AWS re:Invent

CON326-R

Running Kubernetes applications on AWS Fargate

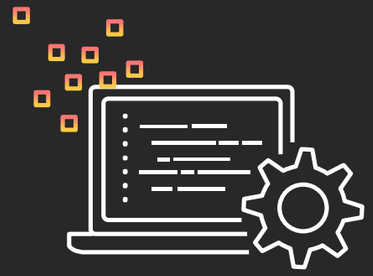
Nathan Taber
Senior Product Manager
Amazon Web Services

Massimo Re Ferrè
Principal Developer Advocate
Amazon Web Services

Make AWS the BEST PLACE to run KUBERNETES

Production Workloads

Native and upstream



OSS Contribution

Seamless integrations

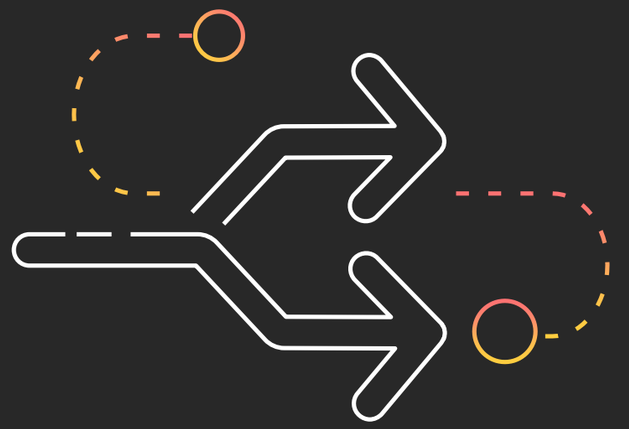
Production workloads

Single tenant

Multi-AZ and highly available architecture
by default

99.9% Service Level Agreement
for every cluster

Native and upstream



Upstream conformant

Integration testing
with Kubernetes tooling

APIs and existing tooling
just work

OSS contributions

AWS contributes
bug fixes, security patches, and tooling improvements

Open-source components
Contribute to or maintain over 30 OSS projects on GitHub for Kubernetes

