
AWS
re:Invent

NET301

Choosing the right load balancer for serverless applications

Corina Motoi

Senior Solutions Architect

AWS

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

Elastic Load Balancing

Load balancing in the serverless world

Architectural patterns



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Refresh containers knowledge



<<< Task networking in AWS Fargate (AWS Compute Blog)

Running Kubernetes on AWS Fargate – AWS Online Tech Talks (Youtube link) >>>

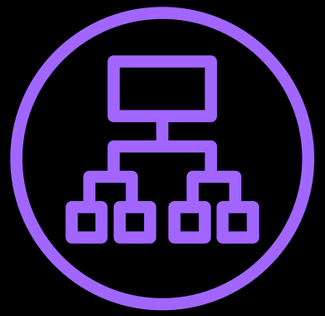


Elastic Load Balancing (ELB)

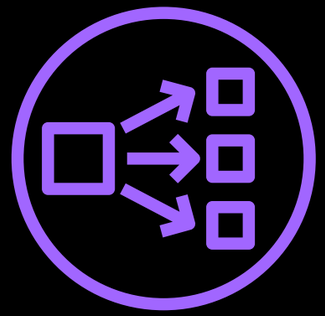
AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

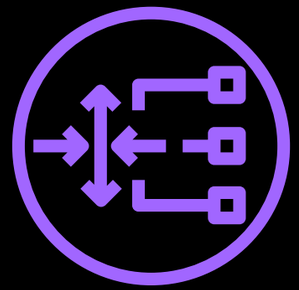
The family



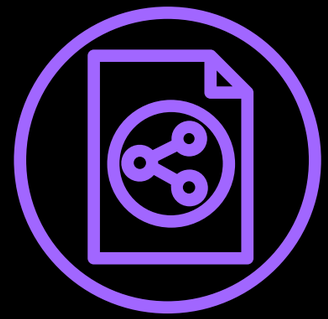
Application Load Balancer (ALB)



Network Load Balancer (NLB)

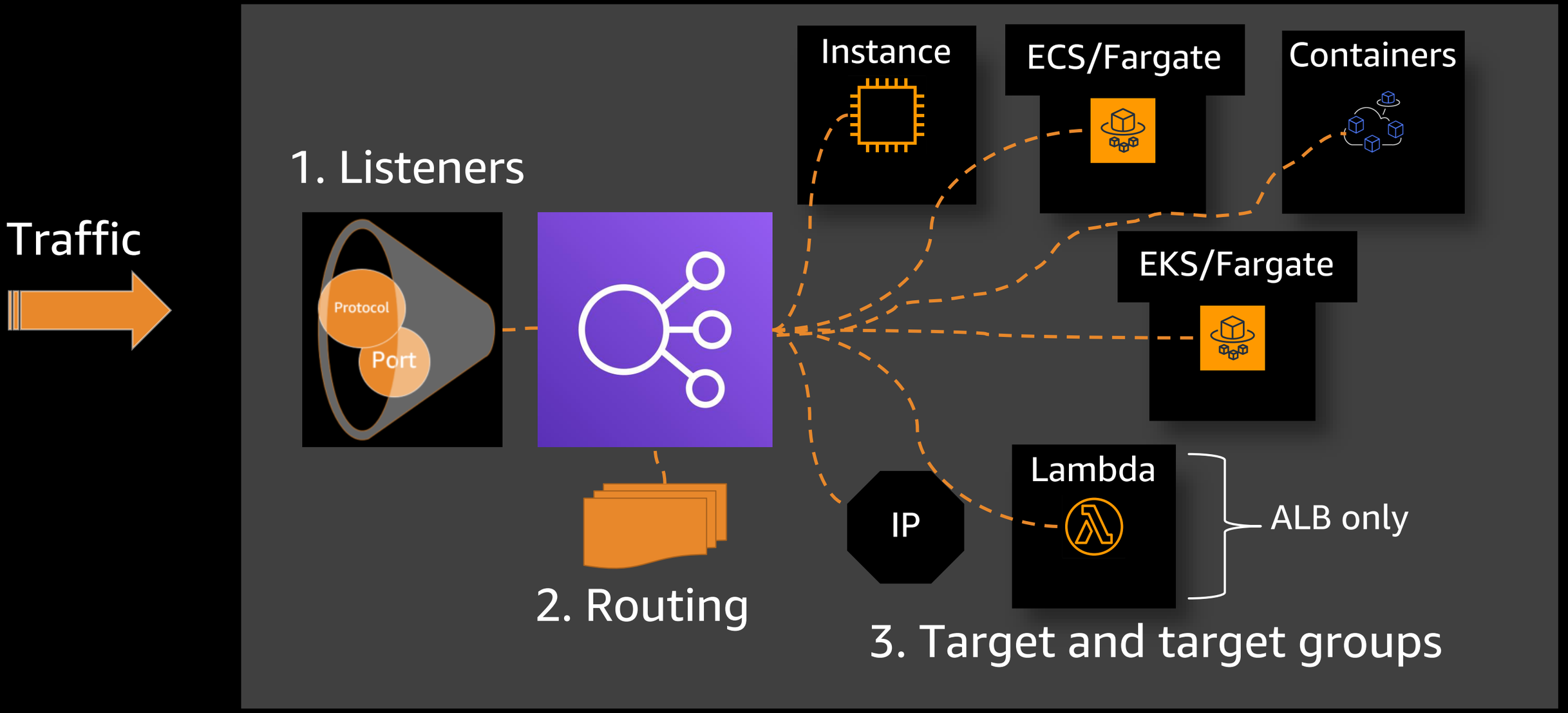


Gateway Load Balancer (GWLB)



Classic Load Balancer (CLB)

Elastic Load Balancing



Elastic Load Balancing

	ALB	NLB	GWLB	CLB
Listener protocols	HTTP, HTTPS, gRPC	TCP, UDP, TLS	IP	TCP, SSL/TLS, HTTP, HTTPS
Static IP (Use your own Elastic IP)		✓		
Terminates flow/proxy behavior	✓	✓		✓
Routing rules based on the content of the request	✓			
Route TCP connections and UDP flows		✓		
AWS PrivateLink		✓	✓	
EC2 instances	✓	✓	✓	✓
IP addresses	✓	✓	✓	
AWS Lambda	✓			
Containers (AWS Fargate)	✓	✓		

Listeners and connectivity

Request routing

Target and target groups

ALB vs. NLB vs. CLB

RE:INVENT 2019 – YOUTUBE – GET THE MOST FROM ELASTIC LOAD BALANCING FOR DIFFERENT WORKLOADS (NET407-R2)



Load balancing in the serverless world

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Serverless services

SMALL PART OF THEM



AWS Lambda



AWS Fargate



Amazon API Gateway

Lambda as a target for ALB

WHY?

Benefits / use cases

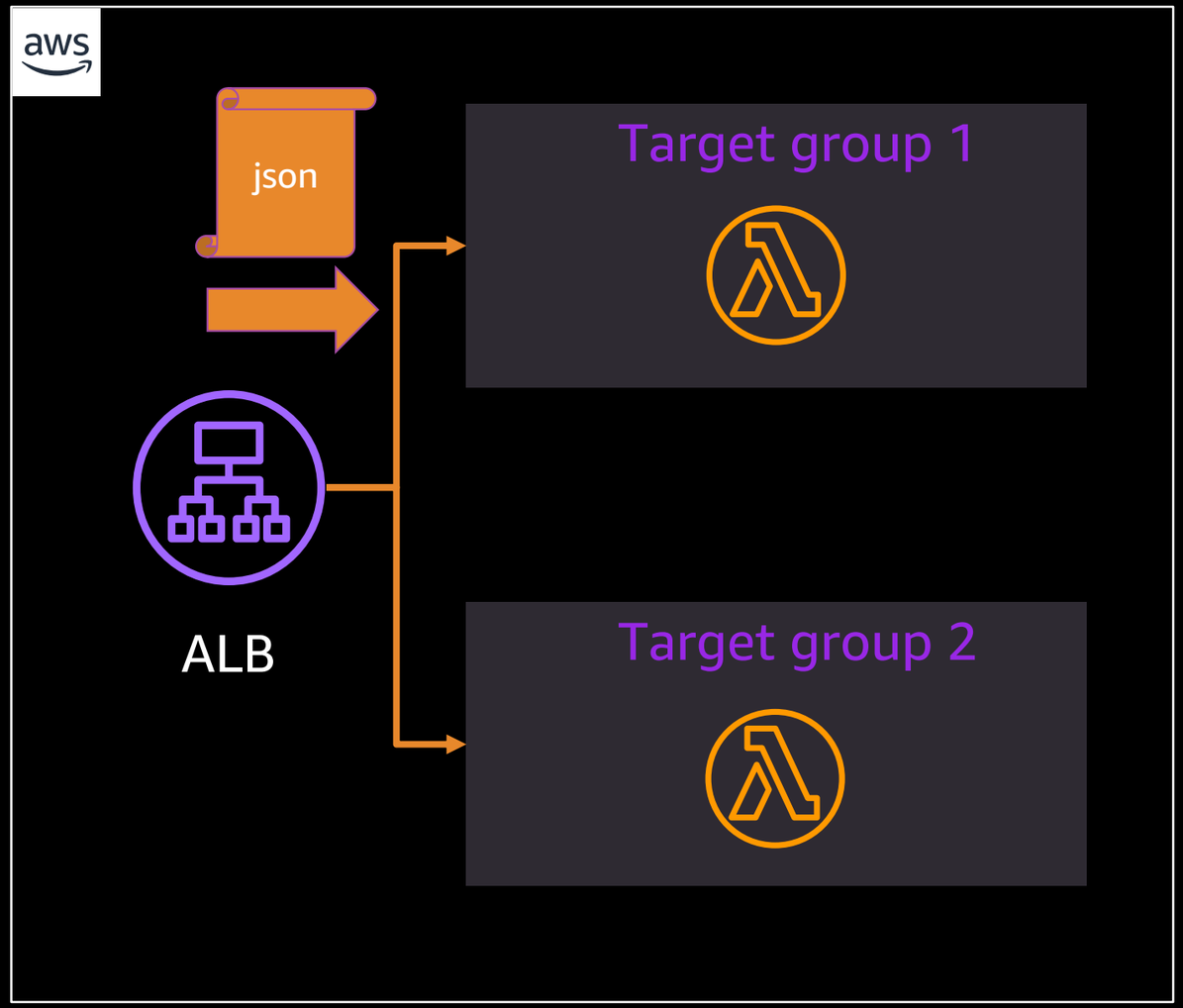
- Host applications with a mix of compute services, such as Amazon EC2 and Lambda
- Build serverless websites
- Make use of the ALB content-based routing rules and weighted target groups to route the traffic to multiple Lambda functions
- Create custom targets for ALB, like object upload / download to Amazon S3

Lambda as a target for ALB

HOW IT WORKS: THE REQUEST

- Supported by both HTTP / HTTPS
- ALB > Lambda in {json}
- Additional headers added by ALB to the request

```
{
  ....
      "x-amzn-trace-id": "Root=1-5bdb40ca-
556d8b0c50dc66f0511bf520",
      "x-forwarded-for": "72.21.198.66",
      "x-forwarded-port": "443",
      "x-forwarded-proto": "https"
   },
   "isBase64Encoded": false,
   "body": "request_body"
}
```

