
AWS
re:Invent
NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

CON312

Auto-scale your web application using AWS App Runner

Jason Woodlee (he/him)

Senior Software Development Manager
Amazon

Ryan Coleman (he/him)

Principal Product Manager
Amazon

Web Application Properties

1 Web Applications & API Servers

Applications that serve HTTP-based requests

2 Multi-Concurrent

The application is a long-running server that serves many concurrent requests during its lifetime

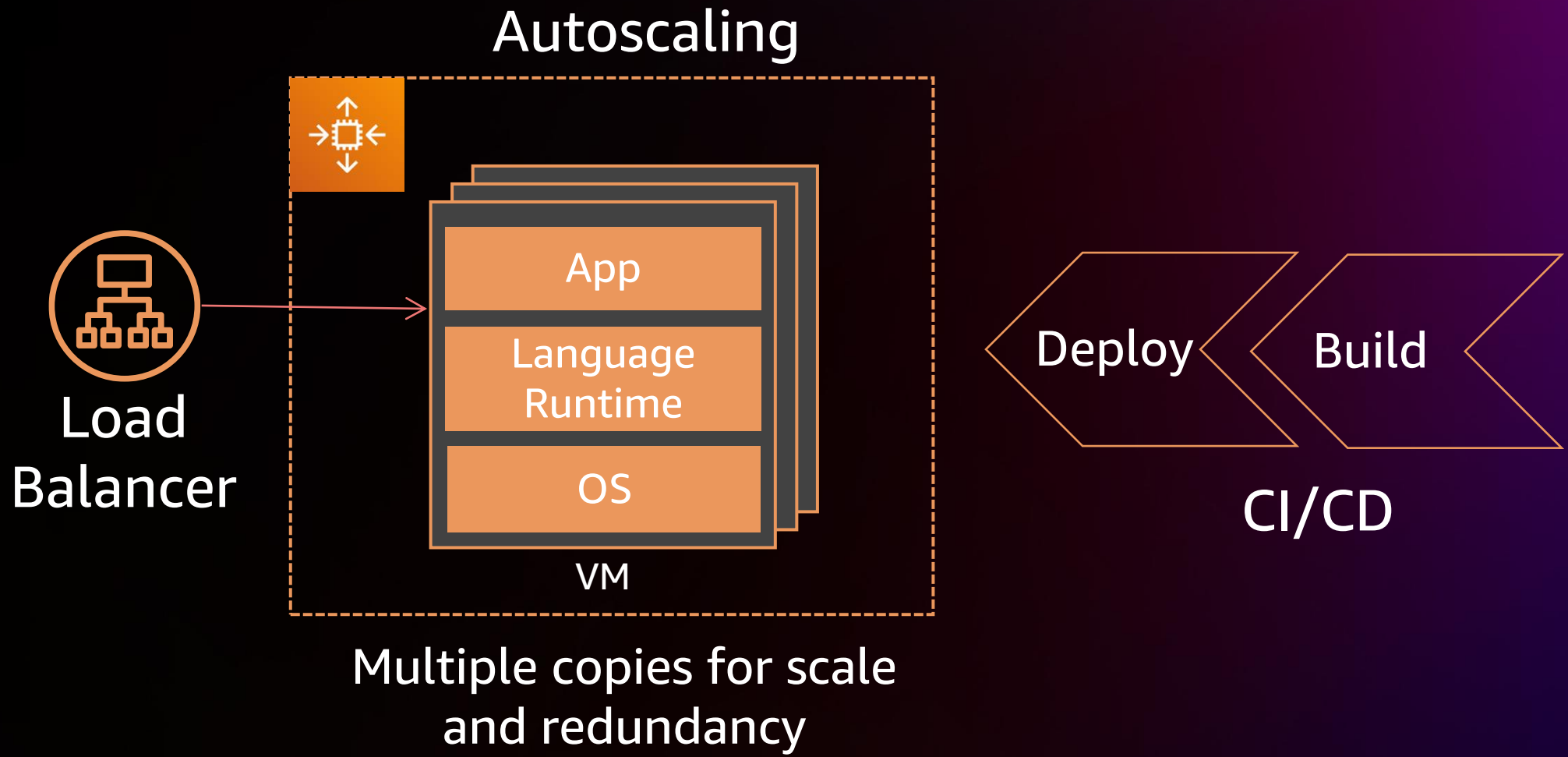
3 Stateless

Requests are processed independent of one another and do not depend on local state. State maybe stored external to the application instance (e.g.: Amazon RDS, Amazon DynamoDB, etc.)

4 Minimal Background Processing

Any processing outside the context of a request must be limited. No background worker threads or cron jobs.