Seamless integrations

Identity

Audits

Routing

Compliance

Monitoring

Logging

Ingress

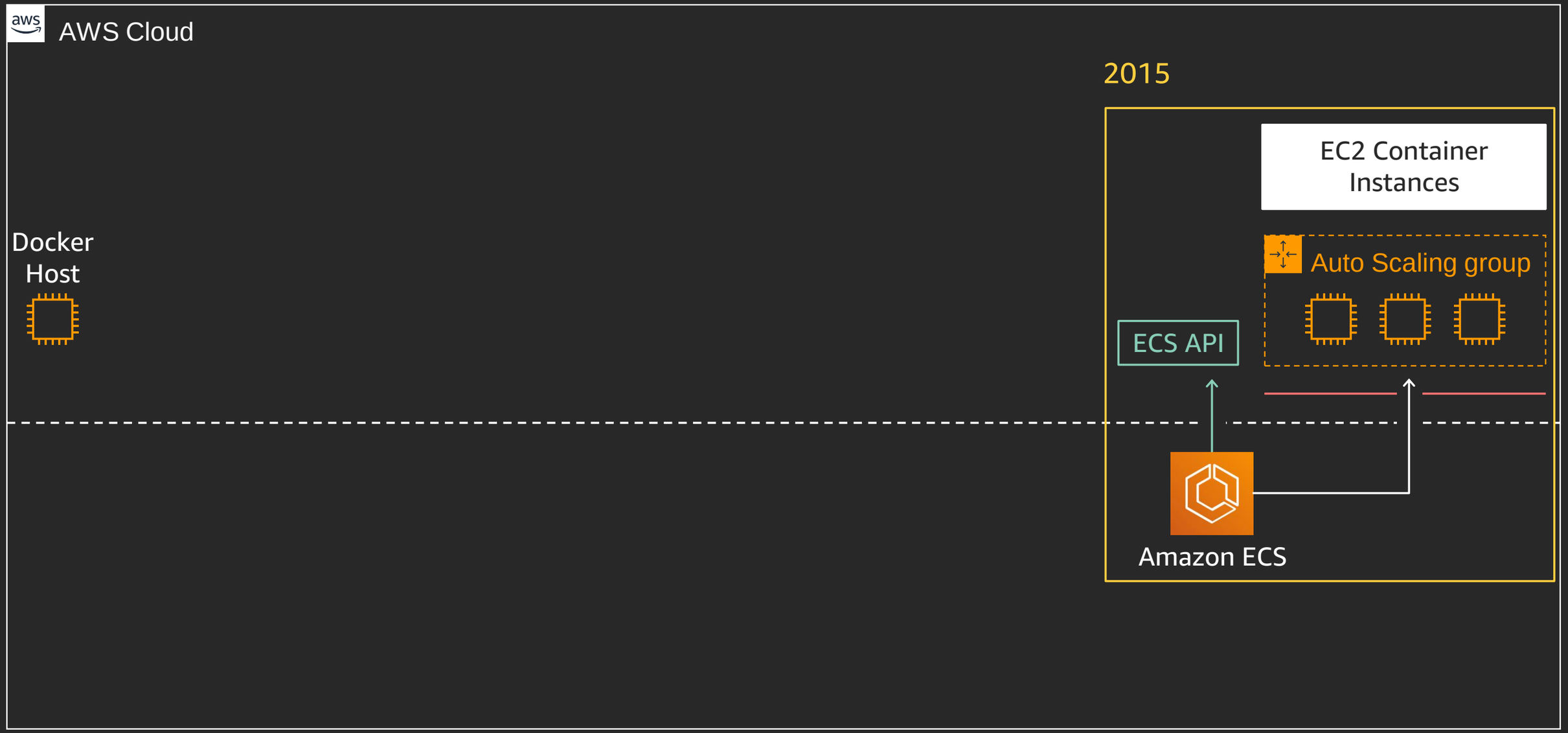
Security

Databases

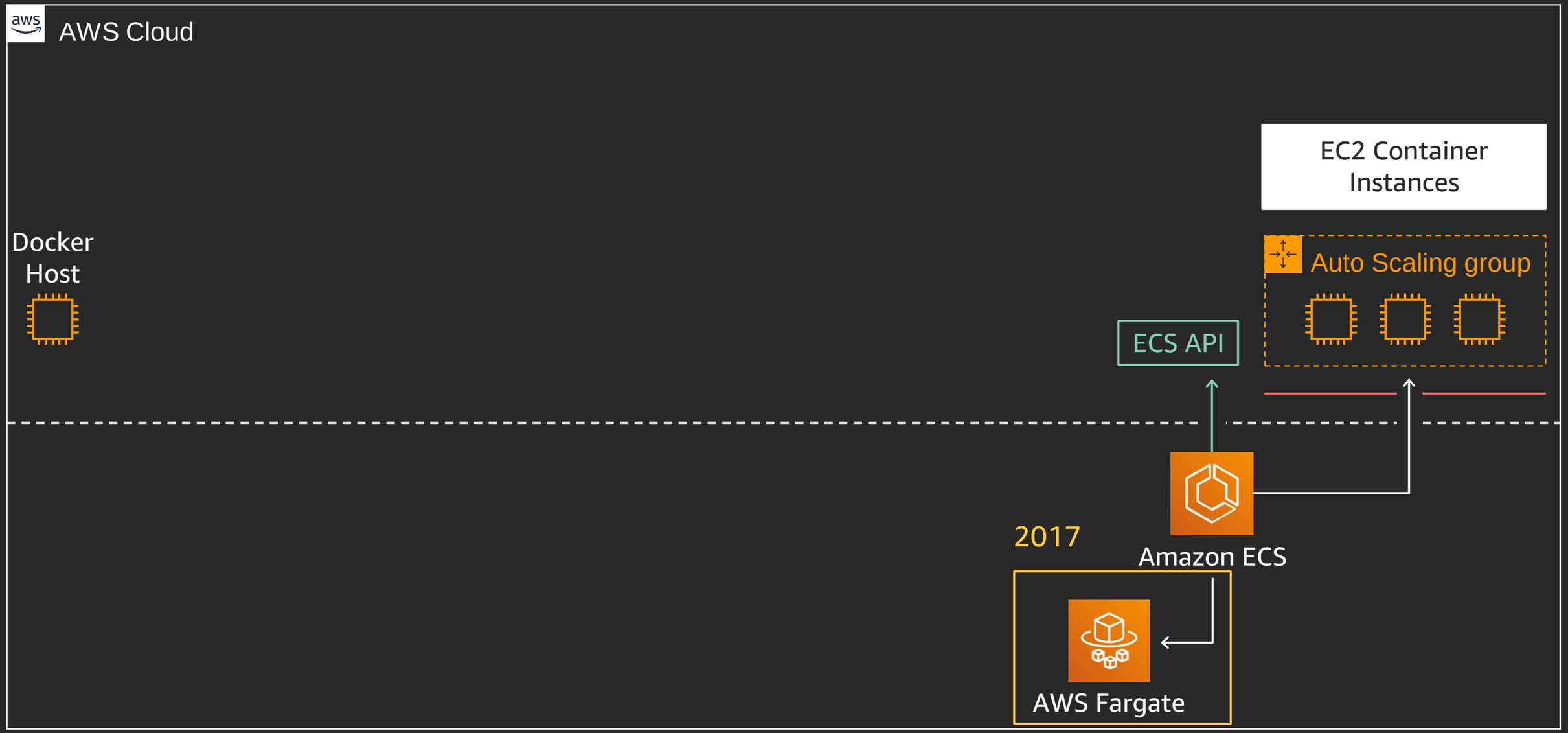

All the building blocks for Kubernetes
in one place

Containers options on AWS – over time

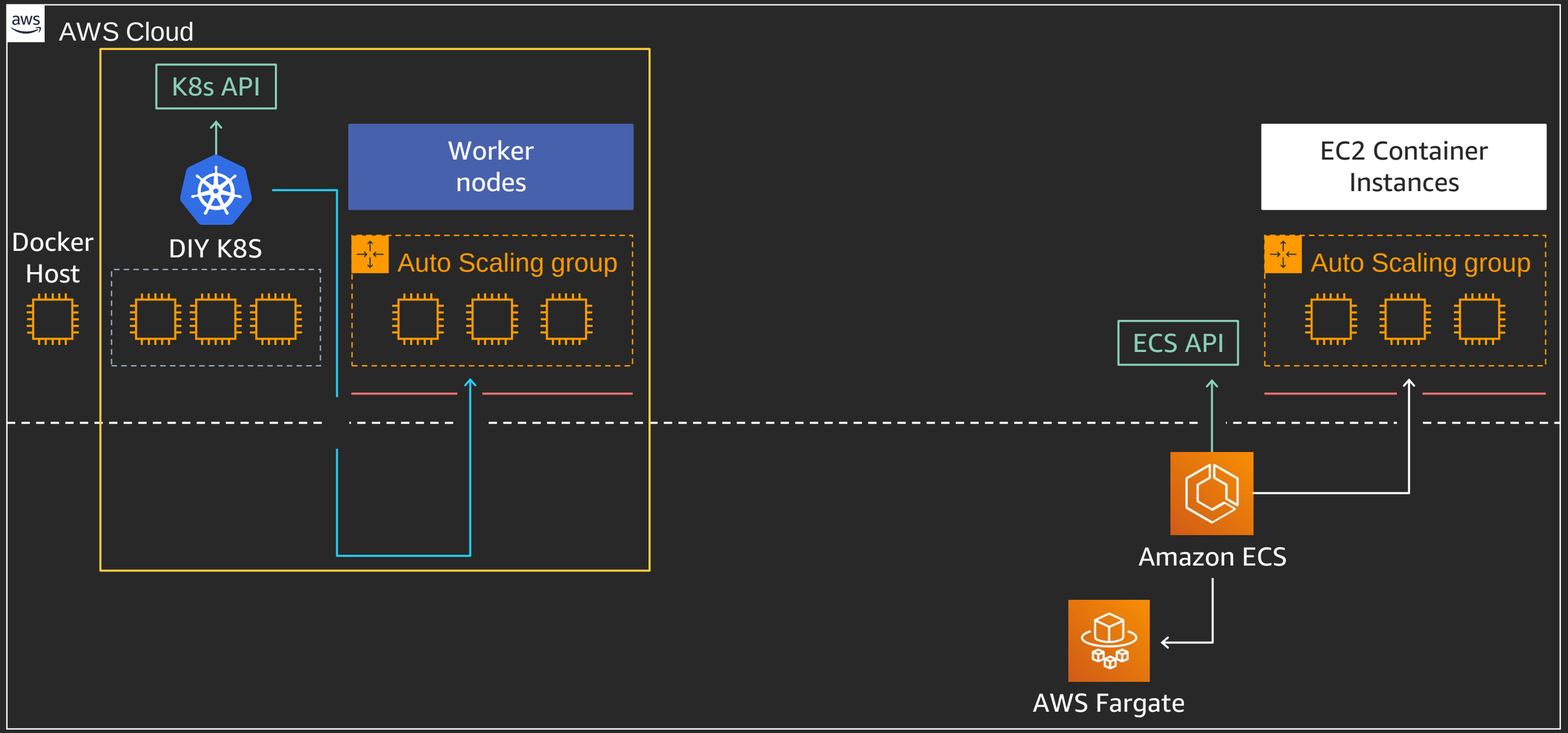
Containers options on AWS – over time



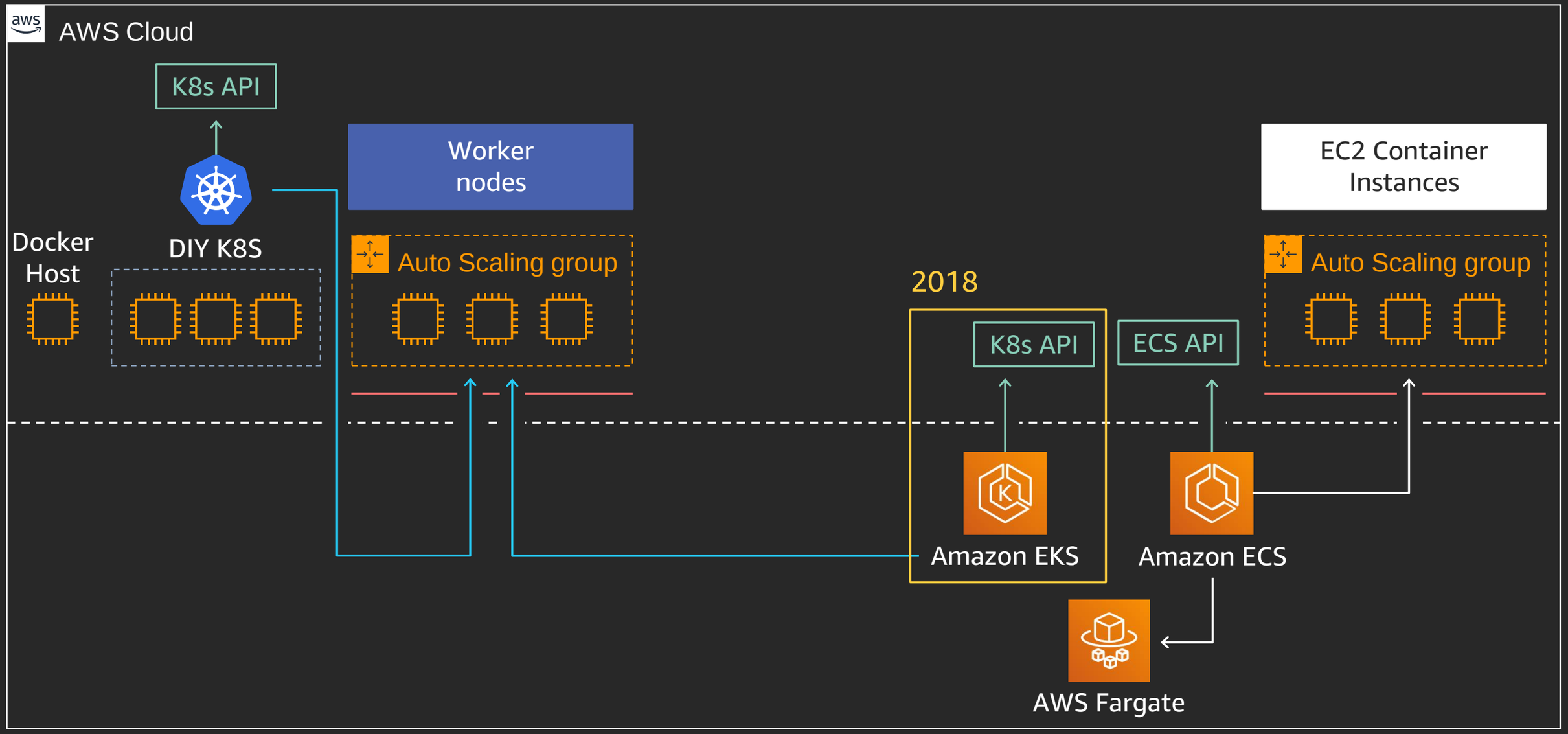
Containers options on AWS – over time



Containers options on AWS – over time

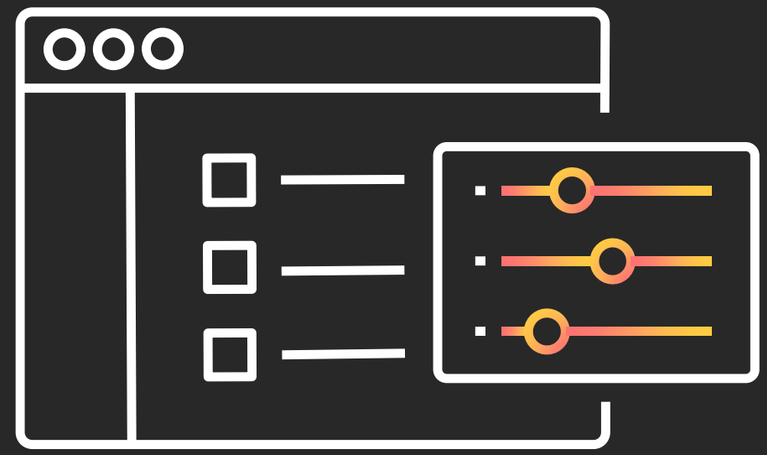


Containers options on AWS – over time



Phase 1

Management of the Kubernetes control plane

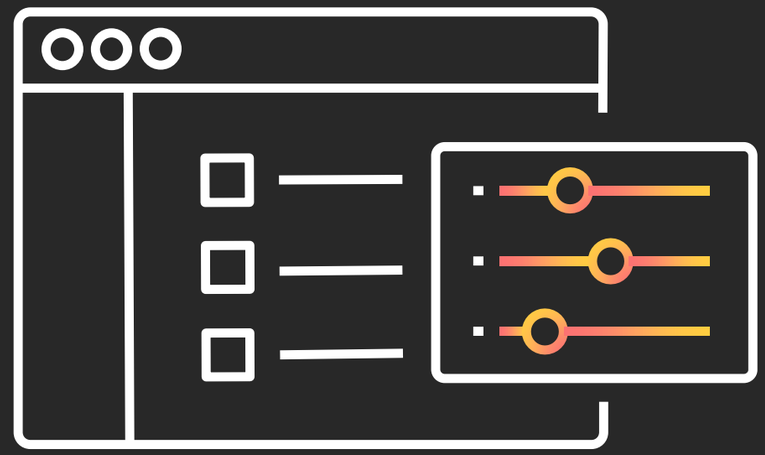


Phase 1

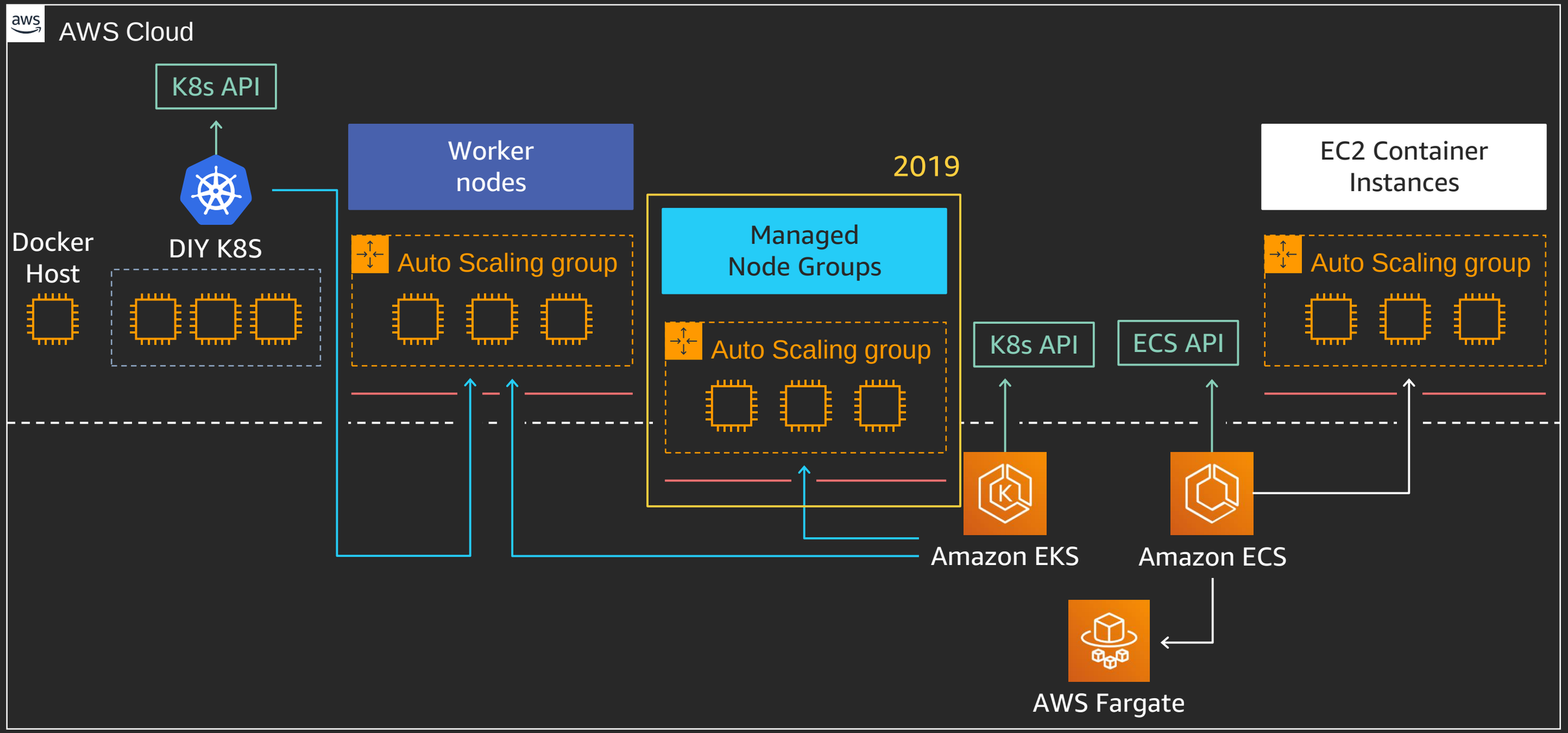
Management of the Kubernetes control plane

Phase 2

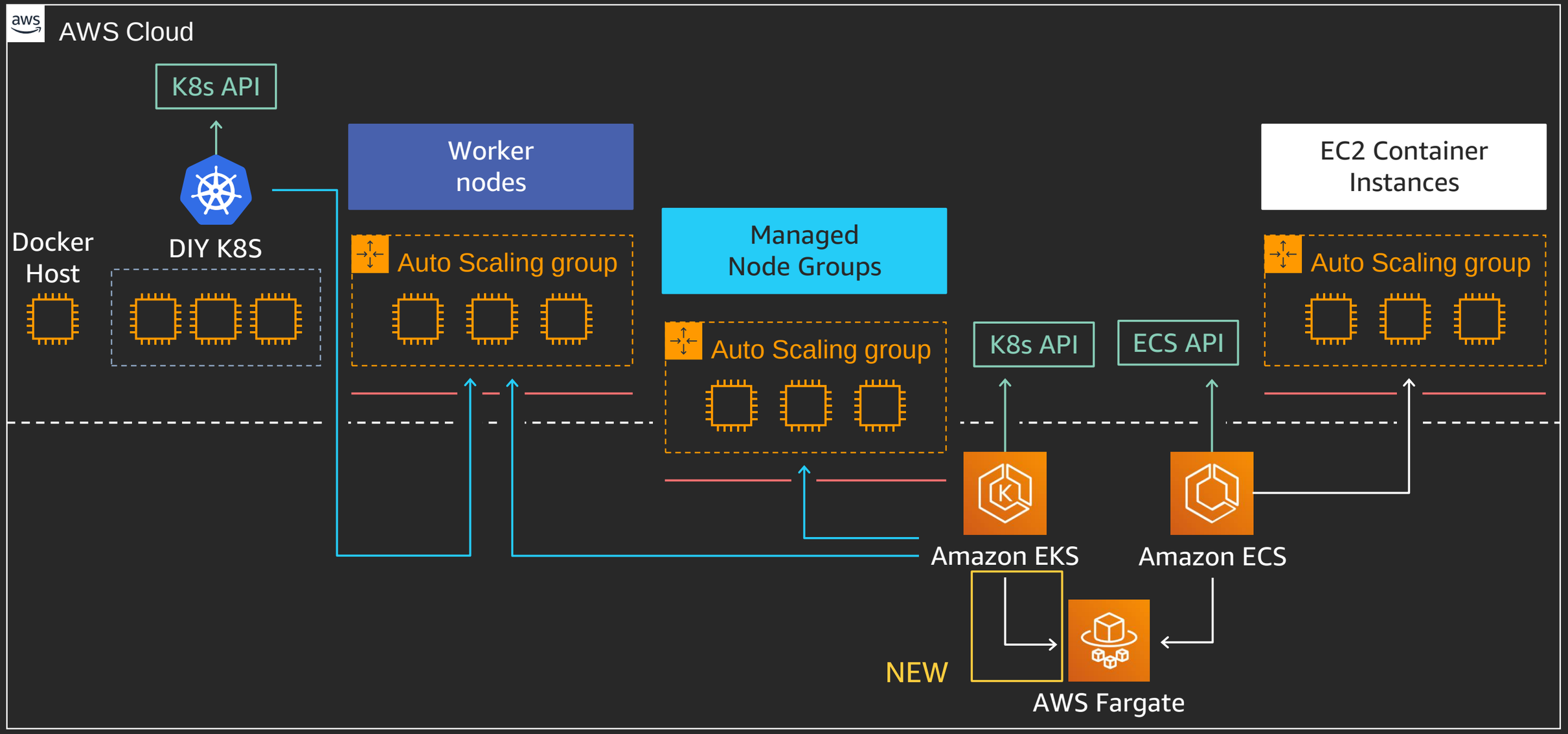
Management of the Kubernetes data plane



Containers options on AWS – over time



Containers options on AWS – over time



You should be able to write your code and have it run, without having to worry about configuring complex management tools.

This is the vision behind AWS Fargate.

Dr. Werner Vogels

CTO, Amazon.com

Amazon EKS on Fargate

Bring existing pods

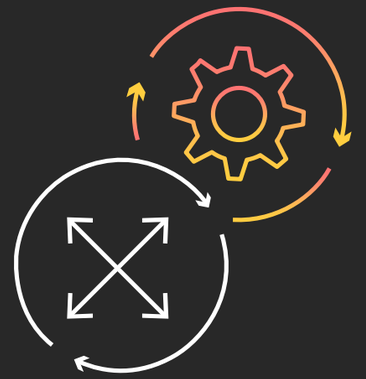
You don’t need to change your existing pods.

Fargate works with existing workflows and services that run on Kubernetes.

Production ready

Launch pods quickly. Easily run pods across multiple AZs for high availability.

Each pod runs in an isolated compute environment.



Rightsized and integrated

Only pay for the resources you need to run your pods.

Includes native AWS integrations for networking and security.

