

# Disclaimer: These slides are copyrighted and strictly for personal use only

- This document is reserved for people enrolled into the [AWS Certified Cloud Practitioner Course](#)
- Please do not share this document, it is intended for personal use and exam preparation only, thank you.
- If you've obtained these slides for free on a website that is not the course's website, please reach out to [piracy@datacumulus.com](mailto:piracy@datacumulus.com). Thanks!
- Best of luck for the exam and happy learning!

# AWS Certified Cloud Practitioner Course

## CLF-C01

# Welcome! We're starting in 5 minutes



- We're going to prepare for the **Cloud Practitioner exam – CLF-C01**
  - It's a challenging certification, so this course will be long and interesting
  - Basic IT knowledge is helpful, but I will explain everything
- 
- We will cover over **40 AWS services** (out of the 200+ in AWS)
  - AWS / IT Beginners welcome! (but take your time, it's not a race)
  - **Learn by doing** – key learning technique!  
This course mixes both theory & hands on

# Sample question: Certified Cloud Practitioner

**Which AWS service would simplify the migration of a database to AWS?**

- A) AWS Storage Gateway                          <= we will learn
  - B) AWS Database Migration Service            <= correct answer
  - C) Amazon EC2                                  <= we will learn
  - D) Amazon AppStream 2.0                      <= distractor (over 200 services in AWS)
- 
- [https://d1.awsstatic.com/training-and-certification/docs-cloud-practitioner/AWS-Certified-Cloud-Practitioner\\_Sample-Questions.pdf](https://d1.awsstatic.com/training-and-certification/docs-cloud-practitioner/AWS-Certified-Cloud-Practitioner_Sample-Questions.pdf)

# About me

- I'm Stephane!
- 9x AWS Certified (so far!)
- Worked with AWS many years: built websites, apps, streaming platforms
- Veteran Instructor on AWS (Certifications, CloudFormation, Lambda, EC2...)
- You can find me on
  - LinkedIn: <https://www.linkedin.com/in/stephanemaarek>
  - Medium: <https://medium.com/@stephane.maarek>
  - Twitter: <https://twitter.com/stephanemaarek>
  - GitHub: <https://github.com/simplesteph>

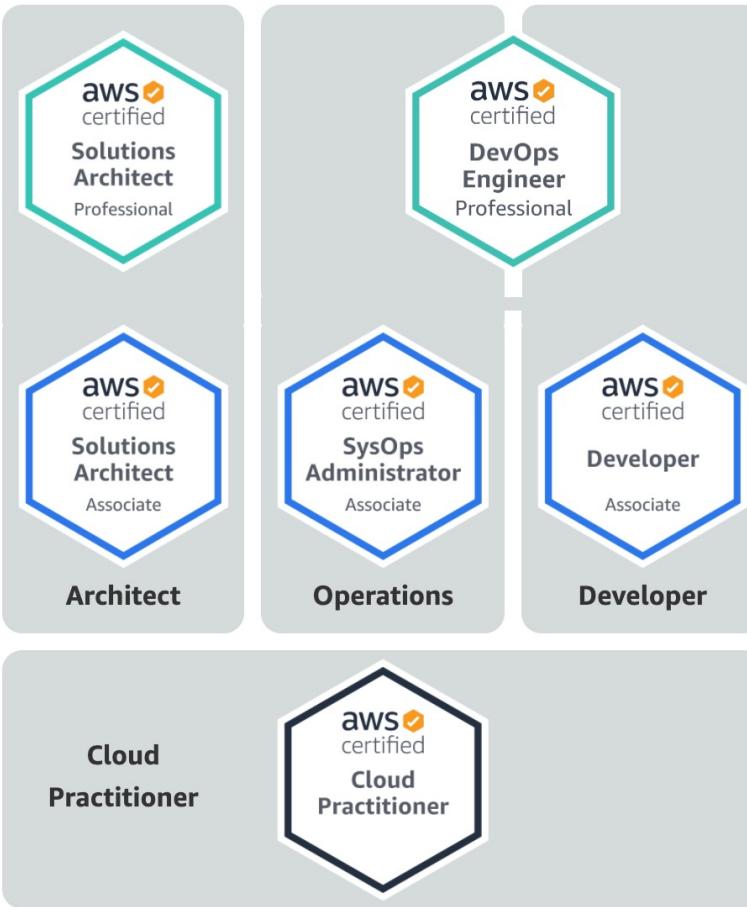


 **4.6** Instructor Rating  
 **107,634** Reviews  
 **356,650** Students  
 **31** Courses

# Your AWS Certification journey

## Professional

**Two years** of comprehensive experience designing, operating, and troubleshooting solutions using the AWS Cloud



## Associate

**One year** of experience solving problems and implementing solutions using the AWS Cloud

## Foundational

**Six months** of fundamental AWS Cloud and industry knowledge

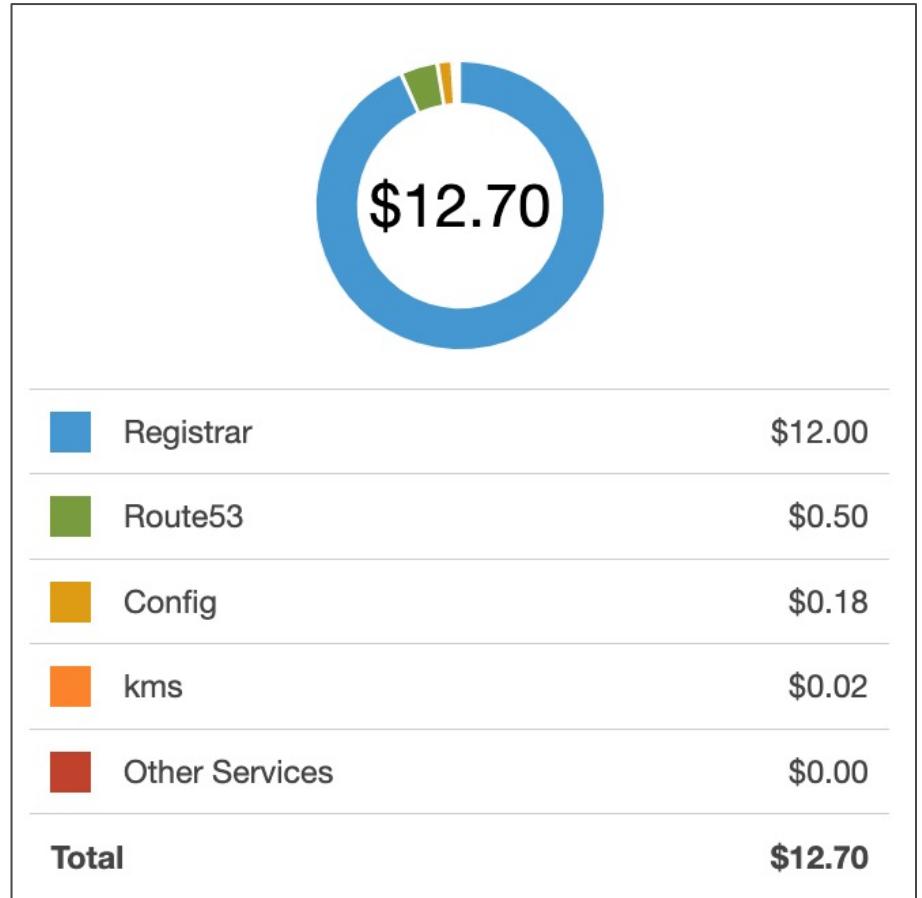
## Specialty

Technical AWS Cloud experience in the Specialty domain as specified in the **exam guide**



# Estimated Cost for this Course

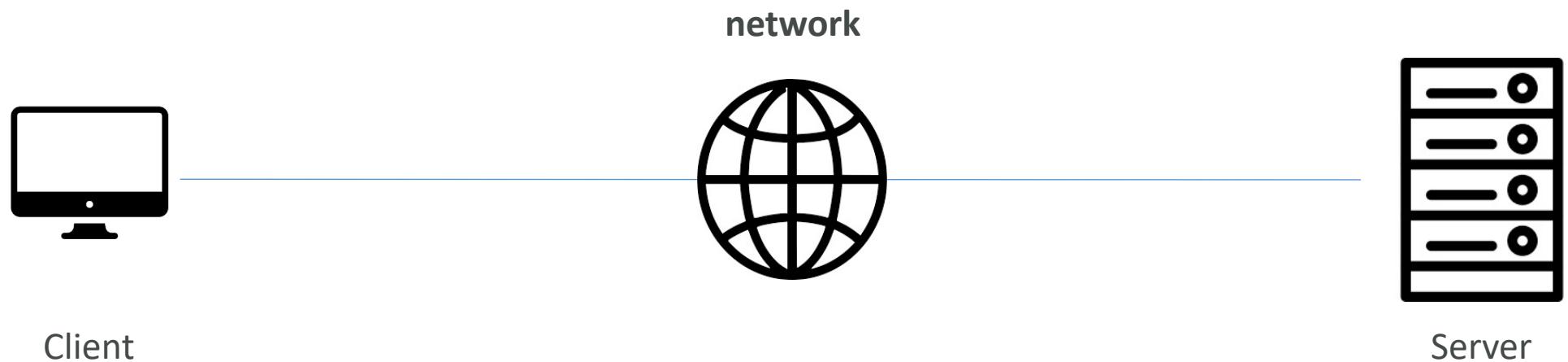
- Most of the services we'll use will be within the AWS Free Tier = \$0
- If I use a service which will cost you money, I will mention it
- You can read more about the AWS Free Tier at:  
<https://aws.amazon.com/free/>



# Udemy Tips

# What is Cloud Computing Section

# How websites work



Clients have IP addresses

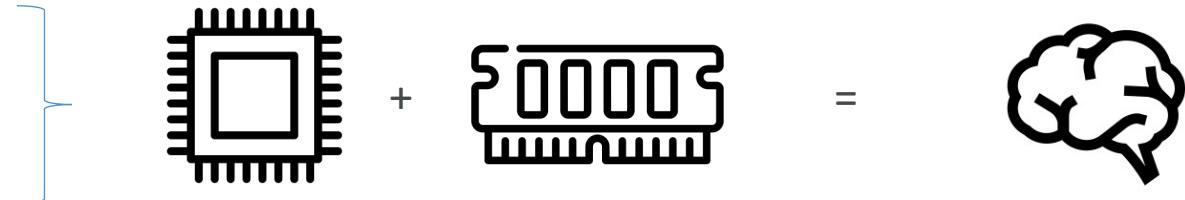
Servers have IP addresses

Just like when you're sending post mail!

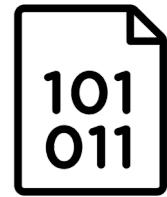


# What is a server composed of?

- Compute: CPU
- Memory: RAM



- Storage: Data



- Database: Store data in a structured way

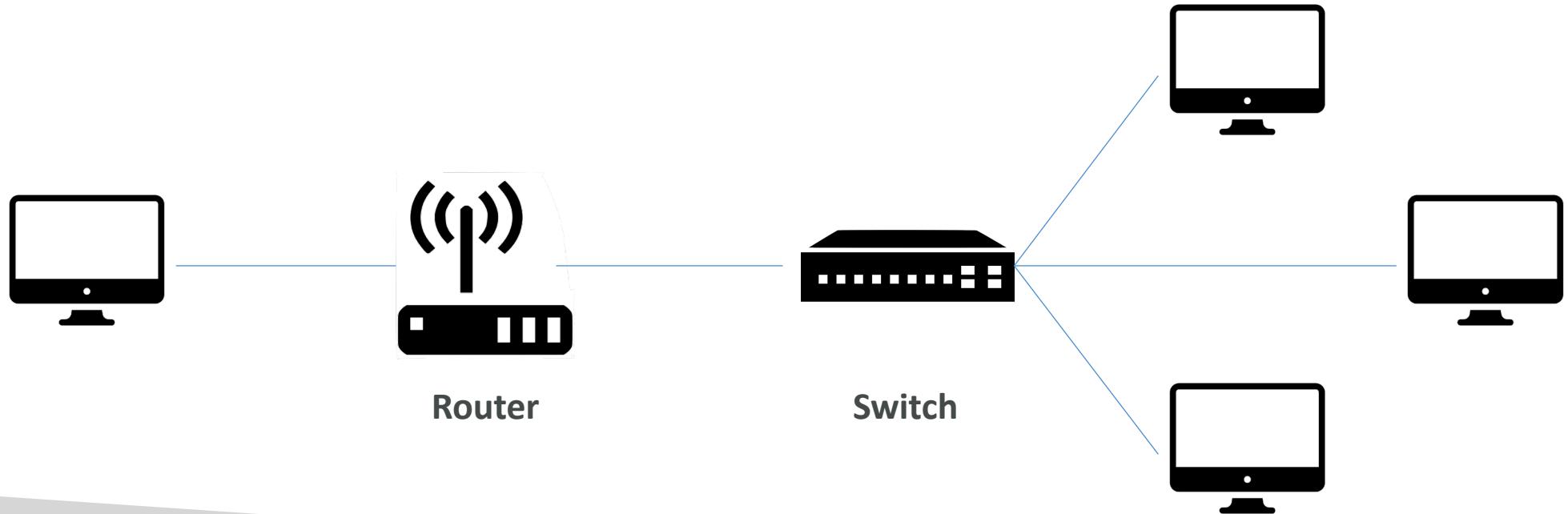


- Network: Routers, switch, DNS server

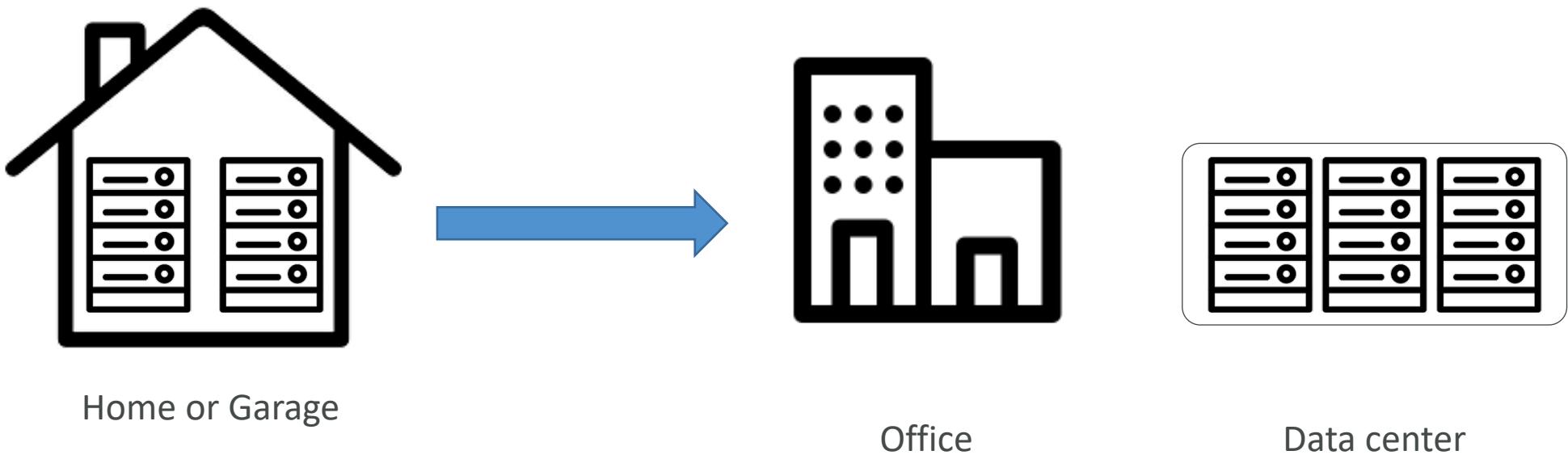


# IT Terminology

- **Network:** cables, routers and servers connected with each other
- **Router:** A networking device that forwards data packets between computer networks. They know where to send your packets on the internet!
- **Switch:** Takes a packet and send it to the correct server / client on your network

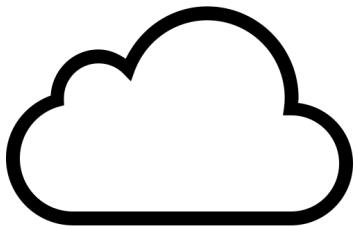


# Traditionally, how to build infrastructure



# Problems with traditional IT approach

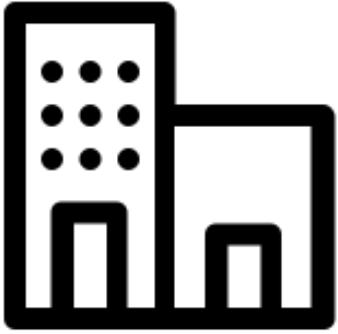
- Pay for the rent for the data center
- Pay for power supply, cooling, maintenance
- Adding and replacing hardware takes time
- Scaling is limited
- Hire 24/7 team to monitor the infrastructure
- How to deal with disasters? (earthquake, power shutdown, fire...)
- Can we externalize all this?



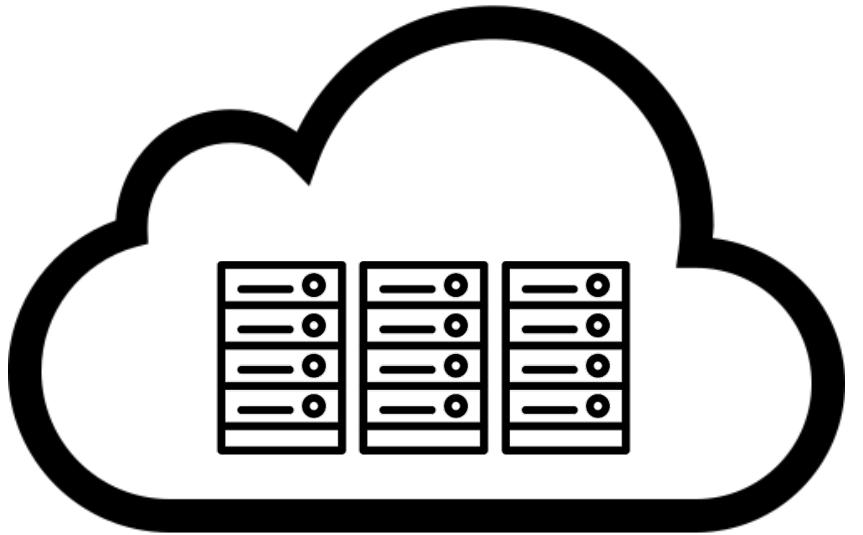
# What is Cloud Computing?



- 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.



Office



The Cloud

# You've been using some Cloud services



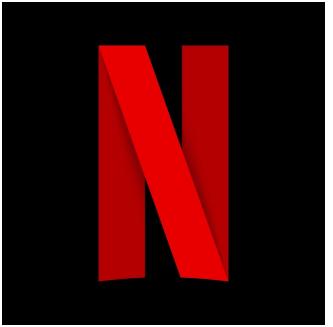
## Gmail

- E-mail cloud service
- Pay for ONLY your emails stored (no infrastructure, etc.)



## Dropbox

- Cloud Storage Service
- Originally built on AWS



## Netflix

- Built on AWS
- Video on Demand

# 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



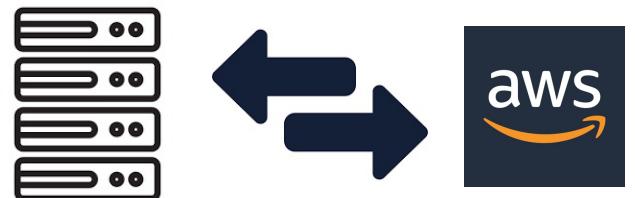
## Public Cloud:

- 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

# Problems solved by the Cloud

- **Flexibility:** change resource types when needed
- **Cost-Effectiveness:** pay as you go, for what you use
- **Scalability:** accommodate larger loads by making hardware stronger or adding additional nodes
- **Elasticity:** ability to scale out and scale-in when needed
- **High-availability and fault-tolerance:** build across data centers
- **Agility:** rapidly develop, test and launch software applications

# Types of Cloud Computing

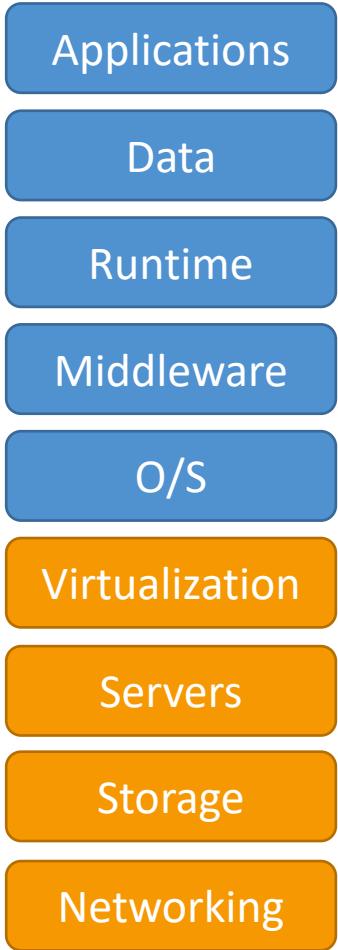
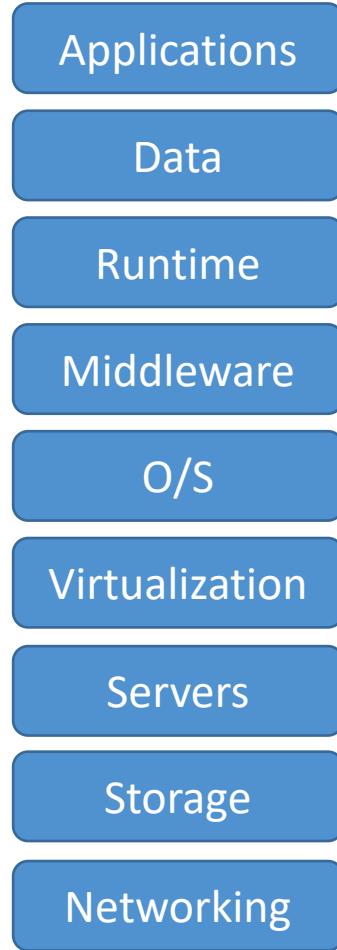
- **Infrastructure as a Service (IaaS)**
  - Provide building blocks for cloud IT
  - Provides networking, computers, data storage space
  - Highest level of flexibility
  - Easy parallel with traditional on-premises IT
- **Platform as a Service (PaaS)**
  - Removes the need for your organization to 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)

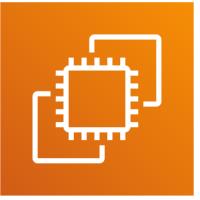


Managed by you

Managed by others

# Example of Cloud Computing Types

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

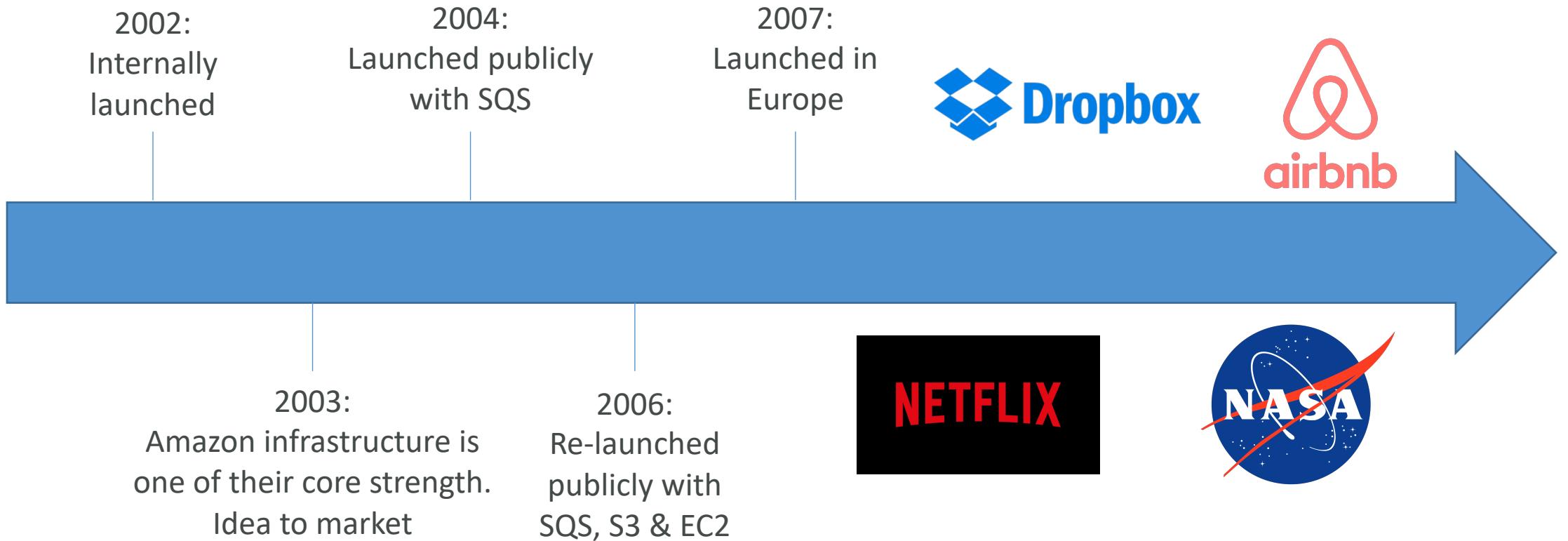


# Pricing of the Cloud – Quick Overview

- AWS has 3 pricing fundamentals, following the pay-as-you-go pricing model
- **Compute:**
  - Pay for compute time
- **Storage:**
  - Pay for data stored in the Cloud
- **Data transfer OUT of the Cloud:**
  - Data transfer IN is free
  - Solves the expensive issue of traditional IT



# AWS Cloud History



# AWS Cloud Number Facts

- In 2019, AWS had \$35.02 billion in annual revenue
- AWS accounts for 47% of the market in 2019 (Microsoft is 2nd with 22%)
- Pioneer and Leader of the AWS Cloud Market for the 9th consecutive year
- Over 1,000,000 active users

Figure 1. Magic Quadrant for Cloud Infrastructure as a Service, Worldwide



Gartner Magic Quadrant

# 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



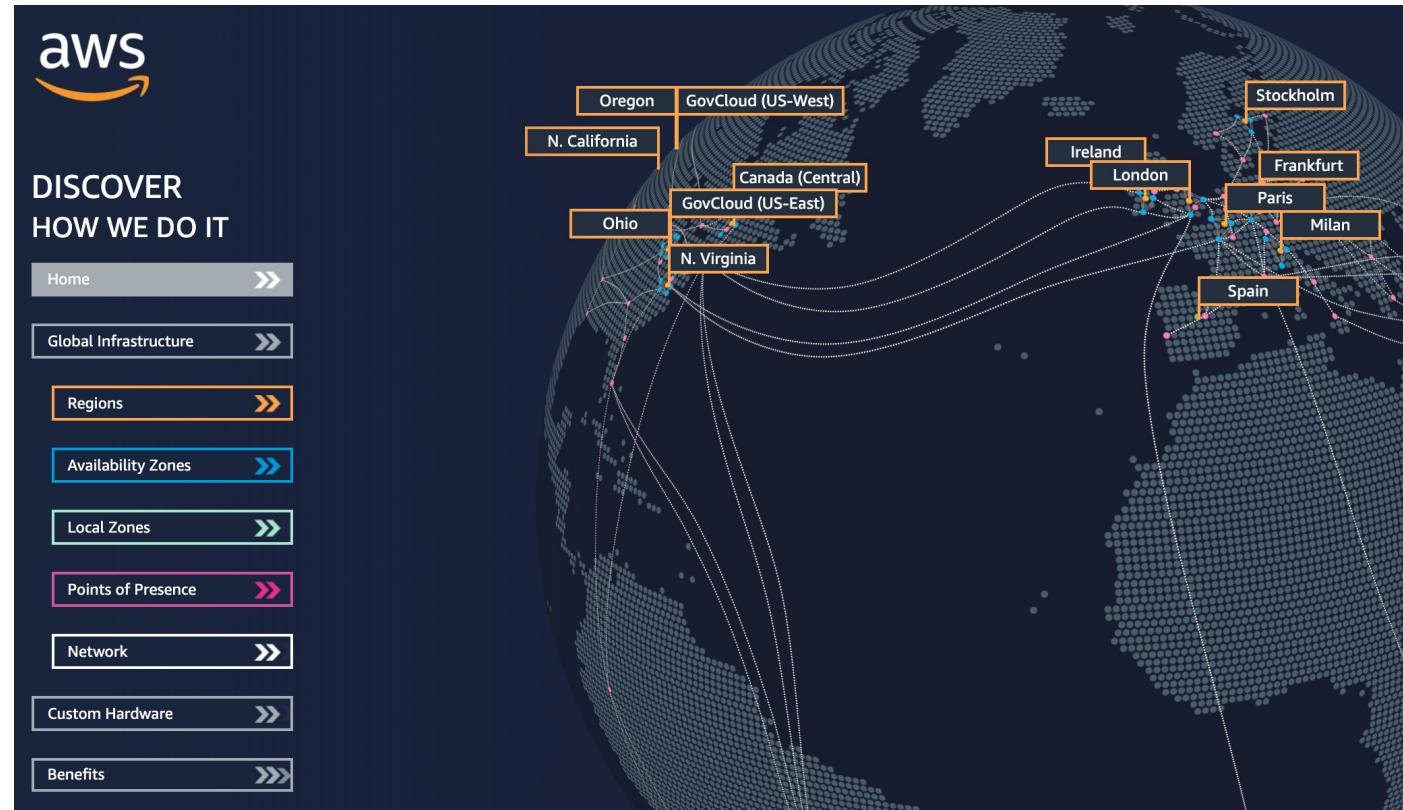
21ST  
CENTURY  
FOX

ACTIVISION



# 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

# How to choose an AWS Region?

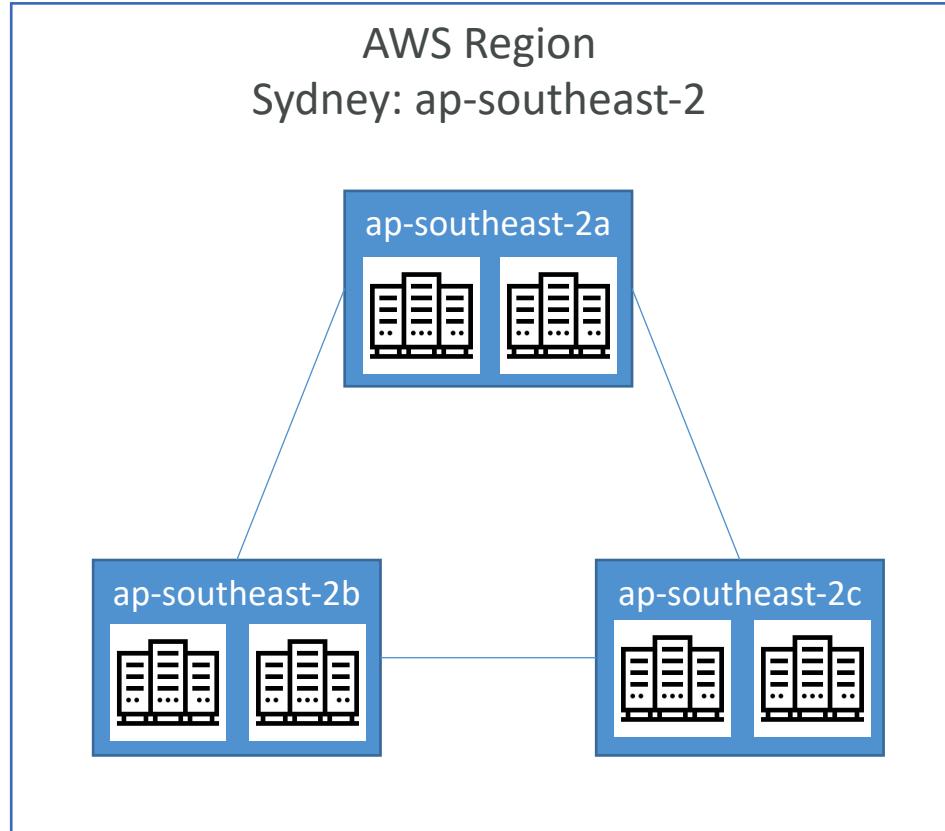
If you need to launch a new application,  
where should you do it?



- **Compliance** with data governance and legal requirements: data never leaves a region without your explicit permission
- **Proximity** to customers: reduced latency
- **Available services** within a Region: new services and new features aren't available in every Region
- **Pricing**: pricing varies region to region and is transparent in the service pricing page

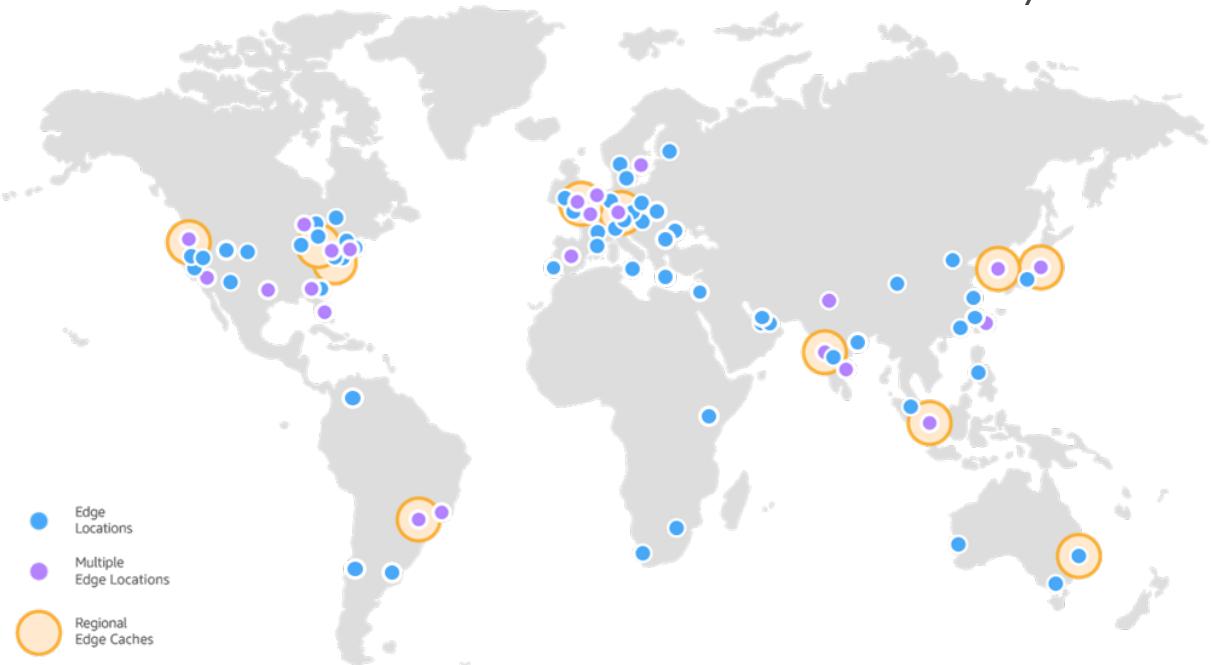
# AWS Availability Zones

- Each region has many availability zones (usually 3, min is 2, max is 6). Example:
  - ap-southeast-2a
  - ap-southeast-2b
  - ap-southeast-2c
- Each availability zone (AZ) is one or more discrete data centers with redundant power, networking, and connectivity
- They're separate from each other, so that they're isolated from disasters
- They're connected with high bandwidth, ultra-low latency networking



# 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

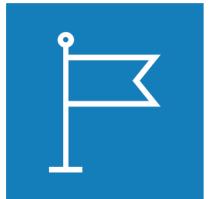


<https://aws.amazon.com/cloudfront/features/>

# Tour of the AWS Console



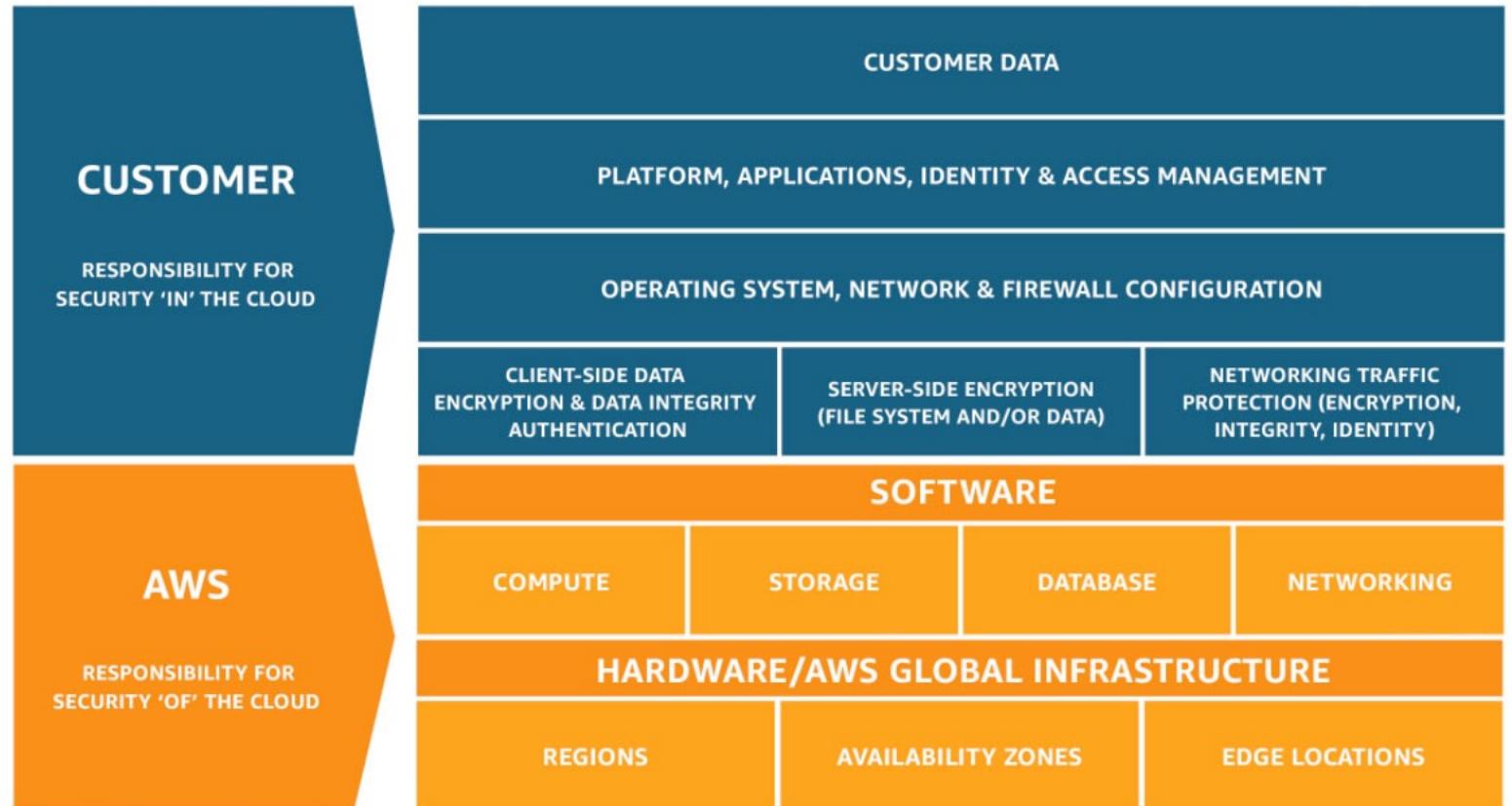
- AWS has Global Services:
  - Identity and Access Management (IAM)
  - Route 53 (DNS service)
  - CloudFront (Content Delivery Network)
  - WAF (Web Application Firewall)
- Most AWS services are Region-scoped:
  - Amazon EC2 (Infrastructure as a Service)
  - Elastic Beanstalk (Platform as a Service)
  - Lambda (Function as a Service)
  - Rekognition (Software as a Service)
- Region Table: <https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services>



# Shared Responsibility Model diagram

CUSTOMER = RESPONSIBILITY FOR THE SECURITY IN THE CLOUD

AWS = RESPONSIBILITY FOR THE SECURITY OF THE CLOUD



<https://aws.amazon.com/compliance/shared-responsibility-model/>

# AWS Acceptable Use Policy

- <https://aws.amazon.com/aup/>
- No Illegal, Harmful, or Offensive Use or Content
- No Security Violations
- No Network Abuse
- No E-Mail or Other Message Abuse

# 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



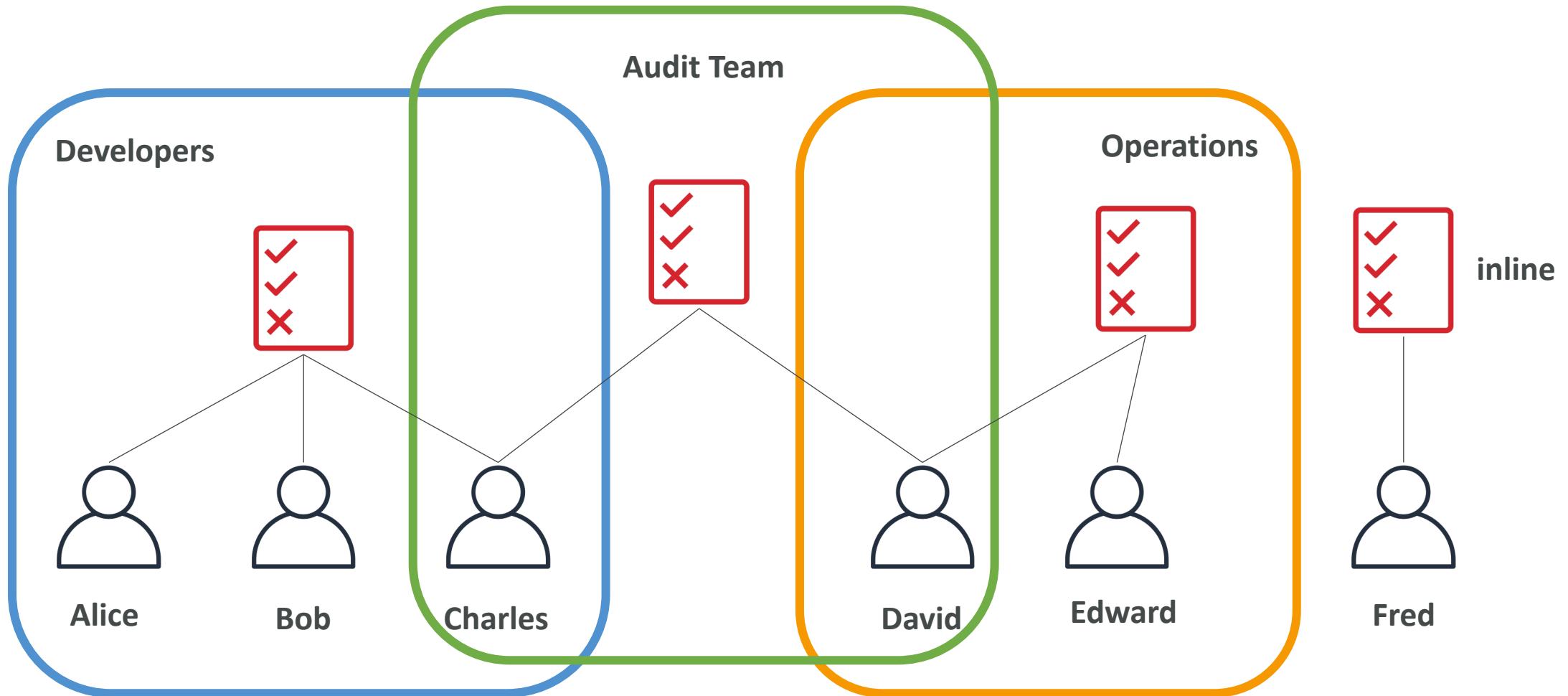
# IAM: Permissions

- 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 Policies Structure

- Consists of
  - **Version:** policy language version, always include “2012-10-17”
  - **Id:** an identifier for the policy (optional)
  - **Statement:** one or more individual statements (required)
- Statements consists of
  - **Sid:** an identifier for the statement (optional)
  - **Effect:** whether the statement allows or denies access (Allow, Deny)
  - **Principal:** account/user/role to which this policy applied to
  - **Action:** list of actions this policy allows or denies
  - **Resource:** list of resources to which the actions applied to
  - **Condition:** conditions for when this policy is in effect (optional)

```
{  
  "Version": "2012-10-17",  
  "Id": "S3-Account-Permissions",  
  "Statement": [  
    {  
      "Sid": "1",  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": ["arn:aws:iam::123456789012:root"]  
      },  
      "Action": [  
        "s3:GetObject",  
        "s3:PutObject"  
      ],  
      "Resource": ["arn:aws:s3:::mybucket/*"]  
    }  
  ]  
}
```

# 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



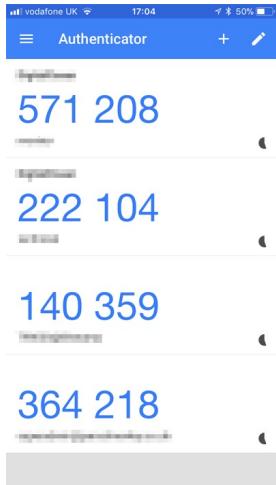
- Users have access to your account and can possibly change configurations or delete resources in your AWS account
- You want to protect your Root Accounts and IAM users
- MFA = password you know + security device you own



- 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)

Support for multiple tokens on a single device.



Authy  
(multi-device)

## Universal 2nd Factor (U2F) Security Key



YubiKey by Yubico (3<sup>rd</sup> party)

Support for multiple root and IAM users using a single security key

# MFA devices options in AWS

## Hardware Key Fob MFA Device



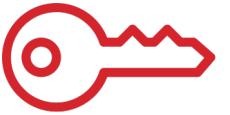
Provided by Gemalto (3<sup>rd</sup> party)

## Hardware Key Fob MFA Device for AWS GovCloud (US)



Provided by SurePassID (3<sup>rd</sup> party)

# 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 (Fake) 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> <a href="#">X</a>

