Manual do usuário

AWS Batch

AWS Batch: Manual do usuário

Table of Contents

O que é o AWS Batch?

O AWS Batch ajuda a executar workloads de computação em lote na Nuvem AWS. A computação em lote é uma maneira comum para desenvolvedores, cientistas e engenheiros acessarem grandes quantidades de recursos de computação. O AWS Batch remove as tarefas gerais de configuração e gerenciamento da infraestrutura necessária, semelhante ao software de computação em lote tradicional. Esse serviço pode fornecer recursos com eficiência em resposta a trabalhos enviados para eliminar restrições de capacidade, reduzir os custos de computação e entregar resultados rapidamente.

Como um serviço totalmente gerenciado, o AWS Batch ajuda a executar workloads de computação em lote de qualquer escala. O AWS Batch provisiona automaticamente recursos de computação e otimiza a distribuição da workload com base na quantidade e na escala das workloads. Com o AWS Batch, não há necessidade de instalar ou gerenciar o software de computação em lote, para que você possa concentrar seu tempo na análise de resultados e na resolução de problemas.

Tópicos

Componentes do AWS Batch

O AWS Batch simplifica a execução de trabalhos em lote em várias zonas de disponibilidade dentro de uma região. Você pode criar ambientes computacionais do AWS Batch em uma VPC nova ou existente. Depois que um ambiente computacional estiver ativo e associado a uma fila de trabalho, você poderá fazer as definições de trabalho que especificam quais imagens de contêineres do Docker executarão seus trabalhos. As imagens de contêiner são armazenadas em registros de contêiner e extraídas deles, que podem existir dentro ou fora de sua infraestrutura da AWS.

Trabalhos

Uma unidade de trabalho (como um script de shell, um executável do Linux ou uma imagem de contêiner do Docker) que você envia para o AWS Batch. Ela tem um nome e é executada como um aplicativo em contêineres no AWS Fargate ou em recursos do Amazon EC2 no seu

ambiente de computação usando parâmetros que você especifica em uma definição de trabalho. Os trabalhos podem fazer referência a outros trabalhos por nome ou por ID e podem ser dependentes da conclusão de outros trabalhos. Para obter mais informações, consulte Tarefas.

Definições de trabalho

As definições de trabalhos especificam como os trabalhos devem ser executados. Você pode pensar em uma definição de trabalho como um esquema para os recursos em seu trabalho. Você pode fornecer ao seu trabalho um perfil do IAM para fornecer acesso a outros recursos da AWS. Você também especifica os requisitos de memória e CPU. A definição de trabalho também pode controlar as propriedades do contêiner, as variáveis de ambiente e pontos de montagem para armazenamento persistente. Muitas das especificações em uma definição de trabalho podem ser substituídas especificando-se novos valores ao enviar trabalhos individuais. Para obter mais informações, consulte Definições de trabalho

Filas de trabalhos

Quando enviar um trabalho do AWS Batch, você o enviará para uma fila de trabalho específica, onde ele fica até que seja programado em um ambiente computacional. Você pode associar um ou mais ambientes de computação a uma fila de trabalhos. Você também pode atribuir valores de prioridade para esses ambientes de computação e até mesmo para as próprias filas de trabalhos. Por exemplo, você pode ter uma fila de alta prioridade à qual você envia trabalhos urgentes e uma fila de baixa prioridade para trabalhos que podem ser executados a qualquer momento quando os recursos computacionais forem mais baratos.

Ambiente de computação

Um ambiente computacional é um conjunto de recursos computacionais gerenciados ou não gerenciados que são usados para executar trabalhos. Com ambientes de computação gerenciados, você pode especificar o tipo de computação desejado (Fargate ou EC2) em vários níveis de detalhe. Você pode configurar ambientes computacionais que usem um determinado tipo de instância EC2, um modelo específico, como `c5.2xlarge` ou `m5.10xlarge`. Ou você pode optar apenas por especificar que deseja usar os tipos de instância mais recentes. Também é possível especificar os números mínimo, desejado e máximo de vCPUs para o ambiente com a quantia que você está disposto a pagar por uma instância spot como uma porcentagem do preço da instância sob demanda e um conjunto de destino de sub-redes da VPC. O AWS Batch inicia, gerencia e encerra com eficiência os tipos de computação conforme necessário. Você também pode gerenciar seus próprios ambientes computacionais. Nesse caso, você é responsável por configurar e escalar as instâncias

em um cluster do Amazon ECS que o AWS Batch cria para você. Para obter mais informações, consulte Ambiente de computação.

Conceitos básicos

Comece a usar o AWS Batch criando uma definição de trabalho, o ambiente de computação e uma fila de trabalhos no console do AWS Batch.

O assistente de primeira execução do AWS Batch oferece a você a opção de criar um ambiente de computação e uma fila de trabalhos e enviar uma amostra de trabalho Hello World. Se você já tem uma imagem do Docker que desja iniciar no AWS Batch, pode criar uma definição de tarefa com essa imagem e enviá-la para sua fila. Para obter mais informações, consulte Começando com AWS Batch.

Painel

No painel do AWS Batch, você pode monitorar trabalhos recentes, as filas de trabalhos e os ambientes de computação. Por padrão, os seguintes widgets do painel são exibidos:

- Visão geral do trabalho – para obter mais informações sobre trabalhos do AWS Batch, consulte Tarefas.
- Visão geral da fila de trabalhos – para obter mais informações fila de trabalhos do AWS Batch, consulte Filas de tarefas.
- Visão geral do ambiente de computação – para obter mais informações sobre ambientes de computação do AWS Batch, consulte Ambiente de computação.

Você pode personalizar os widgets que são exibidos na página do painel. As seções a seguir descrevem widgets adicionais que você pode instalar.

Fila de trabalhos única

Esse widget exibe informações detalhadas sobre uma única fila de trabalhos.

Para adicionar este widget, siga essas etapas.

1. Abra o console do AWS Batch.
2. Na barra de navegação, selecione a Região da AWS desejada.

3. No painel de navegação, escolha Dashboard (Painel).
4. Escolha Adicionar widgets.
5. Em Fila de um único trabalho, escolha Adicionar widget.
6. Em Fila de trabalhos, escolha a fila de trabalhos desejada.
7. Em Status do trabalho, escolha os status do trabalho que você deseja exibir.
8. (Opcional) Desative Exibir ambientes de computação conectados se não desejar exibir as propriedades dos ambientes computacionais.
9. Em Propriedades do ambiente de computação, selecione as propriedades desejadas.
10. Escolha Add (Adicionar).

CloudWatch Container Insights

Esse widget exibe métricas agregadas para ambientes de computação e trabalhos do AWS Batch. Para obter mais informações sobre o Container Insights, consulte CloudWatch Container Insights.

Para adicionar este widget, siga essas etapas.

1. Abra o console do AWS Batch.
2. Na barra de navegação, selecione a Região da AWS desejada.
3. No painel de navegação, escolha Dashboard (Painel).
4. Escolha Adicionar widgets.
5. Em Container insights, escolha Adicionar widget.
6. Em Ambiente de computação, escolha o ambiente de computação que você deseja.
7. Escolha Add (Adicionar).

Logs de trabalho

Esse widget exibe logs diferentes das trabalhos em um local conveniente. Para obter mais informações sobre os logs de trabalhos, consulte the section called “ Logs de trabalho”.

Para adicionar este widget, siga essas etapas.

1. Abra o console do AWS Batch.
2. Na barra de navegação, selecione a Região da AWS desejada.

3. No painel de navegação, escolha Dashboard (Painel).
4. Escolha Adicionar widgets.
5. Em Logs de trabalhos, escolha Adicionar widget.
6. Em ID de trabalho, insira o ID do trabalho que você deseja.
7. Escolha Add (Adicionar).

Configuração com o AWS Batch

Se você já tiver se cadastrado no Amazon Web Services (AWS) e usado o Amazon Elastic Compute Cloud (Amazon EC2) ou Amazon Elastic Container Service (Amazon ECS), poderá usá-lo em breve. AWS Batch O processo de configuração para os dois serviços é semelhante. Isso ocorre porque AWS Batch usa instâncias de contêiner do Amazon ECS em seus ambientes computacionais. Para usar a AWS CLI com o AWS Batch, você deve usar uma versão da AWS CLI que ofereça suporte aos recursos mais recentes do AWS Batch. Se você sentir falta do suporte a um determinado recurso do AWS Batch na AWS CLI, atualize para a versão mais recente. Para obter mais informações, consulte http://aws.amazon.com/cli/.

> **Note**
>
> Como o AWS Batch usa componentes do Amazon EC2, você usa o console do Amazon EC2 para muitas dessas etapas.

Conclua as tarefas a seguir para configuração do AWS Batch. Caso já tenha concluído qualquer uma dessas etapas, você pode ignorá-las e passar à instalação da AWS CLI.

Tópicos

Cadastrar-se em uma Conta da AWS

Se você ainda não tem uma Conta da AWS, siga as etapas a seguir para criar uma.

Para se cadastrar em uma Conta da AWS

1. Abra https://portal.aws.amazon.com/billing/signup.

2. Siga as instruções on-line.

   Parte do procedimento de aplicação envolve receber uma chamada telefônica e digitar um código de verificação no teclado do telefone.

   Quando você se cadastra em uma Conta da AWS, um Usuário raiz da conta da AWS é criado. O usuário raiz tem acesso a todos os Serviços da AWS e atributos na conta. Como prática recomendada de segurança, atribua acesso administrativo a um usuário administrativo e use somente o usuário raiz para realizar as tarefas que exigem acesso do usuário raiz.

A AWS envia um e-mail de confirmação depois que o processo de cadastramento é concluído. A qualquer momento, é possível visualizar as atividades da conta atual e gerenciar sua conta acessando https://aws.amazon.com/ e selecionando Minha conta.

Criar um usuário administrativo

Depois de se inscrever em uma Conta da AWS, proteja seu Usuário raiz da conta da AWS, habilite o AWS IAM Identity Center e crie um usuário administrativo para não usar o usuário raiz em tarefas cotidianas.

Proteger seu Usuário raiz da conta da AWS

1. Faça login no AWS Management Console como o proprietário da conta ao escolher a opção Usuário raiz e inserir o endereço de e-mail da Conta da AWS. Na próxima página, digite sua senha.

   Para obter ajuda ao fazer login usando o usuário raiz, consulte Fazer login como usuário raiz no Guia do usuário do Início de Sessão da AWS.

2. Ative a autenticação multifator (MFA) para o usuário raiz.c

   Para obter instruções, consulte Habilitar um dispositivo MFA virtual para o usuário raiz de sua conta da Conta da AWS para seu (console) no Guia do usuário do IAM.

Criar um usuário administrativo

1. Habilitar o IAM Identity Center.

Para obter instruções, consulte Enabling AWS IAM Identity Center no Manual do Usuário do AWS IAM Identity Center.

2. No Centro de Identidade do IAM, conceda acesso administrativo a um usuário administrativo.

   Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configure user access with the default Diretório do Centro de Identidade do IAM no Manual do Usuário do AWS IAM Identity Center.

Login como usuário administrativo

- Para fazer login com seu usuário do Centro de Identidade do IAM, use a URL de login que foi enviada ao seu endereço de e-mail quando você criou o usuário do Centro do Usuário do IAM.

  Para obter ajuda com o login utilizando um usuário do Centro de Identidade do IAM, consulte Fazer login no portal de acesso da AWS, no Guia do usuário do Início de Sessão da AWS.

Criar funções do IAM para seus ambientes de computação e instâncias de contêiner

Os ambientes de computação e instâncias de contêiner do AWS Batch exigem credenciais de conta da Conta da AWS para fazer chamadas para outras APIs da AWS em seu nome. Você deve criar um perfil do IAM; que fornece essas credenciais para seus ambientes de computação e instâncias de contêiner e, em seguida, associar esse perfil aos seus ambientes de computação.

> **Note**
>
> Os perfis de instância de contêiner e ambiente de computação do AWS Batch são criados para você automaticamente na primeira execução do console. Portanto, se você pretende usar o console do AWS Batch, pode avançar para a próxima seção. Se você planeja usar a AWS CLI, conclua os procedimentos em Usando funções vinculadas a serviços para AWS Batch e Perfil de instância do Amazon ECS antes de criar seu primeiro ambiente de computação.

Criar um par de chaves

AWSA usa criptografia de chave pública para proteger as informações de logon da instância. Uma instância do Linux, como uma instância de contêiner do AWS Batch não possui senha a ser usada para acesso SSH. Você usa um par de chaves para fazer login na sua instância com segurança. Você especifica o nome do par de chaves ao criar o ambiente de computação e fornece a chave privada ao fazer logon usando SSH.

Se ainda não tiver criado um par de chaves, você poderá criar um usando o console do Amazon EC2. Observe que, se quiser iniciar instâncias em várias Regiões da AWS, crie um par de chaves em cada região. Para obter mais informações sobre regiões, consulte Regiões e zonas de disponibilidade no Manual do usuário do para instâncias do Linux Amazon EC2.

Para criar um par de chaves

1. Abra o console do Amazon EC2 em https://console.aws.amazon.com/ec2/.
2. Na barra de navegação, selecione uma Região da AWS para o par de chaves. Você pode selecionar qualquer região que esteja disponível, independentemente de sua localização: No entanto, pares de chaves são específicos de uma região. Por exemplo, se você planeja executar uma instância na região US West (Oregon), deve criar um par de chaves para a instância na mesma região.
3. No painel de navegação, escolha Pares de chaves, Criar par de chaves.
4. Na caixa de diálogo Criar par de chaves, para Nome do par de chaves, insira um nome para o novo par de chaves e escolha Criar. Escolha um nome que seja fácil de memorizar, como seu nome de usuário, seguido por `-key-pair` mais o nome da região. Por exemplo, eu-par de chaves-uswest2.
5. O arquivo de chave privada é baixado automaticamente pelo navegador. O nome do arquivo base é o especificado por você como sendo o nome do par de chaves, e a extensão do nome do arquivo é `.pem`. Salve o arquivo de chave privada em um lugar seguro.

> **Important**
>
> Esta é a única chance de você salvar o arquivo de chave privada. Será necessário fornecer o nome do par de chaves ao executar uma instância e a chave privada correspondente sempre que for estabelecida uma conexão com a instância.

6. Se você usar um cliente de SSH em um computador Mac ou Linux para se conectar à instância do Linux, use o seguinte comando para definir as permissões do arquivo de chave privada. Dessa forma, só você pode lê-lo.

```
$ chmod 400 your_user_name-key-pair-region_name.pem
```

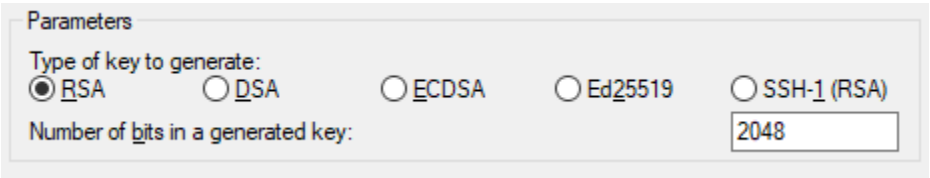
Para obter mais informações, consulte Pares de chaves do Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Linux.

Para se conectar à instância usando o par de chaves

Para se conectar à instância do Linux em um computador no qual o Mac ou o Linux esteja em execução, especifique o arquivo `.pem` ao cliente SSH com a opção `-i` e o caminho para a chave privada. Para se conectar à sua instância Linux a partir de um computador executando o Windows, use um MindTerm ou o PuTTY. Se você planeja usar PuTTY, precisa instalá-lo e usar o procedimento a seguir para converter o arquivo `.pem` em um arquivo `.ppk`.

(Opcional) Para se preparar para se conectar a uma instância do Linux a partir do Windows usando PuTTY

1. Baixe e instale o PuTTY em http://www.chiark.greenend.org.uk/~sgtatham/putty/. Instale o pacote inteiro.
2. Inicie o PuTTYgen (por exemplo, no menu Start, escolha All Programs, PuTTY, and PuTTYgen).
3. Em Tipo de chave a ser gerada, escolha RSA. Se você estiver usando uma versão mais antiga do PuTTYgen, escolha SSH-2 RSA.



4. Escolha Load. Por padrão, o PuTTYgen exibe somente arquivos com a extensão `.ppk`. Para localizar o arquivo `.pem`, escolha a opção para exibir arquivos de todos os tipos.

5. Selecione o arquivo de chave privada que você criou no procedimento anterior e escolha Abrir. Escolha OK para descartar a caixa de diálogo de confirmação.

6. Escolha Save private key. A PuTTYgen exibe um aviso sobre salvar a chave sem uma senha. Escolha Sim.
7. Especifique o mesmo nome da chave usado para o par de chaves. O PuTTY adiciona automaticamente a extensão de arquivo `.ppk`.

Crie uma VPC

Com a Amazon Virtual Private Cloud (Amazon VPC) você pode iniciar recursos da AWS em uma rede virtual definida por você. É altamente recomendável ativar as instâncias de contêiner em um VPC.

Caso tenha uma VPC padrão, você também pode ignorar esta seção e avançar à próxima tarefa, Crie um grupo de segurança. Para determinar se você tem uma VPC padrão, consulte Plataformas compatíveis no console do Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Linux.

Para obter mais informações sobre como criar VPCs, consulte Criação de uma VPC no Guia do usuário da Amazon VPC. Consulte a tabela a seguir para determinar quais opções selecionar.

Opção	Valor
Recursos a criar	Somente VPC
Nome	Forneça um nome opcional para a sua VPC.
Bloco IPv4 CIDR	Entrada manual IPv4 CIDR O tamanho do bloco CIDR deve ter um tamanho entre /16 e /28.
Bloco IPv6 CIDR	Sem bloco IPv6 CIDR
Locação	Padrão

Para saber mais sobre a Amazon VPC, consulte O Que é Amazon VPC? no Guia de usuário Amazon VPC.

Crie um grupo de segurança

Os security groups funcionam como um firewall para instâncias de contêiner de ambiente de computação associadas, controlando o tráfego de entrada e de saída no nível de instância de contêiner. Um grupo de segurança só pode ser usado na VPC na qual ele é criado.

É possível adicionar regras a um grupo de segurança que permite a você se conectar à instância de contêiner do endereço IP usando SSH. Você também pode adicionar regras que permitam o acesso HTTP e HTTPS de entrada e saída de qualquer lugar. Adicione eventuais regras a portas abertas exigidas pelas tarefas.

Caso pretenda ativar instâncias de contêiner em várias regiões, você precisa criar um grupo de segurança em cada região. Para obter mais informações, consulte Regiões e zonas de disponibilidade no Guia do usuário do Amazon EC2 para instâncias do Linux.

> **Note**
>
> O endereço IP público do computador local é necessário e pode ser obtido por meio de um serviço. Por exemplo, fornecemos o seguinte serviço: http://checkip.amazonaws.com/ ou https://checkip.amazonaws.com/. Para localizar outro serviço que forneça o endereço IP, use a frase de busca "qual é o meu endereço IP". Caso esteja se conectando por meio de um provedor de serviços de Internet (ISP) ou atrás de um firewall sem um endereço IP estático, descubra o intervalo de endereços IP usado por computadores cliente.

Para criar um grupo de segurança usando o console

1. Abra o console da Amazon VPC em https://console.aws.amazon.com/vpc/.
2. No painel de navegação, escolha Grupos de segurança.
3. Escolha Create grupo de segurança (Criar grupo de segurança).
4. Insira um nome e uma descrição para o grupo de segurança. Não é possível alterar o nome e a descrição de um grupo de segurança após ele ser criado.
5. Em VPC, escolha a VPC.
6. Por padrão, novos grupos de segurança começam com apenas uma regra de saída que permite que todo o tráfego deixe as instâncias. Adicione regras para permitir qualquer tráfego de entrada ou para restringir o tráfego de saída.

As instâncias de contêiner do AWS Batch não exigem que nenhuma porta de entrada esteja aberta. No entanto, talvez você queira adicionar uma regra SSH. No entanto, você pode adicionar uma regra SSH para fazer login na instância de contêiner e examinar os contêineres em trabalhos com comandos do Docker. Se desejar que sua instância de contêiner hospede um trabalho que executa um servidor da Web, você também pode adicionar regras para HTTP. Conclua as seguintes etapas para adicionar essas regras de security group opcionais.

Na guia Entrada, crie as seguintes regras e escolha Criar:

- Escolha Add Rule. Para Tipo, escolha HTTP. Para Origem, escolha Qualquer lugar (`0.0.0.0/0`).
- Escolha Add Rule. Para Tipo, escolha SSH. Para Source, escolha Custom IP e especifique o endereço IP público do seu computador ou rede na notação Roteamento sem classe entre domínios (CIDR). Se sua empresa alocar endereços de um intervalo, especifique o intervalo inteiro, como `203.0.113.0/24`. Para especificar um único endereço IP em notação CIDR, escolha Meu IP. Isso adiciona o prefixo de roteamento `/32` ao endereço IP público.

> **Note**
>
> Por motivos de segurança, não recomendamos que você permita acesso SSH de todos os endereços IP (`0.0.0.0/0`) à sua instância, exceto para fins de teste e somente por um curto período.

7. É possível adicionar etiquetas agora ou pode adicioná-las mais tarde. Para adicionar uma tag, escolha Add new tag, e insira a chave e o valor da tag.
8. Escolha Create grupo de segurança.

Para criar um grupo de segurança usando a linha de comando, consulte create-security-group(AWS CLI)

Para obter mais informações sobre os grupos de segurança, consulte Como trabalhar com grupos de segurança.

Instalar a AWS CLI

Para usar a AWS CLI com o AWS Batch, instale a versão mais recente da AWS CLI. Para obter informações sobre a instalação da AWS CLI ou a atualização para a versão mais recente, consulte Interface da linha de comando da AWS no Guia do usuário do AWS Command Line Interface.

Começando com AWS Batch

Você pode usar o assistente de AWS Batch primeira execução para começar AWS Batch rapidamente. Depois de concluir os pré-requisitos, você pode usar o assistente de primeira execução para criar um ambiente de computação, uma definição de trabalhos e uma fila de trabalhos.

Você também pode enviar um exemplo de job “Hello World” usando o assistente AWS Batch de primeira execução para testar sua configuração. Se você já tem uma imagem do Docker na qual deseja iniciar AWS Batch, pode usar essa imagem para criar uma definição de tarefa.

Pré-requisitos

Certifique-se de fazer o seguinte antes de iniciar o assistente de AWS Batch primeira execução:

- Conclua as etapas que estão descritas em Configuração com o AWS Batch.
- Verifique se você Conta da AWS tem as permissões necessárias.

Conceitos básicos – Amazon EC2

O Amazon Elastic Compute Cloud (Amazon EC2) oferece uma capacidade de computação escalável na Nuvem AWS. O uso do Amazon EC2 elimina a necessidade de investir em hardware inicialmente, portanto, você pode desenvolver e implantar aplicativos com mais rapidez.

É possível usar o Amazon EC2 para executar quantos servidores virtuais forem necessários, configurar a segurança e as redes e gerenciar o armazenamento. O Amazon EC2 permite aumentar ou reduzir a escala verticalmente para lidar com alterações nos requisitos ou com picos em popularidade, reduzindo sua necessidade de prever o tráfego.

Crie um ambiente de computação

Para criar um ambiente de computação para uma orquestração do Amazon EC2, faça o seguinte:

1. Abra o assistente de primeira execução do console do AWS Batch.
2. Em Selecionar tipo de orquestração, escolha Amazon Elastic Compute Cloud (Amazon EC2).
3. Escolha Próximo.

4. Na seção Configuração do ambiente de computação, em Nome, especifique um nome exclusivo para seu ambiente de computação. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).
5. Para a Perfil de instância, escolha criar um novo perfil de instância ou use um existente que tenha as permissões de IAM necessárias anexadas. Esse perfil de instância permite que as instâncias de contêiner do Amazon ECS em seu ambiente computacional façam chamadas para as operações de AWS API necessárias. Para ter mais informações, consulte Perfil de instância do Amazon ECS.
6. (Opcional) Uma tag é um rótulo atribuído a um recurso. Para adicionar uma tag ou uma tag do Amazon EC2, expanda Tags e escolha Adicionar tag. Insira um par valor-chave e escolha Adicionar tag novamente.

   > **Important**
   >
   > Se você escolher Adicionar tag, deverá inserir um par chave-valor e escolher Adicionar tag novamente ou escolher Remover tag.

7. (Opcional) Na seção Configuração da instância para Usar instâncias spot do Amazon EC2, ative Ativar o uso de instâncias Spot.
8. (Apenas spot) Em Porcentagem máxima no preço sob demanda, insira a porcentagem máxima de preços sob demanda que você deseja pagar pelos recursos spot.
9. (Opcional) (Apenas Spot) Em Função de frota spot, escolha uma função do Amazon EC2 da frota spot do perfil do IAM a ser aplicada ao seu ambiente de computação spot. Se você ainda não tiver um perfil do IAM da frota spot do Amazon EC2 existente, crie um primeiro. Para ter mais informações, consulte Perfil de frota spot Amazon EC2.

   > **Important**
   >
   > Para marcar suas Instâncias Spot na criação, sua função IAM do Amazon EC2 Spot Fleet deve usar a política gerenciada mais recente do SpotFleetTaggingRoleAmazonEC2. A política SpotFleetRole gerenciada do AmazonEC2 não tem as permissões necessárias para marcar instâncias spot. Para ter mais informações, consulte Instâncias spot sem tags na criação e the section called “Marcando seus Recursos”.

10. Em Mínimo de vCPUs, escolha o número mínimo de vCPUs do EC2 que seu ambiente de computação mantém, independentemente da demanda da fila de trabalhos.

11. Em vCPUs desejados, escolha o número de vCPUs do EC2 com que seu ambiente de computação é executado. Conforme a demanda da fila de trabalhos aumenta, o AWS Batch aumenta o número desejado de vCPUs e adiciona instâncias do EC2. O número de vCPUs pode aumentar até o número máximo de vCPUs. À medida que a demanda diminui, AWS Batch diminui o número desejado de vCPUs e remove instâncias. O número diminui até o número mínimo de vCPUs.
12. Em Máximo de vCPUs, escolha o número máximo de vCPUs do EC2 para o qual seu ambiente de computação deve aumentar, independentemente da demanda da fila de trabalhos.
13. Em Tipos de instância permitidos, escolha os tipos de instância do Amazon EC2 que podem ser executados. Você pode especificar famílias de instâncias para iniciar qualquer tipo de instância dentro dessas famílias (por exemplo `c5`, `c5n`, ou `p3`). Ou você pode especificar tamanhos específicos dentro de uma família (como `c5.8xlarge`). Os tipos de instância Metal não estão nas famílias de instâncias. Por exemplo, `c5` não inclui `c5.metal`. Você também pode escolher `optimal` para selecionar tipos de instância (das famílias de instâncias C4, M4, e R4 que correspondam à demanda de suas filas de trabalho.

> **Note**
>
> Ao criar um ambiente de computação, os tipos de instância selecionados para ele devem compartilhar a mesma arquitetura. Por exemplo, você não pode misturar instâncias ARM e x86 no mesmo ambiente de computação.

> **Note**
>
> AWS Batch dimensiona as GPUs com base na quantidade necessária em suas filas de trabalho. Para usar o agendamento de GPU, o ambiente de computação deve incluir tipos de instância das famílias `p2`, `p3`, `p4`, `p5`, `g3`, `g3s`, `g4`, ou `g5`.

> **Note**
>
> Atualmente, o `optimal` usa tipos de instância das famílias de instâncias C4, M4 e R4. Regiões da AWS Nesse caso, não há tipos de instância dessas famílias de instâncias, tipos de instância da C5M5, e famílias de R5 instâncias são usadas.

14. Expanda Additional configuration.
15. (Opcional) Em Grupo de posicionamento, insira um nome de grupo de posicionamento para agrupar recursos no ambiente de computação.
16. (Opcional) Para o par de chaves EC2, escolha um par de chaves pública e privada como credenciais de segurança ao se conectar à instância. Para obter mais informações sobre pares de chaves Amazon EC2, consulte Amazon EC2 key pairs and Linux instances.
17. Para Estratégia de alocação, escolha a estratégia de alocação a ser usada ao selecionar tipos de instância na lista de tipos de instância permitidos. BEST_FIT_PROGRESSIVE costuma ser a melhor opção para ambientes de computação EC2 sob demanda, e SPOT_CAPACITY_OPTIMIZED, para ambientes de computação EC2 Spot. Para ter mais informações, consulte the section called “Estratégias de alocação”.
18. (Opcional) Em Configuração do EC2, escolha Adicionar configuração do EC2. Escolha o tipo de imagem e os valores de substituição do ID da imagem para fornecer informações AWS Batch para selecionar Amazon Machine Images (AMIs) para instâncias no ambiente computacional. Se a substituição do ID da imagem não for especificada para cada tipo de imagem, AWS Batch seleciona uma AMI otimizada recente do Amazon ECS. Se nenhum tipo de imagem for especificado, o padrão será Amazon Linux 2 para instância sem GPU e sem AWS Graviton.

    > **Important**
    >
    > Para usar uma AMI personalizada, escolha o tipo de imagem e insira a ID da AMI personalizada na caixa de substituição de ID da imagem.

    Amazon Linux 2

    Padrão para todas as famílias de instâncias AWS baseadas em Graviton (por exemplo,, `C6g` `M6gR6g`, `eT4g`) e pode ser usado para todos os tipos de instâncias que não sejam de GPU.

    Amazon Linux 2 (GPU)

    Padrão para todas as famílias de instâncias de GPU (por exemplo, `P4` `eG4`) e pode ser usado para todos os tipos de instância não AWS baseados em Graviton.

    Amazon Linux

    Pode ser usado para famílias de instâncias sem GPU e sem AWS Graviton. O suporte padrão para a AMI do Amazon Linux terminou. Para obter mais informações, consulte AMI do Amazon Linux.

> **Note**
>
> A AMI que você escolher para um ambiente de computação deve corresponder à arquitetura dos tipos de instância que você pretende usar para este ambiente. Por exemplo, se o ambiente de computação usar tipos de instância A1, a AMI de recursos de computação escolhida deverá oferecer suporte a instâncias Arm. O Amazon ECS vende as versões x86 e Arm da Amazon ECS optimized Amazon Linux 2 AMI. Para obter mais informações, consulte AMI do Amazon Linux 2 otimizada para Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

19. (Opcional) Em Modelo de execução, selecione um modelo de execução existente do Amazon EC2 para configurar seus recursos de computação. A versão padrão do modelo é preenchida automaticamente. Para ter mais informações, consulte Suporte a modelo de execução.

> **Note**
>
> Em um modelo de execução, é possível especificar uma AMI personalizada que você tenha criado.

20. (Opcional) Em Versão do modelo de execução, insira `$Default`, `$Latest` ou um número de versão específico para ser usado.

> **Important**
>
> Depois que o ambiente de computação é criado, a versão do modelo de execução usada não é alterada, mesmo que a versão `$Default` ou `$Latest` do modelo de execução seja atualizada. Para usar uma nova versão do modelo de execução, primeiro crie um ambiente de computação e adicione o novo ambiente de computação à fila de trabalhos existente. Em seguida, remova o ambiente de computação antigo da fila de trabalhos e exclua o ambiente de computação antigo.

21. Na seção Configuração de rede:

    a. Em ID da Nuvem privada virtual (VPC), selecione uma Amazon VPC.

b. Em Sub-redes, estão listadas as sub-redes para a sua Conta da AWS . Se você quiser criar um conjunto personalizado de sub-redes, escolha Limpar sub-redes e, em seguida, escolha as sub-redes desejadas.

> **Important**
>
> Os recursos de computação devem se comunicar com o endpoint da VPC do Amazon ECS por meio de um VPC endpoint ou de vários endereços IP públicos. Para obter mais informações, consulte VPC endpoint de interface do Amazon ECS (AWS PrivateLink). Se sua instância não tiver um VPC endpoint configurado ou um endereço IP público, você poderá usar a conversão de endereços de rede (NAT). Para obter mais informações NAT, consulte NAT gateways e Criando uma Nuvem Privada Virtual .

c. Em Grupos de segurança, escolha os grupos de segurança do Amazon EC2 que você deseja associar à instância. Se você quiser criar um conjunto personalizado de grupos de segurança, escolha Limpar grupos de segurança. Escolha os grupos de segurança que você deseja.

22. Escolha Próximo.

Crie uma fila de trabalhos

Uma fila de trabalhos armazena seus trabalhos enviados até que o AWS Batch Agendador execute o trabalho em um recurso em seu ambiente computacional. Para obter mais informações, consulte Filas de tarefas

Para criar uma fila de trabalhos para uma orquestração do Amazon EC2, faça o seguinte:

1. Na seção Configuração da fila de trabalhos, em Nome, especifique um nome exclusivo para seu ambiente de computação. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).
2. Em Prioridade, insira um número inteiro entre 0 e 100 para a fila de trabalhos.

> **Important**
>
> Valores inteiros mais altos são atribuídos a uma prioridade mais alta pelo AWS Batch Scheduler.

3. Escolha Próximo.

Crie uma definição de trabalho

AWS Batch as definições de tarefas especificam como as tarefas devem ser executadas. Embora cada tarefa deva fazer referência a uma definição de tarefa, muitos parâmetros que são especificados na definição de tarefa podem ser substituídos em runtime.

Para criar a definição de trabalho:

1. Na seção Configuração geral:

   a. Na seção Configuração geral, em Nome, especifique um nome exclusivo para seu ambiente de computação. Os nomes podem ter até 128 caracteres. O nome pode conter letras minúsculas e maiúsculas, números, hifens e (-) e sublinhados (_).

   b. (Opcional) Em Tempo limite de execução, insira a quantidade de tempo (em segundos) após a qual um trabalho inacabado termina.

   > **Important**
   >
   > O tempo limite mínimo é de 60 segundos.

   c. (Opcional) Uma tag é um rótulo atribuído a um recurso. Para adicionar uma tag, expanda Tags e escolha Adicionar tag. Insira um par valor-chave e escolha Adicionar tag novamente.

   > **Important**
   >
   > Se você escolher Adicionar tag, deverá inserir um par chave-valor e escolher Adicionar tag novamente ou escolher Remover tag.

   d. (Opcional) Ative Propagar tags para propagar tags para o Amazon Elastic Container Service.

2. Na seção Configuração do contêiner:

   a. Em Imagem, insira o nome da imagem usada para iniciar o contêiner. Por padrão, todas as imagens no registro do Docker Hub estão disponíveis. Você também pode especificar outros repositórios no formato repository-url/image:tag. O parâmetro pode ter até 255 caracteres. O parâmetro conter letras maiúsculas e minúsculas, números, hifens (-),

sublinhados (_), dois pontos (:), pontos (.), barras (/) e sinais de número (#). O parâmetro é mapeado para `Image` na seção Criar um contêiner da API remota do Docker e o parâmetro `IMAGE` de docker run.

> **Note**
>
> A arquitetura da imagem do Docker deve corresponder à arquitetura do processador dos recursos de computação nos quais eles foram programados. Por exemplo, imagens Arm baseadas Docker em só podem ser executadas em recursos de computação baseados em Arm.

- As imagens em repositórios públicos Amazon ECR usam convenções de nomenclatura completas `registry/repository[:tag]` ou `registry/repository[@digest]` (por exemplo, `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).
- As imagens em repositórios do Amazon ECR usam as convenções de nomenclatura `registry/repository:tag` completa (por exemplo, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*).
- As imagens em repositórios oficiais no Docker Hub usam um único nome (por exemplo, `ubuntu` ou `mongo`).
- As imagens em outros repositórios no Docker Hub são qualificadas com um nome de organização (por exemplo, `amazon/amazon-ecs-agent`).
- Imagens em outros repositórios online também são qualificadas por um nome de domínio (por exemplo, `quay.io/assemblyline/ubuntu`).

b. Para Command, especifique o comando a ser passado para o contêiner. Esse parâmetro é mapeado para `Cmd` na seção Criar um contêiner da Docker Remote API e o parâmetro `COMMAND` de docker run. Para obter mais informações sobre o parâmetro `CMD` da Docker, consulte https://docs.docker.com/engine/reference/builder/#cmd.

> **Note**
>
> Você pode usar os valores padrão de substituição de parâmetros e marcadores no seu comando. Para ter mais informações, consulte Parâmetros.

c. (Opcional) Em Função de execução, especifique um perfil do IAM que conceda aos atendentes de contêiner do Amazon ECS permissão para fazer chamadas da API da AWS

em seu nome. Esse atributo usa perfis do IAM do Amazon ECS para tarefas. Para obter mais informações, consulte Perfis do IAM para execução de tarefa do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

d. (Opcional) Para a configuração do Job Role, escolha um papel do IAM que tenha permissões para as AWS APIs. Esse atributo usa perfis do IAM do Amazon ECS para tarefas. Para mais informações, consulte Funções do IAM para Tarefas no Guia de Desenvolvedor Amazon Elastic Container Service.

> **Note**
>
> Somente funções que tenham o relacionamento de confiança Função da tarefa do Amazon Elastic Container Service são mostradas aqui. Para obter mais informações sobre a criação de uma função do IAM para seus AWS Batch trabalhos, consulte Como criar uma função e uma política do IAM para suas tarefas no Amazon Elastic Container Service Developer Guide.

e. (Opcional) Você pode adicionar parâmetros à definição do trabalho como mapeamentos de chave-valor para substituir os padrões de definição do trabalho. Para adicionar um parâmetro:

- Em Parâmetros, escolha Adicionar parâmetro. Insira um par valores-chave e escolha Adicionar parâmetronovamente.

> **Important**
>
> Se você escolher Adicionar parâmetro, deverá configurar pelo menos um parâmetro ou escolher Remover parâmetro..

f. Na seção Configuração do ambiente em vCPUs, especifique o número de vCPUs a serem reservadas para o contêiner. Esse parâmetro é mapeado para `CpuShares` na seção Criar um Contêiner da API remota do Docker e a opção `--cpu-shares` para docker run. Cada vCPU equivale a 1.024 compartilhamentos de CPU.

g. Em Memória, especifique o limite rígido (em MiB) de memória a ser apresentado ao contêiner do trabalho. Caso seu contêiner tente exceder a memória especificada, o mesmo será interrompido. Esse parâmetro é mapeado para `Memory` na seção Criar um Contêiner da API Remota Docker e para a opção `--memory` para docker run.

h. Em Número de GPUs, escolha o número de GPUs a serem reservadas para o contêiner.

i. (Opcional) Em Configuração de variáveis de ambiente, escolha Adicionar variáveis de ambiente para adicionar variáveis de ambiente a serem passadas para o contêiner. Esse parâmetro é mapeado para `Env` na seção Criar um Contêiner da API remota do Docker e a opção `--env` para docker run.

j. (Opcional) Em Segredos, escolha Adicionar segredo para adicionar segredos como pares de nome-valor. Esses segredos são expostos no contêiner. Para obter mais informações, consulte SecretOptions em Parâmetros de definição de trabalho para ContainerProperties.

k. (Opcional) Na seção Configuração do Linux:

   i. Para Usuário, insira o nome do usuário a ser usado dentro do contêiner. Esse parâmetro é mapeado para `User` na seção Criar um Contêiner da API remota do Docker e a opção `--user` para docker run.

   ii. Para oferecer ao contêiner de trabalho permissões elevadas na instância do host (semelhante ao usuário `root`), arraste o controle deslizante Privilegiado para a direita. Esse parâmetro é mapeado para `Privileged` na seção Criar um Contêiner da API remota do Docker e a opção `--privileged` para docker run.

   iii. Ative Ativar processo init para executar um processo `init` dentro do contêiner. Este processo encaminha sinais e colhe processos.

l. (Opcional) Na seção Configuração do sistema de arquivos:

   i. Ative Habilitar sistema de arquivos somente para leitura para remover o acesso de gravação ao volume.

   ii. Para Tamanho da memória compartilhada, insira o tamanho (em MiB) do volume `/dev/shm`.

   iii. Para Tamanho máximo de troca, insira a quantidade total de memória de troca (em MiB) que o contêiner pode usar.

   iv. Em Swappiness, insira um valor entre 0 e 100 para indicar o comportamento de troca de memória do contêiner. Se você não especificar um valor e a troca estiver ativada, o valor assumirá 60 como padrão. Para obter mais informações, consulte swappiness em Parâmetros de definição de trabalho para ContainerProperties.

   v. (Opcional) Expanda Configuração adicional.

   vi. Em Tmpfs, escolha Adicionar tmpfs para adicionar uma montagem `tmpfs`.

   vii. Em Dispositivos, escolha Adicionar dispositivo para adicionar um dispositivo:

A. Em Container path (Caminho do contêiner), especifique o caminho na instância de contêiner para expor o dispositivo mapeado para a instância de host. Se isso for deixado em branco, o caminho de host será usado no contêiner.

B. Em Host path, especifique o caminho de um dispositivo na instância de host.

C. Em Permissions, selecione uma ou mais permissões para serem aplicadas ao dispositivo no contêiner. As permissões disponíveis são LER, GRAVAR e MKNOD.

viii. (Opcional) Para Configuração de ulimits, escolha Adicionar ulimit para adicionar um valor de `ulimits` ao contêiner. Insira os valores Nome, Limite flexível e Limite rígido e então escolha Adicionar ulimit.

3. Escolha Próximo.

Crie um trabalho

Para criar um trabalho, faça o seguinte:

1. Na seção Configuração do trabalho, em Nome, especifique um nome exclusivo para o trabalho. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).
2. Escolha Próximo.

Examinar e criar

Na página Revisar e criar, revise as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Criar recursos.

Conceitos básicos – Fargate

AWS O Fargate inicia e dimensiona a computação de acordo com os requisitos de recursos que você especifica para o contêiner. Com o Fargate, você não precisa provisionar em excesso nem pagar por servidores adicionais. Para obter mais informações, consulte Fargate.

Crie um ambiente de computação

Para criar um ambiente de computação para uma orquestração do Fargate, faça o seguinte:

1. Abra o assistente de primeira execução do console do AWS Batch.

2. For Selecionar tipo de orquestração, escolha Fargate.
3. Escolha Próximo.
4. Na seção Configuração do ambiente de computação, em Nome, especifique um nome exclusivo para seu ambiente de computação. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).
5. (Opcional) Uma tag é um rótulo atribuído a um recurso. Para adicionar uma tag, expanda Tags e escolha Adicionar tag. Insira um par valor-chave e escolha Adicionar tag novamente.

   > **Important**
   >
   > Se você escolher Adicionar tag, deverá inserir um par chave-valor e escolher Adicionar tag novamente ou escolher Remover tag.

6. (Opcional) Na seção Configuração da instância para Usar capacidade do Fargate Spot , ative Ativar uso de instâncias spot.
7. Em Número máximo de vCPUs, insira o número máximo de vCPUs que a instância pode usar.
8. Na seção Configuração de rede:

   a. Em ID da Nuvem privada virtual (VPC), selecione uma Amazon VPC.

   b. Em Sub-redes, estão listadas as sub-redes para a sua Conta da AWS . Se você quiser criar um conjunto personalizado de sub-redes, escolha Limpar sub-redes e, em seguida, escolha as sub-redes desejadas.

      > **Important**
      >
      > Os recursos de computação devem se comunicar com o endpoint da VPC do Amazon ECS por meio de um VPC endpoint ou de vários endereços IP públicos. Para obter mais informações, consulte VPC endpoint de interface do Amazon ECS (AWS PrivateLink). Se sua instância não tiver um VPC endpoint configurado ou um endereço IP público, você poderá usar a conversão de endereços de rede (NAT). Para obter mais informações NAT, consulte NAT gateways e Criando uma Nuvem Privada Virtual .

   c. Em Grupos de segurança, escolha os grupos de segurança do Amazon EC2 que você deseja associar à instância. Se você quiser criar um conjunto personalizado de grupos de segurança, escolha Limpar grupos de segurança. Escolha os grupos de segurança que você deseja.

9. Escolha Próximo.

Crie uma fila de trabalhos

Uma fila de trabalhos armazena seus trabalhos enviados até que o AWS Batch Agendador execute o trabalho em um recurso em seu ambiente computacional. Para criar uma fila de trabalhos:

Para criar uma fila de trabalhos para uma orquestração do Fargate, faça o seguinte:

1. Na seção Configuração da fila de trabalhos, em Nome, especifique um nome exclusivo para seu ambiente de computação. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).
2. Em Prioridade, insira um número inteiro entre 0 e 100 para a fila de trabalhos.

   > **Important**
   >
   > Valores inteiros mais altos são atribuídos a uma prioridade mais alta pelo AWS Batch Scheduler.

3. Escolha Próximo.

Crie uma definição de trabalho

Para criar a definição de trabalho:

1. Na seção Configuração geral:

   a. Em Nome, insira um nome de definição de trabalho personalizado.

      Na seção Configuração geral, em Nome, especifique um nome exclusivo para seu ambiente de computação. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).

   b. (Opcional) Em Tempo limite de execução, insira a quantidade de tempo (em segundos) após a qual um trabalho inacabado termina.

> **Important**
>
> O tempo limite mínimo é de 60 segundos.

c. (Opcional) Uma tag é um rótulo atribuído a um recurso. Para adicionar uma tag, expanda Tags e escolha Adicionar tag. Insira um par valor-chave e escolha Adicionar tag novamente.

> **Important**
>
> Se você escolher Adicionar tag, deverá inserir um par chave-valor e escolher Adicionar tag novamente ou escolher Remover tag.

d. (Opcional) Ative Propagar tags para propagar tags para o Amazon Elastic Container Service.

2. Na seção Configuração da plataforma Fargate:

a. (Opcional) Em Versão da plataforma Fargate, insira o ambiente de runtime específico que você deseja.

b. Em Plataforma de Runtime, selecione LINUX ou Windows.

c. (Somente Windows) Em Família de sistema operacional, selecione um sistema operacional.

d. Em Arquitetura de CPU, selecione a arquitetura de CPU que você deseja.

e. (Opcional) Ative a opção Atribuir IP público para atribuir um endereço IP público.

f. Em Armazenamento temporário, insira a quantidade de armazenamento temporário que você deseja.

> **Note**
>
> Por padrão, são usados 20 GiB de armazenamento temporário. Para usar armazenamento temporário adicional, insira um valor entre 21 GiB e 100 GiB.

g. Para Função de execução, escolha uma função de execução de tarefas que permita que agentes do Amazon Elastic Container Service (Amazon ECS) AWS façam chamadas em seu nome. Por exemplo, você pode escolher ecsTaskExecutionFunção.

3. Na seção Configuração do contêiner:

a. Em Imagem, insira o nome da imagem usada para iniciar o contêiner. Por padrão, todas as imagens no registro do Docker Hub estão disponíveis. Você também pode especificar outros repositórios no formato repository-url/image:tag. O parâmetro pode ter até 255 caracteres. Pode conter letras maiúsculas, minúsculas, números, hifens (-), sublinhados (_), dois pontos (:), pontos (.), barras (/) e jogos da velha (#). O parâmetro é mapeado para `Image` na seção Criar um contêiner da API remota do Docker e o parâmetro `IMAGE` de docker run.

> **Note**
>
> A arquitetura da imagem do Docker deve corresponder à arquitetura do processador dos recursos de computação nos quais eles foram programados. Por exemplo, imagens Arm baseadas Docker em só podem ser executadas em recursos de computação baseados em Arm.

- As imagens em repositórios públicos Amazon ECR usam convenções de nomenclatura completas `registry/repository[:tag]` ou `registry/repository[@digest]` (por exemplo, `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).
- As imagens em repositórios do Amazon ECR usam as convenções de nomenclatura `registry/repository:tag` completa (por exemplo, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*).
- As imagens em repositórios oficiais no Docker Hub usam um único nome (por exemplo, `ubuntu` ou `mongo`).
- As imagens em outros repositórios no Docker Hub são qualificadas com um nome de organização (por exemplo, `amazon/amazon-ecs-agent`).
- Imagens em outros repositórios online também são qualificadas por um nome de domínio (por exemplo, `quay.io/assemblyline/ubuntu`).

b. Para Command, especifique o comando a ser passado para o contêiner. Esse parâmetro é mapeado para `Cmd` na seção Criar um contêiner da Docker Remote API e o parâmetro `COMMAND` de docker run. Para obter mais informações sobre o parâmetro `CMD` da Docker, consulte https://docs.docker.com/engine/reference/builder/#cmd.

> **Note**
>
> Você pode usar os valores padrão de substituição de parâmetros e marcadores no seu comando. Para ter mais informações, consulte Parâmetros.

> **Tip**
>
> Escolha Info para revisar exemplos de códigos Bash e JSON.

c. (Opcional) Você pode adicionar parâmetros à definição do trabalho como mapeamentos de chave-valor para substituir os padrões de definição do trabalho. Para adicionar um parâmetro:

   - Em Parâmetros, escolha Adicionar parâmetro. Insira um par valores-chave e escolha Adicionar parâmetronovamente.

> **Important**
>
> Se você escolher Adicionar parâmetro, deverá configurar pelo menos um parâmetro ou escolher Remover parâmetro..

d. (Opcional) Na seção Configuração do ambiente para configuração da função Job, escolha uma função do IAM que forneça permissão para usar as AWS APIs.

e. Na seção Configuração do ambiente em vCPUs, especifique o número de vCPUs a serem reservadas para o contêiner. Esse parâmetro é mapeado para `CpuShares` na seção Criar um Contêiner da API remota do Docker e a opção `--cpu-shares` para docker run. Cada vCPU equivale a 1.024 compartilhamentos de CPU.

f. Em Memória, especifique o limite rígido (em MiB) de memória a ser apresentado ao contêiner do trabalho. Caso seu contêiner tente exceder a memória especificada, o mesmo será interrompido. Esse parâmetro é mapeado para `Memory` na seção Criar um Contêiner da API Remota Docker e para a opção `--memory` para docker run.

g. (Opcional) Em Variáveis de ambiente, escolha Adicionar variáveis de ambiente para adicionar variáveis de ambiente a serem passadas para o contêiner. Esse parâmetro é mapeado para `Env` na seção Criar um Contêiner da API remota do Docker e a opção `--env` para docker run.

4. Escolha Próximo.

Crie um trabalho

Para criar uma trabalho do Fargate, faça o seguinte:

1. Na seção Configuração do trabalho, em Nome, especifique um nome exclusivo para o trabalho. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).
2. Escolha Próximo.

Examinar e criar

Na página Revisar e criar, revise as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Criar recursos.

Começando a usar AWS Batch no Amazon EKS

AWS Batch on Amazon EKS é um serviço gerenciado para programar e escalar cargas de trabalho em lotes em clusters existentes do Amazon EKS. AWS Batch não cria, administra nem executa operações de ciclo de vida de seus clusters do Amazon EKS em seu nome. AWS Batch a orquestração aumenta e diminui os nós gerenciados AWS Batch e executa pods nesses nós.

AWS Batch não toca em nós, grupos de nós com escalabilidade automática ou ciclos de vida de pods que não estejam associados a ambientes AWS Batch computacionais em seu cluster Amazon EKS. AWS Batch Para operar de forma eficaz, sua função vinculada a serviços precisa de permissões de controle de acesso Kubernetes baseado em função (RBAC) em seu cluster Amazon EKS existente. Para obter mais informações, consulte Usar autorização RBAC na documentação do Kubernetes.

AWS Batch requer um Kubernetes namespace no qual possa incluir pods como AWS Batch trabalhos. Recomendamos um namespace dedicado para isolar os AWS Batch pods das outras cargas de trabalho do cluster.

Depois AWS Batch de receber acesso ao RBAC e estabelecer um namespace, você pode associar esse cluster Amazon EKS a um ambiente AWS Batch computacional usando a operação de API. CreateComputeEnvironment Uma fila de trabalhos pode ser associada a esse novo ambiente computacional do Amazon EKS. AWS Batch os trabalhos são enviados para a fila de trabalhos com

base em uma definição de trabalho do Amazon EKS usando a operação de SubmitJobAPI. AWS Batch em seguida, inicia os nós AWS Batch gerenciados e coloca os trabalhos da fila de trabalhos como Kubernetes pods no cluster EKS associado a um ambiente AWS Batch computacional.

As seções a seguir abordam como se configurar AWS Batch no Amazon EKS.

Sumário

Pré-requisitos

Antes de iniciar este tutorial, você deve instalar e configurar as seguintes ferramentas e recursos necessários para criar e gerenciar tanto os recursos do Amazon EKS AWS Batch quanto os do Amazon EKS.

- AWS CLI – Uma ferramenta de linha de comando para trabalhar com os serviços AWS , incluindo o Amazon EKS. Este guia exige que você use a versão 2.8.6 ou superior, ou a versão 1.26.0 ou superior. Para obter mais informações, consulte Como instalar, atualizar e desinstalar a AWS CLI no Guia do usuário da AWS Command Line Interface . Depois de instalar o AWS

CLI, recomendamos que você também o configure. Para obter mais informações, consulte Configuração rápida com o `aws configure` no Manual do usuário do AWS Command Line Interface .

- **`kubectl`**: uma ferramenta de linha de comando para trabalhar com clusters do Kubernetes. Este guia requer que você use a versão `1.23` ou superior. Para obter mais informações, consulte Instalar ou atualizar o `kubectl` no Guia do usuário do Amazon EKS.
- **`eksctl`**— Uma ferramenta de linha de comando para trabalhar com clusters do Amazon EKS que automatiza muitas tarefas individuais. Este guia requer que você use a versão `0.115.0` ou superior. Para obter mais informações, consulte Instalar ou atualizar o `eksctl` no Guia do usuário do Amazon EKS.
- Permissões de IAM necessárias — O diretor de segurança do IAM que você está usando deve ter permissões para trabalhar com funções do IAM e funções vinculadas a serviços do Amazon EKS AWS CloudFormation, além de uma VPC e recursos relacionados. Para obter mais informações, consulte Ações, recursos e chaves de condição para o Amazon Elastic Kubernetes Service e Uso de funções vinculadas a serviços no Guia do usuário do IAM. Você deve concluir todas as etapas deste manual como o mesmo usuário.
- Criação de um cluster do Amazon EKS — Para obter mais informações, consulte Introdução ao Amazon EKS — `eksctl` no Guia do usuário do Amazon EKS.

> **Note**
>
> AWS Batch só é compatível com clusters do Amazon EKS com endpoints de servidor de API que têm acesso público, acessível à Internet pública. Por padrão, os endpoints do servidor de API dos clusters Amazon EKS têm acesso público. Para obter mais informações, consulte o Controle de acesso ao endpoint do cluster do Amazon EKS no Guia do Usuário do Amazon EKS.

> **Note**
>
> AWS Batch não fornece orquestração de nós gerenciados para CoreDNS ou outros pods de implantação. Se você precisar do CoreDNS, consulte Adicionar o complemento CoreDNS do Amazon EKS no Guia do usuário do Amazon EKS. Ou use `eksctl create cluster create` para criar o cluster. Ele inclui o CoreDNS por padrão.

- Permissões — Os usuários que chamam a operação da CreateComputeEnvironmentAPI para criar um ambiente computacional que usa os recursos do Amazon EKS precisam de permissões para a operação da `eks:DescribeCluster` API. Usar o AWS Management Console para criar um recurso computacional usando os recursos do Amazon EKS requer permissões para `eks:DescribeCluster` e. `eks:ListClusters`

Etapa 1: Preparando seu cluster Amazon EKS para AWS Batch

Todas as etapas são necessárias.

1. Crie um namespace dedicado para trabalhos AWS Batch

   Use o `kubectl` para criar um namespace.

   ```
   $ namespace=my-aws-batch-namespace
   $ cat - <<EOF | kubectl create -f -
   {
     "apiVersion": "v1",
     "kind": "Namespace",
     "metadata": {
       "name": "${namespace}",
       "labels": {
         "name": "${namespace}"
       }
     }
   }
   EOF
   ```

   Saída:

   ```
   namespace/my-aws-batch-namespace created
   ```

2. Habilite o acesso por meio do controle de acesso com base em perfil (RBAC)

   Use `kubectl` para criar uma Kubernetes função para o cluster AWS Batch para permitir observar nós e pods e vincular a função. Você deve fazer isso uma vez para cada cluster do EKS.

> **Note**
>
> Para obter mais informações sobre como usar a autorização RBAC, consulte Usando a autorização RBAC no Guia do usuário. Kubernetes

```
$ cat - <<EOF | kubectl apply -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: aws-batch-cluster-role
rules:
  - apiGroups: [""]
    resources: ["namespaces"]
    verbs: ["get"]
  - apiGroups: [""]
    resources: ["nodes"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["configmaps"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["apps"]
    resources: ["daemonsets", "deployments", "statefulsets", "replicasets"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["clusterroles", "clusterrolebindings"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: aws-batch-cluster-role-binding
subjects:
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
```

```
  name: aws-batch-cluster-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

Saída:

```
clusterrole.rbac.authorization.k8s.io/aws-batch-cluster-role created
clusterrolebinding.rbac.authorization.k8s.io/aws-batch-cluster-role-binding created
```

Crie uma Kubernetes função com escopo de namespace para gerenciar e fazer o ciclo de vida dos AWS Batch pods e vinculá-los. Você deve fazer isso uma vez para cada namespace exclusivo.

```
$ namespace=my-aws-batch-namespace
$ cat - <<EOF | kubectl apply -f - --namespace "${namespace}"
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: aws-batch-compute-environment-role
  namespace: ${namespace}
rules:
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["create", "get", "list", "watch", "delete", "patch"]
  - apiGroups: [""]
    resources: ["serviceaccounts"]
    verbs: ["get", "list"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["roles", "rolebindings"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: aws-batch-compute-environment-role-binding
  namespace: ${namespace}
subjects:
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
```

```
  name: aws-batch-compute-environment-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

Saída:

```
role.rbac.authorization.k8s.io/aws-batch-compute-environment-role created
rolebinding.rbac.authorization.k8s.io/aws-batch-compute-environment-role-binding
 created
```

Atualize o mapa Kubernetes `aws-auth` de configuração para mapear as permissões anteriores do RBAC para a função vinculada ao serviço. AWS Batch

```
$ eksctl create iamidentitymapping \
    --cluster my-cluster-name \
    --arn "arn:aws:iam::<your-account>:role/AWSServiceRoleForBatch" \
    --username aws-batch
```

Saída:

```
2022-10-25 20:19:57 [#]  adding identity "arn:aws:iam::<your-account>:role/
AWSServiceRoleForBatch" to auth ConfigMap
```

> **Note**
>
> O caminho `aws-service-role/batch.amazonaws.com/` foi removido do ARN so perfil vinculado ao serviço. Isso ocorre devido a um problema com o mapa de configuração `aws-auth`. Para obter mais informações, consulte Funções com caminhos não funcionam quando o caminho é incluído em seu ARN no. aws-authconfigmap

Etapa 2: Criação de um ambiente de computação do Amazon EKS

AWS Batch ambientes computacionais definem parâmetros de recursos computacionais para atender às suas necessidades de carga de trabalho em lotes. Em um ambiente computacional gerenciado, AWS Batch ajuda você a gerenciar a capacidade e os tipos de instância dos recursos computacionais (Kubernetesnós) dentro do seu cluster Amazon EKS. Isso se baseia na

especificação do recurso de computação que você define ao criar o ambiente de computação. Você pode usar instâncias sob demanda do EC2 ou instâncias spot do EC2.

Agora que a função AWSServiceRoleForBatchvinculada ao serviço tem acesso ao seu cluster Amazon EKS, você pode criar AWS Batch recursos. Primeiro, crie um ambiente computacional que aponte para seu cluster do Amazon EKS.

```
$ cat <<EOF > ./batch-eks-compute-environment.json
{
  "computeEnvironmentName": "My-Eks-CE1",
  "type": "MANAGED",
  "state": "ENABLED",
  "eksConfiguration": {
    "eksClusterArn": "arn:aws:eks:<region>:123456789012:cluster/<cluster-name>",
    "kubernetesNamespace": "my-aws-batch-namespace"
  },
  "computeResources": {
    "type": "EC2",
    "allocationStrategy": "BEST_FIT_PROGRESSIVE",
    "minvCpus": 0,
    "maxvCpus": 128,
    "instanceTypes": [
        "m5"
    ],
    "subnets": [
        "<eks-cluster-subnets-with-access-to-internet-for-image-pull>"
    ],
    "securityGroupIds": [
        "<eks-cluster-sg>"
    ],
    "instanceRole": "<eks-instance-profile>"
  }
}
EOF
$ aws batch create-compute-environment --cli-input-json file://./batch-eks-compute-
environment.json
```

Observações

- O `serviceRole` parâmetro não deve ser especificado e, em seguida, a função AWS Batch vinculada ao serviço será usada. AWS Batch no Amazon EKS só oferece suporte à função AWS Batch vinculada ao serviço.

- Somente`BEST_FIT_PROGRESSIVE`,`SPOT_CAPACITY_OPTIMIZED`, e as estratégias de `SPOT_PRICE_CAPACITY_OPTIMIZED` alocação são suportadas para ambientes computacionais Amazon EKS.

  > **Note**
  >
  > Em vez disso, recomendamos utilizar `SPOT_PRICE_CAPACITY_OPTIMIZED` em vez de `SPOT_CAPACITY_OPTIMIZED` na maioria das instâncias.

- Para o `instanceRole`, consulte Criar o perfil do IAM do nó do Amazon EKS e Habilitar o acesso da entidade principal do IAM ao seu cluster no Guia do usuário do Amazon EKS. Se você estiver usando redes de pods, consulte Como configurar o plugin CNI da Amazon VPC para o Kubernetes para usar perfis do IAM para contas de serviço no Guia do usuário do Amazon EKS.
- Uma forma de fazer com que as sub-redes funcionem para o parâmetro `subnets` é usar as sub-redes públicas dos grupos de nós gerenciados do Amazon EKS que foram criadas pelo `eksctl` ao criar um cluster do Amazon EKS. Caso contrário, use sub-redes que tenham um caminho de rede compatível com a extração de imagens.
- O parâmetro `securityGroupIds` pode usar o mesmo grupo de segurança do cluster do Amazon EKS. Esse comando recupera o ID do grupo de segurança do cluster.

```
$ eks describe-cluster \
    --name <cluster-name> \
    --query cluster.resourcesVpcConfig.clusterSecurityGroupId
```

- A manutenção de um ambiente computacional Amazon EKS é uma responsabilidade compartilhada. Para ter mais informações, consulte Responsabilidade compartilhada dos nós Kubernetes.

> **Important**
>
> É importante confirmar se o ambiente de computação está íntegro antes de continuar. A operação DescribeComputeEnvironmentsda API pode ser usada para fazer isso.
>
> ```
> $ aws batch describe-compute-environments --compute-environments My-Eks-CE1
> ```

Confirme se o parâmetro `status` não está `INVALID`. Se estiver, veja o parâmetro `statusReason` para saber a causa. Para ter mais informações, consulte Solução de problemas AWS Batch.

Etapa 3: Criar uma fila de trabalhos e anexar o ambiente de computação

```
$ aws batch describe-compute-environments --compute-environments My-Eks-CE1
```

Os trabalhos enviados para essa nova fila de trabalhos são executados como pods em nós AWS Batch gerenciados que se juntaram ao cluster Amazon EKS associado ao seu ambiente computacional.

```
$ cat <<EOF > ./batch-eks-job-queue.json
 {
    "jobQueueName": "My-Eks-JQ1",
    "priority": 10,
    "computeEnvironmentOrder": [
      {
        "order": 1,
        "computeEnvironment": "My-Eks-CE1"
      }
    ]
  }
EOF
$ aws batch create-job-queue --cli-input-json file://./batch-eks-job-queue.json
```

Etapa 4: Criar uma definição de trabalho

```
$ cat <<EOF > ./batch-eks-job-definition.json
{
  "jobDefinitionName": "MyJobOnEks_Sleep",
  "type": "container",
  "eksProperties": {
    "podProperties": {
      "hostNetwork": true,
      "containers": [
        {
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
```

```
          "command": [
            "sleep",
            "60"
          ],
          "resources": {
            "limits": {
              "cpu": "1",
              "memory": "1024Mi"
            }
          }
        }
      ],
      "metadata": {
        "labels": {
          "environment": "test"
        }
      }
    }
  }
}
EOF
$ aws batch register-job-definition --cli-input-json file://./batch-eks-job-
definition.json
```

Observações

- Somente trabalhos em um único contêiner são suportados.
- Há considerações sobre os parâmetros `cpu` e `memory`. Para ter mais informações, consulte Considerações sobre memória e vCPU para AWS Batch no Amazon EKS.

Etapa 5: Enviar um trabalho

```
$ aws batch submit-job --job-queue My-Eks-JQ1 \
    --job-definition MyJobOnEks_Sleep --job-name My-Eks-Job1
$ aws batch describe-jobs --job <jobId-from-submit-response>
```

Observações

- Somente trabalhos em um único contêiner são suportados.

- Certifique-se de estar familiarizado com todas as considerações relevantes para os parâmetros `cpu` e `memory`. Para ter mais informações, consulte Considerações sobre memória e vCPU para AWS Batch no Amazon EKS.
- Para obter mais informações sobre a execução de trabalhos nos recursos do Amazon EKS, consulteTrabalhos do Amazon EKS.

(Opcional) Envie um trabalho com substituições

Esse trabalho substitui o comando passado para o contêiner.

```
$ cat <<EOF > ./submit-job-override.json
{
  "jobName": "EksWithOverrides",
  "jobQueue": "My-Eks-JQ1",
  "jobDefinition": "MyJobOnEks_Sleep",
  "eksPropertiesOverride": {
    "podProperties": {
      "containers": [
        {
          "command": [
            "/bin/sh"
          ],
          "args": [
            "-c",
            "echo hello world"
          ]
        }
      ]
    }
  }
}
EOF
$ aws batch submit-job --cli-input-json file://./submit-job-override.json
```

Observações

- AWS Batch limpa agressivamente os frutos após a conclusão dos trabalhos para reduzir a carga para. Kubernetes Para examinar os detalhes de um trabalho, o log deve ser configurado. Para ter mais informações, consulte Use o CloudWatch Logs para monitorar AWS Batch trabalhos do Amazon EKS.

- Para melhorar a visibilidade dos detalhes das operações, ative o log do ambiente de gerenciamento Amazon EKS. Para obter mais informações, consulte Logs do ambiente de gerenciamento do Amazon EKS no Guia do usuário do Amazon EKS.
- A sobrecarga de Daemonsets e kubelets afeta os recursos de vCPU e memória disponíveis, especificamente a escalabilidade e o posicionamento do trabalho. Para ter mais informações, consulte Considerações sobre memória e vCPU para AWS Batch no Amazon EKS.

Introdução aos AWS Batch clusters privados do Amazon EKS

AWS Batch é um serviço gerenciado que orquestra cargas de trabalho em lotes em seus clusters do Amazon Elastic Kubernetes Service (Amazon EKS). Isso inclui filas, rastreamento de dependências, tentativas e prioridades gerenciadas de tarefas, gerenciamento de pods e escalabilidade de nós. Esse recurso conecta seu cluster privado existente do Amazon EKS AWS Batch para executar seus trabalhos em grande escala. Você pode usar `eksctl`(uma interface de linha de comando para o Amazon EKS), o AWS console ou o AWS Command Line Interfacepara criar um cluster privado do Amazon EKS com todos os outros recursos necessários. Support para clusters privados do Amazon EKS no Amazon geralmente AWS Batch está disponível no mercado comercial, Regiões da AWS onde AWS Batch está disponível.

Os clusters somente privados do Amazon EKS não têm acesso de entrada/saída à Internet e têm apenas sub-redes privadas. Os endpoints do Amazon VPC são usados para permitir o acesso privado a outros serviços. AWS `eksctl`suporta a criação de clusters totalmente privados usando uma Amazon VPC e sub-redes preexistentes. `eksctl`também cria endpoints da Amazon VPC na Amazon VPC fornecida e modifica as tabelas de rotas para as sub-redes fornecidas.

Cada sub-rede deve ter uma tabela de rotas explícita associada a ela porque `eksctl` não modifica a tabela de rotas principal. Seu cluster deve extrair imagens de um registro de contêiner que esteja em sua Amazon VPC. Além disso, você pode criar um Amazon Elastic Container Registry em sua Amazon VPC e copiar imagens de contêiner nele para que seus nós possam extrair. Para obter mais informações, consulte Copiar uma imagem de contêiner de um repositório para outro. Para começar a usar os repositórios privados do Amazon ECR, consulte Repositórios privados do Amazon ECR.

Opcionalmente, você pode criar uma regra de cache pull through com o Amazon ECR. Depois que uma regra de cache pull through é criada para um registro público externo, você pode extrair uma imagem desse registro público externo usando o identificador de recursos uriform (URI) do registro privado do Amazon ECR. Em seguida, o Amazon ECR cria um repositório e armazena a imagem em cache. Quando uma imagem em cache é extraída usando o URI de registro privado do

Amazon ECR, o Amazon ECR verifica o registro remoto para ver se há uma nova versão da imagem e atualiza seu registro privado até uma vez a cada 24 horas.

Sumário

Pré-requisitos

Antes de iniciar este tutorial, você deve instalar e configurar as seguintes ferramentas e recursos necessários para criar e gerenciar tanto os recursos do Amazon EKS AWS Batch quanto os do Amazon EKS. Você também precisa criar todos os recursos necessários, incluindo VPC, sub-redes, tabelas de rotas, VPC endpoints e cluster Amazon EKS. Você precisa usar AWS CLI o.

- AWS CLI— Uma ferramenta de linha de comando para trabalhar com AWS serviços, incluindo o Amazon EKS. Este guia exige que você use a versão 2.8.6 ou superior, ou a versão 1.26.0 ou superior. Para obter mais informações, consulte Como instalar, atualizar e desinstalar a AWS CLI no Guia do usuário da AWS Command Line Interface .

  Depois de instalar o AWS CLI, recomendamos que você o configure. Para obter mais informações, consulte Configuração rápida com o `aws configure` no Manual do usuário do AWS Command Line Interface .
- **`kubectl`**— Uma ferramenta de linha de comando para trabalhar com Kubernetes clusters. Este guia requer que você use a versão `1.23` ou superior. Para obter mais informações, consulte Instalar ou atualizar o `kubectl` no Guia do usuário do Amazon EKS.
- **`eksctl`**— Uma ferramenta de linha de comando para trabalhar com clusters do Amazon EKS que automatiza muitas tarefas individuais. Este guia requer que você use a versão `0.115.0` ou superior. Para obter mais informações, consulte Instalar ou atualizar o `eksctl` no Guia do usuário do Amazon EKS.

- Permissões obrigatórias AWS Identity and Access Management (IAM) — O diretor de segurança do IAM que você está usando deve ter permissões para trabalhar com funções do IAM e funções vinculadas a serviços do Amazon EKS AWS CloudFormation, além de uma VPC e recursos relacionados. Para obter mais informações, consulte Ações, recursos e chaves de condição para o Amazon Elastic Kubernetes Service e Uso de funções vinculadas a serviços no Guia do usuário do IAM. Você deve concluir todas as etapas deste manual como o mesmo usuário.
- Criação de um cluster do Amazon EKS — Para obter mais informações, consulte Introdução ao Amazon EKS — `eksctl` no Guia do usuário do Amazon EKS.

  > **Note**
  >
  > AWS Batch não fornece orquestração de nós gerenciados para CoreDNS ou outros pods de implantação. Se você precisar do CoreDNS, consulte Adicionar o complemento CoreDNS do Amazon EKS no Guia do usuário do Amazon EKS. Ou use `eksctl create cluster create` para criar o cluster. Ele inclui o CoreDNS por padrão.

- Permissões — Os usuários que chamam a operação da CreateComputeEnvironmentAPI para criar um ambiente computacional que usa os recursos do Amazon EKS precisam de permissões para a operação da `eks:DescribeCluster` API. Usar o AWS Management Console para criar um recurso computacional usando os recursos do Amazon EKS requer permissões para `eks:DescribeCluster` e. `eks:ListClusters`
- Crie um cluster EKS privado na região us-east-1 usando o arquivo de `eksctl` configuração de amostra.

  ```
  kind: ClusterConfig
  apiVersion: eksctl.io/v1alpha5
  availabilityZones:
      - us-east-1a
      - us-east-1b
      - us-east-1d
  managedNodeGroups:
      privateNetworking: true
  privateCluster:
      enabled: true
      skipEndpointCreation: false
  ```

  Crie seus recursos usando o comando: `eksctl create cluster -f clusterConfig.yaml`

- Os nós gerenciados em lote devem ser implantados em sub-redes que tenham os endpoints de interface VPC de que você precisa. Para obter mais informações, consulte Requisitos de cluster privado.

Etapa 1: Preparando seu cluster EKS para AWS Batch

Todas as etapas são necessárias.

1. Crie um namespace dedicado para trabalhos AWS Batch

   Use o `kubectl` para criar um namespace.

   ```
   $ namespace=my-aws-batch-namespace
   $ cat - <<EOF | kubectl create -f -
   {
     "apiVersion": "v1",
     "kind": "Namespace",
     "metadata": {
       "name": "${namespace}",
       "labels": {
         "name": "${namespace}"
       }
     }
   }
   EOF
   ```

   Saída:

   ```
   namespace/my-aws-batch-namespace created
   ```

2. Habilite o acesso por meio do controle de acesso com base em perfil (RBAC)

   Use o `kubectl` para criar uma função do Kubernetes para o cluster do AWS Batch para permitir observar nós e pods e vincular a função. Você deve fazer isso uma vez para cada cluster do Amazon EKS.

   > **Note**
   >
   > Para obter mais informações sobre o uso de autorização RBAC, consulte Usar a autorização RBAC na documentação do Kubernetes.

```
$ cat - <<EOF | kubectl apply -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: aws-batch-cluster-role
rules:
  - apiGroups: [""]
    resources: ["namespaces"]
    verbs: ["get"]
  - apiGroups: [""]
    resources: ["nodes"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["get", "list", "watch"]
  - apiGroups: [""]
    resources: ["configmaps"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["apps"]
    resources: ["daemonsets", "deployments", "statefulsets", "replicasets"]
    verbs: ["get", "list", "watch"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["clusterroles", "clusterrolebindings"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: aws-batch-cluster-role-binding
subjects:
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: aws-batch-cluster-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

Saída:

```
clusterrole.rbac.authorization.k8s.io/aws-batch-cluster-role created
clusterrolebinding.rbac.authorization.k8s.io/aws-batch-cluster-role-binding created
```

Crie uma Kubernetes função com escopo de namespace para gerenciar e fazer o ciclo de vida dos AWS Batch pods e vinculá-los. Você deve fazer isso uma vez para cada namespace exclusivo.

```
$ namespace=my-aws-batch-namespace
$ cat - <<EOF | kubectl apply -f - --namespace "${namespace}"
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: aws-batch-compute-environment-role
  namespace: ${namespace}
rules:
  - apiGroups: [""]
    resources: ["pods"]
    verbs: ["create", "get", "list", "watch", "delete", "patch"]
  - apiGroups: [""]
    resources: ["serviceaccounts"]
    verbs: ["get", "list"]
  - apiGroups: ["rbac.authorization.k8s.io"]
    resources: ["roles", "rolebindings"]
    verbs: ["get", "list"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: aws-batch-compute-environment-role-binding
  namespace: ${namespace}
subjects:
- kind: User
  name: aws-batch
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: aws-batch-compute-environment-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

Saída:

```
role.rbac.authorization.k8s.io/aws-batch-compute-environment-role created
rolebinding.rbac.authorization.k8s.io/aws-batch-compute-environment-role-binding
 created
```

Atualize o mapa Kubernetes `aws-auth` de configuração para mapear as permissões anteriores do RBAC para a função vinculada ao serviço. AWS Batch

```
$ eksctl create iamidentitymapping \
    --cluster my-cluster-name \
    --arn "arn:aws:iam::<your-account>:role/AWSServiceRoleForBatch" \
    --username aws-batch
```

Saída:

```
2022-10-25 20:19:57 [#]  adding identity "arn:aws:iam::<your-account>:role/
AWSServiceRoleForBatch" to auth ConfigMap
```

> **Note**
>
> O caminho `aws-service-role/batch.amazonaws.com/` foi removido do ARN so perfil vinculado ao serviço. Isso ocorre devido a um problema com o mapa de configuração `aws-auth`. Para obter mais informações, consulte Funções com caminhos não funcionam quando o caminho é incluído em seu ARN no. aws-authconfigmap

Etapa 2: Criação de um ambiente de computação do Amazon EKS

AWS Batch ambientes computacionais definem parâmetros de recursos computacionais para atender às suas necessidades de carga de trabalho em lotes. Em um ambiente computacional gerenciado, AWS Batch ajuda você a gerenciar a capacidade e os tipos de instância dos recursos computacionais (Kubernetesnós) dentro do seu cluster Amazon EKS. Isso se baseia na especificação do recurso de computação que você define ao criar o ambiente de computação. Você pode usar instâncias sob demanda do EC2 ou instâncias spot do EC2.

Agora que a função AWSServiceRoleForBatchvinculada ao serviço tem acesso ao seu cluster Amazon EKS, você pode criar AWS Batch recursos. Primeiro, crie um ambiente computacional que aponte para seu cluster do Amazon EKS.

```
$ cat <<EOF > ./batch-eks-compute-environment.json
{
  "computeEnvironmentName": "My-Eks-CE1",
  "type": "MANAGED",
  "state": "ENABLED",
  "eksConfiguration": {
    "eksClusterArn": "arn:aws:eks:<region>:123456789012:cluster/<cluster-name>",
    "kubernetesNamespace": "my-aws-batch-namespace"
  },
  "computeResources": {
    "type": "EC2",
    "allocationStrategy": "BEST_FIT_PROGRESSIVE",
    "minvCpus": 0,
    "maxvCpus": 128,
    "instanceTypes": [
        "m5"
    ],
    "subnets": [
        "<eks-cluster-subnets-with-access-to-the-image-for-image-pull>"
    ],
    "securityGroupIds": [
        "<eks-cluster-sg>"
    ],
    "instanceRole": "<eks-instance-profile>"
  }
}
EOF
$ aws batch create-compute-environment --cli-input-json file://./batch-eks-compute-
environment.json
```

Observações

- O `serviceRole` parâmetro não deve ser especificado e, em seguida, a função AWS Batch vinculada ao serviço será usada. AWS Batch no Amazon EKS só oferece suporte à função AWS Batch vinculada ao serviço.
- Somente`BEST_FIT_PROGRESSIVE`,`SPOT_CAPACITY_OPTIMIZED`, e as estratégias de `SPOT_PRICE_CAPACITY_OPTIMIZED` alocação são suportadas para ambientes computacionais Amazon EKS.

> **Note**
>
> Recomendamos utilizar `SPOT_PRICE_CAPACITY_OPTIMIZED` em vez de `SPOT_CAPACITY_OPTIMIZED` na maioria das instâncias.

- Para o `instanceRole`, consulte Criar o perfil do IAM do nó do Amazon EKS e Habilitar o acesso da entidade principal do IAM ao seu cluster no Guia do usuário do Amazon EKS. Se você estiver usando redes de pods, consulte Como configurar o plugin CNI da Amazon VPC para o Kubernetes para usar perfis do IAM para contas de serviço no Guia do usuário do Amazon EKS.
- Uma forma de fazer com que as sub-redes funcionem para o parâmetro `subnets` é usar as sub-redes públicas dos grupos de nós gerenciados do Amazon EKS que foram criadas pelo `eksctl` ao criar um cluster do Amazon EKS. Caso contrário, use sub-redes que tenham um caminho de rede compatível com a extração de imagens.
- O parâmetro `securityGroupIds` pode usar o mesmo grupo de segurança do cluster do Amazon EKS. Esse comando recupera o ID do grupo de segurança do cluster.

```
$ eks describe-cluster \
    --name <cluster-name> \
    --query cluster.resourcesVpcConfig.clusterSecurityGroupId
```

- A manutenção de um ambiente computacional Amazon EKS é uma responsabilidade compartilhada. Para obter mais informações, consulte Segurança no Amazon EKS.

> **Important**
>
> É importante confirmar se o ambiente de computação está íntegro antes de continuar. A operação DescribeComputeEnvironmentsda API pode ser usada para fazer isso.
>
> ```
> $ aws batch describe-compute-environments --compute-environments My-Eks-CE1
> ```
>
> Confirme se o parâmetro `status` não está `INVALID`. Se estiver, veja o parâmetro `statusReason` para saber a causa. Para ter mais informações, consulte Solução de problemas AWS Batch.

Etapa 3: Criar uma fila de trabalhos e anexar o ambiente de computação

```
$ aws batch describe-compute-environments --compute-environments My-Eks-CE1
```

Os trabalhos enviados para essa nova fila de trabalhos são executados como pods em nós AWS Batch gerenciados que se juntaram ao cluster Amazon EKS associado ao seu ambiente computacional.

```
$ cat <<EOF > ./batch-eks-job-queue.json
 {
    "jobQueueName": "My-Eks-JQ1",
    "priority": 10,
    "computeEnvironmentOrder": [
      {
        "order": 1,
        "computeEnvironment": "My-Eks-CE1"
      }
    ]
  }
EOF
$ aws batch create-job-queue --cli-input-json file://./batch-eks-job-queue.json
```

Etapa 4: Criar uma definição de trabalho

No campo de imagem da definição do trabalho, em vez de fornecer um link para a imagem em um repositório ECR público, forneça o link para a imagem armazenada em nosso repositório ECR privado. Veja o exemplo de definição de tarefa a seguir:

```
$ cat <<EOF > ./batch-eks-job-definition.json
{
  "jobDefinitionName": "MyJobOnEks_Sleep",
  "type": "container",
  "eksProperties": {
    "podProperties": {
      "hostNetwork": true,
      "containers": [
        {
          "image": "account-id.dkr.ecr.region.amazonaws.com/amazonlinux:2",
          "command": [
            "sleep",
            "60"
```

```
          ],
          "resources": {
            "limits": {
              "cpu": "1",
              "memory": "1024Mi"
            }
          }
        }
      ],
      "metadata": {
        "labels": {
          "environment": "test"
        }
      }
    }
  }
}
EOF
$ aws batch register-job-definition --cli-input-json file://./batch-eks-job-
definition.json
```

Para executar comandos kubectl, você precisará de acesso privado ao seu cluster Amazon EKS. Isso significa que todo o tráfego para seu servidor de API de cluster deve vir da VPC do seu cluster ou de uma rede conectada.

Etapa 5: Enviar um trabalho

```
$ aws batch submit-job - -job-queue My-Eks-JQ1 \
    - -job-definition MyJobOnEks_Sleep - -job-name My-Eks-Job1
$ aws batch describe-jobs - -job <jobId-from-submit-response>
```

Observações

- Somente trabalhos em um único contêiner são suportados.
- Certifique-se de estar familiarizado com todas as considerações relevantes para os parâmetros `cpu` e `memory`. Para ter mais informações, consulte Considerações sobre memória e vCPU para AWS Batch no Amazon EKS.
- Para obter mais informações sobre a execução de trabalhos nos recursos do Amazon EKS, consulteTrabalhos do Amazon EKS.

(Opcional) Envie um trabalho com substituições

Esse trabalho substitui o comando passado para o contêiner.

```
$ cat <<EOF > ./submit-job-override.json
{
  "jobName": "EksWithOverrides",
  "jobQueue": "My-Eks-JQ1",
  "jobDefinition": "MyJobOnEks_Sleep",
  "eksPropertiesOverride": {
    "podProperties": {
      "containers": [
        {
          "command": [
            "/bin/sh"
          ],
          "args": [
            "-c",
            "echo hello world"
          ]
        }
      ]
    }
  }
}
EOF
$ aws batch submit-job - -cli-input-json file://./submit-job-override.json
```

Observações

- AWS Batch limpa agressivamente os frutos após a conclusão dos trabalhos para reduzir a carga para. Kubernetes Para examinar os detalhes de um trabalho, o log deve ser configurado. Para ter mais informações, consulte Use o CloudWatch Logs para monitorar AWS Batch trabalhos do Amazon EKS.
- Para melhorar a visibilidade dos detalhes das operações, ative o log do ambiente de gerenciamento Amazon EKS. Para obter mais informações, consulte Logs do ambiente de gerenciamento do Amazon EKS no Guia do usuário do Amazon EKS.
- A sobrecarga de Daemonsets e kubelets afeta os recursos de vCPU e memória disponíveis, especificamente a escalabilidade e o posicionamento do trabalho. Para ter mais informações, consulte Considerações sobre memória e vCPU para AWS Batch no Amazon EKS.

Solução de problemas

Se os nós lançados por AWS Batch não tiverem acesso ao repositório Amazon ECR (ou a qualquer outro repositório) que armazena sua imagem, seus trabalhos poderão permanecer no estado STARTING. Isso ocorre porque o pod não conseguirá baixar a imagem e executar seu AWS Batch trabalho. Se você clicar no nome do pod lançado por AWS Batch , poderá ver a mensagem de erro e confirmar o problema. A mensagem de erro deve ser semelhante à seguinte:

```
Failed to pull image "public.ecr.aws/amazonlinux/amazonlinux:2": rpc error: code =
Unknown desc = failed to pull and unpack image
"public.ecr.aws/amazonlinux/amazonlinux:2": failed to resolve reference
"public.ecr.aws/amazonlinux/amazonlinux:2": failed to do request: Head
"https://public.ecr.aws/v2/amazonlinux/amazonlinux/manifests/2": dial tcp: i/o timeout
```

Para outros cenários comuns de solução de problemas, consulte Solução de problemas AWS Batch. Para obter informações sobre a solução de problemas com base no status do pod, consulte Como eu soluciono o status do pod no Amazon EKS? .

Tarefas

Empregos são a unidade de trabalho iniciada por AWS Batch. Os trabalhos podem ser invocados como aplicativos em contêineres executáveis em instâncias de contêiner do ECS em um cluster do ECS.

Os trabalhos em contêineres podem fazer referência a uma imagem de contêiner, comando e parâmetros de contêiner. Para ter mais informações, consulte Parâmetros de definição de trabalho para ContainerProperties.

Você pode enviar um grande número de trabalhos simples e independentes.

Tópicos

Enviando um trabalho

Depois de registrar uma definição de trabalho, você pode enviá-la como um trabalho para uma fila de AWS Batch trabalhos. Você pode substituir muitos dos parâmetros especificados no campo de definição de trabalho em runtime.

Para enviar um trabalho

1. Abra o AWS Batch console em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione o Região da AWS a ser usado.
3. No painel de navegação, escolha Tarefas.
4. Escolha Enviar Novo Trabalho.
5. Em Nome, insira um nome exclusivo para a sua definição de trabalho. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).
6. Para Definição de Trabalho, escolha uma definição de trabalho existente para o seu trabalho. Para ter mais informações, consulte Como criar uma definição de tarefa de nó único .
7. Para Fila de Trabalhos, escolha uma fila de trabalhos existente. Para ter mais informações, consulte Como criar uma fila de tarefas.
8. Em Dependências do Trabalho, escolha Adicionar Dependência do Trabalho .
   - Em ID do Trabalho, insira a ID do trabalho para todas as dependências. Em seguida, escolha Adicionar Dependências do Trabalho. Um trabalho pode ter até 20 dependências. Para ter mais informações, consulte Dependências do trabalho.
9. (Somente para trabalhos de matriz) Em Tamanho da Matriz, especifique um tamanho entre 2 e 10.000.
10. (Opcional) Expanda Tags e então, escolha Adicionar Tag para adicionar tags ao recurso. Insira uma chave e um valor opcional e então escolha Adicionar Tag.
11. Escolha Próxima página.
12. Na seção Substituições de Trabalho:
    a. (Opcional) Em Prioridade de agendamento, insira um valor de prioridade de agendamento entre 0 e 100. Valores mais altos têm maior prioridade.
    b. (Opcional) Para Tentativas de Trabalho, insira o número máximo de vezes que AWS Batch tenta mover o trabalho para um status `RUNNABLE`. Você pode inserir um número inteiro entre 1 e 10. Para ter mais informações, consulte Repetições de trabalho automatizadas.
    c. (Opcional) Em Tempo Limite de Execução, insira o valor do tempo limite (em segundos). O tempo limite de execução é o período de tempo antes que um trabalho não concluído seja encerrado. Se uma tentativa exceder o tempo limite de duração, ela será parada e o status

será alterado para `FAILED`. Para ter mais informações, consulte Tempos limite do trabalho. O valor mínimo é 60 segundos.

> **Important**
>
> Não confie em trabalhos executados com recursos do Fargate por mais de 14 dias. Depois de 14 dias, os recursos do Fargate podem não estar mais disponíveis e é provável que o trabalho ser encerrado.

d. (Opcional) Ative Propagar Tags para propagar tags do trabalho e da definição de trabalho para a tarefa do Amazon ECS.

13. Expanda Configuração Adicional.
14. (Opcional) Em Repetir Condições de Estratégia, escolha Adicionar Avaliação na Saída. Insira pelo menos um valor de parâmetro e escolha uma Ação. Para cada conjunto de condições, Ação deve ser definida como Tentar Novamente ou Sair. Essas ações significam o seguinte:

    - Tentar novamente — AWS Batch tenta novamente até que o número de tentativas de trabalho que você especificou seja atingido.
    - Sair — AWS Batch para de tentar novamente o trabalho.

    > **Important**
    >
    > Se você escolher Adicionar Avaliação na Saída, configure pelo menos um parâmetro e uma Ação, ou escolha Remover Avaliação na Saída.

15. Em Parâmetros, escolha Adicionar Parâmetros para adicionar espaços reservados de substituição de parâmetros. Em seguida, insira uma Chave e um Valor opcional.
16. Na seção Substituição de Contêineres:

    a. Para Command, especifique o comando a ser passado para o contêiner. Para comandos simples, insira o comando da mesma forma que você faz para um mensagem de comando. Para comandos mais complicados, por exemplo, com caracteres especiais, utilize a sintaxe JSON.

> **Note**
>
> O parâmetro não pode conter uma string vazia.

b. Para vCPUs, especifique o número de vCPUs a serem reservadas para o contêiner. Esse parâmetro é mapeado para `CpuShares` na seção Criar um Contêiner da API remota do Docker e a opção `--cpu-shares` para docker run. Cada vCPU equivale a 1.024 compartilhamentos de CPU. Você deve especificar pelo menos uma vCPU.

c. Em Memória, insira o limite de memória que está disponível para o contêiner. Caso seu contêiner tente exceder a memória especificada, o mesmo será interrompido. Esse parâmetro é mapeado para `Memory` na seção Criar um Contêiner da API Remota Docker e para a opção `--memory` para docker run. Você deve especificar pelo menos 4 MiB de memória para uma tarefa.

> **Note**
>
> Para maximizar o uso dos recursos, priorize a memória para trabalhos de um tipo específico de instância. Para ter mais informações, consulte Recurso de Computação Gerenciamento de Memória.

d. (Opcional) Em Número de GPUs, escolha o número de GPUs a serem reservadas para o contêiner.

e. (Opcional) Em Variáveis de Ambiente, escolha Adicionar Variável de Ambiente para adicionar variáveis de ambiente como pares de nome/valor. Essas variáveis serão passadas para o contêiner.

f. Escolha Próxima página.

g. Em Revisão de Trabalho, revise as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Criar Definição de Trabalho.

Estados da tarefa

Quando você envia um trabalho para uma fila de AWS Batch trabalhos, o trabalho entra no `SUBMITTED` estado. Em seguida, ele passa pelos seguintes estados, até ter êxito (sai com código `0`) ou falhar (saindo com um código diferente de zero). Os trabalhos do AWS Batch podem ter os seguintes estados:

`SUBMITTED`

Um trabalho que foi enviado para a fila e ainda não foi avaliado pelo programador. O programador avalia o trabalho para determinar se ele tem dependências pendentes na conclusão bem-sucedida de todos os outros trabalhos. Se houver dependências, o trabalho será movido para `PENDING`. Se não houver dependências, o trabalho será movido para `RUNNABLE`.

`PENDING`

Um trabalho que reside na fila e ainda não pode ser executado devido a dependência de outro trabalho ou recurso. Após as dependências serem atendidas, o trabalho é movido para `RUNNABLE`.

`RUNNABLE`

Um trabalho que reside na fila, não tem dependências pendentes e, portanto, está pronto para ser programado para um host. Os trabalhos nesse estado são iniciados assim que recursos suficientes estão disponíveis em um dos ambientes computacionais mapeados para a fila do trabalho. No entanto, os trabalhos podem permanecer nesse estado indefinidamente quando recursos suficientes estiverem indisponíveis.

> **Note**
>
> Se os trabalhos não progredirem para `STARTING`, consulte Trabalhos presos no status `RUNNABLE` na seção de solução de problemas.

`STARTING`

Esses trabalhos foram programados para um host e as operações de inicialização de contêiner relevantes estão em andamento. Após a imagem de contêiner ser obtida e o contêiner estar em execução, o trabalho ocorre a transição do trabalho para `RUNNING`.

`RUNNING`

O trabalho é executado como um contêiner, em uma instância de contêiner do Amazon ECS em um ambiente de computação. Quando o contêiner do trabalho é encerrado, o código de saída do processo determina se o trabalho foi bem-sucedido ou não. Um código de saída `0` indica êxito, e qualquer código de saída diferente de zero, falha. Se o trabalho associado a uma falha na tentativa tiver tentativas restantes em sua configuração de estratégia de repetição

opcional, o trabalho será movido para `RUNNABLE` novamente. Para ter mais informações, consulte Repetições de trabalho automatizadas.

> **Note**
>
> Os registros de `RUNNING` trabalhos estão disponíveis em CloudWatch Registros. O grupo de logs é `/aws/batch/job`, e o formato do nome do fluxo de logs, o seguinte: *`first200CharsOfJobDefinitionName`*`/default/`*`ecs_task_id`*. Esse formato pode mudar no futuro.
> Depois que um trabalho atinge o `RUNNING` status, você pode recuperar programaticamente o nome do fluxo de registros com a operação da DescribeJobsAPI. Para obter mais informações, consulte Exibir dados de registro enviados para CloudWatch registros no Guia do usuário do Amazon CloudWatch Logs. Por padrão, esses logs nunca expiram. No entanto, é possível modificar o período de retenção. Para obter mais informações, consulte Alterar a retenção de dados do log em CloudWatch registros no Guia do usuário do Amazon CloudWatch Logs.

