Otimização da economia empresarial com arquiteturas sem servidor

Setembro de 2017

Avisos

Este documento é disponibilizado apenas para fins informativos. Ele relaciona as atuais ofertas de produtos e práticas da AWS a contar da data de emissão deste documento, que estão sujeitas a alterações sem aviso prévio. Os clientes são responsáveis por fazer sua própria avaliação independente das informações neste documento e de qualquer uso dos produtos ou serviços da AWS, cada um dos quais é fornecido “como está”, sem garantia de qualquer tipo, expressa ou implícita. Este documento não cria quaisquer garantias, representações, compromissos contratuais, condições ou seguros da AWS, suas afiliadas, fornecedores ou licenciadores. As responsabilidades e as obrigações da AWS com os seus clientes são controladas por contratos dela, e este documento não é parte, nem modifica, qualquer contrato entre a AWS e seus clientes.

Sumário

Resumo

O objetivo deste whitepaper é ajudar diretores de informação (CIOs), diretores de tecnologia (CTOs) e arquitetos seniores a obter informações sobre as arquiteturas sem servidor e a conhecer seu impacto no tempo de entrada no mercado, na agilidade da equipe e na economia da TI. Ao eliminar servidores ociosos e subutilizados no design e simplificar drasticamente os designs de software baseados em nuvem, as abordagens sem servidor estão mudando rapidamente o cenário da TI.

Este whitepaper apresenta os princípios básicos das abordagens sem servidor e o portfólio sem servidor da AWS, além de incluir uma série de estudos de caso que ilustram como as empresas existentes já desfrutam de agilidade e de benefícios econômicos significativos com a adoção das abordagens sem servidor. Este documento ilustra como organizações de todos os portes podem usar arquiteturas sem servidor para arquitetar sistemas reativos e baseados em eventos, além de fornecer rapidamente microsserviços nativos da nuvem por uma fração dos custos convencionais.

Introdução

Muitas empresas já desfrutam dos benefícios da execução de aplicativos na nuvem pública, incluindo a redução de custos com o faturamento pago conforme o uso e a maior agilidade com o uso de recursos de TI sob demanda. Vários estudos em tipos de aplicativos e setores diferentes demonstraram que a migração das arquiteturas de aplicativos existentes para a nuvem reduz o custo total de propriedade (TCO) e o tempo de entrada no mercado.[1]

Com relação às soluções em nuvem local e privada, a criação, a implantação e o gerenciamento de frotas de servidores e dos aplicativos executados neles são significativamente mais simples na nuvem pública. No entanto, as empresas hoje têm opções adicionais que vão além do servidor clássico ou das arquiteturas baseadas em VM para aproveitar a nuvem pública. Embora, para as empresas, a nuvem elimine a necessidade de compra e de manutenção de hardware próprio, *qualquer* arquitetura baseada em servidor ainda exige que a escalabilidade e a confiabilidade sejam arquitetadas. As empresas também precisam assumir os desafios de aplicar patches e de implantá-los nas frotas de servidores conforme os aplicativos evoluem. Além disso, elas precisam escalar as frotas de servidores considerando as cargas de pico e tentar reduzi-las quando e onde possível para diminuir os custos – tudo isso protegendo a experiência dos usuários finais e a integridade dos sistemas internos. Servidores ociosos e subutilizados já provaram ser caros e um desperdício. Os analistas estimam que, na prática, a capacidade de até 85% dos servidores é subutilizada.[2]

Os serviços de computação sem servidor, como o AWS Lambda, são desenvolvidos para enfrentar esses desafios oferecendo às empresas uma maneira diferente de abordar o design de aplicativos, com redução inerente nos custos e no tempo de entrada no mercado. *O AWS Lambda elimina a complexidade de lidar com servidores em todos os níveis da pilha de tecnologia, além de apresentar um modelo de faturamento de pagamento mediante solicitação, em que são eliminados os custos da capacidade computacional ociosa.* Além disso, as funções Lambda permitem que as organizações adotem arquiteturas de microsserviços. Eliminar a infraestrutura e migrar para o modelo Lambda oferece duas vantagens econômicas:

- Problemas como servidores ociosos simplesmente deixam de existir, junto com suas consequências econômicas. Um serviço de computação sem servidor como o AWS Lambda nunca fica “inativo”, uma vez que as cobranças apenas são acumuladas quando algum trabalho útil é executado, com granularidade de faturamento em nível de milissegundos.

- O gerenciamento de frotas (incluindo aplicação de patches de segurança, implantações e monitoramento de servidores) não é mais necessário. Isso significa que não é necessário manter as ferramentas associadas, os processos e os plantões exigidos para dar suporte ao tempo de atividade da frota de servidores 24 horas por dia, 7 dias por semana. Usar o Lambda para criar microsserviços ajuda as organizações a serem mais ágeis. Sem a sobrecarga do gerenciamento de servidores, as empresas podem direcionar seus recursos de TI escassos para o que é importante: os negócios.

Com custos de infraestrutura reduzidos significativamente, equipes mais ágeis e focadas e entrada no mercado mais rápida, as empresas que já adotaram as abordagens sem servidor estão obtendo uma vantagem importante em relação à concorrência.

Noções básicas sobre os aplicativos sem servidor

A vantagem da abordagem sem servidor mencionada acima é atraente, mas quais são as considerações para a implementação prática? O que diferencia um aplicativo sem servidor dos aplicativos convencionais baseados em servidor?

Os aplicativos sem servidor são arquitetados de modo que os desenvolvedores podem se concentrar em sua competência principal: codificar a lógica de negócios real. Muitos dos componentes padronizados do aplicativo, como os servidores web, bem como todo o trabalho pesado genérico, como o software para lidar com a confiabilidade e a escalabilidade, não exigem nenhum esforço do desenvolvedor. O que resta é uma abordagem limpa e funcional, em que a lógica de negócios é acionada apenas quando necessário: um usuário móvel envia uma mensagem, uma imagem é carregada na nuvem, registros chegam em um fluxo e assim por diante. Uma abordagem assíncrona e baseada em eventos para o design do aplicativo (embora não necessária) é muito comum em aplicativos sem servidor, uma vez que se encaixa perfeitamente no conceito do código que é executado (e incorre em custos) somente quando há trabalho a ser feito.

