AWS ELASTIC LOAD BALANCING (ELB)

Elastic Load Balancing automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, containers, IP addresses, and Lambda functions.
It can handle the varying load of your application traffic in a single Availability Zone or across multiple Availability Zones.
Elastic Load Balancing Working process:
Load Balancer accepts traffic from clients and routes to registerd instances in om=ne or more zones. It also monitors the health of registered targets. If any one is unhealthy it route the request to other target using its Private IP.

Components:
Listener is a process that checks for connection requests.
Elastic Load Balancing offers **three types of load balancers**:
1. Application Load Balancer (HTTP/HTTPS): An Application Load Balancer makes routing decisions at the application layer (HTTP/HTTPS), supports path-based routing, and can route requests to one or more ports on each container instance in your cluster.
Application Load Balancers support dynamic host port mapping.
Process:
When a load balancer node receives a request then Listner apply the rule based on priority and select the target using routing algorithm.
A routing algorithm is a set of step-by-step operations used to direct Internet traffic efficiently. When a packet of data leaves its source, there are many different paths it can take to its destination. The routing algorithm is used to determine mathematically the best path to take

2. Network Load Balancer (TCP/SSL): A Network Load Balancer makes routing decisions at the transport layer (TCP/SSL). It can handle millions of requests per second.
Network Load Balancers support dynamic host port mapping
Process:
When Load balancer receives the connection, first it selects target using flow hash algorithm and routes TCP connections from different clients with different source ports to targets
flow hash algorithm is the process where information flow with high speed to application with out any data modifications by encryption.

3. Classic Load Balancer: A Classic Load Balancer makes routing decisions at either the transport layer (TCP/SSL) or the application layer (HTTP/HTTPS).
Classic Load Balancers currently require a fixed relationship between the load balancer port and the container instance port.
Process:
When load balancer node receives a request from client and selects the target node using TCP listerners and HTTP and HTTPS Listeners. http—80 https--443

DIFF B/W APP LOAD BALANCER VS N/W LOAD BALANCER
1. The Application Load Balancer works at the Application Layer (Layer 7 of the OSI model). **Vs** The network load balancer works at layers 3 & 4.

