AWS Serverless, AWS Lambda

What is a serverless architecture

Serverless Services in AWS

What is AWS Lambda

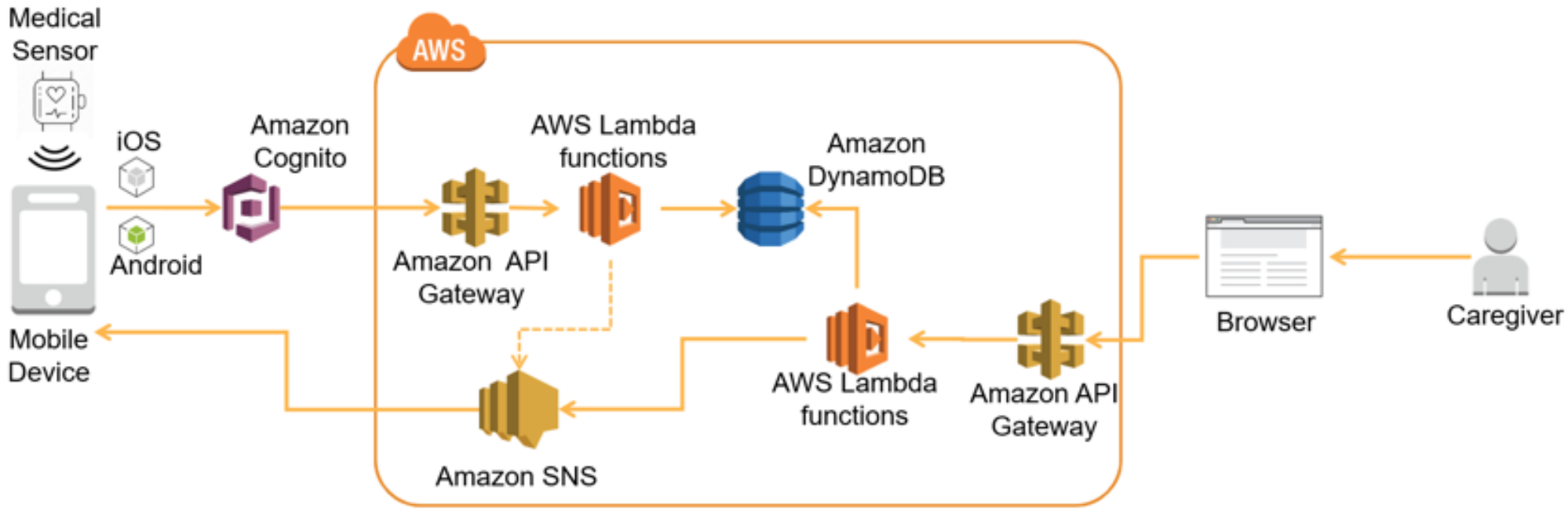
What is a serverless architecture

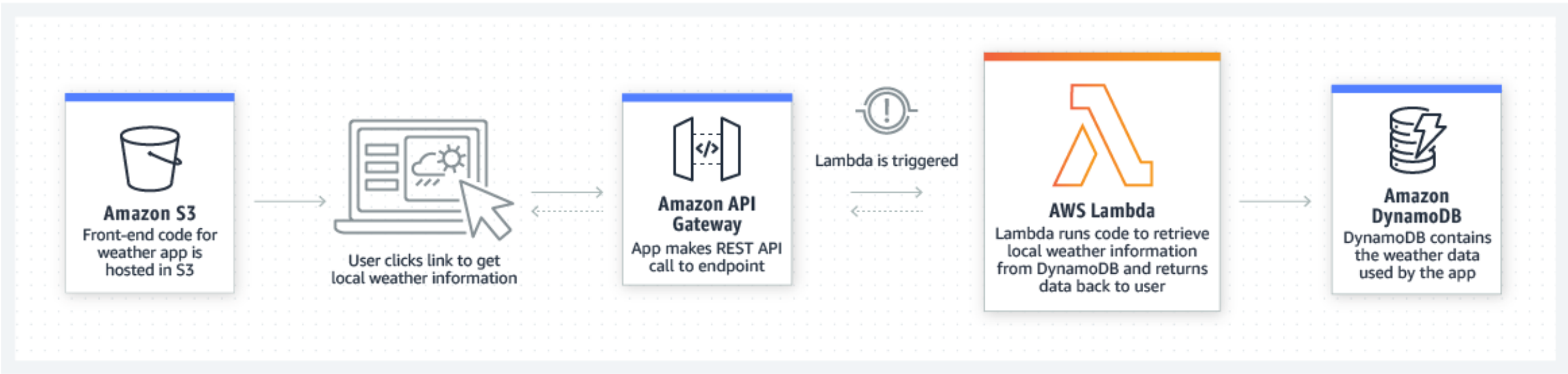
- Is a way to build and run applications and services without having to manage infrastructure and scale, maintain servers.
- Your application still runs on servers, but all the server management is done by AWS
- AWS offers technologies for running code, managing data, and integrating applications, all without managing servers
- It feature automatic scaling, built-in high availability, and a pay-for-use billing model to increase agility and optimize costs.
- Serverless applications start with AWS Lambda, natively integrated with over 200 AWS services and software as a service (SaaS) applications
- Serverless **does not mean there are no servers**. it means you just don’t manage / provision / see them.

