Manual do usuário

AWS AppConfig

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

AWS AppConfig: Manual do usuário

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

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

Table of Contents

O que AWS AppConfigé 1
Comece com AWS AppConfig 1
AWS AppConfig casos de uso 1
Visão geral dos benefícios 2
Como AWS AppConfig funciona 3
Preços para AWS AppConfig 5
AWS AppConfig cotas 5
Recursos adicionais do 6
Blogs 6
SDKs 6
Conf AWS AppConfig iguração 7
Inscreva-se para um Conta da AWS 7
Criar um usuário com acesso administrativo 7
Conceder acesso programático 9
Entendendo IPv6 o suporte 11
Configurar permissões para reversão automática 11
Etapa 1: criar a política de permissão para reversão com base em alarmes CloudWatch 12
Etapa 2: criar a função do IAM para reversão com base em alarmes CloudWatch 13
Etapa 3: Adicionar uma relação de confiança 14
Criando 16
Noções básicas sobre o perfil do IAM do perfil de configuração 18
Criar namespaces 20
Criando um AWS AppConfig aplicativo (console) 20
Criação de um AWS AppConfig aplicativo (linha de comando) 21
Criar ambientes 23
Criação de um AWS AppConfig ambiente (console) 23
Criação de um AWS AppConfig ambiente (linha de comando) 24
Criando um perfil de configuração no AWS AppConfig 26
Criação de um perfil de configuração de sinalizadores de atributos 30
Criação de um perfil de configuração de formato livre 63
Criar um perfil de configuração para fontes de dados não nativas 79
Implantação 81
Como trabalhar com estratégias de implantação 82
Usar estratégias de implantação predefinidas 85

Criar uma estratégia de implantação ... 87
Implantar uma configuração ... 91
Implantar uma configuração (console) ... 92
Implantar uma configuração (linha de comando) ... 93
Implantação com CodePipeline ... 96
Como funciona a integração ... 97
Reverter uma configuração ... 97
Recuperação ... 100
O que é AWS AppConfig Agente? ... 101
Como usar o AWS AppConfig Agent para recuperar dados de configuração ... 103
Usando o AWS AppConfig Agent com AWS Lambda ... 104
Usando o AWS AppConfig Agent com o Amazon EC2 e máquinas locais ... 216
Usando o AWS AppConfig Agent com o Amazon ECS e o Amazon EKS ... 236
Recuperar sinalizadores de atributos ... 257
Usar um manifesto para habilitar recursos de recuperação adicionais ... 260
Gerar um cliente usando a especificação OpenAPI ... 271
Trabalhando com o modo de desenvolvimento local do AWS AppConfig agente ... 274
Considerações sobre o uso de navegadores e dispositivos móveis ... 279
Dados de configuração e recuperação de sinalizadores ... 279
Autenticação e Amazon Cognito ... 280
Armazenamento em cache ... 280
Segmentação ... 281
Largura de banda (casos de uso em dispositivos móveis) ... 282
Casos de uso de sinalizadores adicionais ... 282
Recuperando dados de configuração sem AWS AppConfig o Agente ... 282
(Exemplo) Recuperando uma configuração chamando AWS AppConfig APIs ... 284
Estendendo AWS AppConfig os fluxos de trabalho ... 287
Entendendo AWS AppConfig as extensões ... 287
Etapa 1: determine o que você deseja fazer com as extensões ... 288
Etapa 2: determine quando você deseja que a extensão seja executada ... 289
Etapa 3: crie uma associação de extensão ... 290
Etapa 4: implante uma configuração e verifique se as ações da extensão são executadas ... 291
Trabalhando com extensões criadas AWS por autoria ... 291
Usando os eventos AWS AppConfig de implantação na EventBridge extensão Amazon ... 292
Usando os eventos AWS AppConfig de implantação na extensão Amazon SNS ... 294
Usando os eventos AWS AppConfig de implantação na extensão Amazon SQS ... 297

Usar a extensão Jira ... 299
Passo a passo: Criação de extensões personalizadas AWS AppConfig ... 305
Etapa 1: criar uma função Lambda para uma extensão personalizada AWS AppConfig ... 306
Etapa 2: configurar permissões para uma AWS AppConfig extensão personalizada ... 313
Etapa 3: criar uma AWS AppConfig extensão personalizada ... 314
Etapa 4: criar uma associação de extensão para uma AWS AppConfig extensão personalizada ... 319
Exemplos de código ... 321
Criar ou atualizar uma configuração de forma livre gravada no armazenamento de configuração hospedado ... 321
Criar um perfil de configuração para um segredo armazenado no Secrets Manager ... 324
Implantar um perfil de configuração ... 325
Usando o AWS AppConfig Agente para ler um perfil de configuração de formato livre ... 330
Usando o AWS AppConfig Agente para ler um sinalizador de recurso específico ... 332
Usando o AWS AppConfig Agente para recuperar um sinalizador de recurso com variantes ... 333
Usando a ação GetLatestConfiguration da API para ler um perfil de configuração de formato livre ... 335
Limpar o ambiente ... 346
Deletion protection (Proteção contra exclusão) ... 352
Ignorar ou forçar uma verificação de proteção contra exclusão ... 353
Segurança ... 355
Implemente o acesso de privilégio mínimo ... 355
Criptografia de dados em repouso para AWS AppConfig ... 356
AWS PrivateLink ... 361
Considerações ... 361
Como criar um endpoint de interface ... 362
Criar uma política de endpoint ... 362
Alternância de chaves do Secrets Manager ... 363
Configurando a rotação automática dos segredos do Secrets Manager implantados pelo AWS AppConfig ... 363
Monitoramento ... 366
CloudTrail troncos ... 367
AWS AppConfig eventos de dados em CloudTrail ... 368
AWS AppConfig eventos de gerenciamento em CloudTrail ... 370
AWS AppConfig exemplos de eventos ... 370
Métricas de registro para chamadas AWS AppConfig de planos de dados ... 372

Criando um alarme para uma CloudWatch métrica 374
Monitorar implantações para reversão automática 375
Métricas recomendadas para monitorar a reversão automática 376
Histórico do documento 383
..... cdxiv

O que AWS AppConfigé

AWS AppConfig sinalizadores de recursos e configurações dinâmicas ajudam os criadores de software a ajustar com rapidez e segurança o comportamento do aplicativo em ambientes de produção sem implantações completas de código. AWS AppConfig acelera a frequência de lançamento de software, melhora a resiliência do aplicativo e ajuda você a resolver problemas emergentes com mais rapidez.

Com sinalizadores de atributos, você pode liberar gradualmente novos recursos para os usuários e medir o impacto dessas mudanças antes de implantar totalmente os novos recursos para todos os usuários. Com sinalizadores operacionais e configurações dinâmicas, você pode atualizar listas de bloqueio, listas de permissões, limites de controle de utilização, verbosidade de registros em log e realizar outros ajustes operacionais para responder rapidamente a problemas nos ambientes de produção.

Comece com AWS AppConfig

O vídeo a seguir pode ajudar você a entender os recursos do AWS AppConfig.

Veja mais AWS vídeos no YouTube canal da Amazon Web Services.

AWS AppConfig casos de uso

AWS AppConfig oferece suporte a um amplo espectro de casos de uso:

- Sinalizadores e botões de atributos: libere com segurança novos recursos para seus clientes em um ambiente controlado. Reverta instantaneamente as alterações em caso de algum problema.
- Ajuste de aplicativos: introduza cuidadosamente as alterações nos aplicativos e, ao mesmo tempo, teste o impacto dessas mudanças com os usuários em ambientes de produção.
- Lista de permissões ou lista de bloqueio: controle o acesso a atributos premium ou bloqueie instantaneamente usuários específicos sem implantar um novo código.
- Armazenamento de configurações centralizado: mantenha seus dados de configuração organizados e consistentes em todos os workloads. Você pode usar AWS AppConfig para implantar dados de configuração armazenados no armazenamento de configuração AWS AppConfig hospedado AWS Secrets Manager, no Systems Manager Parameter Store ou no Amazon S3.

Visão geral dos benefícios

A breve visão geral a seguir descreve os benefícios do uso do AWS AppConfig.

Melhore a eficiência e libere as alterações mais rapidamente

O uso de sinalizadores de atributos com novos recursos acelera o processo de liberação de alterações nos ambientes de produção. Em vez de depender de ramificações de desenvolvimento de longa duração que exigem mesclagens complicadas antes do lançamento, os sinalizadores de atributos permitem escrever softwares usando desenvolvimento baseado em troncos. Os sinalizadores de recursos permitem que você implante com segurança o código de pré-lançamento em um CI/CD pipeline que está oculto para os usuários. Quando estiver pronto para lançar as alterações, você poderá atualizar o sinalizador de atributos sem implantar um novo código. Depois do lançamento, o sinalizador ainda poderá funcionar como uma chave de bloqueio para desativar um novo atributo ou recursos sem a necessidade de reverter a implantação do código.

Evite alterações ou falhas não intencionais com atributos de segurança integrados

AWS AppConfig oferece os seguintes recursos de segurança para ajudá-lo a evitar a ativação de sinalizadores de recursos ou a atualização de dados de configuração que possam causar falhas no aplicativo.

- Validadores: o validador garante que os dados de configuração estejam sintática e semanticamente corretos antes de implantar as alterações nos ambientes de produção.
- Estratégias de implantação: uma estratégia de implantação permite liberar lentamente as alterações nos ambientes de produção em questão de minutos ou horas.
- Monitoramento e reversão automática: AWS AppConfig integra-se à Amazon CloudWatch para monitorar alterações em seus aplicativos. Se o aplicativo não estiver íntegro devido a uma alteração incorreta na configuração e essa alteração disparar um alarme CloudWatch, reverterá AWS AppConfig automaticamente a alteração para minimizar o impacto nos usuários do aplicativo.

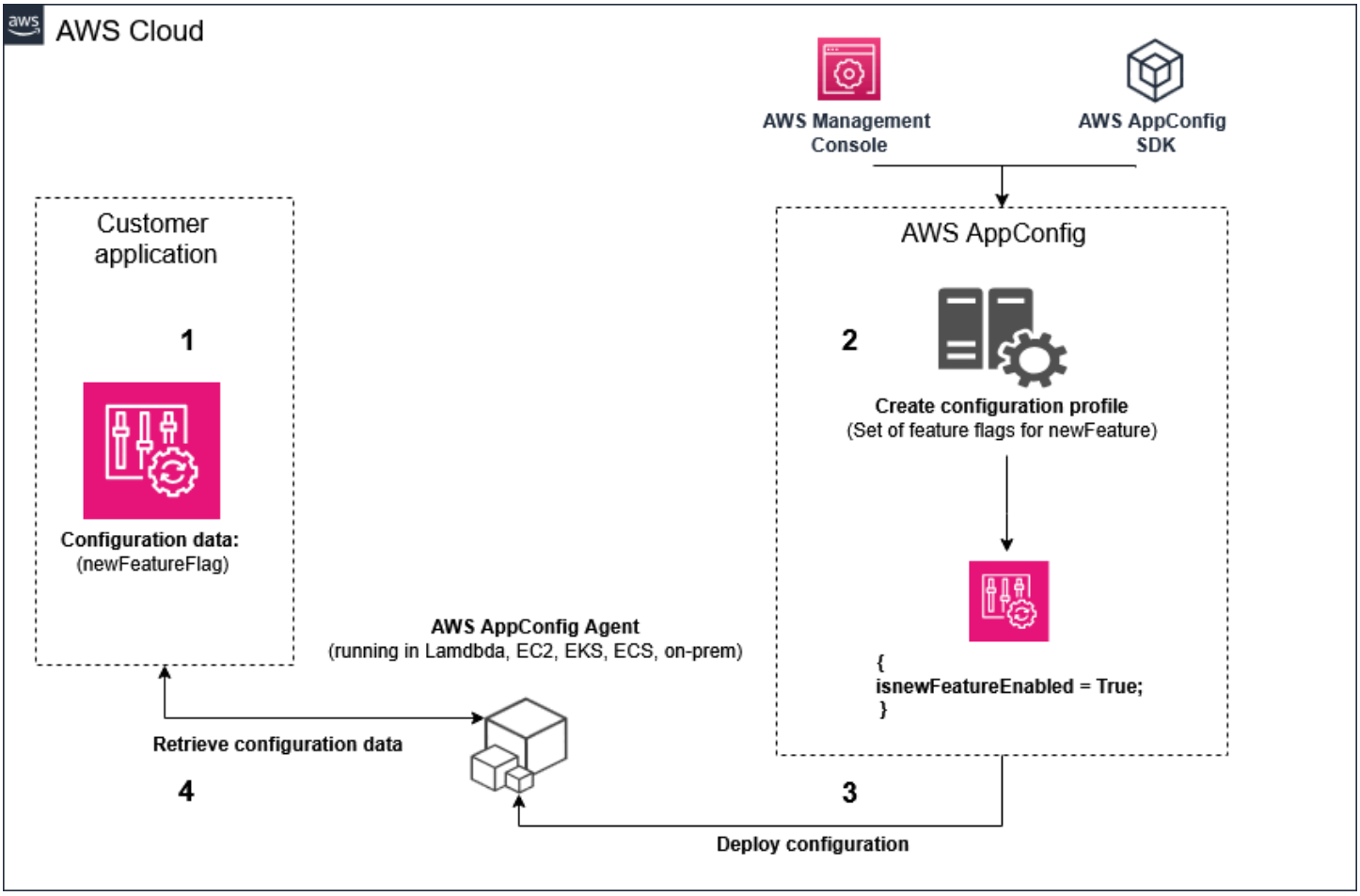
Implantações seguras e escaláveis para sinalizadores de atributos

AWS AppConfig se integra ao AWS Identity and Access Management (IAM) para fornecer acesso refinado e baseado em funções ao serviço. AWS AppConfig também se integra com AWS Key Management Service (AWS KMS) para criptografia e AWS CloudTrail auditoria. Antes de serem lançados para clientes externos, todos os controles de AWS AppConfig segurança

foram inicialmente desenvolvidos e validados por clientes internos que usam o serviço em grande escala.

Como AWS AppConfig funciona

Esta seção fornece uma descrição de alto nível de como AWS AppConfig funciona.



1. Identifique os valores de configuração no código que você deseja gerenciar AWS AppConfig

   Antes de criar um perfil de configuração no AWS AppConfig, recomendamos que você identifique os dados de configuração em seu código que você deseja gerenciar dinamicamente usando AWS AppConfig. Bons exemplos incluem sinalizadores ou botões de atributos, listas de permissões e bloqueios, verbosidade de registros em log, limites de serviço e regras de controle de utilização, para citar alguns. Esses tipos de configuração mudam com frequência e podem causar problemas quando não estão corretos.

Se seus dados de configuração já existirem na nuvem, por exemplo, no Parameter Store ou no Amazon S3, você pode aproveitar os recursos de AWS AppConfig validação, implantação e extensão para simplificar ainda mais o gerenciamento dos dados de configuração.

2. Crie um perfil de configuração no AWS AppConfig

Um perfil de configuração inclui, entre outras coisas, um URI que permite AWS AppConfig localizar seus dados de configuração em seu local armazenado e um tipo de perfil. AWS AppConfig suporta dois tipos de perfil de configuração: sinalizadores de recursos e configurações de forma livre. Ambos os tipos podem reduzir o risco e a complexidade do desenvolvimento e da implantação do software ao desvincular as versões dos atributos das implantações de código. Eles também permitem a entrega contínua e a mitigação de riscos por meio de distribuições em etapas. Além disso, os sinalizadores de recursos permitem testes em produção com usuários reais, enquanto as configurações de formato livre permitem que você recupere dados de configuração de outros serviços. AWS Ambos os tipos de perfil permitem iteração, experimentação, personalização e gerenciamento mais rápidos do ciclo de vida do software. Para obter mais informações sobre a criação de um perfil de configuração, consulte Criando um perfil de configuração no AWS AppConfig.

Um perfil de configuração também pode incluir validadores opcionais para garantir que seus dados de configuração estejam sintática e semanticamente corretos. AWS AppConfig executa uma verificação usando os validadores quando você inicia uma implantação. Se algum erro for detectado, a implantação será revertida para os dados de configuração anteriores.

Ao criar um perfil de configuração, você também cria um aplicativo no AWS AppConfig. Um aplicativo é um namespace ou um constructo organizacional, como uma pasta.

3. Implantar dados de configuração

Quando você inicia uma implantação, AWS AppConfig executa as seguintes tarefas:

1. Recupera os dados de configuração do armazenamento de dados subjacente usando o nome do caminho de localização no perfil de configuração.
2. Verifica se os dados de configuração estão sintática e semanticamente corretos usando os validadores especificados ao criar o perfil de configuração.
3. Envia uma cópia dos dados ao AWS AppConfig Agente para serem lidos pelo seu aplicativo. Essa cópia é chamada de dados implantados.

Para obter mais informações sobre como implantar uma configuração, consulte Implantando sinalizadores de recursos e dados de configuração no AWS AppConfig.

4. Recuperar a configuração

   Para recuperar os dados, seu aplicativo faz uma chamada HTTP para o servidor localhost em que o AWS AppConfig Agente armazenou em cache uma cópia local dos dados de configuração implantados. A recuperação de dados é um evento medido. AWS AppConfig O agente oferece suporte a vários casos de uso, conforme descrito emComo usar o AWS AppConfig Agent para recuperar dados de configuração.

   Se o AWS AppConfig Agente não for compatível com seu caso de uso, você poderá configurar seu aplicativo AWS AppConfig para pesquisar atualizações de configuração chamando diretamente as ações StartConfigurationSessione da GetLatestConfigurationAPI.

   Para obter mais informações sobre como recuperar uma configuração, consulte Recuperando sinalizadores de recursos e dados de configuração em AWS AppConfig.

Preços para AWS AppConfig

O preço do AWS AppConfig é pay-as-you-go baseado nos dados de configuração e na recuperação do sinalizador de recursos. Recomendamos usar o AWS AppConfig Agente para ajudar a otimizar os custos. Para obter mais informações, consulte AWS Systems Manager Preço.

AWS AppConfig cotas

Você pode visualizar informações sobre AWS AppConfig endpoints e cotas de serviço no. Referência geral da Amazon Web Services

> **Note**
>
> AWS AppConfig é uma capacidade de AWS Systems Manager.

Para obter informações sobre cotas para serviços que armazenam AWS AppConfig configurações, consulte. Noções básicas sobre cotas e limitações do armazenamento de configuração

Recursos adicionais do

Os recursos a seguir podem ajudar você a aprender mais sobre AWS AppConfig.

Blogs

Os blogs a seguir podem ajudar você a aprender mais sobre AWS AppConfig e seus recursos:

- Por que você deve usar AWS AppConfig
- Liberte o poder das bandeiras de recursos com AWS AppConfig
- Uso de sinalizadores de atributos do AWS AppConfig
- Práticas recomendadas para validar sinalizadores de AWS AppConfig recursos e dados de configuração

SDKs

Para obter informações sobre AWS AppConfig idiomas específicos SDKs, consulte os seguintes recursos:

- AWS Command Line Interface
- AWS SDK para .NET
- AWS SDK para C++
- AWS SDK para Go
- AWS SDK para Java V2
- AWS SDK para JavaScript
- AWS SDK para PHP V3
- AWS SDK para Python
- AWS SDK para Ruby V3

Conf AWS AppConfig iguração

Se você ainda não tiver feito isso, inscreva-se Conta da AWS e crie um usuário administrativo.

Inscreva-se para um Conta da AWS

Se você não tiver um Conta da AWS, conclua as etapas a seguir para criar um.

Para se inscrever em um Conta da AWS

1. Abra a https://portal.aws.amazon.com/billing/inscrição.
2. Siga as instruções online.

   Parte do procedimento de inscrição envolve receber uma chamada telefônica ou uma mensagem de texto e inserir um código de verificação pelo teclado do telefone.

   Quando você se inscreve em um Conta da AWS, um Usuário raiz da conta da AWSé criado. O usuário-raiz tem acesso a todos os Serviços da AWS e recursos na conta. Como prática recomendada de segurança, atribua o acesso administrativo a um usuário e use somente o usuário-raiz para executar tarefas que exigem acesso de usuário-raiz.

AWS envia um e-mail de confirmação após a conclusão do processo de inscrição. A qualquer momento, você pode visualizar a atividade atual da sua conta e gerenciar sua conta acessando https://aws.amazon.com/e escolhendo Minha conta.

Criar um usuário com acesso administrativo

Depois de se inscrever em um Conta da AWS, proteja seu Usuário raiz da conta da AWS Centro de Identidade do AWS IAM, habilite e crie um usuário administrativo para que você não use o usuário root nas tarefas diárias.

Proteja seu Usuário raiz da conta da AWS

1. Faça login Console de gerenciamento da AWScomo proprietário da conta escolhendo Usuário raiz e inserindo seu endereço de Conta da AWS e-mail. Na próxima página, insira a 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. Habilite a autenticação multifator (MFA) para o usuário-raiz.

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

Criar um usuário com acesso administrativo

1. Habilita o Centro de Identidade do IAM.

   Para obter instruções, consulte Habilitar o Centro de Identidade do AWS IAM no Guia do usuário do Centro de Identidade do AWS IAM .

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

   Para ver um tutorial sobre como usar o Diretório do Centro de Identidade do IAM como fonte de identidade, consulte Configurar o acesso do usuário com o padrão Diretório do Centro de Identidade do IAM no Guia Centro de Identidade do AWS IAM do usuário.

Iniciar sessão como o usuário com acesso administrativo

- Para fazer login com o seu usuário do Centro de Identidade do IAM, use o URL de login enviado ao seu endereço de e-mail quando o usuário do Centro de Identidade do IAM foi criado.

  Para obter ajuda para fazer login usando um usuário do IAM Identity Center, consulte Como fazer login no portal de AWS acesso no Guia Início de Sessão da AWS do usuário.

Atribuir acesso a usuários adicionais

1. No Centro de Identidade do IAM, crie um conjunto de permissões que siga as práticas recomendadas de aplicação de permissões com privilégio mínimo.

   Para obter instruções, consulte Criar um conjunto de permissões no Guia do usuário do Centro de Identidade do AWS IAM .

2. Atribua usuários a um grupo e, em seguida, atribua o acesso de logon único ao grupo.

   Para obter instruções, consulte Adicionar grupos no Guia do usuário do Centro de Identidade do AWS IAM .

Conceder acesso programático

Os usuários precisam de acesso programático se quiserem interagir com pessoas AWS fora do Console de gerenciamento da AWS. A forma de conceder acesso programático depende do tipo de usuário que está acessando AWS.

Para conceder acesso programático aos usuários, selecione uma das seguintes opções:

Qual usuário precisa de acesso programático?	Para	Por
IAM	(Recomendado) Use as credenciais do console como credenciais temporárias para assinar solicitações programáticas para o AWS CLI, AWS SDKs ou. AWS APIs	Siga as instruções da interface que deseja utilizar. • Para o AWS CLI, consulte Login para desenvolvimento AWS local no Guia AWS Command Line Interface do usuário. • Para AWS SDKs isso, consulte Login para desenvolvimento AWS local no Guia de referência de ferramentas AWS SDKs e ferramentas.
Identidade da força de trabalho (Usuários gerenciados no Centro de Identidade do IAM)	Use credenciais temporárias para assinar solicitações programáticas para o AWS CLI AWS SDKs, ou. AWS APIs	Siga as instruções da interface que deseja utilizar. • Para o AWS CLI, consulte Configurando o AWS CLI para uso Centro de Identidade do AWS IAM no Guia do AWS Command Line Interface usuário. • Para AWS SDKs, ferramentas e AWS APIs, consulte a autenticação do IAM Identity

Qual usuário precisa de acesso programático?	Para	Por
		Center no Guia de referênci a de ferramentas AWS SDKs e ferramentas.
IAM	Use credenciais temporári as para assinar solicitações programáticas para o AWS CLI AWS SDKs, ou. AWS APIs	Siga as instruções em Como usar credenciais temporárias com AWS recursos no Guia do usuário do IAM.
IAM	(Não recomendado) Use credenciais de longo prazo para assinar solicitaç ões programáticas para o AWS CLI, AWS SDKs, ou. AWS APIs	Siga as instruções da interface que deseja utilizar. • Para isso AWS CLI, consulte Autenticação usando credenciais de usuário do IAM no Guia do AWS Command Line Interface usuário. • Para ferramentas AWS SDKs e ferramentas, consulte Autenticar usando credenciais de longo prazo no Guia de referência de ferramentas AWS SDKs e ferramentas. • Para isso AWS APIs, consulte Gerenciamento de chaves de acesso para usuários do IAM no Guia do usuário do IAM.

Entendendo IPv6 o suporte

Tudo AWS AppConfig APIs com suporte IPv4 e IPv6 chamadas completos.

Ambiente de gerenciamento APIs

Use o seguinte endpoint para IPv4 chamadas de IPv6 pilha dupla para o plano de controle:

```
appconfig.Region.api.aws
```

Por exemplo: appconfig.us-east-1.api.aws

IPv4 Somente para isso, use o seguinte URL:

```
appconfig.Region.amazonaws.com
```

Plano de dados APIs

Para chamadas de pilha dupla para o plano de dados, use o seguinte endpoint:

```
appconfigdata.Region.api.aws
```

Por exemplo: appconfig.us-east-1.api.aws

IPv4 Somente para isso, use o seguinte URL:

```
appconfigdata.Region.amazonaws.com
```

> **Note**
>
> Para obter mais informações, consulte endpoints e cotas do AWS AppConfig na Referência geral da AWS.

Configurar permissões para reversão automática

Você pode configurar AWS AppConfig para reverter para uma versão anterior de uma configuração em resposta a um ou mais CloudWatch alarmes da Amazon. Ao configurar uma implantação

para responder aos CloudWatch alarmes, você especifica uma função AWS Identity and Access Management (IAM). AWS AppConfig requer essa função para que possa monitorar CloudWatch os alarmes. Esse procedimento é opcional, mas altamente recomendado.

> **Note**
>
> Observe as seguintes informações:
>
> - O perfil do IAM deve pertencer à sua conta atual. Por padrão, só é AWS AppConfig possível monitorar alarmes de propriedade da conta corrente.
> - Para obter informações sobre métricas a serem monitoradas e como configurar AWS AppConfig a reversão automática, consulte. Monitorar implantações para reversão automática

Use os procedimentos a seguir para criar uma função do IAM que permita AWS AppConfig a reversão com base em CloudWatch alarmes. Esta seção inclui os seguintes procedimentos.

1. Etapa 1: criar a política de permissão para reversão com base em alarmes CloudWatch
2. Etapa 2: criar a função do IAM para reversão com base em alarmes CloudWatch
3. Etapa 3: Adicionar uma relação de confiança

Etapa 1: criar a política de permissão para reversão com base em alarmes CloudWatch

Use o procedimento a seguir para criar uma política do IAM que dê AWS AppConfig permissão para chamar a ação da `DescribeAlarms` API.

Para criar uma política de permissão do IAM para reversão com base em alarmes CloudWatch

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas e, em seguida, Criar política.
3. Na página Criar política, escolha a guia JSON.
4. Substitua o conteúdo padrão na guia JSON pela política de permissão a seguir e, em seguida, escolha Próximo: Tags.

> **Note**
>
> Para retornar informações sobre alarmes CloudWatch compostos, é necessário atribuir * permissões à operação da DescribeAlarmsAPI, conforme mostrado aqui. Não será possível exibir informações sobre alarmes compostos se a permissão `DescribeAlarms` tiver um escopo mais limitado.

JSON

```
{
        "Version":"2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "cloudwatch:DescribeAlarms"
                ],
                "Resource": "*"
            }
        ]
    }
```

5. Insira tags para essa função e escolha Next: Review (Próximo: revisão).
6. Na página Revisão, insira **`SSMCloudWatchAlarmDiscoveryPolicy`** no campo Nome.
7. Selecione Criar política. O sistema retorna para a página Policies (Políticas).

Etapa 2: criar a função do IAM para reversão com base em alarmes CloudWatch

Use o procedimento a seguir para criar um perfil do IAM e atribuir a ele a política que você criou no procedimento anterior.

Para criar uma função do IAM para reversão com base em alarmes CloudWatch

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Roles e Create role.

3. Em Select type of trusted entity (Selecionar o tipo de entidade confiável), escolha AWS service (serviço).
4. Imediatamente em Escolher o serviço que usará essa função, selecione O EC2 permite que instâncias do EC2 chamem os serviços da AWS em seu nome e, em seguida, escolha Próximo: permissões.
5. Na página Política de permissões anexadas, pesquise por SSMCloudWatchAlarmDiscoveryPolicy.
6. Selecione essa política e escolha Next: Tags (Próximo: tags).
7. Insira tags para essa função e escolha Next: Review (Próximo: revisão).
8. Na página Criar função, insira **`SSMCloudWatchAlarmDiscoveryRole`** no campo RNome da função e escolha Criar função.
9. Na página Roles (Funções), escolha a função recém-criada. A página Summary é aberta.

Etapa 3: Adicionar uma relação de confiança

Use o procedimento a seguir para configurar a função que você acabou de criar para confiar no AWS AppConfig.

Para adicionar uma relação de confiança para AWS AppConfig

1. Na página Summary da função que você acabou de criar, escolha a guia Trust Relationships e, em seguida, Edit Trust Relationship.
2. Edite a política para incluir somente "`appconfig.amazonaws.com`", conforme mostrado no exemplo a seguir:

   JSON

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

```
    ]
}
```

3. Selecione Atualizar política de confiança.

Criação de sinalizadores de recursos e dados de configuração de formato livre no AWS AppConfig

Os tópicos desta seção ajudam você a realizar as tarefas a seguir no AWS AppConfig. Essas tarefas criam artefatos importantes para a implantação de dados de configuração.

1. Criar um namespace para o aplicativo

   Para criar um namespace de aplicativo, você cria um AWS AppConfig artefato chamado aplicativo. Um aplicativo é simplesmente uma estrutura organizacional, como uma pasta.

2. Criar ambientes

   Para cada AWS AppConfig aplicativo, você define um ou mais ambientes. Um ambiente é um grupo lógico de AWS AppConfig destinos de implantação, como aplicativos em um `Production` ambiente `Beta` OR. Também é possível definir ambientes para subcomponentes de aplicativos, como os componentes `AWS Lambda functions`, `Containers`, `Web`, `Mobile` e `Back-end`.

   Você pode configurar CloudWatch alarmes da Amazon para cada ambiente para reverter automaticamente alterações problemáticas na configuração. O sistema monitora os alarmes durante uma implantação de configuração. Se um alarme for acionado, o sistema reverterá a configuração.

3. Criar um perfil de configuração

   Dados de configuração são um conjunto de configurações que influenciam o comportamento da aplicação. Um perfil de configuração inclui, entre outras coisas, um URI que permite AWS AppConfig localizar seus dados de configuração em seu local armazenado e um tipo de configuração. AWS AppConfig suporta os seguintes tipos de perfis de configuração:

   - Sinalizadores de recursos: você pode usar sinalizadores de recursos para ativar ou desativar recursos em seus aplicativos ou para configurar características diferentes dos recursos do seu aplicativo usando atributos de sinalizadores. AWS AppConfig armazena configurações de sinalizadores de recursos no repositório de configurações AWS AppConfig hospedado em um formato de sinalizador de recurso que contém dados e metadados sobre seus sinalizadores e os atributos do sinalizador. O URI para configurações do sinalizador de atributos é simplesmente `hosted`.
   - Configurações de formato livre: uma configuração de formato livre pode armazenar dados em qualquer uma das seguintes ferramentas e do Systems Serviços da AWS Manager:

- AWS AppConfig armazenamento de configuração hospedado
- Amazon Simple Storage Service
- AWS CodePipeline
- AWS Secrets Manager
- AWS Systems Manager Armazenamento de parâmetros (SSM)
- Armazenamento de documentos do SSM

> **Note**
>
> Se possível, recomendamos hospedar seus dados de configuração no armazenamento de configuração AWS AppConfig hospedado, pois ele oferece mais recursos e aprimoramentos.

4. (Opcional, mas recomendado) Criar sinalizadores de atributos multivariante

   AWS AppConfig oferece sinalizadores de recursos básicos, que (se ativados) retornam um conjunto específico de dados de configuração por solicitação. Para oferecer melhor suporte aos casos de uso de segmentação de usuários e divisão de tráfego, AWS AppConfig também oferece sinalizadores de recursos de várias variantes, que permitem definir um conjunto de possíveis valores de sinalização a serem retornados para uma solicitação. Também é possível configurar diferentes status (habilitado ou desabilitado) para sinalizadores multivariante. Ao solicitar um sinalizador configurado com variantes, seu aplicativo fornece um contexto que é AWS AppConfig avaliado em relação a um conjunto de regras definidas pelo usuário. Dependendo do contexto especificado na solicitação e das regras definidas para a variante, AWS AppConfig retorna valores de sinalizadores diferentes para o aplicativo.

Tópicos

- Noções básicas sobre o perfil do IAM do perfil de configuração
- Criando um namespace para seu aplicativo no AWS AppConfig
- Criando ambientes para seu aplicativo em AWS AppConfig
- Criando um perfil de configuração no AWS AppConfig

Noções básicas sobre o perfil do IAM do perfil de configuração

É possível criar o perfil do IAM que concede acesso aos dados de configuração usando o AWS AppConfig. Ou pode criar o perfil do IAM você mesmo. Se você criar a função usando AWS AppConfig, o sistema cria a função e especifica uma das seguintes políticas de permissões, dependendo do tipo de fonte de configuração que você escolher.

A origem da configuração é um segredo do Secrets Manager

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
             ],
            "Resource": [
                "arn:aws:secretsmanager:us-
east-1:111122223333:secret:secret_name-a1b2c3"
            ]
        }
    ]
}
```

A origem da configuração é um parâmetro do Parameter Store

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameter"
            ],
            "Resource": [
```

```
                "arn:aws:ssm:us-east-1:111122223333:parameter/parameter_name"
            ]
        }
    ]
    }
```

A origem da configuração é um documento do SSM

JSON

```
{

    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetDocument"
            ],
            "Resource": [
                "arn:aws:ssm:us-east-1:111122223333:document/document_name"
            ]
        }
    ]
}
```

Se você criar a função usando AWS AppConfig, o sistema também criará a seguinte relação de confiança para a função.

JSON

```
{

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

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

Criando um namespace para seu aplicativo no AWS AppConfig

Os procedimentos desta seção ajudam você a criar um AWS AppConfig artefato chamado aplicativo. Um aplicativo é simplesmente uma estrutura organizacional, como uma pasta, que identifica o namespace do aplicativo. Essa construção organizacional tem um relacionamento com uma certa unidade de código executável. Por exemplo, você pode criar um aplicativo chamado MyMobileApp para organizar e gerenciar dados de configuração para um aplicativo móvel instalado por seus usuários. Você deve criar esses artefatos antes de usá-los AWS AppConfig para implantar e recuperar sinalizadores de recursos ou dados de configuração de formato livre.

O procedimento a seguir oferece a opção de associar uma extensão a um perfil de configuração de sinalizador de atributos. Uma extensão aumenta sua capacidade de injetar lógica ou comportamento em diferentes pontos durante o AWS AppConfig fluxo de trabalho de criação ou implantação de uma configuração. Para obter mais informações, consulte Entendendo AWS AppConfig as extensões.

> **Note**
>
> Você pode usar AWS CloudFormation para criar AWS AppConfig artefatos, incluindo aplicativos, ambientes, perfis de configuração, implantações, estratégias de implantação e versões de configuração hospedadas. Para obter mais informações, consulte AWS AppConfig resource type reference no Guia do usuário do AWS CloudFormation .

Tópicos

- Criando um AWS AppConfig aplicativo (console)
- Criação de um AWS AppConfig aplicativo (linha de comando)

Criando um AWS AppConfig aplicativo (console)

Use o procedimento a seguir para criar um AWS AppConfig aplicativo usando o AWS Systems Manager console.

Como criar uma aplicação do

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, escolha Aplicações e Criar aplicação.
3. Em Name (Nome), insira um nome para o aplicativo.
4. Em Description (Descrição), insira informações sobre o aplicativo.
5. (Opcional) Na seção Extensões, selecione uma extensão na lista. Para obter mais informações, consulte Entendendo AWS AppConfig as extensões.
6. (Opcional) Na seção Tags, insira uma chave e um valor opcional. Você pode especificar um máximo de 50 tags para um recurso.
7. Selecione Criar aplicativo.

AWS AppConfig cria o aplicativo e, em seguida, exibe a guia Ambientes. Vá para Criando ambientes para seu aplicativo em AWS AppConfig.

Criação de um AWS AppConfig aplicativo (linha de comando)

O procedimento a seguir descreve como usar o AWS CLI (no Linux ou no Windows) ou Ferramentas da AWS para PowerShell criar um AWS AppConfig aplicativo.

Para criar um aplicativo passo a passo

1. Abra AWS CLI o.
2. Execute o seguinte comando para criar um aplicativo.

   Linux

   ```
   aws appconfig create-application \
     --name A_name_for_the_application \
     --description A_description_of_the_application \
     --tags User_defined_key_value_pair_metadata_for_the_application
   ```

   Windows

   ```
   aws appconfig create-application ^
     --name A_name_for_the_application ^
     --description A_description_of_the_application ^
   ```

```
  --tags User_defined_key_value_pair_metadata_for_the_application
```

PowerShell

```
New-APPCApplication `
  -Name Name_for_the_application `
  -Description Description_of_the_application `
  -Tag Hashtable_type_user_defined_key_value_pair_metadata_for_the_application
```

O sistema retorna informações como estas.

Linux

```
{
   "Id": "Application ID",
   "Name": "Application name",
   "Description": "Description of the application"
}
```

Windows

```
{
   "Id": "Application ID",
   "Name": "Application name",
   "Description": "Description of the application"
}
```

PowerShell

```
ContentLength    : Runtime of the command
Description      : Description of the application
HttpStatusCode   : HTTP Status of the runtime
Id               : Application ID
Name             : Application name
ResponseMetadata : Runtime Metadata
```

Criando ambientes para seu aplicativo em AWS AppConfig

Para cada AWS AppConfig aplicativo, você define um ou mais ambientes. Um ambiente é um grupo lógico de AppConfig destinos de implantação, como aplicativos em um `Beta` `Production` ambiente, AWS Lambda funções ou contêineres. Também é possível definir ambientes para subcomponentes de aplicativos, como os componentes `Web`, `Mobile` e `Back-end`. Você pode configurar os CloudWatch alarmes da Amazon para cada ambiente. O sistema monitora os alarmes durante uma implantação de configuração. Se um alarme for acionado, o sistema reverterá a configuração.

Antes de começar

Se você quiser permitir AWS AppConfig a reversão de uma configuração em resposta a um CloudWatch alarme, deverá configurar uma função AWS Identity and Access Management (IAM) com permissões para permitir a resposta AWS AppConfig aos CloudWatch alarmes. Escolha essa função no procedimento a seguir. Para obter mais informações, consulte Configurar permissões para reversão automática.

Tópicos

- Criação de um AWS AppConfig ambiente (console)
- Criação de um AWS AppConfig ambiente (linha de comando)

Criação de um AWS AppConfig ambiente (console)

Use o procedimento a seguir para criar um AWS AppConfig ambiente usando o AWS Systems Manager console.

Para criar um ambiente

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, selecione Aplicações e escolha o nome de uma aplicação para abrir a página de detalhes.
3. Selecione Ambientes e escolha Criar ambiente.
4. Em Name (Nome), insira um nome para o ambiente.
5. Em Description (Descrição), insira informações sobre o ambiente.

6. (Opcional) Na seção Monitores, escolha o campo perfil do IAM e, depois, escolha o perfil do IAM com permissão para chamar `cloudwatch:DescribeAlarms` nas métricas que você quer monitorar para alarmes.
7. Na lista de CloudWatch alarmes, insira nos Amazon Resource Names (ARNs) uma ou mais métricas para monitorar. AWS AppConfig reverte a implantação da configuração se uma dessas métricas entrar em um `ALARM` estado. Para ter informações sobre as métricas recomendadas, consulte Monitorar implantações para reversão automática.
8. (Opcional) Na seção Associar extensões, selecione uma extensão na lista. Para obter mais informações, consulte Entendendo AWS AppConfig as extensões.
9. (Opcional) Na seção Tags, insira uma chave e um valor opcional. Você pode especificar um máximo de 50 tags para um recurso.
10. Selecione Criar ambiente.

AWS AppConfig cria o ambiente e, em seguida, exibe a página de detalhes do ambiente. Vá para Criando um perfil de configuração no AWS AppConfig.

Criação de um AWS AppConfig ambiente (linha de comando)

O procedimento a seguir descreve como usar o AWS CLI (no Linux ou no Windows) ou Ferramentas da AWS para PowerShell criar um AWS AppConfig ambiente.

Para criar um ambiente passo a passo

1. Abra AWS CLI o.
2. Execute o seguinte comando para criar um ambiente.

   Linux

```
aws appconfig create-environment \
  --application-id The_application_ID \
  --name A_name_for_the_environment \
  --description A_description_of_the_environment \
  --monitors
 "AlarmArn=ARN_of_the_Amazon_CloudWatch_alarm,AlarmArnRole=ARN_of_the_IAM
 role_for_AWS AppConfig_to_monitor_AlarmArn" \
  --tags User_defined_key_value_pair_metadata_of_the_environment
```

Windows

```
aws appconfig create-environment ^
  --application-id The_application_ID ^
  --name A_name_for_the_environment ^
  --description A_description_of_the_environment ^
  --monitors
 "AlarmArn=ARN_of_the_Amazon_CloudWatch_alarm,AlarmArnRole=ARN_of_the_IAM
 role_for_AWS AppConfig_to_monitor_AlarmArn" ^
  --tags User_defined_key_value_pair_metadata_of_the_environment
```

PowerShell

```
New-APPCEnvironment `
  -Name Name_for_the_environment `
  -ApplicationId The_application_ID
  -Description Description_of_the_environment `
  -Monitors
 @{"AlarmArn=ARN_of_the_Amazon_CloudWatch_alarm,AlarmArnRole=ARN_of_the_IAM
 role_for_AWS AppConfig_to_monitor_AlarmArn"} `
  -Tag Hashtable_type_user_defined_key_value_pair_metadata_of_the_environment
```

O sistema retorna informações como estas.

Linux

```
{
   "ApplicationId": "The application ID",
   "Id": "The_environment ID",
   "Name": "Name of the environment",
   "State": "The state of the environment",
   "Description": "Description of the environment",

   "Monitors": [
      {
         "AlarmArn": "ARN of the Amazon CloudWatch alarm",
         "AlarmRoleArn": "ARN of the IAM role for AppConfig to monitor AlarmArn"
      }
   ]
}
```

Windows

```
{
   "ApplicationId": "The application ID",
   "Id": "The environment ID",
   "Name": "Name of the environment",
   "State": "The state of the environment"
   "Description": "Description of the environment",

   "Monitors": [
      {
         "AlarmArn": "ARN of the Amazon CloudWatch alarm",
         "AlarmRoleArn": "ARN of the IAM role for AppConfig to monitor AlarmArn"
      }
   ]
}
```

PowerShell

```
ApplicationId     : The application ID
ContentLength     : Runtime of the command
Description       : Description of the environment
HttpStatusCode    : HTTP Status of the runtime
Id                : The environment ID
Monitors          : {ARN of the Amazon CloudWatch alarm, ARN of the IAM role for
 AppConfig to monitor AlarmArn}
Name              : Name of the environment
Response Metadata : Runtime Metadata
State             : State of the environment
```

Vá para Criando um perfil de configuração no AWS AppConfig.

Criando um perfil de configuração no AWS AppConfig

Dados de configuração são um conjunto de configurações que influenciam o comportamento da aplicação. Um perfil de configuração inclui, entre outras coisas, um URI que permite AWS AppConfig localizar seus dados de configuração em seu local armazenado e um tipo de configuração. AWS AppConfig suporta os seguintes tipos de perfis de configuração:

- Sinalizadores de recursos: você pode usar sinalizadores de recursos para ativar ou desativar recursos em seus aplicativos ou para configurar características diferentes dos recursos do seu aplicativo usando atributos de sinalizadores. AWS AppConfig armazena configurações de sinalizadores de recursos no repositório de configurações AWS AppConfig hospedado em um formato de sinalizador de recurso que contém dados e metadados sobre seus sinalizadores e os atributos do sinalizador. O URI para configurações do sinalizador de atributos é simplesmente `hosted`.
- Configurações de formato livre: uma configuração de formato livre pode armazenar dados em qualquer uma das seguintes ferramentas e do Systems Serviços da AWS Manager:
  - AWS AppConfig armazenamento de configuração hospedado
  - Amazon Simple Storage Service
  - AWS CodePipeline
  - AWS Secrets Manager
  - AWS Systems Manager Armazenamento de parâmetros (SSM)
  - Armazenamento de documentos do SSM

> **Note**
>
> Se possível, recomendamos hospedar seus dados de configuração no armazenamento de configuração AWS AppConfig hospedado, pois ele oferece mais recursos e aprimoramentos.

Confira alguns exemplos de dados de configuração para compreender melhor os diferentes tipos de dados de configuração e como eles podem ser usados em um sinalizador de atributos ou em um perfil de configuração de forma livre.

Dados de configuração de sinalizador de atributos

A configuração de sinalizadores de atributos a seguir habilita ou desabilita pagamentos em dispositivos móveis e pagamentos padrão por região.

JSON

```
{
  "allow_mobile_payments": {
    "enabled": false
  },
```

```
  "default_payments_per_region": {
    "enabled": true
  }
}
```

YAML

```
---
allow_mobile_payments:
  enabled: false
default_payments_per_region:
  enabled: true
```

Dados de configuração operacionais

A configuração de forma livre a seguir impõe limites sobre como uma aplicação processa solicitações.

JSON

```
{
  "throttle-limits": {
    "enabled": "true",
    "throttles": [
      {
        "simultaneous_connections": 12
      },
      {
        "tps_maximum": 5000
      }
    ],
    "limit-background-tasks": [
      true
    ]
  }
}
```

YAML

```
---
throttle-limits:
```

```
  enabled: 'true'
  throttles:
  - simultaneous_connections: 12
  - tps_maximum: 5000
  limit-background-tasks:
  - true
```

Dados de configuração da lista de controle de acesso

Os dados de configuração de forma livre da lista de controle de acesso a seguir especificam quais usuários ou grupos podem acessar uma aplicação.

JSON

```
{
  "allow-list": {
    "enabled": "true",
    "cohorts": [
      {
        "internal_employees": true
      },
      {
        "beta_group": false
      },
      {
        "recent_new_customers": false
      },
      {
        "user_name": "Jane_Doe"
      },
      {
        "user_name": "John_Doe"
      }
    ]
  }
}
```

YAML

```
---
allow-list:
  enabled: 'true'
```

```
cohorts:
- internal_employees: true
- beta_group: false
- recent_new_customers: false
- user_name: Jane_Doe
- user_name: Ashok_Kumar
```

Tópicos

- Criando um perfil de configuração de sinalizador de recurso no AWS AppConfig
- Criando um perfil de configuração de formato livre no AWS AppConfig
- Criar um perfil de configuração para fontes de dados não nativas

Criando um perfil de configuração de sinalizador de recurso no AWS AppConfig

Você pode usar sinalizadores de recursos para ativar ou desativar recursos em seus aplicativos ou para configurar características diferentes dos recursos do seu aplicativo usando atributos de sinalizadores. AWS AppConfig armazena configurações de sinalizadores de recursos no repositório de configurações AWS AppConfig hospedado em um formato de sinalizador de recurso que contém dados e metadados sobre seus sinalizadores e os atributos do sinalizador.

> **Note**
>
> Ao criar um perfil de configuração do sinalizador de recurso, você pode criar um sinalizador de recurso básico como parte do fluxo de trabalho do perfil de configuração. AWS AppConfig também oferece suporte a sinalizadores de recursos de várias variantes. Os sinalizadores de atributos multivariante permitem definir um conjunto de possíveis valores de sinalizador a serem exibidos para uma solicitação. Ao solicitar um sinalizador configurado com variantes, seu aplicativo fornece um contexto que é AWS AppConfig avaliado em relação a um conjunto de regras definidas pelo usuário. Dependendo do contexto especificado na solicitação e das regras definidas para a variante, AWS AppConfig retorna valores de sinalizadores diferentes para o aplicativo.
>
> Para criar sinalizadores de atributos multivariante, primeiro crie um perfil de configuração e, depois, edite todos os sinalizadores no perfil de configuração para adicionar variantes. Para obter mais informações, consulte Criar sinalizadores de atributos multivariante.

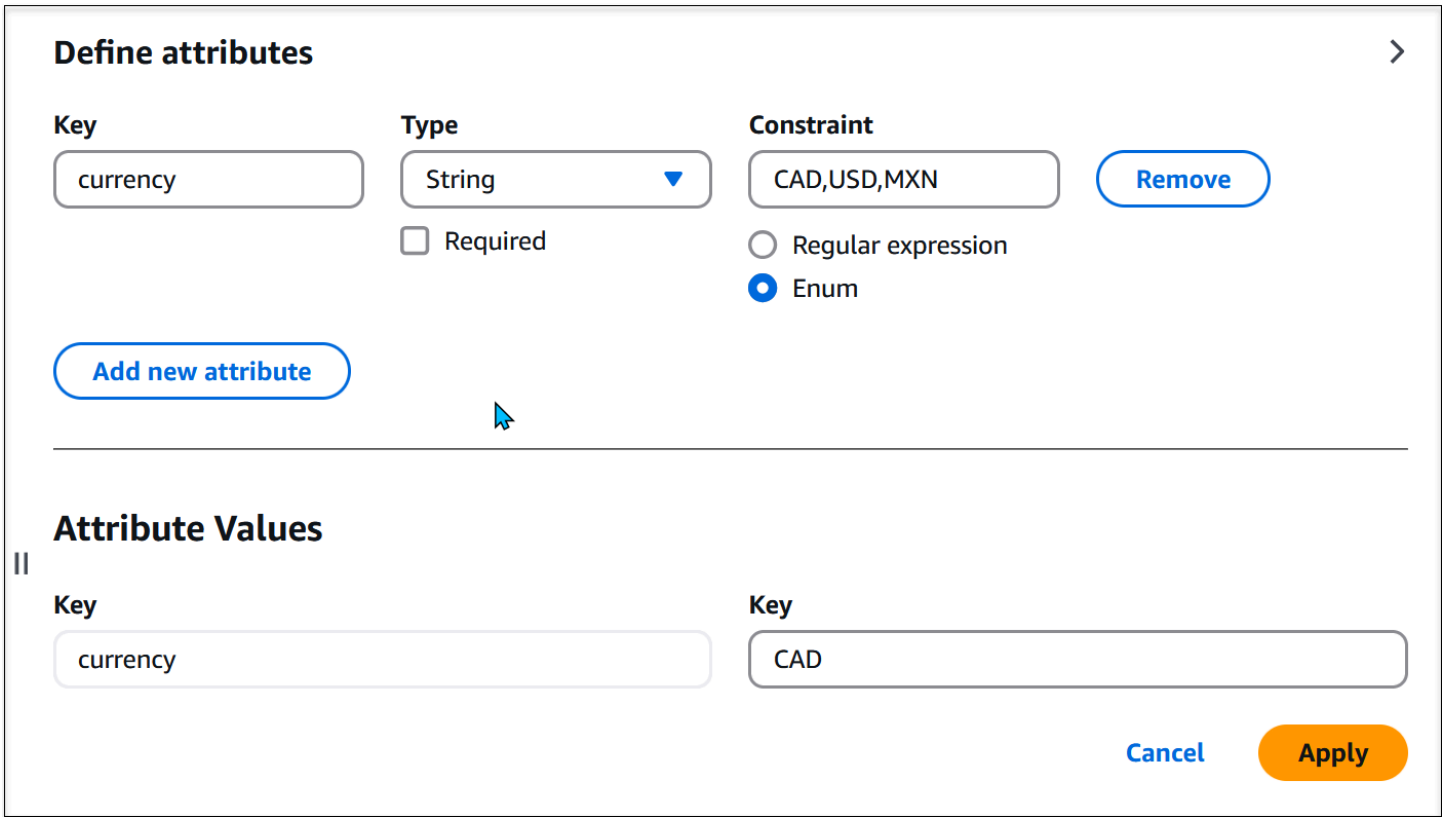
Tópicos

- Noções básicas sobre atributos do sinalizador de atributos
- Criar um perfil de configuração de sinalizador de atributos (console)
- Criar um perfil de configuração de sinalizador de atributos (linha de comandos)
- Criar sinalizadores de atributos multivariante
- Noções básicas sobre a referência de tipo para AWS.AppConfig.FeatureFlags
- Salvar uma versão anterior do sinalizador de atributos em uma nova versão

Noções básicas sobre atributos do sinalizador de atributos

Ao criar um perfil de configuração de sinalizador de atributos ou criar outro sinalizador em um perfil de configuração existente, é possível especificar atributos e restrições correspondentes para o sinalizador. Atributo é um campo que você associa ao sinalizador de atributos para expressar propriedades relacionadas a ele. Os atributos são fornecidos à aplicação com a chave do sinalizador e o valor `enable` ou `disable` do sinalizador.

As restrições impedem que quaisquer valores de atributo inesperados sejam implantados na aplicação. A imagem a seguir mostra um exemplo.



> **Note**
>
> Observe as informações a seguir sobre atributos do sinalizador.
>
> - Para nomes de atributos, a palavra “ativado” é reservada. Você não pode criar nenhum atributo de sinalizadores de atributos chamado “ativado”. Não há outras palavras reservadas.
> - Os atributos de um sinalizador de atributos só são incluídos na resposta de `GetLatestConfiguration` se esse sinalizador estiver ativado.
> - As chaves de atributo de um sinalizador específico devem ser exclusivas.

AWS AppConfig suporta os seguintes tipos de atributos de bandeira e suas restrições correspondentes.

Tipo	Restrição	Description
String	Expressão regular	Padrão Regex para a string
	Enum	Lista de valores aceitáveis para a string
Número	Mínimo	Valor numérico mínimo do atributo
	Máximo	Valor numérico máximo do atributo
Booleano	Nenhum	Nenhum
Matriz de strings	Expressão regular	Padrão Regex para os elementos da matriz
	Enum	Lista de valores aceitáveis para os elementos da matriz
Matriz numérica	Mínimo	Valor numérico mínimo para os elementos da matriz

Tipo	Restrição	Description
	Máximo	Valor numérico máximo para os elementos da matriz

Criar um perfil de configuração de sinalizador de atributos (console)

Use o procedimento a seguir para criar um perfil de configuração do sinalizador de AWS AppConfig recurso usando o AWS AppConfig console. Ao criar o perfil de configuração, você também pode criar um sinalizador de atributos básico.

Para criar um perfil de configuração

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, escolha Aplicações e selecione uma aplicação que você criou em Criando um namespace para seu aplicativo no AWS AppConfig.
3. Na guia Perfis de configuração e sinalizadores de atributos, selecione Criar configuração.
4. Na seção Opções de configuração, escolha sinalizador de atributos.
5. Na seção Perfil de configuração, em ID do perfil de configuração, insira um nome.
6. (Opcional) Expanda Descrição e insira uma descrição.
7. (Opcional) Expanda Opções adicionais e conclua o seguinte, conforme necessário.

    a. Na lista Criptografia, escolha uma chave AWS Key Management Service (AWS KMS) na lista. Essa chave gerenciada pelo cliente permite que você criptografe novas versões de dados de configuração no repositório de configuração AWS AppConfig hospedado. Para ter mais informações sobre essa chave, consulte AWS AppConfig supports customer manager keys em Segurança em AWS AppConfig.

    b. Na seção Tags, selecione Adicionar nova tag e especifique uma chave e um valor opcional.
8. Escolha Próximo.
9. Na seção Definição de sinalizador de atributos, em Nome do sinalizador, insira um nome.
10. Em Chave de sinalizador, insira um identificador de sinalizador para distinguir os sinalizadores no mesmo perfil de configuração. Os sinalizadores no mesmo perfil de configuração não podem ter a mesma chave. Depois que o sinalizador for criado, você poderá editar o nome do sinalizador, mas não a chave do sinalizador.

11. (Opcional) Expanda Descrição e insira informações sobre esse sinalizador.
12. Selecione Este é um sinalizador de curto prazo e, opcionalmente, escolha uma data em que o sinalizador deve ser desativado ou excluído. AWS AppConfig não desativa o sinalizador na data de suspensão de uso.
13. (Opcional) Na seção Atributos do sinalizador de atributos, selecione Definir atributo. Os atributos permitem que você forneça valores adicionais em seu sinalizador. Para ter mais informações sobre atributos e restrições, consulte Noções básicas sobre atributos do sinalizador de atributos.
    a. Em Chave, especifique uma chave de sinalizador e escolha o tipo na lista Tipo. Para ter informações sobre as opções aceitas para os campos Valor e Restrições, consulte a seção mencionada anteriormente sobre atributos.
    b. Selecione Valor obrigatório para especificar se um valor de atributo é obrigatório.
    c. Selecione Definir atributo para adicionar outros atributos.
14. Na seção Valor do sinalizador de atributos, escolha Habilitado para habilitar o sinalizador. Use esse mesmo botão para desabilitar um sinalizador quando ele atingir uma data de depreciação especificada, se aplicável.
15. Escolha Próximo.
16. Na página Revisar e salvar, verifique os detalhes do sinalizador e Salve e continue a implantação.

Vá para Implantando sinalizadores de recursos e dados de configuração no AWS AppConfig.

Criar um perfil de configuração de sinalizador de atributos (linha de comandos)

O procedimento a seguir descreve como usar o AWS Command Line Interface (no Linux ou no Windows) ou o Tools for Windows PowerShell para criar um perfil de configuração do sinalizador de AWS AppConfig recurso. Ao criar o perfil de configuração, você também pode criar um sinalizador de atributos básico.

Como criar uma configuração de sinalizador de atributos

1. Abra AWS CLI o.
2. Crie um perfil de configuração de sinalizadores de atributos especificando seu Tipo como `AWS.AppConfig.FeatureFlags`. O perfil de configuração deve usar `hosted` para o URI de localização.

Linux

```
aws appconfig create-configuration-profile \
  --application-id APPLICATION_ID \
  --name CONFIGURATION_PROFILE_NAME \
  --location-uri hosted \
  --type AWS.AppConfig.FeatureFlags
```

Windows

```
aws appconfig create-configuration-profile ^
  --application-id APPLICATION_ID ^
  --name CONFIGURATION_PROFILE_NAME ^
  --location-uri hosted ^
  --type AWS.AppConfig.FeatureFlags
```

PowerShell

```
New-APPCConfigurationProfile `
  -Name CONFIGURATION_PROFILE_NAME `
  -ApplicationId APPLICATION_ID `
  -LocationUri hosted `
  -Type AWS.AppConfig.FeatureFlags
```

3. Crie seus dados de configuração do sinalizador de atributos. Seus dados devem estar no formato JSON e estar em conformidade com o esquema JSON `AWS.AppConfig.FeatureFlags`. Para obter mais informações sobre o esquema, consulte Noções básicas sobre a referência de tipo para AWS.AppConfig.FeatureFlags.

4. Use a API `CreateHostedConfigurationVersion` para salvar seus dados de configuração do sinalizador de atributos no AWS AppConfig.

Linux

```
aws appconfig create-hosted-configuration-version \
  --application-id APPLICATION_ID \
  --configuration-profile-id CONFIGURATION_PROFILE_ID \
  --content-type "application/json" \
  --content file://path/to/feature_flag_configuration_data.json \
```

```
  --cli-binary-format raw-in-base64-out
```

Windows

```
aws appconfig create-hosted-configuration-version ^
  --application-id APPLICATION_ID ^
  --configuration-profile-id CONFIGURATION_PROFILE_ID ^
  --content-type "application/json" ^
  --content file://path/to/feature_flag_configuration_data.json ^
  --cli-binary-format raw-in-base64-out
```

PowerShell

```
New-APPCHostedConfigurationVersion `
  -ApplicationId APPLICATION_ID `
  -ConfigurationProfileId CONFIGURATION_PROFILE_ID `
  -ContentType "application/json" `
  -Content file://path/to/feature_flag_configuration_data.json
```

O comando carrega o conteúdo especificado para o parâmetro `Content` do disco. O conteúdo deve ser semelhante ao exemplo a seguir.

