ELB

- Introduction to ELB
- Features
- Listeners for Load Balancers
- Related Services
- Benefits
- Use Cases
- Pricing
- Service Health Dashboard

Elastic Load Balancing automatically distributes incoming traffic across multiple Amazon EC2 instances in the cloud.

It enables we to achieve greater levels of fault tolerance in our applications.

- **Elastic Load Balancing automatically distributes incoming traffic across multiple EC2 instances**. We create a load balancer and register instances with the load balancer in **one or more** Availability Zones.
- The load balancer serves as a single point of contact for clients. This enables we to increase the **availability** of our application (HA).
- We can achieve workload management

We can achieve better configuration of scalability

- we can **add and remove EC2 instances** from our load balancer as our needs change, without disrupting the overall flow of information.
- If an EC2 instance **fails**, Elastic Load Balancing **automatically** reroutes the traffic to the remaining running EC2 instances.
- If a failed EC2 instance is restored, Elastic Load Balancing restores the traffic to that instance.

- Elastic Load Balancing can also serve as the first line of **defense against attacks** on our network.
- we can offload the work of **encryption** and **decryption** to our load balancer so that our EC2 instances can focus on their main work

Features of Elastic Load Balancing

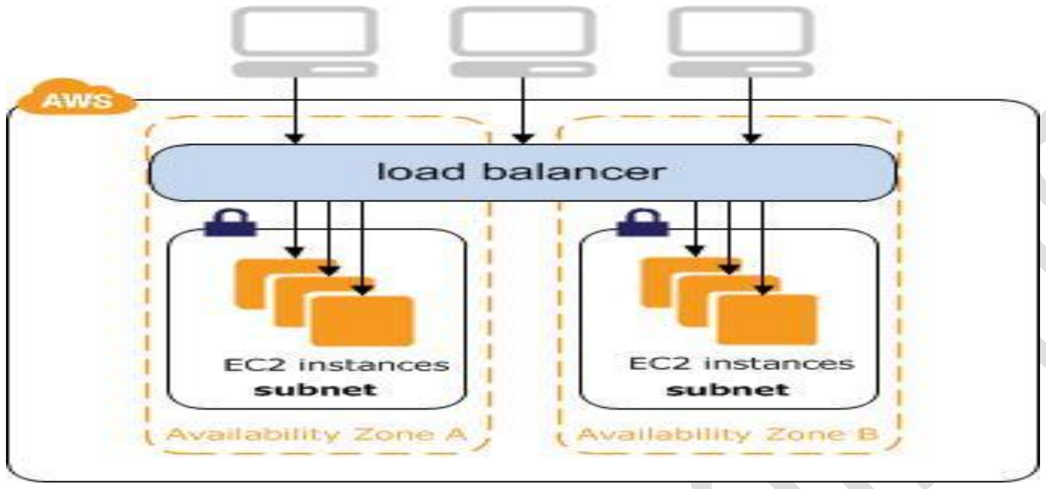
- we can use the operating systems and instance types supported by Amazon EC2. we can configure our EC2 instances to **accept traffic** only from our load balancer.
- we can configure the load balancer to accept traffic using the following **protocols**: HTTP, HTTPS, TCP, and SSL (secure TCP).
- we can configure our load balancer to distribute requests to EC2 instances in multiple **Availability Zones**, minimizing the risk of overloading one single instance. If an entire Availability Zone goes offline, the load balancer routes traffic to instances in other Availability Zones.
- There is **no limit** on the number of connections that our load balancer can attempt to make with our EC2 instances. The number of connections scales with the number of concurrent requests that the load balancer receives.
- we can configure the **health checks** that Elastic Load Balancing uses to monitor the health of the EC2 instances registered with the load balancer so that it can send requests only to the healthy instances.
- we can use end-to-end traffic **encryption** on those networks that use secure (HTTPS/SSL) connections.

- [EC2-VPC] we can create an ***Internet-facing* load balancer**, which takes requests from clients over the Internet and routes them to our EC2 instances, or an ***internal-facing* load balancer**, which takes requests from clients in our VPC and routes them to EC2 instances in our private subnets. Load balancers for a VPC **do not support IPv6** addresses.
- [EC2-Classic] Load balancers in EC2-Classic are always Internet-facing. Load balancers for EC2-Classic support both IPv4 and IPv6 addresses.
- we can monitor our load balancer using **CloudWatch metrics**, access logs, and AWS CloudTrail.

we can **associate** our Internet-facing load balancer with our **domain name**. Because the load balancer receives all requests from clients

Internet-Facing Load Balancers

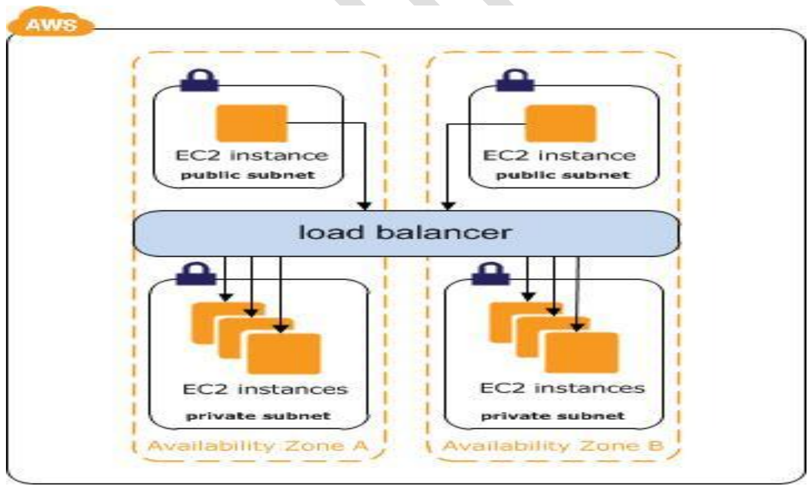
- An Internet-facing load balancer takes requests from clients over the Internet and distributes them across the EC2 instances that are registered with the load balancer



Internal Load Balancers

When we create a load balancer in a VPC, we can make it an internal load balancer or an Internal-facing load balancer.

