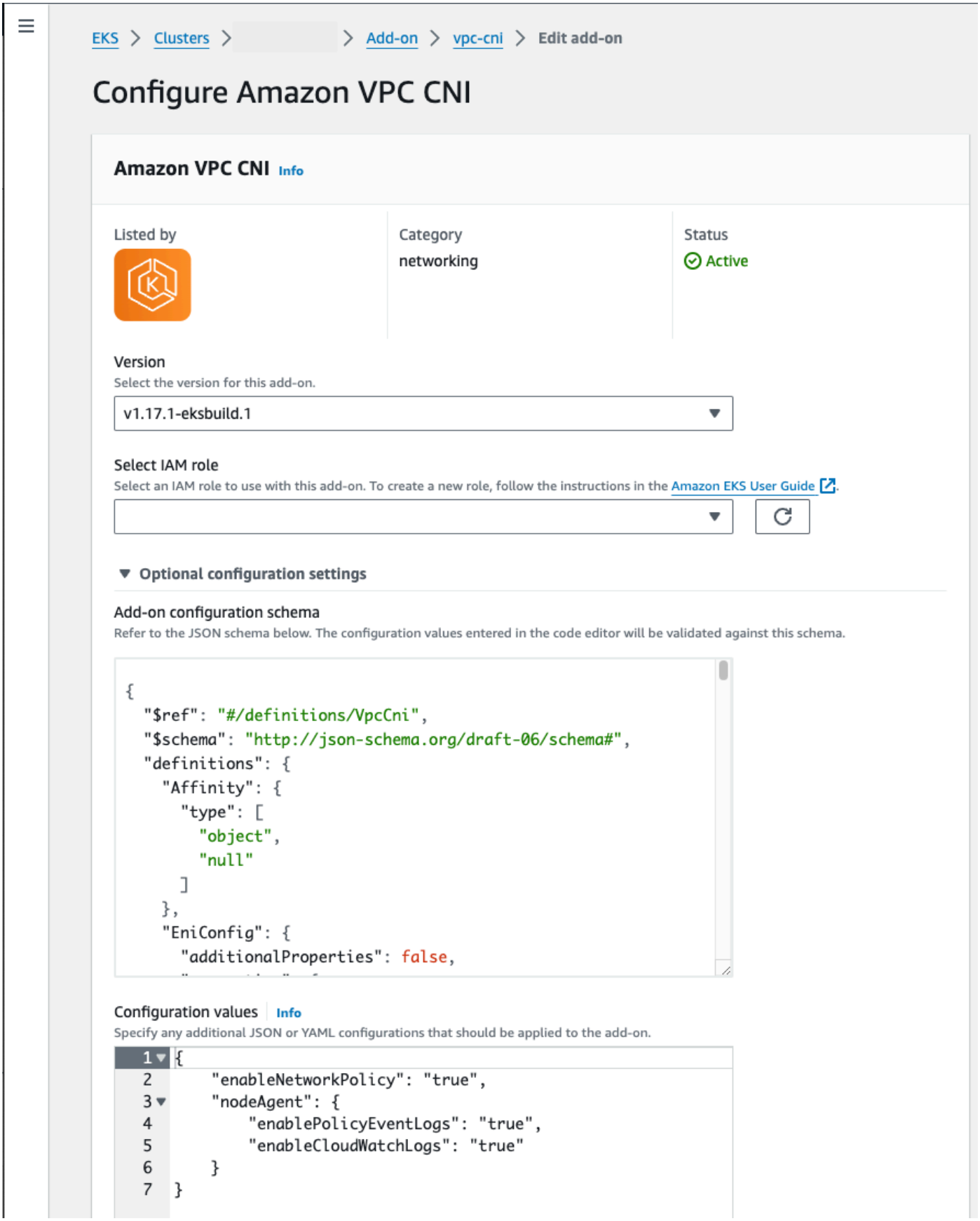
    a. Pilih versi `v1.14.0-eksbuild.3` atau yang lebih baru dalam daftar dropdown Versi.

    b. Perluas pengaturan konfigurasi opsional.

    c. Masukkan kunci JSON tingkat atas `"nodeAgent":` dan nilai adalah objek dengan kunci `"enableCloudWatchLogs":` dan nilai `"true"` dalam nilai Konfigurasi. Teks yang dihasilkan harus berupa objek JSON yang valid. Contoh berikut menunjukkan kebijakan jaringan dan log kebijakan jaringan diaktifkan, dan log dikirim ke CloudWatch Log:

    ```
    {
        "enableNetworkPolicy": "true",
        "nodeAgent": {
            "enablePolicyEventLogs": "true",
            "enableCloudWatchLogs": "true",
        }
    }
    ```

Screenshot berikut menunjukkan contoh skenario ini.


EKS > Clusters > > Add-on > vpc-cni > Edit add-on
Configure Amazon VPC CNI
Amazon VPC CNI Info
Listed by
Category
networking
Status
Active
Version
Select the version for this add-on.
v1.17.1-eksbuild.1
Select IAM role
Select an IAM role to use with this add-on. To create a new role, follow the instructions in the Amazon EKS User Guide.
Optional configuration settings
Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.
{
"$ref": "#/definitions/VpcCni",
"$schema": "http://json-schema.org/draft-06/schema#",
"definitions": {
"Affinity": {
"type": [
"object",
"null"
]
},
"EniConfig": {
"additionalProperties": false,
Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.
1 {
2 "enableNetworkPolicy": "true",
3 "nodeAgent": {
4 "enablePolicyEventLogs": "true",
5 "enableCloudWatchLogs": "true"
6 }
7 }

AWS CLI

- Jalankan AWS CLI perintah berikut. Ganti `my-cluster` dengan nama cluster Anda dan ganti peran IAM ARN dengan peran yang Anda gunakan.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values '{"nodeAgent":
 {"enablePolicyEventLogs": "true", "enableCloudWatchLogs": "true"}}'
```

Add-on yang dikelola sendiri

Helm

Jika Anda telah menginstal Amazon VPC CNI plugin for Kubernetes melaluihelm, Anda dapat memperbarui konfigurasi untuk mengirim log kebijakan jaringan ke CloudWatch Log.

- Jalankan perintah berikut untuk mengaktifkan log kebijakan jaringan dan mengirimkannya ke CloudWatch Log.

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true --set
 nodeAgent.enableCloudWatchLogs=true aws-vpc-cni --namespace kube-system eks/
aws-vpc-cni
```

kubectl

1. Buka `aws-node DaemonSet` di editor Anda.

```
kubectl edit daemonset -n kube-system aws-node
```

2. Ganti `false` dengan `true` dalam dua argumen perintah `--enable-policy-event-logs=false` dan `--enable-cloudwatch-logs=false` dalam `aws-network-policy-agent` wadah dalam `args:` manifes daemonset VPC CNI`aws-node`.

```
    - args:
       - --enable-policy-event-logs=true
```

```
        - --enable-cloudwatch-logs=true
```

Mengirim log kebijakan jaringan dengan Fluent Bit daemonset

Jika Anda menggunakan Fluent Bit dalam daemonset untuk mengirim log dari node Anda, Anda dapat menambahkan konfigurasi untuk menyertakan log kebijakan jaringan dari kebijakan jaringan. Anda dapat menggunakan konfigurasi contoh berikut:

```
    [INPUT]
        Name              tail
        Tag               eksnp.*
        Path              /var/log/aws-routed-eni/network-policy-agent*.log
        Parser            json
        DB                /var/log/aws-routed-eni/flb_npagent.db
        Mem_Buf_Limit     5MB
        Skip_Long_Lines   On
        Refresh_Interval  10
```

Termasuk eBPF SDK

Amazon VPC CNI plugin for KubernetesMenginstal koleksi alat eBPF SDK pada node. Anda dapat menggunakan alat eBPF SDK untuk mengidentifikasi masalah dengan kebijakan jaringan. Misalnya, perintah berikut mencantumkan program yang berjalan pada node.

```
sudo /opt/cni/bin/aws-eks-na-cli ebpf progs
```

Untuk menjalankan perintah ini, Anda dapat menggunakan metode apa pun untuk terhubung ke node.

Kuberneteskebijakan jaringan

Untuk menerapkan kebijakan Kubernetes jaringan, Anda membuat Kubernetes `NetworkPolicy` objek dan menerapkannya ke klaster Anda. `NetworkPolicy`objek dicakup ke namespace. Anda menerapkan kebijakan untuk mengizinkan atau menolak lalu lintas Pods berdasarkan pemilih label, ruang nama, dan rentang alamat IP. Untuk informasi selengkapnya tentang membuat `NetworkPolicy` objek, lihat Kebijakan Jaringan dalam Kubernetes dokumentasi.

Penegakan Kubernetes `NetworkPolicy` objek diimplementasikan menggunakanExtended Berkeley Packet Filter (eBPF). Sehubungan dengan implementasi `iptables` berbasis, ia menawarkan

karakteristik latensi dan kinerja yang lebih rendah, termasuk pengurangan pemanfaatan CPU dan menghindari pencarian berurutan. Selain itu, eBPF probe menyediakan akses ke data kaya konteks yang membantu men-debug masalah tingkat kernel yang kompleks dan meningkatkan observabilitas. Amazon EKS mendukung eksportir eBPF berbasis yang memanfaatkan probe untuk mencatat hasil kebijakan pada setiap node dan mengekspor data ke pengumpul log eksternal untuk membantu debugging. Lihat informasi yang lebih lengkap dalam dokumentasi eBPF.

Jaringan khusus untuk pod

Secara default, ketika Amazon VPC CNI plugin for Kubernetes membuat antarmuka jaringan elastis sekunder (antarmuka jaringan) untuk node Amazon EC2 Anda, itu membuatnya dalam subnet yang sama dengan antarmuka jaringan utama node. Ini juga mengaitkan kelompok keamanan yang sama ke antarmuka jaringan sekunder yang terkait dengan antarmuka jaringan utama. Untuk satu atau beberapa alasan berikut, Anda mungkin ingin plugin membuat antarmuka jaringan sekunder di subnet yang berbeda atau ingin mengaitkan grup keamanan yang berbeda ke antarmuka jaringan sekunder, atau keduanya:

- Ada sejumlah `IPv4` alamat terbatas yang tersedia di subnet tempat antarmuka jaringan utama berada. Ini mungkin membatasi jumlah Pods yang dapat Anda buat di subnet. Dengan menggunakan subnet yang berbeda untuk antarmuka jaringan sekunder, Anda dapat meningkatkan jumlah `IPv4` alamat yang tersedia untuk. Pods
- Untuk alasan keamanan, Anda Pods mungkin perlu menggunakan subnet atau grup keamanan yang berbeda dari antarmuka jaringan utama node.
- Node dikonfigurasi dalam subnet publik, dan Anda ingin menempatkan subnet pribadi. Pods Tabel rute yang terkait dengan subnet publik mencakup rute ke gateway internet. Tabel rute yang terkait dengan subnet pribadi tidak menyertakan rute ke gateway internet.

Pertimbangan

- Dengan mengaktifkan jaringan khusus, tidak ada alamat IP yang ditetapkan ke antarmuka jaringan utama yang ditetapkanPods. Hanya alamat IP dari antarmuka jaringan sekunder yang ditugaskan ke`Pods`.
- Jika klaster Anda menggunakan `IPv6` keluarga, Anda tidak dapat menggunakan jaringan khusus.
- Jika Anda berencana untuk menggunakan jaringan khusus hanya untuk membantu mengurangi kelelahan `IPv4` alamat, Anda dapat membuat klaster menggunakan keluarga sebagai gantinya. `IPv6` Untuk informasi selengkapnya, lihat `IPv6`alamat untuk cluster,Pods, dan services.

- Meskipun Pods dikerahkan ke subnet yang ditentukan untuk antarmuka jaringan sekunder dapat menggunakan subnet dan grup keamanan yang berbeda dari antarmuka jaringan utama node, subnet dan grup keamanan harus berada dalam VPC yang sama dengan node.

Prasyarat

- Keakraban dengan bagaimana Amazon VPC CNI plugin for Kubernetes menciptakan antarmuka jaringan sekunder dan menetapkan alamat IP ke. Pods Untuk informasi lebih lanjut, lihat Alokasi ENI diGitHub.
- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl` versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.
- Kami menyarankan Anda menyelesaikan langkah-langkah dalam topik ini di shell Bash. Jika Anda tidak menggunakan shell Bash, beberapa perintah skrip seperti karakter kelanjutan baris dan cara variabel diatur dan digunakan memerlukan penyesuaian untuk shell Anda. Selain itu, aturan mengutip dan melarikan diri untuk shell Anda mungkin berbeda. Untuk informasi selengkapnya, lihat Menggunakan tanda kutip dengan string AWS CLI di AWS Command Line Interface Panduan Pengguna.

Untuk tutorial ini, kami sarankan menggunakan*`example values`*, kecuali jika dicatat untuk menggantikannya. Anda dapat mengganti apa pun *`example value`* saat menyelesaikan langkah-langkah untuk klaster produksi. Kami merekomendasikan untuk menyelesaikan semua langkah di terminal yang sama. Ini karena variabel diatur dan digunakan di seluruh langkah dan tidak akan ada di terminal yang berbeda.

Perintah dalam topik ini diformat menggunakan konvensi yang tercantum dalam Menggunakan contoh. AWS CLI Jika Anda menjalankan perintah dari baris perintah terhadap sumber daya yang berbeda Wilayah AWS dari default yang Wilayah AWS ditentukan dalam AWS CLI profil yang Anda gunakan, maka Anda perlu menambahkan **`--region region-code`** ke perintah.

Saat Anda ingin menerapkan jaringan khusus ke kluster produksi Anda, lewati keLangkah 2: Konfigurasikan VPC Anda.

Langkah 1: Buat tes VPC dan cluster

Untuk membuat klaster DB

Prosedur berikut membantu Anda membuat pengujian VPC dan cluster dan mengkonfigurasi jaringan khusus untuk cluster itu. Kami tidak menyarankan penggunaan klaster pengujian untuk beban kerja produksi karena beberapa fitur yang tidak terkait yang mungkin Anda gunakan pada klaster produksi tidak tercakup dalam topik ini. Untuk informasi selengkapnya, lihat Membuat klaster Amazon EKS.

1. Tentukan beberapa variabel untuk digunakan dalam langkah-langkah yang tersisa.

```
export cluster_name=my-custom-networking-cluster
account_id=$(aws sts get-caller-identity --query Account --output text)
```

2. Buat sebuah VPC.

   1. Buat VPC menggunakan template Amazon EKS AWS CloudFormation .

```
aws cloudformation create-stack --stack-name my-eks-custom-networking-vpc \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --parameters ParameterKey=VpcBlock,ParameterValue=192.168.0.0/24 \
  ParameterKey=PrivateSubnet01Block,ParameterValue=192.168.0.64/27 \
  ParameterKey=PrivateSubnet02Block,ParameterValue=192.168.0.96/27 \
  ParameterKey=PublicSubnet01Block,ParameterValue=192.168.0.0/27 \
  ParameterKey=PublicSubnet02Block,ParameterValue=192.168.0.32/27
```

      AWS CloudFormation Tumpukan membutuhkan waktu beberapa menit untuk membuatnya. Untuk memeriksa status penyebaran tumpukan, jalankan perintah berikut.

```
aws cloudformation describe-stacks --stack-name my-eks-custom-networking-vpc --
query Stacks\[\].StackStatus  --output text
```

Jangan melanjutkan ke langkah berikutnya sampai output dari perintah tersebut`CREATE_COMPLETE`.

2. Tentukan variabel dengan nilai ID subnet pribadi yang dibuat oleh template.

```
subnet_id_1=$(aws cloudformation describe-stack-resources --stack-name my-eks-
custom-networking-vpc \
    --query "StackResources[?
LogicalResourceId=='PrivateSubnet01'].PhysicalResourceId" --output text)
subnet_id_2=$(aws cloudformation describe-stack-resources --stack-name my-eks-
custom-networking-vpc \
    --query "StackResources[?
LogicalResourceId=='PrivateSubnet02'].PhysicalResourceId" --output text)
```

3. Tentukan variabel dengan Availability Zones dari subnet yang diambil pada langkah sebelumnya.

```
az_1=$(aws ec2 describe-subnets --subnet-ids $subnet_id_1 --query
 'Subnets[*].AvailabilityZone' --output text)
az_2=$(aws ec2 describe-subnets --subnet-ids $subnet_id_2 --query
 'Subnets[*].AvailabilityZone' --output text)
```

3. Buat peran IAM cluster.

   a. Jalankan perintah berikut untuk membuat file JSON kebijakan kepercayaan IAM.

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

b. Buat peran IAM cluster Amazon EKS. Jika perlu, kata pengantar `eks-cluster-role-trust-policy.json` dengan jalur di komputer Anda tempat Anda menulis file pada langkah sebelumnya. Perintah tersebut mengaitkan kebijakan kepercayaan yang Anda buat pada langkah sebelumnya dengan peran tersebut. Untuk membuat peran IAM, prinsipal IAM yang membuat peran harus diberi `iam:CreateRole` tindakan (izin).

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

c. Lampirkan kebijakan terkelola Amazon EKS yang diberi nama `AmazonEKSClusterPolicy`ke peran tersebut. Untuk melampirkan kebijakan IAM ke kepala sekolah IAM, prinsipal yang melampirkan kebijakan harus diberikan salah satu tindakan IAM berikut (izin): atau. `iam:AttachUserPolicy` `iam:AttachRolePolicy`

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

4. Buat kluster Amazon EKS dan konfigurasikan perangkat Anda untuk berkomunikasi dengannya.

a. Buat sebuah klaster.

```
aws eks create-cluster --name my-custom-networking-cluster \
   --role-arn arn:aws:iam::$account_id:role/
myCustomNetworkingAmazonEKSClusterRole \
   --resources-vpc-config subnetIds=$subnet_id_1","$subnet_id_2
```

> **Note**
>
> Mungkin error akan terjadi karena salah satu Availability Zone dalam permintaan Anda tidak memiliki kapasitas yang cukup untuk membuat klaster Amazon EKS. Jika hal ini terjadi, output galat berisi Availability Zones yang dapat mendukung klaster baru. Cobalah untuk kembali membuat klaster dengan setidaknya dua subnet yang terletak di Availability Zones yang didukung untuk akun Anda. Untuk informasi selengkapnya, lihat Kapasitas tidak mencukupi.

b. Cluster membutuhkan waktu beberapa menit untuk membuatnya. Untuk memeriksa status penyebaran cluster, jalankan perintah berikut.

```
aws eks describe-cluster --name my-custom-networking-cluster --query
 cluster.status
```

Jangan melanjutkan ke langkah berikutnya sampai output dari perintah tersebut`"ACTIVE"`.

c. Konfigurasikan `kubectl` untuk berkomunikasi dengan cluster Anda.

```
aws eks update-kubeconfig --name my-custom-networking-cluster
```

Langkah 2: Konfigurasikan VPC Anda

Tutorial ini membutuhkan VPC yang dibuat di. Langkah 1: Buat tes VPC dan cluster Untuk cluster produksi, sesuaikan langkah-langkah yang sesuai untuk VPC Anda dengan mengganti semua *`example values`* dengan milik Anda sendiri.

1. Konfirmasikan bahwa yang Anda instal saat Amazon VPC CNI plugin for Kubernetes ini adalah versi terbaru. Untuk menentukan versi terbaru untuk jenis add-on Amazon EKS dan memperbarui versi Anda ke sana, lihatMemperbarui add-on. Untuk menentukan versi terbaru untuk jenis add-on yang dikelola sendiri dan memperbarui versi Anda ke sana, lihat. Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS
2. Ambil ID VPC cluster Anda dan simpan dalam variabel untuk digunakan di langkah selanjutnya. Untuk cluster produksi, ganti *`my-custom-networking-cluster`*dengan nama cluster Anda.

```
vpc_id=$(aws eks describe-cluster --name my-custom-networking-cluster --query
 "cluster.resourcesVpcConfig.vpcId" --output text)
```

3. Kaitkan blok Classless Inter-Domain Routing (CIDR) tambahan dengan VPC klaster Anda. Blok CIDR tidak dapat tumpang tindih dengan blok CIDR terkait yang ada.

   1. Lihat blok CIDR saat ini yang terkait dengan VPC Anda.

```
aws ec2 describe-vpcs --vpc-ids $vpc_id \
    --query 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

Contoh output adalah sebagai berikut.

```
---------------------------------
```

```
|          DescribeVpcs          |
+-----------------+--------------+
|    CIDRBlock    |    State     |
+-----------------+--------------+
|  192.168.0.0/24 |  associated  |
+-----------------+--------------+
```

2. Kaitkan blok CIDR tambahan ke VPC Anda. Untuk informasi selengkapnya, lihat Mengaitkan blok IPv4 CIDR tambahan dengan VPC Anda di Panduan Pengguna Amazon VPC.

```
aws ec2 associate-vpc-cidr-block --vpc-id $vpc_id --cidr-block 192.168.1.0/24
```

3. Konfirmasikan bahwa blok baru dikaitkan.

```
aws ec2 describe-vpcs --vpc-ids $vpc_id --query
 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

Contoh output adalah sebagai berikut.

```
---------------------------------
|          DescribeVpcs          |
+-----------------+--------------+
|    CIDRBlock    |    State     |
+-----------------+--------------+
|  192.168.0.0/24 |  associated  |
|  192.168.1.0/24 |  associated  |
+-----------------+--------------+
```

Jangan lanjutkan ke langkah berikutnya sampai blok CIDR baru Anda selesai`State`. `associated`

4. Buat subnet sebanyak yang ingin Anda gunakan di setiap Availability Zone tempat subnet Anda ada. Tentukan blok CIDR yang ada di dalam blok CIDR yang Anda kaitkan dengan VPC Anda di langkah sebelumnya.

1. Buat subnet baru. Subnet harus dibuat di blok CIDR VPC yang berbeda dari subnet yang ada, tetapi di Availability Zone yang sama dengan subnet yang ada. Dalam contoh ini, satu subnet dibuat di blok CIDR baru di setiap Availability Zone tempat subnet pribadi saat ini ada. ID subnet yang dibuat disimpan dalam variabel untuk digunakan pada langkah selanjutnya.

NameNilai cocok dengan nilai yang ditetapkan ke subnet yang dibuat menggunakan template Amazon EKS VPC pada langkah sebelumnya. Nama tidak diperlukan. Anda dapat menggunakan nama yang berbeda.

```
new_subnet_id_1=$(aws ec2 create-subnet --vpc-id $vpc_id --availability-zone
 $az_1 --cidr-block 192.168.1.0/27 \
    --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=my-eks-
custom-networking-vpc-PrivateSubnet01},{Key=kubernetes.io/role/internal-
elb,Value=1}]' \
    --query Subnet.SubnetId --output text)
new_subnet_id_2=$(aws ec2 create-subnet --vpc-id $vpc_id --availability-zone
 $az_2 --cidr-block 192.168.1.32/27 \
    --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=my-eks-
custom-networking-vpc-PrivateSubnet02},{Key=kubernetes.io/role/internal-
elb,Value=1}]' \
    --query Subnet.SubnetId --output text)
```

> **Important**
>
> Secara default, subnet baru Anda secara implisit terkait dengan tabel rute utama VPC Anda. Tabel rute ini memungkinkan komunikasi antara semua sumber daya yang digunakan di VPC. Namun, itu tidak memungkinkan komunikasi dengan sumber daya yang memiliki alamat IP yang berada di luar blok CIDR yang terkait dengan VPC Anda. Anda dapat mengaitkan tabel rute Anda sendiri ke subnet Anda untuk mengubah perilaku ini. Untuk informasi selengkapnya, lihat Tabel rute subnet di Panduan Pengguna Amazon VPC.

2. Lihat subnet saat ini di VPC Anda.

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$vpc_id" \
    --query 'Subnets[*].{SubnetId: SubnetId,AvailabilityZone:
 AvailabilityZone,CidrBlock: CidrBlock}' \
    --output table
```

Contoh output adalah sebagai berikut.

```
-------------------------------------------------------------------
|                          DescribeSubnets                        |
+------------------+--------------------+--------------------------+
| AvailabilityZone |     CidrBlock      |         SubnetId         |
```

```
+-----------------+-------------------+---------------------------+
|  us-west-2d     |  192.168.0.0/27   |     subnet-example1       |
|  us-west-2a     |  192.168.0.32/27  |     subnet-example2       |
|  us-west-2a     |  192.168.0.64/27  |     subnet-example3       |
|  us-west-2d     |  192.168.0.96/27  |     subnet-example4       |
|  us-west-2a     |  192.168.1.0/27   |     subnet-example5       |
|  us-west-2d     |  192.168.1.32/27  |     subnet-example6       |
+-----------------+-------------------+---------------------------+
```

Anda dapat melihat subnet di blok `192.168.1.0` CIDR yang Anda buat berada di Availability Zone yang sama dengan subnet di blok CIDR. `192.168.0.0`

Langkah 3: Konfigurasikan Kubernetes sumber daya

Untuk mengkonfigurasi Kubernetes sumber daya

1. Atur variabel `AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG` lingkungan ke `true` dalam `aws-node` `DaemonSet`.

```
kubectl set env daemonset aws-node -n kube-system
 AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG=true
```

2. Ambil ID grup keamanan klaster Anda dan simpan dalam variabel untuk digunakan pada langkah berikutnya. Amazon EKS secara otomatis membuat grup keamanan ini saat Anda membuat klaster.

```
cluster_security_group_id=$(aws eks describe-cluster --name $cluster_name --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId --output text)
```

3. Buat sumber daya `ENIConfig` khusus untuk setiap subnet yang ingin Anda gunakanPods.

   a. Buat file unik untuk setiap konfigurasi antarmuka jaringan.

      Perintah berikut membuat `ENIConfig` file terpisah untuk dua subnet yang dibuat pada langkah sebelumnya. Nilai untuk `name` harus unik. Namanya sama dengan Availability Zone tempat subnet berada. Grup keamanan cluster ditugaskan ke file`ENIConfig`.

```
cat >$az_1.yaml <<EOF
apiVersion: crd.k8s.amazonaws.com/v1alpha1
kind: ENIConfig
```

```
metadata:
  name: $az_1
spec:
  securityGroups:
    - $cluster_security_group_id
  subnet: $new_subnet_id_1
EOF
```

```
cat >$az_2.yaml <<EOF
apiVersion: crd.k8s.amazonaws.com/v1alpha1
kind: ENIConfig
metadata:
  name: $az_2
spec:
  securityGroups:
    - $cluster_security_group_id
  subnet: $new_subnet_id_2
EOF
```

Untuk cluster produksi, Anda dapat membuat perubahan berikut pada perintah sebelumnya:

- Ganti *$cluster_security_group_id*dengan ID grup keamanan yang ada yang ingin Anda gunakan untuk masing-masing`ENIConfig`.
- Sebaiknya beri nama Anda `ENIConfigs` sama dengan Availability Zone yang akan Anda gunakan, bila memungkinkan. `ENIConfig` Anda mungkin perlu menggunakan nama yang berbeda untuk nama Availability Zone karena berbagai alasan. `ENIConfigs` Misalnya, jika Anda memiliki lebih dari dua subnet di Availability Zone yang sama dan ingin menggunakan keduanya dengan jaringan kustom, maka Anda memerlukan beberapa `ENIConfigs` untuk Availability Zone yang sama. Karena masing-masing `ENIConfig` memerlukan nama unik, Anda tidak dapat memberi nama lebih dari satu nama Anda `ENIConfigs` menggunakan nama Availability Zone.

  Jika `ENIConfig` nama Anda tidak semua sama dengan nama Availability Zone, maka ganti *$az_1* dan *$az_2* dengan nama Anda sendiri di perintah sebelumnya dan beri anotasi node Anda dengan yang `ENIConfig` nanti dalam tutorial ini.

> **Note**
>
> Jika Anda tidak menentukan grup keamanan yang valid untuk digunakan dengan cluster produksi dan Anda menggunakan:
>
> - versi `1.8.0` atau yang lebih baru Amazon VPC CNI plugin for Kubernetes, maka kelompok keamanan yang terkait dengan antarmuka elastis network primer node digunakan.
> - versi Amazon VPC CNI plugin for Kubernetes yang lebih awal dari`1.8.0`, maka grup keamanan default untuk VPC ditetapkan ke antarmuka jaringan sekunder.

> **Important**
>
> - `AWS_VPC_K8S_CNI_EXTERNALSNAT=false`adalah pengaturan default dalam konfigurasi untuk plugin Amazon VPC CNI untuk. Kubernetes Jika Anda menggunakan pengaturan default, lalu lintas yang ditujukan untuk alamat IP yang tidak berada dalam salah satu blok CIDR yang terkait dengan VPC Anda menggunakan grup keamanan dan subnet antarmuka jaringan utama node Anda. Subnet dan grup keamanan yang ditentukan dalam Anda `ENIConfigs` yang digunakan untuk membuat antarmuka jaringan sekunder tidak digunakan untuk lalu lintas ini. Untuk informasi selengkapnya tentang pengaturan ini, lihatSNAT untuk Pods.
> - Jika Anda juga menggunakan grup keamanan untukPods, grup keamanan yang ditentukan dalam a `SecurityGroupPolicy` digunakan, bukan grup keamanan yang ditentukan dalam`ENIConfigs`. Untuk informasi selengkapnya, lihat Kelompok keamanan untuk Pods.

b. Terapkan setiap file sumber daya kustom yang Anda buat ke cluster Anda dengan perintah berikut.

```
kubectl apply -f $az_1.yaml
kubectl apply -f $az_2.yaml
```

4. Konfirmasikan bahwa Anda `ENIConfigs` telah dibuat.

```
kubectl get ENIConfigs
```

Contoh output adalah sebagai berikut.

```
NAME         AGE
us-west-2a   117s
us-west-2d   105s
```

5. Jika Anda mengaktifkan jaringan khusus pada klaster produksi dan menamai `ENIConfigs` sesuatu selain Availability Zone tempat Anda menggunakannya, lewati ke langkah berikutnya untuk menerapkan node Amazon EC2.

   Aktifkan Kubernetes untuk menerapkan Availability Zone secara otomatis ke node Amazon EC2 baru yang dibuat di cluster Anda. `ENIConfig`

   1. Untuk test cluster dalam tutorial ini, lompat ke langkah berikutnya.

      Untuk cluster produksi, periksa untuk melihat apakah annotation dengan kunci `k8s.amazonaws.com/eniConfig` untuk variabel `ENI_CONFIG_ANNOTATION_DEF` lingkungan ada dalam spesifikasi kontainer untuk. `aws-node` `DaemonSet`

      ```
      kubectl describe daemonset aws-node -n kube-system | grep
       ENI_CONFIG_ANNOTATION_DEF
      ```

      Jika output dikembalikan, anotasi ada. Jika tidak ada output yang dikembalikan, maka variabel tidak diatur. Untuk cluster produksi, Anda dapat menggunakan pengaturan ini atau pengaturan di langkah berikut. Jika Anda menggunakan pengaturan ini, itu akan mengganti pengaturan pada langkah berikut. Dalam tutorial ini, pengaturan pada langkah berikutnya digunakan.

   2. Perbarui `aws-node` `DaemonSet` agar secara otomatis menerapkan Availability Zone ke node Amazon EC2 baru yang dibuat di cluster Anda. `ENIConfig`

      ```
      kubectl set env daemonset aws-node -n kube-system
       ENI_CONFIG_LABEL_DEF=topology.kubernetes.io/zone
      ```

Langkah 4: Menyebarkan node Amazon EC2

Untuk menyebarkan node Amazon EC2

1. Buat peran IAM node.

   a. Jalankan perintah berikut untuk membuat file JSON kebijakan kepercayaan IAM.

```
cat >node-role-trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

   b. Jalankan perintah berikut untuk menetapkan variabel untuk nama peran Anda. Anda dapat mengganti *myCustomNetworkingAmazonEKSNodeRole* dengan nama apa pun yang Anda pilih.

```
export node_role_name=myCustomNetworkingAmazonEKSNodeRole
```

   c. Buat peran IAM dan simpan Amazon Resource Name (ARN) yang dikembalikan dalam variabel untuk digunakan di langkah selanjutnya.

```
node_role_arn=$(aws iam create-role --role-name $node_role_name --assume-role-
policy-document file://"node-role-trust-relationship.json" \
    --query Role.Arn --output text)
```

   d. Lampirkan tiga kebijakan yang dikelola IAM yang diperlukan ke peran IAM.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name $node_role_name
aws iam attach-role-policy \
```

```
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name $node_role_name
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
    --role-name $node_role_name
```

> **Important**
>
> Untuk kesederhanaan dalam tutorial ini, AmazonEKS_CNI_Policykebijakan dilampirkan ke peran IAM node. Namun, dalam klaster produksi, kami merekomendasikan untuk melampirkan kebijakan ke peran IAM terpisah yang hanya digunakan dengan. Amazon VPC CNI plugin for Kubernetes Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).

2. Buat salah satu jenis grup node berikut. Untuk menentukan jenis instance yang ingin Anda terapkan, lihatMemilih jenis instans Amazon EC2. Untuk tutorial ini, lengkapi template Managed, Without a launch atau dengan template peluncuran tanpa opsi yang ditentukan ID AMI. Jika Anda akan menggunakan grup node untuk beban kerja produksi, sebaiknya Anda membiasakan diri dengan semua opsi grup node yang dikelola dan dikelola sendiri sebelum menerapkan grup node.

   - Dikelola - Terapkan grup node Anda menggunakan salah satu opsi berikut:
     - Tanpa template peluncuran atau dengan template peluncuran tanpa ID AMI yang ditentukan - Jalankan perintah berikut. Untuk tutorial ini, gunakan*example values*. Untuk grup node produksi, ganti semua *example values* dengan milik Anda sendiri. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa.

       ```
       aws eks create-nodegroup --cluster-name $cluster_name --nodegroup-name my-
       nodegroup \
           --subnets $subnet_id_1 $subnet_id_2 --instance-types t3.medium --node-role
        $node_role_arn
       ```

     - Dengan template peluncuran dengan ID AMI yang ditentukan
       1. Tentukan jumlah maksimum yang disarankan Amazon EKS Pods untuk node Anda. Ikuti instruksi diAmazon EKS merekomendasikan maksimum Pods untuk setiap jenis instans

Amazon EC2, tambahkan **`--cni-custom-networking-enabled`** ke langkah 3 dalam topik itu. Perhatikan output untuk digunakan pada langkah berikutnya.

2. Di template peluncuran Anda, tentukan ID AMI Amazon EKS yang dioptimalkan, atau AMI kustom yang dibuat dari AMI Amazon EKS yang dioptimalkan, lalu terapkan grup node menggunakan templat peluncuran dan berikan data pengguna berikut di template peluncuran. Data pengguna ini meneruskan argumen ke dalam `bootstrap.sh` file. Untuk informasi lebih lanjut tentang file bootstrap, lihat bootstrap.sh diGitHub. Anda dapat mengganti *`20`* dengan nilai dari langkah sebelumnya (disarankan) atau nilai Anda sendiri.

```
/etc/eks/bootstrap.sh my-cluster --use-max-pods false --kubelet-extra-args
 '--max-pods=20'
```

Jika Anda telah membuat AMI kustom yang tidak dibangun dari AMI Amazon EKS yang dioptimalkan, maka Anda perlu membuat konfigurasi sendiri.

- Dikelola sendiri

1. Tentukan jumlah maksimum yang disarankan Amazon EKS Pods untuk node Anda. Ikuti instruksi diAmazon EKS merekomendasikan maksimum Pods untuk setiap jenis instans Amazon EC2, tambahkan **`--cni-custom-networking-enabled`** ke langkah 3 dalam topik itu. Perhatikan output untuk digunakan pada langkah berikutnya.
2. Menyebarkan grup simpul menggunakan instruksi diMeluncurkan simpul Amazon Linux yang dikelola sendiri. Tentukan teks berikut untuk BootstrapArgumentsparameter. Anda dapat mengganti *`20`* dengan nilai dari langkah sebelumnya (disarankan) atau nilai Anda sendiri.

```
--use-max-pods false --kubelet-extra-args '--max-pods=20'
```

> **Note**
>
> Jika Anda ingin node dalam cluster produksi mendukung jumlah yang jauh lebih tinggiPods, jalankan skrip Amazon EKS merekomendasikan maksimum Pods untuk setiap jenis instans Amazon EC2 lagi. Juga, tambahkan **`--cni-prefix-delegation-enabled`** opsi ke perintah. Misalnya, *`110`* dikembalikan untuk tipe `m5.large` instance. Untuk petunjuk tentang cara mengaktifkan kemampuan ini, lihatTingkatkan jumlah

alamat IP yang tersedia untuk node Amazon EC2 Anda. Anda dapat menggunakan kemampuan ini dengan jaringan khusus.

Pembuatan grup node membutuhkan waktu beberapa menit. Anda dapat memeriksa status pembuatan grup node terkelola dengan perintah berikut.

```
aws eks describe-nodegroup --cluster-name $cluster_name --nodegroup-name my-nodegroup --query nodegroup.status --output text
```

Jangan melanjutkan ke langkah berikutnya sampai output yang dikembalikan`ACTIVE`.

3. Untuk tutorialnya, Anda dapat melewati langkah ini.

   Untuk cluster produksi, jika Anda tidak memberi nama yang sama `ENIConfigs` dengan Availability Zone yang Anda gunakan untuk mereka, maka Anda harus membubuhi keterangan node Anda dengan `ENIConfig` nama yang harus digunakan dengan node. Langkah ini tidak diperlukan jika Anda hanya memiliki satu subnet di setiap Availability Zone dan Anda menamai Anda `ENIConfigs` dengan nama yang sama dengan Availability Zones Anda. Ini karena secara Amazon VPC CNI plugin for Kubernetes otomatis mengaitkan yang benar `ENIConfig` dengan node untuk Anda ketika Anda mengaktifkannya untuk melakukannya pada langkah sebelumnya.

   a. Dapatkan daftar node di cluster Anda.

      ```
      kubectl get nodes
      ```

      Contoh output adalah sebagai berikut.

      ```
      NAME                                           STATUS   ROLES    AGE     VERSION
      ip-192-168-0-126.us-west-2.compute.internal   Ready    <none>   8m49s
       v1.22.9-eks-810597c
      ip-192-168-0-92.us-west-2.compute.internal    Ready    <none>   8m34s
       v1.22.9-eks-810597c
      ```

   b. Tentukan Availability Zone mana setiap node berada. Jalankan perintah berikut untuk setiap node yang dikembalikan pada langkah sebelumnya.

      ```
      aws ec2 describe-instances --filters Name=network-interface.private-dns-name,Values=ip-192-168-0-126.us-west-2.compute.internal \
      ```

```
--query 'Reservations[].Instances[].{AvailabilityZone:
 Placement.AvailabilityZone, SubnetId: SubnetId}'
```

Contoh output adalah sebagai berikut.

```
[
    {
        "AvailabilityZone": "us-west-2d",
        "SubnetId": "subnet-Example5"
    }
]
```

c. Anotasi setiap node dengan `ENIConfig` yang Anda buat untuk subnet ID dan Availability Zone. Anda hanya dapat membuat anotasi node dengan satu`ENIConfig`, meskipun beberapa node dapat dianotasi dengan yang sama. `ENIConfig` Ganti *`example values`* dengan milik Anda sendiri.

```
kubectl annotate node ip-192-168-0-126.us-west-2.compute.internal
 k8s.amazonaws.com/eniConfig=EniConfigName1
kubectl annotate node ip-192-168-0-92.us-west-2.compute.internal
 k8s.amazonaws.com/eniConfig=EniConfigName2
```

4. Jika Anda memiliki node dalam klaster produksi dengan berjalan Pods sebelum Anda beralih menggunakan fitur jaringan kustom, selesaikan tugas-tugas berikut:

   a. Pastikan Anda memiliki node yang tersedia yang menggunakan fitur jaringan kustom.

   b. Cordon dan tiriskan node untuk mematikan dengan anggun. Pods Untuk informasi selengkapnya, lihat Menguras Node dengan Aman dalam Kubernetes dokumentasi.

   c. Mengakhiri node. Jika node berada dalam grup node terkelola yang ada, Anda dapat menghapus grup node. Salin perintah yang mengikuti ke perangkat Anda. Buat modifikasi berikut pada perintah sesuai kebutuhan dan kemudian jalankan perintah yang dimodifikasi:

   - Ganti *`my-cluster`* dengan nama untuk cluster Anda.
   - Ganti *`my-nodegroup`* dengan nama untuk grup node Anda.

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

Hanya simpul-simpul baru yang terdaftar dengan label `k8s.amazonaws.com/eniConfig` yang menggunakan fitur jaringan kustom.

5. Konfirmasikan bahwa Pods alamat IP diberikan dari blok CIDR yang terkait dengan salah satu subnet yang Anda buat pada langkah sebelumnya.

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

Contoh output adalah sebagai berikut.

```
NAMESPACE     NAME                       READY   STATUS    RESTARTS   AGE     IP
           NODE                                            NOMINATED NODE   READINESS
 GATES
kube-system   aws-node-2rkn4             1/1     Running   0          7m19s
 192.168.0.92    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   aws-node-k96wp             1/1     Running   0          7m15s
 192.168.0.126   ip-192-168-0-126.us-west-2.compute.internal   <none>
 <none>
kube-system   coredns-657694c6f4-smcgr   1/1     Running   0          56m
 192.168.1.23    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   coredns-657694c6f4-stwv9   1/1     Running   0          56m
 192.168.1.28    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   kube-proxy-jgshq           1/1     Running   0          7m19s
 192.168.0.92    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   kube-proxy-wx9vk           1/1     Running   0          7m15s
 192.168.0.126   ip-192-168-0-126.us-west-2.compute.internal   <none>
 <none>
```

Anda dapat melihat bahwa alamat IP `coredns` Pods yang ditetapkan dari blok `192.168.1.0` CIDR yang Anda tambahkan ke VPC Anda. Tanpa jaringan khusus, mereka akan diberi alamat dari blok `192.168.0.0` CIDR, karena itu adalah satu-satunya blok CIDR yang awalnya terkait dengan VPC.

Jika Pod's `spec` berisi`hostNetwork=true`, itu diberikan alamat IP utama dari node. Itu tidak diberikan alamat dari subnet yang Anda tambahkan. Secara default, nilai ini diatur ke `false`. Nilai ini diatur ke `true` untuk `kube-proxy` and Amazon VPC CNI plugin for Kubernetes

(`aws-node`) Pods yang berjalan di cluster Anda. Inilah sebabnya mengapa `kube-proxy` dan plugin `aws-node` Pods tidak diberi `192.168.1.`*x* alamat di output sebelumnya. Untuk informasi selengkapnya tentang Pod's `hostNetwork` setelan, lihat inti PodSpec v1 di referensi Kubernetes API.

Langkah 5: Hapus sumber daya tutorial

Setelah Anda menyelesaikan tutorial, kami sarankan Anda menghapus sumber daya yang Anda buat. Anda kemudian dapat menyesuaikan langkah-langkah untuk mengaktifkan jaringan khusus untuk klaster produksi.

Untuk menghapus sumber daya tutorial

1. Jika grup node yang Anda buat hanya untuk pengujian, maka hapuslah.

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

   Bahkan setelah AWS CLI output mengatakan bahwa cluster dihapus, proses penghapusan mungkin tidak benar-benar lengkap. Proses penghapusan memakan waktu beberapa menit. Konfirmasikan bahwa itu selesai dengan menjalankan perintah berikut.

   ```
   aws eks describe-nodegroup --cluster-name $cluster_name --nodegroup-name my-
   nodegroup --query nodegroup.status --output text
   ```

   Jangan lanjutkan sampai output yang dikembalikan mirip dengan output berikut.

   ```
   An error occurred (ResourceNotFoundException) when calling the DescribeNodegroup
    operation: No node group found for name: my-nodegroup.
   ```

2. Jika grup node yang Anda buat hanya untuk pengujian, maka hapus peran IAM node.

   a. Lepaskan kebijakan dari peran.

      ```
      aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
      policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
      aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
      policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
      aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
      policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
      ```

b. Hapus peran.

```
aws iam delete-role --role-name myCustomNetworkingAmazonEKSNodeRole
```

3. Hapus klaster .

```
aws eks delete-cluster --name $cluster_name
```

Konfirmasikan cluster dihapus dengan perintah berikut.

```
aws eks describe-cluster --name $cluster_name --query cluster.status --output text
```

Ketika output yang mirip dengan berikut ini dikembalikan, cluster berhasil dihapus.

```
An error occurred (ResourceNotFoundException) when calling the DescribeCluster
 operation: No cluster found for name: my-cluster.
```

4. Hapus peran IAM cluster.

a. Lepaskan kebijakan dari peran.

```
aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSClusterRole
 --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy
```

b. Hapus peran.

```
aws iam delete-role --role-name myCustomNetworkingAmazonEKSClusterRole
```

5. Hapus subnet yang Anda buat pada langkah sebelumnya.

```
aws ec2 delete-subnet --subnet-id $new_subnet_id_1
aws ec2 delete-subnet --subnet-id $new_subnet_id_2
```

6. Hapus VPC yang Anda buat.

```
aws cloudformation delete-stack --stack-name my-eks-custom-networking-vpc
```

Tingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda

Setiap instans Amazon EC2 mendukung jumlah maksimum antarmuka jaringan elastis dan jumlah maksimum alamat IP yang dapat ditetapkan ke setiap antarmuka jaringan. Setiap node membutuhkan satu alamat IP untuk setiap antarmuka jaringan. Semua alamat IP lain yang tersedia dapat ditetapkanPods. Masing-masing Pod membutuhkan alamat IP sendiri. Akibatnya, Anda mungkin memiliki node yang memiliki sumber daya komputasi dan memori yang tersedia, tetapi tidak dapat mengakomodasi tambahan Pods karena node telah kehabisan alamat IP untuk ditetapkan. Pods

Dalam topik ini, Anda mempelajari cara meningkatkan jumlah alamat IP secara signifikan yang dapat ditetapkan node Pods dengan menetapkan awalan IP, daripada menetapkan alamat IP sekunder individual ke node Anda. Setiap awalan mencakup beberapa alamat IP. Jika Anda tidak mengonfigurasi klaster untuk penetapan awalan IP, klaster Anda harus membuat lebih banyak panggilan antarmuka pemrograman aplikasi (API) Amazon EC2 untuk mengonfigurasi antarmuka jaringan dan alamat IP yang diperlukan untuk konektivitas. Pod Saat cluster tumbuh ke ukuran yang lebih besar, frekuensi panggilan API ini dapat menyebabkan waktu peluncuran yang lebih lama Pod dan instans. Hal ini mengakibatkan penundaan penskalaan untuk memenuhi permintaan beban kerja yang besar dan runcing, dan menambah biaya dan overhead manajemen karena Anda perlu menyediakan cluster dan VPC tambahan untuk memenuhi persyaratan penskalaan. Untuk informasi selengkapnya, lihat Ambang batas Kubernetes skalabilitas pada. GitHub

Pertimbangan

- Setiap jenis instans Amazon EC2 mendukung jumlah maksimum. Pods Jika grup node terkelola Anda terdiri dari beberapa jenis instans, jumlah maksimum terkecil Pods untuk sebuah instance di cluster diterapkan ke semua node di cluster.
- Secara default, jumlah maksimum Pods yang dapat Anda jalankan pada node adalah 110, tetapi Anda dapat mengubah nomor itu. Jika Anda mengubah nomor dan memiliki grup node terkelola yang sudah ada, AMI berikutnya atau pembaruan template peluncuran grup node Anda akan menghasilkan node baru yang muncul dengan nilai yang diubah.
- Saat beralih dari menetapkan alamat IP ke menetapkan awalan IP, kami menyarankan Anda membuat grup node baru untuk meningkatkan jumlah alamat IP yang tersedia, daripada melakukan penggantian bergulir node yang ada. Berjalan Pods pada node yang memiliki alamat IP dan awalan yang ditetapkan dapat menyebabkan ketidakkonsistenan dalam kapasitas alamat IP yang diiklankan, yang berdampak pada beban kerja future pada node. Untuk cara yang disarankan dalam melakukan transisi, lihat Mengganti semua node selama migrasi dari mode IP Sekunder ke mode Delegasi Awalan atau sebaliknya dalam panduan praktik terbaik Amazon EKS.

- Untuk cluster dengan node Linux saja.
  - Setelah Anda mengonfigurasi add-on untuk menetapkan awalan ke antarmuka jaringan, Anda tidak dapat menurunkan versi Amazon VPC CNI plugin for Kubernetes add-on Anda ke versi yang lebih rendah dari `1.9.0` (atau`1.10.1`) tanpa menghapus semua node di semua grup node di cluster Anda.
  - Jika Anda juga menggunakan grup keamanan untukPods, with `POD_SECURITY_GROUP_ENFORCING_MODE` = `standard` dan `AWS_VPC_K8S_CNI_EXTERNALSNAT` =`false`, ketika Anda Pods berkomunikasi dengan titik akhir di luar VPC Anda, grup keamanan node digunakan, bukan grup keamanan apa pun yang telah Anda tetapkan untuk Anda. Pods

    Jika Anda juga menggunakan grup keamanan untuk Pods, with `POD_SECURITY_GROUP_ENFORCING_MODE` =`strict`, saat `Pods` berkomunikasi dengan titik akhir di luar VPC, grup keamanan `Pod's` akan digunakan.

Prasyarat

- Sebuah klaster yang sudah ada. Untuk menyebarkan satu, lihatMembuat klaster Amazon EKS.
- Subnet tempat node Amazon EKS Anda berada harus memiliki blok yang cukup berdekatan `/28` (untuk `IPv4` cluster) atau (untuk cluster) Classless Inter-Domain `IPv6` Routing `/80` (CIDR). Anda hanya dapat memiliki node Linux dalam sebuah `IPv6` cluster. Menggunakan awalan IP dapat gagal jika alamat IP tersebar di seluruh subnet CIDR. Kami merekomendasikan bahwa berikut:
  - Menggunakan reservasi CIDR subnet sehingga meskipun ada alamat IP dalam rentang cadangan masih digunakan, setelah dirilis, alamat IP tidak dipindahkan. Ini memastikan bahwa awalan tersedia untuk alokasi tanpa segmentasi.
  - Gunakan subnet baru yang secara khusus digunakan untuk menjalankan beban kerja yang awalan IP ditetapkan. Keduanya Windows dan Linux beban kerja dapat berjalan di subnet yang sama saat menetapkan awalan IP.
- Untuk menetapkan awalan IP ke node Anda, node Anda harus berbasis Nitro. AWS Instans yang tidak berbasis Nitro terus mengalokasikan alamat IP sekunder individual, tetapi memiliki jumlah alamat IP yang jauh lebih rendah untuk ditetapkan daripada instance. Pods Nitro-based
- Untuk cluster dengan Linux node saja — Jika cluster Anda dikonfigurasi untuk `IPv4` keluarga, Anda harus memiliki versi `1.9.0` atau versi Amazon VPC CNI plugin for Kubernetes add-on yang lebih baru diinstal. Anda dapat memeriksa versi Anda saat ini dengan perintah berikut.

```
kubectl describe daemonset aws-node --namespace kube-system | grep Image | cut -d "/"
 -f 2
```

Jika cluster Anda dikonfigurasi untuk `IPv6` keluarga, Anda harus menginstal `1.10.1` versi add-on. Jika versi plugin Anda lebih awal dari versi yang diperlukan, Anda harus memperbaruinya. Untuk informasi selengkapnya, lihat bagian pemutakhiranBekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS.

- Untuk cluster dengan Windows node saja
  - Cluster Anda dan versi platformnya harus di, atau lebih lambat dari versi dalam tabel berikut. Untuk memutakhirkan versi cluster Anda, lihatMemperbarui Kubernetes versi cluster Amazon EKS. Jika klaster Anda tidak pada versi platform minimum, maka Anda tidak dapat menetapkan awalan IP ke node Anda hingga Amazon EKS memperbarui versi platform Anda.

Versi Kubernetes	Versi platform
`1.27`	`eks.3`
`1.26`	`eks.4`
`1.25`	`eks.5`

    Anda dapat memeriksa versi Anda saat ini Kubernetes dan platform dengan mengganti *`my-cluster`* perintah berikut dengan nama cluster Anda dan kemudian menjalankan perintah yang dimodifikasi:**`aws eks describe-cluster --name my-cluster --query 'cluster.{"Kubernetes Version": version, "Platform Version": platformVersion}'`**.
  - Windowsdukungan diaktifkan untuk cluster Anda. Untuk informasi selengkapnya, lihat Mengaktifkan Windows dukungan untuk klaster Amazon EKS Anda.

Untuk meningkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda

1. Konfigurasikan cluster Anda untuk menetapkan awalan alamat IP ke node. Selesaikan prosedur pada tab yang cocok dengan sistem operasi node Anda.

Linux

1. Aktifkan parameter untuk menetapkan awalan ke antarmuka jaringan untuk Amazon VPC CNI. `DaemonSet` Saat Anda menerapkan klaster `1.21` atau yang lebih baru, versi `1.10.1` atau yang lebih baru dari Amazon VPC CNI plugin for Kubernetes add-on akan digunakan dengannya. Jika Anda membuat cluster dengan `IPv6` keluarga, pengaturan ini disetel ke secara `true` default. Jika Anda membuat cluster dengan `IPv4` keluarga, pengaturan ini disetel ke secara `false` default.

```
kubectl set env daemonset aws-node -n kube-system
 ENABLE_PREFIX_DELEGATION=true
```

> **Important**
>
> Bahkan jika subnet Anda memiliki alamat IP yang tersedia, jika subnet tidak memiliki `/28` blok bersebelahan yang tersedia, Anda akan melihat kesalahan berikut di log. Amazon VPC CNI plugin for Kubernetes
>
> ```
> InsufficientCidrBlocks: The specified subnet does not have enough free
>  cidr blocks to satisfy the request
> ```
>
> Hal ini dapat terjadi karena fragmentasi alamat IP sekunder yang ada tersebar di subnet. Untuk mengatasi kesalahan ini, buat subnet baru dan luncurkan di Pods sana, atau gunakan reservasi CIDR subnet Amazon EC2 untuk memesan ruang dalam subnet untuk digunakan dengan penetapan awalan. Untuk informasi selengkapnya, lihat Reservasi CIDR Subnet di Panduan Pengguna Amazon VPC.

2. Jika Anda berencana untuk menerapkan grup node terkelola tanpa templat peluncuran, atau dengan templat peluncuran yang belum Anda tentukan ID AMI, dan Anda menggunakan versi Amazon VPC CNI plugin for Kubernetes pada atau yang lebih baru dari versi yang tercantum dalam prasyarat, lalu lewati ke langkah berikutnya. Grup node terkelola secara otomatis menghitung jumlah maksimum Pods untuk Anda.

   Jika Anda menerapkan grup node yang dikelola sendiri atau grup node terkelola dengan templat peluncuran yang telah Anda tentukan ID AMI, Anda harus menentukan jumlah maksimum rekomendasi Amazon EKS Pods untuk node Anda. Ikuti instruksi diAmazon EKS merekomendasikan maksimum Pods untuk setiap jenis instans Amazon EC2,

tambahkan **`--cni-prefix-delegation-enabled`** ke langkah 3. Perhatikan output untuk digunakan di langkah selanjutnya.

> **Important**
>
> Grup node terkelola memberlakukan jumlah maksimum pada nilai. `maxPods` Untuk contoh dengan kurang dari 30 vCPU jumlah maksimum adalah 110 dan untuk semua contoh lainnya jumlah maksimum adalah 250. Jumlah maksimum ini diterapkan apakah delegasi awalan diaktifkan atau tidak.

3. Jika Anda menggunakan klaster `1.21` atau yang lebih baru yang dikonfigurasi`IPv6`, lewati ke langkah berikutnya.

   Tentukan parameter di salah satu opsi berikut. Untuk menentukan opsi mana yang tepat untuk Anda dan nilai apa yang harus disediakan untuk itu, lihat `WARM_PREFIX_TARGET`,`WARM_IP_TARGET`, dan `MINIMUM_IP_TARGET` seterusnyaGitHub.

   Anda dapat mengganti *`example values`* dengan nilai yang lebih besar dari nol.

   - `WARM_PREFIX_TARGET`

     ```
     kubectl set env ds aws-node -n kube-system WARM_PREFIX_TARGET=1
     ```

   - `WARM_IP_TARGET`atau `MINIMUM_IP_TARGET` — Jika salah satu nilai ditetapkan, itu akan menggantikan nilai yang ditetapkan untuk. `WARM_PREFIX_TARGET`

     ```
     kubectl set env ds aws-node -n kube-system WARM_IP_TARGET=5
     ```

     ```
     kubectl set env ds aws-node -n kube-system MINIMUM_IP_TARGET=2
     ```

4. Buat salah satu jenis grup node berikut dengan setidaknya satu jenis instans Amazon EC2 Nitro Amazon Linux 2. Untuk daftar jenis instans Nitro, lihat Instans yang dibangun di Sistem Nitro di Panduan Pengguna Amazon EC2 untuk Instans Linux. Kemampuan ini tidak didukung padaWindows. Untuk opsi yang disertakan*`110`*, ganti dengan nilai dari langkah 3 (disarankan), atau nilai Anda sendiri.

   - Self-managed — Menyebarkan grup node menggunakan instruksi di. Meluncurkan simpul Amazon Linux yang dikelola sendiri Tentukan teks berikut untuk BootstrapArgumentsparameter.

```
--use-max-pods false --kubelet-extra-args '--max-pods=110'
```

Jika Anda menggunakan `eksctl` untuk membuat grup node, Anda dapat menggunakan perintah berikut.

```
eksctl create nodegroup --cluster my-cluster --managed=false --max-pods-per-
node 110
```

- Dikelola - Terapkan grup node Anda menggunakan salah satu opsi berikut:
  - Tanpa template peluncuran atau dengan template peluncuran tanpa ID AMI yang ditentukan - Selesaikan prosedur diMembuat grup simpul terkelola. Grup node terkelola secara otomatis menghitung `max-pods` nilai yang direkomendasikan Amazon EKS untuk Anda.
  - Dengan template peluncuran dengan ID AMI tertentu — Dalam template peluncuran Anda, tentukan ID AMI Amazon EKS yang dioptimalkan, atau AMI kustom yang dibuat dari AMI Amazon EKS yang dioptimalkan, lalu terapkan grup node menggunakan template peluncuran dan berikan data pengguna berikut di template peluncuran. Data pengguna ini meneruskan argumen ke dalam `bootstrap.sh` file. Untuk informasi lebih lanjut tentang file bootstrap, lihat bootstrap.sh diGitHub.

    ```
    /etc/eks/bootstrap.sh my-cluster \
      --use-max-pods false \
      --kubelet-extra-args '--max-pods=110'
    ```

    Jika Anda menggunakan `eksctl` untuk membuat grup node, Anda dapat menggunakan perintah berikut.

    ```
    eksctl create nodegroup --cluster my-cluster --max-pods-per-node 110
    ```

    Jika Anda telah membuat AMI kustom yang tidak dibangun dari AMI Amazon EKS yang dioptimalkan, maka Anda perlu membuat konfigurasi sendiri.

> **Note**
>
> Jika Anda juga ingin menetapkan alamat IP Pods dari subnet yang berbeda dari instans, maka Anda perlu mengaktifkan kemampuan dalam langkah ini. Untuk informasi selengkapnya, lihat Jaringan khusus untuk pod.

Windows

1. Aktifkan penugasan awalan IP.
   a. Buka `amazon-vpc-cni` `ConfigMap` untuk mengedit.

   ```
   kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
   ```

   b. Tambahkan baris berikut ke `data` bagian.

   ```
     enable-windows-prefix-delegation: "true"
   ```

   c. Simpan file dan tutup editor.
   d. Konfirmasikan bahwa baris telah ditambahkan ke`ConfigMap`.

   ```
   kubectl get configmap -n kube-system amazon-vpc-cni -o
    "jsonpath={.data.enable-windows-prefix-delegation}"
   ```

   Jika output yang dikembalikan tidak`true`, maka mungkin ada kesalahan. Coba selesaikan langkahnya lagi.

   > **Important**
   >
   > Bahkan jika subnet Anda memiliki alamat IP yang tersedia, jika subnet tidak memiliki `/28` blok bersebelahan yang tersedia, Anda akan melihat kesalahan berikut dalam peristiwa node.
   >
   > ```
   > "failed to allocate a private IP/Prefix address:
   >  InsufficientCidrBlocks: The specified subnet does not have enough
   >  free cidr blocks to satisfy the request"
   > ```

Hal ini dapat terjadi karena fragmentasi alamat IP sekunder yang ada tersebar di subnet. Untuk mengatasi kesalahan ini, buat subnet baru dan luncurkan di Pods sana, atau gunakan reservasi CIDR subnet Amazon EC2 untuk memesan ruang dalam subnet untuk digunakan dengan penetapan awalan. Untuk informasi selengkapnya, lihat Reservasi CIDR Subnet di Panduan Pengguna Amazon VPC.

2. (Opsional) Tentukan konfigurasi tambahan untuk mengontrol perilaku pra-penskalaan dan penskalaan dinamis untuk klaster Anda. Untuk informasi selengkapnya, lihat Opsi konfigurasi dengan mode Delegasi Awalan aktif. Windows GitHub

   a. Buka `amazon-vpc-cni ConfigMap` untuk mengedit.

   ```
   kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
   ```

   b. Ganti *`example values`* dengan nilai lebih besar dari nol dan tambahkan entri yang Anda butuhkan ke `data` bagian`ConfigMap`. Jika Anda menetapkan nilai untuk salah satu `warm-ip-target` atau`minimum-ip-target`, nilai akan menggantikan nilai yang ditetapkan untuk. `warm-prefix-target`

   ```
   warm-prefix-target: "1"
   warm-ip-target: "5"
   minimum-ip-target: "2"
   ```

   c. Simpan file dan tutup editor.

3. Buat grup Windows node dengan setidaknya satu jenis Nitro instans Amazon EC2. Untuk daftar jenis Nitro instans, lihat Instans yang dibangun di Nitro Sistem di Panduan Pengguna Amazon Amazon EC2 untuk Instans Windows. Secara default, jumlah maksimum Pods yang dapat Anda gunakan ke node adalah 110. Jika Anda ingin menambah atau mengurangi angka itu, tentukan yang berikut ini dalam data pengguna untuk konfigurasi bootstrap. Ganti *`max-pods-quantity`* dengan nilai pod maks Anda.

   ```
   -KubeletExtraArgs '--max-pods=max-pods-quantity'
   ```

   Jika Anda menerapkan grup node terkelola, konfigurasi ini perlu ditambahkan dalam template peluncuran. Untuk informasi selengkapnya, lihat Menyesuaikan node terkelola dengan template peluncuran. Untuk informasi selengkapnya tentang parameter konfigurasi untuk skrip Windows bootstrap, lihatParameter konfigurasi skrip bootstrap.

2. Setelah node Anda di-deploy, lihat node di cluster Anda.

```
kubectl get nodes
```

Contoh output adalah sebagai berikut.

```
NAME                                               STATUS     ROLES    AGE   VERSION
ip-192-168-22-103.region-code.compute.internal   Ready      <none>   19m   v1.XX.X-
eks-6b7464
ip-192-168-97-94.region-code.compute.internal    Ready      <none>   19m   v1.XX.X-
eks-6b7464
```

3. Jelaskan salah satu node untuk menentukan nilai `max-pods` untuk node dan jumlah alamat IP yang tersedia. Ganti *192.168.30.193* dengan `IPv4` alamat atas nama salah satu node Anda yang dikembalikan pada output sebelumnya.

```
kubectl describe node ip-192-168-30-193.region-code.compute.internal | grep 'pods\|
PrivateIPv4Address'
```

Contoh output adalah sebagai berikut.

```
pods:                                       110
vpc.amazonaws.com/PrivateIPv4Address:  144
```

Pada output sebelumnya, `110` adalah jumlah maksimum Pods yang Kubernetes akan disebarkan ke node, meskipun *144* alamat IP tersedia.

Kelompok keamanan untuk Pods

Grup keamanan untuk Pods mengintegrasikan grup keamanan Amazon EC2 dengan. Kubernetes Pods Anda dapat menggunakan grup keamanan Amazon EC2 untuk menentukan aturan yang memungkinkan lalu lintas jaringan masuk dan keluar ke dan dari yang Anda gunakan ke node Pods yang berjalan di banyak jenis instans Amazon EC2 dan Fargate. Untuk penjelasan rinci tentang kemampuan ini, lihat Memperkenalkan grup keamanan untuk posting Pods blog.

Pertimbangan

- Sebelum menerapkan grup keamanan untukPods, pertimbangkan batasan dan ketentuan berikut:
- Grup keamanan untuk tidak Pods dapat digunakan dengan Windows node.

- Grup keamanan untuk Pods dapat digunakan dengan cluster yang dikonfigurasi untuk `IPv6` keluarga yang berisi node Amazon EC2 dengan menggunakan versi 1.16.0 atau yang lebih baru dari plugin Amazon VPC CNI. Anda dapat menggunakan grup keamanan untuk Pods dengan cluster mengkonfigurasi `IPv6` keluarga yang hanya berisi node Fargate dengan menggunakan versi 1.7.7 atau yang lebih baru dari plugin Amazon VPC CNI. Lihat informasi yang lebih lengkap di IPv6alamat untuk cluster,Pods, dan services
- Grup keamanan untuk Pods didukung oleh sebagian besar keluarga instans Amazon EC2 berbasis Nitro, meskipun tidak oleh semua generasi keluarga. Misalnya, keluarga dan generasi `m5` `c5r5`, `p3m6g`,`c6g`,,,, dan `r6g` contoh didukung. Tidak ada jenis instance dalam `t` keluarga yang didukung. Untuk daftar lengkap jenis instans yang didukung, lihat file limits.go di Github. Node Anda harus menjadi salah satu jenis instance terdaftar yang ada `IsTrunkingCompatible: true` di file itu.
- Jika Anda juga menggunakan kebijakan Pod keamanan untuk membatasi akses ke Pod mutasi, maka `eks:vpc-resource-controller` Kubernetes pengguna harus ditentukan dalam Kubernetes `ClusterRoleBinding` untuk `role` yang Anda `psp` tetapkan. Jika Anda menggunakan Amazon EKS default`psp`,`role`, dan`ClusterRoleBinding`, ini adalah `eks:podsecuritypolicy:authenticatedClusterRoleBinding`. Misalnya, Anda menambahkan pengguna ke `subjects:` bagian, seperti yang ditunjukkan pada contoh berikut:

```
[...]
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
    name: system:authenticated
  - apiGroup: rbac.authorization.k8s.io
    kind: User
    name: eks:vpc-resource-controller
  - kind: ServiceAccount
    name: eks-vpc-resource-controller
```

- Jika Anda menggunakan jaringan kustom dan grup keamanan untuk Pods bersama-sama, grup keamanan yang ditentukan oleh grup keamanan untuk Pods digunakan, bukan grup keamanan yang ditentukan dalam`ENIConfig`.
- Jika Anda menggunakan versi `1.10.2` atau sebelumnya dari plugin Amazon VPC CNI dan Anda menyertakan `terminationGracePeriodSeconds` pengaturan dalam Pod spesifikasi Anda, nilai untuk pengaturan tidak bisa nol.
- Jika Anda menggunakan versi `1.10` atau sebelumnya dari plugin Amazon VPC CNI, atau versi `1.11` dengan `POD_SECURITY_GROUP_ENFORCING_MODE =strict`, yang merupakan setelan

default, maka Kubernetes layanan jenis `NodePort` dan `LoadBalancer` menggunakan target instans dengan `externalTrafficPolicy` set ke `Local` tidak didukung dengan Pods yang Anda tetapkan ke grup keamanan. Untuk informasi selengkapnya tentang penggunaan penyeimbang beban dengan target instans, lihat Menyeimbangkan beban jaringan di Amazon EKS

- Jika Anda menggunakan versi `1.10` atau sebelumnya dari plugin Amazon VPC CNI atau versi `1.11` dengan `POD_SECURITY_GROUP_ENFORCING_MODE =strict`, yang merupakan pengaturan default, sumber NAT dinonaktifkan untuk lalu lintas keluar dari grup keamanan Pods yang ditetapkan sehingga aturan grup keamanan keluar diterapkan. Untuk mengakses internet, Pods dengan grup keamanan yang ditetapkan harus diluncurkan pada node yang digunakan dalam subnet pribadi yang dikonfigurasi dengan gateway atau instance NAT. Podsdengan kelompok keamanan yang ditugaskan dikerahkan ke subnet publik tidak dapat mengakses internet.

  Jika Anda menggunakan versi `1.11` atau yang lebih baru dari plugin dengan `POD_SECURITY_GROUP_ENFORCING_MODE =standard`, lalu Pod lintas yang ditujukan untuk di luar VPC diterjemahkan ke alamat IP antarmuka jaringan utama instans. Untuk lalu lintas ini, aturan dalam grup keamanan untuk antarmuka jaringan utama digunakan, bukan aturan dalam kelompok Pod's keamanan.
- Untuk menggunakan kebijakan Calico jaringan dengan Pods yang memiliki grup keamanan terkait, Anda harus menggunakan versi `1.11.0` atau yang lebih baru dari plugin Amazon VPC CNI dan set =. `POD_SECURITY_GROUP_ENFORCING_MODE standard` Jika tidak, arus lalu lintas ke dan dari Pods dengan grup keamanan terkait tidak dikenakan penegakan kebijakan Calico jaringan dan terbatas pada penegakan kelompok keamanan Amazon EC2 saja. Untuk memperbarui versi CNI VPC Amazon Anda, lihat Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS
- Podsberjalan di node Amazon EC2 yang menggunakan grup keamanan dalam cluster yang menggunakan Nodelocal DNScache hanya didukung dengan versi atau yang lebih baru `1.11.0` dari plugin Amazon VPC CNI dan dengan =. `POD_SECURITY_GROUP_ENFORCING_MODE standard` Untuk memperbarui versi plugin Amazon VPC CNI Anda, lihat Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS
- Grup keamanan untuk Pods dapat menyebabkan latensi Pod startup yang lebih tinggi untuk Pods dengan churn tinggi. Hal ini disebabkan pembatasan tingkat dalam pengontrol sumber daya.

Amazon VPC CNI plugin for KubernetesKonfigurasikan grup keamanan untuk Pods

Untuk menyebarkan grup keamanan untuk Pods

Jika Anda menggunakan grup keamanan Pods hanya untuk Fargate, dan tidak memiliki node Amazon EC2 di cluster Anda, lewati ke. Menyebarkan aplikasi contoh

1. Periksa Amazon VPC CNI plugin for Kubernetes versi Anda saat ini dengan perintah berikut:

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
 | cut -d : -f 3
```

Contoh output adalah sebagai berikut.

```
v1.7.6
```

Jika Amazon VPC CNI plugin for Kubernetes versi Anda lebih awal dari`1.7.7`, maka perbarui plugin ke versi `1.7.7` atau yang lebih baru. Lihat informasi yang lebih lengkap di Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS

2. Tambahkan kebijakan IAM `AmazonEKSVPCResourceController`terkelola ke peran klaster yang terkait dengan kluster Amazon EKS Anda. Kebijakan ini memungkinkan peran untuk mengelola antarmuka jaringan, alamat IP pribadi mereka, dan lampiran serta detasemen mereka ke dan dari instance jaringan.

   a. Ambil nama peran IAM cluster Anda dan simpan dalam variabel. Ganti *my-cluster* dengan nama klaster Anda.

```
cluster_role=$(aws eks describe-cluster --name my-cluster --query
 cluster.roleArn --output text | cut -d / -f 2)
```

   b. Lampirkan kebijakan pada peran tersebut.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSVPCResourceController --role-name $cluster_role
```

3. Aktifkan add-on Amazon VPC CNI untuk mengelola antarmuka jaringan Pods dengan menyetel variabel ke dalam`ENABLE_POD_ENI`. `true` `aws-node` DaemonSet Setelah pengaturan ini diatur ke`true`, untuk setiap node di cluster add-on membuat sumber daya kustom cninode. Pengendali

sumber daya VPC membuat dan melampirkan satu antarmuka jaringan khusus yang disebut antarmuka jaringan trunk dengan deskripsi `aws-k8s-trunk-eni`.

```
kubectl set env daemonset aws-node -n kube-system ENABLE_POD_ENI=true
```

> **Note**
>
> Antarmuka jaringan trunk termasuk dalam jumlah maksimum antarmuka jaringan yang didukung oleh tipe instans. Untuk daftar jumlah maksimum antarmuka jaringan yang didukung oleh setiap jenis instans, lihat alamat IP per antarmuka jaringan per jenis instans di Panduan Pengguna Amazon EC2 untuk Instans Linux. Jika simpul Anda sudah memiliki jumlah maksimum antarmuka jaringan standar yang terlampir padanya, maka pengendali sumber daya VPC akan menyimpan sebuah ruang. Anda harus mengurangi ukuran berjalan Anda Pods agar pengontrol dapat melepaskan dan menghapus antarmuka jaringan standar, membuat antarmuka jaringan trunk, dan melampirkannya ke instance.

4. Anda dapat melihat node mana yang memiliki sumber daya `CNINode` khusus dengan perintah berikut. Jika `No resources found` dikembalikan, maka tunggu beberapa detik dan coba lagi. Langkah sebelumnya membutuhkan restart Amazon VPC CNI plugin for KubernetesPods, yang memakan waktu beberapa detik.

```
$ kubectl get cninode -A
     NAME FEATURES
     ip-192-168-64-141.us-west-2.compute.internal
 [{"name":"SecurityGroupsForPods"}]
     ip-192-168-7-203.us-west-2.compute.internal [{"name":"SecurityGroupsForPods"}]
```

Jika Anda menggunakan versi VPC CNI yang lebih lama dari`1.15`, label node digunakan sebagai pengganti sumber daya khusus. `CNINode` Anda dapat melihat node mana yang memiliki label node `aws-k8s-trunk-eni` diatur `true` dengan perintah berikut. Jika `No resources found` dikembalikan, maka tunggu beberapa detik dan coba lagi. Langkah sebelumnya membutuhkan restart Amazon VPC CNI plugin for KubernetesPods, yang memakan waktu beberapa detik.

```
kubectl get nodes -o wide -l vpc.amazonaws.com/has-trunk-attached=true
-
```

Setelah antarmuka jaringan trunk dibuat, Pods diberikan alamat IP sekunder dari trunk atau antarmuka jaringan standar. Antarmuka trunk secara otomatis dihapus jika simpul dihapus.

Saat Anda menyebarkan grup keamanan untuk langkah selanjutnya, pengontrol sumber daya VPC membuat antarmuka jaringan khusus yang disebut antarmuka jaringan cabang dengan deskripsi `aws-k8s-branch-eni` dan mengaitkan grup keamanan ke Pod dalamnya. Antarmuka jaringan cabang dibuat sebagai tambahan dari antarmuka jaringan standar dan trunk yang terlampir pada simpul.

Jika Anda menggunakan probe keaktifan atau kesiapan, maka Anda juga perlu menonaktifkan demux awal TCP, sehingga `kubelet` dapat terhubung ke Pods antarmuka jaringan cabang menggunakan TCP. Untuk menonaktifkan demux awal TCP, jalankan perintah berikut:

```
kubectl patch daemonset aws-node -n kube-system \
  -p '{"spec": {"template": {"spec": {"initContainers": [{"env":
[{"name":"DISABLE_TCP_EARLY_DEMUX","value":"true"}],"name":"aws-vpc-cni-
init"}]}}}}'
```

> **Note**
>
> Jika Anda menggunakan `1.11.0` atau yang lebih baru dari Amazon VPC CNI plugin for Kubernetes add-on dan set `POD_SECURITY_GROUP_ENFORCING_MODE =standard`, seperti yang dijelaskan pada langkah berikutnya, maka Anda tidak perlu menjalankan perintah sebelumnya.

5. Jika klaster Anda menggunakan`NodeLocal DNSCache`, atau Anda ingin menggunakan kebijakan Calico jaringan dengan grup keamanan mereka sendiri, atau Anda memiliki Kubernetes layanan jenis `NodePort` dan `LoadBalancer` menggunakan target instans dengan `externalTrafficPolicy` set `Local` untuk Pods yang ingin Anda tetapkan grup keamanan, maka Anda harus menggunakan versi `1.11.0` atau versi Amazon VPC CNI plugin for Kubernetes add-on yang lebih baru, dan Anda harus mengaktifkan pengaturan berikut: Pods

```
kubectl set env daemonset aws-node -n kube-system
 POD_SECURITY_GROUP_ENFORCING_MODE=standard
```

> **Important**
>
> - PodAturan grup keamanan tidak diterapkan pada lalu lintas antara Pods atau di antara Pods danservices, seperti `kubelet` ataunodeLocalDNS, yang berada di node yang sama. Pod yang menggunakan grup keamanan berbeda pada node yang sama tidak dapat berkomunikasi karena mereka dikonfigurasi dalam subnet yang berbeda, dan routing dinonaktifkan di antara subnet ini.
> - Lalu lintas keluar dari Pods ke alamat di luar VPC adalah alamat jaringan yang diterjemahkan ke alamat IP antarmuka jaringan utama instans (kecuali jika Anda juga `AWS_VPC_K8S_CNI_EXTERNALSNAT=true` telah mengatur). Untuk lalu lintas ini, aturan dalam grup keamanan untuk antarmuka jaringan utama digunakan, bukan aturan dalam kelompok Pod's keamanan.
> - Agar pengaturan ini diterapkan ke yang adaPods, Anda harus memulai ulang Pods atau node yang Pods sedang berjalan.

Menyebarkan aplikasi contoh

Untuk menggunakan grup keamananPods, Anda harus memiliki grup keamanan yang ada dan Menerapkan Amazon EKS `SecurityGroupPolicy` ke klaster Anda, seperti yang dijelaskan dalam prosedur berikut. Langkah-langkah berikut menunjukkan kepada Anda cara menggunakan kebijakan grup keamanan untuk filePod. Kecuali dinyatakan lain, selesaikan semua langkah dari terminal yang sama karena variabel digunakan dalam langkah-langkah berikut yang tidak bertahan di seluruh terminal.

Untuk menyebarkan contoh Pod dengan grup keamanan

1. Buat Kubernetes namespace untuk menyebarkan sumber daya ke. Anda dapat mengganti *`namespace saya`* dengan nama namespace yang ingin Anda gunakan.

   ```
   kubectl create namespace my-namespace
   ```

2. Menerapkan Amazon EKS `SecurityGroupPolicy` ke cluster Anda.

   a. Salin konten berikut ke perangkat Anda. Anda dapat mengganti *`PodSelector`* dengan **`serviceAccountSelector`** jika Anda lebih suka memilih Pods berdasarkan label akun layanan. Anda harus menentukan satu selektor atau yang lainnya.

Sebuah kosong `podSelector` (contoh:`podSelector: {}`) memilih semua Pods di namespace. Anda dapat mengubah *`peran saya`* menjadi nama peran Anda. `serviceAccountSelector` yang kosong memilih semua akun layanan di namespace. Anda dapat mengganti *`my-security-group-policy`*dengan nama untuk namespace Anda `SecurityGroupPolicy` dan *`my-namespace`* dengan namespace yang ingin Anda buat. `SecurityGroupPolicy`

Anda harus mengganti *`my_pod_security_group_id dengan ID`* grup keamanan yang ada. Jika Anda tidak memiliki grup keamanan yang ada, maka Anda harus membuatnya. Untuk informasi selengkapnya, lihat Grup Keamanan Amazon EC2 untuk instans Linux di Panduan Pengguna Amazon EC2 untuk Instans Linux. Anda dapat menentukan 1-5 ID grup keamanan. Jika Anda menentukan lebih dari satu ID, maka kombinasi semua aturan di semua grup keamanan efektif untuk yang dipilihPods.

```
cat >my-security-group-policy.yaml <<EOF
apiVersion: vpcresources.k8s.aws/v1beta1
kind: SecurityGroupPolicy
metadata:
  name: my-security-group-policy
  namespace: my-namespace
spec:
  podSelector:
    matchLabels:
      role: my-role
  securityGroups:
    groupIds:
      - my_pod_security_group_id
EOF
```

Important

Grup keamanan atau grup yang Anda tentukan untuk Pod s Anda harus memenuhi kriteria berikut:

- Mereka harus ada. Jika tidak ada, maka, ketika Anda menerapkan Pod yang cocok dengan pemilih, Anda Pod tetap terjebak dalam proses pembuatan. Jika Anda mendeskripsikannyaPod, Anda akan melihat pesan kesalahan yang mirip dengan yang berikut:`An error occurred (InvalidSecurityGroupID.NotFound) when calling the`

```
CreateNetworkInterface operation: The securityGroup ID
'sg-05b1d815d1EXAMPLE' does not exist.
```

- Mereka harus mengizinkan komunikasi masuk dari grup keamanan yang diterapkan ke node Anda (untuk`kubelet`) melalui port apa pun yang telah Anda konfigurasi probe.
- Mereka harus mengizinkan komunikasi keluar TCP dan UDP port 53 ke grup keamanan yang ditugaskan ke Pods (atau node yang Pods dijalankanCoreDNS). Grup keamanan untuk Anda CoreDNS Pods harus mengizinkan lalu lintas masuk TCP dan UDP port 53 dari grup keamanan yang Anda tentukan.
- Mereka harus memiliki aturan masuk dan keluar yang diperlukan untuk berkomunikasi dengan orang lain Pods yang perlu mereka komunikasikan.
- Mereka harus memiliki aturan yang memungkinkan Pods untuk berkomunikasi dengan pesawat Kubernetes kontrol jika Anda menggunakan kelompok keamanan dengan Fargate. Cara termudah untuk melakukannya adalah dengan menentukan grup keamanan klaster sebagai salah satu grup keamanan.

Kebijakan grup keamanan hanya berlaku untuk yang baru dijadwalkanPods. Mereka tidak mempengaruhi berlariPods.

b. Deploy kebijakan.

```
kubectl apply -f my-security-group-policy.yaml
```

3. Menerapkan aplikasi sampel dengan label yang cocok dengan *my-role* nilai *podSelector* yang Anda tentukan pada langkah sebelumnya.

a. Salin konten berikut ke perangkat Anda. Ganti *nilai contoh* dengan milik Anda sendiri dan kemudian jalankan perintah yang dimodifikasi. Jika Anda *mengganti peran saya*, pastikan itu sama dengan nilai yang Anda tentukan untuk pemilih pada langkah sebelumnya.

```
cat >sample-application.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deployment
  namespace: my-namespace
```

```
  labels:
    app: my-app
spec:
  replicas: 4
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
        role: my-role
    spec:
      terminationGracePeriodSeconds: 120
      containers:
      - name: nginx
        image: public.ecr.aws/nginx/nginx:1.23
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: my-app
  namespace: my-namespace
  labels:
    app: my-app
spec:
  selector:
    app: my-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
EOF
```

b. Deploy aplikasi dengan perintah berikut. Saat Anda menerapkan aplikasi, Amazon VPC CNI plugin for Kubernetes pencocokan `role` label dan grup keamanan yang Anda tentukan pada langkah sebelumnya diterapkan kePod.

```
kubectl apply -f sample-application.yaml
```

4. Lihat yang Pods digunakan dengan aplikasi sampel. Untuk sisa topik ini, terminal ini disebut sebagai`TerminalA`.

```
kubectl get pods -n my-namespace -o wide
```

Contoh output adalah sebagai berikut.

```
NAME                             READY   STATUS    RESTARTS   AGE     IP
       NODE                                            NOMINATED NODE   READINESS
 GATES
my-deployment-5df6f7687b-4fbjm   1/1     Running   0          7m51s   192.168.53.48
    ip-192-168-33-28.region-code.compute.internal   <none>           <none>
my-deployment-5df6f7687b-j9fl4   1/1     Running   0          7m51s
  192.168.70.145   ip-192-168-92-33.region-code.compute.internal   <none>
 <none>
my-deployment-5df6f7687b-rjxcz   1/1     Running   0          7m51s
  192.168.73.207   ip-192-168-92-33.region-code.compute.internal   <none>
 <none>
my-deployment-5df6f7687b-zmb42   1/1     Running   0          7m51s   192.168.63.27
    ip-192-168-33-28.region-code.compute.internal   <none>           <none>
```

> **Note**
>
> - Jika Pods ada yang terjebak di`Waiting` negara bagian, maka jalankan`kubectl describe pod my-deployment-xxxxxxxxxx-xxxxx -n my-namespace`. Jika Anda melihatnya`Insufficient permissions: Unable to create Elastic Network Interface.`, konfirmasikan bahwa Anda menambahkan kebijakan IAM ke peran klaster IAM di langkah sebelumnya.
> - Jika Pods ada yang macet dalam`Pending` status, konfirmasikan bahwa jenis instance node Anda terdaftar `limits.go`dan bahwa produk dari jumlah maksimum antarmuka jaringan cabang yang didukung oleh jenis instance dikalikan kali jumlah node dalam grup node Anda belum terpenuhi. Misalnya, sebuah instans`m5.large` mendukung sembilan antarmuka jaringan cabang. Jika grup simpul Anda memiliki lima simpul, maka maksimum 45 antarmuka jaringan cabang dapat dibuat untuk grup simpul. Ke-46 Pod yang Anda coba terapkan akan berada dalam`Pending` status sampai Pod yang lain yang memiliki grup keamanan terkait dihapus.

Jika Anda menjalankan `kubectl describe pod` ***my-deployment-xxxxxxxxxx-xxxxx*** **-n** ***my-namespace*** dan melihat pesan yang serupa dengan pesan berikut, maka dapat diabaikan dengan aman. Pesan ini mungkin muncul ketika Amazon VPC CNI plugin for Kubernetes mencoba untuk mengatur jaringan host dan gagal saat antarmuka jaringan sedang dibuat. Plugin mencatat peristiwa ini sampai antarmuka jaringan dibuat.

```
Failed to create Pod sandbox: rpc error: code = Unknown desc = failed to set up
 sandbox container
 "e24268322e55c8185721f52df6493684f6c2c3bf4fd59c9c121fd4cdc894579f" network for Pod
 "my-deployment-5df6f7687b-4fbjm": networkPlugin
cni failed to set up Pod "my-deployment-5df6f7687b-4fbjm-c89wx_my-namespace"
 network: add cmd: failed to assign an IP address to container
```

Anda tidak dapat melebihi jumlah maksimum Pods yang dapat dijalankan pada jenis instance. Untuk daftar jumlah maksimum yang dapat Anda jalankan pada setiap jenis instance, lihat eni-max-pods.txt aktif. Pods GitHub Saat Anda menghapus Pod yang memiliki grup keamanan terkait, atau menghapus node yang Pod sedang berjalan, pengontrol sumber daya VPC menghapus antarmuka jaringan cabang. Jika Anda menghapus cluster dengan Pods menggunakan Pods untuk grup keamanan, maka pengontrol tidak menghapus antarmuka jaringan cabang, jadi Anda harus menghapusnya sendiri. Untuk informasi tentang cara menghapus antarmuka jaringan, lihat Menghapus antarmuka jaringan di Panduan Pengguna Amazon EC2 untuk Instans Linux.

5. Di terminal terpisah, masukkan ke salah satuPods. Untuk sisa topik ini, terminal ini disebut sebagai`TerminalB`. Ganti ***5df6f7687b - 4fbjm*** dengan ID salah satu yang dikembalikan dalam output Anda dari langkah sebelumnya. Pods

```
kubectl exec -it -n my-namespace my-deployment-5df6f7687b-4fbjm -- /bin/bash
```

6. Dari shell masuk`TerminalB`, konfirmasikan bahwa aplikasi sampel berfungsi.

```
curl my-app
```

Contoh output adalah sebagai berikut.

```
<!DOCTYPE html>
<html>
<head>
```

```
<title>Welcome to nginx!</title>
[...]
```

Anda menerima output karena semua Pods menjalankan aplikasi dikaitkan dengan grup keamanan yang Anda buat. Grup itu berisi aturan yang memungkinkan semua lalu lintas di antara semua Pods yang terkait dengan grup keamanan. Lalu lintas DNS diizinkan keluar dari grup keamanan tersebut ke grup keamanan klaster, yang terkait dengan node Anda. Node menjalankan CoreDNSPods, yang Anda Pods lakukan pencarian nama.

7. Dari`TerminalA`, hapus aturan grup keamanan yang memungkinkan komunikasi DNS ke grup keamanan klaster dari grup keamanan Anda. Jika Anda tidak menambahkan aturan DNS ke grup keamanan klaster pada langkah sebelumnya, ganti *`$my_cluster_security_group_id`* dengan ID grup keamanan tempat Anda membuat aturan.

```
aws ec2 revoke-security-group-ingress --group-id $my_cluster_security_group_id --
security-group-rule-ids $my_tcp_rule_id
aws ec2 revoke-security-group-ingress --group-id $my_cluster_security_group_id --
security-group-rule-ids $my_udp_rule_id
```

8. Dari`TerminalB`, coba akses aplikasi lagi.

```
curl my-app
```

Contoh output adalah sebagai berikut.

```
curl: (6) Could not resolve host: my-app
```

Upaya gagal karena Pod tidak lagi dapat mengakses CoreDNSPods, yang memiliki grup keamanan cluster yang terkait dengannya. Grup keamanan klaster tidak lagi memiliki aturan grup keamanan yang memungkinkan komunikasi DNS dari grup keamanan yang terkait dengan AndaPod.

Jika Anda mencoba mengakses aplikasi menggunakan alamat IP yang dikembalikan untuk salah satu langkah sebelumnya, Anda masih menerima respons karena semua port diizinkan antara Pods yang memiliki grup keamanan yang terkait dengannya dan pencarian nama tidak diperlukan. Pods

9. Setelah selesai bereksperimen, Anda dapat menghapus contoh kebijakan grup keamanan, aplikasi, dan grup keamanan yang Anda buat. Jalankan perintah berikut dari`TerminalA`.

```
kubectl delete namespace my-namespace
aws ec2 revoke-security-group-ingress --group-id $my_pod_security_group_id --
security-group-rule-ids $my_inbound_self_rule_id
wait
sleep 45s
aws ec2 delete-security-group --group-id $my_pod_security_group_id
```

Beberapa antarmuka jaringan untuk Pods

Multus CNI adalah plugin antarmuka jaringan kontainer (CNI) untuk Amazon EKS yang memungkinkan melampirkan beberapa antarmuka jaringan ke. Pod Untuk informasi lebih lanjut, lihat dokumentasi Multus-CNI di. GitHub

Di Amazon EKS, masing-masing Pod memiliki satu antarmuka jaringan yang ditetapkan oleh plugin Amazon VPC CNI. Dengan Multus, Anda dapat membuat multi-homed Pod yang memiliki banyak antarmuka. Ini dilakukan oleh Multus bertindak sebagai “meta-plugin”; plugin CNI yang dapat memanggil beberapa plugin CNI lainnya. AWSdukungan untuk Multus dikonfigurasi dengan plugin Amazon VPC CNI sebagai plugin delegasi default.

Pertimbangan-pertimbangan

- Amazon EKS tidak akan membangun dan menerbitkan virtualisasi I/O root tunggal (SR-IOV) dan plugin CNI Data Plane Development Kit (DPDK). Namun, Anda dapat mencapai akselerasi paket dengan menghubungkan langsung ke Amazon EC2 Elastic Network Adapters (ENA) melalui perangkat host dan plugin yang dikelola Multus. `ipvlan`
- Amazon EKS mendukung Multus, yang menyediakan proses generik yang memungkinkan rantai sederhana plugin CNI tambahan. Multus dan proses chaining didukung, tetapi tidak AWS akan memberikan dukungan untuk semua plugin CNI yang kompatibel yang dapat dirantai, atau masalah yang mungkin timbul pada plugin CNI yang tidak terkait dengan konfigurasi chaining.
- Amazon EKS menyediakan dukungan dan manajemen siklus hidup untuk plugin Multus, tetapi tidak bertanggung jawab atas alamat IP atau manajemen tambahan yang terkait dengan antarmuka jaringan tambahan. Alamat IP dan pengelolaan antarmuka jaringan default yang menggunakan plugin Amazon VPC CNI tetap tidak berubah.
- Hanya plugin Amazon VPC CNI yang secara resmi didukung sebagai plugin delegasi default. Anda perlu memodifikasi manifes instalasi Multus yang dipublikasikan untuk mengkonfigurasi ulang plugin delegasi default ke CNI alternatif jika Anda memilih untuk tidak menggunakan plugin Amazon VPC CNI untuk jaringan utama.

- Multus hanya didukung saat menggunakan Amazon VPC CNI sebagai CNI utama. Kami tidak mendukung Amazon VPC CNI saat digunakan untuk antarmuka pesanan lebih tinggi, sekunder atau lainnya.
- Untuk mencegah plugin Amazon VPC CNI mencoba mengelola antarmuka jaringan tambahan yang ditetapkanPods, tambahkan tag berikut ke antarmuka jaringan:

  kunci: `node.k8s.amazonaws.com/no_manage`

  nilai: `true`
- Multus kompatibel dengan kebijakan jaringan, tetapi kebijakan harus diperkaya untuk menyertakan port dan alamat IP yang mungkin menjadi bagian dari antarmuka jaringan tambahan yang dilampirkan. Pods

Untuk implementasi berjalan melalui, lihat Multus Setup Guide onGitHub.

Plugin CNI alternatif yang kompatibel

Amazon VPC CNI plugin for KubernetesIni adalah satu-satunya plugin CNI yang didukung oleh Amazon EKS. Amazon EKS berjalan di huluKubernetes, sehingga Anda dapat menginstal plugin CNI alternatif yang kompatibel ke node Amazon EC2 di cluster Anda. Jika Anda memiliki node Fargate di cluster Anda, itu sudah Amazon VPC CNI plugin for Kubernetes ada di node Fargate Anda. Ini satu-satunya plugin CNI yang dapat Anda gunakan dengan node Fargate. Upaya untuk menginstal plugin CNI alternatif pada node Fargate gagal.

Jika Anda berencana untuk menggunakan plugin CNI alternatif di node Amazon EC2, kami sarankan Anda mendapatkan dukungan komersial untuk plugin atau memiliki keahlian internal untuk memecahkan masalah dan berkontribusi perbaikan pada proyek plugin CNI.

Amazon EKS memelihara hubungan dengan jaringan mitra yang menawarkan dukungan untuk plugin CNI alternatif yang kompatibel. Untuk detail tentang versi, kualifikasi, dan pengujian yang dilakukan, lihat dokumentasi mitra berikut.

Mitra	Produk	Dokumentasi
Tigera	Calico	Instruksi instalasi
Isovalen	silia	Instruksi instalasi

Mitra	Produk	Dokumentasi
Juniper	Jaringan Contrail Cloud-Native (CN2)	Instruksi instalasi
VMware	Antrea	Instruksi instalasi

Amazon EKS bertujuan untuk memberikan berbagai pilihan untuk mencakup semua kasus penggunaan. Jika Anda mengembangkan plugin Kubernetes CNI yang didukung secara komersial yang tidak tercantum di sini, hubungi tim mitra kami di aws-container-partners@amazon .com untuk informasi lebih lanjut.

Apa itu AWS Load Balancer Controller?

AWS Load Balancer ControllerMengelola AWS Elastic Load Balancers untuk sebuah Kubernetes cluster. Anda dapat menggunakan controller untuk mengekspos aplikasi cluster Anda ke internet. Pengontrol menyediakan penyeimbang AWS beban yang mengarah ke sumber daya Layanan atau Ingress cluster. Dengan kata lain, controller membuat satu alamat IP atau nama DNS yang menunjuk ke beberapa pod di cluster Anda.

Pengontrol mengawasi Kubernetes Ingress atau Service sumber daya. Sebagai tanggapan, ini menciptakan sumber daya AWS Elastic Load Balancing yang sesuai. Anda dapat mengonfigurasi perilaku spesifik penyeimbang beban dengan menerapkan anotasi ke sumber daya. Kubernetes Misalnya, Anda dapat melampirkan grup AWS keamanan ke load balancer menggunakan anotasi.

Pengontrol menyediakan sumber daya berikut:

Kubernetes `Ingress`

LBC membuat AWS Application Load Balancer (ALB) saat Anda membuat file. Kubernetes `Ingress` Tinjau anotasi yang dapat Anda terapkan ke sumber daya Ingress.

Kubernetes`Layanan` dari `LoadBalancer` jenis

LBC membuat AWS Network Load Balancer (NLB) saat Anda membuat Kubernetes layanan tipe. `LoadBalancer` Tinjau anotasi yang dapat Anda terapkan ke sumber daya Layanan.

Di masa lalu, penyeimbang beban Kubernetes jaringan digunakan untuk target misalnya, tetapi LBC digunakan untuk target IP. Dengan AWS Load Balancer Controller versi `2.3.0` atau yang lebih baru, Anda dapat membuat NLB menggunakan salah satu jenis target. Untuk

informasi selengkapnya terkait tipe-tipe target NLB, lihat jenis Target di Panduan Pengguna untuk Penyeimbang Beban Jaringan.

Pengontrol adalah proyek sumber terbuka yang dikelola. GitHub

Sebelum menerapkan pengontrol, kami sarankan Anda meninjau prasyarat dan pertimbangan di dan. Penyeimbangan beban aplikasi pada Amazon EKS Menyeimbangkan beban jaringan di Amazon EKS Dalam topik tersebut, Anda akan menerapkan aplikasi sampel yang menyertakan penyeimbang AWS beban.

Menyebarkan Controller

- Pelajari caranyathe section called “Instal dengan Helm”.Gunakan prosedur ini jika Anda baru mengenal Amazon EKS. Prosedur ini menggunakan Helm, manajer paket untukKubernetes, dan `eksctl`untuk menyederhanakan menginstal LBC.
- Atau,the section called “Instal dengan Manifests”. Prosedur ini sesuai untuk konfigurasi cluster lanjutan. Ini termasuk cluster dengan akses jaringan terbatas ke pendaftar kontainer publik.

Hapus Versi Usang

- Jika Anda memiliki versi yang tidak digunakan lagi dari AWS Load Balancer Controller instalasi, pelajari caranya. the section called “Migrasi dari Deprecated Controller”
- Versi usang tidak dapat ditingkatkan. Mereka harus dihapus dan versi saat ini AWS Load Balancer Controller diinstal.
- Versi usang meliputi:
  - AWS ALB Ingress Controller untuk Kubernetes (“Ingress Controller”), pendahulu dari. AWS Load Balancer Controller
  - `0.1.`*x*Versi apa pun dari AWS Load Balancer Controller

Penyedia Cloud Legacy

Kubernetestermasuk penyedia cloud lama untuk AWS. Penyedia cloud lama mampu menyediakan penyeimbang AWS beban, mirip dengan. AWS Load Balancer Controller Penyedia cloud lama membuat Classic Load Balancer. Jika Anda tidak menginstalAWS Load Balancer Controller, Kubernetes akan default menggunakan penyedia cloud lama. Anda harus menginstal AWS Load Balancer Controller dan menghindari menggunakan penyedia cloud lama.

> **Important**
>
> Dalam versi 2.5 dan yang lebih baru, AWS Load Balancer Controller menjadi pengontrol default untuk sumber daya Kubernetes layanan dengan `type: LoadBalancer` dan membuat AWS Network Load Balancer (NLB) untuk setiap layanan. Ini dilakukan dengan membuat webhook yang bermutasi untuk layanan, yang menetapkan `spec.loadBalancerClass` bidang `service.k8s.aws/nlb` untuk layanan baru. `type: LoadBalancer` Anda dapat menonaktifkan fitur ini dan kembali menggunakan Cloud Provider lama sebagai pengontrol default, dengan menyetel nilai bagan helm ke. `enableServiceMutatorWebhook` `false` Cluster tidak akan menyediakan Classic Load Balancer baru untuk layanan Anda kecuali Anda menonaktifkan fitur ini. Classic Load Balancer yang ada akan terus bekerja.

Instal AWS Load Balancer Controller menggunakan Helm

Topik ini menjelaskan cara menginstal AWS Load Balancer Controller menggunakan Helm, manajer paket untukKubernetes, dan`eksctl`. Pengontrol diinstal dengan opsi default. Untuk informasi selengkapnya tentang pengontrol, termasuk detail tentang mengonfigurasinya dengan anotasi, lihat AWS Load Balancer ControllerDokumentasi di. GitHub

Pada langkah-langkah berikut, ganti *`example values`* dengan nilai-nilai Anda sendiri.

Prasyarat

Sebelum memulai tutorial ini, Anda harus menginstal dan mengonfigurasi alat-alat dan sumber daya yang Anda butuhkan berikut untuk membuat dan mengelola sebuah klaster Amazon EKS.

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS.
- Penyedia AWS Identity and Access Management (IAM) OpenID Connect (OIDC) yang sudah ada untuk cluster Anda. Untuk menentukan apakah Anda sudah memiliki satu, atau harus membuat satu, lihat Buat OIDC penyedia IAM untuk klaster Anda.
- Pastikan bahwa Amazon VPC CNI plugin for Kubernetes,`kube-proxy`, dan CoreDNS add-on Anda berada pada versi minimum yang tercantum dalam token akun Layanan.
- Keakraban dengan AWS Elastic Load Balancing. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna Penyeimbang Beban Elastis.

- Keakraban dengan layanan Kubernetes dan sumber daya ingress.
- Helm dipasang secara lokal.

Langkah 1: Buat Peran IAM menggunakan **`eksctl`**

> **Note**
>
> Anda hanya perlu membuat Peran IAM untuk AWS Load Balancer Controller satu per AWS akun. Periksa apakah `AmazonEKSLoadBalancerControllerRole` ada di Konsol IAM. Jika peran ini ada, lewati kethe section called “Langkah 2: Instal AWS Load Balancer Controller”.

Buat kebijakan IAM.

1. Unduh kebijakan IAM untuk AWS Load Balancer Controller yang memungkinkannya melakukan panggilan ke AWS API atas nama Anda.

   AWS

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy.json
   ```

   AWS GovCloud (US)

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy_us-gov.json
   ```

   ```
   $ mv iam_policy_us-gov.json iam_policy.json
   ```

2. Buat kebijakan IAM menggunakan kebijakan yang diunduh di langkah sebelumnya.

   ```
   $ aws iam create-policy \
       --policy-name AWSLoadBalancerControllerIAMPolicy \
       --policy-document file://iam_policy.json
   ```

> **Note**
>
> Jika Anda melihat kebijakan di AWS Management Console, konsol akan menampilkan peringatan untuk layanan ELB, tetapi tidak untuk layanan ELB v2. Ini terjadi karena beberapa tindakan dalam kebijakan ada untuk ELB v2, tetapi tidak untuk ELB. Anda dapat mengabaikan peringatan untuk ELB.

Buat Peran IAM menggunakan **`eksctl`**

- Ganti *`my-cluster`* dengan nama cluster Anda, *`111122223333`* dengan ID akun Anda, dan kemudian jalankan perintah. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
$ eksctl create iamserviceaccount \
  --cluster=my-cluster \
  --namespace=kube-system \
  --name=aws-load-balancer-controller \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --attach-policy-
arn=arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --approve
```

Langkah 2: Instal AWS Load Balancer Controller

Instal AWS Load Balancer Controller menggunakan Helm V3

1. Tambahkan repositori bagan `eks-charts` Helm. AWS mempertahankan repositori ini aktif. GitHub

```
$ helm repo add eks https://aws.github.io/eks-charts
```

2. Perbarui repo lokal Anda untuk memastikan bahwa Anda memiliki bagan terbaru.

```
$ helm repo update eks
```

3. Instal AWS Load Balancer Controller.

Ganti *my-cluster* dengan nama klaster Anda. Dalam perintah berikut, aws-load-balancer-controller adalah akun Kubernetes layanan yang Anda buat pada langkah sebelumnya.

Untuk informasi selengkapnya tentang mengonfigurasi bagan helm, lihat values.yamldi. GitHub

```
$ helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
  -n kube-system \
  --set clusterName=my-cluster \
  --set serviceAccount.create=false \
  --set serviceAccount.name=aws-load-balancer-controller
```

a. Jika Anda menerapkan pengontrol ke node Amazon EC2 yang membatasi akses ke layanan metadata instans Amazon EC2 (IMDS), atau jika Anda menerapkan ke Fargate, tambahkan flag berikut ke perintah berikut: helm

   - **--set region=*region-code***
   - **--set vpcId=*vpc-xxxxxxxx***

b. Untuk melihat versi Helm Chart dan Load Balancer Controller yang tersedia, gunakan perintah berikut:

   ```
   helm search repo eks/aws-load-balancer-controller --versions
   ```

> **Important**
>
> Bagan yang diterapkan tidak menerima pembaruan keamanan secara otomatis. Anda perlu memutakhirkan secara manual ke bagan yang lebih baru ketika bagan tersedia. Saat memutakhirkan, ubah *install* ke **upgrade** perintah sebelumnya. helm installPerintah secara otomatis menginstal definisi sumber daya kustom (CRDs) untuk pengontrol. helm upgradePerintah tidak. Jika Anda menggunakan, helm upgrade, Anda harus menginstal file secara manualCRDs. Jalankan perintah berikut untuk menginstalCRDs:
>
> ```
> wget https://raw.githubusercontent.com/aws/eks-charts/master/stable/aws-load-balancer-controller/crds/crds.yaml
> ```

```
kubectl apply -f crds.yaml
```

Langkah 3: Verifikasi bahwa pengontrol diinstal

1. Verifikasikan bahwa pengendali telah dipasang.

   ```
   $ kubectl get deployment -n kube-system aws-load-balancer-controller
   ```

   Contoh output adalah sebagai berikut.

   ```
   NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
   aws-load-balancer-controller   2/2     2            2           84s
   ```

   Anda menerima output sebelumnya jika Anda menerapkan menggunakan Helm. Jika Anda menerapkan menggunakan Kubernetes manifes, Anda hanya memiliki satu replika.
2. Sebelum menggunakan pengontrol untuk menyediakan AWS sumber daya, klaster Anda harus memenuhi persyaratan tertentu. Lihat informasi yang lebih lengkap di Penyeimbangan beban aplikasi pada Amazon EKS dan Menyeimbangkan beban jaringan di Amazon EKS.

Instal AWS Load Balancer Controller add-on menggunakan Kubernetes Manifests

Topik ini menjelaskan cara menginstal pengontrol dengan mengunduh dan menerapkan Kubernetes manifes. Anda dapat melihat dokumentasi lengkap untuk pengontrol aktifGitHub.

Pada langkah-langkah berikut, ganti *example values* dengan nilai-nilai Anda sendiri.

Prasyarat

Sebelum memulai tutorial ini, Anda harus menginstal dan mengonfigurasi alat-alat dan sumber daya yang Anda butuhkan berikut untuk membuat dan mengelola sebuah klaster Amazon EKS.

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS.
- Penyedia AWS Identity and Access Management (IAM) OpenID Connect (OIDC) yang sudah ada untuk klaster Anda. Untuk menentukan apakah Anda sudah memiliki satu, atau harus membuat satu, lihat Buat OIDC penyedia IAM untuk klaster Anda.

- Pastikan bahwa Amazon VPC CNI plugin for Kubernetes,`kube-proxy`, dan CoreDNS add-on Anda berada pada versi minimum yang tercantum dalam token akun Layanan.
- Keakraban dengan AWS Elastic Load Balancing. Untuk informasi selengkapnya, silakan lihat Panduan Pengguna Penyeimbang Beban Elastis.
- Keakraban dengan layanan Kubernetes dan sumber daya ingress.

