https://bit.ly/2zJJ3Fh

aws

AWS Israel Community

aws

Deep dive into new AWS services

AWS Israel Community

- Founded - Feb **2013**
- **87** meetups with **~6700** Members
- Monthly meetups
- No Marketing, No bullshit
- All AWS: AI, BigData, Serverless, Containers, etc

MEET THE TEAM

Shimon Tolts

Arthur Schmunk

Tal Hibner

Niv Yungelson

Eitan Sela

Doron Rogov

Boaz Ziniman

Join the Community!

https://bit.ly/2zJJ3Fh

https://www.meetup.com/AWS-IL/

https://www.meetup.com/AWS-IL/

AWS User Group Israel

AWS Israel Community

Coming meetups

https://aws.org.il/

Deep dive into new AWS services

- Firecracker Container by Niv Yungelson - DevOps Team Lead @ Skycure (acquired by Symantec)
- Lambda Layers by Eitan Sela - Cloud & Big Data Systems Architect @ WeissBeerger

ironSource

Thank you

Firecracker

Niv Yungelson

First of her team

Ruler of all regions

Leader of the AWS-IL user group

Baker of sparkly unicorn cakes

And mother of Nala

Niv_yungelson@Symantec.com

What am I going to talk about

- What’s Firecracker?

- Things that we’re already familiar with that are running on Firecracker

- Advantages of the use of the Firecracker technology

- Why shouldn’t we use it

Disclaimer

Firecracker What?

- MicroVM technology

- KVM based

- Amazon built it for their Lambda and Fargate services

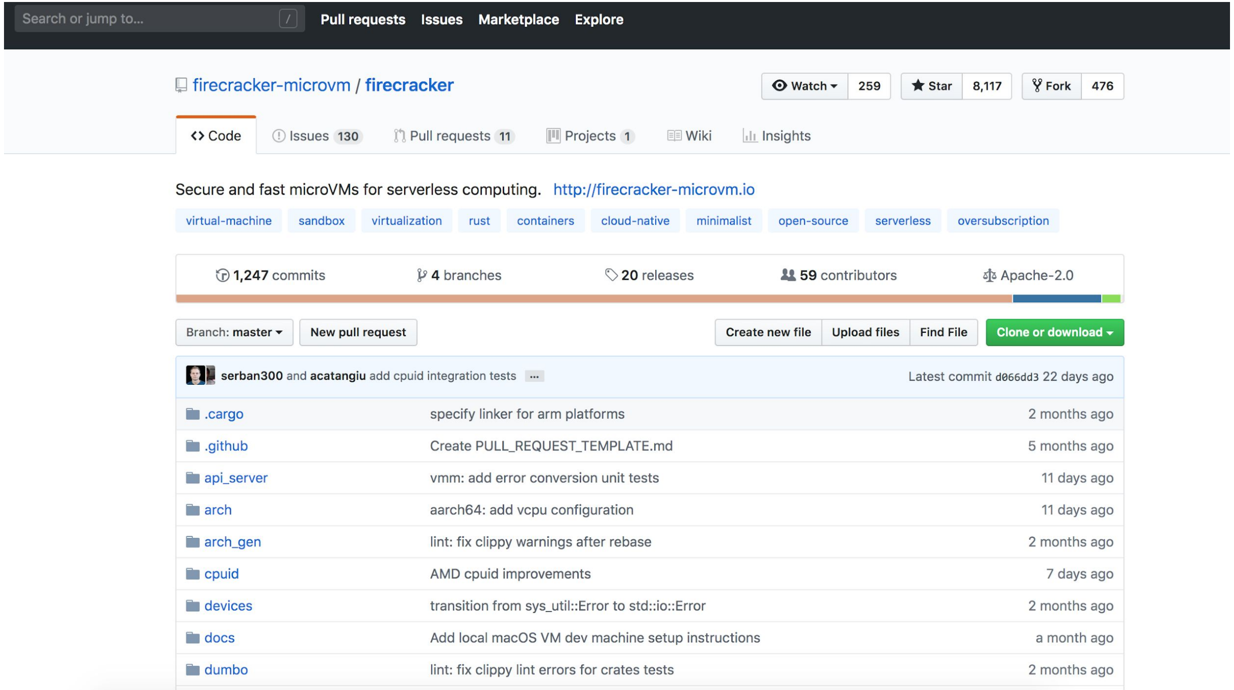
Have we met before?

- Lambda functions – from running on per-costumer EC2 instances managed by AWS, to Firecracker.
- Fargate – From Docker containers on EC2 instances to MicroVMs .

Why is it good?

- More secure –
  - Each VM runs on their own kernel
  - Jailed with seccompBPF (SECure COMPuting with filters)
- Faster - produce 5Mb microVMs that spin up in around 125ms
- Lightweight - About 5 MiB (~5.24 MB) of memory.
- Open Source!

Open source



firecracker-microvm / firecracker

Secure and fast microVMs for serverless computing. http://firecracker-microvm.io

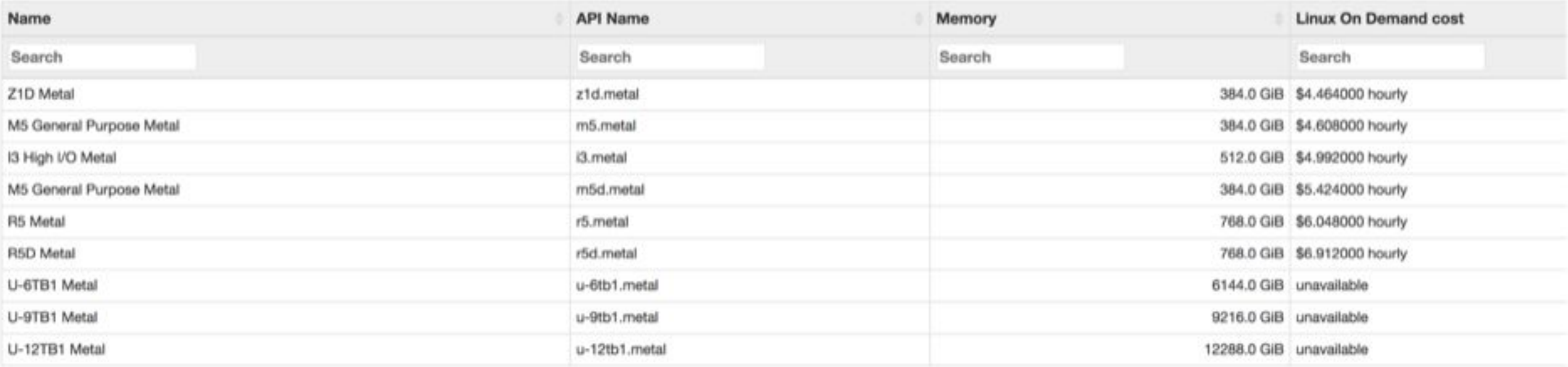
virtual-machine sandbox virtualization rust containers cloud-native minimalist open-source serverless oversubscription

1,247 commits · 4 branches · 20 releases · 59 contributors · Apache-2.0

serban300 and acatangiu add cpuid integration tests ... Latest commit d066dd3 22 days ago

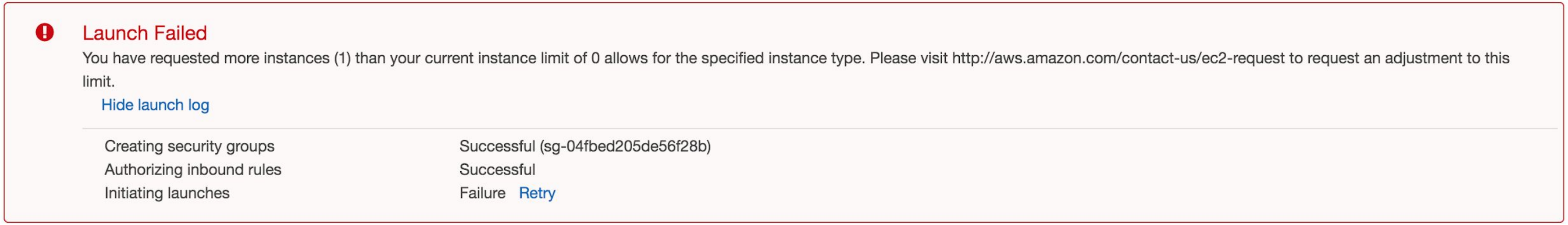
.cargo	specify linker for arm platforms	2 months ago
.github	Create PULL_REQUEST_TEMPLATE.md	5 months ago
api_server	vmm: add error conversion unit tests	11 days ago
arch	aarch64: add vcpu configuration	11 days ago
arch_gen	lint: fix clippy warnings after rebase	2 months ago
cpuid	AMD cpuid improvements	7 days ago
devices	transition from sys_util::Error to std::io::Error	2 months ago
docs	Add local macOS VM dev machine setup instructions	a month ago
dumbo	lint: fix clippy lint errors for crates tests	2 months ago

