serverless architectures

functions, events, cloud native services and things like that…

Dave Townsend @davetownsend

Dave Townsend

Principal Software Engineer
Innovation & Architecture
Matson, Inc

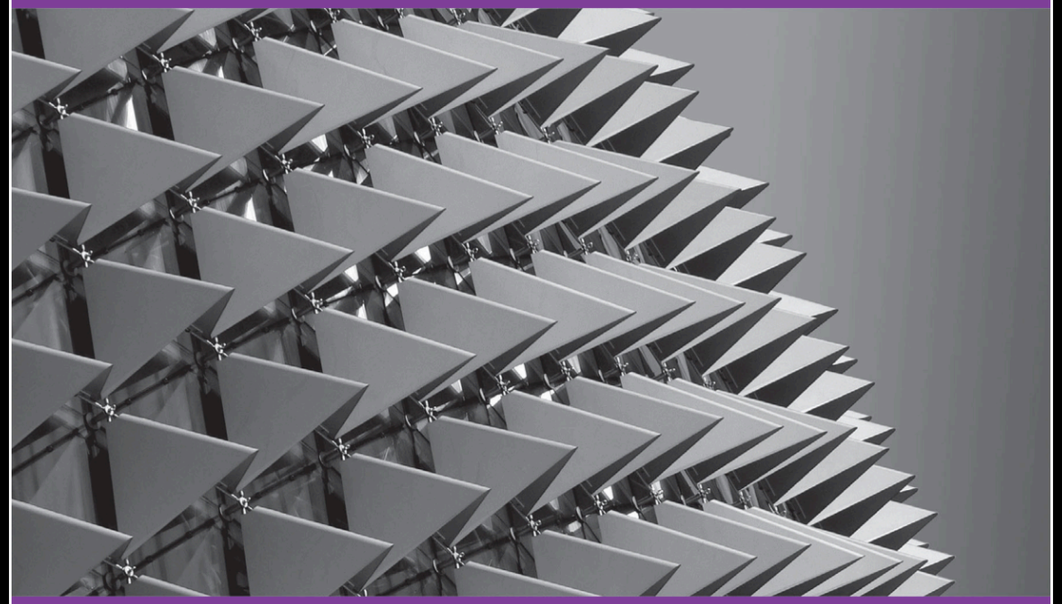
@davetownsend

what is serverless?

O'REILLY®

What is Serverless?

Understanding the Latest Advances in Cloud and Service-Based Architecture



Mike Roberts
& John Chapin

*"**Cloud** really just means **datacenterless,** there's a datacenter somewhere but we don't care."*

– Adrian Cockcroft

serverless

we don't have to think about the servers

also…

A Serverless solution is one that costs you nothing to run if nobody is using it … excluding data storage costs.

— Paul Johnston

why serverless?

(brief) history of compute

(brief) history of compute

physical servers

(brief) history of compute

VMs (on-prem –> cloud)

physical servers

(brief) history of compute

containers

VMs (on-prem –> cloud)

physical servers

(brief) history of compute

serverless

containers

VMs (on-prem –> cloud)

physical servers

(brief) history of compute

serverless

containers

VMs (on-prem –> cloud)

physical servers

(brief) history of compute

serverless

value line

containers

VMs (on-prem –> cloud)

physical servers

instances are not *really* cloud

instances are not *really* cloud

instances do not take advantage of the cloud

instances are not *really* cloud

instances do not take advantage of the cloud

use the services, the services are good

S3 was serverless before serverless was cool.

DynamoDB

SNS

Athena

Lambda

serverless is about
using **managed services**

S3

Kinesis

API Gateway

next evolution in cloud

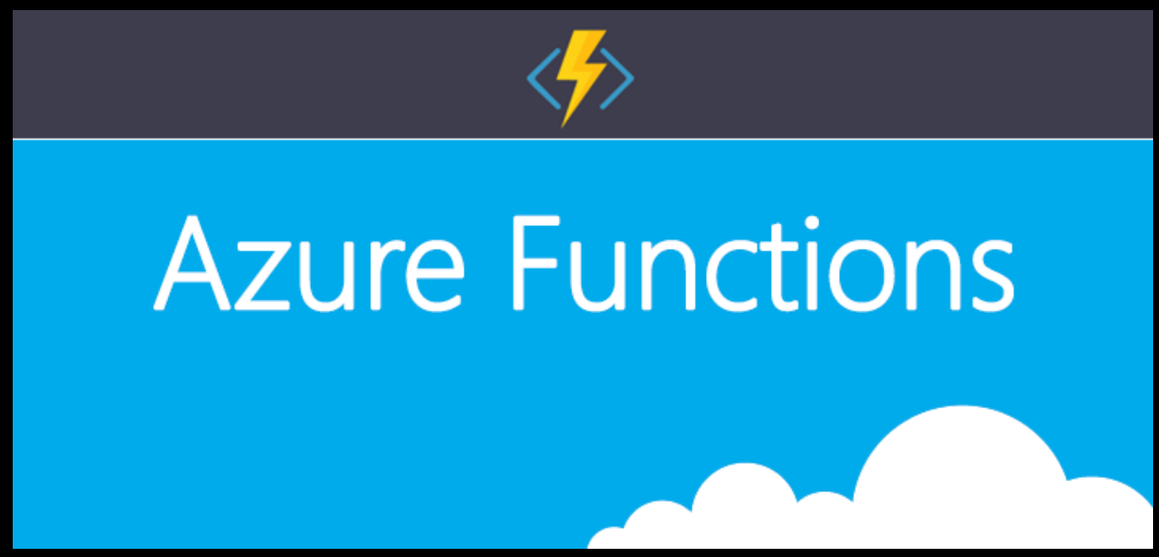
backend as-a service & functions as-a service

