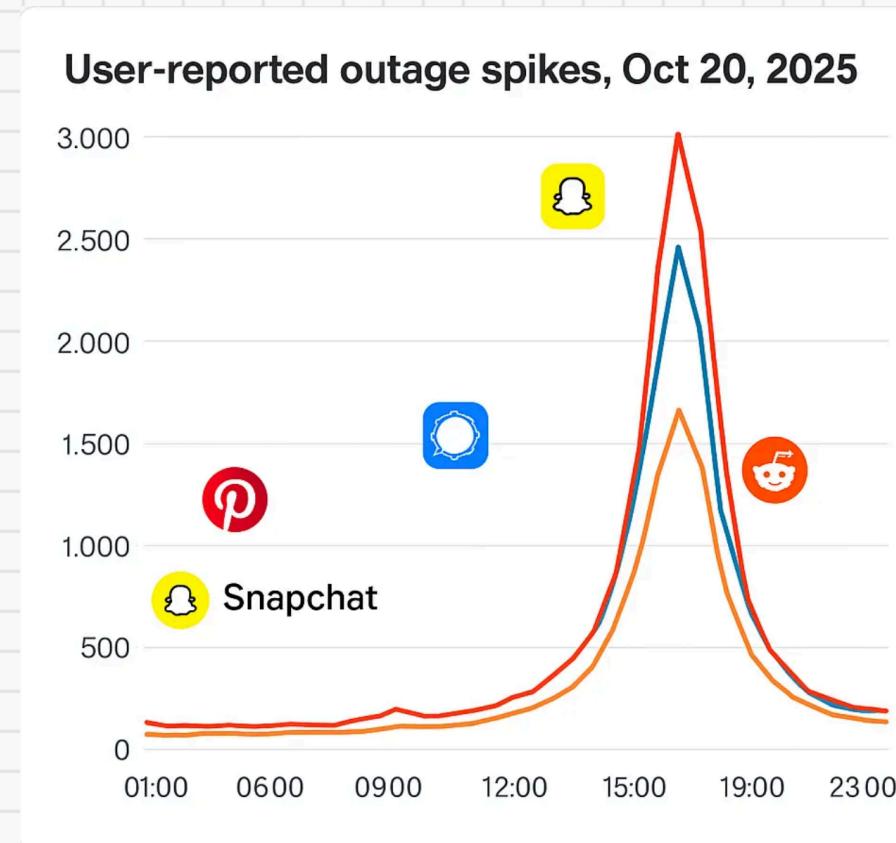


Multi-Region EKS with Global Accelerator

High Availability Kubernetes on AWS

Why Multi-Region ?

AWS US-EAST-1 Outage (October 20–21, 2025)



Who survives or is less affected?