Um aplicativo sem servidor é executado na nuvem pública, em um serviço como o AWS Lambda, que é responsável pelos eventos recebidos ou pelas invocações do cliente, instanciando e executando o código. Esse modelo oferece várias vantagens em comparação ao design convencional de aplicativos baseados em servidor:

- Não há necessidade de provisionar, de implantar, de atualizar, de monitorar ou de gerenciar servidores. O hardware real e o software de servidor são completamente gerenciados pelo provedor de nuvem.
- O aplicativo é escalado automaticamente e acionado pelo seu uso real. Isso é inerentemente diferente dos aplicativos convencionais, que exigem uma frota de receptores e o gerenciamento explícito da capacidade para ser escalado de acordo com o pico de carga.
- Além da escalabilidade, a disponibilidade e a tolerância a falhas são integradas. Não há necessidade de codificação, de configuração ou de gerenciamento para obter o benefício desses recursos.
- A capacidade ociosa não é cobrada. Não é necessário (nem possível) provisionar com antecedência ou provisionar em excesso a capacidade excedente. Em vez disso, o faturamento é pago mediante solicitação e baseado na duração necessária para a execução do código.

Casos de uso de aplicativos sem servidor

O modelo de aplicativo sem servidor é genérico e aplica-se a praticamente qualquer tipo de aplicativo, desde um aplicativo web de uma startup até uma plataforma de análise do comércio de ações de uma empresa Fortune 100. Veja alguns exemplos:

- **Aplicativos web e sites** – A eliminação de servidores possibilita a criação de aplicativos web que custam quase nada quando não há tráfego e, ao mesmo tempo, são escalados para lidar com as cargas de pico, até mesmo com as inesperadas.
- **Back-ends móveis** – Os back-ends móveis sem servidor permitem aos desenvolvedores que priorizam o desenvolvimento de clientes criar facilmente back-ends seguros, altamente disponíveis e perfeitamente escalados, sem serem especialistas em design de sistemas distribuídos.
- **Processamento de logs e de mídias** – As abordagens sem servidor oferecem paralelismo natural, o que simplifica o processamento de cargas de trabalho pesadas de computação, sem a complexidade da criação de sistemas com vários threads ou da escalabilidade manual das frotas de computação.

- **Automação da TI** – As funções sem servidor podem ser anexadas a alarmes e a monitores para fornecer personalização quando necessário. As tarefas Cron e os outros requisitos da infraestrutura de TI são significativamente mais simples de implementar com a remoção do requisito de posse e de manutenção dos servidores para uso próprio, especialmente quando essas tarefas e requisitos são pouco frequentes ou variáveis.
- **Back-ends de IoT** – A capacidade de aproveitar qualquer código, inclusive bibliotecas nativas, simplifica o processo de criação de sistemas baseados em nuvem que podem implementar algoritmos específicos dos dispositivos.
- **Chatbots (incluindo assistentes controlados por voz) e outros sistemas baseados em webhooks** – As abordagens sem servidor são ideais para sistemas baseados em webhooks, como um chatbot. A capacidade de realizar ações (como executar código) somente quando necessário (por exemplo, quando um usuário solicita informações de um chatbot) torna essa abordagem direta e normalmente de baixo custo para essas arquiteturas. Por exemplo, a maioria das habilidades do Alexa para o Amazon Echo são implementadas usando o AWS Lambda.
- **Sequência de cliques e outros processos de dados de streaming** – As soluções sem servidor oferecem flexibilidade para aumentar e diminuir o fluxo de dados, o que permite atender aos requisitos de produtividade sem a complexidade da criação de um sistema de computação escalável para cada aplicativo. Em combinação com uma tecnologia como a do Amazon Kinesis, o AWS Lambda pode oferecer processamento de alta velocidade de registros para análise de sequência de cliques, gatilhos de dados NoSQL, informações sobre o comércio de ações e muito mais.

Além dos casos de uso amplamente adotados discutidos anteriormente, as empresas também estão aplicando abordagens sem servidor aos seguintes domínios:

- Big data, como problemas de mapeamento e de redução, transcodificação de vídeo de alta velocidade, análise do comércio de ações e simulações de Monte Carlo com uso intensivo de computação para solicitações de empréstimo. Os desenvolvedores descobriram que é muito mais fácil aproveitar paralelamente uma abordagem sem servidor,[3] principalmente quando acionada por eventos, levando-os a aplicar cada vez mais as técnicas sem servidor a uma grande variedade de problemas de big data, sem a necessidade de gerenciamento de infraestrutura.

- Baixa latência, processamento personalizado para aplicativos web e ativos distribuídos por meio de redes de entrega de conteúdo. Ao transferir a manipulação de eventos sem servidor para a borda da Internet, os desenvolvedores podem aproveitar a latência mais baixa e a capacidade de personalizar facilmente as recuperações e as obtenções de conteúdo. Isso permite um novo espectro de casos de uso otimizados em relação à latência com base na localização do cliente.
- Dispositivos conectados que permitem que os recursos sem servidor, como as funções do AWS Lambda, sejam executados em dispositivos de Internet das Coisas (IoT) comerciais, residenciais e portáteis. As soluções sem servidor, como as funções Lambda, oferecem uma abstração natural do hardware físico (e até mesmo virtual) subjacente, o que facilita a transição do datacenter para a borda e de uma arquitetura de hardware para outra, sem interromper o modelo de programação.
- Lógica personalizada e manipulação de dados em dispositivos no local, como o AWS Snowball Edge. Por separar a lógica de negócios dos detalhes do ambiente de execução, os aplicativos sem servidor podem funcionar facilmente em uma grande variedade de ambientes, inclusive em um dispositivo.

Normalmente, os aplicativos sem servidor são criados usando uma *arquitetura de microsserviços,* em que um aplicativo é separado em componentes independentes que executam trabalhos distintos. Esses componentes, que são compostos por funções Lambda individuais, juntamente com APIs, filas de mensagens, banco de dados e outros componentes, podem ser implementados, testados e escalados de maneira independente. Na verdade, os aplicativos sem servidor são ideais para microsserviços devido ao modelo baseado em funções. Ao evitar arquiteturas e designs monolíticos, as organizações podem se tornar mais ágeis, uma vez que os desenvolvedores podem realizar implantações incrementais e substituir ou atualizar componentes individuais, como o nível do banco de dados, se necessário.

Em muitos casos, basta isolar a lógica de negócios de um aplicativo para convertê-lo em um aplicativo sem servidor. Serviços como o AWS Lambda dão suporte a linguagens de programação populares e permitem o uso de bibliotecas personalizadas. Tarefas de longa duração são expressas como fluxos de trabalho compostos de funções individuais que funcionam em períodos razoáveis, o que permite que o sistema reinicie ou paralelize unidades de computação individuais conforme necessário.