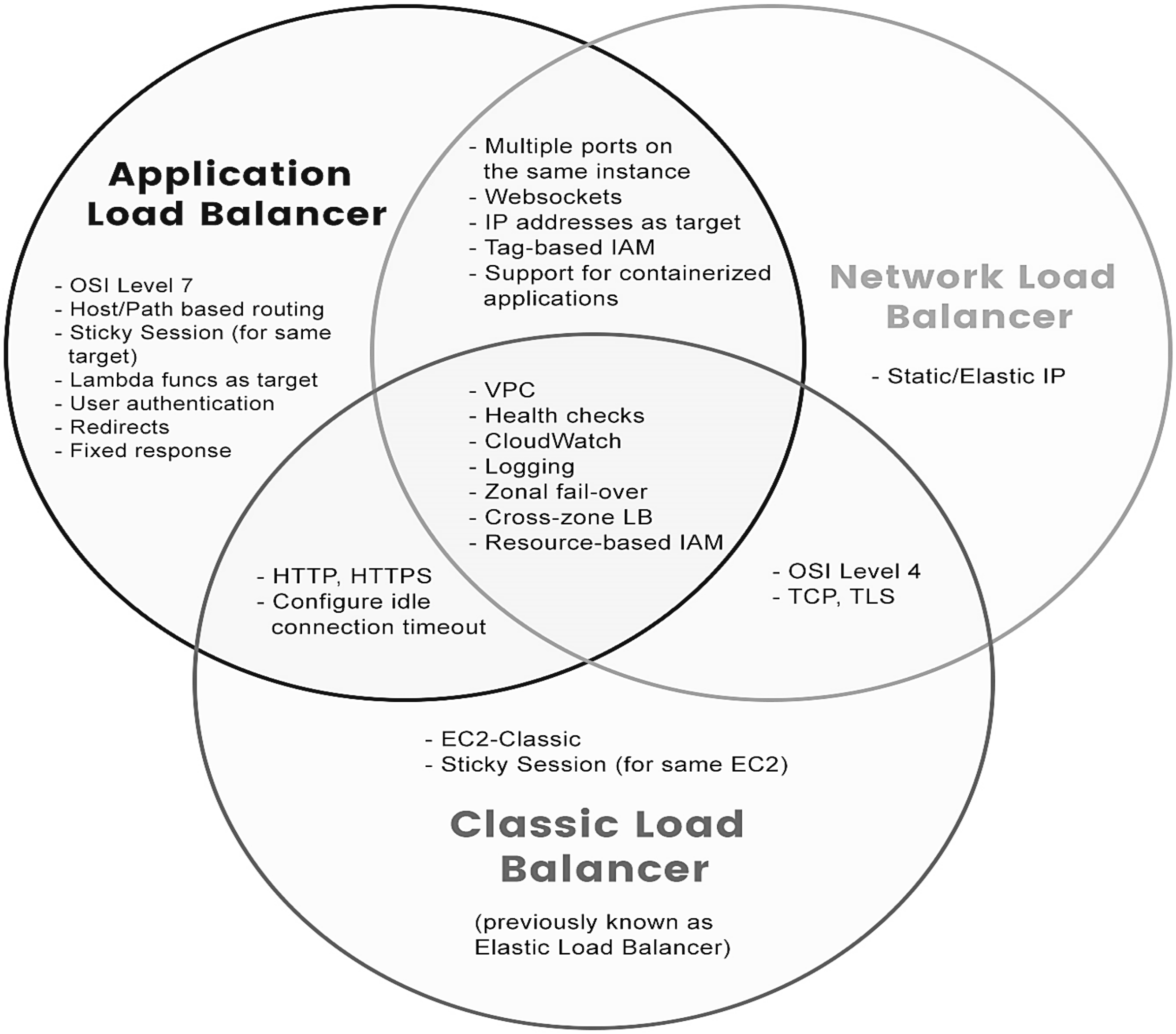
2. App Load Balancer uses routing algorithm. **Vs** N/W Load Balancer uses flow hash algorithm

3. The application load balancer examines the contents of the HTTP request header to determine where to route the request called content based routing. **Vs** The network load balancer just forward requests.

4. Application balancing can assure availability of the application. This is because it bases on HTTP and HTTPS requests. **Vs** Network load balancing cannot assure availability of the application. This is because it bases its decisions solely on network and TCP-layer variables.

5. An application load balancer will determine availability based on successful HTTP GET and also the verify the content based on the input parameters. **Vs** Network load balancer will determine “availability” of a server using ICMP ping, or three-way TCP handshake.

6. Multiple Applications with same ip then application load balancer will differentiate between the two applications by examining the application layer data available to it.
Multiple Applications with same ip then network load balancer will not differentiate between app-A and app-B unless both using different ports.

