

Arquiteturas Multicamadas Sem Servidor da AWS

Como usar o Amazon API Gateway e o AWS Lambda

Novembro de 2015



© 2015, Amazon Web Services, Inc. ou suas afiliadas. Todos os direitos reservados.

Avisos

Este documento é fornecido 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 “no estado em que se encontra”, 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 obrigações da AWS em relação aos seus clientes são controladas por acordos da AWS, e este documento não integra nem modifica qualquer acordo entre a AWS e seus clientes.

Sumário

| | |
|--|----|
| Resumo | 3 |
| Introdução | 4 |
| Visão geral da arquitetura de três camadas | 5 |
| A camada lógica sem servidor | 6 |
| Amazon API Gateway | 7 |
| AWS Lambda | 10 |
| A camada de dados | 13 |
| A camada de apresentação | 15 |
| Amostra dos padrões de arquitetura | 15 |
| Back-end móvel | 16 |
| Site hospedado do Amazon S3 | 17 |
| Ambiente de microserviços | 18 |
| Conclusão | 19 |
| Colaboradores | 19 |
| Observações | 20 |

Resumo

Este whitepaper mostra a você como inovações da Amazon Web Services (AWS) podem mudar a forma como você projeta arquiteturas multicamadas para padrões populares, como microserviços, back-ends móveis e sites públicos. Arquitetos e desenvolvedores já podem usar um padrão de implementação que inclui [Amazon API Gateway](#) e [AWS Lambda](#) para reduzir os ciclos de desenvolvimento e operações necessários para criar e gerenciar operacionalmente aplicações multicamadas.

Introdução

A aplicação multicamadas (três camadas, n camadas, etc.) é um padrão arquitetural básico há décadas. O padrão multicamadas fornece boas diretrizes para você seguir, garantindo componentes de aplicações desacoplados e escaláveis que podem receber gerenciamento e manutenção separados (muitas vezes por equipes distintas). Aplicações multicamadas costumam ser construídas usando uma abordagem SOA (service-oriented architecture, arquitetura orientada a serviços) em relação ao uso de serviços da web. Nessa abordagem, a rede age como a ligação entre camadas. No entanto, existem vários aspectos indiferenciados da criação de uma nova camada de serviços da web como parte da sua aplicação. Boa parte do código escrito dentro de uma aplicação web multicamadas é resultado direto do padrão em si. Entre os exemplos estão um código que integra uma camada a outra, um código que define uma API e um modelo de dados que as camadas usam para se entenderem entre si, bem como código relacionado à segurança que garante que os pontos de integração de camadas não sejam expostos de forma indesejada.

O [Amazon API Gateway](#)¹, um serviço para criação e gerenciamento de APIs, e o [AWS Lambda](#)², um serviço para executar funções de código arbitrárias, podem ser usados em conjunto para simplificar a criação de robustas aplicações multicamadas.

A integração do Amazon API Gateway com o AWS Lambda permite que funções de código definidas pelo usuário sejam acionadas diretamente via solicitação de HTTPS definida pelo usuário. Independentemente do volume de solicitação exigido, tanto o API Gateway quanto o Lambda serão automaticamente dimensionados para dar conta exata das necessidades da sua aplicação. Quando combinados, você pode criar uma camada para sua aplicação que lhe permita escrever o código que importa para sua aplicação e *não* se concentrar em vários outros aspectos indiferenciadores da implementação de uma arquitetura multicamadas – como criação de arquitetura para alta disponibilidade, criação de SDKs do cliente, gerenciamento de servidor/sistema operacional (SO), escalabilidade e implementação de um mecanismo de autorização de cliente.

Mais recentemente, a AWS anunciou a capacidade de criar funções Lambda executadas dentro da sua [Amazon Virtual Private Cloud \(Amazon VPC\)](#)³. Esse recurso amplia os benefícios da combinação do API Gateway com o Lambda, incluindo diversos casos de uso nos quais a privacidade de rede é exigida. Por exemplo, quando você precisa integrar seu serviço da web a um banco de dados relacional que contém informações confidenciais. A integração do Lambda e da Amazon VPC expandiu indiretamente os recursos do Amazon API Gateway, pois fornece aos desenvolvedores a capacidade de definir o próprio conjunto de APIs HTTPS acessíveis pela Internet perante um back-end que continua privado e protegido como parte da Amazon VPC. É possível observar os benefícios desse poderoso padrão entre cada camada de uma arquitetura multicamadas. Este whitepaper se concentra no exemplo mais conhecido da arquitetura multicamadas: a aplicação web de **três camadas**. No entanto, você pode utilizar esse padrão multicamadas muito além da aplicação web típica de três camadas.

