Elastic Load Balancer and Auto Scaling Groups

Services Overview and Hands-on Lab

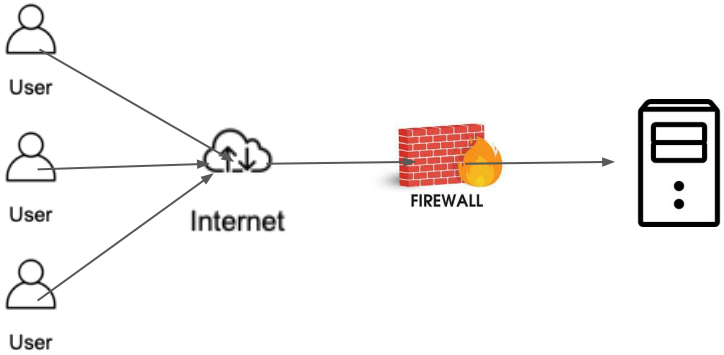
Acknowledgements

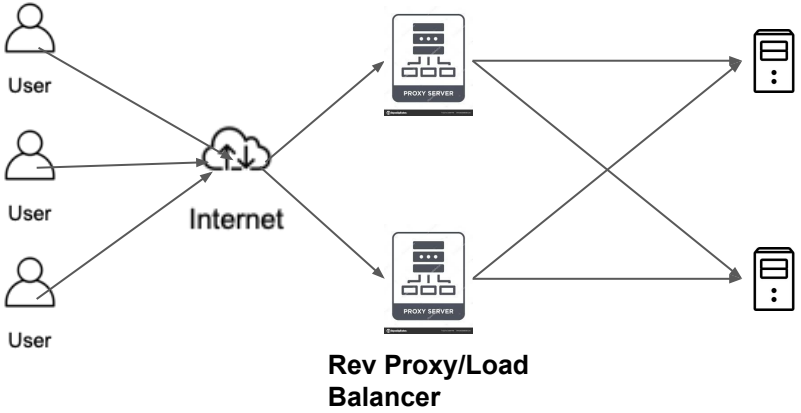
- Most of the picture/slides with diagram with dark background are taken from AWS reInvent Net407-R slides/pdf format
- All other slide contents are created by using information i have learnt from various resources as well as AWS user guide
- For more details on AWS ELB Service, please refer the below URL for ELB user guide
- https://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/what-is-load-balancing.html

Agenda

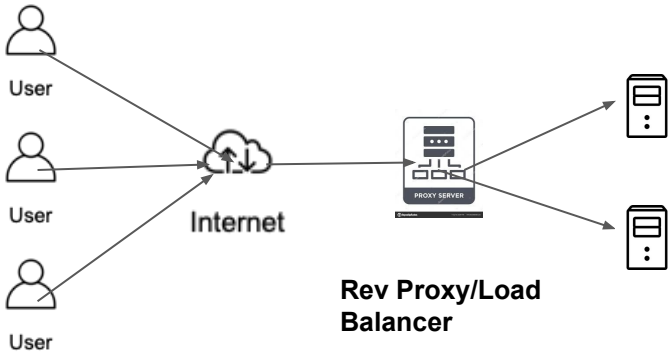
- What’s a Load Balancer? Why do we need it?
- What’s ELB/Elastic Load Balancing?
- How ELB Works?
- ELB Concepts/Fundamentals
- Types of ELB
- Auto Scaling Groups
- Hands-on Lab
    - Simple ELB/ALB
    - ASG Demo
    - ELB + ASG Demo

Load Balancer Evolution.. Do we need it?





Rev Proxy/Load Balancer



Rev Proxy/Load Balancer

- Need to address single point of failure
- Efficiently balance traffic flows
- Availability vs Scalability
- Security Integration
- Health Monitoring of LB, Web/App server
- Automatic scaling
- SSL Termination/bridging
- Cost effective solution etc

Elastic Load Balancing (ELB) Overview

- Elastic Load Balancing (ELB) automatically distributes incoming traffic across multiple targets
  - Targets can be EC2 instances, containers, and IP addresses, in one or more Availability Zones.
- ELB monitors the health of its registered targets, and routes traffic only to the healthy targets.
- Elastic Load Balancing scales itself as incoming traffic changes over time.

Elastic Load Balancing (ELB) Overview (Cont..)

- ELB can be implemented as Internet facing service or Internal Service
- ELB can be implemented in a tiered Architecture to route, load balance traffic across n-tier Application Architecture
- ELB + VPC Security feature provides a robust & scalable networking, security & load balancer Architecture

ELB Benefits

- ELB distributes workloads across multiple compute resources, such as virtual servers.
- ELB increases the availability and fault tolerance of your applications.
- ELB scales itself horizontally depending on application traffic load levels
- Compute resources behind ELB can be scaled horizontally as needed (manually or automatically using ASG), without disrupting the overall application flows.
- ELB provides/uses health checks to monitor the health of the compute resources, so that it can send requests only to the healthy instances.
- ELB can offload the work of encryption and decryption from Web server so that compute resources running application/web server can focus on their main work.
- ELB integrates with many other AWS services such as Cloudwatch, Route53, WAF, ACM etc

