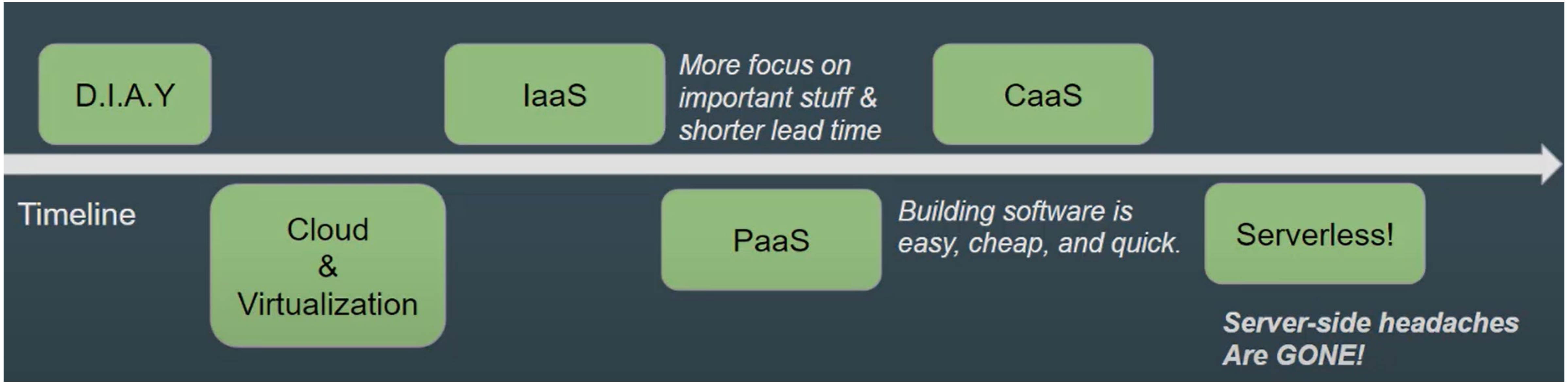
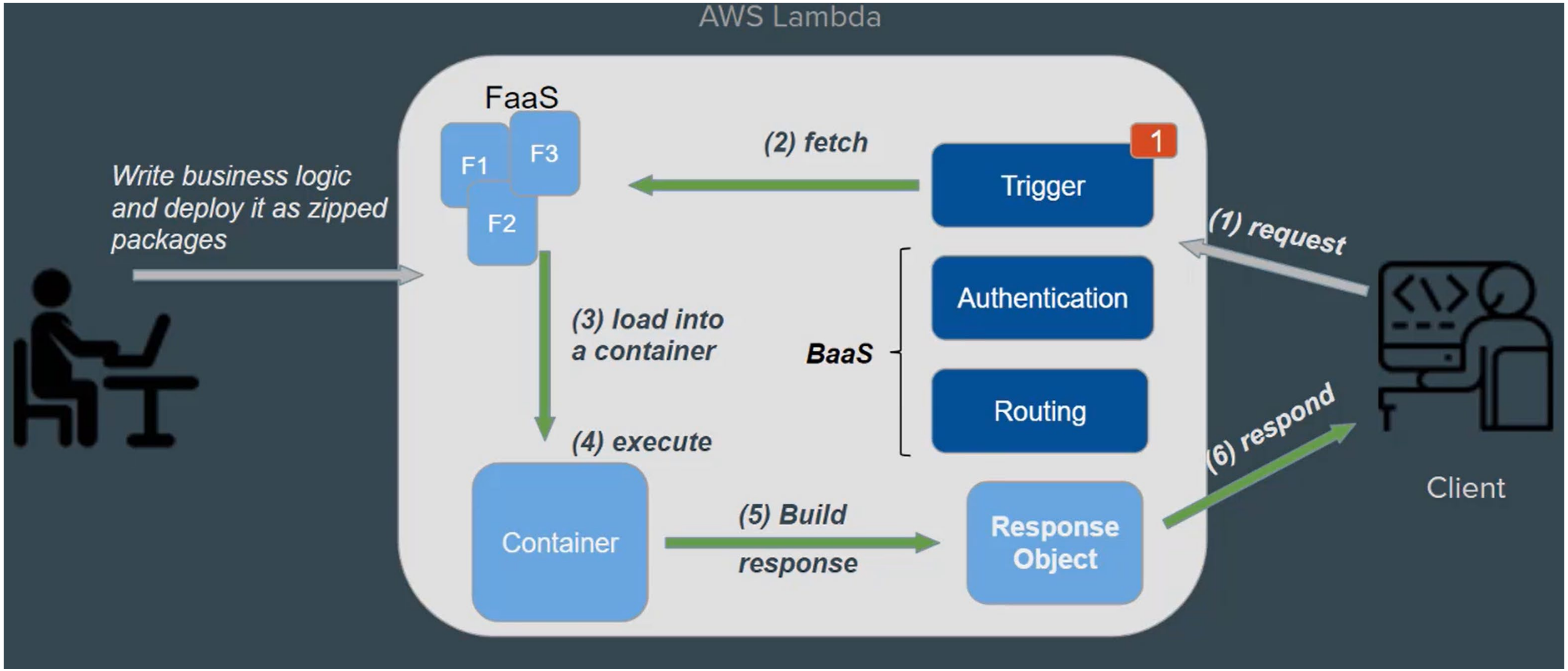
How we reached here?