Visão geral da arquitetura de três camadas

A arquitetura de três camadas é um padrão popular para aplicações voltadas para o usuário. As camadas que formam essa arquitetura incluem a **camada de apresentação**, a **camada lógica** e a **camada de dados**. A camada de apresentação representa o componente com o qual os usuários interagem diretamente (como página da web, interface da aplicação móvel, etc.). A camada lógica contém o código necessário para converter as ações do usuário feitas na camada de apresentação em funcionalidades que orientam o comportamento da aplicação. A camada de dados é formada por mídia de armazenamento (bancos de dados, depósito de objetos, caches, sistemas de arquivos, etc.) que mantém os dados relevantes à aplicação. A Figura 1 mostra um exemplo de uma aplicação simples de três camadas.

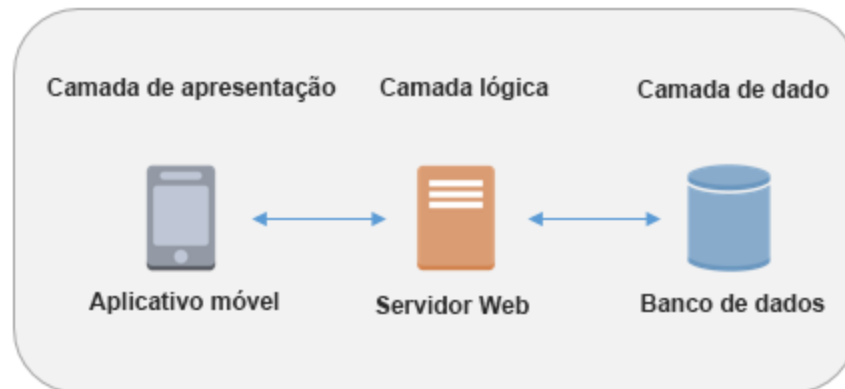


Figura 1: Padrão de arquitetura para uma aplicação simples de três camadas

Existem vários recursos excelentes online nos quais você pode aprender mais sobre o *padrão geral* de arquitetura de três camadas. Este whitepaper se concentra em um padrão de implementação específico para essa arquitetura usando o Amazon API Gateway e o AWS Lambda.

A camada lógica sem servidor

A camada lógica da arquitetura de três camadas representa o cérebro da sua aplicação. É por isso que integrar o Amazon API Gateway e o AWS Lambda para formar sua camada lógica pode ser tão revolucionário. Os recursos dos dois serviços lhe permitem construir uma aplicação de produção sem servidor altamente disponível, escalável e segura. Sua aplicação poderia usar milhares de servidores, mas, ao usar esse padrão, você não precisará gerenciar nenhum. Além disso, usando esses serviços gerenciados juntos, você ganha os seguintes benefícios:

- Nenhum sistema operacional para escolher, proteger, corrigir ou gerenciar.
- Nenhum serviço para corrigir o tamanho, monitorar ou expandir.
- Nenhum risco aos seus custos pelo superprovisionamento.
- Nenhum risco ao seu desempenho pelo subprovisionamento.

Além disso, existem recursos específicos em cada serviço que beneficiam o padrão de arquitetura multicamadas.

Amazon API Gateway

O Amazon API Gateway é um serviço totalmente gerenciado para definir, implementar e manter APIs. Os clientes se integram com APIs usando solicitações HTTPS padrão. Sua aplicabilidade a uma arquitetura multicamadas orientada a serviços é evidente. No entanto, ele tem recursos e qualidades específicos que o tornam uma presença poderosa para sua camada lógica.

Integração com o AWS Lambda

O Amazon API Gateway fornece à sua aplicação uma forma simples (solicitações HTTPS) de aproveitar diretamente a inovação do AWS Lambda. O API Gateway é a ponte que conecta sua camada de apresentação com as funções escritas no AWS Lambda. Depois de definir o relacionamento cliente/servidor usando sua API, o conteúdo da solicitação HTTPS do cliente é passado à função do Lambda para execução. Os conteúdos incluem metadados da solicitação, cabeçalhos da solicitação e corpo da solicitação.

Desempenho estável da API em todo o mundo

Cada implantação do Amazon API Gateway abrange uma distribuição do [Amazon CloudFront](#)⁴. O Amazon CloudFront é um serviço da web de entrega de conteúdo que usa a rede global da Amazon de pontos de presença como pontos de conexão para clientes que se integram à sua API. Isso ajuda a diminuir a latência do tempo de resposta total da sua API. Com o uso de vários pontos de presença em todo o mundo, o Amazon CloudFront também lhe fornece recursos para combater os cenários de ataque DDoS (distributed denial of service, ataques distribuídos de negação de serviço). Para obter mais informações, leia o whitepaper [AWS Best Practices for Combatting DDoS Attacks](#)⁵ (Melhores práticas da AWS para combater ataques DDoS).

Você pode melhorar o desempenho de solicitações específicas da API usando o Amazon API Gateway para armazenar respostas em um cache de memória opcional. Isso não só proporciona benefícios de desempenho para solicitações repetidas de API como também reduz execuções de back-end, que podem reduzir o custo geral.