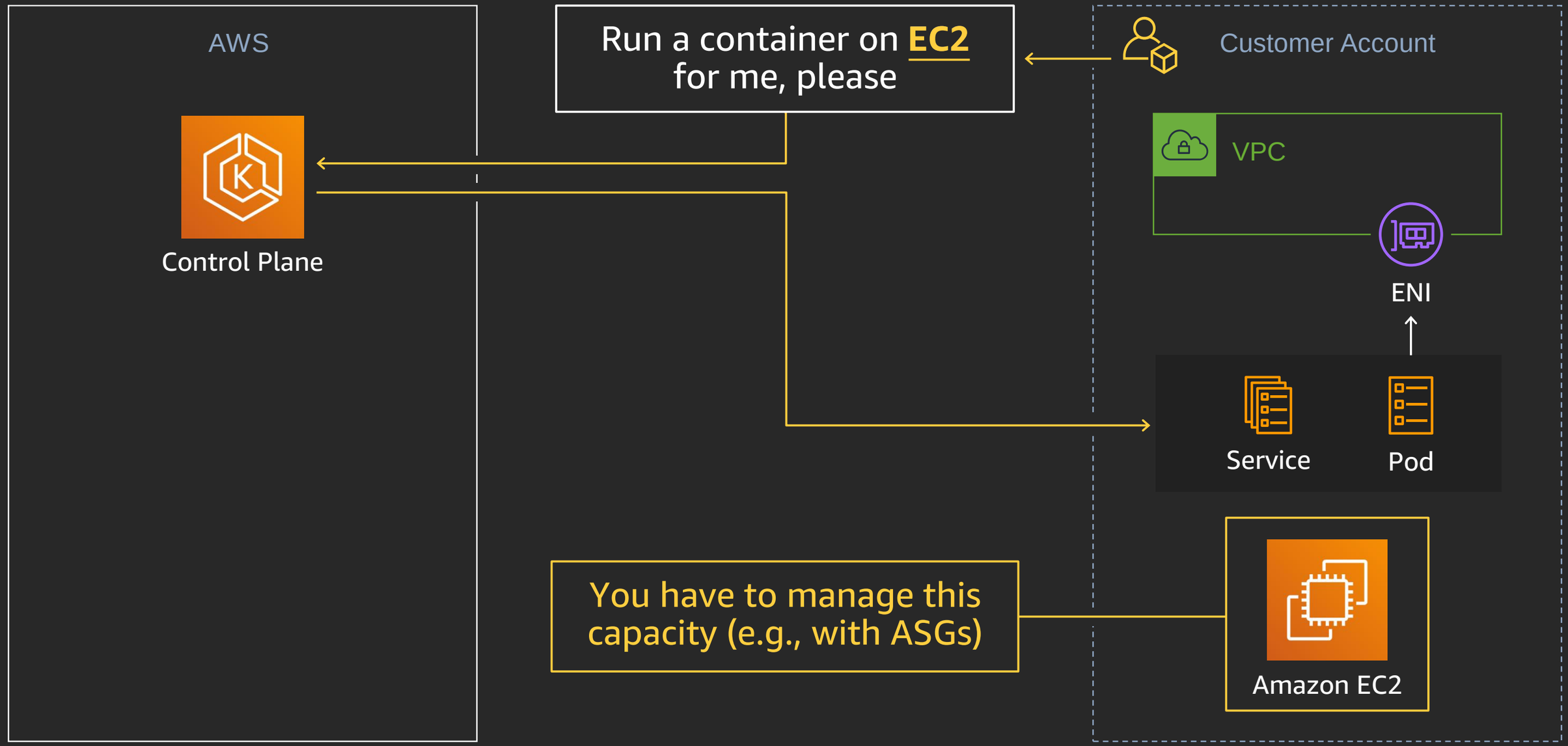
What matters for Fargate

Fargate is a serverless compute platform for containers on AWS

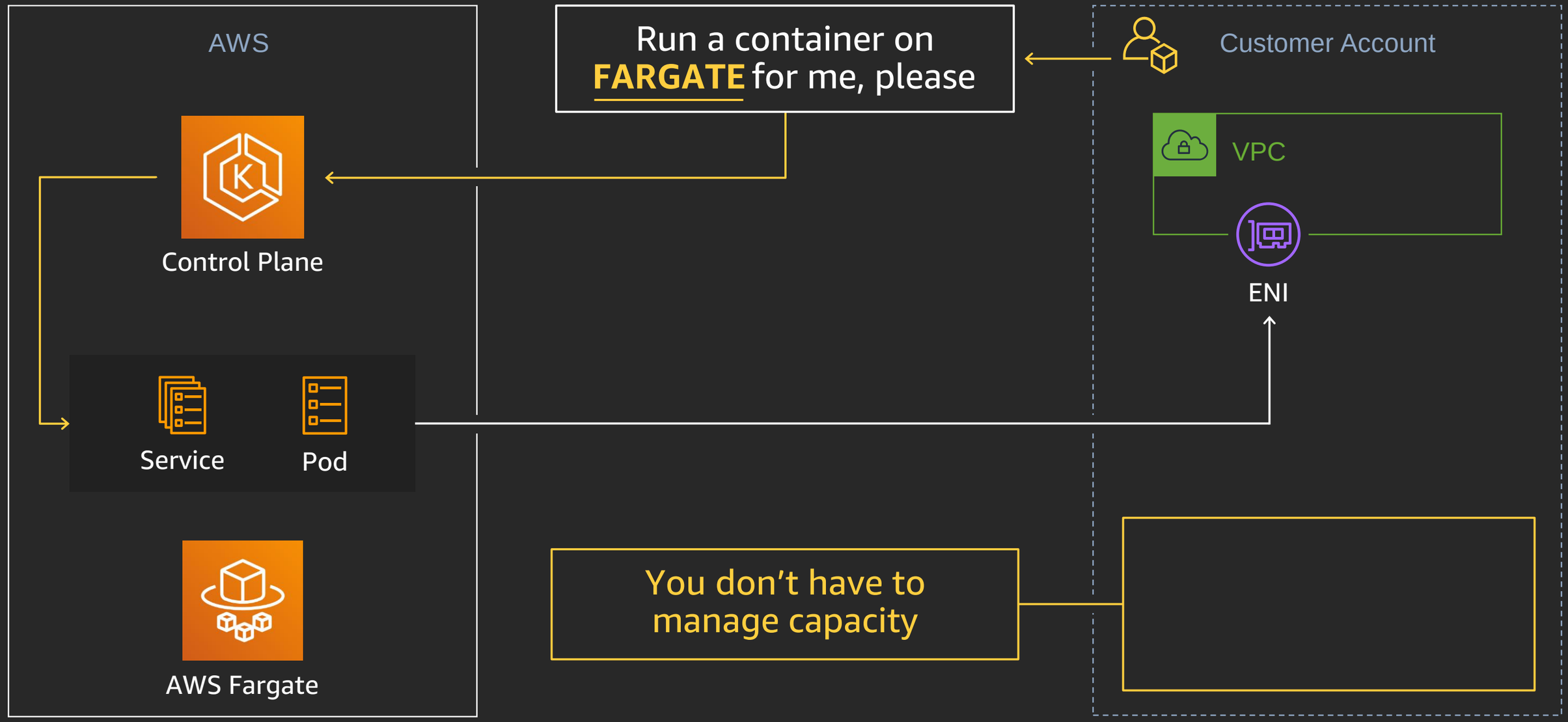
The differences between using EKS and ECS with Fargate are driven by the orchestration system

How EKS works on Fargate

The EC2 flow at 33,000 feet



The Fargate flow at 33,000 feet



Fargate Vs. (Un)Managed Nodes

	Fargate	Managed nodes	Unmanaged nodes
Units of work	Pod	Pod and EC2	Pod and EC2
Unit of charge	Pod	EC2	EC2

Fargate Vs. (Un)Managed Nodes

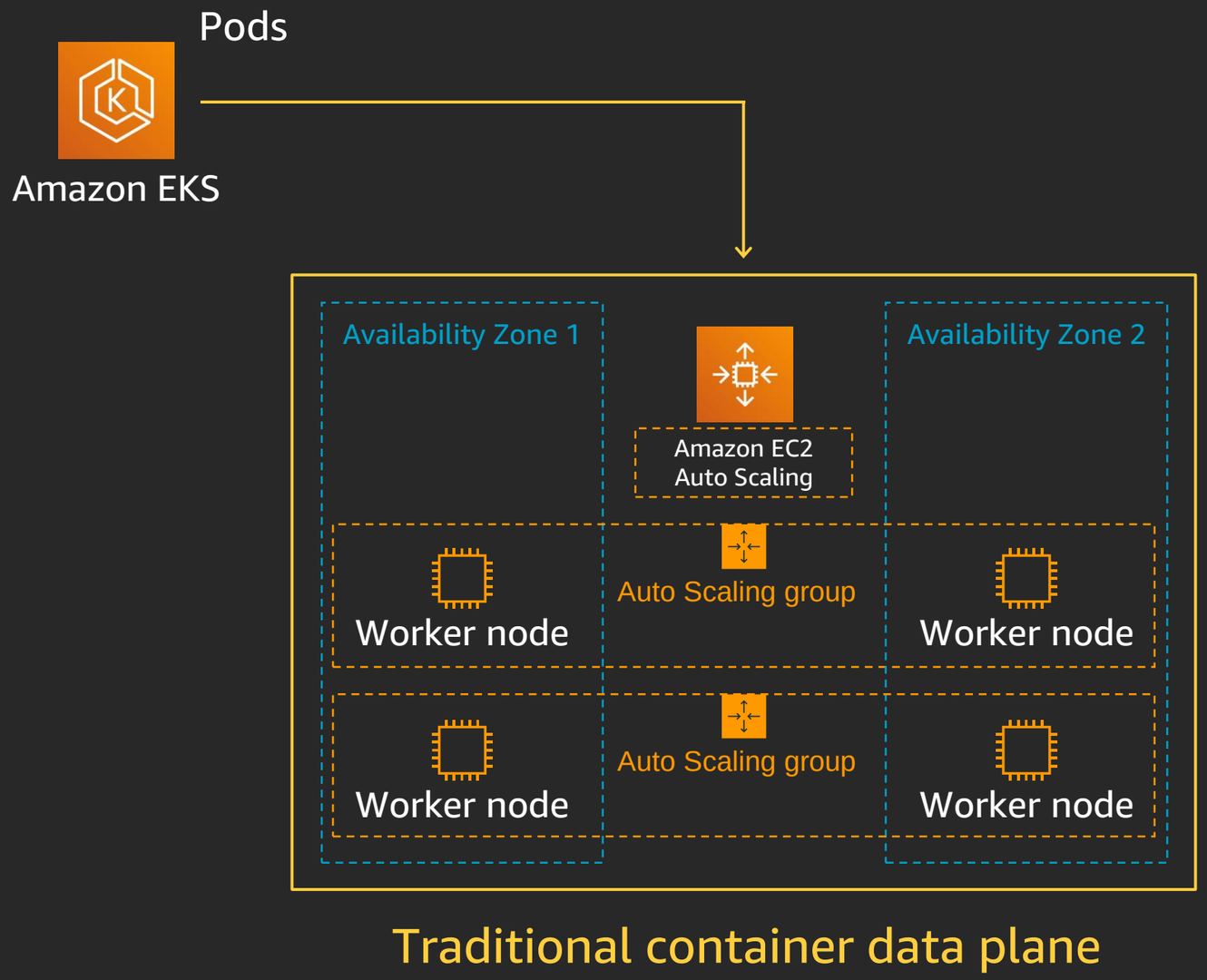
	Fargate	Managed nodes	Unmanaged nodes
Units of work	Pod	Pod and EC2	Pod and EC2
Unit of charge	Pod	EC2	EC2
Host lifecycle	There is no visible host	AWS (SSH is allowed)	Customer
Host AMI	There is no visible host	AWS vetted AMIs	Customer BYO

Fargate vs. (Un)Managed Nodes

	Fargate	Managed nodes	Unmanaged nodes
Units of work	Pod	Pod and EC2	Pod and EC2
Unit of charge	Pod	EC2	EC2
Host lifecycle	There is no visible host	AWS (SSH is allowed)	Customer
Host AMI	There is no visible host	AWS vetted AMIs	Customer BYO
Host : Pods	1 : 1	1 : many	1 : many