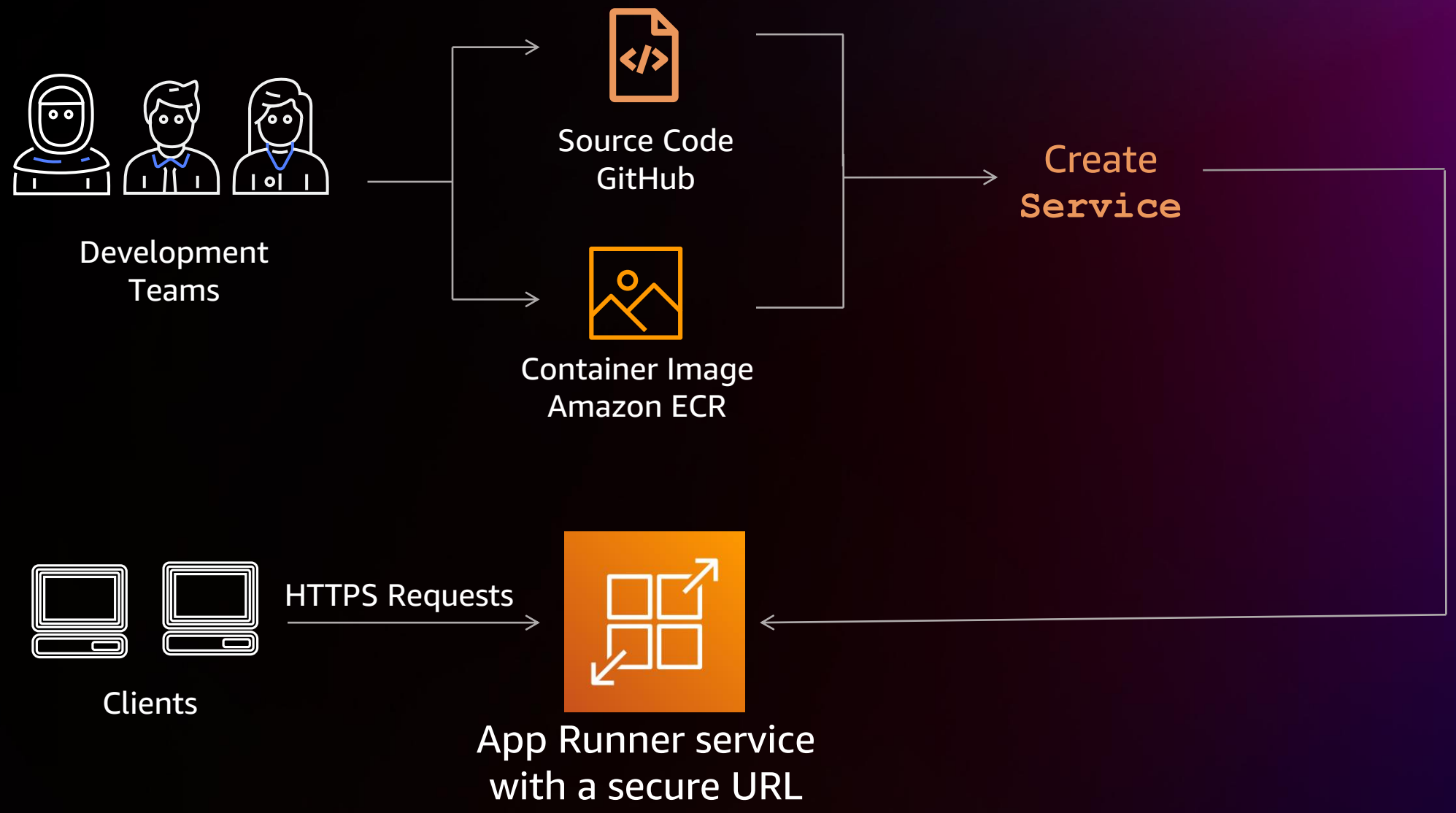
Typical Architecture



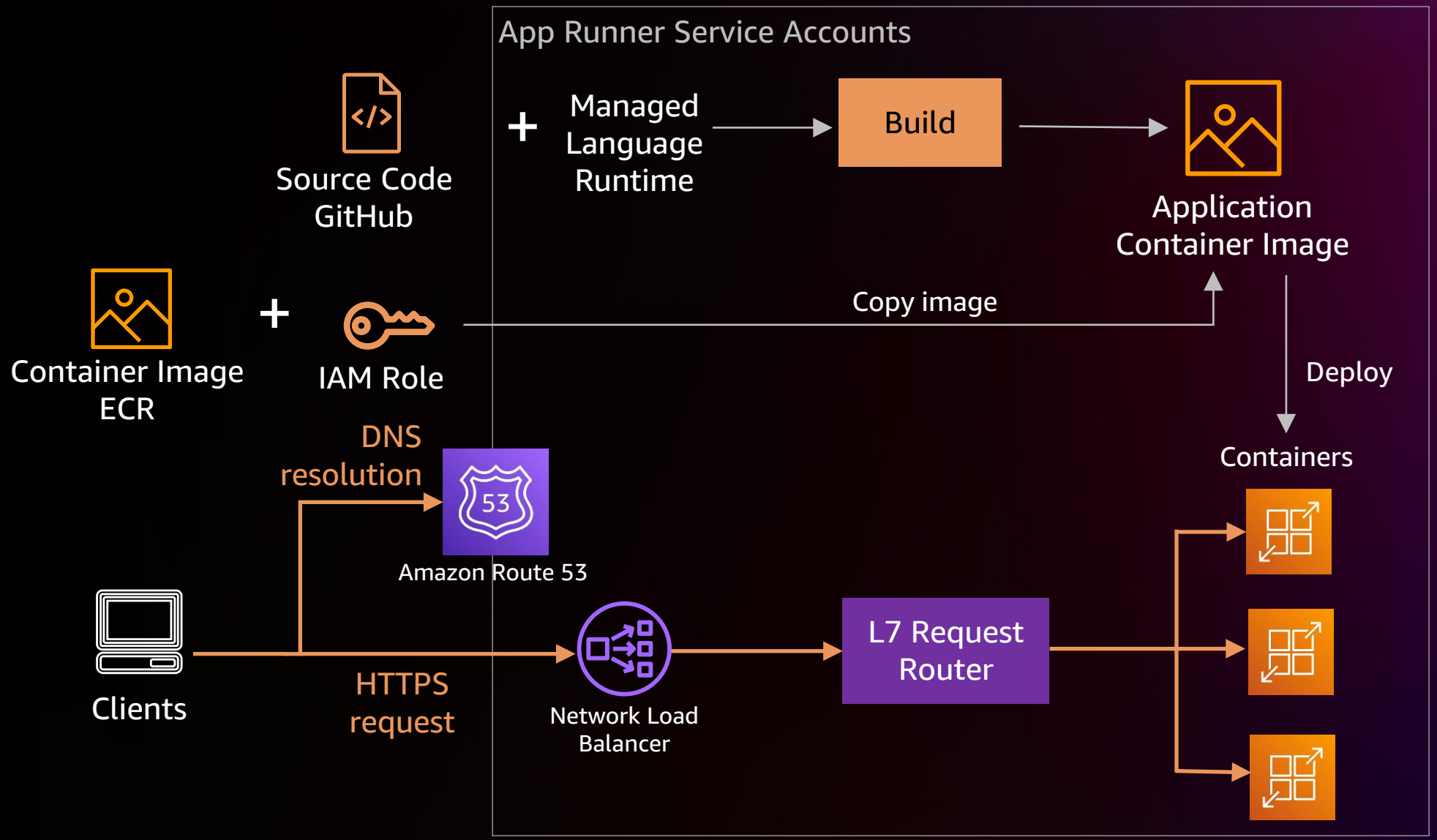
Sample Python Flask App on GitHub

AWS App Runner Experience