Types of Elastic Load Balancers

Elastic Load Balancing

Gateway Load Balancer

Application load balancer

Classic load balancer

Network load balancer

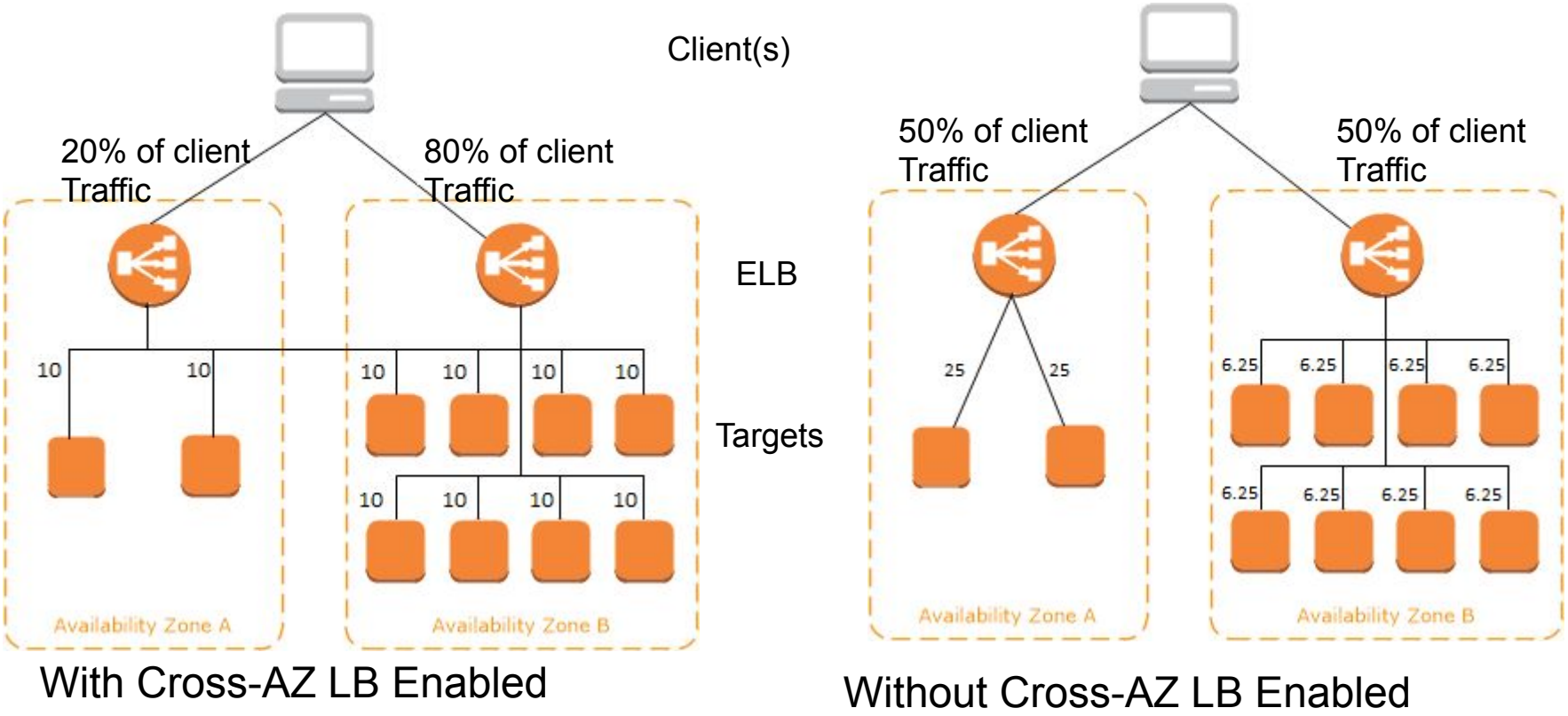
Elastic Load Balancing Fundamentals

- Listener - Port/Protocol pair or combination
- Targets - EC2, ECS, IP Endpoints
- Target groups - Grouping of targets
- Launch Template - Used to templatize Instance launch parameters
- Target Tracking/Scaling Policy - Used to track EC2/ELB Cloudwatch alarm/metric to horizontally scale Target group
- ASG - Auto Scaling Group - Helps to scale Targets/Target groups based on Target tracking policy + Launch Template
- Traffic Routing/Forwarding - Client to Server connectivity

How Elastic Load Balancing Works?

- ELB is configured to accept incoming traffic by specifying one or more *listeners*.
- A listener is a process that checks for connection requests.
- It is configured with a protocol and port number for connections from clients to the load balancer.
- ELB is also configured with a protocol and port number for connections from the load balancer to the targets.
- ELB accepts incoming traffic from clients and routes requests to its registered targets (such as EC2 instances) in one or more Availability Zones.
- ELB monitors the health of its registered targets and ensures that it routes traffic only to healthy targets.
- When ELB detects an unhealthy target, it stops routing traffic to that target. It then resumes routing traffic to that target when it detects that the target is healthy again.

Illustration of How ELB Works?



With Cross-AZ LB Enabled

Without Cross-AZ LB Enabled

Above diagram is from the below AWS user guide with my annotations
https://docs.aws.amazon.com/elasticloadbalancing/latest/userguide/how-elastic-load-balancing-works.html