BaaS **FaaS**

Auth0

Firebase

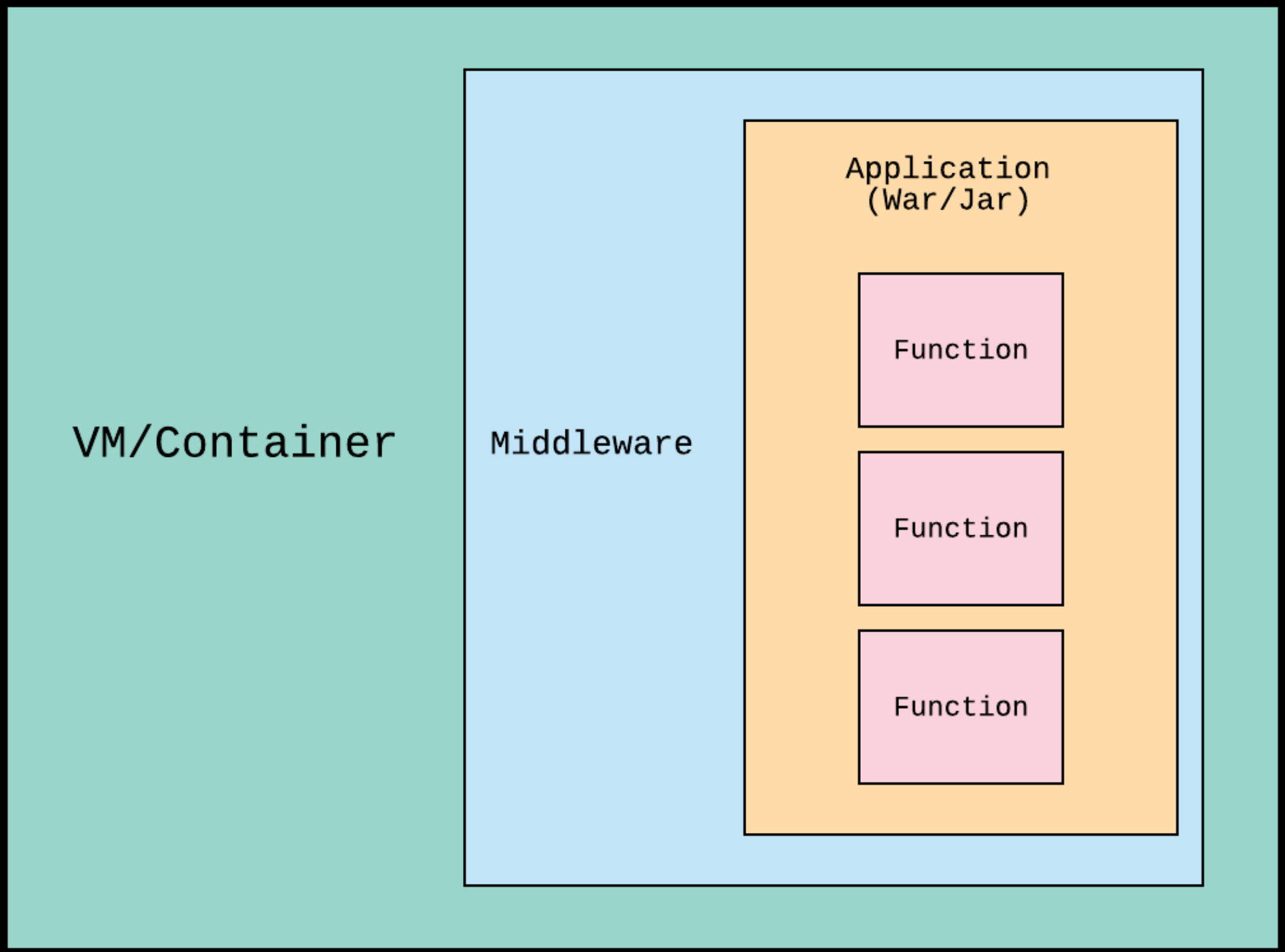
netlify



IBM OpenWhisk

Google Cloud Functions

FaaS


VM/Container
Middleware
Application
(War/Jar)
Function
Function
Function

VM/Container

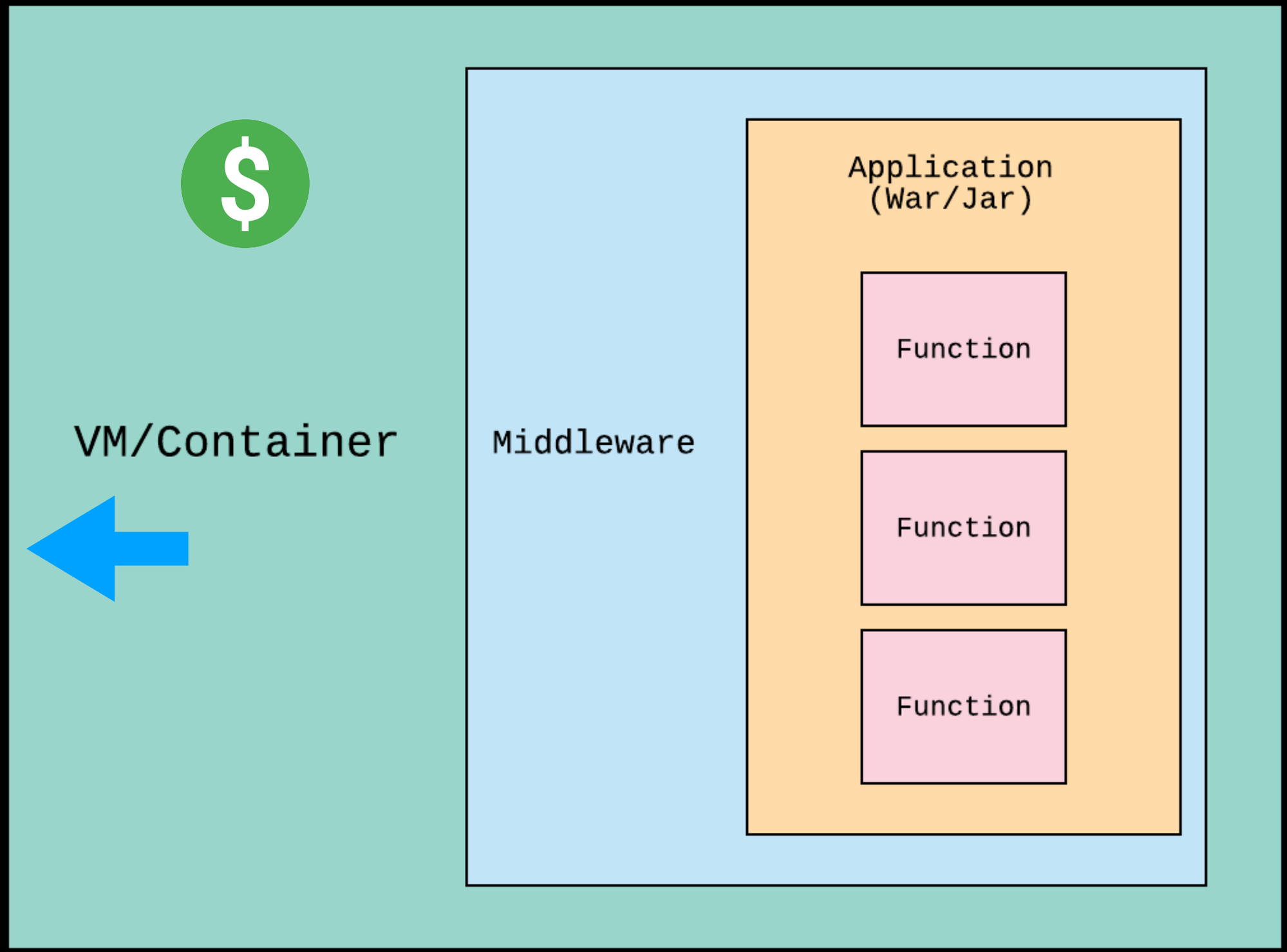
Middleware

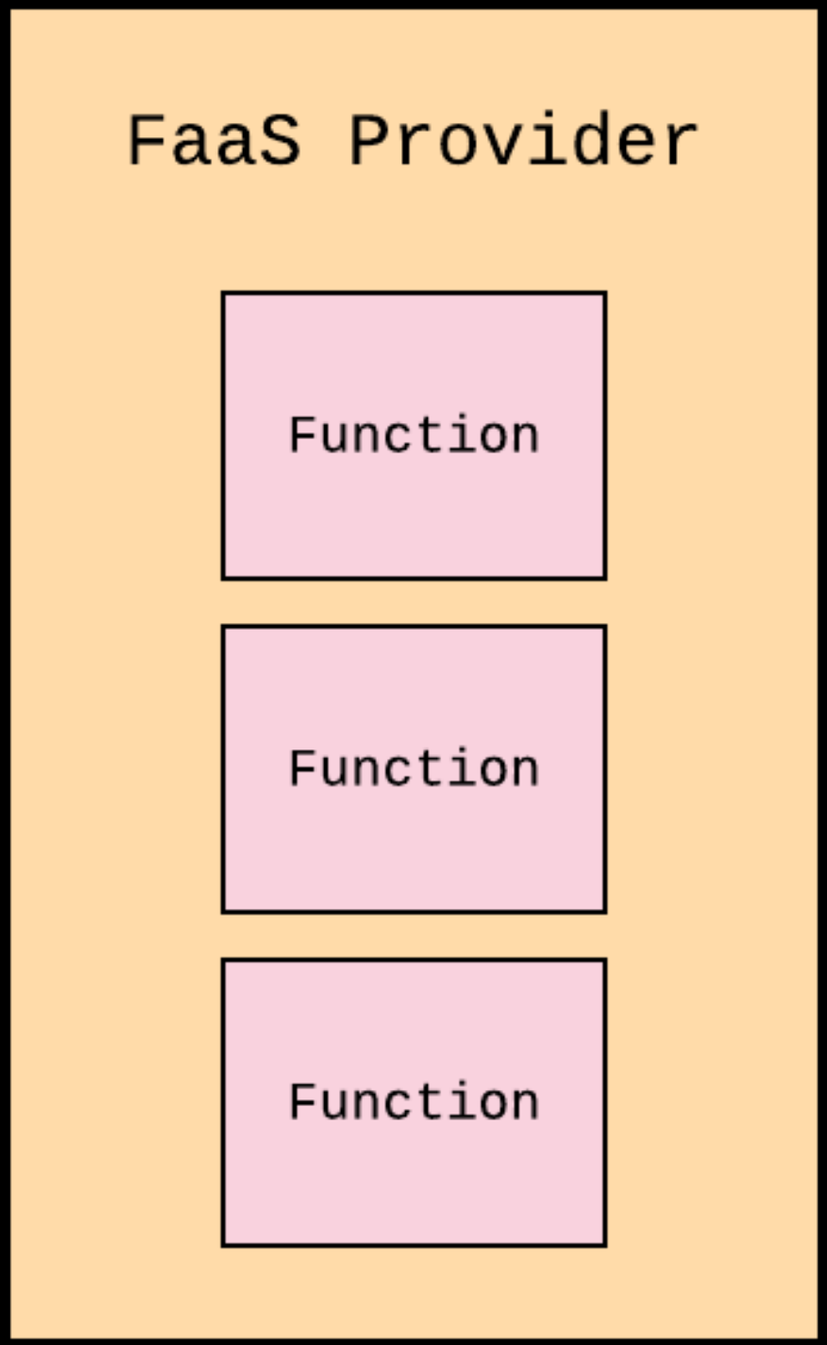
Application
(War/Jar)

