Elastic Load Balancer

Elastic Load Balancing distributes incoming application traffic across multiple EC2 instances, in multiple Availability Zones. This increases the fault tolerance of your applications.
Load balancer can be made interna. The DNS servers resolve the DNS name of your load balancer to the private IP addresses of the load balancer nodes for your internal load balancer
Listeners
Port/Protocol on which LoadBalancer listens
For each requests ELB maintains 2 connections - from client to ELB and from ELB to backend
Connection can be reused.

Supports Layer 4 (TCP) & 7 (HTTP/S)

Connection drainig, max 1h

There's no additional cost for access logs (only S3 storage)
Access logs are disabled by default

Classic Load Balancer

```
- Works with EC2
- Simple balancing - no rules
- Stops serving traffic to an instance when unhealthy

- Cross-zone balancing
    - By default, the load balancer distributes traffic evenly across the
Availability Zones that you enable for your load balancer. To distribute
traffic evenly across all registered instances in all enabled Availability
Zones, enable cross-zone load balancing on your load balancer
- Health checks
    - Stops routing when health check fails
    - Resume routing when health check passes
    - Types:
        - Ping
        - TCP connection attempt
        - Page request

- Sticky Sessions based on cookie (app or elb cookie)
- One EC2 instance - one target
- Does not suport SNI
Connection draining - keep connection for given time when instance
deregistering or unhealthy
Proxy Protocol - when using SSL or TCP it adds soruce IP
```

Application Load Balancer - REGIONAL

```
- Target Groups
- multiple target groups
- Rules (path/host)
- Supports WAF
- Multiple apps on the same host (IP - PORT pairs)
- AutoScalling group can be attached to Target Group
- Perfect for microservices: You can use a microservices architecture to
structure your application as services that you can develop and deploy
independently. You can install one or more of these services on each EC2
instance, with each service accepting connections on a different port. You can
use a single Application Load Balancer to route requests to all the services
for your application. When you register an EC2 instance with a target group,
you can register it multiple times; for each service, register the instance
using the port for the service.
- Request Tracking: You can use request tracing to track HTTP requests from
clients to targets or other services. When the load balancer receives a request
from a client, it adds or updates the X-Amzn-Trace-Id header before sending the
request to the target.
```

AutoScaling - REGIONAL

```
Minimum selected options:
Launch configuration name, Amazon Machine Image (AMI), and instance type
### Groups
    Your EC2 instances are organized into groups so that they can be treated as
a logical unit for the purposes of scaling and management
    - Single Launch Configuration
    - Minimum instances 1
    - Created in specific Availability Zones
    - Health Checks - when to remove instance:
        - By default EC2
        - Can use ELB health check
### Launch Configuration
    Your group uses a launch configuration as a template for its EC2 instances.
When you create a launch configuration, you can specify information such as the
AMI ID, instance type, key pair, security groups, and block device mapping for
your instances
    - Auto Scalling group has only one Launch Configuration
    - You can't modify it, only create new one
    - You can ask for Spot Instances
###  Scaling plans
    A scaling plan tells Auto Scaling when and how to scale. For example, you
can base a scaling plan on the occurrence of specified conditions (dynamic
scaling) or on a schedule.
    - Manual
        - Edit ASG and set new desired
    - Dynamic
        - Simple and Step Policy
            - Add instances if alarm triggers
            - Remove instances if alarm triggers
            - Steps allow you to add more or less depends on threshold breach
```

        - Target Tracking
            - Tries to keep application on target (CPU utilization etc)
        - SQS
            - based on number of messages in message queue (using CloudWatch). Then uses Step/Simple policy
Default Termination Policy

Termination Policy
    1. Auto Scaling determines whether there are instances in multiple Availability Zones. If so, it selects the Availability Zone with the most instances and at least one instance that is not protected from scale in. If there is more than one Availability Zone with this number of instances, Auto Scaling selects the Availability Zone with the instances that use the oldest launch configuration.

    2. Auto Scaling determines which unprotected instances in the selected Availability Zone use the oldest launch configuration. If there is one such instance, it terminates it.

    3. If there are multiple instances that use the oldest launch configuration, Auto Scaling determines which unprotected instances are closest to the next billing hour. If there is one such instance, Auto Scaling terminates it.

    4. If there is more than one unprotected instance closest to the next billing hour, Auto Scaling selects one of these instances at random.

StandBy - you can temporarly remove instance from the group

- Lifecycle hooks - Auto Scaling lifecycle hooks enable you to perform custom actions by pausing instances as Auto Scaling launches or terminates them. For example, while your newly launched instance is paused, you could install or configure software on it.
    1. Auto Scaling puts the instance into a wait state (Pending:Wait or Terminating:Wait). The instance is paused until either you tell Auto Scaling to continue or the timeout period ends
    2. You can then:
        - Define a CloudWatch Events target to invoke a Lambda function when a lifecycle action occurs
        - Define a notification target for the lifecycle hook. Auto Scaling sends a message to the notification target
        - Create a script that runs on the instance as the instance starts

- Auto Scalling Processes:
    - Launch - Adds a new EC2 instance to the group, increasing its capacity
    - Terminate - Removes an EC2 instance from the group, decreasing its capacity.
    - HealthCheck - Checks the health of the instances
    - ReplaceUnhealthy - Terminates instances that are marked as unhealthy and later creates new instances to replace them

```
    - AZRebalance - Balances the number of EC2 instances in the group across
the Availability Zones in the region
    - ScheduledActions -
    - AddToLoadBalancer

    - Processes (triggers) can be suspendend. You have to then reenable them
and trigger manually
```