Langkah 1: Konfigurasikan IAM

> **Note**
>
> Anda hanya perlu membuat Peran IAM untuk AWS Load Balancer Controller satu per AWS akun. Periksa apakah `AmazonEKSLoadBalancerControllerRole` ada di Konsol IAM. Jika peran ini ada, lewati kethe section called “Langkah 2: Instal `cert-manager`”.

Buat kebijakan IAM.

1. Unduh kebijakan IAM untuk AWS Load Balancer Controller yang memungkinkannya melakukan panggilan ke AWS API atas nama Anda.

   AWS

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy.json
   ```

   AWS GovCloud (US)

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy_us-gov.json
   ```

   ```
   $ mv iam_policy_us-gov.json iam_policy.json
   ```

2. Buat kebijakan IAM menggunakan kebijakan yang diunduh di langkah sebelumnya.

   ```
   $ aws iam create-policy \
       --policy-name AWSLoadBalancerControllerIAMPolicy \
       --policy-document file://iam_policy.json
   ```

> **Note**
>
> Jika Anda melihat kebijakan di AWS Management Console, konsol akan menampilkan peringatan untuk layanan ELB, tetapi tidak untuk layanan ELB v2. Ini terjadi karena beberapa tindakan dalam kebijakan ada untuk ELB v2, tetapi tidak untuk ELB. Anda dapat mengabaikan peringatan untuk ELB.

eksctl

Buat Peran IAM menggunakan **`eksctl`**

- Ganti *`my-cluster`* dengan nama cluster Anda, *`111122223333`* dengan ID akun Anda, dan kemudian jalankan perintah. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
$ eksctl create iamserviceaccount \
  --cluster=my-cluster \
  --namespace=kube-system \
  --name=aws-load-balancer-controller \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --attach-policy-
arn=arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --approve
```

AWS CLI and kubectl

Buat Peran IAM menggunakan dan AWS CLI**`kubectl`**

1. Ambil ID OIDC penyedia klaster Anda dan simpan dalam variabel.

```
oidc_id=$(aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
```

2. Tentukan apakah OIDC penyedia IAM dengan ID klaster Anda sudah ada di akun Anda. Anda perlu OIDC dikonfigurasi untuk cluster dan IAM.

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

Jika output dikembalikan, maka Anda sudah memiliki OIDC penyedia IAM untuk cluster Anda. Jika tidak ada output yang dikembalikan, maka Anda harus membuat OIDC penyedia IAM untuk cluster Anda. Untuk informasi selengkapnya, lihat Buat OIDC penyedia IAM untuk klaster Anda.

3. Salin konten berikut ke perangkat Anda. Ganti *111122223333* dengan ID akun Anda. Ganti *region-code* dengan tempat Wilayah AWS cluster Anda berada. Ganti *EXAMPLED539D4633E53DE1B71EXAMPLE* dengan output yang dikembalikan pada langkah sebelumnya. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:` Setelah mengganti teks, jalankan perintah yang dimodifikasi untuk membuat `load-balancer-role-trust-policy.json` file.

```
cat >load-balancer-role-trust-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:aws-load-balancer-controller"
                }
            }
        }
    ]
}
EOF
```

4. Buat peran IAM.

```
aws iam create-role \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --assume-role-policy-document file://"load-balancer-role-trust-policy.json"
```

5. Lampirkan kebijakan IAM terkelola Amazon EKS yang diperlukan ke peran IAM. Ganti *111122223333* dengan ID akun Anda.

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --role-name AmazonEKSLoadBalancerControllerRole
```

6. Salin konten berikut ke perangkat Anda. Ganti *111122223333* dengan ID akun Anda. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:` Setelah mengganti teks, jalankan perintah yang dimodifikasi untuk membuat `aws-load-balancer-controller-service-account.yaml` file.

```
cat >aws-load-balancer-controller-service-account.yaml <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  labels:
    app.kubernetes.io/component: controller
    app.kubernetes.io/name: aws-load-balancer-controller
  name: aws-load-balancer-controller
  namespace: kube-system
  annotations:
    eks.amazonaws.com/role-arn:
 arn:aws:iam::111122223333:role/AmazonEKSLoadBalancerControllerRole
EOF
```

7. Buat akun Kubernetes layanan di cluster Anda. Akun Kubernetes layanan bernama `aws-load-balancer-controller` dianotasi dengan peran IAM yang Anda buat bernama. *AmazonEKSLoadBalancerControllerRole*

```
$ kubectl apply -f aws-load-balancer-controller-service-account.yaml
```

Langkah 2: Instal **`cert-manager`**

Instal `cert-manager` menggunakan salah satu metode berikut untuk menyuntikkan konfigurasi sertifikat ke dalam webhooks. Untuk informasi selengkapnya, lihat Memulai `cert-manager` Dokumentasi.

Kami merekomendasikan menggunakan registri `quay.io` kontainer untuk menginstal`cert-manager`. Jika node Anda tidak memiliki akses ke registri `quay.io` kontainer, Instal `cert-manager` menggunakan Amazon ECR (lihat di bawah).

Quay.io

Instal **`cert-manager`** menggunakan Quay.io

- Jika node Anda memiliki akses ke registri `quay.io` kontainer, instal `cert-manager` untuk menyuntikkan konfigurasi sertifikat ke dalam webhooks.

```
$ kubectl apply \
    --validate=false \
    -f https://github.com/jetstack/cert-manager/releases/download/v1.13.5/cert-manager.yaml
```

Amazon ECR

Instal **`cert-manager`** menggunakan Amazon ECR

1. Instal `cert-manager` menggunakan salah satu metode berikut untuk menyuntikkan konfigurasi sertifikat ke dalam webhooks. Untuk informasi selengkapnya, lihat Memulai `cert-manager` Dokumentasi.
2. Unduh manifes.

```
curl -Lo cert-manager.yaml https://github.com/jetstack/cert-manager/releases/download/v1.13.5/cert-manager.yaml
```

3. Tarik gambar berikut dan dorong ke repositori yang dapat diakses oleh node Anda. Untuk informasi lebih lanjut tentang cara menarik, menandai, dan mendorong gambar ke repositori Anda sendiri, lihat. Salin gambar kontainer dari satu repositori ke repositori lain

```
quay.io/jetstack/cert-manager-cainjector:v1.13.5
quay.io/jetstack/cert-manager-controller:v1.13.5
```

```
quay.io/jetstack/cert-manager-webhook:v1.13.5
```

4. Ganti `quay.io` dalam manifes untuk tiga gambar dengan nama registri Anda sendiri. Perintah berikut mengasumsikan bahwa nama repositori pribadi Anda sama dengan repositori sumber. Ganti *111122223333.dkr.ecr.region-code.amazonaws.com* dengan registri pribadi Anda.

```
$ sed -i.bak -e 's|quay.io|111122223333.dkr.ecr.region-code.amazonaws.com|' ./
cert-manager.yaml
```

5. Terapkan manifes.

```
$ kubectl apply \
    --validate=false \
    -f ./cert-manager.yaml
```

Langkah 3: Instal AWS Load Balancer Controller

Instal AWS Load Balancer Controller menggunakan Kubernetes manifes

1. Unduh spesifikasi pengendali. Untuk informasi selengkapnya tentang pengontrol, lihat dokumentasi diGitHub.

```
curl -Lo v2_7_2_full.yaml https://github.com/kubernetes-sigs/aws-load-balancer-
controller/releases/download/v2.7.2/v2_7_2_full.yaml
```

2. Lakukan pengeditan berikut ke file.

   a. Jika Anda mengunduh `v2_7_2_full.yaml` file, jalankan perintah berikut untuk menghapus `ServiceAccount` bagian dalam manifes. Jika Anda tidak menghapus bagian ini, anotasi yang diperlukan yang Anda buat ke akun layanan pada langkah sebelumnya akan ditimpa. Menghapus bagian ini juga mempertahankan akun layanan yang Anda buat pada langkah sebelumnya jika Anda menghapus pengontrol.

```
$ sed -i.bak -e '596,604d' ./v2_7_2_full.yaml
```

   Jika Anda mengunduh versi file yang berbeda, buka file di editor dan hapus baris berikut.

```
apiVersion: v1
```

```
kind: ServiceAccount
metadata:
  labels:
    app.kubernetes.io/component: controller
    app.kubernetes.io/name: aws-load-balancer-controller
  name: aws-load-balancer-controller
  namespace: kube-system
---
```

b. Ganti `your-cluster-name` di `Deployment spec` bagian file dengan nama cluster Anda dengan mengganti *`my-cluster`* dengan nama cluster Anda.

```
$ sed -i.bak -e 's|your-cluster-name|my-cluster|' ./v2_7_2_full.yaml
```

c. Jika node Anda tidak memiliki akses ke repositori gambar Amazon EKS Amazon ECR, maka Anda perlu menarik gambar berikut dan mendorongnya ke repositori yang dapat diakses oleh node Anda. Untuk informasi selengkapnya tentang cara menarik, menandai, dan mendorong gambar ke repositori Anda sendiri, lihat. Salin gambar kontainer dari satu repositori ke repositori lain

```
public.ecr.aws/eks/aws-load-balancer-controller:v2.7.2
```

Tambahkan nama registri Anda ke manifes. Perintah berikut mengasumsikan bahwa nama repositori pribadi Anda sama dengan repositori sumber dan menambahkan nama registri pribadi Anda ke file. Ganti *`111122223333.dkr.ecr.region-code.amazonaws.com`* dengan registri Anda. Baris ini mengasumsikan bahwa Anda menamai repositori pribadi Anda sama dengan repositori sumber. Jika tidak, ubah `eks/aws-load-balancer-controller` teks setelah nama registri pribadi Anda ke nama repositori Anda.

```
$ sed -i.bak -e 's|public.ecr.aws/eks/aws-load-balancer-
controller|111122223333.dkr.ecr.region-code.amazonaws.com/eks/aws-load-
balancer-controller|' ./v2_7_2_full.yaml
```

d. (Diperlukan hanya untuk Fargate atau IMDS Terbatas)

Jika Anda menerapkan pengontrol ke node Amazon EC2 yang membatasi akses ke layanan metadata instans Amazon EC2 (IMDS), atau jika Anda menerapkan ke Fargate, tambahkan bagian bawah. **`following parameters`** `- args:`

```
[...]
```

```
spec:
      containers:
        - args:
            - --cluster-name=your-cluster-name
            - --ingress-class=alb
            - --aws-vpc-id=vpc-xxxxxxxx
            - --aws-region=region-code


[...]
```

3. Terapkan file.

```
$ kubectl apply -f v2_7_2_full.yaml
```

4. Unduh `IngressClass` dan `IngressClassParams` manifes ke cluster Anda.

```
$ curl -Lo v2_7_2_ingclass.yaml https://github.com/kubernetes-sigs/aws-load-balancer-controller/releases/download/v2.7.2/v2_7_2_ingclass.yaml
```

5. Menerapkan manifes ke klaster Anda.

```
$ kubectl apply -f v2_7_2_ingclass.yaml
```

Langkah 4: Verifikasi bahwa pengontrol diinstal

1. Verifikasikan bahwa pengendali telah dipasang.

```
$ kubectl get deployment -n kube-system aws-load-balancer-controller
```

Contoh output adalah sebagai berikut.

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   2/2     2            2           84s
```

Anda menerima output sebelumnya jika Anda menerapkan menggunakan Helm. Jika Anda menerapkan menggunakan Kubernetes manifes, Anda hanya memiliki satu replika.

2. Sebelum menggunakan pengontrol untuk menyediakan AWS sumber daya, klaster Anda harus memenuhi persyaratan tertentu. Lihat informasi yang lebih lengkap di Penyeimbangan beban aplikasi pada Amazon EKS dan Menyeimbangkan beban jaringan di Amazon EKS.

Migrasi dari Deprecated Controller

Topik ini menjelaskan cara bermigrasi dari versi pengontrol yang tidak digunakan lagi. Lebih khusus lagi, ini menjelaskan cara menghapus versi usang dari. AWS Load Balancer Controller

- Versi usang tidak dapat ditingkatkan. Mereka harus dihapus dan versi LBC saat ini diinstal.
- Versi usang meliputi:
  - AWS ALB Ingress Controller untuk Kubernetes (“Ingress Controller”), pendahulu dari. AWS Load Balancer Controller
  - `0.1.`*x*Versi apa pun dari AWS Load Balancer Controller

Hapus Versi Pengontrol Usang

> **Note**
>
> Anda mungkin telah menginstal versi usang menggunakan Helm atau secara manual dengan manifes. Kubernetes Selesaikan prosedur menggunakan alat yang awalnya Anda instal.

Hapus Ingress Controller menggunakan Helm

1. Jika Anda menginstal bagan `incubator/aws-alb-ingress-controller` Helm, hapus instalannya.

   ```
   $ helm delete aws-alb-ingress-controller -n kube-system
   ```

2. Jika Anda memiliki versi `0.1.`*x* `eks-charts/aws-load-balancer-controller` bagan yang diinstal, hapus instalannya. Pemutakhiran dari `0.1.`*x* ke versi `1.0.0` tidak berfungsi karena ketidakcocokan dengan versi API webhook.

   ```
   $ helm delete aws-load-balancer-controller -n kube-system
   ```

Hapus Ingress Controller menggunakan manifes Kubernetes

1. Periksa untuk melihat apakah pengendali terpasang saat ini.

   ```
   $ kubectl get deployment -n kube-system alb-ingress-controller
   ```

   Ini adalah output jika controller tidak diinstal.

   Kesalahan dari server (NotFound): deployments.apps "" tidak ditemukan alb-ingress-controller

   Ini adalah output jika pengontrol diinstal.

   ```
   NAME                      READY UP-TO-DATE AVAILABLE AGE
   alb-ingress-controller 1/1   1          1         122d
   ```

2. Masukkan perintah berikut untuk menghapus pengendali.

   ```
   $ kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-alb-
   ingress-controller/v1.1.8/docs/examples/alb-ingress-controller.yaml
   kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-alb-
   ingress-controller/v1.1.8/docs/examples/rbac-role.yaml
   ```

Migrasi ke AWS Load Balancer Controller

Untuk bermigrasi dari ALB Ingress Controller Kubernetes keAWS Load Balancer Controller, Anda perlu:

1. Hapus ALB Ingress Controller (lihat di atas).
2. InstalAWS Load Balancer Controller.
3. Tambahkan kebijakan tambahan ke Peran IAM yang digunakan oleh LBC. Kebijakan ini memungkinkan LBC untuk mengelola sumber daya yang dibuat oleh ALB Ingress Controller untuk. Kubernetes

Tambahkan Kebijakan Migrasi ke peran AWS Load Balancer Controller IAM.

1. Unduh kebijakan IAM. Kebijakan ini memungkinkan LBC untuk mengelola sumber daya yang dibuat oleh ALB Ingress Controller untuk. Kubernetes Anda juga dapat view the policy.

```
$ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
controller/v2.7.2/docs/install/iam_policy_v1_to_v2_additional.json
```

2. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:` .

```
$ sed -i.bak -e 's|arn:aws:|arn:aws-us-gov:|' iam_policy_v1_to_v2_additional.json
```

3. Buat kebijakan IAM dan catat ARN yang dikembalikan.

```
$ aws iam create-policy \
  --policy-name AWSLoadBalancerControllerAdditionalIAMPolicy \
  --policy-document file://iam_policy_v1_to_v2_additional.json
```

4. Lampirkan kebijakan IAM ke peran IAM yang digunakan oleh LBC. Ganti *`your-role-name`* dengan nama peran, seperti`AmazonEKSLoadBalancerControllerRole`.

   Jika Anda membuat peran menggunakan`eksctl`, maka untuk menemukan nama peran yang dibuat, buka AWS CloudFormation konsol dan pilih eksctl- ***`my-cluster`*** - - stack. addon-iamserviceaccount-kube-system aws-load-balancer-controller Pilih tab Sumber Daya. Nama peran ada di kolom ID Fisik. Jika cluster Anda berada di AWS GovCloud (AS-Timur) atau AWS GovCloud (AS-Barat) Wilayah AWS, maka ganti `arn:aws:` dengan. `arn:aws-us-gov:`

```
$ aws iam attach-role-policy \
  --role-name your-role-name \
  --policy-arn
 arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerAdditionalIAMPolicy
```

Bekerja dengan add-on CoreDNS Amazon EKS

CoreDNSadalah server DNS yang fleksibel dan dapat diperluas yang dapat berfungsi sebagai DNS Kubernetes cluster. Saat Anda meluncurkan klaster Amazon EKS dengan setidaknya satu node, dua replika CoreDNS gambar akan di-deploy secara default, terlepas dari jumlah node yang digunakan di cluster Anda. CoreDNSPodsMenyediakan resolusi nama untuk semua Pods di cluster. CoreDNSPodsDapat diterapkan ke node Fargate jika cluster Anda menyertakan dengan namespace AWS Fargate profil yang cocok dengan namespace untuk. CoreDNS `deployment` Untuk informasi selengkapnyaCoreDNS, lihat Menggunakan CoreDNS untuk Penemuan Layanan di Kubernetes dokumentasi.

Tabel berikut mencantumkan versi terbaru dari jenis add-on Amazon EKS untuk setiap Kubernetes versi.

Versi Kubernetes	1.29	1.28	1.27	1.26	1.25	1.24	1.23
	v1.11.1-eksbuild	v1.10.1-eksbuild	v1.10.1-eksbuild	v1.9.3-eksbuild.1	v1.9.3-eksbuild.1	v1.9.3-eksbuild.1	v1.8.7-eksbuild.10

> **Important**
>
> Jika Anda mengelola sendiri add-on ini, versi dalam tabel mungkin tidak sama dengan versi yang dikelola sendiri yang tersedia. Untuk informasi selengkapnya tentang memperbarui jenis pengaya yang dikelola sendiri, lihat. Memperbarui add-on yang dikelola sendiri

Pertimbangan CoreDNS peningkatan penting

- Untuk meningkatkan stabilitas dan ketersediaan CoreDNSDeployment, versi `v1.9.3-eksbuild.5` dan yang lebih baru dan `v1.10.1-eksbuild.2` digunakan dengan `PodDisruptionBudget` file. Jika Anda telah menerapkan yang sudah ada`PodDisruptionBudget`, pemutakhiran Anda ke versi ini mungkin gagal. Jika pemutakhiran gagal, menyelesaikan salah satu tugas berikut akan menyelesaikan masalah:
  - Saat melakukan pemutakhiran add-on Amazon EKS, pilih untuk mengganti pengaturan yang ada sebagai opsi resolusi konflik Anda. Jika Anda telah membuat pengaturan khusus lainnya keDeployment, pastikan untuk mencadangkan pengaturan Anda sebelum memutakhirkan sehingga Anda dapat menerapkan kembali pengaturan kustom Anda yang lain setelah peningkatan.
  - Hapus yang sudah ada `PodDisruptionBudget` dan coba upgrade lagi.
- Dalam versi add-on EKS `v1.9.3-eksbuild.3` dan yang lebih baru `v1.10.1-eksbuild.6` dan yang lebih baru, CoreDNS Deployment set `readinessProbe` untuk menggunakan `/ready` endpoint. Titik akhir ini diaktifkan dalam file `Corefile` konfigurasi untukCoreDNS.

  Jika Anda menggunakan kustom`Corefile`, Anda harus menambahkan `ready` plugin ke konfigurasi, sehingga `/ready` titik akhir aktif CoreDNS untuk digunakan probe.

- Dalam versi add-on EKS `v1.9.3-eksbuild.7` dan yang lebih baru `v1.10.1-eksbuild.4` dan yang lebih baru, Anda dapat mengubah file. `PodDisruptionBudget` Anda dapat mengedit add-on dan mengubah pengaturan ini di Pengaturan konfigurasi opsional menggunakan bidang dalam contoh berikut. Contoh ini menunjukkan default`PodDisruptionBudget`.

```
{
    "podDisruptionBudget": {
        "enabled": true,
        "maxUnavailable": 1
        }
}
```

  Anda dapat mengatur `maxUnavailable` atau`minAvailable`, tetapi Anda tidak dapat mengatur keduanya dalam satu`PodDisruptionBudget`. Untuk informasi selengkapnya`PodDisruptionBudgets`, lihat Menentukan a `PodDisruptionBudget` dalam Kubernetesdokumentasi.

  Perhatikan bahwa jika Anda menyetel `enabled` ke`false`, `PodDisruptionBudget` tidak dihapus. Setelah Anda mengatur bidang ini`false`, Anda harus menghapus `PodDisruptionBudget` objek. Demikian pula, jika Anda mengedit add-on untuk menggunakan versi add-on yang lebih lama (menurunkan versi add-on) setelah memutakhirkan ke versi dengan a`PodDisruptionBudget`, add-on tidak dihapus. `PodDisruptionBudget` Untuk menghapus`PodDisruptionBudget`, Anda dapat menjalankan perintah berikut:

```
kubectl delete poddisruptionbudget coredns -n kube-system
```

- Dalam versi add-on EKS `v1.10.1-eksbuild.5` dan yang lebih baru, ubah toleransi default dari `node-role.kubernetes.io/master:NoSchedule` menjadi `node-role.kubernetes.io/control-plane:NoSchedule` mematuhi KEP 2067. Untuk informasi lebih lanjut tentang KEP 2067, lihat KEP-2067: Ganti nama label “master” kubeadm dan taint di Kubernetes Enhancement Proposals (KEPs) di. GitHub

  Dalam versi add-on EKS `v1.8.7-eksbuild.8` dan yang lebih baru `v1.9.3-eksbuild.9` dan yang lebih baru, kedua toleransi diatur agar kompatibel dengan setiap Kubernetes versi.

- Dalam versi add-on EKS `v1.9.3-eksbuild.11` `v1.10.1-eksbuild.7` dan yang lebih baru, CoreDNS Deployment menetapkan nilai default untuk`topologySpreadConstraints`. Nilai default memastikan bahwa CoreDNS Pods tersebar di Availability Zone jika ada node di beberapa

Availability Zone yang tersedia. Anda dapat menetapkan nilai kustom yang akan digunakan sebagai pengganti nilai default. Nilai default berikut:

```
topologySpreadConstraints:
  - maxSkew: 1
    topologyKey: topology.kubernetes.io/zone
    whenUnsatisfiable: ScheduleAnyway
    labelSelector:
      matchLabels:
        k8s-app: kube-dns
```

CoreDNSv **`1.11`** meningkatkan pertimbangan

- Dalam versi add-on EKS `v1.11.1-eksbuild.4` dan yang lebih baru, gambar kontainer didasarkan pada gambar dasar minimal yang dikelola oleh Amazon EKS Distro, yang berisi paket minimal dan tidak memiliki cangkang. Untuk informasi selengkapnya, lihat Amazon EKS Distro. Penggunaan dan pemecahan masalah CoreDNS gambar tetap sama.

Membuat add-on Amazon EKS

Buat jenis add-on Amazon EKS. Memeriksa

Prasyarat

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS.

1. Lihat versi add-on mana yang diinstal pada cluster Anda.

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -d : -f 3
   ```

   Contoh output adalah sebagai berikut.

   ```
   v1.10.1-eksbuild.7
   ```

2. Lihat jenis add-on yang diinstal pada cluster Anda. Bergantung pada alat yang digunakan untuk membuat klaster, saat ini Anda mungkin tidak menginstal jenis add-on Amazon EKS di cluster Anda. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

   Jika nomor versi dikembalikan, Anda memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda dan tidak perlu menyelesaikan langkah-langkah yang tersisa dalam prosedur ini. Jika kesalahan dikembalikan, Anda tidak memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda. Selesaikan langkah-langkah yang tersisa dari prosedur ini untuk menginstalnya.

3. Simpan konfigurasi add-on yang Anda instal saat ini.

```
kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
```

4. Buat add-on menggunakan. AWS CLI Jika Anda ingin menggunakan AWS Management Console atau `eksctl` untuk membuat add-on, lihat Membuat add-on dan tentukan nama `coredns` add-on. Salin perintah yang mengikuti ke perangkat Anda. Buat modifikasi berikut pada perintah, sesuai kebutuhan, dan kemudian jalankan perintah yang dimodifikasi.

   - Ganti *my-cluster* dengan nama klaster Anda.
   - Ganti *v1.11.1-eksbuild.6* dengan versi terbaru yang tercantum dalam tabel versi terbaru untuk versi cluster Anda.

```
aws eks create-addon --cluster-name my-cluster --addon-name coredns --addon-
version v1.11.1-eksbuild.6
```

   Jika Anda telah menerapkan pengaturan khusus ke add-on saat ini yang bertentangan dengan pengaturan default add-on Amazon EKS, pembuatan mungkin gagal. Jika pembuatan gagal, Anda menerima kesalahan yang dapat membantu Anda menyelesaikan masalah. Atau, Anda dapat menambahkan **`--resolve-conflicts OVERWRITE`** ke perintah sebelumnya. Ini memungkinkan add-on untuk menimpa pengaturan kustom yang ada. Setelah Anda membuat add-on, Anda dapat memperbaruinya dengan pengaturan khusus Anda.

5. Konfirmasikan bahwa versi terbaru add-on untuk Kubernetes versi klaster Anda telah ditambahkan ke klaster Anda. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

Mungkin perlu beberapa detik untuk menyelesaikan pembuatan add-on.

Contoh output adalah sebagai berikut.

```
v1.11.1-eksbuild.6
```

6. Jika Anda membuat pengaturan khusus untuk add-on asli Anda, sebelum Anda membuat add-on Amazon EKS, gunakan konfigurasi yang Anda simpan di langkah sebelumnya untuk memperbarui add-on Amazon EKS dengan pengaturan khusus Anda.

Memperbarui add-on Amazon EKS

Perbarui jenis add-on Amazon EKS. Jika Anda belum menambahkan jenis add-on Amazon EKS ke klaster Anda, tambahkan atau lihatMemperbarui add-on yang dikelola sendiri, alih-alih menyelesaikan prosedur ini.

1. Lihat versi add-on mana yang diinstal pada cluster Anda. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 "addon.addonVersion" --output text
```

Contoh output adalah sebagai berikut.

```
v1.10.1-eksbuild.7
```

Jika versi yang dikembalikan sama dengan versi untuk versi cluster Anda di tabel versi terbaru, maka Anda sudah menginstal versi terbaru di cluster Anda dan tidak perlu menyelesaikan sisa prosedur ini. Kubernetes Jika Anda menerima kesalahan, alih-alih nomor versi dalam output Anda, maka Anda tidak memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda. Anda perlu membuat add-on sebelum Anda dapat memperbaruinya dengan prosedur ini.

2. Simpan konfigurasi add-on yang Anda instal saat ini.

```
kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
```

3. Perbarui add-on Anda menggunakan. AWS CLI Jika Anda ingin menggunakan AWS Management Console atau `eksctl` memperbarui add-on, lihatMemperbarui add-on. Salin perintah yang mengikuti ke perangkat Anda. Buat modifikasi berikut pada perintah, sesuai kebutuhan, dan kemudian jalankan perintah yang dimodifikasi.

   - Ganti *my-cluster* dengan nama klaster Anda.
   - Ganti *v1.11.1-eksbuild.6* dengan versi terbaru yang tercantum dalam tabel versi terbaru untuk versi cluster Anda.
   - Opsi **--resolve-conflicts** *PRESERVE mempertahankan* nilai konfigurasi yang ada untuk add-on. Jika Anda telah menetapkan nilai kustom untuk pengaturan add-on, dan Anda tidak menggunakan opsi ini, Amazon EKS menimpa nilai Anda dengan nilai defaultnya. Jika Anda menggunakan opsi ini, kami sarankan untuk menguji setiap bidang dan perubahan nilai pada klaster non-produksi sebelum memperbarui add-on pada cluster produksi Anda. Jika Anda mengubah nilai ini menjadi`OVERWRITE`, semua pengaturan diubah ke nilai default Amazon EKS. Jika Anda telah menetapkan nilai kustom untuk setelan apa pun, nilai tersebut mungkin akan ditimpa dengan nilai default Amazon EKS. Jika Anda mengubah nilai ininone, Amazon EKS tidak mengubah nilai pengaturan apa pun, tetapi pembaruan mungkin gagal. Jika pembaruan gagal, Anda menerima pesan galat untuk membantu menyelesaikan konflik.
   - Jika Anda tidak memperbarui pengaturan konfigurasi, hapus **--configuration-values '{*"replicaCount":3*}'** dari perintah. Jika Anda memperbarui pengaturan konfigurasi, ganti *“ReplicaCount” :3* dengan pengaturan yang ingin Anda atur. Dalam contoh ini, jumlah replika CoreDNS diatur ke3. Nilai yang Anda tentukan harus valid untuk skema konfigurasi. Jika Anda tidak tahu skema konfigurasi, jalankan**aws eks describe-addon-configuration --addon-name coredns --addon-version *v1.11.1-eksbuild.6***, ganti *v1.11.1-eksbuild.6* dengan nomor versi add-on yang ingin Anda lihat konfigurasinya. Skema dikembalikan dalam output. Jika Anda memiliki konfigurasi kustom yang ada, ingin menghapus semuanya, dan mengatur nilai untuk semua pengaturan kembali ke default Amazon EKS, hapus *“ReplicaCount” :3* dari perintah, sehingga Anda kosong. **{}** Untuk informasi selengkapnya tentang CoreDNS setelan, lihat Menyesuaikan Layanan DNS di dokumentasi. Kubernetes

```
aws eks update-addon --cluster-name my-cluster --addon-name coredns --addon-
version v1.11.1-eksbuild.6 \
    --resolve-conflicts PRESERVE --configuration-values '{"replicaCount":3}'
```

Mungkin perlu beberapa detik untuk pembaruan selesai.

4. Konfirmasikan bahwa versi add-on telah diperbarui. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns
```

Mungkin perlu beberapa detik untuk pembaruan selesai.

Contoh output adalah sebagai berikut.

```
{
    "addon": {
        "addonName": "coredns",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.11.1-eksbuild.6",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/coredns/
d2c34f06-1111-2222-1eb0-24f64ce37fa4",
        "createdAt": "2023-03-01T16:41:32.442000+00:00",
        "modifiedAt": "2023-03-01T18:16:54.332000+00:00",
        "tags": {},
        "configurationValues": "{\"replicaCount\":3}"
    }
}
```

Memperbarui add-on yang dikelola sendiri

Important

Sebaiknya tambahkan jenis add-on Amazon EKS ke klaster Anda alih-alih menggunakan jenis add-on yang dikelola sendiri. Jika Anda tidak terbiasa dengan perbedaan antara jenis, lihatthe section called “Add-on Amazon EKS”. Untuk informasi selengkapnya tentang menambahkan add-on Amazon EKS ke klaster Anda, lihatthe section called “Membuat add-on”. Jika Anda tidak dapat menggunakan add-on Amazon EKS, kami mendorong Anda untuk

mengirimkan masalah tentang mengapa Anda tidak dapat ke repositori peta jalan GitHub Containers.

1. Konfirmasikan bahwa Anda memiliki jenis add-on yang dikelola sendiri yang diinstal pada klaster Anda. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

   Jika pesan kesalahan dikembalikan, Anda memiliki jenis add-on yang dikelola sendiri yang diinstal pada klaster Anda. Selesaikan langkah-langkah yang tersisa dalam prosedur ini. Jika nomor versi dikembalikan, Anda memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda. Untuk memperbarui jenis add-on Amazon EKS, gunakan prosedur diMemperbarui add-on Amazon EKS, daripada menggunakan prosedur ini. Jika Anda tidak terbiasa dengan perbedaan antara jenis add-on, lihatAdd-on Amazon EKS.

2. Lihat versi gambar kontainer mana yang saat ini diinstal di cluster Anda.

```
kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
```

   Contoh output adalah sebagai berikut.

```
v1.8.7-eksbuild.2
```

3. Jika CoreDNS versi Anda saat ini `v1.5.0` atau lebih baru, tetapi lebih awal dari versi yang tercantum dalam tabel CoreDNSversi, lewati langkah ini. Jika versi Anda saat ini lebih awal dari`1.5.0`, maka Anda perlu memodifikasi `ConfigMap` for CoreDNS untuk menggunakan add-on penerusan, bukan add-on proxy.

   1. Buka configmap dengan perintah berikut.

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

   2. Ganti `proxy` di baris berikut dengan `forward`. Simpan file dan keluar dari editor.

```
proxy . /etc/resolv.conf
```

4. Jika Anda awalnya menerapkan cluster Anda pada Kubernetes `1.17` atau sebelumnya, Anda mungkin perlu menghapus baris yang dihentikan dari manifes AndaCoreDNS.

   > **Important**
   >
   > Anda harus menyelesaikan langkah ini sebelum memperbarui ke CoreDNS versi`1.7.0`, tetapi disarankan agar Anda menyelesaikan langkah ini bahkan jika Anda memperbarui ke versi sebelumnya.

   1. Periksa untuk melihat apakah CoreDNS manifes Anda memiliki garis.

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

      Jika tidak ada output yang dikembalikan, manifes Anda tidak memiliki baris dan Anda dapat melompat ke langkah berikutnya untuk memperbaruiCoreDNS. Jika output dikembalikan, maka Anda harus menghapus baris.

   2. Edit `ConfigMap` dengan perintah berikut, hapus baris dalam file yang memiliki kata `upstream` di dalamnya. Jangan ubah apa pun di dalam file. Setelah baris dihapus, simpan perubahannya.

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

5. Ambil versi CoreDNS gambar Anda saat ini:

   ```
   kubectl describe deployment coredns -n kube-system | grep Image
   ```

   Contoh output adalah sebagai berikut.

   ```
   602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.8.7-eksbuild.2
   ```

6. Jika Anda memperbarui ke CoreDNS `1.8.3` atau lebih baru, maka Anda perlu menambahkan `endpointslices` izin ke file `system:coredns` Kubernetes`clusterrole`.

   ```
   kubectl edit clusterrole system:coredns -n kube-system
   ```

Tambahkan baris berikut di bawah baris izin yang ada di `rules` bagian file.

```
[...]
- apiGroups:
  - discovery.k8s.io
  resources:
  - endpointslices
  verbs:
  - list
  - watch
[...]
```

7. Perbarui CoreDNS add-on dengan mengganti *602401143452* dan *region-code* dengan nilai dari output yang dikembalikan pada langkah sebelumnya. Ganti *v1.11.1-eksbuild.6*dengan CoreDNS versi yang tercantum dalam tabel versi terbaru untuk Kubernetes versi Anda.

```
kubectl set image deployment.apps/coredns -n kube-system
 coredns=602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.11.1-
eksbuild.6
```

Contoh output adalah sebagai berikut.

```
deployment.apps/coredns image updated
```

8. Periksa lagi versi gambar kontainer untuk mengonfirmasi bahwa itu diperbarui ke versi yang Anda tentukan pada langkah sebelumnya.

```
kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
```

Contoh output adalah sebagai berikut.

```
v1.11.1-eksbuild.6
```

Metrik CoreDNS

CoreDNSsebagai add-on EKS mengekspos metrik dari port CoreDNS on 9153 dalam format Prometheus dalam layanan. `kube-dns` Anda dapat menggunakan Prometheus, agen CloudWatch Amazon, atau sistem lain yang kompatibel untuk mengikis (mengumpulkan) metrik ini.

Untuk contoh konfigurasi scrape yang kompatibel dengan Prometheus dan agen, lihat konfigurasi CloudWatch agen untuk Prometheus CloudWatch di Panduan Pengguna Amazon. CloudWatch

Bekerja dengan add-on Kubernetes **kube-proxy**

> **Important**
>
> Sebaiknya tambahkan jenis add-on Amazon EKS ke klaster Anda alih-alih menggunakan jenis add-on yang dikelola sendiri. Jika Anda tidak terbiasa dengan perbedaan antara jenis, lihatthe section called “Add-on Amazon EKS”. Untuk informasi selengkapnya tentang menambahkan add-on Amazon EKS ke klaster Anda, lihatthe section called “Membuat add-on”. Jika Anda tidak dapat menggunakan add-on Amazon EKS, kami mendorong Anda untuk mengirimkan masalah tentang mengapa Anda tidak dapat ke repositori peta jalan GitHub Containers.

`kube-proxy`Add-on ini diterapkan pada setiap node Amazon EC2 di cluster Amazon EKS Anda. Ini memelihara aturan jaringan pada node Anda dan memungkinkan komunikasi jaringan ke AndaPods. Add-on tidak diterapkan ke node Fargate di cluster Anda. Untuk informasi lebih lanjut, lihat `kube-proxy`di Kubernetes dokumentasi.

Tabel berikut mencantumkan versi terbaru dari jenis add-on Amazon EKS untuk setiap Kubernetes versi.

Versi Kubernetes	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**	**1.24**	**1.23**
	v1.29.1-eksbuild	v1.28.6-eksbuild	v1.27.10-eksbuild.2	v1.26.13-eksbuild.2	v1.25.16-eksbuild.3	v1.24.17-eksbuild.8	v1.23.17-eksbuild.9

> **Important**
>
> Versi dokumentasi sebelumnya tidak benar. `kube-proxy`versiv`1.28.5`,`v1.27.9`, dan `v1.26.12` tidak tersedia.

Jika Anda mengelola sendiri add-on ini, versi dalam tabel mungkin tidak sama dengan versi yang dikelola sendiri yang tersedia.

Ada dua jenis gambar `kube-proxy` kontainer yang tersedia untuk setiap versi cluster Amazon EKS:

- Default — Jenis gambar ini didasarkan pada image Docker berbasis Debian yang dikelola oleh komunitas upstream. Kubernetes
- Minimal - Jenis gambar ini didasarkan pada gambar dasar minimal yang dikelola oleh Amazon EKS Distro, yang berisi paket minimal dan tidak memiliki cangkang. Untuk informasi selengkapnya, lihat Amazon EKS Distro.

Versi gambar **`kube-proxy`** kontainer terkelola mandiri terbaru yang tersedia untuk setiap versi cluster Amazon EKS

Jenis gambar	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**	**1.24**	**1.23**
kube-proxy (tipe default)	Hanya tipe minimal yang tersedia	Hanya tipe minimal yang tersedia	Hanya tipe minimal yang tersedia	Hanya tipe minimal yang tersedia	Hanya tipe minimal yang tersedia	v1.24.1(eksbuil(2	v1.23.16-eksbuild.2
kube-proxy (tipe minimal)	v1.29.1- m inimal- ek sbuild.1	v1.28.6- m inimal- ek sbuild.1	v1.27.1(minimal- e ksbuild	v1.26.1: minimal- e ksbuild	v1.25.1(minimal- e ksbuild	v1.24.1' minimal- e ksbuild	v1.23.17-minimal-e ksbuild.5

Important

- Jenis gambar default tidak tersedia untuk Kubernetes versi `1.25` dan yang lebih baru. Anda harus menggunakan jenis gambar minimal.
- Saat memperbarui jenis add-on Amazon EKS, Anda menentukan versi add-on Amazon EKS yang valid, yang mungkin bukan versi yang tercantum dalam tabel ini. Ini karena versi

add-on Amazon EKS tidak selalu cocok dengan versi gambar kontainer yang ditentukan saat memperbarui jenis pengaya ini yang dikelola sendiri. Saat memperbarui jenis add-on yang dikelola sendiri, Anda menentukan versi gambar kontainer yang valid yang tercantum dalam tabel ini.

Prasyarat

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS.

Pertimbangan

- `Kube-proxy`pada cluster Amazon EKS memiliki kompatibilitas dan kebijakan miring yang sama dengan. Kubernetes Pelajari cara Ambil kompatibilitas versi addon.
- `Kube-proxy`harus versi minor yang sama seperti `kubelet` pada node Amazon EC2 Anda.
- `Kube-proxy`tidak bisa lebih lambat dari versi minor dari bidang kontrol cluster Anda.
- Jika Anda baru saja memperbarui klaster ke versi Kubernetes minor baru, perbarui node Amazon EC2 Anda ke versi minor yang sama sebelum memperbarui `kube-proxy` ke versi minor yang sama dengan node Anda.

Untuk memperbarui add-on yang **`kube-proxy`** dikelola sendiri

1. Konfirmasikan bahwa Anda memiliki jenis add-on yang dikelola sendiri yang diinstal pada klaster Anda. Ganti *`my-cluster`* dengan nama klaster Anda.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name kube-proxy --query
    addon.addonVersion --output text
   ```

   Jika pesan kesalahan dikembalikan, Anda memiliki jenis add-on yang dikelola sendiri yang diinstal pada klaster Anda. Langkah-langkah yang tersisa dalam topik ini adalah untuk memperbarui jenis add-on yang dikelola sendiri. Jika nomor versi dikembalikan, Anda memiliki jenis add-on Amazon EKS yang diinstal pada cluster Anda. Untuk memperbaruinya, gunakan prosedur diMemperbarui add-on, daripada menggunakan prosedur dalam topik ini. Jika Anda tidak terbiasa dengan perbedaan antara jenis add-on, lihatAdd-on Amazon EKS.
2. Lihat versi gambar kontainer mana yang saat ini diinstal di cluster Anda.

```
kubectl describe daemonset kube-proxy -n kube-system | grep Image
```

Contoh output adalah sebagai berikut.

```
Image:    602401143452.dkr.ecr.region-code.amazonaws.com/eks/kube-proxy:v1.25.6-minimal-eksbuild.2
```

Dalam contoh output, *v1.25.6-minimal-eksbuild.2* adalah versi yang diinstal pada cluster.

3. Perbarui kube-proxy add-on dengan mengganti *602401143452* dan *region-code*dengan nilai dari output Anda. pada langkah sebelumnya Ganti *v1.26.2-minimal-eksbuild.2*dengan versi yang tercantum dalam kube-proxy versi gambar kube-proxy kontainer terkelola mandiri terbaru yang tersedia untuk setiap tabel versi cluster Amazon EKS. Anda dapat menentukan nomor versi untuk jenis gambar default atau minimal.

```
kubectl set image daemonset.apps/kube-proxy -n kube-system kube-proxy=602401143452.dkr.ecr.region-code.amazonaws.com/eks/kube-proxy:v1.26.2-minimal-eksbuild.2
```

Contoh output adalah sebagai berikut.

```
daemonset.apps/kube-proxy image updated
```

4. Konfirmasikan bahwa versi baru sekarang diinstal pada cluster Anda.

```
kubectl describe daemonset kube-proxy -n kube-system | grep Image | cut -d ":" -f 3
```

Contoh output adalah sebagai berikut.

```
v1.26.2-minimal-eksbuild.2
```

5. Jika Anda menggunakan x86 dan Arm node di cluster yang sama dan cluster Anda telah diterapkan sebelum 17 Agustus 2020. Lalu, edit manifes kube-proxy Anda guna menyertakan selektor simpul untuk beberapa arsitektur perangkat keras dengan perintah berikut. Ini adalah operasi satu kali. Setelah menambahkan pemilih ke manifes, Anda tidak perlu menambahkannya setiap kali memperbarui add-on. Jika klaster Anda di-deploy pada atau setelah tanggal 17 Agustus 2020, maka kube-proxy sudah memiliki kemampuan multi-arsitektur.

```
kubectl edit -n kube-system daemonset/kube-proxy
```

Tambahkan selektor simpul berikut ke file di dalam editor dan kemudian simpan file. Untuk contoh tempat menyertakan teks ini di editor, lihat file manifes CNI. GitHub Ini memungkinkan Kubernetes untuk menarik gambar perangkat keras yang benar berdasarkan arsitektur perangkat keras node.

```
- key: "kubernetes.io/arch"
  operator: In
  values:
  - amd64
  - arm64
```

6. Jika cluster Anda awalnya dibuat dengan Kubernetes versi `1.14` atau yang lebih baru, maka Anda dapat melewati langkah ini karena `kube-proxy` sudah menyertakan ini`Affinity Rule`. Jika Anda awalnya membuat klaster Amazon EKS dengan Kubernetes versi `1.13` atau versi sebelumnya dan bermaksud menggunakan node Fargate di cluster Anda, maka edit `kube-proxy` manifes Anda untuk menyertakan `NodeAffinity` aturan untuk `kube-proxy` Pods mencegah penjadwalan pada node Fargate. Ini adalah pengeditan satu kali. Setelah Anda menambahkan `Affinity Rule` ke manifes Anda, Anda tidak perlu menambahkannya setiap kali Anda memperbarui add-on. Edit Anda `kube-proxy``DaemonSet`.

   ```
   kubectl edit -n kube-system daemonset/kube-proxy
   ```

   Tambahkan `Affinity Rule` berikut ke bagian `DaemonSet` `spec` dari file di dalam editor dan kemudian simpan file. Untuk contoh tempat menyertakan teks ini di editor, lihat file manifes CNI. GitHub

   ```
   - key: eks.amazonaws.com/compute-type
     operator: NotIn
     values:
     - fargate
   ```

Akses Amazon Elastic Kubernetes Service menggunakan endpoint antarmuka () AWS PrivateLink

Anda dapat menggunakannya AWS PrivateLink untuk membuat koneksi pribadi antara VPC Anda dan Amazon Elastic Kubernetes Service. Anda dapat mengakses Amazon EKS seolah-olah berada di VPC Anda, tanpa menggunakan gateway internet, perangkat NAT, koneksi VPN, atau koneksi. AWS Direct Connect Instans di VPC Anda tidak memerlukan alamat IP publik untuk mengakses Amazon EKS.

Anda membuat koneksi pribadi ini dengan membuat titik akhir antarmuka yang didukung olehAWS PrivateLink. Kami membuat antarmuka jaringan endpoint di setiap subnet yang Anda aktifkan untuk titik akhir antarmuka. Ini adalah antarmuka jaringan yang dikelola pemohon yang berfungsi sebagai titik masuk untuk lalu lintas yang ditujukan untuk Amazon EKS.

Untuk informasi selengkapnya, lihat Mengakses Layanan AWS melalui AWS PrivateLink di Panduan AWS PrivateLink.

Pertimbangan untuk Amazon EKS

- Sebelum Anda menyiapkan titik akhir antarmuka untuk Amazon EKS, tinjau Pertimbangan dalam Panduan. AWS PrivateLink
- Amazon EKS mendukung panggilan ke semua tindakan API-nya melalui titik akhir antarmuka, tetapi tidak ke Kubernetes API. Server Kubernetes API sudah mendukung titik akhir pribadi. Titik akhir pribadi server Kubernetes API membuat titik akhir pribadi untuk server Kubernetes API yang Anda gunakan untuk berkomunikasi dengan klaster Anda (menggunakan alat Kubernetes manajemen seperti`kubectl`). Anda dapat mengaktifkan akses pribadi ke server Kubernetes API sehingga semua komunikasi antara node dan server API tetap berada dalam VPC Anda. AWS PrivateLinkuntuk Amazon EKS API membantu Anda memanggil Amazon EKS API dari VPC Anda tanpa mengekspos lalu lintas ke internet publik.
- Anda tidak dapat mengonfigurasi Amazon EKS agar hanya diakses melalui titik akhir antarmuka.
- Harga standar untuk AWS PrivateLink berlaku untuk titik akhir antarmuka untuk Amazon EKS. Anda ditagih untuk setiap jam bahwa titik akhir antarmuka disediakan di setiap Availability Zone dan untuk data yang diproses melalui titik akhir antarmuka. Untuk informasi selengkapnya, lihat harga AWS PrivateLink.
- Kebijakan titik akhir VPC tidak didukung untuk Amazon EKS. Secara default, akses penuh ke Amazon EKS diizinkan melalui titik akhir antarmuka. Atau, Anda dapat mengaitkan grup keamanan

dengan antarmuka jaringan titik akhir untuk mengontrol lalu lintas ke Amazon EKS melalui titik akhir antarmuka.

- Anda dapat menggunakan log aliran VPC untuk menangkap informasi tentang lalu lintas IP yang pergi ke dan dari antarmuka jaringan, termasuk titik akhir antarmuka. Anda dapat mempublikasikan data log aliran ke Amazon CloudWatch atau Amazon S3. Untuk informasi selengkapnya, lihat Mencatat lalu lintas IP menggunakan Log Aliran VPC di Panduan Pengguna Amazon VPC.
- Anda dapat mengakses Amazon EKS API dari pusat data lokal dengan menghubungkannya ke VPC yang memiliki titik akhir antarmuka. Anda dapat menggunakan AWS Direct Connect atau AWS Site-to-Site VPN menghubungkan situs lokal Anda ke VPC.
- Anda dapat menghubungkan VPC lain ke VPC dengan titik akhir antarmuka menggunakan peering AWS Transit Gateway atau VPC. Peering VPC adalah koneksi jaringan di antara dua VPC. Anda dapat membuat koneksi peering VPC antara VPC Anda, atau dengan VPC di akun lain. VPC bisa berbedaWilayah AWS. Lalu lintas antara VPC peered tetap berada di jaringan. AWS Lalu lintas tidak melintasi internet publik. Transit Gateway adalah hub transit jaringan yang dapat Anda gunakan untuk menghubungkan VPC. Lalu lintas antara VPC dan Transit Gateway tetap berada di jaringan pribadi AWS global. Lalu lintas tidak terpapar ke internet publik.
- Titik akhir antarmuka VPC untuk Amazon EKS hanya dapat diakses melalui. `IPv4` `IPv6`tidak didukung.
- AWS PrivateLinkdukungan tidak tersedia di Asia Pasifik (Hyderabad), Asia Pasifik (Jakarta), Asia Pasifik (Melbourne), Asia Pasifik (Osaka), Kanada Barat (Calgary), Eropa (Spanyol), Eropa (Zurich), Israel (Tel Aviv), atau Timur Tengah (UEA). Wilayah AWS

Buat titik akhir antarmuka untuk Amazon EKS

Anda dapat membuat titik akhir antarmuka untuk Amazon EKS menggunakan konsol VPC Amazon atau AWS Command Line Interface (). AWS CLI Untuk informasi selengkapnya, lihat Membuat titik akhir VPC di Panduan Pengguna AWS PrivateLink.

Buat titik akhir antarmuka untuk Amazon EKS menggunakan nama layanan berikut:

```
com.amazonaws.region-code.eks
```

Fitur DNS pribadi diaktifkan secara default saat membuat titik akhir antarmuka untuk Amazon EKS dan lainnya. Layanan AWS Namun, Anda harus memastikan bahwa atribut VPC berikut diatur `ketrue`: `enableDnsHostnames` dan. `enableDnsSupport` Untuk informasi selengkapnya, lihat

Melihat dan memperbarui atribut DNS untuk VPC Anda di Panduan Pengguna Amazon VPC. Dengan fitur DNS pribadi diaktifkan untuk titik akhir antarmuka:

- Anda dapat membuat permintaan API apa pun ke Amazon EKS menggunakan nama DNS Regional default. Misalnya, `eks.region.amazonaws.com`. Untuk daftar API, lihat Tindakan di Referensi API Amazon EKS.
- Anda tidak perlu membuat perubahan apa pun pada aplikasi Anda yang memanggil API EKS.
- Setiap panggilan yang dilakukan ke titik akhir layanan default Amazon EKS secara otomatis dirutekan melalui titik akhir antarmuka melalui jaringan pribadi. AWS

Beban kerja

Beban kerja Anda diterapkan dalam kontainer, yang digunakan di. Pods Kubernetes A Pod termasuk satu atau lebih kontainer. Biasanya, satu atau lebih Pods yang menyediakan layanan yang sama digunakan dalam suatu Kubernetes layanan. Setelah Anda menerapkan beberapa Pods yang menyediakan layanan yang sama, Anda dapat:

- Melihat informasi tentang beban kerja yang berjalan pada setiap klaster menggunakan AWS Management Console.
- Skala vertikal Pods ke atas atau ke bawah dengan. Kubernetes Vertical Pod Autoscaler
- Skala horizontal jumlah yang Pods dibutuhkan untuk memenuhi permintaan naik atau turun dengan. Kubernetes Penskala Otomatis Pod Horizontal
- Buat penyeimbang beban jaringan eksternal (untuk diakses internetPods) atau internal (untuk pribadiPods) untuk menyeimbangkan lalu lintas jaringan. Pods Keseimbangan beban merutekan lalu lintas di Lapisan 4 dari model OSI.
- Buat Penyeimbangan beban aplikasi pada Amazon EKS untuk menyeimbangkan lalu lintas aplikasi di seluruhPods. Application Load Balancer merutekan lalu lintas di Lapisan 7 dari model OSI.
- Jika Anda baru mengenalKubernetes, topik ini membantu AndaDeploy aplikasi sampel.
- Anda dapat membatasi alamat IP yang dapat ditugaskan ke layanan dengan `externalIPs`.

Deploy aplikasi sampel

Dalam topik ini, Anda menerapkan aplikasi sampel ke klaster Anda.

Prasyarat

- KubernetesCluster yang ada dengan setidaknya satu node. Jika Anda tidak memiliki kluster Amazon EKS yang ada, Anda dapat menerapkannya menggunakan salah satu Memulai dengan Amazon EKS panduan. Jika Anda menerapkan Windows aplikasi, maka Anda harus mengaktifkan Windowsdukungan untuk klaster Anda dan setidaknya satu node Amazon Windows EC2.
- `Kubectl`diinstal pada komputer Anda. Untuk informasi selengkapnya, lihat Menginstal atau memperbarui `kubectl`.
- `Kubectl`dikonfigurasi untuk berkomunikasi dengan cluster Anda. Untuk informasi selengkapnya, lihat Membuat atau memperbarui `kubeconfig` file untuk klaster Amazon EKS.

- Jika Anda berencana untuk menyebarkan beban kerja sampel Anda ke Fargate, maka Anda harus memiliki profil Fargate yang ada yang menyertakan namespace yang sama yang dibuat dalam tutorial ini, yaitu, kecuali Anda mengubah nama. `eks-sample-app` Jika Anda menggunakan salah satu panduan memulai untuk membuat cluster Anda, maka Anda harus membuat profil baru, atau menambahkan namespace ke profil yang ada, karena profil yang dibuat dalam panduan memulai tidak menentukan namespace yang digunakan dalam tutorial ini. VPC Anda juga harus memiliki setidaknya satu subnet pribadi.

Untuk menggunakan aplikasi sampel

Meskipun banyak variabel yang dapat diubah dalam langkah-langkah berikut, kami sarankan hanya mengubah nilai variabel jika ditentukan. Setelah Anda memiliki pemahaman yang lebih baik tentang KubernetesPods, penerapan, dan layanan, Anda dapat bereksperimen dengan mengubah nilai lainnya.

1. Buat namespace. Sebuah namespace memungkinkan Anda untuk mengelompokkan sumber daya di. Kubernetes Untuk informasi selengkapnya, lihat Ruang nama dalam dokumentasi. Kubernetes Jika Anda berencana untuk menerapkan aplikasi sampel AndaAWS Fargate, pastikan bahwa nilai untuk `namespace` aplikasi Anda AWS Fargate profil adalah`eks-sample-app`.

   ```
   kubectl create namespace eks-sample-app
   ```

2. Buat Kubernetes penyebaran. Penerapan sampel ini menarik gambar kontainer dari repositori publik dan menyebarkan tiga replika (individualPods) ke cluster Anda. Untuk mempelajari lebih lanjut, lihat Penerapan dalam dokumentasi. Kubernetes Anda dapat menyebarkan aplikasi ke Linux atau Windows node. Jika Anda menerapkan ke Fargate, maka Anda hanya dapat menerapkan Linux aplikasi.

   a. Simpan konten berikut ini ke file bernama `eks-sample-deployment.yaml`. Kontainer dalam aplikasi sampel tidak menggunakan penyimpanan jaringan, tetapi Anda mungkin memiliki aplikasi yang perlu. Untuk informasi selengkapnya, lihat Penyimpanan.

      Linux

      amd64Atau di `arm64` `values` bawah `kubernetes.io/arch` kunci berarti bahwa aplikasi dapat digunakan untuk salah satu arsitektur perangkat keras (jika Anda memiliki keduanya di cluster Anda). Ini dimungkinkan karena gambar ini adalah gambar multi-

arsitektur, tetapi tidak semuanya. Anda dapat menentukan arsitektur perangkat keras tempat gambar didukung dengan melihat detail gambar di repositori tempat Anda menariknya. Saat menerapkan gambar yang tidak mendukung jenis arsitektur perangkat keras, atau yang Anda tidak ingin gambar disebarkan, hapus jenis itu dari manifes. Untuk informasi selengkapnya, lihat Label Terkenal, Anotasi, dan Taints dalam dokumentasi. Kubernetes

`kubernetes.io/os: linux`Artinya `nodeSelector` jika Anda memiliki Linux dan Windows node (misalnya) di cluster Anda, gambar hanya akan digunakan ke Linux node. Untuk informasi selengkapnya, lihat Label Terkenal, Anotasi, dan Taints dalam dokumentasi. Kubernetes

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eks-sample-linux-deployment
  namespace: eks-sample-app
  labels:
    app: eks-sample-linux-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: eks-sample-linux-app
  template:
    metadata:
      labels:
        app: eks-sample-linux-app
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: kubernetes.io/arch
                operator: In
                values:
                - amd64
                - arm64
      containers:
      - name: nginx
        image: public.ecr.aws/nginx/nginx:1.23
```

```
        ports:
        - name: http
          containerPort: 80
        imagePullPolicy: IfNotPresent
      nodeSelector:
        kubernetes.io/os: linux
```

Windows

`kubernetes.io/os: windows`Artinya jika Anda memiliki Windows dan Linux node (misalnya) di cluster Anda, gambar hanya akan digunakan ke Windows node. Untuk informasi selengkapnya, lihat Label Terkenal, Anotasi, dan Taints dalam dokumentasi. Kubernetes

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eks-sample-windows-deployment
  namespace: eks-sample-app
  labels:
    app: eks-sample-windows-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: eks-sample-windows-app
  template:
    metadata:
      labels:
        app: eks-sample-windows-app
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: beta.kubernetes.io/arch
                operator: In
                values:
                - amd64
      containers:
      - name: windows-server-iis
        image: mcr.microsoft.com/windows/servercore:ltsc2019
```

```
        ports:
        - name: http
          containerPort: 80
        imagePullPolicy: IfNotPresent
        command:
        - powershell.exe
        - -command
        - "Add-WindowsFeature Web-Server; Invoke-WebRequest -UseBasicParsing
 -Uri 'https://dotnetbinaries.blob.core.windows.net/servicemonitor/2.0.1.6/
ServiceMonitor.exe' -OutFile 'C:\\ServiceMonitor.exe'; echo
 '<html><body><br/><br/><marquee><H1>Hello EKS!!!<H1><marquee></body><html>'
 > C:\\inetpub\\wwwroot\\default.html; C:\\ServiceMonitor.exe 'w3svc'; "
      nodeSelector:
        kubernetes.io/os: windows
```

b. Terapkan manifes penerapan ke klaster Anda.

```
kubectl apply -f eks-sample-deployment.yaml
```

3. Buat sebuah layanan. Sebuah layanan memungkinkan Anda untuk mengakses semua replika melalui satu alamat IP atau nama. Untuk informasi selengkapnya, lihat Layanan dalam Kubernetes dokumentasi. Meskipun tidak diterapkan dalam aplikasi sampel, jika Anda memiliki aplikasi yang perlu berinteraksi dengan AWS layanan lain, kami sarankan Anda membuat akun Kubernetes layanan untuk AndaPods, dan mengaitkannya ke akun AWS IAM. Dengan menentukan akun layanan, Anda hanya Pods memiliki izin minimum yang Anda tentukan agar mereka dapat berinteraksi dengan layanan lain. Untuk informasi selengkapnya, lihat IAM role untuk akun layanan.

   a. Simpan konten berikut ke file bernama`eks-sample-service.yaml`. Kubernetesmemberikan layanan alamat IP sendiri yang hanya dapat diakses dari dalam cluster. Untuk mengakses layanan dari luar klaster Anda, gunakan aplikasi AWS Load Balancer Controlleruntuk memuat keseimbangan atau lalu lintas jaringan ke layanan.

      Linux

```
apiVersion: v1
kind: Service
metadata:
  name: eks-sample-linux-service
  namespace: eks-sample-app
  labels:
```

```
    app: eks-sample-linux-app
spec:
  selector:
    app: eks-sample-linux-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

Windows

```
apiVersion: v1
kind: Service
metadata:
  name: eks-sample-windows-service
  namespace: eks-sample-app
  labels:
    app: eks-sample-windows-app
spec:
  selector:
    app: eks-sample-windows-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

b. Terapkan manifes layanan ke cluster Anda.

```
kubectl apply -f eks-sample-service.yaml
```

4. Lihat semua sumber daya yang ada di namespace `eks-sample-app`.

```
kubectl get all -n eks-sample-app
```

Contoh output adalah sebagai berikut.

Jika Anda menerapkan Windows sumber daya, maka semua instance *`linux`* dalam output berikut adalah. `windows` *`Contoh nilai`* lainnya mungkin berbeda dari output Anda.

```
NAME                                                READY   STATUS    RESTARTS   AGE
pod/eks-sample-linux-deployment-65b7669776-m6qxz    1/1     Running   0          27m
pod/eks-sample-linux-deployment-65b7669776-mmxvd    1/1     Running   0          27m
```

```
pod/eks-sample-linux-deployment-65b7669776-qzn22   1/1     Running   0          27m

NAME                                TYPE         CLUSTER-IP      EXTERNAL-IP
 PORT(S)   AGE
service/eks-sample-linux-service    ClusterIP    10.100.74.8     <none>        80/
TCP    32m

NAME                                          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/eks-sample-linux-deployment 3/3     3            3           27m

NAME                                                      DESIRED   CURRENT   READY
   AGE
replicaset.apps/eks-sample-linux-deployment-776d8f8fd8    3         3         3
   27m
```

Dalam output, Anda melihat layanan dan penerapan yang ditentukan dalam manifes sampel yang diterapkan pada langkah sebelumnya. Anda juga melihat tigaPods. Ini karena `3` `replicas` ditentukan dalam manifes sampel. Untuk informasi selengkapnyaPods, lihat Pod dalam Kubernetes dokumentasi. Kubernetessecara otomatis membuat `replicaset` sumber daya, meskipun tidak ditentukan dalam manifes sampel. Untuk informasi selengkapnya`ReplicaSets`, lihat ReplicaSetdi Kubernetes dokumentasi.

> **Note**
>
> Kubernetesmempertahankan jumlah replika yang ditentukan dalam manifes. Jika ini adalah penerapan produksi dan Anda Kubernetes ingin menskalakan jumlah replika secara horizontal atau menskalakan sumber daya komputasi secara vertikal untukPods, gunakan Penskala Otomatis Pod Horizontal dan untuk melakukannya. Vertical Pod Autoscaler

5. Lihat detail layanan yang di-deploy. Jika Anda menerapkan Windows layanan, ganti *`linux`* dengan**`windows`**.

```
kubectl -n eks-sample-app describe service eks-sample-linux-service
```

Contoh output adalah sebagai berikut.

Jika Anda menerapkan Windows sumber daya, maka semua instance *`linux`* dalam output berikut adalah. `windows` *`Contoh nilai`* lainnya mungkin berbeda dari output Anda.

```
Name:              eks-sample-linux-service
Namespace:         eks-sample-app
Labels:            app=eks-sample-linux-app
Annotations:       <none>
Selector:          app=eks-sample-linux-app
Type:              ClusterIP
IP Families:       <none>
IP:                10.100.74.8
IPs:               10.100.74.8
Port:              <unset>  80/TCP
TargetPort:        80/TCP
Endpoints:         192.168.24.212:80,192.168.50.185:80,192.168.63.93:80
Session Affinity:  None
Events:            <none>
```

Pada output sebelumnya, nilai untuk `IP:` adalah alamat IP unik yang dapat dicapai dari node mana pun atau di Pod dalam cluster, tetapi tidak dapat dicapai dari luar cluster. Nilai untuk `Endpoints` adalah alamat IP yang ditetapkan dari dalam VPC Anda ke Pods yang merupakan bagian dari layanan.

6. Lihat detail salah satu yang Pods tercantum dalam output saat Anda melihat namespace di langkah sebelumnya. Jika Anda menerapkan Windows aplikasi, ganti *`linux`* dengan **`windows`** dan ganti *`776d8f8fd8-78w66`* dengan nilai yang dikembalikan untuk salah satu aplikasi AndaPods.

```
kubectl -n eks-sample-app describe pod eks-sample-linux-deployment-65b7669776-m6qxz
```

Output dipersingkat

Jika Anda menerapkan Windows sumber daya, maka semua instance *`linux`* dalam output berikut adalah. `windows` Yang lain *`example values`* mungkin berbeda dari output Anda.

```
Name:         eks-sample-linux-deployment-65b7669776-m6qxz
Namespace:    eks-sample-app
Priority:     0
Node:         ip-192-168-45-132.us-west-2.compute.internal/192.168.45.132
[...]
IP:           192.168.63.93
IPs:
  IP:           192.168.63.93
```

```
Controlled By:  ReplicaSet/eks-sample-linux-deployment-65b7669776
[...]
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
[...]
Events:
  Type    Reason     Age    From
 Message
  ----    ------     ----   ----
 -------
  Normal  Scheduled  3m20s  default-scheduler
 Successfully assigned eks-sample-app/eks-sample-linux-deployment-65b7669776-m6qxz
 to ip-192-168-45-132.us-west-2.compute.internal
[...]
```

Pada output sebelumnya, nilai untuk `IP:` adalah IP unik yang ditetapkan ke Pod dari blok CIDR yang ditetapkan ke subnet tempat node berada. Jika Anda lebih suka menetapkan alamat Pods IP dari blok CIDR yang berbeda, Anda dapat mengubah perilaku default. Untuk informasi selengkapnya, lihat Jaringan khusus untuk pod. Anda juga dapat melihat bahwa Kubernetes penjadwal menjadwalkan Pod pada Node dengan alamat *192.168.45.132*IP.

> **Tip**
>
> Daripada menggunakan baris perintah, Anda dapat melihat banyak detail tentangPods, layanan, penerapan, dan Kubernetes sumber daya lainnya di. AWS Management Console Untuk informasi selengkapnya, lihat Lihat Kubernetes sumber daya.

7. Jalankan shell pada Pod yang Anda jelaskan pada langkah sebelumnya, ganti *65b7669776-m6qxz* dengan ID salah satu dari AndaPods.

   Linux

   ```
   kubectl exec -it eks-sample-linux-deployment-65b7669776-m6qxz -n eks-sample-app
    -- /bin/bash
   ```

Windows

```
kubectl exec -it eks-sample-windows-deployment-65b7669776-m6qxz -n eks-sample-app -- powershell.exe
```

8. Dari Pod shell, lihat output dari server web yang diinstal dengan penerapan Anda di langkah sebelumnya. Anda hanya perlu menentukan nama layanan. Ini diselesaikan ke alamat IP layanan olehCoreDNS, yang digunakan dengan cluster Amazon EKS, secara default.

   Linux

   ```
   curl eks-sample-linux-service
   ```

   Contoh output adalah sebagai berikut.

   ```
   <!DOCTYPE html>
   <html>
   <head>
   <title>Welcome to nginx!</title>
   [...]
   ```

   Windows

   ```
   Invoke-WebRequest -uri eks-sample-windows-service/default.html -UseBasicParsing
   ```

   Contoh output adalah sebagai berikut.

   ```
   StatusCode        : 200
   StatusDescription : OK
   Content           : < h t m l > < b o d y > < b r / > < b r / > < m a r q u e e
    > < H 1 > H e l l o
                       E K S ! ! ! < H 1 > < m a r q u e e > < / b o d y > < h t
    m l >
   ```

9. Dari Pod shell, lihat server DNS untuk file. Pod

   Linux

   ```
   cat /etc/resolv.conf
   ```

Contoh output adalah sebagai berikut.

```
nameserver 10.100.0.10
search eks-sample-app.svc.cluster.local svc.cluster.local cluster.local us-
west-2.compute.internal
options ndots:5
```

Pada output sebelumnya, `10.100.0.10` secara otomatis ditetapkan sebagai `nameserver` untuk semua Pods dikerahkan ke cluster.

Windows

```
Get-NetIPConfiguration
```

Output dipersingkat

```
InterfaceAlias       : vEthernet
[...]
IPv4Address          : 192.168.63.14
[...]
DNSServer            : 10.100.0.10
```

Pada output sebelumnya, secara otomatis `10.100.0.10` ditetapkan sebagai server DNS untuk semua Pods dikerahkan ke cluster.

10. Putuskan sambungan dari Pod dengan mengetik`exit`.
11. Setelah selesai dengan aplikasi sampel, Anda dapat menghapus contoh namespace, layanan, dan deployment dengan perintah berikut.

```
kubectl delete namespace eks-sample-app
```

Langkah Berikutnya

Setelah Anda menerapkan aplikasi sampel, Anda mungkin ingin mencoba beberapa latihan berikut:

- the section called “Penyeimbangan beban aplikasi”
- the section called “Menyeimbangkan beban jaringan”

Vertical Pod Autoscaler

KubernetesVertical Pod Autoscaler secara otomatis menyesuaikan reservasi CPU dan memori Pods untuk membantu "ukuran yang tepat" aplikasi Anda. Penyesuaian ini dapat meningkatkan pemanfaatan sumber daya cluster dan membebaskan CPU dan memori untuk lainnyaPods. Topik ini membantu Anda untuk men-deploy Vertikal Pod Autoscaler untuk klaster Anda dan memverifikasi bahwa itu bekerja.

Prasyarat

- Anda memiliki klaster Amazon EKS yang ada. Jika Anda tidak melakukannya, tinjau Memulai dengan Amazon EKS.
- Anda telah menginstal Server Kubernetes Metrik. Untuk informasi selengkapnya, lihat Menginstal Server Kubernetes Metrik.
- Anda menggunakan klien `kubectl` yang dikonfigurasi untuk berkomunikasi dengan klaster Amazon EKS.
- `1.1.1`OpenSSL atau yang lebih baru diinstal pada perangkat Anda.

Men-deploy Vertical Pod Autoscaler

Pada bagian ini, Anda men-deploy Vertical Pod Autoscaler untuk klaster Anda.

Untuk men-deploy Vertical Pod Autoscaler

1. Buka jendela terminal dan arahkan ke direktori tempat Anda ingin mengunduh kode sumber Vertical Pod Autoscaler.
2. Kloning repositori GitHubkubernetes/autoscaler.

```
git clone https://github.com/kubernetes/autoscaler.git
```

3. Mengubah ke direktori `vertical-pod-autoscaler`.

```
cd autoscaler/vertical-pod-autoscaler/
```

4. (Opsional) Jika Anda telah men-deploy versi lain Vertical Pod Autoscaler, hapus dengan perintah berikut.

```
./hack/vpa-down.sh
```

5. Jika node Anda tidak memiliki akses internet ke registri `registry.k8s.io` kontainer, maka Anda perlu menarik gambar berikut dan mendorongnya ke repositori pribadi Anda sendiri. Untuk informasi selengkapnya tentang cara menarik gambar dan mendorongnya ke repositori pribadi Anda sendiri, lihat. Salin gambar kontainer dari satu repositori ke repositori lain

```
registry.k8s.io/autoscaling/vpa-admission-controller:0.10.0
registry.k8s.io/autoscaling/vpa-recommender:0.10.0
registry.k8s.io/autoscaling/vpa-updater:0.10.0
```

   Jika Anda mendorong gambar ke repositori Amazon ECR pribadi, ganti manifes dengan `registry.k8s.io` registri Anda. Ganti *111122223333* dengan ID akun Anda. Ganti *region-code* dengan tempat Wilayah AWS cluster Anda berada. Perintah berikut mengasumsikan bahwa Anda menamai repositori Anda sama dengan nama repositori dalam manifes. Jika Anda menamai repositori Anda sesuatu yang berbeda, maka Anda harus mengubahnya juga.

```
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/admission-controller-deployment.yaml
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/recommender-deployment.yaml
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/updater-deployment.yaml
```

6. Men-deploy Vertical Pod Autoscaler untuk klaster Anda dengan perintah berikut.

```
./hack/vpa-up.sh
```

7. Verifikasi bahwa Vertical Pod Autoscaler Pods telah berhasil dibuat.

```
kubectl get pods -n kube-system
```

   Contoh output adalah sebagai berikut.

```
NAME                                        READY   STATUS    RESTARTS   AGE
[...]
metrics-server-8459fc497-kfj8w              1/1     Running   0          83m
vpa-admission-controller-68c748777d-ppspd   1/1     Running   0          7s
vpa-recommender-6fc8c67d85-gljpl            1/1     Running   0          8s
vpa-updater-786b96955c-bgp9d                1/1     Running   0          8s
```

Uji pemasangan Vertical Pod Autoscaler

Dalam bagian ini, Anda men-deploy contoh aplikasi untuk memverifikasi bahwa Vertical Pod Autoscaler bekerja.

Untuk menguji instalasi Vertical Pod Autoscaler

1. Deploy `hamster.yaml` contoh Vertikal Pod Autoscaler dengan perintah berikut.

```
kubectl apply -f examples/hamster.yaml
```

2. Dapatkan Pods dari `hamster` contoh aplikasi.

```
kubectl get pods -l app=hamster
```

   Contoh output adalah sebagai berikut.

```
hamster-c7d89d6db-rglf5   1/1     Running   0          48s
hamster-c7d89d6db-znvz5   1/1     Running   0          48s
```

3. Jelaskan salah satu Pods untuk melihat `cpu` dan `memory` reservasi. Ganti *c7d89d6db-rglf5* dengan salah satu ID yang dikembalikan dalam output Anda dari langkah sebelumnya.

```
kubectl describe pod hamster-c7d89d6db-rglf5
```

   Contoh output adalah sebagai berikut.

```
[...]
Containers:
  hamster:
    Container ID:  docker://
e76c2413fc720ac395c33b64588c82094fc8e5d590e373d5f818f3978f577e24
    Image:         registry.k8s.io/ubuntu-slim:0.1
    Image ID:      docker-pullable://registry.k8s.io/ubuntu-
slim@sha256:b6f8c3885f5880a4f1a7cf717c07242eb4858fdd5a84b5ffe35b1cf680ea17b1
    Port:          <none>
    Host Port:     <none>
    Command:
      /bin/sh
    Args:
      -c
```

```
      while true; do timeout 0.5s yes >/dev/null; sleep 0.5s; done
    State:          Running
      Started:      Fri, 27 Sep 2019 10:35:16 -0700
    Ready:          True
    Restart Count:  0
    Requests:
      cpu:        100m
      memory:     50Mi
[...]
```

Anda dapat melihat bahwa Pod cadangan asli 100 millicpu CPU dan 50 mebibytes memori. Untuk contoh aplikasi ini, 100 millicpu kurang dari Pod kebutuhan untuk menjalankan, sehingga CPU dibatasi. Ini juga menyimpan memori jauh lebih sedikit dari yang dibutuhkan. `vpa-recommender`Penerapan Vertical Pod Autoscaler menganalisis `hamster` Pods untuk melihat apakah persyaratan CPU dan memori sesuai. Jika penyesuaian diperlukan, `vpa-updater` peluncuran kembali Pods dengan nilai yang diperbarui.

4. Tunggu `vpa-updater` sampai meluncurkan yang baru `hamster`Pod. Ini harus memakan waktu satu atau dua menit. Anda dapat memantau Pods dengan perintah berikut.

> **Note**
>
> Jika Anda tidak yakin bahwa yang baru Pod telah diluncurkan, bandingkan Pod nama dengan daftar Anda sebelumnya. Saat Pod peluncuran baru, Anda akan melihat Pod nama baru.

```
kubectl get --watch Pods -l app=hamster
```

5. Ketika baru `hamster` Pod dimulai, jelaskan dan lihat reservasi CPU dan memori yang diperbarui.

```
kubectl describe pod hamster-c7d89d6db-jxgfv
```

Contoh output adalah sebagai berikut.

```
[...]
Containers:
  hamster:
```

```
    Container ID:
 docker://2c3e7b6fb7ce0d8c86444334df654af6fb3fc88aad4c5d710eac3b1e7c58f7db
    Image:         registry.k8s.io/ubuntu-slim:0.1
    Image ID:      docker-pullable://registry.k8s.io/ubuntu-
slim@sha256:b6f8c3885f5880a4f1a7cf717c07242eb4858fdd5a84b5ffe35b1cf680ea17b1
    Port:          <none>
    Host Port:     <none>
    Command:
      /bin/sh
    Args:
      -c
      while true; do timeout 0.5s yes >/dev/null; sleep 0.5s; done
    State:          Running
      Started:      Fri, 27 Sep 2019 10:37:08 -0700
    Ready:          True
    Restart Count:  0
    Requests:
      cpu:        587m
      memory:     262144k
[...]
```

Pada output sebelumnya, Anda dapat melihat bahwa `cpu` reservasi meningkat menjadi 587 millicpu, yang lebih dari lima kali nilai aslinya. `memory`Meningkat menjadi 262.144 Kilobyte, yaitu sekitar 250 mebibytes, atau lima kali nilai aslinya. Ini Pod kurang sumber daya, dan Vertical Pod Autoscaler mengoreksi perkiraan dengan nilai yang jauh lebih tepat.

6. Menjelaskan sumber daya `hamster-vpa` untuk melihat rekomendasi baru.

```
kubectl describe vpa/hamster-vpa
```

Contoh output adalah sebagai berikut.

```
Name:         hamster-vpa
Namespace:    default
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"autoscaling.k8s.io/
v1beta2","kind":"VerticalPodAutoscaler","metadata":{"annotations":
{},"name":"hamster-vpa","namespace":"d...
API Version:  autoscaling.k8s.io/v1beta2
Kind:         VerticalPodAutoscaler
Metadata:
```

```
  Creation Timestamp:  2019-09-27T18:22:51Z
  Generation:          23
  Resource Version:    14411
  Self Link:           /apis/autoscaling.k8s.io/v1beta2/namespaces/default/
verticalpodautoscalers/hamster-vpa
  UID:                 d0d85fb9-e153-11e9-ae53-0205785d75b0
Spec:
  Target Ref:
    API Version:  apps/v1
    Kind:         Deployment
    Name:         hamster
Status:
  Conditions:
    Last Transition Time:  2019-09-27T18:23:28Z
    Status:                True
    Type:                  RecommendationProvided
  Recommendation:
    Container Recommendations:
      Container Name:  hamster
      Lower Bound:
        Cpu:     550m
        Memory:  262144k
      Target:
        Cpu:     587m
        Memory:  262144k
      Uncapped Target:
        Cpu:     587m
        Memory:  262144k
      Upper Bound:
        Cpu:     21147m
        Memory:  387863636
Events:          <none>
```

7. Ketika Anda selesai bereksperimen dengan contoh aplikasi, Anda dapat menghapusnya dengan perintah berikut.

```
kubectl delete -f examples/hamster.yaml
```

Penskala Otomatis Pod Horizontal

KubernetesHorizontal Pod Autoscaler secara otomatis menskalakan jumlah Pods dalam penerapan, pengontrol replikasi, atau set replika berdasarkan pemanfaatan CPU sumber daya tersebut. Hal ini dapat membantu aplikasi Anda menskalakan keluar untuk memenuhi peningkatan permintaan atau penskalaan kedalam ketika sumber daya tidak diperlukan, sehingga membebaskan simpul Anda untuk aplikasi lain. Bila Anda mengatur persentase pemanfaatan CPU target, Penskala Otomatis Pod Horizontal menskalakan aplikasi Anda kedalam atau keluar untuk mencoba memenuhi target tersebut.

Horizontal Pod AutoscalerIni adalah sumber daya API standar Kubernetes yang hanya mengharuskan sumber metrik (seperti server Kubernetes metrik) diinstal pada kluster Amazon EKS Anda agar berfungsi. Anda tidak perlu menyebarkan atau menginstal Horizontal Pod Autoscaler pada cluster Anda untuk mulai menskalakan aplikasi Anda. Untuk informasi lebih lanjut, lihat Horizontal Pod Autoscaler di Kubernetes dokumentasi.

Gunakan topik ini untuk mempersiapkan klaster Amazon EKS Anda dan untuk memverifikasi bahwa itu berfungsi dengan aplikasi sampel. Horizontal Pod Autoscaler

> **Note**
>
> Topik ini didasarkan pada Horizontal Pod autoscalerpanduan dalam dokumentasi. Kubernetes

Prasyarat

- Anda memiliki klaster Amazon EKS yang ada. Jika Anda tidak melakukannya, tinjau Memulai dengan Amazon EKS.
- Anda telah menginstal Server Kubernetes Metrik. Untuk informasi selengkapnya, lihat Menginstal Server Kubernetes Metrik.
- Anda menggunakan `kubectl` klien yang dikonfigurasi untuk berkomunikasi dengan klaster Amazon EKS.

Jalankan aplikasi uji Penskala Otomatis Pod Horizontal

Dalam bagian ini, Anda men-deploy contoh aplikasi untuk memverifikasi bahwa Penskala Otomatis Pod Horizontal bekerja.

> **Note**
>
> Contoh ini didasarkan pada panduan Pod autoscaler Horizontal dalam dokumentasi. Kubernetes

Untuk menguji instalasi Penskala Otomatis Pod Horizontal

1. Deploy aplikasi web server Apache sederhana dengan perintah berikut.

   ```
   kubectl apply -f https://k8s.io/examples/application/php-apache.yaml
   ```

   Server web Apache Pod ini diberi batas CPU 500 millicpu dan melayani pada port 80.

2. Buat sumber daya Penskala Otomatis Pod Horizontal untuk deployment `php-apache`.

   ```
   kubectl autoscale deployment php-apache --cpu-percent=50 --min=1 --max=10
   ```

   Perintah ini menciptakan autoscaler yang menargetkan 50 persen pemanfaatan CPU untuk penyebaran, dengan minimal satu Pod dan maksimal sepuluh. Pods Ketika beban CPU rata-rata lebih rendah dari 50 persen, autoscaler mencoba mengurangi jumlah Pods dalam penyebaran, menjadi minimal satu. Ketika beban lebih besar dari 50 persen, autoscaler mencoba untuk meningkatkan jumlah Pods dalam penyebaran, hingga maksimal sepuluh. Untuk informasi selengkapnya, lihat Bagaimana cara HorizontalPodAutoscaler kerja? dalam Kubernetes dokumentasi.

3. Jelaskan penskala otomatis dengan perintah berikut untuk menampilkan detailnya.

   ```
   kubectl get hpa
   ```

   Contoh output adalah sebagai berikut.

   ```
   NAME         REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
   php-apache   Deployment/php-apache   0%/50%    1         10        1          51s
   ```

   Seperti yang Anda lihat, beban CPU saat ini adalah `0%`, karena belum ada muatan di server. PodHitungannya sudah berada pada batas terendahnya (satu), sehingga tidak dapat diskalakan.

4. Membuat beban untuk web server dengan menjalankan kontainer.

```
kubectl run -i \
    --tty load-generator \
    --rm --image=busybox \
    --restart=Never \
    -- /bin/sh -c "while sleep 0.01; do wget -q -O- http://php-apache; done"
```

5. Untuk melihat deployment menskalakan keluar, jalankan perintah berikut secara berkala di terminal yang terpisah dari terminal yang Anda jalankan dari langkah sebelumnya.

```
kubectl get hpa php-apache
```

Contoh output adalah sebagai berikut.

```
NAME         REFERENCE                TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   250%/50%   1         10        5
 4m44s
```

Mungkin perlu waktu lebih dari satu menit untuk jumlah replika meningkat. Selama persentase CPU sebenarnya lebih tinggi dari persentase target, maka jumlah replika meningkat, hingga 10. Dalam hal ini, `250%`, sehingga jumlah `REPLICAS` terus meningkat.

> **Note**
>
> Butuh beberapa menit sebelum Anda melihat jumlah replika mencapai maksimum. Jika hanya 6 replika, misalnya, yang diperlukan untuk beban CPU untuk tetap atau di bawah 50%, maka beban skala tidak akan melampaui 6 replika.

6. Hentikan beban. Di jendela terminal Anda menghasilkan beban, hentikan beban dengan menahan `Ctrl+C` tombol. Anda dapat melihat skala replika kembali ke 1 dengan menjalankan perintah berikut lagi di terminal tempat Anda menonton penskalaan.

```
kubectl get hpa
```

Contoh output adalah sebagai berikut.

```
NAME         REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   0%/50%    1         10        1          25m
```

> **Note**
>
> Kerangka waktu default untuk penskalaan mundur adalah lima menit, sehingga akan memakan waktu sebelum Anda melihat jumlah replika mencapai 1 lagi, bahkan ketika persentase CPU saat ini adalah 0 persen. Kerangka waktu dapat dimodifikasi. Untuk informasi selengkapnya, lihat Horizontal Pod Autoscaler dalam dokumentasi. Kubernetes

7. Setelah Anda selesai bereksperimen dengan aplikasi sampel Anda, hapus sumber daya `php-apache`.

```
kubectl delete deployment.apps/php-apache service/php-apache
 horizontalpodautoscaler.autoscaling/php-apache
```

Menyeimbangkan beban jaringan di Amazon EKS

Lalu lintas jaringan seimbang beban `L4` pada model OSI. Untuk memuat lalu lintas aplikasi keseimbangan di`L7`, Anda menerapkan Kubernetes`ingress`, yang menyediakan AWS Application Load Balancer. Untuk informasi selengkapnya, lihat Penyeimbangan beban aplikasi pada Amazon EKS. Untuk mempelajari lebih lanjut tentang perbedaan antara kedua jenis load balancing, lihat fitur Elastic Load Balancing di AWS situs web.

Saat Anda membuat tipe Kubernetes `Service` of`LoadBalancer`, pengontrol penyeimbang beban penyedia AWS cloud membuat AWS Classic Load Balancer secara default, tetapi juga dapat membuat AWS Network Load Balancer. Pengontrol ini hanya menerima perbaikan bug kritis di masa mendatang. Untuk informasi selengkapnya tentang penggunaan penyeimbang beban penyedia AWS AWS cloud, lihat pengontrol penyeimbang beban penyedia cloud di Kubernetes dokumentasi. Penggunaannya tidak tercakup dalam topik ini.

Kami menyarankan Anda menggunakan versi `2.7.2` atau yang lebih baru AWS Load Balancer Controllerdaripada pengontrol penyeimbang beban penyedia AWS cloud. AWS Load Balancer ControllerIni membuat AWS Network Load Balancers, tetapi tidak membuat AWS Classic Load Balancer. Sisa topik ini adalah tentang menggunakan AWS Load Balancer Controller.

AWS Network Load Balancer dapat memuat lalu lintas jaringan keseimbangan untuk Pods digunakan ke IP Amazon EC2 dan target instans atau ke target IP. AWS Fargate Untuk informasi selengkapnya, lihat AWS Load Balancer Controller di. GitHub

Prasyarat

Sebelum Anda dapat memuat lalu lintas jaringan keseimbangan menggunakanAWS Load Balancer Controller, Anda harus memenuhi persyaratan berikut.

- Memiliki klaster yang ada. Jika Anda tidak memiliki klaster yang ada, lihat Memulai dengan Amazon EKS. Jika Anda perlu memperbarui versi klaster yang ada, lihat Memperbarui Kubernetes versi cluster Amazon EKS.
- Miliki yang AWS Load Balancer Controller diterapkan di cluster Anda. Untuk informasi selengkapnya, lihat Apa itu AWS Load Balancer Controller?. Kami merekomendasikan versi `2.7.2` atau yang lebih baru.
- Setidaknya satu subnet. Jika beberapa subnet yang diberi tag ditemukan di Availability Zone, controller memilih subnet pertama yang subnetnya ID didahulukan secara leksikografis. Subnet harus memiliki setidaknya delapan alamat IP yang tersedia.
- Jika Anda menggunakan AWS Load Balancer Controller versi `2.1.1` atau sebelumnya, subnet harus diberi tag sebagai berikut. Jika menggunakan versi `2.1.2` atau yang lebih baru, tag ini opsional. Anda mungkin ingin menandai subnet jika Anda memiliki beberapa cluster yang berjalan di VPC yang sama, atau beberapa AWS layanan berbagi subnet di VPC, dan ingin lebih banyak kontrol atas tempat penyeimbang beban disediakan untuk setiap cluster. Jika Anda secara eksplisit menentukan ID subnet sebagai anotasi pada objek layanan, maka Kubernetes dan AWS Load Balancer Controller gunakan subnet tersebut secara langsung untuk membuat penyeimbang beban. Penandaan subnet tidak diperlukan jika Anda memilih untuk menggunakan metode ini untuk menyediakan penyeimbang beban dan Anda dapat melewati persyaratan penandaan subnet pribadi dan publik berikut. Ganti *`my-cluster`* dengan nama klaster Anda.
  - Kunci – `kubernetes.io/cluster/`*`my-cluster`*
  - Nilai – `shared` atau `owned`
- Subnet publik dan pribadi Anda harus memenuhi persyaratan berikut, kecuali jika Anda secara eksplisit menentukan ID subnet sebagai anotasi pada objek layanan atau ingress. Jika Anda menyediakan penyeimbang beban dengan secara eksplisit menentukan ID subnet sebagai anotasi pada objek layanan atau ingress, maka AWS Load Balancer Controller gunakan subnet tersebut secara langsung untuk membuat penyeimbang beban Kubernetes dan tag berikut tidak diperlukan.
  - Subnet pribadi — Harus ditandai dalam format berikut. Hal ini agar Kubernetes dan AWS Load Balancer Controller tahu bahwa subnet dapat digunakan untuk penyeimbang beban internal. Jika Anda menggunakan `eksctl` atau AWS AWS CloudFormation template Amazon EKS untuk membuat VPC Anda setelah 26 Maret 2020, maka subnet diberi tag dengan tepat saat dibuat.

Untuk informasi selengkapnya tentang Amazon EKS AWS AWS CloudFormation template-template VPC, lihat Membuat VPC untuk klaster Amazon EKS Anda.

- Kunci – `kubernetes.io/role/internal-elb`
- Nilai – 1

- Subnet publik — Harus ditandai dalam format berikut. Ini agar Kubernetes tahu untuk hanya menggunakan subnet tersebut untuk penyeimbang beban eksternal alih-alih memilih subnet publik di setiap Availability Zone (berdasarkan urutan leksikografis ID subnet). Jika Anda menggunakan `eksctl` atau AWS CloudFormation template Amazon EKS untuk membuat VPC Anda setelah 26 Maret 2020, maka subnet diberi tag dengan tepat saat dibuat. Untuk informasi selengkapnya tentang template AWS CloudFormation VPC Amazon EKS, lihat. Membuat VPC untuk klaster Amazon EKS Anda
  - Kunci – `kubernetes.io/role/elb`
  - Nilai – 1

Jika tag peran subnet tidak ditambahkan secara eksplisit, pengontrol Kubernetes layanan memeriksa tabel rute subnet VPC klaster Anda untuk menentukan apakah subnet bersifat pribadi atau publik. Kami menyarankan agar Anda tidak mengandalkan perilaku ini, dan sebagai gantinya secara eksplisit menambahkan tag peran pribadi atau publik. AWS Load Balancer ControllerTidak memeriksa tabel rute, dan membutuhkan tag pribadi dan publik untuk hadir untuk penemuan otomatis yang sukses.

Pertimbangan

- Konfigurasi penyeimbang beban Anda dikendalikan oleh anotasi yang ditambahkan ke manifes untuk layanan Anda. Anotasi layanan berbeda saat menggunakan AWS Load Balancer Controller daripada saat menggunakan pengontrol penyeimbang beban penyedia AWS cloud. Pastikan untuk meninjau anotasi untuk AWS Load Balancer Controller sebelum menerapkan layanan.
- Saat menggunakan Amazon VPC CNI plugin for Kubernetes, saldo AWS Load Balancer Controller dapat memuat ke IP Amazon EC2 atau target instans dan target IP Fargate. Saat menggunakan plugin CNI yang kompatibel dengan Alternatif, pengontrol hanya dapat memuat keseimbangan ke target instance. Untuk informasi selengkapnya tentang jenis target Network Load Balancer, lihat Jenis target di Panduan Pengguna untuk Network Load Balancers
- Jika Anda ingin menambahkan tag ke penyeimbang beban saat atau setelah dibuat, tambahkan anotasi berikut dalam spesifikasi layanan Anda. Untuk informasi selengkapnya, lihat Tag AWS Sumber Daya dalam AWS Load Balancer Controller dokumentasi.

```
service.beta.kubernetes.io/aws-load-balancer-additional-resource-tags
```

- Anda dapat menetapkan Alamat IP Elastis ke Network Load Balancer dengan menambahkan anotasi berikut. Ganti *example values*dengan alamat IP Elastis Anda. `Allocation IDs` Jumlah `Allocation IDs` harus sesuai dengan jumlah subnet yang digunakan untuk penyeimbang beban. Untuk informasi lebih lanjut, lihat AWS Load Balancer Controllerdokumentasi.

```
service.beta.kubernetes.io/aws-load-balancer-eip-allocations:
 eipalloc-xxxxxxxxxxxxxxxxx,eipalloc-yyyyyyyyyyyyyyyyy
```

- Amazon EKS menambahkan satu aturan masuk ke grup keamanan node untuk lalu lintas klien dan satu aturan untuk setiap subnet penyeimbang beban di VPC untuk pemeriksaan kesehatan untuk setiap Network Load Balancer yang Anda buat. Deployment layanan jenis `LoadBalancer` bisa gagal jika Amazon EKS mencoba untuk membuat aturan yang melebihi kuota untuk jumlah maksimum aturan yang diperbolehkan untuk grup keamanan. Untuk informasi selengkapnya, lihat Grup keamanan untuk Amazon VPC Anda di Panduan Pengguna Amazon VPC. Pertimbangkan opsi berikut untuk meminimalkan kemungkinan melebihi jumlah aturan maksimum untuk grup keamanan:
  - Meminta peningkatan aturan Anda per kuota grup keamanan. Untuk informasi selengkapnya, lihat Permintaan peningkatan kuota dalam Panduan Pengguna Service Quotas.
  - Gunakan target IP, bukan target instance. Dengan target IP, Anda dapat berbagi aturan untuk port target yang sama. Anda dapat menentukan subnet penyeimbang beban secara manual dengan anotasi. Untuk informasi selengkapnya, lihat Anotasi diGitHub.
  - Gunakan ingress, bukan layanan tipe`LoadBalancer`, untuk mengirim lalu lintas ke layanan Anda. AWS Application Load Balancer membutuhkan aturan yang lebih sedikit daripada Network Load Balancer. Anda dapat berbagi ALB di beberapa ingress. Untuk informasi selengkapnya, lihat Penyeimbangan beban aplikasi pada Amazon EKS. Anda tidak dapat membagikan Network Load Balancer di beberapa layanan.
  - Deploy klaster Anda ke beberapa akun.
- Jika Anda Pods berjalan Windows di kluster Amazon EKS, satu layanan dengan penyeimbang beban dapat mendukung hingga 1024 back-end. Pods Masing-masing Pod memiliki alamat IP uniknya sendiri.
- Kami merekomendasikan hanya membuat Network Load Balancers baru dengan. AWS Load Balancer Controller Mencoba mengganti Network Load Balancer yang ada yang dibuat dengan

pengontrol penyeimbang beban penyedia AWS cloud dapat menghasilkan beberapa Network Load Balancer yang dapat menyebabkan downtime aplikasi.

Buat penyeimbang beban jaringan

Anda dapat membuat penyeimbang beban jaringan dengan IP atau target instans.

IP targets

Anda dapat menggunakan target IP dengan Pods dikerahkan ke node Amazon EC2 atau Fargate. KubernetesLayanan Anda harus dibuat sebagai tipe`LoadBalancer`. Untuk informasi selengkapnya, lihat LoadBalancerMengetik Kubernetes dokumentasi.

Untuk membuat penyeimbang beban yang menggunakan target IP, tambahkan anotasi berikut ke manifes layanan dan terapkan layanan Anda. `external`Nilai untuk `aws-load-balancer-type` inilah yang menyebabkanAWS Load Balancer Controller, bukan pengontrol penyeimbang beban penyedia AWS cloud, untuk membuat Network Load Balancer. Anda dapat melihat sampel layanan manifes dengan anotasi.

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "ip"
```

> **Note**
>
> Jika Anda memuat balancing `IPv6`Pods, tambahkan anotasi berikut. Anda hanya dapat memuat keseimbangan `IPv6` ke target IP, bukan target instance. Tanpa anotasi ini, load balancing selesai. `IPv4`
>
> ```
> service.beta.kubernetes.io/aws-load-balancer-ip-address-type: dualstack
> ```

Network Load Balancers dibuat dengan `internal`aws-load-balancer-scheme, secara default. Anda dapat meluncurkan Network Load Balancers di subnet apa pun di VPC klaster Anda, termasuk subnet yang tidak ditentukan saat membuat klaster.

Kubernetesmemeriksa tabel rute untuk subnet Anda untuk mengidentifikasi apakah mereka publik atau pribadi. Sub-jaringan publik memiliki rute langsung ke internet menggunakan gateway internet, tetapi subnet privat tidak.

Jika Anda ingin membuat Network Load Balancer di subnet publik untuk memuat saldo ke node Amazon EC2 (Fargate hanya dapat bersifat pribadi), tentukan dengan anotasi berikut: `internet-facing`

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Note**
>
> `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"`Anotasi masih didukung untuk kompatibilitas mundur. Namun, sebaiknya gunakan anotasi sebelumnya untuk penyeimbang beban baru, bukan. `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"`

> **Important**
>
> Jangan mengedit anotasi setelah membuat layanan Anda. Jika Anda perlu memodifikasinya, hapus objek layanan dan buat lagi dengan nilai yang diinginkan untuk anotasi ini.

Instance targets

Pengontrol penyeimbang beban penyedia AWS cloud membuat Network Load Balancer hanya dengan target instans. Versi `2.2.0` dan yang lebih baru dari AWS Load Balancer Controller juga membuat Network Load Balancers dengan target instans. Kami merekomendasikan untuk menggunakannya, daripada pengontrol penyeimbang beban penyedia AWS cloud, untuk membuat Network Load Balancer baru. Anda dapat menggunakan target instans Network Load Balancer dengan Pods di-deploy ke node Amazon EC2, tetapi tidak ke Fargate. Untuk memuat lalu lintas jaringan keseimbangan di seluruh Pods dikerahkan ke Fargate, Anda harus menggunakan target IP.

Untuk menerapkan Network Load Balancer ke subnet pribadi, spesifikasi layanan Anda harus memiliki anotasi berikut. Anda dapat melihat sampel layanan manifes dengan anotasi. `external`Nilai untuk `aws-load-balancer-type` itulah yang menyebabkan AWS Load Balancer Controller, bukan pengontrol penyeimbang beban penyedia AWS cloud, untuk membuat Network Load Balancer.

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "instance"
```

Network Load Balancers dibuat dengan `internalaws-load-balancer-scheme`, secara default. Untuk Network Load Balancer internal, klaster Amazon EKS Anda harus dikonfigurasi untuk menggunakan setidaknya satu subnet pribadi di VPC Anda. Kubernetesmemeriksa tabel rute untuk subnet Anda untuk mengidentifikasi apakah mereka publik atau pribadi. Sub-jaringan publik memiliki rute langsung ke internet menggunakan gateway internet, tetapi subnet privat tidak.

Jika Anda ingin membuat Network Load Balancer di subnet publik untuk memuat saldo ke node Amazon EC2, tentukan `internet-facing` dengan anotasi berikut:

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Important**
>
> Jangan mengedit anotasi setelah membuat layanan Anda. Jika Anda perlu memodifikasinya, hapus objek layanan dan buat lagi dengan nilai yang diinginkan untuk anotasi ini.

(Opsional) Deploy aplikasi sampel

Prasyarat

- Setidaknya satu subnet publik atau pribadi di VPC cluster Anda.
- Miliki yang AWS Load Balancer Controller diterapkan di cluster Anda. Untuk informasi selengkapnya, lihat Apa itu AWS Load Balancer Controller?. Kami merekomendasikan versi `2.7.2` atau yang lebih baru.

Untuk menggunakan aplikasi sampel

1. Jika Anda menerapkan ke Fargate, pastikan Anda memiliki subnet pribadi yang tersedia di VPC Anda dan buat profil Fargate. Jika Anda tidak menerapkan ke Fargate, lewati langkah ini. Anda dapat membuat profil dengan menjalankan perintah berikut atau AWS Management

Consolemenggunakan nilai yang sama untuk `name` dan `namespace` yang ada di perintah. Ganti *`example values`* dengan milik Anda sendiri.

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --region region-code \
    --name nlb-sample-app \
    --namespace nlb-sample-app
```

2. Deploy aplikasi sampel.

   a. Buat namespace untuk aplikasi.

   ```
   kubectl create namespace nlb-sample-app
   ```

   b. Simpan konten berikut ini ke file bernama *`sample-deployment`*`.yaml` pada komputer Anda.

   ```
   apiVersion: apps/v1
   kind: Deployment
   metadata:
     name: nlb-sample-app
     namespace: nlb-sample-app
   spec:
     replicas: 3
     selector:
       matchLabels:
         app: nginx
     template:
       metadata:
         labels:
           app: nginx
       spec:
         containers:
           - name: nginx
             image: public.ecr.aws/nginx/nginx:1.23
             ports:
               - name: tcp
                 containerPort: 80
   ```

   c. Menerapkan manifes ke klaster.

```
kubectl apply -f sample-deployment.yaml
```

3. Buat layanan dengan Network Load Balancer yang menghadap ke internet yang memuat saldo ke target IP.

   a. Simpan konten berikut ini ke file bernama *sample-service*.yaml pada komputer Anda. Jika Anda menerapkan ke node Fargate, hapus `service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing` baris.

```
apiVersion: v1
kind: Service
metadata:
  name: nlb-sample-service
  namespace: nlb-sample-app
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-type: external
    service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: ip
    service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing
spec:
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
  type: LoadBalancer
  selector:
    app: nginx
```

   b. Menerapkan manifes ke klaster.

```
kubectl apply -f sample-service.yaml
```

4. Verifikasi bahwa layanan telah di-deploy.

```
kubectl get svc nlb-sample-service -n nlb-sample-app
```

   Contoh output adalah sebagai berikut.

```
NAME                TYPE           CLUSTER-IP         EXTERNAL-IP
                                                  PORT(S)        AGE
```

```
sample-service  LoadBalancer   10.100.240.137
 k8s-nlbsampl-nlbsampl-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.region-code.amazonaws.com
  80:32400/TCP   16h
```

> **Note**
>
> Nilai untuk *10.100.240.137* dan *xxxxxxxxxx* - *xxxxxxxxxxxxxxxx* akan berbeda dari output contoh (mereka akan unik untuk penyeimbang beban Anda) dan *us-west-2* mungkin berbeda untuk Anda, tergantung di mana cluster Anda berada. Wilayah AWS

5. Buka Amazon EC2 AWS Management Console. Pilih Grup Target (di bawah Load Balancing) di panel navigasi kiri. Di kolom Nama, pilih nama grup target di mana nilai di kolom Load balancer cocok dengan sebagian nama di `EXTERNAL-IP` kolom output pada langkah sebelumnya. Misalnya, Anda akan memilih grup target bernama `k8s-default-samplese-`*xxxxxxxxxx* jika output Anda sama dengan output sebelumnya. Jenis Target adalah `IP` karena itu ditentukan dalam manifes layanan sampel.
6. Pilih Grup target Anda, kemudian pilih tab Target. Di bawah Target terdaftar, Anda akan melihat tiga alamat IP replika tiga yang di-deploy di langkah sebelumnya. Tunggu sampai status semua target baik sebelum melanjutkan. Mungkin perlu beberapa menit sebelum semua target beradahealthy. Target mungkin dalam unhealthy keadaan sebelum berubah menjadi `healthy` negara.
7. Kirim lalu lintas ke layanan yang diganti *xxxxxxxxxx-xxxxxxxxxxxxxxxx* dan *us-west-2* dengan nilai yang dikembalikan dalam output untuk langkah sebelumnya untuk`EXTERNAL-IP`. Jika Anda menerapkan ke subnet pribadi, maka Anda harus melihat halaman dari perangkat dalam VPC Anda, seperti host bastion. Untuk informasi selengkapnya, lihat Host Linux Bastion di AWS.

   ```
   curl k8s-default-samplese-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.region-code.amazonaws.com
   ```

   Contoh output adalah sebagai berikut.

   ```
   <!DOCTYPE html>
   <html>
   <head>
   <title>Welcome to nginx!</title>
   [...]
   ```
8. Setelah selesai menggunakan contoh deployment, service, dan namespace, hapus saja.

```
kubectl delete namespace nlb-sample-app
```

Penyeimbangan beban aplikasi pada Amazon EKS

Saat Anda membuat Kubernetes`ingress`, AWS Application Load Balancer (ALB) disediakan yang memuat saldo lalu lintas aplikasi. Untuk mempelajari lebih lanjut, lihat Apa itu Application Load Balancer? dalam Panduan Pengguna Application Load Balancers dan Ingress dalam dokumentasi. Kubernetes ALB dapat digunakan dengan Pods yang digunakan ke node atau ke. AWS Fargate Anda dapat men-deploy ALB ke subnet publik atau privat.

Lalu lintas aplikasi seimbang `L7` pada model OSI. Untuk memuat lalu lintas jaringan keseimbangan di`L4`, Anda menggunakan `LoadBalancer` tipe. Kubernetes `service` Jenis ini menyediakan AWS Network Load Balancer. Untuk informasi selengkapnya, lihat Menyeimbangkan beban jaringan di Amazon EKS. Untuk mempelajari selengkapnya tentang perbedaan antara kedua jenis penyeimbangan beban, lihat fitur-fitur Elastic Load Balancing pada situs web AWS .

Prasyarat

Sebelum Anda dapat menyeimbangkan beban lalu lintas aplikasi ke suatu aplikasi, Anda harus memenuhi persyaratan berikut.

- Memiliki klaster yang ada. Jika Anda tidak memiliki klaster yang ada, lihat Memulai dengan Amazon EKS. Jika Anda perlu memperbarui versi klaster yang ada, lihat Memperbarui Kubernetes versi cluster Amazon EKS.
- Miliki yang AWS Load Balancer Controller diterapkan di cluster Anda. Untuk informasi selengkapnya, lihat Apa itu AWS Load Balancer Controller?. Kami merekomendasikan versi `2.7.2` atau yang lebih baru.
- Setidaknya dua subnet harus berada di Availability Zone yang berbeda. AWS Load Balancer ControllerMemilih satu subnet dari setiap Availability Zone. Ketika beberapa subnet yang diberi tag ditemukan di Availability Zone, controller memilih subnet yang subnetnya ID didahulukan secara leksikografis. Setiap subnet harus memiliki setidaknya delapan alamat IP yang tersedia.

  Jika Anda menggunakan beberapa grup keamanan yang dilampirkan ke node pekerja, tepat satu grup keamanan harus diberi tag sebagai berikut. Ganti *`my-cluster`* dengan nama klaster Anda.