```
{
    "flags": {
        "ui_refresh": {
            "name": "UI Refresh"
        }
    },
    "values": {
        "ui_refresh": {
            "enabled": false,
            "attributeValues": {
                "dark_mode_support": true
            }
        }
    },
    "version": "1"
}
```

O sistema retorna informações como estas.

Linux

```
{
   "ApplicationId"          : "ui_refresh",
   "ConfigurationProfileId" : "UI Refresh",
   "VersionNumber"          : "1",
   "ContentType"            : "application/json"
}
```

Windows

```
{
   "ApplicationId"          : "ui_refresh",
   "ConfigurationProfileId" : "UI Refresh",
   "VersionNumber"          : "1",
   "ContentType"            : "application/json"
}
```

PowerShell

```
ApplicationId          : ui_refresh
ConfigurationProfileId : UI Refresh
VersionNumber          : 1
ContentType            : application/json
```

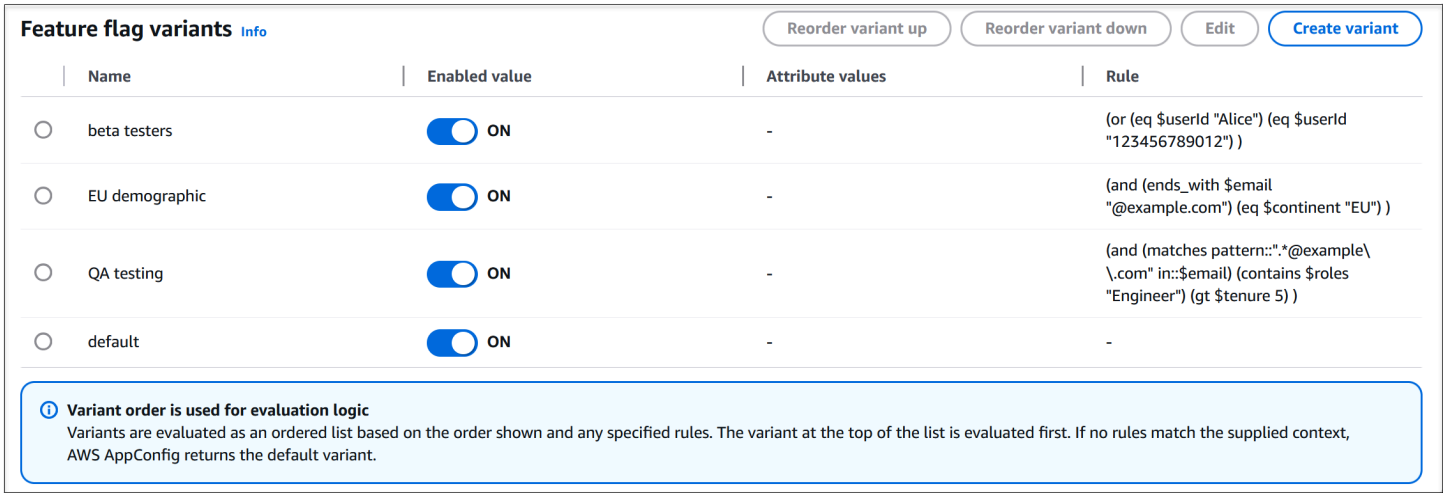
O `service_returned_content_file` contém seus dados de configuração que incluem alguns metadados AWS AppConfig gerados.

> **Note**
>
> Ao criar a versão de configuração hospedada, AWS AppConfig verifica se seus dados estão em conformidade com o esquema `AWS.AppConfig.FeatureFlags` JSON. AWS AppConfig além disso, valida se cada atributo do sinalizador de recurso em seus dados satisfaz as restrições que você definiu para esses atributos.

Criar sinalizadores de atributos multivariante

As variantes de sinalizador de atributos permitem definir um conjunto de possíveis valores de sinalizador a serem exibidos para uma solicitação. Também é possível configurar diferentes status (habilitado ou desabilitado) para sinalizadores multivariante. Ao solicitar um sinalizador configurado com variantes, seu aplicativo fornece um contexto que é AWS AppConfig avaliado em relação a um conjunto de regras definidas pelo usuário. Dependendo do contexto especificado na solicitação e das regras definidas para a variante, AWS AppConfig retorna valores de sinalizadores diferentes para o aplicativo.

A captura de tela a seguir mostra um exemplo de sinalizador de atributos com três variantes definidas pelo usuário e a variante padrão.



Tópicos

- Noções básicas sobre conceitos de sinalizadores de atributos multivariante e casos de uso comuns
- Noções básicas sobre as regras de sinalizador de atributos multivariante
- Criar um sinalizador de atributos multivariante

Noções básicas sobre conceitos de sinalizadores de atributos multivariante e casos de uso comuns

Para ajudar você a entender melhor as variantes de sinalizador de atributos, esta seção explica os conceitos de variante de sinalizador e os casos de uso comuns.

Conceitos

- Sinalizador de recurso: um tipo de AWS AppConfig configuração usado para controlar o comportamento de um recurso em um aplicativo. Um sinalizador tem um status (habilitado ou desabilitado) e um conjunto opcional de atributos que contém valores arbitrários de string, de matriz, numéricos ou boolianos.
- Variante de sinalizador de atributos: uma combinação específica de valores de status e de atributos pertencentes a um sinalizador de atributos. Um sinalizador de atributos pode ter diversas variantes.
- Regra de variante: uma expressão definida pelo usuário usada para selecionar uma variante de sinalizador de atributos. Cada variante tem sua própria regra que AWS AppConfig avalia para determinar se deve devolvê-la ou não.
- Variante padrão: uma variante especial que é exibida quando nenhuma outra é selecionada. Todos os sinalizadores de atributos multivariante têm uma variante padrão.

  A variante padrão deve ser a última na sua ordem de variantes e não pode ter regras associadas a ela. Se não estiver definido por último, AWS AppConfig retornará a `BadRequestException` quando você tentar criar o sinalizador de várias variantes.
- Contexto: chaves e valores definidos pelo usuário transmitidos ao AWS AppConfig no momento da recuperação da configuração. Os valores de contexto são usados durante a avaliação da regra para selecionar a variante de sinalizador de atributos a ser exibida.

> **Note**
>
> AWS AppConfig o agente avalia as regras variantes e determina qual regra se aplica à solicitação com base no contexto fornecido. Para saber mais sobre como recuperar os sinalizadores de atributos multivariante, consulte Recuperar sinalizadores de atributos básicos e multivariante.

Casos de uso comuns

Esta seção descreve dois casos de uso comuns para variantes de sinalizador de atributos.

Segmentação de usuários

A segmentação de usuários é o processo de dividir usuários com base em determinados atributos. Por exemplo, é possível usar variantes de sinalizador para expor um recurso a alguns usuários, mas

não a outros, com base em dados, como ID do usuário, localização geográfica, tipo de dispositivo ou frequência de compra.

Usando o exemplo da frequência de compra, suponha que sua aplicação de comércio eletrônico comporte um recurso para aumentar a fidelidade do cliente. É possível usar variantes de sinalizador para configurar diferentes tipos de incentivo a serem exibidos a um usuário com base na última vez em que ele comprou algo. Um novo usuário pode receber um pequeno desconto para incentivá-lo a se tornar um cliente, enquanto um cliente recorrente poderá receber um desconto maior se comprar algo de uma nova categoria.

Divisão de tráfego

Divisão de tráfego é o processo de selecionar uma variante de sinalizador aleatória, mas consistente, com base em um valor de contexto definido por você. Por exemplo, talvez você queira realizar um experimento em que uma pequena porcentagem de seus usuários (identificados pelo ID de usuário) veja uma variante específica. Ou talvez você queira realizar uma distribuição gradual de recursos, em que um deles seja exposto primeiro a 5% dos usuários, depois a 15%, depois a 40% e, por fim, a 100%, mantendo uma experiência de usuário consistente durante toda a distribuição.

Usando o exemplo de experimentação, é possível usar variantes de sinalizador para testar um novo estilo de botão para a ação principal na página inicial da aplicação e ver se ele gera mais cliques. Em seu experimento, você pode criar uma variante de sinalizador com uma regra de divisão de tráfego que selecione 5% dos usuários para ver o novo estilo, enquanto a variante padrão indica os usuários que devem continuar vendo o estilo existente. Se o experimento for bem-sucedido, você poderá aumentar o valor percentual ou até mesmo determinar que essa variante seja o padrão.

Noções básicas sobre as regras de sinalizador de atributos multivariante

Ao criar uma variante de sinalizador de atributos, especifique uma regra para ela. Regras são expressões que usam valores de contexto como entrada e produzem um resultado booliano como saída. Por exemplo, é possível definir uma regra para selecionar uma variante de sinalizador para usuários beta, identificada pelo ID da conta, testando uma atualização da interface de usuário. No caso desse cenário, faça o seguinte:

1. Crie um perfil de configuração de sinalizador de atributos chamado Atualização da interface de usuário.
2. Crie um sinalizador de atributos chamado ui_refresh.
3. Depois de criá-lo, edite-o para adicionar variantes.

4. Crie e habilite uma nova variante chamada BetaUsers.
5. Defina uma regra para selecionar BetaUsersa variante se o ID da conta do contexto da solicitação estiver em uma lista de contas IDs aprovadas para visualizar a nova experiência beta.
6. Confirme se o status da variante padrão está definido como Desativado.

> **Note**
>
> As variantes são avaliadas como uma lista ordenada com base na ordem em que são definidas no console. A variante no início da lista será avaliada primeiro. Se nenhuma regra corresponder ao contexto fornecido, AWS AppConfig retornará a variante Default.

Quando AWS AppConfig processa a solicitação do sinalizador de recurso, ele compara primeiro o contexto fornecido, que inclui o accountId (neste exemplo) com a variante. BetaUsers Se o contexto corresponder à regra de BetaUsers, AWS AppConfig retornará os dados de configuração da experiência beta. Se o contexto não incluir uma ID da conta ou se a ID da conta terminar em algo diferente de 123, AWS AppConfig retornará os dados de configuração da regra padrão, o que significa que o usuário visualiza a experiência atual na produção.

> **Note**
>
> Para obter mais informações sobre como recuperar os sinalizadores de atributos multivariante, consulte Recuperar sinalizadores de atributos básicos e multivariante.

Definir regras para sinalizadores de atributos multivariante

Regra de variante é uma expressão composta de um ou mais operandos e de um operador. Operando é um valor específico usado durante a avaliação de uma regra. Os valores de operando podem ser estáticos, como um número literal ou uma string, ou variáveis, como o valor encontrado em um contexto ou o resultado de outra expressão. Um operador, como “maior que”, é um teste ou uma ação aplicada aos respectivos operandos que produz um valor. Uma expressão de regra de variante deve produzir um valor “true” ou “false” para ser válida.

Operandos

Tipo	Description	Exemplo
String	Uma sequência de caracteres UTF-8, entre aspas duplas.	`"apple", "####ë# ##š##"`
Inteiro	Um valor inteiro de 64 bits.	`-7, 42`
Float	Um valor de ponto flutuante IEEE 754 de 64 bits.	`3.14, 1.234e-5`
Timestamp	Um momento específico, conforme descrito na nota do W3C sobre formatos de data e hora.	`2012-03-04T05:06:07-08:00, 2024-01`
Booleano	Um valor verdadeiro ou falso.	`true, false`
Valor de contexto.	Um valor parametrizado na forma de $ *key* que é recuperado do contexto durante a avaliação da regra.	`$country, $userId`

Operadores de comparação

Operador	Description	Exemplo
eq	Determina se um valor de contexto é igual a um valor específico.	`(eq $state "Virginia")`
gt	Determina se um valor de contexto é maior do que um valor específico.	`(gt $age 65)`

Operador	Description	Exemplo
gte	Determina se um valor de contexto é maior ou igual a um valor específico.	`(gte $age 65)`
lt	Determina se um valor de contexto é menor do que um valor específico.	`(lt $age 65)`
lte	Determina se um valor de contexto é menor ou igual a um valor específico.	`(lte $age 65)`

Operadores lógicos

Operador	Description	Exemplo

and	Determina se os dois operandos são verdadeiros.	`(and`
		`  (eq $state "Virginia")`
		`  (gt $age 65)`
		`)`

or	Determina se pelo menos um dos operandos é verdadeiro.	`(or`
		`  (eq $state "Virginia")`
		`  (gt $age 65)`
		`)`

not	Reverte o valor de uma expressão.	`(not (eq $state "Virginia"))`

Operadores personalizados

Operador	Description	Exemplo
begins_with	Determina se um valor de contexto começa com um prefixo específico.	`(begins_with $state "A")`
ends_with	Determina se um valor de contexto termina com um prefixo específico.	`(ends_with $email "amazon.com")`
contém	Determina se um valor de contexto contém uma substring específica.	`(contains $promoCode "WIN")`
in	Determina se um valor de contexto está contido em uma lista de constantes.	`(in $userId ["123", "456"])`
matches	Determina se um valor de contexto corresponde a um padrão regex específico.	`(matches in::$greeting pattern::"h.*y")`
exists	Determina se algum valor foi fornecido para uma chave de contexto.	`(exists key::"country")`
dividir	Avalia uma porcentagem específica de tráfego como `true` com base em um hash consistente dos valores de contexto fornecidos. Para obter uma explicaçã o detalhada sobre como o `split` funciona, confira a próxima seção deste tópico, Entender o operador split.	`(split pct::10 by::$user Id seed::"abc")`

Operador	Description	Exemplo
	Observe que `seed` é uma propriedade opcional. Se você não especificar `seed`, o hash será localmente consistente, o que significa que o tráfego será dividido de forma consistente para esse sinalizador, mas outros sinalizadores que receberem o mesmo valor de contexto poderão dividir o tráfego de maneira diferente. Se `seed` for fornecido, cada valor exclusivo dividirá o tráfego de forma consistente entre sinalizad ores de atributos, perfis de configuração e Contas da AWS.	

Entender o operador split

A seção a seguir descreve como o operador `split` se comporta quando usado em diferentes cenários. Como lembrete, `split` é avaliado como `true` para uma determinada porcentagem de tráfego com base em um hash consistente do valor de contexto fornecido. Para entender melhor, considere o seguinte cenário de linha de base que usa a divisão com duas variantes:

```
A: (split by::$uniqueId pct::20)
C: <no rule>
```

Como esperado, fornecer um conjunto aleatório de valores `uniqueId` produz uma distribuição que é aproximadamente:

```
A: 20%
C: 80%
```

Se você adicionar uma terceira variante, mas usar a mesma porcentagem de divisão da seguinte maneira:

```
A: (split by::$uniqueId pct::20)
B: (split by::$uniqueId pct::20)
C: <default>
```

A distribuição fica da seguinte forma:

```
A: 20%
B: 0%
C: 80%
```

Essa distribuição possivelmente inesperada acontece porque cada regra de variante é avaliada em ordem, e a primeira correspondência define a variante retornada. Quando a regra A é avaliada, 20% dos valores `uniqueId` correspondem a ela, então a primeira variante é retornada. Em seguida, a regra B é avaliada. No entanto, todos os valores `uniqueId` que corresponderiam à segunda instrução dividida já foram correspondidos pela regra de variante A. Portanto, nenhum valor corresponde a B. Em vez disso, a variante padrão é retornada.

Agora, considere um terceiro exemplo.

```
A: (split by::$uniqueId pct::20)
B: (split by::$uniqueId pct::25)
C: <default>
```

Como no exemplo anterior, os primeiros 20% dos valores `uniqueId` correspondem à regra A. Para a regra da variante B, 25% de todos os valores `uniqueId` seriam correspondentes. Porém, a maioria deles correspondia à regra A. Isso deixa 5% do total para a variante B, com o restante recebendo a variante C. A distribuição teria a seguinte aparência:

```
A: 20%
B: 5%
C: 75%
```

Usar a propriedade **`seed`**

Você pode usar a propriedade `seed` para garantir que o tráfego seja dividido de maneira consistente para um determinado valor de contexto, independentemente de onde o operador split seja usado. Se você não especificar `seed`, o hash será localmente consistente, o que significa que o tráfego será dividido de forma consistente para esse sinalizador, mas outros sinalizadores que receberem o mesmo valor de contexto poderão dividir o tráfego de maneira diferente. Se `seed` for fornecido, cada valor exclusivo dividirá o tráfego de forma consistente entre sinalizadores de atributos, perfis de configuração e Contas da AWS.

Os clientes costumam usar o mesmo valor `seed` em todas as variantes em um sinalizador ao dividir o tráfego na mesma propriedade de contexto. No entanto, às vezes pode fazer sentido usar um valor de semente diferente. Confira um exemplo que usa sementes diferentes para as regras A e B:

```
A: (split by::$uniqueId pct::20 seed::"seed_one")
B: (split by::$uniqueId pct::25 seed::"seed_two")
C: <default>
```

Como antes, 20% dos valores `uniqueId` equivalentes correspondem à regra A. Isso significa que 80% dos valores são aprovados e testados em relação à regra da variante B. Como a semente é diferente, não há correlação entre os valores que correspondem a A e os que correspondem a B. No entanto, existem somente 80% dos valores `uniqueId` a serem divididos com 25% desse número correspondente à regra B, e 75% não. Isso funciona para a seguinte distribuição:

```
A: 20%
B: 20% (25% of what falls through from A, or 25% of 80%)
C: 60%
```

Criar um sinalizador de atributos multivariante

Use os procedimentos desta seção para criar variantes de um sinalizador de atributos.

Antes de começar

Observe as seguintes informações importantes:

- É possível criar variantes de sinalizadores de atributos existentes editando-os. Não é possível criar variantes de um novo sinalizador de atributos ao criar um perfil de configuração. Primeiro, é necessário concluir o fluxo de trabalho de criação do perfil de configuração. Após a criação do perfil de configuração, é possível adicionar variantes a qualquer sinalizador no perfil de configuração. Para ter informações sobre como criar um perfil de configuração, consulte Criando um perfil de configuração de sinalizador de recurso no AWS AppConfig.

- Para recuperar dados de variantes de sinalização de recurso para as plataformas computacionais Amazon EC2, Amazon ECS e Amazon EKS, você deve AWS AppConfig usar a versão 2.0.4416 ou posterior do Agent.
- Por motivos de desempenho AWS CLI e chamadas do SDK para AWS AppConfig não recuperar dados de variantes. Para obter mais informações sobre o AWS AppConfig Agente, consulteComo usar o AWS AppConfig Agent para recuperar dados de configuração.
- Ao criar uma variante de sinalizador de atributos, especifique uma regra para ela. Regras são expressões que usam o contexto da solicitação como entrada e produzem um resultado booliano como saída. Antes de criar variantes, revise os operandos e os operadores compatíveis para regras de variantes de sinalizador. É possível criar regras antes de criar variantes. Para obter mais informações, consulte Noções básicas sobre as regras de sinalizador de atributos multivariante.

Tópicos

- Criar um sinalizador de atributos multivariante (console)
- Criar um sinalizador de atributos multivariante (linha de comandos)

Criar um sinalizador de atributos multivariante (console)

O procedimento a seguir descreve como criar um sinalizador de recurso de várias variantes para um perfil de configuração existente usando o AWS AppConfig console. Também é possível editar sinalizadores de atributos existentes para criar variantes.

Como criar um sinalizador de atributos multivariante

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, escolha Aplicações e selecione uma aplicação.
3. Na guia Perfis de configuração e sinalizadores de atributoss, escolha um perfil de configuração de sinalizador de atributos existente.
4. Na seção Sinalizadores, escolha Adicionar novo sinalizador.
5. Na seção Definição de sinalizador de atributos, em Nome do sinalizador, insira um nome.
6. Em Chave de sinalizador, insira um identificador de sinalizador para distinguir os sinalizadores no mesmo perfil de configuração. Os sinalizadores no mesmo perfil de configuração não podem ter a mesma chave. Depois que o sinalizador for criado, você poderá editar o nome do sinalizador, mas não a chave do sinalizador.

7. (Opcional) No campo Descrição, insira informações sobre esse sinalizador.
8. Na seção Variantes, selecione Sinalizador multivariante.
9. (Opcional) Na seção Atributos do sinalizador de atributos, selecione Definir atributo. Os atributos permitem que você forneça valores adicionais em seu sinalizador. Para ter mais informações sobre atributos e restrições, consulte Noções básicas sobre atributos do sinalizador de atributos.

    a. Em Chave, especifique uma chave de sinalizador e escolha o tipo na lista Tipo. Para ter informações sobre as opções aceitas para os campos Valor e Restrições, consulte a seção mencionada anteriormente sobre atributos.

    b. Selecione Valor obrigatório para especificar se um valor de atributo é obrigatório.

    c. Selecione Definir atributo para adicionar outros atributos.

    d. Selecione Aplicar para salvar as alterações de atributo.

10. Na seção Variantes de sinalizador de atributos, selecione Criar variante.

    a. Em Nome da variante, insira um nome.

    b. Use o botão de alternância Valor habilitado para habilitar a variante.

    c. Na caixa de texto Regra, insira uma regra.

    d. Se quiser criar variantes adicionais para esse sinalizador, use as opções Criar variante > Criar a variante acima ou Criar uma variante abaixo.

    e. Na seção Variante padrão, use a opção Valor habilitado para habilitar a variante padrão. Também é possível fornecer valores para os atributos definidos na Etapa 10.

    f. Escolha Aplicar.

11. Verifique os detalhes do sinalizador e as respectivas variantes e selecione Criar sinalizador.

Para ter informações sobre como implantar um novo sinalizador de atributos com variantes, consulte Implantando sinalizadores de recursos e dados de configuração no AWS AppConfig.

Criar um sinalizador de atributos multivariante (linha de comandos)

O procedimento a seguir descreve como usar o AWS Command Line Interface (no Linux ou no Windows) ou o Tools for Windows PowerShell para criar um sinalizador de recurso de várias variantes para um perfil de configuração existente. Também é possível editar sinalizadores de atributos existentes para criar variantes.

Antes de começar

Realize as tarefas a seguir antes de criar um sinalizador de atributos multivariante usando a AWS CLI.

- Crie um perfil de configuração de sinalizador de atributos. Para obter mais informações, consulte Criando um perfil de configuração de sinalizador de recurso no AWS AppConfig.
- Atualize para a versão mais recente da AWS CLI. Para ter mais informações, consulte Install or update to the latest version of the AWS CLI no Guia do usuário da versão 2 da AWS Command Line Interface .

Como criar um sinalizador de atributos multivariante

1. Crie um arquivo de configuração em sua máquina local que especifique os detalhes do sinalizador multivariante que você deseja criar. Salve o arquivo com a extensão `.json`. O arquivo deve seguir o esquema JSON `AWS.AppConfig.FeatureFlags`. O conteúdo do esquema do arquivo de configuração será semelhante ao seguinte:

```
{
  "flags": {
    "FLAG_NAME": {
      "attributes": {
          "ATTRIBUTE_NAME": {
          "constraints": {
            "type": "CONSTRAINT_TYPE"
          }
        }
      },
      "description": "FLAG_DESCRIPTION",
      "name": "VARIANT_NAME"
    }
  },
  "values": {
    "VARIANT_VALUE_NAME": {
      "_variants": [
        {
          "attributeValues": {
            "ATTRIBUTE_NAME": BOOLEAN
          },
          "enabled": BOOLEAN,
          "name": "VARIANT_NAME",
          "rule": "VARIANT_RULE"
        },
```

```
        {
          "attributeValues": {
            "ATTRIBUTE_NAME": BOOLEAN
          },
          "enabled": BOOLEAN,
          "name": "VARIANT_NAME",
          "rule": "VARIANT_RULE"
        },
        {
          "attributeValues": {
            "ATTRIBUTE_NAME": BOOLEAN
          },
          "enabled": BOOLEAN,
          "name": "VARIANT_NAME",
          "rule": "VARIANT_RULE"
        },
        {
          "attributeValues": {
            "ATTRIBUTE_NAME": BOOLEAN
          },
          "enabled": BOOLEAN,
          "name": "VARIANT_NAME",
          "rule": "VARIANT_RULE"
        }
      ]
    }
  },
  "version": "VERSION_NUMBER"
}
```

Veja um exemplo com três variantes e a variante padrão.

```
{
  "flags": {
    "ui_refresh": {
      "attributes": {
        "dark_mode_support": {
          "constraints": {
            "type": "boolean"
          }
        }
      },
      "description": "A release flag used to release a new UI",
```

```
      "name": "UI Refresh"
    }
  },
  "values": {
    "ui_refresh": {
      "_variants": [
        {
          "attributeValues": {
            "dark_mode_support": true
          },
          "enabled": true,
          "name": "QA",
          "rule": "(ends_with $email \"qa-testers.mycompany.com\")"
        },
        {
          "attributeValues": {
            "dark_mode_support": true
          },
          "enabled": true,
          "name": "Beta Testers",
          "rule": "(exists key::\"opted_in_to_beta\")"
        },
        {
          "attributeValues": {
            "dark_mode_support": false
          },
          "enabled": true,
          "name": "Sample Population",
          "rule": "(split pct::10 by::$email)"
        },
        {
          "attributeValues": {
            "dark_mode_support": false
          },
          "enabled": false,
          "name": "Default Variant"
        }
      ]
    }
  },
  "version": "1"
}
```

2. Use a API `CreateHostedConfigurationVersion` para salvar seus dados de configuração do sinalizador de atributos no AWS AppConfig.

   Linux

   ```
   aws appconfig create-hosted-configuration-version \
     --application-id APPLICATION_ID \
     --configuration-profile-id CONFIGURATION_PROFILE_ID \
     --content-type "application/json" \
     --content file://path/to/feature_flag_configuration_data.json \
     --cli-binary-format raw-in-base64-out \
     outfile
   ```

   Windows

   ```
   aws appconfig create-hosted-configuration-version ^
     --application-id APPLICATION_ID ^
     --configuration-profile-id CONFIGURATION_PROFILE_ID ^
     --content-type "application/json" ^
     --content file://path/to/feature_flag_configuration_data.json ^
     --cli-binary-format raw-in-base64-out ^
     outfile
   ```

   PowerShell

   ```
   New-APPCHostedConfigurationVersion `
     -ApplicationId APPLICATION_ID `
     -ConfigurationProfileId CONFIGURATION_PROFILE_ID `
     -ContentType "application/json" `
     -Content file://path/to/feature_flag_configuration_data.json `
     -Raw
   ```

   O `service_returned_content_file` contém seus dados de configuração que incluem alguns metadados AWS AppConfig gerados.

   > **Note**
   >
   > Ao criar a versão de configuração hospedada, AWS AppConfig verifica se seus dados estão em conformidade com o esquema `AWS.AppConfig.FeatureFlags`JSON. AWS

AppConfig além disso, valida se cada atributo do sinalizador de recurso em seus dados satisfaz as restrições que você definiu para esses atributos.

Noções básicas sobre a referência de tipo para AWS.AppConfig.FeatureFlags

Use o esquema JSON `AWS.AppConfig.FeatureFlags` como referência para criar seus dados de configuração do sinalizador de atributos.

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "definitions": {
    "flagSetDefinition": {
      "type": "object",
      "properties": {
        "version": {
          "$ref": "#/definitions/flagSchemaVersions"
        },
        "flags": {
          "$ref": "#/definitions/flagDefinitions"
        },
        "values": {
          "$ref": "#/definitions/flagValues"
        }
      },
      "required": ["version"],
      "additionalProperties": false
    },
    "flagDefinitions": {
      "type": "object",
      "patternProperties": {
        "^[a-z][a-zA-Z\\d_-]{0,63}$": {
          "$ref": "#/definitions/flagDefinition"
        }
      },
      "additionalProperties": false
    },
    "flagDefinition": {
      "type": "object",
      "properties": {
        "name": {
          "$ref": "#/definitions/customerDefinedName"
```

```
        },
        "description": {
          "$ref": "#/definitions/customerDefinedDescription"
        },
        "_createdAt": {
          "type": "string"
        },
        "_updatedAt": {
          "type": "string"
        },
        "_deprecation": {
          "type": "object",
          "properties": {
            "status": {
              "type": "string",
              "enum": ["planned"]
            },
            "date": {
              "type": "string",
              "format": "date"
            }
          },
         "additionalProperties": false
        },
        "attributes": {
          "$ref": "#/definitions/attributeDefinitions"
        }
      },
      "additionalProperties": false
    },
    "attributeDefinitions": {
      "type": "object",
      "patternProperties": {
        "^[a-z][a-zA-Z\\d_-]{0,63}$": {
          "$ref": "#/definitions/attributeDefinition"
        }
      },
      "maxProperties": 25,
      "additionalProperties": false
    },
    "attributeDefinition": {
      "type": "object",
      "properties": {
        "description": {
```

```
          "$ref": "#/definitions/customerDefinedDescription"
        },
        "constraints": {
          "oneOf": [
            { "$ref": "#/definitions/numberConstraints" },
            { "$ref": "#/definitions/stringConstraints" },
            { "$ref": "#/definitions/arrayConstraints" },
            { "$ref": "#/definitions/boolConstraints" }
          ]
        }
      },
      "additionalProperties": false
    },
    "flagValues": {
      "type": "object",
      "patternProperties": {
        "^[a-z][a-zA-Z\\d_-]{0,63}$": {
          "$ref": "#/definitions/flagValue"
        }
      },
      "additionalProperties": false
    },
    "flagValue": {
      "type": "object",
      "properties": {
        "enabled": {
          "type": "boolean"
        },
        "_createdAt": {
          "type": "string"
        },
        "_updatedAt": {
          "type": "string"
        },
        "_variants": {
          "type": "array",
          "maxLength": 32,
          "items": {
            "$ref": "#/definitions/variant"
          }
        }
      },
      "patternProperties": {
        "^[a-z][a-zA-Z\\d_-]{0,63}$": {
```

```
          "$ref": "#/definitions/attributeValue",
          "maxProperties": 25
        }
      },
      "additionalProperties": false
    },
    "attributeValue": {
      "oneOf": [
        { "type": "string", "maxLength": 1024 },
        { "type": "number" },
        { "type": "boolean" },
        {
          "type": "array",
          "oneOf": [
            {
              "items": {
                "type": "string",
                "maxLength": 1024
              }
            },
            {
              "items": {
                "type": "number"
              }
            }
          ]
        }
      ],
      "additionalProperties": false
    },
    "stringConstraints": {
      "type": "object",
      "properties": {
        "type": {
          "type": "string",
          "enum": ["string"]
        },
        "required": {
          "type": "boolean"
        },
        "pattern": {
          "type": "string",
          "maxLength": 1024
        },
```

```
        "enum": {
          "type": "array",
          "maxLength": 100,
          "items": {
            "oneOf": [
              {
                "type": "string",
                "maxLength": 1024
              },
              {
                "type": "integer"
              }
            ]
          }
        }
      },
      "required": ["type"],
      "not": {
        "required": ["pattern", "enum"]
      },
      "additionalProperties": false
    },
    "numberConstraints": {
      "type": "object",
      "properties": {
        "type": {
          "type": "string",
          "enum": ["number"]
        },
        "required": {
          "type": "boolean"
        },
        "minimum": {
          "type": "integer"
        },
        "maximum": {
          "type": "integer"
        }
      },
      "required": ["type"],
      "additionalProperties": false
    },
    "arrayConstraints": {
      "type": "object",
```

```
      "properties": {
        "type": {
          "type": "string",
          "enum": ["array"]
        },
        "required": {
          "type": "boolean"
        },
        "elements": {
          "$ref": "#/definitions/elementConstraints"
        }
      },
      "required": ["type"],
      "additionalProperties": false
    },
    "boolConstraints": {
      "type": "object",
      "properties": {
        "type": {
          "type": "string",
          "enum": ["boolean"]
        },
        "required": {
          "type": "boolean"
        }
      },
      "required": ["type"],
      "additionalProperties": false
    },
    "elementConstraints": {
      "oneOf": [
        { "$ref": "#/definitions/numberConstraints" },
        { "$ref": "#/definitions/stringConstraints" }
      ]
    },
    "variant": {
      "type": "object",
      "properties": {
        "enabled": {
          "type": "boolean"
        },
        "name": {
          "$ref": "#/definitions/customerDefinedName"
        },
```

```
        "rule": {
          "type": "string",
          "maxLength": 16384
        },
        "attributeValues": {
          "type": "object",
          "patternProperties": {
            "^[a-z][a-zA-Z\\d_-]{0,63}$": {
              "$ref": "#/definitions/attributeValue"
            }
          },
          "maxProperties": 25,
          "additionalProperties": false
        }
      },
      "required": ["name", "enabled"],
      "additionalProperties": false
    },
    "customerDefinedName": {
      "type": "string",
      "pattern": "^[^\\n]{1,64}$"
    },
    "customerDefinedDescription": {
      "type": "string",
      "maxLength": 1024
    },
    "flagSchemaVersions": {
      "type": "string",
      "enum": ["1"]
    }
  },
  "type": "object",
  "$ref": "#/definitions/flagSetDefinition",
  "additionalProperties": false
}
```

Important

Para recuperar dados de configuração do sinalizador de atributos, seu aplicativo deve chamar a API `GetLatestConfiguration`. Não é possível recuperar dados de configuração do sinalizador de atributos chamando `GetConfiguration`, que está obsoleto.

Para obter mais informações, consulte GetLatestConfiguration na Referência de APIs do AWS AppConfig .

Quando seu aplicativo chama GetLatestConfiguratione recebe uma configuração recém-implantada, as informações que definem seus sinalizadores e atributos de recursos são removidas. O JSON simplificado contém um mapa de chaves que correspondem a cada uma das chaves de sinalizadores que você especificou. O JSON simplificado também contém valores `true` ou `false` mapeados para o atributo `enabled`. Se um sinalizador definir `enabled` como `true`, todos os atributos do sinalizador também estarão presentes. O esquema JSON a seguir descreve o formato da saída JSON.

```
{
  "$schema": "http://json-schema.org/draft-07/schema#",
  "type": "object",
  "patternProperties": {
    "^[a-z][a-zA-Z\\d_-]{0,63}$": {
      "$ref": "#/definitions/attributeValuesMap"
    }
  },
  "additionalProperties": false,
  "definitions": {
    "attributeValuesMap": {
      "type": "object",
      "properties": {
        "enabled": {
          "type": "boolean"
        }
      },
      "required": ["enabled"],
      "patternProperties": {
        "^[a-z][a-zA-Z\\d_-]{0,63}$": {
          "$ref": "#/definitions/attributeValue"
        }
      },
      "maxProperties": 25,
      "additionalProperties": false
    },
    "attributeValue": {
      "oneOf": [
        { "type": "string","maxLength": 1024 },
        { "type": "number" },
```

```
        { "type": "boolean" },
        {
          "type": "array",
          "oneOf": [
            {
              "items": {
                "oneOf": [
                  {
                    "type": "string",
                    "maxLength": 1024
                  }
                ]
              }
            },
            {
              "items": {
                "oneOf": [
                  {
                    "type": "number"
                  }
                ]
              }
            }
          ]
        }
      ],
      "additionalProperties": false
    }
  }
}
```

Salvar uma versão anterior do sinalizador de atributos em uma nova versão

Quando você atualiza um sinalizador de recurso, salva AWS AppConfig automaticamente suas alterações em uma nova versão. Se quiser usar uma versão anterior do sinalizador de atributos, copie-a para uma versão de rascunho e depois salve-a. Não é possível editar e salvar as alterações em uma versão anterior do sinalizador sem salvá-las em uma nova versão.

Para editar uma versão anterior do sinalizador de atributos e salvá-la em uma nova versão

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.

2. No painel de navegação, escolha aplicativos e, depois, o aplicativo com o sinalizador de atributos que deseja editar e salvar em uma nova versão.
3. Na guia Perfis de configuração e sinalizadores de atributos, escolha o perfil de configuração com o sinalizador de atributos que você quer editar e salvar em uma nova versão.
4. Na guia Sinalizadores de atributos, use a lista de versões para escolher a versão que você quer editar e salvar em uma nova versão.
5. Escolha Copiar para a versão de rascunho.
6. No campo Rótulo da versão, insira um novo rótulo (opcional, mas recomendado).
7. No campo Descrição da versão, insira uma nova descrição (opcional, mas recomendado).
8. Escolha Salvar versão.
9. Escolha Iniciar implantação para implantar a nova versão.

Criando um perfil de configuração de formato livre no AWS AppConfig

Dados de configuração são um conjunto de configurações que influenciam o comportamento da aplicação. Um perfil de configuração inclui, entre outras coisas, um URI que permite AWS AppConfig localizar seus dados de configuração em seu local armazenado e um tipo de configuração. Com perfis de configuração de formato livre, você pode armazenar seus dados no repositório de configuração AWS AppConfig hospedado ou em qualquer uma das seguintes ferramentas Serviços da AWS e no Systems Manager:

Local	Tipos de arquivos compatíveis
AWS AppConfig armazenamento de configuração hospedado	YAML, JSON e texto, se adicionados usando o. Console de gerenciamento da AWS Qualquer tipo de arquivo, se adicionado usando a ação AWS AppConfig CreateHostedConfigurationVersionda API.
Amazon Simple Storage Service (Amazon S3)	Any
AWS CodePipeline	Pipeline (conforme definido pelo serviço)
AWS Secrets Manager	Segredo (conforme definido pelo serviço)

Local	Tipos de arquivos compatíveis
AWS Systems Manager Parameter Store	Parâmetros de string padrão e seguros (conforme definido pelo Parameter Store)
AWS Systems Manager armazenamento de documentos (documentos SSM)	YAML, JSON, texto

Um perfil de configuração também pode incluir validadores opcionais para garantir que seus dados de configuração estejam sintática e semanticamente corretos. AWS AppConfig executa uma verificação usando os validadores quando você inicia uma implantação. Se algum erro for detectado, a implantação será interrompida antes de fazer qualquer alteração nos destinos da configuração.

> **Note**
>
> Se possível, recomendamos hospedar seus dados de configuração no armazenamento de configuração AWS AppConfig hospedado, pois ele oferece mais recursos e aprimoramentos.

Para configurações de formato livre armazenadas no repositório de configurações AWS AppConfig hospedado ou nos documentos SSM, você pode criar a configuração de forma livre usando o console do Systems Manager ao criar um perfil de configuração. O processo está descrito posteriormente neste tópico.

Para configurações de formato livre armazenadas no Parameter Store, Secrets Manager ou Amazon S3, você deve primeiro criar o parâmetro, segredo ou objeto e armazená-lo no respectivo armazenamento de configuração. Depois de armazenar os dados de configuração, use o procedimento neste tópico para criar o perfil de configuração.

Tópicos

- Noções básicas sobre validadores
- Noções básicas sobre cotas e limitações do armazenamento de configuração
- Entendendo o armazenamento de configuração AWS AppConfig hospedado
- Noções básicas sobre configurações armazenadas no Amazon S3
- Criação de um perfil AWS AppConfig de configuração de formato livre (console)
- Criação de um perfil AWS AppConfig de configuração de formato livre (linha de comando)

Noções básicas sobre validadores

Ao criar um perfil de configuração, é possível especificar até dois validadores. Um validador garante que os dados da configuração estejam sintaticamente e semanticamente corretos. Se você planeja usar um validador, deve criá-lo antes de criar o perfil de configuração. AWS AppConfig suporta os seguintes tipos de validadores:

- AWS Lambda funções: Compatível com sinalizadores de recursos e configurações de formato livre.
- Esquema JSON: compatível com configurações de formato livre. (valida AWS AppConfig automaticamente os sinalizadores de recursos em relação a um esquema JSON.)

Tópicos

- AWS Lambda validadores de funções
- Validadores do esquema JSON

AWS Lambda validadores de funções

O validadores da função do Lambda devem ser configurados com o esquema de eventos a seguir. O AWS AppConfig usa esse esquema para invocar a função do Lambda. O conteúdo é uma string codificada em base64 e o URI é uma string.

```
{
    "applicationId": "The application ID of the configuration profile being
 validated",
    "configurationProfileId": "The ID of the configuration profile being validated",
    "configurationVersion": "The version of the configuration profile being validated",
    "content": "Base64EncodedByteString",
    "uri": "The configuration uri"
}
```

AWS AppConfig verifica se o cabeçalho `X-Amz-Function-Error` Lambda está definido na resposta. O Lambda define esse cabeçalho se a função gerar uma exceção. Para obter mais informações sobre `X-Amz-Function-Error`, consulte Lidar com erros e tentativas automáticas no AWS Lambda no Guia do desenvolvedor do AWS Lambda .

Veja um exemplo simples de um código de resposta do Lambda para uma validação bem-sucedida.

```
import json

def handler(event, context):
     #Add your validation logic here
     print("We passed!")
```

Veja um exemplo simples de um código de resposta do Lambda para uma validação malsucedida.

```
def handler(event, context):
     #Add your validation logic here
     raise Exception("Failure!")
```

Veja a seguir outro exemplo que valida somente se o parâmetro de configuração for um número primo.

```
function isPrime(value) {
    if (value < 2) {
        return false;
    }

    for (i = 2; i < value; i++) {
        if (value % i === 0) {
            return false;
        }
    }

    return true;
}

exports.handler = async function(event, context) {
    console.log('EVENT: ' + JSON.stringify(event, null, 2));
    const input = parseInt(Buffer.from(event.content, 'base64').toString('ascii'));
    const prime = isPrime(input);
    console.log('RESULT: ' + input + (prime ? ' is' : ' is not') + ' prime');
    if (!prime) {
        throw input + "is not prime";
    }
}
```

AWS AppConfig chama sua validação de Lambda ao chamar as operações de `ValidateConfigurationActivity` API `StartDeployment` e. Você deve fornecer permissões

`appconfig.amazonaws.com` para invocar seu Lambda. Para obter mais informações, consulte Concedendo acesso de funções aos AWS serviços. AWS AppConfig limita o tempo de execução da validação do Lambda a 15 segundos, incluindo a latência de inicialização.

Validadores do esquema JSON

Se você criar uma configuração em um documento do SSM, será necessário especificar ou criar um esquema JSON para essa configuração. Um esquema JSON define as propriedades permitidas para cada definição de configuração de aplicativo. O esquema JSON funciona como um conjunto de regras para garantir que definições de configuração novas ou atualizadas estejam em conformidade com as melhores práticas exigidas pelo aplicativo. Aqui está um exemplo.

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "title": "$id$",
  "description": "BasicFeatureToggle-1",
  "type": "object",
  "additionalProperties": false,
  "patternProperties": {
      "[^\\s]+$": {
          "type": "boolean"
      }
  },
  "minProperties": 1
}
```

Quando você cria uma configuração a partir de um documento SSM, o sistema verifica automaticamente se a configuração está em conformidade com os requisitos do esquema. Caso contrário, o AWS AppConfig retornará um erro de validação.

Important

Observe as seguintes informações importantes sobre validadores do esquema JSON:

- Os dados de configuração armazenados em documentos do SSM devem ser validados em relação a um esquema JSON associado antes que você possa adicionar a configuração ao sistema. Os parâmetros SSM não exigem um método de validação, mas recomendamos que você crie uma verificação de validação para configurações de parâmetros SSM novas ou atualizadas usando. AWS Lambda

- Uma configuração em um documento SSM usa o tipo de documento `ApplicationConfiguration`. O esquema JSON correspondente usa o tipo de documento `ApplicationConfigurationSchema`.
- AWS AppConfig suporta o esquema JSON versão 4.X para esquema embutido. Se a configuração do seu aplicativo exigir uma versão diferente do esquema JSON, será necessário criar um validador do Lambda.

Noções básicas sobre cotas e limitações do armazenamento de configuração

Os repositórios de configuração suportados pelo AWS AppConfig têm as seguintes cotas e limitações.

	AWS AppConfig armazenamento de configuração hospedado	Amazon S3	Systems Manager Parameter Store	AWS Secrets Manager	Systems Manager Document Store	AWS CodePipeline
Limite de tamanho da configuração	Padrão de 2 MB, máximo de 4 MB	2 MB Aplicado por AWS AppConfig, não pelo S3	4 KB (nível gratuito) / 8 KB (parâmetros avançados)	64 KB	64 KB	2 MB Imposta por AWS AppConfig, não CodePipeline
Limite de armazenamento de recursos	1 GB	Ilimitado	10.000 parâmetros (nível gratuito) / 100.000 parâmetros (parâmetr	500.000	500 documentos	Limitado pelo número de perfis de configuração por aplicativ

	AWS AppConfig armazenamento de configuração hospedado	Amazon S3	Systems Manager Parameter Store	AWS Secrets Manager	Systems Manager Document Store	AWS CodePipeline
			os avançados)			o (100 perfis por aplicativo)
Criptografia do lado do servidor	Sim	SSE-S3, SSE-KMS	Sim	Sim	Não	Sim
CloudFormation apoio	Sim	Não é para criar ou atualizar dados	Sim	Sim	Não	Sim
Preços	Gratuito	Consulte Def de preço do Amazon S3	Consulte Definição de preço do AWS Systems Manager	Consulte Definição de preço do AWS Secrets Manager	Gratuito	Consulte Definição de preço do AWS CodePipeline

Entendendo o armazenamento de configuração AWS AppConfig hospedado

AWS AppConfig inclui um armazenamento de configuração interno ou hospedado. As configurações devem ter 2 MB ou menos. O armazenamento de configuração AWS AppConfig hospedado oferece os seguintes benefícios em relação a outras opções de armazenamento de configuração.

- Não é necessário configurar outros serviços, como o Amazon Simple Storage Service (Amazon S3) ou o repositório de parâmetros.

- Você não precisa configurar permissões AWS Identity and Access Management (IAM) para usar o armazenamento de configurações.
- É possível armazenar configurações em YAML, JSON ou como documentos de texto.
- Não há custo para usar o armazenamento.
- É possível criar uma configuração e adicioná-la ao armazenamento ao criar um perfil de configuração.

Noções básicas sobre configurações armazenadas no Amazon S3

Você pode armazenar configurações em um bucket do Amazon Simple Storage Service (Amazon S3). Ao criar o perfil de configuração, você especifica o URI para um único objeto do S3 em um bucket. Você também especifica o Amazon Resource Name (ARN) de uma função AWS Identity and Access Management (IAM) que dá AWS AppConfig permissão para obter o objeto. Antes de criar um perfil de configuração para um objeto do Amazon S3, lembre-se das restrições a seguir.

Restrição	Detalhes
Tamanho	As configurações armazenadas como objetos do S3 podem ter um tamanho máximo de 1 MB.
Object encryption	Um perfil de configuração pode ter como alvo objetos criptografados SSE-S3 e SSE-KMS.
Classes de armazenamento	AWS AppConfig suporta as seguintes classes de armazenamento S3: `STANDARDINTELLIGENT_TIERING` ,`REDUCED_REDUNDANCY` ,`STANDARD_IA` , e. `ONEZONE_IA` As classes a seguir não são compatíveis: todas as classes do S3 Glacier (`GLACIER` e `DEEP_ARCHIVE`).
Versionamento	AWS AppConfig exige que o objeto S3 use o controle de versão.

Configuração de permissões para uma configuração armazenada como um objeto do Amazon S3

Ao criar um perfil de configuração para uma configuração armazenada como um objeto do S3, você deve especificar um ARN para uma função do IAM que AWS AppConfig dê permissão para obter o objeto. A função deve incluir as seguintes permissões:

Permissões para acessar o objeto do S3

- s3: GetObject
- s3: GetObjectVersion

Permissões para listar buckets do S3

s3: ListAllMyBuckets

Permissões para acessar o bucket do S3 em que o objeto está armazenado

- s3: GetBucketLocation
- s3: GetBucketVersioning
- s3: ListBucket
- s3: ListBucketVersions

Conclua o procedimento a seguir para criar uma função que permita AWS AppConfig obter uma configuração armazenada em um objeto do S3.

Criação da política do IAM para acessar um objeto do S3

Use o procedimento a seguir para criar uma política do IAM que permita AWS AppConfig obter uma configuração armazenada em um objeto do S3.

Para criar uma política do IAM para acessar um objeto do S3

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas e, em seguida, Criar política.
3. Na página Criar política, escolha a guia JSON.
4. Atualize a política de exemplo a seguir com informações sobre o bucket do S3 e o objeto de configuração. Depois, cole a política no campo de texto na guia JSON . Substitua *placeholder values* por suas próprias informações.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject",
        "s3:GetObjectVersion"
      ],
      "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/my-configurations/my-
configuration.json"
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation",
        "s3:GetBucketVersioning",
        "s3:ListBucketVersions",
        "s3:ListBucket"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket"
      ]
    },
    {
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    }
  ]
}
```

5. Selecione Revisar política.
6. Na página Review policy (Revisar política), digite um nome na caixa Name (Nome) e, depois, uma descrição.
7. Selecione Criar política. O sistema faz com que você retorne para a página Roles.

Criação do perfil do IAM para acessar um objeto do S3

Use o procedimento a seguir para criar uma função do IAM que permita AWS AppConfig obter uma configuração armazenada em um objeto do S3.

Para criar um perfil do IAM para acessar um objeto do Amazon S3

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis e Criar perfil.
3. Na seção Selecionar tipo de entidade confiável, escolha Serviço da AWS .
4. Na seção Choose a case use (Escolher um caso de uso) em Common use cases (Casos de uso comuns), escolha EC2 e, depois, escolha Next: Permissions (Próximo: Permissões).
5. Na página Attach permissions policy (Anexar política de permissões), na caixa de pesquisa, insira o nome da política criada no procedimento anterior.
6. Selecione essa política e escolha Next: Tags (Próximo: Tags).
7. Na página Adicionar tags (opcional), insira uma chave e um valor opcional e escolha Próximo: revisão.
8. Na página Review (Revisar), digite um nome na caixa Role name (Nome da função) e, depois, uma descrição.
9. Selecione Create role (Criar função). O sistema faz com que você retorne para a página Roles.
10. Na página Roles (Funções), escolha a função que você acabou de criar para abrir a página Summary (Resumo). Anote os valores de Role Name (Nome da função) e Role ARN (ARN da função). Você especificará o ARN da função ao criar o perfil de configuração posteriormente neste tópico.

Criar um relacionamento de confiança

Use o procedimento a seguir para configurar a função que você acabou de criar para confiar no AWS AppConfig.

Como adicionar um relacionamento de confiança

1. Na página Summary da função que você acabou de criar, escolha a guia Trust Relationships e, em seguida, Edit Trust Relationship.
2. Exclua `"ec2.amazonaws.com"` e adicione `"appconfig.amazonaws.com"`, conforme mostrado no exemplo a seguir.

JSON

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

3. Selecione Atualizar política de confiança.

Criação de um perfil AWS AppConfig de configuração de formato livre (console)

Use o procedimento a seguir para criar um perfil de configuração de AWS AppConfig forma livre e (opcionalmente) uma configuração de forma livre usando o console. AWS Systems Manager

Como criar um perfil de configuração de forma livre

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, escolha Aplicações e selecione uma aplicação que você criou em Criando um namespace para seu aplicativo no AWS AppConfig.
3. Selecione a guia Perfis de configuração e sinalizadores de atributos e escolha Criar configuração.
4. Na seção Opções de configuração, escolha Configuração de forma livre.
5. Em ID do perfil de configuração, insira um nome para o perfil de configuração.
6. (Opcional) Expanda Descrição e insira uma descrição.
7. (Opcional) Expanda Opções adicionais e conclua o seguinte, conforme necessário.
   a. Na seção Associar extensões, selecione uma extensão na lista.
   b. Na seção Tags, selecione Adicionar nova tag e especifique uma chave e um valor opcional.

8. Escolha Próximo.
9. Na página Especificar dados de configuração, na seção Definição de configuração, escolha uma opção.
10. Preencha os campos da opção que você selecionou, conforme descrito na tabela a seguir.

Opção selecionada	Detalhes
AWS AppConfig configuração hospedada	Selecione Texto, JSON ou YAML e insira sua configuração no campo. Vá para a Etapa 12 deste procedimento.
Objeto do Amazon S3	Insira o URI do objeto no campo Fonte do objeto do S3 e vá para a Etapa 11 deste procedimento.
AWS CodePipeline	Selecione Próximo e vá para a Etapa 12 deste procedimento.
Segredo do Secrets Manager	Escolha o segredo na lista e vá para a Etapa 11 deste procedimento.
AWS Systems Manager parameter	Escolha o parâmetro na lista e vá para a Etapa 11 deste procedimento.
AWS Systems Manager document	1. Escolha um documento na lista ou selecione Criar novo documento. 2. Se você selecionar Criar novo documento , em Nome do documento, insira um nome. Opcionalmente, expanda Nome da versão e insira um nome para a versão do documento. 3. Na seção Esquema de configuração da aplicação, escolha o esquema JSON usando a lista ou Criar esquema. Se escolher Criar esquema, o Systems Manager abrirá a página Criar esquema. Insira os detalhes do esquema e selecione

Opção selecionada	Detalhes
	Criar esquema de configuração da aplicação. 4. Na seção Content (Conteúdo) escolha YAML ou JSON e insira os dados de configuração no campo.

11. Na seção Função de serviço, escolha Nova função de serviço para AWS AppConfig criar a função do IAM que fornece acesso aos dados de configuração. AWS AppConfig preenche automaticamente o campo Nome da função com base no nome que você inseriu anteriormente. Ou selecione Perfil de serviço existente. Escolha a função usando a lista Role ARN (ARN de função).
12. Opcionalmente, na página Adicionar validadores, escolha Esquema JSON ou AWS Lambda. Se você escolher JSON Schema (Esquema JSON), insira o esquema JSON no campo. Se você escolher AWS Lambda, escolha o nome de recurso da Amazon (ARN) da função e a versão na lista.

> **Important**
>
> Os dados de configuração armazenados em documentos do SSM devem ser validados em relação a um esquema JSON associado antes que você possa adicionar a configuração ao sistema. Os parâmetros SSM não exigem um método de validação, mas recomendamos que você crie uma verificação de validação para configurações de parâmetros SSM novas ou atualizadas usando. AWS Lambda

13. Escolha Próximo.
14. Na página Revisar e salvar, selecione Salve continue a implantação.

> **Important**
>
> Se você criou um perfil de configuração para AWS CodePipeline, deverá criar um pipeline CodePipeline que especifique AWS AppConfig como o provedor de implantação. Você não precisa executar Implantando sinalizadores de recursos e dados de configuração no AWS AppConfig. No entanto, você deve configurar um cliente para receber atualizações de configuração do aplicativo conforme descrito em Recuperando dados de configuração sem AWS AppConfig o Agente. Para obter informações sobre a criação de um pipeline

que especifica AWS AppConfig como provedor de implantação, consulte Tutorial: Criar um pipeline usado AWS AppConfig como provedor de implantação no Guia do AWS CodePipeline usuário.

Vá para Implantando sinalizadores de recursos e dados de configuração no AWS AppConfig.

Criação de um perfil AWS AppConfig de configuração de formato livre (linha de comando)

O procedimento a seguir descreve como usar o AWS CLI (no Linux ou no Windows) ou Ferramentas da AWS para PowerShell criar um perfil de configuração de AWS AppConfig formato livre. Se preferir, você pode usar AWS CloudShell para executar os comandos listados abaixo. Para obter mais informações, consulte O que é o AWS CloudShell? no Guia do usuário do AWS CloudShell .

> **Note**
>
> Para configurações de formato livre hospedadas no repositório de configurações AWS AppConfig hospedado, você especifica o `hosted` URI de localização.

