AWS Lambda using Serverless Framework

- Goran Kopevski -

Agenda

- What is Serverless, Lambda, Serverless framework?
- Development tools
- Demo
- Pros/Cons

What is Serverless concept?

Extensive use of third party services to accomplish tasks that are traditionally taken care of by servers

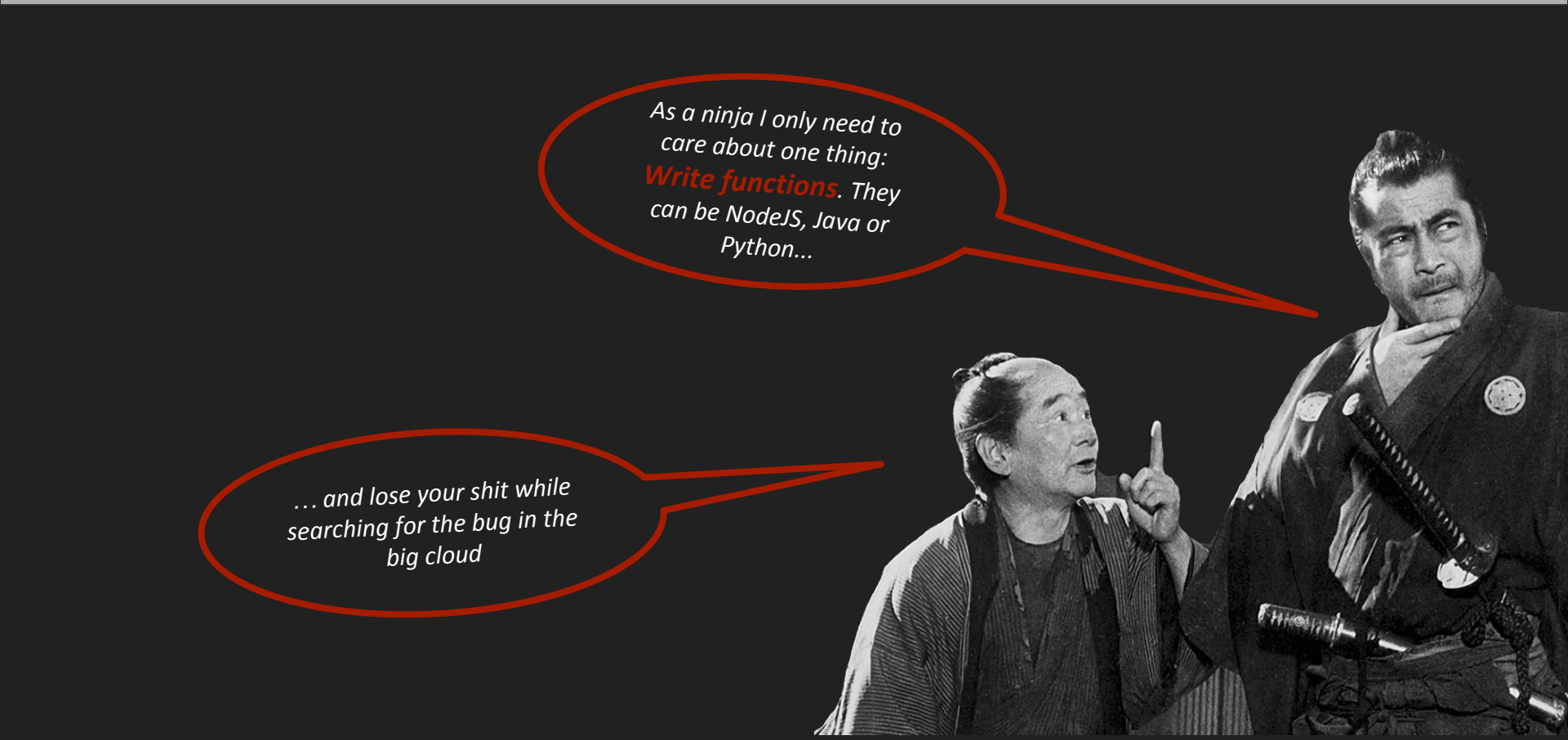
What is λ?

```
(Product, Provider) =>
    `Event-driven, serverless computing platform provided by
       ${Product} as a part of the ${Provider}`
```

Input: (Amazon, AWS), (Microsoft, Azure), (IBM, OpenWhisk)

- Run code without provisioning or managing servers

What is λ?


As a ninja I only need to care about one thing: Write functions. They can be NodeJS, Java or Python...
… and lose your shit while searching for the bug in the big cloud

What is SERVERLESS

- Allows you to deploy auto-scaling, pay-per-execution, event-driven functions to any cloud.
- Currently supports AWS, Apache OpenWhisk, Microsoft Azure

It helps you even more to cut down the boilerplate and ugly configuration that the cloud systems are forcing you to learn or click!

And yes, easy deployment!

What is happening under the hood?

```
(Serverless code) => cloudformation code;
```

- Reusable infrastructure definitions for other platforms
- Write less to achieve more

What is happening under the hood?

```yaml
service: myService

provider:
  name: aws
  runtime: nodejs4.3


functions:
  hello:
    handler: handler.hello

```

```json
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "The AWS CloudFormation template for this Serverless application",
  "Resources": {
    "ServerlessDeploymentBucket": {
      "Type": "AWS::S3::Bucket"
    }
  },
  "Outputs": {
    "ServerlessDeploymentBucketName": {
      "Value": {
        "Ref": "ServerlessDeploymentBucket"
      }
    }
  }
}
```

What is Serverless framework offering?