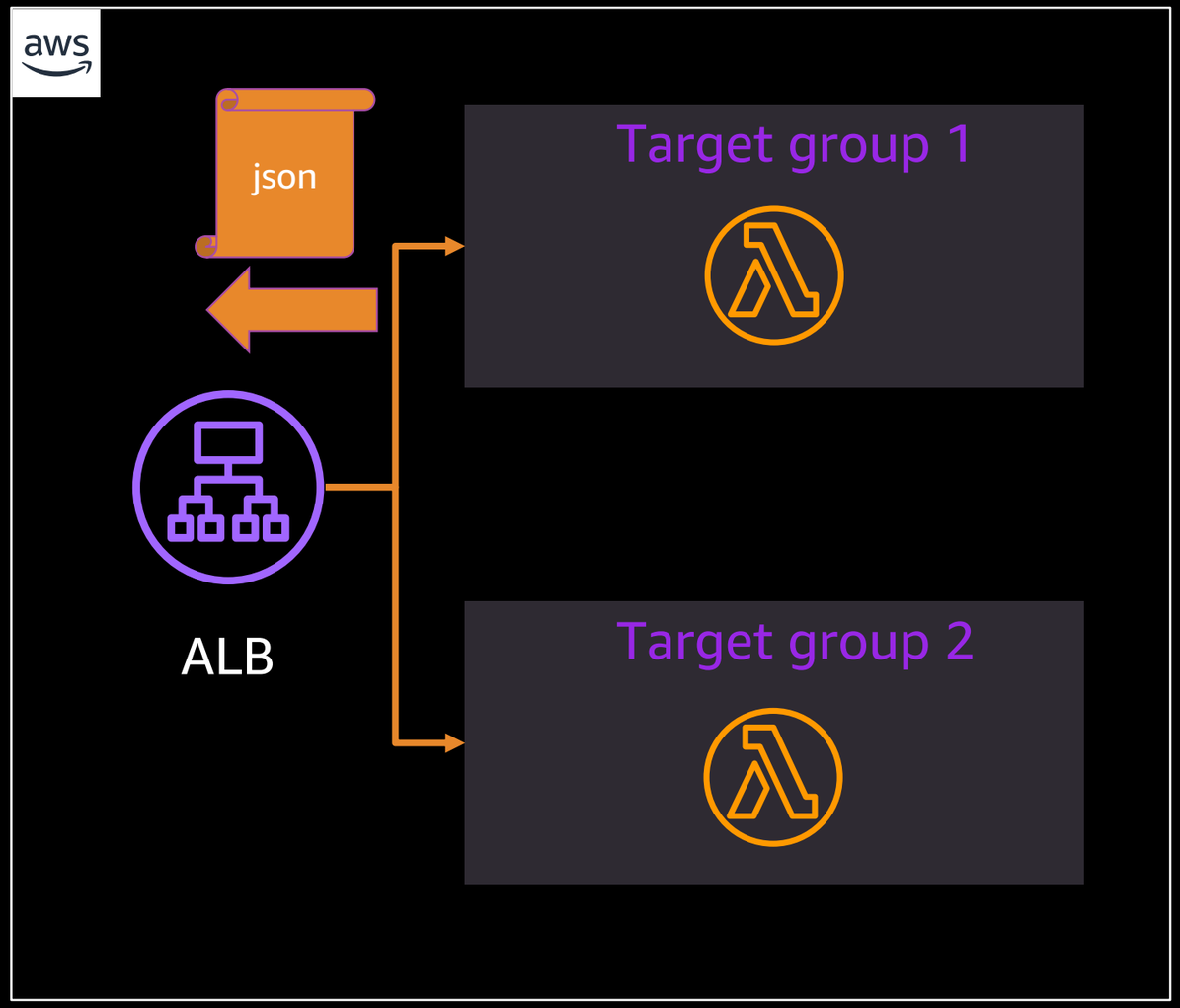


Lambda as a target for ALB

HOW IT WORKS: THE RESPONSE

Lambda > ALB in {json} format

```
{
   "isBase64Encoded": true,
   "statusCode": 200,
   "statusDescription": "200 OK",
   "headers": {
      "Set-cookie": "cookies",
      "Content-Type": "application/json"
   },
   "body": "Hello from Lambda"
}
```



Lambda as a target for ALB

WRAP-UP

Benefits / use cases

- Host applications with a mix of compute services, such as EC2 and Lambda
- Build serverless websites
- Make use of the ALB content-based routing rules and weighted target groups to route the traffic to multiple Lambda functions
- Create custom targets for ALB like object upload / download to S3

Good to know

- One Lambda function per target group
- The Lambda function and ALB must be in the same AWS account and same region
- WebSockets are not supported
- ALB health checks for Lambda are supported and can be enabled via console or AWS CLI

Fargate (ECS) as a target for ELB

WHY?

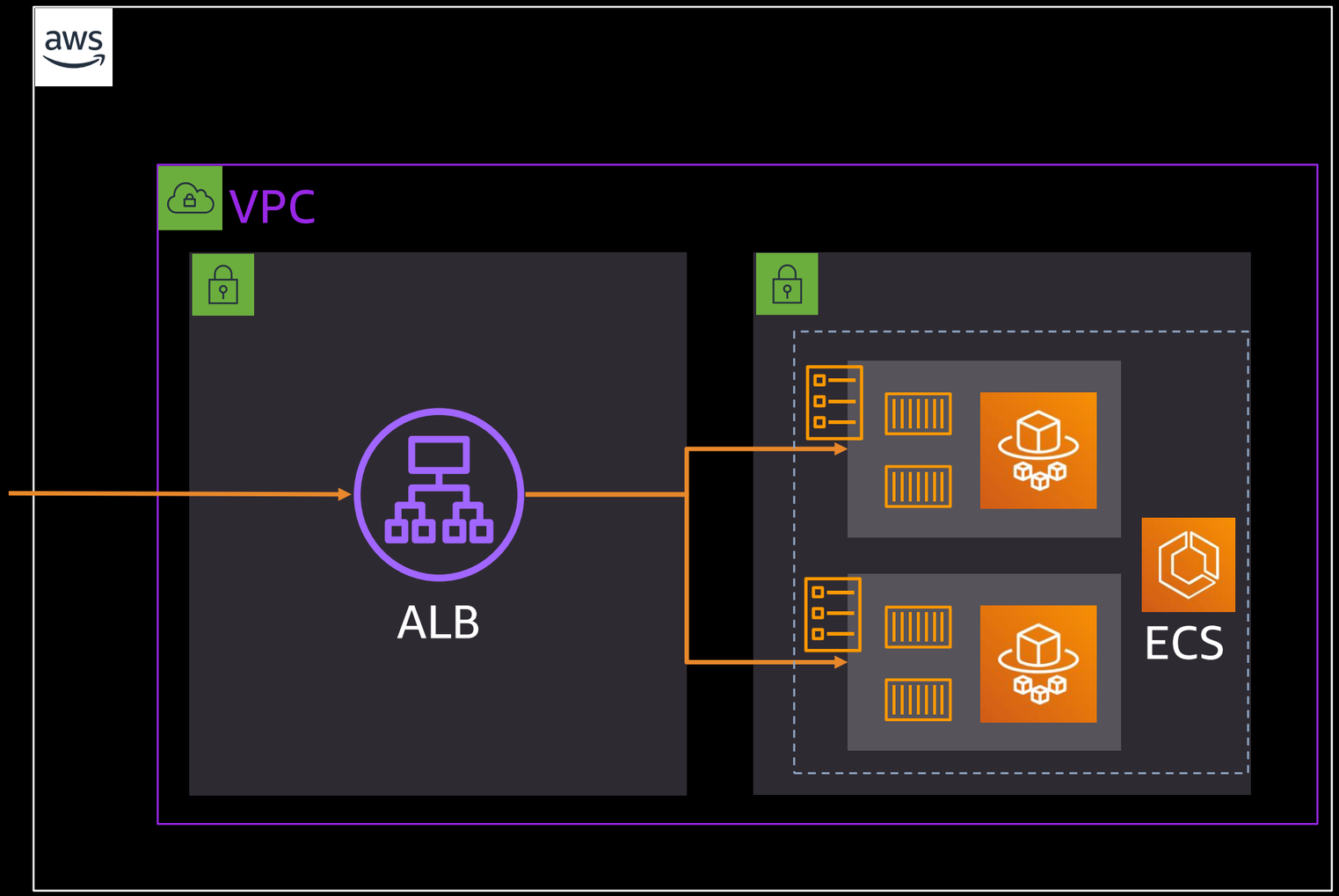
Benefits / use cases

- Host applications with a mix of compute services, such as EC2, Lambda, and containers
- Build microservices applications with spikey, unpredictable traffic patterns
- Run TCP / UDP based applications using NLB
- Multiple services using a single listener on a single ALB

Fargate (ECS) as a target for ELB

HOW IT WORKS

- Supported by ALB and NLB
- awsvpc network mode with **IP address** as the target type
- ALB and the ECS cluster can be in either in private or public subnets



Fargate (ECS) as a target for ELB

WRAP-UP

Benefits / use cases

- Host applications with a mix of compute services, such as EC2, Lambda and containers
- Build microservices applications with spikey, unpredictable traffic patterns
- Run TCP / UDP-based applications using NLB
- Multiple services using a single listener on a single ALB

Good to know

- ALB allows dynamic host port mapping for containers
- A service supports up to five target groups, with mixed compute (Fargate, EC2)
- Recommended to have both the ELB and container health checks configured
  - Fargate version 1.1.0 or greater for container health checks

Fargate (EKS) as a target for ELB

WHY?

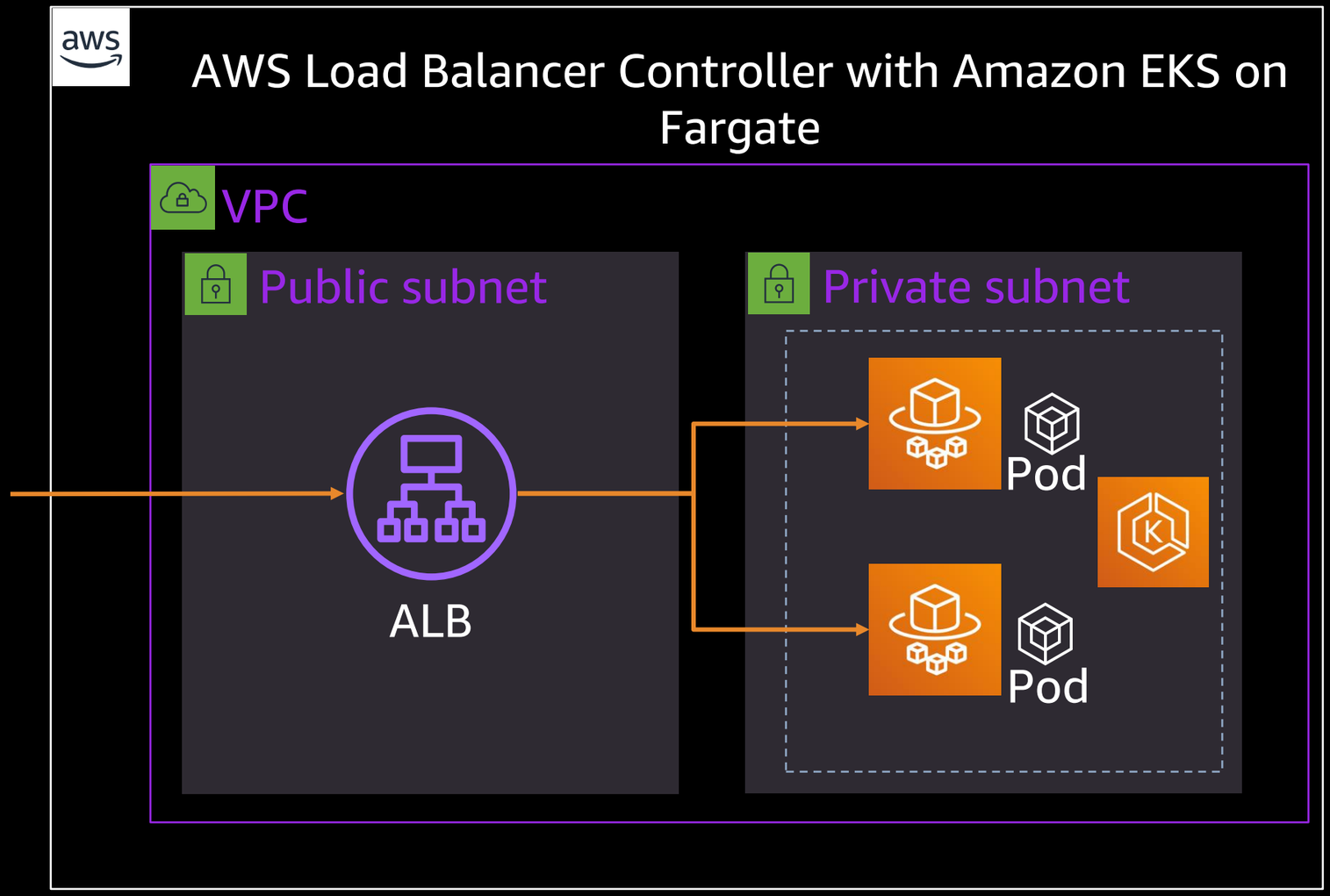
Benefits / use cases

- Simpler to run Kubernetes applications
- AWS manages the lifecycle of the pod and spins up a Fargate worker node to run the pods
- Use AWS Load Balancer controller as an ingress controller for EKS