Para criar um perfil de configuração usando o AWS CLI

1. Abra AWS CLI o.
2. Execute o comando a seguir para criar um perfil de configuração de formato livre.

   Linux

```
aws appconfig create-configuration-profile \
  --application-id APPLICATION_ID \
  --name NAME \
  --description CONFIGURATION_PROFILE_DESCRIPTION \
  --location-uri CONFIGURATION_URI or hosted \
  --retrieval-role-arn IAM_ROLE_ARN \
  --tags TAGS \
  --validators "Content=SCHEMA_CONTENT or LAMBDA_FUNCTION_ARN,Type=JSON_SCHEMA
 or LAMBDA"
```

Windows

```
aws appconfig create-configuration-profile ^
  --application-id APPLICATION_ID ^
  --name NAME ^
  --description CONFIGURATION_PROFILE_DESCRIPTION ^
  --location-uri CONFIGURATION_URI or hosted  ^
  --retrieval-role-arn IAM_ROLE_ARN ^
  --tags TAGS ^
  --validators "Content=SCHEMA_CONTENT or LAMBDA_FUNCTION_ARN,Type=JSON_SCHEMA
 or LAMBDA"
```

PowerShell

```
New-APPCConfigurationProfile `
  -Name NAME `
  -ApplicationId APPLICATION_ID `
  -Description CONFIGURATION_PROFILE_DESCRIPTION `
  -LocationUri CONFIGURATION_URI or hosted `
  -RetrievalRoleArn IAM_ROLE_ARN `
  -Tag TAGS `
  -Validators "Content=SCHEMA_CONTENT or LAMBDA_FUNCTION_ARN,Type=JSON_SCHEMA
 or LAMBDA"
```

Important

Observe as seguintes informações importantes:

- Se você criou um perfil de configuração para AWS CodePipeline, deverá criar um pipeline CodePipeline que especifique AWS AppConfig como o provedor de implantação. Você não precisa executar Implantando sinalizadores de recursos e dados de configuração no AWS AppConfig. No entanto, você deve configurar um cliente para receber atualizações de configuração do aplicativo conforme descrito em Recuperando dados de configuração sem AWS AppConfig o Agente. Para obter informações sobre a criação de um pipeline que especifica AWS AppConfig como provedor de implantação, consulte Tutorial: Criar um pipeline usado AWS AppConfig como provedor de implantação no Guia do AWS CodePipeline usuário.

- Se você criou uma configuração no armazenamento de configuração AWS AppConfig hospedado, poderá criar novas versões da configuração usando as operações da CreateHostedConfigurationVersionAPI. Para ver AWS CLI detalhes e exemplos de comandos dessa operação de API, consulte create-hosted-configuration-versionna Referência de AWS CLI comandos.

Vá para Implantando sinalizadores de recursos e dados de configuração no AWS AppConfig.

Criar um perfil de configuração para fontes de dados não nativas

AWS AppConfig suporta a implantação de dados de configuração da maioria dos armazenamentos de dados. Nativamente, o AWS AppConfig oferece suporte à implantação de dados de configuração armazenados nos seguintes serviços:

- O armazenamento de configuração AWS AppConfig hospedado
- Amazon S3
- AWS Secrets Manager
- AWS Systems Manager Armazenamento de parâmetros
- Systems Manager Document Store
- AWS CodePipeline

Se seus dados de configuração estiverem armazenados em um local sem suporte nativo AWS AppConfig, você poderá criar uma AWS AppConfig extensão para recuperar os dados da fonte. Por exemplo, usando uma AWS AppConfig extensão, você pode recuperar dados de configuração armazenados no Amazon Relational Database Service (Amazon RDS), no Amazon DynamoDB (DynamoDB GitHub) GitLab,,, ou em um repositório local, para citar alguns. Ao implementar uma extensão, você pode aproveitar a AWS AppConfig segurança e DevOps os aprimoramentos de seus aplicativos e ambiente de computação. Você também pode usar esse método ao migrar dados de configuração de sistemas legados para o AWS AppConfig.

A criação de um perfil de configuração para fontes de dados sem suporte nativo no AWS AppConfig envolve os seguintes processos ou ações:

1. Crie uma AWS Lambda função que busque dados da sua fonte de dados. Desde que uma função Lambda possa acessar a fonte de dados, sua AWS AppConfig extensão poderá recuperar os dados.

2. Crie uma AWS AppConfig extensão personalizada que invoque sua função Lambda. Para obter mais informações, consulte Passo a passo: Criação de extensões personalizadas AWS AppConfig.
3. Crie um perfil de configuração AWS AppConfig de formato livre. Especificamente, crie um perfil de configuração que use a definição de configuração hospedada do AWS AppConfig . O perfil de configuração funciona como um armazenamento de dados temporário depois que a função do Lambda recupera a configuração da sua fonte. Seu aplicativo recuperará os dados de configuração do armazenamento de configuração AWS AppConfig hospedado. Para obter mais informações, consulte Criando um perfil de configuração de formato livre no AWS AppConfig.
4. Crie uma associação de extensão que seja acionada usando o ponto de ação do `PRE_CREATE_HOSTED_CONFIGURATION_VERSION`. Para obter mais informações, consulte Etapa 4: criar uma associação de extensão para uma AWS AppConfig extensão personalizada.

Após a configuração, quando o aplicativo solicita uma nova versão dos dados de configuração, o Lambda busca esses dados e os insere no perfil de configuração. Em seguida, o AWS AppConfig salva o perfil de configuração e seus dados de terceiros.

Quando estiver pronto, você poderá implantar o perfil de configuração nos seus aplicativos, assim como qualquer outro tipo de dado de configuração.

> **Note**
>
> Você pode optar por inserir dados de terceiros de acordo com os dados de configuração existentes ou fazer com que todo o conteúdo dos dados de configuração contenha somente os dados de terceiros. Se você quiser alinhar os dados com outros dados existentes, essa lógica deverá fazer parte da função do Lambda que importa os dados da fonte de terceiros.

Migração AWS AppConfig de serviços de configuração antigos e internos

Se você começou a usar AWS AppConfig e ainda tem dados de configuração legados ou sinalizadores de recursos em outro sistema, você pode usar o processo descrito anteriormente neste tópico para migrar do seu sistema legado para dentro. AWS AppConfig Você pode criar uma extensão que extrai dados do seu sistema antigo e os implanta por meio dele. AWS AppConfig O uso dessa AWS AppConfig forma fornece todos os controles e benefícios da proteção de segurança enquanto ainda usa seus armazenamentos de dados antigos.

Implantando sinalizadores de recursos e dados de configuração no AWS AppConfig

Depois de criar os artefatos necessários para trabalhar com sinalizadores de atributos e dados de configuração de formato livre, você pode criar uma nova implantação. Ao criar uma nova implantação, você especifica as seguintes informações:

- O ID do aplicativo
- O ID do perfil de configuração
- A versão da configuração
- O ID do ambiente no qual você deseja implantar os dados de configuração
- O ID da estratégia de implantação que define a rapidez com que você deseja que as alterações entrem em vigor
- Um ID de chave AWS Key Management Service (AWS KMS) para criptografar os dados usando uma chave gerenciada pelo cliente.

Quando você chama a ação StartDeploymentda API, AWS AppConfig executa as seguintes tarefas:

1. Recupera os dados de configuração do armazenamento de dados subjacente usando o URI de localização no perfil de configuração.
2. Verifica se os dados de configuração estão sintática e semanticamente corretos usando os validadores especificados ao criar o perfil de configuração.
3. Armazena em cache uma cópia dos dados para que estejam prontos para serem recuperados pelo seu aplicativo. Essa cópia em cache é chamada de dados implantados.

Você pode mitigar situações em que a implantação de dados de configuração causa erros em seu aplicativo usando uma combinação de estratégias de AWS AppConfig implantação e reversões automáticas com base nos alarmes da Amazon. CloudWatch Uma estratégia de implantação permite que você libere lentamente as alterações nos ambientes de produção em questão de minutos ou horas. Depois de configurado, se um ou mais CloudWatch alarmes entrarem no estado de alarme durante uma implantação, reverterá AWS AppConfig automaticamente seus dados de configuração para a versão anterior. Para ter mais informações sobre estratégias de implantação, consulte Como trabalhar com estratégias de implantação. Para ter mais informações sobre reversões automáticas, consulte Monitorar implantações para reversão automática.

Tópicos

- Como trabalhar com estratégias de implantação
- Implantar uma configuração
- Implantando AWS AppConfig configurações usando CodePipeline
- Reverter uma configuração

Como trabalhar com estratégias de implantação

Uma estratégia de implantação permite que você libere lentamente as alterações nos ambientes de produção em questão de minutos ou horas. Uma estratégia AWS AppConfig de implantação define os seguintes aspectos importantes de uma implantação de configuração.

<table>
<tr><th>Configuração</th><th>Description</th></tr>
<tr><td>Tipo de implantação</td><td>O tipo de implantação define como a configuração é implantada ou implementada. AWS AppConfig suporta tipos de implantação linear e exponencial.
Linear: para esse tipo, AWS AppConfig processa a implantação por incrementos do fator de crescimento distribuídos uniformemente pela implantação. Veja um exemplo de cronograma para uma implantação de 10 horas que usa 20% de crescimento linear:<table><tr><th>Tempo decorrido</th><th>Progresso da implantação</th></tr><tr><td>0 hora</td><td>0%</td></tr><tr><td>2 horas</td><td>20%</td></tr><tr><td>4 horas</td><td>40%</td></tr><tr><td>6 horas</td><td>60%</td></tr></table></td></tr>
</table>

Configuração	Description

Tempo decorrido	Progresso da implantação
8 horas	80%
10 horas	100%

- Exponencial: para esse tipo, o AWS AppConfig processa a implantação exponencialmente usando a seguinte fórmula: `G*(2^N)`. Nessa fórmula, `G` é a porcentagem de etapa especificada pelo usuário e `N` é o número de etapas até que a configuração seja implantada em todos os destinos. Por exemplo, se você especific ar um fator de crescimento de 2, o sistema implementará a configuração da seguinte maneira:

```
2*(2^0)
2*(2^1)
2*(2^2)
```

Expresso numericamente, a implantação é feita da seguinte forma: 2% dos alvos, 4% dos alvos, 8% dos alvos e continua até que a configuração tenha sido implantada em todos os destinos.

Configuração	Description
Percentagem de etapa (fator de crescimento)	Essa configuração especifica a porcentagem de chamadores a serem direcionados durante cada etapa da implantação. **Note** No SDK e na Referência da API do AWS AppConfig, `step percentage` é chamado de `growth factor`.
Tempo de implantação	Essa configuração especifica o período de tempo durante o qual é AWS AppConfig implantado nos hosts. Isso não é um valor de tempo limite. É uma janela de tempo durante a qual a implantação é processada em intervalos.
Tempo de incorporação	Essa configuração especifica a quantidade de tempo que AWS AppConfig monitora os CloudWatch alarmes da Amazon após a configuração ter sido implantada em 100% de suas metas, antes de considerar que a implantação foi concluída. Se um alarme for acionado durante esse período, o AWS AppConfig reverterá a implantação. Você deve configurar as permissões AWS AppConfig para reverter com base nos CloudWatch alarmes. Para obter mais informações, consulte Configurar permissões para reversão automática.

Você pode escolher uma estratégia predefinida incluída AWS AppConfig ou criar a sua própria.

Tópicos

- Usar estratégias de implantação predefinidas

- Criar uma estratégia de implantação

Usar estratégias de implantação predefinidas

AWS AppConfig inclui estratégias de implantação predefinidas para ajudá-lo a implantar rapidamente uma configuração. Em vez de criar suas próprias estratégias, é possível escolher uma das seguintes opções ao implantar uma configuração.

Estratégia de implantação	Description
AppConfig. PercentEvery Linear 20 6 minutos	AWS recomendado: Essa estratégia implanta a configuração para 20% de todos os destinos a cada seis minutos para uma implantação de 30 minutos. O sistema monitora os CloudWatch alarmes da Amazon por 30 minutos. Se nenhum alarme for recebido nesse período, a implantação será concluída. Se um alarme for acionado durante esse período, AWS AppConfig reverte a implantação. Recomendamos usar essa estratégia para implantações de produção porque ela se alinha às AWS melhores práticas e inclui ênfase adicional na segurança da implantação devido à sua longa duração e tempo de cozimento.
AppConfig. Canário 10 por cento 20 minutos	AWS recomendado: Essa estratégia processa a implantação exponencialmente usando um fator de crescimento de 10% durante 20 minutos. O sistema monitora CloudWatch os alarmes por 10 minutos. Se nenhum alarme for recebido nesse período, a implantação será concluída . Se um alarme for acionado durante esse

Estratégia de implantação	Description
	período, AWS AppConfig reverte a implantação. Recomendamos usar essa estratégia para implantações de produção porque ela se alinha às AWS melhores práticas para implantações de configuração.
AppConfig.AllAtOnce	Rápida: Essa estratégia implanta a configuração em todos os destinos de forma imediata. O sistema monitora CloudWatch os alarmes por 10 minutos. Se nenhum alarme for recebido nesse período, a implantação será concluída. Se um alarme for acionado durante esse período, o AWS AppConfig reverterá a implantação.
AppConfig. PercentEvery Linear 50 30 segundos	Teste/demonstração: Essa estratégia implanta a configuração para metade de todos os destinos a cada 30 segundos para uma implantação de um minuto. O sistema monitora os CloudWatch alarmes da Amazon por 1 minuto. Se nenhum alarme for recebido nesse período, a implantação será concluída. Se um alarme for acionado durante esse período, AWS AppConfig reverte a implantação. Recomendamos usar esta estratégia apenas para fins de teste ou demonstração, pois ela tem uma curta duração e tempo de incorporação.

Criar uma estratégia de implantação

Se não quiser utilizar uma das estratégias de implantação predefinidas, você poderá criar uma. Você pode criar um máximo de 20 estratégias de implantação. Ao implantar uma configuração, você pode escolher a estratégia de implantação mais adequada para o aplicativo e o ambiente.

Criação de uma estratégia de AWS AppConfig implantação (console)

Use o procedimento a seguir para criar uma estratégia de AWS AppConfig implantação usando o AWS Systems Manager console.

Como criar uma estratégia de implantação

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, escolha Estratégias de implantação e selecione Criar estratégia de implantação.
3. Em Name (Nome), insira um nome para a estratégia de implantação.
4. Em Description (Descrição), insira informações sobre a estratégia de implantação.
5. Em Deployment type (Tipo de implantação), selecione um tipo.
6. Em Step percentage (Porcentagem de etapa), escolha a porcentagem de chamadores a serem direcionados durante cada etapa da implantação.
7. Em Deployment time (Tempo de implantação), insira a duração total da implantação em minutos ou horas.
8. Em Bake time, insira o tempo total, em minutos ou horas, para monitorar CloudWatch os alarmes da Amazon antes de prosseguir para a próxima etapa de uma implantação ou antes de considerar que a implantação foi concluída.
9. Na seção Tags, insira uma chave e um valor opcional. Você pode especificar um máximo de 50 tags para um recurso.
10. Escolha Create deployment strategy (Criar estratégia de implantação).

> **Important**
>
> Se você criou um perfil de configuração para AWS CodePipeline, deverá criar um pipeline CodePipeline que especifique AWS AppConfig como o provedor de implantação. Você não precisa executar Implantar uma configuração. No entanto, você deve configurar um cliente

para receber atualizações de configuração do aplicativo conforme descrito em Recuperando dados de configuração sem AWS AppConfig o Agente. Para obter informações sobre a criação de um pipeline que especifica AWS AppConfig como provedor de implantação, consulte Tutorial: Criar um pipeline usado AWS AppConfig como provedor de implantação no Guia do AWS CodePipeline usuário.

Vá para Implantar uma configuração.

Criação de uma estratégia de AWS AppConfig implantação (linha de comando)

O procedimento a seguir descreve como usar o AWS CLI (no Linux ou no Windows) ou Ferramentas da AWS para PowerShell criar uma estratégia de AWS AppConfig implantação.

Para criar uma estratégia de implantação passo a passo

1. Abra AWS CLI o.
2. Execute o seguinte comando para criar uma estratégia de implantação.

   Linux

```
aws appconfig create-deployment-strategy \
  --name A_name_for_the_deployment_strategy \
  --description A_description_of_the_deployment_strategy \
  --deployment-duration-in-minutes Total_amount_of_time_for_a_deployment_to_last
 \
  --final-bake-time-in-minutes Amount_of_time_AWS
 AppConfig_monitors_for_alarms_before_considering_the_deployment_to_be_complete
 \
  --growth-
factor The_percentage_of_targets_to_receive_a_deployed_configuration_during_each_interva
 \
  --growth-
type The_linear_or_exponential_algorithm_used_to_define_how_percentage_grows_over_time
 \
  --replicate-
to To_save_the_deployment_strategy_to_a_Systems_Manager_(SSM)_document \
  --tags User_defined_key_value_pair_metadata_of_the_deployment_strategy
```

Windows

```
aws appconfig create-deployment-strategy ^
  --name A_name_for_the_deployment_strategy ^
  --description A_description_of_the_deployment_strategy ^
  --deployment-duration-in-minutes Total_amount_of_time_for_a_deployment_to_last
 ^
  --final-bake-time-in-minutes Amount_of_time_AWS
 AppConfig_monitors_for_alarms_before_considering_the_deployment_to_be_complete
 ^
  --growth-
factor The_percentage_of_targets_to_receive_a_deployed_configuration_during_each_interva
 ^
  --growth-
type The_linear_or_exponential_algorithm_used_to_define_how_percentage_grows_over_time
 ^
  --name A_name_for_the_deployment_strategy ^
  --replicate-
to To_save_the_deployment_strategy_to_a_Systems_Manager_(SSM)_document ^
  --tags User_defined_key_value_pair_metadata_of_the_deployment_strategy
```

PowerShell

```
New-APPCDeploymentStrategy `
  --Name A_name_for_the_deployment_strategy `
  --Description A_description_of_the_deployment_strategy `
  --DeploymentDurationInMinutes Total_amount_of_time_for_a_deployment_to_last `
  --FinalBakeTimeInMinutes Amount_of_time_AWS
 AppConfig_monitors_for_alarms_before_considering_the_deployment_to_be_complete
 `
  --
GrowthFactor The_percentage_of_targets_to_receive_a_deployed_configuration_during_each_i
 `
  --
GrowthType The_linear_or_exponential_algorithm_used_to_define_how_percentage_grows_over_
 `
  --
ReplicateTo To_save_the_deployment_strategy_to_a_Systems_Manager_(SSM)_document
 `
  --
Tag Hashtable_type_User_defined_key_value_pair_metadata_of_the_deployment_strategy
```

O sistema retorna informações como estas.

Linux

```
{
   "Id": "Id of the deployment strategy",
   "Name": "Name of the deployment strategy",
   "Description": "Description of the deployment strategy",
   "DeploymentDurationInMinutes": "Total amount of time the deployment lasted",
   "GrowthType": "The linear or exponential algorithm used to define how
 percentage grew over time",
   "GrowthFactor": "The percentage of targets that received a deployed
 configuration during each interval",
   "FinalBakeTimeInMinutes": "The amount of time AWS AppConfig monitored for
 alarms before considering the deployment to be complete",
   "ReplicateTo": "The Systems Manager (SSM) document where the deployment
 strategy is saved"
}
```

Windows

```
{
   "Id": "Id of the deployment strategy",
   "Name": "Name of the deployment strategy",
   "Description": "Description of the deployment strategy",
   "DeploymentDurationInMinutes": "Total amount of time the deployment lasted",
   "GrowthType": "The linear or exponential algorithm used to define how
 percentage grew over time",
   "GrowthFactor": "The percentage of targets that received a deployed
 configuration during each interval",
   "FinalBakeTimeInMinutes": "The amount of time AWS AppConfig monitored for
 alarms before considering the deployment to be complete",
   "ReplicateTo": "The Systems Manager (SSM) document where the deployment
 strategy is saved"
}
```

PowerShell

```
ContentLength               : Runtime of the command
DeploymentDurationInMinutes : Total amount of time the deployment lasted
Description                 : Description of the deployment strategy
```

```
FinalBakeTimeInMinutes      : The amount of time AWS AppConfig monitored for
 alarms before considering the deployment to be complete
GrowthFactor                : The percentage of targets that received a deployed
 configuration during each interval
GrowthType                  : The linear or exponential algorithm used to define
 how percentage grew over time
HttpStatusCode              : HTTP Status of the runtime
Id                          : The deployment strategy ID
Name                        : Name of the deployment strategy
ReplicateTo                 : The Systems Manager (SSM) document where the
 deployment strategy is saved
ResponseMetadata            : Runtime Metadata
```

Implantar uma configuração

Depois de criar os artefatos necessários para trabalhar com sinalizadores de recursos e dados de configuração de formato livre, você pode criar uma nova implantação usando o Console de gerenciamento da AWS, o ou o AWS CLI SDK. Iniciar uma implantação em AWS AppConfig chama a operação StartDeploymentda API. Essa chamada inclui o AWS AppConfig aplicativo, o ambiente, o perfil IDs de configuração e (opcionalmente) a versão dos dados de configuração a serem implantados. A chamada também inclui o ID da estratégia de implantação a ser usada, que determina como os dados de configuração são implantados.

Se você implantar segredos armazenados em AWS Secrets Manager objetos do Amazon Simple Storage Service (Amazon S3) criptografados com uma chave gerenciada pelo cliente ou parâmetros de cadeia de caracteres seguros armazenados AWS Systems Manager no Parameter Store criptografados com uma chave gerenciada pelo cliente, você deverá especificar um valor para o parâmetro. `KmsKeyIdentifier` Se sua configuração não estiver criptografada ou estiver criptografada com um Chave gerenciada pela AWS, não é necessário especificar um valor para o `KmsKeyIdentifier` parâmetro.

> **Note**
>
> O valor especificado para `KmsKeyIdentifier` deve ser uma chave gerenciada pelo cliente. Não precisa ser a mesma chave que você usou para criptografar sua configuração. Quando você inicia uma implantação com um`KmsKeyIdentifier`, a política de permissão anexada ao seu diretor AWS Identity and Access Management (IAM) deve permitir a `kms:GenerateDataKey` operação.

AWS AppConfig monitora a distribuição para todos os hosts e relata o status. Se uma distribuição falhar, AWS AppConfig reverte a configuração.

> **Note**
>
> Você só pode implantar uma única configuração por vez em determinado ambiente. No entanto, pode implantar uma única configuração em ambientes diferentes ao mesmo tempo.

Implantar uma configuração (console)

Use o procedimento a seguir para implantar uma AWS AppConfig configuração usando o AWS Systems Manager console.

Como implantar uma configuração usando o console

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, escolha Aplicações e selecione uma aplicação que você criou em Criando um namespace para seu aplicativo no AWS AppConfig.
3. Na guia Ambientes, use o botão de opção para escolher um ambiente e, depois, selecione Visualizar detalhes.
4. Selecione Iniciar implantação.
5. Em Configuration (Configuração), escolha uma configuração na lista.
6. Dependendo da origem da configuração, use a lista de versões para escolher a versão que deseja implantar.
7. Em Deployment strategy (Estratégia de implantação), escolha uma estratégia da lista.
8. (Opcional) Em Descrição da implantação, insira uma descrição.
9. Para opções adicionais de criptografia, escolha uma AWS Key Management Service chave na lista.
10. (Opcional) Na seção Tags, escolha Adicionar nova tag e insira uma chave e um valor opcional. Você pode especificar um máximo de 50 tags para um recurso.
11. Selecione Iniciar implantação.

Implantar uma configuração (linha de comando)

O procedimento a seguir descreve como usar o AWS CLI (no Linux ou no Windows) ou Ferramentas da AWS para PowerShell implantar uma AWS AppConfig configuração.

Para implantar uma configuração passo a passo

1. Abra AWS CLI o.
2. Execute o comando a seguir para implantar uma configuração.

   Linux

   ```
   aws appconfig start-deployment \
     --application-id The_application_ID \
     --environment-id The_environment_ID \
     --deployment-strategy-id The_deployment_strategy_ID \
     --configuration-profile-id The_configuration_profile_ID \
     --configuration-version The_configuration_version_to_deploy \
     --description A_description_of_the_deployment \
     --tags User_defined_key_value_pair_metadata_of_the_deployment
   ```

   Windows

   ```
   aws appconfig start-deployment ^
     --application-id The_application_ID ^
     --environment-id The_environment_ID ^
     --deployment-strategy-id The_deployment_strategy_ID ^
     --configuration-profile-id The_configuration_profile_ID ^
     --configuration-version The_configuration_version_to_deploy ^
     --description A_description_of_the_deployment ^
     --tags User_defined_key_value_pair_metadata_of_the_deployment
   ```

   PowerShell

   ```
   Start-APPCDeployment `
     -ApplicationId The_application_ID `
     -ConfigurationProfileId The_configuration_profile_ID `
     -ConfigurationVersion The_configuration_version_to_deploy `
     -DeploymentStrategyId The_deployment_strategy_ID `
     -Description A_description_of_the_deployment `
     -EnvironmentId The_environment_ID `
   ```

```
  -Tag Hashtable_type_user_defined_key_value_pair_metadata_of_the_deployment
```

O sistema retorna informações como estas.

Linux

```
{
   "ApplicationId": "The ID of the application that was deployed",
   "EnvironmentId" : "The ID of the environment",
   "DeploymentStrategyId": "The ID of the deployment strategy that was
 deployed",
   "ConfigurationProfileId": "The ID of the configuration profile that was
 deployed",
   "DeploymentNumber": The sequence number of the deployment,
   "ConfigurationName": "The name of the configuration",
   "ConfigurationLocationUri": "Information about the source location of the
 configuration",
   "ConfigurationVersion": "The configuration version that was deployed",
   "Description": "The description of the deployment",
   "DeploymentDurationInMinutes": Total amount of time the deployment lasted,
   "GrowthType": "The linear or exponential algorithm used to define how
 percentage grew over time",
   "GrowthFactor": The percentage of targets to receive a deployed configuration
 during each interval,
   "FinalBakeTimeInMinutes": Time AWS AppConfig monitored for alarms before
 considering the deployment to be complete,
   "State": "The state of the deployment",

   "EventLog": [
      {
         "Description": "A description of the deployment event",
         "EventType": "The type of deployment event",
         "OccurredAt": The date and time the event occurred,
         "TriggeredBy": "The entity that triggered the deployment event"
      }
   ],

   "PercentageComplete": The percentage of targets for which the deployment is
 available,
   "StartedAt": The time the deployment started,
   "CompletedAt": The time the deployment completed
}
```

Windows

```
{
   "ApplicationId": "The ID of the application that was deployed",
   "EnvironmentId" : "The ID of the environment",
   "DeploymentStrategyId": "The ID of the deployment strategy that was
 deployed",
   "ConfigurationProfileId": "The ID of the configuration profile that was
 deployed",
   "DeploymentNumber": The sequence number of the deployment,
   "ConfigurationName": "The name of the configuration",
   "ConfigurationLocationUri": "Information about the source location of the
 configuration",
   "ConfigurationVersion": "The configuration version that was deployed",
   "Description": "The description of the deployment",
   "DeploymentDurationInMinutes": Total amount of time the deployment lasted,
   "GrowthType": "The linear or exponential algorithm used to define how
 percentage grew over time",
   "GrowthFactor": The percentage of targets to receive a deployed configuration
 during each interval,
   "FinalBakeTimeInMinutes": Time AWS AppConfig monitored for alarms before
 considering the deployment to be complete,
   "State": "The state of the deployment",

   "EventLog": [
      {
         "Description": "A description of the deployment event",
         "EventType": "The type of deployment event",
         "OccurredAt": The date and time the event occurred,
         "TriggeredBy": "The entity that triggered the deployment event"
      }
   ],

   "PercentageComplete": The percentage of targets for which the deployment is
 available,
   "StartedAt": The time the deployment started,
   "CompletedAt": The time the deployment completed
}
```

PowerShell

```
ApplicationId               : The ID of the application that was deployed
```

```
CompletedAt                  : The time the deployment completed
ConfigurationLocationUri     : Information about the source location of the
 configuration
ConfigurationName            : The name of the configuration
ConfigurationProfileId       : The ID of the configuration profile that was
 deployed
ConfigurationVersion         : The configuration version that was deployed
ContentLength                : Runtime of the deployment
DeploymentDurationInMinutes : Total amount of time the deployment lasted
DeploymentNumber             : The sequence number of the deployment
DeploymentStrategyId         : The ID of the deployment strategy that was
 deployed
Description                  : The description of the deployment
EnvironmentId                : The ID of the environment that was deployed
EventLog                     : {Description : A description of the deployment
 event, EventType : The type of deployment event, OccurredAt : The date and time
 the event occurred,
         TriggeredBy : The entity that triggered the deployment event}
FinalBakeTimeInMinutes       : Time AWS AppConfig monitored for alarms before
 considering the deployment to be complete
GrowthFactor                 : The percentage of targets to receive a deployed
 configuration during each interval
GrowthType                   : The linear or exponential algorithm used to define
 how percentage grew over time
HttpStatusCode               : HTTP Status of the runtime
PercentageComplete           : The percentage of targets for which the deployment
 is available
ResponseMetadata             : Runtime Metadata
StartedAt                    : The time the deployment started
State                        : The state of the deployment
```

Implantando AWS AppConfig configurações usando CodePipeline

AWS AppConfig é uma ação de implantação integrada para AWS CodePipeline (CodePipeline). CodePipeline é um serviço de entrega contínua totalmente gerenciado que ajuda você a automatizar seus pipelines de lançamento para atualizações rápidas e confiáveis de aplicativos e infraestrutura. CodePipeline automatiza as fases de criação, teste e implantação do seu processo de lançamento sempre que há uma alteração no código, com base no modelo de lançamento que você define. Para ter mais informações, consulte O que é o AWS CodePipeline?

A integração AWS AppConfig com CodePipeline oferece os seguintes benefícios:

- Os clientes que CodePipeline costumavam gerenciar a orquestração agora têm um meio leve de implantar alterações de configuração em seus aplicativos sem precisar implantar toda a base de código.
- Os clientes que desejam usar AWS AppConfig para gerenciar implantações de configuração, mas estão limitados porque AWS AppConfig não oferecem suporte ao código ou armazenamento de configuração atual, agora têm opções adicionais. CodePipeline suporta AWS CodeCommit, GitHub, e BitBucket (para citar alguns).

> **Note**
>
> AWS AppConfig a integração com só CodePipeline é suportada Regiões da AWS onde CodePipeline está disponível.

Como funciona a integração

Você começa instalando e configurando. CodePipeline Isso inclui adicionar sua configuração a um armazenamento CodePipeline de código compatível. Em seguida, você configura seu AWS AppConfig ambiente executando as seguintes tarefas:

- Crie um namespace e um perfil de configuração
- Escolha uma estratégia de implantação predefinida ou crie sua própria

Depois de concluir essas tarefas, você cria um pipeline CodePipeline que especifica AWS AppConfig como o provedor de implantação. Em seguida, você pode fazer uma alteração na sua configuração e enviá-la para o seu armazenamento de CodePipeline código. O upload da nova configuração inicia automaticamente uma nova implantação em CodePipeline. Depois da conclusão da implantação, você pode verificar as alterações. Para obter informações sobre a criação de um pipeline que especifica AWS AppConfig como provedor de implantação, consulte Tutorial: Criar um pipeline usado AWS AppConfig como provedor de implantação no Guia do AWS CodePipeline usuário.

Reverter uma configuração

Durante uma implantação, é possível atenuar situações em que dados de configuração malformados ou incorretos causam erros na aplicação usando reversões automáticas (se um alarme for acionado

durante uma implantação) ou revertendo os dados de configuração para a versão anterior (se a implantação for concluída com êxito).

Para reversões automáticas, você pode usar uma combinação de estratégias de AWS AppConfig implantação e alarmes da Amazon CloudWatch . Depois de configurado, se um ou mais CloudWatch alarmes entrarem no `ALARM` estado durante uma implantação, AWS AppConfig reverterá automaticamente seus dados de configuração para a versão anterior, evitando interrupções ou erros no aplicativo. Para começar, consulte o Configurar permissões para reversão automática.

> **Note**
>
> Você também pode reverter uma configuração chamando a operação da StopDeploymentAPI enquanto a implantação ainda está em andamento.

Para implantações concluídas com êxito, AWS AppConfig também oferece suporte à reversão dos dados de configuração para uma versão anterior usando o `AllowRevert` parâmetro com a operação da StopDeploymentAPI. Para alguns clientes, a reversão para uma configuração anterior após uma implantação bem-sucedida garante que os dados sejam os mesmos de antes da implantação. A reversão também ignora os monitores de alarme, o que pode impedir o roll forward durante uma emergência na aplicação.

> **Important**
>
> Se você ligar `StopDeployment` com o `AllowRevert` parâmetro ativado, AWS AppConfig reverterá a implantação somente se a implantação for bem-sucedida nas últimas 72 horas. Depois de 72 horas, a implantação não pode mais ser revertida. É necessário criar uma implantação.

Veja um detalhamento da funcionalidade `StopDeployment` com base em diferentes situações.

1. Se `StopDeployment` for chamada em uma implantação em andamento, o estado de implantação resultante será `ROLLED_BACK`.
2. Se `StopDeployment` (com `AllowRevert`) for chamada em uma implantação em andamento, o estado de implantação resultante será `ROLLED_BACK`.
3. Se `StopDeployment` for chamada em uma implantação concluída, uma `BadRequestException` será lançada.

4. Se `StopDeployment` (com `AllowRevert`) for chamada em uma implantação concluída, o estado de implantação resultante será `REVERTED`.
5. Se `StopDeployment` (com `AllowRevert`) for chamada em uma implantação concluída após 72 horas, uma `BadRequestException` será lançada.

Você pode usar o AWS CLI para chamar a StopDeploymentoperação com o `AllowRevert` parâmetro. Aqui está um exemplo de AWS CLI comando que inclui o `AllowRevert` parâmetro.

```
aws appconfig stop-deployment \
    --application-id 339ohji \
    --environment-id 54j1r29 \
    --deployment-number 2 \
    --allow-revert
```

Recuperando sinalizadores de recursos e dados de configuração em AWS AppConfig

Seu aplicativo recupera sinalizadores de recursos e dados de configuração de formato livre estabelecendo uma sessão de configuração usando o serviço de AWS AppConfig dados. Recomendamos que você use o AWS AppConfig Agent para recuperar dados de configuração. O agente (ou a extensão do AWS AppConfig Agent Lambda para ambientes computacionais Lambda) gerencia uma série de chamadas de API e tokens de sessão em seu nome. Basicamente, o fluxo do processo é o seguinte:

1. Você configura o AWS AppConfig Agente como um host local e faz com que o agente faça uma pesquisa AWS AppConfig para atualizações de configuração.
2. O agente chama as ações da GetLatestConfigurationAPI StartConfigurationSessione armazena em cache seus dados de configuração localmente.
3. Para recuperar os dados, seu aplicativo faz uma chamada HTTP para o servidor localhost. AWS AppConfig O agente oferece suporte a vários casos de uso, conforme descrito emComo usar o AWS AppConfig Agent para recuperar dados de configuração.

Se preferir, você pode chamar manualmente essas ações da API para recuperar uma configuração. O processo da API funciona da seguinte maneira:

1. Seu aplicativo estabelece uma sessão de configuração usando a ação `StartConfigurationSession` da API. O cliente da sua sessão então faz chamadas periódicas para `GetLatestConfiguration` para verificar e recuperar os dados mais recentes disponíveis.
2. Ao chamar`StartConfigurationSession`, seu código envia identificadores (ID ou nome) de um AWS AppConfig aplicativo, ambiente e perfil de configuração que a sessão rastreia.
3. Em resposta, AWS AppConfig fornece um `InitialConfigurationToken` a ser entregue ao cliente da sessão e usado na primeira vez em que ele liga `GetLatestConfiguration` para essa sessão.
4. Ao chamar `GetLatestConfiguration`, seu código de cliente envia o valor `ConfigurationToken` mais recente que ele tem e recebe em resposta:
   - `NextPollConfigurationToken`: o valor `ConfigurationToken` a ser usado na próxima chamada para `GetLatestConfiguration`.

- A configuração: os dados mais recentes destinados à sessão. Este pode estar vazio se o cliente já tiver a versão mais recente da configuração.

> **Note**
>
> A recuperação de dados de configuração de um arquivo separado Conta da AWS não é suportada.

Conteúdo

- O que é AWS AppConfig Agente?
- Como usar o AWS AppConfig Agent para recuperar dados de configuração
- AWS AppConfig considerações sobre o uso de navegadores e dispositivos móveis
- Recuperando dados de configuração sem AWS AppConfig o Agente

O que é AWS AppConfig Agente?

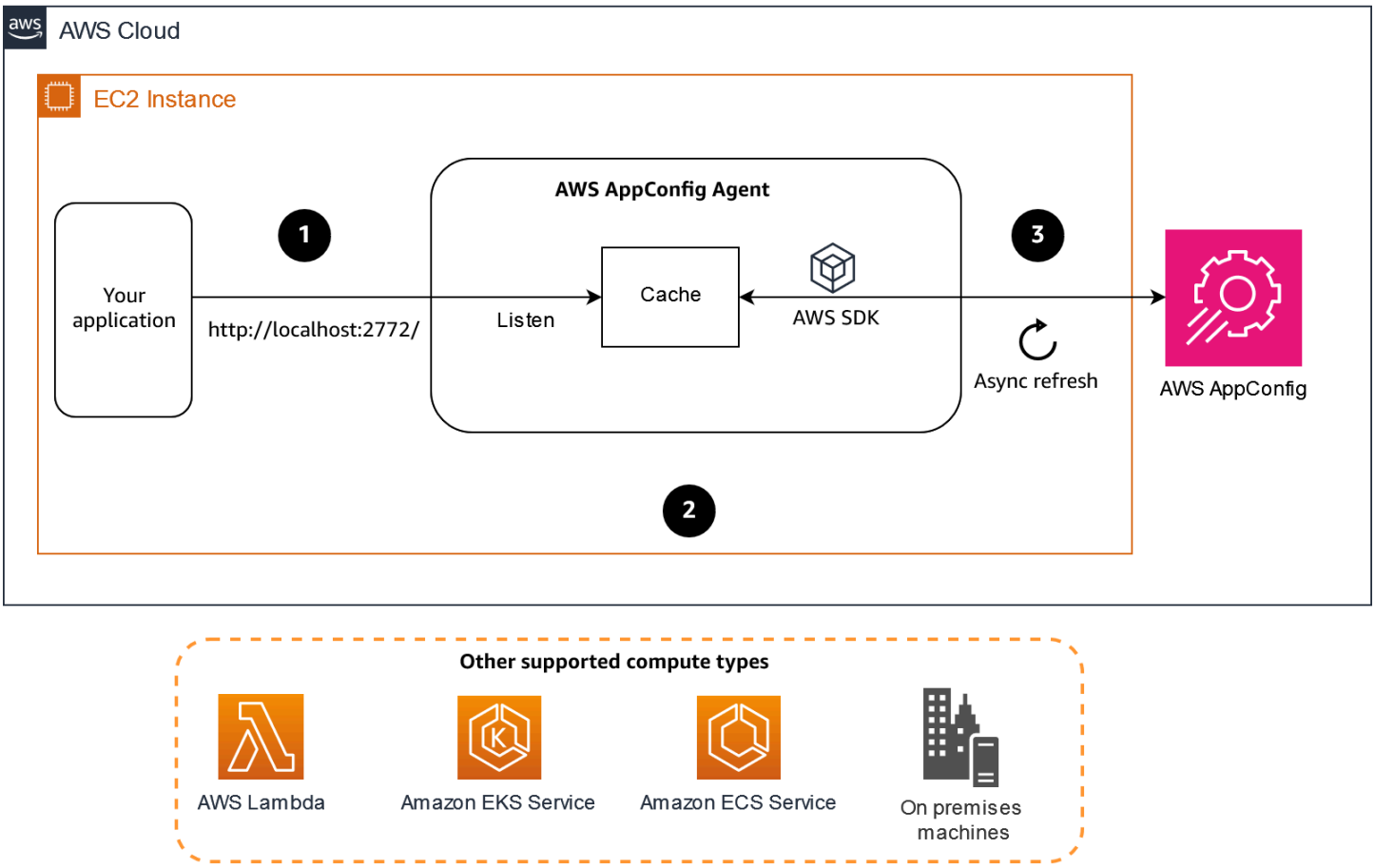
AWS AppConfig O agente é um processo desenvolvido e gerenciado pela Amazon para recuperar dados de configuração do. AWS AppConfig Com o agente, você pode armazenar dados de configuração em cache localmente e consultar de forma assíncrona o serviço de plano de AWS AppConfig dados para obter atualizações. Esse caching/polling processo garante que seus dados de configuração estejam sempre disponíveis para seu aplicativo, minimizando a latência e o custo. O agente não é a única forma de recuperar dados de configuração AWS AppConfig, mas é a forma recomendada. O agente aprimora o processamento e o gerenciamento de aplicativos das seguintes maneiras:

- O agente liga AWS AppConfig em seu nome usando um principal AWS Identity and Access Management (IAM) e gerenciando um cache local de dados de configuração. Ao recuperar dados de configuração do cache local, a aplicação exige menos atualizações de código para gerenciar dados de configuração, recupera dados de configuração em milissegundos e não é prejudicada por problemas de rede que possam afetar as chamadas para esses dados.
- O agente oferece uma experiência nativa para recuperar e resolver sinalizadores de AWS AppConfig recursos.

- Pronto para uso, o agente fornece as práticas recomendadas para estratégias de armazenamento em cache, intervalos de pesquisa e disponibilidade de dados de configuração locais, enquanto rastreia os tokens de configuração necessários para chamadas de serviço subsequentes.
- Durante a execução em segundo plano, o agente consulta periodicamente o serviço de plano de AWS AppConfig dados para atualizações de dados de configuração. Seu aplicativo pode recuperar os dados conectando-se ao localhost na porta 2772 (um valor de porta padrão personalizável) e chamando HTTP GET para recuperar os dados.

> **Note**
>
> AWS AppConfig O agente armazena os dados em cache na primeira vez que o serviço recupera seus dados de configuração. Por esse motivo, a primeira chamada para recuperar dados é mais lenta que as chamadas subsequentes.

O diagrama a seguir mostra como o AWS AppConfig Agente funciona.



1. A aplicação solicita dados de configuração ao agente.
2. O agente exibe dados de um cache em memória.
3. O agente pesquisa de forma assíncrona o AWS AppConfig serviço para obter os dados de configuração mais recentes em uma cadência predefinida. Os dados de configuração mais recentes são sempre armazenados em um cache em memória.

Como usar o AWS AppConfig Agent para recuperar dados de configuração

O AWS AppConfig Agente é o método recomendado para recuperar sinalizadores de AWS AppConfig recursos ou dados de configuração de formato livre. O agente é compatível com todas as formas de AWS computação, incluindo Amazon EC2, Amazon ECS, Amazon EKS e Lambda. Depois de concluir a configuração inicial do agente, usar o agente para recuperar dados de configuração é mais simples do que chamar diretamente. AWS AppConfig APIs O agente

implementa automaticamente as melhores práticas e pode reduzir seu custo de uso AWS AppConfig como resultado de menos chamadas de API para recuperar as configurações.

> **Note**
>
> A recuperação de dados de configuração de um arquivo separado Conta da AWS não é suportada.

Tópicos

- Usando o AWS AppConfig Agent com AWS Lambda
- Usando o AWS AppConfig Agent com o Amazon EC2 e máquinas locais
- Usando o AWS AppConfig Agent com o Amazon ECS e o Amazon EKS
- Recuperar sinalizadores de atributos básicos e multivariante
- Usar um manifesto para habilitar recursos de recuperação adicionais
  - Configurando o AWS AppConfig Agente para recuperar configurações de várias contas
  - Configurando o AWS AppConfig Agente para gravar cópias de configuração em disco
- Gerar um cliente usando a especificação OpenAPI
- Trabalhando com o modo de desenvolvimento local do AWS AppConfig agente

Usando o AWS AppConfig Agent com AWS Lambda

Uma AWS Lambda extensão é um processo complementar que aumenta os recursos de uma função Lambda. Uma extensão pode ser iniciada antes de uma função ser invocada, executada paralelamente a uma função e continuar sendo executada após o processamento de uma invocação da função. Em essência, uma extensão do Lambda é como um cliente que é executado em paralelo a uma invocação do Lambda. Esse cliente paralelo pode interagir com a função a qualquer momento durante o seu ciclo de vida.

Se você usa sinalizadores de AWS AppConfig recursos ou outros dados de configuração dinâmica em uma função Lambda, recomendamos que você adicione a extensão AWS AppConfig Agent Lambda como uma camada à sua função Lambda. Isso simplifica a chamada de sinalizadores de recursos, e a própria extensão inclui as melhores práticas que simplificam o uso e reduzem os AWS AppConfig custos. Custos reduzidos resultam de menos chamadas de API para o AWS AppConfig

serviço e menores tempos de processamento da função Lambda. Para obter mais informações sobre extensões do Lambda, consulte Extensões do Lambda no Guia do desenvolvedor do AWS Lambda .

> **Note**
>
> AWS AppConfig é uma capacidade de AWS Systems Manager. AWS AppConfig o preço é baseado no número de vezes que uma configuração é chamada e recebida. Os custos aumentarão se o Lambda realizar várias inicializações a frio e recuperar novos dados de configuração com frequência.

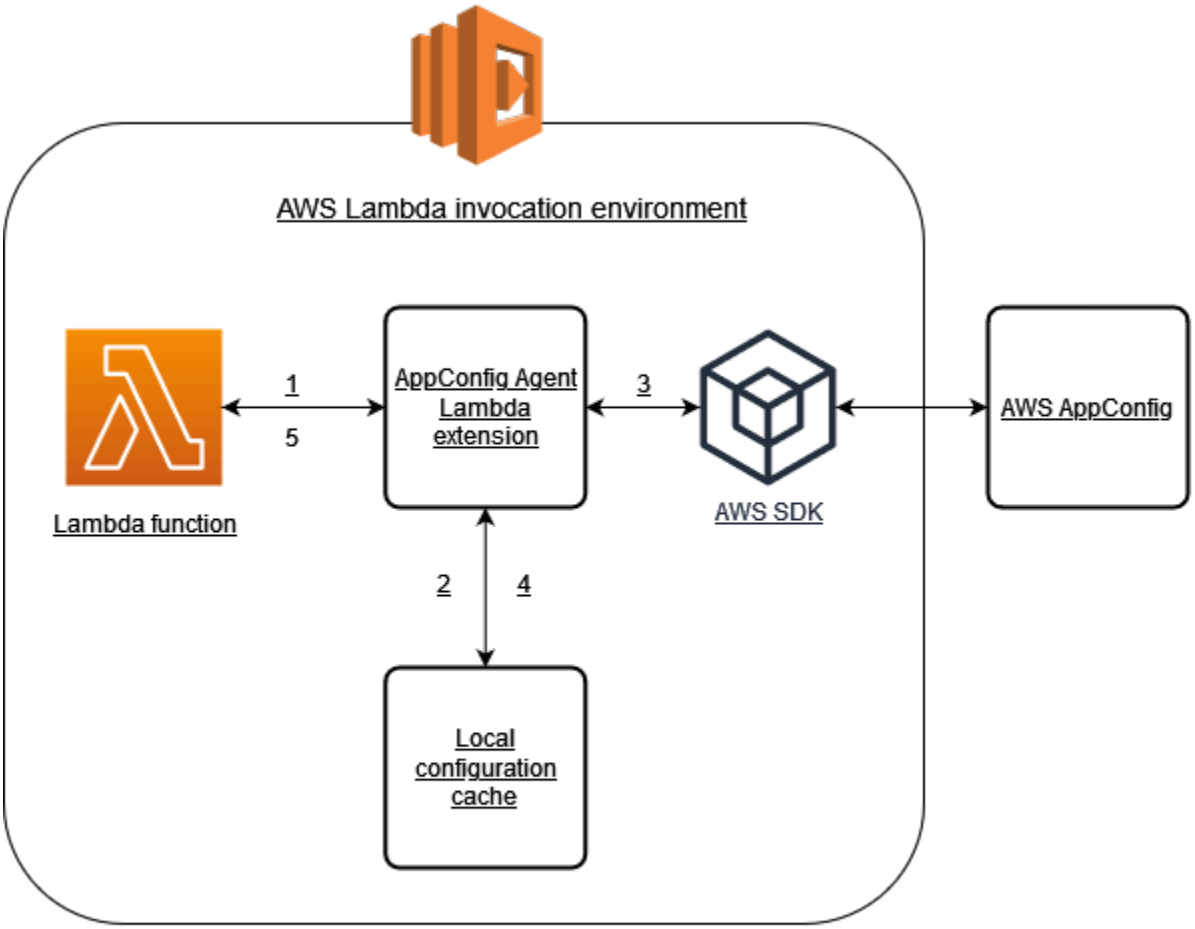
Tópicos

- Entendendo como a AWS AppConfig extensão Agent Lambda funciona
- Adicionando a extensão AWS AppConfig Agent Lambda
- Configurando a extensão do AWS AppConfig Agent Lambda
- Entendendo as versões disponíveis da extensão do AWS AppConfig Agent Lambda

Entendendo como a AWS AppConfig extensão Agent Lambda funciona

Se você usa AWS AppConfig para gerenciar configurações para uma função do Lambda sem extensões do Lambda, deve configurar sua função do Lambda para receber atualizações de configuração por meio da integração com as ações da API. StartConfigurationSessionGetLatestConfiguration

A integração da extensão do AWS AppConfig Agent Lambda com sua função Lambda simplifica esse processo. A extensão se encarrega de chamar o AWS AppConfig serviço, gerenciar um cache local de dados recuperados, rastrear os tokens de configuração necessários para as próximas chamadas de serviço e verificar periodicamente as atualizações de configuração em segundo plano. O diagrama a seguir mostra como funciona.



1. Você configura a extensão do AWS AppConfig Agent Lambda como uma camada da sua função Lambda.
2. Para acessar seus dados de configuração, sua função chama a AWS AppConfig extensão em um endpoint HTTP em `localhost:2772` execução.
3. A extensão mantém um cache local dos dados de configuração. Se os dados não estiverem no cache, a extensão chamará AWS AppConfig para obter os dados de configuração.
4. Ao receber a configuração do serviço, a extensão a armazena no cache local e a transmite para a função do Lambda.
5. AWS AppConfig A extensão Agent Lambda verifica periodicamente se há atualizações em seus dados de configuração em segundo plano. Sempre que sua função do Lambda é invocada, a extensão verifica o tempo decorrido desde que recuperou uma configuração. Se o tempo decorrido for maior que o intervalo de pesquisa configurado, a extensão liga AWS AppConfig

para verificar os dados recém-implantados, atualiza o cache local se houver alguma alteração e redefine o tempo decorrido.

> **Note**
>
> - O Lambda cria instâncias separadas correspondentes ao nível de simultaneidade requerido por sua função. Cada instância é isolada e mantém o próprio cache local dos dados de configuração. Para obter mais informações sobre instâncias e simultaneidade do Lambda, consulte Gerenciamento da simultaneidade para uma função do Lambda.
> - O tempo necessário para que uma alteração de configuração apareça em uma função Lambda, após a implantação de uma configuração atualizada AWS AppConfig, depende da estratégia de implantação usada para a implantação e do intervalo de pesquisa configurado para a extensão.

Adicionando a extensão AWS AppConfig Agent Lambda

Para usar a extensão AWS AppConfig Agent Lambda, você precisa adicionar a extensão ao seu Lambda. Isso pode ser feito adicionando a extensão do AWS AppConfig Agent Lambda à sua função Lambda como uma camada ou habilitando a extensão em uma função Lambda como uma imagem de contêiner.

> **Note**
>
> A AWS AppConfig extensão é independente do tempo de execução e oferece suporte a todos os tempos de execução.

Antes de começar

Antes de ativar a extensão do AWS AppConfig Agent Lambda, faça o seguinte:

- Organize as configurações em sua função do Lambda para que você possa externalizá-las no AWS AppConfig.
- Crie AWS AppConfig artefatos e dados de configuração, incluindo sinalizadores de recursos ou dados de configuração de formato livre. Para obter mais informações, consulte Criação de sinalizadores de recursos e dados de configuração de formato livre no AWS AppConfig.

- Adicione `appconfig:StartConfigurationSession` e `appconfig:GetLatestConfiguration` à política AWS Identity and Access Management (IAM) usada pela função de execução da função Lambda. Para obter mais informações, consulte perfil do IAM para execução do AWS Lambda no Guia do desenvolvedor do AWS Lambda . Para obter mais informações sobre permissões do AWS AppConfig , consulte Ações, recursos e chaves de condição do AWS AppConfig na Referência de autorização do serviço.

Adicionar a extensão do AWS AppConfig Agent Lambda usando uma camada e um ARN

Para usar a extensão AWS AppConfig Agent Lambda, você adiciona a extensão à sua função Lambda como uma camada. Para obter informações sobre como adicionar uma camada à sua função, consulte Configuração de extensões no Guia do desenvolvedor do AWS Lambda . O nome da extensão no AWS Lambda console é AWS- AppConfig -Extension. Observe também que, ao adicionar a extensão como uma camada ao Lambda, é necessário especificar um nome do recurso da Amazon (ARN). Escolha um ARN em uma das listas a seguir que corresponda à plataforma e Região da AWS onde você criou o Lambda.

- Plataforma x86-64
- ARM64 plataforma

Se quiser testar a extensão antes de adicioná-la à sua função, você pode verificar se ela funciona usando o exemplo de código a seguir.

```
import urllib.request


def lambda_handler(event, context):
    url = f'http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name'
    config = urllib.request.urlopen(url).read()
    return config
```

Para testá-la, crie uma nova função do Lambda para Python, adicione a extensão e execute a função do Lambda. Depois de executar a função Lambda, a função AWS AppConfig Lambda retorna a configuração especificada para o caminho http://localhost:2772. Para obter informações sobre como criar uma função do Lambda, consulte Criar uma função do Lambda com o console, no Guia do desenvolvedor do AWS Lambda .

Important

Você pode visualizar os dados de registro da extensão do AWS AppConfig Agent Lambda AWS Lambda nos registros. As entradas de log são precedidas por `appconfig agent`. Aqui está um exemplo.

```
[appconfig agent] 2024/05/07 04:19:01 ERROR retrieve failure for
 'SourceEventConfig:SourceEventConfigEnvironment:SourceEventConfigProfile':
 StartConfigurationSession: api error AccessDenied: User:
 arn:aws:sts::0123456789:assumed-role/us-east-1-LambdaRole/
extension1 is not authorized to perform: sts:AssumeRole on resource:
 arn:aws:iam::0123456789:role/test1 (retry in 60s)
```

Configurando a extensão do AWS AppConfig Agent Lambda

Você pode configurar a extensão alterando as seguintes variáveis de AWS Lambda ambiente. Para obter mais informações, consulte Usando variáveis de AWS Lambda ambiente no Guia do AWS Lambda desenvolvedor.

Pré-busca de dados de configuração

A variável de ambiente `AWS_APPCONFIG_EXTENSION_PREFETCH_LIST` pode melhorar o tempo de inicialização da sua função. Quando a extensão do AWS AppConfig Agent Lambda é inicializada, ela recupera a configuração especificada antes de o AWS AppConfig Lambda começar a inicializar sua função e invocar seu manipulador. Em alguns casos, os dados de configuração já estão disponíveis no cache local antes que sua função os solicite.

Para usar o recurso de pré-busca, defina o valor da variável de ambiente para o caminho correspondente aos seus dados de configuração. Por exemplo, se sua configuração corresponder a um aplicativo, ambiente e perfil de configuração denominados respectivamente “my_application”, “my_environment” e “my_configuration_data”, o caminho será `/applications/my_application/environments/my_environment/configurations/my_configuration_data`. Você pode especificar vários itens de configuração listando-os como uma lista separada por vírgulas (se você tiver um nome de recurso que inclua uma vírgula, use o valor do ID do recurso em vez do nome).

Acesso a dados de configuração a partir de outra conta

A extensão do AWS AppConfig Agent Lambda pode recuperar dados de configuração de outra conta especificando uma função do IAM que concede permissões aos dados. Para configurá-la, siga estas etapas:

1. Na conta em que AWS AppConfig é usada para gerenciar os dados de configuração, crie uma função com uma política de confiança que conceda à conta que executa a função Lambda acesso às `appconfig:GetLatestConfiguration` ações `appconfig:StartConfigurationSession` e, junto com a parte ou a totalidade ARNs correspondente aos recursos de AWS AppConfig configuração.
2. Na conta que executa a função do Lambda, adicione a variável de ambiente `AWS_APPCONFIG_EXTENSION_ROLE_ARN` à função do Lambda com o ARN da função criada na etapa 1.
3. (Opcional) Se necessário, um ID externo pode ser especificado usando a variável de ambiente `AWS_APPCONFIG_EXTENSION_ROLE_EXTERNAL_ID`. Da mesma forma, um nome de sessão pode ser configurado usando a variável de ambiente `AWS_APPCONFIG_EXTENSION_ROLE_SESSION_NAME`.

> **Note**
>
> Observe as seguintes informações:
>
> - A extensão do AWS AppConfig Agent Lambda só pode recuperar dados de uma conta. Se você especificar um perfil do IAM, a extensão não poderá recuperar dados de configuração da conta na qual a função do Lambda está sendo executada.
> - AWS Lambda registra informações sobre a extensão do AWS AppConfig Agent Lambda e a função Lambda usando o Amazon Logs. CloudWatch
> - A tabela a seguir inclui uma coluna de valores de amostra. Dependendo da resolução do monitor, talvez seja necessário rolar até a parte inferior da tabela e depois rolar para a direita para ver a coluna.