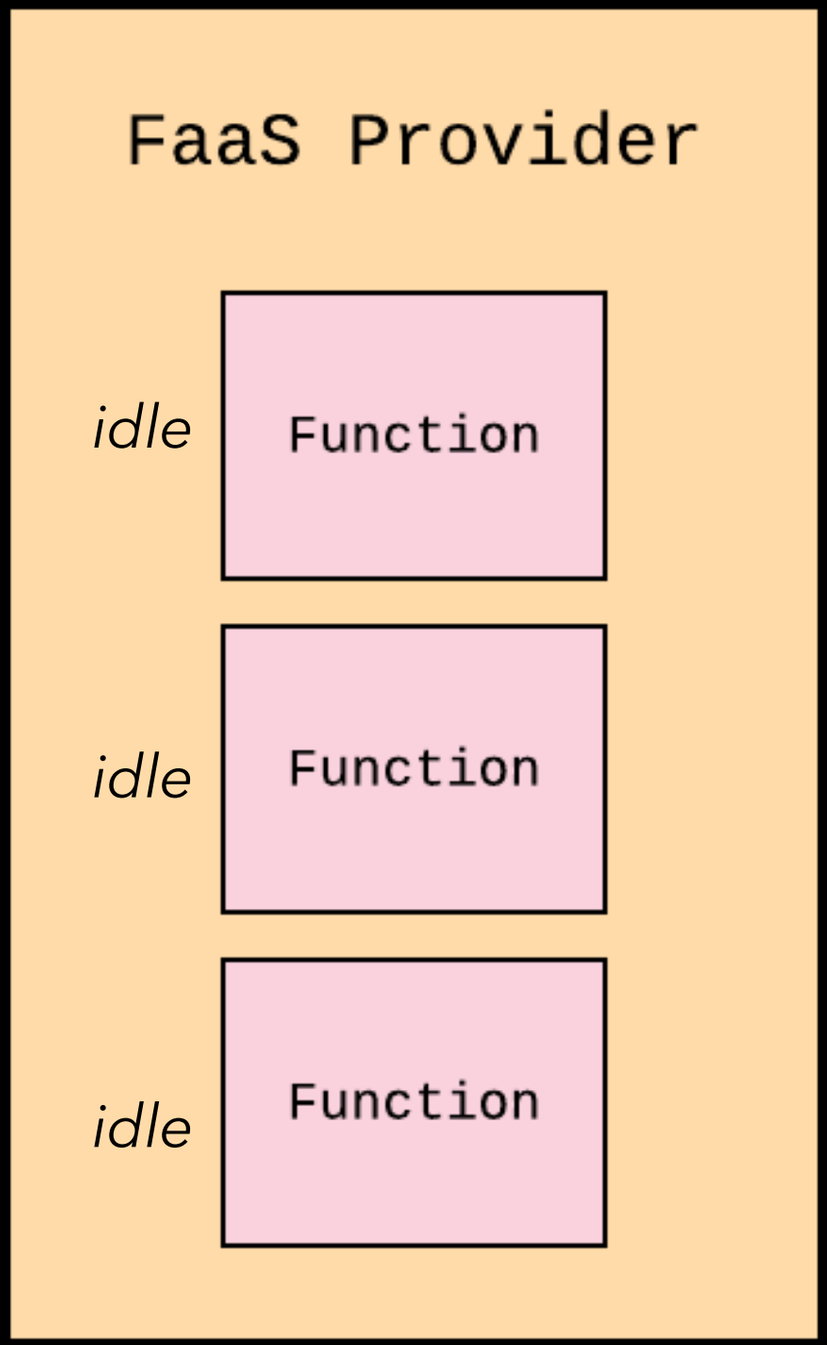
Function

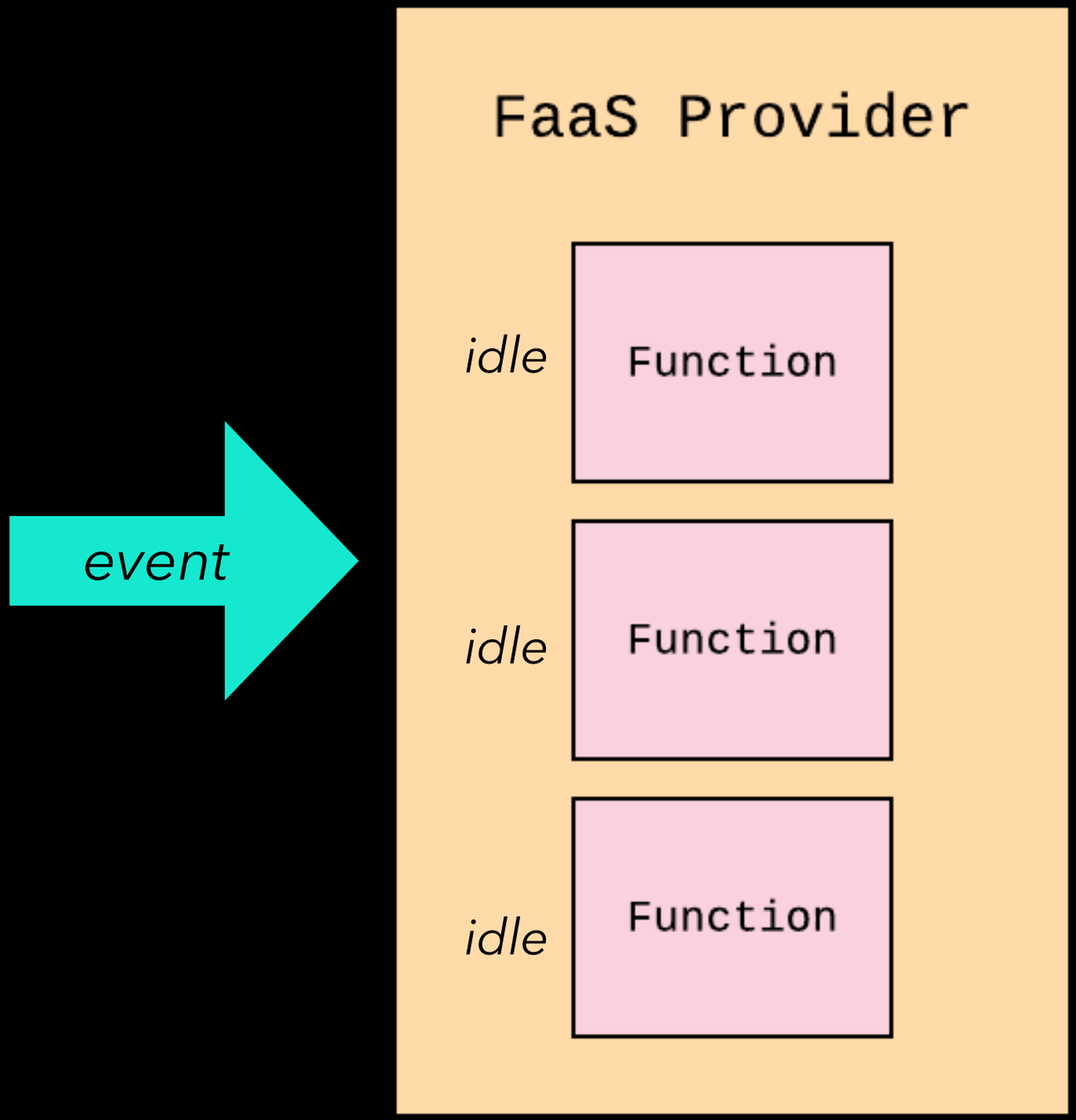
Function

Function


$
VM/Container
Middleware
Application
(War/Jar)
Function
Function
Function


FaaS Provider
Function
Function
Function


FaaS Provider
idle
Function
idle
Function
idle
Function


FaaS Provider
idle
Function
idle
Function
idle
Function
event

event

FaaS Provider

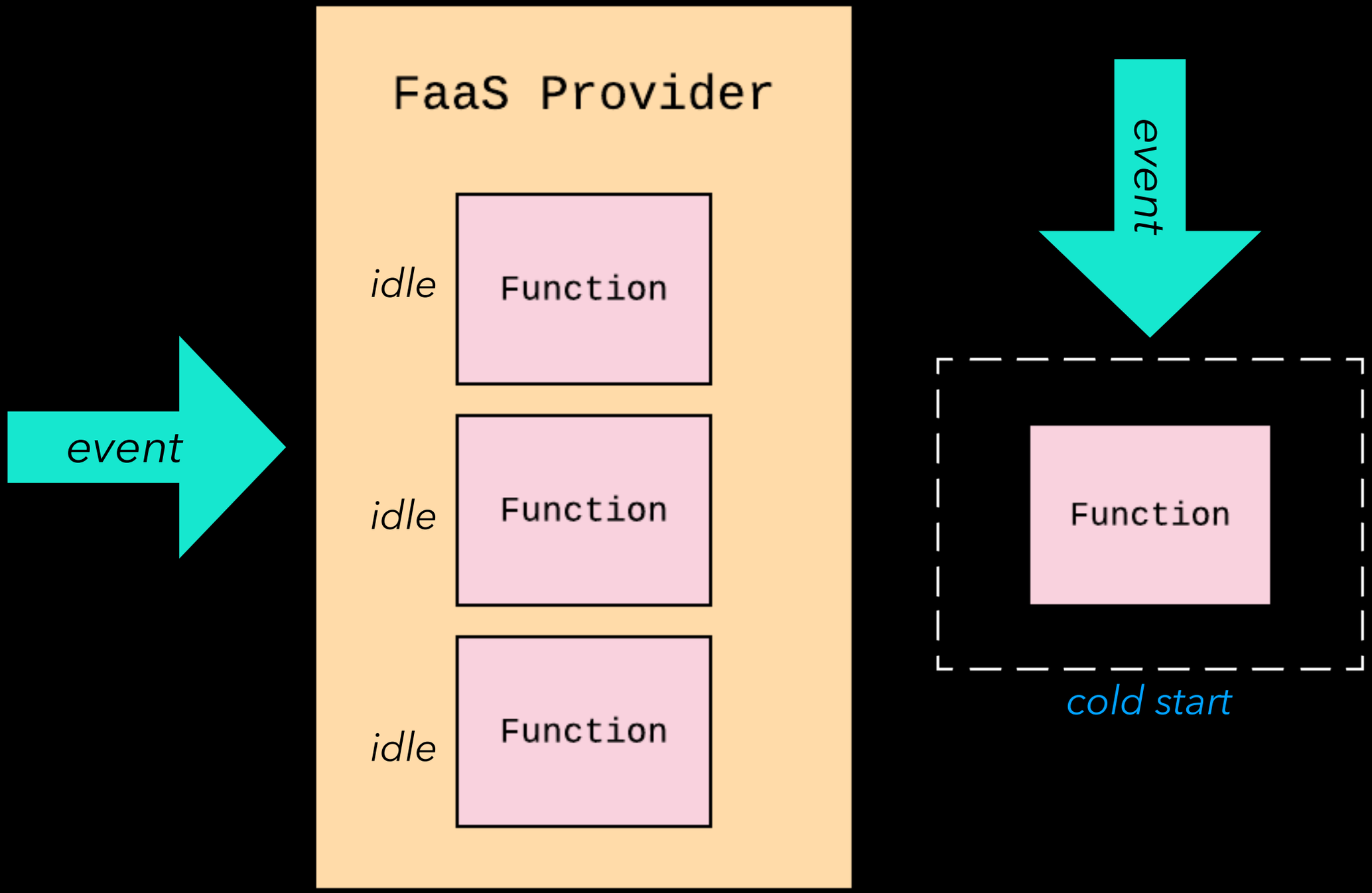
idle Function

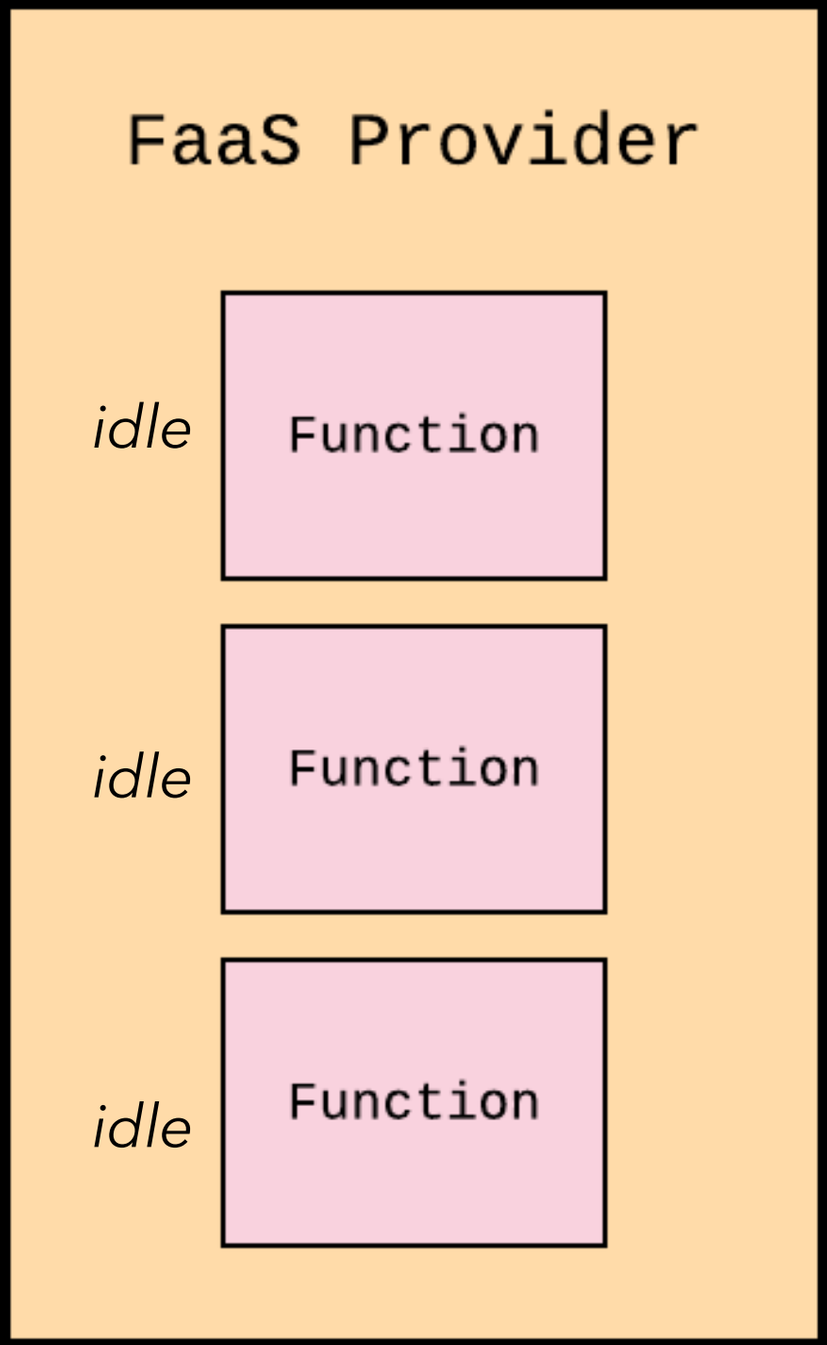
idle Function

idle Function

Function

cold start


FaaS Provider
idle
Function
idle
Function
idle
Function
event
event
Function
cold start


FaaS Provider
idle
Function
idle
Function
idle
Function

lambda language runtimes

Node.js
Java*
Python
C#

lambda language runtimes

Node.js
Java*
Python
C#

framework d'jour not needed here

lambda pricing

first 1 million requests per month are free

$0.20 per 1 million requests thereafter

$0.0000002 per request

(billed in 100ms increments)

lambda runtime config

(23 memory choices) 128MB - 1.5GB

timeout (1- 300 sec)

app logging –> CloudWatch

- cwtail!

`/tmp` gotcha!

serverless characteristics

...not limited to functions.