.metal



Name	API Name	Memory	Linux On Demand cost
Search	Search	Search	Search
Z1D Metal	z1d.metal	384.0 GiB	$4.464000 hourly
M5 General Purpose Metal	m5.metal	384.0 GiB	$4.608000 hourly
I3 High I/O Metal	i3.metal	512.0 GiB	$4.992000 hourly
M5 General Purpose Metal	m5d.metal	384.0 GiB	$5.424000 hourly
R5 Metal	r5.metal	768.0 GiB	$6.048000 hourly
R5D Metal	r5d.metal	768.0 GiB	$6.912000 hourly
U-6TB1 Metal	u-6tb1.metal	6144.0 GiB	unavailable
U-9TB1 Metal	u-9tb1.metal	9216.0 GiB	unavailable
U-12TB1 Metal	u-12tb1.metal	12288.0 GiB	unavailable

<insert Demo Here>

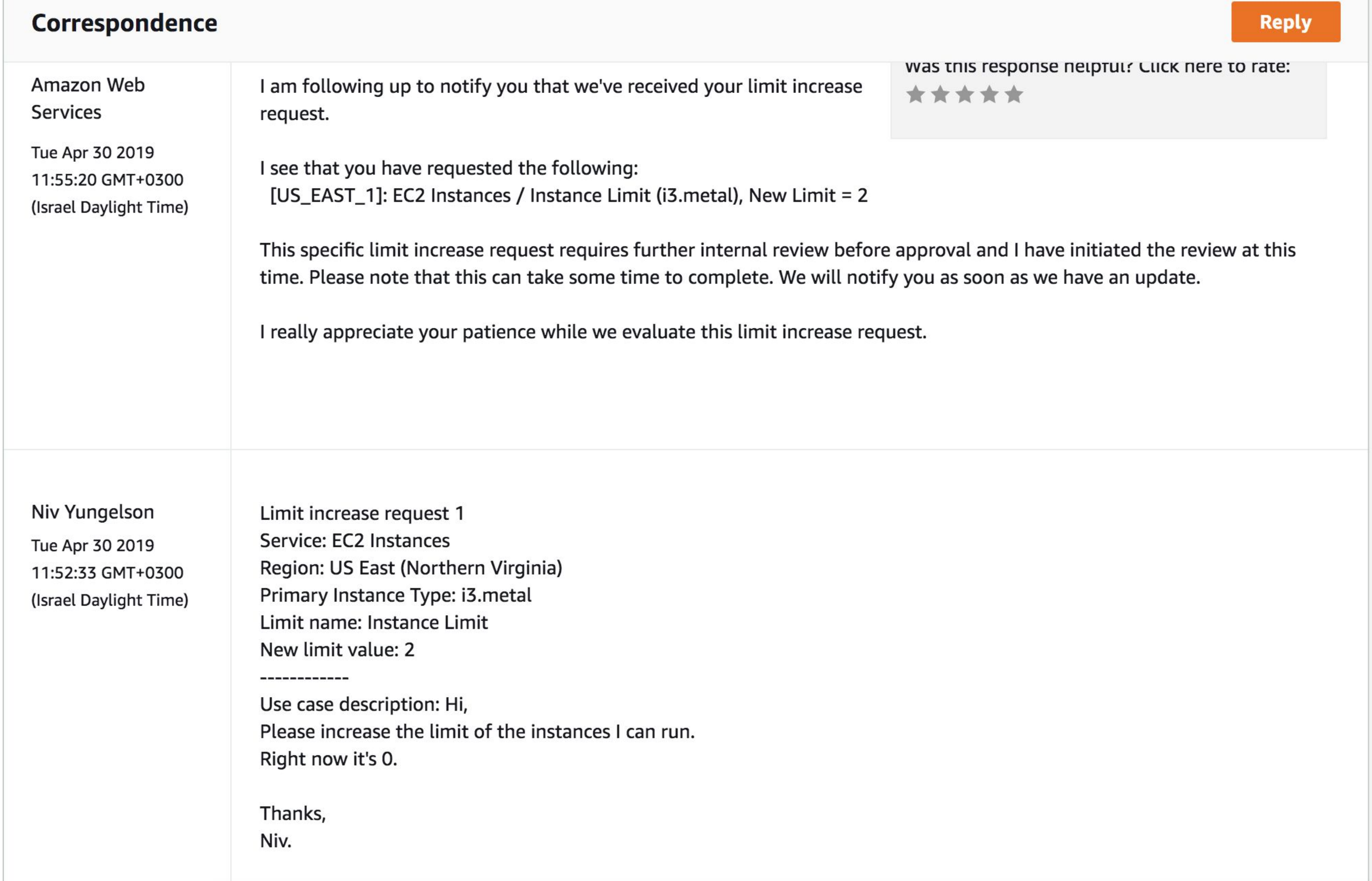


Launch Failed

You have requested more instances (1) than your current instance limit of 0 allows for the specified instance type. Please visit http://aws.amazon.com/contact-us/ec2-request to request an adjustment to this limit.

Hide launch log

Creating security groups	Successful (sg-04fbed205de56f28b)
Authorizing inbound rules	Successful
Initiating launches	Failure Retry



Correspondence

Amazon Web Services
Tue Apr 30 2019
11:55:20 GMT+0300
(Israel Daylight Time)

I am following up to notify you that we've received your limit increase request.

I see that you have requested the following:
[US_EAST_1]: EC2 Instances / Instance Limit (i3.metal), New Limit = 2

This specific limit increase request requires further internal review before approval and I have initiated the review at this time. Please note that this can take some time to complete. We will notify you as soon as we have an update.

I really appreciate your patience while we evaluate this limit increase request.

Niv Yungelson
Tue Apr 30 2019
11:52:33 GMT+0300
(Israel Daylight Time)

Limit increase request 1
Service: EC2 Instances
Region: US East (Northern Virginia)
Primary Instance Type: i3.metal
Limit name: Instance Limit
New limit value: 2

Use case description: Hi,
Please increase the limit of the instances I can run.
Right now it's 0.

Thanks,
Niv.

Why not?

- It provides no support for graphics or other accelerators, and no hardware pass-through, it only works with very recent kernels, and only with specific compilation options.
- Ruins the magic of serverless.

firecracker-containerd

“This repository enables the use of a container runtime, containerd, to manage Firecracker microVMs. Like traditional containers, Firecracker microVMs offer fast start-up and shut-down and minimal overhead. Unlike traditional containers, however, they can provide an additional layer of isolation via the KVM hypervisor.”

Questions?

How to use AWS Lambda Layers

Eitan Sela - Cloud & Big Data Systems Architect

WeissBeerger

A part of the ABInBev Family

eitan.sela@weissbeerger.com

$ whoami

- "Hands-On" system Architect with more than 18 years of experience with billing, banking, information security (DLP) and Cloud IoT/Big Data applications.

- Big Data specialist – Hadoop, Spark, Hive and EMR on AWS.

- Work with vast AWS services and with serverless projects especially.

- Java development, scalability performance and stabilization expert.