  - Kunci – `kubernetes.io/cluster/`*`my-cluster`*
  - Nilai – `shared` atau `owned`

- Jika Anda menggunakan AWS Load Balancer Controller versi `2.1.1` atau versi sebelumnya, subnet harus diberi tag dalam format berikut. Jika Anda menggunakan versi `2.1.2` atau yang lebih baru, penandaan bersifat opsional. Namun, kami menyarankan Anda menandai subnet jika salah satu dari yang berikut ini terjadi. Anda memiliki beberapa cluster yang berjalan di VPC yang sama, atau memiliki AWS beberapa layanan yang berbagi subnet dalam VPC. Atau, Anda ingin lebih banyak kontrol di mana penyeimbang beban disediakan untuk setiap cluster. Ganti *`my-cluster`* dengan nama klaster Anda.
  - Kunci – `kubernetes.io/cluster/`*`my-cluster`*
  - Nilai – `shared` atau `owned`
- Subnet publik dan pribadi Anda harus memenuhi persyaratan berikut. Ini kecuali Anda secara eksplisit menentukan ID subnet sebagai anotasi pada objek layanan atau ingress. Misalnya Anda menyediakan penyeimbang beban dengan secara eksplisit menentukan ID subnet sebagai anotasi pada objek layanan atau ingress. Dalam situasi ini, Kubernetes dan pengontrol penyeimbang AWS beban menggunakan subnet tersebut secara langsung untuk membuat penyeimbang beban dan tag berikut tidak diperlukan.
  - Subnet pribadi — Harus ditandai dalam format berikut. Ini agar Kubernetes dan pengontrol penyeimbang AWS beban tahu bahwa subnet dapat digunakan untuk penyeimbang beban internal. Jika Anda menggunakan `eksctl` atau AWS CloudFormation template Amazon EKS untuk membuat VPC Anda setelah 26 Maret 2020, subnet diberi tag dengan tepat saat dibuat. Untuk informasi selengkapnya tentang template AWS CloudFormation VPC Amazon EKS, lihat. Membuat VPC untuk klaster Amazon EKS Anda
    - Kunci – `kubernetes.io/role/internal-elb`
    - Nilai – `1`
  - Subnet publik — Harus ditandai dalam format berikut. Ini agar Kubernetes tahu hanya menggunakan subnet yang ditentukan untuk penyeimbang beban eksternal. Dengan cara ini, Kubernetes tidak memilih subnet publik di setiap Availability Zone (secara leksikografis berdasarkan ID subnet mereka). Jika Anda menggunakan `eksctl` atau AWS CloudFormation template Amazon EKS untuk membuat VPC Anda setelah 26 Maret 2020, subnet diberi tag dengan tepat saat dibuat. Untuk informasi selengkapnya tentang template AWS CloudFormation VPC Amazon EKS, lihat. Membuat VPC untuk klaster Amazon EKS Anda
    - Kunci – `kubernetes.io/role/elb`
    - Nilai – `1`

  Jika tag peran subnet tidak ditambahkan secara eksplisit, pengontrol Kubernetes layanan memeriksa tabel rute subnet VPC klaster Anda. Ini untuk menentukan apakah subnet bersifat

pribadi atau publik. Kami menyarankan Anda untuk tidak bergantung pada perilaku ini. Sebaliknya, tambahkan tag peran pribadi atau publik secara eksplisit. AWS Load Balancer ControllerItu tidak memeriksa tabel rute. Ini juga membutuhkan tag pribadi dan publik untuk hadir untuk penemuan auto yang sukses.

Pertimbangan

- AWS Load Balancer Controller membuat ALB dan AWS sumber daya pendukung yang diperlukan setiap kali sumber daya Kubernetes ingress dibuat di cluster dengan anotasi. `kubernetes.io/ingress.class: alb` Sumber daya ingress mengonfigurasi ALB untuk merutekan lalu lintas HTTP atau HTTPS ke yang berbeda Pods di dalam cluster. Untuk memastikan bahwa objek ingress Anda menggunakanAWS Load Balancer Controller, tambahkan anotasi berikut ke spesifikasi Kubernetes ingress Anda. Untuk informasi lebih lanjut, lihat spesifikasi Ingress padaGitHub.

```
annotations:
    kubernetes.io/ingress.class: alb
```

> **Note**
>
> Jika Anda memuat balancing `IPv6`Pods, tambahkan anotasi berikut ke spesifikasi ingress Anda. Anda hanya dapat memuat keseimbangan `IPv6` ke target IP, bukan target instance. Tanpa anotasi ini, load balancing selesai. `IPv4`
>
> ```
> alb.ingress.kubernetes.io/ip-address-type: dualstack
> ```

- Ini AWS Load Balancer Controller mendukung mode lalu lintas berikut:
  - Instans – Daftarkan simpul dalam klaster Anda sebagai target untuk ALB. Lalu lintas yang mencapai ALB diarahkan ke `NodePort` layanan Anda dan kemudian diproksi ke Anda. Pods Ini adalah mode lalu lintas default. Anda juga dapat secara eksplisit menentukannya dengan anotasi `alb.ingress.kubernetes.io/target-type: instance`.

    > **Note**
    >
    > KubernetesLayanan Anda harus menentukan jenis `NodePort` atau "LoadBalancer" untuk menggunakan mode lalu lintas ini.

  - IP — Mendaftar Pods sebagai target untuk ALB. Lalu lintas yang mencapai ALB langsung diarahkan ke layanan Pods Anda. Anda harus menentukan anotasi `alb.ingress.kubernetes.io/target-type: ip` untuk menggunakan mode lalu lintas ini. Jenis target IP diperlukan saat target Pods berjalan di Fargate.
- Untuk menandai ALB yang dibuat oleh pengendali, tambahkan anotasi berikut ke pengendali: `alb.ingress.kubernetes.io/tags`. Untuk daftar semua anotasi yang tersedia yang didukung olehAWS Load Balancer Controller, lihat anotasi Ingress di. GitHub
- Memutakhirkan atau menurunkan versi pengontrol ALB dapat memperkenalkan perubahan yang melanggar untuk fitur yang mengandalkannya. Untuk informasi selengkapnya tentang perubahan yang melanggar yang diperkenalkan di setiap rilis, lihat catatan rilis pengontrol ALB diGitHub.

Untuk berbagi penyeimbang beban aplikasi di beberapa sumber daya layanan menggunakan **`IngressGroups`**

Untuk menggabungkan ingress ke grup, tambahkan anotasi berikut ke spesifikasi sumber daya Kubernetes ingress.

```
alb.ingress.kubernetes.io/group.name: my-group
```

Nama grup harus:

- Panjangnya 63 atau lebih sedikit karakter.
- Terdiri dari huruf kecil, angka`-`, dan `.`
- Dimulai dan diakhiri dengan huruf atau angka.

Pengontrol secara otomatis menggabungkan aturan masuk untuk semua ingress dalam grup ingress yang sama. Ini mendukung mereka dengan satu ALB. Sebagian besar anotasi yang didefinisikan pada ingress hanya berlaku untuk jalur yang ditentukan oleh ingress tersebut. Secara default, sumber daya ingress bukan milik grup ingress mana pun.

> **Warning**
>
> Risiko keamanan potensial: Tentukan grup ingress untuk ingress hanya jika semua Kubernetes pengguna yang memiliki izin RBAC untuk membuat atau memodifikasi sumber daya masuk berada dalam batas kepercayaan yang sama. Jika Anda menambahkan anotasi dengan nama grup, Kubernetes pengguna lain mungkin membuat atau memodifikasi

masuknya agar menjadi milik grup ingress yang sama. Melakukan hal tersebut dapat menyebabkan perilaku yang tidak diinginkan, seperti menimpa aturan yang ada dengan aturan prioritas yang lebih tinggi.

Anda dapat menambahkan nomor pesanan sumber daya masuk Anda.

```
alb.ingress.kubernetes.io/group.order: '10'
```

Jumlahnya bisa 1-1000. Angka terendah untuk semua ingress dalam kelompok ingress yang sama dievaluasi terlebih dahulu. Semua masuknya tanpa anotasi ini dievaluasi dengan nilai nol. Aturan duplikat angka yang lebih tinggi dapat menimpa aturan dengan angka yang lebih rendah. Secara default, urutan aturan antara ingress dalam grup ingress yang sama ditentukan namespace dan nama berbasis leksikografis.

Important

Pastikan bahwa setiap ingress dalam grup ingress yang sama memiliki nomor prioritas yang unik. Anda tidak dapat memiliki nomor urutan duplikat di seluruh ingresses.

(Opsional) Deploy aplikasi sampel

Prasyarat

- Setidaknya satu subnet publik atau pribadi di VPC cluster Anda.
- Miliki yang AWS Load Balancer Controller diterapkan di cluster Anda. Untuk informasi selengkapnya, lihat Apa itu AWS Load Balancer Controller?. Kami merekomendasikan versi `2.7.2` atau yang lebih baru.

Untuk menggunakan aplikasi sampel

Anda dapat menjalankan aplikasi sampel pada cluster yang memiliki node Amazon EC2, FargatePods, atau keduanya.

1. Jika Anda tidak menerapkan ke Fargate, lewati langkah ini. Jika Anda men-deploy ke Fargate, buat profil Fargate. Anda dapat membuat profil dengan menjalankan perintah berikut atau AWS

Management Consolemenggunakan nilai yang sama untuk `name` dan `namespace` yang ada di perintah. Ganti *`example values`* dengan milik Anda sendiri.

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --region region-code \
    --name alb-sample-app \
    --namespace game-2048
```

2. Menyebarkan game 2048 sebagai contoh aplikasi untuk memverifikasi bahwa AWS Load Balancer Controller membuat AWS ALB sebagai hasil dari objek ingress. Selesaikan langkah-langkah untuk jenis subnet yang Anda gunakan.

   a. Jika Anda menerapkan ke Pods dalam klaster yang Anda buat bersama `IPv6` keluarga, lewati ke langkah berikutnya.

      - Publik

        ```
        kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
        ```

      - Pribadi

        1. Unduh manifes.

           ```
           curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
           ```

        2. Edit file dan temukan baris yang bertuliskan`alb.ingress.kubernetes.io/ scheme: internet-facing`.
        3. Ubah *`internet-facing`* ke **`internal`** dan simpan file.
        4. Menerapkan manifes ke klaster Anda.

           ```
           kubectl apply -f 2048_full.yaml
           ```

   b. Jika Anda menerapkan ke Pods dalam klaster yang Anda buat bersama `IPv6`keluarga, selesaikan langkah-langkah berikut.

      1. Unduh manifes.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
```

2. Buka file di editor dan tambahkan baris berikut ke anotasi dalam spesifikasi ingress.

```
alb.ingress.kubernetes.io/ip-address-type: dualstack
```

3. Jika Anda menyeimbangkan beban ke internalPods, bukan menghadap internetPods, ubah baris yang mengatakan `alb.ingress.kubernetes.io/scheme: internet-facing` `alb.ingress.kubernetes.io/scheme: internal`
4. Simpan file tersebut.
5. Menerapkan manifes ke klaster Anda.

```
kubectl apply -f 2048_full.yaml
```

3. Setelah beberapa menit, verifikasi bahwa sumber daya ingress dibuat dengan perintah berikut.

```
kubectl get ingress/ingress-2048 -n game-2048
```

Contoh output adalah sebagai berikut.

```
NAME           CLASS    HOSTS   ADDRESS
                       PORTS   AGE
ingress-2048   <none>   *       k8s-game2048-ingress2-xxxxxxxxxx-yyyyyyyyyy.region-code.elb.amazonaws.com   80      2m32s
```

> **Note**
>
> Jika Anda membuat penyeimbang beban di subnet pribadi, nilai ADDRESS di bawah output sebelumnya diawali dengan. `internal-`
> Jika ingress Anda tidak berhasil dibuat setelah beberapa menit, jalankan perintah berikut untuk melihat AWS Load Balancer Controller log. Log ini mungkin berisi pesan kesalahan yang dapat Anda gunakan untuk mendiagnosis masalah dengan penerapan Anda.
>
> ```
> kubectl logs -f -n kube-system -l app.kubernetes.io/instance=aws-load-balancer-controller
> ```

4. Jika Anda menggunakan subnet publik, buka browser dan arahkan ke ADDRESS URL dari output perintah sebelumnya untuk melihat contoh aplikasi. Jika Anda tidak melihat apa pun, segarkan browser Anda dan coba lagi. Jika Anda menerapkan ke subnet pribadi, maka Anda harus melihat halaman dari perangkat dalam VPC Anda, seperti host bastion. Untuk informasi selengkapnya, lihat Host Linux Bastion di AWS.
5. Ketika Anda selesai bereksperimen dengan aplikasi sampel Anda, hapus dengan menjalankan salah satu perintah berikut.

   - Jika Anda menerapkan manifes, daripada menerapkan salinan yang Anda unduh, gunakan perintah berikut.

     ```
     kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
     ```

   - Jika Anda mengunduh dan mengedit manifes, gunakan perintah berikut.

     ```
     kubectl delete -f 2048_full.yaml
     ```

Membatasi alamat IP eksternal yang dapat ditugaskan ke layanan

KubernetesLayanan dapat dicapai dari dalam cluster melalui:

- Alamat IP cluster yang ditetapkan secara otomatis oleh Kubernetes
- Alamat IP apa pun yang Anda tentukan untuk properti `externalIPs` dalam spesifikasi layanan. Alamat IP eksternal tidak dikelola oleh Kubernetes dan merupakan tanggung jawab administrator cluster. Alamat IP eksternal yang ditentukan dengan `externalIPs` berbeda dari alamat IP eksternal yang ditetapkan ke layanan jenis `LoadBalancer` oleh penyedia cloud.

Untuk mempelajari lebih lanjut tentang Kubernetes layanan, lihat Layanan dalam Kubernetes dokumentasi. Anda dapat membatasi alamat IP yang dapat ditentukan untuk `externalIPs` dalam spesifikasi layanan.

Untuk membatasi alamat IP yang dapat ditentukan untuk **`externalIPs`** dalam spesifikasi layanan

1. Terapkan `cert-manager` untuk mengelola sertifikat webhook. Untuk informasi lebih lanjut, lihat `cert-manager`dokumentasi.

```
kubectl apply -f https://github.com/jetstack/cert-manager/releases/download/v1.5.4/
cert-manager.yaml
```

2. Verifikasi bahwa `cert-manager` Pods sedang berjalan.

```
kubectl get pods -n cert-manager
```

Contoh output adalah sebagai berikut.

```
NAME                                       READY   STATUS    RESTARTS   AGE
cert-manager-58c8844bb8-nlx7q              1/1     Running   0          15s
cert-manager-cainjector-745768f6ff-696h5   1/1     Running   0          15s
cert-manager-webhook-67cc76975b-4v4nk      1/1     Running   0          14s
```

3. Tinjau layanan yang ada untuk memastikan bahwa tidak satu pun dari mereka yang memiliki alamat IP eksternal yang ditetapkan kepada mereka tidak terdapat dalam blok CIDR yang alamatnya ingin Anda batasi.

```
kubectl get services -A
```

Contoh output adalah sebagai berikut.

```
NAMESPACE                     NAME                                     TYPE
   CLUSTER-IP       EXTERNAL-IP     PORT(S)         AGE
cert-manager                  cert-manager                             ClusterIP
   10.100.102.137   <none>          9402/TCP        20m
cert-manager                  cert-manager-webhook                     ClusterIP
   10.100.6.136     <none>          443/TCP         20m
default                       kubernetes                               ClusterIP
   10.100.0.1       <none>          443/TCP         2d1h
externalip-validation-system   externalip-validation-webhook-service   ClusterIP
   10.100.234.179   <none>          443/TCP         16s
kube-system                   kube-dns                                 ClusterIP
   10.100.0.10      <none>          53/UDP,53/TCP   2d1h
my-namespace                  my-service                               ClusterIP
   10.100.128.10    192.168.1.1     80/TCP          149m
```

Jika salah satu nilai adalah alamat IP yang tidak berada dalam blok yang ingin Anda batasi aksesnya, Anda harus mengubah alamatnya agar berada di dalam blok, dan men-deploy ulang

layanan. Misalnya, pelayanan `my-service` di output sebelumnya memiliki alamat IP eksternal yang ditetapkan untuk itu yang tidak berada di dalam contoh blok CIDR pada langkah 5.

4. Unduh manifes webhook IP eksternal. Anda juga dapat melihat kode sumber untuk webhook diGitHub.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/externalip-webhook.yaml
```

5. Tentukan blok CIDR. Buka file yang diunduh dalam editor Anda dan hapus # pada awal baris berikut.

```
#args:
#- --allowed-external-ip-cidrs=10.0.0.0/8
```

Ganti `10.0.0.0/8` dengan blok CIDR Anda sendiri. Anda dapat menentukan blok sebanyak yang Anda inginkan. Jika Anda menentukan beberapa blok, tambahkan koma di antara blok.

6. Jika cluster Anda tidak dalam `us-west-2`Wilayah AWS, maka `r eplaceus-west-2`,`602401143452`, dan `amazonaws.com` dalam file dengan perintah berikut. Sebelum menjalankan perintah, ganti *`region-code`* dan *`111122223333`* dengan nilai untuk Anda Wilayah AWS dari daftar diPendaftar gambar kontainer Amazon.

```
sed -i.bak -e 's|602401143452|111122223333|' externalip-webhook.yaml
sed -i.bak -e 's|us-west-2|region-code|' externalip-webhook.yaml
sed -i.bak -e 's|amazonaws.com||' externalip-webhook.yaml
```

7. Menerapkan manifes ke klaster Anda.

```
kubectl apply -f externalip-webhook.yaml
```

Upaya untuk menyebarkan layanan ke klaster Anda dengan alamat IP yang ditentukan untuk `externalIPs` itu tidak terkandung dalam blok yang Anda tentukan dalam langkah Tentukan blok CIDR akan gagal.

Salin gambar kontainer dari satu repositori ke repositori lain

Topik ini menjelaskan cara menarik gambar kontainer dari repositori yang tidak dapat diakses oleh node Anda dan mendorong gambar ke repositori yang dapat diakses oleh node Anda. Anda dapat mendorong gambar ke Amazon ECR atau repositori alternatif yang dapat diakses oleh node Anda.

Prasyarat

- DockerMesin dipasang dan dikonfigurasi di komputer Anda. Untuk petunjuk, lihat Menginstal Docker Mesin di Docker dokumentasi.
- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atauAWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaruAWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLIVersi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Titik akhir VPC antarmuka untuk Amazon ECR jika Anda ingin node Anda menarik gambar kontainer dari atau mendorong gambar kontainer ke repositori ECR Amazon pribadi melalui jaringan Amazon. Untuk informasi selengkapnya, lihat Membuat titik akhir VPC untuk Amazon ECR di Panduan Pengguna Amazon Elastic Container Registry.

Selesaikan langkah-langkah berikut untuk menarik gambar kontainer dari repositori dan mendorongnya ke repositori Anda sendiri. Dalam contoh berikut yang disediakan dalam topik ini, gambar untuk pembantu Amazon VPC CNI plugin for Kubernetes metrik ditarik. Ketika Anda mengikuti langkah-langkah ini, pastikan untuk mengganti *`example values`* dengan nilai Anda sendiri.

Untuk menyalin gambar kontainer dari satu repositori ke repositori lain

1. Jika Anda belum memiliki repositori Amazon ECR atau repositori lain, buat repositori yang dapat diakses oleh node Anda. Perintah berikut membuat repositori pribadi Amazon ECR. Nama repositori pribadi Amazon ECR harus dimulai dengan huruf. Ini hanya dapat berisi huruf kecil, angka, tanda hubung (-), garis bawah (_), dan garis miring maju (/). Untuk informasi selengkapnya, lihat Membuat repositori pribadi di Panduan Pengguna Amazon Elastic Container Registry.

   Anda dapat mengganti *`cni-metrics-helper`* dengan apa pun yang Anda pilih. Sebagai praktik terbaik, buat repositori terpisah untuk setiap gambar. Kami merekomendasikan ini karena tag gambar harus unik dalam repositori. Ganti *`region-code`* dengan yang Wilayah AWSdidukung oleh Amazon ECR.

```
aws ecr create-repository --region region-code --repository-name cni-metrics-helper
```

2. Tentukan registri, repositori, dan tag (opsional) dari gambar yang perlu ditarik oleh node Anda. Informasi ini dalam `registry/repository[:tag]` format.

   Banyak topik Amazon EKS tentang menginstal gambar mengharuskan Anda menerapkan file manifes atau menginstal gambar menggunakan bagan Helm. Namun, sebelum Anda menerapkan file manifes atau menginstal bagan Helm, pertama-tama lihat konten manifes atau `values.yaml` file bagan. Dengan begitu, Anda dapat menentukan registri, repositori, dan tag yang akan ditarik.

   Misalnya, Anda dapat menemukan baris berikut dalam file manifes untuk pembantu Amazon VPC CNI plugin for Kubernetes metrik. Registri adalah`602401143452.dkr.ecr.us-west-2.amazonaws.com`, yang merupakan registri pribadi Amazon ECR. Repositori adalah. `cni-metrics-helper`

   ```
   image: "602401143452.dkr.ecr.us-west-2.amazonaws.com/cni-metrics-helper:v1.12.6"
   ```

   Anda dapat melihat variasi berikut untuk lokasi gambar:

   - Hanya`repository-name:tag`. Dalam hal ini, `docker.io` biasanya registri, tetapi tidak Kubernetes ditentukan karena menambahkan ke nama repositori secara default jika tidak ada registri yang ditentukan.
   - `repository-name/repository-namespace/repository:tag`. Namespace repositori bersifat opsional, tetapi terkadang ditentukan oleh pemilik repositori untuk mengkategorikan gambar. Misalnya, semua gambar Amazon EC2 di Galeri Publik Amazon ECR menggunakan namespace. `aws-ec2`

   Sebelum menginstal gambar dengan Helm, lihat `values.yaml` file Helm untuk menentukan lokasi gambar. Misalnya, `values.yaml` file untuk helper Amazon VPC CNI plugin for Kubernetes metrik menyertakan baris berikut.

   ```
   image:
     region: us-west-2
     tag: v1.12.6
     account: "602401143452"
     domain: "amazonaws.com"
   ```

3. Tarik gambar kontainer yang ditentukan dalam file manifes.

   a. Jika Anda menarik dari registri publik, seperti Galeri Publik Amazon ECR, Anda dapat melompat ke sub-langkah berikutnya, karena otentikasi tidak diperlukan. Dalam contoh ini, Anda mengautentikasi ke registri pribadi Amazon ECR yang berisi repositori untuk gambar helper metrik CNI. Amazon EKS mempertahankan gambar di setiap registri yang terdaftar diPendaftar gambar kontainer Amazon. Anda dapat mengautentikasi ke salah satu pendaftar dengan mengganti *602401143452* dan *region-code* dengan informasi untuk registri yang berbeda. Registri terpisah ada untuk masing-masing Wilayah AWStempat Amazon EKS didukung.

   ```
   aws ecr get-login-password --region region-code | docker login --username AWS --password-stdin 602401143452.dkr.ecr.region-code.amazonaws.com
   ```

   b. Tarik gambar. Dalam contoh ini, Anda menarik dari registri yang Anda autentikasi di sub-langkah sebelumnya. Ganti *602401143452* dan *region-code* dengan informasi yang Anda berikan di sub-langkah sebelumnya.

   ```
   docker pull 602401143452.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
   ```

4. Tandai gambar yang Anda tarik dengan registri, repositori, dan tag Anda. Contoh berikut mengasumsikan bahwa Anda menarik gambar dari file manifes dan akan mendorongnya ke repositori pribadi Amazon ECR yang Anda buat pada langkah pertama. Ganti *111122223333* dengan ID akun Anda. Ganti *region-code* dengan tempat Wilayah AWS Anda membuat repositori pribadi Amazon ECR Anda.

   ```
   docker tag cni-metrics-helper:v1.12.6 111122223333.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
   ```

5. Otentikasi ke registri Anda. Dalam contoh ini, Anda mengautentikasi ke registri pribadi Amazon ECR yang Anda buat pada langkah pertama. Untuk informasi selengkapnya, lihat Autentikasi registri di Panduan Pengguna Amazon Elastic Container Registry.

   ```
   aws ecr get-login-password --region region-code | docker login --username AWS --password-stdin 111122223333.dkr.ecr.region-code.amazonaws.com
   ```

6. Dorong gambar ke repositori Anda. Dalam contoh ini, Anda mendorong gambar ke repositori pribadi Amazon ECR yang Anda buat pada langkah pertama. Untuk informasi selengkapnya, lihat Mendorong Docker gambar di Panduan Pengguna Amazon Elastic Container Registry.

```
docker push 111122223333.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
```

7. Perbarui file manifes yang Anda gunakan untuk menentukan gambar pada langkah sebelumnya dengan gambar yang Anda dorong. `registry/repository:tag` Jika Anda menginstal dengan bagan Helm, sering ada opsi untuk menentukan. `registry/repository:tag` Saat memasang bagan, tentukan gambar `registry/repository:tag` yang Anda dorong ke repositori Anda.

Pendaftar gambar kontainer Amazon

Saat Anda menerapkan add-on AWS Amazon EKS ke kluster Anda, node Anda akan menarik gambar kontainer yang diperlukan dari registri yang ditentukan dalam mekanisme penginstalan untuk add-on, seperti manifes instalasi atau file Helm. `values.yaml` Gambar diambil dari repositori pribadi Amazon EKS Amazon ECR. Amazon EKS mereplikasi gambar ke repositori di setiap Amazon EKS yang didukung. Wilayah AWS Node Anda dapat menarik gambar kontainer melalui internet dari salah satu pendaftar berikut. Atau, node Anda dapat menarik gambar melalui jaringan Amazon jika Anda membuat titik akhir VPC antarmuka untuk Amazon ECR () AWS PrivateLink di VPC Anda. Registri memerlukan otentikasi dengan akun AWS IAM. Node Anda mengautentikasi menggunakan peran IAM node Amazon EKS, yang memiliki izin dalam kebijakan IAM ContainerRegistryReadOnly terkelola AmazonEC2 yang terkait dengannya.

Wilayah AWS	Registri
af-south-1	877085696533.dkr. ecr.af-south-1.amazonaws.com
ap-east-1	800184023465.dkr. ecr.ap-east-1.amazonaws.com
ap-northeast-1	602401143452.dkr. ecr.ap-northeast-1.amazonaws.com

Wilayah AWS	Registri
ap-northeast-2	602401143452.dkr. ecr.ap-northeast-2 .amazonaws.com
ap-northeast-3	602401143452.dkr. ecr.ap-northeast-3 .amazonaws.com
ap-south-1	602401143452.dkr. ecr.ap-south-1.ama zonaws.com
ap-south-2	900889452093.dkr. ecr.ap-south-2.ama zonaws.com
ap-southeast-1	602401143452.dkr. ecr.ap-southeast-1 .amazonaws.com
ap-southeast-2	602401143452.dkr. ecr.ap-southeast-2 .amazonaws.com
ap-southeast-3	296578399912.dkr. ecr.ap-southeast-3 .amazonaws.com
ap-southeast-4	491585149902.dkr. ecr.ap-southeast-4 .amazonaws.com
ca-central-1	602401143452.dkr. ecr.ca-central-1.a mazonaws.com
ca-barat-1	761377655185.dkr. ecr.ca-west-1.amaz onaws.com
cn-north-1	918309763551.dkr. ecr.cn-north-1.ama zonaws.com .cn
cn-northwest-1	961992271922.dkr. ecr.cn-northwest-1 .amazonaws.com .cn
eu-central-1	602401143452.dkr. ecr.eu-central-1.a mazonaws.com

Wilayah AWS	Registri
eu-central-2	900612956339.dkr. ecr.eu-central-2.amazonaws.com
eu-north-1	602401143452.dkr. ecr.eu-north-1.amazonaws.com
eu-south-1	590381155156.dkr. ecr.eu-south-1.amazonaws.com
eu-south-2	455263428931.dkr. ecr.eu-south-2.amazonaws.com
eu-west-1	602401143452.dkr. ecr.eu-west-1.amazonaws.com
eu-west-2	602401143452.dkr. ecr.eu-west-2.amazonaws.com
eu-west-3	602401143452.dkr. ecr.eu-west-3.amazonaws.com
pusat-1	066635153087.dkr. ecr.il-central-1.amazonaws.com
me-south-1	558608220178.dkr. ecr.me-south-1.amazonaws.com
eu-central-1	759879836304.dkr. ecr.me-central-1.amazonaws.com
sa-east-1	602401143452.dkr. ecr.sa-east-1.amazonaws.com
us-east-1	602401143452.dkr. ecr.us-east-1.amazonaws.com
us-east-2	602401143452.dkr. ecr.us-east-2.amazonaws.com

Wilayah AWS	Registri
us-gov-east-1	151742754352.dkr.ecr. us-gov-east-1.amazonaws.com
us-gov-west-1	013241004608.dkr.ecr. us-gov-west-1.amazonaws.com
us-west-1	602401143452.dkr. ecr.us-west-1.amazonaws.com
us-west-2	602401143452.dkr. ecr.us-west-2.amazonaws.com

Add-on Amazon EKS

Add-on adalah perangkat lunak yang menyediakan kemampuan operasional pendukung untuk Kubernetes aplikasi, tetapi tidak spesifik untuk aplikasi. Ini termasuk perangkat lunak seperti agen observabilitas atau Kubernetes driver yang memungkinkan cluster berinteraksi dengan AWS sumber daya yang mendasarinya untuk jaringan, komputasi, dan penyimpanan. Perangkat lunak add-on biasanya dibangun dan dikelola oleh Kubernetes komunitas, penyedia cloud seperti AWS, atau vendor pihak ketiga. Amazon EKS secara otomatis menginstal add-on yang dikelola sendiri seperti Amazon VPC CNI plugin for Kubernetes,`kube-proxy`, dan CoreDNS untuk setiap cluster. Anda dapat mengubah konfigurasi default add-on dan memperbaruinya bila diinginkan.

Add-on Amazon EKS menyediakan instalasi dan pengelolaan serangkaian add-on yang dikuratori untuk kluster Amazon EKS. Semua add-on Amazon EKS menyertakan patch keamanan terbaru, perbaikan bug, dan divalidasi oleh untuk AWS bekerja dengan Amazon EKS. Add-on Amazon EKS memungkinkan Anda untuk secara konsisten memastikan bahwa kluster Amazon EKS Anda aman dan stabil serta mengurangi jumlah pekerjaan yang perlu Anda lakukan untuk menginstal, mengonfigurasi, dan memperbarui add-on. Jika add-on yang dikelola sendiri, seperti `kube-proxy` sudah berjalan di cluster Anda dan tersedia sebagai add-on Amazon EKS, maka Anda dapat menginstal add-on `kube-proxy` Amazon EKS untuk mulai memanfaatkan kemampuan add-on Amazon EKS.

Anda dapat memperbarui bidang konfigurasi terkelola Amazon EKS tertentu untuk add-on Amazon EKS melalui Amazon EKS API. Anda juga dapat memodifikasi bidang konfigurasi yang tidak dikelola oleh Amazon EKS langsung di dalam Kubernetes klaster setelah add-on dimulai. Ini termasuk

mendefinisikan bidang konfigurasi tertentu untuk add-on jika berlaku. Perubahan ini tidak ditimpa oleh Amazon EKS setelah mereka dibuat. Ini dimungkinkan menggunakan fitur penerapan Kubernetes sisi server. Untuk informasi selengkapnya, lihat Manajemen lapangan Kubernetes.

Anda dapat menggunakan add-on Amazon EKS dengan jenis node Amazon EKS apa pun.

Pertimbangan

- Untuk mengonfigurasi add-on untuk cluster, prinsipal IAM Anda harus memiliki izin IAM untuk bekerja dengan add-on. Untuk informasi selengkapnya, lihat tindakan dengan namanya `Addon` dalam Tindakan yang ditentukan oleh Amazon Elastic Kubernetes Service.
- Add-on Amazon EKS berjalan pada simpul yang Anda sediakan atau konfigurasikan untuk klaster Anda. Jenis simpul mencakup instans Amazon EC2 dan Fargate.
- Anda dapat memodifikasi bidang yang tidak dikelola oleh Amazon EKS untuk menyesuaikan penginstalan add-on Amazon EKS. Untuk informasi selengkapnya, lihat Manajemen lapangan Kubernetes.
- Jika Anda membuat klaster dengan add-on AWS Management Console, Amazon EKS `kube-proxy` Amazon VPC CNI plugin for Kubernetes, dan CoreDNS Amazon EKS secara otomatis ditambahkan ke cluster Anda. Jika Anda gunakan `eksctl` untuk membuat cluster Anda dengan `config` file, juga `eksctl` dapat membuat cluster dengan Amazon EKS add-on. Jika Anda membuat klaster Anda menggunakan `eksctl` tanpa `config` file atau dengan alat lain, yang dikelola sendiri `kube-proxy` Amazon VPC CNI plugin for Kubernetes, dan CoreDNS add-on diinstal, bukan add-on Amazon EKS. Anda dapat mengelolanya sendiri atau menambahkan add-on Amazon EKS secara manual setelah pembuatan cluster.
- `eks:addon-cluster-admin`Itu `ClusterRoleBinding` mengikat `eks:addon-manager` Kubernetes identitas. `cluster-admin` `ClusterRole` Peran tersebut memiliki izin yang diperlukan untuk `eks:addon-manager` identitas untuk membuat Kubernetes ruang nama dan menginstal add-on ke ruang nama. Jika `eks:addon-cluster-admin` `ClusterRoleBinding` dihapus, kluster Amazon EKS akan terus berfungsi, namun Amazon EKS tidak lagi dapat mengelola add-on apa pun. Semua cluster yang dimulai dengan versi platform berikut menggunakan yang baru`ClusterRoleBinding`.

Versi Kubernetes	Platform EKS
1.29	eks.2
1.28	eks.12
1.27	eks.14
1.26	eks.15
1.25	eks.16
1.24	eks.17

Anda dapat menambahkan, memperbarui, atau menghapus add-on Amazon EKS menggunakan Amazon EKS API,, AWS Management Console AWS CLI, dan `eksctl`. Untuk informasi selengkapnya, lihat Mengelola add-on Amazon EKS. Anda juga dapat membuat Amazon EKS add-on menggunakan AWS CloudFormation.

Pengaya Amazon EKS yang tersedia dari Amazon EKS

Add-on Amazon EKS berikut tersedia untuk dibuat di cluster Anda. Anda selalu dapat melihat daftar terbaru dari add-on yang tersedia menggunakan`eksctl`, AWS Management Console, atau. AWS CLI Untuk melihat semua add-on yang tersedia atau untuk menginstal add-on, lihat. Membuat add-on Jika add-on memerlukan izin IAM, maka Anda harus memiliki penyedia IAM OpenID Connect (OIDC) untuk klaster Anda. Untuk menentukan apakah Anda memilikinya, atau membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda. Anda dapat memperbarui atau menghapus add-on setelah Anda menginstalnya.

Pilih add-on untuk mempelajari lebih lanjut tentang hal itu dan persyaratan pemasangannya.

Amazon VPC CNI plugin for Kubernetes

- Nama – `vpc-cni`
- Deskripsi - Plugin antarmuka jaringan Kubernetes kontainer (CNI) yang menyediakan jaringan VPC asli untuk cluster Anda. Jenis add-on yang dikelola sendiri atau dikelola ini diinstal pada setiap node Amazon EC2, secara default.

- Izin IAM yang diperlukan - Pengaya ini menggunakan peran IAM untuk kemampuan akun layanan Amazon EKS. Jika klaster Anda menggunakan `IPv4` keluarga, izin di AmazonEks_CNI_Policy diperlukan. Jika klaster Anda menggunakan `IPv6` keluarga, Anda harus membuat kebijakan IAM dengan izin dalam mode IPv6. Anda dapat membuat peran IAM, melampirkan salah satu kebijakan padanya, dan membuat anotasi akun Kubernetes layanan yang digunakan oleh add-on dengan perintah berikut.

  Ganti *`my-cluster`* dengan nama cluster Anda dan *`AmazonEKSVPCCNIRole`* dengan nama untuk peran Anda. Jika klaster Anda menggunakan `IPv6` keluarga, ganti *`AmazonEKS_CNI_Policy`* dengan nama kebijakan yang Anda buat. Perintah ini mengharuskan Anda telah `eksctl`menginstal pada perangkat Anda. Jika Anda perlu menggunakan alat lain untuk membuat peran, lampirkan kebijakan padanya, dan beri anotasi pada akun Kubernetes layanan, lihat. Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM

  ```
  eksctl create iamserviceaccount --name aws-node --namespace kube-system --cluster my-cluster --role-name AmazonEKSVPCCNIRole \
      --role-only --attach-policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy --approve
  ```

- Informasi tambahan — Untuk mempelajari lebih lanjut tentang pengaturan add-on yang dapat dikonfigurasi, lihat aws-vpc-cni-k8s aktif. GitHub Untuk mempelajari lebih lanjut tentang plugin, lihat Proposal: Plugin CNI untuk Kubernetes jaringan melalui AWS VPC. Untuk informasi selengkapnya tentang membuat add-on, lihatMembuat add-on Amazon EKS.
- Perbarui informasi - Anda hanya dapat memperbarui satu versi minor pada satu waktu. Misalnya, jika versi Anda saat ini `1.27.`*`x`*`-eksbuild.`*`y`* dan Anda ingin memperbaruinya`1.29.`*`x`*`-eksbuild.`*`y`*, maka Anda harus memperbarui versi Anda saat ini `1.28.`*`x`*`-eksbuild.`*`y`* dan kemudian memperbaruinya lagi ke`1.29.`*`x`*`-eksbuild.`*`y`*. Untuk informasi selengkapnya tentang memperbarui add-on, lihatMemperbarui add-on Amazon EKS.

CoreDNS

- Nama – `coredns`
- Deskripsi — Server DNS yang fleksibel dan dapat diperluas yang dapat berfungsi sebagai DNS Kubernetes cluster. Jenis pengaya ini yang dikelola sendiri atau dikelola telah diinstal, secara default, saat Anda membuat klaster. Saat Anda meluncurkan klaster Amazon EKS dengan setidaknya satu node, dua replika CoreDNS gambar akan di-deploy secara default, terlepas dari jumlah node yang digunakan di cluster Anda. CoreDNSPodsMenyediakan resolusi nama untuk

semua Pods di cluster. Anda dapat menerapkan node CoreDNS Pods ke Fargate jika klaster Anda menyertakan dengan namespace AWS Fargate profil yang cocok dengan namespace untuk. CoreDNS `deployment`

- Izin IAM yang diperlukan - Pengaya ini tidak memerlukan izin apa pun.
- Informasi tambahan — Untuk mempelajari lebih lanjut tentang CoreDNS, lihat Menggunakan CoreDNS untuk Penemuan Layanan dan Menyesuaikan Layanan DNS dalam dokumentasi. Kubernetes

`Kube-proxy`

- Nama – `kube-proxy`
- Deskripsi - Mempertahankan aturan jaringan pada setiap node Amazon EC2. Ini memungkinkan komunikasi jaringan ke AndaPods. Jenis add-on yang dikelola sendiri atau dikelola ini diinstal pada setiap node Amazon EC2 di klaster Anda, secara default.
- Izin IAM yang diperlukan - Pengaya ini tidak memerlukan izin apa pun.
- Informasi tambahan — Untuk mempelajari selengkapnya`kube-proxy`, lihat `kube-proxy`di Kubernetes dokumentasi.
- Perbarui informasi - Sebelum memperbarui versi Anda saat ini, pertimbangkan persyaratan berikut:
  - `Kube-proxy`pada kluster Amazon EKS memiliki kompatibilitas dan kebijakan miring yang sama dengan. Kubernetes
  - `Kube-proxy`harus versi minor yang sama seperti `kubelet` pada node Amazon EC2 Anda.
  - `Kube-proxy`tidak bisa lebih lambat dari versi minor dari bidang kontrol cluster Anda.
  - `kube-proxy`Versi pada node Amazon EC2 Anda tidak boleh lebih dari dua versi minor lebih awal dari bidang kontrol Anda. Misalnya, jika control plane Anda menjalankan Kubernetes 1.29, maka versi `kube-proxy` minor tidak boleh lebih awal dari 1.27.
  - Jika Anda baru saja memperbarui klaster ke versi Kubernetes minor baru, perbarui node Amazon EC2 Anda ke versi minor yang sama sebelum memperbarui `kube-proxy` ke versi minor yang sama dengan node Anda.

Driver CSI Amazon EBS

- Nama – `aws-ebs-csi-driver`
- Deskripsi - Plugin Kubernetes Container Storage Interface (CSI) yang menyediakan penyimpanan Amazon EBS untuk cluster Anda.

- Izin IAM yang diperlukan - Pengaya ini menggunakan peran IAM untuk kemampuan akun layanan Amazon EKS. Izin dalam kebijakan `AmazonEBSCSIDriverPolicy` AWS terkelola diperlukan. Anda dapat membuat peran IAM dan melampirkan kebijakan terkelola padanya dengan perintah berikut. Ganti *`my-cluster`* dengan nama cluster Anda dan *`AmazonEKS_EBS_CSI_DriverRole`* dengan nama untuk peran Anda. Perintah ini mengharuskan Anda telah `eksctl`menginstal pada perangkat Anda. Jika Anda perlu menggunakan alat yang berbeda atau Anda perlu menggunakan kunci KMS khusus untuk enkripsi, lihatMembuat peran IAM driver Amazon EBS CSI.

```
eksctl create iamserviceaccount \
    --name ebs-csi-controller-sa \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKS_EBS_CSI_DriverRole \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy
 \
    --approve
```

- Informasi tambahan — Untuk mempelajari lebih lanjut tentang add-on, lihatDriver CSI Amazon EBS.

Driver Amazon EFS CSI

- Nama – `aws-efs-csi-driver`
- Deskripsi - Plugin Kubernetes Container Storage Interface (CSI) yang menyediakan penyimpanan Amazon EFS untuk cluster Anda.
- Izin IAM yang diperlukan - Pengaya ini menggunakan peran IAM untuk kemampuan akun layanan Amazon EKS. Izin dalam kebijakan `AmazonEFSCSIDriverPolicy` AWS terkelola diperlukan. Anda dapat membuat peran IAM dan melampirkan kebijakan terkelola padanya dengan perintah berikut. Ganti *`my-cluster`* dengan nama cluster Anda dan *`AmazonEKS_EFS_CSI_DriverRole`* dengan nama untuk peran Anda. Perintah ini mengharuskan Anda `eksctl`menginstal pada perangkat Anda. Jika Anda perlu menggunakan alat yang berbeda, lihatMembuat peran IAM.

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
eksctl create iamserviceaccount \
    --name efs-csi-controller-sa \
    --namespace kube-system \
    --cluster $cluster_name \
```

```
    --role-name $role_name \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy
 \
    --approve
TRUST_POLICY=$(aws iam get-role --role-name $role_name --query
 'Role.AssumeRolePolicyDocument' | \
    sed -e 's/efs-csi-controller-sa/efs-csi-*/' -e 's/StringEquals/StringLike/')
aws iam update-assume-role-policy --role-name $role_name --policy-document
 "$TRUST_POLICY"
```

- Informasi tambahan — Untuk mempelajari lebih lanjut tentang add-on, lihatDriver Amazon EFS CSI.

Mountpointuntuk Driver Amazon S3 CSI

- Nama – `aws-mountpoint-s3-csi-driver`
- Deskripsi - Plugin Kubernetes Container Storage Interface (CSI) yang menyediakan penyimpanan Amazon S3 untuk cluster Anda.
- Izin IAM yang diperlukan - Pengaya ini menggunakan peran IAM untuk kemampuan akun layanan Amazon EKS. Peran IAM yang dibuat akan memerlukan kebijakan yang memberikan akses ke S3. Ikuti rekomendasi izin Mountpoint IAM saat membuat kebijakan. Atau, Anda dapat menggunakan kebijakan AWS terkelola AmazonS3FullAccess, tetapi kebijakan terkelola ini memberikan lebih banyak izin daripada yang diperlukan. Mountpoint

  Anda dapat membuat peran IAM dan melampirkan kebijakan Anda padanya dengan perintah berikut. *Ganti cluster saya dengan nama cluster Anda, kode wilayah dengan kode yang benar, AmazonEKS_S3_CSI_ dengan nama untuk Wilayah AWS peran Anda, dan AmazonEKS_S3_CSI_ _ARN DriverRole dengan peran ARN. DriverRole* Perintah ini mengharuskan Anda `eksctl`menginstal pada perangkat Anda. Untuk petunjuk tentang penggunaan konsol IAM atau AWS CLI, lihatMembuat peran IAM.

```
CLUSTER_NAME=my-cluster
REGION=region-code
ROLE_NAME=AmazonEKS_S3_CSI_DriverRole
POLICY_ARN=AmazonEKS_S3_CSI_DriverRole_ARN
eksctl create iamserviceaccount \
    --name s3-csi-driver-sa \
    --namespace kube-system \
    --cluster $CLUSTER_NAME \
```

```
    --attach-policy-arn $POLICY_ARN \
    --approve \
    --role-name $ROLE_NAME \
    --region $REGION \
    --role-only
```

- Informasi tambahan — Untuk mempelajari lebih lanjut tentang add-on, lihatMountpointuntuk driver Amazon S3 CSI.

Pengontrol snapshot CSI

- Nama – `snapshot-controller`
- Deskripsi - Pengontrol snapshot Container Storage Interface (CSI) memungkinkan penggunaan fungsionalitas snapshot di driver CSI yang kompatibel, seperti driver Amazon EBS CSI.
- Izin IAM yang diperlukan - Pengaya ini tidak memerlukan izin apa pun.
- Informasi tambahan — Untuk mempelajari lebih lanjut tentang add-on, lihatPengontrol snapshot CSI.

AWS Distro untuk OpenTelemetry

- Nama – `adot`
- Deskripsi — AWS Distro for OpenTelemetry (ADOT) adalah distribusi proyek yang aman, siap produksi, dan AWS didukung. OpenTelemetry
- Izin IAM yang diperlukan - Pengaya ini hanya memerlukan izin IAM jika Anda menggunakan salah satu sumber daya kustom yang telah dikonfigurasi sebelumnya yang dapat dipilih melalui konfigurasi lanjutan.
- Informasi tambahan — Untuk informasi selengkapnya, lihat Memulai dengan AWS Distro untuk OpenTelemetry menggunakan Eks Add-On di AWS Distro untuk dokumentasi. OpenTelemetry

  ADOT mengharuskan itu `cert-manager` diterapkan di cluster sebagai prasyarat, jika tidak, add-on ini tidak akan berfungsi jika diterapkan secara langsung menggunakan properti Amazon EKS Terraform. `cluster_addons` Untuk persyaratan lebih lanjut, lihat Persyaratan untuk Memulai dengan AWS Distro untuk OpenTelemetry menggunakan Eks Add-On di AWS Distro untuk dokumentasi. OpenTelemetry

Amazon GuardDuty agen

- Nama – `aws-guardduty-agent`
- Deskripsi - Amazon GuardDuty adalah layanan pemantauan keamanan yang menganalisis dan memproses sumber data dasar termasuk peristiwa AWS CloudTrail manajemen dan log aliran VPC Amazon. Amazon GuardDuty juga memproses fitur, seperti log Kubernetes audit dan pemantauan runtime.
- Izin IAM yang diperlukan - Pengaya ini tidak memerlukan izin apa pun.
- Informasi tambahan — Untuk informasi selengkapnya, lihat Runtime Monitoring untuk klaster Amazon EKS di Amazon. GuardDuty
  - Untuk mendeteksi potensi ancaman keamanan di kluster Amazon EKS Anda, aktifkan pemantauan GuardDuty runtime Amazon dan terapkan agen GuardDuty keamanan ke kluster Amazon EKS Anda.

Agen CloudWatch Observabilitas Amazon

- Nama – `amazon-cloudwatch-observability`
- Deskripsi Amazon CloudWatch Agent adalah layanan pemantauan dan observabilitas yang disediakan oleh AWS. Add-on ini menginstal CloudWatch Agen dan mengaktifkan Sinyal CloudWatch Aplikasi dan CloudWatch Wawasan Kontainer dengan peningkatan observabilitas untuk Amazon EKS.
- Izin IAM yang diperlukan - Pengaya ini menggunakan peran IAM untuk kemampuan akun layanan Amazon EKS. Izin dalam AWSXrayWriteOnlyAccessdan kebijakan CloudWatchAgentServerPolicy AWS terkelola diperlukan. Anda dapat membuat peran IAM, melampirkan kebijakan terkelola padanya, dan membuat anotasi akun Kubernetes layanan yang digunakan oleh add-on dengan perintah berikut. Ganti *`my-cluster`* dengan nama cluster Anda dan *`AmazonEKS_Observability_role`* dengan nama untuk peran Anda. Perintah ini mengharuskan Anda telah `eksctl`menginstal pada perangkat Anda. Jika Anda perlu menggunakan alat lain untuk membuat peran, lampirkan kebijakan padanya, dan beri anotasi pada akun Kubernetes layanan, lihat. Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM

```
eksctl create iamserviceaccount \
    --name cloudwatch-agent \
    --namespace amazon-cloudwatch \
    --cluster my-cluster \
    --role-name AmazonEKS_Observability_Role \
    --role-only \
```

```
    --attach-policy-arn arn:aws:iam::aws:policy/AWSXrayWriteOnlyAccess \
    --attach-policy-arn arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy \
    --approve
```

- Informasi tambahan — Untuk informasi selengkapnya, lihat Menginstal CloudWatch agen.

Agen Identitas Amazon EKS Pod

- Nama – `eks-pod-identity-agent`
- Deskripsi — Amazon EKS Pod Identity menyediakan kemampuan untuk mengelola kredensyal untuk aplikasi Anda, mirip dengan cara profil Amazon EC2 instans memberikan kredensyal ke instans EC2.
- Izin IAM yang diperlukan - Izin pengguna add-on ini dari. IAM role simpul Amazon EKS
- Perbarui informasi - Anda hanya dapat memperbarui satu versi minor pada satu waktu. Misalnya, jika versi Anda saat ini `1.27.`*x*`-eksbuild.`*y* dan Anda ingin memperbaruinya`1.29.`*x*`-eksbuild.`*y*, maka Anda harus memperbarui versi Anda saat ini `1.28.`*x*`-eksbuild.`*y* dan kemudian memperbaruinya lagi ke`1.29.`*x*`-eksbuild.`*y*. Untuk informasi selengkapnya tentang memperbarui add-on, lihatMemperbarui add-on Amazon EKS.

Add-on Amazon EKS tambahan dari vendor perangkat lunak independen

Selain daftar add-on Amazon EKS sebelumnya, Anda juga dapat menambahkan berbagai pilihan perangkat lunak operasional Amazon EKS add-on dari vendor perangkat lunak independen. Pilih add-on untuk mempelajari lebih lanjut tentang hal itu dan persyaratan pemasangannya.

Temukan, dapatkan, dan terapkan add-on dari AWS Marketplace ke Amazon EKS (). YouTube

Accuknox

- Penerbit - Accuknox
- Nama – `accuknox_kubearmor`
- Ruang nama — `kubearmor`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Memulai dengan KubeArmor dalam KubeArmor dokumentasi.

NetApp

- Penerbit - NetApp
- Nama – `netapp_trident-operator`
- Ruang nama — `trident`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Mengonfigurasi add-on Astra Trident EKS di dokumentasi. NetApp

Calyptia

- Penerbit - Calyptia
- Nama – `calyptia_fluent-bit`
- Ruang nama — `calytia-fluentbit`
- Nama akun layanan - `clyptia-fluentbit`
- AWS kebijakan IAM terkelola — AWSMarketplaceMeteringRegisterUsage.
- Perintah untuk membuat peran IAM yang diperlukan - Perintah berikut mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk cluster Anda. Untuk menentukan apakah Anda memilikinya, atau membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda. Ganti *my-cluster* dengan nama cluster Anda dan *my-calyptia-role* dengan nama untuk peran Anda. Perintah ini mengharuskan Anda telah `eksctl`menginstal pada perangkat Anda. Jika Anda perlu menggunakan alat yang berbeda untuk membuat peran dan membuat anotasi akun Kubernetes layanan, lihat. Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM

```
eksctl create iamserviceaccount --name service-account-name  --namespace calyptia-
fluentbit --cluster my-cluster --role-name my-calyptia-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/
AWSMarketplaceMeteringRegisterUsage --approve
```

- Petunjuk penyiapan dan penggunaan — Lihat Calyptia for Fluent Bit dalam dokumentasi Calyptia.

Cribl

- Penerbit - Cribl

- Nama – `cribl_cribledge`
- Ruang nama — `cribledge`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Menginstal Add-on Cribl Amazon EKS untuk Edge di dokumentasi Cribl.

Dynatrace

- Penerbit - Dynatrace
- Nama – `dynatrace_dynatrace-operator`
- Ruang nama — `dynatrace`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat pemantauan Kubernetes di dokumentasi. dynatrace

Datree

- Penerbit - Datree
- Nama – `datree_engine-pro`
- Ruang nama — `datree`
- Nama akun layanan - datree-webhook-server-awsmp
- AWS kebijakan IAM terkelola — AWSLicenseManagerConsumptionPolicy.
- Perintah untuk membuat peran IAM yang diperlukan - Perintah berikut mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk cluster Anda. Untuk menentukan apakah Anda memilikinya, atau membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda. Ganti *`my-cluster`* dengan nama cluster Anda dan *`my-datree-role`* dengan nama untuk peran Anda. Perintah ini mengharuskan Anda telah `eksctl`menginstal pada perangkat Anda. Jika Anda perlu menggunakan alat yang berbeda untuk membuat peran dan membuat anotasi akun Kubernetes layanan, lihat. Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM

```
eksctl create iamserviceaccount --name datree-webhook-server-awsmp --namespace datree
 --cluster my-cluster --role-name my-datree-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Amazon EKS-Intergrasi di dokumentasi Datree.

Datadog

- Penerbit - Datadog
- Nama – `datadog_operator`
- Ruang nama — `datadog-agent`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Menginstal Agen Datadog di Amazon EKS dengan Add-on Operator Datadog di dokumentasi Datadog.

Groundcover

- Penerbit - groundcover
- Nama – `groundcover_agent`
- Ruang nama — `groundcover`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Menginstal Add-on Amazon EKS penutup tanah di dokumentasi penutup tanah.

Grafana Labs

- Penerbit - Grafana Labs
- Nama – `grafana-labs_kubernetes-monitoring`

- Ruang nama — `monitoring`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Mengonfigurasi Pemantauan Kubernetes sebagai Add-on dengan Amazon EKS di dokumentasi Grafana Labs.

HA Proxy

- Penerbit - HA Proxy
- Nama – `haproxy-technologies_kubernetes-ingress-ee`
- Ruang nama — `haproxy-controller`
- Nama akun layanan - `customer defined`
- AWS kebijakan IAM terkelola — AWSLicenseManagerConsumptionPolicy.
- Perintah untuk membuat peran IAM yang diperlukan - Perintah berikut mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk cluster Anda. Untuk menentukan apakah Anda memilikinya, atau membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda. Ganti *my-cluster* dengan nama cluster Anda dan *my-haproxy-role* dengan nama untuk peran Anda. Perintah ini mengharuskan Anda telah `eksctl`menginstal pada perangkat Anda. Jika Anda perlu menggunakan alat yang berbeda untuk membuat peran dan membuat anotasi akun Kubernetes layanan, lihat. Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM

```
eksctl create iamserviceaccount --name service-account-name  --namespace haproxy-
controller --cluster my-cluster --role-name my-haproxy-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Instal HAProxy Enterprise Kubernetes Ingress Controller di Amazon EKS dari AWS dalam dokumentasi HAProxy.

Kpow

- Penerbit - Factorhouse
- Nama – `factorhouse_kpow`

- Ruang nama — `factorhouse`
- Nama akun layanan - `kpow`
- AWS kebijakan IAM terkelola - AWSLicenseManagerConsumptionPolicy
- Perintah untuk membuat peran IAM yang diperlukan - Perintah berikut mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk cluster Anda. Untuk menentukan apakah Anda memilikinya, atau membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda. Ganti *`my-cluster`* dengan nama cluster Anda dan *`my-kpow-role`* dengan nama untuk peran Anda. Perintah ini mengharuskan Anda telah `eksctl`menginstal pada perangkat Anda. Jika Anda perlu menggunakan alat yang berbeda untuk membuat peran dan membuat anotasi akun Kubernetes layanan, lihat. Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM

```
eksctl create iamserviceaccount --name kpow --namespace factorhouse --cluster my-
cluster --role-name my-kpow-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk pengaturan dan penggunaan — Lihat AWS Marketplace LM dalam Kpow dokumentasi.

Kubecost

- Penerbit - Kubecost
- Nama – `kubecost_kubecost`
- Ruang nama — `kubecost`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Integrasi Penagihan AWS Cloud di Kubecost dokumentasi.
- Jika cluster Anda versi `1.23` atau lebih baru, Anda harus the section called “Driver CSI Amazon EBS” menginstal di cluster Anda. jika tidak, Anda akan menerima kesalahan.

Kasten

- Penerbit - Kasten by Veeam

- Nama – `kasten_k10`
- Ruang nama — `kasten-io`
- Nama akun layanan - `k10-k10`
- AWS kebijakan IAM terkelola — AWSLicenseManagerConsumptionPolicy.
- Perintah untuk membuat peran IAM yang diperlukan - Perintah berikut mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk cluster Anda. Untuk menentukan apakah Anda memilikinya, atau membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda. Ganti *my-cluster* dengan nama cluster Anda dan *my-kasten-role* dengan nama untuk peran Anda. Perintah ini mengharuskan Anda telah `eksctl`menginstal pada perangkat Anda. Jika Anda perlu menggunakan alat yang berbeda untuk membuat peran dan membuat anotasi akun Kubernetes layanan, lihat. Mengkonfigurasi akun Kubernetes layanan untuk mengambil peran IAM

```
eksctl create iamserviceaccount --name k10-k10 --namespace kasten-io --cluster my-
cluster --role-name my-kasten-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Menginstal K10 saat AWS menggunakan Amazon EKS Add-on di dokumentasi Kasten.
- Informasi tambahan - Jika kluster Amazon EKS Anda versi Kubernetes `1.23` atau yang lebih baru, Anda harus menginstal driver Amazon EBS CSI di cluster Anda dengan default. `StorageClass`

Kong

- Penerbit - Kong
- Nama – `kong_konnect-ri`
- Ruang nama — `kong`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Menginstal Add-on Kong Gateway EKS di dokumentasi Kong.

LeakSignal

- Penerbit - LeakSignal
- Nama – `leaksignal_leakagent`
- Ruang nama — `leakagent`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Instal LeakAgent add-on dalam LeakSignal dokumentasi.

New Relic

- Penerbit - New Relic
- Nama – `new-relic_kubernetes-operator`
- Ruang nama — `newrelic`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Menginstal Add-on Relik Baru untuk EKS di dokumentasi New Relic.

Rafay

- Penerbit - Rafay
- Nama – `rafay-systems_rafay-operator`
- Ruang nama — `rafay-system`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Menginstal Add-on Rafay Amazon EKS di dokumentasi Rafay.

Solo.io

- Penerbit - Solo.io
- Nama – `solo-io_istio-distro`
- Ruang nama — `istio-system`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Menginstal Istio di dokumentasi Solo.io.

Stormforge

- Penerbit - Stormforge
- Nama – `stormforge_optimize-Live`
- Ruang nama — `stormforge-system`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Menginstal StormForge Agen dalam StormForge dokumentasi.

Splunk

- Penerbit - Splunk
- Nama – `splunk_splunk-otel-collector-chart`
- Ruang nama — `splunk-monitoring`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Instal add-on Splunk untuk Amazon EKS di dokumentasi Splunk.

Teleport

- Penerbit - Teleport
- Nama – `teleport_teleport`
- Ruang nama — `teleport`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk penyiapan dan penggunaan — Lihat Cara Kerja Teleport dalam Teleport dokumentasi.

Tetrate

- Penerbit — Tetrate Io
- Nama – `tetrate-io_istio-distro`
- Ruang nama — `istio-system`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk pengaturan dan penggunaan — Lihat situs web Tetrate Istio Distro.

Upbound Universal Crossplane

- Penerbit - Upbound
- Nama – `upbound_universal-crossplane`
- Ruang nama — `upbound-system`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk pengaturan dan penggunaan - Lihat Upbound Universal Crossplane (UXP) di dokumentasi Upbound.

Upwind

- Penerbit - Upwind
- Nama – `upwind`
- Ruang nama — `upwind`
- Nama akun layanan — Akun layanan tidak digunakan dengan add-on ini.
- AWS kebijakan IAM terkelola — Kebijakan terkelola tidak digunakan dengan add-on ini.
- Izin IAM khusus - Izin khusus tidak digunakan dengan add-on ini.
- Petunjuk pengaturan dan penggunaan - Lihat langkah-langkah instalasi dalam dokumentasi Upwind.

Mengelola add-on Amazon EKS

Pengaya Amazon EKS adalah seperangkat perangkat lunak add-on yang dikuratori untuk kluster Amazon EKS. Semua pengaya Amazon EKS:

- termasuk patch keamanan terbaru dan perbaikan bug.
- divalidasi oleh AWS untuk bekerja dengan Amazon EKS.
- mengurangi jumlah pekerjaan yang diperlukan untuk mengelola perangkat lunak add-on.

AWS Management Console Ini memberi tahu Anda ketika versi baru tersedia untuk add-on Amazon EKS. Anda cukup memulai pembaruan, dan Amazon EKS memperbarui perangkat lunak add-on untuk Anda.

Untuk daftar add-on yang tersedia, lihatPengaya Amazon EKS yang tersedia dari Amazon EKS. Untuk informasi selengkapnya tentang manajemen Kubernetes lapangan, lihat Manajemen lapangan Kubernetes

Prasyarat

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS.
- Jika Anda membuat add-on yang menggunakan akun Kubernetes layanan dan peran IAM, maka Anda harus memiliki penyedia AWS Identity and Access Management (IAM) OpenID Connect (OIDC) untuk klaster Anda. Untuk menentukan apakah Anda memiliki satu untuk cluster Anda, atau untuk membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda.

Membuat add-on

Anda dapat membuat add-on Amazon EKS menggunakan`eksctl`, AWS Management Console, atau. AWS CLI Jika add-on memerlukan peran IAM, lihat detail untuk add-on spesifik Pengaya Amazon EKS yang tersedia dari Amazon EKS untuk detail tentang membuat peran.

eksctl

Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Untuk membuat add-on Amazon EKS menggunakan **`eksctl`**

1. Lihat nama add-on yang tersedia untuk versi cluster. Ganti *`1.29`* dengan versi cluster Anda.

   ```
   eksctl utils describe-addon-versions --kubernetes-version 1.29 | grep AddonName
   ```

   Contoh output adalah sebagai berikut.

   ```
   "AddonName": "aws-ebs-csi-driver",
                           "AddonName": "coredns",
                           "AddonName": "kube-proxy",
                           "AddonName": "vpc-cni",
                           "AddonName": "adot",
                           "AddonName": "dynatrace_dynatrace-operator",
                           "AddonName": "upbound_universal-crossplane",
                           "AddonName": "teleport_teleport",
                           "AddonName": "factorhouse_kpow",
                           [...]
   ```

2. Lihat versi yang tersedia untuk add-on yang ingin Anda buat. Ganti *`1.29`* dengan versi cluster Anda. Ganti *`name-of-addon`* dengan nama add-on yang ingin Anda lihat versinya. Nama harus salah satu nama yang dikembalikan pada langkah sebelumnya.

   ```
   eksctl utils describe-addon-versions --kubernetes-version 1.29 --name name-of-addon | grep AddonVersion
   ```

Output berikut adalah contoh dari apa yang dikembalikan untuk add-on bernama`vpc-cni`. Anda dapat melihat bahwa add-on memiliki beberapa versi yang tersedia.

```
"AddonVersions": [
    "AddonVersion": "v1.12.0-eksbuild.1",
    "AddonVersion": "v1.11.4-eksbuild.1",
    "AddonVersion": "v1.10.4-eksbuild.1",
    "AddonVersion": "v1.9.3-eksbuild.1",
```

3. Tentukan apakah add-on yang ingin Anda buat adalah Amazon EKS atau AWS Marketplace add-on. Ini AWS Marketplace memiliki add-on pihak ketiga yang mengharuskan Anda menyelesaikan langkah-langkah tambahan untuk membuat add-on.

```
eksctl utils describe-addon-versions --kubernetes-version 1.29 --name name-of-addon | grep ProductUrl
```

Jika tidak ada output yang dikembalikan, maka add-on adalah Amazon EKS. Jika output dikembalikan, maka add-on adalah AWS Marketplace add-on. Output berikut adalah untuk add-on bernama`teleport_teleport`.

```
"ProductUrl": "https://aws.amazon.com/marketplace/pp?
sku=3bda70bb-566f-4976-806c-f96faef18b26"
```

Anda dapat mempelajari lebih lanjut tentang add-on di AWS Marketplace dengan URL yang dikembalikan. Jika add-on memerlukan langganan, Anda dapat berlangganan add-on melalui. AWS Marketplace Jika Anda akan membuat add-on dari AWS Marketplace, maka prinsipal IAM yang Anda gunakan untuk membuat add-on harus memiliki izin untuk membuat peran terkait layanan. AWSServiceRoleForAWSLicenseManagerRole Untuk informasi selengkapnya tentang menetapkan izin ke entitas IAM, lihat Menambahkan dan menghapus izin identitas IAM di Panduan Pengguna IAM.

4. Buat add-on Amazon EKS. Salin perintah yang mengikuti ke perangkat Anda. Buat modifikasi berikut pada perintah sesuai kebutuhan dan kemudian jalankan perintah yang dimodifikasi:

- Ganti *my-cluster* dengan nama klaster Anda.
- Ganti *name-of-addon* dengan nama add-on yang ingin Anda buat.

- Jika Anda menginginkan versi add-on yang lebih awal dari versi terbaru, maka ganti *latest* dengan nomor versi yang dikembalikan dalam output dari langkah sebelumnya yang ingin Anda gunakan.
- Jika add-on menggunakan peran akun layanan, ganti *111122223333* dengan ID akun Anda dan ganti *role-name* dengan nama peran. Untuk petunjuk cara membuat peran untuk akun layanan Anda, lihat dokumentasi untuk add-on yang Anda buat. Menentukan peran akun layanan mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk klaster Anda. Untuk menentukan apakah Anda memiliki satu untuk cluster Anda, atau untuk membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda.

  Jika add-on tidak menggunakan peran akun layanan, hapus***--service-account-role-arn*** **arn:aws:iam::*111122223333*:role/*role-name***.
- Perintah contoh ini menimpa konfigurasi versi add-on yang dikelola sendiri yang ada, jika ada. Jika Anda tidak ingin menimpa konfigurasi add-on yang dikelola sendiri, hapus opsi. *--force* Jika Anda menghapus opsi, dan add-on Amazon EKS perlu menimpa konfigurasi add-on yang dikelola sendiri, maka pembuatan add-on Amazon EKS gagal dengan pesan kesalahan untuk membantu Anda menyelesaikan konflik. Sebelum menentukan opsi ini, pastikan add-on Amazon EKS tidak mengelola pengaturan yang perlu Anda kelola, karena pengaturan tersebut ditimpa dengan opsi ini.

  ```
  eksctl create addon --cluster my-cluster --name name-of-addon --version latest
   \
      --service-account-role-arn arn:aws:iam::111122223333:role/role-name --
  force
  ```

Anda dapat melihat daftar semua opsi yang tersedia untuk perintah.

```
eksctl create addon --help
```

Untuk informasi selengkapnya tentang opsi yang tersedia, lihat Addons dalam `eksctl` dokumentasi.

AWS Management Console

Untuk membuat add-on Amazon EKS menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel navigasi kiri, pilih Clusters, lalu pilih nama cluster yang ingin Anda buat add-on.

3. Pilih tab Add-ons.
4. Pilih Get more add-ons
5. Pilih add-on yang ingin Anda tambahkan ke cluster Anda. Anda dapat menambahkan add-on dan AWS Marketplace add-on Amazon EKS sebanyak yang Anda butuhkan.

   Untuk AWS Marketplaceadd-on, prinsipal IAM yang Anda gunakan untuk membuat add-on harus memiliki izin untuk membaca hak untuk add-on dari. AWS LicenseManager AWS LicenseManager memerlukan peran AWSServiceRoleForAWSLicenseManagerRoleterkait layanan (SLR) yang memungkinkan AWS sumber daya mengelola lisensi atas nama Anda. SLR adalah persyaratan satu kali, per akun, dan Anda tidak perlu membuat SLR terpisah untuk setiap add-on atau setiap cluster. Untuk informasi selengkapnya tentang menetapkan izin ke prinsipal IAM, lihat Menambahkan dan menghapus izin identitas IAM di Panduan Pengguna IAM.

   Jika AWS Marketplace add-on yang ingin Anda instal tidak terdaftar, Anda dapat mencari add-on yang tersedia dengan memasukkan teks di kotak pencarian. Dalam opsi Pemfilteran, Anda juga dapat memfilter berdasarkan kategori, vendor, atau model harga dan kemudian memilih add-on dari hasil pencarian. Setelah Anda memilih add-on yang ingin Anda instal, pilih Berikutnya.
6. Pada halaman Konfigurasi pengaturan add-on yang dipilih:

   - Pilih Lihat opsi berlangganan untuk membuka formulir Opsi berlangganan. Tinjau detail Harga dan bagian Hukum, lalu pilih tombol Berlangganan untuk melanjutkan.
   - Untuk Versi, pilih versi yang ingin Anda instal. Kami merekomendasikan versi yang ditandai terbaru, kecuali setiap add-on yang Anda buat merekomendasikan versi yang berbeda. Untuk menentukan apakah add-on memiliki versi yang direkomendasikan, lihat dokumentasi untuk add-on yang Anda buat.
   - Jika semua add-on yang Anda pilih memiliki Memerlukan langganan di bawah Status, pilih Berikutnya. Anda tidak dapat mengonfigurasi add-on tersebut lebih lanjut sampai Anda berlangganan mereka setelah klaster Anda dibuat. Untuk add-on yang tidak memiliki langganan Memerlukan di bawah Status:
     - Untuk peran Select IAM, terima opsi default, kecuali add-on memerlukan izin IAM. Jika add-on memerlukan AWS izin, Anda dapat menggunakan peran IAM dari node (Tidak disetel) atau peran yang ada yang Anda buat untuk digunakan dengan add-on. Jika tidak ada peran untuk dipilih, maka Anda tidak memiliki peran yang ada. Terlepas dari opsi mana yang Anda pilih, lihat dokumentasi untuk add-on yang Anda buat untuk membuat

kebijakan IAM dan melampirkannya ke peran. Memilih peran IAM mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk klaster Anda. Untuk menentukan apakah Anda memiliki satu untuk cluster Anda, atau untuk membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda.

- Pilih Pengaturan konfigurasi opsional.
  - Jika add-on memerlukan konfigurasi, masukkan di kotak Nilai konfigurasi. Untuk menentukan apakah add-on memerlukan informasi konfigurasi, lihat dokumentasi untuk add-on yang Anda buat.
  - Pilih salah satu opsi yang tersedia untuk metode Resolusi Konflik.
- Pilih Berikutnya.

7. Pada halaman Review dan add, pilih Create. Setelah penginstalan add-on selesai, Anda melihat add-on yang diinstal.
8. Jika salah satu add-on yang Anda instal memerlukan langganan, selesaikan langkah-langkah berikut:
   1. Pilih tombol Berlangganan di sudut kanan bawah untuk add-on. Anda dibawa ke halaman untuk add-on di. AWS Marketplace Baca informasi tentang add-on seperti Ikhtisar Produk dan Informasi Harga.
   2. Pilih tombol Lanjutkan Berlangganan di kanan atas halaman add-on.
   3. Baca Syarat dan Ketentuan. Jika Anda menyetujuinya, pilih Terima Ketentuan. Mungkin perlu beberapa menit untuk memproses langganan. Saat langganan sedang diproses, tombol Kembali ke Amazon EKS Console berwarna abu-abu.
   4. Setelah langganan selesai diproses, tombol Kembali ke Amazon EKS Console tidak lagi berwarna abu-abu. Pilih tombol untuk kembali ke tab Add-on konsol Amazon EKS untuk klaster Anda.
   5. Untuk add-on yang Anda berlangganan, pilih Hapus dan instal ulang lalu pilih Instal ulang add-on. Pemasangan add-on dapat memakan waktu beberapa menit. Saat Instalasi selesai, Anda dapat mengonfigurasi add-on.

AWS CLI

Prasyarat

Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS

CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.

Untuk membuat add-on Amazon EKS menggunakan AWS CLI

1. Tentukan add-on mana yang tersedia. Anda dapat melihat semua add-on yang tersedia, jenisnya, dan penerbitnya. Anda juga dapat melihat URL untuk add-on yang tersedia melalui. AWS Marketplace Ganti *`1.29`* dengan versi cluster Anda.

```
aws eks describe-addon-versions --kubernetes-version 1.29 \
    --query 'addons[].{MarketplaceProductUrl: marketplaceInformation.productUrl,
 Name: addonName, Owner: owner Publisher: publisher, Type: type}' --output table
```

Contoh output adalah sebagai berikut.

```
-------------------------------------------------------------------------------------------
|
 DescribeAddonVersions
       |
+-----------------------------------------------------------------
+------------------------------+-----------------+--------------
+---------------------+
|                     MarketplaceProductUrl                      |
 Name              |      Owner       |  Publisher   |        Type         |
+-----------------------------------------------------------------
+------------------------------+-----------------+--------------
+---------------------+
|  None                                                          |  aws-ebs-csi-
driver           |  aws             |  eks         |  storage            |
|  None                                                          |  coredns
                |  aws             |  eks         |  networking         |
|  None                                                          |  kube-proxy
                |  aws             |  eks         |  networking         |
|  None                                                          |  vpc-cni
                |  aws             |  eks         |  networking         |
```

```
|  None                                                                  |  adot
                |  aws             |  eks         |  observability      |
|  https://aws.amazon.com/marketplace/pp/prodview-brb73nceicv7u |
 dynatrace_dynatrace-operator |  aws-marketplace |  dynatrace   |  monitoring
      |
|  https://aws.amazon.com/marketplace/pp/prodview-uhc2iwi5xysoc |
 upbound_universal-crossplane |  aws-marketplace |  upbound     |  infra-
management   |
|  https://aws.amazon.com/marketplace/pp/prodview-hd2ydsrgqy4li |
 teleport_teleport            |  aws-marketplace |  teleport    |  policy-
management  |
|  https://aws.amazon.com/marketplace/pp/prodview-vgghgqdsplhvc |
 factorhouse_kpow             |  aws-marketplace |  factorhouse |  monitoring
      |
|  [...]                                                                 |  [...]
                |  [...]           |  [...]       |  [...]              |
+-----------------------------------------------------------------
+------------------------------+------------------+--------------
+---------------------+
```

Output Anda mungkin berbeda. Dalam contoh keluaran ini, ada tiga add-on berbeda yang tersedia dari jenis `networking` dan lima add-on dengan penerbit tipe. `eks` Add-on dengan `aws-marketplace` di `Owner` kolom mungkin memerlukan langganan sebelum Anda dapat menginstalnya. Anda dapat mengunjungi URL untuk mempelajari lebih lanjut tentang add-on dan berlangganan.

2. Anda dapat melihat versi mana yang tersedia untuk setiap add-on. Ganti *`1.29`* dengan versi cluster Anda dan ganti *`vpc-cni`* dengan nama add-on yang dikembalikan pada langkah sebelumnya.

```
aws eks describe-addon-versions --kubernetes-version 1.29 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

Contoh output adalah sebagai berikut.

```
-----------------------------------------
|          DescribeAddonVersions         |
+-----------------+----------------------+
| Defaultversion  |       Version        |
+-----------------+----------------------+
|  False          |  v1.12.0-eksbuild.1  |
```

```
|  True           |  v1.11.4-eksbuild.1  |
|  False          |  v1.10.4-eksbuild.1  |
|  False          |  v1.9.3-eksbuild.1   |
+-----------------+----------------------+
```

Versi dengan `True` di `Defaultversion` kolom adalah versi yang add-on dibuat dengan, secara default.

3. (Opsional) Temukan opsi konfigurasi untuk add-on yang Anda pilih dengan menjalankan perintah berikut:

```
aws eks describe-addon-configuration --addon-name vpc-cni --addon-
version v1.12.0-eksbuild.1
```

```
{
    "addonName": "vpc-cni",
    "addonVersion": "v1.12.0-eksbuild.1",
    "configurationSchema": "{\"$ref\":\"#/definitions/VpcCni\",\"$schema
\":\"http://json-schema.org/draft-06/schema#\",\"definitions\":{\"Cri\":
{\"additionalProperties\":false,\"properties\":{\"hostPath\":{\"$ref\":
\"#/definitions/HostPath\"}},\"title\":\"Cri\",\"type\":\"object\"},\"Env
\":{\"additionalProperties\":false,\"properties\":{\"ADDITIONAL_ENI_TAGS
\":{\"type\":\"string\"},\"AWS_VPC_CNI_NODE_PORT_SUPPORT\":{\"format\":
\"boolean\",\"type\":\"string\"},\"AWS_VPC_ENI_MTU\":{\"format\":\"integer
\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_CONFIGURE_RPFILTER\":{\"format
\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG\":
{\"format\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_EXTERNALSNAT
\":{\"format\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_LOGLEVEL
\":{\"type\":\"string\"},\"AWS_VPC_K8S_CNI_LOG_FILE\":{\"type
\":\"string\"},\"AWS_VPC_K8S_CNI_RANDOMIZESNAT\":{\"type\":
\"string\"},\"AWS_VPC_K8S_CNI_VETHPREFIX\":{\"type\":\"string
\"},\"AWS_VPC_K8S_PLUGIN_LOG_FILE\":{\"type\":\"string\"},
\"AWS_VPC_K8S_PLUGIN_LOG_LEVEL\":{\"type\":\"string\"},\"DISABLE_INTROSPECTION
\":{\"format\":\"boolean\",\"type\":\"string\"},\"DISABLE_METRICS\":{\"format
\":\"boolean\",\"type\":\"string\"},\"DISABLE_NETWORK_RESOURCE_PROVISIONING
\":{\"format\":\"boolean\",\"type\":\"string\"},\"ENABLE_POD_ENI\":{\"format
\":\"boolean\",\"type\":\"string\"},\"ENABLE_PREFIX_DELEGATION\":{\"format
\":\"boolean\",\"type\":\"string\"},\"WARM_ENI_TARGET\":{\"format\":\"integer
\",\"type\":\"string\"},\"WARM_PREFIX_TARGET\":{\"format\":\"integer\",
\"type\":\"string\"}},\"title\":\"Env\",\"type\":\"object\"},\"HostPath\":
{\"additionalProperties\":false,\"properties\":{\"path\":{\"type\":\"string\"}},
\"title\":\"HostPath\",\"type\":\"object\"},\"Limits\":{\"additionalProperties
```

```
\":false,\"properties\":{\"cpu\":{\"type\":\"string\"},\"memory\":{\"type
\":\"string\"}},\"title\":\"Limits\",\"type\":\"object\"},\"Resources\":
{\"additionalProperties\":false,\"properties\":{\"limits\":{\"$ref\":\"#/
definitions/Limits\"},\"requests\":{\"$ref\":\"#/definitions/Limits\"}},
\"title\":\"Resources\",\"type\":\"object\"},\"VpcCni\":{\"additionalProperties
\":false,\"properties\":{\"cri\":{\"$ref\":\"#/definitions/Cri\"},\"env\":
{\"$ref\":\"#/definitions/Env\"},\"resources\":{\"$ref\":\"#/definitions/
Resources\"}},\"title\":\"VpcCni\",\"type\":\"object\"}}}"
}
```

Outputnya adalah skema JSON standar.

Berikut adalah contoh nilai konfigurasi yang valid, dalam format JSON, yang bekerja dengan skema di atas.

```
{
  "resources": {
    "limits": {
      "cpu": "100m"
    }
  }
}
```

Berikut adalah contoh nilai konfigurasi yang valid, dalam format YAMAL, yang bekerja dengan skema di atas.

```
  resources:
    limits:
      cpu: 100m
```

4. Buat add-on Amazon EKS. Salin perintah yang mengikuti ke perangkat Anda. Buat modifikasi berikut pada perintah sesuai kebutuhan dan kemudian jalankan perintah yang dimodifikasi:

   - Ganti *my-cluster* dengan nama klaster Anda.
   - Ganti *vpc-cni* dengan nama add-on yang dikembalikan dalam output dari langkah sebelumnya yang ingin Anda buat.
   - Ganti *version-number* dengan versi yang dikembalikan dalam output dari langkah sebelumnya yang ingin Anda gunakan.
   - Jika add-on menggunakan akun Kubernetes layanan dan peran IAM, ganti *111122223333* dengan ID akun Anda dan *role-name* dengan nama peran IAM yang ada yang telah Anda

buat. Untuk petunjuk tentang cara membuat peran, lihat dokumentasi untuk add-on yang Anda buat. Menentukan peran akun layanan mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk klaster Anda. Untuk menentukan apakah Anda memiliki satu untuk cluster Anda, atau untuk membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda.

Jika add-on tidak menggunakan akun Kubernetes layanan dan peran IAM, hapus. **--service-account-role-arn arn:aws:iam::*111122223333*:role/*role-name***

- Contoh perintah ini menimpa `--configuration-values` opsi dari versi add-on yang dikelola sendiri yang ada, jika ada. Ganti ini dengan nilai konfigurasi yang diinginkan, seperti string atau input file. Jika Anda tidak ingin memberikan nilai konfigurasi, hapus `--configuration-values` opsi. Jika Anda tidak AWS CLI ingin menimpa konfigurasi add-on yang dikelola sendiri yang ada, hapus opsi. *`--resolve-conflicts OVERWRITE`* Jika Anda menghapus opsi, dan add-on Amazon EKS perlu menimpa konfigurasi add-on yang dikelola sendiri, maka pembuatan add-on Amazon EKS gagal dengan pesan kesalahan untuk membantu Anda menyelesaikan konflik. Sebelum menentukan opsi ini, pastikan add-on Amazon EKS tidak mengelola pengaturan yang perlu Anda kelola, karena pengaturan tersebut ditimpa dengan opsi ini.

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name --
configuration-values '{"resources":{"limits":{"cpu":"100m"}}}' --resolve-
conflicts OVERWRITE
```

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name --
configuration-values 'file://example.yaml' --resolve-conflicts OVERWRITE
```

Untuk daftar lengkap opsi yang tersedia, lihat `create-addon` di Referensi Baris Perintah Amazon EKS. Jika add-on yang Anda buat telah `aws-marketplace` terdaftar di `Owner` kolom langkah sebelumnya, maka pembuatan mungkin gagal, dan Anda mungkin menerima pesan kesalahan yang mirip dengan kesalahan berikut.

```
{
    "addon": {
```

```
        "addonName": "addon-name",
        "clusterName": "my-cluster",
        "status": "CREATE_FAILED",
        "addonVersion": "version",
        "health": {
            "issues": [
                {
                    "code": "AddonSubscriptionNeeded",
                    "message": "You are currently not subscribed to this add-
on. To subscribe, visit the AWS Marketplace console, agree to the seller EULA,
 select the pricing type if required, then re-install the add-on"
[...]
```

Jika Anda menerima kesalahan yang mirip dengan kesalahan pada output sebelumnya, kunjungi URL di output dari langkah sebelumnya untuk berlangganan add-on. Setelah berlangganan, jalankan `create-addon` perintah lagi.

Memperbarui add-on

Amazon EKS tidak secara otomatis memperbarui add-on saat versi baru dirilis atau setelah Anda memperbarui klaster ke versi Kubernetes minor baru. Untuk memperbarui add-on untuk cluster yang ada, Anda harus memulai pembaruan. Setelah Anda memulai pembaruan, Amazon EKS memperbarui add-on untuk Anda. Sebelum memperbarui add-on, tinjau dokumentasi saat ini untuk add-on. Untuk daftar add-on yang tersedia, lihatPengaya Amazon EKS yang tersedia dari Amazon EKS. Jika add-on memerlukan peran IAM, lihat detail untuk add-on spesifik Pengaya Amazon EKS yang tersedia dari Amazon EKS untuk detail tentang membuat peran.

Anda dapat memperbarui add-on Amazon EKS menggunakan`eksctl`, AWS Management Console, atau. AWS CLI

eksctl

Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Untuk memperbarui add-on Amazon EKS menggunakan **eksctl**

1. Tentukan versi add-on dan add-on saat ini yang diinstal pada cluster Anda. Ganti *my-cluster* dengan nama klaster Anda.

```
eksctl get addon --cluster my-cluster
```

Contoh output adalah sebagai berikut.

```
NAME        VERSION              STATUS  ISSUES  IAMROLE  UPDATE AVAILABLE
coredns     v1.8.7-eksbuild.2    ACTIVE  0
kube-proxy  v1.23.7-eksbuild.1   ACTIVE  0                v1.23.8-eksbuild.2
vpc-cni     v1.10.4-eksbuild.1   ACTIVE  0                v1.12.0-
eksbuild.1,v1.11.4-eksbuild.1,v1.11.3-eksbuild.1,v1.11.2-eksbuild.1,v1.11.0-
eksbuild.1
```

Output Anda mungkin terlihat berbeda, tergantung pada add-on dan versi yang Anda miliki di cluster Anda. Anda dapat melihat bahwa dalam contoh keluaran sebelumnya, dua add-on yang ada di cluster memiliki versi yang lebih baru yang tersedia di `UPDATE AVAILABLE` kolom.

2. Perbarui add-on.

   1. Salin perintah yang mengikuti ke perangkat Anda. Buat modifikasi berikut pada perintah sesuai kebutuhan:

      - Ganti *my-cluster* dengan nama klaster Anda.
      - Ganti *region-code* dengan tempat Wilayah AWS cluster Anda berada.
      - Ganti *vpc-cni* dengan nama add-on yang dikembalikan dalam output dari langkah sebelumnya yang ingin Anda perbarui.
      - Jika Anda ingin memperbarui ke versi lebih awal dari versi terbaru yang tersedia, maka ganti *latest* dengan nomor versi yang dikembalikan dalam output dari langkah sebelumnya yang ingin Anda gunakan. Beberapa add-on memiliki versi yang direkomendasikan. Untuk informasi selengkapnya, lihat dokumentasi untuk add-on yang Anda perbarui.
      - Jika add-on menggunakan akun Kubernetes layanan dan peran IAM, ganti *111122223333* dengan ID akun Anda dan *role-name* dengan nama peran IAM yang ada yang telah Anda buat. Untuk petunjuk tentang cara membuat peran, lihat dokumentasi untuk add-on yang Anda buat. Menentukan peran akun layanan

mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk klaster Anda. Untuk menentukan apakah Anda memiliki satu untuk cluster Anda, atau untuk membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda.

Jika add-on tidak menggunakan akun Kubernetes layanan dan peran IAM, hapus baris tersebut. **`serviceAccountRoleARN: arn:aws:iam::111122223333:role/role-name`**

- *`preserve`*Opsi mempertahankan nilai yang ada untuk add-on. Jika Anda telah menetapkan nilai kustom untuk pengaturan add-on, dan Anda tidak menggunakan opsi ini, Amazon EKS menimpa nilai Anda dengan nilai defaultnya. Jika Anda menggunakan opsi ini, sebaiknya Anda menguji setiap bidang dan perubahan nilai pada klaster non-produksi sebelum memperbarui add-on pada cluster produksi Anda. Jika Anda mengubah nilai ini menjadi`overwrite`, semua pengaturan diubah ke nilai default Amazon EKS. Jika Anda telah menetapkan nilai kustom untuk setelan apa pun, nilai tersebut mungkin akan ditimpa dengan nilai default Amazon EKS. Jika Anda mengubah nilai ini`none`, Amazon EKS tidak mengubah nilai pengaturan apa pun, tetapi pembaruan mungkin gagal. Jika pembaruan gagal, Anda menerima pesan galat untuk membantu menyelesaikan konflik.

```
cat >update-addon.yaml <<EOF
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code

addons:
- name: vpc-cni
  version: latest
  serviceAccountRoleARN: arn:aws:iam::111122223333:role/role-name
  resolveConflicts: preserve
EOF
```

2. Jalankan perintah yang dimodifikasi untuk membuat `update-addon.yaml` file.
3. Terapkan file konfigurasi ke cluster Anda.

```
eksctl update addon -f update-addon.yaml
```

Untuk informasi selengkapnya tentang memperbarui add-on, lihat Addons dalam dokumentasi. `eksctl`

AWS Management Console

Untuk memperbarui add-on Amazon EKS menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel navigasi kiri, pilih Clusters, lalu pilih nama cluster yang ingin Anda konfigurasikan add-on.
3. Pilih tab Add-ons.
4. Pilih kotak di kanan atas kotak add-on dan kemudian pilih Edit.
5. Pada halaman Konfigurasi ***`nama addon`***:
   - Pilih Versi yang ingin Anda gunakan. Add-on mungkin memiliki versi yang direkomendasikan. Untuk informasi selengkapnya, lihat dokumentasi untuk add-on yang Anda perbarui.
   - Untuk peran Select IAM, Anda dapat menggunakan peran IAM dari node (Tidak disetel) atau peran yang ada yang Anda buat untuk digunakan dengan add-on. Jika tidak ada peran untuk dipilih, maka Anda tidak memiliki peran yang ada. Terlepas dari opsi mana yang Anda pilih, lihat dokumentasi untuk add-on yang Anda buat untuk membuat kebijakan IAM dan melampirkannya ke peran. Memilih peran IAM mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk klaster Anda. Untuk menentukan apakah Anda memiliki satu untuk cluster Anda, atau untuk membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda.
   - Untuk`Code editor`, masukkan informasi konfigurasi khusus add-on apa pun. Untuk informasi selengkapnya, lihat dokumentasi untuk add-on yang Anda perbarui.
   - Untuk metode Resolusi konflik, pilih salah satu opsi. Jika Anda telah menetapkan nilai kustom untuk pengaturan add-on, kami merekomendasikan opsi Preserve. Jika Anda tidak memilih opsi ini, Amazon EKS menimpa nilai Anda dengan nilai defaultnya. Jika Anda menggunakan opsi ini, sebaiknya Anda menguji setiap bidang dan perubahan nilai pada klaster non-produksi sebelum memperbarui add-on pada cluster produksi Anda.
6. Pilih Perbarui.

AWS CLI

Prasyarat

Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get`,, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.

Untuk memperbarui add-on Amazon EKS menggunakan AWS CLI

1. Lihat daftar add-on yang diinstal. Ganti *`my-cluster`* dengan nama klaster Anda.

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

   Contoh output adalah sebagai berikut.

   ```
   {
       "addons": [
           "coredns",
           "kube-proxy",
           "vpc-cni"
       ]
   }
   ```

2. Lihat versi add-on saat ini yang ingin Anda perbarui. Ganti *`my-cluster`* dengan nama cluster Anda dan *`vpc-cni`* dengan nama add-on yang ingin Anda perbarui.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
    "addon.addonVersion" --output text
   ```

   Contoh output adalah sebagai berikut.

   ```
   v1.10.4-eksbuild.1
   ```

3. Anda dapat melihat versi add-on mana yang tersedia untuk versi cluster Anda. Ganti *1.29* dengan versi cluster Anda dan *vpc-cni* dengan nama add-on yang ingin Anda perbarui.

```
aws eks describe-addon-versions --kubernetes-version 1.29 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

Contoh output adalah sebagai berikut.

```
-----------------------------------------
|          DescribeAddonVersions         |
+-----------------+---------------------+
| Defaultversion  |       Version       |
+-----------------+---------------------+
|  False          |  v1.12.0-eksbuild.1 |
|  True           |  v1.11.4-eksbuild.1 |
|  False          |  v1.10.4-eksbuild.1 |
|  False          |  v1.9.3-eksbuild.1  |
+-----------------+---------------------+
```

Versi dengan `True` di `Defaultversion` kolom adalah versi yang add-on dibuat dengan, secara default.

4. Perbarui add-on Anda. Salin perintah yang mengikuti ke perangkat Anda. Buat modifikasi berikut pada perintah, sesuai kebutuhan, dan kemudian jalankan perintah yang dimodifikasi.

- Ganti *my-cluster* dengan nama klaster Anda.
- Ganti *vpc-cni* dengan nama add-on yang ingin Anda perbarui yang dikembalikan dalam output dari langkah sebelumnya.
- Ganti *version-number* dengan versi yang dikembalikan dalam output dari langkah sebelumnya yang ingin Anda perbarui. Beberapa add-on memiliki versi yang direkomendasikan. Untuk informasi selengkapnya, lihat dokumentasi untuk add-on yang Anda perbarui.
- Jika add-on menggunakan akun Kubernetes layanan dan peran IAM, ganti *111122223333* dengan ID akun Anda dan *role-name* dengan nama peran IAM yang ada yang telah Anda buat. Untuk petunjuk tentang cara membuat peran, lihat dokumentasi untuk add-on yang Anda buat. Menentukan peran akun layanan mengharuskan Anda memiliki penyedia IAM OpenID Connect (OIDC) untuk klaster Anda. Untuk menentukan apakah Anda memiliki

satu untuk cluster Anda, atau untuk membuatnya, lihatBuat OIDC penyedia IAM untuk klaster Anda.

Jika add-on tidak menggunakan akun Kubernetes layanan dan peran IAM, hapus baris tersebut. **`serviceAccountRoleARN: arn:aws:iam::`*`111122223333`*`:role/`*`role-name`*

- Opsi **`--resolve-conflicts`** *`PRESERVE`* *mempertahankan* nilai yang ada untuk add-on. Jika Anda telah menetapkan nilai kustom untuk pengaturan add-on, dan Anda tidak menggunakan opsi ini, Amazon EKS menimpa nilai Anda dengan nilai defaultnya. Jika Anda menggunakan opsi ini, sebaiknya Anda menguji setiap bidang dan perubahan nilai pada klaster non-produksi sebelum memperbarui add-on pada cluster produksi Anda. Jika Anda mengubah nilai ini menjadi`overwrite`, semua pengaturan diubah ke nilai default Amazon EKS. Jika Anda telah menetapkan nilai kustom untuk setelan apa pun, nilai tersebut mungkin akan ditimpa dengan nilai default Amazon EKS. Jika Anda mengubah nilai ini`none`, Amazon EKS tidak mengubah nilai pengaturan apa pun, tetapi pembaruan mungkin gagal. Jika pembaruan gagal, Anda menerima pesan galat untuk membantu menyelesaikan konflik.

- Jika Anda ingin menghapus semua konfigurasi kustom maka lakukan pembaruan menggunakan *`--configuration-values '{}'`*opsi. Ini menetapkan semua konfigurasi kustom kembali ke nilai default. Jika Anda tidak ingin mengubah konfigurasi kustom Anda, jangan berikan *`--configuration-values`*tanda. Jika Anda ingin menyesuaikan konfigurasi kustom maka ganti *`{}`* dengan parameter baru. Untuk melihat daftar parameter, lihat melihat langkah skema konfigurasi di bagian buat add-on.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name --
configuration-values '{}' --resolve-conflicts PRESERVE
```

5. Periksa status pembaruan. Ganti *`my-cluster`* dengan nama cluster Anda dan *`vpc-cni`* dengan nama add-on yang Anda perbarui.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

Contoh output adalah sebagai berikut.

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-cluster",
        "status": "UPDATING",
[...]
```

Pembaruan selesai ketika statusnya`ACTIVE`.

Menghapus add-on

Saat Anda menghapus add-on Amazon EKS:

- Tidak ada downtime untuk fungsionalitas yang disediakan add-on.
- Jika add-on memiliki peran IAM yang terkait dengannya, peran IAM tidak akan dihapus.
- Amazon EKS berhenti mengelola pengaturan untuk add-on.
- Konsol berhenti memberi tahu Anda saat versi baru tersedia.
- Anda tidak dapat memperbarui add-on menggunakan AWS alat atau API apa pun.
- Anda dapat memilih untuk meninggalkan perangkat lunak add-on di cluster Anda sehingga Anda dapat mengelolanya sendiri, atau Anda dapat menghapus perangkat lunak add-on dari cluster Anda. Anda hanya harus menghapus perangkat lunak add-on dari cluster Anda jika tidak ada sumber daya di cluster Anda yang bergantung pada fungsionalitas yang disediakan add-on.

Anda dapat menghapus add-on Amazon EKS dari cluster Anda menggunakan`eksctl`, file AWS Management Console, atau file. AWS CLI

eksctl

Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Untuk menghapus add-on Amazon EKS menggunakan **eksctl**

1. Tentukan add-on saat ini yang diinstal pada cluster Anda. Ganti *my-cluster* dengan nama klaster Anda.

```
eksctl get addon --cluster my-cluster
```

   Contoh output adalah sebagai berikut.

```
NAME        VERSION              STATUS  ISSUES  IAMROLE  UPDATE AVAILABLE
coredns     v1.8.7-eksbuild.2    ACTIVE  0
kube-proxy  v1.23.7-eksbuild.1   ACTIVE  0
vpc-cni     v1.10.4-eksbuild.1   ACTIVE  0
[...]
```

   Output Anda mungkin terlihat berbeda, tergantung pada add-on dan versi yang Anda miliki di cluster Anda.

2. Hapus add-on. Ganti *my-cluster*dengan nama cluster Anda dan *name-of-add-on* dengan nama add-on dikembalikan dalam output dari langkah sebelumnya yang ingin Anda hapus. Jika Anda menghapus ***--preserve*** opsi, selain Amazon EKS tidak lagi mengelola add-on, perangkat lunak add-on dihapus dari cluster Anda.

```
eksctl delete addon --cluster my-cluster --name name-of-addon --preserve
```

AWS Management Console

Untuk menghapus add-on Amazon EKS menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel navigasi kiri, pilih Clusters, lalu pilih nama cluster yang ingin Anda hapus add-on Amazon EKS.
3. Pilih tab Add-ons.
4. Pilih kotak centang di kanan atas kotak add-on dan kemudian pilih Hapus. Pilih Pertahankan di klaster jika Anda ingin Amazon EKS berhenti mengelola pengaturan untuk add-on, tetapi ingin mempertahankan perangkat lunak add-on di klaster Anda sehingga Anda dapat mengelola sendiri semua pengaturan untuk add-on tersebut. Ketik nama add-on dan kemudian pilih Hapus.

AWS CLI

Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Untuk menghapus add-on Amazon EKS menggunakan AWS CLI

1. Lihat daftar add-on yang diinstal. Ganti *`my-cluster`* dengan nama klaster Anda.

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

   Contoh output adalah sebagai berikut.

```
{
    "addons": [
        "coredns",
        "kube-proxy",
        "vpc-cni",
        "name-of-addon"
    ]
}
```

2. Hapus add-on yang diinstal. Ganti *`my-cluster`* dengan nama cluster Anda dan *`name-of-add-on`* dengan nama add-on yang ingin Anda hapus. ***`--preserve`***Menghapus menghapus perangkat lunak add-on dari cluster Anda.

```
aws eks delete-addon --cluster-name my-cluster --addon-name name-of-addon --
preserve
```

   Output contoh yang disingkat adalah sebagai berikut.

```
{
    "addon": {
        "addonName": "name-of-add-on",
        "clusterName": "my-cluster",
        "status": "DELETING",
[...]
```

3. Periksa status penghapusan. Ganti *my-cluster* dengan nama cluster Anda dan *name-of-addon* dengan nama add-on yang Anda hapus.

```
aws eks describe-addon --cluster-name my-cluster --addon-name name-of-addon
```

Setelah add-on dihapus, contoh output adalah sebagai berikut.

```
An error occurred (ResourceNotFoundException) when calling the DescribeAddon
 operation: No addon: name-of-addon found in cluster: my-cluster
```

Ambil kompatibilitas versi addon

Gunakan describe-addon-verisionsAPI untuk membuat daftar versi addon EKS yang tersedia, dan versi Kubernetes mana yang didukung oleh setiap versi addon.

Ambil kompatibilitas versi addon ()AWS CLI

1. Verifikasi AWS CLI yang diinstal dan bekerja dengan`aws sts get-caller-identity`. Jika perintah ini tidak berfungsi, pelajari cara Memulai dengan AWS CLI.
2. Tentukan nama addon yang ingin Anda ambil informasi kompatibilitas versi, seperti. `amazon-cloudwatch-observability`
3. Tentukan versi Kubernetes dari klaster Anda, seperti. `1.28`
4. Gunakan AWS CLI untuk mengambil versi addon yang kompatibel dengan versi Kubernetes dari klaster Anda.

```
aws eks describe-addon-versions --addon-name amazon-cloudwatch-observability --
kubernetes-version 1.29
```

Contoh output adalah sebagai berikut.

```
{
    "addons": [
        {
            "addonName": "amazon-cloudwatch-observability",
            "type": "observability",
            "addonVersions": [
                {
                    "addonVersion": "v1.5.0-eksbuild.1",
```

```
                    "architecture": [
                        "amd64",
                        "arm64"
                    ],
                    "compatibilities": [
                        {
                            "clusterVersion": "1.28",
                            "platformVersions": [
                                "*"
                            ],
                            "defaultVersion": true
                        }
                    ],
[...]
```

Output ini menunjukkan bahwa versi addon `v1.5.0-eksbuild.1` kompatibel dengan versi cluster Kubernetes. `1.28`

Manajemen lapangan Kubernetes

Add-on Amazon EKS diinstal ke klaster Anda menggunakan konfigurasi standar dan praktik terbaik. Untuk informasi selengkapnya tentang menambahkan add-on Amazon EKS ke klaster Anda, lihatAdd-on Amazon EKS.

Anda mungkin ingin menyesuaikan konfigurasi add-on Amazon EKS untuk mengaktifkan fitur-fitur lanjutan. Amazon EKS menggunakan fitur penerapan Kubernetes sisi server untuk mengaktifkan pengelolaan add-on oleh Amazon EKS tanpa menimpa konfigurasi Anda untuk pengaturan yang tidak dikelola oleh Amazon EKS. Untuk informasi selengkapnya, lihat Terapkan Sisi Server dalam dokumentasi. Kubernetes Untuk mencapai hal ini, Amazon EKS mengelola seperangkat bidang minimum untuk setiap add-on yang diinstalnya. Anda dapat mengubah semua bidang yang tidak dikelola oleh Amazon EKS, atau proses bidang Kubernetes kontrol lainnya seperti`kube-controller-manager`, tanpa masalah.

> **Important**
>
> Memodifikasi bidang yang dikelola oleh Amazon EKS mencegah Amazon EKS mengelola add-on dan dapat mengakibatkan perubahan Anda ditimpa saat add-on diperbarui.

Lihat status manajemen bidang

Anda dapat menggunakan `kubectl` untuk melihat bidang mana yang dikelola oleh Amazon EKS untuk add-on Amazon EKS apa pun.

Untuk melihat status manajemen bidang

1. Tentukan add-on mana yang ingin Anda periksa. Untuk melihat semua `deployments` dan `DaemonSets` diterapkan ke cluster Anda, lihatLihat Kubernetes sumber daya.
2. Melihat bidang terkelola untuk add-on dengan menjalankan perintah berikut:

```
kubectl get type/add-on-name -n add-on-namespace -o yaml
```

   Misalnya, Anda dapat melihat bidang terkelola untuk CoreDNS add-on dengan perintah berikut.

```
kubectl get deployment/coredns -n kube-system -o yaml
```

   Manajemen lapangan tercantum di bagian berikut dalam output yang dikembalikan.

```
[...]
managedFields:
  - apiVersion: apps/v1
    fieldsType: FieldsV1
    fieldsV1:
[...]
```

> **Note**
>
> Jika Anda tidak melihat `managedFields` di output, tambahkan **`--show-managed-fields`** ke perintah dan jalankan lagi. Versi `kubectl` yang Anda gunakan menentukan apakah bidang terkelola dikembalikan secara default.

Memahami sintaks manajemen bidang di API Kubernetes

Saat Anda melihat detail untuk Kubernetes objek, bidang terkelola dan tidak terkelola dikembalikan dalam output. Bidang terkelola dapat berupa salah satu dari jenis berikut:

- Dikelola sepenuhnya - Semua kunci untuk bidang dikelola oleh Amazon EKS. Modifikasi nilai apa pun menyebabkan konflik.
- Dikelola sebagian - Beberapa kunci untuk bidang dikelola oleh Amazon EKS. Hanya modifikasi pada kunci yang dikelola secara eksplisit oleh Amazon EKS yang menyebabkan konflik.

Kedua jenis bidang ditandai dengan`manager: eks`.

Setiap kunci adalah `.` mewakili bidang itu sendiri, yang selalu memetakan ke set kosong, atau string yang mewakili sub-bidang atau item. Output untuk manajemen lapangan terdiri dari jenis deklarasi berikut:

- `f:`*`name`*, *`name`* di mana nama bidang dalam daftar.
- `k:`*`keys`*, di mana *`keys`* adalah peta bidang item daftar.
- `v:`*`value`*, di *`value`* mana nilai format JSON yang tepat dari item daftar.
- `i:`*`index`*, di *`index`* mana posisi item dalam daftar.

Bagian output berikut untuk CoreDNS add-on menggambarkan deklarasi sebelumnya:

- Bidang yang dikelola sepenuhnya - Jika bidang terkelola memiliki `f:` (bidang) yang ditentukan, tetapi tidak ada `k:` (kunci), maka seluruh bidang dikelola. Modifikasi pada nilai apa pun di bidang ini menyebabkan konflik.

  Pada output berikut, Anda dapat melihat bahwa wadah bernama `coredns` dikelola oleh`eks`. Sub-bidang `argsimage`, dan `imagePullPolicy` sub-bidang juga dikelola oleh`eks`. Modifikasi pada nilai apa pun di bidang ini menyebabkan konflik.

```
[...]
f:containers:
  k:{"name":"coredns"}:
  .: {}
  f:args: {}
  f:image: {}
  f:imagePullPolicy: {}
[...]
manager: eks
[...]
```

- Bidang yang dikelola sebagian — Jika kunci terkelola memiliki nilai yang ditentukan, kunci yang dideklarasikan dikelola untuk bidang tersebut. Memodifikasi kunci yang ditentukan menyebabkan konflik.

  Dalam output berikut, Anda dapat melihat bahwa `eks` mengelola `config-volume` dan `tmp` volume yang diatur dengan `name` kunci.

  ```
  [...]
  f:volumes:
    k:{"name":"config-volume"}:
      .: {}
      f:configMap:
        f:items: {}
        f:name: {}
      f:name: {}
    k:{"name":"tmp"}:
      .: {}
      f:name: {}
  [...]
  manager: eks
  [...]
  ```

- Menambahkan kunci ke bidang yang dikelola sebagian — Jika hanya nilai kunci tertentu yang dikelola, Anda dapat menambahkan kunci tambahan dengan aman, seperti argumen, ke bidang tanpa menyebabkan konflik. Jika Anda menambahkan kunci tambahan, pastikan bidang tersebut tidak dikelola terlebih dahulu. Menambahkan atau memodifikasi nilai apa pun yang dikelola menyebabkan konflik.

  Pada output berikut, Anda dapat melihat bahwa `name` kunci dan `name` bidang dikelola. Menambahkan atau memodifikasi nama kontainer apa pun menyebabkan konflik dengan kunci terkelola ini.

