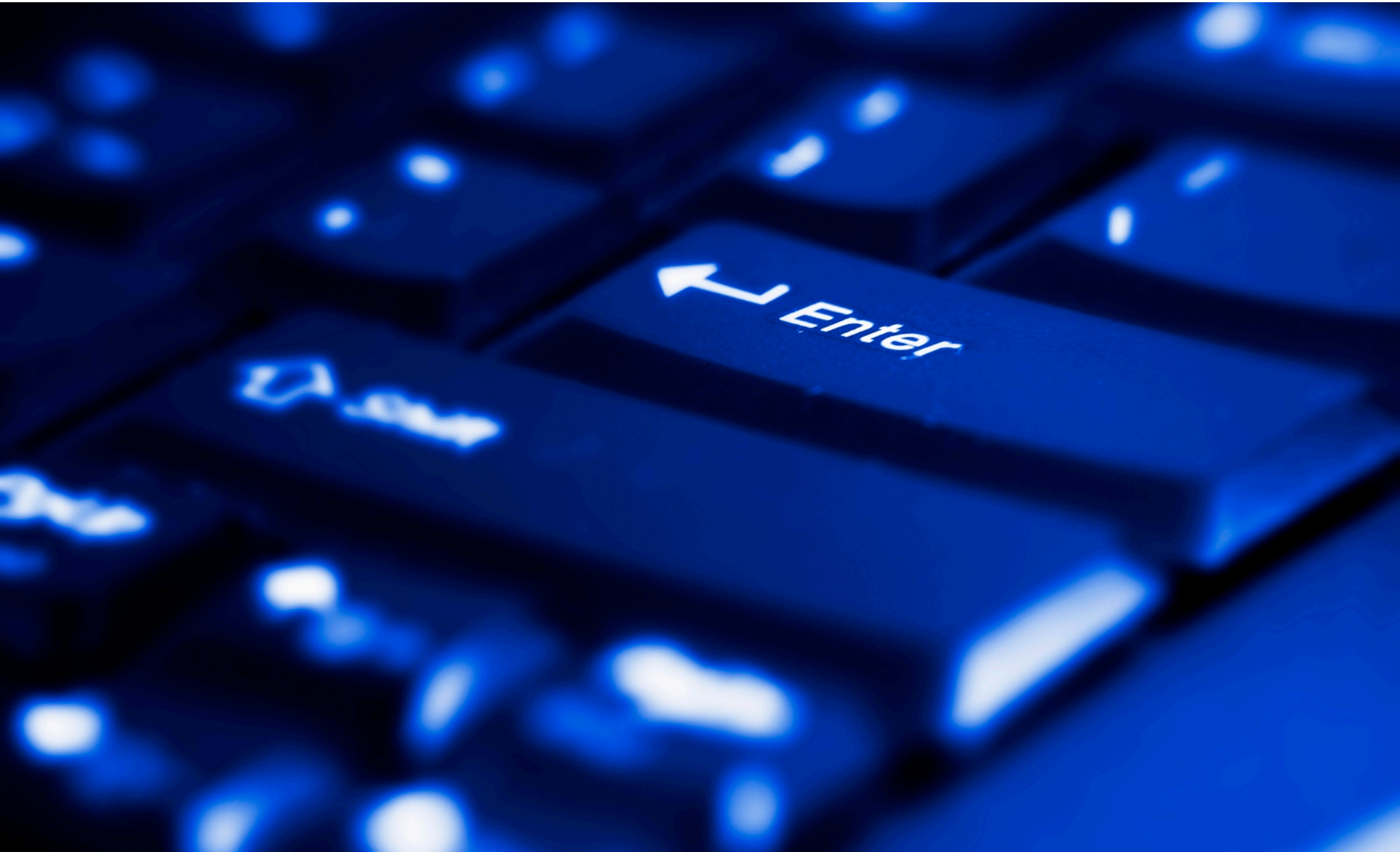
bns it

Serverless

Modern Application Architectures



Serverless

"Serverless architectures refer to applications that significantly depend on third-party services (knows as Backend as a Service or "BaaS") or on custom code that's run in ephemeral containers (Function as a Service or "FaaS"), the best known vendor host of which currently is AWS Lambda. By using these ..."