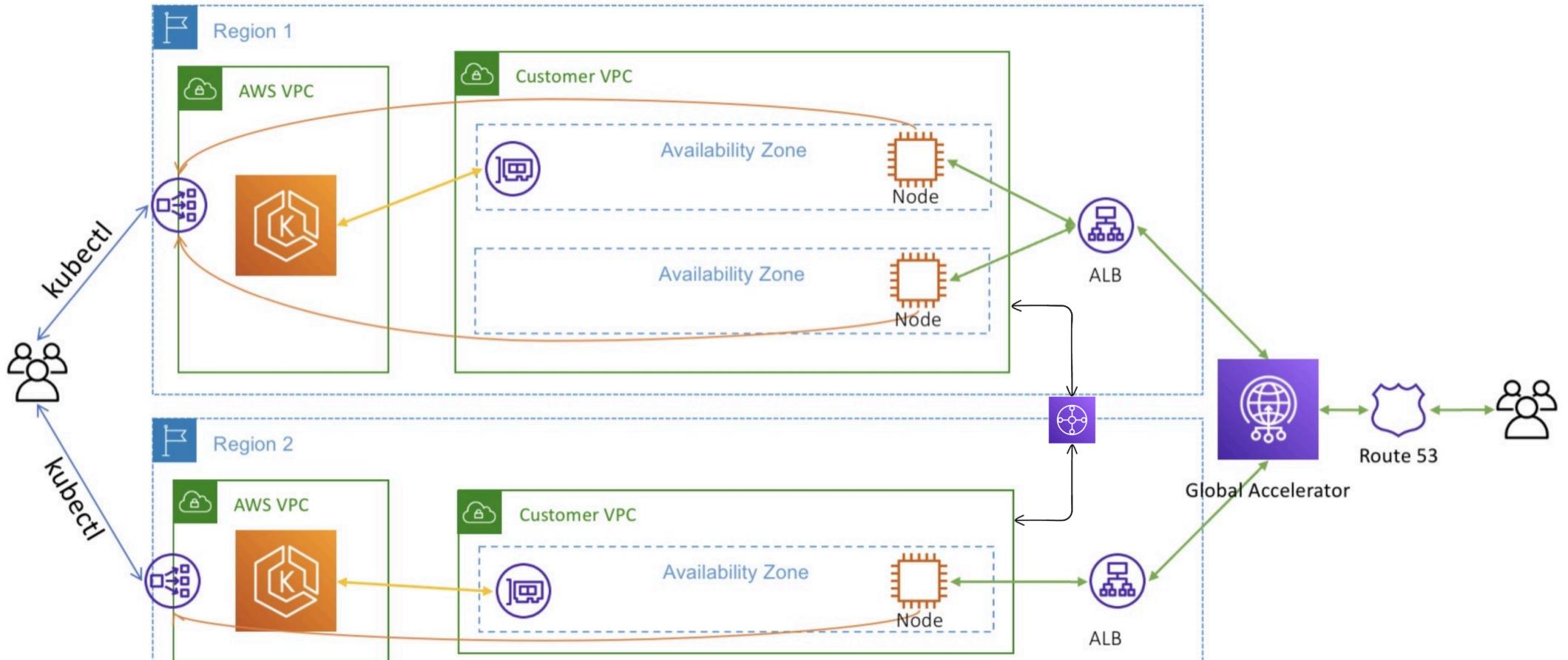
Architecture Overview

Multi-region stateless applications using Amazon EKS

Key Components:

- **EKS Clusters:** Primary (us-east-1) and Secondary (us-west-2)
- **Global Accelerator:** Static anycast IPs with automatic failover
- **Route53:** DNS management with alias records
- **Network Load Balancers:** Regional endpoints
- **Transit Gateway:** Cross-region connectivity

Deployment Architecture



Network Architecture & Traffic Flow

Regional Setup:

- Primary (us-east-1): VPC `10.0.0.0/16`, 3 AZs, EKS v1.31
- Secondary (us-west-2): VPC `10.1.0.0/16`, 3 AZs, EKS v1.31
- Worker Nodes: t3.medium (Min: 1, Max: 6, Desired: 3)
- Application: Nginx (3 replicas/region)

Traffic Flow:

1. Client → Global Accelerator (anycast IPs)
2. Global Accelerator → Regional NLB (primary: 100%)
3. NLB → EKS Service LoadBalancer
4. Service → Application Pods

Global Accelerator & Failover

Static IPs:

- 52.223.29.64 / 166.117.186.20

Configuration:

- **Protocol:** TCP Port 80
- **Affinity:** SOURCE_IP
- **Primary:** 100% (active)
- **Secondary:** 0% (standby)

Failover Mechanism:

- **Health Checks:** Global Accelerator monitors NLB health
- **Automatic Failover:** Traffic shifts to secondary region on failure
- **Traffic Dial:** Gradual traffic shifting (0-100%)

Transit Gateway: Cross-Region Connectivity

Purpose:

- Private VPC peering between regions (10.0.0.0/16 <--> 10.1.0.0/16)
- Pod-to-pod communication across EKS clusters
- Shared services access (databases, caching, monitoring)

Use Cases:

- Stateful applications requiring cross-region sync
- Centralized logging/monitoring infrastructure
- Multi-region microservices communication

