

Scalability & High Availability

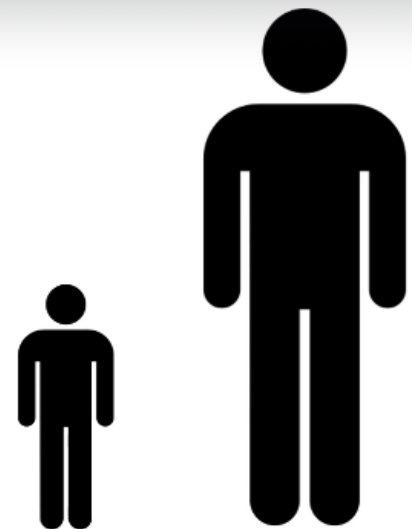
- Scalability means that an application / system can handle greater loads by adapting.
- There are two kinds of scalability:
 - Vertical Scalability
 - Horizontal Scalability (= elasticity)
- Scalability is linked but different to High Availability
- Let's deep dive into the distinction, using a call center as an example

1.

2. Vertical scalability:

Vertical Scalability

- Vertical Scalability means increasing the size of the instance



junior operator

senior operator

3.

4. In the context of call center...a junior op will take less call...but a senior op can take more call

5. Similarly in here..a t2.micro will run less tasks compared to t2.large

6. basically...

Sure, here is an example of vertical scalability in AWS:

- You have an Amazon EC2 instance running a web application. As the application becomes more popular, you need to increase its capacity to handle the increased traffic.
- You can vertically scale the instance by increasing its CPU and memory resources. This can be done by resizing the instance to a larger size or by adding more instances to your fleet.
- For example, if you are currently running a t2.micro instance, you could resize it to a t2.medium instance. This would give you twice the CPU and memory resources, which would allow your application to handle more traffic.

7.

8. Horizontal scalability

9. Like if one instance cannot handle the load..then we will create multiple instances of same type to handle the load

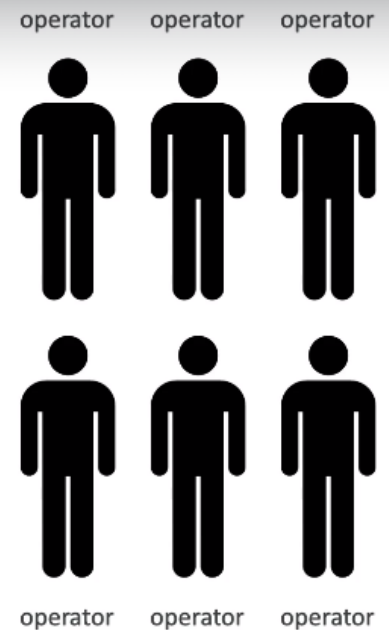
Sure, here is an example of horizontal scalability in AWS:

- You have an Amazon EC2 instance running a web application. As the application becomes more popular, you need to increase its capacity to handle the increased traffic.
- You can horizontally scale the application by adding more instances to your fleet. This can be done by creating a new instance and adding it to your fleet, or by using an Auto Scaling group to automatically add instances as needed.
- For example, if you currently have a single t2.micro instance, you could create a new t2.micro instance and add it to your fleet. This would give you twice the capacity as before, which would allow your application to handle even more traffic.
- You could also use an Auto Scaling group to automatically add instances as needed. For example, you could create an Auto Scaling group that scales up to 10 instances when the CPU utilization on your instances reaches 80%. This would ensure that your application always has enough capacity to handle the load.

10.