Mike Roberts

https://martinfowler.com/articles/serverless.html

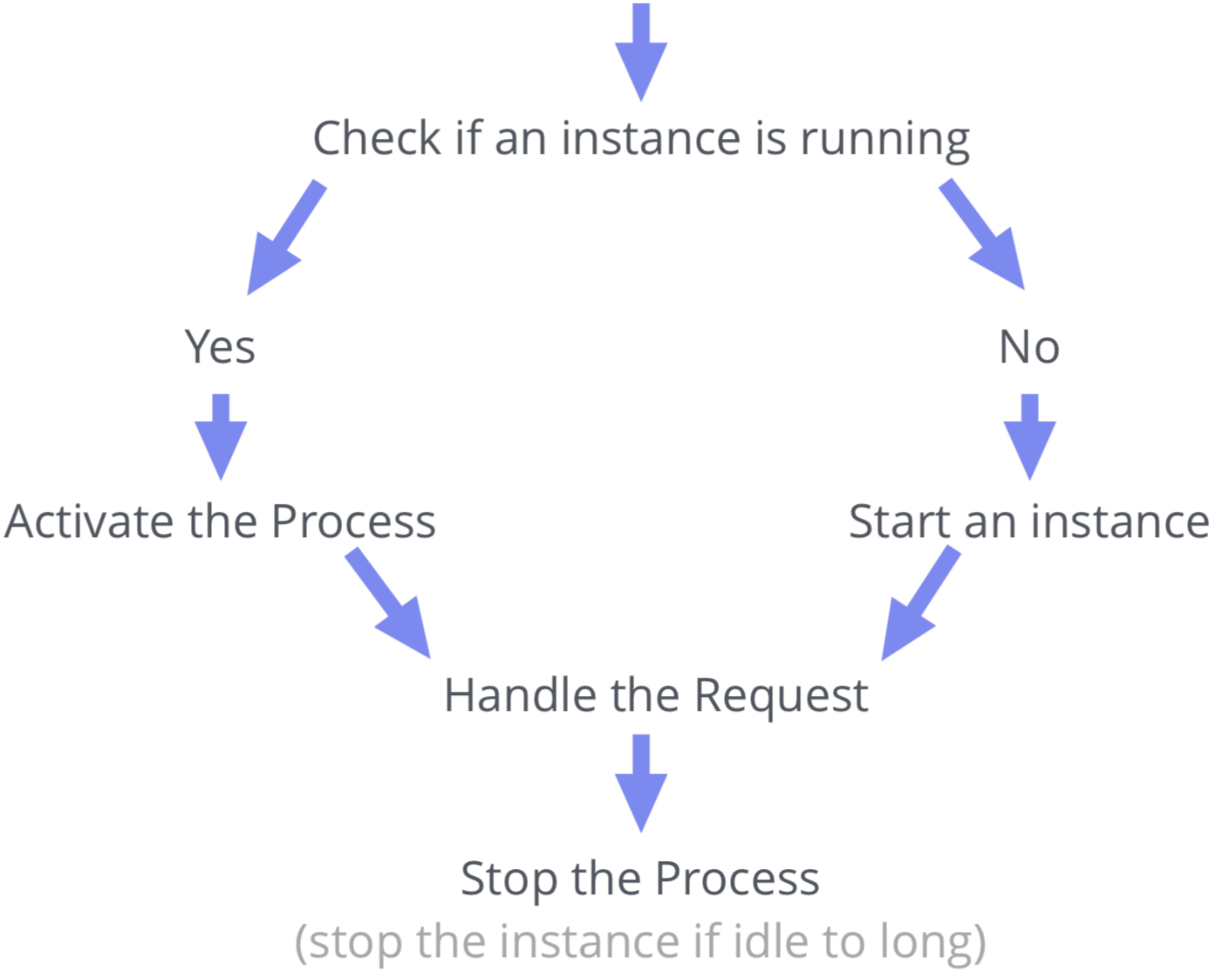
Faas

# Function as a unit of application logic

# New instances based on invocation demand

# Events trigger a function

Faas



Check if an instance is running

Yes

No

Activate the Process

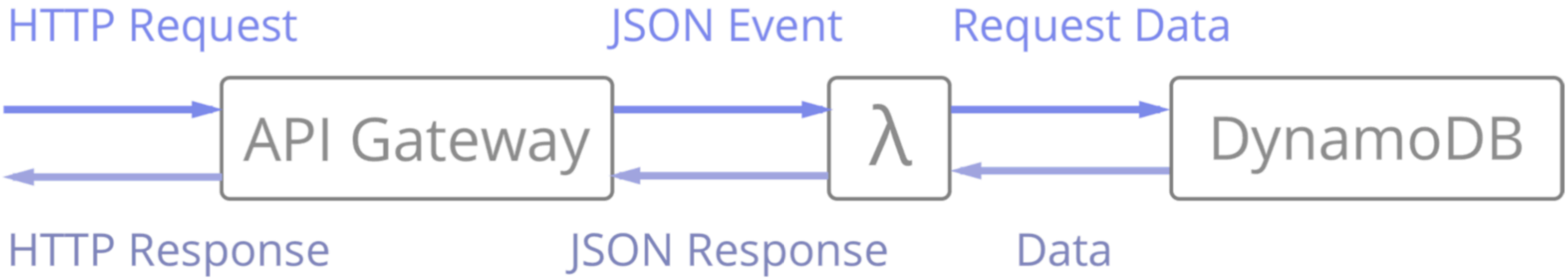
Start an instance

Handle the Request

Stop the Process

(stop the instance if idle to long)