implicit scalability & HA

built in autoscale and provisioning

implicit HA by default

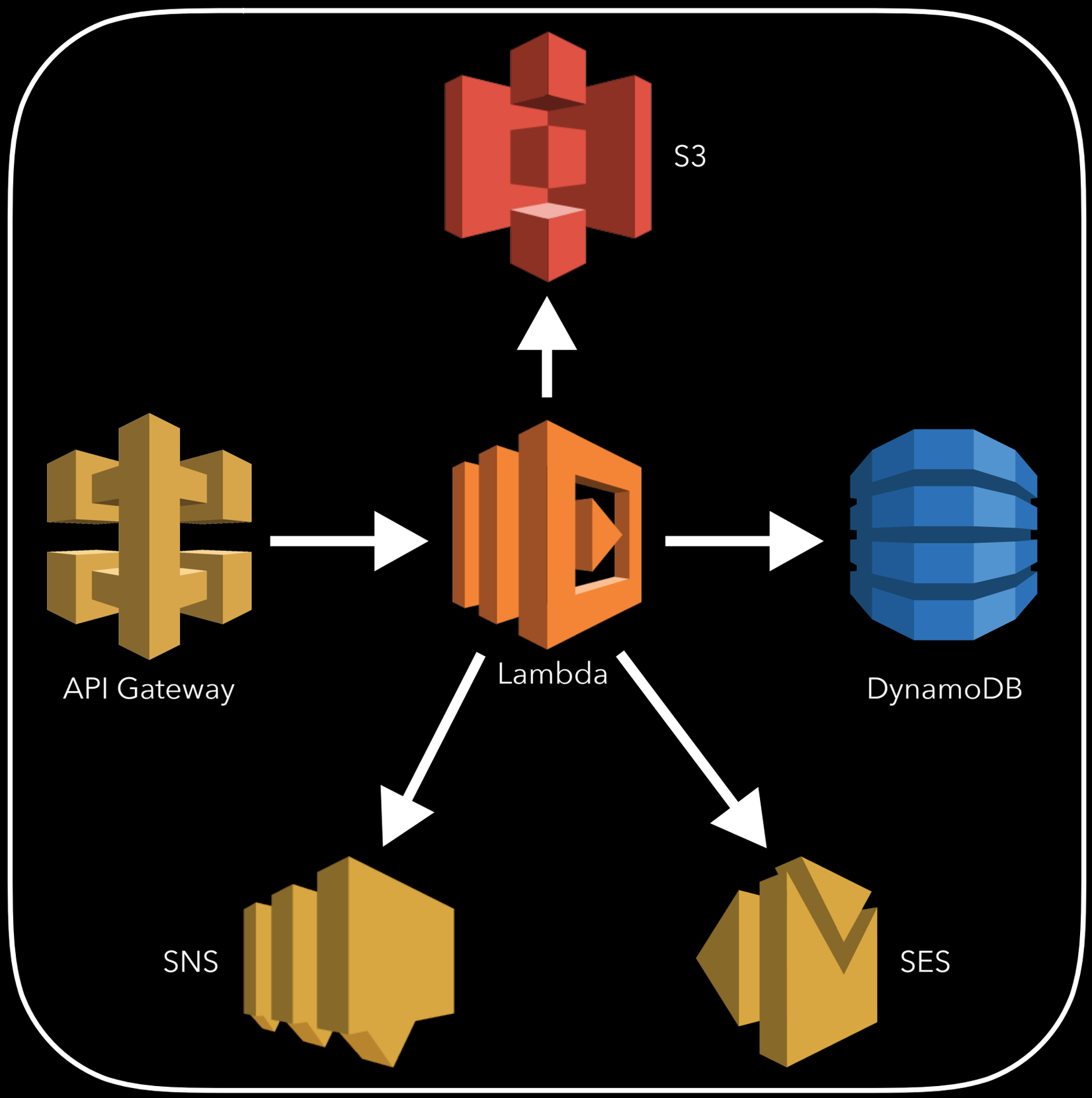
does *not* mean implicit DR

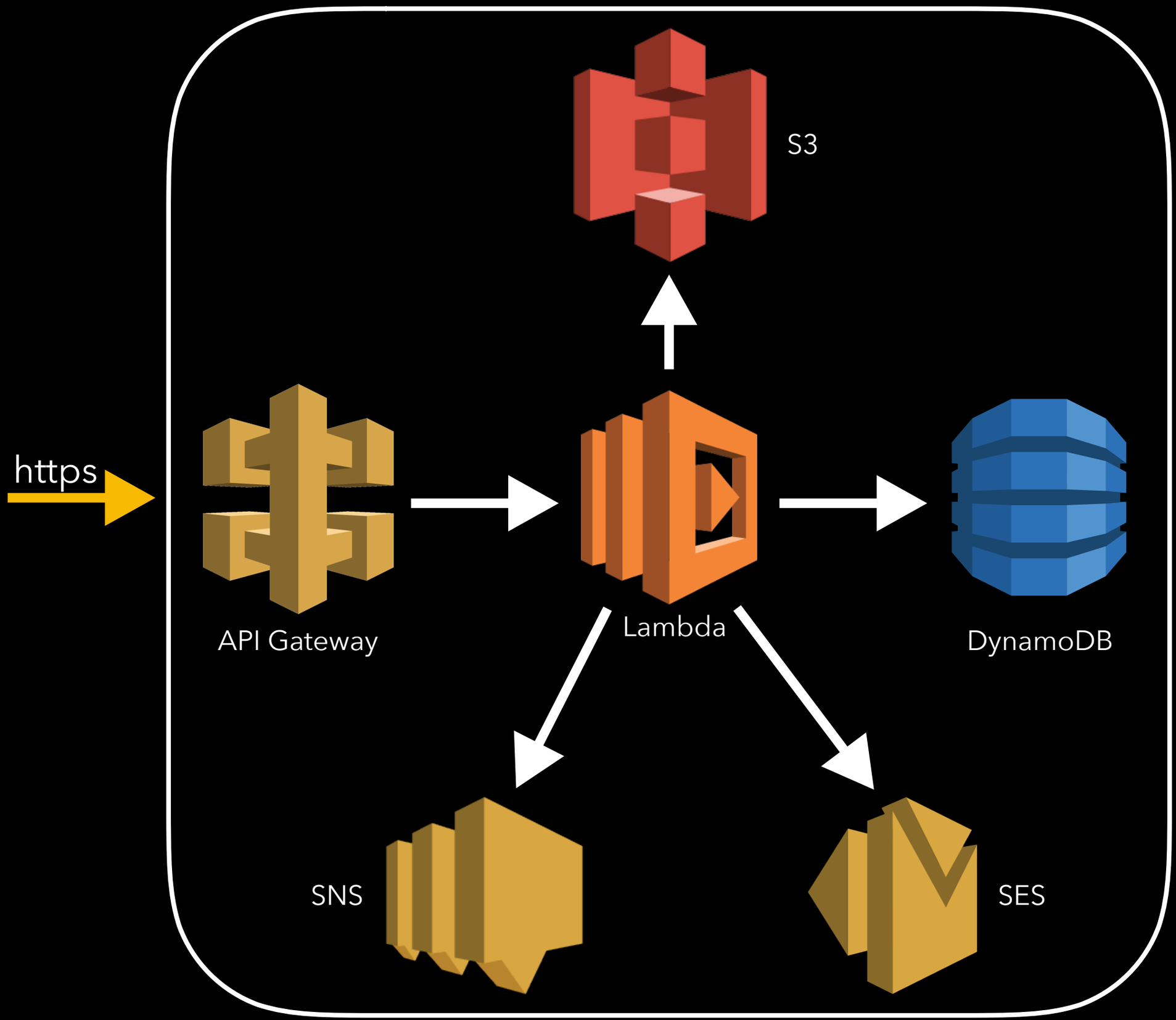
pay-per execution

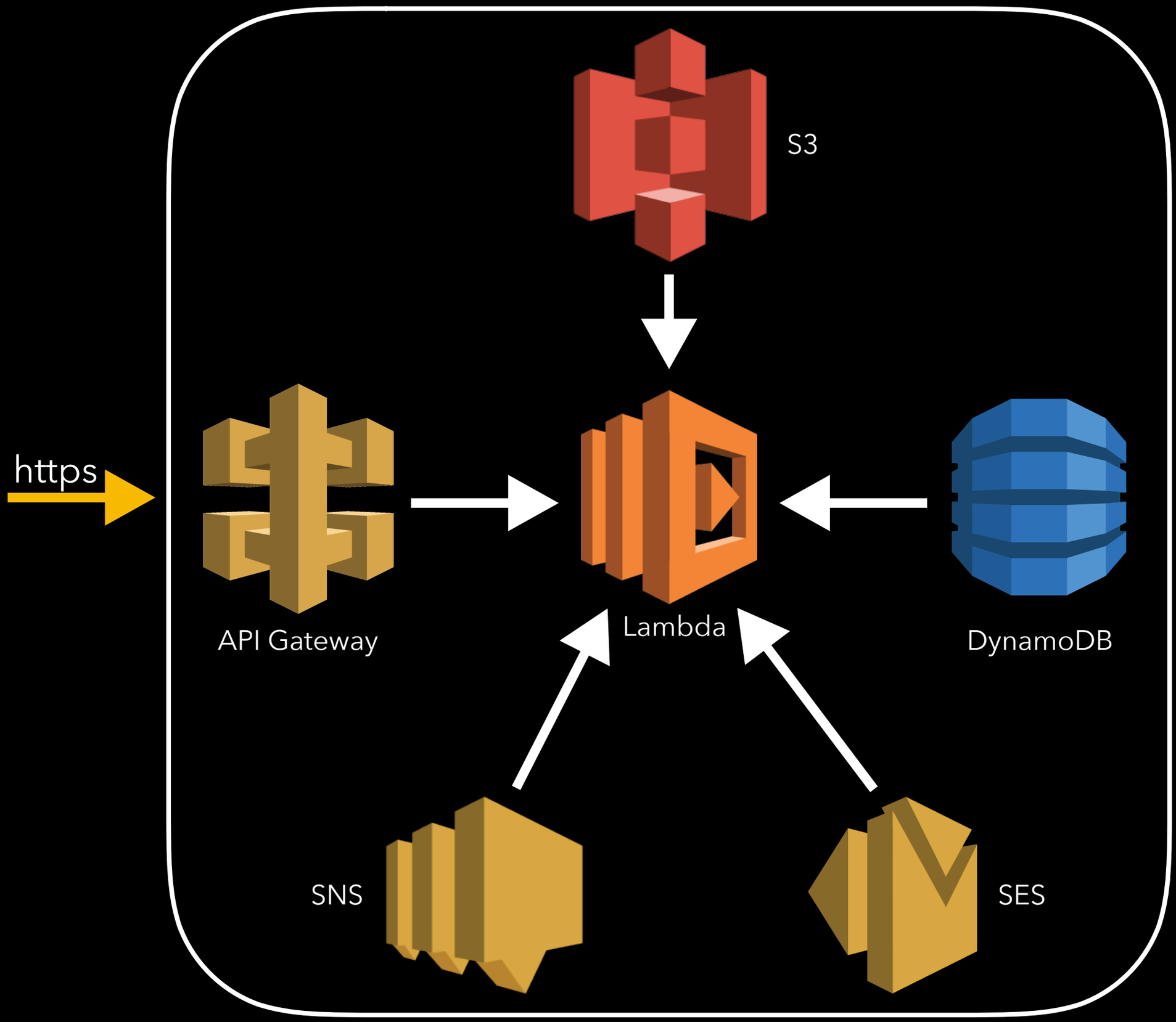
never pay for idle

billed for usage only

no/low-cost DR 😀👍


S3
API Gateway
Lambda
DynamoDB
SNS
SES


S3
https
API Gateway
Lambda
DynamoDB
SNS
SES


S3
https
API Gateway
Lambda
DynamoDB
SNS
SES

- Amazon S3
- Amazon DynamoDB
- Amazon Kinesis Streams
- Amazon Simple Notification Service
- Amazon Simple Email Service
- Amazon Cognito
- AWS CloudFormation
- Amazon CloudWatch Logs
- Amazon CloudWatch Events
- AWS CodeCommit
- Scheduled Events (powered by Amazon CloudWatch Events)
- AWS Config
- Amazon Alexa
- Amazon Lex
- Amazon API Gateway

current list (Sept 2017) of supported event sources

"Event-driven architecture will become an essential skill in supporting digital business transformation By 2018"

- Gartner

http://www.gartner.com/newsroom/id/3758164

use cases (work loads)

use cases (work loads)

not for everything….

