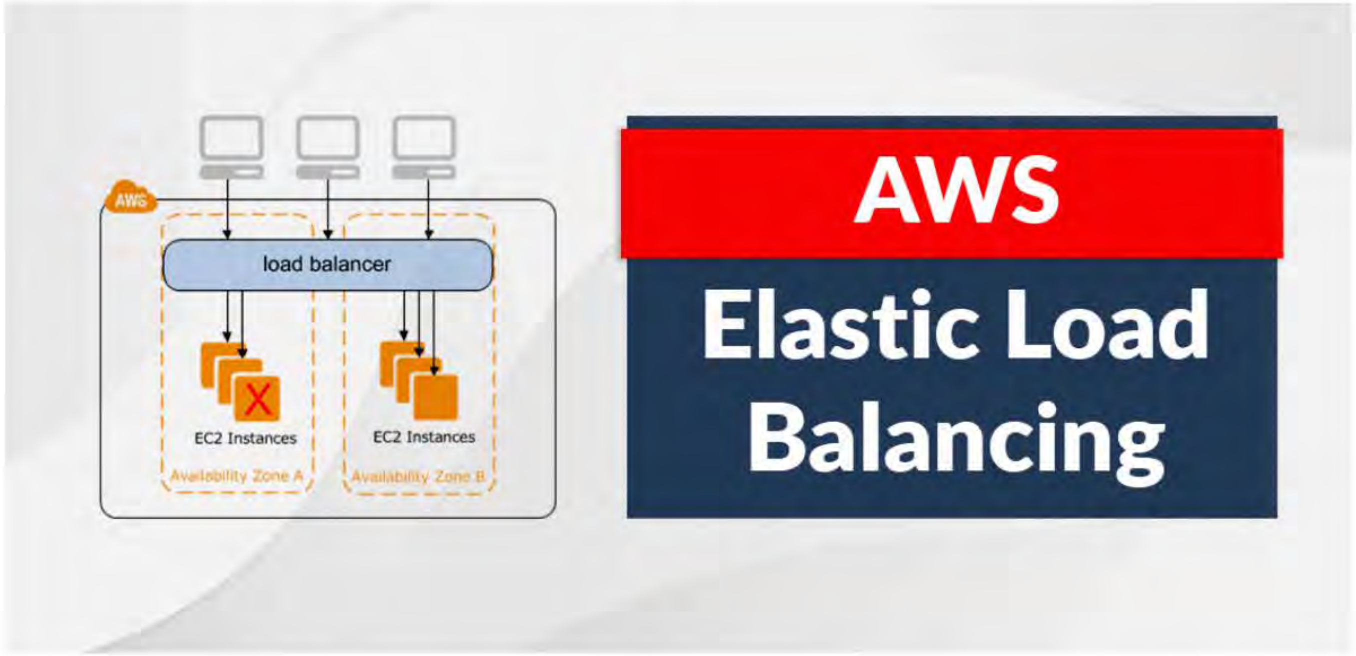
iamneo

AWS Load Balancer

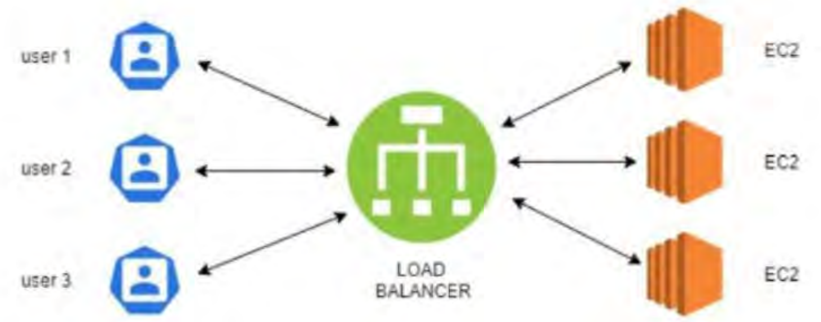
www.iamneo.ai

AWS Load Balancer



Amazon Web Services (AWS) Load Balancer distributes traffic to multiple targets such as EC2 instances or containers, which increases the availability of your applications. In this presentation, we will explore different types of Load Balancers and how to create and configure them.

Load Balancing



- ➢ Single point of access (DNS) to your application. Provide
- ➢ SSL termination (HTTPS) for your websites.
- ➢ Separate public traffic from private traffic.
- ➢ It is integrated with many AWS offerings / services.
- ➢ You can setup internal (private) or external (public) ELBs.
- ➢ AWS guarantees that it will be working AWS takes care of upgrades, maintenance, high availability.

