Interview Questions & Answers on Load Balancing

1. What is load balancing, and why is it important?

Answer: Load balancing is the process of distributing incoming network traffic across multiple backend servers to ensure efficient utilization, prevent overload, and improve system availability.

- **Ensures High Availability**: Prevents system downtime by redirecting traffic in case of server failure.
- **Optimizes Resource Utilization**: Spreads requests evenly to avoid overloading a single server.
- **Improves Performance**: Reduces latency by routing traffic to the best-performing server.
- **Enhances Scalability**: Supports horizontal scaling by adding more servers as demand grows.
- **Increases Fault Tolerance**: Redirects requests if a server fails, ensuring system reliability.

---

2. Explain the difference between Layer 4 and Layer 7 load balancing.

Answer:

Layer 4 Load Balancing (Transport Layer)

- Operates at the **network transport level** (TCP/UDP).
- Distributes traffic based on **IP addresses and port numbers** without inspecting request content.
- Faster and more efficient for **simple traffic distribution**.
- Examples: **AWS Network Load Balancer (NLB), HAProxy (L4 Mode)**.

Layer 7 Load Balancing (Application Layer)

- Works at the **application level** (HTTP/HTTPS).
- Routes requests based on **content, headers, cookies, or URL paths**.
- Supports **advanced features** like SSL termination, caching, and authentication.
- Examples: **AWS Application Load Balancer (ALB), Nginx, Traefik**.

Key Difference:
Layer 4 is **faster** but **less flexible**, while Layer 7 is **intelligent** but **adds overhead**.

---

3. How does a load balancer handle high availability and failover?

Answer:

- **Health Checks**: Continuously monitors server health using **ping, HTTP checks, or TCP checks**.
- **Automatic Failover**: If a server becomes unresponsive, the load balancer redirects traffic to healthy servers.
- **Redundancy**: Can be deployed in **active-active** or **active-passive** configurations.
- **Session Persistence**: Maintains user sessions across multiple requests to prevent disruptions.
- **Global Load Balancing**: Uses **GeoDNS** or **Anycast Routing** to distribute traffic across data centers.

---

4. Compare Round Robin and Least Connections strategies.

Answer:

Round Robin

- Sends requests to servers in a **circular order** (Server 1 → Server 2 → Server 3 → Repeat).
- **Best for:** Uniform workloads and **servers with equal capacity**.
- **Limitations:** Can overload servers if they have different processing power.

Least Connections

- Sends requests to the server with the **fewest active connections**.
- **Best for:** Scenarios where some requests take longer than others (e.g., database queries).
- **Limitations:** Requires tracking active connections, increasing computational overhead.

Key Difference:
Round Robin is **simpler** but assumes equal server capacity, while Least Connections **dynamically** adjusts based on load.

---

5. What are the advantages of Weighted Load Balancing?

Answer:
Weighted Load Balancing assigns **different priorities** to servers based on their capacity.

- **Better Resource Utilization**: High-performance servers receive more traffic.
- **Custom Traffic Distribution**: Allows fine-tuned control over request routing.
- **Supports Heterogeneous Environments**: Works well when servers have **different processing power**.
- **Examples**: Weighted Round Robin, Weighted Least Connections.

---

6. When would you use a software load balancer over a hardware one?

Answer:

Software Load Balancer

- Runs as an application on standard hardware.
- **Pros**:
    - Cost-effective and flexible.
    - Easily scalable (deployed in containers or VMs).

    - Supports **open-source solutions** like Nginx, HAProxy, Envoy.

- **Cons**:
    - Requires server resources.
    - May introduce additional latency under heavy traffic.

Hardware Load Balancer

- A dedicated device optimized for handling large-scale traffic.
- **Pros**:
    - High performance with dedicated hardware acceleration.
    - Built-in security features (e.g., **DDoS protection**).
- **Cons**:
    - Expensive and less flexible.
    - Harder to scale dynamically.

Use Case:

- Use **software load balancers** for **cloud-native applications**.
- Use **hardware load balancers** for **enterprise-level, high-traffic systems**.

---

7. How would you design a scalable load balancing solution for a large e-commerce site?

Answer:

1. **Use Multiple Load Balancers**:
    - Deploy **primary and secondary load balancers** for redundancy.
    - Distribute traffic globally using **DNS-based load balancing**.
2. **Choose the Right Load Balancer**:

- Use **Layer 7** load balancing for dynamic content.
- Use **Layer 4** load balancing for database connections.

3. **Implement Load Balancing Strategies**:
   - **Round Robin** for static content servers.
   - **Least Connections** for dynamic request handling.
4. **Ensure High Availability**:
   - Use **auto-scaling groups** to handle traffic spikes.
   - Implement **health checks** to detect and bypass failed servers.
5. **Optimize Performance**:
   - Enable **caching** (e.g., CDN) to reduce load on backend servers.
   - Use **Gzip compression and minification** to reduce response sizes.

---

8. What factors should be considered when choosing a load balancing strategy?

Answer:

1. **Traffic Pattern**:
   - If traffic is **evenly distributed**, use **Round Robin**.
   - If requests vary in complexity, use **Least Connections**.
2. **Server Capacity**:
   - If servers have different capacities, use **Weighted Load Balancing**.
3. **Session Persistence**:
   - If user sessions must be maintained, use **Sticky Sessions**.
4. **Performance vs. Complexity**:
   - Layer 4 is **faster but less flexible**.

- Layer 7 is **slower but allows intelligent routing**.

5. **Scalability Needs**:
    - For **cloud-native applications**, use **cloud-based load balancers** (e.g., AWS ELB).
    - For **on-premises applications**, use **software or hardware-based solutions**.

---

9. How does a load balancer improve security?

Answer:

1. **DDoS Protection**
    - Detects and blocks **malicious traffic spikes**.
    - Some hardware load balancers provide **built-in DDoS mitigation**.
2. **SSL Termination**
    - Offloads **SSL decryption** from backend servers.
    - Ensures **secure connections** with HTTPS.
3. **Access Control**
    - Restricts access using **firewalls and IP whitelisting**.
4. **Application Firewall Integration**
    - Prevents **SQL injection, cross-site scripting (XSS), and other attacks**.
5. **Rate Limiting**
    - Limits requests per second to prevent abuse.