modern apps

DevOps and CI/CD for Modern Applications

Serverless, Containers and AWS Proton

Wayne Wang, Enterprise Support Manager (EDU)
Daisy Daivasagaya, GTM Specialist for Container Services

Agenda

- DevOps
- Continuous Integration
- Continuous Delivery/Deployment
- Infrastructure as Code
- Blue-Green Deployments for Modern Applications
- AWS Proton

Why DevOps?

Why does DevOps matter?

5x

Lower change failure rate

440x

Faster from commit to deploy

46x

More frequent deployments

44%

More time spent on new features and code

30x

More Frequent Deployments

200x

Shorter Lead Times

60x

Fewer Failures

168x

Faster Recovery

Source: Puppet 2017 State of DevOps Report

What is DevOps?

- Cultural philosophies
- Practices
- Tools

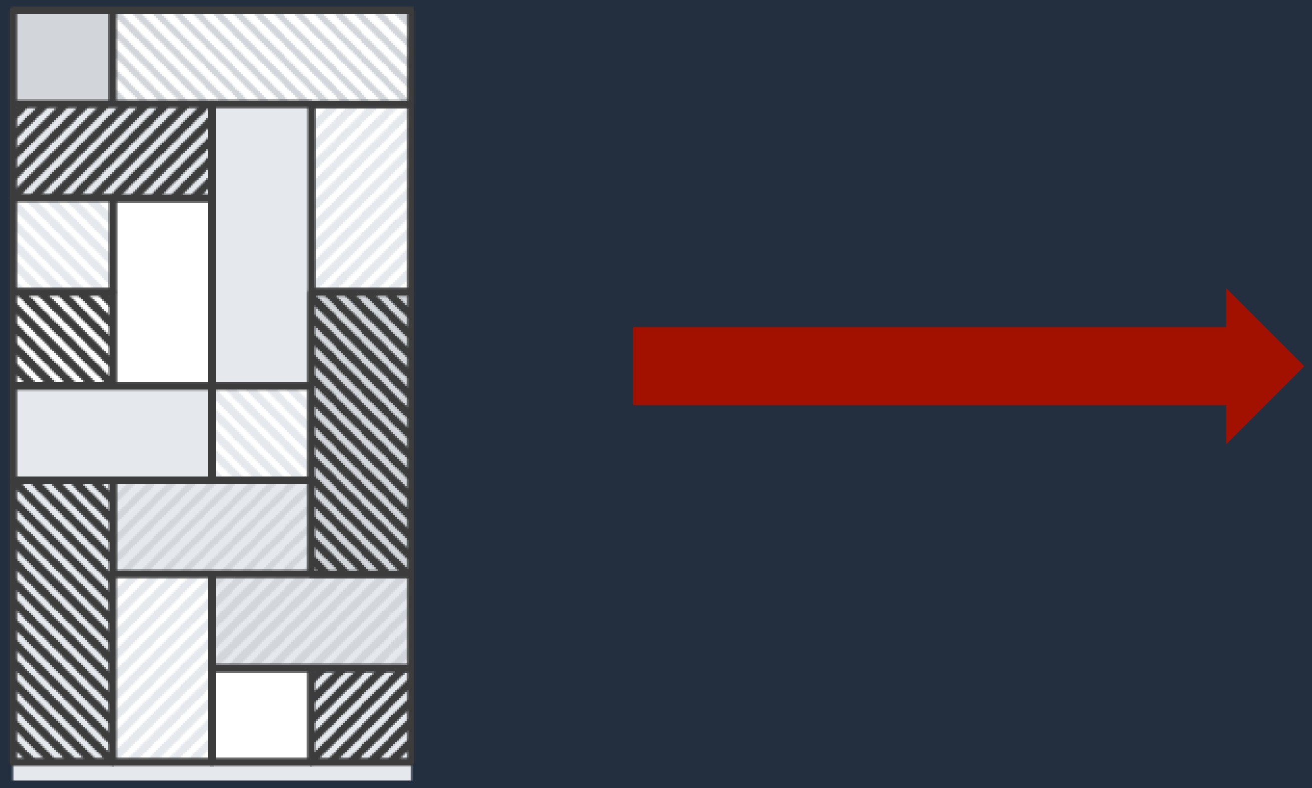
DevOps culture

- Dev & Ops coming together
  - No more “silos”
- Shared responsibility
- Ownership
- Visibility and communication

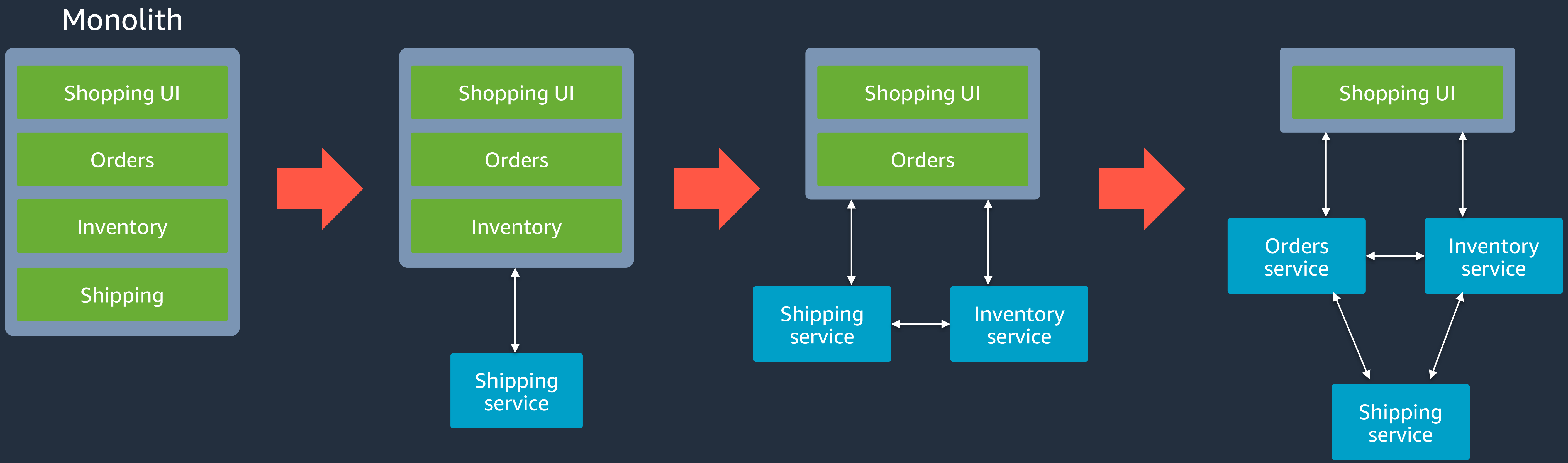
aws modern apps

DevOps practices

- Microservices
  - Moving away from “monolithic” application architecture to many individual services



Monolith to microservices



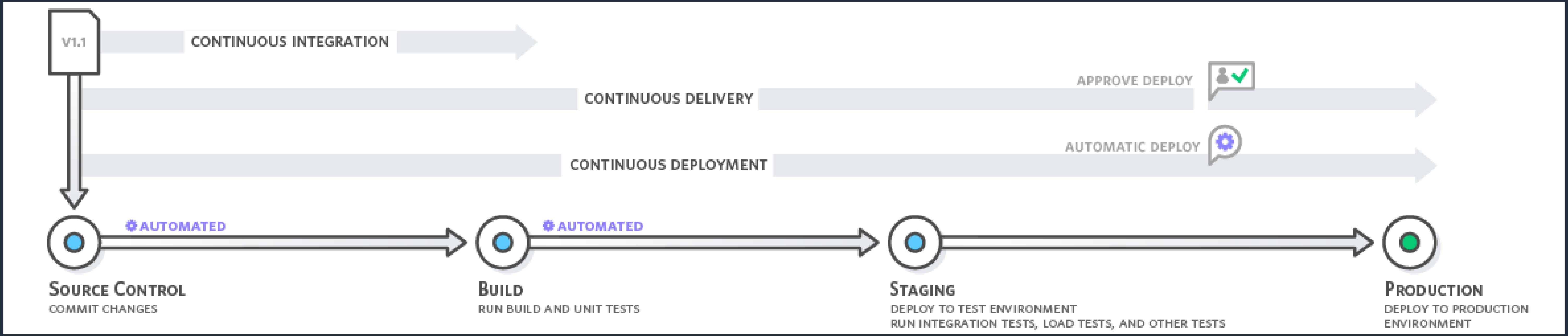
Strangler Application Pattern:

https://www.martinfowler.com/bliki/StranglerApplication.html

aws modern apps

DevOps practices

- Continuous Integration
- Continuous Delivery & Deployment



Continuous integration goals

Continuous integration

1. Automatically kick off a new release when new code is checked in
2. Build and test code in a consistent, repeatable environment
3. Continually have an artifact ready for deployment
4. Continually close feedback loop when build fails

aws modern apps

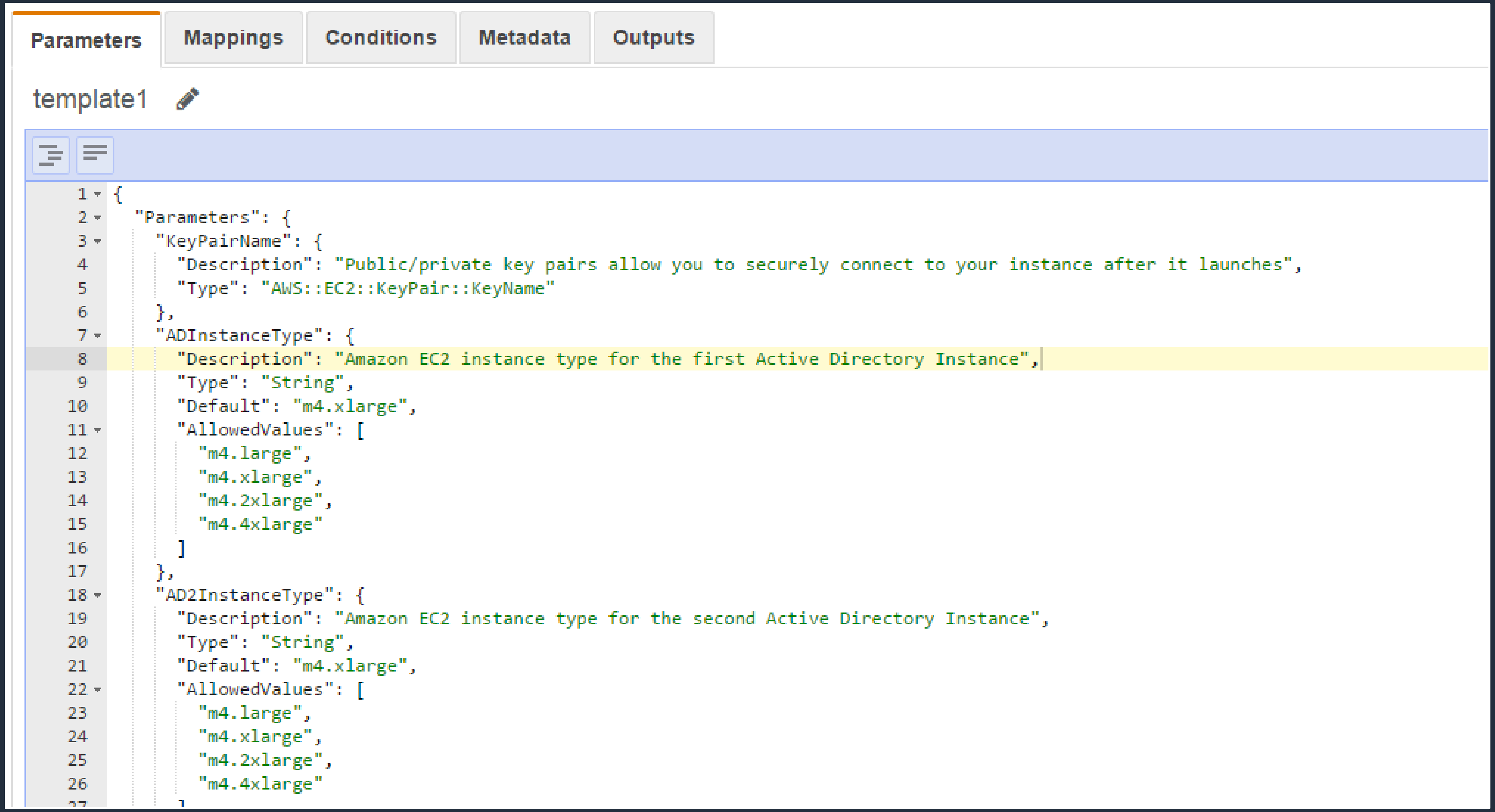
Continuous deployment/delivery goals

Continuous deployment/delivery

1. Automatically deploy new changes to staging environments for testing
2. Deploy to production safely without impacting customers
3. Deliver to customers faster: Increase deployment frequency, and reduce change lead time and change failure rate

DevOps practices

- Infrastructure as Code
    - Model your AWS resources using code



```
{
  "Parameters": {
    "KeyPairName": {
      "Description": "Public/private key pairs allow you to securely connect to your instance after it launches",
      "Type": "AWS::EC2::KeyPair::KeyName"
    },
    "ADInstanceType": {
      "Description": "Amazon EC2 instance type for the first Active Directory Instance",
      "Type": "String",
      "Default": "m4.xlarge",
      "AllowedValues": [
        "m4.large",
        "m4.xlarge",
        "m4.2xlarge",
        "m4.4xlarge"
      ]
    },
    "AD2InstanceType": {
      "Description": "Amazon EC2 instance type for the second Active Directory Instance",
      "Type": "String",
      "Default": "m4.xlarge",
      "AllowedValues": [
        "m4.large",
        "m4.xlarge",
        "m4.2xlarge",
        "m4.4xlarge"
```

aws modern apps

Infrastructure as code goals

Infrastructure as code

1. Make infrastructure changes repeatable and predictable
2. Release infrastructure changes using the same tools as code changes
3. Replicate production environment in a staging environment to enable continuous testing

Model function environments with AWS Serverless Application Model (SAM)

- Open source framework for building serverless applications on AWS
- Shorthand syntax to express functions, APIs, databases, and event source mappings
- Transforms and expands SAM syntax into AWS CloudFormation syntax on deployment
- Supports all AWS CloudFormation resource types

https://aws.amazon.com/serverless/sam/

aws modern apps

DevOps practices

- Monitoring and Logging
  - Track and analyze metrics and logs
  - Understand real-time performance of infrastructure and application

Elements of Modern Applications

aws modern apps

Elements of Modern Applications

- Application Architecture: Modular Microservices
- Software Delivery: Automation, Abstraction, & Standardization
- Data Strategy: Decoupled & Purpose Built
- Operations: As Serverless as Possible
- Management & Governance: Programmatic Guardrails

aws modern apps

Approaches to modern application development

- Simplify environment management with serverless technologies
- Reduce the impact of code changes with microservice architectures
- Automate operations by modeling applications & infrastructure as code
- Accelerate the delivery of new, high-quality services with CI/CD
- Gain insight across resources and applications by enabling observability
- Protect customers and the business with end-to-end security & compliance

Approaches to modern application development

AWS Lambda

Serverless functions

Event-driven
Many language runtimes
Data source integrations
No server management