Fargate (EKS) as a target for ELB

HOW IT WORKS

- Supported by ALB & NLB
- AWS Load Balancer Controller supports IP address targeting mode
- Ingress annotations: service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip“
- Pods can run only in private subnets



Fargate (EKS) as a target for ELB

WRAP-UP

Benefits / use cases

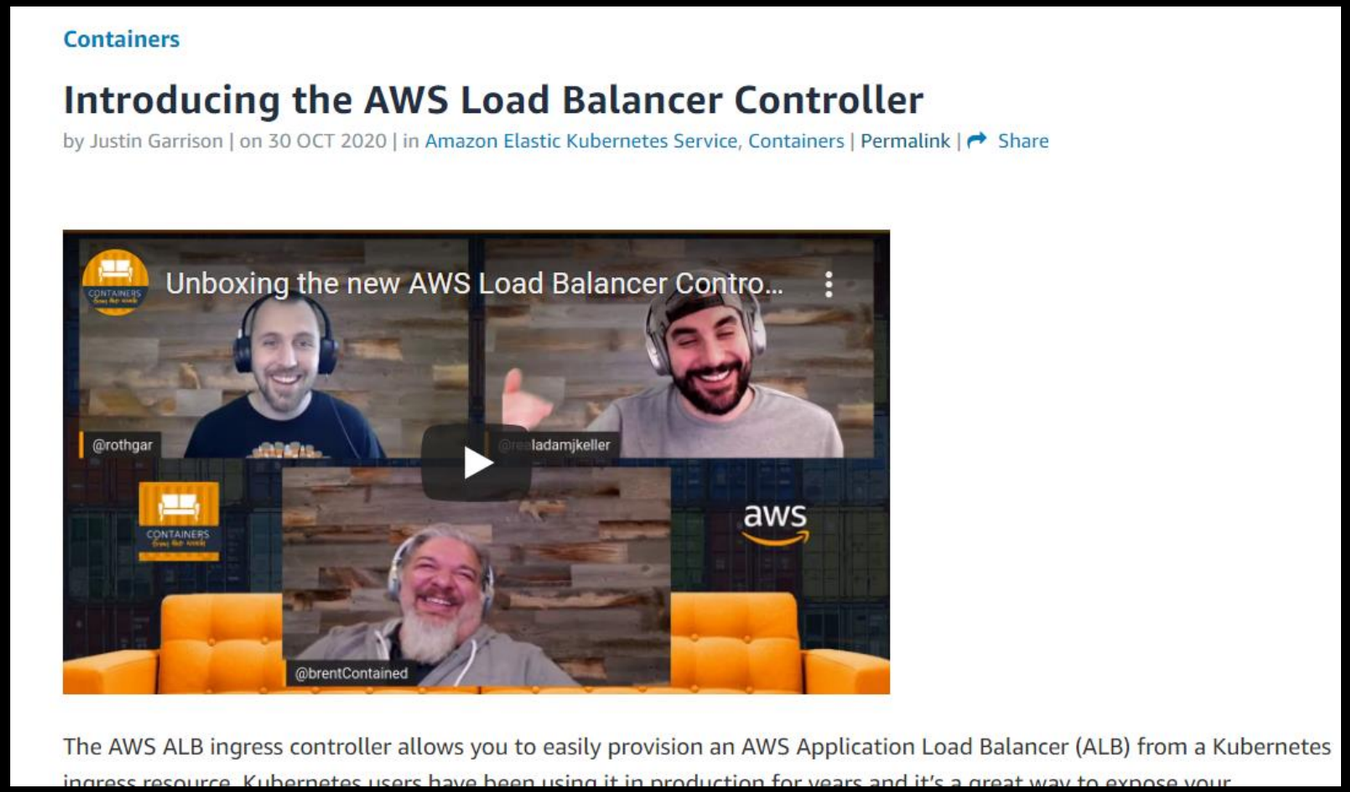
- Simpler to run Kubernetes applications
- AWS manages the lifecycle of the pod and spins up a Fargate worker node to run the pods
- Use AWS Load Balancer controller as an ingress controller for EKS

Good to know

- AWS Load Balancer Controller share ALBs with multiple Kubernetes ingress rules
- Support for custom resource (CR) called TargetGroupBinding to expose your pods using an existing target group
- Maximum of 4 vCPU and 30Gb memory per pod
- Support for fully private clusters

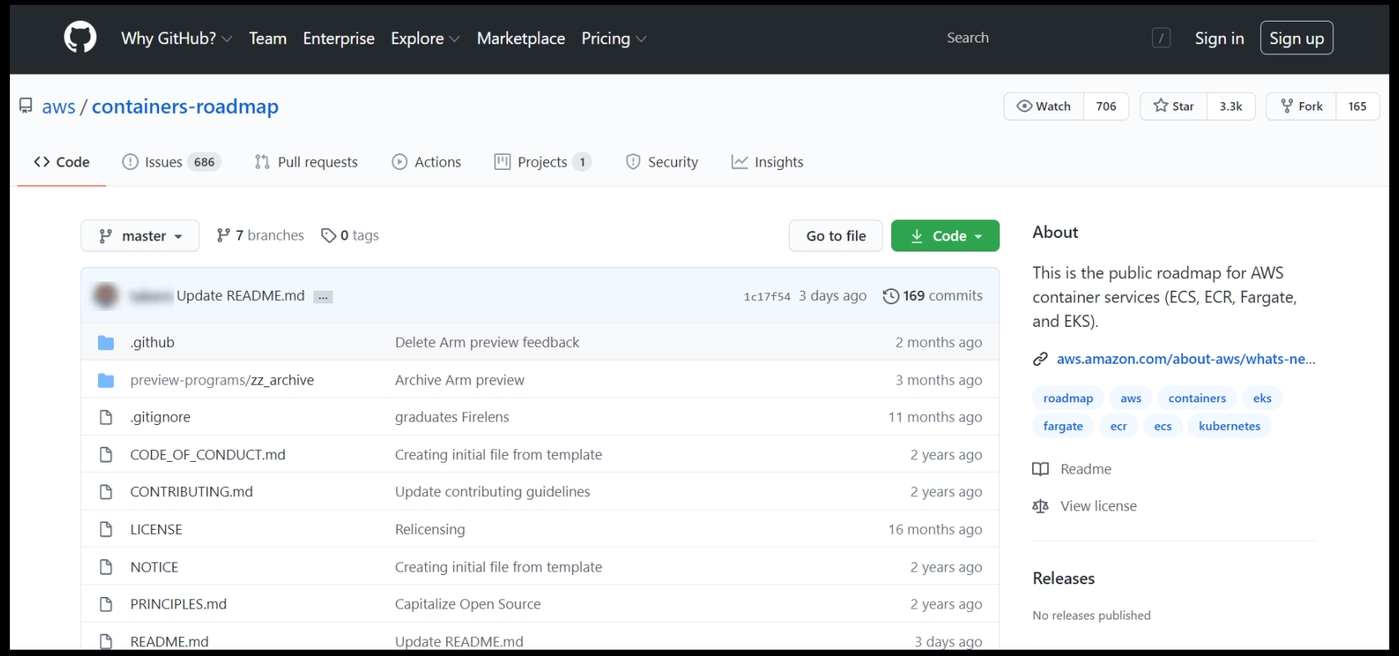
Introducing the AWS Load Balancer Controller

BLOG POST



Containers roadmap

GITHUB



Amazon API Gateway vs. ALB

APIGW	ALB
• Serverless API management platform	• AWS managed load-balancing service
• Path-based routing	• Rule-based routing, mixed
• Built-in authentication and authorization, and integration with Amazon Cognito	• Integration with Amazon Cognito for authentication
• Request and response size 6MB	• Request and response size 1MB
• Throttling	• ALB behind NLB provides a static IP for the HTTP endpoint

Amazon API Gateway vs. ALB

APIGW



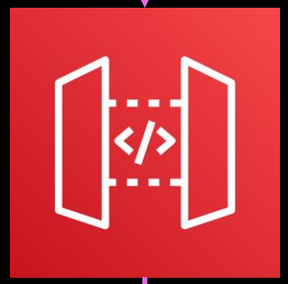
X HTTP APIs

✓ REST APIs

AWS WAF

10,000 RPS (requests per second)