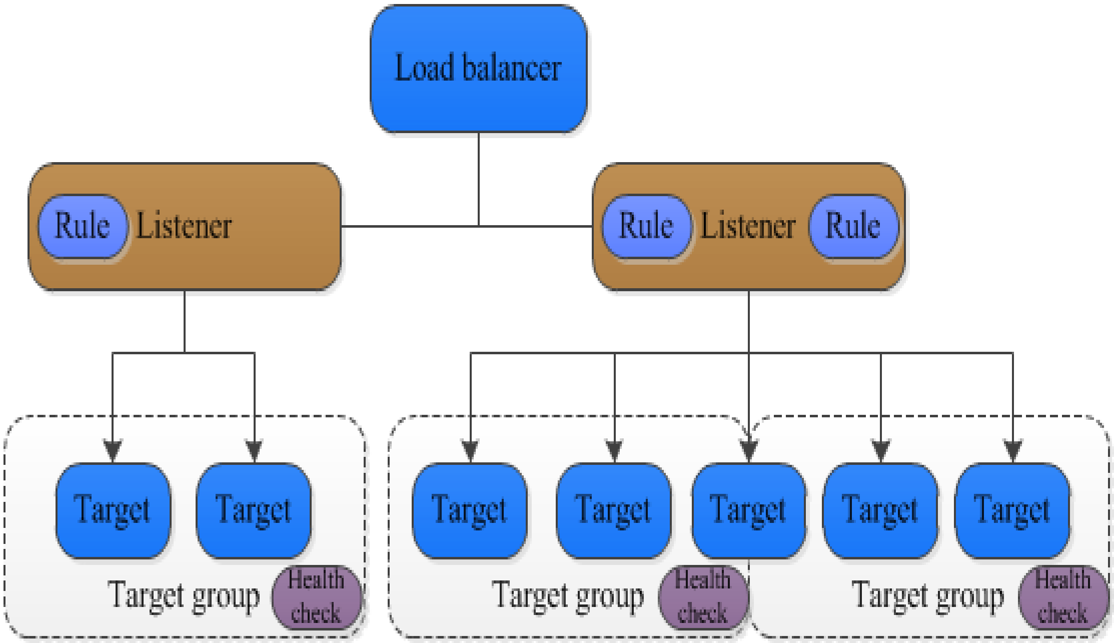


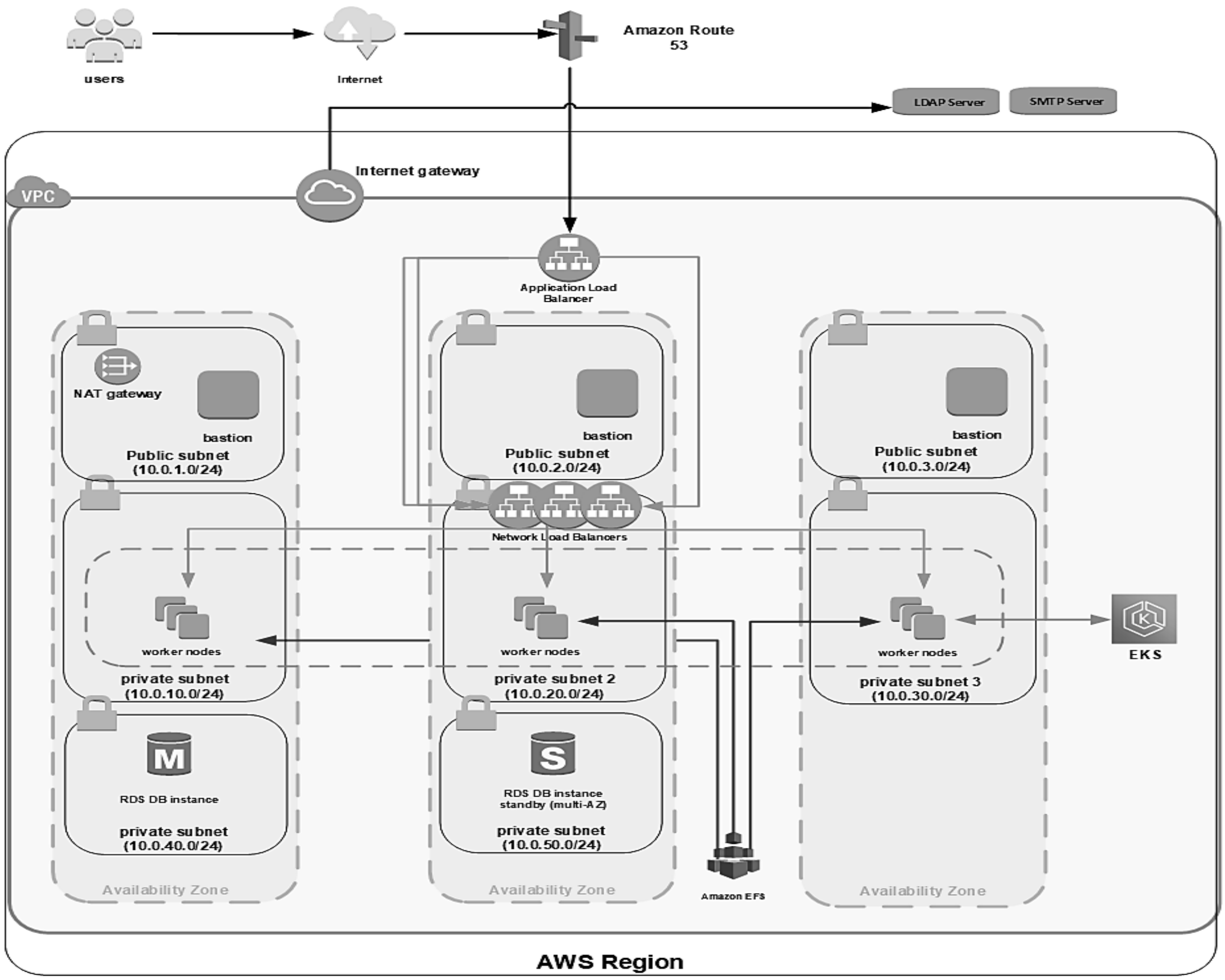
Load Balancers Scheme types:
1. **Internal Load balancer:** nodes of an internal load balancer have only private IP addresses which is used to route requests in that VPC only.
2. **Internet-facing Load Balancer:** nodes of an internet-facing load balancer have public IP addresses which is used to route requests.