Variável de ambiente	Detalhes	Valor padrão	Exemplos de valores
`AWS_APPCONFIG_EXTE`	Esta variável de ambiente especifica a porta na qual é	2772	2772

Variável de ambiente	Detalhes	Valor padrão	Exemplos de valores
`NSION_HTT P_PORT`	executado o servidor HTTP local que hospeda a extensão.		
`AWS_APPCO NFIG_EXTE NSION_LOG _LEVEL`	Esta variável de ambiente especifica o nível de detalhes que o agente registra. Cada nível inclui o nível atual e todos os níveis superiore s. O valor diferenci a maiúsculas de minúsculas. Do mais detalhado ao menos detalhado , os níveis de log são: `trace`, `debug`, `info`, `warn`, `error`, `fatal` e none. O log do `trace` contém informaçõ es detalhadas, incluindo informaçõ es de tempo, sobre o agente.	info	trace depurar info warn erro fatal nenhuma
`AWS_APPCO NFIG_EXTE NSION_MAX _CONNECTIONS`	Esta variável de ambiente configura o número máximo de conexões que a extensão usa para recuperar as configurações do AWS AppConfig.	3	3

Variável de ambiente	Detalhes	Valor padrão	Exemplos de valores
`AWS_APPCONFIG_EXTENSION_POLL_INTERVAL_SECONDS`	Essa variável de ambiente controla a frequência com que o agente pesquisa dados AWS AppConfig de configuração atualizados. É possível especificar um número de segundos para o intervalo. Você também pode especificar um número com uma unidade de tempo: s para segundos, m para minutos, e h para horas. Se nenhuma unidade for especificada, o agente usará segundos como padrão. Por exemplo, 60, 60 s e 1 min resultam no mesmo intervalo de pesquisa.	45	45 45s 5 minutos 1h

Variável de ambiente	Detalhes	Valor padrão	Exemplos de valores
`AWS_APPCONFIG_EXTENSION_POLL_TIMEOUT_MILLIS`	Essa variável de ambiente controla o tempo máximo, em milissegundos, em que a extensão espera por uma resposta AWS AppConfig ao atualizar os dados no cache. Se AWS AppConfig não responder no período de tempo especificado, a extensão ignora esse intervalo de pesquisa e retorna os dados em cache atualizados anteriormente.	3000ms	3000 300ms 5s

Variável de ambiente	Detalhes	Valor padrão	Exemplos de valores
`AWS_APPCONFIG_EXTENSION_PREFETCH_LIST`	Essa variável de ambiente especifica os dados de configuração que o agente solicita AWS AppConfig assim que é iniciado. Vários identificadores de configuração podem ser disponibilizados em uma lista separada por vírgulas. A pré-busca de dados de configuração do AWS AppConfig pode reduzir significativamente o tempo de inicialização a frio da sua função.	Nenhum	MyApp:MyEnv:MyConfig abcd123:efgh456:ijkl789 MyApp::Config1, ::Config2 MyEnv MyApp MyEnv
`AWS_APPCONFIG_EXTENSION_PROXY_HEADERS`	Esta variável de ambiente especifica os cabeçalhos exigidos pelo proxy referenciado na variável de ambiente `AWS_APPCONFIG_EXTENSION_PROXY_URL` . O valor é uma lista de cabeçalhos separados por vírgula.	Nenhum	header: value h1: v1, h2: v2

Variável de ambiente	Detalhes	Valor padrão	Exemplos de valores
`AWS_APPCONFIG_EXTENSION_PROXY_URL`	Essa variável de ambiente especifica a URL do proxy a ser usada para conexões da AWS AppConfig extensão a. Serviços da AWSHTTPSe HTTP URLs são suportados.	Nenhum	http://localhost:7474 https://my-proxy.example.com
`AWS_APPCONFIG_EXTENSION_ROLE_ARN`	Essa variável de ambiente especifica o ARN da função do IAM correspondente a uma função que deve ser assumida pela extensão para AWS AppConfig recuperar a configuração.	Nenhum	arn: aws: iam: :123456789012: role/ MyRole
`AWS_APPCONFIG_EXTENSION_ROLE_EXTERNAL_ID`	Esta variável de ambiente especifica o ID externo a ser usado em conjunto com o ARN da função assumida.	Nenhum	MyExternalId
`AWS_APPCONFIG_EXTENSION_ROLE_SESSION_NAME`	Esta variável de ambiente especifica o nome da sessão a ser associado às credenciais do perfil do IAM assumido.	Nenhum	AWSAppConfigAgentSession

Variável de ambiente	Detalhes	Valor padrão	Exemplos de valores
`AWS_APPCONFIG_EXTENSION_SERVICE_REGION`	Essa variável de ambiente especifica uma região alternativa que a extensão deve usar para chamar o AWS AppConfig serviço. Quando indefinida, a extensão usa o endpoint na região atual.	Nenhum	us-east-1 eu-west-1
`AWS_APPCONFIG_EXTENSION_MANIFEST`	Essa variável de ambiente configura o AWS AppConfig Agente para aproveitar os recursos adicionais por configuração, como recuperações de várias contas e salvamento da configuração em disco. Para obter mais informações sobre esses recursos, consulte Usar um manifesto para habilitar recursos de recuperação adicionais.	Nenhum	Ao usar a AWS AppConfig configuração como manifesto :`MyApp:MyEnv:MyManifestConfig` . Ao carregar o manifesto do disco: `file:/path/to/manifest.json`

Variável de ambiente	Detalhes	Valor padrão	Exemplos de valores
`AWS_APPCONFIG_EXTENSION_WAIT_ON_MANIFEST`	Essa variável de ambiente configura o AWS AppConfig Agente para esperar até que o manifesto seja processado antes de concluir a inicialização.	true	verdadeiro false

Entendendo as versões disponíveis da extensão do AWS AppConfig Agent Lambda

Este tópico inclui informações sobre as versões da extensão AWS AppConfig do Agent Lambda. A extensão AWS AppConfig Agent Lambda oferece suporte às funções Lambda desenvolvidas para as plataformas x86-64 e (Graviton2). ARM64 Para funcionar corretamente, sua função Lambda deve ser configurada para usar o Amazon Resource Name (ARN) específico para o Região da AWS local em que está atualmente hospedada. Você pode visualizar Região da AWS os detalhes do ARN posteriormente nesta seção.

Important

Observe os seguintes detalhes importantes sobre a extensão do AWS AppConfig Agent Lambda.

- A ação `GetConfiguration` da API tornou-se obsoleta em 28 de janeiro de 2022. As chamadas para receber dados de configuração devem usar o `StartConfigurationSession` e `GetLatestConfiguration` APIs em vez disso. Se você estiver usando uma versão da extensão do AWS AppConfig Agent Lambda criada após 28 de janeiro de 2022, precisará configurar as permissões para a nova. APIs Para obter mais informações, consulte Recuperando dados de configuração sem AWS AppConfig o Agente.
- AWS AppConfig suporta todas as versões listadas emVersões de extensão mais antigas. É recomendável atualizar periodicamente para a versão mais recente para aproveitar os aprimoramentos das extensões.

Tópicos

- AWS AppConfig Notas de lançamento da extensão Agent Lambda
- Como encontrar o número da versão da sua extensão do Lambda
- Plataforma x86-64
- ARM64 plataforma
- Versões de extensão mais antigas

AWS AppConfig Notas de lançamento da extensão Agent Lambda

A tabela a seguir descreve as alterações feitas nas versões recentes da extensão AWS AppConfig Lambda.

Versão	Data de lançamento	Observações
2.0.11962.0	20/02/2026	Suporte aprimorado ao ambiente, pequenos aprimoramentos e correções de erros.
2.0.8693	20/11/2025	Suporte aprimorado ao ambiente, pequenos aprimoramentos e correções de erros. Foi adicionado suporte para o seguinte Regiões da AWS • Ásia-Pacífico (Taipei), ap-east-2 • Ásia-Pacífico (Nova Zelândia), ap-southeast-6 • Ásia-Pacífico (Tailândia), ap-southeast-7 • México (Centro), mx-central-1

Versão	Data de lançamento	Observações
2.0.2037	05/12/2025	Caminho `/ping` adicionado, que expõe uma verificação de integridade simples que retorna a versão desse agente. Também inclui pequenos aprimoramentos e correções de bugs.
2.0.1079	12/12/2024	Correções e aprimoramentos secundários.
2.0.719	08/08/2024	Correções e aprimoramentos secundários.
2.0.678	23/07/2024	Aprimoramentos para comportar destinos, variantes e divisões de sinalizadores de atributos. Para obter mais informações, consulte Criar sinalizadores de atributos multivariante.
2.0.501	07/01/2024	Correções e aprimoramentos secundários.

Versão	Data de lançamento	Observações
2.0.358	12/01/2023	Foi adicionado suporte para os seguintes recursos de recuperação: • Recuperação de várias contas: use o AWS AppConfig Agente de uma conta primária ou de recuperação Conta da AWS para recuperar dados de configuração de várias contas de fornecedores. • Gravar cópia da configuração no disco: use o AWS AppConfig Agente para gravar dados de configuração no disco. Esse recurso permite que os clientes com aplicações que leem dados de configuração do disco se integrem ao AWS AppConfig.
2.0.181	14/08/2023	Foi adicionado suporte para o Região da AWS il-central-1 de Israel (Tel Aviv).

Versão	Data de lançamento	Observações
2.0.165	21/02/2023	Correções de erros secundárias. Não restringe mais o uso da extensão a versões específicas de tempo de execução por meio do AWS Lambda console. Foi adicionado suporte para o seguinte Regiões da AWS: • Oriente Médio (EAU) me-central-1 • Ásia-Pacífico (Hyderabad) ap-south-2 • Ásia-Pacífico (Melbourne) ap-southeast-4 • Europa (Espanha) eu-south-2 • Europa (Zurique) eu-central-2

Versão	Data de lançamento	Observações
2.0.122	23/08/2022	Foi adicionado suporte para um proxy de tunelamento, que pode ser configurado com as variáveis de ambiente `AWS_APPCONFIG_EXTENSION_PROXY_URL` e `AWS_APPCONFIG_EXTENSION_PROXY_HEADERS`. Foi adicionado o .NET 6 como runtime. Para obter mais informações sobre variáveis de ambiente, consulte Configurando a extensão do AWS AppConfig Agent Lambda.
2.0.58	05/03/2022	Suporte aprimorado para processadores Graviton2 (ARM64) no Lambda.

Versão	Data de lançamento	Observações
2.0.45	15/03/2022	Foi adicionado suporte para chamar um único sinalizador de atributos. Anteriormente, os clientes chamavam sinalizadores de atributos agrupados em um perfil de configuração e precisavam analisar a resposta do lado do cliente. Com essa versão, os clientes podem usar um parâmetro `flag=<flag-name>` ao chamar o endpoint de localhost HTTP para obter o valor de um único sinalizador. Também foi adicionado suporte inicial para processadores Graviton2 (ARM64).

Como encontrar o número da versão da sua extensão do Lambda

Use o procedimento a seguir para localizar o número da versão da sua extensão do AWS AppConfig Agente Lambda atualmente configurada. Para funcionar corretamente, sua função Lambda deve ser configurada para usar o Amazon Resource Name (ARN) específico para o Região da AWS local em que está atualmente hospedada.

1. Faça login no Console de gerenciamento da AWS e abra o AWS Lambda console em https://console.aws.amazon.com/lambda/.
2. Escolha a função do Lambda na qual deseja adicionar a camada `AWS-AppConfig-Extension`.
3. Na seção Camadas, escolha Adicionar uma camada.
4. Na seção Escolher uma camada, escolha AWS- AppConfig -Extensão na lista de AWS camadas.
5. Use a lista Versão para escolher um número de versão.
6. Escolha Adicionar.

7. Use a guia Testar para testar a função.
8. Depois que o teste for concluído, visualize a saída de logs. Localize a versão da extensão do AWS AppConfig Agent Lambda na seção Detalhes da execução. Essa versão deve corresponder à exigida URLs para essa versão.

Plataforma x86-64

Ao adicionar a extensão como uma camada ao seu Lambda, você deve especificar um ARN. Escolha um ARN na tabela a seguir que corresponda ao local em Região da AWS que você criou o Lambda. Elas ARNs são para funções Lambda desenvolvidas para a plataforma x86-64.

Versão 2.0.11962.0

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:296`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:252`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:359`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:348`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:239`
Oeste do Canadá (Calgary)	`arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension:147`

Região	ARN
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:270`
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:195`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:278`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:217`
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:248`
Europa (Estocolmo)	`arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:342`
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:226`
Europa (Espanha)	`arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:189`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:219`

Região	ARN
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:221`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:228`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:245`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:248`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:247`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:233`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:288`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:231`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:163`

Região	ARN
Ásia-Pacífico (Malásia)	arn:aws:lambda:ap-southeast-5:631746059939:layer:AWS-AppConfig-Extension:136
Ásia-Pacífico (Mumbai)	arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:264
Ásia-Pacífico (Hyderabad)	arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:192
Ásia-Pacífico (Nova Zelândia)	arn:aws:lambda:ap-southeast-6:381491832265:layer:AWS-AppConfig-Extension:58
Ásia-Pacífico (Tailândia)	arn:aws:lambda:ap-southeast-7:851725616657:layer:AWS-AppConfig-Extension:109
Ásia-Pacífico (Taipei)	arn:aws:lambda:ap-east-2:730335625313:layer:AWS-AppConfig-Extension:118
América do Sul (São Paulo)	arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:302
México (Centro)	arn:aws:lambda:mx-central-1:891376990304:layer:AWS-AppConfig-Extension:115
África (Cidade do Cabo)	arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:234

Região	ARN
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension:168`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:206`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:244`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:184`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:182`

ARM64 plataforma

Ao adicionar a extensão como uma camada ao seu Lambda, você deve especificar um ARN. Escolha um ARN na tabela a seguir que corresponda ao local em Região da AWS que você criou o Lambda. Elas ARNs são para funções Lambda desenvolvidas para a ARM64 plataforma.

Versão 2.0.11962.0

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:229`

Região	ARN
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:204`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension-Arm64:236`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:250`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension-Arm64:159`
Oeste do Canadá (Calgary)	`arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension-Arm64:137`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:213`
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension-Arm64:153`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:216`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:169`

Região	ARN
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension-Arm64:167`
Europa (Estocolmo)	`arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension-Arm64:201`
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension-Arm64:154`
Europa (Espanha)	`arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension-Arm64:150`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension-Arm64:156`
Ásia-Pacífico (Taipei)	`arn:aws:lambda:ap-east-2:730335625313:layer:AWS-AppConfig-Extension-Arm64:92`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:198`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension-Arm64:156`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension-Arm64:162`

Região	ARN
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:185`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:231`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension-Arm64:168`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension-Arm64:148`
Ásia-Pacífico (Malásia)	`arn:aws:lambda:ap-southeast-5:631746059939:layer:AWS-AppConfig-Extension-Arm64:111`
Ásia-Pacífico (Nova Zelândia)	`arn:aws:lambda:ap-southeast-6:381491832265:layer:AWS-AppConfig-Extension-Arm64:48`
Ásia-Pacífico (Tailândia)	`arn:aws:lambda:ap-southeast-7:851725616657:layer:AWS-AppConfig-Extension-Arm64:108`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:206`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension-Arm64:150`

Região	ARN
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension-Arm64:190`
México (Centro)	`arn:aws:lambda:mx-central-1:891376990304:layer:AWS-AppConfig-Extension-Arm64:114`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension-Arm64:162`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension-Arm64:162`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension-Arm64:172`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension-Arm64:151`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension-Arm64:141`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension-Arm64:143`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension-Arm64:130`

Região	ARN
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension-Arm64:128`

Versões de extensão mais antigas

Esta seção lista as ARNs e Regiões da AWS para versões mais antigas da extensão AWS AppConfig Lambda. Esta lista não contém informações de todas as versões anteriores da extensão do Lambda do AWS AppConfig Agent, mas será atualizada quando novas versões forem lançadas.

Tópicos

- Versões de extensão mais antigas (plataforma x86-64)
- Versões de extensão mais antigas (ARM64 plataforma)

Versões de extensão mais antigas (plataforma x86-64)

As tabelas a seguir listam ARNs e as Regiões da AWS versões mais antigas da extensão AWS AppConfig Agent Lambda desenvolvida para a plataforma x86-64.

Data substituída pela nova extensão: 17/02/2026

Versão 2.0.8693

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:279`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:235`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:348`

Região	ARN
Oeste dos EUA (Oregon)	arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:335
Canadá (Central)	arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:228
Oeste do Canadá (Calgary)	arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension:130
Europa (Frankfurt)	arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:261
Europa (Zurique)	arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:178
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:261
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:207
Europa (Paris)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:235
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:333

Região	ARN
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:215`
Europa (Espanha)	`arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:176`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:205`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:203`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:217`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:228`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:239`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:234`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:224`

Região	ARN
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:272`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:222`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:152`
Ásia-Pacífico (Malásia)	`arn:aws:lambda:ap-southeast-5:631746059939:layer:AWS-AppConfig-Extension:127`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:248`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:179`
Ásia-Pacífico (Nova Zelândia)	`arn:aws:lambda:ap-southeast-6:381491832265:layer:AWS-AppConfig-Extension:41`
Ásia-Pacífico (Tailândia)	`arn:aws:lambda:ap-southeast-7:851725616657:layer:AWS-AppConfig-Extension:98`
Ásia-Pacífico (Taipei)	`arn:aws:lambda:ap-east-2:730335625313:layer:AWS-AppConfig-Extension:100`

Região	ARN
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:288`
México (Centro)	`arn:aws:lambda:mx-central-1:891376990304:layer:AWS-AppConfig-Extension:98`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:225`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension:155`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:195`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:227`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:184`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:182`

Data substituída pela nova extensão: 20/11/2025

Versão 2.0.2037

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:207`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:162`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:258`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:262`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:152`
Oeste do Canadá (Calgary)	`arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension:57`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:189`
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:106`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:189`

Região	ARN
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:133
Europa (Paris)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:162
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:259
Europa (Milão)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:140
Europa (Espanha)	arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:102
China (Pequim)	arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:133
China (Ningxia)	arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:131
Ásia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:142
Ásia-Pacífico (Tóquio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:155

Região	ARN
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:165`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:159`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:156`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:199`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:150`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:78`
Ásia-Pacífico (Malásia)	`arn:aws:lambda:ap-southeast-5:631746059939:layer:AWS-AppConfig-Extension:55`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:175`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:104`

Região	ARN
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:215`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:152`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension:81`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:120`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:154`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:110`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:109`

Data da substituição pela nova extensão: 20/05/2025

Versão 2.0.1079

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:174`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:133`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:223`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:230`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:123`
Oeste do Canadá (Calgary)	`arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension:27`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:159`
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:77`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:160`

Região	ARN
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:121`
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:133`
Europa (Estocolmo)	`arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:225`
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:111`
Europa (Espanha)	`arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:74`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:106`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:104`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:113`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:126`

Região	ARN
Ásia-Pacífico (Seul)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:136
Asia Pacific (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:130
Ásia-Pacífico (Singapura)	arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:134
Ásia-Pacífico (Sydney)	arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:165
Ásia-Pacífico (Jacarta)	arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:121
Ásia-Pacífico (Melbourne)	arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:49
Ásia-Pacífico (Malásia)	arn:aws:lambda:ap-southeast-5:631746059939:layer:AWS-AppConfig-Extension:26
Ásia-Pacífico (Mumbai)	arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:146
Ásia-Pacífico (Hyderabad)	arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:75

Região	ARN
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:179`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:123`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension:52`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:91`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:125`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:80`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:80`

Data da substituição pela nova extensão: 12/12/2024

Versão 2.0.719

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:173`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:132`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:221`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:229`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:121`
Oeste do Canadá (Calgary)	`arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension:27`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:158`
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:75`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:159`

Região	ARN
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:120
Europa (Paris)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:132
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:224
Europa (Milão)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:110
Europa (Espanha)	arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:72
China (Pequim)	arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:104
China (Ningxia)	arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:102
Ásia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:112
Ásia-Pacífico (Tóquio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:125

Região	ARN
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:135`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:129`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:132`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:164`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:120`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:48`
Ásia-Pacífico (Malásia)	`arn:aws:lambda:ap-southeast-5:631746059939:layer:AWS-AppConfig-Extension:25`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:145`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:74`

Região	ARN
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:178`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:122`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension:50`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:90`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:124`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:79`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:79`

Data da substituição pela nova extensão: 08/08/2024

Versão 2.0.678

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:167`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:126`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:213`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:223`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:116`
Oeste do Canadá (Calgary)	`arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension:21`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:152`
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:70`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:153`

Região	ARN
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:114
Europa (Paris)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:126
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:218
Europa (Milão)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:104
Europa (Espanha)	arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:67
China (Pequim)	arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:99
China (Ningxia)	arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:97
Ásia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:106
Ásia-Pacífico (Tóquio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:119

Região	ARN
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:129`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:123`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:127`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:158`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:114`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:42`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:139`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:68`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:172`

Região	ARN
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:116`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension:45`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:84`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:118`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:73`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:73`

Data da substituição pela nova extensão: 23/07/2024

Versão 2.0.501

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:153`

Região	ARN
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:112`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:195`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:210`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:101`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:136`
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:53`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:144`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:99`
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:111`

Região	ARN
Europa (Estocolmo)	`arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:201`
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:89`
Europa (Espanha)	`arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:50`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:85`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:83`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:91`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:104`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:114`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:107`

Região	ARN
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:112`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:142`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:98`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:26`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:125`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:53`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:155`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:102`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension:28`

Região	ARN
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:68`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:103`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:59`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:59`

Data da substituição pela nova extensão: 01/07/2024

Versão 2.0.358

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:128`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:93`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:141`

Região	ARN
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:161`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:93`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:106`
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:47`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:125`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:93`
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:98`
Europa (Estocolmo)	`arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:159`
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:83`

Região	ARN
Europa (Espanha)	`arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:44`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:76`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:76`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:83`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:98`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:108`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:101`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:106`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:106`

Região	ARN
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:79`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:20`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:107`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:47`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:128`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:83`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension:22`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:49`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:85`

Região	ARN
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:54`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:54`

Data da substituição pela nova extensão: 01/12/2023

Versão 2.0.181

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:113`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:81`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:124`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:146`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:81`

Região	ARN
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:93`
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:32`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:110`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:81`
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:82`
Europa (Estocolmo)	`arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:142`
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:73`
Europa (Espanha)	`arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:29`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:68`

Região	ARN
China (Ningxia)	arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:68
Ásia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:73
Ásia-Pacífico (Tóquio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:84
Ásia-Pacífico (Seul)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:93
Asia Pacific (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:86
Ásia-Pacífico (Singapura)	arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:91
Ásia-Pacífico (Sydney)	arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:93
Ásia-Pacífico (Jacarta)	arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:64
Ásia-Pacífico (Melbourne)	arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:5

Região	ARN
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:94`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:32`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:113`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:73`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension:7`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:34`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:73`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:46`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:46`

Data da substituição pela nova extensão: 14/08/2023

Versão 2.0.165

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:110`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:79`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:121`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:143`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:79`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:91`
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension:29`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:108`

Região	ARN
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:79`
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:80`
Europa (Estocolmo)	`arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:139`
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:71`
Europa (Espanha)	`arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension:26`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:66`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:66`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:71`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:82`

Região	ARN
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:91`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:84`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:89`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:91`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:60`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension:2`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:92`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension:29`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:110`

Região	ARN
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:71`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension:31`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:71`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:44`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:44`

Data da substituição pela nova extensão: 21/02/2023

Versão 2.0.122

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:82`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:59`

Região	ARN
Oeste dos EUA (N. da Califórnia)	arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:93
Oeste dos EUA (Oregon)	arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:114
Canadá (Central)	arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:59
Europa (Frankfurt)	arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:70
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:82
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:59
Europa (Paris)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:60
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:111
Europa (Milão)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:54

Região	ARN
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:52`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:52`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:54`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:62`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:70`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:59`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:64`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:70`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:37`

Região	ARN
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:71`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:82`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:54`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:54`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:29`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:29`

Data da substituição pela nova extensão: 23/08/2022

Versão 2.0.58

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:69`

Região	ARN
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:50`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:78`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:101`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:50`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:59`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:69`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:50`
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:51`
Europa (Estocolmo)	`arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:98`

Região	ARN
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:47`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:46`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:46`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:47`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:49`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:59`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:46`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:51`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:59`

Região	ARN
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:24`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:60`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:69`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:47`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:47`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:23`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:23`

Data da substituição pela nova extensão: 21/04/2022

Versão 2.0.45

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:68`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:49`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:77`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:100`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:49`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:58`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:68`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:49`
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:50`

Região	ARN
Europa (Estocolmo)	`arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:97`
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:46`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:45`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:45`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:46`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:48`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:58`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:45`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:50`

Região	ARN
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:58`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:23`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:59`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:68`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:46`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:46`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:22`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:22`

Data da substituição pela nova extensão: 15/03/2022

Versão 2.0.30

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension:61`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension:47`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension:61`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension:89`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension:47`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension:54`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension:59`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension:47`
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension:48`

Região	ARN
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension:86
Europa (Milão)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension:44
China (Pequim)	arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension:43
China (Ningxia)	arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension:43
Ásia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension:44
Ásia-Pacífico (Tóquio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension:45
Ásia-Pacífico (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension:42
Ásia-Pacífico (Seul)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension:54
Ásia-Pacífico (Singapura)	arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension:45

Região	ARN
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension:54`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension:13`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension:55`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension:61`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension:44`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension:44`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension:20`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension:20`

Versões de extensão mais antigas (ARM64 plataforma)

As tabelas a seguir listam ARNs e Regiões da AWS as versões mais antigas da extensão AWS AppConfig Agent Lambda desenvolvida para a ARM64 plataforma.

Data substituída pela nova extensão: 17/02/2026

Versão 2.0.8693

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:212`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:187`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension-Arm64:225`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:237`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension-Arm64:148`
Oeste do Canadá (Calgary)	`arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension-Arm64:120`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:204`
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension-Arm64:136`

Região	ARN
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:199
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:159
Europa (Paris)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension-Arm64:154
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension-Arm64:192
Europa (Milão)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension-Arm64:143
Europa (Espanha)	arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension-Arm64:137
Ásia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension-Arm64:145
Ásia-Pacífico (Taipei)	arn:aws:lambda:ap-east-2:730335625313:layer:AWS-AppConfig-Extension-Arm64:74
Ásia-Pacífico (Tóquio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:181

Região	ARN
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension-Arm64:147`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension-Arm64:149`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:176`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:215`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension-Arm64:159`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension-Arm64:137`
Ásia-Pacífico (Malásia)	`arn:aws:lambda:ap-southeast-5:631746059939:layer:AWS-AppConfig-Extension-Arm64:102`
Ásia-Pacífico (Nova Zelândia)	`arn:aws:lambda:ap-southeast-6:381491832265:layer:AWS-AppConfig-Extension-Arm64:31`
Ásia-Pacífico (Tailândia)	`arn:aws:lambda:ap-southeast-7:851725616657:layer:AWS-AppConfig-Extension-Arm64:97`

Região	ARN
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:190`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension-Arm64:137`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension-Arm64:176`
México (Centro)	`arn:aws:lambda:mx-central-1:891376990304:layer:AWS-AppConfig-Extension-Arm64:97`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension-Arm64:153`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension-Arm64:151`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension-Arm64:155`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension-Arm64:138`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension-Arm64:127`

Região	ARN
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension-Arm64:125`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension-Arm64:130`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension-Arm64:128`

Data substituída pela nova extensão: 20/11/2025

Versão 2.0.2037

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:140`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:114`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension-Arm64:135`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:164`

Região	ARN
Canadá (Central)	arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension-Arm64:72
Oeste do Canadá (Calgary)	arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension-Arm64:47
Europa (Frankfurt)	arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:132
Europa (Zurique)	arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension-Arm64:64
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:127
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:85
Europa (Paris)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension-Arm64:81
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension-Arm64:118
Europa (Milão)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension-Arm64:68

Região	ARN
Europa (Espanha)	`arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension-Arm64:63`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension-Arm64:70`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:108`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension-Arm64:73`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension-Arm64:74`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:108`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:142`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension-Arm64:87`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension-Arm64:63`

Região	ARN
Ásia-Pacífico (Malásia)	`arn:aws:lambda:ap-southeast-5:631746059939:layer:AWS-AppConfig-Extension-Arm64:30`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:117`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension-Arm64:62`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension-Arm64:103`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension-Arm64:80`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension-Arm64:76`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension-Arm64:82`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension-Arm64:64`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension-Arm64:55`

Região	ARN
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension-Arm64:53`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension-Arm64:56`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension-Arm64:55`

Data da substituição pela nova extensão: 20/05/2025

Versão 2.0.1079

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:107`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:85`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension-Arm64:100`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:132`

Região	ARN
Canadá (Central)	arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension-Arm64:43
Oeste do Canadá (Calgary)	arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension-Arm64:18
Europa (Frankfurt)	arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:102
Europa (Zurique)	arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension-Arm64:35
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:98
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:73
Europa (Paris)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension-Arm64:52
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension-Arm64:84
Europa (Milão)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension-Arm64:39

Região	ARN
Europa (Espanha)	arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension-Arm64:35
Ásia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension-Arm64:41
Ásia-Pacífico (Tóquio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:79
Ásia-Pacífico (Seul)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension-Arm64:44
Asia Pacific (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension-Arm64:45
Ásia-Pacífico (Singapura)	arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:86
Ásia-Pacífico (Sydney)	arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:108
Ásia-Pacífico (Jacarta)	arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension-Arm64:58
Ásia-Pacífico (Melbourne)	arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension-Arm64:34

Região	ARN
Ásia-Pacífico (Malásia)	`arn:aws:lambda:ap-southeast-5:631746059939:layer:AWS-AppConfig-Extension-Arm64:1`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:88`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension-Arm64:33`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension-Arm64:67`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension-Arm64:51`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension-Arm64:47`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension-Arm64:53`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension-Arm64:35`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension-Arm64:28`

Região	ARN
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension-Arm64:26`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension-Arm64:26`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension-Arm64:26`

Data da substituição pela nova extensão: 12/12/2024

Versão 2.0.678

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:106`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:84`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension-Arm64:98`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:131`

Região	ARN
Canadá (Central)	arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension-Arm64:41
Oeste do Canadá (Calgary)	arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension-Arm64:17
Europa (Frankfurt)	arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:101
Europa (Zurique)	arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension-Arm64:33
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:97
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:72
Europa (Paris)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension-Arm64:51
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension-Arm64:83
Europa (Milão)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension-Arm64:38

Região	ARN
Europa (Espanha)	`arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension-Arm64:33`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension-Arm64:40`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:78`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension-Arm64:43`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension-Arm64:44`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:84`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:107`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension-Arm64:57`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension-Arm64:33`

Região	ARN
Ásia-Pacífico (Malásia)	`arn:aws:lambda:ap-southeast-5:631746059939:layer:AWS-AppConfig-Extension-Arm64:1`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:87`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension-Arm64:32`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension-Arm64:66`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension-Arm64:50`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension-Arm64:46`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension-Arm64:52`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension-Arm64:33`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension-Arm64:26`

Região	ARN
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension-Arm64:24`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension-Arm64:25`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension-Arm64:25`

Data da substituição pela nova extensão: 08/08/2024

Versão 2.0.678

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:100`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:78`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension-Arm64:90`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:125`

Região	ARN
Oeste do Canadá (Calgary)	arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension:11
Canadá (Central)	arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension-Arm64:36
Europa (Frankfurt)	arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:95
Europa (Zurique)	arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension-Arm64:28
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:91
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:66
Europa (Paris)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension-Arm64:45
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension-Arm64:77
Europa (Milão)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension-Arm64:32

Região	ARN
Europa (Espanha)	arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension-Arm64:28
Ásia-Pacífico (Hong Kong)	arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension-Arm64:34
Ásia-Pacífico (Tóquio)	arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:72
Ásia-Pacífico (Seul)	arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension-Arm64:37
Asia Pacific (Osaka)	arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension-Arm64:38
Ásia-Pacífico (Singapura)	arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:79
Ásia-Pacífico (Sydney)	arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:101
Ásia-Pacífico (Jacarta)	arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension-Arm64:51
Ásia-Pacífico (Melbourne)	arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension-Arm64:27

Região	ARN
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:81`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension-Arm64:26`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension-Arm64:60`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension-Arm64:44`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension-Arm64:40`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension-Arm64:46`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension-Arm64:28`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension-Arm64:21`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension-Arm64:19`

Região	ARN
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension-Arm64:19`
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension-Arm64:19`

Data da substituição pela nova extensão: 23/07/2024

Versão 2.0.501

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:86`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:64`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension-Arm64:72`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:112`
Oeste do Canadá (Calgary)	`arn:aws:lambda:ca-west-1:436199621743:layer:AWS-AppConfig-Extension:1`

Região	ARN
Canadá (Central)	arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension-Arm64:21
Europa (Frankfurt)	arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:79
Europa (Zurique)	arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension-Arm64:11
Europa (Irlanda)	arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:82
Europa (Londres)	arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:51
Europa (Paris)	arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension-Arm64:30
Europa (Estocolmo)	arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension-Arm64:60
Europa (Milão)	arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension-Arm64:17
Europa (Espanha)	arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension-Arm64:11

Região	ARN
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension-Arm64:19`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:57`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension-Arm64:22`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension-Arm64:22`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:64`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:85`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension-Arm64:35`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension-Arm64:11`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:67`

Região	ARN
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension-Arm64:11`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension-Arm64:43`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension-Arm64:30`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension-Arm64:24`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension-Arm64:31`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension-Arm64:11`
China (Pequim)	`arn:aws-cn:lambda:cn-north-1:615057806174:layer:AWS-AppConfig-Extension-Arm64:7`
China (Ningxia)	`arn:aws-cn:lambda:cn-northwest-1:615084187847:layer:AWS-AppConfig-Extension-Arm64:5`
AWS GovCloud (Leste dos EUA)	`arn:aws-us-gov:lambda:us-gov-east-1:946561847325:layer:AWS-AppConfig-Extension-Arm64:5`

Região	ARN
AWS GovCloud (Oeste dos EUA)	`arn:aws-us-gov:lambda:us-gov-west-1:946746059096:layer:AWS-AppConfig-Extension-Arm64:5`

Data da substituição pela nova extensão: 01/07/2024

Versão 2.0.358

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:61`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:45`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension-Arm64:18`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:63`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension-Arm64:13`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:49`

Região	ARN
Europa (Zurique)	`arn:aws:lambda:eu-central-2:758369105281:layer:AWS-AppConfig-Extension-Arm64:5`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:63`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:45`
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension-Arm64:17`
Europa (Estocolmo)	`arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension-Arm64:18`
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension-Arm64:11`
Europa (Espanha)	`arn:aws:lambda:eu-south-2:586093569114:layer:AWS-AppConfig-Extension-Arm64:5`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension-Arm64:11`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:51`

Região	ARN
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension-Arm64:16`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension-Arm64:16`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:58`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:49`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension-Arm64:16`
Ásia-Pacífico (Melbourne)	`arn:aws:lambda:ap-southeast-4:307021474294:layer:AWS-AppConfig-Extension-Arm64:5`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:49`
Ásia-Pacífico (Hyderabad)	`arn:aws:lambda:ap-south-2:489524808438:layer:AWS-AppConfig-Extension-Arm64:5`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension-Arm64:16`

Região	ARN
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension-Arm64:11`
Oriente Médio (Emirados Árabes Unidos)	`arn:aws:lambda:me-central-1:662846165436:layer:AWS-AppConfig-Extension-Arm64:5`
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension-Arm64:13`
Israel (Tel Aviv)	`arn:aws:lambda:il-central-1:895787185223:layer:AWS-AppConfig-Extension-Arm64:5`

Data da substituição pela nova extensão: 01/12/2023

Versão 2.0.181

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:46`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:33`
Oeste dos EUA (N. da Califórnia)	`arn:aws:lambda:us-west-1:958113053741:layer:AWS-AppConfig-Extension-Arm64:1`

Região	ARN
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:48`
Canadá (Central)	`arn:aws:lambda:ca-central-1:039592058896:layer:AWS-AppConfig-Extension-Arm64:1`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:36`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:48`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:33`
Europa (Paris)	`arn:aws:lambda:eu-west-3:493207061005:layer:AWS-AppConfig-Extension-Arm64:1`
Europa (Estocolmo)	`arn:aws:lambda:eu-north-1:646970417810:layer:AWS-AppConfig-Extension-Arm64:1`
Europa (Milão)	`arn:aws:lambda:eu-south-1:203683718741:layer:AWS-AppConfig-Extension-Arm64:1`
Ásia-Pacífico (Hong Kong)	`arn:aws:lambda:ap-east-1:630222743974:layer:AWS-AppConfig-Extension-Arm64:1`

Região	ARN
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:37`
Ásia-Pacífico (Seul)	`arn:aws:lambda:ap-northeast-2:826293736237:layer:AWS-AppConfig-Extension-Arm64:1`
Asia Pacific (Osaka)	`arn:aws:lambda:ap-northeast-3:706869817123:layer:AWS-AppConfig-Extension-Arm64:1`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:43`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:36`
Ásia-Pacífico (Jacarta)	`arn:aws:lambda:ap-southeast-3:418787028745:layer:AWS-AppConfig-Extension-Arm64:1`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:36`
América do Sul (São Paulo)	`arn:aws:lambda:sa-east-1:000010852771:layer:AWS-AppConfig-Extension-Arm64:1`
África (Cidade do Cabo)	`arn:aws:lambda:af-south-1:574348263942:layer:AWS-AppConfig-Extension-Arm64:1`

Região	ARN
Oriente Médio (Bahrein)	`arn:aws:lambda:me-south-1:559955524753:layer:AWS-AppConfig-Extension-Arm64:1`

Data da substituição pela nova extensão: 30/03/2023

Versão 2.0.165

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:43`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:31`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:45`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:34`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:46`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:31`

Região	ARN
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:35`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:41`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:34`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:34`

Data da substituição pela nova extensão: 21/02/2023

Versão 2.0.122

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:15`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:11`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:16`

Região	ARN
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:13`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:20`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:11`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:15`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:16`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:13`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:13`

Data da substituição pela nova extensão: 23/08/2022

Versão 2.0.58

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:2`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:2`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:3`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:2`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:7`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:2`
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:2`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:3`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:2`

Região	ARN
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:2`

Data da substituição pela nova extensão: 21/04/2022

Versão 2.0.45

Região	ARN
Leste dos EUA (Norte da Virgínia)	`arn:aws:lambda:us-east-1:027255383542:layer:AWS-AppConfig-Extension-Arm64:1`
Leste dos EUA (Ohio)	`arn:aws:lambda:us-east-2:728743619870:layer:AWS-AppConfig-Extension-Arm64:1`
Oeste dos EUA (Oregon)	`arn:aws:lambda:us-west-2:359756378197:layer:AWS-AppConfig-Extension-Arm64:2`
Europa (Frankfurt)	`arn:aws:lambda:eu-central-1:066940009817:layer:AWS-AppConfig-Extension-Arm64:1`
Europa (Irlanda)	`arn:aws:lambda:eu-west-1:434848589818:layer:AWS-AppConfig-Extension-Arm64:6`
Europa (Londres)	`arn:aws:lambda:eu-west-2:282860088358:layer:AWS-AppConfig-Extension-Arm64:1`

Região	ARN
Ásia-Pacífico (Tóquio)	`arn:aws:lambda:ap-northeast-1:980059726660:layer:AWS-AppConfig-Extension-Arm64:1`
Ásia-Pacífico (Singapura)	`arn:aws:lambda:ap-southeast-1:421114256042:layer:AWS-AppConfig-Extension-Arm64:2`
Ásia-Pacífico (Sydney)	`arn:aws:lambda:ap-southeast-2:080788657173:layer:AWS-AppConfig-Extension-Arm64:1`
Ásia-Pacífico (Mumbai)	`arn:aws:lambda:ap-south-1:554480029851:layer:AWS-AppConfig-Extension-Arm64:1`

Usando o AWS AppConfig Agent com o Amazon EC2 e máquinas locais

Você pode se integrar AWS AppConfig com aplicativos executados em suas instâncias Linux do Amazon Elastic Compute Cloud (Amazon EC2) usando o Agent. AWS AppConfig O agente aprimora o processamento e o gerenciamento de aplicativos das seguintes maneiras:

- O agente liga AWS AppConfig em seu nome usando uma função AWS Identity and Access Management (IAM) e gerenciando um cache local de dados de configuração. Ao extrair dados de configuração do cache local, seu aplicativo exige menos atualizações de código para gerenciar dados de configuração, recupera dados de configuração em milissegundos e não é afetado por problemas de rede que podem afetar as chamadas para esses dados.*
- O agente oferece uma experiência nativa para recuperar e resolver sinalizadores de AWS AppConfig recursos.
- Pronto para uso, o agente fornece as práticas recomendadas para estratégias de armazenamento em cache, intervalos de pesquisa e disponibilidade de dados de configuração locais, enquanto rastreia os tokens de configuração necessários para chamadas de serviço subsequentes.
- Durante a execução em segundo plano, o agente consulta periodicamente o plano de AWS AppConfig dados para atualizações de dados de configuração. Seu aplicativo pode recuperar

os dados conectando-se ao localhost na porta 2772 (um valor de porta padrão personalizável) e chamando HTTP GET para recuperar os dados.

* O AWS AppConfig agente armazena os dados em cache na primeira vez que o serviço recupera seus dados de configuração. Por esse motivo, a primeira chamada para recuperar dados é mais lenta que as chamadas subsequentes.

Tópicos

- Etapa 1: (obrigatória) crie recursos e configure permissões
- Etapa 2: (Obrigatório) Instalar e iniciar o AWS AppConfig agente nas instâncias do Amazon EC2
- Etapa 3: (opcional, mas recomendado) Enviar arquivos de log para o CloudWatch Logs
- Etapa 4: (opcional) Usando variáveis de ambiente para configurar o AWS AppConfig agente para o Amazon EC2
- Etapa 5: (obrigatória) recuperação de dados de configuração
- Etapa 6 (opcional, mas recomendada): automatizar as atualizações do Agente AWS AppConfig

Etapa 1: (obrigatória) crie recursos e configure permissões

Para se integrar AWS AppConfig com aplicativos executados em suas instâncias do Amazon EC2, você deve criar AWS AppConfig artefatos e dados de configuração, incluindo sinalizadores de recursos ou dados de configuração de formato livre. Para obter mais informações, consulte Criação de sinalizadores de recursos e dados de configuração de formato livre no AWS AppConfig.

Para recuperar dados de configuração hospedados por AWS AppConfig, seus aplicativos devem ser configurados com acesso ao plano de AWS AppConfig dados. Para dar acesso aos seus aplicativos, atualize a política de permissões do IAM atribuída à função de instância do Amazon EC2. Especificamente, você deve adicionar as ações `appconfig:StartConfigurationSession` e `appconfig:GetLatestConfiguration` à política. Exemplo:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "appconfig:StartConfigurationSession",
                "appconfig:GetLatestConfiguration"
            ],
            "Resource": "*"
        }
    ]
}
```

Para obter informações sobre como adicionar permissões a uma política, consulte Adicionar e remover permissões de identidade do IAM no Guia do usuário do IAM.

Etapa 2: (Obrigatório) Instalar e iniciar o AWS AppConfig agente nas instâncias do Amazon EC2

AWS AppConfig O agente está hospedado em um bucket do Amazon Simple Storage Service (Amazon S3) que é gerenciado por. AWS Use o procedimento a seguir para instalar a versão mais recente do agente em sua instância do Linux. Se seu aplicativo estiver distribuído entre várias instâncias, você deverá executar este procedimento em cada instância que hospeda o aplicativo.

> **Note**
>
> Observe as seguintes informações:
>
> - AWS AppConfig O agente está disponível para sistemas operacionais Linux que executam a versão 4.15 ou superior do kernel. Sistemas baseados em Debian, como o Ubuntu, não são suportados.
> - O agente oferece suporte a x86_64 e arquiteturas. ARM64
> - Para aplicativos distribuídos, recomendamos adicionar os comandos de instalação e inicialização aos dados de usuário do Amazon EC2 do seu grupo do Auto Scaling. Se fizer isso, cada instância executará os comandos automaticamente. Para ter mais informações, consulte Run commands on your Linux instance at launch no Guia do usuário do Amazon EC2. Além disso, consulte Tutorial: configurar dados do usuário para recuperar o estado do ciclo de vida do destino por meio de metadados da instância no Guia do usuário do Amazon EC2 Auto Scaling.
> - Os procedimentos deste tópico descrevem como realizar ações como instalar o agente fazendo login na instância para executar o comando. Você pode executar os comandos em uma máquina cliente local e direcionar-se a uma ou mais instâncias usando o

Run Command, que é uma ferramenta do AWS Systems Manager. Para obter mais informações, consulte Comando AWS Systems Manager Run no Guia do usuário do AWS Systems Manager .

- AWS AppConfig O agente nas instâncias Linux do Amazon EC2 é um `systemd` serviço.

Para instalar e iniciar o AWS AppConfig Agente em uma instância

1. Faça login na sua instância do Linux.
2. Abra um terminal ou execute um dos seguintes comandos com permissões de administrador:

   x86_64

   ```
   sudo yum install https://s3.amazonaws.com/aws-appconfig-downloads/aws-appconfig-agent/linux/x86_64/latest/aws-appconfig-agent.rpm
   ```

   ARM64

   ```
   sudo yum install https://s3.amazonaws.com/aws-appconfig-downloads/aws-appconfig-agent/linux/arm64/latest/aws-appconfig-agent.rpm
   ```

   Se você quiser instalar uma versão específica do AWS AppConfig Agente, `latest` substitua a URL por um número de versão específico. Veja um exemplo para x86_64:

   ```
   sudo yum install https://s3.amazonaws.com/aws-appconfig-downloads/aws-appconfig-agent/linux/x86_64/2.0.2/aws-appconfig-agent.rpm
   ```
3. Para iniciar o agente, execute o seguinte comando:

   ```
   sudo systemctl start aws-appconfig-agent
   ```
4. Execute o seguinte comando para verificar se o agente está em execução:

   ```
   sudo systemctl status aws-appconfig-agent
   ```

   Se houver êxito, o comando retornará informações como as seguintes:

   ```
   aws-appconfig-agent.service - aws-appconfig-agent
     ...
   ```

```
  Active: active (running) since Mon 2023-07-26 00:00:00 UTC; 0s ago
  ...
```

> **Note**
>
> Para interromper o agente, execute o seguinte comando:
>
> ```
> sudo systemctl stop aws-appconfig-agent
> ```

Etapa 3: (opcional, mas recomendado) Enviar arquivos de log para o CloudWatch Logs

Por padrão, o AWS AppConfig Agente publica registros no STDERR. O Systemd redireciona STDOUT e STDERR para todos os serviços em execução na instância Linux para o diário do systemd. Você pode visualizar e gerenciar dados de log no diário do systemd se estiver executando o AWS AppConfig Agent em apenas uma ou duas instâncias. Uma solução melhor, uma solução altamente recomendada para aplicativos distribuídos, é gravar arquivos de log em disco e, em seguida, usar o CloudWatch agente da Amazon para carregar os dados de log na AWS nuvem. Além disso, você pode configurar o CloudWatch agente para excluir arquivos de log antigos da sua instância, o que evita que ela fique sem espaço em disco.

Para habilitar o log em disco, você deve definir a variável de ambiente `LOG_PATH`, conforme descrito em Etapa 4: (opcional) Usando variáveis de ambiente para configurar o AWS AppConfig agente para o Amazon EC2.

Para começar a usar o CloudWatch agente, consulte Coletar métricas e registros de instâncias do Amazon EC2 e servidores locais com o CloudWatch agente no Guia do usuário da Amazon CloudWatch . Você pode usar o Quick Setup, uma ferramenta no Systems Manager para instalar rapidamente o CloudWatch agente. Para obter mais informações, consulte Gerenciamento do host de Configuração Rápida no Guia do usuário do AWS Systems Manager .

> **Warning**
>
> Se você optar por gravar arquivos de log no disco sem usar o CloudWatch agente, deverá excluir os arquivos de log antigos. AWS AppConfig O agente gira automaticamente os

arquivos de log a cada hora. Se você não excluir os arquivos de log antigos, sua instância poderá ficar sem espaço em disco.

Depois de instalar o CloudWatch agente na sua instância, crie um arquivo de configuração do CloudWatch agente. O arquivo de configuração instrui o CloudWatch agente sobre como trabalhar com os arquivos de log do AWS AppConfig agente. Para obter mais informações sobre a criação de um arquivo de configuração do CloudWatch agente, consulte Criar o arquivo de configuração do CloudWatch agente.

Adicione a `logs` seção a seguir ao arquivo de configuração do CloudWatch agente na instância e salve suas alterações:

```
"logs": {
  "logs_collected": {
    "files": {
      "collect_list": [
        {
          "file_path": "/path_you_specified_for_logging",
          "log_group_name": "${YOUR_LOG_GROUP_NAME}/aws-appconfig-agent.log",
          "auto_removal": true
        },
        ...
      ]
    },
    ...
  },
  ...
}
```

Se o valor de `auto_removal` for`true`, o CloudWatch agente excluirá automaticamente os arquivos de log do AWS AppConfig Agente rotacionados.

Etapa 4: (opcional) Usando variáveis de ambiente para configurar o AWS AppConfig agente para o Amazon EC2

Você pode configurar o AWS AppConfig Agente para o Amazon EC2 usando variáveis de ambiente. Para definir variáveis de ambiente para um serviço do `systemd`, crie um arquivo de unidade drop-in. O exemplo a seguir mostra como criar um arquivo de unidade drop-in para definir o nível de registro do AWS AppConfig agente como. `DEBUG`

Exemplo de como criar um arquivo de unidade drop-in para variáveis de ambiente

1. Faça login na sua instância do Linux.
2. Abra um terminal ou execute o comando a seguir com permissões de administrador. O comando cria um diretório de configuração:

```
sudo mkdir /etc/systemd/system/aws-appconfig-agent.service.d
```

3. Execute o comando a seguir para criar o arquivo de unidade drop-in. *file_name*Substitua por um nome para o arquivo. A extensão deve ser `.conf`:

```
sudo touch /etc/systemd/system/aws-appconfig-agent.service.d/file_name.conf
```

4. Insira as informações no arquivo da unidade drop-in. O exemplo a seguir adiciona uma seção `Service` que define uma variável de ambiente. O exemplo define o nível de log do AWS AppConfig Agent como `DEBUG`.

```
[Service]
Environment=LOG_LEVEL=DEBUG
```

5. Execute o comando a seguir para recarregar a configuração do systemd:

```
sudo systemctl daemon-reload
```

6. Execute o comando a seguir para reiniciar o AWS AppConfig Agente:

```
sudo systemctl restart aws-appconfig-agent
```

Você pode configurar o AWS AppConfig Agent for Amazon EC2 especificando as seguintes variáveis de ambiente em um arquivo de unidade drop-in.

> **Note**
>
> A tabela a seguir inclui uma coluna de valores de amostra. Dependendo da resolução do monitor, talvez seja necessário rolar até a parte inferior da tabela e depois rolar para a direita para ver a coluna.

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)

ACCESS_TOKEN	Esta variável de ambiente define um token que deve ser fornecido ao solicitar dados de configuração do servidor HTTP do agente. O valor do token deve ser definido no cabeçalho de autorização da solicitação HTTP com um tipo de autorização `Bearer`. Aqui está um exemplo.
	`GET /applications/my_app/...`
	`Host: localhost:2772`
	`Authorization: Bearer <token value>`
		Nenhum	MyAccessToken

BACKUP_DIRECTORY	Essa variável de ambiente permite que o AWS AppConfig Agente salve um backup de cada configuração recuperada no diretório especificado.	Nenhum	/path/to/backups

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
	Important As configurações copiadas em disco não são criptografadas. Se sua configuração contiver dados confidenciais, recomendamos AWS AppConfig que você pratique o princípio do menor privilégio com as permissões do sistema de arquivos. Para obter mais informações, consulte Segurança em AWS AppConfig.		

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`HTTP_PORT`	Esta variável de ambiente especific a a porta na qual o servidor HTTP do agente é executado.	2772	2772
`HTTP_HOST`	A variável HTTP_HOST controla como o AWS AppConfig Agente se vincula às interfaces de rede. O comportamento de vinculação difere com base no ambiente de execução para garantir segurança e acessibilidade ideais.	ECS, SEMANAS • Vinculação padrão: todas as interfaces de rede (0.0.0.0) EC2 e no local • Vinculação padrão: somente localhost • IPv4 endereço: 127.0.0. 1:2772 • IPv6 endereço: [::1] :2772	Opções de configura ção personalizadas. Você pode substitui r o comportamento padrão usando estes valores: • `all`(vincula a todas as interfaces) • `localhost` (vincula-se explicitamente às interfaces localhost) • Endereço IP específico (por exemplo`192.168.1 .1`) • Nome de host personalizado

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`LOG_LEVEL`	Esta variável de ambiente especifica o nível de detalhes que o agente registra. Cada nível inclui o nível atual e todos os níveis superiores. O valor diferencia maiúsculas de minúsculas. Do mais detalhado ao menos detalhado, os níveis de log são: `trace`, `debug`, `info`, `warn`, `error`, `fatal` e `none`. O log do `trace` contém informações detalhadas, incluindo informações de tempo, sobre o agente.	info	trace depurar info warn erro fatal nenhuma
`LOG_PATH`	O local do disco em que os logs são gravados. Se não especificado, os logs serão gravados em stderr.	Nenhum	/path/to/logs/agent.log

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`MANIFEST`	Essa variável de ambiente configura o AWS AppConfig Agente para aproveitar os recursos adicionais por configuração, como recuperações de várias contas e salvamento da configuração em disco. Para obter mais informações sobre esses recursos, consulte Usar um manifesto para habilitar recursos de recuperação adicionais.	Nenhum	Ao usar a AWS AppConfig configuração como manifesto :`MyApp:MyEnv:MyManifestConfig` . Ao carregar o manifesto do disco: `file:/path/to/manifest.json`
`MAX_CONNECTIONS`	Esta variável de ambiente configura o número máximo de conexões que o agente usa para recuperar configurações do AWS AppConfig.	3	3

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`POLL_INTERVAL`	Essa variável de ambiente controla a frequência com que o agente pesquisa dados AWS AppConfig de configuração atualizados. É possível especific ar um número de segundos para o intervalo. Você também pode especificar um número com uma unidade de tempo: s para segundos, m para minutos, e h para horas. Se nenhuma unidade for especificada, o agente usará segundos como padrão. Por exemplo, 60, 60 s e 1 min resultam no mesmo intervalo de pesquisa.	45 segundos	45 45s 5 minutos 1h

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`PREFETCH_LIST`	Essa variável de ambiente especific a os dados de configuração que o agente solicita AWS AppConfig assim que é iniciado. Podem ser fornecidos vários identificadores de configuração em uma lista separada por vírgulas.	Nenhum	MyApp:MyE nv:MyConfig abcd123:efgh456:ij kl789 MyApp::Co nfig1, ::Config2 MyEnv MyApp MyEnv

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
PRELOAD_BACKUPS	Se definido como`true`, o AWS AppConfig Agente carrega os backups de configuração encontrados os BACKUP_DIRECTORY na memória e verifica imediatamente se existe uma versão mais recente do serviço. Se definido como `false`, o AWS AppConfig Agent só carregará o conteúdo de um backup de configuração se não conseguir recuperar dados de configuração do serviço; por exemplo, se houver um problema com a rede.	true	verdadeiro false

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
PROXY_HEADERS	Esta variável de ambiente específica a cabeçalhos que são exigidos pelo proxy referenciado na variável de ambiente PROXY_URL. O valor é uma lista de cabeçalhos separados por vírgula.	Nenhum	header: value h1: v1, h2: v2
PROXY_URL	Essa variável de ambiente específica a a URL do proxy a ser usada para conexões do agente com Serviços da AWS, inclusive AWS AppConfig. HTTPSe HTTP URLs são suportados.	Nenhum	http://localhost:7474 https://my-proxy.example.com

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`REQUEST_TIMEOUT`	Essa variável de ambiente controla a quantidade de tempo do qual o agente espera por uma resposta. AWS AppConfig Se o serviço não responder , a solicitação falhará. Se a solicitação for para a recuperação inicial de dados, o agente retornará um erro ao seu aplicativo. Se o tempo limite ocorrer durante uma verificação de dados atualizados em segundo plano, o agente registrar á o erro e tentará novamente após um pequeno atraso. Você pode especific ar o número de milissegundos para o tempo limite. Você pode também especific ar um número com uma unidade de	3000ms	3000 3000ms 5s

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
	tempo: ms, para milissegundos, e s, para segundos. Se nenhuma unidade for especificada, o agente usará milissegundos como padrão. Por exemplo, 5000, 5000 ms e 5 s resultam no mesmo valor de tempo limite da solicitação.		
`ROLE_ARN`	Essa variável de ambiente especifica o Amazon Resource Name (ARN) de uma função do IAM. AWS AppConfig O agente assume essa função para recuperar os dados de configuração.	Nenhum	arn: aws: iam: :123456789012: role/ MyRole
`ROLE_EXTERNAL_ID`	Esta variável de ambiente especifica o ID externo a ser usado com o ARN da função assumida.	Nenhum	MyExternalId

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`ROLE_SESSION_NAME`	Esta variável de ambiente especifica o nome da sessão a ser associado às credenciais do perfil do IAM assumido.	Nenhum	AWSAppConfigAgentSession
`SERVICE_REGION`	Essa variável de ambiente especifica uma alternativa Região da AWS que o AWS AppConfig Agente usa para chamar o AWS AppConfig serviço. Se não for definida, o agente tentará determinar a região atual. Se não for possível, o agente não iniciará.	Nenhum	us-east-1 eu-west-1
`WAIT_ON_MANIFEST`	Essa variável de ambiente configura o AWS AppConfig Agente para esperar até que o manifesto seja processado antes de concluir a inicialização.	true	verdadeiro false

Etapa 5: (obrigatória) recuperação de dados de configuração

Você pode recuperar dados de configuração do AWS AppConfig Agente usando uma chamada HTTP localhost. Os exemplos a seguir usam `curl` com um cliente HTTP. Você pode chamar o agente usando qualquer cliente HTTP disponível compatível com a linguagem do aplicativo ou com as bibliotecas disponíveis, incluindo um AWS SDK.

Para recuperar o conteúdo completo de qualquer configuração implantada

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name"
```

Para recuperar um único sinalizador e seus atributos de uma configuração do AWS AppConfig do tipo **`Feature Flag`**

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name?flag=flag_name"
```

Para acessar vários sinalizadores e seus atributos a partir de uma configuração do AWS AppConfig do tipo **`Feature Flag`**

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name?
flag=flag_name_one&flag=flag_name_two"
```

Etapa 6 (opcional, mas recomendada): automatizar as atualizações do Agente AWS AppConfig

AWS AppConfig O agente é atualizado periodicamente. Para garantir que você esteja executando a versão mais recente do AWS AppConfig Agent em suas instâncias, recomendamos adicionar os comandos a seguir aos seus dados de usuário do Amazon EC2. Você pode adicionar os comandos aos dados do usuário na instância ou no grupo do Auto Scaling do EC2. O script instala e inicia a versão mais recente do agente sempre que uma instância é iniciada ou reinicializada.

```
#!/bin/bash
# install the latest version of the agent
yum install -y https://s3.amazonaws.com/aws-appconfig-downloads/aws-appconfig-agent/
linux/x86_64/latest/aws-appconfig-agent.rpm
# optional: configure the agent
```

```
mkdir /etc/systemd/system/aws-appconfig-agent.service.d
echo "${MY_AGENT_CONFIG}" > /etc/systemd/system/aws-appconfig-agent.service.d/
overrides.conf
systemctl daemon-reload
# start the agent
systemctl start aws-appconfig-agent
```

Usando o AWS AppConfig Agent com o Amazon ECS e o Amazon EKS

Você pode se integrar AWS AppConfig ao Amazon Elastic Container Service (Amazon ECS) e ao Amazon Elastic Kubernetes Service (Amazon EKS) usando o Agent. AWS AppConfig O agente funciona como um contêiner auxiliar executado junto com seus aplicativos de contêiner Amazon ECS e Amazon EKS. O agente aprimora o processamento e o gerenciamento de aplicativos em contêineres das seguintes maneiras:

- O agente liga AWS AppConfig em seu nome usando uma função AWS Identity and Access Management (IAM) e gerenciando um cache local de dados de configuração. Ao extrair dados de configuração do cache local, seu aplicativo exige menos atualizações de código para gerenciar dados de configuração, recupera dados de configuração em milissegundos e não é afetado por problemas de rede que podem afetar as chamadas para esses dados.*
- O agente oferece uma experiência nativa para recuperar e resolver sinalizadores de AWS AppConfig recursos.
- Pronto para uso, o agente fornece as práticas recomendadas para estratégias de armazenamento em cache, intervalos de pesquisa e disponibilidade de dados de configuração local, enquanto rastreia os tokens de configuração necessários para chamadas de serviço subsequentes.
- Durante a execução em segundo plano, o agente consulta periodicamente o plano de AWS AppConfig dados para atualizações de dados de configuração. Seu aplicativo em contêiner pode recuperar os dados conectando-se ao localhost na porta 2772 (um valor de porta padrão personalizável) e chamando HTTP GET para recuperar os dados.
- AWS AppConfig O agente atualiza os dados de configuração em seus contêineres sem precisar reiniciá-los ou reciclá-los.