  ```
  [...]
  f:containers:
    k:{"name":"coredns"}:
  [...]
      f:name: {}
  [...]
  manager: eks
  [...]
  ```

Memverifikasi gambar kontainer selama penerapan

Jika Anda menggunakan AWS Signerdan ingin memverifikasi gambar kontainer yang ditandatangani pada saat penerapan, Anda dapat menggunakan salah satu solusi berikut:

- Gatekeeper dan Ratify — Gunakan Gatekeeper sebagai pengontrol penerimaan dan Ratifikasi dikonfigurasi dengan AWS Signer plugin sebagai pengait web untuk memvalidasi tanda tangan.
- Kyverno — Mesin Kubernetes kebijakan yang dikonfigurasi dengan AWS Signer plugin untuk memvalidasi tanda tangan.

> **Note**
>
> Sebelum memverifikasi tanda tangan gambar kontainer, konfigurasikan kebijakan penyimpanan kepercayaan dan kepercayaan Notasi, seperti yang dipersyaratkan oleh pengontrol penerimaan yang Anda pilih.

Pelatihan machine learning menggunakan Elastic Fabric Adapter

Topik ini menjelaskan cara mengintegrasikan Elastic Fabric Adapter (EFA) dengan Pods digunakan di cluster Amazon EKS Anda. Elastic Fabric Adapter (EFA) adalah antarmuka jaringan untuk instans Amazon EC2 yang dapat Anda aktifkan untuk menjalankan aplikasi yang membutuhkan komunikasi antar-simpul tingkat tinggi dalam hitungan skala pada AWS. Ini merupakan sistem operasi antarmuka bypass hardware yang meningkatkan performa komunikasi antar-instans, yang sangat penting untuk skala aplikasi ini. Dengan EFA, aplikasi Komputasi Performa Tinggi (HPC) yang menggunakan aplikasi Message Passing Interface (MPI) dan Machine Learning (ML) yang juga menggunakan NVIDIA Collective Communications Library (NCCL) yang dapat menskalakan ke ribuan CPU atau GPU. Hasilnya, Anda mendapatkan kinerja aplikasi klaster HPC lokal dengan elastisitas dan fleksibilitas cloud sesuai permintaan. AWS Mengintegrasikan EFA dengan aplikasi yang berjalan di klaster Amazon EKS dapat mengurangi waktu untuk menyelesaikan beban kerja pelatihan yang terdistribusi pada skala besar tanpa harus menambahkan instans tambahan terhadap klaster Anda. Untuk informasi selengkapnya, lihat Elastic Fabric Adaptor.

Plugin EFA yang dijelaskan dalam topik ini sepenuhnya mendukung instans P4d Amazon EC2, yang mewakili awal mula perkembangan yang terdistribusi saat ini pada machine learning di cloud. Setiap instans `p4d.24xlarge` memiliki delapan GPU NVIDIA A100, dan 400 Gbps GPUDirectRDMA diatas EFA. GPUDirectRDMA memungkinkan Anda untuk dapat berkomunikasi secara langsung melalui

GPU-ke-GPU pada seluruh simpul dengan bypass CPU, meningkatkan komunikasi bandwidth kolektif dan menurunkan latensi. Amazon EKS dan EFA terintegrasi dengan instans P4d yang menyediakan metode mulus untuk mengambil keuntungan dari kinerja tertinggi dari komputasi instans Amazon EC2 untuk pelatihan machine learning yang terdistribusi.

Prasyarat

- Sebuah klaster Amazon EKS yang sudah ada. Jika Anda belum memiliki klaster yang belum ada, gunakan salah satu panduan Memulai dengan Amazon EKS kami untuk membuatnya. Klaster Anda harus men-deploy pada VPC yang memiliki setidaknya satu subnet privat dengan memiliki cukup alamat IP yang tersedia untuk men-deploy pada simpul. Subnet privat harus memiliki akses internet luar yang disediakan oleh perangkat eksternal, seperti gateway NAT.

  Jika Anda berencana untuk menggunakan `eksctl` untuk membuat grup node Anda, juga `eksctl` dapat membuat cluster untuk Anda.
- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`apt-get`,`, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl` versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.
- Anda harus menginstal Amazon VPC CNI plugin for Kubernetes versi `1.7.10` atau yang lebih baru sebelum meluncurkan node pekerja yang mendukung beberapa Adaptor Kain Elastis, sepertip`4d.24xlarge`. Untuk informasi selengkapnya tentang memperbarui Amazon VPC CNI plugin for Kubernetes versi Anda, lihatBekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS.

Buat grup simpul

Prosedur berikut ini membantu Anda menciptakan grup simpul dengan grup simpul didukung `p4d.24xlarge` dengan antarmuka EFA dan RDMA GPUDirect, dan menjalankan contoh tes NVIDIA Collective Communications Library (NCCL) untuk Performa NCCL multi-simpul menggunakan EFA. Contohnya dapat menggunakan templat untuk didistribusikan pada pelatihan deep learning di Amazon EKS dengan menggunakan EFA.

1. Tentukan jenis instans Amazon EC2 yang mendukung EFA yang tersedia di tempat Wilayah AWS Anda ingin menerapkan node. Ganti *`region-code`* dengan Wilayah AWS yang Anda inginkan untuk menyebarkan grup node Anda.

   ```
   aws ec2 describe-instance-types --region region-code --filters Name=network-
   info.efa-supported,Values=true \
       --query "InstanceTypes[*].[InstanceType]" --output text
   ```

   Saat Anda menerapkan node, jenis instance yang ingin Anda terapkan harus tersedia di tempat klaster Anda berada. Wilayah AWS

2. Tentukan Availability Zone tempat tipe instans yang ingin Anda gunakan tersedia. Dalam tutorial ini, jenis `p4d.24xlarge` instance digunakan dan harus dikembalikan dalam output untuk Wilayah AWS yang Anda tentukan pada langkah sebelumnya. Saat Anda menerapkan node di cluster produksi, ganti *`p4d.24xlarge`* dengan jenis instance apa pun yang dikembalikan pada langkah sebelumnya.

   ```
   aws ec2 describe-instance-type-offerings --region region-code --location-type
    availability-zone --filters Name=instance-type,Values=p4d.24xlarge \
       --query 'InstanceTypeOfferings[*].Location' --output text
   ```

   Contoh output adalah sebagai berikut.

   ```
   us-west-2a    us-west-2c    us-west-2b
   ```

   Perhatikan Availability Zones yang dikembalikan untuk digunakan di langkah selanjutnya. Saat Anda menyebarkan node ke cluster, VPC Anda harus memiliki subnet dengan alamat IP yang tersedia di salah satu Availability Zone yang dikembalikan dalam output.

3. Buat grup node menggunakan salah satu `eksctl` atau AWS CLI dan AWS CloudFormation.

eksctl

Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

1. Salin isi berikut ke file bernama *`efa-cluster.yaml`*. Ganti *`example values`* dengan milik Anda sendiri. Anda dapat mengganti *`p4d.24xlarge`* dengan instance yang berbeda, tetapi jika Anda melakukannya, pastikan bahwa nilai untuk `availabilityZones` adalah Availability Zones yang dikembalikan untuk jenis instance di langkah 1.

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

metadata:
  name: my-efa-cluster
  region: region-code
  version: "1.XX"

iam:
  withOIDC: true

availabilityZones: ["us-west-2a", "us-west-2c"]

managedNodeGroups:
  - name: my-efa-ng
    instanceType: p4d.24xlarge
    minSize: 1
    desiredCapacity: 2
    maxSize: 3
    availabilityZones: ["us-west-2a"]
    volumeSize: 300
    privateNetworking: true
    efaEnabled: true
```

2. Membuat grup simpul yang dikelola dalam klaster yang sudah ada.

```
eksctl create nodegroup -f efa-cluster.yaml
```

Jika Anda tidak memiliki klaster yang sudah ada, Anda dapat menjalankan perintah berikut ini untuk membuat klaster dan grup simpul.

```
eksctl create cluster -f efa-cluster.yaml
```

> **Note**
>
> Karena tipe instance yang digunakan dalam contoh ini memiliki GPU, `eksctl` secara otomatis menginstal plugin perangkat NVIDIA Kubernetes pada setiap instance untuk Anda.

AWS CLI and AWS CloudFormation

Terdapat beberapa persyaratan di jaringan EFA, termasuk membuat grup keamanan khusus EFA, membuat Amazon EC2 Grup penempatan, dan menciptakan templat peluncuran yang menentukan satu atau lebih antarmuka EFA, dan termasuk instalasi driver EFA sebagai bagian dari data pengguna Amazon EC2. Untuk pelajari selengkapnya tentang persyaratan EFA, lihat Memulai EFA dan MPI di Panduan Pengguna Amazon EC2 untuk Instans Linux. Langkah-langkah berikut membuat semua ini untuk Anda. Ganti semua *contoh Nilai* dengan nilai Anda sendiri.

1. Mengatur beberapa variabel yang digunakan dalam langkah selanjutnya. Ganti semua *example values* dengan milik Anda sendiri. Ganti *my-cluster* dengan nama klaster Anda yang sudah ada. Nilai untuk kemudian `node_group_resources_name` digunakan untuk membuat AWS CloudFormation tumpukan. Nilai untuk `node_group_name` kemudian akan digunakan untuk membuat grup simpul di klaster Anda.

   ```
   cluster_name="my-cluster"
   cluster_region="region-code"
   node_group_resources_name="my-efa-nodegroup-resources"
   node_group_name="my-efa-nodegroup"
   ```

2. Identifikasi subnet privat pada VPC Anda di Availability Zone yang sama dengan tipe instans di mana Anda dapat men-deploy berada.

   a. Ambil kembali versi klaster Anda dan simpan dalam variabel untuk digunakan dalam langkah selanjutnya.

```
cluster_version=$(aws eks describe-cluster \
    --name $cluster_name \
    --query "cluster.version" \
    --output text)
```

b. Mengambil kembali ID VPC klaster Anda yang sudah ada dan menyimpannya dalam variabel untuk digunakan di langkah selanjutnya.

```
vpc_id=$(aws eks describe-cluster \
    --name $cluster_name \
    --query "cluster.resourcesVpcConfig.vpcId" \
    --output text)
```

c. Mengambil kembali ID dari grup keamanan pesawat pengendali untuk klaster Anda dan menyimpannya dalam variabel untuk digunakan di langkah selanjutnya.

```
control_plane_security_group=$(aws eks describe-cluster \
    --name $cluster_name \
    --query "cluster.resourcesVpcConfig.clusterSecurityGroupId" \
    --output text)
```

d. Dapatkan daftar subnet ID di VPC Anda yang berada di Availability Zone dengan kembali ke langkah 1.

```
aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=$vpc_id" "Name=availability-
zone,Values=us-west-2a" \
    --query 'Subnets[*].SubnetId' \
    --output text
```

Jika tidak ada output yang dikembalikan, coba Availability Zone yang berbeda dengan kembali ke langkah 1. Jika tidak ada subnet yang dikembalikan kepada Anda di Availability Zone pada langkah 1, maka Anda perlu membuat subnet di Availability Zone dengan kembali ke langkah 1. Jika Anda tidak memiliki ruang di VPC Anda untuk membuat subnet lain, maka Anda dapat menambahkan blok CIDR ke VPC dan membuat subnet di blok CIDR baru, atau membuat cluster baru di VPC baru.

e. Menentukan apakah subnet adalah subnet privat dengan memeriksa tabel rute untuk subnet.

```
aws ec2 describe-route-tables \
    --filter Name=association.subnet-id,Values=subnet-0d403852a65210a29 \
    --query "RouteTables[].Routes[].GatewayId" \
    --output text
```

Contoh output adalah sebagai berikut.

```
local
```

Jika outputnya `local igw-02adc64c1b72722e2`, maka subnet merupakan subnet publik. Anda harus memilih subnet privat di Availability Zone dengan kembali ke langkah 1. Setelah Anda mengidentifikasi subnet privat, catat ID untuk digunakan di langkah selanjutnya.

f. Mengatur variabel dengan ID subnet privat dari langkah sebelumnya untuk digunakan pada langkah-langkah selanjutnya.

```
subnet_id=your-subnet-id
```

3. Unduh AWS CloudFormation template.

```
curl -O https://raw.githubusercontent.com/aws-samples/aws-efa-eks/main/
cloudformation/efa-p4d-managed-nodegroup.yaml
```

4. Salin teks berikut ke komputer Anda. Ganti *p4d.24xlarge* dengan tipe instance dari langkah 1. Ganti *subnet-0d403852a65210a29* dengan ID subnet pribadi yang Anda identifikasi pada langkah 2.b.v. Ganti *path-to-downloaded-cfn-template* dengan jalur ke `efa-p4d-managed-nodegroup.yaml` yang Anda unduh di langkah sebelumnya. Ganti *your-public-key-name* dengan nama kunci publik Anda. Setelah Anda membuat penggantian, jalankan perintah yang dimodifikasi.

```
aws cloudformation create-stack \
    --stack-name ${node_group_resources_name} \
    --capabilities CAPABILITY_IAM \
    --template-body file://path-to-downloaded-cfn-template \
    --parameters \
       ParameterKey=ClusterName,ParameterValue=${cluster_name} \
       ParameterKey=ClusterControlPlaneSecurityGroup,ParameterValue=
${control_plane_security_group} \
```

```
        ParameterKey=VpcId,ParameterValue=${vpc_id} \
        ParameterKey=SubnetId,ParameterValue=${subnet_id} \
        ParameterKey=NodeGroupName,ParameterValue=${node_group_name} \
        ParameterKey=NodeImageIdSSMParam,ParameterValue=/aws/service/eks/
optimized-ami/${cluster_version}/amazon-linux-2-gpu/recommended/image_id \
        ParameterKey=KeyName,ParameterValue=your-public-key-name \
        ParameterKey=NodeInstanceType,ParameterValue=p4d.24xlarge
```

5. Menentukan kapan Anda men-deploy tumpukan pada langkah sebelumnya yang telah di-deploy.

   ```
   aws cloudformation wait stack-create-complete --stack-name
    $node_group_resources_name
   ```

   Tidak ada output dari perintah sebelumnya, tetapi prompt shell Anda tidak kembali hingga tumpukan telah dibuat.

6. Membuat grup simpul Anda menggunakan sumber daya yang dibuat oleh tumpukan AWS CloudFormation pada langkah sebelumnya.

   a. Ambil informasi dari AWS CloudFormation tumpukan yang digunakan dan simpan dalam variabel.

   ```
   node_instance_role=$(aws cloudformation describe-stacks \
       --stack-name $node_group_resources_name \
       --query='Stacks[].Outputs[?OutputKey==`NodeInstanceRole`].OutputValue'
    \
       --output text)
   launch_template=$(aws cloudformation describe-stacks \
       --stack-name $node_group_resources_name \
       --query='Stacks[].Outputs[?OutputKey==`LaunchTemplateID`].OutputValue'
    \
       --output text)
   ```

   b. Membuat grup simpul yang dikelola dengan menggunakan templat peluncuran dan simpul IAM role yang dibuat pada langkah sebelumnya.

   ```
   aws eks create-nodegroup \
       --cluster-name $cluster_name \
       --nodegroup-name $node_group_name \
       --node-role $node_instance_role \
       --subnets $subnet_id \
   ```

```
    --launch-template id=$launch_template,version=1
```

c. Mengonfirmasi bahwa simpul telah diciptakan.

```
aws eks describe-nodegroup \
   --cluster-name ${cluster_name} \
   --nodegroup-name ${node_group_name} | jq -r .nodegroup.status
```

Jangan lanjutkan hingga status telah kembali dari perintah sebelumnya `ACTIVE`. Membutuhkan waktu beberapa menit hingga simpul sudah siap.

7. Jika Anda memilih jenis instans GPU, Anda harus menerapkan plugin perangkat NVIDIA untuk. Kubernetes Ganti *vX.X.X* dengan s-device-plugin versi NVIDIA/K8 yang Anda inginkan sebelum menjalankan perintah berikut.

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

4. Men-deploy plugin perangkat EFA Kubernetes.

Plugin perangkat EFA Kubernetes mendeteksi serta mengiklankan antarmuka EFA sebagai sumber daya yang dapat dialokasikan ke Kubernetes. Aplikasi dapat menggunakan jenis sumber daya yang diperluas `vpc.amazonaws.com/efa` dalam spesifikasi Pod permintaan seperti CPU dan memori. Untuk informasi selengkapnya, lihat Mengkonsumsi sumber daya yang diperluas dalam Kubernetes dokumentasi. Setelah diminta, plugin secara otomatis menetapkan dan memasang antarmuka EFA ke file. Pod Menggunakan plugin perangkat menyederhanakan pengaturan EFA dan tidak memerlukan a Pod untuk berjalan dalam mode istimewa.

```
kubectl apply -f https://raw.githubusercontent.com/aws-samples/aws-efa-eks/main/
manifest/efa-k8s-device-plugin.yml
```

(Opsional) mend-deploy sebuah sampel aplikasi yang kompatibel dengan EFA

Men-deploy Operasi MPI Kubeflow

Untuk tes NCCL Anda dapat menerapkan Operasi Kubeflow MPI. Operasi MPI mempermudah untuk menjalankan pelatihan terdistribusi tipe-Allreduce di Kubernetes. Untuk informasi lebih lanjut, lihat Operator MPI diGitHub.

```
kubectl apply -f https://raw.githubusercontent.com/kubeflow/mpi-operator/master/deploy/
v2beta1/mpi-operator.yaml
```

Menjalankan multi-simpul uji performa NCCL untuk memverifikasi GPUDirectrDMA/EFA

Untuk memverifikasi Performa NCCL dengan GPUDirectRDMA melalui EFA, jalankan uji standar Performa NCCL. Untuk informasi lebih lanjut, lihat repo NCCL-Tests resmi di. GitHub Anda dapat menggunakan contoh Dockerfile yang disertakan dengan pengujian ini yang sudah dibuat untuk keduanya NVIDIA CUDA`11.2`dan versi terbaru EFA.

Sebagai alternatif, Anda dapat mengunduh AWS Docker gambar yang tersedia dari repo Amazon ECR.

> **Important**
>
> Sebuah pertimbangan penting yang diperlukan untuk mengadopsi EFA dengan Kubernetes untuk mengkonfigurasi dan mengelola Huge Pages sebagai sumber daya dalam klaster. Untuk informasi selengkapnya, lihat Mengelola Halaman Besar dalam Kubernetes dokumentasi. Instans Amazon EC2 dengan diinstalnya pra-alokasi 5128 2M Huge Pages EFA driver, yang dapat diminta sebagai sumber daya untuk dikonsumsi dalam spesifikasi pekerjaan Anda.

Selesaikan langkah-langkah berikut ini untuk menjalankan dua simpul uji kinerja NCCL. Pada contoh tes tugas NCCL, setiap pekerja meminta delapan GPU, 5210Mi dari hugepages-2Mi, empat EFA, dan memori sebesar 8000Mi, yang berarti secara efektif setiap pekerja mengkonsumsi semua sumber daya dari instans `p4d.24xlarge`.

1. Buat tugas NCCL-test.

   ```
   kubectl apply -f https://raw.githubusercontent.com/aws-samples/aws-efa-eks/main/
   examples/simple/nccl-efa-tests.yaml
   ```

   Contoh output adalah sebagai berikut.

   nccl-tests-efa mpijob.kubeflow.org/ dibuat

2. Lihat lari AndaPods.

```
kubectl get pods
```

Contoh output adalah sebagai berikut.

```
NAME                                READY   STATUS     RESTARTS   AGE
nccl-tests-efa-launcher-nbql9  0/1     Init:0/1   0          2m49s
nccl-tests-efa-worker-0          1/1     Running    0          2m49s
nccl-tests-efa-worker-1          1/1     Running    0          2m49s
```

Operator MPI membuat peluncur Pod dan 2 pekerja Pods (satu di setiap node).

3. Lihat log untuk `efa-launcher`Pod. Ganti *wzr8j* dengan nilai dari output Anda.

```
kubectl logs -f nccl-tests-efa-launcher-nbql9
```

Untuk contoh lainnya, lihat repositori sampel Amazon EKS EFA di. GitHub

Inferensi machine learning menggunakan AWS Inferentia

Topik ini menjelaskan cara membuat klaster Amazon EKS dengan simpul yang berjalan Inf1 Amazon EC2 instans dan (opsional) men-deploy aplikasi sampel. Instans Amazon EC2 Inf1 didukung oleh chip AWS Inferentia, yang dibuat khusus AWS untuk memberikan kinerja tinggi dan inferensi biaya terendah di cloud. Model pembelajaran mesin digunakan ke wadah menggunakan AWS Neuron, kit pengembangan perangkat lunak khusus (SDK) yang terdiri dari kompiler, runtime, dan alat profil yang mengoptimalkan kinerja inferensi pembelajaran mesin chip Inferentia. AWS Neuron mendukung kerangka kerja pembelajaran mesin populer seperti TensorFlow, PyTorch, dan MXNet.

> **Note**
>
> Perangkat Neuron ID logis harus berdekatan. Jika beberapa perangkat Neuron yang Pod meminta dijadwalkan pada jenis `inf1.6xlarge` atau `inf1.24xlarge` instance (yang memiliki lebih dari satu perangkat Neuron), itu Pod akan gagal dimulai jika Kubernetes penjadwal memilih ID perangkat yang tidak berdekatan. Untuk informasi selengkapnya, lihat ID logis perangkat harus bersebelahan. GitHub

Prasyarat

- Miliki `eksctl` yang terinstal pada komputer Anda. Jika Anda belum menginstalnya, lihat Instalasi di `eksctl` dokumentasi.
- Miliki `kubectl` yang ter-install pada komputer Anda. Untuk informasi selengkapnya, lihat Menginstal atau memperbarui `kubectl`.
- (Opsional) Miliki `python3` yang ter-install pada komputer Anda. Jika Anda belum menginstalnya, lihat Mengunduh Python untuk petunjuk instalasi.

Membuat klaster

Untuk membuat klaster dengan simpul instans Inf1 Amazon EC2

1. Membuat klaster dengan simpul instans Inf1 Amazon EC2. Anda dapat mengganti *`inf1.2xlarge`* dengan jenis instans Inf1 apa pun. `eksctl`Utilitas mendeteksi bahwa Anda meluncurkan grup node dengan tipe `Inf1` instans dan akan memulai node Anda menggunakan salah satu AMI Amazon Linux Amazon yang dioptimalkan Amazon EKS yang dioptimalkan.

   > **Note**
   >
   > Anda tidak dapat menggunakan peran IAM untuk akun layanan dengan TensorFlow Serving.

   ```
   eksctl create cluster \
       --name inferentia \
       --region region-code \
       --nodegroup-name ng-inf1 \
       --node-type inf1.2xlarge \
       --nodes 2 \
       --nodes-min 1 \
       --nodes-max 4 \
       --ssh-access \
       --ssh-public-key your-key \
       --with-oidc
   ```

> **Note**
>
> Perhatikan nilai baris output berikut. Ini digunakan dalam langkah (opsional) selanjutnya.
>
> ```
> [9]  adding identity "arn:aws:iam::111122223333:role/
> eksctl-inferentia-nodegroup-ng-in-NodeInstanceRole-FI7HIYS3BS09" to auth
>  ConfigMap
> ```

Saat meluncurkan grup node dengan `Inf1` instance, `eksctl` secara otomatis menginstal plugin Kubernetes perangkat AWS Neuron. Plugin ini mengiklankan perangkat Neuron sebagai sumber daya sistem ke Kubernetes penjadwal, yang dapat diminta oleh wadah. Sebagai tambahan untuk kebijakan IAM simpul Amazon EKS default, Amazon S3 hanya membaca kebijakan akses yang ditambahkan sehingga aplikasi sampel, dibahas dalam langkah berikutnya, dapat memuat model terlatih dari Amazon S3.

2. Pastikan semua Pods sudah dimulai dengan benar.

```
kubectl get pods -n kube-system
```

Output yang disingkat:

```
NAME                                   READY   STATUS    RESTARTS   AGE
[...]
neuron-device-plugin-daemonset-6djhp   1/1     Running   0          5m
neuron-device-plugin-daemonset-hwjsj   1/1     Running   0          5m
```

(Opsional) Menyebarkan gambar aplikasi TensorFlow Penyajian

Sebuah model terlatih harus dikompilasikan ke target Inferentia sebelum dapat di-deploy pada instans Inferentia. Untuk melanjutkan, Anda memerlukan TensorFlow model yang dioptimalkan Neuron yang disimpan di Amazon S3. Jika Anda belum memilikinya SavedModel, ikuti tutorial untuk membuat model ResNet 50 yang kompatibel dengan Neuron dan unggah hasilnya SavedModel ke S3. ResNet-50 adalah model pembelajaran mesin populer yang digunakan untuk tugas pengenalan gambar. Untuk informasi lebih lanjut tentang kompilasi model Neuron, lihat Chip AWS Inferentia Dengan DLAMI di Panduan Pengembang. AWS Deep Learning AMI

Manifes penerapan sampel mengelola wadah penyajian inferensi bawaan yang TensorFlow disediakan oleh AWS Deep Learning Containers. Di dalam wadah adalah AWS Neuron Runtime dan aplikasi TensorFlow Serving. Daftar lengkap Deep Learning Containers pra-bangun yang dioptimalkan untuk Neuron dipertahankan di GitHub bawah Gambar yang Tersedia. Saat start-up, DLC akan mengambil model Anda dari Amazon S3, meluncurkan Neuron TensorFlow Serving dengan model yang disimpan, dan menunggu permintaan prediksi.

Jumlah perangkat Neuron yang dialokasikan untuk aplikasi penyajian Anda agar dapat disesuaikan dengan mengubah sumber daya `aws.amazon.com/neuron` dalam deployment yaml. Harap dicatat bahwa komunikasi antara TensorFlow Serving dan runtime Neuron terjadi melalui GRPC, yang membutuhkan penerusan `IPC_LOCK` kemampuan ke wadah.

1. Menambahkan Kebijakan IAM `AmazonS3ReadOnlyAccess` untuk peran instans simpul yang dibuat di langkah 1 dari Membuat klaster. Hal ini diperlukan agar aplikasi sampel dapat memuat model terlatih dari Amazon S3.

   ```
   aws iam attach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
       --role-name eksctl-inferentia-nodegroup-ng-in-NodeInstanceRole-FI7HIYS3BS09
   ```

2. Buat file bernama `rn50_deployment.yaml` dengan isi berikut ini. Memperbarui kode wilayah dan jalur model agar sesuai dengan pengaturan yang Anda inginkan. Nama model adalah untuk tujuan identifikasi ketika klien membuat permintaan ke TensorFlow server. Contoh ini menggunakan nama model untuk mencocokkan contoh skrip klien ResNet 50 yang akan digunakan pada langkah selanjutnya untuk mengirim permintaan prediksi.

   ```
   aws ecr list-images --repository-name neuron-rtd --registry-id 790709498068 --
   region us-west-2
   ```

   ```
   kind: Deployment
   apiVersion: apps/v1
   metadata:
     name: eks-neuron-test
     labels:
       app: eks-neuron-test
       role: master
   spec:
     replicas: 2
     selector:
       matchLabels:
   ```

```
      app: eks-neuron-test
      role: master
  template:
    metadata:
      labels:
        app: eks-neuron-test
        role: master
    spec:
      containers:
        - name: eks-neuron-test
          image: 763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-inference-
neuron:1.15.4-neuron-py37-ubuntu18.04
          command:
            - /usr/local/bin/entrypoint.sh
          args:
            - --port=8500
            - --rest_api_port=9000
            - --model_name=resnet50_neuron
            - --model_base_path=s3://your-bucket-of-models/resnet50_neuron/
          ports:
            - containerPort: 8500
            - containerPort: 9000
          imagePullPolicy: IfNotPresent
          env:
            - name: AWS_REGION
              value: "us-east-1"
            - name: S3_USE_HTTPS
              value: "1"
            - name: S3_VERIFY_SSL
              value: "0"
            - name: S3_ENDPOINT
              value: s3.us-east-1.amazonaws.com
            - name: AWS_LOG_LEVEL
              value: "3"
          resources:
            limits:
              cpu: 4
              memory: 4Gi
              aws.amazon.com/neuron: 1
            requests:
              cpu: "1"
              memory: 1Gi
          securityContext:
            capabilities:
```

```
            add:
              - IPC_LOCK
```

3. Men-deploy model.

```
kubectl apply -f rn50_deployment.yaml
```

4. Membuat file bernama `rn50_service.yaml` dengan konten berikut. Port HTTP dan gRPC dibuka untuk menerima permintaan prediksi.

```
kind: Service
apiVersion: v1
metadata:
  name: eks-neuron-test
  labels:
    app: eks-neuron-test
spec:
  type: ClusterIP
  ports:
    - name: http-tf-serving
      port: 8500
      targetPort: 8500
    - name: grpc-tf-serving
      port: 9000
      targetPort: 9000
  selector:
    app: eks-neuron-test
    role: master
```

5. Buat Kubernetes layanan untuk aplikasi Melayani TensorFlow model Anda.

```
kubectl apply -f rn50_service.yaml
```

(Opsional) Buat prediksi terhadap layanan TensorFlow Serving Anda

1. Untuk menguji secara lokal, meneruskan port gRPC ke layanan `eks-neuron-test`.

```
kubectl port-forward service/eks-neuron-test 8500:8500 &
```

2. Membuat skrip Python yang disebut `tensorflow-model-server-infer.py` dengan konten berikut. Skrip ini menjalankan inferensi melalui gRPC, yang merupakan kerangka kerja layanan.

```
import numpy as np
   import grpc
   import tensorflow as tf
   from tensorflow.keras.preprocessing import image
   from tensorflow.keras.applications.resnet50 import preprocess_input
   from tensorflow_serving.apis import predict_pb2
   from tensorflow_serving.apis import prediction_service_pb2_grpc
   from tensorflow.keras.applications.resnet50 import decode_predictions

   if __name__ == '__main__':
       channel = grpc.insecure_channel('localhost:8500')
       stub = prediction_service_pb2_grpc.PredictionServiceStub(channel)
       img_file = tf.keras.utils.get_file(
           "./kitten_small.jpg",
           "https://raw.githubusercontent.com/awslabs/mxnet-model-server/master/
docs/images/kitten_small.jpg")
       img = image.load_img(img_file, target_size=(224, 224))
       img_array = preprocess_input(image.img_to_array(img)[None, ...])
       request = predict_pb2.PredictRequest()
       request.model_spec.name = 'resnet50_inf1'
       request.inputs['input'].CopyFrom(
           tf.make_tensor_proto(img_array, shape=img_array.shape))
       result = stub.Predict(request)
       prediction = tf.make_ndarray(result.outputs['output'])
       print(decode_predictions(prediction))
```

3. Menjalankan penulisan untuk mengirimkan prediksi ke layanan Anda.

```
python3 tensorflow-model-server-infer.py
```

Contoh output adalah sebagai berikut.

```
[[(u'n02123045', u'tabby', 0.68817204), (u'n02127052', u'lynx', 0.12701613),
 (u'n02123159', u'tiger_cat', 0.08736559), (u'n02124075', u'Egyptian_cat',
 0.063844085), (u'n02128757', u'snow_leopard', 0.009240591)]]
```

Manajemen klaster

Bab ini mencakup topik berikut untuk membantu mengelola klaster Anda. Anda juga dapat melihat informasi tentang Kubernetessumber daya Anda denganAWS Management Console.

- KubernetesDashboard adalah tujuan umum, UI berbasis web untuk Kubernetes cluster. Hal ini memungkinkan pengguna untuk mengelola aplikasi yang berjalan di cluster dan memecahkan masalah mereka, serta mengelola cluster itu sendiri. Untuk informasi lain, lihat: GitHub repositori KubernetesDashboard.
- Menginstal Server Kubernetes Metrik— Server Kubernetes Metrik adalah agregator data penggunaan sumber daya di klaster Anda. Ini tidak digunakan secara default di klaster Anda, tetapi digunakan oleh Kubernetes add-on, seperti Kubernetes Dasbor dan. Penskala Otomatis Pod Horizontal Dalam topik ini Anda akan mempelajari cara menginstal Server Metrik.
- Menggunakan Helm dengan Amazon EKS— Manajer paket Helm untuk Kubernetes membantu Anda menginstal dan mengelola aplikasi di klaster AndaKubernetes. Topik ini membantu Anda menginstal dan menjalankan binari Helm sehingga Anda dapat menginstal dan mengelola grafik menggunakan Helm CLI pada komputer lokal Anda.
- Menandai sumber daya Amazon EKS Anda – Untuk membantu Anda mengelola sumber daya Amazon EKS, Anda dapat menetapkan metadata Anda sendiri ke setiap sumber daya dalam bentuk tanda. Topik ini menjelaskan penandaan dan menunjukkan kepada Anda cara membuatnya.
- Amazon EKS service quotas – Akun AWS Anda memiliki kuota default, yang sebelumnya disebut sebagai batas, untuk masing-masing layanan AWS. Pelajari tentang kuota untuk Amazon EKS dan cara meningkatkannya.

Pemantauan biaya

Pemantauan biaya adalah aspek penting dalam mengelola Kubernetes cluster Anda di Amazon EKS. Dengan mendapatkan visibilitas ke biaya klaster Anda, Anda dapat mengoptimalkan pemanfaatan sumber daya, menetapkan anggaran, dan membuat keputusan berdasarkan data tentang penerapan Anda. Amazon EKS menyediakan dua solusi pemantauan biaya, masing-masing dengan keunggulan uniknya sendiri, untuk membantu Anda melacak dan mengalokasikan biaya Anda secara efektif:

AWS Data alokasi biaya split penagihan untuk Amazon EKS — Fitur asli ini terintegrasi secara mulus dengan Konsol AWS Penagihan, memungkinkan Anda menganalisis dan mengalokasikan

biaya menggunakan antarmuka dan alur kerja yang sama yang Anda gunakan untuk layanan lain. AWS Dengan alokasi biaya terpisah, Anda dapat memperoleh wawasan tentang Kubernetes biaya Anda secara langsung di samping pengeluaran Anda yang lain AWS , sehingga lebih mudah untuk mengoptimalkan biaya secara holistik di seluruh lingkungan Anda. AWS Anda juga dapat memanfaatkan fitur AWS Penagihan yang ada seperti Cost Categories dan Cost Anomaly Detection untuk lebih meningkatkan kemampuan manajemen biaya Anda. Untuk informasi selengkapnya, lihat Memahami data alokasi biaya terpisah di Panduan Pengguna AWS Penagihan.

Kubecost— Amazon EKS mendukung Kubecost, alat pemantauan biaya Kubernetes. Kubecost menawarkan pendekatan Kubernetes-native yang kaya fitur untuk pemantauan biaya, menyediakan rincian biaya granular oleh sumber daya Kubernetes, rekomendasi pengoptimalan biaya, serta dasbor dan laporan. out-of-the-box Kubecost juga mengambil data harga yang akurat dengan mengintegrasikan dengan Laporan AWS Biaya dan Penggunaan, memastikan Anda mendapatkan tampilan yang tepat dari biaya Amazon EKS Anda. Pelajari cara menginstal Kubecost.

AWS Penagihan - Alokasi Biaya Split

Pemantauan biaya menggunakan data alokasi biaya AWS terpisah untuk Amazon EKS

Anda dapat menggunakan data alokasi biaya AWS terpisah untuk Amazon EKS untuk mendapatkan visibilitas biaya granular untuk kluster Amazon EKS Anda. Hal ini memungkinkan Anda untuk menganalisis, mengoptimalkan, dan biaya tolak bayar dan penggunaan untuk aplikasi AndaKubernetes. Anda mengalokasikan biaya aplikasi ke unit bisnis individual dan tim berdasarkan CPU Amazon EC2 dan sumber daya memori yang dikonsumsi oleh Kubernetes aplikasi Anda. Data alokasi biaya terpisah untuk Amazon EKS memberikan visibilitas ke dalam biaya per Pod, dan memungkinkan Anda untuk mengumpulkan data biaya per Pod menggunakan namespace, cluster, dan primitif lainnya. Kubernetes Berikut ini adalah contoh Kubernetes primitif yang dapat Anda gunakan untuk menganalisis data alokasi biaya Amazon EKS.

- Nama klaster
- Deployment
- Namespace
- Simpul
- Nama Beban Kerja
- Jenis Beban Kerja

Untuk informasi selengkapnya tentang penggunaan data alokasi biaya terpisah, lihat Memahami data alokasi biaya terpisah di Panduan Pengguna AWS Penagihan.

Mengatur Laporan Biaya dan Penggunaan

Anda dapat mengaktifkan Data Alokasi Biaya Pisahkan untuk EKS di Konsol Manajemen Biaya AWS Command Line Interface, atau AWS SDK.

Gunakan yang berikut ini untuk Data Alokasi Biaya Split:

1. Pilih untuk Membagi Data Alokasi Biaya. Untuk informasi selengkapnya, lihat Mengaktifkan data alokasi biaya terpisah di AWS Cost and Usage Report Panduan Pengguna.
2. Sertakan data dalam laporan baru atau yang sudah ada.
3. Lihat laporannya. Anda dapat menggunakan konsol Billing and Cost Management atau melihat file laporan di Amazon Simple Storage Service.

Kubecost

Amazon EKS mendukungKubecost, yang dapat Anda gunakan untuk memantau biaya yang dipecah berdasarkan Kubernetes sumber daya termasukPods, node, ruang nama, dan label. Sebagai administrator Kubernetes platform dan pemimpin keuangan, Anda dapat menggunakannya Kubecost untuk memvisualisasikan rincian biaya Amazon EKS, mengalokasikan biaya, dan membebankan kembali unit organisasi seperti tim aplikasi. Anda dapat memberikan data biaya yang transparan dan akurat kepada tim internal dan unit bisnis Anda berdasarkan AWS tagihan aktual mereka. Selain itu, Anda juga bisa mendapatkan rekomendasi khusus untuk pengoptimalan biaya berdasarkan lingkungan infrastruktur dan pola penggunaan dalam cluster mereka. Untuk informasi selengkapnyaKubecost, lihat Kubecostdokumentasi.

Amazon EKS menyediakan bundel yang AWS dioptimalkan Kubecost untuk visibilitas biaya cluster. Anda dapat menggunakan perjanjian AWS dukungan yang ada untuk mendapatkan dukungan.

Prasyarat

- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan satu, lihatMemulai dengan Amazon EKS. Cluster harus memiliki node Amazon EC2 karena Anda tidak dapat berjalan di node Kubecost Fargate.
- Alat baris `kubectl` perintah diinstal pada perangkat Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi

cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl` versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.

- Helm versi 3.9.0 atau yang lebih baru dikonfigurasi pada perangkat Anda atau. AWS CloudShell Untuk menginstal atau memperbarui Helm, lihat the section called “Menggunakan Helm”.
- Jika klaster Anda versi `1.23` atau yang lebih baru, Anda harus the section called “Driver CSI Amazon EBS” menginstal klaster Anda.

Untuk menginstal Kubecost

1. Tentukan versi Kubecost untuk menginstal. Anda dapat melihat versi yang tersedia di kubecost/cost-analyzer di Amazon ECR Public Gallery. Untuk informasi selengkapnya tentang kompatibilitas Kubecost versi dan Amazon EKS, lihat Persyaratan Lingkungan dalam dokumentasi Kubecost.
2. Instal Kubecost dengan perintah berikut. *Ganti kubecost-version dengan nilai retreived dari ECR, seperti 1.108.1.*

   ```
   helm upgrade -i kubecost oci://public.ecr.aws/kubecost/cost-analyzer --
   version kubecost-version \
       --namespace kubecost --create-namespace \
       -f https://raw.githubusercontent.com/kubecost/cost-analyzer-helm-chart/develop/
   cost-analyzer/values-eks-cost-monitoring.yaml
   ```

   Kubecostmerilis versi baru secara teratur. Anda dapat memperbarui versi Anda menggunakan `helm upgrade`. Secara default, instalasi termasuk Prometheusserver lokal dan`kube-state-metrics`. Anda dapat menyesuaikan penerapan untuk menggunakan Amazon Managed Service for Prometheus dengan mengikuti dokumentasi di Integrasi dengan Amazon EKS pemantauan biaya. Untuk daftar semua pengaturan lain yang dapat Anda konfigurasikan, lihat contoh file konfigurasi GitHub.
3. Pastikan yang dibutuhkan Pods sedang berjalan.

   ```
   kubectl get pods -n kubecost
   ```

   Contoh output adalah sebagai berikut.

   ```
   NAME                                           READY   STATUS    RESTARTS   AGE
   kubecost-cost-analyzer-b9788c99f-5vj5b         2/2     Running   0          3h27m
   ```

```
kubecost-kube-state-metrics-99bb8c55b-bn2br   1/1     Running   0          3h27m
kubecost-prometheus-server-7d9967bfc8-9c8p7   2/2     Running   0          3h27m
```

4. Di perangkat Anda, aktifkan penerusan port untuk mengekspos dasbor. Kubecost

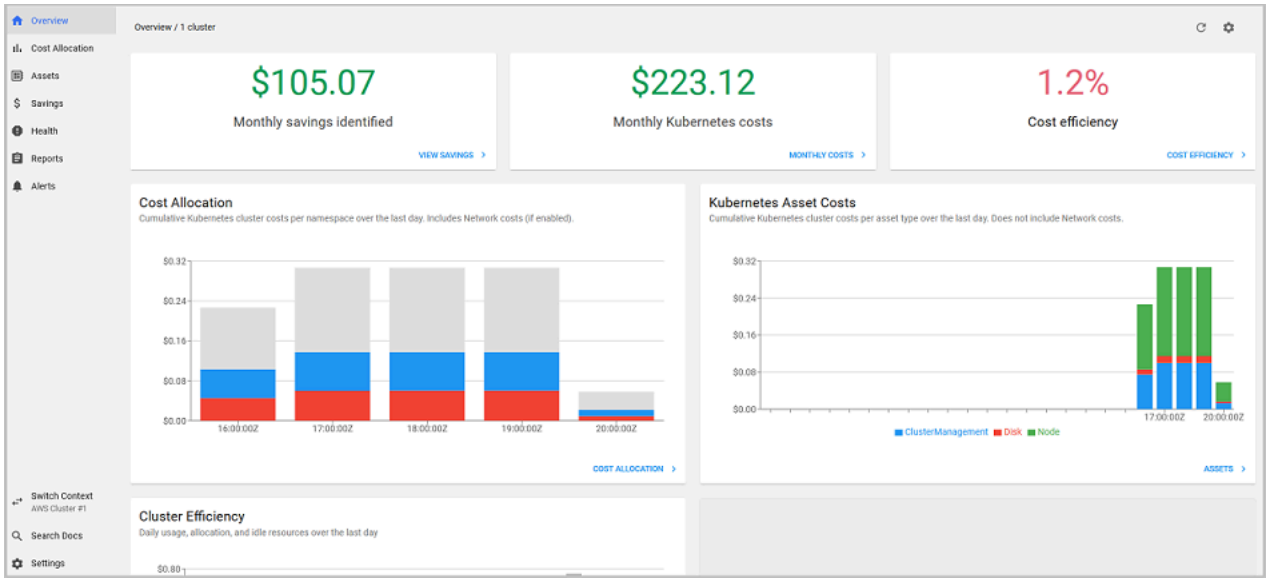
```
kubectl port-forward --namespace kubecost deployment/kubecost-cost-analyzer 9090
```

Atau, Anda dapat menggunakan AWS Load Balancer Controlleruntuk mengekspos Kubecost dan menggunakan Amazon Cognito untuk otentikasi, otorisasi, dan manajemen pengguna. Untuk informasi selengkapnya, lihat Cara menggunakan Application Load Balancer dan Amazon Cognito untuk mengautentikasi pengguna untuk aplikasi web Anda Kubernetes.

5. Pada perangkat yang sama yang Anda selesaikan langkah sebelumnya, buka browser web dan masukkan alamat berikut.

```
http://localhost:9090
```

Anda melihat halaman Kubecost Ikhtisar di browser Anda. Mungkin diperlukan 5-10 menit Kubecost untuk mengumpulkan metrik. Anda dapat melihat pengeluaran Amazon EKS Anda, termasuk biaya klaster kumulatif, biaya Kubernetes aset terkait, dan pengeluaran agregat bulanan.



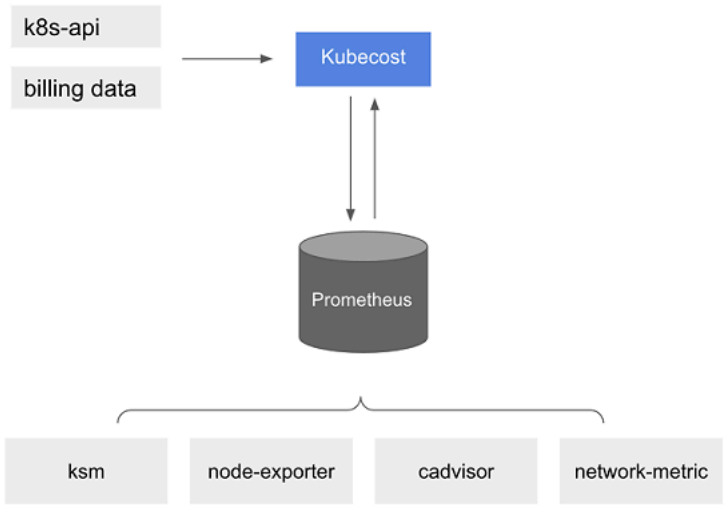
6. Untuk melacak biaya di tingkat klaster, beri tag sumber daya Amazon EKS Anda untuk penagihan. Untuk informasi selengkapnya, lihat Menandai sumber daya Anda untuk penagihan.

Anda juga dapat melihat informasi berikut dengan memilihnya di panel kiri dasbor:

- Alokasi biaya — Lihat biaya Amazon EKS bulanan dan biaya kumulatif untuk setiap ruang nama Anda dan dimensi lainnya selama tujuh hari terakhir. Ini berguna untuk memahami bagian mana dari aplikasi Anda yang berkontribusi terhadap pengeluaran Amazon EKS.
- Aset — Lihat biaya aset AWS infrastruktur yang terkait dengan sumber daya Amazon EKS Anda.

Fitur tambahan

- Metrik biaya ekspor - Pemantauan biaya yang dioptimalkan Amazon EKS digunakan dengan Kubecost danPrometheus, yang merupakan sistem pemantauan sumber terbuka dan database deret waktu. Kubecostmembaca metrik dari Prometheus dan kemudian melakukan perhitungan alokasi biaya dan menulis metrik kembali ke. Prometheus KubecostFront-end membaca metrik dari Prometheus dan menunjukkannya di antarmuka pengguna. Kubecost Arsitektur diilustrasikan dalam diagram berikut.



  Dengan Prometheuspra-instal, Anda dapat menulis kueri untuk mencerna Kubecost data ke dalam sistem intelijen bisnis Anda saat ini untuk analisis lebih lanjut. Anda juga dapat menggunakannya sebagai sumber data untuk Grafanadasbor Anda saat ini untuk menampilkan biaya klaster Amazon EKS yang akrab dengan tim internal Anda. Untuk mempelajari lebih lanjut tentang cara menulis Prometheus kueri, lihat `readme` file PrometheusKonfigurasi GitHub atau gunakan contoh model Grafana JSON di repositori KubecostGithub sebagai referensi.
- AWS Cost and Usage Report integrasi — Untuk melakukan perhitungan alokasi biaya untuk klaster Amazon EKS Anda, Kubecost ambil informasi harga publik Layanan AWS dan AWS sumber

daya dari API Daftar AWS Harga. Anda juga dapat mengintegrasikan Kubecost dengan AWS Cost and Usage Reportuntuk meningkatkan akurasi informasi harga khusus untuk Anda Akun AWS. Informasi ini mencakup program diskon perusahaan, penggunaan instans cadangan, paket tabungan, dan penggunaan spot. Untuk mempelajari lebih lanjut tentang cara kerja AWS Cost and Usage Report integrasi, lihat Integrasi Penagihan AWS Cloud di Kubecost dokumentasi.

Hapus Kubecost

Anda dapat menghapus Kubecost dari cluster Anda dengan perintah berikut.

```
helm uninstall kubecost --namespace kubecost
kubectl delete ns kubecost
```

Pertanyaan umum

Lihat pertanyaan dan jawaban umum berikut tentang penggunaan Kubecost dengan Amazon EKS.

Apa perbedaan antara bundel khusus Kubecost dan versi gratis Kubecost (juga dikenal sebagaiOpenCost)?

AWS dan Kubecost berkolaborasi untuk menawarkan versi yang disesuaikan dari. Kubecost Versi ini mencakup subset fitur komersial tanpa biaya tambahan. Lihat tabel berikut untuk fitur yang disertakan dalam bundel kustomKubecost.

Fitur	Kubecosttingkat gratis	Amazon EKS dioptimalkan bundel Kubecost kustom	Kubecost Enterprise
Deployment	Pengguna di-host	Pengguna di-host	Pengguna di-host atau Kubecost dihosting (SaaS)
Jumlah cluster yang didukung	Tidak terbatas.	Tidak terbatas.	Tidak terbatas.
Database yang didukung	Lokal Prometheus	Layanan Terkelola Lokal Prometheus atau Amazon untuk Prometheus	Prometheus, Layanan Dikelola Amazon untuk Prometheus,, atau Cortex Thanos

Fitur	Kubecosttingkat gratis	Amazon EKS dioptimalkan bundel Kubecost kustom	Kubecost Enterprise
Dukungan retensi basis data	15 hari	Data historis tak terbatas	Data historis tak terbatas
KubecostRetensi API (ETL)	15 hari	15 hari	Data historis tak terbatas
Visibilitas biaya cluster	Cluster tunggal	Multi-cluster terpadu	Multi-cluster terpadu
Visibilitas cloud hybrid	-	Amazon EKS dan Amazon EKS Anywhere cluster	Dukungan multi-cloud dan hybrid-cloud
Peringatan dan laporan berulang	-	Peringatan efisiensi, peringatan anggaran, peringatan perubahan belanja, dan lebih banyak lagi yang didukung	Peringatan efisiensi, peringatan anggaran, peringatan perubahan belanja, dan lebih banyak lagi yang didukung
Laporan tersimpan	-	Laporan menggunak an data 15 hari	Laporan menggunak an data historis tak terbatas
Integrasi penagihan cloud	Diperlukan untuk setiap cluster individu	Dukungan harga khusus untuk AWS (termasuk beberapa cluster dan beberapa akun)	Dukungan harga khusus untuk AWS (termasuk beberapa cluster dan beberapa akun)
Rekomendasi tabungan	Wawasan cluster tunggal	Wawasan cluster tunggal	Wawasan multi-clu ster
Tata Kelola: Audit	-	-	Audit peristiwa biaya historis

Fitur	Kubecosttingkat gratis	Amazon EKS dioptimalkan bundel Kubecost kustom	Kubecost Enterprise
Dukungan masuk tunggal (SSO)	-	Amazon Cognito didukung	Okta, Auth0, PingID, KeyCloak
Kontrol akses berbasis peran (RBAC) dengan SAFL **2.0**	-	-	Okta, Auth0, PingID, Keycloak
Pelatihan dan orientasi perusahaan	-	-	Pelatihan dan FinOps orientasi layanan lengkap

Apa fitur retensi Kubecost API (ETL)?

Fitur Kubecost ETL menggabungkan dan mengatur metrik ke visibilitas biaya permukaan pada berbagai tingkat granularitas (seperti,, dan). `namespace-level` `pod-level` `deployment-level` Untuk Kubecost paket kustom, pelanggan mendapatkan data dan wawasan dari metrik selama 15 hari terakhir.

Apa fitur peringatan dan laporan berulang? Peringatan dan laporan apa yang disertakan?

Kubecostperingatan memungkinkan tim menerima pembaruan tentang Kubernetes pengeluaran waktu nyata serta pengeluaran cloud. Laporan berulang memungkinkan tim menerima tampilan belanja historis Kubernetes dan cloud yang disesuaikan. Keduanya dapat dikonfigurasi menggunakan Kubecost UI atau Helm nilai. Mereka mendukung email,Slack, danMicrosoft Teams.

Apa yang termasuk dalam laporan yang disimpan?

Kubecostlaporan yang disimpan adalah pandangan metrik biaya dan efisiensi yang telah ditentukan sebelumnya. Mereka termasuk biaya berdasarkan cluster, namespace, label, dan banyak lagi.

Apa itu integrasi penagihan cloud?

Integrasi dengan API AWS penagihan memungkinkan Kubecost untuk menampilkan out-of-cluster biaya (seperti Amazon S3). Selain itu, memungkinkan Kubecost untuk mendamaikan Kubecost

prediksi in-cluster dengan data penagihan aktual untuk memperhitungkan penggunaan spot, paket tabungan, dan diskon perusahaan.

Apa yang termasuk dalam rekomendasi tabungan?

Kubecostmemberikan wawasan dan otomatisasi untuk membantu pengguna mengoptimalkan Kubernetes infrastruktur dan pengeluaran mereka.

Apakah ada biaya untuk fungsi ini?

Tidak. Anda dapat menggunakan versi ini tanpa Kubecost biaya tambahan. Jika Anda menginginkan Kubecost fitur tambahan yang tidak termasuk dalam paket ini, Anda dapat membeli lisensi perusahaan Kubecost melalui AWS Marketplace, atau dari Kubecost langsung.

Apakah dukungan tersedia?

Ya. Anda dapat membuka kasus dukungan dengan AWS Support tim di Kontak AWS.

Apakah saya memerlukan lisensi untuk menggunakan Kubecost fitur yang disediakan oleh integrasi Amazon EKS?

Tidak.

Dapatkah saya berintegrasi Kubecost dengan AWS Cost and Usage Report pelaporan yang lebih akurat?

Ya. Anda dapat mengonfigurasi Kubecost untuk mengambil data dari AWS Cost and Usage Report untuk mendapatkan visibilitas biaya yang akurat, termasuk diskon, harga Spot, harga instans yang dipesan, dan lainnya. Untuk informasi selengkapnya, lihat Integrasi Penagihan AWS Cloud di Kubecost dokumentasi.

Apakah versi ini mendukung manajemen biaya cluster Kubernetes yang dikelola sendiri di Amazon EC2?

Tidak. Versi ini hanya kompatibel dengan cluster Amazon EKS.

Dapat Kubecost melacak biaya untuk Amazon EKS AWS Fargate?

Kubecostmemberikan upaya terbaik untuk menunjukkan visibilitas biaya cluster untuk Amazon EKS di Fargate, tetapi dengan akurasi yang lebih rendah dibandingkan dengan Amazon EKS di Amazon EC2. Hal ini terutama disebabkan oleh perbedaan dalam bagaimana Anda ditagih untuk penggunaan Anda. Dengan Amazon EKS di Fargate, Anda ditagih untuk sumber daya yang dikonsumsi. Dengan Amazon EKS di node Amazon EC2, Anda ditagih untuk sumber daya yang

disediakan. Kubecostmenghitung biaya node Amazon EC2 berdasarkan spesifikasi node, yang mencakup CPU, RAM, dan penyimpanan sementara. Dengan Fargate, biaya dihitung berdasarkan sumber daya yang diminta untuk Fargate. Pods

Bagaimana saya bisa mendapatkan pembaruan dan versi baruKubecost?

Anda dapat memutakhirkan Kubecost versi Anda menggunakan prosedur peningkatan Helm standar. Versi terbaru ada di Galeri Publik Amazon ECR.

Apakah **`kubectl-cost`** CLI didukung? Bagaimana cara menginstalnya?

Ya. `Kubectl-cost`adalah alat open source oleh Kubecost (Apache 2.0 License) yang menyediakan akses CLI Kubernetes ke metrik alokasi biaya. Untuk menginstal`kubectl-cost`, lihat Instalasi di GitHub.

Apakah antarmuka Kubecost pengguna didukung? Bagaimana cara mengaksesnya?

Kubecostmenyediakan dasbor web yang dapat Anda akses melalui penerusan `kubectl` port, ingress, atau penyeimbang beban. Anda juga dapat menggunakan file AWS Load Balancer Controller untuk mengekspos Kubecost dan menggunakan Amazon Cognito untuk otentikasi, otorisasi, dan manajemen pengguna. Untuk informasi selengkapnya, lihat Cara menggunakan Application Load Balancer dan Amazon Cognito untuk mengautentikasi pengguna untuk aplikasi web Kubernetes Anda di blog. AWS

Apakah Amazon EKS Anywhere didukung?

Tidak.

Menginstal Server Kubernetes Metrik

Server Kubernetes Metrik adalah agregator data penggunaan sumber daya di klaster Anda, dan tidak digunakan secara default di kluster Amazon EKS. Untuk informasi selengkapnya, lihat Server Kubernetes Metrik aktif. GitHub Server Metrik biasanya digunakan oleh Kubernetes add on lain, seperti Penskala Otomatis Pod Horizontal atau KubernetesDashboard. Untuk informasi selengkapnya, lihat Pipa metrik sumber daya dalam Kubernetes dokumentasi. Topik ini menjelaskan cara menerapkan Server Kubernetes Metrik di klaster Amazon EKS Anda.

> **Important**
>
> Metrik dimaksudkan untuk point-in-time analisis dan bukan sumber yang akurat untuk analisis historis. Mereka tidak dapat digunakan sebagai solusi pemantauan atau untuk

tujuan penskalaan non-otomatis lainnya. Untuk informasi tentang alat pemantauan, lihatObservabilitas di Amazon EKS.

Deploy Server Metrik

1. Deploy Server Metrik dengan perintah berikut:

```
kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
```

2. Verifikasi bahwa `metrics-server` penyebaran menjalankan jumlah yang diinginkan Pods dengan perintah berikut.

```
kubectl get deployment metrics-server -n kube-system
```

Contoh output adalah sebagai berikut.

```
NAME             READY   UP-TO-DATE   AVAILABLE   AGE
metrics-server   1/1     1            1           6m
```

Menggunakan Helm dengan Amazon EKS

Manajer paket Helm untuk Kubernetes membantu Anda menginstal dan mengelola aplikasi di Kubernetes cluster Anda. Untuk informasi selengkapnya, lihat Dokumentasi Helm. Topik ini membantu Anda menginstal dan menjalankan biner Helm sehingga Anda dapat menginstal dan mengelola grafik menggunakan Helm CLI pada sistem lokal Anda.

Important

Sebelum Anda dapat menginstal grafik Helm di Amazon EKS klaster Anda, Anda harus konfigurasi `kubectl` untuk bekerja untuk Amazon EKS. Jika Anda belum melakukannya, lihat Membuat atau memperbarui `kubeconfig` file untuk klaster Amazon EKS sebelum melanjutkan. Jika perintah berikut berhasil untuk klaster Anda, Anda dikonfigurasi dengan benar.

```
kubectl get svc
```

Untuk menginstal biner Helm pada sistem lokal Anda

1. Jalankan perintah yang sesuai untuk sistem operasi klien Anda.

   - Jika Anda menggunakan macOS dengan Homebrew, instal biner dengan perintah berikut.

     ```
     brew install helm
     ```

   - Jika Anda menggunakan Windows Chocolatey, instal binari dengan perintah berikut.

     ```
     choco install kubernetes-helm
     ```

   - Jika Anda menggunakanLinux, instal binari dengan perintah berikut.

     ```
     curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 >
      get_helm.sh
     chmod 700 get_helm.sh
     ./get_helm.sh
     ```

     > **Note**
     >
     > Jika Anda mendapatkan pesan yang `openssl` harus diinstal terlebih dahulu, Anda dapat menginstalnya dengan perintah berikut.
     >
     > ```
     > sudo yum install openssl
     > ```

2. Untuk mengambil biner baru di PATH, Tutup jendela terminal Anda saat ini dan buka jendela baru.
3. Lihat versi Helm yang Anda instal.

   ```
   helm version | cut -d + -f 1
   ```

   Contoh output adalah sebagai berikut.

   ```
   v3.9.0
   ```

4. Pada titik ini, Anda dapat menjalankan perintah Helm apa pun (seperti `helm install` *chart-name*) untuk menginstal, memodifikasi, menghapus, atau kueri Helm bagan di klaster Anda. Jika Anda baru mengenal Helm dan tidak memiliki bagan tertentu untuk dipasang, Anda dapat:

- Percobaan dengan memasang contoh bagan. Melihat Memasang contoh bagan di Helm Panduan Quick Start.
- Membuat contoh grafik dan mengajukannya ke Amazon ECR. Untuk informasi selengkapnya, lihat Mengajukan grafik Helm di Panduan Pengguna Registri Kontainer Elastis Amazon.
- Instal bagan Amazon EKS dari GitHub repo eks-charts atau dari. ArtifactHub

Menandai sumber daya Amazon EKS Anda

Anda dapat menggunakan tag untuk membantu mengelola sumber daya Amazon EKS Anda. Topik ini memberikan gambaran umum pada fungsi tanda serta menunjukkan bagaimana Anda bisa membuat tanda.

Topik

> **Note**
>
> Tag adalah jenis metadata yang terpisah dari Kubernetes label dan anotasi. Untuk informasi selengkapnya tentang jenis metadata lainnya ini, lihat bagian berikut dalam dokumentasi: Kubernetes
>
> - Label dan Selektor
> - Anotasi

Dasar tanda

Tanda merupakan label yang Anda tetapkan ke sumber daya AWS. Setiap tag terdiri dari kunci dan nilai opsional.

Dengan tag, Anda dapat mengkategorikan sumber daya AndaAWS. Misalnya, Anda dapat mengkategorikan sumber daya berdasarkan tujuan, pemilik, atau lingkungan. Bila Anda memiliki banyak sumber daya dari jenis yang sama, Anda dapat menggunakan tag yang Anda tetapkan ke sumber daya tertentu untuk mengidentifikasi sumber daya tersebut dengan cepat. Misalnya, Anda dapat menentukan satu set tanda untuk klaster Amazon EKS yang dapat membantu Anda melacak setiap pemilik dan tingkat tumpukan klaster. Kami menyarankan agar Anda merancang serangkaian kunci tanda yang konsisten untuk setiap jenis sumber daya. Kemudian Anda dapat mencari dan memfilter sumber daya berdasarkan tanda yang Anda tambahkan.

Setelah Anda menambahkan sebuah tanda, Anda dapat mengedit kunci serta nilai tanda atau menghilangkan tanda dari sumber daya kapanpun yang Anda mau. Jika Anda menghapus sebuah sumber daya, tanda apapun untuk sumber daya tersebut juga dihapus.

Tanda tidak memiliki makna semantik pada Amazon EKS dan diartikan secara jelas sebagai serangkaian karakter saja. Anda dapat mengatur nilai tag ke string kosong. Namun, Anda tidak dapat mengatur nilai tag ke null. Jika Anda menambahkan tanda yang memiliki kunci yang sama dengan tanda yang ada pada sumber daya tersebut, nilai yang baru akan menimpa nilai sebelumnya.

Jika Anda menggunakan AWS Identity and Access Management (IAM), Anda dapat mengontrol pengguna mana dalam akun AWS Anda yang memiliki izin untuk mengelola tanda.

Menandai Sumber Daya Anda

Tag dukungan sumber daya Amazon EKS berikut:

- kluster
- kelompok simpul terkelola
- Profil Fargate

Anda dapat menandai sumber daya ini menggunakan yang berikut:

- Jika Anda menggunakan konsol Amazon EKS, Anda dapat menerapkan tag ke sumber daya baru atau yang sudah ada kapan saja. Anda dapat melakukannya dengan menggunakan tabel Tanda pada halaman sumber daya yang relevan. Untuk informasi selengkapnya, lihat Bekerja dengan tanda menggunakan konsol.
- Jika Anda menggunakan`eksctl`, Anda dapat menerapkan tag ke sumber daya ketika mereka dibuat menggunakan `--tags` opsi.

- Jika Anda menggunakanAWS CLI, Amazon EKS API, atau AWS SDK, Anda dapat menerapkan tag ke sumber daya baru menggunakan `tags` parameter pada tindakan API yang relevan. Anda dapat memasang tanda ke sumber daya yang ada menggunakan tindakan API `TagResource`. Untuk informasi lebih lanjut, lihat TagResource.

Saat Anda menggunakan beberapa tindakan pembuatan sumber daya, Anda juga dapat menentukan tag untuk sumber daya pada saat yang sama saat Anda membuatnya. Jika tag tidak dapat diterapkan saat sumber daya sedang dibuat, sumber daya gagal dibuat. Mekanisme ini memastikan bahwa sumber daya yang ingin Anda tag dibuat dengan tag yang Anda tentukan atau tidak dibuat sama sekali. Jika Anda menandai sumber daya saat membuatnya, Anda tidak perlu menjalankan skrip penandaan khusus setelah membuat sumber daya.

Tag tidak menyebar ke sumber daya lain yang terkait dengan sumber daya yang Anda buat. Misalnya, tag profil Fargate tidak menyebar ke sumber daya lain yang terkait dengan profil Fargate, seperti Pods yang dijadwalkan dengannya.

Batasan tanda

Pembatasan berikut berlaku untuk tag:

- Maksimal 50 tag dapat dikaitkan dengan sumber daya.
- Kunci tag tidak dapat diulang untuk satu sumber daya. Setiap kunci tag harus unik, dan hanya dapat memiliki satu nilai.
- Panjang tombol dapat mencapai 128 karakter di UTF-8.
- Nilai dapat mencapai 256 karakter di UTF-8.
- Jika beberapa Layanan AWS dan sumber daya menggunakan skema penandaan Anda, batasi jenis karakter yang Anda gunakan. Beberapa layanan mungkin memiliki batasan pada karakter yang diizinkan. Umumnya, karakter yang diizinkan adalah huruf, angka, spasi, dan karakter berikut: **`+ - = . _ : /@`**.
- Kunci dan nilai tag peka huruf besar-kecil.
- Jangan gunakan `aws:`, `AWS:`, atau kombinasi huruf besar atau kecil seperti prefiks baik untuk kunci atau nilai. Ini disimpan hanya untuk penggunaan AWS. Anda tidak dapat mengedit atau menghapus kunci atau nilai tanda dengan prefiks ini. Tag dengan awalan ini tidak dihitung terhadap tags-per-resource batas Anda.

Menandai sumber daya Anda untuk penagihan

Saat Anda menerapkan tag ke kluster Amazon EKS, Anda dapat menggunakannya untuk alokasi biaya di Laporan Biaya & Penggunaan Anda. Data pengukuran dalam Laporan Biaya & Penggunaan Anda menunjukkan penggunaan di semua kluster Amazon EKS Anda. Untuk informasi selengkapnya, lihat laporan AWS biaya dan penggunaan di Panduan AWS Billing Pengguna.

Tag alokasi biaya yang AWS dihasilkan, khususnya`aws:eks:cluster-name`, memungkinkan Anda memecah biaya instans Amazon EC2 menurut kluster Amazon EKS individual di Cost Explorer. Namun, tag ini tidak menangkap biaya pesawat kontrol. Tag secara otomatis ditambahkan ke instans Amazon EC2 yang berpartisipasi dalam klaster Amazon EKS. Perilaku ini terjadi terlepas dari apakah instance disediakan menggunakan grup node terkelola Amazon EKSKarpenter, atau langsung dengan Amazon EC2. Tag khusus ini tidak dihitung terhadap batas 50 tag. Untuk menggunakan tag, pemilik akun harus mengaktifkannya di AWS Billing konsol atau dengan menggunakan API. Ketika pemilik akun AWS Organizations manajemen mengaktifkan tag, tag juga diaktifkan untuk semua akun anggota organisasi.

Anda juga dapat mengatur informasi penagihan berdasarkan sumber daya yang memiliki nilai kunci tag yang sama. Misalnya, Anda dapat menandai beberapa sumber daya dengan nama aplikasi tertentu, dan kemudian mengatur informasi penagihan Anda. Dengan begitu, Anda dapat melihat total biaya aplikasi itu di beberapa layanan. Untuk informasi selengkapnya tentang pengaturan laporan alokasi biaya dengan tanda, lihat Laporan Alokasi Biaya Bulanan dalam Panduan Pengguna AWS Billing.

> **Note**
>
> Jika Anda baru saja mengaktifkan pelaporan, data untuk bulan berjalan tersedia untuk dilihat setelah 24 jam.

Cost Explorer adalah alat pelaporan yang tersedia sebagai bagian dari Tingkat AWS Gratis. Anda dapat menggunakan Cost Explorer untuk melihat bagan sumber daya Amazon EKS Anda dari 13 bulan terakhir. Anda juga dapat memperkirakan berapa banyak yang akan Anda belanjakan untuk tiga bulan ke depan. Anda dapat melihat pola berapa banyak yang Anda habiskan untuk AWS sumber daya dari waktu ke waktu. Misalnya, Anda dapat menggunakannya untuk mengidentifikasi area yang memerlukan penyelidikan lebih lanjut dan melihat tren yang dapat Anda gunakan untuk memahami biaya Anda. Anda juga dapat menentukan rentang waktu untuk data, dan melihat data waktu menurut hari atau bulan.

Bekerja dengan tanda menggunakan konsol

Menggunakan konsol Amazon EKS, Anda dapat mengelola tag yang terkait dengan cluster baru atau yang sudah ada dan grup node terkelola.

Saat Anda memilih halaman khusus sumber daya di konsol Amazon EKS, halaman tersebut menampilkan daftar sumber daya tersebut. Misalnya, jika Anda memilih Cluster dari panel navigasi kiri, konsol akan menampilkan daftar kluster Amazon EKS. Saat Anda memilih sumber daya dari salah satu daftar (misalnya, klaster tertentu) yang mendukung tanda ini, Anda dapat melihat dan mengelola tandanya di tab Tanda.

Anda juga dapat menggunakan Editor Tag diAWS Management Console, yang menyediakan cara terpadu untuk mengelola tag Anda. Untuk informasi selengkapnya, lihat Menandai AWS sumber daya Anda dengan Editor AWS Tag di Panduan Pengguna Editor Tag.

Menambahkan tag pada sumber daya pada pembuatan

Anda dapat menambahkan tanda ke klaster Amazon EKS, grup simpul terkelola, dan profil Fargate ketika Anda membuat fitur tersebut. Untuk informasi selengkapnya, lihat Membuat klaster Amazon EKS.

Menambahkan dan menghapus tag pada sumber daya

Anda dapat menambahkan atau menghapus tag yang terkait dengan cluster Anda langsung dari halaman sumber daya.

Untuk menambahkan atau menghapus sebuah tanda pada sebuah sumber daya individu

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pada bilah navigasi, pilih yang Wilayah AWS akan digunakan.
3. Pada panel navigasi sebelah kiri, pilih Klaster.
4. Pilih klaster tertentu.
5. Pilih tab Tag, lalu pilih Kelola tag.
6. Pada halaman Kelola tag, tambahkan atau hapus tag Anda seperlunya.
   - Untuk menambahkan tanda, pilih Tambahkan tanda. Kemudian tentukan kunci dan nilai untuk setiap tag.
   - Untuk menghapus tag, pilih Hapus tag.

7. Ulangi proses ini untuk setiap tag yang ingin Anda tambahkan atau hapus.
8. Pilih Perbarui untuk menyelesaikan.

Cara menggunakan tanda dengan menggunakan CLI, API, atau **eksctl**

Gunakan perintah berikut AWS CLI atau operasi Amazon EKS API untuk menambahkan, memperbarui, mendaftarkan, serta menghapus tanda untuk sumber daya Anda. Anda hanya dapat menggunakan `eksctl` untuk menambahkan tag sekaligus membuat sumber daya baru dengan satu perintah.

Dukungan penandaan untuk sumber daya Amazon EKS Anda

Tugas	AWS CLI	AWS Tools for Windows PowerShell	Tindakan API
Penambahan atau penimpaan satu tanda atau lebih.	tag-resource	Add-EKSResourceTag	TagResource
Penghapusan satu tanda atau lebih.	untag-resource	Remove-EKSResourceTag	UntagResource

Contoh-contoh berikut menunjukkan cara menambahkan atau menghilangkan tanda sumber daya menggunakan AWS CLI.

Contoh 1: Tandai klaster yang sudah ada

Perintah berikut akan menandai klaster yang sudah ada.

```
aws eks tag-resource --resource-arn resource_ARN --tags team=devs
```

Contoh 2: Hapus tanda dari klaster yang sudah ada

Perintah berikut akan menghapus tanda dari klaster yang sudah ada.

```
aws eks untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

Contoh 3: Cantumkan tanda untuk sumber daya

Perintah berikut mencantumkan tag yang terkait dengan sumber daya yang ada.

```
aws eks list-tags-for-resource --resource-arn resource_ARN
```

Saat Anda menggunakan beberapa tindakan pembuatan sumber daya, Anda dapat menentukan tag pada saat yang sama saat Anda membuat sumber daya. Tindakan berikut mendukung penentuan tag saat Anda membuat sumber daya.

Tugas	AWS CLI	AWS Tools for Windows PowerShell	Tindakan API	**eksctl**
Buat sebuah klaster	create-cluster	New-EKSCluster	CreateCluster	create cluster
Buat grup pengelola simpul*	create-nodegroup	New-EKSNodegroup	CreateNodegroup	create nodegroup
Buat profil Fargate	create-fargate-profile	New-EKSFargateProfile	CreateFargateProfile.html	create fargateprofile

* Jika Anda ingin juga menandai instans Amazon EC2 saat membuat grup node terkelola, buat grup node terkelola menggunakan templat peluncuran. Untuk informasi selengkapnya, lihat Penandaan instans Amazon EC2. Jika instances sudah ada, Anda dapat menambahkan tanda pada instans secara manual. Untuk informasi lebih lanjut, lihat Penandaan sumber daya Anda di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Amazon EKS service quotas

Amazon EKS telah terintegrasi dengan Service Quotas, AWS layanan yang dapat Anda gunakan untuk melihat dan mengelola kuota Anda dari lokasi pusat. Untuk informasi selengkapnya, lihat Apa itu Service Quotas? di Panduan Pengguna Service Quotas. Dengan integrasi Service Quotas, Anda dapat dengan cepat mencari nilai Amazon EKS dan kuota AWS Fargate layanan Anda menggunakan dan. AWS Management Console AWS CLI

AWS Management Console

Untuk melihat kuota layanan Amazon EKS dan Fargate menggunakan AWS Management Console

1. Buka konsol Kuota Layanan di https://console.aws.amazon.com/servicequotas/.
2. Di panel navigasi kiri, pilih Layanan AWS.
3. Dari Layanan AWSdaftar, cari dan pilih Amazon Elastic Kubernetes Service (Amazon EKS) atau. AWS Fargate

   Dalam daftar Kuota layanan, Anda dapat melihat nama kuota layanan, nilai yang diterapkan (jika tersedia), kuota AWS default, dan apakah nilai kuota dapat disesuaikan.
4. Untuk melihat informasi tambahan tentang service quotas, seperti deskripsi, pilih nama kuota.
5. (Opsional) Untuk meminta peningkatan kuota, pilih kuota yang ingin Anda tingkatkan, kemudian pilih Meminta peningkatan kuota, masukkan atau pilih informasi yang diperlukan, dan pilih Minta.

Untuk bekerja lebih lanjut dengan kuota layanan menggunakan AWS Management Console, lihat Panduan Pengguna Service Quotas. Untuk meminta penambahan kuota, lihat Meminta Penambahan Kuota dalam Panduan Pengguna Kuota Layanan.

AWS CLI

Untuk melihat kuota layanan Amazon EKS dan Fargate menggunakan AWS CLI

Jalankan perintah berikut untuk menampilkan kuota Amazon EKS Anda.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code eks \
    --output table
```

Jalankan perintah berikut untuk menampilkan kuota Fargate Anda.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code fargate \
```

```
    --output table
```

> **Note**
>
> Kuota yang dikembalikan adalah jumlah tugas Amazon ECS atau Amazon EKS Pods yang dapat berjalan secara bersamaan di Fargate di akun ini saat ini. Wilayah AWS

Untuk bekerja lebih banyak dengan kuota layanan menggunakan AWS CLI, lihat `service-quotas`di AWS CLI Command Reference. Untuk meminta kenaikan kuota, lihat `request-service-quota-increase` perintah di AWS CLI Command Reference.

Service quotas

Nama	Default	Dapat disesuaikan	Deskripsi
Akses entri per cluster	Setiap Wilayah yang didukung: 3.000	Tidak	Jumlah maksimum entri akses per cluster.
Klaster	Setiap Wilayah yang didukung: 100	Ya	Jumlah maksimum kluster EKS di akun ini di Wilayah saat ini.
Grup keamanan bidang kendali per klaster	Setiap Wilayah yang didukung: 4	Tidak	Jumlah maksimum grup keamanan bidang kontrol per cluster (ini ditentukan saat Anda membuat cluster).
Langganan Perusahaan EKS Anywhere	Setiap Wilayah yang didukung: 10	Ya	Jumlah maksimum Langganan Perusahaan EKS Anywhere di akun ini di Wilayah saat ini.

Nama	Default	Dapat disesuaikan	Deskripsi
Profil Fargate per klaster	Setiap Wilayah yang didukung: 10	Ya	Jumlah maksimum profil Fargate per cluster.
Pasangan label per pemilih profil Fargate	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum pasangan label per pemilih profil Fargate.
Grup simpul terkelola per klaster	Setiap Wilayah yang didukung: 30	Ya	Jumlah maksimum grup node terkelola per cluster.
Simpul per grup simpul terkelola	Setiap Wilayah yang didukung: 450	Ya	Jumlah maksimum node per grup node terkelola.
Rentang CIDR akses titik akhir publik per klaster	Setiap Wilayah yang didukung: 40	Tidak	Jumlah maksimum rentang CIDR akses endpoint publik per cluster (ini ditentukan saat Anda membuat atau memperbarui cluster).
Cluster terdaftar	Setiap Wilayah yang didukung: 10	Ya	Jumlah maksimum cluster terdaftar di akun ini di Wilayah saat ini.
Pemilih per profil Fargate	Setiap Wilayah yang didukung: 5	Ya	Jumlah maksimum pemilih per profil Fargate.

Note

Nilai default adalah kuota awal yang ditetapkan oleh AWS. Nilai default ini terpisah dari nilai kuota yang diterapkan aktual dan kuota layanan maksimum yang mungkin. Untuk informasi

selengkapnya, lihat Terminologi dalam Service Quotas di Panduan Pengguna Service Quotas.

Kuota layanan ini tercantum di bawah Amazon Elastic Kubernetes Service (Amazon EKS) di konsol Service Quotas. Untuk meminta peningkatan kuota untuk nilai yang ditampilkan sebagai dapat disesuaikan, lihat Meminta peningkatan kuota dalam Panduan Pengguna Service Quotas.

AWS Fargate kuota layanan

AWS FargateLayanan di konsol Service Quotas mencantumkan beberapa kuota layanan. Tabel berikut hanya menjelaskan kuota yang berlaku untuk Amazon EKS. Anda dapat mengonfigurasi alarm yang memberi tahu Anda saat penggunaan mendekati kuota layanan. Untuk informasi selengkapnya, lihat Membuat CloudWatch alarm untuk memantau metrik penggunaan sumber daya Fargate.

Baru Akun AWS mungkin memiliki kuota awal yang lebih rendah yang dapat meningkat seiring waktu. Fargate terus memantau penggunaan akun di masing-masing akun Wilayah AWS, dan kemudian secara otomatis meningkatkan kuota berdasarkan penggunaan. Anda juga dapat meminta peningkatan kuota untuk nilai yang ditampilkan sebagai dapat disesuaikan. Untuk informasi selengkapnya, lihat Meminta peningkatan kuota di Panduan Pengguna Service Quotas.

Nama	Default	Dapat disesuaikan	Deskripsi
Jumlah sumber daya vCPU Sesuai Permintaan Fargate	6	Ya	Jumlah vCPU Fargate yang dapat berjalan bersamaan sebagai Fargate On-Demand di akun ini di Wilayah saat ini.

Note

Nilai default adalah kuota awal yang ditetapkan oleh AWS. Nilai default ini terpisah dari nilai kuota yang diterapkan aktual dan kuota layanan maksimum yang mungkin. Untuk informasi

selengkapnya, lihat Terminologi dalam Service Quotas di Panduan Pengguna Service Quotas.

Note

Fargate juga memberlakukan tugas Amazon ECS dan kuota tingkat peluncuran Amazon Pods EKS. Untuk informasi selengkapnya, lihat AWS Fargate membatasi kuota di panduan Amazon ECS.

Keamanan di Amazon EKS

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 di cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. Untuk Amazon EKS, AWS bertanggung jawab atas bidang Kubernetes kontrol, yang mencakup node dan `etcd` database bidang kontrol. Auditor pihak ketiga secara teratur menguji dan memverifikasi keefektifan keamanan kami sebagai bagian dari program kepatuhan AWS. Untuk mempelajari tentang program kepatuhan yang berlaku untuk Amazon EKS, lihat Layanan AWS yang Dicakup oleh Program Kepatuhan.
- Kemanan di dalam cloud – Tanggung jawab Anda meliputi area berikut.
  - Konfigurasi keamanan bidang data, mencakup konfigurasi grup keamanan yang memungkinkan lalu lintas berpindah dari bidang kendali Amazon EKS ke VPC pelanggan
  - Konfigurasi simpul dan kontainer itu sendiri
  - Sistem pengoperasiansimpul (termasuk pembaruan dan patch keamanan)
  - Perangkat lunak aplikasi terkait lainnya:
    - Menyiapkan dan mengelola kendali jaringan, misalnya aturan firewall
    - Mengelola identitas tingkat platform dan manajemen akses, baik dengan ataupun selain IAM
  - Sensitivitas data Anda, persyaratan perusahaan Anda, dan hukum dan peraturan yang berlaku

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

> **Note**
>
> Linuxkontainer terdiri dari grup kontrol (cgroups) dan ruang nama yang membantu membatasi apa yang dapat diakses kontainer, tetapi semua kontainer berbagi Linux kernel yang sama dengan instans Amazon EC2 host. Menjalankan kontainer sebagai pengguna asal (UID 0)

atau memberikan akses kontainer ke sumber daya host atau namespaces, seperti jaringan host atau namespace host PID, sangat tidak disarankan karena hal itu dapat mengurangi efektivitas isolasi yang disediakan oleh kontainer.

Topik

Penandatanganan sertifikat

API Kubernetes Sertifikat mengotomatiskan penyediaan kredensi X.509. API ini memiliki antarmuka baris perintah untuk klien Kubernetes API untuk meminta dan memperoleh sertifikat X.509 dari Certificate Authority (CA). Anda dapat menggunakan sumber daya `CertificateSigningRequest` (CSR) untuk meminta tanda tangan yang dilambangkan menandatangani sertifikat. Permintaan Anda disetujui atau ditolak sebelum ditandatangani. Kubernetesmendukung penandatangan bawaan dan penandatangan khusus dengan perilaku yang terdefinisi dengan baik. Dengan cara ini, klien dapat memprediksi apa yang terjadi pada CSR mereka. Untuk mempelajari lebih lanjut tentang penandatanganan sertifikat, lihat permintaan penandatanganan.

Salah satu penandatangan bawaan adalah`kubernetes.io/legacy-unknown`. `v1beta1`API sumber daya CSR menghormati penandatangan yang tidak dikenal lama ini. Namun, `v1` API CSR yang stabil tidak memungkinkan `signerName` untuk disetel ke`kubernetes.io/legacy-unknown`.

Versi Amazon EKS `1.21` dan sebelumnya mengizinkan `legacy-unknown` nilai seperti `signerName` di `v1beta1` CSR API. API ini memungkinkan Amazon EKS Certificate Authority (CA) untuk

menghasilkan sertifikat. Namun, dalam Kubernetes versi`1.22`, `v1beta1` CSR API digantikan oleh `v1` CSR API. API ini tidak mendukung signerName dari “legacy-unknown.” Jika ingin menggunakan Amazon EKS CA untuk membuat sertifikat di klaster, Anda harus menggunakan penandatangan khusus. Itu diperkenalkan dalam versi Amazon EKS`1.22`. Untuk menggunakan versi CSR `v1` API dan menghasilkan sertifikat baru, Anda harus memigrasikan manifes dan klien API yang ada. Sertifikat yang ada yang dibuat dengan `v1beta1` API yang ada valid dan berfungsi hingga sertifikat kedaluwarsa. Ini termasuk yang berikut:

- Distribusi kepercayaan: Tidak ada. Tidak ada kepercayaan atau distribusi standar untuk penandatangan ini dalam sebuah Kubernetes cluster.
- Subjek yang diizinkan: Apa saja
- Ekstensi x509 yang diizinkan: Menghormati subjectAltName dan ekstensi penggunaan kunci dan membuang ekstensi lainnya
- Penggunaan kunci yang diizinkan: Tidak boleh menyertakan penggunaan di luar ["encipherment kunci”, “tanda tangan digital”, “autentikasi server"]

  > **Note**
  >
  > Penandatanganan sertifikat klien tidak didukung.

- Kedaluwarsa/masa pakai sertifikat: 1 tahun (default dan maksimum)
- CA bit diizinkan/tidak diizinkan: Tidak diizinkan

Contoh pembuatan CSR dengan SignerName

Langkah-langkah ini menunjukkan cara menghasilkan sertifikat penyajian untuk `myserver.default.svc` menggunakan `signerName: beta.eks.amazonaws.com/app-serving` nama DNS. Gunakan ini sebagai panduan untuk lingkungan Anda sendiri.

1. Jalankan `openssl genrsa -out myserver.key 2048` perintah untuk menghasilkan kunci pribadi RSA.

   ```
   openssl genrsa -out myserver.key 2048
   ```

2. Jalankan perintah berikut untuk menghasilkan permintaan sertifikat.

```
openssl req -new -key myserver.key -out myserver.csr -subj "/
CN=myserver.default.svc"
```

3. Hasilkan `base64` nilai untuk permintaan CSR dan simpan dalam variabel untuk digunakan di langkah selanjutnya.

```
base_64=$(cat myserver.csr | base64 -w 0 | tr -d "\n")
```

4. Jalankan perintah berikut untuk membuat file bernama`mycsr.yaml`. Dalam contoh berikut, `beta.eks.amazonaws.com/app-serving` adalah`signerName`.

```
cat >mycsr.yaml <<EOF
apiVersion: certificates.k8s.io/v1
kind: CertificateSigningRequest
metadata:
  name: myserver
spec:
  request: $base_64
  signerName: beta.eks.amazonaws.com/app-serving
  usages:
    - digital signature
    - key encipherment
    - server auth
EOF
```

5. Kirim CSR.

```
kubectl apply -f mycsr.yaml
```

6. Menyetujui sertifikat penyajian.

```
kubectl certificate approve myserver
```

7. Verifikasi bahwa sertifikat telah dikeluarkan.

```
kubectl get csr myserver
```

Contoh output adalah sebagai berikut.

```
NAME       AGE     SIGNERNAME                              REQUESTOR
 CONDITION
myserver   3m20s   beta.eks.amazonaws.com/app-serving   kubernetes-admin
 Approved,Issued
```

8. Ekspor sertifikat yang dikeluarkan.

```
kubectl get csr myserver -o jsonpath='{.status.certificate}'| base64 -d
 > myserver.crt
```

Pertimbangan penandatanganan sertifikat sebelum meningkatkan klaster Anda ke 1,24 Kubernetes

Di dalam Kubernetes `1.23` dan sebelumnya, sertifikat `kubelet` penyajian dengan IP yang tidak dapat diverifikasi dan Nama Alternatif Subjek DNS (SAN) secara otomatis dikeluarkan dengan SAN yang tidak dapat diverifikasi. SAN dihilangkan dari sertifikat yang disediakan. Di klaster `1.24` dan yang lebih baru, sertifikat `kubelet` penyajian tidak dikeluarkan jika SAN tidak dapat diverifikasi. Ini mencegah `kubectl exec` dan `kubectl logs` perintah bekerja.

Sebelum memutakhirkan klaster ke`1.24`, tentukan apakah klaster Anda memiliki permintaan penandatanganan sertifikat (CSR) yang belum disetujui dengan menyelesaikan langkah-langkah berikut:

1. Jalankan perintah berikut.

```
kubectl get csr -A
```

Contoh output adalah sebagai berikut.

```
NAME        AGE   SIGNERNAME                      REQUESTOR
                        REQUESTEDDURATION   CONDITION
csr-7znmf   90m   kubernetes.io/kubelet-serving
 system:node:ip-192-168-42-149.region.compute.internal      <none>
 Approved
csr-9xx5q   90m   kubernetes.io/kubelet-serving
 system:node:ip-192-168-65-38.region.compute.internal      <none>
 Approved, Issued
```

Jika output yang dikembalikan menunjukkan CSR dengan `kubernetes.io/kubelet-serving`penandatangan yang `Approved` tetapi tidak `Issued` untuk node, maka Anda harus menyetujui permintaan tersebut.

2. Menyetujui CSR secara manual. Ganti `csr-7znmf` dengan nilai Anda sendiri.

```
kubectl certificate approve csr-7znmf
```

Untuk menyetujui CSR secara otomatis di masa mendatang, sebaiknya Anda menulis pengontrol persetujuan yang dapat secara otomatis memvalidasi dan menyetujui CSR yang berisi IP atau DNS SAN yang tidak dapat diverifikasi oleh Amazon EKS.

Identity and access management untuk Amazon EKS

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

Audiens

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

Pengguna layanan – Jika Anda menggunakan layanan Amazon EKS untuk melakukan tugas, maka administrator Anda akan memberikan kredensial dan izin yang dibutuhkan. Saat Anda menggunakan lebih banyak fitur di Amazon EKS untuk melakukan pekerjaan, Anda mungkin memerlukan izin tambahan. Memahami bagaimana akses dikelola dapat membantu Anda untuk meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur di Amazon EKS, lihat Menyelesaikan masalah IAM.

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

Administrator IAM – Jika Anda seorang administrator IAM, Anda mungkin ingin mempelajari detail tentang bagaimana Anda dapat menulis kebijakan untuk mengelola akses ke Amazon EKS. Untuk melihat contoh kebijakan berbasis identitas Amazon EKS yang dapat Anda gunakan di IAM, lihat Contoh kebijakan berbasis identitas Amazon EKS.

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 kredensyal Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas gabungan, 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 kredensyal Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang penggunaan metode yang disarankan untuk menandatangani permintaan sendiri, lihat Menandatangani permintaan AWS API di 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 Menggunakan autentikasi multi-faktor (MFA) di AWS 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 Anda. 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.

Pengguna dan grup IAM

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

Grup IAM adalah identitas yang menentukan kumpulan pengguna IAM. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin untuk beberapa pengguna sekaligus. Grup membuat izin lebih mudah dikelola untuk sekelompok besar pengguna. Misalnya, Anda dapat memiliki grup yang bernama 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, silakan lihat Kapan harus membuat pengguna IAM (bukan peran) 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. Anda dapat mengambil peran IAM untuk sementara AWS Management Console dengan beralih peran. 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 Menggunakan peran IAM dalam Panduan Pengguna IAM.

Peran IAM dengan kredensial sementara berguna dalam situasi berikut:

- Akses pengguna gabungan – Untuk menetapkan izin ke sebuah identitas gabungan, Anda dapat membuat peran dan menentukan izin untuk peran tersebut. Saat identitas terfederasi mengautentikasi, identitas tersebut akan dikaitkan dengan peran dan diberi izin yang ditentukan oleh peran tersebut. Untuk informasi tentang peran-peran untuk federasi, lihat Membuat peran untuk Penyedia Identitas pihak ketiga dalam Panduan Pengguna IAM. Jika Anda menggunakan Pusat Identitas IAM, Anda perlu mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah identitas tersebut diautentikasi, Pusat Identitas IAM mengorelasikan izin yang diatur ke peran dalam IAM. Untuk informasi tentang rangkaian izin, lihat Rangkaian 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 (pengguna utama tepercaya) dengan akun berbeda untuk mengakses sumber daya yang ada 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 kebijakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Bagaimana peran IAM berbeda dari kebijakan berbasis sumber daya dalam Panduan Pengguna IAM.
- Akses lintas layanan — Beberapa Layanan AWS menggunakan fitur lain Layanan AWS. Sebagai contoh, ketika 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. Saat Anda menggunakan beberapa layanan, Anda mungkin melakukan tindakan yang kemudian memulai 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 saat membuat permintaan FAS, lihat Teruskan sesi akses.
  - Peran layanan – Peran layanan adalah peran IAM yang diambil oleh layanan untuk melakukan tindakan atas nama Anda. Administrator IAM dapat membuat, memodifikasi, dan menghapus

peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat Membuat peran untuk mendelegasikan izin ke Layanan AWS dalam Panduan pengguna IAM.

- Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke. Layanan AWS Layanan dapat menggunakan 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 kredensyal sementara untuk aplikasi yang berjalan pada instans EC2 dan membuat atau permintaan API. AWS CLI AWS Cara ini lebih dianjurkan daripada menyimpan kunci akses dalam instans EC2. Untuk menetapkan AWS peran ke instans EC2 dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. Profil instans berisi peran dan memungkinkan program yang berjalan di instans EC2 mendapatkan kredensial sementara. Untuk informasi selengkapnya, lihat Menggunakan peran IAM untuk memberikan izin ke aplikasi yang berjalan di instans Amazon EC2 dalam Panduan Pengguna IAM.

Untuk mempelajari apakah kita harus menggunakan peran IAM atau pengguna IAM, lihat Kapan harus membuat peran IAM (bukan pengguna) dalam 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 dapat menentukan permintaan yang diizinkan atau ditolak. Sebagian besar kebijakan disimpan AWS sebagai dokumen JSON. Untuk informasi selengkapnya tentang struktur dan konten 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, pengguna utama manakah yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

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

Kebijakan IAM mendefinisikan izin untuk suatu tindakan terlepas dari metode yang Anda gunakan untuk operasi. Sebagai contoh, anggap saja 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 pengguna dan peran, di sumber daya mana, dan dengan ketentuan apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat Membuat kebijakan IAM dalam Panduan Pengguna IAM.

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan terkelola. 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 terkelola atau kebijakan inline, lihat Memilih antara kebijakan terkelola 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 yang dilampiri kebijakan tersebut, kebijakan ini menentukan jenis tindakan yang dapat dilakukan oleh pengguna utama tertentu di sumber daya tersebut dan apa ketentuannya. Anda harus menentukan pengguna utama 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 (ACL)

Daftar kontrol akses (ACL) mengendalikan pengguna utama mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACL sama dengan kebijakan berbasis sumber daya, meskipun tidak menggunakan format dokumen kebijakan JSON.

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACL. Untuk mempelajari ACL selengkapnya, silakan lihat Gambaran umum daftar kontrol akses (ACL) di Panduan Developer Layanan Penyimpanan Ringkas Amazon.

Tipe kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Tipe-tipe kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda berdasarkan tipe kebijakan yang lebih umum.

- Batasan izin – Batasan izin adalah fitur lanjutan di mana Anda menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas IAM (pengguna 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 secara eksplisit terhadap salah satu kebijakan ini akan mengesampingkan izin tersebut. Untuk informasi selengkapnya tentang batasan izin, lihat Batasan izin untuk entitas IAM dalam Panduan Pengguna IAM.
- Kebijakan kontrol layanan (SCP) — SCP 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 di sebuah organisasi, maka Anda dapat menerapkan kebijakan kontrol layanan (SCP) 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 Organisasi dan SCP, lihat Cara kerja SCP dalam Panduan Pengguna AWS Organizations .
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda teruskan sebagai parameter saat Anda membuat sesi sementara secara terprogram untuk peran atau pengguna gabungan. 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 membatalkan izin. Untuk informasi selengkapnya, lihat Kebijakan sesi dalam Panduan Pengguna IAM.

Beberapa jenis kebijakan

Ketika beberapa jenis kebijakan berlaku untuk sebuah 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 cara Amazon EKS bekerja sama dengan IAM

Sebelum Anda menggunakan IAM untuk mengelola akses ke Amazon EKS, Anda harus memahami fitur-fitur IAM apa yang tersedia untuk digunakan dengan Amazon EKS. Untuk mendapatkan tampilan tingkat tinggi tentang cara Amazon EKS dan AWS layanan lainnya bekerja dengan IAM, lihat AWS layanan yang bekerja dengan IAM di Panduan Pengguna IAM.

Topik

- Kebijakan berbasis identitas Amazon EKS
- kebijakan berbasis sumber daya Amazon EKS
- Otorisasi berdasarkan tanda Amazon EKS
- IAM role Amazon EKS

Kebijakan berbasis identitas Amazon EKS

Dengan kebijakan berbasis identitas IAM, Anda dapat menentukan apakah tindakan dan sumber daya diizinkan atau ditolak, serta persyaratan terkait diizinkan atau ditolak-nya tindakan tersebut. Amazon EKS mendukung tindakan, sumber daya, dan kunci syarat tertentu. Untuk mempelajari semua elemen yang Anda gunakan dalam kebijakan JSON, lihat Referensi elemen kebijakan IAM JSON dalam Panduan Pengguna IAM.

Tindakan

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, pengguna utama mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam 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 pelaksanaan operasi yang terkait.

Tindakan kebijakan di Amazon EKS menggunakan prefiks berikut sebelum tindakan: `eks:`. Misalnya, untuk memberikan izin kepada seseorang agar mendapatkan informasi deskriptif tentang klaster

Amazon EKS, Anda menyertakan tindakan `DescribeCluster` dalam kebijakan mereka. Pernyataan kebijakan harus memuat elemen `Action` atau `NotAction`.

Untuk menetapkan beberapa tindakan dalam satu pernyataan, pisahkan dengan koma seperti berikut:

```
"Action": ["eks:action1", "eks:action2"]
```

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

```
"Action": "eks:Describe*"
```

Untuk melihat daftar tindakan Amazon EKS, lihat Tindakan yang ditentukan oleh Amazon Elastic Kubernetes Service di Referensi Otorisasi Layanan.

Sumber daya

Administrator dapat menggunakan kebijakan AWS JSON untuk menentukan siapa yang memiliki akses ke apa. Yaitu, pengguna utama mana yang dapat melakukan tindakan pada sumber daya apa, dan dalam kondisi apa.

Elemen kebijakan JSON `Resource` menentukan objek atau beberapa 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 mengindikasikan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

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

Sumber daya cluster Amazon EKS memiliki ARN berikut.

```
arn:aws:eks:region-code:account-id:cluster/cluster-name
```

Untuk informasi selengkapnya tentang format ARN, lihat nama sumber daya Amazon (ARN) dan ruang nama AWS layanan.

Misalnya, untuk menentukan cluster dengan nama *my-cluster* dalam pernyataan Anda, gunakan ARN berikut:

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/my-cluster"
```

Untuk menentukan semua cluster milik akun tertentu dan Wilayah AWS, gunakan wildcard (*):

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/*"
```

Beberapa tindakan Amazon EKS, seperti untuk membuat sumber daya, tidak dapat dilakukan pada sumber daya tertentu. Dalam kasus tersebut, Anda harus menggunakan wildcard (*).

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

Untuk melihat daftar jenis resource Amazon EKS dan ARNnya, lihat Sumber daya yang ditentukan oleh Amazon Elastic Kubernetes Service di Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap resource, lihat Tindakan yang ditentukan oleh Amazon Elastic Kubernetes Service.

Kunci syarat

Amazon EKS menentukan set kunci syaratnya sendiri dan juga mendukung penggunaan beberapa kunci syarat global. Untuk melihat semua kunci kondisi AWS global, lihat Kunci Konteks Kondisi AWS Global di Panduan Pengguna IAM.

Anda dapat menyetel kunci kondisi saat mengaitkan OpenID Connect penyedia ke klaster Anda. Untuk informasi selengkapnya, lihat Contoh kebijakan IAM.

Semua tindakan Amazon EC2 mendukung kunci syarat `aws:RequestedRegion` dan `ec2:Region`. Untuk informasi selengkapnya, lihat Contoh: Membatasi Akses ke Spesifik Wilayah AWS.

Untuk daftar kunci kondisi Amazon EKS, lihat Ketentuan yang ditentukan oleh Amazon Elastic Kubernetes Service di Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat digunakan untuk menggunakan kunci kondisi, lihat Tindakan yang ditentukan oleh Amazon Elastic Kubernetes Service.

Contoh-contoh

Untuk melihat contoh kebijakan berbasis identitas Amazon EKS, lihat Contoh kebijakan berbasis identitas Amazon EKS.

Saat Anda membuat klaster Amazon EKS, prinsipal IAM yang membuat klaster secara otomatis diberikan `system:masters` izin dalam konfigurasi kontrol akses berbasis peran (RBAC) klaster di bidang kontrol Amazon EKS. Prinsipal ini tidak muncul dalam konfigurasi yang terlihat, jadi pastikan untuk melacak prinsipal mana yang awalnya membuat cluster. Untuk memberikan kepala sekolah IAM tambahan kemampuan untuk berinteraksi dengan cluster Anda, edit bagian `aws-auth ConfigMap` dalam Kubernetes dan buat Kubernetes `rolebinding` atau `clusterrolebinding` dengan nama `group` yang Anda tentukan di. `aws-auth ConfigMap`

Untuk informasi lebih lanjut tentang bekerja dengan ConfigMap, lihatBerikan akses ke Kubernetes API .

kebijakan berbasis sumber daya Amazon EKS

Amazon EKS tidak mendukung kebijakan berbasis sumber daya.

Otorisasi berdasarkan tanda Amazon EKS

Anda dapat melampirkan tanda ke sumber daya Amazon EKS atau meneruskan tanda dalam sebuah permintaan ke Amazon EKS. Untuk mengendalikan akses berdasarkan tanda, Anda dapat memberikan informasi tentang tanda di elemen syarat kebijakan menggunakan kunci kondisi `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, atau `aws:TagKeys`. Untuk informasi selengkapnya tentang penandaan sumber daya Amazon EKS, lihat Menandai sumber daya Amazon EKS Anda. Untuk informasi selengkapnya tentang tindakan yang dapat Anda gunakan dengan tag dalam kunci kondisi, lihat Tindakan yang ditentukan oleh Amazon EKS di Referensi Otorisasi Layanan.

IAM role Amazon EKS

Peran IAM adalah entitas dalam AWS akun Anda yang memiliki izin tertentu.

Menggunakan kredensial sementara dengan Amazon EKS

Anda dapat menggunakan kredensial sementara untuk masuk dengan gabungan, menjalankan IAM role, atau menjalankan peran lintas akun. Anda memperoleh kredensi keamanan sementara dengan memanggil operasi AWS STS API seperti AssumeRoleatau. GetFederationToken

Amazon EKS mendukung penggunaan kredensial sementara.

Peran terkait layanan

Peran terkait AWS layanan memungkinkan layanan mengakses sumber daya di layanan lain untuk menyelesaikan tindakan atas nama Anda. Peran tertaut layanan muncul di akun IAM Anda dan dimiliki oleh layanan tersebut. Administrator dapat melihat tetapi tidak dapat mengedit izin untuk peran terkait layanan.

Amazon EKS mendukung peran tertaut-layanan. Untuk informasi selengkapnya tentang cara membuat atau mengelola peran tertaut layanan Amazon EKS, lihat Menggunakan peran tertaut layanan untuk Amazon EKS.

Peran layanan

Fitur ini memungkinkan layanan untuk menerima peran layanan atas nama Anda. Peran ini mengizinkan layanan untuk mengakses sumber daya di layanan lain untuk menyelesaikan tindakan atas nama Anda. Peran layanan muncul di akun IAM Anda dan dimiliki oleh akun tersebut. Ini berarti administrator IAM dapat mengubah izin untuk peran ini. Namun, hal itu dapat merusak fungsionalitas layanan.

Amazon EKS mendukung peran layanan. Untuk informasi lebih lanjut, lihat IAM role klaster Amazon EKS dan IAM role simpul Amazon EKS.

Memilih IAM role di Amazon EKS

Saat Anda membuat sumber daya kluster di Amazon EKS, Anda harus memilih peran untuk memungkinkan Amazon EKS mengakses beberapa AWS sumber daya lain atas nama Anda. Jika sebelumnya Anda telah membuat peran layanan, maka Amazon EKS akan memberi Anda daftar peran untuk dipilih. Penting untuk memilih peran yang memiliki kebijakan yang dikelola oleh Amazon EKS. Lihat informasi yang lebih lengkap di Periksa apakah peran klaster sudah ada dan Periksa apakah peran simpul sudah ada.

Contoh kebijakan berbasis identitas Amazon EKS

Secara default, pengguna IAM dan IAM role tidak memiliki izin untuk membuat atau memodifikasi sumber daya Amazon EKS. Mereka juga tidak dapat melakukan tugas menggunakan AWS Management Console, AWS CLI, atau AWS API. Administrator IAM harus membuat kebijakan IAM yang memberikan izin kepada pengguna dan peran untuk melakukan operasi API tertentu pada

sumber daya yang diperlukan. Administrator kemudian harus melampirkan kebijakan tersebut ke pengguna IAM atau grup yang memerlukan izin tersebut.

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

Saat Anda membuat klaster Amazon EKS, prinsipal IAM yang membuat klaster secara otomatis diberikan `system:masters` izin dalam konfigurasi kontrol akses berbasis peran (RBAC) klaster di bidang kontrol Amazon EKS. Prinsipal ini tidak muncul dalam konfigurasi yang terlihat, jadi pastikan untuk melacak prinsipal mana yang awalnya membuat cluster. Untuk memberikan kepala sekolah IAM tambahan kemampuan untuk berinteraksi dengan cluster Anda, edit bagian `aws-auth` `ConfigMap` dalam Kubernetes dan buat Kubernetes `rolebinding` atau `clusterrolebinding` dengan nama `group` yang Anda tentukan di. `aws-auth` `ConfigMap`

Untuk informasi lebih lanjut tentang bekerja dengan ConfigMap, lihatBerikan akses ke Kubernetes API .

Topik

- Praktik terbaik kebijakan
- Menggunakan konsol Amazon EKS
- Izinkan pengguna IAM untuk melihat izin mereka sendiri
- Buat Kubernetes cluster di AWS Cloud
- Buat Kubernetes cluster lokal di Outpost
- Perbarui Kubernetes klaster
- Buat daftar atau deskripsikan semua klaster

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus sumber daya Amazon EKS di akun Anda. Tindakan ini dikenai biaya untuk Akun AWS Anda. 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 pekerjaan di 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 melakukan ini dengan menentukan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, juga dikenal sebagai izin hak akses paling rendah. Untuk informasi selengkapnya tentang cara menggunakan IAM untuk menerapkan izin, lihat Kebijakan dan izin di IAM di Panduan Pengguna IAM.
- Gunakan kondisi dalam kebijakan IAM untuk membatasi akses lebih lanjut – Anda dapat menambahkan 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: Syarat di Panduan Pengguna IAM.
- Menggunakan IAM Access Analyzer untuk memvalidasi kebijakan IAM Anda guna memastikan izin yang aman dan berfungsi – 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 Analizer Akses IAM di 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 mewajibkan MFA saat operasi API dipanggil, tambahkan kondisi MFA pada kebijakan Anda. Untuk informasi selengkapnya, lihat Mengonfigurasi akses API yang dilindungi MFA di Panduan Pengguna IAM.

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

Menggunakan konsol Amazon EKS

Untuk mengakses konsol Amazon EKS, prinsipal IAM, harus memiliki set izin minimum. Izin ini memungkinkan kepala sekolah untuk membuat daftar dan melihat detail tentang sumber daya Amazon EKS di AWS akun Anda. Jika Anda membuat kebijakan berbasis identitas yang lebih ketat

daripada izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana dimaksud untuk prinsipal dengan kebijakan yang dilampirkan padanya.

Untuk memastikan bahwa prinsipal IAM Anda masih dapat menggunakan konsol Amazon EKS, buat kebijakan dengan nama unik Anda sendiri, seperti. `AmazonEKSAdminPolicy` Lampirkan kebijakan ke kepala sekolah. Untuk informasi lebih lanjut, lihat Menambahkan dan menghapus izin identitas IAM dalam Panduan Pengguna IAM.

> **Important**
>
> Contoh kebijakan berikut memungkinkan prinsipal untuk melihat informasi pada tab Konfigurasi di konsol. Untuk melihat informasi pada tab Ikhtisar dan Sumber Daya di AWS Management Console, prinsipal juga memerlukan Kubernetes izin. Untuk informasi selengkapnya, lihat Izin yang diperlukan.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "eks.amazonaws.com"
                }
            }
        }
    ]
}
```

Anda tidak perlu mengizinkan izin konsol minimum untuk prinsipal yang melakukan panggilan hanya ke atau API. AWS CLI AWS Sebagai alternatif, hanya izinkan akses ke tindakan yang cocok dengan operasi API yang sedang Anda coba lakukan.

