**ELASTIC LOAD BALANCER [ELB]**

**Scalability**

* Scalability means than an application / system can handle greater loads by adapting.
* There are 2 kinds of scalaibility:
  + Vertical scalability
* Means Increasing the size of the instance.[t2.micro to t2.large]
* Its very common for non distributed systems such as database.
* RDS & ElastiCache are services that can scale vertically.
* There is limit how much you can vertically scale [hardware limit]
  + Horizontol scalability [scale out /in[
* Means increasing the number of instances /system for ur app.
* It implies distributed systems.
* Common for web applications /modern applications.
* Auto scaling group [ASG]
* Load Balancer

**High Availability**

* Usually goes hand in hand with horizontal scaling
* Run instances for the same application across multi AZ.
* Means running your application / system in atleast 2 Data centers (==Aavailability zone).
* The goal of High availability is to survive a data center loss.
* High availability can be active.[for horizontal scaling]
* High availability can be pasive.[for RDS Mmulti AZ]
* Auto scaling group multi AZ
* Load balancer multi AZ

**Elastic Load Balancer**

Load balancer:

* Load balancers are servers that forward internet traffic to multiple servers [EC2 Instances] downstream.
* Spread load across downtstream instances.
* Expose single point of access DNS your application
* Seamlessly handles failure of downstream instances.
* Do regular health checks to your instances.
* Provide SSL termination (https) for your websites.
* Hhigh availability across zones
* Enforce stickiness with cookies.
* Seprate public traffic from private traffic.

**Elastic Load Balancer [EC2 load balancer]**

Elastic Load Balancing distributes incoming application or network traffic across multiple targets, such as Amazon EC2 instances, containers, and IP addresses, in multiple Availability Zones. Elastic Load Balancing scales your load balancer as traffic to your application changes over time. It can automatically scale to the vast majority of workloads.

**Why EC2 load balancer**

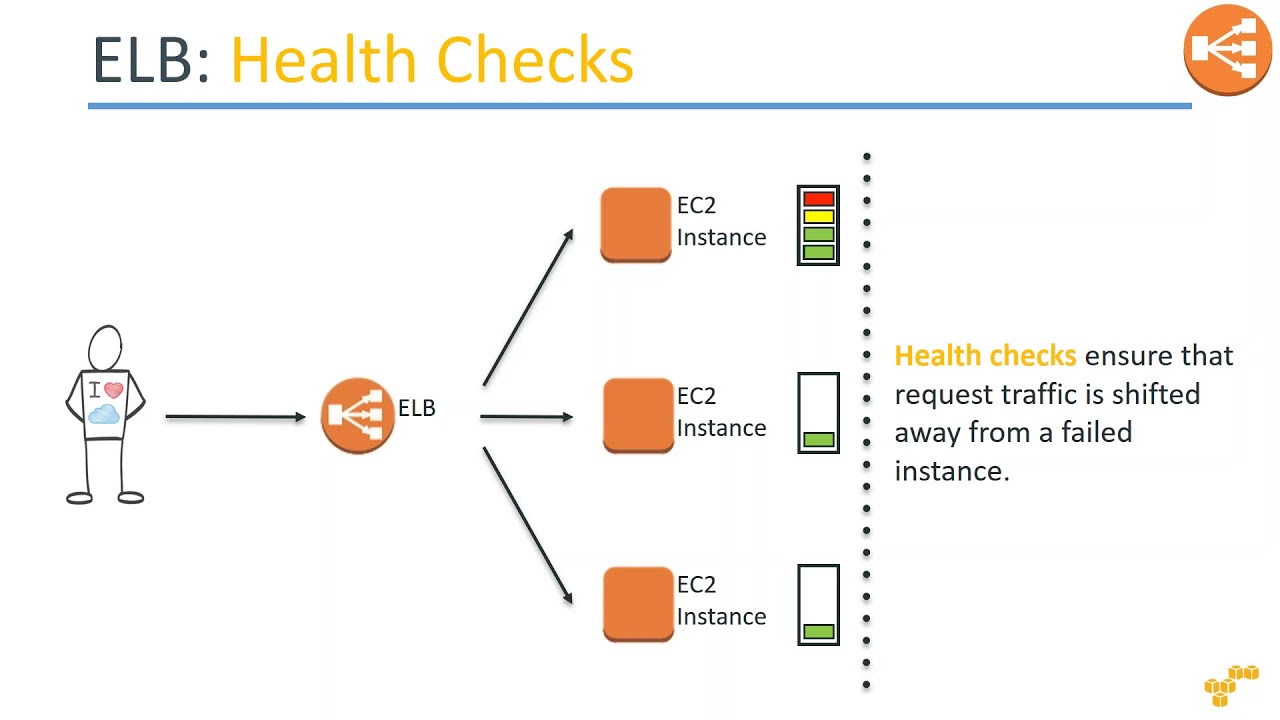
* ELB is a managed load balancer
* AWS takes care of upgrades maintanence high availability
* It costs less to setup your own load balancer.

