Minesweeping on AWS

AWS Serverless 踩雷實務經驗談

WanCW, TechCCU 2016

- Who I Am
- What is Serverless
    - AWS Serverless Solution
- Why Serverless ?
- Why NOT Serverless (Now) ?
- Conclusion

Who I Am

Who I Am

- WanCW (GitHub, Twitter, ……)
- Backend Developer at KKTV
  - OTT Video Service from KKBOX
  - https://kktv.me/

Who I Am

- WanCW (GitHub, Twitter, ……)
- Backend Developer at KKTV
  - OTT Video Service from KKBOX
  - https://kktv.me/

What I do

- Coding for API ~~servers~~
- IT Operations (= use AWS)

What is "Serverless" ?

What is "Serverless" ?

Components of Web/API Server:

- Host/Instance/Container
- Language Runtime
- Application

What is "Serverless" ?

Components of Web/API Server:

- Host/Instance/Container - IaaS
- Language Runtime
- Application

What is "Serverless" ?

Components of Web/API Server:

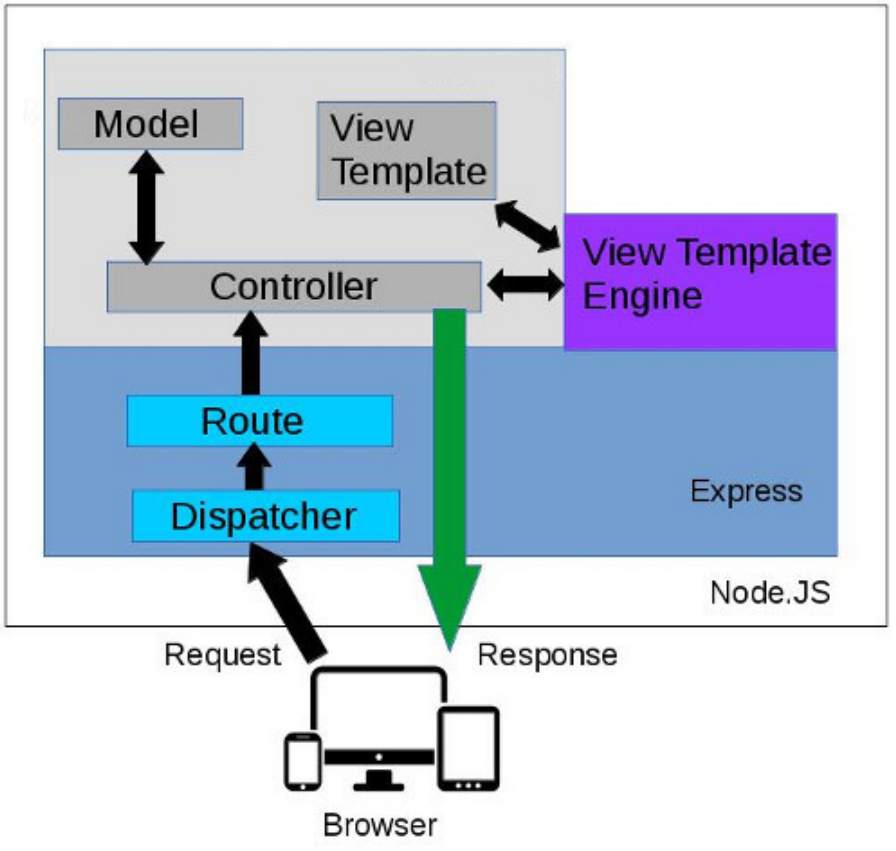
- Host/Instance/Container - IaaS
- Language Runtime - PaaS
- Application

What is "Serverless" ?

Components of Web/API Server:

- Host/Instance/Container - IaaS
- Language Runtime - PaaS
- Application - serverless
    - Your Code + Framework

from Your Code + Framework



to Your Code ~~+ Framework~~

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

  console.log("value1 = " + event.key1);
  console.log("value2 = " + event.key2);

  let sucess = doSomething();

  if (success) {
    callback(null, "some success message");
  } else {
    callback("some error type");
  }