Horizontal Scalability

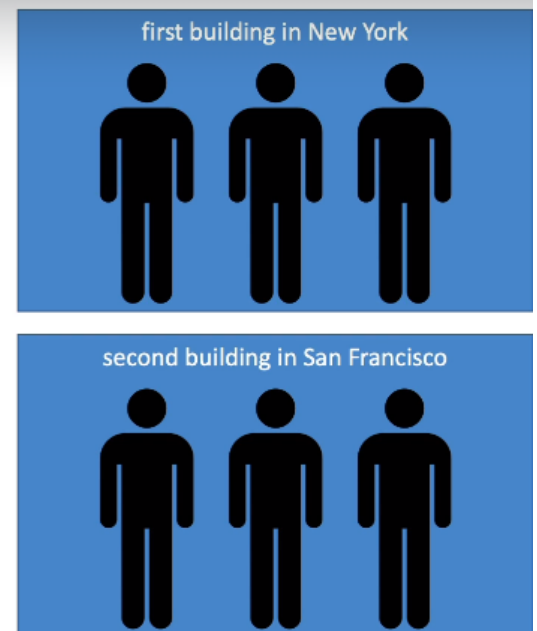
- Horizontal Scalability means increasing the number of instances / systems for your application
- Horizontal scaling implies distributed systems.
- This is very common for web applications / modern applications
- It's easy to horizontally scale thanks the cloud offerings such as Amazon EC2




11. 
12. High availability:

High Availability

- High Availability usually goes hand in hand with horizontal scaling
- High availability means running your application / system in at least 2 Availability Zones
- The goal of high availability is to survive a data center loss (disaster)



13. 
14. High availability and scalability for ec2

High Availability & Scalability For EC2

- Vertical Scaling: Increase instance size (= scale up / down)
 - From: t2.nano - 0.5G of RAM, 1 vCPU
 - To: u-12tb1.metal – 12.3 TB of RAM, 448 vCPUs
- Horizontal Scaling: Increase number of instances (= scale out / in)
 - Auto Scaling Group
 - Load Balancer
- High Availability: Run instances for the same application across multi AZ
 - Auto Scaling Group multi AZ
 - Load Balancer multi AZ

15. 

16. Now for horizontal, this is when you increase the number of instances,

17. it's called scaling out, or when you decrease the number of instances,

18. it's called scaling in. And for this, we'll be using an auto scaling group and a load balancer.

19. For high availability we run the same instance in different AZ

20. Difference bw scalability and Elasticity

Scalability vs Elasticity (vs Agility)

- **Scalability:** ability to accommodate a larger load by making the hardware stronger (scale up), or by adding nodes (scale out)
- **Elasticity:** once a system is scalable, elasticity means that there will be some “auto-scaling” so that the system can scale based on the load. This is “cloud-friendly”: pay-per-use, match demand, optimize costs
- **Agility:** (not related to scalability - distractor) new IT resources are only a click away, which means that you reduce the time to make those resources available to your developers from weeks to just minutes.

21.

Feature	Scalability	Elasticity
Definition	The ability of a system to increase its capacity to handle more load.	The ability of a system to automatically scale up or down in response to changes in load.
Implementation	Can be done manually or automatically.	Usually implemented using Auto Scaling.
Benefits	Can handle increased traffic.	Can handle increased traffic without any downtime.
Drawbacks	Can be expensive to scale manually.	Can be complex to implement.

22.

Sure. Scalability and elasticity are two important concepts in cloud computing. They are often used interchangeably, but there is a subtle difference between the two.

- **Scalability** refers to the ability of a system to increase its capacity to handle more load. This can be done by adding more resources to the system, such as more CPU, memory, or storage.
- **Elasticity** refers to the ability of a system to automatically scale up or down in response to changes in load. This is often done using a technique called Auto Scaling.

23.