EKS data plane options

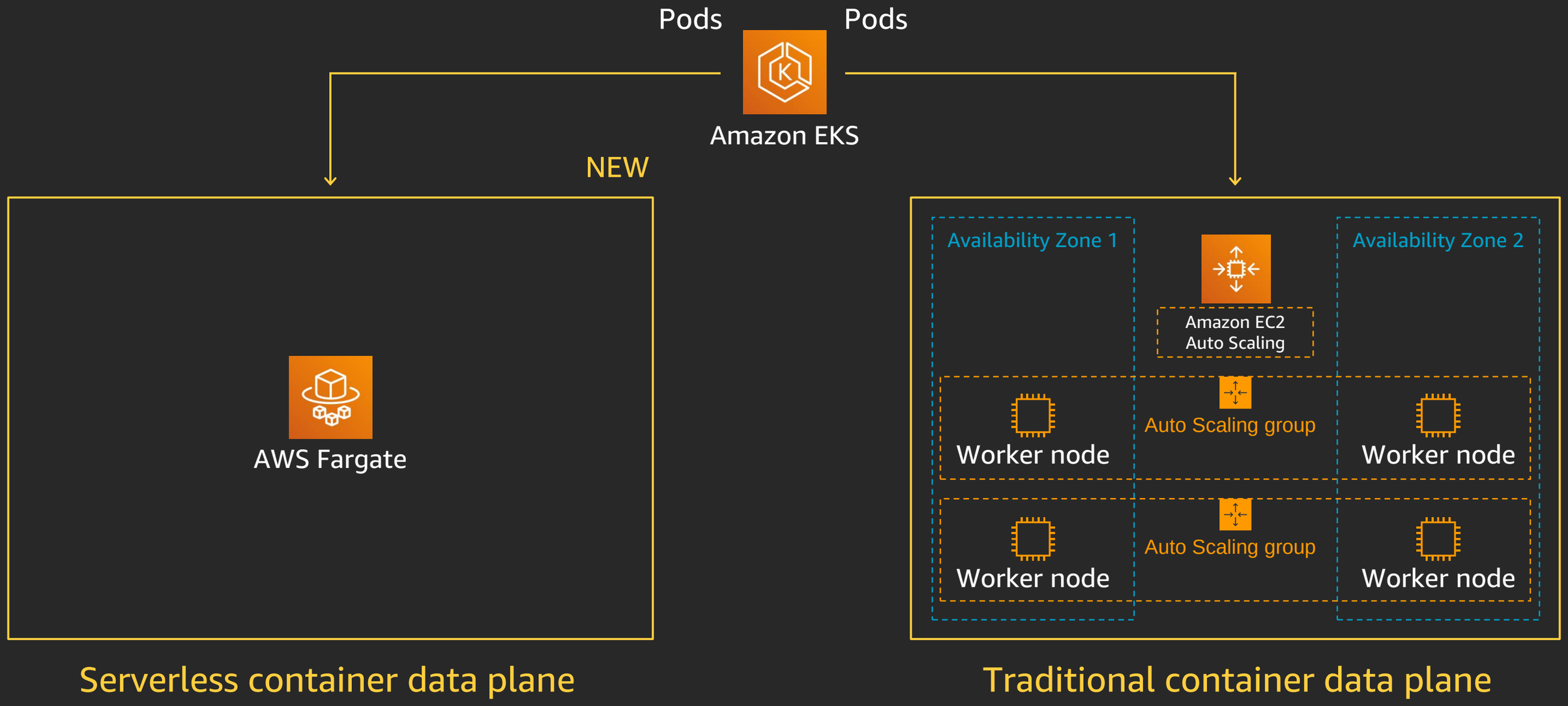
Worker nodes only



Traditional container data plane

EKS data plane options

Mixed mode

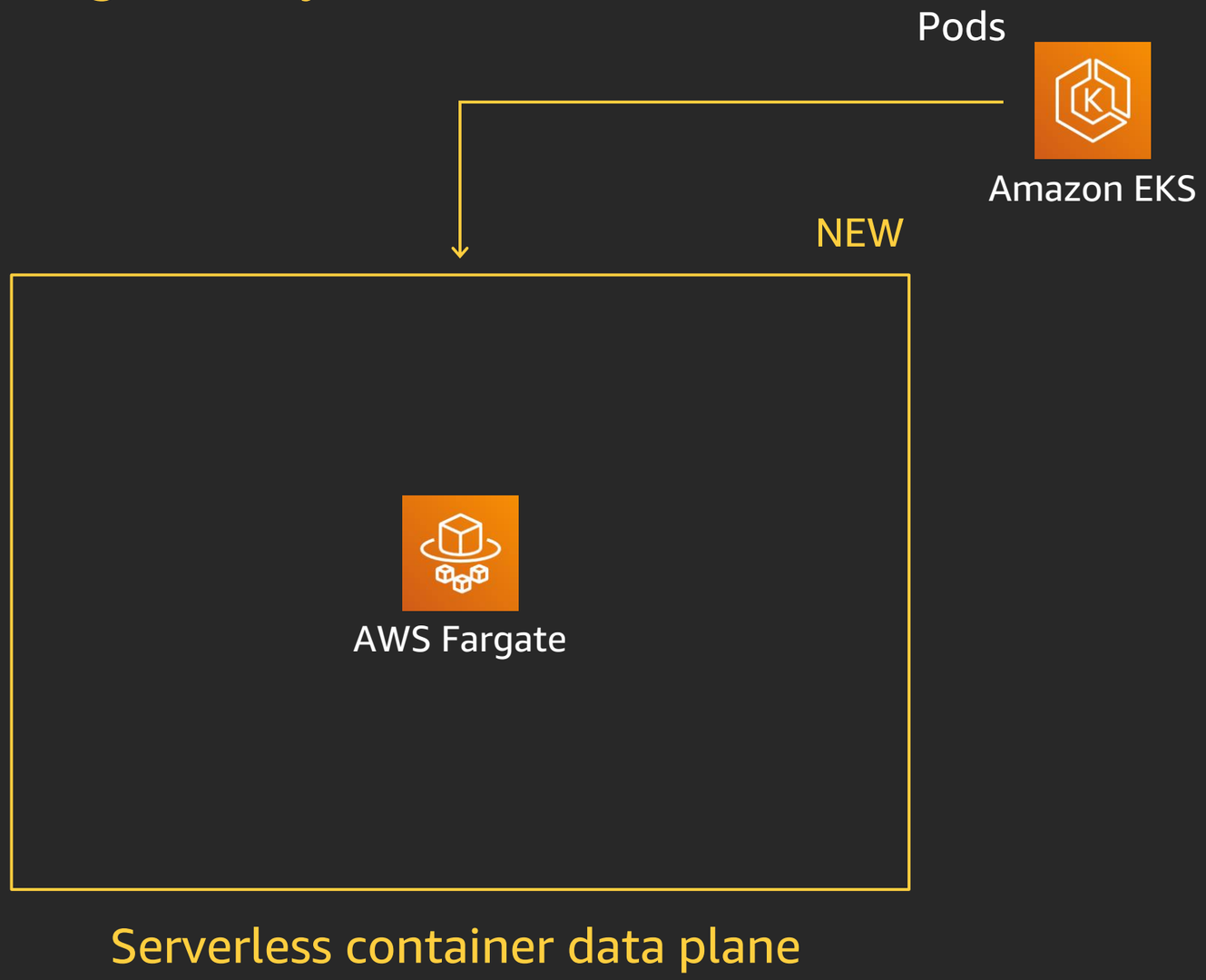


Serverless container data plane

Traditional container data plane

EKS data plane options

Fargate only

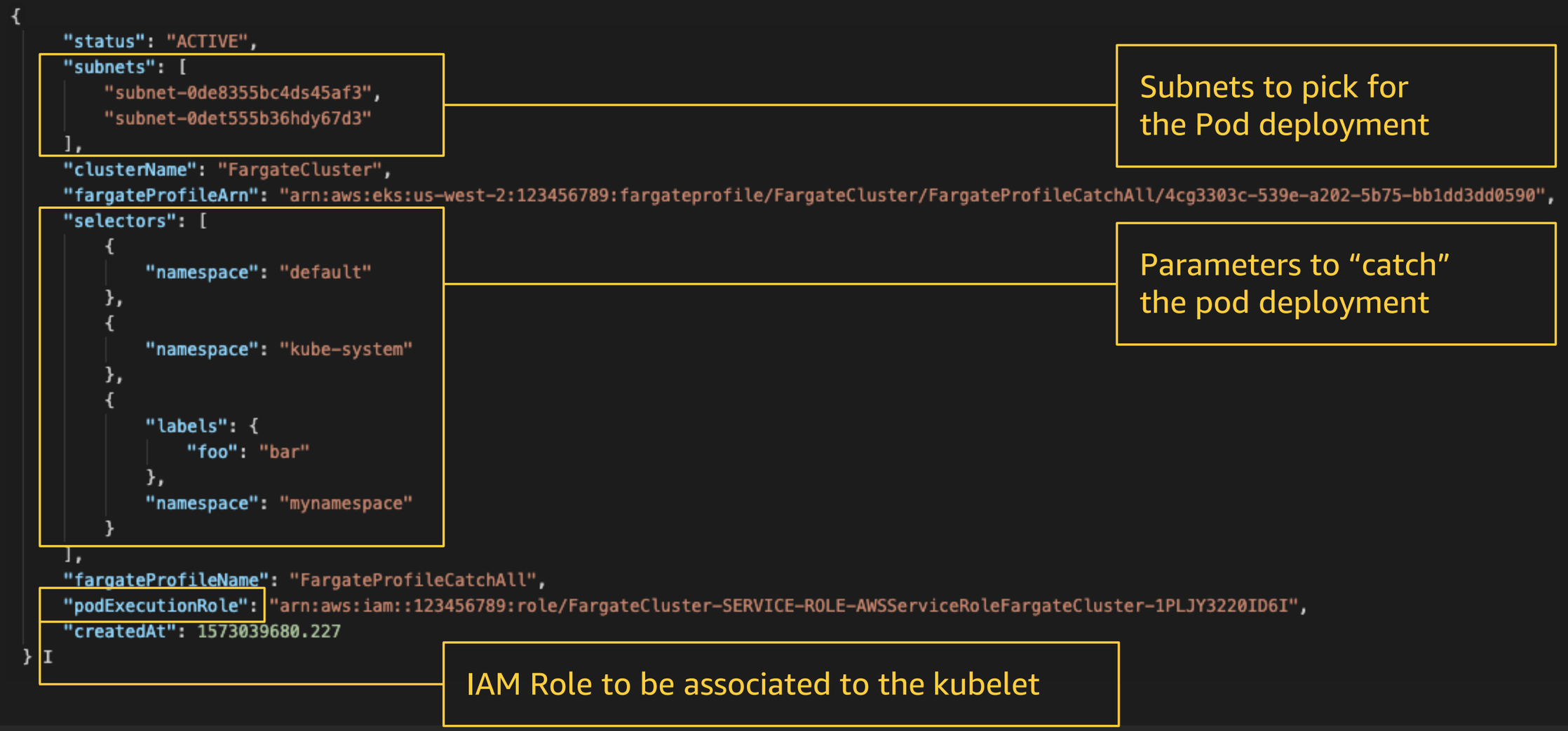


Demo

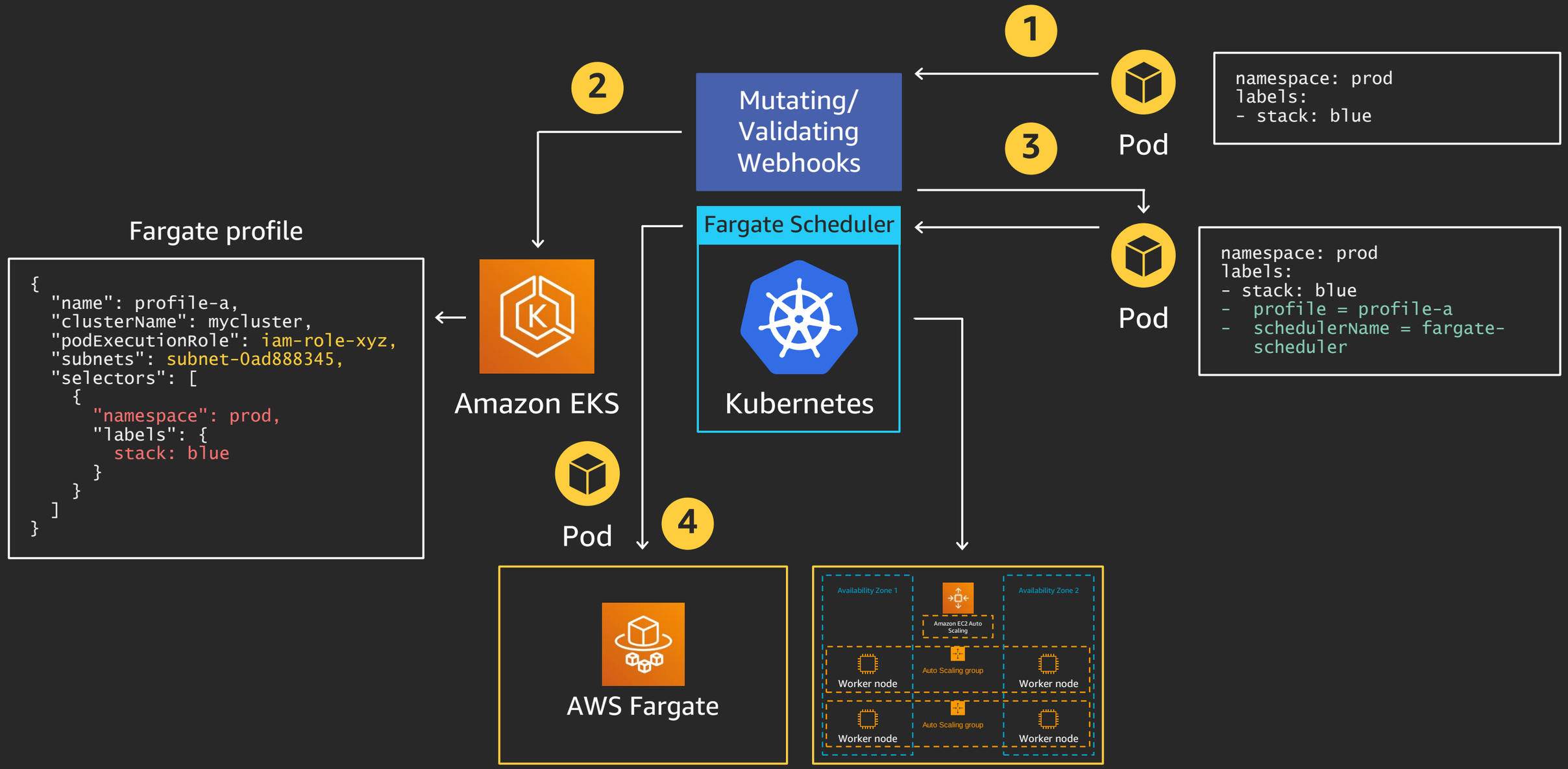
Kubernetes and EKS: Objects and constructs

Kubernetes and EKS: Objects and constructs

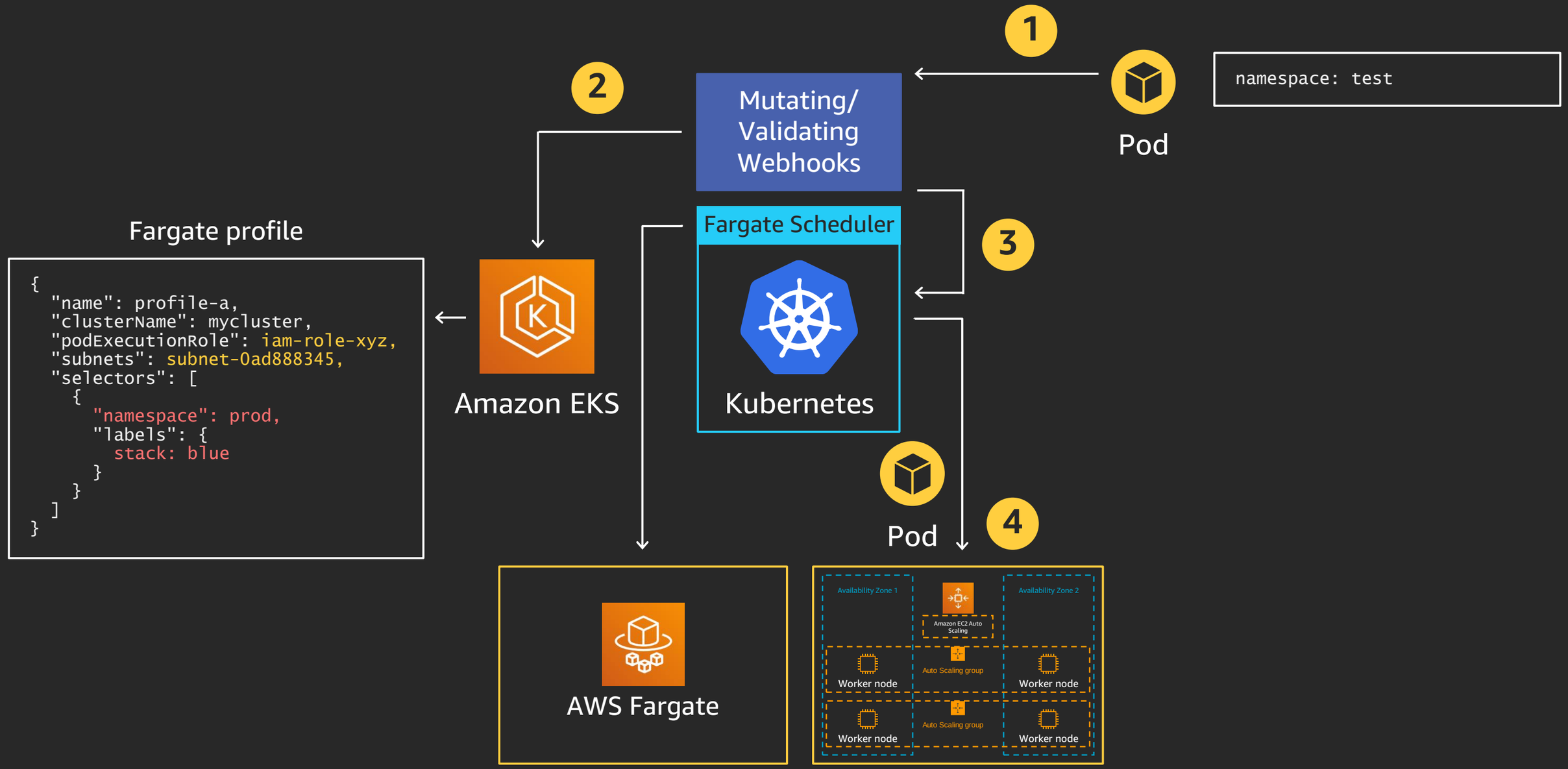
Fargate profile template



Simplified deployment flow



Simplified deployment flow



Need a custom pod spec to deploy to Fargate?

No

You can configure EKS to deploy to Fargate…
without touching your pod spec

Example

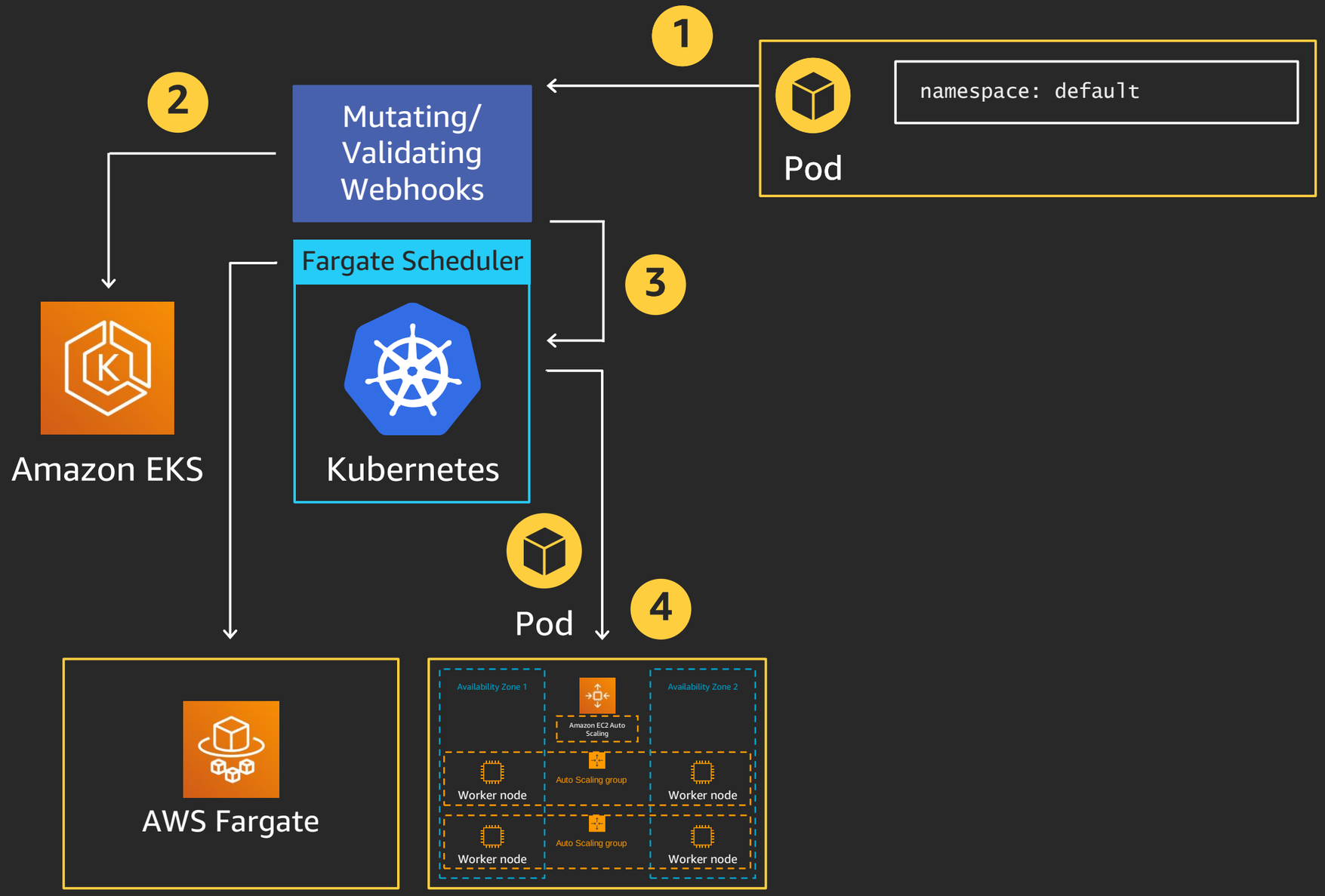
Run a pod on standard worker nodes

Create a Fargate profile that matches the pod namespace (and optionally, labels)