Faas


HTTP Request
JSON Event
Request Data
API Gateway
λ
DynamoDB
HTTP Response
JSON Response
Data

Benefits

📈 Auto scaling

💵 Pay per execution

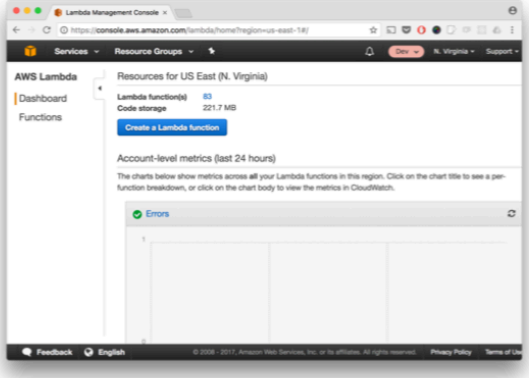
🎪 Event Driven

🗝️ Security

Providers

- AWS Lambda
- Google Cloud Functions
- Microsoft Functions
- IBM Openwhisk
- Auth0 Webtasks
- IronIO IronFunctions

Deploy via UI



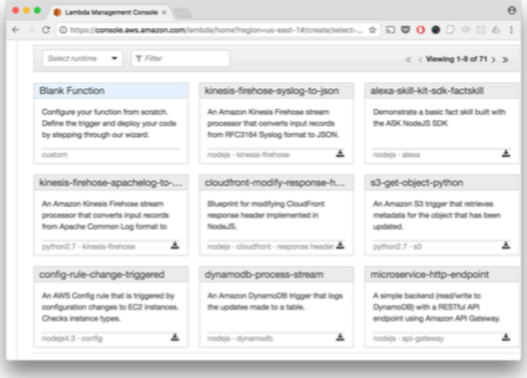
Resources for US East (N. Virginia)

Lambda function(s) 83
Code storage 221.7 MB

Account-level metrics (last 24 hours)

The charts below show metrics across **all** your Lambda functions in this region. Click on the chart title to see a per-function breakdown, or click on the chart body to view the metrics in CloudWatch.

© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.


Lambda Management Console
https://console.aws.amazon.com/lambda/home?region=us-east-1#/create/select-...
Select runtime
Filter
Viewing 1-9 of 71

Blank Function: Configure your function from scratch. Define the trigger and deploy your code by stepping through our wizard. (custom)

kinesis-firehose-syslog-to-json: An Amazon Kinesis Firehose stream processor that converts input records from RFC3164 Syslog format to JSON. (nodejs · kinesis-firehose)

alexa-skill-kit-sdk-factskill: Demonstrate a basic fact skill built with the ASK NodeJS SDK (nodejs · alexa)

kinesis-firehose-apachelog-to-...: An Amazon Kinesis Firehose stream processor that converts input records from Apache Common Log format to (python2.7 · kinesis-firehose)

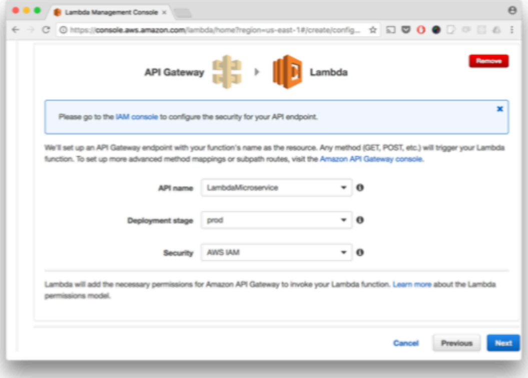
cloudfront-modify-response-h...: Blueprint for modifying CloudFront response header implemented in NodeJS. (nodejs · cloudfront · response header)

s3-get-object-python: An Amazon S3 trigger that retrieves metadata for the object that has been updated. (python2.7 · s3)

config-rule-change-triggered: An AWS Config rule that is triggered by configuration changes to EC2 instances. Checks instance types. (nodejs4.3 · config)

dynamodb-process-stream: An Amazon DynamoDB trigger that logs the updates made to a table. (nodejs · dynamodb)

microservice-http-endpoint: A simple backend (read/write to DynamoDB) with a RESTful API endpoint using Amazon API Gateway. (nodejs · api-gateway)

