WASHINGTON LUIZ PERONI

MACHINE LEARNING ENGINEERING

BIG DATA CLOUD PLATFORMS

AULA 04

SUMÁRIO

O QUE VEM POR AÍ?

Olá, pessoal! Boas-vindas à aula de AWS Lambda, um serviço de código como serviço (runtime) orientado a eventos que ajuda na arquitetura de sistemas desacoplados, altamente disponíveis e escaláveis.

O Amazon Lambda é um serviço de runtime de código totalmente gerenciado pela AWS que permite comunicação e execução orientados a eventos entre diferentes serviços da AWS e partes de sistemas distribuídos. A Lambda permite o desacoplamento de sistemas e por ser totalmente gerenciado e sem servidor, facilita a implantação e manutenção. Suportando uma série de linguagens de programação, o AWS Lambda é um serviço que ajuda a empresa no desenvolvimento e na escalabilidade de suas aplicações.

Vamos demostrar alguns casos de uso em conjunto com outros serviços, como Glue, Athena e S3, para que vocês possam aplicar seus conhecimentos no mundo real.

HANDS ON

Demonstraremos na prática, dentro do console da AWS, o uso do AWS Lambda, mostrando desde a tela inicial e os conceitos básicos até um caso de uso real de integração com Glue, Athena e S3. Demonstraremos todo poder da programação python orientada a eventos para limitar escopo entre aplicações e, assim, trazer robustez e escala para a arquitetura desacoplada em Big Data. Esse conteúdo será divido nas seguintes etapas:

- AWS Lambda via console: configuração inicial.
- AWS Lambda: linguagens disponíveis, arquitetura básica.
- AWS Lambda: código python sendo disparado por um evento do S3.
- Encadeamento de serviço: S3 disparando Lambda que escreve numa fila SQS e que, por sua vez, pode disparar outra Lambda.

SAIBA MAIS

Computação Serverless

Computação serverless, apesar do nome, não significa a ausência de servidores. Na verdade, refere-se a um modelo de computação em nuvem que abstrai o gerenciamento de servidores do desenvolvedor. Isso permite que os desenvolvedores se concentrem na lógica de seus aplicativos, sem se preocupar com a infraestrutura subjacente.

Funcionamento

Na computação serverless, o provedor de nuvem gerencia a infraestrutura de servidores e provisiona recursos automaticamente em resposta a eventos ou solicitações. O desenvolvedor fornece apenas o código que precisa ser executado e o provedor de nuvem se encarrega do resto.

Exemplos de serviços serverless:

- **Funções como serviço (FaaS):** permitem que os desenvolvedores executem código em resposta a eventos específicos, como um upload de arquivo ou uma chamada API.
- **Backend como serviço (BaaS):** fornece funcionalidades prontas para uso, como autenticação, armazenamento e bancos de dados.
- **Eventos como serviço (EaaS):** permitem que os desenvolvedores criem e gerenciem eventos em nuvem.

Vantagens da computação serverless:

- **Agilidade e escalabilidade:** os recursos são provisionados automaticamente, o que permite que os aplicativos escalem rapidamente para cima ou para baixo de acordo com a demanda.
- **Custo-benefício:** os desenvolvedores pagam apenas pelos recursos que usam, o que pode reduzir significativamente os custos de infraestrutura.
- **Foco no desenvolvimento:** o desenvolvedor pode se concentrar na lógica do aplicativo sem se preocupar com a infraestrutura.

Desvantagens da computação serverless:

- **Falta de controle:** o desenvolvedor tem menos controle sobre a infraestrutura subjacente.
- **Desafios de depuração:** pode ser difícil depurar problemas em um ambiente serverless.

AWS Lambda: da concepção à adoção ampla

2014: anúncio na AWS re:Invent: em novembro, a AWS anuncia o Lambda em beta privado, prometendo uma nova maneira de executar código sem provisionar ou gerenciar servidores.