RESTful, WebSocket APIs



Public endpoints such as S3, Amazon SQS, etc., including external endpoints

ALB

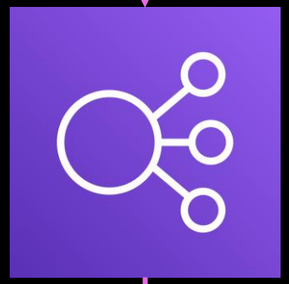


✓ HTTP APIs

AWS WAF

Scales automatically

HTTP, HTTPS APIs



Mix of EC2, containers, on-premises and Lambda

Architectural patterns

Serverless microservices with ALB

Serverless microservices with NLB

Exposing private microservices using APIGW

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Architectural patterns

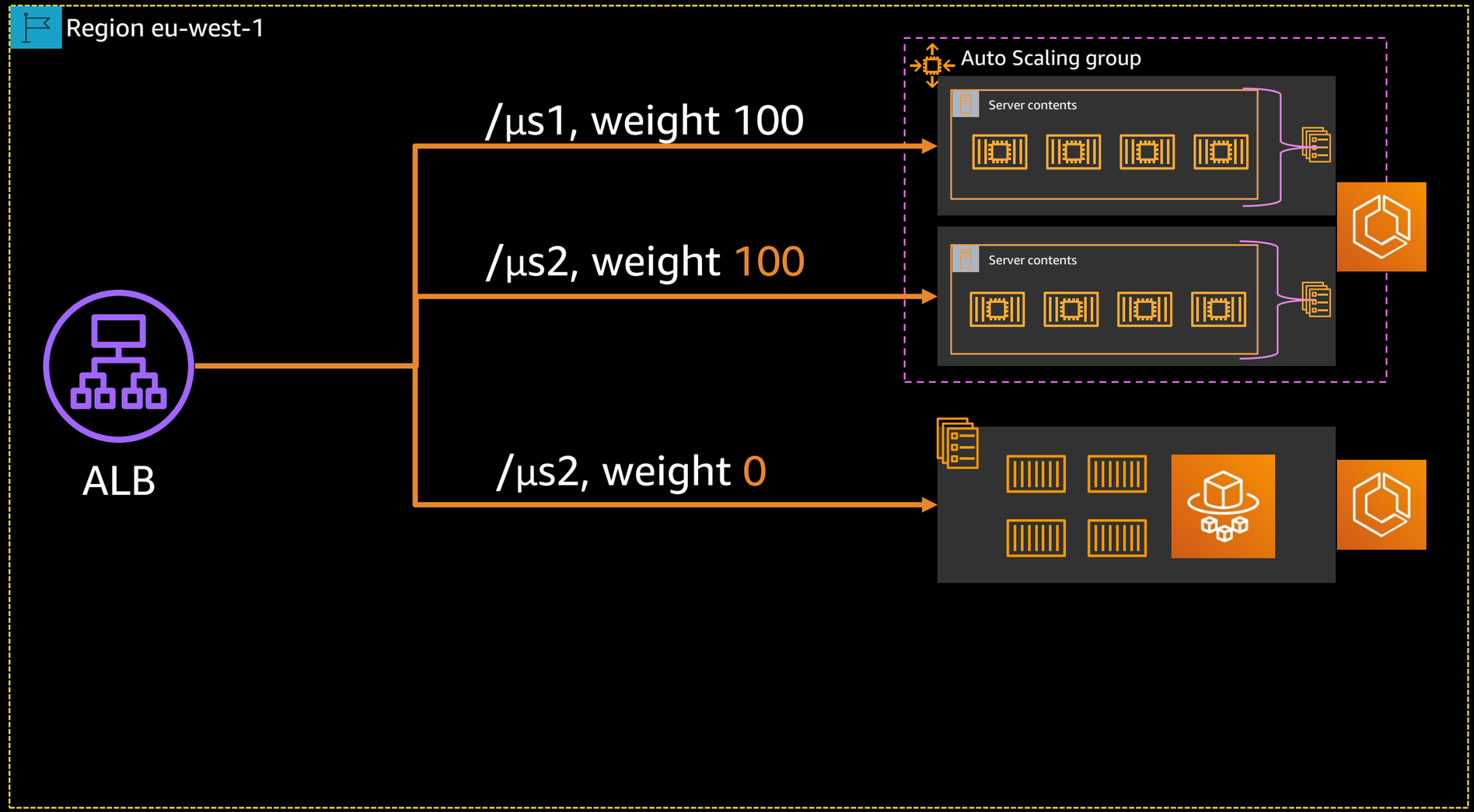
Serverless microservices with ALB

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

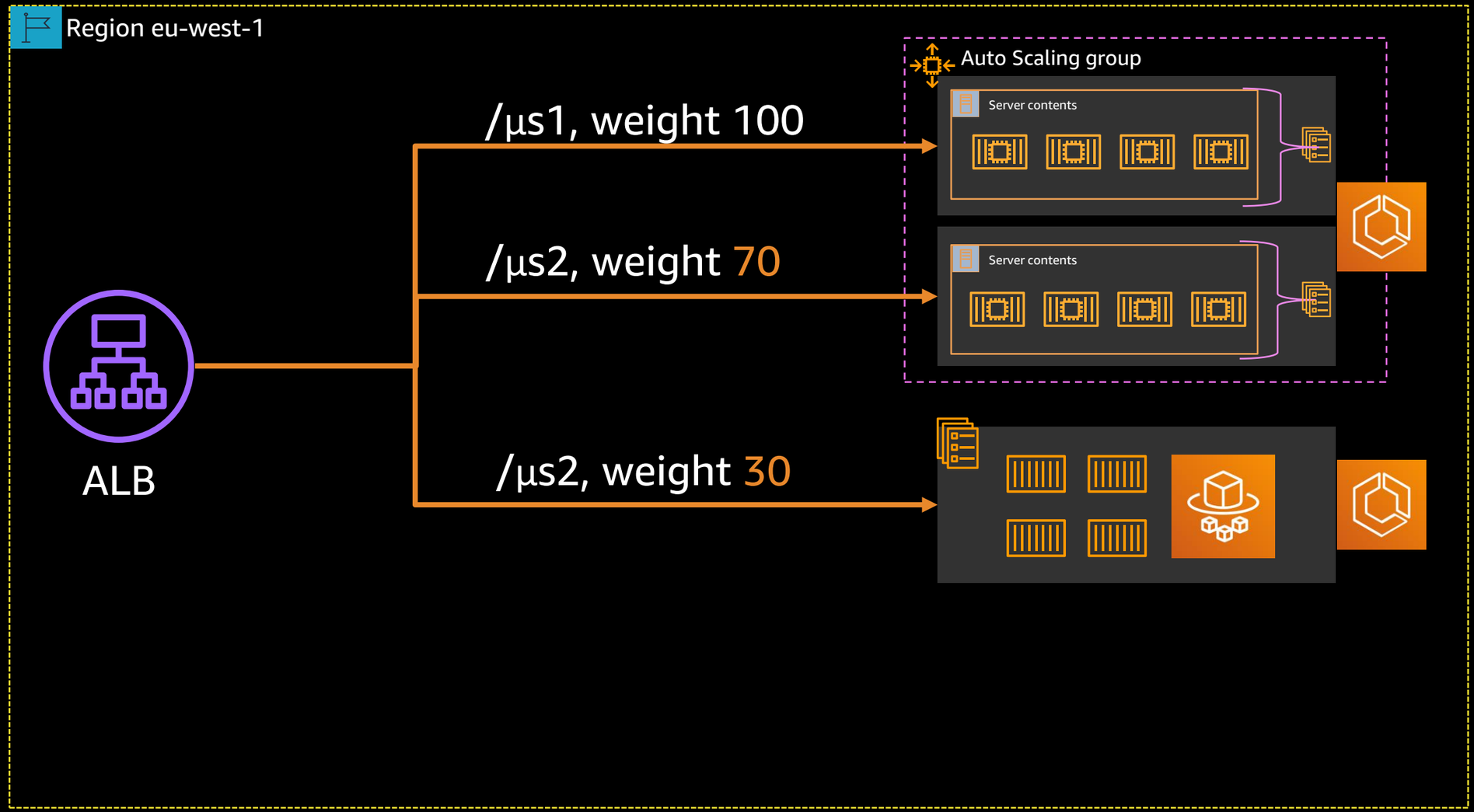
Introducing serverless microservices

- HTTP / HTTPS APIs
- Make use of ALB weighted target groups



