~~We don't need no stinking servers~~

Serverless – there are servers

Adrian Pomilio

Sr. UI Architect & Platform Architecture Lead
Mapp Digital / www.mapp.com

Twitter: adrianpomilio

github: adrianpomilio

linkedIn : adrianpomilio

The Story

Requirements are gathered

Development begins

Setup the infrastructure

Deploy happens

I just want to build software

Time to gopher what you know

What we know

What is it that we do very well, what are our strengths?

Code

We write code

Design

We design applications

Resources

We work with databases, APIs, file systems and more

What is “serverless”

Servers

Yes, there are still servers, we just don’t maintain them anymore

Concept

Serverless is not a specific technology

Serverless is about small pieces of functionality

Complexity

While making things simpler complexity creeps in

Serverless Providers

This is not an exhaustive list

Amazon

Lambda functions, api gateway, etc.

Microsoft

Azure Functions, Fabric

Google

Cloud platform

IBM

Bluemix

The common pieces of an AWS serverless application

API Gateway

Lambda

S3

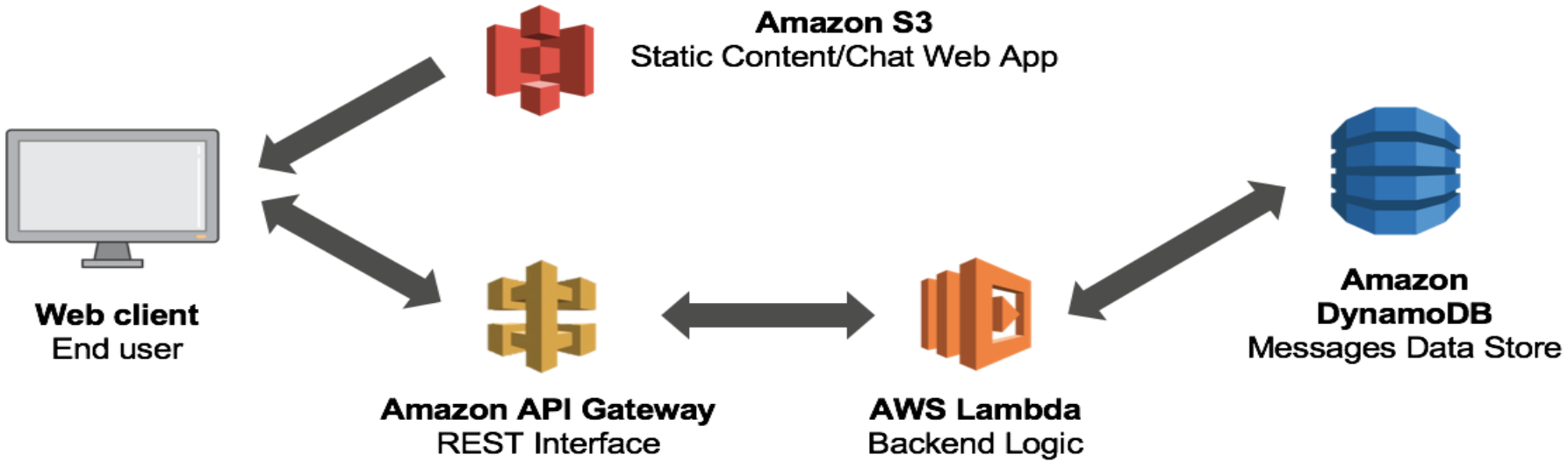
Cognito

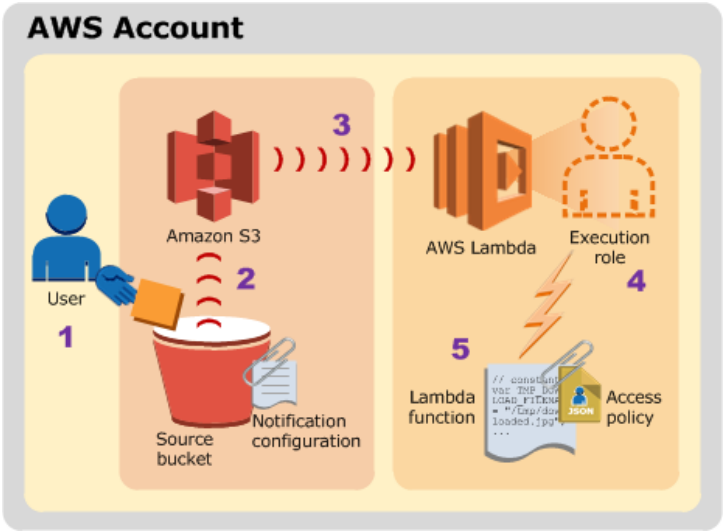
DynamoDB

The AWS Serverless Approach

What it all looks like in a picture

Diagram credited to: https://aws.amazon.com/blogs/compute/surviving-the-zombie-apocalypse-with-serverless-microservices/





http://docs.aws.amazon.com/lambda/latest/dg/intro-invocation-modes.html

Events

API Gateway

AWS API Gateway lets you focus on building your APIs. From easy creation of endpoints to fully managed resources. Plus, you only pay on calls to your API.

Streamline development

AWS allows for multiple APIs and versions at the same time

Scalable

Automatically scales across the network

APIs without servers

AWS manages all the infrastructure

Security, monitoring

Leverage AWS security and monitoring services

Lambda

Run your code, don’t worry about servers. No more provisioning or managing these resources. You only pay when the code runs.

Containers

Ready to go containers for many languages

Scaling

Automatically scales when your code is invoked/triggered

Metered

You only pay for compute time. Codes not running, you don’t pay.

Lambda internals

How does Lambda work at a high level?

Code