`SUCCEEDED`

O trabalho foi concluído com êxito com um código de saída `0`. O estado do trabalho para `SUCCEEDED` trabalhos persiste AWS Batch por pelo menos 7 dias.

> **Note**
>
> Os registros de `SUCCEEDED` trabalhos estão disponíveis em CloudWatch Registros. O grupo de logs é `/aws/batch/job`, e o formato do nome do fluxo de logs, o seguinte: *`first200CharsOfJobDefinitionName`*`/default/`*`ecs_task_id`*. Esse formato pode mudar no futuro.
> Depois que um trabalho atinge o `RUNNING` status, você pode recuperar programaticamente o nome do fluxo de registros com a operação da DescribeJobsAPI. Para obter mais informações, consulte Exibir dados de registro enviados para CloudWatch registros no Guia do usuário do Amazon CloudWatch Logs. Por padrão, esses logs nunca expiram. No entanto, é possível modificar o período de retenção. Para obter mais informações, consulte Alterar a retenção de dados do log em CloudWatch registros no Guia do usuário do Amazon CloudWatch Logs.

`FAILED`

Ocorreu falha na tarefa em todas as tentativas. O estado da tarefa para tarefas `FAILED` persiste no AWS Batch por pelo menos 7 dias.

> **Note**
>
> Os registros de `FAILED` trabalhos estão disponíveis em CloudWatch Registros. O grupo de logs é `/aws/batch/job`, e o formato do nome do fluxo de logs, o seguinte: *`first200CharsOfJobDefinitionName`*`/default/`*`ecs_task_id`*. Esse formato pode mudar no futuro.
> Depois que um trabalho atinge o `RUNNING` status, você pode recuperar programaticamente seu fluxo de registros com a operação da DescribeJobsAPI. Para obter mais informações, consulte Exibir dados de registro enviados para CloudWatch registros no Guia do usuário do Amazon CloudWatch Logs. Por padrão, esses logs nunca expiram. No entanto, é possível modificar o período de retenção. Para obter mais informações, consulte Alterar a retenção de dados do log em CloudWatch registros no Guia do usuário do Amazon CloudWatch Logs.

AWS Batch variáveis do ambiente de trabalho

AWS Batch define variáveis de ambiente específicas em trabalhos de contêiner. Essas variáveis de ambiente fornecem introspecção para os contêineres dentro dos trabalhos. Você pode utilizar os valores dessas variáveis na lógica de seus aplicativos. Todas as variáveis AWS Batch definidas começam com o `AWS_BATCH_` prefixo. Esse é um prefixo de variável de ambiente protegido. Você não pode usar esse prefixo para suas próprias variáveis, em definições de trabalhos ou substituições.

As seguintes variáveis de ambiente estão disponíveis em contêineres de trabalho:

`AWS_BATCH_CE_NAME`

Essa variável é definida como o nome do ambiente de computação no qual o trabalho é colocado.

`AWS_BATCH_JOB_ARRAY_INDEX`

Essa variável só é definida em trabalhos de matriz filhos. O índice de trabalho de matriz começa com 0, e cada trabalho filho recebe um número de índice exclusivo. Por exemplo, um trabalho de matriz com 10 filhos tem valores de índice de 0 a 9. Você pode utilizar esse valor de índice para

controlar como os filhos do trabalho de matriz serão diferenciados. Para ter mais informações, consulte Tutorial: usando o índice de trabalho de matriz para controle de diferenciação de trabalhos.

`AWS_BATCH_JOB_ARRAY_SIZE`

Essa variável é definida para o tamanho do trabalho da matriz pai. O tamanho do trabalho da matriz pai é passado para o trabalho de matriz filho nessa variável.

`AWS_BATCH_JOB_ATTEMPT`

Essa variável é definida como o número de tentativas de trabalho. A primeira tentativa recebe o número 1. Para ter mais informações, consulte Repetições de trabalho automatizadas.

`AWS_BATCH_JOB_ID`

Essa variável é definida como a ID do AWS Batch trabalho.

`AWS_BATCH_JOB_KUBERNETES_NODE_UID`

Essa variável é definida como o Kubernetes UID do objeto de nó que está no cluster Kubernetes em que o pod é executado. Essa variável é definida somente para trabalhos executados nos recursos do Amazon EKS. Para obter mais informações, consulte UIDs na Kubernetesdocumentação do.

`AWS_BATCH_JOB_MAIN_NODE_INDEX`

Essa variável é definida apenas em tarefas em paralelo de vários nós. Essa variável é definida como o número índice do nó principal do trabalho. O código do seu aplicativo pode comparar o `AWS_BATCH_JOB_MAIN_NODE_INDEX` ao `AWS_BATCH_JOB_NODE_INDEX` em um nó individual para determinar se ele é o nó principal.

`AWS_BATCH_JOB_MAIN_NODE_PRIVATE_IPV4_ADDRESS`

Essa variável é definida apenas em nós filhos de trabalhos em paralelo de vários nós. Essa variável não está presente no nó principal, mas é definida como o endereço IPv4 privado do nó principal do trabalho. O código da aplicação do nó filho pode utilizar esse endereço para comunicar-se com o nó principal.

`AWS_BATCH_JOB_NODE_INDEX`

Essa variável é definida apenas em tarefas em paralelo de vários nós. Essa variável é definida como o número do índice de nós do nó. O índice do nó começa em 0, e cada nó filho recebe um número índice exclusivo. Por exemplo, uma tarefa em paralelo de vários nós com 10 filhos possui valores índice de 0-9.

`AWS_BATCH_JOB_NUM_NODES`

Essa variável é definida apenas em tarefas em paralelo de vários nós. Essa variável é definida como o número de nós solicitado para sua tarefa paralela de vários nós.

`AWS_BATCH_JQ_NAME`

Essa variável é definida como o nome da fila de trabalhos para a qual o trabalho foi enviado.

Repetições de trabalho automatizadas

Você pode aplicar uma estratégia de repetição aos seus trabalhos e a definições de trabalho que permitam que trabalhos com falha sofram novas tentativas automaticamente. Os possíveis cenários de falha incluem os seguintes:

- Qualquer código de saída diferente de zero de um trabalho de contêiner
- Falha de instância ou encerramento do Amazon EC2
- Erro ou interrupção do AWS serviço interno

Quando um trabalho é enviado a uma fila de trabalhos e colocado no estado `RUNNING`, isso é considerado uma tentativa. Por padrão, cada trabalho tem uma tentativa de movimentação para o estado de trabalho `SUCCEEDED` ou `FAILED`. No entanto, tanto a definição de trabalho quanto o fluxo de trabalho do envio de trabalho podem ser utilizados para especificar uma estratégia de repetição entre 1 e 10 tentativas. Se evaluateOnExitfor especificado, ele pode conter até 5 estratégias de repetição. Se evaluateOnExitfor especificado, mas nenhuma das estratégias de repetição corresponder, o trabalho será repetido. Para trabalhos que não correspondam à saída, adicione uma entrada final que saia por qualquer motivo. Por exemplo, esse objeto `evaluateOnExit` tem duas entradas com ações de `RETRY` e uma entrada final com uma ação de `EXIT`.

```
"evaluateOnExit": [
    {
        "action": "RETRY",
        "onReason": "AGENT"
    },
    {
        "action": "RETRY",
        "onStatusReason": "Task failed to start"
    },
    {
```

```
        "action": "EXIT",
        "onReason": "*"
    }
]
```

No tempo de execução, a variável de ambiente `AWS_BATCH_JOB_ATTEMPT` é definida como o número de tentativas de trabalho correspondentes do contêiner. A primeira tentativa é numerada 1, e as tentativas subsequentes, em ordem ascendente (por exemplo, 2, 3, 4).

Por exemplo, suponha que uma tentativa de trabalho falhe por algum motivo e o número de tentativas especificado no campo de configuração de nova tentativa seja maior do que o número `AWS_BATCH_JOB_ATTEMPT`. Em seguida, o trabalho será devolvido ao estado `RUNNABLE`. Para ter mais informações, consulte Estados da tarefa.

> **Note**
>
> Os trabalhos cancelados ou encerrados não são repetidos. Além disso, os trabalhos que falham devido a uma definição de trabalho inválida não são repetidos.

Para obter mais informações, consulte Estratégia de repetição, Como criar uma definição de tarefa de nó único , Enviando um trabalho e Códigos de Erro de Tarefas Interrompidas.

Dependências do trabalho

Ao enviar um AWS Batch trabalho, você pode especificar os IDs dos trabalhos dos quais o trabalho depende. Ao fazer isso, o programador do AWS Batch garante que seu trabalho será executado somente depois que as dependências especificadas forem concluídas com êxito. Depois de concluído com êxito, o trabalho dependente transiciona de `PENDING` para `RUNNABLE` e, em seguida, para `STARTING` e `RUNNING`. Se uma das dependências do trabalho falhar, o trabalho dependente transicionará automaticamente de `PENDING` para `FAILED`.

Por exemplo, o trabalho A pode expressar uma dependência de até 20 outros trabalhos, que devem obter êxito antes que ele possa ser executado. Você pode, então, enviar trabalhos adicionais que dependam do trabalho A e de até 19 outros trabalhos.

Para trabalhos de matriz, você pode especificar uma dependência do tipo `SEQUENTIAL` sem especificar uma ID do trabalho, de forma que cada trabalho de matriz filho seja concluído sequencialmente, a começar pelo índice 0. Você também pode especificar uma dependência do tipo

N_TO_N com uma ID do trabalho. Deste modo, cada índice filho dessa tarefa precisa aguardar, para que o índice filho correspondente de cada dependência seja concluído antes de começar. Para ter mais informações, consulte Trabalhos de matriz.

Para enviar um AWS Batch trabalho com dependências, consulteEnviando um trabalho.

Tempos limite do trabalho

Você pode configurar uma duração de tempo limite para seus trabalhos para que, se um trabalho for executado por mais tempo do que isso, AWS Batch encerre o trabalho. Por exemplo, caso você tenha um trabalho que sabe que deve demorar apenas 15 minutos para ser concluído. Às vezes, o aplicativo fica preso em um loop e executa para sempre. Nesse caso, você pode definir um tempo limite de 30 minutos para encerrar a tarefa presa.

> **Important**
>
> Por padrão, AWS Batch não tem um tempo limite de trabalho. Se você não definir um tempo limite do trabalho, o mesmo será executado até a saída do contêiner.

Especifique um parâmetro `attemptDurationSeconds`, que deve ser pelo menos 60 segundos, seja na definição de trabalho ou no envio do trabalho. Quando esse número de segundos tiver passado após o registro de `startedAt` data e hora da tentativa de trabalho, o trabalho AWS Batch será encerrado. No recurso de computação, o contêiner de tarefa recebe um sinal `SIGTERM` para que seu aplicativo possa encerrar normalmente. Se o contêiner ainda estiver em execução após 30 segundos, um sinal `SIGKILL` será enviado para encerrar o contêiner.

Os encerramentos por tempo limite são processados com base no melhor esforço. Você não deve esperar pelo encerramento por tempo limite no momento exato em que a tentativa de trabalho expirar (pode demorar alguns segundos adicionais). Se o aplicativo obrigar a execução do tempo limite exato, você deve implementar essa lógica no aplicativo. Caso tenha uma grande quantidade de tarefas atingindo o tempo limite concomitantemente, os encerramentos por tempo limite se comportarão como uma fila por ordem de chegada, na qual os trabalhos são encerrados em lotes.

> **Note**
>
> Não há valor máximo de tempo limite para um AWS Batch trabalho.

Se um trabalho for encerrado devido a uma duração de tempo limite excedido, ele não será executado novamente. Se ocorrer uma falha na tentativa de trabalho por si só, ela poderá ser iniciada novamente, caso repetições estejam habilitadas. Além disso, a contagem do tempo limite é reiniciada para a nova tentativa.

> **Important**
>
> Não se pode esperar que trabalhos executados com recursos Fargate sejam executados por um período maior que 14 dias. Se a duração do tempo limite exceder 14 dias, os recursos Fargate podem não estar mais disponíveis, e o trabalho será encerrado.

Em trabalhos de matriz, os trabalhos filho têm a mesma configuração de tempo limite do trabalho pai.

Para obter informações sobre como enviar um AWS Batch trabalho com uma configuração de tempo limite, consulte. Enviando um trabalho

Trabalhos do Amazon EKS

Um emprego é a menor unidade de trabalho em AWS Batch. Um AWS Batch trabalho no Amazon EKS tem um one-to-one mapeamento para um Kubernetes pod. Uma definição de AWS Batch trabalho é um modelo para um AWS Batch trabalho. Ao enviar um AWS Batch trabalho, você faz referência a uma definição de trabalho, direciona uma fila de trabalhos e fornece um nome para um trabalho. Na definição de trabalho de um AWS Batch trabalho no Amazon EKS, o parâmetro EKSProperties define o conjunto de parâmetros que um trabalho no AWS Batch Amazon EKS suporta. Em uma SubmitJobsolicitação, o eksPropertiesOverrideparâmetro permite a substituição de alguns parâmetros comuns. Dessa forma, você pode usar modelos de definições de trabalhos para vários trabalhos. Quando um trabalho é enviado para seu cluster Amazon EKS, AWS Batch transforma o trabalho em um `podspec` (). `Kind: Pod` O `podspec` usa alguns AWS Batch parâmetros adicionais para garantir que os trabalhos sejam escalados e programados corretamente. AWS Batch combina rótulos e contaminações para garantir que os trabalhos sejam executados somente em nós AWS Batch gerenciados e que outros pods não sejam executados nesses nós.

> **Important**
>
> - Se o `hostNetwork` parâmetro não for definido explicitamente em uma definição de trabalho do Amazon EKS, o modo de rede do pod assumirá como AWS Batch padrão

o modo host. Mais especificamente, as seguintes configurações serão aplicadas: `hostNetwork=true` e `dnsPolicy=ClusterFirstWithHostNet`.

- AWS Batch limpa os pods de trabalho logo após um pod concluir seu trabalho. Para ver os logs do aplicativo do pod, configure um serviço de log para seu cluster. Para ter mais informações, consulte Use o CloudWatch Logs para monitorar AWS Batch trabalhos do Amazon EKS.

Mapeie um trabalho em execução para um pod e um nó

As `podProperties` de um trabalho em execução têm parâmetros `podName` e `nodeName` definidos para a tentativa de trabalho atual. Use a operação DescribeJobsda API para visualizar esses parâmetros.

A seguir, um exemplo de saída.

```
$ aws batch describe-jobs --job 2d044787-c663-4ce6-a6fe-f2baf7e51b04
{
 "jobs": [
  {
   "status": "RUNNING",
   "jobArn": "arn:aws:batch:us-east-1:123456789012:job/2d044787-c663-4ce6-a6fe-
f2baf7e51b04",
   "jobDefinition": "arn:aws:batch:us-east-1:123456789012:job-definition/
MyJobOnEks_SleepWithRequestsOnly:1",
   "jobQueue": "arn:aws:batch:us-east-1:123456789012:job-queue/My-Eks-JQ1",
   "jobId": "2d044787-c663-4ce6-a6fe-f2baf7e51b04",
   "eksProperties": {
    "podProperties": {
     "nodeName": "ip-192-168-55-175.ec2.internal",
     "containers": [
      {
       "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
       "resources": {
        "requests": {
         "cpu": "1",
         "memory": "1024Mi"
        }
       }
      }
     ],
```

```
      "podName": "aws-batch.b0aca953-ba8f-3791-83e2-ed13af39428c"
    }
   }
  }
 ]
}
```

Para um trabalho com novas tentativas ativadas, o `podName` final `nodeName` de cada tentativa concluída está no parâmetro de `eksAttempts` lista da operação da DescribeJobsAPI. O `podName` e `nodeName` da tentativa de execução atual estarão no objeto `podProperties`.

Como mapear um pod em execução de volta ao seu trabalho

Um pod tem rótulos que indicam `jobId` a extremidade `uuid` do ambiente computacional ao qual ele pertence. AWS Batch injeta variáveis de ambiente para que o tempo de execução do trabalho possa referenciar as informações do trabalho. Para ter mais informações, consulte AWS Batch variáveis do ambiente de trabalho. Você pode visualizar isso executando o seguinte comando. A saída é a seguinte:

```
$ kubectl describe pod aws-batch.14638eb9-d218-372d-ba5c-1c9ab9c7f2a1 -n my-aws-batch-
namespace
Name:         aws-batch.14638eb9-d218-372d-ba5c-1c9ab9c7f2a1
Namespace:    my-aws-batch-namespace
Priority:     0
Node:         ip-192-168-45-88.ec2.internal/192.168.45.88
Start Time:   Wed, 26 Oct 2022 00:30:48 +0000
Labels:       batch.amazonaws.com/compute-environment-uuid=5c19160b-
d450-31c9-8454-86cf5b30548f
              batch.amazonaws.com/job-id=f980f2cf-6309-4c77-a2b2-d83fbba0e9f0
              batch.amazonaws.com/node-uid=a4be5c1d-9881-4524-b967-587789094647
...
Status:       Running
IP:           192.168.45.88
IPs:
  IP:  192.168.45.88
Containers:
  default:
    Image:         public.ecr.aws/amazonlinux/amazonlinux:2
    ...
    Environment:
      AWS_BATCH_JOB_KUBERNETES_NODE_UID:  a4be5c1d-9881-4524-b967-587789094647
      AWS_BATCH_JOB_ID:                   f980f2cf-6309-4c77-a2b2-d83fbba0e9f0
```

```
      AWS_BATCH_JQ_NAME:                  My-Eks-JQ1
      AWS_BATCH_JOB_ATTEMPT:              1
      AWS_BATCH_CE_NAME:                  My-Eks-CE1

...
```

Recursos que o AWS Batch Amazon EKS Jobs oferece suporte

Esses são os recursos AWS Batch específicos que também são comuns aos Kubernetes trabalhos executados no Amazon EKS:

- Dependências do trabalho
- Trabalhos de matriz
- Tempos limite do trabalho
- Repetições de trabalho automatizadas
- Programação de compartilhamento justo

Kubernetes**`Secrets`** e **`ServiceAccounts`**

AWS Batch suporta referências Kubernetes `Secrets` e. `ServiceAccounts` Você pode configurar pods para usar funções do IAM para contas de serviço do Amazon EKS. Para obter mais informações, consulte Como Configurar Pods para Utilizar uma Conta de Serviço Kubernetes no Manual do Usuário Amazon EKS.

Documentação relacionada

- Considerações sobre memória e vCPU para AWS Batch no Amazon EKS
- Para criar um trabalho baseado em GPU nos recursos do Amazon EKS
- Trabalhos presos no status `RUNNABLE`

Trabalhos de matriz

Um trabalho de matriz é um trabalho que compartilha parâmetros comuns, como a definição do trabalho, vCPUs e a memória. Ele é executado como coleções de tópicos de trabalhos básicos porém separados, que podem ser distribuídos entre vários hosts e executados simultaneamente. Os trabalhos de matriz são a maneira mais eficiente de executar trabalhos extremamente em paralelo, como simulações de Monte Carlo, varreduras paramétricas ou trabalhos grandes de renderização.

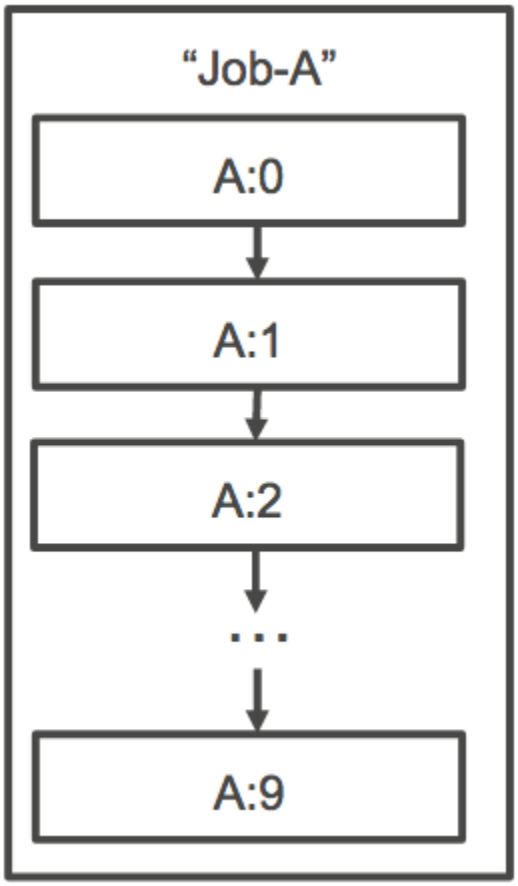
AWS Batch os trabalhos de matriz são enviados da mesma forma que os trabalhos normais. No entanto, você especifica o tamanho da matriz (entre 2 e 10.000) para definir quantos trabalhos filho devem ser executados no campo da matriz. Se você enviar um trabalho com tamanho da matriz 1.000, haverá uma única execução de trabalho gerando 1.000 trabalhos filho. O trabalho de matriz é uma referência ou ponteiro para gerenciar todos os trabalhos filho. Você pode enviar grandes workloads com uma única consulta. O tempo limite especificado no campo do parâmetro `attemptDurationSeconds` é aplicado em cada trabalho filho. O trabalho pai da matriz não tem um tempo limite.

Quando você envia um trabalho de matriz, o trabalho de matriz principal recebe uma ID de AWS Batch trabalho normal. Cada trabalho filho tem a mesma ID base. No entanto, o índice da matriz do trabalho filho é anexado até o fim da ID pai, como *`example_job_ID`*`:0` para o primeiro trabalho filho da matriz.

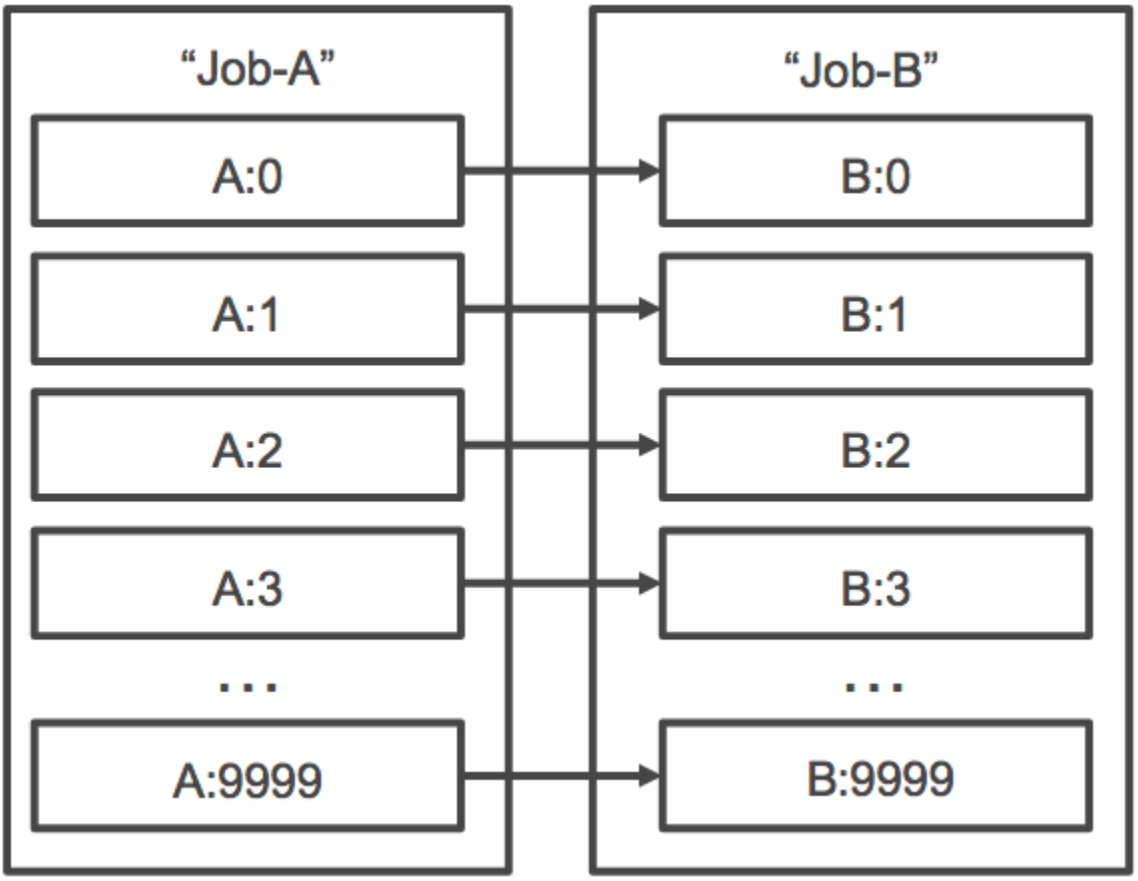
O trabalho de matriz pai pode inserir um status `SUBMITTED`, `PENDING`, `FAILED` ou `SUCCEEDED`. Um trabalho de matriz pai é atualizado para `PENDING` quando qualquer trabalho filho é atualizado para `RUNNABLE`. Para obter mais informações sobre dependências do trabalho, consulte Dependências do trabalho.

Durante o tempo de execução, a variável de ambiente `AWS_BATCH_JOB_ARRAY_INDEX` é definida como o número do índice da matriz do trabalho correspondente do contêiner. O primeiro índice do trabalho de matriz é numerado com `0` e as tentativas subsequentes são numeradas em ordem crescente (por exemplo 1, 2 e 3). Você pode utilizar esse valor de índice para controlar como os filhos do trabalho de matriz serão diferenciados. Para ter mais informações, consulte Tutorial: usando o índice de trabalho de matriz para controle de diferenciação de trabalhos.

Para dependências do trabalhos de matriz, você pode especificar um tipo para cada dependência, como `SEQUENTIAL` ou `N_TO_N`. Você pode especificar uma dependência do tipo `SEQUENTIAL` (sem especificar ID de trabalho), de forma que cada trabalho de matriz filho seja concluído sequencialmente, começando com o índice 0. Por exemplo, se você enviar um trabalho de matriz com o tamanho da matriz 100 e especificar uma dependência do tipo `SEQUENTIAL`, 100 trabalhos filho serão gerados sequencialmente. O primeiro trabalho filho precisará ter êxito antes que o próximo trabalho filho comece. A figura abaixo mostra o trabalho A, um trabalho de matriz com um tamanho da matriz somando 10. Cada trabalho no índice filho do trabalho A depende do trabalho filho anterior. O trabalho A:1 não pode iniciar até que o trabalho A:0 termine.



Você também pode especificar uma dependência do tipo `N_TO_N` com ID do trabalho para trabalhos de matriz. Deste modo, cada índice filho dessa tarefa precisa aguardar, para que o índice filho correspondente de cada dependência seja concluído antes de começar. A figura a seguir mostra o trabalho A e o trabalho B, dois trabalhos de matriz de tamanho 10.000 cada. Cada trabalho no índice filho do trabalho B depende do índice correspondente no trabalho A. O trabalho B:1 não pode iniciar até que o trabalho A:1 termine.



Se você cancelar ou encerrar um trabalho de matriz pai, todos os trabalhos filho serão cancelados ou encerrados com ele. Você pode cancelar ou encerrar trabalhos filho individuais (movendo-os para o status `FAILED`) sem afetar os outros trabalhos filho. No entanto, se um trabalho de matriz filho falhar (sozinho, por ter sido cancelado ou por encerramento manual), o trabalho pai também falhará.

Exemplo do fluxo de trabalho de trabalhos de matriz

Um fluxo de trabalho comum para AWS Batch os clientes é executar um trabalho de configuração pré-requisito, executar uma série de comandos em um grande número de tarefas de entrada e, em seguida, concluir com um trabalho que agrega resultados e grava dados resumidos no Amazon S3, DynamoDB, Amazon Redshift ou Aurora.

Por exemplo: .

- `JobA`: um trabalho de não matriz padrão que executa uma rápida listagem e validação de metadados de objetos em um bucket do Amazon S3, `BucketA`. A sintaxe SubmitJobJSON é a seguinte.

```
{
    "jobName": "JobA",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobA-list-and-validate:1"
}
```

- `JobB`: um trabalho de matriz com 10 mil cópias que depende do `JobA`, que, por sua vez, executa comandos com uso intensivo de CPU para cada objeto em `BucketA` e carrega resultados para `BucketB`. A sintaxe SubmitJobJSON é a seguinte.

```
{
    "jobName": "JobB",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobB-CPU-Intensive-Processing:1",
    "containerOverrides": {
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "4096"
            },
            {
                "type": "VCPU",
                "value": "32"
            }
        ]
   }
    "arrayProperties": {
        "size": 10000
    },
    "dependsOn": [
        {
            "jobId": "JobA_job_ID"
  }
    ]
}
```

- `JobC`: outro trabalho de matriz com 10 mil cópias, que depende do `JobB` com um modelo de dependência de `N_TO_N`, que executa comandos com uso intensivo de memória em relação a cada item em`BucketB`, grava metadados no DynamoDB e carrega saída resultante em `BucketC`. A sintaxe SubmitJobJSON é a seguinte.

```
{
    "jobName": "JobC",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobC-Memory-Intensive-Processing:1",
    "containerOverrides": {
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "32768"
            },
            {
                "type": "VCPU",
                "value": "1"
            }
        ]
   }
    "arrayProperties": {
        "size": 10000
    },
    "dependsOn": [
        {
            "jobId": "JobB_job_ID",
            "type": "N_TO_N"
        }
    ]
}
```

- JobD: um trabalho de matriz que executa 10 etapas de validação, no qual cada uma precisa consultar o DynamoDB e pode interagir com qualquer um dos buckets do Amazon S3 mencionados acima. Cada uma das etapas no JobD executa o mesmo comando. No entanto, o comportamento é diferente baseado no valor da variável de ambiente AWS_BATCH_JOB_ARRAY_INDEX dentro do contêiner do trabalho. Essas etapas de validação são executadas sequencialmente (por exemplo, JobD:0 e, em seguida, JobD:1). A sintaxe SubmitJobJSON é a seguinte.

```
{
    "jobName": "JobD",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobD-Sequential-Validation:1",
    "containerOverrides": {
        "resourceRequirements": [
```

```
            {
                "type": "MEMORY",
                "value": "32768"
            },
            {
                "type": "VCPU",
                "value": "1"
            }
        ]
   }
    "arrayProperties": {
        "size": 10
    },
    "dependsOn": [
        {
            "jobId": "JobC_job_ID"
        },
        {
            "type": "SEQUENTIAL"
        },

    ]
}
```

- JobE: um trabalho de não matriz final, que executa algumas operações de limpeza simples e envia uma notificação SNS da Amazon, com uma mensagem avisando que o pipeline foi concluído e vinculado a URL de saída. A sintaxe SubmitJobJSON é a seguinte.

```
{
    "jobName": "JobE",
    "jobQueue": "ProdQueue",
    "jobDefinition": "JobE-Cleanup-and-Notification:1",
    "parameters": {
        "SourceBucket": "s3://JobD-Output-Bucket",
        "Recipient": "pipeline-notifications@mycompany.com"
    },
    "dependsOn": [
        {
            "jobId": "JobD_job_ID"
        }
    ]
}
```

Tutorial: usando o índice de trabalho de matriz para controle de diferenciação de trabalhos

Este tutorial descreve instruções de uso da variável de ambiente `AWS_BATCH_JOB_ARRAY_INDEX` para diferenciar os trabalhos filho. Cada trabalho filho é atribuído a essa variável. O exemplo utiliza o número de índice do trabalho filho para ler uma linha específica em um arquivo. Em seguida, ele substitui o parâmetro associado a esse número de linha por um comando dentro do contêiner do trabalho. O resultado é que você pode ter vários AWS Batch trabalhos que executam a mesma imagem do Docker e argumentos de comando. No entanto, os resultados serão diferentes porque o índice de trabalho da matriz é usado como um modificador.

Neste tutorial, um arquivo de texto é criado com todas as cores do arco-íris, cada uma em sua própria linha. Em seguida, você cria um script de ponto de entrada para um contêiner do Docker que converte o índice em um valor, que pode ser usado para um número de linha no campo do arquivo de cores. O índice inicia em zero, mas os números da linha iniciam em um. Crie um Dockerfile que copie os arquivos de cores e índice na imagem de contêiner, e defina o `ENTRYPOINT` da imagem como o script de ponto de entrada. O Dockerfile e os recursos são incorporados em uma imagem do Docker, que é enviada para o Amazon ECR. Em seguida, você registra uma definição de tarefa que usa sua nova imagem de contêiner, envia uma tarefa de AWS Batch matriz com essa definição de tarefa e visualiza os resultados.

Pré-requisitos

Este tutorial tem os seguintes pré-requisitos:

- Um ambiente AWS Batch computacional. Para ter mais informações, consulte Criação de um ambiente de computação.
- Uma fila AWS Batch de trabalhos e um ambiente computacional associado. Para ter mais informações, consulte Como criar uma fila de tarefas.
- O AWS CLI instalado em seu sistema local. Para obter mais informações, consulte Instalando AWS Command Line Interface no AWS Command Line Interface Manual do Usuário.
- Docker instalado em seu sistema local. Para obter mais informações, consulte Sobre o Docker CE no campo de documentação do Docker.

Etapa 1: criar uma imagem de contêiner

Você pode usar o `AWS_BATCH_JOB_ARRAY_INDEX` em uma definição de trabalho no parâmetro de comando. No entanto, recomendamos criar uma imagem de contêiner que use a variável em um script de ponto de entrada. Esta seção descreve instruções para criação essa imagem de contêiner.

Para criar sua imagem de contêiner do Docker

1. Crie novo diretório, a ser usado como seu workspace de imagem do Docker, e navegue até ele.
2. Crie um arquivo de nome `colors.txt` em seu diretório do workspace e cole o conteúdo a seguir.

```
red
orange
yellow
green
blue
indigo
violet
```

3. Crie um arquivo de nome `print-color.sh` em seu diretório do workspace e cole o conteúdo a seguir.

> **Note**
>
> A variável `LINE` é definida como `AWS_BATCH_JOB_ARRAY_INDEX` +1, pois o índice de matriz inicia em 0, mas os números de linha iniciam em com 1. A variável `COLOR` é definida como a cor no `colors.txt` associado ao número de linha.

```
#!/bin/sh
LINE=$((AWS_BATCH_JOB_ARRAY_INDEX + 1))
COLOR=$(sed -n ${LINE}p /tmp/colors.txt)
echo My favorite color of the rainbow is $COLOR.
```

4. Crie um arquivo de nome `Dockerfile` em seu diretório do workspace e cole nele o conteúdo a seguir. Esse Dockerfile copia os arquivos anteriores em seu contêiner e define o script de ponto de entrada a ser executado quando o contêiner for iniciado.

```
FROM busybox
```

```
COPY print-color.sh /tmp/print-color.sh
COPY colors.txt /tmp/colors.txt
RUN chmod +x /tmp/print-color.sh
ENTRYPOINT /tmp/print-color.sh
```

5. Compile sua imagem do Docker.

```
$ docker build -t print-color .
```

6. Teste seu contêiner com o script a seguir. Esse script define a variável `AWS_BATCH_JOB_ARRAY_INDEX` como 0 localmente e, em seguida, a acrescenta, para simular o que um trabalho de matriz com sete filhos faz.

```
$ AWS_BATCH_JOB_ARRAY_INDEX=0
while [ $AWS_BATCH_JOB_ARRAY_INDEX -le 6 ]
do
    docker run -e AWS_BATCH_JOB_ARRAY_INDEX=$AWS_BATCH_JOB_ARRAY_INDEX print-color
    AWS_BATCH_JOB_ARRAY_INDEX=$((AWS_BATCH_JOB_ARRAY_INDEX + 1))
done
```

A saída vem a seguir.

```
My favorite color of the rainbow is red.
My favorite color of the rainbow is orange.
My favorite color of the rainbow is yellow.
My favorite color of the rainbow is green.
My favorite color of the rainbow is blue.
My favorite color of the rainbow is indigo.
My favorite color of the rainbow is violet.
```

Etapa 2: envie sua imagem para o Amazon ECR

Agora que você criou e testou o contêiner do Docker, envie-o para um repositório de imagens. Este exemplo usa o Amazon ECR, mas você pode usar outro registro, como DockerHub.

1. Crie um repositório de imagens do Amazon ECR para armazenar sua imagem de contêiner. Este exemplo usa somente AWS CLI o. mas você também pode usar AWS Management Console o. Para obter mais informações, consulte Criando um Repositório no campo do Manual do Usuário do Amazon Elastic Container Registry.

```
$ aws ecr create-repository --repository-name print-color
```

2. Marque sua imagem `print-color` com o URI do repositório do Amazon ECR retornado na etapa anterior.

```
$ docker tag print-color aws_account_id.dkr.ecr.region.amazonaws.com/print-color
```

3. Inicie a sessão no seu registro do Amazon ECR. Para obter mais informações, consulte Autenticação de Registro no campo do Manual do Usuário do Amazon Elastic Container Registry.

```
$ aws ecr get-login-password \
    --region region | docker login \
    --username AWS \
    --password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

4. Envie sua imagem para o Amazon ECR.

```
$ docker push aws_account_id.dkr.ecr.region.amazonaws.com/print-color
```

Etapa 3: criar e registrar uma definição de trabalho

Agora que sua imagem do Docker está em um registro de imagem, você pode especificá-la em uma definição de AWS Batch tarefa. Assim, você pode usá-la posteriormente para executar um trabalho de matriz. Este exemplo usa apenas o AWS CLI. No entanto, você também pode usar o AWS Management Console. Para ter mais informações, consulte Como criar uma definição de tarefa de nó único .

Para criar uma definição de trabalho

1. Crie um arquivo de nome `print-color-job-def.json` em seu diretório do workspace e cole o conteúdo a seguir. Substitua o URI do repositório de imagem pelo seu próprio URI da imagem.

```
{
  "jobDefinitionName": "print-color",
  "type": "container",
  "containerProperties": {
    "image": "aws_account_id.dkr.ecr.region.amazonaws.com/print-color",
    "resourceRequirements": [
```

```
        {
            "type": "MEMORY",
            "value": "250"
        },
        {
            "type": "VCPU",
            "value": "1"
        }
    ]
  }
}
```

2. Registre a definição do trabalho com AWS Batch.

```
$ aws batch register-job-definition --cli-input-json file://print-color-job-def.json
```

Etapa 4: enviar um trabalho AWS Batch de matriz

Depois de registrar sua definição de tarefa, você pode enviar uma tarefa de AWS Batch matriz que usa sua nova imagem de contêiner.

Para enviar um trabalho AWS Batch de matriz

1. Crie um arquivo de nome `print-color-job.json` em seu diretório do workspace e cole o conteúdo a seguir.

> **Note**
>
> Este exemplo usa a fila de trabalhos mencionada na seção the section called “Pré-requisitos”.

```
{
  "jobName": "print-color",
  "jobQueue": "existing-job-queue",
  "arrayProperties": {
    "size": 7
  },
  "jobDefinition": "print-color"
```

```
}
```

2. Envie o trabalho para sua fila de AWS Batch trabalhos. Anote a ID do trabalho retornado na saída.

```
$ aws batch submit-job --cli-input-json file://print-color-job.json
```

3. Descreva o status do trabalho e espere até que o trabalho seja movido para `SUCCEEDED`.

Etapa 5: visualizar seus logs de trabalhos de matriz

Depois que seu trabalho atingir o `SUCCEEDED` status, você poderá visualizar os CloudWatch registros no contêiner do trabalho.

Para ver os registros do seu trabalho em CloudWatch Registros

1. Abra o AWS Batch console em https://console.aws.amazon.com/batch/.
2. No painel de navegação à esquerda, escolha Trabalhos.
3. Para Fila de Trabalhos, selecione uma fila.
4. Na seção Status, escolha bem-sucedido.
5. Para exibir todas os trabalhos filho de seu trabalho matriz, selecione a ID do trabalho que foi retornado na seção anterior.
6. Para consultar os logs do contêiner do trabalho, selecione um dos trabalhos filho e escolha Visualizar Logs.

7. Visualize os outros logs de trabalho filho. Cada trabalho retorna uma cor diferente do arco-íris.

Trabalhos paralelos de vários nós

Você pode usar trabalhos paralelos de vários nós para executar trabalhos únicos, que englobem várias instâncias do Amazon EC2. Com trabalhos paralelo de vários nós AWS Batch , você pode executar aplicativos de computação de alta performance em grande escala e treinamento em modelo de GPU distribuído sem a necessidade de iniciar, configurar e gerenciar diretamente os recursos do Amazon EC2. Uma tarefa paralela de AWS Batch vários nós é compatível com qualquer estrutura que ofereça suporte à comunicação entre nós baseada em IP. Os exemplos incluem Apache MXNet TensorFlow, Caffe2 ou Message Passing Interface (MPI).

Trabalhos paralelos de vários nós de vários nós são enviados como uma única tarefa. No entanto, sua definição de trabalho (ou substituições de nó de envio de trabalho) especifica o número de nós a serem criados para o trabalho e quais grupos de nós criar. Cada trabalho paralelo de vários nós contém um nó principal que é executado primeiro. Depois que o nó principal estiver ativo, os nós filhos serão executados e iniciados. O trabalho será concluído somente se o nó principal sair. Nesse caso, todos nós secundários serão terminados. Para ter mais informações, consulte Grupos de nós.

Os nós de trabalho paralelos de vários nós são de locatário único. Isso significa que um único contêiner de trabalho é executado em cada instância do Amazon EC2.

O status final do trabalho (`SUCCEEDED` ou `FAILED`) é determinado pelo status final do trabalho do nó principal. Para obter o status de um trabalho paralelo de vários nós, descreva a tarefa usando a ID da tarefa retornada quando você enviou a tarefa. Se precisar dos detalhes dos nós filhos, você deverá descrever cada nó filho individualmente. Você pode endereçar os nós usando a notação *#N* (começando com 0). Por exemplo, para acessar os detalhes do segundo nó de um trabalho, descreva *aws_batch_job_id #1 usando* a operação da API. AWS Batch DescribeJobs As informações de `started`, `stoppedAt`, `statusReason` e `exit` para um trabalho paralelo de vários nós são preenchidas a partir do nó principal.

Se você especificar novas repetições de trabalho, uma falha no nó principal fará com que outra tentativa ocorra. Falhas no nó filho não causam a ocorrência de mais tentativas. Cada nova tentativa de um tarefa em paralelo de vários nós atualiza a tentativa correspondente de seus nós filhos associados.

Para executar trabalhos paralelos de vários nós AWS Batch, o código do aplicativo deve conter as estruturas e bibliotecas necessárias para a comunicação distribuída.

Variáveis de ambiente

Em tempo de execução, cada nó é configurado com as variáveis de ambiente padrão que todos os AWS Batch trabalhos recebem. Além disso, os nós são configurados com as seguintes variáveis de ambiente específicas de trabalhos paralelos de vários nós:

`AWS_BATCH_JOB_MAIN_NODE_INDEX`

Essa variável é definida como o número índice do nó principal do trabalho. O código do seu aplicativo pode comparar o `AWS_BATCH_JOB_MAIN_NODE_INDEX` ao `AWS_BATCH_JOB_NODE_INDEX` em um nó individual para determinar se ele é o nó principal.

`AWS_BATCH_JOB_MAIN_NODE_PRIVATE_IPV4_ADDRESS`

Essa variável é definida apenas em nós filhos de trabalhos em paralelo de vários nós. Essa variável não está presente no nó principal. Essa variável é definida como o endereço IPv4 privado do nó principal da tarefa. O código da aplicação do nó filho pode utilizar esse endereço para comunicar-se com o nó principal.

`AWS_BATCH_JOB_NODE_INDEX`

Essa variável é definida como o número do índice de nós do nó. O índice do nó começa em 0, e cada nó filho recebe um número índice exclusivo. Por exemplo, uma tarefa em paralelo de vários nós com 10 filhos possui valores índice de 0-9.

`AWS_BATCH_JOB_NUM_NODES`

Essa variável é definida como o número de nós solicitados por você para seu trabalho paralelo de vários nós.

Grupos de nós

Um grupo de nós é um grupo idêntico de nós de trabalhos que compartilham as mesmas propriedades do contêiner. Você pode usar AWS Batch para especificar até cinco grupos de nós distintos para cada trabalho.

Cada grupo pode ter suas próprias imagens de contêiner, comandos, variáveis de ambiente e assim por diante. Por exemplo, você pode enviar um trabalho que exija uma instância única `c5.xlarge` para o nó principal e cinco nós filhos da instância `c5.xlarge`. Cada um desses grupos de nós distintos pode especificar diferentes imagens de contêiner ou comandos a serem executados para cada trabalho.

Como alternativa, todos os nós em seu trabalho podem usar um único grupo de nós. Além disso, o código do aplicativo pode diferenciar as funções do nó, como o nó principal e o nó filho. Ele faz isso comparando a variável de ambiente `AWS_BATCH_JOB_MAIN_NODE_INDEX` com seu próprio valor para `AWS_BATCH_JOB_NODE_INDEX`. Você pode ter até 1.000 nós em um único trabalho. Esse é o limite padrão para instâncias em um cluster do Amazon ECS. Você pode solicitar que esse limite seja aumentado.

> **Note**
>
> Atualmente, todos os grupos de nós em um trabalho paralelo de vários nós devem usar o mesmo tipo de instância.

Ciclo de vida do trabalho

Quando você envia um trabalho paralelo de vários nós, o trabalho entra no status `SUBMITTED`. Em seguida, o trabalho aguarda a conclusão de todas as dependências do trabalho. O trabalho também passa para o status `RUNNABLE`. Por fim, AWS Batch provisiona a capacidade da instância necessária para executar seu trabalho e executa essas instâncias.

Cada tarefa em paralelo de vários nós contém um nó principal. O nó principal é uma subtarefa única que AWS Batch monitora para determinar o resultado do trabalho de vários nós enviado. O nó principal é iniciado primeiro e passa para o status de `STARTING`. O valor do tempo limite especificado no campo de parâmetro `attemptDurationSeconds` se aplica a todo o trabalho, não aos nós.

Quando o nó principal atinge o status de `RUNNING` depois que o contêiner do nó estiver em execução, os nós filhos são iniciados e também são movidos para o status de `STARTING`. Os nós filhos são exibidos em uma ordem aleatória. Não há garantias sobre o tempo ou a ordem de execução do nó filho. Para garantir que todos os nós das tarefas estejam no status de `RUNNING` após a execução do contêiner do nó, o código do seu aplicativo pode consultar a API do AWS Batch para obter informações do nó principal e do nó filho. Como alternativa, o código da aplicação pode esperar até que todos os nós estejam online, antes de iniciar qualquer tarefa de processamento distribuído. O endereço IP privado do nó principal está disponível como a variável de ambiente `AWS_BATCH_JOB_MAIN_NODE_PRIVATE_IPV4_ADDRESS` em cada nó filho. Seu código de aplicação pode usar essas informações para coordenar e comunicar dados entre cada tarefa.

À medida que os nós individuais saem, eles são movidos para `SUCCEEDED` ou `FAILED`, a depender do código de saída. Se o nó principal for encerrado, a tarefa será considerada terminada e todos

os nós filhos serão parados. Se um nó filho morrer, AWS Batch não executará nenhuma ação nos outros nós do trabalho. Se você não quiser que seu trabalho continue com um número de nós reduzido, você deve incluir isso no código do aplicativo. Isso termina ou cancela o trabalho.

Considerações sobre o ambiente de computação

Há várias coisas a serem consideradas ao criar ambientes de computação para executar trabalhos paralelos de vários nós com o AWS Batch.

- Trabalhos paralelos de vários nós não são suportados por ambientes de computação `UNMANAGED`.
- Se você deseja enviar trabalhos paralelos de vários nós para um ambiente de computação, crie um grupo com posicionamento em cluster em uma Zona de Disponibilidade única e associe-a aos seus recursos de computação. Isso mantém seus trabalhos paralelos de vários nós em um agrupamento lógico de instâncias próximas com alto potencial de fluxo de rede. Para obter mais informações, consulte Grupo de Posicionamento no Manual do Usuário para instâncias do Linux do Amazon EC2.
- Trabalhos paralelos de vários nós não são suportados em ambientes de computação que usam instâncias spot.
- AWS Batch trabalhos paralelos de vários nós usam o modo de `awsvpc` rede do Amazon ECS, que fornece aos contêineres de trabalhos paralelos de vários nós as mesmas propriedades de rede das instâncias do Amazon EC2. Cada contêiner de trabalho paralelo de vários nós obtém sua própria interface de rede elástica, um endereço IP privado primário e um nome de host DNS interno. A interface de rede é criada na mesma sub-rede VPC que seu recurso de computação do host. Todos os grupos de segurança aplicados aos seus recursos de computação também são aplicados a ele. Para obter mais informações, consulte Redes de Tarefas com o Modo de Rede Awsvpc no Guia do Desenvolvedor do Amazon Elastic Container Service.
- Seu ambiente de computação não pode ter mais de cinco grupos de segurança associados a ele.
- O modo de rede `awsvpc` não fornece as interfaces de rede elásticas para trabalhos paralelos de vários nós com endereços IP públicos. Para acessar a internet, seus recursos de computação devem ser iniciados em uma sub-rede privada configurada para usar um gateway NAT. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC. A comunicação entre nós deve usar o endereço IP privado ou o nome do host DNS para o nó. Trabalhoso paralelos de vários nós executads em recursos de computação em sub-redes públicas não têm acesso à rede de saída. Para criar uma VPC com sub-redes privadas e um gateway NAT, consulte Criando uma Nuvem Privada Virtual .

- As interfaces de rede elástica criadas e anexadas aos seus recursos de computação não podem ser desassociadas manualmente nem modificadas pela sua conta. Isso visa prevenir a exclusão acidental de uma interface de rede elástica associada a uma tarefa em execução. Para liberar as interfaces de rede elásticas para uma tarefa, encerre a tarefa.
- Seu ambiente de computação deve ter máximo de vCPUs suficiente para suportar seus trabalhos paralelos de vários nós.
- Sua cota de instância do Amazon EC2 inclui o número de instâncias exigidas para executar seu trabalho. Por exemplo, suponha que sua tarefa exija 30 instâncias, mas sua conta só pode executar 20 instâncias em uma região. Nesse caso, seu trabalho ficará preso no status `RUNNABLE`.
- Se você especificar um tipo de instância para um grupo de nós em um trabalhos paralelos de vários nós, seu ambiente de computação deverá executar esse tipo de instância.

Trabalhos de GPU

Os trabalhos de GPU permitem executar trabalhos que usam uma GPU da instância.

Os seguintes tipos de instância do Amazon EC2 baseados em GPU são suportados. Para obter mais informações, consulte Instâncias G3 do Amazon EC2, Instâncias G4 do Amazon EC2, Instâncias G5 do Amazon EC2, Instâncias P2 do Amazon EC2, Instâncias P3 do Amazon EC2, Instâncias P4d do Amazon EC2 e Instâncias P5 do Amazon EC2.

Tipo de instância	GPUs	Memória da GPU	vCPUs	Memória	Largura de banda de rede
g3s.xlarge	1	8 GiB	4	30,5 GiB	10 Gbps
g3.4xlarge	1	8 GiB	16	122 GiB	Até 10 Gbps
g3.8xlarge	2	16 GiB	32	244 GiB	10 Gbps
g3.16xlarge	4	32 GiB	64	488 GiB	25 Gbps
g4dn.xlarge	1	16 GiB	4	16 GiB	Até 25 Gbps
g4dn.2xlarge	1	16 GiB	8	32 GiB	Até 25 Gbps
g4dn.4xlarge	1	16 GiB	16	64 GiB	Até 25 Gbps

Tipo de instância	GPUs	Memória da GPU	vCPUs	Memória	Largura de banda de rede
g4dn.8xlarge	1	16 GiB	32	128 GiB	50 Gbps
g4dn.12xlarge	4	64 GiB	48	192 GiB	50 Gbps
g4dn.16xlarge	1	16 GiB	64	256 GiB	50 Gbps
g5.xlarge	1	24 GiB	4	16 GiB	Até 10 Gbps
g5.2xlarge	1	24 GiB	8	32 GiB	Até 10 Gbps
g5.4xlarge	1	24 GiB	16	64 GiB	Até 25 Gbps
g5.8xlarge	1	24 GiB	32	128 GiB	25 Gbps
g5.16xlarge	1	24 GiB	64	256 GiB	25 Gbps
g5.12xlarge	4	96 GiB	48	192 GiB	40 Gbps
g5.24xlarge	4	96 GiB	96	384 GiB	50 Gbps
g5.48xlarge	8	192 GiB	192	768 GiB	100 Gbps
p2.xlarge	1	12 GiB	4	61 GiB	Alta
p2.8xlarge	8	96 GiB	32	488 GiB	10 Gbps
p2.16xlarge	16	192 GiB	64	732 GiB	20 Gbps
p3.2xlarge	1	16 GiB	8	61 GiB	Até 10 Gbps
p3.8xlarge	4	64 GiB	32	244 GiB	10 Gbps
p3.16xlarge	8	128 GiB	64	488 GiB	25 Gbps
p3dn.24xlarge	8	256 GiB	96	768 GiB	100 Gbps
p4d.24xlarge	8	320 GiB	96	1152 GiB	4x100 Gbps
p5.48xlarge	8	640 GiB	192	2 TiB	32x100 Gbps

> **Note**
>
> Somente os tipos de instância que oferecem suporte a uma GPU NVIDIA e usam uma arquitetura x86_64 são compatíveis com trabalhos de GPU no AWS Batch. Por exemplo, as famílias de instâncias `G4ad` e `G5g` não são suportadas.

O parâmetro resourceRequirements da definição de trabalho especifica o número de GPUs a serem fixadas ao contêiner. Esse número de GPUs não está disponível para nenhum outro trabalho executado nessa instância pela duração esse trabalho. Todos os tipos de instâncias em um ambiente de computação que executem trabalhos de GPU devem ser da famílias de instâncias `p2`, `p3`, `p4`, `p5`, `g3`, `g3s`, `g4` ou `g5`. Se isso não for feito, um trabalho de GPU poderá ficar preso no status `RUNNABLE`.

Os trabalhos que não usarem as GPUs podem ser executados em instâncias de GPU. No entanto, eles podem custar mais para serem executados nas instâncias de GPU do que em instâncias semelhantes sem GPU. A depender da vCPU específica, da memória e do tempo necessário, essas tarefas que não são de GPU podem bloquear a execução de tarefas de GPU.

Para criar um trabalho baseado em GPU nos recursos do Amazon EKS

Esta seção aborda como executar uma workload da GPU Amazon EKS no AWS Batch.

Sumário

Para criar um cluster do Kubernetes baseado em GPU no Amazon EKS

Antes de criar um cluster do Kubernetes baseado em GPU no Amazon EKS, você deve ter concluído as etapas em Começando a usar AWS Batch no Amazon EKS. Além disso, considere também o seguinte:

- AWS Batch suporta tipos de instância com GPUs NVIDIA.

- Por padrão, AWS Batch seleciona a AMI acelerada do Amazon EKS com a Kubernetes versão que corresponde à versão do plano de controle do cluster do Amazon EKS.

```
$ cat <<EOF > ./batch-eks-gpu-ce.json
{
  "computeEnvironmentName": "My-Eks-GPU-CE1",
  "type": "MANAGED",
  "state": "ENABLED",
  "eksConfiguration": {
    "eksClusterArn": "arn:aws:eks:<region>:<account>:cluster/<cluster-name>",
    "kubernetesNamespace": "my-aws-batch-namespace"
  },
  "computeResources": {
    "type": "EC2",
    "allocationStrategy": "BEST_FIT_PROGRESSIVE",
    "minvCpus": 0,
    "maxvCpus": 1024,
    "instanceTypes": [
      "p3dn.24xlarge",
      "p4d.24xlarge"
    ],
    "subnets": [
        "<eks-cluster-subnets-with-access-to-internet-for-image-pull>"
    ],
    "securityGroupIds": [
        "<eks-cluster-sg>"
    ],
    "instanceRole": "<eks-instance-profile>"
  }
}
EOF

$ aws batch create-compute-environment --cli-input-json file://./batch-eks-gpu-ce.json
```

AWS Batch não gerencia o plug-in do dispositivo NVIDIA GPU em seu nome. Você deve instalar esse plug-in em seu cluster Amazon EKS e permitir que ele atinja os AWS Batch nós. Para obter mais informações, consulte Habilitando o GPU Support em Kubernetes on. GitHub

Para configurar o plug-in do NVIDIA dispositivo (`DaemonSet`) para direcionar os AWS Batch nós, execute os comandos a seguir.

```
# pull nvidia daemonset spec
$ curl -O https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/v0.12.2/nvidia-device-plugin.yml
# using your favorite editor, add Batch node toleration
# this will allow the DaemonSet to run on Batch nodes
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"

$ kubectl apply -f nvidia-device-plugin.yml
```

Não recomendamos que você misture workloads baseadas em computação (CPU e memória) com workloads baseados em GPU nos mesmos pares de ambiente de computação e fila de trabalhos. Isso ocorre porque os trabalhos de computação podem usar a capacidade da GPU.

Para anexar filas de trabalhos, execute os comandos a seguir.

```
$ cat <<EOF > ./batch-eks-gpu-jq.json
 {
    "jobQueueName": "My-Eks-GPU-JQ1",
    "priority": 10,
    "computeEnvironmentOrder": [
      {
        "order": 1,
        "computeEnvironment": "My-Eks-GPU-CE1"
      }
    ]
  }
EOF

$ aws batch create-job-queue --cli-input-json file://./batch-eks-gpu-jq.json
```

Para criar uma definição de trabalho de GPU do Amazon EKS

Somente o `nvidia.com/gpu` é suportado no momento, e o valor do recurso definido deve ser um número inteiro. Não é possível usar frações da GPU. Para obter mais informações, consulte Agendar GPUs na Kubernetesdocumentação do .

Para registrar uma definição de trabalho de GPU para o Amazon EKS, execute os seguintes comandos.

```
$ cat <<EOF > ./batch-eks-gpu-jd.json
{
    "jobDefinitionName": "MyGPUJobOnEks_Smi",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "hostNetwork": true,
            "containers": [
                {
                    "image": "nvcr.io/nvidia/cuda:10.2-runtime-centos7",
                    "command": ["nvidia-smi"],
                    "resources": {
                        "limits": {
                            "cpu": "1",
                            "memory": "1024Mi",
                            "nvidia.com/gpu": "1"
                        }
                    }
                }
            ]
        }
    }
}
EOF

$ aws batch register-job-definition --cli-input-json file://./batch-eks-gpu-jd.json
```

Para executar um trabalho de GPU em seu Cluster do Amazon EKS

O recurso de GPU não é compactável. AWS Batch cria uma especificação de pod para trabalhos de GPU em que o valor da solicitação é igual ao valor dos limites. Isso é um requisito do Kubernetes.

Para enviar um trabalho de GPU, use os comandos a seguir.

```
$ aws batch submit-job --job-queue My-Eks-GPU-JQ1 --job-definition MyGPUJobOnEks_Smi --
job-name My-Eks-GPU-Job

# locate information that can help debug or find logs (if using Amazon CloudWatch Logs
 with Fluent Bit)
$ aws batch describe-jobs --job <job-id> | jq '.jobs[].eksProperties.podProperties |
 {podName, nodeName}'
{
  "podName": "aws-batch.f3d697c4-3bb5-3955-aa6c-977fcf1cb0ca",
```

```
  "nodeName": "ip-192-168-59-101.ec2.internal"
}
```

Pesquisar e filtrar AWS Batch vagas

Você pode listar os trabalhos em uma fila de trabalhos usando o AWS Batch console. No entanto, quando há muitos trabalhos na fila de trabalhos, pode ser difícil encontrar um trabalho específico.

Você pode usar Pesquisar e Filtrar para listar trabalhos que correspondam aos critérios de pesquisa especificados por você.

1. Abra o AWS Batch console do.
2. Escolha Trabalhos.
3. Ative Pesquisar e Filtrar.

> **Note**
>
> Se tiver vários trabalhos, esse processo poderá levar alguns minutos.

4. Na caixa Selecionar uma Fila de Trabalho, escolha a fila de trabalhos que você deseja pesquisar.
5. Na caixa Filtrar Recursos por Propriedade ou Valor, escolha uma das propriedades listadas.
6. Escolha o operador que deseja usar. Por exemplo, escolha Status =.

> **Tip**
>
> Para usar uma propriedade ou um operador diferente, feche os critérios atuais. Em seguida, escolha a propriedade e a operadora que você deseja.

7. Insira ou escolha um valor de propriedade. Por exemplo, insira todo ou parte do nome do trabalho, ou escolha Status = EXECUTÁVEL.
8. Escolha o trabalho que você deseja na lista de filtros.

> **Tip**
>
> Se não encontrar o trabalho desejado, role pela lista de filtros.

Logs de trabalho

Você pode configurar seus AWS Batch trabalhos para enviar informações de registro para o CloudWatch Logs. Assim, você pode visualizar logs diferentes de trabalhos em um local conveniente. Para ter mais informações, consulte Como usar o CloudWatch Logs com AWS Batch.

Você também pode usar os registros de trabalhos no AWS Batch console para monitorar ou solucionar problemas de um AWS Batch trabalho.

1. Abra o AWS Batch Console.
2. Escolha Trabalhos.
3. Em Fila de Trabalhos, escolha a fila de trabalhos desejada.

> **Tip**
>
> Se houver vários trabalhos na fila de trabalhos, você poderá ativar Pesquisa e Filtragem para encontrar um trabalho mais rapidamente. Para ter mais informações, consulte Pesquisar e filtrar AWS Batch vagas.

4. Em Status, escolha os status do trabalho que você deseja exibir.
5. Escolha o trabalho desejado.
6. Na página Detalhes, role para baixo até Logs de Trabalho.
7. Escolha Recuperar Logs.
8. Para Autorização necessária**OK**, insira e escolha Autorizar para aceitar CloudWatch cobranças da Amazon.

> **Note**
>
> Para revogar sua autorização para CloudWatch cobranças:
>
> 1. No painel de navegação esquerdo, escolha Permissões.
> 2. Em Logs de Trabalho, escolha Editar.
> 3. Desmarque a caixa de CloudWatch seleção Autorizar o uso do Batch.
> 4. Escolha Salvar alterações.

9. Revise os dados de registro do AWS Batch trabalho.

> **Tip**
>
> Você pode filtrar o log com base em Palavras-chave, Máximo de Resultados e Classificação. Você também pode escolher um dos intervalos de tempo padrão ou criar um intervalo personalizado para personalizar os resultados.

Informações sobre o trabalho

Você pode analisar as informações sobre o trabalho do AWS Batch , como status, definição de trabalho e informações do contêiner.

1. Abra o AWS Batch Console.
2. Escolha Trabalhos.
3. Em Fila de Trabalhos, escolha a fila de trabalhos desejada.

> **Tip**
>
> Se houver vários trabalhos na fila de trabalhos, você poderá ativar Pesquisa e Filtragem para encontrar um trabalho mais rapidamente. Para ter mais informações, consulte Pesquisar e filtrar AWS Batch vagas.

4. Escolha o trabalho desejado.

> **Note**
>
> Você também pode usar o AWS Command Line Interface (AWS CLI) para ver detalhes sobre um AWS Batch trabalho. Para obter mais informações, consulte describe-jobs na AWS CLI Referência de Comandos.

Definições de trabalho

AWS Batch as definições de tarefas especificam como as tarefas devem ser executadas. Embora cada tarefa deva fazer referência a uma definição de tarefa, muitos parâmetros que são especificados na definição de tarefa podem ser substituídos em tempo de execução.

Conteúdo

Alguns dos atributos especificados em uma definição de tarefa incluem:

- Qual imagem de docker usar com o contêiner em seu trabalho
- Quantos vCPUs e a quantidade de memória a ser usada com o contêiner
- O comando que o contêiner deve executar quando iniciado
- Quais (se houver alguma) variáveis de ambiente devem ser passadas para o contêiner quando iniciado
- Todos os volumes de dados que devem ser usados com o contêiner
- Qual função do IAM (se houver) que seu trabalho deve usar para obter AWS permissões

Para obter uma descrição completa dos parâmetros disponíveis em uma definição de tarefa, consulte Parâmetros de definição de trabalho para ContainerProperties.

Como criar uma definição de tarefa de nó único

Antes de poder executar trabalhos no AWS Batch, você deve criar uma definição de tarefa. Esse processo varia ligeiramente entre trabalhos em paralelo de nó único e de vários nós. Este tópico aborda especificamente como criar uma definição de trabalho para um trabalho do AWS Batch que não seja um trabalho em paralelo com vários nós.

Você pode criar uma definição de trabalho em paralelo de vários nós nos recursos do Amazon Elastic Container Service. Para obter mais informações, consulte the section called “Criar uma definição de tarefa em paralelo de vários nós”.

Tópicos

Como criar uma definição de trabalho de nó único nos recursos do Amazon EC2

Para criar uma nova definição de trabalho em recursos do Amazon EC2:

1. Abra o console do AWS Batch em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, escolha a Região da AWS a ser usada.
3. No painel de navegação esquerdo, escolha Definições de trabalho.
4. Escolha Create (Criar).
5. Em Tipo de orquestração, escolha Amazon Elastic Compute Cloud (Amazon EC2).
6. Em Configuração da plataforma EC2, desative o processamento de Ativar paralelo de vários nós.
7. Em Nome, insira um nome exclusivo para a sua definição de trabalho. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas e maiúsculas, números, hifens e (-) e sublinhados (_).
8. (Opcional) Em Tempo limite de execução, insira o valor do tempo limite (em segundos). O tempo limite de execução é o período de tempo antes que um trabalho não concluído seja encerrado. Se uma tentativa exceder o tempo limite, ela será interrompida e passada para um status

`FAILED`. Para obter mais informações, consulte Tempos limite do trabalho. O valor mínimo é 60 segundos.

9. (Opcional) Ative Prioridade de agendamento. Insira um valor de prioridade de agendamento entre 0 e 100. Valores mais altos têm maior prioridade.
10. (Opcional) Para Tentativas de trabalho, insira o número de vezes que o AWS Batch tenta mover o trabalho para o status `RUNNABLE`. Insira um número inteiro entre 1 e 10.
11. (Opcional) Em Repetir as condições da estratégia, escolha Adicionar avaliação na saída. Insira pelo menos um valor de parâmetro e escolha uma Ação. Para cada conjunto de condições, Ação deve ser definida como Tentar novamente ou Sair. Essas ações significam o seguinte:

    - Tentar novamente – O AWS Batch tenta novamente até que o número de tentativas de trabalho que você especificou seja atingido.
    - Sair – O AWS Batch para de tentar novamente o trabalho.

    > **Important**
    >
    > Se você escolher Adicionar avaliação na saída, deverá configurar pelo menos um parâmetro e ou escolher uma Ação ou escolher Remover avaliação na saída.

12. (Opcional) Expanda Tags e então escolha Adicionar tag para adicionar tags ao recurso. Insira uma chave e um valor opcional e então escolha Adicionar tag.
13. (Opcional) Ative Propagar tags para propagar tags do trabalho e da definição de trabalho para a tarefa do Amazon ECS.
14. Escolha Próxima página.
15. Na seção Configuração do contêiner:

    a. Para Imagem, escolha a imagem do Docker a ser usada em seu trabalho. Por padrão, as imagens no registro do Docker Hub estão disponíveis. Você também pode especificar outros repositórios com *`repository-url`*`/`*`image`*`:`*`tag`*. Os nomes podem ter até 225 caracteres. Pode conter letras maiúsculas e minúsculas, números, hifens (-), sublinhados (_), dois pontos (:), barras (/) e sinais de número (#). Esse parâmetro é mapeado para `Image` na seção Criar um contêiner da API remota do Docker e o parâmetro `IMAGE` de docker run.

> **Note**
>
> A arquitetura da imagem do Docker deve corresponder à arquitetura do processador dos recursos de computação nos quais eles foram programados. Por exemplo, imagens do Docker baseadas no Arm só podem ser executadas em recursos de computação baseados no Arm.

- As imagens em repositórios públicos do Amazon ECR usam as convenções de nomenclatura `registry/repository[:tag]` ou `registry/repository[@digest]` completa (por exemplo, `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).
- As imagens em repositórios do Amazon ECR usam as convenções de nomenclatura `registry/repository[:tag]` completa (por exemplo, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*).
- As imagens em repositórios oficiais no Docker Hub usam um único nome (por exemplo, `ubuntu` ou `mongo`).
- As imagens em outros repositórios no Docker Hub são qualificadas com um nome de organização (por exemplo, `amazon/amazon-ecs-agent`).
- As imagens em outros repositórios online são ainda mais qualificadas por um nome de domínio (por exemplo, `quay.io/assemblyline/ubuntu`).

b. Em Sintaxe de comando, escolha Bash ou JSON.

c. Para Comando, especifique o comando a ser passado para o contêiner. Para comandos mais simples, insira o comando da mesma forma que você faz para um prompt de comando. Em seguida, verifique se o resultado JSON está correto e passe para o Docker daemon. Para comandos mais complicados (por exemplo, com caracteres especiais), use a sintaxe JSON.

> **Tip**
>
> Escolha Informações para visualizar Bash e amostras de códigos de JSON.

Esse parâmetro é mapeado para `Cmd` na seção Criar um contêiner da API remota do Docker e o parâmetro `COMMAND` de docker run. Para obter mais informações sobre o parâmetro `CMD` do Docker, consulte https://docs.docker.com/engine/reference/builder/#cmd.

> **Note**
>
> Você pode usar os valores padrão de substituição de parâmetros e espaços reservados no seu comando. Para obter mais informações, consulte Parâmetros.

d. (Opcional) Para Função de execução, especifique um perfil do IAM que conceda aos atendentes de contêiner do Amazon ECS permissão para fazer chamadas da API da AWS em seu nome. Esse atributo usa perfis do IAM do Amazon ECS para tarefas. Para obter mais informações, consulte Perfis do IAM para execução de tarefa do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

e. Para a configuração de Perfil de trabalho, escolha um perfil do IAM que tenha permissões para as APIs da AWS. Esse atributo usa perfis do IAM do Amazon ECS para tarefas. Para obter mais informações, consulte Funções do IAM para tarefas no Guia do desenvolvedor do Amazon Elastic Container Service.

> **Note**
>
> Somente funções que tenham o relacionamento de confiança Função da tarefa do Amazon Elastic Container Service são mostradas aqui. Para obter mais informações sobre como criar um perfil do IAM para os trabalhos do seu AWS Batch, consulte Como criar um perfvil do IAM e uma política para suas tarefas no Guia do desenvolvedor do Amazon Elastic Container Service.

16. Em Parâmetros, escolha Adicionar parâmetros para adicionar espaços reservados para substituição de parâmetros como pares de chave e valor opcionais.
17. Na seção Configuração do ambiente:

    a. Para vCPUs, especifique o número de vCPUs a serem reservadas para o contêiner. Esse parâmetro é mapeado para `CpuShares` na seção Criar um contêiner da API remota do Docker e a opção `--cpu-shares` para docker run. Cada vCPU é equivalente a 1.024 compartilhamentos de CPU. Você deve especificar pelo menos uma vCPU.

    b. Em Memória, insira o limite de memória disponível para o contêiner. Caso tente exceder a quantidade memória especificada aqui, o contêiner será interrompido. Esse parâmetro é mapeado para `Memory` na seção Criar um contêiner da API remota do Docker e a opção `--memory` para docker run. Você deve especificar pelo menos 4 MiB de memória para um trabalho.

> **Note**
>
> Para maximizar o uso dos recursos, priorize a memória para trabalhos de um tipo específico de instância. Para obter mais informações, consulte Recurso de Computação Gerenciamento de Memória.

c. Em Número de GPUs, escolha o número de GPUs a serem reservadas para o contêiner.

d. (Opcional) Em Variáveis de ambiente, escolha Adicionar variáveis de ambiente para adicionar variáveis de ambiente como pares de nome/valor. Essas variáveis serão passadas para o contêiner.

e. (Opcional) Em Segredos, escolha Adicionar segredo para adicionar segredos como pares de nome-valor. Esses segredos são expostos no contêiner. Para obter mais informações, consulte SecretOptions em Parâmetros de definição de trabalho para ContainerProperties.

18. Escolha Próxima página.

19. Na seção Configuração do Linux:

    a. Para Usuário, insira o nome do usuário a ser usado dentro do contêiner. Esse parâmetro é mapeado para `User` na seção Criar um contêiner da API remota do Docker e a opção `--user` para docker run.

    b. (Opcional) Para oferecer ao contêiner de trabalho permissões elevadas na instância do host (semelhante ao usuário `root`), arraste o controle deslizante Privilegiado para a direita. Esse parâmetro é mapeado para `Privileged` na seção Criar um contêiner da API remota do Docker e a opção `--privileged` para docker run.

    c. (Opcional) Ative Ativar processo init para executar um processo `init` dentro do contêiner. Este processo encaminha sinais e colhe processos.

20. (Opcional) Na seção Configuração do sistema de arquivos:

    a. Ative Habilitar sistema de arquivos somente para leitura para remover o acesso de gravação ao volume.

    b. Para Tamanho da memória compartilhada, insira o tamanho (em MiB) do volume `/dev/shm`.

    c. Em Tamanho máximo de troca, insira a quantidade total de memória de troca (em MiB) que o contêiner pode usar.

    d. Em Swappiness, insira um valor entre 0 e 100 para indicar o comportamento de troca de memória do contêiner. Se você não especificar um valor e a troca estiver ativada, o

valor assumirá 60 como padrão. Para obter mais informações, consulte swappiness em Parâmetros de definição de trabalho para ContainerProperties.

e. (Opcional) Expanda Configuração adicional.

f. (Opcional) Para Tmpfs, escolha Adicionar tmpfs para adicionar uma montagem `tmpfs`.

g. (Opcional) Em Dispositivos, escolha Adicionar dispositivo para adicionar um dispositivo:

   i. Em Container path (Caminho do contêiner), especifique o caminho na instância de contêiner para expor o dispositivo mapeado para a instância de host. Se isso for deixado em branco, o caminho de host será usado no contêiner.

   ii. Em Host path (Caminho do host), especifique o caminho de um dispositivo na instância de host.

   iii. Em Permissões, selecione uma ou mais permissões para serem aplicadas ao dispositivo. As permissões disponíveis são LER, GRAVAR e MKNOD.

h. (Opcional) Para Configuração de volumes, escolha Adicionar volume para criar uma lista de volumes a serem passados para o contêiner. Insira Nome e Caminho de origem para o volume e escolha Adicionar volume. Você também pode optar por ativar o Habilitar EFS.

i. (Opcional) Em Pontos de montagem, escolha Adicionar configuração de pontos de montagem para adicionar pontos de montagem para volumes de dados. Você deve especificar o volume de origem e o caminho do contêiner. Esses pontos de montagem são passados para o Docker daemon em uma instância de contêiner. Você também pode optar por tornar o volume Somente para leitura.

j. (Opcional) Para Configuração de ulimits, escolha Adicionar ulimit para adicionar um valor de `ulimits` ao contêiner. Insira os valores Nome, Limite flexível e Limite rígido e então escolha Adicionar ulimit.

21. (Opcional) Na seção Configuração do log:

a. Em Driver de log, escolha o driver de log a ser usado. Para obter mais informações sobre os drivers de log disponíveis, consulte logDriver emParâmetros de definição de trabalho para ContainerProperties.

> **Note**
>
> Por padrão, o driver de log `awslogs` é usado.

b. Em Opções, escolha Adicionar opção para adicionar uma opção. Insira um par nome-valor e então escolha Adicionar opção.

c. Em Segredos, escolha Adicionar segredo. Insira um par nome-valor e escolha Adicionar segredo para adicionar um segredo.

> **Tip**
>
> Para obter mais informações, consulte SecretOptions em Parâmetros de definição de trabalho para ContainerProperties.

22. Escolha Próxima página.
23. Em Avaliação da definição de trabalho, revise as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Criar definição de trabalho.

Como criar uma definição de trabalho de nó único nos recursos do AWS Fargate

Para criar uma nova definição de trabalho em recursos do AWS Fargate:

1. Abra o console AWS Batch em https://console.aws.amazon.com/batch/.
2. Na barra de navegação superior, escolha a Região da AWS a ser usada.
3. No painel de navegação esquerdo, escolha Definições de Tarefa.
4. Escolha Criar.
5. Para Tipo de orquestração, escolha Fargate. Para ter mais informações, consulte AWS Batch no AWS Fargate.
6. Em Nome, insira um nome exclusivo para a sua definição de trabalho. Os nomes podem ter até 128 caracteres. Podem conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).
7. (Opcional) Em Tempo Limite de Execução, insira o valor do tempo limite (em segundos). O tempo limite de execução é o período de tempo antes que um trabalho não concluído seja encerrado. Caso uma tentativa exceda o tempo limite, ela será interrompida e transferida para `FAILED`status. Para mais informações, consulte Tempos limite do trabalho. O valor mínimo é 60 segundos.
8. (Opcional) Ative Prioridade de Agendamento. Insira um valor de prioridade de agendamento entre 0 e 100. Valores mais altos têm maior prioridade sobre valores mais baixos.
9. (Opcional) Expanda Tags e então escolha Adicionar tag para adicionar tags ao recurso. Ative Propagar tags para propagar tags do trabalho e da definição do trabalho.

10. Na seção Configuração da plataforma Fargate:

    a. Para a plataforma Runtime, escolha a arquitetura do ambiente de computação.

    b. Em Família do sistema operacional, escolha o sistema operacional do ambiente de computação.

    c. Em Arquitetura da CPU, escolha a arquitetura vCPU.

    d. Para Versão da plataforma Fargate, insira `LATEST` ou uma versão específica do ambiente de runtime.

    e. (Opcional) Ative a opção Atribuir IP públicopara atribuir um endereço IP público a uma interface de rede de tarefas do Fargate. Para um trabalho que é executado em uma sub-rede privada para enviar tráfego de saída para a Internet, a sub-rede privada exige que um gateway NAT seja anexado para rotear solicitações para a Internet. Ao fazer isso, você poderá extrair imagens de contêineres. Para obter mais informações, consulte Rede de tarefas do Amazon ECS no Guia do desenvolvedor do Serviço Amazon Elastic Container.

    f. (Opcional) Em Armazenamento temporário, insira a quantidade de armazenamento temporário a ser alocada para a tarefa. A quantidade de armazenamento temporário deve estar entre 21 GiB e 200 GiB. Por padrão, 20 GiB de armazenamento temporário são alocados se você não inserir um valor.

    > **Note**
    >
    > O armazenamento temporário requer a versão da plataforma Fargate 1.4 ou posterior.

    g. Para Função de execução, especifique um perfil do IAM que conceda aos atendentes de contêiner do Amazon ECS e do Fargate permissão para fazer chamadas da API da AWS em seu nome. Esse atributo usa perfis do IAM do Amazon ECS para funcionalidade de tarefa. Para obter mais informações, incluindo pré-requisitos de configuração, consulte Perfis do IAM para execução de tarefa do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

    h. Para Tentativas de trabalho, insira o número de vezes que o AWS Batch tenta mover o trabalho para o status de `RUNNABLE`. Insira um número inteiro entre 1 e 10.

    i. (Opcional) Em Repetir as condições da estratégia, escolha Adicionar avaliação na saída. Insira pelo menos um valor de parâmetro e escolha uma Ação. Para cada conjunto de condições, Ação deve ser definida como Tentar Novamente ou Sair. Essas ações significam o seguinte:

- Tentar novamente – AWS Batch tenta novamente até que o número de tentativas de trabalho que você especificou seja atingido.
- Sair – O AWS Batch para de tentar novamente o trabalho.

> **Important**
>
> Se você escolher Adicionar avaliação na saída, deverá configurar pelo menos um parâmetro e escolher uma Ação ou escolher Remover avaliação na saída.

11. Escolha Próxima página.
12. Na seção Configuração do contêiner:

    a. Para Imagem, escolha a imagem do Docker a ser usada em seu trabalho. Por padrão, imagens no registro Docker Hub estarão disponíveis. Você também pode especificar outros repositórios com *`repository-url`*`/`*`image`*`:`*`tag`*. Os nomes podem ter até 225 caracteres. Pode conter letras maiúsculas, minúsculas, números, hifens (-), sublinhados (_), dois pontos (:), pontos (.), barras (/) e jogos da velha (#). Esse parâmetro é mapeado para `Image` na seção Criar um contêiner da Docker Remote API e o parâmetro `IMAGE` de docker run.

> **Note**
>
> Docker arquitetura da imagem do Docker deve corresponder à arquitetura do processador dos recursos de computação nos quais eles foram programados. Por exemplo, imagens Arm baseadas Docker em só podem ser executadas em recursos de computação baseados em Arm.

- As imagens em repositórios públicos Amazon ECR usam convenções de nomenclatura completas `registry/repository[:tag]` ou `registry/repository[@digest]` (por exemplo, `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).
- As imagens em repositórios Amazon ECR usam convenções de nomenclatura completas `registry/repository[:tag]` (por exemplo, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*).
- Imagens em repositórios oficiais em Docker Hub usam um único nome (por exemplo, `ubuntu` ou `mongo`).

- Imagens em outros repositórios Docker Hub são qualificadas com um nome de organização (por exemplo, `amazon/amazon-ecs-agent`).
- Imagens em outros repositórios online também são qualificadas por um nome de domínio (por exemplo, `quay.io/assemblyline/ubuntu`).

b. Para a Sintaxe de comando, escolha Bash ou JSON.

c. Para Command, especifique o comando a ser passado para o contêiner. Para comandos simples, insira o comando como você faz para um prompt de comando e, em seguida, verifique se o resultado JSON está correto. Foi passado para o daemon do Docker. Para comandos mais complicados (por exemplo, com caracteres especiais), use a sintaxe JSON.

> **Tip**
>
> Escolha Informações para visualizar Bash e amostras de códigos de JSON.

Esse parâmetro é mapeado para `Cmd` na seção Criar um contêiner da Docker Remote API e o parâmetro `COMMAND` de docker run. Para obter mais informações sobre o parâmetro Docker `CMD`, consulte https://docs.docker.com/engine/reference/builder/#cmd.

> **Note**
>
> Você pode usar os valores padrão de substituição de parâmetros e espaços reservados no seu comando. Para ter mais informações, consulte Parâmetros.

d. (Opcional) Adicione parâmetros à definição do trabalho como mapeamentos de nome-valor para substituir os padrões de definição do trabalho. Para adicionar um parâmetro:

- Em Parâmetros, escolha Adicionar parâmetros, insira um par nome-valor e então escolha Adicionar parâmetro.

> **Important**
>
> Se você escolher Adicionar parâmetro, deverá configurar pelo menos um parâmetro ou escolher Remover parâmetro

e. Na seção Configuração do ambiente:

i. Para a configuração de Perfil de trabalho, escolha um perfil do IAM que tenha permissões para as APIs da AWS. Esse atributo usa perfis do IAM do Amazon ECS para funcionalidade de tarefa. Para mais informações, consulte Funções do IAM para Tarefas no Guia de Desenvolvedor Amazon Elastic Container Service.

> **Note**
>
> Somente funções que tenham o relacionamento de confiança Função da tarefa do Amazon Elastic Container Service são mostradas aqui. Para obter mais informações sobre como criar um perfil do IAM para os trabalhos do seu AWS Batch, consulte Como criar um perfil do IAM e uma política para suas tarefas no Guia do desenvolvedor do Amazon Elastic Container Service.

ii. Para vCPUs, especifique o número de vCPUs a serem reservadas para o contêiner. Esse parâmetro é mapeado para `CpuShares` na seção Criar um Contêiner da API remota do Docker e a opção `--cpu-shares` para docker run. Cada vCPU equivale a 1.024 compartilhamentos de CPU. Você deve especificar pelo menos uma vCPU.

iii. Em Memória, insira o limite de memória que está disponível para o contêiner. Caso seu contêiner tente exceder a memória especificada, o mesmo será interrompido. Esse parâmetro é mapeado para `Memory` na seção Criar um Contêiner da API Remota Docker e para a opção `--memory` para docker run. Você deve especificar pelo menos 4 MiB de memória para uma tarefa.

Se você usa o GuardDuty Runtime Monitoring, há uma pequena sobrecarga de memória para o agente GuardDuty de segurança. Portanto, o limite de memória deve incluir o tamanho do agente GuardDuty de segurança. Para obter informações sobre os limites de memória do GuardDuty Security Agent, consulte Limites de CPU e memória no Guia GuardDuty do usuário. Para obter informações sobre as melhores práticas, consulte Como faço para corrigir erros de falta de memória em minhas tarefas do Fargate depois de ativar o Runtime Monitoring no Amazon ECS Developer Guide.

> **Note**
>
> Para maximizar o uso dos recursos, priorize a memória para trabalhos de um tipo específico de instância. Para ter mais informações, consulte Recurso de Computação Gerenciamento de Memória.

f. (Opcional) Em Variáveis de Ambiente, escolha Adicionar Variável de Ambiente para adicionar variáveis de ambiente como pares nome-valor. Essas variáveis serão passadas para o contêiner.

g. (Opcional) Em Segredos, escolha Adicionar segredo para adicionar segredos como pares de nome-valor. Esses segredos são expostos no contêiner. Para obter mais informações, consulte SecretOptions em Parâmetros de definição de trabalho para ContainerProperties.

h. Escolha Próxima página.

13. (Opcional) Na seção Configuração do Linux:

a. Em Usuário, insira um nome do usuário a ser usado dentro do contêiner.

b. Ative Ativar processo init para executar um processo init dentro do contêiner. Este processo encaminha sinais e colhe processos.

c. Ative Habilitar sistema de arquivos somente para leitura para remover o acesso de gravação ao volume.

d. (Opcional) Expanda Configuração adicional.

e. Em Configuração dos pontos de montagem, escolha Adicionar configuração de pontos de montagem para adicionar pontos de montagem para volumes de dados. Você deve especificar o volume de origem e o caminho do contêiner. Esses pontos de montagem são passados para o Docker daemon em uma instância de contêiner.

f. Para Configuração de volumes, escolha Adicionar volume para criar uma lista de volumes a serem passados para o contêiner. Insira Nome e Caminho de origem para o volume, e então escolha Adicionar volume.

g. Na seção Configuração do log:

i. (Opcional) Em Driver de log, escolha o driver de log a ser usado. Para obter mais informações sobre os drivers de log disponíveis, consulte logDriver emParâmetros de definição de trabalho para ContainerProperties.

> **Note**
>
> Por padrão, o driver de log `awslogs` é usado.

ii. (Opcional) Em Opções, escolha Adicionar opção para adicionar uma opção. Insira um par nome-valor e então escolha Adicionar opção.

iii. (Opcional) Em Segredos, escolha Adicionar segredo para adicionar um segredo. Em seguida, insira um par nome-valor e escolha Adicionar segredo.

> **Tip**
>
> Para obter mais informações, consulte SecretOptions em Parâmetros de definição de trabalho para ContainerProperties.

14. Escolha Próxima página.
15. Para a Avaliação da definição de trabalho, revise as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Criar Definição de Trabalho.

Como criar uma definição de tarefa de nó único nos recursos Amazon EKS

Para criar uma nova definição de tarefa nos recursos Amazon Elastic Kubernetes Service:

1. Abra o console AWS Batch em https://console.aws.amazon.com/batch/.
2. Na barra de navegação superior, escolha a Região da AWS a ser usada.
3. No painel de navegação esquerdo, escolha Definições de Tarefa.
4. Escolha Criar.
5. Em Tipo de Orquestração, escolha Elastic Kubernetes Service (EKS).
6. Em Nome, insira um nome exclusivo para a sua definição de trabalho. Os nomes podem ter até 128 caracteres. Podem conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).
7. (Opcional) Em Tempo Limite de Execução, insira o valor do tempo limite (em segundos). O tempo limite de execução é o período de tempo antes que um trabalho não concluído seja encerrado. Caso uma tentativa exceda o tempo limite, ela será interrompida e transferida para `FAILED`status. Para mais informações, consulte Tempos limite do trabalho. O valor mínimo é 60 segundos.
8. (Opcional) Ative Prioridade de Agendamento. Insira um valor de prioridade de agendamento entre 0 e 100. Valores mais altos têm maior prioridade sobre valores mais baixos.
9. (Opcional) Expanda Tags e então, escolha Adicionar Tag para incluir tags no recurso.
10. Escolha Próxima página.
11. Na seção Propriedades pod EKS:

a. Em Nome da Conta de Serviço, insira uma conta que forneça uma identidade para processos executados em pod.

b. Ative a rede Host para usar o modelo de rede Kubernetes pod e abrir uma porta de recepção para conexões de entrada. Desative essa configuração somente para comunicações de saída.

c. Em Política DNS, escolha uma das opções a seguir:

- Sem Valor (Nulo) – O pod ignora as configurações DNS do ambiente Kubernetes.
- Padrão – O pod herda a configuração de resolução de nomes do nó onde for executado.

> **Note**
>
> Se uma política DNS não for especificada, o Padrão não será a política DNS padrão. O ClusterFirst será usado no lugar.

- ClusterFirst – Qualquer consulta ao DNS que não corresponda ao sufixo de domínio de cluster configurado será encaminhada para o servidor de nomes upstream herdado do nó.
- ClusterFirstWithHostNet – Utilize caso a Rede Host esteja ativada.

d. (Opcional) Em Rótulos de Pod, escolha Adicionar Rótulos de Pod e então, insira um par nome-valor.

> **Important**
>
> O prefixo de um rótulo de pod não pode conter `kubernetes.io/`, `k8s.io/` ou `batch.amazonaws.com/`.

e. Escolha Próxima página.

f. Na seção Configuração de Contêiner:

i. Em Nome, insira um nome exclusivo para o contêiner. O nome deve começar com uma letra ou número e pode conter até 63 caracteres. Pode conter letras minúsculas, maiúsculas, números e hífens (-).

ii. Para Imagem, escolha a imagem Docker a ser usada em sua tarefa. Por padrão, imagens no registro Docker Hub estarão disponíveis. Você também pode especificar outros repositórios com *`repository-url`*`/`*`image`*`:`*`tag`*. O nome pode ter até 255

caracteres. Pode conter letras maiúsculas, minúsculas, números, hifens (-), sublinhados (_), dois pontos (:), pontos (.), barras (/) e jogos da velha (#). Esse parâmetro é mapeado para `Image` na seção Criar um Contêiner da API remota Docker e do parâmetro `IMAGE` de docker run

> **Note**
>
> A arquitetura de imagem do Docker deve corresponder à arquitetura do processador dos recursos de computação nos quais eles foram programados. Por exemplo, imagens Arm baseadas Docker em só podem ser executadas em recursos de computação baseados em Arm.

- As imagens em repositórios públicos Amazon ECR usam convenções de nomenclatura completas `registry/repository[:tag]` ou `registry/repository[@digest]` (por exemplo, `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).
- As imagens em repositórios Amazon ECR usam convenções de nomenclatura completas `registry/repository[:tag]` (por exemplo, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*).
- Imagens em repositórios oficiais em Docker Hub usam um único nome (por exemplo, `ubuntu` ou `mongo`).
- Imagens em outros repositórios Docker Hub são qualificadas com um nome de organização (por exemplo, `amazon/amazon-ecs-agent`).
- Imagens em outros repositórios online também são qualificadas por um nome de domínio (por exemplo, `quay.io/assemblyline/ubuntu`).

iii. (Opcional) Em Política de Extração de Imagem, escolha quando as imagens serão puxadas.

iv. (Opcional) Em Comando, insira um comando Bash ou JSON para passar para o contêiner.

v. (Opcional) Em Argumentos, insira argumentos a serem transmitidos ao contêiner. Caso um argumento não seja fornecido, o comando de imagem de contêiner será usado.

g. (Opcional) Você pode adicionar parâmetros a definição do trabalho como mapeamentos de nome-valor para substituir padrões da definição da tarefa. Para adicionar um parâmetro:

- Em Parâmetros, insira um par nome-valor e então, escolha Adicionar Parâmetro.

> **Important**
>
> Se escolher Adicionar Parâmetro, deverá configurar pelo menos um parâmetro, ou escolher Remover Parâmetro

h. Na seção Configuração de Ambiente:

i. Para vCPUs, insira o número de vCPUs a serem reservados para o contêiner. Esse parâmetro é mapeado para `CpuShares` na seção Criar um Contêiner da API Remota Docker e para a opção `--cpu-shares` para docker run. Cada vCPU equivale a 1.024 compartilhamentos de CPU. Você deve especificar pelo menos uma vCPU.

ii. Em Memória, insira o limite de memória disponível para o contêiner. Caso seu contêiner tente exceder a memória especificada, o mesmo será interrompido. Esse parâmetro é mapeado para `Memory` na seção Criar um Contêiner da API Remota Docker e para a opção `--memory` para docker run. Você deve especificar pelo menos 4 MiB de memória para uma tarefa.

> **Note**
>
> Para maximizar o uso dos recursos, priorize a memória para trabalhos de um tipo específico de instância. Para mais informações, consulte Recurso de Computação Gerenciamento de Memória.

i. (Opcional) Em Variáveis de Ambiente, escolha Adicionar Variável de Ambiente para adicionar variáveis de ambiente como pares nome-valor. Essas variáveis serão passadas para o contêiner.

j. (Opcional) Em Montagem de Volume:

i. Escolha Adicionar Montagem de Volume.

ii. Insira um Nome e, em seguida, um Caminho de Montagem no contêiner onde o volume estiver montado.

iii. Escolha Somente Leitura para remover permissões de registro no volume.

iv. Escolha Adicionar Montagem de Volume.

k. (Opcional) Em Executar como Usuário, insira uma ID de usuário para executar o processo do contêiner.

> **Note**
>
> A ID do usuário deve existir na imagem para que o contêiner seja executado.

l. (Opcional) Em Executar como Grupo, insira uma ID de grupo para executar o tempo de processamento runtime do contêiner.

> **Note**
>
> A ID do grupo deve existir na imagem para que o contêiner seja executado.

m. (Opcional) Para conceder ao contêiner de trabalho permissões elevadas na instância do host (semelhante ao usuário `root`), arraste o slider Privilegiado para a direita. Esse parâmetro é mapeado para `Privileged` na seção Criar um Contêiner da API Remota Docker e para a opção `--privileged` para docker run.

n. (Opcional) Ative o Sistema de Arquivamento Raiz Somente para Leitura para remover o acesso de registro ao sistema de arquivamento raiz.

o. (Opcional) Ative Executar como Não Raiz para executar contêineres em pod como usuário não raiz.

> **Note**
>
> Caso a Execução como Não Raiz esteja ligada, o kubelet valida a imagem em runtime para verificar se a mesma não será executada como UID 0.

p. Escolha Próxima página.

12. Para a Avaliação da definição de trabalho, revise as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Criar Definição de Trabalho.

Criar uma definição de tarefa em paralelo de vários nós

Antes de poder executar trabalhos no AWS Batch, você deve criar uma definição de tarefa. Esse processo varia ligeiramente para tarefas em paralelo de nó único e de vários nós. Este tópico aborda

como criar uma definição de trabalho para um trabalho em paralelo de vários nós do AWS Batch. Para obter mais informações, consulte Trabalhos paralelos de vários nós .

> **Note**
>
> AWSO Fargate não oferece suporte a trabalhos paralelos de de vários nós.

Criar uma definição de trabalho em paralelo de vários nós nos recursos do Amazon EC2

Para criar uma definição de tarefa de nó único, consulte Como criar uma definição de tarefa de nó único .

Para criar uma definição de trabalho paralelo de vários nós nos recursos do Amazon Elastic Compute Cloud:

1. Abra o console do AWS Batch em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione a Região da AWS a ser usada.
3. No painel de navegação, escolha Definições de trabalho.
4. Escolha Create.
5. Para o Tipo de orquestração, escolha Amazon Elastic Compute Cloud (Amazon EC2).
6. Em Habilitar paralelo de vários nós, ative o paralelo de vários nós.
7. Para Nome, insira um nome exclusivo para a sua definição de trabalho. O nome pode ter até 128 caracteres e pode conter letras minúsculas e maiúsculas, números, hifens (-) e sublinhados (_).
8. (Opcional) Em Execution timeout, especifique o número máximo de segundos permitidos para as execuções das tentativas de trabalho. Se uma tentativa exceder o tempo limite, ela será interrompida, e o status será alterado para `FAILED`. Para obter mais informações, consulte Tempos limite do trabalho.
9. (Opcional) Ative a Prioridade de agendamento. Insira um valor de prioridade de agendamento entre 0 e 100. Valores mais altos têm maior prioridade sobre valores mais baixos.
10. (Opcional) Para Tentativas de trabalho, insira o número de vezes que elas AWS Batch tentam mover o trabalho para o `RUNNABLE` status. Insira um número inteiro entre 1 e 10.

11. (Opcional) Em Repetir as condições da estratégia, escolha Adicionar avaliação na saída. Insira pelo menos um valor de parâmetro e escolha uma Ação. Para cada conjunto de condições, Ação deve ser definida como Tentar Novamente ou Sair. Essas ações significam o seguinte:

    - Tentar novamente – AWS Batch tenta novamente até que o número de tentativas de trabalho que você especificou seja atingido.
    - Sair – O AWS Batch para de tentar novamente o trabalho.

    > **Important**
    >
    > Se você escolher Add evaluate on exit, deverá configurar pelo menos um parâmetro e escolher uma Action ou escolher Remove evaluate on exit.

12. (Opcional) Expanda Tags e então, escolha Adicionar Tag para adicionar tags ao recurso. Insira uma chave e um valor opcional e, em seguida, selecione Add tag. Você também pode ativar a opção Propagate tags para propagá-las do trabalho e da definição de trabalho para a tarefa do Amazon ECS.
13. Escolha Próxima página.
14. Em Number of nodes, insira o número total de nós a serem usados em sua tarefa.
15. Em Main node, insira o índice do nó a ser usado para o nó principal. O índice do nó principal padrão é `0`.
16. Em Instance Type, selecione um tipo de instância.

    > **Note**
    >
    > O tipo de instância que você escolher se aplica a todos os nós.

17. Em Parâmetros, escolha Adicionar parâmetros para adicionar espaços reservados para substituição de parâmetros como pares de chave e valor opcionais.
18. Na seção Node ranges:

    a. Selecione Add node range. Isso cria uma seção Node range.

    b. Em Target nodes, especifique o intervalo para o seu grupo de nós usando a anotação *`range_start`*`:`*`range_end`*.

Você pode criar até cinco intervalos de nós para os nós especificados para seu trabalho. Os intervalos de nós usam o valor do índice para um nó e o índice do nó começa em 0. Certifique-se de que o valor do índice final do intervalo do seu grupo final de nós seja um a menos do que o número de nós que você especificou. Por exemplo, suponha que você especificou 10 nós e deseja usar um único grupo de nós. Então, seu intervalo final é 9.

c. Para Image, escolha a imagem Docker a ser usada em seu trabalho. Por padrão, as imagens no registro do Docker Hub estão disponíveis. Você também pode especificar outros repositórios com *`repository-url`*`/`*`image`*`:`*`tag`*. O nome pode ter até 225 caracteres. Pode conter letras maiúsculas e minúsculas, números, hifens (-), sublinhados (_), dois pontos (:), pontos (.), barras (/) e sinais de número (#). Esse parâmetro é mapeado para `Image` na seção Criar um contêiner da Docker Remote API e o parâmetro `IMAGE` de docker run.

> **Note**
>
> A arquitetura da imagem do Docker deve corresponder à arquitetura do processador dos recursos de computação nos quais eles foram programados. Por exemplo, imagens Docker baseadas em Arm só podem ser executadas em recursos de computação baseados em Arm.

- As imagens em repositórios públicos Amazon ECR usam as convenções de nomenclatura completas `registry/repository[:tag]` ou `registry/repository[@digest]` (por exemplo, `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).
- As imagens nos repositórios do Amazon ECR usam a convenção de `registry/repository[:tag]` nomenclatura completa. Por exemplo, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*
- As imagens em repositórios oficiais no Docker Hub usam um único nome (por exemplo, `ubuntu` ou `mongo`).
- As imagens em outros repositórios no Docker Hub são qualificadas com um nome de organização (por exemplo, `amazon/amazon-ecs-agent`).
- As imagens em outros repositórios online são ainda mais qualificadas por um nome de domínio (por exemplo, `quay.io/assemblyline/ubuntu`).

d. Para a Sintaxe de comando, escolha Bash ou JSON.

e. Para Command, especifique o comando a ser passado para o contêiner. Para comandos simples, você pode digitar o comando como faria em um prompt de comando na guia Delimitado por espaço. Em seguida, verifique se o resultado JSON está correto. O resultado do JSON é passado para o Docker daemon. Para comandos mais complicados (por exemplo, com caracteres especiais), você pode alternar para a guia JSON e inserir a matriz de string equivalente.

   Esse parâmetro é mapeado para `Cmd` na seção Criar um contêiner da Docker Remote API e o parâmetro `COMMAND` de docker run. Para obter mais informações sobre o parâmetro Docker `CMD`, consulte https://docs.docker.com/engine/reference/builder/#cmd.

   > **Note**
   >
   > Você pode usar os valores padrão de substituição de parâmetros e espaços reservados no seu comando. Para obter mais informações, consulte Parâmetros.

f. Para vCPUs, especifique o número de vCPUs a serem reservadas para o contêiner. Esse parâmetro é mapeado para `CpuShares` na seção Criar um contêiner da API remota do Docker e a opção `--cpu-shares` para docker run. Cada vCPU é equivalente a 1.024 compartilhamentos de CPU. Você deve especificar pelo menos uma vCPU.

g. Para Memória, especifique o limite rígido (em MiB) de memória a ser apresentado ao contêiner do trabalho. Caso seu contêiner tente exceder a memória especificada, o mesmo será interrompido. Esse parâmetro é mapeado para `Memory` na seção Criar um contêiner da API remota do Docker e a opção `--memory` para docker run. Você deve especificar pelo menos 4 MiB de memória para um trabalho.

   > **Note**
   >
   > Para maximizar a utilização de recursos, você pode fornecer aos seus trabalhos o máximo de memória possível para um determinado tipo de instância. Para obter mais informações, consulte Recurso de Computação Gerenciamento de Memória.

h. (Opcional) Em Number of GPUs, especifique o número de GPUs que a tarefa vai usar. A tarefa será executada em um contêiner com o número especificado de GPUs fixadas a esse contêiner.

i. (Opcional) Para Job role, é possível especificar uma função do perfil do IAM que fornece o contêiner no trabalho com permissões para usar as APIs da AWS. Esse atributo usa perfis

do IAM Amazon ECS para funcionalidade de tarefa. Para obter mais informações, consulte IAM Roles for Tasks no Guia do desenvolvedor do Amazon Elastic Container Service.

> **Note**
>
> Para trabalhos que são executados nos recursos do Fargate, é necessário um perfil de trabalho.

> **Note**
>
> Somente funções que tenham o relacionamento de confiança Função da tarefa do Amazon Elastic Container Service são mostradas aqui. Para obter mais informações sobre como criar um perfil do IAM para os trabalhos do seu AWS Batch, consulte Como criar um perfvil do IAM e uma política para suas tarefas no Guia do desenvolvedor do Amazon Elastic Container Service.

j. (Opcional) Em Execution role, especifique um perfil do IAM que conceda aos agentes de contêiner do Amazon ECS permissão para fazer chamadas de API AWS em seu nome. Esse atributo usa perfis do IAM Amazon ECS para funcionalidade de tarefa. Para obter mais informações, consulte Perfis do IAM para execução de tarefa do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

19. (Opcional) Expanda Configuração adicional.

    a. Em Environment variables, escolha Add environment variable para adicionar variáveis de ambiente como pares de nome e valor. Essas variáveis serão passadas para o contêiner.

    b. Para Job role, é possível especificar uma função do perfil do IAM que fornece o contêiner no seu trabalho com permissões para usar as APIs da AWS. Esse atributo usa perfis do IAM Amazon ECS para funcionalidade de tarefa. Para obter mais informações, consulte IAM Roles for Tasks no Guia do desenvolvedor do Amazon Elastic Container Service.

    > **Note**
    >
    > Para trabalhos que são executados nos recursos do Fargate, é necessário um perfil de trabalho.

> **Note**
>
> Somente funções que tenham o relacionamento de confiança Função da tarefa do Amazon Elastic Container Service são mostradas aqui. Para obter mais informações sobre como criar um perfil do IAM para trabalhos AWS Batch, consulte Como criar uma política e um perfil do IAM para suas tarefas no Amazon Elastic Container Service Developer Guide.

c. Para Execution role, especifique um perfil do IAM que conceda aos agentes de contêiner do Amazon ECS permissão para fazer chamadas de API AWS em seu nome. Esse atributo usa perfis do IAM Amazon ECS para funcionalidade de tarefa. Para obter mais informações, consulte Perfis do IAM para execução de tarefa do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

20. Na seção Configuração de segurança:

    a. (Opcional) Para oferecer ao contêiner da tarefa privilégios elevados na instância do host (semelhante ao usuário `root`), selecione Privileged. Esse parâmetro é mapeado para `Privileged` na seção Criar um contêiner da API remota do Docker e a opção `--privileged` para docker run.

    b. (Opcional) Para Usuário, insira o nome do usuário a ser usado dentro do contêiner. Esse parâmetro é mapeado para `User` na seção Criar um contêiner da API remota do Docker e a opção `--user` para docker run.

    c. (Opcional) Em Segredos, escolha Adicionar segredo para adicionar segredos como pares de nome-valor. Esses segredos são expostos no contêiner. Para obter mais informações, consulte SecretOptions em Parâmetros de definição de trabalho para ContainerProperties.

21. Na seção de Configuração do Linux:

    a. Ative Habilitar sistema de arquivos somente para leitura para remover o acesso de gravação ao volume.

    b. (Opcional) Ative o processo init para executar um processo `init` dentro do contêiner. Este processo encaminha sinais e colhe processos.

    c. Para Tamanho da memória compartilhada, insira o tamanho (em MiB) do volume `/dev/shm`.

d. Para Tamanho máximo de troca, insira a quantidade total de memória de troca (em MiB) que o contêiner pode usar.

e. Em Swappiness, insira um valor entre 0 e 100 para indicar o comportamento de troca de memória do contêiner. Se você não especificar um valor e a troca estiver ativada, o valor padrão será 60. Para obter mais informações, consulte swappiness em Parâmetros de definição de trabalho para ContainerProperties.

f. (Opcional) Em Dispositivos, escolha Adicionar dispositivo para adicionar um dispositivo:

   i. Em Container path (Caminho do contêiner), especifique o caminho na instância de contêiner para expor o dispositivo mapeado para a instância de host. Se isso for deixado em branco, o caminho de host será usado no contêiner.

   ii. Em Host path, especifique o caminho de um dispositivo na instância de host.

   iii. Em Permissions, selecione uma ou mais permissões para serem aplicadas ao dispositivo no contêiner. As permissões disponíveis são LER, GRAVAR e MKNOD.

22. (Opcional) Em Pontos de montagem, escolha Adicionar configuração de pontos de montagem para adicionar pontos de montagem para volumes de dados. Você deve especificar o volume de origem e o caminho do contêiner. Esses pontos de montagem são passados para o Docker daemon em uma instância de contêiner. Você também pode optar por tornar o volume somente para leitura.

23. (Opcional) Para Configuração de ulimits, escolha Adicionar ulimit para adicionar um `ulimits` valor ao contêiner. Insira os valores de Nome, Limite flexível, e Limite rígido e, em seguida, escolha Adicionar ulimit.

24. (Opcional) Em Configuração de volumes, escolha Add volume para criar uma lista de volumes a serem passados para o contêiner. Insira o nome e o caminho de origem para o volume e escolha Add volume. Você também pode optar por ativar o Habilitar EFS.

25. (Opcional) Para Tmpfs, escolha Adicionar tmpfs para adicionar uma montagem `tmpfs`.

26. (Opcional) Na seção Configuração de registro:

    a. Em Driver de log, escolha o driver de log a ser usado. Para obter mais informações sobre os drivers de log disponíveis, consulte LogDriver emParâmetros de definição de trabalho para ContainerProperties.

    > **Note**
    >
    > Por padrão, o driver de log `awslogs` é usado.

b. Em Opções, escolha Adicionar opção para adicionar uma opção. Insira um par nome-valor e então escolha Adicionar opção.

c. Em Segredos, escolha Adicionar segredo. Insira um par nome-valor e escolha Adicionar segredo para adicionar um segredo.

> **Tip**
>
> Para obter mais informações, consulte SecretOptions em Parâmetros de definição de trabalho para ContainerProperties.

27. Escolha Próxima página.
28. Para a Revisão de Definição de Tarefa, revise as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Criar Definição de Trabalho.

Criação de definições de trabalho usando ContainerProperties

Veja a seguir um modelo de definição de tarefa vazio que inclui um único contêiner. Você pode usar esse modelo para criar sua definição de trabalho, que pode ser salva em um arquivo e usada com a AWS CLI `--cli-input-json` opção. Para mais informações sobre esses parâmetros, consulte Parâmetros de definição de trabalho para ContainerProperties.