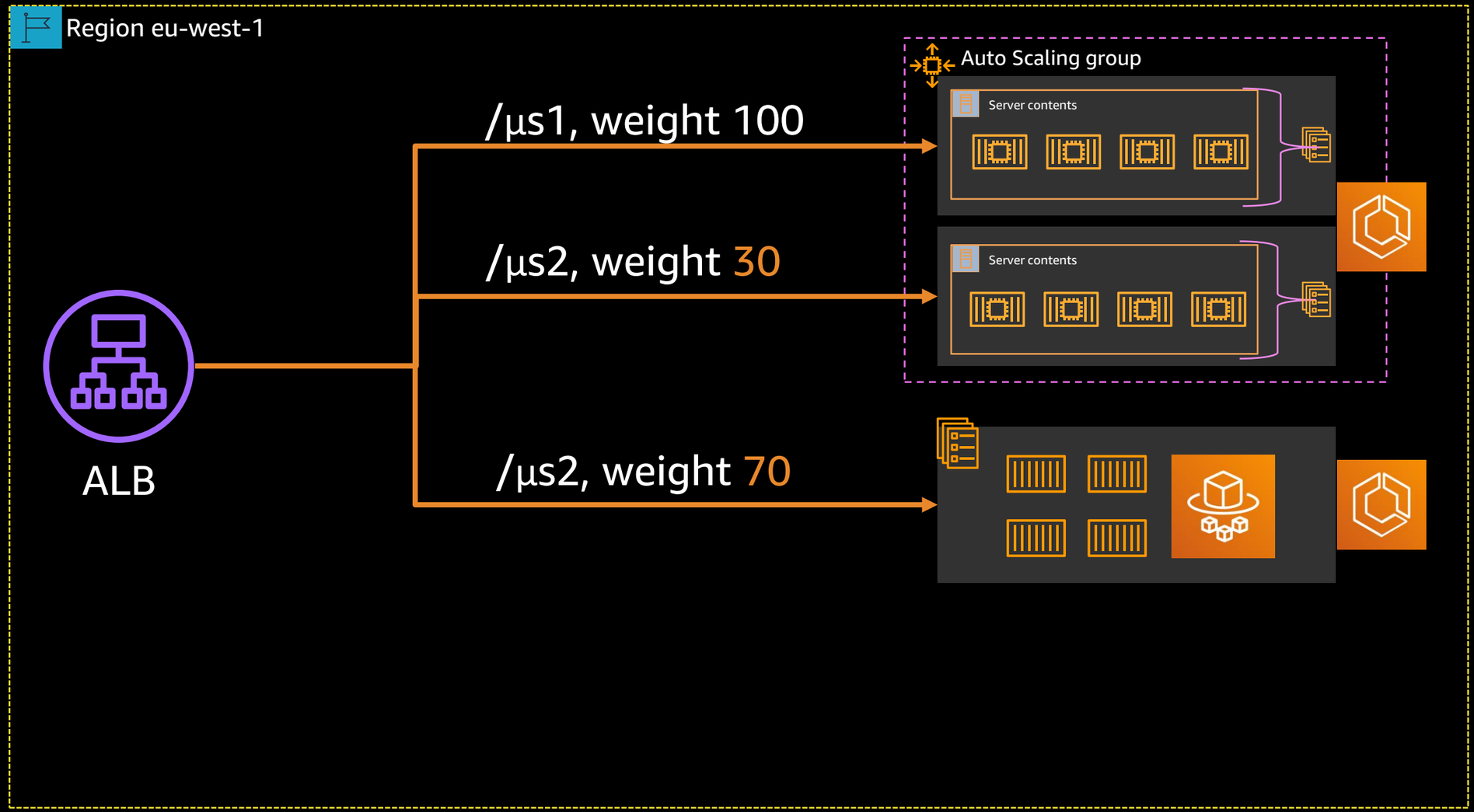
Introducing serverless microservices

- HTTP / HTTPS APIs
- Make use of ALB weighted target groups



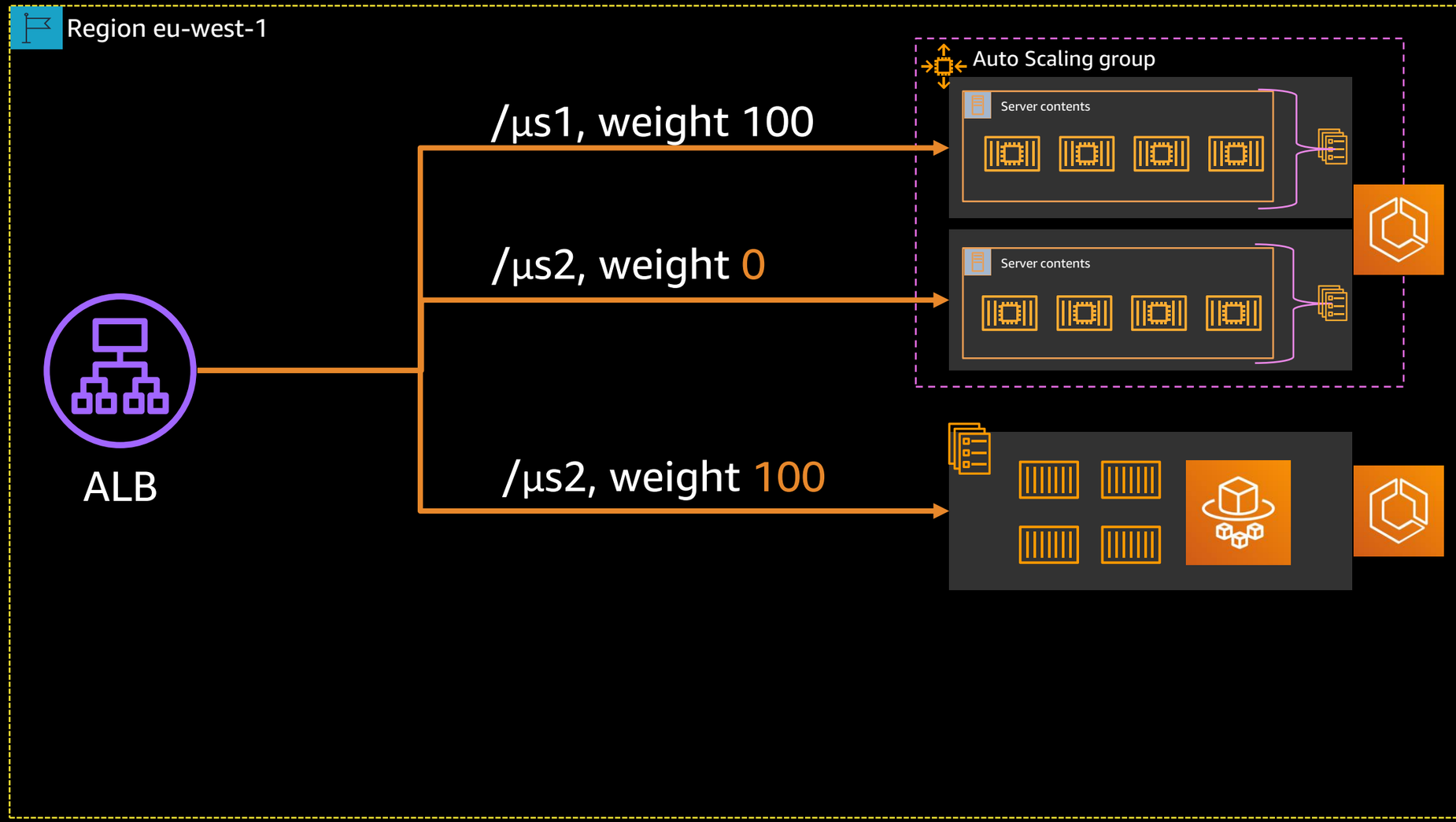
Introducing serverless microservices

- HTTP / HTTPS APIs
- Make use of ALB weighted target groups



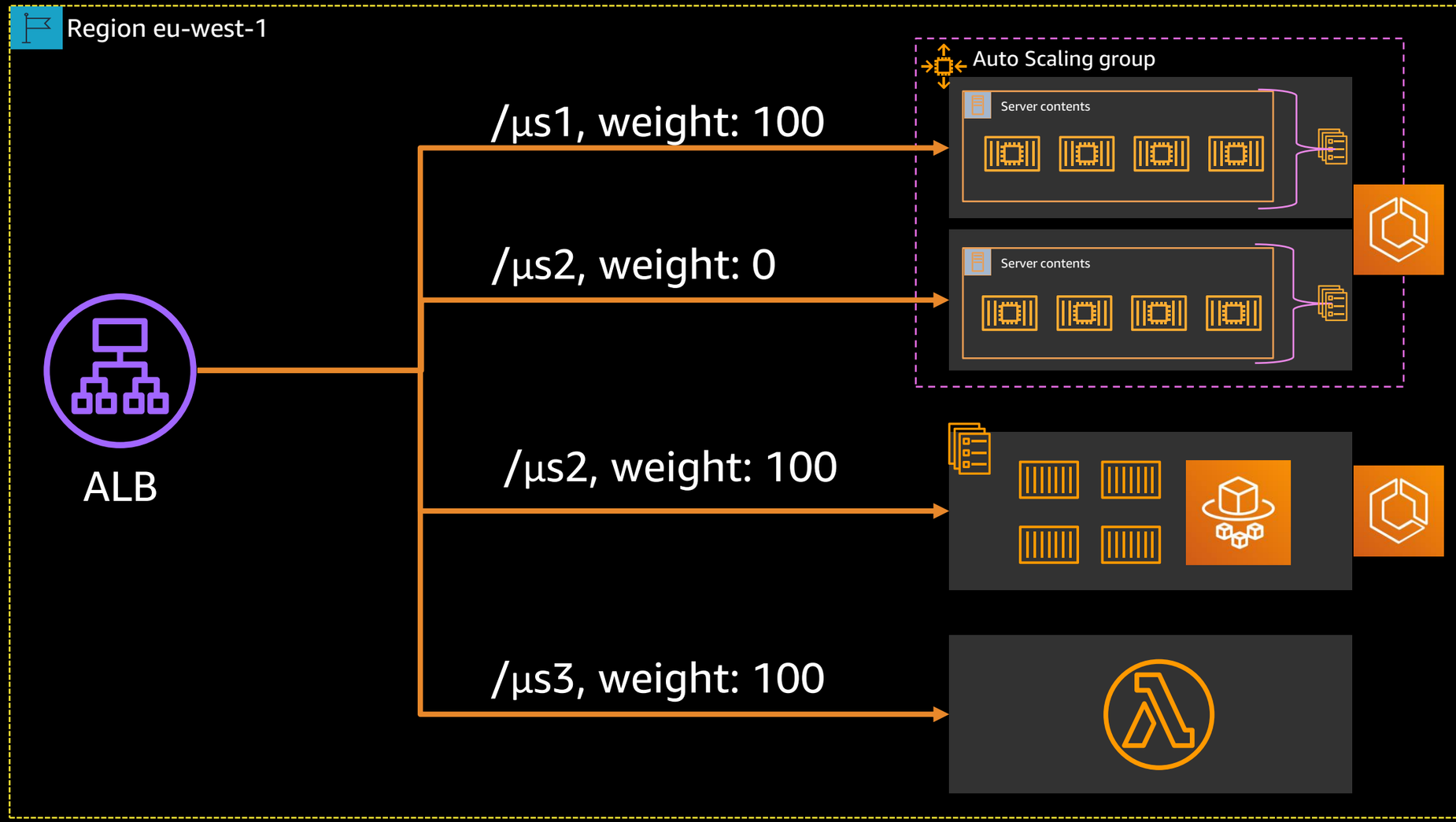
Running serverless microservices

- HTTP / HTTPS APIs
- Make use of ALB weighted target groups
- Containers operational management is moved to Fargate



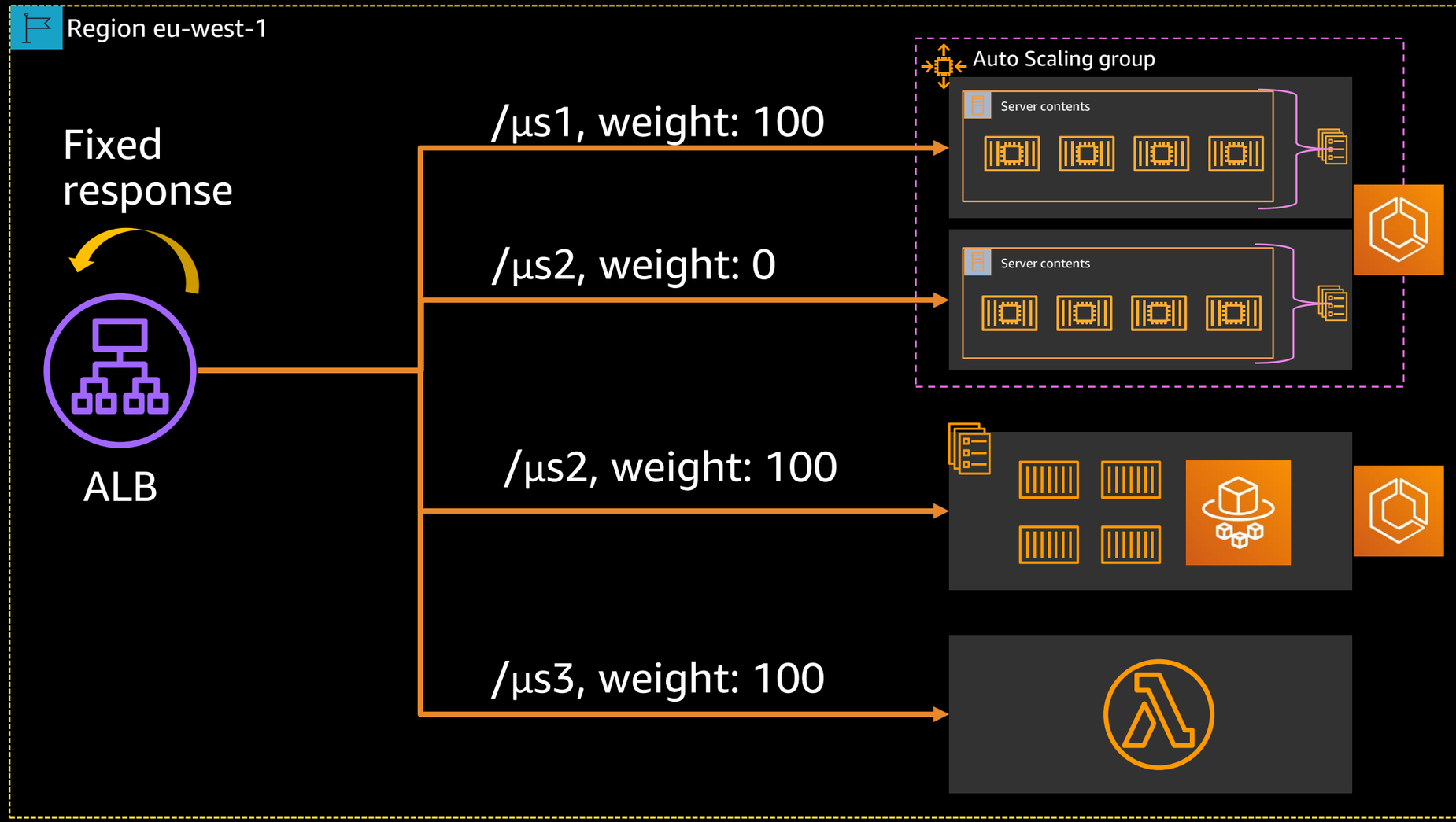
Running more serverless microservices

- HTTP / HTTPS APIs
- Make use of ALB weighted target groups
- Containers operational management is moved to Fargate
- Develop new serverless microservices