- Access key ID: AKIASK4E37PV4983d6C
- Secret Access Key: AZPN3z0jWozWCndljhB0Uh8239aIbzBzO5fqkZq
- 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
→ ~ █
```

# What's the AWS SDK?

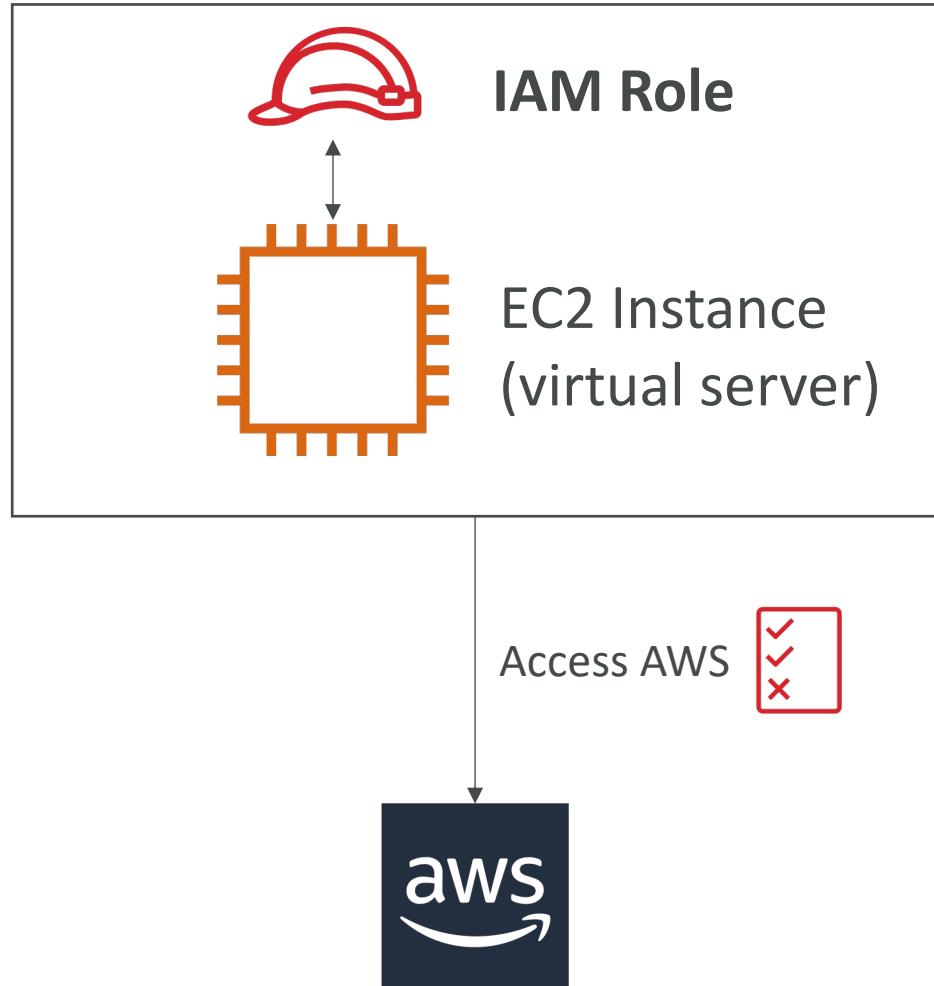


- AWS Software Development Kit (AWS SDK)
- Language-specific APIs (set of libraries)
- Enables you to access and manage AWS services programmatically
- Embedded within your application
- Supports
  - SDKs (JavaScript, Python, PHP, .NET, Ruby, Java, Go, Node.js, C++)
  - Mobile SDKs (Android, iOS, ...)
  - IoT Device SDKs (Embedded C, Arduino, ...)
- Example: AWS CLI is built on AWS SDK for Python



# 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

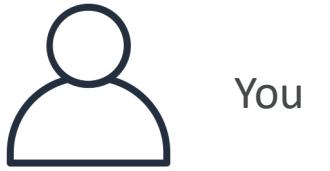
- **IAM Credentials Report (account-level)**
  - a report that lists all your account's users and the status of their various credentials
- **IAM Access Advisor (user-level)**
  - Access advisor shows the service permissions granted to a user and when those services were last accessed.
  - You can use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



You

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

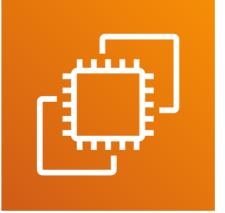
# IAM Section – Summary



- **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' offering
- 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 – General Purpose

- Great for a diversity of workloads such as web servers or code repositories
- Balance between:
  - Compute
  - Memory
  - Networking
- In the course, we will be using the t2.micro which is a General Purpose EC2 instance

## General Purpose

General purpose instances provide a balance of compute, memory and networking resources, and can be used for a variety of diverse workloads. These instances are ideal for applications that use these resources in equal proportions such as web servers and code repositories.

Mac	T4g	T3	T3a	T2	M6g	M5	M5a	M5n	M5zn	M4	A1
-----	-----	----	-----	----	-----	----	-----	-----	------	----	----

\* this list will evolve over time, please check the AWS website for the latest information

# EC2 Instance Types – Compute Optimized

- Great for compute-intensive tasks that require high performance processors:
  - Batch processing workloads
  - Media transcoding
  - High performance web servers
  - High performance computing (HPC)
  - Scientific modeling & machine learning
  - Dedicated gaming servers

## **Compute Optimized**

Compute Optimized Instances are ideal for compute bound applications that benefit from high performance processors. Instances belonging to this family are well suited for batch processing workloads, media transcoding, high performance web servers, high performance computing (HPC), scientific modeling, dedicated gaming servers and ad server engines, machine learning inference and other compute intensive applications.

C6g    C6gn    C5    C5a    C5n    C4

\* this list will evolve over time, please check the AWS website for the latest information

# EC2 Instance Types – Memory Optimized

- Fast performance for workloads that process large data sets in memory
- Use cases:
  - High performance, relational/non-relational databases
  - Distributed web scale cache stores
  - In-memory databases optimized for BI (business intelligence)
  - Applications performing real-time processing of big unstructured data

## Memory Optimized

Memory optimized instances are designed to deliver fast performance for workloads that process large data sets in memory.

R6g

R5

R5a

R5b

R5n

R4

X1e

X1

High Memory

z1d

\* this list will evolve over time, please check the AWS website for the latest information

# EC2 Instance Types – Storage Optimized

- Great for storage-intensive tasks that require high, sequential read and write access to large data sets on local storage
- Use cases:
  - High frequency online transaction processing (OLTP) systems
  - Relational & NoSQL databases
  - Cache for in-memory databases (for example, Redis)
  - Data warehousing applications
  - Distributed file systems

## Storage Optimized

Storage optimized instances are designed for workloads that require high, sequential read and write access to very large data sets on local storage. They are optimized to deliver tens of thousands of low-latency, random I/O operations per second (IOPS) to applications.

I3    I3en    D2    D3    D3en    H1

\* this list will evolve over time, please check the AWS website for the latest information

# EC2 Instance Types: example

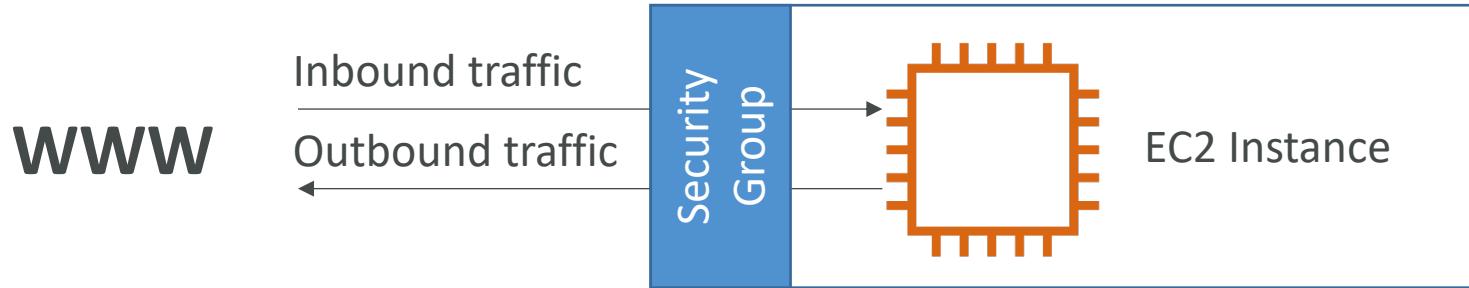
Instance	vCPU	Mem (GiB)	Storage	Network Performance	EBS Bandwidth (Mbps)
t2.micro	1	1	EBS-Only	Low to Moderate	
t2.xlarge	4	16	EBS-Only	Moderate	
c5d.4xlarge	16	32	1 x 400 NVMe SSD	Up to 10 Gbps	4,750
r5.16xlarge	64	512	EBS Only	20 Gbps	13,600
m5.8xlarge	32	128	EBS Only	10 Gbps	6,800

**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

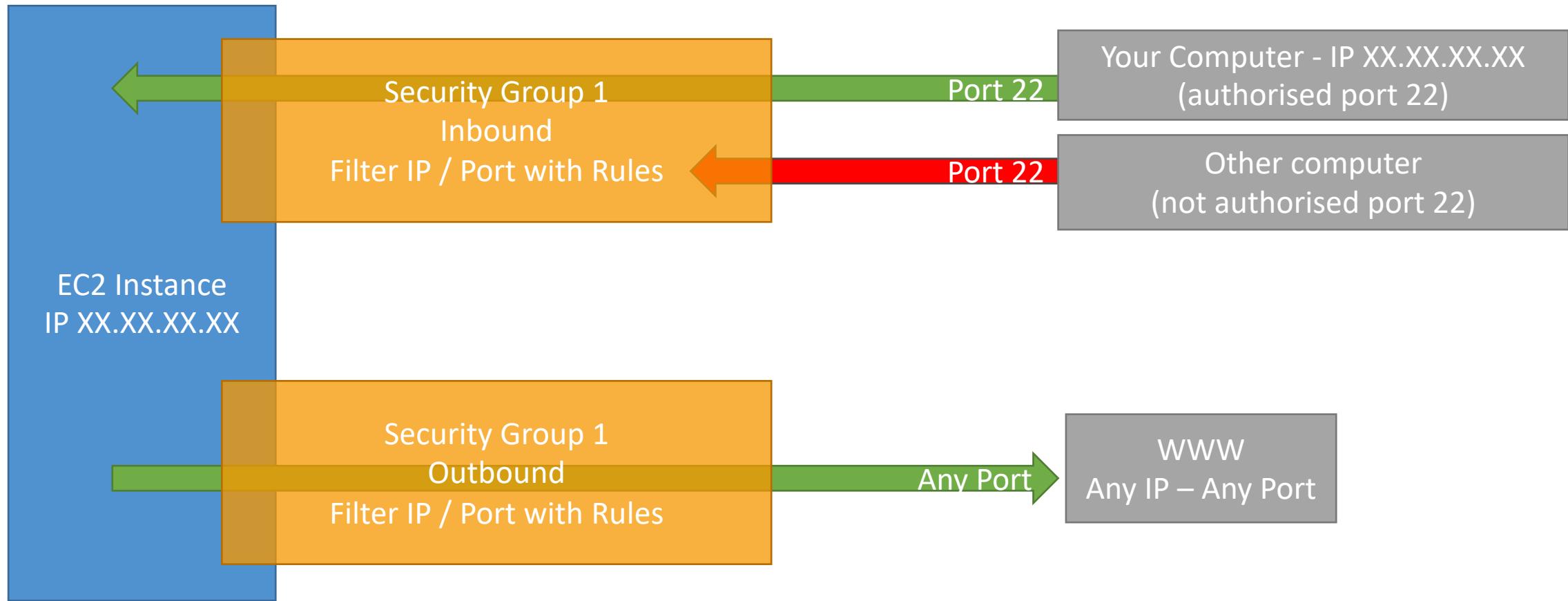
# Security Groups

## 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 <span>i</span>	Protocol <span>i</span>	Port Range <span>i</span>	Source <span>i</span>	Description <span>i</span>
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



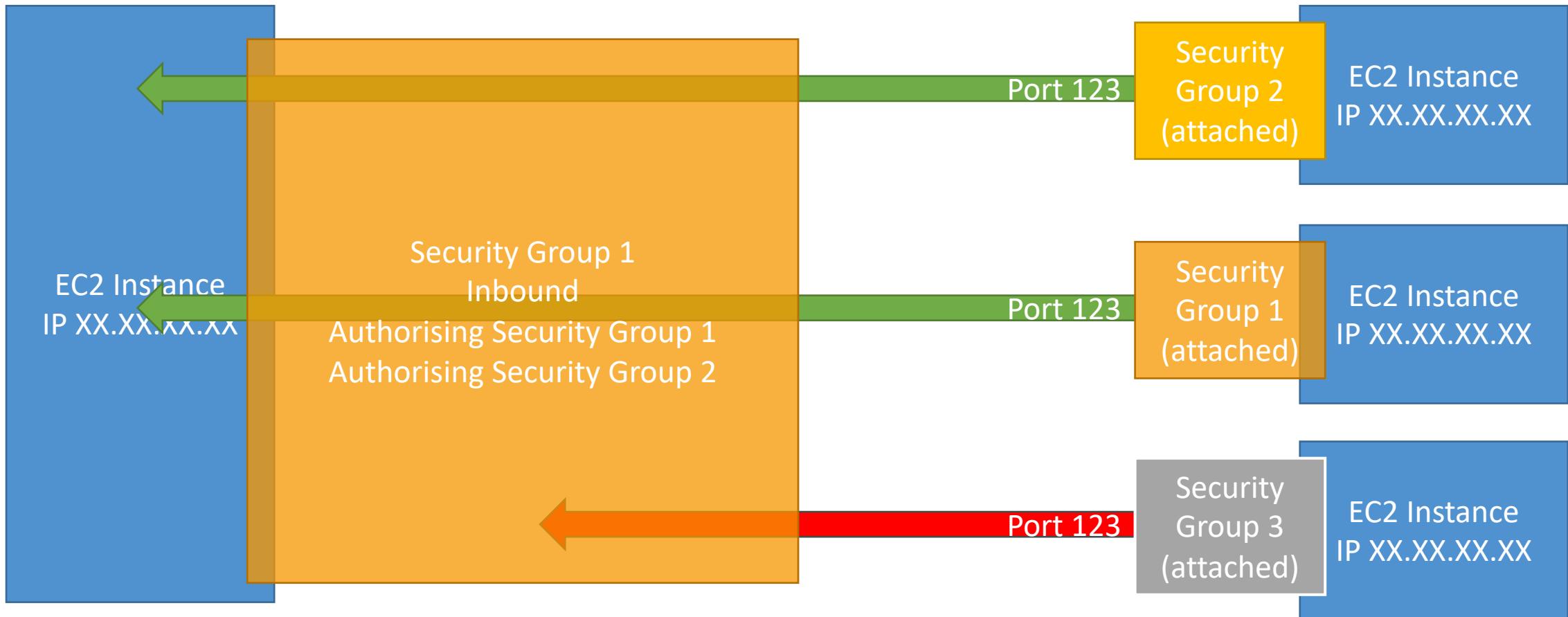
# Security Groups

## 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

# Referencing other security groups

## Diagram



# 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

# SSH Summary Table

	SSH	Putty	EC2 Instance Connect
Mac	✓		✓
Linux	✓		✓
Windows < 10		✓	✓
Windows >= 10	✓	✓	✓

# Which Lectures to watch

- Mac / Linux:
  - SSH on Mac/Linux lecture
- Windows:
  - Putty Lecture
  - If Windows 10: SSH on Windows 10 lecture
- All:
  - EC2 Instance Connect lecture

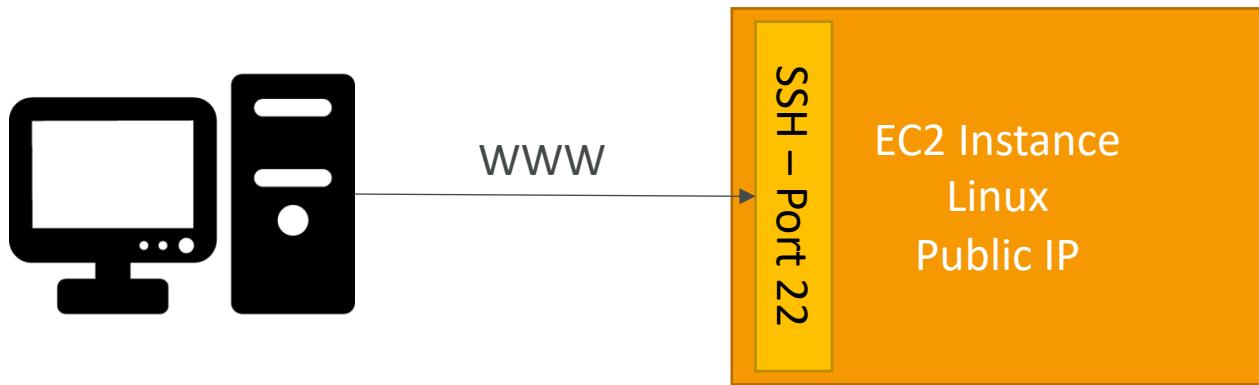
# SSH troubleshooting

- Students have the most problems with SSH
- If things don't work...
  1. Re-watch the lecture. You may have missed something
  2. Read the troubleshooting guide
  3. Try EC2 Instance Connect
- If one method works (SSH, Putty or EC2 Instance Connect) you're good
- If no method works, that's okay, the course won't use SSH much

# 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.

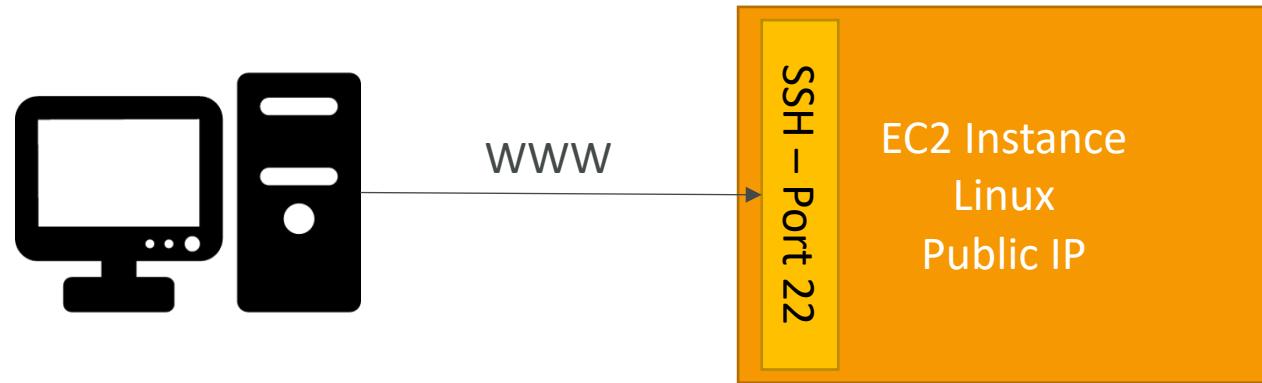


- We will see how we can configure OpenSSH `~/.ssh/config` to facilitate the SSH into our EC2 instances

# 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
- **Reserved:** (MINIMUM 1 year)
  - Reserved Instances: long workloads
  - Convertible Reserved Instances: long workloads with flexible instances
  - Scheduled Reserved Instances: example – every Thursday between 3 and 6 pm
- **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

# EC2 On Demand

- Pay for what you use:
  - Linux - billing per second, after the first minute
  - All other operating systems (ex:Windows) - 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
- Reservation period: 1 year = + discount | 3 years = +++ discount
- Purchasing options: no upfront | partial upfront = + | All upfront = ++ discount
- Reserve a specific instance type
- Recommended for steady-state usage applications (think database)
- **Convertible Reserved Instance**
  - can change the EC2 instance type
  - Up to 45% discount
- **Scheduled Reserved Instances**
  - launch within time window you reserve
  - When you require a fraction of day / week / month
  - Commitment for 1 year only



# 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

- An Amazon EC2 Dedicated Host is a physical server with EC2 instance capacity fully dedicated to your use. Dedicated Hosts can help you address **compliance requirements** and reduce costs by allowing you to **use your existing server-bound software licenses**.
- Allocated for your account for a 3-year period reservation
- More expensive
- 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 running 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

# 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.
- **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

# Price Comparison

## Example – m4.large – us-east-1

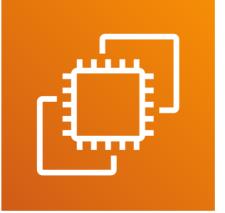
Price Type	Price (per hour)
On-demand	\$0.10
Spot Instance (Spot Price)	\$0.032 - \$0.045 (up to 90% off)
Spot Block (1 to 6 hours)	~ Spot Price
Reserved Instance (12 months) – no upfront	\$0.062
Reserved Instance (12 months) – all upfront	\$0.058
Reserved Instance (36 months) – no upfront	\$0.043
Reserved <b>Convertible</b> Instance (12 months) – no upfront	\$0.071
Reserved <b>Scheduled</b> Instance (recurring schedule on 12 months term)	\$0.090 – \$0.095 (5%-10% off)
Dedicated Host	On-demand price
Dedicated Host Reservation	Up to 70% off

# 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

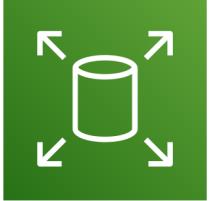
# EC2 Section – Summary



- **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?

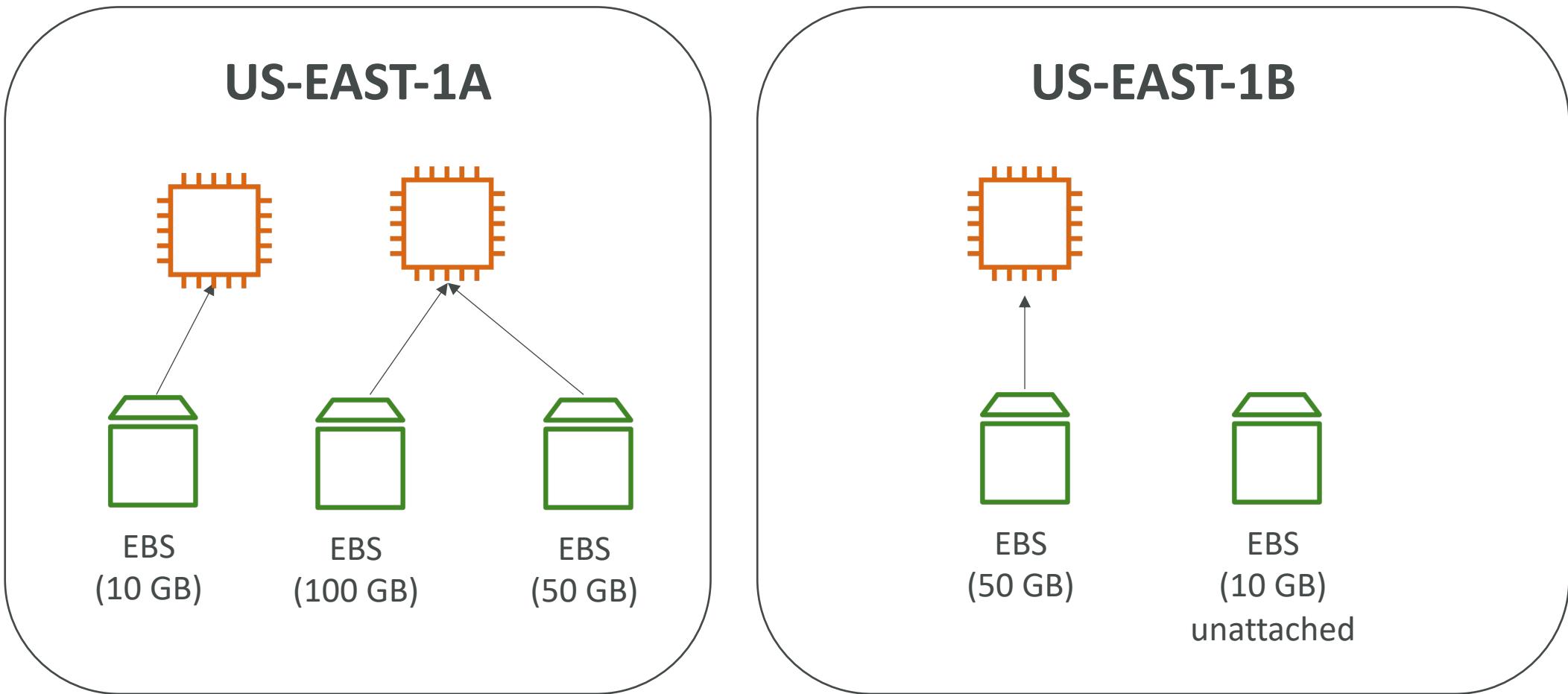


- An **EBS (Elastic Block Store) Volume** is a **network** drive you can attach to your instances while they run
- It allows your instances to persist data, even after their termination
- They can only be mounted to one instance at a time (at the CCP level)
- They are bound to a specific availability 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



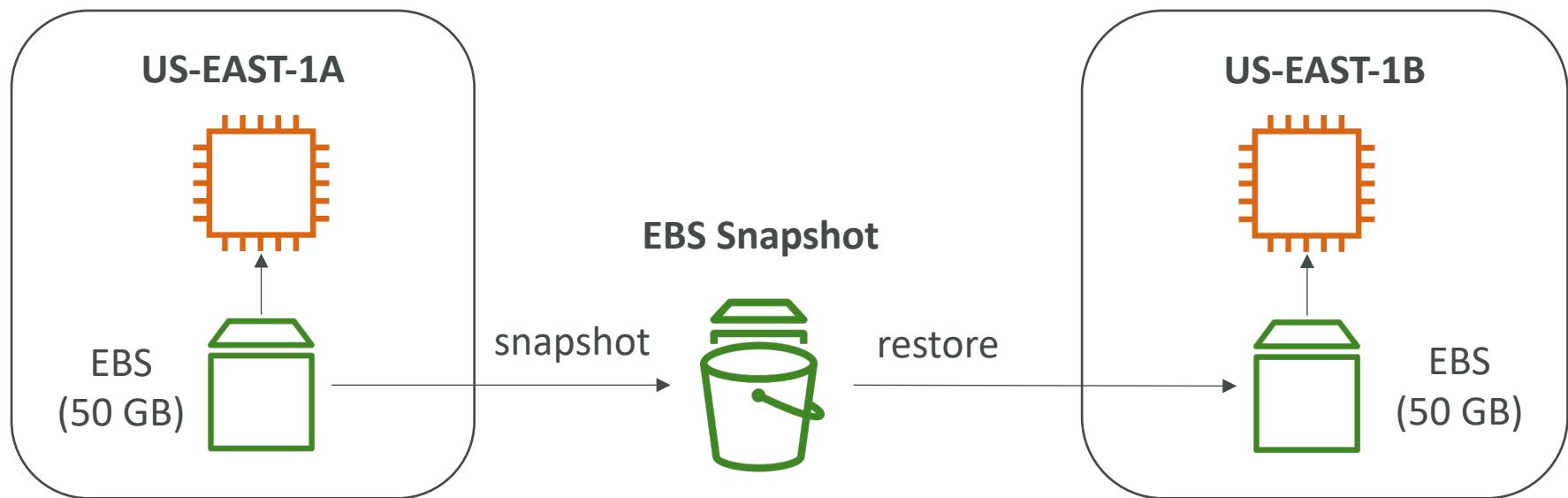
# EBS – Delete on Termination attribute

Volume Type <small>i</small>	Device <small>i</small>	Snapshot <small>i</small>	Size (GiB) <small>i</small>	Volume Type <small>i</small>	IOPS <small>i</small>	Throughput (MB/s) <small>i</small>	Delete on Termination <small>i</small>	Encryption <small>i</small>
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 ▾ <span style="color:red;">X</span>
<a href="#">Add New Volume</a>								

- 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



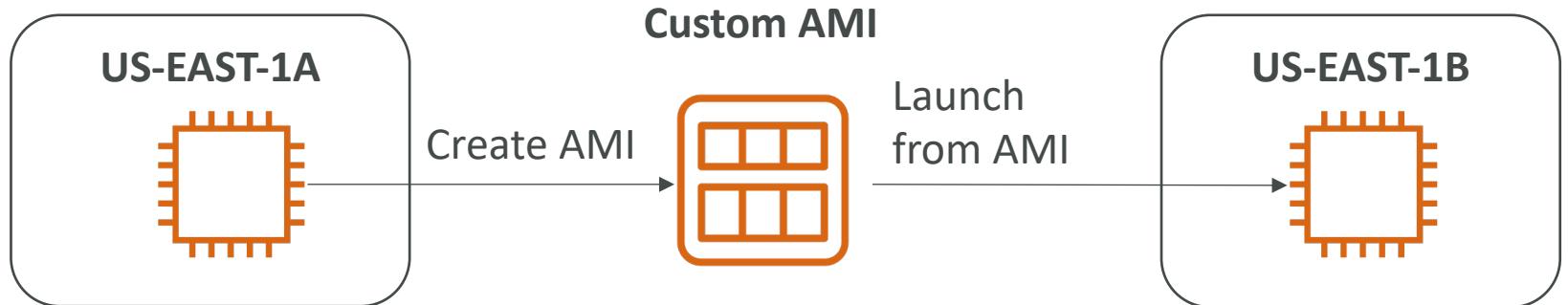


# AMI Overview

- 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)

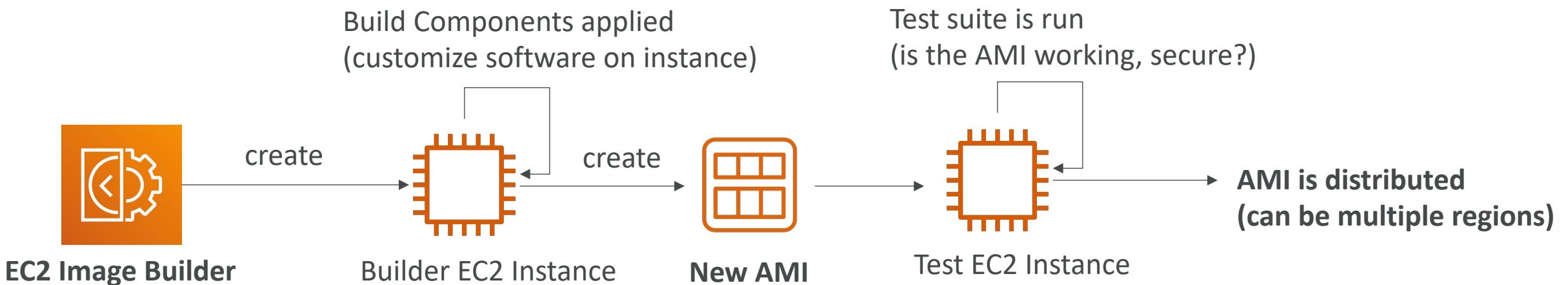
- Start an EC2 instance and customize it
- Stop the instance (for data integrity)
- Build an AMI – this will also create EBS snapshots
- Launch instances from other AMIs



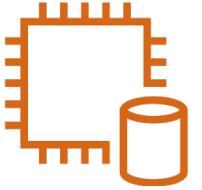
# EC2 Image Builder



- 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

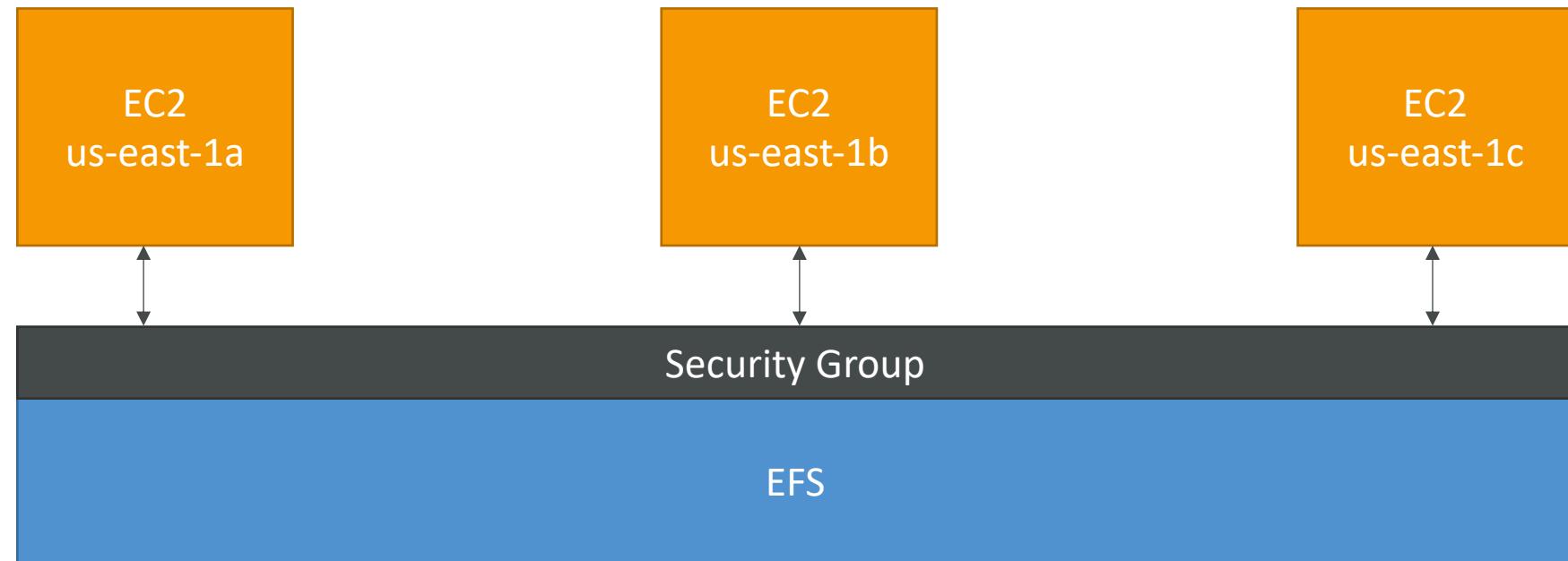
Very high IOPS

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

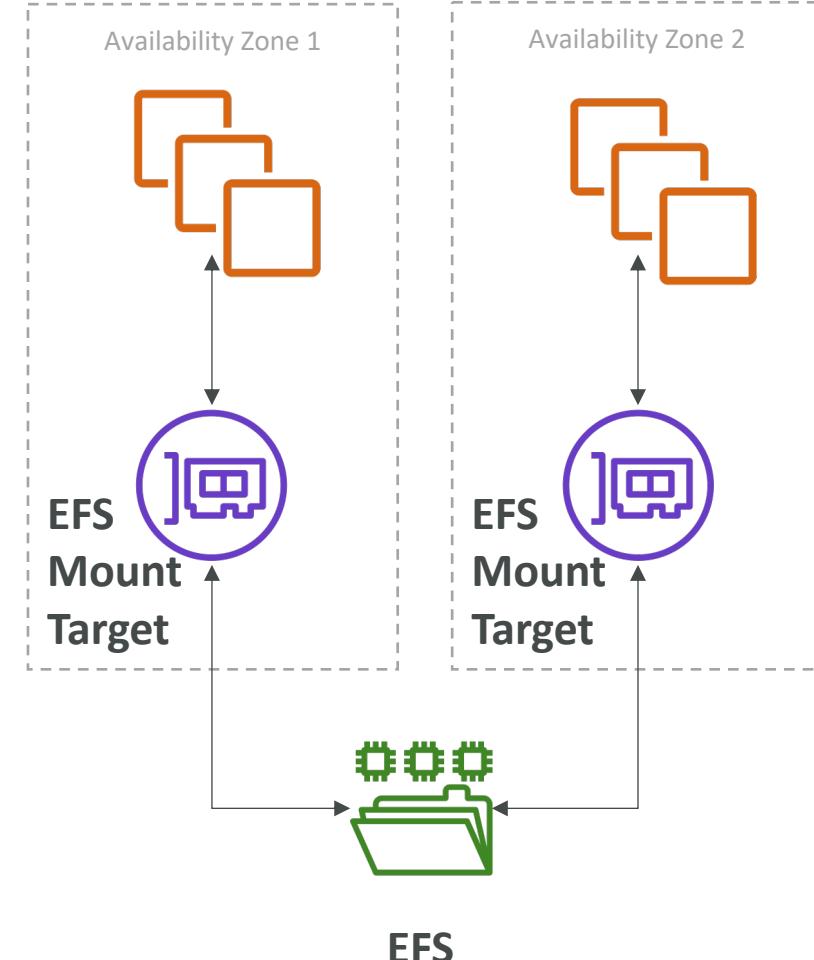
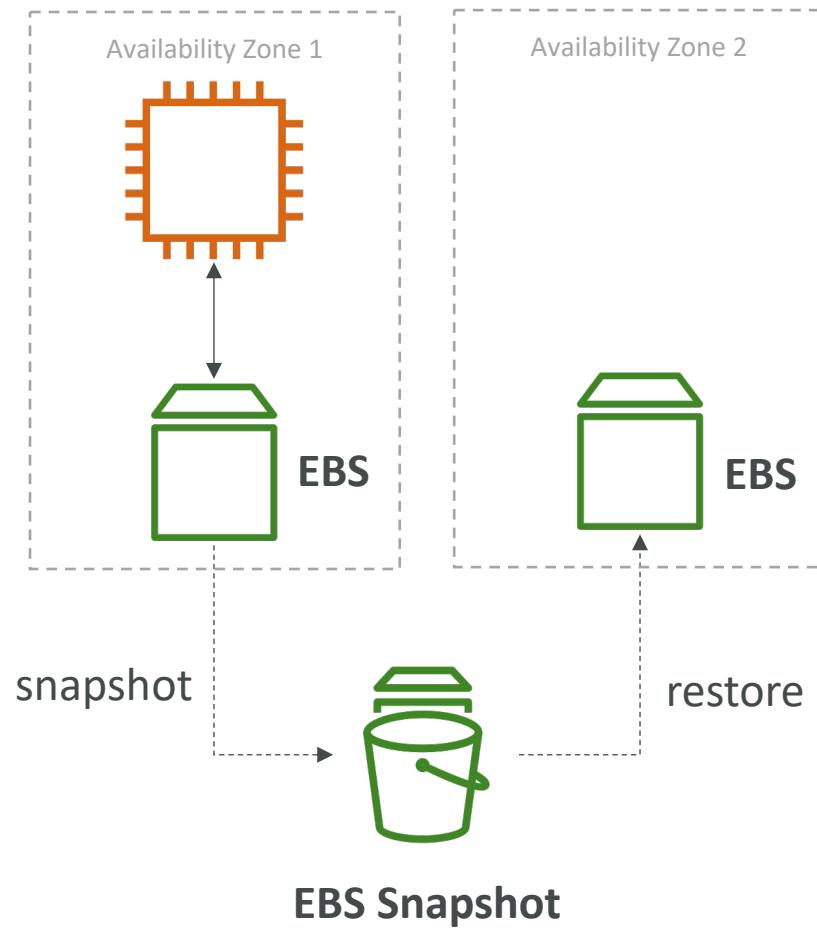
# EFS – Elastic File System



- 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

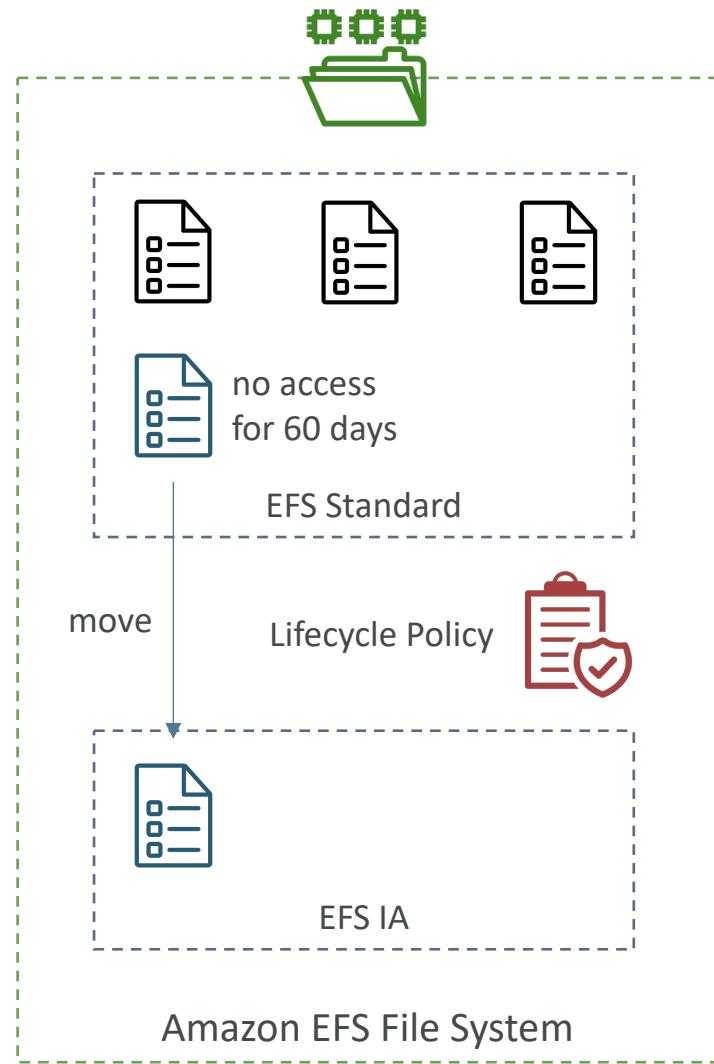


# EBS vs EFS



# 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

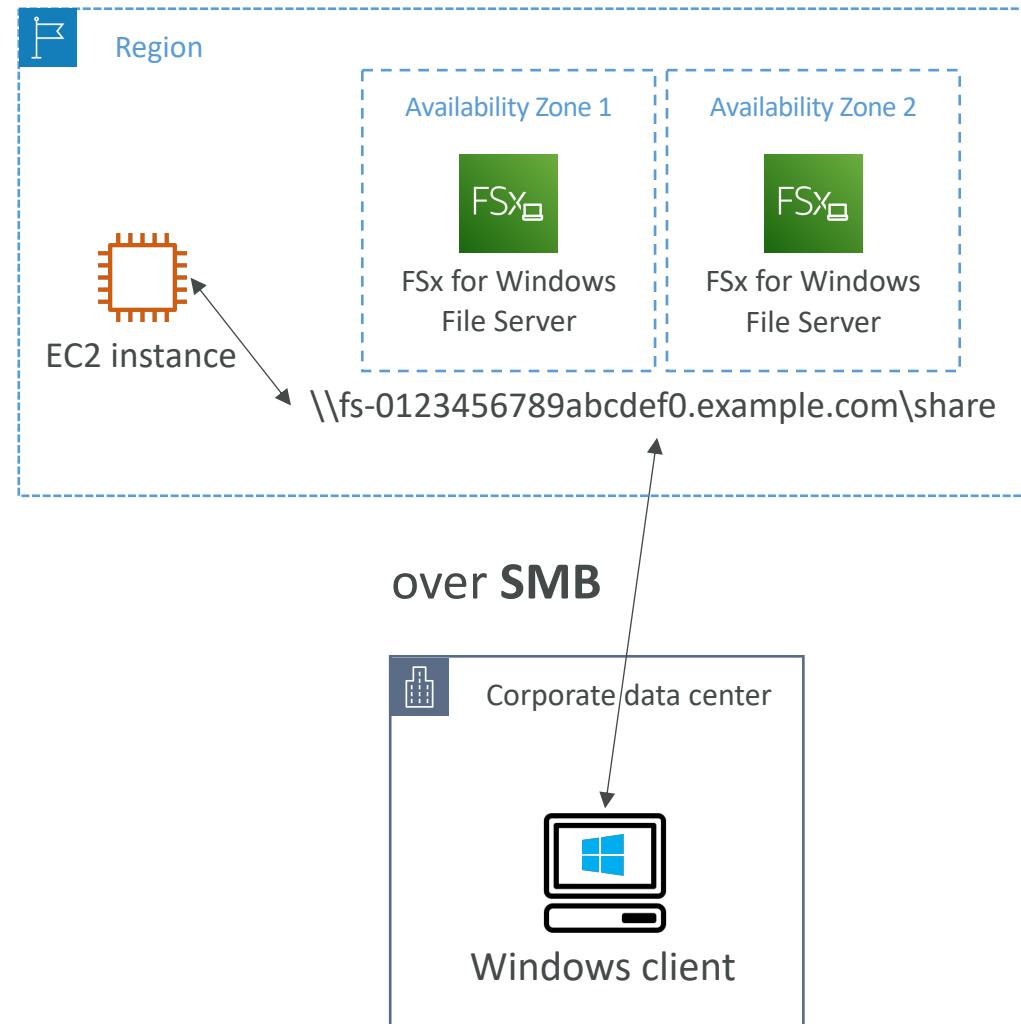


- Infrastructure
- Replication for data for EBS volumes & EFS drives
- Replacing faulty hardware
- Ensuring their employees cannot access your data
- Setting up backup / snapshot procedures
- Setting up data encryption
- Responsibility of any data on the drives
- Understanding the risk of using EC2 Instance Store

# Amazon FSx for Windows File Server



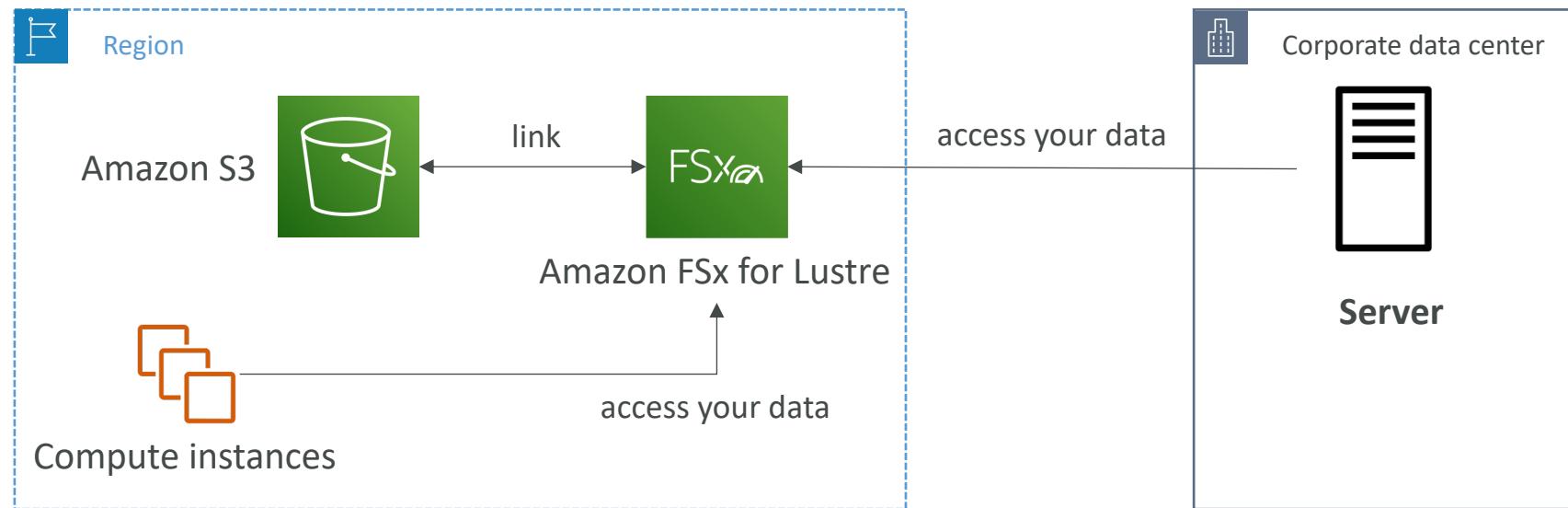
- A fully managed, highly reliable, and scalable **Windows native** shared file system
- Built on **Windows File Server**
- Supports **SMB protocol** & Windows NTFS
- Integrated with Microsoft Active Directory
- Can be accessed from AWS or your on-premise infrastructure



# Amazon FSx for Lustre



- A fully managed, high-performance, scalable file storage for **High Performance Computing (HPC)**
- The name Lustre is derived from “Linux” and “cluster”
- Machine Learning, Analytics, Video Processing, Financial Modeling, ...
- Scales up to 100s GB/s, millions of IOPS, sub-ms latencies



# 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

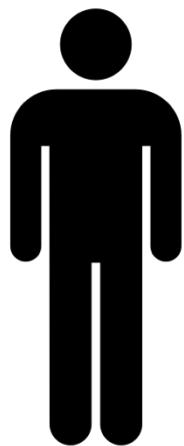
# Elastic Load Balancing & Auto Scaling Groups Section

# Scalability & High Availability

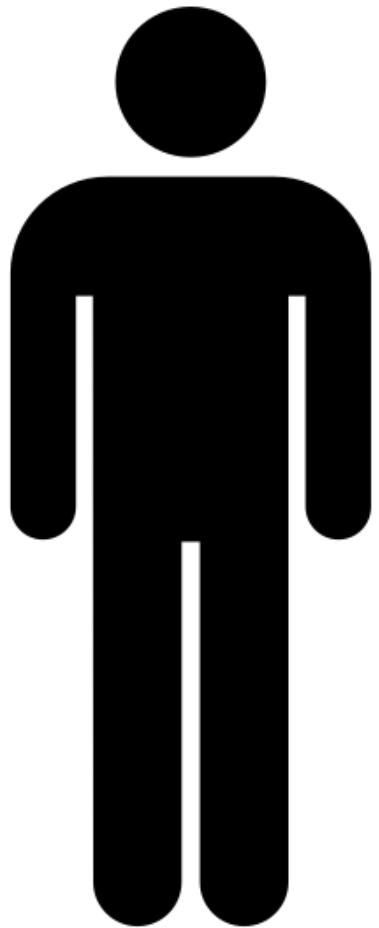
- Scalability means that an application / system can handle greater loads by adapting.
- There are two kinds of scalability:
  - Vertical Scalability
  - Horizontal Scalability (= elasticity)
- Scalability is linked but different to High Availability
- Let's deep dive into the distinction, using a call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

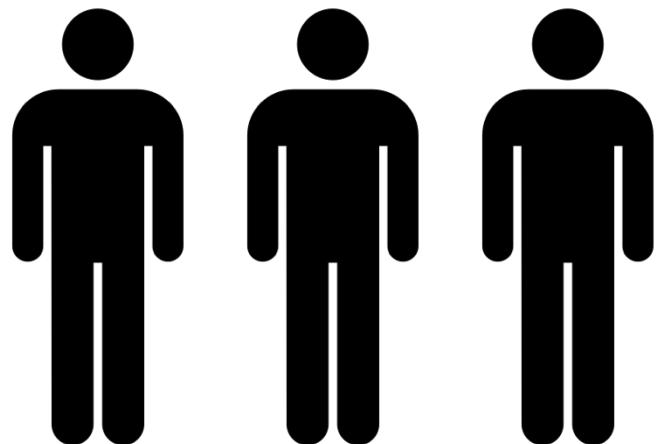
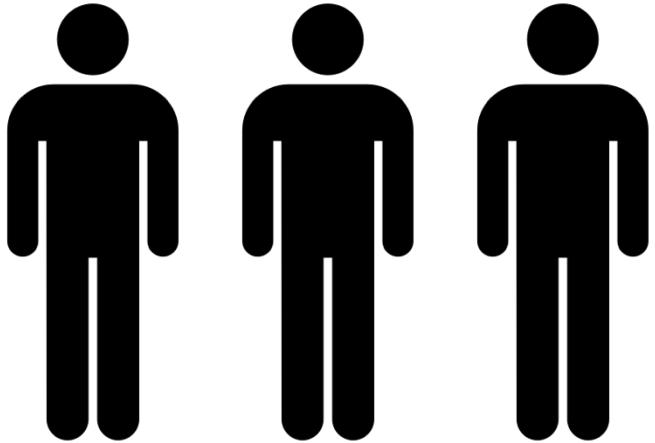


senior operator

# 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

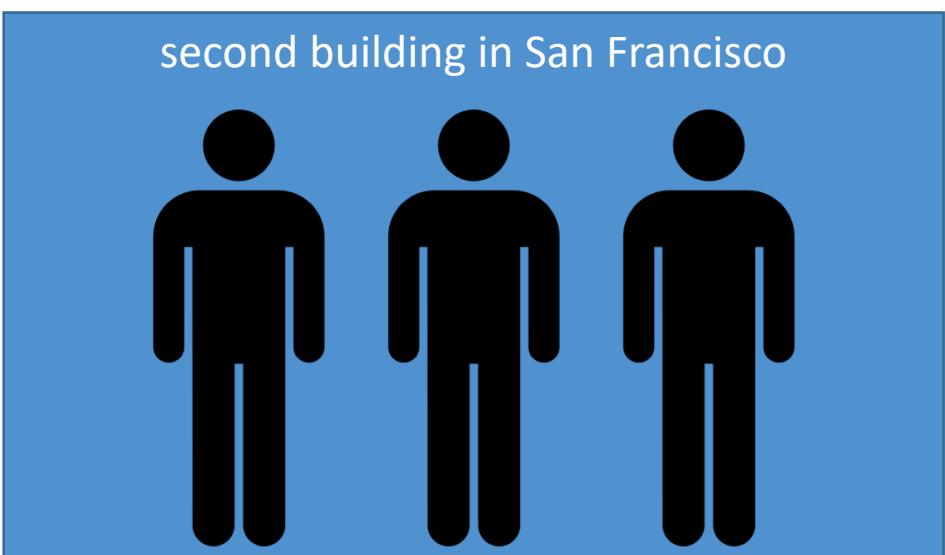
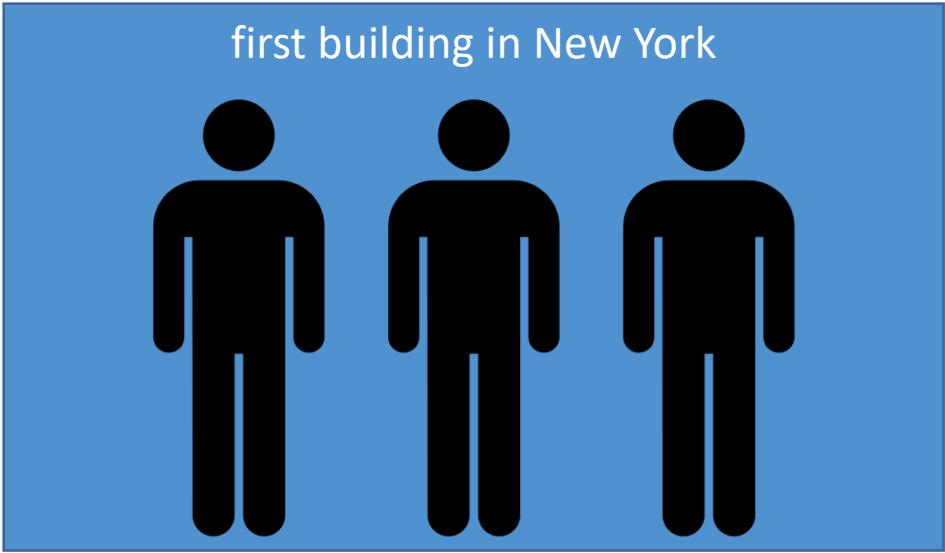
operator operator operator



operator operator operator

# 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-12tbl.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

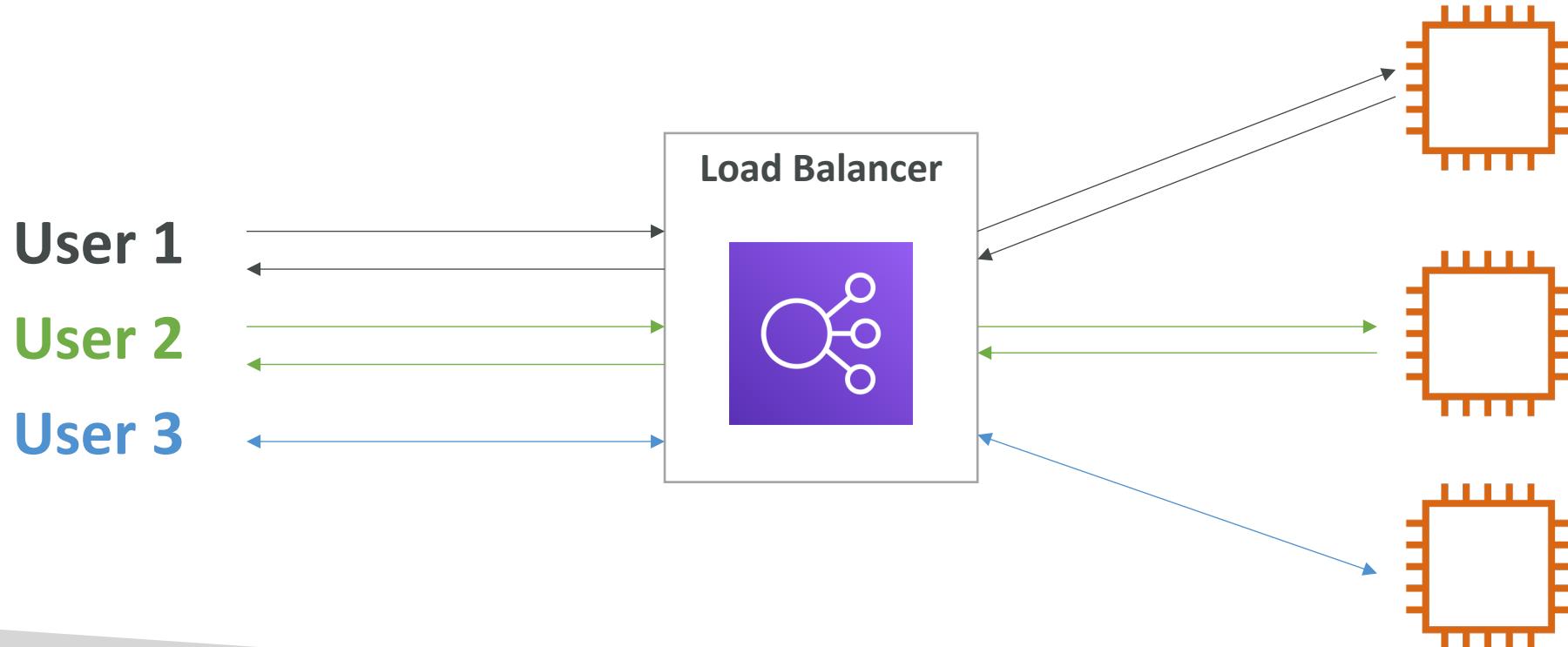
# Scalability vs Elasticity (vs Agility)

- **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.



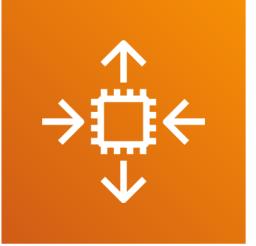
# Why use a load balancer?

- 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?

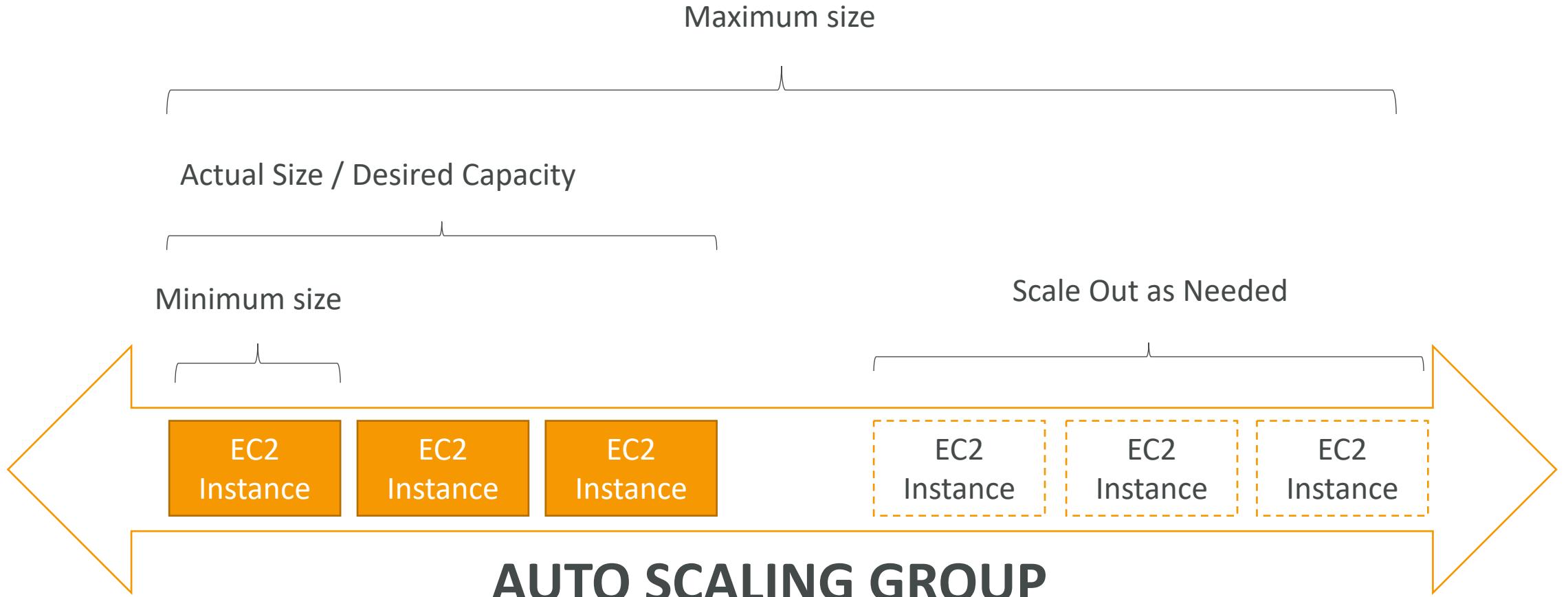
- An ELB (Elastic Load Balancer) is a **managed load balancer**
  - AWS guarantees that it will be working
  - AWS takes care of upgrades, maintenance, high availability
  - AWS provides only a few configuration knobs
- It costs less to setup your own load balancer but it will be a lot more effort on 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?

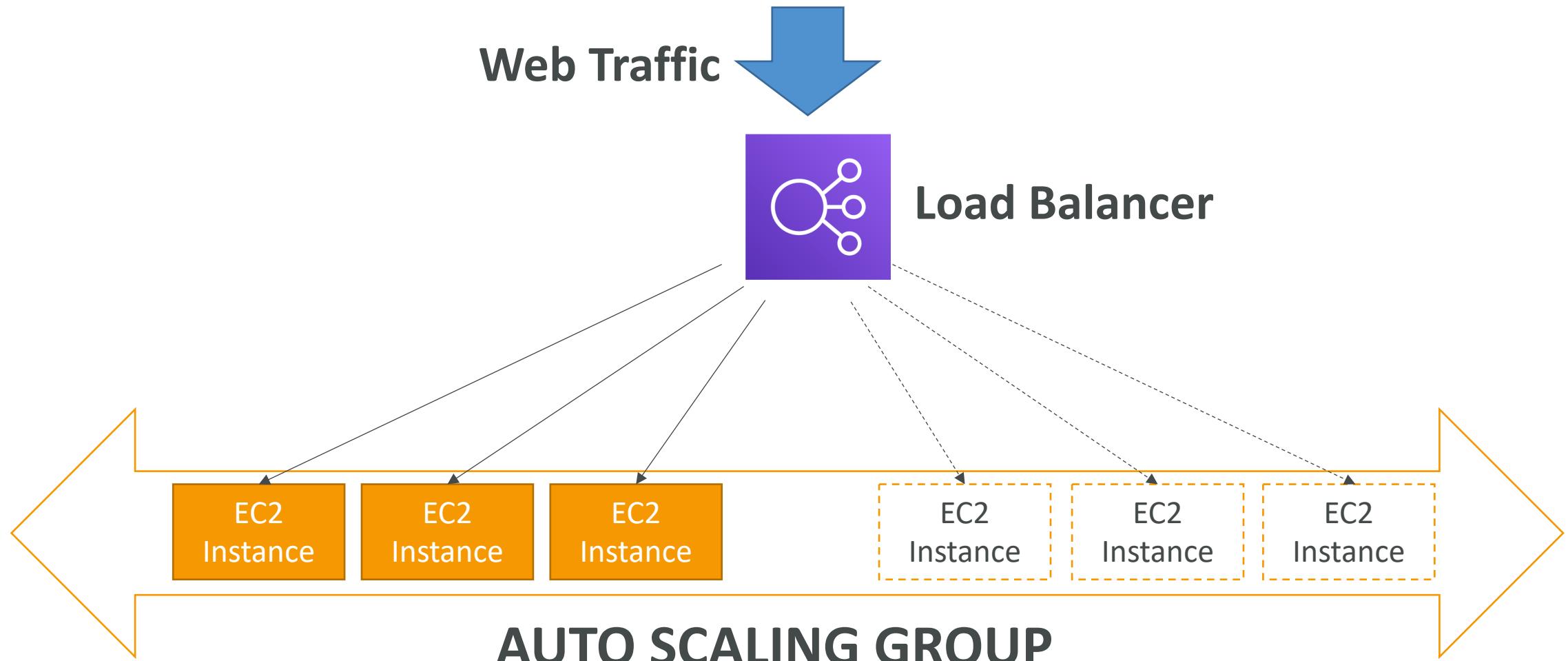


- In real-life, the load on your websites and application can change
- In the cloud, you can create and get rid of servers very quickly
- The goal of an Auto Scaling Group (ASG) is to:
  - Scale out (add EC2 instances) to match an increased load
  - Scale in (remove EC2 instances) to match a decreased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

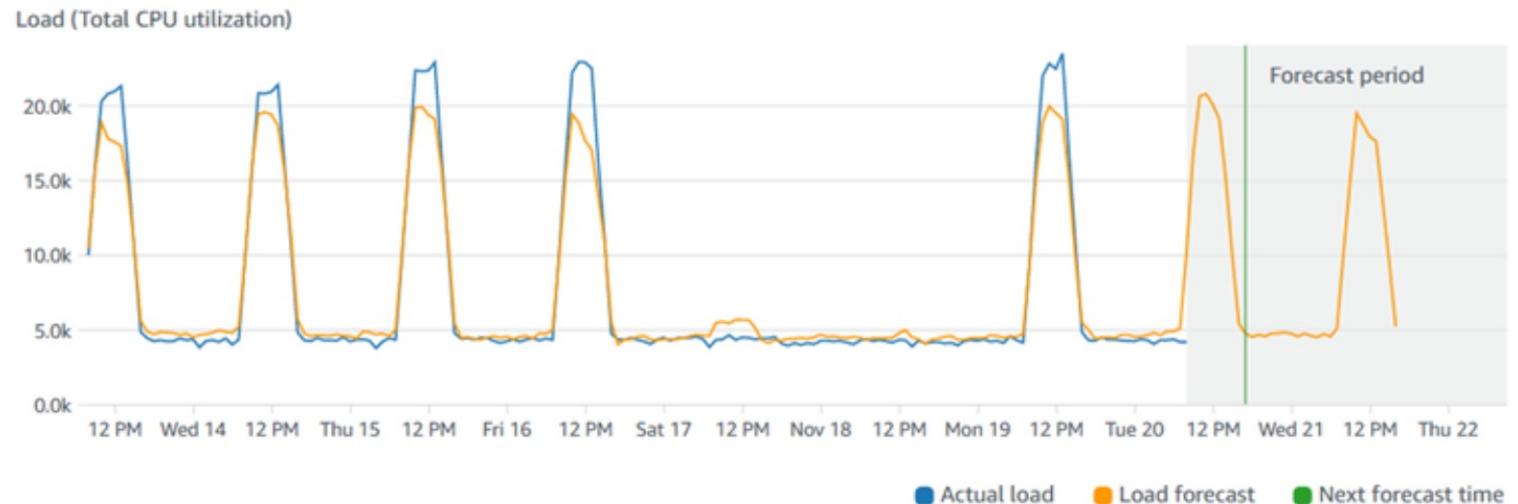
- **Manual Scaling:** Update the size of an ASG manually
- **Dynamic Scaling:** Respond to changing demand
  - **Simple / Step Scaling**
    - When a CloudWatch alarm is triggered (example CPU > 70%), then add 2 units
    - When a CloudWatch alarm is triggered (example CPU < 30%), then remove 1
  - **Target 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

# Amazon S3 Section

# Section introduction



- Amazon S3 is one of the main building blocks of AWS
- It's advertised as "infinitely scaling" storage
- Many websites use Amazon S3 as a backbone
- Many AWS services use Amazon S3 as an integration as well
- We'll have a step-by-step approach to S3
- The CCP exam requires "deeper" knowledge about S3

# S3 Use cases

- Backup and storage
- Disaster Recovery
- Archive
- Hybrid Cloud storage
- Application hosting
- Media hosting
- Data lakes & big data analytics
- Software delivery
- Static website



Nasdaq stores 7 years of data into S3 Glacier



Sysco runs analytics on its data and gain business insights

# Amazon S3 Overview - Buckets

- Amazon S3 allows people to store objects (files) in “buckets” (directories)
- Buckets must have a **globally unique name** (across all regions all accounts)
- Buckets are defined at the region level
- S3 looks like a global service but buckets are created in a region
- Naming convention
  - No uppercase
  - No underscore
  - 3-63 characters long
  - Not an IP
  - Must start with lowercase letter or number



# Amazon S3 Overview - Objects

- Objects (files) have a Key
- The **key** is the **FULL** path:
  - s3://my-bucket/**my\_file.txt**
  - s3://my-bucket/**my\_folder1/another\_folder/my\_file.txt**
- The key is composed of **prefix** + **object name**
  - s3://my-bucket/**my\_folder1/another\_folder**/**my\_file.txt**
- There's no concept of "directories" within buckets (although the UI will trick you to think otherwise)
- Just keys with very long names that contain slashes ("/")



# Amazon S3 Overview – Objects (continued)

- Object values are the content of the body:
  - Max Object Size is 5TB (5000GB)
  - If uploading more than 5GB, must use “multi-part upload”
- Metadata (list of text key / value pairs – system or user metadata)
- Tags (Unicode key / value pair – up to 10) – useful for security / lifecycle
- Version ID (if versioning is enabled)



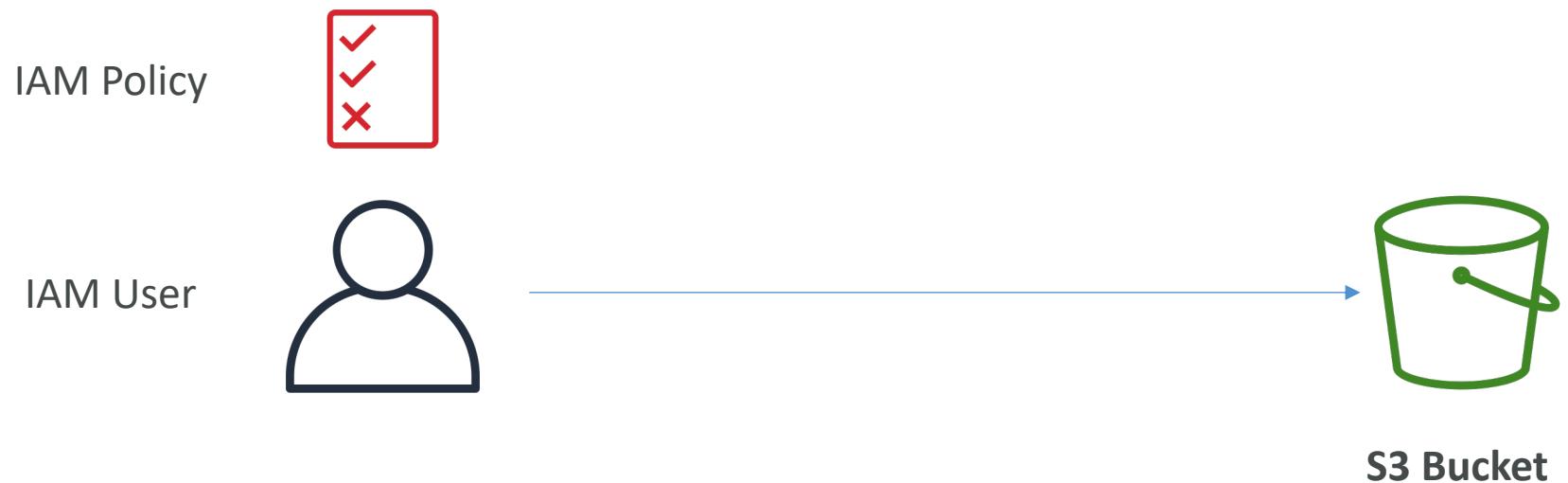
# S3 Security

- **User based**
  - IAM policies - which API calls should be allowed for a specific user from IAM console
- **Resource Based**
  - Bucket Policies - bucket wide rules from the S3 console - allows cross account
  - Object Access Control List (ACL) – finer grain
  - Bucket Access Control List (ACL) – less common
- **Note:** an IAM principal can access an S3 object if
  - the user IAM permissions allow it OR the resource policy ALLOWS it
  - AND there's no explicit DENY
- **Encryption:** encrypt objects in Amazon S3 using encryption keys

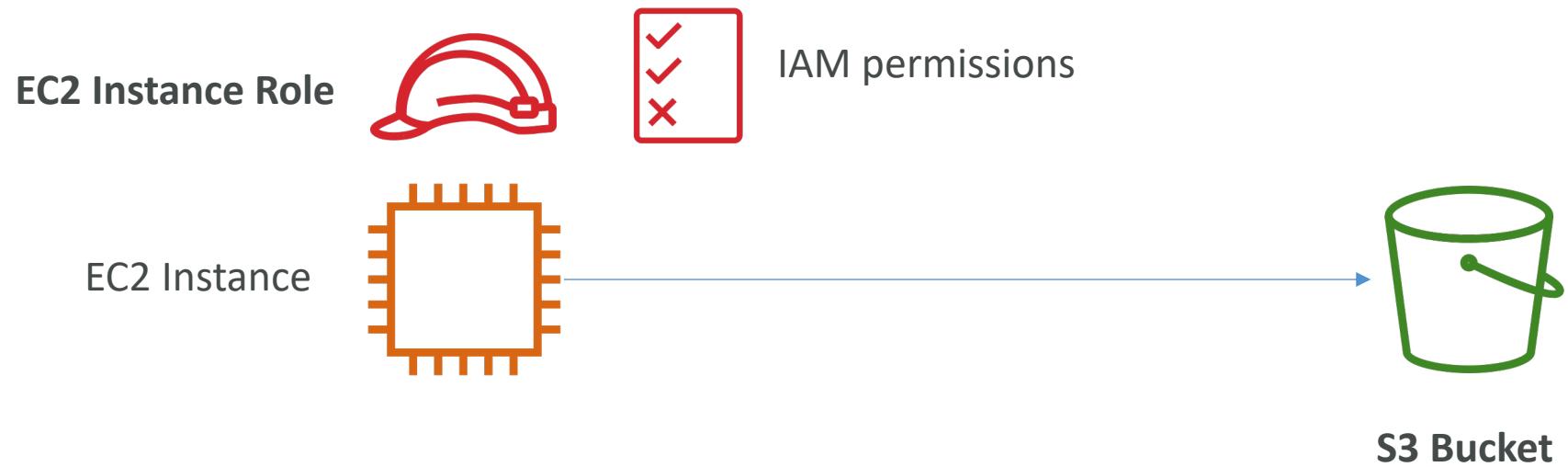
# Example: Public Access - Use Bucket Policy



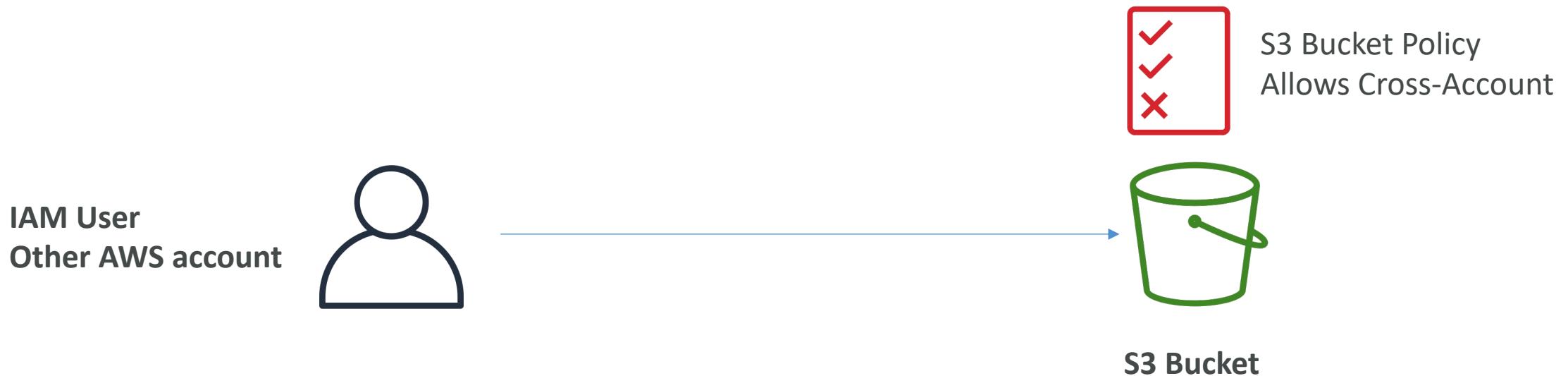
# Example: User Access to S3 – IAM permissions



# Example: EC2 instance access - Use IAM Roles



# Advanced: Cross-Account Access – Use Bucket Policy

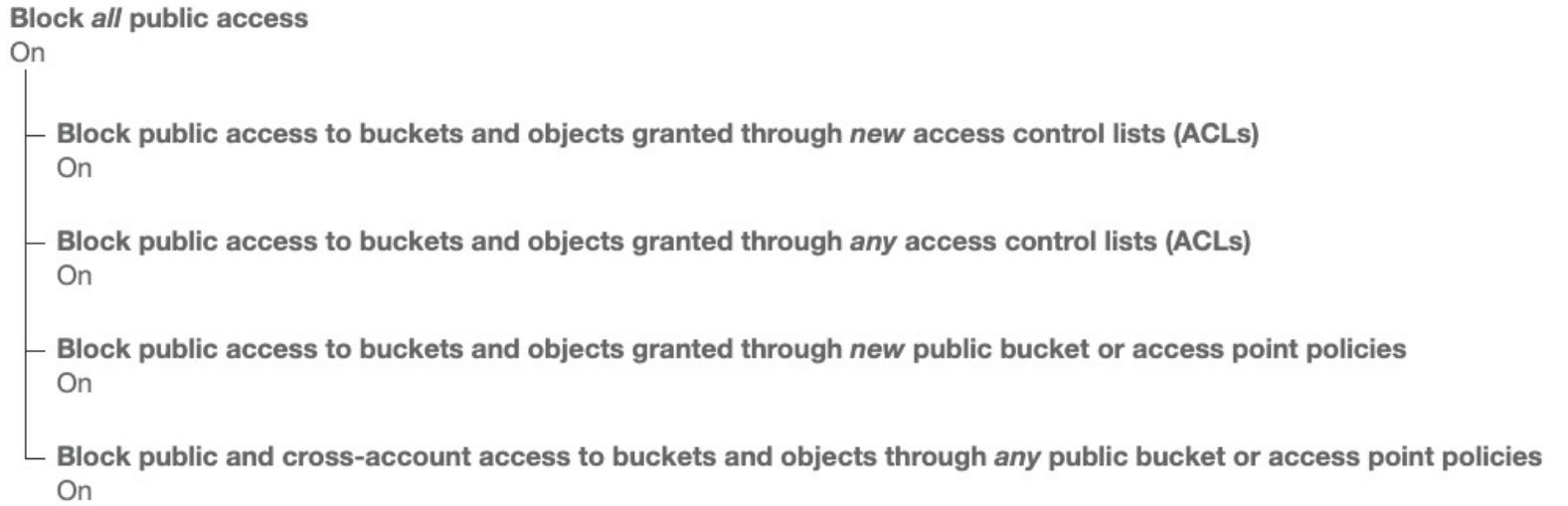


# S3 Bucket Policies

- JSON based policies
  - Resources: buckets and objects
  - Actions: Set of API to Allow or Deny
  - Effect: Allow / Deny
  - Principal: The account or user to apply the policy to
- Use S3 bucket for policy to:
  - Grant public access to the bucket
  - Force objects to be encrypted at upload
  - Grant access to another account (Cross Account)

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Sid": "PublicRead",  
      "Effect": "Allow",  
      "Principal": "*",  
      "Action": [  
        "s3:GetObject"  
      ],  
      "Resource": [  
        "arn:aws:s3:::examplebucket/*"  
      ]  
    }  
  ]  
}
```

# Bucket settings for Block Public Access



- These settings were created to prevent company data leaks
- If you know your bucket should never be public, leave these on
- Can be set at the account level

# S3 Websites

- S3 can host static websites and have them accessible on the www
- The website URL will be:
  - <bucket-name>.s3-website-<AWS-region>.amazonaws.com  
OR
  - <bucket-name>.s3-website.<AWS-region>.amazonaws.com
- If you get a 403 (Forbidden) error, make sure the bucket policy allows public reads!