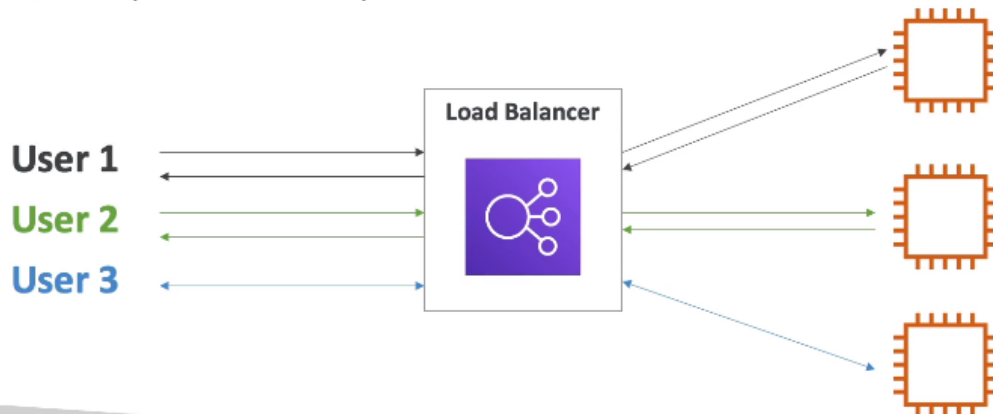
Elastic Load Balance(ELB) Overview:

1. Load Balancing:

What is load balancing?



- Load balancers are servers that forward internet traffic to multiple servers (EC2 Instances) downstream.



- 2.
3. Load Balancer will direct the traffic to multiple instance as we can see in the picture
4. Why use a load balancer

Why use a load balancer?

- Spread load across multiple downstream instances
- Expose a single point of access (DNS) to your application
- Seamlessly handle failures of downstream instances
- Do regular health checks to your instances
- Provide SSL termination (HTTPS) for your websites
- High availability across zones

- 5.

There are many reasons why we use load balancing in AWS. Here are some of the most common reasons:

- **To improve performance:** Load balancing can improve the performance of your applications by distributing traffic across multiple servers. This can help to reduce latency and improve the overall user experience.
- **To increase availability:** Load balancing can also help to increase the availability of your applications by distributing traffic across multiple servers. If one server fails, the load balancer will automatically route traffic to the remaining servers. This can help to prevent your application from going offline.
- **To simplify management:** Load balancing can simplify the management of your applications by providing a single point of contact for all traffic. This can make it easier to monitor your applications and troubleshoot problems.
- **To improve security:** Load balancing can also improve the security of your applications by providing a layer of protection between your users and your servers. The load balancer can filter traffic and block malicious requests.

6.

Here are some of the benefits of using load balancing in AWS:

- **Elasticity:** Load balancers in AWS can be scaled up or down automatically in response to changes in traffic. This means that you can easily handle spikes in traffic without having to worry about overloading your servers.
- **High availability:** Load balancers in AWS are highly available and can even be configured to be active-active. This means that your applications will be available even if one of your load balancers fails.
- **Security:** Load balancers in AWS can be configured to filter traffic and block malicious requests. This can help to protect your applications from security threats.