```
{
    "jobDefinitionName": "",
    "type": "container",
    "parameters": {
        "KeyName": ""
    },
    "schedulingPriority": 0,
    "containerProperties": {
        "image": "",
        "vcpus": 0,
        "memory": 0,
        "command": [
            ""
        ],
        "jobRoleArn": "",
        "executionRoleArn": "",
        "volumes": [
            {
```

```
                "host": {
                    "sourcePath": ""
                },
                "name": "",
                "efsVolumeConfiguration": {
                    "fileSystemId": "",
                    "rootDirectory": "",
                    "transitEncryption": "ENABLED",
                    "transitEncryptionPort": 0,
                    "authorizationConfig": {
                        "accessPointId": "",
                        "iam": "DISABLED"
                    }
                }
            }
        ],
        "environment": [
            {
                "name": "",
                "value": ""
            }
        ],
        "mountPoints": [
            {
                "containerPath": "",
                "readOnly": true,
                "sourceVolume": ""
            }
        ],
        "readonlyRootFilesystem": true,
        "privileged": true,
        "ulimits": [
            {
                "hardLimit": 0,
                "name": "",
                "softLimit": 0
            }
        ],
        "user": "",
        "instanceType": "",
        "resourceRequirements": [
            {
                "value": "",
                "type": "MEMORY"
```

```
            }
        ],
        "linuxParameters": {
            "devices": [
                {
                    "hostPath": "",
                    "containerPath": "",
                    "permissions": [
                        "WRITE"
                    ]
                }
            ],
            "initProcessEnabled": true,
            "sharedMemorySize": 0,
            "tmpfs": [
                {
                    "containerPath": "",
                    "size": 0,
                    "mountOptions": [
                        ""
                    ]
                }
            ],
            "maxSwap": 0,
            "swappiness": 0
        },
        "logConfiguration": {
            "logDriver": "syslog",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ]
        },
        "secrets": [
            {
                "name": "",
                "valueFrom": ""
            }
        ],
```

```
        "networkConfiguration": {
            "assignPublicIp": "DISABLED"
        },
        "fargatePlatformConfiguration": {
            "platformVersion": ""
        }
    },
    "nodeProperties": {
        "numNodes": 0,
        "mainNode": 0,
        "nodeRangeProperties": [
            {
                "targetNodes": "",
                "container": {
                    "image": "",
                    "vcpus": 0,
                    "memory": 0,
                    "command": [
                        ""
                    ],
                    "jobRoleArn": "",
                    "executionRoleArn": "",
                    "volumes": [
                        {
                            "host": {
                                "sourcePath": ""
                            },
                            "name": "",
                            "efsVolumeConfiguration": {
                                "fileSystemId": "",
                                "rootDirectory": "",
                                "transitEncryption": "DISABLED",
                                "transitEncryptionPort": 0,
                                "authorizationConfig": {
                                    "accessPointId": "",
                                    "iam": "ENABLED"
                                }
                            }
                        }
                    ],
                    "environment": [
                        {
                            "name": "",
                            "value": ""
```

```
                    }
                ],
                "mountPoints": [
                    {
                        "containerPath": "",
                        "readOnly": true,
                        "sourceVolume": ""
                    }
                ],
                "readonlyRootFilesystem": true,
                "privileged": true,
                "ulimits": [
                    {
                        "hardLimit": 0,
                        "name": "",
                        "softLimit": 0
                    }
                ],
                "user": "",
                "instanceType": "",
                "resourceRequirements": [
                    {
                        "value": "",
                        "type": "MEMORY"
                    }
                ],
                "linuxParameters": {
                    "devices": [
                        {
                            "hostPath": "",
                            "containerPath": "",
                            "permissions": [
                                "WRITE"
                            ]
                        }
                    ],
                    "initProcessEnabled": true,
                    "sharedMemorySize": 0,
                    "tmpfs": [
                        {
                            "containerPath": "",
                            "size": 0,
                            "mountOptions": [
                                ""
```

```
                            ]
                        }
                    ],
                    "maxSwap": 0,
                    "swappiness": 0
                },
                "logConfiguration": {
                    "logDriver": "awslogs",
                    "options": {
                        "KeyName": ""
                    },
                    "secretOptions": [
                        {
                            "name": "",
                            "valueFrom": ""
                        }
                    ]
                },
                "secrets": [
                    {
                        "name": "",
                        "valueFrom": ""
                    }
                ],
                "networkConfiguration": {
                    "assignPublicIp": "DISABLED"
                },
                "fargatePlatformConfiguration": {
                    "platformVersion": ""
                }
            }
        }
    ]
},
"retryStrategy": {
    "attempts": 0,
    "evaluateOnExit": [
        {
            "onStatusReason": "",
            "onReason": "",
            "onExitCode": "",
            "action": "RETRY"
        }
    ]
```

```
    },
    "propagateTags": true,
    "timeout": {
        "attemptDurationSeconds": 0
    },
    "tags": {
        "KeyName": ""
    },
    "platformCapabilities": [
        "EC2"
    ],
    "eksProperties": {
        "podProperties": {
            "serviceAccountName": "",
            "hostNetwork": true,
            "dnsPolicy": "",
            "containers": [
                {
                    "name": "",
                    "image": "",
                    "imagePullPolicy": "",
                    "command": [
                        ""
                    ],
                    "args": [
                        ""
                    ],
                    "env": [
                        {
                            "name": "",
                            "value": ""
                        }
                    ],
                    "resources": {
                        "limits": {
                            "KeyName": ""
                        },
                        "requests": {
                            "KeyName": ""
                        }
                    },
                    "volumeMounts": [
                        {
                            "name": "",
```

```
                        "mountPath": "",
                        "readOnly": true
                    }
                ],
                "securityContext": {
                    "runAsUser": 0,
                    "runAsGroup": 0,
                    "privileged": true,
                    "readOnlyRootFilesystem": true,
                    "runAsNonRoot": true
                }
            }
        ],
        "volumes": [
            {
                "name": "",
                "hostPath": {
                    "path": ""
                },
                "emptyDir": {
                    "medium": "",
                    "sizeLimit": ""
                },
                "secret": {
                    "secretName": "",
                    "optional": true
                }
            }
        ]
    }
  }
}
```

Note

Você pode gerar um modelo de definição de tarefa de contêiner único com o seguinte comando: AWS CLI

```
$ aws batch register-job-definition --generate-cli-skeleton
```

Parâmetros de definição de trabalho para ContainerProperties

As definições de trabalho que eu uso `ContainerProperties`são divididas em várias partes:

- o nome da definição do trabalho
- o tipo de definição do trabalho
- os padrões do espaço reservado para substituição de parâmetros
- o contêiner para as propriedades do trabalho
- as propriedades do Amazon EKS para a definição do trabalho que são necessárias para trabalhos executados em recursos do Amazon EKS
- as propriedades do nó que são necessárias para um trabalho paralelo de vários nós
- os requisitos da plataforma necessários para trabalhos executados em recursos Fargate
- os detalhes padrão de propagação de tag da definição do trabalho
- a estratégia de repetição padrão para a definição do trabalho
- a prioridade de agendamento padrão para a definição do trabalho
- as tags padrão para a definição do trabalho
- o tempo limite padrão para a definição do trabalho

Sumário

- Timeout (Tempo limite)

Nome da definição do trabalho

`jobDefinitionName`

Ao registrar uma definição de trabalho, você especifica um nome. O nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_). A primeira definição de trabalho que está registrada com esse nome recebe uma revisão de 1. Todas as definições de trabalho subsequentes registrados com esse nome recebem um número de revisão incremental.

Tipo: Sequência

Obrigatório: Sim

Tipo

`type`

Ao registrar uma definição de trabalho, você especifica o tipo de trabalho. Se o trabalho for executado em recursos do Fargate, então `multinode` não será suportado. Para obter mais informações sobre trabalho em paralelo com vários nós, consulte the section called “Criar uma definição de tarefa em paralelo de vários nós”.

Tipo: Sequência

Valores válidos: `container` | `multinode`

Obrigatório: Sim

Parâmetros

`parameters`

Quando você envia um trabalho, pode especificar os parâmetros que devem substituir os marcadores ou os parâmetros de definição de trabalho padrão. Os parâmetros nas solicitações de envio de trabalho têm precedência sobre os padrões em uma definição de trabalho. Isso significa que você pode usar a mesma definição de trabalho para vários outros que usam o

mesmo formato. Você também pode alterar programaticamente os valores no comando no momento do envio.

Tipo: Mapa de string para string

Obrigatório: Não

Ao registrar uma definição de trabaho, você pode usar espaços reservados para substituição de parâmetros no campo `command` de propriedades de contêiner de um trabalho. A sintaxe é a seguinte.

```
"command": [
    "ffmpeg",
    "-i",
    "Ref::inputfile",
    "-c",
    "Ref::codec",
    "-o",
    "Ref::outputfile"
]
```

No exemplo acima, há os espaços reservados de substituição de parâmetro *`Ref::inputfile`*, *`Ref::codec`* e *`Ref::outputfile`* no comando. Você pode usar o objeto `parameters` na definição do trabalho para configurar valores padrão para esses espaços reservados. Por exemplo, para definir um padrão para o espaço reservado *`Ref::codec`*, você especificará o seguinte na definição do trabalho:

```
"parameters" : {"codec" : "mp4"}
```

Quando essa definição de trabalho for enviada para execução, o argumento *`Ref::codec`* no comando para o contêiner será substituído pelo valor padrão, `mp4`.

Propriedades do contêiner

Ao registrar uma definição de trabalho você deve especificar uma lista de propriedades de contêiner que são passadas para o daemon do Docker em uma instância de contêiner quando o trabalho é feito. As seguintes propriedades de contêiner são permitidas em uma definição de trabalho. Para trabalhos de nó único, essas propriedades de contêiner são configuradas no nível de definição do trabalho Para trabalhos em paralelo de vários nós, as propriedades de contêiner são definidas no nível de Propriedades de nó para cada grupo de nós.

`command`

O comando que é passado para o contêiner. Esse parâmetro é mapeado para `Cmd` na seção Criar um contêiner da Docker Remote API e o parâmetro `COMMAND` de docker run. Para obter mais informações sobre o parâmetro `CMD` da Docker, consulte https://docs.docker.com/engine/reference/builder/#cmd.

```
"command": ["string", ...]
```

Tipo: Matriz de strings

Obrigatório: Não

`environment`

As variáveis de ambiente a serem passadas para um contêiner. Esse parâmetro é mapeado para `Env` na seção Criar um contêiner da Docker Remote API e a opção `--env` para docker run.

> **Important**
>
> Não recomendamos que você use variáveis de ambiente de texto simples para informações confidenciais, como dados de credenciais.

> **Note**
>
> Variáveis de ambiente não podem começar com `AWS_BATCH`. Essa convenção de nomenclatura é reservada para variáveis definidas pelo AWS Batch serviço.

Tipo: Matriz de pares de valores-chave

Obrigatório: Não

`name`

O nome da variável de ambiente.

Tipo: Sequência

Obrigatório: Sim, quando `environment` for usado.

`value`

O valor da variável de ambiente.

Tipo: Sequência

Obrigatório: Sim, quando `environment` for usado.

```
"environment" : [
    { "name" : "envName1", "value" : "envValue1" },
    { "name" : "envName2", "value" : "envValue2" }
]
```

`executionRoleArn`

Ao registrar uma definição de trabalho, você pode especificar um perfil do IAM. O perfil fornece ao agente de contêiner do Amazon ECS permissões para chamar as ações da API especificadas nas políticas colaboradoras em seu nome. Para trabalhos executados em recursos do Fargate, você deve fornecer um perfil de execução. Para ter mais informações, consulte AWS Batch função de execução do IAM.

Tipo: Sequência

Obrigatório: Não

`fargatePlatformConfiguration`

A configuração da plataforma para trabalhos que são executados nos recursos do Fargate. Os trabalhos que são executados nos recursos do EC2 não devem especificar esse parâmetro.

Tipo: objeto FargatePlatformConfiguration

Obrigatório: Não

`platformVersion`

Use a versão da plataforma AWS Fargate para os trabalhos ou `LATEST` para usar uma versão recente e aprovada da plataforma Fargate AWS .

Tipo: sequência

Padrão: `LATEST`

Obrigatório: Não

`image`

A imagem usada para iniciar um trabalho. Esta string é passada diretamente para o daemon do Docker. As imagens no registro do Docker Hub estão disponíveis por padrão. Você também pode especificar outros repositórios com *`repository-url`*`/`*`image`*`:`*`tag`*. São permitidos até 255 letras (caixa alta e baixa), números, hifens, sublinhados, dois pontos, ponto, barras e sinais numéricos. Esse parâmetro é mapeado para `Image` na seção Criar um contêiner da Docker Remote API e o parâmetro `IMAGE` de docker run.

> **Note**
>
> Docker arquitetura da imagem do Docker deve corresponder à arquitetura do processador dos recursos de computação nos quais eles foram programados. Por exemplo, imagens Arm baseadas Docker em só podem ser executadas em recursos de computação baseados em Arm.

- As imagens em repositórios públicos do Amazon ECR usam as convenções de nomenclatura completa`registry/repository[:tag]` ou `registry/repository[@digest]` (por exemplo, `public.ecr.aws/`*`registry_alias`*`/`*`my-web-app`*`:`*`latest`*).
- As imagens nos repositórios do Amazon ECR usam a convenção de `registry/repository:[tag]` nomenclatura completa. Por exemplo, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*.
- As imagens em repositórios oficiais no Docker Hub usam um único nome (por exemplo, `ubuntu` ou `mongo`).
- As imagens em outros repositórios no Docker Hub são qualificadas com um nome de organização (por exemplo, `amazon/amazon-ecs-agent`).
- Imagens em outros repositórios online também são qualificadas por um nome de domínio (por exemplo, `quay.io/assemblyline/ubuntu`).

Tipo: Sequência

Obrigatório: Sim

`instanceType`

O tipo de instância a ser usado para trabalhos paralelos de vários nós. Atualmente, todos os grupos de nós em um trabalho em paralelo de vários nós devem usar o mesmo tipo de instância.

Esse parâmetro não é válido para trabalhos de contêiner de nó único ou para trabalhos que são executados em recursos Fargate.

Tipo: Sequência

Obrigatório: Não

`jobRoleArn`

Ao registrar uma definição de trabalho, você pode especificar um perfil do IAM. A função fornece ao contêiner de trabalho permissões para chamar as ações da API especificadas em suas políticas colaboradoras em seu nome. Para obter mais informações, consulte Perfis do IAM para tarefas no Guia do desenvolvedor do Amazon Elastic Container Service.

Tipo: Sequência

Obrigatório: Não

`linuxParameters`

Modificações específicas do Linux que são aplicadas ao contêiner, como detalhes para mapeamentos do dispositivo.

```
"linuxParameters": {
    "devices": [
        {
            "hostPath": "string",
            "containerPath": "string",
            "permissions": [
                "READ", "WRITE", "MKNOD"
            ]
        }
    ],
    "initProcessEnabled": true|false,
    "sharedMemorySize": 0,
    "tmpfs": [
        {
            "containerPath": "string",
            "size": integer,
            "mountOptions": [
                "string"
            ]
        }
    ],
```

```
    "maxSwap": integer,
    "swappiness": integer
}
```

Tipo: objeto LinuxParameters

Obrigatório: Não

`devices`

Lista de dispositivos mapeados para o contêiner. Esse parâmetro é mapeado `Devices` na seção Criar um contêiner da Docker Remote API e na opção `--device` de execução do docker.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Tipo: Matriz de objetos Device

Obrigatório: Não

`hostPath`

Caminho onde está o dispositivo disponível na instância de contêiner host.

Tipo: Sequência

Obrigatório: Sim

`containerPath`

Caminho em que o dispositivo é exposto no contêiner. Se isso não for especificado, o dispositivo será exposto ao mesmo caminho do host.

Tipo: Sequência

Obrigatório: Não

`permissions`

Permissões para o dispositivo no contêiner. Se não for especificado, as permissões serão definidas como `READ`, `WRITE` e `MKNOD`.

Tipo: Matriz de strings

Obrigatório: Não

Valores válidos: `READ` | `WRITE` | `MKNOD`

`initProcessEnabled`

Se verdadeiro, execute um processo `init` dentro do contêiner que encaminha sinais e colhe processos. Esse parâmetro mapeia para a opção `--init` para execução do docker. Este parâmetro requer a versão 1.25 ou posterior da Docker Remote API em sua instância de contêiner. Para verificar a versão da Docker Remote API na instância de contêiner, faça login na instância de contêiner e execute o seguinte comando: `sudo docker version | grep "Server API version"`

Tipo: Booliano

Obrigatório: Não

`maxSwap`

Especifica a quantidade total de memória de troca (em MiB) que um trabalho pode usar. Esse parâmetro será convertido na opção `--memory-swap` para execução do docker em que o valor é a soma da memória do contêiner mais o valor de `maxSwap`. Para obter mais informações, consulte Detalhes do `--memory-swap` na documentação do Docker.

Se um valor `maxSwap` de `0` for especificado, o contêiner não usará a troca. Os valores aceitos são `0` ou qualquer número inteiro positivo. Se o parâmetro `maxSwap` for omitido, o contêiner usará a configuração de troca para a instância de contêiner na qual ele está sendo executado. Um valor `maxSwap` deve ser definido para que o parâmetro `swappiness` seja usado.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Tipo: Inteiro

Obrigatório: Não

`sharedMemorySize`

O valor do tamanho (em MiB) do volume `/dev/shm`. Esse parâmetro mapeia para a opção `--shm-size` para execução do docker.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Tipo: Inteiro

Obrigatório: Não

`swappiness`

É possível usar isso para ajustar o comportamento de troca de memória de um contêiner. Um valor `swappiness` de `0` fará com que a troca não ocorra, a menos que seja absolutamente necessário. Um valor `swappiness` de `100` fará com que as páginas sejam trocadas de forma agressiva. Os valores aceitos são números inteiros entre `0` e `100`. Se o parâmetro `swappiness` não for especificado, será usado um valor padrão de `60`. Se nenhum valor for especificado para `maxSwap`, esse parâmetro será ignorado. Se `maxSwap` estiver definido como 0, o contêiner não usará troca. Esse parâmetro mapeia para a opção `--memory-swappiness` para execução do docker.

Considere o seguinte ao usar uma configuração de troca de contêiner.

- O espaço de troca deve ser habilitado e alocado na instância de contêiner para os contêineres usarem.

  > **Note**
  >
  > As AMIs otimizadas do Amazon ECS não têm a troca habilitada por padrão. É necessário habilitar a troca na instância para usar esse recurso. Para obter mais informações, consulte Volumes de troca de armazenamento de instâncias no Guia do usuário do Amazon EC2 para instâncias do Linux ou Como alocar memória para funcionar como espaço de troca em uma instância do Amazon EC2 usando um arquivo de troca?.

- Os parâmetros de espaço de troca só são suportados para definições de trabalho usando recursos do EC2.
- Se os parâmetros `maxSwap` e `swappiness` forem omitidos de uma definição de trabalho, cada contêiner terá um valor de `swappiness` padrão de 60. Além disso, o uso total de troca tem um limite de duas vezes a reserva de memória do contêiner.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Tipo: Inteiro

Obrigatório: Não

`tmpfs`

O caminho do contêiner, as opções de montagem e o tamanho da montagem do tmpfs.

Tipo: Matriz de objetos Tmpfs

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Obrigatório: Não

`containerPath`

O caminho absoluto do arquivo no contêiner onde o volume tmpfs é montado.

Tipo: Sequência

Obrigatório: Sim

`mountOptions`

A lista de opções de montagem do volume tmpfs.

Valores válidos: `"defaults" | "ro" | "rw" | "suid" | "nosuid" | "dev" | "nodev" | "exec" | "noexec" | "sync" | "async" | "dirsync" | "remount" | "mand" | "nomand" | "atime" | "noatime" | "diratime" | "nodiratime" | "bind" | "rbind" | "unbindable" | "runbindable" | "private" | "rprivate" | "shared" | "rshared" | "slave" | "rslave" | "relatime" | "norelatime" | "strictatime" | "nostrictatime" | "mode" | "uid" | "gid" | "nr_inodes" | "nr_blocks" | "mpol"`

Tipo: Matriz de strings

Obrigatório: Não

`size`

O tamanho (em MiB) do volume tmpfs.

Tipo: Inteiro

Obrigatório: Sim

`logConfiguration`

A especificação de configuração do log para o contêiner.

Esse parâmetro é mapeado `LogConfig` na seção Criar um contêiner da Docker Remote API e na opção `--log-driver` de execução do docker. Por padrão, os contêineres usam o mesmo driver de registro que o daemon do Docker usa. No entanto, o contêiner pode usar um driver de registro diferente do daemon do Docker especificando um driver de log com esse parâmetro na definição de contêiner. Para usar um driver de registro diferente para um contêiner, o sistema de registro deve ser configurado na instância de contêiner ou em outro servidor de registro para fornecer opções de registro remoto. Para obter mais informações sobre as opções para drivers de log compatíveis diferentes, consulte Configure logging drivers na documentação do Docker.

> **Note**
>
> AWS Batch atualmente oferece suporte a um subconjunto dos drivers de registro disponíveis para o daemon do Docker (mostrado no tipo de dados). LogConfiguration

Este parâmetro requer a versão 1.18 da Docker Remote API ou posterior em sua instância de contêiner. Para verificar a versão da Docker Remote API na instância de contêiner, faça login na instância de contêiner e execute o seguinte comando: `sudo docker version | grep "Server API version"`

```
"logConfiguration": {
    "devices": [
        {
            "logDriver": "string",
            "options": {
                "optionName1" : "optionValue1",
                "optionName2" : "optionValue2"
```

```
            }
            "secretOptions": [
              {
                  "name" : "secretOptionName1",
                  "valueFrom" : "secretOptionArn1"
              },
              {
                  "name" : "secretOptionName2",
                  "valueFrom" : "secretOptionArn2"
              }
            ]
        }
    ]
}
```

Tipo: objeto LogConfiguration

Obrigatório: Não

`logDriver`

O driver de log a ser usado para o contêiner. Por padrão, AWS Batch ativa o driver de `awslogs` log. Os valores válidos listados para esse parâmetro são drivers de log com os quais o agente de contêiner do Amazon ECS pode se comunicar por padrão.

Esse parâmetro é mapeado `LogConfig` na seção Criar um contêiner da Docker Remote API e na opção `--log-driver` de execução do docker. Por padrão, os trabalhos usam o mesmo driver de registro que o daemon do Docker usa. No entanto, o trabalho pode usar um driver de registro diferente do daemon do Docker especificando um driver de log com esse parâmetro na definição do trabalho. Se você quiser especificar outro driver de registro em log para um trabalho, o sistema de log deve ser configurado na instância de contêiner no ambiente de computação. Ou, como alternativa, configure-o em outro servidor de log para fornecer opções de registro remoto. Para obter mais informações sobre as opções para drivers de log compatíveis diferentes, consulte Configure logging drivers na documentação do Docker.

> **Note**
>
> AWS Batch atualmente oferece suporte a um subconjunto dos drivers de registro que estão disponíveis para o daemon do Docker. Os drivers de log adicionais podem estar disponíveis em versões futuras do agente de contêiner do Amazon ECS.

Os drivers de log compatíveis são `awslogs`, `fluentd`, `gelf`, `json-file`, `journald`, `logentries`, `syslog` e `splunk`.

> **Note**
>
> Os trabalhos em execução nos recursos do Fargate são restritos aos drivers de log `awslogs` e `splunk`.

Este parâmetro requer a versão 1.18 da Docker Remote API ou posterior em sua instância de contêiner. Para verificar a versão da Docker Remote API na instância de contêiner, faça login na instância de contêiner e execute o seguinte comando: `sudo docker version | grep "Server API version"`

> **Note**
>
> O agente de contêiner do Amazon ECS que é executado em uma instância de contêiner deve registrar os drivers de registro que estão disponíveis nessa instância com o variável de ambiente `ECS_AVAILABLE_LOGGING_DRIVERS`. Caso contrário, os contêineres colocados nessa instância não poderão usar essas opções de configuração de log. Para obter mais informações, consulte Configuração do Agente de Contêineres do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

`awslogs`

Especifica o driver de registro do Amazon CloudWatch Logs. Para obter mais informações, consulte Usar o driver de log awslogs o driver de registro do Amazon CloudWatch Logs na documentação do Docker.

`fluentd`

Especifica o driver de registro do Fluentd. Para obter mais informações, incluindo uso e opções, consulte Driver de registro Fluentd na documentação do Docker.

`gelf`

Especifica o driver de registro Graylog Extended Format (GELF). Para obter mais informações, incluindo uso e opções, consulte Driver de registro Graylog Extended Format na documentação do Docker.

`journald`

Especifica o driver de registro journald. Para obter mais informações, incluindo uso e opções, consulte Driver de registro Journald na documentação do Docker.

`json-file`

Especifica o driver de registro de arquivos JSON. Para obter mais informações, incluindo uso e opções, consulte driver de registro de arquivo JSON na documentação do Docker.

`splunk`

Especifica o driver de registro do Splunk. Para obter mais informações, incluindo uso e opções, consulte driver de registro do Splunk na documentação do Docker.

`syslog`

Especifica o driver de registro em log do syslog. Para obter mais informações, incluindo uso e opções, consulte driver de registro de log do Syslog na documentação do Docker.

Tipo: Sequência

Obrigatório: Sim

Valores válidos: `awslogs` | `fluentd` | `gelf` | `journald` | `json-file` | `splunk` | `syslog`

> **Note**
>
> Se você tem um driver personalizado que não está listado anteriormente e gostaria de trabalhar com o agente de contêiner do Amazon ECS, você pode bifurcar o projeto do agente de contêiner do Amazon ECS que está disponível GitHub e personalizá-lo para funcionar com esse driver. Incentivamos você a enviar solicitações pull para alterações que você gostaria de ter incluído. No entanto, o Amazon Web Services atualmente não oferece suporte para solicitações a execução de cópias modificadas desse software.

`options`

Opções de configuração de log para enviar para um driver de log personalizado para o contêiner.

Este parâmetro requer que a versão 1.19 da Docker Remote API ou posterior em sua instância de contêiner.

Tipo: Mapa de string para string

Obrigatório: Não

`secretOptions`

Um objeto que representa o segredo a ser passado para a configuração de log. Para ter mais informações, consulte Especificando dados sigilosos.

Tipo: Matriz de objeto

Obrigatório: Não

`name`

O nome da opção do driver de log a ser definida no trabalho.

Tipo: Sequência

Obrigatório: Sim

`valueFrom`

O Nome de recurso da Amazon (ARN) do segredo a ser exposto à configuração de log do contêiner. Os valores suportados são o ARN completo do segredo do Secrets Manager ou o ARN completo do parâmetro na SSM Parameter Store.

> **Note**
>
> Se o parâmetro SSM Parameter Store existir na Região da AWS mesma tarefa que você está iniciando, você poderá usar o ARN completo ou o nome do parâmetro. Se o parâmetro existir em uma Região diferente, o ARN completo deverá ser especificado.

Tipo: Sequência

Obrigatório: Sim

`memory`

Este parâmetro está obsoleto, use *`resourceRequirements`* alternativamente.

O número de MiB de memória a ser reservado para o contêiner.

Como exemplo de como usar `resourceRequirements`, se sua definição de trabalho contiver uma sintaxe semelhante à seguinte.

```
"containerProperties": {
  "memory": 512
}
```

A sintaxe equivalente usando `resourceRequirements` é a seguinte.

```
"containerProperties": {
  "resourceRequirements": [
    {
      "type": "MEMORY",
      "value": "512"
    }
  ]
}
```

Tipo: Inteiro

Obrigatório: Sim

`mountPoints`

Os pontos de montagem para volumes de dados no contêiner. Esse parâmetro é mapeado para `Volumes` na seção Criar um contêiner da Docker Remote API e a opção `--volume` para docker run.

```
"mountPoints": [
                {
                  "sourceVolume": "string",
                  "containerPath": "string",
                  "readOnly": true|false
                }
              ]
```

Tipo: Matriz de objeto

Obrigatório: Não

`sourceVolume`

O nome do volume a ser montado.

Tipo: Sequência

Obrigatório: Sim, quando `mountPoints` for usado.

`containerPath`

O caminho no contêiner no qual montar o volume de host.

Tipo: Sequência

Obrigatório: Sim, quando `mountPoints` for usado.

`readOnly`

Caso o valor seja `true`, o contêiner tem acesso somente leitura ao volume. Caso esse valor seja `false`, o contêiner pode gravar no volume.

Tipo: Booliano

Obrigatório: não

Padrão: Falso

`networkConfiguration`

A configuração de rede para trabalhos que estão em execução nos recursos do Fargate. Os trabalhos que são executados nos recursos do EC2 não devem especificar esse parâmetro.

```
"networkConfiguration": {
   "assignPublicIp": "string"
}
```

Tipo: Matriz de objeto

Obrigatório: Não

`assignPublicIp`

Indica se o trabalho tem um endereço IP público. Isso é necessário se o trabalho precisar de acesso externo à rede.

Tipo: Sequência

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: não

Padrão: `DISABLED`

`privileged`

Quando esse parâmetro é verdadeiro, o contêiner recebe permissões elevadas na instância de contêiner host (semelhante ao usuário `root`). Esse parâmetro é mapeado para `Privileged` na seção Criar um contêiner da Docker Remote API e a opção `--privileged` para docker run. Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate. Não o forneça nem o especifique como falso.

```
"privileged": true|false
```

Tipo: Booliano

Obrigatório: Não

`readonlyRootFilesystem`

Quando esse parâmetro é verdadeiro, o contêiner recebe acesso somente leitura ao sistema de arquivos raiz. Esse parâmetro é mapeado para `ReadonlyRootfs` na seção Criar um contêiner da Docker Remote API e a opção `--read-only` para docker run.

```
"readonlyRootFilesystem": true|false
```

Tipo: Booliano

Obrigatório: Não

`resourceRequirements`

O tipo e a quantidade de um recurso a serem atribuídos a um contêiner. Os recursos suportados incluem `GPU`, `MEMORY` e `VCPU`.

```
"resourceRequirements" : [
  {
    "type": "GPU",
    "value": "number"
  }
```

```
]
```

Tipo: Matriz de objeto

Obrigatório: Não

`type`

O tipo de recurso a ser atribuído a um contêiner. Os recursos suportados incluem `GPU`, `MEMORY` e `VCPU`.

Tipo: Sequência

Obrigatório: Sim, quando `resourceRequirements` for usado.

`value`

A quantidade do recurso especificado a ser reservado para o contêiner. Os valores variam de acordo com o `type` especificado.

Tipo="GPU"

O número de GPUs físicas a serem reservadas para o contêiner. O número de GPUs reservadas para todos os contêineres em um trabalho não deve exceder o número de GPUs disponíveis no recurso de computação no qual o trabalho é executado

Tipo="MEMORY"

O limite rígido (em MiB) de memória a ser apresentado ao contêiner. Caso tente exceder a memória especificada aqui, o contêiner será excluído. Esse parâmetro é mapeado `Memory` na seção Criar um contêiner da Docker Remote API e na opção `--memory` de execução do docker. Você deve especificar pelo menos 4 MiB de memória para uma tarefa. Isso é necessário, mas pode ser especificado em vários lugares para trabalhos paralelos de vários nós (MNP). Ele deve ser especificado para cada nó pelo menos uma vez. Esse parâmetro é mapeado `Memory` na seção Criar um contêiner da Docker Remote API e na opção `--memory` de execução do docker.

> **Note**
>
> Se você deseja maximizar a utilização de recursos fornecendo aos trabalhos o máximo de memória possível para um determinado tipo de instância, consulte Recurso de Computação Gerenciamento de Memória.

Para trabalhos executados em recursos do Fargate, `value` deve corresponder a um dos valores suportados. Além disso, os valores `VCPU` devem ser um dos valores compatíveis com esse valor de memória.

VCPU	MEMORY
0.25 vCPU	512, 1024 e 2048 MiB
0.5 vCPU	1024-4096 MiB em incrementos de 1024 MiB
1 vCPU	2048-8192 MiB em incrementos de 1024 MiB
2 vCPU	4096-16384 MiB em incrementos de 1024 MiB
4 vCPU	8192-30720 MiB em incrementos de 1024 MiB
8 vCPU	16384-61440 MiB em incrementos de 4096 MiB
16 vCPU	32768-122880 MiB em incrementos de 8192 MiB

type="VCPU"

O número de vCPUs reservado para o trabalho. Esse parâmetro é mapeado para `CpuShares` na seção Criar um contêiner da Docker Remote API e a opção `--cpu-shares` para execução do docker. Cada vCPU equivale a 1.024 compartilhamentos de CPU. Para recursos do EC2, você deve especificar pelo menos uma vCPU. Isso é necessário, mas pode ser especificado em vários lugares. Ele deve ser especificado para cada nó pelo menos uma vez.

Para trabalhos que são executados em recursos do Fargate, `value` deve corresponder a um dos valores suportados, e os valores `MEMORY` devem ser um dos valores suportados para esse valor de VCPU. Os valores compatíveis são 0,25, 0,5, 1, 2, 4, 8 e 16

O padrão para a cota de contagem de recursos de vCPU sob demanda do Fargate é 6 vCPUs. Para obter mais informações sobre cotas de serviço, consulte AWS quotas na Referência geral da Amazon Web Services.

Tipo: Sequência

Obrigatório: Sim, quando `resourceRequirements` for usado.

`secrets`

Os segredos do trabalho que são expostos como variáveis de ambiente. Para ter mais informações, consulte Especificando dados sigilosos.

```
"secrets": [
    {
      "name": "secretName1",
      "valueFrom": "secretArn1"
    },
    {
      "name": "secretName2",
      "valueFrom": "secretArn2"
    }
    ...
]
```

Tipo: Matriz de objeto

Obrigatório: Não

`name`

O nome da variável de ambiente que contém o segredo.

Tipo: Sequência

Obrigatório: Sim, quando `secrets` for usado.

`valueFrom`

O segredo a ser exposto ao contêiner. Os valores com suporte são o Nome de recurso da Amazon (ARN) completo do segredo do Secrets Manager ou o ARN completo do parâmetro no SSM Parameter Store.

> **Note**
>
> Se o parâmetro SSM Parameter Store existir no Região da AWS mesmo trabalho que você está iniciando, você poderá usar o ARN completo ou o nome do parâmetro. Se o parâmetro existir em uma Região diferente, o ARN completo deverá ser especificado.

Tipo: Sequência

Obrigatório: Sim, quando `secrets` for usado.

`ulimits`

Uma lista de valores `ulimits` a ser definida no contêiner. Esse parâmetro é mapeado para `Ulimits` na seção Criar um contêiner da Docker Remote API e a opção `--ulimit` para docker run.

```
"ulimits": [
  {
    "name": string,
    "softLimit": integer,
    "hardLimit": integer
  }
  ...
]
```

Tipo: Matriz de objeto

Obrigatório: Não

`name`

O `type` do `ulimit`.

Tipo: Sequência

Obrigatório: Sim, quando `ulimits` for usado.

`hardLimit`

O limite rígido do tipo `ulimit`.

Tipo: Inteiro

Obrigatório: Sim, quando `ulimits` for usado.

`softLimit`

O limite flexível do tipo `ulimit`.

Tipo: Inteiro

Obrigatório: Sim, quando `ulimits` for usado.

`user`

O nome de usuário a ser usado dentro do contêiner. Esse parâmetro é mapeado para `User` na seção Criar um contêiner da Docker Remote API e a opção `--user` para docker run.

```
"user": "string"
```

Tipo: Sequência

Obrigatório: Não

`vcpus`

Este parâmetro foi suspenso, use *`resourceRequirements`* alternativamente.

O número de vCPUs reservado para o contêiner.

Como exemplo de como usar `resourceRequirements`, se sua definição de trabalho contiver linhas semelhantes a esta:

```
"containerProperties": {
  "vcpus": 2
}
```

As linhas equivalentes usadas `resourceRequirements` são as seguintes.

```
"containerProperties": {
  "resourceRequirements": [
    {
      "type": "VCPU",
      "value": "2"
    }
  ]
}
```

Tipo: Inteiro

Obrigatório: Sim

`volumes`

Ao registrar uma definição de trabalho, você pode especificar uma lista de volumes que são passados para o daemon do Docker em uma instância de contêiner. Os parâmetros a seguir são permitidos na propriedades de contêiner:

```
"volumes": [
  {
    "name": "string",
    "host": {
      "sourcePath": "string"
    },
    "efsVolumeConfiguration": {
      "authorizationConfig": {
        "accessPointId": "string",
        "iam": "string"
      },
      "fileSystemId": "string",
      "rootDirectory": "string",
      "transitEncryption": "string",
      "transitEncryptionPort": number
    }
  }
]
```

`name`

O nome do volume. São permitidos até 255 letras (caixa alta e baixa), números, hífens e sublinhados. Este nome é referenciado no parâmetro `sourceVolume` da definição de contêiner `mountPoints`.

Tipo: Sequência

Obrigatório: Não

`host`

O conteúdo do parâmetro `host` determina se o seu volume de dados persiste na instância de contêiner host e onde ele é armazenado. Se o parâmetro `host` estiver vazio, o daemon do Docker atribuirá um caminho de host para o volume de dados. No entanto, não há garantia de que os dados persistirão após os contêineres associados a eles deixarem de ser executados.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Tipo: Objeto

Obrigatório: Não

`sourcePath`

O caminho na instância de contêiner host apresentada ao contêiner. Caso esse parâmetro esteja vazio, o daemon do Docker atribui um caminho host para você.

Caso o parâmetro `host` contenha um local de arquivo `sourcePath`, o volume de dados persiste no local especificado na instância de contêiner host até você excluí-lo manualmente. Caso o valor `sourcePath` não exista na instância de contêiner do host, o daemon do Docker o criará. Caso o local exista, o conteúdo da pasta do caminho de origem é exportado.

Tipo: Sequência

Obrigatório: Não

`efsVolumeConfiguration`

Esse parâmetro é especificado quando você estiver usando um sistema de arquivos do Amazon Elastic File System para o armazenamento de tarefas. Para ter mais informações, consulte Volumes Amazon EFS.

Tipo: Objeto

Obrigatório: Não

`authorizationConfig`

Detalhes de configuração de autorização do sistema de arquivamento Amazon EFS.

Tipo: Sequência

Obrigatório: Não

`accessPointId`

O ID do ponto de acesso do Amazon EFS a ser usado. Se um ponto de acesso for especificado, o valor do diretório raiz especificado no `EFSVolumeConfiguration` deve ser omitido ou definido como. `/` Isso impõe o caminho definido no ponto de acesso EFS. Se um ponto de acesso for usado, a criptografia em trânsito deverá ser habilitada em `EFSVolumeConfiguration`. Para mais informações, consulte Trabalhando com Pontos de Acesso Amazon EFS no Guia de Usuário Amazon Elastic File System.

Tipo: Sequência

Obrigatório: não

`iam`

Determina se deve ser usada a função do IAM de AWS Batch trabalho definida em uma definição de tarefa ao montar o sistema de arquivos Amazon EFS. Em caso positivo, a criptografia de trânsito deve estar habilitada em `EFSVolumeConfiguration`. Se o parâmetro for omitido, o valor padrão `DISABLED` será usado. Para ter mais informações, consulte Usando pontos de acesso Amazon EFS.

Tipo: Sequência

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: Não

`fileSystemId`

A ID do sistema de arquivamento Amazon EFS a ser usada.

Tipo: string

Obrigatório: Não

`rootDirectory`

O diretório dentro do sistema de arquivamento Amazon EFS a ser montado como diretório raiz dentro do host. Caso esse parâmetro seja omitido, a raiz do volume Amazon EFS será usada. Especificar `/` tem o mesmo efeito que omitir esse parâmetro. O tamanho máximo é de 4.096 caracteres.

> **Important**
>
> Se um ponto de acesso do EFS for especificado no `authorizationConfig`, o parâmetro do diretório raiz deverá ser omitido ou definido como `/`. Isso impõe o caminho definido no ponto de acesso do Amazon EFS.

Tipo: Sequência

Obrigatório: Não

`transitEncryption`

Determina se a criptografia deve ou não ser habilitada para dados do Amazon EFS em trânsito entre o host do Amazon ECS e o servidor do Amazon EFS. A criptografia de trânsito deverá ser habilitada caso a autorização IAM Amazon EFS for usada. Se o parâmetro for omitido, o valor padrão `DISABLED` será usado. Para mais informações, consulte Criptografando Dados em Trânsito no Guia de Usuário Amazon Elastic File System.

Tipo: Sequência

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: Não

`transitEncryptionPort`

A porta a ser usada ao enviar dados criptografados entre o host do Amazon ECS e o servidor do Amazon EFS. Caso não especifique uma porta de criptografia em trânsito, a estratégia de seleção de porta usada pelo assistente de montagem Amazon EFS será utilizada. O valor precisar estar compreendido entre 0 e 65.535. Para mais informações, consulte Auxiliar de Montagem EFS no Guia de Usuário Amazon Elastic File System.

Tipo: Inteiro

Obrigatório: Não

Amazon EKS properties

Um objeto com diversas propriedades específicas de trabalhos baseados no Amazon EKS. Isso não pode ser especificado para definições de trabalho baseados no Amazon ECS.

`podProperties`

As propriedades dos recursos do pod Kubernetes de um trabalho.

Tipo: objeto EksPodProperties

Obrigatório: Não

`containers`

As propriedades do contêiner que é usado no pod do Amazon EKS.

Tipo: objeto EksContainer

Obrigatório: Não

`args`

Uma matriz de argumentos para o ponto de entrada. Se isso não for especificado, o `CMD` da imagem do contêiner será usado. Isso corresponde ao membro `args` na parte do Entrypoint do Pod em Kubernetes. As referências de variáveis de ambiente são expandidas usando o ambiente do contêiner.

Se a variável de ambiente referenciada não existir, a referência no comando não será alterada. Por exemplo, se a referência for a "`$(NAME1)`" e a variável de ambiente `NAME1` não existir, a string de comando continuará a ser "`$(NAME1)`". `$$` será substituído por `$` e a string resultante não será expandida. Por exemplo, `$$(VAR_NAME)` é passado como `$(VAR_NAME)`, independentemente da variável de ambiente `VAR_NAME` existir ou não. Para obter mais informações, consulte CMD na referência do Dockerfile e Definir um comando e argumentos para um pod na Kubernetes documentação.

Tipo: Matriz de strings

Obrigatório: Não

`command`

O ponto de entrada para o contêiner. Isso não é executado dentro de uma shell. Se isso não for especificado, o `ENTRYPOINT` da imagem do contêiner será usado. As referências de variáveis de ambiente são expandidas usando o ambiente do contêiner.

Se a variável de ambiente referenciada não existir, a referência no comando não será alterada. Por exemplo, se a referência for a "`$(NAME1)`" e a variável de ambiente `NAME1`

não existir, a string de comando continuará a ser "`$(NAME1)`". `$$` será substituído por `$` e a string resultante não será expandida. Por exemplo, `$$(VAR_NAME)` será passado como `$(VAR_NAME)` independentemente da variável de ambiente `VAR_NAME` existir ou não. O ponto de entrada não pode ser atualizado. Para obter mais informações, consulte ENTRYPOINT na referência do Dockerfile e Definir um comando e argumentos para um contêiner e Entrypoint na Kubernetes documentação.

Tipo: Matriz de strings

Obrigatório: Não

`env`

As variáveis de ambiente a serem passadas para um contêiner.

> **Note**
>
> Variáveis de ambiente não podem começar com "`AWS_BATCH`". Essa convenção de nomenclatura é reservada para variáveis que AWS Batch definem.

Tipo: matriz de objetos EksContainerEnvironmentVariable

Obrigatório: Não

`name`

O nome da variável de ambiente.

Tipo: Sequência

Obrigatório: Sim

`value`

O valor da variável de ambiente.

Tipo: Sequência

Obrigatório: Não

`image`

A imagem do Docker usada para iniciar um contêiner.

Tipo: Sequência

Obrigatório: Sim

`imagePullPolicy`

A política de extração de imagens para o contêiner. Os valores compatíveis são `Always`, `IfNotPresent` e `Never`. Por padrão, esse parâmetro é `IfNotPresent`. No entanto, se a tag `:latest` for especificada, o padrão será `Always`. Para obter mais informações, consulte Atualizando imagens na Kubernetes documentação.

Tipo: Sequência

Obrigatório: Não

`name`

O nome do contêiner. Se o nome não for especificado, o nome padrão "`Default`" será usado. Cada contêiner em um pod deve ter um nome exclusivo.

Tipo: Sequência

Obrigatório: Não

`resources`

O tipo e a quantidade de um recurso a serem atribuídos a um contêiner. Os recursos suportados incluem `memory`, `cpu` e `nvidia.com/gpu`. Para obter mais informações, consulte Resource de recursos para pods e contêineres na Kubernetes documentação.

Tipo: objeto EksContainerResourceRequirements

Obrigatório: Não

`limits`

O tipo e a quantidade dos recursos a reservar para o contêiner. Os valores variam de acordo com o `name` especificado. Os recursos podem ser solicitados usando o `limits` ou os objetos do `requests`.

memory

O limite rígido de memória (em MiB) para o contêiner, usando inteiros, com um sufixo "Mi". Caso o container tente exceder a memória especificada, o contêiner será encerrado. Você deve especificar pelo menos 4 MiB de memória para um

trabalho. `memory`pode ser especificada em `limits`, `requests` ou em ambos. Se `memory` for especificado nos dois lugares, o valor especificado em `limits` deverá ser igual ao valor especificado em `requests`.

> **Note**
>
> Para maximizar a utilização de recursos, forneça aos trabalhos o máximo de memória possível para o tipo de instância específico que você está usando. Para saber como, consulte Recurso de Computação Gerenciamento de Memória.

cpu

O número de CPUs reservado para o contêiner. Os valores devem ser um múltiplo par de `0.25`. `cpu` pode ser especificado em `limits`, `requests` ou em ambos. Se `cpu` for especificada em ambos, o valor especificado em `limits` deverá ser no mínimo igual ao valor especificado em `requests`.

nvidia.com/gpu

O número de GPUs reservado para o contêiner. Os valores devem ser um inteiro par. `memory` pode ser especificado em `limits`, `requests` ou em ambos. Se `memory` for especificado nos dois lugares, o valor especificado em `limits` deverá ser igual ao valor especificado em `requests`.

Tipo: Mapa de string para string

Restrições de tamanho do valor: tamanho mínimo 1. Comprimento máximo de 256.

Obrigatório: Não

`requests`

O tipo e a quantidade dos recursos a solicitar para o contêiner. Os valores variam de acordo com o `name` especificado. Os recursos podem ser solicitados usando o `limits` ou os objetos do `requests`.

memory

O limite rígido de memória (em MiB) para o contêiner, usando inteiros, com um sufixo "Mi". Caso o container tente exceder a memória especificada, o contêiner

será encerrado. Você deve especificar pelo menos 4 MiB de memória para um trabalho. `memory`pode ser especificada em `limits`, `requests` ou em ambos. Se `memory` for especificado em ambos, o valor especificado em `limits` deverá ser igual ao valor especificado em `requests`.

> **Note**
>
> Se você deseja maximizar a utilização de recursos fornecendo aos trabalhos o máximo de memória possível para um determinado tipo de instância, consulte Recurso de Computação Gerenciamento de Memória.

cpu

O número de CPUs reservado para o contêiner. Os valores devem ser um múltiplo par de `0.25`. `cpu` pode ser especificado em `limits`, `requests` ou em ambos. Se `cpu` for especificada em ambos, o valor especificado em `limits` deverá ser no mínimo igual ao valor especificado em `requests`.

nvidia.com/gpu

O número de GPUs reservado para o contêiner. Os valores devem ser um inteiro par. `nvidia.com/gpu` pode ser especificado em `limits`, `requests` ou em ambos. Se `nvidia.com/gpu` for especificado em ambos, o valor especificado em `limits` deverá ser igual ao valor especificado em `requests`.

Tipo: Mapa de string para string

Restrições de tamanho do valor: tamanho mínimo 1. Comprimento máximo de 256.

Obrigatório: Não

`securityContext`

O contexto de segurança de um trabalho. Para obter mais informações, consulte Configure um contexto de segurança para um pod ou contêiner na Kubernetes documentação.

Tipo: objeto EksContainerSecurityContext

Obrigatório: Não

`privileged`

Quando esse parâmetro é `true`, o contêiner recebe permissões elevadas na instância de contêiner host. O nível das permissões é semelhante às permissões do usuário `root`. O valor padrão é `false`. Esse parâmetro é mapeado para a `privileged` política nas políticas de segurança do pod Privileged na Kubernetes documentação.

Tipo: Booliano

Obrigatório: Não

`readOnlyRootFilesystem`

Quando esse parâmetro é `true`, o contêiner recebe acesso somente para leitura ao sistema de arquivos raiz. O valor padrão é `false`. Esse parâmetro é mapeado para a `ReadOnlyRootFilesystem` política nas políticas de segurança do pod de sistemas de arquivos e volumes na documentação Kubernetes.

Tipo: Booliano

Obrigatório: Não

`runAsGroup`

Quando esse parâmetro é especificado, o contêiner é executado como o ID de grupo especificado (`gid`). Se esse parâmetro não for especificado, o padrão será o grupo que for especificado nos metadados da imagem. Esse parâmetro mapeia para `RunAsGroup` e `MustRunAs` política nas políticas de segurança do pod de grupos e usuários na Kubernetes documentação.

Tipo: Long

Obrigatório: Não

`runAsNonRoot`

Quando esse parâmetro é especificado, o contêiner é executado como um usuário com um valor de `uid` diferente de 0. Se esse parâmetro não for especificado, essa regra será imposta. Esse parâmetro mapeia para `RunAsUser` e `MustRunAsNonRoot` política nas políticas de segurança do pod de grupos e usuários na Kubernetes documentação.

Tipo: Long

Obrigatório: Não

`runAsUser`

Quando esse parâmetro é especificado, o contêiner é executado como o ID de usuário especificado (`uid`). Se esse parâmetro não for especificado, o padrão será o usuário que for especificado nos metadados da imagem. Esse parâmetro mapeia para `RunAsUser` e `MustRanAs` política nas políticas de segurança do pod de grupos e usuários na Kubernetes documentação.

Tipo: Long

Obrigatório: Não

`volumeMounts`

O volume é montado para um contêiner para um trabalho do Amazon EKS. Para obter mais informações sobre volumes e montagens de volumes em Kubernetes, consulte Volumes na Kubernetes documentação.

Tipo: matriz de objetos EksContainerVolumeMount

Obrigatório: Não

`mountPath`

O caminho no contêiner em que o volume está montado.

Tipo: Sequência

Obrigatório: Não

`name`

O nome do volume a ser montado. Isso deve corresponder ao nome de um dos volumes no pod.

Tipo: Sequência

Obrigatório: Não

`readOnly`

Caso o valor seja `true`, o contêiner tem acesso somente leitura ao volume. Caso contrário, o contêiner pode gravar no volume. O valor padrão é `false`.

Tipo: booliano

Obrigatório: Não

`dnsPolicy`

A política de DNS para o pod. O valor padrão é `ClusterFirst`. Se o parâmetro `hostNetwork` não for especificado, o padrão será `ClusterFirstWithHostNet`. `ClusterFirst` indica que qualquer consulta ao DNS que não corresponda ao sufixo de domínio de cluster configurado será encaminhada para o servidor de nomes upstream herdado do nó. Se nenhum valor foi especificado `dnsPolicy` na operação da RegisterJobDefinitionAPI, nenhum valor será retornado `dnsPolicy` por nenhuma das DescribeJobDefinitionsoperações da DescribeJobsAPI. A configuração de especificação do pod conterá `ClusterFirst` ou `ClusterFirstWithHostNet`, dependendo do valor do parâmetro `hostNetwork`. Para obter mais informações, consulte política Pod's DNS na Kubernetes documentação.

Valores válidos: `Default` | `ClusterFirst` | `ClusterFirstWithHostNet`

Tipo: Sequência

Obrigatório: Não

`hostNetwork`

Indica se o pod usa o endereço IP da rede dos hosts. O valor padrão é `true`. Definir essa opção como `false` ativa o modelo de rede de pods Kubernetes. A maioria das AWS Batch cargas de trabalho é somente de saída e não exige a sobrecarga da alocação de IP para cada pod para conexões de entrada. Para obter mais informações, consulte Espaços de nomes de Host e rede de Pods na Kubernetes documentação.

Tipo: Booliano

Obrigatório: Não

`serviceAccountName`

O nome da conta de serviço que é usada para executar o pod. Para obter mais informações, consulte Kubernetes contas de serviço e Configure uma conta de serviço Kubernetes para assumir um perfil do IAM no Amazon EKS User Guide e Configure contas de serviço para pods na Kubernetes documentação.

Tipo: Sequência

Obrigatório: Não

`volumes`

Especifica os volumes para uma definição de trabalho que usa recursos do Amazon EKS.

Tipo: matriz de objetos EksVolume

Obrigatório: Não

emptyDir

Especifica a configuração de um volume Kubernetes `emptyDir`. Um volume `emptyDir` é criado pela primeira vez quando um pod é atribuído a um nó. Ele existirá enquanto o pod estiver sendo executado nesse nó. O volume `emptyDir` está inicialmente vazio. Todos os contêineres no pod podem ler e gravar os arquivos no volume `emptyDir`. No entanto, o volume `emptyDir` pode ser montado no mesmo caminho ou em caminhos diferentes em cada contêiner. Quando um pod é removido de um nó por qualquer motivo, os dados no `emptyDir` são excluídos permanentemente. Para obter mais informações, consulte emptyDir na Kubernetes documentação.

Tipo: objeto EksEmptyDir

Obrigatório: Não

medium

A mídia para armazenar o volume. O valor padrão é uma string vazia, que usa o armazenamento do nó.

""

(Padrão) Usar o armazenamento em disco do nó.

"Memory" (Memória)

Usar o volume `tmpfs` que tem o suporte da RAM do nó. O conteúdo do volume é perdido quando o nó é reinicializado e qualquer armazenamento no volume é contabilizado no limite de memória do contêiner.

Tipo: Sequência

Obrigatório: Não

sizeLimit

O tamanho máximo do volume. Por padrão, não há tamanho máximo definido.

Tipo: Sequência

Restrições de tamanho: tamanho mínimo 1. Tamanho máximo de 256.

Obrigatório: Não

hostPath

Especifica a configuração de um volume Kubernetes `hostPath`. Um volume `hostPath` monta um arquivo ou diretório existente do sistema de arquivos do nó host no pod. Para obter mais informações, consulte hostPath na Kubernetes documentação.

Tipo: objeto EksHostPath

Obrigatório: Não

path

O caminho do arquivo ou diretório no host a ser montado em contêineres no pod.

Tipo: Sequência

Obrigatório: Não

name

O nome do volume. O nome deve ser permitido como um nome de subdomínio do DNS. Para obter mais informações, consulte Nomes de subdomínios DNS na Kubernetes documentação.

Tipo: Sequência

Obrigatório: Sim

secret

Especifica a configuração de um volume Kubernetes `secret`. Para obter mais informações, consulte secret na Kubernetes documentação.

Tipo: objeto EksSecret

Obrigatório: Não

optional

Especifica se o segredo ou as chaves do segredo devem ser definidas.

Tipo: Booliano

Obrigatório: Não

secretName

O nome do segredo. O nome deve ser permitido como um nome de subdomínio do DNS. Para obter mais informações, consulte Nomes de subdomínios DNS na Kubernetes documentação.

Tipo: Sequência

Obrigatório: Sim

Recursos da plataforma

`platformCapabilities`

Os recursos da plataforma exigidos pela definição do cargo. Se nenhum valor for especificado, o padrão será `EC2`. Para trabalhos executados em recursos do Fargate, `FARGATE` é especificado.

> **Note**
>
> Se o trabalho for executado em recursos do Amazon EKS, você não deve especificar`platformCapabilities`.

Tipo: Sequência

Valores válidos: `EC2` | `FARGATE`

Obrigatório: Não

Propagar tags

`propagateTags`

Especifica se as tags devem ser propagadas a partir do trabalho ou da definição de trabalho para a tarefa correspondente do Amazon ECS. Se nenhum valor for especificado, as tags não serão propagadas. As tags só podem ser propagadas para as tarefas quando as tarefas são criadas.

Para tags com o mesmo nome, as tags de trabalho têm prioridade sobre as tags de definições de trabalho. Se o número total de tags combinadas do trabalho e da definição do trabalho for superior a 50, o trabalho será movido para o estado `FAILED`.

> **Note**
>
> Se o trabalho for executado em recursos do Amazon EKS, você não deve especificar`propagateTags`.

Tipo: Booliano

Obrigatório: Não

Propriedades de nó

`nodeProperties`

Ao registrar uma definição de trabalho paralela de vários nós, você deve especificar uma lista de propriedades de nós. Essas propriedades do nó definem o número de nós a serem usados em seu trabalho, o índice do nó principal e os diferentes intervalos de nós a serem usados. Se o trabalho for executado em recursos do Fargate, não especifique `nodeProperties`. Em seu lugar, use `containerProperties`. As seguintes propriedades de nó são permitidas em uma definição de trabalho. Para ter mais informações, consulte Trabalhos paralelos de vários nós .

> **Note**
>
> Se o trabalho for executado em recursos do Amazon EKS, você não deve especificar`nodeProperties`.

Tipo: objeto NodeProperties

Obrigatório: Não

`mainNode`

Especifica o índice de nó para o nó principal de um trabalho em paralelo de vários nós. Esse valor de índice do nó deve ser menor que o número de nós.

Tipo: Inteiro

Obrigatório: Sim

`numNodes`

O número de nós que são associados a um trabalho em paralelo de vários nós.

Tipo: Inteiro

Obrigatório: Sim

`nodeRangeProperties`

Uma lista de intervalos de nós e suas propriedades que são associadas a uma trabalho em paralelo de vários nós.

> **Note**
>
> Um grupo de nós é um grupo idêntico de nós de trabalhos que compartilham as mesmas propriedades do contêiner. Você pode usar AWS Batch para especificar até cinco grupos de nós distintos para cada trabalho.

Tipo: matriz de objetos NodeRangeProperty

Obrigatório: Sim

`targetNodes`

O intervalo de nós, usando valores de índice de nó. Um intervalo de `0:3` indica nós com valores de índice de `0` a `3`. Se o valor do intervalo inicial for omitido (`:n`), então 0 será usado para iniciar o intervalo. Se o valor do intervalo final for omitido (`n:`), o índice de nó mais alto possível será usado para finalizar o intervalo. Seus intervalos de nós acumulativos devem considerar todos os nós (`0:n`). Você pode aninhar intervalos de nós (por exemplo, `0:10` e `4:5`). Nesse caso, as propriedades do intervalo `4:5` substituem as propriedades de `0:10`.

Tipo: Sequência

Obrigatório: Não

`container`

Os detalhes do contêiner para o intervalo de nós. Para ter mais informações, consulte Propriedades do contêiner.

Tipo: objeto ContainerProperties

Obrigatório: Não

Estratégia de repetição

`retryStrategy`

Ao registrar uma definição de trabalho, você pode opcionalmente especificar uma estratégia de nova tentativa a ser usada para trabalhos com falha que são enviados com essa definição de trabalho. Qualquer estratégia de repetição especificada durante uma SubmitJoboperação substitui a estratégia de repetição definida aqui. Por padrão, cada trabalho é tentado uma vez. Se você especificar mais de uma tentativa, o trabalho será repetido se falhar. Exemplos de uma tentativa de falha incluem o trabalho retornar um código de saída diferente de zero ou a instância de contêiner ser encerrada. Para ter mais informações, consulte Repetições de trabalho automatizadas.

Tipo: objeto RetryStrategy

Obrigatório: Não

`attempts`

O número de vezes para mover um trabalho para o status `RUNNABLE`. Você pode especificar entre 1 e 10 tentativas. Se `attempts` for maior que um, o trabalho será repetido essa quantidade de vezes se falhar, até ser movido para `RUNNABLE`.

```
"attempts": integer
```

Tipo: inteiro

Obrigatório: Não

`evaluateOnExit`

Matriz de até 5 objetos que especificam condições sob as quais o trabalho deve ser repetido ou considerado com falha. Se esse parâmetro for especificado, o parâmetro `attempts`

também deverá ser especificado. Se `evaluateOnExit` for especificado, mas nenhuma das entradas corresponder, o trabalho será repetido.

```
"evaluateOnExit": [
   {
      "action": "string",
      "onExitCode": "string",
      "onReason": "string",
      "onStatusReason": "string"
   }
]
```

Tipo: matriz de objetos EvaluateOnExit

Obrigatório: Não

`action`

Especifica a ação a ser executada se todas as condições especificadas (`onStatusReason`, `onReason`, e `onExitCode`) forem atendidas. Os valores não diferenciam maiúsculas e minúsculas

Tipo: Sequência

Obrigatório: Sim

Valores válidos: `RETRY` | `EXIT`

`onExitCode`

Contém um padrão glob para corresponder à representação decimal do `ExitCode` retornado para um trabalho. O padrão pode ter até 512 caracteres de comprimento. Ele pode conter somente números. Ele não pode conter letras ou caracteres especiais. Ele pode, opcionalmente, terminar com um asterisco (*) para que apenas o início da string precise ser uma correspondência exata.

Tipo: Sequência

Obrigatório: Não

`onReason`

Contém um padrão glob para corresponder ao `Reason` retornado para um trabalho. O padrão pode ter até 512 caracteres de comprimento. Ele pode conter letras, números,

pontos (.), dois pontos (:) e espaço em branco (incluindo espaços ou tabulações). Ele pode, opcionalmente, terminar com um asterisco (*) para que apenas o início da string precise ser uma correspondência exata.

Tipo: Sequência

Obrigatório: Não

`onStatusReason`

Contém um padrão glob para corresponder ao `StatusReason` retornado para um trabalho. O padrão pode ter até 512 caracteres de comprimento. Ele pode conter letras, números, pontos (.), dois pontos (:) e espaço em branco (incluindo espaços ou tabulações). Ele pode, opcionalmente, terminar com um asterisco (*) para que apenas o início da string precise ser uma correspondência exata.

Tipo: Sequência

Obrigatório: Não

Prioridade de agendamento

`schedulingPriority`

A prioridade de agendamento para trabalhos enviados com essa definição de trabalho. Isso afeta apenas os trabalhos nas filas de trabalhos com uma política de compartilhamento justo. Trabalhos com prioridade de agendamento mais alta são agendados antes de trabalhos com prioridade de agendamento mais baixa.

O valor mínimo suportado é 0 e o valor máximo suportado é 9999.

Tipo: Inteiro

Obrigatório: Não

Tags

`tags`

Etiquetas de pares de valores-chave a serem associadas à definição do trabalho. Para ter mais informações, consulte Marcando seus Recursos AWS Batch.

Tipo: Mapa de string para string

Obrigatório: Não

Timeout (Tempo limite)

`timeout`

Você pode configurar uma duração de tempo limite para seus trabalhos para que, se um trabalho for executado por mais tempo do que isso, AWS Batch encerre o trabalho. Para ter mais informações, consulte Tempos limite do trabalho. Se um trabalho for encerrado devido a um tempo limite, ele não será tentado novamente. Qualquer configuração de tempo limite especificada durante uma SubmitJoboperação substitui a configuração de tempo limite definida aqui. Para ter mais informações, consulte Tempos limite do trabalho.

Tipo: objeto JobTimeout

Obrigatório: Não

`attemptDurationSeconds`

O tempo de duração em segundos (medido a partir do registro de data e hora `startedAt` da tentativa de trabalho) após o encerramento de trabalhos inacabados AWS Batch . O valor mínimo do tempo limite é 60 segundos.

Em trabalhos de matriz, o tempo limite se aplica aos trabalhos secundários, não ao trabalho de matriz principal.

Para trabalhos paralelos de vários nós (MNP), o tempo limite se aplica ao trabalho inteiro, não aos nós individuais.

Tipo: Inteiro

Obrigatório: Não

Criação de definições de trabalho usando EcsProperties

Com o uso das definições de AWS Batch tarefas `EcsProperties`, você pode modelar hardware, sensores, ambientes 3D e outras simulações em contêineres separados. Você pode usar esse recurso para organizar logicamente os componentes da carga de trabalho e separá-los do aplicativo

principal. Esse recurso pode ser usado AWS Batch no Amazon Elastic Container Service (Amazon ECS), no Amazon Elastic Kubernetes Service (Amazon EKS) e. AWS Fargate

`ContainerProperties`versus definições `EcsProperties` de trabalho

Você pode optar por usar `ContainerProperties`nossas definições de `EcsProperties`tarefa de acordo com seu caso de uso. Em um alto nível, executar AWS Batch trabalhos com `EcsProperties` é semelhante à execução de trabalhos com um`ContainerProperties`.

A estrutura antiga de definição de tarefas, usando`ContainerProperties`, permanece suportada. Se você atualmente tem fluxos de trabalho usando essa estrutura, você pode continuar a executá-los.

A principal diferença é que há um novo objeto adicionado à definição do trabalho para acomodar as definições `EcsProperties` baseadas.

Por exemplo, uma definição de trabalho usada `ContainerProperties` no Amazon ECS e no Fargate tem a seguinte estrutura:

```
{
   "containerProperties": {
      ...
      "image": "my_ecr_image1",
      ...
  },
...
}
```

Uma definição de trabalho usada `EcsProperties` no Amazon ECS e no Fargate tem a seguinte estrutura:

```
{
  "ecsProperties": {
    "taskProperties": [{
      "containers": [
        {
          ...
          "image": "my_ecr_image1",
          ...
        },
        {
          ...
```

```
          "image": "my_ecr_image2",
          ...
        },
```

Mudanças gerais nas AWS Batch APIs

A seguir, descrevemos algumas das principais diferenças ao usar os tipos de dados `EcsProperties` e da `EcsProperties` API:

- Muitos dos parâmetros usados no interior `ContainerProperties` aparecem dentro`TaskContainerProperties`. Alguns exemplos incluem`command`,`image`, `privileged`secrets`,, `users` e. Todos eles podem ser encontrados dentro TaskContainerProperties.
- Alguns dos `TaskContainerProperties` parâmetros não têm equivalentes funcionais na estrutura legada. Alguns exemplos incluem`dependsOn`,`essential`, `name`ipcMode`,, `pid`Mode e. Para obter mais informações, consulte EcsTaskDetails`TaskContainerProperties`e.

  Além disso, alguns `ContainerProperties` parâmetros não têm equivalentes ou aplicação na `EcsProperties` estrutura. Em `taskProperties`, `container` foi substituído por `containers` para que o novo objeto possa aceitar até dez elementos. Para obter mais informações, consulte:ContainerProperties e:containersRegisterJobDefinition. EcsTaskProperties
- `taskRoleArn`é funcionalmente equivalente a. `jobRoleArn` Para obter mais informações, consulte EcsTaskProperties: taskRoleArn e ContainerProperties: jobRoleArn.
- Você pode incluir de um (1) a dez (10) contêineres na `EcsProperties` estrutura. Para obter mais informações, consulte:containersEcsTaskProperties.
- Os objetos `taskProperties` e instanceTypes são matrizes, mas atualmente aceitam somente um elemento. Por exemplo, :taskProperties e:instanceTypesEcsProperties. NodeRangeProperty

Definições de trabalho de vários contêineres para o Amazon ECS

Para acomodar a estrutura de vários contêineres do Amazon ECS, alguns dos tipos de dados da API são diferentes. Por exemplo,

- `ecsProperties`é o mesmo nível da definição `containerProperties` de contêiner único. Para obter mais informações, consulte EcsPropertieso Guia de referência AWS Batch da API.
- `taskProperties`contém as propriedades definidas para a tarefa do Amazon ECS. Para obter mais informações, consulte EcsPropertieso Guia de referência AWS Batch da API.

- `containers`inclui informações semelhantes às da definição `containerProperties` de contêiner único. A principal diferença é que `containers` permite definir até dez contêineres. Para obter mais informações, consulte ecs:Containers TaskProperties no Guia de AWS Batch referência da API.
- `essential`parâmetro indica como o contêiner afeta o trabalho. Todos os contêineres essenciais devem ser concluídos com êxito (sair como 0) para que o trabalho progrida. Se um contêiner marcado como essencial falhar (sair como diferente de 0), o trabalho falhará.

  O valor padrão é `true` e pelo menos um contêiner deve ser marcado como`essential`. Para obter mais informações, consulte `essential` no AWS Batch API Reference Guide (Guia de referência da API do &CDSlong;).
- Com o `dependsOn`parâmetro, você pode definir uma lista de dependências de contêineres. Para obter mais informações, consulte `dependsOn` no AWS Batch API Reference Guide (Guia de referência da API do &CDSlong;).

> **Note**
>
> A complexidade da `dependsOn` lista e o tempo de execução do contêiner associado podem afetar o horário de início do seu trabalho. Se as dependências demorarem muito para serem executadas, o trabalho permanecerá em um `STARTING` estado até que seja concluído.

Para obter mais informações sobre a estrutura `ecsProperties` e, consulte a sintaxe de RegisterJobDefinitionsolicitação para ECSProperties.

Definições de tarefas de vários contêineres para o Amazon EKS

Para acomodar a estrutura de vários contêineres do Amazon EKS, alguns dos tipos de dados da API são diferentes. Por exemplo,

- `name`é um identificador exclusivo para o contêiner. Esse objeto não é necessário para um único contêiner, mas é necessário ao definir vários contêineres em um pod. Quando `name` não está definido para contêineres únicos, o nome padrão,`default`, é aplicado.
- `initContainers`são definidos dentro do tipo de eksPodPropertiesdados. Eles são executados antes dos contêineres do aplicativo, sempre são executados até a conclusão e devem ser concluídos com êxito antes do início do próximo contêiner.

Esses contêineres são registrados com o agente Amazon EKS Connector e mantêm as informações de registro no armazenamento de dados de back-end do Amazon Elastic Kubernetes Service. O `initContainers` objeto pode aceitar até dez (10) elementos. Para obter mais informações, consulte Init Containers na Kubernetesdocumentação.

> **Note**
>
> O `initContainers` objeto pode afetar a hora de início do seu trabalho. Se `initContainers` demorarem muito para serem executados, o trabalho permanecerá em um `STARTING` estado até que seja concluído.

- `shareProcessNamespace`indica se os contêineres no pod podem compartilhar o mesmo namespace de processo. O valor padrão é `false`. Definir isso `true` para permitir que os contêineres vejam e sinalizem processos em outros contêineres localizados no mesmo pod.
- Cada contêiner tem importância. Todos os contêineres devem ser concluídos com êxito (sair como 0) para que o trabalho seja bem-sucedido. Se um contêiner falhar (sair como diferente de 0), o trabalho falhará.

Para obter mais informações sobre a estrutura `eksProperties` e, consulte a sintaxe de RegisterJobDefinitionsolicitação para EksProperties.

AWS Batch cenários de trabalho usando EcsProperties

Para ilustrar como as definições de AWS Batch tarefas que `EcsProperties` você usa podem ser estruturadas com base em suas necessidades, este tópico apresenta as seguintes `RegisterJobDefinition`cargas úteis. Você pode copiar esses exemplos em um arquivo, personalizá-los de acordo com suas necessidades e, em seguida, usar o AWS Command Line Interface (AWS CLI) para chamar`RegisterJobDefinition`.

AWS Batch trabalho para o Amazon Elastic Container Service no Amazon Elastic Compute Cloud

```
{
    "jobDefinitionName": "multicontainer-ecs-ec2",
    "type": "container",
    "ecsProperties": {
        "taskProperties": [
```

```
        {
          "containers": [
            {
              "name": "c1",
              "essential": false,
              "command": [
                "echo",
                "hello world"
              ],
              "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
              "resourceRequirements": [
                {
                  "type": "VCPU",
                  "value": "2"
                },
                {
                  "type": "MEMORY",
                  "value": "4096"
                }
              ]
            },
            {
              "name": "c2",
              "essential": true,
              "command": [
                "echo",
                "hello world"
              ],
              "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
              "resourceRequirements": [
                {
                  "type": "VCPU",
                  "value": "6"
                },
                {
                  "type": "MEMORY",
                  "value": "12288"
                }
              ]
            }
          ]
        }
      ]
}
```

```
}
```

AWS Batch trabalho para o Amazon ECS em AWS Fargate

```
{
    "jobDefinitionName": "multicontainer-ecs-fargate",
    "type": "container",
    "platformCapabilities": [
        "FARGATE"
    ],
    "ecsProperties": {
        "taskProperties": [
          {
            "containers": [
              {
                "name": "c1",
                "command": [
                  "echo",
                  "hello world"
                ],
                "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                "resourceRequirements": [
                  {
                    "type": "VCPU",
                    "value": "2"
                  },
                  {
                    "type": "MEMORY",
                    "value": "4096"
                  }
                ]
              },
              {
                "name": "c2",
                "essential": true,
                "command": [
                  "echo",
                  "hello world"
                ],
                "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                "resourceRequirements": [
                  {
                    "type": "VCPU",
```

```
                    "value": "6"
                  },
                  {
                    "type": "MEMORY",
                    "value": "12288"
                  }
                ]
              }
            ],
            "executionRoleArn": "arn:aws:iam::1112223333:role/ecsTaskExecutionRole"
          }
        ]
  }
}
```

AWS Batch trabalho para o Amazon Elastic Kubernetes Service

```
{
  "jobDefinitionName": "multicontainer-eks",
  "type": "container",
  "eksProperties": {
    "podProperties": {
      "shareProcessNamespace": true,
      "initContainers": [
        {
          "name": "init-container",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "echo"
          ],
          "args": [
            "hello world"
          ],
          "resources": {
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        },
        {
          "name": "init-container-2",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
```

```
          "command": [
            "echo",
            "my second init container"
          ],
          "resources": {
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        }
      ],
      "containers": [
        {
          "name": "c1",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "echo world"
         ],
          "resources": {
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        },
        {
          "name": "sleep-container",
          "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
          "command": [
            "sleep",
            "20"
          ],
          "resources": {
            "requests": {
              "cpu": "1",
              "memory": "512Mi"
            }
          }
        }
      ]
    }
  }
```

```
}
```

Trabalho paralelo de vários nós (MNP) com vários contêineres por AWS Batch nó

```
{
  "jobDefinitionName": "multicontainer-mnp",
  "type": "multinode",
  "nodeProperties": {
    "numNodes": 6,
    "mainNode": 0,
    "nodeRangeProperties": [
      {
        "targetNodes": "0:5",
        "ecsProperties": {
          "taskProperties": [
            {
              "containers": [
                {
                  "name": "range05-c1",
                  "command": [
                    "echo",
                    "hello world"
                  ],
                  "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                  "resourceRequirements": [
                    {
                      "type": "VCPU",
                      "value": "2"
                    },
                    {
                      "type": "MEMORY",
                      "value": "4096"
                    }
                  ]
                },
                {
                  "name": "range05-c2",
                  "command": [
                    "echo",
                    "hello world"
                  ],
                  "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
                  "resourceRequirements": [
```

```
                    {
                      "type": "VCPU",
                      "value": "2"
                    },
                    {
                      "type": "MEMORY",
                      "value": "4096"
                    }
                  ]
                }
              ]
            }
          ]
        }
      }
    ]
  }
}
```

Usar o driver de log awslogs

Por padrão, o AWS Batch permite que o driver de log `awslogs` envie informações de log ao CloudWatch Logs. Você pode usar esse atributo para visualizar logs diferentes dos contêineres em um local indicado e evita que os logs de contêiner ocupem espaço em disco nas instâncias de contêiner. Este tópico ajuda você a configurar o driver de log `awslogs` em suas definições de trabalho.

> **Note**
>
> No console do AWS Batch, você pode configurar o driver de log `awslogs` registro na seção de configuração de log ao criar uma definição de trabalho.

> **Note**
>
> O tipo de informações registradas em log pelos contêineres em seu trabalho depende principalmente do comando `ENTRYPOINT`. Por padrão, os logs capturados mostram a saída do comando que você normalmente vê em um terminal interativo, se executasse o contêiner localmente, que são os fluxos de E/S `STDOUT` e `STDERR`. O driver de log `awslogs` simplesmente envia esses logs do Docker para o CloudWatch Logs. Para obter mais

informações sobre como os logs do Docker são processados, incluindo maneiras alternativas de capturar fluxos ou dados de arquivos diferentes, consulte Visualizar logs de um contêiner ou serviço na documentação do Docker.

Para enviar logs de sistema a partir das instâncias de contêiner para o CloudWatch Logs, consulte Como usar o CloudWatch Logs com AWS Batch. Para obter mais informações sobre o CloudWatch Logs, consulte Monitorar arquivos de log e Cotas do CloudWatch Logs no Guia do usuário do Amazon CloudWatch Logs.

Opções disponíveis do driver de log awslogs

O driver de log `awslogs` dá suporte às seguintes opções em definições de trabalho do AWS Batch. Para obter mais informações, consulte Driver de registro do CloudWatch Logs na documentação do Docker.

`awslogs-region`

Obrigatório: não

Especifique a região para a qual o driver de log `awslogs` deve enviar os logs do Docker. Por padrão, a região usada é a mesma do trabalho. É possível optar por enviar todos os logs de trabalhos em regiões distintas para uma única região no CloudWatch Logs. Isso permite que todos sejam visíveis em um único local. Como alternativa, você pode separá-los por região para uma abordagem mais granular. No entanto, quando escolher esta opção, certifique-se de que os grupos de logs especificados existam na região especificada.

`awslogs-group`

Obrigatório: opcional

Com a opção `awslogs-group`, você pode especificar o grupo de logs para o qual o driver de log `awslogs` envia seus fluxos de logs. Se esse campo não for especificado, será usado `aws/batch/job`.

`awslogs-stream-prefix`

Obrigatório: opcional

Com a opção `awslogs-stream-prefix`, é possível associar um fluxo de logs ao prefixo especificado e a ID da tarefa do Amazon ECS do trabalho do AWS Batch à qual o contêiner

pertence. Caso você especifique um prefixo com essa opção, o fluxo de log utiliza o seguinte formato:

```
prefix-name/default/ecs-task-id
```

`awslogs-datetime-format`

Obrigatório: não

Essa opção define um padrão de início de várias linhas no formato `strftime` em Python. Uma mensagem de log é formada por uma linha em conformidade com o padrão e as linhas seguintes que não correspondem ao padrão. Assim, a linha em conformidade é o delimitador entre as mensagens de log.

Um exemplo de um caso de uso para esse formato é a análise da saída, como um despejo de pilha, que poderia ser registrado em várias entradas. O padrão correto permite que ele seja capturado em uma única entrada.

Para obter mais informações, consulte awslogs-datetime-format.

Essa opção sempre terá precedência se os `awslogs-datetime-format` e `awslogs-multiline-pattern` estiverem configurados.

> **Note**
>
> O registro em várias linhas executa a análise da expressão regular e a correspondência de todas as mensagens de log. Isso pode ter um impacto negativo na performance do registro em log.

`awslogs-multiline-pattern`

Obrigatório: não

Essa opção define um padrão inicial de várias linhas usando uma expressão regular. Uma mensagem de log é formada por uma linha em conformidade com o padrão e as linhas seguintes que não correspondem ao padrão. Assim, a linha em conformidade é o delimitador entre as mensagens de log.

Para obter mais informações, consulte awslogs-multiline-pattern na documentação do Docker.

Essa opção será ignorada se `awslogs-datetime-format` também estiver configurado.

> **Note**
>
> O registro em várias linhas executa a análise da expressão regular e a correspondência de todas as mensagens de log. Isso pode ter um impacto negativo na performance do registro em log.

`awslogs-create-group`

Obrigatório: não

Especifique se você deseja que o grupo de logs seja criado automaticamente. Se essa opção não for especificada, o padrão será `false`.

> **Warning**
>
> Essa opção não é recomendada. Recomendamos que você crie o grupo de logs com antecedência usando a ação da API CreateLogGroup do CloudWatch Logs, pois cada trabalho tenta criar o grupo de logs, aumentando a chance de falha do trabalho.

> **Note**
>
> A política do IAM para seu perfil de execução deve incluir a permissão `logs:CreateLogGroup` antes que você tente usar `awslogs-create-group`.

Como especificar uma configuração de log na definição de trabalho

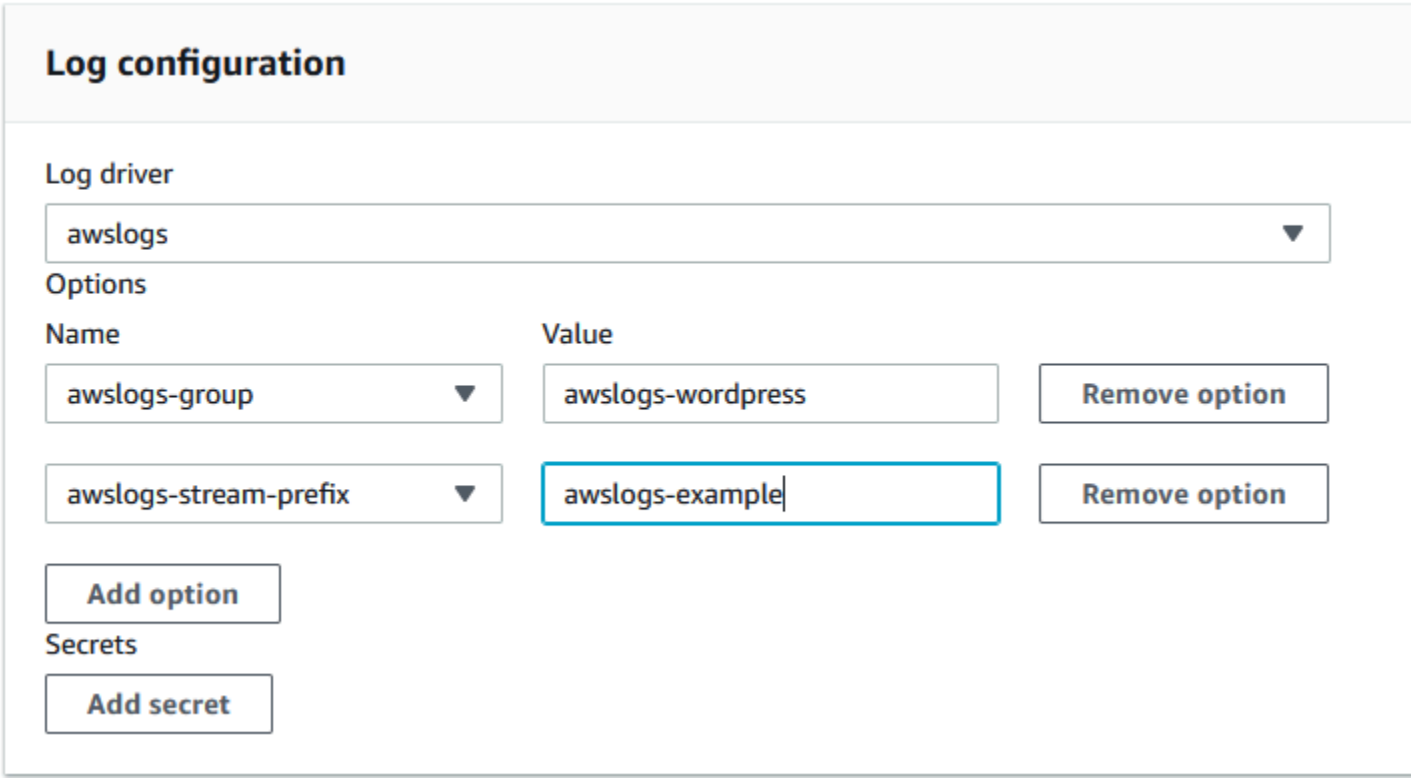
Por padrão, o AWS Batch ativa o driver de log `awslogs`. Esta seção descreve como personalizar a configuração de log `awslogs` de um trabalho. Para obter mais informações, consulte Como criar uma definição de tarefa de nó único .

Os trechos JSON de configuração de log a seguir têm um objeto `logConfiguration` especificado para cada trabalho. Um é para um trabalho do WordPress que envia logs para um grupo de logs chamado `awslogs-wordpress` e o outro é para um contêiner MySQL que envia logs para um grupo de logs chamado `awslogs-mysql`. Ambos os contêineres usam o prefixo de fluxo de log `awslogs-example`.