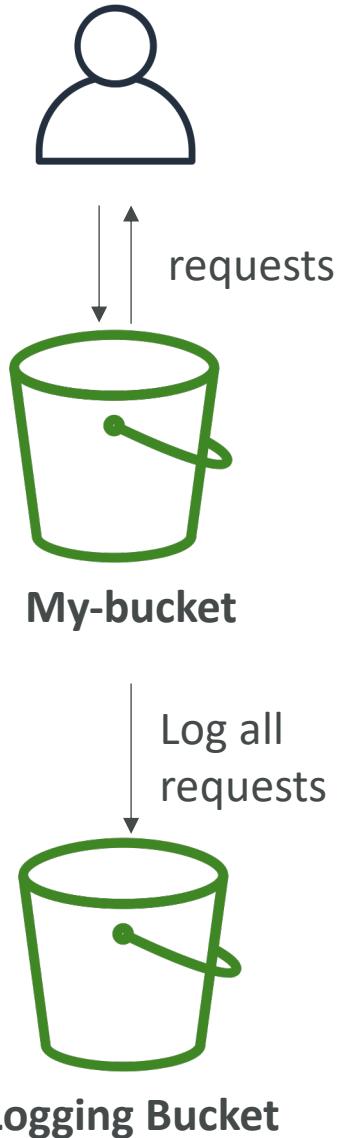
# Amazon S3 - Versioning



- You can version your files in Amazon S3
- It is enabled at the **bucket level**
- Same key overwrite will increment the “version”: 1, 2, 3....
- It is best practice to version your buckets
  - Protect against unintended deletes (ability to restore a version)
  - Easy roll back to previous version
- Notes:
  - Any file that is not versioned prior to enabling versioning will have version “null”
  - Suspending versioning does not delete the previous versions

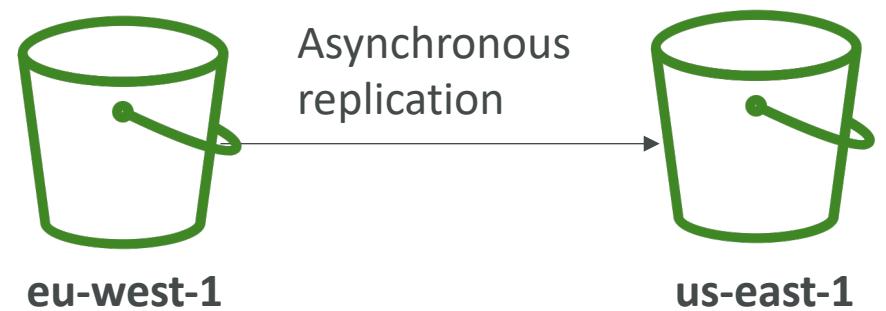
# S3 Access Logs

- For audit purpose, you may want to log all access to S3 buckets
- Any request made to S3, from any account, authorized or denied, will be logged into another S3 bucket
- That data can be analyzed using data analysis tools...
- Very helpful to come down to the root cause of an issue, or audit usage, view suspicious patterns, etc...



# S3 Replication (CRR & SRR)

- Must enable **versioning** in source and destination
  - Cross Region Replication (CRR)
  - Same Region Replication (SRR)
  - Buckets can be in different accounts
  - Copying is asynchronous
  - Must give proper IAM permissions to S3
- 
- CRR - Use cases: compliance, lower latency access, replication across accounts
  - SRR – Use cases: log aggregation, live replication between production and test accounts



# S3 Storage Classes

- Amazon S3 Standard - General Purpose
- Amazon S3 Standard-Infrequent Access (IA)
- Amazon S3 One Zone-Infrequent Access
- Amazon S3 Intelligent Tiering
- Amazon Glacier
- Amazon Glacier Deep Archive
  
- Amazon S3 Reduced Redundancy Storage (deprecated - omitted)

# S3 Durability and Availability

- Durability:
  - High durability (99.99999999%, 11 9's) of objects across multiple AZ
  - If you store 10,000,000 objects with Amazon S3, you can on average expect to incur a loss of a single object once every 10,000 years
  - Same for all storage classes
- Availability:
  - Measures how readily available a service is
  - S3 standard has 99.99% availability, which means it will not be available 53 minutes a year
  - Varies depending on storage class

# S3 Standard – General Purposes

- 99.99% Availability
  - Used for frequently accessed data
  - Low latency and high throughput
  - Sustain 2 concurrent facility failures
- 
- Use Cases: Big Data analytics, mobile & gaming applications, content distribution...

# S3 Standard – Infrequent Access (IA)

- Suitable for data that is less frequently accessed, but requires rapid access when needed
  - 99.9% Availability
  - Lower cost compared to Amazon S3 Standard, but retrieval fee
  - Sustain 2 concurrent facility failures
- 
- Use Cases: As a data store for disaster recovery, backups...

# S3 Intelligent-Tiering

- 99.9% Availability
- Same low latency and high throughput performance of S3 Standard
- **Cost-optimized by** automatically moving objects between two access tiers based on changing access patterns:
  - Frequent access
  - Infrequent access
- Resilient against events that impact an entire Availability Zone

# S3 One Zone - Infrequent Access (IA)

- Same as IA but data is stored in a single AZ
- 99.5% Availability
- Low latency and high throughput performance
- Lower cost compared to S3-IA (by 20%)
  
- Use Cases: Storing secondary backup copies of on-premise data, or storing data you can recreate

# Amazon Glacier & Glacier Deep Archive



- Low cost object storage (in GB/month) meant for archiving / backup
- Data is retained for the longer term (years)
- Various retrieval options of time + fees for retrieval:
- **Amazon Glacier** – cheap:
  - Expedited (1 to 5 minutes)
  - Standard (3 to 5 hours)
  - Bulk (5 to 12 hours)
- **Amazon Glacier Deep Archive** – cheapest:
  - Standard (12 hours)
  - Bulk (48 hours)

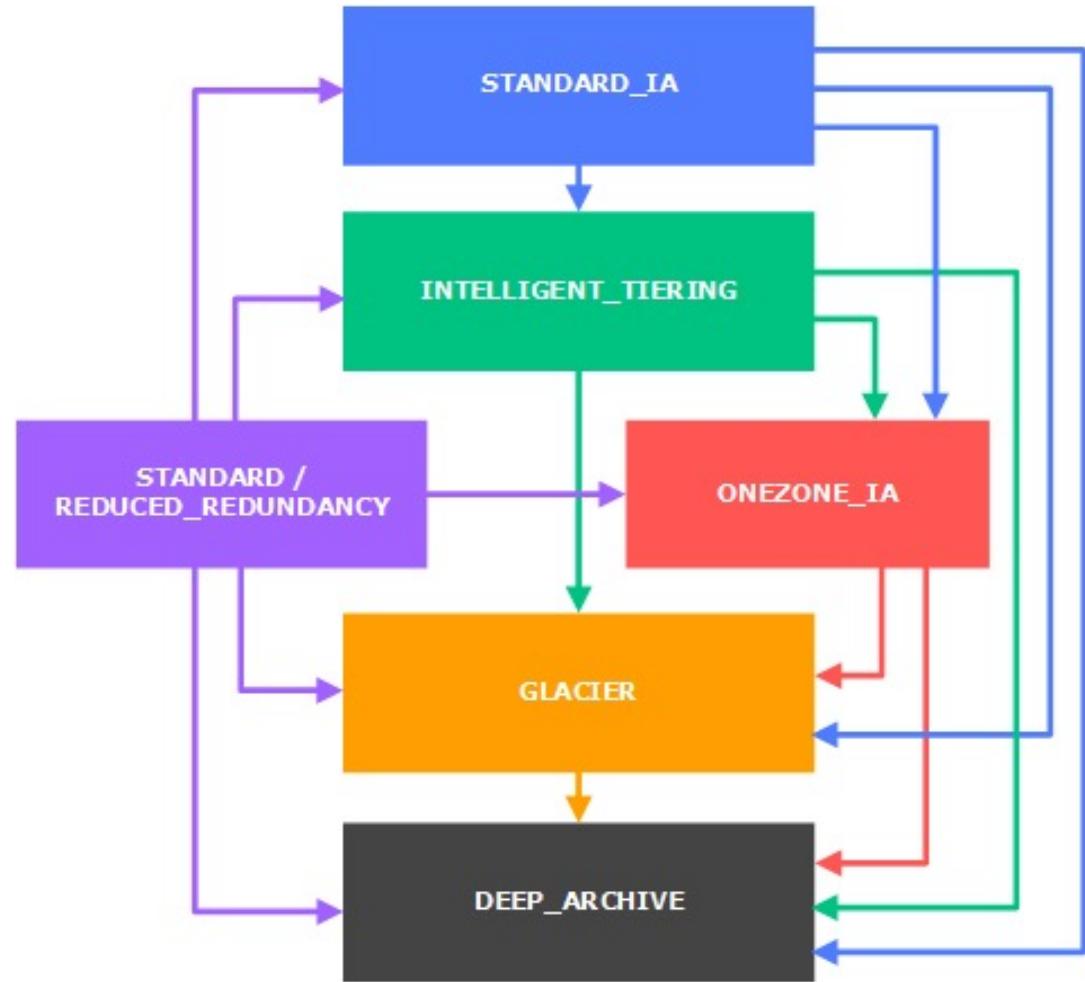
# S3 Storage Classes Comparison

	S3 Standard	S3 Intelligent-Tiering	S3 Standard-IA	S3 One Zone-IA	S3 Glacier	S3 Glacier Deep Archive
<b>Designed for durability</b>	99.999999999% (11 9's)					
<b>Designed for availability</b>	99.99%	99.9%	99.9%	99.5%	99.99%	99.99%
<b>Availability SLA</b>	99.9%	99%	99%	99%	99.9%	99.9%
<b>Availability Zones</b>	≥3	≥3	≥3	1	≥3	≥3
<b>Minimum capacity charge per object</b>	N/A	N/A	128KB	128KB	40KB	40KB
<b>Minimum storage duration charge</b>	N/A	30 days	30 days	30 days	90 days	180 days
<b>Retrieval fee</b>	N/A	N/A	per GB retrieved	per GB retrieved	per GB retrieved	per GB retrieved

<https://aws.amazon.com/s3/storage-classes/>

# S3 – Moving between storage classes

- You can transition objects between storage classes
- For infrequently accessed object, move them to STANDARD\_IA
- For archive objects you don't need in real-time, GLACIER or DEEP\_ARCHIVE
- Moving objects can be automated using a **lifecycle configuration**



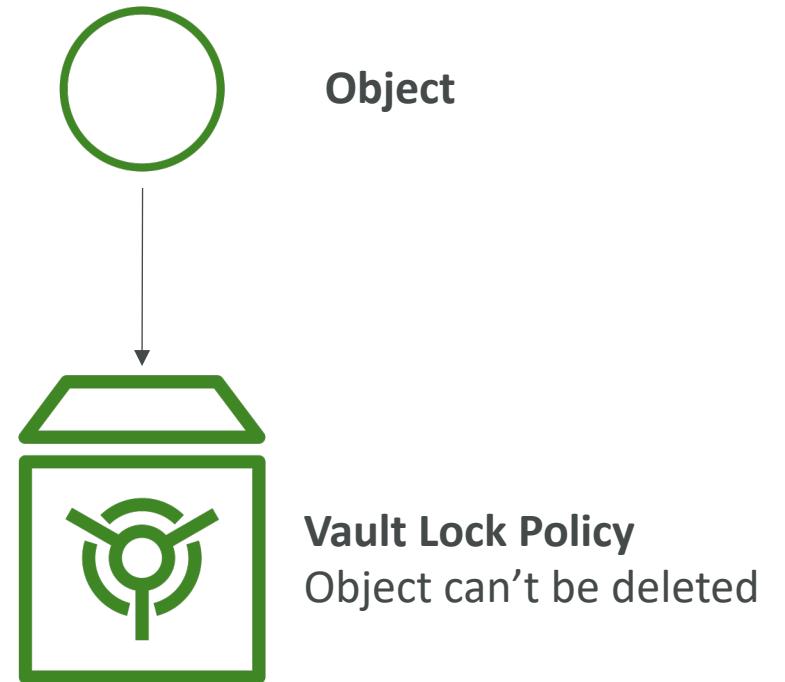
# S3 Object Lock & Glacier Vault Lock

- **S3 Object Lock**

- Adopt a WORM (Write Once Read Many) model
- Block an object version deletion for a specified amount of time

- **Glacier Vault Lock**

- Adopt a WORM (Write Once Read Many) model
- Lock the policy for future edits (can no longer be changed)
- Helpful for compliance and data retention



# Shared Responsibility Model for S3



- Infrastructure (global security, durability, availability, sustain concurrent loss of data in two facilities)
- Configuration and vulnerability analysis
- Compliance validation
- S3 Versioning
- S3 Bucket Policies
- S3 Replication Setup
- Logging and Monitoring
- S3 Storage Classes
- Data encryption at rest and in transit

# AWS Snow Family

- Highly-secure, portable devices to collect and process data at the edge, and migrate data into and out of AWS

- Data migration:



Snowcone



Snowball Edge



Snowmobile

- Edge computing:



Snowcone



Snowball Edge

# Data Migrations with AWS Snow Family

	Time to Transfer		
	100 Mbps	1Gbps	10Gbps
10 TB	12 days	30 hours	3 hours
100 TB	124 days	12 days	30 hours
1 PB	3 years	124 days	12 days

## Challenges:

- Limited connectivity
- Limited bandwidth
- High network cost
- Shared bandwidth (can't maximize the line)
- Connection stability

AWS Snow Family: offline devices to perform data migrations

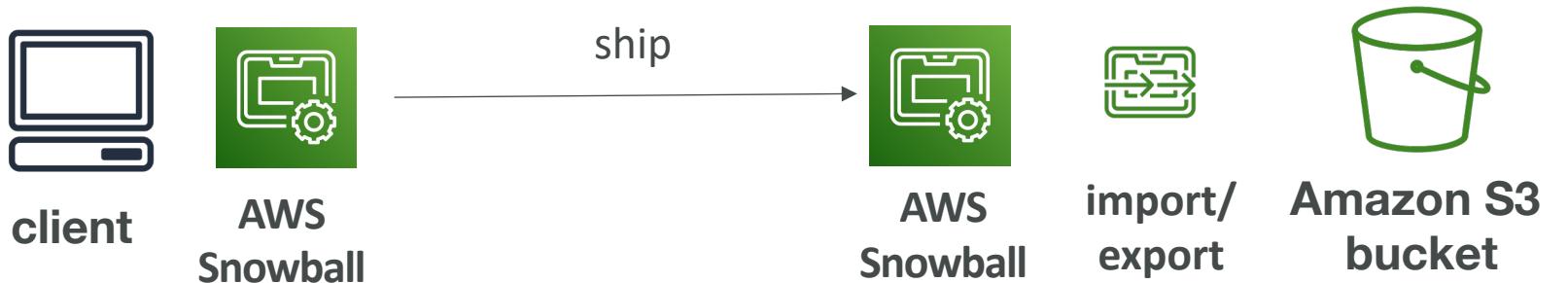
If it takes more than a week to transfer over the network, use Snowball devices!

# Diagrams

- Direct upload to S3:



- With Snow Family:



# Snowball Edge (for data transfers)



- Physical data transport solution: move TBs or PBs of data in or out of AWS
- Alternative to moving data over the network (and paying network fees)
- Pay per data transfer job
- Provide block storage and Amazon S3-compatible object storage
- **Snowball Edge Storage Optimized**
  - 80 TB of HDD capacity for block volume and S3 compatible object storage
- **Snowball Edge Compute Optimized**
  - 42 TB of HDD capacity for block volume and S3 compatible object storage
- Use cases: large data cloud migrations, DC decommission, disaster recovery



# AWS Snowcone



- Small, portable computing, anywhere, rugged & secure, withstands harsh environments
- Light (4.5 pounds, 2.1 kg)
- Device used for edge computing, storage, and data transfer
- **8 TBs of usable storage**
- Use Snowcone where Snowball does not fit (space-constrained environment)
- Must provide your own battery / cables
- Can be sent back to AWS offline, or connect it to internet and use **AWS DataSync** to send data



# AWS Snowmobile



- Transfer exabytes of data (1 EB = 1,000 PB = 1,000,000 TBs)
- Each Snowmobile has 100 PB of capacity (use multiple in parallel)
- High security: temperature controlled, GPS, 24/7 video surveillance
- Better than Snowball if you transfer more than 10 PB

# AWS Snow Family for Data Migrations



**Snowcone**



**Snowball Edge**



**Snowmobile**

	<b>Snowcone</b>	<b>Snowball Edge Storage Optimized</b>	<b>Snowmobile</b>
Storage Capacity	8 TB usable	80 TB usable	< 100 PB
Migration Size	Up to 24 TB, online and offline	Up to petabytes, offline	Up to exabytes, offline
DataSync agent	Pre-installed		
Storage Clustering		Up to 15 nodes	

# Snow Family – Usage Process

1. Request Snowball devices from the AWS console for delivery
2. Install the snowball client / AWS OpsHub on your servers
3. Connect the snowball to your servers and copy files using the client
4. Ship back the device when you're done (goes to the right AWS facility)
5. Data will be loaded into an S3 bucket
6. Snowball is completely wiped

# What is Edge Computing?

- Process data while it's being created on **an edge location**
  - A truck on the road, a ship on the sea, a mining station underground...



- These locations may have
  - Limited / no internet access
  - Limited / no easy access to computing power
- We setup a **Snowball Edge / Snowcone** device to do edge computing
- Use cases of Edge Computing:
  - Preprocess data
  - Machine learning at the edge
  - Transcoding media streams
- Eventually (if need be) we can ship back the device to AWS (for transferring data for example)

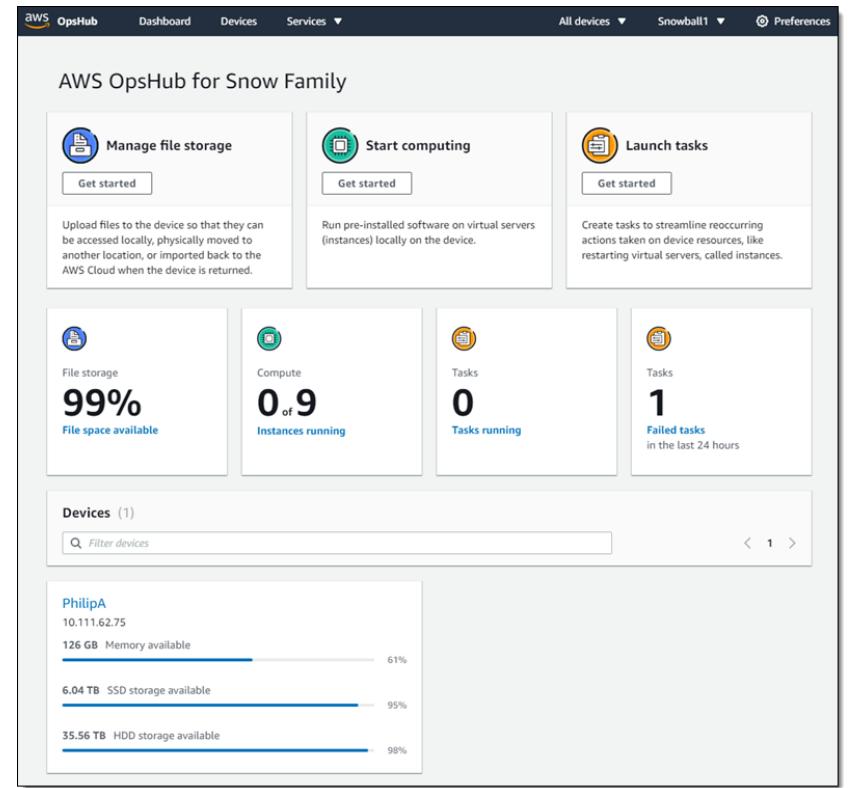
# Snow Family – Edge Computing

- Snowcone (smaller)
  - 2 CPUs, 4 GB of memory, wired or wireless access
  - USB-C power using a cord or the optional battery
- Snowball Edge – Compute Optimized
  - 52 vCPUs, 208 GiB of RAM
  - Optional GPU (useful for video processing or machine learning)
  - 42 TB usable storage
- Snowball Edge – Storage Optimized
  - Up to 40 vCPUs, 80 GiB of RAM
  - Object storage clustering available
- All: Can run EC2 Instances & AWS Lambda functions (using AWS IoT Greengrass)
- Long-term deployment options: 1 and 3 years discounted pricing



# AWS OpsHub

- Historically, to use Snow Family devices, you needed a CLI (Command Line Interface tool)
- Today, you can use **AWS OpsHub** (a software you install on your computer / laptop) to manage your Snow Family Device
  - Unlocking and configuring single or clustered devices
  - Transferring files
  - Launching and managing instances running on Snow Family Devices
  - Monitor device metrics (storage capacity, active instances on your device)
  - Launch compatible AWS services on your devices (ex: Amazon EC2 instances, AWS DataSync, Network File System (NFS))



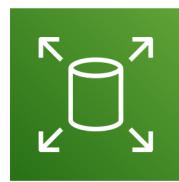
<https://aws.amazon.com/blogs/aws/aws-snowball-edge-update/>

# Hybrid Cloud for Storage

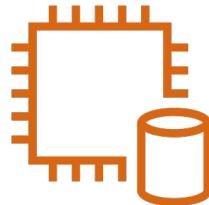
- AWS is pushing for "hybrid cloud"
  - Part of your infrastructure is on-premises
  - Part of your infrastructure is on the cloud
- This can be due to
  - Long cloud migrations
  - Security requirements
  - Compliance requirements
  - IT strategy
- S3 is a proprietary storage technology (unlike EFS / NFS), so how do you expose the S3 data on-premise?
- AWS Storage Gateway!

# AWS Storage Cloud Native Options

## BLOCK



Amazon EBS



EC2 Instance  
Store

## FILE



Amazon EFS

## OBJECT



Amazon S3



Glacier

# AWS Storage Gateway

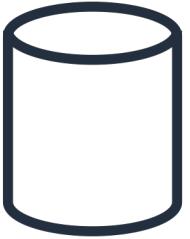
- Bridge between on-premise data and cloud data in S3
- Hybrid storage service to allow on-premises to seamlessly use the AWS Cloud
- Use cases: disaster recovery, backup & restore, tiered storage
- Types of Storage Gateway:
  - File Gateway
  - Volume Gateway
  - Tape Gateway
- No need to know the types at the exam



# Amazon S3 – Summary

- **Buckets vs Objects:** global unique name, tied to a region
- **S3 security:** IAM policy, S3 Bucket Policy (public access), S3 Encryption
- **S3 Websites:** host a static website on Amazon S3
- **S3 Versioning:** multiple versions for files, prevent accidental deletes
- **S3 Access Logs:** log requests made within your S3 bucket
- **S3 Replication:** same-region or cross-region, must enable versioning
- **S3 Storage Classes:** Standard, IA, IZ-IA, Intelligent, Glacier, Glacier Deep Archive
- **S3 Lifecycle Rules:** transition objects between classes
- **S3 Glacier Vault Lock / S3 Object Lock:** WORM (Write Once Read Many)
- **Snow Family:** import data onto S3 through a physical device, edge computing
- **OpsHub:** desktop application to manage Snow Family devices
- **Storage Gateway:** hybrid solution to extend on-premises storage to S3

# Databases Section



# Databases Intro

- Storing data on disk (EFS, EBS, EC2 Instance Store, S3) can have its limits
- Sometimes, you want to store data in a database...
- You can **structure** the data
- You build **indexes** to efficiently **query** / **search** through the data
- You define **relationships** between your **datasets**
  
- Databases are **optimized for a purpose** and come with different features, shapes and constraints

# Relational Databases

- Looks just like Excel spreadsheets, with links between them!
- Can use the SQL language to perform queries / lookups



# NoSQL Databases

- NoSQL = non-SQL = non relational databases
- NoSQL databases are purpose built for specific data models and have flexible schemas for building modern applications.
- Benefits:
  - Flexibility: easy to evolve data model
  - Scalability: designed to scale-out by using distributed clusters
  - High-performance: optimized for a specific data model
  - Highly functional: types optimized for the data model
- Examples: Key-value, document, graph, in-memory, search databases

# NoSQL data example: JSON

- JSON = JavaScript Object Notation
- JSON is a common form of data that fits into a NoSQL model
- Data can be nested
- Fields can **change** over time
- Support for new types: arrays, etc...

```
{  
  "name": "John",  
  "age": 30,  
  "cars": [  
    "Ford",  
    "BMW",  
    "Fiat"  
,  
  "address": {  
    "type": "house",  
    "number": 23,  
    "street": "Dream Road"  
  }  
}
```

# Databases & Shared Responsibility on AWS

- AWS offers use to **manage** different databases
- **Benefits** include:
  - Quick Provisioning, High Availability, Vertical and Horizontal Scaling
  - Automated Backup & Restore, Operations, Upgrades
  - Operating System Patching is handled by AWS
  - Monitoring, alerting
- Note: many databases technologies could be run on EC2, but you must handle yourself the resiliency, backup, patching, high availability, fault tolerance, scaling...

# AWS RDS Overview

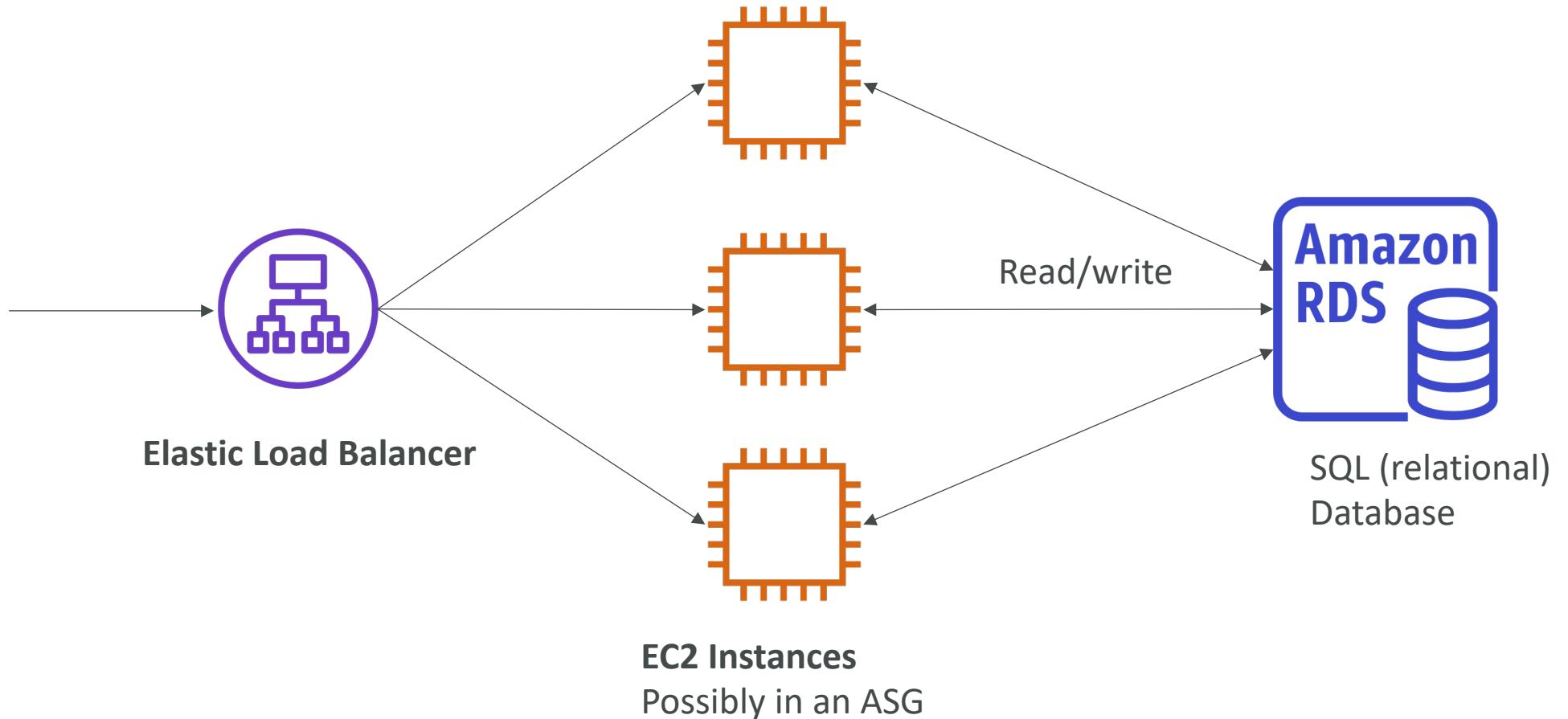


- RDS stands for Relational Database Service
- It's a managed DB service for DB use SQL as a query language.
- It allows you to create databases in the cloud that are managed by AWS
  - Postgres
  - MySQL
  - MariaDB
  - Oracle
  - Microsoft SQL Server
  - Aurora (AWS Proprietary database)

# Advantage over using RDS versus deploying DB on EC2

- RDS is a managed service:
  - Automated provisioning, OS patching
  - Continuous backups and restore to specific timestamp (Point in Time Restore)!
  - Monitoring dashboards
  - Read replicas for improved read performance
  - Multi AZ setup for DR (Disaster Recovery)
  - Maintenance windows for upgrades
  - Scaling capability (vertical and horizontal)
  - Storage backed by EBS (gp2 or io1)
- BUT you can't SSH into your instances

# RDS Solution Architecture





# Amazon Aurora

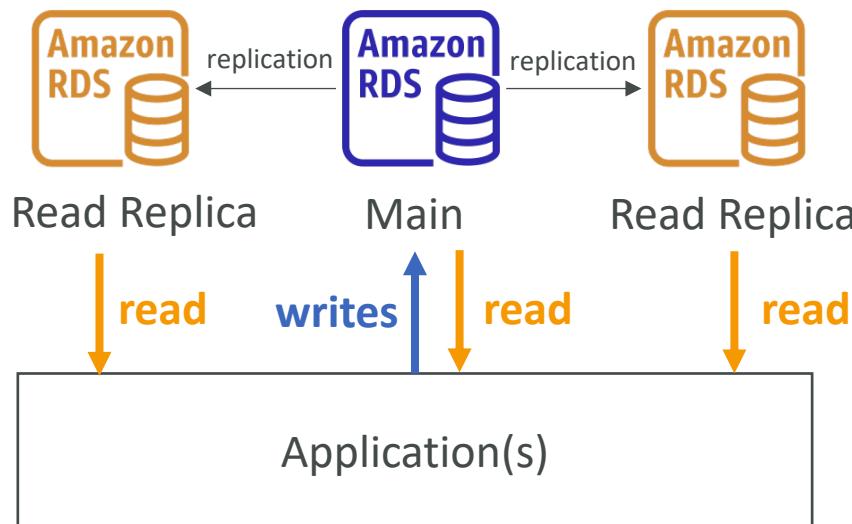
- Aurora is a proprietary technology from AWS (not open sourced)
- PostgreSQL and MySQL are both supported as Aurora DB
- Aurora is “AWS cloud optimized” and claims 5x performance improvement over MySQL on RDS, over 3x the performance of Postgres on RDS
- Aurora storage automatically grows in increments of 10GB, up to 64 TB.
- Aurora costs more than RDS (20% more) – but is more efficient
- Not in the free tier



# RDS Deployments: Read Replicas, Multi-AZ

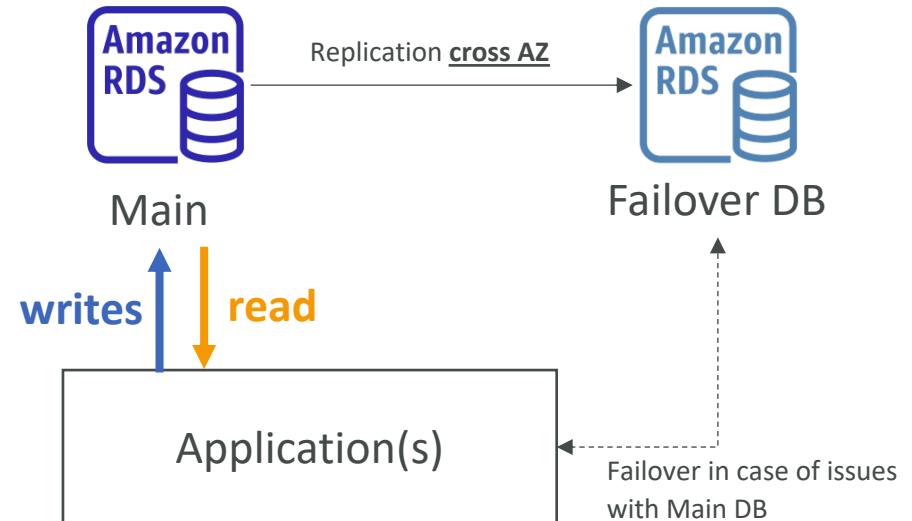
- **Read Replicas:**

- Scale the read workload of your DB
- Can create up to 5 Read Replicas
- Data is only written to the main DB



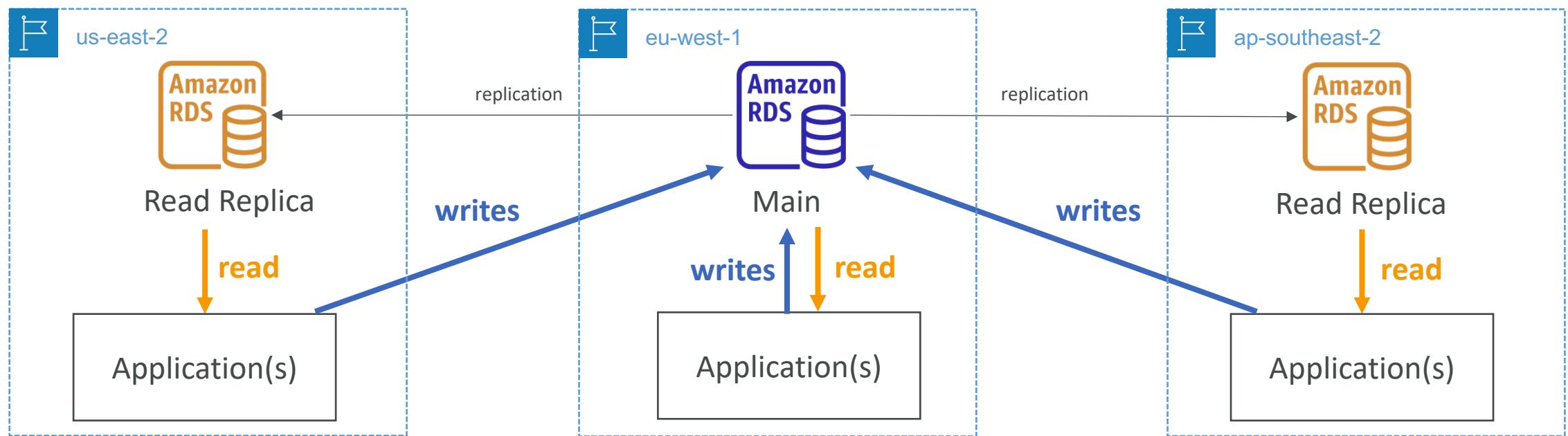
- **Multi-AZ:**

- Failover in case of AZ outage (high availability)
- Data is only read/written to the main database
- Can only have 1 other AZ as failover



# RDS Deployments: Multi-Region

- Multi-Region (Read Replicas)
  - Disaster recovery in case of region issue
  - Local performance for global reads
  - Replication cost

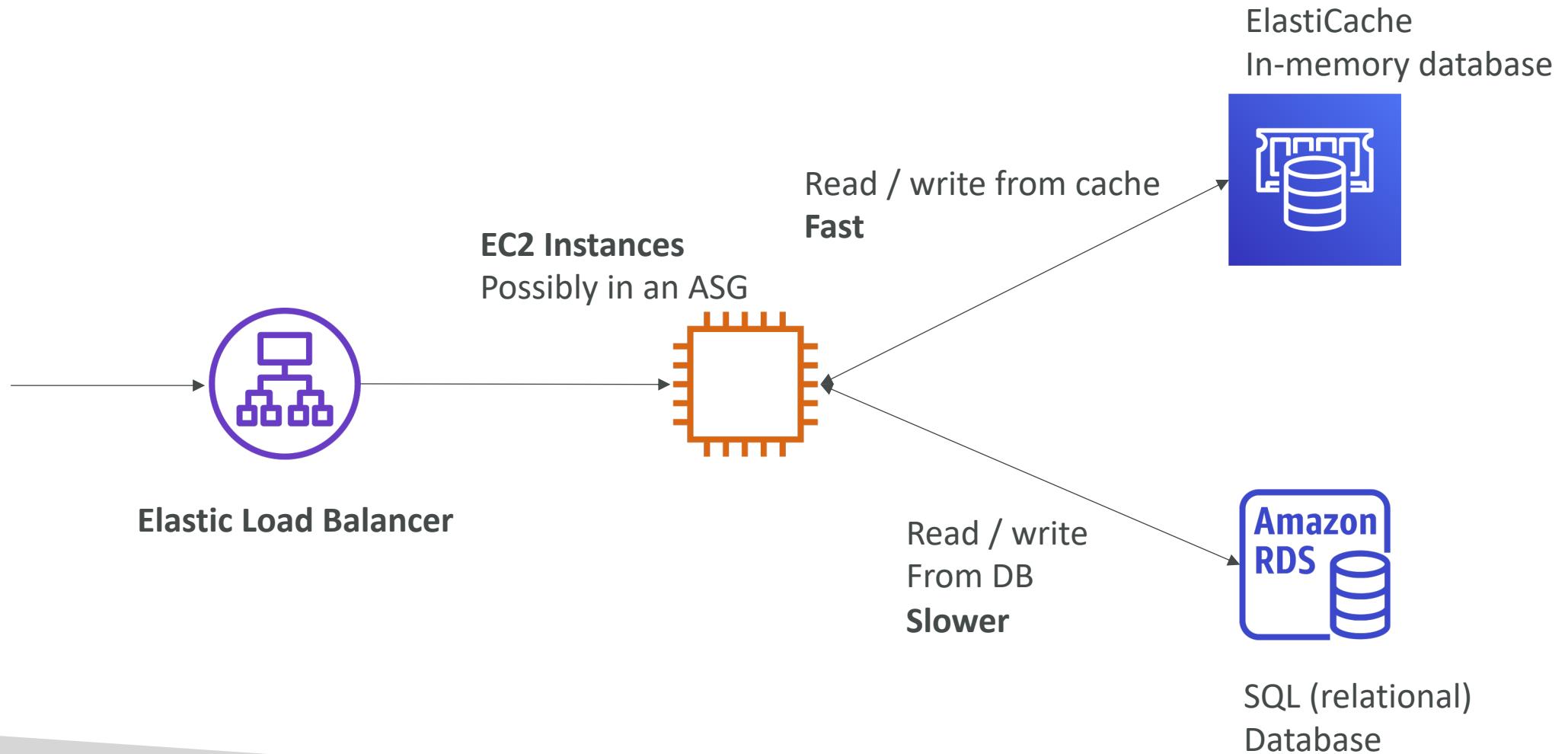




# Amazon ElastiCache Overview

- The same way RDS is to get managed Relational Databases...
- ElastiCache is to get managed Redis or Memcached
- Caches are **in-memory databases** with high performance, low latency
- Helps reduce load off databases for read intensive workloads
  
- AWS takes care of OS maintenance / patching, optimizations, setup, configuration, monitoring, failure recovery and backups

# ElastiCache Solution Architecture - Cache



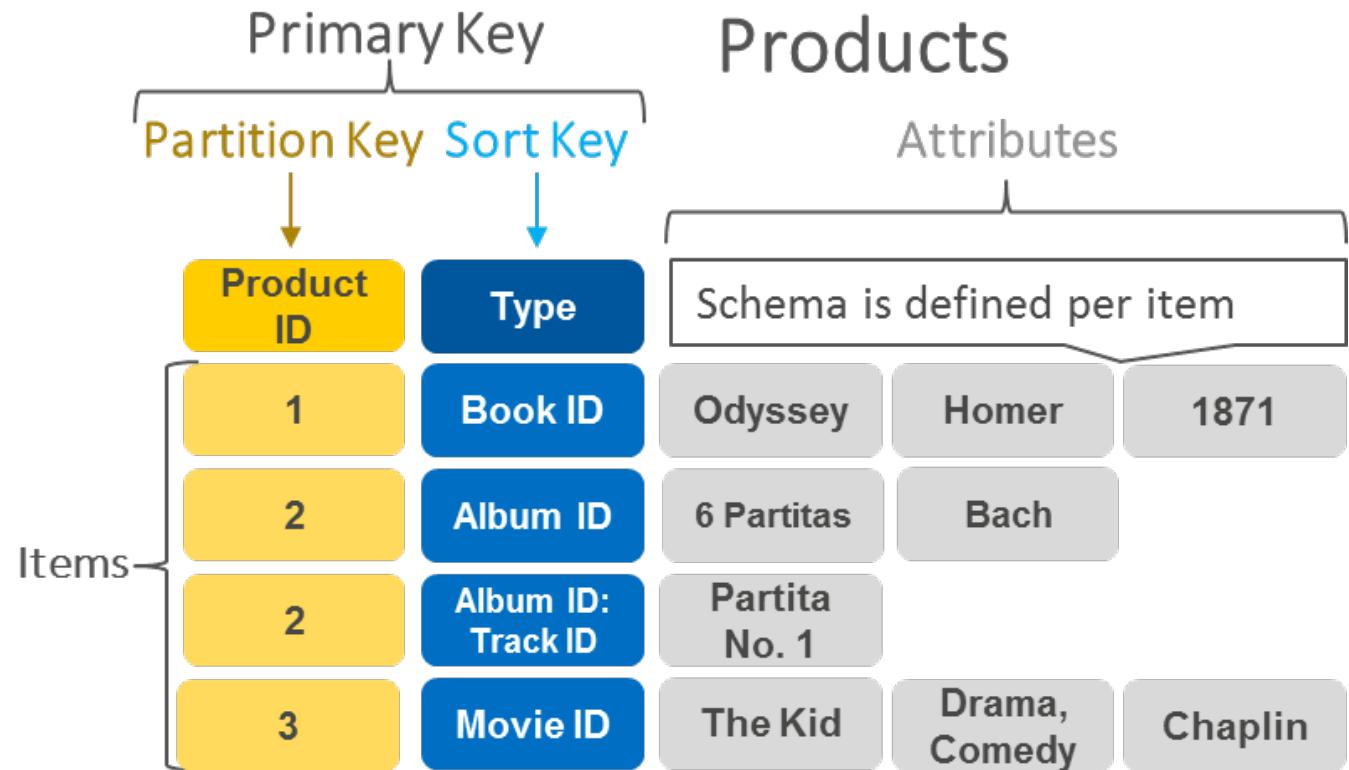
# DynamoDB



- Fully Managed Highly available with replication across 3 AZ
- **NoSQL database - not a relational database**
- Scales to massive workloads, distributed “serverless” database
- Millions of requests per seconds, trillions of row, 100s of TB of storage
- Fast and consistent in performance
- **Single-digit millisecond latency – low latency retrieval**
- Integrated with IAM for security, authorization and administration
- Low cost and auto scaling capabilities

# DynamoDB – type of data

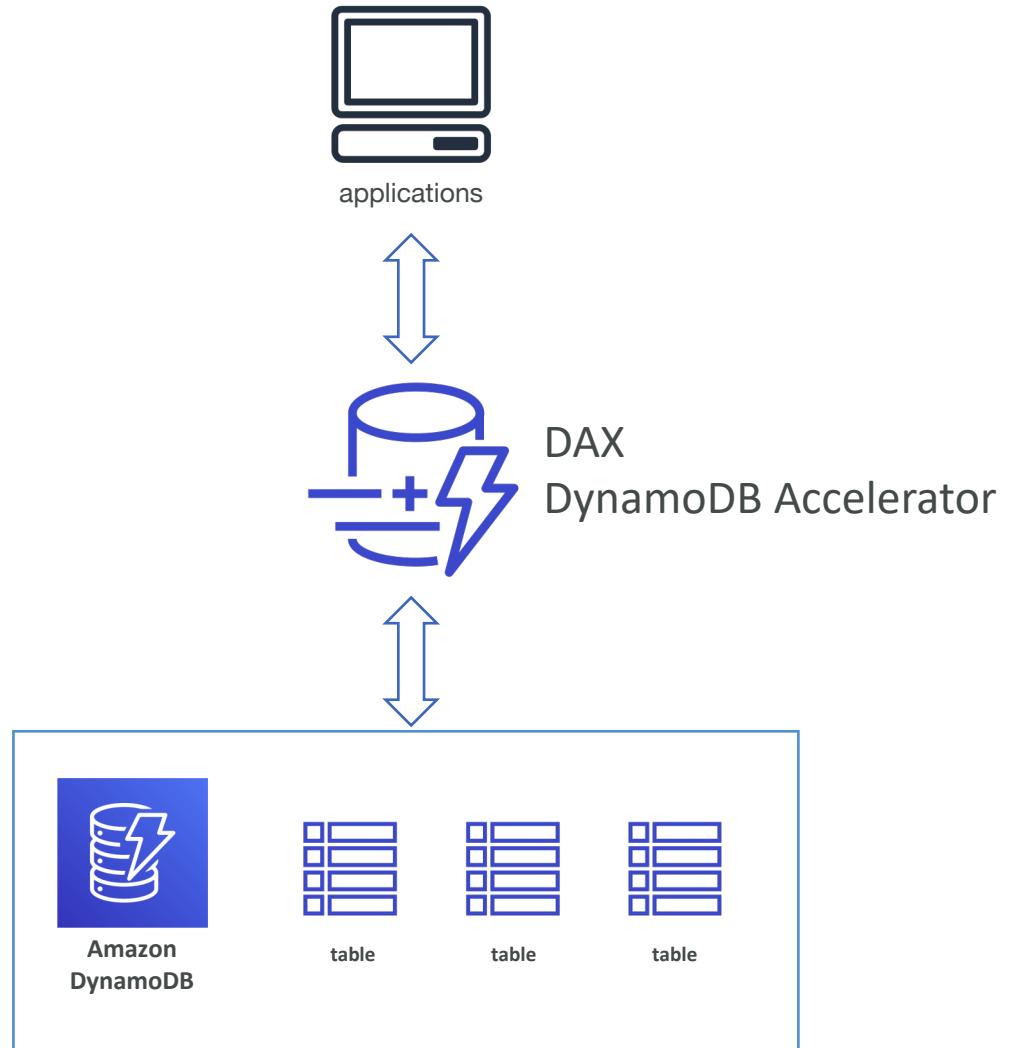
- DynamoDB is a key/value database



<https://aws.amazon.com/nosql/key-value/>

# DynamoDB Accelerator - DAX

- Fully Managed in-memory cache for DynamoDB
- 10x performance improvement – single-digit millisecond latency to microseconds latency – when accessing your DynamoDB tables
- Secure, highly scalable & highly available
- Difference with ElastiCache at the CCP level: **DAX is only used for and is integrated with DynamoDB**, while ElastiCache can be used for other databases

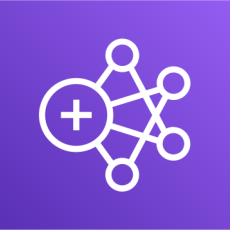


# Redshift Overview



- Redshift is based on PostgreSQL, but it's not used for OLTP
- It's **OLAP** – online analytical processing (analytics and data warehousing)
- Load data once every hour, not every second
- 10x better performance than other data warehouses, scale to PBs of data
- **Columnar** storage of data (instead of row based)
- Massively Parallel Query Execution (MPP), highly available
- Pay as you go based on the instances provisioned
- Has a SQL interface for performing the queries
- BI tools such as AWS Quicksight or Tableau integrate with it

# Amazon EMR



- EMR stands for “Elastic MapReduce”
- EMR helps creating **Hadoop clusters (Big Data)** to analyze and process vast amount of data
- The clusters can be made of hundreds of EC2 instances
- Also supports Apache Spark, HBase, Presto, Flink...
- EMR takes care of all the provisioning and configuration
- Auto-scaling and integrated with Spot instances
- Use cases: data processing, machine learning, web indexing, big data...

# Athena Overview

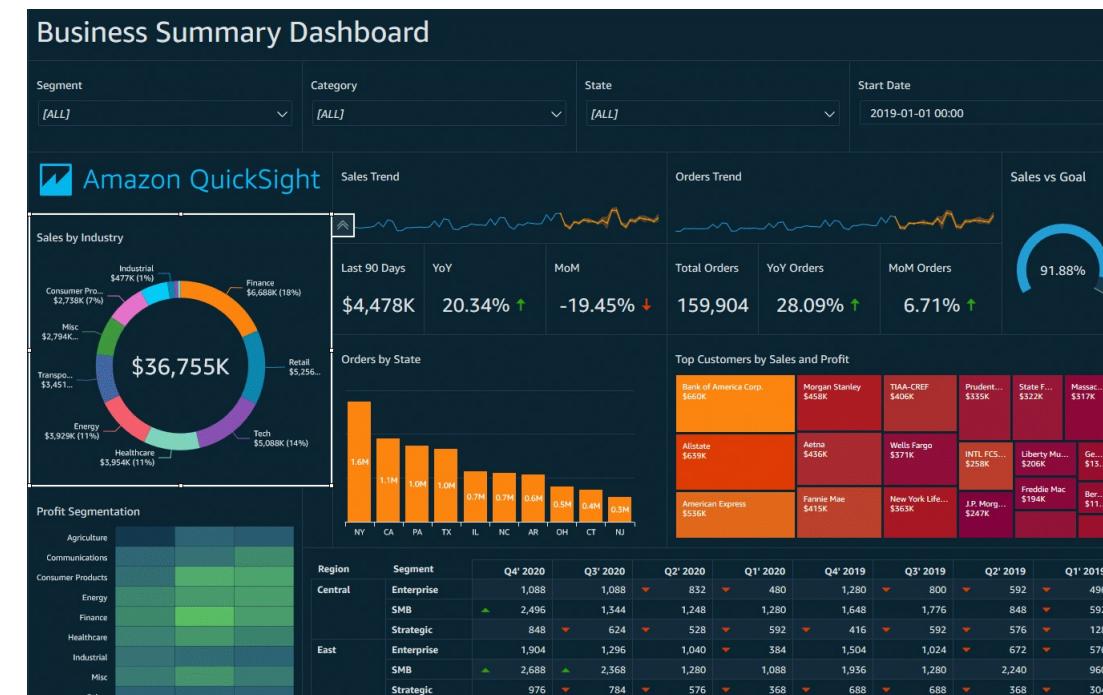


- Fully Serverless database with SQL capabilities
  - Used to query data in S3
  - Pay per query
  - Output results back to S3
  - Secured through IAM
- 
- **Use Case:** one-time SQL queries, serverless queries on S3, log analytics

# Amazon QuickSight



- Serverless machine learning-powered business intelligence service to create interactive dashboards
- Fast, automatically scalable, embeddable, with per-session pricing
- Use cases:
  - Business analytics
  - Building visualizations
  - Perform ad-hoc analysis
  - Get business insights using data
- Integrated with RDS, Aurora, Athena, Redshift, S3...



<https://aws.amazon.com/quicksight/>

# DocumentDB

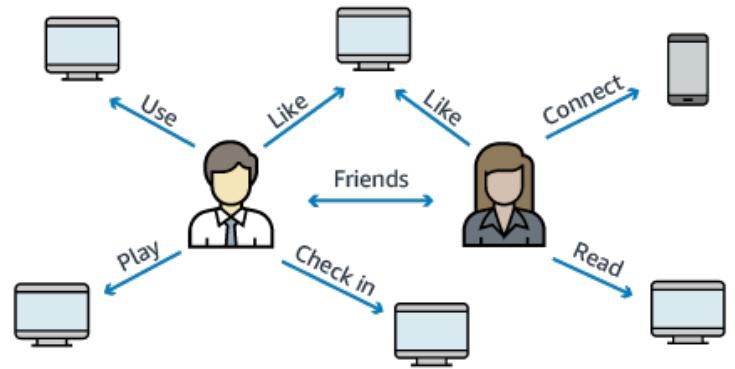


- Aurora is an “AWS-implementation” of PostgreSQL / MySQL ...
- DocumentDB is the same for MongoDB (which is a NoSQL database)
  
- MongoDB is used to store, query, and index JSON data
- Similar “deployment concepts” as Aurora
- Fully Managed, highly available with replication across 3 AZ
- Aurora storage automatically grows in increments of 10GB, up to 64 TB.
  