Izinkan pengguna IAM untuk melihat izin mereka sendiri

Contoh ini menunjukkan cara membuat kebijakan yang mengizinkan para 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": "*"
        }
    ]
```

```
}
```

Buat Kubernetes cluster di AWS Cloud

Wilayah AWS Anda dapat mengganti Wilayah AWS dengan Wilayah AWS yang Anda inginkan untuk membuat cluster di. Jika Anda melihat peringatan yang mengatakan Tindakan dalam kebijakan Anda tidak mendukung izin tingkat sumber daya dan mengharuskan Anda untuk memilih AWS Management Console, tindakan tersebut dapat `All resources` diabaikan dengan aman. Jika akun Anda sudah memiliki *AWSServiceRoleForAmazonEKS*peran, Anda dapat menghapus `iam:CreateServiceLinkedRole` tindakan dari kebijakan. Jika Anda pernah membuat kluster Amazon EKS di akun Anda, maka peran ini sudah ada, kecuali Anda menghapusnya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:CreateCluster",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/
eks.amazonaws.com/AWSServiceRoleForAmazonEKS",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "iam:AWSServiceName": "eks"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/cluster-role-name"
        }
    ]
}
```

Buat Kubernetes cluster lokal di Outpost

Wilayah AWS Anda dapat mengganti Wilayah AWS dengan Wilayah AWS yang Anda inginkan untuk membuat cluster di. Jika Anda melihat peringatan yang mengatakan Tindakan dalam kebijakan Anda tidak mendukung izin tingkat sumber daya dan mengharuskan Anda untuk memilih AWS Management Console, tindakan tersebut dapat `All resources` diabaikan dengan aman. Jika akun Anda sudah memiliki `AWSServiceRoleForAmazonEKSLocalOutpost` peran, Anda dapat menghapus `iam:CreateServiceLinkedRole` tindakan dari kebijakan. Jika Anda pernah membuat kluster lokal Amazon EKS di Outpost di akun Anda, maka peran ini sudah ada, kecuali Anda menghapusnya.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:CreateCluster",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        },
        {
            "Action": [
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "iam:GetRole"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/outposts.eks-
local.amazonaws.com/AWSServiceRoleForAmazonEKSLocalOutpost"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole",
                "iam:ListAttachedRolePolicies"
            ]
            "Resource": "arn:aws:iam::111122223333:role/cluster-role-name"
        },
```

```
        {
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:TagInstanceProfile",
                "iam:AddRoleToInstanceProfile",
                "iam:GetInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile"
            ],
            "Resource": "arn:aws:iam::*:instance-profile/eks-local-*",
            "Effect": "Allow"
        },
    ]
}
```

Perbarui Kubernetes klaster

Wilayah AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:UpdateClusterVersion",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        }
    ]
}
```

Buat daftar atau deskripsikan semua klaster

Kebijakan contoh ini mencakup izin minimum yang diperlukan untuk mencantumkan dan menjelaskan semua klaster di akun Anda. Seorang kepala sekolah IAM harus dapat membuat daftar dan mendeskripsikan cluster untuk menggunakan perintah. `update-kubeconfig` AWS CLI

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "eks:DescribeCluster",
                "eks:ListClusters"
            ],
            "Resource": "*"
        }
    ]
}
```

Menggunakan peran tertaut layanan untuk Amazon EKS

Amazon Elastic Kubernetes Service AWS Identity and Access Management menggunakan peran terkait layanan (IAM). Peran tertaut layanan adalah jenis IAM role unik yang terkait langsung dengan Amazon EKS. Peran terkait layanan telah ditentukan sebelumnya oleh Amazon EKS dan mencakup semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda.

Topik

- Menggunakan peran untuk klaster Amazon EKS
- Menggunakan peran untuk grup simpul Amazon EKS
- Menggunakan peran untuk profil Amazon EKS Fargate
- Menggunakan peran untuk menghubungkan Kubernetes cluster ke Amazon EKS
- Menggunakan peran untuk kluster lokal Amazon EKS di Outpost

Menggunakan peran untuk klaster Amazon EKS

Amazon Elastic Kubernetes Service AWS Identity and Access Management menggunakan peran terkait layanan (IAM). Peran tertaut layanan adalah jenis IAM role unik yang terkait langsung dengan Amazon EKS. Peran terkait layanan telah ditentukan sebelumnya oleh Amazon EKS dan mencakup semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda.

Peran tertaut layanan memudahkan pengaturan Amazon EKS karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Amazon EKS menentukan izin atas peran tertaut layanan, dan jika tidak ada ketentuan lain, hanya Amazon EKS yang dapat menjalankan perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, dan kebijakan izin tersebut tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran tertaut layanan jika telah menghapus sumber daya terkait. Ini dapat melindungi sumber daya Amazon EKS karena Anda tidak dapat secara ceroboh menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran tertaut layanan, lihat layanan AWS yang bekerja bersama IAM dan mencari layanan yang memuat Ya di dalam kolom Peran tertaut layanan. Pilih Ya bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Izin peran tertaut layanan untuk Amazon EKS

Amazon EKS menggunakan peran tertaut layanan yang diberi nama `AWSServiceRoleForAmazonEKS` – Peran ini mengizinkan Amazon EKS untuk mengelola klaster di akun Anda. Kebijakan terlampir memungkinkan peran untuk mengelola sumber daya berikut: antarmuka jaringan, grup keamanan, log, dan VPC.

> **Note**
>
> Peran tertaut layanan `AWSServiceRoleForAmazonEKS` berbeda dari peran yang diperlukan untuk pembuatan klaster. Untuk informasi selengkapnya, lihat IAM role klaster Amazon EKS.

Peran tertaut layanan `AWSServiceRoleForAmazonEKS` mempercayai layanan berikut untuk mengambil peran tersebut:

- `eks.amazonaws.com`

Kebijakan izin peran mengizinkan Amazon EKS untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- `AmazonEKSServiceRolePolicy`

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran tertaut layanan. Untuk informasi selengkapnya, lihat Izin peran tertaut layanan dalam Panduan Pengguna IAM.

Membuat peran tertaut layanan untuk Amazon EKS

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda membuat klaster di AWS Management Console, API AWS CLI, atau AWS API, Amazon EKS membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran tertaut layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat peran tersebut di akun Anda. Ketika Anda membuat klaster, Amazon EKS membuatkan peran tertaut layanan lagi untuk Anda.

Mengedit Peran tertaut layanan untuk Amazon EKS

Amazon EKS tidak mengizinkan Anda untuk mengedit peran tertaut layanan `AWSServiceRoleForAmazonEKS`. Setelah membuat peran tertaut layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit deskripsi peran ini menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit peran tertaut layanan dalam Panduan Pengguna IAM.

Menghapus peran tertaut layanan untuk Amazon EKS

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran tertaut layanan, kami sarankan supaya Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas tak terpakai yang tidak dipantau atau dipelihara secara aktif. Namun, Anda harus membersihkan peran tertaut-layanan sebelum dapat menghapusnya secara manual.

Membersihkan peran tertaut-layanan

Sebelum dapat menggunakan IAM untuk menghapus peran tertaut-layanan, Anda harus terlebih dahulu menghapus semua sumber daya yang digunakan oleh peran tersebut.

> **Note**
>
> Jika layanan Amazon EKS menggunakan peran saat Anda mencoba untuk menghapus sumber daya, maka penghapusan tersebut kemungkinan gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba lagi.

Untuk menghapus sumber daya Amazon EKS yang digunakan oleh peran **`AWSServiceRoleForAmazonEKS`**.

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel navigasi sebelah kiri, pilih Klaster.
3. Jika klaster Anda memiliki grup simpul atau profil Fargate, Anda harus menghapusnya sebelum dapat menghapus klaster. Untuk informasi lebih lanjut, lihat Menghapus grup simpul terkelola dan Menghapus profil Fargate.
4. Pada halaman Klaster, pilih klaster yang ingin Anda hapus dan pilih Hapus.

5. Ketik nama klaster di jendela konfirmasi penghapusan, dan kemudian pilih Hapus.
6. Ulangi prosedur ini untuk klaster lain di akun Anda. Tunggu sampai semua operasi penghapusan selesai.

Hapus peran tertaut layanan secara manual

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus peran `AWSServiceRoleForAmazonEKS` terkait layanan. Untuk informasi selengkapnya, lihat Menghapus peran tertaut layanan dalam Panduan Pengguna IAM.

Wilayah yang didukung untuk peran tertaut-layanan Amazon EKS

Amazon EKS mendukung penggunaan peran tertaut-layanan di semua wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihat titik akhir dan kuota Amazon EKS.

Menggunakan peran untuk grup simpul Amazon EKS

Amazon EKS menggunakan AWS Identity and Access Management peran terkait layanan (IAM). Peran tertaut layanan adalah jenis IAM role unik yang terkait langsung dengan Amazon EKS. Peran terkait layanan telah ditentukan sebelumnya oleh Amazon EKS dan mencakup semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda.

Peran tertaut layanan memudahkan pengaturan Amazon EKS karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Amazon EKS menentukan izin atas peran tertaut layanan, dan jika tidak ada ketentuan lain, hanya Amazon EKS yang dapat menjalankan perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, dan kebijakan izin tersebut tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran tertaut layanan jika telah menghapus sumber daya terkait. Ini dapat melindungi sumber daya Amazon EKS karena Anda tidak dapat secara ceroboh menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran tertaut layanan, lihat layanan AWS yang bekerja bersama IAM dan mencari layanan yang memuat Ya di dalam kolom Peran tertaut layanan. Pilih Ya bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Izin peran tertaut layanan untuk Amazon EKS

Amazon EKS menggunakan peran tertaut layanan yang diberi nama `AWSServiceRoleForAmazonEKSNodegroup` – Peran tersebut mengizinkan Amazon EKS untuk

mengelola grup simpul di akun Anda. Kebijakan terlampir mengizinkan peran untuk mengelola sumber daya berikut: Grup Auto Scaling, grup keamanan, templat peluncuran, dan profil instans IAM.

Peran tertaut layanan `AWSServiceRoleForAmazonEKSNodegroup` mempercayai layanan berikut untuk mengambil peran tersebut:

- `eks-nodegroup.amazonaws.com`

Kebijakan izin peran mengizinkan Amazon EKS untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- `AWSServiceRoleForAmazonEKSNodegroup`

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran tertaut layanan. Untuk informasi selengkapnya, lihat Izin peran tertaut layanan dalam Panduan Pengguna IAM.

Membuat peran tertaut layanan untuk Amazon EKS

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda CreateNodegroup berada di AWS Management Console, the AWS CLI, atau AWS API, Amazon EKS membuat peran terkait layanan untuk Anda.

> **Important**
>
> Peran tertaut layanan ini dapat muncul di akun Anda jika Anda menyelesaikan tindakan di layanan lain yang menggunakan fitur yang disupport oleh peran ini. Jika Anda menggunakan layanan Amazon EKS sebelum 1 Januari 2017, ketika mulai mendukung peran terkait layanan, maka Amazon EKS membuat AWSServiceRoleForAmazonEKSNodegroup peran di akun Anda. Untuk mempelajari lebih lanjut, lihat Peran baru muncul di akun IAM saya.

Membuat peran terkait layanan di Amazon EKS (API)AWS

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda membuat grup node terkelola di AWS Management Console, API AWS CLI, atau AWS API, Amazon EKS membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran terkait layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Ketika Anda membuat grup simpul terkelola lainnya, Amazon EKS membuatkan peran tertaut layanan lagi untuk Anda.

Mengedit peran tertaut-layanan untuk Amazon EKS

Amazon EKS tidak mengizinkan Anda untuk mengedit peran tertaut layanan `AWSServiceRoleForAmazonEKSNodegroup`. Setelah membuat peran tertaut layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit deskripsi peran ini menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit peran tertaut layanan dalam Panduan Pengguna IAM.

Menghapus peran tertaut layanan untuk Amazon EKS

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran tertaut layanan, kami sarankan supaya Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas tak terpakai yang tidak dipantau atau dipelihara secara aktif. Namun, Anda harus membersihkan peran tertaut-layanan sebelum dapat menghapusnya secara manual.

Membersihkan peran tertaut-layanan

Sebelum dapat menggunakan IAM untuk menghapus peran tertaut-layanan, Anda harus terlebih dahulu menghapus semua sumber daya yang digunakan oleh peran tersebut.

> **Note**
>
> Jika layanan Amazon EKS menggunakan peran saat Anda mencoba untuk menghapus sumber daya, maka penghapusan tersebut kemungkinan gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba lagi.

Untuk menghapus sumber daya Amazon EKS yang digunakan oleh peran **`AWSServiceRoleForAmazonEKSNodegroup`**.

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pada panel navigasi sebelah kiri, pilih Klaster.
3. Pilih tab Compute.
4. Di bagian Node groups, pilih grup node yang akan dihapus.
5. Ketik nama grup simpul di jendela konfirmasi penghapusan, dan kemudian pilih Hapus.

6. Ulangi prosedur ini untuk grup simpul lainnya dalam klaster. Tunggu sampai semua operasi penghapusan selesai.

Hapus peran tertaut layanan secara manual

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus peran `AWSServiceRoleForAmazonEKSNodegroup` terkait layanan. Untuk informasi selengkapnya, lihat Menghapus peran tertaut layanan dalam Panduan Pengguna IAM.

Wilayah yang didukung untuk peran tertaut-layanan Amazon EKS

Amazon EKS mendukung penggunaan peran tertaut-layanan di semua wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihat titik akhir dan kuota Amazon EKS.

Menggunakan peran untuk profil Amazon EKS Fargate

Amazon EKS menggunakan AWS Identity and Access Management peran terkait layanan (IAM). Peran tertaut layanan adalah jenis IAM role unik yang terkait langsung dengan Amazon EKS. Peran terkait layanan telah ditentukan sebelumnya oleh Amazon EKS dan mencakup semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda.

Peran tertaut layanan memudahkan pengaturan Amazon EKS karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Amazon EKS menentukan izin atas peran tertaut layanan, dan jika tidak ada ketentuan lain, hanya Amazon EKS yang dapat menjalankan perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, dan kebijakan izin tersebut tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran tertaut layanan jika telah menghapus sumber daya terkait. Ini dapat melindungi sumber daya Amazon EKS karena Anda tidak dapat secara ceroboh menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran tertaut layanan, lihat layanan AWS yang bekerja bersama IAM dan mencari layanan yang memuat Ya di dalam kolom Peran tertaut layanan. Pilih Ya bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Izin peran tertaut layanan untuk Amazon EKS

Amazon EKS menggunakan peran terkait layanan bernama `AWSServiceRoleForAmazonEKSForFargate` — Peran ini memungkinkan Amazon EKS

Fargate untuk mengonfigurasi jaringan VPC yang diperlukan untuk Fargate. Pods Kebijakan terlampir memungkinkan peran untuk membuat dan menghapus antarmuka jaringan elastis dan menggambarkan Antarmuka dan sumber daya jaringan elastis.

Peran tertaut layanan `AWSServiceRoleForAmazonEKSForFargate` mempercayai layanan berikut untuk mengambil peran tersebut:

- `eks-fargate.amazonaws.com`

Kebijakan izin peran mengizinkan Amazon EKS untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- `AmazonEKSForFargateServiceRolePolicy`

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran tertaut layanan. Untuk informasi selengkapnya, lihat Izin peran tertaut layanan dalam Panduan Pengguna IAM.

Membuat peran tertaut layanan untuk Amazon EKS

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda membuat profil Fargate di, API AWS Management Console, atau AWS API AWS CLI, Amazon EKS membuat peran terkait layanan untuk Anda.

> **Important**
>
> Peran tertaut layanan ini dapat muncul di akun Anda jika Anda menyelesaikan tindakan di layanan lain yang menggunakan fitur yang disupport oleh peran ini. Jika Anda menggunakan layanan Amazon EKS sebelum 13 Desember 2019, ketika mulai mendukung peran terkait layanan, maka Amazon EKS membuat AWSServiceRoleForAmazonEKSForFargate peran tersebut di akun Anda. Untuk mempelajari lebih lanjut, lihat Peran baru muncul di akun IAM saya.

Membuat peran terkait layanan di Amazon EKS (API)AWS

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda membuat profil Fargate di, API AWS Management Console, atau AWS API AWS CLI, Amazon EKS membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran terkait layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Ketika Anda membuat grup simpul terkelola lainnya, Amazon EKS membuatkan peran tertaut layanan lagi untuk Anda.

Mengedit peran tertaut-layanan untuk Amazon EKS

Amazon EKS tidak mengizinkan Anda untuk mengedit peran tertaut layanan `AWSServiceRoleForAmazonEKSForFargate`. Setelah membuat peran tertaut layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit deskripsi peran ini menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit peran tertaut layanan dalam Panduan Pengguna IAM.

Menghapus peran tertaut layanan untuk Amazon EKS

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran tertaut layanan, kami sarankan supaya Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas tak terpakai yang tidak dipantau atau dipelihara secara aktif. Namun, Anda harus membersihkan peran tertaut-layanan sebelum dapat menghapusnya secara manual.

Membersihkan peran tertaut-layanan

Sebelum dapat menggunakan IAM untuk menghapus peran tertaut-layanan, Anda harus terlebih dahulu menghapus semua sumber daya yang digunakan oleh peran tersebut.

> **Note**
>
> Jika layanan Amazon EKS menggunakan peran saat Anda mencoba untuk menghapus sumber daya, maka penghapusan tersebut kemungkinan gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba lagi.

Untuk menghapus sumber daya Amazon EKS yang digunakan oleh peran **`AWSServiceRoleForAmazonEKSForFargate`**.

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pada panel navigasi sebelah kiri, pilih Klaster.
3. Pada halaman Klaster, pilih klaster Anda.
4. Pilih tab Compute.

5. Jika ada profil Fargate di bagian profil Fargate, pilih masing-masing satu per satu, lalu pilih Hapus.
6. Ketik nama profil di jendela konfirmasi penghapusan, dan kemudian pilih Hapus.
7. Ulangi prosedur ini untuk profil Fargate lainnya di dalam klaster dan untuk klaster lainnya di akun Anda.

Untuk menghapus peran tertaut-layanan secara manual

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus peran AWSServiceRoleForAmazonEKSForFargate terkait layanan. Untuk informasi selengkapnya, lihat Menghapus peran tertaut layanan dalam Panduan Pengguna IAM.

Wilayah yang didukung untuk peran tertaut-layanan Amazon EKS

Amazon EKS mendukung penggunaan peran tertaut-layanan di semua wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihat titik akhir dan kuota Amazon EKS.

Menggunakan peran untuk menghubungkan Kubernetes cluster ke Amazon EKS

Amazon EKS menggunakan AWS Identity and Access Management peran terkait layanan (IAM). Peran tertaut layanan adalah jenis IAM role unik yang terkait langsung dengan Amazon EKS. Peran terkait layanan telah ditentukan sebelumnya oleh Amazon EKS dan mencakup semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda.

Peran tertaut layanan memudahkan pengaturan Amazon EKS karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Amazon EKS menentukan izin atas peran tertaut layanan, dan jika tidak ada ketentuan lain, hanya Amazon EKS yang dapat menjalankan perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, dan kebijakan izin tersebut tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran tertaut layanan jika telah menghapus sumber daya terkait. Ini dapat melindungi sumber daya Amazon EKS karena Anda tidak dapat secara ceroboh menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran tertaut layanan, lihat layanan AWS yang bekerja bersama IAM dan mencari layanan yang memuat Ya di dalam kolom Peran tertaut layanan. Pilih Ya bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Izin peran tertaut layanan untuk Amazon EKS

Amazon EKS menggunakan peran terkait layanan bernama `AWSServiceRoleForAmazonEKSConnector` — Peran ini memungkinkan Amazon EKS menghubungkan Kubernetes kluster. Kebijakan terlampir memungkinkan peran mengelola sumber daya yang diperlukan untuk terhubung ke Kubernetes klaster terdaftar Anda.

Peran tertaut layanan `AWSServiceRoleForAmazonEKSConnector` mempercayai layanan berikut untuk mengambil peran tersebut:

- `eks-connector.amazonaws.com`

Kebijakan izin peran mengizinkan Amazon EKS untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- `AmazonEKSConnectorServiceRolePolicy`

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran tertaut layanan. Untuk informasi selengkapnya, lihat Izin peran tertaut layanan dalam Panduan Pengguna IAM.

Membuat peran tertaut layanan untuk Amazon EKS

Anda tidak perlu membuat peran terkait layanan secara manual untuk menghubungkan kluster. Saat Anda menghubungkan cluster di AWS Management Console, the, AWS CLI`eksctl`, atau AWS API, Amazon EKS membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran terkait layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat kembali peran tersebut di akun Anda. Saat Anda menghubungkan kluster, Amazon EKS membuat peran terkait layanan untuk Anda lagi.

Mengedit Peran tertaut layanan untuk Amazon EKS

Amazon EKS tidak mengizinkan Anda untuk mengedit peran tertaut layanan `AWSServiceRoleForAmazonEKSConnector`. Setelah membuat peran tertaut layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat mengedit deskripsi peran ini menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit peran tertaut layanan dalam Panduan Pengguna IAM.

Menghapus peran tertaut layanan untuk Amazon EKS

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran tertaut layanan, kami sarankan supaya Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas tak terpakai yang tidak dipantau atau dipelihara secara aktif. Namun, Anda harus membersihkan peran tertaut-layanan sebelum dapat menghapusnya secara manual.

Membersihkan peran tertaut-layanan

Sebelum dapat menggunakan IAM untuk menghapus peran tertaut-layanan, Anda harus terlebih dahulu menghapus semua sumber daya yang digunakan oleh peran tersebut.

> **Note**
>
> Jika layanan Amazon EKS menggunakan peran saat Anda mencoba untuk menghapus sumber daya, maka penghapusan tersebut kemungkinan gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba lagi.

Untuk menghapus sumber daya Amazon EKS yang digunakan oleh peran **`AWSServiceRoleForAmazonEKSConnector`**.

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pada panel navigasi sebelah kiri, pilih Klaster.
3. Pada halaman Klaster, pilih klaster Anda.
4. Pilih tab Deregister dan kemudian pilih tab Ok.

Menghapus peran tertaut layanan secara manual

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus peran AWSServiceRoleForAmazonEKSConnector terkait layanan. Untuk informasi selengkapnya, lihat Menghapus peran tertaut layanan dalam Panduan Pengguna IAM.

Menggunakan peran untuk kluster lokal Amazon EKS di Outpost

Amazon Elastic Kubernetes Service AWS Identity and Access Management menggunakan peran terkait layanan (IAM). Peran tertaut layanan adalah jenis IAM role unik yang terkait langsung dengan Amazon EKS. Peran terkait layanan telah ditentukan sebelumnya oleh Amazon EKS dan mencakup semua izin yang diperlukan layanan untuk memanggil AWS layanan lain atas nama Anda.

Peran tertaut layanan memudahkan pengaturan Amazon EKS karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. Amazon EKS menentukan izin atas peran tertaut layanan, dan jika tidak ada ketentuan lain, hanya Amazon EKS yang dapat menjalankan perannya. Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin, dan kebijakan izin tersebut tidak dapat dilampirkan ke entitas IAM lainnya.

Anda dapat menghapus peran tertaut layanan jika telah menghapus sumber daya terkait. Ini dapat melindungi sumber daya Amazon EKS karena Anda tidak dapat secara ceroboh menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran tertaut layanan, lihat layanan AWS yang bekerja bersama IAM dan mencari layanan yang memuat Ya di dalam kolom Peran tertaut layanan. Pilih Ya bersama tautan untuk melihat dokumentasi peran tertaut layanan untuk layanan tersebut.

Izin peran tertaut layanan untuk Amazon EKS

Amazon EKS menggunakan peran terkait layanan bernama `AWSServiceRoleForAmazonEKSLocalOutpost` — Peran ini memungkinkan Amazon EKS mengelola kluster lokal di akun Anda. Kebijakan terlampir memungkinkan peran mengelola sumber daya berikut: antarmuka jaringan, grup keamanan, log, dan instans Amazon EC2.

> **Note**
>
> Peran tertaut layanan `AWSServiceRoleForAmazonEKSLocalOutpost` berbeda dari peran yang diperlukan untuk pembuatan klaster. Untuk informasi selengkapnya, lihat IAM role klaster Amazon EKS.

Peran tertaut layanan `AWSServiceRoleForAmazonEKSLocalOutpost` mempercayai layanan berikut untuk mengambil peran tersebut:

- `outposts.eks-local.amazonaws.com`

Kebijakan izin peran mengizinkan Amazon EKS untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- `AmazonEKSServiceRolePolicy`

Anda harus mengonfigurasi izin untuk mengizinkan entitas IAM (seperti pengguna, grup, atau peran) untuk membuat, mengedit, atau menghapus peran tertaut layanan. Untuk informasi selengkapnya, lihat Izin peran tertaut layanan dalam Panduan Pengguna IAM.

Membuat peran tertaut layanan untuk Amazon EKS

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda membuat klaster di AWS Management Console, API AWS CLI, atau AWS API, Amazon EKS membuat peran terkait layanan untuk Anda.

Jika Anda menghapus peran tertaut layanan ini, dan ingin membuatnya lagi, Anda dapat mengulangi proses yang sama untuk membuat peran tersebut di akun Anda. Ketika Anda membuat klaster, Amazon EKS membuatkan peran tertaut layanan lagi untuk Anda.

Mengedit Peran tertaut layanan untuk Amazon EKS

Amazon EKS tidak mengizinkan Anda untuk mengedit peran tertaut layanan `AWSServiceRoleForAmazonEKSLocalOutpost`. Setelah membuat peran terkait layanan, Anda tidak dapat mengubah nama peran karena berbagai entitas mungkin mereferensikan peran tersebut. Namun, Anda dapat menyunting penjelasan peran menggunakan IAM. Untuk informasi selengkapnya, lihat Mengedit peran tertaut layanan dalam Panduan Pengguna IAM.

Menghapus peran tertaut layanan untuk Amazon EKS

Jika Anda tidak perlu lagi menggunakan fitur atau layanan yang memerlukan peran tertaut layanan, kami sarankan supaya Anda menghapus peran tersebut. Dengan begitu, Anda tidak memiliki entitas tak terpakai yang tidak dipantau atau dipelihara secara aktif. Namun, Anda harus membersihkan peran tertaut-layanan sebelum dapat menghapusnya secara manual.

Membersihkan peran tertaut-layanan

Sebelum dapat menggunakan IAM untuk menghapus peran tertaut-layanan, Anda harus terlebih dahulu menghapus semua sumber daya yang digunakan oleh peran tersebut.

> **Note**
>
> Jika layanan Amazon EKS menggunakan peran saat Anda mencoba untuk menghapus sumber daya, maka penghapusan tersebut kemungkinan gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba lagi.

Untuk menghapus sumber daya Amazon EKS yang digunakan oleh peran **`AWSServiceRoleForAmazonEKSLocalOutpost`**.

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Di panel navigasi kiri, pilih Amazon EKS Clusters.
3. Jika klaster Anda memiliki grup simpul atau profil Fargate, Anda harus menghapusnya sebelum dapat menghapus klaster. Untuk informasi lebih lanjut, lihat Menghapus grup simpul terkelola dan Menghapus profil Fargate.
4. Pada halaman Klaster, pilih klaster yang ingin Anda hapus dan pilih Hapus.
5. Ketik nama klaster di jendela konfirmasi penghapusan, dan kemudian pilih Hapus.
6. Ulangi prosedur ini untuk klaster lain di akun Anda. Tunggu sampai semua operasi penghapusan selesai.

Hapus peran tertaut layanan secara manual

Gunakan konsol IAM, the AWS CLI, atau AWS API untuk menghapus peran `AWSServiceRoleForAmazonEKSLocalOutpost` terkait layanan. Untuk informasi selengkapnya, lihat Menghapus peran tertaut layanan dalam Panduan Pengguna IAM.

Wilayah yang didukung untuk peran tertaut-layanan Amazon EKS

Amazon EKS mendukung penggunaan peran tertaut-layanan di semua wilayah tempat layanan tersedia. Untuk informasi selengkapnya, lihat titik akhir dan kuota Amazon EKS.

IAM role klaster Amazon EKS

Peran IAM cluster Amazon EKS diperlukan untuk setiap cluster. Kubernetescluster yang dikelola oleh Amazon EKS menggunakan peran ini untuk mengelola node dan Cloud Provider lama menggunakan peran ini untuk membuat penyeimbang beban dengan Elastic Load Balancing untuk layanan.

Sebelum Anda dapat membuat klaster Amazon EKS, Anda harus membuat peran IAM dengan salah satu kebijakan IAM berikut:

- `AmazonEKSClusterPolicy`
- Kebijakan IAM khusus. Izin minimal yang mengikuti memungkinkan Kubernetes klaster mengelola node, tetapi tidak mengizinkan Penyedia Cloud lama untuk membuat penyeimbang beban dengan Elastic Load Balancing. Kebijakan IAM kustom Anda harus memiliki setidaknya izin berikut:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:*:*:instance/*",
      "Condition": {
        "ForAnyValue:StringLike": {
          "aws:TagKeys": "kubernetes.io/cluster/*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeVpcs",
        "ec2:DescribeDhcpOptions",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
  ]
}
```

Note

Sebelum 3 Oktober 2023, ClusterPolicyAmazonEks diperlukan pada peran IAM untuk setiap cluster.
Sebelum 16 April 2020, ServicePolicyAmazonEks juga diperlukan dan nama yang disarankan adalah. `eksServiceRole` Dengan peran tertaut-layanan `AWSServiceRoleForAmazonEKS`, kebijakan tersebut tidak lagi diperlukan untuk klaster yang dibuat pada atau setelah tanggal 16 April 2020.

Periksa apakah peran klaster sudah ada

Anda dapat menggunakan prosedur berikut untuk memeriksa dan melihat apakah akun Anda sudah memiliki peran klaster Amazon EKS.

Untuk memeriksa **`eksClusterRole`** di konsol IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Cari daftar peran untuk `eksClusterRole`. Jika peran yang menyertakan `eksClusterRole` tidak ada, maka lihat Membuat peran klaster Amazon EKS untuk membuat peran tersebut. Jika peran yang mencakup `eksClusterRole` ada, kemudian pilih peran untuk melihat kebijakan terlampir.
4. Pilih Izin.
5. Pastikan bahwa kebijakan terkelola ClusterPolicy AmazonEks dilampirkan pada peran tersebut. Jika kebijakan terlampir, tandanya peran klaster Amazon EKS Anda dikonfigurasi dengan benar.
6. Pilih Trust relationship, lalu pilih Edit trust policy.
7. Verifikasi bahwa hubungan kepercayaan berisi kebijakan berikut. Jika hubungan kepercayaan sesuai dengan kebijakan berikut, pilih Cancel (Batalkan). Jika hubungan kepercayaan tidak cocok, salin kebijakan ke jendela Edit kebijakan kepercayaan dan pilih Perbarui kebijakan.

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

Membuat peran klaster Amazon EKS

Anda dapat menggunakan AWS Management Console atau AWS CLI untuk membuat peran cluster.

AWS Management Console

Untuk membuat peran klaster Amazon EKS Anda di konsol IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Pilih Peran, kemudian Buat peran.
3. Di bawah Jenis entitas tepercaya, pilih AWS layanan.
4. Dari daftar Layanan AWS dropdown Use case for other, pilih EKS.
5. Pilih EKS - Cluster untuk kasus penggunaan Anda, lalu pilih Berikutnya.
6. Pada tab Tambahkan izin, pilih Berikutnya.
7. Untuk nama Peran, masukkan nama unik untuk peran Anda, seperti**`eksClusterRole`**.
8. Untuk Deskripsi, masukkan teks deskriptif seperti**`Amazon EKS - Cluster role`**.
9. Pilih Buat peran.

AWS CLI

1. Salin isi berikut ke file bernama *`cluster-trust-policy.json`*.

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

2. Buat peran. Anda dapat mengganti ***`eksClusterRole`*** dengan nama apa pun yang Anda pilih.

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

3. Lampirkan kebijakan IAM yang diperlukan ke peran tersebut.

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

IAM role simpul Amazon EKS

`kubelet`Daemon node Amazon EKS melakukan panggilan ke AWS API atas nama Anda. Simpul menerima izin untuk panggilan API ini melalui profil instans IAM dan kebijakan terkait. Sebelum Anda dapat meluncurkan simpul dan mendaftarkannya ke dalam sebuah klaster, Anda harus membuat IAM role untuk digunakan oleh simpul ketika diluncurkan. Persyaratan ini berlaku untuk simpul yang diluncurkan dengan AMI yang dioptimalkan oleh Amazon EKS dan disediakan oleh Amazon, atau dengan AMI simpul lainnya yang ingin Anda gunakan. Selain itu, persyaratan ini berlaku untuk grup node terkelola dan node yang dikelola sendiri.

> **Note**
>
> Anda tidak dapat menggunakan peran yang sama yang digunakan untuk membuat cluster apa pun.

Sebelum membuat node, Anda harus membuat peran IAM dengan izin berikut:

- Izin `kubelet` untuk menjelaskan sumber daya Amazon EC2 di VPC, seperti yang disediakan oleh kebijakan. `AmazonEKSWorkerNodePolicy` Kebijakan ini juga memberikan izin untuk Agen Identitas Pod Amazon EKS.
- Izin `kubelet` untuk menggunakan gambar kontainer dari Amazon Elastic Container Registry (Amazon ECR) Registry, seperti yang disediakan oleh kebijakan. `AmazonEC2ContainerRegistryReadOnly` Izin untuk menggunakan image kontainer dari Amazon Elastic Container Registry (Amazon ECR) diperlukan karena add-on bawaan untuk pod yang menjalankan jaringan yang menggunakan image kontainer dari Amazon ECR.
- (Opsional) Izin untuk Amazon EKS Pod Identity Agent untuk menggunakan `eks-auth:AssumeRoleForPodIdentity` action tersebut guna mengambil kredensial Pod. Jika Anda tidak menggunakan WorkerNodePolicyAmazoneks, maka Anda harus memberikan izin ini selain izin EC2 untuk menggunakan EKS Pod Identity.
- (Opsional) Jika Anda tidak menggunakan IRSA atau EKS Pod Identity untuk memberikan izin ke pod VPC CNI, maka Anda harus memberikan izin untuk VPC CNI pada peran instance. Anda dapat

menggunakan kebijakan AmazonEKS_CNI_Policy terkelola (jika Anda membuat klaster dengan IPv4 keluarga) atau kebijakan IPv6 yang Anda buat (jika Anda membuat klaster dengan IPv6 keluarga). Namun, alih-alih melampirkan kebijakan ke peran ini, sebaiknya Anda melampirkan kebijakan tersebut ke peran terpisah yang digunakan khusus untuk add-on Amazon VPC CNI. Untuk informasi selengkapnya tentang membuat peran terpisah untuk add-on Amazon VPC CNI, lihat. Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA)

> **Note**
>
> Sebelum 3 Oktober 2023, AmazonEKSWorkerNodePolicy dan AmazonEC2ContainerRegistryReadOnly diperlukan pada peran IAM untuk setiap grup node yang dikelola.
> Grup node Amazon EC2 harus memiliki peran IAM yang berbeda dari profil Fargate. Untuk informasi selengkapnya, lihat Peran IAM Pod eksekusi Amazon EKS.

Periksa apakah peran simpul sudah ada

Anda dapat menggunakan prosedur berikut untuk memeriksa dan melihat apakah akun Anda sudah memiliki peran simpul Amazon EKS.

Untuk memeriksa **eksNodeRole** di dalam konsol IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Cari daftar peran untukeksNodeRole,AmazonEKSNodeRole, atauNodeInstanceRole. Jika peran dengan salah satu nama itu tidak ada, maka lihat Membuat IAM role simpul Amazon EKS untuk membuat peran tersebut. Jika peran yang berisieksNodeRole,AmazonEKSNodeRole, atau NodeInstanceRole memang ada, pilih peran untuk melihat kebijakan terlampir.
4. Pilih Izin.
5. Pastikan bahwa kebijakan terkelola WorkerNodePolicy AmazonEks dan ContainerRegistryReadOnly Amazonec2 dilampirkan ke peran atau kebijakan khusus dilampirkan dengan izin minimal.

> **Note**
>
> Jika kebijakan AmazonEks_CNI_Policy dilampirkan ke peran, kami sarankan untuk menghapusnya dan melampirkannya ke peran IAM yang dipetakan ke akun layanan sebagai gantinya. `aws-node` Kubernetes Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).

6. Pilih Trust relationship, lalu pilih Edit trust policy.
7. Verifikasi bahwa hubungan kepercayaan berisi kebijakan berikut. Jika hubungan kepercayaan sesuai dengan kebijakan berikut, pilih Cancel (Batalkan). Jika hubungan kepercayaan tidak cocok, salin kebijakan ke jendela Edit kebijakan kepercayaan dan pilih Perbarui kebijakan.

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

Membuat IAM role simpul Amazon EKS

Anda dapat membuat peran IAM node dengan AWS Management Console atau. AWS CLI

AWS Management Console

Untuk membuat peran simpul Amazon EKS di dalam konsol IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Pada halaman Peran, pilih Buat peran.
4. Pada halaman Pilih entitas tepercaya, lakukan hal berikut:

a. Di bagian Jenis entitas tepercaya, pilih AWS layanan.

b. Di bawah Kasus penggunaan, pilih EC2.

c. Pilih Berikutnya.

5. Pada halaman Tambahkan izin, lampirkan kebijakan khusus atau lakukan hal berikut:

a. Di dalam kotak Filter kebijakan, masukkan **`AmazonEKSWorkerNodePolicy`**.

b. Pilih kotak centang di sebelah kiri WorkerNodePolicyAmazonEks di hasil pencarian.

c. Pilih Hapus filter.

d. Di dalam kotak Filter kebijakan, masukkan **`AmazonEC2ContainerRegistryReadOnly`**.

e. Pilih kotak centang di sebelah kiri AmazonEC2 ContainerRegistryReadOnly di hasil pencarian.

Kebijakan terkelola Amazoneks_CNI_Policy, atau kebijakan IPv6 yang Anda buat juga harus dilampirkan ke peran ini atau peran lain yang dipetakan ke akun layanan. `aws-node` Kubernetes Sebaiknya tetapkan kebijakan ke peran yang terkait dengan akun Kubernetes layanan alih-alih menetapkannya ke peran ini. Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).

f. Pilih Berikutnya.

6. Pada halaman Nama, tinjau, dan buat, lakukan hal berikut:

a. Untuk nama Peran, masukkan nama unik untuk peran Anda, seperti**`AmazonEKSNodeRole`**.

b. Untuk Deskripsi, ganti teks saat ini dengan teks deskriptif seperti**`Amazon EKS - Node role`**.

c. Di bawah Tambahkan tag (Opsional), tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tanda di IAM, lihat Menandai sumber daya IAM di Panduan Pengguna IAM.

d. Pilih Buat peran.

AWS CLI

1. Jalankan perintah berikut untuk membuat `node-role-trust-relationship.json` file.

```
cat >node-role-trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

2. Buat peran IAM.

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

3. Lampirkan dua kebijakan yang dikelola IAM yang diperlukan ke peran IAM.

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

4. Lampirkan salah satu kebijakan IAM berikut ke peran IAM tergantung pada keluarga IP mana Anda membuat klaster. Kebijakan harus dilampirkan pada peran ini atau peran yang terkait dengan akun Kubernetes `aws-node` layanan yang digunakan untuk Amazon VPC CNI plugin for Kubernetes. Sebaiknya tetapkan kebijakan ke peran yang terkait dengan akun Kubernetes layanan. Untuk menetapkan kebijakan ke peran yang terkait dengan akun Kubernetes layanan, lihat[Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA)](#).

   - IPv4

     ```
     aws iam attach-role-policy \
     ```

```
--policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
--role-name AmazonEKSNodeRole
```

- IPv6

1. Salin teks berikut dan simpan ke file bernama *vpc-cni-ipv6-policy*.json.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
```

2. Buat kebijakan IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-
document file://vpc-cni-ipv6-policy.json
```

3. Lampirkan kebijakan IAM ke peran IAM. Ganti *111122223333* dengan ID akun Anda.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
  --role-name AmazonEKSNodeRole
```

Peran IAM Pod eksekusi Amazon EKS

Peran Pod eksekusi Amazon EKS diperlukan untuk berjalan Pods pada AWS Fargate infrastruktur.

Saat klaster Anda membuat Pods AWS Fargate infrastruktur, komponen yang berjalan di infrastruktur Fargate harus melakukan panggilan ke AWS API atas nama Anda. Ini agar mereka dapat melakukan tindakan seperti menarik gambar kontainer dari Amazon ECR atau merutekan log ke AWS layanan lain. Peran Pod eksekusi Amazon EKS memberikan izin IAM untuk melakukan ini.

Saat membuat profil Fargate, Anda harus menentukan peran Pod eksekusi untuk komponen Amazon EKS yang berjalan di infrastruktur Fargate menggunakan profil. Peran ini ditambahkan ke kontrol akses berbasis Kubernetes peran klaster (RBAC) untuk otorisasi. Hal ini memungkinkan `kubelet` yang berjalan di infrastruktur Fargate untuk mendaftar dengan cluster Amazon EKS Anda sehingga dapat muncul di cluster Anda sebagai node.

> **Note**
>
> Profil Fargate harus memiliki peran IAM yang berbeda dari grup node Amazon EC2.

> **Important**
>
> Kontainer yang berjalan di Fargate tidak Pod dapat mengasumsikan izin IAM yang terkait dengan peran eksekusi. Pod Untuk memberikan kontainer di Fargate Anda Pod izin untuk mengakses AWS layanan lain, Anda harus menggunakannya. IAM role untuk akun layanan

Sebelum Anda membuat profil Fargate, Anda harus membuat peran IAM dengan. `AmazonEKSFargatePodExecutionRolePolicy`

Periksa peran Pod eksekusi yang sudah ada yang dikonfigurasi dengan benar

Anda dapat menggunakan prosedur berikut untuk memeriksa dan melihat apakah akun Anda sudah memiliki peran Pod eksekusi Amazon EKS yang dikonfigurasi dengan benar. Untuk menghindari masalah keamanan wakil yang membingungkan, penting bahwa peran membatasi akses berdasarkan`SourceArn`. Anda dapat memodifikasi peran eksekusi sesuai kebutuhan untuk menyertakan dukungan untuk profil Fargate di cluster lain.

Untuk memeriksa peran Pod eksekusi Amazon EKS di konsol IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Pada halaman Peran, cari daftar peran untuk AmazonEks FargatePodExecutionRole. Jika peran tidak ada, lihat Membuat peran Pod eksekusi Amazon EKS untuk membuat peran. Jika peran itu memang ada, pilih perannya.
4. Di halaman Amazoneks, FargatePodExecutionRole lakukan hal berikut:
   a. Pilih Izin.
   b. Pastikan bahwa kebijakan terkelola FargatePodExecutionRolePolicyAmazonEks Amazon dilampirkan pada peran tersebut.
   c. Pilih Hubungan kepercayaan.
   d. Pilih Edit kebijakan kepercayaan.
5. Pada halaman Edit kebijakan kepercayaan, verifikasi bahwa hubungan kepercayaan berisi kebijakan berikut dan memiliki baris untuk profil Fargate di klaster Anda. Jika demikian, pilih Batalkan.

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

   Jika kebijakan cocok tetapi tidak memiliki baris yang menentukan profil Fargate di klaster, Anda dapat menambahkan baris berikut di bagian `ArnLike` atas objek. Ganti *region-code* dengan

klaster Anda, *111122223333* dengan ID akun Anda, dan *my-cluster* dengan nama cluster Anda. Wilayah AWS

```
"aws:SourceArn": "arn:aws:eks:region-code:111122223333:fargateprofile/my-cluster/
*",
```

Jika kebijakan tidak cocok, salin kebijakan lengkap sebelumnya ke dalam formulir dan pilih Perbarui kebijakan. Ganti *region-code* dengan tempat Wilayah AWS cluster Anda berada. Jika Anda ingin menggunakan peran yang sama Wilayah AWS di semua akun Anda, ganti *kode wilayah -1 us-iso-east us-isob-east* dengan. * Ganti *111122223333* dengan ID akun Anda dan *my-cluster* dengan nama cluster Anda. Jika Anda ingin menggunakan peran yang sama untuk semua cluster di akun Anda, ganti *my-cluster* dengan*.

Membuat peran Pod eksekusi Amazon EKS

Jika Anda belum memiliki peran Pod eksekusi Amazon EKS untuk klaster Anda, Anda dapat menggunakan AWS Management Console atau AWS CLI untuk membuatnya.

AWS Management Console

Untuk membuat peran AWS FargatePod eksekusi dengan AWS Management Console

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Pada halaman Peran, pilih Buat peran.
4. Pada halaman Pilih entitas tepercaya, lakukan hal berikut:
   a. Di bagian Jenis entitas tepercaya, pilih AWS layanan.
   b. Dari daftar Layanan AWS dropdown Use case for other, pilih EKS.
   c. Pilih EKS - Fargate Pod.
   d. Pilih Berikutnya.
5. Pada halaman Tambahkan izin, pilih Berikutnya.
6. Pada halaman Nama, tinjau, dan buat, lakukan hal berikut:
   a. Untuk nama Peran, masukkan nama unik untuk peran Anda, seperti**AmazonEKSFargatePodExecutionRole**.

b. Di bawah Tambahkan tag (Opsional), tambahkan metadata ke peran dengan melampirkan tag sebagai pasangan nilai kunci. Untuk informasi selengkapnya tentang penggunaan tanda di IAM, lihat Menandai sumber daya IAM di Panduan Pengguna IAM.

c. Pilih Buat peran.

7. Pada halaman Peran, cari daftar peran untuk AmazonEks FargatePodExecutionRole. Pilih peran.

8. Di halaman Amazoneks, FargatePodExecutionRole lakukan hal berikut:

a. Pilih Hubungan kepercayaan.

b. Pilih Edit kebijakan kepercayaan.

9. Pada halaman Edit kebijakan kepercayaan, lakukan hal berikut:

a. Salin dan tempel konten berikut ke dalam formulir Edit kebijakan kepercayaan. Ganti *kode wilayah* dengan Wilayah AWS cluster Anda. Jika Anda ingin menggunakan peran yang sama Wilayah AWS di semua akun Anda, ganti *kode wilayah -1 us-iso-east us-isob-east* dengan. * Ganti *111122223333* dengan ID akun Anda dan *my-cluster* dengan nama cluster Anda. Jika Anda ingin menggunakan peran yang sama untuk semua cluster di akun Anda, ganti *my-cluster* dengan*.

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

b. Pilih Perbarui kebijakan.

AWS CLI

Untuk membuat peran AWS FargatePod eksekusi dengan AWS CLI

1. Salin dan tempel konten berikut ke file bernama*pod-execution-role-trust-policy*.json. Ganti *kode wilayah* dengan Wilayah AWS cluster Anda. Jika Anda ingin menggunakan peran yang sama Wilayah AWS di semua akun Anda, ganti *kode wilayah -1 us-iso-east us-isob-east* dengan. * Ganti *111122223333* dengan ID akun Anda dan *my-cluster* dengan nama cluster Anda. Jika Anda ingin menggunakan peran yang sama untuk semua cluster di akun Anda, ganti *my-cluster* dengan*.

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

2. Buat peran IAM Pod eksekusi.

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

3. Lampirkan kebijakan terkelola IAM Amazon EKS yang diperlukan untuk peran tersebut.

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

Peran IAM konektor Amazon EKS

Anda dapat menghubungkan Kubernetes cluster untuk melihatnya di file Anda AWS Management Console. Untuk terhubung ke Kubernetes cluster, buat peran IAM.

Periksa peran konektor EKS yang ada

Anda dapat menggunakan prosedur berikut untuk memeriksa dan melihat apakah akun Anda sudah memiliki peran konektor Amazon EKS.

Untuk memeriksa **`AmazonEKSConnectorAgentRole`** di konsol IAM

1. Buka konsol IAM di https://console.aws.amazon.com/iam/.
2. Di panel navigasi sebelah kiri, pilih Peran.
3. Cari daftar peran untuk `AmazonEKSConnectorAgentRole`. Jika peran yang menyertakan `AmazonEKSConnectorAgentRole` tidak ada, maka lihat Membuat peran agen konektor Amazon EKS untuk membuat peran tersebut. Jika peran yang mencakup `AmazonEKSConnectorAgentRole` ada, kemudian pilih peran untuk melihat kebijakan terlampir.
4. Pilih Izin.
5. Pastikan bahwa kebijakan terkelola ConnectorAgentPolicy AmazonEks dilampirkan pada peran tersebut. Jika kebijakan dilampirkan, peran konektor Amazon EKS Anda dikonfigurasi dengan benar.
6. Pilih Trust relationship, lalu pilih Edit trust policy.
7. Verifikasi bahwa hubungan kepercayaan berisi kebijakan berikut. Jika hubungan kepercayaan sesuai dengan kebijakan berikut, pilih Cancel (Batalkan). Jika hubungan kepercayaan tidak cocok, salin kebijakan ke jendela Edit kebijakan kepercayaan dan pilih Perbarui kebijakan.

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

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

Membuat peran agen konektor Amazon EKS

Anda dapat menggunakan AWS Management Console atau AWS CloudFormation untuk membuat peran agen konektor.

AWS CLI

1. Buat file bernama `eks-connector-agent-trust-policy.json` yang berisi JSON berikut untuk digunakan untuk peran IAM.

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

2. Buat file bernama `eks-connector-agent-policy.json` yang berisi JSON berikut untuk digunakan untuk peran IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SsmControlChannel",
            "Effect": "Allow",
            "Action": [
                "ssmmessages:CreateControlChannel"
```

```
            ],
            "Resource": "arn:aws:eks:*:*:cluster/*"
        },
        {
            "Sid": "ssmDataplaneOperations",
            "Effect": "Allow",
            "Action": [
                "ssmmessages:CreateDataChannel",
                "ssmmessages:OpenDataChannel",
                "ssmmessages:OpenControlChannel"
            ],
            "Resource": "*"
        }
    ]
}
```

3. Buat peran agen Amazon EKS Connector menggunakan kebijakan kepercayaan dan kebijakan yang Anda buat di item daftar sebelumnya.

```
aws iam create-role \
     --role-name AmazonEKSConnectorAgentRole \
     --assume-role-policy-document file://eks-connector-agent-trust-policy.json
```

4. Lampirkan kebijakan ke peran agen Amazon EKS Connector Anda.

```
aws iam put-role-policy \
     --role-name AmazonEKSConnectorAgentRole \
     --policy-name AmazonEKSConnectorAgentPolicy \
     --policy-document file://eks-connector-agent-policy.json
```

AWS CloudFormation

Untuk membuat peran agen konektor Amazon EKS Anda dengan AWS CloudFormation.

1. Simpan AWS CloudFormation template berikut ke file teks di sistem lokal Anda.

> **Note**
>
> Template ini juga menciptakan peran terkait layanan yang seharusnya dibuat saat `registerCluster` API dipanggil. Lihat Menggunakan peran untuk menghubungkan Kubernetes cluster ke Amazon EKS untuk detail.

```
---
AWSTemplateFormatVersion: '2010-09-09'
Description: 'Provisions necessary resources needed to register clusters in EKS'
Parameters: {}
Resources:
  EKSConnectorSLR:
    Type: AWS::IAM::ServiceLinkedRole
    Properties:
      AWSServiceName: eks-connector.amazonaws.com

  EKSConnectorAgentRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Action: [ 'sts:AssumeRole' ]
            Principal:
              Service: 'ssm.amazonaws.com'

  EKSConnectorAgentPolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyName: EKSConnectorAgentPolicy
      Roles:
        - {Ref: 'EKSConnectorAgentRole'}
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: 'Allow'
            Action: [ 'ssmmessages:CreateControlChannel' ]
            Resource:
            - Fn::Sub: 'arn:${AWS::Partition}:eks:*:*:cluster/*'
```

```
          - Effect: 'Allow'
            Action: [ 'ssmmessages:CreateDataChannel',
 'ssmmessages:OpenDataChannel', 'ssmmessages:OpenControlChannel' ]
            Resource: "*"
Outputs:
  EKSConnectorAgentRoleArn:
    Description: The agent role that EKS connector uses to communicate with
 Layanan AWS.
    Value: !GetAtt EKSConnectorAgentRole.Arn
```

2. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
3. Pilih Buat tumpukan (baik dengan sumber daya baru atau sumber daya yang ada.
4. Untuk Tentukan templat, pilih Unggah sebuah file templat, kemudian pilih Pilih file.
5. Pilih file yang telah Anda buat sebelumnya, dan kemudian pilih Selanjutnya.
6. Untuk Nama tumpukan, masukkan nama untuk peran Anda, misalnya `eksConnectorAgentRole`, kemudian pilih Selanjutnya.
7. Pada halaman Konfigurasikan opsi tumpukan, pilih Selanjutnya.
8. Di halaman Tinjauan, tinjau informasi Anda, nyatakan bahwa tumpukan dapat membuat sumber daya IAM, kemudian pilih Buat.

AWS kebijakan terkelola untuk Amazon Elastic Kubernetes Service

Kebijakan AWS terkelola adalah kebijakan mandiri yang dibuat dan dikelola oleh AWS. AWS Kebijakan terkelola dirancang untuk memberikan izin bagi banyak kasus penggunaan umum sehingga Anda dapat mulai menetapkan izin kepada pengguna, grup, dan peran.

Perlu diingat bahwa kebijakan AWS terkelola mungkin tidak memberikan izin hak istimewa paling sedikit untuk kasus penggunaan spesifik Anda karena tersedia untuk digunakan semua pelanggan. AWS Kami menyarankan Anda untuk mengurangi izin lebih lanjut dengan menentukan kebijakan yang dikelola pelanggan yang khusus untuk kasus penggunaan Anda.

Anda tidak dapat mengubah izin yang ditentukan dalam kebijakan AWS terkelola. Jika AWS memperbarui izin yang ditentukan dalam kebijakan AWS terkelola, pembaruan akan memengaruhi semua identitas utama (pengguna, grup, dan peran) yang dilampirkan kebijakan tersebut. AWS kemungkinan besar akan memperbarui kebijakan AWS terkelola saat baru Layanan AWS diluncurkan atau operasi API baru tersedia untuk layanan yang ada.

Untuk informasi selengkapnya, lihat Kebijakan yang dikelola AWS dalam Panduan Pengguna IAM.

AWS kebijakan terkelola: Amazoneks_CNI_Policy

Anda dapat melampirkan `AmazonEKS_CNI_Policy` ke entitas IAM Anda. Sebelum Anda membuat grup node Amazon EC2, kebijakan ini harus dilampirkan ke peran IAM node, atau ke peran IAM yang digunakan secara khusus oleh. Amazon VPC CNI plugin for Kubernetes Ini dimaksudkan agar dapat melakukan tindakan atas nama Anda. Kami menyarankan Anda melampirkan kebijakan ke peran yang hanya digunakan oleh plugin. Untuk informasi selengkapnya, lihat Bekerja dengan add-on Amazon VPC CNI plugin for Kubernetes Amazon EKS dan Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).

Detail izin

Kebijakan ini mencakup izin yang memungkinkan Amazon EKS untuk menyelesaikan tugas-tugas berikut:

- **`ec2:*NetworkInterface`**dan **`ec2:*PrivateIpAddresses`** — Memungkinkan plugin Amazon VPC CNI untuk melakukan tindakan seperti menyediakan Antarmuka Jaringan Elastis dan alamat IP untuk menyediakan jaringan Pods untuk aplikasi yang berjalan di Amazon EKS.
- **`ec2`**baca tindakan - Memungkinkan plugin Amazon VPC CNI untuk melakukan tindakan seperti mendeskripsikan instance dan subnet untuk melihat jumlah alamat IP gratis di subnet VPC Amazon Anda. VPC CNI dapat menggunakan alamat IP gratis di setiap subnet untuk memilih subnet dengan alamat IP paling gratis untuk digunakan saat membuat elastic network interface.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat Amazoneks_CNI_Policy di Panduan Referensi Kebijakan Terkelola. AWS

AWS kebijakan terkelola: AmazonEks ClusterPolicy

Anda dapat melampirkan `AmazonEKSClusterPolicy` ke entitas IAM Anda. Sebelum membuat klaster, Anda harus memiliki peran IAM klaster dengan kebijakan ini terlampir. Kubernetescluster yang dikelola oleh Amazon EKS melakukan panggilan ke AWS layanan lain atas nama Anda. Mereka melakukan ini untuk mengelola sumber daya yang Anda gunakan dengan layanan.

Kebijakan ini mencakup izin yang memungkinkan Amazon EKS untuk menyelesaikan tugas-tugas berikut:

- **`autoscaling`** – Baca dan perbarui konfigurasi grup Auto Scaling. Izin ini tidak digunakan oleh Amazon EKS tetapi tetap berada di dalam kebijakan untuk kompatibilitas ke belakang.
- **`ec2`** – Kerjakan dengan volume dan sumber daya jaringan yang terkait dengan simpul Amazon EC2. Hal ini diperlukan agar bidang Kubernetes kontrol dapat menggabungkan instans ke klaster dan secara dinamis menyediakan serta mengelola volume Amazon EBS yang diminta oleh Kubernetes volume persisten.
- **`elasticloadbalancing`** – Kerjakan dengan Elastic Load Balancers dan tambahkan simpul ke Elastic Load Balancers tersebut sebagai target. Hal ini diperlukan agar bidang Kubernetes kontrol dapat secara dinamis menyediakan Elastic Load Balancer yang diminta oleh layanan. Kubernetes
- **`iam`** – Buat peran tertaut-layanan. Hal ini diperlukan agar bidang Kubernetes kontrol dapat secara dinamis menyediakan Elastic Load Balancer yang diminta oleh layanan. Kubernetes
- **`kms`** – Baca kunci dari AWS KMS. Ini diperlukan untuk pesawat Kubernetes kontrol untuk mendukung enkripsi rahasia Kubernetes rahasia yang disimpan di dalamnya`etcd`.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat ClusterPolicyAmazonEks di Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonEks FargatePodExecutionRolePolicy

Anda dapat melampirkan `AmazonEKSFargatePodExecutionRolePolicy` ke entitas IAM Anda. Sebelum Anda dapat membuat profil Fargate, Anda harus membuat peran Pod eksekusi Fargate dan melampirkan kebijakan ini padanya. Lihat informasi yang lebih lengkap di Buat peran eksekusi Fargate Pod dan AWS Fargate profil.

Kebijakan ini memberikan peran izin yang menyediakan akses ke sumber daya AWS layanan lain yang diperlukan untuk menjalankan Amazon EKS di Pods Fargate.

Detail izin

Kebijakan ini mencakup izin yang memungkinkan Amazon EKS untuk menyelesaikan tugas-tugas berikut:

- **`ecr`**— Memungkinkan Pod yang berjalan di Fargate untuk menarik gambar kontainer yang disimpan di Amazon ECR.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat FargatePodExecutionRolePolicyAmazonEks di Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonEks ForFargateServiceRolePolicy

Anda tidak dapat melampirkan `AmazonEKSForFargateServiceRolePolicy` ke entitas IAM Anda. Kebijakan ini dilampirkan ke peran tertaut layanan yang mengizinkan Amazon EKS untuk melakukan tindakan atas nama Anda. Untuk informasi lebih lanjut, lihat AWSServiceRoleforAmazonEKSForFargate.

Kebijakan ini memberikan izin yang diperlukan kepada Amazon EKS untuk menjalankan tugas Fargate. Kebijakan ini hanya digunakan jika Anda memiliki simpul Fargate.

Detail izin

Kebijakan ini mencakup izin berikut yang memungkinkan Amazon EKS untuk menyelesaikan tugas berikut.

- **`ec2`** – Buat dan hapus Antarmuka Jaringan Elastis, dan jelaskan tentang Antarmuka Jaringan Elastis serta sumber daya-sumber daya. Ini diperlukan agar layanan Amazon EKS Fargate dapat mengonfigurasi jaringan VPC yang diperlukan untuk Pod Fargate.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat ForFargateServiceRolePolicyAmazonEks di Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonEks ServicePolicy

Anda dapat melampirkan `AmazonEKSServicePolicy` ke entitas IAM Anda. Cluster yang dibuat sebelum 16 April 2020, mengharuskan Anda untuk membuat peran IAM dan melampirkan kebijakan ini padanya. Cluster yang dibuat pada atau setelah 16 April 2020, tidak mengharuskan Anda untuk membuat peran dan tidak mengharuskan Anda untuk menetapkan kebijakan ini. Saat Anda membuat klaster menggunakan prinsipal IAM yang memiliki `iam:CreateServiceLinkedRole` izin, peran terkait layanan AWS ServiceRoleforAmazonEKS secara otomatis dibuat untuk Anda. Peran tertaut-layanan memiliki AWS kebijakan terkelola: AmazonEks ServiceRolePolicy yang terlampir padanya.

Kebijakan ini memungkinkan Amazon EKS untuk membuat dan mengelola sumber daya yang diperlukan guna mengoperasikan klaster Amazon EKS.

Detail izin

Kebijakan ini mencakup izin berikut yang memungkinkan Amazon EKS untuk menyelesaikan tugas berikut.

- **`eks`**— Perbarui Kubernetes versi cluster Anda setelah Anda memulai pembaruan. Izin ini tidak digunakan oleh Amazon EKS tetapi tetap berada dalam kebijakan untuk kompatibilitas ke belakang.
- **`ec2`** – Kerjakan dengan Antarmuka Jaringan Elastis dan sumber daya jaringan lainnya serta tanda-tanda. Ini diperlukan oleh Amazon EKS untuk mengonfigurasi jaringan yang memfasilitasi komunikasi antara node dan bidang Kubernetes kontrol.
- **`route53`** – Kaitkan VPC dengan zona yang di-hosting. Ini diperlukan oleh Amazon EKS untuk mengaktifkan jaringan endpoint pribadi untuk server API Kubernetes cluster Anda.
- **`logs`** – Log acara. Ini diperlukan agar Amazon EKS dapat mengirimkan log pesawat Kubernetes kontrol ke CloudWatch.
- **`iam`** – Buat peran tertaut-layanan. Ini diperlukan agar Amazon EKS dapat membuat peran `AWSServiceRoleForAmazonEKS`terkait layanan atas nama Anda.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat ServicePolicyAmazonEks di Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonEks ServiceRolePolicy

Anda tidak dapat melampirkan `AmazonEKSServiceRolePolicy` ke entitas IAM Anda. Kebijakan ini dilampirkan pada peran tertaut layanan yang mengizinkan Amazon EKS untuk melakukan tindakan atas nama Anda. Untuk informasi selengkapnya, lihat Izin peran tertaut layanan untuk Amazon EKS. Saat Anda membuat klaster menggunakan prinsipal IAM yang memiliki `iam:CreateServiceLinkedRole` izin, peran terkait layanan AWS ServiceRoleforAmazonEKS secara otomatis dibuat untuk Anda dan kebijakan ini dilampirkan padanya.

Kebijakan ini memungkinkan peran terkait layanan untuk memanggil AWS layanan atas nama Anda.

Detail izin

Kebijakan ini mencakup izin berikut yang memungkinkan Amazon EKS untuk menyelesaikan tugas berikut.

- **`ec2`**— Buat dan jelaskan Antarmuka Jaringan Elastis dan instans Amazon EC2, grup keamanan klaster, dan VPC yang diperlukan untuk membuat klaster.
- **`iam`** – Buat daftar semua kebijakan terkelola yang dilampirkan ke IAM role. Ini diperlukan agar Amazon EKS dapat mencantumkan dan memvalidasi semua kebijakan dan izin terkelola yang diperlukan untuk membuat klaster.

- Kaitkan VPC dengan zona yang dihosting — Ini diperlukan oleh Amazon EKS untuk mengaktifkan jaringan titik akhir pribadi untuk server API Kubernetes cluster Anda.
- Peristiwa log - Ini diperlukan agar Amazon EKS dapat mengirimkan log pesawat Kubernetes kontrol ke CloudWatch.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat ServiceRolePolicyAmazonEks di Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonekSVPC ResourceController

Anda dapat melampirkan kebijakan `AmazonEKSVPCResourceController` ke identitas IAM Anda. Jika Anda menggunakan grup keamanan untuk Pods, Anda harus melampirkan kebijakan ini ke Anda IAM role klaster Amazon EKS untuk melakukan tindakan atas nama Anda.

Kebijakan ini memberikan izin peran klaster untuk mengelola Antarmuka Jaringan Elastis dan alamat IP untuk simpul.

Detail izin

Kebijakan ini mencakup izin yang memungkinkan Amazon EKS untuk menyelesaikan tugas-tugas berikut:

- **`ec2`**— Kelola Antarmuka Jaringan Elastis dan alamat IP untuk mendukung grup dan Windows node Pod keamanan.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat AmazonEkSVPC ResourceController di Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonEks WorkerNodePolicy

Anda dapat melampirkan `AmazonEKSWorkerNodePolicy` ke entitas IAM Anda. Anda harus melampirkan kebijakan ini ke IAM role simpul yang Anda tentukan ketika membuat simpul Amazon EC2 yang mengizinkan Amazon EKS untuk melakukan tindakan atas nama Anda. Jika Anda membuat grup simpul menggunakan `eksctl`, ia akan membuat IAM role simpul dan melampirkan kebijakan ini ke peran secara otomatis.

Kebijakan ini memberikan izin kepada simpul Amazon EKS Amazon EC2 untuk terhubung ke klaster Amazon EKS.

Detail izin

Kebijakan ini mencakup izin yang memungkinkan Amazon EKS untuk menyelesaikan tugas-tugas berikut:

- **`ec2`** – Baca volume instans dan informasi jaringan. Ini diperlukan agar Kubernetes node dapat menjelaskan informasi tentang sumber daya Amazon EC2 yang diperlukan agar node dapat bergabung dengan kluster Amazon EKS.
- **`eks`** – Secara opsional, deskripsikan klaster sebagai bagian dari bootstrapping simpul.
- **`eks-auth:AssumeRoleForPodIdentity`**— Izinkan pengambilan kredensyal untuk beban kerja EKS pada node. Hal ini diperlukan agar EKS Pod Identity berfungsi dengan baik.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat WorkerNodePolicyAmazonEks di Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AWSServiceRoleForAmazonEKSNodegroup

Anda tidak dapat melampirkan `AWS ServiceRoleForAmazonEKSNodegroup` ke entitas IAM Anda. Kebijakan ini dilampirkan pada peran tertaut layanan yang mengizinkan Amazon EKS untuk melakukan tindakan atas nama Anda. Untuk informasi selengkapnya, lihat Izin peran tertaut layanan untuk Amazon EKS.

Kebijakan ini mengabulkan izin peran `AWS ServiceRoleForAmazonEKSNodegroup` yang memungkinkannya untuk membuat dan mengelola grup simpul Amazon EC2 di akun Anda.

Detail izin

Kebijakan ini mencakup izin yang memungkinkan Amazon EKS untuk menyelesaikan tugas-tugas berikut:

- **`ec2`** – Kerjakan bersama grup keamanan, tanda, dan templat peluncuran. Ini diperlukan oleh grup simpul yang dikelola Amazon EKS untuk mengaktifkan konfigurasi akses jarak jauh. Selain itu, grup simpul yang dikelola Amazon EKS membuat templat peluncuran atas nama Anda. Ini untuk mengonfigurasi grup Amazon EC2 Auto Scaling yang mendukung setiap grup simpul terkelola.
- **`iam`** – Buat peran tertaut layanan dan berikan peran. Ini diperlukan oleh grup simpul yang dikelola Amazon EKS dalam mengelola profil instans untuk peran yang diteruskan ketika membuat grup simpul terkelola. Profil instans ini digunakan oleh instans Amazon EC2 yang diluncurkan sebagai bagian dari grup simpul terkelola. Amazon EKS perlu membuat peran tertaut-layanan untuk layanan lain seperti grup Amazon EC2 Auto Scaling. Izin ini digunakan dalam pembuatan grup node terkelola.

- **`autoscaling`** – Kerjakan bersama grup Auto Scaling keamanan. Ini diperlukan oleh grup simpul yang dikelola Amazon EKS untuk mengelola grup Amazon EC2 Auto Scaling yang mendukung setiap grup simpul terkelola. Ini juga digunakan untuk mendukung fungsionalitas seperti mengusir Pods ketika node dihentikan atau didaur ulang selama pembaruan grup node.

Untuk melihat versi terbaru dokumen kebijakan JSON, lihat AWSServiceRoleForAmazonEKSNodegroupdi Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonEBSCSI DriverPolicy

`AmazonEBSCSIDriverPolicy`Kebijakan ini memungkinkan driver Amazon EBS Container Storage Interface (CSI) untuk membuat, memodifikasi, melampirkan, melepaskan, dan menghapus volume atas nama Anda. Ini juga memberikan izin driver EBS CSI untuk membuat dan menghapus snapshot, dan untuk membuat daftar instance, volume, dan snapshot Anda.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat AmazonEBSCSI DriverServiceRolePolicy di Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonefSCSI DriverPolicy

`AmazonEFSCSIDriverPolicy`Kebijakan ini memungkinkan Amazon EFS Container Storage Interface (CSI) untuk membuat dan menghapus titik akses atas nama Anda. Ini juga memberikan izin driver Amazon EFS CSI untuk mencantumkan sistem file titik akses Anda, target pemasangan, dan zona ketersediaan Amazon EC2.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat AmazonefSCSI DriverServiceRolePolicy di Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonEks LocalOutpostClusterPolicy

Anda dapat melampirkan kebijakan ini ke entitas IAM. Sebelum membuat klaster lokal, Anda harus melampirkan kebijakan ini ke peran klaster Anda. Kubernetescluster yang dikelola oleh Amazon EKS melakukan panggilan ke AWS layanan lain atas nama Anda. Mereka melakukan ini untuk mengelola sumber daya yang Anda gunakan dengan layanan.

`AmazonEKSLocalOutpostClusterPolicy`Termasuk izin berikut:

- **`ec2`**— Izin yang diperlukan untuk instans Amazon EC2 agar berhasil bergabung dengan cluster sebagai instans bidang kontrol.

- **`ssm`**— Memungkinkan koneksi Amazon EC2 Systems Manager ke instans control plane, yang digunakan oleh Amazon EKS untuk berkomunikasi dan mengelola cluster lokal di akun Anda.
- **`logs`**— Memungkinkan instance untuk mendorong log ke Amazon CloudWatch.
- **`secretsmanager`**— Memungkinkan instance untuk mendapatkan dan menghapus data bootstrap untuk instance bidang kontrol dengan aman. AWS Secrets Manager
- **`ecr`**— Memungkinkan Pods dan wadah yang berjalan pada instance bidang kontrol untuk menarik gambar kontainer yang disimpan di Amazon Elastic Container Registry.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat LocalOutpostClusterPolicyAmazonEks di Panduan Referensi Kebijakan AWS Terkelola.

AWS kebijakan terkelola: AmazonEks LocalOutpostServiceRolePolicy

Anda tidak dapat melampirkan kebijakan ini ke entitas IAM Anda. Saat Anda membuat klaster menggunakan prinsipal IAM yang memiliki `iam:CreateServiceLinkedRole` izin, Amazon EKS secara otomatis membuat peran `AWSServiceRoleforAmazonEKSLocalOutpost`terkait layanan untuk Anda dan melampirkan kebijakan ini padanya. Kebijakan ini memungkinkan peran terkait layanan untuk memanggil AWS layanan atas nama Anda untuk kluster lokal.

`AmazonEKSLocalOutpostServiceRolePolicy`Termasuk izin berikut:

- **`ec2`**— Memungkinkan Amazon EKS bekerja dengan keamanan, jaringan, dan sumber daya lainnya agar berhasil meluncurkan dan mengelola instans pesawat kontrol di akun Anda.
- **`ssm`**— Memungkinkan koneksi Amazon EC2 Systems Manager ke instans control plane, yang digunakan oleh Amazon EKS untuk berkomunikasi dan mengelola cluster lokal di akun Anda.
- **`iam`**— Memungkinkan Amazon EKS mengelola profil instans yang terkait dengan instans bidang kontrol.
- **`secretsmanager`**— Memungkinkan Amazon EKS memasukkan data bootstrap untuk instance bidang kontrol AWS Secrets Manager sehingga dapat direferensikan dengan aman selama bootstrap instance.
- **`outposts`**— Memungkinkan Amazon EKS mendapatkan informasi Outpost dari akun Anda agar berhasil meluncurkan cluster lokal di Outpost.

Untuk melihat versi terbaru dari dokumen kebijakan JSON, lihat LocalOutpostServiceRolePolicyAmazonEks di Panduan Referensi Kebijakan AWS Terkelola.

Amazon EKS memperbarui kebijakan AWS terkelola

Lihat detail tentang pembaruan kebijakan AWS terkelola untuk Amazon EKS sejak layanan ini mulai melacak perubahan ini. Untuk mendapatkan pemberitahuan otomatis tentang perubahan pada halaman ini, Anda bisa berlangganan umpan RSS di halaman riwayat Dokumen Amazon EKS.

Perubahan	Deskripsi	Tanggal
Amazoneks_CNI_POLICY - Update ke kebijakan yang ada	Amazon EKS menambahkan `ec2:DescribeSubnets` izin baru untuk memungkinkan Anda melihat jumlah alamat IP gratis di subnet VPC Amazon Anda. Amazon VPC CNI plugin for Kubernetes VPC CNI dapat menggunakan alamat IP gratis di setiap subnet untuk memilih subnet dengan alamat IP paling gratis untuk digunakan saat membuat elastic network interface.	Maret 4, 2024
WorkerNodePolicyAmazoneks - Perbarui ke kebijakan yang ada	Amazon EKS menambahkan izin baru untuk mengizinkan Identitas Pod EKS. Amazon EKS Pod Identity Agent menggunakan peran node.	26 November 2023
Memperkenalkan AmazonefS CSI DriverPolicy.	AWS memperkenalkan`AmazonEFS CSIDriverPolicy` .	26 Juli 2023
Menambahkan izin ke ClusterPolicyAmazoneks.	Menambahkan `ec2:DescribeAvailabilityZones` izin untuk mengizinkan Amazon EKS mendapatkan detail AZ selama penemuan otomatis subnet sambil membuat penyeimbang beban.	7 Februari 2023

Perubahan	Deskripsi	Tanggal
Kondisi kebijakan yang diperbarui di DriverPolicyAmazonEBSCSI.	Menghapus kondisi kebijakan yang tidak valid dengan karakter wildcard di bidang kunci. `StringLike` Juga menambahkan kondisi `ec2:ResourceTag/kubernetes.io/created-for/pvc/name: "*"` baru`ec2:DeleteVolume` , yang memungkinkan driver EBS CSI untuk menghapus volume yang dibuat oleh plugin in-tree.	17 November 2022
Menambahkan izin ke LocalOutpostServiceRolePolicyAmazoneks.	Ditambahkan`ec2:DescribeVPCAttribute` , `ec2:GetConsoleOutput` dan `ec2:DescribeSecret` untuk memungkinkan validasi prasyarat yang lebih baik dan kontrol siklus hidup terkelola. Juga ditambahkan `ec2:DescribePlacementGroups` dan `"arn:aws:ec2:*:*:placement-group/*"` `ec2:RunInstances` untuk mendukung kontrol penempatan pesawat kontrol instans Amazon EC2 di Outposts.	24 Oktober 2022
Perbarui izin Amazon Elastic Container Registry di LocalOutpostClusterPolicyAmazonEks.	Tindakan yang dipindahkan `ecr:GetDownloadUrlForLayer` dari semua bagian sumber daya ke bagian cakupan. Menambahkan sumber daya`arn:aws:ecr:*:*:repository/eks/*` . Sumber daya yang dihapus`arn:aws:ecr:*:*:repository/eks/eks-certificates-controller-public` . Sumber daya ini dicakup oleh `arn:aws:ecr:*:*:repository/eks/*` sumber daya tambahan.	20 Oktober 2022

Perubahan	Deskripsi	Tanggal
Menambahkan izin ke LocalOutpostClusterPolicyAmazoneks.	Menambahkan repositori `arn:aws:ecr:*:*:repository/kubelet-config-updater` Amazon Elastic Container Registry sehingga instance bidang kontrol cluster dapat memperbarui beberapa argumen. `kubelet`	31 Agustus 2022
Memperkenalkan AmazonEksLocalOutpostClusterPolicy.	AWS memperkenalkan`AmazonEKSLocalOutpostClusterPolicy` .	Agustus 24, 2022
Memperkenalkan AmazonEksLocalOutpostServiceRolePolicy.	AWS memperkenalkan`AmazonEKSLocalOutpostServiceRolePolicy` .	23 Agustus 2022
Memperkenalkan DriverPolicyAmazonEBSCSI.	AWS memperkenalkan`AmazonEBSCSIDriverPolicy` .	4 April 2022
Menambahkan izin ke WorkerNodePolicyAmazoneks.	Ditambahkan `ec2:DescribeInstanceTypes` untuk mengaktifkan AMI yang dioptimalkan Amazon EKS yang dapat menemukan properti tingkat instans secara otomatis.	Maret 21, 2022
Menambahkan izin ke AWSServiceRoleForAmazonEKSNodegroup.	Menambahkan `autoscaling:EnableMetricsCollection` izin untuk mengizinkan Amazon EKS mengaktifkan pengumpulan metrik.	13 Desember 2021
Menambahkan izin ke ClusterPolicyAmazoneks.	Izin `ec2:DescribeAccountAttributes` , `ec2:DescribeAddresses` , dan `ec2:DescribeInternetGateways` ditambahkan agar Amazon EKS diizinkan untuk membuat peran tertaut layanan bagi Penyeimbang Beban Jaringan.	17 Juni 2021

Perubahan	Deskripsi	Tanggal
Amazon EKS mulai melacak perubahan.	Amazon EKS mulai melacak perubahan untuk kebijakan yang AWS dikelola.	17 Juni 2021

Menyelesaikan masalah IAM

Topik ini mencakup beberapa kesalahan umum yang mungkin Anda temui saat menggunakan Amazon EKS dengan IAM dan cara yang dilakukan untuk mengatasinya.

AccessDeniedException

Jika Anda menerima `AccessDeniedException` saat memanggil operasi AWS API, kredensyal utama IAM yang Anda gunakan tidak memiliki izin yang diperlukan untuk melakukan panggilan itu.

```
An error occurred (AccessDeniedException) when calling the DescribeCluster operation:
User: arn:aws:iam::111122223333:user/user_name is not authorized to perform:
eks:DescribeCluster on resource: arn:aws:eks:region:111122223333:cluster/my-cluster
```

Dalam pesan contoh sebelumnya, pengguna tidak memiliki izin untuk memanggil operasi Amazon EKS `DescribeCluster` API. Untuk memberikan izin admin Amazon EKS ke kepala sekolah IAM, lihat. Contoh kebijakan berbasis identitas Amazon EKS

Untuk informasi umum selengkapnya tentang IAM, tinjau Mengontrol akses menggunakan kebijakan di Panduan Pengguna IAM.

Tidak dapat melihat Node di tab Compute atau apa pun di tab Resources dan Anda menerima kesalahan di AWS Management Console

Anda mungkin melihat pesan kesalahan konsol yang menyatakan `Your current user or role does not have access to Kubernetes objects on this EKS cluster`. Pastikan bahwa pengguna utama IAM yang Anda gunakan AWS Management Console dengan memiliki izin yang diperlukan. Untuk informasi selengkapnya, lihat Izin yang diperlukan.

aws-auth **`ConfigMap`** tidak memberikan akses ke cluster

AWS IAM Authenticator tidak mengizinkan jalur dalam peran ARN yang digunakan dalam. `ConfigMap` Karena itu, sebelum Anda tentukan `rolearn`, hapus jalur. Misalnya,

perubahan `arn:aws:iam::111122223333:role/team/developers/eks-admin` ke `arn:aws:iam::111122223333:role/eks-admin`.

Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan bahwa Anda tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran ke Amazon EKS.

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 mendapatkan peran ke layanan.

Contoh kesalahan berikut terjadi saat pengguna IAM bernama `marymajor` mencoba menggunakan konsol tersebut untuk performa tindakan di Amazon EKS. 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 di luar AWS akun saya untuk mengakses sumber daya Amazon EKS saya

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau pengguna 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 (ACL), Anda dapat menggunakan kebijakan tersebut untuk memberi pengguna akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, konsultasikan hal berikut:

- Untuk mempelajari apakah Amazon EKS mendukung fitur ini, lihat Bagaimana cara Amazon EKS bekerja sama 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 Memberikan akses kepada pengguna eksternal yang sah (federasi identitas) dalam Panduan Pengguna IAM.
- Untuk mempelajari perbedaan antara penggunaan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat Perbedaan antara peran IAM dan kebijakan berbasis sumber daya di Panduan Pengguna IAM.

Kontainer pod menerima kesalahan berikut: **`An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region`**

Container Anda menerima error ini jika aplikasi Anda secara eksplisit membuat permintaan ke endpoint AWS STS global (`https://sts.amazonaws`) dan akun Kubernetes layanan Anda dikonfigurasi untuk menggunakan endpoint regional. Anda dapat menyelesaikan masalah dengan salah satu opsi berikut:

- Perbarui kode aplikasi Anda untuk menghapus panggilan eksplisit ke titik akhir AWS STS global.
- Perbarui kode aplikasi Anda untuk membuat panggilan eksplisit ke titik akhir regional seperti. `https://sts.us-west-2.amazonaws.com` Aplikasi Anda harus memiliki redundansi bawaan untuk memilih Wilayah AWS yang berbeda jika terjadi kegagalan layanan di. Wilayah AWS Untuk informasi selengkapnya, lihat Mengelola AWS STSWilayah AWS dalam Panduan Pengguna IAM.
- Konfigurasikan akun layanan Anda untuk menggunakan titik akhir global. Semua versi sebelumnya `1.22` menggunakan titik akhir global secara default, tetapi versi `1.22` dan kluster yang lebih baru menggunakan titik akhir regional secara default. Untuk informasi selengkapnya, lihat Konfigurasikan AWS Security Token Service titik akhir untuk akun layanan.

Default Amazon EKS membuat Kubernetes peran dan pengguna

Saat Anda membuat Kubernetes cluster, beberapa Kubernetes identitas default dibuat di cluster itu agar berfungsi dengan baik. Kubernetes Amazon EKS membuat Kubernetes identitas untuk setiap

komponen defaultnya. Identitas menyediakan kontrol otorisasi Kubernetes berbasis peran (RBAC) untuk komponen cluster. Untuk informasi selengkapnya, lihat Menggunakan Otorisasi RBAC dalam dokumentasi. Kubernetes

Saat Anda menginstal add-on opsional ke klaster Anda, Kubernetes identitas tambahan mungkin ditambahkan ke klaster Anda. Untuk informasi selengkapnya tentang identitas yang tidak dibahas oleh topik ini, lihat dokumentasi untuk add-on.

Anda dapat melihat daftar Kubernetes identitas Amazon EKS yang dibuat di cluster Anda menggunakan alat baris `kubectl` perintah AWS Management Console atau. Semua identitas pengguna muncul di log `kube` audit yang tersedia untuk Anda melalui Amazon CloudWatch.

AWS Management Console

Prasyarat

Prinsipal IAM yang Anda gunakan harus memiliki izin yang dijelaskan dalam. Izin yang diperlukan

Untuk melihat identitas yang dibuat Amazon EKS menggunakan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Dalam daftar Clusters, pilih cluster yang berisi identitas yang ingin Anda lihat.
3. Pilih tab Sumber Daya.
4. Di bawah Jenis sumber daya, pilih Otorisasi.
5. Pilih, ClusterRoles, ClusterRoleBindings, Peran, atau RoleBindings. Semua sumber daya yang diawali dengan eks dibuat oleh Amazon EKS. Sumber daya identitas tambahan yang dibuat Amazon EKS adalah:

   - ClusterRoleDan ClusterRoleBindingdiberi nama aws-node. Sumber daya aws-node mendukung Amazon VPC CNI plugin for Kubernetes, yang diinstal Amazon EKS di semua cluster.
   - Yang ClusterRolebernama vpc-resource-controller-roledan ClusterRoleBindingbernama vpc-resource-controller-rolebinding. Sumber daya ini mendukung pengontrol sumber daya VPC Amazon, yang dipasang Amazon EKS di semua cluster.

   Selain sumber daya yang Anda lihat di konsol, identitas pengguna khusus berikut ada di klaster Anda, meskipun tidak terlihat dalam konfigurasi cluster:

   - **`eks:cluster-bootstrap`**- Digunakan untuk `kubectl` operasi selama bootstrap cluster.

- **`eks:support-engineer`**— Digunakan untuk operasi manajemen cluster.

6. Pilih sumber daya tertentu untuk melihat detailnya. Secara default, Anda ditampilkan informasi dalam tampilan Terstruktur. Di sudut kanan atas halaman detail, Anda dapat memilih Tampilan mentah untuk melihat semua informasi sumber daya.

Kubectl

Prasyarat

Entitas yang Anda gunakan (AWS Identity and Access Management (IAM) atau OpenID Connect ()OIDC) untuk mencantumkan Kubernetes sumber daya di klaster harus diautentikasi oleh IAM atau penyedia identitas AndaOIDC. Entitas harus diberikan izin untuk menggunakan `list` kata kerja Kubernetes `get` dan untuk`Role`, `ClusterRoleRoleBinding`, dan `ClusterRoleBinding` sumber daya di klaster yang Anda inginkan untuk dikerjakan oleh entitas tersebut. Untuk informasi selengkapnya tentang pemberian entitas IAM akses ke klaster Anda, lihat. the section called “Berikan akses ke API Kubernetes” Untuk informasi selengkapnya tentang pemberian entitas yang diautentikasi oleh OIDC penyedia Anda sendiri akses ke klaster Anda, lihat. Mengautentikasi pengguna untuk klaster Anda dari penyedia OpenID Connect identitas

Untuk melihat identitas Amazon EKS yang dibuat menggunakan **`kubectl`**

Jalankan perintah untuk jenis sumber daya yang ingin Anda lihat. Semua sumber daya yang dikembalikan yang diawali dengan eks dibuat oleh Amazon EKS. Selain sumber daya yang dikembalikan dalam output dari perintah, identitas pengguna khusus berikut ada di cluster Anda, meskipun tidak terlihat dalam konfigurasi cluster:

- **`eks:cluster-bootstrap`**- Digunakan untuk `kubectl` operasi selama bootstrap cluster.
- **`eks:support-engineer`**— Digunakan untuk operasi manajemen cluster.

ClusterRoles— `ClusterRoles` dicakup ke klaster Anda, jadi izin apa pun yang diberikan untuk peran berlaku untuk sumber daya di Kubernetes namespace apa pun di cluster.

Perintah berikut mengembalikan semua Amazon EKS yang dibuat Kubernetes `ClusterRoles` di cluster Anda.

```
kubectl get clusterroles | grep eks
```

Selain `ClusterRoles` dikembalikan dalam output yang diawali dengan, berikut ini `ClusterRoles` ada.

- **`aws-node`**— Ini `ClusterRole` mendukung Amazon VPC CNI plugin for Kubernetes, yang dipasang Amazon EKS di semua cluster.
- **`vpc-resource-controller-role`**— Ini `ClusterRole` mendukung pengontrol sumber daya Amazon VPC, yang dipasang Amazon EKS di semua cluster.

Untuk melihat spesifikasi untuk a`ClusterRole`, ganti *`eks:k8s-metrics`* dalam perintah berikut dengan `ClusterRole` return in output dari perintah sebelumnya. Contoh berikut mengembalikan spesifikasi untuk *`ClusterRoleeks:k8s-metrics`*.

```
kubectl describe clusterrole eks:k8s-metrics
```

Contoh output adalah sebagai berikut.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources         Non-Resource URLs  Resource Names  Verbs
  ---------         -----------------  --------------  -----
                    [/metrics]         []              [get]
  endpoints         []                 []              [list]
  nodes             []                 []              [list]
  pods              []                 []              [list]
  deployments.apps  []                 []              [list]
```

ClusterRoleBindings— `ClusterRoleBindings` dicakup ke cluster Anda.

Perintah berikut mengembalikan semua Amazon EKS yang dibuat Kubernetes `ClusterRoleBindings` di cluster Anda.

```
kubectl get clusterrolebindings | grep eks
```

Selain `ClusterRoleBindings` dikembalikan dalam output, berikut ini `ClusterRoleBindings` ada.

- **aws-node**— Ini `ClusterRoleBinding` mendukung Amazon VPC CNI plugin for Kubernetes, yang dipasang Amazon EKS di semua cluster.
- **vpc-resource-controller-rolebinding**— Ini `ClusterRoleBinding` mendukung pengontrol sumber daya Amazon VPC, yang dipasang Amazon EKS di semua cluster.

Untuk melihat spesifikasi untuk a`ClusterRoleBinding`, ganti *`eks:k8s-metrics`* dalam perintah berikut dengan `ClusterRoleBinding` return in output dari perintah sebelumnya. Contoh berikut mengembalikan spesifikasi untuk *`ClusterRoleBindingeks:k8s-metrics`*.

```
kubectl describe clusterrolebinding eks:k8s-metrics
```

Contoh output adalah sebagai berikut.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  eks:k8s-metrics
Subjects:
  Kind  Name             Namespace
  ----  ----             ---------
  User  eks:k8s-metrics
```

Peran — `Roles` dicakup ke namespace. Kubernetes Semua Amazon EKS `Roles` yang dibuat dicakup ke namespace. `kube-system`

Perintah berikut mengembalikan semua Amazon EKS yang dibuat Kubernetes `Roles` di cluster Anda.

```
kubectl get roles -n kube-system | grep eks
```

Untuk melihat spesifikasi untuk a`Role`, ganti *`eks:k8s-metrics`* dalam perintah berikut dengan nama yang `Role` dikembalikan dalam output dari perintah sebelumnya. Contoh berikut mengembalikan spesifikasi untuk *`Roleeks:k8s-metrics`*.

```
kubectl describe role eks:k8s-metrics -n kube-system
```

Contoh output adalah sebagai berikut.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources         Non-Resource URLs  Resource Names             Verbs
  ---------         -----------------  --------------             -----
  daemonsets.apps   []                 [aws-node]                 [get]
  deployments.apps  []                 [vpc-resource-controller]  [get]
```

RoleBindings— `RoleBindings` dicakup ke namespace. Kubernetes Semua Amazon EKS `RoleBindings` yang dibuat dicakup ke namespace. `kube-system`

Perintah berikut mengembalikan semua Amazon EKS yang dibuat Kubernetes `RoleBindings` di cluster Anda.

```
kubectl get rolebindings -n kube-system | grep eks
```

Untuk melihat spesifikasi untuk a`RoleBinding`, ganti *`eks:k8s-metrics`* dalam perintah berikut dengan `RoleBinding` return in output dari perintah sebelumnya. Contoh berikut mengembalikan spesifikasi untuk *`RoleBindingeks:k8s-metrics`*.

```
kubectl describe rolebinding eks:k8s-metrics -n kube-system
```

Contoh output adalah sebagai berikut.

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  Role
  Name:  eks:k8s-metrics
Subjects:
  Kind  Name             Namespace
  ----  ----             ---------
  User  eks:k8s-metrics
```

Validasi kepatuhan untuk Amazon Elastic Kubernetes Service

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

Anda bisa 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, serta hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- Panduan Memulai Cepat Keamanan dan Kepatuhan — Panduan penerapan ini membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar AWS yang berfokus pada keamanan dan kepatuhan.
- Arsitektur untuk Keamanan dan Kepatuhan HIPAA di Amazon Web Services — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat aplikasi yang memenuhi syarat HIPAA.

  > **Note**
  >
  > Tidak semua Layanan AWS memenuhi syarat HIPAA. Untuk informasi selengkapnya, lihat Referensi Layanan yang Memenuhi Syarat HIPAA.

- Sumber Daya Kepatuhan AWS – Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- AWSPanduan 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 di Panduan Developer AWS Config – Layanan AWS Config menilai seberapa baik konfigurasi sumber daya Anda dalam mematuhi praktik-praktik internal, pedoman industri, dan regulasi internal.
- AWS Security Hub— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnyaAWS. Security Hub menggunakan kontrol keamanan untuk

mengevaluasi AWS sumber daya Anda dan untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat referensi kontrol Security Hub.

- AWS Audit Manager – Layanan AWS ini akan membantu Anda untuk terus-menerus mengaudit penggunaan AWS untuk menyederhanakan bagaimana Anda mengelola risiko dan kepatuhan terhadap regulasi dan standar industri.

Ketahanan dalam Amazon EKS

Infrastruktur global AWS dibangun di sekitar Wilayah AWS dan Availability Zone. Wilayah AWS menyediakan beberapa Availability Zone yang terpisah secara fisik dan terisolasi yang terhubung dengan jaringan latensi rendah, throughput tinggi, dan jaringan yang sangat berlebihan. Dengan Availability Zone, Anda dapat merancang dan mengoperasikan aplikasi dan basis data yang secara otomatis melakukan failover di antara Availability Zone tanpa gangguan. Availability Zone memiliki ketersediaan yang lebih baik, menoleransi kegagalan, dan dapat diskalakan dibandingkan satu atau beberapa infrastruktur pusat data tradisional.

Amazon EKS menjalankan dan menskalakan bidang Kubernetes kontrol di beberapa AWS Availability Zone untuk memastikan ketersediaan yang tinggi. Amazon EKS secara otomatis menskalakan instans pesawat kontrol berdasarkan beban, mendeteksi, dan mengganti instans bidang kontrol yang tidak sehat, dan secara otomatis menambal bidang kontrol. Setelah Anda memulai pembaruan versi, Amazon EKS memperbarui bidang kontrol untuk Anda, menjaga ketersediaan tinggi bidang kontrol selama pembaruan.

Bidang kontrol ini terdiri dari setidaknya dua instance server API dan tiga `etcd` instance yang berjalan di tiga Availability Zone dalam file. Wilayah AWS Amazon EKS:

- Secara aktif memantau beban pada instans bidang kendali dan secara otomatis menskalakannya guna memastikan kinerja yang tinggi.
- Secara otomatis mendeteksi dan mengganti instance bidang kontrol yang tidak sehat, memulai ulang di seluruh Availability Zone sesuai kebutuhan. Wilayah AWS
- Memanfaatkan arsitektur Wilayah AWS untuk menjaga ketersediaan tinggi. Karena itu, Amazon EKS mampu menawarkan SLA untuk ketersediaan titik akhir server API.

Untuk informasi selengkapnya tentang Wilayah AWS dan Availability Zone, lihat infrastruktur AWS global.

Keamanan infrastruktur di Amazon EKS

Sebagai layanan terkelola, Amazon Elastic Kubernetes Service dilindungi oleh keamanan jaringan global. AWS 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 Amazon EKS 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 dengan menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan pengguna utama IAM. Atau Anda dapat menggunakan AWS Security Token Service (AWS STS) untuk membuat kredensial keamanan sementara untuk menandatangani permintaan.

Ketika Anda membuat klaster Amazon EKS, Anda menentukan subnet VPC untuk digunakan oleh klaster Anda. Amazon EKS membutuhkan subnet, setidaknya di dua Availability Zone. Kami merekomendasikan VPC dengan subnet publik dan pribadi sehingga Kubernetes dapat membuat penyeimbang beban publik di subnet publik yang memuat lalu lintas keseimbangan untuk Pods berjalan di node yang berada di subnet pribadi.

Untuk informasi selengkapnya tentang pertimbangan VPC, lihat Persyaratan dan pertimbangan Amazon EKS VPC dan subnet.

Jika Anda membuat grup VPC dan node dengan AWS CloudFormation templat yang disediakan dalam Memulai dengan Amazon EKS panduan, maka grup keamanan bidang kontrol dan node Anda dikonfigurasi dengan pengaturan yang kami rekomendasikan.

Untuk informasi selengkapnya tentang pertimbangan grup keamanan, lihat Persyaratan dan pertimbangan grup keamanan Amazon EKS.

Saat Anda membuat klaster baru, Amazon EKS membuat titik akhir untuk server Kubernetes API terkelola yang Anda gunakan untuk berkomunikasi dengan klaster Anda (menggunakan alat

Kubernetes manajemen sepertikubectl). Secara default, endpoint server API ini bersifat publik ke internet, dan akses ke server API diamankan menggunakan kombinasi AWS Identity and Access Management (IAM) dan Kubernetes Role Based Access Control (RBAC) asli.

Anda dapat mengaktifkan akses pribadi ke server Kubernetes API sehingga semua komunikasi antara node dan server API tetap berada dalam VPC Anda. Anda dapat membatasi alamat IP yang dapat mengakses server API Anda dari internet, atau menonaktifkan sepenuhnya akses internet ke server API.

Untuk informasi selengkapnya tentang cara memodifikasi akses titik akhir klaster, lihat Memodifikasi akses titik akhir klaster.

Anda dapat menerapkan kebijakan Kubernetes jaringan dengan Amazon VPC CNI atau alat pihak ketiga seperti Project. Calico Untuk informasi selengkapnya tentang menggunakan Amazon VPC CNI untuk kebijakan jaringan, lihat. Konfigurasikan klaster Anda untuk kebijakan Kubernetes jaringan Proyek Calico adalah proyek open source pihak ketiga. Untuk informasi selengkapnya, lihat Calicodokumentasi Proyek.

Analisis konfigurasi dan kelemahan di Amazon EKS

Keamanan adalah pertimbangan penting untuk mengonfigurasi dan memeliharaKubernetescluster dan aplikasi. ParameterPusat Keamanan Internet (CIS)KubernetesPatokanmenyediakan panduan untuk konfigurasi keamanan simpul Amazon EKS. Tolok ukur:

- Berlaku untuk simpul Amazon EC2 (baik yang dikelola maupun swakelola) di mana Anda bertanggung jawab atas konfigurasi keamananKuberneteskomponen.
- Menyediakan cara standar yang disetujui komunitas untuk memastikan bahwa Anda telah mengonfigurasiKubernetescluster dan node dengan aman saat menggunakan Amazon EKS.
- Terdiri dari empat bagian; konfigurasi pencatatan bidang kendali, konfigurasi keamanan simpul, kebijakan, dan layanan terkelola.
- Mendukung semuaKubernetesversi yang saat ini tersedia di Amazon EKS dan dapat dijalankan menggunakankube-bangku, alat open source standar untuk memeriksa konfigurasi menggunakan benchmark CISKubernetes kluster.

Untuk mempelajari selengkapnya, lihat Memperkenalkan Tolok Ukur CIS Amazon EKS.

Versi platform Amazon EKS mewakili kemampuan bidang kendali klaster, termasuk kemampuan bidang kendali klaster, termasukKubernetesBendera server API diaktifkan dan saat

iniKubernetesVersi patch. Klaster baru di-deploy dengan versi platform terbaru. Untuk detail, lihat Amazon EKS versi platform.

Anda dapatmemperbarui klaster Amazon EKSke yang lebih baruKubernetesversi. Sebagai baruKubernetesversi tersedia di Amazon EKS, kami sarankan supaya Anda memperbarui klaster secara proaktif untuk menggunakan versi terbaru yang tersedia. Untuk informasi lebih lanjut tentangKubernetesversi di EKS, lihatKubernetesVersi Amazon EKS.

Melacak peristiwa keamanan atau privasi untuk AmazonLinux2 diAmazonLinuxPusat Keamananatau berlangganan yang terkaitUmpan RSS. Kejadian keamanan dan privasi mencakup gambaran umum masalah yang terpengaruh, paket, dan petunjuk untuk memperbarui instans Anda guna memperbaiki masalah.

Anda dapat menggunakan Amazon Inspector untuk memeriksa aksesibilitas jaringan yang tidak diinginkan dari simpul Anda dan kelemahan pada instans Amazon EC2 tersebut.

Praktik terbaik keamanan untuk Amazon EKS

Praktik terbaik keamanan Amazon EKS dipertahankan di Github:https://aws.github.io/aws-eks-best-practices/keamanan/docs/

Kebijakan keamanan pod

Pengontrol penerimaan kebijakan Kubernetes Pod keamanan memvalidasi permintaan Pod pembuatan dan pembaruan terhadap seperangkat aturan. Secara default, klaster Amazon EKS dikirimkan dengan kebijakan keamanan yang sepenuhnya permisif tanpa batasan. Untuk informasi selengkapnya, lihat Kebijakan Keamanan Pod di Kubernetes dokumentasi.

> **Note**
>
> The `PodSecurityPolicy` (PSP) tidak digunakan lagi dalam Kubernetes versi `1.21` dan dihapus di. Kubernetes `1.25` PSPssedang diganti dengan Pod Security Admission (PSA), sebuah pengontrol masuk bawaan yang mengimplementasikan kontrol keamanan yang diuraikan dalam Pod Security Standards (PSS). PSA dan PSS keduanya telah mencapai status fitur beta, dan diaktifkan di Amazon EKS secara default. Untuk mengatasi PSP penghapusan di`1.25`, kami sarankan Anda menerapkan PSS di Amazon EKS. Untuk informasi selengkapnya, lihat Menerapkan Standar Keamanan Pod di Amazon EKS di AWS blog.

Kebijakan Pod keamanan default Amazon EKS

Cluster Amazon EKS dengan Kubernetes versi `1.13` atau lebih tinggi memiliki kebijakan Pod keamanan default bernama`eks.privileged`. Kebijakan ini tidak memiliki batasan pada jenis apa yang Pod dapat diterima ke dalam sistem, yang setara dengan berjalan Kubernetes dengan `PodSecurityPolicy` pengontrol dinonaktifkan.

> **Note**
>
> Kebijakan ini dibuat untuk memelihara kompatibilitas ke belakang dengan klaster yang tidak memiliki pengendali `PodSecurityPolicy` yang aktif. Anda dapat membuat kebijakan yang lebih ketat untuk klaster Anda, juga untuk namespace individual serta akun layanan, dan kemudian hapus kebijakan default untuk mengaktifkan kebijakan yang lebih ketat.

Anda dapat melihat kebijakan default dengan perintah berikut.

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

Contoh output adalah sebagai berikut.

```
NAME             PRIV    CAPS    SELINUX    RUNASUSER   FSGROUP    SUPGROUP
 READONLYROOTFS   VOLUMES
eks.privileged   true    *       RunAsAny   RunAsAny    RunAsAny   RunAsAny   false
      *
```

Untuk lebih jelasnya, Anda dapat mendeskripsikan kebijakan dengan perintah berikut.

```
kubectl describe psp eks.privileged
```

Contoh output adalah sebagai berikut.

```
Name:  eks.privileged

Settings:
  Allow Privileged:                       true
  Allow Privilege Escalation:             0xc0004ce5f8
  Default Add Capabilities:               <none>
  Required Drop Capabilities:             <none>
```

```
  Allowed Capabilities:                  *
  Allowed Volume Types:                  *
  Allow Host Network:                    true
  Allow Host Ports:                      0-65535
  Allow Host PID:                        true
  Allow Host IPC:                        true
  Read Only Root Filesystem:             false
  SELinux Context Strategy: RunAsAny
    User:                                <none>
    Role:                                <none>
    Type:                                <none>
    Level:                               <none>
  Run As User Strategy: RunAsAny
    Ranges:                              <none>
  FSGroup Strategy: RunAsAny
    Ranges:                              <none>
  Supplemental Groups Strategy: RunAsAny
    Ranges:                              <none>
```

Anda dapat melihat file YAMB lengkap untuk kebijakan `eks.privileged` Pod keamanan, peran klaster, dan peran klaster yang mengikat. Instal atau pulihkan kebijakan Pod keamanan default

Hapus kebijakan Pod keamanan Amazon EKS default

Jika Anda membuat kebijakan yang lebih ketat untuk AndaPods, maka setelah melakukannya, Anda dapat menghapus kebijakan `eks.privileged` Pod keamanan Amazon EKS default untuk mengaktifkan kebijakan kustom Anda.

> **Important**
>
> Jika Anda menggunakan versi `1.7.0` atau yang lebih baru dari plugin CNI dan Anda menetapkan kebijakan Pod keamanan khusus ke akun `aws-node` Kubernetes layanan yang digunakan untuk `aws-node` Pods digunakan oleh Daemonset, maka kebijakan tersebut harus ada `NET_ADMIN` di `allowedCapabilities` bagiannya bersama dengan `hostNetwork: true` dan `privileged: true` di kebijakan tersebut. `spec`

Untuk menghapus kebijakan Pod keamanan default

1. Buat file bernama *`privileged-podsecuritypolicy.yaml`*dengan konten dalam file contoh diInstal atau pulihkan kebijakan Pod keamanan default.

2. Hapus YAML dengan perintah berikut. Ini menghapus kebijakan Pod keamanan default, yang`ClusterRole`, dan yang `ClusterRoleBinding` terkait dengannya.

```
kubectl delete -f privileged-podsecuritypolicy.yaml
```

Instal atau pulihkan kebijakan Pod keamanan default

Jika Anda memutakhirkan dari versi sebelumnyaKubernetes, atau telah memodifikasi atau menghapus kebijakan `eks.privileged` Pod keamanan Amazon EKS default, Anda dapat memulihkannya dengan langkah-langkah berikut.

Untuk menginstal atau memulihkan kebijakan Pod keamanan default

1. Buat file bernama *`privileged-podsecuritypolicy.yaml`* dengan konten berikut.

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: eks.privileged
  annotations:
    kubernetes.io/description: 'privileged allows full unrestricted access to
      Pod features, as if the PodSecurityPolicy controller was not enabled.'
    seccomp.security.alpha.kubernetes.io/allowedProfileNames: '*'
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
spec:
  privileged: true
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  volumes:
  - '*'
  hostNetwork: true
  hostPorts:
  - min: 0
    max: 65535
  hostIPC: true
  hostPID: true
  runAsUser:
    rule: 'RunAsAny'
```

```
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'RunAsAny'
  fsGroup:
    rule: 'RunAsAny'
  readOnlyRootFilesystem: false

---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:podsecuritypolicy:privileged
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
rules:
- apiGroups:
  - policy
  resourceNames:
  - eks.privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use

---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:podsecuritypolicy:authenticated
  annotations:
    kubernetes.io/description: 'Allow all authenticated users to create privileged
 Pods.'
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:podsecuritypolicy:privileged
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
```

```
    name: system:authenticated
```

2. Terapkan YAML dengan perintah berikut.

```
kubectl apply -f privileged-podsecuritypolicy.yaml
```

FAQ (PSP) penghapusan kebijakan keamanan Pod

`PodSecurityPolicy`tidak digunakan lagi di Kubernetes`1.21`, dan telah dihapus di. Kubernetes `1.25` Jika Anda menggunakan PodSecurityPolicy klaster Anda, maka Anda harus bermigrasi ke Standar Keamanan Kubernetes Pod bawaan (PSS) atau ke policy-as-code solusi sebelum memutakhirkan klaster Anda ke versi **`1.25`** untuk menghindari gangguan pada beban kerja Anda. Pilih pertanyaan yang sering diajukan untuk mempelajari lebih lanjut.