D.I.A.Y
IaaS
More focus on important stuff & shorter lead time
CaaS
Timeline
Cloud & Virtualization
PaaS
Building software is easy, cheap, and quick.
Serverless!
Server-side headaches Are GONE!

How does it work?



What is Serverless

- No servers to provision or manage
- Automatically scales with usage
- Never pay for idle
- Highly available

Serverless Services

Amazon DynamoDB

Amazon API Gateway

AWS Step Functions

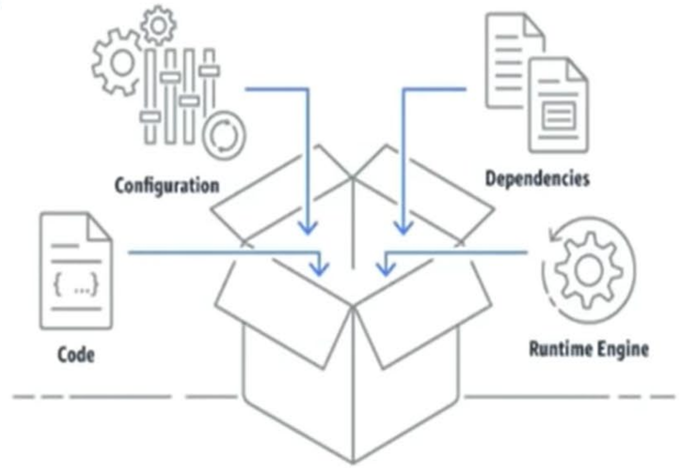
Amazon Simple Queue Service

AWS Lambda

A compute service that lets you run code without provisioning or managing servers

What is Container

Standard unit of software that packages up code and all its dependencies



The application runs quickly and reliably from one computing environment to another

Container Orchestrators

Kubernetes (K8)

Amazon EKS

Amazon ECS

Docker Swarm

Serverless

Underlying infrastructure managed by Cloud Provider

- Scales automatically
- No Patching headache

Can't install software (e.g. Webserver, Appserver) in underlying environment

- Code libraries can be installed

Easy selection of compute power

- 128 MB to 3 GB memory
- 1 sec to 15 Minutes time limit

No attached hard disk, deployment package size limited

Superpower - Easier to onboard, focus on solving business problem from get go

Container

Users control underlying infrastructure - VM Size, OS, AMI etc.

- Requires management and orchestration
- Need to make master node HA, handle VM failover, AMI rehydration etc.

Install almost any software

- Prepackaged images with different softwares available

Adjustment of VM parameters requires some work

- Think of it as changing EC2 instance type on a running instance

Hard Disks attached to nodes

Superpower - Complete control of environment, rich ecosystem

Serverless

Shines at event driven architectures

- Native integration with other services
- Example - Triggered by S3, Kinesis

Suited when traffic is unpredictable

- Autoscaling
- Pay as you go

Microservices

- API Gateway integration
- Code is modular without software dependencies, e.g - python APIs
- Easier to migrate Cloud native, green fields apps
- Consider VPC Cold Start Latency

Kryptonite - For brown field monoliths to Lambda, major refactoring needed

Container

Faster migration to cloud with other softwares

- Webserver, Appserver
- App requires third party software

Suited when traffic is predictable

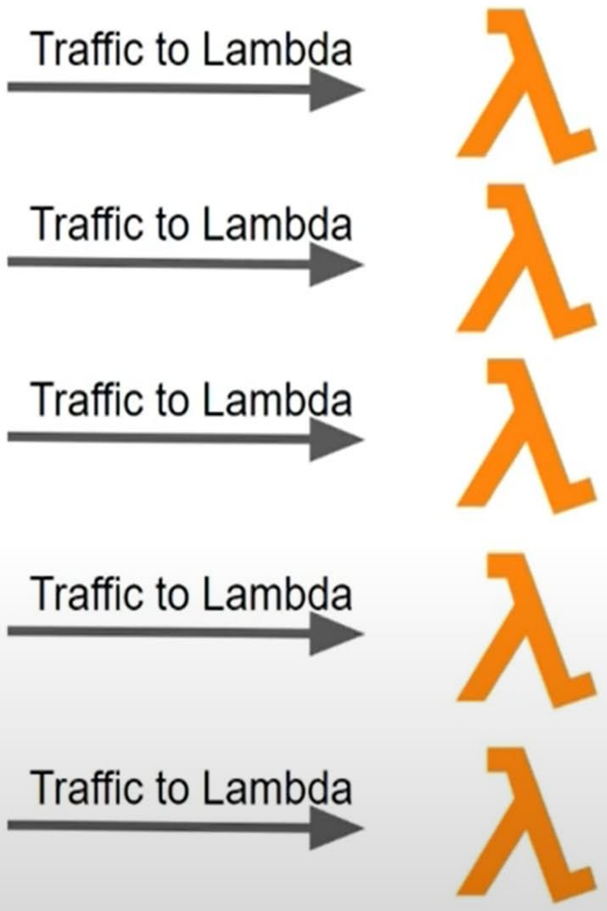
- You pay for the underlying VM regardless
- Scales an entire VM

Microservices!

