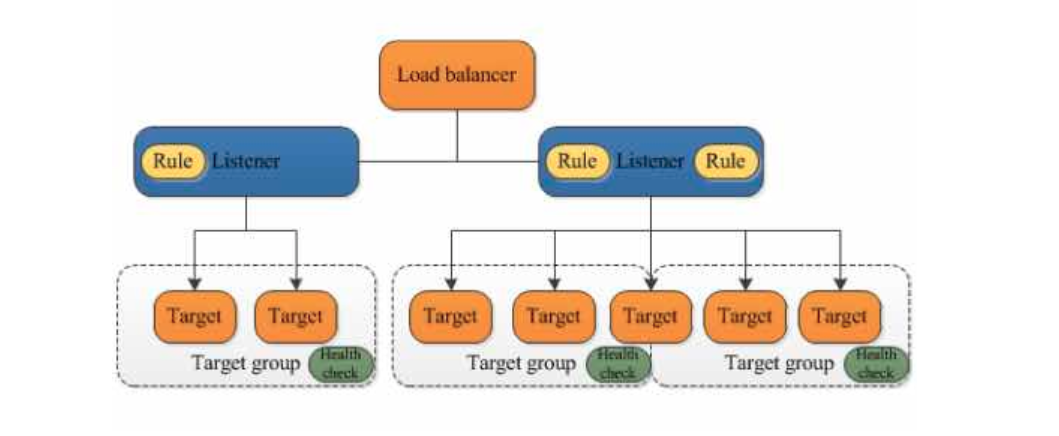
ELASTIC LOAD BALANCING

* + Distributes incoming application or network traffic across multiple targets, such as EC2 instances, containers, and IP addresses, in multiple AZs
  + Upon creation must specify a public subnet from at least two AZs
* GENERAL FEATURES
  + Accepts incoming traffic and routes requests to registered requesters
  + Monitors the health of targets and routes traffic to only healthy targets
  + Enable deletion protection to prevent load balancers from being deleted accidentally. (The default is disabled)
  + Deleting an ELB will not delete the instances registered to it
  + **Cross Zone Load Balancing** - when enabled, each load balancer node distributes traffic across the registered targets in all enabled AZs
  + Supports SSL offloading which is a feature that allows the ELB to bypass the SSL termination by removing the SSL-based encryption from the incoming traffic
* THREE TYPES

APPLICATION LOAD BALANCER (7th Layer -- Applicaiton Layer)

* Functions at the application layer, the seventh layer of the Open Systems Interconnection (OSI) model.
* **Open Systems Interconnection Model**
  + **Layer 1: Physical Layer**
    - Responsible for the transmission and reception of unstructured raw data between a device and a physical transmission medium. Examples would be USB connection
  + **Layer 2: Data Link Layer**
    - Provides node to node data transfer - a link between two directly connected nodes.
    - It detects and possibly corrects errors that may occur in the physical layer
    - It defines the protocol to establish and terminate a connection between physically connected devices
    - IEEE 802 data link layer has 2 sublayers
      * Medium access control ( MAC) layer - responsible for controlling how devices in a network gain access to a medium and permission to transmit data
      * Logical Link Control – Responsible for identifying and encapsulating network layer protocols, and controls error checking and frame synchronization
  + **Layer 3: Network Layer**
    - Provides the functional and procedural means of transferring variable length data sequences (called packets) from one node to another connected in “different networks”
    - NETWORK
      * A medium into which many nodes can be connected, on every node has an address and which permits nodes connected to it to transfer messages to other nodes connected to it by merely providing the content of a message and the address of the destination node
  + **Layer 4: Transport Layer**
    - Provides the functional and procedural means of transferring variable-length data sequences from a source to a destination host, while maintaining the quality of service functions
  + **Layer 5: Session Layer**
    - Controls the dialogues (connections) between computers
    - It establishes, manages and terminates the connection between the local and remote applications
  + **Layer 6: Presentation Layer**
    - Establishes context between application-;ayer entities, in which the application layer entities may use different syntax and semantics if the representation service provides a mapping between them
  + **Layer 7: Application Layer**
    - The layer that is closest to the user which means that this layer and the user interact directly with the application
* Allows HTTP and HTTPS
* **Components**
  + **Load Balancer** which serves as the single point of contact for clients
  + **Listener** which checks for connection requests from the clients. You must define a default rule for each listener that specifies a target group, condition, and priority
  + **Target Group** routes requests to one or more registered targets. You can register a target with multiple target groups, and configure health checks on a per target group basis

.