A abordagem sem servidor é sempre adequada?

Quase todos os aplicativos modernos podem ser modificados para serem executados com êxito e, na maioria dos casos, de maneira mais econômica e escalável em uma plataforma sem servidor. No entanto, em certos casos a abordagem sem servidor não é a melhor opção:

- Quando o objetivo é explicitamente evitar alterações em um aplicativo.
- Quando um controle refinado do ambiente é necessário, como a especificação de patches específicos de sistemas operacionais ou o acesso a operações de redes de baixo nível, para que o código seja executado corretamente.
- Quando um aplicativo no local não foi migrado para a nuvem pública.

Avaliação da plataforma sem servidor de um provedor de nuvem

Ao arquitetar um aplicativo sem servidor, as empresas e as organizações precisam considerar não só a funcionalidade de computação sem servidor que executa o código do aplicativo. Os aplicativos completos sem servidor exigem uma ampla variedade de serviços, de ferramentas e de recursos que incluem armazenamento, sistema de mensagens, diagnóstico e muito mais. Um portfólio sem servidor incompleto ou fragmentado de um provedor de nuvem pode ser problemático para desenvolvedores sem servidor, que podem ter de retornar às arquiteturas baseadas em servidor caso não consigam codificar com êxito em um nível consistente de abstração.

Uma plataforma sem servidor é formada pelo conjunto de serviços que compõem o aplicativo sem servidor, como os componentes de computação e de armazenamento, bem como as ferramentas necessárias para criar, implantar e diagnosticar os aplicativos sem servidor. A execução de um aplicativo sem servidor em produção exige uma plataforma confiável e flexível e que possa atender às demandas de pequenas startups até corporações globais no mundo todo. A plataforma deve escalar *todos* os elementos do aplicativo e fornecer confiabilidade de ponta a ponta. Como nos aplicativos convencionais, ajudar os desenvolvedores a criar e fornecer soluções sem servidor é um desafio multidimensional. Para atender às necessidades das empresas de grande porte em vários setores, uma plataforma sem servidor deve oferecer os seguintes recursos:

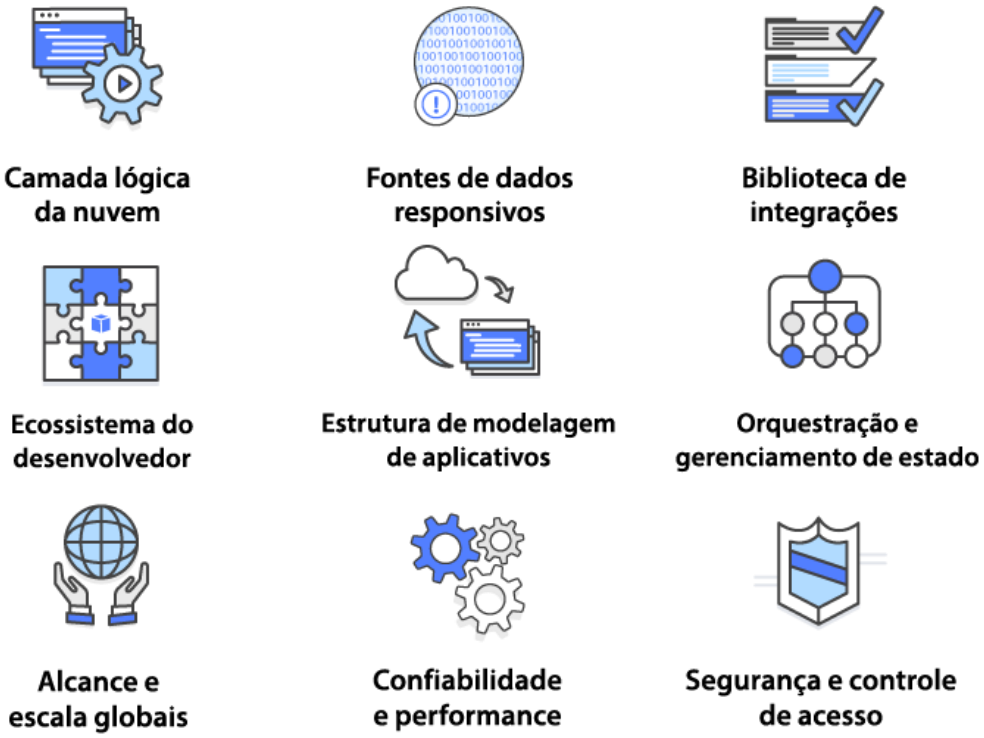

Figura 1: Os recursos de uma plataforma sem servidor

- Uma *camada de lógica de nuvem* de alta performance, escalável e confiável.
- *Fontes de dados e de eventos* próprias responsivas e *conectividade simples a sistemas de terceiros*.
- *Bibliotecas de integração* que permitem um início fácil aos desenvolvedores e a adição rápida e segura de novos padrões às soluções existentes.
- Um *ecossistema para desenvolvedores* vibrante que os ajuda a descobrir e aplicar soluções a vários domínios e a um grande conjunto de casos de uso e de sistemas de terceiros.
- Uma coleção de *estruturas de modelagem de aplicativos* adequada aos propósitos desejados.
- *Orquestração* oferecendo gerenciamento de estados e de fluxos de trabalho.
- *Escala global* e grande alcance que inclui certificação de programa de garantia.

- *Confiabilidade integrada e performance em escala*, sem a necessidade de provisionar capacidade em nenhum nível de escala.
- *Segurança integrada*, juntamente com *controle de acesso flexível* para recursos e serviços próprios e de terceiros.

No núcleo de qualquer plataforma sem servidor reside a *camada de lógica de nuvem* responsável por executar as funções que representam a lógica de negócios. Como essas funções normalmente são executadas em resposta a eventos, a simples *integração a fontes de eventos próprias e de terceiros* é essencial para simplificar a expressão das soluções e para permitir que sejam escaladas automaticamente em resposta às cargas de trabalho variáveis. Por exemplo, as funções sem servidor podem precisar ser executadas sempre que um objeto é criado em um depósito de objetos ou em toda atualização em um banco de dados NoSQL sem servidor. As arquiteturas sem servidor eliminam o código da escalabilidade e do gerenciamento normalmente necessário para integrar esses sistemas, transferindo essa carga operacional ao provedor de nuvem.

