

# What is Cloud Computing?



- Cloud computing is the on-demand delivery of compute power, database storage, applications, and other IT resources
- Through a cloud services platform with pay-as-you-go pricing
- You can provision exactly the right type and size of computing resources you need
- You can access as many resources as you need, almost instantly
- Simple way to access servers, storage, databases and a set of application services
- Amazon Web Services owns and maintains the network-connected hardware required for these application services, while you provision and use what you need via a web application.

# The Deployment Models of the Cloud

## Private Cloud:

- Cloud services used by a single organization, not exposed to the public.
- Complete control
- Security for sensitive applications
- Meet specific business needs



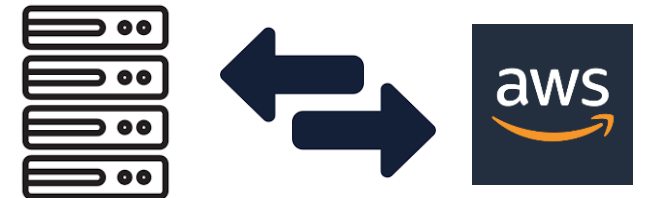
## Public Cloud:

- Cloud resources owned and operated by a third-party cloud service provider delivered over the Internet.
- Six Advantages of Cloud Computing



## Hybrid Cloud:

- Keep some servers on premises and extend some capabilities to the Cloud
- Control over sensitive assets in your private infrastructure
- Flexibility and cost-effectiveness of the public cloud



# The Five Characteristics of Cloud Computing

- On-demand self service:
  - Users can provision resources and use them without human interaction from the service provider
- Broad network access:
  - Resources available over the network, and can be accessed by diverse client platforms
- Multi-tenancy and resource pooling:
  - Multiple customers can share the same infrastructure and applications with security and privacy
  - Multiple customers are serviced from the same physical resources
- Rapid elasticity and scalability:
  - Automatically and quickly acquire and dispose resources when needed
  - Quickly and easily scale based on demand
- Measured service:
  - Usage is measured, users pay correctly for what they have used

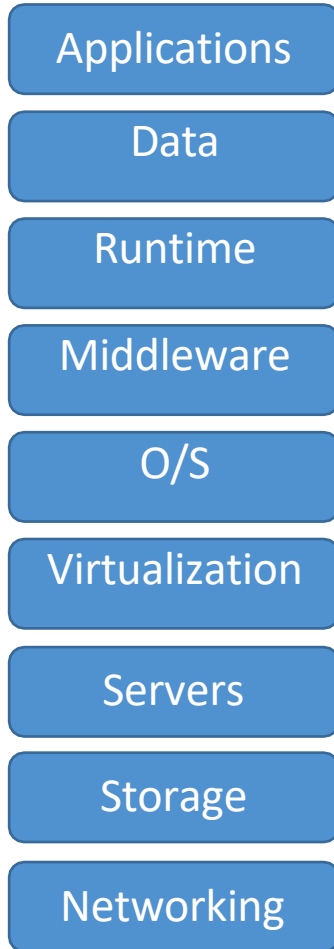
# Six Advantages of Cloud Computing

- Trade capital expense (CAPEX) for operational expense (OPEX)
  - Pay On-Demand: don't own hardware
  - Reduced Total Cost of Ownership (TCO) & Operational Expense (OPEX)
- Benefit from massive economies of scale
  - Prices are reduced as AWS is more efficient due to large scale
- Stop guessing capacity
  - Scale based on actual measured usage
- Increase speed and agility
- Stop spending money running and maintaining data centers
- Go global in minutes: leverage the AWS global infrastructure

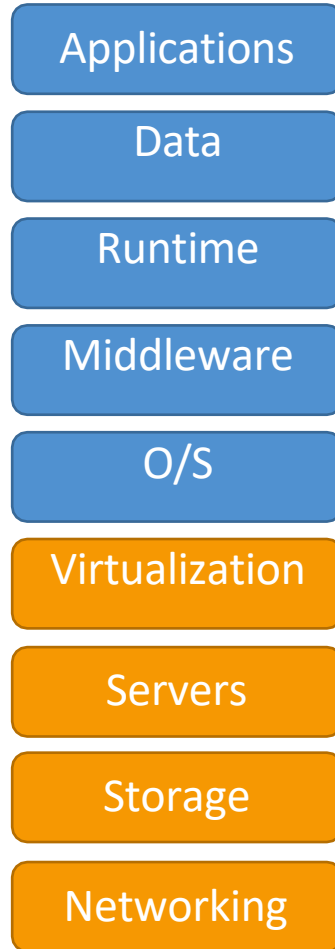
# Types of Cloud Computing

- Infrastructure as a Service (IaaS)
  - Provide building blocks for cloud IT
  - Provides networking, computers, data storage space
  - Highest level of flexibility
  - Easy parallel with traditional on-premises IT
- Platform as a Service (PaaS)
  - Removes the need for your organization to manage the underlying infrastructure
  - Focus on the deployment and management of your applications
- Software as a Service (SaaS)
  - Completed product that is run and managed by the service provider

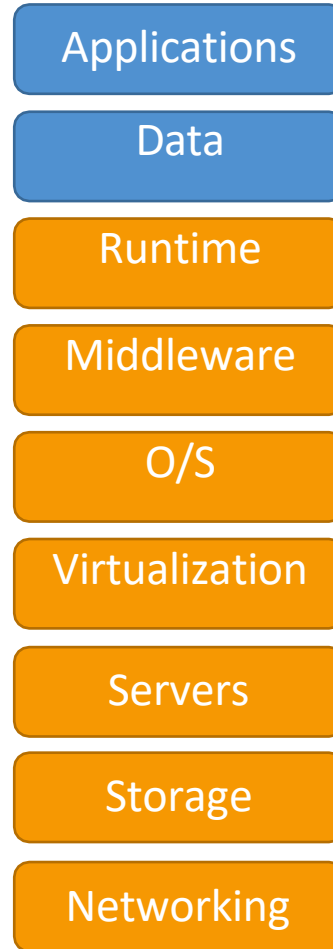
## On-premises



## Infrastructure as a Service (IaaS)



## Platform as a Service (PaaS)



## Software as a Service (SaaS)

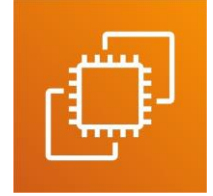


Managed by you

Managed by others

# Example of Cloud Computing Types

- Infrastructure as a Service:
  - Amazon EC2 (on AWS)
  - GCP, Azure, Digital Ocean,
- Platform as a Service:
  - Elastic Beanstalk (on AWS)
- Software as a Service:
  - Many AWS services (ex: Rekognition for Machine Learning)
  - Google Apps (Gmail), Dropbox, Zoom



# AWS Cloud Use Cases

- AWS enables you to build sophisticated, scalable applications
- Applicable to a diverse set of industries
- Use cases include
  - Enterprise IT, Backup & Storage, Big Data analytics
  - Website hosting, Mobile & Social Apps
  - Gaming



21ST  
CENTURY  
FOX

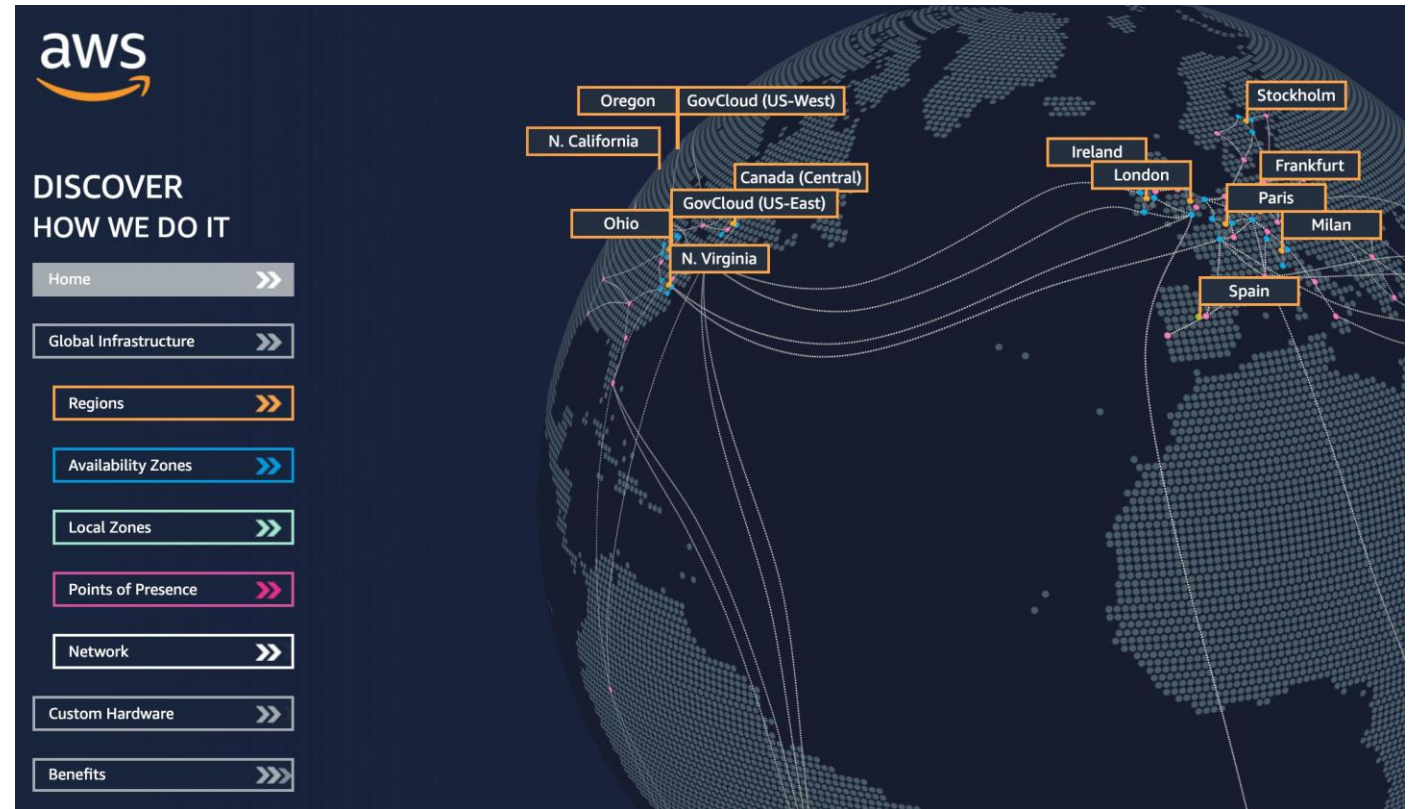
ACTIVISION

NETFLIX



# AWS Global Infrastructure

- AWS Regions
- AWS Availability Zones
- AWS Data Centers
- AWS Edge Locations / Points of Presence
- <https://infrastructure.aws/>



# AWS Regions

- AWS has Regions all around the world
- Names can be us-east-1, eu-west-3 ...
- A region is a cluster of data centers
- Most AWS services are region-scoped



