Getting Started with AWS Lambda and the Serverless Cloud

Dr. Tim Wagner
General Manager, AWS Lambda and Amazon API Gateway

AWS Tokyo Summit, June 2 , 2016

amazon web services

サーバレスコンピューティングとは？

- VM
  - スケーリングの単位としてのマシン
  - ハードウェアを抽象化
- コンテナ
  - スケーリングの単位としてのアプリ
  - OSを抽象化
- サーバーレス
  - スケーリングの単位として機能
  - 言語ランタイムを抽象化

EC2

ECS

AWS Lambda

どのようにして選択するか？

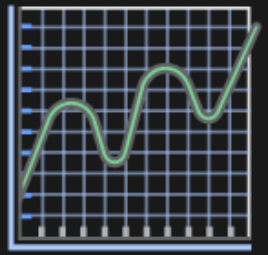
- VM
  - 「マシン、ストレージ、ネットワーキング、OSを構成したい」 **EC2**
- コンテナ
  - 「サーバーを実行し、アプリケーションを構成し、スケーリングを制御したい」 **ECS**
- サーバーレス
  - 「コードを必要なときに実行したい」 **AWS Lambda**

アジェンダ

概要
ユースケース
新しいサービスとベストプラクティス
デモ

AWS Lambdaのメリット

サーバーの管理
が不要



継続的な
スケーリング

アイドル時間は
課金されない：
コールドサーバーはない
（経理担当者が喜ぶ）

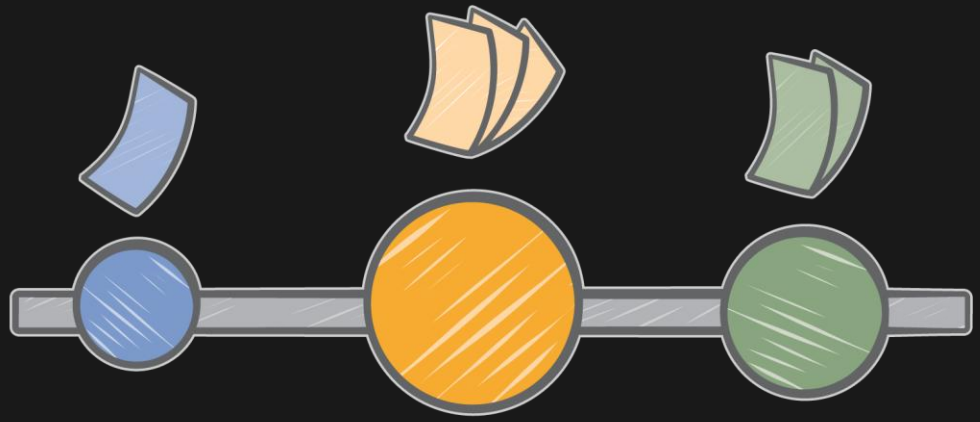
Democratized Scale

クラウドはスーパーコンピュータ
サーバーレスでプログラムできる

リクエストごとに課金

- 100ミリ秒ごとに0.21セントでコンピュート時間を購入
- リクエストの料金は0.2セント
- 時間ごと、日ごと、または月ごとの最低料金なし
- デバイスごとの料金なし