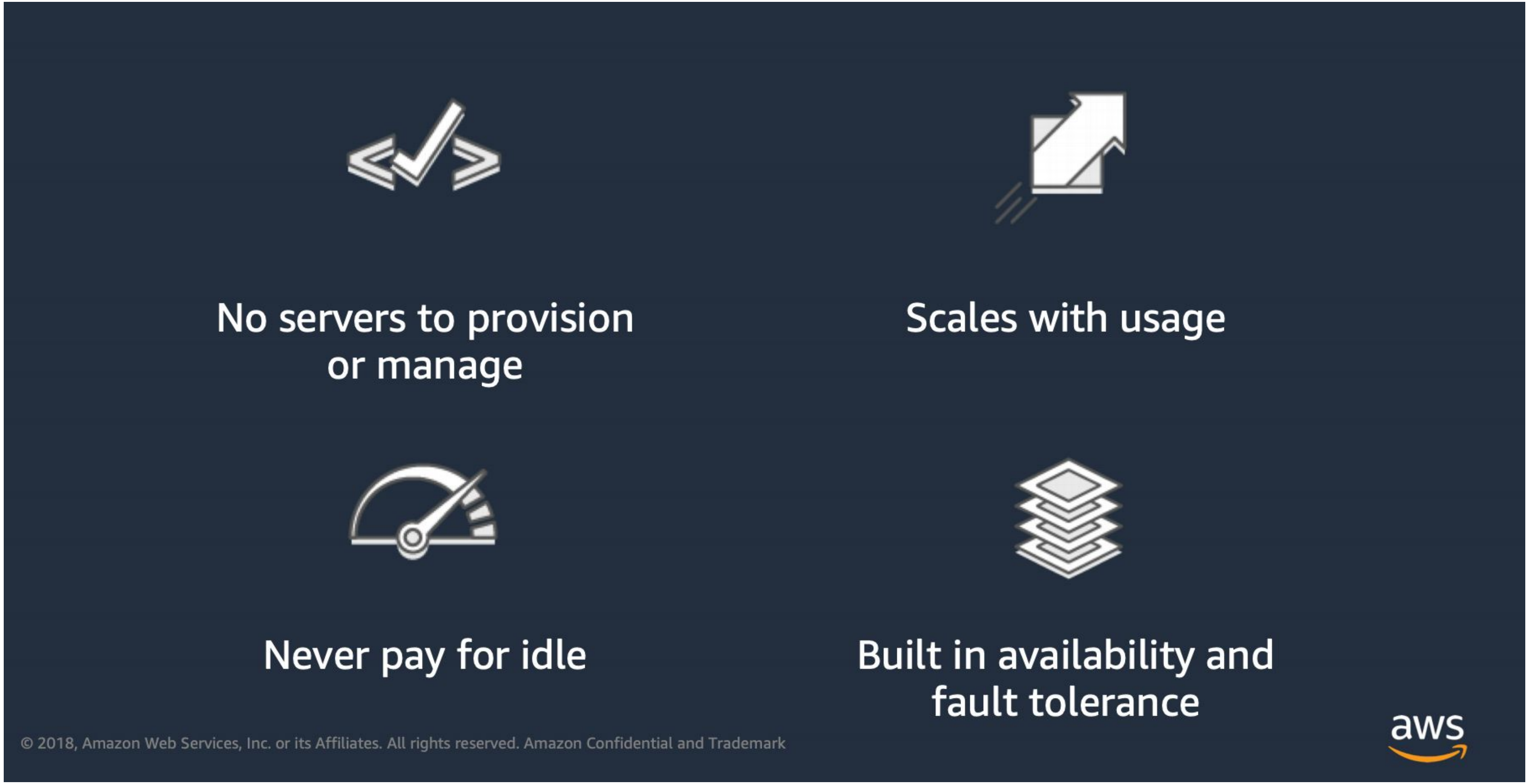
- Love to share my experience in lectures and meetups.

in https://www.linkedin.com/in/eitan-sela-6144033

What we’re going to talk today

- Quick review on serverless
- Previously on AWS Lambda
- Anatomy of a Lambda function
- Introducing AWS Lambda Layers
- Demo
- Few ideas on best practices

What are the benefits of serverless?



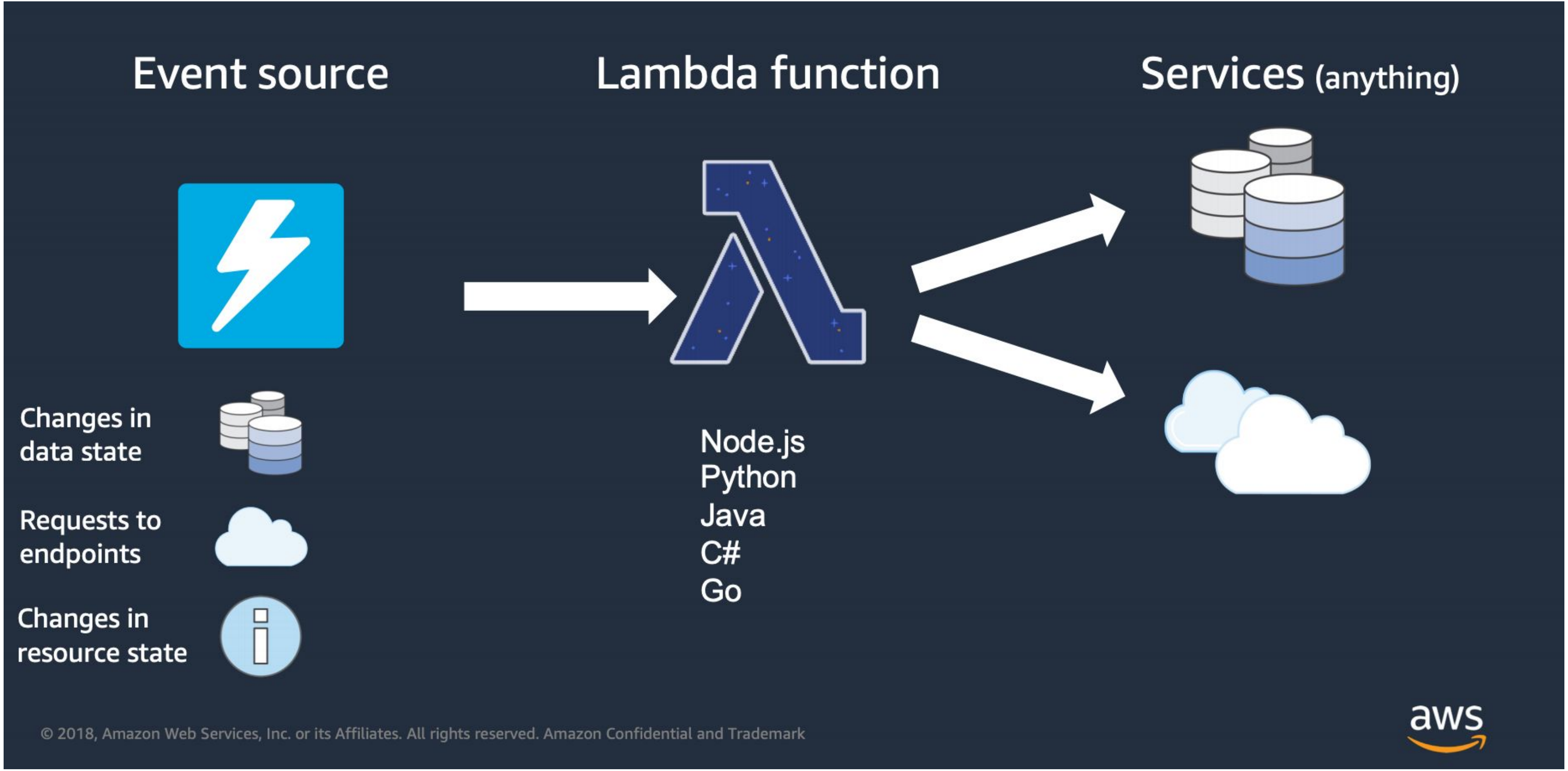
No servers to provision or manage

Scales with usage

Never pay for idle

Built in availability and fault tolerance

Serverless Applications



Event source

Lambda function

Services (anything)