<https://aws.amazon.com/about-aws/global-infrastructure/>

**US East (N. Virginia)** us-east-1

US East (Ohio) us-east-2

US West (N. California) us-west-1

US West (Oregon) us-west-2

Africa (Cape Town) af-south-1

Asia Pacific (Hong Kong) ap-east-1

Asia Pacific (Mumbai) ap-south-1

Asia Pacific (Seoul) ap-northeast-2

Asia Pacific (Singapore) ap-southeast-1

Asia Pacific (Sydney) ap-southeast-2

Asia Pacific (Tokyo) ap-northeast-1

Canada (Central) ca-central-1

Europe (Frankfurt) eu-central-1

Europe (Ireland) eu-west-1

Europe (London) eu-west-2

Europe (Paris) eu-west-3

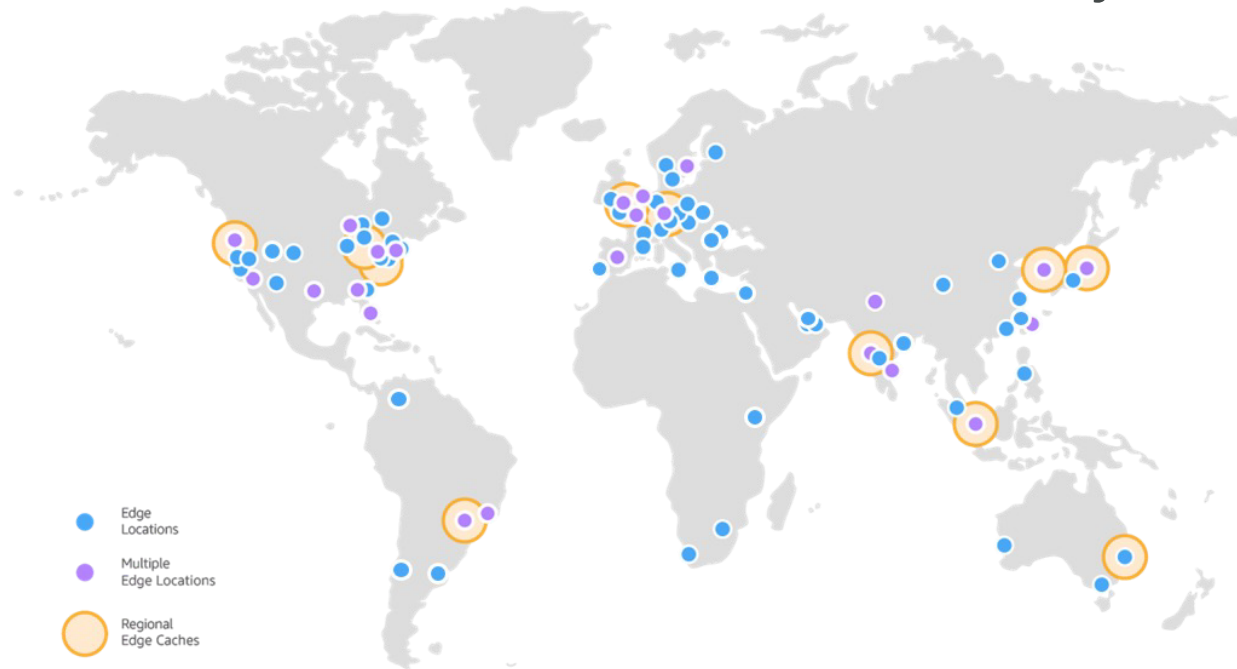
Europe (Stockholm) eu-north-1

Middle East (Bahrain) me-south-1

South America (São Paulo) sa-east-1

# AWS Points of Presence (Edge Locations)

- Amazon has 216 Points of Presence (205 Edge Locations & 11 Regional Caches) in 84 cities across 42 countries
- Content is delivered to end users with lower latency



<https://aws.amazon.com/cloudfront/features/>

# IAM Section

# IAM: Users & Groups



- IAM = Identity and Access Management, Global service
- Root account created by default, shouldn't be used or shared
- Users are people within your organization, and can be grouped
- Groups only contain users, not other groups
- Users don't have to belong to a group, and user can belong to multiple groups



# AWS Identity and Access Management (IAM)

- **Authentication** (is it the right user?) and
- **Authorization** (do they have the right access?)
- **Identities** can be
  - AWS users or
  - Federated users (externally authenticated users)
- Provides very **granular** control
  - Limit a single user:
    - to perform single action
    - on a specific AWS resource
    - from a specific IP address
    - during a specific time window

# Important IAM Concepts

- **IAM users:** Users created in an AWS account
  - Has credentials attached (name/password or access keys)
- **IAM groups:** Collection of IAM users
- **Roles:** Temporary identities
  - Does NOT have credentials attached
  - (Advantage) Expire after a set period of time
- **Policies:** Define permissions
  - **AWS managed policies** - Standalone policy predefined by AWS
  - **Customer managed policies** - Standalone policy created by you
  - **Inline policies** - Directly embedded into a user, group or role



# IAM: Permissions

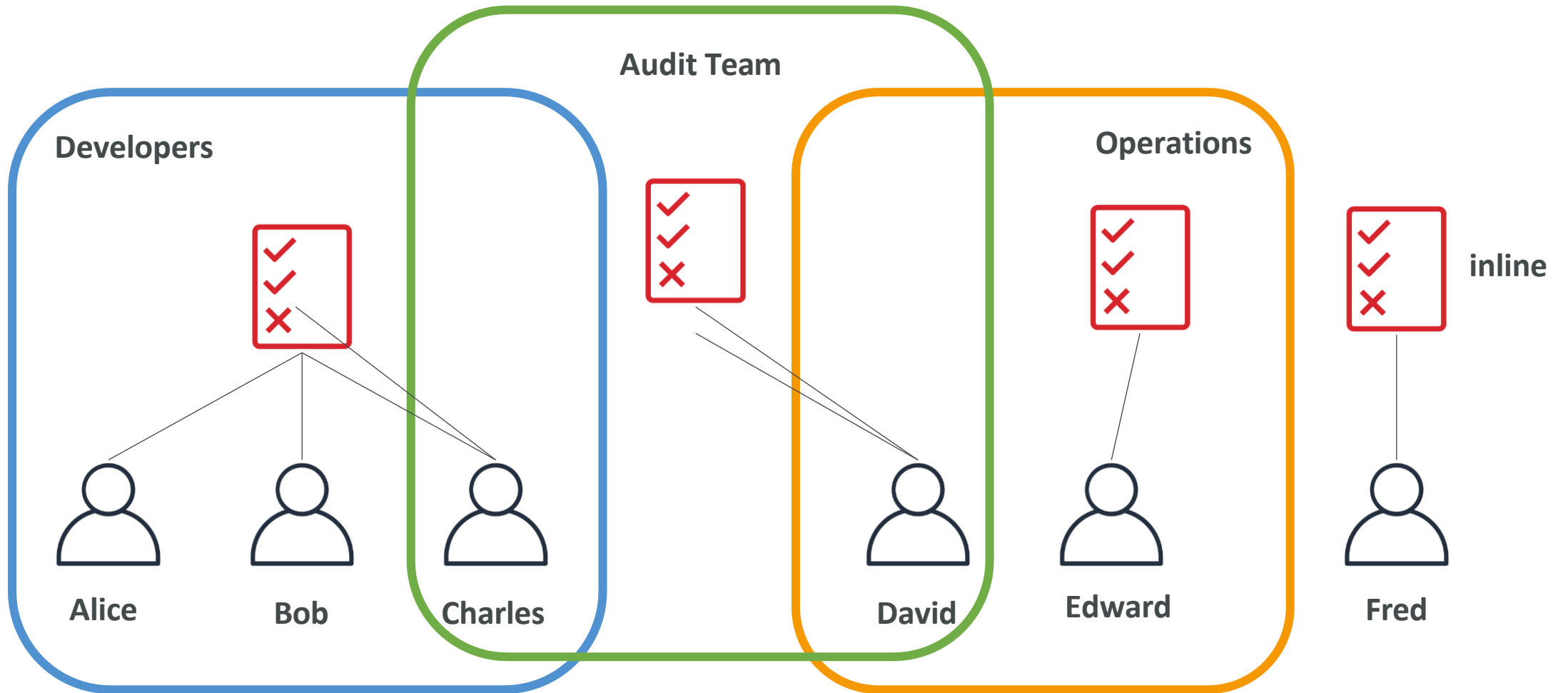
- Users or Groups can be assigned JSON documents called policies
- These policies define the permissions of the users
- In AWS you apply the least privilege principle: don't give more permissions than a user needs

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:Describe*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "elasticloadbalancing:Describe*",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "cloudwatch:ListMetrics",
        "cloudwatch:GetMetricStatistics",
        "cloudwatch:Describe*"
      ],
      "Resource": "*"
    }
  ]
}
```





# IAM Policies inheritance



# IAM – Password Policy

- Strong passwords = higher security for your account
- In AWS, you can setup a password policy:
  - Set a minimum password length
  - Require specific character types:
    - including uppercase letters
    - lowercase letters
    - numbers
    - non-alphanumeric characters
  - Allow all IAM users to change their own passwords
  - Require users to change their password after some time (password expiration)
  - Prevent password re-use

# Multi Factor Authentication - MFA



- Users have access to your account and can possibly change configurations or delete resources in your AWS account
- You want to protect your Root Accounts and IAM users
- MFA = password *you know* + security device *you own*



Alice

Password

+



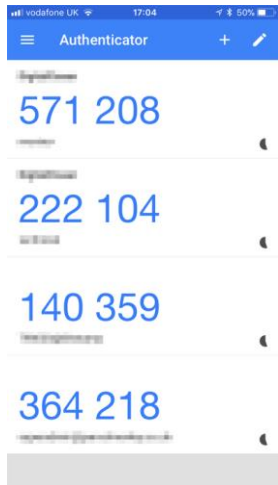
=>

Successful login

- Main benefit of MFA:  
if a password is stolen or hacked, the account is not compromised

# MFA devices options in AWS

## Virtual MFA device



**Google Authenticator**  
(phone only)



**Authy**  
(multi-device)

Support for multiple tokens on a single device.

# How can users access AWS ?




- To access AWS, you have three options:
  - AWS Management Console (protected by password + MFA)
  - AWS Command Line Interface (CLI): protected by access keys
  - AWS Software Developer Kit (SDK) - for code: protected by access keys
- Access Keys are generated through the AWS Console
- Users manage their own access keys
- Access Keys are secret, just like a password. Don't share them
- Access Key ID ~ = username
- Secret Access Key ~ = password

# Example Access Keys

## Access keys

Use access keys to make secure REST or HTTP Query protocol requests to AWS service APIs. For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent key rotation. [Learn more](#)

Create access key

| Access key ID        | Created                   | Last used | Status |   |
|----------------------|---------------------------|-----------|--------|---|
| AKIASK4E37PV4TU3RD6C | 2020-05-25 15:13 UTC+0100 | N/A       | Active | <a href="#">Make inactive</a>  |

- Access key ID: AKIASK4E37PV4983d6C
- Secret Access Key: AZPN3zoiWozWCndIjhB0Unh8239a1bzO5fqqkZq
- Remember: don't share your access keys

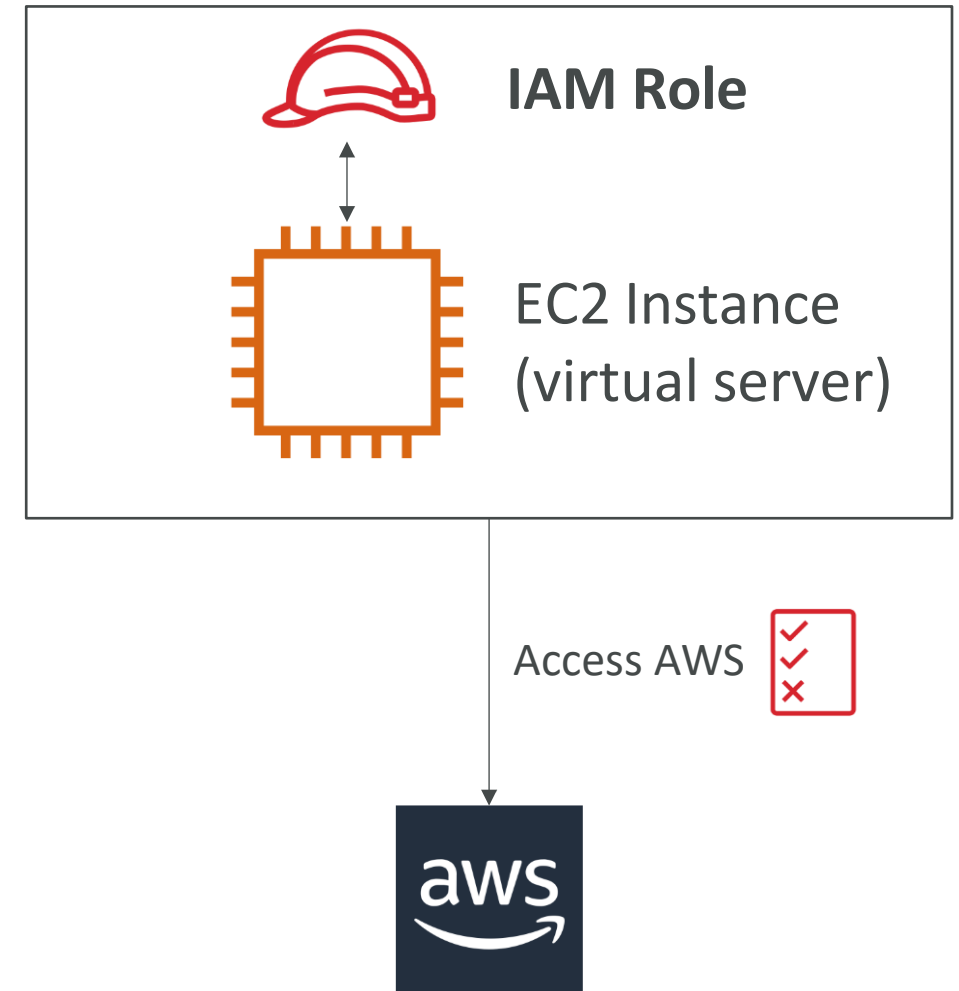
# What's the AWS CLI?

- A tool that enables you to interact with AWS services using commands in your command-line shell
- Direct access to the public APIs of AWS services
- You can develop scripts to manage your resources
- It's open-source <https://github.com/aws/aws-cli>
- Alternative to using AWS Management Console