Desenvolver com êxito em uma plataforma sem servidor exige que uma empresa possa começar facilmente, inclusive ao encontrar modelos prontos para casos de uso comuns, independentemente de envolverem serviços próprios ou de terceiros. Essas *bibliotecas de integração* são essenciais para transmitir padrões bem-sucedidos (como no processamento de fluxos de registros ou na implementação de webhooks), principalmente durante o período em que os desenvolvedores estão migrando de arquiteturas baseadas em servidor para as sem servidor. Uma necessidade intimamente relacionada é ter um *ecossistema amplo e diversificado* ao redor da plataforma principal. Um ecossistema grande e vibrante ajuda os desenvolvedores a descobrir e a usar soluções da comunidade e facilita a contribuição de novas ideias e abordagens. Dada a variedade de cadeias de ferramentas em uso para o gerenciamento do ciclo de vida dos aplicativos, um ecossistema íntegro também é necessário para assegurar que cada linguagem, Integrated Development Environment (IDE – Ambiente de desenvolvimento integrado) e tecnologia de criação empresarial tenha tempos de execução, plug-ins e soluções de software livre necessários para integrar a criação e a implantação dos aplicativos sem servidor às abordagens existentes. É fundamental também que os aplicativos sem servidor aproveitem os investimentos existentes, incluindo o conhecimento dos desenvolvedores em relação às estruturas como o Express e o Flask e às linguagens de programação populares. Um ecossistema amplo fornece uma aceleração importante entre domínios e permite que os desenvolvedores adaptem o código existente mais prontamente em uma arquitetura sem servidor.

As *estruturas de modelagem de aplicativos*, como o AWS Serverless Application Model (AWS SAM) de especificação aberta, permitem que um desenvolvedor expresse os componentes que formam um aplicativo sem servidor e habilite as ferramentas e os fluxos de trabalho necessários para criar, implantar e monitorar esses aplicativos. Outra estrutura fundamental para o sucesso de uma plataforma sem servidor é o *gerenciamento de orquestrações e de estados*. A natureza em grande parte stateless da computação sem servidor exige um mecanismo complementar para permitir fluxos de trabalho de longa duração. As soluções de orquestração permitem que os desenvolvedores coordenem os vários componentes típicos relacionados dos aplicativos em um aplicativo sem servidor, possibilitando ainda que esses aplicativos sejam compostos de funções pequenas e de curta duração. Os serviços de orquestração também simplificam o tratamento de erros e fornecem integração a sistemas legados e fluxos de trabalho, incluindo os que são executados por mais tempo que as próprias funções sem servidor normalmente permitem.

Para dar suporte a clientes em todo o mundo, inclusive às corporações multinacionais de alcance global, uma plataforma sem servidor deve oferecer *escala global*, com datacenters e pontos de presença espalhados no mundo todo. Os pontos de presença são importantes para aproximar a computação sem servidor de baixa latência aos usuários finais. Uma vez que a plataforma, e não o desenvolvedor do aplicativo, é responsável por fornecer a escalabilidade e a alta disponibilidade dos aplicativos sem servidor, a *confiabilidade* intrínseca deles é fundamental. Recursos como novas tentativas integradas e filas de mensagens mortas de eventos não processados ajudam os desenvolvedores a construir sistemas robustos com confiabilidade de ponta a ponta usando abordagens sem servidor. A performance é igualmente importante, principalmente com relação à baixa latência (sobrecarga), uma vez que os tempos de execução e o código do cliente são instanciados sob demanda em um aplicativo sem servidor.

Por fim, a plataforma deve ter uma grande variedade de *controles de segurança e de acesso*, incluindo suporte a redes virtuais privadas, permissões baseadas em funções e em acesso, integração robusta a mecanismos de controle de acesso e de autenticação baseados em APIs (incluindo sistemas legados e de terceiros) e suporte para criptografar elementos de aplicativos, como configurações de variáveis de ambiente. Os sistemas sem servidor, por seu design, oferecem um nível inerentemente maior de segurança e de controle pelos seguintes motivos:

- **Gerenciamento de frotas de primeira classe, incluindo aplicação de patches de segurança** – Em um sistema como o AWS Lambda, os servidores que executam as solicitações são submetidos a monitoramento, a ciclos de ativação/desativação e a verificações de segurança constantes. A aplicação de patches pode ocorrer em questão de horas da disponibilidade das atualizações de segurança importantes, ao contrário de muitas frotas de computação empresariais que podem ter SLAs mais flexíveis para a aplicação de patches e atualizações.
- **Vida útil limitada do servidor** – Cada computador que executa o código do cliente no AWS Lambda é submetido a ciclos de ativação/desativação várias vezes ao dia, o que limita a exposição a ataques e assegura a aplicação constante de patches e de atualizações de segurança e de sistema operacional.
- **Autenticação mediante solicitação, controle de acesso e auditoria** – Toda solicitação de computação executada no AWS Lambda, independentemente da origem, é autenticada, autorizada a acessar recursos especificados e totalmente auditada de maneira individual. As solicitações externas aos datacenters da AWS realizadas por meio do Amazon API Gateway fornecem sistemas de defesa adicionais voltados para a Internet, incluindo defesas contra ataques Denial of Service (DoS – Negação de serviço). As empresas que migram para arquiteturas sem servidor podem usar o AWS CloudTrail para obter informações detalhadas de quais usuários estão acessando quais sistemas e com quais privilégios, além de usar o AWS Lambda para processar os registros de auditoria por meio de programação.

A plataforma sem servidor da AWS

Desde a apresentação do Lambda em 2014, a AWS tem trabalhado na criação de uma plataforma completa sem servidor. Ela conta com uma grande coleção de serviços totalmente gerenciados permitindo às organizações criar aplicativos sem servidor que podem se integrar perfeitamente a outros serviços da AWS e de terceiros. A Figura ilustra um subconjunto dos componentes da plataforma sem servidor da AWS e suas relações.