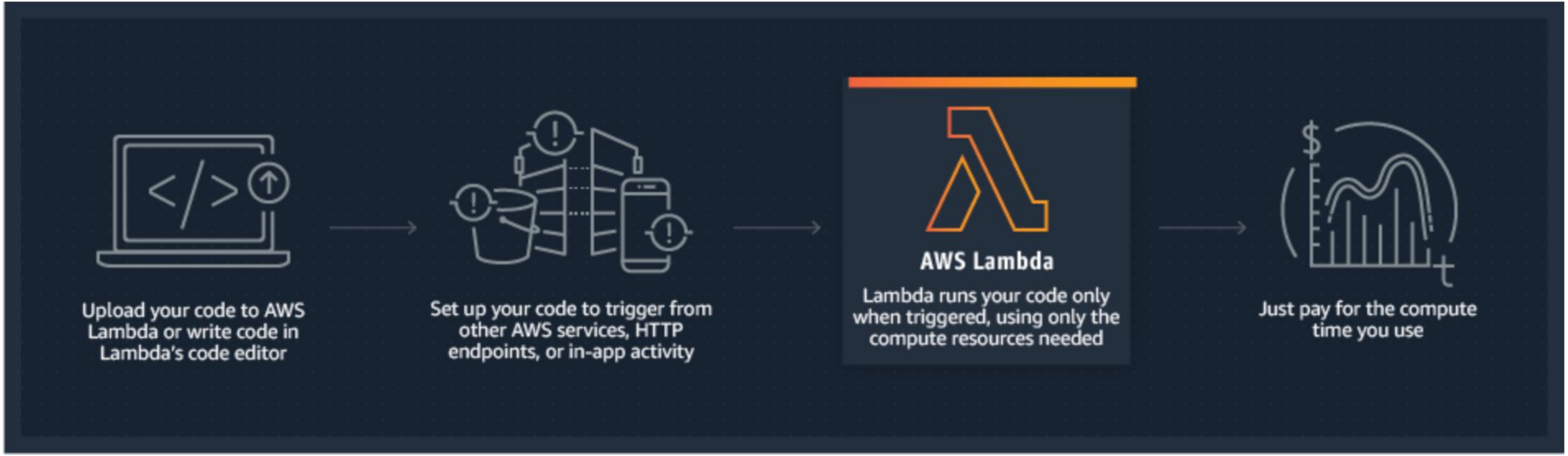
Changes in data state

Requests to endpoints

Changes in resource state

Node.js
Python
Java
C#
Go

How it works



Upload your code to AWS Lambda or write code in Lambda's code editor

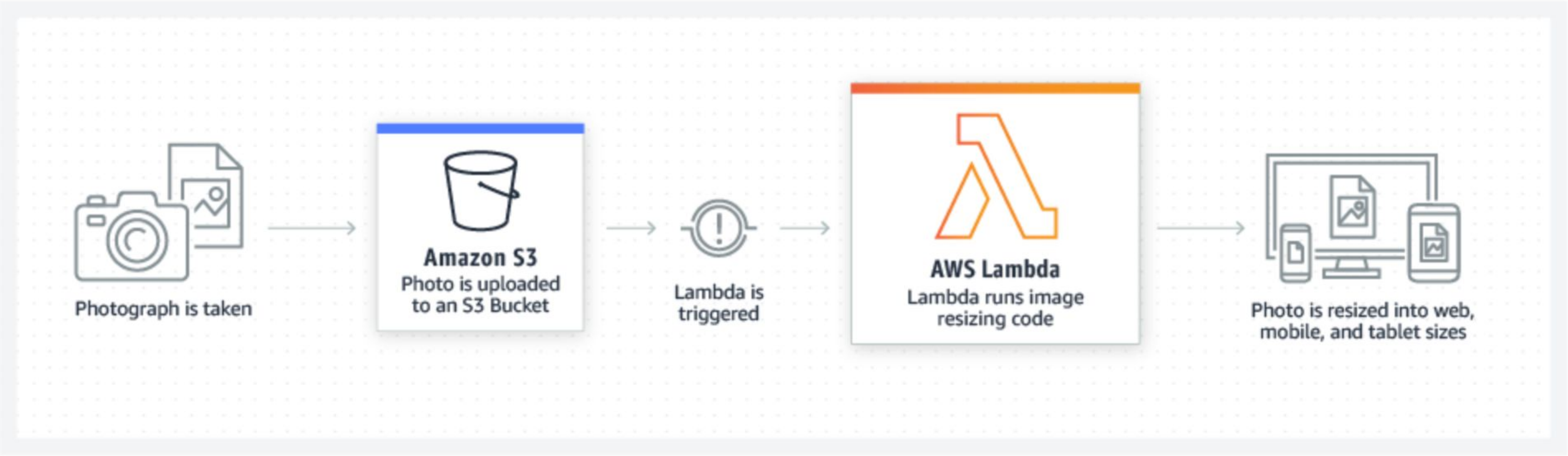
Set up your code to trigger from other AWS services, HTTP endpoints, or in-app activity

AWS Lambda

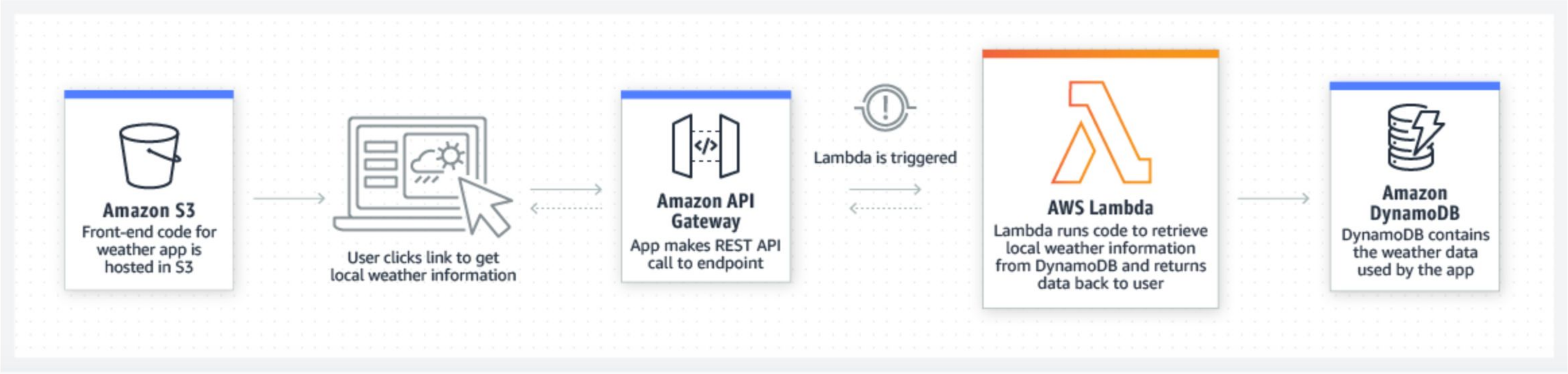
Lambda runs your code only when triggered, using only the compute resources needed

Just pay for the compute time you use

Use cases - REAL-TIME FILE PROCESSING


Photograph is taken
Amazon S3
Photo is uploaded to an S3 Bucket
Lambda is triggered
AWS Lambda
Lambda runs image resizing code
Photo is resized into web, mobile, and tablet sizes

Use cases - WEB APPLICATIONS



Amazon S3
Front-end code for weather app is hosted in S3

User clicks link to get local weather information

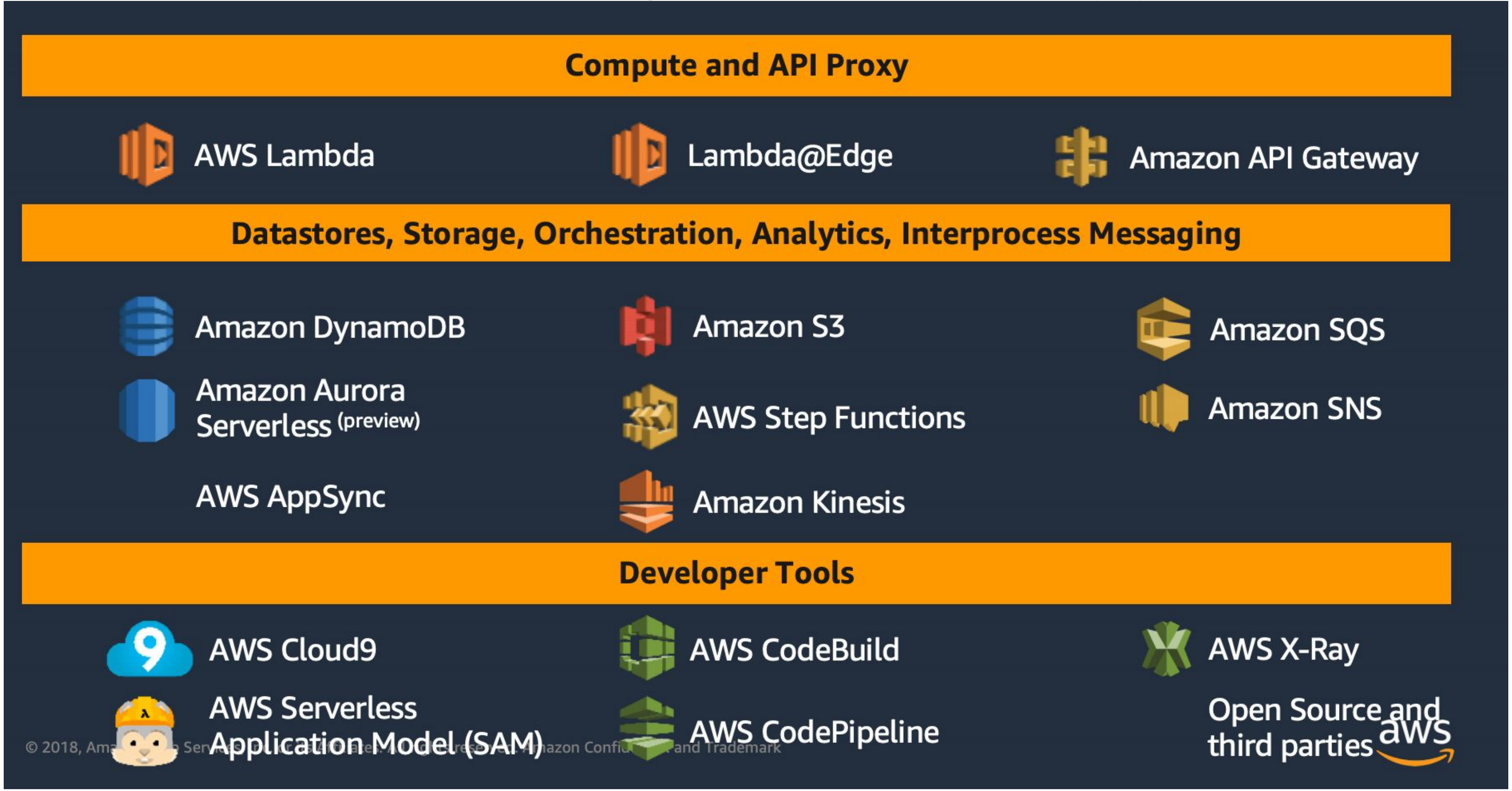
Amazon API Gateway
App makes REST API call to endpoint