```
→ ~ aws s3 cp myfile.txt s3://ccp-mybucket/myfile.txt
upload: ./myfile.txt to s3://ccp-mybucket/myfile.txt
→ ~ aws s3 ls s3://ccp-mybucket
2021-05-14 03:22:52          0 myfile.txt
→ ~
```

# IAM Roles for Services

- Some AWS service will need to perform actions on your behalf
- To do so, we will assign permissions to AWS services with IAM Roles
- Common roles:
  - EC2 Instance Roles
  - Lambda Function Roles
  - Roles for CloudFormation





# IAM Security Tools

- IAM Credentials Report (account-level)
  - a report that lists all your account's users and the status of their various credentials
- IAM Access Advisor (user-level)
  - Access advisor shows the service permissions granted to a user and when those services were last accessed.
  - You can use this information to revise your policies.

# IAM Guidelines & Best Practices



- Don't use the root account except for AWS account setup
- One physical user = One AWS user
- Assign users to groups and assign permissions to groups
- Create a strong password policy
- Use and enforce the use of Multi Factor Authentication (MFA)
- Create and use Roles for giving permissions to AWS services
- Use Access Keys for Programmatic Access (CLI / SDK)
- Audit permissions of your account with the IAM Credentials Report
- Never share IAM users & Access Keys

# Shared Responsibility Model for IAM



- Infrastructure (global network security)
- Configuration and vulnerability analysis
- Compliance validation



You

- Users, Groups, Roles, Policies management and monitoring
- Enable MFA on all accounts
- Rotate all your keys often
- Use IAM tools to apply appropriate permissions
- Analyze access patterns & review permissions

# IAM Section - Summary



- Users: mapped to a physical user, has a password for AWS Console
- Groups: contains users only
- Policies: JSON document that outlines permissions for users or groups
- Roles: for EC2 instances or AWS services
- Security: MFA + Password Policy
- AWS CLI: manage your AWS services using the command-line
- AWS SDK: manage your AWS services using a programming language
- Access Keys: access AWS using the CLI or SDK
- Audit: IAM Credential Reports & IAM Access Advisor

# EC2 Section

# Amazon EC2



- EC2 is one of the most popular of AWS' offerings
- EC2 = Elastic Compute Cloud = Infrastructure as a Service
- It mainly consists in the capability of :
  - Renting virtual machines (EC2)
  - Storing data on virtual drives (EBS)
  - Distributing load across machines (ELB)
  - Scaling the services using an auto-scaling group (ASG)
- Knowing EC2 is fundamental to understand how the Cloud works

# EC2 sizing & configuration options

- Operating System (OS): Linux, Windows or Mac OS
- How much compute power & cores (CPU)
- How much random-access memory (RAM)
- How much storage space:
  - Network-attached (EBS & EFS)
  - hardware (EC2 Instance Store)
- Network card: speed of the card, Public IP address
- Firewall rules: security group
- Bootstrap script (configure at first launch): EC2 User Data


# EC2 User Data

- It is possible to bootstrap our instances using an [EC2 User data](#) script.
- [bootstrapping](#) means launching commands when a machine starts
- That script is [only run once](#) at the instance [first start](#)
- EC2 user data is used to automate boot tasks such as:
  - Installing updates
  - Installing software
  - Downloading common files from the internet
  - Anything you can think of
- The EC2 User Data Script runs with the root user



# Hands-On:

## Launching an EC2 Instance running Linux

- We'll be launching our first virtual server using the AWS Console
  - We'll get a first high-level approach to the various parameters
  - We'll see that our web server is launched using EC2 user data
  - We'll learn how to start / stop / terminate our instance.
- 

# EC2 Instance Types - Overview

- You can use different types of EC2 instances that are optimised for different use cases (<https://aws.amazon.com/ec2/instance-types/>)
- AWS has the following naming convention:

m5.2xlarge

- **m**: instance class
- **5**: generation (AWS improves them over time)
- **2xlarge**: size within the instance class

General Purpose

Compute Optimized

Memory Optimized

Accelerated Computing

Storage Optimized

Instance Features

Measuring Instance  
Performance

# EC2 Instance Types: example

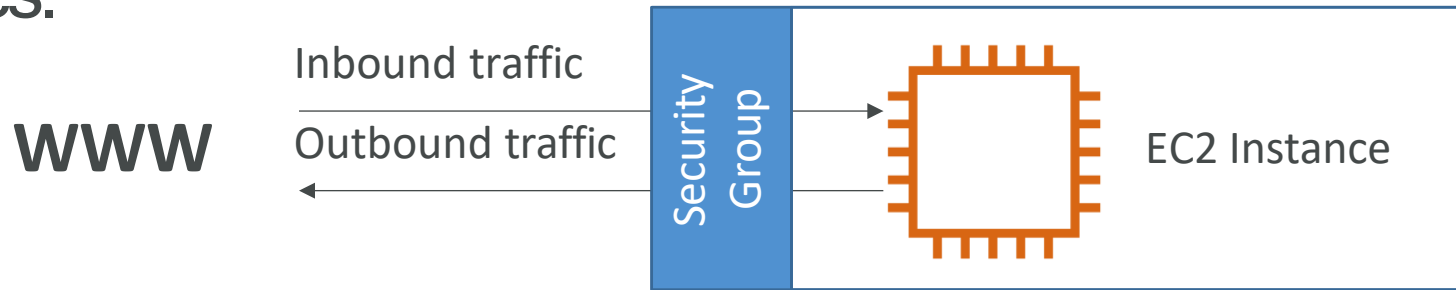