An internal load balancer routes traffic to our EC2 instances in **private subnets**. The clients must have access to the private subnets.



Create an Internet-facing load balancer and **register** the web servers with it. Create an internal load balancer and register the database servers with it. The web servers receive requests from the Internet-facing load balancer and send requests for the database servers to the internal load balancer.

Related Services

Elastic Load Balancing works with the following services to improve the availability and scalability of our applications.

- **Amazon EC2 —** Virtual servers that run our applications in the cloud.
- **Auto Scaling —** Ensures that we are running our desired number of instances, even if an instance fails, and enables we to automatically increase or decrease the number of instances as the demand on our instances changes.
- If we enable Auto Scaling with Elastic Load Balancing, instances that are launched by Auto Scaling are **automatically registered with the load balancer**, and instances that are terminated by Auto Scaling are automatically **de-registered** from the load balancer.
- **Amazon CloudWatch —** Enables we to monitor the health state of our instances and take action as needed.
- **Amazon Route 53 —** Provides a reliable and cost-effective way to route visitors to websites by translating domain names (such as www.example.com) into the numeric IP addresses (such as 192.0.2.1) that computers use to connect to each other.
- AWS assigns URLs to our AWS resources, such as our load balancers.
- However, we might want a URL that is easy for our users to remember. For example, **we can map our domain name to our load balancer**.
- If we don't have a domain name, we can search for available domains and register them using Amazon Route 53. If we have an existing domain name, we can transfer it to Amazon Route 53

Benefits

Available

- Achieve higher levels of fault tolerance for our applications by using Elastic Load Balancing to automatically route traffic across multiple instances and multiple Availability Zones.

- Elastic Load Balancing ensures that only **healthy Amazon EC2 instances receive traffic by detecting unhealthy instances** and rerouting traffic across the remaining healthy instances.
- If all of our EC2 instances in one Availability Zone are unhealthy, and we have set up EC2 instances in multiple Availability Zones, Elastic Load Balancing will route traffic to our healthy EC2 instances in those other zones

Elastic

- Elastic Load Balancing automatically scales its request handling capacity to meet the demands of application traffic.
- Additionally, Elastic Load Balancing offers **integration** with **Auto Scaling** to ensure that we have back-end capacity to meet varying levels of traffic levels without requiring manual intervention.

Secure

- Elastic Load Balancing works with Amazon **Virtual Private Cloud** (VPC) to provide robust networking and security features.
- We can create an internal (non-internet facing) load balancer to route traffic using **private IP addresses** within our virtual network.
- we can implement a multi-tier architecture using internal and internet-facing load balancers. With this multi-tier architecture, our application infrastructure can use private IP addresses and security groups, allowing us to expose only the internet-facing tier with public IP addresses.
- Elastic Load Balancing provides integrated certificate management and SSL decryption.

Achieving Better Fault Tolerance

- we can build fault tolerant applications by placing our Amazon EC2 instances in multiple Availability Zones. To achieve even more fault tolerance with less manual intervention, we can use Elastic Load Balancing.

DNS Failover for Elastic Load Balancing

- we can use Amazon Route 53 health checking and DNS failover features to enhance the availability of the applications running behind Elastic Load Balancers.
- Using **Route 53 DNS failover**, we can run applications in multiple AWS regions and designate alternate load balancers for failover **across regions**. In the event that our application is unresponsive, Route 53 will remove the unavailable load balancer endpoint from service and direct traffic to an alternate load balancer in another region.

Auto Scaling

- We can integrate ELB with auto scaling to scale up or down to desired level.
- We can set the conditions at auto scaling level to achieve desired goals.

Listeners for our Load Balancer

Before we start using Elastic Load Balancing, we must configure one or more ***listeners*** for our load balancer.

A listener is a process that checks for connection requests. It is configured with a protocol and a port for front-end (client to load balancer) connections, and a protocol and a port for back-end (load balancer to back-end instance) connections.

Elastic Load Balancing supports the following protocols:

HTTP

HTTPS (secure HTTP)

TCP

SSL (secure TCP)

- The HTTPS protocol uses the SSL protocol to establish secure connections over the HTTP layer.
- We can also use the SSL protocol to establish secure connections over the TCP layer.
- If the front-end connection uses TCP or SSL, then our back-end connections can use either TCP or SSL. If the front-end connection uses HTTP or HTTPS, then our back-end connections can use either HTTP or HTTPS.
- Back-end instances can listen on ports 1-65535.
- Elastic Load Balancing provides Secure Sockets Layer (SSL) negotiation configurations, known as *security policy*, to negotiate connections between the clients and the load balancer. When we use HTTPS/SSL for our front-end connection, we can use either a predefined security policy or a custom security policy.
- **we must install an SSL certificate on our load balancer**. The load balancer uses this certificate

Pricing

- If our AWS account is less than 12 months old, we are eligible to use the free tier. The free tier includes 750 hours per month of Amazon EC2 usage, and 750 hours per month of Elastic Load Balancing, plus 15 GB of data processing.

- With Amazon Web Services, we pay only for what we use.
- For Elastic Load Balancing, we pay for each hour or portion of an hour that the **service is running**, and we pay for each **gigabyte of data that is transferred** through our load balancer