}}
```

Just a generic function

AWS Serverless Solution

AWS Serverless Solution

- API Gateway
  - Map HTTP to backend services
  - REST interface

AWS Serverless Solution

- API Gateway
- Lambda
  - Where to put your logic
  - Runtime
    - Python
    - Node.js
    - Java

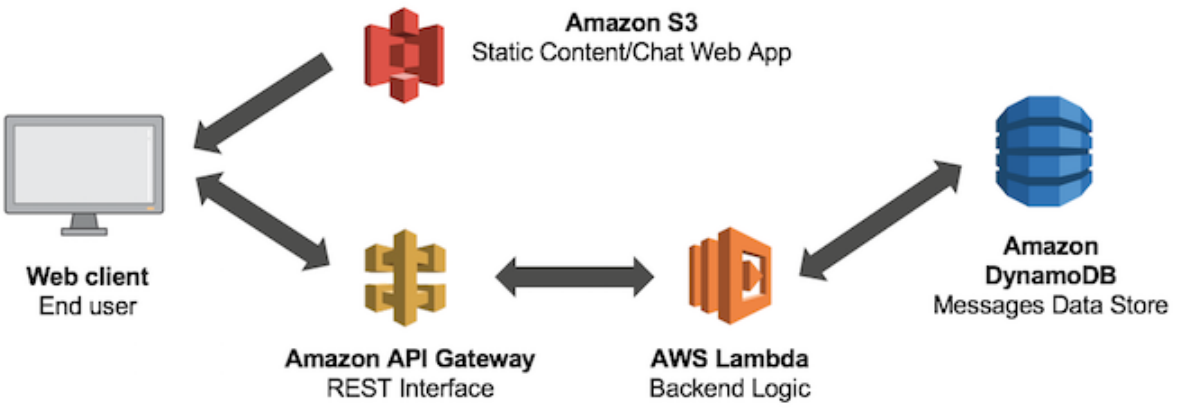
AWS Serverless Solution

- API Gateway
- Lambda
- DynamoDB
  - NoSQL DB
  - Document-oriented

AWS Serverless Solution

- API Gateway
- Lambda
- DynamoDB
- S3
  - Serve static content

AWS Serverless Solution



Why Serverless ?

Why Serverless ?

- (Almost) No Operation Cost

Why Serverless ?

- (Almost) No Operation Cost
- Unlimited Scale-Out

Why Serverless ?

- (Almost) No Operation Cost
- Unlimited Scale-Out
- General-Purpose Code

Why NOT Serverless (Now) ?

Why NOT Serverless (Now) ?

Problems We Met:

- Cold-Start
- Limits Reached
- Unstable Implementation

Problem: Cold-Start

- First invocation latency
  - Node.js: ~15s
  - Java: ~30s (reportedly)
- Pattern: **Prewarming**
- Pattern: **Lazy-evaluation**

Problem: Limits (Default)

AWS Lambda:

- Concurrent execution: 100/region
- Deployment package size: 50MB
- Total package size: 75GB/region

API Gateway:

- Throttle limit: 1000reqs/sec

Problem: Unstability

- Unstable Network
  e.g. Lost Redis Connection w/o Reason
- API Gateway **Interal Error** ~~*(WTF?!)*~~
- Lack of Information & Controllability
  - "***Open (AWS) ticket!***"

Why NOT Serverless (Now) ?

Other Issues:

- Code Organization among Functions
- Only 1 Success HTTP Status Code

Issue: Code Organization

- Common Logic between Functions
  - DRY/Shared Library?
  - Package Size Limitation?

Issue: Non-Default HTTP Status

API Gateway HTTP Status Code:

- 1 Default (for success)
- Others are Determined by Error Response

Pattern: **Error as Response**

Issue: Non-Default HTTP Status

API Gateway HTTP Status Code:

- 1 Default (for success)
- Others are Determined by Error Response

Pattern: **Error as Response**

Update at Sep. 20: **Lambda Function Proxy**

Conclusion for "Serverless" (1)

Pros:

- Low Cost

Cons:

- Bad Code Practice
- Lack of Controls

Conclusion for "Serverless" (2)

Good for:

- Quick Prototyping
- Background Worker

Bad for:

- High-Concurrency Service
- Hihgly-Coupled API Logic

Q & A

Extra

Pattern Examples

Pattern: Prewarm, Object Cache

```
var bootstrap = () => {
    console.log('start bootstrap')
    return new Promise((resolve, reject)=> {
        if(! ec2_object_cache.hasOwnProperty(region)){
            console.log('create ec2 object cache')
            AWS.config.update({region: region });
            ec2 = P.promisifyAll(new AWS.EC2())
            // save in the cache
            ec2_object_cache[region] = ec2
            resolve()
        } else {
            console.log('got ec2 object cache')
            ec2 = ec2_object_cache[region]
            resolve()
        }
    })
```

source: @pahudnet

Pattern: Error as Response

```
// Returns 302 or 301
var err = new Error("HandlerDemo.ResponseFound");
err.name = "http://a-different-uri";
context.done(err, {});
```

- 難懂、不易轉換回傳統架構

source: Redirection in a Serverless API with AWS Lambda and Amazon API Gateway

~~Pattern: Error as Response~~

New Feature: Lambda Function Proxy

```
var response = {
    statusCode: responseCode,
    headers: {
        "x-custom-header" : "my custom header value"
    },
    body: JSON.stringify(responseBody)
};
console.log("response: " + JSON.stringify(response))
context.succeed(response);
```