Demo

```
resource "aws_globalaccelerator_endpoint_group" "secondary" {
  listener_arn = aws_globalaccelerator_listener.main.id

  endpoint_group_region    = local.regions.secondary
  traffic_dial_percentage = 0

  endpoint_configuration {
    endpoint_id = "arn:aws:elasticloadbalancing:ap-northeast-1:488309743291:loadbalancer/net/a4d0
    weight      = 100
  }
}
```

```
# Run: ./get-https-arns.sh
```

```
resource "aws_globalaccelerator_endpoint_group" "primary" {
  listener_arn = aws_globalaccelerator_listener.main.id

  endpoint_group_region    = local.regions.primary
  traffic_dial_percentage = 100
```

```
resource "aws_globalaccelerator_endpoint_group" "primary" {
  listener_arn = aws_globalaccelerator_listener.main.id

  endpoint_group_region    = local.regions.primary
  traffic_dial_percentage = 50

  endpoint_configuration {
    endpoint_id = "arn:aws:elasticloadbalancing:ap-south-1:4883097432"
    weight      = 100
  }
}

resource "aws_globalaccelerator_endpoint_group" "secondary" {
  listener_arn = aws_globalaccelerator_listener.main.id

  endpoint_group_region    = local.regions.secondary
  traffic_dial_percentage = 50

  endpoint_configuration {
    endpoint_id = "arn:aws:elasticloadbalancing:ap-northeast-1:488309"
    weight      = 100
  }
} 16h rajendr... multi-region failover test
```

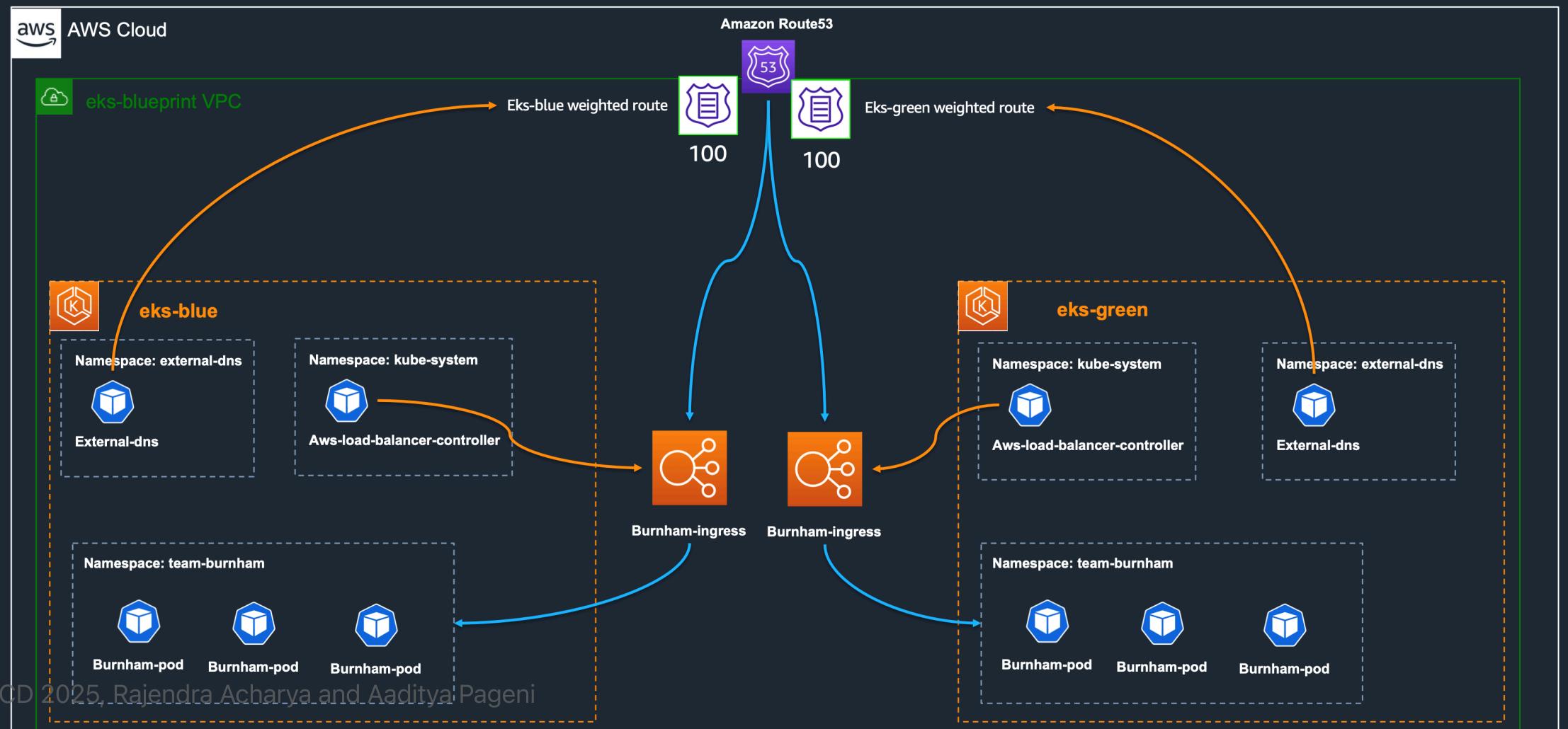
Blue Green Deployment ?

