AWS
re:Invent

NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

CON406

A close look at AWS Fargate and AWS App Runner

Archana Srikanta (she/her)

Principal Software Engineer
AWS

Agenda

Compute services evolution

Amazon Elastic Compute Cloud (Amazon EC2)
released 2006
→
AWS Elastic Beanstalk
released 2011
→
Amazon Elastic Container Service (Amazon ECS)
released 2015
→
AWS Fargate
released 2017
→
AWS App Runner
released 2021

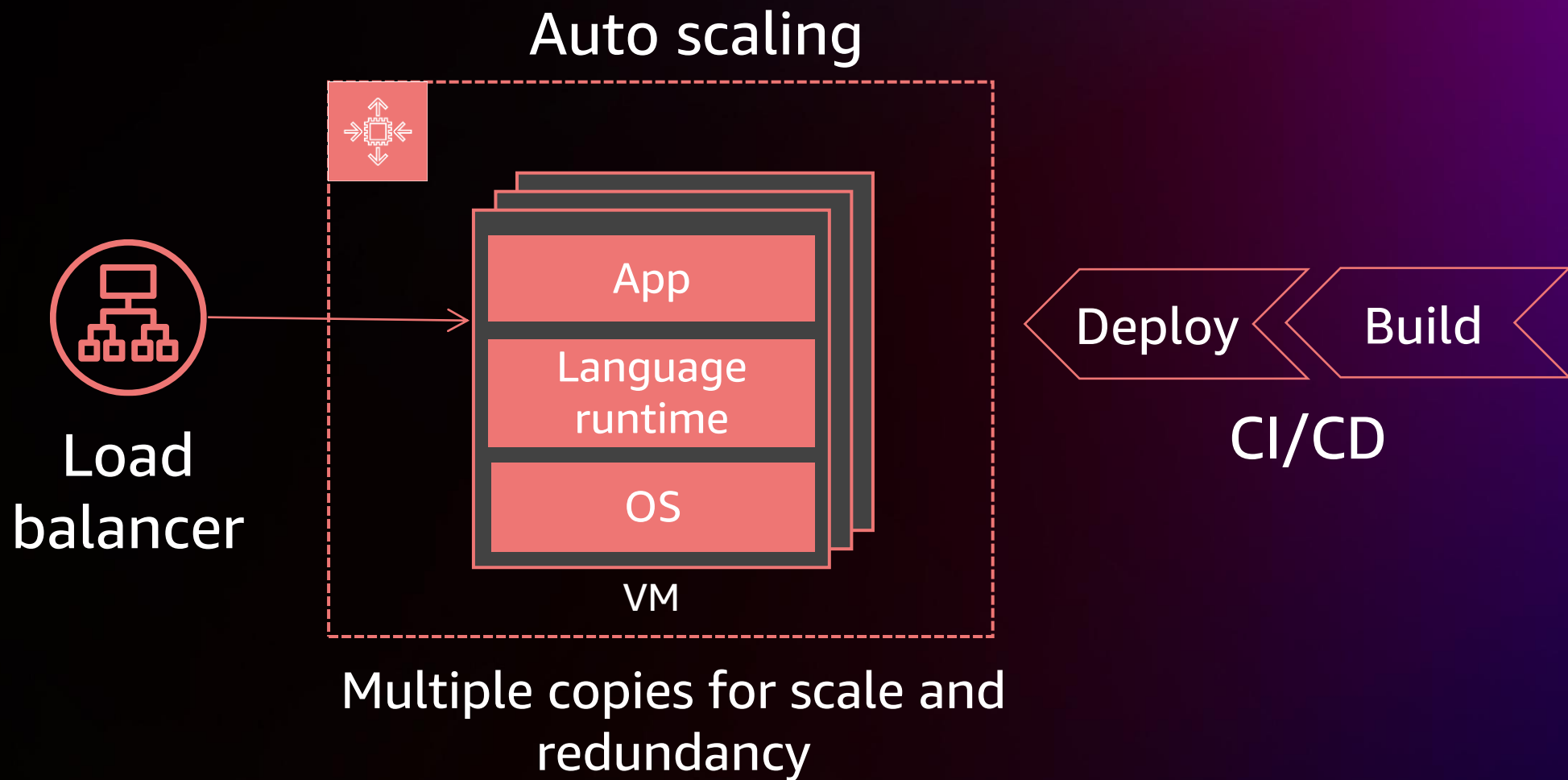## Under-the-hood architecture

Amazon ECS
←
AWS Fargate
←
AWS App Runner

Security considerations

Availability considerations

Compute services evolution

Typical web application architecture

Auto scaling

App

Language runtime

OS

VM

Load balancer
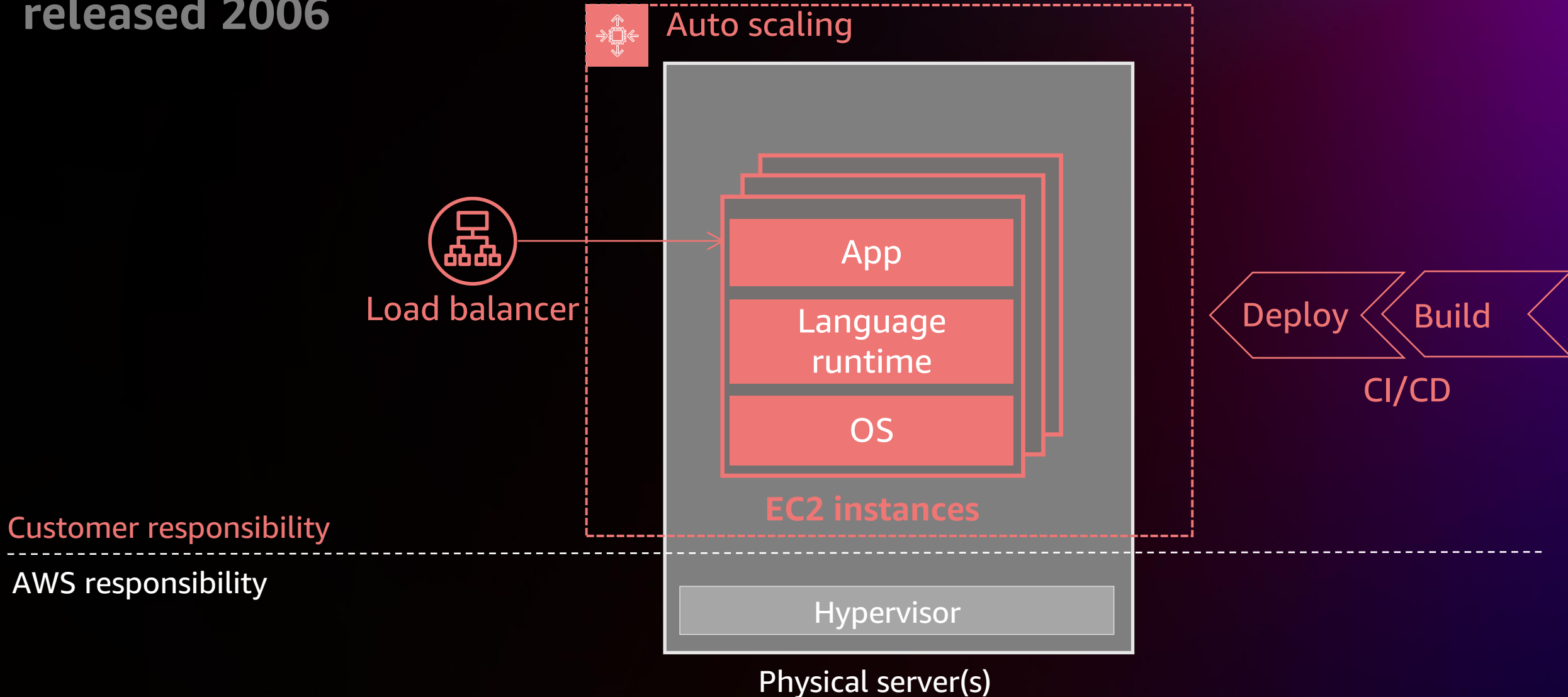
Deploy

Build

CI/CD

Multiple copies for scale and redundancy

Shared responsibility on AWS

Application security and availability is a shared responsibility between AWS and the customer

