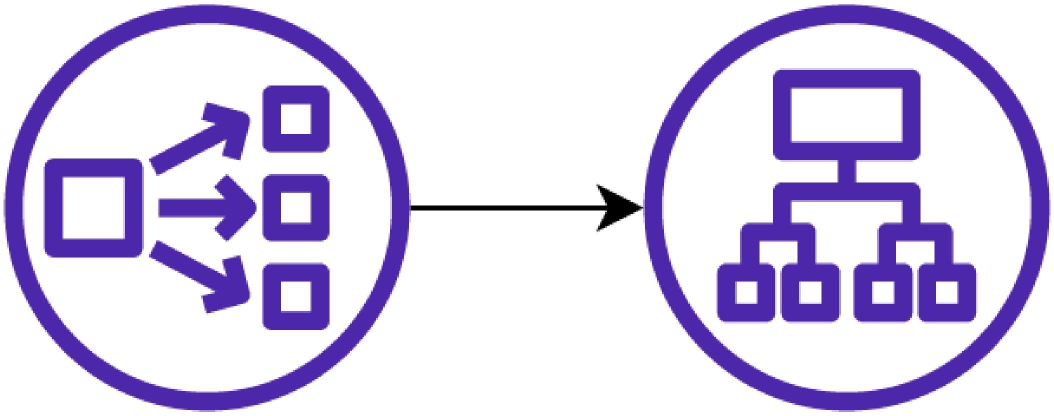

ALB and NLB

What is ELB (Elastic Load Balancer)?

Think of **ELB** as a **traffic cop at a junction**, deciding **which vehicle (user request)** should go to which **lane (server/instance)** based on traffic, availability, and rules.

In cloud architecture, ELB:

- Distributes **incoming traffic** across multiple targets (EC2s, containers, Lambda, etc.)
- Helps achieve **high availability**
- Prevents **overloading** a single server
- Adds a layer of **fault tolerance**

Application Load Balancer (ALB) – Smart Traffic Controller

Use case: Best for **web applications (HTTP/HTTPS)**

Analogy:
Imagine ALB as a **hotel receptionist** who knows exactly **which room a guest should go to based on their request**.

- If a guest asks for **spa**, they get directed to the **spa floor**
- If another asks for **restaurant**, they get routed to the **dining hall**

In tech terms:

- ALB can route traffic **based on URL path, host name, headers, etc.**
- Supports **Layer 7 (Application Layer)**
- Ideal for **microservices** (e.g., /login → auth service, /orders → order service)
- Also supports **WebSocket**, **SSL termination**, **health checks**

Network Load Balancer (NLB) – Fast and Strong

Use case: Best for **high-performance, low-latency, TCP/UDP traffic**

Analogy:
NLB is like a **bouncer at a nightclub**. No talking, no questions — just quickly **lets in the verified guests** through the **right door**, based on their ID.

- Handles **millions of requests per second**
- Works at **Layer 4 (Transport Layer)** and Verifies Health at Layer 4 (TCP/UDP level)
- Supports **TCP, UDP, TLS**
- Great for apps that need **static IPs**, **low latency**, or **high network throughput** (like gaming servers, video streaming, or financial apps)

Security & Health

- Both ALB and NLB support **security groups**, **SSL certificates**, and **target group health checks**
- If one target fails, traffic is **automatically redirected** to healthy targets

Application Load Balancer (ALB) – Layer 7

Use when you need intelligent, content-based routing.

Example 1: Web App with Multiple Services

You have a website with different components:

- /login → Auth service
- /products → Product catalog service
- /cart → Checkout service

ALB can **route requests to different target groups** based on the URL path. Perfect for **microservices** or **monoliths with separate APIs**.

Example 2: Host-based Routing for Multiple Domains

You **host multiple apps on one load balancer:**

- shop.yoursite.com → Shopping app
- admin.yoursite.com → Admin dashboard
- api.yoursite.com → Backend API

ALB supports host-based routing**, directing traffic based on the domain** name.

Network Load Balancer (NLB) – Layer 4

Use when you need ultra-fast, high-performance traffic handling with minimal logic.

Example 1: TCP-based App or API

You run a **custom TCP service**, like a **gaming server** or **VoIP application**, which doesn't use HTTP.

NLB is ideal for **raw TCP/UDP traffic**. It can handle **millions of connections per second** with **very low latency**.

Example 2: Banking or Payment System

You're building a **financial service** that must:

- Use TLS directly with clients (not offloaded at LB)
- Maintain static IPs for firewall rules
- Require fast connection handling

NLB supports **TLS passthrough**, **static IP**, and **Elastic IP**, making it ideal for **strict security or compliance needs**.

ALB vs NLB in a Real-Life Web App (Like Amazon)

Let’s say we’re building a **web app like Amazon**:
It’s a secure website, it uses **HTTPS**, and users can browse products, add items to cart, and proceed to **payment**.

Step 1: User Browsing the Website

- User visits your site: https://www.mysite.com
- They view products, add to cart, check profile, etc.

This is all done using **HTTPS (HTTP over TLS)**
So we use **Application Load Balancer (ALB)**

- ALB routes /products, /cart, /login to different microservices
- ALB understands **URLs, headers, paths** – it’s smart

ALB is perfect for **web apps, APIs, and anything using HTTP/HTTPS**

Step 2: User Clicks “Proceed to Payment”

Now here’s where it gets interesting 👇

Option 1: You Use an External Payment Gateway (like Razorpay, Stripe, PayPal)

- User is redirected to https://secure.razorpay.com or similar
- Payment is done **outside your AWS setup**
- Still HTTPS, so it's handled via ALB

In this case, **NLB is not involved**
ALB is enough to manage the whole traffic flow

Option 2: You Have Your Own Payment Engine

Some companies (like banks or large fintech firms) build **in-house payment systems**.

- Here, your frontend sends secure data to **backend payment servers**
- These may use **raw TCP**, **custom protocols**, or **TLS passthrough**
- They don't use HTTP — just **pure encrypted communication**

This is where **NLB (Network Load Balancer)** comes in.

- NLB handles **TCP, UDP, and TLS traffic** and It forwards traffic quickly and securely to your backend payment processors

Example: Real-time banking, internal payment apps, gaming servers, or stock trading platforms

Thank you