Incentivo à inovação

O trabalho de desenvolvimento necessário para construir qualquer aplicação nova é um investimento. Você precisa justificar isso a fim de iniciar o projeto. Ao reduzir a quantidade de investimento necessária para as tarefas de desenvolvimento e o tempo, você fica livre para mais experimentos e para inovar com mais liberdade.

Para várias aplicações multicamadas baseadas em serviços da web, a camada de apresentação é facilmente fragmentada entre os usuários (dispositivos móveis independentes, navegadores da web, etc.). Esses usuários também não costumam estar geograficamente limitados. Uma camada lógica desacoplada, no entanto, não é fisicamente fragmentada pelos usuários. Todos os usuários dependem da mesma infraestrutura executada na sua camada lógica, o que amplia a importância da infraestrutura. Muitas vezes se propõe pegar atalhos ao implementar inicialmente sua camada lógica ("não precisamos instrumentar as métricas no lançamento"; "o uso inicial será baixo, vamos nos preocupar com a escalção depois"; etc.) como mecanismo para entregar uma nova aplicação com mais rapidez. Isso pode acarretar débito técnico e risco operacional quando você tem de implantar essas alterações a uma aplicação que já está sendo executada na produção. O Amazon API Gateway permite que *you* pegue atalhos e entregue mais rapidamente, porque o serviço já os implementou para você.

O tempo de vida global de uma aplicação pode ser desconhecido ou sabidamente curto. Criar um business case para uma nova aplicação multicamadas pode ser difícil por esses motivos. Pode ficar mais fácil quando seu ponto inicial já inclui os recursos gerenciados fornecidos pelo Amazon API Gateway e você só começa a incorrer custos de infraestrutura depois de os APIs começarem a receber solicitações. Para obter mais informações, consulte [Definição de preço do Amazon API Gateway](#).⁶

Iteração rápida, agilidade mantida

Com novas aplicações, a base de usuários ainda pode ser mal definida (tamanho, padrões de uso, etc.). A camada lógica deve manter a agilidade enquanto a base de usuários toma forma. Sua aplicação e seus negócios devem ser capazes de mudar e acomodar a mudança de expectativas dos primeiros adotantes.

O Amazon API Gateway reduz o número de ciclos de desenvolvimento necessários para levar uma API da concepção à implantação. O Amazon API Gateway fornece a capacidade de criar [Mock Integrations](#)⁷, que lhe permitem gerar respostas de API diretamente do API Gateway com o qual as aplicações clientes podem desenvolver enquanto, paralelamente, a lógica completa de back-end é desenvolvida. Esse benefício se aplica não só na primeira implantação da API, mas também depois de a empresa decidir que a aplicação (e a API existente) precisam mudar rapidamente em resposta aos seus usuários. O API Gateway e o AWS Lambda habilitam versionamento, de forma que a funcionalidade e as dependências de cliente existentes possam continuar intactas, enquanto novas funcionalidades são liberadas como uma versão independente de API/função.

Segurança

Implementar a camada lógica de uma aplicação web pública de três camadas como um serviço da web eleva imediatamente o tópico de segurança. A aplicação precisa garantir que somente clientes autorizados tenham acesso à camada lógica (que é exposta sobre a rede). O Amazon API Gateway aborda o tópico de segurança por formas que podem lhe dar a certeza de que o back-end é seguro. Para controle de acesso, não confie em fornecer às aplicações do cliente strings de chave de API estáticas; elas podem ser extraídas dos clientes e usadas em outros lugares. Você pode aproveitar diversas formas como o Amazon API Gateway contribui para proteger a camada lógica:

- Todas as solicitações das suas APIs podem ser feitas via HTTPS, de forma a habilitar a criptografia em trânsito.
- As funções do seu AWS Lambda podem restringir o acesso, de forma que exista um relacionamento de confiança somente entre uma API em particular dentro do Amazon API Gateway e uma função particular no AWS Lambda. Não haverá outra forma de invocar essa função do Lambda, exceto ao usar a API pela qual você optou a exposição dela.

- O Amazon API Gateway permite gerar SDKs de clientes a serem integrados com suas APIs. Esse SDK também gerencia a assinatura de solicitações quando as APIs exigem autenticação. Essas credenciais da API usadas no lado do cliente para autenticação são diretamente passadas para a função do AWS Lambda – onde poderá ocorrer mais autenticação dentro do código que você tem e grava, se necessário.
- Cada combinação de recurso/método que você cria como parte da API recebe seu próprio Nome de Recurso da Amazon (ARN), que pode ser verificado [nas políticas do AWS Identity and Access Management \(IAM\)](#)⁸.
 - Isso significa que suas APIs são tratadas como cidadãs de primeira classe, junto com outras APIs de propriedade da AWS. As políticas do IAM podem ser granularizadas; elas podem fazer referência a recursos/métodos específicos de uma API criada usando o Amazon API Gateway.
 - O acesso à API é regido pelas políticas do IAM que você cria fora do contexto do código de aplicação. Isso significa que você não precisa escrever nenhum código para ser conscientizado ou aplicar esses níveis de acesso. O código não pode conter bugs nem ser explorado, caso não exista.
 - Autorizar os clientes a usarem a autorização [AWS Signature versão 4 \(SigV4\)](#)⁹ e as políticas do IAM para acesso à API permite que essas mesmas credenciais restrinjam ou permitam acesso a outros serviços e recursos da AWS conforme o necessário (por exemplo, buckets do Amazon S3 ou tabelas do Amazon DynamoDB).