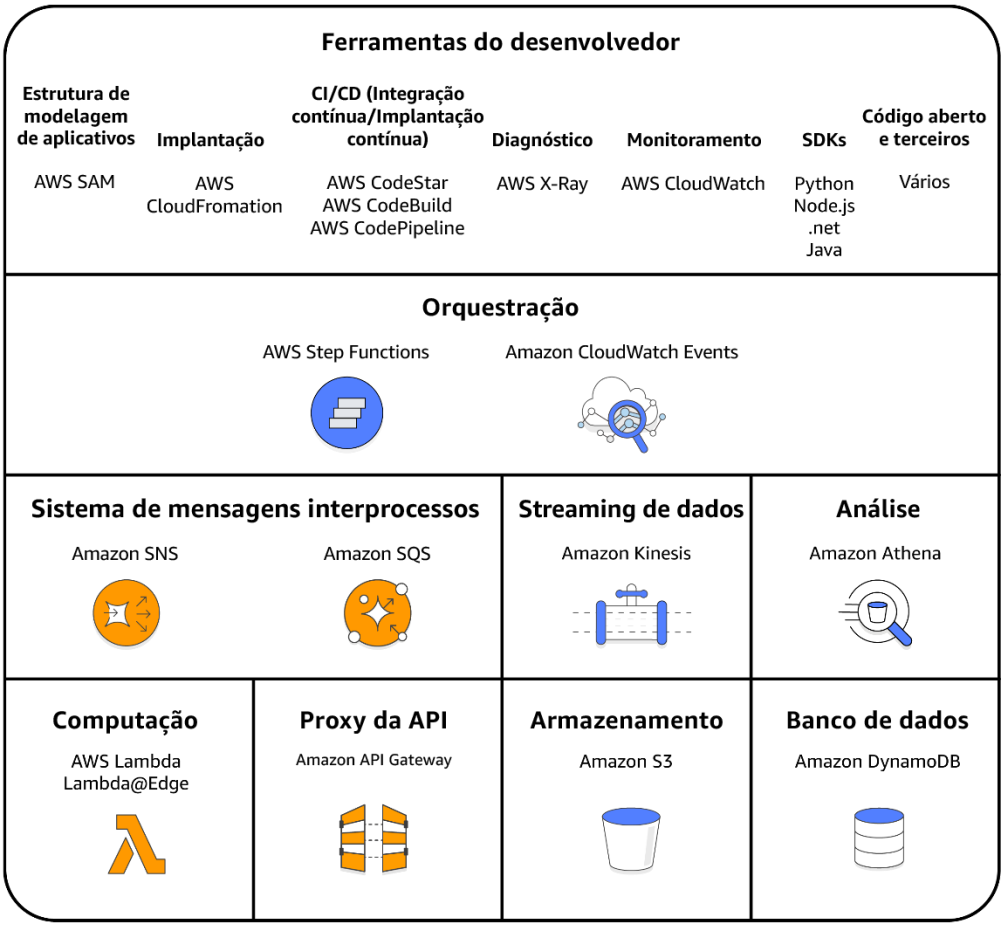

Figura 2: Os componentes da plataforma sem servidor da AWS

Os recursos da plataforma sem servidor da AWS

A AWS fornece todos os principais recursos identificados na seção anterior como requisitos para uma plataforma sem servidor completa. A camada lógica de nuvem é fornecida pelo AWS Lambda, uma oferta de computação sem servidor de larga escala, sem provisionamento e baseada em funções. O AWS Lambda é complementado pelo AWS Lambda@Edge, que oferece suporte semelhante para executar funções Lambda com latência extremamente baixa usando roteamento otimizado por borda, e pelo AWS Greengrass, que permite que as funções Lambda sejam executadas em dispositivos conectados, incluindo o AWS Snowball.

As funções Lambda podem ser facilmente acionadas por uma variedade de eventos próprios e de terceiros, o que permite aos desenvolvedores criar sistemas reativos e acionados por eventos (veja a Figura 3), sem o incômodo de precisar configurar e gerenciar a infraestrutura. Quando há vários eventos simultâneos, o Lambda simplesmente executa mais cópias da função em paralelo, respondendo a cada gatilho individual. As funções Lambda são escaladas precisamente de acordo com o tamanho da carga de trabalho até a solicitação individual. Como resultado, não é possível que um servidor ou contêiner fique ocioso. O problema de desperdício em despesas com infraestrutura é eliminado *pelo design* em arquiteturas que usam funções Lambda.

A *Função como Serviço* (FaaS) é uma abordagem para criar sistemas de computação baseados em eventos que depende das funções como a unidade de implementação e de execução. *FaaS sem servidor* é um tipo de FaaS em que não há máquinas virtuais nem contêineres no modelo de programação e em que o provedor fornece confiabilidade integrada e escalabilidade sem provisionamento.

Figura 3: A relação entre computação baseada em eventos, FaaS e FaaS sem servidor.

Os recursos de computação sem servidor da AWS fornecidos pelo Lambda são um elemento fundamental dos seguintes serviços gerenciados oferecidos pela AWS, os quais se integram perfeitamente uns aos outros:

- Amazon API Gateway – Endpoints HTTP para funções Lambda, incluindo uma gama completa de recursos de gerenciamento de API e de proxy de API.
- Amazon S3 – As funções Lambda podem ser usadas como gatilhos de eventos automáticos quando um objeto é criado, copiado ou excluído.
- Amazon DynamoDB – As funções Lambda podem ser usadas para processar todas as alterações feitas em uma tabela de banco de dados.
- Amazon SNS – As mensagens podem ser roteadas para as funções Lambda para processamento, o que adiciona a capacidade de responder dinamicamente ao conteúdo publicado.
- Amazon SQS – As mensagens nas filas podem ser facilmente processadas pelas funções Lambda.
- Amazon Kinesis Streams – O processamento de registros em sequência de dados de streaming é fornecido pelas funções Lambda, o que facilita a criação de mecanismos de análise quase em tempo real.
- Amazon Kinesis Firehose – As funções Lambda podem ser aplicadas automaticamente aos registros ingeridos por um Firehose, o que facilita a adição de recursos de transformação, de filtragem e de análise a um fluxo de dados.
- Amazon Athena – As funções Lambda podem ser acionadas automaticamente para cada objeto no conjunto de resultados de uma consulta.
- AWS Step Functions – Várias funções Lambda podem ser orquestradas para criar fluxos de trabalho de longa duração para processos centrados em humanos e automatizados.
- Amazon CloudWatch Events – As funções Lambda podem ser usadas para responder automaticamente a eventos, inclusive de terceiros.
- Amazon Aurora – Os gatilhos de banco de dados podem ser escritos como funções Lambda.

O Lambda fornece uma biblioteca de integração com blueprints para uma grande variedade de serviços de terceiros, incluindo Slack, Algorithmia, Twilio, Loggly, Splunk, SumoLogic, Box e outros, o que permite aos desenvolvedores criar aplicativos responsivos que incluem análises, algoritmos avançados, comunicações e muito mais com apenas algumas linhas de código. Uma grande variedade de estruturas de aplicativos web, incluindo o Express (para aplicativos NodeJS) e o Flask (para aplicativos Python), foi aprimorada para funcionar bem com as funções Lambda. Os projetos de aplicativos web de software livre incluem o Serverless Framework, o Sparta, o Chalice e muitos outros.