* O AWS AppConfig agente armazena os dados em cache na primeira vez que o serviço recupera seus dados de configuração. Por esse motivo, a primeira chamada para recuperar dados é mais lenta que as chamadas subsequentes.

Antes de começar

Para se integrar AWS AppConfig aos seus aplicativos de contêiner, você deve criar AWS AppConfig artefatos e dados de configuração, incluindo sinalizadores de recursos ou dados de configuração de formato livre. Para obter mais informações, consulte Criação de sinalizadores de recursos e dados de configuração de formato livre no AWS AppConfig.

Para recuperar dados de configuração hospedados por AWS AppConfig, seus aplicativos de contêiner devem ser configurados com acesso ao plano de AWS AppConfig dados. Para dar acesso aos seus aplicativos, atualize a política de permissões do IAM que é usada pelo perfil do IAM do seu serviço de contêiner. Especificamente, você deve adicionar as ações `appconfig:StartConfigurationSession` e `appconfig:GetLatestConfiguration` à política. Os perfis do IAM de serviço de contêineres incluem o seguinte:

- A função de tarefa do Amazon ECS
- A função do nó do Amazon EKS
- A função de execução do AWS Fargate pod (se seus contêineres do Amazon EKS usarem o Fargate para processamento computacional)

Para obter informações sobre como adicionar permissões a uma política, consulte Adicionar e remover permissões de identidade do IAM no Guia do usuário do IAM.

Tópicos

- Iniciando o AWS AppConfig agente para integração com o Amazon ECS
- Iniciando o AWS AppConfig agente para a integração com o Amazon EKS
- (Opcional) Executando AWS AppConfig como um DaemonSet no Amazon EKS
- (Opcional) Usando variáveis de ambiente para configurar o AWS AppConfig Agente para Amazon ECS e Amazon EKS
- Recuperação de dados de configuração de aplicativos em execução no Amazon ECS e no Amazon EKS

Iniciando o AWS AppConfig agente para integração com o Amazon ECS

O contêiner auxiliar do AWS AppConfig agente está disponível automaticamente em seu ambiente Amazon ECS. Para usá-lo, você deve iniciá-lo, conforme descrito no procedimento a seguir.

Para iniciar o Amazon ECS (console)

1. Abra o console na https://console.aws.amazon.com/ecs/v2.
2. No painel de navegação, escolha Task definitions (Definições de tarefa).
3. Escolha a definição de tarefa para seu aplicativo e, em seguida, selecione a revisão mais recente.
4. Escolha Criar revisão, Criar revisão.
5. Escolha Adicionar mais contêineres.
6. Em Nome, insira um nome exclusivo para o contêiner do AWS AppConfig agente.
7. Em URI da imagem, insira: **`public.ecr.aws/aws-appconfig/aws-appconfig-agent:2.x`**
8. Em Contêiner essencial, escolha Sim.
9. Na seção Mapeamentos de portas, escolha Adicionar mapeamento de portas.
10. Em Porta do contêiner, insira **`2772`**.

> **Note**
>
> AWS AppConfig O agente é executado na porta 2772, por padrão. Você pode especificar uma porta diferente.

11. Escolha Criar. O Amazon ECS cria uma nova revisão do contêiner e exibe os detalhes.
12. No painel de navegação, escolha Clusters e, em seguida, escolha o cluster do aplicativo na lista.
13. Na guia Serviços, selecione o serviço para seu aplicativo.
14. Selecione Atualizar.
15. Em Configuração de implantação, em Revisão, escolha a revisão mais recente.
16. Selecione Atualizar. O Amazon ECS implanta a definição de tarefa mais recente.
17. Depois que a implantação for concluída, você poderá verificar se o AWS AppConfig Agente está sendo executado na guia Configuração e tarefas. Na guia Tarefas, escolha a tarefa em execução.
18. Na seção Contêineres, verifique se o contêiner do AWS AppConfig agente está listado.
19. Para verificar se o AWS AppConfig agente foi iniciado, escolha a guia Registros. Localize uma declaração como a seguinte para o contêiner do AWS AppConfig Agente: `[appconfig agent] 1970/01/01 00:00:00 INFO serving on localhost:2772`

> **Note**
>
> Observe as seguintes informações:
>
> - AWS AppConfig O agente é um processo de longa duração. Como prática recomendada para contêineres do Amazon ECS, configure verificações de integridade para seus contêineres, definindo especificamente a dependência do contêiner para a condição HEALTHY. Para obter mais informações, consulte ContainerDependencya Amazon Elastic Container Service API Reference.
> - Você pode ajustar o comportamento padrão do AWS AppConfig Agente inserindo ou alterando variáveis de ambiente. Para mais informações sobre as variáveis de ambiente disponíveis, consulte (Opcional) Usando variáveis de ambiente para configurar o AWS AppConfig Agente para Amazon ECS e Amazon EKS. Para obter informações sobre como alterar variáveis de ambiente no Amazon ECS, consulte Passar variáveis de ambiente para um contêiner no Guia do desenvolvedor do Amazon Elastic Container Service.

Iniciando o AWS AppConfig agente para a integração com o Amazon EKS

O contêiner auxiliar do AWS AppConfig Agent está disponível automaticamente em seu ambiente Amazon EKS. Para usá-lo, é necessário iniciá-lo. O procedimento a seguir descreve como usar a ferramenta de linha de comandos `kubectl` do Amazon EKS para executar o agente.

> **Note**
>
> Antes de continuar, verifique se o arquivo `kubeconfig` está atualizado. Para obter mais informações sobre como criar ou editar um arquivo `kubeconfig`, consulte Criação ou atualização de um arquivo kubeconfig para um cluster do Amazon EKS no Guia do usuário do Amazon EKS.

Para iniciar o AWS AppConfig Agent (ferramenta de linha de comando kubectl)

1. Abra o manifesto da aplicação e verifique se sua aplicação do Amazon EKS está sendo executada como uma implantação de contêiner único. O conteúdo do arquivo deve ser semelhante ao seguinte:

```
apiVersion: apps/v1
```

```
kind: Deployment
metadata:
  name: my-app
  namespace: my-namespace
  labels:
    app: my-application-label
spec:
  replicas: 1
  selector:
    matchLabels:
      app: my-application-label
  template:
    metadata:
      labels:
        app: my-application-label
    spec:
      containers:
      - name: my-app
        image: my-repo/my-image
        imagePullPolicy: IfNotPresent
```

2. Adicione os detalhes da definição do contêiner do AWS AppConfig Agente ao seu manifesto de implantação.

```
- name: appconfig-agent
        image: public.ecr.aws/aws-appconfig/aws-appconfig-agent:2.x
        ports:
        - name: http
          containerPort: 2772
          protocol: TCP
        env:
        - name: SERVICE_REGION
          value: Região da AWS
        imagePullPolicy: IfNotPresent
```

> **Note**
>
> Observe as seguintes informações:
>
> - AWS AppConfig O agente é executado na porta 2772, por padrão. Você pode especificar uma porta diferente.

- Você pode ajustar o comportamento padrão do AWS AppConfig Agente inserindo variáveis de ambiente. Para obter mais informações, consulte (Opcional) Usando variáveis de ambiente para configurar o AWS AppConfig Agente para Amazon ECS e Amazon EKS.
- Para*Região da AWS*, especifique o Região da AWS código (por exemplo,`us-west-1`) em que o AWS AppConfig Agente recupera os dados de configuração.

3. Execute o comando `kubectl` a seguir para aplicar as alterações ao cluster. *my-deployment*Substitua pelo nome do seu manifesto de implantação.

```
kubectl apply -f my-deployment.yml
```

4. Após a conclusão da implantação, verifique se o AWS AppConfig Agente está em execução. Use o comando a seguir para visualizar o arquivo de log do pod do aplicativo.

```
kubectl logs -n my-namespace -c appconfig-agent my-pod
```

Localize uma declaração como a seguinte para o contêiner do AWS AppConfig Agente: `[appconfig agent] 1970/01/01 00:00:00 INFO serving on localhost:2772`

> **Note**
>
> Você pode ajustar o comportamento padrão do AWS AppConfig Agente inserindo ou alterando variáveis de ambiente. Para mais informações sobre as variáveis de ambiente disponíveis, consulte (Opcional) Usando variáveis de ambiente para configurar o AWS AppConfig Agente para Amazon ECS e Amazon EKS.

(Opcional) Executando AWS AppConfig como um DaemonSet no Amazon EKS

Com o Amazon EKS, você pode executar o AWS AppConfig Agent como um sidecar, o que resulta em um contêiner de agente por pod de aplicativo. Ou, se preferir, você pode executar o AWS AppConfig Agente como um DaemonSet, o que resulta em um contêiner de agente por nó em seu cluster.

> **Note**
>
> Se você executar o AWS AppConfig Agent como um DaemonSet, o agente será executado em um pod separado, o que significa que você não pode acessá-lo com chamadas para `localhost`. É necessário injetar ou descobrir o endereço IP do pod do agente para chamá-lo.

Para executar o AWS AppConfig Agent como um DaemonSet, crie um arquivo de manifesto com o conteúdo a seguir. Substitua o *highlighted* texto por detalhes do seu aplicativo e ambiente. Em *Região da AWS*, especifique um código de Região da AWS (por exemplo, `us-west-1`).

```
apiVersion: apps/v1
kind: DaemonSet
metadata:
  name: aws-appconfig-agent
  namespace: my_namespace
  labels:
    app: my_application_label
spec:
   selector:
    matchLabels:
      app: my_application_label
  template:
    metadata:
      labels:
        app: my_application_label
    spec:
      containers:
      - name: aws-appconfig-agent
        image: public.ecr.aws/aws-appconfig/aws-appconfig-agent:2.x
        ports:
        - name: http
          containerPort: 2772
          protocol: TCP
        env:
        - name: SERVICE_REGION
          value: Região da AWS
        imagePullPolicy: IfNotPresent
      # set a high priority class to ensure the agent is running on every node
      priorityClassName: system-node-critical
```

Execute o comando a seguir para aplicar o AWS AppConfig Agente DaemonSet ao seu cluster. *aws_appconfig_agent_daemonset*Substitua pelo nome do seu DaemonSet manifesto.

```
kubectl apply -f aws_appconfig_agent_daemonset.yml
```

(Opcional) Usando variáveis de ambiente para configurar o AWS AppConfig Agente para Amazon ECS e Amazon EKS

Você pode configurar o AWS AppConfig Agente alterando as seguintes variáveis de ambiente para seu contêiner de agente.

> **Note**
>
> A tabela a seguir inclui uma coluna de valores de amostra. Dependendo da resolução do monitor, talvez seja necessário rolar até a parte inferior da tabela e depois rolar para a direita para ver a coluna.

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`ACCESS_TOKEN`	Esta variável de ambiente define um token que deve ser fornecido ao solicitar dados de configuração do servidor HTTP do agente. O valor do token deve ser definido no cabeçalho de autorização da solicitação HTTP com um tipo de autorização `Bearer`. Aqui está um exemplo.	Nenhum	MyAccessToken

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)

	`GET /applications/my_app/...`
	
	`Host: localhost:2772`
	
	`Authorization: Bearer <token value>`		

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`BACKUP_DIRECTORY`	Essa variável de ambiente permite que o AWS AppConfig Agente salve um backup de cada configuração recuperada no diretório especificado. **Important** As configurações copiadas em disco não são criptografadas. Se sua configuração contiver dados confidenciais, recomendamos AWS AppConfig que você pratique o princípio do menor privilégio com as permissões do sistema de arquivos. Para obter mais informações, consulte	Nenhum	/path/to/backups

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
	Segurança em AWS AppConfig.		
`HTTP_PORT`	Esta variável de ambiente especifica a porta na qual o servidor HTTP do agente é executado.	2772	2772
`HTTP_HOST`	A variável HTTP_HOST controla como o AWS AppConfig Agente se vincula às interfaces de rede. O comportamento de vinculação difere com base no ambiente de execução para garantir segurança e acessibilidade ideais.	ECS, SEMANAS • Vinculação padrão: todas as interfaces de rede (0.0.0.0) EC2 e no local • Vinculação padrão: somente localhost • IPv4 endereço: 127.0.0. 1:2772 • IPv6 endereço: [::1] :2772	Opções de configuração personalizadas. Você pode substituir o comportamento padrão usando estes valores: • `all`(vincula a todas as interfaces) • `localhost` (vincula-se explicitamente às interfaces localhost) • Endereço IP específico (por exemplo`192.168.1.1`) • Nome de host personalizado

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`LOG_LEVEL`	Esta variável de ambiente especifica o nível de detalhes que o agente registra. Cada nível inclui o nível atual e todos os níveis superiores. O valor diferencia maiúsculas de minúsculas. Do mais detalhado ao menos detalhado, os níveis de log são: `trace`, `debug`, `info`, `warn`, `error`, `fatal` e none. O log do `trace` contém informações detalhadas, incluindo informações de tempo, sobre o agente.	info	trace depurar info warn erro fatal nenhuma
`LOG_PATH`	O local do disco em que os logs são gravados. Se não especificado, os logs serão gravados em stderr.	Nenhum	/path/to/logs/agent.log

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`MANIFEST`	Essa variável de ambiente configura o AWS AppConfig Agente para aproveitar os recursos adicionais por configuração, como recuperações de várias contas e salvamento da configuração em disco. Para obter mais informações sobre esses recursos, consulte Usar um manifesto para habilitar recursos de recuperação adicionais.	Nenhum	Ao usar a AWS AppConfig configuração como manifesto :`MyApp:MyEnv:MyManifestConfig` . Ao carregar o manifesto do disco: `file:/path/to/manifest.json`
`MAX_CONNECTIONS`	Esta variável de ambiente configura o número máximo de conexões que o agente usa para recuperar configurações do AWS AppConfig.	3	3

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
POLL_INTERVAL	Essa variável de ambiente controla a frequência com que o agente pesquisa dados AWS AppConfig de configuração atualizados. É possível especificar um número de segundos para o intervalo. Você também pode especificar um número com uma unidade de tempo: s para segundos, m para minutos, e h para horas. Se nenhuma unidade for especificada, o agente usará segundos como padrão. Por exemplo, 60, 60 s e 1 min resultam no mesmo intervalo de pesquisa.	45 segundos	45 45s 5 minutos 1h

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
PREFETCH_LIST	Essa variável de ambiente especific a os dados de configuração que o agente solicita AWS AppConfig assim que é iniciado. Podem ser fornecidos vários identificadores de configuração em uma lista separada por vírgulas.	Nenhum	MyApp:MyE nv:MyConfig abcd123:efgh456:ij kl789 MyApp::Co nfig1, ::Config2 MyEnv MyApp MyEnv

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
PRELOAD_BACKUPS	Se definido como`true`, o AWS AppConfig Agente carrega os backups de configuração encontrados os BACKUP_DIRECTORY na memória e verifica imediatamente se existe uma versão mais recente do serviço. Se definido como `false`, o AWS AppConfig Agent só carregará o conteúdo de um backup de configuração se não conseguir recuperar dados de configuração do serviço; por exemplo, se houver um problema com a rede.	true	verdadeiro false

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
PROXY_HEADERS	Esta variável de ambiente específica a cabeçalhos que são exigidos pelo proxy referenciado na variável de ambiente PROXY_URL. O valor é uma lista de cabeçalhos separados por vírgula.	Nenhum	header: value h1: v1, h2: v2
PROXY_URL	Essa variável de ambiente específica a a URL do proxy a ser usada para conexões do agente com Serviços da AWS, inclusive AWS AppConfig. HTTPSe HTTP URLs são suportados.	Nenhum	http://localhost:7474 https://my-proxy.example.com

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`REQUEST_TIMEOUT`	Essa variável de ambiente controla a quantidade de tempo do qual o agente espera por uma resposta. AWS AppConfig Se o serviço não responder , a solicitação falhará. Se a solicitação for para a recuperação inicial de dados, o agente retornará um erro ao seu aplicativo. Se o tempo limite ocorrer durante uma verificação de dados atualizados em segundo plano, o agente registrar á o erro e tentará novamente após um pequeno atraso. Você pode especific ar o número de milissegundos para o tempo limite. Você pode também especific ar um número com uma unidade de	3000ms	3000 3000ms 5s

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
	tempo: ms, para milissegundos, e s, para segundos. Se nenhuma unidade for especificada, o agente usará milissegundos como padrão. Por exemplo, 5000, 5000 ms e 5 s resultam no mesmo valor de tempo limite da solicitação.		
`ROLE_ARN`	Essa variável de ambiente especifica o Amazon Resource Name (ARN) de uma função do IAM. AWS AppConfig O agente assume essa função para recuperar os dados de configuração.	Nenhum	arn: aws: iam: :123456789012: role/ MyRole
`ROLE_EXTERNAL_ID`	Esta variável de ambiente especifica o ID externo a ser usado com o ARN da função assumida.	Nenhum	MyExternalId

Variável de ambiente	Detalhes	Valor padrão	Exemplo(s) de valor(es)
`ROLE_SESSION_NAME`	Esta variável de ambiente especifica o nome da sessão a ser associado às credenciais do perfil do IAM assumido.	Nenhum	AWSAppConfigAgentSession
`SERVICE_REGION`	Essa variável de ambiente especifica uma alternativa Região da AWS que o AWS AppConfig Agente usa para chamar o AWS AppConfig serviço. Se não for definida, o agente tentará determinar a região atual. Se não for possível, o agente não iniciará.	Nenhum	us-east-1 eu-west-1
`WAIT_ON_MANIFEST`	Essa variável de ambiente configura o AWS AppConfig Agente para esperar até que o manifesto seja processado antes de concluir a inicialização.	true	verdadeiro false

Recuperação de dados de configuração de aplicativos em execução no Amazon ECS e no Amazon EKS

Você pode recuperar dados de configuração do AWS AppConfig Agent para aplicativos em execução no Amazon ECS e no Amazon EKS usando uma chamada HTTP localhost. Os exemplos a seguir usam `curl` com um cliente HTTP. Você pode chamar o agente usando qualquer cliente HTTP disponível compatível com a linguagem do aplicativo ou com as bibliotecas disponíveis.

> **Note**
>
> Para recuperar dados de configuração se a aplicação usar uma barra invertida, por exemplo, “test-backend/test-service”, você precisará usar a codificação de URL.

Para recuperar o conteúdo completo de qualquer configuração implantada

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name"
```

Para recuperar um único sinalizador e seus atributos de uma configuração do AWS AppConfig do tipo **`Feature Flag`**

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name?flag=flag_name"
```

Para acessar vários sinalizadores e seus atributos a partir de uma configuração do AWS AppConfig do tipo **`Feature Flag`**

```
$ curl "http://localhost:2772/applications/application_name/
environments/environment_name/configurations/configuration_name?
flag=flag_name_one&flag=flag_name_two"
```

A chamada retorna metadados de configuração em cabeçalhos HTTP, incluindo a versão da configuração, o tipo de conteúdo e o rótulo da versão da configuração (se aplicável). O corpo da resposta do agente inclui o conteúdo da configuração. Exemplo:

```
HTTP/1.1 200 OK
Configuration-Version: 1
Content-Type: application/json
Date: Tue, 18 Feb 2025 20:20:16 GMT
Content-Length: 31

My test config
```

Recuperar sinalizadores de atributos básicos e multivariante

Para configurações de sinalizadores de recursos (configurações do tipo`AWS.AppConfig.FeatureFlags`), o AWS AppConfig Agente permite que você recupere um único sinalizador ou um subconjunto de sinalizadores em uma configuração. A recuperação de um ou dois sinalizadores será útil se o seu caso de uso precisar usar apenas alguns sinalizadores do perfil de configuração. Os exemplos a seguir usam cURL.

> **Note**
>
> A capacidade de chamar um único sinalizador de recurso ou um subconjunto de sinalizadores em uma configuração só está disponível na versão 2.0.45 e AWS AppConfig superior do Agent.

Você pode recuperar dados AWS AppConfig de configuração de um endpoint HTTP local. Para acessar um sinalizador específico ou uma lista de sinalizadores, use o parâmetro de consulta `?flag=`*`FLAG_KEY`* para um perfil de configuração do AWS AppConfig .

Como recuperar um único sinalizador e os respectivos atributos

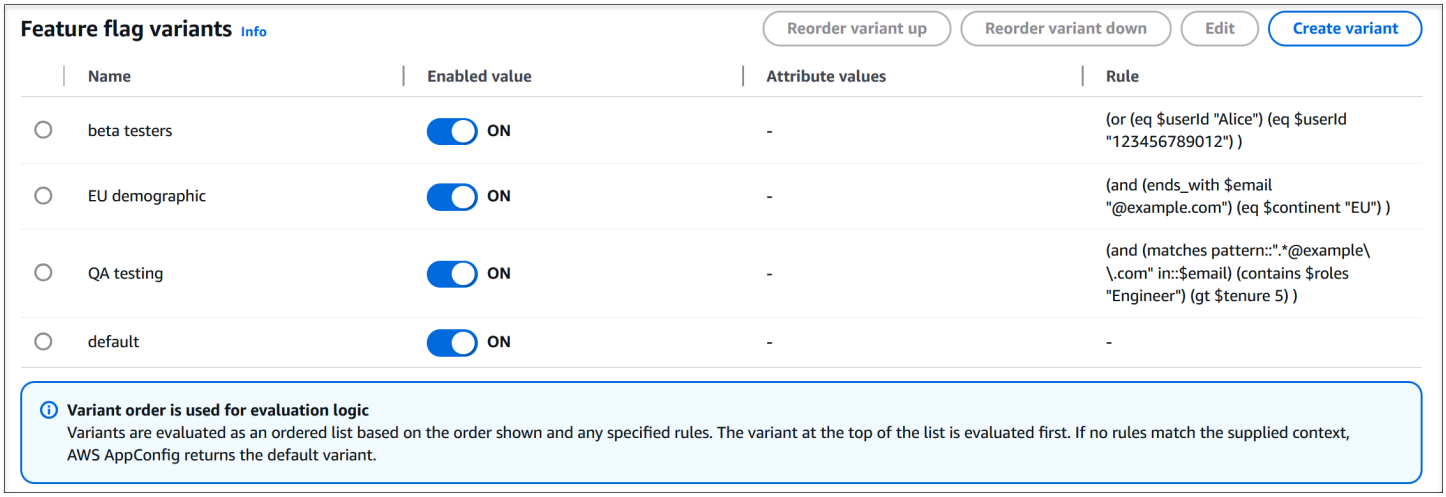
```
curl "http://localhost:2772/applications/APPLICATION_NAME/
environments/ENVIRONMENT_NAME/configurations/CONFIGURATION_NAME?flag=FLAG_KEY"
```

Como recuperar vários sinalizadores e os respectivos atributos

```
curl "http://localhost:2772/applications/APPLICATION_NAME/
environments/ENVIRONMENT_NAME/configurations/CONFIGURATION_NAME?
flag=FLAG_KEY_ONE&flag=FLAG_KEY_TWO"
```

Como recuperar variantes do sinalizador de atributos com base no contexto do chamador

Os exemplos de cURL a seguir mostram como recuperar variantes do sinalizador de atributos com base no contexto do chamador. Para melhor ilustrar como fazer essas chamadas, esta seção usa exemplos de chamadas com base em um cenário em que um cliente criou variantes semelhantes a estas:



> **Note**
>
> Para recuperar as variantes do sinalizador, você deve usar a versão mais recente do AWS AppConfig Agent em seu ambiente computacional. Para ter mais informações, consulte os tópicos abaixo sobre como atualizar, instalar ou adicionar o agente para cada um dos seguintes ambientes computacionais:
>
> - Para ambientes de computação do Lambda: Adicionando a extensão AWS AppConfig Agent Lambda
> - Para ambientes de computação do Amazon EC2: Etapa 2: (Obrigatório) Instalar e iniciar o AWS AppConfig agente nas instâncias do Amazon EC2
> - Para ambientes de computação do Amazon ECS: Iniciando o AWS AppConfig agente para integração com o Amazon ECS
> - Para ambientes de computação do Amazon EKS: Iniciando o AWS AppConfig agente para a integração com o Amazon EKS

Para recuperar dados do sinalizador usando o contexto do chamador de jane_doe@example.org (que não optou pelo programa beta):

```
curl http://localhost:2772/applications/UIRefresh/environments/Production/
configurations/Features \
```

```
-H "Context: email=jane_doe@example.org" \
-H "Context: opted_in_to_beta=false"
{
  "ui_refresh": {"_variant":"QA","dark_mode_support":true,"enabled":true}
}
```

Para recuperar dados do sinalizador usando o contexto do chamador de jane_doe@example.org (que optou pelo programa beta):

```
curl http://localhost:2772/applications/UIRefresh/environments/Production/
configurations/Features \
-H "Context: email=jane_doe@example.org" \
-H "Context: opted_in_to_beta=true"
{
  "ui_refresh": {"_variant":"QA","dark_mode_support":true,"enabled":true}
}
```

Para recuperar dados do sinalizador usando o contexto do chamador de jane_doe@qa-testers.example.org (que é testadora de garantia de qualidade na organização de exemplo):

```
curl http://localhost:2772/applications/UIRefresh/environments/Production/
configurations/Features \
-H "Context: email=jane_doe@qa-testers.example.org"
{
  "ui_refresh": {"_variant":"QA","dark_mode_support":true,"enabled":true}
}
```

Como recuperar dados do sinalizador sem o contexto do chamador (que exibe a variante padrão)

```
curl http://localhost:2772/applications/UIRefresh/environments/Production/
configurations/Features
{
"ui_refresh": {"_variant":"Default Variant","enabled":false}
}
```

Como recuperar dados do sinalizador referentes a um cenário de divisão de tráfego para determinar se um em cada dez chamadores aleatórios recebe a variante “Sample Population”

```
for i in {0..9} do ; \
curl http://localhost:2772/applications/UIRefresh/environments/Production/
configurations/Features \
```

```
-H "Context: email=$i@example.org"
{
  "ui_refresh": {"_variant":"Default Variant","enabled":false}
}
{
  "ui_refresh": {"_variant":"Default Variant","enabled":false}
}
{
  "ui_refresh": {"_variant":"Default Variant","enabled":false}
}
{
  "ui_refresh": {"_variant":"Default Variant","enabled":false}
}
{
  "ui_refresh": {"_variant":"Sample
 Population","dark_mode_support":false,"enabled":true}
}
{
  "ui_refresh": {"_variant":"Default Variant","enabled":false}
}
{
  "ui_refresh": {"_variant":"Default Variant","enabled":false}
}
{
  "ui_refresh": {"_variant":"Default Variant","enabled":false}
}
{
  "ui_refresh": {"_variant":"Default Variant","enabled":false}
}
{
  "ui_refresh": {"_variant":"Default Variant","enabled":false}
}
{
  "ui_refresh": {"_variant":"Default Variant","enabled":false}
}
```

Usar um manifesto para habilitar recursos de recuperação adicionais

AWS AppConfig O agente oferece os seguintes recursos adicionais para ajudá-lo a recuperar configurações para seus aplicativos.

- Configurando o AWS AppConfig Agente para recuperar configurações de várias contas: use o AWS AppConfig Agente de uma conta primária ou de recuperação Conta da AWS para recuperar dados de configuração de várias contas de fornecedores.
- Configurando o AWS AppConfig Agente para gravar cópias de configuração em disco: use o AWS AppConfig Agent para gravar dados de configuração em disco. Esse recurso permite que os clientes com aplicações que leem dados de configuração do disco se integrem ao AWS AppConfig.

Noções básicas sobre manifestos do agente

Para habilitar esses recursos AWS AppConfig do Agente, você cria um manifesto. Manifesto é um conjunto de dados de configuração que você fornece para controlar as ações que o agente pode realizar. Um manifesto é escrito em JSON. Ele contém um conjunto de chaves de nível superior que correspondem às diferentes configurações que você implantou usando. AWS AppConfig

Um manifesto pode incluir várias configurações. Além disso, cada configuração no manifesto pode identificar um ou mais recursos do agente a serem usados na configuração especificada. O conteúdo do manifesto usa o seguinte formato:

```
{
    "application_name:environment_name:configuration_name": {
        "agent_feature_to_enable_1": {
            "feature-setting-key": "feature-setting-value"
        },
        "agent_feature_to_enable_2": {
            "feature-setting-key": "feature-setting-value"
        }
    }
}
```

Veja um exemplo de JSON para um manifesto com duas configurações. A primeira configuração (*MyApp*) não usa nenhum recurso do AWS AppConfig Agente. A segunda configuração (*My2ndApp*) usa a cópia da configuração de gravação em disco e os recursos de recuperação de várias contas:

```
{
        "MyApp:Test:MyAllowListConfiguration": {},

        "My2ndApp:Beta:MyEnableMobilePaymentsFeatureFlagConfiguration": {
            "credentials": {
                "roleArn": "arn:aws:us-west-1:iam::123456789012:role/MyTestRole",
                "roleExternalId": "00b148e2-4ea4-46a1-ab0f-c422b54d0aac",
```

```
                "roleSessionName": "AwsAppConfigAgent",
                "credentialsDuration": "2h"
            },
            "writeTo": {
                "path": "/tmp/aws-appconfig/my-2nd-app/beta/my-enable-payments-feature-
flag-configuration.json"
            }
        }
    }
```

Como fornecer um manifesto do agente

Você pode armazenar o manifesto como um arquivo em um local onde o AWS AppConfig Agente possa lê-lo. Ou você pode armazenar o manifesto como uma AWS AppConfig configuração e direcionar o agente para ele. Para fornecer um manifesto do agente, é necessário definir uma variável de ambiente `MANIFEST` com um dos seguintes valores:

Local do manifesto	Valor de variável de ambiente	Caso de uso
Arquivo	arquivo:/path/to/agent-manifest.json	Use esse método se o manifesto não mudar com frequência.
AWS AppConfig configuração	*application-name*:*environment-name*:*configuration-name*	Use esse método para atualizações dinâmicas. Você pode atualizar e implantar um manifesto armazenado o AWS AppConfig como configuração da mesma forma que armazena outras AWS AppConfig configurações.
Variável de ambiente	Conteúdo do manifesto (JSON)	Use esse método se o manifesto não mudar com frequência. Esse método é útil em ambientes de contêiner em que é mais fácil definir uma variável de ambiente do que expor um arquivo.

Para obter mais informações sobre a configuração de variáveis para o AWS AppConfig Agent, consulte o tópico relevante para seu caso de uso:

- Configurando a extensão do AWS AppConfig Agent Lambda
- Usando o AWS AppConfig agente com o Amazon EC2
- Usando o AWS AppConfig Agent com o Amazon ECS e o Amazon EKS

Configurando o AWS AppConfig Agente para recuperar configurações de várias contas

Você pode configurar o AWS AppConfig Agente para recuperar configurações de várias Contas da AWS inserindo substituições de credenciais no manifesto do Agente. AWS AppConfig As substituições de credenciais incluem o Amazon Resource Name (ARN) de uma função AWS Identity and Access Management (IAM), um ID da função, um nome de sessão e a duração de quanto tempo o agente pode assumir a função.

Você insere esses detalhes em uma seção de “credenciais” no manifesto. A seção “credenciais” usa o seguinte formato:

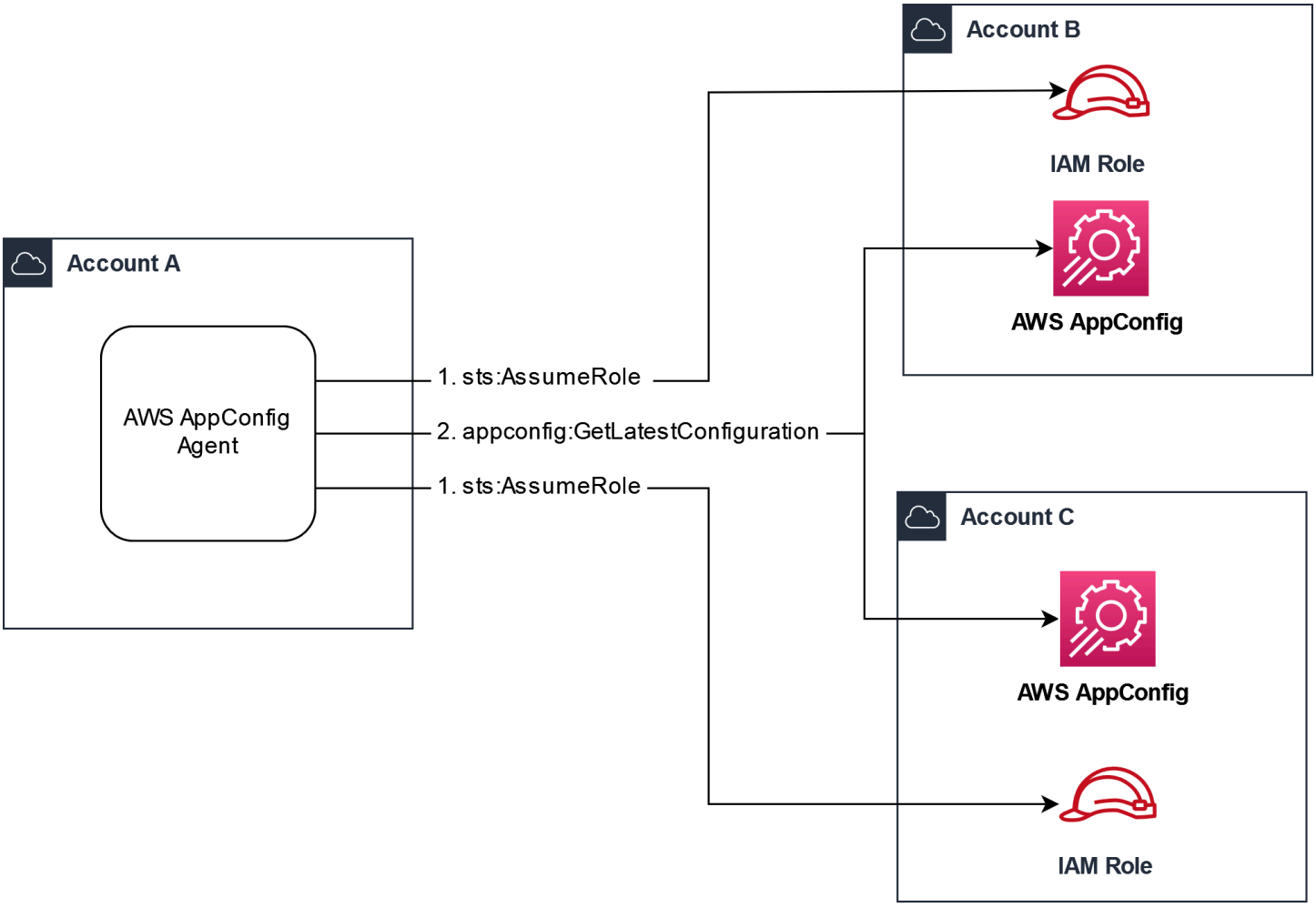
```
{
    "application_name:environment_name:configuration_name": {
        "credentials": {
            "roleArn": "arn:partition:iam::account_ID:role/roleName",
            "roleExternalId": "string",
            "roleSessionName": "string",
            "credentialsDuration": "time_in_hours"
        }
    }
}
```

Exemplo:

```
{
    "My2ndApp:Beta:MyEnableMobilePaymentsFeatureFlagConfiguration": {
        "credentials": {
            "roleArn": "arn:aws:us-west-1:iam::123456789012:role/MyTestRole",
            "roleExternalId": "00b148e2-4ea4-46a1-ab0f-c422b54d0aac",
            "roleSessionName": "AWSAppConfigAgent",
            "credentialsDuration": "2h"
```

```
        }
    }
}
```

Antes de recuperar uma configuração, o agente lê os detalhes da credencial da configuração no manifesto e, depois, assume o perfil do IAM especificado para essa configuração. É possível especificar um conjunto diferente de substituições de credencial para configurações diferentes em um único manifesto. O diagrama a seguir mostra como o AWS AppConfig Agente, ao ser executado na Conta A (a conta de recuperação), assume funções separadas especificadas para as Contas B e C (as contas do fornecedor) e, em seguida, chama a operação da GetLatestConfigurationAPI para recuperar os dados de configuração da AWS AppConfig execução nessas contas:



Configurar permissões para recuperar dados de configuração das contas fornecedoras

AWS AppConfig O agente em execução na conta de recuperação precisa de permissão para recuperar dados de configuração das contas do fornecedor. Você concede permissão ao agente criando uma função AWS Identity and Access Management (IAM) em cada uma das contas do

fornecedor. AWS AppConfig O agente na conta de recuperação assume essa função para obter dados das contas do fornecedor. Conclua os procedimentos nesta seção para criar uma política de permissões do IAM, um perfil do IAM e adicionar substituições do agente ao manifesto.

Antes de começar

Colete as informações a seguir antes de criar uma política de permissão e um perfil no IAM.

- O IDs para cada um Conta da AWS. A conta de recuperação é a conta que chamará outras contas para dados de configuração. As contas fornecedoras são as contas que fornecerão os dados de configuração à conta de recuperação.
- O nome da função do IAM usada AWS AppConfig na conta de recuperação. Aqui está uma lista das funções usadas por AWS AppConfig, por padrão:
  - Para o Amazon Elastic Compute Cloud (Amazon EC2) AWS AppConfig , usa a função de instância.
  - Para AWS Lambda, AWS AppConfig usa a função de execução do Lambda.
  - Para o Amazon Elastic Container Service (Amazon ECS) e o Amazon Elastic Kubernetes Service (Amazon EKS), usa a função de contêiner. AWS AppConfig

  Se você configurou o AWS AppConfig Agente para usar uma função diferente do IAM especificando a variável de `ROLE_ARN` ambiente, anote esse nome.

Criar a política de permissões

Use o procedimento a seguir para criar uma política de permissões usando o console do IAM. Conclua o procedimento em cada um deles Conta da AWS que fornecerá dados de configuração para a conta de recuperação.

Para criar uma política do IAM

1. Faça login Console de gerenciamento da AWS na conta de um fornecedor.
2. Abra o console do IAM em https://console.aws.amazon.com/iam/.
3. No painel de navegação, selecione Políticas e, em seguida, Criar política.
4. Escolha a opção JSON.
5. No Editor de políticas, substitua o JSON padrão pela declaração de política a seguir. Atualize cada um *`example resource placeholder`* com os detalhes da conta do fornecedor.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "appconfig:StartConfigurationSession",
                "appconfig:GetLatestConfiguration"
            ],
            "Resource": "arn:aws:appconfig:us-
east-1:111122223333:application/vendor_application_ID/
environment/vendor_environment_ID/configuration/vendor_configuration_ID"
        }
    ]
}
```

Veja um exemplo abaixo:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "appconfig:StartConfigurationSession",
            "appconfig:GetLatestConfiguration"
        ],
        "Resource": "arn:aws:appconfig:us-east-2:111122223333:application/
abc123/environment/def456/configuration/hij789"
    }
   ]
}
```

6. Escolha Próximo.
7. No campo Nome da política, insira um nome.

8. (Opcional) Em Adicionar tags, adicione um ou mais pares de chave-valor de tag para organizar, monitorar ou controlar o acesso para essa política.
9. Selecione Criar política. O sistema retorna para a página Policies (Políticas).
10. Repita esse procedimento em cada uma delas Conta da AWS que fornecerá dados de configuração para a conta de recuperação.

Crie o perfil do IAM.

Use o procedimento a seguir para criar um perfil do IAM usando o console do IAM. Conclua o procedimento em cada um deles Conta da AWS que fornecerá dados de configuração para a conta de recuperação.

Para criar um perfil do IAM

1. Faça login Console de gerenciamento da AWS na conta de um fornecedor.
2. Abra o console do IAM em https://console.aws.amazon.com/iam/.
3. No painel de navegação, selecione Perfis e, depois, Criar política.
4. Em Tipo de entidade confiável, escolha Conta da AWS.
5. Na seção Conta da AWS, escolha Outra Conta da AWS.
6. No campo ID da conta, insira o ID da conta de recuperação.
7. (Opcional) Como prática recomendada de segurança para esse perfil assumido, selecione Exigir ID externo e insira uma string.
8. Escolha Próximo.
9. Na página Adicionar permissões, use o campo Pesquisar para localizar a política criada no procedimento anterior. Marque a caixa de seleção ao lado do nome.
10. Escolha Próximo.
11. Em Nome do perfil, insira um nome.
12. (Opcional) Em Description (Descrição), insira uma descrição.
13. Em Etapa 1: selecionar entidades confiáveis, selecione Editar. Substitua a política de confiança JSON padrão pela política a seguir. Atualize cada um *`example resource placeholder`* com as informações da sua conta de recuperação.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS":
 "arn:aws:iam::111122223333:role/appconfig_role_in_retrieval_account"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

14. (Opcional) em Tags, adicione um ou mais pares de valores tag-chave para organizar, monitorar ou controlar acesso para essa função.
15. Selecione Create role (Criar função). O sistema faz com que você retorne para a página Roles.
16. Procure o perfil que você acabou de criar. Escolha-o. Na seção ARN, copie o ARN. Você especificará essas informações no próximo procedimento.

Adicionar substituições de credencial ao manifesto

Depois de criar perfil do IAM em sua conta fornecedora, atualize o manifesto na conta de recuperação. Especificamente, adicione o bloco de credenciais e o ARN do perfil do IAM para recuperar dados de configuração da conta fornecedora. Veja o formato JSON:

```
{
    "vendor_application_name:vendor_environment_name:vendor_configuration_name": {
        "credentials": {
            "roleArn":
 "arn:partition:iam::vendor_account_ID:role/name_of_role_created_in_vendor_account",
            "roleExternalId": "string",
            "roleSessionName": "string",
            "credentialsDuration": "time_in_hours"
        }
    }
}
```

Exemplo:

```
{
    "My2ndApp:Beta:MyEnableMobilePaymentsFeatureFlagConfiguration": {
        "credentials": {
            "roleArn": "arn:aws:us-west-1:iam::123456789012:role/MyTestRole",
            "roleExternalId": "00b148e2-4ea4-46a1-ab0f-c422b54d0aac",
            "roleSessionName": "AwsAppConfigAgent",
            "credentialsDuration": "2h"
        }
    }
}
```

Confirmar se a recuperação de várias contas está funcionando

Você pode validar se esse agente é capaz de recuperar dados de configuração de várias contas revisando os registros do AWS AppConfig agente. O log de nível `INFO` dos dados iniciais recuperados para "`YourApplicationName:YourEnvironmentName:YourConfigurationName`" é o melhor indicador de recuperações bem-sucedidas. Se as recuperações estiverem falhando, você deverá ver um log de nível `ERROR` indicando o motivo da falha. Aqui está um exemplo de recuperação bem-sucedida de uma conta fornecedora:

```
[appconfig agent] 2023/11/13 11:33:27 INFO AppConfig Agent 2.0.x
[appconfig agent] 2023/11/13 11:33:28 INFO serving on localhost:2772
[appconfig agent] 2023/11/13 11:33:28 INFO retrieved initial data for
 'MyTestApplication:MyTestEnvironment:MyDenyListConfiguration' in XX.Xms
```

Configurando o AWS AppConfig Agente para gravar cópias de configuração em disco

Você pode configurar o AWS AppConfig Agente para armazenar automaticamente uma cópia de uma configuração em disco em texto simples. Esse recurso permite que os clientes com aplicações que leem dados de configuração do disco se integrem ao AWS AppConfig.

Esse recurso não foi projetado para ser usado como um recurso de backup de configuração. AWS AppConfig Agent não lê os arquivos de configuração copiados no disco. Se você quiser fazer backup das configurações em disco, consulte as variáveis de `PRELOAD_BACKUP` ambiente `BACKUP_DIRECTORY` e de uso do agente com o Amazon EC2 ou o uso do AWS AppConfig agente com AWS AppConfig o Amazon ECS e o Amazon EKS.

Warning

Observe as seguintes informações importantes sobre esse recurso:

- As configurações salvas em disco são armazenadas em texto simples e são legíveis. Não habilite esse recurso para configurações que incluam dados confidenciais.
- Esse recurso faz gravações no disco local. Use o princípio de privilégio mínimo para permissões do sistema de arquivos. Para obter mais informações, consulte Implemente o acesso de privilégio mínimo.

Como habilitar a gravação de cópia de configuração em disco

1. Edite o manifesto.
2. Escolha a configuração que você AWS AppConfig deseja gravar no disco e adicione um `writeTo` elemento. Exemplo:

```
{
    "application_name:environment_name:configuration_name": {
        "writeTo": {
            "path": "path_to_configuration_file"
        }
    }
}
```

Exemplo:

```
{
    "MyTestApp:MyTestEnvironment:MyNewConfiguration": {
        "writeTo": {
            "path": "/tmp/aws-appconfig/mobile-app/beta/enable-mobile-payments"
        }
    }
}
```

3. Salve as alterações. O arquivo configuration.json será atualizado sempre que novos dados de configuração forem implantados.

Confirmar se a cópia de gravação da configuração em disco está funcionando

Você pode validar se cópias de uma configuração estão sendo gravadas em disco examinando os registros do AWS AppConfig agente. A entrada de `INFO` registro com a frase “INFO escreveu a configuração '*application*:*environment*:*configuration*' para*file_path*" indica que o AWS AppConfig Agente grava cópias da configuração no disco.

Exemplo:

```
[appconfig agent] 2023/11/13 11:33:27 INFO AppConfig Agent 2.0.x
[appconfig agent] 2023/11/13 11:33:28 INFO serving on localhost:2772
[appconfig agent] 2023/11/13 11:33:28 INFO retrieved initial data for
 'MobileApp:Beta:EnableMobilePayments' in XX.Xms
[appconfig agent] 2023/11/13 17:05:49 INFO wrote configuration
 'MobileApp:Beta:EnableMobilePayments' to /tmp/configs/your-app/your-env/your-
config.json
```

Gerar um cliente usando a especificação OpenAPI

Você pode usar a seguinte especificação YAML para que a OpenAPI crie um SDK usando uma ferramenta como o OpenAPI Generator. Você pode atualizar essa especificação para incluir valores codificados para aplicativo, ambiente ou configuração. Você também pode adicionar caminhos adicionais (se tiver vários tipos de configuração) e incluir esquemas de configuração para gerar modelos com tipos específicos de configuração para seus clientes SDK. Para obter mais informações sobre a OpenAPI (também conhecida como Swagger), consulte a especificação da OpenAPI.

```
openapi: 3.0.0
info:
  version: 1.0.0
  title: AWS AppConfig Agent API
  description: An API model for AWS AppConfig Agent.
servers:
  - url: http://localhost:{port}/
    variables:
      port:
        default:
          '2772'
paths:
  /applications/{Application}/environments/{Environment}/configurations/
{Configuration}:
    get:
      operationId: getConfiguration
      tags:
```

```
        - configuration
      parameters:
        - in: path
          name: Application
          description: The application for the configuration to get. Specify either the
application name or the application ID.
          required: true
          schema:
            type: string
        - in: path
          name: Environment
          description: The environment for the configuration to get. Specify either the
environment name or the environment ID.
          required: true
          schema:
            type: string
        - in: path
          name: Configuration
          description: The configuration to get. Specify either the configuration name
or the configuration ID.
          required: true
          schema:
            type: string
        - in: query
          name: flag
          description: The key(s) of the feature flag(s) to retrieve. If not provided,
all flags are returned.
          required: false
          schema:
            type: array
            items:
              type: string
        - in: header
          name: context
          description: Request context used to evaluate multi-variant feature flags.
          required: false
          schema:
            type: array
            items:
              type: string
              pattern: '^\w+=\w+$'
      responses:
        200:
          headers:
```

```
            ConfigurationVersion:
              schema:
                type: string
          content:
            application/octet-stream:
              schema:
                type: string
                format: binary
          description: successful config retrieval
        400:
          description: BadRequestException
          content:
            application/text:
              schema:
                $ref: '#/components/schemas/Error'
        404:
          description: ResourceNotFoundException
          content:
            application/text:
              schema:
                $ref: '#/components/schemas/Error'
        500:
          description: InternalServerException
          content:
            application/text:
              schema:
                $ref: '#/components/schemas/Error'
        502:
          description: BadGatewayException
          content:
            application/text:
              schema:
                $ref: '#/components/schemas/Error'
        504:
          description: GatewayTimeoutException
          content:
            application/text:
              schema:
                $ref: '#/components/schemas/Error'

components:
  schemas:
    Error:
      type: string
```

```
      description: The response error
```

Trabalhando com o modo de desenvolvimento local do AWS AppConfig agente

AWS AppConfig O agente suporta um modo de desenvolvimento local. Se você habilitar o modo de desenvolvimento local, o agente lerá os dados de configuração de um diretório especificado no disco. Ele não recupera dados de configuração de AWS AppConfig. É possível simular implantações de configuração atualizando arquivos no diretório especificado. Recomendamos o modo de desenvolvimento local para os seguintes casos de uso:

- Testar diferentes versões de configuração antes de implantá-las usando o AWS AppConfig.
- Testar diferentes opções de configuração para um novo atributo antes de confirmar as alterações no seu repositório de código.
- Testar diferentes cenários de configuração para verificar se funcionam conforme o esperado.

> **Warning**
>
> Não use o modo de desenvolvimento local em ambientes de produção. Esse modo não oferece suporte a recursos de AWS AppConfig segurança importantes, como validação de implantação e reversões automatizadas.

Use o procedimento a seguir para configurar o AWS AppConfig Agente para o modo de desenvolvimento local.

Para configurar o AWS AppConfig Agente para o modo de desenvolvimento local

1. Instale o agente usando o método descrito para seu ambiente computacional. AWS AppConfig O agente trabalha com o seguinte Serviços da AWS:
   - AWS Lambda
   - Amazon EC2
   - Amazon ECS e Amazon EKS
2. Se o agente estiver em execução, interrompa-o.

3. Adicione `LOCAL_DEVELOPMENT_DIRECTORY` à lista de variáveis de ambiente. Especifique um diretório no sistema de arquivos que forneça permissões de leitura ao agente. Por exemplo, `./tmp/local_configs`
4. Crie um arquivo no diretório. O nome do arquivo deve usar o seguinte formato:

   ```
   application_name:environment_name:configuration_profile_name
   ```

   Exemplo:

   ```
   Mobile:Development:EnableMobilePaymentsFeatureFlagConfiguration
   ```

   > **Note**
   >
   > - Para ver exemplos de sinalizadores de atributos que você pode adicionar a um arquivo no seu diretório `LOCAL_DEVELOPMENT_DIRECTORY`, consulte Exemplos de sinalizadores de recursos para o modo de desenvolvimento local do AWS AppConfig agente.
   > - (Opcional) É possível controlar o tipo de conteúdo que o agente exibe para seus dados de configuração com base na extensão atribuída ao arquivo. Por exemplo, se você nomear o arquivo com uma extensão .json, o agente exibirá um tipo de conteúdo de `application/json` quando a aplicação o solicitar. Se você omitir a extensão, o agente usará `application/octet-stream` para o tipo de conteúdo. Se precisar de um controle preciso, poderá fornecer uma extensão no formato `.type%subtype`. O agente exibirá um tipo de conteúdo de `.type/subtype`.
5. Execute o comando a seguir para reiniciar o agente e solicitar dados de configuração.

   ```
   curl http://localhost:2772/applications/application_name/
   environments/environment_name/configurations/configuration_name
   ```

O agente confere as alterações no arquivo local no intervalo de pesquisa especificado para ele. Se o intervalo de pesquisa não for especificado, o agente usará o intervalo padrão de 45 segundos. Essa verificação no intervalo da pesquisa garante que o agente se comporte da mesma forma em um ambiente de desenvolvimento local e quando configurado para interagir com o serviço. AWS AppConfig

> **Note**
>
> Para implantar uma nova versão de um arquivo de configuração de desenvolvimento local, atualize o arquivo com novos dados.

Exemplos de sinalizadores de recursos para o modo de desenvolvimento local do AWS AppConfig agente

Esta seção inclui exemplos de sinalizadores de recursos que você pode usar com o AWS AppConfig Agent no modo de desenvolvimento local. O modo de desenvolvimento local espera dados do sinalizador de atributos no formato de tempo de recuperação dos dados. O formato de tempo de recuperação é o formato retornado quando o sinalizador é recuperado da GetLatestConfigurationAPI, que contém apenas o valor do sinalizador. O formato de tempo de recuperação não inclui a definição completa de um sinalizador (conforme passada para a CreateHostedConfigurationVersionAPI). A definição completa de um sinalizador também contém informações como nomes e valores de atributos, restrições e o estado ativado do sinalizador.

Tópicos

- Exemplos básicos de sinalizadores de atributos
- Exemplos de sinalizadores de atributos multivariante

Exemplos básicos de sinalizadores de atributos

Use os seguintes exemplos de sinalizadores de recursos básicos com o AWS AppConfig Agent no modo de desenvolvimento local.

> **Note**
>
> Se você quiser que o agente reporte o tipo de conteúdo dos seus dados de sinalização de recurso local como `application/json` (como faria ao recuperar dados de sinalização AWS AppConfig em um ambiente que não está no modo de desenvolvimento local), seus arquivos de sinalização de recurso locais devem usar a extensão.json. Por exemplo, `.Local:MyFeatureFlags:SampleB1.json`

Exemplo 1: um único sinalizador que representa uma atualização da interface do usuário.

```
{
    "ui_refresh": {
        "enabled": true,
        "new_styleguide_colors": true
    }
}
```

Exemplo 2: vários sinalizadores que representam sinalizadores de atributos operacionais.

```
{
   "background_worker": {
        "enabled": true,
        "num_threads": 4,
        "queue_name": "MyWorkQueue"
   },
   "emergency_shutoff_switch": {
        "enabled": false
   },
   "logger_settings": {
        "enabled": true,
        "level": "INFO"
   }
}
```

Exemplos de sinalizadores de atributos multivariante

O formato de tempo de recuperação de uma configuração de sinalizador de atributos que contém pelo menos um sinalizador de atributos multivariante é representado como dados do Amazon Ion, em vez de dados JSON. Nesse formato, os sinalizadores multivariante são representados como uma lista anotada, e os sinalizadores básicos são representados como uma string anotada. Os elementos da lista de um sinalizador multivariante são uma tupla (uma lista com tamanho de dois), que representa uma única variante, ou uma string, que representa a variante padrão. Em uma tupla variante, o primeiro elemento é uma expressão “s” que representa a regra da variante, e o segundo elemento é uma string que representa o conteúdo da variante.