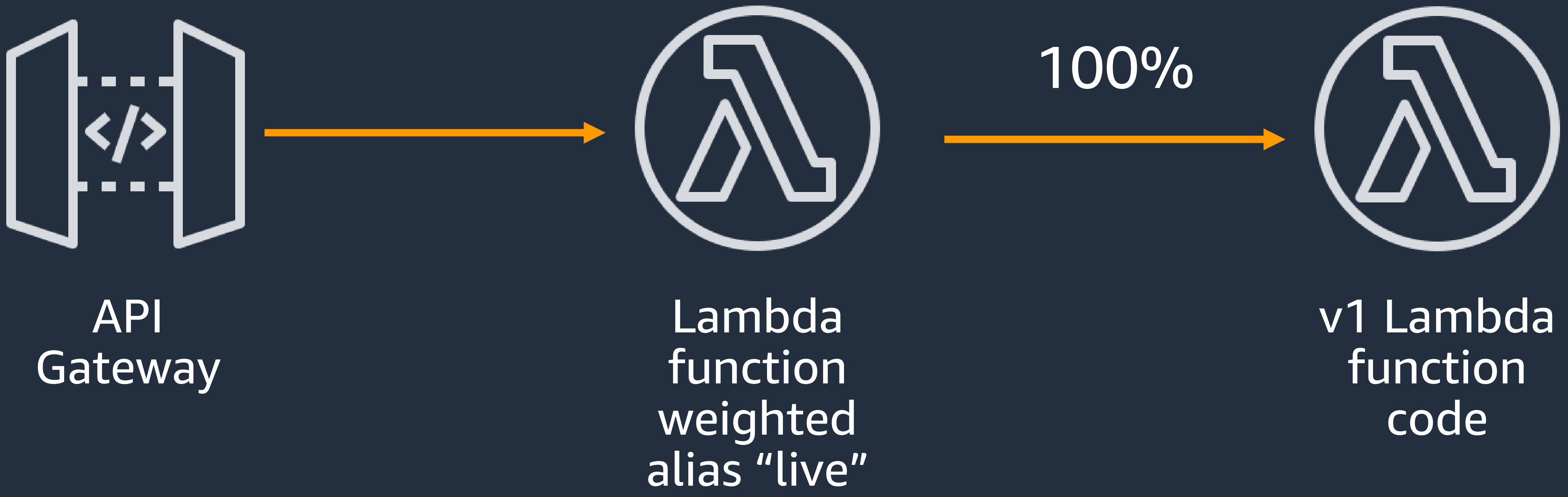
AWS Fargate

Serverless containers

Long-running
Abstracts the OS
Fully managed orchestration
Fully managed cluster scaling