```
"logConfiguration": {
    "logDriver": "awslogs",
    "options": {
        "awslogs-group": "awslogs-wordpress",
        "awslogs-stream-prefix": "awslogs-example"
    }
}
```

```
"logConfiguration": {
    "logDriver": "awslogs",
    "options": {
        "awslogs-group": "awslogs-mysql",
        "awslogs-stream-prefix": "awslogs-example"
    }
}
```

No console do AWS Batch, a configuração de log da definição do trabalho do `wordpress` é especificada conforme mostrado na imagem.



Depois que tiver registrado uma definição de tarefa com o driver de log `awslogs` em uma configuração de log de definição de trabalho, você poderá enviar um trabalho com essa definição

de trabalho para começar a enviar logs ao CloudWatch Logs. Para obter mais informações, consulte Enviando um trabalho.

Especificando dados sigilosos

Com AWS Batch, você pode injetar dados confidenciais em suas tarefas armazenando seus dados confidenciais em AWS Secrets Manager secretos ou AWS Systems Manager nos parâmetros do Parameter Store, para, em seguida, e, referenciá-los na sua definição de tarefa.

Dados sigilosos podem ser expostos a uma tarefa das seguintes formas:

- Para injetar dados confidenciais em seus contêineres como variáveis de ambiente, use o `secrets` parâmetro de definição de tarefa.
- Para fazer referência a informações confidenciais na configuração de log de uma tarefa, use o `secretOptions` parâmetro de definição de trabalho.

Tópicos

Especificando dados sigilosos usando segredos do Secrets Manager

Com AWS Batch, você pode injetar dados confidenciais em seus trabalhos armazenando seus dados confidenciais em AWS Secrets Manager segredos e, em seguida, referenciando-os na definição de seu trabalho. Os dados sigilosos armazenados em segredos do Secrets Manager podem ser expostos a um trabalho como variáveis de ambiente, ou como parte da configuração do log.

Ao injetar um segredo como uma variável de ambiente, você pode especificar uma chave JSON ou versão de um segredo a ser injetado. Esse processo ajuda a controlar os dados sigilosos expostos a sua tarefa. Para mais informações sobre versionamento de segredos, consulte Termos e Conceitos Importantes para AWS Secrets Manager no AWS Secrets Manager Guia do Usuário.

Considerações para especificar dados sigilosos usando Secrets Manager

As informações a seguir devem ser consideradas quando o Secrets Manager for usado para especificar dados sigilosos para tarefas.

- Para injetar um segredo usando uma chave JSON específica ou versão de um segredo, sua instância de contêiner no ambiente de computação do atendente de contêiner do Amazon ECS instalado deve ser 1.37.0 ou superior. Recomendamos usar a versão mais recente do atendente de contêiner. Para mais informações sobre como verificar a versão do agente e atualizá-la para a versão mais recente, consulte Atualizando Atendente de Contêiner Amazon ECS no Guia do Desenvolvedor Amazon Elastic Container Service.

  Para injetar o conteúdo completo de um segredo como uma variável de ambiente ou injetar um segredo em uma configuração de log, sua instância de contêiner deve ter a versão 1.23.0 ou posterior do agente de contêiner.
- Somente segredos que armazenam dados de texto, que são segredos criados com o `SecretString` parâmetro da CreateSecretAPI, são suportados. Segredos que armazenam dados binários, que são segredos criados com o `SecretBinary` parâmetro da CreateSecretAPI, não são compatíveis.
- Ao usar uma definição de trabalho que faça referência a segredos Secrets Manager para recuperar dados confidenciais para suas tarefas, caso também esteja usando VPC endpoint de interface, você deverá criar endpoints de VPC de interface para Secrets Manager. Para mais informações, consulte Usando Secrets Manager com endpoints VPC no AWS Secrets Manager Guia de Usuário.
- Os dados sigilosos são injetados na tarefa quando a mesma é iniciada. Caso o segredo seja subsequentemente atualizado ou alternado, a tarefa não receberá o valor atualizado automaticamente. Você deve iniciar uma nova tarefa para forçar o serviço a iniciá-la com o valor do segredo atualizado.

Permissões do IAM necessárias para AWS Batch segredos

Para este atributo, você precisa possuir a função de execução e referenciá-la em sua definição de trabalho. Isso permite que o atendente de contêiner puxe os recursos Secrets Manager necessários. Para mais informações, consulte AWS Batch função de execução do IAM.

Para prover acesso aos segredos Secrets Manager criados por você, adicione manualmente as permissões a seguir como política em linha ao papel de execução. Para mais informações, consulte Adicionando e Removendo Políticas do IAM no Guia de Usuário do IAM.

- `secretsmanager:GetSecretValue`– obrigatório caso faça referência a um segredo Secrets Manager.

- `kms:Decrypt`– obrigatório apenas se o seu segredo usar uma chave KMS personalizada, não a chave padrão. O ARN da sua chave personalizada deve ser adicionado como recurso.

O exemplo de política em linha a seguir adiciona as permissões obrigatórias.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:<secret_name>",
        "arn:aws:kms:<region>:<aws_account_id>:key/<key_id>"
      ]
    }
  ]
}
```

Injetando dados sigilosos como uma variável de ambiente

Na definição do seu trabalho, você pode especificar os seguintes itens:

- Objeto `secrets` contendo o nome da variável de ambiente a ser definido no campo
- Nome de recurso da Amazon (ARN) do segredo Secrets Manager
- Parâmetros adicionais contendo dados sigilosos a serem apresentados a tarefa

O exemplo a seguir mostra a sintaxe completa a ser especificada para o segredo Secrets Manager.

```
arn:aws:secretsmanager:region:aws_account_id:secret:secret-name:json-key:version-
stage:version-id
```

A seção a seguir descreve os parâmetros adicionais. Esses parâmetros são opcionais. No entanto, caso opte por não utilizá-los, você precisará incluir dois pontos `:` para valores padrão. Abaixo, exemplos provendo mais contexto.

`json-key`

Especifica o nome da chave em um par de chave/ valor, com o valor que deseje definir como o valor da variável de ambiente. Somente valores formato JSON são compatíveis. Caso não especifique uma chave JSON, o conteúdo completo do segredo será utilizado.

`version-stage`

Especifique o rótulo de preparação da versão do segredo que deseja usar. Caso um rótulo de preparação de versão seja especificado, você não poderá especificar um ID da versão. Se nenhum estágio de versão for especificado, o comportamento padrão será recuperar o segredo com o rótulo de preparação `AWSCURRENT`.

Rótulos de preparação são usados para monitoramento de diferentes versões de um segredo quando eles forem atualizados ou alternados. Cada versão de um segredo tem um ou mais rótulos de preparação e uma ID. Para obter mais informações, consulte Key Terms and AWS Concepts for Secrets Manager no Guia AWS Secrets Manager do Usuário.

`version-id`

Especifica o identificador exclusivo da versão do segredo que você deseja usar. Se uma ID da versão for especificada, você não poderá especificar um rótulo de preparação da versão. Se nenhuma ID de versão for especificada, o comportamento padrão será recuperar o segredo com o rótulo de preparação `AWSCURRENT`.

IDs da versão são usadas para monitoramento de diferentes versões de um segredo quando atualizados ou alternados. Cada versão de um segredo tem uma ID. Para obter mais informações, consulte Key Terms and AWS Concepts for Secrets Manager no Guia AWS Secrets Manager do Usuário.

Exemplo de definições de contêiner

Os exemplos a seguir mostram maneiras de fazer referência a segredos Secrets Manager nas suas definições de contêiner.

Example Referenciando um segredo completo

A seguir, um trecho de definição de tarefa mostrando formato, ao fazer referência ao texto completo de um segredo Secrets Manager.

```
{
  "containerProperties": [{
```

```
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
    }]
  }]
}
```

Example Referenciando uma chave específica dentro de um segredo

Veja a seguir um exemplo de saída de um get-secret-valuecomando que exibe o conteúdo de um segredo junto com o rótulo de preparação da versão e o ID da versão associado a ele.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "VersionId": "871d9eca-18aa-46a9-8785-981dd39ab30c",
    "SecretString": "{\"username1\":\"password1\",\"username2\":\"password2\",
\"username3\":\"password3\"}",
    "VersionStages": [
        "AWSCURRENT"
    ],
    "CreatedDate": 1581968848.921
}
```

Faça referência a uma chave específica de saída anterior em uma definição de contêiner especificando o nome da chave no fim do ARN.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::"
    }]
  }]
}
```

Example Referenciando uma versão secreta específica

A seguir, um exemplo de saída de comando describe-secret exibindo o conteúdo não criptografado de um segredo junto aos metadados de todas as versões do segredo.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "Description": "Example of a secret containing application authorization data.",
    "RotationEnabled": false,
    "LastChangedDate": 1581968848.926,
    "LastAccessedDate": 1581897600.0,
    "Tags": [],
    "VersionIdsToStages": {
        "871d9eca-18aa-46a9-8785-981dd39ab30c": [
            "AWSCURRENT"
        ],
        "9d4cb84b-ad69-40c0-a0ab-cead36b967e8": [
            "AWSPREVIOUS"
        ]
    }
}
```

Faça referência a um rótulo específico de preparação de versão de saída anterior em uma definição de contêiner especificando o nome da chave no fim do ARN.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf::AWSPREVIOUS:"
    }]
  }]
}
```

Faça referência a uma ID da versão específica da saída anterior em uma definição de contêiner especificando o nome da chave no final do ARN.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf::9d4cb84b-ad69-40c0-a0ab-cead36b967e8"
    }]
  }]
```

```
}
```

Example Referenciando uma chave específica e um rótulo de estágio de versão de um segredo

A seguir, exemplo de como fazer referência a uma chave específica dentro de um segredo e rótulo de estágio de versão específico.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1:AWSPREVIOUS:"
    }]
  }]
}
```

Para uma chave específica e ID da versão, use a sintaxe a seguir.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::9d4cb84b-ad69-40c0-a0ab-cead36b967e8"
    }]
  }]
}
```

Injetando dados sigilosos em uma configuração de registro em log

Em sua definição de trabalho, ao especificar uma `logConfiguration`, você poderá fazê-lo `secretOptions` com o nome da opção de registro de log a ser definido no contêiner e ARN completo do segredo Secrets Manager contendo os dados sigilosos a serem apresentados ao contêiner.

A seguir, trecho de uma definição de trabalho mostrando formato, ao referenciar um segredo Secrets Manager.

```
{
  "containerProperties": [{
    "logConfiguration": [{
```

```
      "logDriver": "splunk",
      "options": {
        "splunk-url": "https://cloud.splunk.com:8080"
      },
      "secretOptions": [{
        "name": "splunk-token",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      }]
    }]
  }]
}
```

Criando um AWS Secrets Manager segredo

Você pode usar o console Secrets Manager para criar um segredo para seus dados sigilosos. Para mais informações, consulte Criando um Segredo Básico no AWS Secrets Manager Guia do Usuário.

Para criar um segredo básico

Use Secrets Manager para criar um segredo para seus dados sigilosos.

1. Abra o console Secrets Manager em https://console.aws.amazon.com/secretsmanager/.
2. Escolha Armazenar Novo Segredo.
3. Em Selecionar Tipo de Segredo, selecione Outro Tipo de Segredo.
4. Especifique os detalhes do seu segredo personalizado como pares de Chave e Valor. Por exemplo, você pode especificar uma chave de `UserName` e, em seguida, fornecer o nome do usuário apropriado como valor. Adicione uma segunda chave com o nome de `Password` e texto da senha como valor. Você também pode adicionar entradas para nome de banco de dados, endereço de servidor ou porta TCP. Adicione quantos pares quiser para armazenar as informações obrigatórias.

   Como alternativa, você pode escolher a guia Texto Simples e inserir o valor do segredo como desejar.
5. Escolha a chave de AWS KMS criptografia que você deseja usar para criptografar o texto protegido no segredo. Caso não escolha uma opção, o Secrets Manager verificará a existência de uma chave padrão para a conta e a utilizará, caso exista. Caso não exista uma chave padrão, o Secrets Manager criará uma automaticamente. Você também pode escolher Adicionar Nova

Chave para criar uma chave personalizada KMS específica para esse segredo. Para criar sua própria chave do KMS, você deve ter permissões para criar chaves do KMS na sua conta.

6. Escolha Próximo.
7. Em Nome do Segredo, digite nome e caminho opcionais, como **`production/MyAwesomeAppSecret`** ou **`development/TestSecret`**, e, em seguida, escolha Avançar. Você pode adicionar opcionalmente uma descrição para ajudá-lo a lembrar o objetivo desse segredo mais tarde.

   O nome do segredo deve ter apenas letras ASCII, dígitos, ou qualquer um dos seguintes caracteres: /_+=.@-
8. (Opcional) Nesse momento, você poderá configurar a rotação para o seu segredo. Para esse procedimento, deixe em Desabilitar Rotação Automática e escolha Avançar.

   Para obter informações sobre como configurar a rotação em segredos novos ou existentes, consulte Como girar seus AWS Secrets Manager segredos.
9. Analise suas configurações e, em seguida, escolha Armazenar Segredo para salvar tudo que inseriu como novo segredo no Secrets Manager.

Especificando dados sigilosos usando Systems Manager Parameter Store

Com AWS Batch, você pode injetar dados confidenciais em seus contêineres armazenando seus dados confidenciais nos parâmetros do AWS Systems Manager Parameter Store e, em seguida, referenciando-os na definição do contêiner.

Tópicos

- Considerações para especificar dados confidenciais usando Systems Manager Parameter Store
- Permissões do IAM necessárias para AWS Batch segredos
- Injetando dados confidenciais como variável de ambiente
- Injetando dados confidenciais em uma configuração de log
- Criando um AWS Systems Manager parâmetro do Parameter Store

Considerações para especificar dados confidenciais usando Systems Manager Parameter Store

As informações a seguir devem ser consideradas ao especificar dados confidenciais para contêineres usando parâmetros Systems Manager Parameter Store.

- Esse recurso exige que sua instância de contêiner tenha a versão 1.23.0 ou posterior do agente de contêiner. Recomendamos usar a versão mais recente do atendente de contêiner. Para mais informações sobre como verificar a versão do agente e atualizá-la para a mais recente, consulte Atualizando Atendente de Contêiner Amazon ECS no Guia do Desenvolvedor Amazon Elastic Container Service.
- Os dados confidenciais são injetados no contêiner do seu trabalho o mesmo for iniciado. Caso o segredo ou parâmetro Parameter Store seja posteriormente atualizado ou alternado, o contêiner não receberá o valor atualizado automaticamente. Você deve iniciar uma nova tarefa para forçar seu início com segredos atualizados.

Permissões do IAM necessárias para AWS Batch segredos

Para este atributo, você precisa possuir a função de execução e referenciá-la em sua definição de trabalho. Isso permite que o agente de contêineres do Amazon ECS extraia os AWS Systems Manager recursos necessários. Para ter mais informações, consulte AWS Batch função de execução do IAM.

Para fornecer acesso aos AWS Systems Manager parâmetros do Parameter Store que você cria, adicione manualmente as seguintes permissões como uma política embutida à função de execução. Para mais informações, consulte Adicionando e Removendo Políticas do IAM no Guia de usuário do IAM.

- `ssm:GetParameters`–-Obrigatório se fizer referência a um parâmetro Systems Manager Parameter Store em uma definição de tarefa.
- `secretsmanager:GetSecretValue`–-Obrigatório se fizer referência a um segredo Secrets Manager diretamente, ou se o parâmetro Systems Manager Parameter Store fizer referência a um segredo Secrets Manager em uma definição de tarefa.
- `kms:Decrypt`–-obrigatório somente se o segredo usar uma chave personalizada KMS e não a chave padrão. O ARN da chave personalizada deve ser adicionado como recurso.

O exemplo de política alinhada a seguir adiciona as permissões necessárias:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetParameters",
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:ssm:<region>:<aws_account_id>:parameter/<parameter_name>",
        "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:<secret_name>",
        "arn:aws:kms:<region>:<aws_account_id>:key/<key_id>"
      ]
    }
  ]
}
```

Injetando dados confidenciais como variável de ambiente

Em sua definição de contêiner, especifique `secrets` com o nome da variável de ambiente a ser definida no contêiner e o ARN completo do parâmetro Systems Manager Parameter Store contendo os dados sigilosos a serem apresentados.

A seguir, trecho de uma definição de tarefa mostrando formato ao referenciar um parâmetro Systems Manager Parameter Store. Se o parâmetro Systems Manager Parameter Store existir na mesma Região da tarefa sendo iniciada, você poderá usar o ARN completo ou nome do parâmetro. Se o parâmetro existir em uma Região diferente, o ARN completo deverá ser especificado.

```
{
  "containerProperties": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

Injetando dados confidenciais em uma configuração de log

Em sua definição de contêiner, ao especificar `logConfiguration`, você poderá especificar `secretOptions` com o nome da opção de registro de log a ser definida no contêiner e o ARN completo do parâmetro Systems Manager Parameter Store contendo os dados confidenciais a serem apresentados.

> **Important**
>
> Se o parâmetro Systems Manager Parameter Store existir na mesma Região da tarefa sendo iniciada, você poderá usar o ARN completo ou nome do parâmetro. Se o parâmetro existir em uma Região diferente, o ARN completo deverá ser especificado.

A seguir, trecho de uma definição de tarefa mostrando formato ao referenciar um parâmetro Systems Manager Parameter Store.

```
{
  "containerProperties": [{
    "logConfiguration": [{
      "logDriver": "fluentd",
      "options": {
        "tag": "fluentd demo"
      },
      "secretOptions": [{
        "name": "fluentd-address",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
      }]
    }]
  }]
}
```

Criando um AWS Systems Manager parâmetro do Parameter Store

Você pode usar o AWS Systems Manager console para criar um parâmetro do Systems Manager Parameter Store para seus dados confidenciais. Para mais informações, consulte Demonstração: Crie e Use um Parâmetro em um Comando (Console) no AWS Systems Manager Guia de Usuário.

Para criar um parâmetro Parameter Store

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/.

2. No painel de navegação, escolha Parameter Store, Criar Parâmetro.
3. Para Nome, digite uma hierarquia e nome de parâmetro. Por exemplo, digite `test/database_password`.
4. Para Descrição, digite uma descrição opcional.
5. Em Tipo, escolha Cadeia de caracteres StringList, ou SecureString.

> **Note**
>
> - Se você escolher SecureString, o campo ID da chave KMS será exibido. Se não fornecer o ID da chave KMS, o ARN da chave KMS, um apelido ou apelido ARN, o sistema usará `alias/aws/ssm`. Essa é a chave KMS padrão para o Systems Manager. Para evitar o uso dessa chave, escolha uma personalizada. Para mais informações, consulte Use Parâmetros de Strings Seguros no AWS Systems Manager Guia de uUsuário.
> - Ao criar um parâmetro de string seguro no console usando o parâmetro `key-id` com um apelido personalizado de chave KMS ou apelido ARN, você deve especificar o prefixo `alias/` antes do apelido. A seguir, um exemplo de ARN:
>
>   ```
>   arn:aws:kms:us-east-2:123456789012:alias/MyAliasName
>   ```
>
>   A seguir, um exemplo de apelido:
>
>   ```
>   alias/MyAliasName
>   ```

6. Em Valor, digite um valor. Por exemplo, `MyFirstParameter`. Se você escolher SecureString, o valor será mascarado exatamente como você o inseriu.
7. Escolha Criar Parâmetro.

Autenticação de registro privado para trabalhos

A autenticação de registro privado para uso de trabalhos AWS Secrets Manager permite que você armazene suas credenciais com segurança e, em seguida, faça referência a elas na definição de seu trabalho. Isso fornece uma forma de referenciar imagens de contêiner que existem em registros privados fora dos quais AWS é necessária autenticação em suas definições de trabalho. Esse recurso é suportado por trabalhos hospedados em instâncias do Amazon EC2 e no Fargate.

> **Important**
>
> Se sua definição de trabalho fizer referência a uma imagem armazenada no Amazon ECR, esse tópico não se aplica. Para obter mais informações, consulte Usar imagens do Amazon ECR com o Amazon ECS no Guia do usuário do Amazon Elastic Container Registry.

Para trabalhos hospedados em instâncias do Amazon EC2, esse recurso requer uma versão `1.19.0` ou posterior do agente de contêiner. Recomendamos usar a versão mais recente do atendente de contêiner. Para obter informações sobre como verificar a versão do seu agente e atualizar para a versão mais recente, consulte Atualização do agente de contêiner do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

Para trabalhos hospedados no Fargate, esse recurso requer uma versão da plataforma `1.2.0` ou posterior. Para obter informações, consulte as versões da plataforma AWS Fargate Linux no Amazon Elastic Container Service Developer Guide.

Em sua definição do contêiner, especifique o objeto `repositoryCredentials` com os detalhes do segredo que você criou. O segredo que você menciona pode ser de uma conta diferente Região da AWS ou diferente do trabalho que o usa.

> **Note**
>
> Ao usar a AWS Batch API ou o AWS SDK, se o segredo existir da Região da AWS mesma forma que o trabalho que você está iniciando, você poderá usar o ARN completo ou o nome do segredo. AWS CLI Se o segredo existir em outra conta, o ARN completo do segredo deve ser especificado. Ao usar o AWS Management Console, o ARN completo do segredo deve ser especificado sempre.

Veja a seguir um trecho de uma definição de tarefa que mostra os parâmetros necessários:

```
"containerProperties": [
  {
    "image": "private-repo/private-image",
    "repositoryCredentials": {
      "credentialsParameter":
 "arn:aws:secretsmanager:region:123456789012:secret:secret_name"
    }
```

```
  }
]
```

Permissões do IAM necessárias para a autenticação de registro privado

A função de execução é necessária para usar esse recurso. Isso permite que o agente de contêiner obtenha a imagem do contêiner. Para ter mais informações, consulte AWS Batch função de execução do IAM.

Para fornecer acesso aos segredos que você cria, adicione as seguintes permissões como uma política embutida à função de execução. Para obter mais informações, consulte Adicionar e remover políticas do IAM.

- `secretsmanager:GetSecretValue`
- `kms:Decrypt`: exigido somente se a chave usar uma chave do KMS personalizada e não a chave padrão. O nome do recurso da Amazon (ARN) da chave personalizada deve ser adicionado como um recurso.

Veja a seguir um exemplo de política em linha que adiciona as permissões.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region:123456789012:secret:secret_name",
        "arn:aws:kms:region:123456789012:key/key_id"
      ]
    }
  ]
}
```

Uso da autenticação de registro privado

Para criar um segredo básico

Use AWS Secrets Manager para criar um segredo para suas credenciais de registro privado.

1. Abra o AWS Secrets Manager console em https://console.aws.amazon.com/secretsmanager/.
2. Escolha Armazenar Novo Segredo.
3. Em Selecionar Tipo de Segredo, selecione Outro Tipo de Segredo.
4. Selecione Plaintext (Texto simples) e insira suas credenciais de registro privado usando o seguinte formato:

   ```
   {
     "username" : "privateRegistryUsername",
     "password" : "privateRegistryPassword"
   }
   ```

5. Escolha Próximo.
6. Em Secret name (Nome de segredo), insira um nome e caminho opcionais, como **`production/MyAwesomeAppSecret`** ou **`development/TestSecret`** e escolha Next (Avançar). Você pode adicionar opcionalmente uma descrição para ajudá-lo a lembrar o objetivo desse segredo mais tarde.

   O nome do segredo deve ter somente letras ASCII, números ou qualquer um dos seguintes caracteres: `/_+=.@-`.
7. (Opcional) Nesse momento, será possível configurar a rotação para o segredo. Para esse procedimento, deixe em Desabilitar Rotação Automática e escolha Avançar.

   Para obter instruções sobre como configurar a rotação em segredos novos ou existentes, consulte Como girar seus AWS Secrets Manager segredos.
8. Reveja suas configurações e, em seguida, escolha Store secret (Armazenar segredo) para salvar tudo o que você inseriu como novo segredo no Secrets Manager.

Registre uma definição de trabalho e, em Registro privado, ative a autenticação de registro privado. Em seguida, em ARN ou nome do Secrets Manager, insira o nome do recurso da Amazon (ARN) do segredo. Para ter mais informações, consulte Permissões do IAM necessárias para a autenticação de registro privado.

Volumes Amazon EFS

O Amazon Elastic File System (Amazon EFS) oferece armazenamento simples e escalável de arquivos para uso com suas tarefas AWS Batch. Com Amazon EFS, a capacidade de armazenamento é elástica. Ela é escalada automaticamente a medida que arquivos são adicionados e removidos. Suas aplicações podem ter o armazenamento de que precisarem, quando precisarem.

Use os sistemas de arquivamento Amazon EFS com AWS Batch para exportar dados do sistema de arquivamento em sua frota de instâncias de contêiner. Assim, seus trabalhos têm acesso ao mesmo armazenamento persistente. Contudo, você deve configurar a AMI da instância de contêiner para montar o sistema de arquivos Amazon EFS antes que o daemon Docker seja iniciado. Além disso, as definições de tarefa devem referenciar montagens de volume na instância de contêiner para poderem usa o sistema de arquivamento. As seções a seguir lhe ajudarão nesse início de uso do Amazon EFS com AWS Batch.

Considerações de volume Amazon EFS

As seguintes informações devem ser consideradas ao usar volumes Amazon EFS:

- Para trabalhos usando recursos EC2, o suporte ao sistema de arquivamento Amazon EFS foi adicionado como visualização prévia pública com a versão AMI otimizada Amazon ECS `20191212` com a versão 1.35.0 do atendente de contêiner. No entanto, o suporte ao sistema de arquivamento Amazon EFS tornou-se disponível para o público com a versão AMI otimizada Amazon ECS `20200319` com a versão 1.38.0 do atendente de contêiner, que incluía atributos de ponto de acesso Amazon EFS e autorização IAM. Recomendamos o uso da versão otimizada AMI Amazon ECS `20200319` ou superior para aproveitamento desses recursos. Para mais informações, consulte Versões AMI otimizadas Amazon ECS no Guia de Desenvolvedor Amazon Elastic Container Service.

  > **Note**
  >
  > Caso crie sua própria AMI, você deverá usar o atendente de contêiner 1.38.0 ou superior, `ecs-init` versão 1.38.0-1 ou superior e executar os comandos a seguir em sua instância Amazon EC2. Tudo isso para habilitar o plug-in de volume Amazon ECS. Os comandos dependem do uso do Amazon Linux 2 ou Amazon Linux como imagem base.
  >
  > Amazon Linux 2
  >
  > ```
  > $ yum install amazon-efs-utils
  > ```

```
systemctl enable --now amazon-ecs-volume-plugin
```

Amazon Linux

```
$ yum install amazon-efs-utils
sudo shutdown -r now
```

- Para tarefas utilizando recursos Fargate, o suporte de sistema de arquivamento Amazon EFS foi adicionado ao utilizar a versão 1.4.0 ou superior da plataforma. Para mais informações, consulte AWSVersões de Plataforma Fargate no Guia de Desenvolvedor Amazon Elastic Container Service.
- Ao especificar volumes Amazon EFS em tarefas usando recursos Fargate, o mesmo cria um contêiner supervisor responsável pelo gerenciamento do volume Amazon EFS. O contêiner supervisor usa uma pequena quantidade de memória da tarefa. O contêiner supervisor fica visível ao consultar a versão 4 do endpoint de metadados de tarefa. Para mais informações, consulte Endpoint de Metadados de Tarefas Versão 4 do Guia de Usuário Amazon Elastic Container Service para AWS Fargate.

Usando pontos de acesso Amazon EFS

Os pontos de acesso Amazon EFS são pontos de entrada específicos da aplicação em um sistema de arquivamento EFS que ajudam a gerenciar o acesso a aplicações a conjuntos de dados compartilhados. Para mais informações sobre pontos de acesso Amazon EFS e como controlar o acesso a eles, consulte Trabalhando com Pontos de Acesso Amazon EFS no Guia de Usuário Amazon Elastic File System.

Os pontos de acesso podem impor uma identidade de usuário, inclusive grupos POSIX do usuário, para todas as solicitações do sistema de arquivamento feitas por meio do ponto de acesso. Os pontos de acesso também podem impor um diretório raiz diferente para o sistema de arquivamento fazendo com que clientes só possam acessar dados no diretório especificado ou em seus subdiretórios.

> **Note**
>
> Ao criar um ponto de acesso EFS, você especifica um caminho no sistema de arquivamento para servir como diretório raiz. Ao fazer referência ao sistema de arquivos EFS com uma ID

de ponto de acesso na definição de trabalho AWS Batch, o diretório raiz deve ser omitido ou definido para / Isso impõe o caminho definido no ponto de acesso EFS.

É possível usar um perfil do IAM de uma tarefa AWS Batch para forçar aplicativos específicos a usarem um ponto de acesso específico. Ao combinar políticas do IAM com pontos de acesso, é possível oferecer acesso fácil e seguro a conjuntos de dados específicos para seus aplicativos. Esse atributo usa perfis do IAM Amazon ECS para funcionalidade de tarefa. Para mais informações, consulte Funções do IAM para Tarefas no Guia de Desenvolvedor Amazon Elastic Container Service.

Especificando um sistema de arquivamento Amazon EFS na definição de tarefa

Para usar volumes do sistema de arquivamento Amazon EFS em seus contêineres, você deve especificar as configurações de volume e ponto de montagem na definição de tarefa. O trecho JSON de definição de tarefa a seguir mostra a sintaxe para os objetos `volumes` e `mountPoints` de um contêiner:

```
{
    "containerProperties": [
        {
            "image": "amazonlinux:2",
            "command": [
                "ls",
                "-la",
                "/mount/efs"
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEfsVolume",
                    "containerPath": "/mount/efs",
                    "readOnly": true
                }
            ],
            "volumes": [
                {
                    "name": "myEfsVolume",
                    "efsVolumeConfiguration": {
                        "fileSystemId": "fs-12345678",
                        "rootDirectory": "/path/to/my/data",
                        "transitEncryption": "ENABLED",
```

```
                        "transitEncryptionPort": integer,
                        "authorizationConfig": {
                            "accessPointId": "fsap-1234567890abcdef1",
                            "iam": "ENABLED"
                        }
                    }
                }
            ]
        }
    ]
}
```

`efsVolumeConfiguration`

Tipo: objeto

Obrigatório: não

Esse parâmetro é especificado quando volumes Amazon EFS são usados.

`fileSystemId`

Tipo: string

Obrigatório: sim

A ID do sistema de arquivamento Amazon EFS a ser usada.

`rootDirectory`

Tipo: string

Obrigatório: não

O diretório dentro do sistema de arquivamento Amazon EFS a ser montado como diretório raiz dentro do host. Caso esse parâmetro seja omitido, a raiz do volume Amazon EFS será usada. Especificar `/` tem o mesmo efeito que omitir esse parâmetro. Ele pode ter até 4.096 caracteres de comprimento.

> **Important**
>
> Se um ponto de acesso EFS for especificado em `authorizationConfig`, o parâmetro do diretório raiz deverá ser omitido ou definido para `/`. Isso impõe o caminho definido no ponto de acesso EFS.

`transitEncryption`

Tipo: string

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: não

Determina se a criptografia deve ou não ser habilitada para dados Amazon EFS em trânsito entre host AWS Batch e servidor Amazon EFS. A criptografia de trânsito deverá ser habilitada caso a autorização IAM Amazon EFS for usada. Se o parâmetro for omitido, o valor padrão `DISABLED` será usado. Para mais informações, consulte Criptografando Dados em Trânsito no Guia de Usuário Amazon Elastic File System.

`transitEncryptionPort`

Tipo: inteiro

Obrigatório: não

A porta a ser usada ao enviar dados criptografados do host AWS Batch para o servidor Amazon EFS. Caso não especifique uma porta de criptografia em trânsito, a estratégia de seleção de porta usada pelo assistente de montagem Amazon EFS será utilizada. O valor precisar estar compreendido entre 0 e 65.535. Para mais informações, consulte Auxiliar de Montagem EFS no Guia de Usuário Amazon Elastic File System.

`authorizationConfig`

Tipo: objeto

Obrigatório: não

Detalhes de configuração de autorização do sistema de arquivamento Amazon EFS.

`accessPointId`

Tipo: string

Obrigatório: não

ID do ponto de acesso a ser usado. Se um ponto de acesso for especificado, o valor do diretório raiz em `efsVolumeConfiguration` deverá ser omitido, ou definido como `/`. Isso impõe o caminho definido no ponto de acesso EFS. Se um ponto de acesso for usado, a criptografia em trânsito deverá ser habilitada em `EFSVolumeConfiguration`. Para

mais informações, consulte Trabalhando com Pontos de Acesso Amazon EFS no Guia de Usuário Amazon Elastic File System.

`iam`

Tipo: string

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: não

Determina se é necessário utilizar o perfil do IAM de tarefa AWS Batch estabelecido em uma definição de tarefa ao montar o sistema de arquivamento Amazon EFS. Em caso positivo, a criptografia de trânsito deve estar habilitada em `EFSVolumeConfiguration`. Se o parâmetro for omitido, o valor padrão `DISABLED` será usado. Para mais informações sobre perfis do IAM de execução, consulte AWS Batch função de execução do IAM.

Definições de trabalho de exemplo

As definições de tarefas de exemplo a seguir ilustram como usar padrões comuns, como variáveis de ambiente, substituição de parâmetros e montagens do volume.

Use variáveis de ambiente

O exemplo de definição de tarefa a seguir usa variáveis de ambiente para especificar um tipo de arquivo e o URL do Amazon S3. Esse exemplo específico é da postagem do blog de computação Criar uma tarefa simples "Buscar e executar" do AWS Batch. O script `fetch_and_run.sh` que é descrito na postagem do blog usa essas variáveis do ambiente para fazer download do script `myjob.sh` do S3 e declarar seu tipo de arquivo.

Embora o comando e as variáveis do ambiente sejam criptografados na definição de tarefa neste exemplo, você pode especificar as substituições de comando e variável de ambiente para tornar a definição de tarefa mais versátil.

```
{
    "jobDefinitionName": "fetch_and_run",
    "type": "container",
    "containerProperties": {
        "image": "123456789012.dkr.ecr.us-east-1.amazonaws.com/fetch_and_run",
        "resourceRequirements": [
```

```
            {
                "type": "MEMORY",
                "value": "2000"
            },
            {
                "type": "VCPU",
                "value": "2"
            }
        ],
        "command": [
            "myjob.sh",
            "60"
        ],
        "jobRoleArn": "arn:aws:iam::123456789012:role/AWSBatchS3ReadOnly",
        "environment": [
            {
                "name": "BATCH_FILE_S3_URL",
                "value": "s3://my-batch-scripts/myjob.sh"
            },
            {
                "name": "BATCH_FILE_TYPE",
                "value": "script"
            }
        ],
        "user": "nobody"
    }
}
```

Como usar substituição de parâmetros

O exemplo de definição de tarefa a seguir ilustra como permitir a substituição de parâmetros e definir valores padrão.

As declarações `Ref::` na seção `command` são usadas para definir marcadores para a substituição de parâmetros. Quando envia um trabalho com essa definição de tarefa, você especifica as substituições de parâmetro para preencher esses valores, como `inputfile` e `outputfile`. A seção `parameters` a seguir define um padrão para o `codec`, mas você pode substituir esse parâmetro conforme necessário.

Para obter mais informações, consulte Parâmetros.

```
{
```

```
    "jobDefinitionName": "ffmpeg_parameters",
    "type": "container",
    "parameters": {"codec": "mp4"},
    "containerProperties": {
        "image": "my_repo/ffmpeg",
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "2000"
            },
            {
                "type": "VCPU",
                "value": "2"
            }
        ],
        "command": [
            "ffmpeg",
            "-i",
            "Ref::inputfile",
            "-c",
            "Ref::codec",
            "-o",
            "Ref::outputfile"
        ],
        "jobRoleArn": "arn:aws:iam::123456789012:role/ECSTask-S3FullAccess",
        "user": "nobody"
    }
}
```

Testar funcionalidade de GPU

O exemplo de definição de tarefa a seguir testa se a AMI de workload de GPU descrita em Usar uma AMI de workload de GPU está configurada corretamente. Este exemplo de definição de trabalho executa o exemplo do classificador de MNIST profundo do Tensorflow do GitHub.

```
{
    "containerProperties": {
        "image": "tensorflow/tensorflow:1.8.0-devel-gpu",
        "resourceRequirements": [
            {
                "type": "MEMORY",
                "value": "32000"
            },
```

```
            {
                "type": "VCPU",
                "value": "8"
            }
        ],
        "command": [
            "sh",
            "-c",
            "cd /tensorflow/tensorflow/examples/tutorials/mnist; python mnist_deep.py"
        ]
    },
    "type": "container",
    "jobDefinitionName": "tensorflow_mnist_deep"
}
```

É possível criar um arquivo com o texto JSON anterior chamado `tensorflow_mnist_deep.json` e registrar uma definição de trabalho do AWS Batch com o seguinte comando:

```
aws batch register-job-definition --cli-input-json file://tensorflow_mnist_deep.json
```

Trabalho em paralelo de vários nós

O exemplo de definição de trabalho ilustra um trabalho em paralelo de vários nós. Para obter mais detalhes, consulte Criar um fluxo de trabalho de dinâmica molecular fortemente acoplado com trabalhos paralelos de vários nós no AWS Batch no blog AWS Compute.

```
{
  "jobDefinitionName": "gromacs-jobdef",
  "jobDefinitionArn": "arn:aws:batch:us-east-2:123456789012:job-definition/gromacs-
jobdef:1",
  "revision": 6,
  "status": "ACTIVE",
  "type": "multinode",
  "parameters": {},
  "nodeProperties": {
    "numNodes": 2,
    "mainNode": 0,
    "nodeRangeProperties": [
      {
        "targetNodes": "0:1",
        "container": {
          "image": "123456789012.dkr.ecr.us-east-2.amazonaws.com/gromacs_mpi:latest",
```

```
          "resourceRequirements": [
              {
                  "type": "MEMORY",
                  "value": "24000"
              },
              {
                  "type": "VCPU",
                  "value": "8"
              }
          ],
          "command": [],
          "jobRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
          "ulimits": [],
          "instanceType": "p3.2xlarge"
        }
      }
    ]
  }
}
```

Filas de tarefas

Tarefas são enviadas para uma fila, onde permanecem até que sejam programadas para execução em um ambiente de computação. Uma AWS conta pode ter várias filas de tarefas. Por exemplo, você pode criar uma fila com instâncias sob demanda Amazon EC2 para tarefas de alta prioridade e outra com instâncias spot Amazon EC2 para tarefas de menor prioridade. As filas de tarefas têm uma prioridade usada pelo programador para determinar quais tarefas em quais filas devem ser avaliadas para execução em primeiro lugar.

Tópicos

Como criar uma fila de tarefas

Antes de enviar trabalhos em AWS Batch, você precisa criar uma fila. Ao criar uma fila de tarefas, você associa um ou mais ambientes de computação à fila e lhes atribui uma ordem de preferência.

Você também pode definir uma prioridade para a fila de tarefas para determinar a ordem na qual o programador AWS Batch irá posicionar as tarefas. Isso quer dizer que, se um ambiente de computação for associado a mais de uma fila de tarefas, aquela com prioridade mais alta terá preferência.

Como criar uma fila de tarefas Fargate

Para criar uma fila de tarefas

1. Abra o console AWS Batch em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione Região da AWS a ser usada.
3. No painel de navegação, escolha Filas de Tarefas.
4. Escolha Criar.
5. Para Tipo de Orquestração, escolha Fargate.
6. Para Nome, insira um nome exclusivo para a sua fila de tarefas. Ele pode ter até 128 caracteres e conter letras minúsculas, maiúsculas, números e sublinhados (_).

7. Em Prioridade, insira um valor inteiro para a prioridade da fila de tarefas. As filas com prioridade mais alta serão executadas antes das filas com prioridade mais baixa associadas ao mesmo ambiente de computação. A prioridade é determinada em ordem decrescente. Por exemplo, uma fila de tarefas com valor de prioridade 10 tem preferência de programação em relação a uma fila com valor de prioridade 1.
8. (Opcional) Para Política de Agendamento do Nome do Recurso da Amazon (ARN), escolha uma política de agendamento existente.
9. Em Ambientes de Computação Conectados, selecione um ou mais ambientes de computação da lista para associar à lista de tarefas. Selecione ambientes de computação na ordem que quiser que a fila tente posicionar a fila de tarefas. O programador de tarefas usará a ordem do ambiente de computação para determinar qual ambiente de computação deverá iniciar uma tarefa específica. Antes de poder associá-los a uma fila de tarefas, os ambientes de computação devem estar no estado `VALID`. Você pode associar até três ambientes de computação a uma fila de tarefas.

   > **Note**
   >
   > Todos os ambientes de computação associados a uma fila devem compartilhar o mesmo modelo de provisionamento. AWS Batch não é compatível com a mistura de modelos de provisionamento em uma única fila de tarefas.

10. Em Computar Ordem de Provisionamento, escolha as setas para cima e para baixo para configurar a ordem desejada.
11. Escolha Criar Fila de Tarefas para concluir e criar sua fila.

Como criar uma fila de tarefas Amazon EC2

Para criar uma fila de tarefas Amazon EC2

1. Abra o console AWS Batch em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione Região da AWS a ser usada.
3. No painel de navegação, escolha Filas de Tarefas.
4. Escolha Criar.
5. Em Tipo de orquestração, escolha Amazon Elastic Compute Cloud (Amazon EC2).

6. Em Nome, insira um nome exclusivo para a fila de tarefas. Ele pode ter até 128 caracteres e conter letras minúsculas, maiúsculas, números e sublinhados (_).
7. Em Prioridade, insira um valor inteiro para a prioridade da fila de tarefas. As filas com prioridade mais alta serão executadas antes das filas com prioridade mais baixa associadas ao mesmo ambiente de computação. A prioridade é determinada em ordem decrescente. Por exemplo, uma fila de tarefas com valor de prioridade 10 tem preferência de programação em relação a uma fila com valor de prioridade 1.
8. (Opcional) Para Política de Agendamento do Nome do Recurso da Amazon (ARN), escolha uma política de agendamento existente.
9. Em Ambientes de Computação Conectados, selecione um ou mais ambientes de computação da lista para associar à lista de tarefas. Selecione ambientes de computação na ordem que quiser que a fila tente posicionar a fila de tarefas. O programador de trabalhos usará a ordem do ambiente de computação para determinar o ambiente de computação responsável por iniciar uma tarefa específica. Antes de poder associá-los a uma fila de tarefas, os ambientes de computação devem estar no estado `VALID`. Você pode associar até três ambientes de computação a uma fila de tarefas. Caso não possua um ambiente de computação existente, escolha Criar Ambiente Computacional

   > **Note**
   >
   > Todos os ambientes de computação associados a uma fila devem compartilhar o mesmo modelo de provisionamento. AWS Batch não é compatível com a mistura de modelos de provisionamento em uma única fila de tarefas.

10. Em Computar Ordem de Provisionamento, escolha as setas para cima e para baixo para configurar a ordem desejada.
11. Escolha Criar Fila de Tarefas para concluir e criar sua fila.

Como criar uma fila de trabalhos Amazon EKS

Para criar uma fila de trabalhos Amazon EKS

1. Abra o console AWS Batch em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione Região da AWS a ser usada.
3. No painel de navegação, escolha Filas de Tarefas.
4. Escolha Criar.

5. Em Tipo de Orquestração, escolha Amazon Elastic Kubernetes Service (Amazon EKS).
6. Em Nome, insira um nome exclusivo para a fila de tarefas. Ele pode ter até 128 caracteres e conter letras minúsculas, maiúsculas, números e sublinhados (_).
7. Para Prioridade, insira um valor inteiro para a prioridade da fila de tarefas. As filas com prioridade mais alta serão executadas antes das filas com prioridade mais baixa associadas ao mesmo ambiente de computação. A prioridade é determinada em ordem decrescente. Por exemplo, uma fila de tarefas com valor de prioridade 10 tem preferência de programação em relação a uma fila com valor de prioridade 1.
8. (Opcional) Para Política de Agendamento do Nome do Recurso da Amazon (ARN), escolha uma política de agendamento existente.
9. Em Ambientes de Computação Conectados, selecione um ou mais ambientes de computação da lista para associar à lista de tarefas. Selecione ambientes de computação na ordem que quiser que a fila tente posicionar a fila de tarefas. O programador de trabalhos usará a ordem do ambiente de computação para determinar o ambiente de computação responsável por iniciar uma tarefa específica. Antes de poder associá-los a uma fila de tarefas, os ambientes de computação devem estar no estado `VALID`. Você pode associar até três ambientes de computação a uma fila de tarefas.

   > **Note**
   >
   > Todos os ambientes de computação associados a uma fila devem compartilhar o mesmo modelo de provisionamento. AWS Batch não é compatível com a mistura de modelos de provisionamento em uma única fila de tarefas.

   > **Note**
   >
   > Todos os ambientes de computação associados a uma fila de trabalhos devem compartilhar a mesma arquitetura. AWS Batch não oferece suporte à mistura de tipos de arquitetura de ambiente de computação em uma única fila de tarefas.

10. Em Computar Ordem de Provisionamento, escolha as setas para cima e para baixo para configurar a ordem desejada.
11. Escolha Criar Fila de Tarefas para concluir e criar sua fila.

Modelo de fila de trabalhos

O seguinte é um modelo de fila de tarefas vazio. Você pode usar esse modelo para criar sua fila de trabalhos. Em seguida, você pode salvar essa fila de trabalhos em um arquivo e usá-la com a AWS CLI `--cli-input-json` opção. Para obter mais informações sobre esses parâmetros, consulte CreateJobQueuea Referência AWS Batch da API.

```
{
   "computeEnvironmentOrder": [
      {
         "computeEnvironment": "",
         "order": 0
      }
   ],
   "jobQueueName": "",
   "jobStateTimeLimitActions": [
      {
         "state": "RUNNABLE",
         "action": "CANCEL",
   "maxTimeSeconds": 0,
         "reason": ""

      }
   ],
   "priority": 0,
   "schedulingPolicyArn": "",
   "state": "ENABLED",
   "tags": {
      "KeyName": ""
   }
}
```

> **Note**
>
> Você pode gerar o modelo de fila de tarefas anterior com o comando a seguir AWS CLI .
>
> ```
> $ aws batch create-job-queue --generate-cli-skeleton
> ```

Parâmetros da fila de trabalhos

As filas de trabalhos são divididas em quatro componentes básicos: nome, estado, prioridade e ordem do ambiente computacional. Esta seção discute os componentes associados a esses componentes.

Tópicos

Nome da fila de trabalhos

`jobQueueName`

O nome da fila de trabalhos. São permitidos até 128 letras (caixa alta e baixa), números e sublinhados.

Tipo: sequência

Obrigatório: Sim

Ações de limite de tempo do estado da fila de trabalhos

`jobStateTimeLimitActions`

O conjunto de ações que são AWS Batch executadas em trabalhos que permanecem no início da fila de trabalhos no estado especificado por mais tempo do que os horários especificados. AWS Batch executará cada ação depois de `maxTimeSeconds` ter passado. (Observação: o valor mínimo para `maxTimeSeconds` é 600 (10 minutos) e seu valor máximo é 86.400 (24 horas).)

Tipo: matriz de objetos `JobStateTimeLimitActions`

Obrigatório: Não

Prioridade

`priority`

A prioridade da fila de trabalhos. As filas de trabalhos com uma prioridade mais alta (ou um valor inteiro mais alto para o parâmetro `priority`) são avaliadas primeiro quando associadas ao mesmo ambiente de computação. A prioridade é determinada em ordem decrescente, por exemplo, uma fila de trabalhos com um valor de prioridade 10 tem preferência de programação com relação a uma fila de trabalhos com um valor de prioridade 1. Todos os ambientes computacionais devem ser Amazon EC2 `EC2` (`SPOTou`) ou Fargate (ou). `FARGATE` `FARGATE_SPOT` Os ambientes computacionais Amazon EC2 e Fargate não podem ser misturados.

Tipo: inteiro

Obrigatório: sim

Política de agendamento

`schedulingPolicyArn`

O Nome de recurso da Amazon (ARN) da política de agendamento da fila de trabalhos. Filas de trabalho que não têm uma política de agendamento são programadas em um modelo FIFO (por ordem de chegada). Depois que uma fila de trabalhos tiver uma política de agendamento, ela poderá ser substituída, mas não poderá ser removida. Uma fila de trabalhos sem uma política de agendamento é agendada como fila de trabalhos FIFO (por ordem de chegada) e não pode ter uma política de agendamento adicionada. Filas de trabalhos com uma política de agendamento podem ter, no máximo, 500 identificadores ativos de compartilhamento justo. Quando o limite é atingido, os envios de trabalhos que adicionam um novo identificador de compartilhamento justo falham.

Tipo: string

Obrigatório: não

State

`state`

O estado da fila de trabalhos. Se o estado da fila de trabalhos for `ENABLED` (o valor padrão), é possível aceitar trabalhos. Se o estado da fila de trabalhos for `DISABLED`, novos trabalhos não poderão ser adicionados à fila, mas os trabalhos que já estão na fila poderão ser concluídos.

Tipo: sequência

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: Não

Ordem do ambiente de computação

`computeEnvironmentOrder`

O conjunto de ambientes de computação mapeados para uma fila de trabalhos e sua ordem em relação aos outros. O programador de trabalhos usa esse parâmetro para determinar qual ambiente de computação deve executar um trabalho específico. Os ambientes de computação devem estar no estado `VALID` antes de serem associados a uma fila de trabalhos. Você pode associar até três ambientes de computação a uma fila de tarefas. Todos os ambientes computacionais devem ser Amazon EC2 `EC2` (`SPOTou`) ou Fargate (ou). `FARGATE` `FARGATE_SPOT` Os ambientes computacionais Amazon EC2 e Fargate não podem ser misturados.

> **Note**
>
> Todos os ambientes computacionais associados a uma fila de trabalhos devem compartilhar a mesma arquitetura. AWS Batch não oferece suporte à mistura de tipos de arquitetura de ambiente computacional em uma única fila de trabalhos.

Tipo: matriz de objetos ComputeEnvironmentOrder

Obrigatório: Sim

`computeEnvironment`

O Nome de recurso da Amazon (ARN) do ambiente de computação.

Tipo: sequência

Obrigatório: Sim

`order`

A ordem do ambiente de computação. Os ambientes de computação são testados em ordem ascendente. Por exemplo, se dois ambientes de computação estiverem associados a uma fila de trabalhos, o ambiente de computação com um valor inteiro `order` mais baixo será tentado para colocação primeiro.

Tags

`tags`

Tags de pares de chave-valor a serem associadas à fila de trabalhos. Para ter mais informações, consulte Marcando seus Recursos AWS Batch.

Tipo: Mapa de string para string

Obrigatório: Não

Agendamento de trabalhos

O programador do AWS Batch avalia quando, onde e como executar trabalhos que são enviados a uma fila de trabalhos. Se você não especificar uma política de agendamento ao criar uma fila de trabalhos, o agendador de trabalhos do AWS Batch usará como padrão uma estratégia FIFO (por ordem de entrada). Uma estratégia FIFO (por ordem de chegada) pode fazer com que trabalhos importantes fiquem “presos” atrás de trabalhos que foram enviados anteriormente. Ao especificar uma política de agendamento diferente, você pode alocar recursos de computação de acordo com suas necessidades específicas.

> **Note**
>
> Se você quiser programar a ordem específica em que os trabalhos são executados, use o parâmetro `dependsOn` em SubmitJob para especificar as dependências de cada trabalho.

Se você criar uma política de agendamento e anexá-la a uma fila de trabalhos, o agendamento de compartilhamento justo será ativado. Se a fila de trabalhos tiver uma política de agendamento, a política de agendamento determinará a ordem em que os trabalhos serão executados. Para obter mais informações, consulte Políticas de agendamento.

Identificadores de compartilhamento

Você pode usar identificadores de compartilhamento para marcar trabalhos e diferenciar usuários e workloads. O agendador do AWS Batch rastreia o uso de cada identificador de compartilhamento justo usando a fórmula *`(T * weightFactor)`*, onde *`T`* é o uso da vCPU ao longo do tempo. O agendador seleciona trabalhos com o menor uso do identificador de compartilhamento. Você pode usar um identificador de compartilhamento justo sem substituí-lo.

> **Note**
>
> Os identificadores de compartilhamento são exclusivos em uma fila de trabalhos e não são agregados nas filas de trabalhos.

Você pode definir a prioridade do agendamento para configurar a ordem em que os trabalhos são executados em um identificador de compartilhamento. Trabalhos com prioridade de agendamento

mais alta são agendados primeiro. Se você não especificar uma política de agendamento, todos os trabalhos enviados à fila de trabalhos serão agendados na ordem FIFO. Ao enviar um trabalho, você não pode especificar um identificador de compartilhamento ou a prioridade de agendamento.

> **Note**
>
> Os recursos de computação anexados são alocados igualmente entre todos os identificadores de compartilhamento, a menos que sejam explicitamente substituídos.

Programação de compartilhamento justo

O agendamento de compartilhamento justo fornece um conjunto de controles para ajudar a agendar trabalhos.

> **Note**
>
> Para obter mais informações sobre parâmetros de política de agendamento, consulte Parâmetros de política de agendamento.

- Segundos de degradação de compartilhamento – o período de tempo (em segundos) que o agendador do AWS Batch usa para calcular uma porcentagem de compartilhamento justo para cada identificador de compartilhamento justo. Um valor zero indica que somente o uso atual é medido. Um tempo de degradação mais longo dá mais peso ao tempo.

  > **Note**
  >
  > O período de tempo de degradação é calculado como: *`shareDecaySeconds + OrderMinutes`*, onde *`OrderMinutes`* é o tempo na ordem em minutos.

- Reserva de computação – impede que trabalhos em um único identificador de compartilhamento usem todos os recursos anexados à fila de trabalhos. A proporção reservada é *`computeReservation/100)^ActiveFairShares`*, onde ActiveFairShares é o número de identificadores de compartilhamento justo ativos.

> **Note**
>
> Se um identificador de compartilhamento tiver trabalhos em um estado `SUBMITTED`, `PENDING`, `RUNNABLE`,`STARTING` ou `RUNNING`, ele será considerado um identificador de compartilhamento ativo. Após o período de degradação expirar, um identificador de compartilhamento é considerado inativo.

- Fator de ponderação – o fator de ponderação para o identificador de compartilhamento justo. O valor padrão é 1. Um valor menor permite que trabalhos do identificador de compartilhamento sejam executados ou forneça runtime adicional ao identificador de compartilhamento. Por exemplo, trabalhos que usam um identificador de compartilhamento com um fator de ponderação de 0,125 (1/8) obtêm oito vezes os recursos de computação dos trabalhos que usam um identificador de compartilhamento com um fator de ponderação 1.

  > **Note**
  >
  > Você só precisa definir esse atributo quando precisar atualizar o fator de peso padrão de 1.

Ambiente de computação

As filas de trabalho são mapeadas para um ou mais ambientes de computação. Os ambientes de computação contêm as instâncias de contêiner do Amazon ECS que são usadas para executar trabalhos em lote em contêineres. Um ambiente de computação específico também podem ser mapeado para uma ou mais filas de trabalho. Em uma fila de trabalho, cada ambiente de computação associado tem um pedido que é usado pelo programador para determinar onde os trabalhos que estão prontos para serem executados serão executados. Se o primeiro ambiente de computação tiver um status de `VALID` e recursos disponíveis, o trabalho será agendado para uma instância de contêiner nesse ambiente de computação. Se o primeiro ambiente de computação tiver um status de `INVALID` ou não puder fornecer um recurso de computação adequado, o agendador tentará executar o trabalho no próximo ambiente de computação.

Tópicos

Ambientes de computação gerenciados

Você pode usar um ambiente computacional gerenciado para AWS Batch gerenciar a capacidade e os tipos de instância dos recursos computacionais dentro do ambiente. Isso se baseia nas especificações do recurso de computação que você define ao criar o ambiente de computação. Você pode optar por usar instâncias sob demanda do Amazon EC2 e instâncias spot do Amazon EC2.

Como alternativa, você pode usar a capacidade do Fargate e do Fargate Spot em seu ambiente de computação gerenciado. Ao usar instâncias spot, você pode, como opção, definir um preço máximo. Assim, as instâncias spot são iniciadas apenas quando o preço da instância spot estiver abaixo de uma porcentagem especificada do preço sob demanda.

> **Important**
>
> As instâncias Fargate Spot não são suportadas no. Windows containers on AWS Fargate Uma fila de trabalhos será bloqueada se um FargateWindows trabalho for enviado a uma fila de trabalhos que usa somente ambientes computacionais Fargate Spot.

Ambientes de computação gerenciados iniciam instâncias do Amazon EC2 na VPC e nas sub-redes que você especifica e, em seguida, as registram em um cluster do Amazon ECS. As instâncias do Amazon EC2 precisam de acesso de rede externo para se comunicar com o endpoint de serviço do Amazon ECS. Algumas sub-redes não fornecem às instâncias do Amazon EC2 endereços IP públicos. Se suas instâncias do Amazon EC2 não tiverem um endereço IP público, elas deverão usar a conversão de endereço de rede (NAT) para obter esse acesso. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC. Para obter mais informações sobre como criar uma VPC, consulte Criando uma Nuvem Privada Virtual .

Por padrão, os ambientes computacionais AWS Batch gerenciados usam uma versão recente e aprovada da AMI otimizada do Amazon ECS para recursos computacionais. No entanto, você pode querer criar sua própria AMI a ser usada para seus ambientes de computação gerenciados por vários motivos. Para ter mais informações, consulte Recursos de computação de AMIs.

> **Note**
>
> AWS Batch não atualiza automaticamente as AMIs em um ambiente computacional após sua criação. Por exemplo, ele não atualiza as AMIs no seu ambiente de computação quando uma versão mais recente da AMI otimizada do Amazon ECS é liberada. Você é responsável pelo gerenciamento do sistema operacional convidado. Isso inclui quaisquer atualizações e patches de segurança. Você também é responsável por quaisquer outros utilitários ou aplicativos de software que instalar nos recursos de computação. Há duas maneiras de usar uma nova AMI para seus AWS Batch trabalhos. O método original é concluir as seguintes etapas:
>
> 1. Crie um novo ambiente de computação com a nova AMI.

2. Adicione o ambiente de computação a uma fila de trabalhos existente.
3. Remova o antigo ambiente de computação da fila de trabalhos.
4. Exclua o ambiente de computação anterior.

Em abril de 2022, AWS Batch foi adicionado suporte aprimorado para atualização de ambientes computacionais. Para ter mais informações, consulte Criação de um ambiente de computação. Para utilizar a atualização aprimorada de ambientes de computação para atualizar AMIs, siga estas regras:

- Não defina o parâmetro da função de serviço (`serviceRole`) nem o defina como a função AWSServiceRoleForBatchvinculada ao serviço.
- Defina o parâmetro da estratégia de alocação (`allocationStrategy`) como `BEST_FIT_PROGRESSIVE`, `SPOT_CAPACITY_OPTIMIZED` ou `SPOT_PRICE_CAPACITY_OPTIMIZED`.
- Defina o parâmetro de atualização para a versão mais recente da imagem (`updateToLatestImageVersion`) como `true`.
- Não especifique uma ID de AMI em `imageId`, `imageIdOverride` (em `ec2Configuration`) ou no modelo de execução (`launchTemplate`). Nesse caso, AWS Batch seleciona a AMI otimizada mais recente do Amazon ECS que é suportada AWS Batch no momento em que a atualização da infraestrutura é iniciada. Como alternativa, é possível especificar o ID da AMI os parâmetros `imageId` ou `imageIdOverride` ou o modelo de inicialização identificado pelas propriedades `LaunchTemplate`. A alteração de qualquer uma dessas propriedades inicia uma atualização da infraestrutura. Se a ID da AMI for especificada no modelo de execução, ela não poderá ser substituída pela especificação de uma ID da AMI nos parâmetros `imageId` ou `imageIdOverride`. Ela só pode ser substituída especificando um modelo de lançamento diferente. Ou, se a versão do modelo de lançamento estiver definida como `$Default` ou `$Latest`, definindo uma nova versão padrão para o modelo de execução (se for `$Default`) ou adicionando uma nova versão ao modelo de execução (se for `$Latest`).

Se essas regras forem seguidas, qualquer atualização que inicie uma atualização de infraestrutura fará com que o ID da AMI seja novamente selecionado. Se a configuração da `version` no modelo de execução (`launchTemplate`) for definida como `$Latest` ou `$Default`, a versão mais recente ou padrão do modelo de inicialização será avaliada no

momento da atualização da infraestrutura, mesmo que o `launchTemplate` não tenha sido atualizado.

Consideração ao criar trabalhos paralelos de vários nós

AWS Batch recomenda a criação de ambientes de computação dedicados para executar trabalhos paralelos de vários nós (MNP) e trabalhos não MNP. Isso se deve à forma como a capacidade de computação é criada em seu ambiente de computação gerenciado. Ao criar um novo ambiente de computação gerenciado, se você especificar um valor `minvCpu` maior que zero, o AWS Batch criará um pool de instâncias para uso somente com trabalhos não MNP. Se um trabalho paralelo de vários nós for enviado, AWS Batch criará uma nova capacidade de instância para executar os trabalhos paralelos de vários nós. Nos casos em que há trabalhos paralelos de um e vários nós em execução no mesmo ambiente computacional em que um `maxvCpus` valor `minvCpus` ou é definido, se os recursos computacionais necessários não estiverem disponíveis, AWS Batch aguardará a conclusão dos trabalhos atuais antes de criar os recursos computacionais necessários para executar os novos trabalhos.

Ambientes de computação não gerenciados

Em um ambiente de computação não gerenciado, você gerencia seus próprios recursos de computação. Você deve verificar se a AMI que usa para seus recursos de computação atende à especificação da AMI da instância de contêiner. Para obter mais informações, consulte Especificação da AMI do recurso de computação e Como criar uma AMI de recursos de computação.

> **Note**
>
> AWS Os recursos do Fargate não são suportados em ambientes computacionais não gerenciados.

Depois de criar seu ambiente computacional não gerenciado, use a operação de DescribeComputeEnvironmentsAPI para ver os detalhes do ambiente computacional. Encontre o cluster do Amazon ECS que está associado ao ambiente, e inicie manualmente suas instâncias de contêiner nesse cluster do Amazon ECS.

O AWS CLI comando a seguir também fornece o ARN do cluster Amazon ECS.

```
$ aws batch describe-compute-environments \
    --compute-environments unmanagedCE \
    --query "computeEnvironments[].ecsClusterArn"
```

Para obter mais informações, consulte Iniciando uma instância de contêiner do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service. Ao executar os recursos de computação, especifique o ARN do cluster do Amazon ECS que os recursos devem registrar com os seguintes dados de usuário do Amazon EC2. *ecsClusterArn*Substitua pelo ARN do cluster que você obteve com o comando anterior.

```
#!/bin/bash
echo "ECS_CLUSTER=ecsClusterArn" >> /etc/ecs/ecs.config
```

Recursos de computação de AMIs

Por padrão, os ambientes computacionais AWS Batch gerenciados usam uma versão recente e aprovada da AMI otimizada do Amazon ECS para recursos computacionais. No entanto, você pode querer criar sua própria AMI a ser usada para seus ambientes de computação gerenciados por vários motivos. Se você precisar de alguma das opções a seguir, recomendamos que crie sua própria AMI:

- Aumentar o tamanho do armazenamento dos volumes de dados ou raiz da AMI
- Adicionar volumes de armazenamento de instância com suporte para tipos de instância do Amazon EC2
- Inspecionar o agente de contêiner do Amazon ECS
- Personalizando o Docker
- Configurar uma AMI de workload de GPU que permite que os contêineres acessem o hardware de GPU nos tipos de instância do Amazon EC2 compatíveis

> **Note**
>
> Depois que um ambiente computacional é criado, AWS Batch não atualiza as AMIs no ambiente computacional. AWS Batch também não atualiza as AMIs em seu ambiente computacional quando uma versão mais recente da AMI otimizada do Amazon ECS está disponível. Você é responsável pelo gerenciamento do sistema operacional convidado. Isso inclui quaisquer atualizações e patches de segurança. Você também é responsável

por quaisquer outros utilitários ou aplicativos de software que instalar nos recursos de computação. Para usar uma nova AMI para seus AWS Batch trabalhos, faça o seguinte:

1. Crie um novo ambiente de computação com a nova AMI.
2. Adicione o ambiente de computação a uma fila de trabalhos existente.
3. Remova o antigo ambiente de computação da fila de trabalhos.
4. Exclua o ambiente de computação anterior.

Em abril de 2022, AWS Batch foi adicionado suporte aprimorado para atualização de ambientes computacionais. Para ter mais informações, consulte Criação de um ambiente de computação. Para utilizar a atualização aprimorada de ambientes de computação para atualizar AMIs, siga estas regras:

- Não defina o parâmetro da função de serviço (`serviceRole`) nem o defina como a função AWSServiceRoleForBatchvinculada ao serviço.
- Defina o parâmetro da estratégia de alocação (`allocationStrategy`) como `BEST_FIT_PROGRESSIVE`, `SPOT_CAPACITY_OPTIMIZED` ou `SPOT_PRICE_CAPACITY_OPTIMIZED`.
- Defina o parâmetro de atualização para a versão mais recente da imagem (`updateToLatestImageVersion`) como `true`.
- Não especifique uma ID de AMI em `imageId`, `imageIdOverride` (em `ec2Configuration`) ou no modelo de execução (`launchTemplate`). Quando você não especifica uma ID de AMI, AWS Batch seleciona a última AMI otimizada do Amazon ECS que AWS Batch oferece suporte no momento em que a atualização da infraestrutura é iniciada. Como alternativa, você pode especificar a ID da AMI nos parâmetros `imageId` ou `imageIdOverride`. Ou pode especificar o modelo de execução que é identificado pelas propriedades `LaunchTemplate`. A alteração de qualquer uma dessas propriedades inicia uma atualização da infraestrutura. Se a ID da AMI for especificada no modelo de lançamento, a ID da AMI não poderá ser substituída pela especificação de uma ID da AMI nos parâmetros `imageId` ou `imageIdOverride`. A ID da AMI só pode ser substituída pela especificação de um modelo de lançamento diferente. Se a versão do modelo de lançamento estiver definida como `$Default` ou `$Latest`, a ID da AMI poderá ser substituída definindo uma nova versão padrão para o modelo de lançamento (se `$Default`) ou adicionando uma nova versão ao modelo de lançamento (se `$Latest`).

Se essas regras forem seguidas, qualquer atualização que acione uma atualização de infraestrutura fará com que a ID da AMI seja novamente selecionada. Se a configuração da `version` no modelo de execução (`launchTemplate`) for definida como `$Latest` ou `$Default`, a versão mais recente ou padrão do modelo de inicialização será avaliada no momento da atualização da infraestrutura, mesmo que o `launchTemplate` não tenha sido atualizado.

Tópicos

Especificação da AMI do recurso de computação

A especificação básica da AMI de recursos AWS Batch computacionais consiste no seguinte:

Obrigatório

- A distribuição Linux moderna executando pelo menos a versão 3.10 do Linux kernel em uma AMI de tipo de virtualização HVM. Não há suporte para contêineres do Windows.

  Important

  Tarefas paralelas de vários nós só podem ser executadas em recursos de computação que foram iniciados em uma instância do Amazon Linux com o pacote `ecs-init` instalado. Recomendamos que você use a AMI otimizada para Amazon ECS ao criar o ambiente de computação. Você pode fazer isso sem especificar uma AMI personalizada. Para ter mais informações, consulte Trabalhos paralelos de vários nós .

- O agente de contêiner do Amazon ECS. É recomendável usar a versão mais recente. Para obter mais informações, consulte Installing the Amazon ECS Container Agent no Guia do desenvolvedor do Amazon Elastic Container Service.

- O driver de log `awslogs` deve ser especificado como um driver de log disponível com a variável de ambiente `ECS_AVAILABLE_LOGGING_DRIVERS` quando o agente de contêiner do Amazon ECS é iniciado. Para obter mais informações, consulte Configuração do Agente de Contêineres do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.
- Um daemon do Docker em execução, pelo menos, na versão 1.9 e quaisquer dependências de execução do Docker. Para mais informações, consulte Verificar dependências do runtime na documentação do Docker.

> **Note**
>
> Recomendamos a versão do Docker que é fornecida e testada com a versão de agente de contêiner do Amazon ECS que você está usando. O Amazon ECS fornece um changelog para a variante Linux da AMI otimizada para Amazon ECS em. GitHub Para obter mais informações, consulte Log de alterações.

Recomendado

- Um processo de inicialização e de nanny para executar e monitorar o agente do Amazon ECS. A AMI otimizada para Amazon ECS usa o processo de inicialização `ecs-init` e outros sistemas operacionais podem usar `systemd`. Para obter mais informações e exemplos, consulte Example container instance User Data Configuration Scripts no Guia do desenvolvedor do Amazon Elastic Container Service. Para obter mais informações sobre`ecs-init`, consulte o `ecs-init`projeto em GitHub. No mínimo, os ambientes de computação gerenciados exigem que o agente do Amazon ECS seja iniciado na inicialização. Se o agente do Amazon ECS não estiver em execução no seu recurso computacional, ele não poderá aceitar trabalhos do. AWS Batch

As AMIs otimizadas para Amazon ECS são pré-configuradas com esses requisitos e com essas recomendações. Recomendamos que você use a AMI otimizada para Amazon ECS ou uma AMI do Amazon Linux com o pacote `ecs-init` que está instalado para seus recursos de computação. Escolha outra AMI se o aplicativo precisar de um sistema operacional específico ou uma versão do Docker que ainda não está disponível nessas AMIs. Para obter mais informações, consulte Amazon ECS-Optimized AMI no Guia do desenvolvedor do Amazon Elastic Container Service.

Como criar uma AMI de recursos de computação

É possível criar uma personalizada de recursos de computação para usar em seus ambientes de computação. Para obter instruções, consulte o Especificação da AMI do recurso de computação. Então, depois de criar uma AMI personalizada, você pode criar um ambiente de computação que usa essa AMI, a que você pode associar uma fila de trabalhos. Por fim, comece a enviar trabalhos para essa fila.

Para criar uma AMI de recursos de computação personalizada

1. Escolha uma AMI base para começar. A AMI básica deve usar virtualização de HVM. A AMI básica não pode ser uma AMI do Windows.

   > **Note**
   >
   > A AMI que você escolher para um ambiente de computação deve corresponder à arquitetura dos tipos de instância que você deseja usar para este ambiente. Por exemplo, se o ambiente de computação usar tipos de instância A1, a AMI de recursos de computação escolhida deverá oferecer suporte a instâncias Arm. O Amazon ECS vende as versões x86 e Arm da AMI do Amazon Linux 2 otimizada para Amazon ECS. Para obter mais informações, consulte AMI do Amazon Linux 2 otimizada para Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

   A AMI do Amazon Linux 2 otimizada para Amazon ECS é a AMI padrão para recursos de computação em ambientes de computação gerenciados. A AMI do Amazon Linux 2 otimizada o para Amazon ECS é pré-configurada e testada no AWS Batch por engenheiros da AWS. É uma AMI mínima com a qual você pode começar a usar e fazer com que seus recursos de computação sejam executados na AWS rapidamente. Para obter mais informações, consulte AMI otimizada para Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

   Como alternativa, você pode escolher outra variante do Amazon Linux 2 e instalar o pacote `ecs-init` com os seguintes comandos. Para obter mais informações, consulte Instalar o atendente de contêiner do Amazon ECS em uma instância do EC2 do Amazon Linux 2 no Guia do desenvolvedor do Amazon Elastic Container Service:

   ```
   $ sudo amazon-linux-extras disable docker
   ```

```
$ sudo amazon-linux-extras install ecs-init
```

Por exemplo, se deseja executar workloads de GPU em seus recursos de computação do AWS Batch, você pode começar com a Amazon Linux Deep Learning AMI. Em seguida, configure a AMI para executar trabalhos do AWS Batch. Para obter mais informações, consulte Usar uma AMI de workload de GPU.

> **Important**
>
> Você pode escolher uma AMI básica que não seja compatível com o pacote `ecs-init`. No entanto, se fizer isso, você deverá configurar uma forma de iniciar o atendente do Amazon ECS na inicialização e mantê-lo em execução. Você também pode ver vários exemplos de scripts de configuração de dados do usuário que usam `systemd` para iniciar e monitorar o atendente de contêiner do Amazon ECS. Para obter mais informações, consulte Scripts de configuração de dados de usuário de instância de contêiner de exemplo no Guia do desenvolvedor do Amazon Elastic Container Service.

2. Execute uma instância de sua AMI base selecionada com as opções de armazenamento adequadas para sua AMI. Você pode configurar o tamanho e o número de volumes do Amazon EBS conectados ou volumes de armazenamento de instância se o tipo de instância selecionado for compatível com eles. Para obter mais informações, consulte Como iniciar uma instância e Armazenamento de instância do Amazon EC2 no Guia do usuário do Amazon EC2 para instâncias do Linux.
3. Conecte-se à sua instância com o SSH e execute todas as tarefas de configuração necessárias. Isso pode incluir qualquer uma das ou todas as seguintes etapas:

   - Como instalar o atendente de contêiner do Amazon ECS. Para obter mais informações, consulte Instalar o atendente de contêiner do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.
   - Configuração de um script para formatar volumes de armazenamento de instâncias.
   - Adição de volume de armazenamento de instância ou sistemas de arquivos Amazon EFS para o arquivo `/etc/fstab` para que eles sejam montados na inicialização.
   - Configuração de opções do Docker, como habilitar a depuração ou ajustar o tamanho da imagem base.
   - Instalação de pacotes ou cópia de arquivos.

Para obter mais informações, consulte Conectando-se à sua Instância do Linux usando SSH no Guia do usuário do Amazon EC2 para instâncias do Linux.

4. Se você iniciou o atendente de contêiner do Amazon ECS em sua instância, deve interrompê-lo e remover todos os arquivos persistentes do ponto de verificação de dados antes de criar sua AMI. Caso contrário, se você não fizer isso, o atendente não iniciará nas instâncias que são executadas a partir da sua AMI.

   a. Interrompa o atendente de contêiner do Amazon ECS.

      - AMI do Amazon Linux 2 otimizada para Amazon ECS:

        ```
        sudo systemctl stop ecs
        ```

      - AMI do Amazon Linux otimizada para Amazon ECS:

        ```
        sudo stop ecs
        ```

   b. Remova os arquivos de ponto de verificação de dados persistentes. Por padrão, esses arquivos estão localizados no diretório `/var/lib/ecs/data/`. Use o comando a seguir para remover esses arquivos, se houver algum.

      ```
      sudo rm -rf /var/lib/ecs/data/*
      ```

5. Crie uma nova AMI da sua instância em execução. Para obter mais informações, consulte Criar uma AMI do Linux baseada no Amazon EBS no Guia do usuário do Amazon EC2 para instâncias do Linux.

Para usar a nova AMI com o AWS Batch

1. Após a criação da nova AMI, crie um ambiente de computação com a nova AMI. Para fazer isso escolha o tipo de imagem e insira a ID da AMI personalizada na caixa de substituição da ID da imagem ao criar o ambiente de computação do AWS Batch. Para obter mais informações, consulte the section called “Para criar um ambiente de computação gerenciado usando recursos do EC2”.

> **Note**
>
> A AMI que você escolher para um ambiente de computação deve corresponder à arquitetura dos tipos de instância que você deseja usar para este ambiente. Por exemplo, se o ambiente de computação usar tipos de instância A1, a AMI de recursos de computação escolhida deverá oferecer suporte a instâncias Arm. O Amazon ECS vende as versões x86 e Arm da AMI do Amazon Linux 2 otimizada para Amazon ECS. Para obter mais informações, consulte AMI do Amazon Linux 2 otimizada para Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

2. Crie uma fila de trabalhos e associe seu novo ambiente de computação. Para obter mais informações, consulte Como criar uma fila de tarefas.

> **Note**
>
> Todos os ambientes computacionais associados a uma fila de trabalhos devem compartilhar a mesma arquitetura. O AWS Batch não oferece suporte à mistura de tipos de arquitetura de ambiente computacional em uma única fila de trabalhos.

3. (Opcional) Envie um trabalho de amostra para sua nova fila de trabalhos. Para obter mais informações, consulte Definições de trabalho de exemplo, Como criar uma definição de tarefa de nó único e Enviando um trabalho.

Usar uma AMI de workload de GPU

Para executar cargas de trabalho de GPU em seus recursos de computação do AWS Batch, você deve usar uma AMI com suporte para GPU. Para obter mais informações, consulte Como trabalhar com GPUs no Amazon ECS e AMIs otimizadas para Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

Em ambientes de computação gerenciados, se o ambiente de computação epecificar qualquer família de instância ou tipo de instância `p2`, `p3`, `p4`, `p5`, `g3`, `g3s`, `g4` ou `g5`, o AWS Batch o usa uma AMI otimizada para GPU do Amazon ECS.

Em ambientes de computação não gerenciados, é recomendada uma AMI otimizada para GPU do Amazon ECS. Você pode usar as operações AWS Command Line Interface ou AWS Systems

Manager Parameter Store GetParameter, GetParameters, e GetParametersByPath para recuperar os metadados para as AMIs otimizadas para GPU do Amazon ECS recomendadas.

> **Note**
>
> A família de p5 instância só é compatível com versões iguais ou posteriores à AMI otimizada para GPU 20230912 do Amazon ECS, e elas são incompatíveis com os tipos de instância p2 e g2. Se você precisar usar instâncias p5, certifique-se de que seu ambiente de computação não contenha instâncias p2 ou g2 e use o Batch AMI padrão mais recente. A criação de um novo ambiente de computação usará a AMI mais recente, mas se você estiver atualizando seu ambiente de computação para incluir p5, você pode garantir que está usando a AMI mais recente configurando o `updateToLatestImageVersion` como `true` nas propriedades `ComputeResource`. Para obter mais informações sobre a compatibilidade do AMI com instâncias de GPU, consulte Como trabalhar com GPUs no Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

Os exemplos a seguir mostram como usar o comando GetParameter.

AWS CLI

```
$ aws ssm get-parameter --name /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended \
                        --region us-east-2 --output json
```

A saída inclui as informações de AMI no parâmetro `Value`.

```
{
    "Parameter": {
        "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended",
        "LastModifiedDate": 1555434128.664,
        "Value": "{\"schema_version\":1,\"image_name\":\"amzn2-ami-ecs-gpu-
hvm-2.0.20190402-x86_64-ebs\",\"image_id\":\"ami-083c800fe4211192f\",\"os\":\"Amazon
 Linux 2\",\"ecs_runtime_version\":\"Docker version 18.06.1-ce\",\"ecs_agent_version
\":\"1.27.0\"}",
        "Version": 9,
        "Type": "String",
        "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended"
    }
```

```
}
```

Python

```
from __future__ import print_function

import json
import boto3

ssm = boto3.client('ssm', 'us-east-2')

response = ssm.get_parameter(Name='/aws/service/ecs/optimized-ami/amazon-linux-2/
gpu/recommended')
jsonVal = json.loads(response['Parameter']['Value'])
print("image_id   = " + jsonVal['image_id'])
print("image_name = " + jsonVal['image_name'])
```

A saída inclui apenas o ID da AMI e nome da AMI:

```
image_id   = ami-083c800fe4211192f
image_name = amzn2-ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs
```

Os exemplos a seguir demonstram o uso de GetParameters.

AWS CLI

```
$ aws ssm get-parameters --names  /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended/image_name \
                                  /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended/image_id \
                         --region us-east-2 --output json
```

A saída inclui os metadados completos para cada um dos parâmetros:

```
{
    "InvalidParameters": [],
    "Parameters": [
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_id",
```

```
            "LastModifiedDate": 1555434128.749,
            "Value": "ami-083c800fe4211192f",
            "Version": 9,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_id"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_name",
            "LastModifiedDate": 1555434128.712,
            "Value": "amzn2-ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs",
            "Version": 9,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_name"
        }
    ]
}
```

Python

```
from __future__ import print_function

import boto3

ssm = boto3.client('ssm', 'us-east-2')

response = ssm.get_parameters(
            Names=['/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_name',
                   '/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_id'])
for parameter in response['Parameters']:
    print(parameter['Name'] + " = " + parameter['Value'])
```

A saída inclui o ID da AMI e nome da AMI, usando o caminho completo para os nomes.

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_id =
 ami-083c800fe4211192f
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_name = amzn2-
ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs
```

Os exemplos a seguir mostram como usar o comando GetParametersByPath.

AWS CLI

```
$ aws ssm get-parameters-by-path --path /aws/service/ecs/optimized-ami/amazon-
linux-2/gpu/recommended \
                                --region us-east-2 --output json
```

A saída inclui todos os metadados completos para todos os parâmetros no caminho especificado.

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
ecs_agent_version",
            "LastModifiedDate": 1555434128.801,
            "Value": "1.27.0",
            "Version": 8,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/ecs_agent_version"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
ecs_runtime_version",
            "LastModifiedDate": 1548368308.213,
            "Value": "Docker version 18.06.1-ce",
            "Version": 1,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/ecs_runtime_version"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_id",
            "LastModifiedDate": 1555434128.749,
            "Value": "ami-083c800fe4211192f",
            "Version": 9,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_id"
        },
        {
```

```
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
image_name",
            "LastModifiedDate": 1555434128.712,
            "Value": "amzn2-ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs",
            "Version": 9,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/image_name"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
os",
            "LastModifiedDate": 1548368308.143,
            "Value": "Amazon Linux 2",
            "Version": 1,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/os"
        },
        {
            "Name": "/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/
schema_version",
            "LastModifiedDate": 1548368307.914,
            "Value": "1",
            "Version": 1,
            "Type": "String",
            "ARN": "arn:aws:ssm:us-east-2::parameter/aws/service/ecs/optimized-ami/
amazon-linux-2/gpu/recommended/schema_version"
        }
    ]
}
```

Python

```
from __future__ import print_function

import boto3

ssm = boto3.client('ssm', 'us-east-2')

response = ssm.get_parameters_by_path(Path='/aws/service/ecs/optimized-ami/amazon-
linux-2/gpu/recommended')
for parameter in response['Parameters']:
```

```
    print(parameter['Name'] + " = " + parameter['Value'])
```

A saída inclui os valores de todos os nomes de parâmetro no caminho especificado, usando o caminho completo para os nomes.

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/ecs_agent_version =
 1.27.0
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/ecs_runtime_version =
 Docker version 18.06.1-ce
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_id =
 ami-083c800fe4211192f
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/image_name = amzn2-
ami-ecs-gpu-hvm-2.0.20190402-x86_64-ebs
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/os = Amazon Linux 2
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/recommended/schema_version = 1
```

Para obter mais informações, consulte Recuperar os metadados da AMI otimizada para Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

Depreciação do Amazon Linux

O Amazon Linux AMI (também chamado de Amazon Linux 1) chegou ao fim em 31 de dezembro de 2023. AWS Batch encerrou o suporte para o Amazon Linux AMI, pois não receberá nenhuma atualização de segurança ou correção de erros a partir de 1º de janeiro de 2024. Para obter mais informações sobre o Amazon Linux end-of-life, consulte as perguntas frequentes da AL.

Recomendamos que você atualize os ambientes computacionais existentes baseados no Amazon Linux para o Amazon Linux 2023 para evitar interrupções imprevistas na carga de trabalho e continuar recebendo atualizações de segurança e outras.

Seus ambientes computacionais usando o Amazon Linux AMI podem continuar funcionando além da end-of-life data de 31 de dezembro de 2023. No entanto, esses ambientes computacionais não receberão mais novas atualizações de software, patches de segurança ou correções de bugs da AWS. É sua responsabilidade manter esses ambientes computacionais na Amazon Linux AMI posteriormente end-of-life. Recomendamos migrar ambientes AWS Batch computacionais para o Amazon Linux 2023 ou o Amazon Linux 2 para manter o desempenho e a segurança ideais.

Para obter ajuda na migração AWS Batch do Amazon Linux AMI para o Amazon Linux 2023 ou Amazon Linux 2, consulte Atualização de ambientes computacionais -. AWS Batch

Suporte a modelo de execução

AWS Batch suporta o uso de modelos de lançamento do Amazon EC2 com seus ambientes de computação EC2. O suporte do modelo de execução permite modificar a configuração padrão de seus recursos de computação de AWS Batch sem exigir que você crie AMIs personalizadas.

> **Note**
>
> Os modelos de lançamento não são compatíveis com os recursos do AWS Fargate.

Você deve criar um modelo de execução antes de associá-lo a um ambiente de computação. É possível criar um modelo de execução usando o console Amazon EC2. Ou você pode usar o AWS CLI ou um AWS SDK. Por exemplo, o arquivo JSON a seguir representa um modelo de execução que redimensiona o volume de dados do Docker para a do recurso de computação padrão de e também o configura para ser criptografado.

```
{
    "LaunchTemplateName": "increase-container-volume-encrypt",
    "LaunchTemplateData": {
        "BlockDeviceMappings": [
            {
                "DeviceName": "/dev/xvda",
                "Ebs": {
                    "Encrypted": true,
                    "VolumeSize": 100,
                    "VolumeType": "gp2"
                }
            }
        ]
    }
}
```

Você pode criar o modelo de execução anterior salvando o JSON para um arquivo chamado `lt-data.json` e executando o seguinte comando da AWS CLI:

```
aws ec2 --region <region> create-launch-template --cli-input-json file://lt-data.json
```

Para obter mais informações sobre modelos de execução, consulte Executar uma instância de um modelo de execução no Guia do usuário do Amazon EC2 para instâncias do Linux.

Se você usar um modelo de execução para criar seu ambiente de computação, poderá mover os seguintes parâmetros de ambiente de computação existentes para o seu modelo de execução:

> **Note**
>
> Suponha que qualquer um desses parâmetros (exceto as tags do Amazon EC2) seja especificado no modelo de execução e na configuração do ambiente de computação. Então, os parâmetros do ambiente de computação têm precedência. As tags do Amazon EC2 são mescladas entre o modelo de lançamento e a configuração do ambiente de computação. Se houver uma colisão na chave de tag, o valor na configuração do ambiente de computação tem precedência.

- Pares de chaves do Amazon EC2
- ID da AMI do Amazon EC2
- IDs de grupos de segurança
- Etiquetas do Amazon EC2

Os seguintes parâmetros de modelo de execução são ignorados por AWS Batch:

- Tipo de instância (especifique os tipos de instâncias desejados ao criar o ambiente de computação)
- Função de instância (especifique a função de instância desejada ao criar o ambiente de computação)
- Sub-redes de interface de rede (especifique as sub-redes desejadas ao criar seu ambiente de computação)
- Opções de mercado de instâncias (o AWS Batch deve controlar a configuração da instância spot)
- Desativar o encerramento da API (o AWS Batch deve controlar o ciclo de vida da instância)

AWS Batch atualiza o modelo de execução somente com uma nova versão do modelo de execução durante as atualizações da infraestrutura. Para obter mais informações, consulte Criação de um ambiente de computação.

Dados de usuário do Amazon EC2 em modelos de execução

Você pode fornecer os dados de usuário do Amazon EC2 em seu modelo de execução que é executado por cloud-init quando suas instâncias são iniciadas. Seus dados de usuário podem executar cenários de configuração comuns, incluindo, dentre outros:

- Incluindo usuários ou grupos
- Instalar pacotes
- Criar partições e sistemas de arquivos

Os dados do usuário do Amazon EC2 em modelos de execução devem estar no formato MIME multi-part archive Isso ocorre porque seus dados de usuário são mesclados com outros dados de usuário AWS Batch necessários para configurar seus recursos de computação. Você pode combinar vários blocos de dados de usuário em um único arquivo MIME de várias partes. Por exemplo, convém combinar um boothook de nuvem que configure o daemon do Docker com um script do shell de dados do usuário que grave informações do agente de contêiner do Amazon ECS.

Se você estiver usando o AWS CloudFormation, o tipo AWS::CloudFormation::Init pode ser usado com o script auxiliar cfn-init para executar cenários de configuração comuns.

Um arquivo em várias partes MIME consiste nos seguintes componentes:

- O tipo de conteúdo e a declaração de limite da parte: `Content-Type: multipart/mixed; boundary="==BOUNDARY=="`
- A declaração da versão MIME: `MIME-Version: 1.0`
- Um ou mais blocos de dados do usuário que contêm os seguintes componentes:
  - O limite de abertura, que sinaliza o início de um bloco de dados do usuário: `--==BOUNDARY==` Você deve manter a linha antes desse limite em branco.
  - A declaração de tipo de conteúdo para o bloco: `Content-Type: text/cloud-config; charset="us-ascii"`. Para mais informações sobre tipos de conteúdo, consulte a documentação do Cloud-Init. Você deve manter a linha após o branco da declaração do tipo de conteúdo.
  - O conteúdo de dados do usuário, por exemplo, uma lista de comandos de shell ou diretivas do `cloud-init`.
- O limite de fechamento que sinaliza o fim do arquivo MIME de várias partes: `--==BOUNDARY==--` Você deve manter a linha antes do branco do limite de fechamento.

Veja a seguir um exemplo de arquivo MIME com várias partes que você pode usar para criar seu próprio.

> **Note**
>
> Se você adicionar dados de usuário a um modelo de execução no console do Amazon EC2, você pode colá-los como texto simples ou fazer upload de um arquivo. Ou você pode fazer o upload de um arquivo. Se você usa a AWS CLI ou um SDK da AWS, primeiro, você deve codificar `base64` os dados de usuário e enviar essa sequência de caracteres como o valor do parâmetro `UserData` quando você chama CreateLaunchTemplate, conforme mostrado neste arquivo JSON.
>
> ```
> {
>     "LaunchTemplateName": "base64-user-data",
>     "LaunchTemplateData": {
>         "UserData":
>  "ewogICAgIkxhdW5jaFRlbXBsYXRlTmFtZSI6ICJpbmNyZWFzZS1jb250YWluZXItdm9sdW..."
>     }
> }
> ```

Exemplos

- Exemplo: montar um sistema de arquivos existente do Amazon EFS
- Exemplo: substituir a configuração padrão do agente de contêiner do Amazon ECS
- Exemplo: montar um sistema de arquivos do Amazon FSx para Lustre

Exemplo: montar um sistema de arquivos existente do Amazon EFS

Example

Este exemplo de arquivo MIME de várias partes configura o recurso de computação para instalar o pacote `amazon-efs-utils` e montar um sistema de arquivos existente do Amazon EFS em `/mnt/efs`.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
```

```
Content-Type: text/cloud-config; charset="us-ascii"

packages:
- amazon-efs-utils

runcmd:
- file_system_id_01=fs-abcdef123
- efs_directory=/mnt/efs

- mkdir -p ${efs_directory}
- echo "${file_system_id_01}:/ ${efs_directory} efs tls,_netdev" >> /etc/fstab
- mount -a -t efs defaults

--==MYBOUNDARY==--
```

Exemplo: substituir a configuração padrão do agente de contêiner do Amazon ECS

Example

Este exemplo de arquivo MIME de várias partes substitui as configurações padrão de limpeza de imagem de docker para um recurso de computação.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
echo ECS_IMAGE_CLEANUP_INTERVAL=60m >> /etc/ecs/ecs.config
echo ECS_IMAGE_MINIMUM_CLEANUP_AGE=60m >> /etc/ecs/ecs.config

--==MYBOUNDARY==--
```

Exemplo: montar um sistema de arquivos do Amazon FSx para Lustre

Example

Este exemplo de arquivo MIME de várias partes configura o recurso de computação para instalar o pacote `lustre2.10` a partir da Biblioteca de extras e montar um sistema de arquivos FSx for Lustre existente em `/scratch` e um nome de montagem de `fsx`. Este exemplo é para o Amazon Linux

2. Para obter instruções de instalação para outras distribuições Linux, consulte Installing the Lustre Client no Amazon FSx para Lustre User Guide. Para obter mais informações, consulte Mounting your Amazon FSx file system automatically no Guia do usuário do Amazon FSx para Lustre.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

runcmd:
- file_system_id_01=fs-0abcdef1234567890
- region=us-east-2
- fsx_directory=/scratch
- amazon-linux-extras install -y lustre2.10
- mkdir -p ${fsx_directory}
- mount -t lustre ${file_system_id_01}.fsx.${region}.amazonaws.com@tcp:fsx
 ${fsx_directory}

--==MYBOUNDARY==--
```

Nos membros volumes e mountPoints das propriedades do contêiner, os pontos de montagem devem ser mapeados para o contêiner.

```
{
    "volumes": [
        {
            "host": {
                "sourcePath": "/scratch"
            },
            "name": "Scratch"
        }
    ],
    "mountPoints": [
        {
            "containerPath": "/scratch",
            "sourceVolume": "Scratch"
        }
    ],
}
```

Criação de um ambiente de computação

Antes de executar trabalhos no AWS Batch, você precisa criar um ambiente computacional. Você pode criar um ambiente computacional gerenciado onde AWS Batch gerencia as instâncias do Amazon EC2 ou os recursos do AWS Fargate dentro do ambiente com base em suas especificações. Ou, como alternativa, você pode criar um ambiente de computação não gerenciado onde você gerencia a configuração da instância do Amazon EC2 dentro do ambiente.