Lambda is triggered

AWS Lambda
Lambda runs code to retrieve local weather information from DynamoDB and returns data back to user

Amazon DynamoDB
DynamoDB contains the weather data used by the app

Services for Building Serverless Applications



Compute and API Proxy

- AWS Lambda
- Lambda@Edge
- Amazon API Gateway

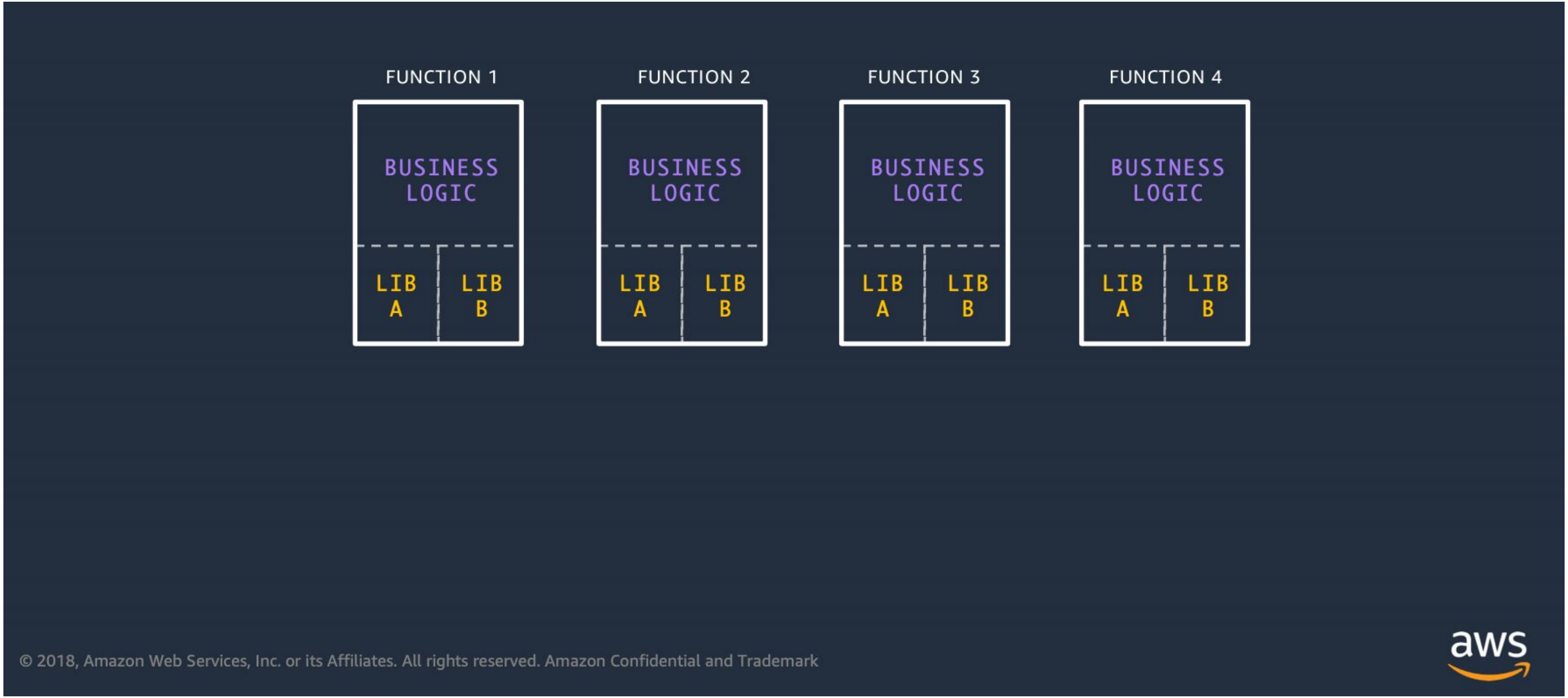
Datastores, Storage, Orchestration, Analytics, Interprocess Messaging

- Amazon DynamoDB
- Amazon Aurora Serverless (preview)
- AWS AppSync
- Amazon S3
- AWS Step Functions
- Amazon Kinesis
- Amazon SQS
- Amazon SNS

Developer Tools

- AWS Cloud9
- AWS Serverless Application Model (SAM)
- AWS CodeBuild
- AWS CodePipeline
- AWS X-Ray
- Open Source and third parties

Previously on AWS Lambda



FUNCTION 1

BUSINESS LOGIC

LIB A

LIB B

FUNCTION 2

BUSINESS LOGIC

LIB A

LIB B

FUNCTION 3

BUSINESS LOGIC

LIB A

LIB B

FUNCTION 4

BUSINESS LOGIC

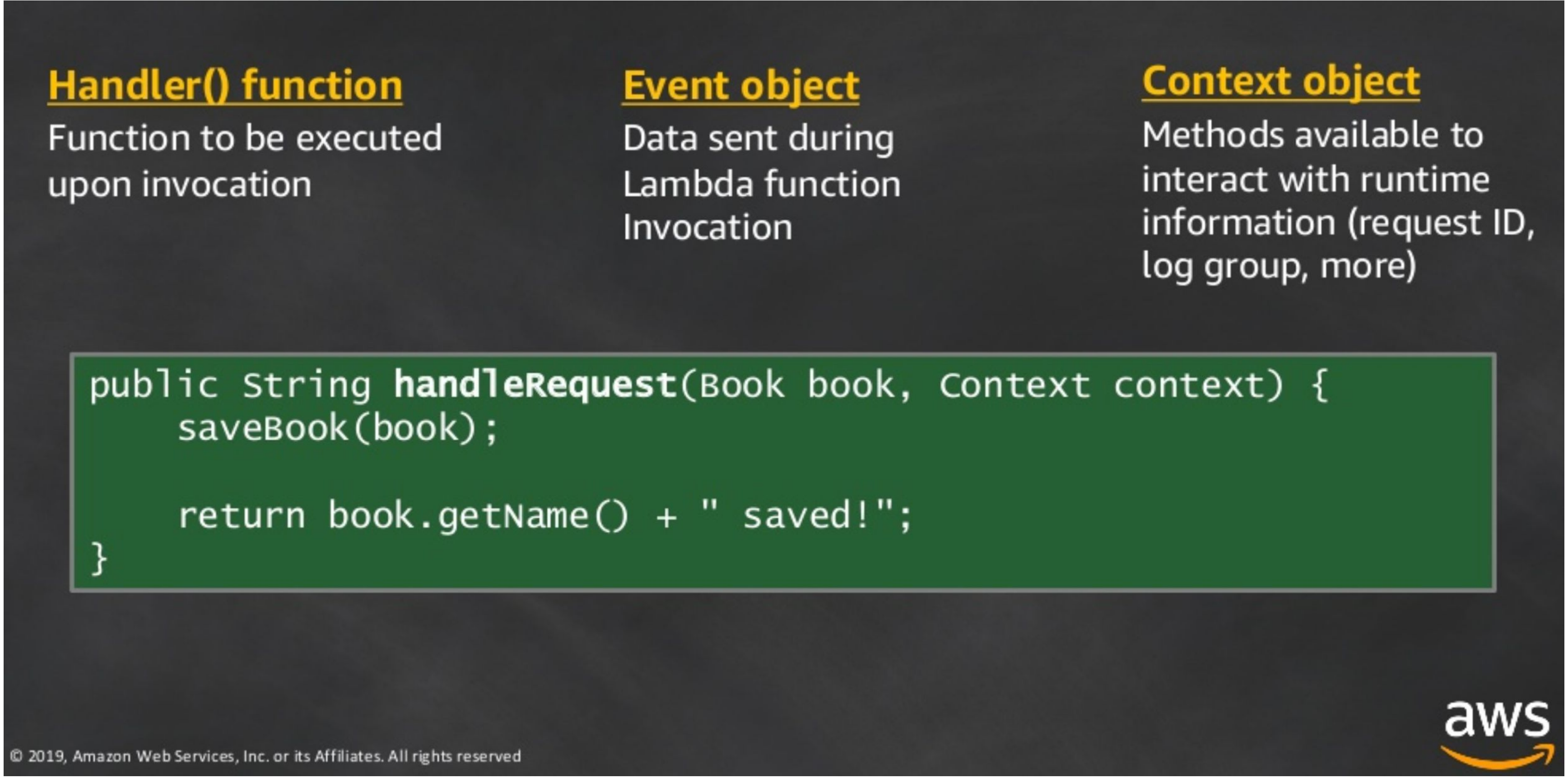
LIB A

LIB B

How?

- How about common functions?
- How to separate lib, modules, frameworks?
- How to share common libs?