Experience



App Runner Under the Hood



Application Instance : Unit of Scaling

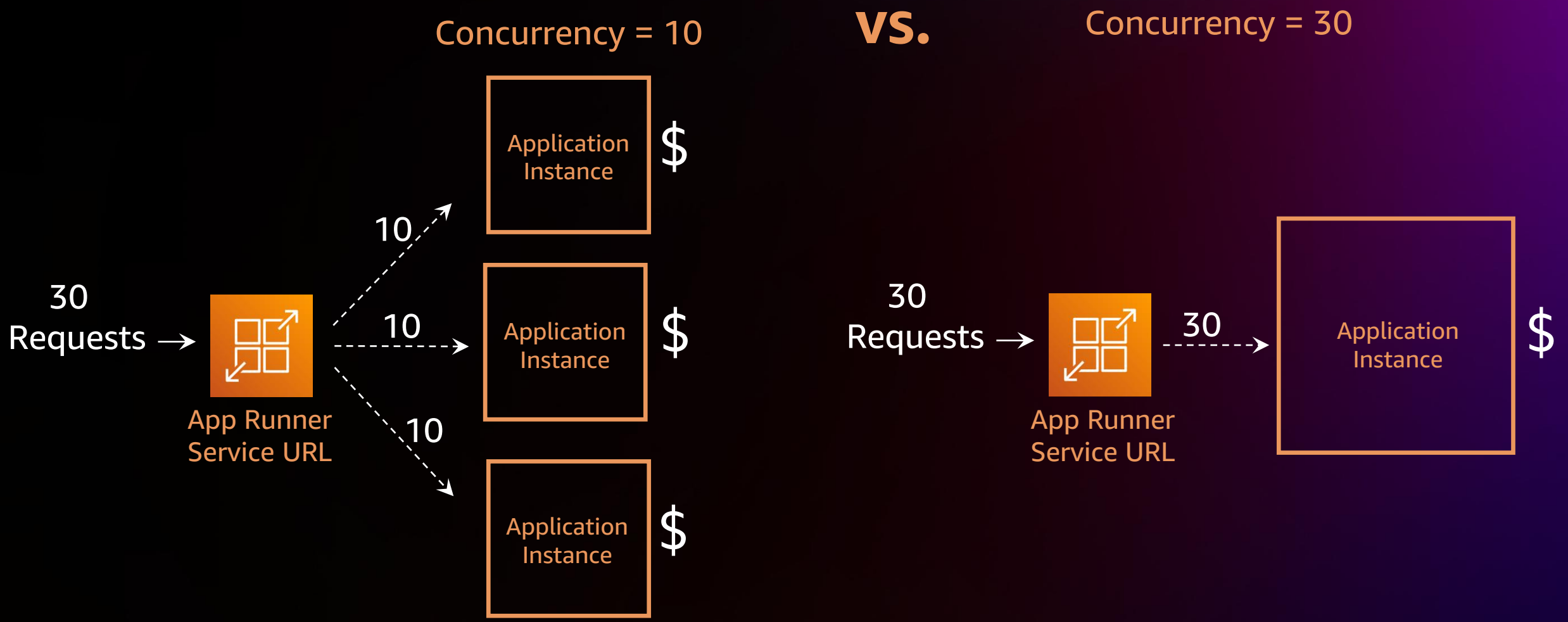
A running copy of your application in App Runner