The code is actually stored in S3

Metadata

The metadata for the code is in DynamoDB

Execution

Execution is happening on an EC2 type instance

IAM

The function assumes an IAM role

Lambda basics

There are some basic concepts you need to keep in mind when using Lambda

Deployment

Deployed to service as a function. Functions are uploaded as a zip file.

Functions are concise

Functions are small concise pieces of logic. They usually perform one specific task.

Code organization

You can treat each function as a separate project. Your code has to be broken down and keenly organized.

Language and platform support

node, PHP, Python, Java, Ruby, Windows and .NET

Lambda caveats

Process exited before completing request

Yo code is broke, function exited wo/ calling context.done()

Timeouts

Long running processes will time out, and long process cost you more money. No Fibonacci please.

State

Lambdas are triggered, they go away when not in use.

HTTP Requests

You need API Gateway to trigger the Lambdas

EC2 Replacement

No, it's not.

VPC support?

Yes, w additional configuration. No dedicated tenancy VPC support.

Can I unit test Lambda functions?

Yes, you can test locally with a runner.

Yes, you can test once deployed, but you have to be clever. (Lambda calls Lambda records output, API Gateway using http testing harnesses)

Can I monitor my functions?

Yes, Lambdas are automatically setup with CloudWatch support.

Execution time limits?

Yes, a max of 300 seconds.

I have more questions

Go here: https://aws.amazon.com/lambda/faqs/

What about?

Development workflow

The different development work flows.

Manually through the AWS Console

Don’t, just stop. You will hate life. Do it another way.

AWS CLI

Better - AWS CLI tool allows you to do everything you need to. How’s your shell scripting?

Use a framework

Frameworks bring sanity, and consistency.

www.serverless.com

http://apex.run/

Cost

Are you ramping up?

Do you really know how many users you have?

Resources

Do you have the resources for EC2 or server management?

Can you scale?

Monolith vs small

Are you a monolith?

Can you break it down to small chunks?

Language and platform support

Is your architecture supported?

Can you transition to a supported platform?

Is serverless for me

Questions