Anatomy of a Lambda function



Handler() function

Function to be executed upon invocation

Event object

Data sent during Lambda function Invocation

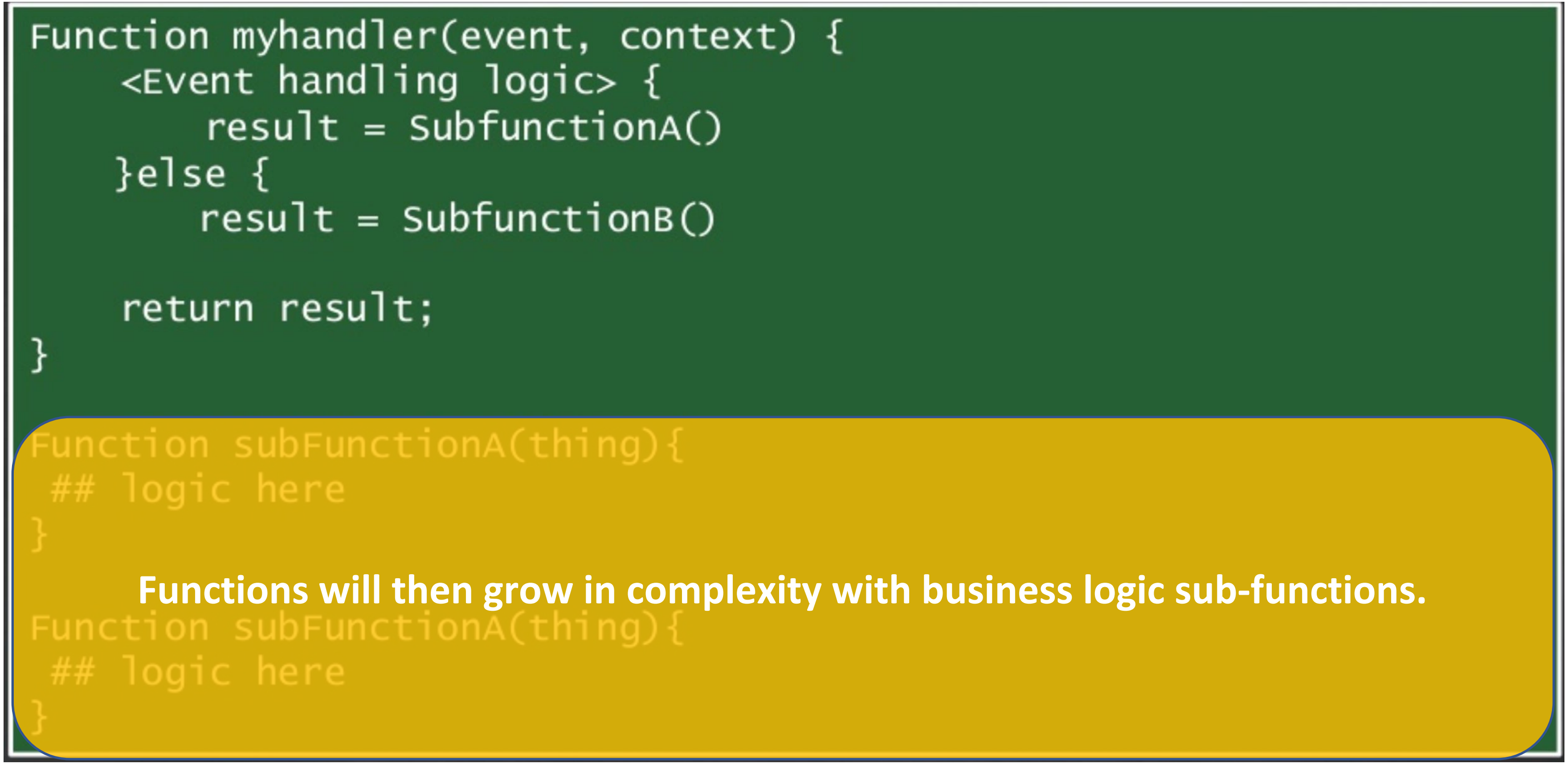
Context object

Methods available to interact with runtime information (request ID, log group, more)

```
public String handleRequest(Book book, Context context) {
    saveBook(book);

    return book.getName() + " saved!";
}
```

Anatomy of a Lambda function



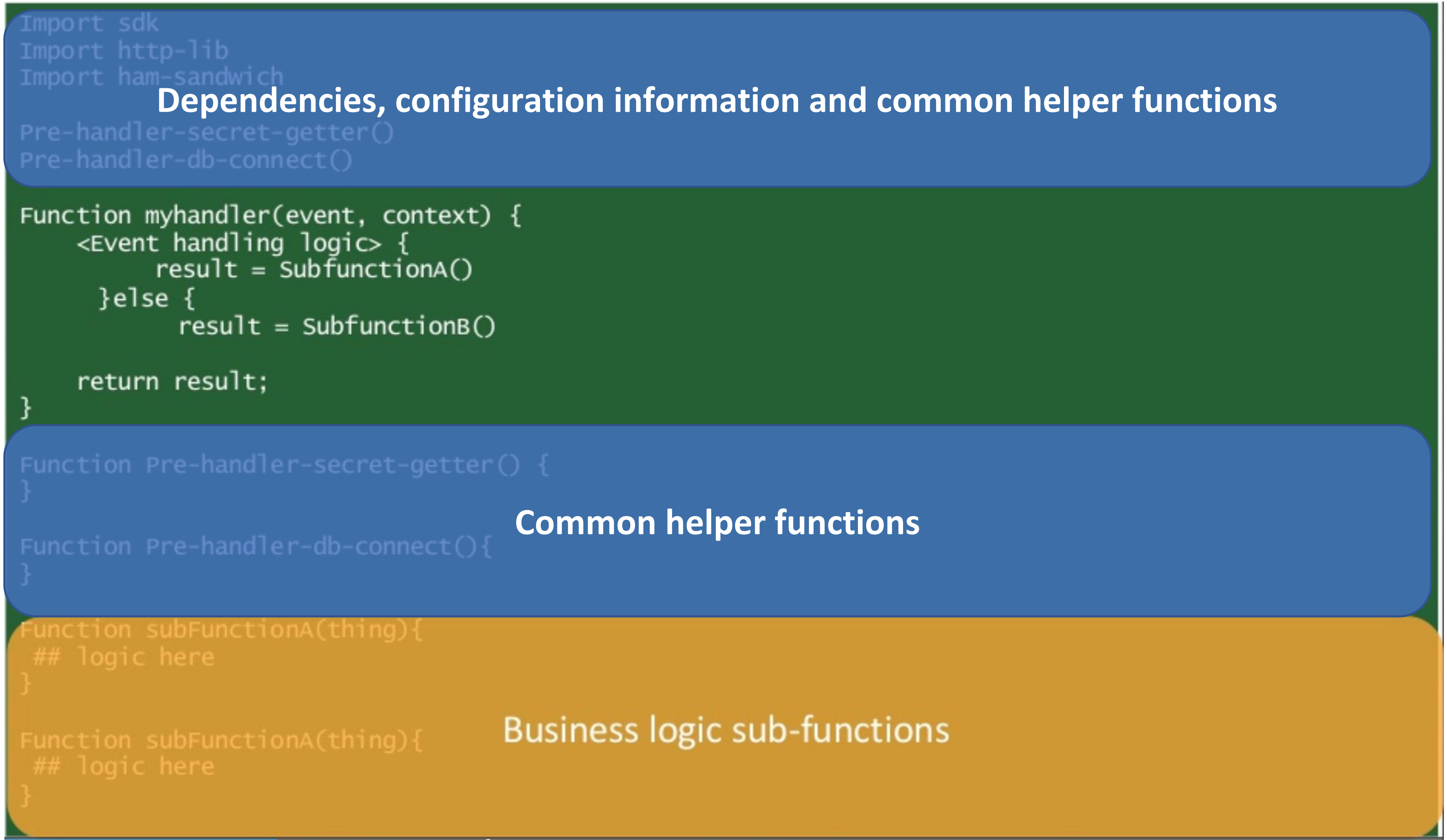
```
Function myhandler(event, context) {
    <Event handling logic> {
        result = SubfunctionA()
    }else {
        result = SubfunctionB()

    return result;
}

Function subFunctionA(thing){
 ## logic here
}

Function subFunctionA(thing){
 ## logic here
}
```

Functions will then grow in complexity with business logic sub-functions.



```
Import sdk
Import http-lib
Import ham-sandwich

Pre-handler-secret-getter()
Pre-handler-db-connect()
```

Dependencies, configuration information and common helper functions

```
Function myhandler(event, context) {
    <Event handling logic> {
          result = SubfunctionA()
      }else {
           result = SubfunctionB()

    return result;
}
```

```
Function Pre-handler-secret-getter() {
}

Function Pre-handler-db-connect(){
}
```