- Automatically scales to workloads with millions of requests per seconds

# Amazon Neptune



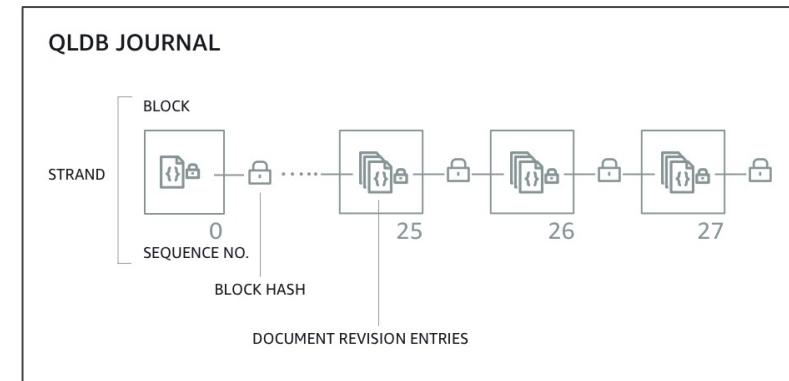
- Fully managed **graph** database
- A popular **graph dataset** would be a **social network**
  - Users have friends
  - Posts have comments
  - Comments have likes from users
  - Users share and like posts...
- Highly available across 3 AZ, with up to 15 read replicas
- Build and run applications working with highly connected datasets – optimized for these complex and hard queries
- Can store up to billions of relations and query the graph with milliseconds latency
- Highly available with replications across multiple AZs
- Great for knowledge graphs (Wikipedia), fraud detection, recommendation engines, social networking



# Amazon QLDB



- QLDB stands for "Quantum Ledger Database"
- A ledger is a book **recording financial transactions**
- Fully Managed, Serverless, High available, Replication across 3 AZ
- Used to **review history of all the changes made to your application data** over time
- **Immutable** system: no entry can be removed or modified, cryptographically verifiable



- 2-3x better performance than common ledger blockchain frameworks, manipulate data using SQL
- Difference with Amazon Managed Blockchain: **no decentralization component**, in accordance with financial regulation rules

<https://docs.aws.amazon.com/qldb/latest/developerguide/ledger-structure.html>

# Amazon Managed Blockchain



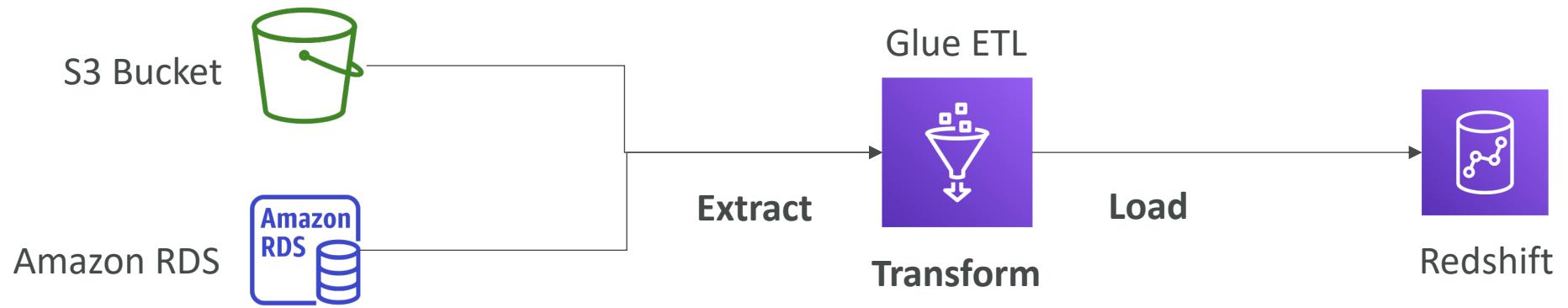
- Blockchain makes it possible to build applications where multiple parties can execute transactions **without the need for a trusted, central authority.**
- Amazon Managed Blockchain is a managed service to:
  - Join public blockchain networks
  - Or create your own scalable private network
- Compatible with the frameworks Hyperledger Fabric & Ethereum



# AWS Glue

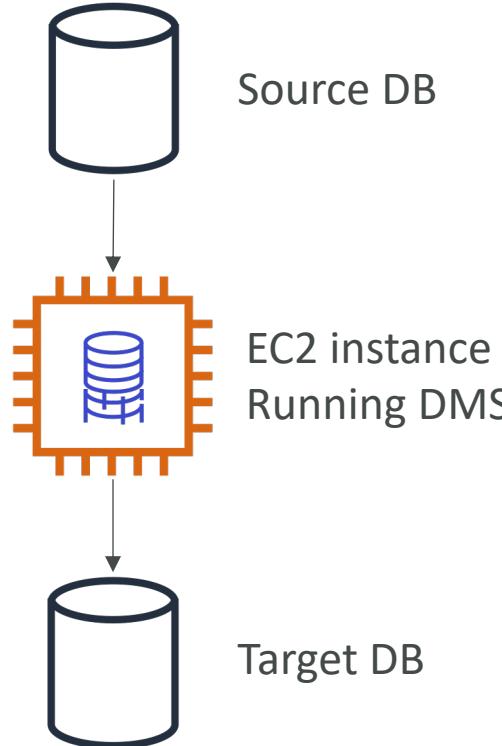


- Managed **extract, transform, and load (ETL)** service
- Useful to prepare and transform data for analytics
- Fully **serverless** service



- Glue Data Catalog: catalog of datasets
  - can be used by Athena, Redshift, EMR

# DMS – Database Migration Service



- Quickly and securely migrate databases to AWS, resilient, self healing
- The source database remains available during the migration
- Supports:
  - Homogeneous migrations: ex Oracle to Oracle
  - Heterogeneous migrations: ex Microsoft SQL Server to Aurora

# Databases & Analytics Summary in AWS

- Relational Databases - OLTP: RDS & Aurora (SQL)
- Differences between Multi-AZ, Read Replicas, Multi-Region
- In-memory Database: ElastiCache
- Key/Value Database: DynamoDB (serverless) & DAX (cache for DynamoDB)
- Warehouse - OLAP: Redshift (SQL)
- Hadoop Cluster: EMR
- Athena: query data on Amazon S3 (serverless & SQL)
- QuickSight: dashboards on your data (serverless)
- DocumentDB: “Aurora for MongoDB” (JSON – NoSQL database)
- Amazon QLDB: Financial Transactions Ledger (immutable journal, cryptographically verifiable)
- Amazon Managed Blockchain: managed Hyperledger Fabric & Ethereum blockchains
- Glue: Managed ETL (Extract Transform Load) and Data Catalog service
- Database Migration: DMS
- Neptune: graph database

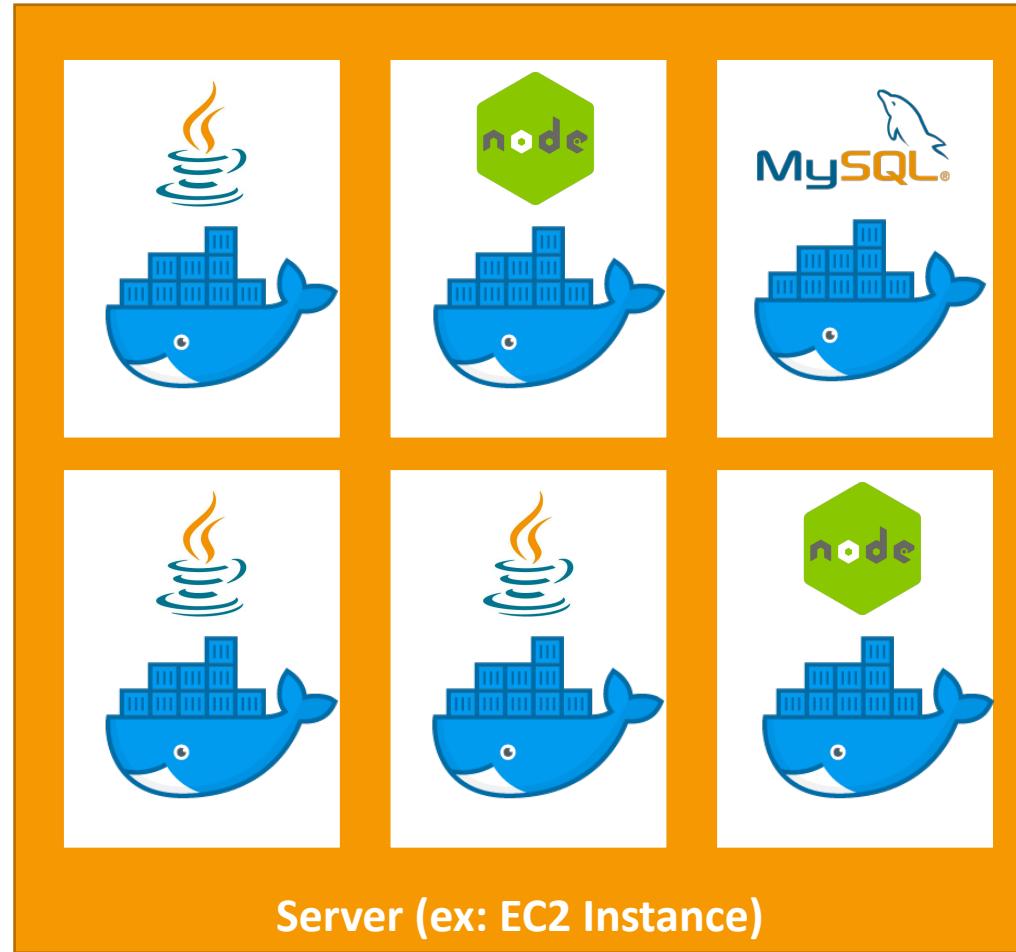
# Other Compute Section

# What is Docker?



- Docker is a software development platform to deploy apps
- Apps are packaged in **containers** that can be run on any OS
- **Apps run the same, regardless of where they're run**
  - Any machine
  - No compatibility issues
  - Predictable behavior
  - Less work
  - Easier to maintain and deploy
  - Works with any language, any OS, any technology
- Scale containers up and down very quickly (seconds)

# Docker on an OS

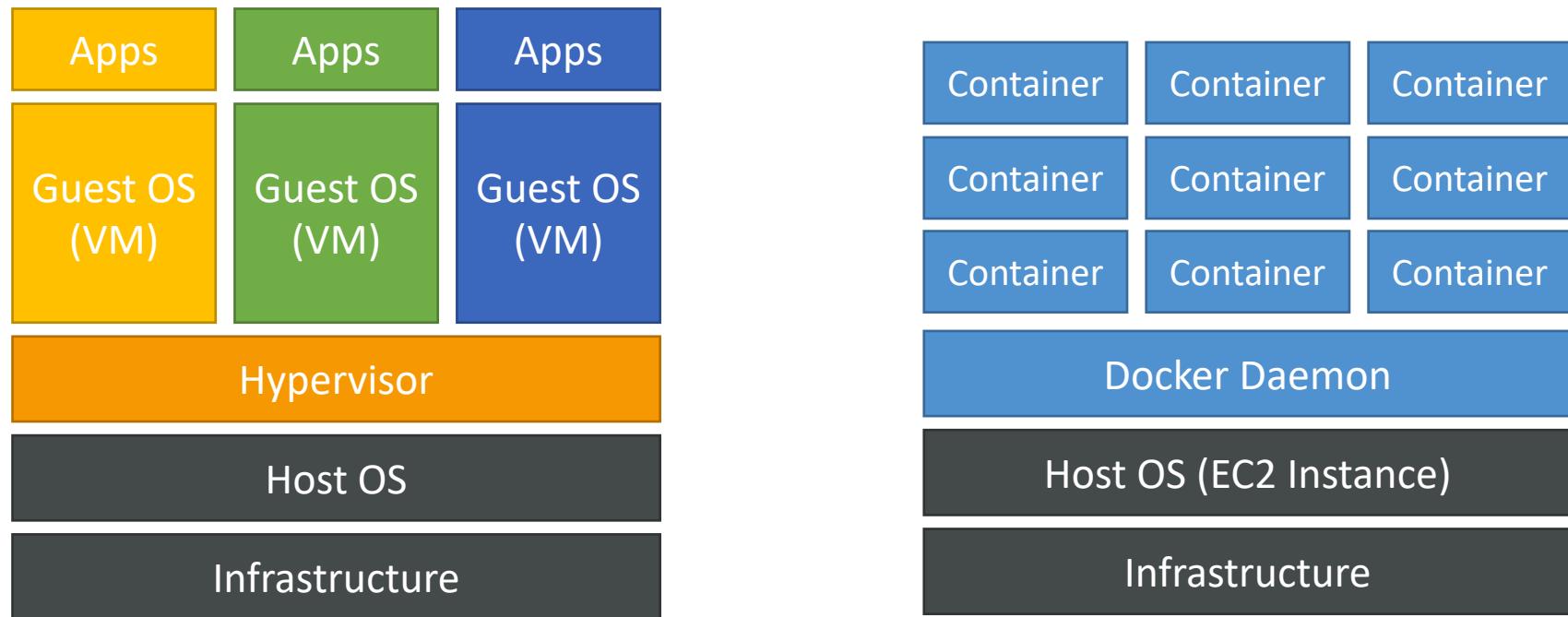


# Where Docker images are stored?

- Docker images are stored in Docker Repositories
- Public: Docker Hub <https://hub.docker.com/>
  - Find base images for many technologies or OS:
  - Ubuntu
  - MySQL
  - NodeJS, Java...
- Private: Amazon ECR (Elastic Container Registry)

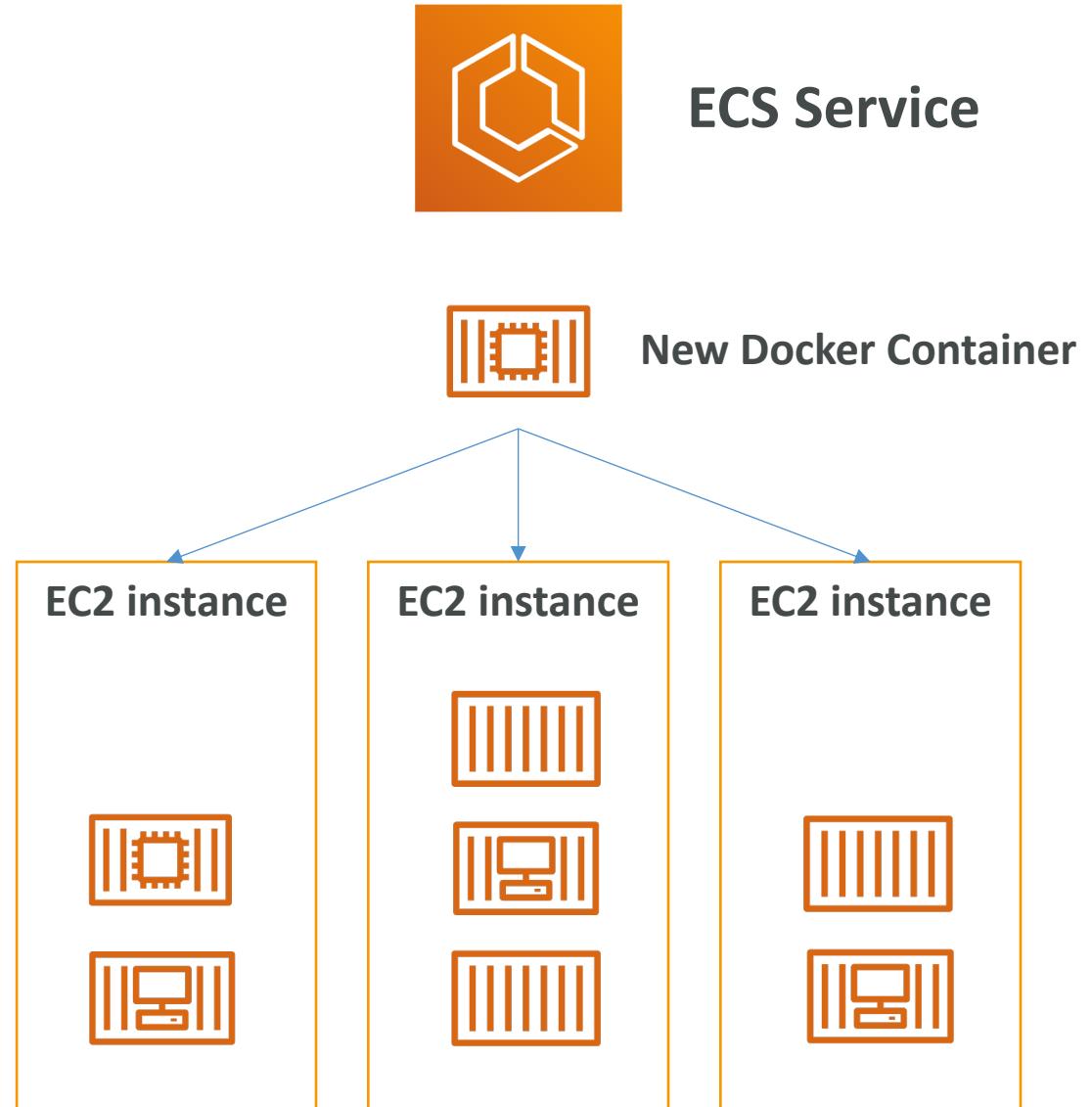
# Docker versus Virtual Machines

- Docker is "sort of" a virtualization technology, but not exactly
- Resources are shared with the host => many containers on one server



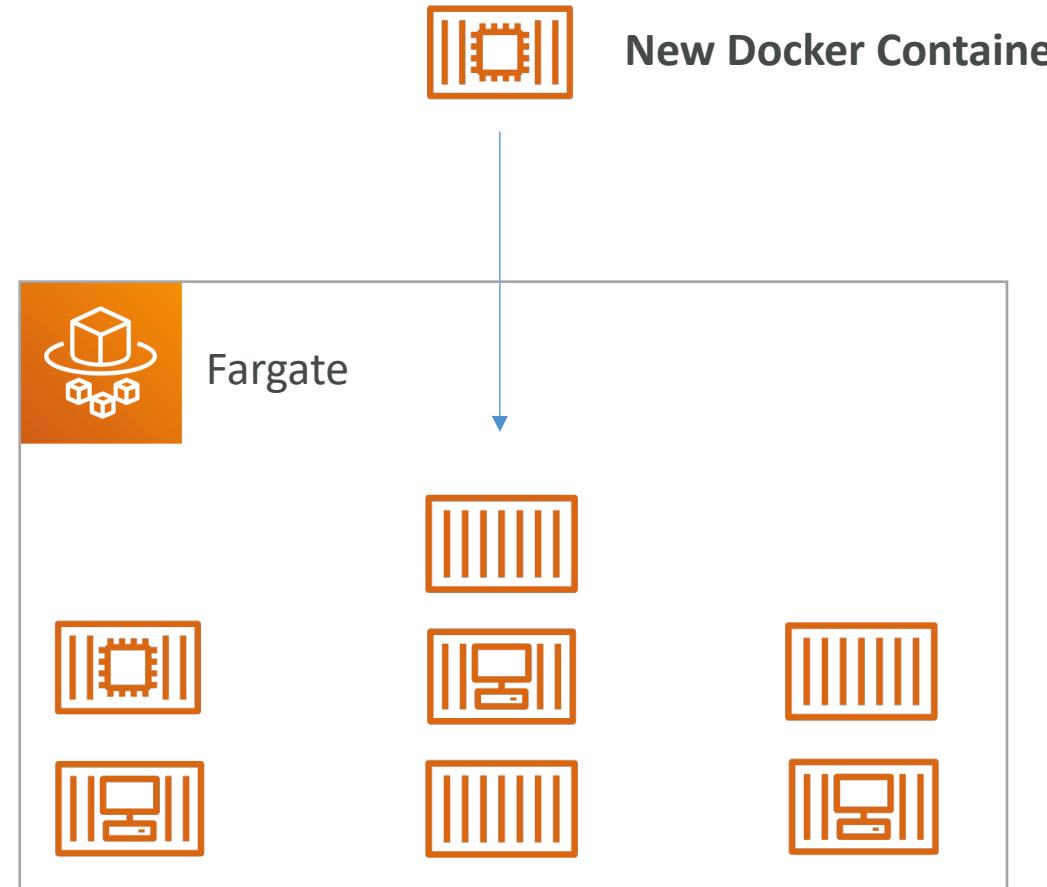
# ECS

- ECS = Elastic Container Service
- Launch Docker containers on AWS
- You must provision & maintain the infrastructure (the EC2 instances)
- AWS takes care of starting / stopping containers
- Has integrations with the Application Load Balancer



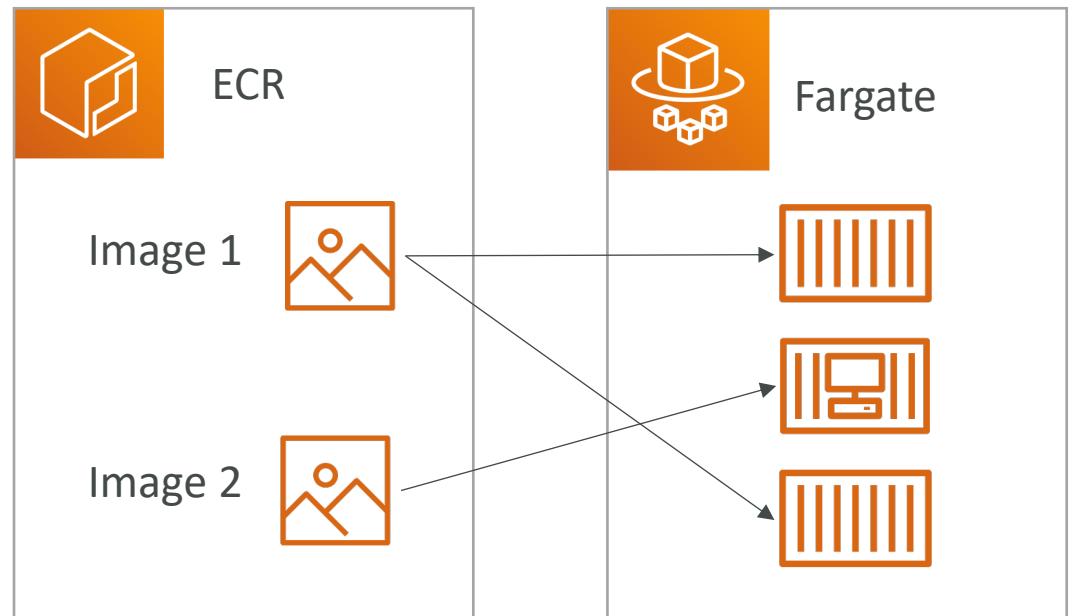
# Fargate

- Launch Docker containers on AWS
- You do not provision the infrastructure (no EC2 instances to manage) – simpler!
- Serverless offering
- AWS just runs containers for you based on the CPU / RAM you need



# ECR

- Elastic Container Registry
- Private Docker Registry on AWS
- This is where you **store your Docker images** so they can be run by ECS or Fargate



# What's serverless?

- Serverless is a new paradigm in which the developers don't have to manage servers anymore...
- They just deploy code
- They just deploy... functions !
- Initially... Serverless == FaaS (Function as a Service)
- Serverless was pioneered by AWS Lambda but now also includes anything that's managed: “databases, messaging, storage, etc.”
- **Serverless does not mean there are no servers...**  
it means you just don't manage / provision / see them

# So far in this course...



Amazon S3



DynamoDB

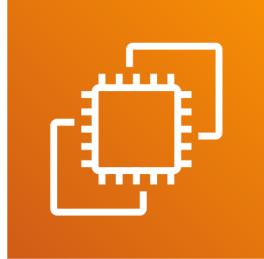


Fargate



Lambda

# Why AWS Lambda



Amazon EC2

- Virtual Servers in the Cloud
  - Limited by RAM and CPU
  - Continuously running
  - Scaling means intervention to add / remove servers
- 



Amazon Lambda

- Virtual functions – no servers to manage!
- Limited by time - short executions
- Run on-demand
- Scaling is automated!

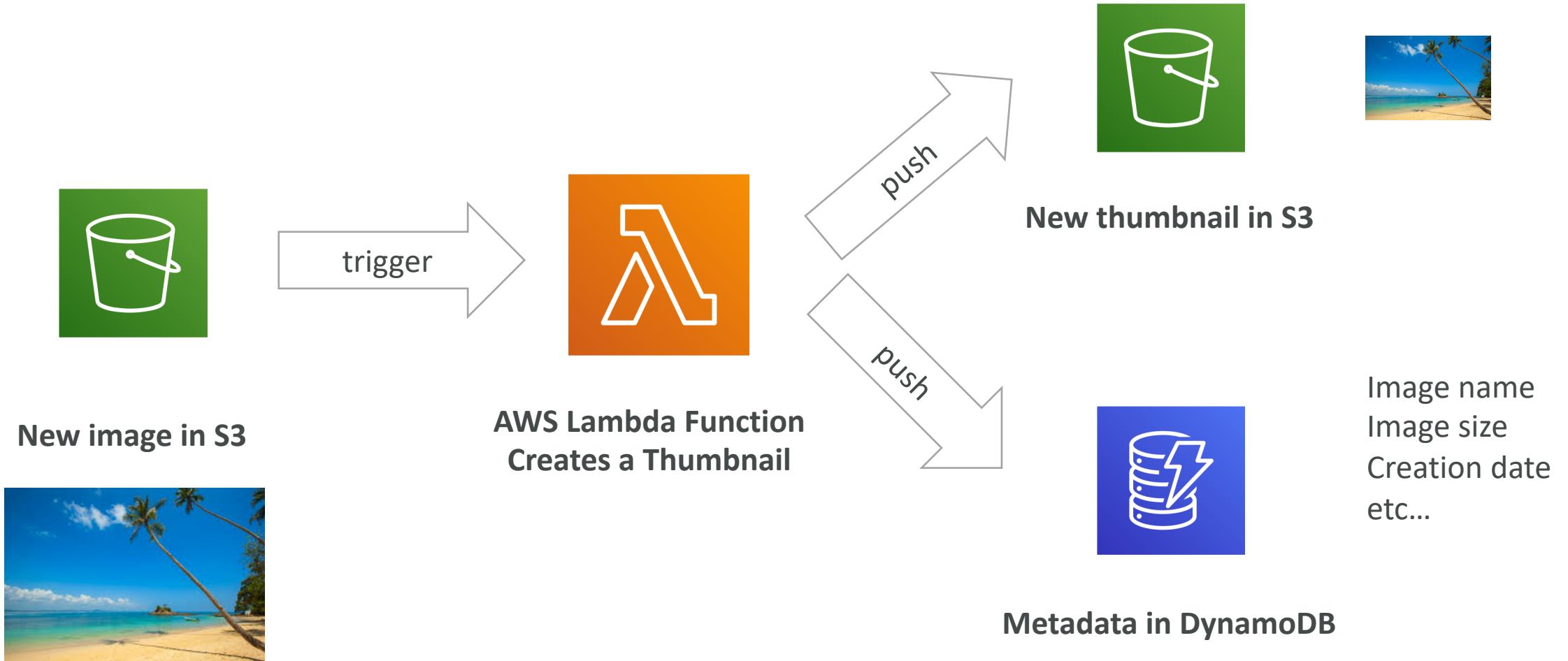
# Benefits of AWS Lambda

- Easy Pricing:
  - Pay per request and compute time
  - Free tier of 1,000,000 AWS Lambda requests and 400,000 GBs of compute time
- Integrated with the whole AWS suite of services
- **Event-Driven:** functions get invoked by AWS when needed
- Integrated with many programming languages
- Easy monitoring through AWS CloudWatch
- Easy to get more resources per functions (up to 10GB of RAM!)
- Increasing RAM will also improve CPU and network!

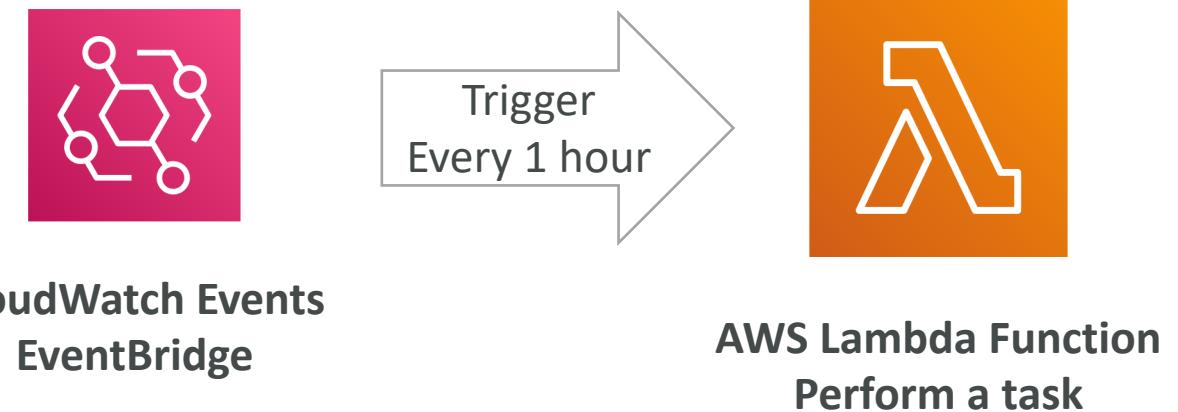
# AWS Lambda language support

- Node.js (JavaScript)
- Python
- Java (Java 8 compatible)
- C# (.NET Core)
- Golang
- C# / Powershell
- Ruby
- Custom Runtime API (community supported, example Rust)
- Lambda Container Image
  - The container image must implement the Lambda Runtime API
  - ECS / Fargate is preferred for running arbitrary Docker images

# Example: Serverless Thumbnail creation



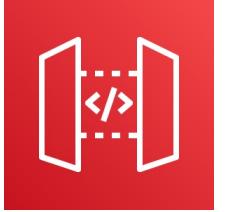
# Example: Serverless CRON Job



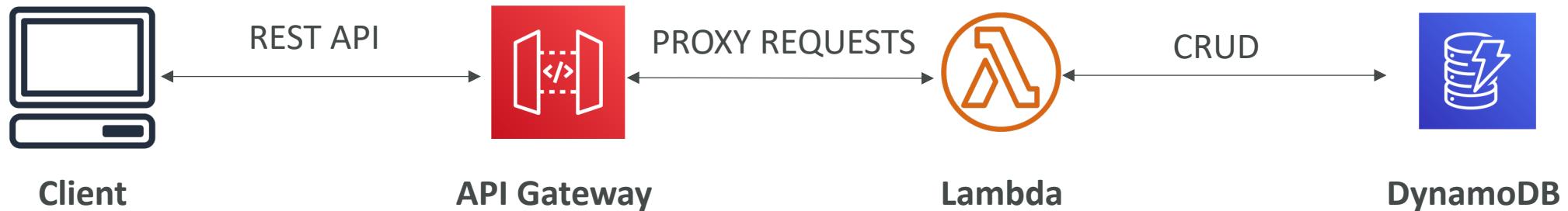
# AWS Lambda Pricing: example

- You can find overall pricing information here:  
<https://aws.amazon.com/lambda/pricing/>
- Pay per calls:
  - First 1,000,000 requests are free
  - \$0.20 per 1 million requests thereafter (\$0.0000002 per request)
- Pay per duration: (in increment of 1 ms)
  - 400,000 GB-seconds of compute time per month for FREE
  - == 400,000 seconds if function is 1 GB RAM
  - == 3,200,000 seconds if function is 128 MB RAM
  - After that \$1.00 for 600,000 GB-seconds
- It is usually very cheap to run AWS Lambda so it's very popular

# Amazon API Gateway



- Example: building a serverless API



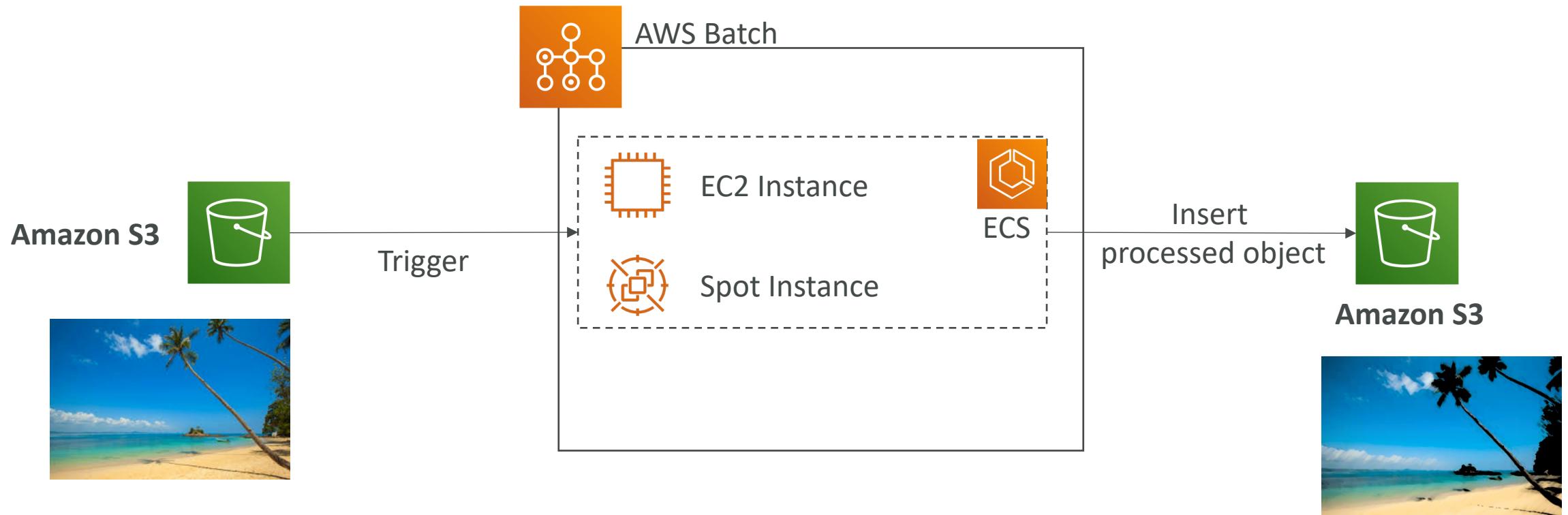
- Fully managed service for developers to easily create, publish, maintain, monitor, and secure APIs
- **Serverless** and scalable
- Supports RESTful APIs and WebSocket APIs
- Support for security, user authentication, API throttling, API keys, monitoring...

# AWS Batch



- Fully managed batch processing at any scale
- Efficiently run 100,000s of computing batch jobs on AWS
- A “batch” job is a job with a start and an end (opposed to continuous)
- Batch will dynamically launch **EC2 instances** or **Spot Instances**
- AWS Batch provisions the right amount of compute / memory
- You submit or schedule batch jobs and AWS Batch does the rest!
- Batch jobs are defined as **Docker images** and run on **ECS**
- Helpful for cost optimizations and focusing less on the infrastructure

# AWS Batch – Simplified Example



# Batch vs Lambda

- Lambda:
  - Time limit
  - Limited runtimes
  - Limited temporary disk space
  - Serverless
- Batch:
  - No time limit
  - Any runtime as long as it's packaged as a Docker image
  - Rely on EBS / instance store for disk space
  - Relies on EC2 (can be managed by AWS)



# Amazon Lightsail



- Virtual servers, storage, databases, and networking
- Low & predictable pricing
- Simpler alternative to using EC2, RDS, ELB, EBS, Route 53...
- Great for people with **little cloud experience!**
- Can setup notifications and monitoring of your Lightsail resources
- Use cases:
  - Simple web applications (has templates for LAMP, Nginx, MEAN, Node.js...)
  - Websites (templates for WordPress, Magento, Plesk, Joomla)
  - Dev / Test environment
- Has high availability but no auto-scaling, limited AWS integrations

# Other Compute - Summary

- **Docker:** container technology to run applications
- **ECS:** run Docker containers on EC2 instances
- **Fargate:**
  - Run Docker containers without provisioning the infrastructure
  - Serverless offering (no EC2 instances)
- **ECR:** Private Docker Images Repository
- **Batch:** run batch jobs on AWS across managed EC2 instances
- **Lightsail:** predictable & low pricing for simple application & DB stacks

# Lambda Summary

- Lambda is Serverless, Function as a Service, seamless scaling, reactive
- **Lambda Billing:**
  - By the time run x by the RAM provisioned
  - By the number of invocations
- **Language Support:** many programming languages except (arbitrary) Docker
- **Invocation time:** up to 15 minutes
- **Use cases:**
  - Create Thumbnails for images uploaded onto S3
  - Run a Serverless cron job
- **API Gateway:** expose Lambda functions as HTTP API

# Deploying and Managing Infrastructure at Scale Section



# What is CloudFormation

- CloudFormation is a declarative way of outlining your AWS Infrastructure, for any resources (most of them are supported).
- For example, within a CloudFormation template, you say:
  - I want a security group
  - I want two EC2 instances using this security group
  - I want an S3 bucket
  - I want a load balancer (ELB) in front of these machines
- Then CloudFormation creates those for you, in the **right order**, with the **exact configuration** that you specify

# Benefits of AWS CloudFormation (1/2)

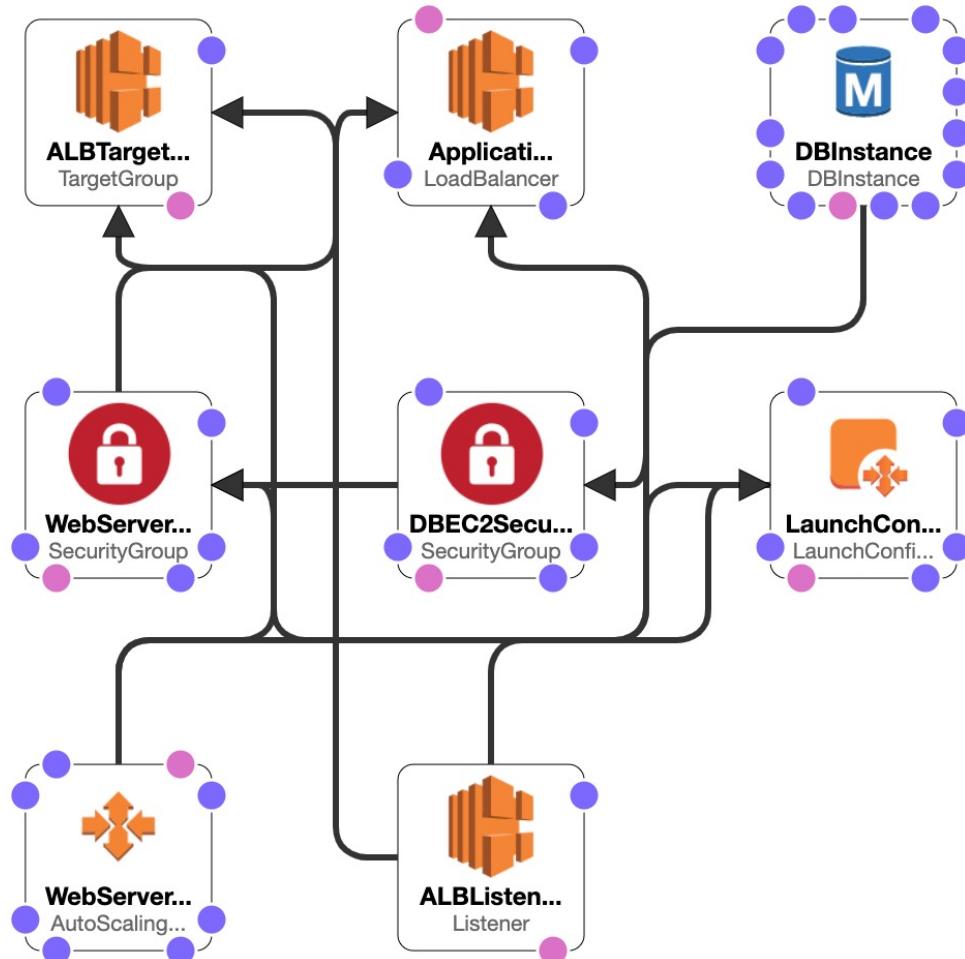
- Infrastructure as code
  - No resources are manually created, which is excellent for control
  - Changes to the infrastructure are reviewed through code
- Cost
  - Each resources within the stack is tagged with an identifier so you can easily see how much a stack costs you
  - You can estimate the costs of your resources using the CloudFormation template
  - Savings strategy: In Dev, you could automation deletion of templates at 5 PM and recreated at 8 AM, safely

# Benefits of AWS CloudFormation (2/2)

- Productivity
  - Ability to destroy and re-create an infrastructure on the cloud on the fly
  - Automated generation of Diagram for your templates!
  - Declarative programming (no need to figure out ordering and orchestration)
- Don't re-invent the wheel
  - Leverage existing templates on the web!
  - Leverage the documentation
- Supports (almost) all AWS resources:
  - Everything we'll see in this course is supported
  - You can use "custom resources" for resources that are not supported

# CloudFormation Stack Designer

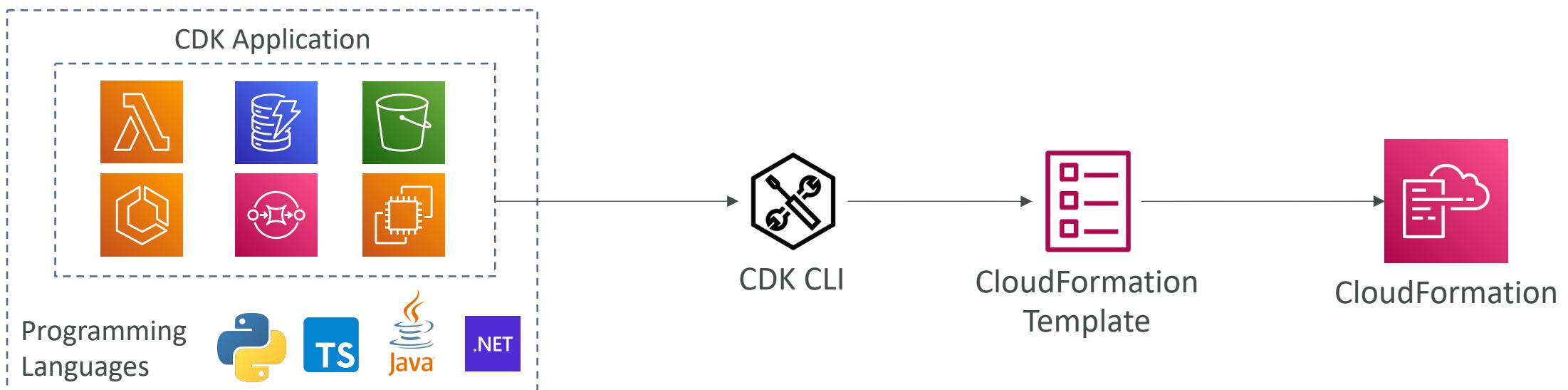
- Example: WordPress CloudFormation Stack
- We can see all the resources
- We can see the relations between the components





# AWS Cloud Development Kit (CDK)

- Define your cloud infrastructure using a familiar language:
  - JavaScript/TypeScript, Python, Java, and .NET
- The code is “compiled” into a CloudFormation template (JSON/YAML)
- You can therefore deploy infrastructure and application runtime code together
  - Great for Lambda functions
  - Great for Docker containers in ECS / EKS



# CDK Example

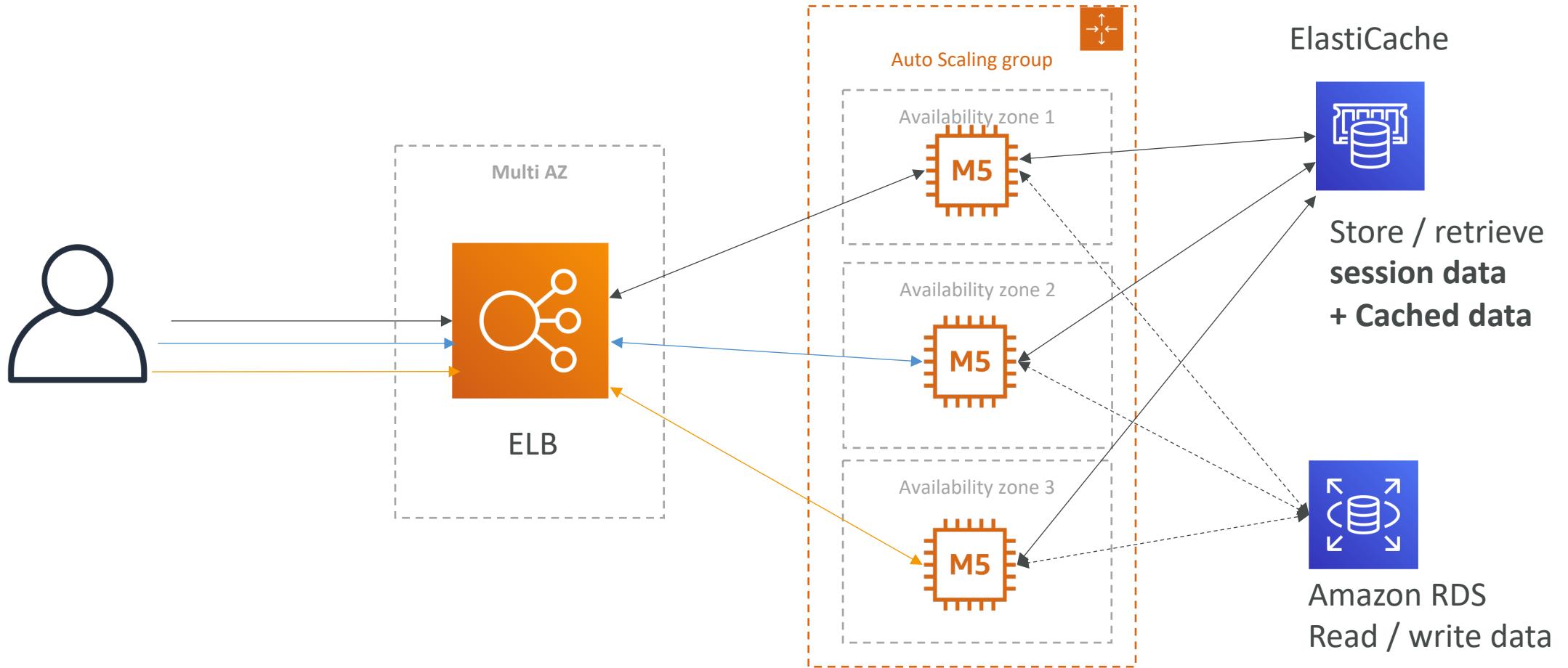
```
export class MyEcsConstructStack extends core.Stack {
  constructor(scope: core.App, id: string, props?: core.StackProps) {
    super(scope, id, props);

    const vpc = new ec2.Vpc(this, "MyVpc", {
      maxAzs: 1 // Default is all AZs in region
    });

    const cluster = new ecs.Cluster(this, "MyCluster", {
      vpc
    });

    // Create a load-balanced Fargate service and make it public
    new ecs_patterns.ApplicationLoadBalancedFargateService(this, "My
      cluster: cluster, // Required
      cpu: 512, // Default is 256
      desiredCount: 6, // Default is 1
      taskImageOptions: { image: ecs.ContainerImage.fromRegistry("an
      memoryLimitMiB: 2048, // Default is 512
      publicLoadBalancer: true // Default is false
    });
  }
}
```

# Typical architecture: Web App 3-tier



# Developer problems on AWS

- Managing infrastructure
  - Deploying Code
  - Configuring all the databases, load balancers, etc
  - Scaling concerns
- 
- Most web apps have the same architecture (ALB + ASG)
  - All the developers want is for their code to run!
  - Possibly, consistently across different applications and environments

# AWS Elastic Beanstalk Overview



- Elastic Beanstalk is a developer centric view of deploying an application on AWS
- It uses all the component's we've seen before: EC2, ASG, ELB, RDS, etc...
- But it's all in one view that's easy to make sense of!
- We still have full control over the configuration
  
- Beanstalk = Platform as a Service (PaaS)
- Beanstalk is free but you pay for the underlying instances

# Elastic Beanstalk

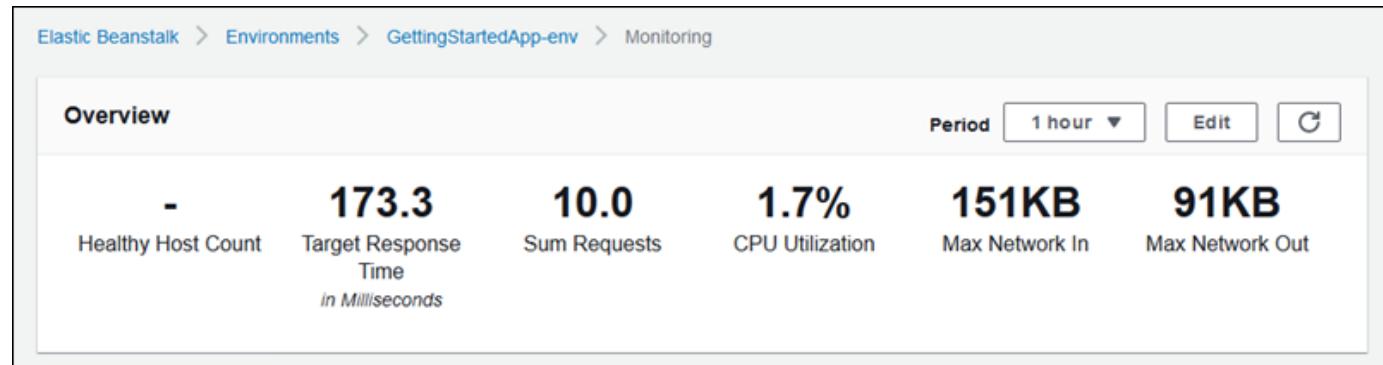
- Managed service
  - Instance configuration / OS is handled by Beanstalk
  - Deployment strategy is configurable but performed by Elastic Beanstalk
  - Capacity provisioning
  - Load balancing & auto-scaling
  - Application health-monitoring & responsiveness
- Just the application code is the responsibility of the developer
- Three architecture models:
  - Single Instance deployment: good for dev
  - LB + ASG: great for production or pre-production web applications
  - ASG only: great for non-web apps in production (workers, etc..)

# Elastic Beanstalk

- Support for many platforms:
  - Go
  - Java SE
  - Java with Tomcat
  - .NET on Windows Server with IIS
  - Node.js
  - PHP
  - Python
  - Ruby
  - Packer Builder
- Single Container Docker
- Multi-Container Docker
- Preconfigured Docker
- If not supported, you can write your custom platform (advanced)

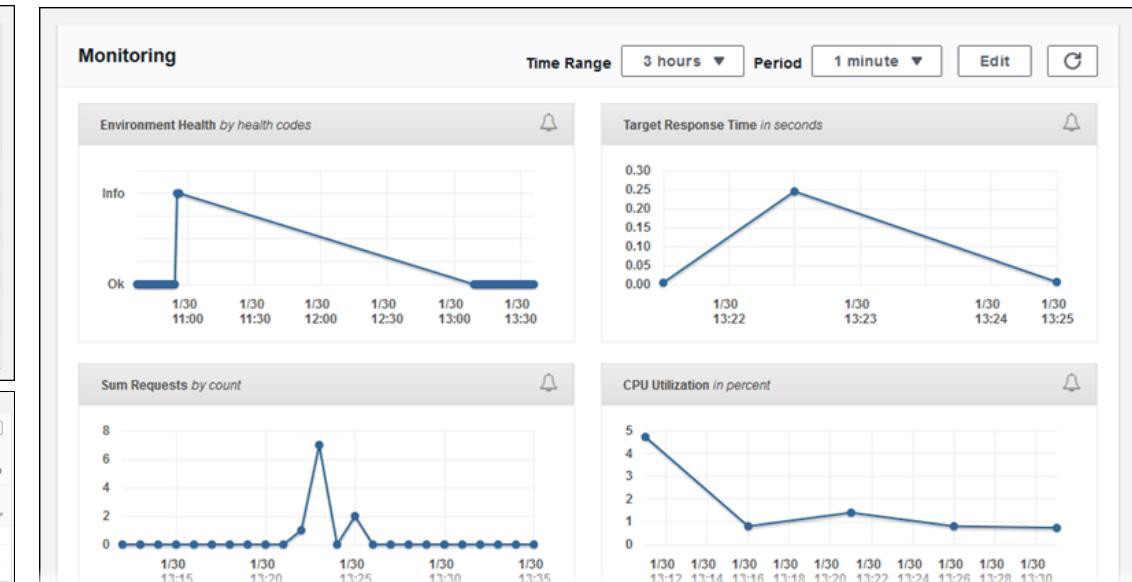
# Elastic Beanstalk – Health Monitoring

- Health agent pushes metrics to CloudWatch
- Checks for app health, publishes health events



**Recent events**

Time	Type	Details
2020-01-28 16:06:04 UTC-0800	INFO	Environment health has transitioned from Severe to Ok.
2020-01-28 16:05:04 UTC-0800	INFO	Added instance [i-03280193ba1ba4171] to your environment.
2020-01-28 16:05:04 UTC-0800	WARN	Removed instance [i-04a427bbb9994ba5] from your environment due to a EC2 health check failure.
2020-01-28 16:03:04 UTC-0800	WARN	Environment health has transitioned from Ok to Severe. ELB processes are not healthy on all instances. None of the instances are sending data. ELB health is failing or not available for all instances.
2020-01-28 15:19:06 UTC-0800	INFO	Environment health has transitioned from Info to Ok. Application update completed 75 seconds ago and took 22 seconds.



**Enhanced health overview**

Instances: 2 Total: 2 Ok

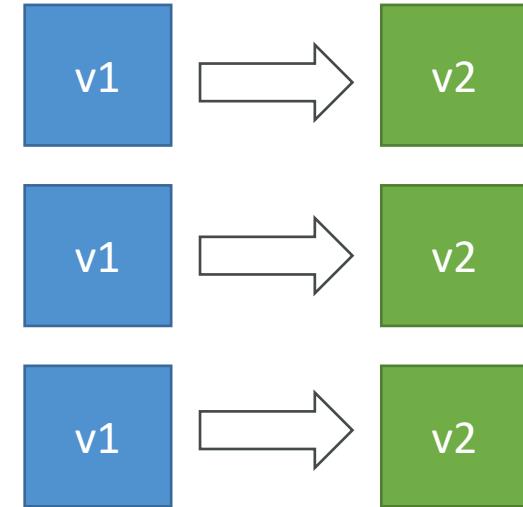
Instance ID	Status	Running	Deployment ID	Requests/sec	2xx Responses	3xx Responses	4xx Responses	5xx Responses	P99 Latency	P90 Latency	P75 Latency	P50 Latency	P10 Latency	Load5 average	Load5 average	CPU utilization User%	CPU utilization Sys%	CPU utilization Idle%	CPU utilization I/O wait%
Overall	Ok	N/A	N/A	0.4	100%	0.0%	0.0%	0.0%	0.002	0.002	0.002	0.001	N/A	N/A	N/A	N/A	N/A	N/A	
i-00227b07c4cfa1334	Ok	2 hours	3	0.2	2	0	0	0	0.002	0.002	0.002	0.002	0.00	0.00	0.00	0.00	99.9	0.0	
i-03280193ba1ba4171	Ok	19 days	3	0.2	2	0	0	0.001	0.001	0.001	0.001	0.00	0.00	0.00	0.00	99.9	0.0		

# AWS CodeDeploy

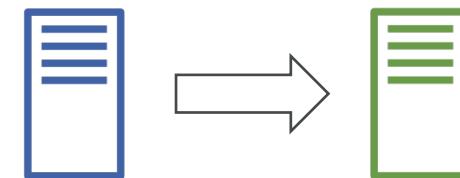


- We want to deploy our application automatically
- Works with EC2 Instances
- Works with On-Premises Servers
- Hybrid service
- Servers / Instances must be provisioned and configured ahead of time with the CodeDeploy Agent

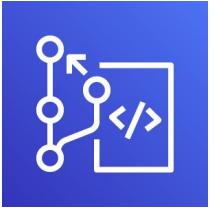
EC2 Instances being upgraded



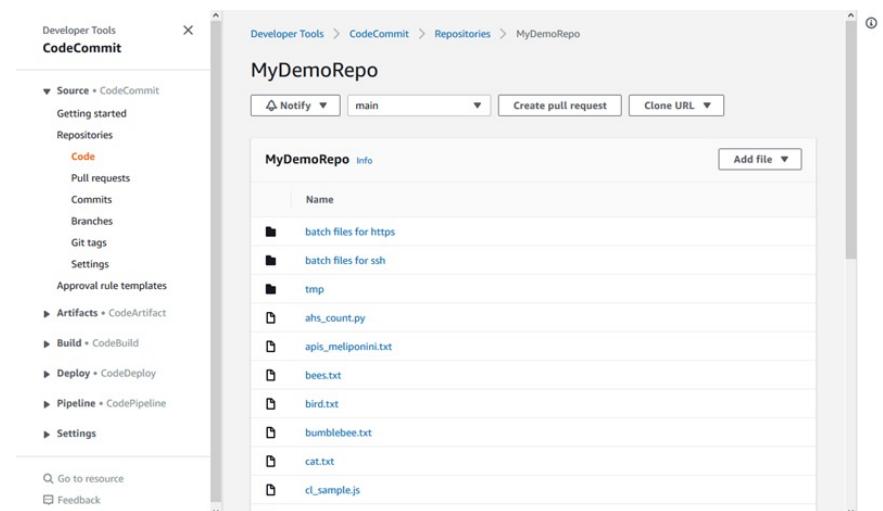
On-premises Servers being upgraded



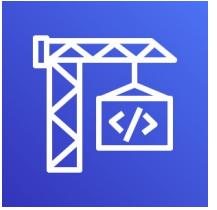
# AWS CodeCommit



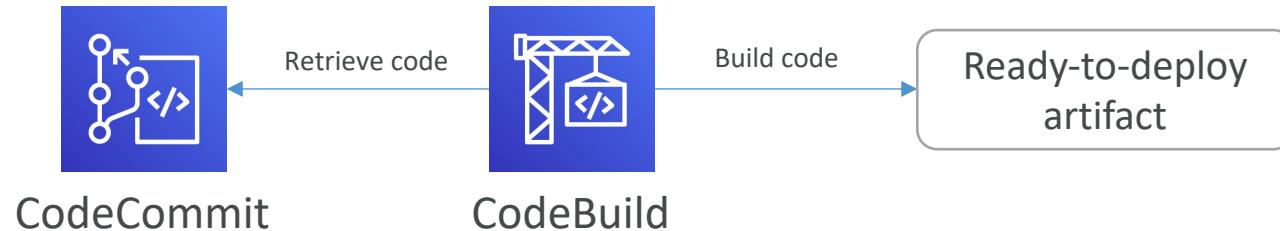
- Before pushing the application code to servers, it needs to be stored somewhere
- Developers usually store **code in a repository, using the Git technology**
- A famous public offering is GitHub, AWS' competing product is **CodeCommit**
- CodeCommit:
  - Source-control service that **hosts Git-based repositories**
  - Makes it easy to **collaborate with others on code**
  - The code changes are automatically **versioned**
- Benefits:
  - Fully managed
  - Scalable & highly available
  - Private, Secured, Integrated with AWS



# AWS CodeBuild



- Code building service in the cloud (name is obvious)
- Compiles source code, run tests, and produces packages that are ready to be deployed (by CodeDeploy for example)

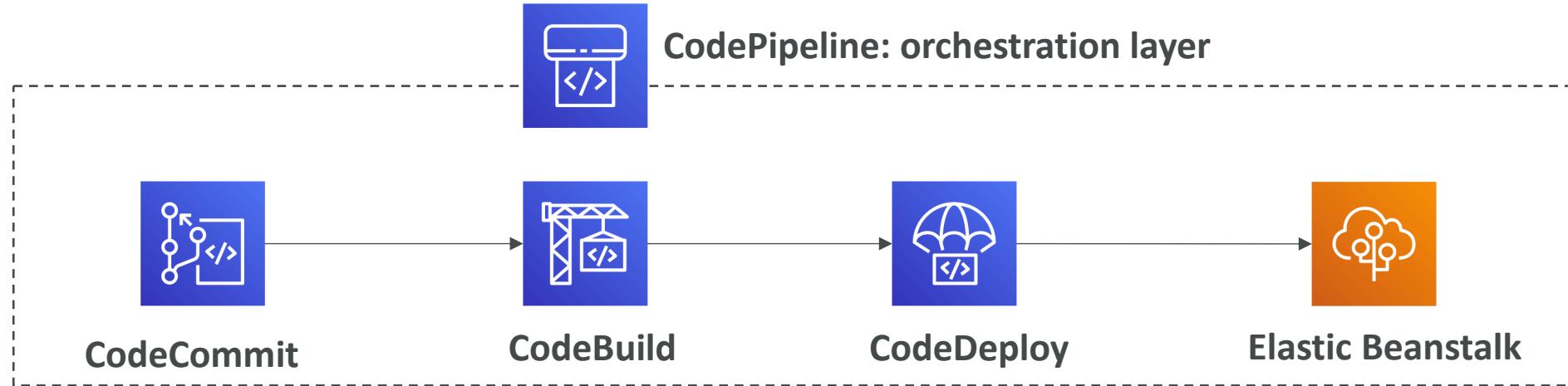


- Benefits:
  - Fully managed, serverless
  - Continuously scalable & highly available
  - Secure
  - Pay-as-you-go pricing – only pay for the build time

# AWS CodePipeline



- Orchestrate the different steps to have the code automatically pushed to production
  - Code => Build => Test => Provision => Deploy
  - Basis for CICD (Continuous Integration & Continuous Delivery)
- Benefits:
  - Fully managed, compatible with CodeCommit, CodeBuild, CodeDeploy, Elastic Beanstalk, CloudFormation, GitHub, 3rd-party services (GitHub...) & custom plugins...
  - Fast delivery & rapid updates

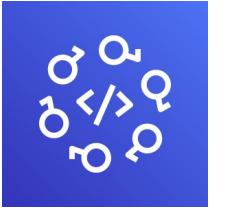


# AWS CodeArtifact



- Software packages depend on each other to be built (also called code dependencies), and new ones are created
- Storing and retrieving these dependencies is called **artifact management**
- Traditionally you need to setup your own artifact management system
- **CodeArtifact** is a secure, scalable, and cost-effective **artifact management** for software development
- Works with common dependency management tools such as Maven, Gradle, npm, yarn, twine, pip, and NuGet
- Developers and CodeBuild can then retrieve dependencies straight from **CodeArtifact**

# AWS CodeStar



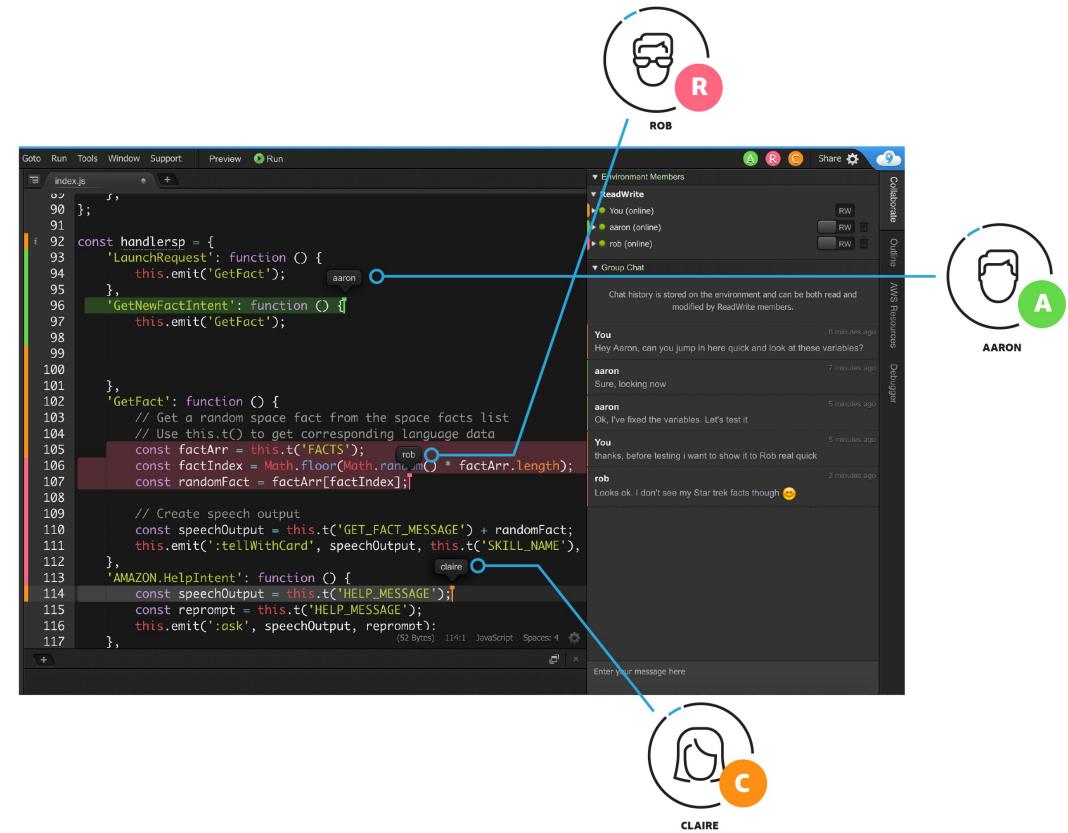
- Unified UI to easily manage software development activities **in one place**
- “Quick way” to get started to correctly set-up CodeCommit, CodePipeline, CodeBuild, CodeDeploy, Elastic Beanstalk, EC2, etc...
- Can edit the code “in-the-cloud” using **AWS Cloud9**

The screenshot shows the AWS CodeStar dashboard for an "EC2 Web Application". The left sidebar includes links for Dashboard, Code, Build, Deploy, Pipeline, Team, and Project. The main area has three main sections: "Application activity" showing CPUUtilization over time with a sharp peak around 22:00; "Continuous deployment" showing a pipeline from "Source" (CodeCommit) through "Build" (CodeBuild) to "Application" (CodeDeploy); and "Commit history" listing recent commits from users JD, HH, TO, T, and J. A "Team wiki tile" section at the bottom allows for sharing markdown notes.