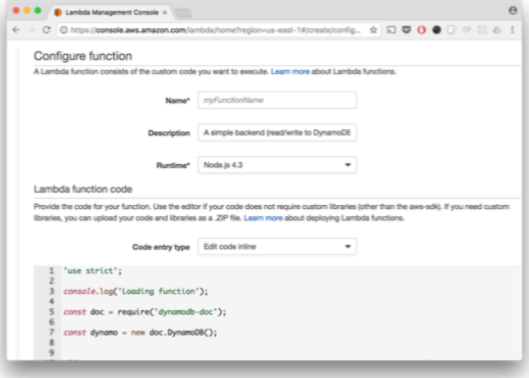

Lambda Management Console
https://console.aws.amazon.com/lambda/home?region=us-east-1#/create/config...
API Gateway
Lambda
Remove
Cancel
Previous
Next

Please go to the IAM console to configure the security for your API endpoint.

We'll set up an API Gateway endpoint with your function's name as the resource. Any method (GET, POST, etc.) will trigger your Lambda function. To set up more advanced method mappings or subpath routes, visit the Amazon API Gateway console.

API name: LambdaMicroservice
Deployment stage: prod
Security: AWS IAM

Lambda will add the necessary permissions for Amazon API Gateway to invoke your Lambda function. Learn more about the Lambda permissions model.


Lambda Management Console
https://console.aws.amazon.com/lambda/home?region=us-east-1#/create/config...

Configure function

A Lambda function consists of the custom code you want to execute. Learn more about Lambda functions.

Name*: myFunctionName
Description: A simple backend (read/write to DynamoDE
Runtime*: Node.js 4.3

Lambda function code

Provide the code for your function. Use the editor if your code does not require custom libraries (other than the aws-sdk). If you need custom libraries, you can upload your code and libraries as a .ZIP file. Learn more about deploying Lambda functions.

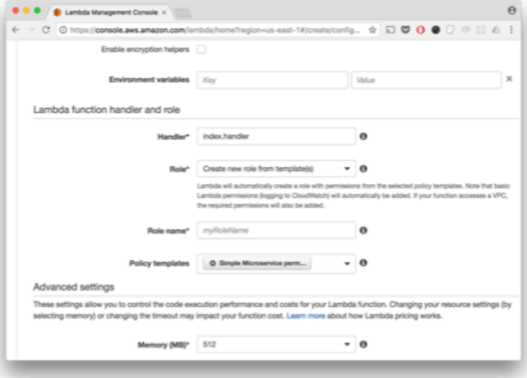
Code entry type: Edit code inline

```
'use strict';

console.log('Loading function');

const doc = require('dynamodb-doc');

const dynamo = new doc.DynamoDB();
```


Lambda Management Console
https://console.aws.amazon.com/lambda/home?region=us-east-1#/create/config...

Enable encryption helpers □

Environment variables: Key Value

Lambda function handler and role

Handler*: index.handler
Role*: Create new role from template(s)

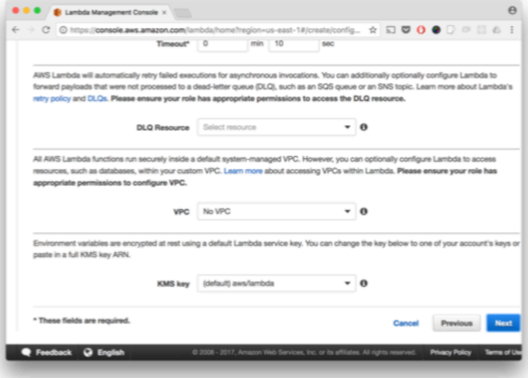
Lambda will automatically create a role with permissions from the selected policy templates. Note that basic Lambda permissions (logging to CloudWatch) will automatically be added. If your function accesses a VPC, the required permissions will also be added.

Role name*: myRoleName
Policy templates: Simple Microservice perm...

Advanced settings

These settings allow you to control the code execution performance and costs for your Lambda function. Changing your resource settings (by selecting memory) or changing the timeout may impact your function cost. Learn more about how Lambda pricing works.