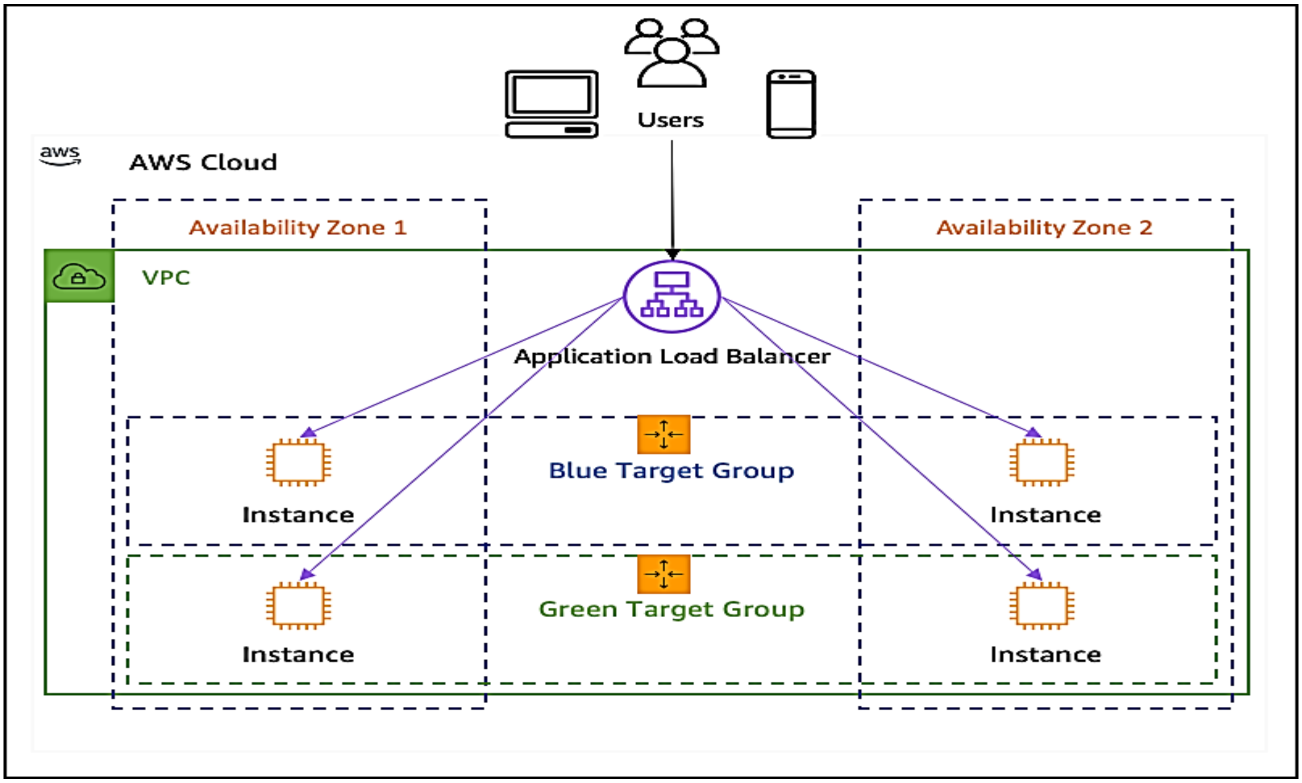
Note: If your application has multiple tiers, you can design an architecture that uses both internal and internet-facing load balancers.
Note: Application uses web servers and application server with internet-facing load balancer and internal load balancer can be used respectively.