# AWS Cloud9



- AWS Cloud9 is a cloud IDE (Integrated Development Environment) for writing, running and debugging code
- “Classic” IDE (like IntelliJ, Visual Studio Code...) are downloaded on a computer before being used
- A cloud IDE can be used within a web browser, meaning you can work on your projects from your office, home, or anywhere with internet with no setup necessary
- AWS Cloud9 also allows for code collaboration in real-time (pair programming)



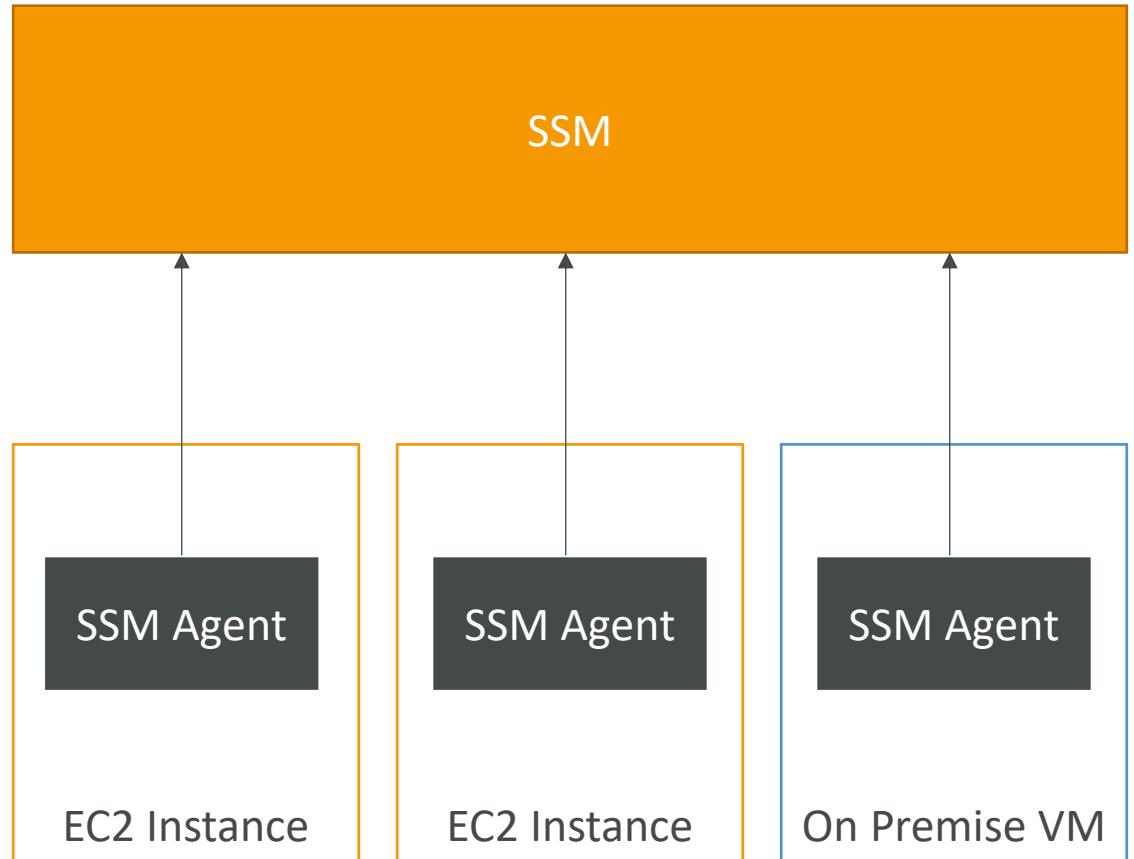
# AWS Systems Manager (SSM)



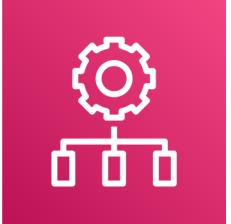
- Helps you manage your **EC2** and **On-Premises** systems at scale
- Another **Hybrid** AWS service
- Get operational insights about the state of your infrastructure
- Suite of 10+ products
- Most important features are:
  - Patching automation for enhanced compliance
  - Run commands across an entire fleet of servers
  - Store parameter configuration with the SSM Parameter Store
- Works for both Windows and Linux OS

# How Systems Manager works

- We need to install the SSM agent onto the systems we control
- Installed by default on Amazon Linux AMI & some Ubuntu AMI
- If an instance can't be controlled with SSM, it's probably an issue with the SSM agent!
- Thanks to the SSM agent, we can **run commands, patch & configure** our servers



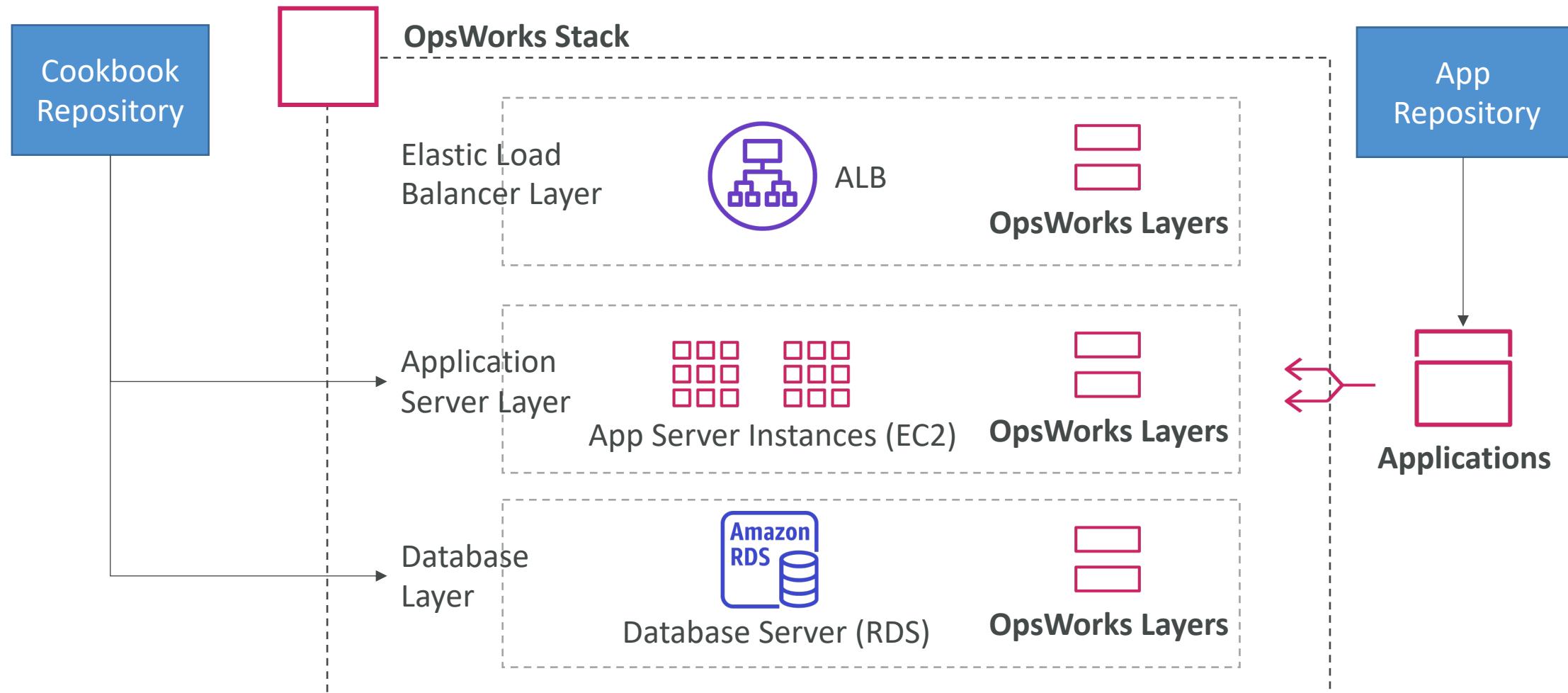
# AWS OpsWorks



- Chef & Puppet help you perform server configuration automatically, or repetitive actions
- They work great with EC2 & On-Premises VM
- AWS OpsWorks = Managed Chef & Puppet
- It's an alternative to AWS SSM
- Only provision standard AWS resources:
  - EC2 Instances, Databases, Load Balancers, EBS volumes...
- In the exam: Chef or Puppet needed => AWS OpsWorks



# OpsWorks Architecture



# Deployment - Summary

- **CloudFormation:** (AWS only)
  - Infrastructure as Code, works with almost all of AWS resources
  - Repeat across Regions & Accounts
- **Beanstalk:** (AWS only)
  - Platform as a Service (PaaS), limited to certain programming languages or Docker
  - Deploy code consistently with a known architecture: ex, ALB + EC2 + RDS
- **CodeDeploy** (hybrid): deploy & upgrade any application onto servers
- **Systems Manager** (hybrid): patch, configure and run commands at scale
- **OpsWorks** (hybrid): managed Chef and Puppet in AWS

# Developer Services - Summary

- **CodeCommit:** Store code in private git repository (version controlled)
- **CodeBuild:** Build & test code in AWS
- **CodeDeploy:** Deploy code onto servers
- **CodePipeline:** Orchestration of pipeline (from code to build to deploy)
- **CodeArtifact:** Store software packages / dependencies on AWS
- **CodeStar:** Unified view for allowing developers to do CI/CD and code
- **Cloud9:** Cloud IDE (Integrated Development Environment) with collab
- **AWS CDK:** Define your cloud infrastructure using a programming language

# Global Infrastructure Section

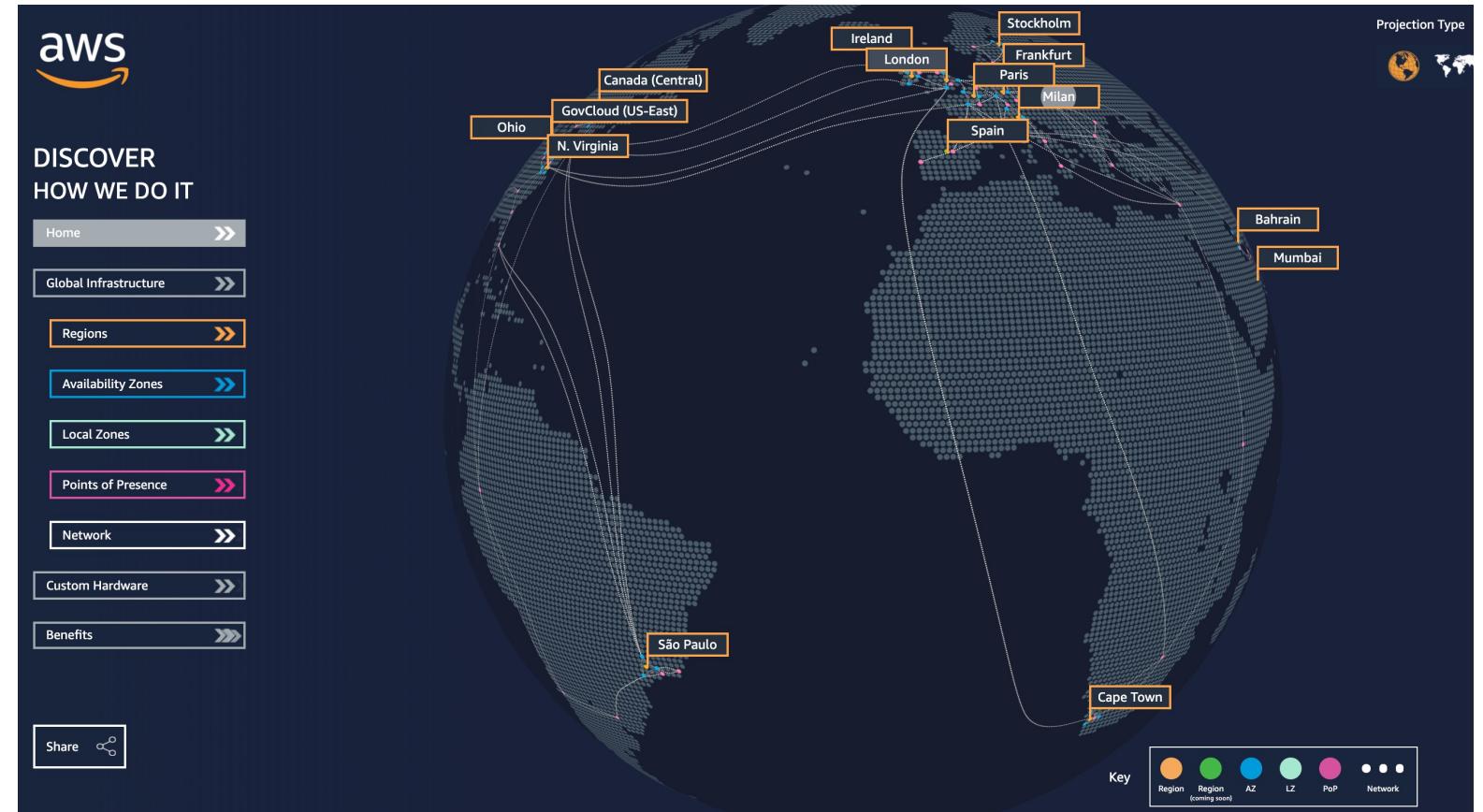
# Why make a global application?



- A **global application** is an application deployed in **multiple geographies**
- On AWS: this could be **Regions** and / or **Edge Locations**
- **Decreased Latency**
  - Latency is the time it takes for a network packet to reach a server
  - It takes time for a packet from Asia to reach the US
  - Deploy your applications closer to your users to decrease latency, better experience
- **Disaster Recovery (DR)**
  - If an AWS region goes down (earthquake, storms, power shutdown, politics)...
  - You can fail-over to another region and have your application still working
  - A DR plan is important to increase the availability of your application
- **Attack protection:** distributed global infrastructure is harder to attack

# Global AWS Infrastructure

- Regions: For deploying applications and infrastructure
- Availability Zones: Made of multiple data centers
- Edge Locations (Points of Presence): for content delivery as close as possible to users
- More at:  
<https://infrastructure.aws/>



# Global Applications in AWS

- **Global DNS: Route 53**
  - Great to route users to the closest deployment with least latency
  - Great for disaster recovery strategies
- **Global Content Delivery Network (CDN): CloudFront**
  - Replicate part of your application to AWS Edge Locations – decrease latency
  - Cache common requests – improved user experience and decreased latency
- **S3 Transfer Acceleration**
  - Accelerate global uploads & downloads into Amazon S3
- **AWS Global Accelerator:**
  - Improve global application availability and performance using the AWS global network

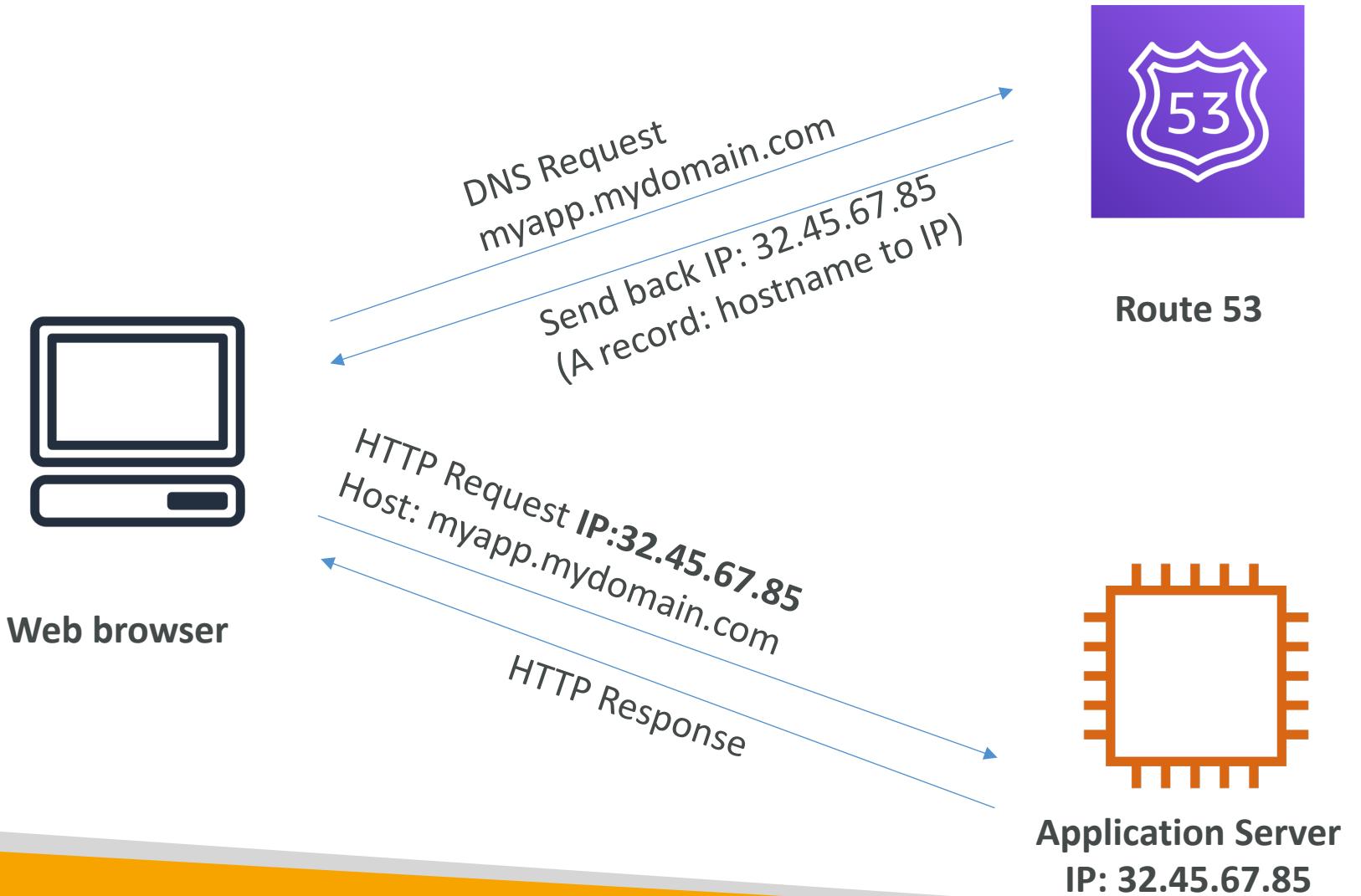


# Amazon Route 53 Overview



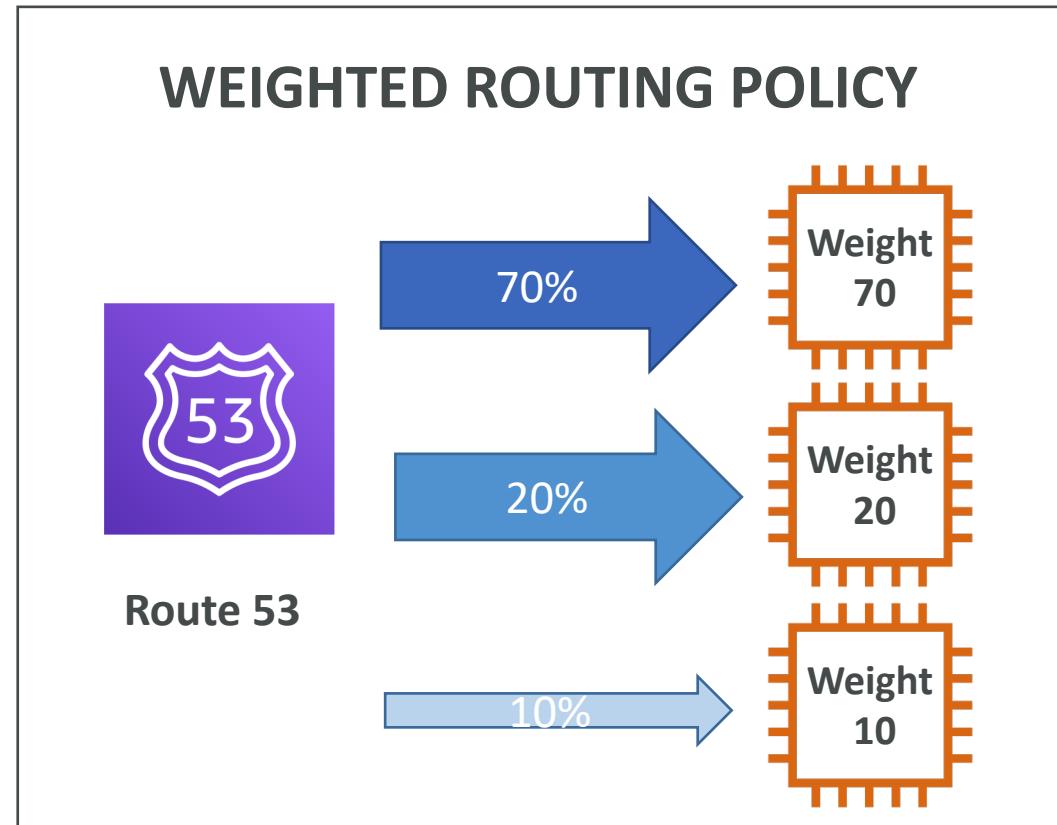
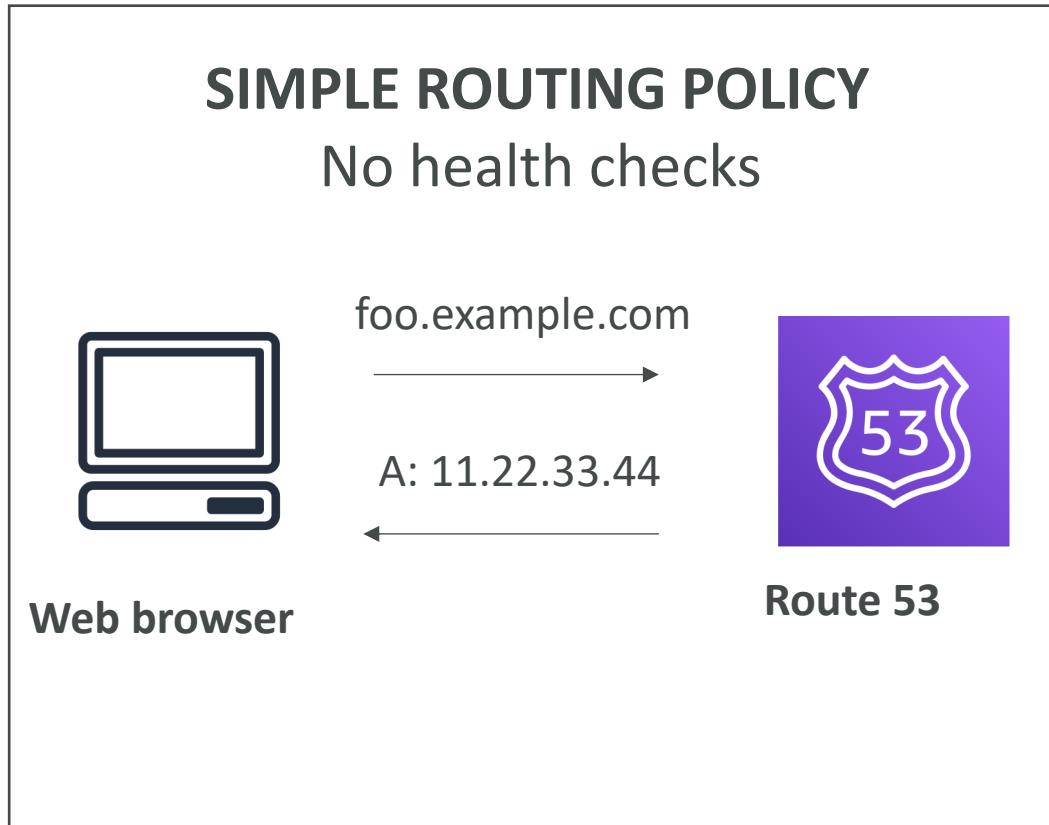
- Route53 is a Managed DNS (Domain Name System)
- DNS is a collection of rules and records which helps clients understand how to reach a server through URLs.
- In AWS, the most common records are:
  - www.google.com => 12.34.56.78 == A record (IPv4)
  - www.google.com => 2001:0db8:85a3:0000:0000:8a2e:0370:7334 == AAAA IPv6
  - search.google.com => www.google.com == CNAME: hostname to hostname
  - example.com => AWS resource == Alias (ex: ELB, CloudFront, S3, RDS, etc...)

# Route 53 – Diagram for A Record



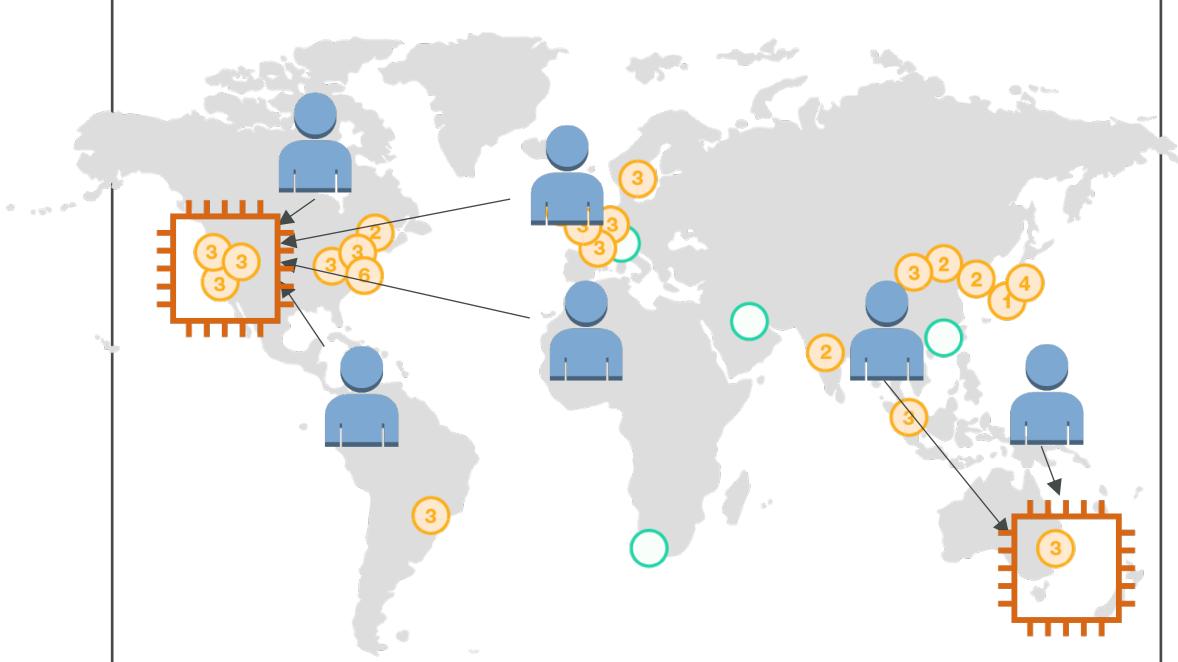
# Route 53 Routing Policies

- Need to know them at a high-level for the Cloud Practitioner Exam



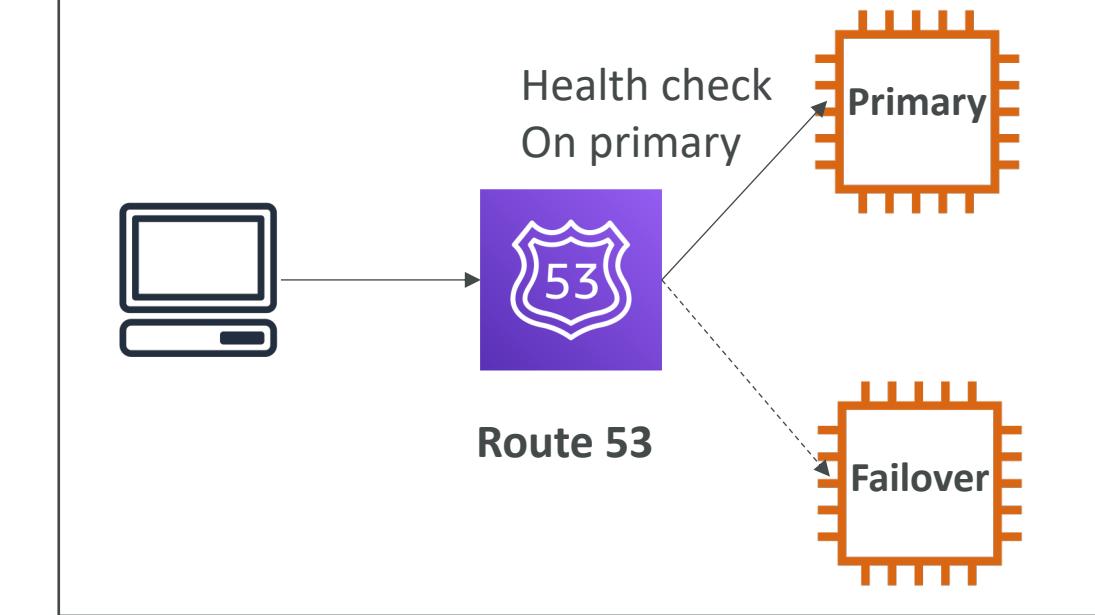
# Route 53 Routing Policies

## LATENCY ROUTING POLICY

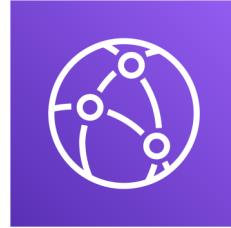


## FAILOVER ROUTING POLICY

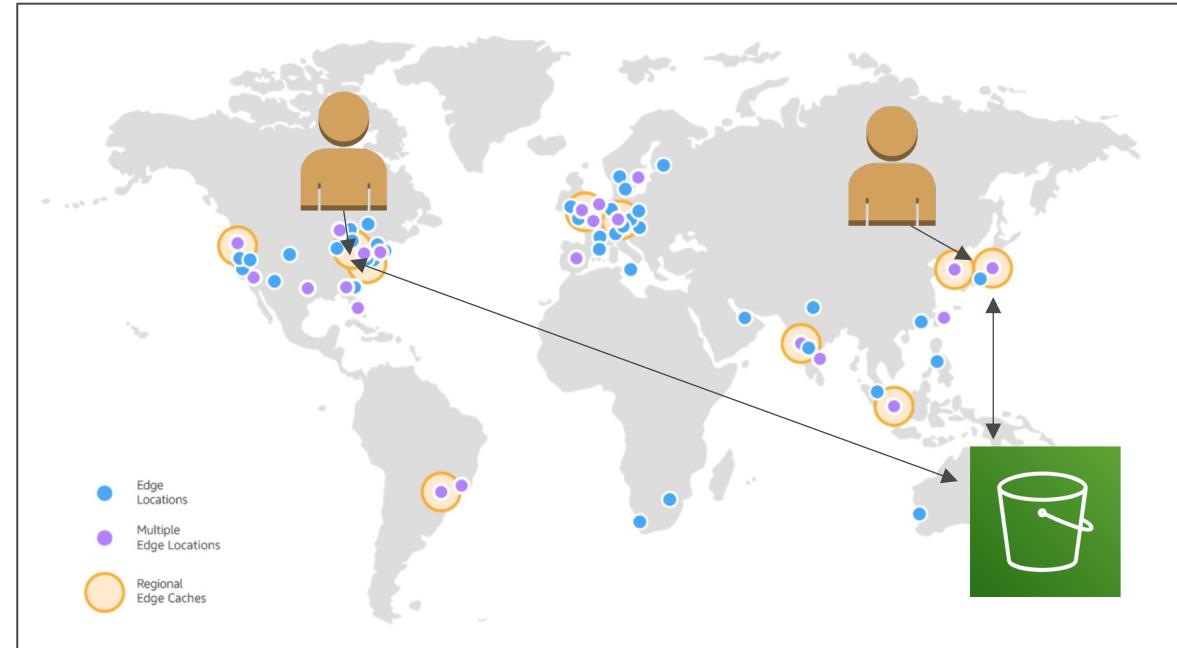
Disaster Recovery



# AWS CloudFront



- Content Delivery Network (CDN)
- Improves read performance, content is cached at the edge
- Improves users experience
- 216 Point of Presence globally (edge locations)
- DDoS protection (because worldwide), integration with Shield, AWS Web Application Firewall

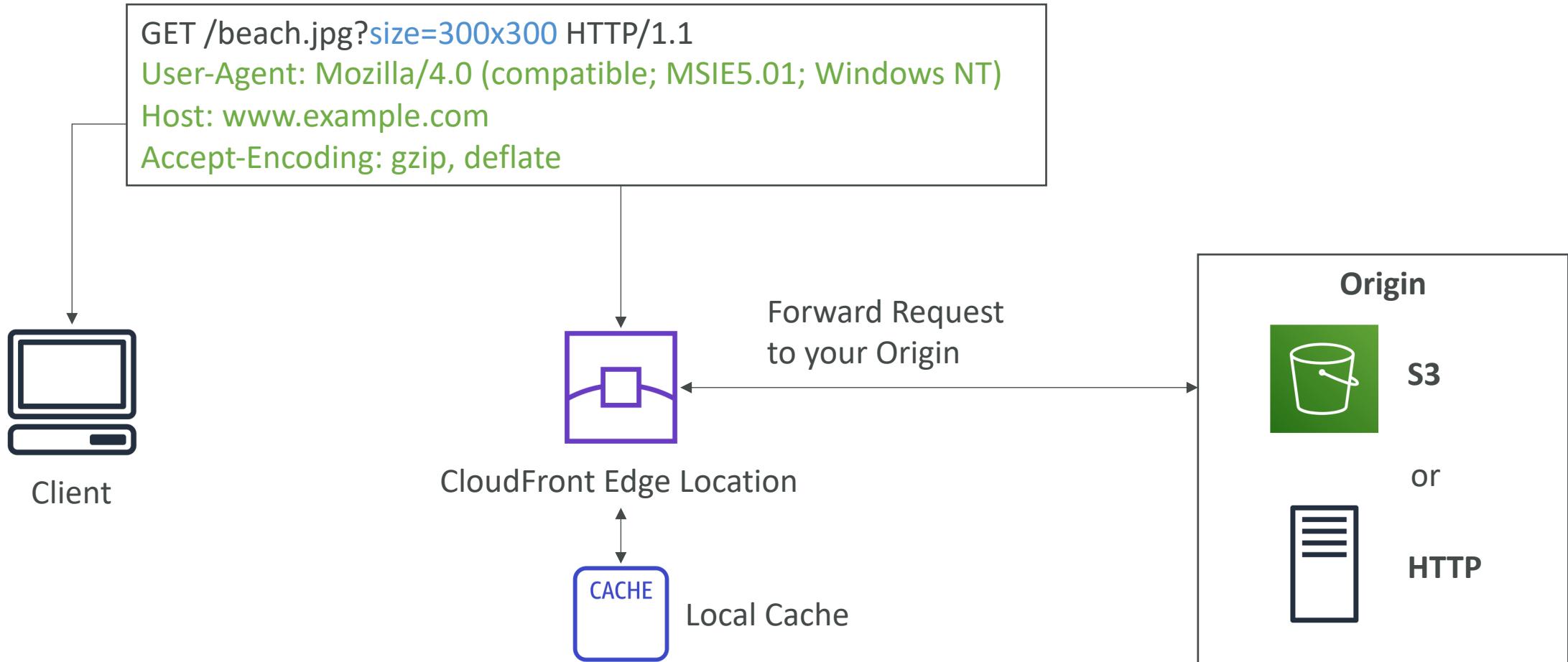


Source: <https://aws.amazon.com/cloudfront/features/?nc=sn&loc=2>

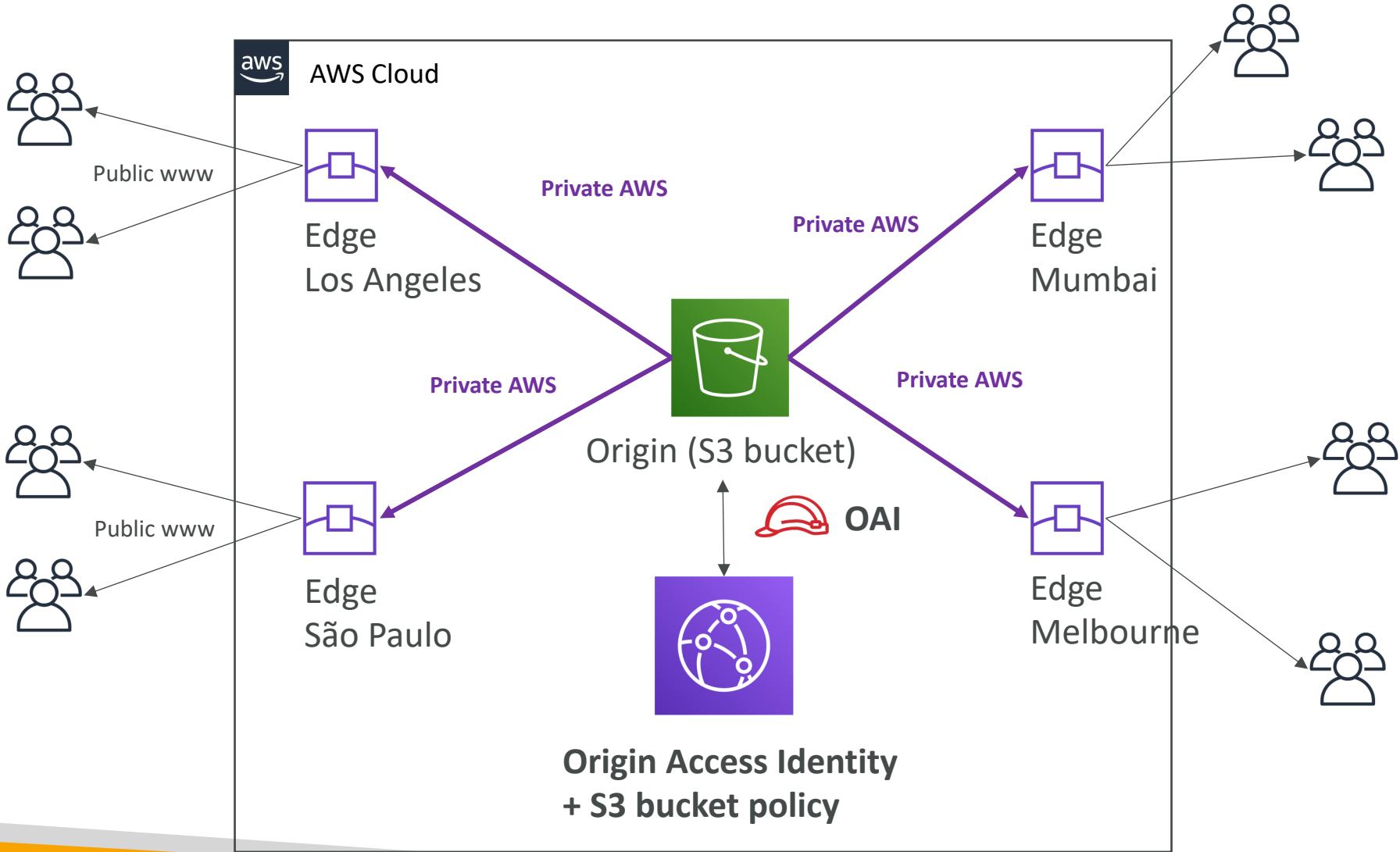
# CloudFront – Origins

- S3 bucket
  - For distributing files and caching them at the edge
  - Enhanced security with CloudFront Origin Access Identity (OAI)
  - CloudFront can be used as an ingress (to upload files to S3)
- Custom Origin (HTTP)
  - Application Load Balancer
  - EC2 instance
  - S3 website (must first enable the bucket as a static S3 website)
  - Any HTTP backend you want

# CloudFront at a high level



# CloudFront – S3 as an Origin

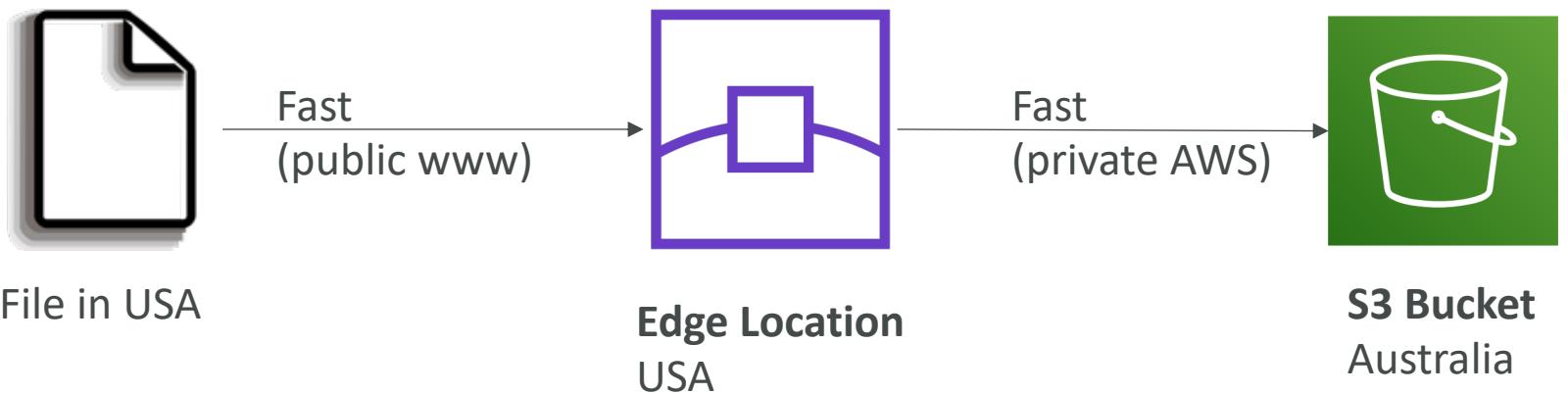


# CloudFront vs S3 Cross Region Replication

- CloudFront:
  - Global Edge network
  - Files are cached for a TTL (maybe a day)
  - Great for static content that must be available everywhere
- S3 Cross Region Replication:
  - Must be setup for each region you want replication to happen
  - Files are updated in near real-time
  - Read only
  - Great for dynamic content that needs to be available at low-latency in few regions

# S3 Transfer Acceleration

- Increase transfer speed by transferring file to an AWS edge location which will forward the data to the S3 bucket in the target region

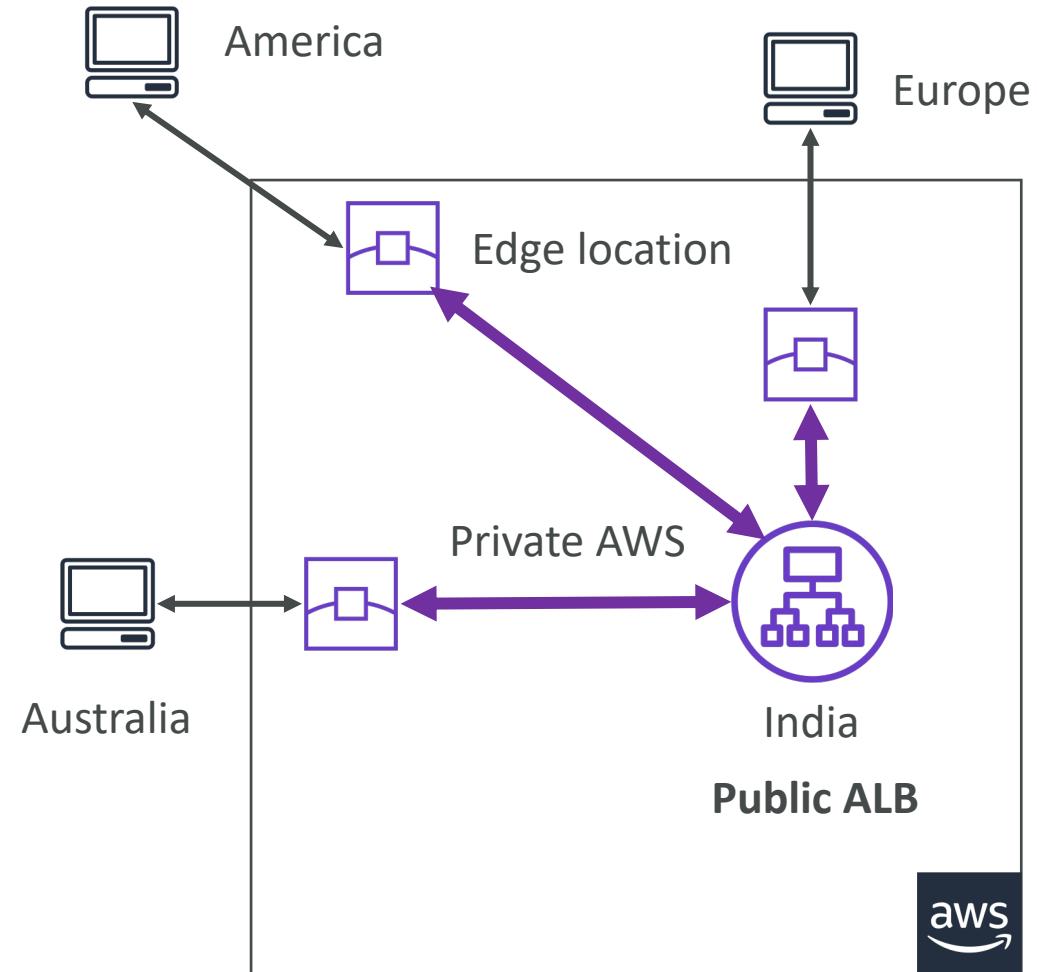


Test the tool at: <https://s3-accelerate-speedtest.s3-accelerate.amazonaws.com/en/accelerate-speed-comparison.html>

# AWS Global Accelerator

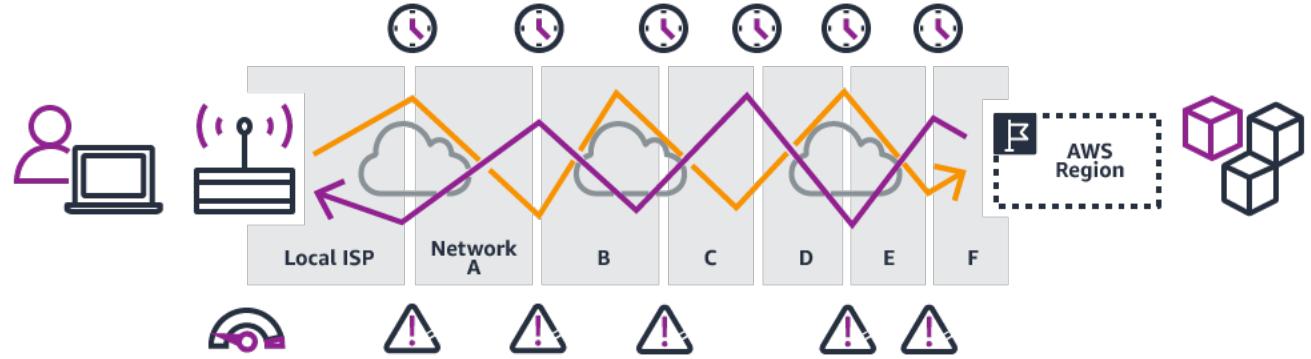


- Improve global application availability and performance using the AWS global network
- Leverage the AWS internal network to optimize the route to your application (60% improvement)
- 2 Anycast IP are created for your application and traffic is sent through **Edge Locations**
- The Edge locations send the traffic to your application

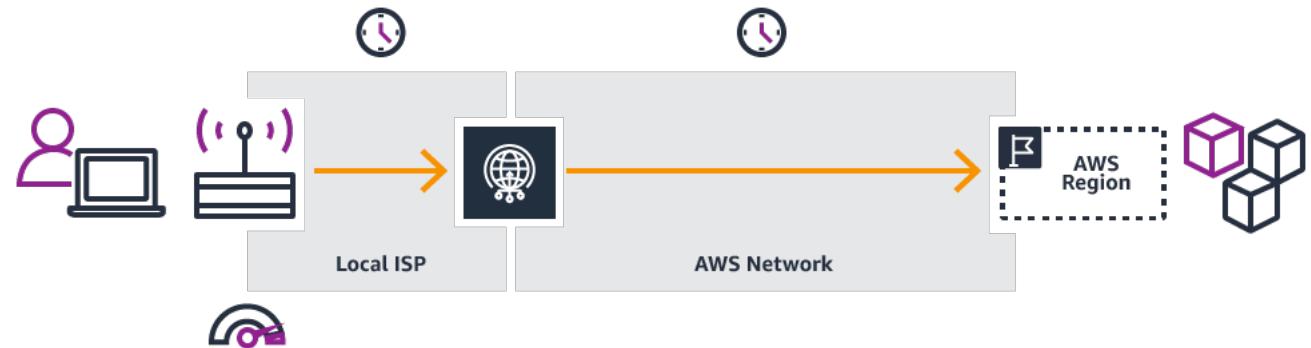


# AWS Global Accelerator

Without Global Accelerator



With Global Accelerator

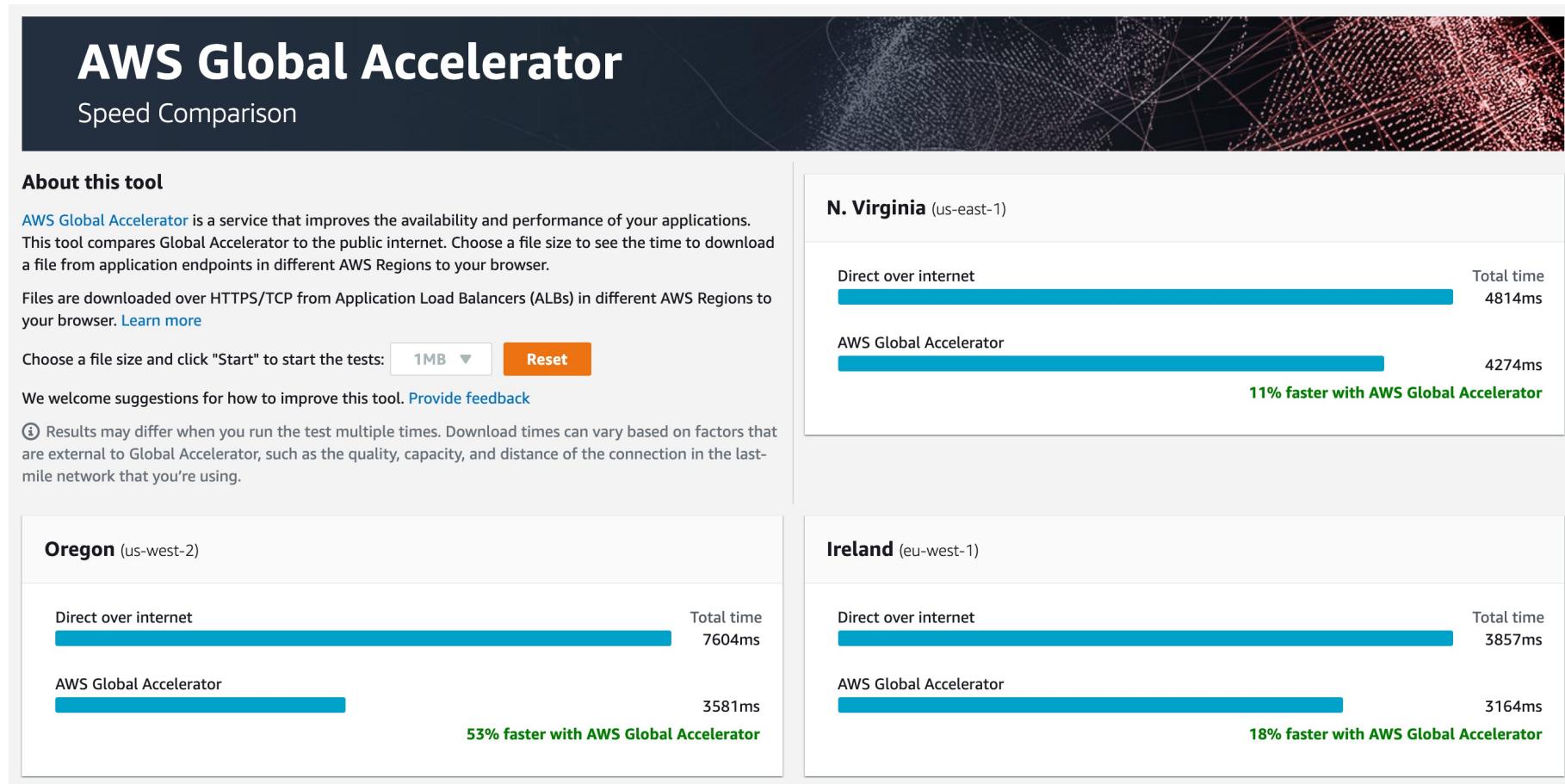


<https://aws.amazon.com/global-accelerator>

# AWS Global Accelerator vs CloudFront

- They both use the AWS global network and its edge locations around the world
- Both services integrate with AWS Shield for DDoS protection.
- **CloudFront – Content Delivery Network**
  - Improves performance for your cacheable content (such as images and videos)
  - Content is served at the edge
- **Global Accelerator**
  - No caching, proxying packets at the edge to applications running in one or more AWS Regions.
  - Improves performance for a wide range of applications over TCP or UDP
  - Good for HTTP use cases that require static IP addresses
  - Good for HTTP use cases that required deterministic, fast regional failover

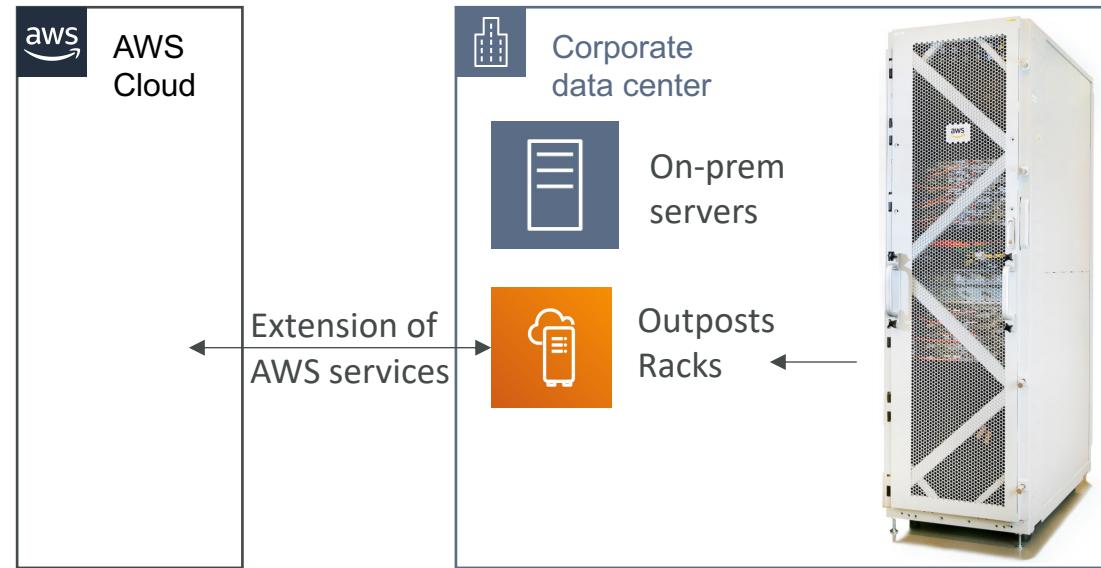
<https://speedtest.globalaccelerator.aws/#/>



# AWS Outposts



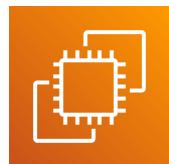
- **Hybrid Cloud:** businesses that keep an on-premises infrastructure alongside a cloud infrastructure
- Therefore, two ways of dealing with IT systems:
  - One for the AWS cloud (using the AWS console, CLI, and AWS APIs)
  - One for their on-premises infrastructure
- **AWS Outposts** are “server racks” that offers the same AWS infrastructure, services, APIs & tools to build your own applications on-premises just as in the cloud
- **AWS will setup and manage “Outposts Racks”** within your on-premises infrastructure and you can start leveraging AWS services on-premises
- **You are responsible for the Outposts Rack physical security**



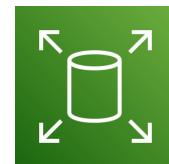
# AWS Outposts



- Benefits:
  - Low-latency access to on-premises systems
  - Local data processing
  - Data residency
  - Easier migration from on-premises to the cloud
  - Fully managed service
- Some services that work on Outposts:



Amazon EC2



Amazon EBS



Amazon S3



Amazon EKS



Amazon ECS



Amazon RDS

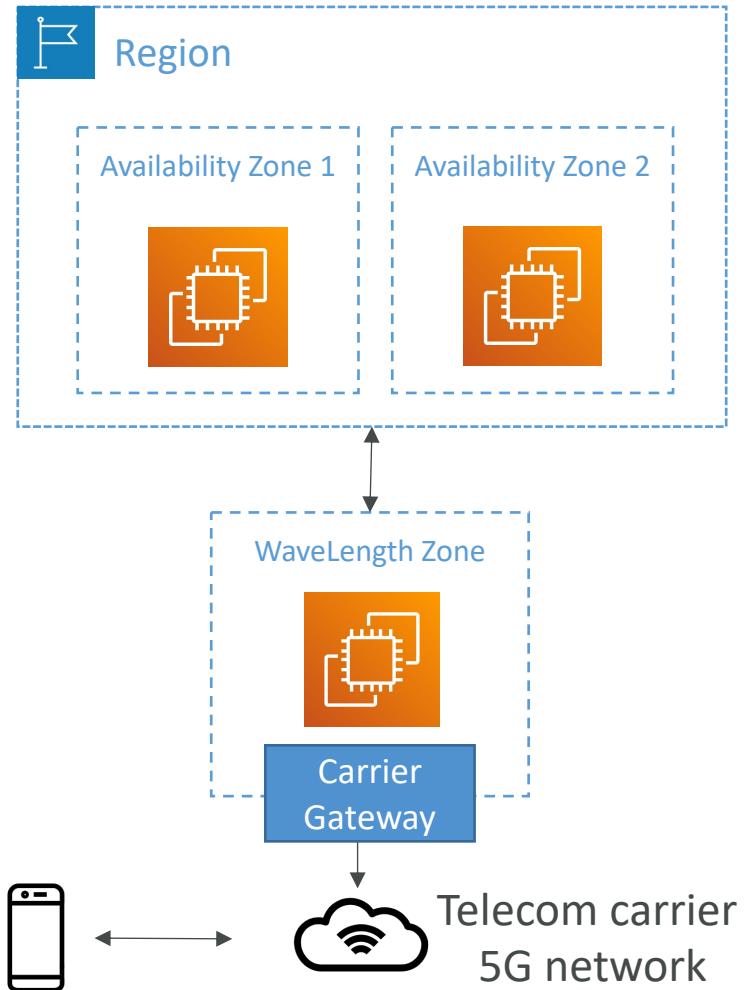


Amazon EMR

# AWS WaveLength



- **WaveLength Zones** are infrastructure deployments embedded within the telecommunications providers' datacenters at the edge of the 5G networks
- Brings AWS services to the edge of the 5G networks
- Example: EC2, EBS, VPC...
- Ultra-low latency applications through 5G networks
- Traffic doesn't leave the Communication Service Provider's (CSP) network
- High-bandwidth and secure connection to the parent AWS Region
- No additional charges or service agreements
- Use cases: Smart Cities, ML-assisted diagnostics, Connected Vehicles, Interactive Live Video Streams, AR/VR, Real-time Gaming, ...