Types of AWS Load Balancers

Application Load Balancer

Network Load Balancer

Classic LoadBalancer

Gateway Load Balancer

Types of AWS Load Balancers

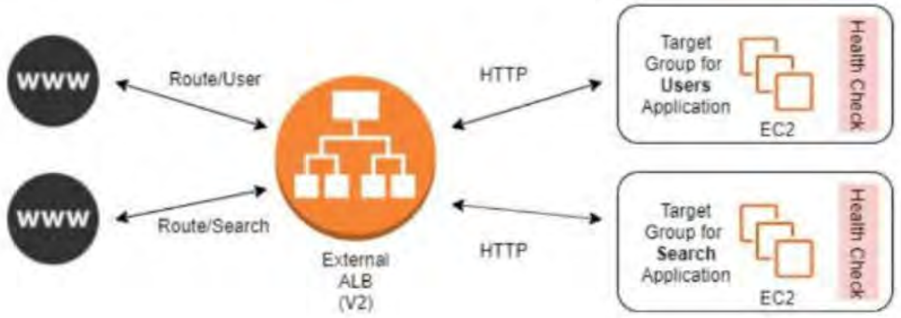
Application Load Balancer

Network Load Balancer

Classic LoadBalancer

Gateway Load Balancer

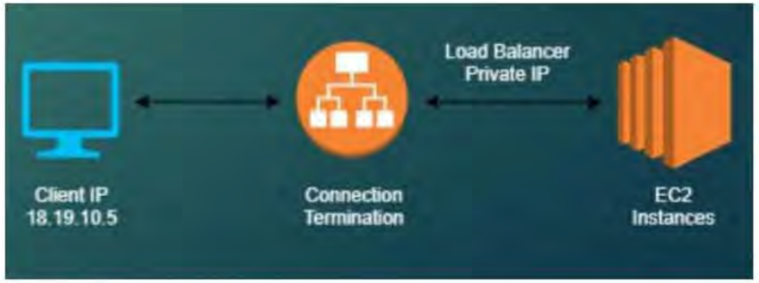
Application Load Balancer (V2)



Application load balancers (Layer 7) allow you to do:

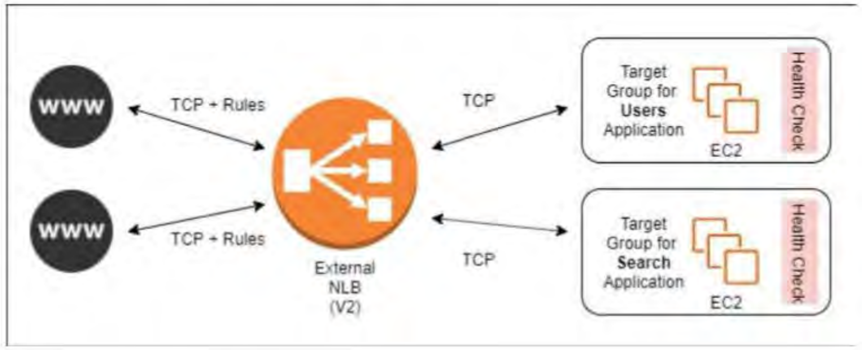
- ➢ Load balancing for multiple HTTP applications across machines (target groups).
- ➢ Load balancing for multiple applications running on the same machine(ex: containers).
- ➢ Load balancing based on route (path) in URL.
- ➢ Load balancing based on Hostname in URL.

Application Load Balancer (V2)



- ALB is perfect for micro services & container-based applications.
- Port mapping feature to redirect to a dynamic port.
- Previously we used to create one Classic Load Balancer per application.
- That was very expensive and inefficient!
- Stickiness can be enabled at the target group level.
- ALB support HTTP/HTTPS & Web sockets protocols.
- The application servers don't see the IP of the client directly. Instead, it is inserted in the header X-Forwarded-For.

Network Load Balancer (V2)