アイドル時間への課金なし！



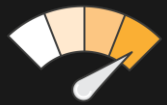
無料利用枠

毎月100万件のリクエストと400,000GBのコンピュートが無料

AWS Lambdaの使用

BYOC（Bring Your Own Code）

- Node.js、Java、Python
- カスタムライブラリも利用可能（ネイティブライブラリを含む）



シンプルなリソースモデル

- 128MB～1.5GBのメモリの選択が可能
- それに比例したCPUとネットワークの割り当て
- 実際の利用状況をレポート

柔軟な使用法

- イベントの呼び出しまたは送信
- 他のAWSサービスとの統合
- サーバーレスエコシステム全体の構築

柔軟な認可

- VPCを含め、リソースへのアクセスをセキュアに付与
- 関数を誰が呼び出せるのかを細かく制御

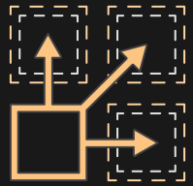
AWS Lambdaの使用

プログラミングモデル

- AWS SDK（PythonおよびNode.js）
- Lambdaは"Webサーバー"
- プロセス、スレッド、/tmp、ソケットの標準使用

ステートレス

- Amazon DynamoDB、S3、ElastiCacheを使ってデータを永続化
- インフラストラクチャへの親和性がない（"マシンにはログイン"できない）



オーサリング機能

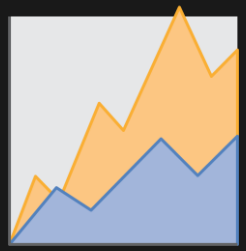
- コンソールのWYSIWYGエディタを使って直接オーサリング
- コードを.zipとしてパッケージ化し、LambdaまたはS3にアップロード
- EclipseとVisual Studioのプラグイン
- コマンドラインツール