Client connectivity

Client to Elastic Load Balancer connections

Client connectivity

Route 53 → ELB

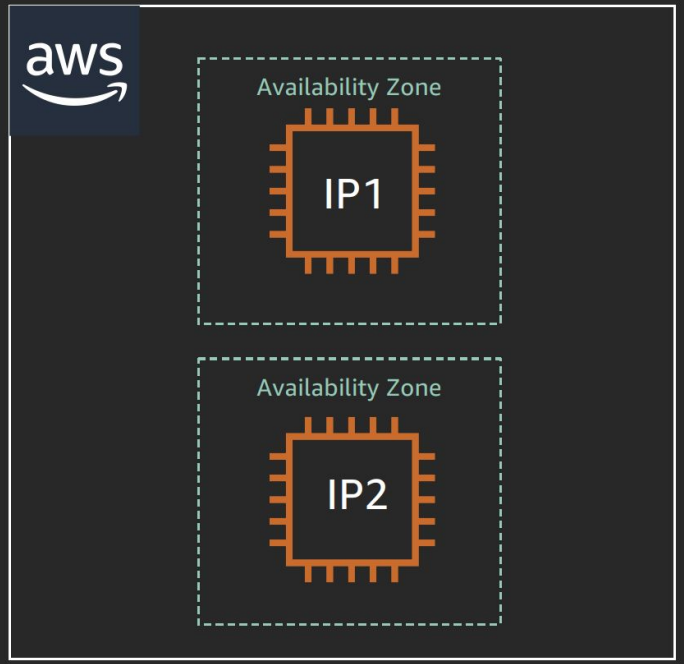
Amazon Route 53

Users

*Route 53 aliases

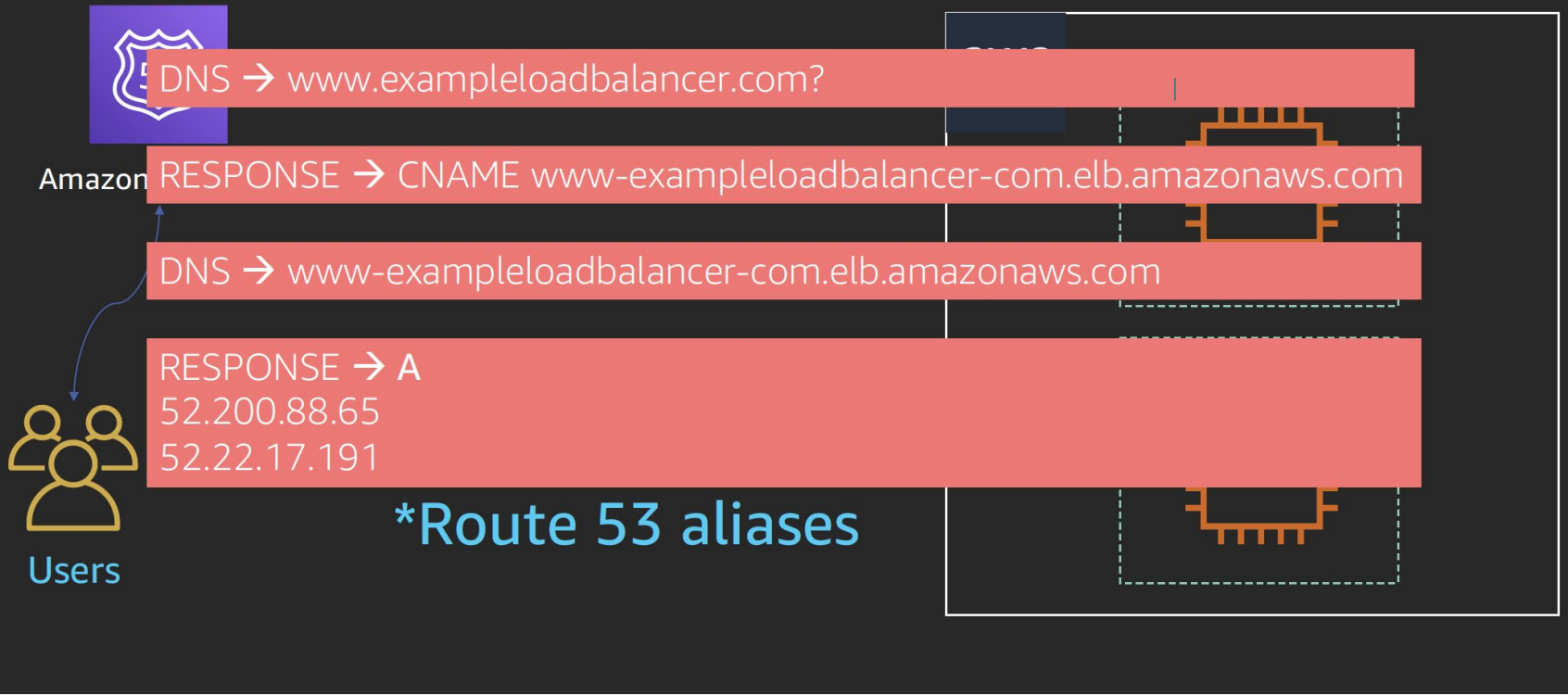
ELB

www.exampleloadbalancer.com*



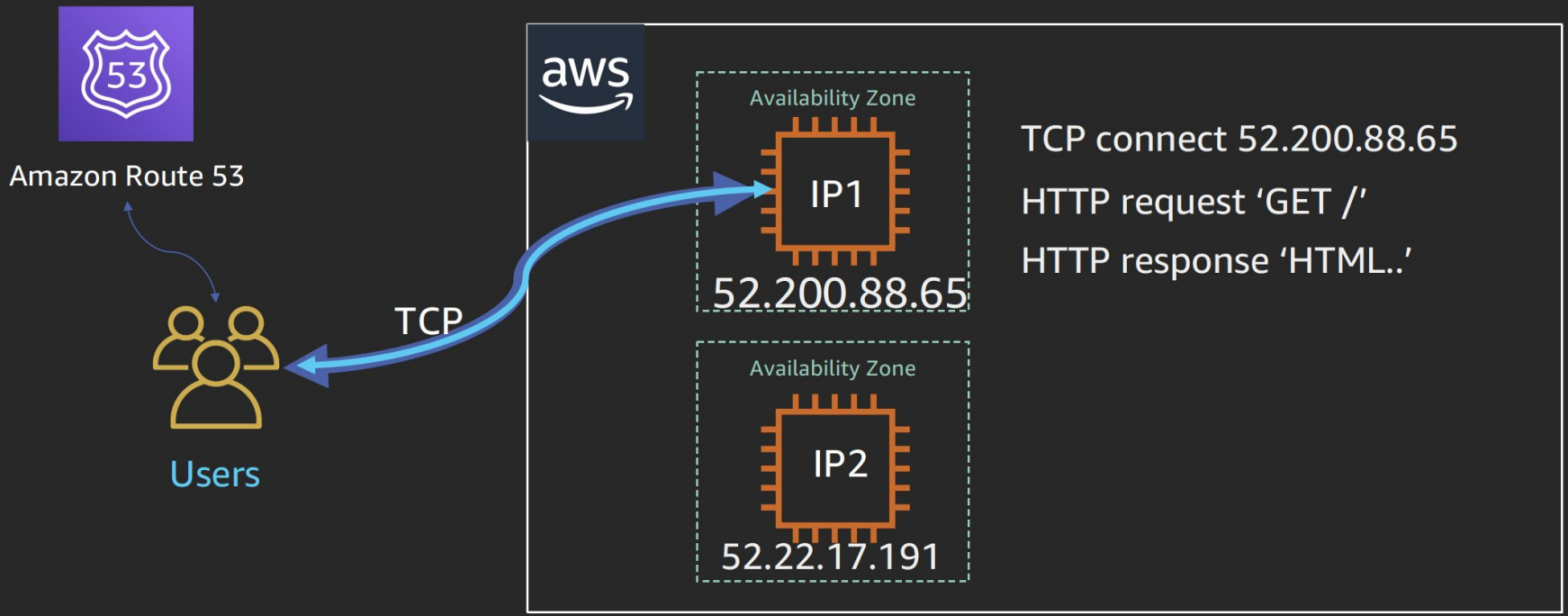
Client connectivity