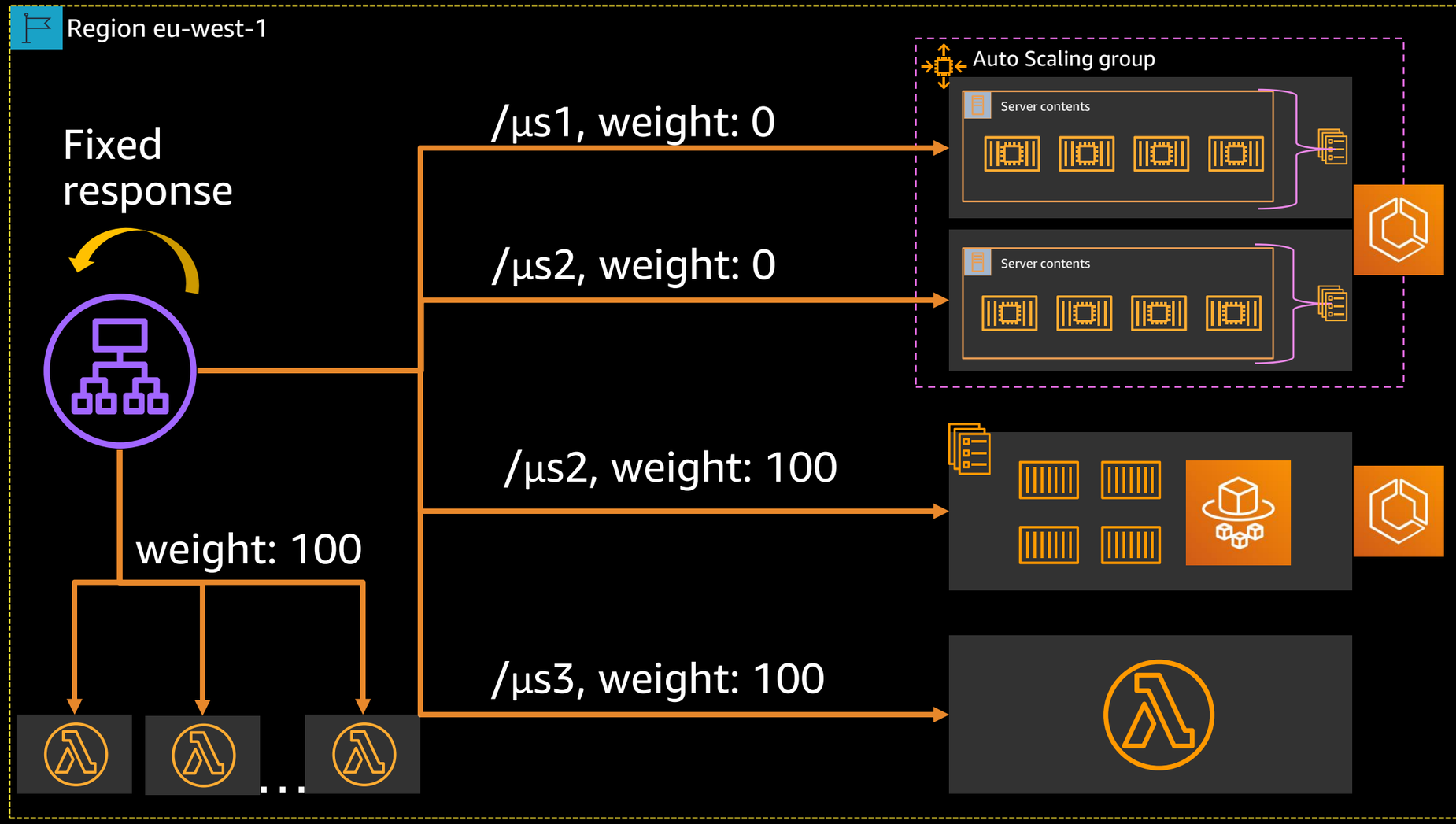
Running more serverless microservices

- HTTP / HTTPS APIs
- Make use of ALB weighted target groups
- Containers operational management is moved to Fargate
- Develop new serverless microservices
- ALB fixed response

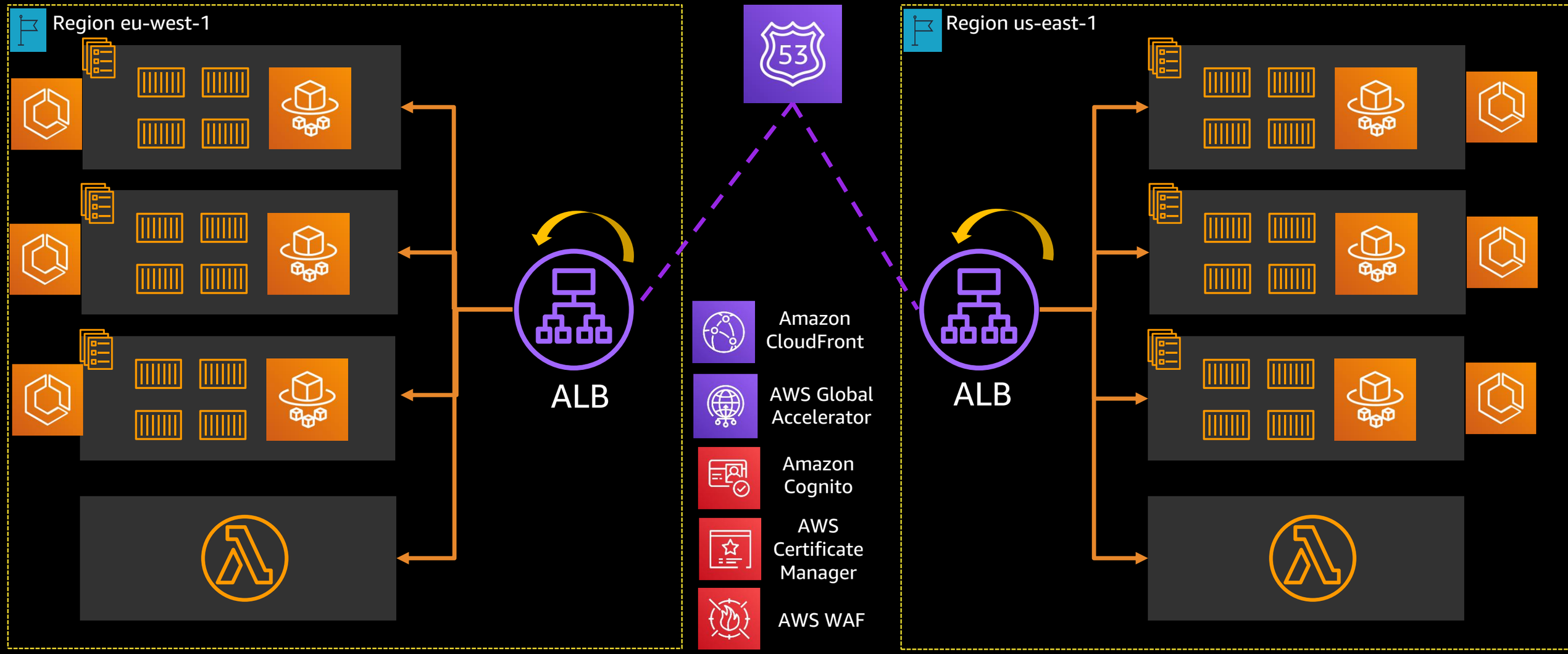


Running more and more serverless microservices

- HTTP / HTTPS APIs
- Make use of ALB weighted target groups
- Containers operational management is moved to Fargate
- Develop new serverless microservices
- ALB fixed response