モニタリングとロギング

- リクエスト、エラー、レイテンシー、スロットリングの組み込みメトリック
- Amazon CloudWatch Logsの組み込み

ユースケース

Lambda: ユースケース



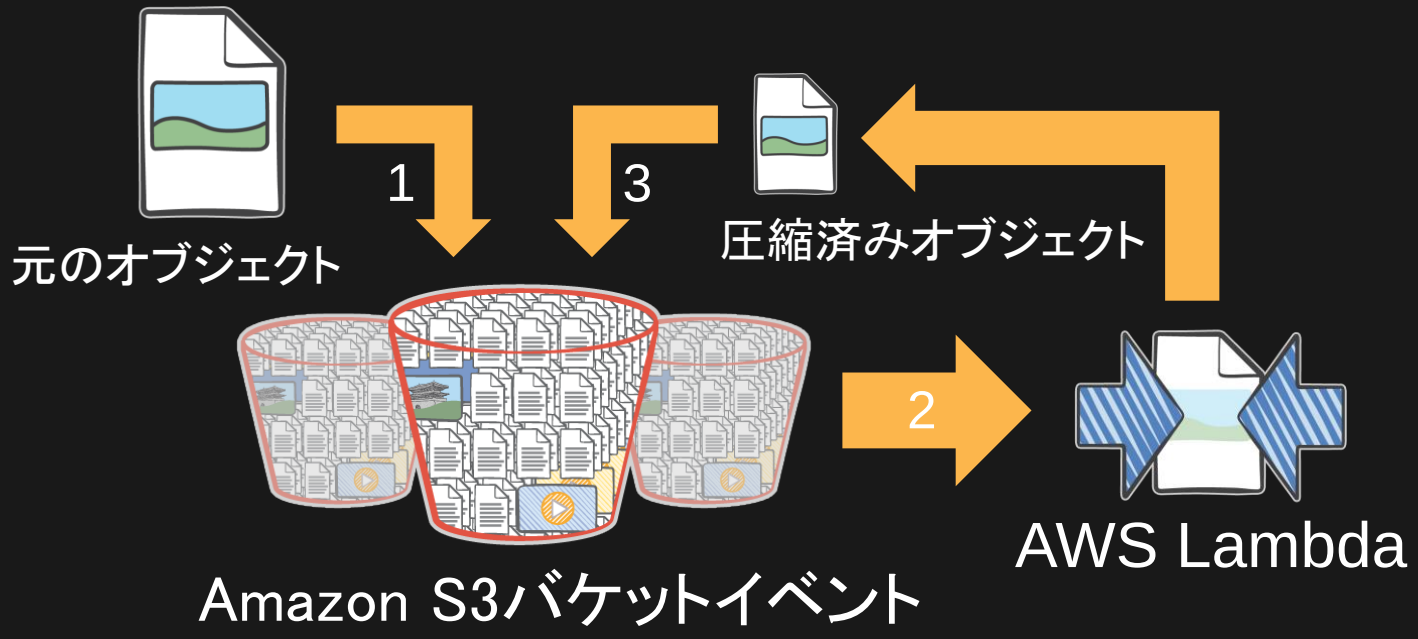
データ処理

バックエンド

サーバーレス
アプリケーション
エコシステム

ユースケース：データ処理

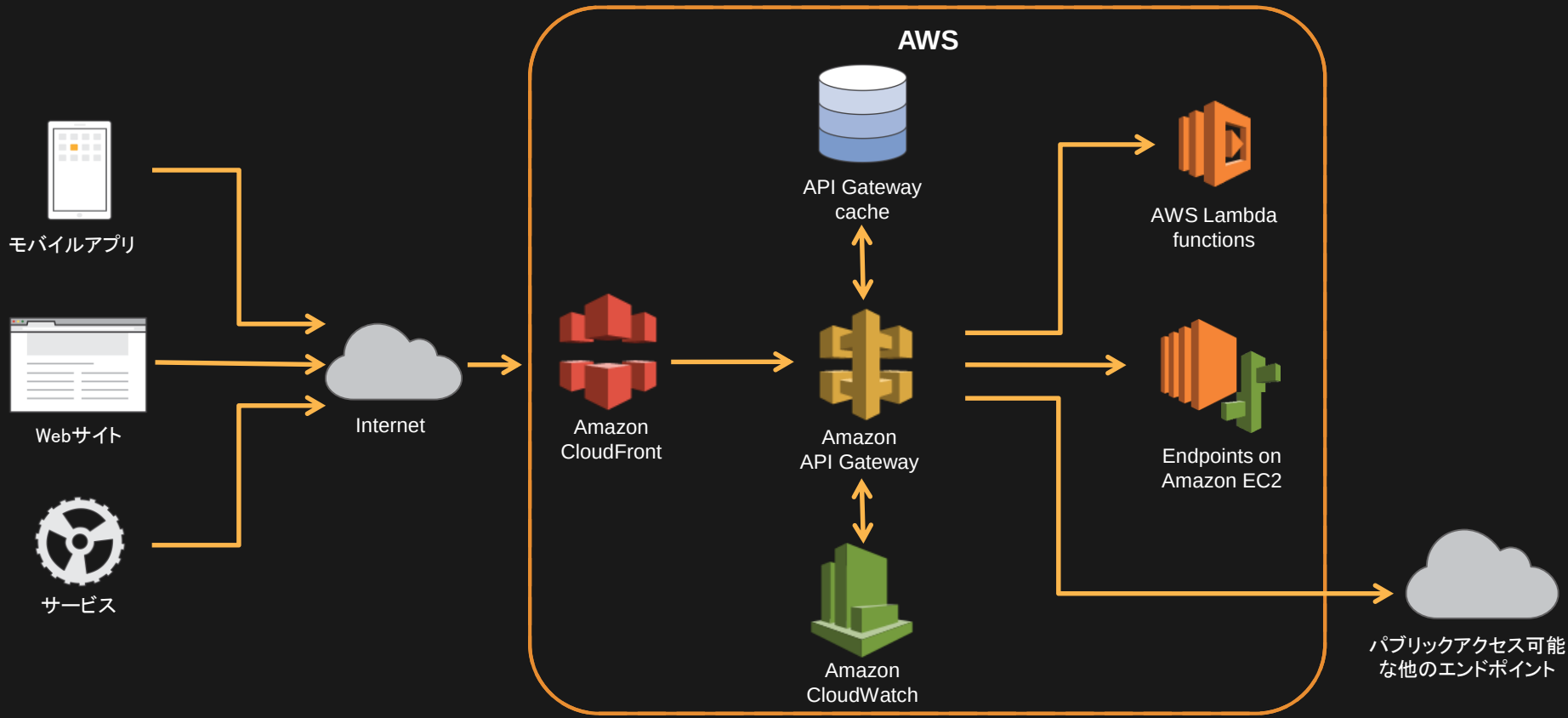