Apa ituPSP?

PodSecurityPolicyadalah pengontrol penerimaan bawaan yang memungkinkan administrator cluster untuk mengontrol aspek spesifikasi yang sensitif terhadap keamanan. Pod Jika Pod memenuhi persyaratan nyaPSP, Pod diterima ke cluster seperti biasa. Jika a Pod tidak memenuhi PSP persyaratan, Pod ditolak dan tidak dapat dijalankan.

Apakah PSP penghapusan khusus untuk Amazon EKS atau dihapus di hulu? Kubernetes

Ini adalah perubahan hulu dalam Kubernetes proyek, dan bukan perubahan yang dibuat di Amazon EKS. PSPtidak digunakan lagi Kubernetes `1.21` dan dihapus di. Kubernetes `1.25` KubernetesKomunitas mengidentifikasi masalah kegunaan yang serius denganPSP. Ini termasuk secara tidak sengaja memberikan izin yang lebih luas daripada yang dimaksudkan dan kesulitan dalam memeriksa yang PSPs berlaku dalam situasi tertentu. Masalah ini tidak dapat diatasi tanpa membuat perubahan yang melanggar. Ini adalah alasan utama mengapa Kubernetes masyarakat memutuskan untuk menghapus PSP.

Bagaimana saya bisa memeriksa apakah saya menggunakan PSPs di cluster Amazon EKS saya?

Untuk memeriksa apakah Anda menggunakan PSPs di cluster Anda, Anda dapat menjalankan perintah berikut:

```
kubectl get psp
```

Untuk melihat Pods bahwa PSPs di cluster Anda berdampak, jalankan perintah berikut. Perintah ini menampilkan Pod nama, namespace, dan: PSPs

```
kubectl get pod -A -o jsonpath='{range.items[?(@.metadata.annotations.kubernetes
\.io/psp)]}{.metadata.name}{"\t"}{.metadata.namespace}{"\t"}
{.metadata.annotations.kubernetes\.io/psp}{"\n"}'
```

Jika saya menggunakan PSPs di cluster Amazon EKS saya, apa yang dapat saya lakukan?

Sebelum memutakhirkan klaster ke`1.25`, Anda harus memigrasikan PSPs ke salah satu alternatif berikut:

- Kubernetes PSS.

- olicy-as-code Solusi P dari Kubernetes lingkungan.

Menanggapi PSP penghentian dan kebutuhan berkelanjutan untuk mengontrol Pod keamanan sejak awal, Kubernetes komunitas membuat solusi bawaan dengan (PSS) dan Pod Security Admission (PSA). Webhook PSA mengimplementasikan kontrol yang didefinisikan dalam file. PSS

Anda dapat meninjau praktik terbaik untuk bermigrasi PSPs ke built-in PSS dalam Panduan Praktik Terbaik EKS. Kami juga merekomendasikan untuk meninjau blog kami tentang Menerapkan Standar Keamanan Pod di Amazon EKS. Referensi tambahan termasuk Migrasi dari PodSecurityPolicy ke Built-in PodSecurity Admission Controller dan Mapping PodSecurityPolicies to Pod Security Standards.

olicy-as-code Solusi P menyediakan pagar pembatas untuk memandu pengguna klaster dan mencegah perilaku yang tidak diinginkan melalui kontrol otomatis yang ditentukan. olicy-as-code Solusi P biasanya menggunakan Kubernetes Dynamic Admission Controllers untuk mencegat alur permintaan server Kubernetes API menggunakan panggilan webhook. olicy-as-code Solusi P mengubah dan memvalidasi muatan permintaan berdasarkan kebijakan yang ditulis dan disimpan sebagai kode.

Ada beberapa policy-as-code solusi open source yang tersedia untukKubernetes. Untuk meninjau praktik terbaik migrasi PSPs ke policy-as-code solusi, lihat olicy-as-code bagian P pada GitHub halaman Keamanan Pod.

Saya melihat PSP panggilan **eks.privileged** di cluster saya. Apa itu dan apa yang bisa saya lakukan?

Cluster Amazon EKS dengan Kubernetes versi `1.13` atau lebih tinggi memiliki default PSP yang diberi `eks.privileged` nama. Kebijakan ini dibuat dalam `1.24` dan klaster sebelumnya. Ini tidak digunakan dalam `1.25` dan selanjutnya cluster. Amazon EKS secara otomatis memigrasikan ini PSP ke penegakan hukum PSS berbasis. Tidak ada tindakan yang diperlukan di pihak Anda.

Akankah Amazon EKS membuat perubahan apa pun untuk PSPs ditampilkan di cluster saya yang ada saat saya memperbarui cluster saya ke versi**1.25**?

Tidak. Selain itu`eks.privileged`, yang PSP dibuat oleh Amazon EKS, tidak ada perubahan yang dilakukan ke yang lain PSPs di cluster Anda saat Anda meningkatkan ke`1.25`.

Akankah Amazon EKS mencegah pembaruan cluster ke versi **1.25** jika saya belum bermigrasi? PSP

Tidak. Amazon EKS tidak akan mencegah pembaruan klaster ke versi `1.25` jika Anda PSP belum bermigrasi.

Bagaimana jika saya lupa memigrasikan PSPs ke PSS/PSA atau ke policy-as-code solusi sebelum memperbarui cluster saya ke versi? **1.25** Bisakah saya bermigrasi setelah memperbarui cluster saya?

Saat klaster yang berisi a PSP ditingkatkan ke Kubernetes versi`1.25`, server API tidak mengenali PSP sumber daya di dalamnya`1.25`. Hal ini dapat mengakibatkan Pods mendapatkan cakupan keamanan yang salah. Untuk daftar lengkap implikasi, lihat Memigrasi dari PodSecurityPolicy ke Pengontrol Penerimaan Bawaan. PodSecurity

Bagaimana perubahan ini memengaruhi keamanan pod untuk beban kerja Windows?

Kami tidak mengharapkan dampak spesifik apa pun pada beban kerja Windows. PodSecurityContext memiliki bidang yang disebut `windowsOptions` dalam `PodSpec` `v1` API untuk WindowsPods. Ini menggunakan PSS dalam Kubernetes`1.25`. Untuk informasi selengkapnya dan praktik terbaik

tentang penerapan beban kerja Windows, lihat Panduan dan Kubernetes dokumentasi Praktik Terbaik EKS. PSS

MenggunakanAWS Rahasia Secrets Manager denganKubernetes

Untuk menampilkan rahasia dari Secrets Manager dan parameter dari Parameter Store sebagai file yang dipasang di Amazon EKSPods, Anda dapat menggunakanAWS Secrets and Configuration Provider (ASCP) untuk KubernetesSecrets Store CSI Driver.

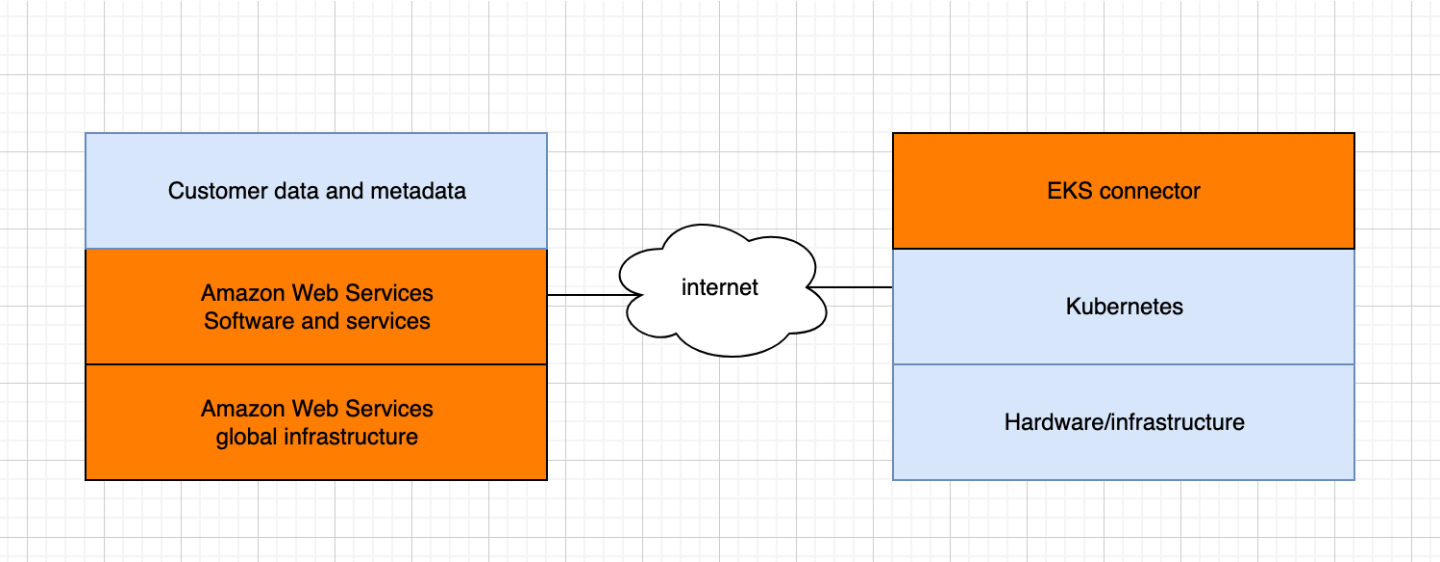
Dengan ASCP, Anda dapat menyimpan dan mengelola secret Anda dalam Secrets Manager, lalu mengambilnya kembali melalui beban kerja yang berjalan di Amazon EKS. Anda dapat menggunakan peran dan kebijakan IAM untuk membatasi akses ke rahasia AndaKubernetesPods secara spesifik dalam klaster. ASCP mengambilPod identitas dan pertukaran identitas untuk peran IAM. ASCP mengasumsikan peran IAM dariPod, dan kemudian dapat mengambil rahasia dari Secrets Manager yang berwenang untuk peran itu.

Jika Anda menggunakan rotasi otomatis Secrets Manager untuk rahasia Anda, Anda juga dapat menggunakan fitur rekonsiler rotasi Secrets Store CSI Driver untuk memastikan Anda mengambil rahasia terbaru dari Secrets Manager.

Untuk informasi lebih lanjut, lihat MenggunakanAWS Secrets Manager.

Pertimbangan Amazon EKS Connector

Amazon EKS Connector adalah komponen open source yang berjalan di Kubernetes cluster Anda. Cluster ini dapat ditempatkan di luar AWS lingkungan. Ini menciptakan pertimbangan tambahan untuk tanggung jawab keamanan. Konfigurasi ini dapat diilustrasikan dengan diagram berikut. Oranye mewakili AWS tanggung jawab, dan biru mewakili tanggung jawab pelanggan:



Topik ini menjelaskan perbedaan dalam model tanggung jawab jika cluster yang terhubung berada di luarAWS.

AWStanggung jawab

- Memelihara, membangun, dan mengirimkan Amazon EKS Connector, yang merupakan komponen open source yang berjalan di Kubernetes cluster pelanggan dan berkomunikasi dengannyaAWS.
- Menjaga keamanan komunikasi lapisan transportasi dan aplikasi antara Kubernetes cluster dan AWS layanan yang terhubung.

Tanggung jawab pelanggan

- Kuberneteskeamanan khusus cluster, khususnya di sepanjang baris berikut:
  - Kubernetesrahasia harus dienkripsi dan dilindungi dengan benar.
  - Kunci akses ke `eks-connector` namespace.
- Mengkonfigurasi izin kontrol akses berbasis peran (RBAC) untuk mengelola akses utama IAM dari. AWS Untuk petunjuk, silakan lihat Memberikan akses ke prinsipal IAM untuk melihat Kubernetes sumber daya di cluster.
- Menginstal dan meningkatkan Konektor Amazon EKS.
- Mempertahankan perangkat keras, perangkat lunak, dan infrastruktur yang mendukung Kubernetes cluster yang terhubung.
- Mengamankan AWS akun mereka (misalnya, dengan menjaga kredensi pengguna root Anda).

Lihat Kubernetes sumber daya

Anda dapat melihat Kubernetes sumber daya yang diterapkan ke cluster Anda dengan file. AWS Management Console Anda tidak dapat melihat Kubernetes sumber daya dengan AWS CLI atau `eksctl`. Untuk melihat Kubernetes sumber daya menggunakan alat baris perintah, gunakan. `kubectl`

Prasyarat

Untuk melihat tab Resources dan bagian Nodes pada tab Compute di AWS Management Console, prinsipal IAM yang Anda gunakan harus memiliki IAM dan izin tertentu. Kubernetes Untuk informasi selengkapnya, lihat Izin yang diperlukan.

Untuk melihat Kubernetes sumber daya dengan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Dalam daftar Clusters, pilih cluster yang berisi Kubernetes sumber daya yang ingin Anda lihat.
3. Pilih tab Sumber Daya.
4. Pilih grup tipe sumber daya yang ingin Anda lihat sumber daya, seperti Beban kerja. Anda melihat daftar jenis sumber daya di grup itu.
5. Pilih jenis sumber daya, seperti Deployment, di grup Beban kerja. Anda melihat deskripsi jenis sumber daya, tautan ke Kubernetes dokumentasi untuk informasi selengkapnya tentang jenis sumber daya, dan daftar sumber daya dari jenis tersebut yang digunakan di klaster Anda. Jika daftar kosong, maka tidak ada sumber daya dari jenis itu yang diterapkan ke cluster Anda.
6. Pilih sumber daya untuk melihat informasi lebih lanjut tentangnya. Coba contoh berikut:
   - Pilih grup Beban kerja, pilih jenis sumber daya Deployment, lalu pilih sumber daya coredns. Ketika Anda memilih sumber daya, Anda berada dalam tampilan Terstruktur, secara default. Untuk beberapa jenis sumber daya, Anda melihat bagian Pod dalam tampilan Terstruktur. Bagian ini mencantumkan yang Pods dikelola oleh beban kerja. Anda dapat memilih yang Pod terdaftar untuk melihat informasi tentangPod. Tidak semua jenis sumber daya menampilkan informasi dalam Tampilan Terstruktur. Jika Anda memilih tampilan Raw di sudut kanan atas halaman untuk sumber daya, Anda akan melihat respons JSON lengkap dari Kubernetes API untuk sumber daya.
   - Pilih grup Cluster dan kemudian pilih jenis sumber daya Node. Anda melihat daftar semua node di cluster Anda. Node dapat berupa jenis simpul Amazon EKS. Ini adalah daftar yang

sama yang Anda lihat di bagian Nodes ketika Anda memilih tab Compute untuk cluster Anda. Pilih sumber daya node dari daftar. Dalam tampilan Terstruktur, Anda juga melihat bagian Pod. Bagian ini menunjukkan Anda semua Pods berjalan pada node.

Izin yang diperlukan

Untuk melihat tab Resources dan bagian Nodes pada tab Compute di AWS Management Console, prinsipal IAM yang Anda gunakan harus memiliki IAM dan izin minimum tertentu. Kubernetes Selesaikan langkah-langkah berikut untuk menetapkan izin yang diperlukan untuk kepala sekolah IAM Anda.

1. Pastikan bahwa`eks:AccessKubernetesApi`, dan izin IAM lain yang diperlukan untuk melihat Kubernetes sumber daya, ditetapkan ke prinsipal IAM yang Anda gunakan. Untuk informasi selengkapnya tentang cara mengedit izin untuk prinsipal IAM, lihat Mengontrol akses untuk prinsipal di Panduan Pengguna IAM. Untuk informasi selengkapnya tentang cara mengedit izin untuk peran, lihat Memodifikasi kebijakan izin peran (konsol) di Panduan Pengguna IAM.

   Contoh kebijakan berikut mencakup izin yang diperlukan bagi prinsipal untuk melihat Kubernetes sumber daya untuk semua klaster di akun Anda. Ganti *111122223333* dengan ID akun AWS Anda.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": [
                   "eks:ListFargateProfiles",
                   "eks:DescribeNodegroup",
                   "eks:ListNodegroups",
                   "eks:ListUpdates",
                   "eks:AccessKubernetesApi",
                   "eks:ListAddons",
                   "eks:DescribeCluster",
                   "eks:DescribeAddonVersions",
                   "eks:ListClusters",
                   "eks:ListIdentityProviderConfigs",
                   "iam:ListRoles"
               ],
               "Resource": "*"
   ```

```
        },
        {
            "Effect": "Allow",
            "Action": "ssm:GetParameter",
            "Resource": "arn:aws:ssm:*:111122223333:parameter/*"
        }
    ]
}
```

Untuk melihat node dalam cluster yang terhubung, peran IAM konektor Amazon EKS harus dapat meniru prinsipal di cluster. Hal ini memungkinkan Konektor Amazon EKS untuk memetakan prinsipal ke Kubernetes pengguna.

2. Buat Kubernetes `rolebinding` atau `clusterrolebinding` yang terikat pada Kubernetes `role` atau `clusterrole` yang memiliki izin yang diperlukan untuk melihat Kubernetes sumber daya. Untuk mempelajari lebih lanjut tentang Kubernetes peran dan binding peran, lihat Menggunakan Otorisasi RBAC dalam dokumentasi. Kubernetes Anda dapat menerapkan salah satu manifes berikut ke cluster Anda yang membuat `role` dan `rolebinding` atau a `clusterrole` dan `clusterrolebinding` dengan Kubernetes izin yang diperlukan:

   Lihat Kubernetes sumber daya di semua ruang nama

   Nama grup dalam file tersebut adalah`eks-console-dashboard-full-access-group`. Terapkan manifes ke cluster Anda dengan perintah berikut:

   ```
   kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-full-access.yaml
   ```

   Lihat Kubernetes sumber daya di namespace tertentu

   Namespace dalam file ini adalah. `default` Nama grup dalam file tersebut adalah`eks-console-dashboard-restricted-access-group`. Terapkan manifes ke cluster Anda dengan perintah berikut:

   ```
   kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-restricted-access.yaml
   ```

   Jika Anda perlu mengubah nama Kubernetes grup, namespace, izin, atau konfigurasi lain dalam file, maka unduh file tersebut dan edit sebelum menerapkannya ke cluster Anda:

1. Unduh file dengan salah satu perintah berikut:

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-full-access.yaml
```

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-restricted-access.yaml
```

2. Edit file seperlunya.
3. Terapkan manifes ke cluster Anda dengan salah satu perintah berikut:

```
kubectl apply -f eks-console-full-access.yaml
```

```
kubectl apply -f eks-console-restricted-access.yaml
```

3. Petakan prinsipal IAM ke Kubernetes pengguna atau grup di. `aws-auth` `ConfigMap` Anda dapat menggunakan alat seperti `eksctl` untuk memperbarui `ConfigMap` atau Anda dapat memperbaruinya secara manual dengan mengeditnya.

> **Important**
>
> Kami merekomendasikan menggunakan`eksctl`, atau alat lain, untuk mengedit`ConfigMap`. Untuk informasi tentang alat lain yang dapat Anda gunakan, lihat Menggunakan alat untuk membuat perubahan `aws-auth`ConfigMap pada panduan praktik terbaik Amazon EKS. Format yang tidak benar `aws-auth` `ConfigMap` dapat menyebabkan Anda kehilangan akses ke cluster Anda.

eksctl

Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

1. Lihat pemetaan saat ini di. `ConfigMap` Ganti *`my-cluster`* dengan nama klaster Anda. Ganti *`region-code`* dengan tempat Wilayah AWS cluster Anda berada.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

Contoh output adalah sebagai berikut.

```
ARN
                  USERNAME                                GROUPS
            ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

2. Tambahkan pemetaan untuk peran. Contoh ini mengasumsikan bahwa Anda melampirkan izin IAM pada langkah pertama ke peran bernama. *my-console-viewer-role* Ganti *111122223333* dengan ID akun Anda.

```
eksctl create iamidentitymapping \
    --cluster my-cluster \
    --region=region-code \
    --arn arn:aws:iam::111122223333:role/my-console-viewer-role \
    --group eks-console-dashboard-full-access-group \
    --no-duplicate-arns
```

> **Important**
>
> Peran ARN tidak dapat menyertakan jalur seperti. `role/my-team/developers/my-role` Format ARN harus. `arn:aws:iam::111122223333:role/my-role` Dalam contoh ini, `my-team/developers/` perlu dihapus.

Contoh output adalah sebagai berikut.

```
[...]
2022-05-09 14:51:20 [#]  adding identity "arn:aws:iam::111122223333:role/my-
console-viewer-role" to auth ConfigMap
```

3. Tambahkan pemetaan untuk pengguna. Praktik terbaik IAM menyarankan agar Anda memberikan izin untuk peran, bukan pengguna. Contoh ini mengasumsikan bahwa Anda

melampirkan izin IAM pada langkah pertama ke pengguna bernama. *my-user* Ganti *111122223333* dengan ID akun Anda.

```
eksctl create iamidentitymapping \
    --cluster my-cluster \
    --region=region-code \
    --arn arn:aws:iam::111122223333:user/my-user \
    --group eks-console-dashboard-restricted-access-group \
    --no-duplicate-arns
```

Contoh output adalah sebagai berikut.

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws:iam::111122223333:user/my-
user" to auth ConfigMap
```

4. Lihat pemetaan di lagi. `ConfigMap`

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

Contoh output adalah sebagai berikut.

```
ARN
                  USERNAME                                GROUPS
                    ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
arn:aws:iam::111122223333:role/my-console-viewer-role
                                                          eks-console-
dashboard-full-access-group
arn:aws:iam::111122223333:user/my-user
                                                          eks-console-
dashboard-restricted-access-group
```

Edit ConfigMap manually

Untuk informasi selengkapnya tentang menambahkan pengguna atau peran ke dalam `aws-authConfigMap`, lihat[Tambahkan prinsipal IAM ke kluster Amazon EKS Anda](#).

1. Buka `aws-auth` `ConfigMap` untuk mengedit.

```
kubectl edit -n kube-system configmap/aws-auth
```

2. Tambahkan pemetaan ke `aws-auth``ConfigMap`, tetapi jangan mengganti pemetaan yang ada. Contoh berikut menambahkan pemetaan antara prinsipal IAM dengan izin ditambahkan pada langkah pertama dan grup yang dibuat pada langkah sebelumnya: Kubernetes
   - *my-console-viewer-role*Peran dan`eks-console-dashboard-full-access-group`.
   - *my-user*Pengguna dan`eks-console-dashboard-restricted-access-group`.

   Contoh ini mengasumsikan bahwa Anda melampirkan izin IAM pada langkah pertama ke peran bernama *my-console-viewer-role* dan nama pengguna. *my-user* Ganti *111122223333* dengan ID AWS akun Anda.

```
apiVersion: v1
data:
mapRoles: |
  - groups:
    - eks-console-dashboard-full-access-group
    rolearn: arn:aws:iam::111122223333:role/my-console-viewer-role
    username: my-console-viewer-role
mapUsers: |
  - groups:
    - eks-console-dashboard-restricted-access-group
    userarn: arn:aws:iam::111122223333:user/my-user
    username: my-user
```

   > **Important**
   >
   > Peran ARN tidak dapat menyertakan jalur seperti. `role/my-team/developers/my-console-viewer-role` Format ARN harus. `arn:aws:iam::111122223333:role/my-console-viewer-role` Dalam contoh ini, `my-team/developers/` perlu dihapus.

3. Simpan file, dan tutup editor teks Anda.

Observabilitas di Amazon EKS

Anda dapat mengamati data Anda di Amazon EKS menggunakan banyak alat pemantauan atau pencatatan yang tersedia. Data log Amazon EKS Anda dapat dialirkan ke Layanan AWS atau ke alat mitra untuk analisis data. Ada banyak layanan yang tersedia di AWS Management Console yang menyediakan data untuk memecahkan masalah Amazon EKS Anda.

Setelah memilih Cluster di panel navigasi kiri konsol Amazon EKS, Anda dapat melihat kesehatan dan detail klaster dengan memilih nama cluster Anda. Untuk melihat detail tentang Kubernetes sumber daya yang ada yang digunakan ke klaster Anda, lihatLihat Kubernetes sumber daya.

Pemantauan adalah bagian penting dalam menjaga keandalan, ketersediaan, dan kinerja Amazon EKS dan AWS solusi Anda. Kami menyarankan Anda mengumpulkan data pemantauan dari semua bagian AWS solusi Anda. Dengan begitu, Anda dapat lebih mudah men-debug kegagalan multi-titik jika terjadi. Sebelum Anda mulai memantau Amazon EKS, pastikan bahwa rencana pemantauan Anda menjawab pertanyaan-pertanyaan berikut.

- Apa tujuan Anda? Apakah Anda memerlukan pemberitahuan waktu nyata jika skala cluster Anda dramatis?
- Sumber daya apa yang perlu diamati?
- Seberapa sering Anda perlu mengamati sumber daya ini? Apakah perusahaan Anda ingin merespons risiko dengan cepat?
- Alat apa yang ingin Anda gunakan? Jika Anda sudah menjalankan AWS Fargate sebagai bagian dari peluncuran Anda, maka Anda dapat menggunakan router log bawaan.
- Siapa yang ingin Anda lakukan untuk melakukan tugas pemantauan?
- Kepada siapa Anda ingin pemberitahuan dikirim ketika terjadi kesalahan?

Pencatatan dan pemantauan di Amazon EKS

Amazon EKS menyediakan alat bawaan untuk pencatatan dan pemantauan. Pencatatan pesawat kontrol merekam semua panggilan API ke kluster Anda, mengaudit informasi yang menangkap pengguna yang melakukan tindakan apa pada kluster Anda, dan informasi berbasis peran. Untuk informasi selengkapnya, lihat Pencatatan dan pemantauan di Amazon EKS di Panduan AWSPreskriptif.

Pencatatan pesawat kontrol Amazon EKS menyediakan log audit dan diagnostik langsung dari bidang kontrol Amazon EKS ke CloudWatch Log di akun Anda. Log ini memudahkan Anda untuk mengamankan dan menjalankan klaster Anda. Anda dapat memilih jenis log yang tepat yang Anda butuhkan, dan log dikirim sebagai aliran log ke grup untuk setiap kluster Amazon EKS. CloudWatch Untuk informasi selengkapnya, lihat Pencatatan bidang kendali Amazon EKS.

> **Note**
>
> Saat Anda memeriksa log autentikator Amazon EKS di Amazon CloudWatch, entri ditampilkan yang berisi teks yang mirip dengan teks contoh berikut.
>
> ```
> level=info msg="mapping IAM role" groups="[]"
>  role="arn:aws:iam::111122223333:role/XXXXXXXXXXXXXXXXXX-
> NodeManagerRole-XXXXXXXX" username="eks:node-manager"
> ```
>
> Entri yang berisi teks ini diharapkan. `username` adalah peran layanan internal Amazon EKS yang melakukan operasi spesifik untuk grup simpul terkelola dan Fargate.
> Untuk logging tingkat rendah dan dapat disesuaikan, maka Kuberneteslogging tersedia.

Amazon EKS terintegrasi denganAWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di Amazon EKS. CloudTrail menangkap semua panggilan API untuk Amazon EKS sebagai acara. Panggilan yang tertangkap meliputi panggilan dari konsol Amazon EKS dan panggilan kode ke operasi API Amazon EKS. Untuk informasi selengkapnya, lihat Mencatat panggilan API Amazon EKS dengan AWS CloudTrail.

Server Kubernetes API mengekspos sejumlah metrik yang berguna untuk pemantauan dan analisis. Untuk informasi selengkapnya, lihat Metrik-metrik Prometheus.

Fluent BitUntuk mengonfigurasi Amazon CloudWatch log kustom, lihat Menyiapkan Fluent Bit di Panduan CloudWatch Pengguna Amazon.

Alat pencatatan dan pemantauan Amazon EKS

Amazon Web Services menyediakan berbagai alat yang dapat Anda gunakan untuk memantau Amazon EKS. Anda dapat mengonfigurasi beberapa alat untuk mengatur pemantauan otomatis, tetapi beberapa memerlukan panggilan manual. Kami menyarankan Anda mengotomatiskan tugas pemantauan sebanyak yang diizinkan oleh lingkungan dan toolset yang ada.

Alat Pencatatan

Area	Alat	Deskripsi	Pengaturan
Aplikasi	CloudWatch Wawasan Kontainer Amazon	Ini mengumpulkan, mengumpulkan, dan merangkum metrik dan log dari aplikasi dan layanan mikro Anda yang terkontainer.	Prosedur pengaturan
Bidang kontrol	AWS CloudTrail	Ini mencatat panggilan API oleh pengguna, peran, atau layanan.	Prosedur pengaturan
Beberapa area untuk AWS Fargate contoh	AWS Fargaterouter log	MisalnyaAWS Fargate, ini mengalirkan log ke AWS layanan atau alat mitra. Penggunaan AWSuntuk Fluent Bit. Log dapat dialirkan ke alat lain Layanan AWS atau mitra.	Prosedur pengaturan

Alat Pemantauan

Area	Alat	Deskripsi	Pengaturan
Aplikasi	CloudWatch Wawasan Kontainer	CloudWatch Container Insights mengumpulkan, mengumpulkan, dan merangkum metrik dan log dari aplikasi dan layanan mikro dalam kontainer Anda.	Prosedur pengaturan
Aplikasi	AWSDistro untuk OpenTelemetry (ADOT)	Ini mengumpulkan dan mengirimkan metrik yang berkorelasi, melacak data, dan metadata ke AWS layanan pemantauan atau mitra. Itu dapat diatur melalui CloudWatch Wawasan Kontainer.	Prosedur pengaturan
Aplikasi	DevOpsGuru Amazon	Ini mendeteksi kinerja dan ketersediaan operasional tingkat simpul.	Prosedur pengaturan

Area	Alat	Deskripsi	Pengaturan
Aplikasi	AWS X-Ray	Ini menerima data jejak tentang aplikasi Anda. Data jejak ini mencakup permintaan masuk dan keluar serta metadata tentang permintaan. Untuk Amazon EKS, implementasinya memerlukan OpenTelemetry add-on.	Prosedur pengaturan
Aplikasi	Operator CloudWatch Observabilitas Amazon	Operator CloudWatch Observabilitas Amazon mengumpulkan metrik, log, dan data penelusuran. Ini mengirim mereka ke Amazon CloudWatch danAWS X-Ray.	Prosedur pengaturan

Area	Alat	Deskripsi	Pengaturan
Bidang kontrol	Prometheus	CloudWatch Tingkat konsumsi log, penyimpanan arsip, dan pemindaian data berlaku untuk log bidang kontrol yang diaktifkan.	Prosedur pengaturan

Metrik-metrik Prometheus

Prometheusadalah database pemantauan dan deret waktu yang menggores titik akhir. Ini memberikan kemampuan untuk query, agregat, dan menyimpan data yang dikumpulkan. Anda juga dapat menggunakannya untuk peringatan dan agregasi peringatan. Topik ini menjelaskan cara mengatur Prometheus sebagai opsi terkelola atau open source. Memantau metrik bidang kontrol Amazon EKS adalah kasus penggunaan yang umum.

Layanan Terkelola Amazon untuk Prometheus adalah layanan pemantauan dan peringatan Prometheus yang kompatibel yang memudahkan pemantauan aplikasi dan infrastruktur dalam wadah dalam skala besar. Ini adalah layanan yang dikelola sepenuhnya yang secara otomatis menskalakan konsumsi, penyimpanan, kueri, dan peringatan metrik Anda. Ini juga terintegrasi dengan layanan AWS keamanan untuk memungkinkan akses cepat dan aman ke data Anda. Anda dapat menggunakan bahasa kueri PromQL sumber terbuka untuk menanyakan metrik dan memperingatkannya.

Untuk informasi selengkapnya tentang cara menggunakan Prometheus metrik setelah Anda mengaktifkannya, lihat Panduan Pengguna Layanan Terkelola Amazon untuk Prometheus.

Aktifkan Prometheus metrik saat membuat klaster

> **Important**
>
> Layanan Terkelola Amazon untuk sumber daya Prometheus berada di luar siklus hidup klaster dan perlu dipertahankan secara independen dari klaster. Saat Anda menghapus

klaster Anda, pastikan juga untuk menghapus pencakar yang berlaku untuk menghentikan biaya yang berlaku. Untuk informasi selengkapnya, lihat Menemukan dan menghapus pencakar di Amazon Managed Service for Prometheus User Guide.

Saat membuat cluster baru, Anda dapat mengaktifkan opsi untuk mengirim metrik. Prometheus Dalam AWS Management Console, opsi ini ada di langkah Konfigurasi observabilitas untuk membuat cluster baru. Untuk informasi selengkapnya, lihat Membuat klaster Amazon EKS.

Prometheusmenemukan dan mengumpulkan metrik dari cluster Anda melalui model berbasis tarik yang disebut scraping. Scraper disiapkan untuk mengumpulkan data dari infrastruktur cluster dan aplikasi kontainer Anda.

Saat Anda mengaktifkan opsi untuk mengirim Prometheus metrik, Layanan Terkelola Amazon untuk Prometheus menyediakan scraper tanpa agen yang dikelola sepenuhnya. Gunakan opsi konfigurasi lanjutan berikut untuk menyesuaikan scraper default sesuai kebutuhan.

Scraper alias

(Opsional) Masukkan alias unik untuk scraper.

Tujuan

Pilih Layanan Terkelola Amazon untuk ruang kerja Prometheus. Ruang kerja adalah ruang logis yang didedikasikan untuk penyimpanan dan kueri metrik. Prometheus Dengan ruang kerja ini, Anda akan dapat melihat Prometheus metrik di seluruh akun yang memiliki akses ke sana. Opsi Buat ruang kerja baru memberi tahu Amazon EKS untuk membuat ruang kerja atas nama Anda menggunakan alias Workspace yang Anda berikan. Dengan opsi Pilih ruang kerja yang ada, Anda dapat memilih ruang kerja yang ada dari daftar tarik-turun. Untuk informasi selengkapnya tentang ruang kerja, lihat Mengelola ruang kerja di Amazon Managed Service for Prometheus User Guide.

Akses layanan

Bagian ini merangkum izin yang Anda berikan saat mengirim Prometheus metrik:

- Izinkan Layanan Terkelola Amazon untuk Prometheus mendeskripsikan kluster Amazon EKS yang tergores
- Izinkan penulisan jarak jauh ke ruang Prometheus kerja yang Dikelola Amazon

Jika `AmazonManagedScraperRole` sudah ada, scraper menggunakannya.
Pilih `AmazonManagedScraperRole` tautan untuk melihat detail Izin. Jika

`AmazonManagedScraperRole` belum ada, pilih tautan Lihat detail izin untuk melihat izin tertentu yang Anda berikan dengan mengirimkan Prometheus metrik.

Subnet

Lihat subnet yang akan diwarisi scraper. Jika Anda perlu mengubahnya, kembali ke buat cluster Tentukan langkah jaringan.

Grup keamanan

Lihat grup keamanan yang akan diwarisi oleh scraper. Jika Anda perlu mengubahnya, kembali ke buat cluster Tentukan langkah jaringan.

Konfigurasi scraper

Ubah konfigurasi scraper dalam format YAMAL sesuai kebutuhan. Untuk melakukannya, gunakan formulir atau unggah file YAMM pengganti. Untuk informasi selengkapnya, lihat Konfigurasi pengikis di Amazon Managed Service for Prometheus User Guide.

Amazon Managed Service untuk Prometheus mengacu pada scraper tanpa agen yang dibuat bersama cluster sebagai kolektor terkelola. AWS Untuk informasi selengkapnya tentang kolektor AWS terkelola, lihat kolektor AWS terkelola di Panduan Pengguna Layanan Terkelola Amazon untuk Prometheus.

> **Important**
>
> Anda harus mengatur `aws-auth ConfigMap` untuk memberikan izin dalam cluster scraper. Untuk informasi selengkapnya, lihat Mengonfigurasi klaster Amazon EKS Anda di Panduan Pengguna Layanan Terkelola Amazon untuk Prometheus.

Melihat Prometheus detail scraper

Setelah membuat cluster dengan opsi Prometheus metrik diaktifkan, Anda dapat melihat detail Prometheus scraper Anda. Saat melihat cluster Anda di AWS Management Console, pilih tab Observability. Tabel menunjukkan daftar pencakar untuk cluster, termasuk informasi seperti ID scraper, alias, status, dan tanggal pembuatan.

Untuk melihat detail lebih lanjut tentang scraper, pilih tautan ID scraper. Misalnya, Anda dapat melihat konfigurasi scraper, Nama Sumber Daya Amazon (ARN), URL penulisan jarak jauh, dan informasi jaringan. Anda dapat menggunakan ID scraper sebagai input ke Amazon Managed Service

untuk operasi Prometheus API seperti dan. `DescribeScraper` `DeleteScraper` Anda juga dapat menggunakan API untuk membuat lebih banyak pencakar.

Untuk informasi selengkapnya tentang penggunaan Prometheus API, lihat Referensi API Amazon Managed Service for Prometheus API.

Menyebarkan menggunakan PrometheusHelm

Atau, Anda dapat menyebarkan Prometheus ke cluster Anda dengan Helm V3. Jika Anda sudah Helm menginstal, Anda dapat memeriksa versi Anda dengan `helm version` perintah. Helmadalah manajer paket untuk Kubernetes cluster. Untuk informasi lebih lanjut tentang Helm dan cara menginstalnya, lihatMenggunakan Helm dengan Amazon EKS.

Setelah mengonfigurasi Helm klaster Amazon EKS, Anda dapat menggunakannya untuk menerapkan Prometheus dengan langkah-langkah berikut.

Untuk menyebarkan menggunakan PrometheusHelm

1. Buat Prometheus namespace.

```
kubectl create namespace prometheus
```

2. Tambahkan bagan repositori `prometheus-community`.

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-
charts
```

3. MenyebarkanPrometheus.

```
helm upgrade -i prometheus prometheus-community/prometheus \
    --namespace prometheus \
    --set
 alertmanager.persistentVolume.storageClass="gp2",server.persistentVolume.storageClass="gp2
```

> **Note**
>
> Jika terdapat kesalahan `Error: failed to download "stable/prometheus" (hint: running `helm repo update` may help)` saat menjalankan perintah ini, jalankan `helm repo update prometheus-community`, dan kemudian coba jalankan perintah Langkah 2 lagi.

Jika terdapat kesalahan `Error: rendered manifests contain a resource that already exists`, jalankan `helm uninstall` *`your-release-name`* `-n` *`namespace`*, lalu coba jalankan kembali perintah Langkah 3.

4. Verifikasi bahwa semua yang Pods ada di `prometheus` namespace berada dalam status. `READY`

```
kubectl get pods -n prometheus
```

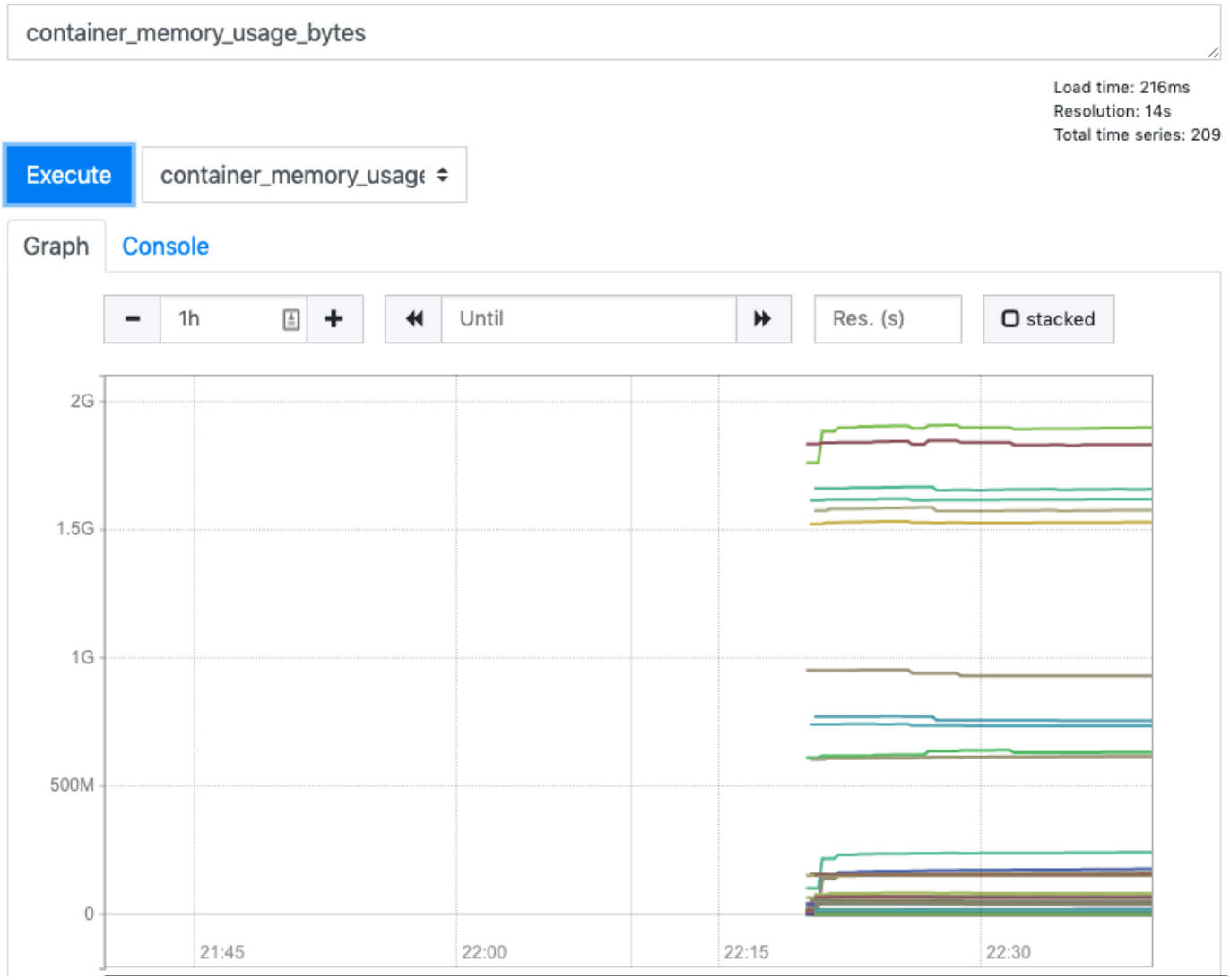
Contoh output adalah sebagai berikut.

```
NAME                                             READY   STATUS    RESTARTS   AGE
prometheus-alertmanager-59b4c8c744-r7bgp         1/2     Running   0          48s
prometheus-kube-state-metrics-7cfd87cf99-jkz2f   1/1     Running   0          48s
prometheus-node-exporter-jcjqz                   1/1     Running   0          48s
prometheus-node-exporter-jxv2h                   1/1     Running   0          48s
prometheus-node-exporter-vbdks                   1/1     Running   0          48s
prometheus-pushgateway-76c444b68c-82tnw          1/1     Running   0          48s
prometheus-server-775957f748-mmht9               1/2     Running   0          48s
```

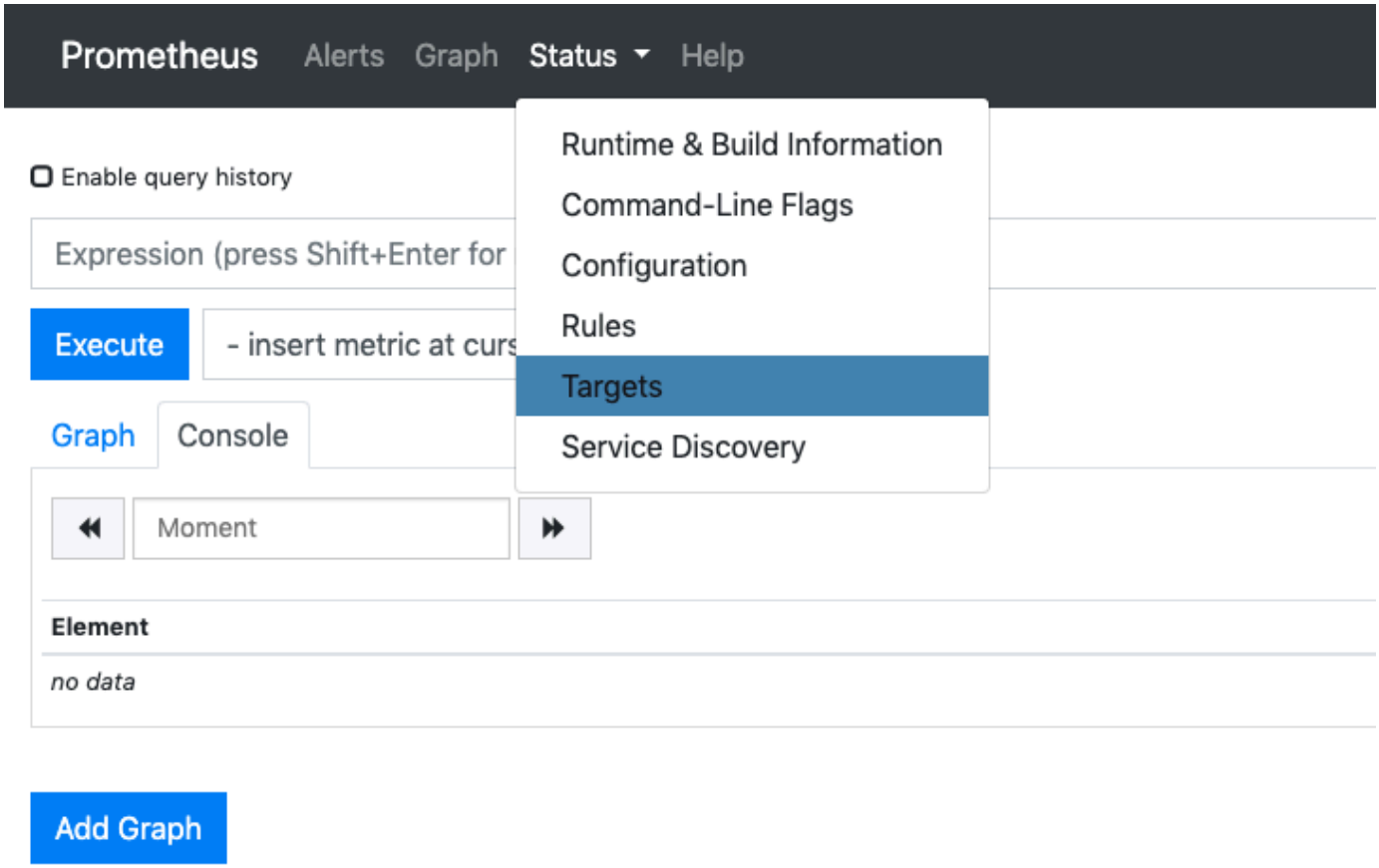
5. Gunakan `kubectl` untuk mem-port meneruskan Prometheus konsol ke mesin lokal Anda.

```
kubectl --namespace=prometheus port-forward deploy/prometheus-server 9090
```

6. Arahkan browser web `http://localhost:9090` untuk melihat Prometheus konsol.
7. Pilih metrik dari menu - masukkan metrik pada kursor, lalu pilih Eksekusi. Pilih tabel Grafik untuk menampilkan metrik dari waktu ke waktu. Citra berikut menunjukkan `container_memory_usage_bytes` dari waktu ke waktu.



8. Dari bilah atas navigasi, pilih Status, kemudian Target.



Semua Kubernetes titik akhir yang terhubung Prometheus menggunakan penemuan layanan ditampilkan.

Melihat metrik mentah bidang kontrol

Sebagai alternatif untuk penerapanPrometheus, server Kubernetes API mengekspos sejumlah metrik yang direpresentasikan dalam format. Prometheus Metrik ini berguna untuk pemantauan dan analisis. Mereka diekspos secara internal melalui titik akhir metrik yang mengacu pada HTTP API. `/metrics` Sama seperti titik akhir lainnya, titik akhir ini terpapar pada bidang kontrol Amazon EKS. Titik akhir ini terutama berguna untuk melihat metrik tertentu. Untuk menganalisis metrik dari waktu ke waktu, kami sarankan untuk menerapkanPrometheus.

Untuk melihat output metrik mentah, gunakan `kubectl` dengan bendera `--raw`. Perintah ini memungkinkan Anda untuk melewati jalur HTTP dan kembali ke respons baku.

```
kubectl get --raw /metrics
```

Contoh output adalah sebagai berikut.

```
[...]
# HELP rest_client_requests_total Number of HTTP requests, partitioned by status code,
 method, and host.
# TYPE rest_client_requests_total counter
rest_client_requests_total{code="200",host="127.0.0.1:21362",method="POST"} 4994
rest_client_requests_total{code="200",host="127.0.0.1:443",method="DELETE"} 1
rest_client_requests_total{code="200",host="127.0.0.1:443",method="GET"} 1.326086e+06
rest_client_requests_total{code="200",host="127.0.0.1:443",method="PUT"} 862173
rest_client_requests_total{code="404",host="127.0.0.1:443",method="GET"} 2
rest_client_requests_total{code="409",host="127.0.0.1:443",method="POST"} 3
rest_client_requests_total{code="409",host="127.0.0.1:443",method="PUT"} 8
# HELP ssh_tunnel_open_count Counter of ssh tunnel total open attempts
# TYPE ssh_tunnel_open_count counter
ssh_tunnel_open_count 0
# HELP ssh_tunnel_open_fail_count Counter of ssh tunnel failed open attempts
# TYPE ssh_tunnel_open_fail_count counter
ssh_tunnel_open_fail_count 0
```

Output mentah ini mengembalikan kata per kata yang ditampilkan server API. Metrik yang berbeda dicantumkan berdasarkan baris, dengan setiap baris termasuk nama metrik, tag, dan nilai.

```
metric_name{"tag"="value"[,...]}
            value
```

Dukungan add-on Amazon EKS untuk Amazon CloudWatch

Amazon CloudWatch Observabilitymengumpulkan log, metrik, dan data jejak waktu nyata. Ini mengirim mereka ke Amazon CloudWatch dan AWS X-Ray. Anda dapat menginstal add-on ini untuk mengaktifkan Sinyal CloudWatch Aplikasi dan CloudWatch Container Insights dengan observabilitas yang ditingkatkan untuk Amazon EKS. Ini membantu Anda memantau kesehatan dan kinerja infrastruktur dan aplikasi kontainer Anda. Amazon CloudWatch Observability OperatorIni dirancang untuk menginstal dan mengkonfigurasi komponen yang diperlukan.

Amazon EKS mendukung Amazon CloudWatch Observability Operator sebagai add-on Amazon EKS. Add-on memungkinkan Container Insights pada node keduanya Linux dan Windows pekerja di cluster. Untuk mengaktifkan Container InsightsWindows, versi add-on Amazon EKS harus `1.5.0` atau lebih tinggi. Saat ini, Sinyal CloudWatch Aplikasi tidak didukung di Amazon EKSWindows.

Topik di bawah ini menjelaskan cara memulai penggunaan Amazon CloudWatch Observability Operator untuk klaster Amazon EKS Anda.

- Untuk petunjuk cara menginstal add-on ini, lihat Menginstal CloudWatch agen menggunakan add-on CloudWatch Observability Amazon EKS di Panduan Pengguna Amazon CloudWatch .
- Untuk informasi selengkapnya tentang Sinyal CloudWatch Aplikasi, lihat Sinyal Aplikasi.
- Untuk informasi selengkapnyaContainer Insights, lihat Menggunakan Container Insights di Panduan CloudWatch Pengguna Amazon.

Pencatatan bidang kendali Amazon EKS

Pencatatan pesawat kontrol Amazon EKS menyediakan log audit dan diagnostik langsung dari bidang kontrol Amazon EKS ke CloudWatch Log di akun Anda. Log ini memudahkan Anda untuk mengamankan dan menjalankan klaster Anda. Anda dapat memilih jenis log yang tepat yang Anda butuhkan, dan log dikirim sebagai aliran log ke grup untuk setiap kluster Amazon EKS. CloudWatch Untuk informasi selengkapnya, lihat CloudWatchpencatatan Amazon.

Anda dapat mulai menggunakan pencatatan bidang kendali Amazon EKS dengan memilih jenis log mana yang ingin Anda aktifkan untuk setiap klaster Amazon EKS yang baru atau yang sudah ada. Anda dapat mengaktifkan atau menonaktifkan setiap jenis log pada basis per-cluster menggunakanAWS Management Console, AWS CLI (versi `1.16.139` atau lebih tinggi), atau melalui Amazon EKS API. Saat diaktifkan, log secara otomatis dikirim dari kluster Amazon EKS ke CloudWatch Log di akun yang sama.

Saat Anda menggunakan pencatatan bidang kendali Amazon EKS, Anda dikenai harga standar Amazon EKS untuk setiap klaster yang Anda jalankan. Anda akan dikenakan biaya konsumsi data CloudWatch Log standar dan biaya penyimpanan untuk setiap log yang dikirim ke CloudWatch Log dari cluster Anda. Anda juga dikenakan biaya untuk setiap sumber daya AWS, seperti instans Amazon EC2 atau volume Amazon EBS, yang Anda sediakan sebagai bagian dari klaster Anda.

Jenis log bidang kendali klaster berikut tersedia. Setiap jenis log sesuai dengan komponen bidang Kubernetes kontrol. Untuk mempelajari lebih lanjut tentang komponen ini, lihat KubernetesKomponen dalam Kubernetes dokumentasi.

Server API (**`api`**)

> Server API cluster Anda adalah komponen bidang kontrol yang mengekspos Kubernetes API. Jika Anda mengaktifkan log server API saat meluncurkan klaster, atau segera setelahnya, log

tersebut menyertakan flag server API yang digunakan untuk memulai server API. Untuk informasi selengkapnya, lihat `kube-apiserver`dan kebijakan audit dalam Kubernetes dokumentasi.

Audit (**`audit`**)

Kuberneteslog audit menyediakan catatan pengguna individu, administrator, atau komponen sistem yang telah memengaruhi klaster Anda. Untuk informasi selengkapnya, lihat Audit dalam Kubernetes dokumentasi.

Authenticator () **`authenticator`**

Log Authenticator unik untuk Amazon EKS. Log ini mewakili komponen bidang kontrol yang digunakan Amazon EKS untuk otentikasi Kubernetes Role Based Access Control (RBAC) menggunakan kredenal IAM. Untuk informasi selengkapnya, lihat Manajemen klaster.

Manajer pengontrol (**`controllerManager`**)

Manajer pengontrol mengelola loop kontrol inti yang dikirimkan bersamaKubernetes. Untuk informasi lebih lanjut, lihat kube-controller-managerdi Kubernetes dokumentasi.

Penjadwal () **`scheduler`**

Komponen scheduler mengelola kapan dan di mana harus dijalankan Pods di cluster Anda. Untuk informasi selengkapnya, lihat kube-scheduler di dokumentasi. Kubernetes

Mengaktifkan dan menonaktifkan log bidang kendali

Secara default, log bidang kontrol cluster tidak dikirim ke CloudWatch Log. Anda harus mengaktifkan setiap jenis log satu per satu untuk mengirim log untuk klaster Anda. CloudWatch Tingkat konsumsi log, penyimpanan arsip, dan pemindaian data berlaku untuk log bidang kontrol yang diaktifkan. Untuk informasi lebih lanjut, lihat CloudWatch harga.

Untuk memperbarui konfigurasi pencatatan bidang kontrol, Amazon EKS memerlukan hingga lima alamat IP yang tersedia di setiap subnet. Saat Anda mengaktifkan jenis log, maka log tersebut akan dikirim dengan tingkat verbositas log sebesar 2.

AWS Management Console

Untuk mengaktifkan atau menonaktifkan log bidang kendali dengan AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih nama klaster untuk menampilkan informasi klaster Anda.

3. Pilih tab Observability.
4. Di bagian Control plane logging, pilih Manage logging.
5. Untuk setiap jenis log individu, pilih apakah jenis log harus dihidupkan atau dimatikan. Secara default, setiap jenis log dimatikan.
6. Pilih Simpan perubahan untuk menyelesaikan.

AWS CLI

Untuk mengaktifkan atau menonaktifkan log bidang kendali dengan AWS CLI

1. Periksa versi AWS CLI Anda dengan perintah berikut.

```
aws --version
```

   Jika AWS CLI versi Anda lebih awal dari`1.16.139`, Anda harus terlebih dahulu memperbarui ke versi terbaru. Untuk menginstal atau memutakhirkan AWS CLI, lihat Menginstal AWS Command Line Interface dalam Panduan Pengguna AWS Command Line Interface.

2. Perbarui konfigurasi ekspor log bidang kendali milik klaster Anda dengan perintah AWS CLI. Ganti *`my-cluster`* dengan nama cluster Anda dan tentukan nilai akses titik akhir yang Anda inginkan.

> **Note**
>
> Perintah berikut mengirimkan semua jenis log yang tersedia ke CloudWatch Log.

```
aws eks update-cluster-config \
    --region region-code \
    --name my-cluster \
    --logging '{"clusterLogging":[{"types":
["api","audit","authenticator","controllerManager","scheduler"],"enabled":true}]}'
```

   Contoh output adalah sebagai berikut.

```
{
    "update": {
        "id": "883405c8-65c6-4758-8cee-2a7c1340a6d9",
```

```
        "status": "InProgress",
        "type": "LoggingUpdate",
        "params": [
            {
                "type": "ClusterLogging",
                "value": "{\"clusterLogging\":[{\"types\":[\"api\",\"audit\",
\"authenticator\",\"controllerManager\",\"scheduler\"],\"enabled\":true}]}"
            }
        ],
        "createdAt": 1553271814.684,
        "errors": []
    }
}
```

3. Pantau status pembaruan konfigurasi log Anda dengan perintah berikut, menggunakan nama klaster dan ID pembaruan yang dikembalikan oleh perintah sebelumnya. Pembaruan Anda selesai saat status muncul sebagai berikut `Successful`.

```
aws eks describe-update \
    --region region-code\
    --name my-cluster \
    --update-id 883405c8-65c6-4758-8cee-2a7c1340a6d9
```

Contoh output adalah sebagai berikut.

```
{
    "update": {
        "id": "883405c8-65c6-4758-8cee-2a7c1340a6d9",
        "status": "Successful",
        "type": "LoggingUpdate",
        "params": [
            {
                "type": "ClusterLogging",
                "value": "{\"clusterLogging\":[{\"types\":[\"api\",\"audit\",
\"authenticator\",\"controllerManager\",\"scheduler\"],\"enabled\":true}]}"
            }
        ],
        "createdAt": 1553271814.684,
        "errors": []
    }
}
```

Melihat log bidang kendali klaster

Setelah Anda mengaktifkan salah satu jenis log bidang kontrol untuk kluster Amazon EKS Anda, Anda dapat melihatnya di CloudWatch konsol.

Untuk mempelajari selengkapnya tentang melihat, menganalisis, dan mengelola log in CloudWatch, lihat Panduan Pengguna CloudWatch Log Amazon.

Untuk melihat log bidang kontrol klaster Anda di CloudWatch konsol

1. Buka konsol CloudWatch . Tautan dapat membuka konsol dan menampilkan grup log yang tersedia saat ini, serta memfilternya dengan prefiks `/aws/eks`.
2. Pilih klaster yang ingin Anda lihat log-nya. Format nama grup log adalah `/aws/eks/`*`my-cluster`*`/cluster`.
3. Pilih aliran log untuk melihat. Daftar berikut mendeskripsikan format nama aliran log untuk setiap jenis log.

   > **Note**
   >
   > Saat data aliran log bertambah, nama aliran log dirotasikan. Ketika beberapa aliran log ada untuk jenis log tertentu, Anda dapat melihat aliran log terbaru dengan mencari nama aliran log dengan waktu peristiwa Terakhir terbaru.

   - KubernetesLog komponen server API (**`api`**) - `kube-apiserver-`*`1234567890abcdef01234567890abcde`*
   - Audit (**`audit`**) – `kube-apiserver-audit-`*`1234567890abcdef01234567890abcde`*
   - Authenticator (**`authenticator`**) – `authenticator-`*`1234567890abcdef01234567890abcde`*
   - Manajer pengendali (**`controllerManager`**) – `kube-controller-manager-`*`1234567890abcdef01234567890abcde`*
   - Penjadwal (**`scheduler`**) – `kube-scheduler-`*`1234567890abcdef01234567890abcde`*
4. Lihat melalui peristiwa aliran log.

   Misalnya, Anda akan melihat flag server API awal untuk cluster saat melihat bagian atas. `kube-apiserver-`*`1234567890abcdef01234567890abcde`*

> **Note**
>
> Jika Anda tidak melihat log server API pada awal pengaliran log, maka kemungkinan file log server API telah dirotasikan di server sebelum Anda mengaktifkan pencacatan server API di server. File log apa pun yang diputar sebelum pencatatan server API diaktifkan tidak dapat diekspor ke. CloudWatch
> Namun, Anda dapat membuat klaster baru dengan Kubernetes versi yang sama dan mengaktifkan pencatatan server API saat membuat klaster. klaster yang memiliki versi platform dengan bendera yang sama yang telah diaktifkan, sehingga flag Anda harus sesuai dengan bendera klaster yang baru. Ketika Anda selesai melihat flag untuk cluster baru di CloudWatch, Anda dapat menghapus cluster baru.

Mencatat panggilan API Amazon EKS dengan AWS CloudTrail

Amazon EKS terintegrasi denganAWS CloudTrail. CloudTrail adalah layanan yang menyediakan catatan tindakan oleh pengguna, peran, atau AWS layanan di Amazon EKS. CloudTrail menangkap semua panggilan API untuk Amazon EKS sebagai peristiwa. Ini termasuk panggilan dari konsol Amazon EKS dan dari panggilan kode ke operasi Amazon EKS API.

Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman CloudTrail acara secara terus menerus ke bucket Amazon S3. Ini termasuk acara untuk Amazon EKS. Jika Anda tidak mengonfigurasi jejak, Anda masih dapat melihat peristiwa terbaru di CloudTrail konsol dalam Riwayat acara. Menggunakan informasi yang CloudTrail dikumpulkan, Anda dapat menentukan beberapa detail tentang permintaan. Misalnya, Anda dapat menentukan kapan permintaan dibuat ke Amazon EKS, alamat IP tempat permintaan dibuat, dan siapa yang mengajukan permintaan.

Untuk mempelajari selengkapnya CloudTrail, lihat Panduan AWS CloudTrail Pengguna.

Topik

- Informasi Amazon EKS di CloudTrail
- Memahami entri file log Amazon EKS
- Aktifkan pengumpulan metrik grup Auto Scaling

Informasi Amazon EKS di CloudTrail

Saat Anda membuatAWS akun, juga CloudTrail diaktifkan diAWS akun Anda. Ketika aktivitas terjadi di Amazon EKS, aktivitas tersebut dicatat dalam CloudTrail kejadian bersama kejadianAWS layanan lainnya di Riwayat kejadian. Anda dapat melihat, mencari, dan mengunduh peristiwa terbaru di akun AWS Anda. Untuk informasi lebih lanjut, lihat Menampilkan kejadian dengan riwayat CloudTrail kejadian.

Untuk catatan kejadian yang sedang berlangsung di akun AWS Anda, termasuk peristiwa untuk Amazon EKS, buatlah jejak. Jejak CloudTrail memungkinkan pengiriman berkas log ke bucket Amazon S3. Secara default, saat Anda membuat jejak di konsol, jejak tersebut berlaku untuk semua Wilayah AWS. Jejak mencatat kejadian dari semua Wilayah AWS di partisi AWS dan mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat mengonfigurasiAWS layanan lainnya untuk dianalisis lebih lanjut dan bertindak berdasarkan data kejadian yang dikumpulkan di CloudTrail log. Untuk informasi selengkapnya, lihat sumber daya berikut ini.

- Gambaran umum untuk membuat jejak
- CloudTrail Layanan yang didukung dan integrasi
- Mengkonfigurasi notifikasi Amazon SNS untuk CloudTrail
- Menerima Berkas CloudTrail Log dari Beberapa Wilayah dan Menerima Berkas CloudTrail Log dari Beberapa Akun

Semua tindakan Amazon EKS dicatat oleh CloudTrail dan didokumentasikan di Referensi Amazon EKS API. Misalnya, panggilan ke `CreateCluster`, `ListClusters`dan `DeleteCluster`bagian menghasilkan entri dalam file CloudTrail log.

Setiap peristiwa atau entri log berisi informasi tentang jenis identitas IAM yang membuat permintaan, dan kredensi mana yang digunakan. Jika kredensi sementara digunakan, entri menunjukkan bagaimana kredensi diperoleh.

Untuk informasi selengkapnya, lihat Elemen userIdentity CloudTrail .

Memahami entri file log Amazon EKS

Jejak adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai berkas log ke bucket Amazon S3 yang Anda tentukan. CloudTrail berkas log berisi satu atau beberapa entri log. Suatu peristiwa mewakili permintaan tunggal dari semua sumber dan mencakup informasi tentang tindakan yang diminta. Ini mencakup informasi seperti tanggal dan waktu tindakan dan parameter permintaan

yang digunakan. CloudTrail Berkas log bukan jejak tumpukan terurut dari panggilan API publik, sehingga berkas tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkan `CreateCluster`tindakan.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
    "arn": "arn:aws:iam::111122223333:user/username",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "username"
  },
  "eventTime": "2018-05-28T19:16:43Z",
  "eventSource": "eks.amazonaws.com",
  "eventName": "CreateCluster",
  "awsRegion": "region-code",
  "sourceIPAddress": "205.251.233.178",
  "userAgent": "PostmanRuntime/6.4.0",
  "requestParameters": {
    "resourcesVpcConfig": {
      "subnetIds": [
        "subnet-a670c2df",
        "subnet-4f8c5004"
      ]
    },
    "roleArn": "arn:aws:iam::111122223333:role/AWSServiceRoleForAmazonEKS-
CAC1G1VH3ZKZ",
    "clusterName": "test"
  },
  "responseElements": {
    "cluster": {
      "clusterName": "test",
      "status": "CREATING",
      "createdAt": 1527535003.208,
      "certificateAuthority": {},
      "arn": "arn:aws:eks:region-code:111122223333:cluster/test",
      "roleArn": "arn:aws:iam::111122223333:role/AWSServiceRoleForAmazonEKS-
CAC1G1VH3ZKZ",
      "version": "1.10",
      "resourcesVpcConfig": {
        "securityGroupIds": [],
```

```
        "vpcId": "vpc-21277358",
        "subnetIds": [
          "subnet-a670c2df",
          "subnet-4f8c5004"
        ]
      }
    }
  },
  "requestID": "a7a0735d-62ab-11e8-9f79-81ce5b2b7d37",
  "eventID": "eab22523-174a-499c-9dd6-91e7be3ff8e3",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Log entri untuk Peran Tertaut Layanan Amazon EKS

Peran tertaut layanan Amazon EKS membuat panggilan API keAWS sumber daya. CloudTrail entri log denganusername: AWSServiceRoleForAmazonEKS danusername: AWSServiceRoleForAmazonEKSNodegroup muncul untuk panggilan yang dibuat oleh peran tertaut layanan Amazon EKS. Untuk informasi selengkapnya tentang Amazon EKS dan peran tertaut layanan, lihat Menggunakan peran tertaut layanan untuk Amazon EKS.

Contoh berikut menunjukkan entri CloudTrail log yang menunjukkanDeleteInstanceProfile tindakan yang dibuat oleh peran terkaitAWSServiceRoleForAmazonEKSNodegroup layanan, dicatat dalamsessionContext.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3WHGPEZ7SJ2CW55C5:EKS",
        "arn": "arn:aws:sts::111122223333:assumed-role/
AWSServiceRoleForAmazonEKSNodegroup/EKS",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA3WHGPEZ7SJ2CW55C5",
                "arn": "arn:aws:iam::111122223333:role/aws-service-role/eks-
nodegroup.amazonaws.com/AWSServiceRoleForAmazonEKSNodegroup",
```

```
                "accountId": "111122223333",
                "userName": "AWSServiceRoleForAmazonEKSNodegroup"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2020-02-26T00:56:33Z"
            }
        },
        "invokedBy": "eks-nodegroup.amazonaws.com"
    },
    "eventTime": "2020-02-26T00:56:34Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "DeleteInstanceProfile",
    "awsRegion": "region-code",
    "sourceIPAddress": "eks-nodegroup.amazonaws.com",
    "userAgent": "eks-nodegroup.amazonaws.com",
    "requestParameters": {
        "instanceProfileName": "eks-11111111-2222-3333-4444-abcdef123456"
    },
    "responseElements": null,
    "requestID": "11111111-2222-3333-4444-abcdef123456",
    "eventID": "11111111-2222-3333-4444-abcdef123456",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Aktifkan pengumpulan metrik grup Auto Scaling

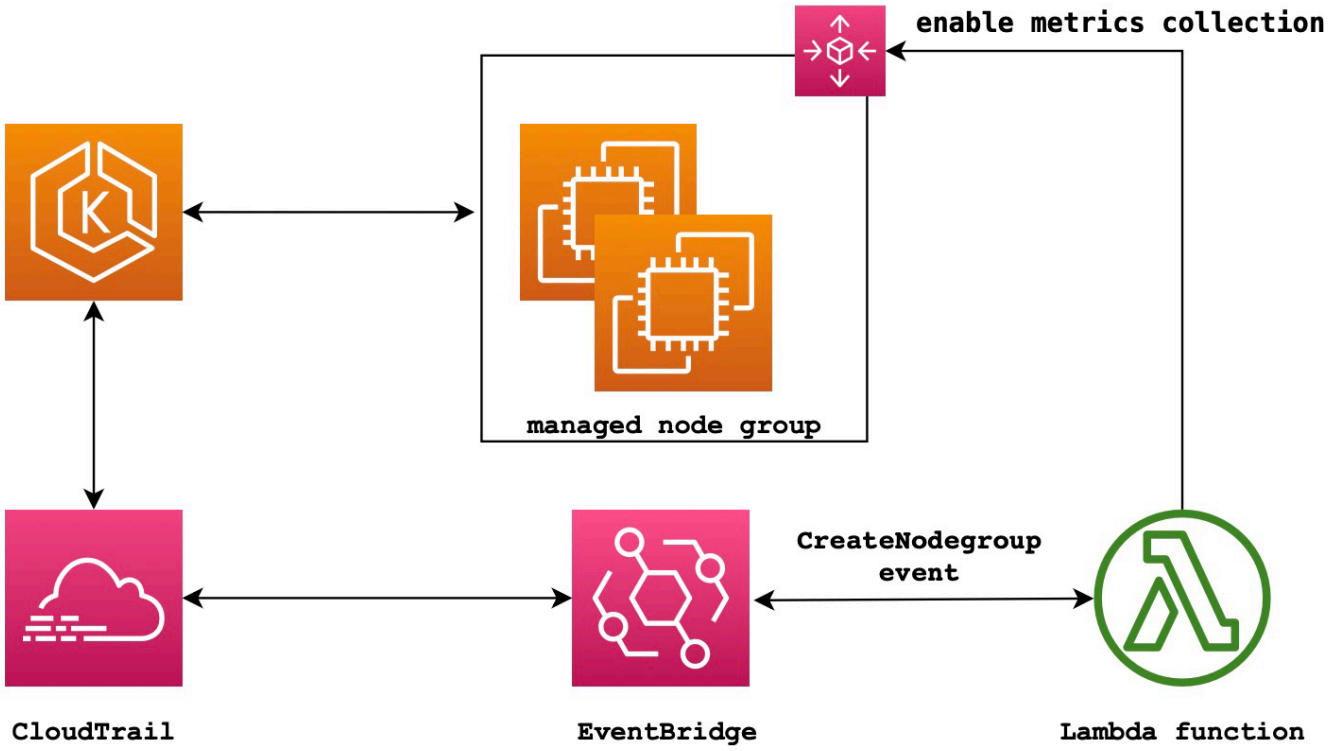
Topik ini menjelaskan cara mengaktifkan kumpulan metrik grup Auto Scaling menggunakan dan. AWS LambdaAWS CloudTrail Amazon EKS tidak secara otomatis mengaktifkan pengumpulan metrik grup untuk grup Auto Scaling yang dibuat untuk node terkelola.

Anda dapat menggunakan metrik grup Auto Scaling untuk melacak perubahan dalam grup Auto Scaling dan untuk menyetel alarm pada nilai ambang batas. Metrik grup Auto Scaling tersedia di konsol Auto Scaling atau konsol Amazon. CloudWatch Setelah diaktifkan, grup Auto Scaling mengirimkan data sampel ke Amazon CloudWatch setiap menit. Tidak ada biaya untuk mengaktifkan metrik ini.

Dengan mengaktifkan kumpulan metrik grup Auto Scaling, Anda akan dapat memantau penskalaan grup node terkelola. Metrik grup Auto Scaling melaporkan ukuran minimum, maksimum, dan yang diinginkan dari grup Auto Scaling. Anda dapat membuat alarm jika jumlah node dalam grup node

berada di bawah ukuran minimum, yang akan menunjukkan grup node yang tidak sehat. Melacak ukuran grup node juga berguna dalam menyesuaikan jumlah maksimum sehingga bidang data Anda tidak kehabisan kapasitas.

Saat Anda membuat grup node terkelola, AWS CloudTrail kirimkan `CreateNodegroup` acara ke Amazon EventBridge. Dengan membuat EventBridge aturan Amazon yang cocok dengan `CreateNodegroup` peristiwa tersebut, Anda memicu fungsi Lambda untuk mengaktifkan kumpulan metrik grup untuk grup Auto Scaling yang terkait dengan grup node terkelola.



Untuk mengaktifkan pengumpulan metrik grup Auto Scaling

1. Buat peran IAM untuk Lambda.

```
LAMBDA_ROLE=$(aws iam create-role \
  --role-name lambda-asg-enable-metrics \
  --assume-role-policy-document '{"Version": "2012-10-17","Statement":
 [{ "Effect": "Allow", "Principal": {"Service": "lambda.amazonaws.com"}, "Action":
 "sts:AssumeRole"}]}' \
  --output text \
  --query 'Role.Arn')
```

```
echo $LAMBDA_ROLE
```

2. Buat kebijakan yang memungkinkan mendeskripsikan grup node Amazon EKS dan mengaktifkan pengumpulan metrik grup Auto Scaling.

```
cat > /tmp/lambda-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
            "eks:DescribeNodegroup",
            "autoscaling:EnableMetricsCollection"
            ],
        "Resource": [
            "*"
        ]
      }
    ]
}
EOF
LAMBDA_POLICY_ARN=$(aws iam create-policy \
  --policy-name lambda-asg-enable-metrics-policy \
  --policy-document file:///tmp/lambda-policy.json \
  --output text \
  --query 'Policy.Arn')
echo $LAMBDA_POLICY_ARN
```

3. Lampirkan kebijakan ke peran IAM untuk Lambda.

```
aws iam attach-role-policy \
  --policy-arn $LAMBDA_POLICY_ARN \
  --role-name lambda-asg-enable-metrics
```

4. Tambahkan kebijakan `AWSLambdaBasicExecutionRole` terkelola, yang memiliki izin yang diperlukan fungsi untuk menulis log ke CloudWatch Log.

```
aws iam attach-role-policy \
  --role-name lambda-asg-enable-metrics \
  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole
```

5. Buat kode Lambda.

```
cat > /tmp/lambda-handler.py <<EOF
import json
import boto3
import time
import logging

eks = boto3.client('eks')
autoscaling = boto3.client('autoscaling')

logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    ASG_METRICS_COLLLECTION_TAG_NAME = "ASG_METRICS_COLLLECTION_ENABLED"
    initial_retry_delay = 10
    attempts = 0

    #print(event)

    if not event["detail"]["eventName"] == "CreateNodegroup":
        print("invalid event.")
        return -1

    clusterName = event["detail"]["requestParameters"]["name"]
    nodegroupName = event["detail"]["requestParameters"]["nodegroupName"]
    try:
        metricsCollectionEnabled = event["detail"]["requestParameters"]["tags"]
[ASG_METRICS_COLLLECTION_TAG_NAME]
    except KeyError:
        print(ASG_METRICS_COLLLECTION_TAG_NAME, "tag not found.")
        return

    # Check if metrics collection is enabled in tags
    if metricsCollectionEnabled.lower() != "true":
        print("Metrics collection is not enabled in nodegroup tags.")
        return

    # Get the name of the associated autoscaling group
    print("Getting the autoscaling group name for nodegroup=", nodegroupName, ",
 cluster=", clusterName )
    for i in range(0,10):
        try:
```

```
            autoScalingGroup =
 eks.describe_nodegroup(clusterName=clusterName,nodegroupName=nodegroupName)
["nodegroup"]["resources"]["autoScalingGroups"][0]["name"]
        except:
            attempts += 1
            print("Failed to obtain the associated autoscaling group for
 nodegroup", nodegroupName, "Retrying in", initial_retry_delay*attempts,
 "seconds.")
            time.sleep(initial_retry_delay*attempts)
        else:
            break

    print("Enabling metrics collection on autoscaling group ", autoScalingGroup)

    # Enable metrics collection in the autoscaling group
    try:
        enableMetricsCollection =
 autoscaling.enable_metrics_collection(AutoScalingGroupName=autoScalingGroup,Granularity="1
    except:
        print("Unable to enable metrics collection on nodegroup=",nodegroup)
    print("Enabled metrics collection on nodegroup", nodegroupName)
EOF
```

6. Buat paket deployment.

```
cd /tmp
zip function.zip lambda-handler.py
```

7. Buat fungsi Lambda.

```
LAMBDA_ARN=$(aws lambda create-function --function-name asg-enable-metrics-
collection \
  --zip-file fileb://function.zip --handler lambda-handler.lambda_handler \
  --runtime python3.9 \
  --timeout 600 \
  --role $LAMBDA_ROLE \
  --output text \
  --query 'FunctionArn')
echo $LAMBDA_ARN
```

8. Buat EventBridge aturan.

```
RULE_ARN=$(aws events put-rule --name CreateNodegroupRuleToLambda \
```

```
  --event-pattern "{\"source\":[\"aws.eks\"],\"detail-type\":[\"AWS API Call via
 CloudTrail\"],\"detail\":{\"eventName\":[\"CreateNodegroup\"],\"eventSource\":
[\"eks.amazonaws.com\"]}}" \
  --output text \
  --query 'RuleArn')
echo $RULE_ARN
```

9. Tambahkan fungsi Lambda sebagai target.

```
aws events put-targets --rule CreateNodegroupRuleToLambda \
 --targets "Id"="1","Arn"="$LAMBDA_ARN"
```

10. Tambahkan kebijakan yang memungkinkan EventBridge untuk menjalankan fungsi Lambda.

```
aws lambda add-permission \
  --function-name asg-enable-metrics-collection \
  --statement-id CreateNodegroupRuleToLambda \
  --action 'lambda:InvokeFunction' \
  --principal events.amazonaws.com \
  --source-arn $RULE_ARN
```

Fungsi Lambda memungkinkan pengumpulan metrik grup Auto Scaling untuk setiap grup node terkelola yang Anda beri tag dengan disetel. `ASG_METRICS_COLLLECTION_ENABLED TRUE` Untuk mengonfirmasi bahwa kumpulan metrik grup Auto Scaling diaktifkan, navigasikan ke grup Auto Scaling terkait di konsol Amazon EC2. Di tab Monitoring, Anda akan melihat bahwa kotak centang Aktifkan diaktifkan.

Dukungan add-on Amazon EKS untuk Operator ADOT

Amazon EKS mendukung penggunaanAWS Management Console, AWS CLI dan Amazon EKS API untuk menginstal dan mengelola AWSDistro for OpenTelemetry (ADOT) Operator. Ini memudahkan aplikasi Anda yang berjalan di Amazon EKS untuk mengirim metrik dan melacak data ke beberapa opsi layanan pemantauan seperti Amazon CloudWatch, Prometheus, dan X-Ray.

Untuk informasi selengkapnya, lihat Memulai dengan AWS Distro untuk OpenTelemetry menggunakan Eks Add-On di AWS Distro untuk dokumentasi. OpenTelemetry

Lebih banyak AWS layanan terintegrasi dengan Amazon EKS

Selain layanan yang tercakup dalam bagian lain, Amazon EKS bekerja dengan lebih banyak AWS layanan untuk memberikan solusi tambahan. Topik ini mengidentifikasi beberapa layanan lain yang menggunakan Amazon EKS untuk menambahkan fungsionalitas, atau layanan yang digunakan Amazon EKS untuk melakukan tugas.

Topik

Membuat sumber daya Amazon EKS dengan AWS CloudFormation

Amazon EKS terintegrasi dengan AWS CloudFormation, layanan yang membantu Anda memodelkan dan mengatur sumber daya AWS sehingga Anda dapat lebih cepat dalam membuat dan mengelola sumber daya dan infrastruktur Anda. Anda membuat templat yang menjelaskan semua sumber daya AWS yang Anda inginkan, misalnya klaster Amazon EKS dan AWS CloudFormation mengurus penyediaan dan konfigurasi sumber daya tersebut untuk Anda.

Saat Anda menggunakan AWS CloudFormation, Anda dapat menggunakan kembali templat Anda untuk mengatur sumber daya Amazon EKS Anda secara konsisten dan berulang kali. Cukup jelaskan sumber daya Anda sekali dan kemudian sediakan sumber daya yang sama berulang-ulang dalam beberapa akun dan Wilayah AWS.

Templat Amazon EKS dan AWS CloudFormation

Untuk menyediakan dan mengonfigurasi sumber daya untuk Amazon EKS dan layanan terkait, Anda harus memahami templat AWS CloudFormation. Templat adalah file teks yang diformat dalam

JSON atau YAML. Templat ini menjelaskan sumber daya yang ingin Anda sediakan di tumpukan AWS CloudFormation Anda. Jika Anda tidak terbiasa dengan JSON atau YAML, Anda dapat menggunakan AWS CloudFormation Designer untuk membantu Anda memulai dengan templat AWS CloudFormation. Untuk informasi selengkapnya, lihat Apa itu AWS CloudFormation Designer? dalam AWS CloudFormation Panduan Pengguna.

Amazon EKS mendukung menciptakan klaster dan grup simpul di AWS CloudFormation. Untuk informasi selengkapnya, termasuk contoh templat JSON dan YAML untuk sumber daya Amazon EKS Anda, lihat Amazon EKS referensi jenis sumber daya dalam AWS CloudFormation Panduan Pengguna.

Pelajari selengkapnya tentang AWS CloudFormation

Untuk mempelajari selengkapnya tentang AWS CloudFormation, lihat sumber daya berikut:

- AWS CloudFormation
- AWS CloudFormationPanduan Pengguna
- AWS CloudFormationPanduan Pengguna Baris Perintah

Amazon EKS danAWSZona Lokal

SesiAWSZona Lokal adalah perpanjangan dariWilayah AWSdi kedekatan geografis dengan pengguna Anda. Local Zones memiliki koneksi sendiri ke internet dan mendukung AWS Direct Connect. Sumber daya yang dibuat di Local Zone dapat melayani pengguna lokal dengan komunikasi latensi sangat rendah. Untuk informasi selengkapnya, lihat Local Zones.

Amazon EKS mendukung sumber daya tertentu di Local Zones. Perangkat lunak ini mencakup:simpul Amazon EC2 yang dikelola sendiri, volume Amazon EBS, dan Application Load Balancers (ALB). Kami menyarankan agar Anda mempertimbangkan hal berikut ketika menggunakan Local Zones sebagai bagian dari klaster Amazon EKS Anda.

Simpul

Anda tidak dapat membuat grup node terkelola atau node Fargate di Local Zones dengan Amazon EKS. Namun demikian, Anda dapat membuat simpul Amazon EC2 yang dikelola sendiri di Local Zones menggunakan API Amazon EC2,AWS CloudFormation, atau`eksctl`. Untuk informasi selengkapnya, lihat Simpul yang dikelola sendiri.

Arsitektur jaringan

- Amazon EKS dikelolaKuberneteskontrol pesawat selalu berjalan diWilayah AWS. Amazon EKS dikelolaKubernetescontrol plane tidak dapat berjalan di Local Zone. Karena Local Zones muncul sebagai subnet dalam VPC Anda,Kubernetesmelihat sumber daya Local Zone Anda sebagai bagian dari subnet tersebut.
- Amazon EKSKubernetesklaster berkomunikasi dengan instans Amazon EC2 yang Anda jalankan diWilayah AWSatau Zona Lokal menggunakan Amazon EKS dikelolaantarmuka jaringan elastis. Untuk mempelajari selengkapnya tentang arsitektur jaringan Amazon EKS, lihat Jaringan Amazon EKS.
- Tidak seperti subnet regional, Amazon EKS tidak dapat menempatkan antarmuka jaringan ke subnet Local Zone Anda. Ini berarti bahwa Anda tidak harus menentukan subnet Local Zone ketika Anda membuat klaster Anda.

Deep Learning Containers

AWSDeep Learning Containers adalah satu setDockergambar untuk pelatihan dan melayani model diTensorFlowdi Amazon EKS dan Amazon Elastic Container Service (Amazon ECS). Deep Learning Containers menyediakan lingkungan yang dioptimalkanTensorFlow,NVIDIA CUDA(untuk instans GPU), danIntelMKL(untuk instans CPU) pustaka dan tersedia di Amazon ECR.

Untuk mulai menggunakanAWSKontainer Pembelajaran Mendalam di Amazon EKS, lihatPengaturan Amazon EKSdi dalamAWSPanduan Pengembang Kontainer Pembelajaran Mendalam.

Kisi Amazon VPC

Amazon VPC Lattice adalah layanan jaringan aplikasi yang dikelola sepenuhnya yang dibangun langsung ke dalam infrastrukturAWS jaringan yang dapat Anda gunakan untuk menghubungkan, mengamankan, dan memantau layanan Anda di beberapa akun dan Virtual Private Clouds (VPC). Dengan Amazon EKS, Anda dapat memanfaatkan Amazon VPC Lattice melalui penggunaanAWS Gateway API Controller, implementasi API Kubernetes Gateway. Dengan menggunakan Amazon VPC Lattice, Anda dapat mengatur konektivitas lintas klaster denganKubernetes semantik standar dengan cara yang sederhana dan konsisten. Untuk mulai menggunakan Amazon VPC Lattice dengan Amazon EKS, lihat Panduan Pengguna Pengontrol APIAWS Gateway.

AWS Resilience Hub

AWS Resilience Hubmenilai ketahanan klaster Amazon EKS dengan menganalisis infrastrukturnya. AWS Resilience Hubmenggunakan konfigurasi kontrol aksesKubernetes berbasis peran (RBAC) untuk menilaiKubernetes beban kerja yang diterapkan ke klaster Anda. Untuk informasi selengkapnya, lihat MengaktifkanAWS Resilience Hub akses ke klaster Amazon EKS Anda di PanduanAWS Resilience Hub Pengguna.

Amazon GuardDuty

EKS Protection in Amazon GuardDuty menyediakan cakupan deteksi ancaman untuk membantu Anda melindungi kluster Amazon EKS di AWS lingkungan Anda. Perlindungan EKS mencakup Pemantauan Log Audit EKS dan Pemantauan Runtime EKS. Untuk informasi selengkapnya, lihat Perlindungan EKS Amazon GuardDuty di Panduan Amazon GuardDuty Pengguna. Anda dapat menginstal GuardDuty agen ke cluster Anda sebagai add-on Amazon EKS. Untuk informasi selengkapnya, lihat Pengaya Amazon EKS yang tersedia dari Amazon EKS.

Menggunakan Amazon Security Lake dengan Amazon EKS

Amazon Security Lake adalah layanan danau data keamanan yang dikelola sepenuhnya yang memungkinkan Anda memusatkan data keamanan dari berbagai sumber, termasuk Amazon EKS. Dengan mengintegrasikan Amazon EKS dengan Security Lake, Anda dapat memperoleh wawasan yang lebih dalam tentang aktivitas yang dilakukan pada Kubernetes sumber daya Anda dan meningkatkan postur keamanan klaster Amazon EKS Anda.

> **Note**
>
> Untuk informasi selengkapnya tentang penggunaan Security Lake dengan Amazon EKS dan menyiapkan sumber data, lihat dokumentasi Amazon Security Lake.

Manfaat menggunakan Security Lake dengan Amazon Amazon EKS

Data keamanan terpusat — Security Lake secara otomatis mengumpulkan dan memusatkan data keamanan dari kluster Amazon EKS Anda, bersama dengan data dari layanan lain, penyedia SaaS AWS , sumber lokal, dan sumber pihak ketiga. Ini memberikan pandangan komprehensif tentang postur keamanan Anda di seluruh organisasi Anda.

Format data standar — Security Lake mengubah data yang dikumpulkan ke dalam format Open Cybersecurity Schema Framework (OCSF), yang merupakan skema open-source standar. Normalisasi ini memungkinkan analisis dan integrasi yang lebih mudah dengan alat dan layanan keamanan lainnya.

Deteksi ancaman yang ditingkatkan — Dengan menganalisis data keamanan terpusat, termasuk log pesawat kontrol Amazon EKS, Anda dapat mendeteksi aktivitas yang berpotensi mencurigakan dalam kluster Amazon EKS Anda dengan lebih efektif. Ini membantu dalam mengidentifikasi dan menanggapi insiden keamanan dengan segera.

Manajemen data yang disederhanakan — Security Lake mengelola siklus hidup data keamanan Anda dengan pengaturan retensi dan replikasi yang dapat disesuaikan. Ini menyederhanakan tugas manajemen data dan memastikan bahwa Anda menyimpan data yang diperlukan untuk tujuan kepatuhan dan audit.

Mengaktifkan Danau Keamanan untuk Amazon EKS

Untuk mulai menggunakan Security Lake dengan Amazon EKS, ikuti langkah-langkah berikut:

1. Aktifkan pencatatan pesawat kontrol Amazon EKS untuk kluster EKS Anda. Lihat Mengaktifkan dan menonaktifkan log bidang kontrol untuk instruksi terperinci.
2. Tambahkan Amazon EKS Audit Log sebagai sumber di Security Lake. Security Lake kemudian akan mulai mengumpulkan informasi mendalam tentang aktivitas yang dilakukan pada sumber daya Kubernetes yang berjalan di kluster EKS Anda.
3. Konfigurasikan pengaturan retensi dan replikasi untuk data keamanan Anda di Security Lake berdasarkan kebutuhan Anda.
4. Gunakan data OCSF yang dinormalisasi yang disimpan di Security Lake untuk respons insiden, analitik keamanan, dan integrasi dengan AWS layanan lain atau alat pihak ketiga. Misalnya, Anda dapat Menghasilkan wawasan keamanan dari data Amazon Security Lake menggunakan Amazon OpenSearch Ingestion.

Menganalisis Log EKS di Danau Keamanan

Security Lake menormalkan peristiwa log EKS ke format OCSF, membuatnya lebih mudah untuk menganalisis dan menghubungkan data dengan peristiwa keamanan lainnya. Anda dapat menggunakan berbagai alat dan layanan, seperti Amazon Athena, Amazon, atau alat analisis

keamanan pihak ketiga QuickSight, untuk menanyakan dan memvisualisasikan data yang dinormalisasi.

Untuk informasi lebih lanjut tentang pemetaan OCSF untuk peristiwa log EKS, lihat referensi pemetaan di repositori OCSF. GitHub

Amazon Detective

Amazon Detective membantu Anda menganalisis, menyelidiki, dan mengidentifikasi akar penyebab temuan keamanan atau aktivitas mencurigakan dengan cepat. Detective secara otomatis mengumpulkan data log dari sumber daya Anda. AWS Kemudian menggunakan pembelajaran mesin, analisis statistik, dan teori grafik untuk menghasilkan visualisasi yang membantu Anda melakukan penyelidikan keamanan yang lebih cepat dan lebih efisien. Agregasi data Detective prebuilt, ringkasan, dan konteks membantu Anda menganalisis dan menentukan sifat dan tingkat kemungkinan masalah keamanan dengan cepat. Untuk informasi selengkapnya, lihat Panduan Pengguna Detektif Amazon.

Detective mengatur Kubernetes dan AWS data ke dalam temuan seperti:

- Detail klaster Amazon EKS, termasuk identitas IAM yang menciptakan cluster dan peran layanan cluster. Anda dapat menyelidiki aktivitas AWS dan Kubernetes API dari identitas IAM ini dengan Detective.
- Detail kontainer, seperti konteks gambar dan keamanan. Anda juga dapat meninjau detail untuk dihentikan. Pods
- KubernetesAktivitas API, termasuk tren keseluruhan dalam aktivitas API dan detail panggilan API tertentu. Misalnya, Anda dapat menampilkan jumlah panggilan Kubernetes API yang berhasil dan gagal yang dikeluarkan selama rentang waktu yang dipilih. Selain itu, bagian tentang panggilan API yang baru diamati mungkin berguna untuk mengidentifikasi aktivitas yang mencurigakan.

Log audit Amazon EKS adalah paket sumber data opsional yang dapat ditambahkan ke grafik perilaku Detektif Anda. Anda dapat melihat paket sumber opsional yang tersedia, dan statusnya di akun Anda. Untuk informasi selengkapnya, lihat log audit Amazon EKS untuk Detektif di Panduan Pengguna Detektif Amazon.

Gunakan Amazon Detective dengan Amazon EKS

Untuk meninjau temuan untuk cluster Amazon EKS

Sebelum Anda dapat meninjau temuan, Detektif harus diaktifkan setidaknya selama 48 jam dalam waktu Wilayah AWS yang sama dengan cluster Anda. Untuk informasi selengkapnya, lihat Menyiapkan Detektif Amazon di Panduan Pengguna Detektif Amazon.

1. Buka konsol Detective di https://console.aws.amazon.com/detective/.
2. Dari panel navigasi kiri, pilih Cari.
3. Pilih Pilih jenis dan kemudian pilih EKS cluster.
4. Masukkan nama cluster atau ARN lalu pilih Cari.
5. Di hasil penelusuran, pilih nama klaster yang ingin Anda lihat aktivitasnya. Untuk informasi selengkapnya tentang apa yang dapat Anda lihat, lihat Keseluruhan aktivitas Kubernetes API yang melibatkan klaster Amazon EKS di Panduan Pengguna Detektif Amazon.

Pemecahan masalah Amazon EKS

Bab ini mencakup beberapa kesalahan umum yang mungkin dapat Anda temui saat menggunakan Amazon EKS dan cara yang dilakukan untuk mengatasinya. Jika Anda perlu memecahkan masalah area Amazon EKS tertentu, lihat topik terpisah Menyelesaikan masalah IAMMemecahkan masalah di Amazon EKS Connector, dan Pemecahan Masalah untuk ADOT menggunakan topik Pengaya EKS.

Untuk informasi pemecahan masalah lainnya, lihat konten Pusat Pengetahuan tentang Amazon Elastic Kubernetes Service di. AWS re:Post

Kapasitas tidak mencukupi

Jika Anda menerima kesalahan berikut saat mencoba membuat klaster Amazon EKS, maka salah satu Availability Zone yang Anda tentukan tidak memiliki kapasitas yang cukup untuk mendukung klaster.

```
Cannot create cluster 'example-cluster' because region-1d, the targeted Availability Zone, does not currently have sufficient capacity to support the cluster. Retry and choose from these Availability Zones: region-1a, region-1b, region-1c
```

Mencoba lagi membuat klaster Anda dengan subnet di klaster VPC yang meng-host di Availability Zone dikembalikan oleh pesan kesalahan ini.

Ada Availability Zone dimana cluster tidak dapat berada. Bandingkan Availability Zones tempat subnet Anda berada dengan daftar Availability Zone di. Persyaratan dan pertimbangan subnet

Simpul gagal untuk bergabung dengan klaster

Ada beberapa alasan umum yang mencegah simpul bergabung dengan klaster:

- Jika node adalah node yang dikelola, Amazon EKS menambahkan entri ke `aws-auth ConfigMap` saat Anda membuat grup node. Jika entri dihapus atau dimodifikasi, maka Anda perlu menambahkannya kembali. Untuk informasi lebih lanjut, masukkan **`eksctl create iamidentitymapping --help`** di terminal Anda. Anda dapat melihat `aws-auth ConfigMap` entri Anda saat ini dengan mengganti *`my-cluster`* dalam perintah berikut dengan nama cluster Anda dan kemudian menjalankan perintah yang dimodifikasi:. **`eksctl get iamidentitymapping --cluster`** ***`my-cluster`*** ARN peran yang Anda tentukan tidak dapat

menyertakan jalur selain. / Misalnya, jika nama peran Andadevelopment/apps/my-role, Anda harus mengubahnya menjadi my-role saat menentukan ARN untuk peran tersebut. Pastikan Anda menentukan ARN peran IAM node (bukan ARN profil instance).

Jika node dikelola sendiri, dan Anda belum membuat entri akses untuk ARN peran IAM node, jalankan perintah yang sama yang terdaftar untuk node terkelola. Jika Anda telah membuat entri akses untuk ARN untuk peran IAM node Anda, maka itu mungkin tidak dikonfigurasi dengan benar dalam entri akses. Pastikan bahwa ARN peran IAM node (bukan ARN profil instance) ditentukan sebagai ARN utama dalam entri atau entri akses Anda. aws-auth ConfigMap Untuk informasi selengkapnya tentang entri akses, lihatKelola entri akses.

- AWS CloudFormation Template ClusterNamedi node Anda tidak sama persis dengan nama cluster yang Anda inginkan untuk bergabung dengan node Anda. Memasukkan nilai yang salah ke bidang ini menyebabkan konfigurasi yang salah pada file /var/lib/kubelet/kubeconfig simpul, dan simpul tidak akan bergabung dengan klaster.
- Simpul tidak ditandai sebagai dimiliki oleh klaster. Simpul Anda harus memiliki tanda berikut yang diterapkan untuk mereka, di mana *my-cluster* diganti dengan nama klaster Anda.

Kunci	Nilai
kubernetes.io/cluster/ *my-cluster*	owned

- Simpul mungkin tidak dapat mengakses klaster menggunakan alamat IP publik. Pastikan bahwa simpul yang di-deploy di subnet publik memiliki alamat IP publik. Jika tidak, Anda dapat mengaitkan alamat IP Elastis ke simpul setelah diluncurkan. Untuk informasi selengkapnya, lihat Mengaitkan alamat IP Elastis dengan instans berjalan atau antarmuka jaringan. Jika subnet publik tidak diatur secara otomatis untuk menetapkan alamat IP publik ke instans yang di-deploy untuk itu, maka kami merekomendasikan untuk mengaktifkan pengaturan tersebut. Untuk informasi selengkapnya, lihat Memodifikasi atribut IPv4 pengalamatan publik untuk subnet Anda. Jika simpul di-deploy ke subnet privat, maka subnet harus memiliki rute ke gateway NAT yang memiliki alamat IP publik yang ditugaskan untuk itu.
- AWS STS Titik akhir untuk tempat Wilayah AWS Anda menerapkan node tidak diaktifkan untuk akun Anda. Untuk mengaktifkan wilayah, lihat Mengaktifkan dan menonaktifkan AWS STS dalam file. Wilayah AWS
- Node tidak memiliki entri DNS pribadi, sehingga kubelet log berisi node "" not found kesalahan. Pastikan bahwa VPC tempat node dibuat memiliki nilai yang ditetapkan untuk domain-

name dan domain-name-servers seperti Options dalam file. DHCP options set Nilai defaultnya adalah domain-name:<region>.compute.internal dan domain-name-servers:AmazonProvidedDNS. Untuk informasi selengkapnya, lihat Set opsi DHCP di Panduan Pengguna Amazon VPC.

- Jika node dalam grup node terkelola tidak terhubung ke cluster dalam waktu 15 menit, masalah kesehatan "NodeCreationFailure" akan dipancarkan dan status konsol akan disetel ke. Create failed Untuk Windows AMI yang memiliki waktu peluncuran lambat, masalah ini dapat diselesaikan menggunakan peluncuran cepat.

Untuk mengidentifikasi dan memecahkan masalah penyebab umum yang mencegah node pekerja bergabung dengan cluster, Anda dapat menggunakan runbook. AWSSupport-TroubleshootEKSWorkerNode Untuk informasi selengkapnya, lihat AWSSupport-TroubleshootEKSWorkerNode di referensi buku runbook Otomatisasi AWS Systems Manager .

Tidak sah atau akses ditolak (**kubectl**)

Jika Anda menerima salah satu kesalahan berikut saat menjalankan kubectl perintah, maka Anda belum kubectl mengonfigurasi dengan benar untuk Amazon EKS atau kredensi untuk prinsipal IAM (peran atau pengguna) yang Anda gunakan jangan dipetakan ke Kubernetes nama pengguna yang memiliki izin yang cukup untuk objek Kubernetes di kluster Amazon EKS Anda.

- `could not get token: AccessDenied: Access denied`
- `error: You must be logged in to the server (Unauthorized)`
- `error: the server doesn't have a resource type "svc"`

Ini bisa disebabkan oleh salah satu alasan berikut:

- Cluster dibuat dengan kredensi untuk satu prinsipal IAM dan kubectl dikonfigurasi untuk menggunakan kredensi untuk prinsipal IAM yang berbeda. Untuk mengatasi hal ini, perbarui kube config file Anda untuk menggunakan kredensi yang membuat klaster. Untuk informasi selengkapnya, lihat Membuat atau memperbarui kubeconfig file untuk klaster Amazon EKS.
- Jika klaster Anda memenuhi persyaratan platform minimum di bagian prasyaratKelola entri akses, entri akses tidak ada dengan prinsipal IAM Anda. Jika ada, itu tidak memiliki nama Kubernetes grup yang diperlukan yang ditentukan untuknya, atau tidak memiliki kebijakan akses yang tepat yang terkait dengannya. Untuk informasi selengkapnya, lihat Kelola entri akses.

- Jika klaster Anda tidak memenuhi persyaratan platform minimum diKelola entri akses, entri dengan prinsipal IAM Anda tidak ada di `aws-authConfigMap`. Jika ada, itu tidak dipetakan ke nama Kubernetes grup yang terikat ke Kubernetes `Role` atau `ClusterRole` dengan izin yang diperlukan. Untuk informasi selengkapnya tentang objek otorisasi Kubernetes berbasis peran (RBAC), lihat Menggunakan otorisasi RBAC dalam dokumentasi. Kubernetes Anda dapat melihat `aws-auth` `ConfigMap` entri Anda saat ini dengan mengganti *`my-cluster`* dalam perintah berikut dengan nama cluster Anda dan kemudian menjalankan perintah yang dimodifikasi:. **`eksctl get iamidentitymapping --cluster`** ***`my-cluster`*** Jika entri untuk ARN kepala sekolah IAM Anda tidak ada di`ConfigMap`, masukkan **`eksctl create iamidentitymapping --help`** terminal Anda untuk mempelajari cara membuatnya.

Jika Anda menginstal dan mengkonfigurasi AWS CLI, Anda dapat mengkonfigurasi kredenal IAM yang Anda gunakan. Untuk informasi lebih lanjut, lihat Mengonfigurasi AWS CLI di Panduan Pengguna AWS Command Line Interface . Anda juga dapat mengonfigurasi `kubectl` untuk menggunakan peran IAM, jika Anda mengasumsikan peran IAM untuk mengakses Kubernetes objek di klaster Anda. Untuk informasi selengkapnya, lihat Membuat atau memperbarui `kubeconfig` file untuk klaster Amazon EKS.

`hostname doesn't match`

Versi Python sistem Anda harus `2.7.9` atau lebih baru. Jika tidak, Anda menerima `hostname doesn't match` kesalahan dengan AWS CLI panggilan ke Amazon EKS. Untuk informasi selengkapnya, lihat Apa itu kesalahan “nama host tidak cocok”? di Python Permintaan Pertanyaan yang Sering Diajukan.

`getsockopt: no route to host`

Dockerberjalan dalam kisaran `172.17.0.0/16` CIDR di kluster Amazon EKS. Kami merekomendasikan agar sub-jaringan VPC klaster Anda tidak tumpang tindih dengan kisaran ini. Jika tidak, Anda akan menerima kesalahan berikut:

```
Error: : error upgrading connection: error dialing backend: dial tcp
 172.17.<nn>.<nn>:10250: getsockopt: no route to host
```

Instances failed to join the Kubernetes cluster

Jika Anda menerima kesalahan `Instances failed to join the Kubernetes cluster` dalam AWS Management Console, pastikan bahwa akses endpoint pribadi kluster diaktifkan, atau bahwa Anda telah mengonfigurasi blok CIDR dengan benar untuk akses titik akhir publik. Untuk informasi selengkapnya, lihat Kendali akses titik akhir klaster Amazon EKS.

Kode kesalahan grup node terkelola

Jika grup node terkelola Anda mengalami masalah kesehatan perangkat keras, Amazon EKS mengembalikan kode kesalahan untuk membantu Anda mendiagnosis masalah tersebut. Pemeriksaan kesehatan ini tidak mendeteksi masalah perangkat lunak karena didasarkan pada pemeriksaan kesehatan Amazon EC2. Daftar berikut menjelaskan kode kesalahan.

AccessDenied

Amazon EKS atau satu atau beberapa node terkelola Anda gagal mengautentikasi atau mengotorisasi dengan server API Kubernetes cluster Anda. Untuk informasi lebih lanjut tentang menyelesaikan penyebab umum, lihatMemperbaiki penyebab umum `AccessDenied` kesalahan untuk grup node terkelola. WindowsAMI pribadi juga dapat menyebabkan kode kesalahan ini di samping pesan `Not authorized for images` kesalahan. Untuk informasi selengkapnya, lihat `Not authorized for images`.

AmiIdNotFound

Kami tidak dapat menemukan ID AMI yang terkait dengan template peluncuran Anda. Pastikan bahwa AMI ada dan dibagikan dengan akun Anda.

AutoScalingGroupNotFound

Kami tidak dapat menemukan grup Auto Scaling yang terkait dengan grup node terkelola. Anda dapat membuat ulang grup Auto Scaling dengan pengaturan yang sama untuk memulihkan.

ClusterUnreachable

Amazon EKS atau satu atau beberapa node terkelola tidak dapat berkomunikasi dengan server API Kubernetes cluster Anda. Hal ini dapat terjadi jika ada gangguan jaringan atau jika waktu keluar permintaan pemrosesan server API.

EC2 SecurityGroupNotFound

Kami tidak dapat menemukan grup keamanan cluster untuk cluster. Anda harus membuat ulang klaster Anda.

EC2 SecurityGroupDeletionFailure

Kami tidak dapat menghapus grup keamanan akses jarak jauh untuk grup node terkelola Anda. Menghapus dependensi dari grup keamanan.

EC2 LaunchTemplateNotFound

Kami tidak dapat menemukan template peluncuran Amazon EC2 untuk grup node terkelola Anda. Anda harus membuat ulang grup node Anda untuk memulihkan.

EC2 LaunchTemplateVersionMismatch

Versi template peluncuran Amazon EC2 untuk grup node terkelola Anda tidak cocok dengan versi yang dibuat Amazon EKS. Anda mungkin dapat kembali ke versi yang dibuat Amazon EKS untuk memulihkan.