| Instance    | vCPU | Mem (GiB) | Storage          | Network Performance | EBS Bandwidth (Mbps) |
|-------------|------|-----------|------------------|---------------------|----------------------|
| t2.micro    | 1    | 1         | EBS-Only         | Low to Moderate     |                      |
| t2.xlarge   | 4    | 16        | EBS-Only         | Moderate            |                      |
| c5d.4xlarge | 16   | 32        | 1 x 400 NVMe SSD | Up to 10 Gbps       | 4,750                |
| r5.16xlarge | 64   | 512       | EBS Only         | 20 Gbps             | 13,600               |
| m5.8xlarge  | 32   | 128       | EBS Only         | 10 Gbps             | 6,800                |

**t2.micro is part of the AWS free tier (up to 750 hours per month)**

Great website: <https://instances.vantage.sh>

# Introduction to Security Groups

- Security Groups are the fundamental of network security in AWS
- They control how traffic is allowed into or out of our EC2 Instances.



- Security groups only contain **allow** rules
- Security groups rules can reference by IP or by security group

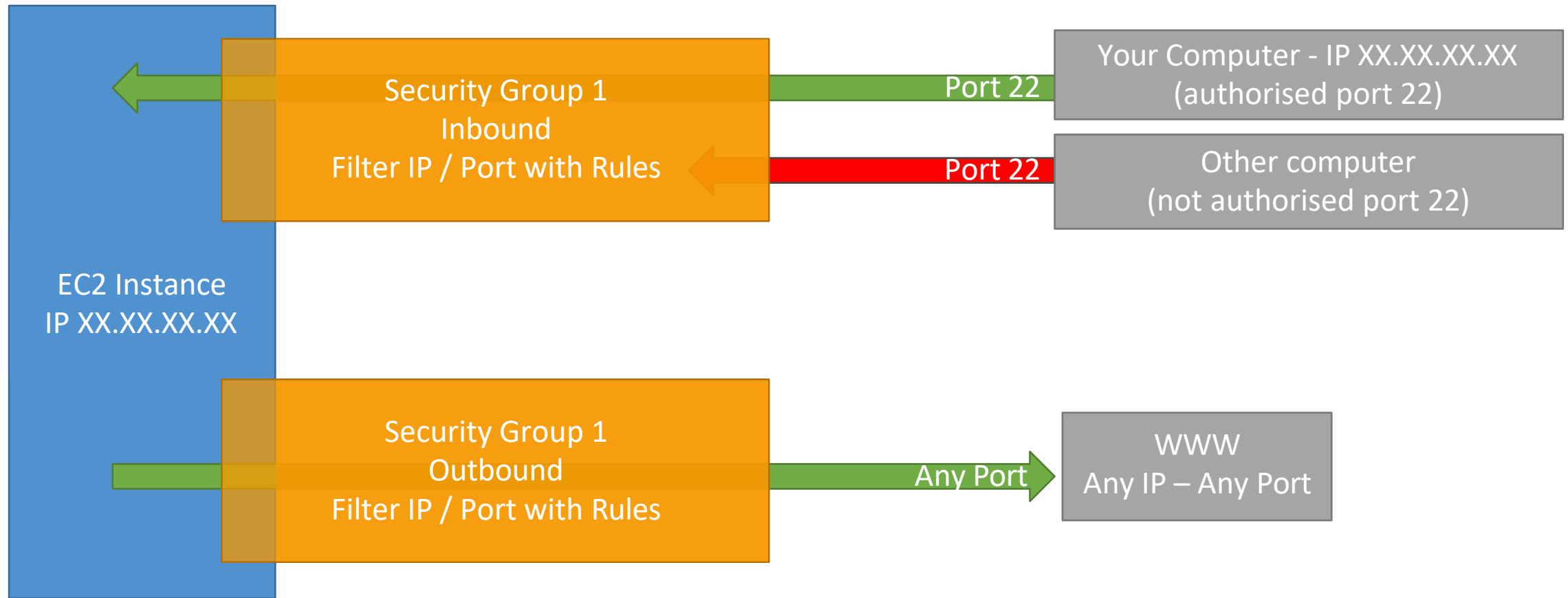
# Security Groups

## Deeper Dive

- Security groups are acting as a “firewall” on EC2 instances
- They regulate:
  - Access to Ports
  - Authorised IP ranges - IPv4 and IPv6
  - Control of inbound network (from other to the instance)
  - Control of outbound network (from the instance to other)

| Type ⓘ          | Protocol ⓘ | Port Range ⓘ | Source ⓘ          | Description ⓘ  |
|-----------------|------------|--------------|-------------------|----------------|
| HTTP            | TCP        | 80           | 0.0.0.0/0         | test http page |
| SSH             | TCP        | 22           | 122.149.196.85/32 |                |
| Custom TCP Rule | TCP        | 4567         | 0.0.0.0/0         | java app       |

# Security Groups Diagram



# Security Groups

## Good to know

- Can be attached to multiple instances
- Locked down to a region / VPC combination
- Does live “outside” the EC2 – if traffic is blocked the EC2 instance won’t see it
- It’s good to maintain one separate security group for SSH access
- If your application is not accessible (time out), then it’s a security group issue
- If your application gives a “connection refused” error, then it’s an application error or it’s not launched
- All inbound traffic is **blocked** by default
- All outbound traffic is **authorised** by default

# Classic Ports to know

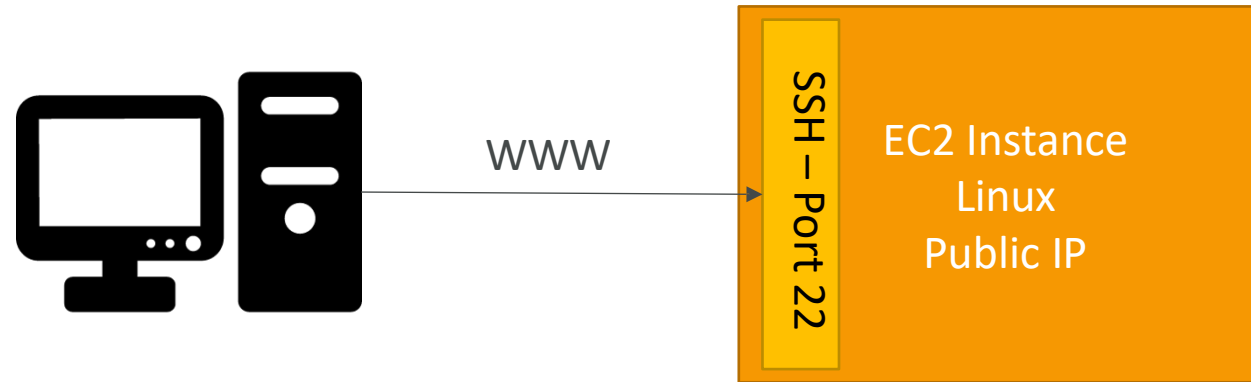
- 22 = SSH (Secure Shell) - log into a Linux instance
- 21 = FTP (File Transfer Protocol) - upload files into a file share
- 22 = SFTP (Secure File Transfer Protocol) - upload files using SSH
- 80 = HTTP - access unsecured websites
- 443 = HTTPS - access secured websites
- 3389 = RDP (Remote Desktop Protocol) - log into a Windows instance



# How to SSH into your EC2 Instance

## Linux / Mac OS X

- We'll learn how to SSH into your EC2 instance using [Linux / Mac](#)
- SSH is one of the most important function. It allows you to control a remote machine, all using the command line.

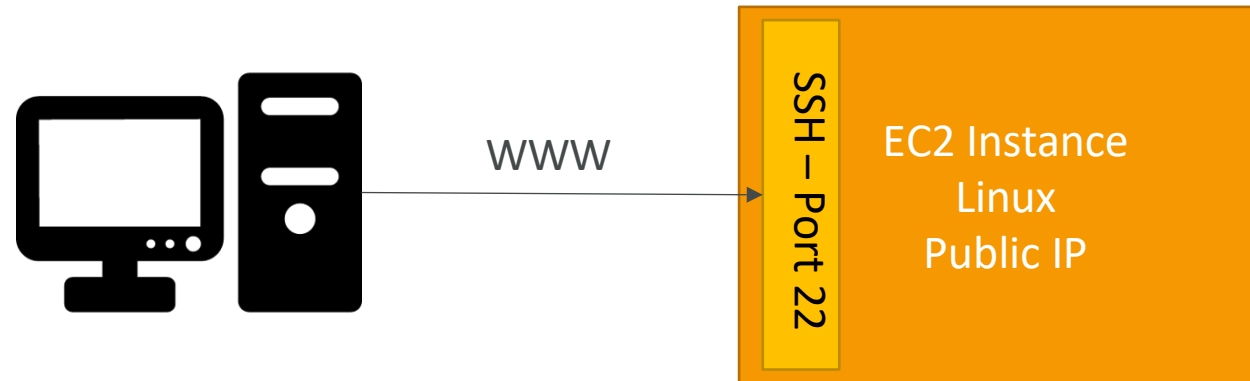


- We will see how we can configure OpenSSH [~/.ssh/config](#) to facilitate the SSH into our EC2 instances

# How to SSH into your EC2 Instance

## Windows

- We'll learn how to SSH into your EC2 instance using **Windows**
- SSH is one of the most important function. It allows you to control a remote machine, all using the command line.



- We will configure all the required parameters necessary for doing SSH on Windows using the free tool **Putty**.

# EC2 Instance Connect

- Connect to your EC2 instance within your browser
- No need to use your key file that was downloaded
- The “magic” is that a temporary key is uploaded onto EC2 by AWS
- Works only out-of-the-box with Amazon Linux 2
- Need to make sure the port 22 is still opened!

# EC2 Instances Purchasing Options

- On-Demand Instances - short workload, predictable pricing, pay by second
- Reserved (1 & 3 years)
  - Reserved Instances - long workloads
  - Convertible Reserved Instances - long workloads with flexible instances
- Savings Plans (1 & 3 years) - commitment to an amount of usage, long workload
- Spot Instances - short workloads, cheap, can lose instances (less reliable)
- Dedicated Hosts - book an entire physical server, control instance placement