例：Amazon S3バケットトリガー



モバイルアプリやIoTのためのスケーラブルなバックエンド

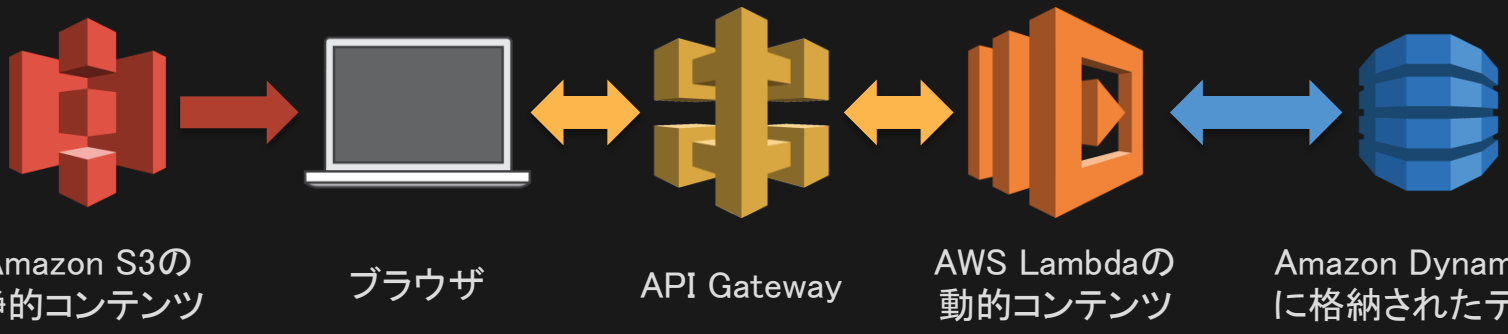
1. どちらかを選択:
    a. モバイルアプリ:AWS Mobile SDK + Amazon Cognito(認可)
    b. IoTデバイス:AWS IoT
2. AWS Lambdaの「モバイルバックエンド」ブループリント
3. データストレージはAmazon DynamoDB