good for *a lot* of things

questionable work loads

long running batch processes

- lambda max run-time 5min

low-latency systems

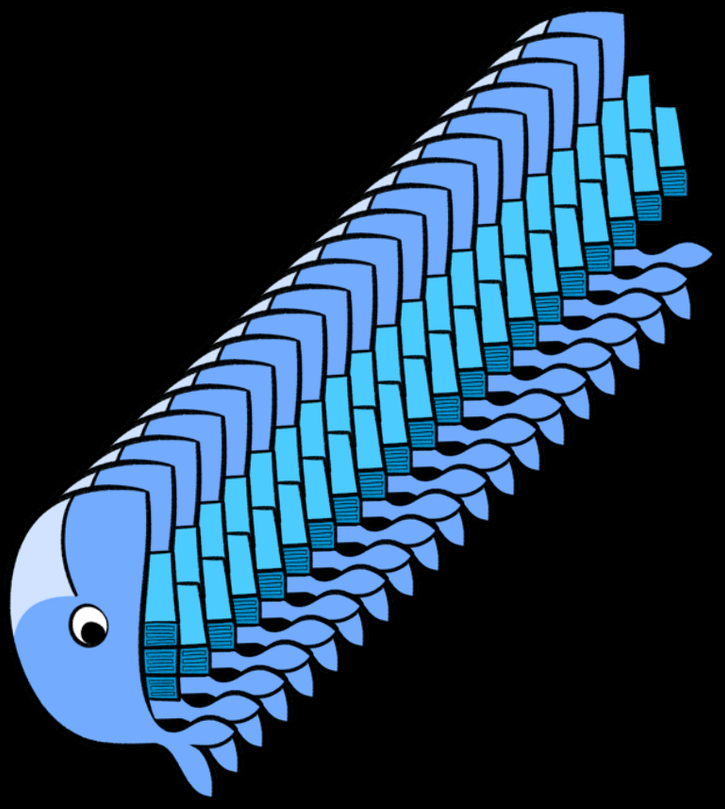
heavy continuous cpu load

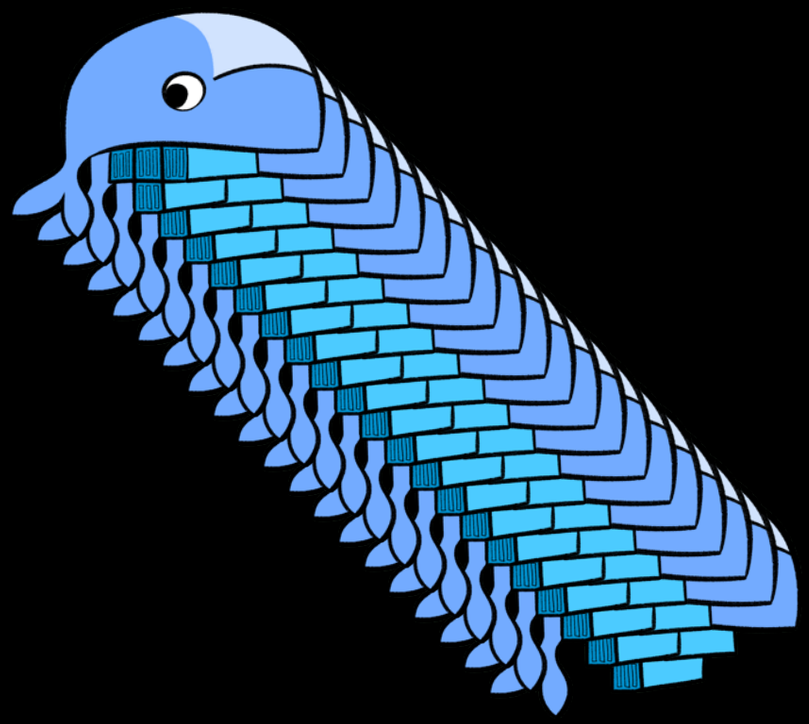
- ~ 75% is break even on lambda cost

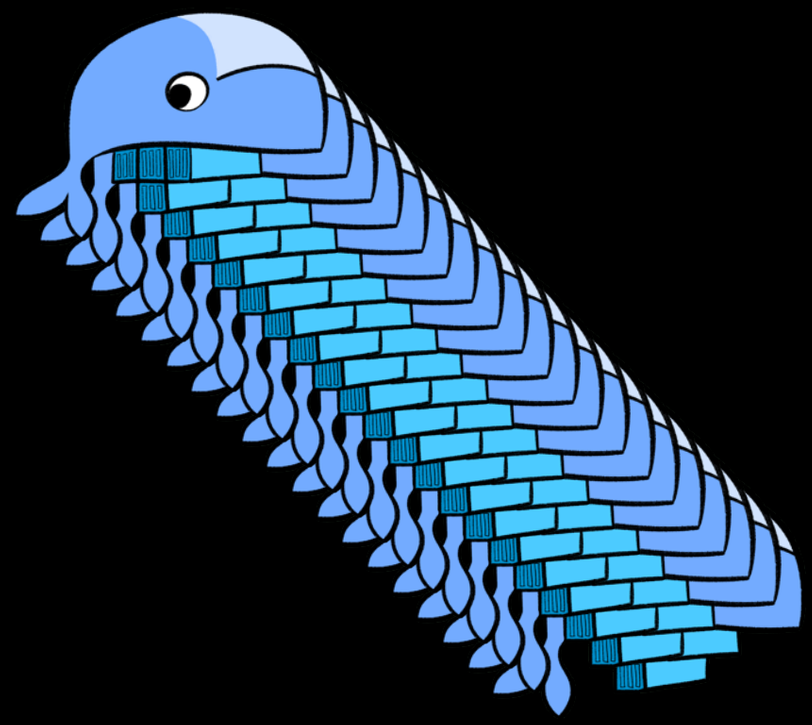
popular work loads

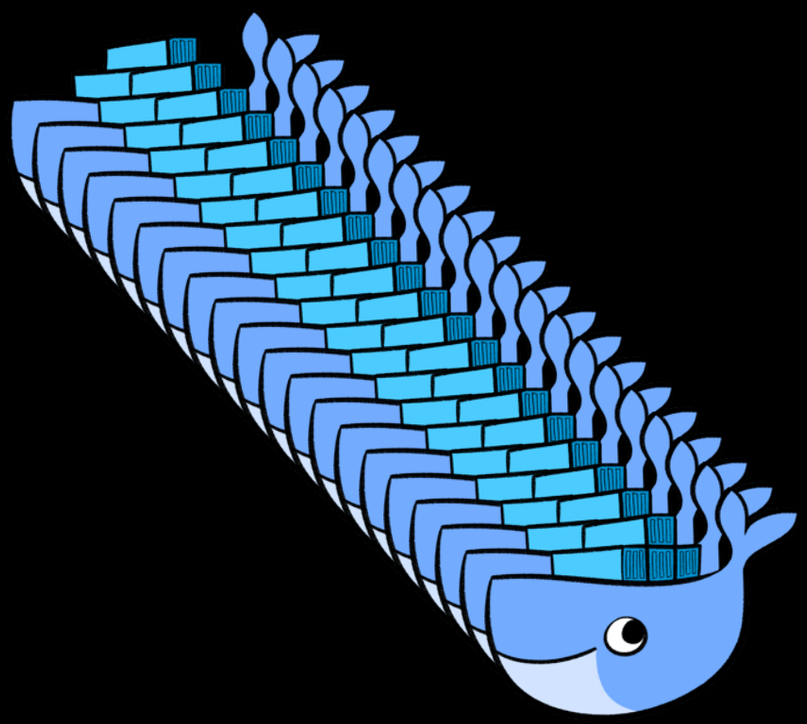
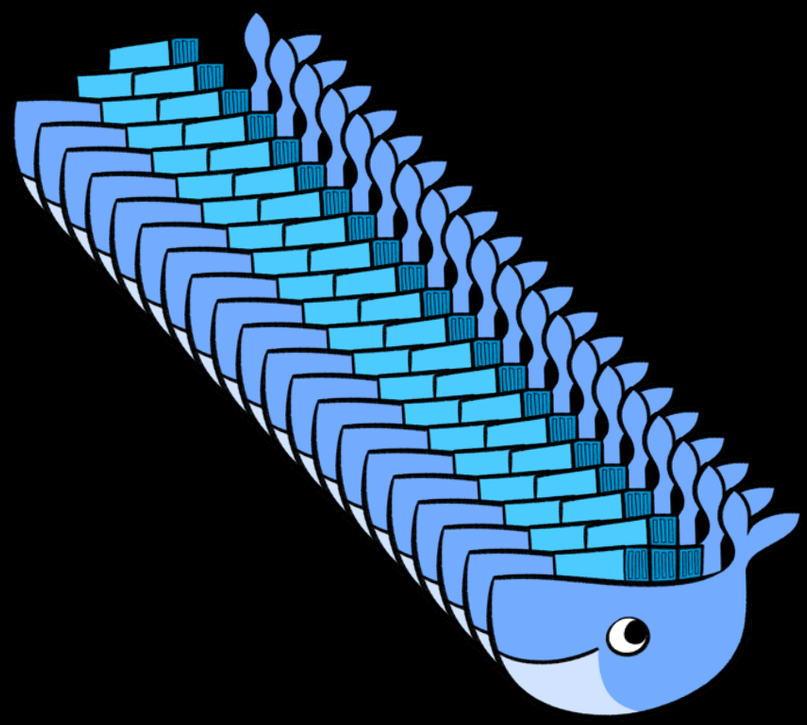
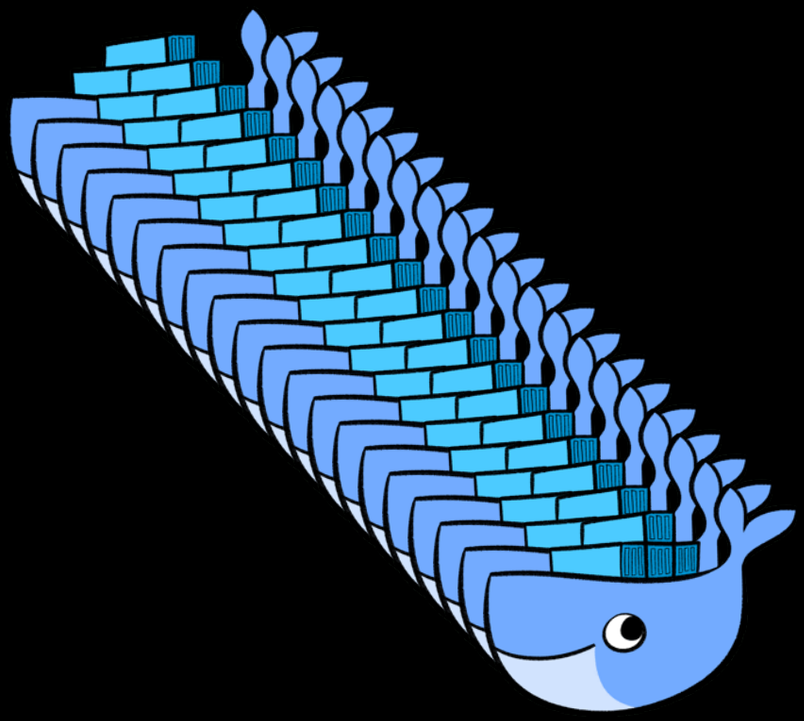
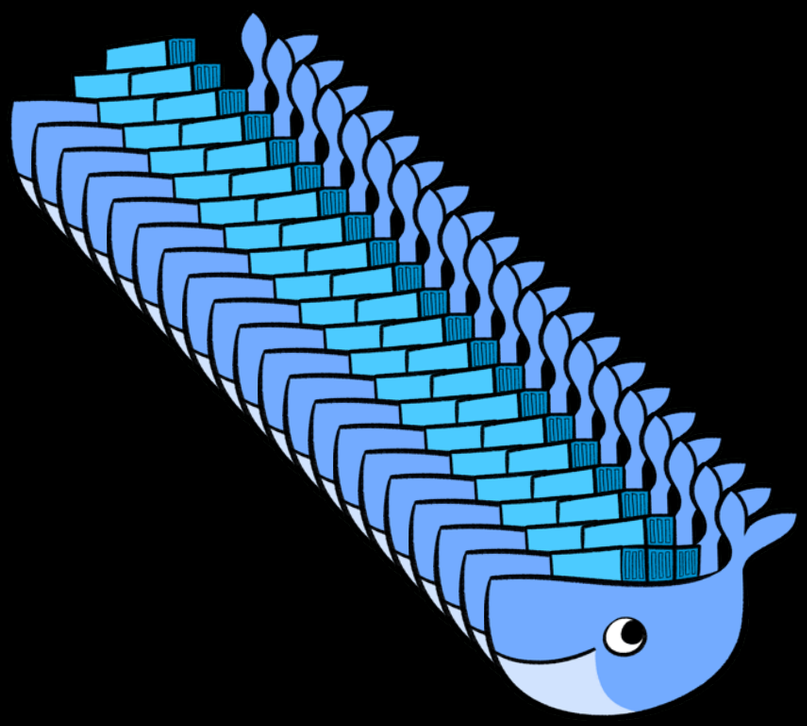
web applications
mobile backends
data stream processing
chat bots
IoT
alexa skills
OPS automation
security auditing
video transcoding
microservices