# Global Applications in AWS - Summary



- **Global DNS: Route 53**
  - Great to route users to the closest deployment with least latency
  - Great for disaster recovery strategies



- **Global Content Delivery Network (CDN): CloudFront**
  - Replicate part of your application to AWS Edge Locations – decrease latency
  - Cache common requests – improved user experience and decreased latency



- **S3 Transfer Acceleration**
  - Accelerate global uploads & downloads into Amazon S3



- **AWS Global Accelerator:**
  - Improve global application availability and performance using the AWS global network



- **AWS Outposts:**
  - Deploy Outposts Racks in your own Data Centers to extend AWS services



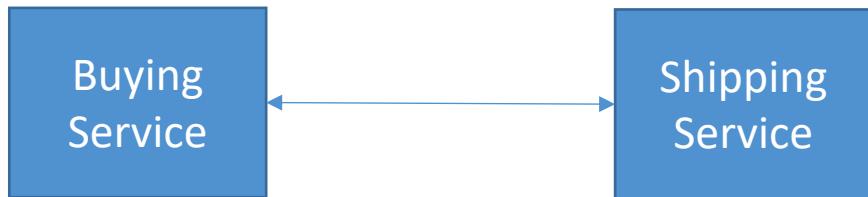
- **AWS WaveLength:**
  - Brings AWS services to the edge of the 5G networks
  - Ultra-low latency applications

# Cloud Integration Section

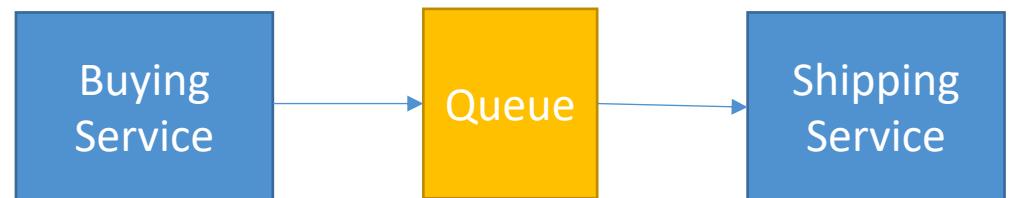
# Section Introduction

- When we start deploying multiple applications, they will inevitably need to communicate with one another
- There are two patterns of application communication

**1) Synchronous communications  
(application to application)**



**2) Asynchronous / Event based  
(application to queue to application)**

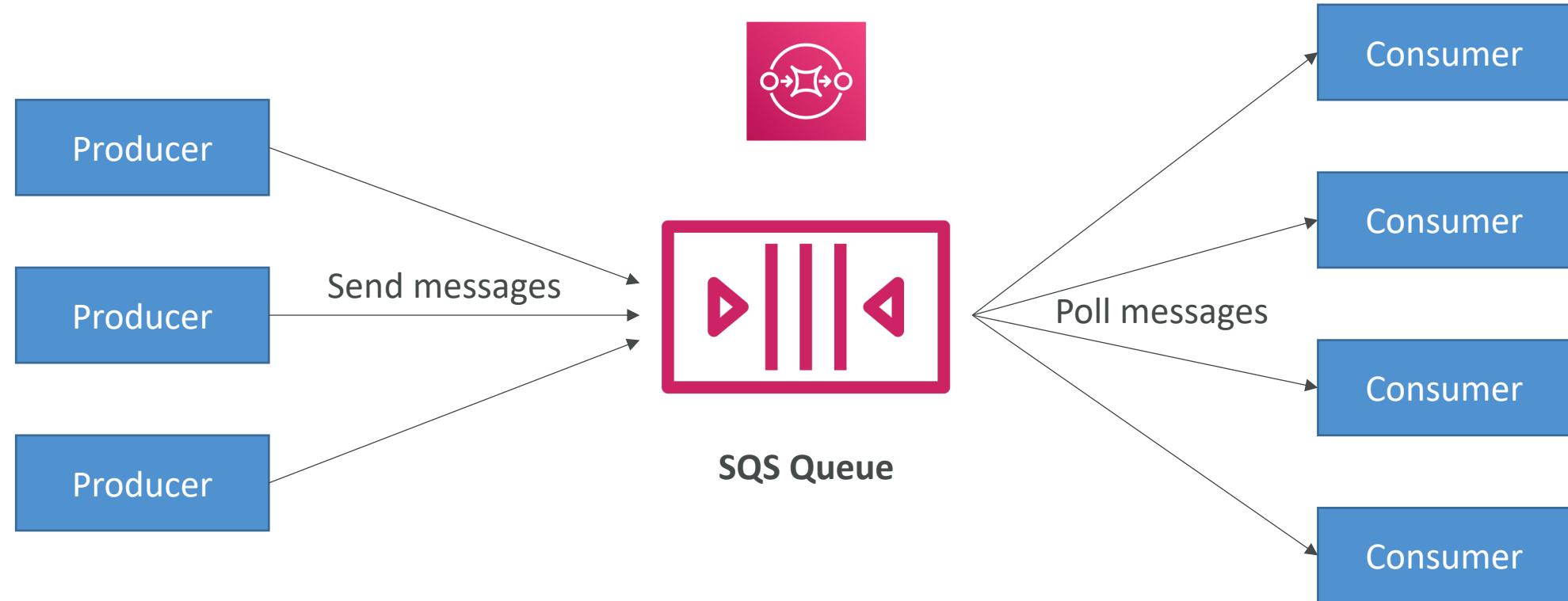


# Section Introduction

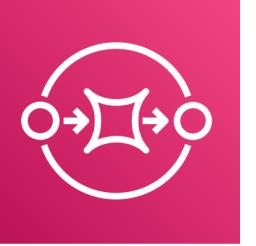
- Synchronous between applications can be problematic if there are sudden spikes of traffic
- What if you need to suddenly encode 1000 videos but usually it's 10?
- In that case, it's better to **decouple** your applications:
  - using SQS: queue model
  - using SNS: pub/sub model
  - using Kinesis: real-time data streaming model (out of scope for the exam)
- These services can scale independently from our application!

# Amazon SQS – Simple Queue Service

## What's a queue?

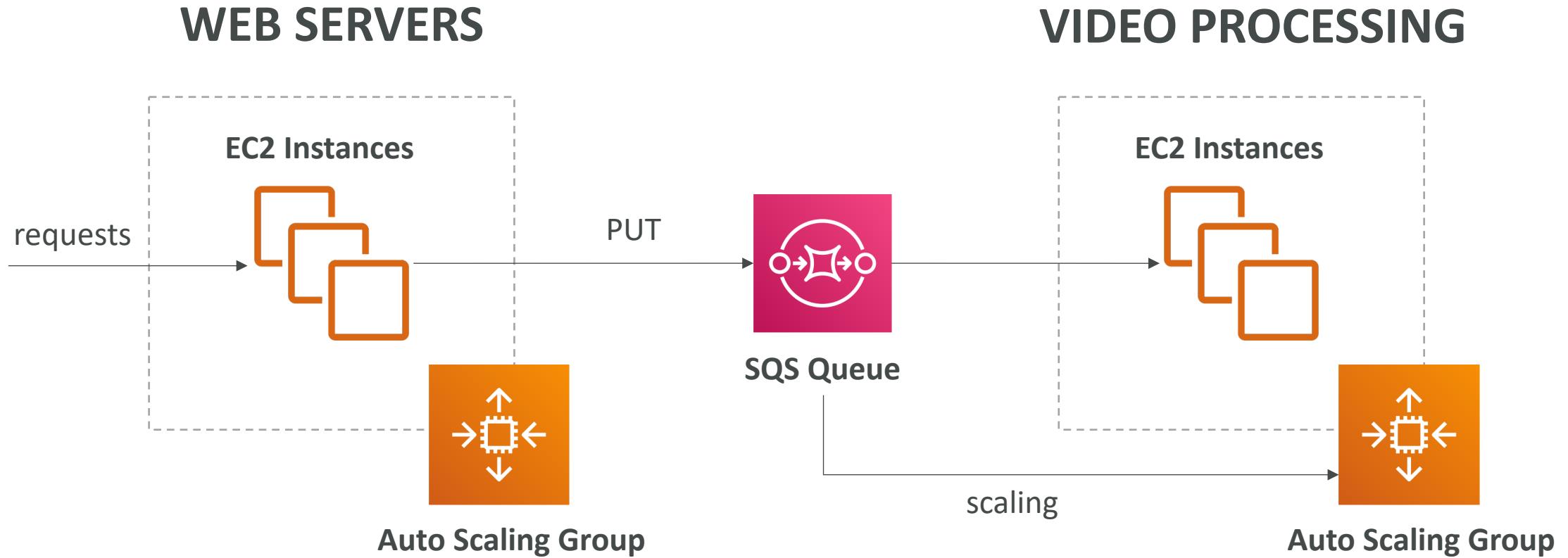


# Amazon SQS – Standard Queue



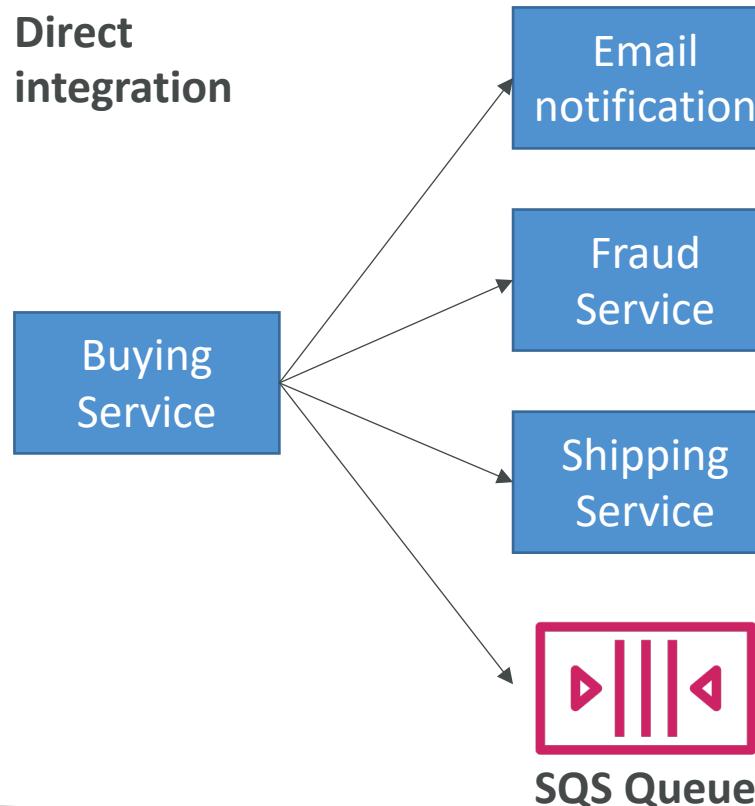
- Oldest AWS offering (over 10 years old)
- Fully managed service (~serverless), use to **decouple** applications
- Scales from 1 message per second to 10,000s per second
- Default retention of messages: 4 days, maximum of 14 days
- No limit to how many messages can be in the queue
- **Messages are deleted after they're read by consumers**
- Low latency (<10 ms on publish and receive)
- Consumers share the work to read messages & scale horizontally

# SQS to decouple between application tiers

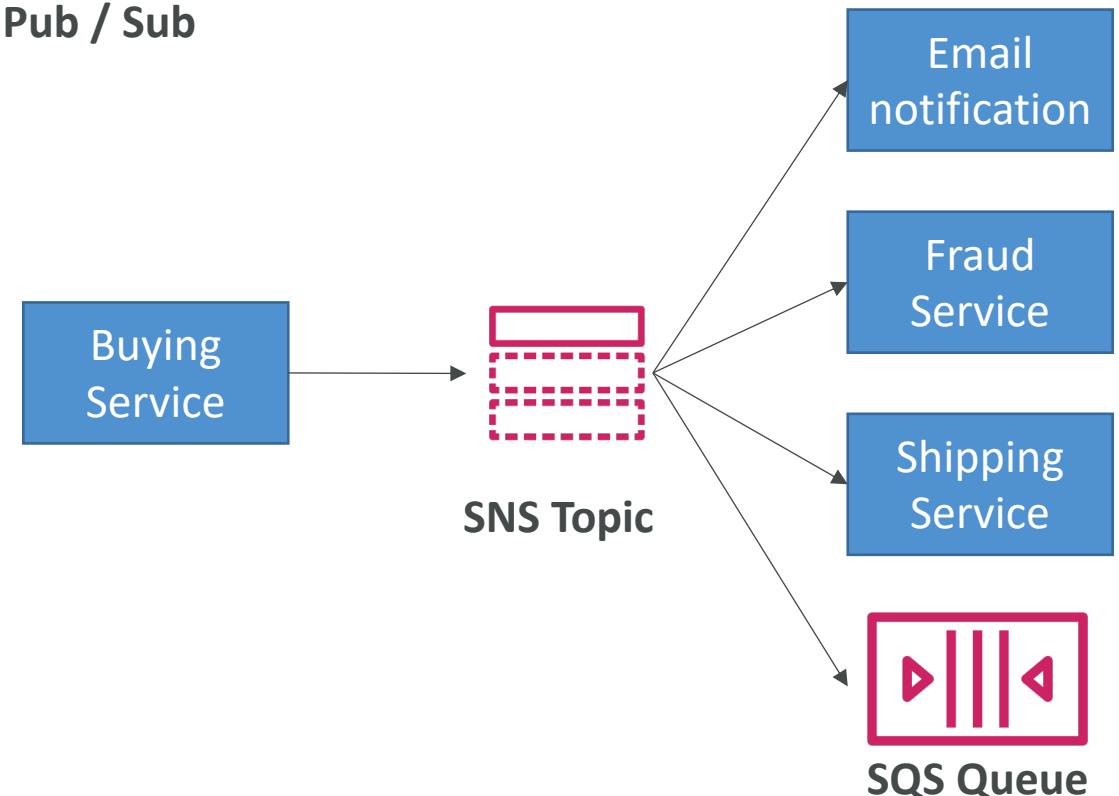


# Amazon SNS

- What if you want to send one message to many receivers?



**Pub / Sub**



# Amazon SNS



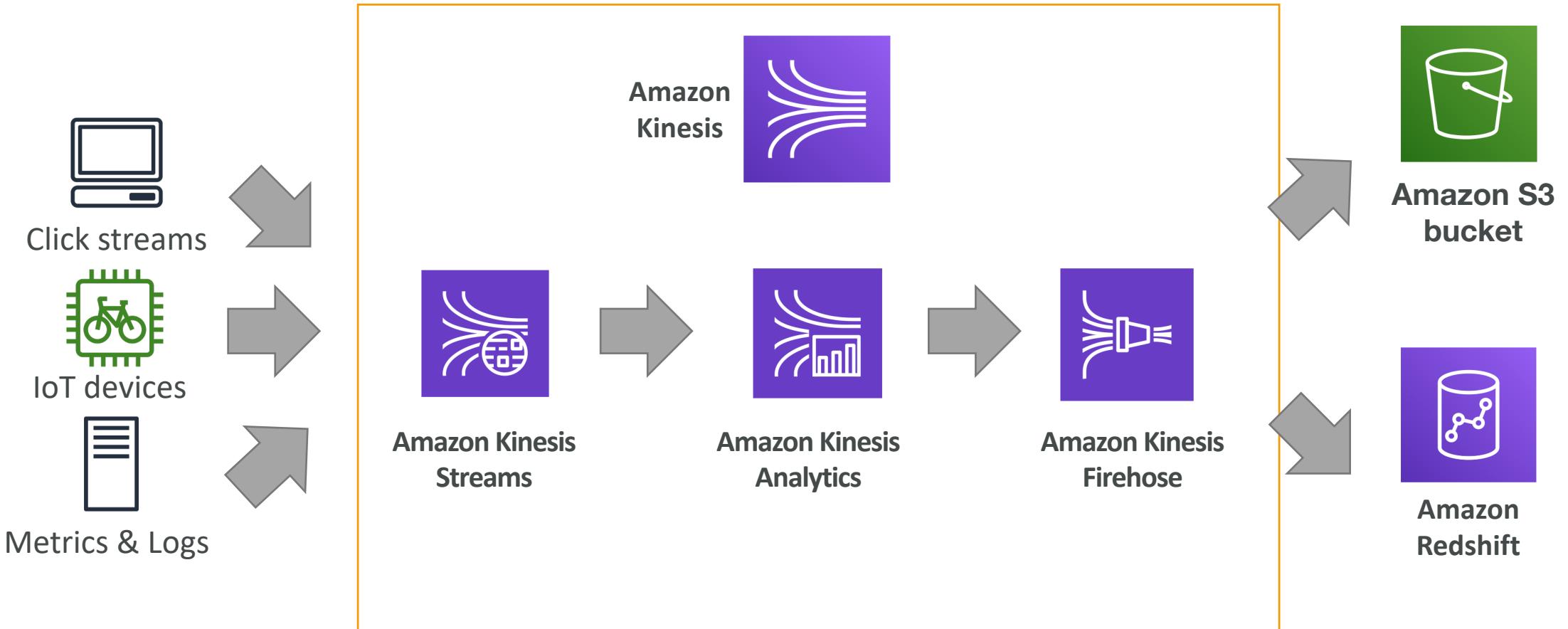
- The “event publishers” only sends message to one SNS topic
- As many “event subscribers” as we want to listen to the SNS topic notifications
- Each subscriber to the topic **will get all the messages**
- Up to 10,000,000 subscriptions per topic, 100,000 topics limit
- SNS Subscribers can be:
  - HTTP / HTTPS (with delivery retries – how many times)
  - Emails, SMS messages, Mobile Notifications
  - SQS queues (fan-out pattern), Lambda Functions (write-your-own integration)

# Amazon Kinesis

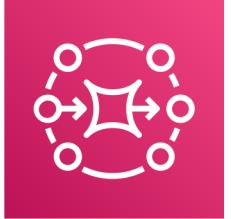


- For the exam: Kinesis = real-time big data streaming
- Managed service to collect, process, and analyze real-time streaming data at any scale
- Too detailed for the Cloud Practitioner exam but good to know:
  - **Kinesis Data Streams:** low latency streaming to ingest data at scale from hundreds of thousands of sources
  - **Kinesis Data Firehose:** load streams into S3, Redshift, ElasticSearch, etc...
  - **Kinesis Data Analytics:** perform real-time analytics on streams using SQL
  - **Kinesis Video Streams:** monitor real-time video streams for analytics or ML

# Kinesis (high level overview)



# Amazon MQ



- SQS, SNS are “cloud-native” services, and they’re using proprietary protocols from AWS.
  - Traditional applications running from on-premise may use open protocols such as: MQTT, AMQP, STOMP, Openwire, WSS
  - When migrating to the cloud, instead of re-engineering the application to use SQS and SNS, we can use Amazon MQ
  - Amazon MQ = managed Apache ActiveMQ
- 
- Amazon MQ doesn’t “scale” as much as SQS / SNS
  - Amazon MQ runs on a dedicated machine (not serverless)
  - Amazon MQ has both queue feature (~SQS) and topic features (~SNS)

# Integration Section – Summary

- **SQS:**
  - Queue service in AWS
  - Multiple Producers, messages are kept up to 14 days
  - Multiple Consumers share the read and delete messages when done
  - Used to **decouple** applications in AWS
- **SNS:**
  - Notification service in AWS
  - Subscribers: Email, Lambda, SQS, HTTP, Mobile...
  - Multiple Subscribers, send all messages to all of them
  - No message retention
- **Kinesis:** real-time data streaming, persistence and analysis
- **Amazon MQ:** managed Apache MQ in the cloud (MQTT, AMQP.. protocols)

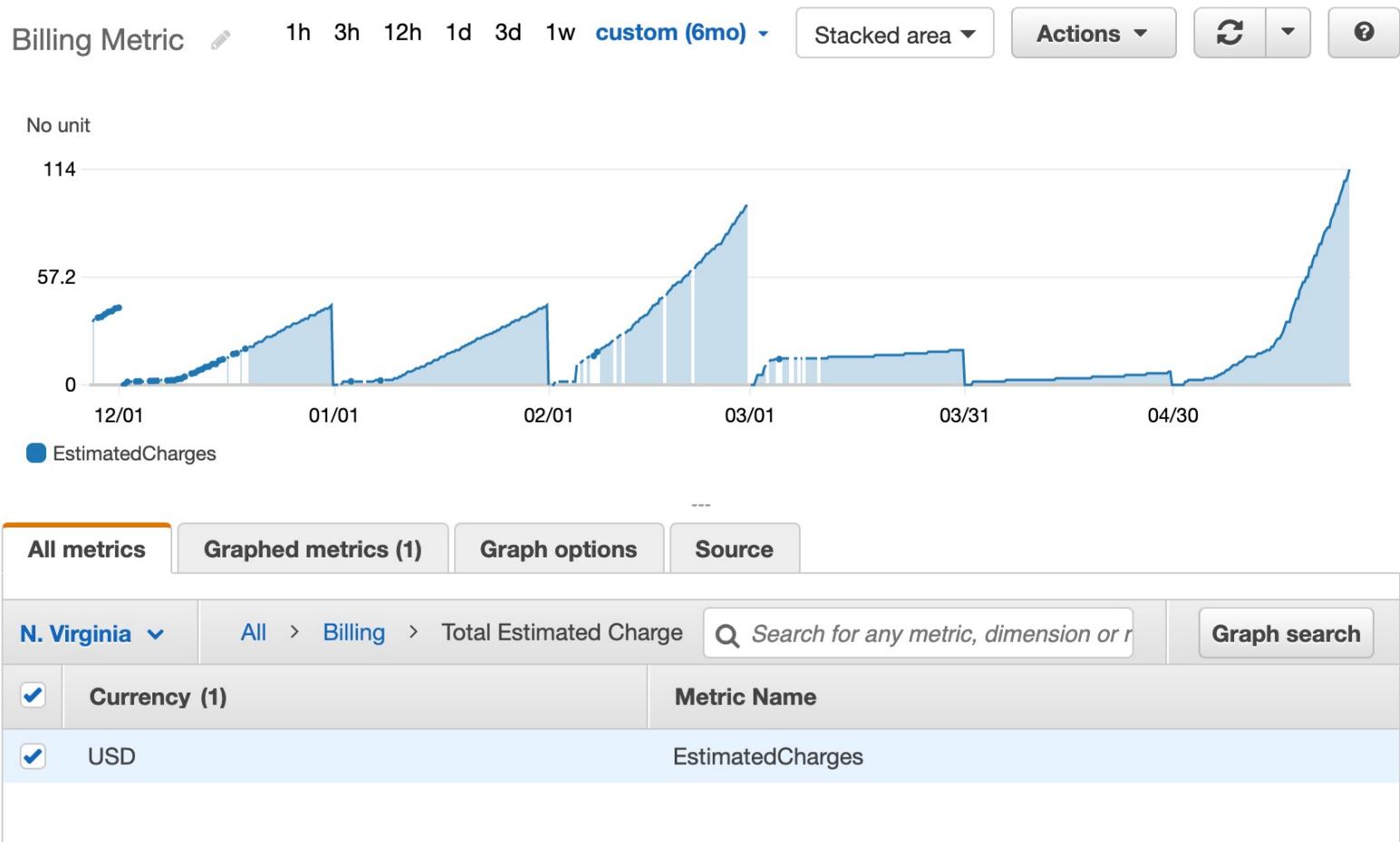
# Cloud Monitoring Section

# Amazon CloudWatch Metrics



- CloudWatch provides metrics for every services in AWS
- **Metric** is a variable to monitor (CPUUtilization, NetworkIn...)
- Metrics have **timestamps**
- Can create **CloudWatch dashboards** of metrics

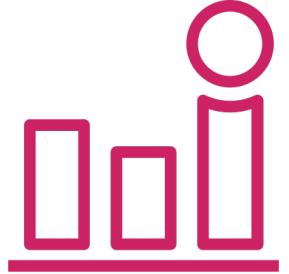
# Example: CloudWatch Billing metric (us-east-1)



# Important Metrics

- **EC2 instances:** CPU Utilization, Status Checks, Network (not RAM)
  - Default metrics every 5 minutes
  - Option for Detailed Monitoring (\$\$\$): metrics every 1 minute
- **EBS volumes:** Disk Read/Writes
- **S3 buckets:** BucketSizeBytes, NumberOfObjects, AllRequests
- **Billing:** Total Estimated Charge (only in us-east-1)
- **Service Limits:** how much you've been using a service API
- **Custom metrics:** push your own metrics

# Amazon CloudWatch Alarms



- Alarms are used to trigger notifications for any metric
- Alarms actions...
  - Auto Scaling: increase or decrease EC2 instances “desired” count
  - EC2 Actions: stop, terminate, reboot or recover an EC2 instance
  - SNS notifications: send a notification into an SNS topic
- Various options (sampling, %, max, min, etc...)
- Can choose the period on which to evaluate an alarm
- Example: create a billing alarm on the CloudWatch Billing metric
- Alarm States: OK, INSUFFICIENT\_DATA, ALARM

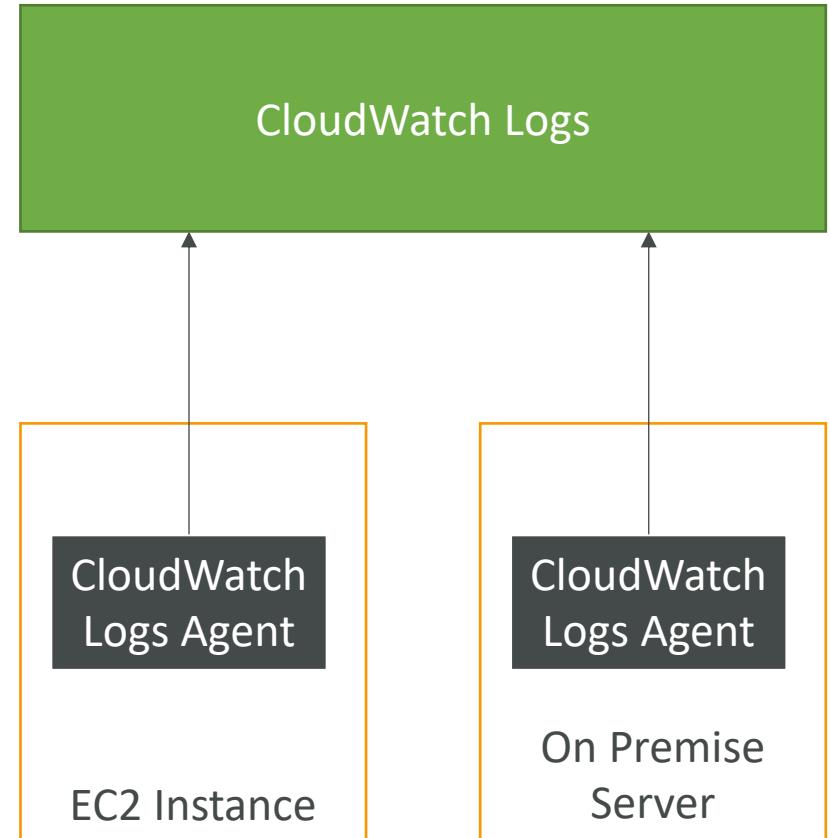


# Amazon CloudWatch Logs

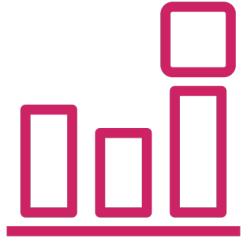
- CloudWatch Logs can collect log from:
  - Elastic Beanstalk: collection of logs from application
  - ECS: collection from containers
  - AWS Lambda: collection from function logs
  - CloudTrail based on filter
  - CloudWatch log agents: on EC2 machines or on-premises servers
  - Route53: Log DNS queries
- Enables **real-time monitoring** of logs
- Adjustable CloudWatch Logs retention

# CloudWatch Logs for EC2

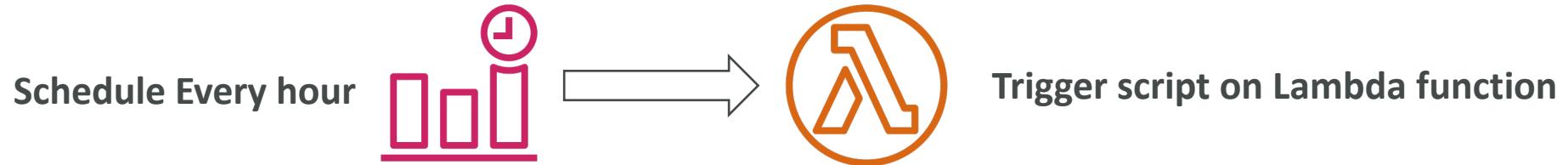
- By default, no logs from your EC2 instance will go to CloudWatch
- You need to run a CloudWatch agent on EC2 to push the log files you want
- Make sure IAM permissions are correct
- The CloudWatch log agent can be setup on-premises too



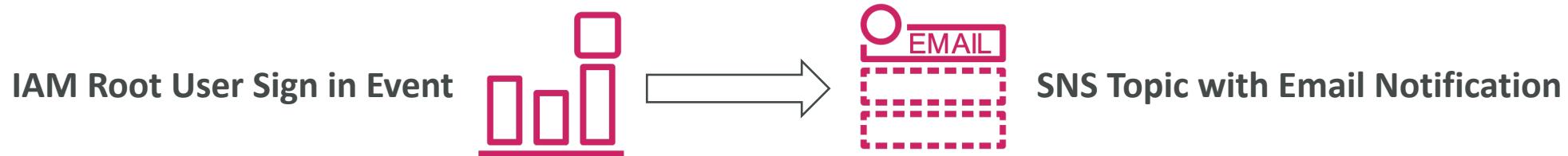
# Amazon CloudWatch Events



- Schedule: Cron jobs (scheduled scripts)



- Event Pattern: Event rules to react to a service doing something



- Trigger Lambda functions, send SQS/SNS messages...

# Amazon EventBridge



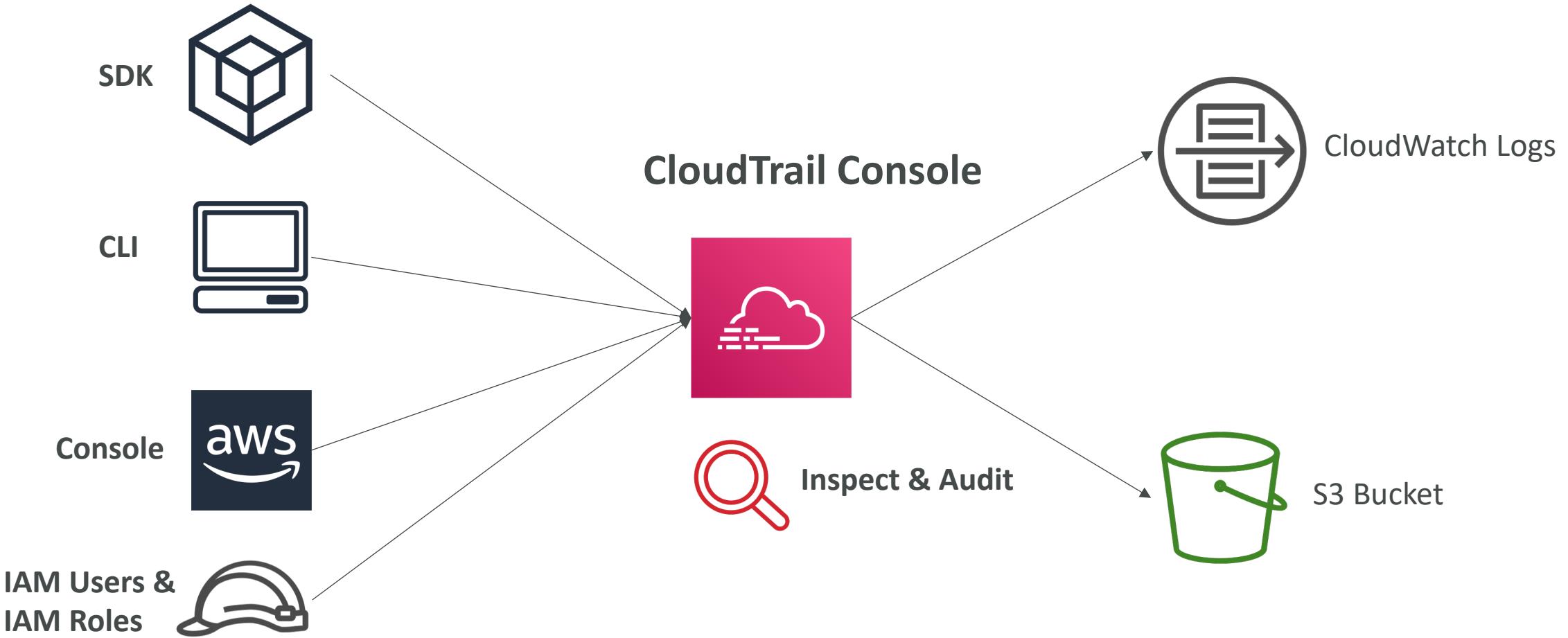
- EventBridge is the next evolution of CloudWatch Events
- **Default event bus:** generated by AWS services (CloudWatch Events)
- **Partner event bus:** receive events from SaaS service or applications (Zendesk, DataDog, Segment, Auth0...)
- **Custom Event buses:** for your own applications
- **Schema Registry:** model event schema
- EventBridge has a different name to mark the new capabilities
- The CloudWatch Events name will be replaced with EventBridge



# AWS CloudTrail

- Provides governance, compliance and audit for your AWS Account
- CloudTrail is enabled by default!
- Get an history of events / API calls made within your AWS Account by:
  - Console
  - SDK
  - CLI
  - AWS Services
- Can put logs from CloudTrail into CloudWatch Logs or S3
- A trail can be applied to All Regions (default) or a single Region.
- If a resource is deleted in AWS, investigate CloudTrail first!

# CloudTrail Diagram





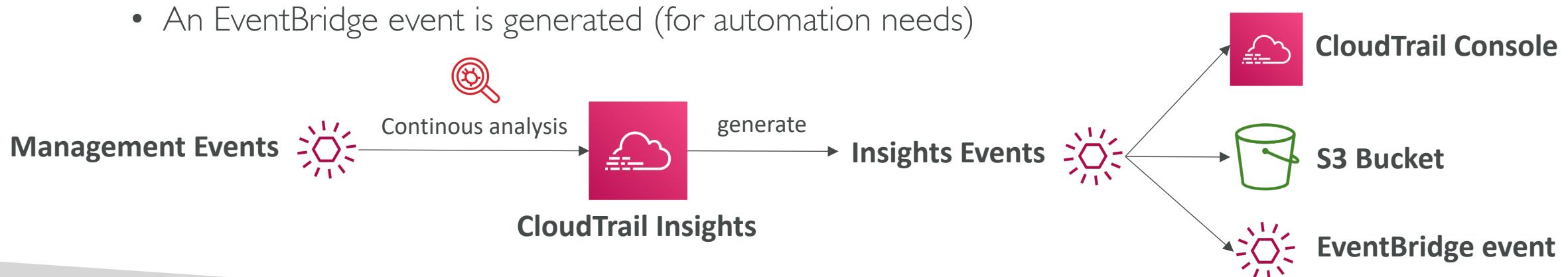
# CloudTrail Events

- Management Events:
  - Operations that are performed on resources in your AWS account
  - Examples:
    - Configuring security (IAM `AttachRolePolicy`)
    - Configuring rules for routing data (Amazon EC2 `CreateSubnet`)
    - Setting up logging (AWS CloudTrail `CreateTrail`)
  - By default, trails are configured to log management events.
  - Can separate Read Events (that don't modify resources) from Write Events (that may modify resources)
- Data Events:
  - By default, data events are not logged (because high volume operations)
  - Amazon S3 object-level activity (ex: `GetObject`, `DeleteObject`, `PutObject`): can separate Read and Write Events
  - AWS Lambda function execution activity (the `Invoke` API)
- CloudTrail Insights Events:
  - See next slide ☺



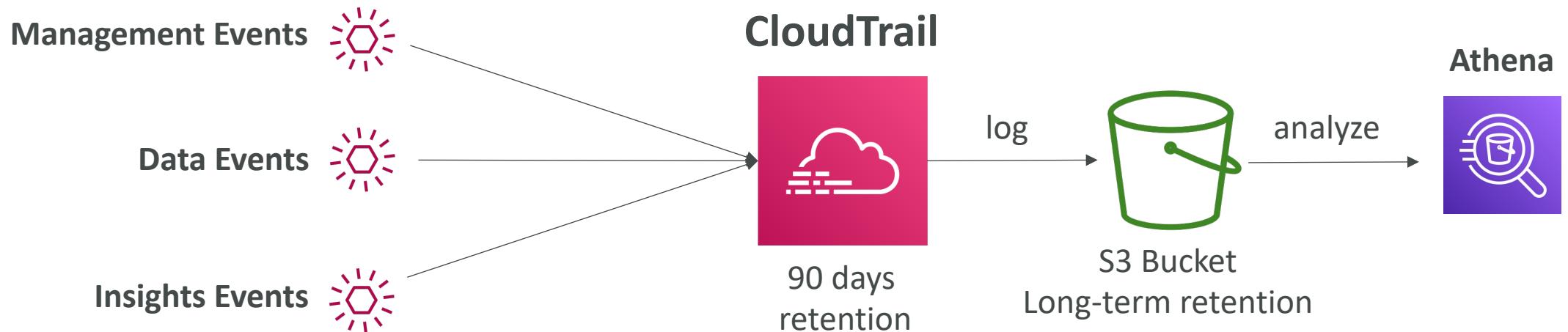
# CloudTrail Insights

- Enable CloudTrail Insights to detect unusual activity in your account:
  - inaccurate resource provisioning
  - hitting service limits
  - Bursts of AWS IAM actions
  - Gaps in periodic maintenance activity
- CloudTrail Insights analyzes normal management events to create a baseline
- And then continuously analyzes write events to detect unusual patterns
  - Anomalies appear in the CloudTrail console
  - Event is sent to Amazon S3
  - An EventBridge event is generated (for automation needs)



# CloudTrail Events Retention

- Events are stored for 90 days in CloudTrail
- To keep events beyond this period, log them to S3 and use Athena



# AWS X-Ray



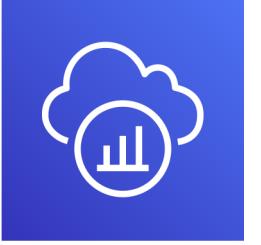
- Debugging in Production, the good old way:
  - Test locally
  - Add log statements everywhere
  - Re-deploy in production
- Log formats differ across applications and log analysis is hard.
- Debugging: one big monolith “easy”, distributed services “hard”
- No common views of your entire architecture
- Enter... AWS X-Ray!

# AWS X-Ray

## Visual analysis of our applications



# AWS X-Ray advantages

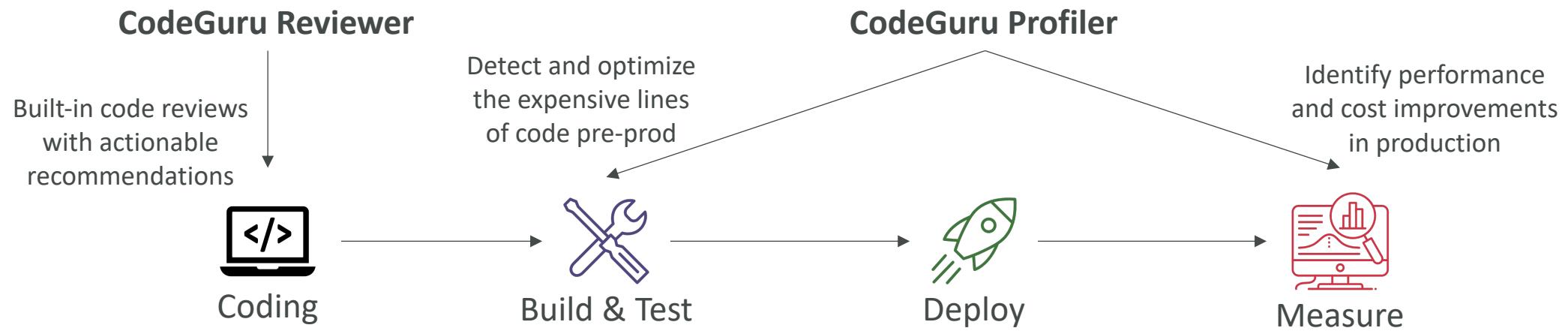


- Troubleshooting performance (bottlenecks)
- Understand dependencies in a microservice architecture
- Pinpoint service issues
- Review request behavior
- Find errors and exceptions
- Are we meeting time SLA?
- Where I am throttled?
- Identify users that are impacted

# Amazon CodeGuru



- An ML-powered service for automated code reviews and application performance recommendations
- Provides two functionalities
  - **CodeGuru Reviewer:** automated code reviews for static code analysis (development)
  - **CodeGuru Profiler:** visibility/recommendations about application performance during runtime (production)



# Amazon CodeGuru Reviewer

- Identify critical issues, security vulnerabilities, and hard-to-find bugs
- Example: common coding best practices, resource leaks, security detection, input validation
- Uses Machine Learning and automated reasoning
- Hard-learned lessons across millions of code reviews on 1000s of open-source and Amazon repositories
- Supports Java and Python
- Integrates with GitHub, Bitbucket, and AWS CodeCommit

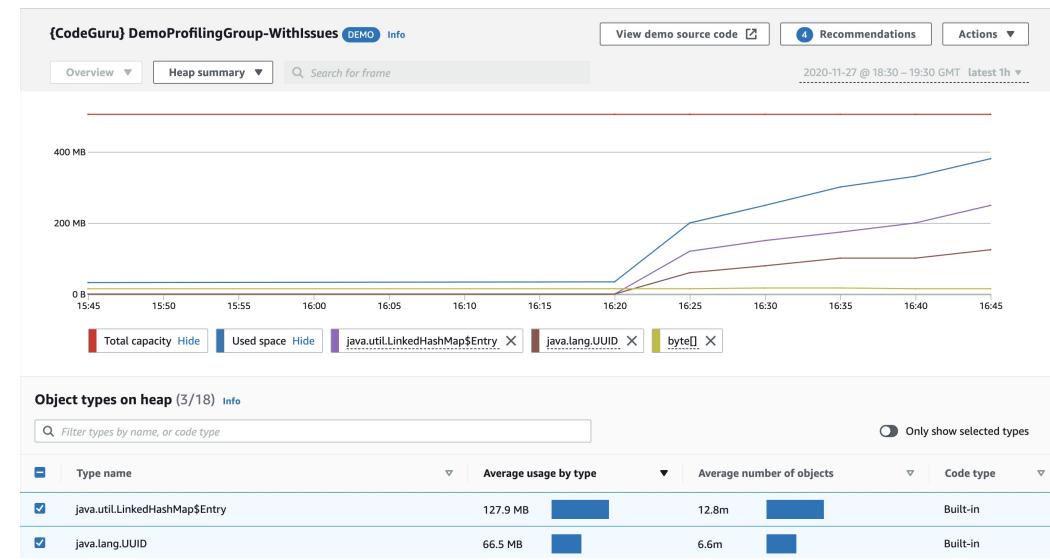
The screenshot shows the Amazon CodeGuru Reviewer interface. At the top, there's a navigation bar with 'CodeGuru' > 'Code reviews' > 'mw2tsa56o0000000'. Below it is a section titled 'RepositoryAnalysis-amazon-codeguru-reviewer-sample-app-master-mw2tsa56o0000000'. This section includes a 'Details' card with information like Status (Completed), Recommendations (4), Metered lines of code (80), ARN, Time created (10 Nov 2020 08:08:47 AM GMT-0800), and Last updated (10 Nov 2020 08:11:44 AM GMT-0800). To the right, there's a summary table with columns for Type (RepositoryAnalysis), Provider (GitHub), Repository (amazon-codeguru-reviewer-sample-app), and Branch name (master). Below this, there's a 'Recommendations (4)' section with a search bar. It lists three recommendations:

- EventHandler.java Line: 79**  
This code appears to be waiting for a resource before it runs. You could use the waiters feature to help improve efficiency. Consider using ObjectExists or ObjectNotExists. For more information, see <https://aws.amazon.com/blogs/developer/waiters-in-the-aws-sdk-for-java/>  
Was this helpful?
- EventHandler.java Line: 100**  
This code might not produce accurate results if the operation returns paginated results instead of all results. Consider adding another call to check for additional results.  
Was this helpful?
- EventHandler.java Line: 100**  
This code uses an outdated API. [ListObjectsV2](#) is the revised List Objects API, and we recommend you use this revised API for new application developments.  
Was this helpful?

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

# Amazon CodeGuru Profiler

- Helps understand the runtime behavior of your application
- Example: identify if your application is consuming excessive CPU capacity on a logging routine
- Features:
  - Identify and remove code inefficiencies
  - Improve application performance (e.g., reduce CPU utilization)
  - Decrease compute costs
  - Provides heap summary (identify which objects using up memory)
  - Anomaly Detection
- Support applications running on AWS or on-premise
- Minimal overhead on application



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

# AWS Status - Service Health Dashboard



- Shows all regions, all services health
- Shows historical information for each day
- Has an RSS feed you can subscribe to
- <https://status.aws.amazon.com/>

The screenshot shows the AWS Service Health Dashboard interface. At the top, there's a navigation bar with the AWS logo and the text "SERVICE HEALTH DASHBOARD". Below it, a breadcrumb trail says "Amazon Web Services » Service Health Dashboard" and a sub-instruction "Get a personalized view of AWS service health". A prominent yellow button labeled "Open the Personal Health Dashboard" is visible. The main content area is titled "Current Status - May 26, 2020 PDT". It includes a note about staying updated on service availability and submitting issues via "Contact Us". Below this, there are tabs for "North America", "South America", "Europe", "Africa", "Asia Pacific", and "Middle East", with "North America" being the active tab. A "Recent Events" section shows "No recent events." with a green checkmark icon. The "Remaining Services" section lists several AWS services, each with a green checkmark icon, "Details" link, and an "RSS" icon. All listed services are currently operating normally.

Region	Service	Status	RSS
North America	Alexa for Business (N. Virginia)	Service is operating normally	
North America	Amazon API Gateway (Montreal)	Service is operating normally	
North America	Amazon API Gateway (N. California)	Service is operating normally	
North America	Amazon API Gateway (N. Virginia)	Service is operating normally	
North America	Amazon API Gateway (Ohio)	Service is operating normally	
North America	Amazon API Gateway (Oregon)	Service is operating normally	
North America	Amazon AppStream 2.0 (N. Virginia)	Service is operating normally	
North America	Amazon AppStream 2.0 (Oregon)	Service is operating normally	
North America	Amazon Athena (Montreal)	Service is operating normally	

# AWS Personal Health Dashboard

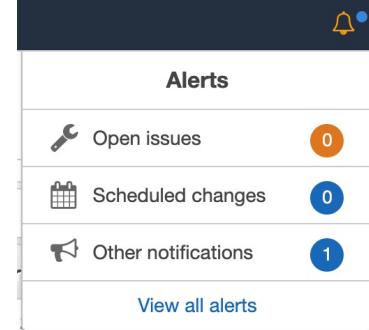


- AWS Personal Health Dashboard provides **alerts** and **remediation guidance** when AWS is experiencing **events** that may impact you.
- While the Service Health Dashboard displays the general status of AWS services, Personal Health Dashboard gives you a **personalized view** into the performance and availability of the AWS services underlying your AWS resources.
- The dashboard displays **relevant** and **timely** information to help you manage events in progress and provides proactive notification to help you plan for scheduled activities.

# AWS Personal Health Dashboard



- Global service <https://phd.aws.amazon.com/>
- Shows how AWS outages directly impact you & your AWS resources
- Alert, remediation, proactive, scheduled activities



Event log

<input type="button" value="Add filter"/> <span style="float: right;">↻</span> <span style="float: right;">?</span>							
	Event	Status	Region/AZ	Start time	Last update time	Affected resources	Event category
<input type="radio"/>	ElasticContainerRegistry operational issue	Closed	us-west-2	May 22, 2020 at 11:48:49 PM U...	May 22, 2020 at 11:49:31 PM U...	-	🔧 Issue
<input type="radio"/>	CodeBuild operational notification	-	-	May 21, 2020 at 11:20:00 PM U...	May 21, 2020 at 11:35:26 PM U...	1 entity	📣 Notification
<input type="radio"/>	ElasticsearchService operational issue	Closed	us-east-1	May 21, 2020 at 3:44:30 PM UT...	May 21, 2020 at 4:38:20 PM UT...	-	🔧 Issue
<input type="radio"/>	Batch operational issue	Closed	us-west-1	May 10, 2020 at 3:38:49 AM UT...	May 10, 2020 at 5:55:46 AM UT...	-	🔧 Issue
<input type="radio"/>	ElasticContainerService operational issue	Closed	us-west-1	May 10, 2020 at 3:31:30 AM UT...	May 10, 2020 at 5:52:25 AM UT...	-	🔧 Issue
<input type="radio"/>	CloudFormation operational issue	Closed	us-west-2	April 30, 2020 at 9:47:10 PM UT...	April 30, 2020 at 11:11:31 PM U...	-	🔧 Issue
<input type="radio"/>	CloudFront operational issue	Closed	-	April 21, 2020 at 11:57:30 PM U...	April 22, 2020 at 12:28:15 AM U...	-	🔧 Issue
...							

# Monitoring Summary

- **CloudWatch:**
  - **Metrics:** monitor the performance of AWS services and billing metrics
  - **Alarms:** automate notification, perform EC2 action, notify to SNS based on metric
  - **Logs:** collect log files from EC2 instances, servers, Lambda functions...
  - **Events (or EventBridge):** react to events in AWS, or trigger a rule on a schedule
- **CloudTrail:** audit API calls made within your AWS account
- **CloudTrail Insights:** automated analysis of your CloudTrail Events
- **X-Ray:** trace requests made through your distributed applications
- **Service Health Dashboard:** status of all AWS services across all regions
- **Personal Health Dashboard:** AWS events that impact your infrastructure
- **Amazon CodeGuru:** automated code reviews and application performance recommendations

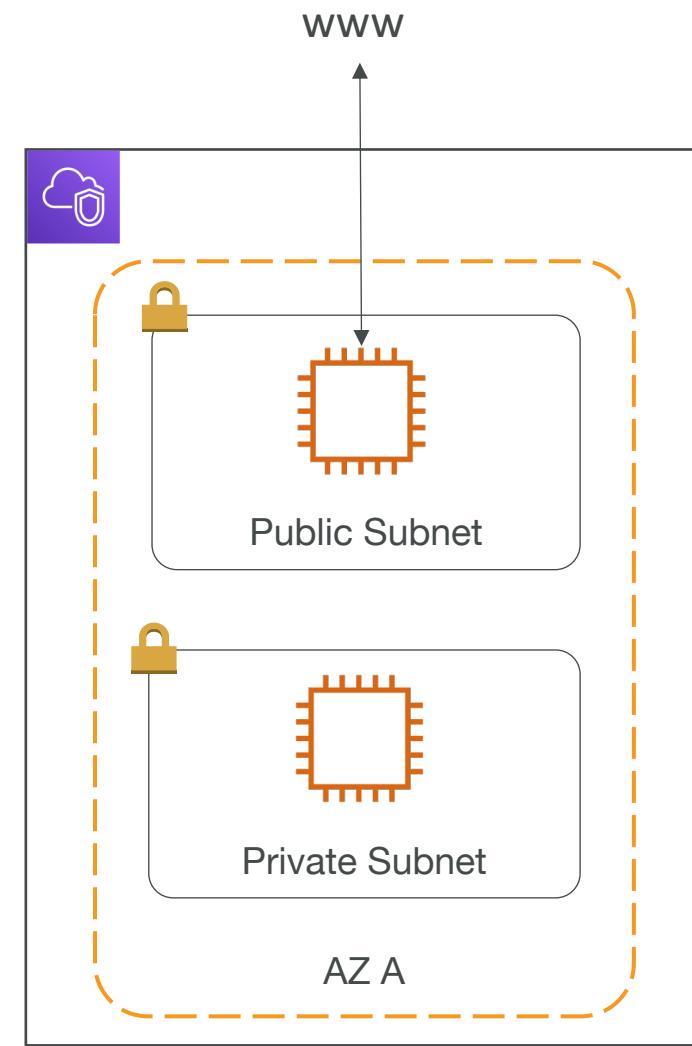
# VPC Section

# VPC – Crash Course

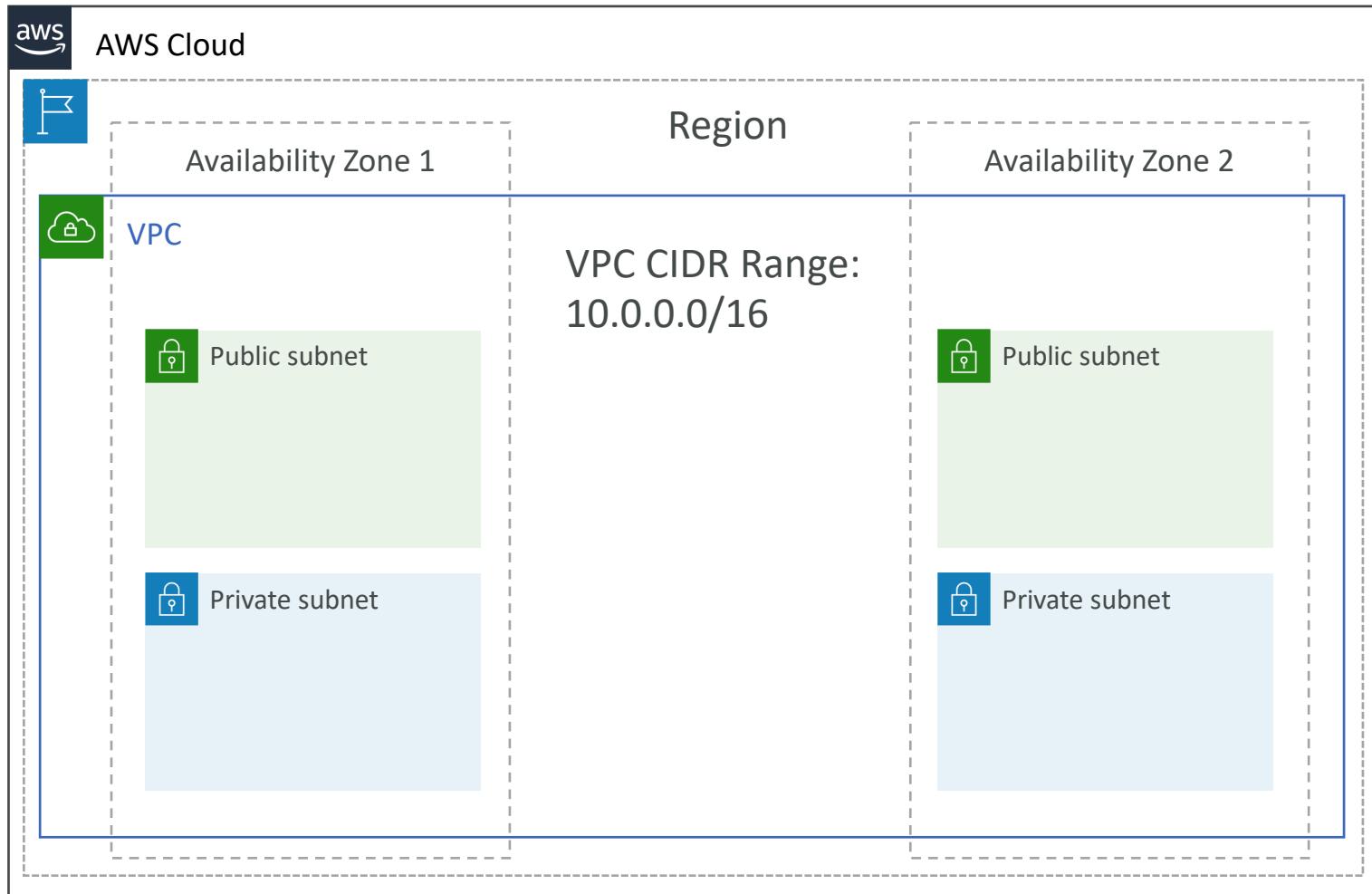
- VPC is something you should know in depth for the AWS Certified Solutions Architect Associate & AWS Certified SysOps Administrator
- At the AWS Certified Cloud Practitioner Level, you should know about:
  - VPC, Subnets, Internet Gateways & NAT Gateways
  - Security Groups, Network ACL (NACL), VPC Flow Logs
  - VPC Peering, VPC Endpoints
  - Site to Site VPN & Direct Connect
  - Transit Gateway
- I will just give you an overview, less than 1 or 2 questions at your exam.
- We'll have a look at the “default VPC” (created by default by AWS for you)
- There is a summary lecture at the end. It's okay if you don't understand it all

# VPC & Subnets Primer

- **VPC - Virtual Private Cloud:** private network to deploy your resources (regional resource)
- **Subnets** allow you to partition your network inside your VPC (Availability Zone resource)
- A **public subnet** is a subnet that is accessible from the internet
- A **private subnet** is a subnet that is not accessible from the internet
- To define access to the internet and between subnets, we use **Route Tables**.