Multi-region serverless μservices architecture



Architectural patterns

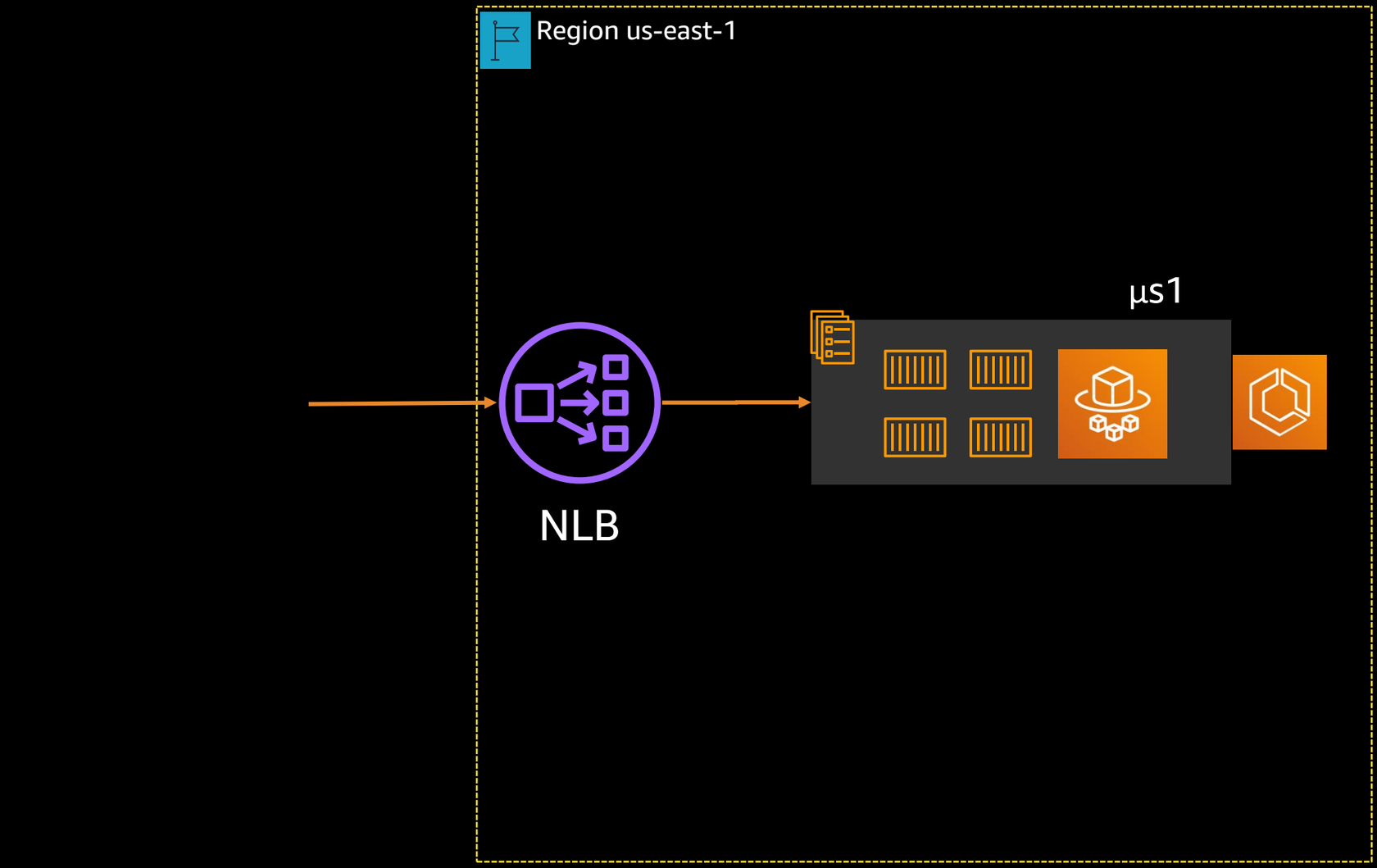
Serverless microservices with NLB

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

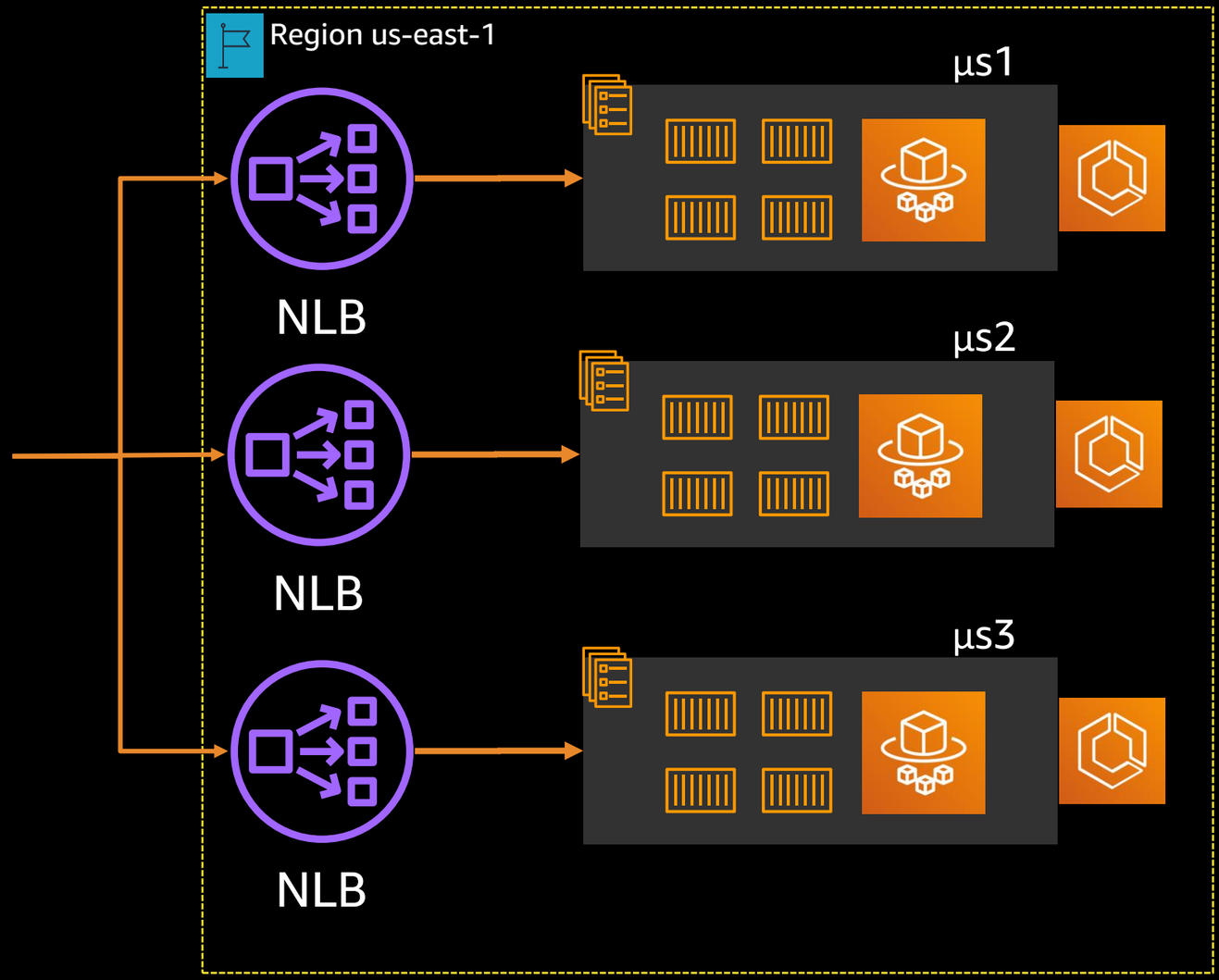
Serverless microservices with NLB

- TCP / UDP support
- Use cases
  - Non HTTP / HTTPS APIs
  - High throughput
  - Ultra low latency
  - Static IP needed
- NLB > microservice (1:1)



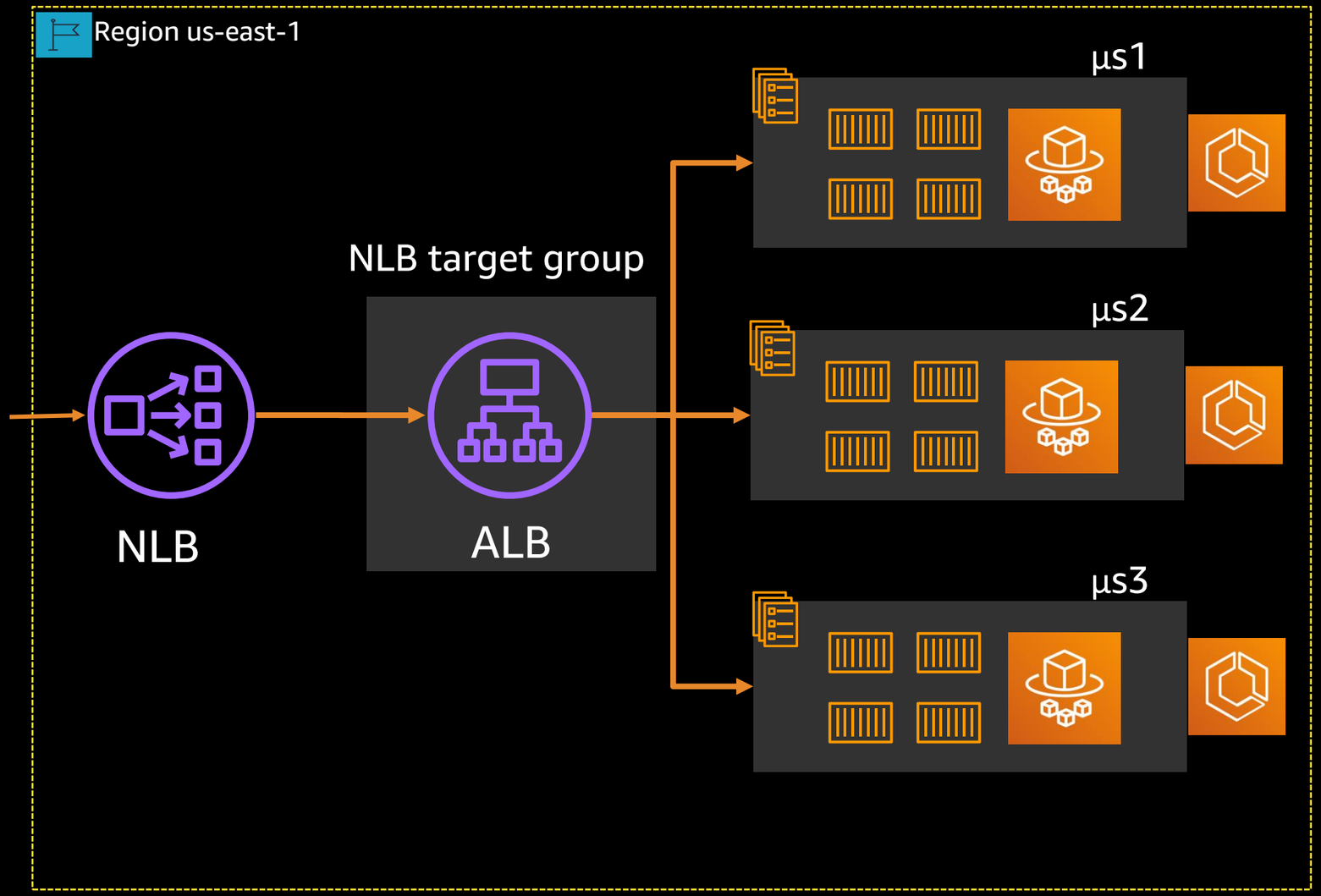
Serverless microservices with NLB

- TCP / UDP support
- Use cases
  - Non HTTP / HTTPS APIs
  - High throughput
  - Ultra low latency
  - Static IP needed
- NLB > microservice (1:1)



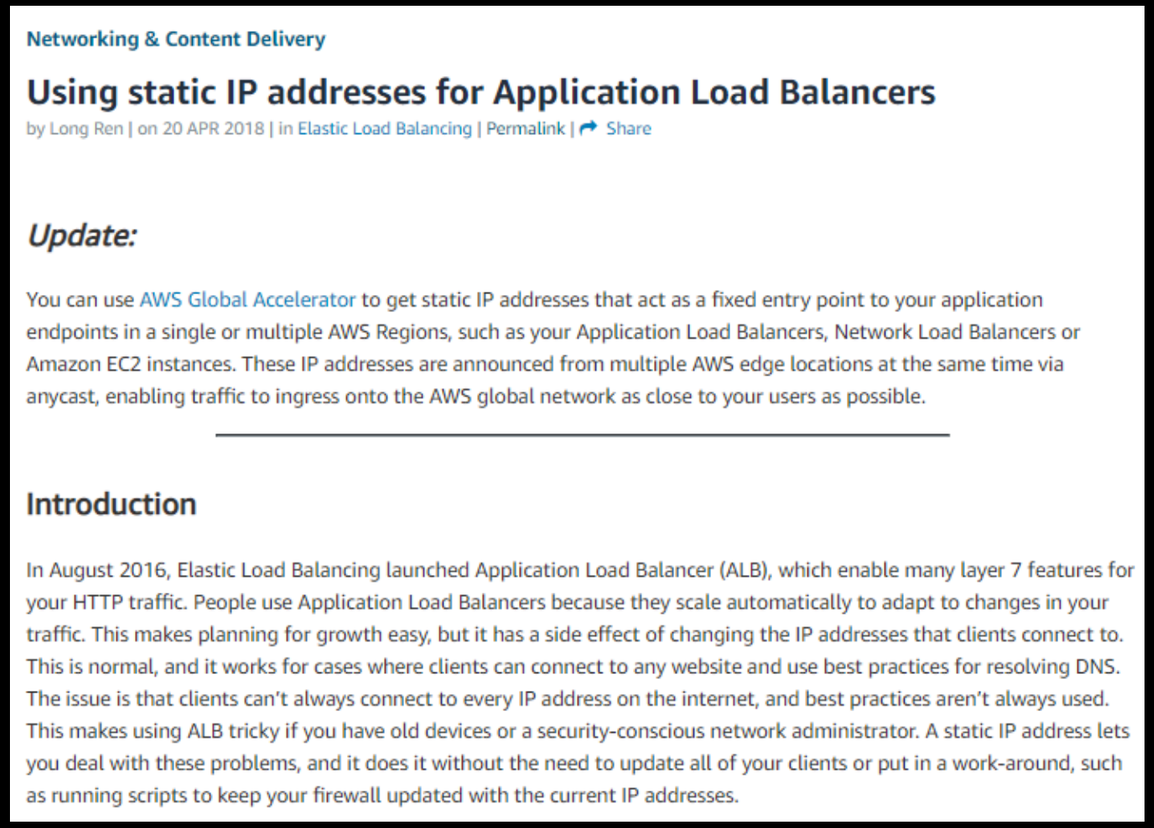
Serverless microservices with NLB

- TCP / UDP support
- Use cases
  - Non HTTP / HTTPS APIs
  - High throughput
  - Ultra low latency
  - Static IP needed
- NLB > microservice (1:1)
- NLB > ALB > Microservices (1:1:many)



ALB as target for the NLB

BLOG POST – USING STATIC IP ADDRESSES FOR APPLICATION LOAD BALANCERS



Networking & Content Delivery

Using static IP addresses for Application Load Balancers

by Long Ren | on 20 APR 2018 | in Elastic Load Balancing | Permalink | Share

Update:

You can use AWS Global Accelerator to get static IP addresses that act as a fixed entry point to your application endpoints in a single or multiple AWS Regions, such as your Application Load Balancers, Network Load Balancers or Amazon EC2 instances. These IP addresses are announced from multiple AWS edge locations at the same time via anycast, enabling traffic to ingress onto the AWS global network as close to your users as possible.

Introduction

In August 2016, Elastic Load Balancing launched Application Load Balancer (ALB), which enable many layer 7 features for your HTTP traffic. People use Application Load Balancers because they scale automatically to adapt to changes in your traffic. This makes planning for growth easy, but it has a side effect of changing the IP addresses that clients connect to. This is normal, and it works for cases where clients can connect to any website and use best practices for resolving DNS. The issue is that clients can't always connect to every IP address on the internet, and best practices aren't always used. This makes using ALB tricky if you have old devices or a security-conscious network administrator. A static IP address lets you deal with these problems, and it does it without the need to update all of your clients or put in a work-around, such as running scripts to keep your firewall updated with the current IP addresses.