Amazon API Gateway: サーバレス API

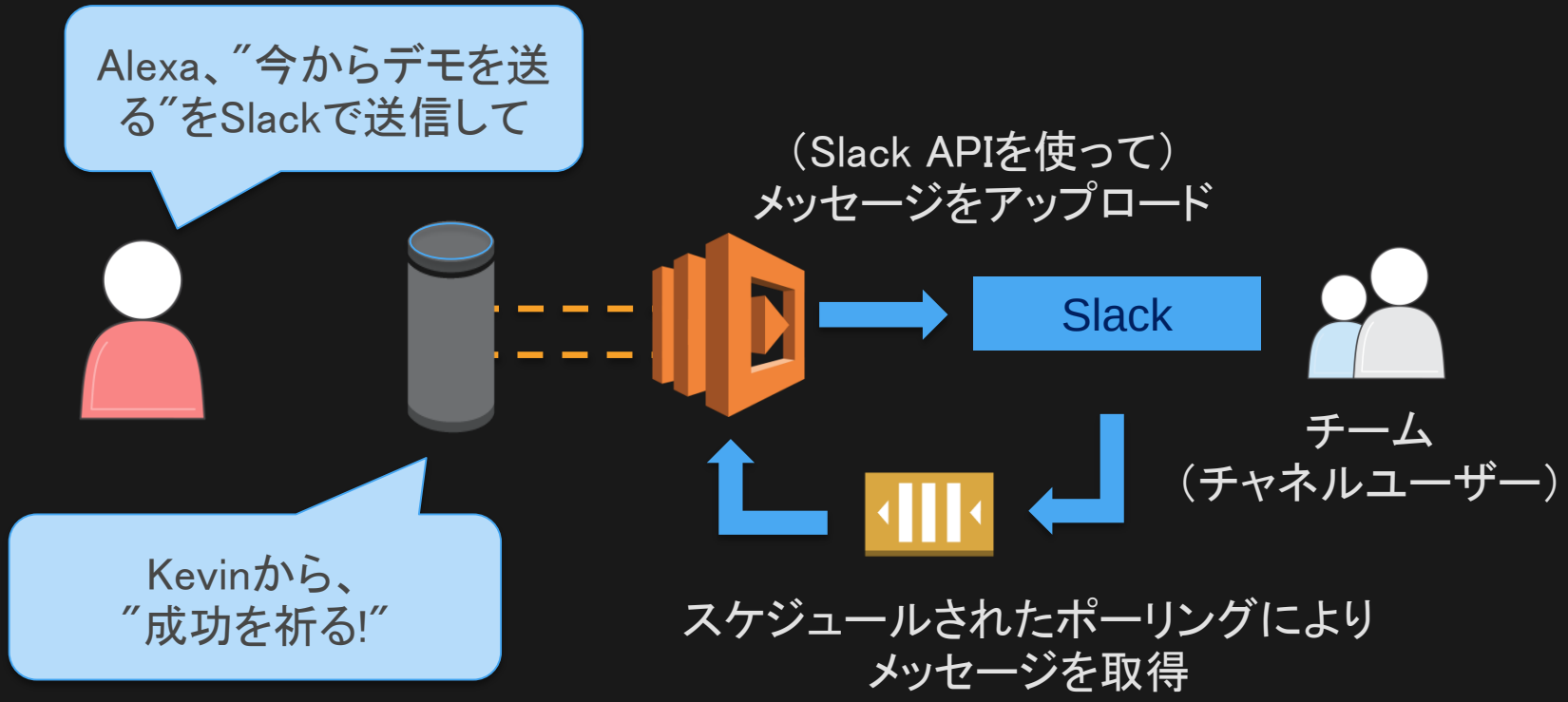


Use case: サーバレスウェブアプリ

1. 静的コンテンツを提供するAmazon S3
2. 動的コンテンツを提供するAWS Lambda
3. HTTPアクセスを提供するAmazon API Gateway
4. NoSQLデータストレージを提供するAmazon DynamoDB



Use case: 新しいアプリケーションエコシステム:
Alexaアプリ + Slack = サーバーレスボット!



AWS Lambda API Gateway Customers

新機能とベストプラクティス

新機能

re:Invent 2015

- Python
- スケジュールされた関数
- 実行時間の延長（5分）
- バージョニング

re:Invent以降

- コードのストレージ上限が5GBから75GBに
- カスタムVPC
- 1分刻みのスケジュール
- 新リージョン：フランクフルト
- Node.js 4.3.2
- 1クリックのCORSセットアップ
- ステージ変数
- カスタム（Lambda）認可
- 組み込みのSwaggerインポート／エクスポート
- AWS CloudFormationがAPI Gatewayとバージョンをサポート