AWS Elastic Beanstalk
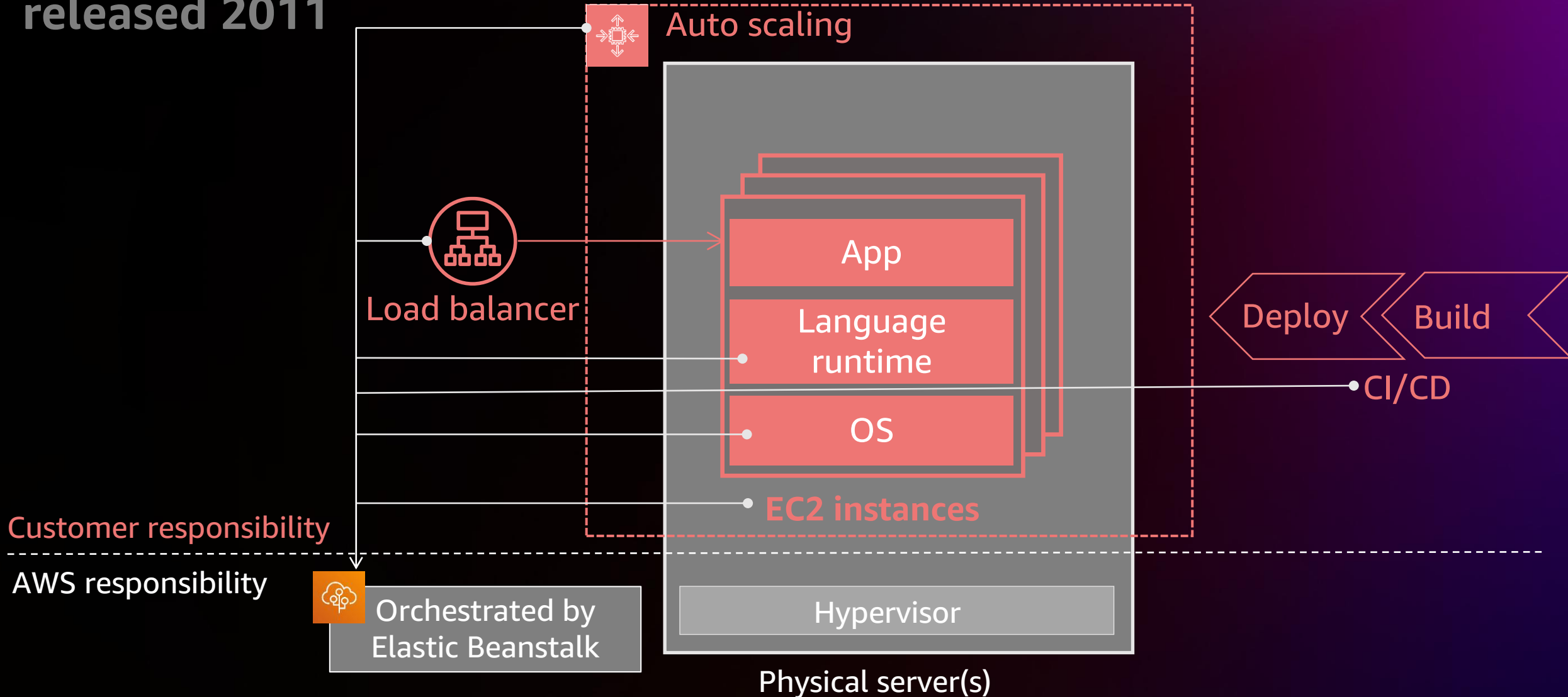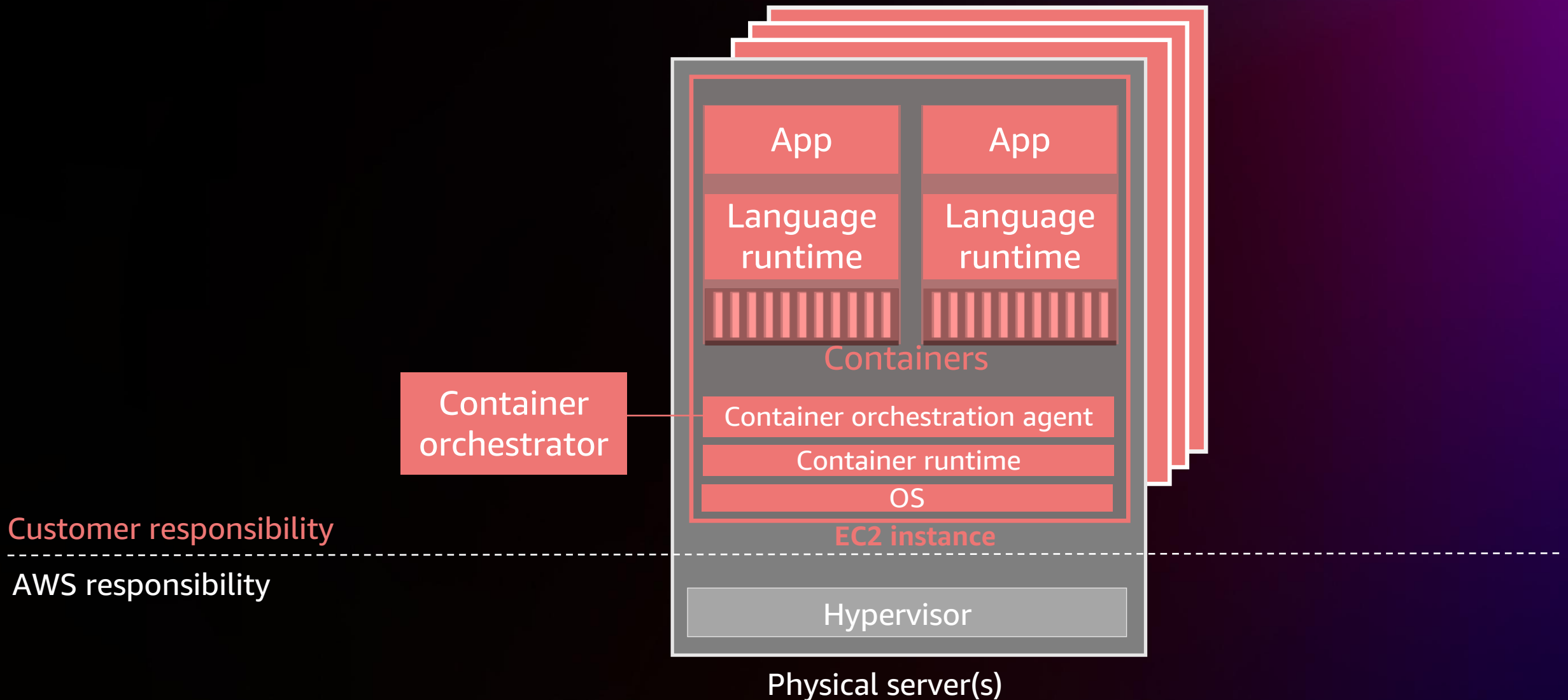**released 2011**