Visão: o Lambda foi idealizado para eliminar a necessidade de gerenciar infraestrutura, permitindo que os desenvolvedores se concentrassem na lógica de seus aplicativos.

Funcionalidades: suporte inicial para Node.js, Python e Java como linguagens de runtime, com invocação por meio de eventos de serviços AWS, como S3 e CloudWatch.

Lançamento oficial (2015) : em março, o Lambda é lançado para o público em geral, expandindo o suporte para Ruby e Go como linguagens de runtime.

Crescimento da comunidade: a comunidade de desenvolvedores do Lambda cresce rapidamente, impulsionada pela simplicidade e flexibilidade da plataforma.

Casos de uso: adoção em diversos setores para processamento de eventos, automação de tarefas, integração de sistemas e backends móveis.

Novas funcionalidades (2016): introdução de Lambda Layers para empacotar bibliotecas e dependências e suporte para Amazon API Gateway como fonte de eventos.

Expansão do alcance: integração com outros serviços AWS, como DynamoDB, Kinesis e SNS, ampliando as possibilidades de aplicações serverless.

Exemplos de sucesso: Netflix, Coca-Cola e Airbnb relatam casos de sucesso com o Lambda, demonstrando escalabilidade e economia de custos.

Anúncios importantes (2017): lançamento do Lambda@Edge, para execução de código em edge locations da AWS, e do AWS Lambda Runtime Interface (CRI), para abstrair a lógica de runtime.

Evolução da plataforma: suporte para mais linguagens de runtime, incluindo C# e PowerShell, e aprimoramentos na experiência de desenvolvedor com o console e APIs.

Adoção em larga escala: empresas de todos os portes e setores reconhecem o valor do Lambda para agilidade, escalabilidade e economia.

2018-presente - crescimento contínuo: o Lambda se torna um dos serviços AWS mais populares, com milhares de novos usuários e casos de uso a cada mês.

Inovações frequentes: adição de funcionalidades, como Lambda Destinations para roteamento de eventos, Lambda Extensions para customização de runtime e AWS Lambda Powertools para bibliotecas de utilitários.

Maturidade e confiabilidade: o Lambda se consolida como uma plataforma robusta e confiável para aplicações serverless de missão crítica.

Exemplos de arquitetura prática

Processamento de arquivos: função Lambda acionada por eventos de upload no S3 para processar, transformar e analisar arquivos.

Automação de tarefas: função Lambda agendada pelo CloudWatch para realizar tarefas repetitivas em intervalos regulares.

Integração de sistemas: função Lambda como intermediário para comunicação entre diferentes sistemas e APIs.

Backend móvel: função Lambda como backend para APIs móveis, oferecendo escalabilidade e flexibilidade.

Vantagens e desvantagens do AWS Lambda

Vantagens:

1. Escalabilidade automática:

- **Descrição:** o Lambda escala automaticamente a quantidade de instâncias em execução com base na demanda, sem necessidade de intervenção manual.
- **Exemplo prático:** um aplicativo web que recebe um pico de tráfego repentino. O Lambda aumenta automaticamente a capacidade para lidar com o aumento de solicitações sem que o desenvolvedor precise se preocupar com provisionamento de servidores.

2. Agilidade no desenvolvimento:

- **Descrição:** o Lambda permite que os desenvolvedores se concentrem na lógica do código, sem se preocupar com a infraestrutura.
- **Exemplo prático:** criar um microsserviço que processa pagamentos em uma aplicação e-commerce. Com o Lambda, o desenvolvedor pode escrever o código em sua linguagem preferida e integrá-lo com o sistema de pagamento sem precisar gerenciar servidores.

3. Economia de custos:

- **Descrição:** o Lambda cobra apenas pelo tempo de execução e recursos utilizados, o que significa que você paga apenas pelo que usa.
- **Exemplo prático:** uma aplicação que processa arquivos em lote. O Lambda pode ser configurado para executar apenas quando novos arquivos forem enviados, reduzindo significativamente os custos de infraestrutura.