AWS Lambda

No seu cerne, o AWS Lambda permite código arbitrário gravado em qualquer uma das linguagens compatíveis (Node, baseada em JVM e Python, desde novembro de 2015) para ser disparado em resposta a um evento. Esse evento pode ser um dos vários disparos programáticos que a AWS disponibiliza, chamados de **origem do evento** ([veja as origens do evento atualmente compatíveis aqui](#)¹⁰). Vários casos de uso populares para o AWS Lambda giram em torno de workflows de processamento de dados orientados por eventos, como processamento de arquivos armazenados no [Amazon Simple Storage Service \(Amazon S3\)](#)¹¹ ou streaming de registros de dados do [Amazon Kinesis](#)¹².

Quando usada em conjunto com o Amazon API Gateway, uma função do AWS Lambda pode existir dentro do contexto de um serviço da web típico, e pode ser disparada diretamente por uma solicitação HTTPS. O Amazon API Gateway age como porta da frente da sua camada lógica, mas agora você precisa executar a lógica por trás dessas APIs. É aí onde entra o AWS Lambda.

Sua lógica de negócios vai aqui

O AWS Lambda permite escrever funções de código, chamadas **handlers**, que serão executadas quando disparadas por um evento. Por exemplo, você pode gravar um handler que será disparado quando ocorrer um evento como uma solicitação de HTTPS na sua API. O Lambda lhe permite criar handlers modulares no seu nível de granularidade escolhido (um por API e um por método de API) que pode ser atualizado, invocado e alterado de forma independente. O handler então fica livre para entrar em contato com outras dependências que tem (como outras funções que você carregou com seu código, bibliotecas, binários nativos ou até mesmo serviços da web externos). O Lambda lhe permite criar um pacote com todas as dependências necessárias na sua definição de função durante a criação. Ao criar sua função, você especifica qual método dentro do pacote de implantação agirá como handler da solicitação. Você tem a liberdade de reutilizar o mesmo pacote de implantação para várias definições de função do Lambda; cada função do Lambda pode ter um handler único dentro do mesmo pacote de implantação. No padrão de arquitetura multicamadas sem servidor, cada uma das APIs criadas no Amazon API Gateway se integrará a uma função do Lambda (e o handler dentro dela) que executa a lógica de negócios necessária.

Integração com a Amazon VPC

O AWS Lambda, o cerne da sua camada lógica, será o componente diretamente integrado à camada de dados. Como a camada de dados muitas vezes contém informações confidenciais de negócios ou usuário, a camada de dados deve ser bem protegida. Para os serviços da AWS com os quais você pode fazer a integração de uma função do Lambda, você pode gerenciar o controle de acesso usando políticas da IAM. Entre esses serviços estão Amazon S3, Amazon DynamoDB, Amazon Kinesis, Amazon Simple Queue Service (Amazon SQS), Amazon Simple Notification Service (Amazon SNS), outras funções do AWS Lambda e muito mais. No entanto, você pode ter um componente que rege seu próprio controle de acesso, como um banco de dados relacional. Com componentes como esse, você poderia conquistar mais segurança ao implementá-los dentro de um ambiente de rede privada – uma [Amazon Virtual Private Cloud \(Amazon VPC\)](#)¹³.

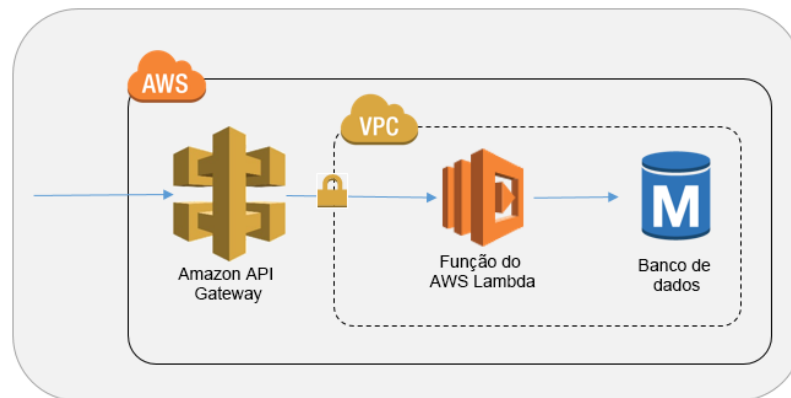


Figura 2: Padrão arquitetônico usando uma VPC

O uso de uma VPC significa que os bancos de dados e outras mídias de armazenamento das quais sua lógica de negócios depende podem ficar inacessíveis pela Internet. A VPC também garante que a *única* forma de interagir com os seus dados pela Internet será por APIs que você definiu e função do código do Lambda que você escreveu.