Auto scaling

Load balancer
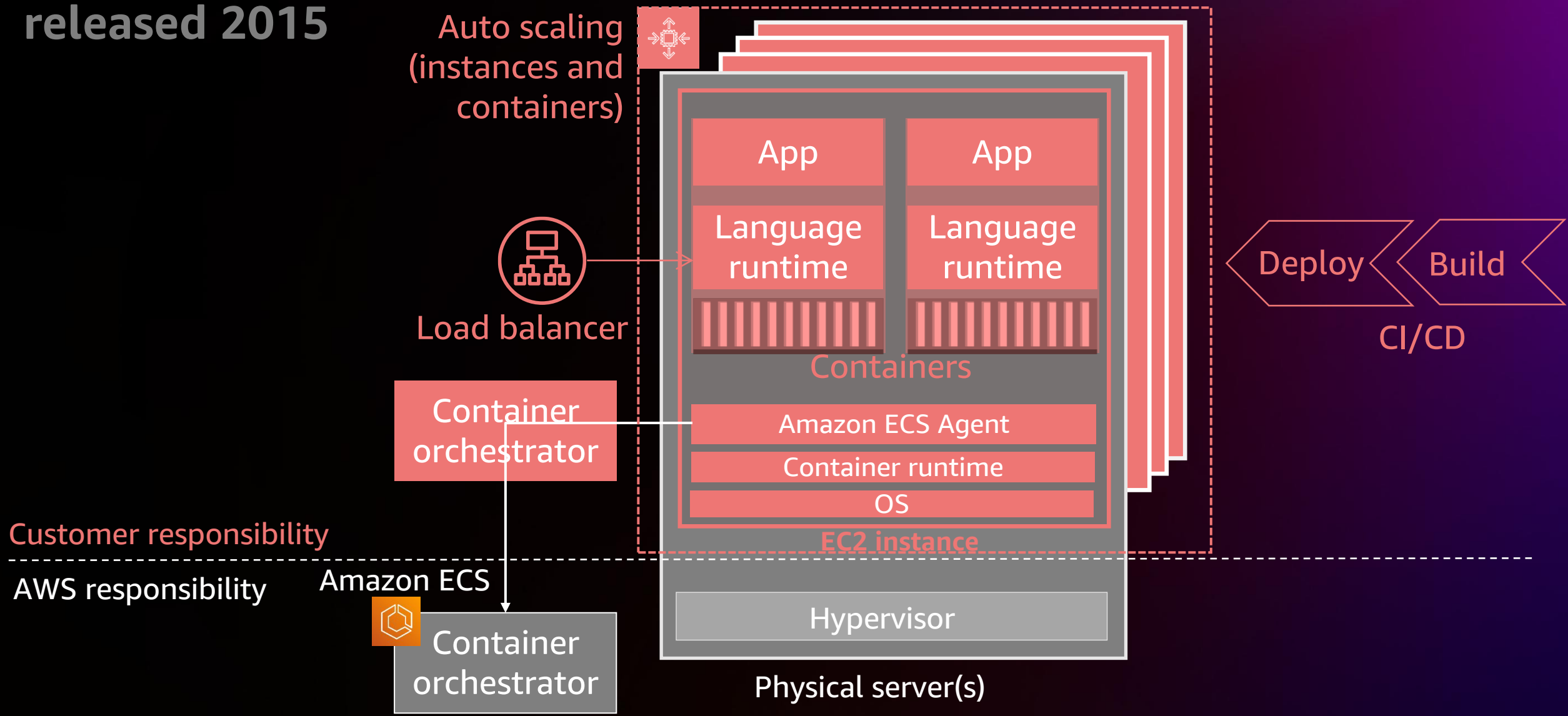
App

Language runtime

Deploy   Build

CI/CD

OS

EC2 instances

Customer responsibility

AWS responsibility

Orchestrated by Elastic Beanstalk

Hypervisor

Physical server(s)

Container orchestrators

App

App

Language runtime

Language runtime

Container orchestrator

Containers

Container orchestration agent

Container runtime

OS

Customer responsibility

EC2 instance

AWS responsibility

Hypervisor

Physical server(s)

Amazon Elastic Container Service (ECS)

released 2015

Auto scaling (instances and containers)

Load balancer

Container orchestrator

Customer responsibility

AWS responsibility

Amazon ECS

Container orchestrator

App

App

Language runtime

Language runtime

Containers

Amazon ECS Agent

Container runtime

OS

EC2 instance

Hypervisor

Physical server(s)

Deploy

Build

CI/CD

AWS Fargate
released 2017

Auto scaling containers

Load balancer

App

App

Language runtime

Language runtime

Containers

Deploy

Build

CI/CD

Fargate Agent

Container runtime

OS

EC2 instance

Customer responsibility

AWS responsibility

Amazon ECS

Container orchestrator

Hypervisor

Physical server(s)

AWS App Runner
released 2021

Auto scaling containers

Load balancer

Customer responsibility

AWS responsibility

Amazon ECS

Container orchestrator

App

App

Language runtime

Language runtime

Containers

Fargate Agent

Container runtime

OS

EC2 instance

Hypervisor

Physical server(s)