Route 53 → ELB



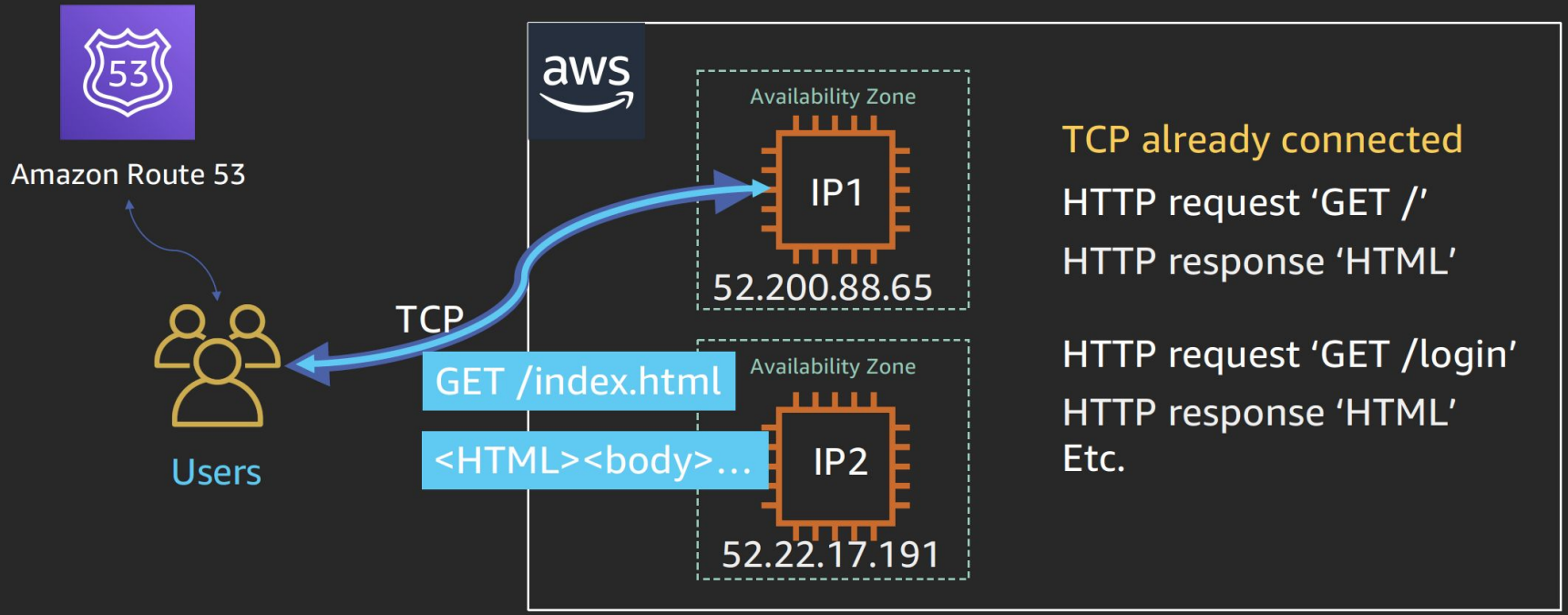
Client connectivity

Route 53 → ELB



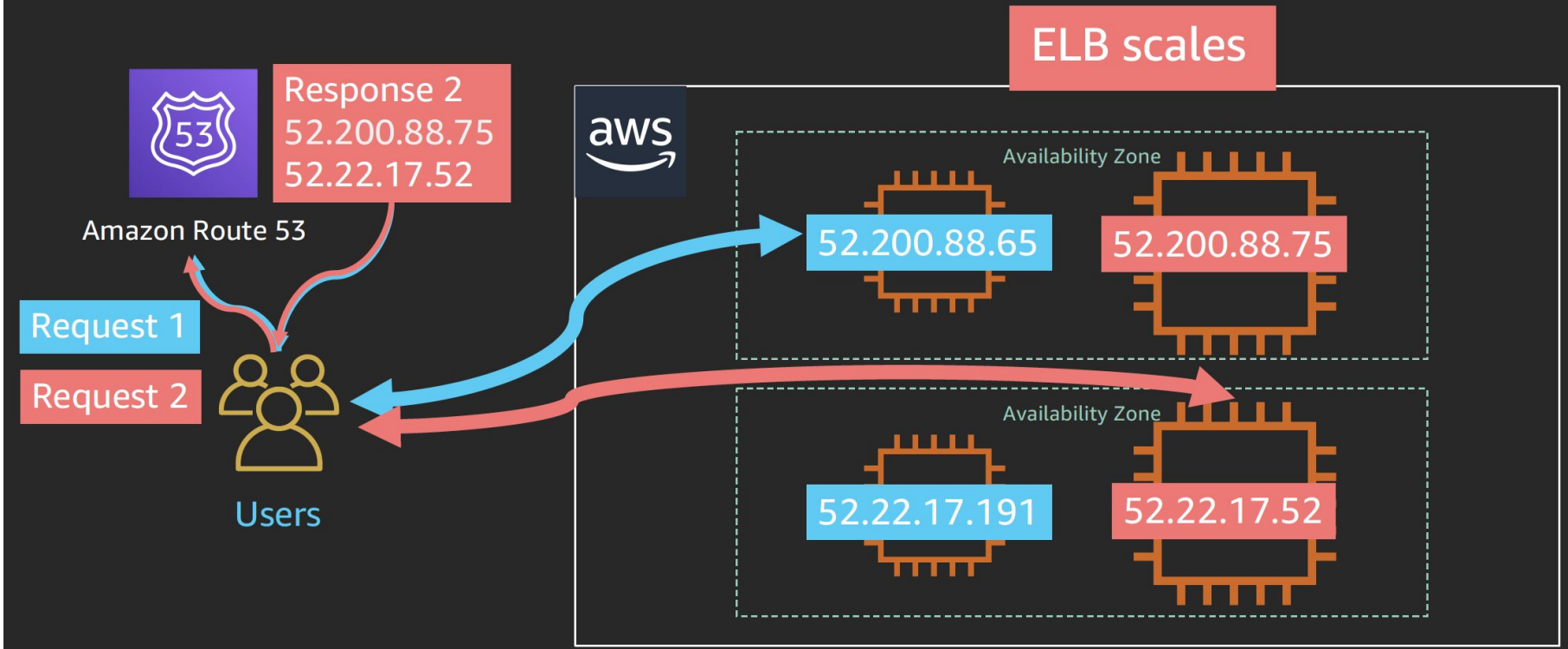
Client connectivity

Route 53 → ELB



Client connectivity

Route 53 → ELB

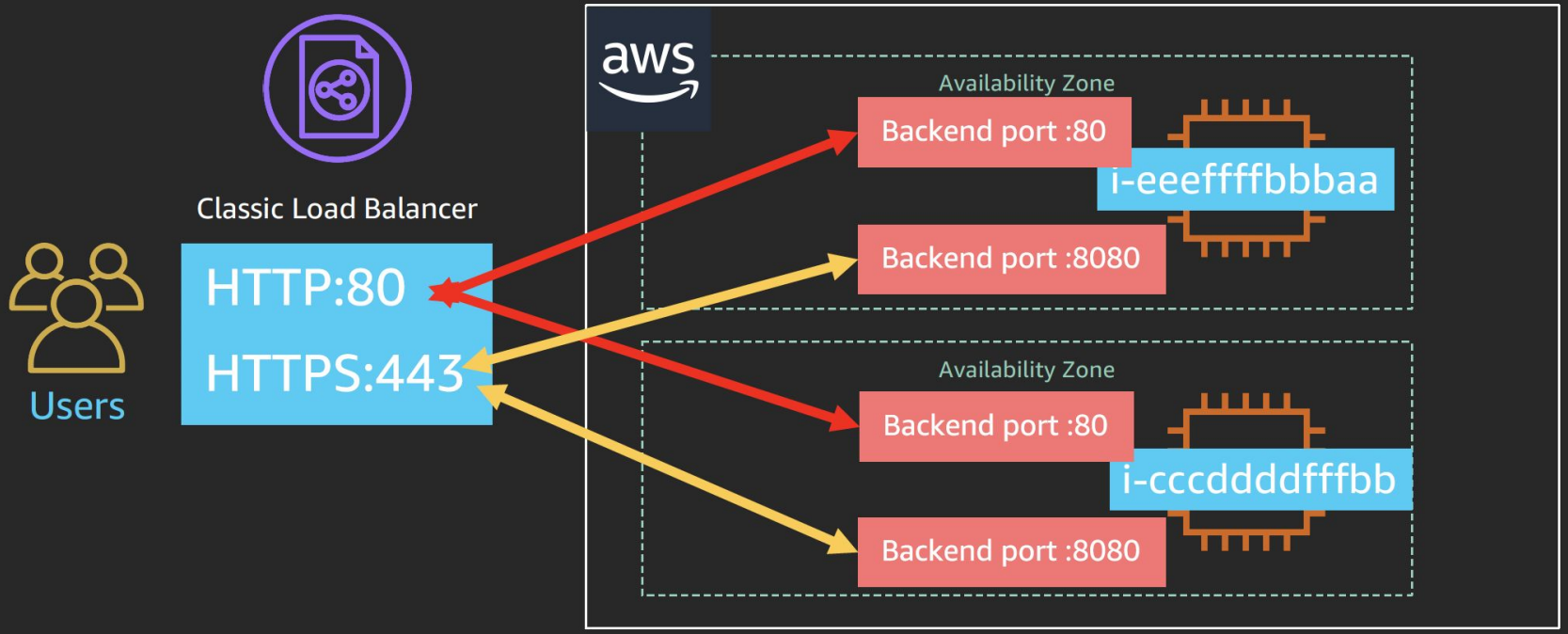


Client connectivity and request routing

Classic Load Balancer request routing