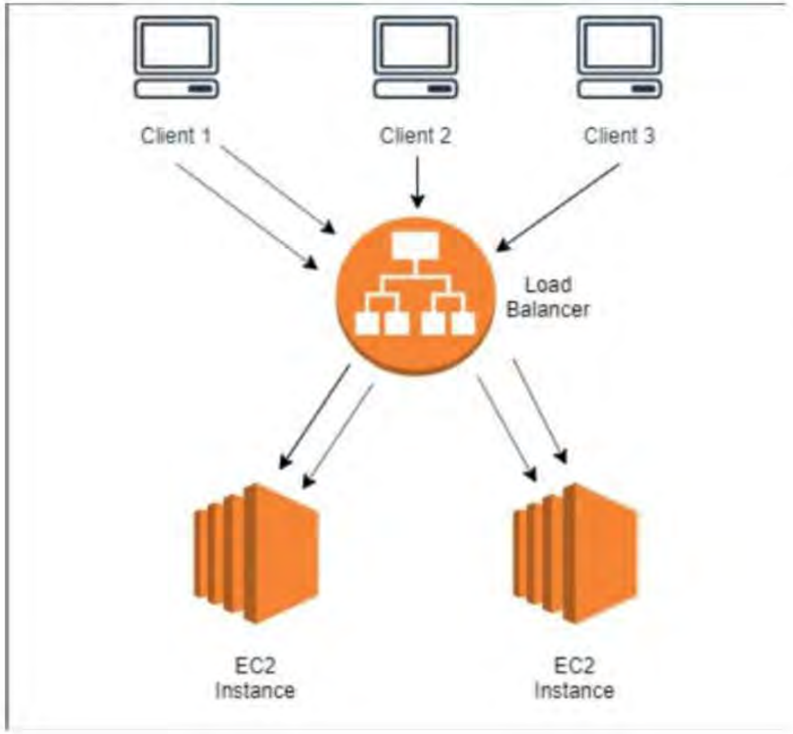


- ➢ Network load balancers (Layer 4) allow you to :
  - ➢ Forward TCP traffic to your instances
  - ➢ Handle millions of request per seconds
  - ➢ Support for static IP or elastic IP
  - ➢ Support Cross Zone Balancing.
- ➢ Network Load Balancers are mostly used for extreme performance and should not be the default load balancer.

Important Notes:

- ➢ CLB, ALB & NLB support SSL certificates and provide SSL termination.
- ➢ All Load Balancers have health check capability.
- ➢ ALB is a great fit with ECS (Docker).
- ➢ All Load Balancers in AWS has a static host name.
- ➢ Do not change and use underlying IP.
- ➢ Network Load Balancer can directly see the client IP.
- ➢ 4xx errors are client induced errors.
- ➢ 5xx errors are application induced errors.
- ➢ 503 means at capacity or no registered target
- ➢ If the Load Balancer can't connect to your application, check your security groups.

Load Balancer Stickiness



- Stickiness is nothing but the same client is always redirected to the same instance behind a load balancer.
- This works for Classic Load Balancers & Application Load Balancers.
- "Cookies" are used for stickiness and those cookies has an Expiry date which you can control.
- Enabling stickiness may bring imbalance to the load over the backend EC2 instances as it will stick to one instance and send traffic only to that instance for certain amount of time

Steps for Creating and Configuring Load Balancers using AWS

Follow these steps to create and configure a load balancer in AWS:

1. Choose the appropriate type of load balancer for your use case, such as Application Load Balancer or Network Load Balancer.
2. Create a target group for your load balancer that specifies the resources you want to distribute traffic to, such as EC2 instances or containers.
3. Configure your load balancer listeners to specify the protocols and ports to use for incoming traffic.
4. Configure your load balancer routing to specify the rules for distributing traffic among your resources.
5. Set up health checks to ensure that your load balancer is sending traffic only to healthy resources.
6. Configure Auto Scaling to automatically adjust the number of resources in your target group based on traffic demand.
7. Manage your load balancer security groups to ensure that only authorized traffic is allowed.

Configuring Listeners, Target Groups, and Routing Rules

Listeners, target groups, and routing rules are the building blocks of an AWS load balancer.

Listeners

A listener is a process that checks for connection requests. Add a listener in the AWS Management Console by selecting your load balancer, going to the Listeners tab, and configuring the protocol and port.

Target Groups

A target group is a group of resources that the load balancer distributes traffic to. Create a target group in the AWS Management Console by navigating to Target Groups, clicking Create target group, and specifying the target type, protocol, port, and VPC.

Routing Rules

Routing rules determine how the load balancer distributes traffic among your target groups. Configure routing rules by selecting your listener, adding rules under the Rules tab, and setting conditions and actions to route traffic to specific target groups.