Deploy

Build

CI/CD

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

App Runner experience

Development teams

Source code
GitHub

Create
`Connection`
Authorize AWS
to access
source code

Container Image
Amazon ECR

Create
`IAM Role`
Permissions to
pull image

Create
`Service`

Clients

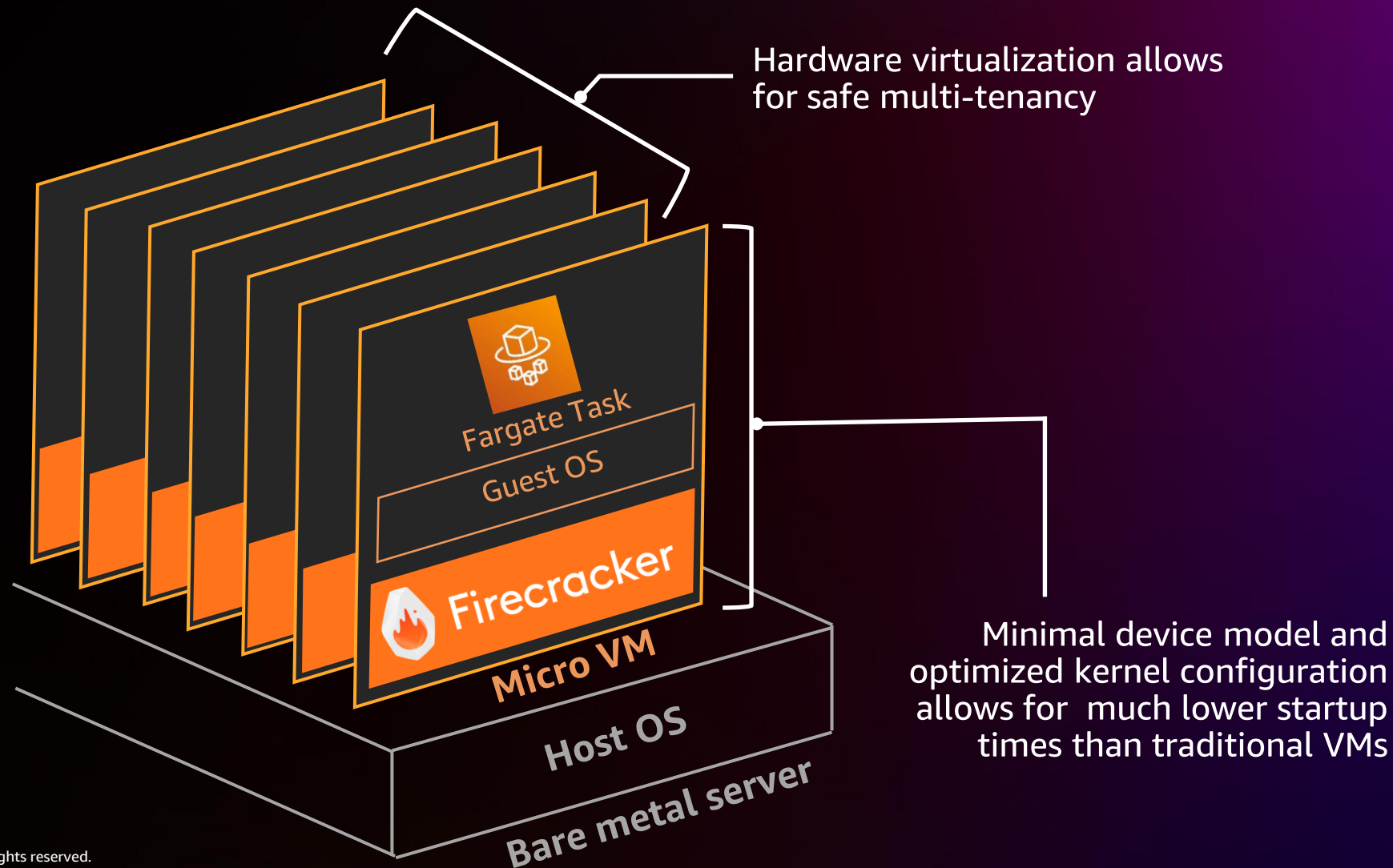HTTPS requests

secure
App Runner
service URL

Under the hood

App Runner: Under the hood
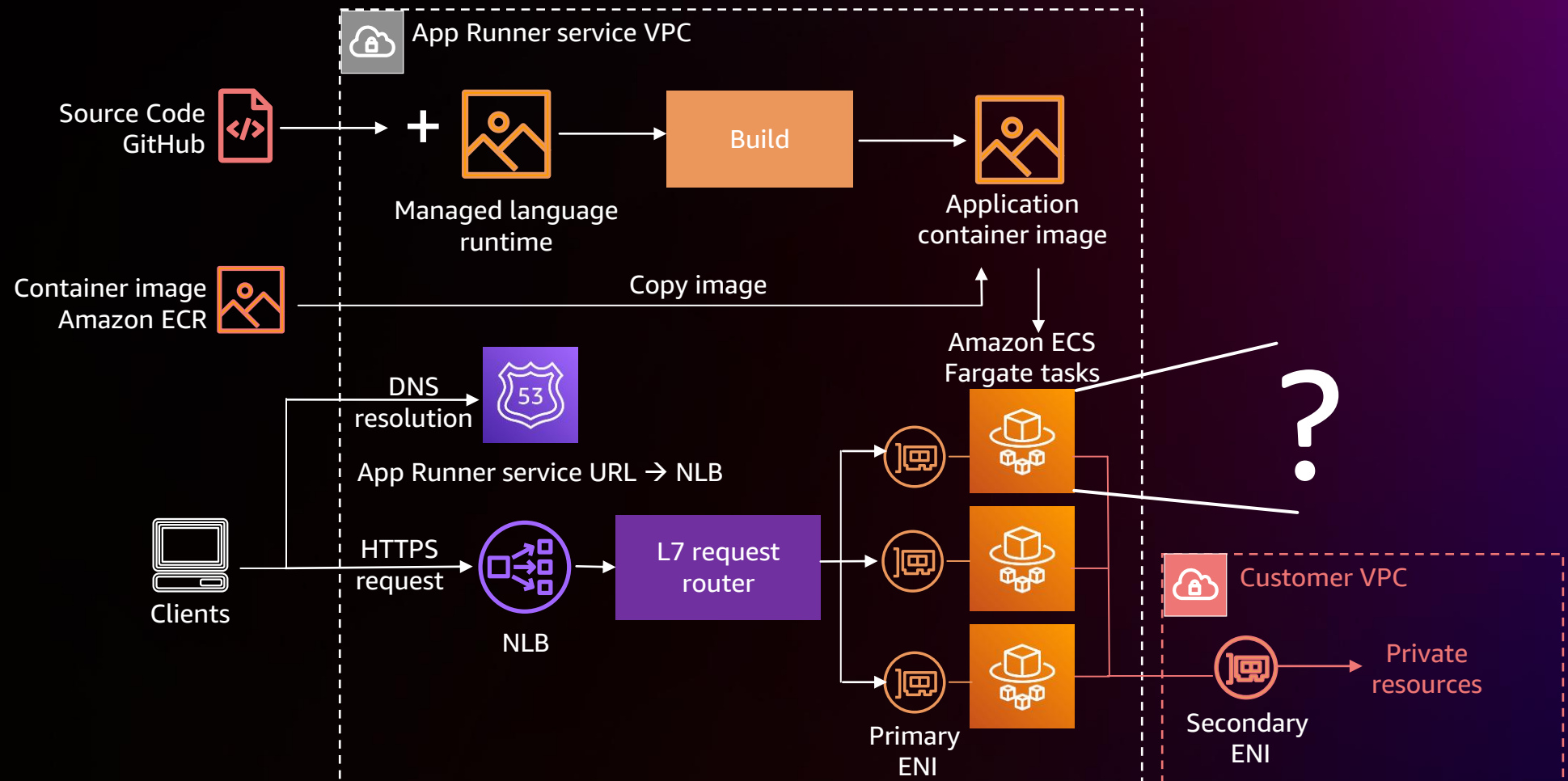
Firecracker

Open source virtualization technology that is purpose built for container workloads

Hardware virtualization allows for safe multi-tenancy

Fargate Task

Guest OS

Firecracker

Micro VM

Host OS

Bare metal server

Minimal device model and optimized kernel configuration allows for much lower startup times than traditional VMs