Segurança

Para executar uma função do Lambda, ela deve ser disparada por um evento ou serviço com permissão para tal via política do IAM. É possível criar uma função do Lambda que não possa ser executada *de jeito nenhum* a menos que invocada por uma solicitação do API Gateway definida por você. Seu código só será processado como parte do seu caso de uso válido, definido pela API criada.

Cada função do Lambda em si assume um papel no IAM, uma capacidade que precisa ser concedida por um relacionamento de confiança do IAM. O papel do IAM define os outros serviços/recursos da AWS com os quais sua função do Lambda poderá interagir (como uma tabela do Amazon DynamoDB ou um bucket do Amazon S3). Os serviços aos quais sua função tem acesso serão definidos e controlados de fora da função em si. É algo sutil, mas poderoso. O código escrito por você pode ficar livre do armazenamento ou da recuperação de credenciais da AWS: isso significa que você não precisa usar hardcoding nas teclas da API nem escrever código para recuperá-los e armazená-los na memória. A habilitação da função do Lambda para acessar os serviços aos quais tem permissão, conforme definido pela função do IAM, é gerenciada por você e pelo serviço em si.

A camada de dados

Ao usar o AWS Lambda como sua camada lógica, você ganha uma série de opções de armazenamento físico de dados para sua camada de dados. Essas opções se encaixam em duas amplas categorias: Armazenamentos de dados hospedados em Amazon VPC e armazenamentos de dados habilitados por IAM. O AWS Lambda tem a capacidade de se integrar com segurança aos dois.

Armazenamentos de dados hospedados em Amazon VPC

A integração do AWS Lambda com a Amazon VPC permite que as funções se integrem com diversas tecnologias de armazenamento físico de dados de forma privada e segura.

- [Amazon RDS¹⁴](#)

Use qualquer um dos mecanismos que o Amazon Relational Database Service (Amazon RDS) disponibiliza. Conecte-se ao Amazon RDS diretamente pelo código escrito no Lambda, da mesma forma como faria fora do Lambda, mas com a vantagem da integração simples com AWS Key Management Service (AWS KMS) para criptografia de credencial ao banco de dados.

- [Amazon ElastiCache¹⁵](#)

Integre suas funções do Lambda com um cache de memória gerenciada para incrementar o desempenho da aplicação.

- [Amazon RedShift¹⁶](#)

Você pode construir funções que fazem uma consulta segura a um data warehouse corporativo com o objetivo de criar relatórios e painéis ou recuperar resultados de consultas ad-hoc.

- Serviço da web privado hospedado por [Amazon Elastic Compute Cloud \(Amazon EC2\)¹⁷](#)

Você pode ter aplicações existentes executadas como serviço da web privadamente dentro de uma VPC. Faça solicitações de HTTP sobre a rede de VPC com lógica privada a partir de uma função do Lambda.

Armazenamentos de dados habilitados por IAM

Como o AWS Lambda é integrado ao IAM, ele pode usar o IAM para garantir a integração com qualquer serviço da AWS que possa ser aproveitado diretamente usando as APIs da AWS.

- [Amazon DynamoDB¹⁸](#)

O Amazon DynamoDB é o banco de dados NoSQL de escalabilidade infinita da AWS. Pense no Amazon DynamoDB quando você quiser recuperar registros de dados (400 KB ou menos, no momento de redação deste documento) com desempenho de milissegundo de único dígito, independentemente da escala. Usando o controle de acesso granularizado do Amazon DynamoDB, suas funções do Lambda podem seguir as melhores práticas de menos privilégios ao consultar dados específicos no DynamoDB.

- [Amazon S3¹⁹](#)

O Amazon Simple Storage Service (Amazon S3) fornece armazenamento de objetos na escala da Internet. O Amazon S3 é projetado para durabilidade de 99,999999999% dos objetos, então pense em usá-lo quando sua aplicação precisar de um armazenamento barato e resiliente. Além disso, como o Amazon S3 é projetado para até 99,99% de disponibilidade dos objetos ao longo de um ano, pense em usá-lo quando sua aplicação exigir armazenamento altamente disponível. Objetos armazenados no Amazon S3 (arquivos, imagens, logs, qualquer dado binário) podem ser acessados diretamente via HTTP. As funções do Lambda podem se comunicar com segurança com o Amazon S3 via endpoints virtuais privados, e os dados dentro do S3 podem ser restringidos somente à política do IAM associada à função do Lambda.