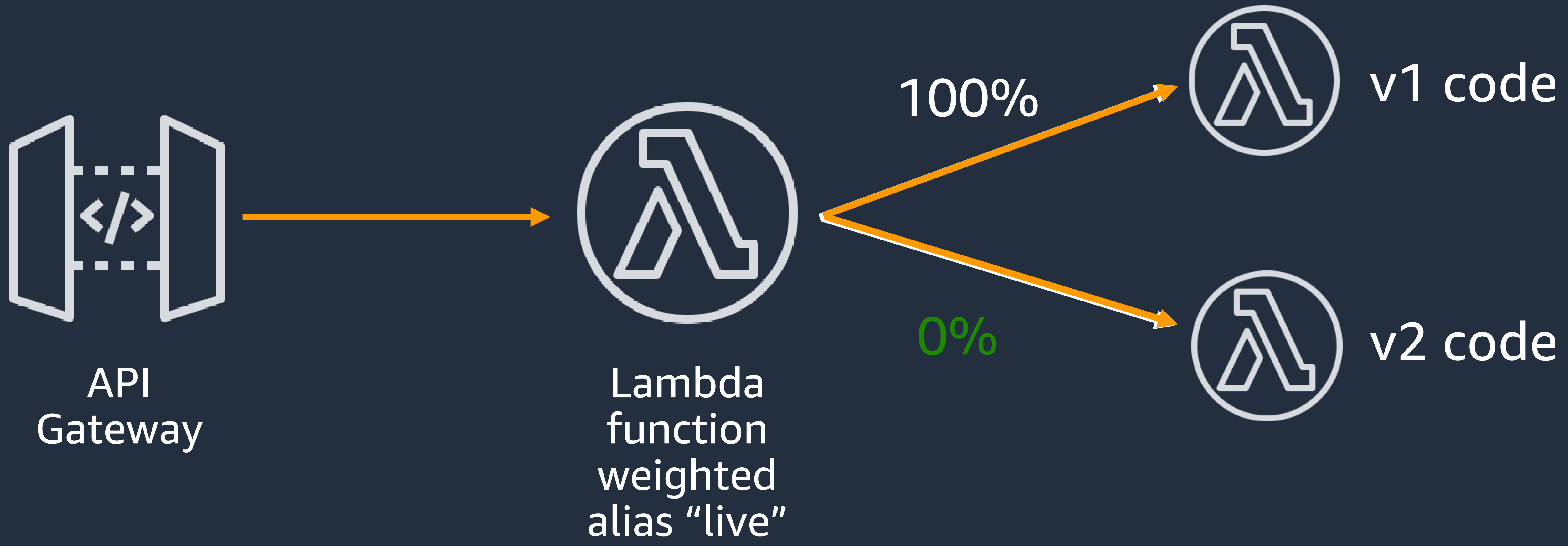
Serverless Blue-Green Deployments

CodeDeploy-Lambda canary deployment



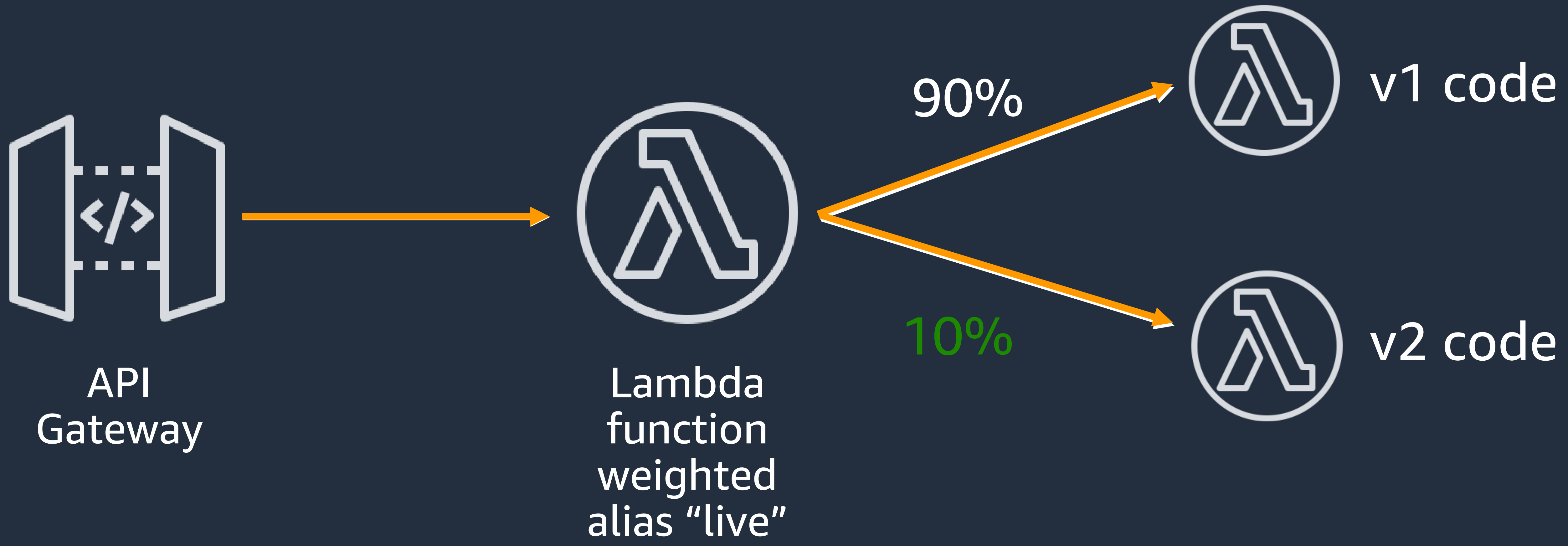
CodeDeploy-Lambda canary deployment

Run hook against v2 code before it receives traffic



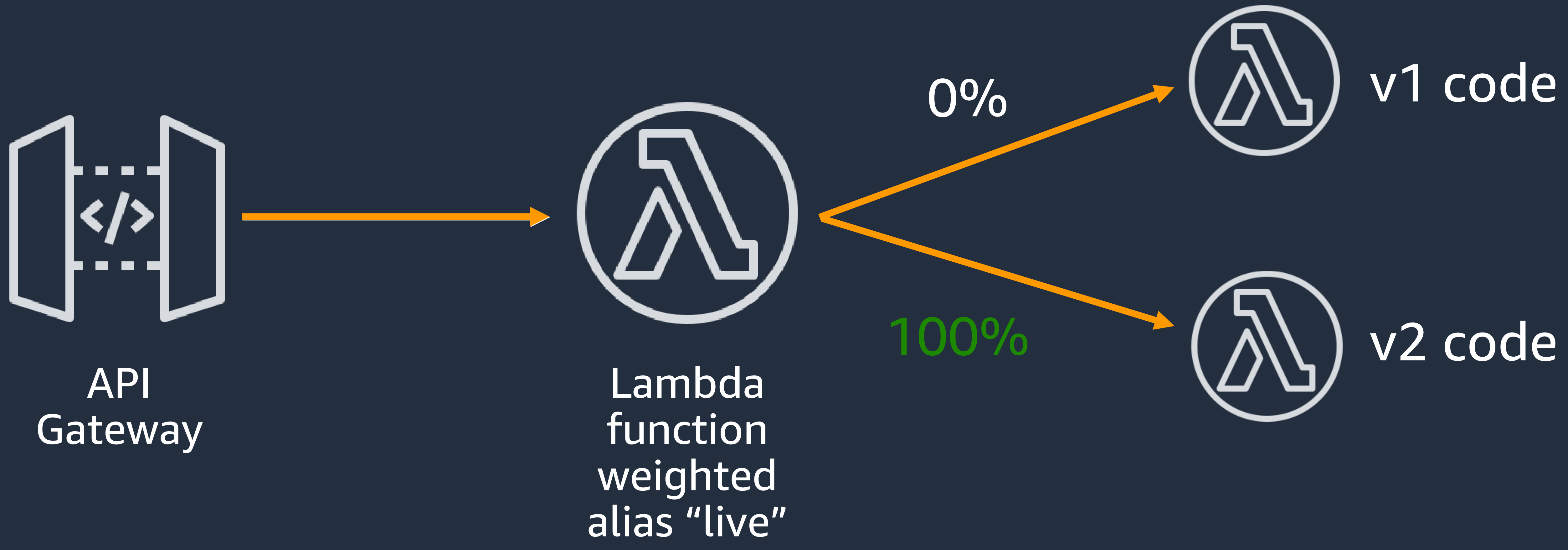
CodeDeploy-Lambda canary deployment

Wait for 10 minutes, roll back in case of alarm



CodeDeploy-Lambda canary deployment

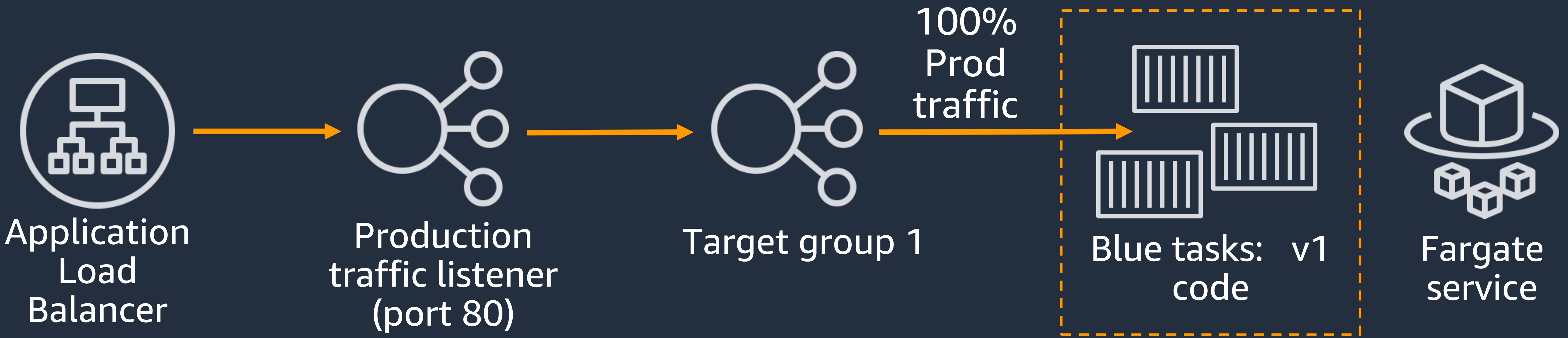
