

Elastic Load Balancing & Auto Scaling Groups Section

Scalability & High Availability

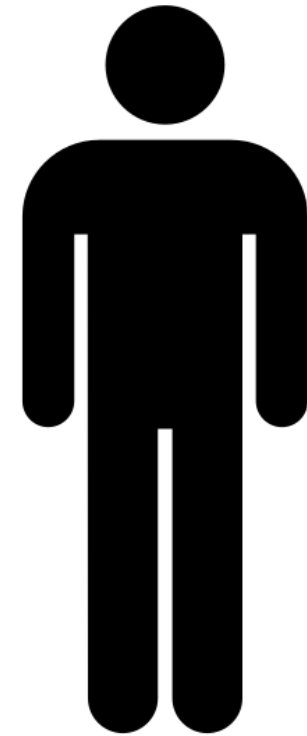
- Scalability means that an application / system can handle greater loads by adapting.
- There are two kinds of scalability:
 - Vertical Scalability
 - Horizontal Scalability (= elasticity)
- Scalability is linked but different to High Availability
- Let's deep dive into the distinction, using a call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

Vertical Scalability

- Vertical Scalability means increasing the size of the instance
- For example, your application runs on a t2.micro
- Scaling that application vertically means running it on a t2.large
- Vertical scalability is very common for non distributed systems, such as a database.
- There's usually a limit to how much you can vertically scale (hardware limit)



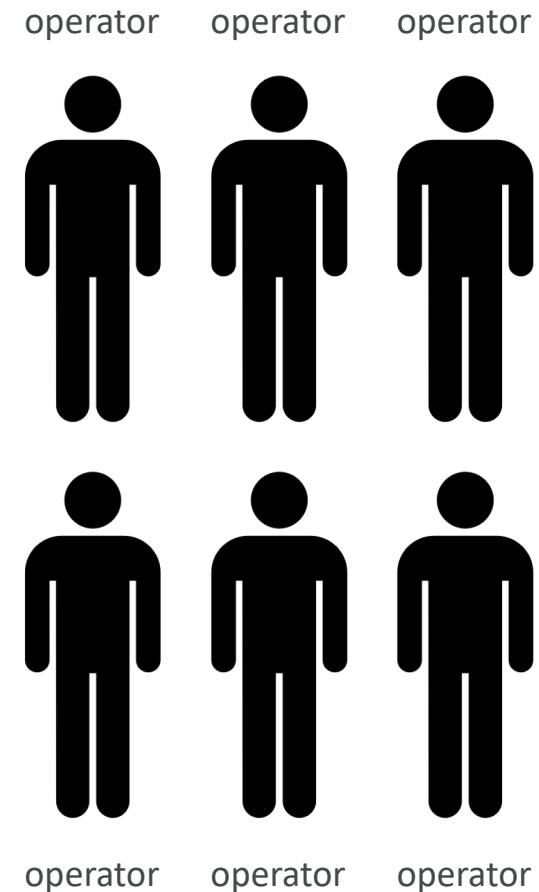
junior operator



senior operator

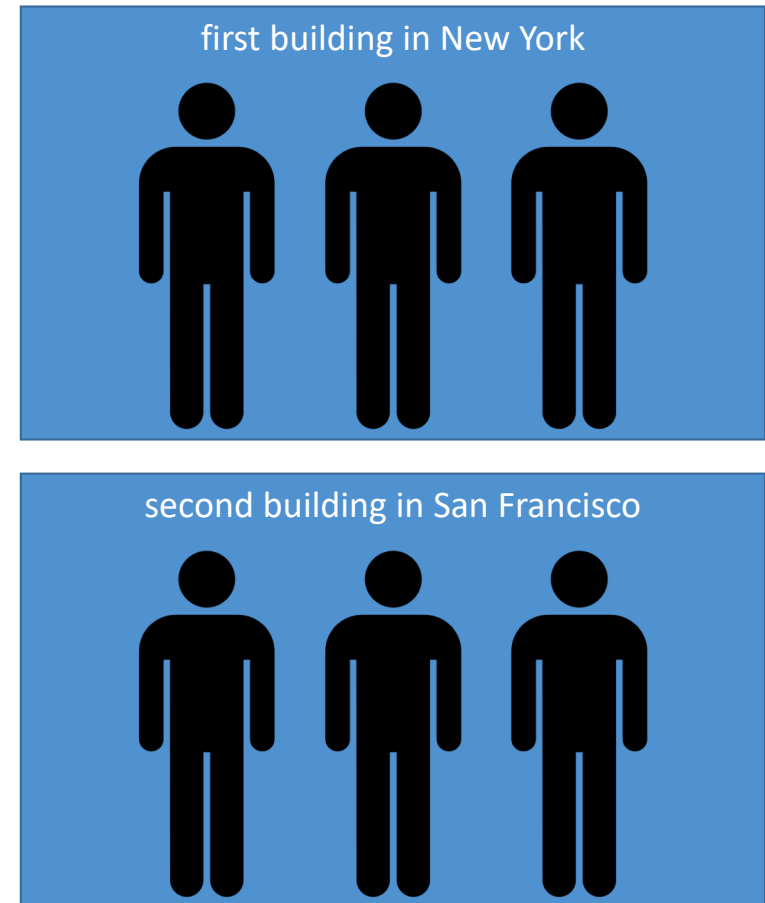
Horizontal Scalability

- 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



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)