Why Blue-Green

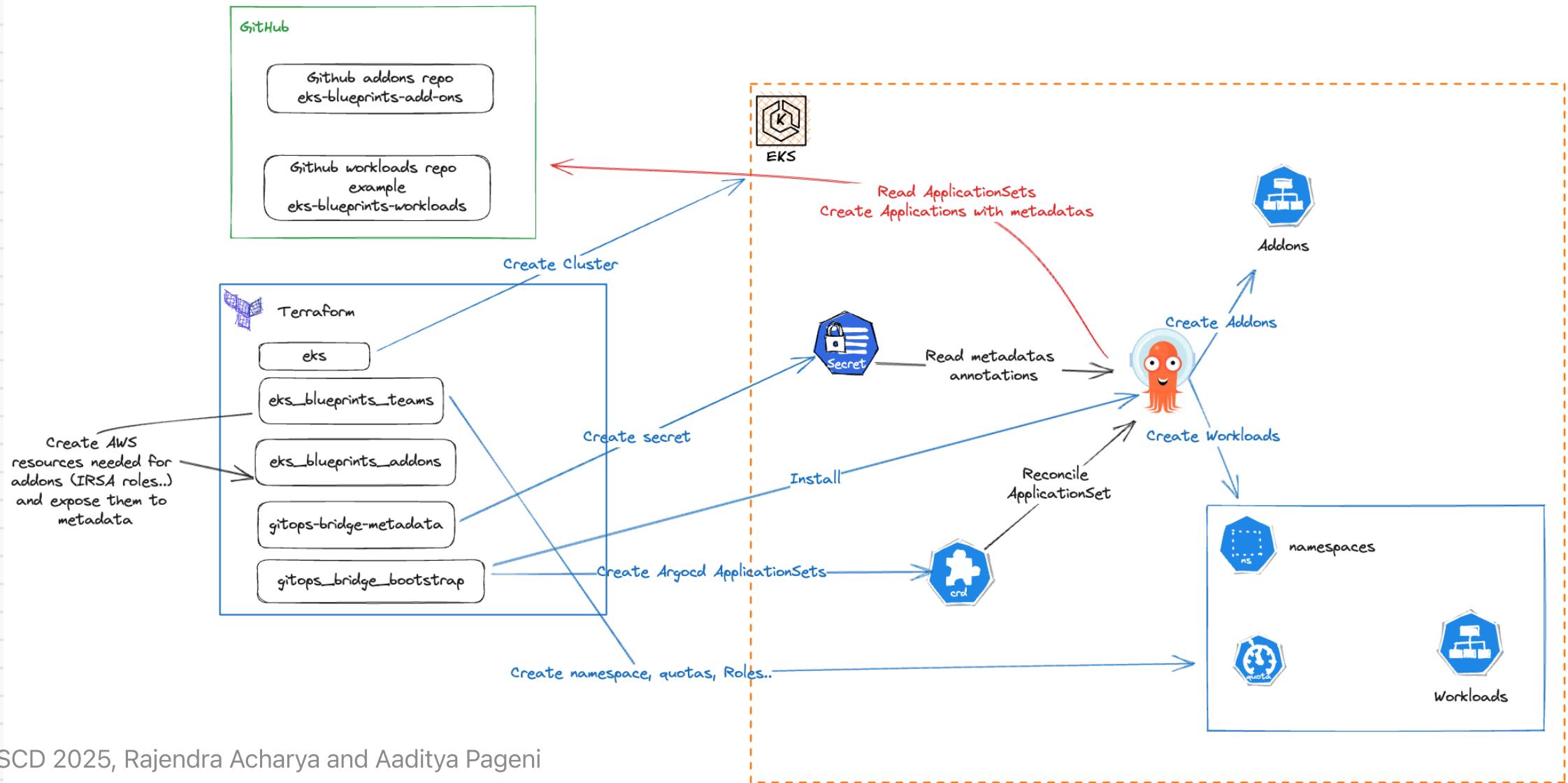
- Minimize downtime during upgrades
- Easy rollback to previous version
- Safer testing in production: Green can be fully tested before live traffic hits it.