- Dedicated Instances - no other customers will share your hardware
- Capacity Reservations - reserve capacity in a specific AZ for any duration

# EC2 On Demand

- Pay for what you use:
  - Linux or Windows - billing per second, after the first minute
  - All other operating systems - billing per hour
- Has the highest cost but no upfront payment
- No long-term commitment
- Recommended for short-term and un-interrupted workloads, where you can't predict how the application will behave

# EC2 Reserved Instances

- Up to 72% discount compared to On-demand
- You reserve a specific instance attributes (Instance Type, Region, Tenancy, OS)
- Reservation Period - 1 year (+discount) or 3 years (+++discount)
- Payment Options - No Upfront (+), Partial Upfront (++), All Upfront (+++)
- Reserved Instance's Scope - Regional or Zonal (reserve capacity in an AZ)
- Recommended for steady-state usage applications (think database)
- You can buy and sell in the Reserved Instance Marketplace
- Convertible Reserved Instance
  - Can change the EC2 instance type, instance family, OS, scope and tenancy
  - Up to 66% discount

**Note:** the % discounts are different from the video as AWS change them over time – the exact numbers are not needed for the exam. This is just for illustrative purposes ©

# EC2 Savings Plans

- Get a discount based on long-term usage (up to 72% - same as RIs)
- Commit to a certain type of usage (\$10/hour for 1 or 3 years)
- Usage beyond EC2 Savings Plans is billed at the On-Demand price
- Locked to a specific instance family & AWS region (e.g., M5 in us-east-1)
- Flexible across:
  - Instance Size (e.g., m5.xlarge, m5.2xlarge)
  - OS (e.g., Linux, Windows)
  - Tenancy (Host, Dedicated, Default)

# EC2 Spot Instances



- Can get a discount of up to 90% compared to On-demand
- Instances that you can “lose” at any point of time if your max price is less than the current spot price
- The MOST cost-efficient instances in AWS
- Useful for workloads that are resilient to failure
  - Batch jobs
  - Data analysis
  - Image processing
  - Any distributed workloads
  - Workloads with a flexible start and end time
- Not suitable for critical jobs or databases



# EC2 Dedicated Hosts

- A physical server with EC2 instance capacity fully dedicated to your use
- Allows you address compliance requirements and use your existing server-bound software licenses (per-socket, per-core, per-VM software licenses)
- Purchasing Options:
  - On-demand - pay per second for active Dedicated Host
  - Reserved - 1 or 3 years (No Upfront, Partial Upfront, All Upfront)
- The most expensive option
- Useful for software that have complicated licensing model (BYOL - Bring Your Own License)
- Or for companies that have strong regulatory or compliance needs

# EC2 Dedicated Instances

- Instances run on hardware that's dedicated to you
- May share hardware with other instances in same account
- No control over instance placement (can move hardware after Stop / Start)

| Characteristic   | Dedicated Instances | Dedicated Hosts |
|--|---------------------|-----------------|
| Enables the use of dedicated physical servers          | X                   | X               |
| Per instance billing (subject to a \$2 per region fee) | X                   |                 |
| Per host billing                                       |                     | X               |
| Visibility of sockets, cores, host ID                  |                     | X               |
| Affinity between a host and instance                   |                     | X               |
| Targeted instance placement                            |                     | X               |
| Automatic instance placement                           | X                   | X               |
| Add capacity using an allocation request               |                     | X               |

# EC2 Capacity Reservations

- Reserve On-Demand instances capacity in a specific AZ for any duration
- You always have access to EC2 capacity when you need it
- No time commitment (create/cancel anytime), no billing discounts
- Combine with Regional Reserved Instances and Savings Plans to benefit from billing discounts
- You're charged at On-Demand rate whether you run instances or not
- Suitable for short-term, uninterrupted workloads that needs to be in a specific AZ