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

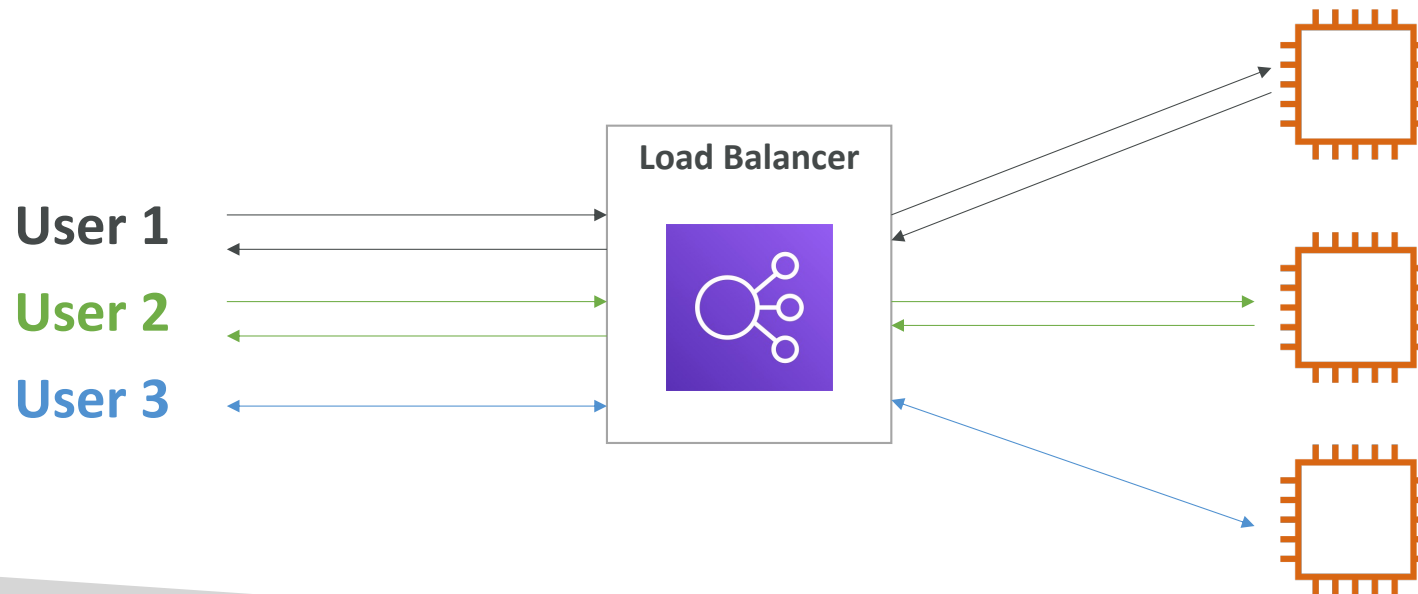
Scalability vs Elasticity (vs Agility)

- **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.

What is load balancing?



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

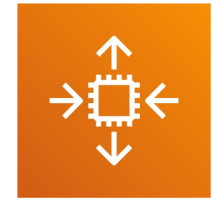


Why use a load balancer?

- 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

Why use an Elastic Load Balancer?