Client connectivity and request routing

Route 53 → ELB (listeners) → Backend target



Client connectivity and request routing

Route 53 → ELB (listeners) → Backend target

Classic Load Balancer

HTTP/HTTPS

Routes per HTTP request

Least outstanding request routing

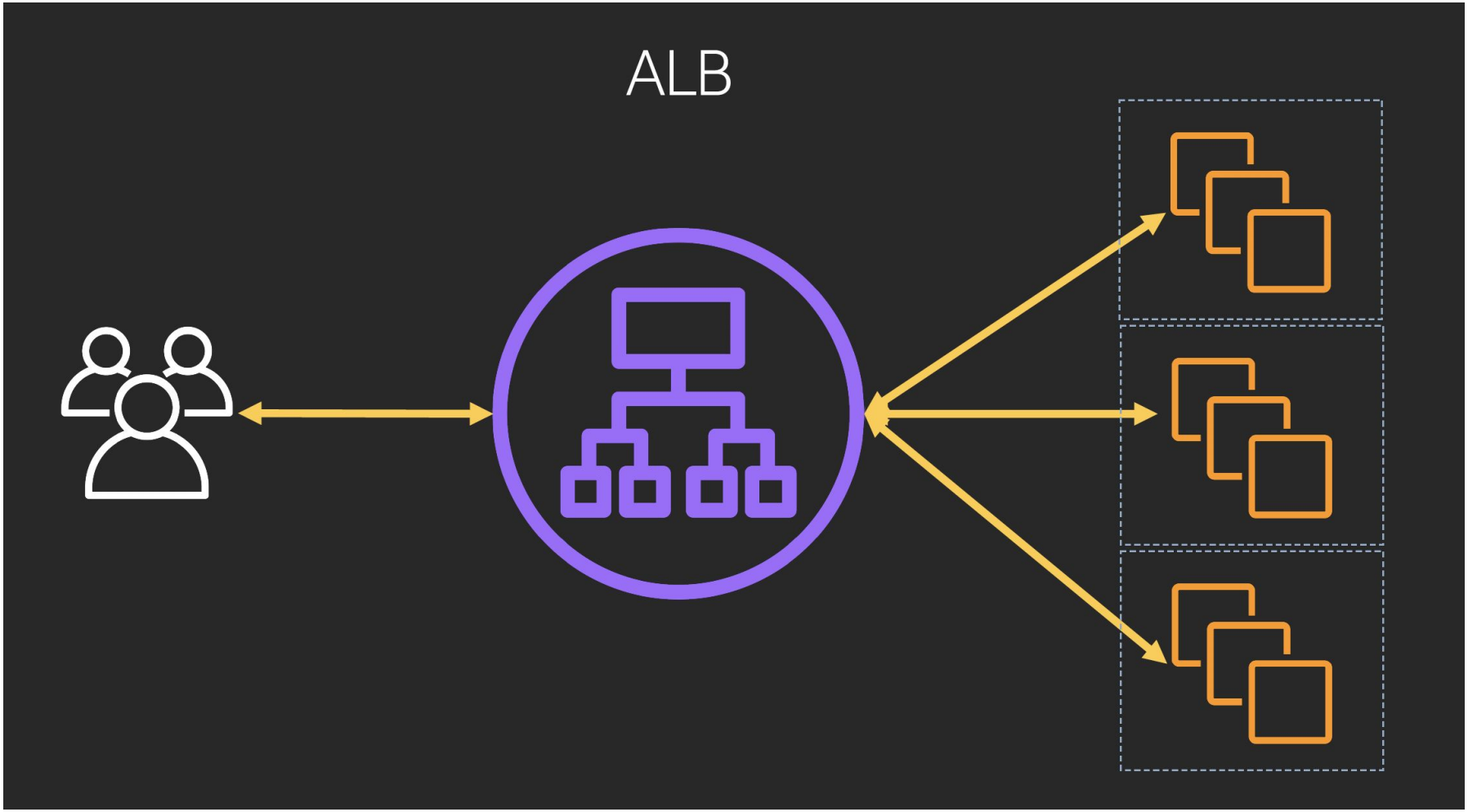
TCP/TLS (SSL)