# Which purchasing option is right for me?



- On demand: coming and staying in resort whenever we like, we pay the full price
- Reserved: like planning ahead and if we plan to stay for a long time, we may get a good discount.
- Savings Plans: pay a certain amount per hour for certain period and stay in any room type (e.g., King, Suite, Sea View, ... )
- Spot instances: the hotel allows people to bid for the empty rooms and the highest bidder keeps the rooms. You can get kicked out at any time
- Dedicated Hosts: We book an entire building of the resort
- Capacity Reservations: you book a room for a period with full price even you don't stay in it

# Price Comparison

## Example - m4.large - us-east-1

| Price Type                                    | Price (per hour)                             |
|---|--|
| On-Demand                                     | \$0.10                                       |
| Spot Instance (Spot Price)                    | \$0.038 - \$0.039 (up to 61% off)            |
| Reserved Instance (1 year)                    | \$0.062 (No Upfront) - \$0.058 (All Upfront) |
| Reserved Instance (3 years)                   | \$0.043 (No Upfront) - \$0.037 (All Upfront) |
| EC2 Savings Plan (1 year)                     | \$0.062 (No Upfront) - \$0.058 (All Upfront) |
| Reserved <b>Convertible</b> Instance (1 year) | \$0.071 (No Upfront) - \$0.066 (All Upfront) |
| Dedicated Host                                | On-Demand Price                              |
| Dedicated Host Reservation                    | Up to 70% off                                |
| Capacity Reservations                         | On-Demand Price                              |

# Shared Responsibility Model for EC2

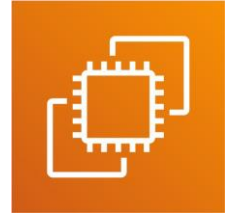


- Infrastructure (global network security)
- Isolation on physical hosts
- Replacing faulty hardware
- Compliance validation



- Security Groups rules
- Operating-system patches and updates
- Software and utilities installed on the EC2 instance
- IAM Roles assigned to EC2 & IAM user access management
- Data security on your instance

# EC2 Section - Summary



- EC2 Instance: AMI (OS) + Instance Size (CPU + RAM) + Storage + security groups + EC2 User Data
- Security Groups: Firewall attached to the EC2 instance
- EC2 User Data: Script launched at the first start of an instance
- SSH: start a terminal into our EC2 Instances (port 22)
- EC2 Instance Role: link to IAM roles
- Purchasing Options: On-Demand, Spot, Reserved (Standard + Convertible + Scheduled), Dedicated Host, Dedicated Instance

# EC2 Instance Storage Section



# What's an EBS Volume?



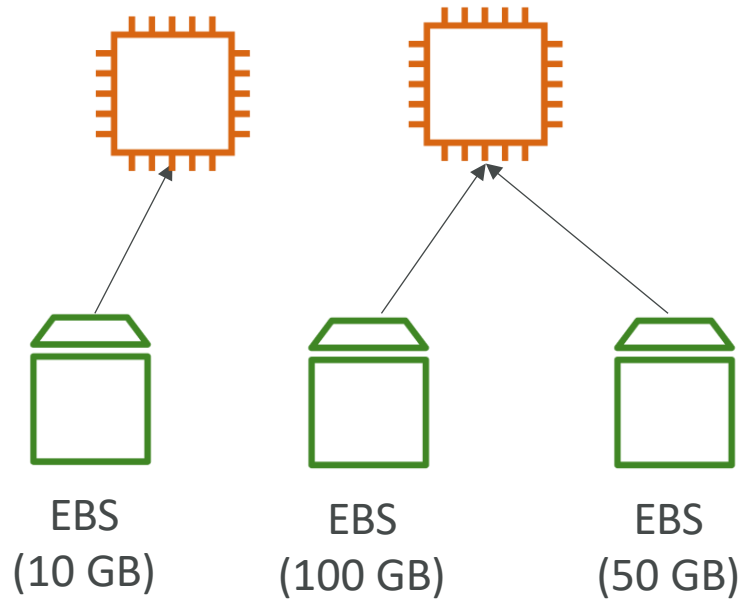
- An **EBS (Elastic Block Store) Volume** is a **network** drive you can attach to your instances while they run
- It allows your instances to persist data, even after their termination
- They can only be mounted to one instance at a time (at the CCP level)
- They are bound to a specific availability zone
- Analogy: Think of them as a “network USB stick”
- Free tier: 30 GB of free EBS storage of type General Purpose (SSD) or Magnetic per month

# EBS Volume

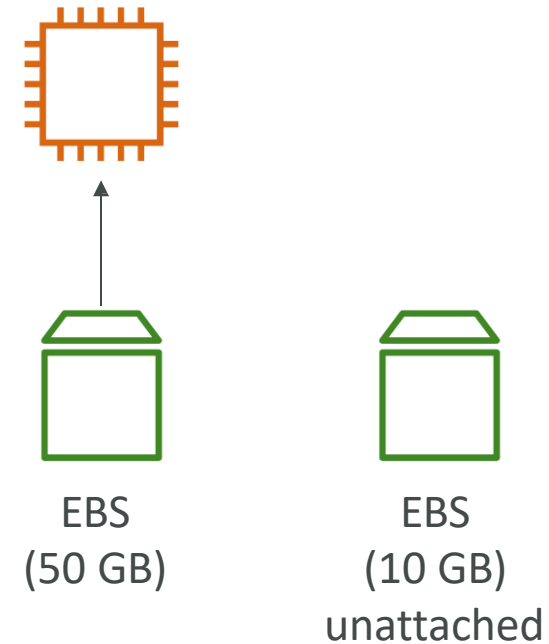
- It's a network drive (i.e. not a physical drive)
  - It uses the network to communicate the instance, which means there might be a bit of latency
  - It can be detached from an EC2 instance and attached to another one quickly
- It's locked to an Availability Zone (AZ)
  - An EBS Volume in us-east-1a cannot be attached to us-east-1b
  - To move a volume across, you first need to snapshot it
- Have a provisioned capacity (size in GBs, and IOPS)
  - You get billed for all the provisioned capacity
  - You can increase the capacity of the drive over time

# EBS Volume - Example

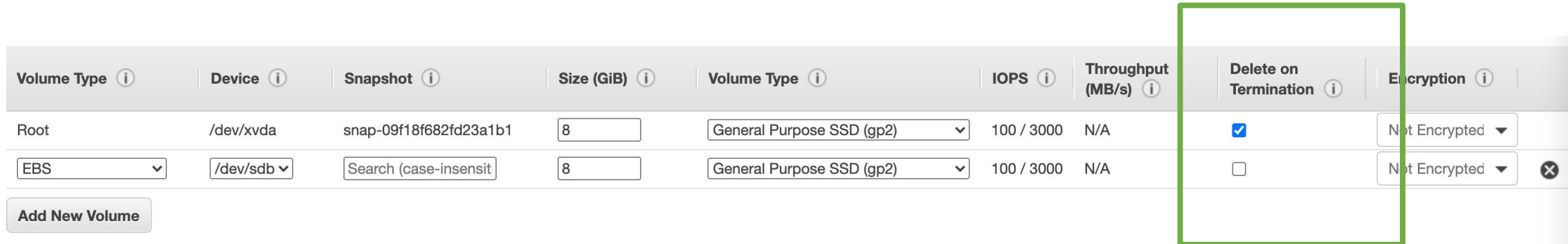
## US-EAST-1A



## US-EAST-1B



# EBS - Delete on Termination attribute



| Volume Type ⓘ | Device ⓘ   | Snapshot ⓘ             | Size (GiB) ⓘ | Volume Type ⓘ               | IOPS ⓘ     | Throughput (MB/s) ⓘ | Delete on Termination ⓘ             | Encryption ⓘ    |
|---------------|------------|------------------------|--------------|-----------------------------|------------|---------------------|-------------------------------------|-----------------|
| Root          | /dev/xvda  | snap-09f18f682fd23a1b1 | 8            | General Purpose SSD (gp2) ▼ | 100 / 3000 | N/A                 | <input checked="" type="checkbox"/> | Not Encrypted ▼ |