Complete deployment



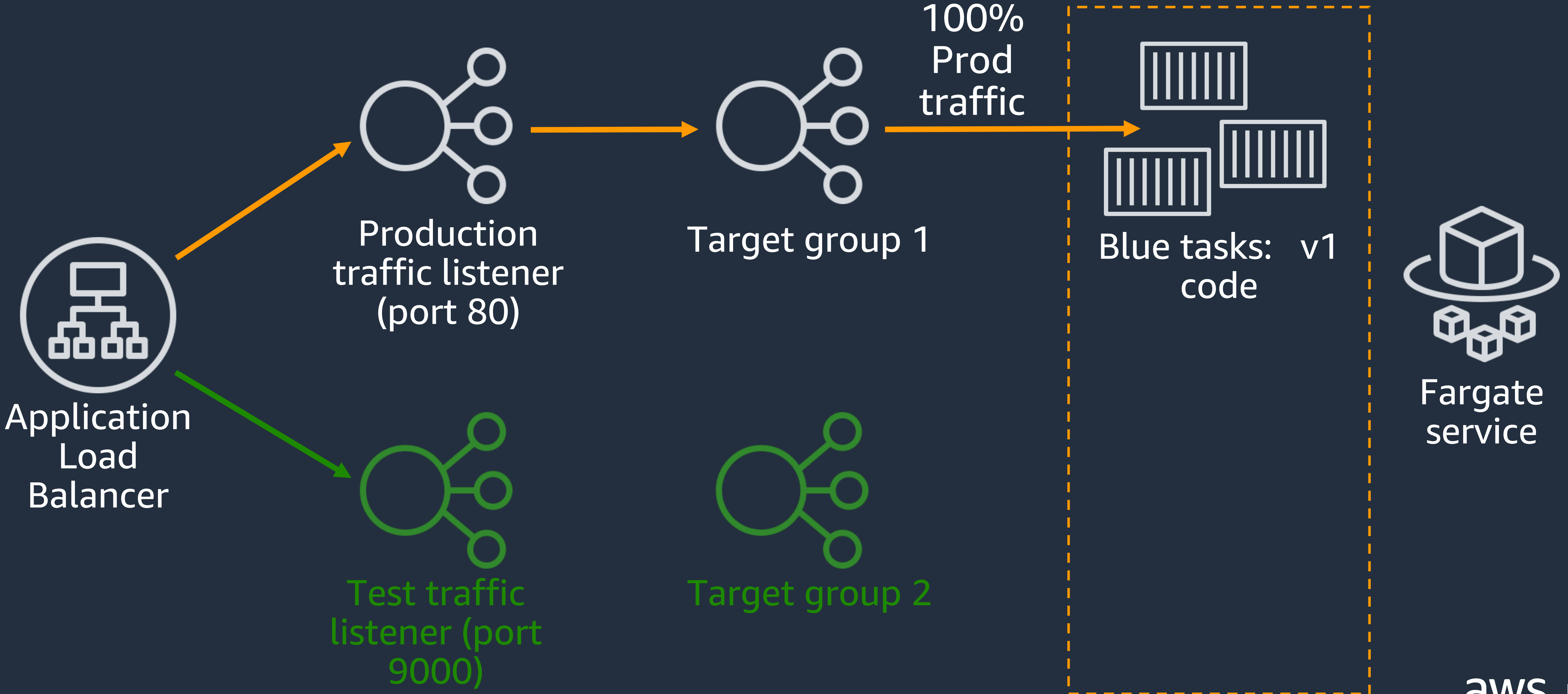
Container Blue-Green Deployments

aws modern apps

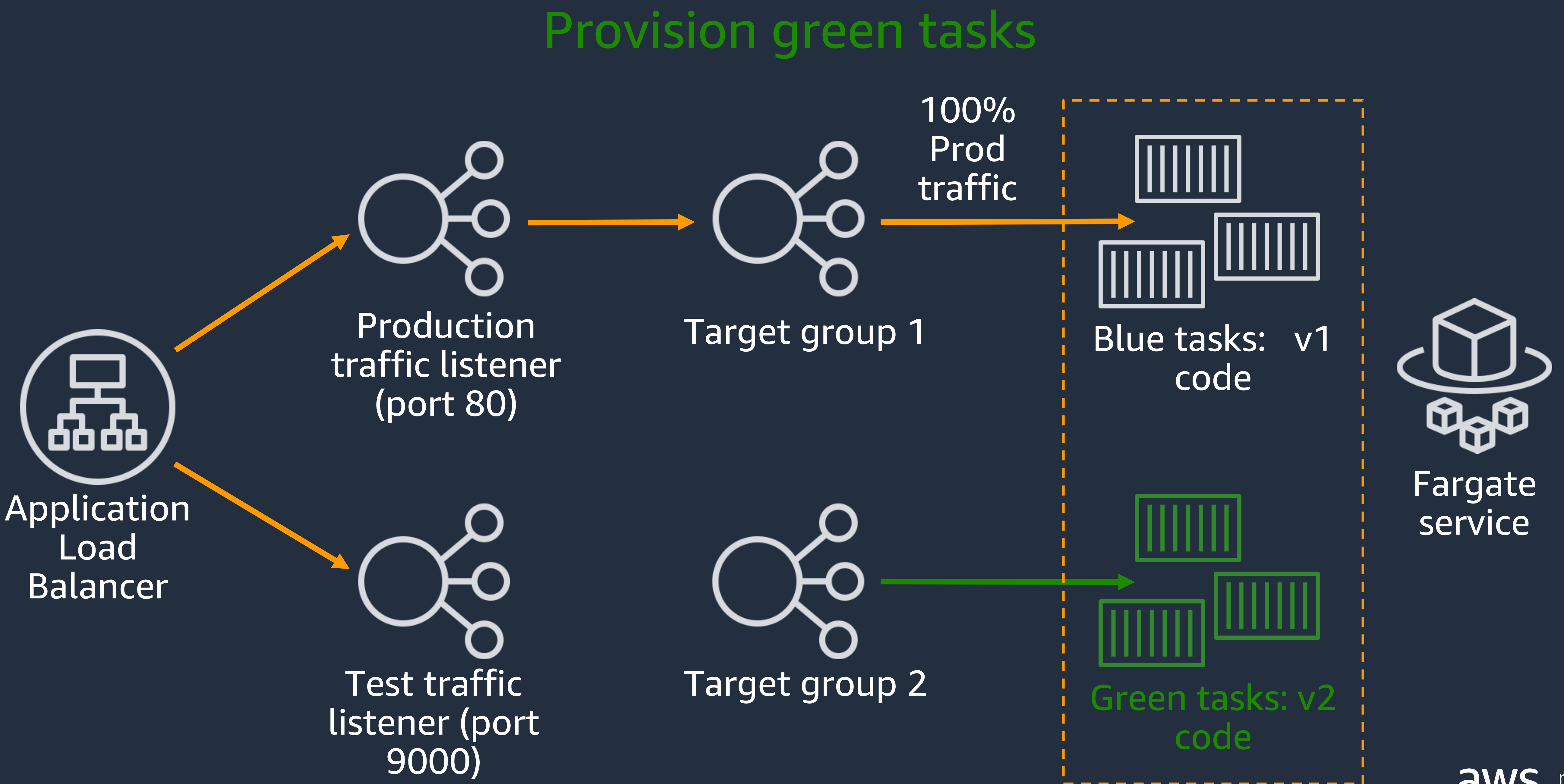
CodeDeploy-ECS blue-green deployment



CodeDeploy-ECS blue-green deployment

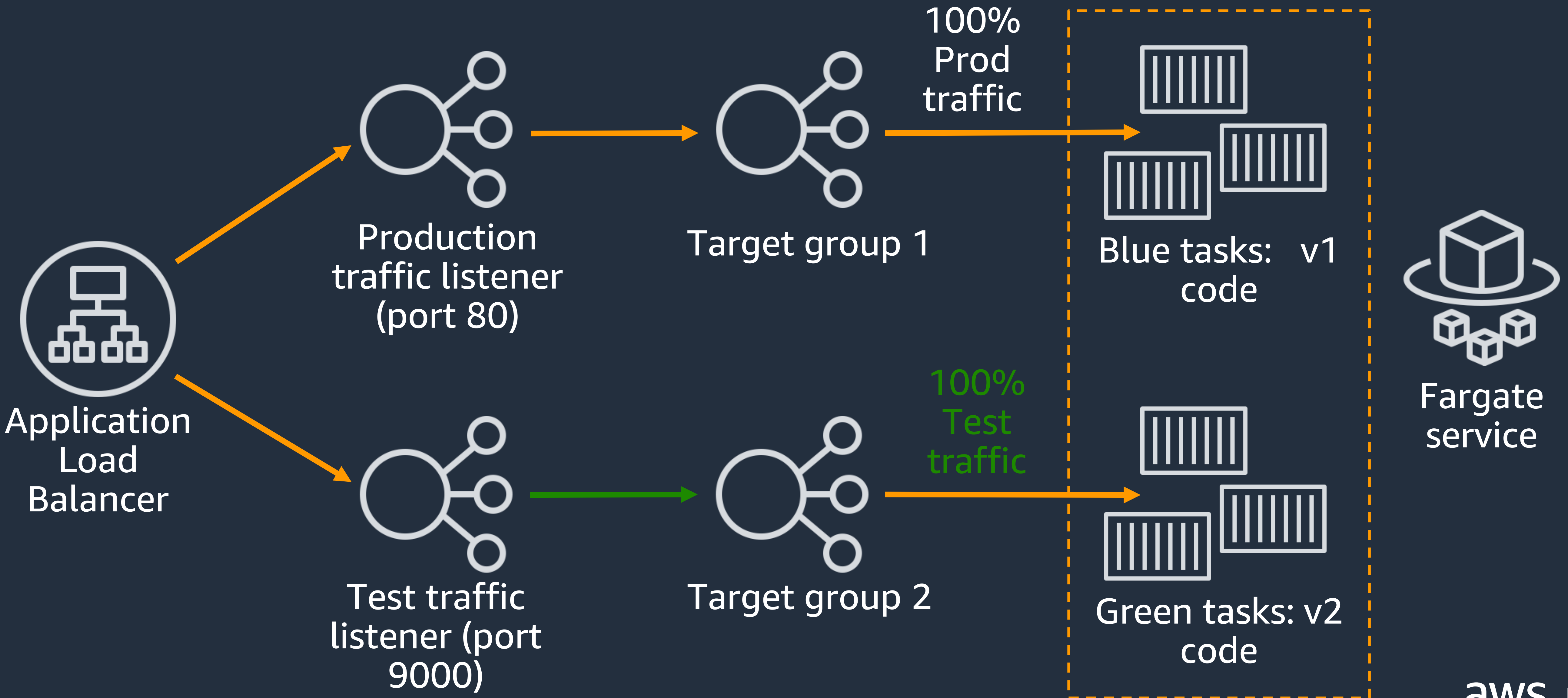

100% Prod traffic
Production traffic listener (port 80)
Target group 1
Blue tasks: v1 code
Application Load Balancer
Fargate service
Test traffic listener (port 9000)
Target group 2