Para que o agente interprete corretamente esses arquivos, seus arquivos de sinalizadores de atributos locais devem usar a seguinte extensão: `.application%ion%type=AWS.AppConfig.FeatureFlags`. Por exemplo, `.Local:MyFeatureFlags:SampleMV1.application%ion%type=AWS.AppConfig.FeatureFlags`

Exemplo 1: um sinalizador multivariante que representa uma versão em camadas de um novo atributo.

```
'tiered_release'::[
  [
    (or (and (eq $group "Tier1") (split by::$userId pct::1 seed::"2025.01.01")) (and
 (eq $group "Tier2") (split by::$userId pct::7 seed::"2025.01.01"))),
    '''{"_variant": "ShowFeature", "enabled": true}'''
  ],
  '''{"_variant": "HideFeature", "enabled": false}'''
]
```

Exemplo 2: vários sinalizadores que representam diferentes exibições de UX com base no ID do usuário. Os dois primeiros sinalizadores são multivariante, e o sinalizador final é básico.

```
'colorway'::[
  [
    (contains $userId "beta"),
    '''{"_variant": "BetaTesters", "enabled": true, "background": "blue", "foreground":
 "red"}''',
  ],
  [
    (split by::$userId pct::10),
    '''{"_variant": "SplitRollOutRedAndBlue", "enabled": true, "background": "blue",
 "foreground": "red"}''',
  ],
  '''{"_variant": "default", "enabled": true, "background": "green", "foreground":
 "green"}''',
]

'simple_feature'::[
  [
    (contains $userId "beta"),
    '''{"_variant": "BetaTesters", "enabled": true}'''
  ],
  '''{"_variant": "default", "enabled": false}'''
]

'button_color'::'''{"enabled": true, "color": "orange"}'''
```

AWS AppConfig considerações sobre o uso de navegadores e dispositivos móveis

Os sinalizadores de atributos permitem que você atualize a experiência das suas páginas da web e aplicativos móveis em tempo real, sem a sobrecarga, o risco ou a rigidez de um lançamento na loja de aplicativos. Com sinalizadores de atributos, você pode liberar gradualmente uma alteração para sua base de usuários no momento da escolha. Se houver um erro, você poderá reverter instantaneamente a alteração sem exigir que os usuários atualizem para uma nova versão do software. Em suma, os sinalizadores de atributos fornecem maior controle e flexibilidade ao implantar alterações no seu aplicativo.

As seções a seguir descrevem considerações importantes sobre o uso de sinalizadores de AWS AppConfig recursos com páginas da Web e dispositivos móveis.

Tópicos

- Dados de configuração e recuperação de sinalizadores
- Autenticação e Amazon Cognito
- Armazenamento em cache
- Segmentação
- Largura de banda (casos de uso em dispositivos móveis)
- Casos de uso de sinalizadores adicionais

Dados de configuração e recuperação de sinalizadores

Para casos de uso de navegadores e dispositivos móveis, muitos clientes optam por empregar uma camada de proxy entre a web ou o aplicativo móvel e o AWS AppConfig. Isso separa o volume de AWS AppConfig chamadas do tamanho da sua base de usuários, o que reduz os custos. Ele também permite que você aproveite o AWS AppConfig Agente, que otimiza o desempenho de recuperação de sinalizadores e oferece suporte a recursos como sinalizadores de várias variantes. AWS AppConfig recomenda usar AWS Lambda para criar o proxy. Em vez de recuperar sinalizadores diretamente de AWS AppConfig, configure a extensão AWS AppConfig Lambda para recuperar seus sinalizadores de recursos em uma função Lambda. Escreva a função para aceitar os parâmetros de AWS AppConfig recuperação da solicitação do evento e retornar os dados de configuração correspondentes na resposta do Lambda. Exponha seu proxy à Internet usando a função URLsLambda.

Depois de configurar seu proxy, considere a frequência com que você recupera os dados. Em geral, os casos de uso de dispositivos móveis não exigem intervalos de pesquisa de alta frequência. Configure o AWS AppConfig Agente para atualizar os dados com AWS AppConfig mais frequência do que o aplicativo atualiza a partir do proxy.

Autenticação e Amazon Cognito

A função Lambda URLs suporta duas formas de controle de acesso e. `AWS_IAM` `NONE` Use `NONE` se você preferir implementar a própria autenticação e autorização na sua função do Lambda. `NONE` também será a opção recomendada se seu caso de uso permitir expor seu endpoint ao público e se os dados de configuração não contiverem dados sensíveis. Para todos os outros casos de uso, use o `AWS_IAM`.

> **Important**
>
> Se você expor seu endpoint à Internet sem autenticação, certifique-se de que seus dados de configuração não vazem dados confidenciais, incluindo informações de identificação pessoal (PII), usuários IDs ou nomes de recursos não lançados.

Se você optar por usar o `AWS_IAM`, precisará gerenciar as credenciais com o Amazon Cognito. Para começar a usar o Amazon Cognito, crie um banco de identidades. Um banco de identidades permite que você envie credenciais de curto prazo ao seu aplicativo para usuários autenticados ou convidados. Você precisará adicionar funções no banco de identidades que permitam que as pessoas usem `InvokeFunctionUrl` para sua função do Lambda. Ao fazer isso, você permite que as instâncias do seu aplicativo acessem as credenciais necessárias para recuperar seus dados de configuração.

Ao trabalhar com o Amazon Cognito no seu aplicativo, considere usar AWS Amplify. O Amplify simplifica as interações entre mobile/web aplicativos AWS e fornece suporte integrado para o Amazon Cognito.

Armazenamento em cache

Ao usar AWS AppConfig, você deve sempre armazenar seus dados de configuração em cache localmente no dispositivo ou no navegador. O armazenamento em cache oferece as seguintes vantagens:

- Melhora o desempenho reduzindo a latência e o consumo de bateria

- Oferece estabilidade ao eliminar dependências de acesso à rede
- Diminui os custos ao reduzir a frequência de recuperação de dados

Em relação a casos de uso de dispositivos móveis, recomendamos que você implemente caches na memória e persistentes no dispositivo. Configure seu aplicativo para tentar recuperar a configuração desejada do cache em memória e voltar a buscar no proxy, se necessário. Após a recuperação bem-sucedida do seu proxy, atualize o cache em memória e, em seguida, mantenha a configuração no dispositivo. Use um processo em segundo plano para iterar o cache e atualizar cada configuração. Ao buscar a configuração pela primeira vez após a inicialização do aplicativo, caso a recuperação não tenha sido bem-sucedida, passe para a configuração persistente (e use-a para alimentar o cache em memória).

Segmentação

Convém segmentar a experiência de sinalização de atributos em toda a sua base de clientes ao usar sinalizadores de atributos. Para fazer isso, adicione contexto para as chamadas de recuperação de sinalizadores e configure regras para retornar diversas variantes dos sinalizadores de atributos com base no contexto fornecido. Por exemplo, você pode ter uma variante de sinalização de atributos para usuários do iOS 18.X, uma variante para usuários do iOS 17.X e um sinalizador padrão para todas as outras versões do iOS. Com variantes, você pode configurar cada versão iOS do seu aplicativo para atingir a mesma configuração no mesmo ambiente, mas com base no contexto fornecido na chamada de recuperação (por exemplo, “version”: “i OS18 .1"), os dispositivos receberão a variante apropriada da configuração.

> **Note**
>
> Se você estiver usando variantes de sinalizadores de AWS AppConfig recursos para um caso de uso móvel, deverá usar o AWS AppConfig Agente e um proxy para recuperar os sinalizadores de recursos.

Se você optar por não usar o AWS AppConfig Agent para recuperar sinalizadores de recursos, poderá aproveitar AWS AppConfig ambientes para uma segmentação simples e de baixa cardinalidade. Um ambiente é um grupo de implantação lógico para suas segmentações. Além de particionar suas configurações para ambientes de desenvolvimento, teste e produção, você pode subdividir sua base de clientes criando ambientes específicos para dispositivos móveis, como tipo de dispositivo (tablet x smartphone) ou versões principais do sistema operacional. Com ambientes

separados, é possível implantar conjuntos iguais ou diferentes de dados de configuração para atender aos requisitos específicos da sua base de clientes.

Largura de banda (casos de uso em dispositivos móveis)

Em geral, tente manter o tamanho de cada conjunto de sinalizadores pequeno. Os casos de uso em dispositivos móveis tendem a envolver restrições de baixa largura de banda. Minimizar o tamanho dos seus dados será útil para manter uma experiência consistente em toda a sua base de usuários. Além disso, considere que, como os dispositivos móveis geralmente operam em ambientes com pouca e nenhuma largura de banda, o armazenamento em cache no dispositivo é essencial. O código do aplicativo que falha se nenhum dado de configuração puder ser recuperado também é essencial.

Casos de uso de sinalizadores adicionais

O poder dos sinalizadores de atributos vai além da conveniência do lançamento de recursos. Sinalizadores operacionais prolongados podem ser usados para melhorar a postura operacional do seu aplicativo. Por exemplo, você pode criar um botão de monitoramento de desempenho que emite métricas e dados de depuração adicionais durante um evento. Como alternativa, convém manter e ajustar as taxas de atualização do aplicativo para um segmento da sua base de clientes.

Recuperando dados de configuração sem AWS AppConfig o Agente

A forma recomendada de recuperar dados de configuração AWS AppConfig é usando o agente desenvolvido e gerenciado pela Amazon. AWS AppConfig Com o agente, você pode armazenar dados de configuração em cache localmente e consultar de forma assíncrona o serviço de plano de AWS AppConfig dados para obter atualizações. Esse caching/polling processo garante que seus dados de configuração estejam sempre disponíveis para seu aplicativo, minimizando a latência e o custo. Se preferir não usar o agente, você pode ligar para o público APIs diretamente do serviço de plano de AWS AppConfig dados.

O serviço de plano de dados usa duas ações de API StartConfigurationSessionGetLatestConfiguratione. O serviço do plano de dados também usa pontos finais separados do plano de AWS AppConfig controle.

> **Note**
>
> O serviço de plano de dados substitui o processo anterior de recuperação de dados de configuração usando a ação de API `GetConfiguration`. A API `GetConfiguration` está obsoleta.

Como funciona

Veja como funciona o processo de chamada direta AWS AppConfig APIs usando o serviço de plano de dados.

Seu aplicativo recupera os dados de configuração estabelecendo primeiro uma sessão de configuração usando a operação da StartConfigurationSessionAPI. O cliente da sua sessão então faz chamadas periódicas para GetLatestConfiguration para verificar e recuperar os dados mais recentes disponíveis.

Ao chamar `StartConfigurationSession`, seu código envia as seguintes informações:

- Identificadores (ID ou nome) de um AWS AppConfig aplicativo, ambiente e perfil de configuração que a sessão rastreia.
- (Opcional) O tempo mínimo que o cliente da sessão deve esperar entre chamadas para `GetLatestConfiguration`.

Em resposta, AWS AppConfig fornece um `InitialConfigurationToken` a ser entregue ao cliente da sessão e usado na primeira vez em que ele liga `GetLatestConfiguration` para essa sessão.

> **Important**
>
> Esse token só deve ser usado uma vez em sua primeira chamada para `GetLatestConfiguration`. Você deve usar o novo token na resposta `GetLatestConfiguration` (`NextPollConfigurationToken`) em cada chamada subseqüente para `GetLatestConfiguration`. Para oferecer suporte a casos de uso de pesquisas longas, os tokens são válidos por até 24 horas. Se uma chamada `GetLatestConfiguration` usar um token expirado, o sistema retornará `BadRequestException`.

Ao chamar `GetLatestConfiguration`, seu código de cliente envia o valor `ConfigurationToken` mais recente que ele tem e recebe em resposta:

- `NextPollConfigurationToken`: o valor `ConfigurationToken` a ser usado na próxima chamada para `GetLatestConfiguration`.
- `NextPollIntervalInSeconds`: o tempo que o cliente deve esperar até fazer sua próxima chamada para `GetLatestConfiguration`.
- A configuração: os dados mais recentes destinados à sessão. Este pode estar vazio se o cliente já tiver a versão mais recente da configuração.

Important

Observe as seguintes informações importantes:

- A StartConfigurationSessionAPI só deve ser chamada uma vez por aplicativo, ambiente, perfil de configuração e cliente para estabelecer uma sessão com o serviço. Isso geralmente é feito no startup do aplicativo ou imediatamente antes da primeira recuperação de uma configuração.
- Se sua configuração for implantada usando um `KmsKeyIdentifier`, sua solicitação para receber a configuração deverá incluir permissão para chamar `kms:Decrypt`. Para obter mais informações, consulte Descriptografar na Referência da API do AWS Key Management Service .
- A operação da API usada anteriormente para recuperar dados de configuração, `GetConfiguration`, está obsoleta. A operação `GetConfiguration` da API não é compatível com configurações criptografadas.

(Exemplo) Recuperando uma configuração chamando AWS AppConfig APIs

O AWS CLI exemplo a seguir demonstra como recuperar dados de configuração usando as operações de AWS AppConfig dados `StartConfigurationSession` e `GetLatestConfiguration` API. O primeiro comando inicia uma sessão de configuração. Essa chamada inclui o IDs (ou nomes) do AWS AppConfig aplicativo, do ambiente e do perfil de configuração. A API retorna um `InitialConfigurationToken` usado para buscar seus dados de configuração.

```
aws appconfigdata start-configuration-session \
    --application-identifier application_name_or_ID \
    --environment-identifier environment_name_or_ID \
    --configuration-profile-identifier configuration_profile_name_or_ID
```

O sistema responde com informações no seguinte formato.

```
{
   "InitialConfigurationToken": initial configuration token
}
```

Depois de iniciar uma sessão, use InitialConfigurationTokento call GetLatestConfigurationpara buscar seus dados de configuração. Os dados de configuração são salvos no arquivo `mydata.json`.

```
aws appconfigdata get-latest-configuration \
    --configuration-token initial configuration token mydata.json
```

A primeira chamada para `GetLatestConfiguration` usa o `ConfigurationToken` obtido de `StartConfigurationSession`. As informações a seguir são retornadas.

```
{
    "NextPollConfigurationToken" : next configuration token,
    "ContentType" : content type of configuration,
    "NextPollIntervalInSeconds" : 60
}
```

As chamadas subsequentes para `GetLatestConfiguration` devem fornecer `NextPollConfigurationToken` da resposta anterior.

```
aws appconfigdata get-latest-configuration \
    --configuration-token next configuration token mydata.json
```

Important

Observe os detalhes importantes a seguir sobre a operação `GetLatestConfiguration` da API:

- A resposta `GetLatestConfiguration` inclui uma seção `Configuration` que mostra os dados de configuração. A seção `Configuration` só aparece se o sistema encontrar

dados de configuração novos ou atualizados. Se o sistema não encontrar dados de configuração novos ou atualizados, os dados de `Configuration` estarão vazios.

- Você recebe um novo `ConfigurationToken` a cada resposta de `GetLatestConfiguration`.
- Recomendamos ajustar a frequência de pesquisas das chamadas de API `GetLatestConfiguration` com base no seu orçamento, na frequência esperada de implantações de configuração e no número de destinos para uma configuração.

Estendendo AWS AppConfig fluxos de trabalho usando extensões

Uma extensão aumenta sua capacidade de injetar lógica ou comportamento em diferentes pontos durante o AWS AppConfig fluxo de trabalho de criação ou implantação de uma configuração. Por exemplo, você pode usar extensões para realizar os seguintes tipos de tarefas (para citar algumas):

- Enviar uma notificação para um tópico do Amazon Simple Notification Service (Amazon SNS) quando um perfil de configuração for implantado.
- Limpar o conteúdo de um perfil de configuração em busca de dados confidenciais antes do início da implantação.
- Criar ou atualizar uma ocorrência do Atlassian Jira sempre que uma alteração for feita em um sinalizador de atributos.
- Mesclar o conteúdo de um serviço ou fonte de dados com seus dados de configuração ao iniciar uma implantação.
- Fazer backup de uma configuração em um bucket do Amazon Simple Storage Service (Amazon S3) sempre que uma configuração for implantada.

Você pode associar esses tipos de tarefas a AWS AppConfig aplicativos, ambientes e perfis de configuração.

Conteúdo

- Entendendo AWS AppConfig as extensões
- Trabalhando com extensões criadas AWS por autoria
- Passo a passo: Criação de extensões personalizadas AWS AppConfig

Entendendo AWS AppConfig as extensões

Este tópico apresenta conceitos e terminologia de AWS AppConfig extensão. As informações são discutidas no contexto de cada etapa necessária para configurar e usar AWS AppConfig extensões.

Tópicos

- Etapa 1: determine o que você deseja fazer com as extensões

- Etapa 2: determine quando você deseja que a extensão seja executada
- Etapa 3: crie uma associação de extensão
- Etapa 4: implante uma configuração e verifique se as ações da extensão são executadas

Etapa 1: determine o que você deseja fazer com as extensões

Você quer receber uma notificação para um webhook que envia mensagens para o Slack sempre que uma AWS AppConfig implantação for concluída? Deseja fazer backup de um perfil de configuração em um bucket do Amazon Simple Storage Service (Amazon S3) antes de uma configuração ser implantada? Deseja limpar os dados de configuração em busca de informações confidenciais antes da implantação da configuração? Você pode usar extensões para realizar esses tipos de tarefas e muito mais. Você pode criar extensões personalizadas ou usar as AWS extensões criadas incluídas com. AWS AppConfig

> **Note**
>
> Para a maioria dos casos de uso, para criar uma extensão personalizada, você deve criar uma AWS Lambda função para realizar qualquer computação e processamento definidos na extensão. Para obter mais informações, consulte Passo a passo: Criação de extensões personalizadas AWS AppConfig.

As extensões AWS criadas a seguir podem ajudá-lo a integrar rapidamente as implantações de configuração com outros serviços. Você pode usar essas extensões no AWS AppConfig console ou chamando ações da API de extensão diretamente do AWS CLI, Ferramentas da AWS para PowerShell, ou do SDK.

Extensão	Description
AWS AppConfig eventos de implantação para EventBridge	Essa extensão envia eventos para o barrament o de eventos EventBridge padrão quando uma configuração é implantada.
AWS AppConfig eventos de implantação no Amazon Simple Notification Service (Amazon SNS)	Esta extensão envia mensagens para um tópico do Amazon SNS que você especifica quando uma configuração é implantada.

Extensão	Description
AWS AppConfig eventos de implantação no Amazon Simple Queue Service (Amazon SQS)	Esta extensão enfileira mensagens em sua fila do Amazon SQS quando uma configuração é implantada.
Extensão de integração — Atlassian Jira	Essas extensões permitem AWS AppConfig criar e atualizar problemas sempre que você fizer alterações em um sinalizador de recurso.

Etapa 2: determine quando você deseja que a extensão seja executada

Uma extensão define uma ou mais ações que ela executa durante um AWS AppConfig fluxo de trabalho. Por exemplo, a `AWS AppConfig deployment events to Amazon SNS` extensão AWS criada inclui uma ação para enviar uma notificação para um tópico do Amazon SNS. Cada ação é invocada quando você interage com AWS AppConfig ou quando AWS AppConfig está executando um processo em seu nome. Eles são chamados de pontos de ação. AWS AppConfig as extensões suportam os seguintes pontos de ação:

Pontos de ação PRE_*: as ações de extensão configuradas em pontos de ação `PRE_*` são aplicadas após a validação da solicitação, mas antes de o AWS AppConfig realizar a atividade que corresponde ao nome do ponto de ação. Essas invocações de ação são processadas ao mesmo tempo da solicitação. Se mais de uma solicitação for feita, as invocações de ação serão executadas sequencialmente. Observe também que os pontos de ação `PRE_*` recebem e podem alterar o conteúdo de uma configuração. Os pontos de ação `PRE_*` também podem responder a um erro e impedir que uma ação aconteça.

- `PRE_CREATE_HOSTED_CONFIGURATION_VERSION`
- `PRE_START_DEPLOYMENT`

Pontos de ação ON_*: uma extensão também pode ser executada paralelamente a um AWS AppConfig fluxo de trabalho usando um ponto de `ON_*` ação. `ON_*`os pontos de ação são invocados de forma assíncrona. `ON_*`os pontos de ação não recebem o conteúdo de uma configuração. Se uma extensão apresentar um erro durante um ponto de ação `ON_*`, o serviço ignorará o erro e continuará o fluxo de trabalho.

- `ON_DEPLOYMENT_START`

- ON_DEPLOYMENT_STEP
- ON_DEPLOYMENT_BAKING
- ON_DEPLOYMENT_COMPLETE
- ON_DEPLOYMENT_ROLLED_BACK

Pontos de ação AT_*: as ações de extensão configuradas nos pontos de ação AT_* são invocadas de maneira síncrona e em paralelo com um fluxo de trabalho do AWS AppConfig . Se uma extensão apresentar um erro durante um ponto de ação AT_*, o serviço interromperá o fluxo de trabalho e reverterá a implantação.

- AT_DEPLOYMENT_TICK

Etapa 3: crie uma associação de extensão

Para criar uma extensão ou configurar uma extensão de AWS autoria, você define os pontos de ação que invocam uma extensão quando um AWS AppConfig recurso específico é usado. Por exemplo, você pode optar por executar a extensão AWS AppConfig deployment events to Amazon SNS e receber notificações sobre um tópico do Amazon SNS sempre que uma implantação de configuração for iniciada para um aplicativo específico. Definir quais pontos de ação invocam uma extensão para um AWS AppConfig recurso específico é chamado de associação de extensão. Uma associação de extensão é uma relação especificada entre uma extensão e um AWS AppConfig recurso, como um aplicativo ou um perfil de configuração.

Um único AWS AppConfig aplicativo pode incluir vários ambientes e perfis de configuração. Se você associar uma extensão a um aplicativo ou ambiente, AWS AppConfig invoca a extensão para qualquer fluxo de trabalho relacionado ao aplicativo ou aos recursos do ambiente, se aplicável.

Por exemplo, digamos que você tenha um AWS AppConfig aplicativo chamado MobileApps que inclui um perfil de configuração chamado AccessList. E digamos que o MobileApps aplicativo inclua ambientes beta, de integração e de produção. Você cria uma associação de extensão para a extensão AWS de notificação criada pelo Amazon SNS e associa a extensão ao aplicativo. MobileApps A extensão de notificação do Amazon SNS é invocada sempre que a configuração é implantada para o aplicativo em qualquer um dos três ambientes.

> **Note**
>
> Você não precisa criar uma extensão para usar extensões de AWS autoria, mas precisa criar uma associação de extensão.

Etapa 4: implante uma configuração e verifique se as ações da extensão são executadas

Depois de criar uma associação, quando uma configuração hospedada é criada ou uma configuração é implantada, AWS AppConfig invoca a extensão e executa as ações especificadas. Quando uma extensão é invocada, se o sistema apresentar um erro durante um ponto de `PRE-*` ação, AWS AppConfig retornará informações sobre esse erro.

Trabalhando com extensões criadas AWS por autoria

AWS AppConfig inclui as seguintes extensões de AWS autoria. Essas extensões podem ajudar você a integrar o AWS AppConfig fluxo de trabalho com outros serviços. Você pode usar essas extensões no Console de gerenciamento da AWS ou chamando ações da API de extensão diretamente do AWS CLI, Ferramentas da AWS para PowerShell, ou do SDK.

Extensão	Description
AWS AppConfig eventos de implantação para EventBridge	Essa extensão envia eventos para o barramento de eventos EventBridge padrão quando uma configuração é implantada.
AWS AppConfig eventos de implantação no Amazon Simple Notification Service (Amazon SNS)	Esta extensão envia mensagens para um tópico do Amazon SNS que você especifica quando uma configuração é implantada.
AWS AppConfig eventos de implantação no Amazon Simple Queue Service (Amazon SQS)	Esta extensão enfileira mensagens em sua fila do Amazon SQS quando uma configuração é implantada.

Extensão	Description
Extensão de integração — Atlassian Jira	Essas extensões permitem AWS AppConfig criar e atualizar problemas sempre que você fizer alterações em um sinalizador de recurso.

Usando os eventos AWS AppConfig de implantação na EventBridge extensão Amazon

A `AWS AppConfig deployment events to Amazon EventBridge` extensão é uma extensão AWS criada que ajuda você a monitorar e agir no fluxo de trabalho de implantação da AWS AppConfig configuração. A extensão envia notificações de eventos para o barramento de eventos EventBridge padrão sempre que uma configuração é implantada. Depois de associar a extensão a um de seus AWS AppConfig aplicativos, ambientes ou perfis de configuração, AWS AppConfig envia notificações de eventos para o barramento de eventos após o início, término e reversão de cada implantação da configuração.

Se você quiser ter mais controle sobre quais pontos de ação enviam EventBridge notificações, você pode criar uma extensão personalizada e inserir o Amazon Resource Name (ARN) do barramento de eventos EventBridge padrão para o campo URI. Para obter mais informações sobre como criar uma extensão, consulte Passo a passo: Criação de extensões personalizadas AWS AppConfig.

> **Important**
>
> Essa extensão suporta somente o barramento de eventos EventBridge padrão.

Usar a extensão

Para usar a `AWS AppConfig deployment events to Amazon EventBridge` extensão, primeiro você anexa a extensão a um de seus AWS AppConfig recursos criando uma associação de extensão. Você cria a associação usando o AWS AppConfig console ou a ação CreateExtensionAssociationda API. Ao criar a associação, você especifica o ARN de um AWS AppConfig aplicativo, ambiente ou perfil de configuração. Se você associar a extensão a um aplicativo ou ambiente, uma notificação de evento será enviada para qualquer perfil de configuração contido no aplicativo ou ambiente especificado.

Depois de criar a associação, quando uma configuração para o AWS AppConfig recurso especificado é implantada, AWS AppConfig invoca a extensão e envia notificações de acordo com os pontos de ação especificados na extensão.

> **Note**
>
> Esta extensão é invocada pelos seguintes pontos de ação:
>
> - `ON_DEPLOYMENT_START`
> - `ON_DEPLOYMENT_COMPLETE`
> - `ON_DEPLOYMENT_ROLLED_BACK`
>
> Não é possível personalizar os pontos de ação desta extensão. Para invocar pontos de ação diferentes, você pode criar sua própria extensão. Para obter mais informações, consulte Passo a passo: Criação de extensões personalizadas AWS AppConfig.

Use os procedimentos a seguir para criar uma associação de AWS AppConfig extensão usando o AWS Systems Manager console ou AWS CLI o.

Para criar uma associação de extensão (console)

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, escolha AWS AppConfig.
3. Na guia Extensões, escolha Adicionar ao recurso.
4. Na seção Detalhes do recurso de extensão, em Tipo de recurso, escolha um tipo de AWS AppConfig recurso. Dependendo do recurso escolhido, AWS AppConfig solicita que você escolha outros recursos.
5. Escolha Criar associação ao recurso.

Aqui está um exemplo de evento enviado para EventBridge quando a extensão é invocada.

```
{
   "version":"0",
   "id":"c53dbd72-c1a0-2302-9ed6-c076e9128277",
   "detail-type":"On Deployment Complete",
```

```
   "source":"aws.appconfig",
   "account":"111122223333",
   "time":"2022-07-09T01:44:15Z",
   "region":"us-east-1",
   "resources":[
      "arn:aws:appconfig:us-east-1:111122223333:extensionassociation/z763ff5"
   ],
   "detail":{
      "InvocationId":"5tfjcig",
       "Parameters":{

      },
      "Type":"OnDeploymentComplete",
      "Application":{
         "Id":"ba8toh7",
         "Name":"MyApp"
      },
      "Environment":{
         "Id":"pgil2o7",
         "Name":"MyEnv"
      },
      "ConfigurationProfile":{
         "Id":"ga3tqep",
         "Name":"MyConfigProfile"
      },
      "DeploymentNumber":1,
      "ConfigurationVersion":"1"
   }
}
```

Usando os eventos AWS AppConfig de implantação na extensão Amazon SNS

A `AWS AppConfig deployment events to Amazon SNS` extensão é uma extensão AWS criada que ajuda você a monitorar e agir no fluxo de trabalho de implantação da AWS AppConfig configuração. A extensão publica mensagens em um tópico do Amazon SNS sempre que uma configuração é implantada. Depois de associar a extensão a um de seus AWS AppConfig aplicativos, ambientes ou perfis de configuração, AWS AppConfig publica uma mensagem no tópico após cada início, término e reversão da implantação da configuração.

Se quiser ter mais controle sobre quais pontos de ação enviam notificações do Amazon SNS, você poderá criar uma extensão personalizada e inserir o nome do recurso da Amazon (ARN) de um

tópico do Amazon SNS no campo URI. Para obter mais informações sobre como criar uma extensão, consulte Passo a passo: Criação de extensões personalizadas AWS AppConfig.

Como usar a extensão

Esta seção descreve como usar a extensão `AWS AppConfig deployment events to Amazon SNS`.

Etapa 1: Configurar AWS AppConfig para publicar mensagens em um tópico

Adicione uma política de controle de acesso ao seu tópico do Amazon SNS concedendo ao AWS AppConfig (`appconfig.amazonaws.com`) permissões de publicação (`sns:Publish`). Para obter mais informações, consulte Casos de exemplo para controle de acesso do Amazon SNS.

Etapa 2: crie uma associação de extensão

Anexe a extensão a um de seus AWS AppConfig recursos criando uma associação de extensão. Você cria a associação usando o AWS AppConfig console ou a ação CreateExtensionAssociationda API. Ao criar a associação, você especifica o ARN de um AWS AppConfig aplicativo, ambiente ou perfil de configuração. Se você associar a extensão a um aplicativo ou ambiente, uma notificação será enviada para qualquer perfil de configuração contido no aplicativo ou ambiente especificado. Ao criar a associação, você deve inserir um valor para o parâmetro `topicArn` que contém o ARN do tópico do Amazon SNS que você deseja usar.

Depois de criar a associação, quando uma configuração para o AWS AppConfig recurso especificado é implantada, AWS AppConfig invoca a extensão e envia notificações de acordo com os pontos de ação especificados na extensão.

> **Note**
>
> Esta extensão é invocada pelos seguintes pontos de ação:
>
> - `ON_DEPLOYMENT_START`
> - `ON_DEPLOYMENT_COMPLETE`
> - `ON_DEPLOYMENT_ROLLED_BACK`
>
> Não é possível personalizar os pontos de ação desta extensão. Para invocar pontos de ação diferentes, você pode criar sua própria extensão. Para obter mais informações, consulte Passo a passo: Criação de extensões personalizadas AWS AppConfig.

Use os procedimentos a seguir para criar uma associação de AWS AppConfig extensão usando o AWS Systems Manager console ou AWS CLI o.

Para criar uma associação de extensão (console)

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, escolha AWS AppConfig.
3. Na guia Extensões, escolha Adicionar ao recurso.
4. Na seção Detalhes do recurso de extensão, em Tipo de recurso, escolha um tipo de AWS AppConfig recurso. Dependendo do recurso escolhido, AWS AppConfig solicita que você escolha outros recursos.
5. Escolha Criar associação ao recurso.

Veja uma amostra da mensagem enviada para o tópico do Amazon SNS quando a extensão é invocada.

```
{
    "Type": "Notification",
    "MessageId": "ae9d702f-9a66-51b3-8586-2b17932a9f28",
    "TopicArn": "arn:aws:sns:us-east-1:111122223333:MySNSTopic",
    "Message": {
        "InvocationId": "7itcaxp",
        "Parameters": {
            "topicArn": "arn:aws:sns:us-east-1:111122223333:MySNSTopic"
        },
        "Application": {
            "Id": "1a2b3c4d",
            "Name": MyApp
        },
        "Environment": {
            "Id": "1a2b3c4d",
            "Name": MyEnv
        },
        "ConfigurationProfile": {
            "Id": "1a2b3c4d",
            "Name": "MyConfigProfile"
        },
        "Description": null,
        "DeploymentNumber": "3",
```

```
        "ConfigurationVersion": "1",
        "Type": "OnDeploymentComplete"
    },
    "Timestamp": "2022-06-30T20:26:52.067Z",
    "SignatureVersion": "1",
    "Signature": "<...>",
    "SigningCertURL": "<...>",
    "UnsubscribeURL": "<...>",
    "MessageAttributes": {
        "MessageType": {
            "Type": "String",
            "Value": "OnDeploymentStart"
        }
    }
}
```

Usando os eventos AWS AppConfig de implantação na extensão Amazon SQS

A `AWS AppConfig deployment events to Amazon SQS` extensão é uma extensão AWS criada que ajuda você a monitorar e agir no fluxo de trabalho de implantação da AWS AppConfig configuração. A extensão enfileira as mensagens em sua fila do Amazon Simple Queue Service (Amazon SQS) sempre que uma configuração é implantada. Depois de associar a extensão a um de seus AWS AppConfig aplicativos, ambientes ou perfis de configuração, coloca uma mensagem na AWS AppConfig fila após cada início, término e reversão de implantação da configuração.

Se quiser ter mais controle sobre quais pontos de ação enviam notificações do Amazon SQS, você pode criar uma extensão personalizada e inserir um nome do recurso da Amazon (ARN) da fila do Amazon SQS no campo URI. Para obter mais informações sobre como criar uma extensão, consulte Passo a passo: Criação de extensões personalizadas AWS AppConfig.

Como usar a extensão

Esta seção descreve como usar a extensão `AWS AppConfig deployment events to Amazon SQS`.

Etapa 1: Configurar AWS AppConfig para enfileirar mensagens

Adicione uma política do Amazon SQS à sua fila do Amazon SQS concedendo ao AWS AppConfig (`appconfig.amazonaws.com`) permissões de envio de mensagens (`sqs:SendMessage`). Para obter mais informações, consulte Exemplos básicos de políticas do Amazon SQS.

Etapa 2: crie uma associação de extensão

Anexe a extensão a um de seus AWS AppConfig recursos criando uma associação de extensão. Você cria a associação usando o AWS AppConfig console ou a ação CreateExtensionAssociationda API. Ao criar a associação, você especifica o ARN de um AWS AppConfig aplicativo, ambiente ou perfil de configuração. Se você associar a extensão a um aplicativo ou ambiente, uma notificação será enviada para qualquer perfil de configuração contido no aplicativo ou ambiente especificado. Ao criar a associação, você deve inserir um parâmetro `Here` que contenha o ARN da fila do Amazon SQS que você deseja usar.

Depois de criar a associação, quando uma configuração para o AWS AppConfig recurso especificado é criada ou implantada, AWS AppConfig invoca a extensão e envia notificações de acordo com os pontos de ação especificados na extensão.

> **Note**
>
> Esta extensão é invocada pelos seguintes pontos de ação:
>
> - `ON_DEPLOYMENT_START`
> - `ON_DEPLOYMENT_COMPLETE`
> - `ON_DEPLOYMENT_ROLLED_BACK`
>
> Não é possível personalizar os pontos de ação desta extensão. Para invocar pontos de ação diferentes, você pode criar sua própria extensão. Para obter mais informações, consulte Passo a passo: Criação de extensões personalizadas AWS AppConfig.

Use os procedimentos a seguir para criar uma associação de AWS AppConfig extensão usando o AWS Systems Manager console ou AWS CLI o.

Para criar uma associação de extensão (console)

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, escolha AWS AppConfig.
3. Na guia Extensões, escolha Adicionar ao recurso.

4. Na seção Detalhes do recurso de extensão, em Tipo de recurso, escolha um tipo de AWS AppConfig recurso. Dependendo do recurso escolhido, AWS AppConfig solicita que você escolha outros recursos.
5. Escolha Criar associação ao recurso.

Veja um exemplo da mensagem enviada para a fila do Amazon SQS quando a extensão é invocada.

```
{
   "InvocationId":"7itcaxp",
   "Parameters":{
      "queueArn":"arn:aws:sqs:us-east-1:111122223333:MySQSQueue"
   },
   "Application":{
      "Id":"1a2b3c4d",
      "Name":MyApp
   },
   "Environment":{
      "Id":"1a2b3c4d",
      "Name":MyEnv
   },
   "ConfigurationProfile":{
      "Id":"1a2b3c4d",
      "Name":"MyConfigProfile"
   },
   "Description":null,
   "DeploymentNumber":"3",
   "ConfigurationVersion":"1",
   "Type":"OnDeploymentComplete"
}
```

Usando a extensão Atlassian Jira para AWS AppConfig

Ao se integrar com o Atlassian Jira, você AWS AppConfig pode criar e atualizar problemas no console Atlassian sempre que você fizer alterações em uma bandeira de recurso em seu para o especificado. Conta da AWS Região da AWS Cada problema do Jira inclui o nome do sinalizador, o ID do aplicativo, o ID do perfil de configuração e os valores do sinalizador. Depois de atualizar, salvar e implantar as alterações do seu sinalizador, o Jira atualiza os problemas existentes com os detalhes da alteração.

> **Note**
>
> O Jira atualiza problemas sempre que você cria ou atualiza um sinalizador de atributos. O Jira também atualiza problemas quando você exclui um atributo de um sinalizador de nível filho de um sinalizador de nível pai. O Jira não registra informações quando você exclui um sinalizador de nível pai.

Para configurar a integração, você deve fazer o seguinte:

- Configurando permissões para integração com o AWS AppConfig Jira
- Configurando o aplicativo de integração do AWS AppConfig Jira

Configurando permissões para integração com o AWS AppConfig Jira

Ao configurar a AWS AppConfig integração com o Jira, você especifica as credenciais de um usuário. Especificamente, você insere o ID da chave de acesso e a chave secreta do usuário no aplicativo AWS AppConfig para Jira. Esse usuário dá permissão ao Jira para se comunicar com AWS AppConfig. AWS AppConfig usa essas credenciais uma vez para estabelecer uma associação entre AWS AppConfig e o Jira. As credenciais não são armazenadas. Você pode remover a associação desinstalando o aplicativo AWS AppConfig for Jira.

A conta do usuário exige uma política de permissão que inclua as seguintes ações:

- `appconfig:CreateExtensionAssociation`
- `appconfig:GetConfigurationProfile`
- `appconfig:ListApplications`
- `appconfig:ListConfigurationProfiles`
- `appconfig:ListExtensionAssociations`
- `sts:GetCallerIdentity`

Execute as seguintes tarefas para criar uma política de permissão do IAM e um usuário para a integração do AWS AppConfig com o Jira:

Tarefas

- Tarefa 1: criar uma política de permissão do IAM para a integração AWS AppConfig com o Jira
- Tarefa 2: Criar um usuário para a integração AWS AppConfig com o Jira

Tarefa 1: criar uma política de permissão do IAM para a integração AWS AppConfig com o Jira

Use o procedimento a seguir para criar uma política de permissão do IAM que permita que o Atlassian Jira se comunique com. AWS AppConfig Recomendamos criar uma política e associar essa política a um novo perfil do IAM. A adição da permissão necessária a uma política e um perfil do IAM existentes viola o princípio do privilégio mínimo e não é recomendada.

Para criar uma política do IAM AWS AppConfig e a integração com o Jira

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, selecione Políticas e, em seguida, Criar política.
3. Na página Criar política, selecione a guia JSON e substitua o conteúdo padrão pela política a seguir. Na política a seguir, substitua*Region*, *account_IDapplication_ID*, e *configuration_profile_ID* por informações do seu ambiente de sinalização de AWS AppConfig recursos.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "appconfig:CreateExtensionAssociation",
                "appconfig:ListExtensionAssociations",
                "appconfig:GetConfigurationProfile"
            ],
            "Resource": [
                "arn:aws:appconfig:us-
east-1:111122223333:application/application_ID",
                "arn:aws:appconfig:us-
east-1:111122223333:application/application_ID/
configurationprofile/configuration_profile_ID"
            ]
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "appconfig:ListApplications"
            ],
            "Resource": [
                "arn:aws:appconfig:us-east-1:111122223333:*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "appconfig:ListConfigurationProfiles"
            ],
            "Resource": [
                "arn:aws:appconfig:us-
east-1:111122223333:application/application_ID"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "sts:GetCallerIdentity",
            "Resource": "*"
        }
    ]
}
```

4. Escolha Próximo: tags.
5. (Opcional) Adicione um ou mais pares de chave-valor de tag para organizar, monitorar ou controlar o acesso para esta função e selecione Next: Review (Próximo: revisar).
6. Na página Review policy (Revisar política), insira um nome na caixa Name (Nome), como **`AppConfigJiraPolicy`**, e insira uma descrição opcional.
7. Selecione Criar política.

Tarefa 2: Criar um usuário para a integração AWS AppConfig com o Jira

Use o procedimento a seguir para criar um usuário para AWS AppConfig a integração com o Atlassian Jira. Depois de criar o usuário, você poderá copiar o ID da chave de acesso e a chave secreta, que serão especificados ao concluir a integração.

Para criar um usuário para a integração AWS AppConfig com o Jira

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Users (Usuários) e Add users (Adicionar usuários).
3. No campo Nome do usuário, insira um nome, como **`AppConfigJiraUser`**.
4. Em Selecionar tipo de AWS credencial, escolha Chave de acesso - Acesso programático.
5. Escolha Próximo: Permissões.
6. Na página Definir permissões, selecione Anexar políticas existentes diretamente. Pesquise a política que você criou e marque a caixa de seleção correspondente em Tarefa 1: criar uma política de permissão do IAM para a integração AWS AppConfig com o Jira e escolha Próximo: Tags.
7. Na página Adicionar tags (opcional), adicione um ou mais pares de chave-valor de tags para organizar, monitorar ou controlar acesso para esse usuário. Escolha Próximo: revisar.
8. Na página Revisar, verifique os detalhes do usuário.
9. Selecione Criar usuário. O sistema exibe o ID da chave de acesso e a chave secreta do usuário. Baixe o arquivo .csv ou copie essas credenciais em um local separado. Você especificará essas credenciais ao configurar a integração.

Configurando o aplicativo de integração do AWS AppConfig Jira

Use o procedimento a seguir para configurar as opções necessárias no aplicativo AWS AppConfig for Jira. Depois de concluir esse procedimento, o Jira cria um novo problema para cada sinalizador de recurso no seu Conta da AWS para o especificado Região da AWS. Se você fizer alterações em um sinalizador de recurso AWS AppConfig, o Jira registrará os detalhes nas edições existentes.

> **Note**
>
> Um sinalizador de AWS AppConfig recurso pode incluir vários atributos de sinalizador de nível infantil. O Jira cria um problema para cada sinalizador de atributos de nível pai. Se alterar um atributo de sinalizador de nível filho, você poderá ver os detalhes dessa alteração no problema do Jira para o sinalizador de nível pai.

Para configurar a integração

1. Faça login no Atlassian Marketplace.

2. No campo de pesquisa, digite **`AWS AppConfig`** e pressione Enter.
3. Instale o aplicativo na sua instância do Jira.
4. No console Atlassian, escolha Gerenciar aplicativos e, em seguida, escolha AWS AppConfig para Jira.
5. Selecione Configurar.
6. Em Detalhes da configuração, escolha Projeto Jira e, em seguida, escolha o projeto que você deseja associar ao seu sinalizador de atributos do AWS AppConfig .
7. Escolha Região da AWS e, em seguida, escolha a região em que seu sinalizador de atributos do AWS AppConfig está localizado.
8. No campo ID do aplicativo, insira o nome do aplicativo do AWS AppConfig que contém seu sinalizador de atributos.
9. No campo ID do perfil de configuração, insira o nome do perfil de configuração do AWS AppConfig do seu sinalizador de atributos.
10. Nos campos ID da chave de acesso e Chave secreta, insira as credenciais que você copiou em Tarefa 2: Criar um usuário para a integração AWS AppConfig com o Jira. Como opção, especifique também um token de sessão.
11. Selecione Enviar.
12. No console da Atlassian, escolha Projetos e, em seguida, escolha o projeto que você selecionou para AWS AppConfig integração. A página Problemas exibe um problema para cada sinalizador de recurso no especificado Conta da AWS Região da AWS e.

Excluindo o aplicativo e AWS AppConfig os dados do for Jira

Se você não quiser mais usar a integração do Jira com sinalizadores de AWS AppConfig recursos, você pode excluir o aplicativo AWS AppConfig for Jira no console da Atlassian. A exclusão do aplicativo de integração faz o seguinte:

- Exclui a associação entre sua instância do Jira e AWS AppConfig
- Exclui os detalhes da sua instância do Jira de AWS AppConfig

Para excluir o aplicativo AWS AppConfig for Jira

1. No console Atlassian, escolha Gerenciar aplicativos.
2. Escolha AWS AppConfig para Jira.

3. Clique em Desinstalar.

Passo a passo: Criação de extensões personalizadas AWS AppConfig

Para criar uma AWS AppConfig extensão personalizada, conclua as tarefas a seguir. Cada tarefa é descrita em mais detalhes nos tópicos mais adiante.

> **Note**
>
> Você pode ver exemplos de AWS AppConfig extensões personalizadas em GitHub:
>
> - Extensão de exemplo que impede implantações com um calendário de moratória `blocked day` usando o calendário de alterações do Systems Manager
> - Extensão de exemplo que impede o vazamento de segredos em dados de configuração usando a ferramenta git-secrets
> - Extensão de exemplo que impede o vazamento de informações de identificação pessoal (PII) em dados de configuração usando o Amazon Comprehend

1. Crie uma AWS Lambda função

   Para a maioria dos casos de uso, para criar uma extensão personalizada, você deve criar uma AWS Lambda função para realizar qualquer computação e processamento definidos na extensão. Uma exceção a essa regra é se você criar versões personalizadas das extensões de notificação criadas pela AWS para adicionar ou remover pontos de ação. Para obter mais detalhes sobre esta exceção, consulte Etapa 3: criar uma AWS AppConfig extensão personalizada.

2. Configure permissões para sua extensão personalizada

   Para configurar permissões para sua extensão personalizada, você pode realizar um dos seguintes procedimentos:

   - Crie uma função de serviço AWS Identity and Access Management (IAM) que inclua `InvokeFunction` permissões.
   - Crie uma política de recursos usando a ação da AddPermissionAPI Lambda.

   Este passo a passo descreve como criar o perfil de serviço do IAM.

3. Crie uma extensão

   Você pode criar uma extensão usando o AWS AppConfig console ou chamando a ação da CreateExtensionAPI do AWS CLI, Ferramentas da AWS para PowerShell, ou do SDK. O passo a passo usa o console.

4. Crie uma associação de extensão

   Você pode criar uma associação de extensão usando o AWS AppConfig console ou chamando a ação da CreateExtensionAssociationAPI do AWS CLI, Ferramentas da AWS para PowerShell, ou do SDK. O passo a passo usa o console.

5. Execute uma ação que invoque a extensão

   Depois de criar a associação, AWS AppConfig invoca a extensão quando os pontos de ação definidos pela extensão ocorrerem para esse recurso. Por exemplo, se você associar uma extensão que contém uma ação `PRE_CREATE_HOSTED_CONFIGURATION_VERSION`, a extensão será invocada toda vez que você criar uma nova versão de configuração hospedada.

Os tópicos nesta seção descrevem cada tarefa envolvida na criação de uma extensão do AWS AppConfig personalizada. Cada tarefa é descrita no contexto de um caso de uso em que o cliente deseja criar uma extensão que faz backup automático de uma configuração em um bucket do Amazon Simple Storage Service (Amazon S3). A extensão é executada sempre que uma configuração hospedada é criada (`PRE_CREATE_HOSTED_CONFIGURATION_VERSION`) ou implantada (`PRE_START_DEPLOYMENT`).

Tópicos

- Etapa 1: criar uma função Lambda para uma extensão personalizada AWS AppConfig
- Etapa 2: configurar permissões para uma AWS AppConfig extensão personalizada
- Etapa 3: criar uma AWS AppConfig extensão personalizada
- Etapa 4: criar uma associação de extensão para uma AWS AppConfig extensão personalizada

Etapa 1: criar uma função Lambda para uma extensão personalizada AWS AppConfig

Para a maioria dos casos de uso, para criar uma extensão personalizada, você deve criar uma AWS Lambda função para realizar qualquer computação e processamento definidos na extensão. Esta seção inclui um exemplo de código da função Lambda para uma extensão personalizada

AWS AppConfig . Esta seção inclui também detalhes de referência de solicitações e respostas de carga útil. Para obter mais informações sobre como criar uma função do Lambda, consulte a seção Conceitos básicos do Lambda no Guia do desenvolvedor do AWS Lambda .

Código de exemplo

O código de exemplo a seguir para uma função Lambda, quando invocado, faz backup automático de uma AWS AppConfig configuração em um bucket do Amazon S3. O backup da configuração é feito sempre que uma nova configuração é criada ou implantada. O exemplo usa parâmetros de extensão para que o nome do bucket não precise ser codificado na função do Lambda. Usando parâmetros de extensão, o usuário pode anexar a extensão a vários aplicativos e fazer backup das configurações em diferentes buckets. O exemplo de código inclui comentários para explicar melhor a função.

Exemplo de função Lambda para uma extensão AWS AppConfig

```
from datetime import datetime
import base64
import json

import boto3


def lambda_handler(event, context):
    print(event)

    # Extensions that use the PRE_CREATE_HOSTED_CONFIGURATION_VERSION and
 PRE_START_DEPLOYMENT
    # action points receive the contents of AWS AppConfig configurations in Lambda
 event parameters.
    # Configuration contents are received as a base64-encoded string, which the lambda
 needs to decode
    # in order to get the configuration data as bytes. For other action points, the
 content
    # of the configuration isn't present, so the code below will fail.
    config_data_bytes = base64.b64decode(event["Content"])

    # You can specify parameters for extensions. The CreateExtension API action lets
 you define
    # which parameters an extension supports. You supply the values for those
 parameters when you
```

```
    # create an extension association by calling the CreateExtensionAssociation API
 action.
    # The following code uses a parameter called S3_BUCKET to obtain the value
 specified in the
    # extension association. You can specify this parameter when you create the
 extension
    # later in this walkthrough.
    extension_association_params = event.get('Parameters', {})
    bucket_name = extension_association_params['S3_BUCKET']
    write_backup_to_s3(bucket_name, config_data_bytes)

    # The PRE_CREATE_HOSTED_CONFIGURATION_VERSION and PRE_START_DEPLOYMENT action
 points can
    # modify the contents of a configuration. The following code makes a minor change
    # for the purposes of a demonstration.
    old_config_data_string = config_data_bytes.decode('utf-8')
    new_config_data_string = old_config_data_string.replace('hello', 'hello!')
    new_config_data_bytes = new_config_data_string.encode('utf-8')

    # The lambda initially received the configuration data as a base64-encoded string
    # and must return it in the same format.
    new_config_data_base64string =
 base64.b64encode(new_config_data_bytes).decode('ascii')

    return {
        'statusCode': 200,
        # If you want to modify the contents of the configuration, you must include the
 new contents in the
        # Lambda response. If you don't want to modify the contents, you can omit the
 'Content' field shown here.
        'Content': new_config_data_base64string
    }


def write_backup_to_s3(bucket_name, config_data_bytes):
    s3 = boto3.resource('s3')
    new_object = s3.Object(bucket_name,
 f"config_backup_{datetime.now().isoformat()}.txt")
    new_object.put(Body=config_data_bytes)
```

Se você desejar usar esse exemplo durante este passo a passo, salve-o com o nome **`MyS3ConfigurationBackUpExtension`** e copie o nome do recurso da Amazon (ARN) da função.

Você especifica o ARN ao criar a função de assumir AWS Identity and Access Management (IAM) na próxima seção. Você especifica o ARN e o nome ao criar a extensão.

Referência de carga útil

Esta seção inclui detalhes de referência de solicitação e resposta de carga útil para trabalhar com AWS AppConfig extensões personalizadas.

Estrutura de solicitações

AtDeploymentTick

```
{
    'InvocationId': 'o2xbtm7',
    'Parameters': {
        'ParameterOne': 'ValueOne',
        'ParameterTwo': 'ValueTwo'
    },
    'Type': 'OnDeploymentStart',
    'Application': {
        'Id': 'abcd123'
    },
    'Environment': {
        'Id': 'efgh456'
    },
    'ConfigurationProfile': {
        'Id': 'ijkl789',
        'Name': 'ConfigurationName'
    },
    'DeploymentNumber': 2,
    'Description': 'Deployment description',
    'ConfigurationVersion': '2',
    'DeploymentState': 'DEPLOYING',
    'PercentageComplete': '0.0'
}
```

Estrutura de solicitações

PreCreateHostedConfigurationVersion

```
{
    'InvocationId': 'vlns753', // id for specific invocation
    'Parameters': {
```

```
        'ParameterOne': 'ValueOne',
        'ParameterTwo': 'ValueTwo'
    },
    'ContentType': 'text/plain',
    'ContentVersion': '2',
    'Content': 'SGVsbG8gZWFydGgh', // Base64 encoded content
    'Application': {
        'Id': 'abcd123',
        'Name': 'ApplicationName'
    },
    'ConfigurationProfile': {
        'Id': 'ijkl789',
        'Name': 'ConfigurationName'
    },
    'Description': '',
    'Type': 'PreCreateHostedConfigurationVersion',
    'PreviousContent': {
        'ContentType': 'text/plain',
        'ContentVersion': '1',
        'Content': 'SGVsbG8gd29ybGQh'
    }
}
```

PreStartDeployment

```
{
    'InvocationId': '765ahdm',
    'Parameters': {
        'ParameterOne': 'ValueOne',
        'ParameterTwo': 'ValueTwo'
    },
    'ContentType': 'text/plain',
    'ContentVersion': '2',
    'Content': 'SGVsbG8gZWFydGgh',
    'Application': {
        'Id': 'abcd123',
        'Name': 'ApplicationName'
    },
    'Environment': {
        'Id': 'ibpnqlq',
        'Name': 'EnvironmentName'
    },
    'ConfigurationProfile': {
```

```
        'Id': 'ijkl789',
        'Name': 'ConfigurationName'
    },
    'DeploymentNumber': 2,
    'Description': 'Deployment description',
    'Type': 'PreStartDeployment'
}
```

Eventos assíncronos

OnStartDeployment, OnDeploymentStep, OnDeployment

```
{
    'InvocationId': 'o2xbtm7',
    'Parameters': {
        'ParameterOne': 'ValueOne',
        'ParameterTwo': 'ValueTwo'
    },
    'Type': 'OnDeploymentStart',
    'Application': {
        'Id': 'abcd123'
    },
    'Environment': {
        'Id': 'efgh456'
    },
    'ConfigurationProfile': {
        'Id': 'ijkl789',
        'Name': 'ConfigurationName'
    },
    'DeploymentNumber': 2,
    'Description': 'Deployment description',
    'ConfigurationVersion': '2'
}
```

Estrutura de respostas

Os exemplos a seguir mostram o que sua função Lambda retorna em resposta à solicitação de uma extensão personalizada AWS AppConfig .

Eventos síncronos PRE_*: resposta bem-sucedida

Se você deseja transformar o conteúdo, use o seguinte:

```
"Content": "SomeBase64EncodedByteArray"
```

Eventos síncronos AT_*: resposta bem-sucedida

Se você quiser controlar as próximas etapas de uma implantação (continuar uma implantação ou revertê-la), defina os atributos `Directive` e `Description` na resposta.

```
"Directive": "ROLL_BACK"
"Description" "Deployment event log description"
```

A `Directive` oferece suporte a dois valores: `CONTINUE` ou `ROLL_BACK`. Use esses enums na sua resposta de carga útil para controlar as próximas etapas de uma implantação.

Eventos síncronos: resposta bem-sucedida

Se você deseja transformar o conteúdo, use o seguinte:

```
"Content": "SomeBase64EncodedByteArray"
```

Se você não deseja transformar o conteúdo, não retorne nada.

Eventos assíncronos: resposta bem-sucedida

Não retorne nada.

Todos os eventos de erro

```
{
        "Error": "BadRequestError",
        "Message": "There was malformed stuff in here",
        "Details": [{
            "Type": "Malformed",
            "Name": "S3 pointer",
            "Reason": "S3 bucket did not exist"
        }]
    }
```

Etapa 2: configurar permissões para uma AWS AppConfig extensão personalizada

Use o procedimento a seguir para criar e configurar uma função de serviço AWS Identity and Access Management (IAM) (ou assumir uma função). AWS AppConfig usa essa função para invocar a função Lambda.

Para criar uma função de serviço do IAM e AWS AppConfig permitir assumi-la

1. Abra o console do IAM em https://console.aws.amazon.com/iam/.
2. No painel de navegação, escolha Perfis e Criar perfil.
3. Em Selecionar tipo de entidade confiável, escolha Política de confiança personalizada.
4. Cole a política JSON a seguir no campo Política de confiança personalizada.

   JSON

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

   Escolha Próximo.
5. Na página Adicionar permissões, escolha Criar política. A página Create policy (Criar política) é aberta em uma nova guia.
6. Escolha a guia JSON e cole a política personalizada a seguir no editor. A ação `lambda:InvokeFunction` é usada para pontos de ação `PRE_*`. A ação `lambda:InvokeAsync` é usada para pontos de ação `ON_*`. *`Your Lambda ARN`*Substitua pelo Amazon Resource Name (ARN) do seu Lambda.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeFunction",
        "lambda:InvokeAsync"
      ],
      "Resource": "arn:aws:lambda:us-east-1:111122223333:function:function-
name"
    }
  ]
}
```

7. Escolha Próximo: tags.
8. Na página Adicionar tags (opcional), adicione um ou mais pares chave-valor e escolha Próximo: revisão.
9. Na página Revisar política, insira um nome e uma descrição e, depois, escolha Criar política.
10. Na guia do navegador da sua política de confiança personalizada, escolha o ícone Atualizar e pesquise a política de permissões que você acabou de criar.
11. Marque a caixa de seleção da política de permissões e escolha Próximo.
12. Na página Nomear, revisar e criar, insira um nome na caixa Nome do perfil e, em seguida, insira uma descrição.
13. Selecione Create role (Criar função). O sistema faz com que você retorne para a página Roles. Escolha Exibir perfil no banner.
14. Copie o ARN. Você especifica esse ARN ao criar a extensão.

Etapa 3: criar uma AWS AppConfig extensão personalizada

Uma extensão define uma ou mais ações que ela executa durante um AWS AppConfig fluxo de trabalho. Por exemplo, a `AWS AppConfig deployment events to Amazon SNS` extensão AWS criada inclui uma ação para enviar uma notificação para um tópico do Amazon SNS. Cada ação

é invocada quando você interage com AWS AppConfig ou quando AWS AppConfig está executando um processo em seu nome. Eles são chamados de pontos de ação. AWS AppConfig as extensões suportam os seguintes pontos de ação:

Pontos de ação PRE_*: as ações de extensão configuradas em pontos de ação `PRE_*` são aplicadas após a validação da solicitação, mas antes de o AWS AppConfig realizar a atividade que corresponde ao nome do ponto de ação. Essas invocações de ação são processadas ao mesmo tempo da solicitação. Se mais de uma solicitação for feita, as invocações de ação serão executadas sequencialmente. Observe também que os pontos de ação `PRE_*` recebem e podem alterar o conteúdo de uma configuração. Os pontos de ação `PRE_*` também podem responder a um erro e impedir que uma ação aconteça.