Memory (MB)*: 512


Lambda Management Console
https://console.aws.amazon.com/lambda/home?region=us-east-1#/create/config...
Cancel
Previous
Next
Feedback
English
Privacy Policy
Terms of Use

Timeout*: 0 min 10 sec

AWS Lambda will automatically retry failed executions for asynchronous invocations. You can additionally optionally configure Lambda to forward payloads that were not processed to a dead-letter queue (DLQ), such as an SQS queue or an SNS topic. Learn more about Lambda's retry policy and DLQs. **Please ensure your role has appropriate permissions to access the DLQ resource.**

DLQ Resource: Select resource

All AWS Lambda functions run securely inside a default system-managed VPC. However, you can optionally configure Lambda to access resources, such as databases, within your custom VPC. Learn more about accessing VPCs within Lambda. **Please ensure your role has appropriate permissions to configure VPC.**

VPC: No VPC

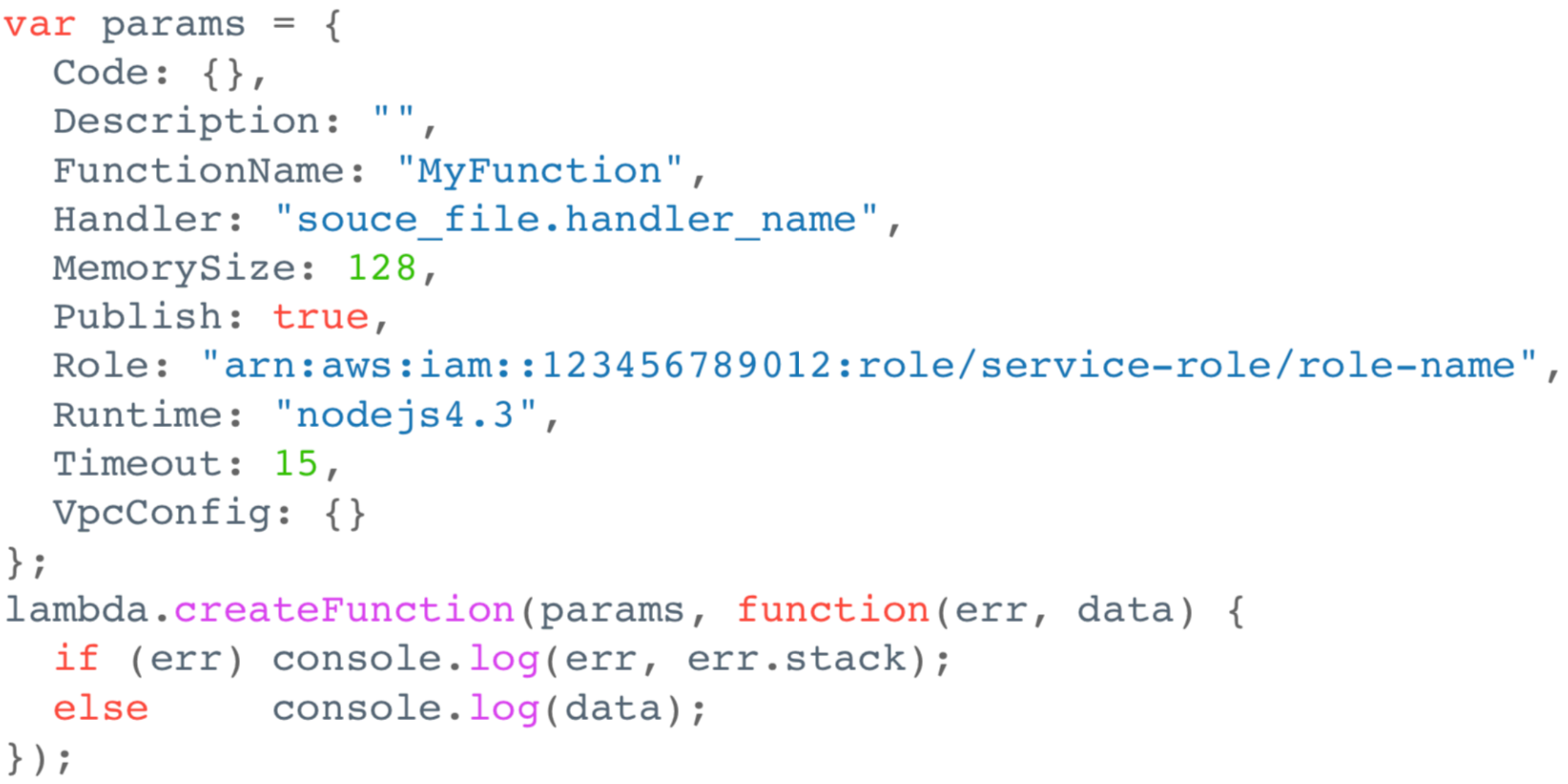
Environment variables are encrypted at rest using a default Lambda service key. You can change the key below to one of your account's keys or paste in a full KMS key ARN.

KMS key: (default) aws/lambda

* These fields are required.