Max concurrency

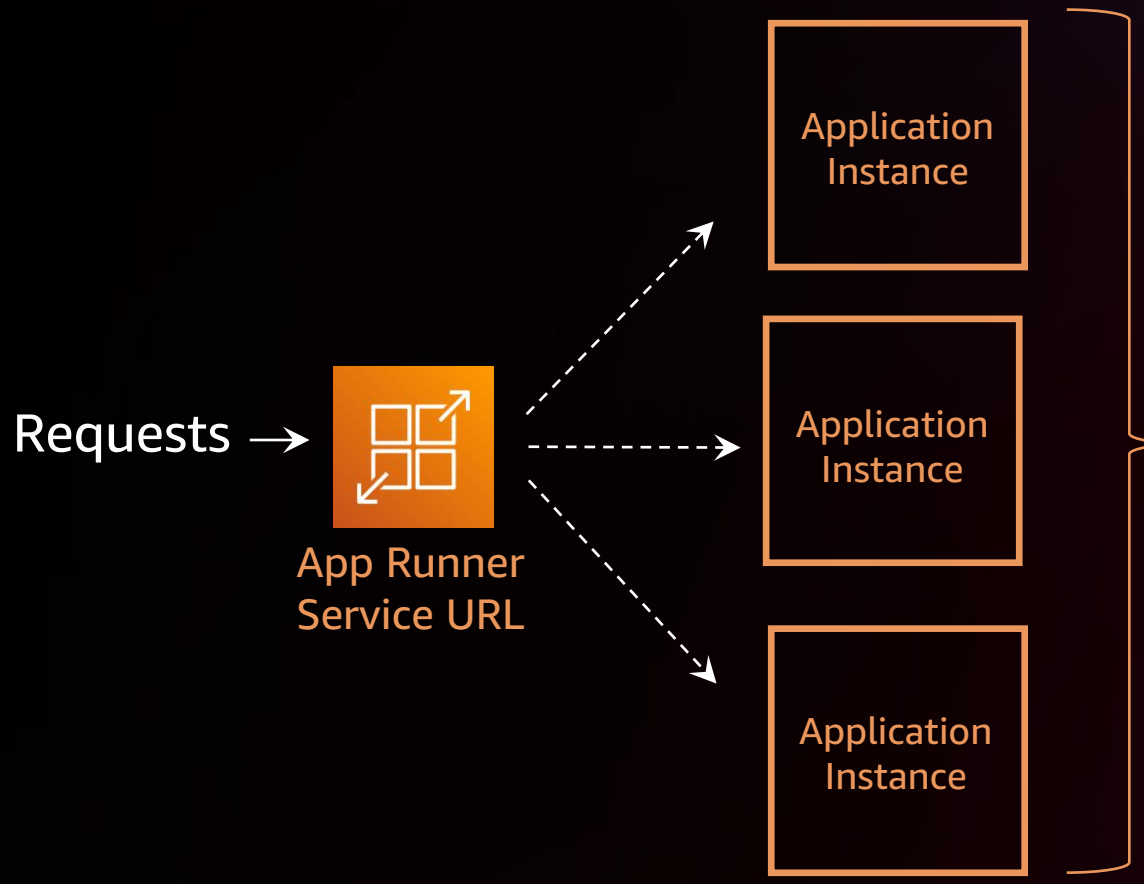
The maximum number of simultaneous requests a single application instance can handle

CPU	Memory
1 vCPU	2GB, 3GB, 4GB
2 vCPU	4GB

Concurrency: Dimension of Scaling



Active vs. Idle Application Instances



Active Application Instances

An instance is `Active` if it is processing requests

Idle Application Instances

An instance is `Idle` if it is running but not processing any requests –
It maybe CPU-throttled in this state

Autoscaling on AWS App Runner

Autoscaling Controls

`Max Concurrency`

The maximum number of simultaneous requests a single application instance can handle

`Minimum Provisioned Instances`

Minimum provisioned instances to avoid cold start latencies. Minimum is 1.