Architectural patterns

Exposing private microservices using APIGW

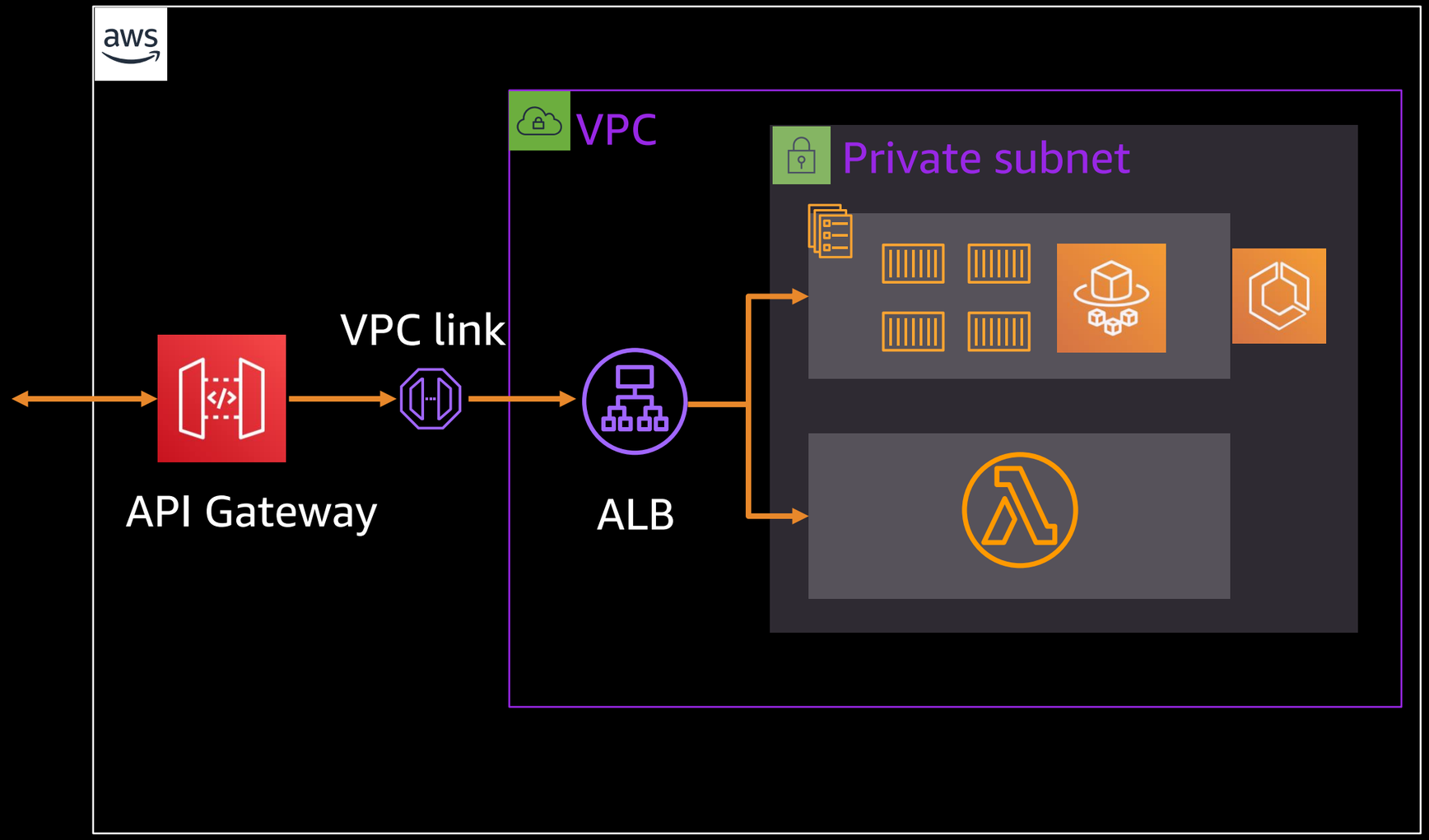
AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Exposing private microservices using APIGW

APPLICATION LOAD BALANCER

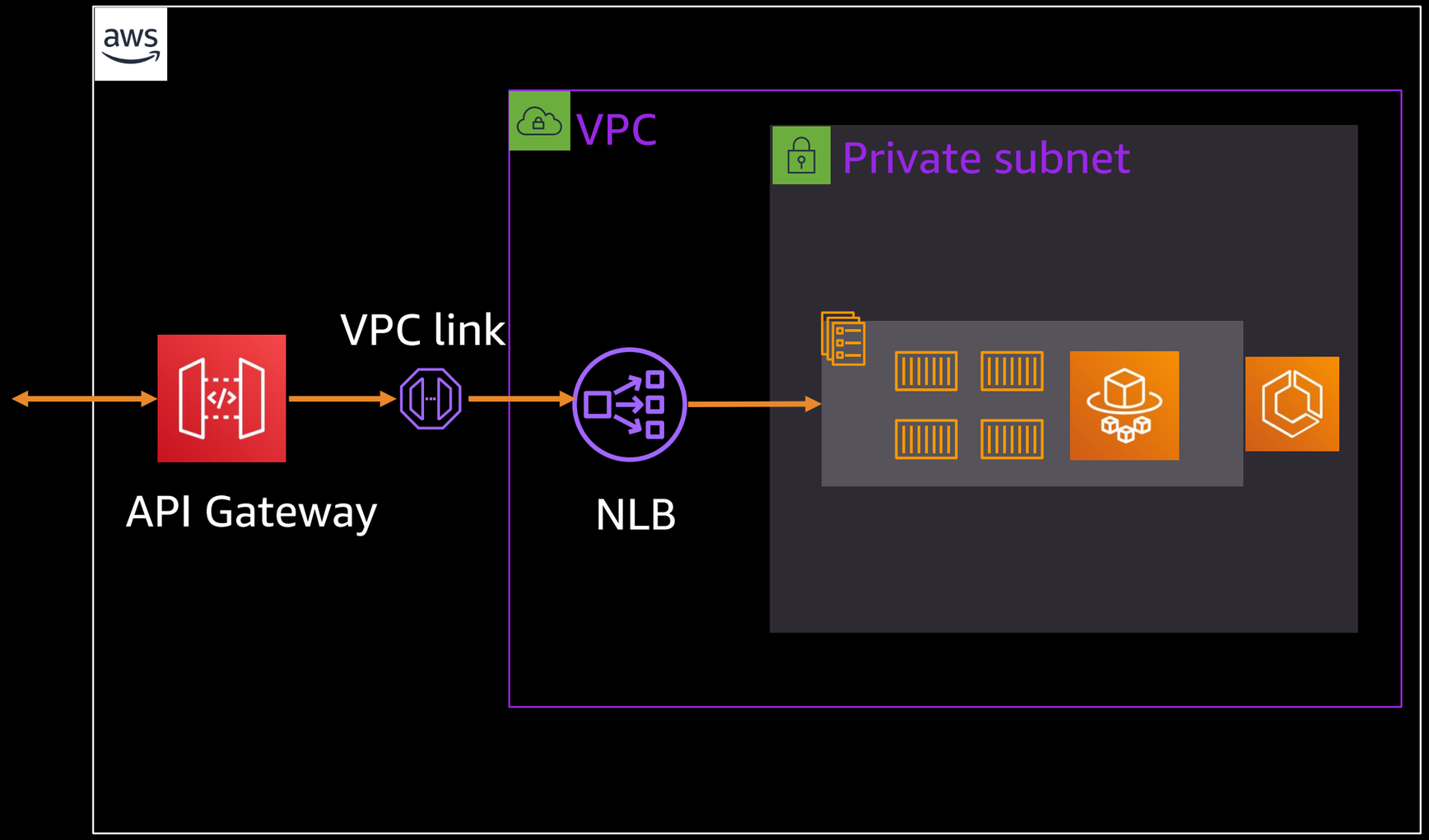
- Use case: Private access to private microservices using HTTP APIs
- APIGW private integration with ALB
- All resources need to be owned by the same AWS account



Exposing private microservices using APIGW

NETWORK LOAD BALANCER

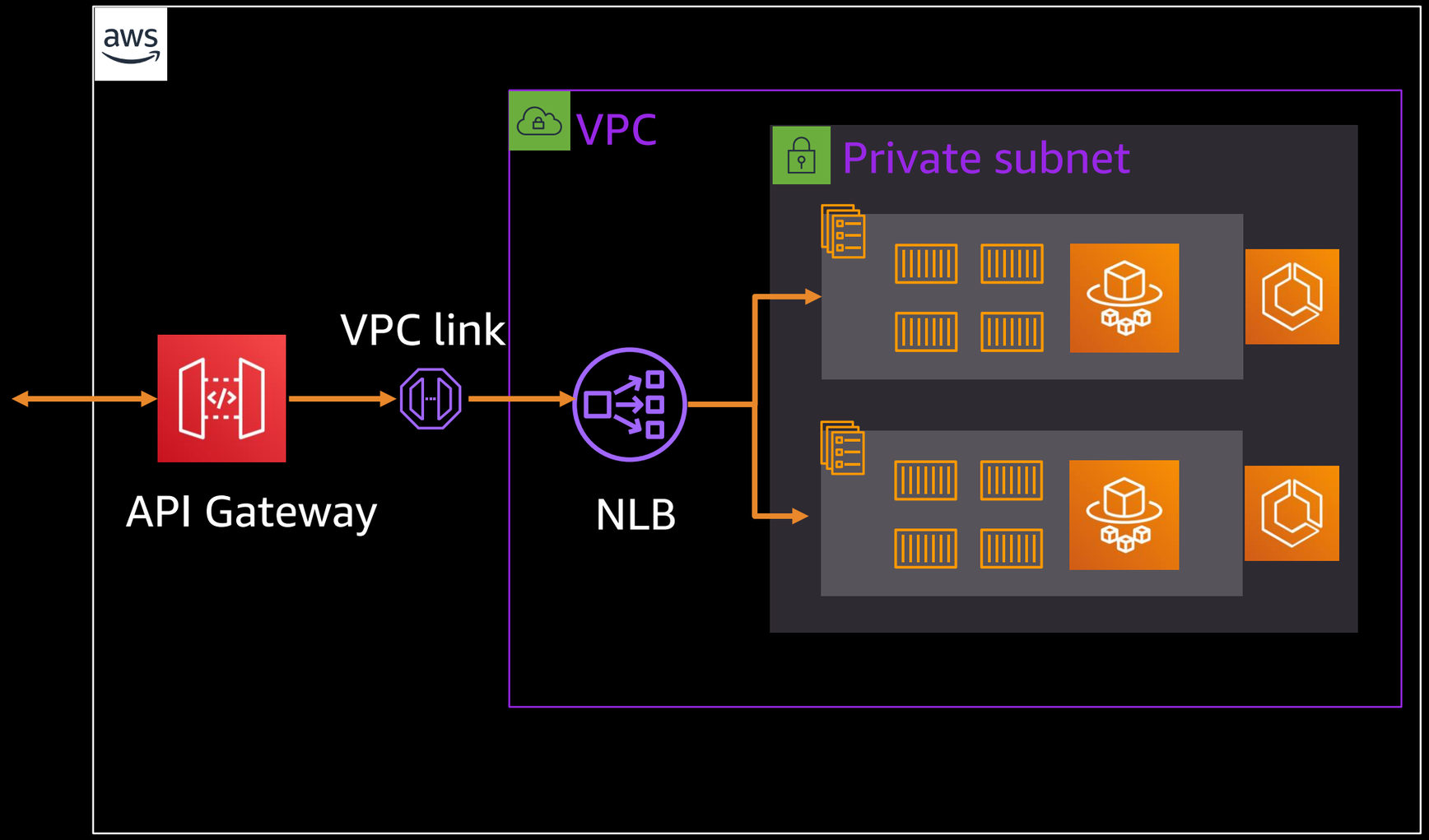
- Use case: Private access to private microservices using REST APIs
- APIGW private integration with NLB
- All resources need to be owned by the same AWS account



Exposing private microservices using APIGW

NETWORK LOAD BALANCER

- Use case: Private access to private microservices using REST APIs
- APIGW private integration with NLB
- All resources need to be owned by the same AWS account
- Multiple services exposed



Related session

- NET402 Elastic Load Balancing: A year of innovations



Thank you!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.