© 2008 - 2017, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Deploy via UI



```
var params = {
  Code: {},
  Description: "",
  FunctionName: "MyFunction",
  Handler: "souce_file.handler_name",
  MemorySize: 128,
  Publish: true,
  Role: "arn:aws:iam::123456789012:role/service-role/role-name",
  Runtime: "nodejs4.3",
  Timeout: 15,
  VpcConfig: {}
};
lambda.createFunction(params, function(err, data) {
  if (err) console.log(err, err.stack);
  else     console.log(data);
});
```

AWS Lambda ecosystem

# EC2 - VMs

- „I want to configure machines, storage, networking, and my OS”

# ECS – Containers

- „I want to run servers, configure applications, and control scaling”

# AWS Lamda

- „Run my code when its needed”

What's supported

# **Event handlers** one function per event type

# **Serverless backends** one function per API / path

# **Data processing** one function per data type

Triggers

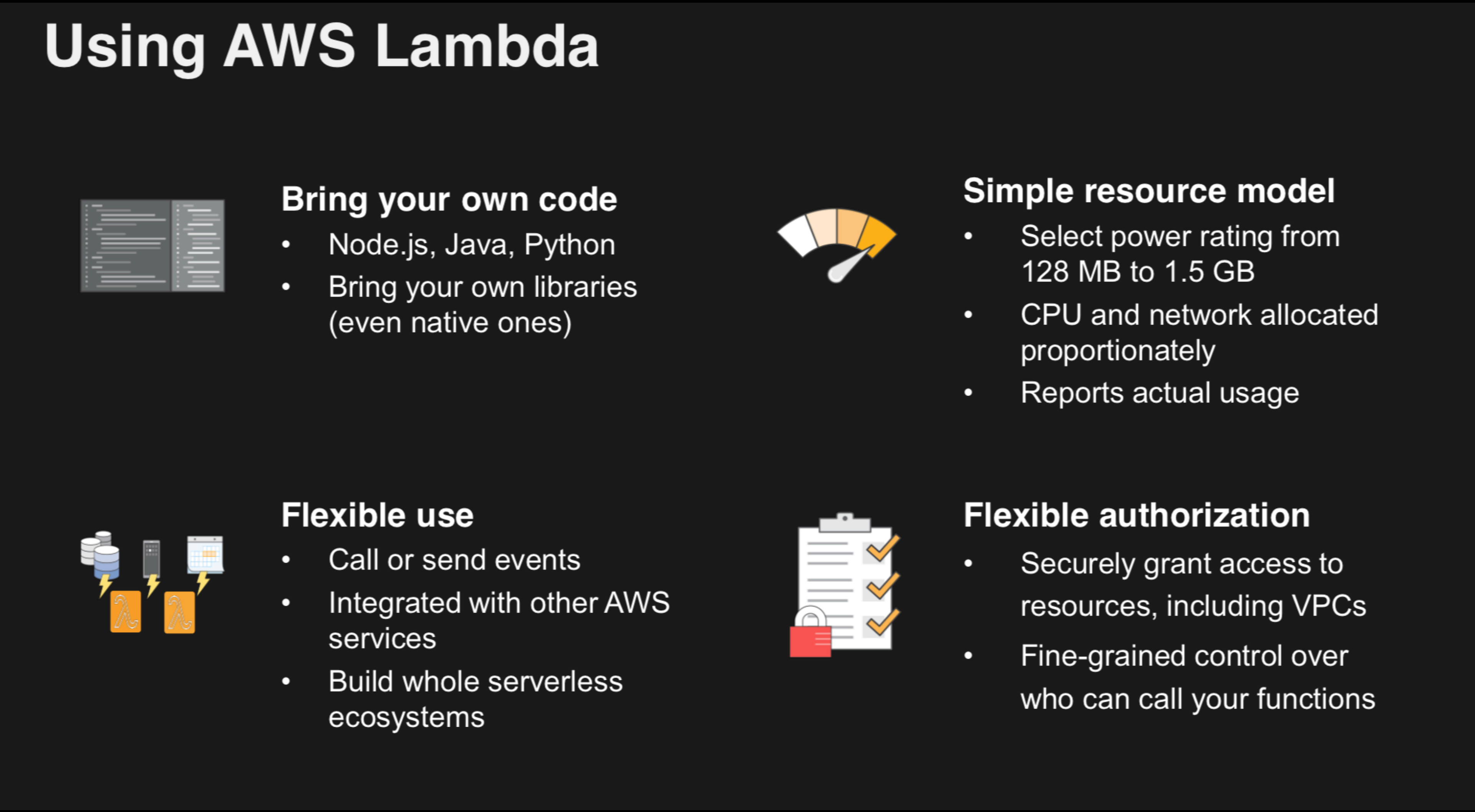
# PUT to an Amazon S3 bucket

# Updates to Amazon DynamoDB table

# Call to an Amazon API Gateway endpoint

# Mobile app back - end call

# And many more……



Using AWS Lambda

Bring your own code

- Node.js, Java, Python
- Bring your own libraries (even native ones)