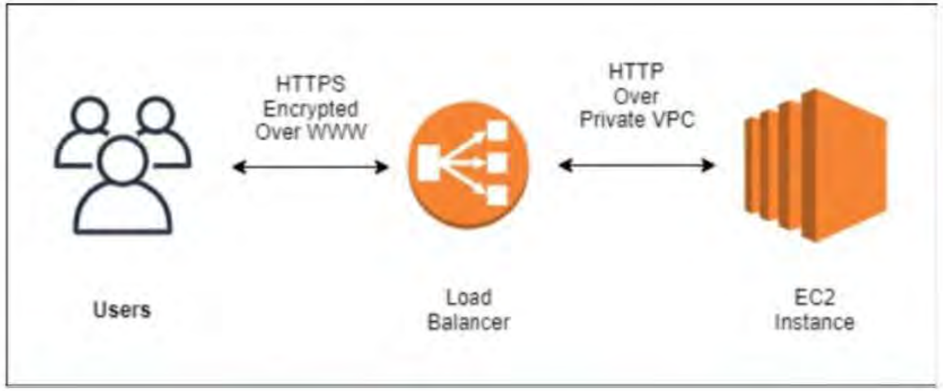
Enabling Health Checks for Automatic Instance Failover

Health checks are a crucial component of an AWS load balancer, as they allow the load balancer to detect when a target is unhealthy and automatically failover to a healthy target. Here's how to enable health checks for your load balancer:

1. Create a target group for your load balancer that specifies the resources you want to distribute traffic to, such as EC2 instances or containers.
2. Configure health checks for the target group to ensure that the load balancer sends traffic only to healthy resources. Health checks can be configured to check the status of the instance, the status of the application running on the instance, or both.
3. Configure your load balancer listeners to specify the target group that the listener should forward traffic to.
4. Configure your load balancer routing rules to specify the actions that the load balancer should take for requests that match the conditions.

Load Balancers SSL Certificates

- The load balancer uses an X.509 certificate (SSL/TLS server certificate)
- You can create upload your own certificates alternatively
- HTTPS listener:
  - You must specify a default certificate
  - You can add an optional list of certs to support multiple domains
  - Clients can use SNI (Server Name Indication) to specify the hostname they reach



Configuring Cross-Zone Load Balancing and Connection Draining

Cross-zone load balancing and connection draining are advanced features of an AWS load balancer that can improve the performance and reliability of your infrastructure. Here's how to configure them:

Cross-Zone Load Balancing

To enable cross-zone load balancing for your load balancer:

1. Open the Amazon EC2 console and navigate to the load balancer that you want to configure.
2. Select the "Attributes" tab and click "Edit".
3. Set "Cross-Zone Load Balancing" to "Enabled".
4. Click "Save" to save your changes.

Configuring Cross-Zone Load Balancing and Connection Draining

Connection Draining

Connection draining is a mechanism that allows the load balancer to complete in-flight requests before terminating a target that has become unhealthy. To enable connection draining for your load balancer:

1. Open the Amazon EC2 console and navigate to the load balancer that you want to configure.
2. Select the "Attributes" tab and click "Edit".
3. Set "Connection Draining" to "Enabled".
4. Specify the amount of time, in seconds, that the load balancer should wait before terminating an unhealthy target.
5. Click "Save" to save your changes.

Integrating with Auto Scaling for Dynamic Scaling of Instances Behind the Load Balancer

Auto Scaling is a powerful tool that allows you to automatically scale the number of instances in your infrastructure up or down based on demand. By integrating your load balancer with Auto Scaling, you can ensure that your infrastructure is always right-sized to handle your workloads. Here's how to configure Auto Scaling with your load balancer:

Step 1: Create an Auto Scaling Group

1. Open the Amazon EC2 console and navigate to the Auto Scaling groups page.
2. Click "Create Auto Scaling group" and follow the on-screen instructions to create your Auto Scaling group.
3. Specify the desired capacity, minimum capacity, and maximum capacity for your group, as well as any other configuration options that you need.
4. Configure your scaling policies to define how your group should scale up and down based on demand.

Integrating with Auto Scaling for Dynamic Scaling of Instances Behind the Load Balancer

Step 2: Register Your Instances with the Load Balancer

To register your instances with your load balancer, you can use either the Amazon EC2 console or the command line interface. Here's how to do it using the console:

1. Open the Amazon EC2 console and navigate to the Instances page.
2. Select the instances that you want to register with your load balancer.
3. Click "Actions", then "Add to Load Balancer".
4. Select the load balancer that you want to register your instances with, then click "Add".