Os aplicativos sem servidor são formados normalmente por várias partes: uma ou mais funções, um banco de dados sem servidor, como o Amazon DynamoDB, bem como uma API para chamadas de clientes ou uma fonte de eventos que aciona o aplicativo. Para manter todas essas peças organizadas, a AWS usa o Serverless Application Model (SAM – Modelo de aplicativo sem servidor) de especificação aberta. Com o SAM, os desenvolvedores podem descrever facilmente as funções, as APIs, as fontes de eventos, as tabelas de banco de dados e as outras partes de um aplicativo sem servidor. Usar o SAM também ajuda os desenvolvedores a gerenciar todas as etapas do ciclo de vida de desenvolvimento de software, sendo que a AWS oferece uma variedade de ferramentas e de serviços para ajudá-los. Isso inclui suporte nativo para testes locais e depuração no IDE desejado (ou por meio de linha de comando) por meio do SAM Local, implantação de aplicativos SAM usando o AWS CloudFormation, suporte para criação de aplicativos SAM no AWS CodeBuild e suporte para CI/CD baseada no GitHub para aplicativos SAM integrados ao AWS CodePipeline. Além do suporte próprio, várias estruturas de software livre, provedores de CI/CD e fornecedores de gerenciamento de performance oferecem suporte para funções Lambda e SAM, incluindo o Serverless Framework, o Claudia, o CloudBees, o Datadog e muito mais. Para obter exemplos adicionais, visite Ferramentas para desenvolvedores de aplicativos sem servidor.[4]

Depois de criar uma função Lambda (ou, no caso de um aplicativo SAM, possivelmente várias funções Lambda operando juntas), os desenvolvedores poderão facilmente monitorá-la usando métricas e logs criados automaticamente no Amazon CloudWatch e no CloudWatch Logs. A AWS também oferece o AWS X-Ray, uma solução de análise de performance e de rastreamento de solicitações entre serviços que permite aos desenvolvedores rastrear a operação e o comportamento de funções individuais, bem como os eventos que elas processam.

As ofertas de plataformas sem servidor da AWS têm um alcance global, com suporte para o AWS Lambda e para o Amazon API Gateway em praticamente todas as regiões da AWS no mundo todo. O Lambda@Edge está disponível em todos os pontos de presença.[5] O Lambda oferece uma variedade de recursos para ajudar os clientes a melhorar a confiabilidade dos seus aplicativos, incluindo novas tentativas automáticas para eventos assíncronos e ordenados e filas de mensagens mortas para capturar eventos que não foram processados com êxito pelo aplicativo. A profunda integração à Amazon Virtual Private Cloud (Amazon VPC) e a variedade flexível dos recursos de autenticação e de controle de acesso fornecidos pelo AWS Lambda permitem que as organizações criem aplicativos seguros conforme as melhores práticas, como o princípio do menor privilégio. O gerenciamento e a segurança do usuário final são igualmente fáceis: o Amazon Cognito oferece autorização e autenticação que podem ser facilmente combinadas com o Amazon API Gateway e com o AWS Lambda, o que permite recursos de login e de registro de usuários sem servidor, incluindo a integração a provedores sociais como o Facebook e os diretórios corporativos.

Estudos de caso

As empresas aplicam arquiteturas sem servidor a casos de uso que incluem validação do comércio de ações, construção de sites de comércio eletrônico e processamento de linguagem natural. O AWS Lambda e o restante do portfólio sem servidor da AWS oferecem flexibilidade para a criação de uma grande variedade de aplicativos, incluindo os que exigem programas de garantia, como os de compatibilidade com a PCI ou com a HIPAA. As seções a seguir ilustram alguns dos casos de uso mais comuns, mas não se trata de uma lista completa. Para obter uma lista completa de referências de clientes e a documentação dos casos de uso, visite Computação sem servidor.[6]

Sites, aplicativos web e back-ends sem servidor

As abordagens sem servidor são ideais para os aplicativos que as cargas podem variar de maneira dinâmica. Usar uma abordagem sem servidor significa que nenhum custo computacional é incorrido quando não há tráfego do usuário final, oferecendo ainda escala instantânea para atender à alta demanda, como uma promoção relâmpago em um site de comércio eletrônico ou uma menção em mídia social que motiva uma onda súbita de tráfego. Em comparação com as abordagens de infraestruturas tradicionais, também é bem mais barato desenvolver, entregar e operar um back-end web ou móvel quando ele é arquitetado sem servidor.

A AWS fornece os serviços de que os desenvolvedores precisam para criar rapidamente esses aplicativos:

- O Amazon S3 oferece uma solução de hospedagem simples para conteúdo estático.
- O AWS Lambda, juntamente com o Amazon API Gateway, fornece suporte para solicitações de APIs dinâmicas usando funções.
- O Amazon DynamoDB oferece uma solução de armazenamento simples para sessão e estado baseado no usuário.
- O Amazon Cognito oferece uma maneira fácil de manipular o registro, a autenticação e o controle de acesso aos recursos do usuário final.
- O AWS SAM pode ser usado por desenvolvedores para descrever os vários elementos de um aplicativo.
- O AWS CodeStar pode configurar uma cadeia de ferramentas de CI/CD com apenas alguns cliques.

Para saber mais, leia o whitepaper Arquiteturas multicamadas sem servidor da AWS, que fornece uma análise detalhada dos padrões para a criação de aplicativos web sem servidor.[7] Para obter arquiteturas de referência completas, leia a Arquitetura de referência sem servidor para criar um aplicativo web[8] e a Arquitetura de referência sem servidor para criar um back-end móvel[9] no GitHub.