スケジュール機能：入門ガイド

関数を（コールドスタートではなく）ウォーム状態に保つにはどうするか？

スケジュールする！

（SQSのように）キューをポーリングするにはどうするか？

関数がキューを読み込むようにスケジュールする。

タイマーを増やすにはどうするか？

スケジュールされた関数に他の関数を非同期で呼び出させる。

1分よりも細かい単位でスケジュールするにはどうするか？

スケジュールされた関数でバックグラウンドタイマーを実行する。

関数のバージョニング：入門ガイド

変更可能な構成情報を取得するにはどうするか？

関数の初期化時に（DynamoDBなどから）読み込む。

構成を関数に取り込み、パブリッシュされたコードから呼び出す。

AWS Lambdaで「ロールバック」するにはどうするか？

エイリアスを使って、エイリアスが指しているものに切り替える

（API GatewayやCloudFormationを集約的に追加）。

Blue/Greenデプロイメントはどのように行うか？

AWS LambdaはFleetデプロイに対処するが、トラフィックをシェーピングしたい場合は2つ目の「交通整理」関数を手前に追加する。

クライアント／デバイスを古いバージョンに固定するにはどうするか？

そのバージョンのARNを直接指定する。

AWS LambdaのVPCの基礎

Lambda関数はすべて、「常」にVPC内で実行される
- セキュリティを「有効」にする必要はない – 常に有効

Lambda関数にVPC内のリソースへのアクセスを付与することも可能
- 方法：VPCのサブネットIDとセキュリティグループIDを関数の設定に追加
- 一般的な用途：RDB、ElastiCache、プライベートEC2エンドポイント
- ピアリングされたVPC、VPNエンドポイント、プライベートS3エンドポイントへのアクセスを許可

VPCにアクセスするように設定された関数はインターネットにアクセスできない…
- マネージドNATを使用するか、VPC内にNATインスタンスが存在しない限り
- …"Auto-assign Public IP"を有効にしたとしても
- …インターネットゲートウェイをVPC内でセットアップしたとしても
- …セキュリティグループがアウトバウンドトラフィックをすべて許可したとしても

AWS LambdaのVPCのベストプラクティス

VPCは必須でない – *必要でない限り使用しない。*

LambdaのVPC機能によって使用されるENIは各自のクォータに計上される。

- ピーク時の並列処理レベルに合わせて十分な数を確保する（可能であれば統合する）。
- これらのENIを削除したり名前を変更したりしない！ ☺

これらのENIに対して十分なIPをサブネットで確保する。

各アベイラビリティゾーンでサブネットを少なくとも1つ指定する。

- そうしないと、Lambdaは正常に動作するものの、耐障害性が低下する。

サーバレス マイクロサービスの作成方法

1. AWS CloudFormation
    - AWS Lambda関数
    - イベントハンドラ（Amazon S3、Amazon DynamoDB）
    - API（Amazon API Gateway）
2. オープンソースフレームワーク（Serverless.com）
3. Flourish
    - サーバーレスのアプリケーションモデル
    - AWSが支援しているGitHubのオープンソースプロジェクト

Coming Soon!

DEMO

Cloud Formationを使ったマイクロサービスのデプロイメント

サーバーレスコンピューティングのマニフェスト

関数はデプロイメントとスケーリングの単位。

プログラミングモデルでは、マシン、VM、コンテナは見えない。

永続的なストレージがあちこちにある。

リクエストごとのスケーリング。キャパシティの過小または過剰なプロビジョニングは不可能。

アイドル時間は課金されない（コールドサーバー／コンテナやそれらのコストは不要）。

関数はどこでも実行できるため、暗黙的な耐障害性がある。

BYOC（Bring Your Own Code）。

メトリックとロギングは普遍的な権利。

Join the serverless revolution!

0. プロダクトマネージャー、ビジネスアナリスト？

aws.amazon.com/lambdaでプロダクトの詳細と導入事例をチェック

amazon web services

1. 開発者？
AWS Lambdaコンソールで関数を作成し、実行
（呼び出しは100万件まで無料！）

amazon web services

2. これであなたもLambda関数のエキスパート! イベントソースやHTTPエンドポイントを追加

amazon web services

3.

モバイル、音声、IoTバックエンドを数行のコードで構築

amazon web services

Lambdaとサーバーレスに関する最新情報はツイッターで:

t: @timallenwagner

Q & A

Links for AWS Lambda and
Amazon API Gateway:

aws.amazon.com/blogs/compute
aws.amazon.com/lambda
AWS Lambda forum

amazon web services