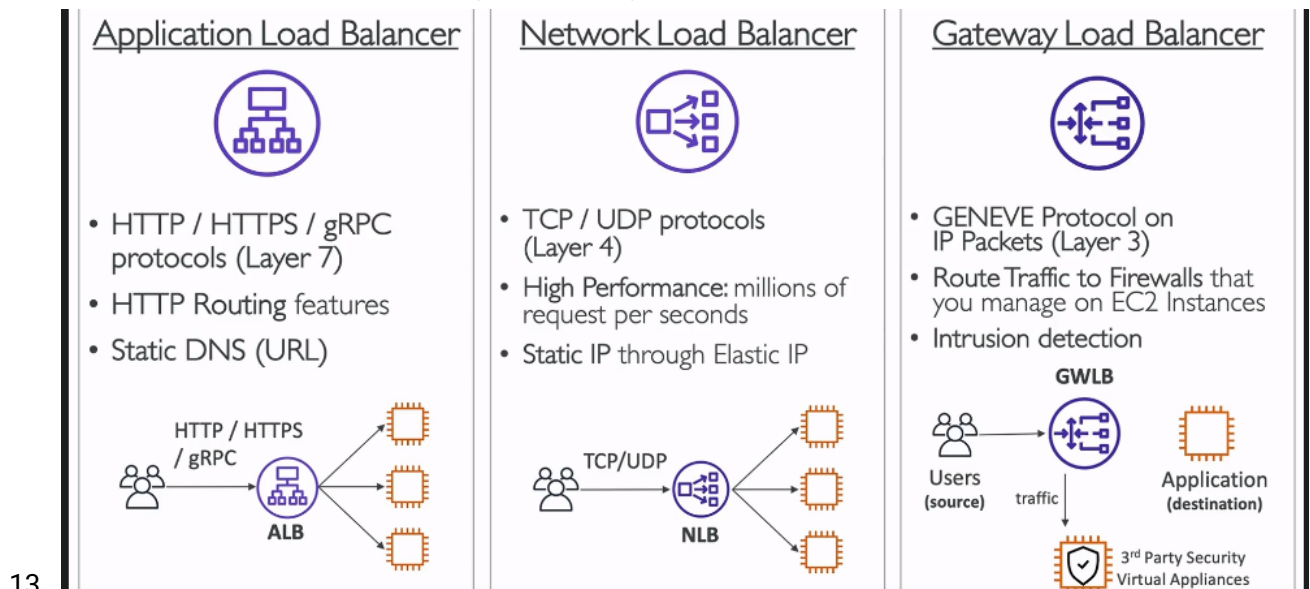
7.

8. Why use an Elastic load balancer?

Why use an Elastic Load Balancer?

- An ELB (Elastic Load Balancer) is a **managed load balancer**
 - AWS guarantees that it will be working
 - AWS takes care of upgrades, maintenance, high availability
 - AWS provides only a few configuration knobs
- It costs less to setup your own load balancer but it will be a lot more effort on your end (maintenance, integrations)
- 4 kinds of load balancers offered by AWS:
 - Application Load Balancer (HTTP / HTTPS only) – Layer 7
 - Network Load Balancer (ultra-high performance, allows for TCP) – Layer 4
 - Gateway Load Balancer – Layer 3
 - Classic Load Balancer (retired in 2023) – Layer 4 & 7

- 9.
10. Diff bw Application Load Balancer, Network Load Balancer, Gateway Load Balancer
11. In application load balancer...basically users access the LB using the protocols such as HTTP/HTTPS/gRPC..then load balancer will routes the traffic to the downstream EC2 instances
12. Similarly in the network load balancer ..the users access LB using TCP/UDP protocol and LB routes them to Instances...similarly for gateway load balancers



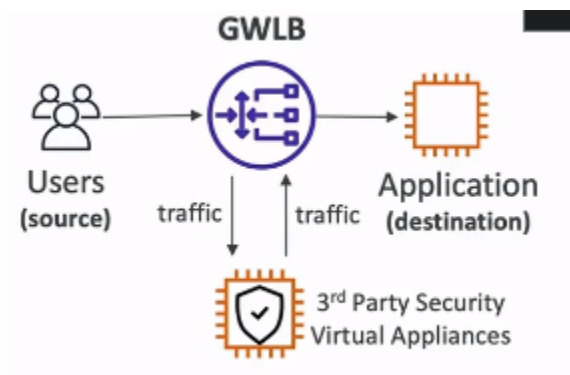
- 13.
14. Example for ALB,NLB,GLB

Sure. Here are some examples of how the three types of load balancers can be used:

- **Application Load Balancer:** An example of an application that would benefit from using an ALB is a web application that serves different content to different users based on their location. For example, a web application that serves English content to users in the United States and Spanish content to users in Latin America could use an ALB to route traffic to different servers based on the user's IP address.
- **Network Load Balancer:** An example of an application that would benefit from using an NLB is a gaming application that requires low latency. For example, a gaming application that allows users to play against each other in real time could use an NLB to route traffic to different servers based on the network location of the users. This would help to ensure that all users have a consistent experience, regardless of their location.
- **Gateway Load Balancer:** An example of an application that would benefit from using a GWLB is a virtual appliance-based application that needs to be isolated from the public internet. For example, an application that uses virtual appliances to provide firewall and intrusion detection services could use a GWLB to route traffic to the virtual appliances in different VPCs. This would help to protect the virtual appliances from unauthorized access.