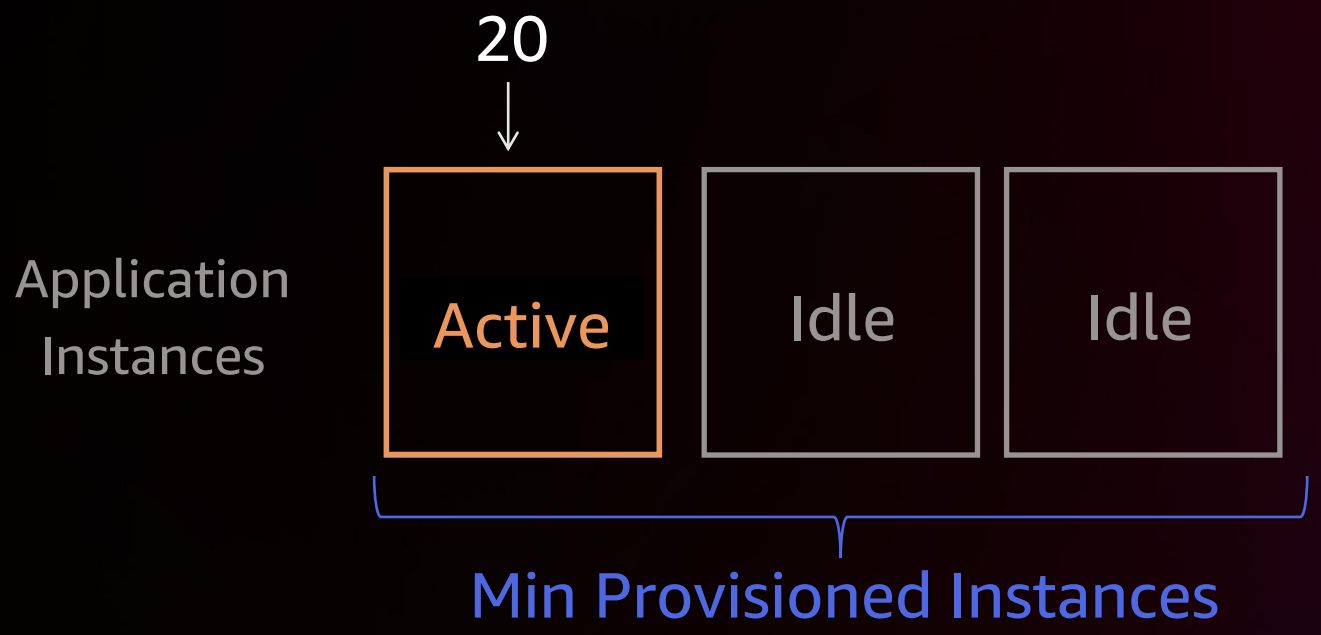
`Maximum Instances`

Upper bound on the number of instances launched to control cost

Autoscaling Example

`concurrency = 30; min=3; max=5`

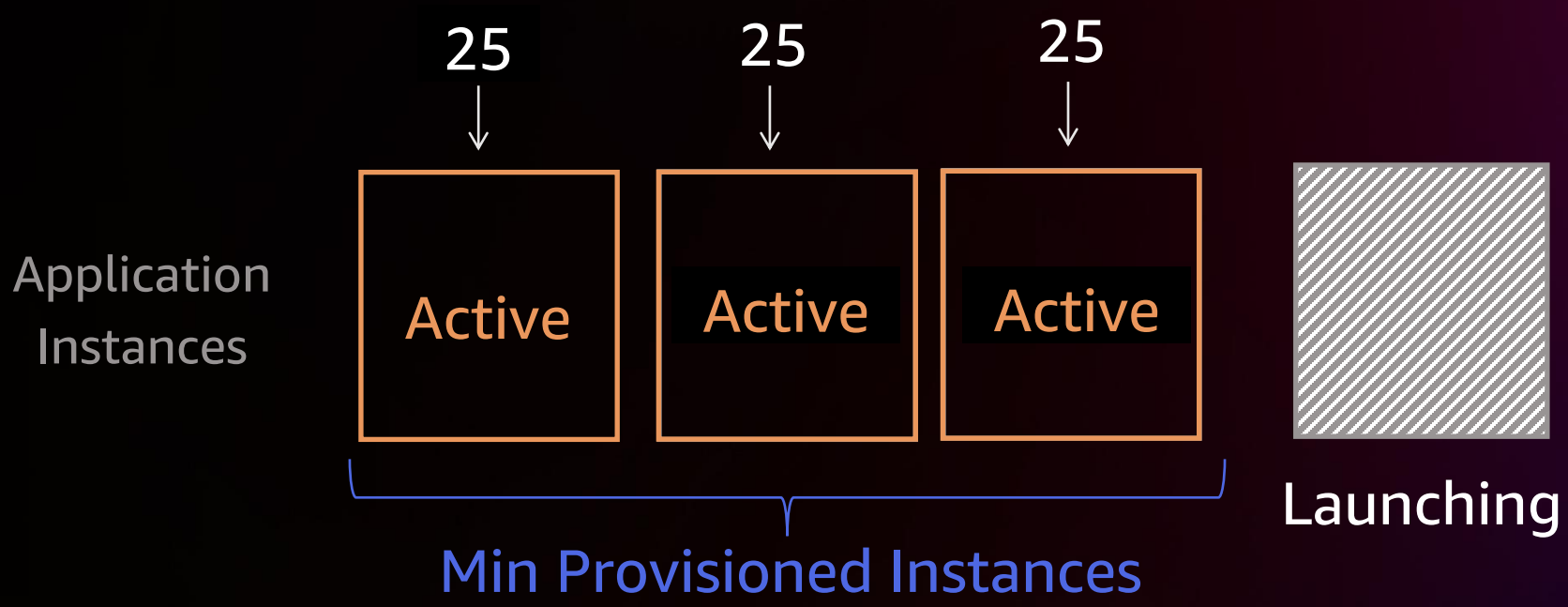
Load: 20 concurrent requests



Autoscaling Example

`concurrency = 30; min=3; max=5`

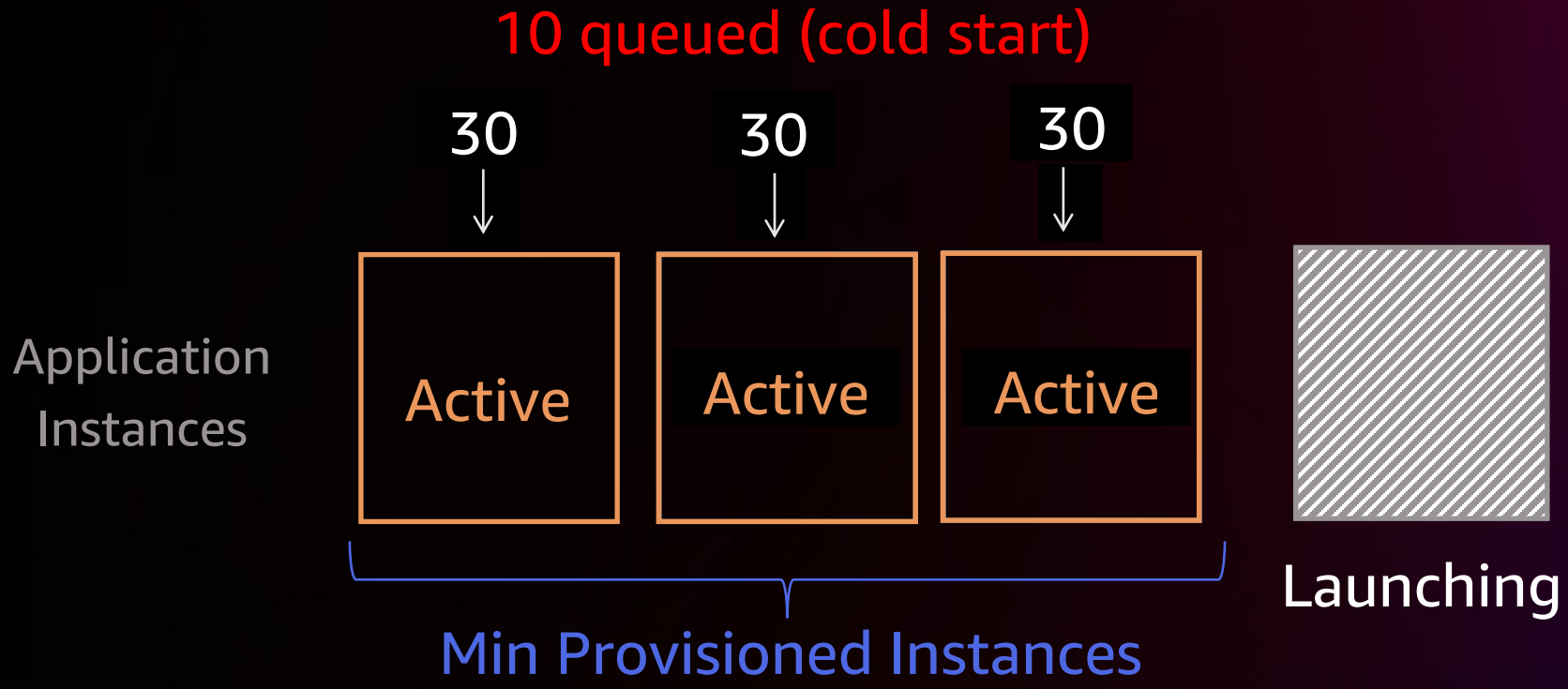
Load: 75 concurrent requests



Autoscaling Example

`concurrency = 30; min=3; max=5`

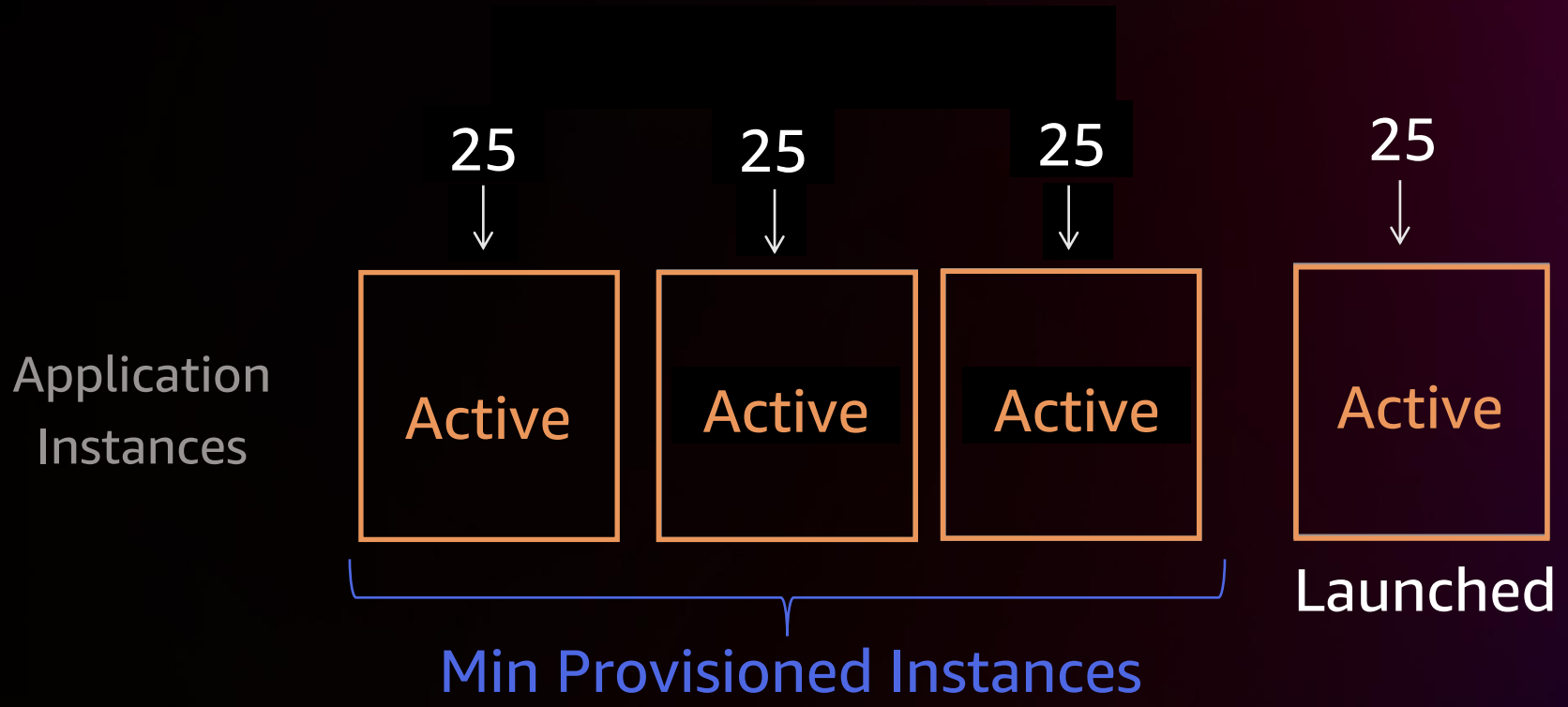
Load: 100 concurrent requests



Autoscaling Example

`concurrency = 30; min=3; max=5`

Load: 100 concurrent requests



Autoscaling Example

`concurrency = 30; min=3; max=5`

Load: 200 concurrent requests

App Runner Pricing

Pricing Dimensions

CPU & Memory “Pay-per-use” Pricing

Pay for CPU and memory only when the instance is active and service requests

When idle, only pay for the memory of min provisioned instances

Build Fee

$0.005 / build-minute

Only applies when starting with source code – Does not apply if starting with a container image