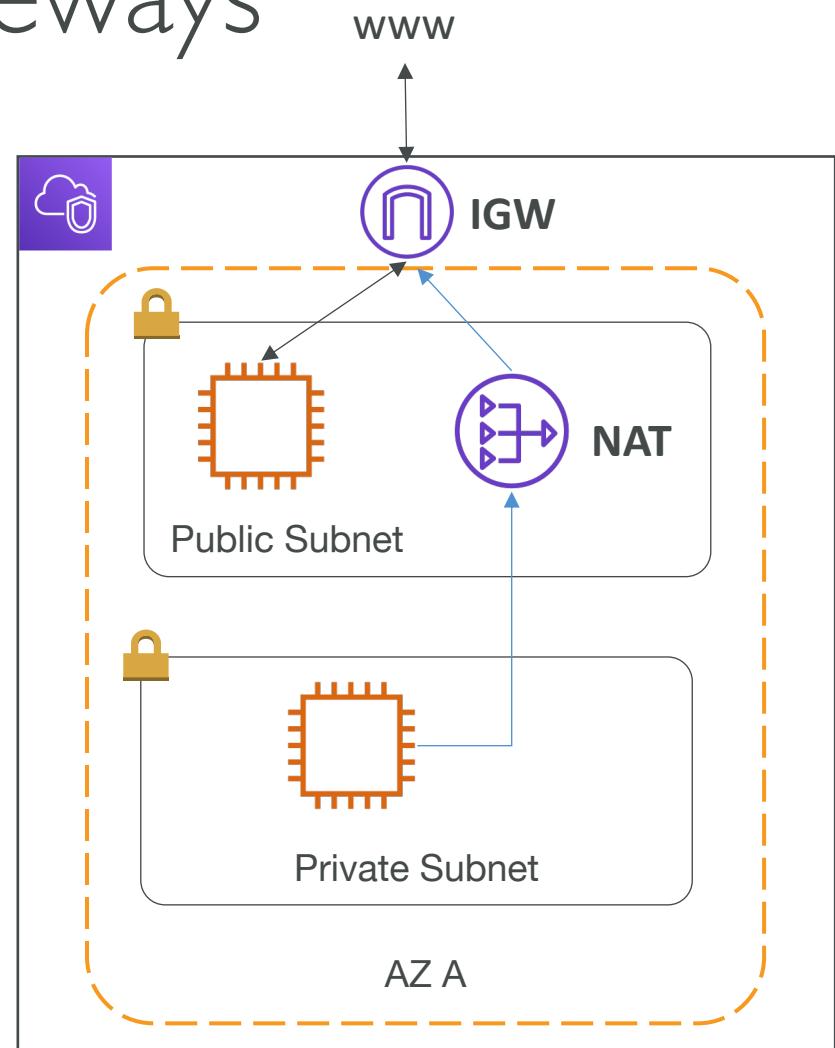


# VPC Diagram



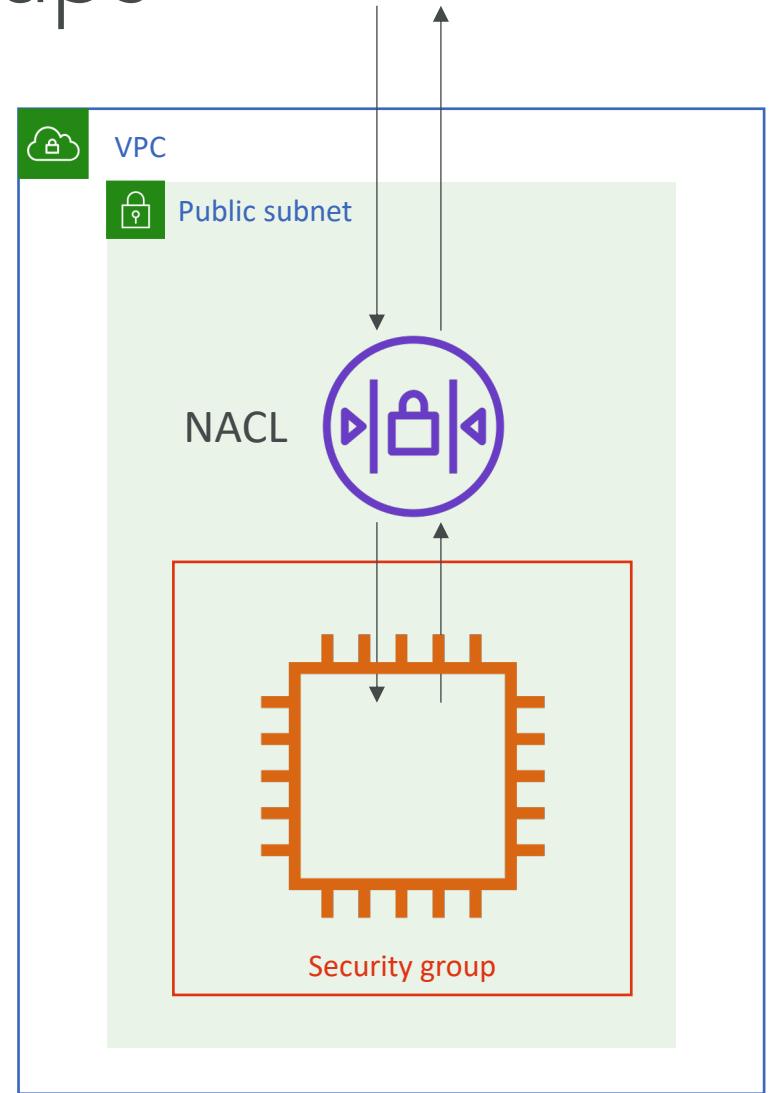
# Internet Gateway & NAT Gateways

- Internet Gateways helps our VPC instances connect with the internet
- Public Subnets have a route to the internet gateway.
- NAT Gateways (AWS-managed) & NAT Instances (self-managed) allow your instances in your Private Subnets to access the internet while remaining private



# Network ACL & Security Groups

- NACL (Network ACL)
  - A firewall which controls traffic from and to subnet
  - Can have ALLOW and DENY rules
  - Are attached at the **Subnet** level
  - Rules only include IP addresses
- Security Groups
  - A firewall that controls traffic to and from **an ENI / an EC2 Instance**
  - Can have only ALLOW rules
  - Rules include IP addresses and other security groups



# Network ACLs vs Security Groups

Security Group	Network ACL
Operates at the instance level	Operates at the subnet level
Supports allow rules only	Supports allow rules and deny rules
Is stateful: Return traffic is automatically allowed, regardless of any rules	Is stateless: Return traffic must be explicitly allowed by rules
We evaluate all rules before deciding whether to allow traffic	We process rules in number order when deciding whether to allow traffic
Applies to an instance only if someone specifies the security group when launching the instance, or associates the security group with the instance later on	Automatically applies to all instances in the subnets it's associated with (therefore, you don't have to rely on users to specify the security group)

[https://docs.aws.amazon.com/vpc/latest/userguide/VPC\\_Security.html#VPC\\_Security\\_Comparison](https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Security.html#VPC_Security_Comparison)

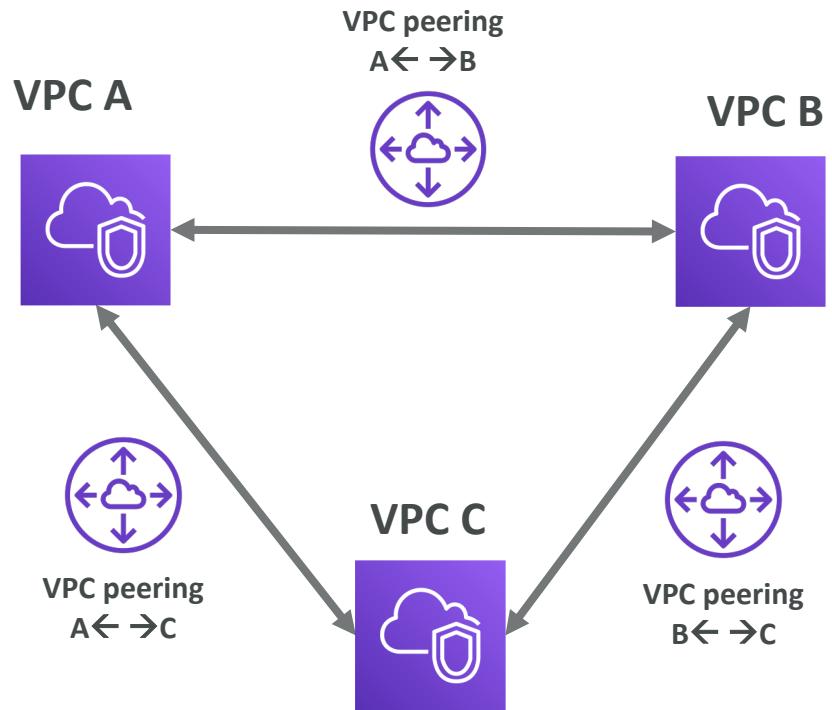


# VPC Flow Logs

- Capture information about IP traffic going into your interfaces:
  - VPC Flow Logs
  - Subnet Flow Logs
  - Elastic Network Interface Flow Logs
- Helps to monitor & troubleshoot connectivity issues. Example:
  - Subnets to internet
  - Subnets to subnets
  - Internet to subnets
- Captures network information from AWS managed interfaces too: Elastic Load Balancers, ElastiCache, RDS, Aurora, etc...
- VPC Flow logs data can go to S3 / CloudWatch Logs

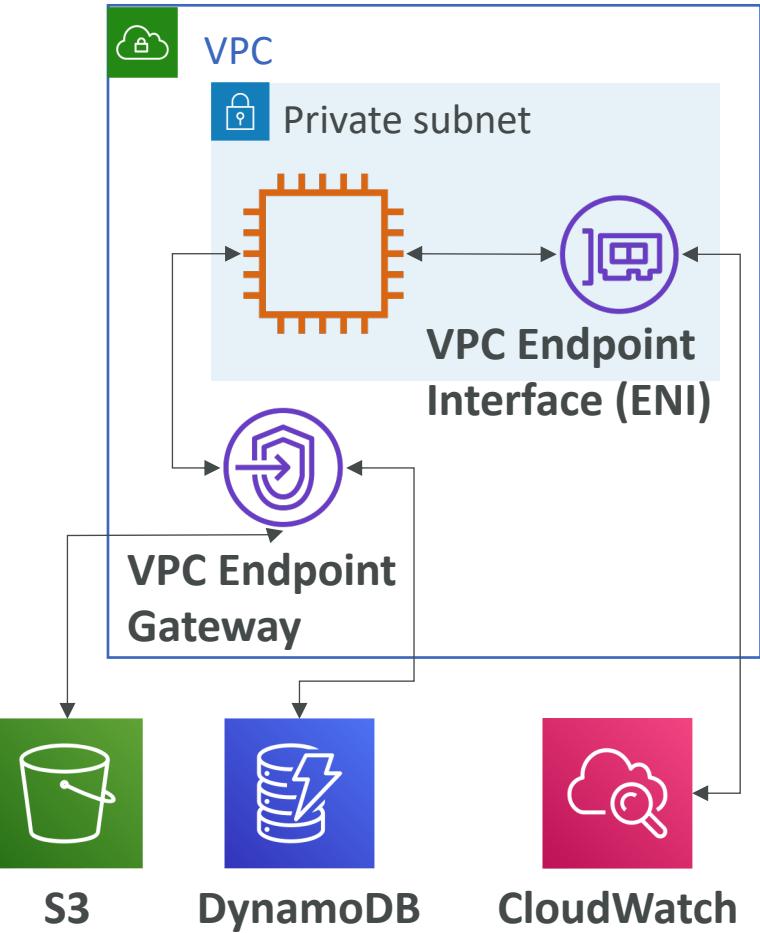
# VPC Peering

- Connect two VPC, privately using AWS' network
- Make them behave as if they were in the same network
- Must not have overlapping CIDR (IP address range)
- VPC Peering connection is **not transitive** (must be established for each VPC that need to communicate with one another)



# VPC Endpoints

- Endpoints allow you to connect to AWS Services **using a private network** instead of the public www network
- This gives you enhanced security and lower latency to access AWS services
- VPC Endpoint **Gateway**: S3 & DynamoDB
- VPC Endpoint **Interface**: the rest



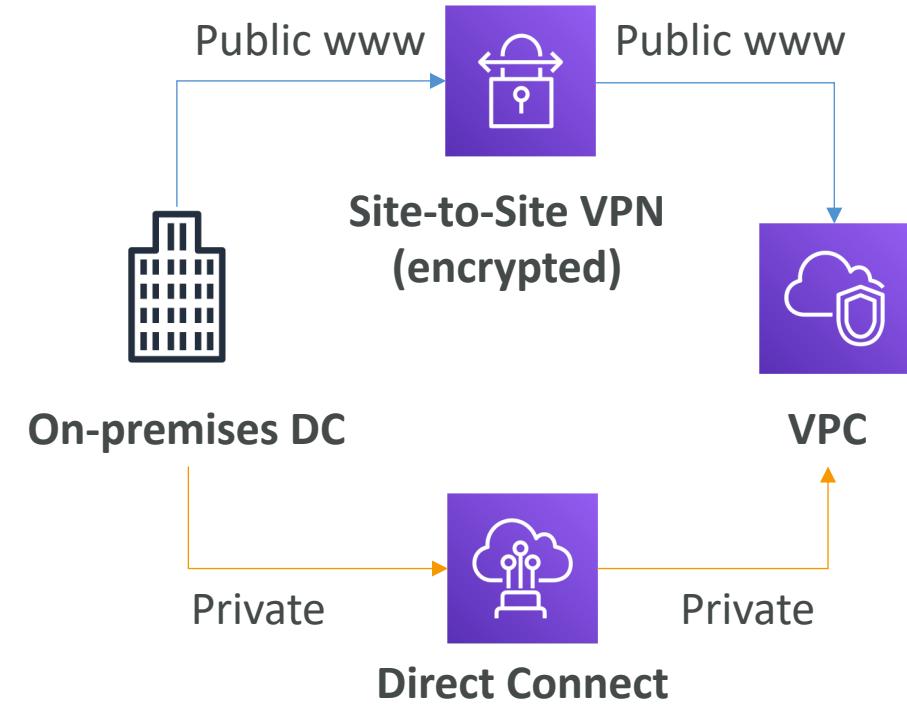
# Site to Site VPN & Direct Connect

- **Site to Site VPN**

- Connect an on-premises VPN to AWS
- The connection is automatically encrypted
- Goes over the public internet

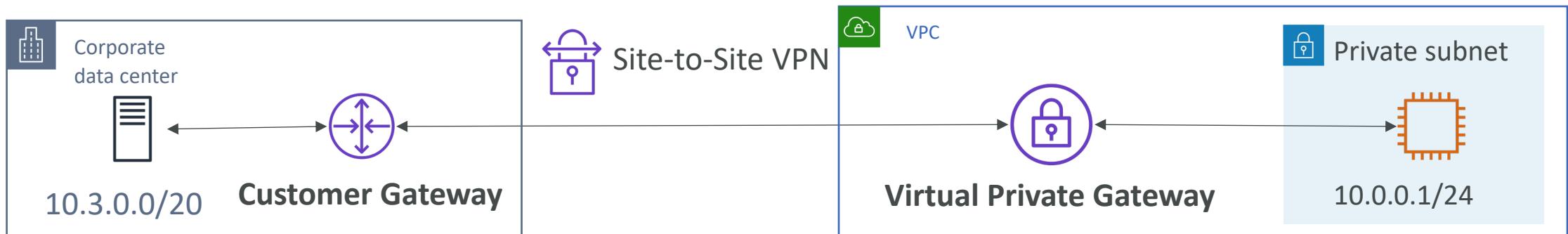
- **Direct Connect (DX)**

- Establish a physical connection between on-premises and AWS
- The connection is private, secure and fast
- Goes over a private network
- Takes at least a month to establish

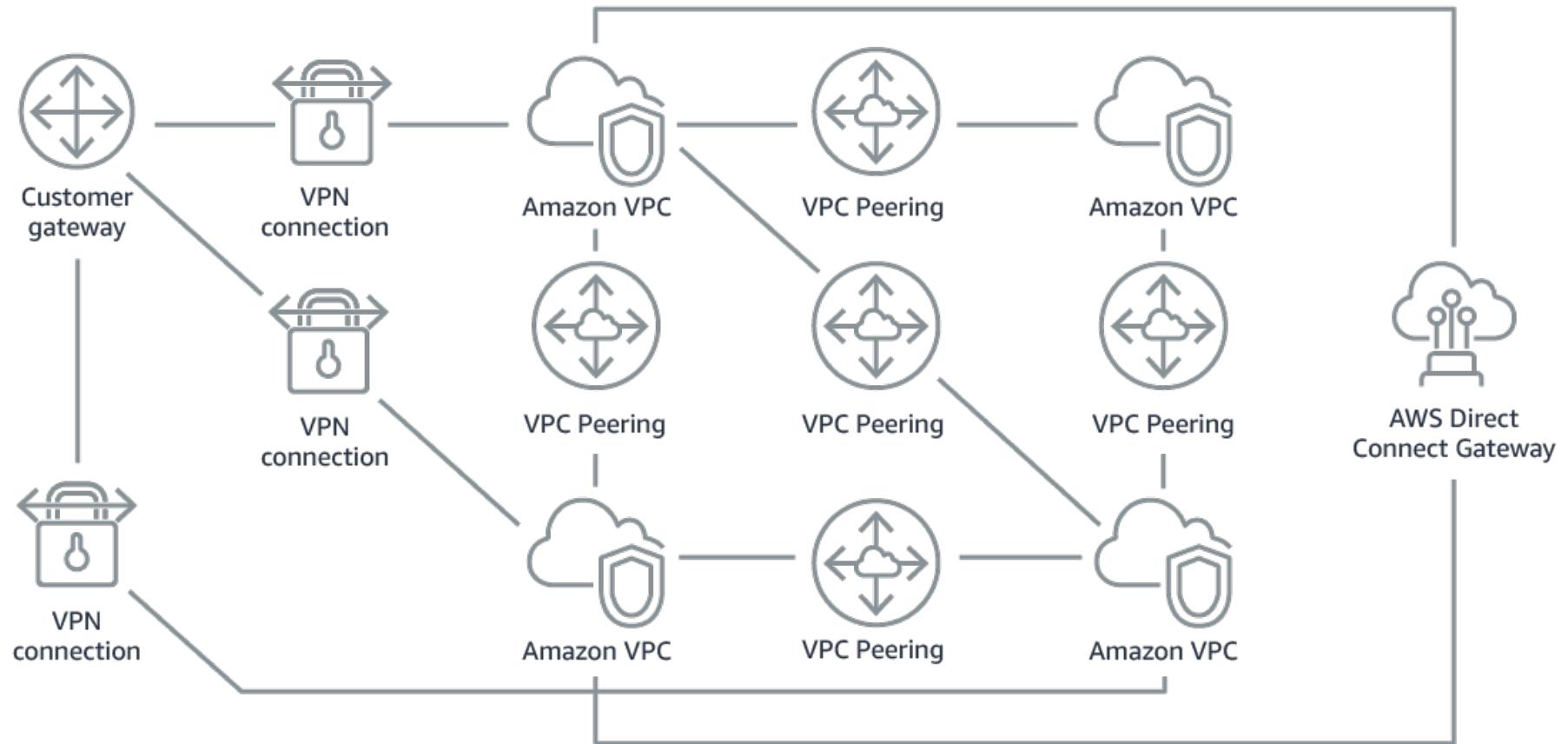


# Site-to-Site VPN

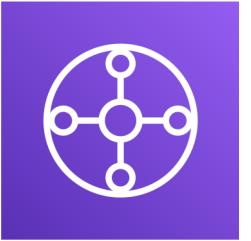
- On-premises: must use a **Customer Gateway** (CGW)
- AWS: must use a **Virtual Private Gateway** (VGW)



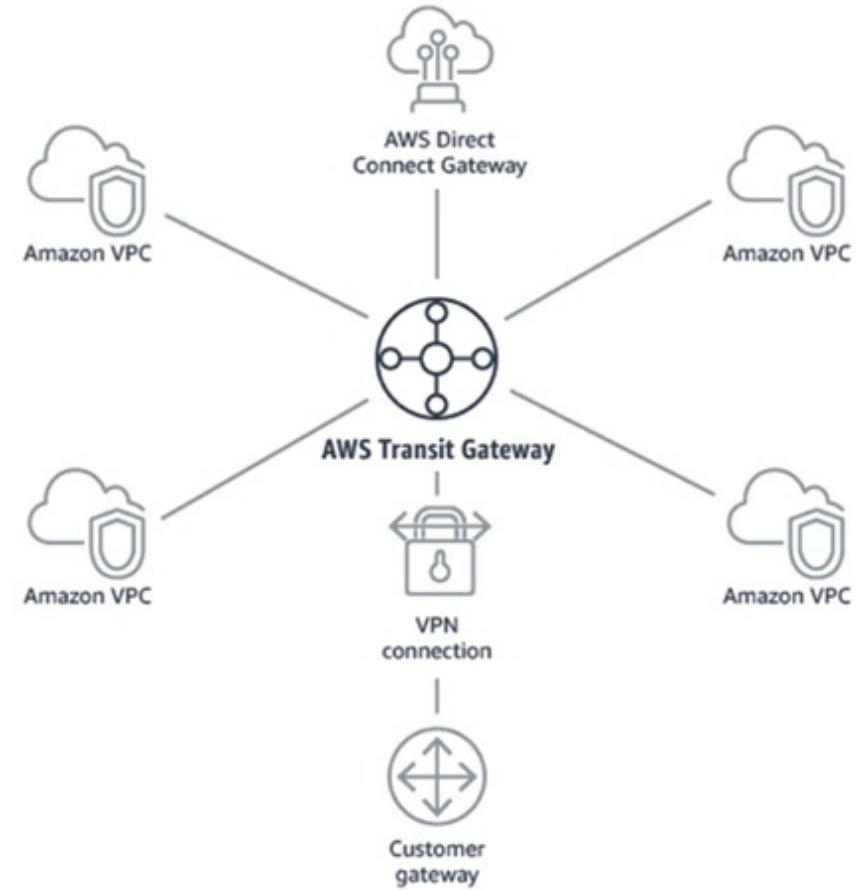
# Network topologies can become complicated



# Transit Gateway



- For having transitive peering between thousands of VPC and on-premises, hub-and-spoke (star) connection
- One single Gateway to provide this functionality
- Works with Direct Connect Gateway, VPN connections



# VPC Closing Comments

- **VPC:** Virtual Private Cloud
- **Subnets:** Tied to an AZ, network partition of the VPC
- **Internet Gateway:** at the VPC level, provide Internet Access
- **NAT Gateway / Instances:** give internet access to private subnets
- **NACL:** Stateless, subnet rules for inbound and outbound
- **Security Groups:** Stateful, operate at the EC2 instance level or ENI
- **VPC Peering:** Connect two VPC with non overlapping IP ranges, nontransitive
- **VPC Endpoints:** Provide private access to AWS Services within VPC
- **VPC Flow Logs:** network traffic logs
- **Site to Site VPN:** VPN over public internet between on-premises DC and AWS
- **Direct Connect:** direct private connection to AWS
- **Transit Gateway:** Connect thousands of VPC and on-premises networks together

# Security & Compliance Section

# AWS Shared Responsibility Model

- AWS responsibility - Security **of** the Cloud
  - Protecting infrastructure (hardware, software, facilities, and networking) that runs all the AWS services
  - Managed services like S3, DynamoDB, RDS, etc.
- Customer responsibility - Security **in** the Cloud
  - For EC2 instance, customer is responsible for management of the guest OS (including security patches and updates), firewall & network configuration, IAM
  - Encrypting application data
- Shared controls:
  - Patch Management, Configuration Management, Awareness & Training

# Example, for RDS



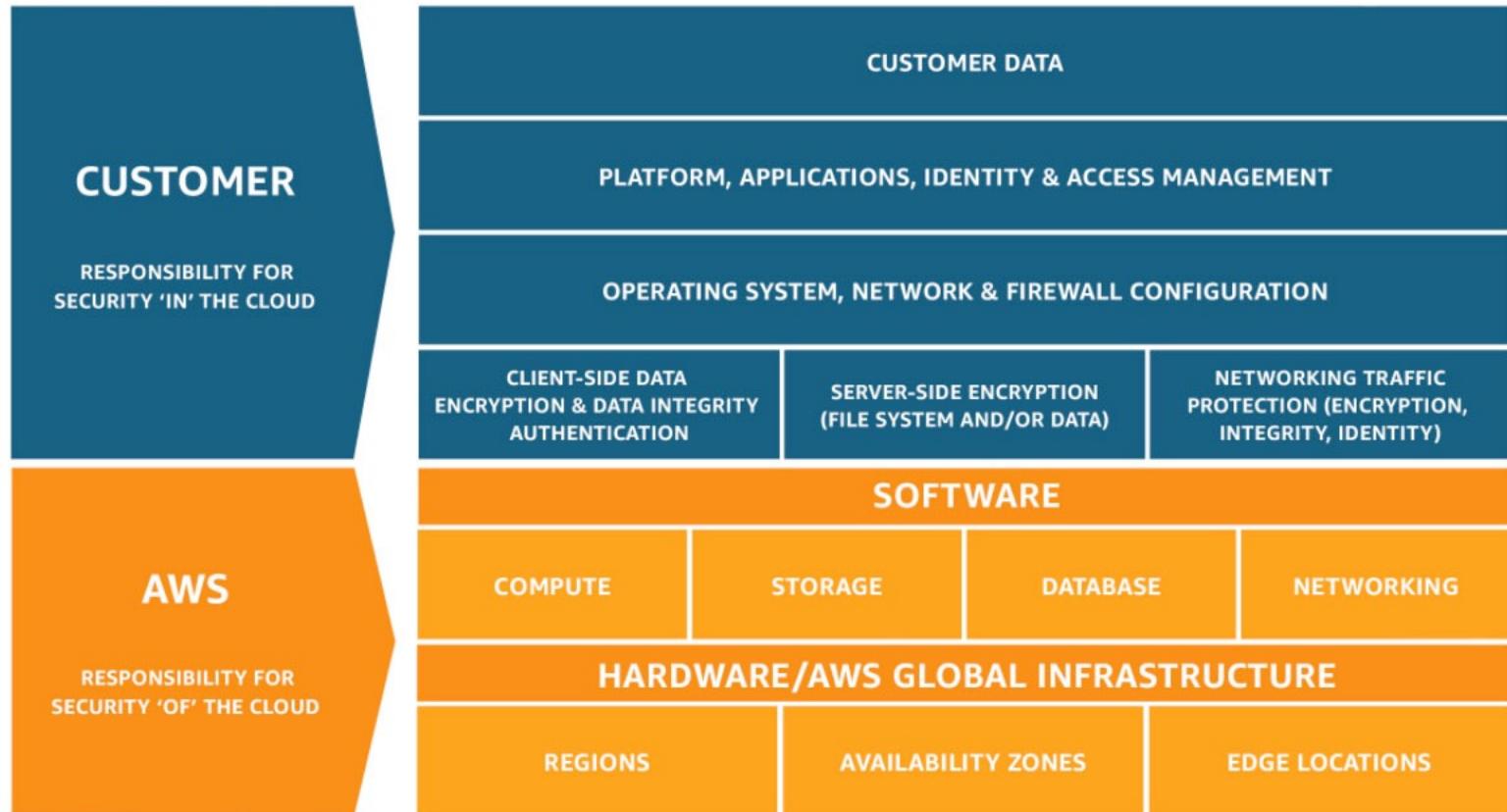
- AWS responsibility:
  - Manage the underlying EC2 instance, disable SSH access
  - Automated DB patching
  - Automated OS patching
  - Audit the underlying instance and disks & guarantee it functions
- Your responsibility:
  - Check the ports / IP / security group inbound rules in DB's SG
  - In-database user creation and permissions
  - Creating a database with or without public access
  - Ensure parameter groups or DB is configured to only allow SSL connections
  - Database encryption setting



# Example, for S3

- AWS responsibility:
  - Guarantee you get unlimited storage
  - Guarantee you get encryption
  - Ensure separation of the data between different customers
  - Ensure AWS employees can't access your data
- Your responsibility:
  - Bucket configuration
  - Bucket policy / public setting
  - IAM user and roles
  - Enabling encryption

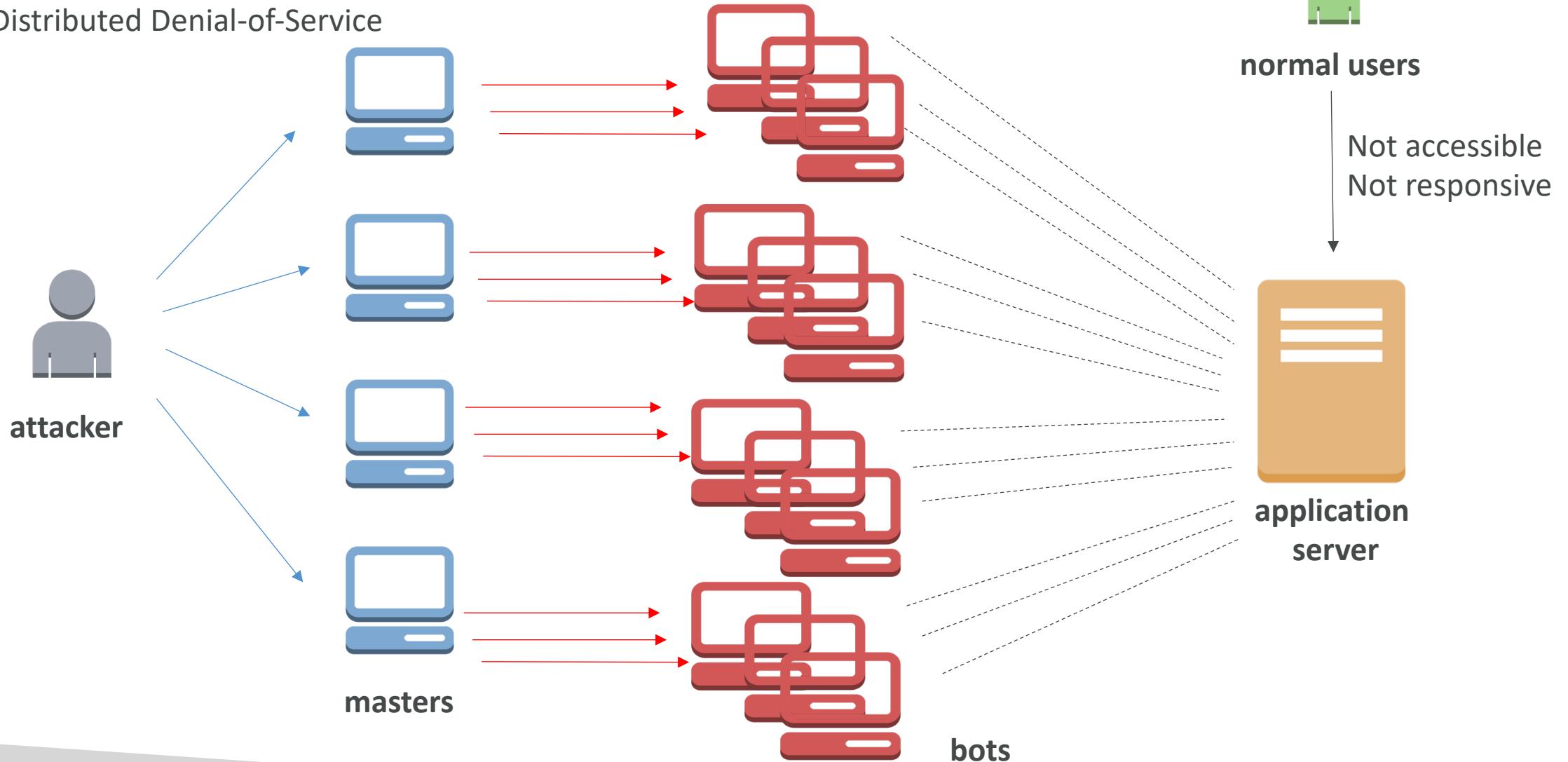
# Shared Responsibility Model diagram



<https://aws.amazon.com/compliance/shared-responsibility-model/>

# What's a DDOS\* Attack?

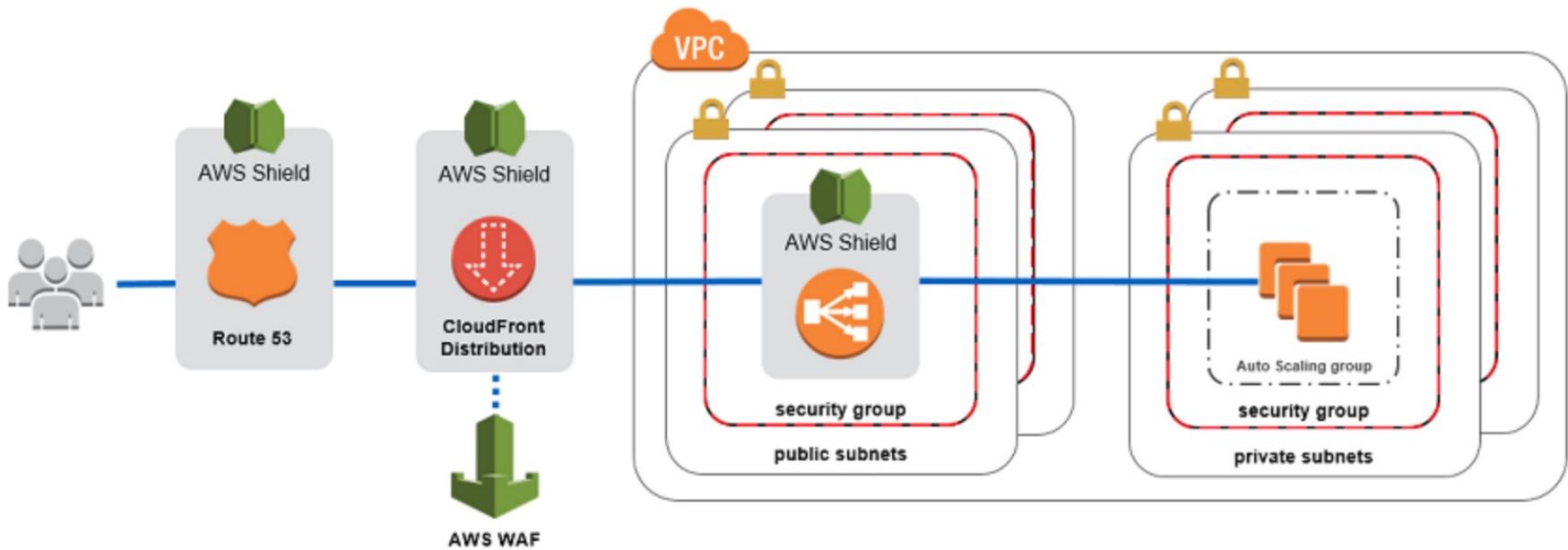
\*Distributed Denial-of-Service



# DDOS Protection on AWS

- **AWS Shield Standard:** protects against DDOS attack for your website and applications, for all customers at no additional costs
- **AWS Shield Advanced:** 24/7 premium DDoS protection
- **AWS WAF:** Filter specific requests based on rules
- **CloudFront and Route 53:**
  - Availability protection using global edge network
  - Combined with AWS Shield, provides attack mitigation at the edge
- Be ready to scale – leverage **AWS Auto Scaling**

# Sample Reference Architecture for DDoS Protection



<https://aws.amazon.com/answers/networking/aws-ddos-attack-mitigation/>

# AWS Shield



- AWS Shield Standard:
  - Free service that is activated for every AWS customer
  - Provides protection from attacks such as SYN/UDP Floods, Reflection attacks and other layer 3/layer 4 attacks
- AWS Shield Advanced:
  - Optional DDoS mitigation service (\$3,000 per month per organization)
  - Protect against more sophisticated attack on [Amazon EC2](#), [Elastic Load Balancing \(ELB\)](#), [Amazon CloudFront](#), [AWS Global Accelerator](#), and [Route 53](#)
  - 24/7 access to AWS DDoS response team (DRP)
  - Protect against higher fees during usage spikes due to DDoS

# AWS WAF – Web Application Firewall



- Protects your web applications from common web exploits (Layer 7)
- Layer 7 is HTTP (vs Layer 4 is TCP)
- Deploy on Application Load Balancer, API Gateway, CloudFront
- Define Web ACL (Web Access Control List):
  - Rules can include IP addresses, HTTP headers, HTTP body, or URI strings
  - Protects from common attack - SQL injection and Cross-Site Scripting (XSS)
  - Size constraints, geo-match (block countries)
  - Rate-based rules (to count occurrences of events) – for DDoS protection



# Penetration Testing on AWS Cloud

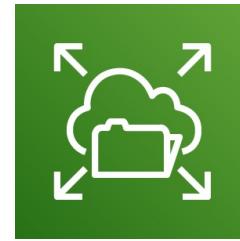
- AWS customers are welcome to carry out security assessments or penetration tests against their AWS infrastructure **without prior approval for 8 services:**
  - Amazon EC2 instances, NAT Gateways, and Elastic Load Balancers
  - Amazon RDS
  - Amazon CloudFront
  - Amazon Aurora
  - Amazon API Gateways
  - AWS Lambda and Lambda Edge functions
  - Amazon Lightsail resources
  - Amazon Elastic Beanstalk environments
- List can increase over time (you won't be tested on that at the exam)



# Penetration Testing on your AWS Cloud

- Prohibited Activities
  - DNS zone walking via Amazon Route 53 Hosted Zones
  - Denial of Service (DoS), Distributed Denial of Service (DDoS), Simulated DoS, Simulated DDoS
  - Port flooding
  - Protocol flooding
  - Request flooding (login request flooding, API request flooding)
- For any other simulated events, contact [aws-security-simulated-event@amazon.com](mailto:aws-security-simulated-event@amazon.com)
- Read more: <https://aws.amazon.com/security/penetration-testing/>

# Data at rest vs. Data in transit



Encrypted at rest on EFS

Encrypted in transit while uploading



Encrypted at rest on S3

- **At rest:** data stored or archived on a device
  - On a hard disk, on a RDS instance, in S3 Glacier Deep Archive, etc.
- **In transit (in motion):** data being moved from one location to another
  - Transfer from on-premises to AWS, EC2 to DynamoDB, etc.
  - Means data transferred on the network
- We want to encrypt data in both states to protect it!
- For this we leverage **encryption keys**

# AWS KMS (Key Management Service)



- Anytime you hear “encryption” for an AWS service, it’s most likely KMS
- KMS = AWS manages the encryption keys for us
- **Encryption Opt-in:**
  - EBS volumes: encrypt volumes
  - S3 buckets: Server-side encryption of objects
  - Redshift database: encryption of data
  - RDS database: encryption of data
  - EFS drives: encryption of data
- **Encryption Automatically enabled:**
  - CloudTrail Logs
  - S3 Glacier
  - Storage Gateway

# CloudHSM

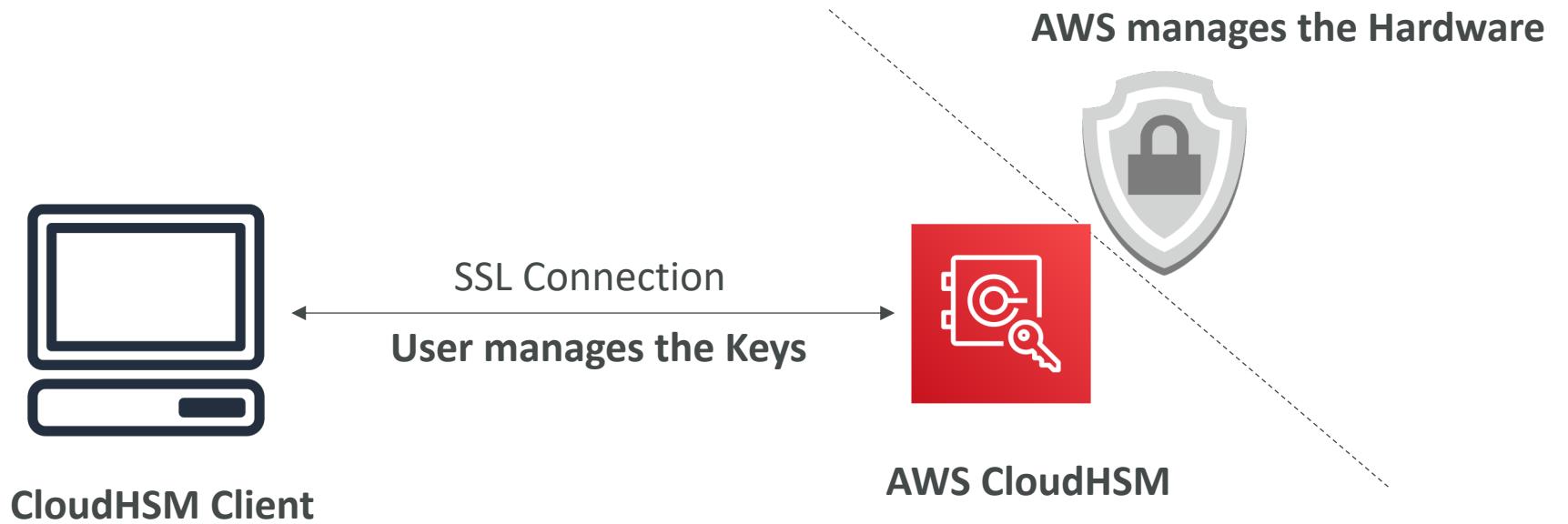


- KMS => AWS manages the software for encryption
- CloudHSM => AWS provisions encryption **hardware**
- Dedicated Hardware (HSM = Hardware Security Module)
- You manage your own encryption keys entirely (not AWS)
- HSM device is tamper resistant, FIPS 140-2 Level 3 compliance



Sample HSM device

# CloudHSM Diagram



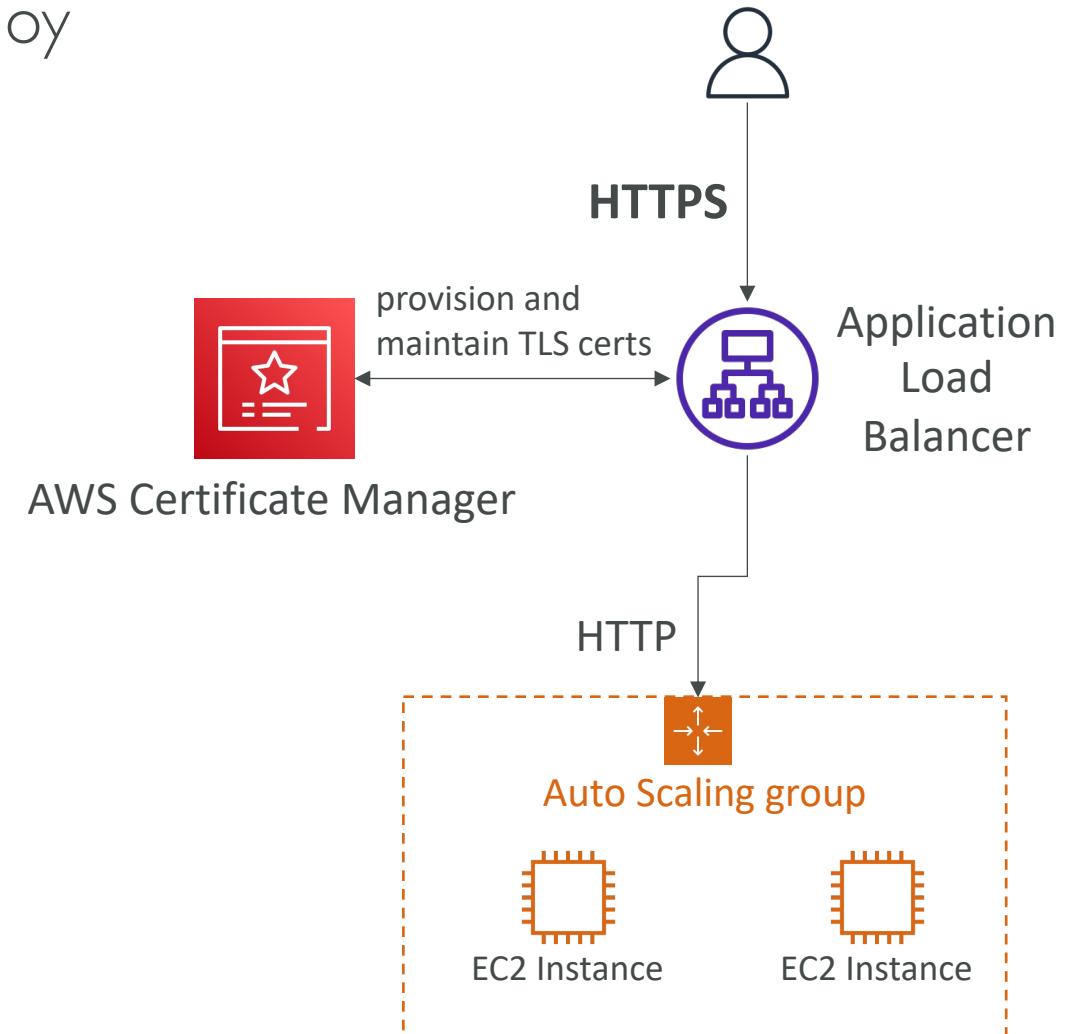
# Types of Customer Master Keys: CMK

- **Customer Managed CMK:**
  - Create, manage and used by the customer, can enable or disable
  - Possibility of rotation policy (new key generated every year; old key preserved)
  - Possibility to bring-your-own-key
- **AWS managed CMK:**
  - Created, managed and used on the customer's behalf by AWS
  - Used by AWS services (aws/s3, aws/ebs, aws/redshift)
- **AWS owned CMK:**
  - Collection of CMKs that an AWS service owns and manages to use in multiple accounts
  - AWS can use those to protect resources in your account (but you can't view the keys)
- **CloudHSM Keys (custom keystore):**
  - Keys generated from your own CloudHSM hardware device
  - Cryptographic operations are performed within the CloudHSM cluster

# AWS Certificate Manager (ACM)



- Lets you easily provision, manage, and deploy SSL/TLS Certificates
- Used to provide in-flight encryption for websites (HTTPS)
- Supports both public and private TLS certificates
- Free of charge for public TLS certificates
- Automatic TLS certificate renewal
- Integrations with (load TLS certificates on)
  - Elastic Load Balancers
  - CloudFront Distributions
  - APIs on API Gateway



# AWS Secrets Manager



- Newer service, meant for storing secrets
- Capability to force **rotation of secrets** every X days
- Automate generation of secrets on rotation (uses Lambda)
- Integration with **Amazon RDS** (MySQL, PostgreSQL, Aurora)
- Secrets are encrypted using KMS
- Mostly meant for RDS integration



# AWS Artifact (not really a service)

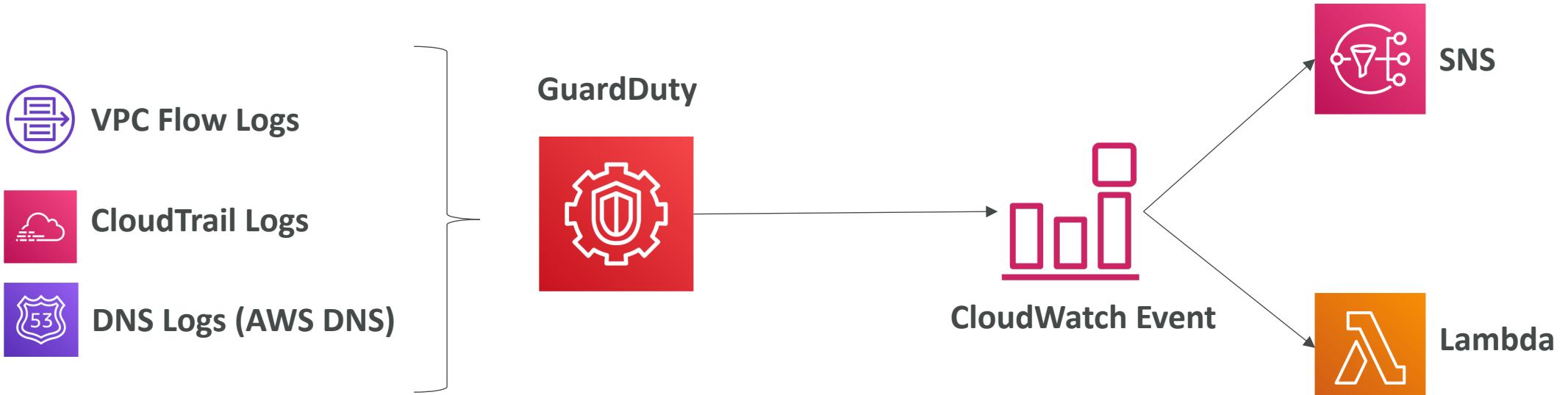
- Portal that provides customers with on-demand access to AWS compliance documentation and AWS agreements
- **Artifact Reports** - Allows you to download AWS security and compliance documents from third-party auditors, like AWS ISO certifications, Payment Card Industry (PCI), and System and Organization Control (SOC) reports
- **Artifact Agreements** - Allows you to review, accept, and track the status of AWS agreements such as the Business Associate Addendum (BAA) or the Health Insurance Portability and Accountability Act (HIPAA) for an individual account or in your organization
- Can be used to support internal audit or compliance



# Amazon GuardDuty

- Intelligent Threat discovery to Protect AWS Account
- Uses Machine Learning algorithms, anomaly detection, 3<sup>rd</sup> party data
- One click to enable (30 days trial), no need to install software
- Input data includes:
  - CloudTrail Logs: unusual API calls, unauthorized deployments
  - VPC Flow Logs: unusual internal traffic, unusual IP address
  - DNS Logs: compromised EC2 instances sending encoded data within DNS queries
- Can setup **CloudWatch Event rules** to be notified in case of findings
- CloudWatch Events rules can target AWS Lambda or SNS

# Amazon GuardDuty



# Amazon Inspector



- Automated Security Assessments for EC2 instances
- Analyze the running OS against known vulnerabilities
- Analyze against unintended network accessibility
- AWS Inspector Agent must be installed on OS in EC2 instances
- After the assessment, you get a report with a list of vulnerabilities



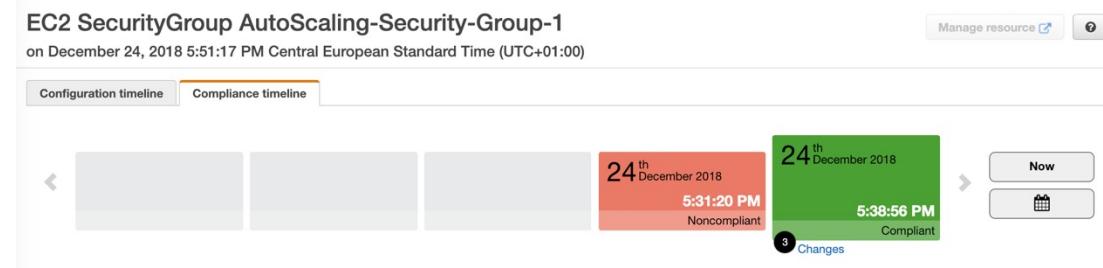
# AWS Config



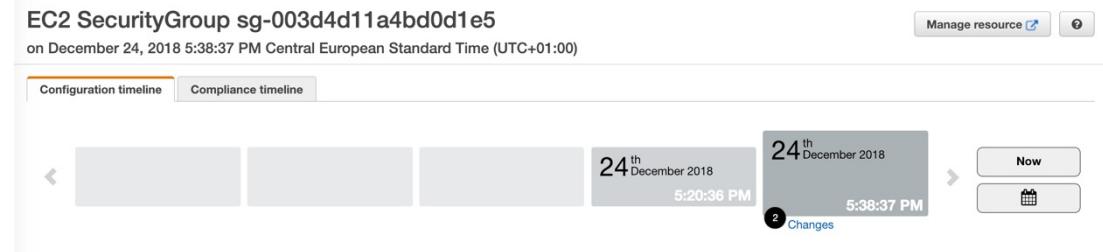
- Helps with auditing and recording compliance of your AWS resources
- Helps record configurations and changes over time
- Possibility of storing the configuration data into S3 (analyzed by Athena)
- Questions that can be solved by AWS Config:
  - Is there unrestricted SSH access to my security groups?
  - Do my buckets have any public access?
  - How has my ALB configuration changed over time?
- You can receive alerts (SNS notifications) for any changes
- AWS Config is a per-region service
- Can be aggregated across regions and accounts

# AWS Config Resource

- View compliance of a resource over time



- View configuration of a resource over time

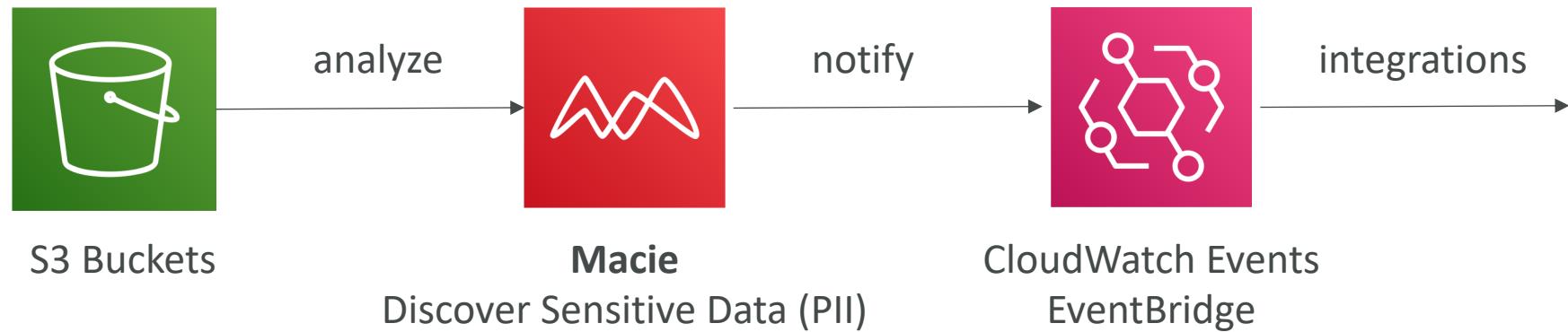


- View CloudTrail API calls if enabled

# Amazon Macie



- Amazon Macie is a fully managed data security and data privacy service that uses machine learning and pattern matching to discover and protect your sensitive data in AWS.
- Macie helps identify and alert you to sensitive data, such as personally identifiable information (PII)

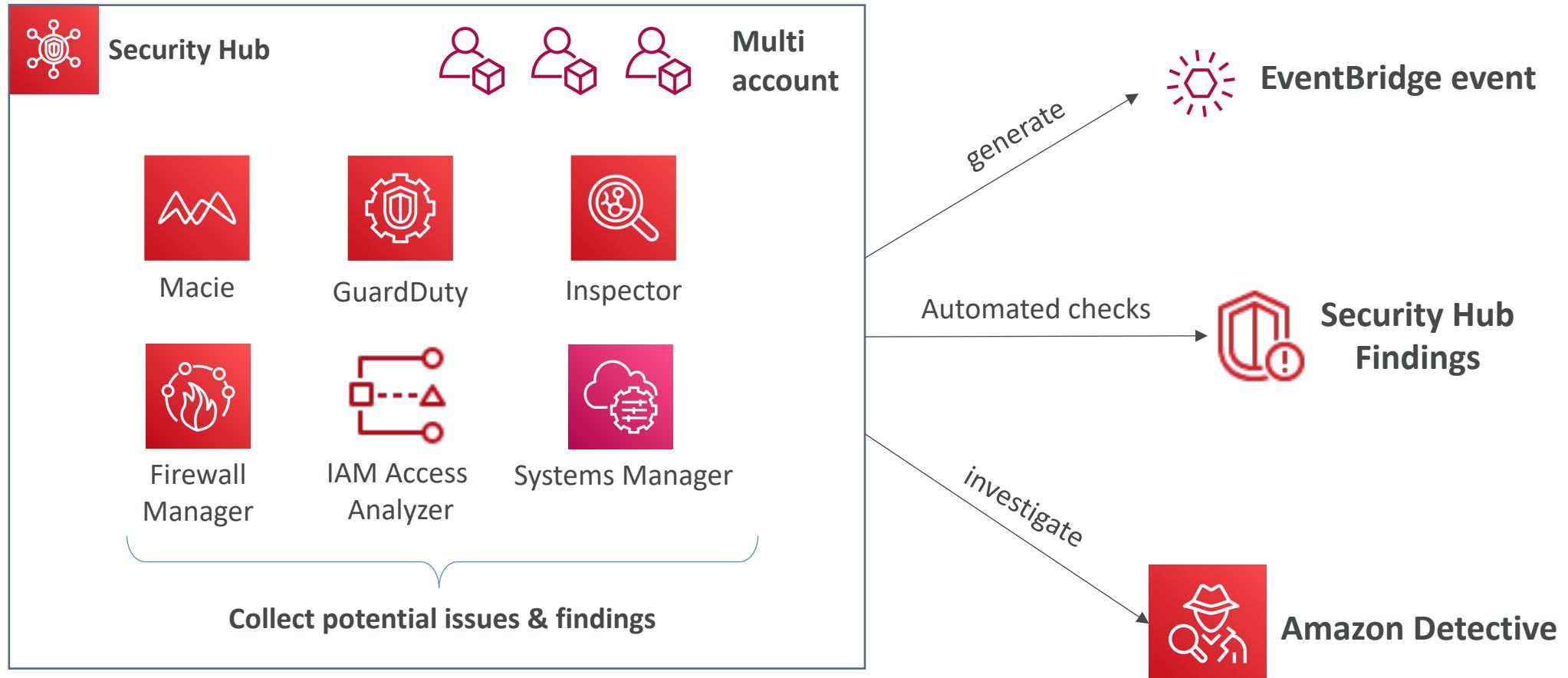




# AWS Security Hub

- Central security tool to manage security across several AWS accounts and automate security checks
- Integrated dashboards showing current security and compliance status to quickly take actions
- Automatically aggregates alerts in predefined or personal findings formats from various AWS services & AWS partner tools:
  - GuardDuty
  - Inspector
  - Macie
  - IAM Access Analyzer
  - AWS Systems Manager
  - AWS Firewall Manager
  - AWS Partner Network Solutions
- Must first enable the AWS Config Service

# AWS Security Hub



# Amazon Detective

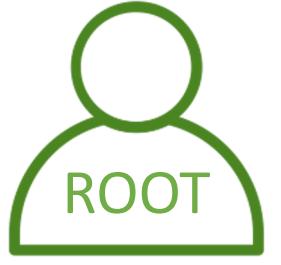


- GuardDuty, Macie, and Security Hub are used to identify potential security issues, or findings
- Sometimes security findings require deeper analysis to isolate the root cause and take action – it's a complex process
- Amazon Detective analyzes, investigates, and quickly identifies the root cause of security issues or suspicious activities (using ML and graphs)
- Automatically collects and processes events from VPC Flow Logs, CloudTrail, GuardDuty and create a unified view
- Produces visualizations with details and context to get to the root cause

# AWS Abuse



- Report suspected AWS resources used for abusive or illegal purposes
- Abusive & prohibited behaviors are:
  - **Spam** – receiving undesired emails from AWS-owned IP address, websites & forums spammed by AWS resources
  - **Port scanning** – sending packets to your ports to discover the unsecured ones
  - **DoS or DDoS attacks** – AWS-owned IP addresses attempting to overwhelm or crash your servers/softwares
  - **Intrusion attempts** – logging in on your resources
  - **Hosting objectionable or copyrighted content** – distributing illegal or copyrighted content without consent
  - **Distributing malware** – AWS resources distributing softwares to harm computers or machines
- Contact the AWS Abuse team: [AWS abuse form](#), or [abuse@amazonaws.com](mailto:abuse@amazonaws.com)



# Root user privileges

- Root user = Account Owner (created when the account is created)
- Has complete access to all AWS services and resources
- **Lock away your AWS account root user access keys!**
- Do not use the root account for everyday tasks, even administrative tasks
- Actions that can be performed only by the root user:
  - Change account settings (account name, email address, root user password, root user access keys)
  - View certain tax invoices
  - Close your AWS account
  - Restore IAM user permissions
  - Change or cancel your AWS Support plan
  - Register as a seller in the Reserved Instance Marketplace
  - Configure an Amazon S3 bucket to enable MFA
  - Edit or delete an Amazon S3 bucket policy that includes an invalid VPC ID or VPC endpoint ID
  - Sign up for GovCloud

# Section Summary: Security & Compliance

- Shared Responsibility on AWS
- **Shield:** Automatic DDoS Protection + 24/7 support for advanced
- **WAF:** Firewall to filter incoming requests based on rules
- **KMS:** Encryption keys managed by AWS
- **CloudHSM:** Hardware encryption, we manage encryption keys
- **AWS Certificate Manager:** provision, manage, and deploy SSL/TLS Certificates
- **Artifact:** Get access to compliance reports such as PCI, ISO, etc...
- **GuardDuty:** Find malicious behavior with VPC, DNS & CloudTrail Logs
- **Inspector:** For EC2 only, install agent and find vulnerabilities

# Section Summary: Security & Compliance

- **Config:** Track config changes and compliance against rules
- **Macie:** Find sensitive data (ex: PII data) in Amazon S3 buckets
- **CloudTrail:** Track API calls made by users within account
- **AWS Security Hub:** gather security findings from multiple AWS accounts
- **Amazon Detective:** find the root cause of security issues or suspicious activities
- **AWS Abuse:** Report AWS resources used for abusive or illegal purposes
- **Root user privileges:**
  - Change account settings
  - Close your AWS account
  - Change or cancel your AWS Support plan
  - Register as a seller in the Reserved Instance Marketplace

# Machine Learning Section

# Amazon Rekognition



- Find objects, people, text, scenes in images and videos using ML
- Facial analysis and facial search to do user verification, people counting
- Create a database of “familiar faces” or compare against celebrities
- Use cases:
  - Labeling
  - Content Moderation
  - Text Detection
  - Face Detection and Analysis (gender, age range, emotions...)
  - Face Search and Verification
  - Celebrity Recognition
  - Pathing (ex: for sports game analysis)

<https://aws.amazon.com/rekognition/>

# Amazon Transcribe



- Automatically convert speech to text
- Uses a **deep learning process** called **automatic speech recognition (ASR)** to convert speech to text quickly and accurately
- Use cases:
  - transcribe customer service calls
  - automate closed captioning and subtitling
  - generate metadata for media assets to create a fully searchable archive



*"Hello my name is Stéphane.  
I hope you're enjoying the course!"*

# Amazon Polly



- Turn text into lifelike speech using deep learning
- Allowing you to create applications that talk

*Hi! My name is Stéphane  
and this is a demo of Amazon Polly*



# Amazon Translate



- Natural and accurate language translation
- Amazon Translate allows you to **localize content** - such as websites and applications - for **international users**, and to easily translate large volumes of text efficiently.