- [Amazon Elasticsearch Service²⁰](#)

O Amazon Elasticsearch Service (Amazon ES) é a versão gerenciada do conhecido mecanismo de pesquisa e analítica, o Elasticsearch. O Amazon ES fornece o provisionamento gerenciado de clusters, detecção de falha e substituição de nós; você pode restringir o acesso à API do Amazon ES usando políticas de IAM.

A camada de apresentação

O Amazon API Gateway abre uma variedade de possibilidades de camada de apresentação. Uma API HTTPS acessível por Internet pode ser consumida por qualquer cliente capaz de comunicação HTTPS. A lista a seguir contém exemplos comuns cujo uso você deveria considerar para a camada de apresentação da sua aplicação:

- Aplicação para celular: Além de se integrar a uma lógica de negócios personalizada via Amazon API Gateway e AWS Lambda, você poderia usar o [Amazon Cognito](#)²¹ como mecanismo para criar e gerenciar identidades de usuário.
- Conteúdo de site estático (como arquivos armazenados no Amazon S3): Você pode habilitar as APIs do Amazon API Gateway para serem compatíveis com CORS (cross-origin resource sharing, compartilhamento de recursos de origem cruzada). Com isso, os navegadores da web invocam diretamente suas APIs de dentro das páginas da web estáticas.
- Qualquer outro dispositivo cliente habilitado para HTTPS: Vários dispositivos conectados são capazes de se comunicar via HTTPS. Não há nada exclusivo nem proprietário em relação à forma como os clientes se comunicam com as APIs criadas usando Amazon API Gateway; é HTTPS puro. Não são necessários softwares nem licenças específicos de cliente.

Amostra dos padrões de arquitetura

Você pode implementar os seguintes padrões populares de arquitetura usando o Amazon API Gateway e o AWS Lambda como a cola que forma sua camada lógica. Por exemplo, usaremos somente os serviços da AWS que não exigem que os usuários gerenciem a própria infraestrutura.