CodeDeploy-ECS blue-green deployment



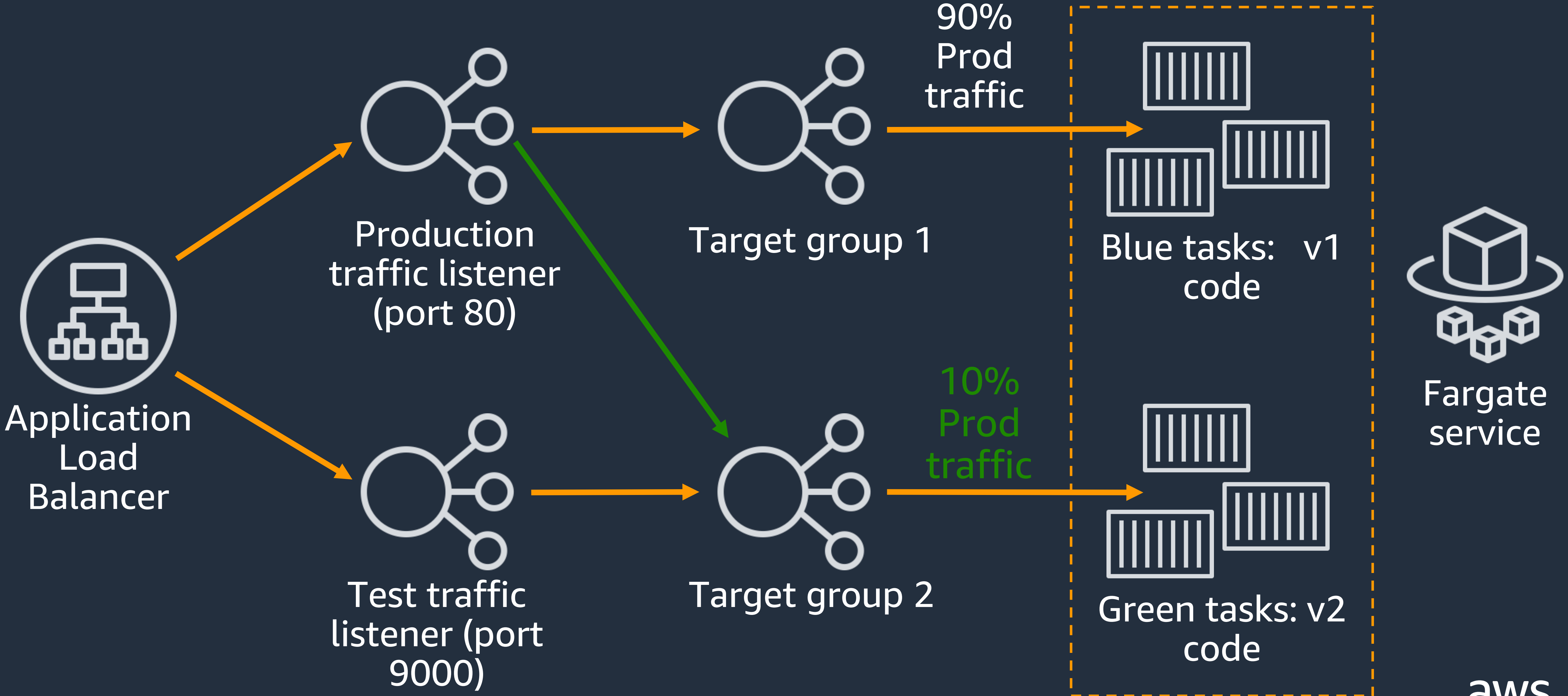
CodeDeploy-ECS blue-green deployment

Run hook against test endpoint before green tasks receive prod traffic



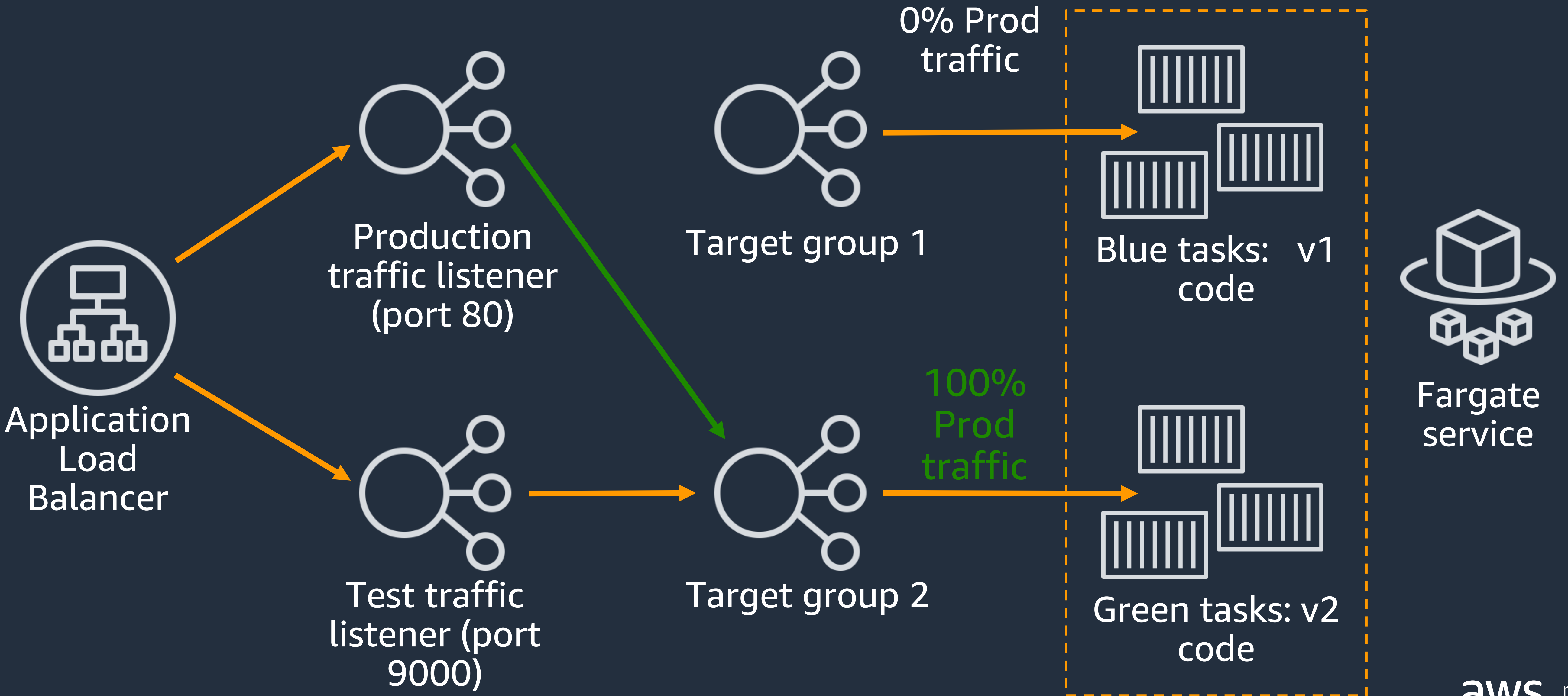
CodeDeploy-ECS blue-green deployment

Flip traffic to green tasks, rollback in case of alarm

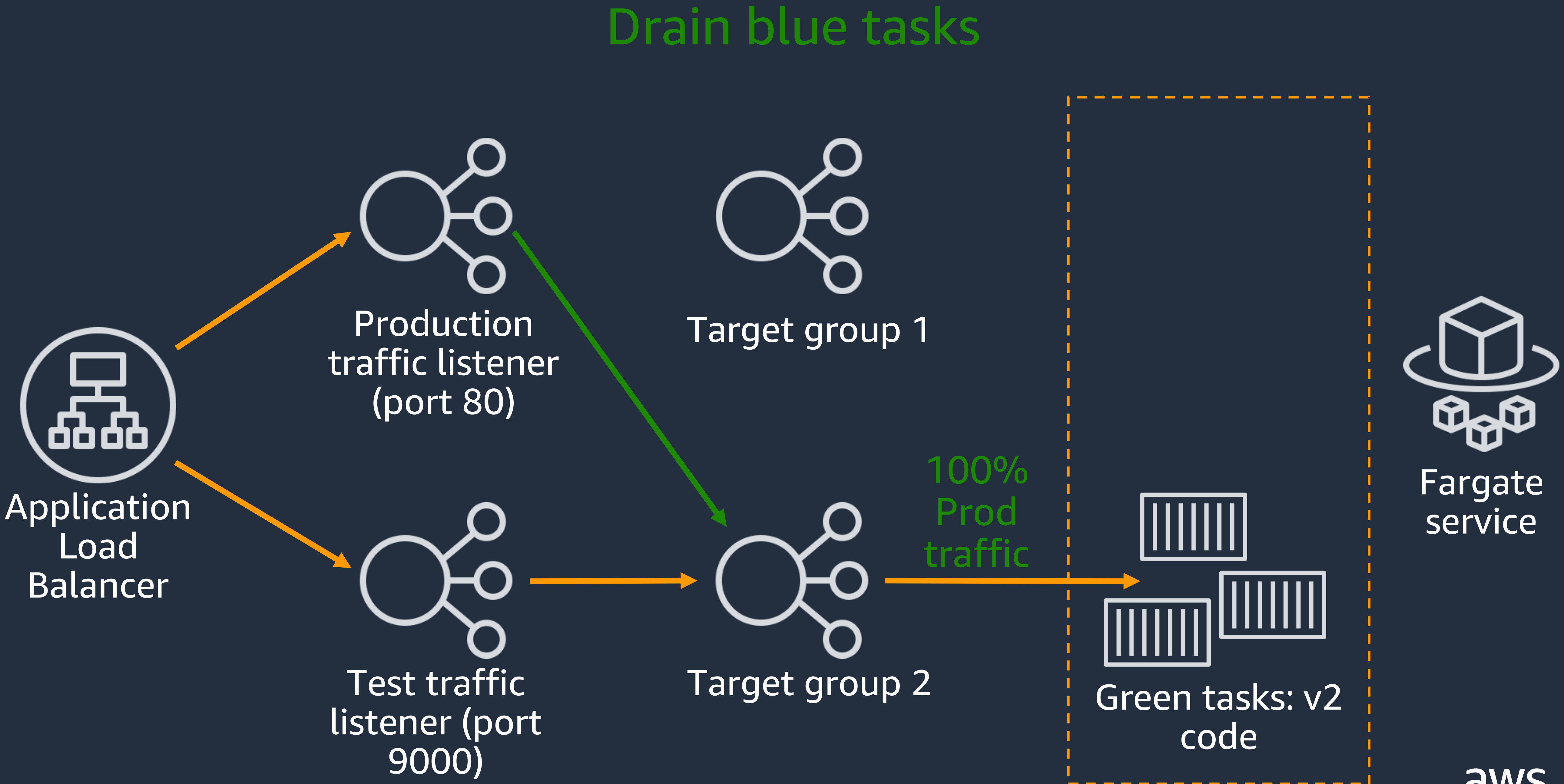


CodeDeploy-ECS blue-green deployment

Flip traffic to green tasks, rollback in case of alarm



CodeDeploy-ECS blue-green deployment



How To Measure Success

aws modern apps

IT/Software delivery performance

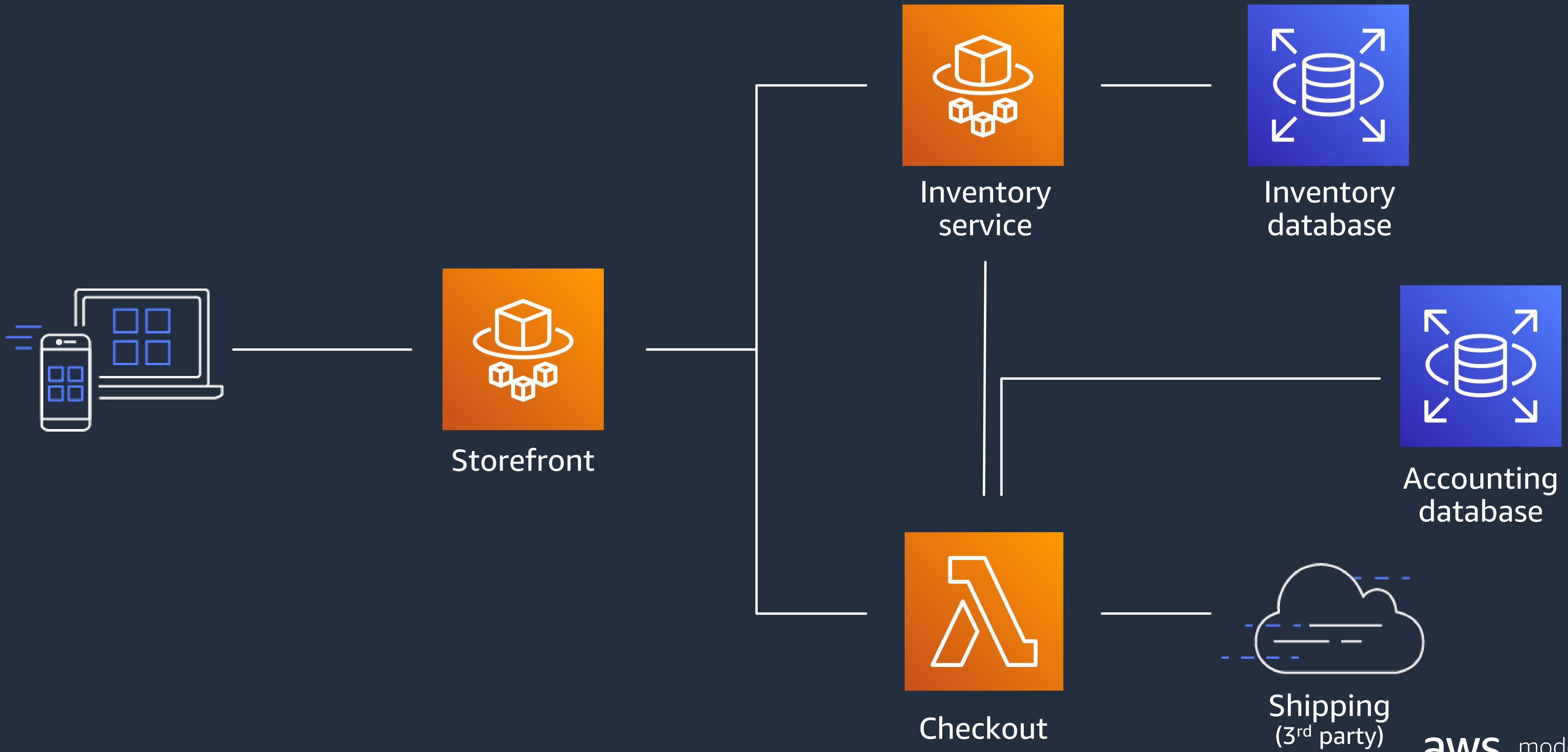
Aspect of Software Delivery Performance	Elite	High	Medium	Low
Deployment Frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	On-demand (multiple deploys per day)	Between once per day and once per week	Between once per week and once per month	Between once per month and once every six months
Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Less than one day	Between one day and one week	Between one week and one month	Between one month and six months
Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?	Less than one hour	Less than one day	Less than one day	Between one week and one month
Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	0-15%	0-15%	0-15%	46-60%