15.

16. GLB



17.

18. Learn more about GLB in bard

ALB Hands on:

1. Here first we are going to create an Ec2 Instance ...Here we the number of instance we care creating are 2

2. In security groups we select existing SG which is launch wizard...it allows the https request in and out for our Ec2 instance
3. Just for knowledge..when troubleshooting and ec2 instance check this

Hello Lola,

Step-1

Check your instance running httpd web server on it to check this open your web browser

http:// <Your-instance-IP>

if not then create instance using user data using below method.

When troubleshooting connectivity to your instances check the following:

1. Ensure the correct ports(22,80) in the Security Group are open(When you creating your instance) that is associated with your instance
2. Ensure you have a public IP address associated with your instance
3. Instance is up and running
4. Make sure you are using Amazon-linux2 AMI
5. Make sure you correctly placed user data while launching your instance
6. Make sure you are hitting http:// <Your-instance-IP> in your browser

Step 2

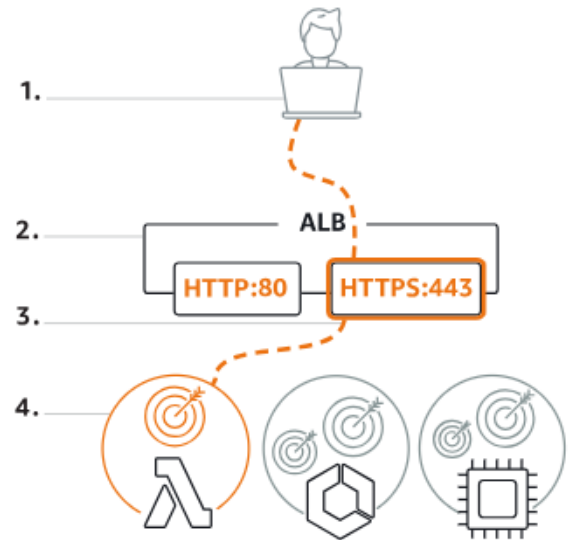
Check security groups settings. The port 80 may be restricted to access.

Check the security group of Load balancer and Also check security group of your ec2 instance. make sure it is allowed for port 80 for both load balancer and ec2 security group.

- 4.
5. Now we get 2 hello words from 2 instances..
6. And so what we'd like to do is to have only one URL to access these two EC2 instances and balance the load between them.
7. For this we will use load balancer
8. Now we will create an ALB for our instances..to create it..go to LB section and click on Create load balance..then we have to choose ALB
9. AWS definition of ALB

▼ How Elastic Load Balancing works

1. Clients make requests to your application.
2. The listeners in your load balancer receive requests matching the protocol and port that you configure.
3. The receiving listener evaluates the incoming request against the rules you specify, and if applicable, routes the request to the appropriate target group. You can use an HTTPS listener to offload the work of TLS encryption and decryption to your load balancer.
4. Healthy targets in one or more target groups receive traffic based on the load balancing algorithm, and the routing rules you specify in the listener.

