Serverless

TACS

Qué es Serverless?

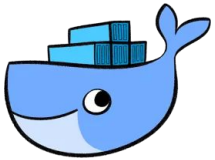
Serverless architectures are application designs that incorporate third-party "Backend as a Service" (BaaS) services, and/or that include custom code run in managed, ephemeral containers on a "Functions as a Service" (FaaS) platform. By using these ideas, and related ones like single-page applications, such architectures remove much of the need for a traditional always-on server component. Serverless architectures may benefit from significantly reduced operational cost, complexity, and engineering lead time, at a cost of increased reliance on vendor dependencies and comparatively immature supporting services.

Martin Fowler

Repaso

- Responsables de aprovisionar y administrar recursos
- Nuestro server permanece activo
- Mantener actualizado el servidor (IaaS)
- Estrategias de escalabilidad
- Failover
- Monitorear estado del server
- Inspeccionar el server ante errores
- +Tareas = +Tiempo = +$

Funciones como servicio -FaaS



```
var dbConnection = createNewDbConnection();

exports.myHandler = function(event, context, callback) {

  var result = dbConnection.makeQuery();

  callback(null, result);

};
```

Qué es Serverless?

- Es código propio que corre en una plataforma de terceros sobre containers efímeros como FaaS
- Es orientado a eventos
- El server esperando todo el tiempo ya no es tan necesario.
- Optimiza costos y complejidad de escala.
- Hay un vendor locking fuerte.

Por qué quiero usar Serverless?

- Simplificar la operatoria IT
  - No manejo más servidores, existen pero son transparentes para mí
  - Nos dedicamos a lógica que si aporta al negocio
- Alta escalabilidad desde el momento cero
  - Preparado para escalar indefinidamente
  - No tenemos que armar un plan de escalabilidad, nuestro proveedor se encarga de esto
- Bajo costo
  - Pagamos por lo que usamos
  - Acompaña económicamente al negocio y a su crecimiento
  - Ideal para Startups
- Confiabilidad
  - Arrancamos con un tecnología donde se paran los grandes

Dónde usar Serverless?

- Problemas simples y puntuales
- Lifecycle de mi función
  - Cold start
  - Ejecuta
  - Stand by (corto)
  - Muere

SI

NO

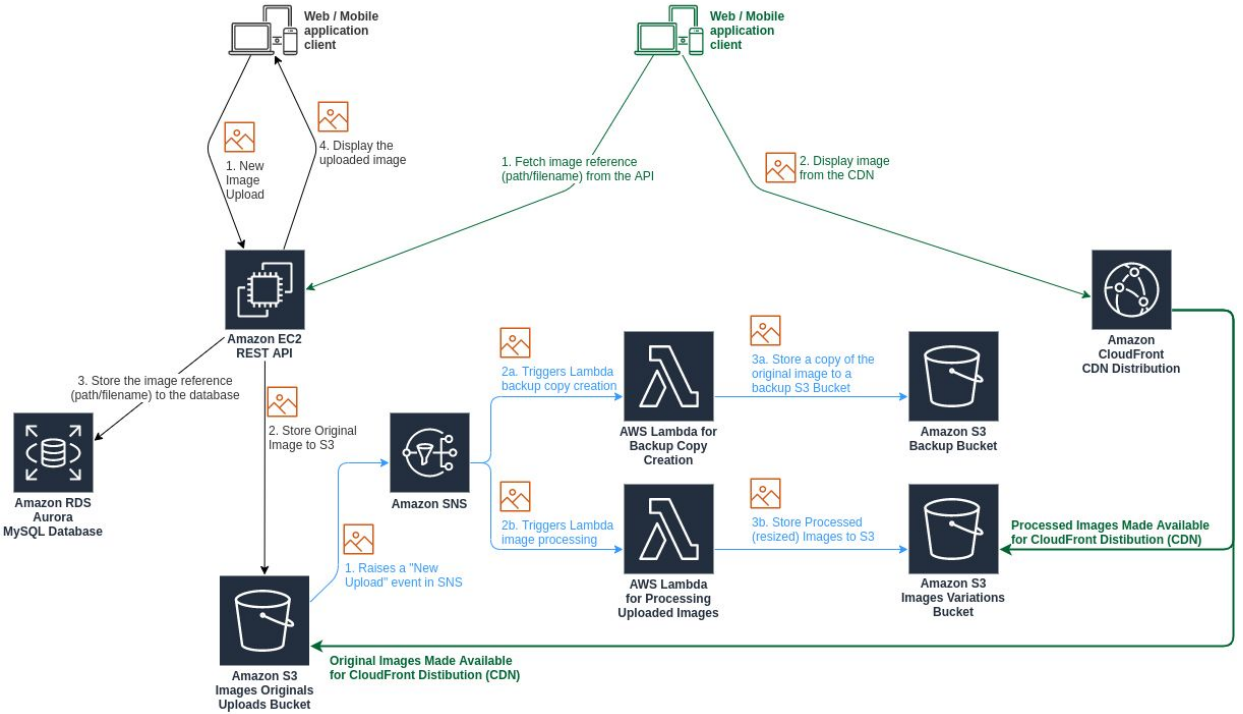
Ventajas

- Pagás por lo que usás
- Comprás procesamiento, mejor estimación de costos
- Built-In elasticity
- Event driven model
- Microservicios - Componentes deployables separados
- Inmutable
- Isolation
- Polyglot - Javascript, Python, Go, JVM languages, .NET
- Stateless
- Fault tolerant by default
- High availability by default