Important

As instâncias Fargate Spot não são suportadas nos seguintes cenários:

- Em contêineres Amazon Linux com arquitetura ARM64.
- Windows containers on AWS Fargate

Uma fila de trabalhos será bloqueada nesses cenários se um trabalho for enviado a uma fila de trabalhos que usa somente ambientes de computação Fargate Spot.

Sumário

Para criar um ambiente computacional gerenciado usando os recursos do AWS Fargate

1. Abra o AWS Batch console em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione o Região da AWS a ser usado.
3. No painel de navegação, escolha Ambientes de computação.
4. Escolha Criar.
5. Configure o ambiente de computação.

> **Note**
>
> Os ambientes de computação para Windows containers on AWS Fargate trabalhos devem ter pelo menos uma vCPU.

a. Para configuração do ambiente de computação, escolha Fargate.

b. Para Nome, especifique um nome exclusivo para seu ambiente de computação. O nome pode conter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).

c. Em Função de serviço, escolha a função vinculada ao serviço que permite que o AWS Batch serviço faça chamadas para as operações de AWS API necessárias em seu nome. Para este exemplo, selecione AWSServiceRoleForBatch. Para ter mais informações, consulte Permissões de função vinculadas ao serviço para AWS Batch.

d. (Opcional) Expanda as Tags. Para adicionar uma tag, escolha Add tag. Insira uma chave e um Valor opcional. Escolha Adicionar Tag.

e. Escolha Próxima página.

6. Na seção Instance configuration:

a. (Opcional) Para usar a capacidade do Fargate Spot, ative o Fargate Spot. Para obter informações sobre o Fargate Spot, consulte Usando o Amazon EC2 Spot e o Fargate_Spot.

b. Para Máximo de vCPUs, escolha o número máximo de vCPUs para o qual seu ambiente de computação deve aumentar, independentemente da demanda da fila de trabalhos.

c. Escolha Próxima página.

7. Configure redes.

> **Important**
>
> Recursos de computação precisam de acesso para se comunicar com o endpoint de serviço do Amazon ECS. Isso pode ser feito por meio de uma interface do endpoint da VPC ou por meio dos das instâncias de contêiner que tenham endereços IP públicos. Para obter mais informações sobre endpoints da VPC de interface, consulte Endpoints da VPC de interface do Amazon ECS (AWS PrivateLink) no Manual do Desenvolvedor do Amazon Elastic Container Service.

Se você não tiver um endpoint da VPC de interface configurado e seus das instâncias de contêiner não tiverem endereços IP públicos, eles deverão usar a conversão de endereço de rede (NAT) para fornecer esse acesso. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC. Para ter mais informações, consulte the section called “Crie uma VPC”.

a. Para ID de Nuvem Privada Virtual (VPC), escolha uma VPC na qual você deseja iniciar suas instâncias.

b. Em Subnets, escolha as sub-redes a serem usadas. Por padrão, todas as sub-redes dentro da VPC selecionadas estão disponíveis.

> **Note**
>
> AWS Batch No momento, o on Fargate não oferece suporte a Locais Zones. Para obter mais informações, consulte Amazon ECS clusters em Local Zones, Wavelength Zones, e AWS Outposts no Amazon Elastic Container Service Developer Guide.

c. Para Security groups, escolha um security group a ser anexado às suas instâncias. Por padrão, o security group padrão para sua VPC é escolhido.

d. Escolha Próxima página.

8. Para Revisar, reveja as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Criar ambiente de computação.

Para criar um ambiente de computação gerenciado usando recursos do EC2

1. Abra o AWS Batch console em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione o Região da AWS a ser usado.
3. No painel de navegação, escolha Ambientes de computação.
4. Escolha Criar.
5. Configure o ambiente.

a. Em configuração do ambiente de computação, escolha Amazon Elastic Compute Cloud (Amazon EC2).

b. Em Tipo de orquestração, escolha Gerenciado.

c. Para Nome, especifique um nome exclusivo para seu ambiente de computação. O nome pode conter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).

d. (Opcional) Para a função de serviço, escolha a função vinculada ao serviço que permite que o AWS Batch serviço faça chamadas para as operações de AWS API necessárias em seu nome. Para este exemplo, selecione AWSServiceRoleForBatch. Para ter mais informações, consulte Permissões de função vinculadas ao serviço para AWS Batch.

e. Para o Instance role, escolha criar um novo perfil de instância ou use um perfil de instância existente que tenha as permissões de IAM necessárias anexadas. Esse perfil de instância permite que as instâncias de contêiner do Amazon ECS criadas para seu ambiente computacional façam chamadas para as operações de AWS API necessárias em seu nome. Para ter mais informações, consulte Perfil de instância do Amazon ECS. Se você optou por criar um novo perfil de instância, a função necessária (`ecsInstanceRole`) será criada para você.

f. (Opcional) Expanda as Tags.

g. (Opcional) Para tags EC2, escolha Adicionar tag para adicionar uma tag aos recursos que são lançados no ambiente computacional. Insira uma chave e um Valor opcional. Escolha Adicionar Tag.

h. (Opcional) Em Tags, escolha Adicionar tag. Insira uma chave e um Valor opcional. Escolha Adicionar Tag.

   Para ter mais informações, consulte Marcando seus Recursos AWS Batch.

i. Escolha Próxima página.

6. Na seção Instance configuration:

   a. (Opcional) Para Habilitar o uso de instâncias Spot, ative o Spot. Para obter mais informações, consulte Spot Instances.

   b. (Somente Spot) Em Maximum % on-demand price, escolha a porcentagem máxima que o preço que uma instância spot deve ter em comparação com o preço sob demanda para esse tipo de instância antes que as instâncias sejam executadas. Por exemplo, se o preço máximo for 20%, o preço spot deverá estar abaixo de 20% do preço atual sob demanda

para essa instância do EC2. Você sempre paga o menor preço (mercado) e nunca mais do que sua porcentagem máxima. Se você deixar esse campo em branco, o valor padrão será 100% do preço sob demanda.

c. (Somente Spot) Para o perfil frota Spot, escolha uma função de IAM existente do Amazon EC2 Frota Spot para aplicar ao seu ambiente de computação Spot. Se você ainda não tiver uma função IAM do Amazon EC2 Frota Spot, deverá criar uma primeiro. Para ter mais informações, consulte Perfil de frota spot Amazon EC2.

> **Important**
>
> Para marcar suas Instâncias Spot na criação, sua função IAM do Amazon EC2 Spot Fleet deve usar a política gerenciada mais recente do SpotFleetTaggingRoleAmazonEC2. A política SpotFleetRole gerenciada do AmazonEC2 não tem as permissões necessárias para marcar instâncias spot. Para ter mais informações, consulte Instâncias spot sem tags na criação e the section called “Marcando seus Recursos”.

d. Para Mínimo de vCPUs, escolha o número mínimo de vCPUs que seu ambiente de computação deve manter, independentemente da demanda da fila de trabalhos.

e. Para vCPUs desejados, escolha o número de vCPUs com o qual seu ambiente de computação deve ser iniciado. À medida que a demanda da fila de trabalhos aumenta, o AWS Batch pode aumentar o número desejado de vCPUs no seu ambiente de computação e adicionar instâncias do EC2, até o máximo de vCPUs. À medida que a demanda diminui, o AWS Batch pode diminuir o número desejado de vCPUs no seu ambiente de computação e remover instâncias, até o mínimo de vCPUs.

f. Para Máximo de vCPUs, escolha o número máximo de vCPUs para o qual seu ambiente de computação deve aumentar, independentemente da demanda da fila de trabalhos.

g. Para Tipos de instância permitidos, escolha os tipos de instância do Amazon EC2 que podem ser iniciados. Você pode especificar famílias de instâncias para iniciar qualquer tipo de instância dentro dessas famílias (por exemplo `c5`, `c5n`, ou `p3`). Ou você pode especificar tamanhos específicos dentro de uma família (como `c5.8xlarge`). Os tipos de instância Metal não estão nas famílias de instâncias. Por exemplo, `c5` não inclui `c5.metal`. Você também pode escolher `optimal` para selecionar tipos de instância (das famílias de instâncias C4, M4, e R4 que correspondam à demanda de suas filas de trabalho.

> **Note**
>
> Ao criar um ambiente de computação, os tipos de instância selecionados para ele devem compartilhar a mesma arquitetura. Por exemplo, você não pode misturar instâncias ARM e x86 no mesmo ambiente de computação.

> **Note**
>
> AWS Batch escalará as GPUs com base na quantidade necessária em suas filas de trabalho. Para usar o agendamento de GPU, o ambiente de computação deve incluir tipos de instância das famílias `p2`, `p3`, `p4`, `p5`, `g3`, `g3s`, `g4`, ou `g5`.

> **Note**
>
> Atualmente, `optimal` usa tipos de instância das famílias de instância C4, M4, e R4. Regiões da AWS Nesse caso, não há tipos de instância dessas famílias de instâncias, tipos de instância da C5M5, e famílias de R5 instâncias são usadas.

h. Expanda Additional configuration.

i. (Opcional) Em Grupo de posicionamento, insira um nome de grupo de posicionamento para agrupar recursos no ambiente de computação.

j. (Opcional) Para o par de chaves EC2, escolha um par de chaves pública e privada como credenciais de segurança ao se conectar à instância. Para obter mais informações sobre pares de chaves Amazon EC2, consulte Amazon EC2 key pairs and Linux instances.

k. Para Estratégia de alocação, escolha a estratégia de alocação a ser usada ao selecionar tipos de instância na lista de tipos de instância permitidos. O BEST_FIT_PROGRESSIVE geralmente é a melhor opção para ambientes de computação sob demanda do EC2, SPOT_CAPACITY_OPTIMIZED e SPOT_PRICE_CAPACITY_OPTIMIZED para ambientes de computação EC2 Spot. Para ter mais informações, consulte the section called “Estratégias de alocação”.

l. (Opcional) Para a configuração do EC2, escolha o tipo de imagem e os valores de substituição do ID da imagem para fornecer informações AWS Batch para selecionar Amazon Machine Images (AMIs) para instâncias no ambiente computacional. Se a

substituição do ID da imagem não for especificada para cada tipo de imagem, AWS Batch seleciona uma AMI otimizada recente do Amazon ECS. Se nenhum tipo de imagem for especificado, o padrão será Amazon Linux 2 para instância sem GPU e sem AWS Graviton.

> **Important**
>
> Para usar uma AMI personalizada, escolha o tipo de imagem e insira a ID da AMI personalizada na caixa de substituição de ID da imagem.

Amazon Linux 2

Padrão para todas as famílias de instâncias AWS baseadas em Graviton (por exemplo,, `C6g` `M6gR6g`, `eT4g`) e pode ser usado para todos os tipos de instâncias que não sejam de GPU.

Amazon Linux 2 (GPU)

Padrão para todas as famílias de instâncias de GPU (por exemplo, `P4` `eG4`) e pode ser usado para todos os tipos de instância que não sejam AWS baseados em Graviton.

Amazon Linux

Pode ser usado para famílias de instâncias sem GPU e sem AWS Graviton. O suporte padrão para a AMI do Amazon Linux terminou. Para obter mais informações, consulte AMI do Amazon Linux.

> **Note**
>
> A AMI que você escolher para um ambiente de computação deve corresponder à arquitetura dos tipos de instância que você pretende usar para este ambiente. Por exemplo, se o ambiente de computação usar tipos de instância A1, a AMI de recursos de computação escolhida deverá oferecer suporte a instâncias Arm. O Amazon ECS vende as versões x86 e Arm da Amazon ECS optimized Amazon Linux 2 AMI. Para obter mais informações, consulte AMI do Amazon Linux 2 otimizada para Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

m. (Opcional) Em Modelo de execução, selecione um modelo de execução existente do Amazon EC2 para configurar seus recursos de computação. A versão padrão do modelo é preenchida automaticamente. Para ter mais informações, consulte Suporte a modelo de execução.

> **Note**
>
> Em um modelo de execução, é possível especificar uma AMI personalizada que você tenha criado.

n. (Opcional) Em Versão do modelo de execução, insira `$Default`, `$Latest` ou um número de versão específico para ser usado.

> **Important**
>
> Se o parâmetro de versão do modelo de execução for `$Default` ou `$Latest`, a versão padrão ou mais recente do modelo de execução especificado será avaliada durante uma atualização de infraestrutura. Se uma ID de AMI diferente for selecionada por padrão ou se a versão mais recente do modelo de execução for selecionada, essa ID de AMI será usada na atualização. Para ter mais informações, consulte the section called “Atualização do ID da AMI”.

o. Escolha Próxima página.

7. Na seção Configuração de rede:

> **Important**
>
> Recursos de computação precisam de acesso para se comunicar com o endpoint de serviço do Amazon ECS. Isso pode ser feito por meio de uma interface do endpoint da VPC ou por meio dos das instâncias de contêiner que tenham endereços IP públicos. Para obter mais informações sobre endpoints da VPC de interface, consulte Endpoints da VPC de interface do Amazon ECS (AWS PrivateLink) no Manual do Desenvolvedor do Amazon Elastic Container Service.
> Se você não tiver um endpoint da VPC de interface configurado e seus das instâncias de contêiner não tiverem endereços IP públicos, eles deverão usar a conversão de endereço de rede (NAT) para fornecer esse acesso. Para obter mais informações,

consulte Gateways NAT no Guia do usuário da Amazon VPC. Para ter mais informações, consulte the section called “Crie uma VPC”.

a. Em Virtual Private Cloud (VPC) ID (Nuvem Privada Virtual Private Cloud), escolha uma VPC na qual executar suas instâncias.

b. Em Sub-redes, selecione a sub-rede que será usada. Por padrão, todas as sub-redes dentro da VPC selecionadas estão disponíveis.

> **Note**
>
> AWS Batch no Amazon EC2 oferece suporte a Locais Zones. Para obter mais informações, consulte Zonas Locais no Amazon EC2 User Guide for Linux Instances e Amazon ECS clusters nas Zonas Locais, Wavelength Zones, e AWS Outposts no Amazon Elastic Container Service Developer Guide.

c. (Opcional) Para Security groups, escolha um grupo de segurança a ser anexado às suas instâncias. Por padrão, o security group padrão para sua VPC é escolhido.

8. Escolha Próxima página.
9. Para Revisar, reveja as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Criar ambiente de computação.

Para criar um ambiente de computação não gerenciado usando recursos EC2

1. Abra o AWS Batch console em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione o Região da AWS a ser usado.
3. Na página Ambientes computacionais selecione Criar.
4. Configure o ambiente.

a. Em configuração do ambiente de computação, escolha Amazon Elastic Compute Cloud (Amazon EC2).

b. Em Tipo de orquestração, escolha Não gerenciado.

5. Para Nome do ambiente de computação, especifique um nome exclusivo para seu ambiente de computação. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).
6. (Opcional) Em Função de serviço, escolha uma função que permita que o AWS Batch serviço faça chamadas para as operações de AWS API necessárias em seu nome. Para este exemplo, selecione AWSBatchServiceRole. Para obter mais informações, consulte the section called “Uso de perfis vinculadas ao serviço”.
7. Para Máximo de vCPUs, escolha o número máximo de vCPUs para o qual seu ambiente de computação deve aumentar, independentemente da demanda da fila de trabalhos.
8. (Opcional) Expanda as Tags. Para adicionar uma tag, escolha Add tag. Insira uma chave e um Valor opcional. Escolha Adicionar Tag. Para ter mais informações, consulte Marcando seus Recursos AWS Batch.
9. Escolha Próxima página.
10. Para Revisar, reveja as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Criar ambiente de computação.

Para criar um ambiente computacional gerenciado usando os recursos do Amazon EKS

1. Abra o AWS Batch console em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione o Região da AWS a ser usado.
3. No painel de navegação, escolha Ambientes de computação.
4. Escolha Criar.
5. Para a configuração do ambiente de computação, escolha Amazon Elastic Kubernetes Service (Amazon EKS).
6. Para Nome, especifique um nome exclusivo para seu ambiente de computação. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).
7. Para a Perfil de instância, escolha criar um novo perfil de instância ou use um existente que tenha as permissões de IAM necessárias anexadas.

> **Note**
>
> Para criar um ambiente computacional no AWS Batch console, escolha um perfil de instância que tenha as `eks:DescribeCluster` permissões `eks:ListClusters` e.

8. Para o cluster EKS, escolha um cluster existente do Amazon EKS.
9. Em Namespace, insira um Kubernetes namespace para agrupar seus AWS Batch processos no cluster.
10. (Opcional) Expanda as Tags. Escolha Adicionar tag e, em seguida, insira um par chave-valor.
11. Escolha Próxima página.
12. (Opcional) Para usar instâncias spot do EC2, ative Habilitar o uso de instâncias spot para usar instâncias spot do Amazon EC2.
13. (Somente Spot) Em Maximum % on-demand price, escolha a porcentagem máxima que o preço que uma instância spot deve ter em comparação com o preço sob demanda para esse tipo de instância antes que as instâncias sejam executadas. Por exemplo, se o preço máximo for 20%, o preço spot deverá estar abaixo de 20% do preço atual sob demanda para essa instância do EC2. Você sempre paga o menor preço (mercado) e nunca mais do que sua porcentagem máxima. Se você deixar esse campo em branco, o valor padrão será 100% do preço sob demanda.
14. (Somente spot) Para o perfil de frota spot, escolha o perfil IAM da frota spot do Amazon EC2 para o ambiente SPOT de computação.

> **Important**
>
> Esse perfil é necessário se a estratégia de alocação definida para o `BEST_FIT` ou se a estratégia de alocação não for especificada.

15. (Opcional) Para Mínimo de vCPUs, escolha o número mínimo de vCPUs que seu ambiente de computação deve manter, independentemente da demanda da fila de trabalhos.
16. (Opcional) Para Máximo de vCPUs, escolha o número máximo de vCPUs para o qual seu ambiente de computação deve aumentar, independentemente da demanda da fila de trabalhos.
17. Para Tipos de instância permitidos, escolha os tipos de instância do Amazon EC2 que podem ser iniciados. Você pode especificar famílias de instâncias para iniciar qualquer tipo de instância dentro dessas famílias (por exemplo `c5`, `c5n`, ou `p3`). Ou você pode especificar tamanhos específicos dentro de uma família (por exemplo,`c5.8xlarge`). Os tipos de instância Metal não

estão nas famílias de instâncias. Por exemplo, `c5` não inclui `c5.metal`. Você também pode optar por selecionar `optimal` tipos de instâncias (das famílias de instâncias C4, M4, e R4) porque precisa corresponder à demanda das suas filas de trabalho.

> **Note**
>
> Ao criar um ambiente de computação, os tipos de instância selecionados para ele devem compartilhar a mesma arquitetura. Por exemplo, você não pode misturar instâncias ARM e x86 no mesmo ambiente de computação.

> **Note**
>
> AWS Batch dimensiona as GPUs com base na quantidade necessária em suas filas de trabalho. Para usar o agendamento de GPU, o ambiente de computação deve incluir tipos de instância das famílias `p2`, `p3`, `p4`, `p5`, `g3`, `g3s`, `g4`, ou `g5`.

> **Note**
>
> Atualmente, `optimal` usa tipos de instância das famílias de instâncias C4, M4 e R4. Regiões da AWS Nesse caso, não há tipos de instância dessas famílias de instâncias, tipos de instância da C5M5, e famílias de R5 instâncias são usadas.

18. (Opcional) Expanda Configuração adicional.

    a. (Opcional) Em Grupo de posicionamento, insira um nome de grupo de posicionamento para agrupar recursos no ambiente de computação.

    b. Em Estratégia de alocação, escolha BEST_FIT_PROGRESSIVE.

    c. (Opcional) Para a Amazon Machine Images (AMIs) Configuration, escolha Adicionar configuração de imagens de máquina da Amazon (amis). Em seguida, escolha um tipo de imagem, insira uma substituição de ID de imagem e uma Kubernetesversão.

    > **Important**
    >
    > Para usar uma AMI personalizada, escolha o tipo de imagem e insira a ID da AMI personalizada na caixa de substituição de ID da imagem.

> **Note**
>
> Se a substituição do ID da imagem não for especificada para cada tipo de imagem, AWS Batch seleciona uma AMI otimizada recente do Amazon ECS. Se nenhum tipo de imagem for especificado, o padrão será Amazon Linux 2 para instância sem GPU e sem AWS Graviton.
>
> Amazon Linux 2
>
> Padrão para todas as famílias de instâncias AWS baseadas em Graviton (por exemplo,, `C6g` `M6gR6g`, `eT4g`) e pode ser usado para todos os tipos de instâncias que não sejam de GPU.
>
> Amazon Linux 2 (GPU)
>
> Padrão para todas as famílias de instâncias de GPU (por exemplo, `P4` `eG4`) e pode ser usado para todos os tipos de instância que não sejam AWS baseados em Graviton.

d. (Opcional) Em Launch template, escolha um modelo de execução existente.

e. (Opcional) Em Launch template version, insira **`$Default`**, **`$Latest`** ou um número de versão.

19. Escolha Próxima página.
20. Em Virtual Private Cloud (VPC) ID (Nuvem Privada Virtual Private Cloud), escolha uma VPC na qual executar as instâncias.
21. Para Subnets, escolha as sub-redes a serem usadas. Por padrão, todas as sub-redes dentro da VPC selecionadas estão disponíveis.

> **Note**
>
> AWS Batch no Amazon EKS é compatível com Locais Zones. Para obter mais informações, consulte Amazon EKS and AWS Local Zones no Guia do usuário do Amazon EKS.

22. (Opcional) Para Security groups, escolha um grupo de segurança a ser anexado às suas instâncias. Por padrão, o grupo de segurança padrão para sua VPC é escolhido.
23. Escolha Próxima página.

24. Para Revisar, reveja as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Criar ambiente de computação.

Modelo de ambiente de computação

O exemplo a seguir mostra um Modelo do Ambiente de Computação vazio. É possível usar esse modelo para criar seu ambiente de computação que pode ser salvo em um arquivo e usado com a opção `--cli-input-json` AWS CLI. Para obter mais informações sobre esses parâmetros, consulte CreateComputeEnvironment no AWS BatchAPI Reference.

```
{
    "computeEnvironmentName": "",
    "type": "UNMANAGED",
    "state": "DISABLED",
    "unmanagedvCpus": 0,
    "computeResources": {
        "type": "EC2",
        "allocationStrategy": "BEST_FIT_PROGRESSIVE",
        "minvCpus": 0,
        "maxvCpus": 0,
        "desiredvCpus": 0,
        "instanceTypes": [
            ""
        ],
        "imageId": "",
        "subnets": [
            ""
        ],
        "securityGroupIds": [
            ""
        ],
        "ec2KeyPair": "",
        "instanceRole": "",
        "tags": {
            "KeyName": ""
        },
        "placementGroup": "",
        "bidPercentage": 0,
        "spotIamFleetRole": "",
        "launchTemplate": {
            "launchTemplateId": "",
            "launchTemplateName": "",
```

```
            "version": ""
        },
        "ec2Configuration": [
            {
                "imageType": "",
                "imageIdOverride": "",
                "imageKubernetesVersion": ""
            }
        ]
    },
    "serviceRole": "",
    "tags": {
        "KeyName": ""
    },
    "eksConfiguration": {
        "eksClusterArn": "",
        "kubernetesNamespace": ""
    }
}
```

> **Note**
>
> Você pode gerar o modelo de ambiente de computação acima com o comando da AWS CLI a seguir.
>
> ```
> $ aws batch create-compute-environment --generate-cli-skeleton
> ```

Parâmetros de ambiente de computação

Os ambientes de computação são divididos em vários componentes básicos: nome, tipo e estado do ambiente de computação, definição do recurso de computação (se for um ambiente de computação gerenciado), a configuração do Amazon EKS (se usar recursos do Amazon EKS), o perfil de serviço a ser usado para fornecer permissões do IAM ao AWS Batch e as tags do ambiente de computação.

Tópicos

- Recursos de computação
- Configuração do Amazon EKS
- Perfil de serviço
- Tags

Nome do ambiente de computação

`computeEnvironmentName`

O nome do seu ambiente de computação. Os nomes podem ter até 128 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).

Tipo: sequência

Obrigatório: Sim

Tipo

`type`

O tipo do ambiente de computação. Selecione `MANAGED` para que o AWS Batch gerencie os recursos de computação do EC2 ou do Fargate que você definir. Para ter mais informações, consulte Recursos de computação. Escolha o `UNMANAGED` para gerenciar seus próprios recursos de computação do EC2.

Tipo: string

Valores válidos: `MANAGED` | `UNMANAGED`

Obrigatório: sim

State

`state`

O estado do ambiente de computação.

Se o estado for `ENABLED`, o agendador do AWS Batch não tentará colocar trabalhos no ambiente. Esses trabalhos são de uma fila de trabalhos associada nos recursos de computação. Se o

ambiente de computação for gerenciado, as instâncias aumentam a escala horizontalmente ou verticalmente de maneira automática de acordo com a demanda da fila de trabalhos.

Se o estado for `DISABLED`, o programador do AWS Batch não tentará colocar trabalhos no ambiente. Os trabalhos que estiverem em estado `STARTING` ou `RUNNING` continuam progredindo normalmente. Os ambientes de computação gerenciados que estiverem no estado `DISABLED` não são dimensionados.

> **Note**
>
> Ambientes computacionais em um estado `DISABLED` podem continuar incorrendo em cobranças. Para evitar cobranças adicionais, desative e exclua o ambiente de computação. Para obter mais informações, consulte DeleteComputeEnvironment na Referência da API da AWS Batch e Evitar cobranças inesperadas no Guia do usuário do AWS Billing.

Quando uma instância está ociosa, a instância tem a escala reduzida verticalmente para o valor `minvCpus`. No entanto, o tamanho da instância não é alterado. Por exemplo, considere uma instância `c5.8xlarge` com um valor `minvCpus` de `4` e um `desiredvCpus` valor de `36`. Essa instância não reduz a escala verticalmente uma instância `c5.large`.

Tipo: string

Valores válidos: `ENABLED` | `DISABLED`

Obrigatório: não

Recursos de computação

`computeResources`

Detalhes dos recursos de computação gerenciados pelo ambiente de computação. Para ter mais informações, consulte Ambiente de computação.

Tipo: objeto ComputeResource

Obrigatório: esse parâmetro é obrigatório para ambientes de computação gerenciados

`type`

O tipo do ambiente de computação. Você pode optar por usar instâncias sob demanda do EC2 (`EC2`) e instâncias spot do EC2 (`SPOT`) ou usar a capacidade do Fargate (`FARGATE`) e a capacidade do Fargate Spot (`FARGATE_SPOT`) em seu ambiente de computação gerenciado. Se você escolher `SPOT`, também deverá especificar uma função de frota spot do Amazon EC2 com o parâmetro `spotIamFleetRole`. Para ter mais informações, consulte Perfil de frota spot Amazon EC2.

Valores válidos: `EC2` | `SPOT` | `FARGATE` | `FARGATE_SPOT`

Obrigatório: sim

`allocationStrategy`

A estratégia de alocação a ser usada para o recurso de computação caso instâncias suficientes do tipo de instância do EC2 mais adequado não possam ser alocadas. Isso pode ser devido à disponibilidade do tipo de instância na Região da AWS ou aos limites de serviço do Amazon EC2. Para ter mais informações, consulte Estratégias de alocação.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

`BEST_FIT` (padrão)

O AWS Batch seleciona um tipo de instância que seja mais adequado às necessidades dos trabalhos com preferência para o tipo de instância de menor custo. Se as instâncias adicionais do tipo de instância selecionado não estiverem disponíveis, o AWS Batch aguardará até que as instâncias adicionais estejam disponíveis. Se não houver instâncias suficientes disponíveis ou você estiver atingindo os limites de serviço do Amazon EC2, não serão executados trabalhos adicionais até que os trabalhos em execução no momento estejam concluídos. Essa estratégia de alocação mantém os custos mais baixos, mas pode limitar a escalabilidade. Se você estiver usando frotas spot com `BEST_FIT`, o perfil do IAM da frota spot deverá ser especificada. Os recursos de computação que usam uma estratégia de alocação `BEST_FIT` não oferecem suporte a atualizações de infraestrutura e não podem atualizar alguns parâmetros. Para ter mais informações, consulte Criação de um ambiente de computação.

> **Note**
>
> O `BEST_FIT` não é compatível com ambientes de computação que usam recursos do Amazon EKS.

`BEST_FIT_PROGRESSIVE`

Use tipos de instância adicionais que sejam grandes o suficiente para atender aos requisitos dos trabalhos na fila. Prefira tipos de instância com um custo menor para cada unidade vCPU. Se as instâncias adicionais dos tipos de instância selecionados anteriormente não estiverem disponíveis, o AWS Batch selecionará novos tipos de instância.

`SPOT_CAPACITY_OPTIMIZED`

(Disponível apenas para recursos de computação de instância spot) Use tipos de instância adicionais que sejam grandes o suficiente para atender aos requisitos dos trabalhos na fila. Prefira tipos de instância com menor probabilidade de serem interrompidos.

`SPOT_PRICE_CAPACITY_OPTIMIZED`

(Disponível apenas para recursos de computação de instância spot) A estratégia de alocação otimizada para preço e capacidade analisa o preço e a capacidade para selecionar os grupos de instâncias spot com menor probabilidade de interrupção e com o preço mais baixo possível.

> **Note**
>
> Em vez disso, recomendamos utilizar `SPOT_PRICE_CAPACITY_OPTIMIZED` em vez de `SPOT_CAPACITY_OPTIMIZED` na maioria das instâncias.

Com as estratégias `BEST_FIT_PROGRESSIVE`, `SPOT_CAPACITY_OPTIMIZED` e `SPOT_PRICE_CAPACITY_OPTIMIZED` usando instâncias sob demanda ou spot, e a estratégia `BEST_FIT` usando instâncias spot, o AWS Batch pode precisar exceder as `maxvCpus` para atender aos requisitos de capacidade. Nesse caso, o AWS Batch nunca excederá `maxvCpus` em mais de uma única instância.

Valores válidos: `BEST_FIT` | `BEST_FIT_PROGRESSIVE` | `SPOT_CAPACITY_OPTIMIZED` | `SPOT_PRICE_CAPACITY_OPTIMIZED`

Obrigatório: não

`minvCpus`

O número mínimo de vCPUs que um ambiente mantém mesmo se um ambiente de computação estiver `DISABLED`.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Tipo: Inteiro

Obrigatório: não

`maxvCpus`

O número máximo de vCPUs que o ambiente de computação do AWS Batch pode suportar.

> **Note**
>
> Com as estratégias de alocação `BEST_FIT_PROGRESSIVE`, `SPOT_CAPACITY_OPTIMIZED` e `SPOT_PRICE_CAPACITY_OPTIMIZED` usando instâncias sob demanda ou spot, e a estratégia `BEST_FIT` usando instâncias spot, o AWS Batch pode precisar exceder as `maxvCpus` para atender aos requisitos de capacidade. Nesse caso, o AWS Batch nunca excederá `maxvCpus` em mais de uma única instância. Por exemplo, o AWS Batch não usa mais do que uma única instância entre as especificadas no ambiente de computação.

Tipo: inteiro

Obrigatório: não

`desiredvCpus`

O número desejado de vCPUs do no ambiente de computação. O AWS Batch modifica esse valor entre os valores mínimo e máximo, com base na demanda da fila de trabalhos.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Tipo: Inteiro

Obrigatório: não

`instanceTypes`

Os tipos de instância que podem ser executados. Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate. Não especifique. Você pode especificar famílias de instâncias para iniciar qualquer tipo de instância nessas famílias (por exemplo, `c5`, `c5n` ou `p3`). Ou você pode especificar tamanhos específicos dentro de uma família (como `c5.8xlarge`). Observe que os tipos de instância metal não estão nas famílias de instâncias (por exemplo, `c5` não inclui `c5.metal`). Você também pode escolher `optimal` para selecionar tipos de instância (das famílias de instâncias C4, M4 e R4) que correspondam à demanda das filas de trabalho.

> **Note**
>
> Ao criar um ambiente de computação, os tipos de instância selecionados para ele devem compartilhar a mesma arquitetura. Por exemplo, você não pode misturar instâncias ARM e x86 no mesmo ambiente de computação.

> **Note**
>
> Atualmente, `optimal` usa tipos de instância das famílias de instâncias C4, M4 e R4. Em Regiões da AWS que não têm tipos de instância dessas famílias de instâncias, são usados os tipos de instância das famílias de instâncias C5, M5 e R5.

Tipo: matriz de strings

Obrigatório: sim

`imageId`

Esse parâmetro está suspenso.

O ID da imagem de máquina da Amazon (AMI) usado para as instâncias iniciadas no ambiente de computação. Este parâmetro é substituído pelo membro `imageIdOverride` da estrutura `Ec2Configuration`.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

> **Note**
>
> A AMI que você escolher para um ambiente de computação deve corresponder à arquitetura dos tipos de instância que você deseja usar para este ambiente. Por exemplo, se o ambiente de computação usar tipos de instância A1, a AMI de recursos de computação escolhida deverá oferecer suporte a instâncias Arm. O Amazon ECS vende as versões x86 e Arm da Amazon ECS optimized Amazon Linux 2 AMI. Para obter mais informações, consulte AMI do Amazon Linux 2 otimizada para Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

Tipo: string

Exigido: não

`subnets`

As sub-redes de VPC em que os recursos de computação são iniciados. Essas sub-redes devem estar na mesma VPC. Os recursos de computação do Fargate podem conter um máximo de 16 sub-redes. Para obter mais informações, consulte VPCs e sub-redes no Guia do usuário da Amazon VPC.

> **Note**
>
> O AWS Batch no Amazon EC2 e o AWS Batch no Amazon EKS são compatíveis com Local Zones. Para obter mais informações, consulte Zonas locais no Guia do usuário do Amazon EC2 para instâncias Linux, Amazon EKS e zonas locais da AWS no Guia do usuário do Amazon EKS e Clusters do Amazon ECS em Local Zones, zonas do Wavelength e AWS Outposts no Guia do desenvolvedor do Amazon Elastic Container Service.

AWS Batch no Fargate atualmente não é compatível com Local Zones.

Ao atualizar ambientes de computação, se você fornecer uma lista vazia de sub-redes VPC, o comportamento resultante será diferente entre os recursos de computação do Fargate e do EC2. Para recursos de computação do Fargate, fornecer uma lista vazia é tratado como se esse parâmetro não tivesse sido especificado e nenhuma alteração tivesse sido feita. Para recursos de computação do EC2, fornecer uma lista vazia remove as sub-redes da VPC do recurso de computação. Se você alterar as sub-redes da VPC, será necessária uma atualização da infraestrutura do ambiente de computação. Esse é o caso dos recursos computacionais do Fargate e do EC2. Para ter mais informações, consulte Criação de um ambiente de computação.

Tipo: matriz de strings

Obrigatório: sim

`securityGroupIds`

Os grupos de segurança do Amazon EC2 associados às instâncias executadas no ambiente de computação. Um ou mais grupos de segurança devem ser especificados, no `securityGroupIds` ou usando um modelo de execução referenciado em `launchTemplate`. Esse parâmetro é necessário para trabalhos que estão em execução nos recursos do Fargate e deve conter pelo menos um grupo de segurança. (O Fargate não suporta modelos de execução.) Se os grupos de segurança forem especificados usando `securityGroupIds` e `launchTemplate`, os valores em `securityGroupIds` serão usados.

Ao atualizar ambientes de computação, se você fornecer uma lista vazia de grupos de segurança, o comportamento resultante será diferente entre os recursos de computação do Fargate e do EC2. Para recursos de computação do Fargate, fornecer uma lista vazia é tratado como se esse parâmetro não tivesse sido especificado e nenhuma alteração tivesse sido feita. Para recursos de computação do EC2, fornecer uma lista vazia remove os grupos de segurança do recurso de computação. Se você alterar os grupos de segurança, será necessária uma atualização da infraestrutura do ambiente de computação. Esse é o caso dos recursos computacionais do Fargate e do EC2. Para ter mais informações, consulte Criação de um ambiente de computação.

Tipo: matriz de strings

Obrigatório: sim

`ec2KeyPair`

O par de chaves do EC2 que é usado para as instâncias iniciadas no ambiente de computação. Você pode usar esse par de chaves para fazer login em suas instâncias com SSH. Ao atualizar um ambiente de computação, se você alterar o par de chaves do EC2, uma atualização da infraestrutura do ambiente de computação será necessária. Para ter mais informações, consulte Criação de um ambiente de computação.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Tipo: string

Exigido: não

`instanceRole`

O perfil de instância do Amazon ECS para anexar às instâncias do Amazon EC2 em um ambiente de computação. Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate. Não especifique. Você pode especificar o nome abreviado ou completo do Nome de recurso da Amazon (ARN) de um perfil de instância. Por exemplo, o `ecsInstanceRole` ou o `arn:aws:iam::`*`aws_account_id`*`:instance-profile/ecsInstanceRole`. Para ter mais informações, consulte Perfil de instância do Amazon ECS.

Ao atualizar um ambiente de computação, alterar essa configuração requer uma atualização da infraestrutura do ambiente de computação. Para ter mais informações, consulte Criação de um ambiente de computação.

Tipo: Sequência

Exigido: não

`tags`

Tags de pares de chave-valor a serem aplicadas às instâncias do EC2 que são iniciadas no ambiente de computação. Por exemplo, você pode especificar `"Name": "AWS Batch Instance - C4OnDemand"` como uma tag para que cada instância do seu ambiente de computação tenha esse nome. Isso é útil para reconhecer suas instâncias do AWS Batch no

console do Amazon EC2. Essas tags não são vistas quando a operação da API AWS Batch ListTagsForResource for usada.

Ao atualizar um ambiente de computação, se você alterar as chaves do EC2, uma atualização da infraestrutura do ambiente de computação será necessária. Para ter mais informações, consulte Criação de um ambiente de computação.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Tipo: mapa de string para string

Obrigatório: Não

`placementGroup`

O placement group do Amazon EC2 para associar aos seus recursos de computação. Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate. Não especifique. Se você pretende enviar tarefas em paralelo de vários nós para seu ambiente de computação, considere a criação de um grupo com posicionamento em cluster e associá-lo aos seus recursos de computação. Isso mantém a sua tarefa paralela de vários nós em um agrupamento lógico de instâncias dentro de uma única zona de disponibilidade com alto potencial de fluxo de rede. Para obter mais informações, consulte Grupo de Posicionamento no Manual do Usuário para instâncias do Linux do Amazon EC2.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Tipo: string

Exigido: não

`bidPercentage`

A porcentagem máxima que o preço de uma instância spot do EC2 pode ter em comparação com o preço do On-Demand para esse tipo de instância antes que as instâncias sejam iniciadas. Por exemplo, se o percentual máximo for 20%, o preço spot deverá ser menos de

20% do preço sob demanda atual para essa instância do EC2. Você sempre paga o menor preço (mercado) e nunca mais do que sua porcentagem máxima. Se você deixar esse campo em branco, o valor padrão será 100% do preço sob demanda. Para a maioria dos casos de uso, recomendamos deixar este campo vazio.

Ao atualizar um ambiente de computação, se você alterar a porcentagem do lance, uma atualização da infraestrutura do ambiente de computação será necessária. Para ter mais informações, consulte Criação de um ambiente de computação.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Obrigatório: não

`spotIamFleetRole`

O nome de recurso da Amazon (ARN) do perfil do IAM da frota spot do Amazon EC2 aplicado a um ambiente de computação `SPOT`. Essa função é necessária se a estratégia de alocação definida para o `BEST_FIT` ou se a estratégia de alocação não for especificada. Para ter mais informações, consulte Perfil de frota spot Amazon EC2.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

> **Important**
>
> Para marcar suas instâncias spot na criação, a função Spot Fleet IAM especificada aqui deve usar a política gerenciada mais recente do AmazonEC2 SpotFleetTaggingRole. A política SpotFleetRole gerenciada anteriormente recomendada do AmazonEC2 não tem as permissões necessárias para marcar instâncias spot. Para ter mais informações, consulte Instâncias spot sem tags na criação.

Tipo: string

Obrigatório: esse parâmetro é obrigatório para ambientes de computação `SPOT`.

`launchTemplate`

Um modelo de execução opcional para associar aos recursos de computação. Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate. Não especifique. Todos os outros parâmetros de recursos de computação que você especifica em uma operação de API CreateComputeEnvironment ou UpdateComputeEnvironment substituem os mesmos parâmetros no modelo de execução. Para usar um modelo de execução, você deve especificar na solicitação o ID ou o nome do modelo de execução, mas não ambos. Para ter mais informações, consulte Suporte a modelo de execução.

Ao atualizar um ambiente computacional, para remover o modelo de execução personalizado e usar o modelo de execução padrão, defina o membro `launchTemplateId` ou `launchTemplateName` da especificação do modelo de execução como uma string vazia. A remoção do modelo de inicialização de um ambiente computacional não remove a AMI especificada no modelo de inicialização, caso tenha sido o usado. Para atualizar a AMI selecionada de um modelo de inicialização, o parâmetro `updateToLatestImageVersion` deve ser definido como `true`. Ao atualizar um ambiente de computação, se você alterar o modelo de execução, uma atualização da infraestrutura do ambiente de computação será necessária. Para ter mais informações, consulte Criação de um ambiente de computação.

Tipo: LaunchTemplateSpecification

objeto

Obrigatório: não

`launchTemplateId`

O ID do modelo de execução.

Tipo: string

Exigido: não

`launchTemplateName`

O nome do modelo de execução.

Tipo: string

Exigido: não

`version`

O número da versão do modelo de execução, `$Latest` ou `$Default`.

Se o valor for `$Latest`, a versão mais recente usará o modelo de execução. Se o valor for `$Default`, a versão padrão usará o modelo de execução. Durante uma atualização da infraestrutura, se `$Latest` ou `$Default` tiver sido especificado para o ambiente de computação, o AWS Batch reavaliará a versão do modelo de execução e poderá usar uma versão diferente do modelo de execução. Isso ocorre mesmo se o modelo de execução não tiver sido especificado na atualização.

Padrão: `$Default`.

Tipo: string

Exigido: não

`ec2Configuration`

Fornece informações usadas para selecionar Imagens de máquina da Amazon (AMIs) para instâncias do EC2 no ambiente de computação. Se `Ec2Configuration` não estiver especificado, o padrão será Amazon Linux 2 (`ECS_AL2`). Antes de 31 de março de 2021, esse padrão era Amazon Linux (`ECS_AL1`) para instâncias não GPU e não AWS Graviton.

Ao atualizar um ambiente de computação, alterar esse parâmetro requer uma atualização da infraestrutura do ambiente de computação. Para ter mais informações, consulte Criação de um ambiente de computação.

> **Note**
>
> Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate.

Tipo: matriz de objetos Ec2Configuration

Obrigatório: não

`imageIdOverride`

ID da AMI que é usado para instâncias executadas no ambiente de computação que correspondem ao tipo de imagem. Essa configuração substitui o `imageId` definido no objeto `computeResource`.

Tipo: string

Exigido: não

`imageKubernetesVersion`

A versão do Kubernetes para o ambiente de computação. Se você não especificar um valor, a versão mais recente compatível com o AWS Batch será usada.

Tipo: String

Restrições de comprimento: comprimento mínimo 1. Tamanho máximo de 256.

Obrigatório: não

`imageType`

O tipo de imagem para corresponder ao tipo de instância para selecionar uma AMI. Os valores compatíveis são diferentes para os recursos do `ECS` e do `EKS`.

ECS

Se o parâmetro `imageIdOverride` não for especificado, será usada uma AMI do Amazon Linux 2 otimizada para Amazon ECS recente (`ECS_AL2`). Se um novo tipo de imagem for especificado em uma atualização, mas nenhuma `imageId` nem um parâmetro `imageIdOverride` forem especificados, a AMI otimizada do Amazon ECS mais recente para esse tipo de imagem com suporte no AWS Batch será usada.

ECS_AL2

Amazon Linux 2: padrão para todas as famílias de instâncias não GPU.

ECS_AL2_NVIDIA

Amazon Linux 2 (GPU): padrão para todas as famílias de instâncias GPU (por exemplo, `P4` e `G4`) e pode ser usado para todos os tipos de instâncias não baseadas no AWS Graviton.

ECS_AL1

Amazon Linux. O Amazon Linux alcançou end-of-life o suporte padrão. Para obter mais informações, consulte AMI do Amazon Linux.

EKS

Se o parâmetro `imageIdOverride` não for especificado, será usada uma AMI do Amazon Linux otimizada para o Amazon EKS (`EKS_AL2`). Se um novo tipo de imagem

for especificado em uma atualização, mas nem um `imageId` nem um parâmetro `imageIdOverride` forem especificados, será usada a AMI otimizada para o Amazon ECS mais recente para esse tipo de imagem que seja compatível com o AWS Batch.

EKS_AL2

Amazon Linux 2: padrão para todas as famílias de instâncias não GPU.

EKS_AL2_NVIDIA

Amazon Linux 2 (acelerado): padrão para todas as famílias de instâncias de GPU (por exemplo, `P4` e `G4`) e pode ser usado para todos os tipos de instâncias não baseadas no AWS Graviton.

Tipo: String

Restrições de comprimento: comprimento mínimo 1. Tamanho máximo de 256.

Obrigatório: sim

Configuração do Amazon EKS

A configuração de cluster do Amazon EKS que é compatível com o ambiente de computação AWS Batch. O cluster deve existir antes que o ambiente de computação possa ser criado.

`eksClusterArn`

O nome do recurso da Amazon (ARN) do cluster do Amazon EKS. Um exemplo é `arn:aws:eks:`*`us-east-1`*`:`*`123456789012`*`:cluster/`*`ClusterForBatch`*.

Tipo: sequência

Obrigatório: Sim

`kubernetesNamespace`

O namespace do cluster do Amazon EKS. O AWS Batch gerencia pods nesse namespace. O valor não pode ser deixado vazio nem nulo. Deve ter menos de 64 caracteres, não pode ser definido como `default`, não pode começar com "`kube-`" e deve corresponder a esta expressão regular: `^[a-z0-9]([-a-z0-9]*[a-z0-9])?$`. Para obter mais informações, consulte Namespaces na documentação do Kubernetes.

Tipo: sequência

Obrigatório: Sim

Tipo: EksConfigurationObjeto

Obrigatório: não

Perfil de serviço

`serviceRole`

O nome do recurso da Amazon (ARN) completo do perfil do IAM que permite ao AWS Batch fazer chamadas para outros serviços da AWS em seu nome. Para ter mais informações, consulte Usando funções vinculadas a serviços para AWS Batch. Recomendamos não especificar o perfil de serviço. Assim, o AWS Batch usa um perfil vinculado ao serviço do AWSServiceRoleForBatch.

> **Important**
>
> Se sua conta já criou a função vinculada a serviços do AWS Batch (AWSServiceRoleForBatch), essa função é usada por padrão para seu ambiente de computação, a menos que você especifique uma função aqui. Se a função vinculada a serviços do AWS Batch não existir na sua conta e nenhuma função for especificada aqui, o serviço tentará criar a função vinculada ao serviço do AWS Batch na sua conta. Para obter mais informações sobre a função vinculada ao serviço do AWSServiceRoleForBatch, consulte Permissões de função vinculadas ao serviço para AWS Batch.
>
> Se o ambiente de computação for criado usando a função vinculada ao serviço AWSServiceRoleForBatch, ele não poderá ser alterado para usar um perfil do IAM normal. Da mesma forma, se o ambiente de computação for criado com um perfil do IAM normal, ele não poderá ser alterado para usar uma função vinculada ao serviço AWSServiceRoleForBatch. Para atualizar os parâmetros do ambiente de computação que exigem uma atualização de infraestrutura para serem alterados, deve ser usada a função vinculada ao serviço AWSServiceRoleForBatch. Para ter mais informações, consulte Criação de um ambiente de computação.

Se a função especificada tiver um caminho diferente de `/`, certifique-se de especificar a função completa de ARN (recomendado) ou prefixar o nome da função com o caminho.

> **Note**
>
> Dependendo de como o perfil de serviço do AWS Batch foi criada, o Nome de recurso da Amazon (ARN) poderá conter o prefixo de caminho `service-role`. Ao especificar somente o nome do perfil de serviço, o AWS Batch assumirá que o ARN não usa o prefixo de caminho `service-role`. Por isso, recomendamos que você especifique o ARN completo do perfil de serviço ao criar ambientes de computação.

Tipo: string

Exigido: não

Tags

`tags`

Tags de pares de chave-valor a serem associadas ao ambiente de computação. Para ter mais informações, consulte Marcando seus Recursos AWS Batch.

Tipo: Mapa de string para string

Obrigatório: Não

Configuração EC2

AWS Batch usa AMIs otimizadas do Amazon ECS para ambientes computacionais EC2 e EC2 Spot. O padrão é o Amazon Linux 2 (`ECS_AL2`). Antes de 31 de março de 2021, esse padrão era Amazon Linux (`ECS_AL1`) para instâncias sem GPU e sem AWS Graviton.

> **Note**
>
> AWS Batch também é compatível com Amazon Linux 2023.

O Amazon Linux AMI (também chamado de Amazon Linux 1) chegou ao fim em 31 de dezembro de 2023. AWS Batch encerrou o suporte para o Amazon Linux AMI, pois não receberá nenhuma

atualização de segurança ou correção de erros a partir de 1º de janeiro de 2024. Para obter mais informações sobre o Amazon Linux end-of-life, consulte as perguntas frequentes da AL.

Recomendamos que você atualize os ambientes computacionais existentes baseados no Amazon Linux para o Amazon Linux 2023 para evitar interrupções imprevistas na carga de trabalho e continuar recebendo atualizações de segurança e outras.

Seus ambientes computacionais usando o Amazon Linux AMI podem continuar funcionando além da end-of-life data de 31 de dezembro de 2023. No entanto, esses ambientes computacionais não receberão mais novas atualizações de software, patches de segurança ou correções de bugs da AWS. É sua responsabilidade manter esses ambientes computacionais na Amazon Linux AMI posteriormente end-of-life. Recomendamos migrar ambientes AWS Batch computacionais para o Amazon Linux 2023 ou o Amazon Linux 2 para manter o desempenho e a segurança ideais.

Para obter ajuda na migração AWS Batch do Amazon Linux AMI para o Amazon Linux 2023 ou Amazon Linux 2, consulte Atualização de ambientes computacionais - AWS Batch

Estratégias de alocação

Quando um ambiente de computação gerenciado é criado, AWS Batch seleciona os tipos de instância `instanceTypes` especificados que melhor atendem às necessidades dos trabalhos. A estratégia de alocação define o comportamento quando AWS Batch precisa de capacidade adicional. Este parâmetro não é aplicável a trabalhos executados em recursos do Fargate. Não especifique este parâmetro.

`BEST_FIT` (padrão)

> AWS Batch seleciona o tipo de instância que melhor se adapta às necessidades dos trabalhos, preferindo o tipo de instância de menor custo. Se instâncias adicionais do tipo de instância selecionado não estiverem disponíveis, AWS Batch aguarda até que as instâncias adicionais estejam disponíveis. Se não houver instâncias suficientes disponíveis ou se o usuário estiver atingindo os limites de cotas de serviço do Amazon EC2, não serão executados trabalhos adicionais até que os trabalhos em execução no momento estejam concluídos. Essa estratégia de alocação mantém os custos mais baixos, mas pode limitar a escalabilidade. Se você estiver usando frota spot com o `BEST_FIT`, o perfil do IAM de frota spot deve ser especificada. O `BEST_FIT` não é compatível com a atualização de ambientes de computação. Para ter mais informações, consulte Criação de um ambiente de computação.

> **Note**
>
> AWS Batch gerencia AWS recursos em sua conta. Ambientes de computação com a estratégia de alocação BEST_FIT originalmente utilizavam configurações de lançamento por padrão. No entanto, o uso de configurações de lançamento com novas AWS contas será restrito ao longo do tempo. Portanto, a partir do final de abril de 2024, os ambientes computacionais BEST_FIT recém-criados usarão modelos de lançamento como padrão. Se sua função de serviço não tiver permissões para gerenciar modelos de lançamento, AWS Batch poderá continuar a utilizar as configurações de lançamento. Os ambientes computacionais existentes continuarão usando as configurações de lançamento.

`BEST_FIT_PROGRESSIVE`

AWS Batch seleciona tipos de instância adicionais que são grandes o suficiente para atender aos requisitos dos trabalhos na fila. Os tipos de instância com um custo menor para cada unidade vCPU são preferidos. Se as instâncias adicionais dos tipos de instância selecionados anteriormente não estiverem disponíveis, o AWS Batch selecionará novos tipos de instância.

`SPOT_CAPACITY_OPTIMIZED`

AWS Batch seleciona um ou mais tipos de instância grandes o suficiente para atender aos requisitos dos trabalhos na fila. Os tipos de instância com menor probabilidade de serem interrompidos são preferidos. Essa estratégia de alocação só está disponível para recursos de computação de instâncias spot.

`SPOT_PRICE_CAPACITY_OPTIMIZED`

A estratégia de alocação otimizada para preço e capacidade analisa o preço e a capacidade para selecionar os grupos de instâncias spot com menor probabilidade de interrupção e com o preço mais baixo possível. Essa estratégia de alocação só está disponível para recursos de computação de instâncias spot.

> **Note**
>
> Em vez disso, recomendamos utilizar `SPOT_PRICE_CAPACITY_OPTIMIZED` em vez de `SPOT_CAPACITY_OPTIMIZED` na maioria das instâncias.

As estratégias `BEST_FIT_PROGRESSIVE` e `BEST_FIT` usam instâncias spot ou sob demanda, e as estratégias `SPOT_CAPACITY_OPTIMIZED` e `SPOT_PRICE_CAPACITY_OPTIMIZED` usam instâncias spot. No entanto, AWS Batch talvez seja necessário exceder `maxvCpus` para atender aos seus requisitos de capacidade. Nesse caso, AWS Batch nunca `maxvCpus` excede em mais de uma única instância.

Criação de um ambiente de computação

Depois de criar um ambiente de computação que usa recursos do EC2, você pode atualizar muitas das configurações do ambiente de computação diretamente. No entanto, a alteração de algumas das configurações exige que AWS Batch substitua as instâncias no ambiente de computação.

Para ambientes de computação que usam recursos do Fargate, você pode atualizar o seguinte.

- `securityGroupIds`
- `subnets`
- `desiredvCpus`
- `maxvCpus`
- `minvCpus`

AWS Batch tem dois mecanismos de atualização. A primeira é uma atualização de escalabilidade em que as instâncias são adicionadas ou removidas do ambiente de computação. A segunda é uma atualização de infraestrutura em que as instâncias no ambiente de computação são substituídas. Uma atualização de infraestrutura leva muito mais tempo do que uma atualização de escalabilidade.

Se você atualizar ambientes de computação com AWS Batch, alterando somente essas configurações, causará uma atualização de escalabilidade: vCPUs desejadas (`desiredvCpus`), vCPUs máximas (`maxvCpus`), vCPUs mínimas (`minvCpus`), função de serviço (`serviceRole`) e estado (`state`).

> **Note**
>
> Quando você atualiza a configuração `desiredvCpus`, o valor deve estar entre os valores `minvCpus` e `maxvCpus`.

Além disso, o valor atualizado `desiredvCpus` deve ser maior que ou igual ao `desiredvCpus` atual. Para obter mais informações, consulte the section called “Mensagem de erro ao atualizar a configuração `desiredvCpus`”.

Se alguma das configurações a seguir for alterada em uma ação de API UpdateComputeEnvironment, AWS Batch iniciará uma atualização de infraestrutura. Uma atualização de infraestrutura exige que a função de serviço seja definida como AWSServiceRoleForBatch (o padrão) e que a estratégia de alocação seja `BEST_FIT_PROGRESSIVE` `SPOT_CAPACITY_OPTIMIZED` ou `SPOT_PRICE_CAPACITY_OPTIMIZED`. `BEST_FIT` não é compatível. Com exceção da função de serviço, todas as configurações que podem ser alteradas para uma atualização de escalabilidade também podem ser alteradas para uma atualização de infraestrutura.

Note

Recomendamos que você use `SPOT_PRICE_CAPACITY_OPTIMIZED` em vez de `SPOT_CAPACITY_OPTIMIZED` na maioria das instâncias.

Durante uma atualização da infraestrutura, o status do ambiente de computação muda para `UPDATING`. Novas instâncias são lançadas usando as configurações atualizadas. Novos trabalhos são agendados nas novas instâncias. Os trabalhos em execução no momento são despachados de acordo com a política de atualização da infraestrutura. Para obter mais informações, consulte UpdateComputeEnvironment e UpdatePolicy na AWS Batch API Reference.

No tipo de dado `UpdatePolicy`, considere os seguintes cenários:

Note

Nesses cenários, o seguinte é verdadeiro. Quando uma instância é encerrada, os trabalhos em execução são interrompidos. Por padrão, esses trabalhos não são repetidos. Para repetir um desses trabalhos após o encerramento de uma instância, configure uma estratégia de repetição do trabalho. Para obter mais informações, consulte the section called “Repetições de trabalho automatizadas” no Guia de Usuário AWS Batch.

- Se a configuração `terminateJobsOnUpdate` estiver definida como `true`, os trabalhos em execução serão encerrados durante uma atualização da infraestrutura. A configuração `jobExecutionTimeoutMinutes` é ignorada.
- Se a configuração `terminateJobsOnUpdate` estiver definida como `false`, os trabalhos poderão ser executados por mais tempo após a atualização da infraestrutura. Esse tempo adicional é configurado na configuração `jobExecutionTimeoutMinutes`. Por padrão, a configuração `jobExecutionTimeoutMinutes` é 30 minutos.

À medida que a capacidade se torna disponível no ambiente de computação, novas instâncias são lançadas com as configurações atualizadas e os trabalhos são iniciados nas novas instâncias. Quando todos os trabalhos são concluídos em instâncias com as configurações antigas, as instâncias antigas são encerradas. O que significa que a capacidade se tornar disponível é que o número desejado de vCPUs está abaixo do número máximo de vCPUs em pelo menos tantas vCPUs quanto exigido pelo menor tipo de instância.

Atualizações da infraestrutura

É necessária uma atualização da infraestrutura para alterar algumas configurações de um ambiente de computação. Se alguma das configurações a seguir for alterada, uma atualização de infraestrutura será iniciada:

> **Important**
>
> O ambiente de computação deve usar a função vinculada ao serviço AWSServiceRoleForBatch para fazer alterações que exijam uma atualização de infraestrutura.
> Se o ambiente de computação usa uma função vinculada ao serviço, ela não pode ser alterada para usar uma função normal do IAM. Da mesma forma, se o ambiente de computação tiver uma função normal do IAM, ela não poderá ser alterada para usar uma função vinculada ao serviço. Portanto, você só pode realizar atualizações de infraestrutura em ambientes de computação que foram criados usando uma função vinculada a serviços.

- A estratégia de alocação (`allocationStrategy`, deve ser `BEST_FIT_PROGRESSIVE`, `SPOT_CAPACITY_OPTIMIZED` ou `SPOT_PRICE_CAPACITY_OPTIMIZED`. Se a estratégia de alocação original for `BEST_FIT`, as atualizações de infraestrutura não serão suportadas.)

> **Note**
>
> Recomendamos que você use `SPOT_PRICE_CAPACITY_OPTIMIZED` em vez de `SPOT_CAPACITY_OPTIMIZED` na maioria das instâncias.

- Porcentagem de oferta (`bidPercentage`)
- Configuração do EC2 (`ec2Configuration`)
- Par de chaves (`ec2KeyPair`)
- ID da imagem (`imageId`)
- Função da instância (`instanceRole`)
- Tipos de instância (`instanceTypes`)
- Modelo de execução (`launchTemplate`)
- Grupo de posicionamento (`placementGroup`)
- Grupos de segurança (`securityGroupIds`)
- Sub-redes VPC(`subnets`)
- Etiquetas EC2 (`tags`)
- Tipo de ambiente de computação (`type`, pode ser um dos `EC2` ou `SPOT`)
- Se deve atualizar para a AMI mais recente que é suportada por AWS Batch durante uma atualização de infraestrutura `updateToLatestImageVersion`

Atualização do ID da AMI

Durante uma atualização de infraestrutura, o ID da AMI do ambiente de computação pode mudar, dependendo se as AMIs estão especificadas em qualquer uma dessas três configurações. As AMIs são especificadas em `imageId` (em `computeResources`), `imageIdOverride` (em `ec2Configuration`) ou no modelo de lançamento especificado em `launchTemplate`. Suponha que nenhuma ID de AMI seja especificada em nenhuma dessas configurações e a configuração `updateToLatestImageVersion` seja `true`. Em seguida, a AMI mais recente otimizada para o Amazon ECS e compatível com AWS Batch é usada para qualquer atualização da infraestrutura.

Se uma ID de AMI for especificada em pelo menos uma dessas configurações, a atualização dependerá de qual configuração forneceu a ID de AMI usada antes da atualização. Quando você cria um ambiente de computação, a prioridade para selecionar uma ID de AMI é primeiro o modelo de execução, depois a configuração `imageId` e, finalmente, a configuração `imageIdOverride`.

No entanto, se a ID da AMI usado for do modelo de execução, a atualização das configurações `imageId0verride` ou `imageId` não a atualizará. A única maneira de atualizar uma ID de AMI selecionada no modelo de execução é atualizando o modelo de execução. Se o parâmetro de versão do modelo de execução for `$Default` ou `$Latest`, a versão padrão ou mais recente do modelo de execução especificado será avaliada. Se uma ID de AMI diferente for selecionada por padrão ou se a versão mais recente do modelo de execução for selecionada, essa ID de AMI será usada na atualização.

Se o modelo de execução não foi usado para selecionar a ID da AMI, a ID da AMI especificada nos parâmetros `imageId` ou `imageId0verride` será usada. Se ambos forem especificados, a ID da AMI especificada no parâmetro `imageId0verride` será usada.

Suponha que o ambiente de computação use uma ID de AMI especificada pelos parâmetros `imageId`, `imageId0verride`, ou `launchTemplate` e você queira usar a AMI otimizada mais recente do Amazon ECS compatível com AWS Batch. Em seguida, a atualização deve remover as configurações que forneceram os IDs de AMI. Para `imageId`, isso requer a especificação de uma string vazia para esse parâmetro. Para `imageId0verride`, isso requer a especificação de uma string vazia para o parâmetro `ec2Configuration`.

Se a ID da AMI veio do modelo de lançamento, você pode mudar para a AMI otimizada mais recente do Amazon ECS que é suportada AWS Batch por uma das seguintes formas:

- Remova o modelo de execução especificando uma string vazia para o parâmetro `launchTemplateId` ou `launchTemplateName`. Isso remove todo o modelo de lançamento, em vez de apenas a ID da AMI.
- Se a versão atualizada do modelo de lançamento não especificar uma ID de AMI, o parâmetro `updateToLatestImageVersion` deverá ser definido como `true`.

Ambientes de computação Amazon EKS

Começando a usar AWS Batch no Amazon EKSfornece um pequeno guia para criar ambientes de computação EKS. Esta seção fornece mais detalhes sobre os ambientes computacionais do Amazon EKS.

Tópicos

- Atualizando a versão Kubernetes do ambiente de computação
- Responsabilidade compartilhada dos nós Kubernetes
- Executando DaemonSet em nós AWS Batch gerenciados
- Como personalizar nós gerenciados com modelos de execução

Seleção de AMI padrão

Ao criar um ambiente computacional Amazon EKS, você não precisa especificar uma Amazon Machine Image (AMI). AWS Batch seleciona uma AMI otimizada para Amazon EKS com base na Kubernetes versão e nos tipos de instância especificados na sua CreateComputeEnvironmentsolicitação. Em geral, recomendamos usar a seleção AMI do padrão. Para obter mais informações, consulte Versões do AMI Linux otimizadas para Amazon EKS no Guia do usuário do Amazon EKS.

Execute o comando a seguir para ver qual tipo de AMI AWS Batch está selecionado para seu ambiente computacional Amazon EKS. O exemplo a seguir é um tipo de instância sem GPU.

```
# compute CE example: indicates Batch has chosen the AL2 x86 or ARM EKS 1.29 AMI,
 depending on instance types
$ aws batch describe-compute-environments --compute-environments My-Eks-CE1 \
    | jq '.computeEnvironments[].computeResources.ec2Configuration'
[
  {
    "imageType": "EKS_AL2",
    "imageKubernetesVersion": "1.29"
  }
]
```

O exemplo a seguir é um tipo de instância de GPU.

```
# GPU CE example: indicates Batch has choosen the AL2 x86 EKS Accelerated 1.29 AMI
$ aws batch describe-compute-environments --compute-environments My-Eks-GPU-CE \
    | jq '.computeEnvironments[].computeResources.ec2Configuration'
[
  {
    "imageType": "EKS_AL2_NVIDIA",
    "imageKubernetesVersion": "1.29"
  }
]
```

Versões do Kubernetes com suporte

AWS Batch no Amazon EKS atualmente oferece suporte às seguintes Kubernetes versões:

- `1.29`
- `1.28`
- `1.27`
- `1.26`
- `1.25`
- `1.24`
- `1.23`

Talvez você veja uma mensagem de erro semelhante à seguinte ao usar a operação de `CreateComputeEnvironment` API ou a operação de `UpdateComputeEnvironment` API para criar ou atualizar um ambiente de computação. Esse problema ocorre se você especificar uma versão Kubernetes não suportada em `EC2Configuration`.

```
At least one imageKubernetesVersion in EC2Configuration is not supported.
```

Para resolver esse problema, exclua o ambiente de computação e recrie-o com uma versão Kubernetes compatível.

Você pode realizar uma pequena atualização de versão no seu cluster Amazon EKS. Por exemplo, você pode atualizar o cluster de `1.xx` para, `1.yy` mesmo que a versão secundária não seja compatível.

No entanto, o status do ambiente de computação pode mudar para `INVALID` depois de uma atualização de versão principal. Por exemplo, se você realizar uma atualização de versão principal de `1.xx` para`2.yy`. Se a versão principal não for compatível com AWS Batch, você verá uma mensagem de erro semelhante à seguinte.

```
reason=CLIENT_ERROR - ... EKS Cluster version [2.yy] is unsupported
```

Atualizando a versão Kubernetes do ambiente de computação

Com AWS Batch, você pode atualizar a Kubernetes versão de um ambiente computacional para oferecer suporte às atualizações de cluster do Amazon EKS. A Kubernetes versão de um ambiente

computacional é a versão do Amazon EKS AMI para os Kubernetes nós que são AWS Batch iniciados para executar trabalhos. Você pode realizar uma atualização de Kubernetes versão em seus nós do Amazon EKS antes ou depois de atualizar a versão do plano de controle do cluster Amazon EKS. Recomendamos que você atualize os nós após atualizar o plano de controle. Para obter mais informações, consulte Atualizando uma versão Kubernetes do cluster Amazon EKS no Guia do usuário do Amazon EKS.

Para atualizar a versão Kubernetes de um ambiente de computação, use a operação API UpdateComputeEnvironment.

```
$ aws batch update-compute-environment \
    --compute-environment <compute-environment-name> \
    --compute-resources \
      'ec2Configuration=[{imageType=EKS_AL2,imageKubernetesVersion=1.23}]'
```

Responsabilidade compartilhada dos nós Kubernetes

A manutenção dos ambientes de computação é uma responsabilidade compartilhada.

- Não altere nem remova AWS Batch nós, rótulos, manchas, namespaces, modelos de lançamento ou grupos de escalonamento automático. Não adicione manchas aos nós AWS Batch gerenciados. Se você fizer alguma dessas alterações, seu ambiente de computação não poderá ser suportado e ocorrerão falhas, incluindo instâncias ociosas.
- Não direcione seus pods para nós AWS Batch gerenciados. Se você direcionar seus pods para os nós gerenciados, ocorrerão falhas no escalonamento e filas de trabalhos paralisadas. Execute cargas de trabalho que não são usadas AWS Batch em nós autogerenciados ou grupos de nós gerenciados. Para obter mais informações, consulte Grupos de nós gerenciados no Guia do usuário do Amazon EKS.
- Você pode DaemonSet direcionar um para ser executado em nós AWS Batch gerenciados. Para ter mais informações, consulte Executando DaemonSet em nós AWS Batch gerenciados.

AWS Batch não atualiza automaticamente as AMIs do ambiente computacional. É sua responsabilidade atualizá-las. Execute o comando a seguir para atualizar suas AMIs para a versão AMI mais recente.

```
$ aws batch update-compute-environment \
    --compute-environment <compute-environment-name> \
```

```
    --compute-resources 'updateToLatestImageVersion=true'
```

AWS Batch não atualiza automaticamente a Kubernetes versão. Execute o comando a seguir para atualizar a versão Kubernetes do seu ambiente de computação para *1.23*.

```
$ aws batch update-compute-environment \
    --compute-environment <compute-environment-name> \
    --compute-resources \
      'ec2Configuration=[{imageType=EKS_AL2,imageKubernetesVersion=1.23}]'
```

Ao atualizar para uma AMI ou versão Kubernetes mais recente, você pode especificar se os trabalhos serão encerrados quando eles forem atualizados (`terminateJobsOnUpdate`) e quanto tempo esperar até que uma instância seja substituída se os trabalhos em execução não forem concluídos (`jobExecutionTimeoutMinutes`.) Para obter mais informações, consulte Criação de um ambiente de computação e a política de atualização de infraestrutura (UpdatePolicy) definida na operação da API UpdateComputeEnvironment.

Executando DaemonSet em nós AWS Batch gerenciados

AWS Batch define manchas nos Kubernetes nós AWS Batch gerenciados. Você pode DaemonSet direcionar um para ser executado em nós AWS Batch gerenciados com o seguinte`tolerations`.

```
tolerations:
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
```

Outra maneira de fazer isso é com o `tolerations` a seguir.

```
tolerations:
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
  effect: "NoSchedule"
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
  effect: "NoExecute"
```

Como personalizar nós gerenciados com modelos de execução

AWS Batch no Amazon EKS oferece suporte a modelos de lançamento. Há restrições sobre o que seu modelo de lançamento pode fazer.

> **Important**
>
> AWS Batch corre`/etc/eks/bootstrap.sh`. Não execute `/etc/eks/bootstrap.sh` em seu modelo de lançamento ou scripts cloud-init user-data. Você pode adicionar outros parâmetros além do parâmetro `--kubelet-extra-args` ao bootstrap.sh. Para fazer isso, defina a variável `AWS_BATCH_KUBELET_EXTRA_ARGS` no arquivo `/etc/aws-batch/batch.config`. Consulte o código a seguir para ver um exemplo.

> **Note**
>
> Se o modelo de lançamento for alterado após CreateComputeEnvironmentser chamado, UpdateComputeEnvironmentdeverá ser chamado para avaliar a versão do modelo de lançamento para substituição.

Tópicos

Adicionando argumentos **`kubelet`** extras

AWS Batch suporta a adição de argumentos extras ao `kubelet` comando. Para obter a lista completa de parâmetros compatíveis, consulte kubelet na documentação Kubernetes. No exemplo a seguir, *`--node-labels mylabel=helloworld`* é adicionado à linha de comando `kubelet`.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
mkdir -p /etc/aws-batch
```

```
echo AWS_BATCH_KUBELET_EXTRA_ARGS=\"--node-labels mylabel=helloworld\" >> /etc/aws-
batch/batch.config

--==MYBOUNDARY==--
```

Configurando o tempo de execução do contêiner

Você pode usar a variável de ambiente AWS Batch `CONTAINER_RUNTIME` para configurar o tempo de execução do contêiner em um nó gerenciado. O exemplo a seguir define o tempo de execução do contêiner para `containerd` quando `bootstrap.sh` for executado. Para obter mais informações, consulte `containerd` na documentação Kubernetes.

> **Note**
>
> A variável de ambiente `CONTAINER_RUNTIME` é equivalente à opção `--container-runtime` de `bootstrap.sh`. Para obter mais informações, consulte Options na documentação Kubernetes.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
mkdir -p /etc/aws-batch

echo CONTAINER_RUNTIME=containerd >> /etc/aws-batch/batch.config

--==MYBOUNDARY==--
```

Montando um volume do Amazon EFS

Você pode usar modelos de execução para montar volumes no nó. No exemplo a seguir, as configurações `cloud-config` `packages` e `runcmd` são usadas. Para obter mais informações, consulte Exemplos de configuração de Nuvem na documentação cloud-init.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="
```

```
--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
- amazon-efs-utils

runcmd:
- file_system_id_01=fs-abcdef123
- efs_directory=/mnt/efs

- mkdir -p ${efs_directory}
- echo "${file_system_id_01}:/ ${efs_directory} efs _netdev,noresvport,tls,iam 0 0"
 >> /etc/fstab
- mount -t efs -o tls ${file_system_id_01}:/ ${efs_directory}

--==MYBOUNDARY==--
```

Para usar esse volume na tarefa, ele deve ser adicionado ao parâmetro EksProperties a. RegisterJobDefinition O exemplo a seguir é uma grande parte da definição do trabalho.

```
{
    "jobDefinitionName": "MyJobOnEks_EFS",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["ls", "-la", "/efs"],
                    "resources": {
                        "limits": {
                            "cpu": "1",
                            "memory": "1024Mi"
                        }
                    },
                    "volumeMounts": [
                        {
                            "name": "efs-volume",
                            "mountPath": "/efs"
                        }
                    ]
                }
```

```
            ],
            "volumes": [
                {
                    "name": "efs-volume",
                    "hostPath": {
                        "path": "/mnt/efs"
                    }
                }
            ]
        }
    }
}
```

No nó, o volume do Amazon EFS é montado no diretório `/mnt/efs`. No contêiner da tarefa do Amazon EKS, o volume é montado no `/efs` diretório.

Suporte a IPv6

AWS Batch suporta clusters Amazon EKS que têm endereços IPv6. Nenhuma personalização é necessária para AWS Batch suporte. No entanto, antes de começar, recomendamos que você analise as considerações e condições descritas em Atribuição de endereços IPv6 para pods e serviços no Guia do usuário do Amazon EKS.

Recurso de Computação Gerenciamento de Memória

Quando o atendente de contêiner do Amazon ECS registra um recurso de computação da instância de contêiner em um ambiente de computação do cluster, o atendente deve determinar a quantidade de memória que o recurso de computação da instância de contêiner tem disponível para reservá-la para os seus trabalhos da tarefa. Devido à sobrecarga de memória da plataforma e a memória ocupada pelo kernel do sistema, esse número é diferente da quantidade de memória instalada para instâncias do Amazon EC2. Por exemplo, uma instância `m4.large` tem 8 GiB de memória instalada. No entanto, isso nem sempre traduz para 8192 MiB de memória disponível para trabalhos nos quais o recurso de computação é registrado.

Suponha que você especifique 8192 MiB para o trabalho e que nenhum dos seus recursos de computação tenha 8192 MiB de memória disponível ou maior para satisfazer a esse requisito. O trabalho não poderá ser colocado em seu ambiente de computação. Se você estiver usando um ambiente de computação gerenciado, o AWS Batch deverá executar um tipo de instância maior para acomodar a solicitação.

A AMI padrão de recursos de computação AWS Batch também reserva 32 MiB de memória para o atendente de contêiner do Amazon ECS e outros processos críticos do sistema. Essa memória não está disponível para alocação de trabalhos. Para obter mais informações, consulte Como Reservar Memória do Sistema.

O atendente de contêiner do Amazon ECS usa a função `ReadMemInfo()` do Docker para consultar a memória disponível total para o sistema operacional. O Linux oferece utilitários de linha de comando para determinar a memória total.

Example - Determinar a memória total do Linux

O comando free retorna a memória total reconhecida pelo sistema operacional.

```
$ free -b
```

O exemplo a seguir é um exemplo de saída para uma instância `m4.large` executando a AMI do Amazon Linux otimizada para Amazon ECS.

```
             total       used       free     shared    buffers     cached
Mem:    8373026816  348180480 8024846336      90112   25534464  205418496
-/+ buffers/cache:  117227520 8255799296
```

Essa instância tem 8373026816 bytes de memória total. Isso significa que há 7985 MiB disponíveis para tarefas.

Como Reservar Memória do Sistema

Se você ocupar toda a memória de um recurso de computação com seus trabalhos, é possível que os trabalhos disputem a memória com processos críticos do sistema e causem uma falha do sistema. O atendente de contêiner do Amazon ECS fornece uma variável de configuração chamada `ECS_RESERVED_MEMORY`. Você pode usar essa variável de configuração para remover um número específico de MiB de memória do grupo alocado aos seus trabalhos. Isso reserva de forma efetiva a memória para processos críticos do sistema.

A AMI AWS Batch padrão de recursos de computação reserva 32 MiB de memória para o atendente de contêiner do Amazon ECS e outros processos críticos do sistema.

Visualizando a Memória do da Instância de Contêiner

É possível visualizar a quantidade de memória que um recurso de computação registra no campo do console do Amazon ECS ou por meio da operação da API DescribeContainerInstances. Se você estiver tentando maximizar a sua utilização de recursos fornecendo aos trabalhos o máximo de memória possível para um tipo de instância específico, você poderá observar a memória disponível para esse recurso de computação e atribuir essa quantidade de memória aos seus trabalhos.

Para visualizar a memória dos recursos de computação

1. Abra o console em https://console.aws.amazon.com/ecs/v2.
2. Escolha os Clusters e, em seguida, escolha o cluster que hospeda os recursos de computação a serem visualizados.

   O nome do cluster do seu ambiente de computação começa com o seu nome do ambiente computacional.
3. Escolha Infraestrutura.
4. Em Instâncias de Contêiner, escolha a opção de instância de contêiner com a qual deseja estabelecer conexão.
5. A seção Recursos e Rede exibe a memória disponível registrada para o recurso de computação.

   O valor da memória Registrado é o mesmo registrado pelo recurso de computação com o Amazon ECS quando a instância foi executada primeiro, e o valor da memória Disponível é o valor ainda não atribuído aos trabalhos.

Considerações sobre memória e vCPU para AWS Batch no Amazon EKS

No AWS Batch do Amazon EKS, você pode especificar os recursos que forem disponibilizados para um contêiner. Por exemplo, você pode especificar valores `requests` ou `limits` para recursos de vCPU e memória.

A seguir, as restrições para especificar os recursos de vCPU:

- Pelo menos um valor de `requests` ou `limits` de vCPU deve ser especificado.
- Uma unidade de vCPU é equivalente a um núcleo físico ou virtual.
- O valor da vCPU deve ser inserido em números inteiros ou em acréscimos de 0,25.
- O menor valor válido de vCPU é 0,25.

- Se ambos forem especificados, o valor de `requests` deverá ser menor que ou igual ao valor de `limits`. Dessa forma, você pode configurar as configurações de vCPU flexíveis e rígidas.
- Os valores de vCPU não podem ser especificados no formato milliCPU. Por exemplo, `100m` não é um valor válido.
- AWS Batch utiliza o valor `requests` para decisões em escala. Se um valor `requests` não for especificado, o valor `limits` será copiado para o valor `requests`.

A seguir. as restrições para especificar os recursos de memória:

- Pelo menos um valor de `requests` ou `limits` de memória deve ser especificado.
- Os valores de memória devem estar em mebibytes (MiBs).
- Se ambos forem especificados, o valor de `requests` deverá ser igual ao valor de `limits`.
- AWS Batch utiliza o valor `requests` para decisões em escala. Se um valor de `requests` não for especificado, o valor de `limits` será copiado para o valor de `requests`.

A seguir. as restrições para especificar os recursos de GPU:

- Se ambos forem especificados, o valor de `requests` deverá ser igual ao valor de `limits`.
- AWS Batch utiliza o valor `requests` para decisões em escala. Se um valor `requests` não for especificado, o valor `limits` será copiado para o valor `requests`.

Exemplos de Definições de Trabalho

O seguinte AWS Batch sobre a configuração de definição de trabalho do Amazon EKS configura compartilhamentos flexíveis de vCPU. Isso permite que o AWS Batch no Amazon EKS use toda a capacidade da vCPU para o tipo de instância. No entanto, caso existam outros trabalhos em execução, ao trabalho será alocado um máximo de 2 vCPUs. A memória é limitada a 2 GB.

```
{
    "jobDefinitionName": "MyJobOnEks_Sleep",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
```

```
                    "command": ["sleep", "60"],
                    "resources": {
                        "requests": {
                            "cpu": "2",
                            "memory": "2048Mi"
                        }
                    }
                }
            ]
        }
    }
}
```

O seguinte AWS Batch na definição de trabalho do Amazon EKS tem um valor `request` de 1 e aloca um máximo de 4 vCPUs para o trabalho.

```
{
    "jobDefinitionName": "MyJobOnEks_Sleep",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["sleep", "60"],
                    "resources": {
                        "requests": {
                            "cpu": "1"
                        },
                        "limits": {
                            "cpu": "4",
                            "memory": "2048Mi"
                        }
                    }
                }
            ]
        }
    }
}
```

O seguinte AWS Batch na configuração de definição de trabalho do Amazon EKS define um valor `limits` de vCPU de 1 e um valor de `limits` de memória de 1 GB.

```
{
    "jobDefinitionName": "MyJobOnEks_Sleep",
    "type": "container",
    "eksProperties": {
        "podProperties": {
            "containers": [
                {
                    "image": "public.ecr.aws/amazonlinux/amazonlinux:2",
                    "command": ["sleep", "60"],
                    "resources": {
                        "limits": {
                            "cpu": "1",
                            "memory": "1024Mi"
                        }
                    }
                }
            ]
        }
    }
}
```

Quando o AWS Batch traduz um AWS Batch no trabalho do Amazon EKS em um pod do Amazon EKS, o AWS Batch copia o valor de `limits` para o valor de `requests`. Isso ocorre quando um valor de `requests` não é especificado. Quando você envia o exemplo de definição de trabalho de precedência, o pod `spec` é o seguinte.

```
apiVersion: v1
kind: Pod
...
spec:
  ...
  containers:
    - command:
        - sleep
        - 60
      image: public.ecr.aws/amazonlinux/amazonlinux:2
      resources:
        limits:
          cpu: 1
          memory: 1024Mi
        requests:
          cpu: 1
```

```
          memory: 1024Mi
    ...
```

Reserva de Memória e Reserva de Nó de CPU

AWS Batch depende da lógica padrão do arquivo `bootstrap.sh` para reservas de vCPU e reservas de memória. Para mais informações sobre o arquivo `bootstrap.sh`, consulte bootstrap.sh. Ao medir o tamanho dos seus recursos de vCPU e memória, considere os exemplos a seguir.

> **Note**
>
> Se nenhuma instância estiver em execução, as reservas de vCPU e reservas de memória podem, de início, afetar a lógica em escala do AWS Batch e a tomada de decisões. Depois que as instâncias estiverem em execução, o AWS Batch ajusta as alocações iniciais.

Exemplo de Reserva de Nó de CPU

O valor da reserva de CPU é calculado em milicores usando o número de vCPUs total disponível para a instância.

Número da vCPU	Porcentagem reservada
1	6%
2	1%
3-4	0,5%
4 e acima	0,25%

Usando os valores que precedem, o seguinte é verdadeiro:

- O valor da reserva de CPU para uma instância `c5.large` com 2 vCPUs é 70 m. Isso é calculado da seguinte maneira: (1*60) + (1*10) = 70 m.
- O valor para a reserva de CPU para uma instância `c5.24xlarge` com 96 vCPUs é 310 m. Isso é calculado da seguinte maneira: (1*60) + (1*10) + (2*5) + (92*2.5) = 310 m.

Neste exemplo, há 1930 (calculadas de 2000-70) unidades de vCPU de milicore disponíveis para executar trabalhos em uma instância `c5.large`. Suponha que seu trabalho requeira 2 unidades de vCPU (2*1000 m). O trabalho não caberá em uma instância única `c5.large`. No entanto, um trabalho que exige `1.75` unidades de vCPU é adequado.

Exemplo de reserva de memória do nó

O valor da reserva de memória é calculado em mebibytes usando o seguinte:

- a capacidade da instância em mebibytes. Por exemplo, uma instância de 8 GB tem 7.748 MiB.
- O valor de `kubeReserved`. O valor de `kubeReserved` é a quantidade de memória a ser reservada para os daemons do sistema. O valor de `kubeReserved` é calculado no campo da seguinte forma: ((11 * número de pods máximo suportado pelo tipo de instância) + 255). Para informações sobre o número máximo de pods compatível com cada tipo de instância, consulte eni-max-pods.txt
- O valor `HardEvictionLimit`. Quando a memória disponível fica abaixo do valor `HardEvictionLimit`, a instância tenta remover os pods.

A fórmula para calcular a memória disponível para alocação é a seguinte: (*`instance_capacity_in_MiB`*) - (11 * (*`maximum_number_of_pods`*)) - 255 - (*`HardEvictionLimitvalor`*)).

Uma instância `c5.large` suporta até 29 pods. Para uma instância `c5.large` de 8 GB com um valor `HardEvictionLimit` de 100 MiB, a memória disponível para alocação é 7074 MiB. Isso é calculado da seguinte maneira: (7748 - (11 * 29) -255 -100) = 7074 MiB. Neste exemplo, um trabalho MiB de 8.192 não cabe nessa instância, embora seja uma instância de 8 gibibyte (GiB).

DaemonSets

Ao usar DaemonSets, considere o seguinte:

- Se nenhum AWS Batch nas instâncias do Amazon EKS estiver em execução, DaemonSets pode afetar inicialmente a lógica em escala do AWS Batch e a tomada de decisões. Inicialmente, AWS Batch aloca 0,5 unidades de vCPU e 500 MiB para o DaemonSets esperado. Depois que as instâncias estiverem em execução, AWS Batch ajusta as alocações iniciais.
- Se um DaemonSet definir limites de vCPU ou limite de memória, os trabalhos AWS Batch no Amazon EKS terão menos recursos. Recomendamos que você mantenha o número de DaemonSets atribuídos a trabalhos AWS Batch o mais baixo possível.

Políticas de agendamento

Você pode usar políticas de agendamento para configurar como os recursos de computação em uma fila de tarefas serão alocados entre usuários ou workloads. Por meio das políticas de agendamento, você pode atribuir diferentes identificadores de compartilhamento justo a workloads ou usuários. AWS Batch atribui a cada identificador de compartilhamento justo uma percentagem do total de recursos disponíveis durante um período de tempo.

A percentagem justa é calculada usando os valores `shareDecaySeconds` e `shareDistribution`. Você pode adicionar tempo à análise de ações justas atribuindo um tempo de decaimento à política. Adicionar tempo dá mais peso ao tempo e menos ao peso definido. Você pode reservar recursos de computação para identificadores de compartilhamento justo que não estejam ativos especificando uma reserva de computação. Para mais informações, consulte Parâmetros de política de agendamento.

Tópicos

Criando uma política de agendamento

Antes de criar uma fila de tarefas com uma política de agendamento, você precisa criar uma política de agendamento. Ao criar uma política de agendamento, você associa um ou mais identificadores de compartilhamento justo, ou prefixos de identificadores de compartilhamento justo, com pesos para a fila e, opcionalmente, atribui um período de decaimento, além de calcular a reserva para a política.

Para criar uma política de agendamento

1. Abra o AWS Batch console em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Políticas de Agendamento, Criar.
4. Em Nome, insira um nome exclusivo para a sua política de agendamento. São permitidos até 128 caracteres (caixa alta e baixa), números, hifens e sublinhados.
5. (Opcional) Para Compartilhar Segundos de Decaimento, insira um valor inteiro para o tempo de decaimento de compartilhamento da política de agendamento. Um tempo maior de decaimento de compartilhamentos levará em consideração o uso de recursos de computação por um

período maior ao agendar trabalhos. Isso permitirá que tarefas utilizando um identificador de compartilhamento justo utilizem mais recursos de computação temporariamente do que o peso do identificador de compartilhamento justo permitiria, caso não estivesse usando recursos de computação recentes.

6. (Opcional) Em Reserva de Computação, insira um valor inteiro para reserva de computação da política de agendamento. A reserva de computação irá separar algumas vCPUs para uso por identificadores de compartilhamento justo não ativos no momento.

   A proporção reservada é `(`*`computeReservation`*`/100)^`*`ActiveFairShares`* onde ActiveFairShares é o número de identificadores de compartilhamento justo ativos.

   Por exemplo, um `computeReservation` valor de 50 indica que AWS Batch deve reservar 50% da VCPU máxima disponível caso haja apenas um identificador de ações justas, 25% em caso de dois identificadores e 12,5% em caso de três identificadores de ações justas. Um `computeReservation` valor de 25 indica que AWS Batch deve reservar 25% da vCPU máxima disponível caso haja apenas um identificador de compartilhamento justo, 6,25% em caso de dois identificadores e 1,56% caso haja três identificadores de compartilhamento justo.

7. Na seção Atributos de Compartilhamento, é possível especificar o identificador de compartilhamento justo e peso de cada identificador de compartilhamento justo associado à política de agendamento.

   a. Escolha Adicionar Identificador de Compartilhamento.

   b. Para Identificador de Compartilhamento, especifique o identificador de compartilhamento justo. Se a string terminar com “*”, isso torna-se um prefixo de identificador de compartilhamento justo para equiparar identificadores de compartilhamento justo para tarefas. Todos os identificadores de compartilhamento justo e respectivos prefixos de em uma política de agendamento precisam ser exclusivos e não podem se sobrepor. Por exemplo, você não pode ter identificadores de compartilhamento justo com prefixo “UsuárioA*” e identificador de compartilhamento justo “UsuárioA1” na mesma política de agendamento.

   c. Para o Fator de Peso, especifique o peso relativo para o identificador de compartilhamento justo. O valor padrão é 1.0. Um valor mais baixo terá prioridade mais alta para recursos de computação. Se um prefixo de identificador de compartilhamento justo for usado, tarefas com identificadores de compartilhamento justo que começarem com o prefixo compartilharão o fator peso. Isso aumenta efetivamente o fator peso dessas tarefas, enquanto diminui sua prioridade individual e mantém o mesmo fator para o prefixo do identificador de compartilhamento justo.

8. (Opcional) Na seção Tags, você pode especificar a chave e o valor de cada tag associada à política de agendamento. Para mais informações, consulte Marcando seus Recursos AWS Batch.
9. Escolha Enviar para finalizar e criar sua política de agendamento.

Modelo de política de agendamento

Um modelo de política de agendamento vazio é mostrado abaixo. É possível usar esse modelo para criar sua política de agendamento, que pode ser salva em um arquivo e usada com a opção `--cli-input-json` da AWS CLI. Para obter mais informações sobre esses parâmetros, consulte CreateSchedulingPolicy na AWS Batch API Reference.