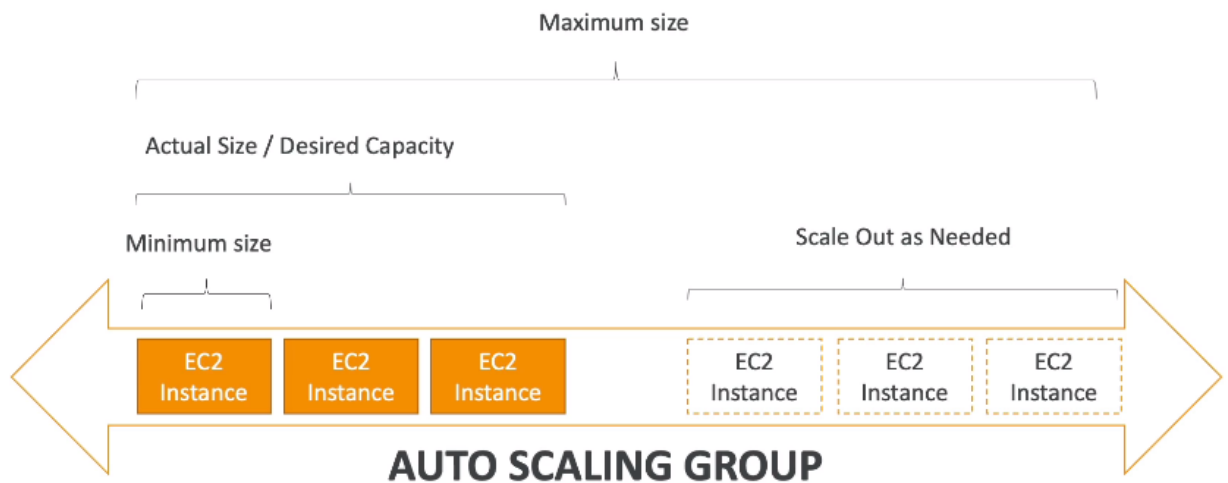



10. _____
11. While creating it ...we choose every A-Z available to use our ALB
12. We will create a new security grp and we allow only http traffic into that
13. In the listener section we have to add a target group(ALB listen data and sends to these target group which are our instances)...while creating target group we have to select instances type...and then we click on next and we register our instances to the target grp
14. Now after load balancer is active ...we can copy its DNS and open the url address
15. Now from the single url..we can access both of our instances
16. If we stopped one instance ..then ALB will make it as unused and display's us the output pf 2nd instance
17. And again if we reinstate the first instance...Load Balancer will know this it makes instance 1 as healthy and gives us 2 instances in single url

What's an Auto Scaling Group?

- In real-life, the load on your websites and application can change
 - In the cloud, you can create and get rid of servers very quickly
 - The goal of an Auto Scaling Group (ASG) is to:
 - Scale out (add EC2 instances) to match an increased load
 - Scale in (remove EC2 instances) to match a decreased load
- 1.
 2. Like for example you have a shopping website..Now on festival days the traffic will be high and at some times traffic will be low
 3. To handle this traffic patterns Auto Scaling Group will be helpful
 4. It automatically registers/deregisters instances based on traffic
 - The goal of an Auto Scaling Group (ASG) is to:
 - Scale out (add EC2 instances) to match an increased load
 - Scale in (remove EC2 instances) to match a decreased load
 - Ensure we have a minimum and a maximum number of machines running
 - Automatically register new instances to a load balancer
 - Replace unhealthy instances
 - Cost Savings: only run at an optimal capacity (principle of the cloud)
 - 5.

Auto Scaling Group in AWS



6. 
7. The min size in ASG is an instance, desired capacity = 3 instances..but our ASG will scale out and scale in based on the traffic load
8. We can also attach a load balancer with auto scaling group
9. Based on traffic ASG will create instances ...and load balancer will know this send traffic equally among instances

ASG Hands On:

1. We can create asg by going into asg section on left and clicking create.
2. Next we give a demo name and create a launch template for our asg..this template will tell asg how to create ec2 instance within it
3. It will be same as launching an instance..we have to select linux,network,storage and in advanced user data...we give some user data...because we want these instances to start with some user data..then click on create launch template
4. Next we select this launch template to our asg and click on next step..in the next step we choose network and AZ subnets...we choose 3 AZ ...click on next

5. In the step 3 we choose load balancer...here we choose an existing load balancer of target group ...and we turn on health check for ELB

Health checks

Health checks increase availability by replacing unhealthy instances. When you use multiple health checks, all are evaluated, and if at least one fails, instance replacement occurs.