Exemplo de cliente – Bustle.com

O Bustle.com é um site de notícias, entretenimento, estilo de vida e moda voltado para as mulheres. Ele alcançou uma redução de custos de aproximadamente 84% ao migrar para uma arquitetura sem servidor baseada no AWS Lambda e no Amazon API Gateway. Os engenheiros do Bustle ganharam mais agilidade, permitindo que eles se concentrassem na criação de novos recursos do produto, em vez de lidarem com o gerenciamento e a escalabilidade da infraestrutura. A equipe do Bustle agora é mais eficiente e usa metade das pessoas normalmente necessárias para criar e operar os sites da escala do Bustle. O back-end sem servidor do Bustle também dá suporte a aplicativos para iOS em duas de suas propriedades da web (Bustle e Romper). Para obter mais informações, leia o Estudo de caso do Bustle.[10]

Back-ends de IoT

Os benefícios que uma arquitetura sem servidor oferece aos aplicativos web e móveis também facilitam a construção de back-ends de IoT e de sistemas de processamento analíticos baseados em dispositivos que são escalados perfeitamente de acordo com o número de dispositivos. Para obter uma arquitetura de referência de exemplo, leia a Arquitetura de referência sem servidor para criar um back-end de IoT no GitHub.[11]

Exemplo de cliente – iRobot

A iRobot, que fabrica robôs como o robô de limpeza Roomba, usa o AWS Lambda em conjunto com o serviço AWS IoT para criar um back-end sem servidor para sua plataforma de IoT. Usando uma arquitetura sem servidor, a equipe de engenharia da iRobot não precisa se preocupar em gerenciar a infraestrutura nem em codificar manualmente para lidar com a disponibilidade e com a escalabilidade. Isso permite que eles inovem mais rapidamente e mantenham o foco nos clientes. Para obter mais informações, veja os slides da apresentação deles na AWS re:Invent 2016 Serverless IoT Back Ends (IOT401)[12] ou assista ao vídeo.[13]

Processamento de dados

Os maiores aplicativos sem servidor processam grandes volumes de dados e, em grande parte, em tempo real. As arquiteturas típicas de processamento de dados sem servidor usam uma combinação do Amazon Kinesis e do AWS Lambda para processar dados de streaming ou combinam o Amazon S3 e o AWS Lambda para acionar a computação em resposta a eventos de criação ou de atualização de objetos. Quando as cargas de trabalho exigem uma orquestração mais complexa do que um simples gatilho, os desenvolvedores podem usar o AWS Step Functions para criar fluxos de trabalho stateful ou de longa duração que invocam uma ou mais funções Lambda conforme progridem. Para saber mais sobre as arquiteturas de processamento de dados sem servidor, leia os seguintes artigos no GitHub:

- Arquitetura de referência sem servidor para processamento de fluxos em tempo real[14]
- Arquitetura de referência sem servidor para processamento de arquivos em tempo real[15]
- Arquitetura de referência de back-end de processamento e de reconhecimento de imagens[16]

Exemplo de cliente – FINRA

A Financial Industry Regulatory Authority (FINRA) usou o AWS Lambda para criar uma solução de processamento de dados sem servidor que lhes permite executar meio trilhão de validações de dados em 37 bilhões de eventos do mercado de ações diariamente. Em sua palestra na AWS re:Invent 2016 intitulada The State of Serverless Computing (SVR311),[17] Tim Griesbach, diretor sênior da FINRA, disse: "Descobrimos que o Lambda nos forneceria a melhor solução para essa solução de nuvem sem servidor. Com o Lambda, o sistema foi mais rápido, barato e escalável. Então, no final das contas, reduzimos nossos custos em mais de 50% e podemos rastreá-lo diariamente, até mesmo de hora em hora."

Exemplo de cliente – Thomson Reuters

A Thomson Reuters, uma empresa de meios de comunicação e informação, criou uma solução de análise de negócios sem servidor que permite que suas equipes de produtos analisem facilmente os dados de uso dos produtos. A solução combina o AWS Lambda, o Amazon Kinesis Streams e o Amazon Kinesis Firehose para coletar e processar dados de eventos de streaming para análise. O resultado, chamado Product Insight, foi lançado dois meses antes do prazo e superou as expectativas técnicas.

Anders Fritz, gerente sênior de inovação de produtos da Thomson Reuters, disse: "Nosso objetivo inicial era acomodar 2.000 eventos por segundo. Nossos testes mostram que o Product Insight na AWS pode processar até 4.000 eventos por segundo e em um ano esperamos aumentar para mais de 10.000 eventos por segundo". Esse número representa mais de 25 bilhões de eventos por mês. Mesmo com essa alta produtividade, o sistema não perdeu nenhum dado desde a criação. "Devido à arquitetura robusta de failover e aos recursos técnicos da AWS, não perdemos um único evento desde que começamos a coletar dados", afirma Fritz. Para obter mais informações, leia o Estudo de caso da Thomson Reuters[18] ou assista à apresentação na AWS re:Invent 2016 Real-time Data Processing Using AWS Lambda (SVR301).[19]

Big data

O AWS Lambda é perfeito para muitas cargas de trabalho de processamento paralelo de alto volume. Para obter um exemplo de arquitetura de referência usando o MapReduce, leia a Arquitetura de referência para executar tarefas MapReduce sem servidor.[20]

Exemplo de cliente – Fannie Mae

A Fannie Mae, uma das principais fontes de financiamento de bancos imobiliários, usa o AWS Lambda para executar uma carga de trabalho "paralelamente embaraçosa" para sua modelagem financeira. A Fannie Mae usa processos de simulação de Monte Carlo para projetar fluxos de caixa futuros das hipotecas que a ajudam a gerenciar o risco das hipotecas. A empresa descobriu que suas grades de computação de alta performance (HPC) existentes não estavam mais atendendo às crescentes necessidades comerciais. A Fannie Mae criou sua nova plataforma no Lambda e o sistema foi expandido com sucesso chegando até a 15.000 execuções simultâneas de funções durante o teste. O novo sistema executou uma simulação em 20 milhões de hipotecas que foi concluída em duas horas, três vezes mais rápido que o sistema antigo. Usando uma arquitetura sem servidor, a Fannie Mae agora pode executar simulações de Monte Carlo em grande escala de maneira econômica, já que não paga por recursos de computação ociosos. Ela também pode acelerar seus cálculos executando várias funções Lambda de maneira simultânea. O tempo de entrada no mercado da Fannie Mae também foi mais curto que o normal, pois ela pôde dispensar o gerenciamento e o monitoramento do servidor, além de eliminar grande parte do código complexo antes necessário para gerenciar a escalabilidade e a confiabilidade do aplicativo. Para obter mais informações, veja a apresentação da Fannie Mae no AWS Summit 2017 SMC303: Real-time Data Processing Using AWS Lambda.[21]

Automação da TI

As abordagens sem servidor eliminam a sobrecarga do gerenciamento de servidores, o que facilita muito a criação e o gerenciamento de tarefas de infraestrutura, como o provisionamento, a configuração, o gerenciamento, os alarmes/monitores e as tarefas cron sincronizadas.