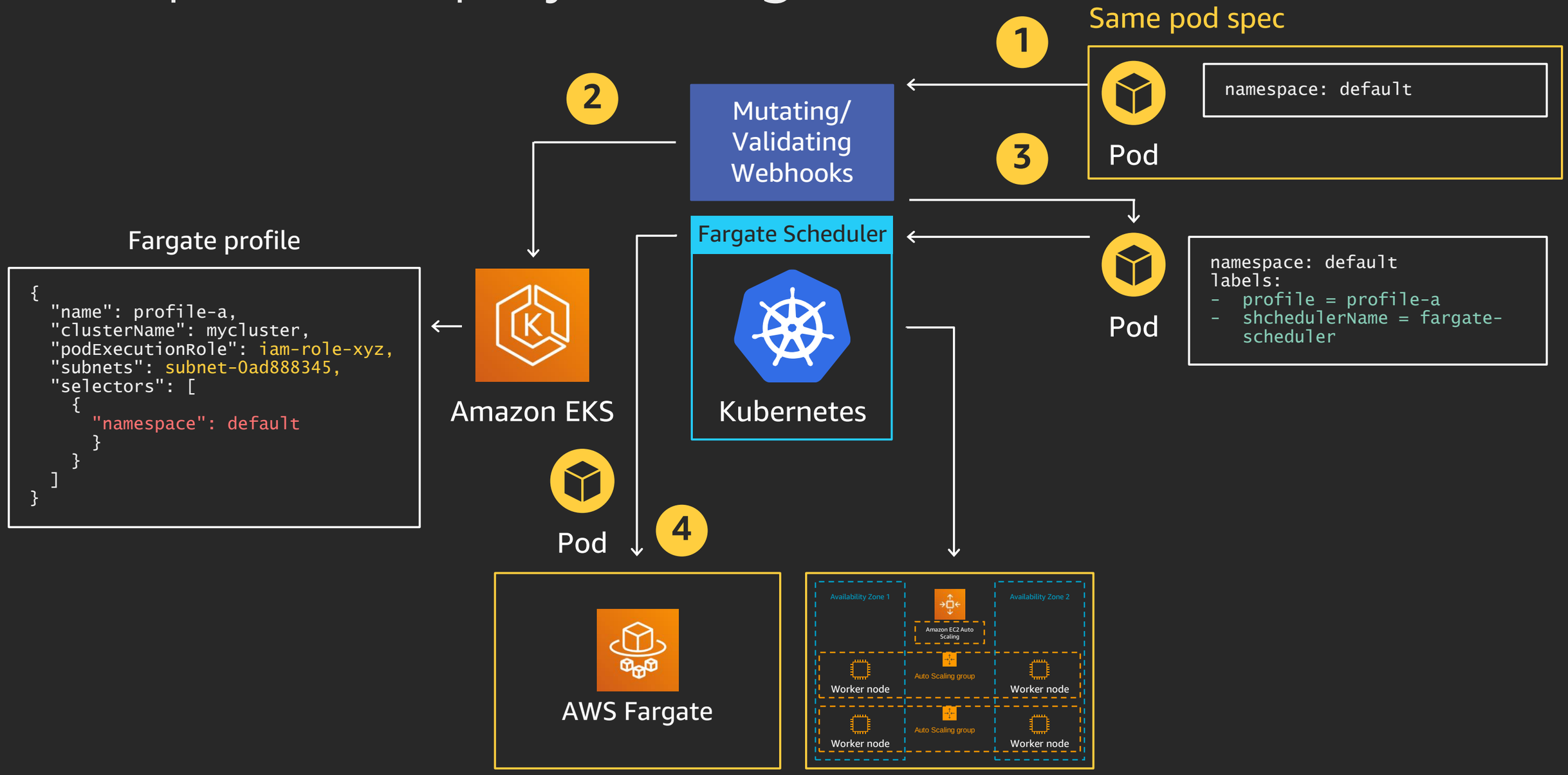
Kill the pod

See the pod restart on Fargate

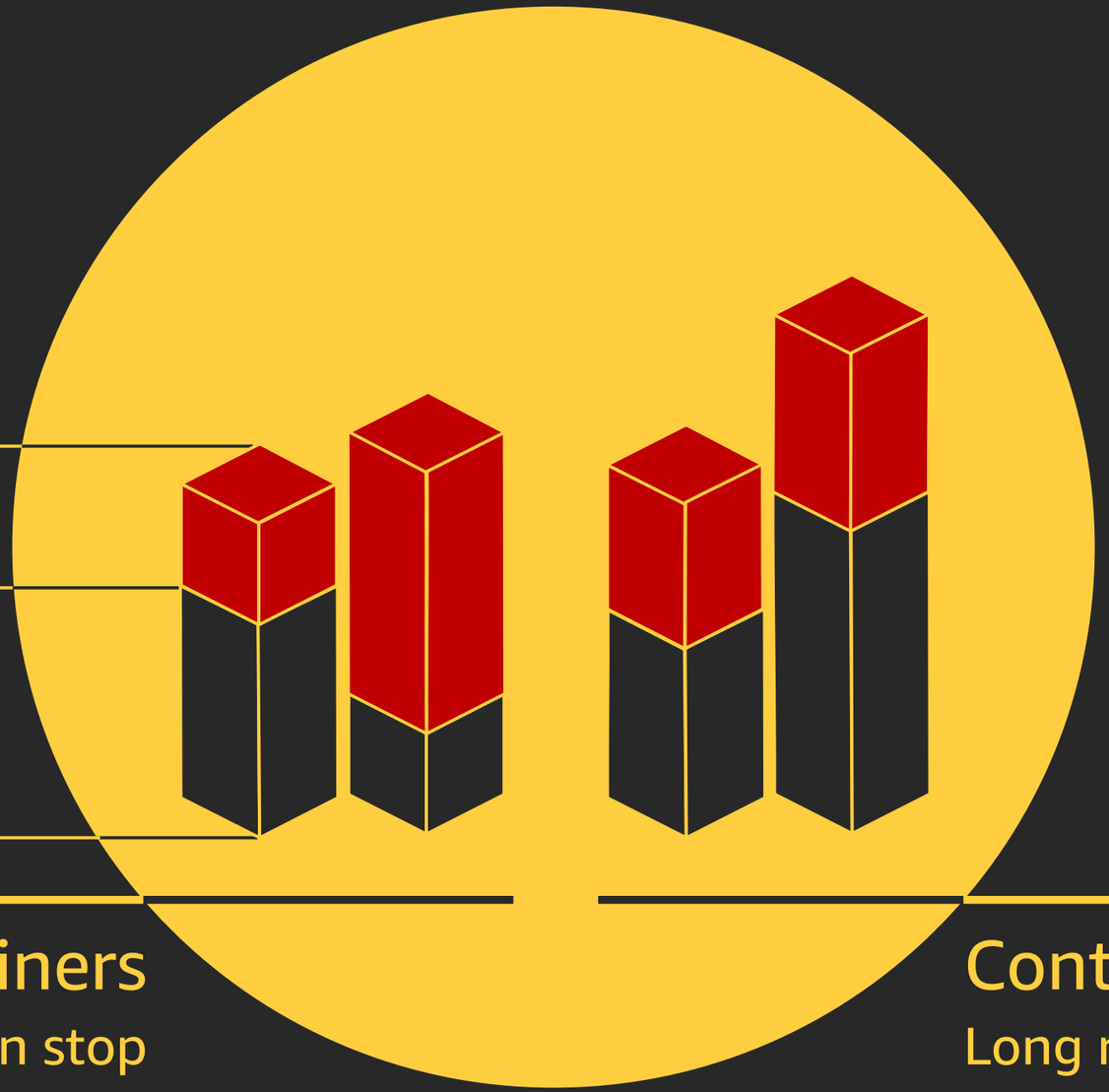
Example: Deploy to workers



Example: Re-deploy to Fargate



How do we pick the size of the pod?



Limit

Request

Init containers

Start sequentially and then stop

Containers

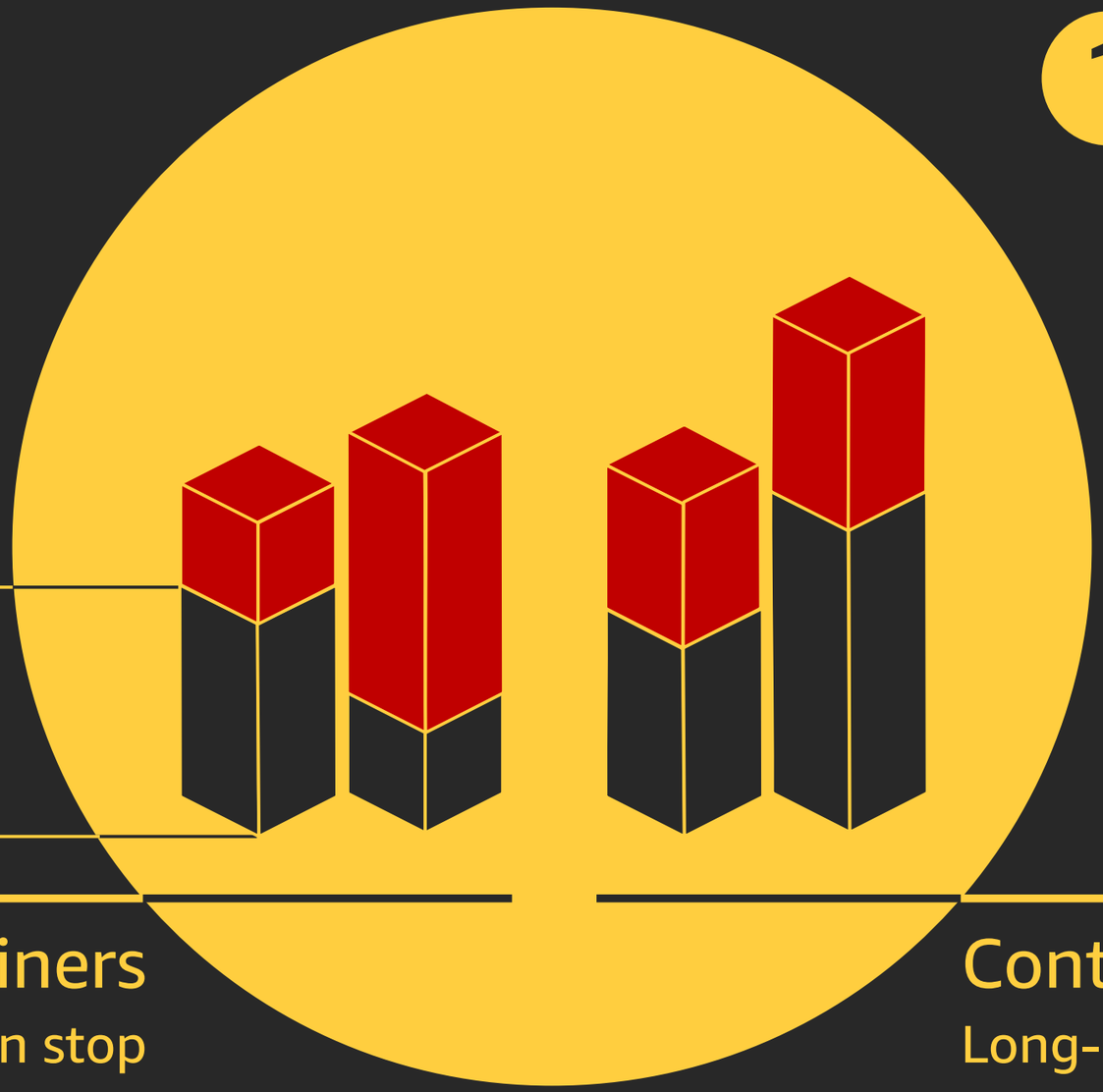
Long running

This is for both Memory and CPU dimensions

```yaml
containers:
- name: wp
  image: wordpress
  resources:
    limits:
      memory: "128Mi"
      cpu: "500m"
    requests:
      memory: "64Mi"
      cpu: "250m"
```

How do we pick the size of the pod?

1 Only requests are considered



Request

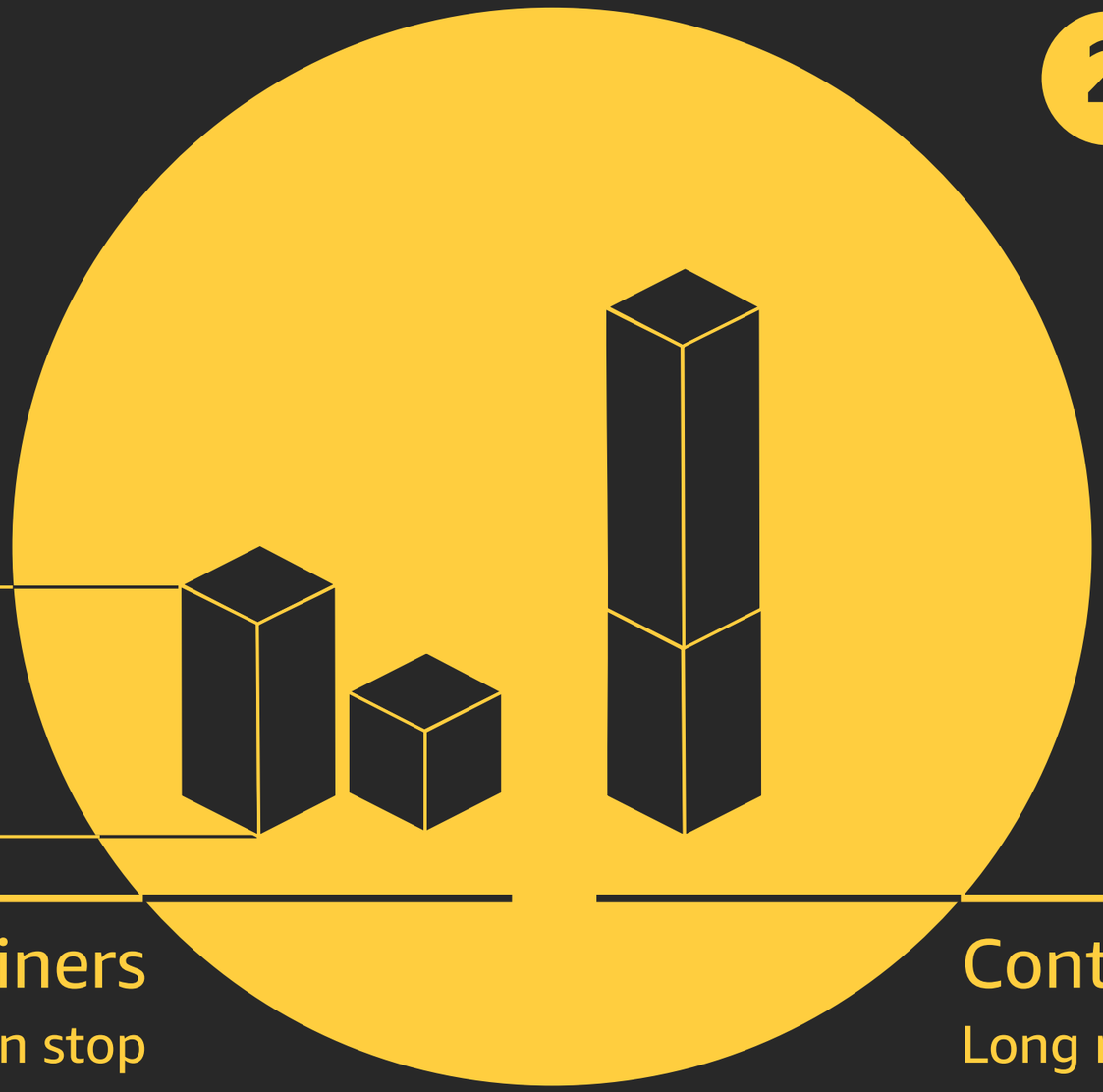
Init containers

Start sequentially and then stop

Containers

Long-running

How do we pick the size of the pod?