Automated Deployment Fee

$1 / service, per month

Only applies if automated deployments are enabled

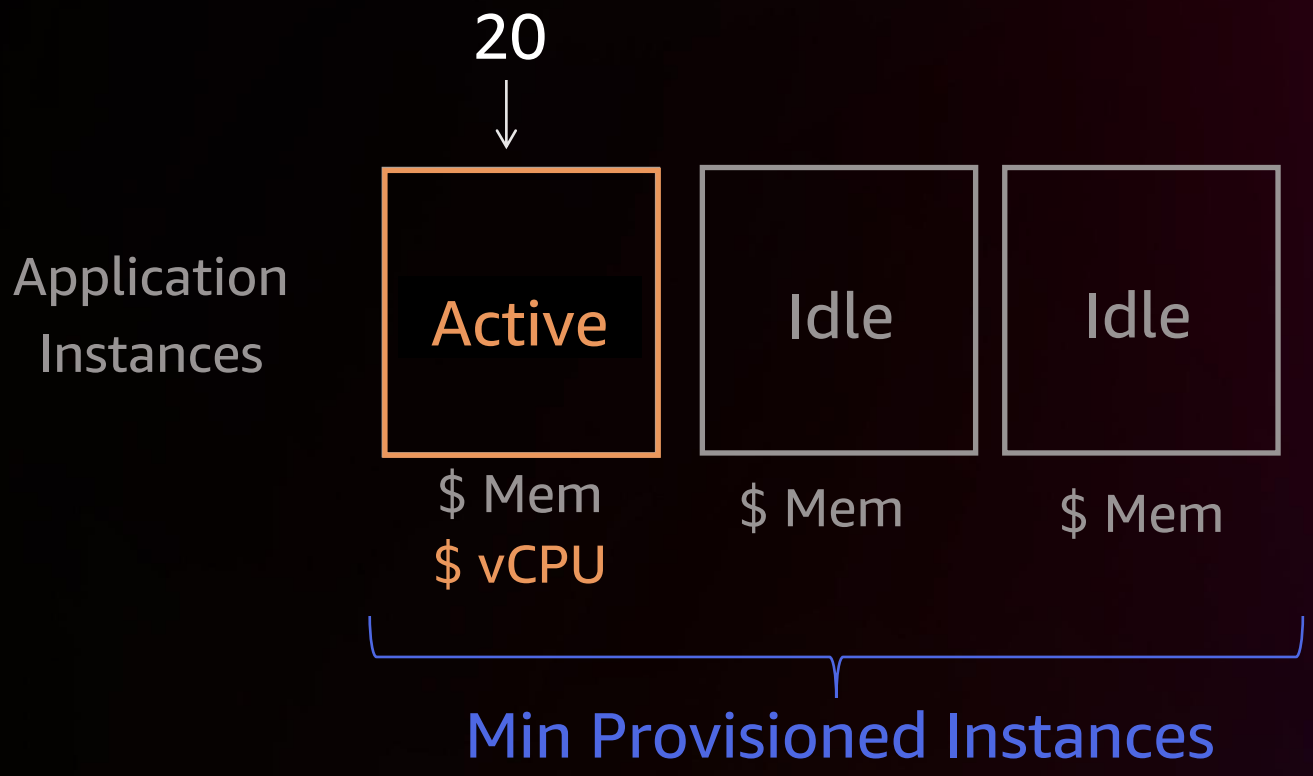
AWS Data Transfer

Standard AWS Data Transfer fees apply to application traffic

Pay-per-Use CPU/Memory Pricing

`concurrency = 30; min=3; max=5`

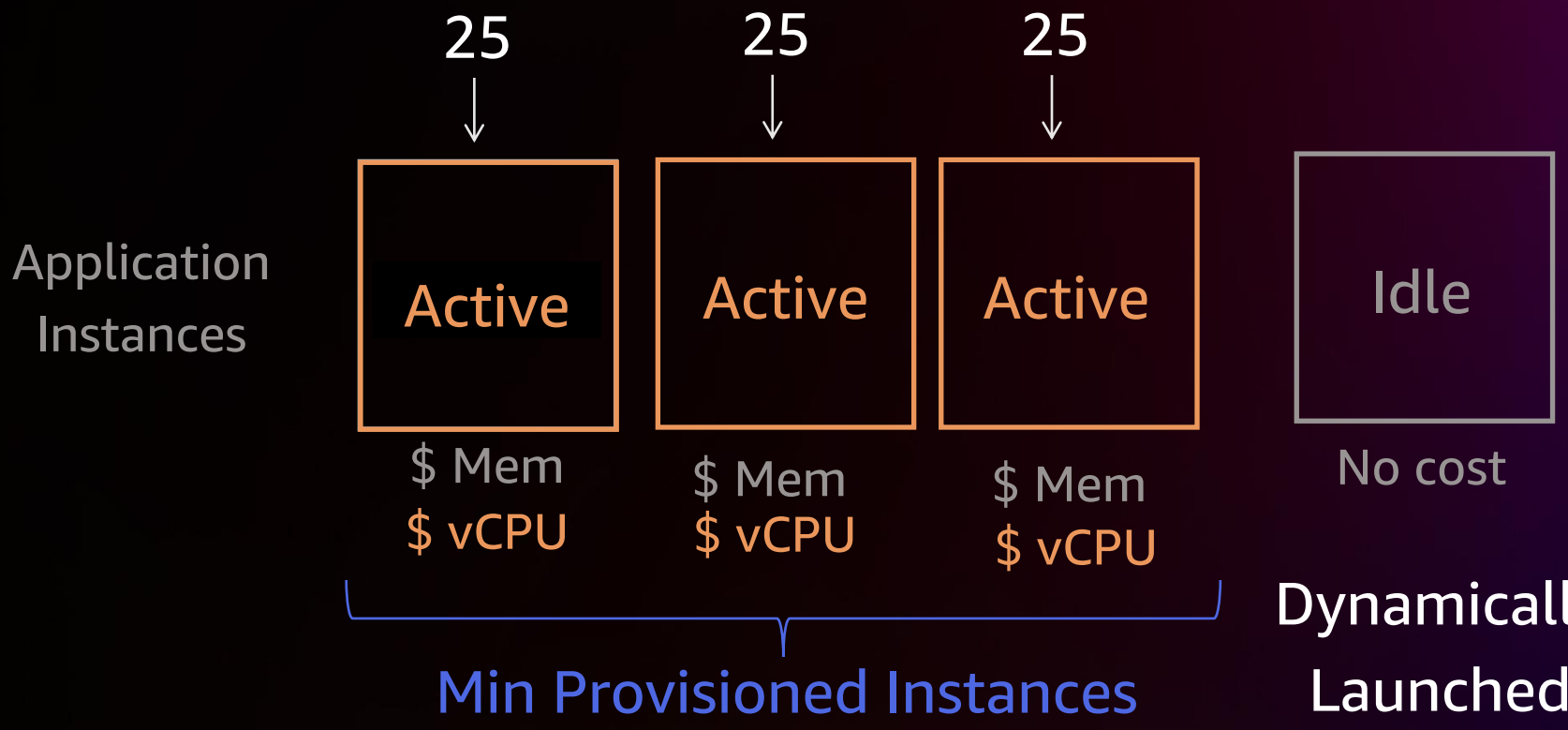
Load: 20 concurrent requests



Pay-per-Use CPU/Memory Pricing

`concurrency = 30; min=3; max=5`

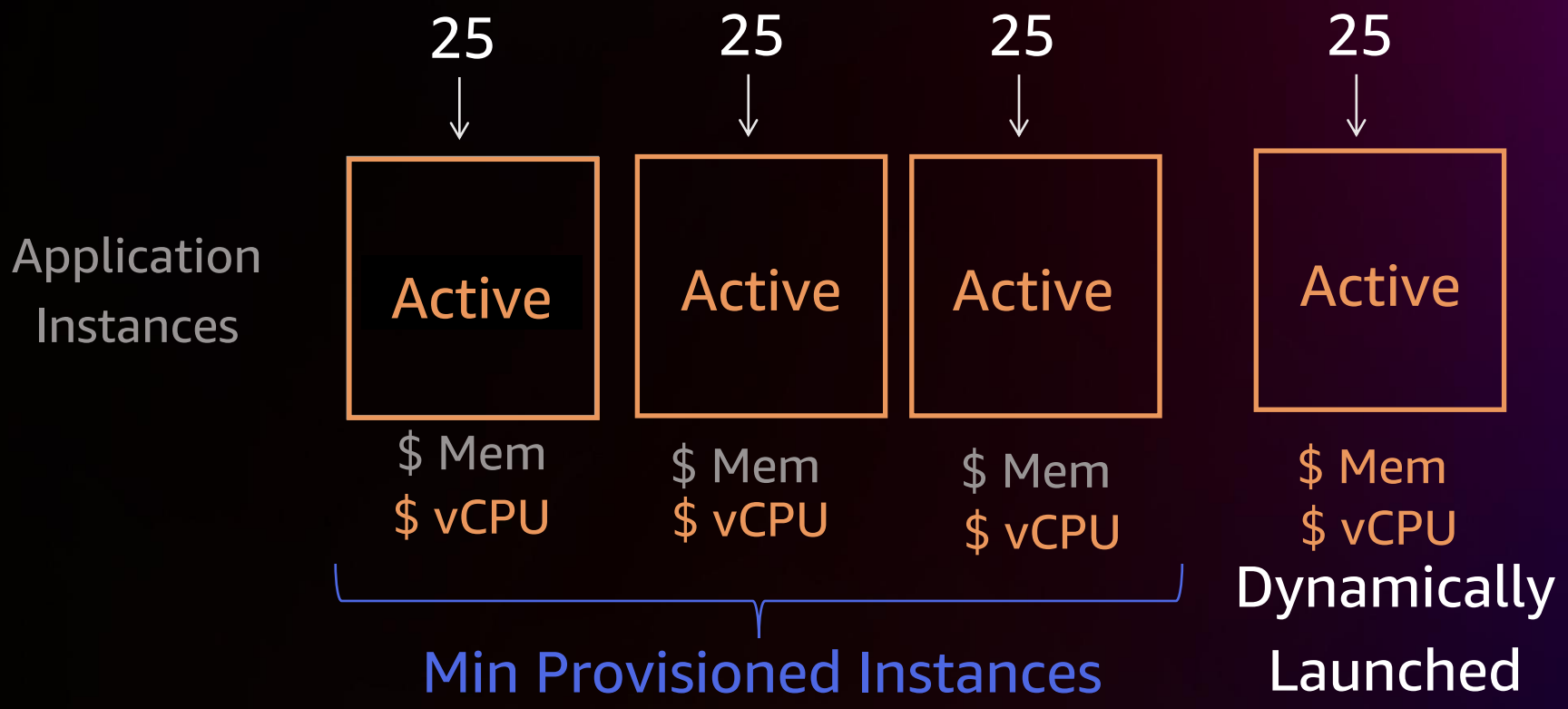
Load: 75 concurrent requests



Pay-per-Use CPU/Memory Pricing

`concurrency = 30; min=3; max=5`

Load: 100 concurrent requests



Takeaways

Takeaways

- App Runner is a simple abstracted experience for HTTP(s) request/reply based web applications and web services
- Autoscaling is one of many aspects that is taken care of behind the scenes
- Application Instance is the unit of scaling and concurrency is the dimension
- Pay-per-use pricing means you only pay for application instances that are serving traffic (provisioned instances aside)

Thank you!

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