Routes per TCP connect

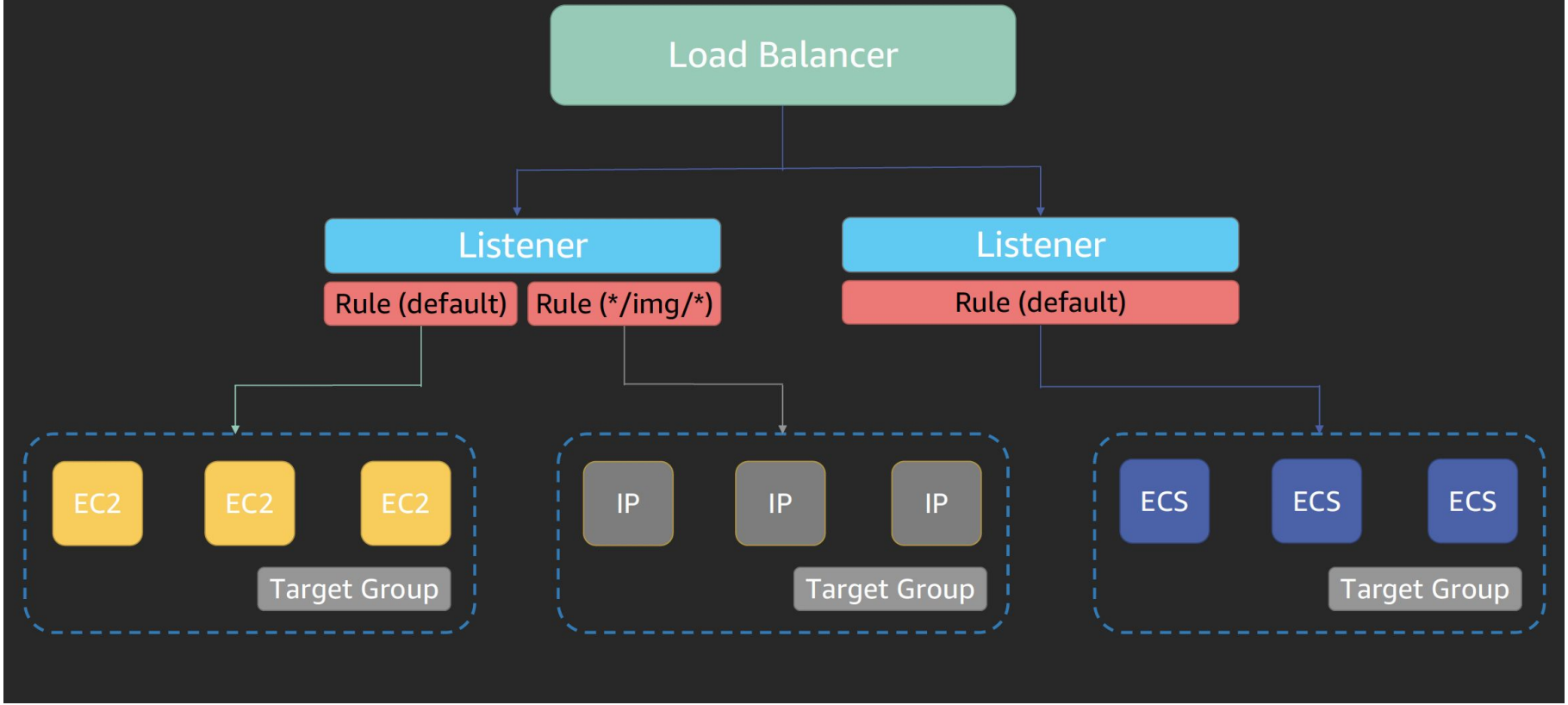
Round robin connection routing

Client connectivity and request routing

Application Load Balancer request routing, target groups & rules


ALB

Logical overview of an ALB



Target Group Rules

Client connectivity and request routing

Route 53 → ELB → Target Group (rules) → Target

Application Load Balancer allows for multiple web services to be hosted behind a single load balancer

Up to 100 rules

Host header
Host: www.example.com
Path
/webapp
HTTP headers
X-MyApp-Source: 'Cloudfront'
HTTP methods
GET, POST
Query parameters and query string
?Stage=Production
Source IP addresses
99.87.0.0/17

Conditions and actions in a rule

5 matches (evaluations) per rule

5 wildcards per rule

100 rules per load balancer

You can combine existing rules

Condition (if)

- Host header
- URI path
- Arbitrary header
- URI query string
- HTTP method

Match 1 OR Match 2 OR Match 3

AND

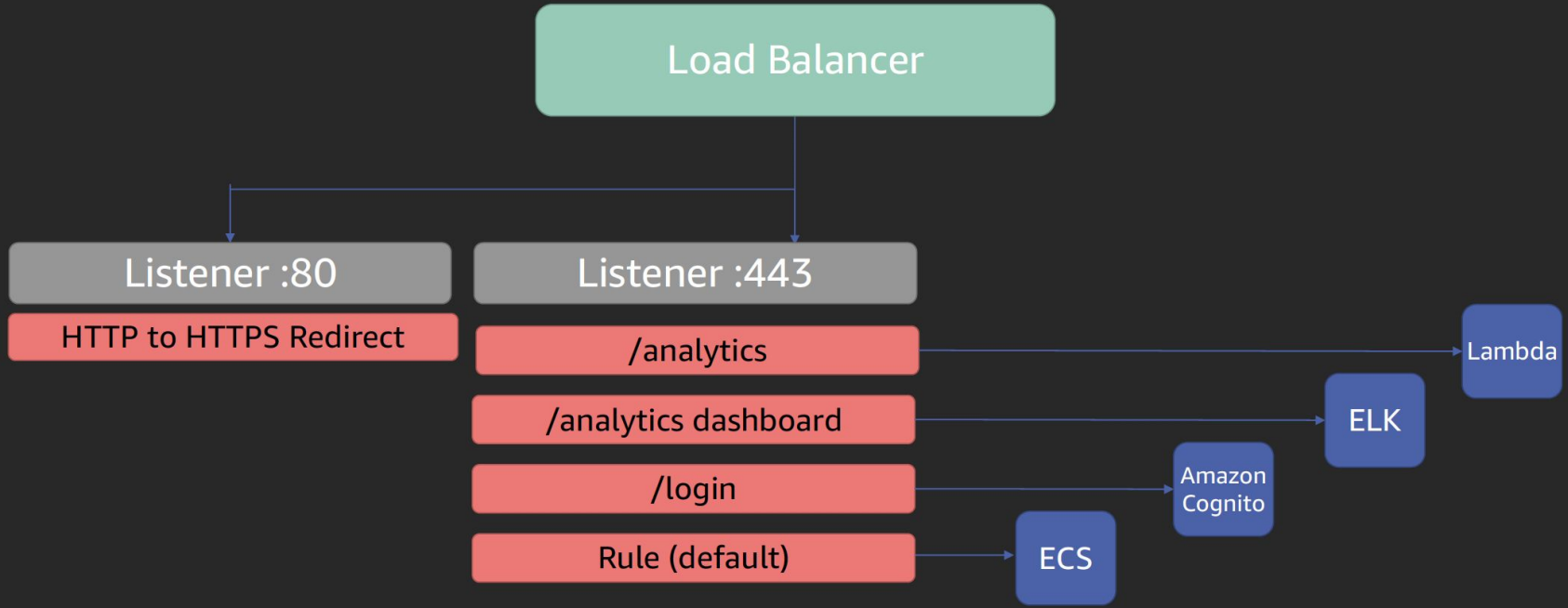
Next condition

Action (then)

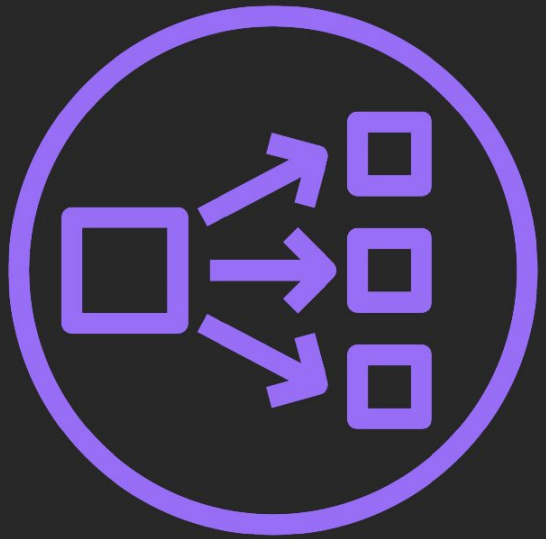
- Forward
- Redirect
- Fixed response
- Authenticate