```
{
    "name": "",
    "fairsharePolicy": {
        "shareDecaySeconds": 0,
        "computeReservation": 0,
        "shareDistribution": [
            {
                "shareIdentifier": "",
                "weightFactor": 0.0
            }
        ]
    },
    "tags": {
        "KeyName": ""
    }
}
```

> **Note**
>
> Você pode gerar o modelo de fila de trabalhos anteriores com o comando da AWS CLI a seguir.
>
> ```
> $ aws batch create-scheduling-policy --generate-cli-skeleton
> ```

Parâmetros de política de agendamento

As políticas de agendamento são divididas em três componentes básicos: o nome, a política de compartilhamento justo e as tags da política de agendamento.

Tópicos

Nome da política de agendamento

`name`

O nome da política de programação. São permitidos até 128 letras (caixa alta e baixa), números, hífens e sublinhados.

Tipo: Sequência

Obrigatório: Sim

Política de compartilhamento justo

`fairsharePolicy`

A política de compartilhamento justo da política de programação.

```
"fairsharePolicy": {
   "computeReservation": number,
   "shareDecaySeconds": number,
   "shareDistribution": [
      {
         "shareIdentifier": "string",
         "weightFactor": number
      }
   ]
}
```

Tipo: Objeto

Obrigatório: Não

`computeReservation`

Um valor usado para reservar parte da vCPU máxima disponível para identificadores de compartilhamento justo que ainda não foram usados.

A proporção reservada é `(`*`computeReservation`*`/100)^`*`ActiveFairShares`* onde ActiveFairShares é o número de identificadores de compartilhamento justo ativos.

Por exemplo, um valor `computeReservation` de 50 indica que AWS Batch deve reservar 50% da vCPU máxima disponível se houver apenas um identificador de ações justas, 25% se houver dois identificadores de ações justas e 12,5% se houver três identificadores de ações justas. Um valor `computeReservation` de 25 indica que AWS Batch deve reservar 25% da vCPU máxima disponível se houver apenas um identificador de compartilhamento justo, 6,25% se houver dois identificadores de compartilhamento justo e 1,56% se houver três identificadores de compartilhamento justo.

Tipo: inteiro

Faixa válida: valor mínimo de 0. Valor máximo de 99

Obrigatório: Não

`shareDecaySeconds`

O período de tempo a ser usado para calcular uma porcentagem de compartilhamento justo para cada identificador de compartilhamento justo em uso, em segundos. Um valor zero (0) indica que somente o uso atual deve ser medido. A degradação permite que os trabalhos executados mais recentemente tenham mais ponderação do que os trabalhos executados anteriormente.

Tipo: inteiro

Faixa válida: valor mínimo de 0. Valor máximo de 604800 (1 semana).

Obrigatório: não

`shareDistribution`

Matriz de objetos que contêm as ponderações para os identificadores de compartilhamento justo da política de compartilhamento justo. Os identificadores de compartilhamento justo que não estiverem incluídos têm uma ponderação padrão de `1.0`.

```
"shareDistribution": [
   {
      "shareIdentifier": "string",
      "weightFactor": number
   }
]
```

Tipo: matriz

Obrigatório: não

`shareIdentifier`

Um identificador de compartilhamento justo ou prefixo de identificador de compartilhamento justo. Se a string terminar com '*', essa string especificará um prefixo de identificador de compartilhamento justo para identificadores de compartilhamento justo que começam com esse prefixo. Por exemplo, se o valor for 1 `UserA*` e o `weightFactor` for 1 e houver dois identificadores de compartilhamento justo que começam com `UserA`, cada um desses identificadores de compartilhamento justo terá um peso de 2; se houver cinco identificadores de compartilhamento justo, cada um terá um peso de 5.

A lista de identificadores de compartilhamento justo em uma política de compartilhamento justo não pode se sobrepor. Por exemplo, você não pode ter um prefixo de identificador de compartilhamento justo de `UserA*` e um identificador de compartilhamento justo de `UserA-1` na mesma política de compartilhamento justo.

Tipo: Sequência

Obrigatório: sim

`weightFactor`

O fator de ponderação para o identificador de compartilhamento justo. O valor padrão é 1.0. Um valor mais baixo tem uma prioridade mais alta para os recursos de computação. Por exemplo, trabalhos que usam um identificador de compartilhamento com um fator de ponderação de 0,125 (1/8) obtêm 8 vezes os recursos de computação dos trabalhos que usam um identificador de compartilhamento com um fator de ponderação 1.

O menor valor com suporte é 0,0001 e o maior valor com suporte é 999,9999.

Tipo: float

Obrigatório: não

Tags

`tags`

Etiquetas de pares de valores-chave a serem associadas à política de agendamento. Para obter mais informações, consulte Marcando seus Recursos AWS Batch.

Tipo:: mapa de string para string

Obrigatório: não

Orquestre trabalhos AWS Batch com máquinas de estado do Step Functions no console do AWS Batch

Você pode usar o console do AWS Batch para visualizar detalhes sobre as máquinas de estado do Step Functions e as funções usadas por elas.

Seções

Visualizar detalhes da máquina de estado

O console do AWS Batch exibe uma lista de máquinas de estado na região da Região da AWS atual que contêm pelo menos uma etapa de fluxo de trabalho que envia um trabalho do AWS Batch.

Escolha uma máquina de estado para visualizar uma representação gráfica do fluxo de trabalho. As etapas destacadas em azul representam trabahos do AWS Batch. Use os controles do gráfico para ampliar e diminuir o zoom e para centralizar o gráfico.

> **Note**
>
> Quando um trabalho do AWS Batch é referenciada dinamicamente com JsonPath na definição da máquina de estado, os detalhes da função não podem ser exibidos no console do AWS Batch. Em vez disso, o nome do trabalho será listado como Dynamic reference (Referência dinâmica), e as etapas correspondentes no gráfico serão desabilitadas.

Como visualizar detalhes da máquina de estado

1. Abra a página de orquestração de fluxo de trabalho com tecnologia Step Functions do console do AWS Batch.
2. Escolha uma máquina de estado.
   <result>

   O console do AWS Batch abrirá a página Details.

```
</result>
```

Para obter mais informações, consulte Step Functions no Developer GuideAWS Step Functions.

Editar uma máquina de estado

Se desejar editar uma máquina de estado, o AWS Batch abre a página Edit definition (Editar definição) do console do Step Functions.

Como editar uma máquina de estado

1. Abra a página de orquestração de fluxo de trabalho com tecnologia Step Functions do console do AWS Batch.
2. Escolha uma máquina de estado.
3. Escolha Edit.

   O console do Step Functions abrirá a página Edit definition.
4. Edite a máquina de estado e escolha Save.

Para obter mais informações sobre a edição de máquinas de estado, consulte Step Functions state machine language no Developer GuideAWS Step Functions.

Executar uma máquina de estado

Se desejar executar uma máquina de estado, o AWS Batch abre a página New execution (Nova execução) do console do Step Functions.

Como executar uma máquina de estado

1. Abra a página de orquestração de fluxo de trabalho com tecnologia Step Functions do console do AWS Batch.
2. Escolha uma máquina de estado.
3. Escolha Executar.

   O console do Step Functions abrirá a página New execution.
4. (Opcional) Edite a máquina de estado e escolha Start execution.

Para obter mais informações sobre a execução de máquinas de estado, consulte os conceitos de execução da máquina de estado do Step Functions no Guia do desenvolvedor do AWS Step Functions.

AWS Batch no AWS Fargate

O AWS Fargate é uma tecnologia que pode ser usada com o AWS Batch para executar contêineres sem a necessidade de gerenciar servidores ou clusters de instâncias do Amazon EC2. Com o AWS Fargate, não é mais necessário provisionar, configurar nem escalar os clusters de máquinas virtuais para executar contêineres. Isso elimina a necessidade de escolher tipos de servidor, decidir quando dimensionar clusters ou otimizar o agrupamento de clusters.

Ao executar seus trabalhos com recursos do Fargate, você empacota sua aplicação em contêineres, especifica os requisitos de CPU e de memória, define as políticas do IAM e de rede e inicia a aplicação. Cada trabalho do Fargate tem seu próprio limite de isolamento e não compartilha o kernel subjacente, os recursos de CPU, os recursos de memória nem a interface de rede elástica com outro trabalho.

Sumário

Quando usar o Fargate

Recomendamos usar o Fargate na maioria dos cenários. O Fargate inicia e escala a computação de acordo com os requisitos de recursos que você especifica para o contêiner. Com o Fargate, você não precisa provisionar em excesso nem pagar por servidores adicionais. Você também não precisa se preocupar com as especificidades dos parâmetros relacionados à infraestrutura, como o tipo de instância. Quando o ambiente de computação precisa ter a escala aumentada verticalmente, os trabalhos executados nos recursos do Fargate podem ser iniciados mais rapidamente. Normalmente, são necessários alguns minutos para ativar uma nova instância do Amazon EC2. No entanto, as tarefas executadas no Fargate podem ser provisionadas em cerca de 30 segundos. O tempo exato necessário depende de vários fatores, incluindo o tamanho da imagem do contêiner e o número de trabalhos.

No entanto, recomendamos que você use o Amazon EC2 se seus trabalhos exigirem qualquer um dos seguintes:

- Mais de 16 vCPUs
- Mais de 120 gibibytes (GiB) de memória
- UMA GPU
- Imagem de máquina da Amazon (AMI) personalizada
- Qualquer um dos parâmetros do linuxParameters

Se você tem um grande número de trabalhos, recomendamos usar a infraestrutura do Amazon EC2. Por exemplo, se o número de trabalhos em execução simultânea exceder os limites de controle de utilização do Fargate. Isso ocorre porque, com o EC2, os trabalhos podem ser enviados em uma taxa maior para os recursos do EC2 do que para os recursos do Fargate. Além disso, mais trabalhos podem ser executados simultaneamente quando você usa o EC2. Para obter mais informações, consulte AWSCotas de serviço do Fargate no Guia do desenvolvedor do Amazon Elastic Container Service.

Definições de trabalho no Fargate

Os trabalhos do AWS Batch no Fargate não oferecem suporte a todos os parâmetros de definição de trabalho disponíveis. Alguns parâmetros são totalmente incompatíveis e outros se comportam de maneira diferente nos trabalhos do Fargate.

A lista a seguir descreve os parâmetros de definição de tarefa que não são válidos ou são restritos nos trabalhos do Fargate.

`platformCapabilities`

Deve ser especificado como `FARGATE`.

```
"platformCapabilities": [ "FARGATE" ]
```

`type`

Deve ser especificado como `container`.

```
"type": "container"
```

Parâmetros em `containerProperties`

`executionRoleArn`

Deve ser especificado para trabalhos em execução nos recursos do Fargate. Para mais informações, consulte Funções do IAM para Tarefas no Guia de Desenvolvedor Amazon Elastic Container Service.

```
"executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole"
```

`fargatePlatformConfiguration`

(Opcional, somente para definições de trabalho do Fargate). Especifica a versão da plataforma do Fargate a `LATEST` ou de uma versão da plataforma recente. Valores possíveis para `platformVersion` são `1.3.0`, `1.4.0` e `LATEST` (padrão).

```
"fargatePlatformConfiguration": { "platformVersion": "1.4.0" }
```

`instanceType`, `ulimits`

Não é aplicável a trabalhos executados nos recursos do Fargate.

`memory`, `vcpus`

Essas configurações devem ser especificadas em `resourceRequirements`

`privileged`

Não especifique esse parâmetro ou especifique `false`.

```
"privileged": false
```

`resourceRequirements`

Os requisitos de memória e vCPU devem ser especificados usando valores compatíveis. Os recursos de GPU não são compatíveis com trabalhos executados nos recursos do Fargate.

Se você usa o GuardDuty Runtime Monitoring, há uma pequena sobrecarga de memória para o agente GuardDuty de segurança. Portanto, o limite de memória deve incluir o tamanho do agente GuardDuty de segurança. Para obter informações sobre os limites de memória do GuardDuty Security Agent, consulte Limites de CPU e memória no Guia GuardDuty do usuário. Para obter informações sobre as melhores práticas, consulte Como faço para corrigir erros de falta de

memória em minhas tarefas do Fargate depois de ativar o Runtime Monitoring no Amazon ECS Developer Guide.

```
"resourceRequirements": [
  {"type": "MEMORY", "value": "512"},
  {"type": "VCPU",   "value": "0.25"}
]
```

Parâmetros em `linuxParameters`

`devices`, `maxSwap`, `sharedMemorySize`, `swappiness`, `tmpfs`

Não é aplicável a trabalhos que sejam executados nos recursos do Fargate.

Parâmetros em `logConfiguration`

`logDriver`

Somente `awslogs` e `splunk` são compatíveis. Para ter mais informações, consulte Usar o driver de log awslogs.

Membros em `networkConfiguration`

`assignPublicIp`

Se a sub-rede privada não tiver um gateway NAT conectado para enviar tráfego para a Internet, `assignPublicIp` deverá ser "`ENABLED`". Para ter mais informações, consulte AWS Batch função de execução do IAM.

Filas de trabalhos no Fargate

As filas de trabalhos do AWS Batch no Fargate permanecem essencialmente inalteradas. A única restrição é que os ambientes de computação listados em `computeEnvironmentOrder` devem ser todos ambientes de computação Fargate (`FARGATE` ou `FARGATE_SPOT`). Os ambientes de computação EC2 e Fargate não podem ser misturados.

Ambientes de computação no Fargate

Ambientes de computação do AWS Batch no Fargate não são compatíveis com todos os parâmetros do ambiente de computação disponíveis. Alguns parâmetros são totalmente incompatíveis. Outros têm requisitos específicos para o Fargate.

A lista a seguir descreve os parâmetros do ambiente de computação que não são válidos ou são restritos nos trabalhos do Fargate.

`type`

Esse parâmetro precisa ser `MANAGED`.

```
"type": "MANAGED"
```

Parâmetros no objeto `computeResources`

`allocationStrategy`, `bidPercentage`, `desiredvCpus`, `imageId`, `instanceTypes`, `ec2Configuration`, `ec2KeyPair`, `instanceRole`, `launchTemplate`, `minvCpus`, `placementGroup`, `spotIamFleetRole`

Eles não são aplicáveis aos ambientes computacionais Fargate e não podem ser fornecidos.

`subnets`

Se as sub-redes listadas nesse parâmetro não tiverem gateways NAT conectados, o parâmetro `assignPublicIp` na definição do trabalho deverá ser definido como `ENABLED`.

`tags`

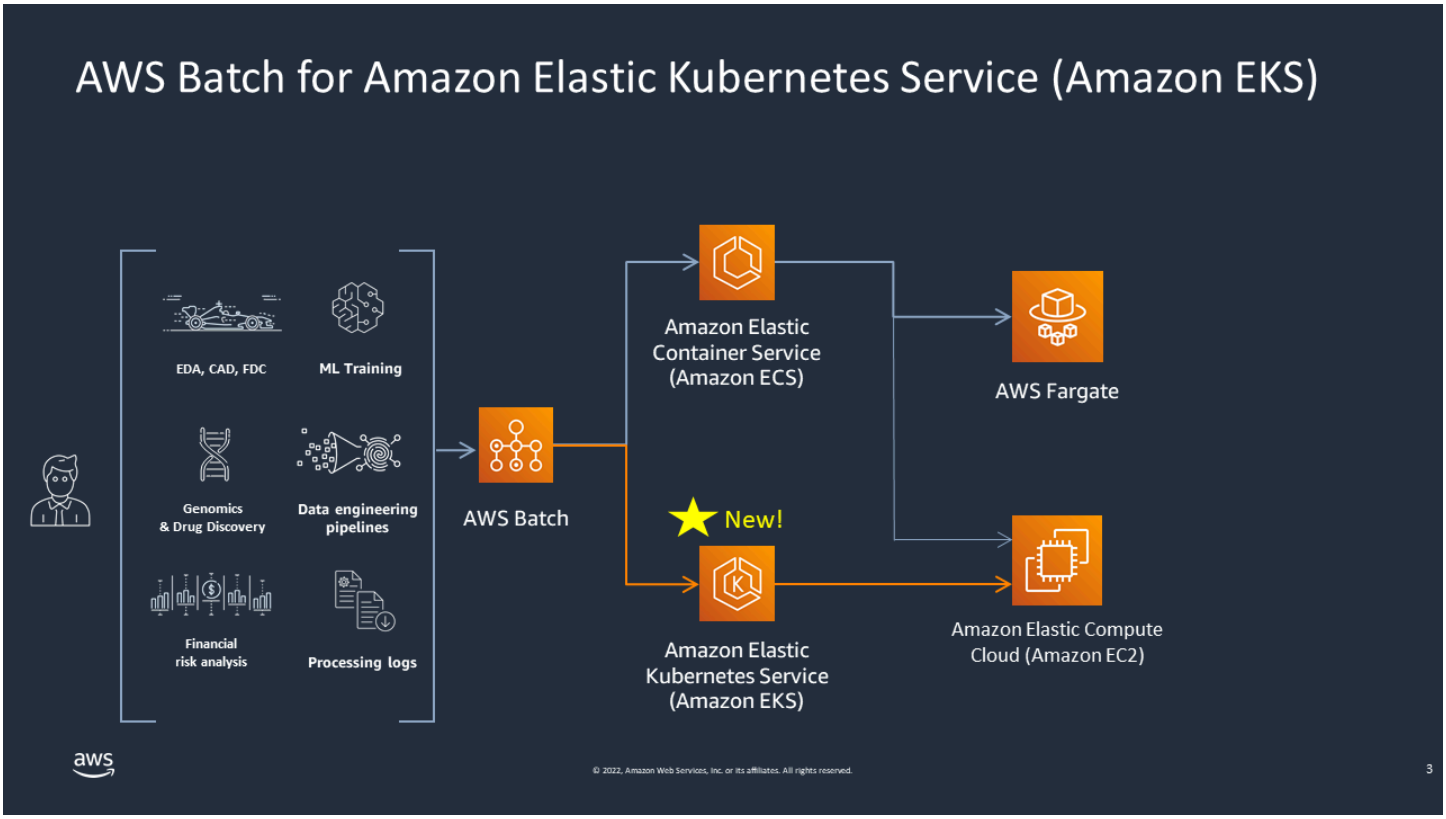
Isso não é aplicável aos ambientes computacionais Fargate e não pode ser fornecido. Para especificar tags para ambientes de computação Fargate, use o parâmetro `tags` que não está no objeto `computeResources`.

`type`

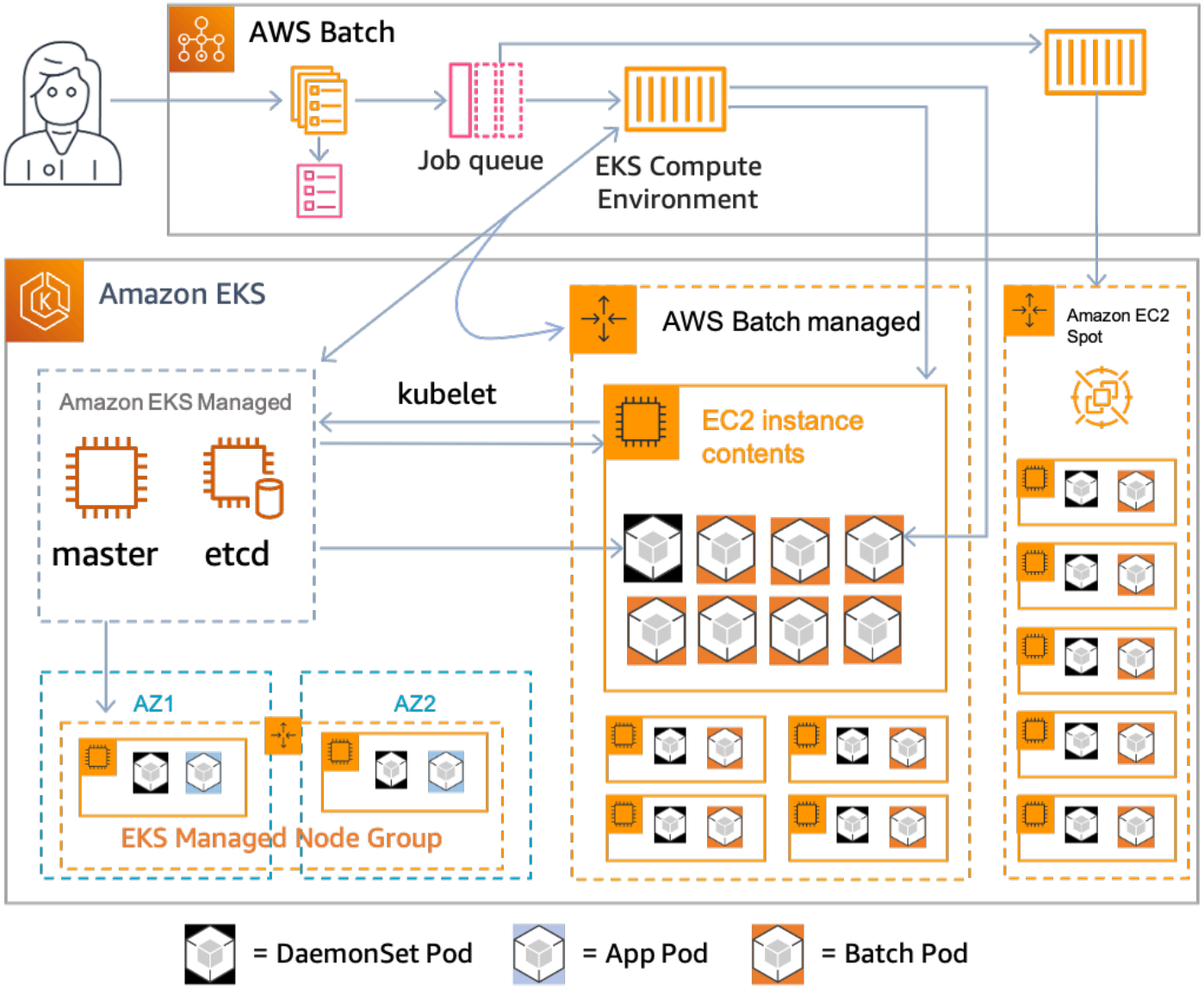
Isso deve ser `FARGATE` ou `FARGATE_SPOT`.

```
"type": "FARGATE_SPOT"
```

AWS Batch no Amazon EKS



AWS Batch simplifica suas cargas de trabalho em lotes nos clusters do Amazon EKS fornecendo recursos gerenciados em lotes. Isso inclui filas, rastreamento de dependências, tentativas e prioridades gerenciadas de tarefas, gerenciamento de pods e escalabilidade de nós. AWS Batch pode lidar com várias zonas de disponibilidade e vários tipos e tamanhos de instâncias do Amazon EC2. AWS Batch integra várias das melhores práticas do Amazon EC2 Spot para executar suas cargas de trabalho de forma tolerante a falhas, permitindo menos interrupções. Você pode usar o AWS Batch para executar um punhado de trabalhos noturnos ou milhões de trabalhos essenciais à missão com confiança.



AWS Batch é um serviço gerenciado que orquestra cargas de trabalho em lotes em seus Kubernetes clusters que são gerenciados pelo Amazon Elastic Kubernetes Service (Amazon EKS). AWS Batch conduz essa orquestração externamente aos seus clusters usando um modelo de “sobreposição”. Como AWS Batch é um serviço gerenciado, não há Kubernetes componentes (por exemplo, operadores ou recursos personalizados) para instalar ou gerenciar em seu cluster. AWS Batch só precisa que seu cluster seja configurado com controles de acesso baseados em funções (RBAC) que permitem AWS Batch a comunicação com o servidor da API. Kubernetes AWS Batch chama Kubernetes APIs para criar, monitorar e excluir Kubernetes pods e nós.

AWS Batch tem lógica de escalabilidade integrada para escalar Kubernetes nós com base na carga da fila de trabalhos com otimizações em termos de alocações de capacidade de trabalho. Quando a fila de trabalhos está vazia AWS Batch , reduz os nós até a capacidade mínima que você define,

que por padrão é zero. AWS Batch gerencia todo o ciclo de vida desses nós e decora os nós com rótulos e manchas. Dessa forma, outras Kubernetes cargas de trabalho não são colocadas nos nós gerenciados pelo AWS Batch. A exceção a isso é`DaemonSets`, que pode direcionar AWS Batch os nós para fornecer monitoramento e outras funcionalidades necessárias para a execução adequada dos trabalhos. Além disso, AWS Batch não executa trabalhos, especificamente pods, em nós do seu cluster que ele não gerencia. Assim, você pode usar lógica e serviços de escalabilidade separadamente para outros aplicativos no cluster.

Para enviar trabalhos para AWS Batch, você interage diretamente com a AWS Batch API. AWS Batch traduz trabalhos em `podspecs` e, em seguida, cria as solicitações para colocar pods em nós gerenciados pelo seu AWS Batch cluster Amazon EKS. Você pode usar ferramentas como `kubectl` para visualizar pods e nós em execução. Quando um pod conclui sua execução, AWS Batch exclui o pod criado para manter uma carga menor no Kubernetes sistema.

Você pode começar conectando um cluster válido do Amazon EKS com AWS Batch o. Em seguida, anexe uma fila de AWS Batch trabalhos a ela e registre uma definição de trabalho do Amazon EKS usando atributos `podspec` equivalentes. Por fim, envie trabalhos usando a operação SubmitJobda API que faz referência à definição do trabalho. Para ter mais informações, consulte Começando a usar AWS Batch no Amazon EKS.

Elastic Fabric Adapter

Um Elastic Fabric Adapter (EFA) é um dispositivo de rede para acelerar aplicativos de Computação de Alto Desempenho (HPC). O AWS Batch oferece suporte a aplicativos que usam o EFA se as condições a seguir são atendidas.

- Para obter uma lista de tipos de instância que oferecem suporte a EFAs, consulte Tipos de instância compatíveis no Guia do usuário do Amazon EC2 para instâncias do Linux.

  > **Tip**
  >
  > Para ver uma lista de tipos de instância que oferecem suporte a EFAs em um Região da AWS, execute o comando a seguir. Em seguida, faça referência cruzada à lista que é retornada com a lista de tipos de instância disponíveis no console AWS Batch.
  >
  > ```
  > $ aws ec2 describe-instance-types  --region us-east-1  --filters Name=network-info.efa-supported,Values=true  --query "InstanceTypes[*].[InstanceType]"  --output text | sort
  > ```

- Para obter uma lista de sistemas operacionais compatíveis, consulte Supported operating systems.
- A AMI tem o driver EFA carregado.
- O grupo de segurança do EFA deve permitir todo o tráfego de entrada e saída de e para o próprio grupo de segurança.
- Todas as instâncias que usam um EFA devem estar no mesmo grupo com posicionamento em cluster
- A definição do trabalho deve incluir um membro `devices` com `hostPath` definido como `/dev/infiniband/uverbs0` para permitir que o dispositivo EFA seja transmitido ao contêiner. Se `containerPath` for especificado, ele também deverá ser definido como `/dev/infiniband/uverbs0`. Se `permissions` estiver definido, ele deverá ser definido como `READ` | `WRITE` | `MKNOD`.

  A localização do membro LinuxParameters será diferente para trabalhos paralelos de vários nós e trabalhos de contêiner de nó único. Os exemplos a seguir demonstram as diferenças, mas estão faltando valores necessários.

  Example Exemplo de trabalho paralelo de vários nós

  ```
  {
  ```

```
  "jobDefinitionName": "EFA-MNP-JobDef",
  "type": "multinode",
  "nodeProperties": {
    ...
    "nodeRangeProperties": [
      {
        ...
        "container": {
          ...
          "linuxParameters": {
            "devices": [
              {
                "hostPath": "/dev/infiniband/uverbs0",
                "containerPath": "/dev/infiniband/uverbs0",
                "permissions": [
                    "READ", "WRITE", "MKNOD"
                ]
              },
            ],
          },
        },
      },
    ],
  },
}
```

Example Exemplo de trabalho de contêiner de nó único.

```
{
  "jobDefinitionName": "EFA-Container-JobDef",
  "type": "container",
  ...
  "containerProperties": {
    ...
    "linuxParameters": {
      "devices": [
        {
          "hostPath": "/dev/infiniband/uverbs0",
        },
      ],
    },
  },
}
```

Para obter mais informações, consulte Elastic Fabric Adapter no Amazon EC2 User Guide for Linux Instances.

AWS Batch Políticas do IAM, funções e permissões

Por padrão, usuários não têm permissão para criar ou modificar AWS Batch recursos ou executar tarefas usando a AWS Batch API, AWS Batch console, ou AWS CLI. Para permitir que usuários realizem essas ações, crie políticas do IAM que concedam permissão aos usuários para recursos e operações API específicas. Em seguida, anexe as políticas aos usuários ou grupos que exijam tais permissões.

Quando você anexa uma política a um usuário ou grupo de usuários, a política concede ou nega as permissões para tarefas específicas em recursos específicos. Para mais informações, consulte Permissões e Políticas em Guia de usuário do IAM. Para mais informações sobre como gerenciar e criar políticas personalizadas do IAM, consulte Gerenciamento de Políticas do IAM.

AWS Batch faz chamadas para outros Serviços da AWS em seu nome. Como resultado, AWS Batch deve autenticar usando suas credenciais. Mais especificamente, AWS Batch autentica criando um perfil do IAM e uma política que forneça essas permissões. Em seguida, associa o perfil aos seus ambientes de computação quando criados. Para mais informações, consulte Perfil de instância do Amazon ECS, Perfis do IAM, Usando Perfis Vinculados a Serviços e Criando um Perfil para Delegar Permissões a um AWS Serviço no Guia de Usuário do IAM.

Conceitos Básicos

Uma política do IAM deve conceder ou negar permissões para usar uma ou mais AWS Batch ações.

Tópicos

Estrutura da política

Os tópicos a seguir explicam a estrutura de uma política do IAM.

Tópicos

Sintaxe da política

A política do IAM é um documento JSON que consiste em uma ou mais declarações. Cada instrução é estruturada da maneira a seguir.

```
{
  "Statement":[{
    "Effect":"effect",
    "Action":"action",
    "Resource":"arn",
    "Condition":{
      "condition":{
    "key":"value"
    }
      }
    }
  ]
}
```

Existem vários elementos que compõem uma instrução:

- Effect: o efeito pode ser `Allow` ou `Deny`. Por padrão, os usuários não têm permissão para usar recursos e ações de API. Então, todas as solicitações são negadas. Uma permissão explícita substitui o padrão. Uma negação explícita substitui todas as permissões.
- Ação: a ação é a ação de API específica para a qual você está concedendo ou negando permissão. Para obter instruções sobre como especificar a ação, consulte Ações do AWS Batch.
- Resource: o recurso afetado pela ação. Com algumas ações de API do AWS Batch é possível incluir recursos específicos na política que podem ser criados ou modificados pela ação. Para

especificar um recurso na declaração, use o respectivo nome de recurso da Amazon (ARN). Para obter mais informações, consulte Permissões no nível do recurso com suporte para ações de API do AWS Batch e Nomes de recursos da Amazon para o AWS Batch. Se a operação da API do AWS Batch atualmente não é compatível com permissões em nível de recurso, inclua um curinga (*) para especificar que todos os recursos podem ser afetados pela ação.

- Condition: condições são opcionais. Elas podem ser usadas para controlar quando a política está em vigor.

Para obter mais informações sobre declarações de exemplo de política do IAM para o AWS Batch, consulte Criando AWS Batch políticas do IAM .

Ações do AWS Batch

Em uma declaração de política do IAM, é possível especificar qualquer ação de API de qualquer serviço que dê suporte ao IAM. Para o AWS Batch, use o seguinte prefixo com o nome da ação da API: `batch:` (por exemplo, `batch:SubmitJob` e `batch:CreateComputeEnvironment`).

Para especificar várias ações em uma única declaração, separe cada ação com uma vírgula.

```
"Action": ["batch:action1", "batch:action2"]
```

Você também pode especificar várias ações incluindo um curinga (*). Por exemplo, é possível especificar todas as ações com um nome começando com a palavra "Describe" (Descreva).

```
"Action": "batch:Describe*"
```

Para especificar todas as ações da API do AWS Batch, inclua um caractere curinga (*).

```
"Action": "batch:*"
```

Para obter uma lista de ações do AWS Batch, consulte Ações na Referência da API do AWS Batch.

Nomes de recursos da Amazon para o AWS Batch

Cada declaração de política do IAM se aplica aos recursos que você especifica usando os nomes dos recursos da Amazon (ARNs).

Um Nome de recurso da Amazon (ARN) tem a seguinte sintaxe geral:

```
arn:aws:[service]:[region]:[account]:resourceType/resourcePath
```

serviço

O serviço (por exemplo, `batch`).

região

A Região da AWS do recurso (por exemplo, `us-east-2`).

conta

O ID da Conta da AWS, sem hífens (por exemplo, `123456789012`).

resourceType

O tipo de recurso (por exemplo, `compute-environment`).

resourcePath

Um caminho que identifica o recurso. É possível usar um curinga (*) nos caminhos.

As operações da API do AWS Batch agora são compatíveis com permissões no nível do recurso em várias operações da API. Para obter mais informações, consulte Permissões no nível do recurso com suporte para ações de API do AWS Batch. Para especificar todos os recursos ou se uma ação de API específica não for compatível com ARNs, inclua um curinga (*) no elemento `Resource`.

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

Verificar se os usuários têm as permissões necessárias

Antes de colocar uma política do IAM em vigor, verifique se ela concede aos usuários as permissões para usar as ações e recursos de API específicos de que eles precisam.

Para fazer isso, primeiro, crie um usuário do para fins de teste e anexe a política do IAM ao usuário de teste. Em seguia, faça uma solicitação como o usuário de teste. Você pode fazer solicitações de teste no console ou com a AWS CLI.

> **Note**
>
> Você também pode testar as políticas usando o Simulador de políticas do IAM. Para obter mais informações sobre o simulador de políticas, consulte Como trabalhar com o simulador de políticas do IAM no Guia do usuário do IAM.

Caso a política não conceda ao usuário as permissões que você esperava ou caso ela seja muito permissiva, você pode ajustar a política conforme necessário. Teste novamente até obter os resultados desejados.

> **Important**
>
> Pode levar alguns minutos para que as alterações de política sejam propagadas até entrarem em vigor. Por isso, recomendamos que você aguarde pelo menos cinco minutos antes de testar as atualizações da política.

Caso uma verificação de autorização falhe, a solicitação retorna uma mensagem codificada com informações de diagnóstico. É possível decodificar a mensagem usando a ação `DecodeAuthorizationMessage`. Para obter mais informações, consulte DecodeAuthorizationMessage na Referência de API do AWS Security Token Service e decode-authorization-message na Referência de comandos da AWS CLI.

Permissões no nível do recurso com suporte para ações de API do AWS Batch

O termo Permissões no nível do recurso se refere à capacidade de especificar quais recursos nos quais os usuários têm permissões para executar ações. O AWS Batch oferece suporte parcial a permissões no nível do recurso. Para algumas ações AWS Batch, você pode controlar quando os usuários têm permissão para usar essas ações com base em condições que precisam ser cumpridas. Você também pode controlar com base nos recursos específicos que os usuários podem usar. Por exemplo, você pode conceder permissões aos usuários para enviar tarefas, mas somente para uma determinada fila de tarefas e apenas com uma determinada definição de tarefa.

A lista a seguir descreve quais ações da API AWS Batch são compatíveis com permissões no nível do recurso: A lista também descreve os recursos compatíveis, os ARNs de recurso e as chaves de condição para cada ação.

> **Important**
>
> Caso uma ação de API do AWS Batch não esteja listada nessa tabela, isso significa que ela não dá suporte a permissões no nível do recurso. Se uma ação da API AWS Batch não for compatível com as permissões em nível de recurso, você poderá conceder aos usuários permissão para usar a ação. No entanto, você deve incluir um caractere curinga (*) para o elemento de recurso da sua declaração de política.

Ações

CancelJob, CreateComputeEnvironment, CreateJobQueue, CreateSchedulingPolicy, DeleteComputeEnvironment, DeleteJobQueue, DeleteSchedulingPolicy, DeregisterJobDefinition, ListTagsForResource, RegisterJobDefinition, SubmitJob, TagResource, TerminateJob, UntagResource, UpdateComputeEnvironment, UpdateSchedulingPolicy, UpdateJobQueue

CancelJob

Cancela um trabalho em uma fila AWS Batch.

Recurso

Trabalho

arn:aws:batch:*region*:*account*:job/*jobId*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

CreateComputeEnvironment

Cria um ambiente de computação do AWS Batch.

Recurso

Ambiente de computação

arn:aws:batch:*region*:*account*:compute-environment/*compute-environment-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Condition keys

`aws:RequestTag/${TagKey}` (String)

Filtra ações com base nas tags transmitidas na solicitação.

`aws:TagKeys` (String)

Filtra ações com base nas chaves de tag transmitidas na solicitação.

CreateJobQueue

Cria uma fila AWS Batch de trabalho.

Recurso

Ambiente de computação

arn:aws:batch:*region*:*account*:compute-environment/*compute-environment-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Job Queue

arn:aws:batch:*region*:*account*:job-queue/*queue-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Política de agendamento

arn:aws:batch:*region*:*account*:scheduling-policy/*scheduling-policy-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Condition keys

`aws:RequestTag/${TagKey}` (String)

Filtra ações com base nas tags transmitidas na solicitação.

`aws:TagKeys` (String)

Filtra ações com base nas chaves de tag transmitidas na solicitação.

DeleteComputeEnvironment

Apaga um ambiente de computação do AWS Batch.

Recurso

Ambiente de computação

arn:aws:batch:*region*:*account*:compute-environment/*compute-environment-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

CreateSchedulingPolicy

Cria uma política AWS Batch de agendamento.

Recurso

Política de agendamento

arn:aws:batch:*region*:*account*:scheduling-policy/*scheduling-policy-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Condition keys

`aws:RequestTag/${TagKey}` (String)

Filtra ações com base nas tags transmitidas na solicitação.

`aws:TagKeys` (String)

Filtra ações com base nas chaves de tag transmitidas na solicitação.

DeleteJobQueue

Exclui a fila de trabalhos especificada. A exclusão da fila de trabalhos eventualmente exclui todos os trabalhos na fila. Os trabalhos são excluídos a uma taxa de cerca de 16 trabalhos por segundo.

Recurso

Job Queue

arn:aws:batch:*region*:*account*:job-queue/*queue-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

DeleteSchedulingPolicy

Exclui a Política de agendamento especificada.

Recurso

Política de agendamento

arn:aws:batch:*region*:*account*:scheduling-policy/*scheduling-policy-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

DeregisterJobDefinition

Cancela o registro de uma definição de trabalho AWS Batch.

Recurso

Job Definition

arn:aws:batch:*region*:*account*:job-definition/*definition-name*:*revision*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

ListTagsForResource

Lista tags para o recurso especificado.

Recurso

Ambiente de computação

arn:aws:batch:*region*:*account*:compute-environment/*compute-environment-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Trabalho

arn:aws:batch:*region*:*account*:job/*jobId*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Job Definition

arn:aws:batch:*region*:*account*:job-definition/*definition-name*:*revision*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Job Queue

arn:aws:batch:*region*:*account*:job-queue/*queue-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Política de agendamento

arn:aws:batch:*region*:*account*:scheduling-policy/*scheduling-policy-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

RegisterJobDefinition

Registra uma definição AWS Batch.

Recurso

Job Definition

arn:aws:batch:*region*:*account*:job-definition/*job-definition-name*

Condition keys

`batch:AWSLogsCreateGroup` (Booliano)

Quando esse parâmetro for verdadeiro, o `awslogs-group` será criado para os logs.

`batch:AWSLogsGroup` (String)

O grupo `awslogs` em que os logs estão localizados.

`batch:AWSLogsRegion` (String)

A região para a qual os logs são enviados.

`batch:AWSLogsStreamPrefix` (String)

O prefixo do fluxo de logs do `awslogs`.

`batch:Image` (String)

A imagem do Docker usada para iniciar um contêiner.

`batch:LogDriver` (String)

O driver de log usado para o trabalho.

`batch:Privileged` (Booliano)

Quando esse parâmetro for verdadeiro, o contêiner para o trabalho recebe permissões elevadas na instância de contêiner host.

`batch:User` (String)

O nome de usuário ou uid numérico a ser usado dentro do contêiner.

`aws:RequestTag/${TagKey}` (String)

Filtra ações com base nas tags transmitidas na solicitação.

`aws:TagKeys` (String)

Filtra ações com base nas chaves de tag transmitidas na solicitação.

SubmitJob

Envia um trabalho AWS Batch a partir de uma definição de trabalho.

Recurso

Trabalho

arn:aws:batch:*region*:*account*:job/*jobId*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Job Definition

arn:aws:batch:*region*:*account*:job-definition/*definition-name*[:*revision*]

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

> **Note**
>
> Essa chave só pode ser usada quando a definição do trabalho Nome de recurso da Amazon (ARN) está no formato `arn:aws:batch:`*region*`:`*account_number*`:job-definition/`*definition-name*`:`*revision*`.`

Job Queue

aarn:aws:batch:*region*:*account*:job-queue/*queue-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

TagResource

Marca o recurso especificado.

Recurso

Ambiente de computação

arn:aws:batch:*region*:*account*:compute-environment/*compute-environment-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Trabalho

arn:aws:batch:*region*:*account*:job/*jobId*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Job Definition

arn:aws:batch:*region*:*account*:job-definition/*definition-name*:*revision*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Job Queue

arn:aws:batch:*region*:*account*:job-queue/*queue-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Política de agendamento

arn:aws:batch:*region*:*account*:scheduling-policy/*scheduling-policy-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Condition keys

`aws:RequestTag/${TagKey}` (String)

Filtra ações com base nas tags transmitidas na solicitação.

`aws:TagKeys` (String)

Filtra ações com base nas chaves de tag transmitidas na solicitação.

TerminateJob

Encerra um trabalho em uma fila de trabalhos do AWS Batch Batch.

Recurso

Trabalho

arn:aws:batch:*region*:*account*:job/*jobId*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

UntagResource

Desmarca o recurso especificado.

Recurso

Ambiente de computação

arn:aws:batch:*region*:*account*:compute-environment/*compute-environment-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Trabalho

arn:aws:batch:*region*:*account*:job/*jobId*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Definição da tarefa

arn:aws:batch:*region*:*account*:job-definition/*definition-name*:*revision*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Job Queue

arn:aws:batch:*region*:*account*:job-queue/*queue-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Política de agendamento

arn:aws:batch:*region*:*account*:scheduling-policy/*scheduling-policy-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Condition keys

`aws:TagKeys` (String)

Filtra ações com base nas chaves de tag transmitidas na solicitação.

UpdateComputeEnvironment

Atualiza um ambiente de computação do AWS Batch.

Recurso

Compute Environment

arn:aws:batch:*region*:*account*:compute-environment/*compute-environment-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

UpdateJobQueue

Atualiza uma fila de trabalhos.

Recurso

Job Queue

arn:aws:batch:*region*:*account*:job-queue/*queue-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Política de agendamento

arn:aws:batch:*region*:*account*:scheduling-policy/*scheduling-policy-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

UpdateSchedulingPolicy

Atualiza uma política de agendamento.

Recurso

Política de agendamento

arn:aws:batch:*region*:*account*:scheduling-policy/*scheduling-policy-name*

Condition keys

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

Chaves de condição para ações API do AWS Batch

O AWS Batch define as chaves de condição a seguir que podem ser usadas no elemento `Condition` de uma política do IAM. É possível usar essas chaves para refinar ainda mais as condições sob as quais a declaração de política se aplica. Para visualizar as chaves de condição globais disponíveis para todos os serviços, consulte Chaves de contexto de condição globais no Guia do usuário do IAM.

`batch:AWSLogsCreateGroup` (Booliano)

Quando esse parâmetro for verdadeiro, o `awslogs-group` será criado para os logs.

`batch:AWSLogsGroup` (String)

O grupo `awslogs` em que os logs estão localizados.

`batch:AWSLogsRegion` (String)

A Região da AWS para a qual os logs são enviados.

`batch:AWSLogsStreamPrefix` (String)

O prefixo do fluxo de logs do `awslogs`.

`batch:Image` (String)

A imagem do Docker usada para iniciar um contêiner.

`batch:LogDriver` (String)

O driver de log usado para o trabalho.

`batch:Privileged` (Booliano)

Quando esse parâmetro é verdadeiro, o contêiner recebe permissões elevadas na instância de contêiner host (semelhante ao usuário raiz).

`aws:ResourceTag/${TagKey}` (String)

Filtra as ações com base nas tags associadas ao recurso.

`aws:RequestTag/${TagKey}` (String)

Filtra ações com base nas tags transmitidas na solicitação.

`batch:ShareIdentifier` (String)

Filtra ações com base no parâmetro `shareIdentifier` enviado para SubmitJob.

`aws:TagKeys` (String)

Filtra ações com base nas chaves de tag transmitidas na solicitação.

`batch:User` (String)

O nome de usuário ou ID de usuário numérico (uid) a ser usado dentro do contêiner para o trabalho.

Exemplo de políticas

Os exemplos a seguir mostram declarações de política que você pode usar para controlar as permissões que os usuários do têm para o AWS Batch.

Exemplos

- Acesso somente leitura
- Restringir a usuário de POSIX, imagem do Docker, nível de privilégio e função no envio do trabalho
- Restringir ao prefixo de definição de trabalho no envio de trabalho
- Restringir à fila de trabalhos
- Negar ação quando todas as chaves de condição correspondem às strings
- Negar ação quando qualquer chave de condição corresponder às strings
- Use a chave de condição batch:ShareIdentifier

Acesso somente leitura

A política a seguir concede permissões aos usuários para usar todas as ações de API do AWS Batch cujos nomes começam com `Describe` e `List`.

Os usuários não podem executar nenhuma ação nesses recursos ou criar novos recursos, a menos que outra declaração conceda permissão para eles fazerem isso. Por padrão, eles não têm permissão para usar ações de API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "batch:Describe*",
                "batch:List*"
            ],
            "Resource": "*"
        }
    ]
}
```

Restringir a usuário de POSIX, imagem do Docker, nível de privilégio e função no envio do trabalho

A política a seguir permite que um usuário POSIX gerencie seu próprio conjunto de definições de tarefas restritos.

Use a primeira e a segunda afirmações para registrar e cancele qualquer nome de definição de trabalho cujo nome seja prefixado com *JobDefA_*.

A primeira afirmação também usa chaves de contexto condicional para restringir o usuário de POSIX, status de privilegiado e valores de imagem de contêiner dentro das `containerProperties` de uma definição de tarefa. Para obter mais informações, consulte RegisterJobDefinition no documento Referência da API do AWS Batch. Neste exemplo, as definições de tarefas só podem ser registradas quando o usuário POSIX está definido como `nobody`. O sinalizador privilegiado está definido como `false`. Por exemplo, a imagem é definida como `myImage` em um repositório do Amazon ECR.

> **Important**
>
> O docker resolve o parâmetro `user` para o `uid` desse usuário de dentro da imagem do contêiner. Na maioria dos casos, isso é encontrado no arquivo `/etc/passwd` dentro da imagem do contêiner. Essa resolução de nomes pode ser evitada usando valores diretos de `uid` tanto na definição de tarefa quanto nas políticas do IAM associadas. Tanto as operações de API do AWS Batch quanto as chaves condicionais do `batch:User` do IAM suportam valores numéricos.

Use a terceira afirmação para restringir a apenas uma função específica a uma definição de trabalho.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "batch:RegisterJobDefinition"
            ],
            "Resource": [
                "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/JobDefA_*"
            ],
```

```
            "Condition": {
                "StringEquals": {
                    "batch:User": [
                        "nobody"
                    ],
                    "batch:Image": [
                        "<aws_account_id>.dkr.ecr.<aws_region>.amazonaws.com/myImage"
                    ]
                },
                "Bool": {
                    "batch:Privileged": "false"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "batch:DeregisterJobDefinition"
            ],
            "Resource": [
                "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/JobDefA_*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::<aws_account_id>:role/MyBatchJobRole"
            ]
        }
    ]
}
```

Restringir ao prefixo de definição de trabalho no envio de trabalho

Use a política a seguir para enviar trabalhos para qualquer fila de trabalhos com qualquer nome de definição de trabalho que comece com *JobDefA*.

> **Important**
>
> Para o escopo de acesso em nível de recurso para envio de tarefa, você deve fornecer os tipos de recursos de definição de tarefa e fila de tarefas.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:SubmitJob"
      ],
      "Resource": [
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/JobDefA_*",
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-queue/*"
      ]
    }
  ]
}
```

Restringir à fila de trabalhos

Use a política a seguir para enviar trabalhos para uma fila de trabalhos específica nomeada queue1 com qualquer nome de definição de trabalho.

> **Important**
>
> Para o escopo de acesso em nível de recurso para envio de tarefa, você deve fornecer os tipos de recursos de definição de tarefa e fila de tarefas.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:SubmitJob"
```

```
      ],
      "Resource": [
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-definition/*",
        "arn:aws:batch:<aws_region>:<aws_account_id>:job-queue/queue1"
      ]
    }
  ]
}
```

Negar ação quando todas as chaves de condição correspondem às strings

A política a seguir nega acesso à operação RegisterJobDefinition da API quando a chave de condição `batch:Image` (ID da imagem do contêiner) for "*`string1`*" e a chave de condição `batch:LogDriver` (driver de log do contêiner) for "*`string2`*." AWS Batch avalia as chaves de condição em cada contêiner. Quando um trabalho abrange vários contêineres, como um trabalho paralelo de vários nós, é possível que os contêineres tenham configurações diferentes. Se várias chaves de condição forem avaliadas em uma declaração, elas serão combinadas usando a lógica AND. Portanto, se alguma das várias chaves de condição não corresponder a um contêiner, o efeito Deny não será aplicado a esse contêiner. Em vez disso, um contêiner diferente no mesmo trabalho pode ser negado.

Para a lista de chaves de condição do AWS Batch, consulte Chaves de condição do AWS Batch na Referência de autorização do serviço. Com exceção de `batch:ShareIdentifier`, todas as chaves de condição do `batch` podem ser usadas dessa forma. A chave de condição `batch:ShareIdentifier` é definida para um trabalho, não para uma definição de trabalho.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Deny",
      "Action": "batch:RegisterJobDefinition",
```

```
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "batch:Image": "string1",
          "batch:LogDriver": "string2"
        }
      }
    }
  ]
}
```

Negar ação quando qualquer chave de condição corresponder às strings

A política a seguir nega acesso à operação RegisterJobDefinition da API quando ou chave de condição `batch:Image` (ID da imagem do contêiner) for "*string1*" ou a chave de condição `batch:LogDriver` (driver de log do contêiner) for "*string2*." Quando um trabalho abrange vários contêineres, como um trabalho paralelo de vários nós, é possível que os contêineres tenham configurações diferentes. Se várias chaves de condição forem avaliadas em uma declaração, elas serão combinadas usando a lógica AND. Portanto, se alguma das várias chaves de condição não corresponder a um contêiner, o efeito `Deny` não será aplicado a esse contêiner. Em vez disso, um contêiner diferente no mesmo trabalho pode ser negado.

Para a lista de chaves de condição do AWS Batch, consulte Chaves de condição do AWS Batch na Referência de autorização do serviço. Com exceção de `batch:ShareIdentifier`, todas as chaves de condição do `batch` podem ser usadas dessa forma. (A chave de condição `batch:ShareIdentifier` é definida para um trabalho, não para uma definição de trabalho.)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Deny",
```

```
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "batch:Image": [
            "string1"
          ]
        }
      }
    },
    {
      "Effect": "Deny",
      "Action": [
        "batch:RegisterJobDefinition"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringEquals": {
          "batch:LogDriver": [
            "string2"
          ]
        }
      }
    }
  ]
}
```

Use a chave de condição **batch:ShareIdentifier**

Use a política a seguir para enviar trabalhos que usam a definição de trabalho `jobDefA` para a fila de trabalhos `jobqueue1` com o identificador de compartilhamento `lowCpu`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "batch:SubmitJob"
      ],
      "Resource": [
        "arn:aws::batch:<aws_region>:<aws_account_id>:job-definition/JobDefA",
        "arn:aws::batch:<aws_region>:<aws_account_id>:job-queue/jobqueue1"
      ],
      "Condition": {
        "StringEquals": {
          "batch:ShareIdentifier": [
            "lowCpu"
          ]
        }
      }
    }
  ]
}
```

AWS Batch Política gerenciada

AWS Batch fornece uma política gerenciada que pode ser anexada aos usuários que concederem permissão para usar AWS Batch recursos e operações API. Você pode aplicar essa política diretamente ou usá-la como ponto de partida para criar suas próprias políticas. Para mais informações sobre cada uma das operações API mencionadas nessas políticas, consulte Ações em AWS BatchReferência API.

AWSBatchFullAccess

Esta política permite o acesso total do administrador a AWS Batch.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "batch:*",
        "cloudwatch:GetMetricStatistics",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeVpcs",
```

```
        "ec2:DescribeImages",
        "ec2:DescribeLaunchTemplates",
        "ec2:DescribeLaunchTemplateVersions",
        "ecs:DescribeClusters",
        "ecs:Describe*",
        "ecs:List*",
        "eks:DescribeCluster",
        "eks:ListClusters",
        "logs:Describe*",
        "logs:Get*",
        "logs:TestMetricFilter",
        "logs:FilterLogEvents",
        "iam:ListInstanceProfiles",
        "iam:ListRoles"
      ],
      "Resource":"*"
    },
    {
      "Effect":"Allow",
      "Action":[
        "iam:PassRole"
      ],
      "Resource":[
        "arn:aws:iam::*:role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/service-role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/ecsInstanceRole",
        "arn:aws:iam::*:instance-profile/ecsInstanceRole",
        "arn:aws:iam::*:role/iaws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/aws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/AWSBatchJobRole*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "iam:CreateServiceLinkedRole"
      ],
      "Resource":"arn:aws:iam::*:role/*Batch*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "batch.amazonaws.com"
          }
      }
    }
```

```
  ]
}
```

Criando AWS Batch políticas do IAM

Você pode criar políticas do IAM específicas para restringir chamadas e recursos a que os usuários em sua conta têm acesso. Em seguida, você pode anexar tais políticas aos usuários.

Ao anexar uma política a um usuário ou grupo de usuários, ela concede ou nega aos usuários permissão para realizar tarefas específicas em recursos específicos. Para mais informações, consulte Permissões e Políticas no Guia de Usuário do IAM. Para instruções sobre como gerenciar e criar políticas do IAM personalizadas, consulte Gerenciando Políticas do IAM.

Perfil de instância do Amazon ECS

AWS Batch ambientes de computação são preenchidos com instâncias de contêiner Amazon ECS. Eles executam o atendente de contêiner Amazon ECS localmente. O atendente de contêiner do Amazon ECS faz chamadas para várias AWSoperações API em seu nome. Portanto, as instâncias de contêiner que executam o atendente exigem uma política e um perfil do IAM para esses serviços para reconhecerem que o atendente pertence a você. Você deve criar um perfil do IAM e um perfil de instância para as instâncias de contêiner poderem usar quando iniciadas. Do contrário, você não poderá criar um ambiente de computação e iniciar instâncias de contêiner nele. Esse requisito se aplica a instâncias de contêiner iniciadas com ou sem o AMI otimizado Amazon ECS, fornecido pela Amazon. Para mais informações, consulte Perfil do IAM da instância de contêiner Amazon ECS no Guia de Desenvolvedor do Amazon Elastic Container Service.

A função e o perfil de instância Amazon ECS são criados automaticamente para você durante a primeira execução do console. No entanto, é possível seguir esses passos para verificar se a sua conta já possui a função e perfil de instância Amazon ECS. As etapas a seguir também abordam como anexar a política do IAM gerenciada.

Para verificar a **`ecsInstanceRole`** no console do IAM

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Roles (Funções).
3. Procure na lista de perfis por `ecsInstanceRole`. Caso o perfil não exista, use os seguintes passos para criá-lo.

a. Escolha Criar Perfil.

b. Em Tipo de Entidade Confiável, escolha AWS service (Serviço da AWS).

c. Em Ocasiões de Uso Comuns, escolhaEC2.

d. Escolha Próximo.

e. Em Políticas de Permissões, pesquise por AmazonEC2ContainerServiceforEC2Role.

f. Escolha a opção próxima a AmazonEC2ContainerServiceforEC2Role e, em seguida, Próximo.

g. Em Nome do Perfil, digite `ecsInstanceRole` e selecione Criar Perfil.

> **Note**
>
> Caso use AWS Management Console para criar um perfil para Amazon EC2, o console criará um perfil de instância de mesmo nome.

Como alternativa, é possível usar AWS CLI para criar o `ecsInstanceRole` perfil do IAM. O exemplo a seguir cria um perfil do IAM com uma política de confiança e uma AWS política gerenciada.

Para criar um perfil do IAM e um de instância (AWS CLI)

1. Crie a seguinte política de confiança e salve-a em um arquivo de texto chamado `ecsInstanceRole-role-trust-policy.json`.

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

2. Use o comando create-role para criar o `ecsInstanceRole` perfil. Especifique a localização do arquivo da política de confiança no `assume-role-policy-document` parâmetro.

```
$ aws iam create-role \
    --role-name ecsInstanceRole \
    --assume-role-policy-document file://ecsInstanceRole-role-trust-policy.json
```

A seguir, uma exemplo de resposta.

```
{
    "Role": {
         "Path": "/",
         "RoleName: "ecsInstanceRole",
         "RoleId": "AROAT46P5RDIY4EXAMPLE",
         "Arn": "arn:aws:iam::123456789012:role/ecsInstanceRole".
        "CreateDate": "2022-12-12T23:46:37.247Z",
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Effect": "Allow",
                    "Principal": {
                        "Service: "ec2.amazonaws.com"
                    }
                    "Action": "sts:AssumeRole",
                 }
              ]
          }
        }
```

3. Use o comando create-instance-profile para criar um perfil de instância chamado `ecsInstanceRole`.

> **Note**
>
> Você precisa criar funções e perfis de instância como ações separadas no AWS CLI e na AWS API.

```
$ aws iam create-instance-profile --instance-profile-name ecsInstanceRole
```

A seguir, uma exemplo de resposta.

```
{
    "InstanceProfile": {
        "Path": "/",
        "InstanceProfileName": "ecsInstanceRole",
        "InstanceProfileId": "AIPAT46P5RDITREXAMPLE",
        "Arn": "arn:aws:iam::123456789012:instance-profile/ecsInstanceRole",
        "CreateDate": "2022-06-30T23:53:34.093Z",
        "Roles": [],    }
}
```

4. Use o comando add-role-to-instance-profile para adicionar a `ecsInstanceRole` função ao `ecsInstanceRole` perfil de instância.

```
aws iam add-role-to-instance-profile \
    --role-name ecsInstanceRole --instance-profile-name ecsInstanceRole
```

5. Use o comando attach-role-policy para anexar a `AmazonEC2ContainerServiceforEC2Role` AWS política gerenciada ao `ecsInstanceRole` perfil.

```
$ aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role \
    --role-name ecsInstanceRole
```

Perfil de frota spot Amazon EC2

Caso crie um ambiente de computação gerenciado que faça uso de instâncias de frota spot Amazon EC2, você deve criar a política de `AmazonEC2SpotFleetTaggingRole`. Esta política concede à frota spot permissão para iniciar, marcar e encerrar instâncias em seu nome. Especifique a função em sua solicitação de frota Spot. Você também precisa ter perfis AWSServiceRoleForEC2Spot e AWSServiceRoleForEC2SpotFleet vinculados a serviços para Amazon EC2 Spot e frota spot. Use as instruções a seguir para criar estes perfis. Para mais informações, consulte Como Usar Perfis Vinculados a Serviço e Como Criar um Perfil para Delegar Permissões a um AWS Serviço no Guia de Usuário do IAM.

Tópicos

- Crie perfils de frota spot Amazon EC2 no AWS Management Console
- Crie perfis de frota spot Amazon EC2 com AWS CLI

Crie perfils de frota spot Amazon EC2 no AWS Management Console

Para criar o **`AmazonEC2SpotFleetTaggingRole`** perfil IAM vinculado ao serviço para a frota spot Amazon EC2

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. Em Gerenciamento de Acesso, escolha Perfis,
3. Em Perfis, escolha Criar Perfil.
4. De Selecionar Entidade Confiável para Tipo de Entidade Confiável, escolha AWS service (Serviço da AWS).
5. Em Casos de Uso para Outros Serviços da AWS, escolha EC2 e, em seguida, EC2 – Marcação de Frota Spot.
6. Escolha Próximo.
7. De Políticas de Permissões para Nome da Política, verifique `AmazonEC2SpotFleetTaggingRole`.
8. Escolha Próximo.
9. Em Nomear, Revisar e Criar:
   a. Em Nome do Perfil, insira um nome para identificar o perfil.
   b. Em Descrição, insira uma breve explicação para a política.
   c. (Opcional) Para Passo 1: Selecionar Entidades Confiáveis, escolha Editar para modificar o código.
   d. (Opcional) Para Passo 2: Adicionar Permissões, escolha Editar para modificar o código.
   e. (Opcional) Em Adicionar Marcações, escolha Adicionar Marcação para adicionar marcações ao recurso.
   f. Escolha Criar Perfil.

> **Note**
>
> No passado, havia duas políticas gerenciadas para o perfil de frota spot Amazon EC2.
>
> - AmazonEC2SpotFleetRole: esta é a política gerenciada original para o perfil de frota spot. No entanto, não recomendamos mais usá-la com AWS Batch. Essa política não oferece suporte à marcação de Instância Spot em ambientes de computação, necessário para utilização do `AWSServiceRoleForBatch` perfil vinculado ao serviço. Caso já tenha

criado um perfil de frota spot com essa política, aplique a nova política recomendada ao perfil. Para mais informações, consulte Instâncias spot sem tags na criação.

- AmazonEC2SpotFleetTaggingRole: este perfil fornece todas as permissões necessárias para marcar instâncias spot Amazon EC2. Use este perfil para permitir marcação de Instâncias Spot em seus AWS Batch ambientes de computação.

Crie perfis de frota spot Amazon EC2 com AWS CLI

Para criar o perfil do IAM AmazonEC2SpotFleetTaggingRole para seus ambientes de computação de Frota Spot

1. Execute o seguinte comando com AWS CLI.

```
$ aws iam create-role --role-name AmazonEC2SpotFleetTaggingRole \
    --assume-role-policy-document '{
    "Version":"2012-10-17",
    "Statement":[
       {
         "Sid":"",
         "Effect":"Allow",
         "Principal": {
           "Service":"spotfleet.amazonaws.com"
          },
          "Action":"sts:AssumeRole"
        }
      ]
    }'
```

2. Para anexar a política gerenciada do IAM AmazonEC2SpotFleetTaggingRole ao seu AmazonEC2SpotFleetTaggingRole perfil, execute o comando a seguir com AWS CLI.

```
$ aws iam attach-role-policy \
  --policy-arn \
    arn:aws:iam::aws:policy/service-role/AmazonEC2SpotFleetTaggingRole \
  --role-name \
    AmazonEC2SpotFleetTaggingRole
```

Para criar um **`AWSServiceRoleForEC2Spot`** perfil IAM vinculado ao serviço Amazon EC2 Spot

> **Note**
>
> Caso o `AWSServiceRoleForEC2Spot` perfil do IAM vinculado ao serviço já exista, você verá uma mensagem de erro semelhante à seguinte.
>
> ```
> An error occurred (InvalidInput) when calling the CreateServiceLinkedRole
>  operation:
> Service role name AWSServiceRoleForEC2Spot has been taken in this account,
>  please try a different suffix.
> ```

- Execute o seguinte comando com AWS CLI.

```
$ aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
```

Para criar o **`AWSServiceRoleForEC2SpotFleet`** perfil do IAM vinculado ao serviço para frota spot Amazon EC2

> **Note**
>
> Caso o `AWSServiceRoleForEC2SpotFleet` perfil do IAM vinculado ao serviço já exista, você verá uma mensagem de erro semelhante à seguinte.
>
> ```
> An error occurred (InvalidInput) when calling the CreateServiceLinkedRole
>  operation:
> Service role name AWSServiceRoleForEC2SpotFleet has been taken in this account,
>  please try a different suffix.
> ```

- Execute o seguinte comando com AWS CLI.

```
$ aws iam create-service-linked-role --aws-service-name spotfleet.amazonaws.com
```

Perfil do IAM do EventBridge

O Amazon EventBridge fornece um fluxo em tempo quase real de eventos de sistema que descrevem alterações nos AWS recursos. AWS Batch trabalhos estão disponíveis como destinos do EventBridge. Por meio de regras simples e rapidamente configuráveis, você pode equivaler eventos e submeter AWS Batch trabalhos em resposta. Antes de submeter AWS Batch trabalhos com regras e destinos do EventBridge, o EventBridge precisa de permissões para executar AWS Batch trabalhos em seu nome.

> **Note**
>
> Ao criar uma regra no console do EventBridge especificando uma AWS Batch fila como destino, você pode criar esse perfil. Para ver uma demonstração de exemplo, consulte AWS Batch empregos como EventBridge alvos. Você pode criar o perfil do EventBridge manualmente usando o console do IAM. Para instruções, consulte Criando um Perfil Usando Políticas de Confiança Customizadas (Console) no Guia de Usuário do IAM.

O relacionamento de confiança para seu perfil do IAM do EventBridge deve fornecer à `events.amazonaws.com` entidade principal do serviço a capacidade de assumir o perfil.

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

Certifique-se de que a política anexada ao seu perfil do IAM do EventBridge conceda `batch:SubmitJob` permissões em seus recursos. No exemplo a seguir, AWS Batch fornece a `AWSBatchServiceEventTargetRole` a política gerenciada para essas permissões.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "batch:SubmitJob"
       ],
      "Resource": "*"
    }
  ]
}
```

AWS Batch Stream de eventos para Amazon EventBridge

Você pode usar o stream de AWS Batch eventos da Amazon EventBridge para receber notificações quase em tempo real sobre o estado atual dos trabalhos em suas filas de trabalho.

Você pode usar EventBridge para obter mais informações sobre seu AWS Batch serviço. Mais especificamente, você pode usá-lo para verificar o progresso dos trabalhos, criar fluxos de trabalho AWS Batch personalizados, gerar relatórios ou métricas de uso ou criar seus próprios painéis. Com AWS Batch e EventBridge, você não precisa de um código de agendamento e monitoramento que pesquise continuamente as mudanças no AWS Batch status do trabalho. Em vez disso, você pode lidar com as mudanças no estado do AWS Batch trabalho de forma assíncrona usando uma variedade de destinos da Amazon. EventBridge Isso inclui AWS Lambda o Amazon Simple Queue Service, o Amazon Simple Notification Service ou o Amazon Kinesis Data Streams.

É garantido que os AWS Batch eventos do stream de eventos sejam entregues pelo menos uma vez. Caso eventos duplicados sejam enviados, o evento fornece informações o suficiente para identificar cópias. Dessa forma, você pode comparar a data e hora do evento e o status do trabalho.

AWS Batch empregos estão disponíveis como EventBridge alvos. Usando regras simples, você pode combinar eventos e enviar AWS Batch trabalhos em resposta a eles. Para obter mais informações, consulte O que é EventBridge? no Guia do EventBridge usuário da Amazon. Você também pode usar EventBridge para programar ações automatizadas que são acionadas automaticamente em determinados momentos usando cron ou avaliando expressões. Para obter mais informações, consulte Criação de uma EventBridge regra da Amazon que é executada de acordo com uma programação no Guia EventBridge do usuário da Amazon. Para ver uma demonstração de exemplo, consulte AWS Batch empregos como EventBridge alvos. Para obter informações sobre o uso do EventBridge Agendador, consulte Configurando o Amazon EventBridge Scheduler no Guia EventBridge do Usuário da Amazon.

Tópicos

- AWS Batch Eventos
- Usando notificações AWS de usuário com AWS Batch
- AWS Batch empregos como EventBridge alvos
- Tutorial: Receptando para AWS Batch EventBridge
- Tutorial: Como enviar alertas do Amazon Simple Notification Service para eventos de trabalho que falharam

AWS Batch Eventos

AWS Batch envia eventos de alteração do status do trabalho para EventBridge. AWS Batch monitora o estado de seus empregos. Se o status de um trabalho enviado anteriormente mudar, um evento será invocado. Por exemplo, se um trabalho no status `RUNNING` passar para o status `FAILED`. Esses eventos são classificados como eventos de alteração do estado de trabalho.

> **Note**
>
> AWS Batch pode adicionar outros tipos de eventos, fontes e detalhes no futuro. Caso você esteja desserializando programaticamente dados JSON do evento, certifique-se de que a aplicação esteja preparada para lidar com propriedades desconhecidas. Isso é para evitar problemas se e quando essas propriedades adicionais forem adicionadas.

Eventos de alteração de estado do trabalho

Toda vez que um trabalho (enviado anteriormente) mudar de estado, um evento será criado. Para obter mais informações sobre estados de AWS Batch trabalho, consulteEstados da tarefa.

> **Note**
>
> Os eventos não são criados para o envio de trabalho inicial.

Example Evento de alteração de estado do trabalho

Os eventos de alteração do estado da tarefa são entregues no formato a seguir. A `detail` seção é semelhante ao JobDetailobjeto retornado de uma operação de DescribeJobsAPI na Referência da AWS Batch API. Para obter mais informações sobre EventBridge parâmetros, consulte Eventos e padrões de eventos no Guia EventBridge do usuário da Amazon.

```
{
    "version": "0",
    "id": "c8f9c4b5-76e5-d76a-f980-7011e206042b",
    "detail-type": "Batch Job State Change",
    "source": "aws.batch",
    "account": "123456789012",
    "time": "2022-01-11T23:36:40Z",
    "region": "us-east-1",
```

```
    "resources": [
        "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-ba5a-4727fcce14a8"
    ],
    "detail": {
        "jobArn": "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-
ba5a-4727fcce14a8",
        "jobName": "event-test",
        "jobId": "4c7599ae-0a82-49aa-ba5a-4727fcce14a8",
        "jobQueue": "arn:aws:batch:us-east-1:123456789012:job-queue/
PexjEHappyPathCanary2JobQueue",
        "status": "RUNNABLE",
        "attempts": [],
        "createdAt": 1641944200058,
        "retryStrategy": {
            "attempts": 2,
            "evaluateOnExit": []
        },
        "dependsOn": [],
        "jobDefinition": "arn:aws:batch:us-east-1:123456789012:job-definition/first-
run-job-definition:1",
        "parameters": {},
        "container": {
            "image": "137112412989.dkr.ecr.us-east-1.amazonaws.com/amazonlinux:latest",
            "command": [
                "sleep",
                "600"
            ],
            "volumes": [],
            "environment": [],
            "mountPoints": [],
            "ulimits": [],
            "networkInterfaces": [],
            "resourceRequirements": [
                {
                    "value": "2",
                    "type": "VCPU"
                }, {
                    "value": "256",
                    "type": "MEMORY"
                }
            ],
            "secrets": []
        },
        "tags": {
```

```
            "resourceArn": "arn:aws:batch:us-
east-1:123456789012:job/4c7599ae-0a82-49aa-ba5a-4727fcce14a8"
        },
        "propagateTags": false,
        "platformCapabilities": []
    }
}
```

Eventos bloqueados na fila de trabalhos

Sempre que AWS Batch detectar um trabalho no `RUNNABLE` estado e, assim, bloquear uma fila, um evento é criado no Amazon Events. CloudWatch Para obter mais informações sobre as causas suportadas de filas bloqueadas, consulte exemplos de mensagens de fila de trabalhos bloqueados. O mesmo motivo também está disponível no `statusReason` campo na ação da `DescribeJobs` API.

Example Evento de alteração de estado do trabalho

Os eventos de alteração do estado da tarefa são entregues no formato a seguir. A `detail` seção é semelhante ao JobDetailobjeto retornado de uma operação de DescribeJobsAPI na Referência da AWS Batch API. Para obter mais informações sobre EventBridge parâmetros, consulte Eventos e padrões de eventos no Guia EventBridge do usuário da Amazon.

```
{
    "version": "0",
    "id": "c8f9c4b5-76e5-d76a-f980-7011e206042b",
    "detail-type": "Batch Job Queue Blocked",
    "source": "aws.batch",
    "account": "123456789012",
    "time": "2022-01-11T23:36:40Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-
ba5a-4727fcce14a8",
        "arn:aws:batch:us-east-1:123456789012:job-queue/PexjEHappyPathCanary2JobQueue"
    ],
    "detail": {
        "jobArn": "arn:aws:batch:us-east-1:123456789012:job/4c7599ae-0a82-49aa-
ba5a-4727fcce14a8",
        "jobName": "event-test",
        "jobId": "4c7599ae-0a82-49aa-ba5a-4727fcce14a8",
```

```
        "jobQueue": "arn:aws:batch:us-east-1:123456789012:job-queue/
PexjEHappyPathCanary2JobQueue",
        "status": "RUNNABLE",
        "statusReason": "blocked-reason"
        "attempts": [],
        "createdAt": 1641944200058,
        "retryStrategy": {
            "attempts": 2,
            "evaluateOnExit": []
        },
        "dependsOn": [],
        "jobDefinition": "arn:aws:batch:us-east-1:123456789012:job-definition/first-
run-job-definition:1",
        "parameters": {},
        "container": {
            "image": "137112412989.dkr.ecr.us-east-1.amazonaws.com/amazonlinux:latest",
            "command": [
                "sleep",
                "600"
            ],
            "volumes": [],
            "environment": [],
            "mountPoints": [],
            "ulimits": [],
            "networkInterfaces": [],
            "resourceRequirements": [
                {
                    "value": "2",
                    "type": "VCPU"
                }, {
                    "value": "256",
                    "type": "MEMORY"
                }
            ],
            "secrets": []
        },
        "tags": {
            "resourceArn": "arn:aws:batch:us-
east-1:123456789012:job/4c7599ae-0a82-49aa-ba5a-4727fcce14a8"
        },
        "propagateTags": false,
        "platformCapabilities": []
    }
```

```
}
```

Usando notificações AWS de usuário com AWS Batch

Você pode usar as notificações de usuários da AWS para configurar canais de entrega para receber notificações sobre eventos do AWS Batch . Você recebe uma notificação quando um evento corresponde a uma regra especificada. É possível receber notificações de eventos por meio de vários canais, incluindo e-mail, notificações de chat do AWS Chatbot ou notificações por push do AWS Console Mobile Application. Você também pode ver as notificações na Central de notificações do console. As notificações do usuário permitem agregação, o que pode reduzir o número de notificações recebidas durante eventos específicos.

Para configurar as notificações do usuário em AWS Batch:

1. Abra o console de AWS Batch.
2. Escolha Dashboard.
3. Escolha Configurar notificações.
4. Em Notificações AWS do usuário, escolha Criar configuração de notificação.

Para obter mais informações sobre como configurar e visualizar as notificações do usuário, consulte Introdução às notificações AWS do usuário.

AWS Batch empregos como EventBridge alvos

EventBridgeA Amazon fornece um fluxo quase em tempo real de eventos do sistema que descrevem mudanças nos recursos da Amazon Web Services. Normalmente, AWS Batch no Amazon Elastic Container Service, no Amazon Elastic Kubernetes Service AWS e no Fargate, os trabalhos estão disponíveis como destinos. EventBridge Usando regras simples, você pode combinar eventos e enviar AWS Batch trabalhos em resposta a eles. Para obter mais informações, consulte O que é EventBridge? no Guia do EventBridge usuário da Amazon.

Você também pode usar EventBridge para programar ações automatizadas que são invocadas em determinados momentos usando cron ou classificando expressões. Para obter mais informações, consulte Criação de uma EventBridge regra da Amazon que é executada de acordo com uma programação no Guia EventBridge do usuário da Amazon.

Para obter informações sobre como criar uma regra que é executada quando um evento corresponde a um padrão de evento, consulte Criação de EventBridge regras da Amazon que reagem a eventos no Guia EventBridge do usuário da Amazon.

Os casos de uso comuns para AWS Batch trabalhos como EventBridge alvo incluem os seguintes casos de uso:

- Um trabalho agendado ocorre em intervalos de tempo regulares. Por exemplo, um trabalho cron ocorre somente em horários de baixa utilização, quando as Instâncias Spot do Amazon EC2 são mais baratas.
- Um AWS Batch trabalho é executado em resposta a uma operação de API que está conectada. CloudTrail Por exemplo, um trabalho é enviado sempre que um objeto é carregado em um bucket específico do Amazon S3. Sempre que isso acontece, o transformador EventBridge de entrada passa o bucket e o nome da chave do objeto para AWS Batch os parâmetros.

> **Note**
>
> Nesse cenário, todos os AWS recursos relacionados devem estar na mesma região. Isso inclui recursos como o bucket, a EventBridge regra e CloudTrail os registros do Amazon S3.

Antes de enviar AWS Batch trabalhos com EventBridge regras e metas, o EventBridge serviço exige várias permissões para executar AWS Batch trabalhos. Ao criar uma regra no EventBridge console que especifica um AWS Batch trabalho como destino, você também pode criar essa função. Para obter mais informações sobre a entidade principal do serviço e as permissões do IAM necessárias para esse perfil, consulte Perfil do IAM do EventBridge.

Criação de um AWS Batch trabalho agendado

O procedimento a seguir aborda como criar um AWS Batch trabalho agendado e a função necessária EventBridge do IAM.

Para criar um AWS Batch trabalho agendado com EventBridge

> **Note**
>
> Esse procedimento funciona para todos os trabalhos AWS Batch do Amazon ECS, Amazon EKS e AWS Fargate.

1. Abra o EventBridge console da Amazon em https://console.aws.amazon.com/events/.
2. Na barra de navegação, selecione o Região da AWS a ser usado.
3. No painel de navegação, escolha Regras.
4. Escolha Create rule.
5. Para Nome, especifique um nome exclusivo para seu ambiente de computação. O nome pode conter até 64 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).

   > **Note**
   >
   > Uma regra não pode ter o mesmo nome que outra na mesma Região e barramento de eventos.

6. (Opcional) Em Description, insira uma descrição para a regra.
7. Em Event bus, escolha o barramento de eventos que você deseja associar a essa regra. Se quiser que essa regra faça a correspondência com eventos provenientes da sua conta, selecione padrão. Quando um AWS service (Serviço da AWS) em sua conta emite um evento, ele sempre vai para o ônibus de eventos padrão da sua conta.
8. (Opcional) Desative a regra no barramento selecionado se não quiser executá-la imediatamente.
9. Em Rule type, escolha Schedule.
10. Escolha Continuar para criar a regra ou Avançar.
11. Em Schedule pattern, siga um destes procedimentos:

    - Escolha um cronograma refinado que seja executado em um horário específico, tal como 8:00 a.m. PST na primeira segunda-feira de cada mês e insira a expressão cron. Para obter mais informações, consulte Expressões Cron no Guia do EventBridge usuário da Amazon.
    - Escolha uma programação que seja executada em uma taxa regular, como a cada 10 minutos. e, em seguida, insira uma expressão rate.

12. Escolha Next (Próximo).
13. Em Target types (Tipos de destino), escolha AWS service (Serviço da AWS).
14. Em Selecionar um destino, escolha Fila de trabalhos em lote. Em seguida, configure o seguinte:

    - Job queue: (Fila de trabalhos:), insira Nome de recurso da Amazon (ARN) da fila de trabalhos na qual o trabalho será programado.
    - Job definition: (Definição do trabalho:) insira o nome e a revisão ou ARN completo da definição do trabalho a ser usado.
    - Job name: (Nome do trabalho:) insira um nome para o trabalho.
    - Array size: (Tamanho da matriz:) (opcional) insira um tamanho de matriz para que o trabalho execute mais de uma cópia. Para ter mais informações, consulte Trabalhos de matriz.
    - Job attempts: (opcional) insira o número de vezes para tentar novamente caso ocorra uma falha no trabalho. Para ter mais informações, consulte Repetições de trabalho automatizadas.
15. Para tipos de destino de fila de trabalhos em Batch, EventBridge precisa de permissão para enviar eventos para o destino. EventBridge pode criar a função do IAM necessária para que sua regra seja executada. Execute um destes procedimentos:

    - Para criar um perfil do IAM automaticamente, escolha Create a new role for this specific resource.
    - Para usar uma função do IAM que você criou anteriormente, escolha Use existing role (Usar função existente)
16. (Opcional) Expanda Additional settings.

    a. Em Configurar entrada de destino, escolha como o texto de um evento é processado antes de ser passado para o destino.

    b. Em Idade máxima do evento, especifique o intervalo de tempo por quanto tempo os eventos não processados são mantidos.

    c. Em Tentativas de repetição, insira o número de vezes que um evento é repetido.

    d. Em Dead-letter queue, escolha uma opção de como os eventos não processados são tratados. Se necessário, especifique a fila do Amazon SQS para usar como fila de mensagens mortas.
17. (Opcional) Selecione Add another target para adicionar outro destino a essa regra.
18. Escolha Next (Próximo).

19. (Opcional) Em Tags, escolha Adicionar nova tag para adicionar um rótulo de recurso à regra. Para obter mais informações, consulte as EventBridge tags da Amazon.
20. Escolha Next (Próximo).
21. Para Revisar e criar, revise as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Create rule.

Para obter mais informações sobre a criação de regras, consulte Criação de uma EventBridge regra da Amazon que é executada de acordo com uma programação no Guia EventBridge do usuário da Amazon.

Criar uma regra com um padrão de evento

O procedimento a seguir aborda como criar uma regra com um padrão de evento.

Para criar uma regra que envia o evento para um destino quando o evento corresponde a um padrão definido

> **Note**
>
> Esse procedimento funciona para todos os trabalhos AWS Batch do Amazon ECS, Amazon EKS e AWS Fargate.

1. Abra o EventBridge console da Amazon em https://console.aws.amazon.com/events/.
2. Na barra de navegação, selecione o Região da AWS a ser usado.
3. No painel de navegação, escolha Regras.
4. Escolha Create rule.
5. Para Nome, especifique um nome exclusivo para seu ambiente de computação. O nome pode conter até 64 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).

   > **Note**
   >
   > Uma regra não pode ter o mesmo nome que outra na mesma Região e barramento de eventos.

6. (Opcional) Em Description, insira uma descrição para a regra.
7. Em Event bus, escolha o barramento de eventos que você deseja associar a essa regra. Se quiser que essa regra faça a correspondência com eventos provenientes da sua conta, selecione padrão. Quando um AWS service (Serviço da AWS) em sua conta emite um evento, ele sempre vai para o ônibus de eventos padrão da sua conta.
8. (Opcional) Desative a regra no barramento selecionado se não quiser executá-la imediatamente.
9. Em Rule type, escolha Rule with an event pattern.
10. Selecione Next (Próximo).
11. Em Origem do evento, escolha AWS eventos ou eventos de EventBridge parceiros.
12. (Opcional) Para evento de amostra:
    a. Em Tipo de evento de amostra, escolha AWS eventos.
    b. Em Eventos de amostra, escolha Batch Job State Change.
13. Em Creation method, escolha Use pattern form.
14. Para padrão de eventos:
    a. Em Event source, escolha Serviços da AWS.
    b. Em AWS service (Serviço da AWS), escolha Batch.
    c. Em Event type, escolha Batch Job State Change.
15. Escolha Next (Próximo).
16. Em Target types (Tipos de destino), escolha AWS service (Serviço da AWS).
17. Em Choose a target type, selecione o tipo de destino. Por exemplo, escolha Batch job queue. Em seguida, especifique o seguinte:
    - Job queue: (Fila de trabalhos:), insira Nome de recurso da Amazon (ARN) da fila de trabalhos na qual o trabalho será programado.
    - Job definition: (Definição do trabalho:) insira o nome e a revisão ou ARN completo da definição do trabalho a ser usado.
    - Job name: (Nome do trabalho:) insira um nome para o trabalho.
    - Array size: (Tamanho da matriz:) (opcional) insira um tamanho de matriz para que o trabalho execute mais de uma cópia. Para ter mais informações, consulte Trabalhos de matriz.
    - Job attempts: (opcional) insira o número de vezes para tentar novamente caso ocorra uma falha no trabalho. Para ter mais informações, consulte Repetições de trabalho automatizadas.

18. Para tipos de destino de fila de trabalhos em Batch, EventBridge precisa de permissão para enviar eventos para o destino. EventBridge pode criar a função do IAM necessária para que sua regra seja executada. Execute um destes procedimentos:

    - Para criar um perfil do IAM automaticamente, escolha Create a new role for this specific resource.
    - Para usar um perfil do IAM que você criou antes, escolha Use existing role.

19. (Opcional) Expanda Additional settings.

    a. Em Configure target input, escolha como o texto de um evento é processado.

    b. Em Idade máxima do evento, especifique o intervalo de tempo por quanto tempo os eventos não processados são mantidos.

    c. Em Tentativas de repetição, insira o número de vezes que um evento é repetido.

    d. Em Dead-letter queue, escolha uma opção de como os eventos não processados são tratados. Se necessário, especifique a fila do Amazon SQS para usar como fila de mensagens mortas.

20. (Opcional) Selecione Add another target (Adicionar outro destino) para adicionar outro destino a essa regra.
21. Escolha Next (Próximo).
22. (Opcional) Em Tags, escolha Adicionar nova tag para adicionar um rótulo de recurso. Para obter mais informações, consulte as EventBridge tags da Amazon no Guia EventBridge do usuário da Amazon.
23. Escolha Next (Próximo).
24. Para Revisar e criar, revise as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Create rule.

    Para obter mais informações sobre a criação de regras, consulte Criação de EventBridge regras da Amazon que reagem a eventos no Guia EventBridge do usuário da Amazon.

Passando informações do evento para um AWS Batch alvo em um cronograma usando o transformador EventBridge de entrada

Você pode usar o transformador EventBridge de entrada para transmitir informações do evento AWS Batch em um envio de trabalho. Isso pode ser especialmente valioso se você invocar trabalhos como resultado de outras informações do evento AWS . Um exemplo é upload de objeto para um bucket

do Amazon S3. Você também pode usar uma definição de tarefa com valores de substituição de parâmetros no comando do contêiner. O transformador EventBridge de entrada pode fornecer os valores dos parâmetros com base nos dados do evento.

Depois, você cria um destino de AWS Batch evento que analisa as informações do evento que o inicia e as transforma em um `parameters` objeto. Quando o trabalho é executado, os parâmetros do evento do acionador são repassados para o comando do contêiner do trabalho.

> **Note**
>
> Nesse cenário, todos os AWS recursos (como buckets, EventBridge regras e CloudTrail registros do Amazon S3) devem estar na mesma região.

Para criar um AWS Batch alvo que usa o transformador de entrada

1. Abra o EventBridge console da Amazon em https://console.aws.amazon.com/events/.
2. Na barra de navegação, selecione o Região da AWS a ser usado.
3. No painel de navegação, escolha Regras.
4. Escolha Create rule.
5. Para Nome, especifique um nome exclusivo para seu ambiente de computação. O nome pode conter até 64 caracteres. Pode conter letras minúsculas, maiúsculas, números, hifens e (-) e sublinhados (_).

   > **Note**
   >
   > Uma regra não pode ter o mesmo nome de outra regra no mesmo barramento de eventos Região da AWS e no mesmo.

6. (Opcional) Em Description, insira uma descrição para a regra.
7. Em Event bus, escolha o barramento de eventos que você deseja associar a essa regra. Se quiser que essa regra faça a correspondência com eventos provenientes da sua conta, selecione padrão. Quando um AWS service (Serviço da AWS) em sua conta emite um evento, ele sempre vai para o ônibus de eventos padrão da sua conta.
8. (Opcional) Desative a regra no barramento selecionado se não quiser executá-la imediatamente.
9. Em Rule type, escolha Schedule.
10. Escolha Continuar para criar a regra ou Avançar.

11. Em Schedule pattern, siga um destes procedimentos:

    - Escolha um cronograma refinado que seja executado em um horário específico, tal como 8:00 a.m. PST na primeira segunda-feira de cada mês e insira a expressão cron. Para obter mais informações, consulte Expressões Cron no Guia do EventBridge usuário da Amazon.
    - Escolha uma programação que seja executada em uma taxa regular, como a cada 10 minutos. e, em seguida, insira uma expressão rate.
12. Escolha Next (Próximo).
13. Em Target types (Tipos de destino), escolha AWS service (Serviço da AWS).
14. Em Selecionar um destino, escolha Fila de trabalhos em lote. Em seguida, configure o seguinte:

    - Job queue: (Fila de trabalhos:), insira Nome de recurso da Amazon (ARN) da fila de trabalhos na qual o trabalho será programado.
    - Job definition: (Definição do trabalho:) insira o nome e a revisão ou ARN completo da definição do trabalho a ser usado.
    - Job name: (Nome do trabalho:) insira um nome para o trabalho.
    - Array size: (Tamanho da matriz:) (opcional) insira um tamanho de matriz para que o trabalho execute mais de uma cópia. Para ter mais informações, consulte Trabalhos de matriz.
    - Job attempts: (opcional) insira o número de vezes para tentar novamente caso ocorra uma falha no trabalho. Para ter mais informações, consulte Repetições de trabalho automatizadas.
15. Para tipos de destino de fila de trabalhos em Batch, EventBridge precisa de permissão para enviar eventos para o destino. EventBridge pode criar a função do IAM necessária para que sua regra seja executada. Execute um destes procedimentos:

    - Para criar um perfil do IAM automaticamente, escolha Create a new role for this specific resource.
    - Para usar uma função do IAM que você criou anteriormente, escolha Use existing role (Usar função existente)
16. (Opcional) Expanda Additional settings.
17. Na seção Additional settings, para Configure target input, escolha Input Transformer.
18. Escolha Configure input transformer.
19. (Opcional) Para evento de amostra:

    a. Em Tipo de evento de amostra, escolha AWS eventos.

    b. Em Eventos de amostra, escolha Batch Job State Change.

20. Na seção Target input transformer, para Input path, especifique os valores a serem analisados no evento de acionamento. Por exemplo, para analisar o evento Batch Job State Change, use o seguinte formato JSON.

```
{
    "instance": "$.detail.jobId",
    "state": "$.detail.status"
}
```

21. Em Template, insira o seguinte:

```
{
    "instance": <jobId> ,
    "status": <status>
}
```

22. Selecione a opção Confirmar.
23. Em Idade máxima do evento, especifique o intervalo de tempo por quanto tempo os eventos não processados são mantidos.
24. Em Tentativas de repetição, insira o número de vezes que um evento é repetido.
25. Em Dead-letter queue, escolha uma opção de como os eventos não processados são tratados. Se necessário, especifique a fila do Amazon SQS para usar como fila de mensagens mortas.
26. (Opcional) Selecione Add another target (Adicionar outro destino) para adicionar outro destino a essa regra.
27. Escolha Next (Próximo).
28. (Opcional) Em Tags, escolha Adicionar nova tag para adicionar um rótulo de recurso. Para obter mais informações, consulte as EventBridge tags da Amazon no Guia EventBridge do usuário da Amazon.
29. Escolha Next (Próximo).
30. Para Revisar e criar, revise as etapas de configuração. Se precisar fazer alterações, escolha Edit (Editar). Quando terminar, escolha Create rule.

Tutorial: Receptando para AWS Batch EventBridge

Neste tutorial, você irá configurar uma AWS Lambda função simples para receptar AWS Batch eventos de trabalho e registrá-los em um fluxo de logs CloudWatch Logs.

Pré-requisitos

Este tutorial pressupõe um ambiente de computação e uma fila de de tarefas prontas para aceitar tarefas. Caso não possua um ambiente de computação e fila de tarefas em execução para capturar eventos, siga os passos em Começando com AWS Batch para criar um. Ao final deste tutorial, você poderá enviar opcionalmente uma tarefa para essa fila e testar se configurou a função do Lambda corretamente.

Etapa 1: Criar a Função do Lambda

Neste procedimento, você criará uma função do Lambda simples para funcionar como destino AWS Batch de mensagens do fluxo de eventos.

Para criar uma função do Lambda de destino

1. Abra o console AWS Lambda em https://console.aws.amazon.com/lambda/.
2. Escolha Criar Função e, em seguida, Criar do Início.
3. Para Nome da Função, insira batch-event-stream-handler.
4. ParaRuntime, escolha Python 3.8.
5. Escolha Criar Função.
6. Na seção Fonte do Código, edite o código de amostra de acordo com o exemplo a seguir:

```
import json


def lambda_handler(event, _context):
    # _context is not used
    del _context
    if event["source"] != "aws.batch":
        raise ValueError("Function only supports input from events with a source
 type of: aws.batch")

    print(json.dumps(event))
```

   Essa é uma função Python 3.8 simples, que imprime eventos enviados por AWS Batch. Se tudo estiver configurado corretamente, no final deste tutorial, os detalhes do evento aparecerão no fluxo de log do CloudWatch Logs associado a essa função do Lambda.
7. Escolha Implantar.

Etapa 2: Registrar Regra de Evento

Nesta seção, você criará uma regra de evento EventBridge que captura eventos de trabalho vindos dos seus recursos AWS Batch. Esta regra captura todos os eventos vindos de AWS Batch dentro da conta na qual estiver definida. As próprias mensagens de trabalho contêm informações sobre a origem do evento, inclusive fila de trabalhos onde foi inserido. Você pode usar essa informação para filtrar e classificar eventos de forma programática.

> **Note**
>
> Caso use o AWS Management Console para criar uma regra de evento, o console adicionará automaticamente as permissões do IAM ao EventBridge para chamar sua função do Lambda. No entanto, se criar uma regra de evento usando AWS CLI, você deverá conceder permissões explicitamente. Para mais informações, consulte Eventos e Padrões de Evento no Guia de Usuário Amazon EventBridge.

Para criar sua regra EventBridge

1. Abra o console Amazon EventBridge em https://console.aws.amazon.com/events/.
2. No painel de navegação, escolha Regras.
3. Escolha Criar Regra.
4. Insira um nome e uma descrição para a regra.