Fargate: Under the hood
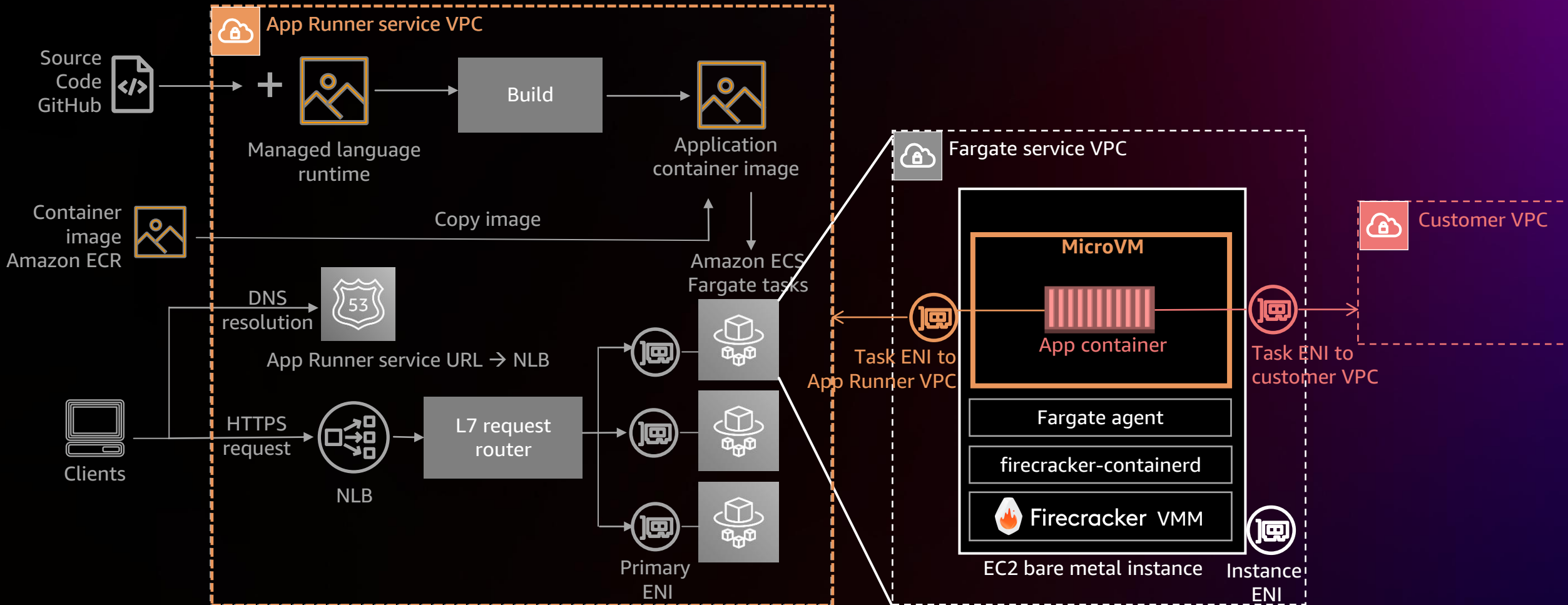
Amazon ECS Orchestration: Under the hood
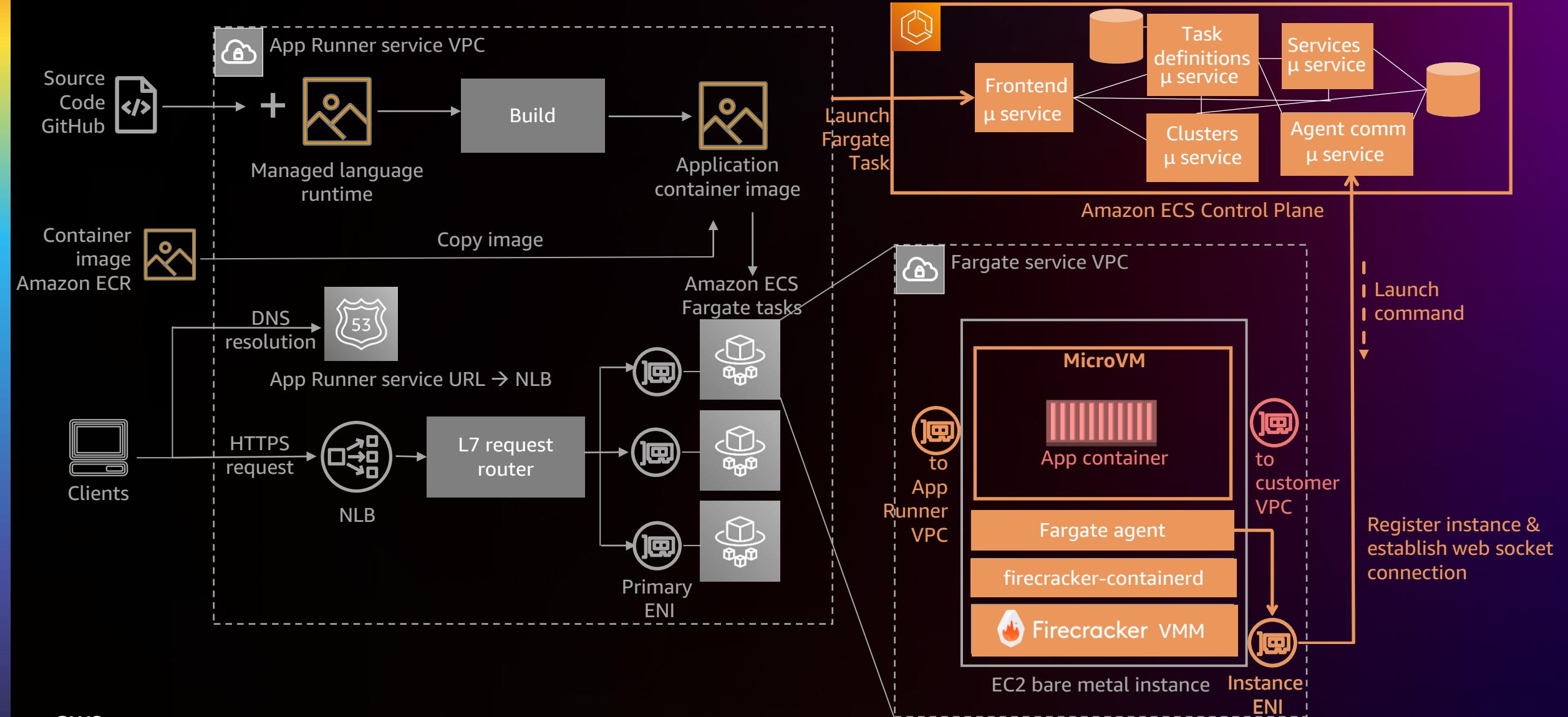
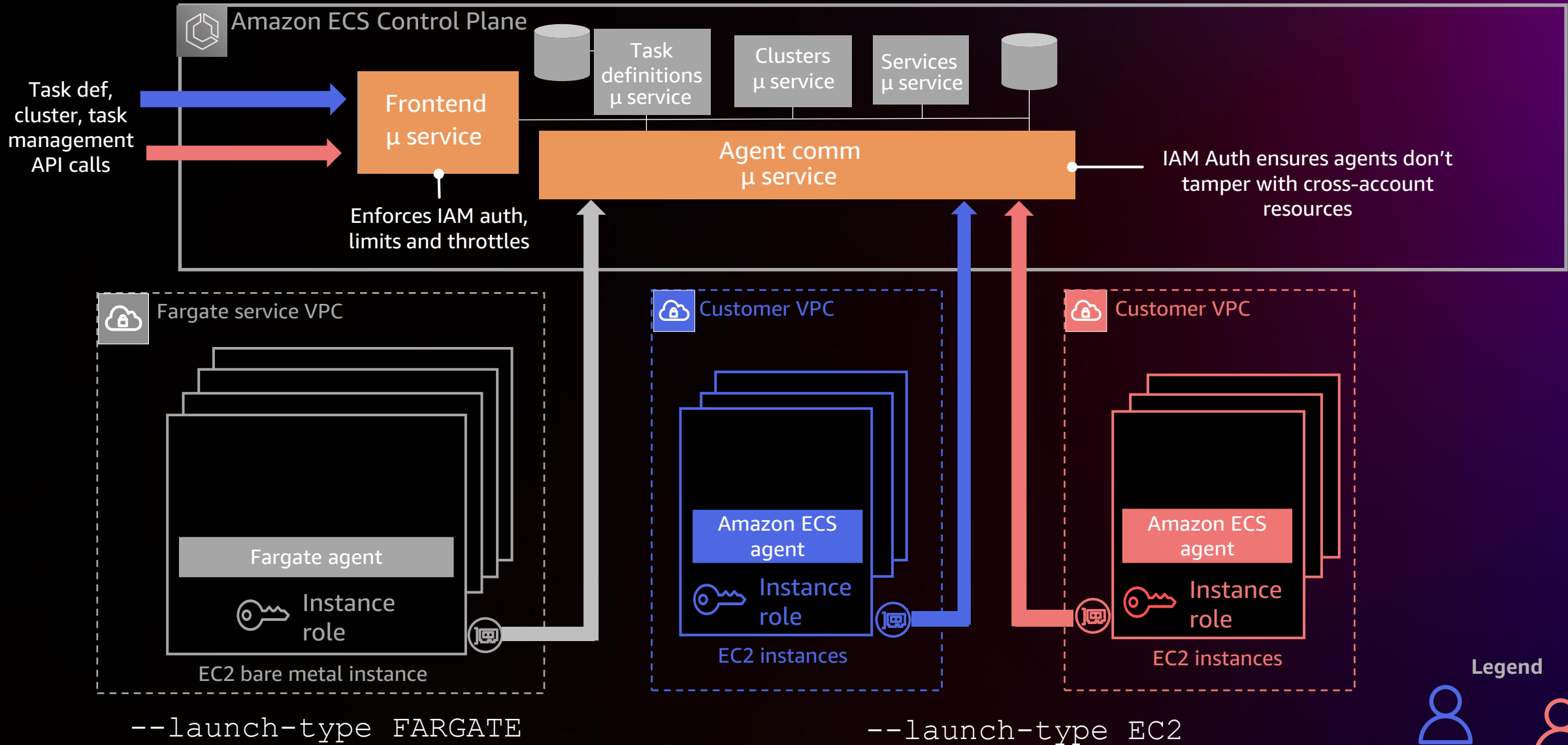# Security considerations