2

Requests for ALL long-running containers are added together

Request

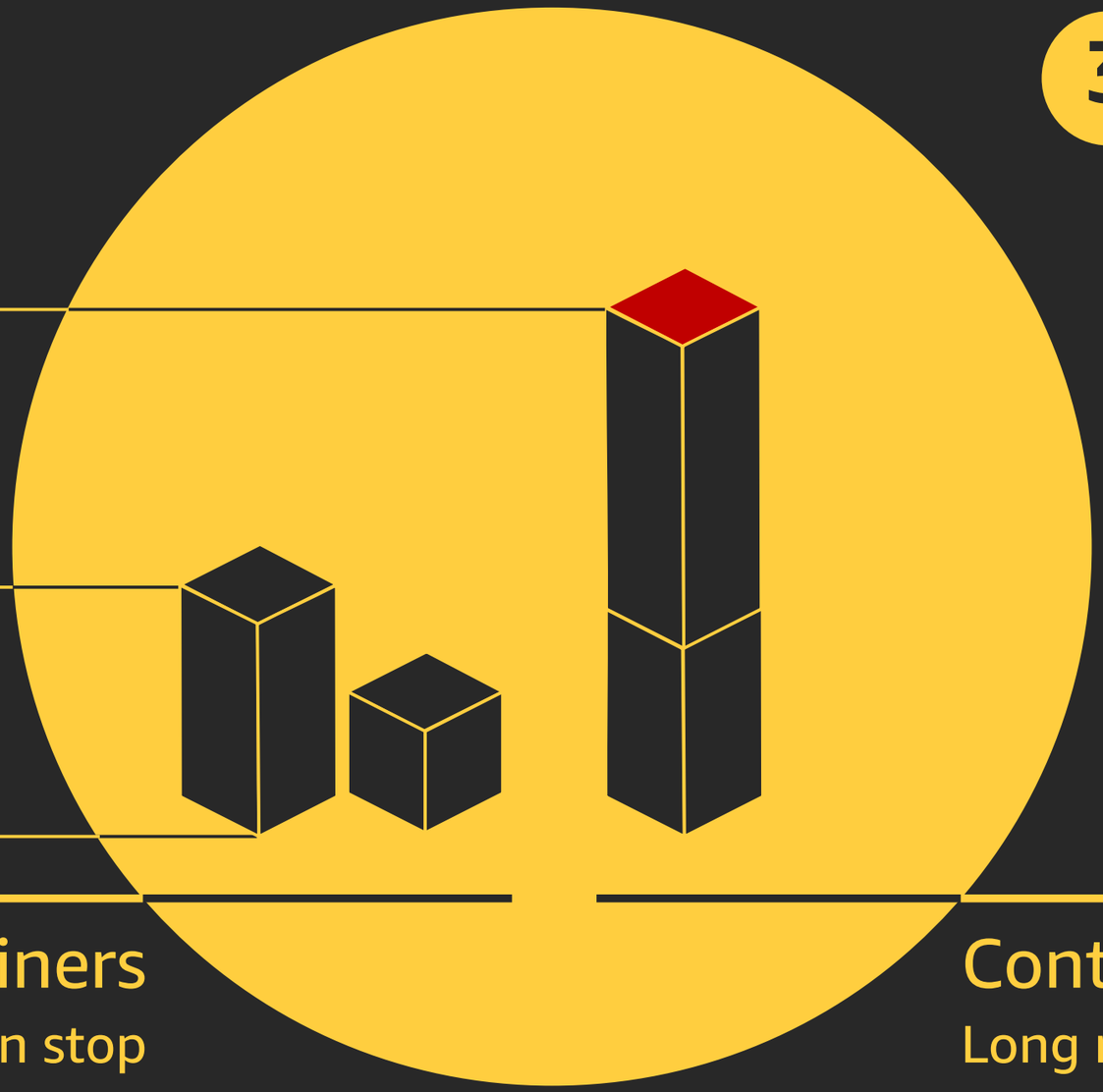
Init containers

Start sequentially and then stop

Containers

Long running

How do we pick the size of the pod?



This

Request

Init containers

Start sequentially and then stop

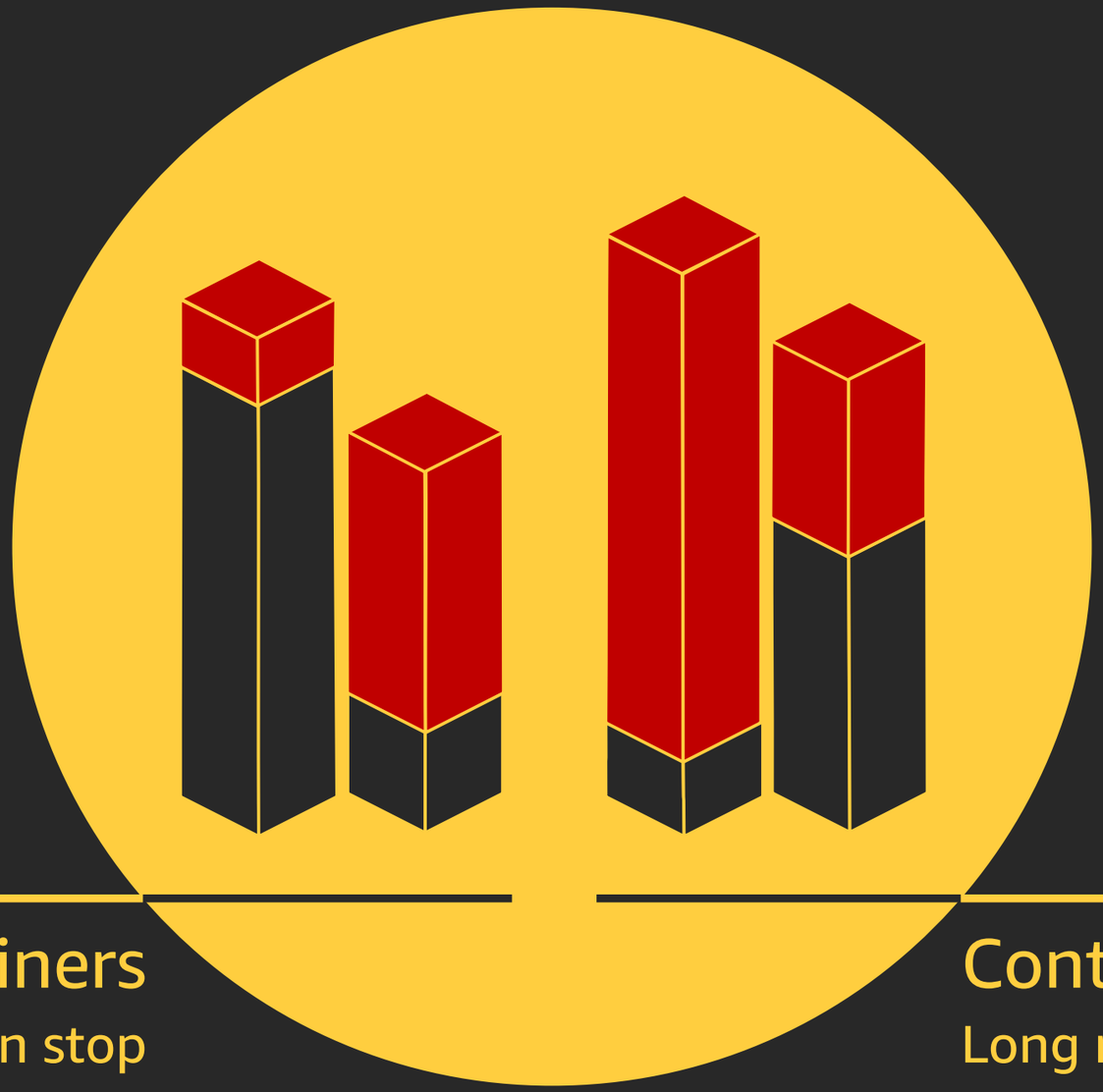
Containers

Long running

3

The biggest number is taken and used to size the Fargate pod

How do we pick the size of the pod?



Another example?

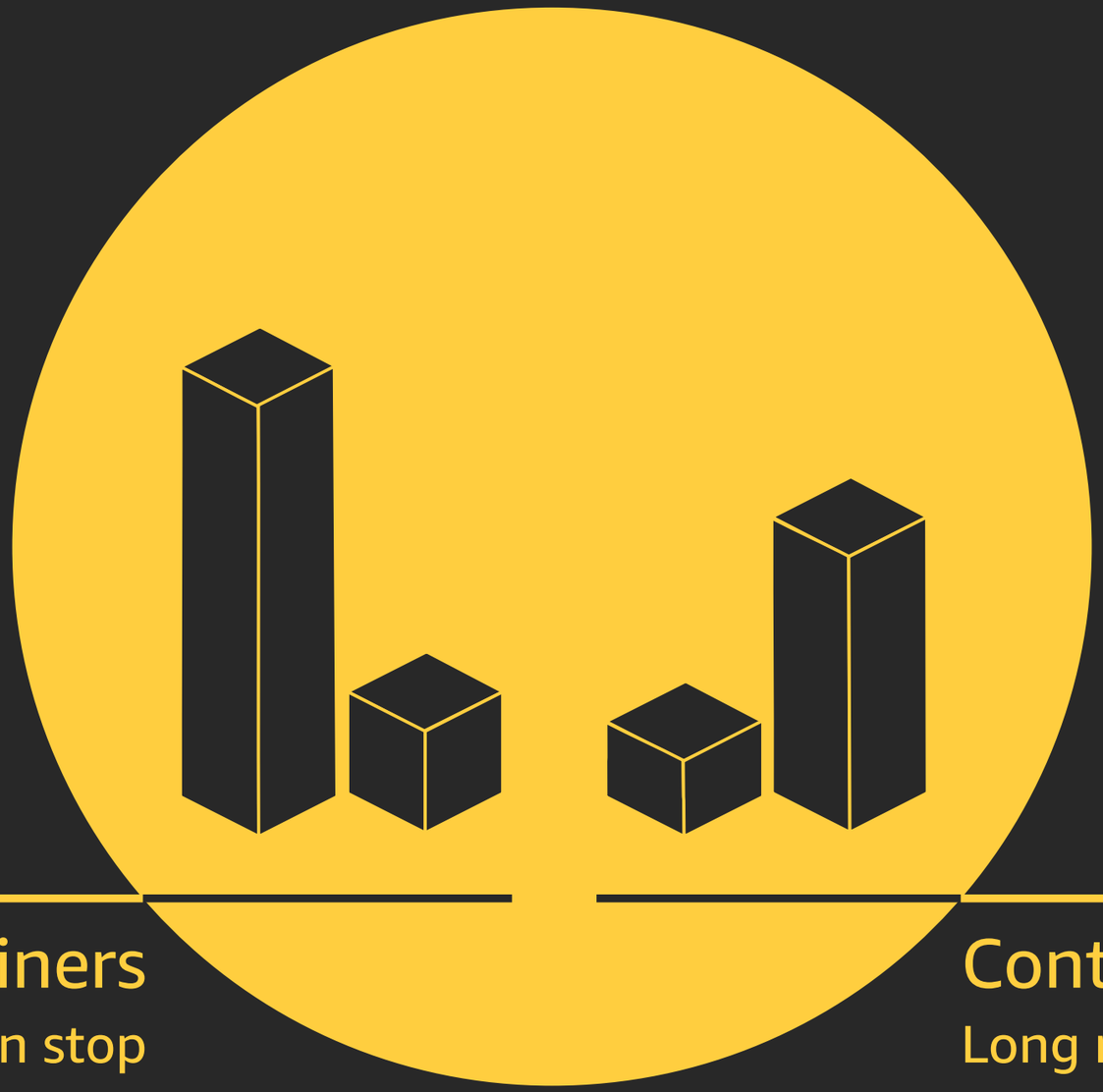
Init containers

Start sequentially and then stop

Containers

Long running

How do we pick the size of the pod?



Let's consider the requests only

Init containers

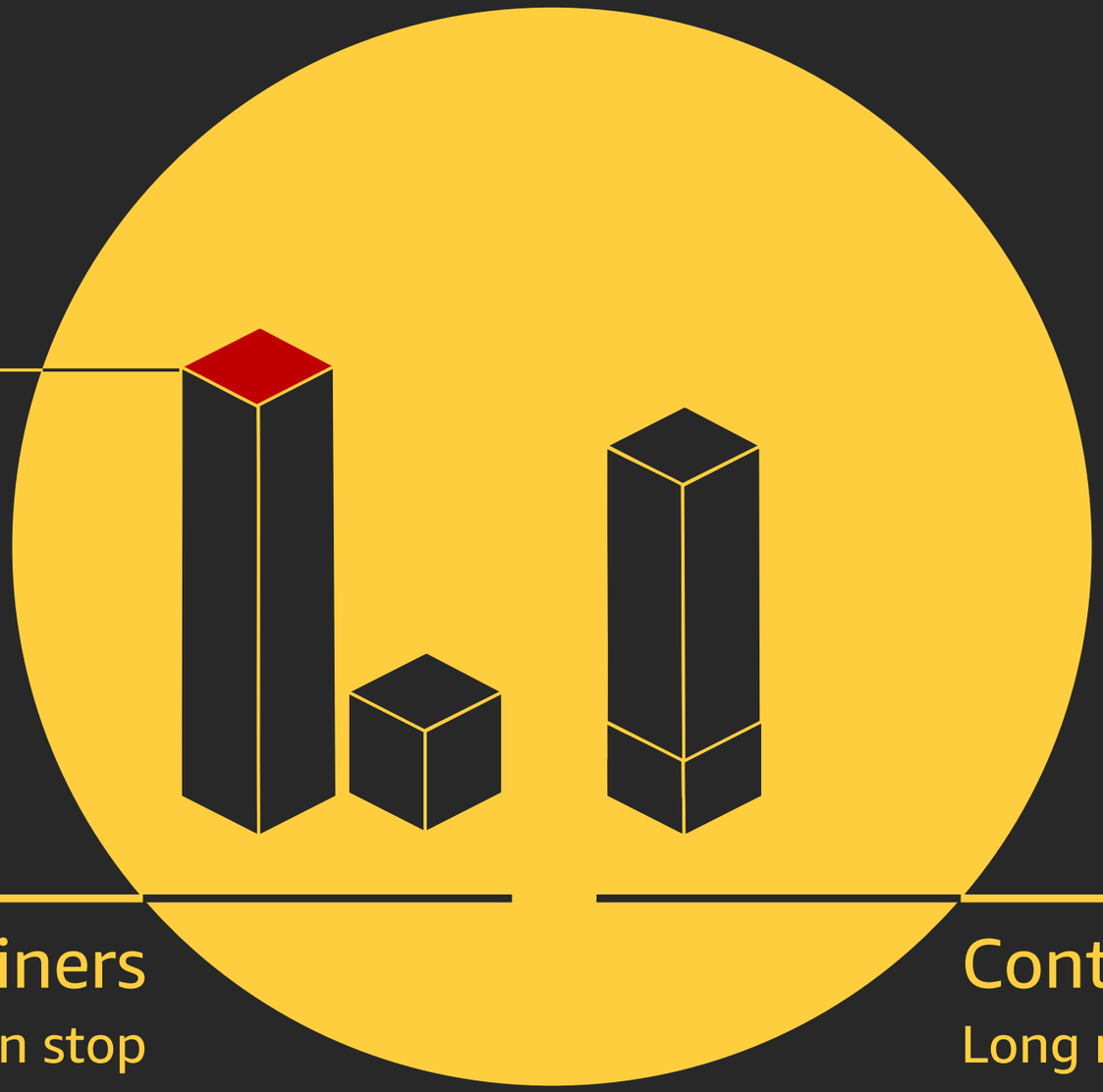
Start sequentially and then stop

Containers

Long running

How do we pick the size of the pod?