- Services
- Functions
- IAM
- Events
  - SNS, SQS
- Working with resources
- Working with other AWS services

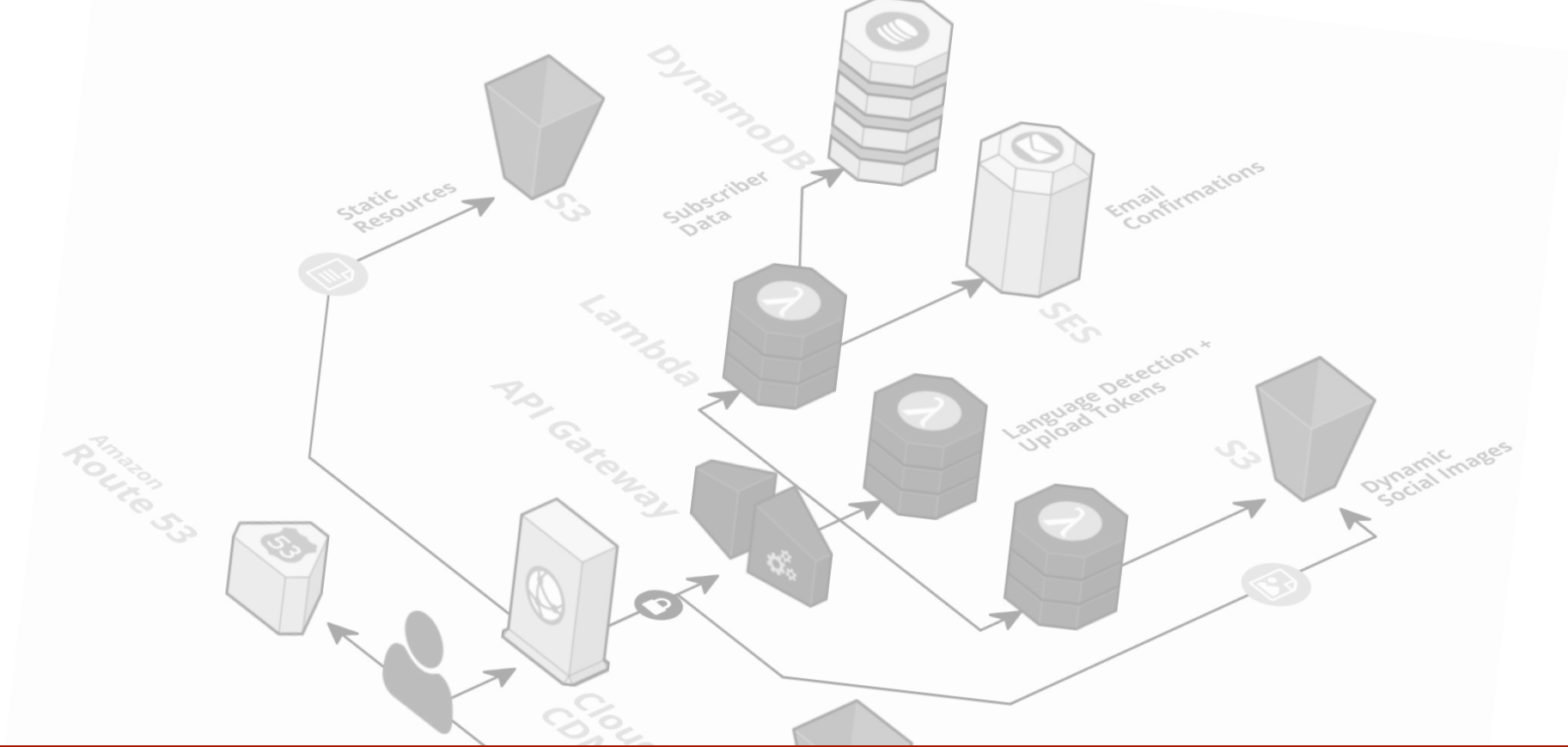
Development tools

It is easy:

- We need env variables for:
  - AWS_SECRET_ACCESS_KEY
  - AWS_ACCESS_KEY_ID
- Node and node package for serverless (sls)

We are ready for development

```
> sls create --template aws-nodejs --path myService
> sls deploy
```


Static Resources
S3
DynamoDB
Subscriber Data
Email Confirmations
SES
Lambda
API Gateway
Language Detection + Upload Tokens
S3
Dynamic Social Images
Amazon Route 53

DEMO

Plugins

- Extending or overwriting the core behaviour of the Serverless framework
- There is already big community which is contributing
- You can find the plugins at https://github.com/serverless/plugins

- Plugin which you must have in order to work more efficiently: serverless-offline

https://github.com/dherault/serverless-offline

PROS

- Easy understandable concept
- FasS - scaling costs
  - We don’t need to care about parallelism
- Reduced operational cost
- BaaS - reduced development cost
- Reduced packaging and deployment complexity
- Time to market / experimentation
- Operational management

- Abstractions over Vendor implementations

CONS

- Multitenancy Problems
- Vendor control
- Startup latency
  - This is debatable
- Execution Duration
- Really harder local development
- Testing
- Deployment
- Monitoring / Debugging

CONS

- Multitenancy Problems
- Vendor control
- Startup latency
  - This is debatable
- Execution Duration
- ~~Really harder local development~~
- ~~Testing~~
- ~~Deployment~~
- ~~Monitoring / Debugging~~

Should we use λ functions?

- New approach of development
- Still needs to mature
- Serverless is not efficient for long-running applications
- We need to learn the Amazon “tips and tricks” in order to pay less
- There are no “local” operations

Should we use λ functions?

But definitely useful for:

- *Message driven applications*
- *Sync processes*
- *Scheduling*
- *Automation of things*
- *Experiments and POCs*
- *CI/CD code builders*


QUESTIONS

Thank you!

Goran Kopevski

@ Cuponation

gkopevski@gmail.com