Exemplo de cliente – Autodesk

A Autodesk, criadora de programas de software de engenharia e de design 3D, usa o AWS Lambda para automatizar os processos de criação e de gerenciamento de contas da AWS em toda a sua organização de engenharia. A Autodesk estima uma economia de custos de 98% (considerando as economias estimadas em horas de mão de obra gastas provisionando contas). Agora ela pode provisionar contas em apenas dez minutos, em vez das dez horas necessárias para provisionar do processo anterior baseado em infraestrutura. A solução sem servidor permite que a Autodesk provisione automaticamente contas, configure e imponha

normas, além de executar auditorias com maior automação e menos pontos de contato manuais. Para obter mais informações, veja a apresentação da Autodesk no AWS Summit 2017 SMC301: The State of Serverless Computing.[22] Visite o GitHub para saber mais sobre o serviço personalizado da Autodesk.

Casos de uso adicionais

Os casos de uso descritos na seção anterior abordam apenas superficialmente o que é possível fazer com o Lambda e com as outras ofertas sem servidor da AWS. Outros casos de uso incluem uma capacidade avançada de compreensão da linguagem humana por meio de chatbots criados com o Amazon Lex e com o AWS Lambda, a computação de borda global com baixa latência usando o Lambda@Edge com o Amazon CloudFront e o processamento avançado de arquivos no local com funções Lambda dentro de um AWS Snowball. Esses são apenas alguns dos recursos interessantes dessa abordagem versátil. Para saber mais, visite o site do AWS Lambda.[23]

Conclusão

As abordagens sem servidor são desenvolvidas para lidar com dois problemas clássicos de gerenciamento de TI: os servidores ociosos que sugam o balanço patrimonial de uma empresa sem oferecer valor e o custo da criação e da operação de frotas de servidores e de software de servidor, que distraem e desviam os negócios da criação de valor diferenciado para os clientes. O AWS Lambda e as outras ofertas sem servidor da AWS resolvem esses problemas de longa data eliminando os servidores, os contêineres, os discos e os outros recursos em nível de infraestrutura do modelo de programação e de faturamento. Como resultado, os desenvolvedores podem trabalhar com um modelo de aplicativo limpo que os ajuda a entregar mais rapidamente e as organizações pagam apenas pelo trabalho útil. A maneira mais fácil e rápida de arquitetar sistemas reativos baseados em eventos e entregar microsserviços nativos da nuvem é usar arquiteturas sem servidor. Para saber mais e ler os whitepapers sobre tópicos relacionados, visite Computação e aplicativos sem servidor.[24]

Colaboradores

As seguintes organizações e pessoas contribuíram para este documento:

- Tim Wagner, gerente geral dos aplicativos sem servidor da AWS, Amazon Web Services

Leitura adicional

Para obter mais informações, leia estas fontes:

- Serverless Reference Architectures with AWS Lambda, de Werner Vogels, CTO do Amazon.com
- AWS re:Invent 2016: The State of Serverless Computing [apresentação] de Tim Wagner, gerente geral de aplicativos sem servidor da AWS
- The economics of serverless cloud computing, de Owen Rogers, diretor de pesquisa da 451 Research

Arquitetura de referência

- Aplicativos web
- Back-ends móveis
- Back-ends de IoT
- Processamento de arquivos
- Processamento de fluxos
- Processamento de reconhecimento de imagens
- MapReduce

Revisões do documento

Data	Descrição
Setembro de 2017	Primeira publicação

Notes

[1] https://www.forbes.com/sites/moorinsights/2016/04/11/tco-analysis-demonstrates-how-moving-to-the-cloud-can-save-your-company-money/#537e2bd07c4e
http://www.cloudstrategymag.com/articles/86033-understanding-tco-cloud-economics

[2] Em 2012, as estimativas do Gartner em relação à utilização dos datacenters subiram de 7 a 12% (http://www.nytimes.com/2012/09/23/technology/data-centers-waste-vast-amounts-of-energy-belying-industry-image.html). Em 2008, um estudo da McKinsey estimou tal utilização em 6% (https://www.sallan.org/pdf-docs/McKinsey_Data_Center_Efficiency.pdf). Um artigo da Accenture que analisava um conjunto de aplicativos baseados no EC2 estimou uma utilização de aproximadamente 7% (http://ieeexplore.ieee.org/document/6118751/). Em um estudo de 2014, o NRDC e a Anthesis constataram que, em 2013, mais de 30% dos servidores estavam em "coma" profundo (conectados, mas sem fazer nada de valor) (http://anthesisgroup.com/wp-content/uploads/2015/06/Case-Study_DataSupports30PercentComatoseEstimate-FINAL_06032015.pdf).

[3] Occupy the Cloud: Eric Jonas et al., *Distributed Computing for the 99%*, https://arxiv.org/abs/1702.04024.

[4] https://aws.amazon.com/serverless/developer-tools

[5] https://aws.amazon.com/lambda/edge/

[6] https://aws.amazon.com/serverless/

[7] https://d0.awsstatic.com/whitepapers/AWS_Serverless_Multi-Tier_Architectures.pdf

[8] https://github.com/awslabs/lambda-refarch-webapp

[9] https://github.com/awslabs/lambda-refarch-mobilebackend

[10] https://aws.amazon.com/solutions/case-studies/bustle/

[11] https://github.com/awslabs/lambda-refarch-iotbackend

[12] https://www.slideshare.net/AmazonWebServices/aws-reinvent-2016-serverless-iot-back-ends-iot401

[13] https://www.youtube.com/watch?v=gKMaf5E-z7Q

[14] https://github.com/awslabs/lambda-refarch-streamprocessing

[15] https://github.com/awslabs/lambda-refarch-fileprocessing

[16] https://github.com/awslabs/lambda-refarch-imagerecognition

[17] https://www.youtube.com/watch?v=AcGv3qUrRC4&feature=youtu.be&t=1153

[18] https://aws.amazon.com/solutions/case-studies/thomson-reuters/

[19] https://www.youtube.com/watch?v=VFLKOy4GKXQ&feature=youtu.be&t=1449

[20] https://github.com/awslabs/lambda-refarch-mapreduce

[21] https://www.slideshare.net/AmazonWebServices/smc303-realtime-data-processing-using-aws-lambda/28

[22] https://www.slideshare.net/AmazonWebServices/smc301-the-state-of-serverless-computing-75290821/22

[23] https://aws.amazon.com/lambda/

[24] https://aws.amazon.com/serverless/