https://aws.amazon.com/blogs/networking-and-content-delivery/how-to-securely-publish-internet-applications-at-scale-using-application-load-balancer-and-aws-privatelink/

https://docs.microfocus.com/itom/SMAX:2019.11/EKSALB




users
Internet
Amazon Route 53
LDAP Server
SMTP Server
Internet gateway
VPC
Application Load Balancer
NAT gateway
bastion
Public subnet (10.0.1.0/24)
bastion
Public subnet (10.0.2.0/24)
bastion
Public subnet (10.0.3.0/24)
Network Load Balancers
worker nodes
private subnet (10.0.10.0/24)
worker nodes
private subnet 2 (10.0.20.0/24)
worker nodes
private subnet 3 (10.0.30.0/24)
EKS
RDS DB instance
private subnet (10.0.40.0/24)
RDS DB instance standby (multi-AZ)
private subnet (10.0.50.0/24)
Amazon EFS
Availability Zone
Availability Zone
Availability Zone
AWS Region


Users
AWS Cloud
Availability Zone 1
Availability Zone 2
VPC
Application Load Balancer
Instance
Blue Target Group
Instance
Instance
Green Target Group
Instance

AWS Elastic Beanstalk

AWS Elastic Beanstalk is an easy-to-use service for deploying and scaling web applications and services developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache, Nginx, Passenger, and IIS.
You simply upload your application, and Elastic Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.
To use Elastic Beanstalk, you create an application, upload an application version in the form of an application .war file to Elastic Beanstalk

It supports: 1. Apache tomcat 2. Apache HTTP Server 3. Nginx server 4. Passenger 5. docker

Details on changes between platform versions can be found on the AWS Elastic Beanstalk Release Notes page.
When you create an environment, Elastic Beanstalk provisions the resources required to run your application. AWS resources created for an environment include one elastic load balancer (ELB in the diagram), an Auto Scaling group, and one or more Amazon Elastic Compute Cloud (Amazon EC2) instances.
Host Manager runs on each EC2 Instance. It is responsible for
Deploying app, Monitoring app logs and app server, aggregate metrics and storing files in s3.
We can define more than one security group if required.

Terms Related to Elastic Bean Stack:
1. **Runtime:** time required to run your app code
2. **Platform:** Software required to run the app
3. **Platform Version:** Specific number for software. Versions can be of two
a) Supported---with supported components where all components have not reached end of life.
b) Retired ---A platform version with one or more retired components, which have reached their End of Life.
4. **Platform Branch:** platform versions sharing specific (typically major) versions of some of their components, platform branch can be in one of the following states:
a) Supported: supported components. It receives ongoing platform updates, and is recommended for use in production.
b) Beta: A preview, pre-release platform branch. It's experimental in nature.
c) Deprecated: A platform branch with one or more deprecated components. Not recommended in production.
d) Retired: A platform branch with one or more retired components. It doesn't receive platform updates anymore
5. **Platform Update**: A release of new platform versions that contain updates to some components Platform updates have several levels:
a) Major update: changes that are incompatible with existing platform versions need to modify app to run correctly.
b) Minor update: An update that adds functionality that is backward compatible with an existing platform version.
c) Patch update: An update that consists of maintenance releases like bug fixes, security updates, and performance improvements.