EC2 health checks

 Always enabled

Additional health check types - optional [Info](#)

☒ Turn on Elastic Load Balancing health checks **Recommended**

Elastic Load Balancing monitors whether instances are available to handle requests. When it reports an unhealthy instance, EC2 Auto Scaling can replace it on its next periodic check.

6. In step 4 we choose grouping size..

Group size - optional [Info](#)

Specify the size of the Auto Scaling group by changing the desired capacity. You can also specify minimum and maximum capacity limits. Your desired capacity must be within the limit range.

Desired capacity

2

Minimum capacity

1

Maximum capacity

4

7. Then next we skip notifications and tags step and we create our ASG at last
8. As we have kept desired capacity of instance as 2...now our asg will create 2 instances
9. If we go and see our instances ..we 2 instance are being created due to our asg
10. Also our target group which we assigned to asg...will have these 2 instances in its group and load balancer will get this instances
11. If we terminated an instance in asg...as our desired instances is 2...our asg will create a new instance again ..see below pic

PreInService	Launching a new EC2 instance: i-0224b87bdd5c537a4	At 2022-11-30T16:03:10Z an instance was launched in response to an unhealthy instance needing to be replaced.	2022 November 30, 06:03:13 PM +02:00
InProgress	Terminating EC2 instance: i-0ddd5ad0ca4ded077	At 2022-11-30T16:03:10Z an instance was taken out of service in response to an EC2 health check indicating it has been terminated or stopped.	2022 November 30, 06:03:10 PM +02:00

12.


13. ASG will detect unhealthy instances and create new healthy instances replacing unhealthy once
14. This is the power of auto scaling groups

ASG Strategies

1. Manual scaling is like we change the desired instances manually

Auto Scaling Groups – Scaling Strategies

- Manual Scaling: Update the size of an ASG manually
- Dynamic Scaling: Respond to changing demand
 - Simple / Step Scaling
 - When a CloudWatch alarm is triggered (example CPU > 70%), then add 2 units
 - When a CloudWatch alarm is triggered (example CPU < 30%), then remove 1
 - Target Tracking Scaling
 - Example: I want the average ASG CPU to stay at around 40%
 - Scheduled Scaling
 - Anticipate a scaling based on known usage patterns
 - Example: increase the min. capacity to 10 at 5 pm on Fridays

2. 
3. Predictive scaling

Auto Scaling Groups – Scaling Strategies

- Predictive Scaling

- Uses Machine Learning to predict future traffic ahead of time



- 4.
5. Based on ML algo and patterns ..it it scales the traffic

- Predictive Scaling

- Uses Machine Learning to predict future traffic ahead of time
- Automatically provisions the right number of EC2 instances in advance
- Useful when your load has predictable time-based patterns

- 6.

Section cleaning:

1. Now to delete our instances ...first we have to delete our asg...if didnt do so..then asg will create again new instances automatically
2. Next we will delete the load balancer
3. Well, we don't have to because the target group don't cost you any money
4. and the target group is going to be empty because we have deleted the auto-scaling group and we have deleted the low balancer.

ELB & ASG Summary:

ELB & ASG – Summary

- High Availability vs Scalability (vertical and horizontal) vs Elasticity vs Agility in the Cloud
- Elastic Load Balancers (ELB)
 - Distribute traffic across backend EC2 instances, can be Multi-AZ
 - Supports health checks
 - 4 types: Classic (old), Application (HTTP – L7), Network (TCP – L4), Gateway (L3)
- Auto Scaling Groups (ASG)
 - Implement Elasticity for your application, across multiple AZ
 - Scale EC2 instances based on the demand on your system, replace unhealthy
 - Integrated with the ELB

1.

2. ASG when combined with ELB gives high availability,scalability,elasticity and agility in the cloud