Back-end móvel

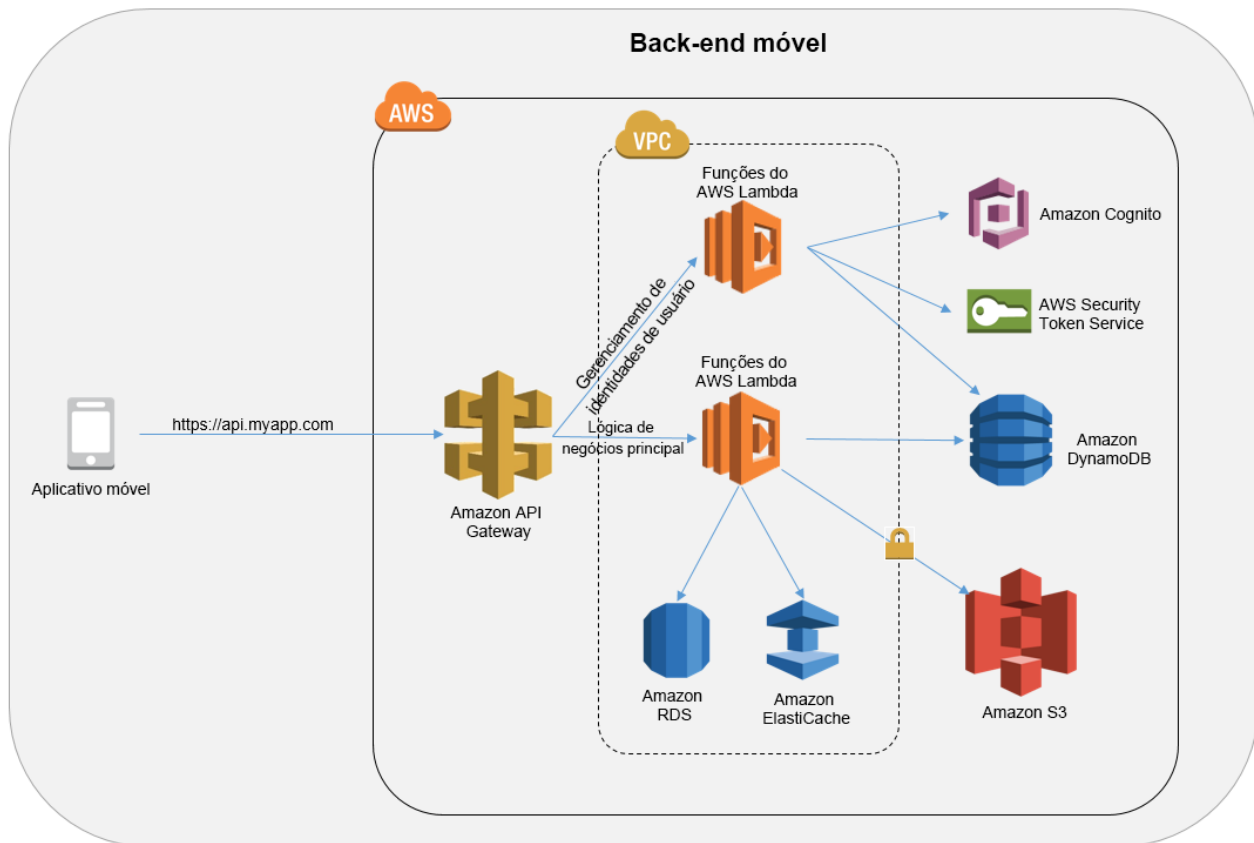


Figura 3: Padrão de arquitetura para um back-end móvel

- **Camada de apresentação:** Uma aplicação móvel executada no smartphone de cada usuário.
- **Camada lógica:** Amazon API Gateway e AWS Lambda. A camada lógica tem distribuição global pelo Amazon CloudFront, criado como parte de cada API do Amazon API Gateway. Um conjunto de funções do Lambda pode ser específico de gerenciamento e autenticação de identidade do usuário/dispositivo e gerenciado pelo Amazon Cognito, que proporciona integração com IAM para credenciais temporárias de acesso do usuário, bem como com conhecidos provedores de identidade terceirizados. Outras funções do Lambda são capazes de definir a lógica de negócios central do back-end móvel.

- **Camada de dados:** Os vários serviços de armazenamento físico de dados podem ser aproveitados conforme o necessário; as opções já foram apresentadas aqui neste documento.

Site hospedado do Amazon S3



Figura 4: Padrão arquitetural para um site estático hospedado no Amazon S3

- **Camada de apresentação:** Conteúdo de site estático armazenado no Amazon S3 e distribuído pelo Amazon CloudFront. A hospedagem de conteúdo de site estático no Amazon S3 é uma alternativa com boa relação custo-benefício ao armazenamento de conteúdo em uma infraestrutura baseada em servidor. No entanto, para um site conter recursos avançados, o conteúdo estático precisa se integrar a um back-end dinâmico.
- **Camada lógica:** Amazon API Gateway e AWS Lambda. O conteúdo web estático hospedado em Amazon S3 pode se integrar diretamente com o Amazon API Gateway, que pode ser compatível com CORS.

- **Camada de dados:** Os vários serviços de armazenamento físico de dados podem ser aproveitados conforme o necessário. Essas opções já foram apresentadas aqui neste documento.