   Uma regra não pode ter o mesmo nome que outra na mesma Região e barramento de eventos.
5. Em Barramento de Eventos, escolha o barramento que deseja associar a essa regra. Se quiser que essa regra faça a correspondência com eventos provenientes da sua conta, selecione AWSBarramento de Eventos Padrão. Quando um serviço AWS em sua conta emite um evento, ele sempre irá para o barramento de eventos padrão da conta.
6. Em Tipo de Regra, escolha Regra com Padrão de Evento.
7. Escolha Próximo.
8. Em Origem do Evento, escolha Outro.
9. Em Padrão de Evento, selecione Padrões Personalizados (Editor JSON).
10. Cole o padrão de evento a seguir na área de texto.

```
{
```

```
  "source": [
    "aws.batch"
  ]
}
```

Essa regra se aplica a todos os grupos AWS Batch e a cada evento AWS Batch. Como alternativa, você pode criar uma regra mais específica para filtrar resultados.

11. Escolha Próximo.
12. Em Tipos de Destino, escolha AWS Serviço.
13. Em Selecionar um Destino, escolha Função do Lambda e selecione sua função.
14. (Opcional) Para Configurações Adicionais, proceda da seguinte forma:
    a. Em Tempo Máximo do Evento, insira um valor entre um minuto (00:01) e 24 horas (24:00).
    b. Em Tentativas de Repetição, insira um número entre 0 e 185.
    c. Em Fila de Mensagens não Entregues, escolha uma fila padrão Amazon SQS como fila de mensagens não entregues. O EventBridge enviará eventos que correspondam a essa regra para a fila de mensagens não entregues caso não sejam entregues com êxito ao destino. Faça um dos procedimentos a seguir:
       - Escolha Nenhum para não usar uma fila de mensagens não entregues.
       - Escolha Selecionar uma Fila Amazon SQS na Conta AWS Atual para usá-la como fila de mensagens não entregues e então, na lista suspensa, selecione a fila a ser usada.
       - Escolha Selecionar uma Fila Amazon SQS em qualquer outra AWS conta como fila de mensagens não entregues e insira o ARN da fila a ser usada. Você deve anexar uma política baseada em recurso à fila responsável por conceder permissão ao EventBridge para enviar mensagens. Para mais informações, consulte Concedendo Permissões à Fila de Mensagens Não Entregues do Guia de usuário Amazon EventBridge.
15. Escolha Próximo.
16. (Opcional) Insira uma ou mais tags para a regra. Para mais informações, consulte Tags Amazon EventBridge em Guia de Usuário Amazon EventBridge.
17. Escolha Próximo.
18. Analise os detalhes da regra e escolha Criar Regra.

Etapa 3: Testar sua Configuração

Agora, você pode testar sua configuração EventBridge enviando uma tarefa para a fila de tarefas. Se tudo for configurado corretamente, sua função do Lambda será acionada e registrará os dados de eventos em um fluxo de logs CloudWatch Logs para a função.

Para testar sua configuração

1. Abra o console AWS Batch em https://console.aws.amazon.com/batch/.
2. Insira uma nova AWS Batch tarefa. Para mais informações, consulte Enviando um trabalho.
3. Abra o console CloudWatch em https://console.aws.amazon.com/cloudwatch/.
4. No painel de navegação, escolha Logs e selecione o grupo de logs para sua função do Lambda (por exemplo, /aws/lambda/*my-function*).
5. Selecione um fluxo de log para visualizar os dados do evento.

Tutorial: Como enviar alertas do Amazon Simple Notification Service para eventos de trabalho que falharam

Neste tutorial, você configura uma regra de EventBridge evento que captura somente eventos de trabalho em que o trabalho foi movido para um `FAILED` status. No final deste tutorial, você pode optar por enviar um trabalho para essa fila de trabalhos. Isso serve para testar se você configurou seus alertas do Amazon SNS corretamente.

Pré-requisitos

Este tutorial pressupõe um ambiente de computação e uma fila de de tarefas prontas para aceitar tarefas. Caso você não tenha um ambiente de computação e fila de trabalho em execução do qual capturar eventos, siga as etapas em Começando com AWS Batch para criar um.

Etapa 1: Criar e se assinar um tópico do Amazon SNS

Neste tutorial, você configura um tópico do Amazon SNS para funcionar como um destino de evento para a nova regra de evento.

Para criar um tópico do Amazon SNS

1. Abra o console do Amazon SNS em https://console.aws.amazon.com/sns/v3/home.

2. Escolha Topics (Tópicos), Create topic (Criar tópico).
3. Em Tipo, escolha Padrão.
4. Em Nome, insira **`JobFailedAlert`** e selecione Criar tópico.
5. Na JobFailedAlerttela, escolha Criar assinatura.
6. Em Protocolo, escolha Email.
7. Em Endpoint, insira um endereço de e-mail ao qual tenha acesso e escolha Criar assinatura.
8. Verifique sua conta de e-mail e espere para receber uma mensagem de e-mail de confirmação de assinatura. Quando você recebê-la, escolha Confirmar assinatura.

Etapa 2: Registrar Regra de Evento

Em seguida, registre uma regra de evento que captura apenas eventos com falha de trabalho.

Para registrar sua EventBridge regra

1. Abra o EventBridge console da Amazon em https://console.aws.amazon.com/events/.
2. No painel de navegação, escolha Regras.
3. Selecione Criar regra.
4. Insira um nome e uma descrição para a regra.

   Uma regra não pode ter o mesmo nome que outra na mesma Região e barramento de eventos.
5. Em Barramento de eventos, selecione o barramento de eventos que você deseja associar a essa regra. Se quiser que essa regra faça a correspondência com eventos provenientes da sua conta, selecione Barramento de eventos padrão da AWS . Quando um AWS serviço em sua conta emite um evento, ele sempre vai para o barramento de eventos padrão da sua conta.
6. Em Rule type (Tipo de regra), selecione Rule with an event pattern (Regra com um padrão de evento).
7. Escolha Próximo.
8. Em Origem do Evento, escolha Outro.
9. Em Padrão de Evento, selecione Padrões Personalizados (Editor JSON).
10. Cole o padrão de evento a seguir na área de texto.

```
{
```

```
  "detail-type": [
    "Batch Job State Change"
  ],
  "source": [
    "aws.batch"
  ],
  "detail": {
    "status": [
      "FAILED"
    ]
  }
}
```

Esse código define uma EventBridge regra que corresponde a qualquer evento em que o status do trabalho estejaFAILED. Para obter mais informações sobre padrões de eventos, consulte Eventos e padrões de eventos no Guia EventBridge do usuário da Amazon.

11. Selecione Next (Próximo).
12. Em Tipos de destino, escolha Serviço da AWS .
13. Em Selecionar um destino, escolha Tópico SNS e, em Tópico, escolha JobFailedAlert.
14. (Opcional) Para Configurações Adicionais, proceda da seguinte forma:

    a. Em Tempo Máximo do Evento, insira um valor entre um minuto (00:01) e 24 horas (24:00).

    b. Em Tentativas de Repetição, insira um número entre 0 e 185.

    c. Para fila de mensagens mortas, escolha se deseja usar uma fila padrão do Amazon SQS como fila de mensagens mortas. EventBridge envia eventos que correspondam a essa regra para a fila de mensagens mortas se não forem entregues com sucesso ao destino. Faça um dos procedimentos a seguir:

       - Escolha None (Nenhum) para não usar uma fila de mensagens não entregues.
       - Escolha Selecionar uma fila do Amazon SQS na AWS conta atual para usar como fila de mensagens mortas e, em seguida, selecione a fila a ser usada no menu suspenso.
       - Escolha Selecionar uma fila do Amazon SQS em outra AWS conta como uma fila de mensagens mortas e, em seguida, insira o ARN da fila a ser usada. Você deve anexar uma política baseada em recursos à fila que conceda EventBridge permissão para enviar mensagens para ela. Para obter mais informações, consulte Conceder permissões para a fila de mensagens mortas no Guia do usuário da Amazon. EventBridge

15. Escolha Próximo.

16. (Opcional) Insira uma ou mais tags para a regra. Para obter mais informações, consulte as EventBridge tags da Amazon no Guia EventBridge do usuário da Amazon.
17. Selecione Next (Próximo).
18. Analise os detalhes da regra e selecione Criar regra.

Etapa 3: Testar a regra

Para testar sua regra, envie um trabalho que saia logo após começar com um código de saída diferente de zero. Caso a regra de evento esteja configurada corretamente, você deverá receber uma mensagem de e-mail em alguns minutos com o texto do evento.

Para testar uma regra

1. Abra o AWS Batch console em https://console.aws.amazon.com/batch/.
2. Envie um novo AWS Batch emprego. Para ter mais informações, consulte Enviando um trabalho. Para o comando do trabalho, substitua este comando para sair do contêiner com um código de saída 1.

   ```
   /bin/sh, -c, 'exit 1'
   ```
3. Verifique seu e-mail para confirmar que você recebeu um e-mail de alerta para a notificação de trabalho com falha.

Regra alternativa: Batch Job Queue Blocked

Para criar uma regra de evento que monitore o Batch Job Queue Blocked, repita as etapas deste tutorial com as seguintes alterações:

1. Na Etapa 1, use *BlockedJobQueue*como nome do tópico.
2. Na Etapa 2, use o seguinte padrão no editor JSON:

```
{
   "detail-type": [
     "Batch Job Queue Blocked"
   ],
   "source": [
     "aws.batch"
   ]
```

```
}
```

Como usar o CloudWatch Logs com AWS Batch

Você pode configurar seus trabalhos AWS Batch nos recursos do EC2 para enviar informações e métricas de log detalhadas para o CloudWatch Logs. Ao fazer isso, você pode exibir logs diferentes das tarefas em um local conveniente. Para obter mais informações sobre o CloudWatch, consulte What is Amazon CloudWatch Logs? no Amazon CloudWatch User Guide.

> **Note**
>
> Por padrão, o CloudWatch Logs está ativado para contêiner do Fargate AWS.

Para ativar e personalizar o registro do CloudWatch Logs, revise as seguintes tarefas de configuração únicas:

- Para ambientes de computação AWS Batch baseados em recursos do EC2, adicione uma política do IAM ao perfil `ecsInstanceRole`. Para obter mais informações, consulte the section called “Adicionar uma política do IAM do CloudWatch Logs”.
- Crie um modelo de lançamento do Amazon EC2 que inclua monitoramento detalhado do CloudWatch e, em seguida, especifique o modelo ao criar seu ambiente de computação AWS Batch. Você também pode instalar o agente do CloudWatch em uma imagem existente e depois especificar a imagem no assistente de primeira execução AWS Batch.
- (Opcional) Configure o driver awslogs. Você pode adicionar parâmetros que alteram o comportamento padrão nos recursos do EC2 e do Fargate. Para obter mais informações, consulte the section called “Usar o driver de log awslogs”.

Adicionar uma política do IAM do CloudWatch Logs

Para que os trabalhos possam enviar dados de log e métricas detalhadas ao CloudWatch Logs, você deve criar uma política do IAM que use as APIs do CloudWatch Logs. Após criar a política, anexe a política a um perfil `ecsInstanceRole`.

> **Note**
>
> Se a política `ECS-CloudWatchLogs` não estiver vinculada ao perfil `ecsInstanceRole`, as métricas básicas ainda poderão ser enviadas para o CloudWatch Logs. No entanto, as

métricas básicas não incluem dados de registro ou métricas detalhadas, como espaço livre em disco.

Ambientes de computação AWS Batch usam recursos do Amazon EC2. Quando você cria um ambiente de computação usando o assistente de primeira execução AWS Batch, AWS Batch cria o perfil `ecsInstanceRole` e configura o ambiente com ela.

Se você não estiver usando o assistente de primeira execução, poderá especificar o perfil `ecsInstanceRole` ao criar um ambiente de computação na API AWS Command Line Interface ou AWS Batch. Para obter mais informações, consulte AWS CLI Command Reference ou AWS Batch API Reference.

Para criar a política de IAM **`ECS-CloudWatchLogs`**

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Policies.
3. Escolha Create policy.
4. Escolha JSON e insira a seguinte política:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": [
                "arn:aws:logs:*:*:*"
            ]
        }
    ]
}
```

5. Escolha Next: Tags.
6. (Opcional) Em Adicionar tags, escolha Adicionar tag para adicionar uma tag à política.

7. Escolha Next: Review.
8. Na página Review policy, em Name, digite **`ECS-CloudWatchLogs`**, e, em seguida, digite uma Description opcional.
9. Escolha Create policy.

Para anexar a política **`ECS-CloudWatchLogs`** a **`ecsInstanceRole`**

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Roles.
3. Selecione `ecsInstanceRole`. Se o perfil não existir, siga os procedimentos em Perfil de instância do Amazon ECS para criar o perfil.
4. Escolha Adicionar permissões e depois Anexar políticas.
5. Selecione a política ECS-CloudWatchLogs e escolha Attach policy.

Baixar e configurar o atendente do CloudWatch

Você pode criar um modelo de inicialização do Amazon EC2 que inclua monitoração do CloudWatch. Para obter mais informações, consulte Launch an instance from a launch template e Advanced details em Amazon EC2 User Guide for Linux Instances.

Você também pode instalar o agente do CloudWatch em uma AMI existente do Amazon EC2 e depois especificar a imagem no assistente de primeira execução AWS Batch. Para obter mais informações, consulte Instalação do agente CloudWatch e Introdução ao AWS Batch.

> **Note**
>
> Os modelos de lançamento não são compatíveis com os AWS Fargate recursos.

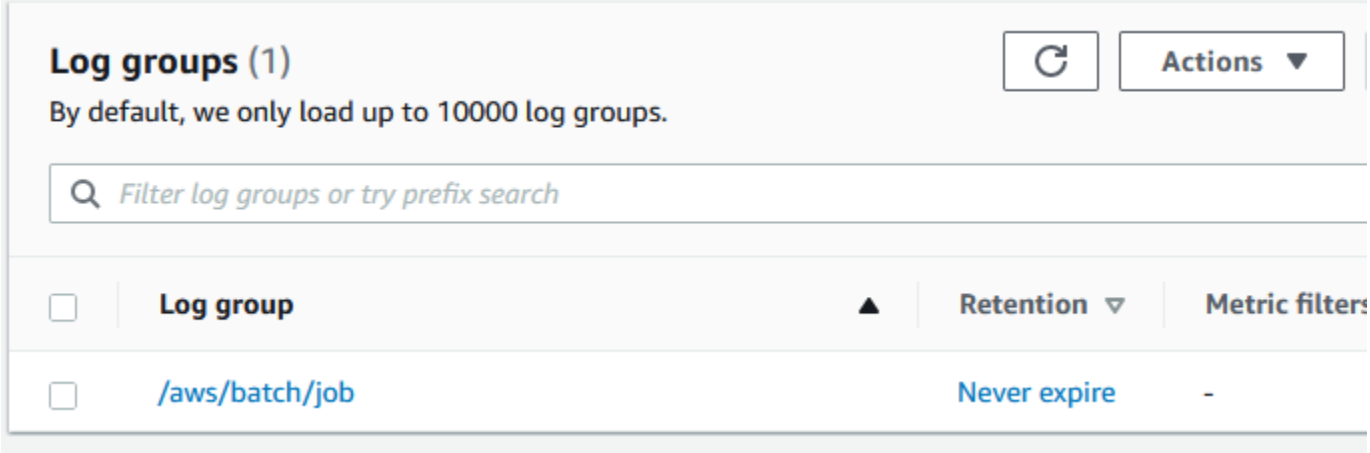
Visualizar CloudWatch Logs

Você pode visualizar e pesquisar os registros do CloudWatch Logs no AWS Management Console.

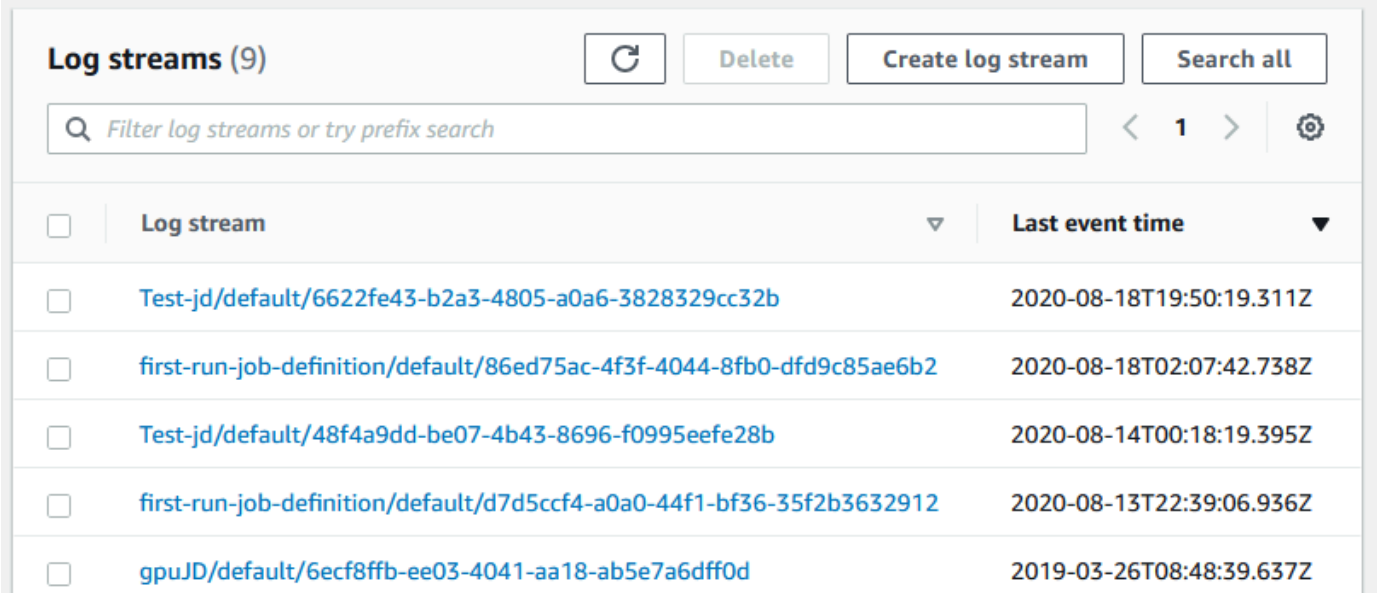
> **Note**
>
> Pode levar alguns minutos para que os dados sejam exibidos no CloudWatch Logs.

Para visualizar os dados do CloudWatch Logs

1. Abra o console do CloudWatch em https://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação à esquerda, escolha Logs e, em seguida, escolha Log groups (Grupos de log).

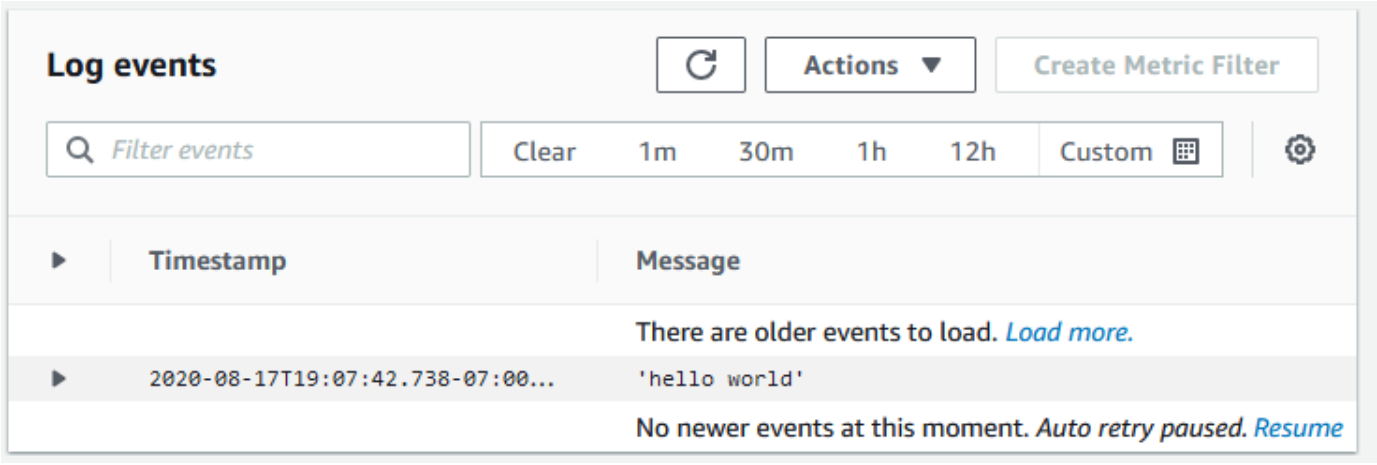


3. Escolha um grupo de logs para visualizar.



4. Escolha um stream de logs para visualizar. Por padrão, os fluxos são identificados pelos primeiros 200 caracteres do nome do trabalho e pelo ID da tarefa do Amazon ECS.

> **Tip**
>
> Para baixar os dados do fluxo de logs, escolha Ações.


Log events
Actions
Create Metric Filter
Filter events
Clear
1m
30m
1h
12h
Custom
Timestamp
Message
There are older events to load. Load more.
2020-08-17T19:07:42.738-07:00...
'hello world'
No newer events at this moment. Auto retry paused. Resume

Use o CloudWatch Logs para monitorar AWS Batch trabalhos do Amazon EKS

Você pode usar o Amazon CloudWatch Logs para monitorar, armazenar e visualizar todos os seus arquivos de log em um único local. Usando o CloudWatch Logs, você pode pesquisar, filtrar e analisar dados de log de várias fontes.

Você pode baixar uma AWS para imagem Fluent Bit que inclui um plug-in para monitorar AWS Batch em trabalhos do Amazon EKS no CloudWatch Logs. Fluent Bit é um processador e encaminhador de registros de código aberto que é compatível com Docker e Kubernetes. Recomendamos que você use Fluent Bit como roteador de log porque ele consome menos recursos do que Fluentd. Para obter mais informações, consulte Using the AWS for Fluent Bit image.

Pré-requisitos

Anexe uma política `CloudWatchAgentServerPolicy` à política AWS Identity and Access Management de seus nós de processamento. Para obter mais informações, consulte Verify prerequisites.

Instalar AWS para Fluent Bit

Para obter instruções sobre como instalar AWS para Fluent Bit e criar os grupos do CloudWatch, consulte Setting up Fluent Bit or Quick Start with the CloudWatch agent and Fluent Bit.

> **Tip**
>
> Lembre-se de que Fluent Bit usa 0.5 de CPU e 100 MB de memória nos nós AWS Batch. Isso reduz a capacidade total disponível para trabalhos AWS Batch. Considere isso ao dimensionar seus trabalhos.

Ativar o Fluent Bit para nós AWS Batch

Para garantir que o DaemonSet de registro Fluent Bit seja executado em nós gerenciados AWS Batch, modifique as tolerações do DaemonSet Fluent Bit:

```
tolerations:
- key: "batch.amazonaws.com/batch-node"
  operator: "Exists"
```

AWS Batch CloudWatch Container Insights

O CloudWatch Container Insights coleta, agrega e resume métricas e logs dos seus AWS Batch ambientes de computação e tarefas. As métricas incluem uso de CPU, memória, disco e rede. Você pode adicionar essas métricas aos painéis do CloudWatch.

Os dados operacionais são coletados como eventos de log de desempenho. Essas entradas usam um esquema JSON estruturado que permite que dados de alta cardinalidade sejam ingeridos e armazenados em escala. Com base nesses dados, o CloudWatch cria métricas agregadas de alto nível no ambiente de computação e nível de tarefa como métricas do CloudWatch. Para obter mais informações, consulte Logs Estruturados do Container Insights para Amazon ECS no Guia de Usuário do Amazon CloudWatch.

> **Important**
>
> O CloudWatch Container Insights é cobrado como “métricas personalizadas” pelo CloudWatch. Para mais informações, consulte Precificação do Amazon CloudWatch Events

Ativar o Container Insights

Você pode ativar o Container Insights para AWS Batch ambientes de computação.

1. Abra o AWS Batch Console.
2. Escolha Ambientes de Computação.
3. Escolha o ambiente de computação que deseja.
4. Para Container Insights, ative o Container Insights para o ambiente de computação.

> **Tip**
>
> Você pode selecionar um intervalo padrão para agregar as métricas ou criar um intervalo personalizado.

Por padrão, as seguintes métricas serão exibidas. Para obter uma lista completa de métricas Amazon ECS Container Insights, consulte Métricas Amazon ECS Container Insights no Guia de Usuário do Amazon CloudWatch.

- **`JobCount`** – Número de tarefas executadas no ambiente de computação.
- **`ContainerInstanceCount`** – Número de instâncias do Amazon Elastic Compute Cloud que executam o atendente do Amazon ECS e estão registradas no ambiente de computação.
- **`MemoryReserved`** – Memória reservada pelas tarefas do ambiente de computação. Essa métrica é coletada apenas para tarefas com uma reserva de memória definida na definição da tarefa.
- **`MemoryUtilized`** – Memória sendo usada por tarefas do ambiente de computação. Essa métrica é coletada apenas para tarefas com reserva de memória definida na definição da tarefa.
- **`CpuReserved`** – Unidades de CPU reservadas pelas tarefas do ambiente de computação. Essa métrica é coletada apenas para tarefas com reserva de CPU definida na definição de tarefa.
- **`CpuUtilized`** – Unidades de CPU usadas por tarefas no ambiente de computação. Essa métrica é coletada apenas para tarefas com reserva de CPU definida na definição de tarefa.
- **`NetworkRxBytes`** - Número de bytes recebidos. Essa métrica está disponível apenas para contêineres em tarefas que usem modos de `awsvpc` ou de rede de ponte.
- **`NetworkTxBytes`** – Número de bytes transmitidos. Essa métrica está disponível apenas para contêineres em tarefas com modos de `awsvpc` ou de rede de ponte.
- **`StorageReadBytes`** – Número de bytes lidos do armazenamento.
- **`StorageWriteBytes`** – Número de bytes gravados no armazenamento.

Registrando AWS Batch log de chamadas API com AWS CloudTrail

O AWS Batch é integrado a AWS CloudTrail, serviço que fornece um registro das ações realizadas por um usuário, perfil ou AWS serviço em AWS Batch. O CloudTrail captura todas as chamadas API para AWS Batch como eventos. As chamadas capturadas incluem as aquelas do AWS Batch console e chamadas de código para AWS Batch operações API. Caso crie uma trilha, voce pode habilitar a entrega contínua de eventos CloudTrail para um bucket Amazon S3, inclusive eventos para AWS Batch. Mesmo que não configure uma trilha, você ainda pode visualizar os eventos mais recentes no console CloudTrail em Histórico de Eventos. Ao fazer uso das informações coletadas pelo CloudTrail, é possível determinar a solicitação feita a AWS Batch, o endereço IP no qual a solicitação foi feita, quem fez a solicitação e quando foi feita, além de detalhes adicionais.

Para saber mais sobre o CloudTrail, consulte o AWS CloudTrail Guia de Usuário.

AWS Batch Informações em CloudTrail

O CloudTrail é habilitado em sua AWS conta ao criá-la. Quando uma atividade ocorre em AWS Batch, a mesma é registrada em um evento CloudTrail junto a outros AWS eventos de serviço em Histórico de Evento. Você pode visualizar, pesquisar e baixar eventos recentes em sua AWS conta. Para mais informações, consulte Visualizando Eventos com Histórico de Eventos CloudTrail.

Para um registro contínuo de eventos em sua AWS conta, inclusive eventos para AWS Batch, crie uma trilha. Uma trilha permite que o CloudTrail entregue arquivos de log a um bucket Amazon S3. Por padrão, ao criar uma trilha no console, a mesma é aplicada a todas as AWSRegiões. A trilha registra logs de eventos de todas as Regiões na AWS divisória e entrega os arquivos do log para o bucket Amazon S3 especificado. Além disso, é possível configurar outros AWS serviços para melhor analisar e agir de acordo com dados coletados do evento nos logs CloudTrail. Para mais informações, consulte:

- Visão Geral para Criar uma Trilha
- Serviços e Integrações Compatíveis com CloudTrail
- Configurando Notificações Amazon SNS para CloudTrail
- Recebendo Arquivos de Log CloudTrail de Várias Regiões e Recebendo Arquivos de Log CloudTrail de Várias Contas

Todas AWS Batch as ações são registradas pelo CloudTrail e documentadas em https://docs.aws.amazon.com/batch/latest/APIReference/. Por exemplo, chamadas para as seções `SubmitJob`, `ListJobs` e `DescribeJobs` geram entradas nos arquivos de log CloudTrail.

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

- Se a solicitação foi feita com credenciais de usuário raiz ou usuário do IAM.
- Se a solicitação foi feita com credenciais de segurança temporárias para um perfil ou usuário federado.
- Se a solicitação foi feita por outro AWS serviço.

Para mais informações, consulte Elemento userIdentity CloudTrail.

Entendendo AWS Batch Entradas de Arquivo de Log

Uma trilha é uma configuração que permite a entrega de eventos como arquivos de log a um bucket Amazon S3 especificado. Os arquivos de log CloudTrail contêm uma ou mais entradas de log. Um evento representa uma única solicitação de qualquer fonte, e inclui informações sobre a ação solicitada, data e hora da ação, parâmetros de solicitação e assim por diante. Os arquivos de log CloudTrail não são um rastreamento de pilha ordenada de chamadas API públicas, portanto não são exibidos em qualquer ordem específica.

O exemplo a seguir mostra uma entrada de log CloudTrail que demonstra a `CreateComputeEnvironment` ação.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE:admin",
    "arn": "arn:aws:sts::012345678910:assumed-role/Admin/admin",
    "accountId": "012345678910",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2017-12-20T00:48:46Z"
      },
      "sessionIssuer": {
```

```
        "type": "Role",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::012345678910:role/Admin",
        "accountId": "012345678910",
        "userName": "Admin"
      }
    }
  },
  "eventTime": "2017-12-20T00:48:46Z",
  "eventSource": "batch.amazonaws.com",
  "eventName": "CreateComputeEnvironment",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.1",
  "userAgent": "aws-cli/1.11.167 Python/2.7.10 Darwin/16.7.0 botocore/1.7.25",
  "requestParameters": {
    "computeResources": {
      "subnets": [
        "subnet-5eda8e04"
      ],
      "tags": {
        "testBatchTags": "CLI testing CE"
      },
      "desiredvCpus": 0,
      "minvCpus": 0,
      "instanceTypes": [
        "optimal"
      ],
      "securityGroupIds": [
        "sg-aba9e8db"
      ],
      "instanceRole": "ecsInstanceRole",
      "maxvCpus": 128,
      "type": "EC2"
    },
    "state": "ENABLED",
    "type": "MANAGED",
    "computeEnvironmentName": "Test"
  },
  "responseElements": {
    "computeEnvironmentName": "Test",
    "computeEnvironmentArn": "arn:aws:batch:us-east-1:012345678910:compute-environment/
Test"
  },
  "requestID": "890b8639-e51f-11e7-b038-EXAMPLE",
```

```
  "eventID": "874f89fa-70fc-4798-bc00-EXAMPLE",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "012345678910"
}
```

Criando uma Nuvem Privada Virtual

Os recursos de computação dos seus ambientes de computação demandam acesso a rede externa para se comunicarem com AWS Batch e com endpoints de serviço Amazon ECS. No entanto, você pode ter tarefas que deseje executar em sub redes privadas. A criação de uma VPC com sub redes públicas e privadas concede flexibilidade para executar tarefas em uma sub rede pública ou privada.

Você pode usar a Amazon Virtual Private Cloud (Amazon VPC) para iniciar AWS recursos em uma rede virtual de sua própria definição. Este tópico fornece um link para o assistente Amazon VPC e uma lista de opções para seleção.

Crie uma VPC

Para informações sobre como criar uma Amazon VPC, consulte Criar uma Única VPC no Guia de Usuário Amazon VPC e use a tabela a seguir para determinar quais opções selecionar.

Opção	Valor
Recursos a criar	Somente VPC
Nome	Forneça um nome opcional para a sua VPC.
Bloco IPv4 CIDR	Entrada manual IPv4 CIDR O tamanho do bloco CIDR deve ter um tamanho entre /16 e /28.
Bloco IPv6 CIDR	Sem bloco IPv6 CIDR
Locação	Padrão

Para saber mais sobre a Amazon VPC, consulte O Que é Amazon VPC? no Guia de usuário Amazon VPC.

Próximos Passos

Depois de criar sua VPC, considere os seguintes próximos passos:

- Crie grupos de segurança para seus recursos públicos e privados, caso precisem de acesso à rede de entrada. Para mais informações, consulte Trabalhe com Grupos de Segurança no Guia de Usuário Amazon VPC.
- Crie um AWS Batch ambiente de computação gerenciado que inicie recursos de computação em sua nova VPC. Para mais informações, consulte Criação de um ambiente de computação. Caso use o assistente de criação de ambiente de computação no AWS Batch console, você poderá especificar a VPC recém-criada e as sub redes públicas ou privadas nas quais desejar lançar suas instâncias.
- Crie uma AWS Batch fila de tarefas mapeada para o seu novo ambiente de computação. Para mais informações, consulte Como criar uma fila de tarefas.
- Crie uma definição de tarefa para executar seus trabalhos. Para mais informações, consulte Como criar uma definição de tarefa de nó único .
- Insira uma tarefa com sua definição de tarefa para a nova fila de tarefas. Este trabalho será posicionado no ambiente de computação criado com VPC e sub redes novas. Para mais informações, consulte Enviando um trabalho.

Segurança em AWS Batch

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

A segurança é uma responsabilidade compartilhada entre você AWS e você. O Modelo de Responsabilidade Compartilhada descreve isso como segurança da nuvem e segurança na nuvem.

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços na AWS nuvem. AWS também fornece serviços que você pode usar com segurança. Auditores terceirizados testam e verificam regularmente a eficácia de nossa segurança como parte dos Programas de Conformidade Programas de AWS de . Para saber mais sobre os programas de conformidade que se aplicam AWS Batch, consulte AWS Serviços no escopo do programa de conformidade AWS .
- Segurança na nuvem - sua responsabilidade é determinada pelo serviço AWS que você usa. Você também é responsável por outros fatores, inclusive a sensibilidade de seus dados, os requisitos da sua empresa, leis e regulamentos aplicáveis.

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

Tópicos

- Identity and Access Management para AWS Batch
- Acesso AWS Batch usando um endpoint de interface
- Validação de conformidade para AWS Batch
- Segurança de infraestrutura em AWS Batch

Identity and Access Management para AWS Batch

AWS Identity and Access Management (IAM) é uma ferramenta AWS service (Serviço da AWS) que ajuda o administrador a controlar com segurança o acesso aos AWS recursos. Os administradores

do IAM controlam quem pode ser autenticado (conectado) e autorizado (tem permissões) a usar AWS Batch os recursos. O IAM é um AWS service (Serviço da AWS) que você pode usar sem custo adicional.

Tópicos

Público

A forma como você usa AWS Identity and Access Management (IAM) difere, dependendo do trabalho que você faz AWS Batch.

Usuário do serviço — Se você usar o AWS Batch serviço para realizar seu trabalho, seu administrador fornecerá as credenciais e as permissões de que você precisa. À medida que você usa mais AWS Batch recursos para fazer seu trabalho, talvez precise de permissões adicionais. Entender como o acesso é gerenciado pode ajudá-lo a solicitar as permissões corretas ao seu administrador. Se não for possível acessar um recurso no AWS Batch, consulte Solução de problemas AWS Batch de identidade e acesso.

Administrador de serviços — Se você é responsável pelos AWS Batch recursos da sua empresa, provavelmente tem acesso total AWS Batch a. É seu trabalho determinar quais AWS Batch recursos e recursos seus usuários do serviço devem acessar. Assim, você deve enviar solicitações ao administrador do IAM para alterar as permissões dos usuários de seu serviço. Revise as informações nesta página para entender os Introdução ao IAM. Para saber mais sobre como sua empresa pode usar o IAM com AWS Batch, consulteComo AWS Batch funciona com o IAM.

Administrador do IAM:Se você for um administrador do IAM, talvez queira saber detalhes sobre como pode gravar políticas para gerenciar o acesso ao AWS Batch. Para ver exemplos de políticas AWS Batch baseadas em identidade que você pode usar no IAM, consulte. Exemplos de políticas baseadas em identidade para AWS Batch

Autenticando com identidades

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

Você pode entrar AWS como uma identidade federada usando credenciais fornecidas por meio de uma fonte de identidade. AWS IAM Identity Center Usuários (IAM Identity Center), a autenticação de login único da sua empresa e suas credenciais do Google ou do Facebook são exemplos de identidades federadas. Quando você faz login como uma identidade federada, o administrador já configurou anteriormente a federação de identidades usando perfis do IAM. Ao acessar AWS usando a federação, você está assumindo indiretamente uma função.

Dependendo do tipo de usuário que você é, você pode entrar no AWS Management Console ou no portal de AWS acesso. Para obter mais informações sobre como fazer login em AWS, consulte Como fazer login Conta da AWS no Guia do Início de Sessão da AWS usuário.

Se você acessar AWS programaticamente, AWS fornece um kit de desenvolvimento de software (SDK) e uma interface de linha de comando (CLI) para assinar criptograficamente suas solicitações usando suas credenciais. Se você não usa AWS ferramentas, você mesmo deve assinar as solicitações. Para obter mais informações sobre como usar o método recomendado para assinar solicitações por conta própria, consulte Assinatura de solicitações de AWS API no Guia do usuário do IAM.

Independentemente do método de autenticação usado, também pode ser exigido que você forneça informações adicionais de segurança. Por exemplo, AWS recomenda que você use a autenticação multifator (MFA) para aumentar a segurança da sua conta. Para saber mais, consulte Autenticação multifator no Guia AWS IAM Identity Center do usuário. Usar a autenticação multifator (MFA) na AWS no Guia do usuário do IAM.

Conta da AWS usuário root

Ao criar uma Conta da AWS, você começa com uma identidade de login que tem acesso completo a todos Serviços da AWS os recursos da conta. Essa identidade é chamada de usuário Conta da AWS

raiz e é acessada fazendo login com o endereço de e-mail e a senha que você usou para criar a conta. É altamente recomendável não usar o usuário raiz para tarefas diárias. Proteja as credenciais do usuário raiz e use-as para executar as tarefas que somente ele pode executar. Para obter a lista completa das tarefas que exigem login como usuário raiz, consulte Tarefas que exigem credenciais de usuário raiz no Guia do usuário do IAM.

Identidade federada

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

Uma identidade federada é um usuário do seu diretório de usuários corporativo, de um provedor de identidade da web AWS Directory Service, do diretório do Identity Center ou de qualquer usuário que acesse usando credenciais fornecidas Serviços da AWS por meio de uma fonte de identidade. Quando as identidades federadas acessam Contas da AWS, elas assumem funções, e as funções fornecem credenciais temporárias.

Para o gerenciamento de acesso centralizado, recomendamos usar o AWS IAM Identity Center. Você pode criar usuários e grupos no IAM Identity Center ou pode se conectar e sincronizar com um conjunto de usuários e grupos em sua própria fonte de identidade para uso em todos os seus Contas da AWS aplicativos. Para mais informações sobre o Centro de Identidade do IAM, consulte “O que é o Centro de Identidade do IAM?” no Guia do usuário do AWS IAM Identity Center .

Grupos e usuários do IAM

Um usuário do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas para uma única pessoa ou aplicativo. Sempre que possível, recomendamos depender de credenciais temporárias em vez de criar usuários do IAM com credenciais de longo prazo, como senhas e chaves de acesso. No entanto, se você tiver casos de uso específicos que exijam credenciais de longo prazo com usuários do IAM, recomendamos alternar as chaves de acesso. Para mais informações, consulte Alterne as chaves de acesso regularmente para casos de uso que exijam credenciais de longo prazo no Guia do usuário do IAM.

Um grupo do IAM é uma identidade que especifica uma coleção de usuários do IAM. Não é possível fazer login como um grupo. É possível usar grupos para especificar permissões para vários usuários de uma vez. Os grupos facilitam o gerenciamento de permissões para grandes conjuntos de usuários. Por exemplo, você pode ter um grupo chamado IAMAdmins e atribuir a esse grupo permissões para administrar recursos do IAM.

Usuários são diferentes de perfis.. Um usuário é exclusivamente associado a uma pessoa ou a uma aplicação, mas um perfil pode ser assumido por qualquer pessoa que precisar dele. Os usuários têm credenciais permanentes de longo prazo, mas os perfis fornecem credenciais temporárias. Para saber mais, consulte Quando criar um usuário do IAM (em vez de um perfil) no Guia do usuário do IAM.

Perfis do IAM

Uma função do IAM é uma identidade dentro da sua Conta da AWS que tem permissões específicas. Ele é semelhante a um usuário do IAM, mas não está associado a uma pessoa específica. Você pode assumir temporariamente uma função do IAM no AWS Management Console trocando de funções. Você pode assumir uma função chamando uma operação de AWS API AWS CLI ou usando uma URL personalizada. Para mais informações sobre métodos para o uso de perfis, consulte Usar perfis do IAM no Guia do usuário do IAM.

Perfis do IAM com credenciais temporárias são úteis nas seguintes situações:

- Acesso de usuário federado: para atribuir permissões a identidades federadas, você pode criar um perfil e definir permissões para ele. Quando uma identidade federada é autenticada, essa identidade é associada ao perfil e recebe as permissões definidas pelo mesmo. Para mais informações sobre perfis para federação, consulte Criar um perfil para um provedor de identidades de terceiros no Guia do usuário do IAM. Se você usar o IAM Identity Center, configure um conjunto de permissões. Para controlar o que suas identidades podem acessar após a autenticação, o IAM Identity Center correlaciona o conjunto de permissões a um perfil no IAM. Para obter informações sobre conjuntos de permissões, consulte Conjuntos de permissões no Guia do Usuário do AWS IAM Identity Center .
- Permissões temporárias para usuários do IAM: um usuário ou um perfil do IAM pode assumir um perfil do IAM para obter temporariamente permissões diferentes para uma tarefa específica.
- Acesso entre contas: é possível usar um perfil do IAM para permitir que alguém (uma entidade principal confiável) em outra conta acesse recursos em sua conta. Os perfis são a principal forma de conceder acesso entre contas. No entanto, com alguns Serviços da AWS, você pode anexar uma política diretamente a um recurso (em vez de usar uma função como proxy). Para saber a diferença entre funções e políticas baseadas em recurso para acesso entre contas, consulte Como os perfis do IAM diferem das políticas baseadas em recurso no Guia do usuário do IAM.
- Acesso entre serviços — Alguns Serviços da AWS usam recursos em outros Serviços da AWS. Por exemplo, quando você faz uma chamada em um serviço, é comum que esse serviço execute aplicações no Amazon EC2 ou armazene objetos no Amazon S3. Um serviço pode fazer isso

usando as permissões do principal de chamada, usando um perfil de serviço ou um perfil vinculado ao serviço.

- Sessões de acesso direto (FAS) — Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.
- Perfil de serviço: um perfil de serviço é um perfil do IAM que um serviço assume para realizar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um no Guia do usuário do IAM.
- Função vinculada ao serviço — Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode assumir o perfil para executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para perfis vinculados ao serviço.

- Aplicativos em execução no Amazon EC2 — Você pode usar uma função do IAM para gerenciar credenciais temporárias para aplicativos que estão sendo executados em uma instância do EC2 e fazendo AWS CLI solicitações de API. AWS É preferível fazer isso a armazenar chaves de acesso na instância do EC2. Para atribuir uma AWS função a uma instância do EC2 e disponibilizá-la para todos os seus aplicativos, você cria um perfil de instância anexado à instância. Um perfil de instância contém o perfil e permite que os programas em execução na instância do EC2 obtenham credenciais temporárias. Para obter mais informações, consulte Usar uma função do IAM para conceder permissões a aplicações em execução nas instâncias do Amazon EC2 no Guia do usuário do IAM.

Para saber se deseja usar os perfis do IAM, consulte Quando criar um perfil do IAM (em vez de um usuário) no Guia do usuário do IAM.

Gerenciamento do acesso usando políticas

Você controla o acesso AWS criando políticas e anexando-as a AWS identidades ou recursos. Uma política é um objeto AWS que, quando associada a uma identidade ou recurso, define suas permissões. AWS avalia essas políticas quando um principal (usuário, usuário raiz ou sessão de função) faz uma solicitação. As permissões nas políticas determinam se a solicitação será permitida ou negada. A maioria das políticas é armazenada AWS como documentos JSON. Para mais informações sobre a estrutura e o conteúdo de documentos de políticas JSON, consulte Visão geral das políticas JSON no Guia do usuário do IAM.

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

Por padrão, usuários e funções não têm permissões. Para conceder aos usuários permissão para executar ações nos recursos de que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM a perfis, e os usuários podem assumir os perfis.

As políticas do IAM definem permissões para uma ação, independentemente do método usado para executar a operação. Por exemplo, suponha que você tenha uma política que permite a ação `iam:GetRole`. Um usuário com essa política pode obter informações de função da AWS Management Console AWS CLI, da ou da AWS API.

Políticas baseadas em identidade

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário, grupo de usuários ou perfil do IAM. Essas políticas controlam quais ações os usuários e funções podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte Criação de política do IAM no Guia do usuário do IAM.

As políticas baseadas em identidade podem ser categorizadas ainda mais como políticas em linha ou políticas gerenciadas. As políticas em linha são anexadas diretamente a um único usuário, grupo ou perfil. As políticas gerenciadas são políticas autônomas que você pode associar a vários usuários, grupos e funções em seu Conta da AWS. As políticas AWS gerenciadas incluem políticas gerenciadas e políticas gerenciadas pelo cliente. Para saber como escolher entre uma política gerenciada ou uma política em linha, consulte Escolher entre políticas gerenciadas e políticas em linha no Guia do usuário do IAM.

Políticas baseadas em recursos

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

Políticas baseadas em recursos são políticas em linha que estão localizadas nesse serviço. Você não pode usar políticas AWS gerenciadas do IAM em uma política baseada em recursos.

Listas de controle de acesso (ACLs)

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

O Amazon S3 e o Amazon VPC são exemplos de serviços que oferecem suporte a ACLs. AWS WAF Para saber mais sobre ACLs, consulte Visão geral da lista de controle de acesso (ACL) no Guia do desenvolvedor do Amazon Simple Storage Service.

Outros tipos de política

AWS oferece suporte a tipos de políticas adicionais menos comuns. Esses tipos de política podem definir o máximo de permissões concedidas a você pelos tipos de política mais comuns.

- Limites de permissões: um limite de permissões é um atributo avançado no qual você define o máximo de permissões que uma política baseada em identidade pode conceder a uma entidade do IAM (usuário ou perfil do IAM). É possível definir um limite de permissões para uma entidade. As permissões resultantes são a interseção das políticas baseadas em identidade de uma entidade e dos seus limites de permissões. As políticas baseadas em recurso que especificam o usuário ou a função no campo `Principal` não são limitadas pelo limite de permissões. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para mais informações sobre limites de permissões, consulte Limites de permissões para identidades do IAM no Guia do usuário do IAM.

- Políticas de controle de serviço (SCPs) — SCPs são políticas JSON que especificam as permissões máximas para uma organização ou unidade organizacional (OU) em. AWS Organizations AWS Organizations é um serviço para agrupar e gerenciar centralmente vários Contas da AWS que sua empresa possui. Se você habilitar todos os atributos em uma organização, poderá aplicar políticas de controle de serviço (SCPs) a qualquer uma ou a todas as contas. O SCP limita as permissões para entidades nas contas dos membros, incluindo cada uma Usuário raiz da conta da AWS. Para mais informações sobre Organizações e SCPs, consulte Como os SCPs funcionam no AWS Organizations Guia do Usuário.
- Políticas de sessão: são políticas avançadas que você transmite como um parâmetro quando cria de forma programática uma sessão temporária para um perfil ou um usuário federado. As permissões da sessão resultante são a interseção das políticas baseadas em identidade do usuário ou do perfil e das políticas de sessão. As permissões também podem ser provenientes de uma política baseada em recurso. Uma negação explícita em qualquer uma dessas políticas substitui a permissão. Para mais informações, consulte Políticas de sessão no Guia do usuário do IAM.

Vários tipos de política

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

Como AWS Batch funciona com o IAM

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

Recursos do IAM que você pode usar com AWS Batch

Atributo do IAM	AWS Batch apoio
Políticas baseadas em identidade	Sim
Políticas baseadas em recurso	Não
Ações de políticas	Sim

Atributo do IAM	AWS Batch apoio
recursos de políticas	Sim
Chaves de condição de políticas	Sim
ACLs	Não
ABAC (tags em políticas)	Sim
Credenciais temporárias	Sim
Permissões de entidade principal	Sim
Perfis de serviço	Sim
Perfis vinculados ao serviço	Sim

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

Políticas baseadas em identidade para AWS Batch

É compatível com políticas baseadas em identidade	Sim

As políticas baseadas em identidade são documentos de políticas de permissões JSON que você pode anexar a uma identidade, como usuário, grupo de usuários ou perfil do IAM. Essas políticas controlam quais ações os usuários e funções podem realizar, em quais recursos e em que condições. Para saber como criar uma política baseada em identidade, consulte Criar políticas do IAM no Guia do usuário do IAM.

Com as políticas baseadas em identidade do IAM, é possível especificar ações ou recursos permitidos ou negados, bem como as condições sob as quais as ações são permitidas ou negadas. Você não pode especificar a entidade principal em uma política baseada em identidade porque ela se aplica ao usuário ou função à qual ela está anexada. Saiba mais sobre todos os elementos que

podem ser usados em uma política JSON consultando Referência de Elementos de Política JSON do IAM no Guia do Usuário do IAM.

Exemplos de políticas baseadas em identidade para o AWS Batch

Para ver exemplos de políticas AWS Batch baseadas em identidade, consulte. Exemplos de políticas baseadas em identidade para AWS Batch

Ações políticas para AWS Batch

Oferece suporte a ações de políticas	Sim

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

O elemento `Action` de uma política JSON descreve as ações que você pode usar para permitir ou negar acesso em uma política. As ações de política geralmente têm o mesmo nome da operação de AWS API associada. Existem algumas exceções, como ações somente de permissão, que não têm uma operação de API correspondente. Há também algumas operações que exigem várias ações em uma política. Essas ações adicionais são chamadas de ações dependentes.

Incluem ações em uma política para conceder permissões para executar a operação associada.

Para ver uma lista de AWS Batch ações, consulte Ações definidas por AWS Batch na Referência de autorização de serviço.

As ações de política AWS Batch usam o seguinte prefixo antes da ação:

```
batch
```

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

```
"Action": [
      "batch:action1",
      "batch:action2"
         ]
```

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

```
"Action": "batch:Describe*"
```

Para ver exemplos de políticas AWS Batch baseadas em identidade, consulte. Exemplos de políticas baseadas em identidade para AWS Batch

Recursos políticos para AWS Batch

Oferece suporte a recursos de políticas	Sim

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

O elemento de política `Resource` JSON especifica o objeto ou os objetos aos quais a ação se aplica. As instruções devem incluir um elemento `Resource` ou um elemento `NotResource`. Como prática recomendada, especifique um recurso usando seu nome do recurso da Amazon (ARN). Isso pode ser feito para ações que oferecem suporte a um tipo de recurso específico, conhecido como permissões em nível de recurso.

Para ações que não oferecem suporte a permissões em nível de recurso, como operações de listagem, use um asterisco (*) para indicar que a instrução se aplica a todos os recursos.

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

Para ver uma lista dos tipos de AWS Batch recursos e seus ARNs, consulte Recursos definidos por AWS Batch na Referência de autorização de serviço. Para saber com quais ações você pode especificar o ARN de cada recurso, consulte Ações definidas pelo AWS Batch.

Para ver exemplos de políticas AWS Batch baseadas em identidade, consulte. Exemplos de políticas baseadas em identidade para AWS Batch

Chaves de condição de políticas para AWS Batch

Compatível com chaves de condição de política específicas do serviço	Sim

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

O elemento `Condition` (ou bloco de `Condition`) permite que você especifique condições nas quais uma instrução está em vigor. O elemento `Condition` é opcional. É possível criar expressões condicionais que usam atendentes de condição, como “igual a” ou “menor que”, para fazer a condição da política corresponder aos valores na solicitação.

Se você especificar vários elementos `Condition` em uma instrução ou várias chaves em um único elemento `Condition`, a AWS os avaliará usando uma operação lógica `AND`. Se você especificar vários valores para uma única chave de condição, AWS avalia a condição usando uma `OR` operação lógica. Todas as condições devem ser atendidas para que as permissões da instrução sejam concedidas.

Você também pode usar variáveis de espaço reservado ao especificar as condições. Por exemplo, é possível conceder a um usuário do IAM permissão para acessar um recurso somente se ele estiver marcado com seu nome de usuário do IAM. Para mais informações, consulte Elementos de política do IAM: variáveis e tags no Guia do usuário do IAM.

AWS suporta chaves de condição globais e chaves de condição específicas do serviço. Para ver todas as chaves de condição AWS globais, consulte as chaves de contexto de condição AWS global no Guia do usuário do IAM.

Para ver uma lista de chaves de AWS Batch condição, consulte Chaves de condição AWS Batch na Referência de autorização de serviço. Para saber com quais ações e recursos você pode usar uma chave de condição, consulte Ações definidas por AWS Batch.

Para ver exemplos de políticas AWS Batch baseadas em identidade, consulte. Exemplos de políticas baseadas em identidade para AWS Batch

Controle de acesso baseado em atributos (ABAC) com AWS Batch

Oferece suporte a ABAC (tags em políticas)	Sim

O controle de acesso por atributo (ABAC) é uma estratégia de autorização que define permissões com base em atributos. Em AWS, esses atributos são chamados de tags. Você pode anexar tags a entidades do IAM (usuários ou funções) e a vários AWS recursos. A marcação de entidades

e recursos é a primeira etapa do ABAC. Em seguida, você cria políticas de ABAC para permitir operações quando a tag da entidade principal corresponder à tag do recurso que ela está tentando acessar.

O ABAC é útil em ambientes que estão crescendo rapidamente e ajuda em situações em que o gerenciamento de políticas se torna um problema.

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

Se um serviço oferecer suporte às três chaves de condição para cada tipo de recurso, o valor será Sim para o serviço. Se um serviço oferecer suporte às três chaves de condição somente para alguns tipos de recursos, o valor será Parcial.

Para mais informações sobre o ABAC, consulte O que é ABAC? no Guia do Usuário do IAM. Para visualizar um tutorial com etapas para configurar o ABAC, consulte Usar controle de acesso por atributo (ABAC) no Guia do usuário do IAM.

Usando credenciais temporárias com AWS Batch

Oferece suporte a credenciais temporárias	Sim

Alguns Serviços da AWS não funcionam quando você faz login usando credenciais temporárias. Para obter informações adicionais, incluindo quais Serviços da AWS funcionam com credenciais temporárias, consulte Serviços da AWS “Trabalhe com o IAM” no Guia do usuário do IAM.

Você está usando credenciais temporárias se fizer login AWS Management Console usando qualquer método, exceto um nome de usuário e senha. Por exemplo, quando você acessa AWS usando o link de login único (SSO) da sua empresa, esse processo cria automaticamente credenciais temporárias. Você também cria automaticamente credenciais temporárias quando faz login no console como usuário e, em seguida, alterna perfis. Para mais informações sobre como alternar funções, consulte Alternar para uma função (console) no Guia do usuário do IAM.

Você pode criar manualmente credenciais temporárias usando a AWS API AWS CLI ou. Em seguida, você pode usar essas credenciais temporárias para acessar AWS. AWS recomenda que você gere credenciais temporárias dinamicamente em vez de usar chaves de acesso de longo prazo. Para obter mais informações, consulte Credenciais de segurança temporárias no IAM.

Permissões de entidade principal entre serviços para o AWS Batch

Suporte para o recurso Encaminhamento de sessões de acesso (FAS)	Sim

Quando você usa um usuário ou uma função do IAM para realizar ações AWS, você é considerado principal. Ao usar alguns serviços, você pode executar uma ação que inicia outra ação em um serviço diferente. O FAS usa as permissões do diretor chamando um AWS service (Serviço da AWS), combinadas com a solicitação AWS service (Serviço da AWS) para fazer solicitações aos serviços posteriores. As solicitações do FAS são feitas somente quando um serviço recebe uma solicitação que requer interações com outros Serviços da AWS ou com recursos para ser concluída. Nesse caso, você precisa ter permissões para executar ambas as ações. Para obter detalhes da política ao fazer solicitações de FAS, consulte Encaminhar sessões de acesso.

Funções de serviço para AWS Batch

Oferece suporte a perfis de serviço	Sim

Um perfil de serviço é um perfil do IAM que um serviço assume para executar ações em seu nome. Um administrador do IAM pode criar, modificar e excluir um perfil de serviço do IAM. Para obter mais informações, consulte Criar um perfil para delegar permissões a um no Guia do usuário do IAM.

> **Warning**
>
> Alterar as permissões de uma função de serviço pode interromper AWS Batch a funcionalidade. Edite os perfis de serviço somente quando o AWS Batch fornecer orientação para isso.

Funções vinculadas a serviços para AWS Batch

Oferece suporte a funções vinculadas ao serviço	Sim

Uma função vinculada ao serviço é um tipo de função de serviço vinculada a um. AWS service (Serviço da AWS) O serviço pode assumir o perfil para executar uma ação em seu nome. As funções vinculadas ao serviço aparecem em você Conta da AWS e são de propriedade do serviço. Um administrador do IAM pode visualizar, mas não pode editar as permissões para perfis vinculados ao serviço.

Para obter detalhes sobre como criar ou gerenciar perfis vinculados a serviços, consulte Serviços da AWS que funcionam com o IAM. Encontre um serviço na tabela que inclua um `Yes` na coluna Perfil vinculado ao serviço. Escolha o link Sim para visualizar a documentação do perfil vinculado a esse serviço.

AWS Batch função de execução do IAM

A função de execução concede ao contêiner e aos AWS Fargate agentes do Amazon ECS permissão para fazer chamadas de AWS API em seu nome.

> **Note**
>
> O papel de execução é suportado pela versão do agente 1.16.0 do atendente de contêiner do Amazon ECS e superior.

É necessário informar o perfil do IAM de execução dependendo dos requisitos da sua tarefa. Você pode ter vários papéis de execução para diferentes objetivos e associações de serviço em sua conta.

> **Note**
>
> Para obter mais informações sobre a função da instância do Amazon ECS, consulte Perfil de instância do Amazon ECS. Para obter informações sobre perfis de serviço, consulte Como AWS Batch funciona com o IAM.

O Amazon ECS fornece a política gerenciada `AmazonECSTaskExecutionRolePolicy`. Esta politica contém as permissões obrigatórias para os casos de uso comuns descritos acima. Talvez seja necessário adicionar políticas em linha ao seu perfil de execução de trabalhos para os casos de uso especiais resumidos abaixo.

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecr:GetAuthorizationToken",
        "ecr:BatchCheckLayerAvailability",
        "ecr:GetDownloadUrlForLayer",
        "ecr:BatchGetImage",
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "*"
    }
  ]
}
```

Você pode usar o procedimento a seguir para verificar se a sua conta já tem o papel de execução e anexar a política do IAM gerenciada, caso necessário.

Para verificar a **`ecsTaskExecutionRole`** no campo de console do IAM

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Funções.
3. Pesquise `ecsTaskExecutionRole` na lista de funções. Se você não conseguir encontrar a função, consulte Como criar o perfil do IAM de execução. Se a função existir, escolha-a para visualizar as políticas anexadas.
4. Na guia Permissões, verifique se a política gerenciada do TaskExecutionRolePolicyAmazonECS está anexada à função. Se a política estiver anexada, isso significa que seu perfil de execução está configurado corretamente. Caso contrário, siga as etapas secundárias abaixo para anexar a política.

   a. Escolha Adicionar Permissões e depois escolha Anexar Políticas.

   b. Pesquise o TaskExecutionRolePolicyAmazonECS.

   c. Marque a caixa à esquerda da política do TaskExecutionRolePolicyAmazonECS e escolha Anexar políticas.
5. Escolha Relações de Confiança.
6. Verifique se a relação de confiança contém a seguinte política. Se a relação de confiança corresponder à política abaixo, a função será configurada corretamente. Se a relação de

confiança não corresponder, escolha Editar Política de Confiança, insira o seguinte e escolha Atualizar Política.

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

Como criar o perfil do IAM de execução

Se a sua conta ainda não tiver um papel de execução, use as etapas a seguir para criar a função.

Para criar o perfil do IAM **`ecsTaskExecutionRole`**

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Roles.
3. Escolha Criar Perfil.
4. Em Tipo de Entidade Confiável, escolha AWS service (Serviço da AWS).
5. Para Serviço ou Caso de Uso, escolha EC2. Em seguida, escolha EC2 novamente.
6. Escolha Próximo.
7. Para políticas de permissões, pesquise TaskExecutionRolePolicyAmazonECS.
8. Escolha a caixa de seleção à esquerda da política do TaskExecutionRolePolicyAmazonECS e, em seguida, escolha Avançar.
9. Em Nome da Função, insira `ecsTaskExecutionRole` e então, escolha Criar Função.

Exemplos de políticas baseadas em identidade para AWS Batch

Por padrão, usuários e funções não têm permissão para criar ou modificar recursos do AWS Batch . Eles também não podem realizar tarefas usando a AWS API AWS Management Console, AWS Command Line Interface (AWS CLI) ou. Para conceder aos usuários permissão para executar ações nos recursos de que eles precisam, um administrador do IAM pode criar políticas do IAM. O administrador pode então adicionar as políticas do IAM aos perfis, e os usuários podem assumir os perfis.

Para saber como criar uma política baseada em identidade do IAM usando esses exemplos de documento de política JSON, consulte Criação de políticas do IAM no Guia do Usuário do IAM.

Para obter detalhes sobre ações e tipos de recursos definidos por AWS Batch, incluindo o formato dos ARNs para cada um dos tipos de recursos, consulte Ações, recursos e chaves de condição AWS Batch na Referência de autorização de serviço.

Tópicos

- Melhores práticas de política
- Usando o AWS Batch console
- Permitir que os usuários visualizem suas próprias permissões

Melhores práticas de política

As políticas baseadas em identidade determinam se alguém pode criar, acessar ou excluir AWS Batch recursos em sua conta. Essas ações podem incorrer em custos para a Conta da AWS. Ao criar ou editar políticas baseadas em identidade, siga estas diretrizes e recomendações:

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

conhecidas como permissões de privilégio mínimo. Para mais informações sobre como usar o IAM para aplicar permissões, consulte Políticas e permissões no IAM no Guia do usuário do IAM.

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

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

Usando o AWS Batch console

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

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

Para garantir que usuários e funções ainda possam usar o AWS Batch console, anexe também a política AWS Batch `ConsoleAccess` ou a política `ReadOnly` AWS gerenciada às entidades. Para obter mais informações, consulte Adicionando Permissões a um Usuário no Guia do Usuário do IAM.

Permitir que os usuários visualizem suas próprias permissões

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

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

Prevenção contra o ataque “Confused deputy” entre serviços

O problema “confused deputy” é um problema de segurança em que uma entidade que não tem permissão para executar uma ação pode coagir uma entidade mais privilegiada a executá-la. Em AWS, a falsificação de identidade entre serviços pode resultar no problema confuso do deputado. A imitação entre serviços pode ocorrer quando um serviço (o serviço de chamada) chama outro serviço (o serviço chamado). O serviço de chamada pode ser manipulado para utilizar as suas permissões para atuar nos recursos de outro cliente em que, de outra forma, ele não teria permissão para acessar. Para evitar isso, a AWS fornece ferramentas que ajudam você a proteger seus dados para todos os serviços com entidades principais de serviço que receberam acesso aos recursos em sua conta.

Recomendamos usar as chaves de contexto de condição `aws:SourceAccount`global `aws:SourceArn`e as chaves de contexto nas políticas de recursos para limitar as permissões que AWS Batch concedem outro serviço ao recurso. Se o valor de `aws:SourceArn` não contém ID da conta, como um ARN do bucket do Amazon S3, você deve usar ambas as chaves de contexto de condição global para limitar as permissões. Se você usa ambas as chaves de contexto de condição global, e o valor `aws:SourceArn` contém o ID da conta, o valor `aws:SourceAccount` e a conta no valor `aws:SourceArn` deverão utilizar a mesma ID de conta quando na mesma declaração de política. Utilize `aws:SourceArn` se quiser que apenas um recurso seja associado ao acesso entre serviços. Use `aws:SourceAccount` se quiser permitir que qualquer recurso nessa conta seja associado ao uso entre serviços.

O valor de `aws:SourceArn` deve ser o recurso AWS Batch armazenado.

A maneira mais eficaz de se proteger contra o problema do substituto confuso é usar a chave de contexto de condição global `aws:SourceArn` com o ARN completo do recurso. Se você não souber o ARN completo do recurso ou estiver especificando vários recursos, utilize a chave de condição de contexto global `aws:SourceArn` com caracteres curingas (`*`) para as partes desconhecidas do ARN. Por exemplo, `.arn:aws:`*`servicename`*`:*:`*`123456789012`*`:*`

Os exemplos a seguir mostram como você pode usar as chaves de contexto de condição `aws:SourceAccount` global `aws:SourceArn` e as chaves de contexto AWS Batch para evitar o confuso problema substituto.

Exemplo 1: função para acessar apenas um ambientes de computação

A função a seguir pode ser usada apenas para acessar um ambiente computacional. O nome do trabalho deve ser especificado como *, porque a fila de trabalhos pode ser associada a vários ambientes de computação.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "batch.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:batch:us-east-1:123456789012:compute-environment/testCE",
            "arn:aws:batch:us-east-1:123456789012:job/*"
          ]
        }
      }
    }
  ]
}
```

Exemplo 2: função para acessar vários ambientes de computação

A função a seguir pode ser usada apenas para acessar vários ambientes computacionais.. O nome do trabalho deve ser especificado como *, porque a fila de trabalhos pode ser associada a vários ambientes de computação.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
```

```
        "Service": "batch.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceAccount": "123456789012"
        },
        "ArnLike": {
          "aws:SourceArn": [
            "arn:aws:batch:us-east-1:123456789012:compute-environment/*",
            "arn:aws:batch:us-east-1:123456789012:job/*"
          ]
        }
      }
    }
  ]
}
```

Solução de problemas AWS Batch de identidade e acesso

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

Tópicos

- Não tenho autorização para executar uma ação no AWS Batch
- Não estou autorizado a realizar iam: PassRole
- Quero permitir que pessoas fora da minha AWS conta acessem meus AWS Batch recursos

Não tenho autorização para executar uma ação no AWS Batch

Se isso AWS Management Console indicar que você não está autorizado a realizar uma ação, entre em contato com o administrador para obter ajuda. O administrador é a pessoa que forneceu o seu nome de usuário e senha.

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

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

Neste caso, Mateo pede ao administrador para atualizar suas políticas e permitir o acesso ao recurso *my-example-widget* usando a ação `batch:GetWidget`. Para obter mais informações sobre como conceder permissões para passar uma função, consulte Conceder permissões a um usuário para passar uma função para um AWS serviço.

Não estou autorizado a realizar iam: PassRole

Se você receber uma mensagem de erro informando que não está autorizado a executar a ação `iam:PassRole`, as suas políticas devem ser atualizadas para permitir que você passe uma função para o AWS Batch.

Alguns Serviços da AWS permitem que você passe uma função existente para esse serviço em vez de criar uma nova função de serviço ou uma função vinculada ao serviço. Para fazê-lo, você deve ter permissões para passar o perfil para o serviço.

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

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

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

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

Quero permitir que pessoas fora da minha AWS conta acessem meus AWS Batch recursos

Você pode criar uma função que os usuários de outras contas ou pessoas fora da sua organização podem usar para acessar seus recursos. Você pode especificar quem é confiável para assumir o perfil. Para serviços que oferecem suporte a políticas baseadas em recursos ou listas de controle de acesso (ACLs), você pode usar essas políticas para conceder às pessoas acesso aos seus recursos.

Para saber mais, consulte:

- Para saber se é AWS Batch compatível com esses recursos, consulteComo AWS Batch funciona com o IAM.
- Para saber como fornecer acesso aos seus recursos em todos os Contas da AWS que você possui, consulte Como fornecer acesso a um usuário do IAM em outro Conta da AWS que você possui no Guia do usuário do IAM.
- Para saber como fornecer acesso aos seus recursos a terceiros Contas da AWS, consulte Como fornecer acesso Contas da AWS a terceiros no Guia do usuário do IAM.
- Para saber como conceder acesso por meio da federação de identidades, consulte Conceder acesso a usuários autenticados externamente (federação de identidades) no Guia do usuário do IAM.
- Para saber a diferença entre usar perfis e políticas baseadas em recursos para acesso entre contas, consulte Como os perfis do IAM diferem de políticas baseadas em recursos no Guia do usuário do IAM.

Usando funções vinculadas a serviços para AWS Batch

AWS Batch usa funções vinculadas ao serviço AWS Identity and Access Management (IAM). Uma função vinculada ao serviço é um tipo exclusivo de função do IAM vinculada diretamente a. AWS Batch As funções vinculadas ao serviço são predefinidas AWS Batch e incluem todas as permissões que o serviço exige para chamar outros AWS serviços em seu nome.

Uma função vinculada ao serviço facilita a configuração AWS Batch porque você não precisa adicionar manualmente as permissões necessárias. AWS Batch define as permissões de suas funções vinculadas ao serviço e, a menos que seja definido de outra forma, só AWS Batch pode assumir suas funções. As permissões definidas incluem a política de confiança e a política de permissões, que não pode ser anexada a nenhuma outra entidade do IAM.

> **Note**
>
> Siga um destes procedimentos para especificar uma função de serviço para um ambiente AWS Batch computacional.
>
> - Use uma string vazia para o perfil de serviço. Isso permite AWS Batch criar a função de serviço.
> - Especifique o perfil de serviço no seguinte formato: `arn:aws:iam::account_number:role/aws-service-role/batch.amazonaws.com/AWSServiceRoleForBatch`.

Para obter mais informações, consulte the section called “Nome do perfil ou ARN incorreto” o Guia AWS Batch do usuário.

Um perfil vinculado a serviço poderá ser excluído somente após a exclusão dos recursos relacionados. Isso protege seus recursos do AWS Batch , pois você não pode remover por engano as permissões de acesso aos recursos.

Para obter informações sobre outros serviços compatíveis com funções vinculadas a serviços, consulte Serviços da AWS compatíveis com o IAM e procure os serviços que apresentam Sim na coluna Função vinculada a serviço. Escolha um Sim com um link para visualizar a documentação do perfil vinculado a esse serviço.

Permissões de função vinculadas ao serviço para AWS Batch

AWS Batch usa a função vinculada ao serviço chamada. AWSServiceRoleForBatch A AWSServiceRoleForBatchfunção permite AWS Batch criar e gerenciar AWS recursos em seu nome.

A função AWSServiceRoleForBatchvinculada ao serviço confia no diretor do `batch.amazonaws.com` serviço para assumir a função.

A política do IAM nomeada BatchServiceRolePolicy AWS Batch permite concluir as seguintes ações em recursos específicos:

- `autoscaling`— Permite AWS Batch criar e gerenciar recursos do Amazon EC2 Auto Scaling. AWS Batch cria e gerencia grupos do Amazon EC2 Auto Scaling para a maioria dos ambientes computacionais.
- `ec2`— Permite AWS Batch controlar o ciclo de vida das instâncias do Amazon EC2, bem como criar e gerenciar modelos e tags de lançamento. AWS Batch cria e gerencia solicitações do EC2 Spot Fleet para alguns ambientes computacionais EC2 Spot.
- `ecs`- Permite AWS Batch criar e gerenciar clusters, definições de tarefas e tarefas do Amazon ECS para execução de trabalhos.
- `eks`- Permite AWS Batch descrever o recurso de cluster Amazon EKS para validações.
- `iam`- Permite AWS Batch validar e passar funções fornecidas pelo proprietário para o Amazon EC2, o Amazon EC2 Auto Scaling e o Amazon ECS.
- `logs`— Permite AWS Batch criar e gerenciar grupos de registros e fluxos de registros para AWS Batch trabalhos.

Você deve configurar permissões para que uma entidade do IAM (por exemplo, um usuário, grupo ou função) crie, edite ou exclua uma função vinculada a serviço. Para mais informações, consulte Permissões de perfil vinculado ao serviço no Guia do usuário do IAM.

Criação de uma função vinculada ao serviço para AWS Batch

Não é necessário criar manualmente uma função vinculada a serviço. Quando você está CreateComputeEnvironment na AWS Management Console AWS CLI, na ou na AWS API e não especifica um valor para o `serviceRole` parâmetro, AWS Batch cria a função vinculada ao serviço para você.

> **Important**
>
> Esse perfil vinculado ao serviço pode aparecer em sua conta se você concluiu uma ação em outro serviço que usa os atributos compatíveis com esse perfil. Além disso, se você estava usando o AWS Batch serviço antes de 10 de março de 2021, quando ele começou a oferecer suporte a funções vinculadas ao serviço, AWS Batch criou a AWSServiceRoleForBatch função em sua conta. Para saber mais, consulte Uma Nova Função Apareceu na minha Conta do IAM.

Se excluir essa função vinculada ao serviço e precisar criá-la novamente, você pode usar esse mesmo processo para recriar a função na sua conta. Quando você CreateComputeEnvironment, AWS Batch cria a função vinculada ao serviço para você novamente.

Editando uma função vinculada ao serviço para AWS Batch

Com AWS Batch, você não pode editar a função AWSServiceRoleForBatch vinculada ao serviço. Depois que você criar um perfil vinculado ao serviço, não poderá alterar o nome do perfil, pois várias entidades podem fazer referência ao perfil. No entanto, você poderá editar a descrição do perfil usando o IAM. Para obter mais informações, consulte Editar uma função vinculada a serviço no Guia do usuário do IAM.

Para permitir que uma entidade do IAM edite a descrição da função AWSServiceRoleForBatch vinculada ao serviço

Adicione a seguinte instrução a política de permissões. Isso permite que que uma entidade do IAM edite a descrição de uma função vinculada ao serviço.

```
{
```

```
    "Effect": "Allow",
    "Action": [
        "iam:UpdateRoleDescription"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/batch.amazonaws.com/
AWSServiceRoleForBatch",
    "Condition": {"StringLike": {"iam:AWSServiceName": "batch.amazonaws.com"}}
}
```

Excluindo uma função vinculada ao serviço para AWS Batch

Se você não precisar mais usar um atributo ou serviço que requeira uma função vinculada a serviço, recomendamos que você a exclua. Dessa forma, você não terá uma entidade não usada não monitorada ou mantida ativamente. No entanto, você deve limpar os recursos de sua função vinculada ao serviço antes de poder exclui-la manualmente.

Para permitir que uma entidade do IAM exclua a função AWSServiceRoleForBatch vinculada ao serviço

Adicione a seguinte instrução a política de permissões. Isso permite que a entidade do IAM delete a função vinculada ao serviço.

```
{
    "Effect": "Allow",
    "Action": [
        "iam:DeleteServiceLinkedRole",
        "iam:GetServiceLinkedRoleDeletionStatus"
    ],
    "Resource": "arn:aws:iam::*:role/aws-service-role/batch.amazonaws.com/
AWSServiceRoleForBatch",
    "Condition": {"StringLike": {"iam:AWSServiceName": "batch.amazonaws.com"}}
}
```

Limpar um perfil vinculado ao serviço

Antes de usar o IAM para excluir uma função vinculada ao serviço, você deve primeiro confirmar se a função não tem sessões ativas e excluir todos os ambientes AWS Batch computacionais que usam a função em todas as AWS regiões em uma única partição.

Para verificar se a função vinculada ao serviço tem uma sessão ativa

1. Abra o console IAM em https://console.aws.amazon.com/iam/.

2. No painel de navegação, escolha Funções e, em seguida, o AWSServiceRoleForBatch nome (não a caixa de seleção).
3. Na página Resumo, escolha Consultor de Acesso e analise as atividades recentes para a função vinculada ao serviço.

> **Note**
>
> Se você não sabe se AWS Batch está usando a AWSServiceRoleForBatch função, tente excluir a função. Se o serviço estiver usando a função, haverá falha ao excluir a função. Você pode visualizar as regiões nas quais a função estiver sendo usada. Se a função estiver sendo usada, você deve aguardar a sessão final antes de excluir a função. Você não pode revogar a sessão de uma função vinculada ao serviço.

Para remover AWS Batch recursos usados pela função vinculada ao AWSServiceRoleForBatch serviço

Você deve excluir todos os ambientes AWS Batch computacionais que usam a AWSServiceRoleForBatch função em todas as AWS regiões antes de excluir a AWSServiceRoleForBatch função.

1. Abra o AWS Batch console em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione a Região a ser usada.
3. No painel de navegação, escolha Ambientes de computação.
4. Selecione o ambiente de computação.
5. Escolha Desabilitar. Espere até que o Estado mude para DESATIVADO.
6. Selecione o ambiente de computação.
7. Escolha Deletar. Confirme que você deseja excluir o ambiente de computação escolhendo Excluir Ambiente de Computação..
8. Repita as etapas 1 a 7 para todos os ambientes de computação usando a função vinculada ao serviço em todas as regiões.

Excluir uma função vinculada ao serviço no IAM (Console)

Você pode usar o console do IAM para excluir uma função vinculada ao serviço.

Para excluir uma função vinculada ao serviço (console)

1. Faça login AWS Management Console e abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação do console do IAM, escolha Perfis. Em seguida, marque a caixa de seleção ao lado AWSServiceRoleForBatch, não o nome ou a linha em si.
3. Clique em Excluir função.
4. Na caixa de diálogo de confirmação, revise os dados do último acesso ao serviço mostrando quando cada uma das funções selecionadas acessou pela última vez um AWS service (Serviço da AWS). Isso ajuda você a confirmar se a função está ativo no momento. Se quiser prosseguir, escolha Sim, Excluir para enviar a função vinculada ao serviço para exclusão.
5. Monitore as notificações do console do IAM para progresso da exclusão da função vinculada ao serviço. Como a exclusão do perfil vinculado ao serviço do IAM é assíncrona, depois de enviar a função para exclusão, a exclusão da tarefa pode obter êxito ser reprovada.

   - Se a tarefa for bem-sucedida, a função será removida da lista e uma notificação de sucesso será exibida na parte superior da página.
   - Se a tarefa obtiver êxito, você poderá escolher Visualizar Detalhes ou Visualizar Recursos a partir das notificações para saber por que a exclusão falhou. Se a exclusão falhou porque a função está usando os recursos do serviço, a notificação incluirá uma lista de recursos, caso o serviço retorne essas informações. Você poderá então limpar os recursos e enviar novamente a exclusão.

     > **Note**
     >
     > Você pode repetir esse processo várias vezes, a depender das informações que o serviço retornar. Por exemplo, a função vinculada ao serviço pode usar seis recursos, e seu serviço pode retornar informações sobre cinco deles. Se você limpar cinco recursos e enviar a função para exclusão novamente, a exclusão falhará e o serviço emitirá relatório sobre o recurso restante. Um serviço pode retornar todos os recursos, alguns deles, ou pode não retornar relatórios de nenhum dos recursos.

   - Se a tarefa falhar e a notificação não incluir uma lista de recursos, o serviço pode não retornar essas informações. Para saber como limpar os recursos para esse serviço, consulte Serviços da AWS que Funcionam com o IAM. Descubra o serviço na tabela e escolha o link Sim para visualizar a documentação da função vinculada ao serviço.

Excluir uma função vinculada ao serviço no IAM (AWS CLI)

Você pode usar os comandos do IAM do AWS Command Line Interface para excluir uma função vinculada ao serviço.

Para excluir uma função vinculado ao serviço (CLI)

1. Como uma função vinculada ao serviço não pode ser excluída se estiver sendo usada ou tiver recursos associados, você deve enviar uma solicitação de exclusão. Essa solicitação pode ser negada se essas condições não forem atendidas. Você deve capturar o `deletion-task-id` da resposta para verificar o status da tarefa de exclusão. Insira o seguinte comando para enviar uma solicitação de exclusão de função vinculada ao serviço:

   ```
   $ aws iam delete-service-linked-role --role-name AWSServiceRoleForBatch
   ```

2. Use o seguinte comando para verificar o status da tarefa de exclusão:

   ```
   $ aws iam get-service-linked-role-deletion-status --deletion-task-id deletion-task-id
   ```

   O status da tarefa de exclusão pode ser `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED`, ou `FAILED`. Se a exclusão falhar, a chamada retornará o motivo de falha para que você possa acionar a solução de problemas. Se a exclusão falhar porque a função estiver usando os recursos do serviço, a notificação incluirá uma lista de recursos, caso o serviço retorne essas informações. Você poderá então limpar os recursos e enviar novamente a exclusão.

   > **Note**
   >
   > Você pode repetir esse processo várias vezes, a depender das informações que o serviço retornar. Por exemplo, a função vinculada ao serviço pode usar seis recursos, e seu serviço pode retornar informações sobre cinco deles. Se você limpar cinco recursos e enviar a função para exclusão novamente, a exclusão falhará e o serviço emitirá relatório sobre o recurso restante. Um serviço pode retornar todos os recursos, alguns deles. Ou talvez não relate qualquer recurso. Saiba como limpar os recursos de um serviço que não reporta qualquer recurso consultando Serviços AWS Suportados pelo IAM. Descubra o seu serviço na tabela e escolha o link Sim para ver a documentação da função vinculada ao serviço.

Excluir uma função vinculada ao serviço no IAM (API da AWS)

Você pode usar a API do IAM para excluir uma função vinculada ao serviço.

Para excluir uma função vinculada ao serviço (API)

1. Para enviar uma solicitação de exclusão para uma lista vinculada ao serviço, ligue. DeleteServiceLinkedRole Na solicitação, especifique o nome da AWSServiceRoleForBatch função.

   Como uma função vinculada ao serviço não podem ser excluída se estiver sendo usada ou tiver recursos associados, você deverá enviar uma solicitação de exclusão. Essa solicitação poderá ser negada se essas condições não forem atendidas. Você deve capturar o `DeletionTaskId` da resposta para verificar o status da tarefa de exclusão.
2. Para verificar o status da exclusão, ligue GetServiceLinkedRoleDeletionStatus. Na solicitação, especifique o `DeletionTaskId`.

   O status da tarefa de exclusão pode ser `NOT_STARTED`, `IN_PROGRESS`, `SUCCEEDED`, ou `FAILED`. Se a exclusão falhar, a chamada retornará o motivo de falha para que você possa acionar a solução de problemas. Se a exclusão falhar porque a função estiver usando os recursos do serviço, a notificação incluirá uma lista de recursos, caso o serviço retorne essas informações. Você poderá então limpar os recursos e enviar novamente a exclusão.

   > **Note**
   >
   > Você pode repetir esse processo várias vezes, a depender das informações que o serviço retornar. Por exemplo, a função vinculada ao serviço pode usar seis recursos, e seu serviço pode retornar informações sobre cinco deles. Se você limpar cinco recursos e enviar a função para exclusão novamente, a exclusão falhará e o serviço emitirá relatório sobre o recurso restante. Um serviço pode retornar todos os recursos, alguns deles, ou pode não retornar relatórios de nenhum dos recursos. Saiba como limpar os recursos de um serviço que não relata qualquer recurso consultandoServiços da AWS que funcionam com o IAM. Descubra o seu serviço na tabela e escolha o link Sim para ver a documentação da função vinculada ao serviço.

Regiões suportadas para funções vinculadas a AWS Batch serviços

AWS Batch suporta o uso de funções vinculadas ao serviço em todas as regiões em que o serviço está disponível. Para obter mais informações, consulte AWS Batch Endpoints do.

AWS políticas gerenciadas para AWS Batch

Você pode usar políticas AWS gerenciadas para simplificar o gerenciamento do acesso à identidade da sua equipe e dos recursos provisionados AWS . AWS as políticas gerenciadas abrangem uma variedade de casos de uso comuns, estão disponíveis por padrão em sua AWS conta e são mantidas e atualizadas em seu nome. Você não pode alterar as permissões nas políticas AWS gerenciadas. Se o requisito for maior flexibilidade, você também pode optar por criar políticas gerenciadas pelo cliente do IAM. Dessa forma, você pode fornecer a sua equipe recursos provisionados apenas com as permissões exatas de que ela precisa.

Para obter mais informações sobre políticas AWS gerenciadas, consulte políticas AWS gerenciadas no Guia do usuário do IAM.

AWS os serviços mantêm e atualizam as políticas AWS gerenciadas em seu nome. Periodicamente, AWS os serviços adicionam permissões adicionais a uma política AWS gerenciada. AWS as políticas gerenciadas provavelmente são atualizadas quando o lançamento ou operação de um novo recurso é disponibilizado. Essas atualizações afetam automaticamente todas as identidades (usuários, grupos e funções) em que a política está anexada. No entanto, eles não removem as permissões nem interrompem as permissões existentes.

Além disso, AWS oferece suporte a políticas gerenciadas para funções de trabalho que abrangem vários serviços. Por exemplo, a política `ReadOnlyAccess` AWS gerenciada fornece acesso somente de leitura a todos os AWS serviços e recursos. Quando um serviço lança um novo recurso, AWS adiciona permissões somente de leitura para novas operações e recursos. Para obter uma lista e descrições das políticas de perfis de trabalho, consulte Políticas gerenciadas pela AWS para perfis de trabalho no Guia do usuário do IAM.

AWS política gerenciada: BatchServiceRolePolicy

A política BatchServiceRolePolicygerenciada do IAM é usada pela função `AWSServiceRoleForBatch`vinculada ao serviço. Isso permite AWS Batch realizar ações em seu

nome. Não é possível anexar essa política a suas entidades do IAM. Para obter mais informações, consulte Usando funções vinculadas a serviços para AWS Batch.

Essa política permite AWS Batch concluir as seguintes ações em recursos específicos:

- `autoscaling`— Permite AWS Batch criar e gerenciar recursos do Amazon EC2 Auto Scaling. AWS Batch cria e gerencia grupos do Amazon EC2 Auto Scaling para a maioria dos ambientes computacionais.
- `ec2`— Permite AWS Batch controlar o ciclo de vida das instâncias do Amazon EC2, bem como criar e gerenciar modelos e tags de lançamento. AWS Batch cria e gerencia solicitações do EC2 Spot Fleet para alguns ambientes computacionais EC2 Spot.
- `ecs`- Permite AWS Batch criar e gerenciar clusters, definições de tarefas e tarefas do Amazon ECS para execução de trabalhos.
- `eks`- Permite AWS Batch descrever o recurso de cluster Amazon EKS para validações.
- `iam`- Permite AWS Batch validar e passar funções fornecidas pelo proprietário para o Amazon EC2, o Amazon EC2 Auto Scaling e o Amazon ECS.
- `logs`— Permite AWS Batch criar e gerenciar grupos de registros e fluxos de registros para AWS Batch trabalhos.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSBatchPolicyStatement1",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeAccountAttributes",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeImages",
                "ec2:DescribeImageAttribute",
                "ec2:DescribeSpotInstanceRequests",
                "ec2:DescribeSpotFleetInstances",
                "ec2:DescribeSpotFleetRequests",
```

```
                "ec2:DescribeSpotPriceHistory",
                "ec2:DescribeSpotFleetRequestHistory",
                "ec2:DescribeVpcClassicLink",
                "ec2:DescribeLaunchTemplateVersions",
                "ec2:RequestSpotFleet",
                "autoscaling:DescribeAccountLimits",
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:DescribeLaunchConfigurations",
                "autoscaling:DescribeAutoScalingInstances",
                "autoscaling:DescribeScalingActivities",
                "eks:DescribeCluster",
                "ecs:DescribeClusters",
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTaskDefinition",
                "ecs:DescribeTasks",
                "ecs:ListClusters",
                "ecs:ListContainerInstances",
                "ecs:ListTaskDefinitionFamilies",
                "ecs:ListTaskDefinitions",
                "ecs:ListTasks",
                "ecs:DeregisterTaskDefinition",
                "ecs:TagResource",
                "ecs:ListAccountSettings",
                "logs:DescribeLogGroups",
                "iam:GetInstanceProfile",
                "iam:GetRole"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AWSBatchPolicyStatement2",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/batch/job*"
        },
        {
            "Sid": "AWSBatchPolicyStatement3",
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents"
            ],
```

```
            "Resource": "arn:aws:logs:*:*:log-group:/aws/batch/job*:log-stream:*"
        },
        {
            "Sid": "AWSBatchPolicyStatement4",
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateOrUpdateTags"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement5",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn",
                        "ecs-tasks.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement6",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "spot.amazonaws.com",
                        "spotfleet.amazonaws.com",
                        "autoscaling.amazonaws.com",
                        "ecs.amazonaws.com"
```

```
                    ]
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement7",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateLaunchTemplate"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement8",
            "Effect": "Allow",
            "Action": [
                "ec2:TerminateInstances",
                "ec2:CancelSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "ec2:DeleteLaunchTemplate"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement9",
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateLaunchConfiguration",
                "autoscaling:DeleteLaunchConfiguration"
            ],
            "Resource":
"arn:aws:autoscaling:*:*:launchConfiguration:*:launchConfigurationName/AWSBatch*"
        },
        {
```

```
            "Sid": "AWSBatchPolicyStatement10",
            "Effect": "Allow",
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:SetDesiredCapacity",
                "autoscaling:DeleteAutoScalingGroup",
                "autoscaling:SuspendProcesses",
                "autoscaling:PutNotificationConfiguration",
                "autoscaling:TerminateInstanceInAutoScalingGroup"
            ],
            "Resource":
 "arn:aws:autoscaling:*:*:autoScalingGroup:*:autoScalingGroupName/AWSBatch*"
        },
        {
            "Sid": "AWSBatchPolicyStatement11",
            "Effect": "Allow",
            "Action": [
                "ecs:DeleteCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:StopTask"
            ],
            "Resource": "arn:aws:ecs:*:*:cluster/AWSBatch*"
        },
        {
            "Sid": "AWSBatchPolicyStatement12",
            "Effect": "Allow",
            "Action": [
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:StopTask"
            ],
            "Resource": "arn:aws:ecs:*:*:task-definition/*"
        },
        {
            "Sid": "AWSBatchPolicyStatement13",
            "Effect": "Allow",
            "Action": [
                "ecs:StopTask"
            ],
            "Resource": "arn:aws:ecs:*:*:task/*/*"
        },
```

```
        {
            "Sid": "AWSBatchPolicyStatement14",
            "Effect": "Allow",
            "Action": [
                "ecs:CreateCluster",
                "ecs:RegisterTaskDefinition"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement15",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:launch-template/*",
                "arn:aws:ec2:*:*:placement-group/*",
                "arn:aws:ec2:*:*:capacity-reservation/*",
                "arn:aws:ec2:*:*:elastic-gpu/*",
                "arn:aws:elastic-inference:*:*:elastic-inference-accelerator/*",
                "arn:aws:resource-groups:*:*:group/*"
            ]
        },
        {
            "Sid": "AWSBatchPolicyStatement16",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSBatchServiceTag": "false"
                }
            }
```

```
        },
        {
            "Sid": "AWSBatchPolicyStatement17",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": [
                        "RunInstances",
                        "CreateLaunchTemplate",
                        "RequestSpotFleet"
                    ]
                }
            }
        }
    ]
}
```

AWS política gerenciada: AWSBatchServiceRolepolítica

A política de permissões de função nomeada AWSBatchServiceRole AWS Batch permite concluir as seguintes ações em recursos específicos:

A política AWSBatchServiceRolegerenciada do IAM geralmente é usada por uma função chamada AWSBatchServiceRolee inclui as seguintes permissões. Seguindo o conselho de segurança padrão de conceder privilégios mínimos, a política AWSBatchServiceRolegerenciada pode ser usada como um guia. Se qualquer permissão concedida na política gerenciada não for necessária para o seu caso de uso, crie uma política personalizada e adicione apenas as permissões necessárias. Essa política e função AWS Batch gerenciadas podem ser usadas com a maioria dos tipos de ambiente computacional, mas o uso de funções vinculadas a serviços é preferível para uma less-error-prone experiência gerenciada com melhor escopo e aprimorada.

- `autoscaling`— Permite AWS Batch criar e gerenciar recursos do Amazon EC2 Auto Scaling. AWS Batch cria e gerencia grupos do Amazon EC2 Auto Scaling para a maioria dos ambientes computacionais.