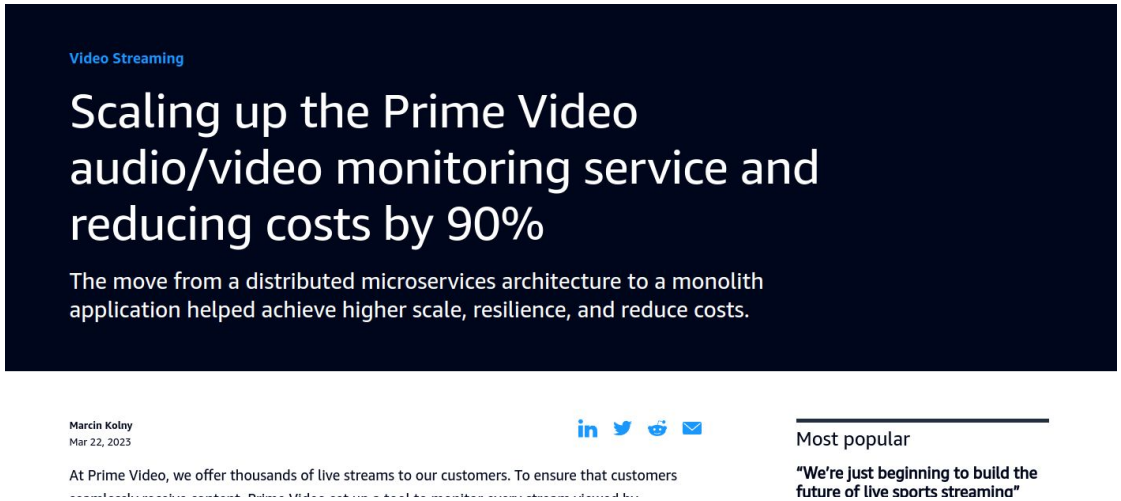
Desventajas

- Governance para evitar el caos
- Complejidad (+deploys,+Apis,+Network failures)
- Lock-in (código + servicios complementarios)
- Limitaciones (memoria, timeout 5-15m, deploy size, concurrency)
- Tests de integración complejos
- Debugging
- Observabilidad (monitoreo, alertas, log aggregation)
- Cold start problem
- Orquestación compleja

Ejemplo en AWS



¿Siempre es más barato?



https://www.primevideotech.com/video-streaming/scaling-up-the-prime-video-audio-video-monitoring-service-and-reducing-costs-by-90

Práctica

Serverless Framework

Serverless framework

-Open source

-Definimos desde una archivo de configuración YML

-Se puede utilizar con gran variedad de providers (AWS, Google,AZURE,etc)

Serverless framework - Guia practica

-Instalar Serverless cli y las credenciales de AWS

https://www.serverless.com/framework/docs/getting-started

serverless config credentials --provider aws --key MYKEY --secret MYSECRET

-Crear un proyecto esqueleto

```
serverless create --template aws-java-maven --name products-api -p aws-java-products-api
```

-Definir los endpoints en el YML

-Definir el handler de cada endpoint

Serverless Framework - Guia practica

-Empaquetar

```
mvn clean install
```

-Testear offline (agregar al final del yml->plugins:-serverless-offline)

```
npm install serverless-offline --save-dev
```

```
serverless offline start --disableCookieValidation
```

-Deploy

```
serverless deploy / serverless logs -f <FUNCTION>
```

-Release resources

```
serverless remove
```

Serverless Frameworks - Tutoriales y información útil

———

-Instalar serverless cli

https://www.serverless.com/framework/docs/getting-started/

-Configurar credenciales para poder hacer el deploy

https://www.serverless.com/framework/docs/providers/aws/cli-reference/config-credentials/

-Ejemplo de rest api

https://www.serverless.com/blog/how-to-create-a-rest-api-in-java-using-dynamodb-and-serverless

-Ejemplo de React app

https://www.youtube.com/watch?v=sMZm8HASKlM

Serverless Frameworks - Tutoriales y información util

-Ejemplo de instalación de paquetes externos (Layers)

https://www.serverless.com/blog/publish-aws-lambda-layers-serverless-framework

-Ejemplo de generador de thumbs

https://www.serverless.com/blog/building-a-serverless-screenshot-service-with-lambda

-Monitoring

https://www.serverless.com/blog/serverless-monitoring-the-good-the-bad-and-the-ugly

-Observability

https://www.serverless.com/blog/best-tools-serverless-observability

-Testing offline (serverless offline start)

https://www.youtube.com/watch?v=ul_85jfM0oo

Serverless Frameworks - Tutoriales y información útil

———

-Handle errors

https://www.serverless.com/blog/monitor-and-debug-all-serverless-errors

-Testing

https://www.serverless.com/blog/how-test-serverless-applications

-CI/CD

https://www.serverless.com/blog/ci-cd-workflow-serverless-apps-with-circleci

Preguntas?

Gracias!

TACS