4. Facilidade de uso:

- **Descrição:** o Lambda oferece uma interface simples e intuitiva para gerenciar funções e eventos.
- **Exemplo prático:** criar uma função Lambda que envia um e-mail de confirmação para usuário que se cadastra em um site. O Lambda pode ser configurado para ser acionado por um evento específico, como a criação de um novo usuário no banco de dados.

5. Ampla variedade de gatilhos:

- **Descrição:** o Lambda pode ser acionado por diversos eventos, como alterações em arquivos no S3, mensagens em filas do SQS, eventos de CloudTrail e muito mais.
- **Exemplo prático:** criar uma função Lambda que redimensiona automaticamente imagens quando elas são carregadas no S3. O Lambda pode ser configurado para ser acionado por um evento de "objeto criado" no S3.

Desvantagens:

1. Tempo de execução limitado:

- **Descrição:** o tempo máximo de execução de uma função Lambda é de 15 minutos.
- **Exemplo prático:** processamento de um grande arquivo de vídeo. Se o processamento levar mais do que 15 minutos, a função Lambda será terminada.
- **Solução:** em tarefas mais longas, é possível usar o AWS Step Functions para orquestrar várias funções Lambda.

2. Limitações de memória:

- **Descrição:** a quantidade máxima de memória disponível para uma função Lambda é de 3 GB.
- **Exemplo prático:** processamento de um conjunto de dados muito grande. Se o conjunto de dados for maior que 3 GB, a função Lambda não poderá processá-lo.
- **Solução:** para conjuntos de dados maiores, é possível usar o Amazon EMR ou o Amazon Redshift.

3. Dificuldade de depuração:

- **Descrição:** o Lambda pode ser difícil de depurar, pois não oferece as mesmas ferramentas de depuração que um ambiente de desenvolvimento tradicional.
- **Exemplo prático:** uma função Lambda que apresenta um erro inesperado. Pode ser difícil identificar a causa do erro sem as ferramentas de depuração adequadas.
- **Solução:** é possível usar o AWS X-Ray para rastrear o desempenho e identificar erros em funções Lambda.

4. Segurança:

- **Descrição:** as funções Lambda são executadas em um ambiente isolado, mas é importante configurar as permissões de acesso corretamente para evitar problemas de segurança.
- **Exemplo prático:** uma função Lambda que tem acesso a dados confidenciais. Se as permissões de acesso não forem configuradas corretamente, os dados podem ser acessados por usuários não autorizados.
- **Solução:** é importante seguir as melhores práticas de segurança ao configurar funções Lambda, como usar o IAM para controlar o acesso aos dados.

O QUE VOCÊ VIU NESTA AULA?

Nessa aula vimos como o AWS Lambda nos ajuda no desacoplamento de sistemas, dividindo escopo e responsabilidade de aplicações para executar tarefas de forma assíncrona e tornando os sistemas mais robustos e escaláveis. Embora isso aumente a complexidade inicial, é importante discutir e rever a arquitetura, para que ela sempre esteja em evolução.

Lambda nos permite pensar só em código, uma vez que é serverless, ou seja, não precisamos nos preocupar em gerenciar e dimensionar instancias EC2. Revejam a aula, pratiquem em casa e, em caso de dúvidas, entrem em contato conosco. Bons estudos!

REFERÊNCIAS

FULLER, M. **AWS Lambda: A Guide to Serverless Microservices**. [S.l.: s.n.], 2017.

POCCIA, D. **AWS Lambda in Action**. Shelter Island: Manning Publications, 2016.

SBARSKI, P. **Serverless Architectures on AWS: With AWS Lambda**. Shelter Island: Manning Publications, 2017.

PALAVRAS-CHAVE

Palavras-chave: Big Data. Microsserviços. Desacoplamento de Software. Engenharia de Software. AWS Lambda. Programação. Eventos. Python. NodeJs.