- `ec2`— Permite AWS Batch gerenciar o ciclo de vida das instâncias do Amazon EC2, bem como criar e gerenciar modelos e tags de lançamento. AWS Batch cria e gerencia solicitações do EC2 Spot Fleet para alguns ambientes computacionais EC2 Spot.
- `ecs`- Permite AWS Batch criar e gerenciar clusters, definições de tarefas e tarefas do Amazon ECS para execução de trabalhos.
- `iam`- Permite AWS Batch validar e passar funções fornecidas pelo proprietário para o Amazon EC2, o Amazon EC2 Auto Scaling e o Amazon ECS.
- `logs`— Permite AWS Batch criar e gerenciar grupos de registros e fluxos de registros para AWS Batch trabalhos.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AWSBatchPolicyStatement1",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeAccountAttributes",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeImages",
                "ec2:DescribeImageAttribute",
                "ec2:DescribeSpotInstanceRequests",
                "ec2:DescribeSpotFleetInstances",
                "ec2:DescribeSpotFleetRequests",
                "ec2:DescribeSpotPriceHistory",
                "ec2:DescribeSpotFleetRequestHistory",
                "ec2:DescribeVpcClassicLink",
                "ec2:DescribeLaunchTemplateVersions",
                "ec2:CreateLaunchTemplate",
                "ec2:DeleteLaunchTemplate",
                "ec2:RequestSpotFleet",
                "ec2:CancelSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "ec2:TerminateInstances",
                "ec2:RunInstances",
```

```
                "autoscaling:DescribeAccountLimits",
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:DescribeLaunchConfigurations",
                "autoscaling:DescribeAutoScalingInstances",
                "autoscaling:DescribeScalingActivities",
                "autoscaling:CreateLaunchConfiguration",
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:SetDesiredCapacity",
                "autoscaling:DeleteLaunchConfiguration",
                "autoscaling:DeleteAutoScalingGroup",
                "autoscaling:CreateOrUpdateTags",
                "autoscaling:SuspendProcesses",
                "autoscaling:PutNotificationConfiguration",
                "autoscaling:TerminateInstanceInAutoScalingGroup",
                "ecs:DescribeClusters",
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTaskDefinition",
                "ecs:DescribeTasks",
                "ecs:ListAccountSettings",
                "ecs:ListClusters",
                "ecs:ListContainerInstances",
                "ecs:ListTaskDefinitionFamilies",
                "ecs:ListTaskDefinitions",
                "ecs:ListTasks",
                "ecs:CreateCluster",
                "ecs:DeleteCluster",
                "ecs:RegisterTaskDefinition",
                "ecs:DeregisterTaskDefinition",
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:UpdateContainerAgent",
                "ecs:DeregisterContainerInstance",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogGroups",
                "iam:GetInstanceProfile",
                "iam:GetRole"
            ],
            "Resource": "*"
        },
        {
```

```
            "Sid": "AWSBatchPolicyStatement2",
            "Effect": "Allow",
            "Action": "ecs:TagResource",
            "Resource": [
                "arn:aws:ecs:*:*:task/*_Batch_*"
            ]
        },
        {
            "Sid": "AWSBatchPolicyStatement3",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn",
                        "ecs-tasks.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement4",
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "spot.amazonaws.com",
                        "spotfleet.amazonaws.com",
                        "autoscaling.amazonaws.com",
                        "ecs.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "AWSBatchPolicyStatement5",
            "Effect": "Allow",
            "Action": [
```

```
                "ec2:CreateTags"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "RunInstances"
                }
            }
        }
    ]
}
```

AWS política gerenciada: AWSBatchFullAccess

A AWSBatchFullAccesspolítica concede AWS Batch às ações acesso total aos AWS Batch recursos. Também concede acesso para descrever e listar ações para serviços Amazon EC2, Amazon ECS, Amazon EKS e CloudWatch IAM. Isso é para que as identidades do IAM, sejam usuários ou funções, possam visualizar os recursos AWS Batch gerenciados que foram criados em seu nome. Por fim, essa política também permite que funções selecionadas do IAM sejam passadas para esses serviços.

Você pode anexar AWSBatchFullAccessàs suas entidades do IAM. AWS Batch também anexa essa política a uma função de serviço que permite AWS Batch realizar ações em seu nome.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "batch:*",
        "cloudwatch:GetMetricStatistics",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeVpcs",
        "ec2:DescribeImages",
        "ec2:DescribeLaunchTemplates",
        "ec2:DescribeLaunchTemplateVersions",
        "ecs:DescribeClusters",
```

```
        "ecs:Describe*",
        "ecs:List*",
        "eks:DescribeCluster",
        "eks:ListClusters",
        "logs:Describe*",
        "logs:Get*",
        "logs:TestMetricFilter",
        "logs:FilterLogEvents",
        "iam:ListInstanceProfiles",
        "iam:ListRoles"
      ],
      "Resource":"*"
    },
    {
      "Effect":"Allow",
      "Action":[
        "iam:PassRole"
      ],
      "Resource":[
        "arn:aws:iam::*:role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/service-role/AWSBatchServiceRole",
        "arn:aws:iam::*:role/ecsInstanceRole",
        "arn:aws:iam::*:instance-profile/ecsInstanceRole",
        "arn:aws:iam::*:role/iaws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/aws-ec2-spot-fleet-role",
        "arn:aws:iam::*:role/AWSBatchJobRole*"
      ]
    },
    {
      "Effect":"Allow",
      "Action":[
        "iam:CreateServiceLinkedRole"
      ],
      "Resource":"arn:aws:iam::*:role/*Batch*",
      "Condition": {
        "StringEquals": {
          "iam:AWSServiceName": "batch.amazonaws.com"
          }
      }
    }
  ]
}
```

AWS Batch atualizações nas políticas AWS gerenciadas

Veja detalhes sobre as atualizações das políticas AWS gerenciadas AWS Batch desde que esse serviço começou a rastrear essas alterações. Para receber alertas automáticos sobre alterações nessa página, assine o feed RSS na página Histórico do AWS Batch documento.

Alteração	Descrição	Data
BatchServiceRolePolicypolítica atualizada	Atualizado para adicionar suporte para descrever o histórico e as Amazon EC2 Auto Scaling atividades de solicitações do Spot Fleet.	5 de dezembro de 2023
AWSBatchServiceRolepolítica adicionada	Atualizado para adicionar IDs de declaração, conceder AWS Batch permissões para `ec2:DescribeSpotFleetRequestHistory` `autoscaling:DescribeScalingActivities` e.	5 de dezembro de 2023
BatchServiceRolePolicypolítica atualizada	Atualizada para adicionar suporte à descrição de clusters do Amazon EKS.	20 de outubro de 2022
AWSBatchFullAccesspolítica atualizada	Atualizada para adicionar suporte a listagem e descrição de clusters do Amazon EKS.	20 de outubro de 2022
BatchServiceRolePolicypolítica atualizada	Atualizada para suportar grupos de reserva de capacidade do Amazon EC2 gerenciados por AWS Resource Groups. Para obter mais informações, consulte Trabalhando com Grupos de Reserva de Capacidade no Manual do Usuário do Amazon EC2 para instâncias do Linux.	18 de maio de 2022

Alteração	Descrição	Data
BatchServiceRolePolicye AWSBatchServiceRolepolíticas atualizadas	Atualizado para adicionar suporte para descrever o status das instâncias AWS Batch gerenciadas no Amazon EC2 para que as instâncias não íntegras sejam substituídas.	6 de dezembro de 2021
BatchServiceRolePolicypolítica atualizada	Atualizada para adicionar suporte para grupos de posicionamento, reserva de capacidade, GPU elástica e recursos do Elastic Inference no Amazon EC2.	26 de março de 2021
BatchServiceRolePolicypolítica adicionada	Com a política BatchServiceRolePolicygerenciada para a função AWSServiceRoleForBatchvinculada ao serviço, você pode usar uma função vinculada ao serviço gerenciada por. AWS Batch Com essa política, você não precisa manter sua própria função para uso em ambientes computacionais.	10 de março de 2021
AWSBatchFullAccess- adicionar permissão para adicionar uma função vinculada ao serviço	Adicione permissões do IAM para permitir que a função AWSServiceRoleForBatchvinculada ao serviço seja adicionada à conta.	10 de março de 2021
AWS Batch começou a rastrear alterações	AWS Batch começou a rastrear as mudanças em suas políticas AWS gerenciadas.	10 de março de 2021

Acesso AWS Batch usando um endpoint de interface

Você pode usar AWS PrivateLink para criar uma conexão privada entre sua VPC e. AWS Batch Você pode acessar como se estivesse em sua VPC, sem o uso de gateway da internet, dispositivo NAT, conexão VPN ou conexão AWS Direct Connect . As instâncias na sua VPC não precisam de endereços IP públicos para acessar AWS Batch.

Você estabelece essa conectividade privada criando um endpoint de interface, desenvolvido pelo AWS PrivateLink. Criaremos um endpoint de interface de rede em cada sub-rede que você habilitar para o endpoint de interface. Estas são interfaces de rede gerenciadas pelo solicitante que servem como ponto de entrada para o tráfego destinado ao AWS Batch.

Para obter mais informações, consulte, consulte VPC Endpoints de Interface no Guia AWS PrivateLink .

Considerações para AWS Batch

Antes de configurar um endpoint de interface para AWS Batch, revise as propriedades e limitações do endpoint de interface no AWS PrivateLink Guia.

AWS Batch suporta fazer chamadas para todas as suas ações de API por meio do endpoint da interface.

Antes de configurar a interface para VPC endpoints AWS Batch, esteja ciente das seguintes considerações:

- Trabalhos usando o tipo de lançamento de recursos Fargate não exigem a interface VPC endpoints para Amazon ECS, mas você pode precisar de endpoints VPC de interface para Amazon ECR, Secrets Manager ou AWS Batch Amazon Logs descritos nos pontos a seguir. CloudWatch
  - Para executar trabalhos, você deve criar VPC endpoints de interface para o Amazon ECS. Para obter mais informações, consulte VPC Endpoints de Interface (AWS PrivateLink) no Guia do desenvolvedor do Amazon Elastic Container Service.
  - Para permitir que seus trabalhos extraiam imagens privadas do Amazon ECR, você deve criar VPC endpoints de interface para o Amazon ECR. Para obter mais informações, consulte VPC Endpoints de interface (AWS PrivateLink) no Manual do Usuário do Amazon Elastic Container Registry.
  - Para permitir que seus trabalhos extraiam dados sigilosos do Secrets Manager, é necessário criar VPC endpoints de interface para o Secrets Manager. Para obter mais informações, consulte Usando o Secrets Manager com Endpoints da VPC no AWS Secrets Manager Manual do Usuário do.
  - Se sua VPC não tiver um gateway de internet e seus trabalhos usarem o driver de `awslogs` log para enviar informações de log para CloudWatch Logs, você deverá criar uma interface VPC endpoint para Logs. CloudWatch Para obter mais informações, consulte Como usar CloudWatch registros com endpoints VPC de interface no Guia do usuário do Amazon CloudWatch Logs.

- Os trabalhos que usam os requisitos dos recursos do EC2 exigem que as instâncias de contêiner nas quais forem iniciadas executem a versão `1.25.1` ou superior do agente de contêiner do Amazon ECS. Para obter mais informações, consulte Versões do Agente de Contêiner do Amazon ECS no Guia do Desenvolvedor do Amazon Elastic Container Service.
- Atualmente, os endpoints da VPC não oferecem suporte a solicitações entre Regiões. Garanta a criação do seu endpoint na mesma Região onde planeja emitir as chamadas de API para o AWS Batch.
- Os endpoints da VPC oferecem suporte somente a DNS fornecidos pela Amazon por meio do Amazon Route 53. Se quiser usar seu próprio DNS, você pode usar o encaminhamento de DNS condicional. Para obter mais informações, consulte Conjuntos de Opções de DHCP no Manual do Usuário da Amazon VPC.
- O grupo de segurança anexado ao endpoint da VPC deve permitir entrada conectada a porta 443 na sub-rede privada da VPC.
- AWS Batch não oferece suporte a endpoints de interface VPC no seguinte: Regiões da AWS
  - Asia Pacific (Osaka) (`ap-northeast-3`)
  - Ásia-Pacífico (Jacarta) (`ap-southeast-3`)

Crie um endpoint de interface para AWS Batch

Você pode criar um endpoint de interface para AWS Batch usar o console Amazon VPC ou AWS Command Line Interface o AWS CLI(). Para obter mais informações, consulte Criar um endpoint de interface no Guia do usuário do AWS PrivateLink .

Crie um endpoint de interface para AWS Batch usar o seguinte nome de serviço:

```
com.amazonaws.region.batch
```

Por exemplo: .

```
com.amazonaws.us-east-2.batch
```

Na partição `aws-cn`, o formato é diferente:

```
cn.com.amazonaws.region.batch
```

Por exemplo: .

```
cn.com.amazonaws.cn-northwest-1.batch
```

Se você habilitar o DNS privado para o endpoint da interface, poderá fazer solicitações de API AWS Batch usando seu nome DNS regional padrão. Por exemplo, `batch.us-east-1.amazonaws.com`.

Para obter mais informações, consulte Acessando um Serviço por meio de um Endpoint de Interface no AWS PrivateLink Guia .

Criar uma política de endpoint para o endpoint da interface

Política de endpoint é um recurso do IAM que você pode anexar ao endpoint de interface. A política de endpoint padrão permite acesso total AWS Batch por meio do endpoint da interface. Para controlar o acesso permitido ao AWS Batch pela VPC, anexe uma política de endpoint personalizada ao endpoint de interface.

Uma política de endpoint especifica as seguintes informações:

- As entidades principais que podem executar ações (Contas da AWS, usuários e funções do IAM).
- As ações que podem ser executadas.
- Os recursos nos quais as ações podem ser executadas.

Para obter mais informações, consulte Controlar o Acesso a Serviços Usando Políticas de Endpoint no AWS PrivateLink Guia.

Exemplo: política de VPC endpoint para ações AWS Batch

O exemplo a seguir refere-se a uma política de endpoint personalizada. Quando você anexa essa política ao seu endpoint de interface, ela concede acesso às AWS Batch ações listadas para todos os diretores em todos os recursos.

```
{
   "Statement": [
      {
         "Principal": "*",
         "Effect": "Allow",
         "Action": [
            "batch:SubmitJob",
            "batch:ListJobs",
            "batch:DescribeJobs"
         ],
```

```
            "Resource":"*"
        }
    ]
}
```

Validação de conformidade para AWS Batch

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

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

Sua responsabilidade de conformidade ao usar Serviços da AWS é determinada pela confidencialidade de seus dados, pelos objetivos de conformidade de sua empresa e pelas leis e regulamentações aplicáveis. AWS fornece os seguintes recursos para ajudar na conformidade:

- Guias de início rápido sobre segurança e conformidade — Esses guias de implantação discutem considerações arquitetônicas e fornecem etapas para a implantação de ambientes básicos AWS focados em segurança e conformidade.
- Arquitetura para segurança e conformidade com a HIPAA na Amazon Web Services — Este whitepaper descreve como as empresas podem usar AWS para criar aplicativos qualificados para a HIPAA.

  > **Note**
  >
  > Nem todos Serviços da AWS são elegíveis para a HIPAA. Para obter mais informações, consulte a Referência dos serviços qualificados pela HIPAA.

- AWS Recursos de https://aws.amazon.com/compliance/resources/ de conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.
- AWS Guias de conformidade do cliente — Entenda o modelo de responsabilidade compartilhada sob a ótica da conformidade. Os guias resumem as melhores práticas de proteção Serviços da AWS e mapeiam as diretrizes para controles de segurança em várias estruturas (incluindo o Instituto Nacional de Padrões e Tecnologia (NIST), o Conselho de Padrões de Segurança do Setor de Cartões de Pagamento (PCI) e a Organização Internacional de Padronização (ISO)).

- Avaliação de recursos com regras no Guia do AWS Config desenvolvedor — O AWS Config serviço avalia o quão bem suas configurações de recursos estão em conformidade com as práticas internas, as diretrizes e os regulamentos do setor.
- AWS Security Hub— Isso AWS service (Serviço da AWS) fornece uma visão abrangente do seu estado de segurança interno AWS. O Security Hub usa controles de segurança para avaliar os recursos da AWS e verificar a conformidade com os padrões e as práticas recomendadas do setor de segurança. Para obter uma lista dos serviços e controles aceitos, consulte a Referência de controles do Security Hub.
- AWS Audit Manager— Isso AWS service (Serviço da AWS) ajuda você a auditar continuamente seu AWS uso para simplificar a forma como você gerencia o risco e a conformidade com as regulamentações e os padrões do setor.

Segurança de infraestrutura em AWS Batch

Como serviço gerenciado, AWS Batch é protegido pela segurança de rede AWS global. Para obter informações sobre serviços AWS de segurança e como AWS proteger a infraestrutura, consulte AWS Cloud Security. Para projetar seu AWS ambiente usando as melhores práticas de segurança de infraestrutura, consulte Proteção de infraestrutura no Security Pillar AWS Well-Architected Framework.

Você usa chamadas de API AWS publicadas para acessar AWS Batch pela rede. Os clientes devem oferecer suporte para:

- Transport Layer Security (TLS). Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Conjuntos de criptografia com sigilo de encaminhamento perfeito (perfect forward secrecy, ou PFS) como DHE (Ephemeral Diffie-Hellman, ou Efêmero Diffie-Hellman) ou ECDHE (Ephemeral Elliptic Curve Diffie-Hellman, ou Curva elíptica efêmera Diffie-Hellman). A maioria dos sistemas modernos, como Java 7 e versões posteriores, comporta esses modos.

Além disso, as solicitações devem ser assinadas utilizando um ID da chave de acesso e uma chave de acesso secreta associada a uma entidade principal do IAM. Ou é possível usar o AWS Security Token Service (AWS STS) para gerar credenciais de segurança temporárias para assinar solicitações.

Você pode chamar essas operações de API de qualquer local de rede, mas AWS Batch oferece suporte a políticas de acesso baseadas em recursos, que podem incluir restrições com base no

endereço IP de origem. Você também pode usar AWS Batch políticas para controlar o acesso de endpoints específicos da Amazon Virtual Private Cloud (Amazon VPC) ou VPCs específicas. Efetivamente, isso isola o acesso à rede a um determinado AWS Batch recurso somente da VPC específica dentro da AWS rede.

Marcando seus Recursos AWS Batch

Para ajudar no gerenciamento de recursos AWS Batch, você pode atribuir seus próprios metadados a cada recurso em forma de tags. Este tópico descreve as tags e como criá-las.

Índice

Conceitos Básicos de Tags

Uma tag um rótulo atribuído a um recurso AWS. Cada tag consiste em uma chave e um valor opcional, ambos definidos por você.

As tags permitem categorizar seus recursos AWS por finalidade, proprietário ou ambiente, por exemplo. Caso possua muitos recursos do mesmo tipo, você pode identificar rapidamente um recurso específico com base nas tags atribuídas a ele. Por exemplo, é possível definir um conjunto de tags para seus serviços AWS Batch para ajudá-lo a rastrear o proprietário e nível da pilha de cada serviço. Recomendamos planejar um conjunto consistente de chaves de tags para cada tipo de recurso.

Tags não são automaticamente atribuídas aos recursos. Após adicionar uma tag, você pode editar as chaves e os valores das tags ou removê-las de um recurso a qualquer momento. Caso exclua um recurso, todas as respectivas tags também serão excluídas.

As tags não têm significado semântico atrelado a AWS Batch e são interpretadas estritamente como string de caracteres. É possível definir o valor de uma tag em uma string vazia, mas não configurar o valor de um tag como nula. Caso adicione uma tag com a mesma chave de outra existente no recurso, o novo valor substituirá o antigo.

Você pode trabalhar com tags usando AWS Management Console, AWS CLI e API de AWS Batch.

Caso esteja usando AWS Identity and Access Management (IAM), você pode controlar quais usuários em sua conta AWS têm permissão para criar, editar ou excluir tags.

Marcando seus Recursos

Você pode marcar ambientes de computação AWS Batch, tarefas, definições de tarefas, filas de tarefas, políticas de agendamento novas ou existentes.

Caso esteja usando o console AWS Batch, você pode aplicar tags a novos recursos quando estes forem criados ou a recursos existentes a qualquer momento, por meio da guia Tags na página de recursos relevante.

Caso esteja usando a API AWS Batch, AWS CLI ou SDK AWS, é possível aplicar tags a novos recursos por meio do parâmetro `tags` na ação API relevante ou, para recursos existentes, da ação API `TagResource`. Para mais informações, consulte TagResource.

Algumas ações de criação de recursos permitem especificar tags para um recurso quando o mesmo for criado. Caso as tags não possam ser aplicadas durante a criação dos recursos, haverá falha no processo de criação de recursos. Isso garante que recursos que você pretenda marcar na criação sejam criados com as tags especificadas ou não. Caso marque recursos no momento da criação, não precisará executar scripts de marcação personalizados após a criação do recurso.

A tabela a seguir descreve os recursos AWS Batch que podem ser marcados com tags e aqueles que podem ser marcados na criação.

Suporte à marcação para recursos AWS Batch

Recurso	Compatível com tags	Compatível com a propagação de tags	Compatível com o uso de tags na criação (API AWS Batch, AWS CLI e SDK AWS)
Ambientes de computação AWS Batch	Sim	Não. As tags de ambiente de computação não são propagadas para nenhum outro recurso. Tags para recursos são especificadas nas tags membros do	Sim

Recurso	Compatível com tags	Compatível com a propagação de tags	Compatível com o uso de tags na criação (API AWS Batch, AWS CLI e SDK AWS)
		objeto ComputerResources transmitidas na operação API CreateComputeEnvironment.	
Tarefas AWS Batch	Sim	Sim	Sim
Definições de trabalho AWS Batch	Sim	Não	Sim
Filas de tarefa AWS Batch	Sim	Não	Sim
Políticas de agendamento AWS Batch	Sim	Não	Sim

Restrições de tag

As restrições básicas a seguir se aplicam a tags:

- Número máximo de tags por recurso –- 50
- Em todos os recursos, cada chave de tag deve ser exclusiva e possuir apenas um valor.
- Comprimento máximo da chave –- 128 caracteres Unicode em UTF-8
- Comprimento máximo do valor –- 256 caracteres Unicode em UTF-8
- Caso seu esquema de marcação seja usado em vários serviços e recursos AWS, lembre-se de que outros serviços podem possuir restrições em caracteres permitidos. Em geral, caracteres permitidos incluem letras, números, espaços representáveis em UTF-8 e os caracteres + - = . _ : / @.
- Chaves e valores de tags diferenciam maiúsculas de minúsculas.

- Não use `aws:`, `AWS:` ou qualquer combinação de letras maiúsculas e minúsculas como um prefixo para chaves ou valores, uma vez que as mesmas são reservadas para uso AWS. Você não pode editar nem excluir chaves ou valores de tags com esse prefixo. Tags com esse prefixo não contam em limites de tags por recurso.

Trabalhando com tags usando o console

Ao utilizar o console do AWS Batch, você gerencia as tags associadas a ambientes de computação, tarefas, definições de tarefas e filas de tarefas novas ou existentes.

Adicionar tags a um recurso individual na criação

Você pode adicionar tags a ambientes de computação AWS Batch, tarefas, definições de tarefas, filas de tarefas e políticas de agendamento ao criá-las.

Adicionando e excluindo tags em um recurso individual

AWS Batch permite adicionar ou excluir tags associadas a clusters diretamente na página do recurso.

Para adicionar ou excluir uma etiqueta em um recurso individual

1. Abra o console AWS Batch em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, escolha a Região a ser usada.
3. No painel de navegação, escolha um tipo de recurso (por exemplo, Filas de Tarefas).
4. Escolha um recurso específico e, em seguida, Editar Tags.
5. Adicione ou exclua tags conforme necessário.
   - Para adicionar uma tag, especifique a chave e o valor nas caixas de texto vazias ao final da lista.
   - Para excluir uma tag, escolha o botão

     Delete icon
     ao lado.
6. Repita esse processo para cada tag que desejar adicionar ou excluir e escolha Editar Tags para concluir.

Trabalhar com tags usando CLI ou a API

Use os seguintes comandos AWS CLI ou operações API AWS Batch para adicionar, atualizar, listar e excluir as tags de seus recursos.

Suporte à marcação para recursos AWS Batch

Tarefa	Ação API	AWS CLI	AWS Tools for Windows PowerShell
Adicione ou sobrescreva uma ou mais tags.	TagResource	tag-resource	Add-BATResourceTag
Exclua uma ou mais tags.	UntagResource	untag-resource	Remove-BATResourceTag
Listar as tags para um recurso	ListTagsForResource	list-tags-for-resource	Get-BATResourceTag

Os exemplos a seguir mostram como marcar ou desmarcar recursos usando AWS CLI.

Exemplo 1: marcar um recurso existente

O comando a seguir marca um recurso existente.

```
aws batch tag-resource --resource-arn resource_ARN --tags team=devs
```

Exemplo 2: desmarcar um recurso existente

O comando a seguir exclui uma tag de um recurso existente.

```
aws batch untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

Exemplo 3: listar etiquetas para um recurso

O comando a seguir lista as tags associadas a um recurso existente.

```
aws batch list-tags-for-resource --resource-arn resource_ARN
```

Algumas ações de criação de recursos permitem especificar tags ao criar o recurso. As ações a seguir são compatíveis com o uso de tags na criação.

Tarefa	Ação API	AWS CLI	AWS Tools for Windows PowerShell
Criar um ambiente de computação	CreateComputeEnvironment	create-compute-environment	New-BATComputeEnvironment
Criar uma fila de tarefas	CreateJobQueue	create-job-queue	New-BATJobQueue
Criar uma política de agendamento	Criar política de agendamento	create-scheduling-policy	New-BATSchedulingPolicy
Registrar uma definição de tarefa	RegisterJobDefinition	register-job-definition	Register-BATJobDefinition
Enviar uma tarefa	SubmitJob	submit-job	Submit-BATJob

Cotas de serviço do AWS Batch

A tabela a seguir fornece as cotas de serviço para o AWS Batch que não podem ser alteradas. Cada cota é específica da região.

Recurso	Quota
Número máximo de filas de trabalho. Para obter mais informações, consulte Filas de tarefas.	50
Número máximo de ambientes de computação no Amazon ECS e no Amazon EKS. Para obter mais informações, consulte Ambiente de computação.	50
Número máximo de ambientes de computação por cluster do Amazon EKS.	5
Número máximo de ambientes de computação para cada fila de trabalho	3
Número máximo de dependências de trabalho para um trabalho	20
O tamanho máximo de definição de trabalho (para operações da API `RegisterJobDefinition`)	24 KiB
O tamanho máximo da carga da tarefa (para operações da API `SubmitJob`)	30 KiB
O tamanho máximo da matriz para trabalhos de matriz	10000
Número máximo de trabalhos no estado `SUBMITTED`	1000000
Número máximo de transações por segundo (TPS) para cada conta para operações `SubmitJob`	50

Dependendo de como você usa o AWS Batch, podem ser aplicadas cotas adicionais. Para saber mais sobre as cotas do Amazon EC2, consulte Amazon EC2 Service Quotas no Referência geral da AWS. Para obter mais informações sobre as cotas do Amazon ECS, consulte Amazon ECS Service Quotas no Referência geral da AWS. Para obter mais informações sobre as cotas do Amazon EKS, consulte Amazon EKS Service Quotas no Referência geral da AWS.

Solução de problemas AWS Batch

Você talvez precise resolver problemas relacionados a ambientes de computação, filas de trabalho, definições de trabalho ou trabalhos. Este capítulo descreve como solucionar e resolver esses problemas em seu AWS Batch ambiente.

AWS Batch usa políticas, funções e permissões do IAM e é executado na infraestrutura Amazon EC2, Amazon ECS e Amazon Elastic AWS Fargate Kubernetes Service. Para solucionar problemas relacionados a esses serviços, consulte o seguinte:

- Solução de problemas do IAM no Guia do Usuário do IAM
- Solução de problema do Amazon ECS no Guia do Desenvolvedor do Amazon Elastic Container Service
- Solução de problemas do Amazon EKS no Guia do Usuário do Amazon EKS
- Solucionar problemas de instâncias do EC2 no Guia do Usuário do Amazon EC2 para instâncias Linux

Sumário

AWS Batch

Ambiente de computação do **`INVALID`**

É possível que você tenha configurado incorretamente um ambiente de computação gerenciado. Se você fez isso, o ambiente de computação entrará em um estado `INVALID` e não poderá aceitar trabalhos para colocação. As seções a seguir descrevem as possíveis causas e como solucionar problemas com base na causa.

Nome do perfil ou ARN incorreto

A causa mais comum de um ambiente computacional entrar em um `INVALID` estado é que a função de AWS Batch serviço ou a função de frota spot do Amazon EC2 tem um nome ou nome de recurso da Amazon (ARN) incorreto. Isso é mais comum em ambientes computacionais criados usando os SDKs AWS CLI ou os AWS SDKs. Quando você cria um ambiente computacional no AWS Management Console, AWS Batch ajuda você a escolher o serviço correto ou as funções do Spot Fleet. No entanto, suponha que você insira manualmente o nome ou o ARN e os insira incorretamente. Então, o ambiente de computação resultante também é `INVALID`.

No entanto, suponha que você insira manualmente o nome ou o ARN de um recurso do IAM em um AWS CLI comando ou no código do SDK. Nesse caso, não é AWS Batch possível validar a string. Em vez disso, AWS Batch deve aceitar o valor ruim e tentar criar o ambiente. Se AWS Batch não conseguir criar o ambiente, o ambiente será movido para um `INVALID` estado e você verá os seguintes erros.

Para um perfil de serviços inválida:

```
CLIENT_ERROR - Not authorized to perform sts:AssumeRole (Service:
AWSSecurityTokenService; Status Code: 403; Error Code: AccessDenied;
Request ID: dc0e2d28-2e99-11e7-b372-7fcc6fb65fe7)
```

Para um perfil de Spot Fleet inválida:

```
CLIENT_ERROR - Parameter: SpotFleetRequestConfig.IamFleetRole
is invalid. (Service: AmazonEC2; Status Code: 400; Error Code:
InvalidSpotFleetRequestConfig; Request ID: 331205f0-5ae3-4cea-
bac4-897769639f8d) Parameter: SpotFleetRequestConfig.IamFleetRole is
invalid
```

Uma causa comum para esse problema é o cenário a seguir. Você só especifica o nome de uma função do IAM ao usar o AWS CLI ou os AWS SDKs, em vez do Amazon Resource Name (ARN) completo. Dependendo de como você criou o perfil, o ARN poderá conter o prefixo de caminho `aws-service-role`. Por exemplo, se você criar manualmente o perfil de serviço do AWS Batch usando os procedimentos em Usando funções vinculadas a serviços para AWS Batch, o ARN do perfil de serviço deverá ser semelhante ao seguinte.

```
arn:aws:iam::123456789012:role/AWSBatchServiceRole
```

No entanto, se você criar o perfil de serviço como parte da primeira execução do assistente do console hoje, o ARN do perfil de serviço terá a seguinte aparência.

```
arn:aws:iam::123456789012:role/aws-service-role/AWSBatchServiceRole
```

Esse problema também pode ocorrer se você anexar a política de AWS Batch nível de serviço (`AWSBatchServiceRole`) a uma função que não seja de serviço. Por exemplo, você pode receber uma mensagem de erro semelhante à seguinte neste cenário:

```
CLIENT_ERROR - User: arn:aws:sts::account_number:assumed-role/batch-replacement-role/
aws-batch is not
     authorized to perform: action on resource ...
```

Para resolver esse problema, execute um dos seguintes procedimentos:

- Use uma string vazia para a função de serviço ao criar o ambiente AWS Batch computacional.
- Especifique o perfil de serviço no seguinte formato: `arn:aws:iam::account_number:role/aws-service-role/batch.amazonaws.com/AWSServiceRoleForBatch`.

Quando você especifica apenas o nome de uma função do IAM ao usar o AWS CLI ou os AWS SDKs, AWS Batch presume que seu ARN não usa o prefixo do caminho. `aws-service-role` Por isso, recomendamos que você especifique o ARN completo para seu perfil do IAM; ao criar ambientes de computação.

Para reparar um ambiente de computação que é configurado incorretamente dessa forma, consulte Reparo de um `INVALID` ambiente de computação.

Reparo de um **`INVALID`** ambiente de computação

Quando você tem um ambiente de computação em um estado `INVALID`, deve atualizá-lo para reparar o parâmetro inválido. Para um Nome do perfil ou ARN incorreto, você pode atualizar o ambiente de computação com o perfil de serviço correto.

Para reparar um ambiente de computação configurado incorretamente

1. Abra o AWS Batch console em https://console.aws.amazon.com/batch/.
2. Na barra de navegação, selecione o Região da AWS a ser usado.
3. No painel de navegação, escolha Ambientes de computação.
4. Na página Ambientes de computação, selecione o botão próximo ao ambiente de computação a ser editado e, em seguida, escolha Edit.
5. Na página Atualizar ambiente de computação, para Perfil de serviço, escolha perfil do IAM a ser usado com seu ambiente de computação. O console AWS Batch exibe somente funções que têm a relação de confiança correta para ambientes de computação.
6. Escolha Salvar para atualizar seu ambiente de computação.

Trabalhos presos no status **`RUNNABLE`**

Suponha que seu ambiente de computação contenha recursos computacionais, mas seus trabalhos não progridam além do status `RUNNABLE` . Então, é provável que algo esteja impedindo que os trabalhos sejam colocados em um recurso computacional e fazendo com que suas filas de trabalhos sejam bloqueadas. Veja como saber se seu trabalho está aguardando sua vez ou travando e bloqueando a fila.

Se AWS Batch detectar que você tem um `RUNNABLE` trabalho à frente e está bloqueando a fila, você receberá um evento de fila de trabalhos bloqueado da Amazon CloudWatch Events com o motivo. O mesmo motivo também é atualizado no `statusReason` campo como parte das chamadas `ListJobs` de `DescribeJobs` API.

Opcionalmente, você pode configurar o `jobStateTimeLimitActions` parâmetro por meio de ações `CreateJobQueue` de `UpdateJobQueue`API.

> **Note**
>
> Atualmente, a única ação que você pode usar `jobStateLimitActions.action` é cancelar um trabalho.

O `jobStateTimeLimitActions` parâmetro é usado para especificar um conjunto de ações que são AWS Batch executadas em trabalhos em um estado específico. Você pode definir um limite de tempo em segundos por meio do `maxTimeSeconds` campo.

Quando um trabalho está em um `RUNNABLE` estado com o definido`statusReason`, AWS Batch executa a ação especificada após `maxTimeSeconds` ter decorrido.

Por exemplo, você pode definir o `jobStateTimeLimitActions` parâmetro para aguardar até 4 horas por qualquer trabalho no `RUNNABLE` estado que esteja aguardando a disponibilidade de capacidade suficiente. Você pode fazer isso configurando `statusReason` `maxTimeSeconds` para `CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY` e como 144000 antes de cancelar o trabalho e permitir que o próximo trabalho avance para o início da fila de trabalhos.

A seguir estão os motivos que ocorrem AWS Batch quando ele detecta que uma fila de trabalhos está bloqueada. Essa lista fornece as mensagens retornadas das ações `ListJobs` e `DescribeJobs` da API. Esses também são os mesmos valores que você pode definir para o `jobStateLimitActions.statusReason` parâmetro.

1. Motivo: todos os ambientes computacionais conectados têm erros de capacidade insuficientes. Quando solicitado, AWS Batch detecta instâncias do Amazon EC2 que apresentam erros de capacidade insuficientes. Cancelar o trabalho, manualmente ou definindo o `jobStateTimeLimitActions` parâmetro ativado`statusReason`, permite que o trabalho subsequente seja movido para o início da fila.
   - **`statusReason`**mensagem enquanto o trabalho está travado: `CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY - Service cannot fulfill the capacity requested for instance type [instanceTypeName]`
   - **`reason`**usado para**`jobStateTimeLimitActions`**: `CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY`

- **`statusReason`**mensagem após o cancelamento do trabalho:
  `Canceled by JobStateTimeLimit action due to reason:
  CAPACITY:INSUFFICIENT_INSTANCE_CAPACITY`

  Observações:

  a. A função AWS Batch de serviço requer `autoscaling:DescribeScalingActivities` permissão para que essa detecção funcione. Se você usa a função `AWSServiceRoleForBatch`vinculada ao serviço (SLR) ou a política `AWSBatchServiceRolePolicy`gerenciada, não precisa realizar nenhuma ação porque suas políticas de permissão estão atualizadas.

  b. Se você usar a SLR ou a política gerenciada, deverá adicionar as `ec2:DescribeSpotFleetRequestHistory` permissões `autoscaling:DescribeScalingActivities` e para poder receber eventos bloqueados da fila de trabalhos e status de trabalho atualizado quando estiver em casa. `RUNNABLE` Além disso, AWS Batch precisa dessas permissões para realizar `cancellation` ações por meio do `jobStateTimeLimitActions` parâmetro, mesmo que estejam configuradas na fila de trabalhos.

  c. No caso de um trabalho paralelo de vários nós (MNP), se o ambiente computacional Amazon EC2 anexado de alta prioridade apresentar `insufficient capacity` erros, ele bloqueará a fila mesmo que um ambiente computacional de prioridade mais baixa apresente esse erro.

2. Motivo: todos os ambientes computacionais têm um `maxvCpus`parâmetro menor do que os requisitos do trabalho. Cancelar o trabalho, manualmente ou definindo o `jobStateTimeLimitActions` parâmetro ativado`statusReason`, permite que o trabalho subsequente seja movido para o início da fila. Opcionalmente, você pode aumentar o `maxvCpus` parâmetro do ambiente computacional primário para atender às necessidades da tarefa bloqueada.

   - **`statusReason`**mensagem enquanto o trabalho está travado:
     `MISCONFIGURATION:COMPUTE_ENVIRONMENT_MAX_RESOURCE - CE(s) associated with the job queue cannot meet the CPU requirement of the job.`
   - **`reason`**usado para**`jobStateTimeLimitActions`**:
     `MISCONFIGURATION:COMPUTE_ENVIRONMENT_MAX_RESOURCE`
   - **`statusReason`**mensagem após o cancelamento do trabalho:
     `Canceled by JobStateTimeLimit action due to reason:
     MISCONFIGURATION:COMPUTE_ENVIRONMENT_MAX_RESOURCE`

3. Motivo: nenhum dos ambientes computacionais tem instâncias que atendam aos requisitos do trabalho. Quando um trabalho solicita recursos, AWS Batch detecta que nenhum ambiente computacional conectado é capaz de acomodar o trabalho recebido. Cancelar o trabalho, manualmente ou definindo o `jobStateTimeLimitActions` parâmetro ativado`statusReason`, permite que o trabalho subsequente seja movido para o início da fila. Opcionalmente, você pode redefinir os tipos de instância permitidos do ambiente computacional para adicionar os recursos de trabalho necessários.

   - **`statusReason`**mensagem enquanto o trabalho está travado: `MISCONFIGURATION:JOB_RESOURCE_REQUIREMENT - The job resource requirement (vCPU/memory/GPU) is higher than that can be met by the CE(s) attached to the job queue.`
   - **`reason`**usado para**`jobStateTimeLimitActions`**: `MISCONFIGURATION:JOB_RESOURCE_REQUIREMENT`
   - **`statusReason`**mensagem após o cancelamento do trabalho: `Canceled by JobStateTimeLimit action due to reason: MISCONFIGURATION:JOB_RESOURCE_REQUIREMENT`

4. Motivo: todos os ambientes computacionais têm problemas de função de serviço. Para resolver isso, compare suas permissões de função de serviço com as permissões de função de serviço AWS Batch gerenciada e resolva quaisquer lacunas.

   É uma prática recomendada usar a AWS Batch SLR em ambientes computacionais para evitar erros semelhantes.

   Cancelar o trabalho, manualmente ou definindo o `jobStateTimeLimitActions` parâmetro ativado`statusReason`, permite que o trabalho subsequente seja movido para o início da fila. Sem resolver o (s) problema (s) da função de serviço, é provável que o próximo trabalho também seja bloqueado. É melhor investigar e resolver esse problema manualmente.

   - **`statusReason`**mensagem enquanto o trabalho está travado: `MISCONFIGURATION:SERVICE_ROLE_PERMISSIONS – Batch service role has a permission issue.`
   - **`reason`**usado para**`jobStateTimeLimitActions`**: `MISCONFIGURATION:SERVICE_ROLE_PERMISSIONS`
   - **`statusReason`**mensagem após o cancelamento do trabalho: `Canceled by JobStateTimeLimit action due to reason: MISCONFIGURATION:SERVICE_ROLE_PERMISSIONS`

5. Motivo: todos os ambientes computacionais são inválidos. Para obter mais informações, consulte ambiente `INVALID` computacional. Observação: você não pode configurar uma ação programável por meio do `jobStateTimeLimitActions` parâmetro para resolver esse erro.
   - **`statusReason`**mensagem enquanto o trabalho está travado: `ACTION_REQUIRED - CE(s) associated with the job queue are invalid.`
6. Motivo: AWS Batch detectou uma fila bloqueada, mas não consegue determinar o motivo. Observação: você não pode configurar uma ação programável por meio do `jobStateTimeLimitActions` parâmetro para resolver esse erro. Para obter mais informações sobre solução de problemas, consulte Por que meu AWS Batch trabalho está preso em RUNNABLE on AWS no re:POST.
   - **`statusReason`**mensagem enquanto o trabalho está travado: `UNDETERMINED - Batch job is blocked, root cause is undetermined.`

Caso você não tenha recebido um evento da CloudWatch Events ou tenha recebido o evento de motivo desconhecido, aqui estão algumas causas comuns desse problema.

O driver de **`awslogs`** log não está configurado em seus recursos computacionais

AWS Batch os jobs enviam suas informações de registro para o CloudWatch Logs. Para ativar, você deve configurar os recursos de computação para usar o driver de log `awslogs`. Suponha que você baseie sua AMI de recursos de computação na AMI otimizada do Amazon ECS (ou Amazon Linux). Em seguida, esse driver é registrado por padrão com o `ecs-init` pacote. Agora, suponha que você use uma AMI base diferente. Em seguida, você deve verificar que o driver de log `awslogs` seja especificado como um driver de log disponível com a variável de ambiente `ECS_AVAILABLE_LOGGING_DRIVERS` quando o agente de contêiner for iniciado. Para ter mais informações, consulte Especificação da AMI do recurso de computação e Como criar uma AMI de recursos de computação.

Recursos insuficientes

Se suas definições de trabalho especificarem mais recursos de CPU ou memória do que seus recursos de computação podem alocar, seus trabalhos nunca serão colocados. Por exemplo, suponha que seu trabalho especifique 4 GiB de memória e que seus recursos de computação tenham menos do que os disponíveis. Então, acontece que o trabalho não pode ser colocado nesses recursos de computação. Nesse caso, você deve reduzir a memória especificada em sua definição de trabalho ou adicionar mais recursos de computação ao seu ambiente. Alguma memória é reservada para o agente de contêiner do Amazon ECS e outros processos críticos

do sistema. Para ter mais informações, consulte Recurso de Computação Gerenciamento de Memória.

Sem acesso à Internet para recursos computacionais

Recursos de computação precisam de acesso para se comunicar com o endpoint de serviço do Amazon ECS. Isso pode ser feito por meio de uma interface do endpoint da VPC ou por meio dos das instâncias de contêiner que tenham endereços IP públicos.

Para obter mais informações sobre endpoints da VPC de interface, consulte Endpoints da VPC de interface do Amazon ECS (AWS PrivateLink) no Manual do Desenvolvedor do Amazon Elastic Container Service.

Se você não tiver um endpoint da VPC de interface configurado e seus das instâncias de contêiner não tiverem endereços IP públicos, eles deverão usar a conversão de endereço de rede (NAT) para fornecer esse acesso. Para obter mais informações, consulte Gateways NAT no Guia do usuário da Amazon VPC. Para ter mais informações, consulte the section called “Crie uma VPC”.

Limite de instâncias do Amazon EC2 atingido

O número de instâncias do Amazon EC2 em que sua conta pode iniciar Região da AWS é determinado pela cota de sua instância do EC2. Alguns tipos de instância também têm uma per-instance-type cota. Para obter mais informações sobre a cota de instância do Amazon EC2 da sua conta, incluindo como solicitar um aumento de limite, consulte Amazon EC2 Service Limits no Guia do usuário do Amazon EC2 para instâncias Linux.

O agente de contêiner do Amazon ECS não está instalado

O agente de contêiner do Amazon ECS deve ser instalado na Amazon Machine Image (AMI) para permitir a AWS Batch execução de trabalhos. O agente do contêiner do Amazon ECS é instalado por padrão nas AMIs otimizadas para o Amazon ECS. Para obter mais informações sobre o agente de contêineres do Amazon ECS, consulte Agente de contêineres do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

Para obter mais informações, consulte Por que meu AWS Batch trabalho está preso no `RUNNABLE` status? em re:POST.

Instâncias spot sem tags na criação

A marcação de instâncias spot para recursos AWS Batch computacionais é suportada a partir de 25 de outubro de 2017. Antes, a política gerenciada de IAM recomendada (`AmazonEC2SpotFleetRole`) para o perfil Amazon EC2 Spot Fleet não continha permissões para marcar instâncias Spot no lançamento. A nova política gerenciada de IAM recomendada é chamada `AmazonEC2SpotFleetTaggingRole`. Ele suporta a marcação de instâncias spot no lançamento.

Para corrigir a marcação de Instância Spot na criação, siga o procedimento a seguir para aplicar a política gerenciada de IAM recomendada atualmente ao seu perfil do Amazon EC2 Frota Spot. Dessa forma, todas as futuras Instâncias Spot criadas com esse perfil têm permissões para aplicar tags de instância quando são criadas.

Para aplicar a política gerenciada de IAM atual ao seu perfil do Amazon EC2 Frota Spot.

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. Escolha Roles e escolha seu perfil de frota spot do seu Amazon EC2.
3. Escolha Attach policy.
4. Selecione o AmazonEC2 SpotFleetTaggingRole e escolha Anexar política.
5. Escolha seu perfil Amazon EC2 Frota Spot novamente para remover a política anterior.
6. Selecione o x à direita da SpotFleetRole política do AmazonEC2 e escolha Desanexar.

As instâncias spot não estão diminuindo

AWS Batch introduziu a função AWSServiceRoleForBatchvinculada ao serviço em 10 de março de 2021. Se nenhum perfil for especificado no parâmetro `serviceRole` do ambiente de computação, esse perfil vinculado ao serviço será usado como perfil de serviço. No entanto, suponha que a função vinculada ao serviço seja usada em um ambiente computacional EC2 Spot, mas a função Spot usada não inclua a política gerenciada do AmazonEC2. SpotFleetTaggingRole Então, a Instância Spot não é escala reduzida. Como resultado, você receberá uma mensagem de erro com a seguinte mensagem: “Você não está autorizado a executar esta operação”. Use as etapas a seguir para atualizar o perfil da frota spot que você usa no parâmetro `spotIamFleetRole`. Para obter mais informações, consulte Uso de funções vinculadas a serviços e Criação de uma função para delegar permissões a um AWS serviço no Guia do usuário do IAM.

Tópicos

- Anexe a política SpotFleetTaggingRole gerenciada do AmazonEC2 à sua função Spot Fleet no AWS Management Console
- Anexe a política SpotFleetTaggingRole gerenciada do AmazonEC2 à sua função Spot Fleet com o AWS CLI

Anexe a política SpotFleetTaggingRole gerenciada do AmazonEC2 à sua função Spot Fleet no AWS Management Console

Para aplicar a política gerenciada de IAM atual ao seu perfil do Amazon EC2 Frota Spot

1. Abra o console IAM em https://console.aws.amazon.com/iam/.
2. Escolha Roles e escolha seu perfil de frota spot do seu Amazon EC2.
3. Escolha Attach policy.
4. Selecione o AmazonEC2 SpotFleetTaggingRole e escolha Anexar política.
5. Escolha seu perfil Amazon EC2 Frota Spot novamente para remover a política anterior.
6. Selecione o x à direita da SpotFleetRole política do AmazonEC2 e escolha Desanexar.

Anexe a política SpotFleetTaggingRole gerenciada do AmazonEC2 à sua função Spot Fleet com o AWS CLI

Os comandos de exemplo pressupõem que sua função de frota spot do Amazon EC2 se chama AmazonEC2. SpotFleetRole Se seu perfil usar um nome diferente, ajuste os comandos para que correspondam.

Para anexar a política SpotFleetTaggingRole gerenciada do AmazonEC2 à sua função Spot Fleet

1. Para anexar a política de IAM SpotFleetTaggingRole gerenciada pelo AmazonEC2 à sua SpotFleetRole função do *AmazonEC2*, execute o comando a seguir usando o. AWS CLI

```
$ aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEC2SpotFleetTaggingRole
 \
    --role-name AmazonEC2SpotFleetRole
```

2. Para separar a política do IAM SpotFleetRole gerenciada pelo AmazonEC2 da sua função do *AmazonEC2*, execute o comando a SpotFleetRole seguir usando o. AWS CLI

```
$ aws iam detach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEC2SpotFleetRole \
    --role-name AmazonEC2SpotFleetRole
```

Não é possível recuperar segredos do Secrets Manager

Se você usar uma AMI com um agente do Amazon ECS anterior à versão 1.16.0-1, deverá usar a variável `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` de configuração do agente do Amazon ECS para usar esse recurso. Você pode adicioná-lo ao arquivo `./etc/ecs/ecs.config` para uma nova instância de contêiner ao criar essa instância. Ou você pode adicioná-lo a uma instância existente. Se adicioná-lo a uma instância existente, reinicie o agente ECS após adicioná-lo. Para obter mais informações, consulte Configuração do Agente de Contêineres do Amazon ECS no Guia do desenvolvedor do Amazon Elastic Container Service.

Não é possível substituir os requisitos de recursos de definição de trabalho

As substituições de memória e vCPU especificadas na estrutura `memory` e nos `vcpus` membros da estrutura ContainerOverrides, transmitida para, não SubmitJobpodem substituir os requisitos de memória e vCPU especificados na estrutura ResourceRequirements na definição do trabalho.

Se tentar substituir esses requisitos de recursos, você poderá ver a mensagem de erro a seguir:

“Esse valor foi enviado em uma chave obsoleta e pode entrar em conflito com o valor fornecido pelos requisitos de recursos da definição de tarefa.”

Para corrigir isso, especifique os requisitos de memória e vCPU no membro ResourceRequirements do ContainerOverrides. Por exemplo, se suas substituições de memória e vCPU forem especificadas nas linhas a seguir.

```
"containerOverrides": {
   "memory": 8192,
   "vcpus": 4
}
```

Altere o código para o seguinte:

```
"containerOverrides": {
```

```
   "resourceRequirements": [
      {
         "type": "MEMORY",
         "value": "8192"
      },
      {
         "type": "VCPU",
         "value": "4"
      }
   ],
}
```

Faça a mesma alteração nos requisitos de memória e vCPU especificados no objeto ContainerProperties na definição do trabalho. Por exemplo, se seus requisitos de memória e vCPU estiverem especificados nas linhas a seguir.

```
{
   "containerProperties": {
      "memory": 4096,
      "vcpus": 2,
}
```

Altere o código para o seguinte:

```
"containerProperties": {
   "resourceRequirements": [
      {
         "type": "MEMORY",
         "value": "4096"
      },
      {
         "type": "VCPU",
         "value": "2"
      }
   ],
}
```

Mensagem de erro ao atualizar a configuração **`desiredvCpus`**

Você vê a seguinte mensagem de erro ao usar a AWS Batch API para atualizar a configuração desejada de vCPUs (`desiredvCpus`).

`Manually scaling down compute environment is not supported. Disconnecting job queues from compute environment will cause it to scale-down to minvCpus.`

Esse problema ocorre se o valor `desiredvCpus` atualizado for menor que o valor `desiredvCpus` atual. Ao atualizar o valor `desiredvCpus`, ambas as seguintes situações devem ser verdadeiras:

- O valor `desiredvCpus` deve estar entre os valores `minvCpus` e `maxvCpus`.
- O valor `desiredvCpus` atualizado deve ser maior ou igual ao valor `desiredvCpus` atual.

AWS Batch no Amazon EKS

Tópicos

Ambiente de computação do **`INVALID`**

É possível que você tenha configurado incorretamente um ambiente de computação gerenciado. Se você fez isso, o ambiente de computação entrará em um estado `INVALID` e não poderá aceitar trabalhos para colocação. As seções a seguir descrevem as possíveis causas e como solucionar problemas com base na causa.

Versão Kubernetes sem suporte

Talvez você veja uma mensagem de erro semelhante à seguinte ao usar a operação de API `CreateComputeEnvironment` ou a operação de API `UpdateComputeEnvironment` para criar ou atualizar um ambiente de computação. Esse problema ocorre se você especificar uma versão Kubernetes não suportada em `EC2Configuration`.

```
At least one imageKubernetesVersion in EC2Configuration is not supported.
```

Para resolver esse problema, exclua o ambiente de computação e recrie-o com uma versão Kubernetes compatível.

Você pode realizar uma pequena atualização de versão no seu cluster Amazon EKS. Por exemplo, você pode atualizar o cluster de `1.xx` para, `1.yy` mesmo que a versão secundária não seja compatível.

No entanto, o status do ambiente de computação pode mudar para `INVALID` depois de uma atualização de versão principal. Por exemplo, se você realizar uma atualização de versão principal de `1.xx` para`2.yy`. Se a versão principal não for compatível com AWS Batch, você verá uma mensagem de erro semelhante à seguinte.

```
reason=CLIENT_ERROR - ... EKS Cluster version [2.yy] is unsupported
```

Para resolver esse problema, especifique uma versão Kubernetes compatível ao usar uma operação de API para criar ou atualizar um ambiente computacional.

AWS Batch no Amazon EKS atualmente oferece suporte às seguintes Kubernetes versões:

- `1.29`
- `1.28`
- `1.27`
- `1.26`
- `1.25`
- `1.24`
- `1.23`

O perfil da instância não existe

Se o perfil de instância especificado não existir, o status do ambiente computacional AWS Batch no Amazon EKS será alterado para`INVALID`. Você vê um erro definido no parâmetro `statusReason` semelhante ao seguinte.

```
CLIENT_ERROR - Instance profile arn:aws:iam::...:instance-profile/<name> does not exist
```

Para resolver esse problema, especifique ou crie um perfil de instância em funcionamento. Para obter mais informações, consulte função do IAM do nó do Amazon EKS no Guia do usuário do Amazon EKS.

Namespace Kubernetes inválido

Se AWS Batch no Amazon EKS não puder validar o namespace para o ambiente computacional, o status do ambiente computacional será alterado para. `INVALID` Por exemplo, esse problema pode ocorrer se o namespace não existir.

Você vê uma mensagem de erro definida no parâmetro `statusReason` semelhante à seguinte.

```
CLIENT_ERROR - Unable to validate Kubernetes Namespace
```

Esse problema pode ocorrer se qualquer uma das seguintes situações for verdadeira:

- A string do namespace Kubernetes na chamada `CreateComputeEnvironment` não existe. Para obter mais informações, consulte CreateComputeEnvironment.
- As permissões de controle de acesso baseado em perfil (RBAC) necessárias para gerenciar o namespace não estão configuradas corretamente.
- AWS Batch não tem acesso ao endpoint do servidor da Kubernetes API Amazon EKS.

Para resolver esse problema, consulte Verifique se o `aws-auth ConfigMap` está configurado corretamente. Para obter mais informações, consulteComeçando a usar AWS Batch no Amazon EKS

Ambiente de computação gerenciado

Suponha que você exclua um cluster do Amazon EKS antes de excluir o anexo AWS Batch no ambiente computacional Amazon EKS. Em seguida, o status do ambiente computacional é alterado para `INVALID`. Nesse cenário, o ambiente computacional não funcionará adequadamente se você recriar o cluster Amazon EKS com o mesmo nome.

Para resolver esse problema, exclua e recrie o ambiente computacional AWS Batch no Amazon EKS.

Os nós não se juntam ao cluster Amazon EKS

AWS Batch no Amazon EKS, reduz a escala de um ambiente computacional se ele determinar que nem todos os nós se juntaram ao cluster do Amazon EKS. Quando o AWS Batch Amazon EKS reduz a escala do ambiente computacional, o status do ambiente computacional é alterado para. `INVALID`

> **Note**
>
> AWS Batch não altera o status do ambiente computacional imediatamente para que você possa depurar o problema.

Você vê uma mensagem de erro definida no parâmetro `statusReason` semelhante às seguintes:

```
Your compute environment has been INVALIDATED and scaled down because
none of the instances joined the underlying ECS Cluster. Common issues
preventing instances joining are the following: VPC/Subnet configuration
preventing communication to ECS, incorrect Instance Profile policy
preventing authorization to ECS, or customized AMI or LaunchTemplate
configurations affecting ECS agent.
```

```
Your compute environment has been INVALIDATED and scaled down because
none of the nodes joined the underlying Amazon EKS Cluster. Common issues
preventing nodes joining are the following: networking configuration
preventing communication to Amazon EKS Cluster, incorrect Amazon EKS
Instance Profile or Kubernetes RBAC policy preventing authorization
to Amazon EKS Cluster, customized AMI or LaunchTemplate configurations
affecting Amazon EKS/Kubernetes node bootstrap.
```

Ao usar uma AMI padrão do Amazon EKS, as causas mais comuns desse problema são as seguintes:

- O perfil da instância não está configurado corretamente. Para obter mais informações, consulte função do IAM do nó do Amazon EKS no Guia do usuário do Amazon EKS.
- As sub-redes não estão configuradas corretamente. Para obter mais informações, consulte os requisitos e considerações sobre VPC e sub-rede do Amazon EKS no Guia do usuário do Amazon EKS.
- O grupo de segurança não está configurado corretamente. Para obter mais informações, consulte Amazon EKS security group requirements and considerations no Amazon EKS User Guide.

> **Note**
>
> Você também pode ver uma notificação de erro no Personal Health Dashboard (PHD).

AWS Batch no Amazon EKS, o trabalho está preso no **RUNNABLE** status

O `ConfigMap` `aws-auth` é criado e aplicado automaticamente ao cluster ao criar um grupo de nós gerenciados ou ao criar um grupo de nós usando `eksctl`. Um `aws-auth` `ConfigMap` é criado inicialmente para permitir que os nós se juntem ao seu cluster. No entanto, você também pode usar o `aws-auth` `ConfigMap` para adicionar acesso com controle de acesso baseado em perfil (RBAC) para usuários e perfis.

Para verificar se o `aws-auth` `ConfigMap` está configurado corretamente:

1. Recupere as funções mapeadas no: `aws-auth` `ConfigMap`

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

2. Verifique se o `roleARN` está configurado da seguinte forma.

   `rolearn: arn:aws:iam::`*`aws_account_number`*`:role/AWSServiceRoleForBatch`

   > **Note**
   >
   > Você também pode revisar os registros do plano de controle do Amazon EKS. Para obter mais informações, consulte Amazon EKS control plane logging no Amazon EKS User Guide.

Para resolver um problema em que um trabalho está preso em um status `RUNNABLE`, recomendamos que você use `kubectl` para reaplicar o manifesto. Para ter mais informações, consulte Etapa 1: Preparando seu cluster Amazon EKS para AWS Batch. Ou, você pode usar `kubectl` para editar manualmente o `aws-auth` `ConfigMap`. Para obter mais informações, consulte Habilitar o acesso ao seu cluster no Amazon EKS User Guide.

Verifique se o **`aws-auth` `ConfigMap`** está configurado corretamente

Para verificar se o `aws-auth` `ConfigMap` está configurado corretamente:

1. Recupere as funções mapeadas no `aws-auth` `ConfigMap`.

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

2. Verifique se o `roleARN` está configurado da seguinte forma.

```
rolearn: arn:aws:iam::aws_account_number:role/AWSServiceRoleForBatch
```

> **Note**
>
> O caminho `aws-service-role/batch.amazonaws.com/` foi removido do ARN so perfil vinculado ao serviço. Isso ocorre devido a um problema com o mapa de configuração `aws-auth`. Para obter mais informações, consulte Perfis com caminhos não funcionam quando o caminho é incluído em seu ARN no aws-authconfigmap.

> **Note**
>
> Você também pode revisar os registros do plano de controle do Amazon EKS. Para obter mais informações, consulte Amazon EKS control plane logging no Amazon EKS User Guide.

Para resolver um problema em que um trabalho está preso em um status `RUNNABLE`, recomendamos que você use `kubectl` para reaplicar o manifesto. Para ter mais informações, consulte Etapa 1: Preparando seu cluster Amazon EKS para AWS Batch. Ou, você pode usar `kubectl` para editar manualmente o `aws-auth ConfigMap`. Para obter mais informações, consulte Habilitar o acesso ao seu cluster no Amazon EKS User Guide.

As permissões ou vinculações do RBAC não estão configuradas corretamente

Se você tiver problemas de permissão ou vinculação de RBAC, verifique se o `aws-batch` Kubernetes perfil pode acessar o namespace Kubernetes:

```
$ kubectl get namespace namespace --as=aws-batch
```

```
$ kubectl auth can-i get ns --as=aws-batch
```

Você também pode usar o comando **`kubectl describe`** para visualizar as autorizações para um perfil de cluster ou namespace Kubernetes.

```
$ kubectl describe clusterrole aws-batch-cluster-role
```

A seguir, um exemplo de saída.

```
Name:         aws-batch-cluster-role
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources                                       Non-Resource URLs  Resource Names
 Verbs
  ---------                                       -----------------  --------------
 -----
  configmaps                                      []                 []
 [get list watch]
  nodes                                           []                 []
 [get list watch]
  pods                                            []                 []
 [get list watch]
  daemonsets.apps                                 []                 []
 [get list watch]
  deployments.apps                                []                 []
 [get list watch]
  replicasets.apps                                []                 []
 [get list watch]
  statefulsets.apps                               []                 []
 [get list watch]
  clusterrolebindings.rbac.authorization.k8s.io  []                 []
 [get list]
  clusterroles.rbac.authorization.k8s.io          []                 []
 [get list]
  namespaces                                      []                 []
 [get]
```

```
$ kubectl describe role aws-batch-compute-environment-role -n my-aws-batch-namespace
```

A seguir, um exemplo de saída.

```
Name:         aws-batch-compute-environment-role
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources                                 Non-Resource URLs  Resource Names  Verbs
```

```
  ---------                               -----------------  --------------  -----
  pods                                    []                 []              [create
get list watch delete patch]
  serviceaccounts                         []                 []              [get list]
  rolebindings.rbac.authorization.k8s.io  []                 []              [get list]
  roles.rbac.authorization.k8s.io         []                 []              [get list]
```

Para resolver esse problema, reaplique as permissões e comandos do RBAC. `rolebinding` Para ter mais informações, consulte Etapa 1: Preparando seu cluster Amazon EKS para AWS Batch.

Práticas Recomendadas para AWS Batch

Você pode usar o AWS Batch para executar uma variedade de workloads computacionais exigentes em grande escala sem gerenciar uma arquitetura complexa. Os trabalhos do AWS Batch podem ser usados em uma ampla variedade de casos de uso em áreas como epidemiologia, jogos e machine learning.

Este tópico aborda as práticas recomendadas a serem consideradas durante o uso do AWS Batch e instruções sobre como executar e otimizar suas workloads ao utilizar do AWS Batch.

Tópicos

Quando utilizar AWS Batch

AWS Batch executa trabalhos em escala a baixo custo, além de prover serviços de filas em escala com otimização de custos. No entanto, nem toda workload é adequada para ser executada utilizando AWS Batch.

- Trabalhos curtos – se um trabalho for executado por apenas alguns segundos, a sobrecarga para agendar o trabalho em lotes poderá demorar mais do que o runtime do trabalho em si. Como solução alternativa, reúna suas tarefas binpack antes de inseri-las em AWS Batch. Em seguida, configure seus trabalhos do AWS Batch para repetir as tarefas. Por exemplo, prepare os argumentos de tarefas individuais em uma tabela do Amazon DynamoDB, ou como um arquivo em um bucket do Amazon S3. Considere agrupar tarefas para que os trabalhos sejam executados por 3-5 minutos cada. Depois de binpack os trabalhos, faça loop pelos seus grupos de tarefas em seu trabalho AWS Batch.
- Trabalhos que devem ser executados imediatamente – AWS Batch pode processar trabalhos rapidamente. No entanto, o AWS Batch é um programador, que otimiza para desempenho de

custo, prioridade de trabalho e throughput. AWS Batch pode exigir tempo para processar suas solicitações. Se você precisar de uma resposta em poucos segundos, uma abordagem baseada em serviço utilizando o Amazon ECS ou o Amazon EKS é mais adequada.

Lista de verificação para execução em escala

Antes de executar uma grande workload em 50 mil ou mais vCPUs, considere a lista de verificação a seguir.

> **Note**
>
> Se você planeja executar uma grande workload em um milhão ou mais vCPUs, ou precisa de orientação para execução em grande escala, entre em contato com sua equipe AWS.

- Verifique suas cotas do Amazon EC2 – verifique suas cotas do Amazon EC2 (também conhecidas como limites) no painel Service Quotas do AWS Management Console. Se necessário, solicite um aumento de cota para seu número máximo de instâncias do Amazon EC2. Lembre-se de que o Amazon EC2 Spot e as instâncias sob demanda têm cotas separadas. Para obter mais informações, consulte Conceitos Básicos de Service Quotas.
- Verifique sua cota do Amazon Elastic Block Store para cada Região – cada instância utiliza um volume GP2 ou GP3 para o sistema operacional. Por padrão, a cota para cada Região da AWS é de 300 TiB. No entanto, cada instância utiliza contagens de uso como parte dessa cota. Portanto, certifique-se de levar isso em consideração ao verificar sua cota do Amazon Elastic Block Store para cada Região. Se sua cota for alcançada, você não poderá criar mais instâncias. Para obter mais informações, consulte Endpoints e Cotas do Amazon Elastic Block Store
- Utilize o Amazon S3 para armazenamento – o Amazon S3 fornece alta taxa de transferência e ajuda a eliminar a suposição sobre quanto armazenamento provisionar baseado no número de trabalhos e instâncias em cada Zona de Disponibilidade. Para obter mais informações, consulte Padrões de Design de Práticas Recomendadas: Otimizando a Performance do Amazon S3.
- Escale gradualmente para identificar gargalos com antecedência – Para um trabalho em execução em um milhão ou mais de vCPUs, comece menor e aumente gradualmente para que você possa identificar gargalos mais cedo. Por exemplo, comece executando em 50 mil vCPUs. Em seguida, aumente a contagem para 200 mil vCPUs, depois 500 mil vCPUs, e assim por diante. Em outras palavras, continue aumentando gradualmente a contagem de vCPUs até alcançar o número de vCPUs desejado.

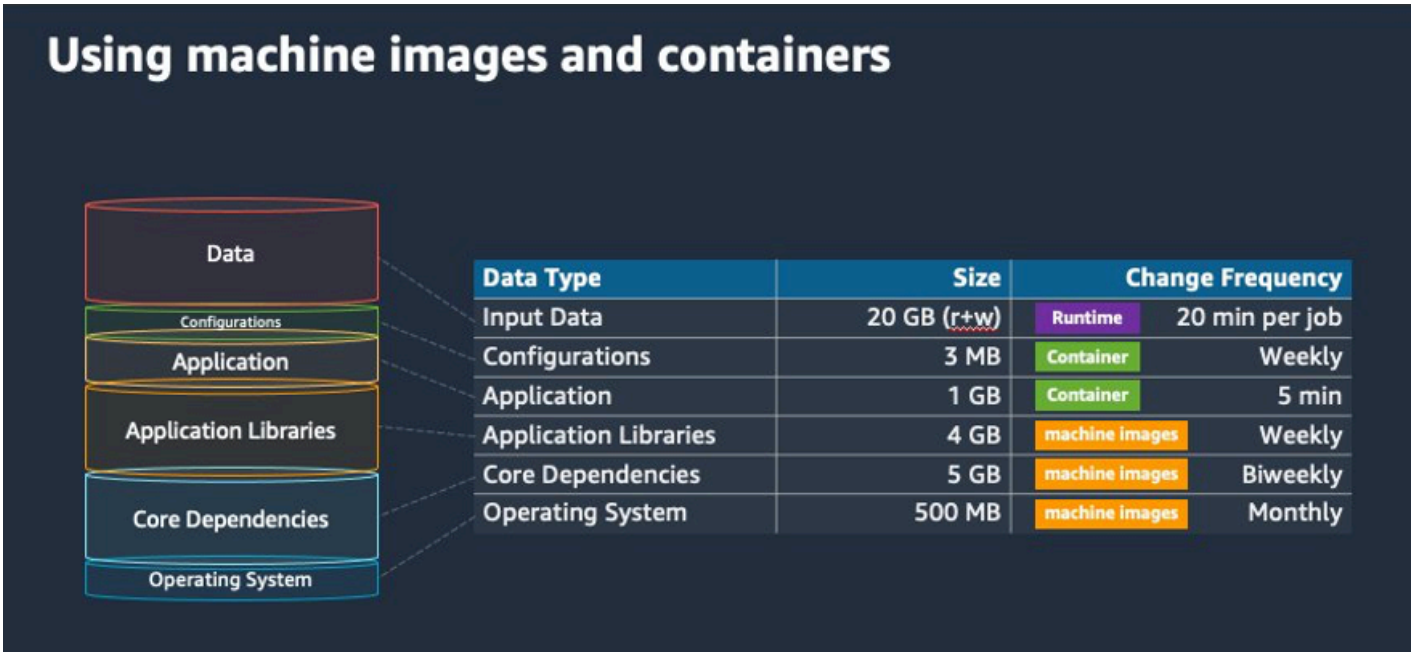
- Monitore para identificar possíveis problemas com antecedência – Para evitar possíveis interrupções e problemas ao executar em escala, certifique-se de monitorar seu aplicativo e sua arquitetura. Podem ocorrer interrupções mesmo ao escalar de 1 mil para 5 mil vCPUs. Você pode utilizar o Amazon CloudWatch Logs para analisar dados de log ou utilizar o CloudWatch Embedded Metrics, com uma biblioteca de cliente. Para obter mais informações, consulte Referência do Atendente do CloudWatch Logs e `aws-embedded-metrics`

Otimize contêineres e AMIs

O tamanho e a estrutura do contêiner são importantes para o primeiro conjunto de trabalhos que você executar. Isso é especialmente verdadeiro quando o contêiner é maior que 4 GB. As imagens de contêiner são integradas em camadas. As camadas são restauradas em paralelo pelo Docker utilizando três threads simultâneas. Você pode aumentar o número de segmentos concomitantes utilizando o parâmetro `max-concurrent-downloads`. Para obter mais informações, consulte Documentação do Docker.

Embora você possa utilizar contêineres maiores, recomendamos otimizar a estrutura e o tamanho do contêiner para tempos de startup mais rápidos.

- Contêineres menores são buscados mais rapidamente – Contêineres menores podem levar a tempos de início de aplicativos mais rápidos. Para diminuir o tamanho do contêiner, descarregue bibliotecas ou arquivos atualizados com pouca frequência para a imagem de máquina da Amazon (AMI). Você também pode associar montagens para dar acesso aos seus contêineres. Para obter mais informações, consulte Associando Montagens.
- Crie camadas de tamanho uniformes em tamanho e quebre camadas grandes – Cada camada é restaurada por uma thread. Portanto, uma camada grande pode impactar significativamente seu tempo de startup de trabalho. É recomendado um tamanho máximo de camada de 2 GB como boa concessão entre um tamanho maior do contêiner e tempos de startup mais rápidos. Você pode executar o comando `docker history your_image_id` para verificar sua estrutura de imagens de contêiner e o tamanho da camada. Para obter mais informações, consulte Documentação do Docker.
- Use o Amazon Elastic Container Registry como seu repositório de contêineres – Quando você executa milhares de trabalhos em paralelo, um repositório autogerenciado pode falhar ou controlar a utilização da taxade transferência. O Amazon ECR funciona em escala e processa workloads de até um milhão de vCPUs.



Escolha a opção direita de recurso de ambiente de computação

AWS Fargate exige menos definições e configurações iniciais do que o Amazon EC2 e provavelmente é mais fácil de utilizar, especialmente se for sua primeira vez. Com o Fargate, você não precisa gerenciar servidores, processar o planejamento de capacidade ou isolar workloads de contêiner para segurança.

Se você têm os seguintes requisitos, recomendamos que use instâncias do Fargate:

- seus trabalhos devem iniciar rapidamente, em menos de 30 segundos.
- Os requisitos dos seus trabalhos são 16 vCPUs ou menos, sem GPUs, e 120 GiB de memória ou menos.

Para obter mais informações, consulte Quando usar o Fargate.

se você têm os seguintes requisitos, recomendamos que utilize instâncias Amazon EC2:

- Caso exija controle aumentado sobre a seleção da instância, ou exige a utilização de tipos de instância específicos.
- Seus trabalhos exigem recursos que o AWS Fargate não pode fornecer, como GPUs, mais memória, uma AMI personalizada, ou o Amazon Elastic Fabric Adapter.
- Caso exija um alto nível de taxa de transferência ou simultaneidade.

- Caso precise personalizar sua AMI, acesse Modelos de Inicialização do Amazon EC2 ou acesse parâmetros especiais do Linux.

Com o Amazon EC2, você pode ajustar com maior precisão sua workload de acordo com seus requisitos específicos e executá-la em escala, se necessário.

Amazon EC2 Sob Demanda, ou Amazon EC2 Spot

A maioria dos clientes AWS Batch utiliza instâncias spot do Amazon EC2 devido a economia em relação às instâncias sob demanda. No entanto, se sua workload for executada por várias horas e não puder ser interrompida, as instâncias sob demanda podem ser mais adequadas para você. Você sempre pode experimentar as instâncias spot primeiro e mudar para sob demanda, se necessário.

Caso tenha os seguintes requisitos e expectativas, utilize as instâncias sob demanda do Amazon EC2:

- quando o runtime dos seus trabalhos for maior que uma hora e você não possa tolerar interrupções na workload.
- Você tem um SLO (objetivo de nível de serviço) rigoroso para sua workload geral e não pode aumentar o tempo de computação.
- As instâncias requeridas têm maior probabilidade de sofrer interrupções.

Caso tenha os seguintes requisitos e expectativas, utilize as instâncias spot do Amazon EC2:

- quando o runtime usual de seus trabalhos for de até 30 minutos.
- Você pode tolerar potenciais interrupções e reagendamento de tarefas como parte da workload. Para obter mais informações, consulte Advisor de Instâncias Spot.
- Tarefas em execução longas podem ser reiniciadas a partir de um ponto de verificação, caso sejam interrompidas.

Você pode misturar os dois modelos de compra enviando na instância spot primeiro e depois utilizando a Instância Sob Demanda como opção fallback. Por exemplo, insira seus trabalhos em uma fila conectada a ambientes computacionais em execução em instâncias spot do Amazon EC2. Se um trabalho for interrompido, capture o evento de Amazon EventBridge e correlacione-o a uma recuperação de instância Spot. Em seguida, reenvie o trabalho para uma fila sob demanda utilizando uma função AWS Lambda ou AWS Step Functions. Para obter mais informações, consulte Tutorial:

Como enviar alertas do Amazon Simple Notification Service para eventos de trabalho que falharam, Práticas Recomendadas para Processar Interrupções da Instância Spot do Amazon EC2 e Gerenciar AWS Batch com Step Functions.

> **Important**
>
> Utilize diferentes tipos, tamanhos e Zonas de Disponibilidade de instâncias para seu ambiente de computação sob demanda para manter a disponibilidade dos grupos de instâncias spot do Amazon EC2 e diminuir a taxa de interrupção.

Use as práticas recomendadas do Amazon EC2 Spot para AWS Batch

Quando você escolhe as instâncias spot do Amazon Elastic Compute Cloud (EC2), é provável que você possa otimizar seu fluxo de trabalho para economizar custos - às vezes, de forma significativa. Para obter mais informações, consulte Práticas Recomendadas para o Amazon EC2 Spot.

Para otimizar seu fluxo de trabalho e economizar custos, considere as seguintes práticas recomendadas do Amazon EC2 Spot para AWS Batch:

- Escolher a estratégia de alocação **`SPOT_CAPACITY_OPTIMIZED`** – AWS Batch escolhe instâncias do Amazon EC2 a partir dos grupos de capacidade spot mais profundos do Amazon EC2. Se estiver preocupado com interrupções, essa é uma escolha adequada. Para obter mais informações, consulte Estratégias de alocação.
- Diversifique os tipos de instância – Para diversificar seus tipos de instância, considere tamanhos e famílias compatíveis e, em seguida, deixe AWS Batch escolher com base em preço ou disponibilidade. Por exemplo, considere `c5.24xlarge` como uma alternativa para `c5.12xlarge` ou famílias `c5a`, `c5n`, `c5d`, `m5` e `m5d`. Para obter mais informações, consulte Seja Flexível sobre Tipos de Instância e Zonas de Disponibilidade.
- Reduza o runtime do trabalho ou o ponto de verificação – Desaconselhamos executar tarefas que levem uma hora ou mais ao utilizar instâncias spot do Amazon EC2 para evitar interrupções. Caso divida ou verifique seus trabalhos em partes menores que consistam em 30 minutos ou menos, você pode reduzir significativamente a possibilidade de interrupções.
- Utilize repetições automatizadas – Para evitar interrupções em configurações de trabalhos AWS Batch, defina repetições automatizadas para trabalhos. Os trabalhos em lote podem ser

interrompidos por qualquer um dos seguintes motivos: um código de saída retornado diferente de zero; um erro de serviço ocorrido; ou uma recuperação de instância ocorrida. Você pode configurar até 10 repetições automatizadas. De início, é recomendado definir 1-3 repetições automatizadas pelo menos. Para obter informações sobre o monitoramento de interrupções spot do Amazon EC2, consulte Painel de Interrupções de Spot.

No AWS Batch, se você definir o parâmetro de repetição, o trabalho será colocado na frente da fila de trabalhos. Ou seja, o trabalho ganha prioridade. Ao criar a definição de trabalho ou inseri-lo no AWS CLI, você pode configurar uma estratégia de repetição. Para obter mais informações, consulte submit-job.

```
$ aws batch submit-job --job-name MyJob \
    --job-queue MyJQ \
    --job-definition MyJD \
    --retry-strategy attempts=2
```

- Utilize novas tentativas personalizadas – Você pode configurar uma estratégia de repetição de trabalho para um código de saída de aplicativo específico ou recuperação de instância. No exemplo a seguir, caso o host cause a falha, o trabalho poderá ser repetido até cinco vezes. No entanto, trabalhos com falha por um motivo diferente serão encerrados, e o status será definido como `FAILED`.

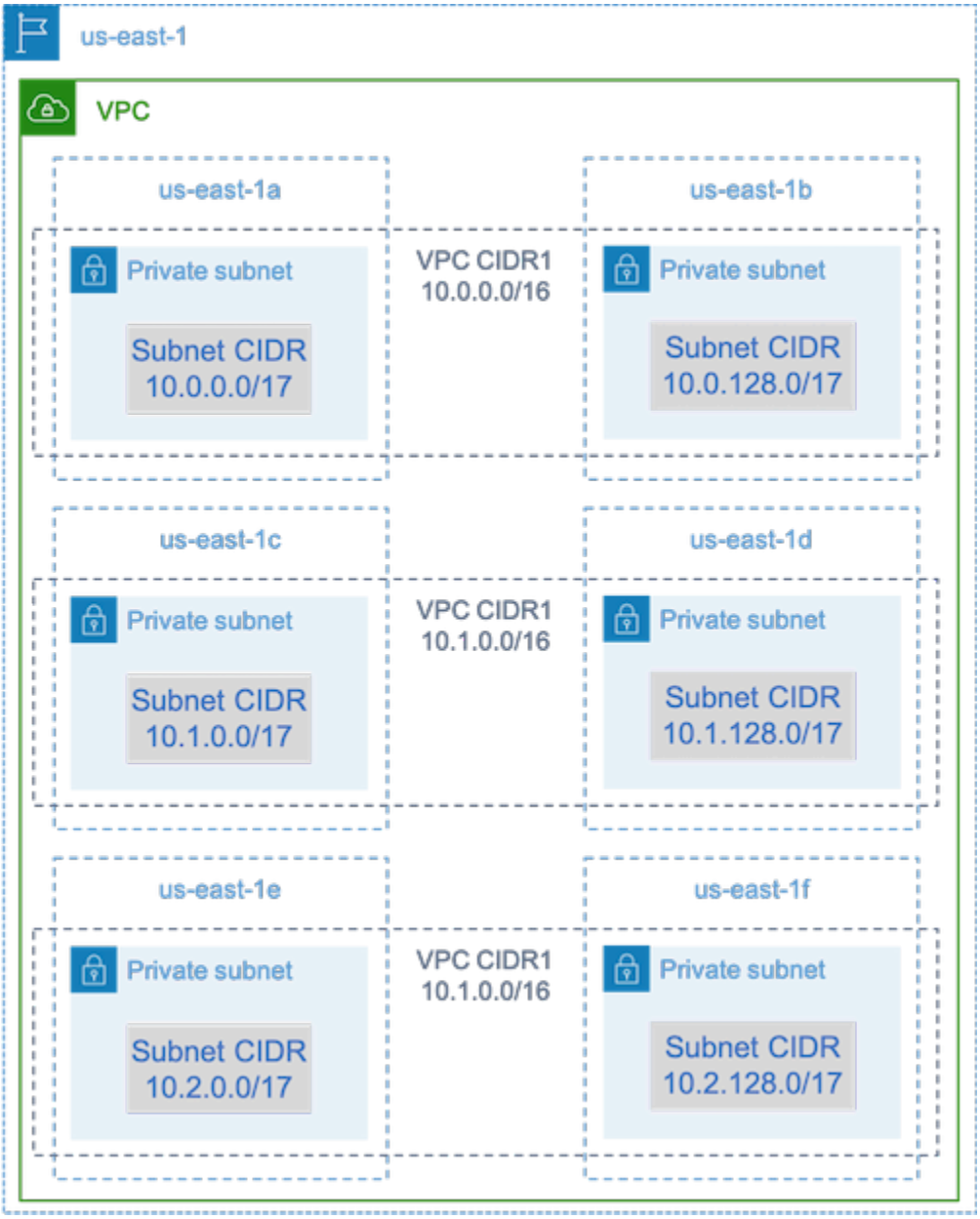
```
"retryStrategy": {
    "attempts": 5,
    "evaluateOnExit":
    [{
        "onStatusReason" :"Host EC2*",
        "action": "RETRY"
    },{
      "onReason" : "*"
        "action": "EXIT"
    }]
}
```

- Use o Painel de Interrupções Spot – Você pode usar o Painel de Interrupções Spot para monitorar interrupções de Spot. O aplicativo fornece métricas sobre instâncias spot do Amazon EC2 assumidas de novo e em cujas Zonas de Disponibilidade as instâncias spot estejam. Para obter mais informações, consulte Painel de Interrupções Spot

Erros Comuns e Solução de Problemas

Erros em AWS Batch ocorrem frequentemente no nível do aplicativo, ou são ocasionados por configurações de instância que não estejam em conformidade com os requisitos específicos obrigatórios do trabalho. Outros problemas incluem trabalhos presos no status `RUNNABLE` ou ambientes computacionais presos em um estado `INVALID`. Para obter mais informações sobre soluções de problemas de trabalhos presos no status `RUNNABLE`, consulte Trabalhos presos no status `RUNNABLE`. Para obter informações sobre solução de problemas em ambientes de computação em um estado `INVALID`, consulteAmbiente de computação do `INVALID`.

- Verifique as Cotas vCPU do Amazon EC2 Spot – Verifique se suas cotas de serviço atuais atendem aos requisitos do trabalho. Por exemplo, ao supor que sua cota de serviço atual seja de 256 vCPUs e que o trabalho exija 10.000 vCPUs. Nesse caso, a cota de serviço não está em conformidade com os requisitos do trabalho. Para obter mais informações e instruções sobre solução de problemas, consulte cotas de serviço do Amazon EC2 e Como aumento a cota de serviços dos meus recursos do Amazon EC2?.
- Trabalhos com falha antes da execução do aplicativo – Alguns trabalhos podem falhar devido a um erro `DockerTimeoutError` ou a um erro `CannotPullContainerError`. Para obter informações sobre solução de problemas, consulte Como resolver o erro “DockerTimeoutError” em AWS Batch.
- Endereços IP insuficientes – O número de endereçamento de IP na sua VPC e sub redes pode limitar o número de instâncias que você pode criar. Use Encaminhamento Entre Domínios Sem Classificação (CIDR) para fornecer mais endereços IP do que o obrigatório para a execução de suas workloads. Se necessário, você também pode compilar uma VPC dedicada com grande espaço de endereço. Por exemplo, você pode criar uma VPC com vários CIDRs em `10.x.0.0/16` e uma sub-rede em cada zona de disponibilidade com um CIDR de `10.x.y.0/17`. Neste exemplo, x está entre 1-4 e y é 0 ou 128. Essa configuração fornece 36.000 endereços IP em cada sub-rede.



- Verifique se as instâncias estão registradas no Amazon EC2 – Se você vê suas instâncias no console do Amazon EC2 mas não vê nenhuma instância de contêiner do Amazon Elastic Container Service em seu cluster do Amazon ECS, o atendente do Amazon ECS pode não estar instalado em uma imagem de máquina da Amazon (AMI). O atendente do Amazon ECS, a entrada de dados do Amazon EC2 em sua AM, ou o modelo de inicialização também podem não estar configurados corretamente. Para isolar a causa raiz, crie uma instância separada do Amazon EC2 ou conecte-se a uma instância existente usando SSH. Para obter mais informações, consulte Configuração do Atendente de Contêiner do Amazon ECS, Locais de Arquivo de Log do Amazon ECS e Recursos de computação de AMIs.

- Analise o painel AWS – Analise o Painel AWS para verificar se os estados esperados da tarefa e se o ambiente computacional estão em escala conforme o esperado. Você também pode analisar os logs de trabalho no CloudWatch.
- Verifique se sua instância foi criada – Se uma instância tiver sido criada, significa que seu ambiente de computação foi escalado conforme o esperado. Se suas instâncias não foram criadas, descubra as sub-redes associadas em seu ambiente de computação para alterá-las. Para obter mais informações, consulte Verificar uma Ação em Escala para um Grupo do Auto Scaling.

  Também recomendamos que você verifique se suas instâncias podem atender aos requisitos de trabalho relacionados. Por exemplo, um trabalho pode exigir 1 TiB de memória, mas o ambiente de computação utiliza um tipo de instância C5 limitado a 192 GB de memória.
- Verifique se suas instâncias estão sendo solicitadas por AWS Batch – Verifique o histórico do grupo do Auto Scaling para ter certeza de que se suas instâncias estão sendo solicitadas por AWS Batch. Esse é um indicador de como o Amazon EC2 tenta adquirir instâncias. Se você receber um estado de erro informando que o Amazon EC2 Spot não pode adquirir uma instância em uma Zona de Disponibilidade específica, talvez esta Zona de Disponibilidade não ofereça uma família de instâncias específica.
- Verifique se as instâncias estão registradas no Amazon ECS – Caso você veja suas instâncias no console do Amazon EC2, mas não veja nenhuma instância de contêiner do Amazon ECS em seu cluster do Amazon ECS, o atendente do Amazon ECS pode não estar instalado em uma imagem de máquina da Amazon (AMI). Além disso, o atendente do Amazon ECS, os Dados Amazon EC2 em sua AMI, ou o modelo de inicialização podem não estar configurados corretamente. Para isolar a causa raiz, crie uma instância separada do Amazon EC2 ou conecte-se a uma instância existente usando SSH. Para obter mais informações, consulte Arquivo de Configuração do Atendente do CloudWatch: Seção Logs, Locais de Arquivos de Log do Amazon ECS e Recursos de computação de AMIs.
- Abra um tíquete de suporte – Se ainda estiver enfrentando problemas após uma solução de problemas e tiver um Plano do Support, abra um tíquete de suporte. No tíquete de suporte, certifique-se de incluir informações sobre o problema, as especificações da workload, a configuração e os resultados do teste. Para obter mais informações, consulte Comparar PlanosAWS Support.
- Analise os fóruns de AWS Batch e HPC – Para obter mais informações, consulte os fóruns AWS Batch e HPC.
- Analise o Painel de Monitoramento de Runtime AWS Batch – Esse painel usa uma arquitetura de tecnologia sem servidor para capturar eventos do Amazon ECS, de AWS Batch, e do Amazon

EC2, fornecer insights sobre trabalhos e instâncias. Para obter mais informações, consulte AWS BatchSolução de Painéis de Monitoramento de Runtime.

Histórico do documento

A tabela a seguir descreve as mudanças importantes na documentação desde o lançamento inicial do AWS Batch. Também atualizamos a documentação com frequência para abordar os comentários enviados por você.

Alteração	Descrição	Data
Versões AWS Batch atualizadas do Amazon EKS suportadas	Atualizou as versões do Amazon EKS que AWS Batch suportam a remoção da versão 1.22.	11 de março de 2024
Versões AWS Batch atualizadas do Amazon EKS suportadas	Atualizou as versões do Amazon EKS que AWS Batch oferecem suporte para incluir a versão 1.29.	29 de fevereiro de 2024
Tentativas de trabalho automatizadas	Corrigiu a amostra de código.	29 de fevereiro de 2024
Adiciona suporte para trabalhos de vários contêineres para AWS Batch	Adiciona suporte para trabalhos de vários contêineres AWS Batch para Amazon Elastic Container Service, Amazon Elastic Kubernetes Service e. AWS Fargate	28 de fevereiro de 2024
Versões AWS Batch atualizadas do Amazon EKS suportadas	Atualizou as versões do Amazon EKS que AWS Batch oferecem suporte para incluir a versão 1.28	27 de janeiro de 2024
Atualizado BatchServiceRolePolicy e AWSBatchServiceRole	BatchServiceRolePolicy Atualizado para adicionar suporte para descrever o	5 de dezembro de 2023

	histórico e as Amazon EC2 Auto Scaling atividades de solicitações do Spot Fleet. AWSBatchServiceRole Atualizado para adicionar IDs de declaração, conceder AWS Batch permissões para `ec2:DescribeSpotFleetRequestHistory` `autoscaling:DescribeScalingActivities` e.	
AWS Batch no Amazon EKS	AWS Batch adiciona suporte para execução de trabalhos em clusters Amazon EKS.	25 de outubro de 2022
Prevenção confusa de delegados entre serviços para AWS Batch	AWS Batch agora fornece uma solução alternativa para o confuso problema de segurança adjunta, que surge quando uma entidade (um serviço ou uma conta) é coagida por outra entidade a realizar uma ação.	6 de junho de 2022

VPC endpoint de interface (AWS PrivateLink)	Foi adicionado suporte para configurar endpoints VPC de interface desenvolvidos por. AWS PrivateLink Isso significa que você pode criar uma conexão privada entre sua VPC AWS Batch sem precisar de acesso por meio de uma instância NAT, uma conexão VPN ou. AWS Direct Connect	15 de abril de 2022
Atualizações aprimoradas do ambiente de computação	AWS Batch atualizações aprimoradas de suporte para ambientes computacionais.	14 de abril de 2022
AWS atualizações de políticas gerenciadas - Atualização das políticas existentes	AWS Batch atualizou as políticas gerenciadas existente s.	6 de dezembro de 2021
Programação de compartil hamento justo	AWS Batch adiciona suporte para adicionar políticas de agendamento às filas de trabalhos.	9 de novembro de 2021
Amazon EFS	AWS Batch adiciona suporte para adicionar sistemas de arquivos Amazon EFS às suas definições de trabalho.	1º de abril de 2021
Perfil adicionado vinculado a serviço	AWS Batch adiciona a função AWSServiceRoleForB atchvinculada ao serviço.	10 de março de 2021
AWS Fargate apoio	AWS Batch adiciona suporte para execução de trabalhos nos recursos do Fargate.	3 de dezembro de 2020

Suporte do Amazon Linux 2	AWS Batch adiciona suporte à seleção automática de AMIs do Amazon Linux 2 no ambiente computacional usando os parâmetros de configuração do EC2.	24 de novembro de 2020
Estratégia de repetição aprimorada	AWS Batch aprimora a estratégia de repetição de empregos. Agora, os trabalhos podem ser repetidos ou interromper novas tentativa s combinando o `ExitCode`, `Reason` ou `StatusReason` de um trabalho com padrões.	20 de outubro de 2020
Marcação de recursos	AWS Batch adiciona suporte para adicionar tags de metadados aos seus ambientes computacionais, definições de tarefas, filas de tarefas e tarefas.	7 de outubro de 2020
Segredos	AWS Batch adiciona suporte para passar segredos para trabalhos.	1º de outubro de 2020
Registro em log	AWS Batch adiciona suporte para especificar drivers de log adicionais para trabalhos.	1º de outubro de 2020
Estratégias de alocação	AWS Batch adiciona suporte a várias estratégias para escolher tipos de instância.	16 de outubro de 2019
Suporte do EFA	AWS Batch adiciona suporte para dispositivos Elastic Fabric Adapter (EFA).	2 de agosto de 2019

Programação de GPUs	AWS Batch adiciona agendamento de GPU. Com esse recurso, você pode especificar o número de GPUs que cada trabalho exige e AWS Batch escala as instâncias de acordo.	4 de abril de 2019
Trabalhos em paralelo de vários nós	AWS Batch adiciona suporte para trabalhos paralelos de vários nós. Você pode usar esse recurso para executar trabalhos únicos que abrangem várias instâncias do Amazon EC2.	19 de novembro de 2018
Permissões em nível de recurso	AWS Batch oferece suporte a permissões em nível de recurso em várias operações de API.	12 de novembro de 2018
Suporte ao modelo de lançamento do Amazon EC2	AWS Batch adiciona suporte ao uso de modelos de lançamento com ambientes computacionais.	12 de novembro de 2018
AWS Batch tempos limite de trabalho	AWS Batch adiciona suporte para o tempo limite do trabalho. Com esse suporte, você pode configurar uma duração de tempo limite específica para seus trabalhos , de forma que, se um trabalho for executado por mais tempo do que o esperado, o AWS Batch encerre.	5 de abril de 2018

AWS Batch empregos como EventBridge alvos	AWS Batch os empregos são disponibilizados como EventBridge alvos. Ao criar regras simples, você pode combinar eventos e enviar trabalhos AWS Batch em resposta a eles.	1º de março de 2018
CloudTrail auditoria para AWS Batch	CloudTrail pode auditar chamadas feitas para ações de AWS Batch API.	10 de janeiro de 2018
Trabalhos de matriz	AWS Batch adiciona suporte para trabalhos de matriz. Você pode usar trabalhos de matriz para varredura de parâmetros e cargas de trabalho de Monte Carlo.	28 de novembro de 2017
AWS Batch Marcação expandida	AWS Batch expande o suporte para a função de marcação. Você pode usar essa função para especificar tags para instâncias spot do Amazon EC2 lançadas em ambientes de computação gerenciados.	26 de outubro de 2017
AWS Batch stream de eventos para EventBridge	AWS Batch adiciona o fluxo de eventos para EventBrid ge. Você pode usar o fluxo de AWS Batch eventos para receber notificações quase em tempo real sobre o estado dos trabalhos enviados às suas filas de trabalhos.	24 de outubro de 2017

Tentativas de trabalho automatizadas	AWS Batch adiciona suporte para novas tentativas de emprego. Com esta atualizaç ão, você pode aplicar uma estratégia de repetição a trabalhos e definições de tarefa que permite que seus trabalhos sejam automatic amente repetidos em caso de falha.	28 de março de 2017
AWS Batch disponibilidade geral	AWS Batch é introduzido, projetado como um meio para você executar cargas de trabalho de computação em lote no Nuvem AWS.	5 de janeiro de 2017

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