www.e-commerce...

Publishing the website

aws

EC2

customer

VPC

!!!!!!!!!!

Handle Request

- Web Server work in EC2 machine (nginx)

8 gb 8gb 8gb

EC2 EC2

IP:x IP:y IP:z

1. Creating an instance just like the same -Ok

Current Resource Management

2. Manage the sequence of instance

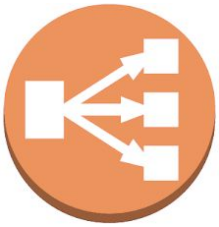
3. Which IP (DNS) will we browse

4. Is the instance healthy now.
5. Failover to instance in the same region

Launch Template

Launch Configuration

TARGET GROUP



Load Balancer

HEALTHY THRESHOLD

INTERVAL= 10 SEC

Unhealthy

CONSECUTIVE SUCCESSFUL HEALTH CHECK

1 2 3

Healthy

UNHEALTHY THRESHOLD

Healthy

CONSECUTIVE FAILED HEALTH CHECK

1 2

Unhealthy

Timeout =5 sec

Launch Template

1

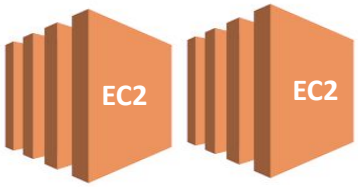
3

Register instance

+++

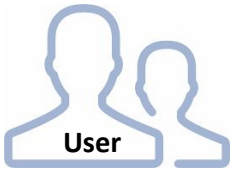
4

Load Balancer

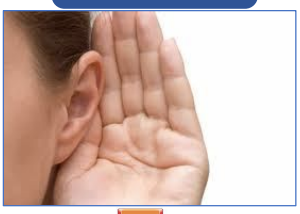


Target Group = EC2 Pool=Reserve

2



Listener



Sequence: **Round Robin**
Least Outstanding Requests

Health Check

Configuration

Choose one of them

Assign DNS name:

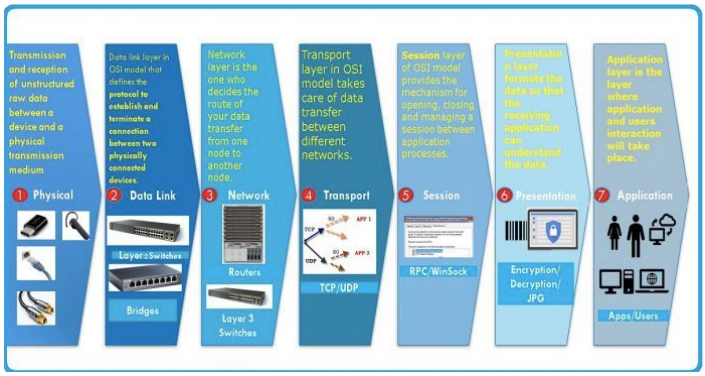
Ip:x
DNS of ELB

Ip:y
DNS of ELB

Ip:z
DNS of ELB

Remaning Problems

- Manage the sequence of instance → Sequence: **Round Robin**, **Least Outstanding Requests**
- Which IP (instance) will we browse on Public Internet → DNS name:
- Is the instance healthy now. → Health Check 



OSI (pronounced as separate letters) is short for Open System Interconnection. **OSI** is an ISO standard for worldwide communications that defines a networking framework for implementing protocols in seven layers.

The application layer (layer-7) is the layer where application and users interaction will take place. Some real-world examples are end-user applications like outlook, File Explorer, Chrome, or some other applications using different protocols HTTP (web communication), HTTPS (Secured web communication), SMB (file transfer).

The **transport** layer (layer-4) in the OSI model takes care of data transfer between different networks.

NLB provides **static IP** address, ALB does not. So use it when you require a static IP for your LB. Similarly, it is the only balancer that can use **Elastic IP addresses**.

- Use NLB when you require **end-to-end SSL encryption**. ALB will always terminate SSL connection, which may be not desired due to strict security requirements.
- NLB is the only balancer type that can be used for API Gateway VpcLink or VPC PrivateLink technologies.
- NLB does not have Security Groups.

When compared to an Application Load Balancer a simple explanation goes like this: Network Load Balancer is used anywhere where the application behind the balancer doesn't work over HTTP(S), but uses some other protocol. Including, but not limited to:

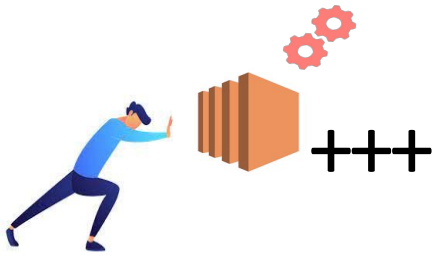
- Legacy applications that implement custom protocol.
- NTP servers.
- SMTP server.
- Database servers.
- MQTT brokers.
- High performance queue servers (ActiveMQ, RabbitMQ, ZeroMQ etc.).
- Message processing applications (think Kafka and Co.).

The other difference between the two is important because network load balancing cannot assure availability of the *application.* This is because it bases its decisions solely on network and TCP-layer variables and has no awareness of the application at all.

Generally a network load balancer will determine “availability” based on the ability of a server to respond to ICMP ping, or to correctly complete the three-way TCP handshake.

An application load balancer goes much deeper, and is capable of determining availability based on not only a successful HTTP GET of a particular page but also the verification that the *content* is as was expected based on the input parameters.

Load Balancer-Stand-alone



REGISTERING

Target Group = Pool

HealthCheckProtocol	The protocol the load balancer uses when performing health checks on targets. The possible protocols are HTTP and HTTPS. The default is the HTTP protocol.
HealthCheckPort	The port the load balancer uses when performing health checks on targets. The default is to use the port on which each target receives traffic from the load balancer.
HealthCheckPath	The ping path that is the destination on the targets for health checks. Specify a valid URI (/*path*?*query*). The default is /.
HealthCheckTimeoutSeconds	The amount of time, in seconds, during which no response from a target means a failed health check. The range is 2–120 seconds. The default is 5 seconds if the target type is instance or ip and 30 seconds if the target type is lambda.
HealthCheckIntervalSeconds	The approximate amount of time, in seconds, between health checks of an individual target. The range is 5–300 seconds. The default is 30 seconds if the target type is instance or ip and 35 seconds if the target type is lambda.
HealthyThresholdCount	The number of consecutive successful health checks required before considering an unhealthy target healthy. The range is 2–10. The default is 5.
UnhealthyThresholdCount	The number of consecutive failed health checks required before considering a target unhealthy. The range is 2–10. The default is 2.
Matcher	The HTTP codes to use when checking for a successful response from a target. You can specify values or ranges of values between 200 and 499. The default