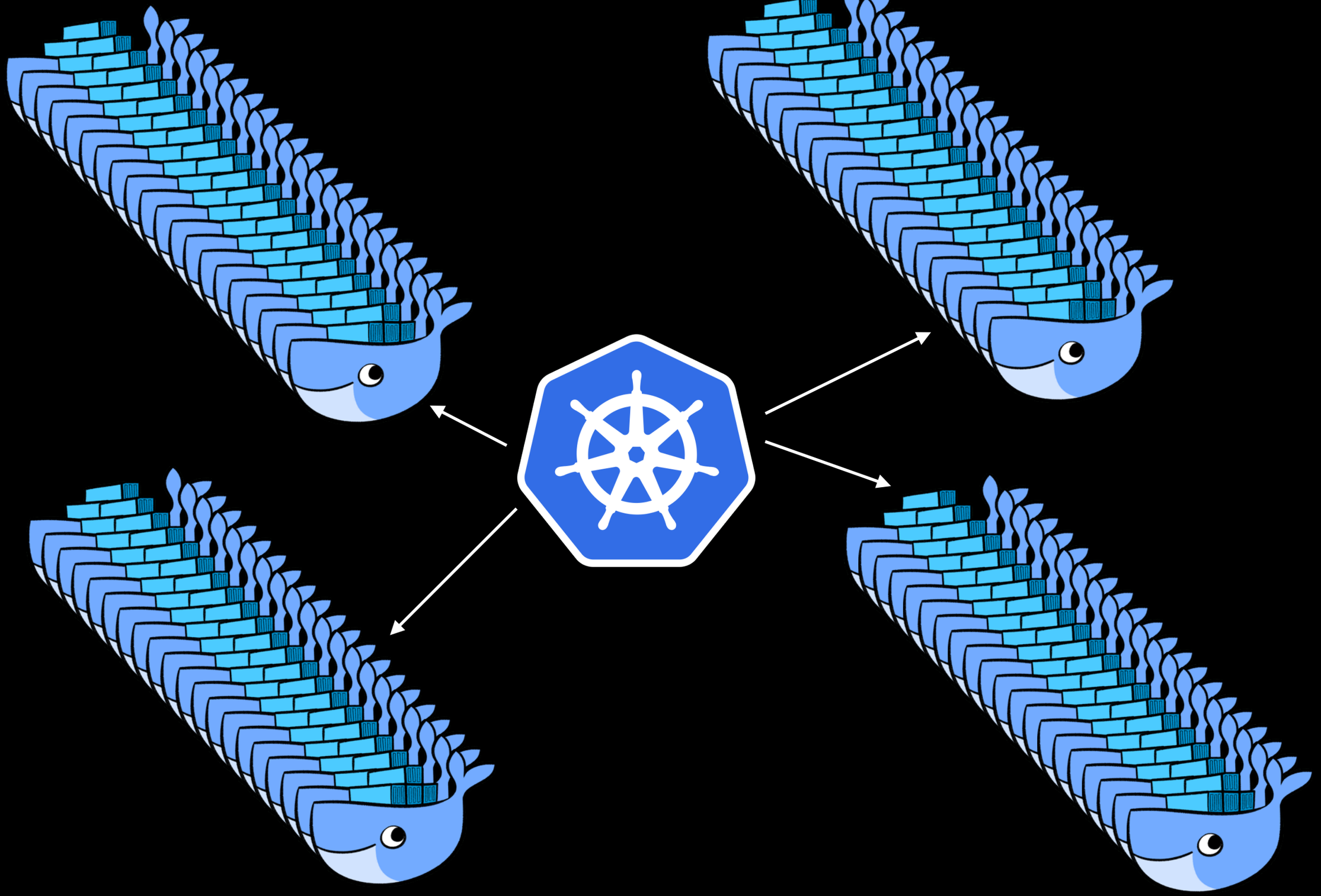
microservices

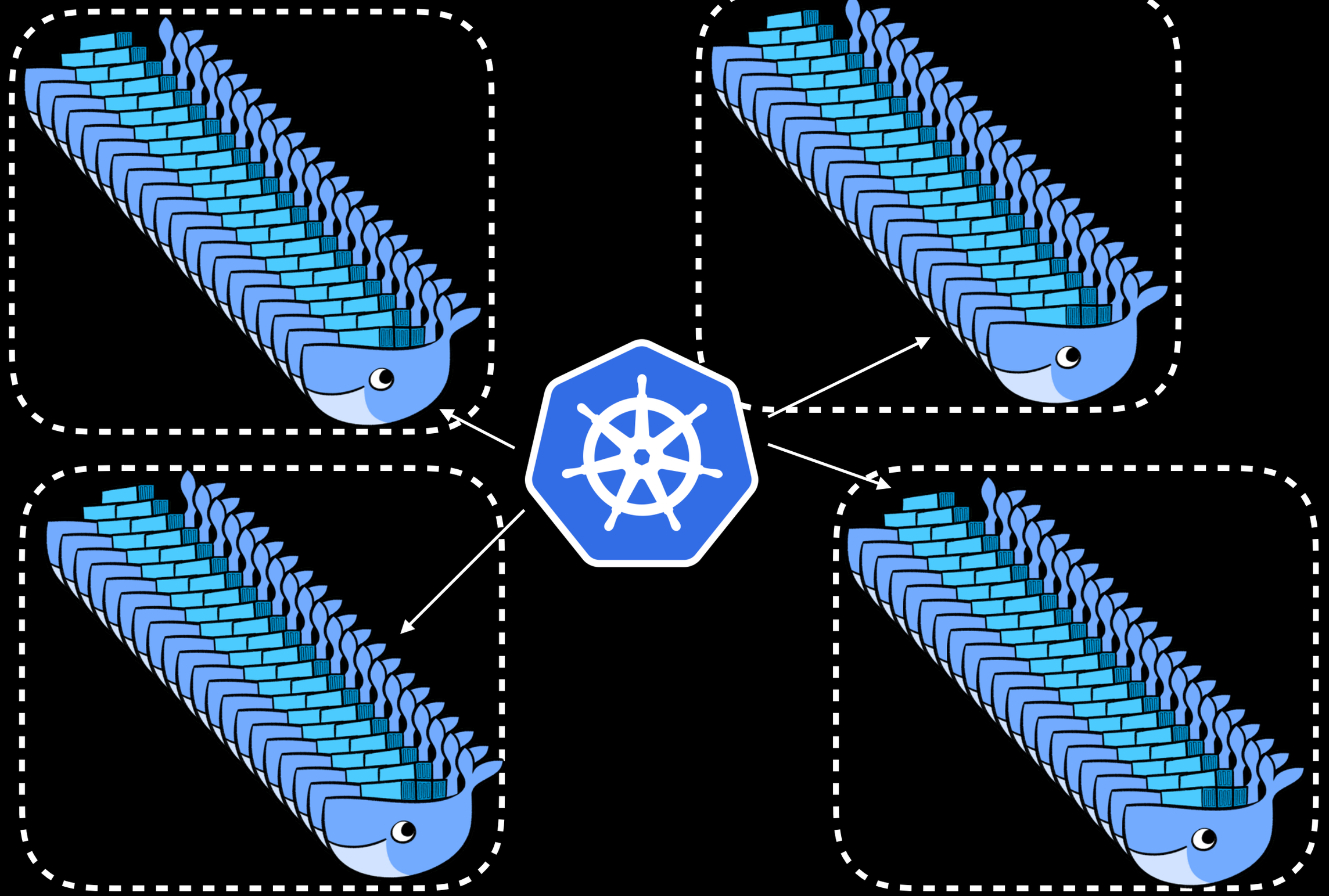


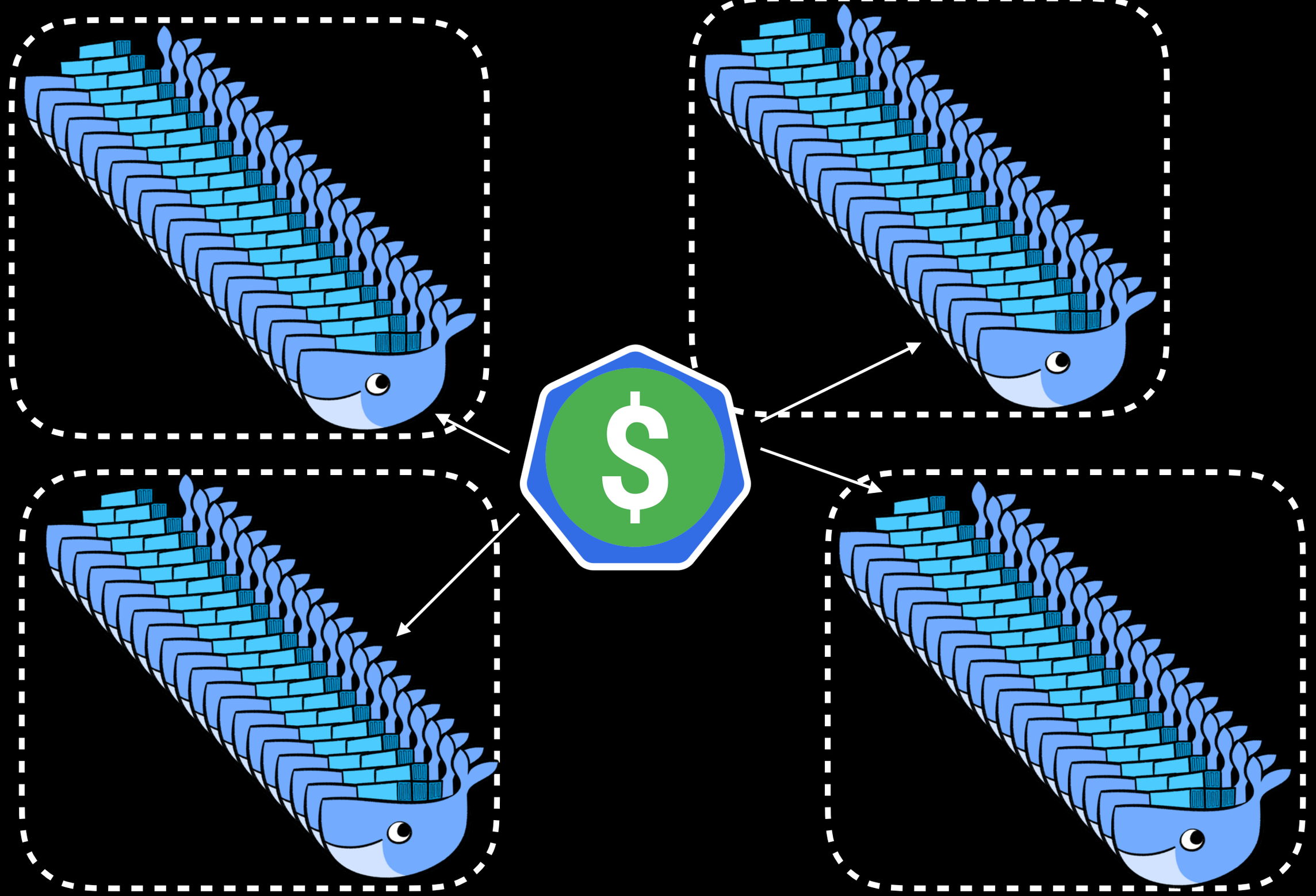


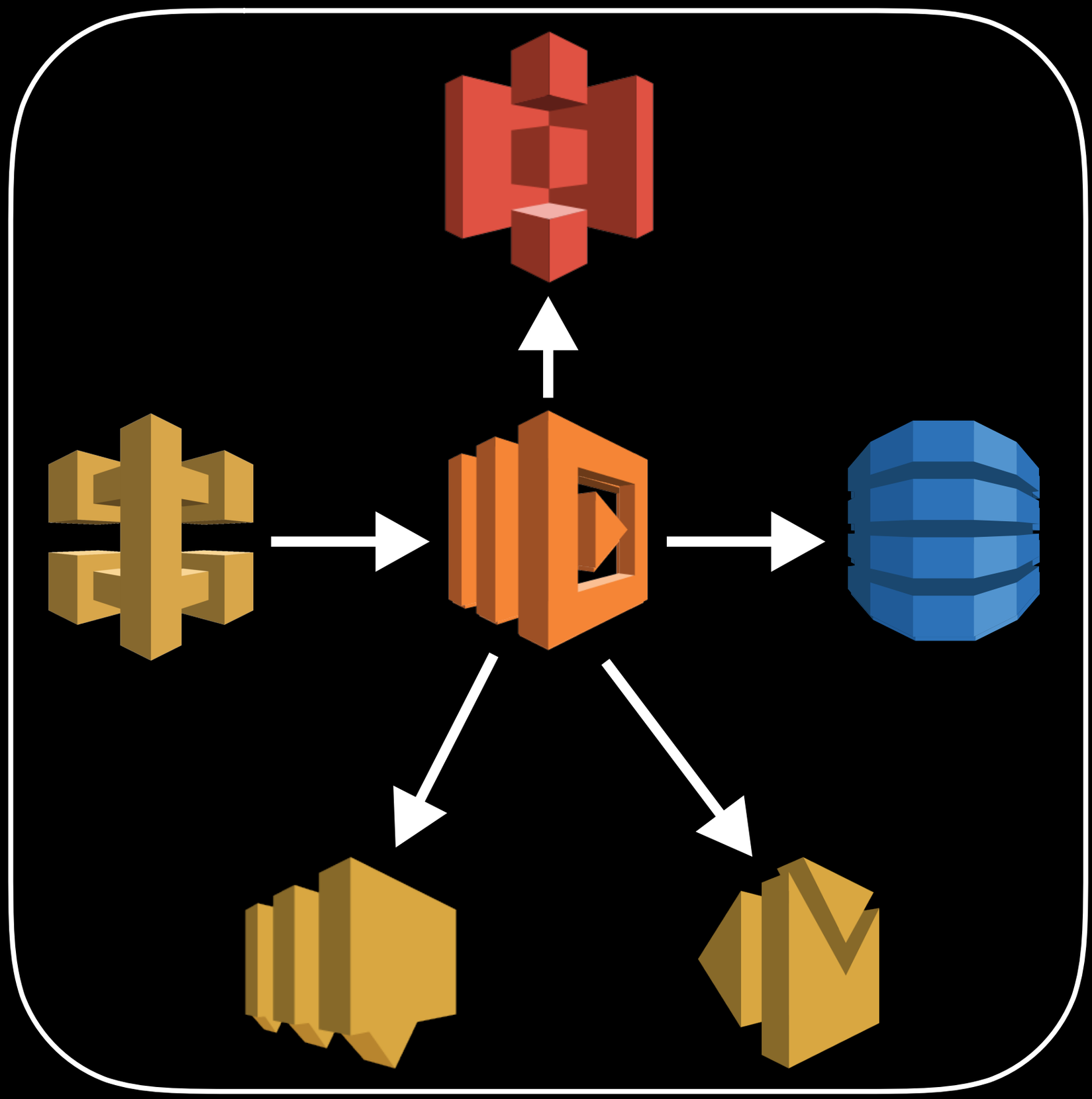


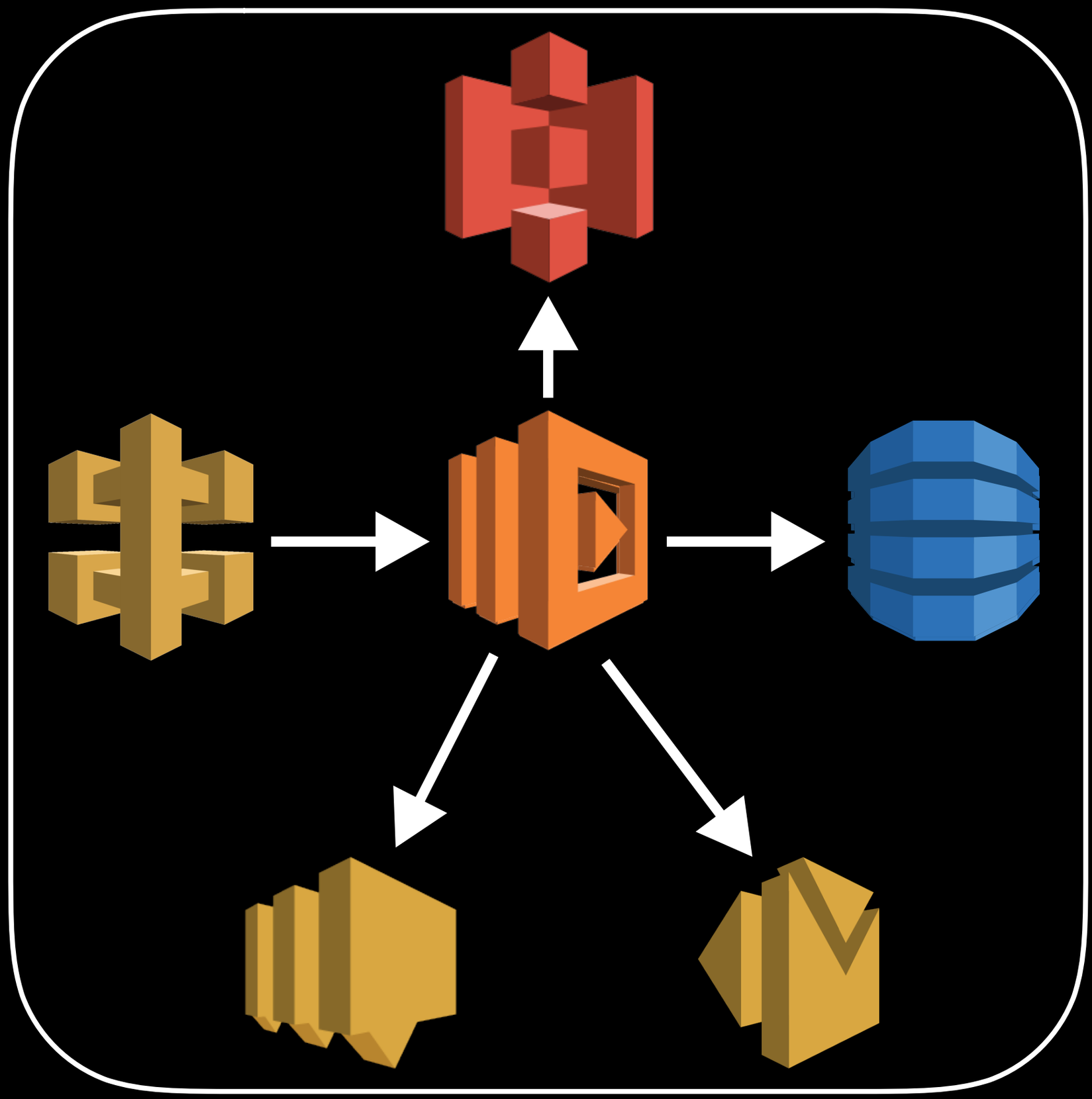















serverless delivers the original promise of the cloud

serverless delivers the original promise of the cloud

cloud native by default…

unless you really mess things up

serverless awareness

stateless (yes/no)

latency (cold starts)

local testing

debugging

tooling

db’s (scale)

security

security

granular IAM policies (key!)