- `PRE_CREATE_HOSTED_CONFIGURATION_VERSION`
- `PRE_START_DEPLOYMENT`

Pontos de ação ON_*: uma extensão também pode ser executada paralelamente a um AWS AppConfig fluxo de trabalho usando um ponto de `ON_*` ação. `ON_*`os pontos de ação são invocados de forma assíncrona. `ON_*`os pontos de ação não recebem o conteúdo de uma configuração. Se uma extensão apresentar um erro durante um ponto de ação `ON_*`, o serviço ignorará o erro e continuará o fluxo de trabalho.

- `ON_DEPLOYMENT_START`
- `ON_DEPLOYMENT_STEP`
- `ON_DEPLOYMENT_BAKING`
- `ON_DEPLOYMENT_COMPLETE`
- `ON_DEPLOYMENT_ROLLED_BACK`

Pontos de ação AT_*: as ações de extensão configuradas nos pontos de ação `AT_*` são invocadas de maneira síncrona e em paralelo com um fluxo de trabalho do AWS AppConfig . Se uma extensão apresentar um erro durante um ponto de ação `AT_*`, o serviço interromperá o fluxo de trabalho e reverterá a implantação.

- `AT_DEPLOYMENT_TICK`

O ponto de ação `AT_DEPLOYMENT_TICK` oferece suporte à integração de monitoramento de terceiros. `AT_DEPLOYMENT_TICK` é invocado durante a orquestração do processamento da

implantação da configuração. Se você usa uma solução de monitoramento de terceiros (por exemplo, Datadog ou New Relic), você pode criar uma AWS AppConfig extensão que verifica os alarmes no ponto de `AT_DEPLOYMENT_TICK` ação e, como proteção de segurança, reverte a implantação se ela acionar um alarme.

Se você usa uma solução de monitoramento de terceiros, como Datadog ou New Relic, pode criar uma AWS AppConfig extensão que verifica os alarmes no ponto de `AT_DEPLOYMENT_TICK` ação e, como proteção de segurança, reverte a implantação se ela acionar um alarme. Para obter mais informações, consulte os seguintes exemplos de integração entre Datadog e New Relic em: GitHub

- Datadog
- New Relic

Para obter mais informações sobre AWS AppConfig extensões, consulte os tópicos a seguir:

- Estendendo AWS AppConfig fluxos de trabalho usando extensões
- Passo a passo: Criação de extensões personalizadas AWS AppConfig

Exemplo de extensão do

O exemplo de extensão a seguir define uma ação que chama o ponto de ação `PRE_CREATE_HOSTED_CONFIGURATION_VERSION`. No campo `Uri`, a ação especifica o nome do recurso da Amazon (ARN) da função do Lambda `MyS3ConfigurationBackUpExtension` criada anteriormente neste passo a passo. A ação também especifica o ARN de função de assumir AWS Identity and Access Management (IAM) criado anteriormente nesta explicação passo a passo.

AWS AppConfig Extensão de amostra

```
{
    "Name": "MySampleExtension",
    "Description": "A sample extension that backs up configurations to an S3 bucket.",
    "Actions": {
        "PRE_CREATE_HOSTED_CONFIGURATION_VERSION": [
            {
                "Name": "PreCreateHostedConfigVersionActionForS3Backup",
                "Uri": "arn:aws:lambda:aws-
region:111122223333:function:MyS3ConfigurationBackUpExtension",
                "RoleArn": "arn:aws:iam::111122223333:role/ExtensionsTestRole"
            }
```

```
        ]
    },
    "Parameters" : {
        "S3_BUCKET": {
            "Required": false
        }
    }
}
```

> **Note**
>
> Para ver a sintaxe da solicitação e as descrições dos campos ao criar uma extensão, consulte o CreateExtensiontópico na Referência da AWS AppConfig API.

Para criar uma extensão (console)

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, escolha AWS AppConfig.
3. Na guia Extensões, escolha Criar extensão.
4. Em Nome da extensão, insira um nome exclusivo. Para a finalidade deste passo a passo, insira **`MyS3ConfigurationBackUpExtension`**. Como opção, insira uma descrição.
5. Na seção Ações, selecione Adicionar nova ação.
6. Em Nome da ação, insira um nome exclusivo. Para a finalidade deste passo a passo, insira **`PreCreateHostedConfigVersionActionForS3Backup`**. Esse nome descreve o ponto de ação usado pela ação e a finalidade da extensão.
7. Na lista Ponto de ação, escolha PRE_CREATE_HOSTED_CONFIGURATION_VERSION.
8. Para Uri, escolha Função do Lambda e, em seguida, escolha a função na lista Função do Lambda. Se você não vê sua função, verifique se você está no mesmo Região da AWS local em que criou a função.
9. Para Perfil do IAM, escolha o perfil que você criou anteriormente neste passo a passo.
10. Na seção Parâmetros de extensão (opcional), escolha Adicionar novo parâmetro.
11. Em Nome do parâmetro, insira um nome. Para a finalidade deste passo a passo, insira **`S3_BUCKET`**.

12. Repita as etapas de 5 a 11 para criar uma segunda ação para o ponto de ação `PRE_START_DEPLOYMENT`.
13. Escolha Criar extensão.

Personalização de extensões AWS de notificação criadas

Não é necessário criar um Lambda ou uma extensão para usar extensões de notificação criadas pela AWS. Basta criar uma associação de extensão e, em seguida, realizar uma operação que chame um dos pontos de ação suportados. Por padrão, as extensões AWS de notificação criadas oferecem suporte aos seguintes pontos de ação:

- `ON_DEPLOYMENT_START`
- `ON_DEPLOYMENT_COMPLETE`
- `ON_DEPLOYMENT_ROLLED_BACK`

Se criar versões personalizadas da extensão `AWS AppConfig deployment events to Amazon SNS` e das extensões `AWS AppConfig deployment events to Amazon SQS`, você poderá especificar os pontos de ação para os quais deseja receber notificações.

> **Note**
>
> A extensão `AWS AppConfig deployment events to EventBridge` não suporta os pontos de ação `PRE_*`. Você pode criar uma versão personalizada se quiser remover alguns dos pontos de ação padrão atribuídos à AWS versão criada.

Você não precisa criar nenhuma função do Lambda se criar versões personalizadas das extensões de notificação criadas pela AWS . Você só precisa especificar um nome do recurso da Amazon (ARN) no campo `Uri` para a nova versão da extensão.

- Para uma extensão de EventBridge notificação personalizada, insira o ARN dos eventos EventBridge padrão no `Uri` campo.
- Para uma extensão de notificação personalizada do Amazon SNS, insira o ARN de um tópico do Amazon SNS no campo `Uri`.
- Para uma extensão de notificação personalizada do Amazon SQS, insira o ARN de uma fila de mensagens do Amazon SQS no campo `Uri`.

Etapa 4: criar uma associação de extensão para uma AWS AppConfig extensão personalizada

Para criar uma extensão ou configurar uma extensão de AWS autoria, você define os pontos de ação que invocam uma extensão quando um AWS AppConfig recurso específico é usado. Por exemplo, você pode optar por executar a extensão `AWS AppConfig deployment events to Amazon SNS` e receber notificações sobre um tópico do Amazon SNS sempre que uma implantação de configuração for iniciada para um aplicativo específico. Definir quais pontos de ação invocam uma extensão para um AWS AppConfig recurso específico é chamado de associação de extensão. Uma associação de extensão é uma relação especificada entre uma extensão e um AWS AppConfig recurso, como um aplicativo ou um perfil de configuração.

Um único AWS AppConfig aplicativo pode incluir vários ambientes e perfis de configuração. Se você associar uma extensão a um aplicativo ou ambiente, AWS AppConfig invoca a extensão para qualquer fluxo de trabalho relacionado ao aplicativo ou aos recursos do ambiente, se aplicável.

Por exemplo, digamos que você tenha um AWS AppConfig aplicativo chamado MobileApps que inclui um perfil de configuração chamado AccessList. E digamos que o MobileApps aplicativo inclua ambientes beta, de integração e de produção. Você cria uma associação de extensão para a extensão AWS de notificação criada pelo Amazon SNS e associa a extensão ao aplicativo. MobileApps A extensão de notificação do Amazon SNS é invocada sempre que a configuração é implantada para o aplicativo em qualquer um dos três ambientes.

Use os procedimentos a seguir para criar uma associação de AWS AppConfig extensão usando o AWS AppConfig console.

Para criar uma associação de extensão (console)

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, escolha AWS AppConfig.
3. Na guia Extensões, escolha um botão de opção para uma extensão e escolha Adicionar ao recurso. Para fins deste passo a passo, escolha MyS3. ConfigurationBackUpExtension
4. Na seção Detalhes do recurso de extensão, em Tipo de recurso, escolha um tipo de AWS AppConfig recurso. Dependendo do recurso escolhido, AWS AppConfig solicita que você escolha outros recursos. Para a finalidade deste passo a passo, escolha Aplicativo.
5. Selecione um aplicativo na lista.

6. Na seção Parâmetros, verifique se S3_BUCKET está listado no campo Chave. No campo Valor, cole o ARN das extensões do Lambda. Por exemplo: `arn:aws:lambda:aws-region:111122223333:function:MyS3ConfigurationBackUpExtension`.
7. Escolha Criar associação ao recurso.

Depois de criar a associação, você pode invocar a extensão `MyS3ConfigurationBackUpExtension` criando um novo perfil de configuração que especifique `hosted` para seu `SourceUri`. Como parte do fluxo de trabalho para criar a nova configuração, AWS AppConfig encontra o ponto de `PRE_CREATE_HOSTED_CONFIGURATION_VERSION` ação. O encontro desse ponto de ação invoca a extensão `MyS3ConfigurationBackUpExtension`, que faz backup automático da configuração recém-criada no bucket do S3 especificado na seção `Parameter` da associação da extensão.

Usando amostras de código para realizar AWS AppConfig tarefas comuns

Esta seção inclui exemplos de código para executar ações comuns AWS AppConfig de forma programática. Recomendamos que você use esses exemplos com Java, Python e execute JavaScript SDKs as ações em um ambiente de teste. Esta seção inclui um exemplos de código para limpar o ambiente de teste depois que você terminar.

Tópicos

- Criar ou atualizar uma configuração de forma livre gravada no armazenamento de configuração hospedado
- Criar um perfil de configuração para um segredo armazenado no Secrets Manager
- Implantar um perfil de configuração
- Usando o AWS AppConfig Agente para ler um perfil de configuração de formato livre
- Usando o AWS AppConfig Agente para ler um sinalizador de recurso específico
- Usando o AWS AppConfig Agente para recuperar um sinalizador de recurso com variantes
- Usando a ação GetLatestConfiguration da API para ler um perfil de configuração de formato livre
- Limpar o ambiente

Criar ou atualizar uma configuração de forma livre gravada no armazenamento de configuração hospedado

Cada um dos exemplos a seguir inclui comentários sobre as ações realizadas pelo código. Os exemplos nesta seção chamam o seguinte APIs:

- CreateApplication
- CreateConfigurationProfile
- CreateHostedConfigurationVersion

Java

```
public CreateHostedConfigurationVersionResponse createHostedConfigVersion() {
        AppConfigClient appconfig = AppConfigClient.create();
```

```
        // Create an application
        CreateApplicationResponse app = appconfig.createApplication(req ->
 req.name("MyDemoApp"));

        // Create a hosted, freeform configuration profile
        CreateConfigurationProfileResponse configProfile =
 appconfig.createConfigurationProfile(req -> req
            .applicationId(app.id())
            .name("MyConfigProfile")
            .locationUri("hosted")
            .type("AWS.Freeform"));

        // Create a hosted configuration version
        CreateHostedConfigurationVersionResponse hcv =
 appconfig.createHostedConfigurationVersion(req -> req
            .applicationId(app.id())
            .configurationProfileId(configProfile.id())
            .contentType("text/plain; charset=utf-8")
            .content(SdkBytes.fromUtf8String("my config data")));

        return hcv;
    }
```

Python

```
import boto3

appconfig = boto3.client('appconfig')

# create an application
application = appconfig.create_application(Name='MyDemoApp')

# create a hosted, freeform configuration profile
config_profile = appconfig.create_configuration_profile(
    ApplicationId=application['Id'],
    Name='MyConfigProfile',
    LocationUri='hosted',
    Type='AWS.Freeform')

# create a hosted configuration version
hcv = appconfig.create_hosted_configuration_version(
    ApplicationId=application['Id'],
```

```
    ConfigurationProfileId=config_profile['Id'],
    Content=b'my config data',
    ContentType='text/plain')
```

JavaScript

```
import {
  AppConfigClient,
  CreateApplicationCommand,
  CreateConfigurationProfileCommand,
  CreateHostedConfigurationVersionCommand,
} from "@aws-sdk/client-appconfig";

const appconfig = new AppConfigClient();

// create an application
const application = await appconfig.send(
  new CreateApplicationCommand({ Name: "MyDemoApp" })
);

// create a hosted, freeform configuration profile
const profile = await appconfig.send(
  new CreateConfigurationProfileCommand({
    ApplicationId: application.Id,
    Name: "MyConfigProfile",
    LocationUri: "hosted",
    Type: "AWS.Freeform",
  })
);

// create a hosted configuration version
await appconfig.send(
  new CreateHostedConfigurationVersionCommand({
    ApplicationId: application.Id,
    ConfigurationProfileId: profile.Id,
    ContentType: "text/plain",
    Content: "my config data",
  })
);
```

Criar um perfil de configuração para um segredo armazenado no Secrets Manager

Cada um dos exemplos a seguir inclui comentários sobre as ações realizadas pelo código. Os exemplos nesta seção chamam o seguinte APIs:

- CreateApplication
- CreateConfigurationProfile

Java

```
private void createSecretsManagerConfigProfile() {
        AppConfigClient appconfig = AppConfigClient.create();

        // Create an application
        CreateApplicationResponse app = appconfig.createApplication(req ->
 req.name("MyDemoApp"));

        // Create a configuration profile for Secrets Manager Secret
        CreateConfigurationProfileResponse configProfile =
 appconfig.createConfigurationProfile(req -> req
            .applicationId(app.id())
            .name("MyConfigProfile")
            .locationUri("secretsmanager://MySecret")
            .retrievalRoleArn("arn:aws:iam::000000000000:role/
RoleTrustedByAppConfigThatCanRetrieveSecret")
            .type("AWS.Freeform"));
    }
```

Python

```
import boto3

appconfig = boto3.client('appconfig')

# create an application
application = appconfig.create_application(Name='MyDemoApp')

# create a configuration profile for Secrets Manager Secret
config_profile = appconfig.create_configuration_profile(
```

```
    ApplicationId=application['Id'],
    Name='MyConfigProfile',
    LocationUri='secretsmanager://MySecret',
    RetrievalRoleArn='arn:aws:iam::000000000000:role/
RoleTrustedByAppConfigThatCanRetrieveSecret',
    Type='AWS.Freeform')
```

JavaScript

```
import {
  AppConfigClient,
  CreateConfigurationProfileCommand,
} from "@aws-sdk/client-appconfig";

const appconfig = new AppConfigClient();

// create an application
const application = await appconfig.send(
  new CreateApplicationCommand({ Name: "MyDemoApp" })
);

// create a configuration profile for Secrets Manager Secret
await appconfig.send(
  new CreateConfigurationProfileCommand({
    ApplicationId: application.Id,
    Name: "MyConfigProfile",
    LocationUri: "secretsmanager://MySecret",
    RetrievalRoleArn: "arn:aws:iam::000000000000:role/
RoleTrustedByAppConfigThatCanRetrieveSecret",
    Type: "AWS.Freeform",
  })
);
```

Implantar um perfil de configuração

Cada um dos exemplos a seguir inclui comentários sobre as ações realizadas pelo código. Os exemplos nesta seção chamam o seguinte APIs:

- CreateApplication
- CreateConfigurationProfile
- CreateHostedConfigurationVersion

- CreateEnvironment
- StartDeployment
- GetDeployment

Java

```
private void createDeployment() throws InterruptedException {
        AppConfigClient appconfig = AppConfigClient.create();

        // Create an application
        CreateApplicationResponse app = appconfig.createApplication(req ->
 req.name("MyDemoApp"));

        // Create a hosted, freeform configuration profile
        CreateConfigurationProfileResponse configProfile =
 appconfig.createConfigurationProfile(req -> req
            .applicationId(app.id())
            .name("MyConfigProfile")
            .locationUri("hosted")
            .type("AWS.Freeform"));

        // Create a hosted configuration version
        CreateHostedConfigurationVersionResponse hcv =
 appconfig.createHostedConfigurationVersion(req -> req
            .applicationId(app.id())
            .configurationProfileId(configProfile.id())
            .contentType("text/plain; charset=utf-8")
            .content(SdkBytes.fromUtf8String("my config data")));


        // Create an environment
        CreateEnvironmentResponse env = appconfig.createEnvironment(req -> req
            .applicationId(app.id())
            .name("Beta")
            // If you have CloudWatch alarms that monitor the health of your
 service, you can add them here and they
            // will trigger a rollback if they fire during an appconfig deployment
            //.monitors(Monitor.builder().alarmArn("arn:aws:cloudwatch:us-
east-1:520900602629:alarm:MyAlarm")
            //
  .alarmRoleArn("arn:aws:iam::520900602629:role/MyAppConfigAlarmRole").build())
        );
```

```
        // Start a deployment
        StartDeploymentResponse deploymentResponse = appconfig.startDeployment(req -
> req
            .applicationId(app.id())
            .configurationProfileId(configProfile.id())
            .environmentId(env.id())
            .configurationVersion(hcv.versionNumber().toString())
            .deploymentStrategyId("AppConfig.Linear50PercentEvery30Seconds")
        );

        // Wait for deployment to complete
        List<DeploymentState> nonFinalDeploymentStates = Arrays.asList(
            DeploymentState.DEPLOYING,
            DeploymentState.BAKING,
            DeploymentState.ROLLING_BACK,
            DeploymentState.VALIDATING);
        GetDeploymentRequest getDeploymentRequest =
 GetDeploymentRequest.builder().applicationId(app.id())

 .environmentId(env.id())

 .deploymentNumber(deploymentResponse.deploymentNumber()).build();
        GetDeploymentResponse deployment =
 appconfig.getDeployment(getDeploymentRequest);
        while (nonFinalDeploymentStates.contains(deployment.state())) {
            System.out.println("Waiting for deployment to complete: " + deployment);
            Thread.sleep(1000L);
            deployment = appconfig.getDeployment(getDeploymentRequest);
        }

        System.out.println("Deployment complete: " + deployment);
    }
```

Python

```
import boto3

appconfig = boto3.client('appconfig')

# create an application
application = appconfig.create_application(Name='MyDemoApp')
```

```
# create an environment
environment = appconfig.create_environment(
    ApplicationId=application['Id'],
    Name='MyEnvironment')

# create a configuration profile
config_profile = appconfig.create_configuration_profile(
    ApplicationId=application['Id'],
    Name='MyConfigProfile',
    LocationUri='hosted',
    Type='AWS.Freeform')

# create a hosted configuration version
hcv = appconfig.create_hosted_configuration_version(
    ApplicationId=application['Id'],
    ConfigurationProfileId=config_profile['Id'],
    Content=b'my config data',
    ContentType='text/plain')

# start a deployment
deployment = appconfig.start_deployment(
    ApplicationId=application['Id'],
    EnvironmentId=environment['Id'],
    ConfigurationProfileId=config_profile['Id'],
    ConfigurationVersion=str(hcv['VersionNumber']),
    DeploymentStrategyId='AppConfig.Linear20PercentEvery6Minutes')
```

JavaScript

```
import {
  AppConfigClient,
  CreateApplicationCommand,
  CreateEnvironmentCommand,
  CreateConfigurationProfileCommand,
  CreateHostedConfigurationVersionCommand,
  StartDeploymentCommand,
} from "@aws-sdk/client-appconfig";

const appconfig = new AppConfigClient();

// create an application
const application = await appconfig.send(
  new CreateApplicationCommand({ Name: "MyDemoApp" })
```

```
);

// create an environment
const environment = await appconfig.send(
  new CreateEnvironmentCommand({
    ApplicationId: application.Id,
    Name: "MyEnvironment",
  })
);

// create a configuration profile
const config_profile = await appconfig.send(
  new CreateConfigurationProfileCommand({
    ApplicationId: application.Id,
    Name: "MyConfigProfile",
    LocationUri: "hosted",
    Type: "AWS.Freeform",
  })
);

// create a hosted configuration version
const hcv = await appconfig.send(
  new CreateHostedConfigurationVersionCommand({
    ApplicationId: application.Id,
    ConfigurationProfileId: config_profile.Id,
    Content: "my config data",
    ContentType: "text/plain",
  })
);

// start a deployment
await appconfig.send(
  new StartDeploymentCommand({
    ApplicationId: application.Id,
    EnvironmentId: environment.Id,
    ConfigurationProfileId: config_profile.Id,
    ConfigurationVersion: hcv.VersionNumber.toString(),
    DeploymentStrategyId: "AppConfig.Linear20PercentEvery6Minutes",
  })
);
```

Usando o AWS AppConfig Agente para ler um perfil de configuração de formato livre

Cada um dos exemplos a seguir inclui comentários sobre as ações realizadas pelo código.

Java

```
public void retrieveConfigFromAgent() throws Exception {
        /*
        In this sample, we will retrieve configuration data from the AWS AppConfig
 Agent.
        The agent is a sidecar process that handles retrieving configuration data
 from AppConfig
        for you in a way that implements best practices like configuration caching.

        For more information about the agent, see How to use AWS AppConfig Agent
        */

        // The agent runs a local HTTP server that serves configuration data
        // Make a GET request to the agent's local server to retrieve the
 configuration data
        URL url = new URL("http://localhost:2772/applications/MyDemoApp/
environments/Beta/configurations/MyConfigProfile");
        HttpURLConnection con = (HttpURLConnection) url.openConnection();
        con.setRequestMethod("GET");
        StringBuilder content;
        try (BufferedReader in = new BufferedReader(new
 InputStreamReader(con.getInputStream()))) {
            content = new StringBuilder();
            int ch;
            while ((ch = in.read()) != -1) {
                content.append((char) ch);
            }
        }
        con.disconnect();
        System.out.println("Configuration from agent via HTTP: " + content);
    }
```

Python

```
# in this sample, we will retrieve configuration data from the AWS AppConfig Agent.
```

```
# the agent is a sidecar process that handles retrieving configuration data from AWS
 AppConfig
# for you in a way that implements best practices like configuration caching.
#
# for more information about the agent, see
# How to use AWS AppConfig Agent
#

import requests

application_name = 'MyDemoApp'
environment_name = 'MyEnvironment'
config_profile_name = 'MyConfigProfile'

# the agent runs a local HTTP server that serves configuration data
# make a GET request to the agent's local server to retrieve the configuration data
response = requests.get(f"http://localhost:2772/applications/{application_name}/
environments/{environment_name}/configurations/{config_profile_name}")
config = response.content
```

JavaScript

```
// in this sample, we will retrieve configuration data from the AWS AppConfig Agent.
// the agent is a sidecar process that handles retrieving configuration data from
 AppConfig
// for you in a way that implements best practices like configuration caching.

// for more information about the agent, see
// How to use AWS AppConfig Agent

const application_name = "MyDemoApp";
const environment_name = "MyEnvironment";
const config_profile_name = "MyConfigProfile";

// the agent runs a local HTTP server that serves configuration data
// make a GET request to the agent's local server to retrieve the configuration data
const url = `http://localhost:2772/applications/${application_name}/environments/
${environment_name}/configurations/${config_profile_name}`;
const response = await fetch(url);
const config = await response.text(); // (use `await response.json()` if your config
 is json)
```

Usando o AWS AppConfig Agente para ler um sinalizador de recurso específico

Cada um dos exemplos a seguir inclui comentários sobre as ações realizadas pelo código.

Java

```
public void retrieveSingleFlagFromAgent() throws Exception {
        /*
          You can retrieve a single flag's data from the agent by providing the
 "flag" query string parameter.
          Note: the configuration's type must be AWS.AppConfig.FeatureFlags
        */

        URL url = new URL("http://localhost:2772/applications/MyDemoApp/
environments/Beta/configurations/MyFlagsProfile?flag=myFlagKey");
        HttpURLConnection con = (HttpURLConnection) url.openConnection();
        con.setRequestMethod("GET");
        StringBuilder content;
        try (BufferedReader in = new BufferedReader(new
 InputStreamReader(con.getInputStream()))) {
            content = new StringBuilder();
            int ch;
            while ((ch = in.read()) != -1) {
                content.append((char) ch);
            }
        }
        con.disconnect();
        System.out.println("MyFlagName from agent: " + content);
    }
```

Python

```
import requests

application_name = 'MyDemoApp'
environment_name = 'MyEnvironment'
config_profile_name = 'MyConfigProfile'
flag_key = 'MyFlag'

# retrieve a single flag's data by providing the "flag" query string parameter
# note: the configuration's type must be AWS.AppConfig.FeatureFlags
```

```
response = requests.get(f"http://localhost:2772/applications/{application_name}/
environments/{environment_name}/configurations/{config_profile_name}?
flag={flag_key}")
config = response.content
```

JavaScript

```
const application_name = "MyDemoApp";
const environment_name = "MyEnvironment";
const config_profile_name = "MyConfigProfile";
const flag_name = "MyFlag";

// retrieve a single flag's data by providing the "flag" query string parameter
// note: the configuration's type must be AWS.AppConfig.FeatureFlags
const url = `http://localhost:2772/applications/${application_name}/environments/
${environment_name}/configurations/${config_profile_name}?flag=${flag_name}`;
const response = await fetch(url);
const flag = await response.json(); // { "enabled": true/false }
```

Usando o AWS AppConfig Agente para recuperar um sinalizador de recurso com variantes

Cada um dos exemplos a seguir inclui comentários sobre as ações realizadas pelo código.

Java

```
public static void retrieveConfigFromAgentWithVariants() throws Exception {
    /*
    This sample retrieves feature flag configuration data
    containing variants from AWS AppConfig Agent.

    For more information about the agent, see How to use AWS AppConfig Agent
    */

    // Make a GET request to the agent's local server to retrieve the configuration
 data
    URL url = new URL("http://localhost:2772/applications/MyDemoApp/environments/
Beta/configurations/MyConfigProfile");
    HttpURLConnection con = (HttpURLConnection) url.openConnection();

    // Provide context in the 'Context' header
```

```
    // In the header value, use '=' to separate context key from context value
    // Note: Multiple context values may be passed either across
    // multiple headers or as comma-separated values in a single header
    con.setRequestProperty("Context", "country=US");

    StringBuilder content;
    try (BufferedReader in = new BufferedReader(new
 InputStreamReader(con.getInputStream()))) {
        content = new StringBuilder();
        int ch;
        while ((ch = in.read()) != -1) {
            content.append((char) ch);
        }
    }
    con.disconnect();
    System.out.println("Configuration from agent via HTTP: " + content);
}
```

Python

```
# This sample retrieve features flag configuration data
# containing variants from AWS AppConfig Agent.

# For more information about the agent, see How to use AWS AppConfig Agent

import requests

application_name = 'MyDemoApp'
environment_name = 'Beta'
configuration_profile_name = 'MyConfigProfile'

# make a GET request to the agent's local server to retrieve the configuration data
response = requests.get(f"http://localhost:2772/applications/{application_name}/
environments/{environment_name}/configurations/{configuration_profile_name}",
                        headers = {
                            "Context": "country=US" # Provide context in the
 'Context' header
                                                    # In the header value, use '='
 to separate context key from context value
                                                    # Note: Multiple context values
 may be passed either across
                                                    # multiple headers or as comma-
separated values in a single header
```

```
                        }
)
print("Configuration from agent via HTTP: ", response.json())
```

JavaScript

```
// This sample retrieves feature flag configuration data
// containing variants from AWS AppConfig Agent.

// For more information about the agent, see How to use AWS AppConfig Agent

const application_name = "MyDemoApp";
const environment_name = "Beta";
const configuration_profile_name = "MyConfigProfile";

const url = `http://localhost:2772/applications/${application_name}/environments/
${environment_name}/configurations/${configuration_profile_name}`;

// make a GET request to the agent's local server to retrieve the configuration data
const response = await fetch(url, {
    method: 'GET',
    headers: {
        'Context': 'country=US' // Provide context in the 'Context' header
                                // In the header value, use '=' to separate context
 key from context value
                                // Note: Multiple context values may be passed
 either across
                                // multiple headers or as comma-separated values in
 a single header
    }
});

const config = await response.json();
console.log("Configuration from agent via HTTP: ", config);
```

Usando a ação GetLatestConfiguration da API para ler um perfil de configuração de formato livre

Cada um dos exemplos a seguir inclui comentários sobre as ações realizadas pelo código. Os exemplos nesta seção chamam o seguinte APIs:

- GetLatestConfiguration
- StartConfigurationSession

Java

```
/*
The example below uses two AWS AppConfig Data APIs: StartConfigurationSession and
 GetLatestConfiguration.
For more information about these APIs, see AWS AppConfig Data.

This class is meant to be used as a singleton to retrieve the latest configuration
 data from AWS AppConfig.
This class maintains a cache of the latest configuration data in addition to the
 configuration token to be
passed to the next GetLatestConfiguration API call.
*/
public class AppConfigApiRetriever {

    /*
     * Set of AppConfig invalid parameter problems that require restarting the
 configuration session.
     * If the GetLatestConfiguration API call fails with any of these problems (e.g.
 token is EXPIRED or CORRUPTED),
     * we need to call StartConfigurationSession again to obtain a new configuration
 token before retrying.
     */
    private final Set<InvalidParameterProblem> SESSION_RESTART_REQUIRED =
            Stream.of(InvalidParameterProblem.EXPIRED,
 InvalidParameterProblem.CORRUPTED)
                    .collect(Collectors.toSet());

    /* AWS AppConfig Data SDK client used to interact with the AWS AppConfig Data
 service.
     */
    private final AppConfigDataClient appConfigData;

    /*
    The configuration token to be passed to the next GetLatestConfiguration API
 call.
     */
    private String configurationToken;
```

```
   /*
   The cached configuration data to be returned when there is no new configuration
data available.
    */
   private SdkBytes configuration;

   public AppConfigApiRetriever() {
       this.appConfigData = AppConfigDataClient.create();
   }

   /*
   Returns the latest configuration data stored in AWS AppConfig.
    */
   public SdkBytes getConfig() {
       /*
       If there is no configuration token yet, get one by starting a new session
with the StartConfigurationSession API.
       Note that this API does not return configuration data. Rather, it returns an
initial configuration token that is
       subsequently passed to the GetLatestConfiguration API.
        */
       if (this.configurationToken == null) {
           startNewSession();
       }

       GetLatestConfigurationResponse response = null;

       try {
           /*
           Retrieve the configuration from the GetLatestConfiguration API,
providing the current configuration token.
           If this caller does not yet have the latest configuration (e.g. this is
the first call to GetLatestConfiguration
           or new configuration data has been deployed since the first call), the
latest configuration data will be returned.
           Otherwise, the GetLatestConfiguration API will not return any data since
the caller already has the latest.
            */
           response = appConfigData.getLatestConfiguration(
                   GetLatestConfigurationRequest.builder()

.configurationToken(this.configurationToken)
                                                   .build());
```

```
        } catch (ResourceNotFoundException e) {
            // Handle resource not found by refreshing the session
            System.err.println("Resource not found — refreshing session and
 retrying...");
            startNewSession();
            response = appConfigData.getLatestConfiguration(
                    GetLatestConfigurationRequest.builder()

 .configurationToken(this.configurationToken)
                                                .build());
        } catch (BadRequestException e) {
            // Handle expired or corrupted token by refreshing the session
            boolean needsNewSession = Optional.ofNullable(e.details())
                                              .map(details ->
 details.invalidParameters()
                                                                     .values()
                                                                     .stream()
                                                                     .anyMatch(val -
> SESSION_RESTART_REQUIRED.contains(val.problem())))
                                              .orElse(false);
            if (needsNewSession) {
                System.err.println("Configuration token expired or corrupted —
 refreshing session and retrying...");
                startNewSession();
                response = appConfigData.getLatestConfiguration(
                        GetLatestConfigurationRequest.builder()

 .configurationToken(this.configurationToken)
                                                        .build());
            } else {
                throw e; // rethrow if it's another kind of bad request
            }
        }

        if (response == null) {
            // Should not happen, but return cached config if no response
            return this.configuration;
        }

        /*
        Save the returned configuration token so that it can be passed to the next
 GetLatestConfiguration API call.
        Warning: Not persisting this token for use in the next
 GetLatestConfiguration API call may result in higher
```

```
        than expected usage costs.
         */
        this.configurationToken = response.nextPollConfigurationToken();

        /*
        If the GetLatestConfiguration API returned configuration data, update the
 cached configuration with the returned data.
        Otherwise, assume the configuration has not changed, and return the cached
 configuration.
         */
        SdkBytes configFromApi = response.configuration();
        if (configFromApi != null && configFromApi.asByteArray().length != 0) {
            this.configuration = configFromApi;
            System.out.println("Configuration contents have changed since the last
 GetLatestConfiguration call, new contents = "
                    + this.configuration.asUtf8String());
        } else {
            System.out.println("GetLatestConfiguration returned an empty response
 because we already have the latest configuration");
        }

        return this.configuration;
    }

    /*
    Starts a new session with AppConfig and retrieves an initial configuration
 token.
     */
    private void startNewSession() {
        StartConfigurationSessionResponse session =
 appConfigData.startConfigurationSession(req -> req
                .applicationIdentifier("MyDemoApp")
                .configurationProfileIdentifier("MyConfig")
                .environmentIdentifier("Beta"));
        this.configurationToken = session.initialConfigurationToken();
    }
}
```

Python

```
# The example below uses two AWS AppConfig Data APIs: StartConfigurationSession and
 GetLatestConfiguration.
# For more information about these APIs, see AWS AppConfig Data.
```

```
#
# This class is meant to be used as a singleton to retrieve the latest configuration
 data from AWS AppConfig.
# This class maintains a cache of the latest configuration data in addition to the
 configuration token to be
# passed to the next GetLatestConfiguration API call.
class AppConfigApiRetriever:
    # Set of AppConfig invalid parameter problems that require restarting the
 configuration session.
    # If the GetLatestConfiguration API call fails with any of these problems (e.g.
 token is EXPIRED or CORRUPTED),
    # we need to call StartConfigurationSession again to obtain a new configuration
 token before retrying.
    SESSION_RESTART_REQUIRED = {"EXPIRED", "CORRUPTED"}

    def __init__(self):
        # AWS AppConfig Data SDK client used to interact with the AWS AppConfig Data
 service.
        self.appconfigdata = boto3.client('appconfigdata')

        # The configuration token to be passed to the next GetLatestConfiguration
 API call.
        self.configuration_token = None

        # The cached configuration data to be returned when there is no new
 configuration data available.
        self.configuration = None

    # Returns the latest configuration data stored in AWS AppConfig.
    def get_config(self):
        # If there is no configuration token yet, get one by starting a new session
 with the StartConfigurationSession API.
        # Note that this API does not return configuration data. Rather, it returns
 an initial configuration token that is
        # subsequently passed to the GetLatestConfiguration API.
        if not self.configuration_token:
            self._start_new_session()

        response = None
        try:
            # Retrieve the configuration from the GetLatestConfiguration API,
 providing the current configuration token.
            # If this caller does not yet have the latest configuration (e.g. this
 is the first call to GetLatestConfiguration
```

```
            # or new configuration data has been deployed since the first call), the
latest configuration data will be returned.
            # Otherwise, the GetLatestConfiguration API will not return any data
since the caller already has the latest.
            response = self.appconfigdata.get_latest_configuration(
                ConfigurationToken=self.configuration_token
            )
        except ClientError as e:
            error_code = e.response.get("Error", {}).get("Code")
            # ResourceNotFoundException — usually means the token/session is invalid
or expired
            if error_code == "ResourceNotFoundException":
                print("Resource not found — refreshing session and retrying...")
                self._start_new_session()
                response = self.appconfigdata.get_latest_configuration(
                    ConfigurationToken=self.configuration_token
                )
            # BadRequestException — check if it's expired or corrupted token
            elif error_code == "BadRequestException":
                details = e.response.get("Error", {}).get("Details", {}) or {}
                invalid_params = details.get("InvalidParameters", {}) or {}
                needs_new_session = any(
                    param.get("Problem") in self.SESSION_RESTART_REQUIRED
                    for param in invalid_params.values()
                )
                if needs_new_session:
                    print("Configuration token expired or corrupted — refreshing
session and retrying...")
                    self._start_new_session()
                    response = self.appconfigdata.get_latest_configuration(
                        ConfigurationToken=self.configuration_token
                    )
                else:
                    raise
            else:
                raise

        if response is None:
            # Should not happen, but return cached config if no response
            return self.configuration

        # Save the returned configuration token so that it can be passed to the next
GetLatestConfiguration API call.
```

```
        # Warning: Not persisting this token for use in the next
 GetLatestConfiguration API call may result in higher
        # than expected usage costs.
        self.configuration_token = response['NextPollConfigurationToken']

        # If the GetLatestConfiguration API returned configuration data, update the
 cached configuration with the returned data.
        # Otherwise, assume the configuration has not changed, and return the cached
 configuration.
        config_stream = response.get('Configuration')
        if config_stream:
            config_from_api = config_stream.read()
            if config_from_api:
                self.configuration = config_from_api
                print(
                    'Configuration contents have changed since the last
 GetLatestConfiguration call, new contents = '
                    + self.configuration.decode('utf-8', errors='ignore')
                )
            else:
                print('GetLatestConfiguration returned an empty response because we
 already have the latest configuration')

        return self.configuration

    # Starts a new session with AppConfig and retrieves an initial configuration
 token.
    def _start_new_session(self):
        session = self.appconfigdata.start_configuration_session(
            ApplicationIdentifier='MyDemoApp',
            ConfigurationProfileIdentifier='MyConfig',
            EnvironmentIdentifier='Beta'
        )
        self.configuration_token = session['InitialConfigurationToken']
```

JavaScript

```
/*
The example below uses two AWS AppConfig Data APIs: StartConfigurationSession and
 GetLatestConfiguration.
For more information about these APIs, see AWS AppConfig Data
.
```

```
This class is meant to be used as a singleton to retrieve the latest configuration
 data from AWS AppConfig.
This class maintains a cache of the latest configuration data in addition to the
 configuration token to be
passed to the next GetLatestConfiguration API call.
*/
class AppConfigApiRetriever {
  constructor() {
    /* AWS AppConfig Data SDK client used to interact with the AWS AppConfig Data
 service.
     */
    this.appconfigdata = new AppConfigDataClient();

    /*
    The configuration token to be passed to the next GetLatestConfiguration API
 call.
     */
    this.configurationToken = null;

    /*
    The cached configuration data to be returned when there is no new configuration
 data available.
     */
    this.configuration = null;
  }

  async startSession() {
    /*
    Starts a new session with the StartConfigurationSession API to get an initial
 configuration token.
    */
    const session = await this.appconfigdata.send(
      new StartConfigurationSessionCommand({
        ApplicationIdentifier: "MyDemoApp",
        ConfigurationProfileIdentifier: "MyConfig",
        EnvironmentIdentifier: "Beta"
      })
    );
    this.configurationToken = session.InitialConfigurationToken;
  }

  /*
  Returns the latest configuration data stored in AWS AppConfig.
   */
```

```
 async getConfig() {
   /*
   If there is no configuration token yet, get one by starting a new session with
the StartConfigurationSession API.
   Note that this API does not return configuration data. Rather, it returns an
initial configuration token that is
   subsequently passed to the GetLatestConfiguration API.
   */
   if (!this.configurationToken) {
     await this.startSession();
   }

   let response;
   try {
     /*
     Retrieve the configuration from the GetLatestConfiguration API, providing the
current configuration token.
     If this caller does not yet have the latest configuration (e.g. this is the
first call to GetLatestConfiguration
     or new configuration data has been deployed since the first call), the latest
configuration data will be returned.
     Otherwise, the GetLatestConfiguration API will not return any data since the
caller already has the latest.
     */
     response = await this.appconfigdata.send(
       new GetLatestConfigurationCommand({
         ConfigurationToken: this.configurationToken
       })
     );
   } catch (err) {
     /*
     Add session restart logic – if the token is invalid or expired, restart the
session and try once more.
     */
     if (err.name === "ResourceNotFoundException" || err.name ===
"BadRequestException") {
       console.warn(
         "Configuration token invalid or expired. Restarting session..."
       );
       await this.startSession();
       response = await this.appconfigdata.send(
         new GetLatestConfigurationCommand({
           ConfigurationToken: this.configurationToken
         })
```

```
        );
      } else {
        throw err;
      }
    }

    /*
    Save the returned configuration token so that it can be passed to the next
 GetLatestConfiguration API call.
    Warning: Not persisting this token for use in the next GetLatestConfiguration
 API call may result in higher
    than expected usage costs.
    */
    this.configurationToken = response.NextPollConfigurationToken;

    /*
    If the GetLatestConfiguration API returned configuration data, update the cached
 configuration with the returned data.
    Otherwise, assume the configuration has not changed, and return the cached
 configuration.
    */
    const configFromApi = response.Configuration
      ? await response.Configuration.transformToString()
      : null;

    if (configFromApi) {
      this.configuration = configFromApi;
      console.log(
        "Configuration contents have changed since the last GetLatestConfiguration
 call, new contents = " +
          this.configuration
      );
    } else {
      console.log(
        "GetLatestConfiguration returned an empty response because we already have
 the latest configuration"
      );
    }

    return this.configuration;
  }
}
```

Limpar o ambiente

Se você executou uma ou mais das amostras de código nesta seção, recomendamos usar uma das amostras a seguir para localizar e excluir os AWS AppConfig recursos criados por essas amostras de código. Os exemplos nesta seção chamam o seguinte APIs:

- ListApplications
- DeleteApplication
- ListEnvironments
- DeleteEnvironments
- ListConfigurationProfiles
- DeleteConfigurationProfile
- ListHostedConfigurationVersions
- DeleteHostedConfigurationVersion

Java

```
/*
    This sample provides cleanup code that deletes all the AWS AppConfig resources
 created in the samples above.

   WARNING: this code will permanently delete the given application and all of its
 sub-resources, including
   configuration profiles, hosted configuration versions, and environments. DO NOT
 run this code against
   an application that you may need in the future.
   */

   public void cleanUpDemoResources() {
       AppConfigClient appconfig = AppConfigClient.create();

       // The name of the application to delete
       // IMPORTANT: verify this name corresponds to the application you wish to
 delete
       String applicationToDelete = "MyDemoApp";


 appconfig.listApplicationsPaginator(ListApplicationsRequest.builder().build()).items().forE
 -> {
```

```
            if (app.name().equals(applicationToDelete)) {
                System.out.println("Deleting App: " + app);
                appconfig.listConfigurationProfilesPaginator(req ->
 req.applicationId(app.id())).items().forEach(cp -> {
                    System.out.println("Deleting Profile: " + cp);
                    appconfig
                        .listHostedConfigurationVersionsPaginator(req -> req
                            .applicationId(app.id())
                            .configurationProfileId(cp.id()))
                        .items()
                        .forEach(hcv -> {
                            System.out.println("Deleting HCV: " + hcv);
                            appconfig.deleteHostedConfigurationVersion(req -> req
                                .applicationId(app.id())
                                .configurationProfileId(cp.id())
                                .versionNumber(hcv.versionNumber()));
                        });
                    appconfig.deleteConfigurationProfile(req -> req
                        .applicationId(app.id())
                        .configurationProfileId(cp.id()));
                });

                appconfig.listEnvironmentsPaginator(req-
>req.applicationId(app.id())).items().forEach(env -> {
                    System.out.println("Deleting Environment: " + env);
                    appconfig.deleteEnvironment(req-
>req.applicationId(app.id()).environmentId(env.id()));
                });

                appconfig.deleteApplication(req -> req.applicationId(app.id()));
            }
        });
    }
```

Python

```
# this sample provides cleanup code that deletes all the AWS AppConfig resources
 created in the samples above.
#
# WARNING: this code will permanently delete the given application and all of its
 sub-resources, including
#   configuration profiles, hosted configuration versions, and environments. DO NOT
 run this code against
```

```
#   an application that you may need in the future.
#

import boto3

# the name of the application to delete
# IMPORTANT: verify this name corresponds to the application you wish to delete
application_name = 'MyDemoApp'

# create and iterate over a list paginator such that we end up with a list of pages,
 which are themselves lists of applications
# e.g. [ [{'Name':'MyApp1',...},{'Name':'MyApp2',...}], [{'Name':'MyApp3',...}] ]
list_of_app_lists = [page['Items'] for page in
 appconfig.get_paginator('list_applications').paginate()]
# retrieve the target application from the list of lists
application = [app for apps in list_of_app_lists for app in apps if app['Name'] ==
 application_name][0]
print(f"deleting application {application['Name']} (id={application['Id']})")

# delete all configuration profiles
list_of_config_lists = [page['Items'] for page in
 appconfig.get_paginator('list_configuration_profiles').paginate(ApplicationId=application['
for config_profile in [config for configs in list_of_config_lists for config in
 configs]:
    print(f"\tdeleting configuration profile {config_profile['Name']}
 (Id={config_profile['Id']})")

    # delete all hosted configuration versions
    list_of_hcv_lists = [page['Items'] for page in
 appconfig.get_paginator('list_hosted_configuration_versions').paginate(ApplicationId=applic
 ConfigurationProfileId=config_profile['Id'])]
    for hcv in [hcv for hcvs in list_of_hcv_lists for hcv in hcvs]:

 appconfig.delete_hosted_configuration_version(ApplicationId=application['Id'],
 ConfigurationProfileId=config_profile['Id'], VersionNumber=hcv['VersionNumber'])
        print(f"\t\tdeleted hosted configuration version {hcv['VersionNumber']}")

    # delete the config profile itself
    appconfig.delete_configuration_profile(ApplicationId=application['Id'],
 ConfigurationProfileId=config_profile['Id'])
    print(f"\tdeleted configuration profile {config_profile['Name']}
 (Id={config_profile['Id']})")

# delete all environments
```

```
list_of_env_lists = [page['Items'] for page in
 appconfig.get_paginator('list_environments').paginate(ApplicationId=application['Id'])]
for environment in [env for envs in list_of_env_lists for env in envs]:
    appconfig.delete_environment(ApplicationId=application['Id'],
 EnvironmentId=environment['Id'])
    print(f"\tdeleted environment {environment['Name']} (Id={environment['Id']})")

# delete the application itself
appconfig.delete_application(ApplicationId=application['Id'])
print(f"deleted application {application['Name']} (id={application['Id']})")
```

JavaScript

```
// this sample provides cleanup code that deletes all the AWS AppConfig resources
 created in the samples above.

// WARNING: this code will permanently delete the given application and all of its
 sub-resources, including
//   configuration profiles, hosted configuration versions, and environments. DO NOT
 run this code against
//   an application that you may need in the future.

import {
  AppConfigClient,
  paginateListApplications,
  DeleteApplicationCommand,
  paginateListConfigurationProfiles,
  DeleteConfigurationProfileCommand,
  paginateListHostedConfigurationVersions,
  DeleteHostedConfigurationVersionCommand,
  paginateListEnvironments,
  DeleteEnvironmentCommand,
} from "@aws-sdk/client-appconfig";

const client = new AppConfigClient();

// the name of the application to delete
// IMPORTANT: verify this name corresponds to the application you wish to delete
const application_name = "MyDemoApp";

// iterate over all applications, deleting ones that have the name defined above
for await (const app_page of paginateListApplications({ client }, {})) {
  for (const application of app_page.Items) {
```

```
    // skip applications that dont have the name thats set
    if (application.Name !== application_name) continue;

    console.log( `deleting application ${application.Name} (id=${application.Id})`);

    // delete all configuration profiles
    for await (const config_page of paginateListConfigurationProfiles({ client },
 { ApplicationId: application.Id })) {
      for (const config_profile of config_page.Items) {
        console.log(`\tdeleting configuration profile ${config_profile.Name} (Id=
${config_profile.Id})`);

        // delete all hosted configuration versions
        for await (const hosted_page of
 paginateListHostedConfigurationVersions({ client },
          { ApplicationId: application.Id, ConfigurationProfileId:
 config_profile.Id }
        )) {
          for (const hosted_config_version of hosted_page.Items) {
            await client.send(
              new DeleteHostedConfigurationVersionCommand({
                ApplicationId: application.Id,
                ConfigurationProfileId: config_profile.Id,
                VersionNumber: hosted_config_version.VersionNumber,
              })
            );
            console.log(`\t\tdeleted hosted configuration version
 ${hosted_config_version.VersionNumber}`);
          }
        }

        // delete the config profile itself
        await client.send(
          new DeleteConfigurationProfileCommand({
            ApplicationId: application.Id,
            ConfigurationProfileId: config_profile.Id,
          })
        );
        console.log(`\tdeleted configuration profile ${config_profile.Name} (Id=
${config_profile.Id})`)
      }

      // delete all environments
```

```
      for await (const env_page of paginateListEnvironments({ client },
 { ApplicationId: application.Id })) {
        for (const environment of env_page.Items) {
          await client.send(
            new DeleteEnvironmentCommand({
              ApplicationId: application.Id,
              EnvironmentId: environment.Id,
            })
          );
          console.log(`\tdeleted environment ${environment.Name} (Id=
${environment.Id})`)
        }
      }
    }

    // delete the application itself
    await client.send(
      new DeleteApplicationCommand({ ApplicationId: application.Id })
    );
    console.log(`deleted application ${application.Name} (id=${application.Id})`)
  }
}
```

Configurando a proteção AWS AppConfig contra exclusão

AWS AppConfig fornece uma configuração de conta para ajudar a impedir que os usuários excluam acidentalmente ambientes e perfis de configuração usados ativamente. AWS AppConfig monitora chamadas para GetLatestConfigurationGetConfiguratione rastreia quais perfis e ambientes de configuração foram incluídos nessas chamadas em um intervalo de 60 minutos (a configuração padrão). Os ambientes ou perfiis de configuração que tenham sido acessados dentro desse intervalo serão considerados ativos. Se você tentar excluir um perfil ou ambiente de configuração ativo, AWS AppConfig retornará um erro. Se necessário, você pode ignorar esse erro usando o parâmetro `DeletionProtectionCheck`. Para obter mais informações, consulte Ignorar ou forçar uma verificação de proteção contra exclusão.

Para configurar a proteção contra exclusão usando o console

Use o procedimento a seguir para configurar a proteção contra exclusão usando o console do AWS Systems Manager .

Para configurar a proteção contra exclusão (console)

1. Abra o AWS Systems Manager console em https://console.aws.amazon.com/systems-manager/appconfig/.
2. No painel de navegação, selecione Configurações.
3. Use o botão de alternância para ativar ou desativar a proteção contra exclusão.
4. Para Período de proteção, configure a definição de um recurso ativo entre 15 e 1.440 minutos.
5. Clique em Aplicar.

Configure a proteção contra exclusão usando o AWS CLI

Use o procedimento a seguir para configurar a proteção contra exclusão usando a AWS CLI. *`value`*Substitua os comandos a seguir pelo valor que você deseja usar em seu ambiente.

> **Note**
>
> Antes de começar, recomendamos que você atualize para a versão mais recente da AWS CLI. Para obter mais informações, consulte Instalar ou atualizar para a versão mais recente da AWS CLI, no Guia do usuário da AWS Command Line Interface .

Para configurar a proteção contra exclusão (CLI)

1. Para visualizar as configurações de proteção contra exclusão atuais, execute o comando a seguir.

```
aws appconfig get-account-settings
```

2. Execute o comando a seguir para ativar ou desativar a proteção contra exclusão. Especifique `false` para desativar a proteção contra exclusão ou `true` para ativá-la.

```
aws appconfig update-account-settings --deletion-protection Enabled=value
```

3. É possível aumentar o intervalo padrão para no máximo 24 horas. Execute o comando a seguir para especificar um novo intervalo.

```
aws appconfig update-account-settings --deletion-protection
 Enabled=true,ProtectionPeriodInMinutes=a number between 15 and 1440
```

Ignorar ou forçar uma verificação de proteção contra exclusão

Para ajudá-lo a gerenciar a proteção contra exclusão, o DeleteEnvironmente DeleteConfigurationProfile APIs inclua um parâmetro chamado`DeletionProtectionCheck`. Esse parâmetro comporta os seguintes valores:

- `BYPASS`: instrui AWS AppConfig a ignorar a verificação de proteção contra exclusão e excluir um perfil de configuração, mesmo que a proteção contra exclusão o tivesse impedido.
- `APPLY`: instrui a verificação da proteção contra exclusão a ser executada, mesmo se a proteção contra exclusão estiver desabilitada em nível de conta. O `APPLY` também força a verificação de proteção contra exclusão a ser executada em recursos criados na última hora, que normalmente são excluídos das verificações de proteção contra exclusão.
- `ACCOUNT_DEFAULT`: a configuração padrão, que instrui o AWS AppConfig a implementar o valor de proteção contra exclusão especificado na API `UpdateAccountSettings`.

> **Note**
>
> Por padrão, o `DeletionProtectionCheck` ignora perfis de configuração e ambientes criados na última hora. A configuração padrão tem como objetivo evitar

que a proteção contra exclusão interfira nos testes e nas demonstrações que criam recursos de curta duração. É possível substituir esse comportamento transmitindo `DeletionProtectionCheck=APPLY` ao chamar `DeleteEnvironment` ou `DeleteConfigurationProfile`.

O passo a passo da CLI a seguir utiliza exemplos de comandos para ilustrar como usar o parâmetro `DeletionProtectionCheck`. *ID*Substitua os comandos a seguir pelo ID dos seus AWS AppConfig artefatos.

1. Solicite GetLatestConfigurationuma configuração implantada.

```
aws appconfigdata get-latest-configuration --configuration-token $(aws
 appconfigdata start-configuration-session --application-identifier ID --
environment-identifier ID --configuration-profile-identifier ID --query
 InitialConfigurationToken) outfile.txt
```

2. Aguarde 60 segundos AWS AppConfig para registrar que a configuração está ativa.
3. Execute o comando a seguir para chamar DeleteEnvironmente aplicar a proteção contra exclusão no ambiente.

```
aws appconfig delete-environment --environment-id ID --application-id ID --
deletion-protection-check APPLY
```

   O comando deve retornar o erro a seguir.

```
An error occurred (BadRequestException) when calling the DeleteEnvironment
 operation: Environment Beta is actively being used in your application and cannot
 be deleted.
```

4. Execute o comando a seguir para ignorar a proteção contra exclusão e excluir o ambiente.

```
aws appconfig delete-environment --environment-id ID --application-id ID --
deletion-protection-check BYPASS
```

Segurança em AWS AppConfig

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

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

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços no Nuvem AWS. AWS também fornece serviços que você pode usar com segurança. 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 aplicáveis AWS Systems Manager, consulte AWS Serviços no escopo do programa de conformidade AWS .
- Segurança na nuvem — Sua responsabilidade é determinada pelo AWS serviço que você usa. Você também é responsável por outros fatores, incluindo a confidencialidade de seus dados, os requisitos da empresa e as leis e regulamentos aplicáveis.

AWS AppConfig é uma ferramenta em AWS Systems Manager. Para entender como aplicar o modelo de responsabilidade compartilhada ao usar AWS AppConfig, consulte Segurança em AWS Systems Manager. Essa seção descreve como configurar o Systems Manager para atingir os objetivos de segurança e conformidade do AWS AppConfig.

Implemente o acesso de privilégio mínimo

Como prática recomendada de segurança, conceda as permissões mínimas exigidas pelas identidades para realizar ações específicas em recursos específicos sob condições específicas. AWS AppConfig O agente oferece dois recursos que permitem que o agente acesse o sistema de arquivos de uma instância ou contêiner: backup e gravação em disco. Se você habilitar esses recursos, verifique se somente o AWS AppConfig Agente tem permissões para gravar nos arquivos de configuração designados no sistema de arquivos. Verifique também se somente os processos necessários para ler esses arquivos de configuração têm a capacidade de fazer isso. A implementação do privilégio de acesso mínimo é fundamental para reduzir o risco de segurança e o impacto que pode resultar de erros ou usuários mal-intencionados.

Para obter mais informações sobre como implementar o acesso com privilégios mínimos, consulte SEC03- BP02 Conceder acesso com privilégios mínimos no Guia do AWS Well-Architected Tool Usuário. Para obter mais informações sobre os recursos do AWS AppConfig Agente mencionados nesta seção, consulteUsar um manifesto para habilitar recursos de recuperação adicionais.

Criptografia de dados em repouso para AWS AppConfig

AWS AppConfig fornece criptografia por padrão para proteger os dados do cliente em repouso usando Chaves pertencentes à AWS.

Chaves pertencentes à AWS— AWS AppConfig usa essas chaves por padrão para criptografar automaticamente os dados implantados pelo serviço e AWS AppConfig hospedados no armazenamento de dados. Você não pode visualizar, gerenciar Chaves pertencentes à AWS, usar ou auditar seu uso. No entanto, não é necessário realizar nenhuma ação nem alterar qualquer programa para proteger as chaves que criptografam seus dados. Para saber mais, consulte Chaves pertencentes à AWS no Guia do desenvolvedor do AWS Key Management Service .

Embora não seja possível desativar essa camada de criptografia ou selecionar um tipo alternativo de criptografia, você pode especificar uma chave gerenciada pelo cliente a ser usada ao salvar dados de configuração hospedados no armazenamento de AWS AppConfig dados e ao implantar seus dados de configuração.

