AWS Elastic Load Balancer (ELB)

Amazon Web Services (AWS) Elastic Load Balancer (ELB) is a service that automatically distributes incoming traffic across multiple targets, such as EC2 instances, containers, and IP addresses. It helps ensure high availability and fault tolerance for applications by balancing the load across multiple instances and by routing traffic only to healthy targets.

Key Types of Elastic Load Balancers

1. **Application Load Balancer (ALB)**
    - **Layer 7 Load Balancer**: Operates at the application layer (HTTP/HTTPS), making routing decisions based on content, such as host names, paths, or headers.
    - **Advanced Routing**: Supports advanced request routing features like URL-based routing, host-based routing, and query string-based routing.
    - **WebSockets Support**: Supports WebSockets for real-time communication.
    - **Security**: Works with SSL/TLS certificates to provide HTTPS endpoints.
    - **Target Types**: Can route traffic to EC2 instances, containers, IP addresses, and Lambda functions.
    - **Health Checks**: Regularly checks the health of targets and routes traffic only to healthy instances.
    - **Sticky Sessions**: Can maintain sticky sessions (session affinity) to ensure that user sessions remain on the same server.
    - **Use Case**: Ideal for modern microservices-based architecture or applications that need advanced traffic routing based on HTTP content.
2. **Network Load Balancer (NLB)**
    - **Layer 4 Load Balancer**: Operates at the transport layer (TCP/UDP), making routing decisions based on IP and port.
    - **High Throughput**: Handles millions of requests per second with ultra-low latencies, suitable for high-performance applications.
    - **Static IP Address**: Can provide a static IP address or assign an Elastic IP (EIP) per Availability Zone.
    - **TLS Termination**: Supports TLS offloading to improve performance and reduce the load on backend targets.
    - **Connection Stickiness**: Sticky sessions based on source IP are supported.
    - **Health Checks**: Continuously checks the health of targets and directs traffic only to healthy targets.
    - **Use Case**: Best suited for high-performance applications that need to handle TCP/UDP traffic with low latency, like real-time gaming, video streaming, or financial applications.
3. **Gateway Load Balancer (GWLB)**
    - **Transparent Gateway**: Combines the functionalities of a load balancer and a gateway to simplify the integration of third-party security appliances like firewalls, intrusion detection systems (IDS), and deep packet inspection (DPI) systems.

   - **Operates at Layer 3**: Routes traffic based on IP addressing.
   - **Auto Scaling**: Automatically scales based on demand to ensure consistent performance of your security appliances.
   - **Use Case**: Primarily used when integrating third-party security or monitoring solutions into your AWS environment.

4. **Classic Load Balancer (CLB)**
   - **Legacy Load Balancer**: Supports both Layer 7 (HTTP/HTTPS) and Layer 4 (TCP/SSL) load balancing.
   - **Simpler Features**: Does not have the advanced routing features of the ALB or the high-performance capabilities of the NLB.
   - **Health Checks**: Performs health checks on targets and routes traffic only to healthy targets.
   - **Use Case**: Best suited for applications that were built using the older EC2-Classic network and don't need advanced routing or high-performance features.

Key Features of ELB

1. **Auto Scaling**: ELB works seamlessly with Auto Scaling to automatically adjust the number of instances based on traffic load. This ensures high availability and cost efficiency by scaling up during peak times and down during low traffic periods.
2. **Health Monitoring**: ELB performs continuous health checks on registered targets and routes traffic only to healthy instances. Unhealthy instances are automatically removed from the rotation until they recover.
3. **Cross-Zone Load Balancing**: Allows traffic to be distributed evenly across instances in multiple Availability Zones, ensuring fault tolerance and high availability.
4. **Security Features**:
   - **SSL/TLS Termination**: ELB can handle SSL/TLS encryption, offloading this burden from the backend servers. This feature improves the performance of backend servers by reducing the CPU load for SSL decryption.
   - **Access Logs**: Provides detailed logging information about incoming requests, such as IP address, request path, and response time.
   - **Security Groups**: Works with AWS security groups to control access to the load balancer and backend instances.
   - **AWS WAF (Web Application Firewall) Integration**: Can be integrated with AWS WAF to protect against common web exploits.
5. **Load Balancer Stickiness**:
   - Also known as session affinity, stickiness allows the load balancer to send requests from the same client to the same target. This can be essential for certain applications that require user sessions to remain on the same server.
6. **Global Accelerator Integration**: ELB can be integrated with AWS Global Accelerator to provide a low-latency global network path for your applications, improving global reach and performance.

7. **Elasticity and Fault Tolerance**: ELB automatically scales with the traffic, providing elasticity to handle increases in traffic while also distributing the traffic across multiple AZs, increasing fault tolerance.
8. **Custom Health Checks**: ELB allows users to define custom health check endpoints, allowing the service to test the health of backend services at more meaningful application-level endpoints rather than simple ping-based checks.
9. **Monitoring and Logging**:
   - **CloudWatch Metrics**: ELB automatically sends metrics to Amazon CloudWatch, such as request counts, latency, and error rates.
   - **Access Logs**: Can be configured to capture detailed information about requests to your load balancer, helping with debugging and compliance.
   - **AWS CloudTrail**: Provides logging for all API calls made to the ELB service for auditing purposes.

Pricing Model

- **Pay-per-Usage**: AWS ELB is priced based on the amount of load balancer capacity (measured in Load Balancer Capacity Units or LCU) and the amount of data processed. Specific pricing includes:
  - **LCU per hour**: Different types of load balancers have different LCU costs based on metrics like new connections, active connections, and processed data.
  - **Data Processed**: Charged based on the amount of data processed through the load balancer.

Use Cases

1. **Microservices Architecture**: ALB is ideal for applications built using microservices, as it supports path-based routing and host-based routing, allowing traffic to be directed to different backend services based on the request content.
2. **Web Applications**: ALB can route traffic to multiple EC2 instances, ensuring high availability and fault tolerance for web applications. With sticky sessions and SSL/TLS termination, ALB is suitable for web apps that require HTTPS and session affinity.
3. **Real-time Applications**: NLB is ideal for real-time, latency-sensitive applications that need to handle millions of connections per second, such as financial systems, online gaming, and IoT devices.
4. **Security Appliances**: GWLB is commonly used for integrating security appliances like firewalls or IDS into AWS architectures, where it scales automatically and simplifies appliance management.

Best Practices

1. **Use Cross-Zone Load Balancing**: This feature distributes traffic evenly across all instances in different Availability Zones, ensuring better fault tolerance and utilization of instances.

2. **Enable Health Checks**: Always configure health checks to remove unhealthy instances from the traffic routing, ensuring that only healthy instances serve your application.
3. **SSL/TLS Offloading**: Offload SSL/TLS termination at the load balancer to reduce CPU usage on your backend servers.
4. **Monitor Performance**: Use CloudWatch metrics and access logs to monitor the performance of your load balancer and backend instances. This can help in identifying bottlenecks or issues early.
5. **Leverage Security Groups and AWS WAF**: Ensure that you apply the principle of least privilege with security groups and use AWS WAF to protect against common web exploits.