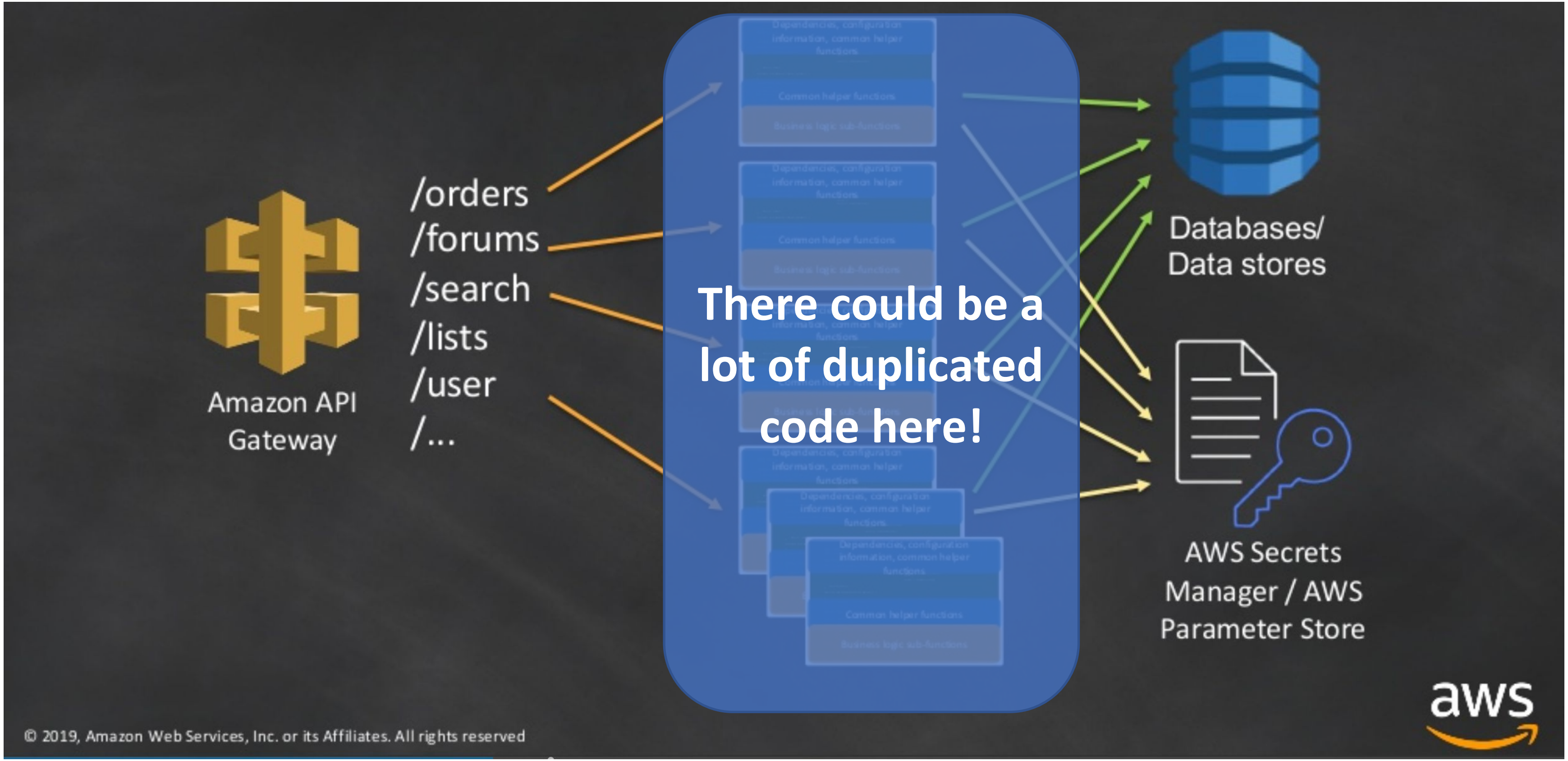
Common helper functions

```
Function subFunctionA(thing){
 ## logic here
}

Function subFunctionA(thing){
 ## logic here
}
```

Business logic sub-functions

Anatomy of a a serverless application



/orders
/forums
/search
/lists
/user
/...

Amazon API Gateway

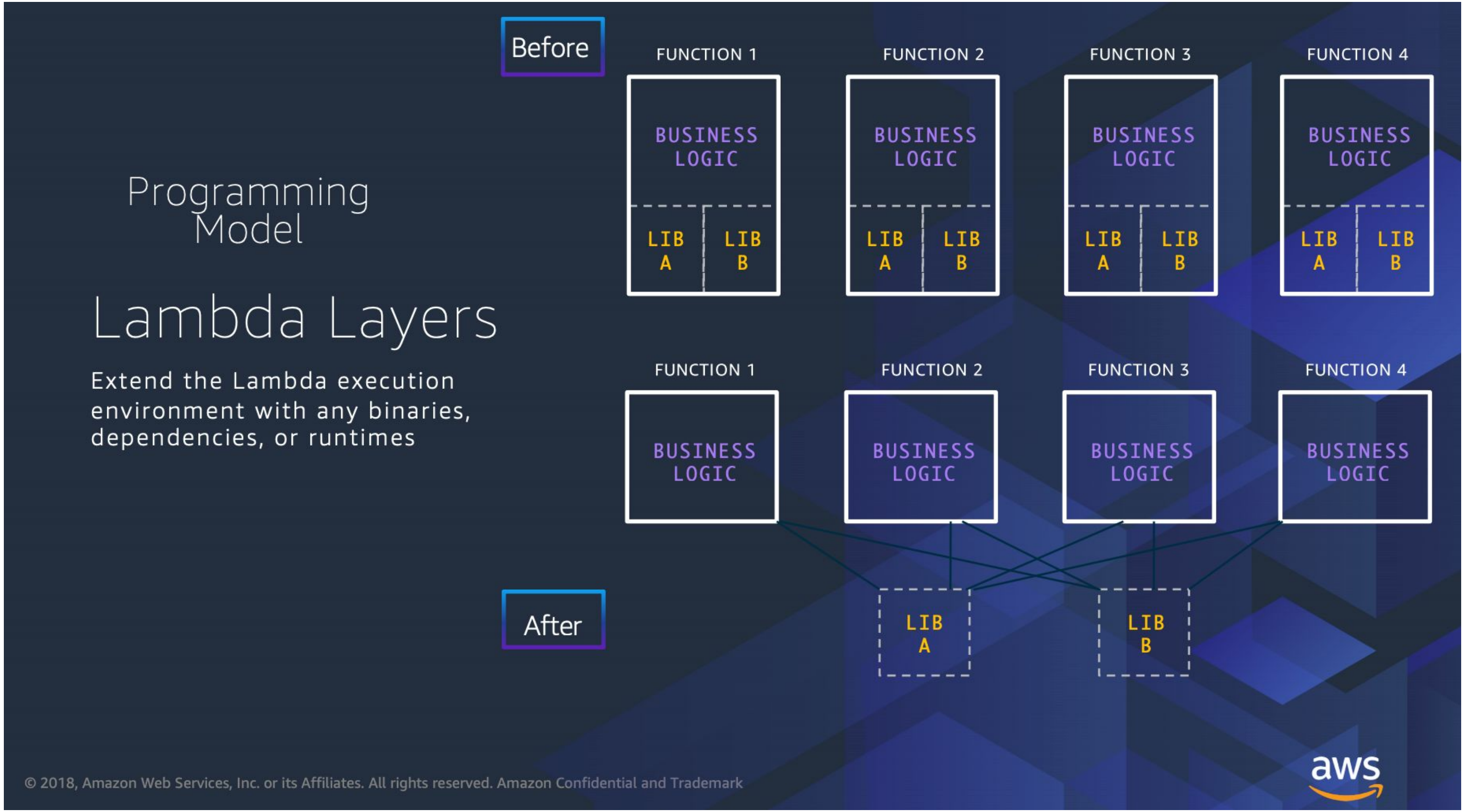
There could be a lot of duplicated code here!