Chaves gerenciadas pelo cliente — AWS AppConfig suporta o uso de uma chave simétrica gerenciada pelo cliente que você cria, possui e gerencia para adicionar uma segunda camada de criptografia sobre a existente Chave pertencente à AWS. Por ter controle total dessa camada de criptografia, você pode realizar tarefas como:

- Estabelecer e manter as políticas e concessões de chaves
- Estabelecer e manter políticas do IAM
- Ativar e desativar políticas de chaves
- Alternar os materiais de criptografia de chave
- Adicionar etiquetas
- Criar réplicas de chaves
- Programar chaves para exclusão

Para obter mais informações, consulte Chave gerenciada pelo cliente no Guia do desenvolvedor do AWS Key Management Service .

AWS AppConfig suporta chaves gerenciadas pelo cliente

AWS AppConfig oferece suporte para criptografia de chave gerenciada pelo cliente para dados de configuração. Para versões de configuração salvas no armazenamento de dados AWS AppConfig hospedado, os clientes podem definir um `KmsKeyIdentifier` no perfil de configuração correspondente. Cada vez que uma nova versão dos dados de configuração é criada usando a operação da `CreateHostedConfigurationVersion` API, AWS AppConfig gera uma chave de AWS KMS dados do `KmsKeyIdentifier` para criptografar os dados antes de armazená-los. Quando os dados são acessados posteriormente, seja durante as operações da `StartDeployment` API `GetHostedConfigurationVersion` ou da API, AWS AppConfig descriptografa os dados de configuração usando informações sobre a chave de dados gerada.

AWS AppConfig também oferece suporte para criptografia de chave gerenciada pelo cliente para dados de configuração implantados. Para criptografar os dados de configuração, os clientes podem fornecer um `KmsKeyIdentifier` para sua implantação. AWS AppConfig gera a chave AWS KMS de dados com isso `KmsKeyIdentifier` para criptografar dados na operação da `StartDeployment` API.

AWS AppConfig acesso à criptografia

Ao criar uma chave gerenciada pelo cliente, siga esta política de chaves para garantir que a chave possa ser usada.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Allow use of the key",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:role/role_name"
            },
            "Action": [
                "kms:Decrypt",
                "kms:GenerateDataKey"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

Para criptografar dados de configuração hospedados com uma chave gerenciada pelo cliente, a chamada de identidade `CreateHostedConfigurationVersion` precisa da seguinte declaração de política, que pode ser atribuída a um usuário, grupo ou função:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "kms:GenerateDataKey",
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-ID"
        }
    ]
}
```

Se estiver usando um segredo do Secrets Manager ou qualquer outro dado de configuração criptografado com uma chave gerenciada pelo cliente, o `retrievalRoleArn` precisará de `kms:Decrypt` para descriptografar e recuperar os dados.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "kms:Decrypt",
      "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-ID"
    }
  ]
}
```

Ao chamar a operação da AWS AppConfig StartDeploymentAPI, a chamada de identidade `StartDeployment` precisa da seguinte política do IAM, que pode ser atribuída a um usuário, grupo ou função:

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:GenerateDataKey*"
      ],
      "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-ID"
    }
  ]
}
```

Ao chamar a operação da AWS AppConfig GetLatestConfigurationAPI, a chamada de identidade `GetLatestConfiguration` precisa da seguinte política, que pode ser atribuída a um usuário, grupo ou função:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "kms:Decrypt",
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/key-ID"
        }
    ]
}
```

Contexto de criptografia

Um contexto de criptografia é um conjunto opcional de pares chave-valor que pode conter informações contextuais adicionais sobre os dados.

AWS KMS usa o contexto de criptografia como dados autenticados adicionais para oferecer suporte à criptografia autenticada. Quando você inclui um contexto de criptografia em uma solicitação para criptografar dados, AWS KMS vincula o contexto de criptografia aos dados criptografados. Para descriptografar os dados, você inclui o mesmo contexto de criptografia na solicitação.

AWS AppConfig contexto de criptografia: AWS AppConfig usa um contexto de criptografia em todas as operações AWS KMS criptográficas para implantações e dados de configuração hospedados criptografados. O contexto contém uma chave correspondente ao tipo de dados e um valor que identifica o dado específico.

Monitorando suas chaves de criptografia para AWS

Ao usar chaves gerenciadas pelo AWS KMS cliente com AWS AppConfig, você pode usar AWS CloudTrail o Amazon CloudWatch Logs para rastrear solicitações AWS AppConfig enviadas para AWS KMS.

O exemplo a seguir é um CloudTrail evento `Decrypt` para monitorar AWS KMS operações chamadas por AWS AppConfig para acessar dados criptografados pela chave gerenciada pelo cliente:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "appconfig.amazonaws.com"
    },
    "eventTime": "2023-01-03T02:22:28z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "Region",
    "sourceIPAddress": "172.12.34.56",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "encryptionContext": {
            "aws:appconfig:deployment:arn":
 "arn:aws:appconfig:Region:account_ID:application/application_ID/
environment/environment_ID/deployment/deployment_ID"
        },
        "keyId": "arn:aws:kms:Region:account_ID:key/key_ID",
```

```
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "account_ID",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:Region:account_ID:key_ID"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "eventCategory": "Management",
    "recipientAccountId": "account_ID",
    "sharedEventID": "dc129381-1d94-49bd-b522-f56a3482d088"
}
```

Acesso AWS AppConfig usando um endpoint de interface ()AWS PrivateLink

Você pode usar AWS PrivateLink para criar uma conexão privada entre sua VPC e. AWS AppConfig Você pode acessar AWS AppConfig como se estivesse em sua VPC, sem o uso de um gateway de internet, dispositivo NAT, conexão VPN ou conexão. Direct Connect As instâncias na sua VPC não precisam de endereços IP públicos para acessar o AWS AppConfig.

Estabeleça essa conectividade privada criando um endpoint de interface, habilitado 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 AppConfig.

Para saber mais, consulte Acessar os Serviços da AWS pelo AWS PrivateLink no Guia do AWS PrivateLink .

Considerações para AWS AppConfig

Antes de configurar um endpoint de interface para AWS AppConfig, consulte as Considerações no AWS PrivateLink Guia.

AWS AppConfig suporta a realização de chamadas para os `appconfigdata`serviços `appconfig`e por meio do endpoint da interface.

Crie um endpoint de interface para AWS AppConfig

Você pode criar um endpoint de interface para AWS AppConfig 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 AppConfig usar os seguintes nomes de serviço:

```
com.amazonaws.region.appconfig
```

```
com.amazonaws.region.appconfigdata
```

Se você habilitar o DNS privado para o endpoint da interface, poderá fazer solicitações de API a AWS AppConfig usando seu nome DNS regional padrão. Por exemplo, `appconfig.us-east-1.amazonaws.com` e `appconfigdata.us-east-1.amazonaws.com`.

Criar uma política de endpoint para o endpoint de interface

Uma política de endpoint é um recurso do IAM que pode ser anexado ao endpoint de interface. A política de endpoint padrão permite acesso total AWS AppConfig por meio do endpoint da interface. Para controlar o acesso AWS AppConfig permitido pela sua VPC, anexe uma política de endpoint personalizada ao endpoint da interface.

Uma política de endpoint especifica as seguintes informações:

- As entidades principais que podem realizar ações (Contas da AWS, usuários do IAM e perfis do IAM).
- As ações que podem ser realizadas.
- Os recursos nos quais as ações podem ser executadas.

Para obter mais informações, consulte Controlar o acesso aos serviços usando políticas de endpoint no Guia do AWS PrivateLink .

Exemplo: política de VPC endpoint para ações AWS AppConfig

O exemplo a seguir refere-se a uma política de endpoint personalizada. Quando anexada ao endpoint da sua interface, essa política concede acesso às ações do AWS AppConfig listadas para todas as entidades principais em todos os recursos.

```
{
   "Statement": [
      {
         "Principal": "*",
         "Effect": "Allow",
         "Action": [
            "appconfig:CreateApplication",
            "appconfig:CreateEnvironment",
            "appconfig:CreateConfigurationProfile",
            "appconfig:StartDeployment",
            "appconfig:GetLatestConfiguration"
            "appconfig:StartConfigurationSession"
         ],
         "Resource":"*"
      }
   ]
}
```

Alternância de chaves do Secrets Manager

Esta seção descreve informações de segurança importantes sobre a AWS AppConfig integração com o Secrets Manager. Para obter informações sobre o Secrets Manager, consulte O que é AWS Secrets Manager? no Guia do AWS Secrets Manager usuário.

Configurando a rotação automática dos segredos do Secrets Manager implantados pelo AWS AppConfig

Alternância é o processo de atualizar periodicamente um segredo armazenado no Secrets Manager. Quando o Secrets Manager alterna um segredo, ele atualiza as credenciais tanto no segredo quanto no banco de dados ou serviço. Você pode configurar a rotação automática de segredos no Secrets Manager usando uma AWS Lambda função para atualizar o segredo e o banco de dados. Para obter mais informações, consulte Alternar os segredos do AWS Secrets Manager no Guia do usuário do AWS Secrets Manager .

Para ativar a rotação de chaves dos segredos do Secrets Manager implantados por AWS AppConfig, atualize sua função de rotação do Lambda e implante o segredo rotacionado.

> **Note**
>
> Implante seu perfil de AWS AppConfig configuração depois que seu segredo for alterado e totalmente atualizado para a nova versão. Você pode determinar se o segredo foi alternado porque o status de `VersionStage` muda de `AWSPENDING` para `AWSCURRENT`. A conclusão da alternância de segredos ocorre dentro da função `finish_secret` dos modelos de alternância do Secrets Manager.

Aqui está um exemplo de função que inicia uma AWS AppConfig implantação após a rotação de um segredo.

```
import time
import boto3
client = boto3.client('appconfig')

def finish_secret(service_client, arn, new_version):
    """Finish the rotation by marking the pending secret as current
    This method finishes the secret rotation by staging the secret staged AWSPENDING
 with the AWSCURRENT stage.
    Args:
        service_client (client): The secrets manager service client
        arn (string): The secret ARN or other identifier
        new_version (string): The new version to be associated with the secret
    """
    # First describe the secret to get the current version
    metadata = service_client.describe_secret(SecretId=arn)
    current_version = None
    for version in metadata["VersionIdsToStages"]:
        if "AWSCURRENT" in metadata["VersionIdsToStages"][version]:
            if version == new_version:
                # The correct version is already marked as current, return
                logger.info("finishSecret: Version %s already marked as AWSCURRENT for
 %s" % (version, arn))
                return
            current_version = version
            break

    # Finalize by staging the secret version current
    service_client.update_secret_version_stage(SecretId=arn, VersionStage="AWSCURRENT",
 MoveToVersionId=new_version, RemoveFromVersionId=current_version)
```

```
    # Deploy rotated secret
    response = client.start_deployment(
            ApplicationId='TestApp',
            EnvironmentId='TestEnvironment',
            DeploymentStrategyId='TestStrategy',
            ConfigurationProfileId='ConfigurationProfileId',
            ConfigurationVersion=new_version,
            KmsKeyIdentifier=key,
            Description='Deploy secret rotated at ' + str(time.time())
        )

    logger.info("finishSecret: Successfully set AWSCURRENT stage to version %s for
secret %s." % (new_version, arn))
```

Monitoramento AWS AppConfig

O monitoramento é uma parte importante da manutenção da confiabilidade, disponibilidade e desempenho de AWS AppConfig suas outras AWS soluções. AWS fornece as seguintes ferramentas de monitoramento para observar AWS AppConfig, relatar quando algo está errado e realizar ações automáticas quando apropriado:

- A Amazon CloudWatch monitora seus AWS recursos e os aplicativos em que você executa AWS em tempo real. Você pode coletar e rastrear métricas, criar painéis personalizados e definir alarmes que o notificam ou que realizam ações quando uma métrica especificada atinge um limite definido. Por exemplo, você pode CloudWatch rastrear o uso da CPU ou outras métricas de suas instâncias do Amazon EC2 e iniciar automaticamente novas instâncias quando necessário. Para obter mais informações, consulte o Guia CloudWatch do usuário da Amazon.
- AWS CloudTrailcaptura chamadas de API e eventos relacionados feitos por ou em nome de sua AWS conta e entrega os arquivos de log para um bucket do Amazon S3 que você especificar. Você pode identificar quais usuários e contas ligaram AWS, o endereço IP de origem a partir do qual as chamadas foram feitas e quando elas ocorreram. Para obter mais informações, consulte o Guia do usuário do AWS CloudTrail.
- O Amazon CloudWatch Logs permite que você monitore, armazene e acesse seus arquivos de log das instâncias do Amazon EC2 e de outras fontes. CloudTrail CloudWatch Os registros podem monitorar as informações nos arquivos de log e notificá-lo quando determinados limites forem atingidos. É possível também arquivar seus dados de log em armazenamento resiliente. Para obter mais informações, consulte o Guia do usuário do Amazon CloudWatch Logs.
- A Amazon EventBridge pode ser usada para automatizar seus AWS serviços e responder automaticamente a eventos do sistema, como problemas de disponibilidade de aplicativos ou alterações de recursos. Os eventos dos AWS serviços são entregues quase EventBridge em tempo real. Você pode escrever regras simples para determinar quais eventos são do seu interesse, e as ações automatizadas a serem tomadas quando um evento corresponder à regra. Para obter mais informações, consulte o Guia EventBridge do usuário da Amazon.

Tópicos

- Registrando chamadas de AWS AppConfig API usando AWS CloudTrail
- Métricas de registro para chamadas AWS AppConfig de planos de dados
- Monitorar implantações para reversão automática

Registrando chamadas de AWS AppConfig API usando AWS CloudTrail

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

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

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

CloudTrail está ativo Conta da AWS quando você cria a conta e você tem acesso automático ao histórico de CloudTrail eventos. O histórico de CloudTrail eventos fornece um registro visível, pesquisável, baixável e imutável dos últimos 90 dias de eventos de gerenciamento registrados em um. Região da AWS Para obter mais informações, consulte Trabalhando com o histórico de CloudTrail eventos no Guia AWS CloudTrail do usuário. Não há CloudTrail cobrança pela visualização do histórico de eventos.

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

CloudTrail trilhas

Uma trilha permite CloudTrail entregar arquivos de log para um bucket do Amazon S3. Todas as trilhas criadas usando o Console de gerenciamento da AWS são multirregionais. Só é possível criar uma trilha de região única ou de várias regiões usando a AWS CLI. É recomendável criar uma trilha multirregional porque você captura todas as atividades Regiões da AWS em sua conta. Ao criar uma trilha de região única, é possível visualizar somente os eventos registrados na Região da AWS da trilha. Para obter mais informações sobre trilhas, consulte Criar uma trilha

para a Conta da AWS e Criar uma trilha para uma organização no Guia do usuário do AWS CloudTrail .

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

CloudTrail Armazenamentos de dados de eventos em Lake

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

CloudTrail Os armazenamentos e consultas de dados de eventos em Lake incorrem em custos. Ao criar um armazenamento de dados de eventos, você escolhe a opção de preço que deseja usar para ele. A opção de preço determina o custo para a ingestão e para o armazenamento de eventos, e o período de retenção padrão e máximo para o armazenamento de dados de eventos. Para obter mais informações sobre CloudTrail preços, consulte AWS CloudTrail Preços.

AWS AppConfig eventos de dados em CloudTrail

Os eventos de dados fornecem informações sobre as operações de recursos realizadas em ou em um recurso (por exemplo, recuperar a configuração mais recente implantada por meio de uma chamada GetLatestConfiguration). Também são conhecidas como operações de plano de dados. Os eventos de dados costumam ser atividades de alto volume. Por padrão, CloudTrail não registra eventos de dados. O histórico de CloudTrail eventos não registra eventos de dados.

Há cobranças adicionais para eventos de dados. Para obter mais informações sobre CloudTrail preços, consulte AWS CloudTrail Preços.

Você pode registrar eventos de dados para os tipos de AWS AppConfig recursos usando o CloudTrail console ou AWS CLI as operações CloudTrail da API. A tabela nesta seção mostra os tipos de recursos disponíveis para AWS AppConfig.

- Para registrar eventos de dados usando o CloudTrail console, crie um armazenamento de dados de trilhas ou eventos para registrar eventos de dados ou atualize um armazenamento de dados de trilhas ou eventos existente para registrar eventos de dados.

  1. Selecione Eventos de dados para registrar eventos de dados em log.
  2. Na lista Tipo de evento de dados, selecione AWS AppConfig.
  3. Selecione o modelo de seletor de logs que deseja usar. Você pode registrar todos os eventos de dados para o tipo de recurso, registrar todos os eventos `readOnly`, registrar todos os eventos `writeOnly` ou criar um modelo de seletor de logs personalizado para filtrar os campos `readOnly`, `eventName` e `resources.ARN`.
  4. Em Nome do seletor, insira AppConfigDataEvents. Para obter informações sobre como habilitar o Amazon CloudWatch Logs para sua trilha de eventos de dados, consulteMétricas de registro para chamadas AWS AppConfig de planos de dados.

- Para registrar eventos de dados usando o AWS CLI, configure o `--advanced-event-selectors` parâmetro para definir o `eventCategory` campo igual `Data` e o `resources.type` campo igual ao valor do tipo de recurso (consulte a tabela). É possível adicionar condições para filtrar os valores dos campos `readOnly`, `eventName` e `resources.ARN`.
  - Para configurar uma trilha e registrar eventos de dados em log, execute o comando put-event-selectors. Para ter mais informações, consulte Logging data events for trails with the AWS CLI.
  - Para configurar um armazenamento de dados de eventos e registrar eventos de dados em log, execute o comando create-event-data-store para criar um armazenamento e registrar eventos de dados em log ou execute o comando update-event-data-store para atualizar um armazenamento existente. Para obter mais informações, consulte Log em log de eventos de dados para trilhas AWS CLI.

A tabela a seguir lista os tipos de AWS AppConfig recursos. A coluna Tipo de evento de dados (console) mostra o valor a ser escolhido na lista Tipo de evento de dados no CloudTrail console. A coluna de valor resources.type mostra o `resources.type` valor, que você especificaria ao configurar seletores de eventos avançados usando o ou. AWS CLI CloudTrail APIs A CloudTrail coluna Dados APIs registrados em mostra as chamadas de API registradas CloudTrail para o tipo de recurso.

Tipo de evento de dados (console)	valor resources.type	Dados APIs registrados em * CloudTrail
AWS AppConfig	`AWS::AppConfig::Configuration`	• GetLatestConfiguration • StartConfigurationSession

*É possível configurar seletores de eventos avançados para filtrar os campos `eventName`, `readOnly` e `resources.ARN` e registrar em log somente os eventos que são importantes para você. Consulte mais informações sobre esses campos em AdvancedFieldSelector.

AWS AppConfig eventos de gerenciamento em CloudTrail

AWS AppConfig registra todas as operações do plano de AWS AppConfig controle como eventos de gerenciamento. Para ver uma lista das operações do plano de AWS AppConfig controle AWS AppConfig registradas CloudTrail, consulte a Referência da AWS AppConfig API.

AWS AppConfig exemplos de eventos

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

O exemplo a seguir mostra um CloudTrail evento que demonstra a StartConfigurationSessionoperação.


```
{
      "eventVersion": "1.09",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/Administrator",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
          "sessionIssuer": {},
          "attributes": {
            "creationDate": "2024-01-11T14:37:02Z",
```

```
            "mfaAuthenticated": "false"
          }
        }
      },
      "eventTime": "2024-01-11T14:45:15Z",
      "eventSource": "appconfig.amazonaws.com",
      "eventName": "StartConfigurationSession",
      "awsRegion": "us-east-1",
      "sourceIPAddress": "203.0.113.0",
      "userAgent": "Boto3/1.34.11 md/Botocore#1.34.11 ua/2.0 os/macos#22.6.0
 md/arch#x86_64 lang/python#3.11.4 md/pyimpl#CPython cfg/retry-mode#legacy
 Botocore/1.34.11",
      "requestParameters": {
        "applicationIdentifier": "rrfexample",
        "environmentIdentifier": "mexampleqe0",
        "configurationProfileIdentifier": "3eexampleu1"
      },
      "responseElements": null,
      "requestID": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
      "eventID": "a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE",
      "readOnly": false,
      "resources": [
        {
          "accountId": "123456789012",
          "type": "AWS::AppConfig::Configuration",
          "ARN": "arn:aws:appconfig:us-east-1:123456789012:application/rrfexample/
environment/mexampleqe0/configuration/3eexampleu1"
        }
      ],
      "eventType": "AwsApiCall",
      "managementEvent": false,
      "recipientAccountId": "123456789012",
      "eventCategory": "Data",
      "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "appconfigdata.us-east-1.amazonaws.com"
      }
    }
```


Para obter informações sobre o conteúdo do CloudTrail registro, consulte o conteúdo do CloudTrail registro no Guia AWS CloudTrail do usuário.

Métricas de registro para chamadas AWS AppConfig de planos de dados

Se você configurou AWS CloudTrail para registrar eventos de AWS AppConfig dados, você pode permitir que o Amazon CloudWatch Logs registre métricas para chamadas para o plano de AWS AppConfig dados. Em seguida, você pode pesquisar e filtrar dados de registro no CloudWatch Logs criando um ou mais filtros de métrica. Os filtros de métricas definem os termos e padrões a serem procurados nos dados de registro à medida que são enviados para o CloudWatch Logs. CloudWatch O Logs usa filtros métricos para transformar dados de registro em CloudWatch métricas numéricas. É possível representar graficamente as métricas ou configurá-las com um alarme.

Antes de começar

Ative o registro de eventos de AWS AppConfig dados em AWS CloudTrail. O procedimento a seguir descreve como ativar o registro de métricas para uma AWS AppConfig trilha existente CloudTrail. Para obter informações sobre como habilitar o CloudTrail registro para chamadas AWS AppConfig de planos de dados, consulteAWS AppConfig eventos de dados em CloudTrail.

Use o procedimento a seguir para permitir que o CloudWatch Logs registre métricas para chamadas para o plano AWS AppConfig de dados.

Para permitir que o CloudWatch Logs registre métricas de chamadas para o plano AWS AppConfig de dados

1. Abra o CloudTrail console em https://console.aws.amazon.com/cloudtrail/.
2. No painel, escolha sua AWS AppConfig trilha.
3. Na seção CloudWatch Logs, selecione Editar.
4. Selecione Ativado.
5. Em Nome do grupo de logs, deixe o nome padrão ou insira um nome. Anote o nome. Posteriormente, você escolherá o grupo de CloudWatch registros no console de registros.
6. Em Nome do perfil, insira um nome.
7. Escolha Salvar alterações.

Use o procedimento a seguir para criar uma métrica e um filtro de métrica para o AWS AppConfig CloudWatch Logs. O procedimento descreve como criar um filtro de métricas para chamadas por `operation` e (opcionalmente) chamadas por `operation` e `Amazon Resource Name (ARN)`.

Para criar uma métrica e um filtro de métrica para o AWS AppConfig CloudWatch Logs

1. Abra o CloudWatch console emhttps://console.aws.amazon.com/cloudwatch/.
2. No painel de navegação, escolha Logs e, em seguida, escolha Grupos de logs.
3. Escolha a caixa de seleção ao lado do grupo de AWS AppConfig registros.
4. Escolha Actions (Ações) e Create metric filter (Criar filtro de métrica).
5. Em Nome do filtro, insira um nome.
6. Em Padrão de filtro, insira o seguinte:

```
{ $.eventSource = "appconfig.amazonaws.com" }
```

7. (Opcional) Na seção Testar padrão, selecione seu grupo de logs na lista Selecionar dados de log para testar. Se CloudTrail não tiver registrado nenhuma chamada, você pode pular esta etapa.
8. Escolha Próximo.
9. Em Namespace da métrica, digite **`AWS AppConfig`**.
10. Em Metric name (Nome da métrica), insira **`Calls`**.
11. Em Valor de métrica, insira **`1`**.
12. Ignore o Valor padrão e a Unidade.
13. Em Nome da dimensão, insira **`operation`**.
14. Em Valor da dimensão, insira **`$.eventName`**.

    (Opcional) É possível inserir uma segunda dimensão que inclua o nome do recurso da Amazon (ARN) que está fazendo a chamada. Para adicionar uma segunda dimensão, em Nome da dimensão, insira **`resource`**. Em Valor da dimensão, insira **`$.resources[0].ARN`**.

    Escolha Próximo.
15. Revise os detalhes do filtro e selecione Criar filtro de métrica.

(Opcional) Você pode repetir esse procedimento para criar um novo filtro métrico para um código de erro específico, como AccessDenied. Em caso afirmativo, insira os seguintes detalhes:

1. Em Nome do filtro, insira um nome.
2. Em Padrão de filtro, insira o seguinte:

```
{ $.errorCode = "codename" }
```

Por exemplo

```
{ $.errorCode = "AccessDenied" }
```

3. Em Namespace da métrica, digite **`AWS AppConfig`**.
4. Em Metric name (Nome da métrica), insira **`Errors`**.
5. Em Valor de métrica, insira **`1`**.
6. Em Valor padrão, insira um zero (0).
7. Ignore Unidade, Dimensões e Alarmes.

Depois de CloudTrail registrar as chamadas de API, você pode ver as métricas em CloudWatch. Para obter mais informações, consulte Visualizar suas métricas e registros no console no Guia do CloudWatch usuário da Amazon. Para ter informações sobre como localizar uma métrica que você criou, consulte Procurar por métricas disponíveis.

> **Note**
>
> Se você configurar a métrica de erro sem dimensão, conforme descrito aqui, poderá visualizar essas métricas na página Métricas sem dimensões.

Criando um alarme para uma CloudWatch métrica

Depois de criar métricas, você pode criar alarmes métricos em CloudWatch. Por exemplo, é possível criar um alarme para a métrica de chamadas do AWS AppConfig que você criou no procedimento anterior. Especificamente, você pode criar um alarme para chamadas para a ação da AWS AppConfig `StartConfigurationSession` API que ultrapassam um limite. Para obter informações sobre como criar um alarme para uma métrica, consulte Criar um CloudWatch alarme com base em um limite estático no Guia do CloudWatch usuário da Amazon. Para obter informações sobre limites padrão para chamadas para o plano de AWS AppConfig dados, consulte Limites padrão do plano de dados no Referência geral da Amazon Web Services.

Monitorar implantações para reversão automática

Durante uma implantação, você pode mitigar situações em que dados de configuração malformados ou incorretos causam erros em seu aplicativo usando uma combinação de estratégias de AWS AppConfig implantação e reversões automáticas com base nos alarmes da Amazon. CloudWatch Depois de configurado, se um ou mais CloudWatch alarmes entrarem no `INSUFFICIENT_DATA` estado `ALARM` or durante uma implantação, AWS AppConfig reverterá automaticamente seus dados de configuração para a versão anterior, evitando interrupções ou erros no aplicativo.

> **Note**
>
> Uma implantação não é revertida automaticamente se as ações tiverem sido desativadas em um CloudWatch alarme associado.
> Você pode desativar e ativar os alarmes usando as ações DisableAlarmActionse EnableAlarmActionsda API ou os enable-alarm-actionscomandos disable-alarm-actionse no AWS CLI.

Você também pode reverter uma configuração chamando a operação da StopDeploymentAPI enquanto a implantação ainda está em andamento.

> **Important**
>
> Para implantações concluídas com êxito, AWS AppConfig também oferece suporte à reversão dos dados de configuração para uma versão anterior usando o `AllowRevert` parâmetro com a operação da StopDeploymentAPI. Para alguns clientes, a reversão para uma configuração anterior após uma implantação bem-sucedida garante que os dados sejam os mesmos de antes da implantação. A reversão também ignora os monitores de alarme, o que pode impedir o roll forward durante uma emergência na aplicação. Para obter mais informações, consulte Reverter uma configuração.

Para configurar reversões automáticas, você especifica o Amazon Resource Name (ARN) de uma ou mais CloudWatch métricas no campo de CloudWatch alarmes ao criar (ou editar) um ambiente. AWS AppConfig Para obter mais informações, consulte Criando ambientes para seu aplicativo em AWS AppConfig.

> **Note**
>
> Se você usa uma solução de monitoramento de terceiros (por exemplo, Datadog ou New Relic), você pode criar uma AWS AppConfig extensão que verifica os alarmes no ponto de `AT_DEPLOYMENT_TICK` ação e, como proteção de segurança, reverte a implantação se ela acionar um alarme. Para obter mais informações, consulte os seguintes exemplos de integração entre Datadog e New Relic em: GitHub
>
> - Datadog
> - New Relic
>
> Para obter mais informações sobre AWS AppConfig extensões, consulte os tópicos a seguir:
>
> - Estendendo AWS AppConfig fluxos de trabalho usando extensões
> - Passo a passo: Criação de extensões personalizadas AWS AppConfig

Métricas recomendadas para monitorar a reversão automática

As métricas que você escolher monitorar dependerão do hardware e do software usados por seus aplicativos. AWS AppConfig os clientes geralmente monitoram as seguintes métricas. Para obter uma lista completa das métricas recomendadas agrupadas por AWS service (Serviço da AWS), consulte Alarmes recomendados no Guia CloudWatch do usuário da Amazon.

Depois de determinar as métricas que você deseja monitorar, use CloudWatch para configurar alarmes. Para obter mais informações, consulte Usando CloudWatch alarmes da Amazon.

Serviço	Métrica	Detalhes
Amazon API Gateway	4 XXError	Esse alarme detecta uma alta taxa de erros do lado do cliente. Isso pode indicar um problema na autorização ou nos parâmetros da solicitação do cliente. Isso também pode significar que um recurso foi removido ou que um cliente

Serviço	Métrica	Detalhes
		está solicitando um recurso que não existe. Considere ativar o Amazon CloudWatch Logs e verificar se há erros que possam estar causando os erros 4XX. Além disso, considere ativar CloudWatch métricas detalhadas para visualizar essa métrica por recurso e método e restringir a origem dos erros. Os erros também podem ser causados por exceder o limite de controle de utilização configurado.
Amazon API Gateway	5XXError	Esse alarme detecta uma alta taxa de erros do lado do cliente. Isso pode indicar que há algo errado no backend da API, na rede ou na integração entre o gateway da API e a API de backend.

Serviço	Métrica	Detalhes
Amazon API Gateway	Latência	Esse alarme detecta alta latência em um estágio. Encontre o valor da métrica `IntegrationLatency` para verificar a latência do backend da API. Se as duas métricas estiverem alinhadas em sua maior parte, o backend da API será a fonte de maior latência e você deve investigar se há problemas . Considere também ativar CloudWatch os registros e verificar se há erros que possam estar causando a alta latência.
Amazon EC2 Auto Scaling	GroupInServiceCapacity	Esse alarme ajuda a detectar quando a capacidade do grupo está abaixo da capacidade desejada para a workload. Para soluciona r problemas, verifique se há falhas de inicialização em suas atividades de escalonam ento e confirme se a configura ção da capacidade desejada está correta.

Serviço	Métrica	Detalhes
Amazon EC2	CPUUtilization	Esse alarme ajuda a monitorar a utilização da CPU de uma instância do EC2. Dependend o da aplicação, níveis de utilização consistentemente altos podem ser normais. Porém, se a performance for prejudicada e a aplicação não for limitada por E/S de disco, memória ou recursos de rede, uma CPU no limite máximo poderá indicar um gargalo de recursos ou problemas de performance da aplicação.
Amazon ECS	CPUReservation	Esse alarme ajuda a detectar uma alta reserva de CPU no cluster do ECS. A alta reserva de CPU pode indicar que o cluster está ficando sem registros CPUs para a tarefa.
Amazon ECS	HTTPCode_Target_5xx_count	Esse alarme ajuda você a detectar uma alta contagem de erros do lado do servidor para o serviço ECS. Isso pode indicar que há erros que fazem com que o servidor não consiga atender às solicitaç ões.

Serviço	Métrica	Detalhes
Amazon EKS com o Container Insights	node_cpu_utilization	Esse alarme ajuda a detectar a alta utilização da CPU nos nós de processamento do cluster do Amazon EKS. Se a utilização for consisten temente alta, isso pode indicar a necessidade de substituir os nós de processamento por instâncias que tenham mais CPU ou a necessidade de escalar o sistema horizonta lmente.
Amazon EKS com o Container Insights	node_memory_utilization	Esse alarme ajuda a detectar a alta utilização de memória nos nós de processamento do cluster do Amazon EKS. Se a utilização for consisten temente alta, isso pode indicar a necessidade de escalar o número de réplicas de pod ou otimizar a sua aplicação.
Amazon EKS com o Container Insights	pod_cpu_utilization_over_po d_limit	Esse alarme ajuda a detectar a alta utilização da CPU em pods do cluster do Amazon EKS. Se a utilização for consistentemente alta, isso poderá indicar a necessidade de aumentar o limite da CPU para o pod afetado.

Serviço	Métrica	Detalhes
Amazon EKS com o Container Insights	pod_memory_utilization_over_pod_limit	Esse alarme ajuda a detectar a alta utilização da CPU em pods do cluster do Amazon EKS. Se a utilização for consistentemente alta, isso poderá indicar a necessidade de aumentar o limite da CPU para o pod afetado.
AWS Lambda	Erros	Esse alarme detecta altas contagens de erros. Os erros incluem as exceções lançadas pelo código, bem como as exceções lançadas pelo runtime do Lambda.
AWS Lambda	Controles de utilização	Esse alarme detecta um alto número de solicitações de invocação com controle de utilização. O controle de utilização ocorre quando não há simultaneidade disponível para aumentar a escala verticalmente.
Lambda Insights	memory_utilization	Esse alarme é usado para detectar se a utilização da memória de uma função do Lambda está se aproximando do limite configurado.

Serviço	Métrica	Detalhes
Amazon S3	4xxErrors	Esse alarme nos ajuda a informar o número total de códigos de status de erro 4xx que são gerados em resposta a solicitações de clientes. Os códigos de erro 403 podem indicar uma política do IAM incorreta e os códigos de erro 404 podem indicar uma aplicação cliente com comportamento incorreto, por exemplo.
Amazon S3	5xxErrors	Esse alarme ajuda a detectar um grande número de erros do lado do servidor. Esses erros indicam que um cliente fez uma solicitação que o servidor não conseguiu concluir. Isso pode ajudar você a correlacionar o problema que sua aplicação está enfrentando por causa do S3.

AWS AppConfig Histórico do documento do Guia do Usuário

A tabela a seguir descreve as mudanças importantes na documentação desde a última versão do AWS AppConfig.

Versão atual da API: 09/10/2019

Alteração	Descrição	Data
Novo exemplo de extensão AT_DEPLOYMENT_TICK	O ponto de ação `AT_DEPLOYMENT_TICK` oferece suporte à integração de monitoramento de terceiros. `AT_DEPLOYMENT_TICK` é invocado durante a orquestração do processamento da implantação da configuração. Se você usa uma solução de monitoramento terceirizada, como a New Relic, pode criar uma AWS AppConfig extensão que verifica os alarmes no ponto de `AT_DEPLOYMENT_TICK` ação e, como proteção de segurança, reverte a implantação se ela acionar um alarme. Para obter mais informações sobre AWS AppConfig extensões, consulte Estender AWS AppConfig fluxos de trabalho usando extensões. Para saber mais sobre as extensões personalizadas, consulte Passo a passo: criação de extensões	24 de fevereiro de 2026

	personalizadas do AWS AppConfig. Para ver uma amostra de código de uma AWS AppConfig extensão que usa o ponto de `AT_DEPLOY MENT_TICK` ação para se integrar à New Relic, consulte o exemplo da New Relic em. GitHub	
Nova versão da extensão AWS AppConfig Agent Lambda	O agente foi atualizado com pequenas melhorias e correções de bugs. Para ver os novos Amazon Resource Names (ARNs) para a extensão, consulte Versões disponíveis da extensão AWS AppConfig Agent Lambda.	20 de fevereiro de 2026
Obsoleto: AWS AppConfig extensão para Evidently CloudWatch	A AWS AppConfig extensão para CloudWatch Evidently não é mais suportada.	20 de fevereiro de 2026
IPv6 apoio	Tudo AWS AppConfig APIs agora com suporte IPv4 e IPv6 chamadas completos. Para obter mais informações, consulte Compreendendo o IPv6 suporte.	23 de abril de 2025

Novo tópico: Salvar uma versão anterior do sinalizador de atributos em uma nova versão	Quando você atualiza um sinalizador de recurso, salva AWS AppConfig automaticamente suas alterações em uma nova versão. Se quiser usar uma versão anterior do sinalizador de atributos, copie-a para uma versão de rascunho e depois salve-a. Não é possível editar e salvar as alterações em uma versão anterior do sinalizador sem salvá-las em uma nova versão. Para saber mais, consulte Salvar uma versão anterior do sinalizador de atributos em uma nova versão.	15 de abril de 2025

Novo tópico: Exemplos de sinalizadores de recursos para o modo de desenvolvimento local do AWS AppConfig agente	AWS AppConfig O agente suporta um modo de desenvolvimento local. Se você habilitar o modo de desenvolvimento local, o agente lerá os dados de configuração de um diretório especificado no disco. Ele não recupera dados de configuração de AWS AppConfig. Para ajudar você a entender melhor como usar o modo de desenvolvimento local, este guia agora contém um tópico com exemplos de sinalizadores de atributos. Para obter mais informações, consulte Exemplos de sinalizadores de recursos para o modo de desenvolvimento local do AWS AppConfig agente.	18 de fevereiro de 2025

Novo tópico: criar um perfil de configuração para fontes de dados não nativas	O tópico descreve o processo de alto nível para usar uma AWS AppConfig extensão para recuperar dados de configuração de fontes que não são suportadas nativamente, incluindo outros AWS serviços como Amazon RDS e Amazon DynamoDB, bem como fontes de terceiros, como,, ou um repositório local. GitHub GitLab Para saber mais, consulte Criar um perfil de configuração para fontes de dados que não são nativas	19 de dezembro de 2024
Tópico atualizado: correção do regex na referência de tipo de sinalizadores de atributos	O esquema json na referência do tipo de sinalizador de atributos mostrava anteriormente o seguinte padrão de regex em vários lugares: `"^[a-z][a-zA-Z\\d-_]{0,63}$"` . O padrão correto de regex é `"^[a-z][a-zA-Z\\d_-]{0,63}$"` . O hífen é listado após o sublinhado. Para obter mais informações, consulte Entendendo a referência de tipo para a AWS. AppConfig. FeatureFlags	18 de dezembro de 2024

Tópicos atualizados: adição de exemplos de variáveis de ambiente	As tabelas que descrevem as variáveis de ambiente nos tópicos a seguir foram atualizadas para incluir exemplos: • (Opcional) Usando variáveis de ambiente para configurar o AWS AppConfig Agente para Amazon ECS e Amazon EKS • (Opcional) Usando variáveis de ambiente para configurar o AWS AppConfig agente para o Amazon EC2 • Configurando a extensão do AWS AppConfig Agent Lambda	12 de dezembro de 2024
Nova seção: Noções básicas sobre o operador split	Uma nova seção usa exemplos para explicar como o operador `split` funciona para uma regra de sinalizador de atributos multivariante. Para saber mais, consulte Noções básicas sobre as regras de sinalizadores de atributos multivariante.	22 de novembro de 2024

Novo ponto de ação da extensão: AT_DEPLOYMENT_TICK	AWS AppConfig lançou um novo ponto de ação para usuários que criam extensões personalizadas. O ponto de ação `AT_DEPLOYMENT_TICK` oferece suporte à integração de monitoramento de terceiros. `AT_DEPLOYMENT_TICK` é invocado durante a orquestração do processamento da implantação da configuração. Se você usa uma solução de monitoramento de terceiros (por exemplo, o Datadog), pode criar uma AWS AppConfig extensão que verifica os alarmes no ponto de `AT_DEPLOYMENT_TICK` ação e, como proteção de segurança, reverte a implantação se ela acionar um alarme. Para obter mais informações sobre AWS AppConfig extensões, consulte Estender AWS AppConfig fluxos de trabalho usando extensões. Para saber mais sobre as extensões personalizadas, consulte Passo a passo: criação de extensões personalizadas do AWS AppConfig. Para ver uma amostra de código de uma AWS AppConfig extensão que usa o ponto de `AT_DEPLOY`	22 de novembro de 2024

	MENT_TICK ação para se integrar ao Datadog, consulte aws-samples/-for-datadog on. aws-appconfig-tick-extn GitHub	
Novo tópico: considera ções sobre o uso de AWS AppConfig dispositivos móveis	Um novo tópico neste guia descreve considerações importantes sobre o uso de sinalizadores de AWS AppConfig recursos em dispositivos móveis. Para obter mais informações, consulte Considerações sobre o uso de dispositivos móveis no AWS AppConfig.	21 de novembro de 2024
Novo recurso: Proteção AWS AppConfig contra exclusão	AWS AppConfig agora fornece uma configuração de conta para ajudar a evitar que os usuários excluam acidentalmente ambientes e perfis de configuração usados ativamente. Para obter mais informações, consulte Configurando a AWS AppConfig proteção contra exclusão.	28 de agosto de 2024
Nova versão da extensão AWS AppConfig Agent Lambda	O agente foi atualizado com pequenas melhorias e correções de bugs. Para ver os novos Amazon Resource Names (ARNs) para a extensão, consulte Versões disponíveis da extensão AWS AppConfig Agent Lambda.	9 de agosto de 2024

Novos exemplos de código para recuperar variantes de sinalizador	Para obter mais informaçõ es, consulte Usando o AWS AppConfig agente para recuperar um sinalizador de recurso com variantes.	9 de agosto de 2024
Nova versão da extensão AWS AppConfig Agent Lambda	O agente foi atualizado para comportar destinos, variantes e divisões de sinalizadores de atributos. Para ver os novos Amazon Resource Names (ARNs) para a extensão, consulte Versões disponíveis da extensão AWS AppConfig Agent Lambda.	23 de julho de 2024

Novo atributo: sinalizadores de atributos multivariante	Os sinalizadores de atributos multivariante permitem definir um conjunto de possíveis valores de sinalizador a serem exibidos para uma solicitação. Também é possível configurar diferentes status (habilitado ou desabilitado) para sinalizadores multivariante. Ao solicitar um sinalizador configurado com variantes, sua aplicação fornece um contexto que o AWS AppConfig avalia em relação a um conjunto de regras definidas pelo usuário. Dependendo do contexto especificado na solicitação e das regras definidas para a variante, AWS AppConfig retorna valores de sinalizadores diferentes para o aplicativo. Para obter mais informações, consulte Criar sinalizadores de atributos multivariante.	23 de julho de 2024
Nova versão da extensão AWS AppConfig Agent Lambda	O agente foi atualizado com pequenas melhorias e correções de bugs. Para ver os novos Amazon Resource Names (ARNs) para a extensão, consulte Versões disponíveis da extensão AWS AppConfig Agent Lambda.	28 de fevereiro de 2024

AWS AppConfig amostras de extensões personalizadas	O tópico Passo a passo: Criação de AWS AppConfig extensões personalizadas agora inclui links para os seguintes exemplos de extensões em: GitHub • Extensão de exemplo que impede implantações com um calendário de moratória `blocked day` usando o calendário de alterações do Systems Manager • Extensão de exemplo que impede o vazamento de segredos em dados de configuração usando a ferramenta git-secrets • Extensão de exemplo que impede o vazamento de informações de identific ação pessoal (PII) em dados de configuração usando o Amazon Comprehend	28 de fevereiro de 2024

Novo tópico: Registro de chamadas de AWS AppConfig API usando AWS CloudTrail	AWS AppConfig é integrado com AWS CloudTrail, um serviço que fornece um registro das ações realizadas por um usuário, função ou AWS serviço em AWS AppConfig. CloudTrail captura todas as chamadas de API AWS AppConfig como eventos. Esse novo tópico fornece conteúdo AWS AppConfig específico em vez de um link para o conteúdo correspondente no Guia do AWS Systems Manager usuário. Para obter mais informações, consulte Logging AWS AppConfig API chamando usando AWS CloudTrail.	18 de janeiro de 2024

AWS AppConfig agora suporta AWS PrivateLink	Você pode usar AWS PrivateLink para criar uma conexão privada entre sua VPC e. AWS AppConfig Você pode acessar AWS AppConfig como se estivesse em sua VPC, sem o uso de um gateway de internet, dispositivo NAT, conexão VPN ou conexão. Direct Connect As instâncias na sua VPC não precisam de endereços IP públicos para acessar o AWS AppConfig. Para obter mais informações, consulte Acesso AWS AppConfig usando um endpoint de interface (AWS PrivateLink).	6 de dezembro de 2023

Recursos adicionais de recuperação de AWS AppConfig agentes e um novo modo de desenvolvimento local	AWS AppConfig O agente oferece os seguintes recursos adicionais para ajudá-lo a recuperar configurações para seus aplicativos. Recursos de recuperação adicionais • Recuperação de várias contas: use o AWS AppConfig Agente de uma conta primária ou de recuperação Conta da AWS para recuperar dados de configuração de várias contas de fornecedores. • Gravar cópia da configura ção no disco: use o AWS AppConfig Agente para gravar dados de configura ção no disco. Esse recurso permite que os clientes com aplicações que leem dados de configuração do disco se integrem ao AWS AppConfig. **Note** A configuração de gravação em disco não foi projetada como um recurso de backup de configuração. AWS AppConfig O agente	1.º de dezembro de 2023

não lê os arquivos de configuração copiados para o disco. Se você quiser fazer backup das configurações em disco, consulte as variáveis de `PRELOAD_BACKUP` ambiente `BACKUP_DIRECTORY` e de uso do agente com o Amazon EC2 ou AWS AppConfig Uso do AWS AppConfig agente com o Amazon ECS e o Amazon EKS.

Modo de desenvolvimento local

AWS AppConfig O agente suporta um modo de desenvolvimento local. Se você habilitar o modo de desenvolvimento local, o agente lerá os dados de configuração de um diretório especificado no disco. Ele não recupera dados de configuração de AWS AppConfig. É possível simular implantações de configuração atualizando arquivos no diretório especificado. Recomendamos o modo de desenvolvimento local para os seguintes casos de uso:

	• Testar diferentes versões de configuração antes de implantá-las usando o AWS AppConfig. • Testar diferentes opções de configuração para um novo atributo antes de confirmar as alterações no seu repositório de código. • Testar diferentes cenários de configuração para verificar se funcionam conforme o esperado.	
Novo tópico de exemplos de código.	Foi adicionado um novo tópico de exemplos de código a este guia. O tópico inclui exemplos em Java, Python e JavaScript para executar programaticamente seis ações comuns. AWS AppConfig	17 de novembro de 2023
Índice revisado para refletir AWS AppConfig melhor o fluxo de trabalho	O conteúdo deste guia do usuário está agora agrupado sob os títulos Criação, Implantação, Recuperação e Extensão de fluxos de trabalho. Essa organização reflete melhor o fluxo de trabalho de uso do AWS AppConfig e tem como objetivo ajudar a tornar o conteúdo mais fácil de ser descoberto.	7 de novembro de 2023

Adição da referência de carga útil	O tópico Criação de uma função do Lambda para uma extensão personalizada do AWS AppConfig agora inclui uma referência de carga útil de solicitação e resposta.	7 de novembro de 2023
Nova estratégia de implantação AWS predefinida	AWS AppConfig agora oferece e recomenda a estratégia de implantação `AppConfig.Linear20PercentEvery6Minutes` predefinida. Para obter mais informações, consulte estratégias de implantação predefinidas.	11 de agosto de 2023
AWS AppConfig integração com o Amazon EC2	Você pode se integrar AWS AppConfig com aplicativos executados em suas instâncias Linux do Amazon Elastic Compute Cloud (Amazon EC2) usando o Agent. AWS AppConfig O agente oferece suporte às arquiteturas x86_64 e ARM64 para o Amazon EC2. Para obter mais informações, consulte Integração do AWS AppConfig com o Amazon EC2.	20 de julho de 2023

CloudFormation suporte para novos AWS AppConfig recursos e um exemplo de sinalizador de recursos	AWS CloudFormation agora oferece suporte ao AWS::AppConfig::Extensione aos AWS::AppConfig::ExtensionAssociationrecursos para ajudar você a começar a usar AWS AppConfig extensões. Os recursos AWS::AppConfig::ConfigurationProfilee AWS::AppConfig::HostedConfigurationVersion agora incluem um exemplo para criar um perfil de configuração de sinalizador de recurso no armazenamento de configuração AWS AppConfig hospedado.	12 de abril de 2023

AWS AppConfig integração com AWS Secrets Manager	AWS AppConfig se integra com AWS Secrets Manager. O Secrets Manager ajuda você a criptografar, armazenar e recuperar credenciais com segurança para bancos de dados e outros serviços. Em vez de codificar credencia is em seus aplicativos, você pode fazer chamadas ao Secrets Manager para recuperar suas credencia is sempre que necessário. O Secrets Manager ajuda você a proteger o acesso aos seus recursos e dados de TI, permitindo que você alterne e gerencie o acesso aos seus segredos. Ao criar um perfil de configura ção de formato livre, é possível escolher o Secrets Manager como fonte de seus dados de configuração. Você deve se integrar ao Secrets Manager e criar um segredo antes de criar o perfil de configuração. Para obter mais informações sobre o Secrets Manager, consulte O que é AWS Secrets Manager? no Guia do AWS Secrets Manager usuário. Para obter informações sobre como criar um perfil de configuração,	2 de fevereiro de 2023

consulte Criação de um perfil de configuração de formato livre.

AWS AppConfig integração com o Amazon ECS e o Amazon EKS	Você pode se integrar AWS AppConfig ao Amazon Elastic Container Service (Amazon ECS) e ao Amazon Elastic Kubernetes Service (Amazon EKS) usando o agente. AWS AppConfig O agente funciona como um contêiner auxiliar executado junto com seus aplicativos de contêiner Amazon ECS e Amazon EKS. O agente aprimora o processamento e o gerenciamento de aplicativos em contêineres das seguintes maneiras: • O agente liga AWS AppConfig em seu nome usando uma função AWS Identity and Access Management (IAM) e gerenciando um cache local de dados de configuração. Ao extrair dados de configuração do cache local, seu aplicativo exige menos atualizações de código para gerenciar dados de configuração, recupera dados de configuração em milissegundos e não é afetado por problemas de rede que podem afetar as chamadas para esses dados.	2 de dezembro de 2022

- O agente oferece uma experiência nativa para recuperar e resolver sinalizadores de AWS AppConfig recursos.
- Pronto para uso, o agente fornece as práticas recomendadas para estratégias de armazenamento em cache, intervalos de pesquisa e disponibilidade de dados de configuração local, enquanto rastreia os tokens de configuração necessários para chamadas de serviço subsequentes.
- Durante a execução em segundo plano, o agente consulta periodicamente o plano de AWS AppConfig dados para atualizações de dados de configuração. Seu aplicativo em contêiner pode recuperar os dados conectando-se ao localhost na porta 2772 (um valor de porta padrão personalizável) e chamando HTTP GET para recuperar os dados.
- O AWS AppConfig agente atualiza os dados de configuração em seus contêineres sem precisar reiniciá-los ou reciclá-los.

Para obter mais informações, consulte Integração do AWS AppConfig com o Amazon ECS e o Amazon EKS.

Nova extensão: AWS AppConfig extensão para CloudWatch Evidently	Você pode usar o Amazon CloudWatch Evidently para validar com segurança novos recursos, disponibilizando-os a uma porcentagem específica de seus usuários enquanto você implanta o recurso. Você pode monitorar a performance do novo atributo para auxiliar na decisão de quando aumentar o tráfego para seus usuários. Isso ajuda você a reduzir riscos e identificar consequências não intencionais antes de iniciar totalmente o recurso. Você também pode realizar A/B experimentos para tomar decisões de design de recursos com base em evidências e dados. A AWS AppConfig extensão do CloudWatch Evidently permite que seu aplicativo atribua variações às sessões do usuário localmente, em vez de chamar a EvaluateFeatureoperação. Uma sessão local reduz os riscos de latência e disponibilidade associados a uma chamada de API. Para obter informações sobre como configurar e usar a extensão, consulte Executar lançamentos e A/B experimentos com o	13 de setembro de 2022

	CloudWatch Evidently no Guia do CloudWatch usuário da Amazon.	
Desaprovação da ação `GetConfiguration` da API	Em 18 de novembro de 2021, AWS AppConfig lançou um novo serviço de plano de dados. Esse serviço substitui o processo anterior de recuperação de dados de configuração usando a ação `GetConfiguration` da API. O serviço de plano de dados usa duas novas ações de API StartConfigurationSessionGetLatestConfiguratione. O serviço de plano de dados também usa novos endpoints. Para obter mais informações, consulte Sobre o serviço AWS AppConfig de plano de dados.	13 de setembro de 2022
Nova versão da extensão AWS AppConfig Agent Lambda	A versão 2.0.122 da extensão Agent AWS AppConfig Lambda já está disponível. A nova extensão usa nomes de recursos da Amazon diferentes (ARNs). Para obter mais informações, consulte Notas de versão da extensão do Lambda do AWS AppConfig Agent.	23 de agosto de 2022

Lançamento de AWS AppConfig extensões	Uma extensão aumenta sua capacidade de injetar lógica ou comportamento em diferentes pontos durante o AWS AppConfig fluxo de trabalho de criação ou implantação de uma configuração. Você pode usar extensões AWS criadas por - autoria ou criar suas próprias. Para obter mais informações, consulte Trabalhando com AWS AppConfig extensões.	12 de julho de 2022
Nova versão da extensão AWS AppConfig Agent Lambda	A versão 2.0.58 da extensão Agent AWS AppConfig Lambda já está disponível. A nova extensão usa nomes de recursos da Amazon diferente s (ARNs). Para obter mais informações, consulte Versões disponíveis da extensão AWS AppConfig Lambda.	3 de maio de 2022

AWS AppConfig integração com o Atlassian Jira	A integração com o Atlassian Jira permite AWS AppConfig criar e atualizar problemas no console Atlassian sempre que você fizer alterações em um sinalizador de recurso em seu para o especificado. Conta da AWS Região da AWS Cada problema do Jira inclui o nome do sinalizador, o ID do aplicativo, o ID do perfil de configuração e os valores do sinalizador. Depois de atualizar, salvar e implantar as alterações do seu sinalizador, o Jira atualiza os problemas existentes com os detalhes da alteração. Para obter mais informações, consulte Integração do AWS AppConfig com o Atlassian Jira.	7 de abril de 2022

Disponibilidade geral de sinalizadores de recursos e suporte à extensão Lambda ARM64 para processadores (Graviton2)	Com sinalizadores de AWS AppConfig recursos, você pode desenvolver um novo recurso e implantá-lo na produção enquanto oculta o recurso dos usuários. Você começa adicionando o sinalizador AWS AppConfig como dados de configura ção. Quando o atributo estiver pronto para ser lançado, você poderá atualizar os dados de configuração do sinalizador sem implantar nenhum código. Esse atributo melhora a segurança do seu ambiente de desenvolvimento e operações porque você não precisa implantar nenhum novo código para liberar o atributo. Para obter mais informações, consulte Criar um perfil de configuração de sinalizadores de atributos. A disponibilidade geral dos sinalizadores de atributos no AWS AppConfig inclui os seguintes aprimoramentos: • O console inclui uma opção para designar um sinalizad or como um sinalizador de curto prazo. Você pode filtrar e classificar a lista de sinalizadores em sinalizad ores de curto prazo.	15 de março de 2022

	• Para clientes que usam sinalizadores de atributos no AWS Lambda, a nova extensão do Lambda permite que você chame sinalizadores de atributos individuais usando um endpoint HTTP. Para obter mais informações, consulte Recuperação de um ou mais sinalizadores a partir de uma configuração de sinalizadores de atributos. Essa atualização também fornece suporte para AWS Lambda extensões desenvolvidas para processadores ARM64 (Graviton2). Para obter mais informações, consulte Versões disponíveis da extensão AWS AppConfig Lambda.	
A ação GetConfiguration da API está obsoleta	A ação `GetConfiguration` da API está obsoleta. As chamadas para receber dados de configuração devem usar o `StartConfigurationSession` e `GetLatestConfiguration` APIs em vez disso. Para obter mais informações sobre eles APIs e como usá-los, consulte Recuperando a configuração.	28 de janeiro de 2022

Novo ARN de região para extensão AWS AppConfig Lambda	AWS AppConfig A extensão Lambda está disponível na nova região Ásia-Pacífico (Osaka). O nome do recurso da Amazon (ARN) é necessário para criar um Lambda na região. Para obter mais informações sobre o ARN da região Ásia-Pacífico (Osaka), consulte Adicionar a extensão AWS AppConfig Lambda.	4 de março de 2021
AWS AppConfig Extensão Lambda	Se você usa AWS AppConfig para gerenciar configurações para uma função Lambda, recomendamos que você adicione a extensão AWS AppConfig Lambda. Essa extensão inclui as melhores práticas que AWS AppConfig simplificam o uso e reduzem os custos. Custos reduzidos resultam de menos chamadas de API para o AWS AppConfig serviço e, separadamente, custos reduzidos de tempos mais curtos de processamento da função Lambda. Para obter mais informações, consulte Integração do AWS AppConfig com extensões do Lambda.	8 de outubro de 2020

Nova seção	Foi adicionada uma nova seção que fornece instruções para configuração do AWS AppConfig. Para obter mais informações, consulte Configurar AWS AppConfig.	30 de setembro de 2020
Adição de procedimentos de linha de comando	Os procedimentos neste guia do usuário agora incluem etapas de linha de comando para o AWS Command Line Interface (AWS CLI) e o Tools for Windows PowerShell. Para obter mais informações, consulte Trabalhando com AWS AppConfig.	30 de setembro de 2020
Lançamento do guia AWS AppConfig do usuário	Use AWS AppConfig uma ferramenta para criar AWS Systems Manager, gerenciar e implantar rapidamente configurações de aplicativos. AWS AppConfig oferece suporte a implantações controladas em aplicativos de qualquer tamanho e inclui verificações e monitoramento de validação integrados. Você pode usar AWS AppConfig com aplicativos hospedados em instâncias do EC2 AWS Lambda, contêineres, aplicativos móveis ou dispositivos de IoT.	31 de julho de 2020

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.