App Runner data plane security

App Runner service VPC

NEW!!
Private service endpoints

Customer VPC

Clients

Customer's responsibility

AWS Private Link endpoint

No task-to-task communication in the App Runner VPC

Not even among tasks of the same App Runner service

NLB

L7 request router

Primary ENI

Primary ENI

Fargate Tasks

Fargate Tasks

Customer VPC

Customer's responsibility

Secondary ENI

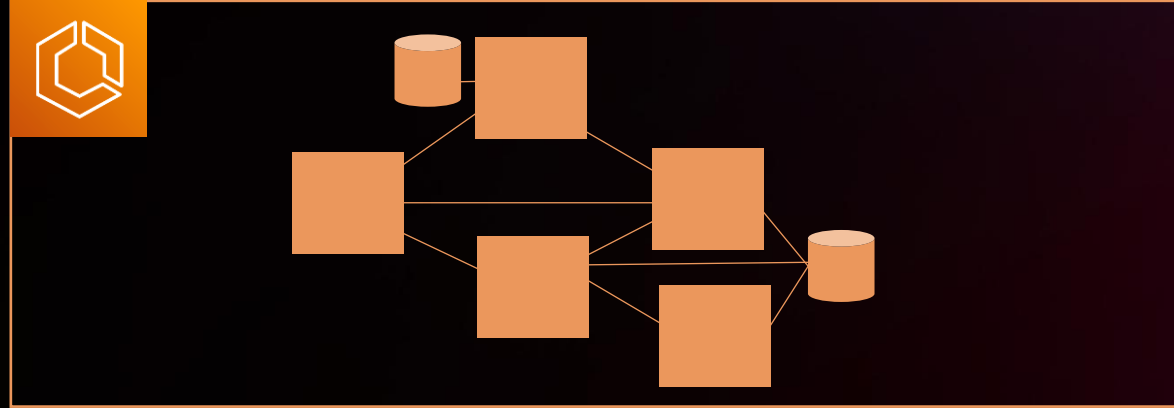Dependencies

Legend

Tenant 1   Tenant 2
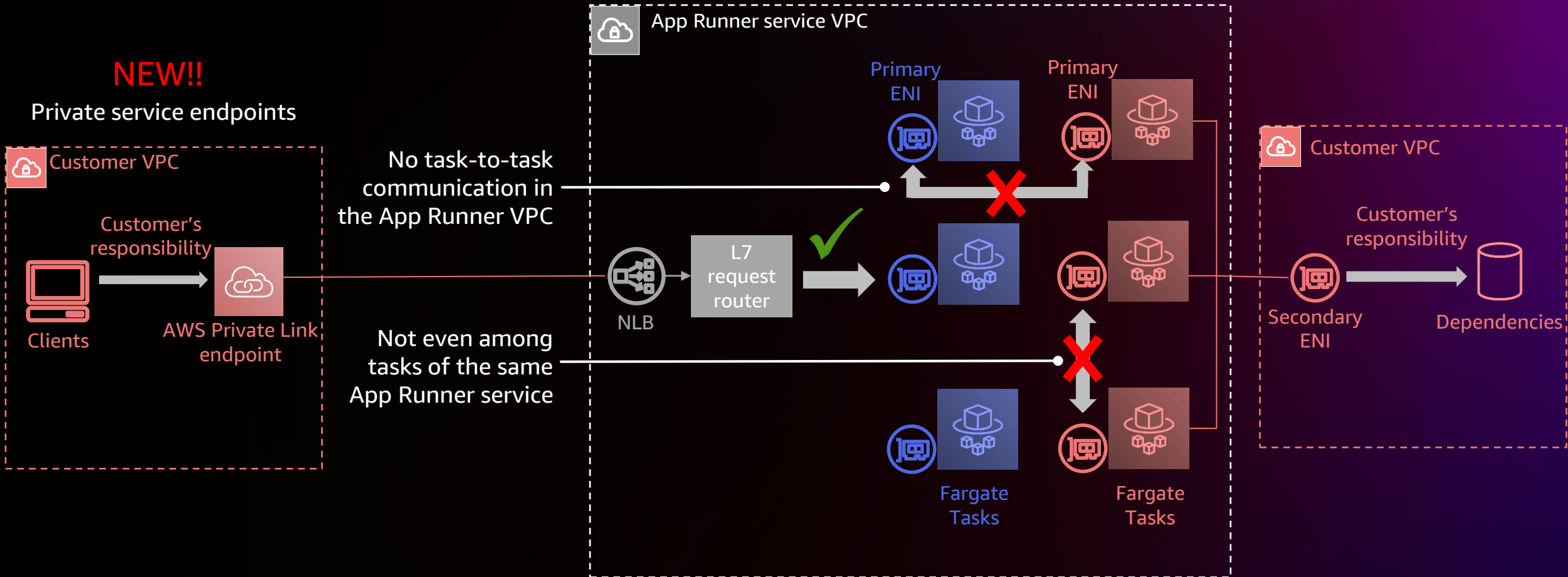
Fargate data plane security
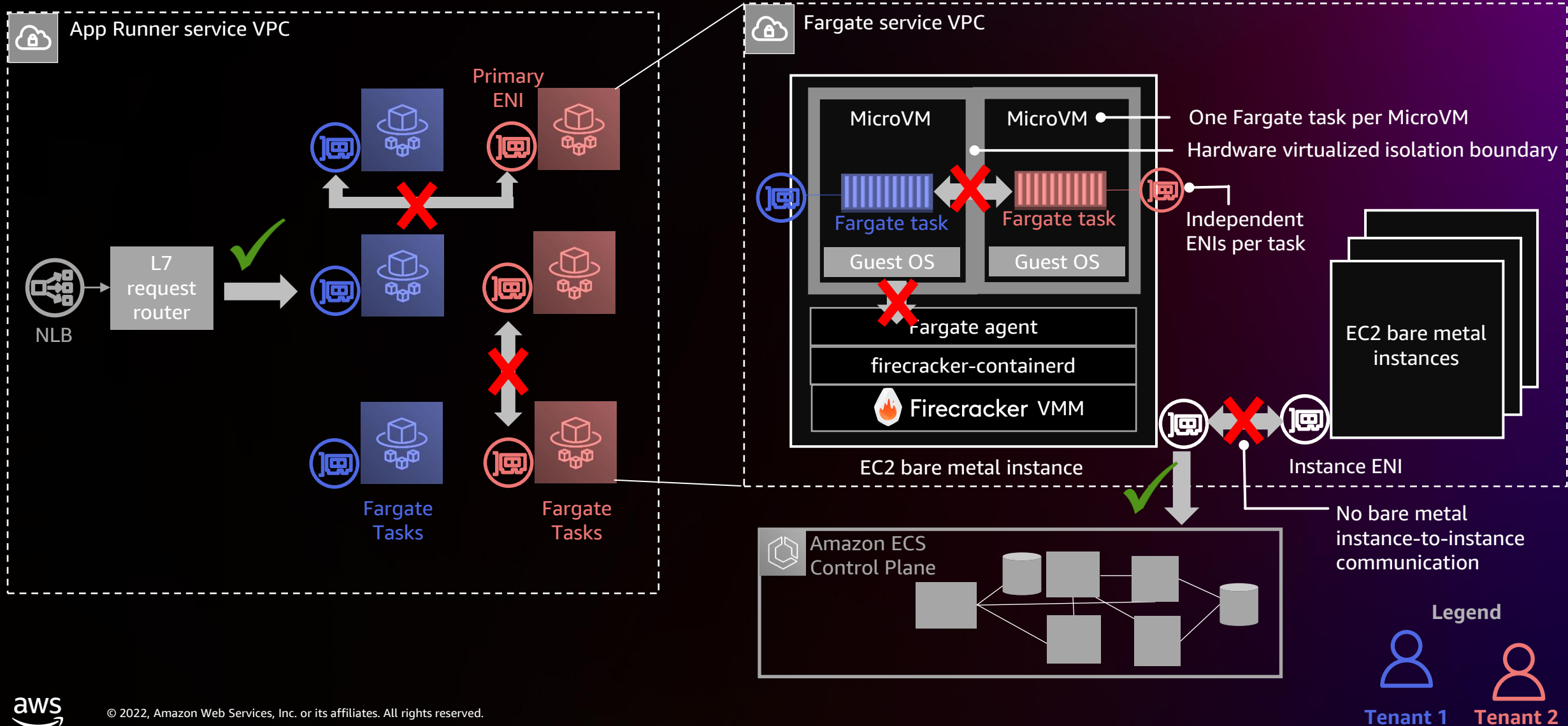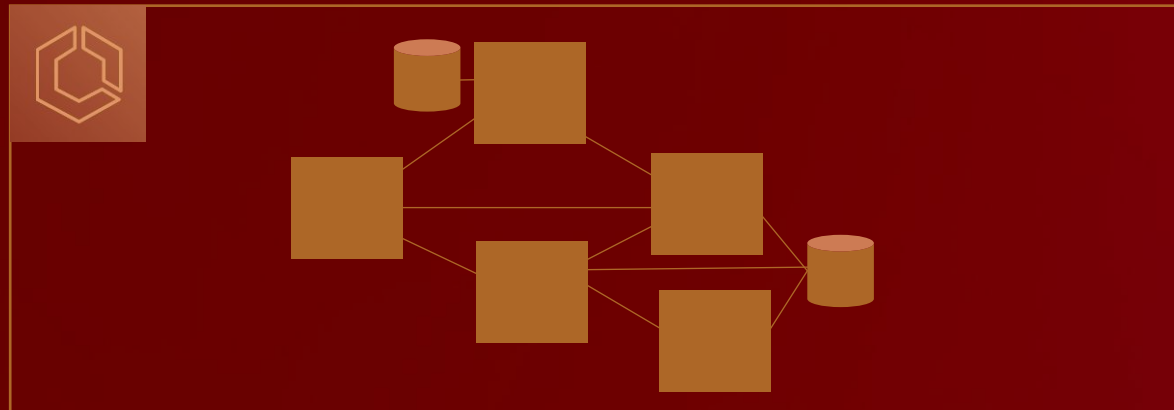
Availability considerations
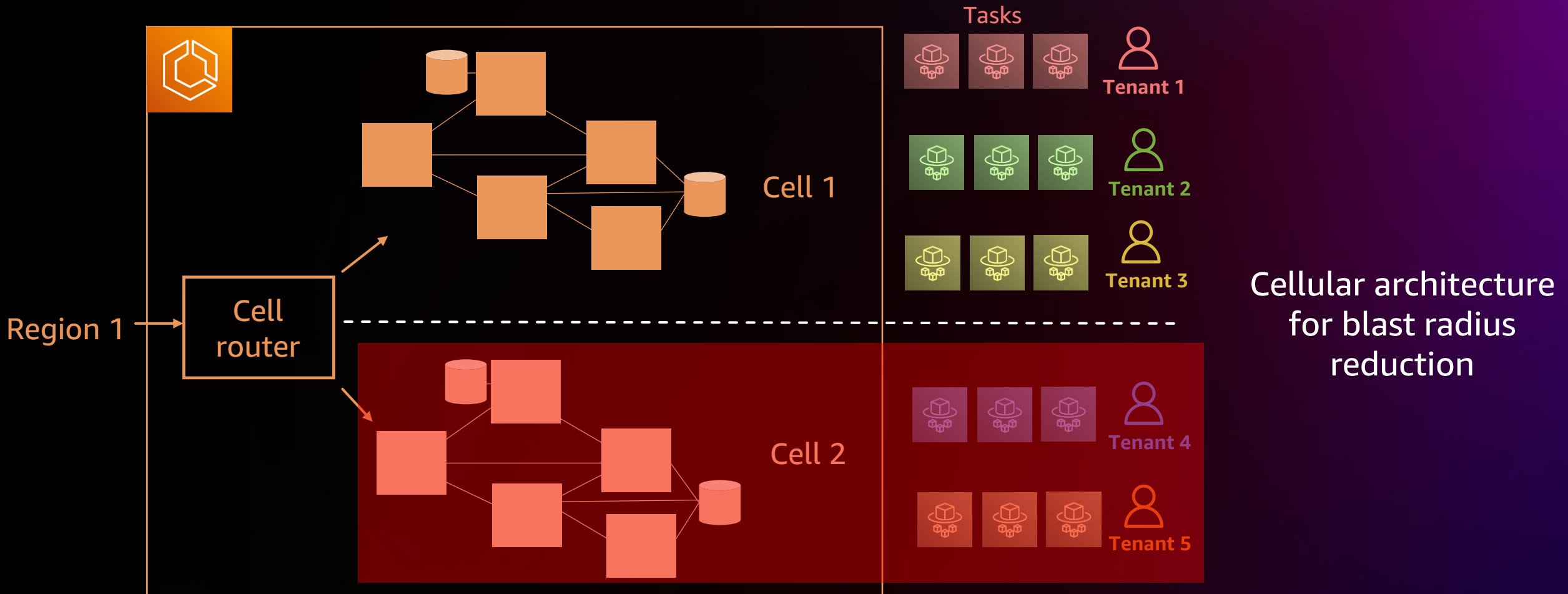
Amazon ECS control plane availability