Simple resource model

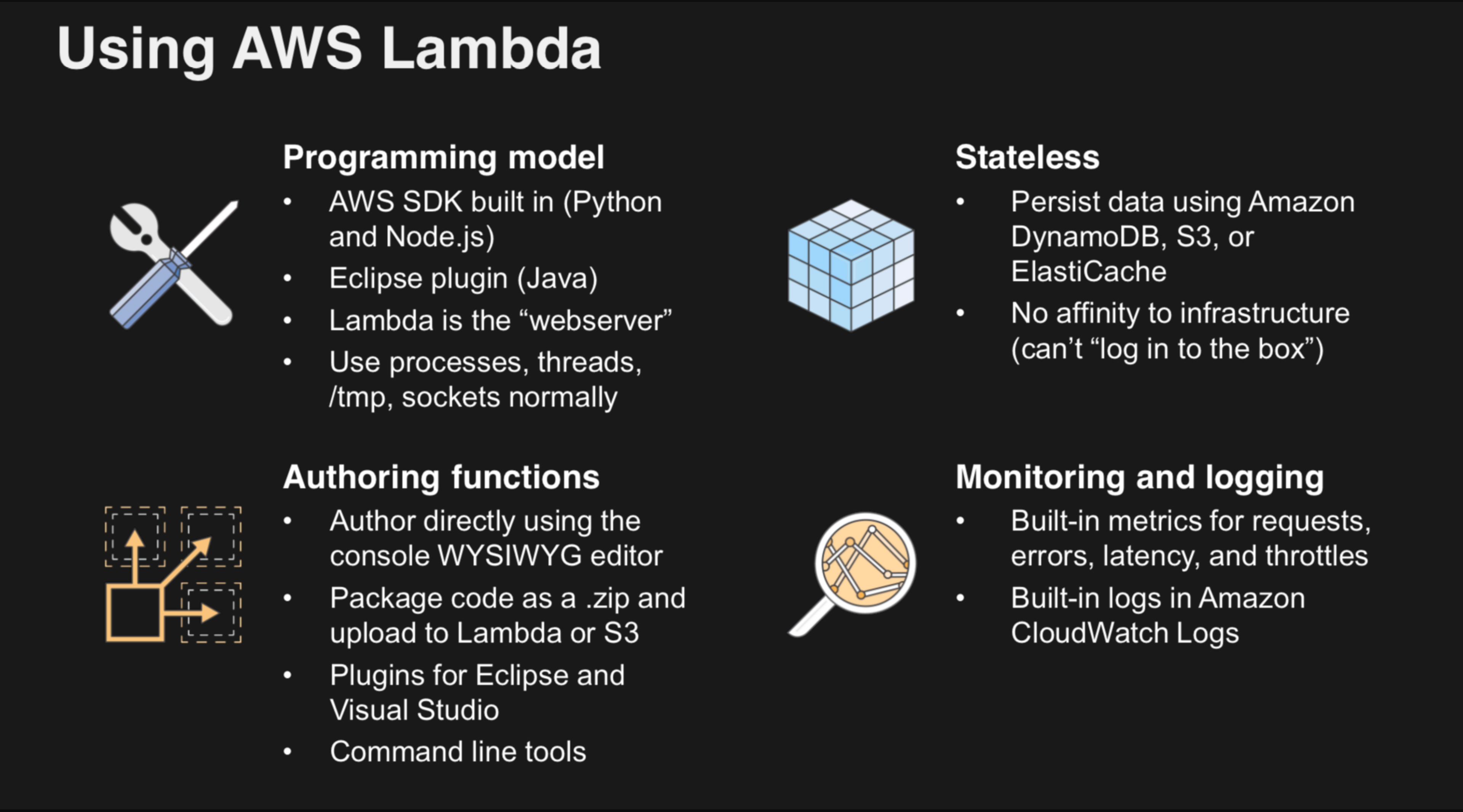
- Select power rating from 128 MB to 1.5 GB
- CPU and network allocated proportionately
- Reports actual usage

Flexible use

- Call or send events
- Integrated with other AWS services
- Build whole serverless ecosystems

Flexible authorization

- Securely grant access to resources, including VPCs
- Fine-grained control over who can call your functions