EKS-Blue-Green deployment setup

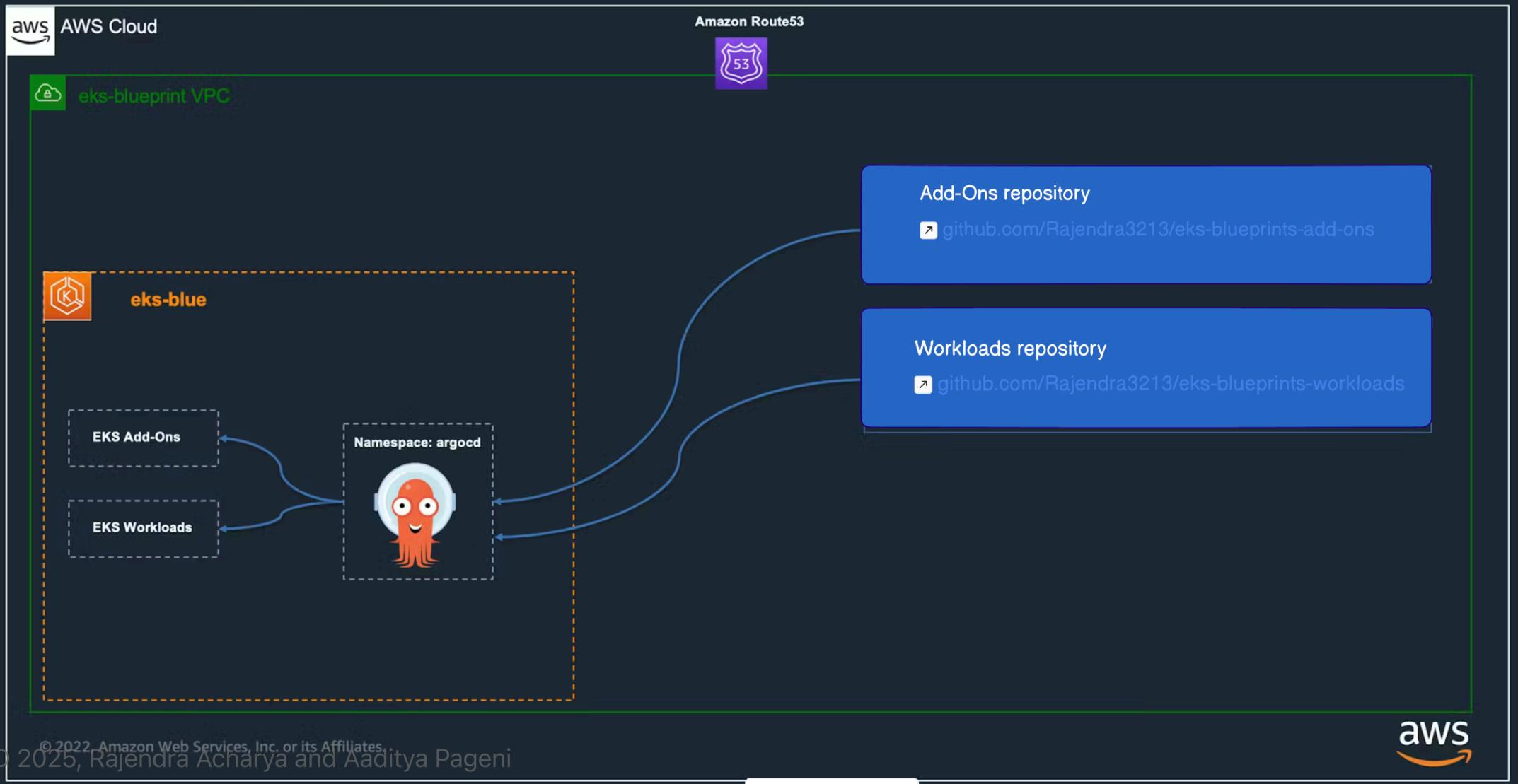
Route 53 Weighed record points to Blue and Green