Databases/ Data stores

AWS Secrets Manager / AWS Parameter Store

Dependencies, configuration information, common helper functions
Common helper functions
Business logic sub-functions
aws

Introducing AWS Lambda Layers



Programming Model

Lambda Layers

Extend the Lambda execution environment with any binaries, dependencies, or runtimes

Before

FUNCTION 1

BUSINESS LOGIC

LIB A

LIB B

FUNCTION 2

BUSINESS LOGIC

LIB A

LIB B

FUNCTION 3

BUSINESS LOGIC

LIB A

LIB B

FUNCTION 4

BUSINESS LOGIC

LIB A

LIB B

After

FUNCTION 1

BUSINESS LOGIC

FUNCTION 2

BUSINESS LOGIC

FUNCTION 3

BUSINESS LOGIC

FUNCTION 4

BUSINESS LOGIC

LIB A

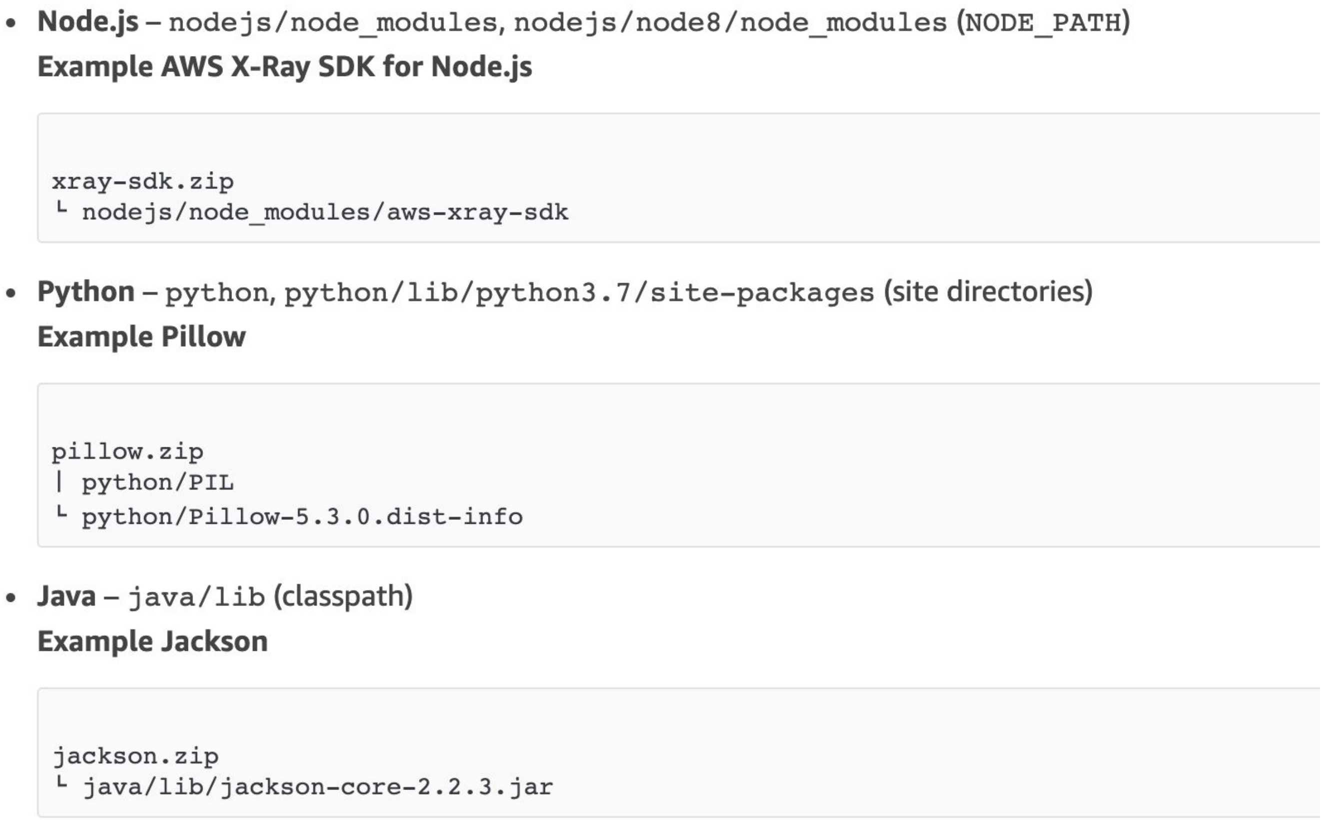
LIB B

When you should use Lambda Layers?

- To share code between functions in the same project, use shared modules in the same repo. Shared modules are put inside a dedicated folder, depends on the runtime (Java, Python, Node.js, etc).
- To share code between functions across projects, publish the shared code as libraries to package managers such as NPM, pip or Gradle.
- To share code (or use) from/to another AWS account:
  - Datadog's Lambda Layer
  - Epsagon Node Layer
  - Epsagon Python Layer

* For other cool Lambda Layers resources: https://github.com/mthenw/awesome-layers

Including Library Dependencies in a Layer



- **Node.js** – `nodejs/node_modules`, `nodejs/node8/node_modules` (`NODE_PATH`)

  Example AWS X-Ray SDK for Node.js

```
xray-sdk.zip
└ nodejs/node_modules/aws-xray-sdk
```

- **Python** – `python`, `python/lib/python3.7/site-packages` (site directories)

  Example Pillow

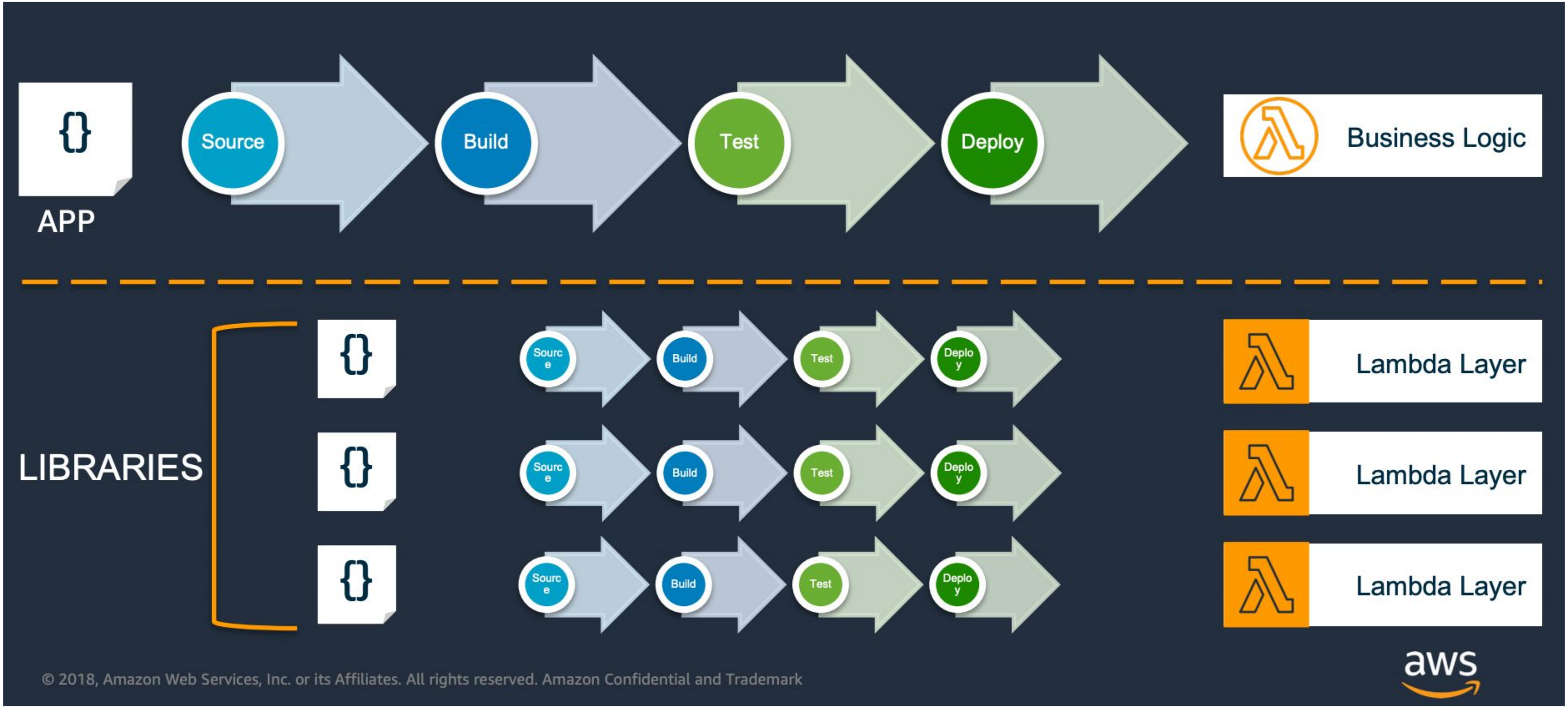
```
pillow.zip
│ python/PIL
└ python/Pillow-5.3.0.dist-info
```

- **Java** – `java/lib` (classpath)

  Example Jackson

```
jackson.zip
└ java/lib/jackson-core-2.2.3.jar
```

CI/CD for App & Dependencies



Demo

https://github.com/eitansela/lmabda-layers-demo

Demo - Steps

- Write a Lambda function code
- Package Lambda function
- Write a Lambda layer code
- Package Lambda layer
- Deploy Lambda function and Lambda layer
- Attached a layer to function
- Call a method
- Verify the results

Few ideas on best practices

- Take advantage of Execution Context reuse to improve the performance of your function.
- Use AWS Lambda Environment Variables to pass operational parameters to your function – works even better with AWS Systems Manager Parameter Store.
- Control the dependencies in your function's deployment package.
- Minimize your deployment package size to its runtime necessities.
- Avoid using recursive code.
- For CI/CD – Use Serverless or SAM.
- Use Lambda Layers.

Q & A