Using AWS Lambda

Programming model

- AWS SDK built in (Python and Node.js)
- Eclipse plugin (Java)
- Lambda is the “webserver”
- Use processes, threads, /tmp, sockets normally

Stateless

- Persist data using Amazon DynamoDB, S3, or ElastiCache
- No affinity to infrastructure (can’t “log in to the box”)

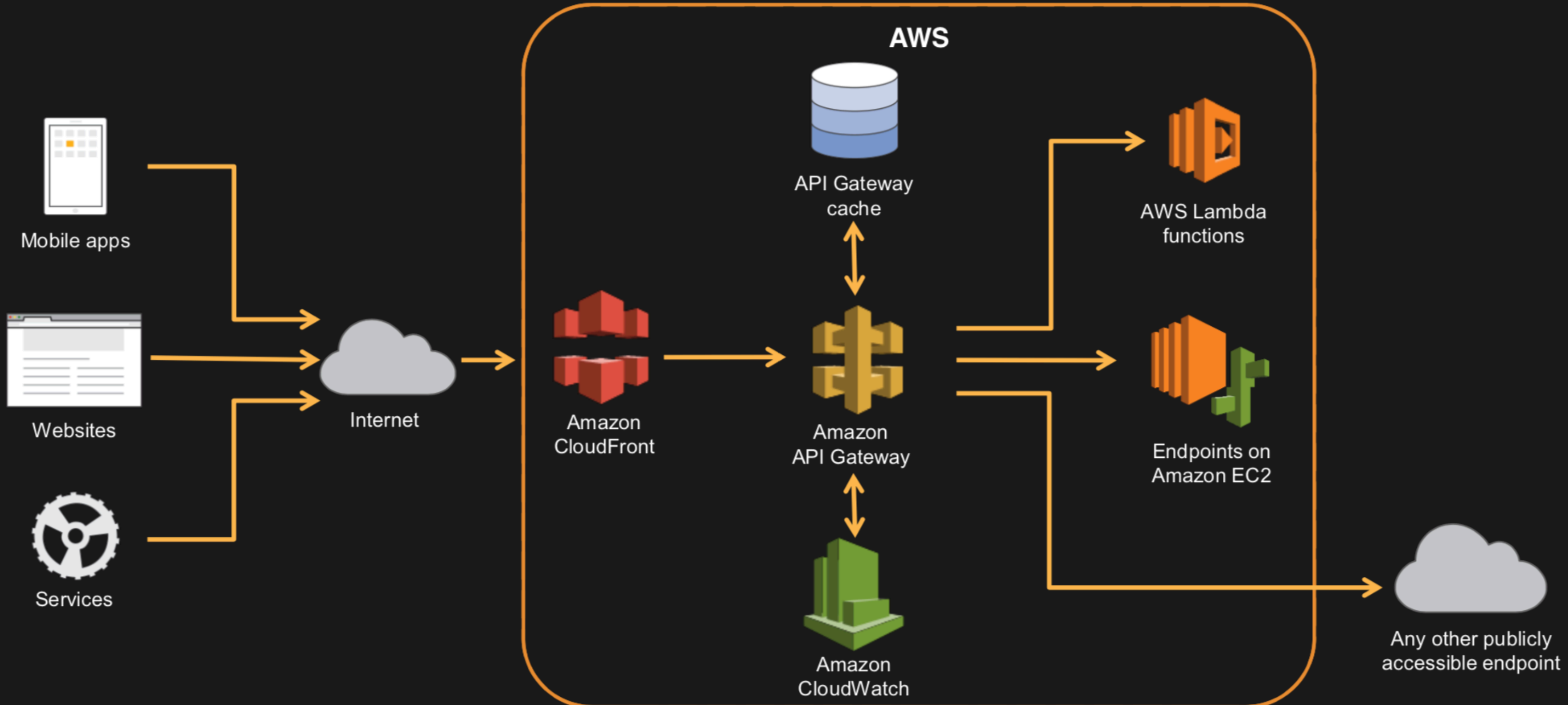
Authoring functions

- Author directly using the console WYSIWYG editor
- Package code as a .zip and upload to Lambda or S3
- Plugins for Eclipse and Visual Studio
- Command line tools

Monitoring and logging

- Built-in metrics for requests, errors, latency, and throttles
- Built-in logs in Amazon CloudWatch Logs

Amazon Serverless Manifesto


AWS
Mobile apps
Websites
Services
Internet
Amazon CloudFront
API Gateway cache
Amazon API Gateway
Amazon CloudWatch
AWS Lambda functions
Endpoints on Amazon EC2
Any other publicly accessible endpoint