Argo Rollout setup



ArgoCD synchronized Add-Ons and Workloads in Cluster

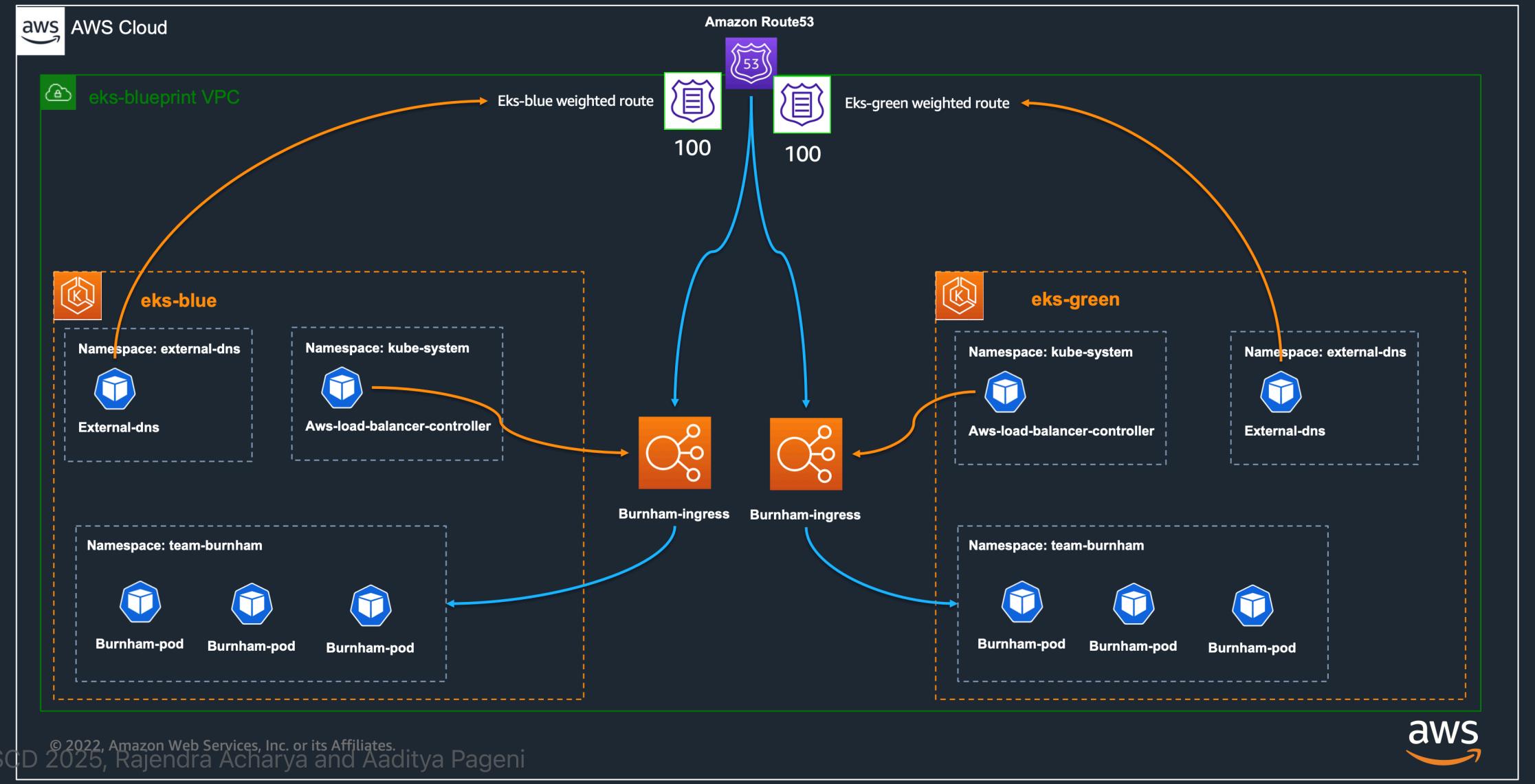


Tips

- Build and Deploy GREEN
- Run Health Checks + alarms
- Shift 10% traffic to GREEN
- Monitor logs + metrics
- Gradually increases traffic (50 % --> 100%)