no long standing servers

code dependencies

data

vendor lock-in

vendor lock-in

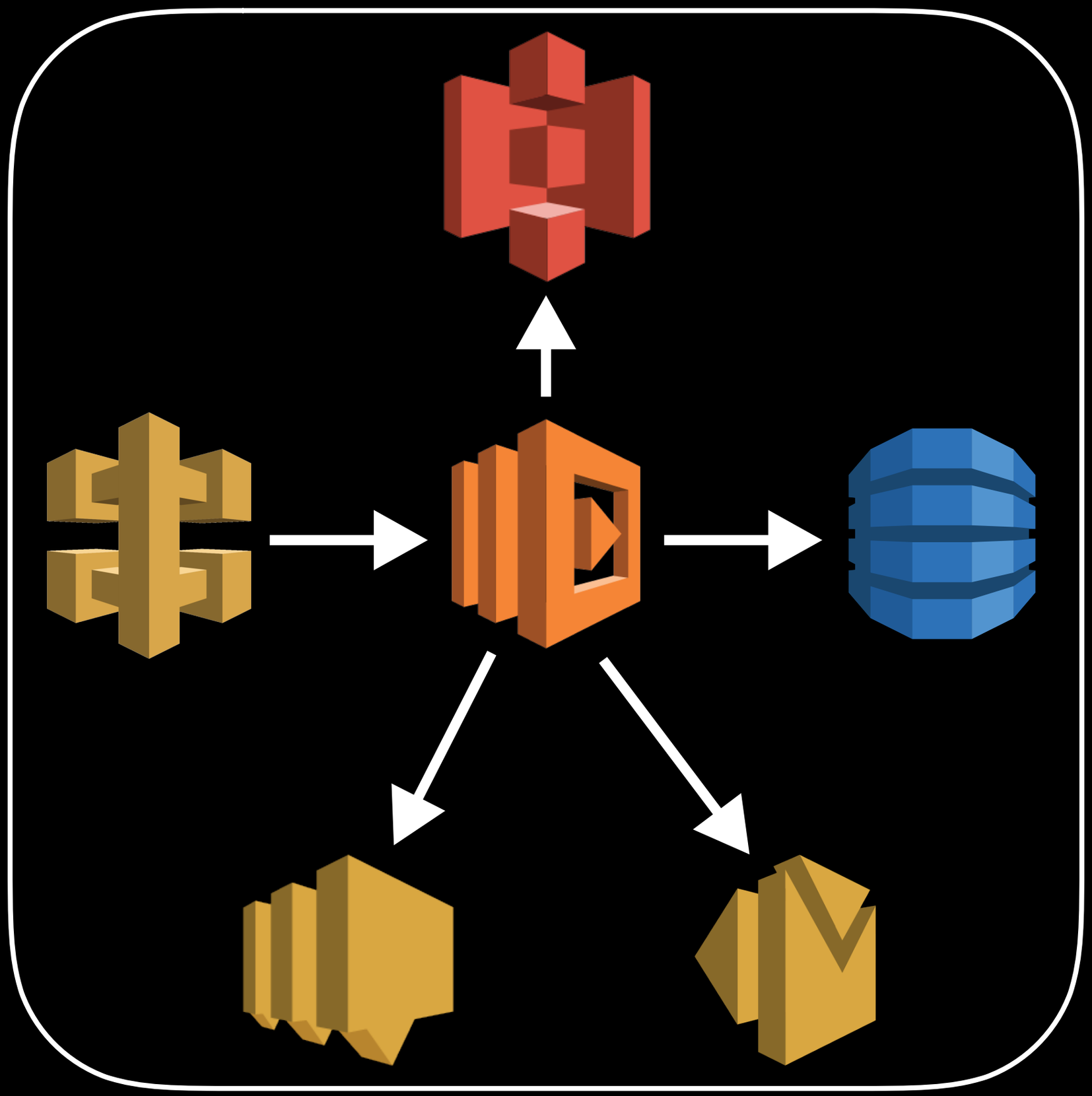
services and data (comes with territory)

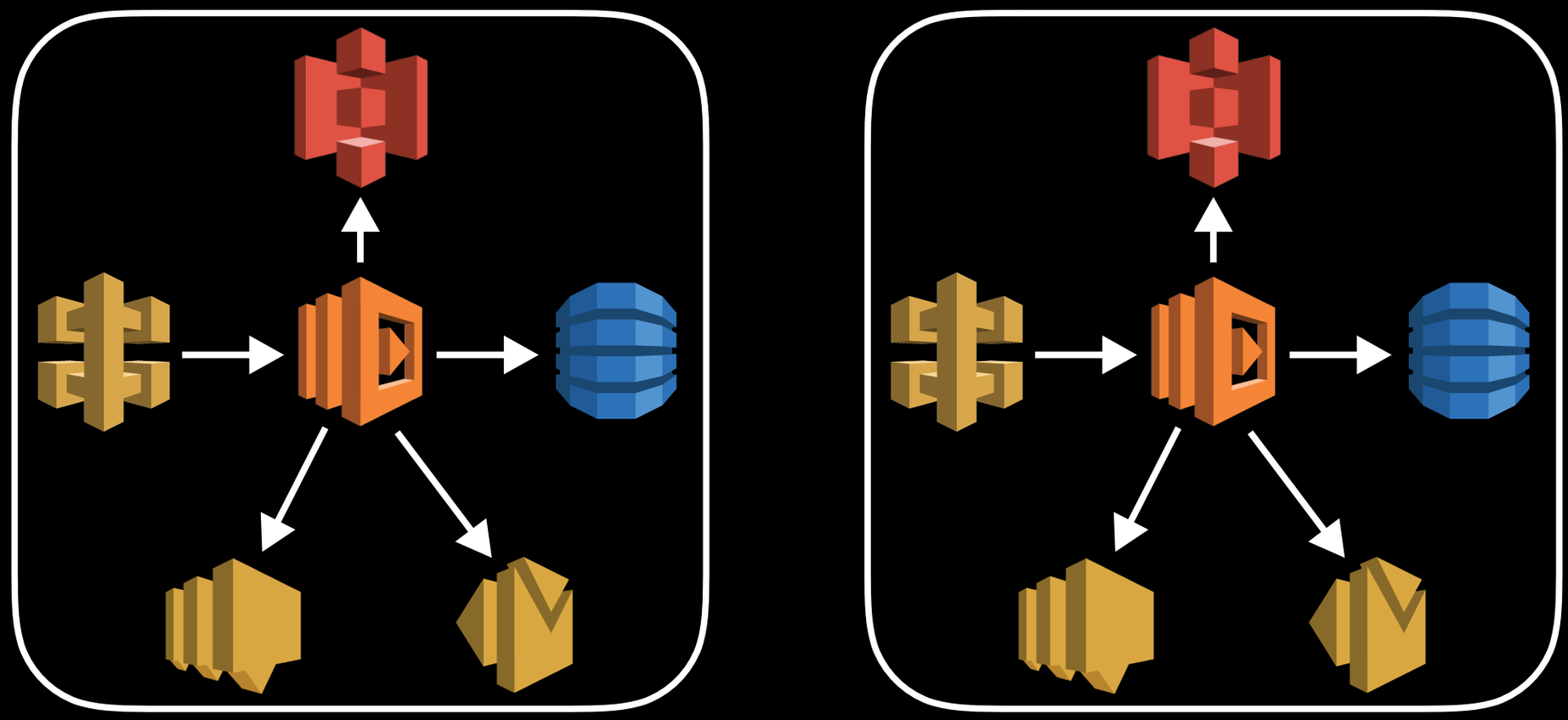
no lock-in is a myth

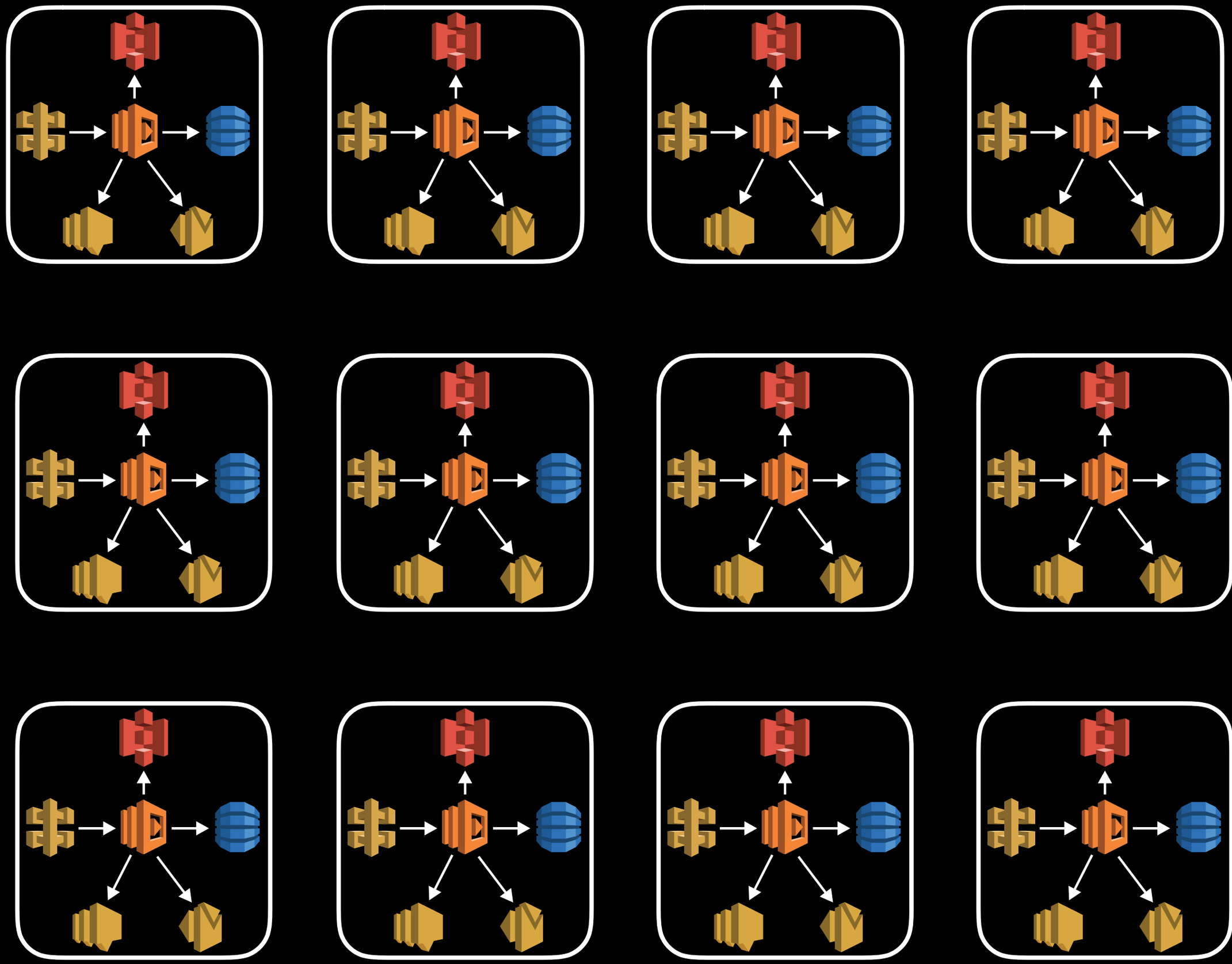
keep function logic in libs (lambda as wrappers)