Serverless Services in AWS

- AWS Lambda
- AWS Fargate
- DynamoDB
- AWS Cognito
- AWS API Gateway
- Amazon S3
- AWS SNS & SQS
- AWS Kinesis Data Firehose
- Aurora Serverless
- Step Functions

Mobile Diagnostics




Amazon S3
Front-end code for weather app is hosted in S3
User clicks link to get local weather information
Amazon API Gateway
App makes REST API call to endpoint
Lambda is triggered
AWS Lambda
Lambda runs code to retrieve local weather information from DynamoDB and returns data back to user
Amazon DynamoDB
DynamoDB contains the weather data used by the app

What is AWS Lambda

- Lambda is a compute service that lets you run code without provisioning or managing servers.
- Lambda runs your code on a high-availability compute infrastructure.
- AWS performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, code monitoring and logging
- you can run code for virtually any type of application or backend service
- AWS Lambda runs your code on a high-availability compute infrastructure
- AWS Lambda executes your code only when needed and scales automatically.
- You pay only for the compute time you consume – No charge when your code is not running.

EC2 Instance vs AWS Lambda

- **Virtual Servers** in the Cloud
- Limited by RAM and CPU
- **Continuously running**
- Scaling means: To add or remove servers

- **Virtual functions** – no servers to manage!
- Limited by time - short executions
- **Run on-demand**
- Scaling is automated!

Benefits of AWS Lambda

- Easy Pricing
  - Pay per request and compute time
  - Free tier of 1,000,000 AWS Lambda requests and 400,000 GBs of compute time
- No servers to manage
- Continuous scaling
- Integrated with the many AWS services
- Integrated with many programming languages
- Easy monitoring through AWS CloudWatch
- Easy to get more resources per functions
- Increasing RAM will also improve CPU and network!

AWS Lambda language support

- Node.js
- Python
- Java (Java 8 compatible)
- C# (.NET Core/ Powershell)
- Golang
- Ruby
- Custom Runtime API

AWS Lambda Functions Configuration

- IAM role (execution role)
  - This is the role that AWS Lambda assumes when it executes the Lambda function on your behalf.
- Handler name
  - The handler refers to the method in your code where AWS Lambda begins execution.
  - AWS Lambda passes any event information, which triggered the invocation, as a parameter to the handler method.
- Language Selection
- Write your source code

AWS Lambda Demo

- Create a simple lambda function, test from console, see CloudWatch logs
- Run the lambda using cli
  - aws lambda list-functions --region ap-south-1
  - aws lambda invoke --function-name mylambda --cli-binary-format raw-in-base64-out --payload '{"key1": "value1", "key2": "value2", "key3": "value3" }' --region ap-south-1 response.json
- Test the Lambda function using Load Balancer
- Lambda with CloudWatch Events
- Lambda with S3 event source

Lambda Function – Services it can access

- Lambda functions can access
  - AWS Services running in AWS VPCs (Ex. Redshift , ElastiCache, RDS instances)
  - Non-AWS Services running on EC2 instances in an AWS VPC
- Additional configuration will be required for VPC access (Security group and subnet IDs)
- AWS Lambda runs your function code securely within an internal AWS VPC (not your VPC) by default.
  - This VPC has connectivity to AWS services and the internet.
  - You can configure a lambda function to connect to private subnets (DBs, Cache instances, or Internal services) in your VPC in your accounts
- To enable your Lambda function to access resources inside your VPC:
  - Provide additional VPC-specific configuration information that includes VPC subnet IDs and security group IDs
  - Lambda will then create an ENI for each subnet and security group attached to the Lambda function