IamInstanceProfileNotFound

Kami tidak dapat menemukan profil instans IAM untuk grup node terkelola Anda. Anda mungkin dapat membuat ulang profil instans dengan pengaturan yang sama untuk memulihkan.

IamNodeRoleNotFound

Kami tidak dapat menemukan peran IAM untuk grup node terkelola Anda. Anda mungkin dapat membuat ulang IAM role dengan pengaturan yang sama untuk memulihkan.

AsgInstanceLaunchFailures

Grup Auto Scaling Anda mengalami kegagalan saat mencoba meluncurkan instance.

NodeCreationFailure

Instans yang Anda luncurkan tidak dapat mendaftar dengan kluster Amazon EKS Anda. Penyebab umum kegagalan ini adalah tidak mencukupi IAM role simpul izin atau kurangnya akses internet keluar untuk simpul. Node Anda harus memenuhi salah satu dari persyaratan berikut:

- Mampu mengakses internet menggunakan alamat IP publik. Kelompok keamanan yang terkait dengan subnet tempat node berada harus mengizinkan komunikasi. Lihat informasi yang lebih lengkap di Persyaratan dan pertimbangan subnet dan Persyaratan dan pertimbangan grup keamanan Amazon EKS.

- Node dan VPC Anda harus memenuhi persyaratan di. Persyaratan klaster pribadi

InstanceLimitExceeded

AWS Akun Anda tidak dapat meluncurkan instance lagi dari jenis instans yang ditentukan. Anda mungkin dapat meminta peningkatan batas instans Amazon EC2 untuk memulihkan.

InsufficientFreeAddresses

Satu atau beberapa subnet yang terkait dengan grup node terkelola Anda tidak memiliki cukup alamat IP yang tersedia untuk node baru.

InternalFailure

Kesalahan ini biasanya disebabkan oleh masalah sisi server Amazon EKS.

Memperbaiki penyebab umum **`AccessDenied`** kesalahan untuk grup node terkelola

Penyebab paling umum kesalahan `AccessDenied` saat melakukan pengoperasian pada grup simpul yang terkelola adalah hilangnya `eks:node-manager` `ClusterRole` atau `ClusterRoleBinding`. Amazon EKS mengatur sumber daya ini di klaster Anda sebagai bagian dari orientasi dengan grup simpul yang terkelola, dan ini diperlukan untuk mengelola grup simpul.

`ClusterRole` mungkin berubah seiring berjalannya waktu, tetapi seharusnya terlihat serupa dengan contoh berikut:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:node-manager
rules:
- apiGroups:
  - ''
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
  - delete
- apiGroups:
  - ''
  resources:
```

```
  - nodes
  verbs:
  - get
  - list
  - watch
  - patch
- apiGroups:
  - ''
  resources:
  - pods/eviction
  verbs:
  - create
```

`ClusterRoleBinding` mungkin berubah seiring berjalannya waktu, tetapi seharusnya terlihat serupa dengan contoh berikut:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:node-manager
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:node-manager
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: eks:node-manager
```

Verifikasi bahwa `eks:node-manager` `ClusterRole` ada.

```
kubectl describe clusterrole eks:node-manager
```

Jika ada, membandingkan output dengan contoh `ClusterRole` sebelumnya.

Verifikasi bahwa `eks:node-manager` `ClusterRoleBinding` ada.

```
kubectl describe clusterrolebinding eks:node-manager
```

Jika ada, bandingkan output dengan contoh `ClusterRoleBinding` sebelumnya.

Jika Anda telah mengidentifikasi `ClusterRole` atau `ClusterRoleBinding` yang hilang atau rusak sebagai penyebab `AcessDenied` ketika meminta pengoperasian grup simpul yang terkelola, Anda dapat mengembalikannya. Simpan isi berikut ke file bernama *`eks-node-manager-role.yaml`*.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:node-manager
rules:
- apiGroups:
  - ''
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
  - delete
- apiGroups:
  - ''
  resources:
  - nodes
  verbs:
  - get
  - list
  - watch
  - patch
- apiGroups:
  - ''
  resources:
  - pods/eviction
  verbs:
  - create
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:node-manager
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:node-manager
subjects:
```

```
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: eks:node-manager
```

Terapkan file.

```
kubectl apply -f eks-node-manager-role.yaml
```

Coba lagi mengoperasikan grup simpul untuk melihat apakah itu menyelesaikan masalah Anda.

Not authorized for images

Salah satu penyebab potensial pesan `Not authorized for images` kesalahan adalah menggunakan Amazon EKS Windows AMI pribadi untuk meluncurkan grup node Windows terkelola. Setelah merilis Windows AMI baru, AWS membuat AMI yang lebih tua dari 4 bulan pribadi, yang membuatnya tidak lagi dapat diakses. Jika grup node terkelola menggunakan Windows AMI pribadi, pertimbangkan untuk memperbarui grup node Windows terkelola Anda. Meskipun kami tidak dapat menjamin bahwa kami dapat menyediakan akses ke AMI yang telah dibuat pribadi, Anda dapat meminta akses dengan mengajukan tiket ke AWS Support. Untuk informasi selengkapnya, lihat Patch, pembaruan keamanan, dan ID AMI di Panduan Pengguna Amazon EC2 untuk Instans Windows.

Node dalam **NotReady** keadaan

Jika node Anda memasuki `NotReady` status, ini mungkin menunjukkan bahwa node tidak sehat dan tidak tersedia untuk menjadwalkan baruPods. Hal ini dapat terjadi karena berbagai alasan, seperti node kekurangan sumber daya yang cukup untuk CPU, memori, atau ruang disk yang tersedia.

Untuk Windows AMI yang dioptimalkan Amazon EKS, tidak ada reservasi untuk sumber daya komputasi yang ditentukan secara default dalam `kubelet` konfigurasi. Untuk membantu mencegah masalah sumber daya, Anda dapat memesan sumber daya komputasi untuk proses sistem `kubelet` dengan memberikan nilai konfigurasi untuk `kube-reserved`dan/atau `system-reserved`. Anda melakukan ini menggunakan parameter `-KubeletExtraArgs` baris perintah dalam skrip bootstrap. Untuk informasi selengkapnya, lihat Reserve Compute Resources for System Daemons dalam Kubernetes dokumentasi dan parameter konfigurasi skrip Bootstrap dalam panduan pengguna ini.

Alat pengumpulan log CNI

Ini Amazon VPC CNI plugin for Kubernetes memiliki skrip pemecahan masalah sendiri yang tersedia di node di. `/opt/cni/bin/aws-cni-support.sh` Anda dapat menggunakan skrip untuk mengumpulkan log diagnostik untuk kasus dukungan dan pemecahan masalah umum.

Menggunakan perintah berikut ini untuk menjalankan skrip di simpul Anda:

```
sudo bash /opt/cni/bin/aws-cni-support.sh
```

> **Note**
>
> Jika skrip tidak ada di lokasi tersebut, maka kontainer CNI gagal untuk menjalankan. Anda dapat mengunduh dan menjalankan skrip secara manual dengan perintah berikut:
>
> ```
> curl -O https://raw.githubusercontent.com/awslabs/amazon-eks-ami/master/log-collector-script/linux/eks-log-collector.sh
> sudo bash eks-log-collector.sh
> ```

Skrip mengumpulkan informasi diagnostik berikut. Versi CNI yang telah di-deploy dapat lebih awal dari versi skrip.

```
      This is version 0.6.1. New versions can be found at https://github.com/awslabs/amazon-eks-ami

Trying to collect common operating system logs...
Trying to collect kernel logs...
Trying to collect mount points and volume information...
Trying to collect SELinux status...
Trying to collect iptables information...
Trying to collect installed packages...
Trying to collect active system services...
Trying to collect Docker daemon information...
Trying to collect kubelet information...
Trying to collect L-IPAMD information...
Trying to collect sysctls information...
Trying to collect networking information...
Trying to collect CNI configuration information...
Trying to collect running Docker containers and gather container data...
```

```
Trying to collect Docker daemon logs...
Trying to archive gathered information...

 Done... your bundled logs are located in /var/
log/eks_i-0717c9d54b6cfaa19_2020-03-24_0103-UTC_0.6.1.tar.gz
```

Informasi diagnostik dikumpulkan dan disimpan di:

```
/var/log/eks_i-0717c9d54b6cfaa19_2020-03-24_0103-UTC_0.6.1.tar.gz
```

Jaringan waktu aktif kontainer belum siap

Anda mungkin mendapatkan kesalahan `Container runtime network not ready` dan otorisasi kesalahan yang mirip dengan berikut ini:

```
4191 kubelet.go:2130] Container runtime network not ready: NetworkReady=false
 reason:NetworkPluginNotReady message:docker: network plugin is not ready: cni config
 uninitialized
4191 reflector.go:205] k8s.io/kubernetes/pkg/kubelet/kubelet.go:452: Failed to list
 *v1.Service: Unauthorized
4191 kubelet_node_status.go:106] Unable to register node
 "ip-10-40-175-122.ec2.internal" with API server: Unauthorized
4191 reflector.go:205] k8s.io/kubernetes/pkg/kubelet/kubelet.go:452: Failed to list
 *v1.Service: Unauthorized
```

Ini dapat terjadi karena salah satu alasan berikut:

1. Anda juga tidak memiliki `aws-auth` `ConfigMap` di cluster Anda atau tidak menyertakan entri untuk peran IAM yang Anda konfigurasikan dengan node Anda.

   `ConfigMap`Entri ini diperlukan jika node Anda memenuhi salah satu kriteria berikut:

   - Node terkelola dalam cluster dengan versi platform Kubernetes atau apa pun.
   - Node yang dikelola sendiri di cluster yang lebih awal dari salah satu versi platform yang tercantum di bagian prasyarat topik. Kelola entri akses

   Untuk mengatasi masalah ini, lihat entri yang ada di Anda `ConfigMap` dengan mengganti *`my-cluster`* dalam perintah berikut dengan nama cluster Anda, lalu jalankan perintah yang dimodifikasi:. **`eksctl get iamidentitymapping --cluster`** ***`my-cluster`*** Jika Anda menerima pesan kesalahan dari perintah, itu mungkin karena klaster Anda tidak memiliki file `aws-`

`authConfigMap`. Perintah berikut menambahkan entri ke`ConfigMap`. Jika `ConfigMap` tidak ada, perintah juga membuatnya. Ganti *111122223333* dengan Akun AWS ID untuk peran IAM dan *NodeRoleMyAmazoneks* dengan nama peran node Anda.

```
eksctl create iamidentitymapping --cluster my-cluster \
    --arn arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --group
 system:bootstrappers,system:nodes \
    --username system:node:{{EC2PrivateDNSName}}
```

ARN peran yang Anda tentukan tidak dapat menyertakan jalur selain. / Misalnya, jika nama peran Anda`development/apps/my-role`, Anda harus mengubahnya menjadi `my-role` saat menentukan ARN peran tersebut. Pastikan Anda menentukan ARN peran IAM node (bukan ARN profil instance).

2. Node yang dikelola sendiri berada dalam klaster dengan versi platform pada versi minimum yang tercantum dalam prasyarat dalam Kelola entri akses topik, tetapi entri tidak tercantum dalam `aws-auth` `ConfigMap` (lihat item sebelumnya) untuk peran IAM node atau entri akses tidak ada untuk peran tersebut. Untuk mengatasi masalah ini, lihat entri akses yang ada dengan mengganti *my-cluster* dalam perintah berikut dengan nama cluster Anda, lalu jalankan perintah yang dimodifikasi:. **aws eks list-access-entries --cluster-name *my-cluster*** Perintah berikut menambahkan entri akses untuk peran IAM node. Ganti *111122223333* dengan Akun AWS ID untuk peran IAM dan *NodeRoleMyAmazoneks* dengan nama peran node Anda. Jika Anda memiliki node Windows, ganti *EC2_Linux* dengan. **EC2_Windows** Pastikan Anda menentukan ARN peran IAM node (bukan ARN profil instance).

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --type EC2_Linux
```

Waktu habis handshake TLS

Ketika node tidak dapat membuat koneksi ke titik akhir server API publik, Anda mungkin melihat kesalahan yang mirip dengan kesalahan berikut.

```
server.go:233] failed to run Kubelet: could not init cloud provider "aws": error
 finding instance i-1111f2222f333e44c: "error listing AWS instances: \"RequestError:
 send request failed\\ncaused by: Post  net/http: TLS handshake timeout\""
```

Proses `kubelet` akan terus-menerus me-respawn dan menguji titik akhir server API. Kesalahan juga dapat terjadi sementara selama prosedur yang performa update klaster bergulir di bidang kendali, seperti perubahan konfigurasi atau versi update.

Untuk mengatasi masalah itu, memeriksa tabel rute dan grup keamanan untuk memastikan bahwa lalu lintas dari simpul dapat mencapai titik akhir publik.

InvalidClientTokenId

Jika Anda menggunakan peran IAM untuk akun layanan untuk Pod atau `DaemonSet` diterapkan ke klaster di China Wilayah AWS, dan belum menyetel variabel `AWS_DEFAULT_REGION` lingkungan dalam spesifikasi, Pod atau `DaemonSet` mungkin menerima kesalahan berikut:

```
An error occurred (InvalidClientTokenId) when calling the GetCallerIdentity operation:
 The security token included in the request is invalid
```

Untuk mengatasi masalah ini, Anda perlu menambahkan variabel `AWS_DEFAULT_REGION` lingkungan ke Pod atau `DaemonSet` spesifikasi Anda, seperti yang ditunjukkan dalam contoh Pod spesifikasi berikut.

```
apiVersion: v1
kind: Pod
metadata:
  name: envar-demo
  labels:
    purpose: demonstrate-envars
spec:
  containers:
  - name: envar-demo-container
    image: gcr.io/google-samples/node-hello:1.0
    env:
    - name: AWS_DEFAULT_REGION
      value: "region-code"
```

Sertifikat penerimaan VPC webhook kedaluwarsa

Jika sertifikat yang digunakan untuk menandatangani webhook penerimaan VPC kedaluwarsa, status penerapan baru Windows Pod tetap berlaku. `ContainerCreating`

Untuk mengatasi masalah jika Anda memiliki Windows dukungan lama di bidang data, lihatMemperbarui sertifikat webhook izin masuk VPC. Jika versi klaster dan platform Anda lebih lambat dari versi yang tercantum dalam prasyarat Windows dukungan, maka sebaiknya Anda menghapus Windows dukungan lama di bidang data Anda dan mengaktifkannya untuk bidang kontrol Anda. Setelah Anda melakukannya, Anda tidak perlu mengelola sertifikat webhook. Untuk informasi selengkapnya, lihat Mengaktifkan Windows dukungan untuk klaster Amazon EKS Anda.

Grup node harus cocok dengan Kubernetes versi sebelum memutakhirkan bidang kontrol

Sebelum Anda memutakhirkan bidang kontrol ke Kubernetes versi baru, versi minor dari node terkelola dan Fargate di cluster Anda harus sama dengan versi versi pesawat kontrol Anda saat ini. Amazon EKS `update-cluster-version` API menolak permintaan hingga Anda memutakhirkan semua node yang dikelola Amazon EKS ke versi cluster saat ini. Amazon EKS menyediakan API untuk memutakhirkan node terkelola. Untuk informasi tentang memutakhirkan Kubernetes versi grup node terkelola, lihatMemperbarui grup simpul terkelola. Untuk memutakhirkan versi node Fargate, hapus pod yang diwakili oleh node dan terapkan kembali pod setelah Anda memutakhirkan bidang kontrol Anda. Untuk informasi selengkapnya, lihat Memperbarui Kubernetes versi cluster Amazon EKS.

Saat meluncurkan banyak node, ada **`Too Many Requests`** kesalahan

Jika Anda meluncurkan banyak node secara bersamaan, Anda mungkin melihat pesan kesalahan di log eksekusi data pengguna Amazon EC2 yang mengatakan. `Too Many Requests` Ini dapat terjadi karena bidang kontrol sedang kelebihan beban dengan `describeCluster` panggilan. Kelebihan beban mengakibatkan pelambatan, node gagal menjalankan skrip bootstrap, dan node gagal bergabung dengan cluster sama sekali.

Pastikan bahwa`--apiserver-endpoint`,`--b64-cluster-ca`, dan `--dns-cluster-ip` argumen sedang diteruskan ke skrip bootstrap node. Saat memasukkan argumen ini, skrip bootstrap tidak perlu melakukan `describeCluster` panggilan, yang membantu mencegah bidang kontrol kelebihan beban. Untuk informasi selengkapnya, lihat Berikan data pengguna untuk meneruskan argumen ke `bootstrap.sh` file yang disertakan dengan Amazon EKS yang dioptimalkanLinux/BottlerocketAMI.

HTTP 401 respons kesalahan tidak sah pada permintaan server Kubernetes API

Anda melihat kesalahan ini jika token akun layanan telah kedaluwarsa di klaster. Pod

Server Kubernetes API klaster Amazon EKS Anda menolak permintaan dengan token yang lebih lama dari 90 hari. Dalam Kubernetes versi sebelumnya, token tidak memiliki kedaluwarsa. Ini berarti bahwa klien yang mengandalkan token ini harus menyegarkannya dalam waktu satu jam. Untuk mencegah server Kubernetes API menolak permintaan Anda karena token tidak valid, versi SDK Kubernetes klien yang digunakan oleh beban kerja Anda harus sama, atau lebih lambat dari versi berikut:

- Versi Go `0.15.7` dan yang lebih baru
- Versi `12.0.0` Python dan yang lebih baru
- Versi Java `9.0.0` dan yang lebih baru
- JavaScript versi `0.10.3` dan yang lebih baru
- Cabang Ruby `master`
- Versi Haskell `0.3.0.0`
- C#versi `7.0.5` dan yang lebih baru

Anda dapat mengidentifikasi semua yang ada Pods di cluster Anda yang menggunakan token basi. Untuk informasi selengkapnya, lihat akun Kubernetes layanan.

Versi platform Amazon EKS lebih dari dua versi di belakang versi platform saat ini

Ini dapat terjadi ketika Amazon EKS tidak dapat memperbarui versi platform cluster Anda secara otomatis. Meskipun ada banyak penyebab untuk ini, beberapa penyebab umum mengikuti. Jika salah satu dari masalah ini berlaku untuk cluster Anda, mungkin masih berfungsi, versi platformnya tidak akan diperbarui oleh Amazon EKS.

Masalah

Peran IAM cluster telah dihapus — Peran ini ditentukan saat cluster dibuat. Anda dapat melihat peran mana yang ditentukan dengan perintah berikut. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-cluster --name my-cluster --query cluster.roleArn --output text | cut
 -d / -f 2
```

Contoh output adalah sebagai berikut.

```
eksClusterRole
```

Solusi

Buat peran IAM cluster baru dengan nama yang sama.

Masalah

Subnet yang ditentukan selama pembuatan cluster telah dihapus — Subnet yang digunakan dengan cluster ditentukan selama pembuatan cluster. Anda dapat melihat subnet mana yang ditentukan dengan perintah berikut. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-cluster --name my-cluster --query cluster.resourcesVpcConfig.subnetIds
```

Contoh output adalah sebagai berikut.

```
[
"subnet-EXAMPLE1",
"subnet-EXAMPLE2"
]
```

Solusi

Konfirmasikan apakah ID subnet ada di akun Anda.

```
vpc_id=$(aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.vpcId --output text)
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$vpc_id" --query
 "Subnets[*].SubnetId"
```

Contoh output adalah sebagai berikut.

```
[
"subnet-EXAMPLE3",
"subnet-EXAMPLE4"
]
```

Jika ID subnet yang dikembalikan dalam output tidak cocok dengan ID subnet yang ditentukan saat cluster dibuat, maka jika Anda ingin Amazon EKS memperbarui cluster, Anda perlu mengubah subnet yang digunakan oleh cluster. Ini karena jika Anda menentukan lebih dari dua subnet saat membuat klaster, Amazon EKS secara acak memilih subnet yang Anda tentukan untuk membuat antarmuka jaringan elastis baru. Antarmuka jaringan ini memungkinkan bidang kontrol untuk berkomunikasi dengan node Anda. Amazon EKS tidak akan memperbarui cluster jika subnet yang dipilihnya tidak ada. Anda tidak memiliki kendali atas subnet mana yang Anda tentukan pada pembuatan cluster yang dipilih Amazon EKS untuk membuat antarmuka jaringan baru.

Saat Anda memulai pembaruan Kubernetes versi untuk klaster Anda, pembaruan dapat gagal karena alasan yang sama.

Masalah

Grup keamanan yang ditentukan selama pembuatan klaster telah dihapus — Jika Anda menetapkan grup keamanan selama pembuatan klaster, Anda dapat melihat ID mereka dengan perintah berikut. Ganti *my-cluster* dengan nama klaster Anda.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.securityGroupIds
```

Contoh output adalah sebagai berikut.

```
[
    "sg-EXAMPLE1"
]
```

Jika `[]` dikembalikan, maka tidak ada grup keamanan yang ditentukan saat cluster dibuat dan grup keamanan yang hilang bukanlah masalahnya. Jika grup keamanan dikembalikan, maka konfirmasikan bahwa grup keamanan ada di akun Anda.

Solusi

Konfirmasikan apakah grup keamanan ini ada di akun Anda.

```
vpc_id=$(aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.vpcId --output text)
aws ec2 describe-security-groups --filters "Name=vpc-id,Values=$vpc_id" --query
 "SecurityGroups[*].GroupId"
```

Contoh output adalah sebagai berikut.

```
[
"sg-EXAMPLE2"
]
```

Jika ID grup keamanan yang ditampilkan dalam output tidak cocok dengan ID grup keamanan yang ditentukan saat kluster dibuat, maka jika Anda ingin Amazon EKS memperbarui klaster, Anda perlu mengubah grup keamanan yang digunakan oleh cluster. Amazon EKS tidak akan memperbarui klaster jika ID grup keamanan yang ditentukan pada pembuatan klaster tidak ada.

Saat Anda memulai pembaruan Kubernetes versi untuk klaster Anda, pembaruan dapat gagal karena alasan yang sama.

Alasan lain mengapa Amazon EKS tidak memperbarui versi platform cluster Anda

- Anda tidak memiliki setidaknya enam (meskipun kami merekomendasikan 16) alamat IP yang tersedia di setiap subnet yang Anda tentukan saat Anda membuat cluster Anda. Jika Anda tidak memiliki cukup alamat IP yang tersedia di subnet, Anda perlu membebaskan alamat IP di subnet atau Anda perlu mengubah subnet yang digunakan oleh cluster untuk menggunakan subnet dengan alamat IP yang cukup tersedia.
- Anda mengaktifkan enkripsi rahasia saat membuat klaster dan AWS KMS kunci yang Anda tentukan telah dihapus. Jika Anda ingin Amazon EKS memperbarui cluster, Anda perlu membuat cluster baru

FAQ kesehatan cluster dan kode kesalahan dengan jalur resolusi

Amazon EKS mendeteksi masalah dengan kluster EKS Anda dan infrastruktur klaster dan menyimpannya di kesehatan klaster. Anda dapat mendeteksi, memecahkan masalah, dan mengatasi masalah klaster lebih cepat dengan bantuan informasi kesehatan cluster. Ini memungkinkan Anda untuk membuat lingkungan aplikasi yang lebih aman dan up-to-date. Selain itu, mungkin tidak mungkin bagi Anda untuk memutakhirkan ke versi yang lebih baru Kubernetes atau Amazon EKS untuk menginstal pembaruan keamanan pada klaster yang terdegradasi sebagai akibat dari masalah dengan infrastruktur atau konfigurasi cluster yang diperlukan. Amazon EKS dapat memakan waktu 3 jam untuk mendeteksi masalah atau mendeteksi bahwa masalah telah teratasi.

Kesehatan cluster Amazon EKS adalah tanggung jawab bersama antara Amazon EKS dan penggunanya. Anda bertanggung jawab atas infrastruktur prasyarat peran IAM dan subnet Amazon VPC, serta infrastruktur lain yang diperlukan, yang harus disediakan terlebih dahulu. Amazon EKS mendeteksi perubahan dalam konfigurasi infrastruktur ini dan cluster.

Untuk mengakses kesehatan klaster Anda di konsol Amazon EKS, cari bagian yang disebut Masalah Kesehatan di tab Ikhtisar halaman detail klaster Amazon EKS. Data ini juga akan tersedia dengan memanggil `DescribeCluster` tindakan di EKS API, misalnya dari dalam AWS Command Line Interface.

Mengapa saya harus menggunakan fitur ini?

Anda akan mendapatkan peningkatan visibilitas ke dalam kesehatan kluster Amazon EKS Anda, dengan cepat mendiagnosis dan memperbaiki masalah apa pun, tanpa perlu menghabiskan waktu debugging atau membuka AWS kasus dukungan. Misalnya: Anda secara tidak sengaja menghapus subnet untuk kluster Amazon EKS, Amazon EKS tidak akan dapat membuat antarmuka dan Kubernetes AWS CLI perintah jaringan lintas akun seperti `kubectl` exec atau log. `kubectl` Ini akan gagal dengan kesalahan: `Error from server: error dialing backend: remote error: tls: internal error.` Sekarang Anda akan melihat masalah kesehatan Amazon EKS yang mengatakan:`subnet-da60e280 was deleted: could not create network interface`.

Bagaimana fitur ini berhubungan atau bekerja dengan AWS layanan lain?

Peran IAM dan subnet VPC Amazon adalah dua contoh infrastruktur prasyarat yang mendeteksi masalah kesehatan klaster. Fitur ini akan mengembalikan informasi terperinci jika sumber daya tersebut tidak dikonfigurasi dengan benar.

Apakah cluster dengan masalah kesehatan menimbulkan biaya?

Ya, setiap cluster Amazon EKS ditagih dengan harga Amazon EKS standar. Fitur kesehatan cluster tersedia tanpa biaya tambahan.

Apakah fitur ini berfungsi dengan kluster Amazon AWS Outposts EKS?

Ya, masalah cluster terdeteksi untuk kluster EKS di AWS Cloud termasuk cluster yang diperluas AWS Outposts dan cluster lokal aktif. AWS Outposts Kesehatan cluster tidak mendeteksi masalah dengan Amazon EKS Anywhere atau Amazon EKS Distro (EKS-D).

Bisakah saya mendapatkan pemberitahuan ketika masalah baru terdeteksi?

Tidak, Anda perlu memeriksa Amazon EKS Console atau memanggil EKS `DescribeCluster` API.

Apakah konsol memberi saya peringatan untuk masalah kesehatan?

Ya, setiap cluster dengan masalah kesehatan akan menyertakan spanduk di bagian atas konsol.

Dua kolom pertama adalah apa yang dibutuhkan untuk nilai respons API. Bidang ketiga dari ClusterIssue objek Kesehatan adalahresourceIds, pengembaliannya tergantung pada jenis masalah.

Kode	Pesan	ResourceIds	Cluster Dapat Dipulihkan?	
SUBNET_NOT_FOUND	Kami tidak dapat menemukan satu atau lebih subnet yang saat ini terkait dengan cluster Anda. Hubungi Amazon EKS update-cluster-config API untuk memperbarui subnet.	Id Subnet	Ya	
SECURITY_GROUP_NOT_FOUND	Kami tidak dapat menemukan satu atau lebih grup keamanan yang saat ini terkait dengan klaster Anda. Hubungi Amazon EKS update-cluster-config API untuk memperbarui grup keamanan	Id grup keamanan	Ya	
IP_NOT_AVAILABLE	Satu atau beberapa subnet yang terkait dengan cluster Anda tidak memiliki cukup alamat IP yang tersedia untuk Amazon EKS untuk melakukan operasi manajemen klaster. Kosongkan alamat di subnet, atau kaitkan subnet yang berbeda ke klaster Anda menggunakan Amazon EKS update-cluster-config API.	Id Subnet	Ya	
VPC_NOT_FOUND	Kami tidak dapat menemukan VPC yang terkait dengan cluster Anda. Anda harus menghapus dan membuat ulang cluster Anda.	VPC id	Tidak	

Kode	Pesan	ResourceIds	Cluster Dapat Dipulihkan?	
ASSUME_ROLE_ACCESS_DENIED	Cluster Anda tidak menggunakan Amazon EKS service-linked-role. Kami tidak dapat mengambil peran yang terkait dengan klaster Anda untuk melakukan operasi manajemen Amazon EKS yang diperlukan. Periksa peran yang ada dan memiliki kebijakan kepercayaan yang diperlukan.	Peran IAM cluster	Ya	
PERMISSION_ACCESS_DENIED	Cluster Anda tidak menggunakan Amazon EKS service-linked-role. Peran yang terkait dengan klaster Anda tidak memberikan izin yang memadai bagi Amazon EKS untuk melakukan operasi manajemen yang diperlukan. Periksa kebijakan yang dilampirkan pada peran klaster dan jika ada kebijakan penolakan terpisah yang diterapkan.	Peran IAM cluster	Ya	
ASSUME_ROLE_ACCESS_DENIED_USING_SLR	Kami tidak dapat mengasumsikan manajemen cluster Amazon EKS service-linked-role. Periksa peran yang ada dan memiliki kebijakan kepercayaan yang diperlukan.	Amazon EKS service-linked-role	Ya	

Kode	Pesan	ResourceIds	Cluster Dapat Dipulihkan?	
PERMISSION_ACCESS_DENIED_USING_SLR	Manajemen klaster Amazon EKS service-linked-role tidak memberikan izin yang cukup bagi Amazon EKS untuk melakukan operasi manajemen yang diperlukan. Periksa kebijakan yang dilampirkan pada peran klaster dan jika ada kebijakan penolakan terpisah yang diterapkan.	Amazon EKS service-linked-role	Ya	
OPT_IN_REQUIRED	Akun Anda tidak memiliki langganan layanan Amazon EC2. Perbarui langganan akun Anda di halaman pengaturan akun Anda.	N/A	Ya	
STS_REGIONAL_ENDPOINT_DISABLED	Titik akhir STS regional dinonaktifkan. Aktifkan titik akhir Amazon EKS untuk melakukan operasi manajemen klaster yang diperlukan.	N/A	Ya	
KMS_KEY_DISABLED	AWS KMS Kunci yang terkait dengan cluster Anda dinonaktifkan. Aktifkan kembali kunci untuk memulihkan cluster Anda.	Sebuah KMS Key Arn	Ya	
KMS_KEY_NOT_FOUND	Kami tidak dapat menemukan AWS KMS kunci yang terkait dengan cluster Anda. Anda harus menghapus dan membuat ulang cluster.	Sebuah KMS Key ARN	Tidak	

Kode	Pesan	ResourceIds	Cluster Dapat Dipulihkan?	
KMS_GRANT_REVOKED	Hibah untuk AWS KMS Kunci yang terkait dengan cluster Anda dicabut. Anda harus menghapus dan membuat ulang cluster.	Sebuah KMS Key Arn	Tidak	

Konektor Amazon EKS

Anda dapat menggunakan Amazon EKS Connector untuk mendaftar dan menghubungkan kesesuaian apa punKubernetescluster keAWSdan memvisualisasikannya di konsol Amazon EKS. Setelah cluster terhubung, Anda dapat melihat status, konfigurasi, dan beban kerja untuk klaster tersebut di konsol Amazon EKS. Anda dapat menggunakan fitur ini untuk melihat kluster yang terhubung di konsol Amazon EKS, tetapi Anda tidak dapat mengelolanya. Konektor Amazon EKS membutuhkan agen yang merupakanproyek open source di Github. Untuk konten teknis tambahan, termasuk pertanyaan umum dan pemecahan masalah, lihatMemecahkan masalah di Amazon EKS Connector

Konektor Amazon EKS dapat menghubungkan jenis berikutKubernetescluster ke Amazon EKS.

- On-premiseKuberneteskelompok
- Cluster yang dikelola sendiri yang berjalan di Amazon EC2
- Cluster terkelola dari penyedia cloud lainnya

Pertimbangan Konektor Amazon EKS

Sebelum Anda menggunakan Amazon EKS Connector, pahami hal berikut:

- Anda harus memiliki hak administratif untukKubernetescluster untuk menghubungkan cluster ke Amazon EKS.
- TheKubernetesCluster harus memilikiLinuxNode pekerja 64-bit (x86) hadir sebelum menghubungkan. Node pekerja ARM tidak didukung.
- Anda harus memiliki node pekerja diKubernetescluster yang memiliki akses keluar kessm.danssmmessages.Titik akhir Manajer Sistem. Untuk informasi lebih lanjut, lihatTitik akhir Manajer SistemdiReferensi Umum AWS.
- Secara default, Anda dapat menghubungkan hingga 10 cluster di Region. Anda dapat meminta peningkatan melaluikonsol kuota layanan. LihatMeminta peningkatan kuotauntuk informasi lebih lanjut.
- Hanya Amazon EKSRegisterCluster,ListClusters,DescribeCluster, danDeregisterClusterAPI didukung untuk eksternalKubernetescluster.
- Anda harus memiliki izin berikut untuk mendaftarkan klaster:

  - eks:RegisterCluster
  - ssm:CreateActivation
  - ssm:DeleteActivation
  - saya:PassRole
- Anda harus memiliki izin berikut untuk membatalkan pendaftaran klaster:
  - eks:DeregisterCluster
  - ssm:DeleteActivation
  - ssm:DeregisterManagedInstance

Peran IAM yang diperlukan untuk Konektor Amazon EKS

Menggunakan Konektor Amazon EKS memerlukan dua peran IAM berikut:

- TheKonektor Amazon EKSperan terkait layanan dibuat saat Anda mendaftarkan klaster untuk pertama kalinya.
- Anda harus membuat peran IAM agen Amazon EKS Connector. Lihat Peran IAM konektor Amazon EKS untuk detail.

Untuk mengaktifkan izin tampilan klaster dan beban kerjaKepala Sekolah IAM, terapkan`eks-connector`dan peran klaster Amazon EKS Connector ke cluster Anda. Ikuti langkah-langkahnya di Memberikan akses ke prinsipal IAM untuk melihat Kubernetes sumber daya di cluster.

Menghubungkan cluster eksternal

Anda dapat menghubungkan Kubernetes cluster eksternal ke Amazon EKS dengan menggunakan beberapa metode dalam proses berikut. Proses ini melibatkan dua langkah: Mendaftarkan cluster dengan Amazon EKS dan menginstal `eks-connector` agen di cluster.

> **Important**
>
> Anda harus menyelesaikan langkah kedua dalam waktu 3 hari setelah menyelesaikan langkah pertama, sebelum pendaftaran berakhir.

Metode konektor

Tidak semua metode untuk menginstal agen dapat digunakan setelah masing-masing metode untuk mendaftarkan cluster. Tabel berikut mencantumkan masing-masing metode pendaftaran dan metode pemasangan agen mana yang dapat digunakan.

Langkah	Metode		
Daftarkan cluster	AWS Management Console	AWS Command Line Interface	`eksctl`
Instal agen	Helm, manifestasi YAMAL	Helm, manifestasi YAMAL	Manifestasi YAMM

Prasyarat

- Pastikan peran agen Amazon EKS Connector telah dibuat. Ikuti langkah-langkahnya di Membuat peran agen konektor Amazon EKS.
- Anda harus memiliki izin berikut untuk mendaftarkan klaster:
  - `eks:RegisterCluster`
  - `ssm:CreateActivation`
  - `ssm:DeleteActivation`
  - `iam:PassRole`

Langkah 1: Mendaftarkan cluster

AWS CLI

Prasyarat

- AWS CLIharus diinstal. Untuk menginstal atau memutakhirkannya, lihat Menginstal AWS CLI.

Untuk mendaftarkan cluster Anda dengan AWS CLI

- Untuk konfigurasi Konektor, tentukan peran IAM agen Amazon EKS Connector Anda. Untuk informasi selengkapnya, lihat Peran IAM yang diperlukan untuk Konektor Amazon EKS.

```
aws eks register-cluster \
     --name my-first-registered-cluster \
     --connector-config roleArn=arn:aws:iam::111122223333:role/
AmazonEKSConnectorAgentRole,provider="OTHER" \
     --region aws-region
```

Contoh output adalah sebagai berikut.

```
{
    "cluster": {
        "name": "my-first-registered-cluster",
        "arn": "arn:aws:eks:region:111122223333:cluster/my-first-registered-
cluster",
        "createdAt": 1627669203.531,
        "ConnectorConfig": {
            "activationId": "xxxxxxxxACTIVATION_IDxxxxxxxx",
            "activationCode": "xxxxxxxxACTIVATION_CODExxxxxxxx",
            "activationExpiry": 1627672543.0,
            "provider": "OTHER",
            "roleArn": "arn:aws:iam::111122223333:role/
AmazonEKSConnectorAgentRole"
        },
        "status": "CREATING"
    }
}
```

Anda menggunakan`aws-region`,`activationId`, dan `activationCode` nilai di langkah berikutnya.

AWS Management Console

Untuk mendaftarkan Kubernetes cluster Anda dengan konsol.

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih Add cluster dan pilih Register untuk membuka halaman konfigurasi.
3. Pada bagian Configure cluster, isi kolom berikut:
   - Nama — Nama unik untuk klaster Anda.

- Provider - Pilih untuk menampilkan daftar dropdown penyedia Kubernetes cluster. Jika Anda tidak tahu penyedia tertentu, pilih Lainnya.
- Peran Konektor EKS - Pilih peran yang akan digunakan untuk menghubungkan cluster.

4. Pilih Daftar cluster.
5. Halaman ikhtisar Cluster ditampilkan. Jika Anda ingin menggunakan bagan Helm, salin `helm install` perintah dan lanjutkan ke langkah berikutnya. Jika Anda ingin menggunakan manifes YAMM, pilih Unduh file YAMM untuk mengunduh file manifes ke drive lokal Anda.

> **Important**
>
> - Ini adalah satu-satunya kesempatan Anda untuk menyalin `helm install` perintah atau mengunduh file ini. Jangan menavigasi jauh dari halaman ini, karena tautan tidak akan dapat diakses dan Anda harus membatalkan pendaftaran cluster dan memulai langkah-langkahnya dari awal.
> - Perintah atau file manifes hanya dapat digunakan sekali untuk cluster terdaftar. Jika Anda menghapus sumber daya dari Kubernetes cluster, Anda harus mendaftarkan ulang cluster dan mendapatkan file manifes baru.

Lanjutkan ke langkah berikutnya untuk menerapkan file manifes ke Kubernetes cluster Anda.

eksctl

Prasyarat

- `eksctl`versi `0.68` atau yang lebih baru harus diinstal. Untuk menginstal atau memutakhirkannya, lihatMemulai dengan Amazon EKS – `eksctl`.

Untuk mendaftarkan cluster Anda dengan **`eksctl`**

1. Daftarkan cluster dengan memberikan nama, penyedia, dan wilayah.

```
eksctl register cluster --name my-cluster --provider my-provider --
region region-code
```

Contoh output:

```
2021-08-19 13:47:26 [#]  creating IAM role "eksctl-202108191941121860 40"
2021-08-19 13:47:26 [#]  registered cluster "<name>" successfully
2021-08-19 13:47:26 [#]  wrote file eks-connector.yaml to <current directory>
2021-08-19 13:47:26 [#]  wrote file eks-connector-clusterrole.yaml to <current
 directory>
2021-08-19 13:47:26 [#]  wrote file eks-connector-console-dashboard-full-access-
group.yaml to <current directory>
2021-08-19 13:47:26 [!]  note: "eks-connector-clusterrole.yaml" and "eks-
connector-console-dashboard-full-access-group.yaml" give full EKS Console access
 to IAM identity "<aws-arn>", edit if required; read https://eksctl.io/usage/
eks-connector for more info
2021-08-19 13:47:26 [#]  run `kubectl apply -f eks-connector.yaml,eks-connector-
clusterrole.yaml,eks-connector-console-dashboard-full-access-group.yaml` before
 expiry> to connect the cluster
```

Ini membuat file di komputer lokal Anda. File-file ini harus diterapkan ke cluster eksternal dalam waktu 3 hari, atau pendaftaran berakhir.

2. Di terminal yang dapat mengakses cluster, terapkan `eks-connector-binding.yaml` file:

```
kubectl apply -f eks-connector-binding.yaml
```

Langkah 2: Memasang **`eks-connector`** agen

Helm chart

1. Jika Anda menggunakan AWS CLI pada langkah sebelumnya, ganti `ACTIVATION_CODE` dan `ACTIVATION_ID` dalam perintah berikut dengan`activationId`, dan `activationCode` nilai masing-masing. Ganti `aws-region` dengan Wilayah AWS yang Anda gunakan pada langkah sebelumnya. Kemudian jalankan perintah untuk menginstal `eks-connector` agen pada cluster pendaftaran:

```
$ helm install eks-connector \
  --namespace eks-connector \
  oci://public.ecr.aws/eks-connector/eks-connector-chart \
  --set eks.activationCode=ACTIVATION_CODE \
  --set eks.activationId=ACTIVATION_ID \
```

```
  --set eks.agentRegion=aws-region
```

Jika Anda menggunakan langkah sebelumnya, gunakan perintah yang Anda salin dari langkah sebelumnya yang memiliki nilai-nilai ini terisi. AWS Management Console

2. Periksa kesehatan `eks-connector` penerapan yang diinstal dan tunggu status cluster terdaftar di Amazon EKS. `ACTIVE`

YAML manifest

Selesaikan koneksi dengan menerapkan file manifes Amazon EKS Connector ke Kubernetes cluster Anda. Untuk melakukan ini, Anda harus menggunakan metode yang dijelaskan sebelumnya. Jika manifes tidak diterapkan dalam tiga hari, pendaftaran Amazon EKS Connector akan kedaluwarsa. Jika koneksi cluster kedaluwarsa, cluster harus dideregistrasi sebelum menghubungkan cluster lagi.

1. Unduh file YAMM Konektor Amazon EKS.

```
curl -O https://amazon-eks.s3.us-west-2.amazonaws.com/eks-connector/manifests/
eks-connector/latest/eks-connector.yaml
```

2. Edit file YAMAL Amazon EKS Connector untuk mengganti semua referensi dari `%AWS_REGION%%EKS_ACTIVATION_ID%`,`aws-region`, `%EKS_ACTIVATION_CODE%` dengan`activationId`,, dan `activationCode` dari output dari langkah sebelumnya.

   Contoh perintah berikut dapat menggantikan nilai-nilai ini.

```
sed -i "s~%AWS_REGION%~$aws-region~g; s~%EKS_ACTIVATION_ID
%~$EKS_ACTIVATION_ID~g; s~%EKS_ACTIVATION_CODE%~$(echo -n $EKS_ACTIVATION_CODE |
 base64)~g" eks-connector.yaml
```

> **Important**
>
> Pastikan kode aktivasi Anda dalam format base64.

3. Di terminal yang dapat mengakses cluster, Anda dapat menerapkan file manifes yang diperbarui dengan menjalankan perintah berikut:

```
kubectl apply -f eks-connector.yaml
```

4. Setelah manifes Amazon EKS Connector dan file YAMM yang mengikat peran diterapkan ke Kubernetes klaster Anda, konfirmasikan bahwa klaster sekarang terhubung.

```
aws eks describe-cluster \
     --name "my-first-registered-cluster" \
     --region AWS_REGION
```

   Outputnya harus mencakup`status=ACTIVE`.

5. (Opsional) Tambahkan tag ke cluster Anda. Untuk informasi selengkapnya, lihat Menandai sumber daya Amazon EKS Anda.

Langkah selanjutnya

Jika Anda memiliki masalah dengan langkah-langkah ini, lihatMemecahkan masalah di Amazon EKS Connector.

Untuk memberikan akses kepada prinsipal IAM tambahan ke konsol Amazon EKS untuk melihat Kubernetes sumber daya di klaster yang terhubung, lihat. Memberikan akses ke prinsipal IAM untuk melihat Kubernetes sumber daya di cluster

Memberikan akses ke prinsipal IAM untuk melihat Kubernetes sumber daya di cluster

Berikan akses kepada prinsipal IAM ke konsol Amazon EKS untuk melihat informasi tentang Kubernetes sumber daya yang berjalan di klaster Anda yang terhubung.

Prasyarat

Prinsipal IAM yang Anda gunakan untuk mengakses AWS Management Console harus memenuhi persyaratan berikut:

- Itu harus memiliki izin `eks:AccessKubernetesApi` IAM.
- Akun layanan Amazon EKS Connector dapat meniru prinsip IAM di cluster. Hal ini memungkinkan Amazon EKS Connector untuk memetakan prinsipal IAM ke Kubernetes pengguna.

Untuk membuat dan menerapkan peran klaster Amazon EKS Connector

1. Unduh templat peran `eks-connector` cluster.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/eks-connector-console-roles/eks-connector-clusterrole.yaml
   ```

2. Edit file YAMAL template peran cluster. Ganti referensi `%IAM_ARN%` dengan Nama Sumber Daya Amazon (ARN) kepala sekolah IAM Anda.
3. Terapkan peran klaster Amazon EKS Connector YAML ke Kubernetes klaster Anda.

   ```
   kubectl apply -f eks-connector-clusterrole.yaml
   ```

Agar prinsipal IAM dapat melihat Kubernetes sumber daya di konsol Amazon EKS, prinsipal harus dikaitkan dengan Kubernetes `role` atau `clusterrole` dengan izin yang diperlukan untuk membaca sumber daya. Untuk informasi selengkapnya, lihat Menggunakan Otorisasi RBAC dalam dokumentasi. Kubernetes

Untuk mengkonfigurasi prinsipal IAM untuk mengakses cluster yang terhubung

1. Anda dapat mengunduh salah satu dari contoh file manifes ini untuk membuat `clusterrole` dan `clusterrolebinding` atau a `role` dan`rolebinding`, masing-masing:

   Lihat Kubernetes sumber daya di semua ruang nama

   Peran `eks-connector-console-dashboard-full-access-clusterrole` cluster memberikan akses ke semua ruang nama dan sumber daya yang dapat divisualisasikan di konsol. Anda dapat mengubah nama`role`, `clusterrole` dan pengikatan yang sesuai sebelum menerapkannya ke cluster Anda. Gunakan perintah berikut untuk mengunduh file sampel.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/eks-connector-console-roles/eks-connector-console-dashboard-full-access-group.yaml
   ```

Lihat Kubernetes sumber daya di namespace tertentu

Namespace dalam file ini adalah`default`, jadi jika Anda ingin menentukan namespace yang berbeda, edit file sebelum menerapkannya ke cluster Anda. Gunakan perintah berikut untuk mengunduh file sampel.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/
eks-connector-console-roles/eks-connector-console-dashboard-restricted-access-
group.yaml
```

2. Edit akses penuh atau file YAMAL akses terbatas untuk mengganti referensi `%IAM_ARN%` dengan Nama Sumber Daya Amazon (ARN) kepala sekolah IAM Anda.
3. Terapkan akses penuh atau akses terbatas file YAML ke Kubernetes cluster Anda. Ganti nilai file YAMAL dengan milik Anda sendiri.

```
kubectl apply -f eks-connector-console-dashboard-full-access-group.yaml
```

Untuk melihat Kubernetes sumber daya di klaster yang terhubung, lihatLihat Kubernetes sumber daya. Data untuk beberapa jenis sumber daya pada tab Sumber Daya tidak tersedia untuk kluster yang terhubung.

Menderegistrasi sebuah cluster

Jika Anda selesai menggunakan cluster yang terhubung, Anda dapat membatalkan pendaftarannya. Setelah dideregistrasi, cluster tidak lagi terlihat di konsol Amazon EKS.

Anda harus memiliki izin berikut untuk memanggil API DeregisterCluster:

- `eks:DeregisterCluster`
- `ssm:DeleteActivation`
- `ssm:DeregisterManagedInstance`

Proses ini melibatkan dua langkah: Membatalkan pendaftaran cluster dengan Amazon EKS dan menghapus instalan agen eks-connector di cluster.

Untuk membatalkan pendaftaran cluster Kubernetes

AWS CLI

Prasyarat

- AWS CLIharus diinstal. Untuk menginstal atau memutakhirkannya, lihat Menginstal AWS CLI.
- Pastikan peran agen Amazon EKS Connector telah dibuat.

Deregister cluster yang terhubung.

```
aws eks deregister-cluster \
    --name my-cluster \
    --region region-code
```

AWS Management Console

1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
2. Pilih Klaster.
3. Pada halaman Clusters, pilih cluster yang terhubung dan pilih Deregister.
4. Konfirmasikan bahwa Anda ingin membatalkan pendaftaran cluster.

eksctl

Prasyarat

- `eksctl`versi `0.68` atau yang lebih baru harus diinstal. Untuk menginstal atau memutakhirkannya, lihatMemulai dengan Amazon EKS – `eksctl`.
- Pastikan peran agen Amazon EKS Connector telah dibuat.

Untuk membatalkan pendaftaran cluster Anda dengan **`eksctl`**

- Untuk konfigurasi Konektor, tentukan peran IAM agen Amazon EKS Connector Anda. Untuk informasi selengkapnya, lihat Peran IAM yang diperlukan untuk Konektor Amazon EKS.

```
eksctl deregister cluster --name my-cluster
```

Untuk membersihkan sumber daya di Kubernetes cluster Anda

Helm

- Jalankan perintah berikut untuk menghapus instalan agen.

```
helm -n eks-connector uninstall eks-connector
```

YAML manifest

1. Hapus file YAMAL Amazon EKS Connector dari Kubernetes cluster Anda.

```
kubectl delete -f eks-connector.yaml
```

2. Jika Anda membuat `clusterrole` atau `clusterrolebindings` untuk prinsipal IAM tambahan untuk mengakses klaster, hapus dari klaster Anda. Kubernetes

Memecahkan masalah di Amazon EKS Connector

Topik ini mencakup beberapa kesalahan umum yang mungkin Anda temui saat menggunakan Konektor Amazon EKS, termasuk petunjuk tentang cara mengatasinya dan solusinya.

Pemecahan masalah dasar

Bagian ini menjelaskan langkah-langkah untuk mendiagnosis masalah jika tidak jelas.

Periksa status Konektor Amazon EKS

Periksa status Konektor Amazon EKS.

```
kubectl get pods -n eks-connector
```

Periksa log Konektor Amazon EKS

Konektor Amazon EKS Pod terdiri dari tiga kontainer. Untuk mengambil log lengkap untuk semua kontainer ini sehingga Anda dapat memeriksanya, jalankan perintah berikut:

- `connector-init`

```
kubectl logs eks-connector-0 --container connector-init -n eks-connector
kubectl logs eks-connector-1 --container connector-init -n eks-connector
```

- `connector-proxy`

```
kubectl logs eks-connector-0 --container connector-proxy -n eks-connector
kubectl logs eks-connector-1 --container connector-proxy -n eks-connector
```

- `connector-agent`

```
kubectl exec eks-connector-0 --container connector-agent -n eks-connector -- cat /
var/log/amazon/ssm/amazon-ssm-agent.log
kubectl exec eks-connector-1 --container connector-agent -n eks-connector -- cat /
var/log/amazon/ssm/amazon-ssm-agent.log
```

Dapatkan nama cluster yang efektif

Cluster Amazon EKS diidentifikasi secara unik oleh `clusterName` dalam satu AWS akun dan. Wilayah AWS Jika Anda memiliki beberapa cluster yang terhubung di Amazon EKS, Anda dapat mengonfirmasi klaster Amazon EKS mana yang terdaftar untuk Kubernetes cluster saat ini. Untuk melakukan ini, masukkan yang berikut ini untuk mengetahui cluster saat ini. `clusterName`

```
kubectl exec eks-connector-0 --container connector-agent -n eks-connector \
  -- cat /var/log/amazon/ssm/amazon-ssm-agent.log | grep -m1 -oE "eks_c:[a-zA-Z0-9_-]+"
 | sed -E "s/^.*eks_c:([a-zA-Z0-9_-]+)_[a-zA-Z0-9]+.*$/\1/"
kubectl exec eks-connector-1 --container connector-agent -n eks-connector \
  -- cat /var/log/amazon/ssm/amazon-ssm-agent.log | grep -m1 -oE "eks_c:[a-zA-Z0-9_-]+"
 | sed -E "s/^.*eks_c:([a-zA-Z0-9_-]+)_[a-zA-Z0-9]+.*$/\1/"
```

Perintah lain-lain

Perintah berikut berguna untuk mengambil informasi yang Anda butuhkan untuk memecahkan masalah.

- Gunakan perintah berikut untuk mengumpulkan gambar yang digunakan oleh Pods di Amazon EKS Connector.

```
kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.containers[*].image}"
 | tr -s '[[:space:]]' '\n'
```

- Gunakan perintah berikut untuk menentukan nama node tempat Amazon EKS Connector berjalan.

```
kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.nodeName}" | tr -s
 '[[:space:]]' '\n'
```

- Jalankan perintah berikut untuk mendapatkan versi Kubernetes klien dan server Anda.

```
kubectl version
```

- Jalankan perintah berikut untuk mendapatkan informasi tentang node Anda.

```
kubectl get nodes -o wide --show-labels
```

Masalah helm: 403 Terlarang

Jika Anda menerima kesalahan berikut saat menjalankan perintah helm install:

```
Error: INSTALLATION FAILED: unexpected status from HEAD request to https://
public.ecr.aws/v2/eks-connector/eks-connector-chart/manifests/0.0.6: 403 Forbidden
```

Anda dapat menjalankan baris berikut untuk memperbaikinya:

```
docker logout public.ecr.aws
```

Kesalahan konsol: cluster macet dalam status Tertunda

Jika klaster macet dalam `Pending` status di konsol Amazon EKS setelah Anda mendaftarkannya, itu mungkin karena Konektor Amazon EKS AWS belum berhasil menghubungkan cluster. Untuk klaster terdaftar, `Pending` status berarti bahwa koneksi tidak berhasil dibuat. Untuk mengatasi masalah ini, pastikan Anda telah menerapkan manifes ke Kubernetes kluster target. Jika Anda menerapkannya ke cluster, tetapi cluster masih dalam `Pending` status, maka `eks-connector` statefulset mungkin tidak sehat. Untuk memecahkan masalah ini, lihat Konektor Amazon EKS Pods sedang crash looping di topik ini.

Kesalahan konsol: **`User “system:serviceaccount:eks-connector:eks-connector” can't impersonate resource “users” in API group “”`** di ruang lingkup cluster

Konektor Amazon EKS menggunakan peniruan identitas Kubernetes pengguna untuk bertindak atas nama kepala sekolah IAM dari. AWS Management Console Setiap prinsipal yang mengakses Kubernetes API dari akun AWS `eks-connector` layanan harus diberikan izin untuk menyamar sebagai Kubernetes pengguna yang sesuai dengan IAM ARN sebagai nama penggunanya. Kubernetes Dalam contoh berikut, IAM ARN dipetakan ke pengguna. Kubernetes

- Pengguna IAM *`john`* dari AWS akun *`111122223333`* dipetakan ke pengguna. Kubernetes Praktik terbaik IAM menyarankan agar Anda memberikan izin untuk peran, bukan pengguna.

  ```
  arn:aws:iam::111122223333:user/john
  ```

- Peran IAM *`admin`* dari AWS akun *`111122223333`* dipetakan ke pengguna: Kubernetes

  ```
  arn:aws:iam::111122223333:role/admin
  ```

  Hasilnya adalah peran IAM ARN, bukan AWS STS sesi ARN.

Untuk petunjuk tentang cara mengonfigurasi `ClusterRole` dan `ClusterRoleBinding` memberikan hak istimewa akun `eks-connector` layanan untuk meniru pengguna yang dipetakan, lihat. Memberikan akses ke prinsipal IAM untuk melihat Kubernetes sumber daya di cluster Pastikan bahwa dalam template, `%IAM_ARN%` diganti dengan IAM ARN dari kepala IAM AWS Management Console .

Kesalahan konsol: **`[...] is forbidden: User [...] cannot list resource “[...] in API group”`** pada lingkup cluster

Pertimbangkan masalah berikut. Konektor Amazon EKS telah berhasil menyamar sebagai prinsipal AWS Management Console IAM yang meminta di cluster target. Kubernetes Namun, kepala sekolah yang ditiru tidak memiliki izin RBAC untuk operasi API. Kubernetes

Untuk mengatasi masalah ini, ada dua metode untuk memberikan izin kepada pengguna tambahan. Jika sebelumnya Anda menginstal eks-konektor melalui bagan helm, Anda dapat dengan mudah memberikan akses kepada pengguna dengan menjalankan perintah berikut. Ganti `userARN1` dan

`userARN2` dengan daftar ARN dari peran IAM untuk memberikan akses untuk melihat sumber daya: Kubernetes

```
helm upgrade eks-connector oci://public.ecr.aws/eks-connector/eks-connector-chart \
    --reuse-values \
    --set 'authentication.allowedUserARNs={userARN1,userARN2}'
```

Atau, sebagai administrator cluster, berikan tingkat hak istimewa RBAC yang sesuai kepada pengguna individu. Kubernetes Untuk informasi selengkapnya dan contoh tambahan, lihat Memberikan akses ke prinsipal IAM untuk melihat Kubernetes sumber daya di cluster.

Kesalahan konsol: Amazon EKS tidak dapat berkomunikasi dengan server API Kubernetes cluster Anda. Cluster harus dalam keadaan AKTIF agar koneksi berhasil. Coba lagi dalam beberapa menit.

Jika layanan Amazon EKS tidak dapat berkomunikasi dengan konektor Amazon EKS di cluster target, itu mungkin karena salah satu alasan berikut:

- Konektor Amazon EKS di cluster target tidak sehat.
- Konektivitas yang buruk atau koneksi yang terputus antara cluster target dan. Wilayah AWS

Untuk mengatasi masalah ini, periksa log Amazon EKS Connector. Jika Anda tidak melihat kesalahan untuk Konektor Amazon EKS, coba lagi koneksi setelah beberapa menit. Jika Anda secara teratur mengalami latensi tinggi atau konektivitas intermiten untuk kluster target, pertimbangkan untuk mendaftarkan ulang cluster ke cluster Wilayah AWS yang terletak lebih dekat dengan Anda.

Konektor Amazon EKS Pods sedang crash looping

Ada banyak alasan yang dapat menyebabkan konektor Pod Amazon EKS memasuki `CrashLoopBackOff` status. Masalah ini kemungkinan melibatkan `connector-init` wadah. Periksa status konektor Amazon EKSPod.

```
kubectl get pods -n eks-connector
```

Contoh output adalah sebagai berikut.

```
NAME              READY   STATUS                  RESTARTS   AGE
```

```
eks-connector-0   0/2     Init:CrashLoopBackOff   1          7s
```

Jika output Anda mirip dengan output sebelumnya, lihat Periksa log Konektor Amazon EKS untuk memecahkan masalah.

Failed to initiate eks-connector: InvalidActivation

Saat Anda memulai Konektor Amazon EKS untuk pertama kalinya, ia mendaftarkan `activationId` dan `activationCode` dengan Amazon Web Services. Pendaftaran mungkin gagal, yang dapat menyebabkan `connector-init` penampung mogok dengan kesalahan yang mirip dengan kesalahan berikut.

```
F1116 20:30:47.261469       1 init.go:43] failed to initiate eks-connector:
 InvalidActivation:
```

Untuk memecahkan masalah ini, pertimbangkan penyebab berikut dan perbaikan yang disarankan:

- Pendaftaran mungkin gagal karena `activationId` dan `activationCode` tidak ada dalam file manifes Anda. Jika ini masalahnya, pastikan bahwa itu adalah nilai yang benar yang dikembalikan dari operasi `RegisterCluster` API, dan itu `activationCode` ada di file manifes. `activationCode`Itu ditambahkan ke Kubernetes rahasia, jadi harus `base64` dikodekan. Untuk informasi selengkapnya, lihat Langkah 1: Mendaftarkan cluster.
- Pendaftaran mungkin gagal karena aktivasi Anda kedaluwarsa. Ini karena, untuk alasan keamanan, Anda harus mengaktifkan Konektor Amazon EKS dalam waktu tiga hari setelah mendaftarkan cluster. Untuk mengatasi masalah ini, pastikan manifes Amazon EKS Connector diterapkan ke Kubernetes cluster target sebelum tanggal dan waktu kedaluwarsa. Untuk mengonfirmasi tanggal kedaluwarsa aktivasi Anda, hubungi operasi `DescribeCluster` API.

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

  Dalam contoh respons berikut, tanggal kedaluwarsa dan waktu dicatat sebagai`2021-11-12T22:28:51.101000-08:00`.

  ```
  {
      "cluster": {
          "name": "my-cluster",
          "arn": "arn:aws:eks:region:111122223333:cluster/my-cluster",
          "createdAt": "2021-11-09T22:28:51.449000-08:00",
  ```

```
        "status": "FAILED",
        "tags": {
        },
        "connectorConfig": {
            "activationId": "00000000-0000-0000-0000-000000000000",
            "activationExpiry": "2021-11-12T22:28:51.101000-08:00",
            "provider": "OTHER",
            "roleArn": "arn:aws:iam::111122223333:role/my-connector-role"
        }
    }
}
```

Jika `activationExpiry` lulus, deregister cluster dan daftarkan lagi. Lakukan ini menghasilkan aktivasi baru.

Node cluster tidak memiliki konektivitas keluar

Agar berfungsi dengan baik, Konektor Amazon EKS memerlukan konektivitas keluar ke beberapa titik AWS akhir. Anda tidak dapat menghubungkan cluster pribadi tanpa konektivitas keluar ke target Wilayah AWS. Untuk mengatasi masalah ini, Anda harus menambahkan konektivitas keluar yang diperlukan. Untuk informasi tentang persyaratan konektor, lihat Pertimbangan Konektor Amazon EKS.

Konektor Amazon EKS Pods sedang dalam **`ImagePullBackOff`** keadaan

Jika Anda menjalankan `get pods` perintah dan Pods berada dalam `ImagePullBackOff` status, mereka tidak dapat berfungsi dengan baik. Jika Konektor Pods Amazon EKS dalam `ImagePullBackOff` keadaan, mereka tidak dapat berfungsi dengan baik. Periksa status Konektor Amazon EKS AndaPods.

```
kubectl get pods -n eks-connector
```

Contoh output adalah sebagai berikut.

```
NAME              READY   STATUS                  RESTARTS   AGE
eks-connector-0   0/2     Init:ImagePullBackOff   0          4s
```

File manifes Konektor Amazon EKS default mereferensikan gambar dari Galeri Publik Amazon ECR. Ada kemungkinan bahwa Kubernetes kluster target tidak dapat menarik gambar dari Galeri Publik

Amazon ECR. Atasi masalah tarik gambar Galeri Publik Amazon ECR, atau pertimbangkan untuk mencerminkan gambar di registri penampung pribadi pilihan Anda.

Pertanyaan umum

T: Bagaimana cara kerja teknologi yang mendasari di balik Konektor Amazon EKS?

J: Konektor Amazon EKS didasarkan pada agenAWS Systems Manager (Systems Manager). Konektor Amazon EKS berjalan sebagai`StatefulSet` padaKubernetes klaster Anda. Ini membuat koneksi dan memproksi komunikasi antara server API klaster Anda dan Amazon Web Services. Hal ini dilakukan untuk menampilkan data klaster di konsol Amazon EKS sampai Anda memutuskan klasterAWS. Agen Systems Manager adalah proyek sumber terbuka. Untuk informasi lebih lanjut tentang proyek ini, lihat halamanGitHub proyek.

T: Saya memilikiKubernetes klaster lokal yang ingin saya sambungkan. Apakah saya perlu membuka port firewall untuk menghubungkannya?

J: Tidak, Anda tidak perlu membuka port firewall apa pun. KubernetesCluster hanya membutuhkan koneksi keluar keWilayah AWS. AWSlayanan tidak pernah mengakses sumber daya di jaringan on-premise Anda. Konektor Amazon EKS berjalan di klaster Anda dan memulai koneksi keAWS. Ketika pendaftaran klaster selesai,AWS hanya mengeluarkan perintah ke Konektor Amazon EKS setelah Anda memulai tindakan dari konsol Amazon EKS yang memerlukan informasi dari serverKubernetes API di klaster Anda.

T: Data apa yang dikirim dari klaster saya keAWS Konektor Amazon EKS?

J: Konektor Amazon EKS mengirimkan informasi teknis yang diperlukan agar klaster Anda didaftarkanAWS. Ini juga mengirimkan metadata klaster dan beban kerja untuk fitur konsol Amazon EKS yang diminta pelanggan. Konektor Amazon EKS hanya mengumpulkan atau mengirimkan data ini jika Anda memulai tindakan dari konsol Amazon EKS atau API Amazon EKS yang mengharuskan data dikirimAWS. Selain nomorKubernetes versi,AWS tidak menyimpan data apa pun secara default. Ini menyimpan data hanya jika Anda mengotorisasi untuk.

T: Dapatkah saya menghubungkan klaster di luar sebuahWilayah AWS?

J: Ya, Anda dapat menghubungkan klaster dari lokasi mana pun ke Amazon EKS. Selain itu, layanan Amazon EKS Anda dapat ditemukan di komersialAWS publik apa punWilayah AWS. Ini bekerja dengan koneksi jaringan yang valid dari klaster Anda ke targetWilayah AWS. Kami menyarankan Anda memilihWilayah AWS yang paling dekat dengan lokasi klaster Anda untuk optimasi kinerja

UI. Misalnya, jika Anda memiliki klaster yang berjalan di Tokyo, hubungkan klaster Anda ke Tokyo (yaitu,`ap-northeast-1`Wilayah AWS) untuk latensi rendah.Wilayah AWS Anda dapat menghubungkan klaster dari lokasi mana pun ke Amazon EKS di salah satu iklan publikWilayah AWS, kecuali Tiongkok atau GovCloud Wilayah AWS.

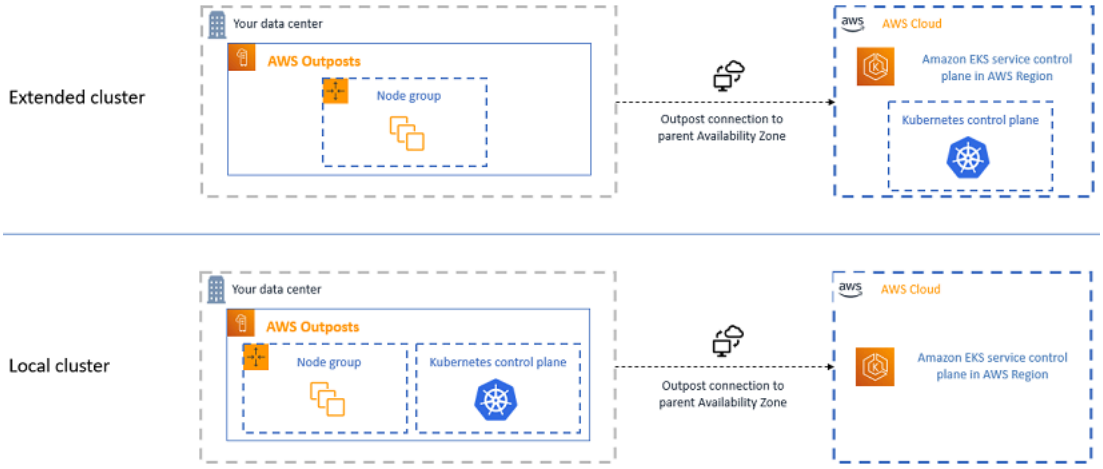
Amazon EKS pada AWS Outposts

Anda dapat menggunakan Amazon EKS untuk menjalankan Kubernetes aplikasi lokal. AWS Outposts Anda dapat menerapkan Amazon EKS di Outposts dengan cara berikut:

- Cluster yang diperluas — Jalankan bidang Kubernetes kontrol di sebuah Wilayah AWS dan node di Outpost Anda.
- Cluster lokal — Jalankan bidang Kubernetes kontrol dan node di Outpost Anda.

Untuk kedua opsi penyebaran, bidang Kubernetes kontrol dikelola sepenuhnya olehAWS. Anda dapat menggunakan API, alat, dan konsol Amazon EKS yang sama yang Anda gunakan di cloud untuk membuat dan menjalankan Amazon EKS di Outposts.

Diagram berikut menunjukkan opsi penyebaran ini.



Kapan menggunakan setiap opsi penerapan

Cluster lokal dan diperluas adalah opsi penyebaran tujuan umum dan dapat digunakan untuk berbagai aplikasi.

Dengan cluster lokal, Anda dapat menjalankan seluruh cluster Amazon EKS secara lokal di Outposts. Opsi ini dapat mengurangi risiko downtime aplikasi yang mungkin diakibatkan oleh pemutusan jaringan sementara ke cloud. Pemutusan jaringan ini dapat disebabkan oleh pemotongan serat atau peristiwa cuaca. Karena seluruh kluster Amazon EKS berjalan secara lokal di Outposts, aplikasi tetap tersedia. Anda dapat melakukan operasi cluster selama jaringan terputus ke cloud. Untuk informasi

selengkapnya, lihat Mempersiapkan pemutusan jaringan. Jika Anda khawatir tentang kualitas koneksi jaringan dari Outposts Anda ke induk Wilayah AWS dan memerlukan ketersediaan tinggi melalui pemutusan jaringan, gunakan opsi penyebaran cluster lokal.

Dengan cluster yang diperluas, Anda dapat menghemat kapasitas di Pos Luar Anda karena bidang Kubernetes kontrol berjalan di induk. Wilayah AWS Opsi ini cocok jika Anda dapat berinvestasi dalam konektivitas jaringan yang andal dan redundan dari Outpost Anda ke. Wilayah AWS Kualitas koneksi jaringan sangat penting untuk opsi ini. Cara Kubernetes menangani pemutusan jaringan antara bidang Kubernetes kontrol dan node dapat menyebabkan downtime aplikasi. Untuk informasi selengkapnya tentang perilakuKubernetes, lihat Penjadwalan, Preemption, dan Penggusuran dalam dokumentasi. Kubernetes

Membandingkan opsi penerapan

Tabel berikut membandingkan perbedaan antara dua opsi.

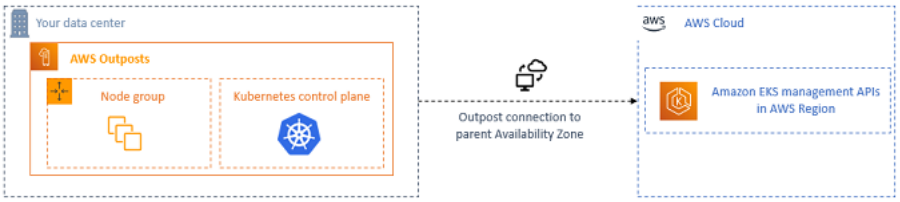
Fitur	Cluster yang diperluas	Cluster lokal
Kuberneteslokasi pesawat kontrol	Wilayah AWS	Pos terdepan
Kubernetesakun pesawat kontrol	Akun AWS	Akun Anda
Ketersediaan Wilayah	Lihat Titik akhir Layanan	AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (California N.), AS Barat (Oregon), Asia Pasifik (Seoul), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Kanada (Tengah), Eropa (Frankfurt), Eropa (Irlandia), Eropa (London), Timur Tengah (Bahrain), dan Amerika Selatan (São Paulo)
Kubernetes versi minor	Versi Amazon EKS yang didukung.	Versi Amazon EKS yang didukung.

Fitur	Cluster yang diperluas	Cluster lokal
Versi platform	Lihat Amazon EKS versi platform	Lihat Amazon EKS versi platform cluster lokal
Faktor bentuk pos terdepan	Rak pos terdepan	Rak pos terdepan
Antarmuka pengguna	AWS Management Console,AWS CLI, Amazon EKS API,, `eksctl`AWS CloudFormation, dan Terraform	AWS Management Console,AWS CLI, Amazon EKS API,, `eksctl`AWS CloudFormation, dan Terraform
Kebijakan terkelola	AmazonEks dan ClusterPolicy AmazonEks ServiceRolePolicy	AmazonEks dan LocalOutpostClusterPolicy AmazonEks LocalOutpostServiceRolePolicy
Cluster VPC dan subnet	Lihat Persyaratan dan pertimbangan Amazon EKS VPC dan subnet	Lihat Persyaratan dan pertimbangan VPC klaster lokal dan subnet Amazon EKS
Akses titik akhir klaster	Publik atau swasta atau keduanya	Hanya pribadi
KubernetesOtentikasi server API	AWS Identity and Access Management(IAM) dan OIDC	IAM dan sertifikat `x.509`
Jenis simpul	Hanya dikelola sendiri	Hanya dikelola sendiri
Jenis komputasi node	Amazon EC2 sesuai permintaan	Amazon EC2 sesuai permintaan
Jenis penyimpanan simpul	Amazon EBS gp2 dan NVMe SSD lokal	Amazon EBS gp2 dan NVMe SSD lokal
AMI yang dioptimalkan Amazon EKS	Amazon Linux, Windows, dan Bottlerocket	Hanya Amazon Linux
Versi IP	`IPv4`hanya	`IPv4`hanya

Fitur	Cluster yang diperluas	Cluster lokal
Pengaya	Pengaya Amazon EKS atau add-on yang dikelola sendiri	Hanya add-on yang dikelola sendiri
Antarmuka Jaringan Kontainer Default	Amazon VPC CNI plugin for Kubernetes	Amazon VPC CNI plugin for Kubernetes
Log pesawat kontrol Kubernetes	CloudWatch Log Amazon	CloudWatch Log Amazon
Penyeimbangan beban	Gunakan AWS Load Balancer Controlleruntuk menyediakan Application Load Balancers saja (tidak ada Network Load Balancers)	Gunakan AWS Load Balancer Controlleruntuk menyediakan Application Load Balancers saja (tidak ada Network Load Balancers)
Rahasia enkripsi amplop	Lihat Mengaktifkan enkripsi rahasia pada cluster yang ada	Tidak didukung
IAM role untuk akun layanan	Lihat IAM role untuk akun layanan	Tidak didukung
Pemecahan Masalah	Lihat Pemecahan masalah Amazon EKS	Lihat Memecahkan masalah kluster lokal untuk Amazon EKS di AWS Outposts

Cluster lokal untuk Amazon EKS di AWS Outposts

Anda dapat menggunakan kluster lokal untuk menjalankan seluruh kluster Amazon EKS secara lokal. AWS Outposts Ini membantu mengurangi risiko downtime aplikasi yang mungkin diakibatkan oleh pemutusan jaringan sementara ke cloud. Pemutusan ini dapat disebabkan oleh pemotongan serat atau peristiwa cuaca. Karena seluruh Kubernetes cluster berjalan secara lokal di Outposts, aplikasi tetap tersedia. Anda dapat melakukan operasi cluster selama pemutusan jaringan ke cloud. Untuk informasi selengkapnya, lihat Mempersiapkan pemutusan jaringan. Diagram berikut menunjukkan penyebaran cluster lokal.



Cluster lokal umumnya tersedia untuk digunakan dengan rak Outposts.

Wilayah AWS yang Didukung

Anda dapat membuat cluster lokal sebagai berikutWilayah AWS: AS Timur (Ohio), AS Timur (Virginia N.), AS Barat (California N.), AS Barat (Oregon), Asia Pasifik (Seoul), Asia Pasifik (Singapura), Asia Pasifik (Sydney), Asia Pasifik (Tokyo), Kanada (Tengah), Eropa (Frankfurt), Eropa (Irlandia), Eropa (London), Timur Tengah (Bahrain), dan Selatan Amerika (Sao Paulo). Untuk informasi mendetail tentang fitur yang didukung, lihatMembandingkan opsi penerapan.

Topik

- Membuat cluster lokal di Outpost
- Amazon EKS versi platform cluster lokal
- Persyaratan dan pertimbangan VPC klaster lokal dan subnet Amazon EKS
- Mempersiapkan pemutusan jaringan
- Pertimbangan kapasitas pertimbangan kapasitas pertimbangan kapasitas
- Memecahkan masalah kluster lokal untuk Amazon EKS di AWS Outposts

Membuat cluster lokal di Outpost

Topik ini memberikan gambaran umum tentang apa yang harus dipertimbangkan saat menjalankan cluster lokal di Outpost. Topik ini juga memberikan instruksi tentang cara menyebarkan cluster lokal di Outpost.

Pertimbangan

> **Important**
>
> - Pertimbangan ini tidak direplikasi dalam dokumentasi Amazon EKS terkait. Jika topik dokumentasi Amazon EKS lainnya bertentangan dengan pertimbangan di sini, ikuti pertimbangannya di sini.

- Pertimbangan ini dapat berubah dan mungkin sering berubah. Jadi, kami sarankan Anda meninjau topik ini secara teratur.
- Banyak pertimbangan yang berbeda dari pertimbangan untuk membuat cluster di. AWS Cloud

- Cluster lokal hanya mendukung rak Outpost. Sebuah cluster lokal tunggal dapat berjalan di beberapa rak Outpost fisik yang terdiri dari satu pos logis. Satu cluster lokal tidak dapat berjalan di beberapa Outposts logis. Setiap Outpost logis memiliki ARN Outpost tunggal.
- Cluster lokal menjalankan dan mengelola bidang Kubernetes kontrol di akun Anda di Outpost. Anda tidak dapat menjalankan beban kerja pada instance bidang Kubernetes kontrol atau memodifikasi komponen bidang Kubernetes kontrol. Node ini dikelola oleh layanan Amazon EKS. Perubahan pada bidang Kubernetes kontrol tidak bertahan melalui tindakan manajemen Amazon EKS otomatis, seperti menambal.
- Cluster lokal mendukung add-on yang dikelola sendiri dan grup node Amazon Linux yang dikelola sendiri. The Amazon VPC CNI plugin for Kubernetes, `kube-proxy`, dan CoreDNSadd-on secara otomatis diinstal pada cluster lokal.
- Cluster lokal memerlukan penggunaan Amazon EBS di Outposts. Pos Luar Anda harus memiliki Amazon EBS yang tersedia untuk penyimpanan pesawat Kubernetes kontrol.
- Cluster lokal menggunakan Amazon EBS di Outposts. Pos Luar Anda harus memiliki Amazon EBS yang tersedia untuk penyimpanan pesawat Kubernetes kontrol. Outposts hanya mendukung volume Amazon EBSgp2.
- Amazon EBS Kubernetes `PersistentVolumes` didukung menggunakan driver Amazon EBS CSI.

Prasyarat

- Keakraban dengan opsi Pertimbangan kapasitas pertimbangan kapasitas pertimbangan kapasitaspenyebaran Outposts,, dan. Persyaratan dan pertimbangan VPC klaster lokal dan subnet Amazon EKS
- Pos terdepan yang ada. Untuk informasi lebih lanjut, lihat Apa yang dimaksud dengan AWS Outposts.
- Alat baris `kubectl` perintah diinstal pada komputer Anda atau AWS CloudShell. Versi dapat sama dengan atau hingga satu versi minor lebih awal atau lebih lambat dari Kubernetes versi cluster Anda. Misalnya, jika versi cluster Anda`1.28`, Anda dapat menggunakan `kubectl`

versi`1.27`,`1.28`, atau `1.29` dengan itu. Untuk menginstal atau memutakhirkan `kubectl`, lihat Menginstal atau memperbarui `kubectl`.

- Versi `2.12.3` atau yang lebih baru atau versi `1.27.160` atau yang lebih baru dari AWS Command Line Interface (AWS CLI) diinstal dan dikonfigurasi pada perangkat Anda atau AWS CloudShell. Untuk memeriksa versi Anda saat ini, gunakan**`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. Package manager seperti `yum`,`apt-get`, atau Homebrew untuk macOS sering beberapa versi di belakang versi terbaru AWS CLI. Untuk menginstal versi terbaru, lihat Menginstal, memperbarui, dan menghapus konfigurasi AWS CLI dan Cepat dengan aws configure di Panduan AWS Command Line Interface Pengguna. AWS CLI Versi yang diinstal AWS CloudShell mungkin juga beberapa versi di belakang versi terbaru. Untuk memperbaruinya, lihat Menginstal AWS CLI ke direktori home Anda di Panduan AWS CloudShell Pengguna.
- Prinsipal IAM (pengguna atau peran) dengan izin `create` dan kluster `describe` Amazon EKS. Lihat informasi yang lebih lengkap di Buat Kubernetes cluster lokal di Outpost dan Buat daftar atau deskripsikan semua klaster.

Saat kluster Amazon EKS lokal dibuat, prinsipal IAM yang membuat cluster ditambahkan secara permanen. Kepala sekolah secara khusus ditambahkan ke tabel otorisasi Kubernetes RBAC sebagai administrator. Entitas ini memiliki `system:masters` izin. Identitas entitas ini tidak terlihat dalam konfigurasi klaster Anda. Jadi, penting untuk mencatat entitas yang membuat cluster dan pastikan Anda tidak pernah menghapusnya. Awalnya, hanya prinsipal yang membuat server dapat melakukan panggilan ke server Kubernetes API menggunakan`kubectl`. Jika Anda menggunakan konsol untuk membuat klaster, pastikan kredenal IAM yang sama ada di rantai kredensi AWS SDK saat Anda menjalankan `kubectl` perintah di klaster. Setelah cluster Anda dibuat, Anda dapat memberikan prinsipal IAM lainnya akses ke cluster Anda.

Untuk membuat kluster lokal Amazon EKS lokal

Anda dapat membuat cluster lokal dengan`eksctl`,, Amazon EKS API AWS CLI, AWS SDK, AWS CloudFormationatau Terraform. AWS Management Console

1. Buat cluster lokal.

   eksctl

   Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Untuk membuat cluster Anda dengan **`eksctl`**

1. Salin konten yang mengikuti ke perangkat Anda. Ganti nilai-nilai berikut dan kemudian jalankan perintah yang dimodifikasi untuk membuat `outpost-control-plane.yaml` file:
   - Ganti *`region-code`* dengan Wilayah AWS yang didukung tempat Anda ingin membuat cluster Anda.
   - Ganti *`my-cluster`* dengan nama untuk cluster Anda. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter. Nama harus unik di dalam Wilayah AWS dan Akun AWS tempat Anda membuat cluster.
   - Ganti *`vpc-ExampleID1`* dan *`subnet-ExampleID1`* dengan ID VPC dan subnet Anda yang ada. VPC dan subnet harus memenuhi persyaratan di. Persyaratan dan pertimbangan VPC klaster lokal dan subnet Amazon EKS
   - Ganti *`uniqueid`* dengan ID Outpost Anda.
   - Ganti *`m5.large`* dengan jenis instance yang tersedia di Outpost Anda. Sebelum memilih jenis instance, lihatPertimbangan kapasitas pertimbangan kapasitas pertimbangan kapasitas. Tiga instance pesawat kontrol dikerahkan. Anda tidak dapat mengubah nomor ini.

```
cat >outpost-control-plane.yaml <<EOF
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: "1.24"

vpc:
  clusterEndpoints:
    privateAccess: true
  id: "vpc-vpc-ExampleID1"
  subnets:
```

```
    private:
      outpost-subnet-1:
        id: "subnet-subnet-ExampleID1"

outpost:
  controlPlaneOutpostARN: arn:aws:outposts:region-code:111122223333:outpost/
op-uniqueid
  controlPlaneInstanceType: m5.large
EOF
```

Untuk daftar lengkap semua opsi dan default yang tersedia, lihat Skema file Support AWS Outposts dan Config dalam dokumentasi. `eksctl`

2. Buat cluster menggunakan file konfigurasi yang Anda buat pada langkah sebelumnya. `eksctl`membuat VPC dan satu subnet di Outpost Anda untuk menyebarkan cluster.

```
eksctl create cluster -f outpost-control-plane.yaml
```

Penyediaan klaster memerlukan waktu beberapa menit. Saat cluster sedang dibuat, beberapa baris output muncul. Baris terakhir output mirip dengan baris contoh berikut.

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

> **Tip**
>
> Untuk melihat sebagian besar opsi yang dapat Anda tentukan saat membuat cluster`eksctl`, gunakan **`eksctl create cluster --help`** perintah. Untuk melihat semua opsi yang tersedia, Anda dapat menggunakan `config` file. Untuk informasi selengkapnya, lihat Menggunakan file config dan skema file config di dokumentasi `eksctl`. Anda dapat menemukan contoh file konfigurasi diGitHub.

`Eksctl`secara otomatis membuat entri akses untuk prinsipal IAM (pengguna atau peran) yang membuat cluster dan memberikan izin administrator utama IAM ke Kubernetes objek di cluster. Jika Anda tidak ingin pembuat klaster memiliki akses administrator ke Kubernetes objek di cluster, tambahkan teks berikut ke file konfigurasi sebelumnya: **`bootstrapClusterCreatorAdminPermissions: false`** (pada tingkat yang sama dengan`metadata`,`vpc`, dan`outpost`). Jika Anda menambahkan opsi, maka setelah

pembuatan cluster, Anda perlu membuat entri akses untuk setidaknya satu prinsipal IAM, atau tidak ada prinsipal IAM yang memiliki akses ke Kubernetes objek di cluster.

AWS Management Console

Prasyarat

VPC dan subnet yang ada yang memenuhi persyaratan Amazon EKS. Untuk informasi selengkapnya, lihat Persyaratan dan pertimbangan VPC klaster lokal dan subnet Amazon EKS.

Untuk membuat cluster Anda dengan AWS Management Console

1. Jika Anda sudah memiliki peran IAM cluster lokal, atau Anda akan membuat cluster Anda dengan`eksctl`, maka Anda dapat melewati langkah ini. Secara default, `eksctl` buat peran untuk Anda.

   a. Jalankan perintah berikut untuk membuat file JSON kebijakan kepercayaan IAM.

   ```
   cat >eks-local-cluster-role-trust-policy.json <<EOF
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
           "Service": "ec2.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
       }
     ]
   }
   EOF
   ```

   b. Buat peran IAM cluster Amazon EKS. Untuk membuat peran IAM, prinsipal IAM yang membuat peran harus diberi `iam:CreateRole` tindakan (izin).

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

   c. Lampirkan kebijakan terkelola Amazon EKS yang diberi nama `AmazonEKSLocalOutpostClusterPolicy`ke peran tersebut. Untuk melampirkan kebijakan IAM ke kepala sekolah IAM, prinsipal yang melampirkan kebijakan harus

diberikan salah satu tindakan IAM berikut (izin): atau. `iam:AttachUserPolicy` `iam:AttachRolePolicy`

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSLocalOutpostClusterPolicy --role-name myAmazonEKSLocalClusterRole
```

2. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
3. Di bagian atas layar konsol, pastikan Anda telah memilih yang didukung Wilayah AWS.
4. Pilih Add cluster dan kemudian pilih Create.
5. Pada halaman Configure cluster, masukkan atau pilih nilai untuk bidang berikut:
   - Kuberneteskontrol lokasi pesawat — Pilih AWS Outposts.
   - Outpost ID - Pilih ID Outpost tempat Anda ingin membuat pesawat kontrol Anda.
   - Jenis instans - Pilih jenis instans. Hanya jenis instans yang tersedia di Outpost Anda yang ditampilkan. Dalam daftar dropdown, setiap jenis instance menjelaskan berapa banyak node yang direkomendasikan untuk jenis instance. Sebelum memilih jenis instance, lihatPertimbangan kapasitas pertimbangan kapasitas pertimbangan kapasitas. Semua replika dikerahkan menggunakan jenis instance yang sama. Anda tidak dapat mengubah jenis instance setelah cluster Anda dibuat. Tiga instance pesawat kontrol dikerahkan. Anda tidak dapat mengubah nomor ini.
   - Nama — Nama untuk klaster Anda. Itu harus unik dalam diri Anda Akun AWS. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter. Nama harus unik di dalam Wilayah AWS dan Akun AWS tempat Anda membuat cluster.
   - Kubernetesversi — Pilih Kubernetes versi yang ingin Anda gunakan untuk cluster Anda. Sebaiknya pilih versi terbaru, kecuali jika Anda perlu menggunakan versi sebelumnya.
   - Peran layanan klaster — Pilih peran IAM klaster Amazon EKS yang Anda buat pada langkah sebelumnya untuk memungkinkan bidang Kubernetes kontrol mengelola AWS sumber daya.
   - Kubernetesakses administrator klaster - Jika Anda ingin prinsipal IAM (peran atau pengguna) yang membuat klaster memiliki akses administrator ke Kubernetes objek di cluster, terima default (izinkan). Amazon EKS membuat entri akses untuk prinsipal IAM dan memberikan izin administrator klaster ke entri akses. Untuk informasi selengkapnya tentang entri akses, lihatKelola entri akses.

Jika Anda menginginkan prinsipal IAM yang berbeda dari prinsipal yang membuat cluster untuk memiliki akses administrator ke objek Kubernetes cluster, pilih opsi disallow. Setelah pembuatan cluster, setiap prinsipal IAM yang memiliki izin IAM untuk membuat entri akses dapat menambahkan entri akses untuk setiap prinsipal IAM yang memerlukan akses ke objek cluster. Kubernetes Untuk informasi selengkapnya tentang izin IAM yang diperlukan, lihat Tindakan yang ditentukan oleh Amazon Elastic Kubernetes Service di Referensi Otorisasi Layanan. Jika Anda memilih opsi larangan dan tidak membuat entri akses apa pun, maka tidak ada prinsipal IAM yang akan memiliki akses ke objek di cluster. Kubernetes

- Tanda — (Opsional) Tambahkan tanda apapun ke klaster Anda. Untuk informasi selengkapnya, lihat Menandai sumber daya Amazon EKS Anda.

Setelah selesai dengan halaman ini, pilih Berikutnya.

6. Pada halaman Tentukan jaringan, pilih nilai untuk kolom berikut:
   - VPC — Pilih VPC yang ada. VPC harus memiliki cukup banyak alamat IP yang tersedia untuk cluster, node apa pun, dan Kubernetes sumber daya lain yang ingin Anda buat. VPC Anda harus memenuhi persyaratan di. Persyaratan dan pertimbangan VPC
   - Subnet — Secara default, semua subnet yang tersedia di VPC yang ditentukan di bidang sebelumnya telah dipilih sebelumnya. Subnet yang Anda pilih harus memenuhi persyaratan diPersyaratan dan pertimbangan subnet.

     Grup keamanan — (Opsional) Tentukan satu atau beberapa grup keamanan yang ingin Anda kaitkan Amazon EKS ke antarmuka jaringan yang dibuatnya. Amazon EKS secara otomatis membuat grup keamanan yang memungkinkan komunikasi antara cluster dan VPC Anda. Amazon EKS mengaitkan grup keamanan ini, dan apa pun yang Anda pilih, ke antarmuka jaringan yang dibuatnya. Untuk informasi selengkapnya tentang grup keamanan klaster yang dibuat Amazon EKS, lihatPersyaratan dan pertimbangan grup keamanan Amazon EKS. Anda dapat mengubah aturan di grup keamanan klaster yang dibuat Amazon EKS. Jika Anda memilih untuk menambahkan grup keamanan Anda sendiri, Anda tidak dapat mengubah grup yang Anda pilih setelah pembuatan klaster. Agar host lokal dapat berkomunikasi dengan titik akhir klaster, Anda harus mengizinkan lalu lintas masuk dari grup keamanan klaster. Untuk cluster yang tidak memiliki koneksi internet ingress dan egress (juga dikenal sebagai cluster pribadi), Anda harus melakukan salah satu hal berikut:

- Tambahkan grup keamanan yang terkait dengan titik akhir VPC yang diperlukan. Untuk informasi selengkapnya tentang titik akhir yang diperlukan, lihat titik akhir VPC antarmuka di. Akses subnet ke Layanan AWS
- Ubah grup keamanan yang dibuat Amazon EKS untuk memungkinkan lalu lintas dari grup keamanan yang terkait dengan titik akhir VPC.

Setelah selesai dengan halaman ini, pilih Berikutnya.

7. Pada halaman Konfigurasi observabilitas, Anda dapat secara opsional memilih opsi pencatatan bidang Metrik dan Kontrol mana yang ingin Anda aktifkan. Secara default, setiap jenis log dimatikan.
   - Untuk informasi selengkapnya tentang opsi Prometheus metrik, lihatAktifkan Prometheus metrik saat membuat klaster.
   - Untuk informasi selengkapnya tentang opsi Pencatatan bidang kontrol, lihatPencatatan bidang kendali Amazon EKS.

   Setelah selesai dengan halaman ini, pilih Berikutnya.
8. Pada halaman Tinjau dan buat, tinjau informasi yang Anda masukkan atau pilih pada halaman sebelumnya. Jika Anda perlu melakukan perubahan, pilih Edit. Saat Anda puas, pilih Buat. Bidang Status menunjukkan CREATING saat cluster disediakan.

   Penyediaan klaster memerlukan waktu beberapa menit.

2. Setelah klaster dibuat, Anda dapat melihat instans bidang kontrol Amazon EC2 yang dibuat.

```
aws ec2 describe-instances --query 'Reservations[*].Instances[*].{Name:Tags[?
Key==`Name`]|[0].Value}' | grep my-cluster-control-plane
```

Contoh output adalah sebagai berikut.