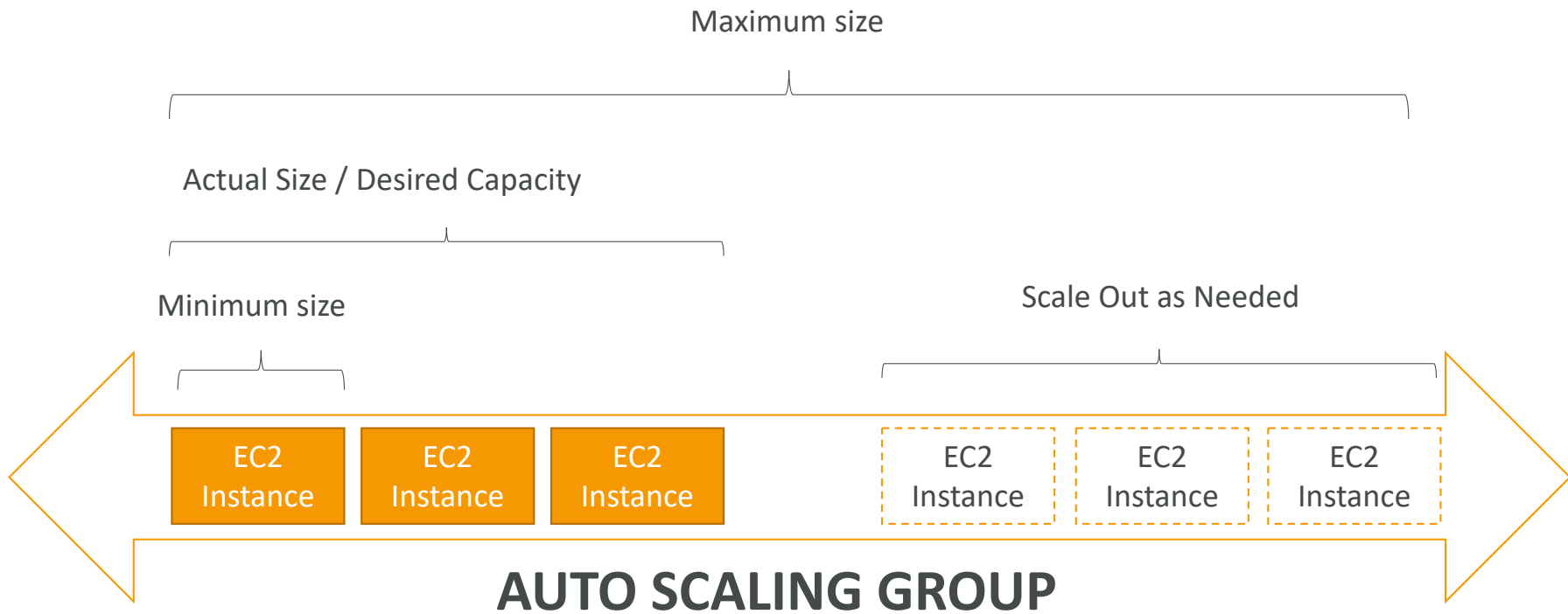
- An ELB (Elastic Load Balancer) is a **managed load balancer**
 - AWS guarantees that it will be working
 - AWS takes care of upgrades, maintenance, high availability
 - AWS provides only a few configuration knobs
- It costs less to setup your own load balancer but it will be a lot more effort on your end (maintenance, integrations)
- 3 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
 - Classic Load Balancer (slowly retiring) – Layer 4 & 7



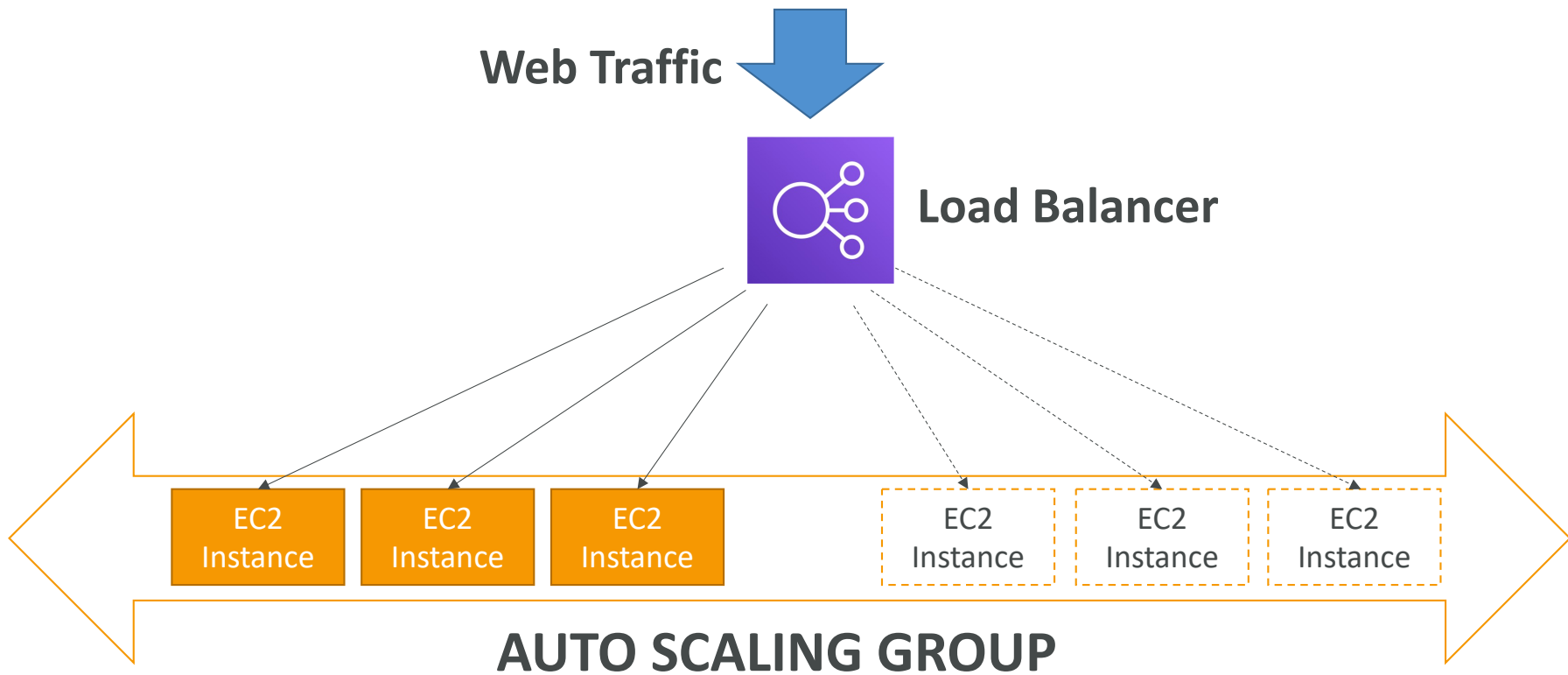
What's an Auto Scaling Group?