* Benefits
  + Path-based and host-based routing
  + Route requests to multiple applications on the same EC2 ( such as containers )
  + Registering targets by IP address, including targets outside of the VPC
  + Support for containerized applications
  + Monitors the health of each service independently
* Cross-zone load balancing is always enabled
* You can also specify Lambda functions are targets to serve HTTP(S) requests
* Supports Load balancer-generated cookies only for sticky sessions.
* Supports
  + HTTP/2
  + Websockets
* Monitoring
  + CloudWatch - retrieve statistics about data points for your load balancers and targets as an ordered set of time-series data, known as metrics
  + Access Logs - capture detailed information about ht requests made toyour load balancer and stre them as log files in S3
  + Request Trackign - track HTTP request
  + CloudTrail Logs - capture detailed information about the cells made to the elastic load balancing API and store them as log files in S3.

NETWORK LOAD BALANCER (4th Layer -- Transport Layer)

* Functions at the 4th layer. Uses **TCP** and **UDP** connections
* **TCP - Transmission Control Protocol**
  + A main protocol of the internet protocol suite
  + TCP provides reliable , ordered, and error-checked delivery of a stream of bytes between applications runnion on hosts communicating via an IP network
  + At the lower levels of the protocol stack, due to [network congestion](https://en.wikipedia.org/wiki/Network_congestion), traffic [load balancing](https://en.wikipedia.org/wiki/Load_balancing_(computing)), or unpredictable network behaviour, IP packets may be [lost](https://en.wikipedia.org/wiki/Packet_loss), duplicated, or [delivered out of order](https://en.wikipedia.org/wiki/Out-of-order_delivery). TCP detects these problems, requests [re-transmission](https://en.wikipedia.org/wiki/Retransmission_(data_networks)) of lost data, rearranges out-of-order data and even helps minimize network congestion to reduce the occurrence of the other problems. If the data still remains undelivered, the source is notified of this failure. Once the TCP receiver has reassembled the sequence of octets originally transmitted, it passes them to the receiving application. Thus, TCP [abstracts](https://en.wikipedia.org/wiki/Abstraction_(computer_science)) the application's communication from the underlying networking details.
  + BASICALLY -- Fixes the errors of data transmission
* **UDP - User Datagram Protocol**
  + A main protocol of the internet protocol suite
  + Uses a simple connectionless communication model with a minimum of protocol mechanisms
* At least 1 subnet must be specified when creating this type of load balancer but the recommended number is 2
* Components:
  + A load balancer serves as the single point of contact for clients
  + A listener checks for connection requests from clients
  + A target group routes requests to one or more registered targets. You can register a target with multiple target groups. You can configure health checks on a per targets group basis.
* Benefits
  + Ability to handle volatile workloads and scale to millions of requests per seconds.
  + Static IP addresses for the load balancer, or assign one Elastic IP address per subnet enabled for the load balancer
  + Support for registering targets by IP address
  + Support for routing requests to multiple applications on a single EC2 instance ( register each instance or IP address with the same target group using multiple ports).
  + Support for containerized applications
  + Support for monitoring the health of each service independently
* Cross-zone load balancing is disabled by default
* Network Load Balancers support connections from clients over inter-region VPC peering, AWS managed VPN, and third party VPN solutions
* YOu can deploy services that rely on UDP protocol, such as Authentication and Authorization Logging, DNS and IOT behind a Network Load Balancer.
* Offers multi-protocol listeners, allowing you to run applications such as DNs that rely on both TCP and UDP protocols on the same port behind a network load balancer
* You cannot enable or disable AZ for a network load balancer after you create it.
* Network Load Balancers use proxy protocol version 2 to send additional connection information such as the source and destination

**Differences between the two main load balancers**

<https://medium.com/awesome-cloud/aws-difference-between-application-load-balancer-and-network-load-balancer-cb8b6cd296a4>

If you are deploying docker containers and using a load balancer to send network traffic to them EC2 container service provides a tight integration with ALB and NLB so you can keep your load balancers in sync as you start updating and stop containers across your fleet.

**Network Load balancer**

This is the distribution of traffic based on network variables, such as IP address and destination ports. It is layer 4 TCP and below and is not designed to take into consideration anything at the application layer such as content type, cookie data, custom headers, user location, or the application behavior. It is context-less, caring only about the network layer information contained within the packets it is directing this way and that.

This is a TCP Load Balancer only that does some NAT magic at the VPC level. It uses Elastic IPs, so it has a static endpoint unlink ALB and CLB ( by default, contact support if this is a requirement for your CLB or ALB. Each Target can be on different ports

**Application Load Balancer**

This is the distribution of requests based on multiple variables, from the network layer to the application layer. It is context-aware and can direct requests based on any single variable as easily as it can a combination of variables. APplications are load balanced based on their peculiar behavior and not solely on server ( operating system or virtualization layer ) information.

This is feature filled L7 load balancer, HTTP and HTTPS listeners only. Provides the ability to route HTTP and HTTPS traffic based upon rules, host based or path based. Like an NLB each target can be on different ports. Even supports HTTP/2. Configurable range of health checks status codes ( CLB only supports 200 ok for HTTP health checks.