```
"Name": "my-cluster-control-plane-id1"
"Name": "my-cluster-control-plane-id2"
"Name": "my-cluster-control-plane-id3"
```

Setiap instance dicemari `node-role.eks-local.amazonaws.com/control-plane` sehingga tidak ada beban kerja yang dijadwalkan pada instance bidang kontrol. Untuk informasi selengkapnya tentang noda, lihat Taints and Tolerations dalam dokumentasi. Kubernetes Amazon EKS terus memantau keadaan cluster lokal. Kami melakukan tindakan manajemen otomatis, seperti patch keamanan dan memperbaiki instans yang tidak sehat. Ketika cluster lokal

terputus dari cloud, kami menyelesaikan tindakan untuk memastikan bahwa klaster diperbaiki ke keadaan sehat setelah tersambung kembali.

3. Jika Anda membuat cluster Anda menggunakan`eksctl`, maka Anda dapat melewati langkah ini. `eksctl`Selesaikan langkah ini untuk Anda. Aktifkan `kubectl` untuk berkomunikasi dengan cluster Anda dengan menambahkan konteks baru ke `kubectl config` file. Untuk petunjuk tentang cara membuat dan memperbarui file, lihatMembuat atau memperbarui `kubeconfig` file untuk klaster Amazon EKS.

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

   Contoh output adalah sebagai berikut.

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

4. Untuk terhubung ke server Kubernetes API cluster lokal Anda, dapatkan akses ke gateway lokal untuk subnet, atau sambungkan dari dalam VPC. Untuk informasi selengkapnya tentang menghubungkan rak Outpost ke jaringan lokal, lihat Cara kerja gateway lokal untuk rak di Panduan Pengguna. AWS Outposts Jika Anda menggunakan Direct VPC Routing dan subnet Outpost memiliki rute ke gateway lokal Anda, alamat IP pribadi dari instance pesawat Kubernetes kontrol secara otomatis disiarkan melalui jaringan lokal Anda. Titik akhir server Kubernetes API cluster lokal di-host di Amazon Route 53 (Route 53). Endpoint layanan API dapat diselesaikan oleh server DNS publik ke alamat IP pribadi server Kubernetes API.

   Instans bidang Kubernetes kontrol cluster lokal dikonfigurasi dengan antarmuka jaringan elastis statis dengan alamat IP pribadi tetap yang tidak berubah sepanjang siklus hidup cluster. Mesin yang berinteraksi dengan server Kubernetes API mungkin tidak memiliki konektivitas ke Route 53 selama pemutusan jaringan. Jika ini masalahnya, kami sarankan untuk mengonfigurasi `/etc/hosts` dengan alamat IP pribadi statis untuk operasi lanjutan. Kami juga merekomendasikan untuk menyiapkan server DNS lokal dan menghubungkannya ke Outpost Anda. Lihat informasi yang lebih lengkap dalam dokumentasi AWS Outposts. Jalankan perintah berikut untuk mengonfirmasi bahwa komunikasi telah dibuat dengan cluster Anda.

   ```
   kubectl get svc
   ```

   Contoh output adalah sebagai berikut.

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

5. (Opsional) Uji otentikasi ke klaster lokal Anda saat berada dalam keadaan terputus dari. AWS Cloud Untuk petunjuk, lihat Mempersiapkan pemutusan jaringan.

Sumber daya internal

Amazon EKS membuat sumber daya berikut di klaster Anda. Sumber dayanya untuk penggunaan internal Amazon EKS. Agar klaster berfungsi dengan baik, jangan mengedit atau memodifikasi sumber daya ini.

- Cermin berikutPods:
  - `aws-iam-authenticator-`*`node-hostname`*
  - `eks-certificates-controller-`*`node-hostname`*
  - `etcd-`*`node-hostname`*
  - `kube-apiserver-`*`node-hostname`*
  - `kube-controller-manager-`*`node-hostname`*
  - `kube-scheduler-`*`node-hostname`*
- Pengaya yang dikelola sendiri berikut ini:
  - `kube-system/coredns`
  - `kube-system/kube-proxy`(tidak dibuat sampai Anda menambahkan node pertama Anda)
  - `kube-system/aws-node`(tidak dibuat sampai Anda menambahkan node pertama Anda). Cluster lokal menggunakan Amazon VPC CNI plugin for Kubernetes plugin untuk jaringan cluster. Jangan mengubah konfigurasi untuk instance control plane (Pod bernama`aws-node-controlplane-*`). Ada variabel konfigurasi yang dapat Anda gunakan untuk mengubah nilai default ketika plugin membuat antarmuka jaringan baru. Untuk informasi lebih lanjut, lihat dokumentasi di GitHub.
- Layanan berikut:
  - `default/kubernetes`
  - `kube-system/kube-dns`
- Sebuah `PodSecurityPolicy` bernama `eks.system`
- Sebuah `ClusterRole` bernama `eks:system:podsecuritypolicy`

- Sebuah `ClusterRoleBinding` bernama `eks:system`
- Default PodSecurityPolicy
- Selain grup keamanan cluster, Amazon EKS membuat grup keamanan di nama Anda Akun AWS `eks-local-internal-do-not-use-or-edit-`*`cluster-name`*`-`*`uniqueid`*. Grup keamanan ini memungkinkan lalu lintas mengalir bebas di antara Kubernetes komponen yang berjalan pada instance bidang kontrol.

Langkah selanjutnya yang disarankan:

- Berikan prinsip IAM yang membuat cluster izin yang diperlukan untuk melihat Kubernetes sumber daya di AWS Management Console
- Berikan akses entitas IAM ke klaster Anda. Jika Anda ingin entitas melihat Kubernetes sumber daya di konsol Amazon EKS, berikan Izin yang diperlukan kepada entitas.
- Konfigurasikan logging untuk klaster Anda
- Biasakan diri Anda dengan apa yang terjadi selama pemutusan jaringan.
- Tambahkan node ke cluster Anda
- Pertimbangkan untuk menyiapkan rencana cadangan untuk Anda`etcd`. Amazon EKS tidak mendukung pencadangan dan pemulihan otomatis `etcd` untuk cluster lokal. Untuk informasi selengkapnya, lihat Mencadangkan `etcd` klaster di Kubernetes dokumentasi. Dua opsi utama digunakan `etcdctl` untuk mengotomatiskan pengambilan snapshot atau menggunakan cadangan volume penyimpanan Amazon EBS.

Amazon EKS versi platform cluster lokal

Versi platform cluster lokal mewakili kemampuan kluster Amazon EKS padaAWS Outposts. Versi menyertakan komponen yang berjalan di bidang Kubernetes kontrol, yang flag server Kubernetes API diaktifkan. Mereka juga menyertakan versi Kubernetes patch saat ini. Setiap versi Kubernetes minor memiliki satu atau lebih versi platform terkait. Versi platform untuk versi Kubernetes minor yang berbeda bersifat independen. Versi platform untuk cluster lokal dan kluster Amazon EKS di cloud bersifat independen.

Ketika versi Kubernetes minor baru tersedia untuk cluster lokal, seperti`1.28`, versi platform awal untuk versi Kubernetes minor dimulai pada`eks-local-outposts.1`. Namun, Amazon EKS merilis versi platform baru secara berkala untuk mengaktifkan pengaturan bidang Kubernetes kontrol baru dan untuk memberikan perbaikan keamanan.

Saat versi platform cluster lokal baru tersedia untuk versi minor:

- Nomor versi platform bertambah (`eks-local-outposts.n+1`).
- Amazon EKS secara otomatis memperbarui semua cluster lokal yang ada ke versi platform terbaru untuk versi Kubernetes minor yang sesuai. Pembaruan otomatis versi platform yang ada diluncurkan secara bertahap. Proses peluncuran mungkin memakan waktu lama. Jika Anda membutuhkan fitur versi platform terbaru segera, kami sarankan Anda membuat cluster lokal baru.
- Amazon EKS boleh menerbitkan AMI simpul baru dengan versi patch yang sesuai. Semua versi patch kompatibel antara bidang Kubernetes kontrol dan AMI simpul untuk satu versi Kubernetes minor.

Versi platform baru tidak memperkenalkan perubahan yang melanggar atau menyebabkan gangguan layanan.

Cluster lokal selalu dibuat dengan versi platform terbaru yang tersedia (`eks-local-outposts.n`) untuk Kubernetes versi yang ditentukan.

Versi platform saat ini dan terbaru dijelaskan dalam tabel berikut.

Kubernetesversi **`1.28`**

Pengontrol penerimaan berikut diaktifkan untuk semua versi `1.28` platform:`CertificateApproval`,,,`CertificateSigning`,`CertificateSubjectRestriction`,`Defau`
`TaintNodesByConditionValidatingAdmissionPolicy`,
dan`ValidatingAdmissionWebhook`.

Versi Kubernetes	Amazon EKS versi platform	Catatan perilisan	Tanggal rilis
`1.28.1`	`eks-local-outposts.1`	Rilis awal Kubernetes versi `1.28` untuk cluster Amazon EKS lokal di Outposts.	4 Oktober 2023

Kubernetesversi **`1.27`**

Pengontrol penerimaan berikut diaktifkan untuk semua versi `1.27` platform:`CertificateApproval`,,,`CertificateSigning`,`CertificateSubjectRestriction`,`Defau`

TaintNodesByConditionValidatingAdmissionPolicy, danValidatingAdmissionWebhook.

Versi Kubernetes	Amazon EKS versi platform	Catatan perilisan	Tanggal rilis
1.27.3	eks-local-outposts.3	Versi platform baru dengan perbaikan keamanan dan penyempurnaan. kube-proxy diperbarui kev1.27.3. Plugin Amazon VPC CNI untuk Kubernetes diperbarui ke. v1.13.2	14 Juli 2023
1.27.1	eks-local-outposts.2	Diperbarui gambar CoreDNS ke v1.10.1	Juni 22, 2023
1.27.1	eks-local-outposts.1	Rilis awal Kubernetes versi 1.27 untuk cluster Amazon EKS lokal di Outposts.	30 Mei 2023

Kubernetesversi **1.26**

Pengontrol penerimaan berikut diaktifkan untuk semua versi 1.26 platform:CertificateApproval,,,CertificateSigning,CertificateSubjectRestriction,Defau TaintNodesByConditionValidatingAdmissionPolicy, danValidatingAdmissionWebhook.

Versi Kubernetes	Amazon EKS versi platform	Catatan perilisan	Tanggal rilis
1.26.6	eks-local-outposts.4	Versi platform baru dengan perbaikan keamanan dan penyempurnaan. kube-proxy diperbarui kev1.26.6. Plugin Amazon VPC CNI untuk	14 Juli 2023

Versi Kubernetes	Amazon EKS versi platform	Catatan perilisan	Tanggal rilis
		Kubernetes diperbarui ke. `v1.13.2`	
`1.26.4`	`eks-local-outposts.3`	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	13 Juli 2023
`1.26.2`	`eks-local-outposts.2`	Diperbarui versi Bottlerocket ke `1.13.2`	2 Mei 2023
`1.26.2`	`eks-local-outposts.1`	Rilis awal Kubernetes versi `1.26` untuk cluster Amazon EKS lokal di Outposts.	11 April 2023

Kubernetesversi **`1.25`**

Pengontrol penerimaan berikut diaktifkan untuk semua versi `1.25` platform:`CertificateApproval`,,,`CertificateSigning`,`CertificateSubjectRestriction`,`Defau`
`StorageObjectInUseProtectionTaintNodesByCondition`,
dan`ValidatingAdmissionWebhook`.

Versi Kubernetes	Amazon EKS versi platform	Catatan perilisan	Tanggal rilis
`1.25.11`	`eks-local-outposts.6`	Versi platform baru dengan perbaikan keamanan dan penyempurnaan. `kube-prox y` diperbarui ke`v1.25.11`. Plugin Amazon VPC CNI untuk Kubernetes diperbarui ke. `v1.13.2`	14 Juli 2023

Versi Kubernetes	Amazon EKS versi platform	Catatan perilisan	Tanggal rilis
1.25.9	eks-local-outposts.5	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	13 Juli 2023
1.25.6	eks-local-outposts.4	Diperbarui versi Bottlerocket ke 1.13.2	2 Mei 2023
1.25.6	eks-local-outposts.3	Sistem operasi instans pesawat kontrol Amazon EKS diperbaru i ke versi Bottlerocket v1.13.1 dan Amazon VPC CNI plugin for Kubernetes diperbarui ke versi. v1.12.6	April 14, 2023
1.25.6	eks-local-outposts.2	Pengumpulan diagnostik yang ditingkatkan untuk instance bidang Kubernetes kontrol.	8 Maret 2023
1.25.6	eks-local-outposts.1	Rilis awal Kubernetes versi 1.25 untuk cluster Amazon EKS lokal di Outposts.	1 Maret 2023

Kubernetesversi **1.24**

Pengontrol penerimaan berikut diaktifkan untuk semua versi 1.24 platform:DefaultStorageClass,,,DefaultTolerationSeconds,LimitRanger,MutatingAdmissionW CertificateSubjectRestrictionRuntimeClass, danDefaultIngressClass.

Versi Kubernetes	Amazon EKS versi platform	Catatan perilisan	Tanggal rilis
1.24.15	eks-local-outposts.6	Versi platform baru dengan perbaikan keamanan dan penyempurnaan. kube-prox	14 Juli 2023

Versi Kubernetes	Amazon EKS versi platform	Catatan perilisan	Tanggal rilis
		y diperbarui kev1.24.15. Plugin Amazon VPC CNI untuk Kubernetes diperbarui ke. v1.13.2	
1.24.13	eks-local-outposts.5	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	13 Juli 2023
1.24.9	eks-local-outposts.4	Diperbarui versi Bottlerocket ke 1.13.2	2 Mei 2023
1.24.9	eks-local-outposts.3	Sistem operasi instans pesawat kontrol Amazon EKS diperbarui ke versi Bottleroc ket v1.13.1 dan Amazon VPC CNI plugin for Kubernetes diperbarui ke versi. v1.12.6	April 14, 2023
1.24.9	eks-local-outposts.2	Pengumpulan diagnostik yang ditingkatkan untuk instance bidang Kubernetes kontrol.	8 Maret 2023
1.24.9	eks-local-outposts.1	Rilis awal Kubernetes versi 1.24 untuk cluster Amazon EKS lokal di Outposts.	Januari 17, 2023

Kubernetesversi **1.23**

Pengontrol penerimaan berikut diaktifkan untuk semua versi 1.23 platform:DefaultStorageClass,,,DefaultTolerationSeconds,LimitRanger,MutatingAdmissionW CertificateSubjectRestrictionRuntimeClass, danDefaultIngressClass.

Versi Kubernetes	Amazon EKS versi platform	Catatan perilisan	Tanggal rilis
1.23.17	eks-local-outposts.5	Versi platform baru dengan perbaikan dan penyempurnaan keamanan.	13 Juli 2023
1.23.15	eks-local-outposts.4	Diperbarui versi Bottlerocket ke 1.13.2	2 Mei 2023
1.23.15	eks-local-outposts.3	Sistem operasi instans pesawat kontrol Amazon EKS diperbaru i ke versi Bottlerocket v1.13.1 dan Amazon VPC CNI plugin for Kubernetes diperbarui ke versi. v1.12.6	April 14, 2023
1.23.15	eks-local-outposts.2	Pengumpulan diagnostik yang ditingkatkan untuk instance bidang Kubernetes kontrol.	8 Maret 2023
1.23.15	eks-local-outposts.1	Rilis awal Kubernetes versi 1.23 untuk cluster Amazon EKS lokal di Outposts.	Januari 17, 2023

Persyaratan dan pertimbangan VPC klaster lokal dan subnet Amazon EKS

Saat membuat cluster lokal, Anda menentukan VPC dan setidaknya satu subnet pribadi yang berjalan di Outposts. Topik ini memberikan gambaran umum tentang persyaratan dan pertimbangan VPC dan subnet untuk klaster lokal Anda.

Persyaratan dan pertimbangan VPC

Saat Anda membuat klaster lokal, VPC yang Anda tentukan harus memenuhi persyaratan dan pertimbangan berikut:

- Pastikan VPC memiliki alamat IP yang cukup untuk cluster lokal, node apa pun, dan Kubernetes sumber daya lain yang ingin Anda buat. Jika VPC yang ingin Anda gunakan tidak memiliki

cukup alamat IP, tingkatkan jumlah alamat IP yang tersedia. Anda dapat melakukan ini dengan mengaitkan blok Classless Inter-Domain Routing (CIDR) tambahan dengan VPC Anda. Anda dapat mengaitkan blok CIDR pribadi (RFC 1918) dan publik (non-RFC 1918) ke VPC Anda baik sebelum atau setelah Anda membuat cluster Anda. Diperlukan waktu cluster hingga 5 jam agar blok CIDR yang Anda kaitkan dengan VPC dikenali.

- VPC tidak dapat menetapkan awalan IP atau blok CIDR IPv6. Karena kendala ini, informasi yang tercakup dalam Tingkatkan jumlah alamat IP yang tersedia untuk node Amazon EC2 Anda dan `IPv6`alamat untuk cluster,Pods, dan services tidak berlaku untuk VPC Anda.
- VPC memiliki nama host DNS dan resolusi DNS diaktifkan. Tanpa fitur ini, cluster lokal gagal dibuat, dan Anda perlu mengaktifkan fitur dan membuat ulang cluster Anda. Untuk informasi selengkapnya, lihat atribut DNS untuk VPC Anda di Panduan Pengguna Amazon VPC.
- Untuk mengakses kluster lokal Anda melalui jaringan lokal Anda, VPC harus dikaitkan dengan tabel rute gateway lokal Outpost Anda. Untuk informasi selengkapnya, lihat asosiasi VPC di AWS Outposts Panduan Pengguna.

Persyaratan dan pertimbangan subnet

Saat Anda membuat cluster, tentukan setidaknya satu subnet pribadi. Jika Anda menentukan lebih dari satu subnet, instance bidang Kubernetes kontrol didistribusikan secara merata di seluruh subnet. Jika lebih dari satu subnet ditentukan, subnet harus ada di Outpost yang sama. Selain itu, subnet juga harus memiliki rute yang tepat dan izin grup keamanan untuk berkomunikasi satu sama lain. Saat Anda membuat cluster lokal, subnet yang Anda tentukan harus memenuhi persyaratan berikut:

- Subnet semuanya berada di Outpost logis yang sama.
- Subnet bersama-sama memiliki setidaknya tiga alamat IP yang tersedia untuk instance bidang Kubernetes kontrol. Jika tiga subnet ditentukan, setiap subnet harus memiliki setidaknya satu alamat IP yang tersedia. Jika dua subnet ditentukan, setiap subnet harus memiliki setidaknya dua alamat IP yang tersedia. Jika satu subnet ditentukan, subnet harus memiliki setidaknya tiga alamat IP yang tersedia.
- Subnet memiliki rute ke gateway lokal rak Outpost untuk mengakses server Kubernetes API melalui jaringan lokal Anda. Jika subnet tidak memiliki rute ke gateway lokal rak Outpost, Anda harus berkomunikasi dengan server Kubernetes API Anda dari dalam VPC.
- Subnet harus menggunakan penamaan berbasis alamat IP. Penamaan berbasis sumber daya Amazon EC2 tidak didukung oleh Amazon EKS.

Akses subnet ke Layanan AWS

Subnet pribadi cluster lokal di Outposts harus dapat berkomunikasi dengan Regional. Layanan AWS Anda dapat mencapai ini dengan menggunakan gateway NAT untuk akses internet keluar atau, jika Anda ingin menjaga semua lalu lintas pribadi dalam VPC Anda, menggunakan titik akhir VPC antarmuka.

Menggunakan gateway NAT

Subnet pribadi cluster lokal di Outposts harus memiliki tabel rute terkait yang memiliki rute ke gateway NAT di subnet publik yang berada di Availability Zone induk Outpost. Subnet publik harus memiliki rute ke gateway internet. Gateway NAT memungkinkan akses internet keluar dan mencegah koneksi masuk yang tidak diminta dari internet ke instance di Outpost.

Menggunakan VPC endpoint antarmuka

Jika subnet pribadi cluster lokal di Outposts tidak memiliki koneksi internet keluar, atau jika Anda ingin menjaga semua lalu lintas pribadi dalam VPC Anda, maka Anda harus membuat titik akhir VPC antarmuka berikut dan titik akhir gateway di subnet Regional sebelum membuat cluster Anda.

Titik akhir	Tipe titik akhir
`com.amazonaws.` *`region-code`* `.ssm`	Antarmuka
`com.amazonaws.` *`region-code`* `.ssmmessages`	Antarmuka
`com.amazonaws.` *`region-code`* `.ec2messages`	Antarmuka
`com.amazonaws.` *`region-code`* `.ec2`	Antarmuka
`com.amazonaws.` *`region-code`* `.secretsmanager`	Antarmuka
`com.amazonaws.` *`region-code`* `.logs`	Antarmuka
`com.amazonaws.` *`region-code`* `.sts`	Antarmuka
`com.amazonaws.` *`region-code`* `.ecr.api`	Antarmuka

Titik akhir	Tipe titik akhir
com.amazonaws. *region-code* .ecr.dkr	Antarmuka
com.amazonaws. *region-code* .s3	Gateway

Titik akhir harus memenuhi persyaratan berikut:

- Dibuat di subnet pribadi yang terletak di Availability Zone induk Outpost Anda
- Miliki nama DNS pribadi diaktifkan
- Memiliki grup keamanan terlampir yang memungkinkan lalu lintas HTTPS masuk dari kisaran CIDR dari subnet pos terdepan pribadi.

Membuat titik akhir menimbulkan biaya. Untuk informasi selengkapnya, lihat Harga AWS PrivateLink. Jika Anda Pods membutuhkan akses ke yang lainLayanan AWS, maka Anda perlu membuat titik akhir tambahan. Untuk daftar lengkap titik akhir, lihat Layanan AWSyang terintegrasi dengan AWS PrivateLink.

Buat VPC

Anda dapat membuat VPC yang memenuhi persyaratan sebelumnya menggunakan salah satu templat berikut: AWS CloudFormation

- Template 1 - Template ini membuat VPC dengan satu subnet pribadi di Outpost dan satu subnet publik di. Wilayah AWS Subnet pribadi memiliki rute ke internet melalui NAT Gateway yang berada di subnet publik di. Wilayah AWS Template ini dapat digunakan untuk membuat cluster lokal di subnet dengan akses internet jalan keluar.
- Template 2 — Template ini membuat VPC dengan satu subnet pribadi di Outpost dan set minimum VPC Endpoint yang diperlukan untuk membuat cluster lokal di subnet yang tidak memiliki akses internet ingress atau egress (juga disebut sebagai subnet pribadi).

Mempersiapkan pemutusan jaringan

Jika jaringan lokal Anda kehilangan konektivitas denganAWS Cloud, Anda dapat terus menggunakan kluster Amazon EKS lokal Anda di Outpost. Topik ini mencakup bagaimana Anda dapat mempersiapkan cluster lokal Anda untuk pemutusan jaringan dan pertimbangan terkait.

Pertimbangan saat menyiapkan klaster lokal Anda untuk pemutusan jaringan:

- Cluster lokal memungkinkan stabilitas dan operasi lanjutan selama pemutusan jaringan sementara yang tidak direncanakan. AWS Outpoststetap merupakan penawaran yang sepenuhnya terhubung yang bertindak sebagai perpanjangan dari AWS Cloud di pusat data Anda. Jika jaringan terputus antara Outpost Anda danAWS Cloud, kami sarankan untuk mencoba memulihkan koneksi Anda. Untuk instruksi, lihat daftar periksa pemecahan masalah jaringan AWS Outposts rak di Panduan Pengguna. AWS Outposts Untuk informasi selengkapnya tentang cara memecahkan masalah dengan kluster lokal, lihat. Memecahkan masalah kluster lokal untuk Amazon EKS di AWS Outposts
- Outposts memancarkan `ConnectedStatus` metrik yang dapat Anda gunakan untuk memantau status konektivitas Outpost Anda. Untuk informasi selengkapnya, lihat Metrik Outposts di Panduan Pengguna. AWS Outposts
- Cluster lokal menggunakan IAM sebagai mekanisme otentikasi default menggunakan authenticator untukAWS Identity and Access Management. Kubernetes IAM tidak tersedia selama pemutusan jaringan. Jadi, cluster lokal mendukung mekanisme otentikasi alternatif menggunakan `x.509` sertifikat yang dapat Anda gunakan untuk terhubung ke cluster Anda selama pemutusan jaringan. Untuk informasi tentang cara mendapatkan dan menggunakan `x.509` sertifikat untuk klaster Anda, lihatMengautentikasi ke cluster lokal Anda selama pemutusan jaringan.
- Jika Anda tidak dapat mengakses Route 53 selama pemutusan jaringan, pertimbangkan untuk menggunakan server DNS lokal di lingkungan lokal Anda. Instans bidang Kubernetes kontrol menggunakan alamat IP statis. Anda dapat mengonfigurasi host yang Anda gunakan untuk terhubung ke cluster Anda dengan nama host endpoint dan alamat IP sebagai alternatif untuk menggunakan server DNS lokal. Untuk informasi selengkapnya, lihat DNS dalam Panduan Pengguna AWS Outposts.
- Jika Anda mengharapkan peningkatan lalu lintas aplikasi selama pemutusan jaringan, Anda dapat menyediakan kapasitas komputasi cadangan di cluster Anda saat terhubung ke cloud. Instans Amazon EC2 sudah termasuk dalam harga. AWS Outposts Jadi, menjalankan instance cadangan tidak memengaruhi biaya AWS penggunaan Anda.
- Selama pemutusan jaringan untuk mengaktifkan operasi pembuatan, pembaruan, dan skala untuk beban kerja, gambar kontainer aplikasi Anda harus dapat diakses melalui jaringan lokal dan cluster Anda harus memiliki kapasitas yang cukup. Cluster lokal tidak meng-host registri kontainer untuk Anda. Jika sebelumnya Pods telah berjalan pada node tersebut, gambar kontainer di-cache pada node. Jika Anda biasanya menarik gambar kontainer aplikasi Anda dari Amazon ECR di cloud, pertimbangkan untuk menjalankan cache atau registri lokal. Cache atau registri lokal sangat

membantu jika Anda memerlukan operasi buat, perbarui, dan skala untuk sumber daya beban kerja selama pemutusan jaringan.

- Cluster lokal menggunakan Amazon EBS sebagai kelas penyimpanan default untuk volume persisten dan driver Amazon EBS CSI untuk mengelola siklus hidup volume persisten Amazon EBS. Selama pemutusan jaringan, Pods yang didukung oleh Amazon EBS tidak dapat dibuat, diperbarui, atau diskalakan. Ini karena operasi ini memerlukan panggilan ke Amazon EBS API di cloud. Jika Anda menerapkan beban kerja stateful pada kluster lokal dan memerlukan operasi pembuatan, pembaruan, atau skala selama pemutusan jaringan, pertimbangkan untuk menggunakan mekanisme penyimpanan alternatif.
- Snapshot Amazon EBS tidak dapat dibuat atau dihapus jika tidak AWS Outposts dapat mengakses API AWS dalam wilayah yang relevan (seperti API untuk Amazon EBS atau Amazon S3).
- Saat mengintegrasikan ALB (Ingress) dengan AWS Certificate Manager (ACM), sertifikat didorong dan disimpan dalam memori instance ALB Compute. AWS Outposts Penghentian TLS saat ini akan terus beroperasi jika terjadi pemutusan sambungan dari. Wilayah AWS Operasi mutasi dalam konteks ini akan gagal (seperti definisi ingress baru, operasi API sertifikat berbasis ACM baru, skala komputasi ALB, atau rotasi sertifikat). Untuk informasi selengkapnya, lihat Memecahkan masalah perpanjangan sertifikat terkelola di Panduan Pengguna. AWS Certificate Manager
- Log bidang kontrol Amazon EKS di-cache secara lokal pada instance bidang Kubernetes kontrol selama pemutusan jaringan. Setelah menyambung kembali, log dikirim ke CloudWatch Log di indukWilayah AWS. Anda dapat menggunakan Prometheus, Grafana, atau solusi mitra Amazon EKS untuk memantau kluster secara lokal menggunakan titik akhir metrik server Kubernetes API atau menggunakan Fluent Bit log.
- Jika Anda menggunakan AWS Load Balancer Controller on Outposts untuk lalu lintas aplikasi, yang ada di Pods depan oleh AWS Load Balancer Controller terus menerima lalu lintas selama jaringan terputus. Baru Pods dibuat selama pemutusan jaringan tidak menerima lalu lintas sampai Outpost tersambung kembali ke. AWS Cloud Pertimbangkan untuk mengatur jumlah replika untuk aplikasi Anda saat terhubung ke AWS Cloud untuk mengakomodasi kebutuhan penskalaan Anda selama pemutusan jaringan.
- Amazon VPC CNI plugin for KubernetesDefault ke mode IP sekunder. Ini dikonfigurasi dengan `WARM_ENI_TARGET =1`, yang memungkinkan plugin untuk menjaga “full elastic network interface” dari alamat IP yang tersedia. Pertimbangkan untuk mengubah`WARM_ENI_TARGET`,`WARM_IP_TARGET`, dan `MINIMUM_IP_TARGET` nilai sesuai dengan kebutuhan penskalaan Anda selama keadaan terputus. Untuk informasi selengkapnya, lihat `readme`file untuk plugin di GitHub. Untuk daftar jumlah maksimum Pods yang didukung oleh setiap jenis instance, lihat `eni-max-pods.txt`file di GitHub.

Mengautentikasi ke cluster lokal Anda selama pemutusan jaringan

AWS Identity and Access Management(IAM) tidak tersedia selama pemutusan jaringan. Anda tidak dapat mengautentikasi ke klaster lokal menggunakan kredensyal IAM saat terputus. Namun, Anda dapat terhubung ke cluster Anda melalui jaringan lokal Anda menggunakan `x509` sertifikat saat terputus. Anda perlu mengunduh dan menyimpan `X509` sertifikat klien untuk digunakan selama pemutusan sambungan. Dalam topik ini, Anda mempelajari cara membuat dan menggunakan sertifikat untuk mengautentikasi ke klaster Anda saat berada dalam keadaan terputus.

1. Buat permintaan penandatanganan sertifikat.

   a. Buat permintaan penandatanganan sertifikat.

   ```
   openssl req -new -newkey rsa:4096 -nodes -days 365 \
       -keyout admin.key -out admin.csr -subj "/CN=admin"
   ```

   b. Buat permintaan penandatanganan sertifikat diKubernetes.

   ```
   BASE64_CSR=$(cat admin.csr | base64 -w 0)
   cat << EOF > admin-csr.yaml
   apiVersion: certificates.k8s.io/v1
   kind: CertificateSigningRequest
   metadata:
     name: admin-csr
   spec:
     signerName: kubernetes.io/kube-apiserver-client
     request: ${BASE64_CSR}
     usages:
     - client auth
   EOF
   ```

2. Buat permintaan penandatanganan sertifikat menggunakan`kubectl`.

   ```
   kubectl create -f admin-csr.yaml
   ```

3. Periksa status permintaan penandatanganan sertifikat.

   ```
   kubectl get csr admin-csr
   ```

   Contoh output adalah sebagai berikut.

```
NAME       AGE   REQUESTOR                      CONDITION
admin-csr  11m   kubernetes-admin               Pending
```

Kubernetesmembuat permintaan penandatanganan sertifikat.

4. Menyetujui permintaan penandatanganan sertifikat.

```
kubectl certificate approve admin-csr
```

5. Periksa kembali status permintaan penandatanganan sertifikat untuk persetujuan.

```
kubectl get csr admin-csr
```

Contoh output adalah sebagai berikut.

```
NAME       AGE   REQUESTOR                      CONDITION
admin-csr  11m   kubernetes-admin               Approved
```

6. Ambil dan verifikasi sertifikat.

a. Ambil sertifikat.

```
kubectl get csr admin-csr -o jsonpath='{.status.certificate}' | base64 --decode
 > admin.crt
```

b. Verifikasi sertifikat.

```
cat admin.crt
```

7. Buat pengikatan peran klaster untuk `admin` pengguna.

```
kubectl create clusterrolebinding admin --clusterrole=cluster-admin \
    --user=admin --group=system:masters
```

8. Hasilkan kubeconfig dengan cakupan pengguna untuk status terputus.

Anda dapat membuat `kubeconfig` file menggunakan `admin` sertifikat yang diunduh. Ganti *`my-cluster`* dan *`apiserver-endpoint`* dalam perintah berikut.

```
aws eks describe-cluster --name my-cluster \
```

```
    --query "cluster.certificateAuthority" \
    --output text | base64 --decode > ca.crt
```

```
kubectl config --kubeconfig admin.kubeconfig set-cluster my-cluster \
    --certificate-authority=ca.crt --server apiserver-endpoint --embed-certs
```

```
kubectl config --kubeconfig admin.kubeconfig set-credentials admin \
    --client-certificate=admin.crt --client-key=admin.key --embed-certs
```

```
kubectl config --kubeconfig admin.kubeconfig set-context admin@my-cluster \
    --cluster my-cluster --user admin
```

```
kubectl config --kubeconfig admin.kubeconfig use-context admin@my-cluster
```

9. Lihat `kubeconfig` file Anda.

```
kubectl get nodes --kubeconfig admin.kubeconfig
```

10. Jika Anda memiliki layanan yang sudah diproduksi di Outpost Anda, lewati langkah ini. Jika Amazon EKS adalah satu-satunya layanan yang berjalan di Outpost Anda dan Outpost saat ini tidak dalam produksi, Anda dapat mensimulasikan pemutusan jaringan. Sebelum Anda masuk ke produksi dengan cluster lokal Anda, simulasikan pemutusan untuk memastikan bahwa Anda dapat mengakses klaster Anda saat berada dalam keadaan terputus.

    a. Terapkan aturan firewall pada perangkat jaringan yang menghubungkan Outpost Anda ke. Wilayah AWS Ini memutus tautan layanan dari Outpost. Anda tidak dapat membuat instance baru. Saat ini instans yang sedang berjalan kehilangan konektivitas ke Wilayah AWS dan internet.

    b. Anda dapat menguji koneksi ke cluster lokal Anda saat terputus menggunakan `x509` sertifikat. Pastikan untuk mengubah Anda `kubeconfig` ke `admin.kubeconfig` yang Anda buat pada langkah sebelumnya. Ganti *`my-cluster`* dengan nama cluster lokal Anda.

    ```
    kubectl config use-context admin@my-cluster --kubeconfig admin.kubeconfig
    ```

    Jika Anda melihat ada masalah dengan kluster lokal Anda saat berada dalam keadaan terputus, sebaiknya buka tiket dukungan.

Pertimbangan kapasitas pertimbangan kapasitas pertimbangan kapasitas

Topik ini memberikan panduan untuk memilih jenis instans bidangKubernetes kontrol dan (opsional) menggunakan grup penempatan untuk memenuhi persyaratan ketersediaan tinggi untuk klaster Amazon EKS lokal Anda di Outpost.

Sebelum Anda memilih jenis instans (sepertim5,c5, ataur5) yang akan digunakan untuk bidangKubernetes kontrol klaster lokal Anda di Outposts, konfirmasikan jenis instans yang tersedia pada konfigurasi Outpost Anda. Setelah Anda mengidentifikasi jenis instans yang tersedia, pilih ukuran instans (sepertilargexlarge,, atau2xlarge) berdasarkan jumlah node yang diperlukan beban kerja Anda. Tabel berikut memberikan rekomendasi untuk memilih ukuran instans.

> **Note**
>
> Ukuran instance harus ditempatkan di Outposts Anda. Pastikan Anda memiliki kapasitas yang cukup untuk tiga contoh ukuran yang tersedia di Outposts Anda untuk seumur hidup klaster lokal Anda. Untuk daftar jenisAmazon EC2 instans yang tersedia, lihat bagian Komputasi dan penyimpanan dalam fiturAWS Outposts rak.

Jumlah simpul simpul simpul simpul	Ukuran instance control plane Kubernetes
1—20	large
21—100	xlarge
101—250	2xlarge
251—500	4xlarge

Penyimpanan untuk bidangKubernetes kontrol memerlukan penyimpanan Amazon EBS 246 GB untuk setiap klaster lokal untuk memenuhietcd IOPS yang diperlukan. Saat klaster lokal dibuat, volume Amazon EBS disediakan secara otomatis untuk Anda.

Penempatan bidang kendali kendali kendali bidang kendali kendali

Bila Anda tidak menentukan grup penempatan dengan`OutpostConfig.ControlPlanePlacement.GroupName` properti, instans Amazon EC2

yang disediakan untuk bidangKubernetes kontrol Anda tidak menerima penegakan penempatan perangkat keras tertentu di seluruh kapasitas dasar yang tersedia di Outpost Anda.

Anda dapat menggunakan grup penempatan untuk memenuhi persyaratan ketersediaan tinggi untuk klaster Amazon EKS lokal Anda di Outpost. Dengan menentukan grup penempatan selama pembuatan klaster, Anda memengaruhi penempatan instance bidangKubernetes kontrol. Instans tersebar di seluruh perangkat keras yang mendasari independen (rak atau host), meminimalkan dampak instans yang berkorelasi pada peristiwa kegagalan perangkat keras.

Persyaratan

Jenis spread yang dapat Anda konfigurasikan tergantung pada jumlah rak Outpost yang Anda miliki dalam penyebaran Anda.

- Deployment dengan satu atau dua rak fisik di Outpost logis tunggal - Anda harus memiliki setidaknya tiga host yang dikonfigurasi dengan jenis instans yang Anda pilih untuk instanceKubernetes control plane Anda. Grup penempatan spread menggunakan spread tingkat host memastikan bahwa semua instance bidangKubernetes kontrol berjalan pada host yang berbeda dalam rak dasar yang tersedia dalam penyebaran Outpost Anda.
- Deployment dengan tiga atau lebih rak fisik di Outpost logis tunggal - Anda harus memiliki setidaknya tiga host yang dikonfigurasi dengan jenis instans yang Anda pilih untuk instanceKubernetes control plane Anda. Grup penempatan spread menggunakan spread level rak memastikan bahwa semua instanceKubernetes control plane berjalan di rak berbeda dalam penyebaran Outpost Anda. Anda juga dapat menggunakan grup penempatan spread tingkat host seperti yang dijelaskan di opsi sebelumnya.

Anda bertanggung jawab untuk membuat grup penempatan yang diinginkan. Anda menentukan grup penempatan saat memanggil`CreateCluster` API. Untuk informasi lebih lanjut tentang Grup penempatan dan cara membuatnya, lihat Grup penempatan di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Pertimbangan-pertimbangan

- Ketika grup penempatan ditentukan, harus ada kapasitas slotted yang tersedia di Outpost Anda untuk berhasil membuat klaster Amazon EKS lokal. Kapasitasnya bervariasi berdasarkan apakah Anda menggunakan tipe host atau rack spread. Jika tidak cukup kapasitas, klaster tetap di`Creating` negara bagian. Anda dapat memeriksa`Insufficient Capacity Error` pada

bidang kesehatan responDescribeCluster API. Anda harus membebaskan kapasitas untuk proses pembuatan untuk maju.

- Selama pembaruan platform klaster lokal dan versi Amazon EKS, instans bidangKubernetes kontrol dari klaster Anda digantikan oleh instans baru menggunakan strategi pembaruan bergulir. Selama proses penggantian ini, setiap instance control plane diakhiri, membebaskan slot masing-masing. Instance baru yang diperbarui disediakan di tempatnya. Contoh diperbarui mungkin ditempatkan di slot yang dirilis. Jika slot dikonsumsi oleh contoh lain yang tidak terkait dan tidak ada lagi kapasitas tersisa yang menghormati persyaratan penyebaran topologi yang diperlukan, maka cluster tetap dalamUpdating keadaan. Anda dapat melihat masing-masingInsufficient Capacity Error di bidang kesehatan responsDescribeCluster API. Anda harus membebaskan kapasitas sehingga proses pembaruan dapat maju dan membangun kembali tingkat ketersediaan tinggi sebelumnya.
- Anda dapat membuat 500 Grup penempatan dapat membuat 500 Grup penempatan dapat membuat 500 Grup penempatan dapat membuat 500 Grup penempatan dapat membuat 500 Grup penempatan dapat membuat 500Wilayah AWS grup penempatan Untuk informasi lebih lanjut, lihat Aturan umum dan batasan di Panduan Pengguna Amazon EC2 untuk Instans Linux.

Memecahkan masalah kluster lokal untuk Amazon EKS di AWS Outposts

Topik ini mencakup beberapa kesalahan umum yang mungkin Anda lihat saat menggunakan kluster lokal dan cara memecahkan masalah. Cluster lokal mirip dengan cluster Amazon EKS di cloud, tetapi ada beberapa perbedaan dalam cara mereka dikelola oleh Amazon EKS.

Perilaku API

Cluster lokal dibuat melalui Amazon EKS API, tetapi dijalankan secara asinkron. Ini berarti bahwa permintaan ke Amazon EKS API segera dikembalikan untuk kluster lokal. Namun, permintaan ini mungkin berhasil, gagal cepat karena kesalahan validasi input, atau gagal dan memiliki kesalahan validasi deskriptif. Perilaku ini mirip dengan Kubernetes API.

Cluster lokal tidak bertransisi ke FAILED status. Amazon EKS mencoba untuk mendamaikan status cluster dengan status yang diinginkan pengguna secara berkelanjutan. Akibatnya, klaster lokal mungkin tetap berada dalam CREATING status untuk jangka waktu yang lama sampai masalah mendasar teratasi.

Jelaskan bidang kesehatan cluster

Masalah klaster lokal dapat ditemukan menggunakan AWS CLI perintah `describe-cluster`Amazon EKS. Masalah klaster lokal muncul oleh `cluster.health` bidang respons `describe-cluster` perintah. Pesan yang terkandung dalam bidang ini mencakup kode kesalahan, pesan deskriptif, dan ID sumber daya terkait. Informasi ini tersedia melalui Amazon EKS API dan AWS CLI hanya. Dalam contoh berikut, ganti *my-cluster* dengan nama cluster lokal Anda.

```
aws eks describe-cluster --name my-cluster --query 'cluster.health'
```

Contoh output adalah sebagai berikut.

```
{
    "issues": [
        {
            "code": "ConfigurationConflict",
            "message": "The instance type 'm5.large' is not supported in Outpost 'my-outpost-arn'.",
            "resourceIds": [
                "my-cluster-arn"
            ]
        }
    ]
}
```

Jika masalahnya tidak dapat diperbaiki, Anda mungkin perlu menghapus cluster lokal dan membuat yang baru. Misalnya, mencoba menyediakan cluster dengan tipe instance yang tidak tersedia di Outpost Anda. Tabel berikut mencakup kesalahan umum terkait kesehatan.

Skenario kesalahan	Kode	Pesan	ResourceIds
Subnet yang disediakan tidak dapat ditemukan.	`ResourceNotFound`	`The subnet ID subnet-id does not exist`	Semua ID subnet yang disediakan
Subnet yang disediakan bukan milik VPC yang sama.	`ConfigurationConflict`	`Subnets specified must belong to the same VPC`	Semua ID subnet yang disediakan

Skenario kesalahan	Kode	Pesan	ResourceIds
Beberapa subnet yang disediakan bukan milik Outpost yang ditentukan.	`ConfigurationConflict`	`Subnet subnet-id expected to be in outpost-arn , but is in other-outpost-arn`	ID subnet bermasalah
Beberapa subnet yang disediakan bukan milik Pos Luar mana pun.	`ConfigurationConflict`	`Subnet subnet-id is not part of any Outpost`	ID subnet bermasalah
Beberapa subnet yang disediakan tidak memiliki cukup alamat gratis untuk membuat antarmuka jaringan elastis untuk instance bidang kontrol.	`ResourceLimitExceeded`	`The specified subnet does not have enough free addresses to satisfy the request.`	ID subnet bermasalah
Jenis instans bidang kontrol yang ditentukan tidak didukung di Outpost Anda.	`ConfigurationConflict`	`The instance type type is not supported in Outpost outpost-arn`	ARN klaster

Skenario kesalahan	Kode	Pesan	ResourceIds
Anda menghenti kan instans Amazon EC2 bidang kontrol `run-instance` atau berhasil, tetapi status yang diamati berubah menjadi. `Terminated` Hal ini dapat terjadi untuk jangka waktu tertentu setelah Outpost Anda tersambung kembali dan kesalahan internal Amazon EBS menyebabkan alur kerja internal Amazon EC2 gagal.	`InternalFailure`	`EC2 instance state "Terminat ed" is unexpected`	ARN klaster
Anda memiliki kapasitas yang tidak mencukupi di Pos Luar Anda. Ini juga dapat terjadi ketika sebuah cluster sedang dibuat jika Outpost terputus dari. Wilayah AWS	`ResourceL imitExceeded`	`There is not enough capacity on the Outpost to launch or start the instance.`	ARN klaster
Akun Anda melebihi kuota grup keamanan Anda.	`ResourceL imitExceeded`	Pesan galat yang dikembalikan oleh Amazon EC2 API	ID VPC Target
Akun Anda melebihi kuota elastic network interface Anda.	`ResourceL imitExceeded`	Pesan galat yang dikembalikan oleh Amazon EC2 API	ID subnet target

Skenario kesalahan	Kode	Pesan	ResourceIds
Instans pesawat kontrol tidak dapat dijangkau. AWS Systems Manager Untuk resolusi, lihatInstans pesawat kontrol tidak dapat dijangkau melalui AWS Systems Manager.	`ClusterUnreachable`	Instans pesawat kontrol Amazon EKS tidak dapat dijangkau melalui SSM. Harap verifikasi SSM dan konfigurasi jaringan Anda, dan rujuk dokumentasi pemecahan masalah EKS on Outposts.	ID instans Amazon EC2
Terjadi kesalahan saat mendapatkan detail untuk grup keamanan terkelola atau elastic network interface.	Berdasarkan kode kesalahan klien Amazon EC2.	Pesan galat yang dikembalikan oleh Amazon EC2 API	Semua ID grup keamanan terkelola
Terjadi kesalahan saat mengotorisasi atau mencabut aturan masuknya grup keamanan. Ini berlaku untuk kelompok keamanan cluster dan pesawat kontrol.	Berdasarkan kode kesalahan klien Amazon EC2.	Pesan galat yang dikembalikan oleh Amazon EC2 API	ID grup keamanan bermasalah
Terjadi kesalahan saat menghapus elastic network interface untuk instance control plane.	Berdasarkan kode kesalahan klien Amazon EC2.	Pesan galat yang dikembalikan oleh Amazon EC2 API	ID antarmuka elastis network yang bermasalah

Tabel berikut mencantumkan kesalahan dari Layanan AWS yang lain yang disajikan di bidang kesehatan `describe-cluster` respons.

Kode kesalahan Amazon EC2	Kode masalah kesehatan cluster	Deskripsi
`AuthFailure`	`AccessDenied`	Kesalahan ini dapat terjadi karena berbagai alasan. Alasan paling umum adalah bahwa Anda secara tidak sengaja menghapus tag yang digunakan layanan untuk mencoret kebijakan peran terkait layanan dari bidang kontrol. Jika ini terjadi, Amazon EKS tidak dapat lagi mengelola dan memantau AWS sumber daya ini.
`UnauthorizedOperation`	`AccessDenied`	Kesalahan ini dapat terjadi karena berbagai alasan. Alasan paling umum adalah bahwa Anda secara tidak sengaja menghapus tag yang digunakan layanan untuk mencoret kebijakan peran terkait layanan dari bidang kontrol. Jika ini terjadi, Amazon EKS tidak dapat lagi mengelola dan memantau AWS sumber daya ini.
`InvalidSubnetID.NotFound`	`ResourceNotFound`	Kesalahan ini terjadi ketika subnet ID untuk aturan masuknya grup keamanan tidak dapat ditemukan.

Kode kesalahan Amazon EC2	Kode masalah kesehatan cluster	Deskripsi
`InvalidPermission.NotFound`	`ResourceNotFound`	Kesalahan ini terjadi ketika izin untuk aturan masuknya grup keamanan tidak benar.
`InvalidGroup.NotFound`	`ResourceNotFound`	Kesalahan ini terjadi ketika grup aturan masuk grup keamanan tidak dapat ditemukan.
`InvalidNetworkInterfaceID.NotFound`	`ResourceNotFound`	Kesalahan ini terjadi ketika ID antarmuka jaringan untuk aturan masuknya grup keamanan tidak dapat ditemukan.
`InsufficientFreeAddressesInSubnet`	`ResourceLimitExceeded`	Kesalahan ini terjadi ketika kuota sumber daya subnet terlampaui.
`InsufficientCapacityOnOutpost`	`ResourceLimitExceeded`	Kesalahan ini terjadi ketika kuota kapasitas pos terlampaui.
`NetworkInterfaceLimitExceeded`	`ResourceLimitExceeded`	Kesalahan ini terjadi ketika kuota elastic network interface terlampaui.
`SecurityGroupLimitExceeded`	`ResourceLimitExceeded`	Kesalahan ini terjadi ketika kuota grup keamanan terlampaui.

Kode kesalahan Amazon EC2	Kode masalah kesehatan cluster	Deskripsi
`VcpuLimitExceeded`	`ResourceLimitExceeded`	Ini diamati saat membuat instans Amazon EC2 di akun baru. Kesalahannya mungkin mirip dengan yang berikut: `You have requested more vCPU capacity than your current vCPU limit of 32 allows for the instance bucket that the specified instance type belongs to. Please visit http://aws.amazon.com/contact-us/ec2-request to request an adjustment to this limit."`
`InvalidParameterValue`	`ConfigurationConflict`	Amazon EC2 mengembalikan kode kesalahan ini jika jenis instans yang ditentukan tidak didukung di Outpost.
Semua kegagalan lainnya	`InternalFailure`	Tidak ada

Tidak dapat membuat atau memodifikasi kluster

Cluster lokal memerlukan izin dan kebijakan yang berbeda dari kluster Amazon EKS yang di-host di cloud. Saat klaster gagal membuat dan menghasilkan `InvalidPermissions` kesalahan, periksa kembali apakah peran klaster yang Anda gunakan memiliki kebijakan terkelola LocalOutpostClusterPolicyAmazonEks yang dilampirkan padanya. Semua panggilan API lainnya memerlukan set izin yang sama dengan kluster Amazon EKS di cloud.

Cluster macet dalam **`CREATING`** keadaan

Jumlah waktu yang dibutuhkan untuk membuat cluster lokal bervariasi tergantung pada beberapa faktor. Faktor-faktor ini termasuk konfigurasi jaringan Anda, konfigurasi Outpost, dan konfigurasi cluster. Secara umum, cluster lokal dibuat dan berubah `ACTIVE` status dalam 15-20 menit. Jika cluster lokal tetap dalam `CREATING` status, Anda dapat `describe-cluster` meminta informasi tentang penyebabnya di bidang `cluster.health` output.

Masalah yang paling umum adalah sebagai berikut:

AWS Systems Manager(Systems Manager) mengalami masalah berikut:

- Cluster Anda tidak dapat terhubung ke instance control plane dari tempat Wilayah AWS Systems Manager berada. Anda dapat memverifikasi ini dengan menelepon `aws ssm start-session --target` *`instance-id`* dari host benteng In-region. Jika perintah itu tidak berfungsi, periksa apakah Systems Manager berjalan pada instance control plane. Atau, solusi lain adalah menghapus cluster dan kemudian membuatnya kembali.
- Instans pesawat kontrol Systems Manager mungkin tidak memiliki akses internet. Periksa apakah subnet yang Anda berikan saat membuat cluster memiliki gateway NAT dan VPC dengan gateway internet. Gunakan penganalisis jangkauan VPC untuk memverifikasi bahwa instance bidang kontrol dapat mencapai gateway internet. Untuk informasi selengkapnya, lihat Memulai dengan VPC Reachability Analyzer.
- Peran ARN yang Anda berikan adalah kebijakan yang hilang. Periksa apakah AWS kebijakan terkelola: AmazonEks LocalOutpostClusterPolicy telah dihapus dari peran. Ini juga dapat terjadi jika AWS CloudFormation tumpukan salah dikonfigurasi.

Beberapa subnet salah dikonfigurasi dan ditentukan saat cluster dibuat:

- Semua subnet yang disediakan harus dikaitkan dengan Outpost yang sama dan harus saling menjangkau. Saat beberapa subnet ditentukan saat kluster dibuat, Amazon EKS mencoba menyebarkan instance bidang kontrol di beberapa subnet.
- Grup keamanan terkelola Amazon EKS diterapkan di elastic network interface. Namun, elemen konfigurasi lain seperti aturan firewall NACL mungkin bertentangan dengan aturan untuk elastic network interface.

Konfigurasi DNS VPC dan subnet salah dikonfigurasi atau hilang

UlasanPersyaratan dan pertimbangan VPC klaster lokal dan subnet Amazon EKS.

Tidak dapat menggabungkan node ke cluster

Penyebab umum:

- Masalah AMI:
  - Anda menggunakan AMI yang tidak didukung. Anda harus menggunakan v20220620 atau yang lebih baru untuk Amazon Amazon Linux AMI yang dioptimalkan oleh Amazon EKS EKS yang dioptimalkan Amazon Linux.
  - Jika Anda menggunakan AWS CloudFormation template untuk membuat node, pastikan itu tidak menggunakan AMI yang tidak didukung.
- Kehilangan AWS IAM Authenticator `ConfigMap` — Jika hilang, Anda harus membuatnya. Untuk informasi selengkapnya, lihat Terapkan `aws-authConfigMap` ke cluster Anda.
- Kelompok keamanan yang salah digunakan — Pastikan untuk digunakan `eks-cluster-sg-`*`cluster-name`*`-`*`uniqueid`* untuk kelompok keamanan node pekerja Anda. Grup keamanan yang dipilih diubah oleh AWS CloudFormation untuk memungkinkan grup keamanan baru setiap kali tumpukan digunakan.
- Mengikuti langkah-langkah VPC tautan pribadi yang tidak terduga — Data CA yang salah (`--b64-cluster-ca`) atau API Endpoint (`--apiserver-endpoint`) diteruskan.
- Kebijakan Pod keamanan yang salah konfigurasi:
  - The CoreDNS and Amazon VPC CNI plugin for Kubernetes Daemonsets harus berjalan pada node agar node dapat bergabung dan berkomunikasi dengan cluster.
  - Ini Amazon VPC CNI plugin for Kubernetes membutuhkan beberapa fitur jaringan istimewa untuk berfungsi dengan baik. Anda dapat melihat fitur jaringan istimewa dengan perintah berikut:`kubectl describe psp eks.privileged`.

  Kami tidak menyarankan untuk memodifikasi kebijakan keamanan pod default. Untuk informasi selengkapnya, lihat Kebijakan keamanan pod.

Mengumpulkan log

Ketika Outpost terputus dari Wilayah AWS yang terkait dengannya, Kubernetes cluster kemungkinan akan terus bekerja secara normal. Namun, jika klaster tidak berfungsi dengan baik, ikuti langkah-langkah pemecahan masalah. Mempersiapkan pemutusan jaringan Jika Anda mengalami masalah lain, hubungiAWS Support. AWS Supportdapat memandu Anda mengunduh dan menjalankan alat pengumpulan log. Dengan begitu, Anda dapat mengumpulkan log dari instance pesawat kontrol

Kubernetes cluster Anda dan mengirimkannya ke AWS Support dukungan untuk penyelidikan lebih lanjut.

Instans pesawat kontrol tidak dapat dijangkau melalui AWS Systems Manager

Jika instans bidang kontrol Amazon EKS tidak dapat dijangkau melalui (Systems AWS Systems Manager Manager), Amazon EKS menampilkan error berikut untuk klaster Anda.

```
Amazon EKS control plane instances are not reachable through SSM. Please verify your
 SSM and network configuration, and reference the EKS on Outposts troubleshooting
 documentation.
```

Untuk mengatasi masalah ini, pastikan bahwa VPC dan subnet Anda memenuhi persyaratan Persyaratan dan pertimbangan VPC klaster lokal dan subnet Amazon EKS dan bahwa Anda menyelesaikan langkah-langkah dalam Menyiapkan Pengelola Sesi di Panduan Pengguna. AWS Systems Manager

Meluncurkan node Amazon Linux yang dikelola sendiri di Outpost

Topik ini menjelaskan bagaimana Anda dapat meluncurkan grup Auto Scaling dari node Amazon Linux di Outpost yang mendaftar dengan cluster Amazon EKS Anda. Cluster dapat berada di AWS Cloud atau di Pos Luar.

Prasyarat

- Pos terdepan yang ada. Untuk informasi lebih lanjut, lihat Apa yang dimaksud dengan AWS Outposts.
- Sebuah klaster Amazon EKS yang sudah ada. Untuk menyebarkan cluster di AWS Cloud, lihatMembuat klaster Amazon EKS. Untuk menyebarkan cluster di Outpost, lihat. Cluster lokal untuk Amazon EKS di AWS Outposts
- Misalkan Anda membuat node Anda di cluster di AWS Cloud dan Anda memiliki subnet di Wilayah AWS mana Anda memiliki AWS Outposts, AWS Wavelength, atau AWS Local Zones diaktifkan. Kemudian, subnet tersebut seharusnya tidak diteruskan saat Anda membuat cluster Anda. Jika Anda membuat node di cluster di Outpost, Anda pasti telah melewati subnet Outpost saat membuat cluster Anda.
- (Direkomendasikan untuk klaster di AWS Cloud) Amazon VPC CNI plugin for Kubernetes Add-on dikonfigurasi dengan peran IAM sendiri yang memiliki kebijakan IAM yang diperlukan yang

melekat padanya. Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA). Cluster lokal tidak mendukung peran IAM untuk akun layanan.

Anda dapat membuat grup node Amazon Linux yang dikelola sendiri dengan `eksctl` atau AWS Management Console (dengan AWS CloudFormation templat). Anda juga dapat menggunakan Terraform.

eksctl

Prasyarat

Versi `0.175.0` atau yang lebih baru dari alat baris `eksctl` perintah yang diinstal pada perangkat Anda atau AWS CloudShell. Untuk menginstal atau memperbarui`eksctl`, lihat Instalasi dalam `eksctl` dokumentasi.

Untuk meluncurkan Linux node yang dikelola sendiri menggunakan **`eksctl`**

1. Jika klaster Anda aktif AWS Cloud dan kebijakan IAM terkelola Amazoneks_CNI_Policy dilampirkan ke IAM role simpul Amazon EKS Anda, kami sarankan untuk menetapkannya ke peran IAM yang Anda kaitkan ke akun layanan sebagai gantinya. Kubernetes `aws-node` Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA). Jika klaster Anda berada di Outpost Anda, kebijakan harus dilampirkan ke peran node Anda.
2. Perintah berikutnya membuat grup simpul dalam klaster yang ada. Cluster harus dibuat menggunakan`eksctl`. Ganti *`al-nodes`* dengan nama untuk grup node Anda. Nama grup node tidak boleh lebih dari 63 karakter. Itu harus dimulai dengan huruf atau digit, tetapi juga dapat menyertakan tanda hubung dan garis bawah untuk karakter yang tersisa. Ganti *`my-cluster`* dengan nama klaster Anda. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter. Jika cluster Anda ada di Outpost, ganti *`id`* dengan ID subnet Outpost. Jika klaster Anda ada di AWS Cloud, ganti *`id`* dengan ID subnet yang tidak Anda tentukan saat membuat klaster. Ganti *`instance-type`* dengan jenis instance yang didukung oleh Outpost Anda. Ganti sisanya *`example values`* dengan nilai Anda sendiri. Node dibuat dengan Kubernetes versi yang sama dengan bidang kontrol, secara default.

   Ganti *`instance-type`* dengan jenis instance yang tersedia di Outpost Anda.

Ganti *my-key* dengan nama pasangan kunci atau kunci publik Amazon EC2 Anda. Kunci ini digunakan untuk SSH ke simpul Anda setelah diluncurkan. Jika Anda belum memiliki pasangan kunci Amazon EC2, Anda dapat membuatnya di AWS Management Console. Untuk informasi selengkapnya, lihat Kunci Pasangan Amazon EC2 dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

Buat grup simpul Anda dengan perintah berikut.

```
eksctl create nodegroup --cluster my-cluster --name al-nodes --node-
type instance-type \
    --nodes 3 --nodes-min 1 --nodes-max 4 --managed=false --node-volume-type gp2
 --subnet-ids subnet-id
```

Jika klaster Anda diterapkan di AWS Cloud:

- Grup node yang Anda gunakan dapat menetapkan `IPv4` alamat Pods dari CIDR blok yang berbeda dari instance. Untuk informasi selengkapnya, lihat Jaringan khusus untuk pod.
- Grup node yang Anda terapkan tidak memerlukan akses internet keluar. Untuk informasi selengkapnya, lihat Persyaratan klaster pribadi.

Untuk daftar lengkap semua opsi dan default yang tersedia, lihat AWS Outposts Support dalam dokumentasi. `eksctl`

Jika node gagal bergabung dengan cluster, maka lihat Simpul gagal untuk bergabung dengan klaster masuk Pemecahan masalah Amazon EKS dan Tidak dapat menggabungkan node ke cluster masukMemecahkan masalah kluster lokal untuk Amazon EKS di AWS Outposts.

Contoh output adalah sebagai berikut. Beberapa baris adalah output sementara node dibuat. Salah satu baris terakhir dari output adalah baris contoh berikutnya.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Opsional) Menyebarkan aplikasi sampel untuk menguji cluster dan Linux node Anda.

AWS Management Console

Langkah 1: Untuk meluncurkan node Amazon Linux yang dikelola sendiri menggunakan AWS Management Console

1. Unduh versi terbaru dari AWS CloudFormation template.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/amazon-eks-nodegroup.yaml
```

2. Buka AWS CloudFormation konsol di https://console.aws.amazon.com/cloudformation.
3. Pilih Buat tumpukan dan kemudian pilih Dengan sumber daya baru (standar).
4. Untuk Menentukan templat, pilih Unggah sebuah file templat dan kemudian pilih Pilih file. Pilih `amazon-eks-nodegroup.yaml` file yang Anda unduh pada langkah sebelumnya dan kemudian pilih Berikutnya.
5. Pada halaman Tentukan detail tumpukan, masukkan parameter berikut yang sesuai, lalu pilih Berikutnya:

   - Nama tumpukan: Pilih nama tumpukan untuk tumpukan AWS CloudFormation Anda. Misalnya, Anda bisa menyebutnya***al-nodes***. Nama hanya dapat berisi karakter alfanumerik (peka huruf besar/kecil) dan tanda hubung. Itu harus dimulai dengan karakter alfabet dan tidak boleh lebih dari 100 karakter.
   - ClusterName: Masukkan nama cluster Anda. Jika nama ini tidak cocok dengan nama cluster Anda, node Anda tidak dapat bergabung dengan cluster.
   - ClusterControlPlaneSecurityGroup: Pilih SecurityGroupsnilai dari AWS CloudFormation output yang Anda hasilkan saat Anda membuat VPC Anda.

     Langkah-langkah berikut menunjukkan satu operasi untuk mengambil grup yang berlaku.

     1. Buka konsol Amazon EKS di https://console.aws.amazon.com/eks/home#/clusters.
     2. Pilih nama cluster.
     3. Pilih tab Jaringan.
     4. Gunakan nilai grup keamanan tambahan sebagai referensi saat memilih dari daftar ClusterControlPlaneSecurityGrouptarik-turun.
   - NodeGroupName: Masukkan nama untuk grup node Anda. Nama ini dapat digunakan nanti untuk mengidentifikasi grup node Auto Scaling yang dibuat untuk node Anda.

- NodeAutoScalingGroupMinSize: Masukkan jumlah minimum node yang dapat diskalakan oleh grup Auto Scaling node Anda.
- NodeAutoScalingGroupDesiredCapacity: Masukkan jumlah node yang diinginkan untuk diskalakan saat tumpukan Anda dibuat.
- NodeAutoScalingGroupMaxSize: Masukkan jumlah maksimum node yang dapat diskalakan oleh grup Auto Scaling node Anda.
- NodeInstanceType: Pilih jenis instance untuk node Anda. Jika cluster Anda berjalan di AWS Cloud, maka untuk informasi lebih lanjut, lihatMemilih jenis instans Amazon EC2. Jika cluster Anda berjalan di Outpost, maka Anda hanya dapat memilih jenis instance yang tersedia di Outpost Anda.
- NodeImageIdSSMParam: Diisi sebelumnya dengan parameter Amazon EC2 Systems Manager dari AMI Amazon EKS yang dioptimalkan baru-baru ini untuk versi variabel. Kubernetes Untuk menggunakan versi Kubernetes minor berbeda yang didukung dengan Amazon EKS, ganti *1.XX* dengan versi lain yang didukung. Sebaiknya tentukan Kubernetes versi yang sama dengan cluster Anda.

  Untuk menggunakan AMI akselerasi Amazon EKS yang dioptimalkan, ganti *amazon-linux-2* dengan**amazon-linux-2-gpu**. Untuk menggunakan AMI Arm yang dioptimalkan Amazon EKS, ganti *amazon-linux-2* dengan**amazon-linux-2-arm64**.

  > **Note**
  >
  > Node Amazon EKS AMI didasarkan pada Amazon Linux. Anda dapat melacak peristiwa keamanan atau privasi untuk Amazon Linux 2 di Pusat Keamanan Amazon Linux atau berlangganan ke Umpan RSS yang terkait. Kejadian keamanan dan privasi mencakup gambaran umum mengenai masalah, paket apa yang terpengaruh, dan cara memperbarui instans Anda untuk memperbaiki masalah tersebut.

- NodeImageId: (Opsional) Jika Anda menggunakan AMI kustom Anda sendiri (bukan AMI yang dioptimalkan Amazon EKS), masukkan ID AMI node untuk Anda Wilayah AWS. Jika Anda menentukan nilai di sini, itu akan mengganti nilai apa pun di bidang NodeImageIdSSMParam.
- NodeVolumeSize: Tentukan ukuran volume root untuk node Anda, di GiB.
- NodeVolumeType: Tentukan jenis volume root untuk node Anda.

- KeyName: Masukkan nama key pair Amazon EC2 SSH yang dapat Anda gunakan untuk terhubung menggunakan SSH ke node Anda setelah diluncurkan. Jika Anda belum memiliki pasangan kunci Amazon EC2, maka Anda dapat membuatnya di AWS Management Console. Untuk informasi selengkapnya, lihat Pasangan kunci Amazon EC2 dalam Panduan Pengguna Amazon EC2 untuk Instans Linux.

  > **Note**
  >
  > Jika Anda tidak menyediakan key pair di sini, pembuatan AWS CloudFormation stack gagal.

- BootstrapArguments: Ada beberapa argumen opsional yang dapat Anda berikan ke node Anda. Untuk informasi lebih lanjut, lihat informasi penggunaan skrip bootstrap diGitHub. Jika Anda menambahkan node ke cluster yang tidak memiliki koneksi internet ingress dan egress (juga dikenal sebagai cluster pribadi), maka Anda harus memberikan argumen bootstrap berikut (sebagai satu baris).

  ```
  --b64-cluster-ca ${CLUSTER_CA} --apiserver-endpoint https://
  ${APISERVER_ENDPOINT} --enable-local-outpost true --cluster-id ${CLUSTER_ID}
  ```

- DisableImDSv1: Secara default, setiap node mendukung Instance Metadata Service Version 1 (IMDSv1) dan IMDSv2. Anda dapat menonaktifkan IMDSv1. Untuk mencegah node future dan Pods dalam grup node menggunakan IMDSv1, atur disableImDSv1 ke true. Untuk informasi selengkapnya tentang IMDS, lihat Mengonfigurasi layanan metadata instans. Untuk informasi selengkapnya tentang membatasi akses ke node Anda, lihat Membatasi akses ke profil instance yang ditetapkan ke node pekerja.
- VpcId: Masukkan ID untuk VPC yang Anda buat. Sebelum memilih VPC, tinjau. Persyaratan dan pertimbangan VPC
- Subnet: Jika cluster Anda berada di Outpost, maka pilih setidaknya satu subnet pribadi di VPC Anda. Sebelum memilih subnet, tinjau persyaratan dan pertimbangan Subnet. Anda dapat melihat subnet mana yang bersifat pribadi dengan membuka setiap subnet link dari tab Networking cluster Anda.

6. Pilih pilihan yang Anda inginkan di halaman Configure stack options, lalu pilih Next.
7. Pilih kotak centang di sebelah kiri Saya mengakui yang AWS CloudFormation mungkin membuat sumber daya IAM. , dan kemudian pilih Buat tumpukan.
8. Setelah tumpukan Anda selesai dibuat, pilih tumpukan di konsol dan pilih Outputs.

9. Rekam NodeInstanceRoleuntuk grup node yang telah dibuat. Anda memerlukan ini saat mengonfigurasi simpul Amazon EKS Anda.

Langkah 2: Untuk mengaktifkan node untuk bergabung dengan cluster Anda

1. Periksa untuk melihat apakah Anda sudah memiliki `aws-authConfigMap`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Jika Anda ditampilkan `aws-authConfigMap`, maka perbarui sesuai kebutuhan.

   a. Buka `ConfigMap` untuk mengedit.

```
kubectl edit -n kube-system configmap/aws-auth
```

   b. Tambahkan `mapRoles` entri baru sesuai kebutuhan. Tetapkan `rolearn` nilai ke NodeInstanceRolenilai yang Anda catat dalam prosedur sebelumnya.

```
[...]
data:
  mapRoles: |
    - rolearn: <ARN of instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
[...]
```

   c. Simpan file dan keluar dari editor teks Anda.

3. Jika Anda menerima kesalahan yang menyatakan "`Error from server (NotFound): configmaps "aws-auth" not found`, maka terapkan stok`ConfigMap`.

   a. Unduh peta konfigurasi.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/aws-auth-cm.yaml
```

   b. Dalam `aws-auth-cm.yaml` file, atur `rolearn` ke NodeInstanceRolenilai yang Anda rekam dalam prosedur sebelumnya. Anda dapat melakukan ini dengan editor teks, atau dengan mengganti *`my-node-instance-role`* dan menjalankan perintah berikut:

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-role|' aws-auth-cm.yaml
```

c. Terapkan konfigurasi. Perintah ini mungkin memerlukan waktu beberapa menit untuk diselesaikan.

```
kubectl apply -f aws-auth-cm.yaml
```

4. Perhatikan status simpul Anda dan tunggu sampai simpul mencapai Status `Ready`.

```
kubectl get nodes --watch
```

Masukkan `Ctrl+C` untuk kembali ke prompt shell.

> **Note**
>
> Jika Anda menerima kesalahan otorisasi atau jenis sumber daya, lihat Tidak sah atau akses ditolak (`kubectl`) di topik pemecahan masalah.

Jika node gagal bergabung dengan cluster, maka lihat Simpul gagal untuk bergabung dengan klaster masuk Pemecahan masalah Amazon EKS dan Tidak dapat menggabungkan node ke cluster masukMemecahkan masalah kluster lokal untuk Amazon EKS di AWS Outposts.

5. Instal driver Amazon EBS CSI. Untuk informasi selengkapnya, lihat Instalasi di GitHub. Di bagian Siapkan izin driver, pastikan untuk mengikuti instruksi untuk opsi Menggunakan profil instans IAM. Anda harus menggunakan kelas `gp2` penyimpanan. Kelas `gp3` penyimpanan tidak didukung.

   Untuk membuat kelas `gp2` penyimpanan di cluster Anda, selesaikan langkah-langkah berikut.

   1. Jalankan perintah berikut untuk membuat `gp2-storage-class.yaml` file.

```
cat >gp2-storage-class.yaml <<EOF
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
  name: ebs-sc
```

```
provisioner: ebs.csi.aws.com
volumeBindingMode: WaitForFirstConsumer
parameters:
  type: gp2
  encrypted: "true"
allowVolumeExpansion: true
EOF
```

2. Menerapkan manifes ke klaster Anda.

```
kubectl apply -f gp2-storage-class.yaml
```

6. (Hanya node GPU) Jika Anda memilih jenis instans GPU dan AMI akselerasi Amazon EKS yang dioptimalkan, Anda harus menerapkan plugin perangkat NVIDIA untuk Kubernetes sebagai DaemonSet di cluster Anda. Ganti *vX.X.X* dengan s-device-plugin versi NVIDIA/K8 yang Anda inginkan sebelum menjalankan perintah berikut.

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

Langkah 3: Tindakan tambahan

1. (Opsional) Menyebarkan aplikasi sampel untuk menguji cluster dan Linux node Anda.
2. Jika cluster Anda digunakan di Outpost, lewati langkah ini. Jika klaster Anda digunakan pada AWS Cloud, informasi berikut adalah opsional. Jika kebijakan IAM terkelola Amazoneks_CNI_Policy dilampirkan ke IAM role simpul Amazon EKS Anda, sebaiknya tetapkan ke peran IAM yang Anda kaitkan ke akun layanan. Kubernetes `aws-node` Untuk informasi selengkapnya, lihat Mengkonfigurasi Amazon VPC CNI plugin for Kubernetes untuk menggunakan peran IAM untuk akun layanan (IRSA).

Proyek Terkait

Proyek sumber terbuka ini memperluas fungsionalitas Kubernetes cluster yang berjalan di atau di luarAWS, termasuk cluster yang dikelola oleh Amazon EKS.

Alat manajemen

Alat manajemen terkait untuk Amazon EKS dan Kubernetes cluster.

eksctl

`eksctl` adalah alat CLI sederhana untuk membuat klaster di Amazon EKS.

- URL Proyek
- Dokumentasi proyek
- AWS blog sumber terbuka: eksctl: klaster Amazon EKS dengan satu perintah

AWSpengontrol untuk Kubernetes

Dengan AWS Controllers forKubernetes, Anda dapat membuat dan mengelola AWS sumber daya langsung dari Kubernetes cluster Anda.

- URL Proyek
- AWSblog open source: operator AWS layanan untuk Kubernetes saat ini tersedia

Flux CD

Flux adalah alat yang dapat Anda gunakan untuk mengelola konfigurasi klaster Anda dengan menggunakan Git. Ini menggunakan operator di cluster untuk memicu penerapan di dalam. Kubernetes Untuk informasi selengkapnya tentang operator, lihat OperatorHub.io diGitHub.

- URL Proyek
- Dokumentasi proyek

CDK untuk Kubernetes

Dengan CDK for Kubernetes (cdk8s), Anda dapat menentukan Kubernetes aplikasi dan komponen menggunakan bahasa pemrograman yang sudah dikenal. Aplikasi cdk8s mensintesis ke dalam Kubernetes manifes standar, yang dapat diterapkan ke cluster mana pun. Kubernetes

- URL Proyek
- Dokumentasi proyek
- AWSblog container: Memperkenalkan cdk8s+: API berbasis niat untuk objek Kubernetes

Jaringan

Proyek jaringan terkait untuk Amazon EKS dan Kubernetes cluster.

Amazon VPC CNI plugin for Kubernetes

Amazon EKS mendukung jaringan VPC asli melalui file. Amazon VPC CNI plugin for Kubernetes Plugin memberikan alamat IP dari VPC Anda ke masing-masing. Pod

- URL Proyek
- Dokumentasi proyek

AWS Load Balancer Controlleruntuk Kubernetes

AWS Load Balancer ControllerIni membantu mengelola AWS Elastic Load Balancers untuk sebuah Kubernetes cluster. Ini memenuhi sumber daya Kubernetes Ingress dengan menyediakan Application Load Balancers. AWS Ini memenuhi sumber daya Kubernetes layanan dengan menyediakan AWS Network Load Balancers.

- URL Proyek
- Dokumentasi proyek

ExternalDNS

ExternalDNS menyinkronkan Kubernetes layanan yang terbuka dan masuk ke penyedia DNS termasuk Amazon Route 53 dan Service Discovery. AWS

- URL Proyek
- Dokumentasi proyek

Machine learning

Proyek pembelajaran mesin terkait untuk Amazon EKS dan Kubernetes cluster.

Kubeflow

Toolkit pembelajaran mesin untukKubernetes.

- URL Proyek
- Dokumentasi proyek
- Blog sumber terbuka AWS: Kubeflow di Amazon EKS

Auto Scaling

Proyek penskalaan otomatis terkait untuk Amazon EKS dan Kubernetes cluster.

Autoscaler klaster

Cluster Autoscaler adalah alat yang secara otomatis menyesuaikan ukuran Kubernetes cluster berdasarkan CPU dan tekanan memori.

- URL Proyek
- Dokumentasi proyek
- Lokakarya Amazon EKS: https://www.eksworkshop.com/

Eskalator

Eskalator adalah batch atau pekerjaan yang dioptimalkan autoscaler horizontal untuk. Kubernetes

- URL Proyek
- Dokumentasi proyek

Pemantauan

Proyek pemantauan terkait untuk Amazon EKS dan Kubernetes cluster.

Prometheus

Prometheusadalah toolkit pemantauan dan peringatan sistem sumber terbuka.

- URL Proyek
- Dokumentasi proyek
- Lokakarya Amazon EKS: https://eksworkshop.com/intermediate/240_monitoring/

Integrasi berkelanjutan/deployment berkesinambungan

Proyek CI/CD terkait untuk Amazon EKS dan Kubernetes cluster.

Jenkins X

Solusi CI/CD untuk aplikasi cloud modern di Amazon EKS dan Kubernetes cluster.

- URL Proyek
- Dokumentasi proyek

Fitur baru dan panduan (roadmap) Amazon EKS

Anda dapat mempelajari fitur baru Amazon EKS dengan menggulir ke feed Apa yang Baru pada halaman Apa yang Baru AWS. Anda juga dapat meninjau petunjuk (roadmap) diGitHub, yang memungkinkan Anda mengetahui tentang fitur dan prioritas yang akan datang sehingga Anda dapat merencanakan bagaimana Anda ingin menggunakan Amazon EKS di future. Anda dapat memberikan umpan balik langsung kepada kami tentang prioritas panduan (roadmap).

Riwayat dokumen untuk Amazon EKS

Tabel berikut menjelaskan pembaruan utama dan fitur baru untuk Panduan Pengguna Amazon EKS. Kami juga rutin memperbarui dokumentasi untuk menjawab umpan balik yang Anda kirimkan kepada kami.

Perubahan	Deskripsi	Tanggal
CloudWatch Container Insights untuk Windows	Amazon CloudWatch Observability OperatorAdd-on sekarang juga memungkinkan Container Insights node Windows pekerja di cluster.	April 10, 2024
Kubernetes	Menambahkan topik konsep Kubernetes baru.	April 5, 2024
Restrukturisasi Akses dan Konten IAM	Pindahkan halaman yang ada terkait dengan topik akses dan IAM, seperti peta konfigurasi auth, entri akses, ID Pod, dan IRSA ke bagian baru. Merevisi isi ikhtisar.	April 2, 2024
Dukungan Bottlerocket OS untuk driver Amazon S3 CSI	Driver Mountpoint untuk Amazon S3 CSI sekarang kompatibel dengan. Bottleroc ket	Maret 13, 2024
AWS pembaruan kebijakan terkelola - Pembaruan ke kebijakan yang ada	Amazon EKS memperbarui kebijakan AWS terkelola yang ada.	Maret 4, 2024
https://docs.aws.amazon.com/eks/latest/userguide/eks-optimized-ami.html#al2023	Amazon Linux 2023 (AL2023) adalah sistem operasi berbasis Linux baru yang dirancang untuk menyediakan lingkungan yang aman, stabil,	Februari 29, 2024

	dan berkinerja tinggi untuk aplikasi cloud Anda.	
EKS Pod Identity dan IRSA mendukung sidecar di Kubernetes1.29	Di Kubernetes1.29, wadah sespan tersedia di cluster Amazon EKS. Kontainer sidecar didukung dengan peran IAM untuk akun layanan atau EKS Pod Identity. Untuk informasi selengkapnya tentang sespan, lihat Kontainer Sidecar di dokumentasi. Kubernetes	Februari 26, 2024
Kubernetesversi 1.29	Menambahkan 1.29 dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	23 Januari 2024
Rilis lengkap: Amazon EKS Extended Support untuk Kubernetes versi	Dukungan Kubernetes versi diperpanjang memungkin kan Anda untuk tetap pada Kubernetes versi tertentu selama lebih dari 14 bulan.	Januari 16, 2024

Deteksi kesehatan klaster Amazon EKS di AWS Cloud	Amazon EKS mendeteksi masalah dengan kluster Amazon EKS Anda dan infrastruktur prasyarat klaster dalam kesehatan klaster. Anda dapat melihat masalah dengan kluster EKS Anda di AWS Management Console dan di `health` cluster di EKS API. Masalah ini merupakan tambahan dari masalah yang terdeteksi oleh dan ditampilkan oleh konsol. Sebelumnya, kesehatan cluster hanya tersedia untuk cluster lokal di AWS Outposts.	28 Desember 2023
Wilayah AWS	Amazon EKS sekarang tersedia di Kanada Barat (Calgary) (`ca-west-1`). Wilayah AWS	Desember 20, 2023
Wawasan cluster	Anda sekarang bisa mendapatkan rekomendasi pada klaster Anda berdasarkan pemeriksaan berulang.	Desember 20, 2023

Sekarang Anda dapat memberikan peran IAM dan akses pengguna ke klaster Anda menggunakan entri akses	Sebelum pengenalan entri akses, Anda memberikan peran IAM dan akses pengguna ke klaster Anda dengan menambahkan entri ke. `aws-auth` `ConfigMap` Sekarang setiap cluster memiliki mode akses, dan Anda dapat beralih menggunakan entri akses pada jadwal Anda. Setelah beralih mode, Anda dapat menambahkan pengguna dengan menambahkan entri akses di AWS CLI, AWS CloudFormation, dan AWS SDK.	18 Desember 2023
Pembaruan versi platform Amazon EKS	Ini adalah versi platform baru dengan perbaikan dan peningkatan keamanan. Ini termasuk versi patch baru dari Kubernetes`1.28.4`,,`1.27.8`, `1.26.11` dan. `1.25.16`	Desember 12, 2023
Mountpointuntuk driver Amazon S3 CSI	Anda sekarang dapat menginstal driver Mountpoint untuk Amazon S3 CSI di kluster Amazon EKS.	27 November 2023
Aktifkan Prometheus metrik saat membuat klaster	Di AWS Management Console, Anda sekarang dapat mengaktifkan Prometheus metrik saat membuat cluster. Anda juga dapat melihat detail Prometheus scraper di tab Observability.	26 November 2023

Identitas Amazon EKS Pod	Amazon EKS Pod Identities mengaitkan peran IAM dengan akun Kubernetes layanan. Dengan fitur ini, Anda tidak perlu lagi memberikan izin tambahan ke peran IAM node. Dengan cara ini, Pods pada node itu dapat memanggil AWS API. Tidak seperti peran IAM untuk akun layanan, EKS Pod Identities sepenuhnya berada di dalam EKS; Anda tidak memerlukan penyedia OIDC identitas.	26 November 2023
AWS pembaruan kebijakan terkelola - Pembaruan ke kebijakan yang ada	Amazon EKS memperbarui kebijakan AWS terkelola yang ada.	26 November 2023
Pengontrol snapshot CSI	Anda sekarang dapat menginstal pengontrol snapshot CSI untuk digunakan dengan driver CSI yang kompatibel, seperti driver Amazon EBS CSI.	17 November 2023
Penulisan ulang topik Operator ADOT	Dukungan add-on Amazon EKS untuk bagian Operator ADOT berlebihan dengan Distro untuk AWS dokumenta si. OpenTelemetry Kami memigrasikan informasi penting yang tersisa ke sumber daya itu untuk mengurangi informasi yang ketinggalan zaman dan tidak konsisten.	14 November 2023

CoreDNSDukungan add-on EKS untuk metrik Prometheus	`v1.8.7-eksbuild.8` Versi`v1.10.1-eksbuild.5` ,`v1.9.3-eksbuild.9` , dan dari add-on EKS untuk CoreDNS mengekspos port yang CoreDNS menerbitkan metrik ke, dalam layanan. `kube-dns` Ini membuatnya lebih mudah untuk memasukkan CoreDNS metrik dalam sistem pemantauan Anda.	10 November 2023
Pengaya Operator CloudWatch Observabilitas Amazon EKS	Menambahkan halaman Operator CloudWatch Observabilitas Amazon EKS.	6 November 2023
Blok Kapasitas untuk instans P5 yang dikelola sendiri di AS Timur (Ohio)	Di US East (Ohio), Anda sekarang dapat menggunakan Blok Kapasitas untuk instans P5 yang dikelola sendiri.	31 Oktober 2023
Cluster mendukung modifikasi subnet dan grup keamanan	Anda dapat memperbarui cluster untuk mengubah subnet dan grup keamanan yang digunakan cluster. Anda dapat memperbarui dari AWS Management Console, versi terbaru dari AWS CLI, AWS CloudFormation, dan `eksctl` versi `v0.164.0-rc.0` atau yang lebih baru. Anda mungkin perlu melakukan ini untuk menyediakan subnet dengan lebih banyak alamat IP yang tersedia agar berhasil meningkatkan versi cluster.	24 Oktober 2023

Peran cluster dan peran grup node terkelola mendukung AWS Identity and Access Management kebijakan yang dikelola pelanggan	Anda dapat menggunakan kebijakan IAM khusus pada peran klaster, bukan kebijakan `AmazonEKSClusterPolicy` AWS terkelola. Selain itu, Anda dapat menggunakan kebijakan IAM khusus pada peran node dalam grup node terkelola, bukan kebijakan `AmazonEKSWorkerNodePolicy` AWS terkelola. Lakukan ini untuk membuat kebijakan dengan hak istimewa paling sedikit untuk memenuhi persyaratan kepatuhan yang ketat.	23 Oktober 2023
Perbaiki tautan ke instalasi eksctl	Perbaiki link install untuk eksctl setelah halaman dipindahkan.	Oktober 6, 2023
Rilis pratinjau: Amazon EKS Extended Support untuk Kubernetes versi	Dukungan Kubernetes versi diperpanjang memungkinkan Anda untuk tetap pada Kubernetes versi tertentu selama lebih dari 14 bulan.	4 Oktober 2023
Hapus referensi ke AWS App Mesh integrasi	Integrasi Amazon EKS dengan AWS App Mesh peninggalan hanya untuk pelanggan App Mesh yang sudah ada.	September 29, 2023
Kubernetesversi `1.28`	Menambahkan `1.28` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	26 September 2023

Cluster yang ada mendukung penegakan kebijakan Kubernetes jaringan di Amazon VPC CNI plugin for Kubernetes	Anda dapat menggunakan kebijakan Kubernetes jaringan di kluster yang ada dengan Amazon VPC CNI plugin for Kubernetes, alih-alih memerlukan solusi pihak ketiga.	15 September 2023
CoreDNSAdd-on Amazon EKS mendukung modifikasi PDB	Anda dapat `PodDisruptionBudget` memodifikasi add-on EKS untuk CoreDNS dalam versi `v1.9.3-eksbuild.7` dan yang lebih baru `v1.10.1-eksbuild.4` dan yang lebih baru.	15 September 2023
Dukungan Amazon EKS untuk subnet bersama	Persyaratan dan pertimbangan subnet Bersama Baru untuk membuat kluster Amazon EKS di subnet bersama.	7 September 2023
Pembaruan untuk Apa itu Amazon EKS?	Menambahkan kasus penggunaan umum baru dan topik Arsitektur. Segarkan topik lainnya.	September 6, 2023
Kubernetespenegakan kebijakan jaringan di Amazon VPC CNI plugin for Kubernetes	Anda dapat menggunakan kebijakan Kubernetes jaringan dengan Amazon VPC CNI plugin for Kubernetes, alih-alih memerlukan solusi pihak ketiga.	29 Agustus 2023
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Israel (Tel Aviv) (`il-central-1`) Wilayah AWS.	1 Agustus 2023

Penyimpanan fana yang dapat dikonfigurasi untuk Fargate	Anda dapat meningkatkan jumlah total penyimpanan sementara untuk setiap Pod berjalan di Amazon EKS Fargate.	31 Juli 2023
Dukungan add-on untuk driver Amazon EFS CSI	Anda sekarang dapat menggunakan AWS Management Console, AWS CLI, dan API untuk mengelola driver Amazon EFS CSI.	26 Juli 2023
AWS pembaruan kebijakan terkelola - Kebijakan baru	Amazon EKS menambahkan kebijakan AWS terkelola baru.	26 Juli 2023
Kubernetespembaruan versi untuk 1.27, 1.26, 1.25, dan 1.24 sekarang tersedia untuk cluster lokal di AWS Outposts	Kubernetespembaruan versi ke 1.27.3, 1.26.6, 1.25.11, dan 1.24.15 sekarang tersedia untuk cluster lokal di AWS Outposts	Juli 20, 2023
Dukungan awalan IP untuk node Windows	Menetapkan awalan IP ke node Anda dapat memungkinkan Anda untuk meng-host jumlah node yang jauh lebih tinggi daripada yang Anda bisa saat menetapkan alamat IP sekunder individual ke node Anda. Pods	6 Juli 2023
Amazon FSx untuk driver OpenZFS CSI	Anda sekarang dapat menginstal Amazon FSx untuk driver OpenZFS CSI di cluster Amazon EKS.	Juni 30, 2023

Podspada node Linux dalam `IPv4` cluster sekarang dapat berkomunikasi dengan `IPv6` endpoint.	Setelah menetapkan alamat IPv6 ke node Anda, Pods '`IPv4`alamat' Anda adalah alamat jaringan yang diterjemahkan ke `IPv6` alamat node yang sedang berjalan.	19 Juni 2023
Windowsgrup simpul terkelola di AWS GovCloud (US) Regions	Dalam AWS GovCloud (US) Regions, grup node terkelola Amazon EKS sekarang dapat menjalankan Windows kontainer.	30 Mei 2023
Kubernetesversi `1.27`	Menambahkan `1.27` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	24 Mei 2023
Kubernetesversi `1.26`	Menambahkan `1.26` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	11 April 2023
Tanpa Domain gMSA	Anda sekarang dapat menggunakan domainless gMSA dengan. Windows Pods	Maret 27, 2023
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Asia Pasifik (Melbourne) (`ap-southeast-4`) Wilayah AWS.	Maret 10, 2023
Driver CSI Cache File Amazon	Anda sekarang dapat menginstal driver Cache File CSI Amazon di kluster Amazon EKS.	3 Maret 2023

Kubernetesversi 1.25 sekarang tersedia untuk cluster lokal di AWS Outposts	Anda sekarang dapat membuat cluster lokal Amazon EKS di Outpost menggunak an Kubernetes versi `1.22` —`1.25`.	1 Maret 2023
Kubernetesversi `1.25`	Menambahkan `1.25` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	22 Februari 2023
AWS pembaruan kebijakan terkelola - Pembaruan ke kebijakan yang ada	Amazon EKS memperbarui kebijakan AWS terkelola yang ada.	7 Februari 2023
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Asia Pasifik (Hyderabad) (`ap-south-2`), Eropa (Zurich) (), dan Eropa (Spanyol`eu-central-2`) (). `eu-south-2` Wilayah AWS	6 Februari 2023
Kubernetesversi `1.21` - sekarang `1.24` tersedia untuk cluster lokal di AWS Outposts.	Anda sekarang dapat membuat cluster lokal Amazon EKS di Outpost menggunak an Kubernetes versi `1.21` —`1.24`. Sebelumnya, hanya versi `1.21` yang tersedia.	Januari 17, 2023
Amazon EKS sekarang mendukung AWS PrivateLink	Anda dapat menggunakan file AWS PrivateLink untuk membuat koneksi pribadi antara VPC Anda dan Amazon EKS.	16 Desember 2022

Dukungan Windows grup node terkelola	Anda sekarang dapat menggunakan Windows untuk grup node terkelola Amazon EKS.	Desember 15, 2022
Add-on Amazon EKS dari vendor perangkat lunak independen sekarang tersedia di AWS Marketplace	Anda sekarang dapat menelusuri dan berlangga nan add-on Amazon EKS dari vendor perangkat lunak independen melalui. AWS Marketplace	28 November 2022
AWS pembaruan kebijakan terkelola - Pembaruan ke kebijakan yang ada	Amazon EKS memperbarui kebijakan AWS terkelola yang ada.	17 November 2022
Kubernetesversi `1.24`	Menambahkan `1.24` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	15 November 2022
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Timur Tengah (UEA) (`me-central-1`) Wilayah AWS.	November 3, 2022
AWS pembaruan kebijakan terkelola - Pembaruan ke kebijakan yang ada	Amazon EKS memperbarui kebijakan AWS terkelola yang ada.	24 Oktober 2022
AWS pembaruan kebijakan terkelola - Pembaruan ke kebijakan yang ada	Amazon EKS memperbarui kebijakan AWS terkelola yang ada.	20 Oktober 2022
Cluster lokal aktif sekarang AWS Outposts tersedia	Anda sekarang dapat membuat cluster lokal Amazon EKS di Outpost.	19 September 2022

Kuota berbasis vCPU Fargate	Fargate beralih dari Pod kuota berbasis ke kuota berbasis vCPU.	September 8, 2022
AWS pembaruan kebijakan terkelola - Pembaruan ke kebijakan yang ada	Amazon EKS memperbarui kebijakan AWS terkelola yang ada.	31 Agustus 2022
Pemantauan biaya	Amazon EKS sekarang mendukungKubecost, yang memungkinkan Anda memantau biaya yang dipecah berdasarkan Kubernetes sumber daya termasukPods, node, ruang nama, dan label.	Agustus 24, 2022
AWS pembaruan kebijakan terkelola - Kebijakan baru	Amazon EKS menambahkan kebijakan AWS terkelola baru.	Agustus 24, 2022
AWS pembaruan kebijakan terkelola - Kebijakan baru	Amazon EKS menambahkan kebijakan AWS terkelola baru.	23 Agustus 2022
Menandai sumber daya untuk penagihan	Menambahkan dukungan tag alokasi biaya yang `aws:eks:cluster-name` dihasilkan untuk semua cluster.	Agustus 16, 2022
Wildcard profil Fargate	Menambahkan dukungan untuk wildcard profil Fargate dalam kriteria pemilih untuk ruang nama, kunci label, dan nilai label.	Agustus 16, 2022
Kubernetesversi `1.23`	Menambahkan `1.23` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	Agustus 11, 2022

Lihat Kubernetes sumber daya di AWS Management Console	Anda sekarang dapat melihat informasi tentang Kubernetes sumber daya yang digunakan ke klaster Anda menggunakan file. AWS Management Console	Mei 3, 2022
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Asia Pasifik (Jakarta) (`ap-southeast-3`) Wilayah AWS.	2 Mei 2022
Halaman observabilitas dan dukungan add-on ADOT	Ditambahkan halaman Observability dan AWS Distro untuk OpenTelemetry (ADOT).	21 April 2022
Kubernetesversi `1.22`	Menambahkan `1.22` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	4 April 2022
AWS pembaruan kebijakan terkelola - Kebijakan baru	Amazon EKS menambahkan kebijakan AWS terkelola baru.	4 April 2022
Menambahkan rincian patch Fargate Pod	Saat memutakhirkan Pods Fargate, Amazon EKS pertama-tama mencoba Pods mengusir berdasarkan anggaran gangguan Anda. Pod Anda dapat membuat aturan acara untuk bereaksi terhadap penggusuran yang gagal sebelum Pods dihapus.	1 April 2022

Rilis penuh: Dukungan add-on untuk driver Amazon EBS CSI	Anda sekarang dapat menggunakan AWS Management Console, AWS CLI, dan API untuk mengelola driver Amazon EBS CSI.	31 Maret 2022
AWS Outposts pembaruan konten	Petunjuk untuk menerapkan kluster Amazon EKS pada AWS Outposts.	22 Maret 2022
AWS pembaruan kebijakan terkelola - Pembaruan ke kebijakan yang ada	Amazon EKS memperbarui kebijakan AWS terkelola yang ada.	Maret 21, 2022
`containerd` Dukungan Windows	Anda sekarang dapat memilih `containerd` runtime untuk Windows node.	Maret 14, 2022
Menambahkan pertimbangan Amazon EKS Connector ke dokumentasi keamanan	Menjelaskan model tanggung jawab bersama yang berkaitan dengan cluster yang terhubung.	25 Februari 2022
Tetapkan `IPv6` alamat ke layanan Pods dan Anda	Anda sekarang dapat membuat klaster `1.21` atau yang lebih baru yang menetapkan `IPv6` alamat ke layanan Pods dan Anda.	6 Januari 2022
AWS pembaruan kebijakan terkelola - Pembaruan ke kebijakan yang ada	Amazon EKS memperbarui kebijakan AWS terkelola yang ada.	13 Desember 2021
Rilis pratinjau: Dukungan add-on untuk driver Amazon EBS CSI	Sekarang Anda dapat melihat pratinjau menggunakan AWS Management Console AWS CLI,, dan API untuk mengelola driver Amazon EBS CSI.	Desember 9, 2021

Dukungan autoscaler Karpenter	Anda sekarang dapat menggunakan proyek open-source Karpenter untuk menskalakan otomatis node Anda.	29 November 2021
Dukungan Kubernetes filter Bit Lancar di logging Fargate	Anda sekarang dapat menggunakan Kubernetes filter Fluent Bit dengan logging Fargate.	November 10, 2021
Windowsdukungan tersedia di bidang kontrol	Windowsdukungan sekarang tersedia di pesawat kontrol Anda. Anda tidak perlu lagi mengaktifkannya di pesawat data Anda.	November 9, 2021
Bottlerocketditambahkan sebagai tipe AMI untuk grup node terkelola	BottlerocketSebelumnya, hanya tersedia sebagai opsi node yang dikelola sendiri. Sekarang dapat dikonfigurasi sebagai grup node terkelola , mengurangi upaya yang diperlukan untuk memenuhi persyaratan kepatuhan node.	28 Oktober 2021
Dukungan driver DL1	AMI Amazon Linux kustom sekarang mendukung beban kerja pembelajaran mendalam untuk Amazon Linux 2. Pengaktifan ini memungkinkan konfigurasi dasar lokal atau cloud generik.	25 Oktober 2021

Dukungan video VT1	AMI Amazon Linux kustom sekarang mendukung VT1 untuk beberapa distribusi. Pengaktifan ini mengiklankan perangkat Xilinx U30 di cluster Amazon EKS Anda.	13 September 2021
Konektor Amazon EKS sekarang tersedia	Anda dapat menggunakan Amazon EKS Connector untuk mendaftarkan dan menghubungkan Kubernetes klaster kesesuaian apa pun ke AWS dan memvisualisasikannya di konsol Amazon EKS.	8 September 2021
Amazon EKS Anywhere sekarang tersedia	Amazon EKS Anywhere adalah opsi penerapan baru untuk Amazon EKS yang dapat Anda gunakan untuk membuat dan mengoperasikan Kubernetes cluster lokal.	8 September 2021
Amazon FSx untuk driver NetApp ONTAP CSI	Menambahkan topik yang merangkum Amazon FSx NetApp untuk driver ONTAP CSI dan memberikan tautan ke referensi lain.	2 September 2021
Grup node terkelola sekarang secara otomatis menghitung maksimum Pods yang disarankan Amazon EKS untuk node	Grup node terkelola sekarang menghitung otomatis maksimum Amazon EKS Pods untuk node yang Anda gunakan tanpa template peluncuran, atau dengan templat peluncuran yang belum Anda tentukan ID AMI.	Agustus 30, 2021

Hapus pengelolaan pengaturan add-on Amazon EKS tanpa menghapus perangkat lunak add-on Amazon EKS	Anda sekarang dapat menghapus add-on Amazon EKS tanpa menghapus perangkat lunak add-on dari cluster Anda.	Agustus 20, 2021
Buat multi-homed menggunakan Pods Multus	Anda sekarang dapat menambahkan beberapa antarmuka jaringan ke Pod menggunakan Multus.	2 Agustus 2021
Tambahkan lebih banyak alamat IP ke node Linux Amazon EC2 Anda	Anda sekarang dapat menambahkan lebih banyak alamat IP secara signifikan ke node Linux Amazon EC2 Anda. Ini berarti Anda dapat menjalankan kepadatan yang lebih tinggi Pods pada setiap node.	27 Juli 2021
`containerd` runtime bootstrap	Amazon EKS yang dioptimalkan Amazon Linux Amazon Amazon Machine Image (AMI) sekarang berisi flag bootstrap yang dapat Anda gunakan untuk mengaktifkan `containerd` runtime di Amazon EKS yang dioptimalkan dan Bottlerocket AMI. Bendera ini tersedia di semua Kubernetes versi AMI yang didukung.	19 Juli 2021
Kubernetesversi `1.21`	Ditambahkan `1.21` dukungan Kubernetes versi.	19 Juli 2021

Ditambahkan topik kebijakan terkelola	Daftar semua kebijakan dan perubahan yang dikelola Amazon EKS IAM yang dibuat sejak 17 Juni 2021.	17 Juni 2021
Gunakan grup keamanan untuk Pods dengan Fargate	Anda sekarang dapat menggunakan grup keamanan untuk Pods dengan Fargate, selain menggunakannya dengan node Amazon EC2.	1 Juni 2021
Ditambahkan CoreDNS dan `kube-proxy` Amazon EKS add-on	Amazon EKS sekarang dapat membantu Anda mengelola add-on `kube-proxy` Amazon EKS CoreDNS dan Amazon untuk cluster Anda.	19 Mei 2021
Kubernetesversi `1.20`	Menambahkan `1.20` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	18 Mei 2021
AWS Load Balancer Controller `r2.2.0`dirilis	Anda sekarang dapat menggunakan AWS Load Balancer Controller untuk membuat Elastic Load Balancers menggunakan instance atau target IP.	14 Mei 2021
Node taints untuk grup node terkelola	Amazon EKS sekarang mendukung penambahan taint simpul ke grup-grup simpul terkelola.	11 Mei 2021
Enkripsi rahasia untuk cluster yang ada	Amazon EKS sekarang mendukung penambahan enkripsi rahasia ke cluster yang ada.	26 Februari 2021

Kubernetesversi `1.19`	Menambahkan `1.19` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	16 Februari 2021
Amazon EKS sekarang mendukung penyedia identitas OpenID Connect (OIDC) sebagai metode untuk mengautentikasi pengguna ke versi `1.16` atau kluster yang lebih baru.	Penyedia identitas OIDC dapat digunakan dengan, atau sebagai pengganti AWS Identity and Access Management (IAM).	12 Februari 2021
Lihat sumber daya node dan beban kerja di AWS Management Console	Anda sekarang dapat melihat detail tentang node terkelola , dikelola sendiri, dan Fargate serta beban kerja yang Anda Kubernetes gunakan di. AWS Management Console	1 Desember 2020
Menerapkan jenis Instance Spot dalam grup node terkelola	Anda sekarang dapat men-deploy beberapa jenis Spot atau Instans Sesuai Permintaa n ke grup simpul terkelola.	1 Desember 2020
Amazon EKS sekarang dapat mengelola add-on khusus untuk klaster Anda	Anda dapat mengelola sendiri add-on, atau mengizinkan Amazon EKS mengontrol peluncuran dan versi add-on melalui Amazon EKS API.	1 Desember 2020

Bagikan ALB di beberapa Ingress	Anda sekarang dapat berbagi AWS Application Load Balancer (ALB) di beberapa Ingresses. Kubernetes Di masa lalu, Anda harus menerapkan ALB terpisah untuk setiap Ingress.	23 Oktober 2020
Dukungan target IP NLB	Anda sekarang dapat menerapkan Network Load Balancer dengan target IP. Ini berarti Anda dapat menggunakan NLB untuk memuat lalu lintas jaringan keseimbangan ke Fargate Pods dan langsung ke Pods yang berjalan di node Amazon EC2.	23 Oktober 2020
Kubernetesversi `1.18`	Menambahkan `1.18` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	13 Oktober 2020
Tentukan blok CIDR khusus untuk penetapan alamat IP Kubernetes layanan.	Anda sekarang dapat menentukan blok CIDR kustom yang Kubernetes menetapkan alamat IP layanan dari.	29 September 2020
Tetapkan grup keamanan ke individu Pods	Anda sekarang dapat mengaitkan grup keamanan yang berbeda dengan beberapa individu Pods yang berjalan di banyak jenis instans Amazon EC2.	9 September 2020

Terapkan Bottlerocket pada node Anda	Anda sekarang dapat menyebarkan node yang menjalankan Bottlerocket.	31 Agustus 2020
Kemampuan untuk meluncurkan node Arm umumnya tersedia	Sekarang Anda dapat meluncurkan simpul Arm di grup simpul terkelola dan dikelola sendiri.	17 Agustus 2020
Templat peluncuran grup node terkelola dan AMI kustom	Sekarang Anda dapat menerapkan grup node terkelola yang menggunakan templat peluncuran Amazon EC2. Peluncuran templat dapat menentukan kustom AMI, jika Anda mau.	17 Agustus 2020
Dukungan EFS untuk AWS Fargate	Anda sekarang dapat menggunakan Amazon EFS dengan AWS Fargate.	17 Agustus 2020
Pembaruan versi platform Amazon EKS	Ini adalah versi platform baru dengan perbaikan dan peningkatan keamanan. Ini termasuk dukungan UDP untuk layanan jenis `LoadBalancer` saat menggunakan Network Load Balancers dengan Kubernetes versi `1.15` atau yang lebih baru. Untuk informasi selengkapnya, lihat masalah Izinkan UDP untuk AWS Network Load Balancer. GitHub	12 Agustus 2020

Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Afrika (Cape Town) (`af-south-1`) dan Eropa (Milan) (`eu-south-1`) Wilayah AWS.	6 Agustus 2020
Metrik penggunaan Fargate	AWS Fargate menyediakan metrik CloudWatch penggunaan yang memberikan visibilitas ke dalam penggunaan sumber daya Fargate On-Demand akun Anda.	3 Agustus 2020
Kubernetesversi `1.17`	Menambahkan `1.17` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	10 Juli 2020
Membuat dan mengelola sumber daya App Mesh dari dalam Kubernetes dengan pengontrol App Mesh untuk Kubernetes	Anda dapat membuat dan mengelola sumber daya App Mesh dari dalamKubernetes. Pengontrol juga secara otomatis menyuntikkan proxy Envoy dan kontainer init ke dalam wadah Pods yang Anda terapkan.	18 Juni 2020
Amazon EKS sekarang mendukung node Amazon EC2 Inf1	Anda dapat menambahkan simpul Inf1 Amazon EC2 ke klaster Anda.	4 Juni 2020
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di AWS GovCloud (AS-Timur) (`us-gov-east-1`) dan AWS GovCloud (AS-Barat) (`us-gov-west-1`). Wilayah AWS	13 Mei, 2020

Kubernetes1.12tidak lagi didukung di Amazon EKS	Kubernetesversi 1.12 tidak lagi didukung di Amazon EKS. Perbarui 1.12 cluster apa pun ke versi 1.13 atau yang lebih baru untuk menghindari gangguan layanan.	12 Mei 2020
Kubernetesversi 1.16	Menambahkan 1.16 dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	30 April 2020
Menambahkan peran AWSServiceRoleForAmazonEKSterkait layanan	Menambahkan peran AWSServiceRoleForAmazonEKSterkait layanan.	16 April 2020
Kubernetesversi 1.15	Menambahkan 1.15 dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	10 Maret 2020
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Beijing (cn-north-1) dan Ningxia (cn-northwest-1) Wilayah AWS.	26 Februari 2020
FSx for Lustre CSI driver	Menambahkan topik untuk menginstal driver FSx for Lustre CSI Kubernetes 1.14 di cluster Amazon EKS.	23 Desember 2019
Membatasi akses jaringan ke titik akhir akses publik klaster	Dengan pembaruan ini, Anda dapat menggunakan Amazon EKS untuk membatasi rentang CIDR yang dapat berkomunikasi ke titik akhir akses publik server API. Kubernetes	20 Desember 2019

Menyelesaikan alamat titik akhir akses pribadi untuk klaster dari luar VPC	Dengan pembaruan ini, Anda dapat menggunakan Amazon EKS untuk menyelesaikan titik akhir akses pribadi server Kubernetes API dari luar VPC.	13 Desember 2019
(Beta) Node instans Amazon EC2 A1 Amazon EC2	Luncurkan simpul instans Amazon EC2 A1 Amazon EC2 yang terdaftar dengan klaster Amazon EKS Anda.	4 Desember 2019
Membuat cluster di AWS Outposts	Amazon EKS sekarang mendukung pembuatan klaster di AWS Outposts.	3 Desember 2019
AWS Fargate di Amazon EKS	KubernetesCluster Amazon EKS sekarang mendukung berjalan Pods di Fargate.	3 Desember 2019
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Kanada (Tengah) (`ca-central-1`) Wilayah AWS.	21 November 2019
Grup simpul terkelola	Grup node terkelola Amazon EKS mengotomatiskan penyediaan dan pengelola an siklus hidup node (instans Amazon EC2) untuk kluster Amazon EKS. Kubernetes	18 November 2019
Pembaruan versi platform Amazon EKS	Versi platform baru untuk ditangani `CVE-2019-11253` .	6 November 2019

Kubernetes1.11tidak lagi didukung di Amazon EKS	Kubernetesversi 1.11 tidak lagi didukung di Amazon EKS. Harap perbarui 1.11 cluster apa pun ke versi 1.12 atau lebih tinggi untuk menghindari gangguan layanan.	4 November 2019
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Amerika Selatan (São Paulo) (sa-east-1). Wilayah AWS	16 Oktober 2019
Windowsdukungan	Cluster Amazon EKS yang menjalankan Kubernetes versi 1.14 sekarang mendukung beban Windows kerja.	7 Oktober 2019
Penskalaan otomatis	Menambahkan chapter untuk membahas beberapa jenis Kubernetes penskalaan otomatis yang didukung pada kluster Amazon EKS.	30 September 2019
KubernetesPembaruan dasbor	Topik yang diperbarui untuk menginstal Kubernetes Dasbor di kluster Amazon EKS untuk menggunakan 2.0 versi beta.	28 September 2019
Pengemudi Amazon EFS CSI	Menambahkan topik untuk menginstal driver Amazon EFS CSI di cluster Kubernetes 1.14 Amazon EKS.	19 September 2019

Parameter Amazon EC2 Systems Manager untuk Amazon EKS yang dioptimalkan AMI ID	Penambahan topik untuk mengambil ID AMI yang dioptimalkan dengan Amazon EKS menggunakan parameter Amazon EC2 Systems Manager. Dengan parameter ini, Anda tidak perlu mencari ID AMI.	18 September 2019
Penandaan sumber daya Amazon EKS	Anda dapat mengelola penandaan kluster Amazon EKS Anda.	16 September 2019
Pengemudi Amazon EBS CSI	Menambahkan topik untuk menginstal driver Amazon EBS CSI di kluster Kubernetes `1.14` Amazon EKS.	9 September 2019
Amazon EKS baru yang dioptimalkan AMI ditambal untuk `CVE-2019-9512` dan `CVE-2019-9514`	Amazon EKS telah memperbarui AMI Amazon EKS yang dioptimalkan untuk mengatasi `CVE-2019-9512` dan `CVE-2019-9514` .	6 September 2019
Mengumumkan penghentian di Amazon EKS Kubernetes `1.11`	Amazon EKS menghentikan dukungan untuk Kubernetes versi `1.11` pada 4 November 2019.	4 September 2019
Kubernetesversi `1.14`	Menambahkan `1.14` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	3 September 2019

Peran IAM untuk akun layanan	Dengan peran IAM untuk akun layanan di kluster Amazon EKS, Anda dapat mengaitkan peran IAM dengan akun layanan. Kubernetes Dengan fitur ini, Anda tidak perlu lagi memberikan izin tambahan ke peran IAM node. Dengan cara ini, Pods pada node itu dapat memanggil AWS API.	3 September 2019
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Timur Tengah (Bahrain) (`me-south-1`) Wilayah AWS.	29 Agustus 2019
Pembaruan versi platform Amazon EKS	Versi platform baru untuk ditangani `CVE-2019-9512` dan `CVE-2019-9514` .	28 Agustus 2019
Pembaruan versi platform Amazon EKS	Versi platform baru untuk ditangani `CVE-2019-11247` dan `CVE-2019-11249` .	5 Agustus 2019
Ekspansi Wilayah Amazon EKS	Amazon EKS sekarang tersedia di Asia Pasifik (Hong Kong) (`ap-east-1`) Wilayah AWS.	31 Juli 2019
Kubernetes`1.10`tidak lagi didukung di Amazon EKS	Kubernetesversi `1.10` tidak lagi didukung di Amazon EKS. Perbarui `1.10` cluster apa pun ke versi `1.11` atau lebih tinggi untuk menghindari gangguan layanan.	30 Juli 2019

Ditambahkan topik pada ALB ingress controller	AWS ALB Ingress Controller for Kubernetes adalah pengontrol yang menyebabkan ALB dibuat saat sumber daya ingress dibuat.	11 Juli 2019
Amazon EKS baru yang dioptimalkan AMI	Menghapus biner `kubectl` yang tidak diperlukan dari AMI.	3 Juli 2019
Kubernetesversi `1.13`	Menambahkan `1.13` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	Selasa, 18 Juni 2019
Amazon EKS baru yang dioptimalkan AMI ditambal `AWS-2019-005`	Amazon EKS telah memperbarui AMI Amazon EKS yang dioptimalkan untuk mengatasi kerentanan yang dijelaskan di `AWS-2019-005` .	17 Juni 2019
Mengumumkan penghentian dukungan di Kubernetes`1.10` Amazon EKS	Amazon EKS berhenti mendukung Kubernetes versi `1.10` pada 22 Juli 2019.	21 Mei 2019
Pembaruan versi platform Amazon EKS	Versi platform baru untuk Kubernetes `1.11` dan `1.10` cluster untuk mendukung nama DNS kustom dalam `kubelet` sertifikat dan meningkatkan `etcd` kinerja.	21 Mei 2019

Perintah AWS CLI`get-token`	`aws eks get-token` Perintah ditambahkan ke AWS CLI. Anda tidak perlu lagi menginstal AWS IAM Authenticator Kubernetes untuk membuat token keamanan klien untuk komunikasi server API cluster. Tingkatkan AWS CLI instalasi Anda ke versi terbaru untuk menggunakan fungsi baru ini. Untuk informasi selengkapnya, lihat Menginstal AWS Command Line Interface dalam Panduan Pengguna AWS Command Line Interface .	10 Mei 2019
Memulai dengan `eksctl`	Panduan memulai ini menjelaskan bagaimana Anda dapat menginstal semua sumber daya yang diperlukan untuk memulai menggunakan Amazon EKS`eksctl`. Ini adalah utilitas baris perintah sederhana untuk membuat dan mengelola Kubernetes cluster di Amazon EKS.	10 Mei 2019

Pembaruan versi platform Amazon EKS	Versi platform baru untuk Kubernetes `1.12` cluster untuk mendukung nama DNS kustom dalam `kubelet` sertifikat dan meningkatkan `etcd` kinerja. Ini memperbai ki bug yang menyebabkan `kubelet` daemon node meminta sertifikat baru setiap beberapa detik.	8 Mei 2019
Prometheustutorial	Menambahkan topik untuk diterapkan Prometheus ke kluster Amazon EKS Anda.	5 April 2019
Amazon EKS mengontrol pencatatan pesawat	Dengan pembaruan ini, Anda bisa mendapatkan log audit dan diagnostik langsung dari panel kontrol Amazon EKS. Anda dapat menggunakan CloudWatch log ini di akun Anda sebagai referensi untuk mengamankan dan menjalank an cluster.	4 April 2019
Kubernetesversi `1.12`	Menambahkan `1.12` dukungan Kubernetes versi untuk cluster baru dan upgrade versi.	28 Maret 2019
Menambahkan panduan memulai App Mesh	Menambahkan dokumentasi untuk memulai dengan App Mesh danKubernetes.	27 Maret 2019

Akses pribadi titik akhir server API Amazon EKS	Menambahkan dokumentasi untuk menonaktifkan akses publik untuk titik akhir server Kubernetes API klaster Amazon EKS Anda.	19 Maret 2019
Menambahkan topik untuk menginstal Server Kubernetes Metrik	Server Kubernetes Metrik adalah agregator data penggunaan sumber daya di klaster Anda.	18 Maret 2019
Ditambahkan daftar proyek open source terkait	Proyek open source ini memperluas fungsionalitas Kubernetes cluster yang berjalan AWS, termasuk cluster yang dikelola oleh Amazon EKS.	15 Maret 2019
Ditambahkan topik untuk menginstal Helm lokal	Manajer `helm` paket untuk Kubernetes membantu Anda menginstal dan mengelola aplikasi di Kubernetes cluster Anda. Topik ini menunjukkan cara menginstal dan menjalankan `helm` dan `tiller` binari secara lokal. Dengan begitu, Anda dapat menginstal dan mengelola bagan menggunakan Helm CLI di sistem lokal Anda.	11 Maret 2019
Pembaruan versi platform Amazon EKS	Versi platform baru yang memperbarui Kubernetes `1.11` kluster Amazon EKS ke tingkat patch `1.11.8` ke alamat `CVE-2019-1002100`.	8 Maret 2019

Peningkatan batas cluster	Amazon EKS telah meningkatkan jumlah cluster yang dapat Anda buat Wilayah AWS dari 3 menjadi 50.	13 Februari 2019
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Eropa (London) (`eu-west-2`), Eropa (Paris) (`eu-west-3`), dan Asia Pasifik (Mumbai) (`ap-south-1`) Wilayah AWS.	13 Februari 2019
Amazon EKS baru yang dioptimalkan AMI ditambal `ALAS-2019-1156`	Amazon EKS telah memperbarui AMI Amazon EKS yang dioptimalkan untuk mengatasi kerentanan yang dijelaskan di `ALAS-2019-1156`.	11 Februari 2019
Amazon EKS baru yang dioptimalkan AMI ditambal `ALAS2-2019-1141`	Amazon EKS telah memperbarui AMI Amazon EKS yang dioptimalkan untuk mengatasi CVE yang direferensikan. `ALAS2-2019-1141`	9 Januari 2019
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Asia Pasifik (Seoul) (`ap-northeast-2`) Wilayah AWS.	9 Januari 2019

Ekspansi wilayah Amazon EKS	Amazon EKS sekarang tersedia dalam tambahan berikut Wilayah AWS: Eropa (Frankfurt) (`eu-central-1`), Asia Pasifik (Tokyo) (`ap-northeast-1`), Asia Pasifik (Singapura) (`ap-southeast-1`), dan Asia Pasifik (Sydney) (`ap-southeast-2`).	19 Desember 2018
Pembaruan kluster Amazon EKS	Menambahkan dokumentasi untuk pembaruan Kubernetes versi cluster Amazon EKS dan penggantian node.	12 Desember 2018
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Eropa (Stockholm) (`eu-north-1`) Wilayah AWS.	11 Desember 2018
Pembaruan versi platform Amazon EKS	Versi platform baru memperbarui Kubernetes ke tingkat patch `1.10.11` ke alamat `CVE-2018-1002105`.	4 Desember 2018
Ditambahkan `1.0.0` dukungan versi untuk ALB ingress controller	Pengontrol ingress ALB merilis versi `1.0.0` dengan dukungan formal dari. AWS	20 November 2018
Menambahkan dukungan untuk konfigurasi jaringan CNI	Amazon VPC CNI plugin for KubernetesVersi `1.2.1` sekarang mendukung konfigurasi jaringan khusus untuk antarmuka Pod jaringan sekunder.	16 Oktober 2018

Menambahkan dukungan untuk `MutatingAdmissionWebhook` dan `ValidatingAdmissionWebhook`	Amazon EKS versi platform `1.10-eks.2` sekarang mendukung `MutatingAdmissionWebhook` dan `ValidatingAdmissionWebhook` pengendali izin masuk.	10 Oktober 2018
Menambahkan informasi AMI mitra	Canonical telah bermitra dengan Amazon EKS untuk membuat simpul AMI yang dapat digunakan dalam klaster Anda.	3 Oktober 2018
Menambahkan instruksi untuk AWS CLI `update-kubeconfig` perintah	Amazon EKS telah menambahkan `update-kubeconfig` ke AWS CLI untuk menyederh anakan proses pembuatan `kubeconfig` file untuk mengakses cluster Anda.	21 September 2018
AMI Amazon EKS baru yang dioptimalkan	Amazon EKS telah memperbarui AMI yang telah dioptimalkan dengan Amazon EKS AMI (dengan dan tanpa dukungan GPU) untuk memberikan berbagai perbaikan keamanan dan optimasi AMI.	13 September 2018
Wilayah AWS Ekspansi Amazon EKS	Amazon EKS sekarang tersedia di Wilayah Eropa (Irlandia) (`eu-west-1`).	5 September 2018

Pembaruan versi platform Amazon EKS	Versi platform baru dengan dukungan untuk layer Kubernetes agregasi dan Horizontal Pod Autoscaler (HPA).	31 Agustus 2018
Amazon EKS baru mengoptimalkan dukungan AMI dan GPU	Amazon EKS telah memperbarui AMI Amazon EKS yang dioptimalkan untuk menggunakan template AWS CloudFormation node baru dan skrip bootstrap. Selain itu, AMI yang dioptimalkan dengan Amazon EKS dengan dukungan GPU yang baru telah tersedia.	22 Agustus 2018
Amazon EKS baru yang dioptimalkan AMI ditambal `ALAS2-2018-1058`	Amazon EKS telah memperbarui AMI Amazon EKS yang dioptimalkan untuk mengatasi CVE yang direferensikan. `ALAS2-2018-1058`	14 Agustus 2018
Amazon EKS mengoptimalkan skrip pembuatan AMI	Amazon EKS telah menjadikan skrip pembuatan sebagai sumber terbuka yang digunakan untuk membangun AMI yang dioptimalkan dengan Amazon EKS. Skrip build ini sekarang tersedia diGitHub.	10 Juli 2018
Amazon EKS rilis awal	Dokumentasi awal untuk peluncuran layanan	5 Juni 2018

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