| EBS ▼         | /dev/sdb ▼ | Search (case-insensit  | 8            | General Purpose SSD (gp2) ▼ | 100 / 3000 | N/A                 | <input type="checkbox"/>            | Not Encrypted ▼ |

Add New Volume

- Controls the EBS behaviour when an EC2 instance terminates
  - By default, the root EBS volume is deleted (attribute enabled)
  - By default, any other attached EBS volume is not deleted (attribute disabled)
- This can be controlled by the AWS console / AWS CLI
- Use case: preserve root volume when instance is terminated

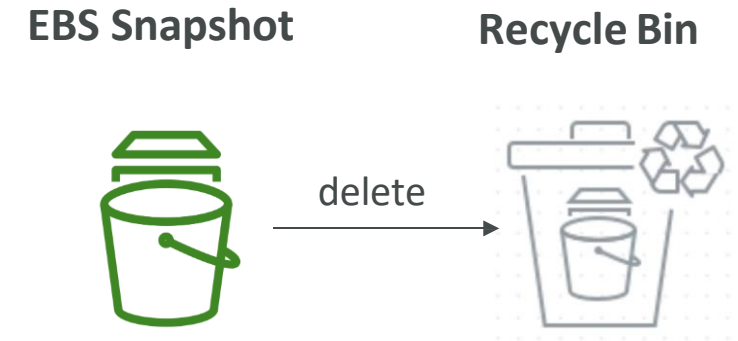
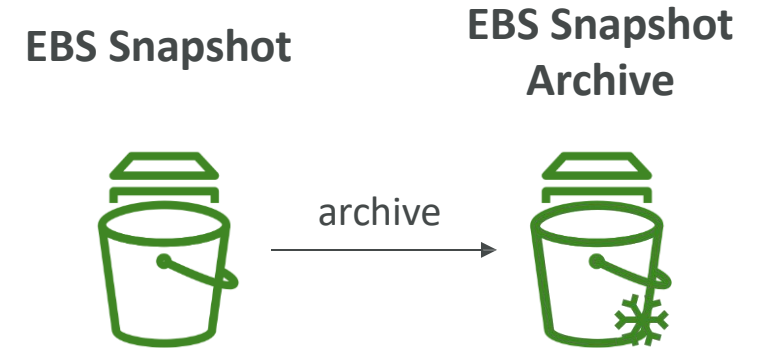
# EBS Snapshots

- Make a backup (snapshot) of your EBS volume at a point in time
- Not necessary to detach volume to do snapshot, but recommended
- Can copy snapshots across AZ or Region



# EBS Snapshots Features

- EBS Snapshot Archive
  - Move a Snapshot to an "archive tier" that is 75% cheaper
  - Takes within 24 to 72 hours for restoring the archive
- Recycle Bin for EBS Snapshots
  - Setup rules to retain deleted snapshots so you can recover them after an accidental deletion
  - Specify retention (from 1 day to 1 year)



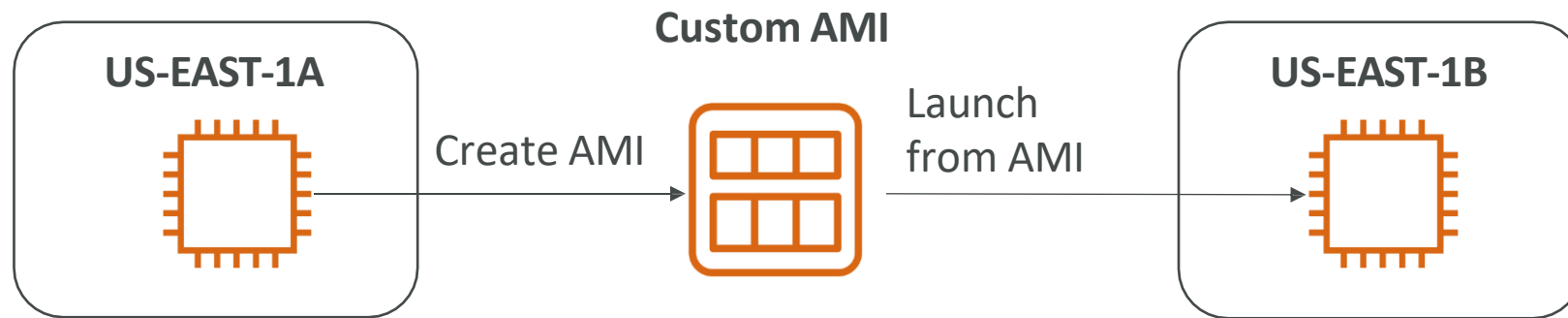
# AMI Overview



- AMI = Amazon Machine Image
- AMI are a customization of an EC2 instance
  - You add your own software, configuration, operating system, monitoring...
  - Faster boot / configuration time because all your software is pre-packaged
- AMI are built for a specific region (and can be copied across regions)
- You can launch EC2 instances from:
  - A Public AMI: AWS provided
  - Your own AMI: you make and maintain them yourself
  - An AWS Marketplace AMI: an AMI someone else made (and potentially sells)

# AMI Process (from an EC2 instance)

- Start an EC2 instance and customize it
- Stop the instance (for data integrity)
- Build an AMI - this will also create EBS snapshots
- Launch instances from other AMIs

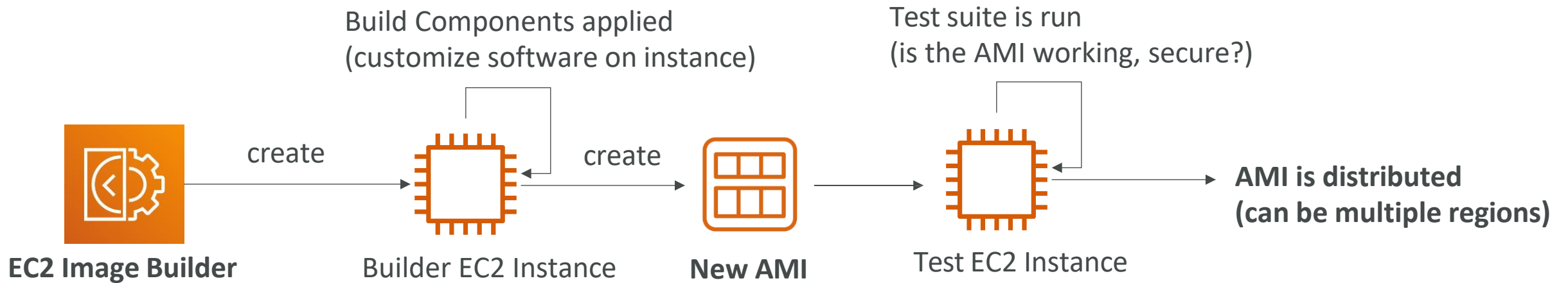




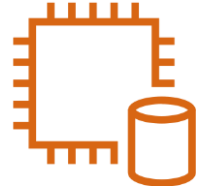
# EC2 Image Builder



- Used to automate the creation of Virtual Machines or container images
- => Automate the creation, maintain, validate and test EC2 AMIs
- Can be run on a schedule (weekly, whenever packages are updated, etc...)
- Free service (only pay for the underlying resources)



# EC2 Instance Store



- EBS volumes are network drives with good but “limited” performance
- If you need a high-performance hardware disk, use EC2 Instance Store
- Better I/O performance
- EC2 Instance Store lose their storage if they're stopped (ephemeral)
- Good for buffer / cache / scratch data / temporary content
- Risk of data loss if hardware fails
- Backups and Replication are your responsibility

# Local EC2 Instance Store

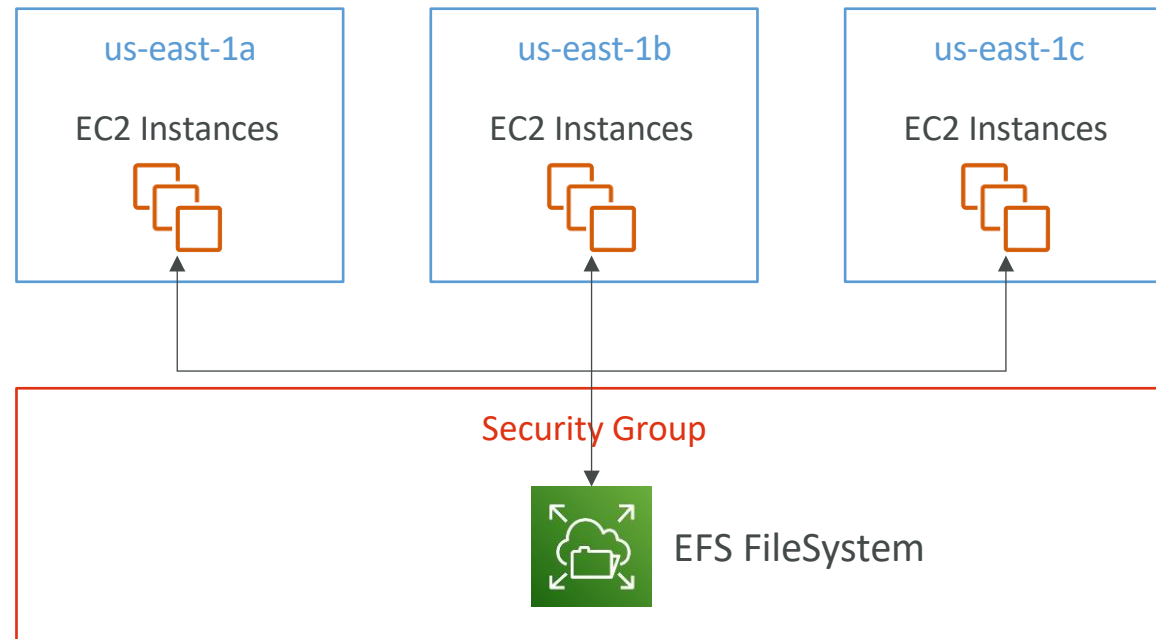
Very high IOPS

| Instance Size  | 100% Random Read IOPS | Write IOPS  |
|----------------|-----------------------|-------------|
| i3.large *     | 100,125               | 35,000      |
| i3.xlarge *    | 206,250               | 70,000      |
| i3.2xlarge     | 412,500               | 180,000     |
| i3.4xlarge     | 825,000               | 360,000     |
| i3.8xlarge     | 1.65 million          | 720,000     |
| i3.16xlarge    | 3.3 million           | 1.4 million |
| i3.metal       | 3.3 million           | 1.4 million |
| i3en.large *   | 42,500                | 32,500      |
| i3en.xlarge *  | 85,000                | 65,000      |
| i3en.2xlarge * | 170,000               | 130,000     |
| i3en.3xlarge   | 250,000               | 200,000     |
| i3en.6xlarge   | 500,000               | 400,000     |
| i3en.12xlarge  | 1 million             | 800,000     |
| i3en.24xlarge  | 2 million             | 1.6 million |
| i3en.metal     | 2 million             | 1.6 million |

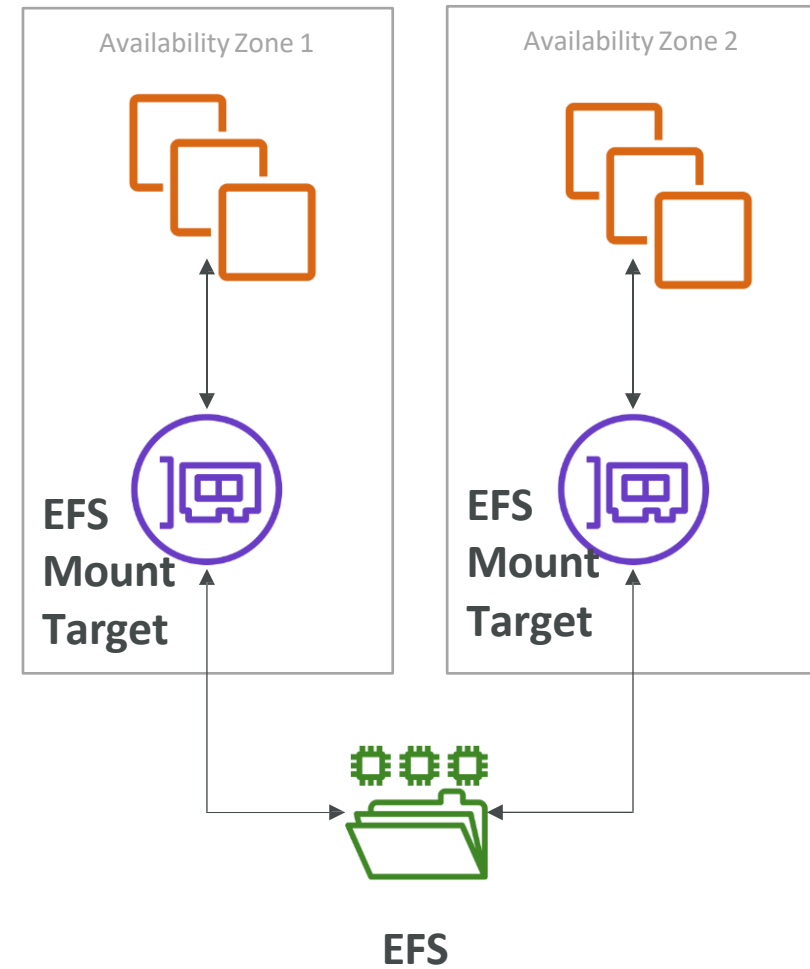
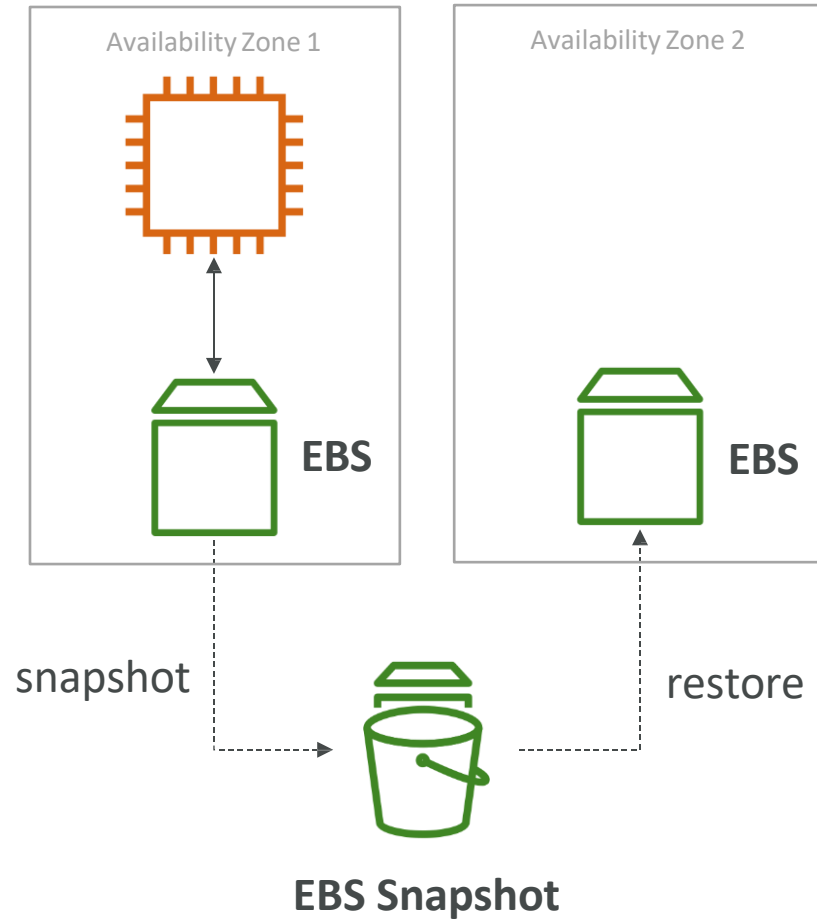
# EFS - Elastic File System



- Managed NFS (network file system) that can be mounted on 100s of EC2
- EFS works with Linux EC2 instances in multi-AZ
- Highly available, scalable, expensive (3x gp2), pay per use, no capacity planning

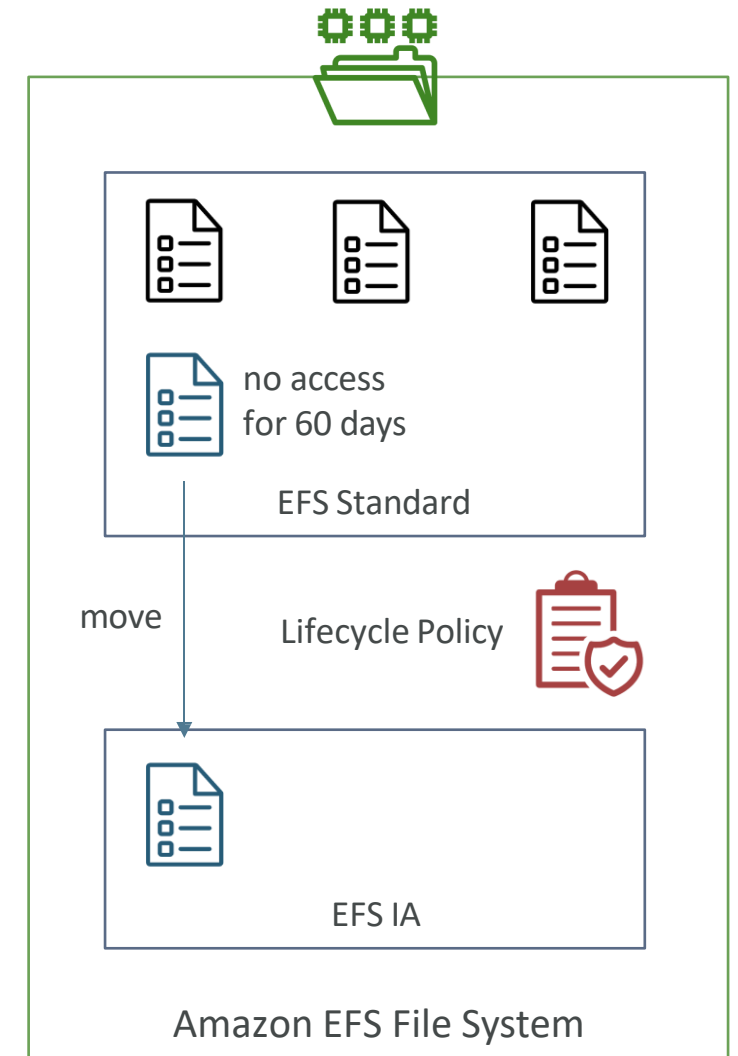


# EBS vs EFS



# EFS Infrequent Access (EFS-IA)

- Storage class that is cost-optimized for files not accessed every day
- Up to 92% lower cost compared to EFS Standard
- EFS will automatically move your files to EFS-IA based on the last time they were accessed