Client connectivity and request routing

Route 53 → ELB → Target Group (rules) → Target

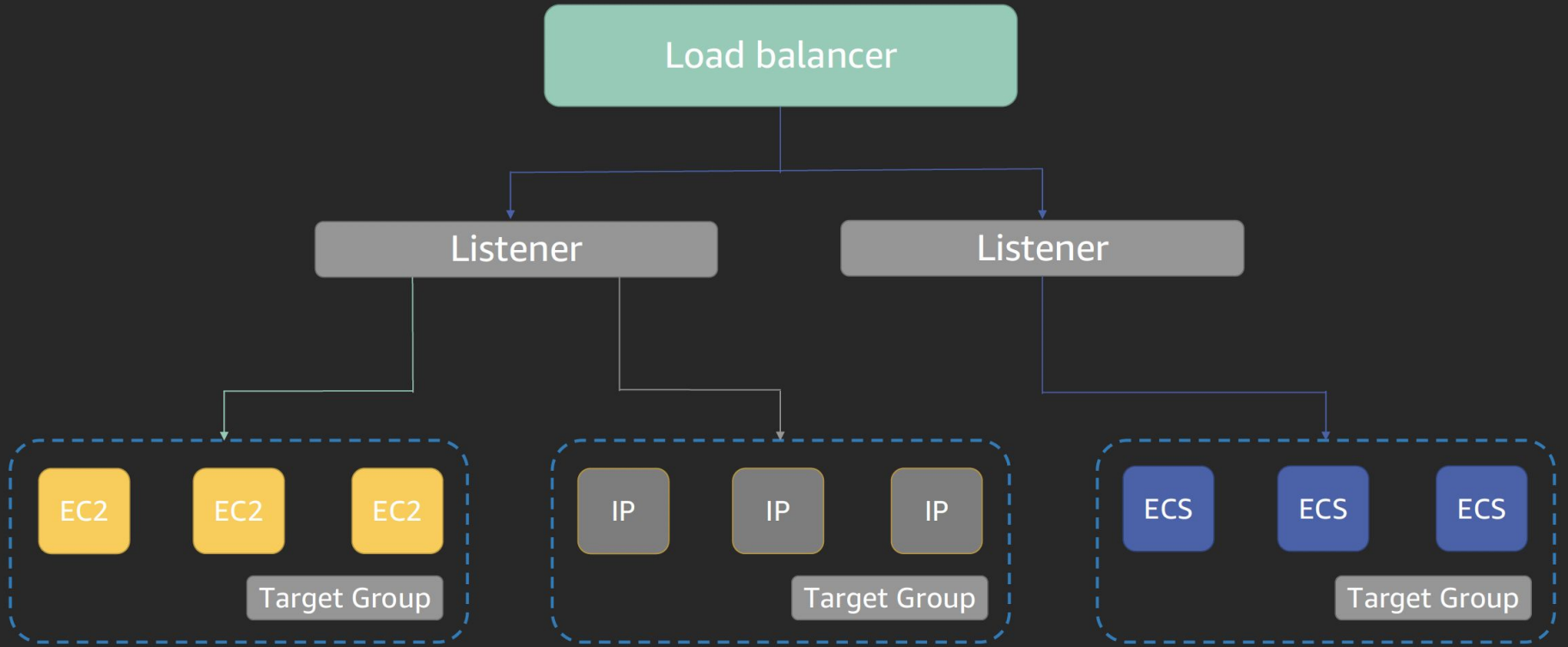


Network Load Balancer

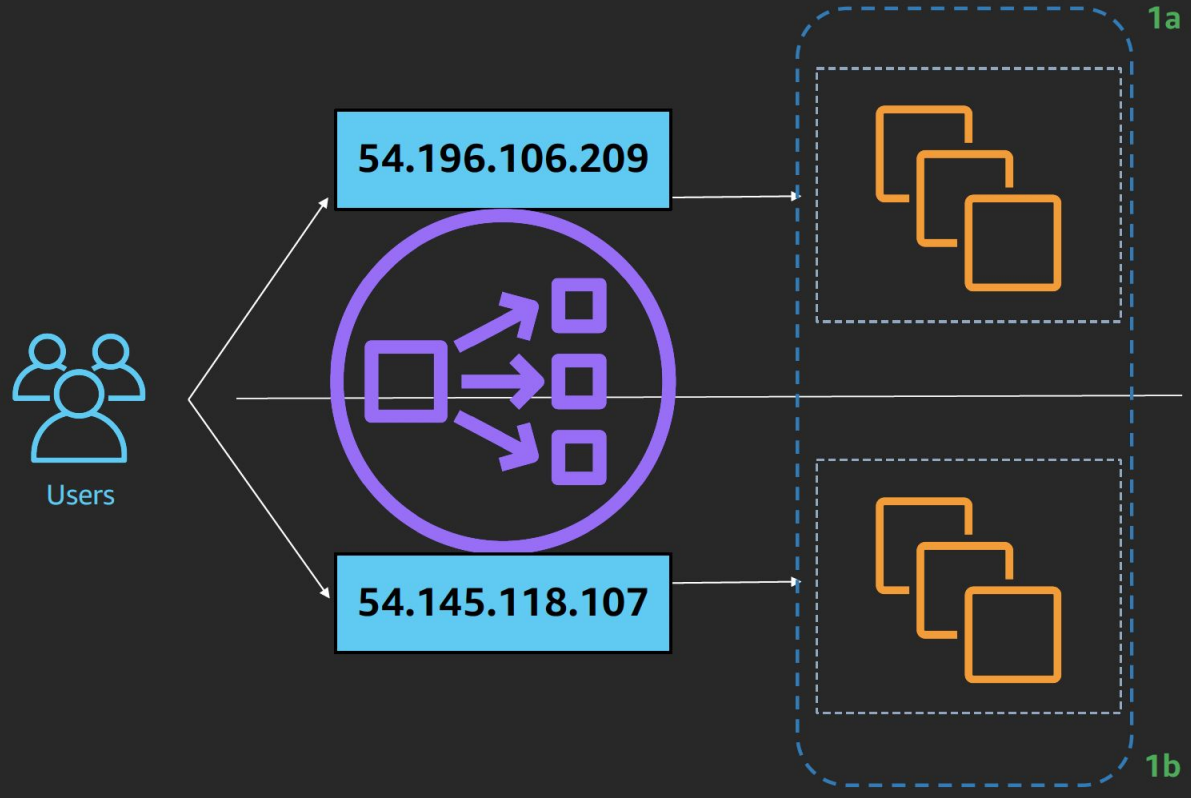


NLB

Logical overview of NLB

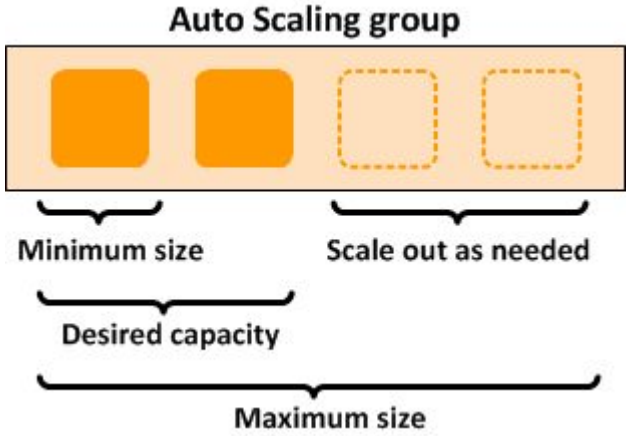


NLB Elastic IP addresses



Assigning Elastic IP provides a single IP address per Availability Zone per load balancer that will not change