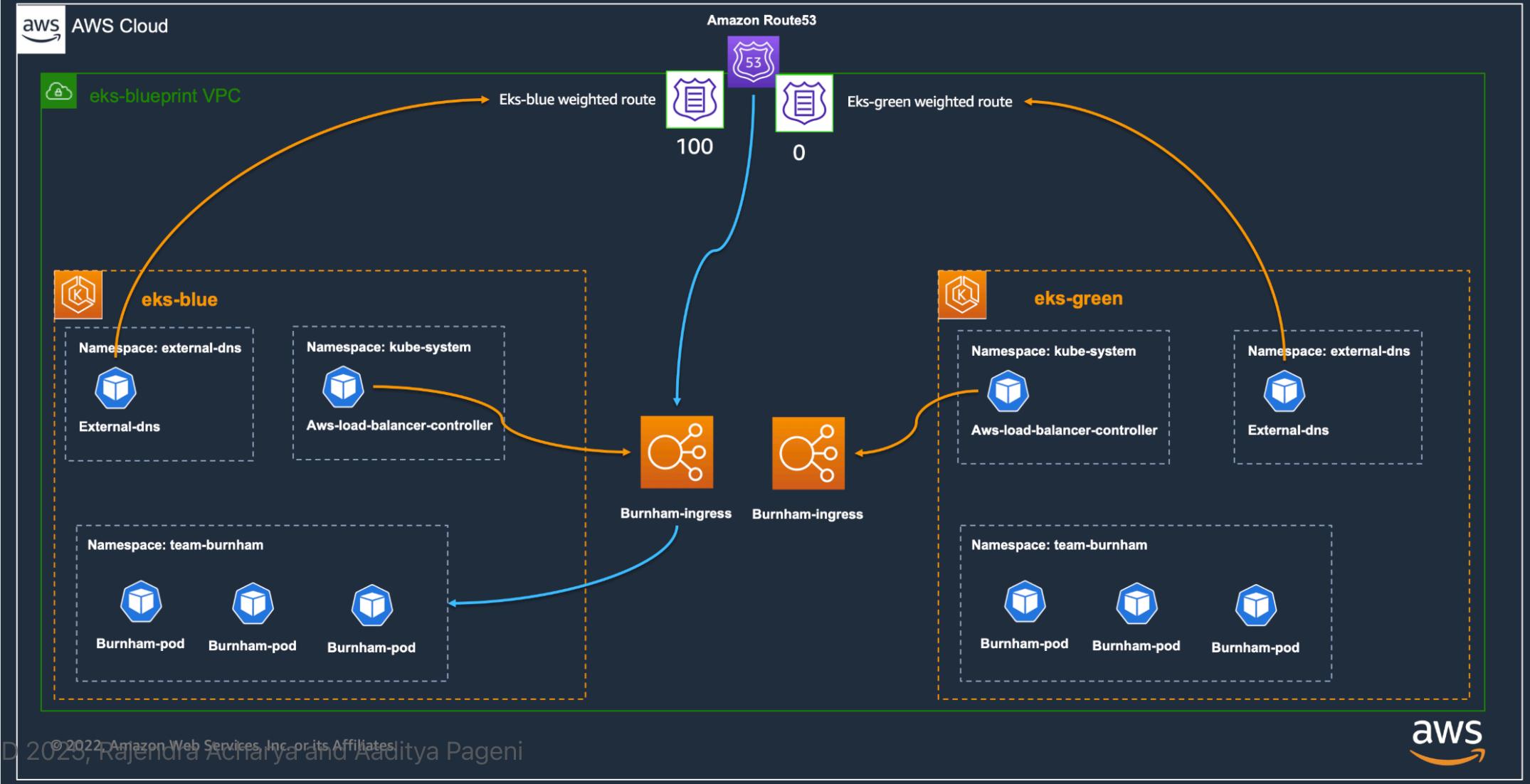
Demo

Route 53 Weighed record points to Blue and Green



Which correspond to :

Route 53 Weighed record points to Blue



Thanks

Questions?

Let's talk infrastructures



Aaditya



Rajendra