Region 1

Region 2

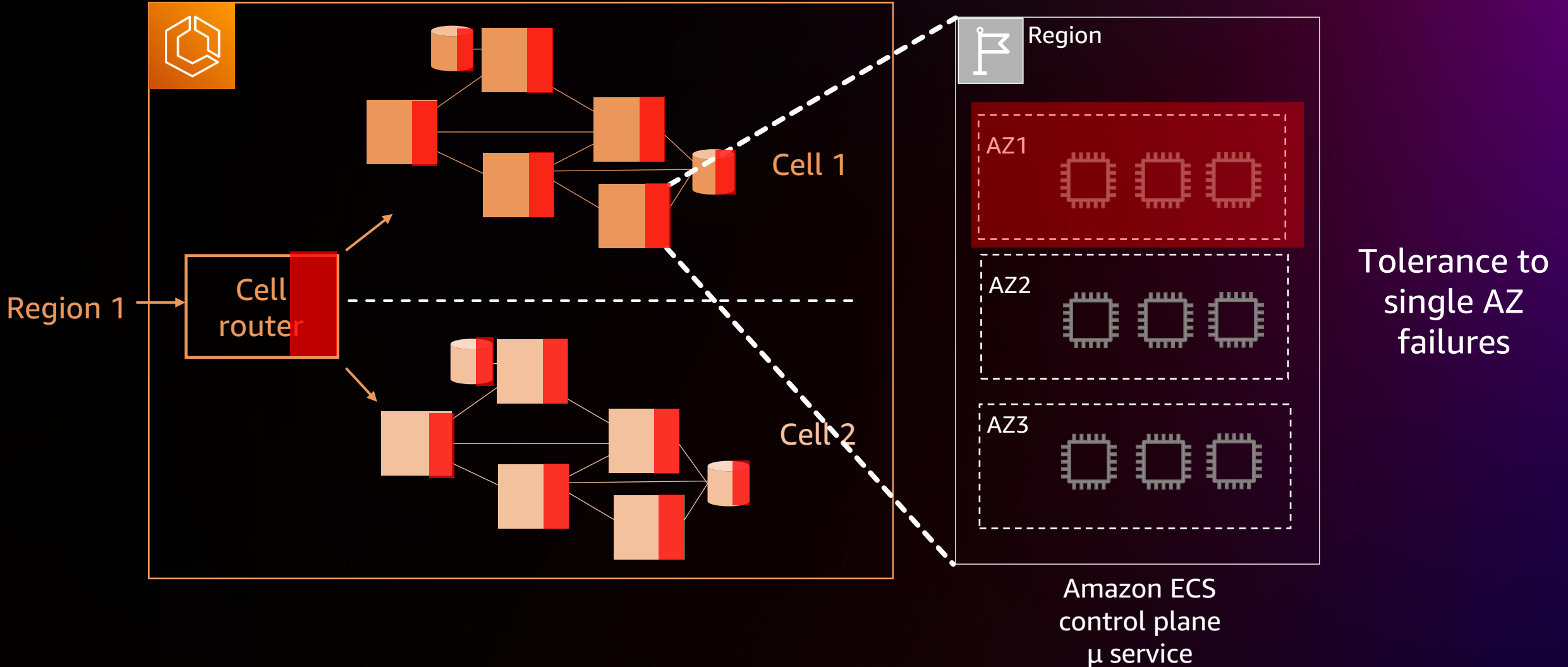Complete independence between regions

Amazon ECS control plane availability

Tasks

Tenant 1
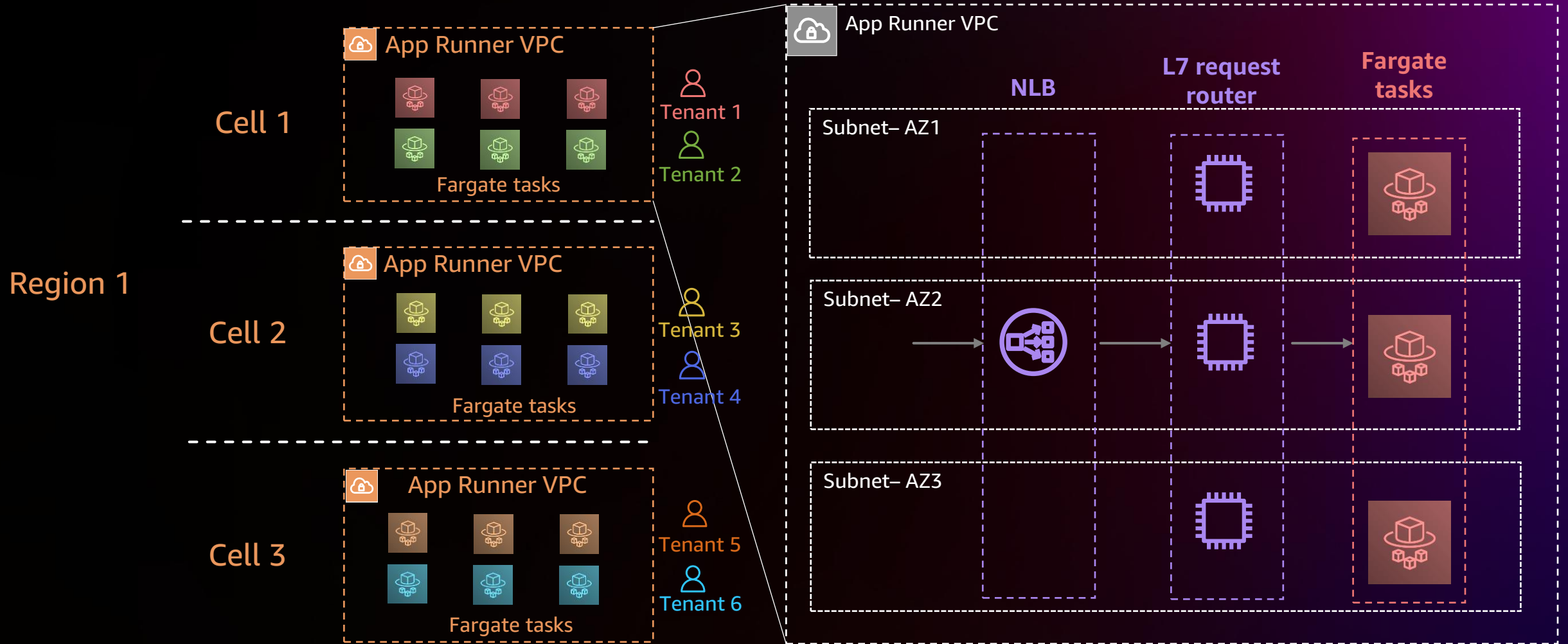
Tenant 2

Tenant 3

Cell 1

Region 1 → Cell router

Cell 2

Tenant 4

Tenant 5

Cellular architecture for blast radius reduction

Amazon ECS control plane availability

Region 1 → Cell router

Cell 1

Cell 2

Region

AZ1

AZ2

AZ3

Tolerance to single AZ failures

Amazon ECS control plane µ service

Fargate data plane availability

AZ1　　　　　　　　　AZ2　　　　　　　　　AZ3

Fargate VPC — Cell 2
Bare metal instances

Scale out ↑

Region 1

Fargate VPC
Bare metal instances

Fargate VPC
Bare metal instances

Fargate VPC — Cell 1
Bare metal instances

← Fixed-size cells →

Region 2

Fargate VPC

Fargate VPC

Fargate VPC
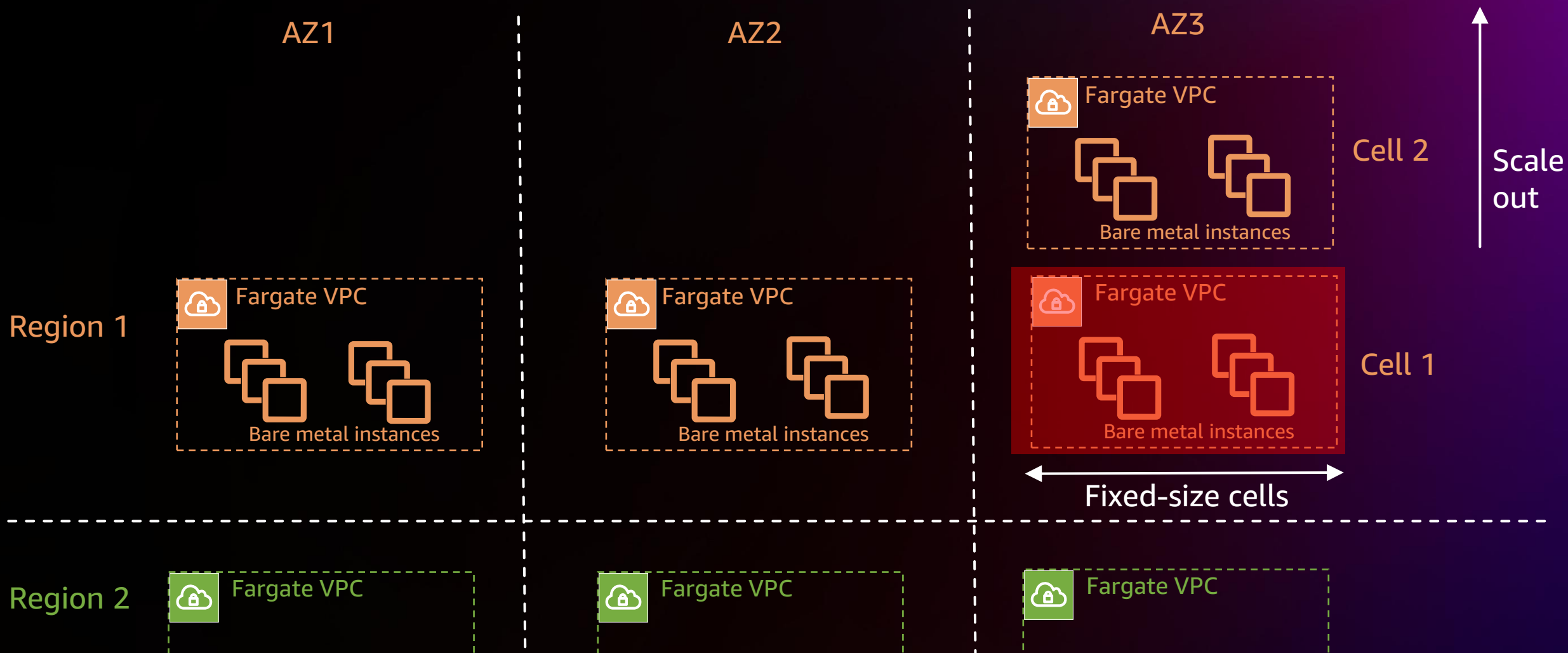
App Runner data plane availability

Takeaways

Takeaways

- Rich portfolio of services for hosting containerized applications on AWS

- Each one offers a different level of abstraction

- They all have incredible thought put into their security and availability stance

- Start your experiments with the highest abstraction service you can use, only moving down if you need to

Thank you!

Archana Srikanta

@ArchanaSrikanta (Twitter)

Please complete the session
survey in the **mobile app**