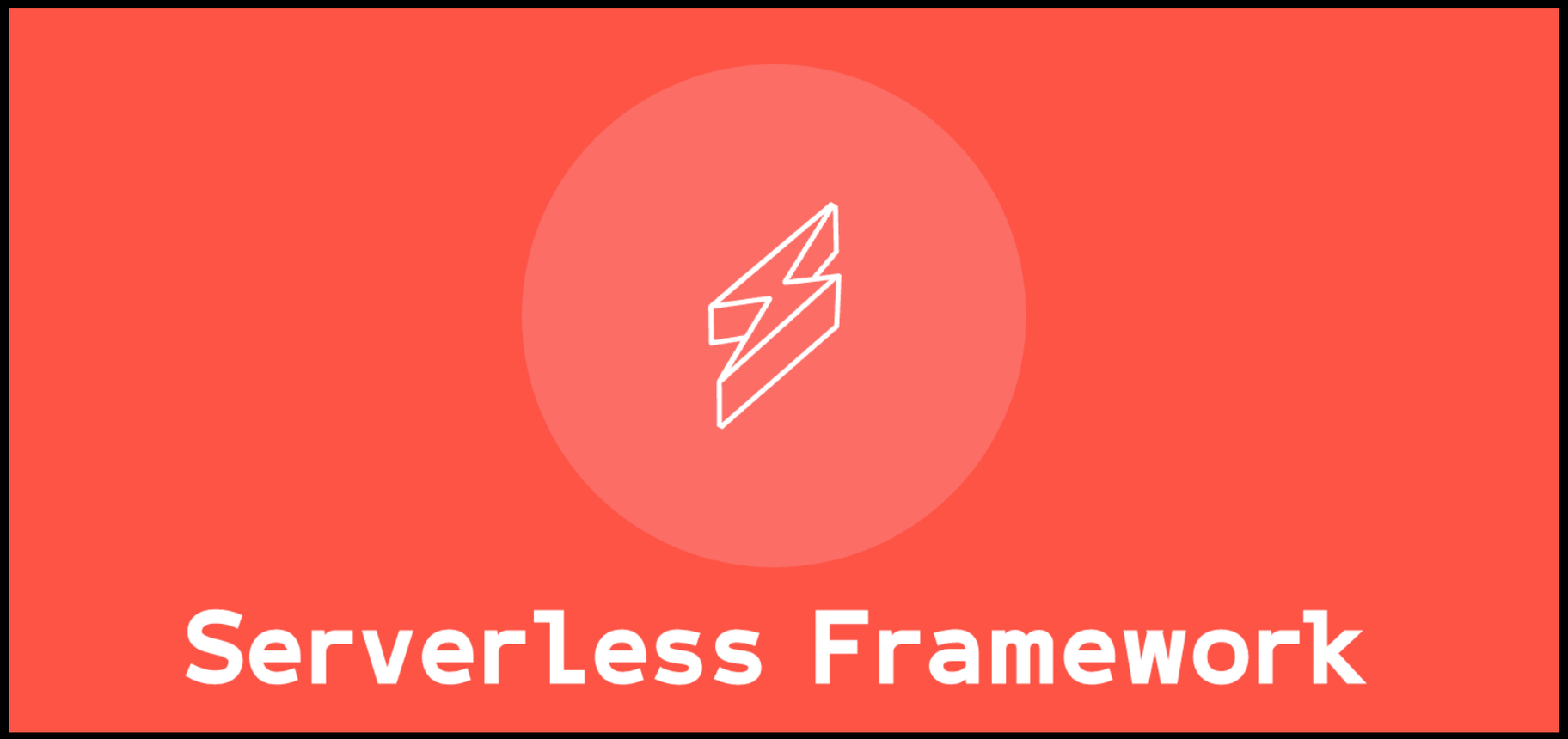
cloud agnostic not practical

pick a solid provider and exploit it!



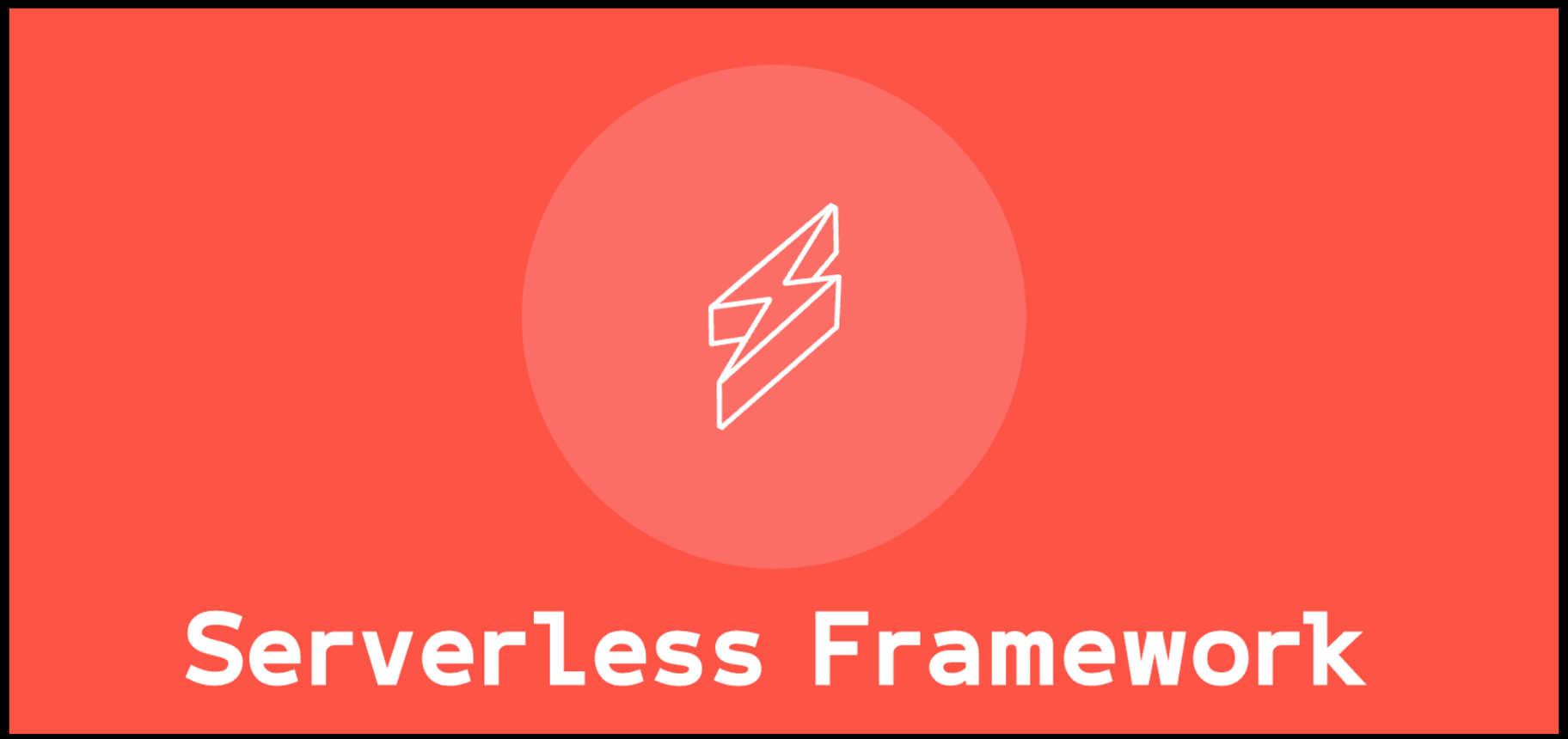





Serverless Framework

Serverless Framework

provisioning
orchestration
&
deployment



CloudFormation abstraction

+

aws cli

$ npm -i serverless -g

$ npm -i serverless -g

$ sls create --template aws-nodejs --path serverless-demo

$ npm -i serverless -g

$ sls create --template aws-nodejs --path serverless-demo

$ sls create --template aws-java-gradle --path serverless-demo

serverless.yml — TrackMyContainer-services

serverless.yml

```yaml
provider:
  name: aws
  region: us-west-2
  runtime: nodejs6.10

functions:
  trackShipmentByContainer:
    handler: containerTrack.trackShipmentByContainer
    memorySize: 256
    timeout: 10
    events:
      - http:
          path: /shipments/cn/{containerNumber}
          method: get
          cors: true

  forwardSNSToSlack:
    handler: slackForwarder.forwardSNSToSlack
    memorySize: 256
    timeout: 10
    events:
      - sns:
          topicName: track-my-container-alerts
          displayName: Track My Container Alerts

resources:
  Resources:
    DynamoDbTable:
      Type: AWS::DynamoDB::Table
      Properties:
        TableName: tmc-config
        AttributeDefinitions:
          - AttributeName: configKey
            AttributeType: S
        KeySchema:
          - AttributeName: configKey
            KeyType: HASH
        ProvisionedThroughput:
          ReadCapacityUnits: 2
          WriteCapacityUnits: 1
```

master 0 0 Ln 9, Col 28 Spaces: 2 UTF-8 LF YAML

```yaml
provider:
  name: aws
  region: us-west-2
  runtime: nodejs6.10

functions:
  trackShipmentByContainer:
    handler: containerTrack.trackShipmentByContainer
    memorySize: 256
    timeout: 10
    events:
      - http:
          path: /shipments/cn/{containerNumber}
          method: get
          cors: true

  forwardSNSToSlack:
    handler: slackForwarder.forwardSNSToSlack
    memorySize: 256
    timeout: 10
    events:
      - sns:
          topicName: track-my-container-alerts
          displayName: Track My Container Alerts

resources:
  Resources:
    DynamoDbTable:
      Type: AWS::DynamoDB::Table
      Properties:
        TableName: tmc-config
        AttributeDefinitions:
```

```yaml
provider:
  name: aws
  region: us-west-2
  runtime: nodejs6.10

functions:
  trackShipmentByContainer:
    handler: containerTrack.trackShipmentByContainer
    memorySize: 256
    timeout: 10
    events:
      - http:
          path: /shipments/cn/{containerNumber}
          method: get
          cors: true

  forwardSNSToSlack:
    handler: slackForwarder.forwardSNSToSlack
    memorySize: 256
    timeout: 10
    events:
      - sns:
          topicName: track-my-container-alerts
          displayName: Track My Container Alerts

resources:
  Resources:
    DynamoDbTable:
      Type: AWS::DynamoDB::Table
      Properties:
        TableName: tmc-config
        AttributeDefinitions:
```

```yaml
provider:
  name: aws
  region: us-west-2
  runtime: nodejs6.10

functions:
  trackShipmentByContainer:
    handler: containerTrack.trackShipmentByContainer
    memorySize: 256
    timeout: 10
    events:
      - http:
          path: /shipments/cn/{containerNumber}
          method: get
          cors: true

  forwardSNSToSlack:
    handler: slackForwarder.forwardSNSToSlack
    memorySize: 256
    timeout: 10
    events:
      - sns:
          topicName: track-my-container-alerts
          displayName: Track My Container Alerts

resources:
  Resources:
    DynamoDbTable:
      Type: AWS::DynamoDB::Table
      Properties:
        TableName: tmc-config
        AttributeDefinitions:
```

```yaml
provider:
  name: aws
  region: us-west-2
  runtime: nodejs6.10

functions:
  trackShipmentByContainer:
    handler: containerTrack.trackShipmentByContainer
    memorySize: 256
    timeout: 10
    events:
      - http:
          path: /shipments/cn/{containerNumber}
          method: get
          cors: true

  forwardSNSToSlack:
    handler: slackForwarder.forwardSNSToSlack
    memorySize: 256
    timeout: 10
    events:
      - sns:
          topicName: track-my-container-alerts
          displayName: Track My Container Alerts

resources:
  Resources:
    DynamoDbTable:
      Type: AWS::DynamoDB::Table
      Properties:
        TableName: tmc-config
        AttributeDefinitions:
```

```
    timeout: 10
    events:
      - http:
          path: /shipments/cn/{containerNumber}
          method: get
          cors: true

  forwardSNSToSlack:
    handler: slackForwarder.forwardSNSToSlack
    memorySize: 256
    timeout: 10
    events:
      - sns:
          topicName: track-my-container-alerts
          displayName: Track My Container Alerts

resources:
  Resources:
    DynamoDbTable:
      Type: AWS::DynamoDB::Table
      Properties:
        TableName: tmc-config
        AttributeDefinitions:
          - AttributeName: configKey
            AttributeType: S
        KeySchema:
          - AttributeName: configKey
            KeyType: HASH
        ProvisionedThroughput:
          ReadCapacityUnits: 2
          WriteCapacityUnits: 1
```

deployment

deployment

```
$ sls deploy
```

serverless deployments are
infrastructure *and* code

serverless deployments are infrastructure *and* code

infrastructure as code ❤️

areas to explore

observability

serverless ops

hybrid serverless applications

resources

Serverless Architectures

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

The Future of Serverless Compute

https://www.infoq.com/articles/future-serverless

Why the Fuss About Serverless

https://hackernoon.com/why-the-fuss-about-serverless-4370b1596da0

Serverless Security: What's Left To Protect

https://youtu.be/CiyUD_rI8D8

Serverless Computing and Applications

https://aws.amazon.com/serverless

Serverless guide

https://serverless.github.io/guide

Serverless Framework

https://serverless.com/framework