- 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
 - 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)

Auto Scaling Group in AWS



Auto Scaling Group in AWS With Load Balancer

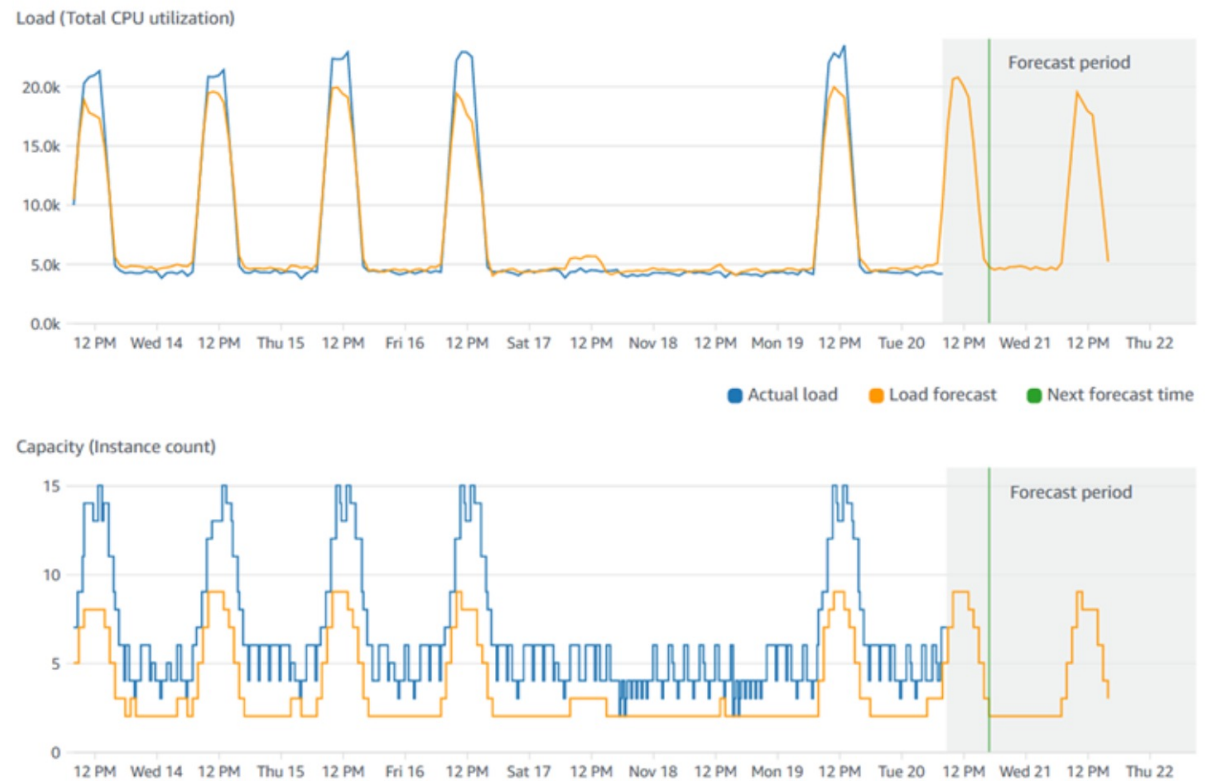


Auto Scaling Groups – Scaling Strategies

- **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

Auto Scaling Groups – Scaling Strategies

- **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



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
 - 3 types: Application LB (HTTP – L7), Network LB (TCP – L4), Classic LB (old)
- **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