**Load balancer benefits**

* A load balancer distributes workloads across multiple compute resources, such as virtual servers.
* Using a load balancer increases the availability and fault tolerance of your applications.
* You can add and remove compute resources from your load balancer as your needs change, without disrupting the overall flow of requests to your applications.
* You can configure health checks, which monitor the health of the compute resources, so that the load balancer sends requests only to the healthy ones

**Health Checks**

* They enable the load balancer to know if instances it forward traffic to are available to reply to requests.
* The health check is done on a port and route [/heath is common]
* If the response is not 200 ok then the instance is unhealthy.



**Types of Load Balancer on AWS [3kinds of managed Load balancer]**

* Classic Load balancer [v1 old generation] 2009] HTTP HTTPS TCP
* Application Load Balancer [ V2 new generation] 2016] HTTP HTTPS WEBSOCKET
* Network Load Balancer [V2 new generation] 2017] TCP TLS[secure tcp] & UDP
* You can set up internal [private] or external [public] ELBs.

**Load Balancer Security Group**

USER

ELB

EC2

HTTPS/HTTP

Source: Anywher [0.0.0.0]

HTTP

Source : Loadbalancer

**Troubleshooting**

* 4xx errors are client induced errors
* 5xx errors are application induced errors
* Load balancer Errrors 503 means at capacity or no registered target.
* If LB cant connect to your application check your security group.

**Monitoring**

ELB access logs will log all access requests [so you can debug per request]

Cloud watch metrics will give you aggregate statistics [ex: connections count]

**Classic Load Balancer [CLB] [V1]**

* Supports TCP [layer 4] & HTTP/HTTPS[layer 7]
* Health checks are either TCP or HTTP based
* Fixed host name [DNS]

**Application Load Balancer [V2] [Layer 7]**

* Supports HTTP HTTPS & Websocket
* Load Balancing to multiple HTTP applications across machines [target group]
* Load Balancing to multiple applications on the same machine [ex: container]
* Support redirect from HTTP to HTTPS
* Routing tables to different target groups:

1. Routing based on path in URL [example.com/users & example.com/posts]
2. Routing based on hostname in URL [one.example.com & other.example.com]
3. Routing based on Query, String, Headers.

[example.com/users?id=123&order=false]

* ALB are great fit for microservices & container based applications [ex:docker & ecs]
* Has a port mapping feature to redirect to a dynamic port in ECS.
* In comparison, we need multiple Classic Load Balancer per application.
* Target Groups

1. EC2 instances [can be managed by an Auto Scaling group] HTTP
2. ECS tasks [managed by ECS itself] HTTP
3. Lambda functions –HTTP request is transmitted into a JSON event.
4. Ip address must be private ip
5. ALB can route to multiple target groups
6. Health checks are at the target group level.

* Fixed/static DNS/hostame name
* The application servers don’t see the IP of the client directly

The true IP of the client is inserted in the header X-forwarded-For

We can also get the port [X-forwarded-port and porto (X-forwarded-Proto]

**Hands On**

**EC2 Dashboard………Load Balancer…..Create**

Basic configuration

Listeners [Load Balancer port number]

Availability zone

Security Group

Target Group: {u can create multiple target group..each can hv multiple instanc}

Name

Type: Instance / IP / Lamda function

Protocol:

Port:

Health checks:

Protocol: HTTP

Path: /

**Select the instances & add to registered**

**Create**

You can able to edit ther target group to create new target group.

**Rules**

* Select created Loadbalancer…Listeners….view/edit rules
* Based on path /new…….then forward to first target grp
* Based on hostname then forward to 2nd target grp
* Based on path /test then forward to 3rd target group

**Network Load Balancer [NLB V2] [Layer 4]**

* Supports TCP [TLS] & UDP
* Handles millions of request per seconds [High performance]
* Less latency ~100ms [vs 400ms for ALB]
* NLB has one static IP per AZ and supports assigning Elastic IP [helpful for whitelisting specific IP] you can u have 2 entry ip
* Type: Instance / IP
* Here you need to attach the security group manually post NLB creation.

Choose NLB and select Target group…Targets..choose one of your instance.. choose inbound security group enable port from anywhere…..means EC2 can access from anywhere through only NLB.

**Load Balancer Stickiness**

* It is possible to implement stickiness so that the same client is always redirected to the same instance behind a load balancer.
* This works for classic load balancer & ALB.
* The cookie used for stickiness has an expiration date you control
* Use case: make sure the user dosent lose his session data
* Enabling stickiness may bring your imbalance to the load over the backend EC2.

**Cross Zone Load Balancing**

AZ2

AZ2

AZ1

EC2

EC2

EC2

EC2

EC2

EC2

* With cross zone load Balancing: each load balancer instances distributes evenly across all registered instances in all AZ.

**Classic Load Balancer**

* Disabled by default
* No charges for inter AZ data if enabled.