Source: State of DevOps 2019

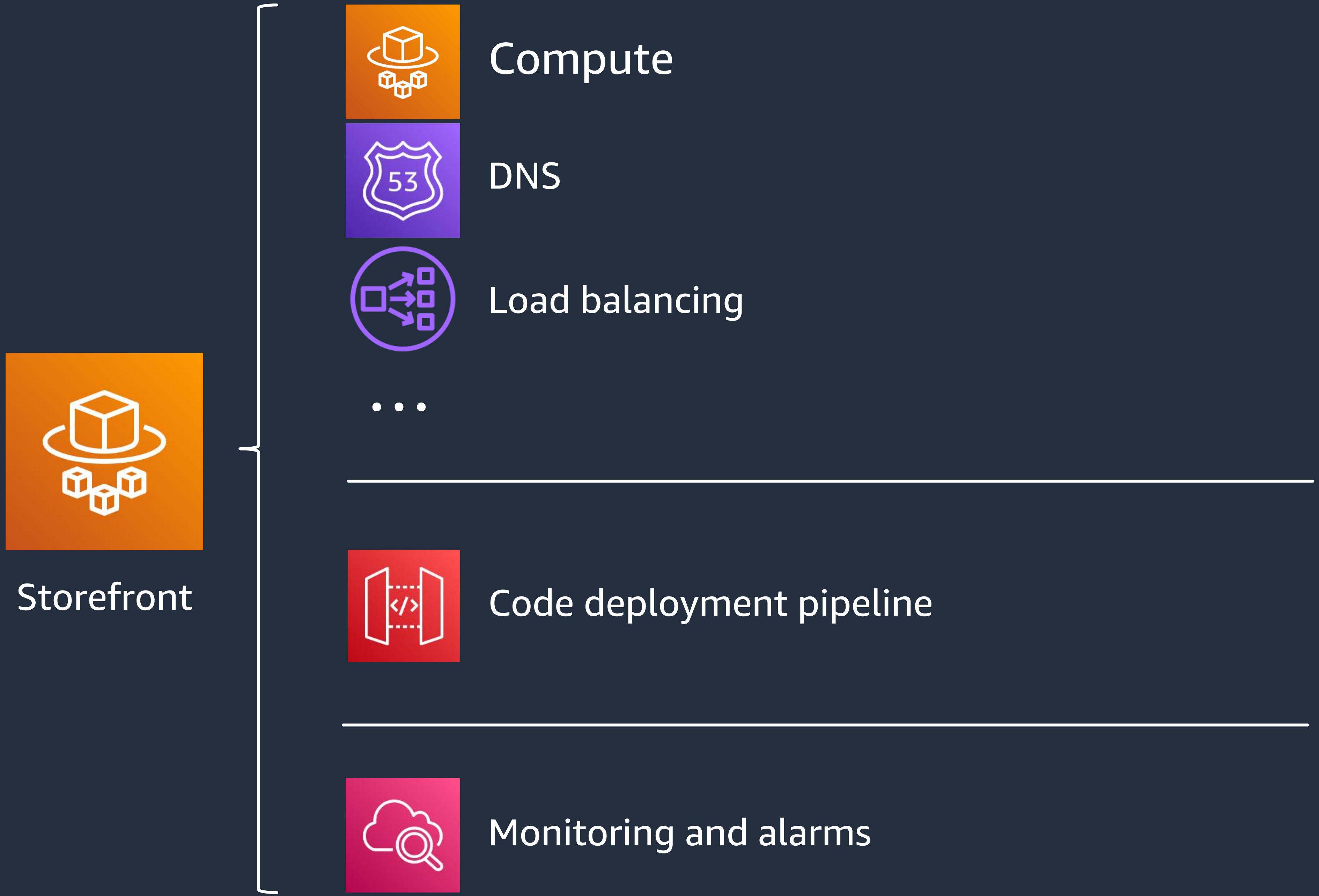
AWS Proton

modern apps

Using microservices for speed and agility



“Simple” is not so simple



Customers are building internal developer platforms on AWS to tie it all together

intuit®

Modern SaaS

Switchboard

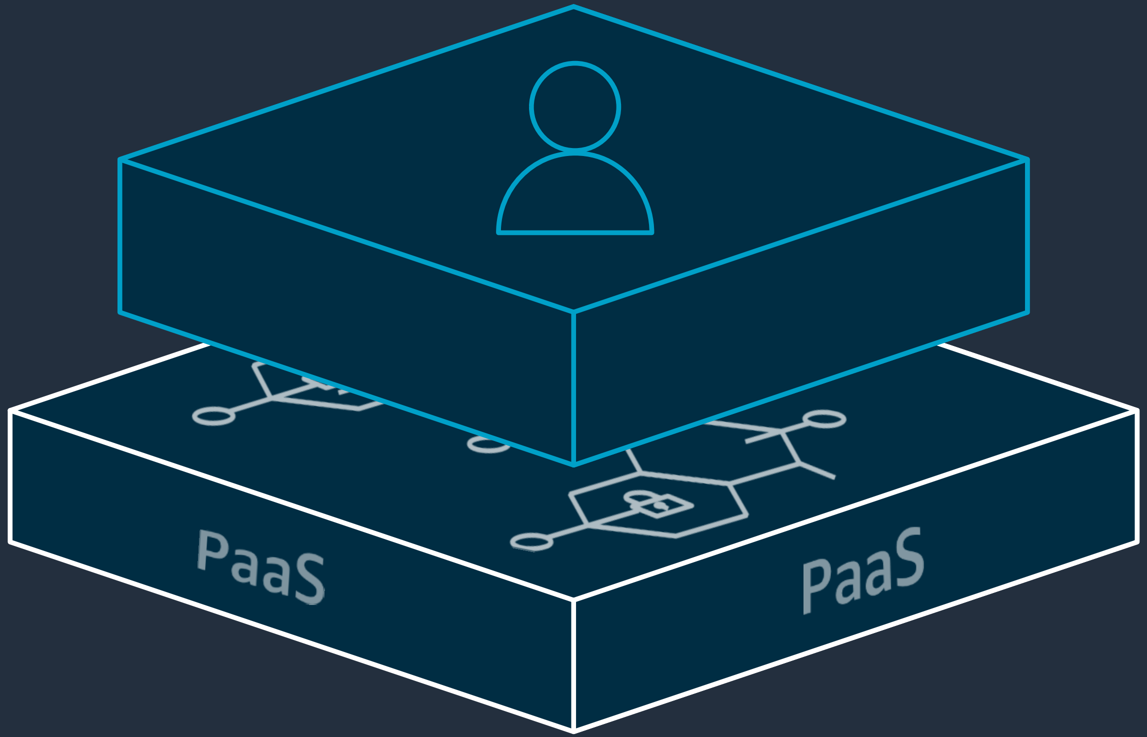
Skyscanner

Slingshot

These platforms unlock innovation by increasing developer productivity and accelerating software delivery

aws modern apps

Abstract away operator overhead



- Creating a developer platform is hard
- Need to stand up a consistent stack to improve everyone's productivity
- Also to provide better guardrails for protection
- Managing and updating hundreds or thousands of deployed microservices is painful

Application management as a spectrum

PaaS

Platform as a Service

↓ Flexibility & Control

↑ Developer Productivity

Ideal

↑ Flexibility & Control

↑ Developer Productivity

IaaS

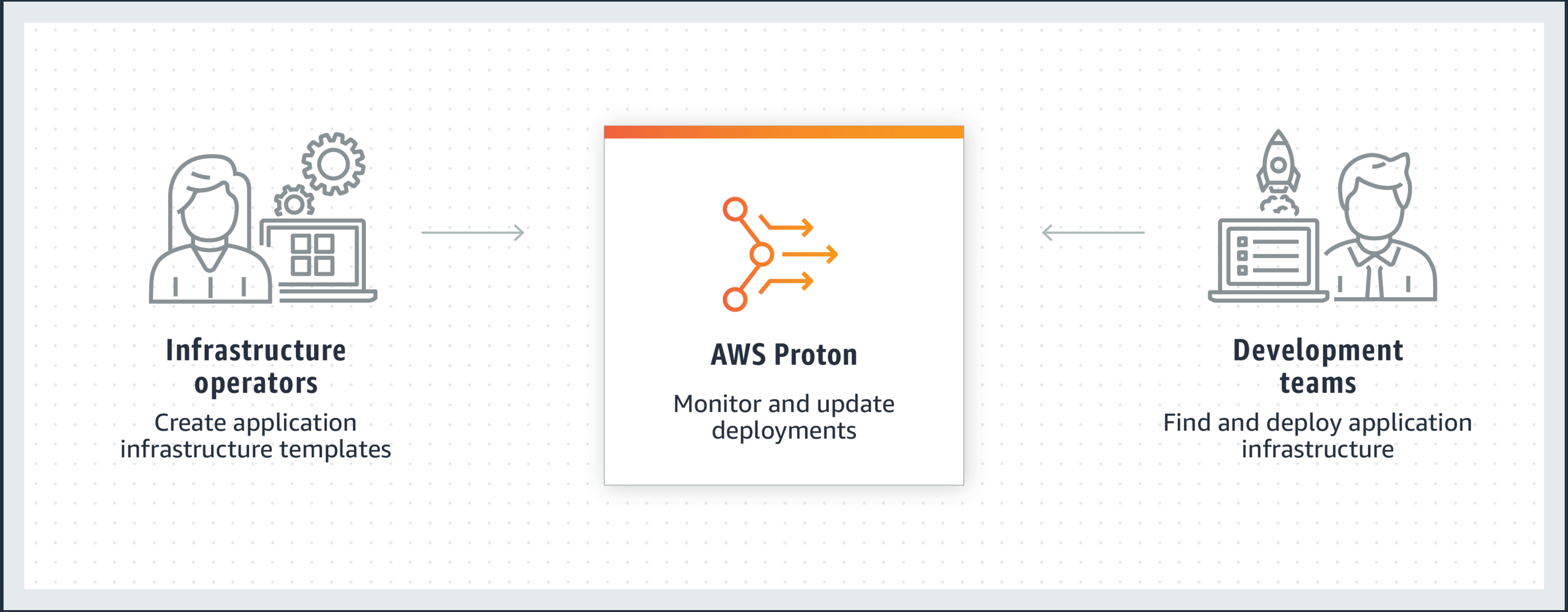
Infrastructure as a Service

↑ Flexibility & Control

↓ Developer Productivity

AWS Proton

Increase control over your cloud infrastructure, accelerating the pace of innovation for your development teams



Infrastructure operators

Create application infrastructure templates

AWS Proton

Monitor and update deployments

Development teams

Find and deploy application infrastructure

AWS Proton Demo