This



Let’s add all long-running containers and pick the biggest number

Init containers

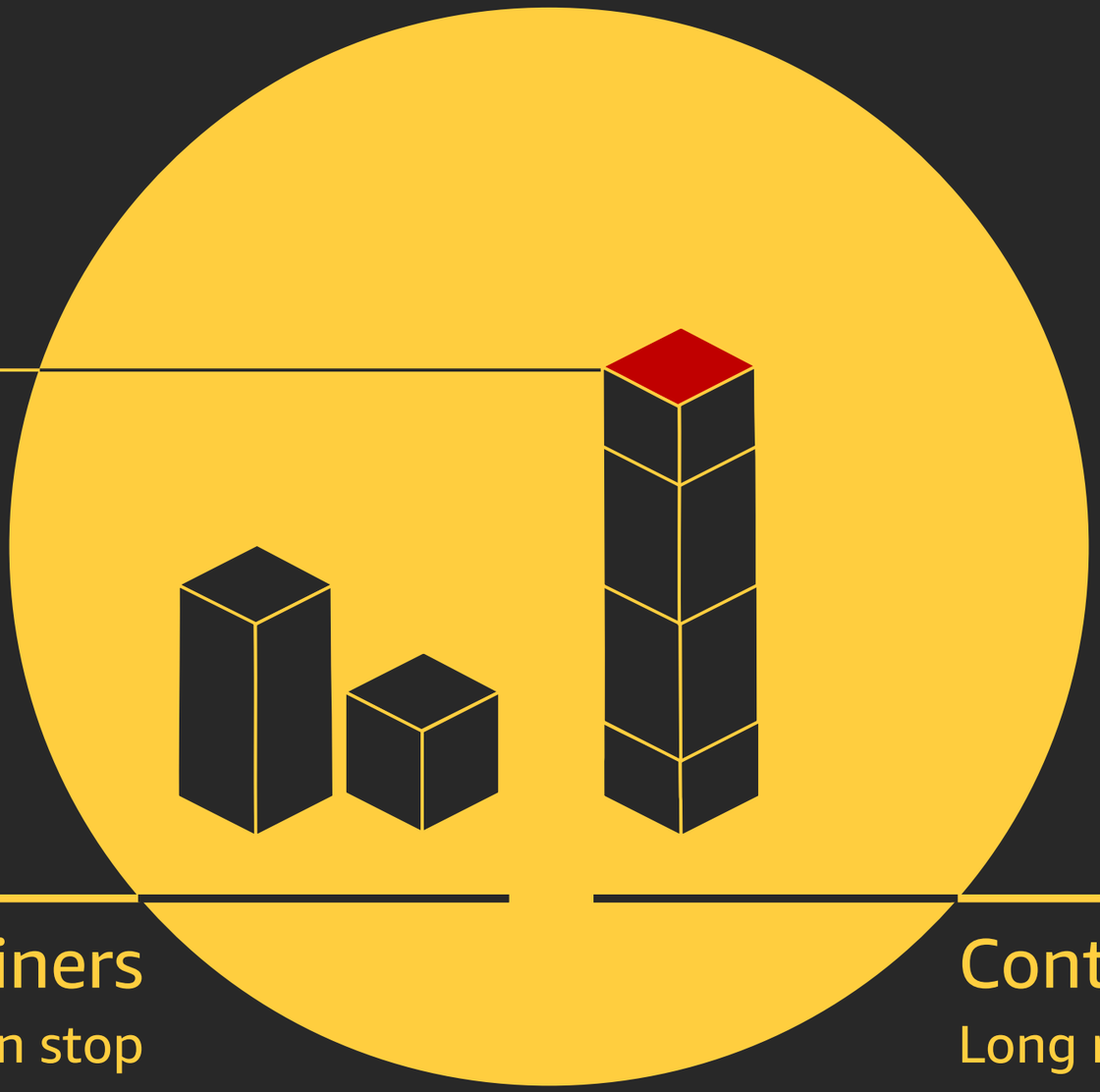
Start sequentially and then stop

Containers

Long running

How do we pick the size of the pod?

This



How do we go from this Pod config example to a Fargate size?

Init containers

Start sequentially and then stop

Containers

Long running

How do we pick the size of the pod?

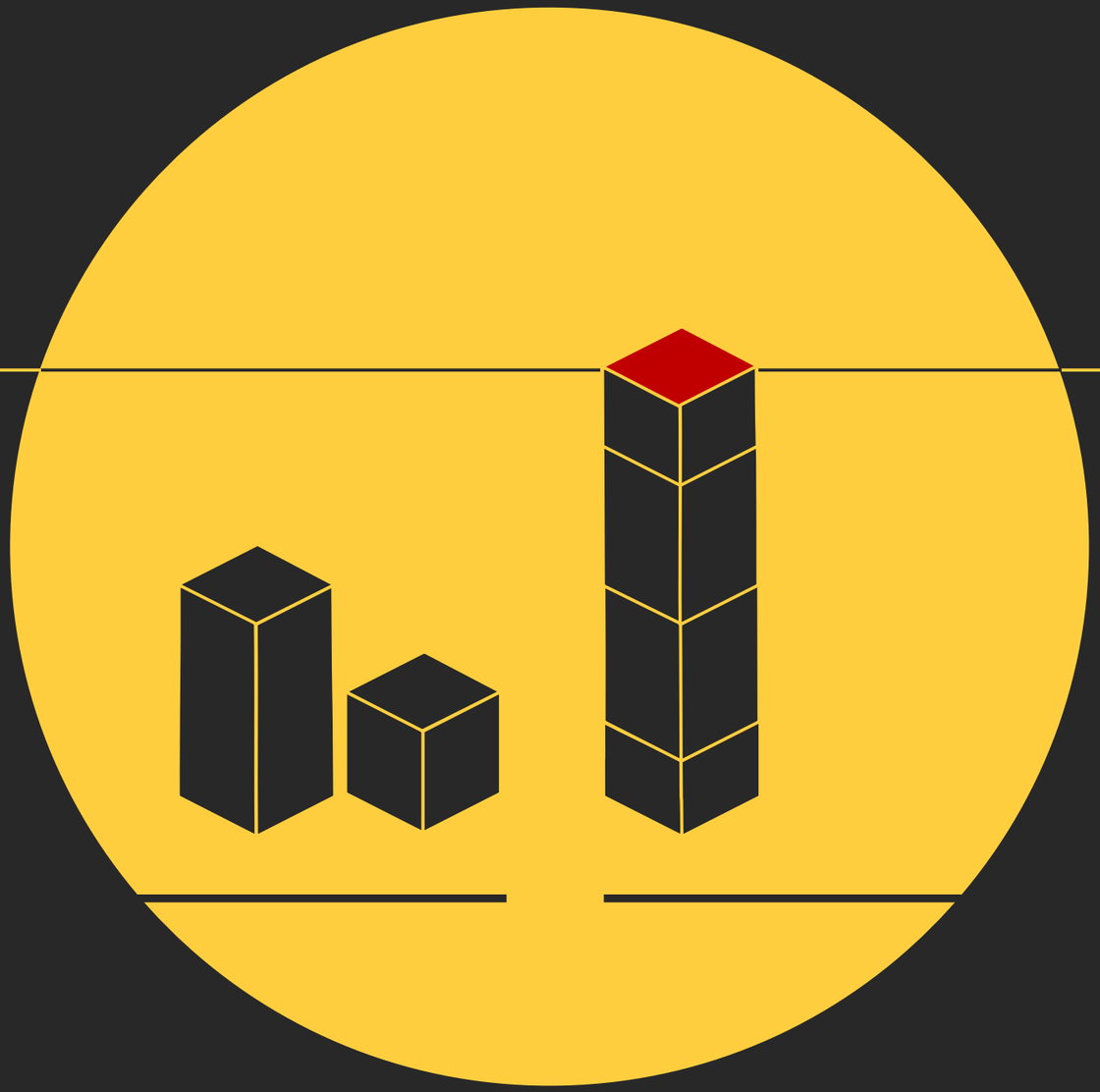
This

CPU	Memory
256 (.25 vCPU)	512MB, 1GB, 2GB
512 (.5 vCPU)	1GB, 2GB, 3GB, 4GB
1024 (1 vCPU)	2GB, 3GB, 4GB, 5GB, 6GB, 7GB, 8GB
2048 (2 vCPU)	Between 4GB and 16GB in 1GB increments
4096 (4 vCPU)	Between 8GB and 30GB in 1GB increments

Fargate task size combinations

How do we pick the size of the pod?

This



Kubernetes components

+256MB

CPU

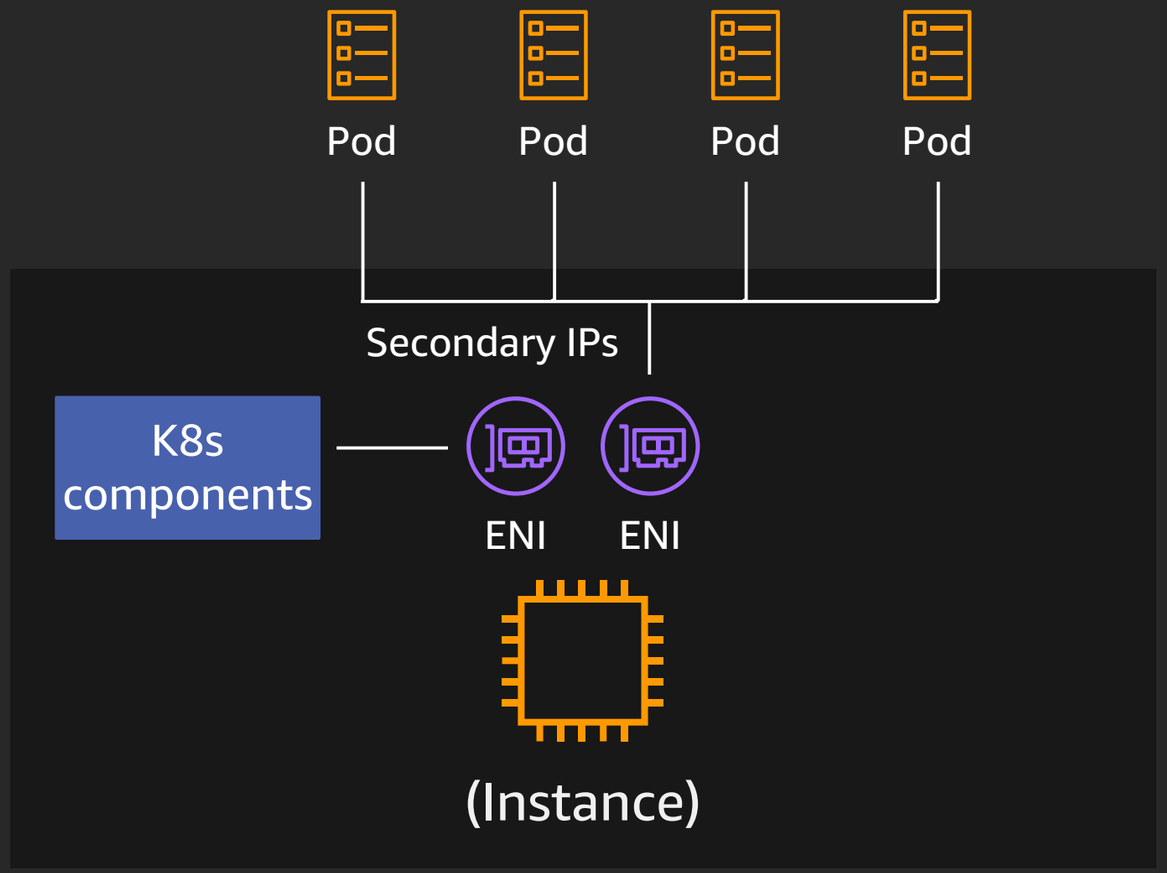
Closest config (rounded up) is picked

MEM

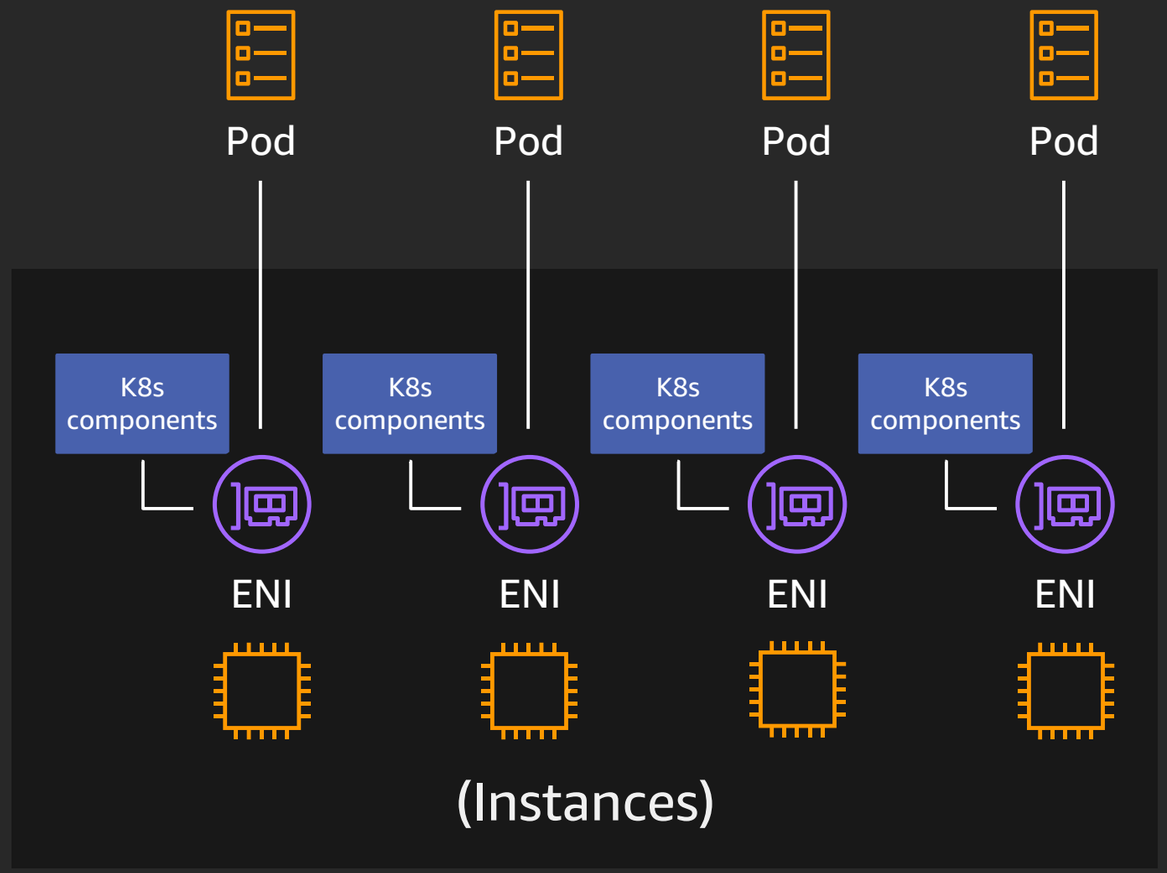
CPU	Memory
256 (.25 vCPU)	512MB, 1GB, 2GB
512 (.5 vCPU)	1GB, 2GB, 3GB, 4GB
1024 (1 vCPU)	2GB, 3GB, 4GB, 5GB, 6GB, 7GB, 8GB
2048 (2 vCPU)	Between 4GB and 16GB in 1GB increments
4096 (4 vCPU)	Between 8GB and 30GB in 1GB increments