Invoking Lambda Functions

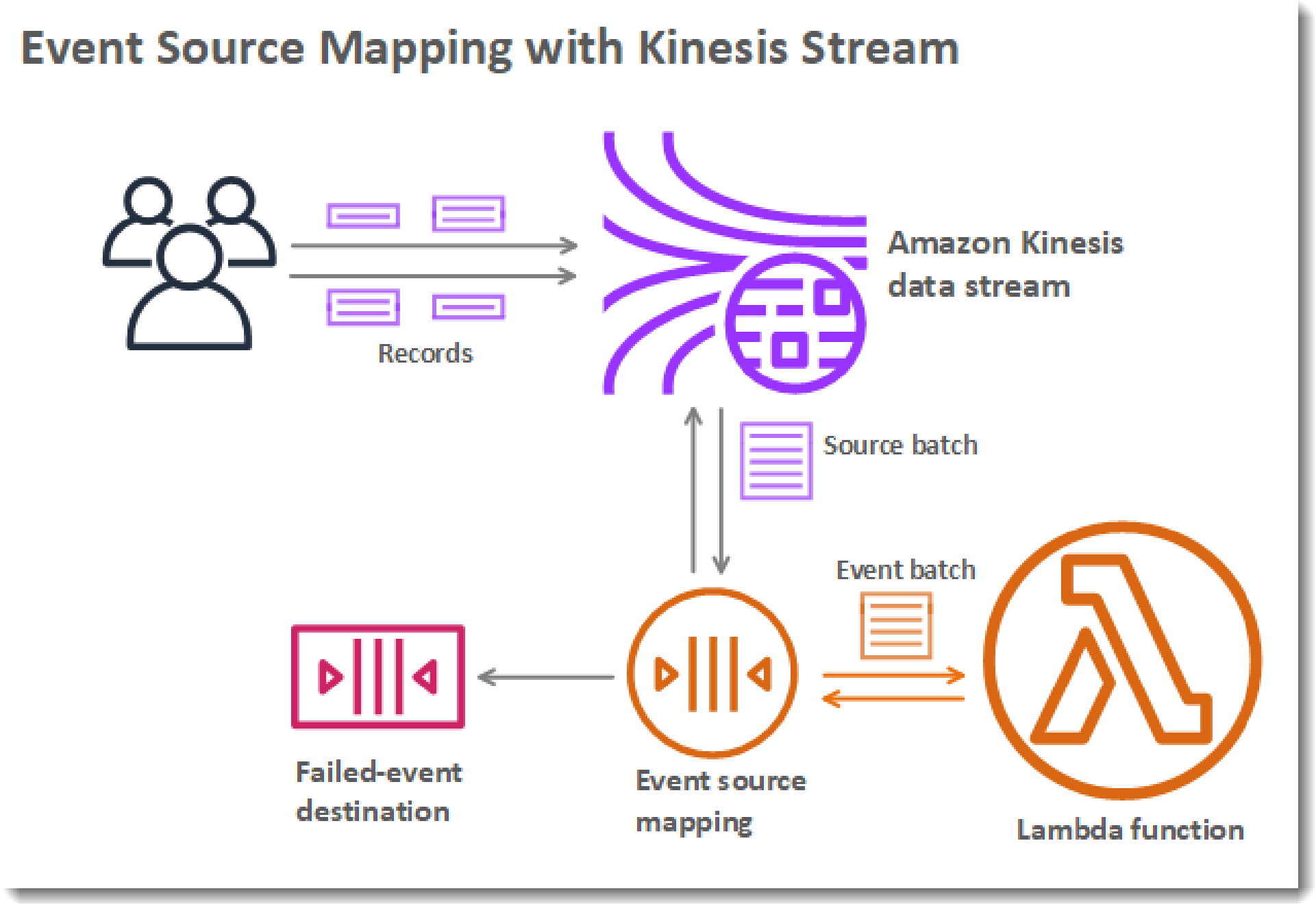
- On-demand Lambda function invocation
  - Amazon API Gateway
  - Load Balancer
  - Web/Mobile application
  - Using SDKs
- Scheduled events
  - Regular, scheduled basis.
  - Cron Jobs
- Event Source Mapping
  - An event source is the AWS service or custom application that publishes events,
  - A Lambda function is the custom code that processes the events.
  - Event sources publish events that cause the Lambda function to be invoked

Supported AWS event sources

- Amazon S3
- Amazon DynamoDB
- Amazon Kinesis Streams
- Amazon Simple Notification Service
- Amazon Simple Email Service
- Amazon Cognito
- AWS CloudFormation
- Amazon CloudWatch Logs
- Amazon CloudWatch Events

AWS CodeCommit
AWS Config
Amazon Alexa
Amazon Lex
Amazon API Gateway
AWS IoT Button
Amazon CloudFront
Amazon Kinesis Firehose
Amazon SQS

Event Source Mapping with Kinesis Stream



Use Cases of AWS Lambda

- You have a photo sharing application. Your application upload photos, and the application stores these user photos in an Amazon S3 bucket.
- Lambda function triggers for each photo upload and resize it for web. Mobile and tab size

AWS Lambda Limits

- Execution
  - Memory allocation: 128 MB – 3008 MB (64 MB increments)
  - Maximum execution time: 900 seconds (15 minutes)
  - Environment variables (4 KB)
  - Concurrency executions: 1000 (can be increased)
- Deployment
  - Lambda function deployment size (compressed .zip): 50 MB
  - Size of uncompressed deployment (code + dependencies): 250 MB
  - Can use the /tmp directory to load other files at startup
  - Size of environment variables: 4 KB

Monitoring and Pricing

- AWS Lambda automatically monitors Lambda functions on your behalf, reporting metrics through Amazon CloudWatch
- Log Captured in CloudWatch in Log Group
- Pricing: https://aws.amazon.com/lambda/pricing/
- Pay per calls:
  - First 1,000,000 requests are free
- Pay per duration: (in increment of 100ms)
  - 400,000 GB-seconds of compute time per month if FREE
  - == 400,000 seconds if function is 1GB RAM
  - == 200,000 seconds if function is 2 GB RAM