- Enable EFS-IA with a Lifecycle Policy
- Example: move files that are not accessed for 60 days to EFS-IA
- Transparent to the applications accessing EFS



# Shared Responsibility Model for EC2 Storage



- Infrastructure
- Replication for data for EBS volumes & EFS drives
- Replacing faulty hardware
- Ensuring their employees cannot access your data

- Setting up backup / snapshot procedures
- Setting up data encryption
- Responsibility of any data on the drives
- Understanding the risk of using EC2 Instance Store

# Amazon FSx - Overview



- Launch 3rd party high-performance file systems on AWS
- Fully managed service



**FSx for Lustre**



**FSx for  
Windows File  
Server**



**FSx for  
NetApp ONTAP**



# EC2 Instance Storage - Summary

- EBS volumes:
  - network drives attached to one EC2 instance at a time
  - Mapped to an Availability Zones
  - Can use EBS Snapshots for backups / transferring EBS volumes across AZ
- AMI: create ready-to-use EC2 instances with our customizations
- EC2 Image Builder: automatically build, test and distribute AMIs
- EC2 Instance Store:
  - High performance hardware disk attached to our EC2 instance
  - Lost if our instance is stopped / terminated
- EFS: network file system, can be attached to 100s of instances in a region
- EFS-IA: cost-optimized storage class for infrequent accessed files
- FSx for Windows: Network File System for Windows servers
- FSx for Lustre: High Performance Computing Linux file system

# Elastic Load Balancing & Auto Scaling Groups Section

# Scalability & High Availability

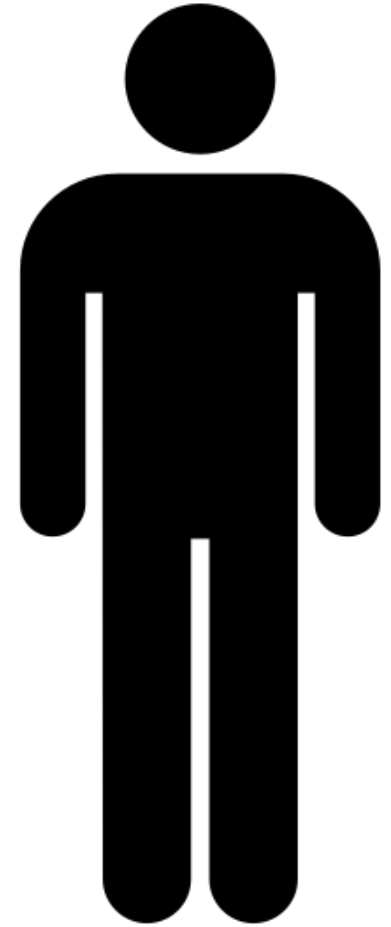
- Scalability means that an application / system can handle greater loads by adapting.
- There are two kinds of scalability:
  - Vertical Scalability
  - Horizontal Scalability (= elasticity)
- Scalability is linked but different to High Availability
- Let's deep dive into the distinction, using a call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



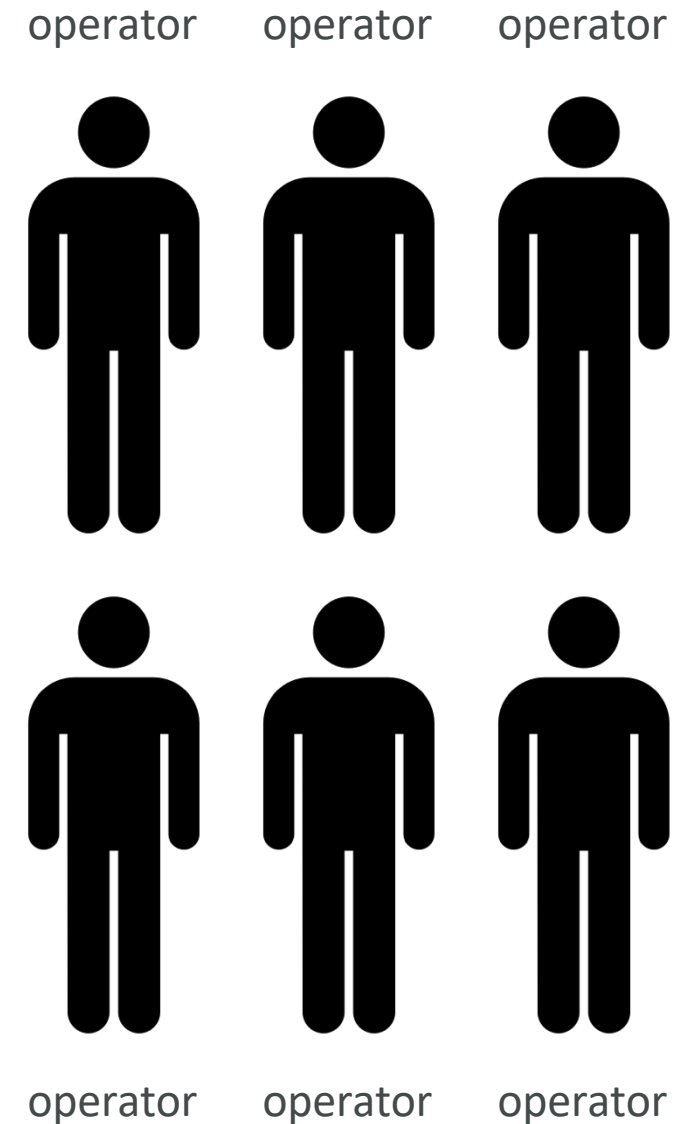
junior operator



senior operator

# Horizontal Scalability

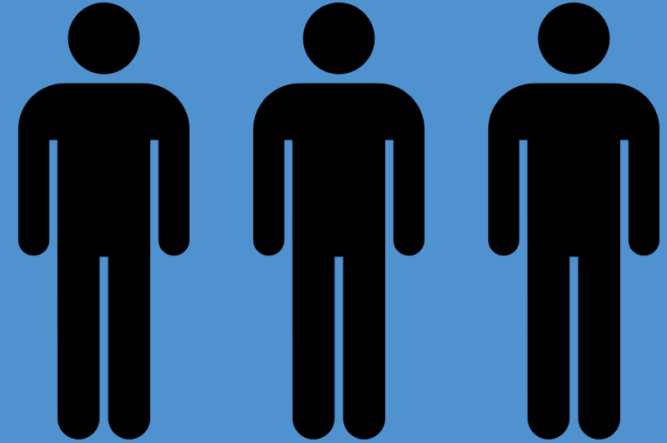
- Horizontal Scalability means increasing the number of instances / systems for your application
- Horizontal scaling implies distributed systems.
- This is very common for web applications / modern applications
- It's easy to horizontally scale thanks the cloud offerings such as Amazon EC2



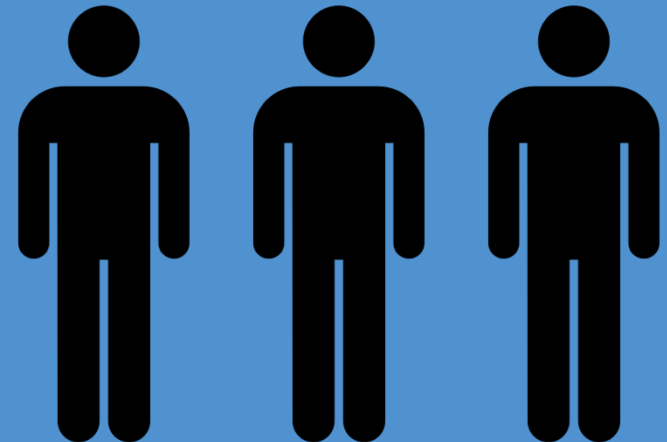
# High Availability

- 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)

first building in New York



second building in San Francisco



# 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)

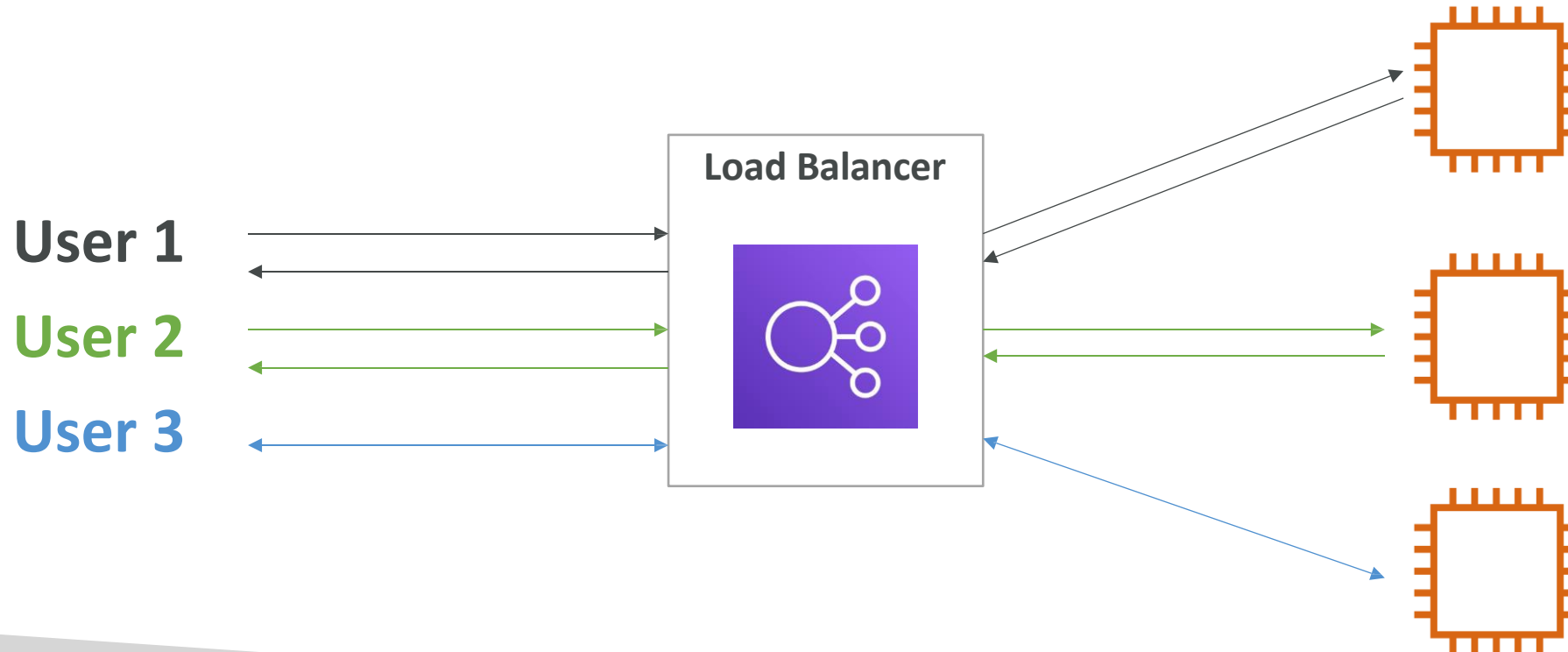
- Scalability: ability to accommodate a larger load by making the hardware stronger (scale up), or by adding nodes (scale out)
- Elasticity: once a system is scalable, elasticity means that there will be some “auto-scaling” so that the system can scale based on the load. This is “cloud-friendly”: pay-per-use, 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
- 

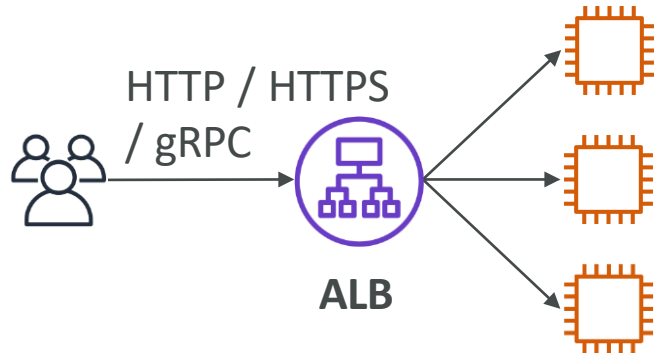
# Why use an Elastic Load Balancer?

- 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)
- 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

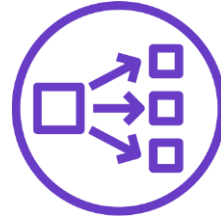
## Application Load Balancer



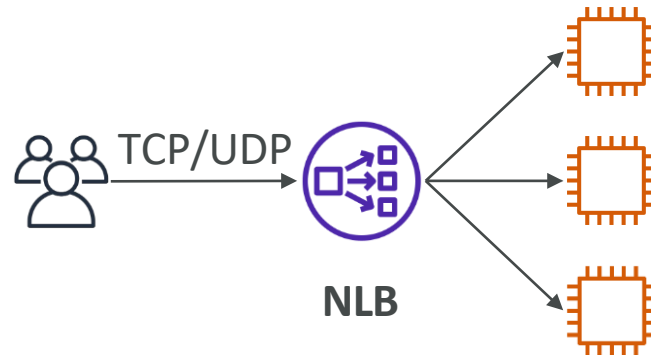
- HTTP / HTTPS / gRPC protocols (Layer 7)
- HTTP Routing features
- Static DNS (URL)



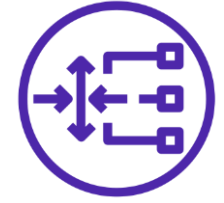
## Network Load Balancer



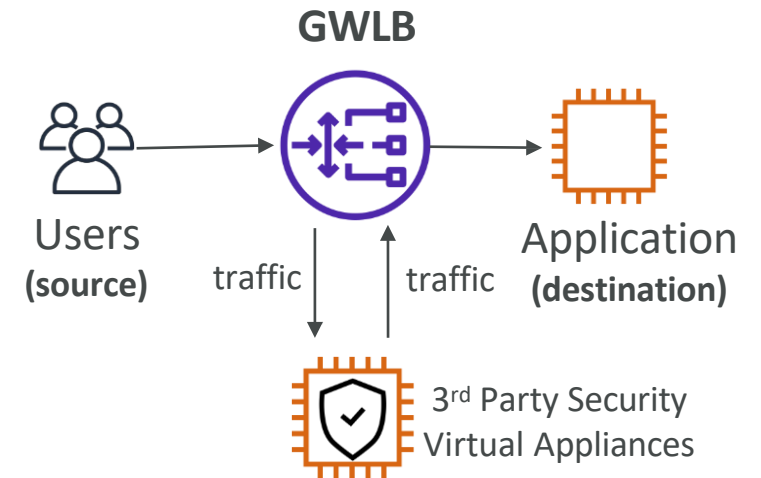
- TCP / UDP protocols (Layer 4)
- High Performance: millions of request per seconds
- Static IP through Elastic IP



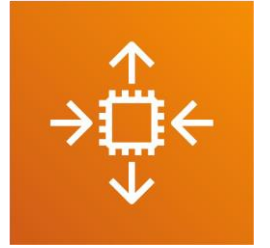
## Gateway Load Balancer



- GENEVE Protocol on IP Packets (Layer 3)
- Route Traffic to Firewalls that you manage on EC2 Instances
- Intrusion detection

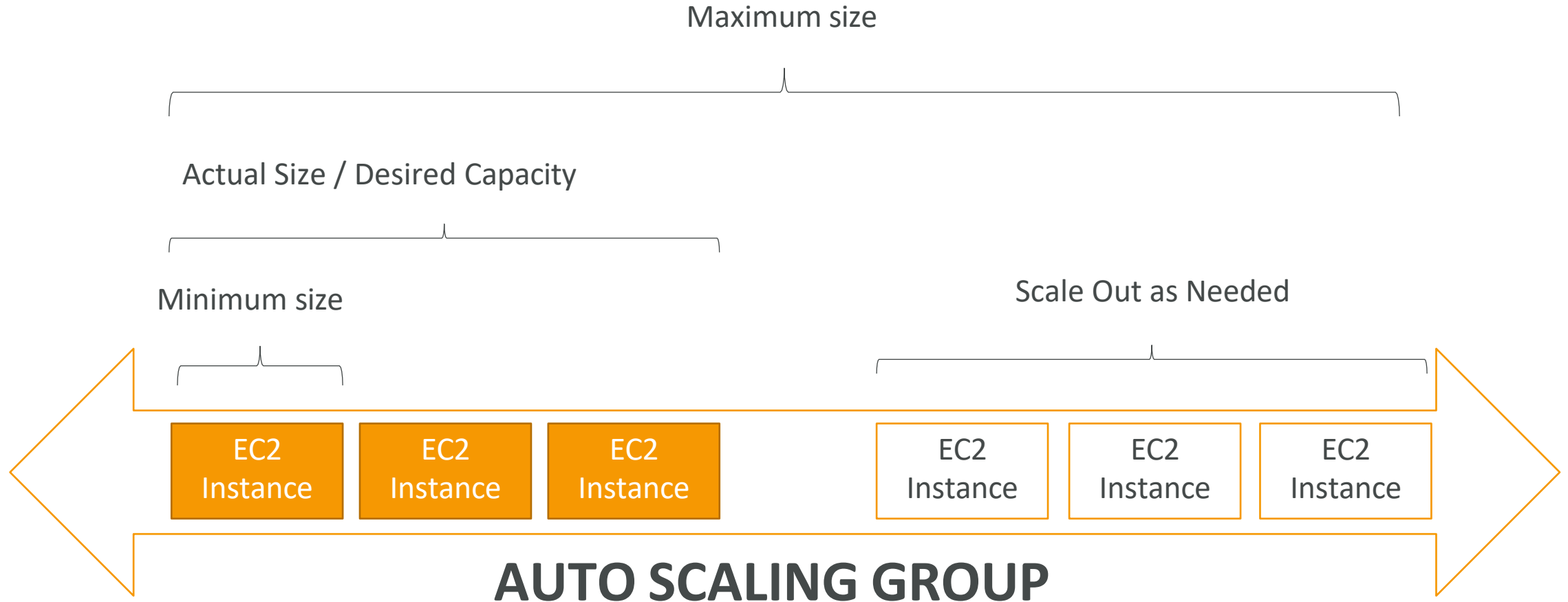


# What's an Auto Scaling Group?

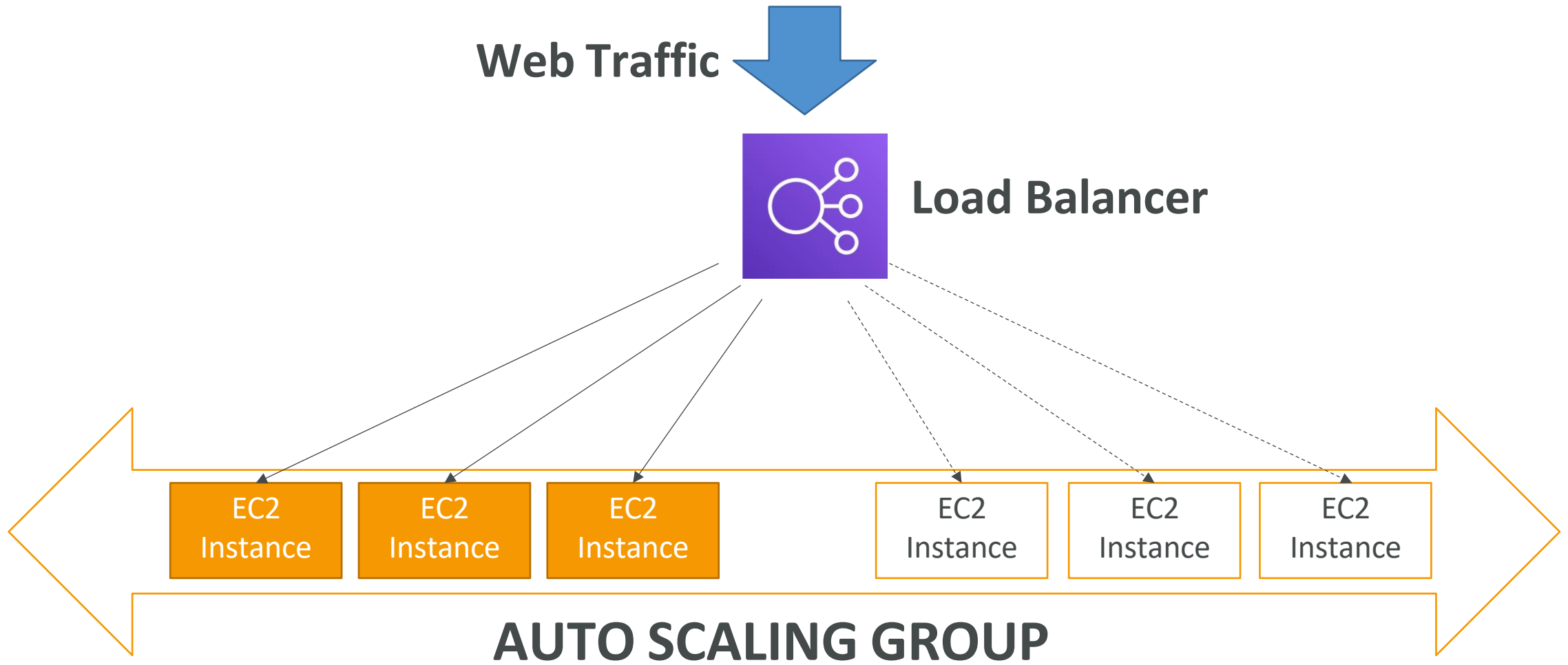


- In real-life, the load on your websites and application can change
- In the cloud, you can create and get rid of servers very quickly
- The goal of an Auto Scaling Group (ASG) is to:
  - Scale out (add EC2 instances) to match an increased load
  - Scale in (remove EC2 instances) to match a decreased 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