Fargate task size combinations

Networking architecture

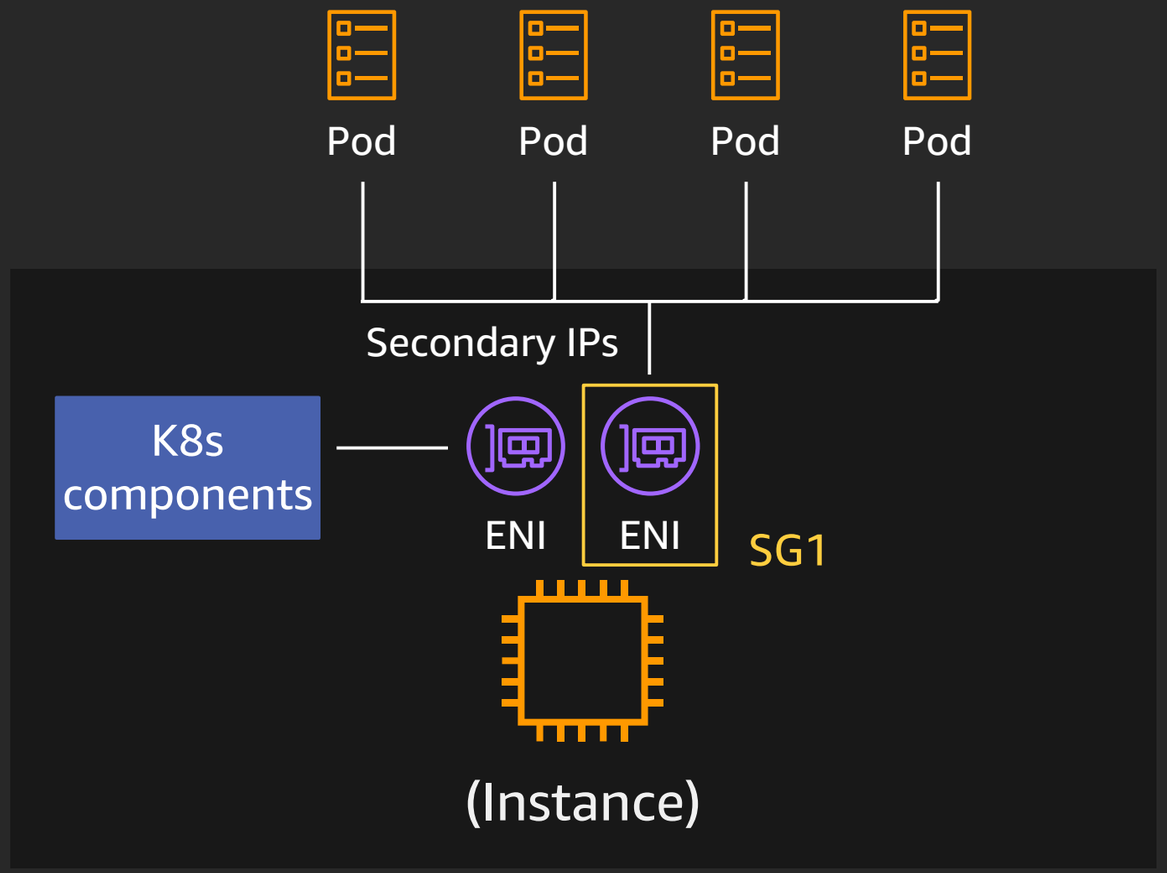


Worker nodes data plane

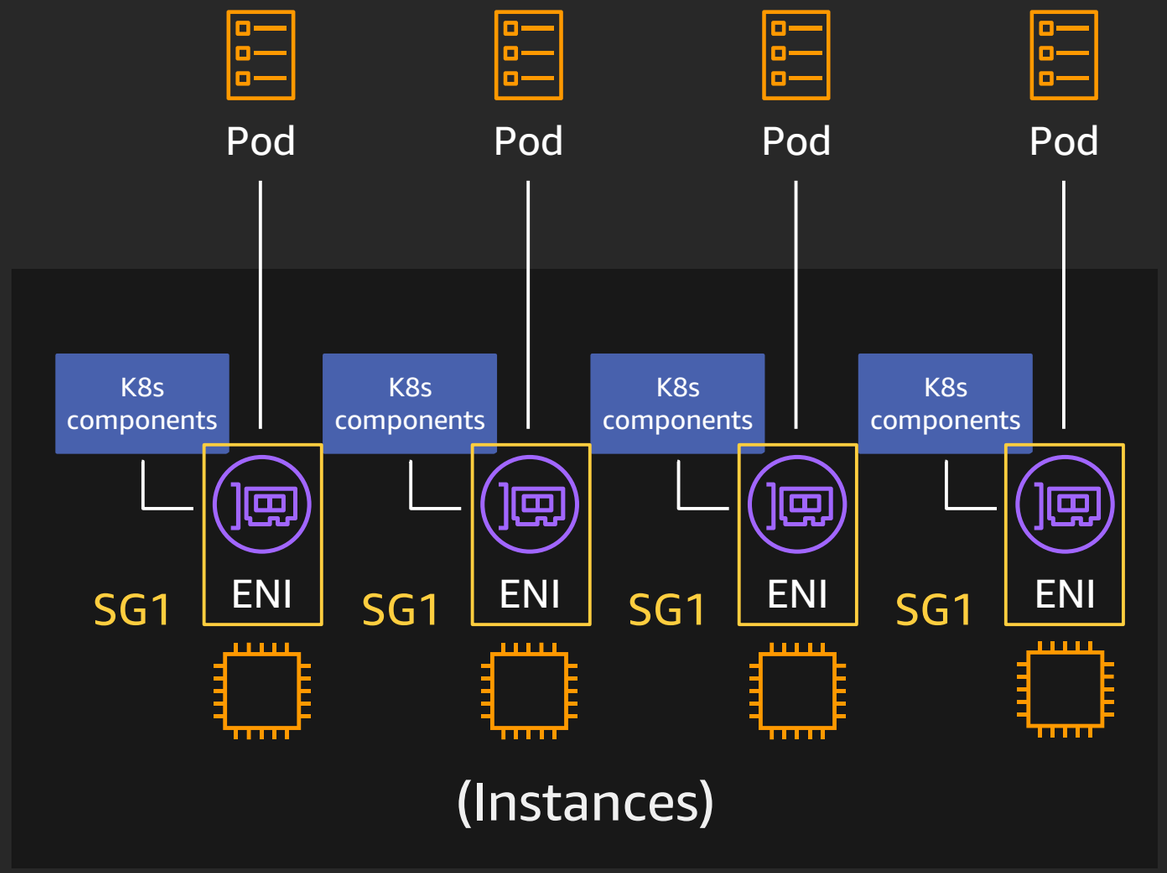


Fargate data plane

Security group considerations (at GA)

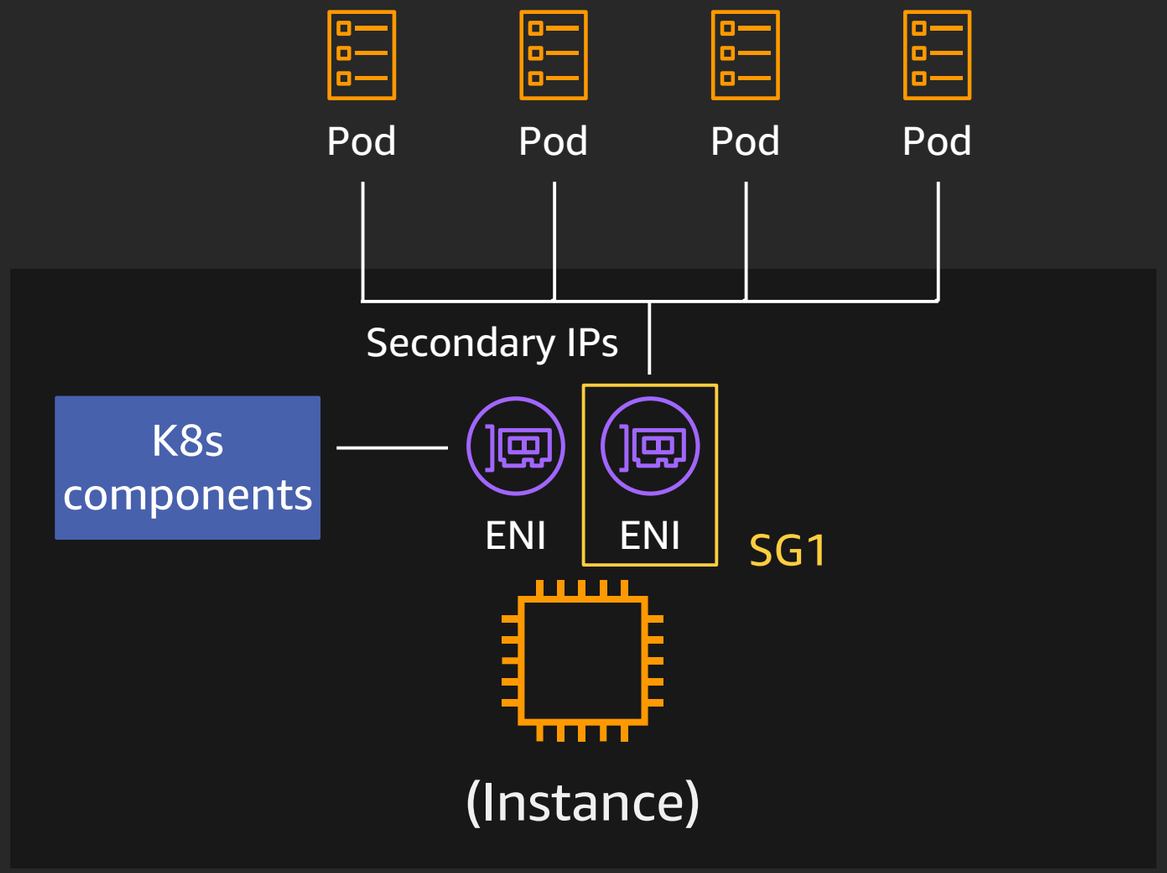


Worker nodes data plane

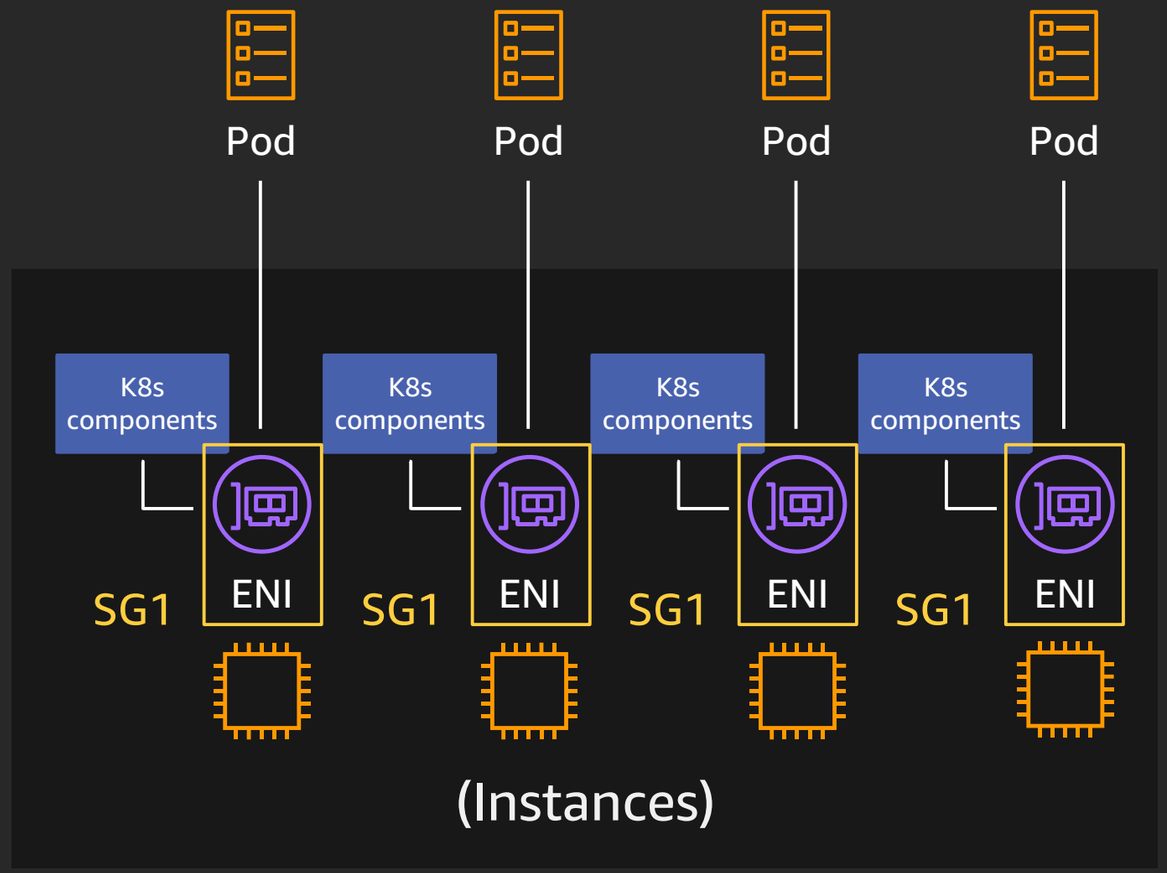


Fargate data plane

Security group considerations (at GA)



Worker nodes data plane



Fargate data plane

At GA the same SG will apply to ALL ENIs

Load Balancers considerations

ALB Ingress works as it normally does

Latest version includes the required code changes to make it work

NLB support is coming soon

Stay tuned

CLB will not work because it must target EC2 instances

There are no EC2 instances with EKS/Fargate

Storage options with EKS for Fargate

Fargate provides a local storage space for containers to share

This space is ephemeral and only lives for the time the pod lives

Persistent storage for Fargate is a frequent ask from customers

We are investigating the possibilities

EKSCTL support

EKSCTL supports Fargate and EC2 worker nodes

Make sure you use the latest version

It is possible to create a Fargate-only cluster

Or a combination of Fargate and managed node groups

EKSCTL takes care of some undifferentiated heavy lifting

Such as creating the Fargate profiles and more

```
> eksctl create cluster --fargate
```

Getting started

Recap: EKS for Fargate introduces UX changes

Things you no longer need to do

- Manage Kubernetes worker nodes
- Pay for unused capacity
- Use K8s Cluster Autoscaler (CA)

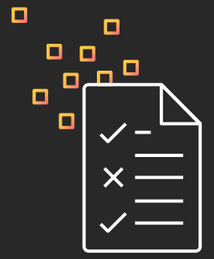
Things you get out of the box

- VM isolation at pod level
- Pod level billing
- Easy chargeback in multi-tenant scenarios

Things you can't do (for now)

- Deploy Daemonsets
- Use service type LoadBalancer (CLB/NLB)
- Running privileged containers
- Run stateful workloads

Limits: Things to keep in mind



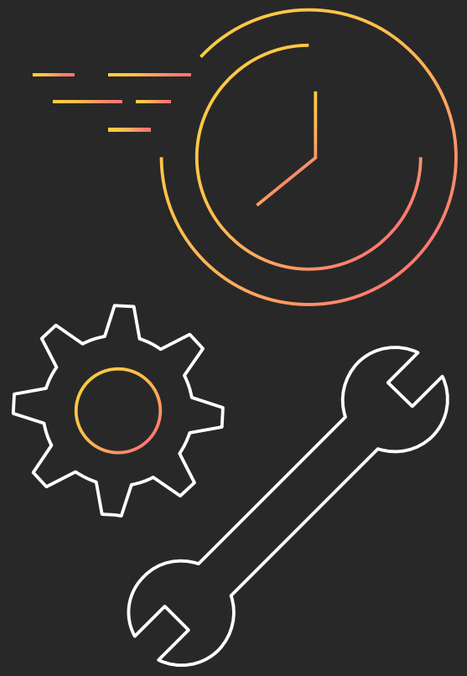
AWS accounts have a soft limit of 100 Fargate tasks/pods per region

You increase this limit

Due to the nature of the solution, there's a limit of 5,000 pods per cluster

K8s tests up to
5,000 workers per cluster

Scalability: Things to keep in mind



Single individual pod start time may be longer on Fargate than on EC2

Each pod deployment sources a virtual node first from the Fargate fleet

Pod deployments at scale may be faster due to Fargate parallelism

E.g., think of the delay that Cluster Autoscaler can introduce in sourcing new EC2 capacity

Pricing

Standard EKS cluster pricing $0.20 per hour

Standard Fargate Pricing for vCPU and memory

Availability

Available today for all new 1.14 clusters

- Create a new cluster
- Update a 1.13 cluster to 1.14
- We'll automatically update existing 1.14 clusters in the coming weeks

Use EKS with Fargate in

- Virginia (us-east-1)
- Ohio (us-east-2)
- Dublin (eu-west-1)
- Tokyo (ap-northeast-1)

Thank you!

Nathan Taber
@nctaber

Massimo Re Ferrè
@mreferre

AWS re:Invent

aws

Please complete the session survey in the mobile app.