Auto Scaling Group



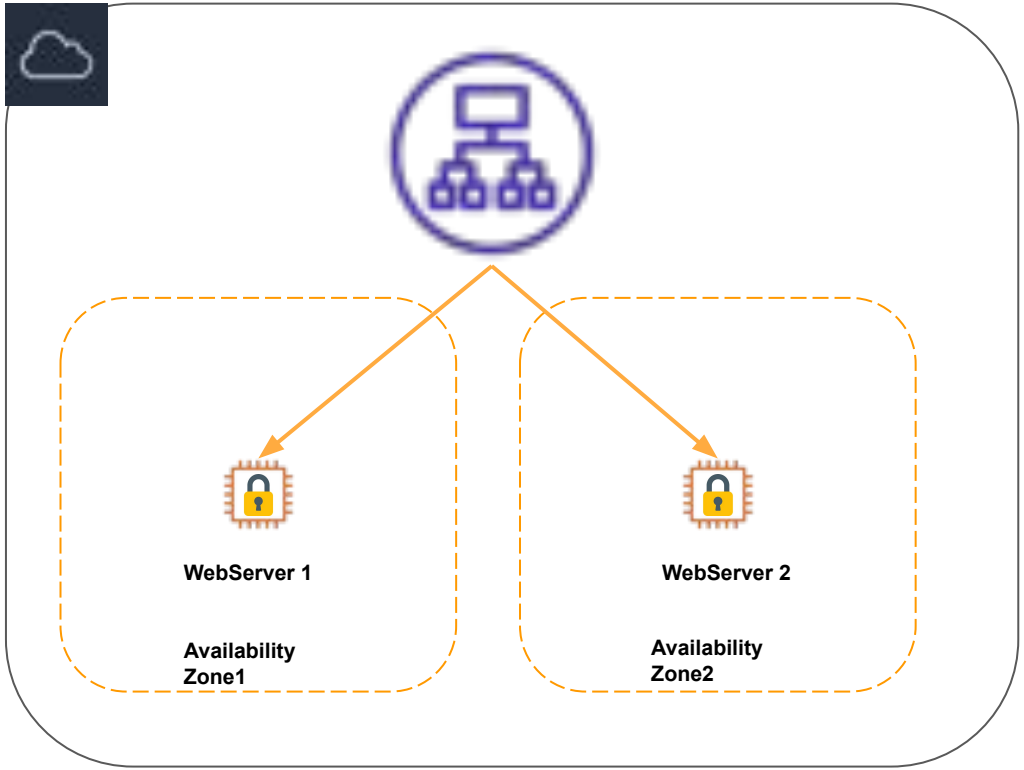
- ASG Provides a way for the end user to define minimum, desired and maximum target capacity
- ASG ensures desired number of target/instance is launched
- With help of Cloudwatch it then tracks the health and adjusts dynamically the number of instances/targets
- Picture/Diagram from : https://docs.aws.amazon.com/autoscaling/ec2/userguide/what-is-amazon-ec2-auto-scaling.html

Auto Scaling Group

- ASG uses Launch Template to launch instances as needed/governed by Target tracking/scaling policy
- ASG is integrated with ELB/Target group
- ELB is not aware of the presence of ASG
- For more information refer/read the below AWS user guide
- https://docs.aws.amazon.com/autoscaling/ec2/userguide/what-is-amazon-ec2-auto-scaling.html

What will be building today?

Simple ELB/ALB Setup shown below



Use Default VPC for this hands-on lab

1. Instantiate two EC2 instances with apache2 web app
2. Make sure to use two different AZ
3. Check if both the servers are accessible externally
4. Bring up ALB and add the above two servers into a target group
5. Make sure the web servers are accessible only through ALB
6. Test ALB load balances traffic across both servers in two different AZ

Auto Scaling Group Hands-on Lab

- Create Launch Template under EC2 console
- Make sure there are no instances running (started manually)
- Ensure there are no ELB, Target groups
- Create an Auto Scaling Group with min=1, desired=2, max = 3 instances
- Tweak health check, default cool down timers (to expedite / speed up scaling)
- Confirm two instances are launched automatically using Launch Template
- Terminate an instance and confirm ASG automatically launches another instance to keep the instance count to the desired number

ELB + Auto Scaling Group Hands-on Lab Cheat sheet

1. Create Target group for ELB (Update deregistration delay to a very low value)
2. Create SG for ELB (to allow http access from everywhere)
3. Create ELB (ALB) n use existing TG, attach ELB SG
4. Create SG for EC2 instances (allow access only from ELBSG)
5. Create LaunchTemplate (Make sure to use user data to install webserver)
6. Create ASG, attach it to ELB and use existing TG (Set ASG Thresholds, desired, min, max), ASG/Advance config tweak def cooldown period
7. Auto scale policy??
8. Test Scale out, scale in
9. use stress tool

User Data to bootstrap Apache2 Webserver

```
#!/bin/bash

#Install Apache/httpd Web Server

yum update -y

yum install -y httpd.x86_64

systemctl start httpd.service

systemctl enable httpd.service

echo “Web Server is running on $(hostname -f)” > /var/www/html/index.html
```

Install cpu stress tool on AMZN AMI

```
sudo amazon-linux-extras install epel -y
```

```
Use above command to install AMZN AMI Extras Repo
```

sudo yum install stress

Above command to install "stress" package

stress --cpu 1 --timeout 300

Backup Slides

Listeners and connectivity

ELB comparison	ALB	NLB	CLB
Listener protocols	HTTP, HTTPS,HTTP/2	TCP, UDP, TLS	TCP, SSL, HTTP, HTTPS
TLS/SSL termination	✓	✓	✓
IAM Certificates	✓	✓	✓
ACM Certificates (with automated rotation)	✓	✓	✓
Websockets (HTTP upgrade)	✓		
Static IP (Use your own Elastic IP)		✓	
Preserve client IP		✓	
X-forwarded HTTP headers	✓		✓
Proxy protocol on TCP connections		✓	✓
VPC IPv6	✓	✓	
AWS PrivateLink		✓	

Request routing

ELB comparison	ALB	NLB	CLB
HTTP request load balancing	✓		✓
TCP flow/connection load balancing		✓	✓
UDP flow load balancing		✓	
HTTP host header	✓		
HTTP method	✓		
Source IP address or CIDR	✓		
Arbitrary HTTP header	✓		
HTTP query string parameter	✓		
Combined rules	✓		
AWS WAF	✓		
Weighted Target Group routing	✓		

Targets and target groups

ELB comparison	ALB	NLB	CLB
EC2 instances	✓	✓	✓
IP addresses	✓	✓	
Per-target ports	✓	✓	
Shared-target ports			✓
HTTP request logging	✓		✓
TCP connection logging		✓(TLS only)	✓
Containers (ECS, EKS, Kubernetes, Docker)	✓	✓	
Security groups	✓		✓
HTTP(S) health checks	✓	✓	✓
TCP health checks	✓	✓	✓