- Easy to move API with dependencies, e.g. Spring Boot with Discovery layer
- Consider cost and complexity for green field

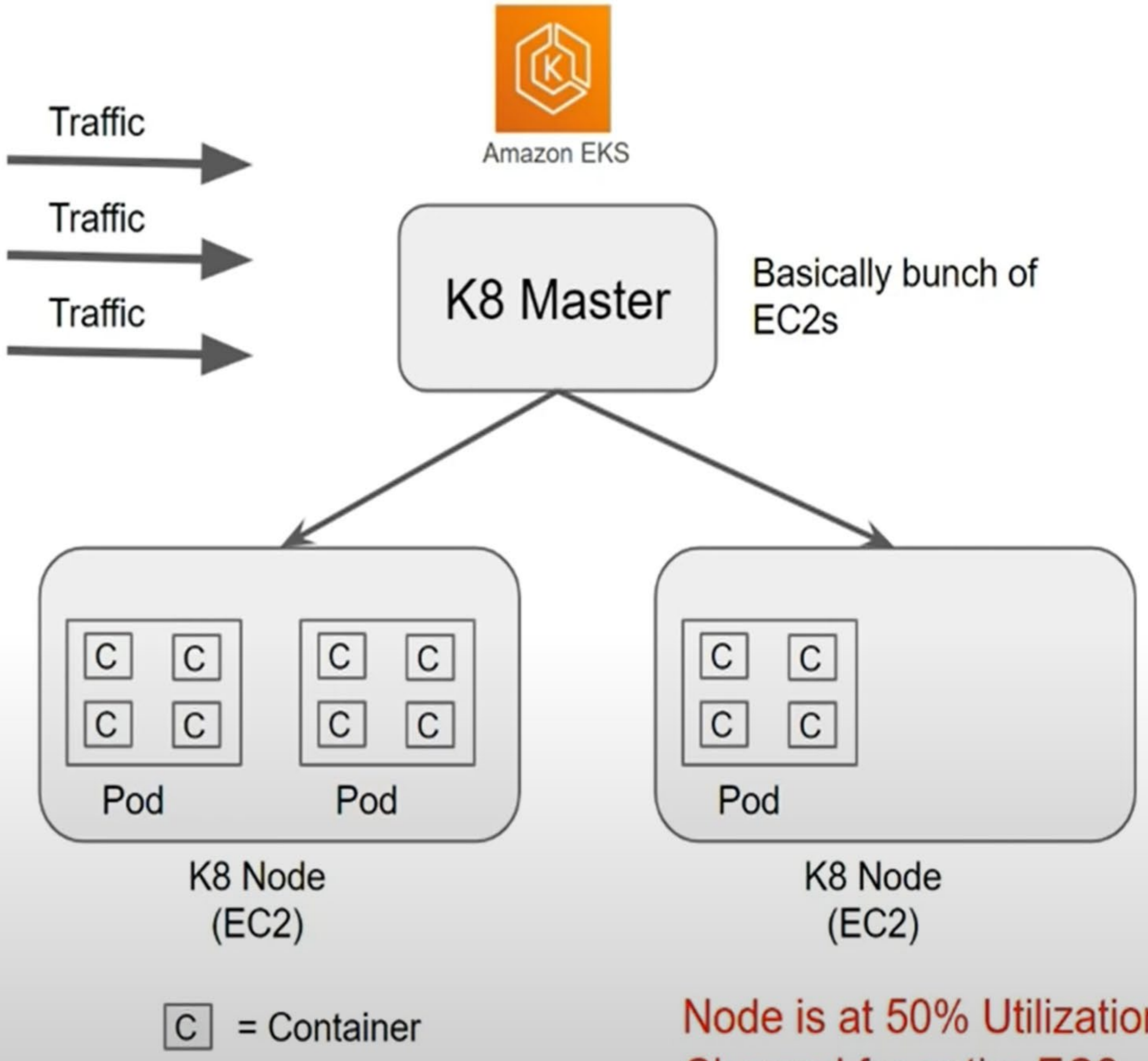
Kryptonite - Steep learning curve with multitudes of choices along with significant Day 2 operational overhead

Serverless



Pay for what you use

Container



Node is at 50% Utilization
Charged for entire EC2
Pay for idle resources