Integrating with Auto Scaling for Dynamic Scaling of Instances Behind the Load Balancer

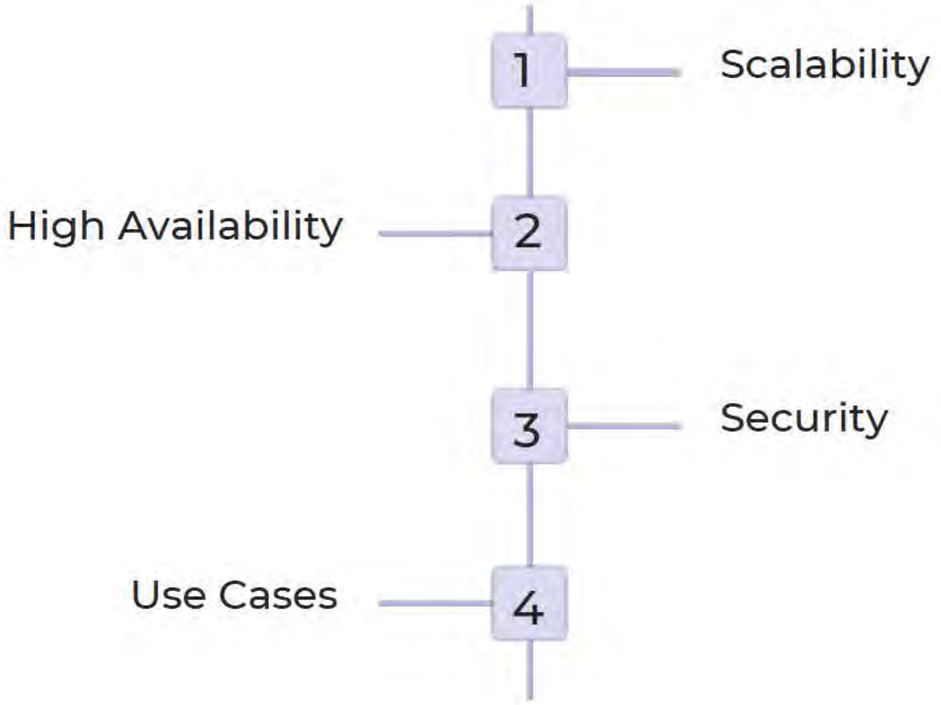
Step 3: Configure Your Load Balancer to Use Your Auto Scaling Group

To configure your load balancer to use your Auto Scaling group, you need to update your target group. Here's how to do it:

1. Open the Amazon EC2 console and navigate to the target groups page.
2. Select the target group that you want to update.
3. Click "Edit", then select your Auto Scaling group from the "Registered" tab.
4. Click "Save" to save your changes.

By integrating your load balancer with Auto Scaling, you can ensure that your infrastructure is always optimized for your workloads. Your Auto Scaling group will automatically adjust the number of instances in your infrastructure based on demand, while your load balancer will distribute traffic evenly across all healthy instances, ensuring that your users receive a high-quality experience.

Benefits and Use Cases of AWS Load Balancers



Pricing of AWS Load Balancers

	Application Load Balancer	Network Load Balancer	Classic Load Balancer
Price per hour Price per	$0.0225	$0.024	$0.025
LCU Load Balancer	$0.008	$0.008	$0.025
Capacity Units (LCU)	2,048 LCUs per hour	1 LCU per hour	1 LCU per hour

Load Balancer pricing is based on the number of hours and Load Balancer Capacity Units (LCUs) used per hour. Application Load Balancers are cheaper than Network and Classic Load Balancers.

Troubleshooting Common Issues with AWS Load Balancers

1 **High Latency**

Check the network and application performance, validate the health of your targets, and explore options for scaling your instances.

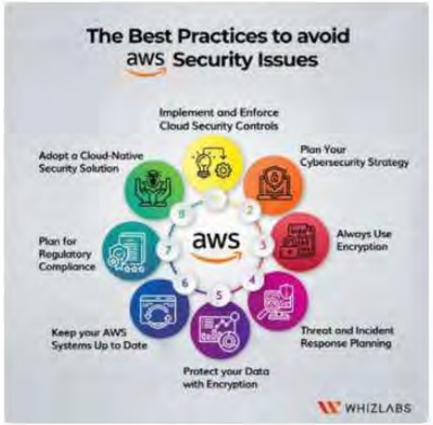
2 **HTTP 503 Errors**

Ensure that the target instances are configured correctly, and check that they're receiving traffic from the Load Balancer. Switch to a healthy target group if needed.

3 **SSL Certificate Issues**

Ensure that the certificate is valid and review the security group rules to enable incoming traffic on appropriate ports.

Best Practices for Implementing AWS Load Balancers


The Best Practices to avoid
aws Security Issues
Implement and Enforce Cloud Security Controls
Plan Your Cybersecurity Strategy
Always Use Encryption
Threat and Incident Response Planning
Protect your Data with Encryption
Keep your AWS Systems Up to Date
Plan for Regulatory Compliance
Adopt a Cloud-Native Security Solution
aws
WHIZLABS

Security

Automation

Logging and Monitoring