**Application Load Balancer**

* Always ON [can’t be disabled
* No charges for inter AZ data

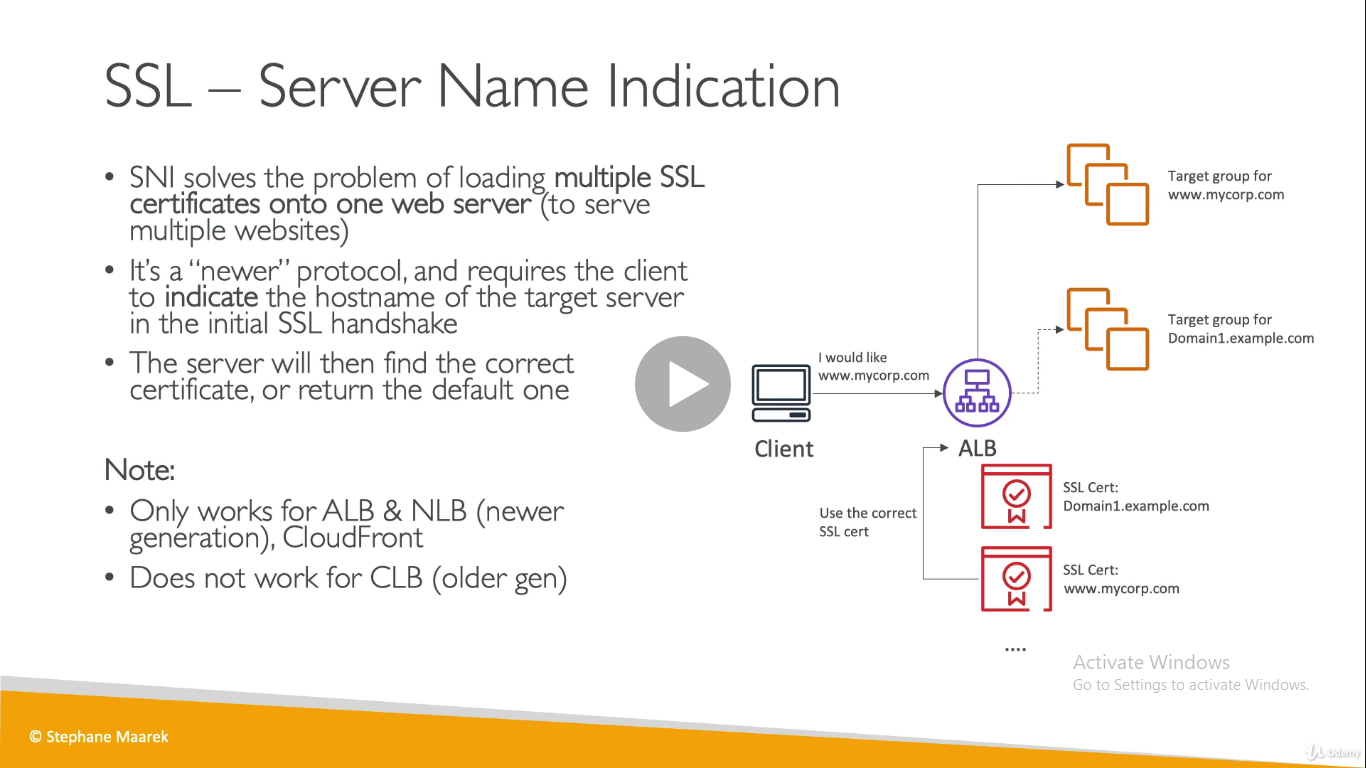
**Network Load Balancer**

* + Disabled by default
  + You pay charges for inter AZ data

**SSL/TLS basics**

* An SSL Certificate allows traffic between your clients and your load balancer to be encrypted in transit. [in-flight encryption]
* SSL refers to secure socket layer used to encrypt connections
* TLS refers to Transport Layer Security which is a new version
* Public SSL certificates are issued by Certificate Authorities [CA]
* Comodo Symantec GoDaddy Global sign Digicert Letsencrypt….
* SSL certificates have an expiration date [you set] and must be renewed.
* The load balancer uses an x.509 ceritificate [SSL/TLS certificate]
* You can manage certificates using ACM [AWS Certificate Manager]
* You can create upload you own certificates alternatively.
* HTTPS listener:
  + You must specify a default certificate
  + You can add an optional list of certs to support multiple domains
  + Clients can use SNI [Server Name Indication] to specify the hostnamethey reach.
  + Ability to specify a security policy to support older versions of ssl /tls [legacy clients]

**Server Name Indication [SNI]**



**ELB : SSL certificates**

* CLB : supports only one SSL certs

**Application Load Balancer (v2)**

* Multiple listeners with multiple ssl certs
* Uses SNI to make it work.

**Network Load Balancer**

* Multiple listeners with multiple ssl certs
* Uses SNI to make it work.

**Hands On**

* Select the created Load Balancer…..Listeners….add/edit listeners ..
  + Protocol [https]: port [443]
  + Forward to : selct the target grp
  + Default ssl crt: ACM/ IAM/ import

**ELB : Connection Draining**

* It’s the time to complete “in-flight request” while the instance is de-registering or unhealthy
* If CLB: then its called connection draining
* If Traget grp: then its called Deregistration delay [for ALB & NLB]
* Stops sending new request to the instance which is de-registering
* Deregistration delay is bet 1 to 3600 sec, by def is 300 sec.
* Can be disabled [set to value 0]
* Set to a low vaule if your request are short