Ambiente de microsserviços

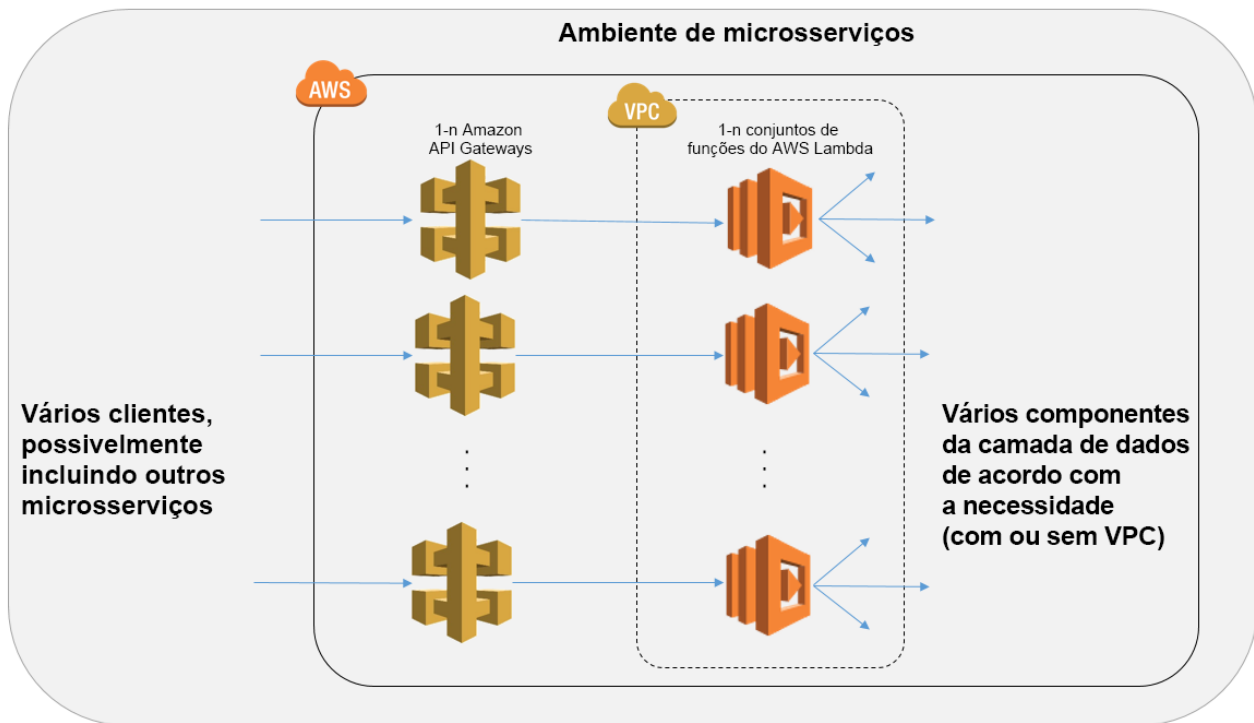


Figura 5: Padrão arquitetônico para um ambiente de microsserviços

O padrão arquitetônico de **Microsserviços** não está ligado à típica arquitetura de três camadas abrangida neste whitepaper. Na arquitetura de microsserviços, existe um desacoplamento maciço dos componentes de software, de forma que os benefícios da arquitetura multicamadas sejam ampliados. Uma API criada com Amazon API Gateway, e funções subsequentemente executadas pelo AWS Lambda, é tudo que você precisa para construir um microsserviço. Sua equipe tem a liberdade de usar esses serviços para desacoplar e fragmentar seu ambiente até o nível de granularidade desejado.

No geral, o ambiente de microsserviços pode introduzir as seguintes dificuldades: custos indiretos repetidos pela criação de cada novo microsserviço, problemas com a otimização da densidade/utilização do servidor, complexidade da execução de múltiplas versões de múltiplos microsserviços simultaneamente e proliferação de requisitos de código do lado do cliente para se integrar a vários serviços separados.

No entanto, ao criar microsserviços usando o padrão sem servidor da AWS, esses problemas ficam mais fáceis de se resolver e, em alguns casos, simplesmente desaparecem. O padrão de microsserviços da AWS diminui a barreira para criação de cada microsserviço subsequente (o Amazon API Gateway leva em conta até a clonagem de APIs existentes). A otimização da utilização do servidor não é mais relevante com este padrão. Tanto o API Gateway quanto o Lambda permitem recursos de versionamento mais simples. Por fim, o Amazon API Gateway fornece SDKs de clientes gerados programaticamente em diversas linguagens conhecidas, de forma a reduzir os custos indiretos de integração.

Conclusão

O padrão da arquitetura multicamadas incentiva as melhores práticas da criação de componentes de aplicações fáceis de manter, desacoplados e escaláveis. Quando você cria uma camada lógica na qual a integração ocorre via Amazon API Gateway e a computação ocorre dentro do AWS Lambda, você está prestes a se dar conta dessas metas, ao mesmo tempo reduzindo a quantidade de esforço para alcançá-las. Juntos, esses serviços fornecem um front-end de API HTTPS para seus clientes e um ambiente seguro dentro da VPC para executar a lógica de negócios. Isso lhe permite aproveitar os vários cenários populares nos quais é possível usar esses serviços gerenciados em vez de gerenciar sozinho a infraestrutura típica baseada em servidor.

Colaboradores

As seguintes organizações e pessoas contribuíram para este documento:

Andrew Baird, arquiteto de soluções da AWS

Stefano Buliani, gerente sênior de produto, Tecnologia, serviços móveis da AWS

Vyom Nagrani, gerente sênior de produto, serviços móveis da AWS

Ajay Nair, gerente sênior de produto, serviços móveis da AWS

Observações

- ¹ <http://aws.amazon.com/api-gateway/>
- ² <http://aws.amazon.com/lambda/>
- ³ <https://aws.amazon.com/vpc/>
- ⁴ <https://aws.amazon.com/cloudfront/>
- ⁵ https://do.awsstatic.com/whitepapers/DDoS_White_Paper_June2015.pdf
- ⁶ <https://aws.amazon.com/api-gateway/pricing/>
- ⁷ <http://docs.aws.amazon.com/apigateway/latest/developerguide/how-to-mock-integration.html>
- ⁸ <http://aws.amazon.com/iam/>
- ⁹ <http://docs.aws.amazon.com/general/latest/gr/signature-version-4.html>
- ¹⁰ <http://docs.aws.amazon.com/lambda/latest/dg/intro-core-components.html#intro-core-components-event-sources>
- ¹¹ <https://aws.amazon.com/s3/>
- ¹² <https://aws.amazon.com/kinesis/>
- ¹³ <https://aws.amazon.com/vpc/>
- ¹⁴ <https://aws.amazon.com/rds/>
- ¹⁵ <https://aws.amazon.com/elasticache/>
- ¹⁶ <https://aws.amazon.com/redshift/>
- ¹⁷ <https://aws.amazon.com/ec2/>
- ¹⁸ <https://aws.amazon.com/dynamodb/>
- ¹⁹ <https://aws.amazon.com/s3/storage-classes/>
- ²⁰ <https://aws.amazon.com/elasticsearch-service/>
- ²¹ <https://aws.amazon.com/cognito/>