Source language

Auto (auto) ▾

Hi my name is Stéphane

Target language

French (fr) ▾

Bonjour, je m'appelle Stéphane.

Portuguese (pt) ▾

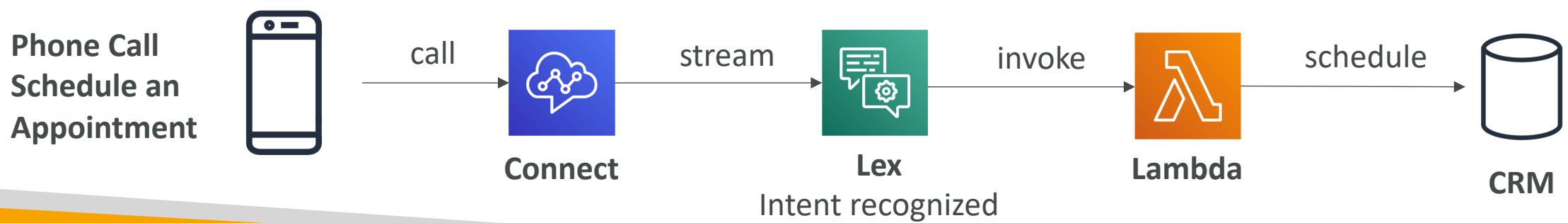
Oi, meu nome é Stéphane.

Hindi (hi) ▾

हाय मेरा नाम स्टीफन है

# Amazon Lex & Connect

- **Amazon Lex:** (same technology that powers Alexa)
  - Automatic Speech Recognition (ASR) to convert speech to text
  - Natural Language Understanding to recognize the intent of text, callers
  - Helps build chatbots, call center bots
- **Amazon Connect:**
  - Receive calls, create contact flows, cloud-based virtual contact center
  - Can integrate with other CRM systems or AWS
  - No upfront payments, 80% cheaper than traditional contact center solutions

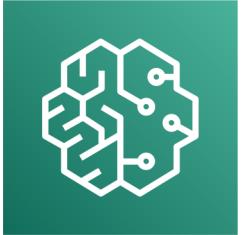




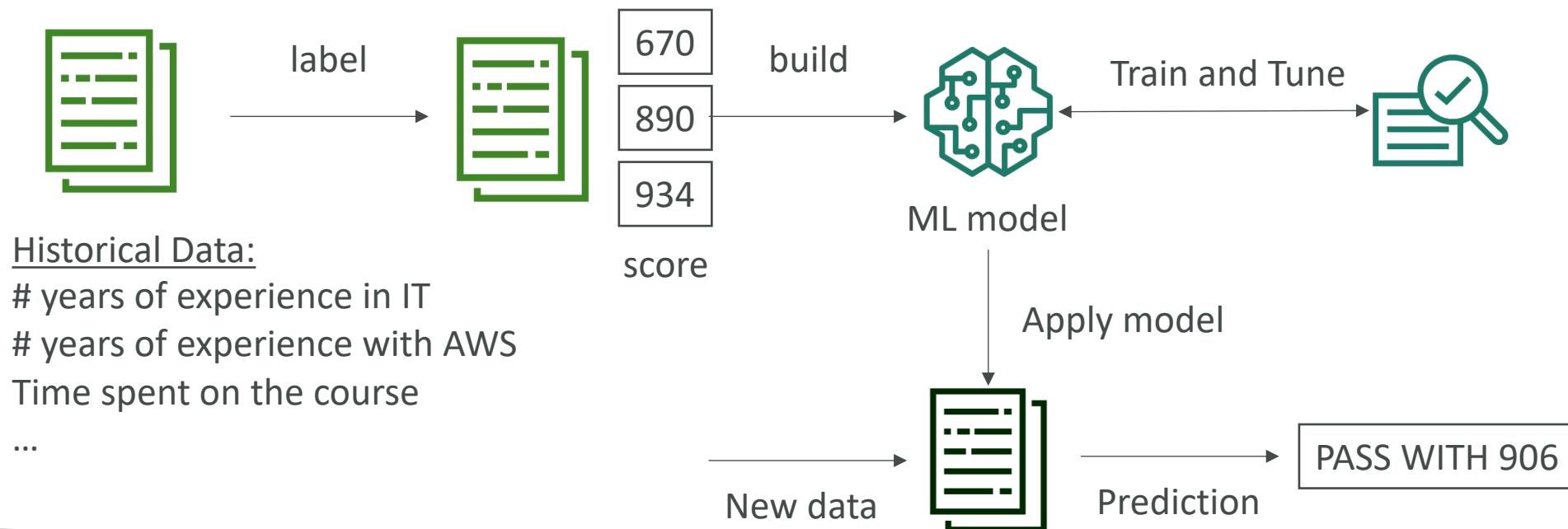
# Amazon Comprehend

- For Natural Language Processing – NLP
- Fully managed and serverless service
- Uses machine learning to find insights and relationships in text
  - Language of the text
  - Extracts key phrases, places, people, brands, or events
  - Understands how positive or negative the text is
  - Analyzes text using tokenization and parts of speech
  - Automatically organizes a collection of text files by topic
- Sample use cases:
  - analyze customer interactions (emails) to find what leads to a positive or negative experience
  - Create and groups articles by topics that Comprehend will uncover

# Amazon SageMaker



- Fully managed service for developers / data scientists to build ML models
- Typically, difficult to do all the processes in one place + provision servers
- Machine learning process (simplified): predicting your exam score



# Amazon Forecast



- Fully managed service that uses ML to deliver highly accurate forecasts
- Example: predict the future sales of a raincoat
- 50% more accurate than looking at the data itself
- Reduce forecasting time from months to hours
- Use cases: Product Demand Planning, Financial Planning, Resource Planning, ...

## Historical Time-series Data:

Product features

Prices

Discounts

Website traffic

Store locations

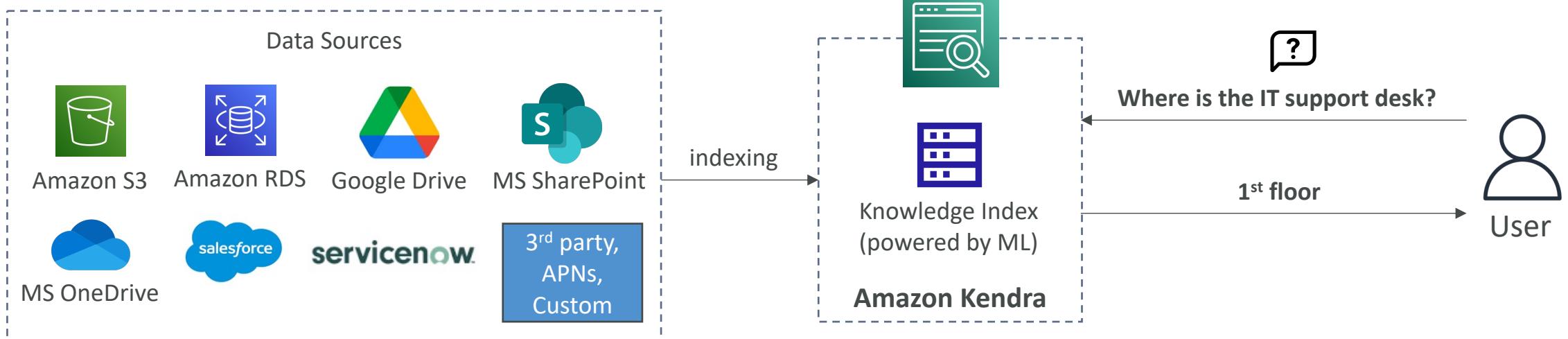
...



# Amazon Kendra



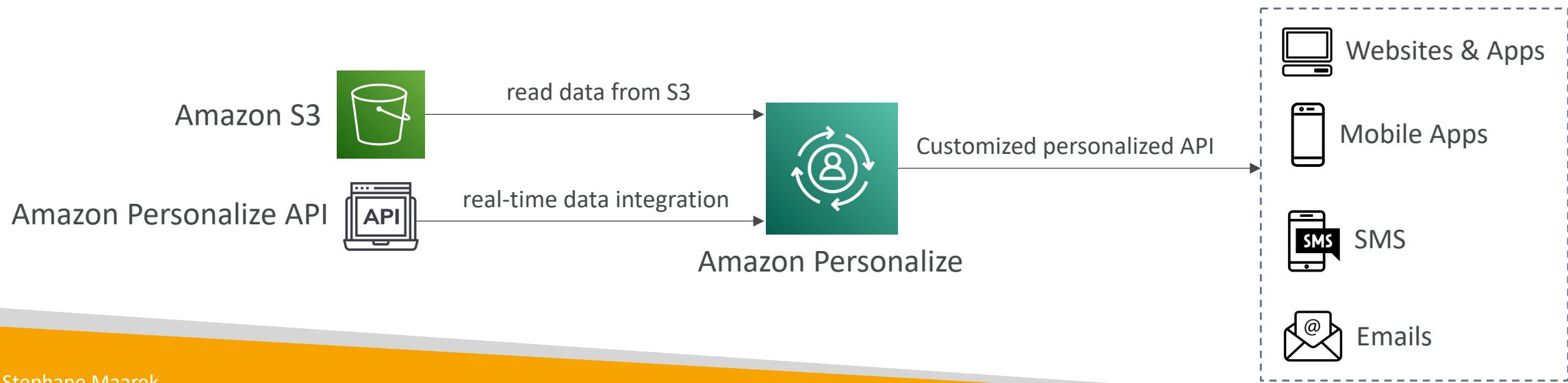
- Fully managed **document search service** powered by Machine Learning
- Extract answers from within a document (text, pdf, HTML, PowerPoint, MS Word, FAQs...)
- Natural language search capabilities
- Learn from user interactions/feedback to promote preferred results (**Incremental Learning**)
- Ability to manually fine-tune search results (importance of data, freshness, custom, ...)



# Amazon Personalize



- Fully managed ML-service to build apps with real-time personalized recommendations
- Example: personalized product recommendations/re-ranking, customized direct marketing
  - Example: User bought gardening tools, provide recommendations on the next one to buy
- Same technology used by Amazon.com
- Integrates into existing websites, applications, SMS, email marketing systems, ...
- Implement in days, not months (you don't need to build, train, and deploy ML solutions)
- Use cases: retail stores, media and entertainment...

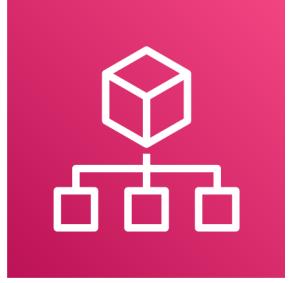


# AWS Machine Learning - Summary

- **Rekognition:** face detection, labeling, celebrity recognition
- **Transcribe:** audio to text (ex: subtitles)
- **Polly:** text to audio
- **Translate:** translations
- **Lex:** build conversational bots – chatbots
- **Connect:** cloud contact center
- **Comprehend:** natural language processing
- **SageMaker:** machine learning for every developer and data scientist
- **Forecast:** build highly accurate forecasts
- **Kendra:** ML-powered search engine
- **Personalize:** real-time personalized recommendations

# Account Management, Billing & Support Section

# AWS Organizations



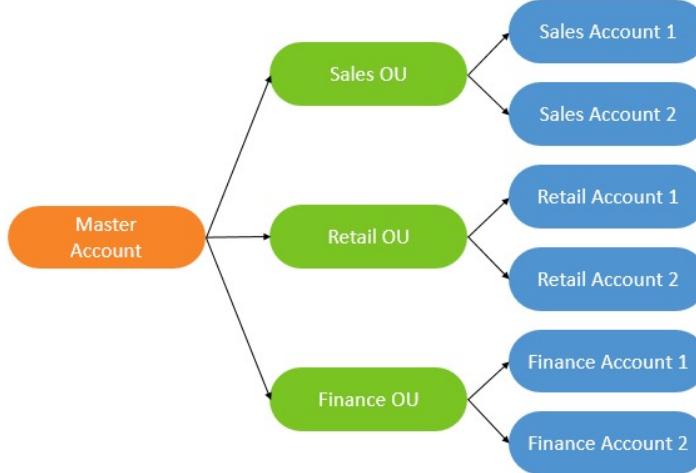
- Global service
- Allows to manage **multiple AWS accounts**
- The main account is the master account
- Cost Benefits:
  - Consolidated Billing across all accounts - single payment method
  - Pricing benefits from aggregated usage (volume discount for EC2, S3...)
  - Pooling of Reserved EC2 instances for optimal savings
- API is available to automate AWS account creation
- Restrict account privileges using Service Control Policies (SCP)

# Multi Account Strategies

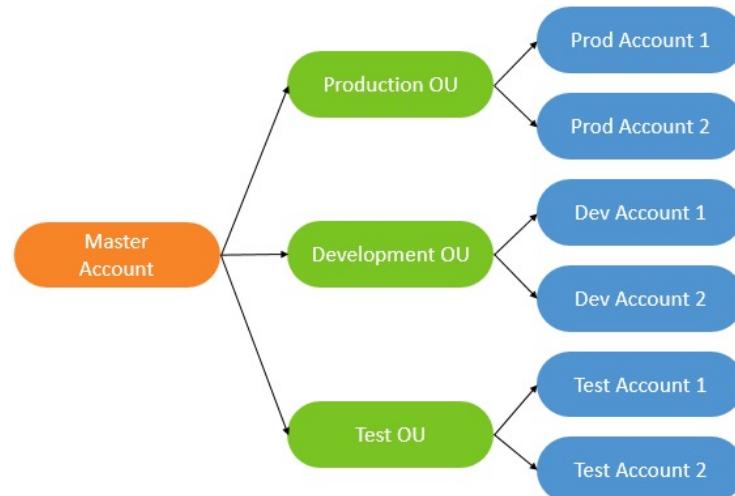
- Create accounts per department, per cost center, per dev / test / prod, based on regulatory restrictions (using SCP), for better resource isolation (ex:VPC), to have separate per-account service limits, isolated account for logging
- Multi Account vs One Account Multi VPC
- Use tagging standards for billing purposes
- Enable CloudTrail on all accounts, send logs to central S3 account
- Send CloudWatch Logs to central logging account

# Organizational Units (OU) - Examples

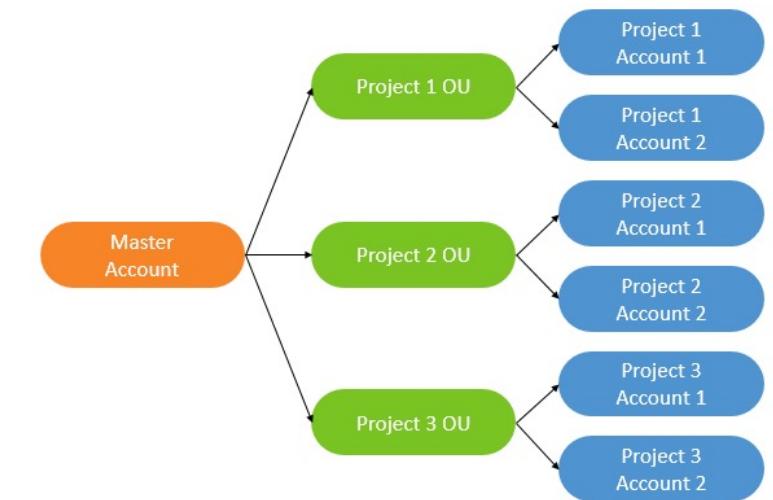
**Business Unit**



**Environmental Lifecycle**

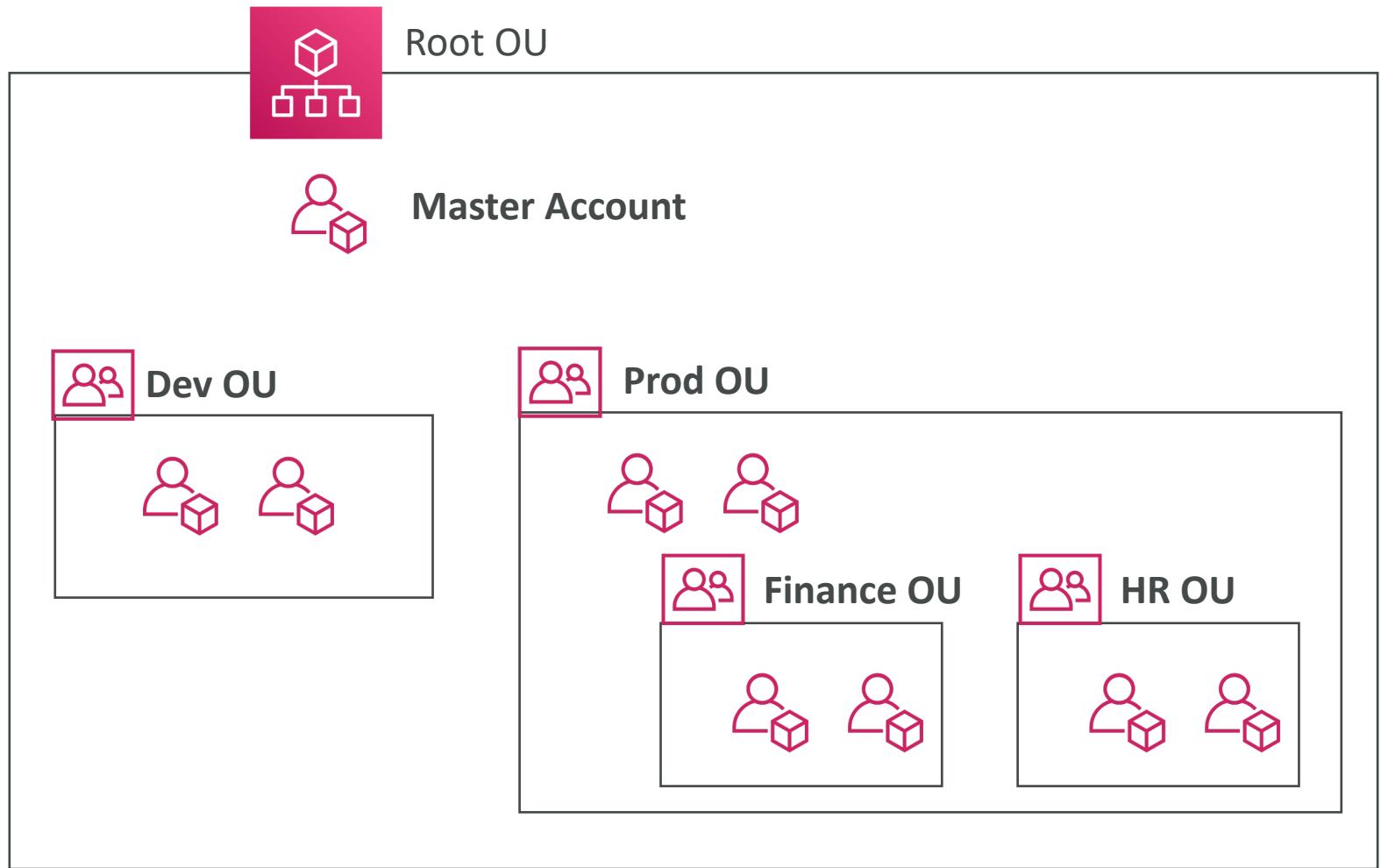


**Project-based**



<https://aws.amazon.com/answers/account-management/aws-multi-account-billing-strategy/>

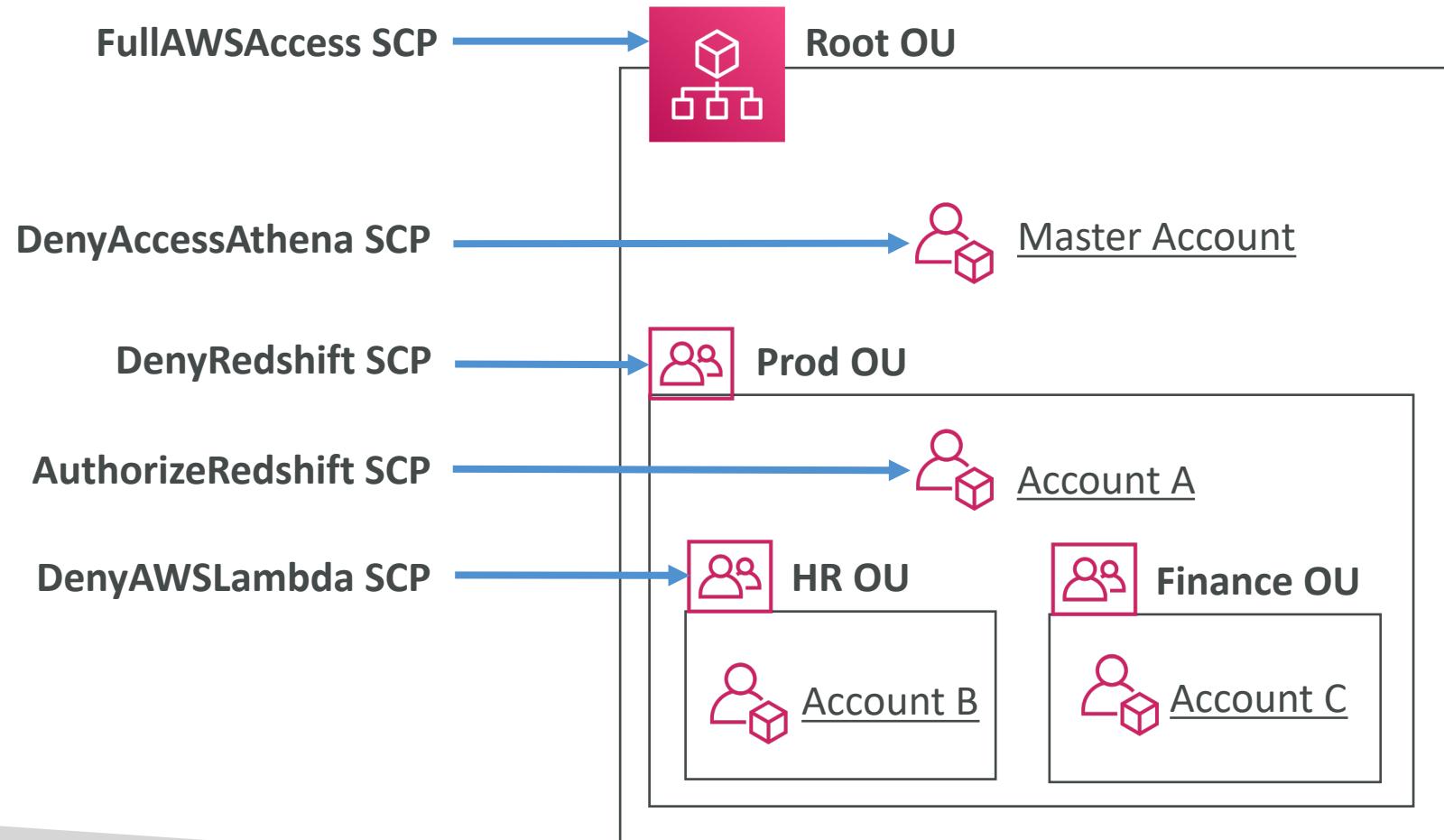
# AWS Organization



# Service Control Policies (SCP)

- Whitelist or blacklist IAM actions
- Applied at the **OU** or **Account** level
- Does not apply to the Master Account
- SCP is applied to all the **Users** and **Roles** of the Account, including Root user
- The SCP does not affect service-linked roles
  - Service-linked roles enable other AWS services to integrate with AWS Organizations and can't be restricted by SCPs.
- SCP must have an explicit Allow (does not allow anything by default)
- Use cases:
  - Restrict access to certain services (for example: can't use EMR)
  - Enforce PCI compliance by explicitly disabling services

# SCP Hierarchy



- Master Account
  - Can do anything
  - (no SCP apply)
- Account A
  - Can do anything
  - EXCEPT access Redshift (explicit Deny from Prod OU)
- Account B
  - Can do anything
  - EXCEPT access Redshift (explicit Deny from Prod OU)
  - EXCEPT access Lambda (explicit Deny from HR OU)
- Account C
  - Can do anything
  - EXCEPT access Redshift (explicit Deny from Prod OU)

# SCP Examples

## Blacklist and Whitelist strategies

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowsAllActions",  
            "Effect": "Allow",  
            "Action": "*",  
            "Resource": "*"  
        },  
        {  
            "Sid": "DenyDynamoDB",  
            "Effect": "Deny",  
            "Action": "dynamodb:*",  
            "Resource": "*"  
        }  
    ]  
}
```

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

More examples: [https://docs.aws.amazon.com/organizations/latest/userguide/orgs\\_manage\\_policies\\_example-scps.html](https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_example-scps.html)

# AWS Control Tower



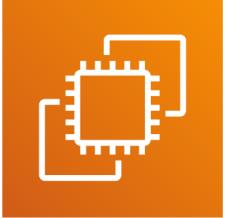
- Easy way to set up and govern a secure and compliant multi-account AWS environment based on best practices
- Benefits:
  - Automate the set up of your environment in a few clicks
  - Automate ongoing policy management using guardrails
  - Detect policy violations and remediate them
  - Monitor compliance through an interactive dashboard
- AWS Control Tower runs on top of AWS Organizations:
  - It automatically sets up AWS Organizations to organize accounts and implement SCPs (Service Control Policies)

# Pricing Models in AWS

- AWS has 4 pricing models:
- **Pay as you go:** pay for what you use, remain agile, responsive, meet scale demands
- **Save when you reserve:** minimize risks, predictably manage budgets, comply with long-terms requirements
  - Reservations are available for EC2 Reserved Instances, DynamoDB Reserved Capacity, ElastiCache Reserved Nodes, RDS Reserved Instance, Redshift Reserved Nodes
- **Pay less by using more:** volume-based discounts
- **Pay less as AWS grows**

# Free services & free tier in AWS

- IAM
  - VPC
  - Consolidated Billing
  - Elastic Beanstalk
  - CloudFormation
  - Auto Scaling Groups
  - Free Tier: <https://aws.amazon.com/free/>
    - EC2 t2.micro instance for a year
    - S3, EBS, ELB, AWS Data transfer
- 
- ⚠ You do pay for the resources created**



# Compute Pricing – EC2

- Only charged for what you use
- Number of instances
- Instance configuration:
  - Physical capacity
  - Region
  - OS and software
  - Instance type
  - Instance size
- ELB running time and amount of data processed
- Detailed monitoring

# Compute Pricing – EC2

- **On-demand instances:**
  - Minimum of 60s
  - Pay per second (Linux) or per hour (Windows)
- **Reserved instances:**
  - Up to 75% discount compared to On-demand on hourly rate
  - 1- or 3-years commitment
  - All upfront, partial upfront, no upfront
- **Spot instances:**
  - Up to 90% discount compared to On-demand on hourly rate
  - Bid for unused capacity
- **Dedicated Host:**
  - On-demand
  - Reservation for 1 year or 3 years commitment
- **Savings plans** as an alternative to save on sustained usage

# Compute Pricing – Lambda & ECS

- Lambda:

- Pay per call
- Pay per duration



- ECS:

- EC2 Launch Type Model: No additional fees, you pay for AWS resources stored and created in your application



- Fargate:

- Fargate Launch Type Model: Pay for vCPU and memory resources allocated to your applications in your containers



# Storage Pricing – S3



- **Storage class:** S3 Standard, S3 Infrequent Access, S3 One-Zone IA, S3 Intelligent Tiering, S3 Glacier and S3 Glacier Deep Archive
- Number and size of objects: Price can be tiered (based on volume)
- Number and type of requests
- Data transfer OUT of the S3 region
- S3 Transfer Acceleration
- Lifecycle transitions
  
- Similar service: EFS (pay per use, has infrequent access & lifecycle rules)

# Storage Pricing - EBS



- Volume type (based on performance)
- Storage volume in GB per month provisionned
- IOPS:
  - General Purpose SSD: Included
  - Provisioned IOPS SSD: Provisionned amount in IOPS
  - Magnetic: Number of requests
- Snapshots:
  - Added data cost per GB per month
- Data transfer:
  - Outbound data transfer are tiered for volume discounts
  - Inbound is free

# Database Pricing - RDS



- Per hour billing
- Database characteristics:
  - Engine
  - Size
  - Memory class
- Purchase type:
  - On-demand
  - Reserved instances (1 or 3 years) with required up-front
- Backup Storage: There is no additional charge for backup storage up to 100% of your total database storage for a region.

# Database Pricing - RDS



- Additional storage (per GB per month)
- Number of input and output requests per month
- Deployment type (storage and I/O are variable):
  - Single AZ
  - Multiple AZs
- Data transfer:
  - Outbound data transfer are tiered for volume discounts
  - Inbound is free

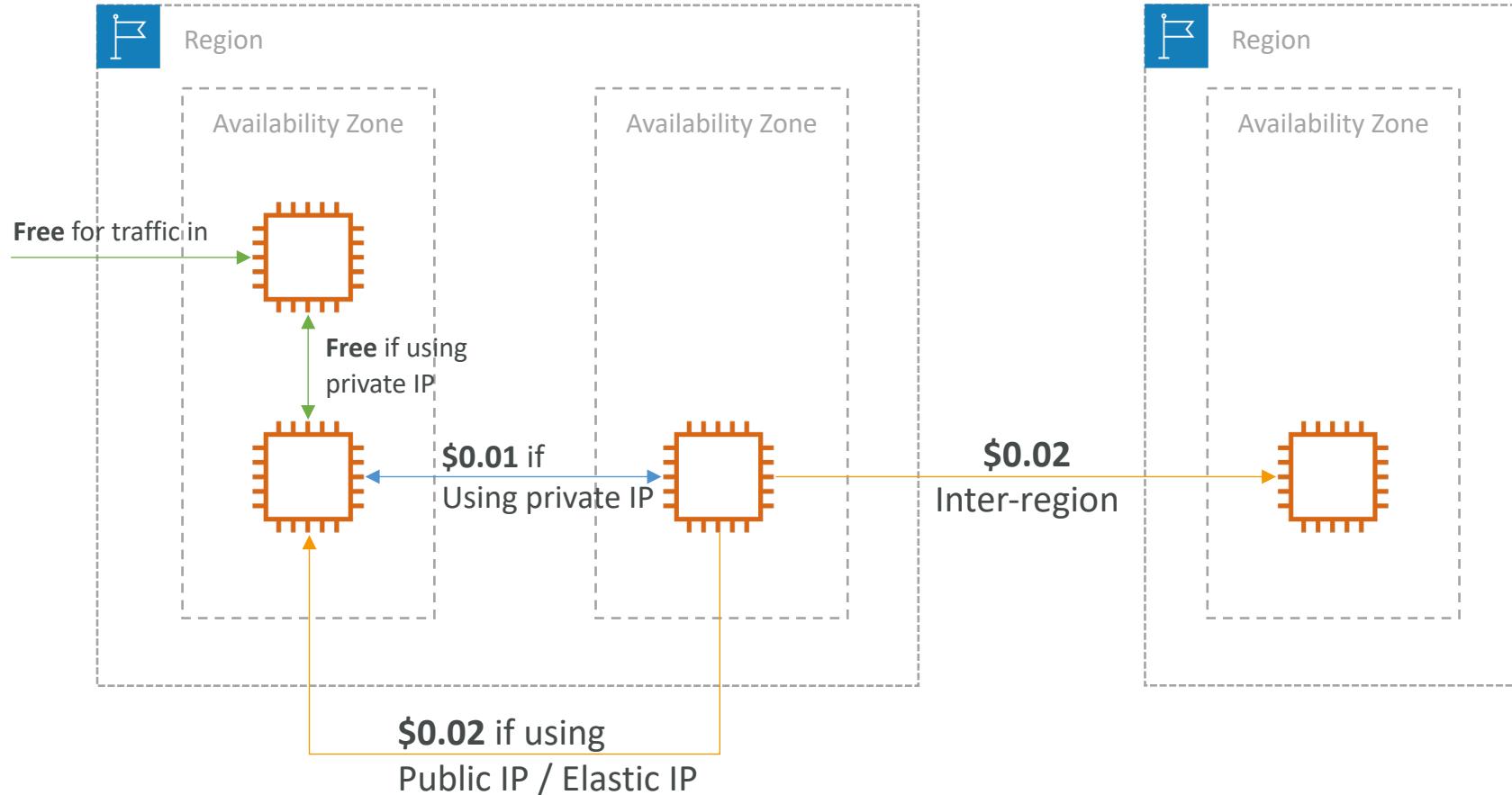


# Content Delivery – CloudFront

- Pricing is different across different geographic regions
- Aggregated for each edge location, then applied to your bill
- Data Transfer Out (volume discount)
- Number of HTTP/HTTPS requests

Per Month	United States & Canada	Europe & Israel	South Africa, Kenya, & Middle East	South America	Japan	Australia	Singapore, South Korea, Taiwan, Hong Kong, & Philippines	India
First 10TB	\$0.085	\$0.085	\$0.110	\$0.110	\$0.114	\$0.114	\$0.140	\$0.170
Next 40TB	\$0.080	\$0.080	\$0.105	\$0.105	\$0.089	\$0.098	\$0.135	\$0.130
Next 100TB	\$0.060	\$0.060	\$0.090	\$0.090	\$0.086	\$0.094	\$0.120	\$0.110

# Networking Costs in AWS per GB - Simplified



- Use Private IP instead of Public IP for good savings and better network performance
- Use same AZ for maximum savings (at the cost of high availability)



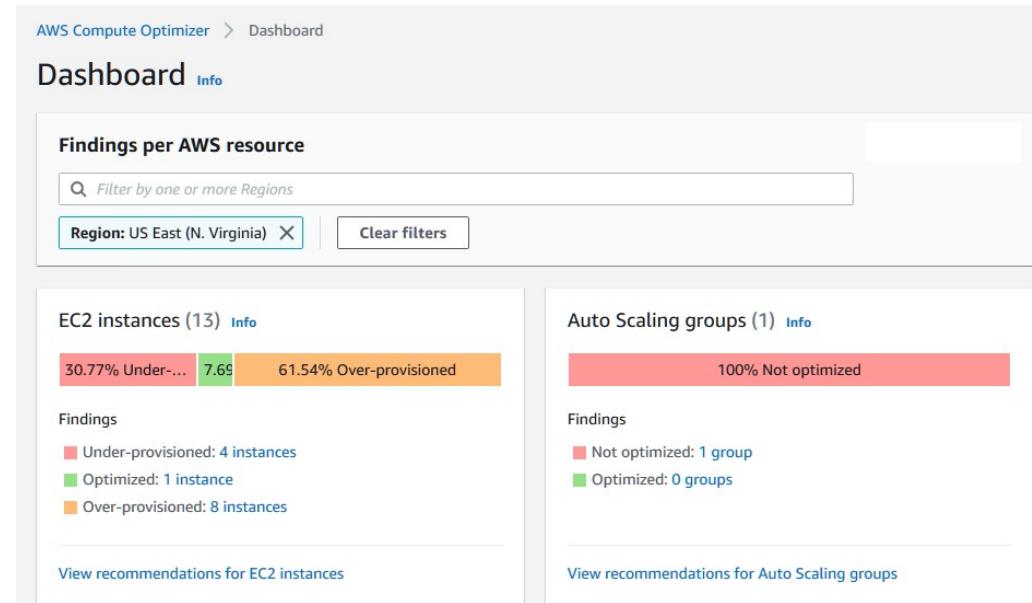
# Savings Plan

- Commit a certain \$ amount per hour for 1 or 3 years
- Easiest way to setup long-term commitments on AWS
- **EC2 Savings Plan**
  - Up to 72% discount compared to On-Demand
  - Commit to usage of individual instance families in a region (e.g. C5 or M5)
  - Regardless of AZ, size (m5.xl to m5.4xl), OS (Linux/Windows) or tenancy
  - All upfront, partial upfront, no upfront
- **Compute Savings Plan**
  - Up to 66% discount compared to On-Demand
  - Regardless of Family, Region, size, OS, tenancy, compute options
  - Compute Options: EC2, Fargate, Lambda
- Setup from the AWS Cost Explorer console
- Estimate pricing at <https://aws.amazon.com/savingsplans/pricing/>

# AWS Compute Optimizer



- Reduce costs and improve performance by recommending optimal AWS resources for your workloads
- Helps you choose optimal configurations and right-size your workloads (over/under provisioned)
- Uses Machine Learning to analyze your resources' configurations and their utilization CloudWatch metrics
- Supported resources
  - EC2 instances
  - EC2 Auto Scaling Groups
  - EBS volumes
  - Lambda functions
- Lower your costs by up to 25%
- Recommendations can be exported to S3



# Billing and Costing Tools

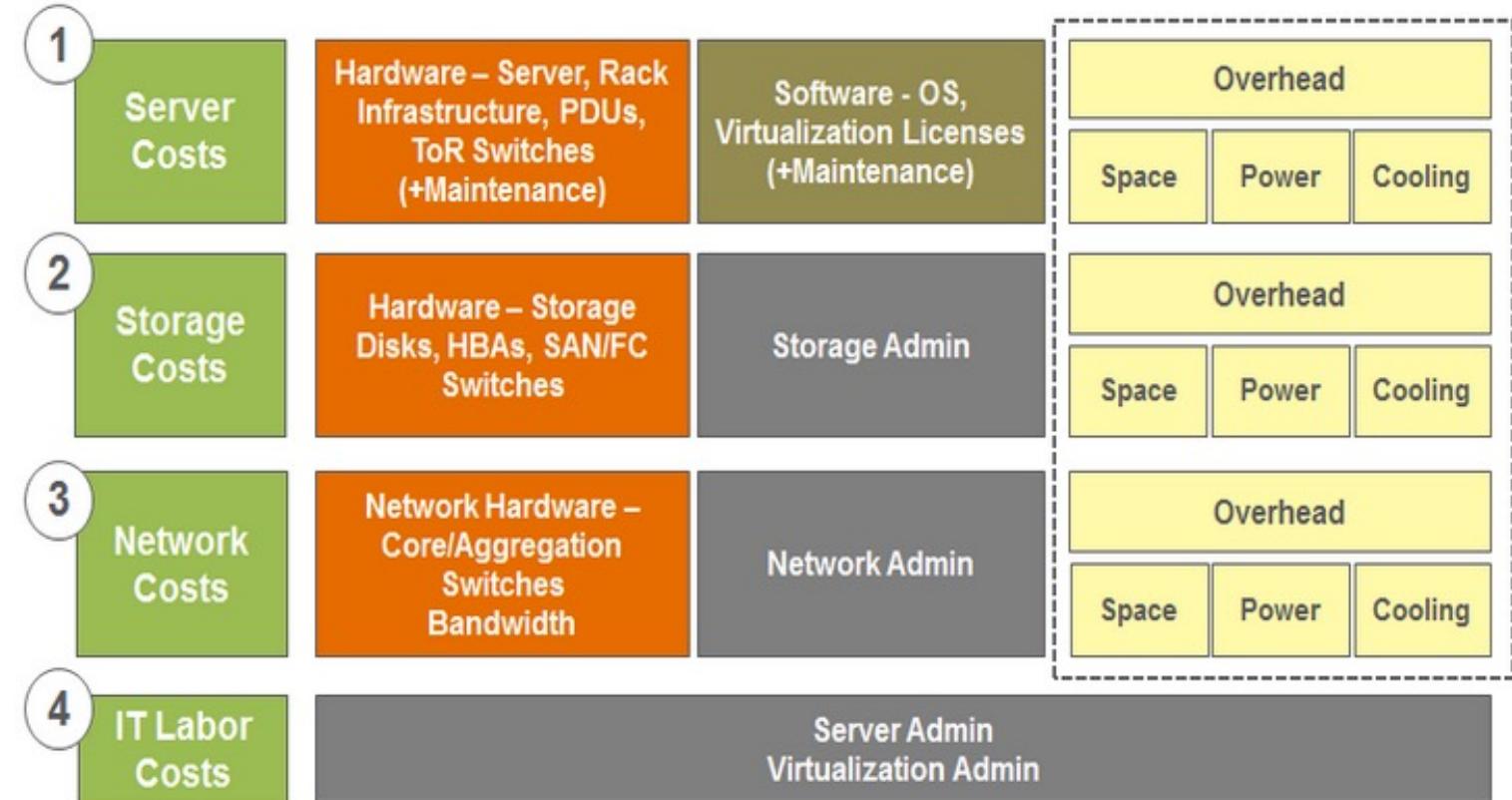


- Estimating costs in the cloud:
  - TCO Calculator
  - Simple Monthly Calculator / Pricing Calculator
- Tracking costs in the cloud:
  - Billing Dashboard
  - Cost Allocation Tags
  - Cost and Usage Reports
  - Cost Explorer
- Monitoring against costs plans:
  - Billing Alarms
  - Budgets

# AWS Total Cost of Ownership (TCO) Calculators

- AWS helps you reduce Total Cost of Ownership (TCO) by **reducing the need to invest in large capital expenditures** and providing a **pay-as-you-go model**
- The TCO calculators allow you to **estimate the cost savings** when using AWS and provide a detailed set of reports that can be used in **executive presentations**.
- **Compare** the cost of your applications in an **on-premises** or traditional hosting environment **to AWS**: Server, Storage, Network, IT Labor
- <https://awstcocalculator.com/>

# Points of comparison of On-Premises vs AWS



<https://aws.amazon.com/blogs/aws/the-new-aws-tco-calculator/>

# TCO Example

Select Currency

What type of environment are you comparing against?  On-Premises  Colocation

Which AWS region is ideal for your geo requirements?

Servers

Are you comparing physical servers or virtual machines?  Physical Servers  Virtual Machines

Provide your configuration details:

App. Name <i>i</i>	Number of VMs <i>i</i>	CPU Cores <i>i</i>	Memory(GB) <i>i</i>	Server Type <i>i</i>	Guest OS <i>i</i>	Hypervisor <i>i</i>	DB Engine <i>i</i>	
demo app	10	16	32	Non DB <i>▼</i>	Linux <i>▼</i>	VMware <i>▼</i>		X
database	1	4	16	DB <i>▼</i>	Linux <i>▼</i>	VMware <i>▼</i>	SQL Server Stat	X

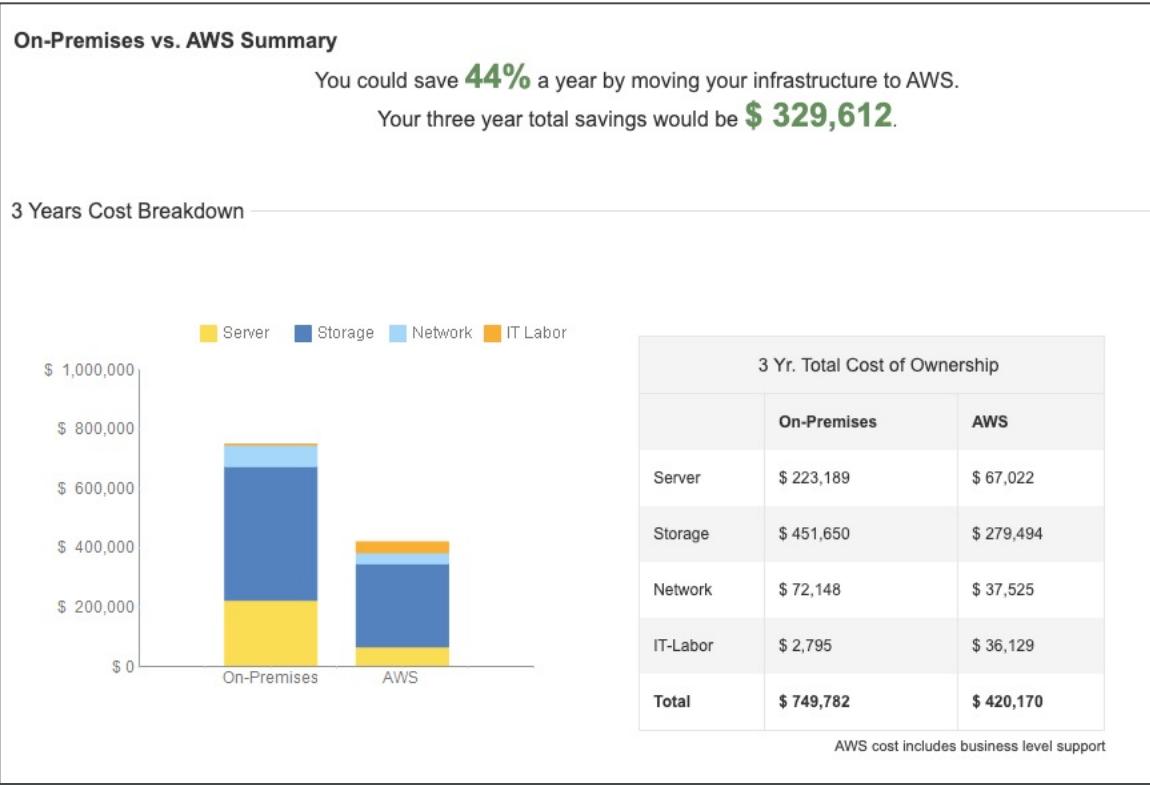
Total no.of VMs: 11

Storage

Provide your storage footprint details

Storage Type <i>i</i>	Raw Storage Capacity <i>i</i>	% Accessed Infrequently <i>i</i>	Disk Type <i>i</i>
SAN <i>▼</i>	300 <i>TB</i> <i>▼</i>		SSD <i>▼</i>

# TCO Result



## PDF Report (25 pages)



### Total Cost of Ownership (TCO) Comparison

This report includes a total cost of ownership (TCO) comparison between running your application in an on-premises or colocation infrastructure and AWS. The on-premises/colocation infrastructure is based on the description you provided in the online tool. The AWS infrastructure is an approximation of the infrastructure you described. These calculations use third-party estimates and assumptions. This calculator provides an estimate of usage charges for AWS services based on certain information you provide. Your monthly charges will be based on your actual usage of AWS services and may vary from the estimates the calculator has provided.

### Notices

© 2014 Amazon Web Services, Inc. This report is provided for informational purposes only. Amazon Web Services, Inc. is not responsible for any damages related to the information in this report, which is provided "as is" without warranty of any kind, whether express, implied, or statutory. Nothing in this report creates any warranties or representations from Amazon Web Services, Inc., its affiliates, suppliers, or licensors. This report does not modify the applicable terms and conditions governing your use of Amazon Web Services technologies, including the Amazon Web Services website. This report represents Amazon Web Services' current product offerings as of the date of issue and are subject to change without notice.

# Simple Monthly Calculator / Pricing Calculator

- Note: deprecated service (June 30<sup>th</sup> 2020)
- Replaced by AWS Pricing Calculator <https://calculator.aws/>
- Estimate the cost for your architecture solution.

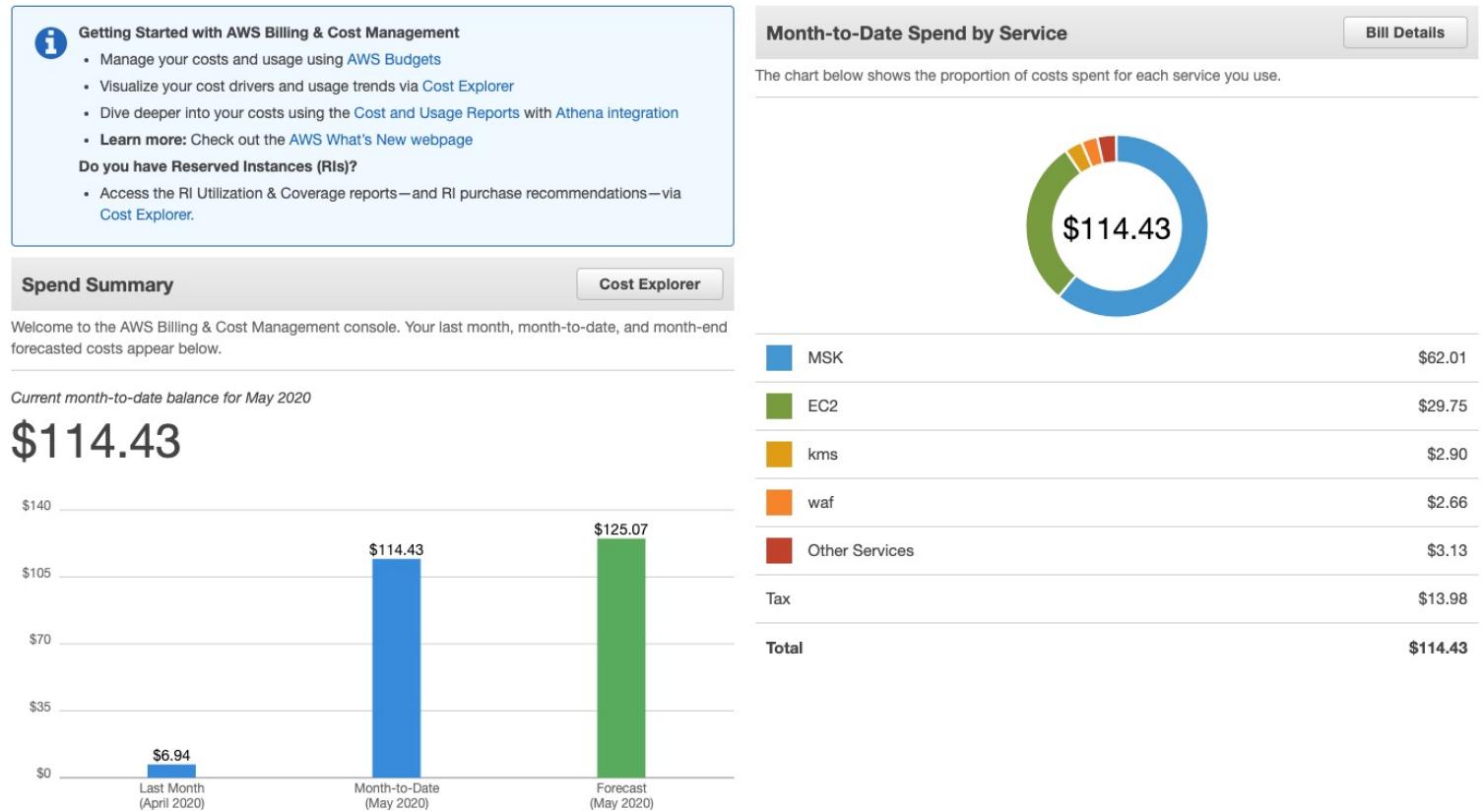
The screenshot shows the AWS Pricing Calculator interface. At the top, it displays the total estimated cost for the first 12 months: **62,191.68 USD**. Below this, under the heading "Services (2)", there are two items:

- Amazon Aurora MySQL-Compatible**: Region: US East (Ohio). Monthly cost: **5,110.80 USD**.
- Aurora MySQL-Compatible**: Change records per statement (0.38), (1 instances) db.r5.12xlarge Memory optimized OnDemand, Storage amount (300 GB). Monthly cost: **5,110.80 USD**.

At the bottom, there is a section for a "Quick estimate" for **Amazon EC2**, Region: US East (Ohio), with a monthly cost of **71.84 USD**.

# AWS Billing Dashboard

## Billing & Cost Management Dashboard



# AWS Free Tier Dashboard

All Free Tier services by usage

Service	Free Tier usage limit	Current usage	Forecasted usage	Month-to-date actual usage	Month-end forecasted usage
AWS Lambda	1,000,000 free requests per month for AWS Lambda	585,089 Requests	697,606 Requests	<div style="width: 58.51%;">58.51%</div>	<div style="width: 69.76%;">69.76%</div>
Amazon Simple Notification Service	1,000,000 Requests for Amazon Simple Notification Service (APS2)	575,640 Requests	686,340 Requests	<div style="width: 57.56%;">57.56%</div>	<div style="width: 68.63%;">68.63%</div>
AWS Lambda	400,000 seconds of compute time per month for AWS Lambda	61,973 seconds	73,891 seconds	<div style="width: 15.49%;">15.49%</div>	<div style="width: 18.47%;">18.47%</div>
AWS Key Management Service	20,000 free requests per month for AWS Key Management Service	1,533 Requests	1,828 Requests	<div style="width: 7.66%;">7.66%</div>	<div style="width: 9.14%;">9.14%</div>
AmazonCloudWatch	5 GB of Log Data Ingestion for Amazon Cloudwatch	0 GB	0 GB	<div style="width: 5.81%;">5.81%</div>	<div style="width: 6.92%;">6.92%</div>
AmazonCloudWatch	5 GB of Log Data Archive for Amazon Cloudwatch	0 GB-Mo	0 GB-Mo	<div style="width: 4.78%;">4.78%</div>	<div style="width: 5.70%;">5.70%</div>
Amazon Simple Notification Service	1,000 email notifications for Amazon Simple Notification Service (USE1)	25 Notifications	30 Notifications	<div style="width: 2.50%;">2.50%</div>	<div style="width: 2.98%;">2.98%</div>
Amazon Simple Queue Service	1,000,000 Requests of Amazon Simple Queue Service	11,323 Requests	13,501 Requests	<div style="width: 1.13%;">1.13%</div>	<div style="width: 1.35%;">1.35%</div>
CodeBuild	100 build minutes per month of build.general1.small compute type usage for AWS CodeBuild	1 minutes	1 minutes	<div style="width: 1.00%;">1.00%</div>	<div style="width: 1.19%;">1.19%</div>
AWS Step Functions	4,000 state transitions per month for AWS Step Functions	15 StateTransitions	18 StateTransitions	<div style="width: 0.38%;">0.38%</div>	<div style="width: 0.45%;">0.45%</div>

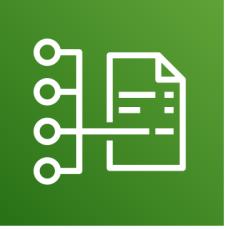
# Cost Allocation Tags

- Use cost allocation tags to track your AWS costs on a detailed level
- AWS generated tags
  - Automatically applied to the resource you create
  - Starts with Prefix aws: (e.g. aws: createdBy)
- User-defined tags
  - Defined by the user
  - Starts with Prefix user:

Total Cost	user:Owner	user:Stack	user:Cost Center	user:Application
0.95	DbAdmin	Test	80432	Widget2
0.01	DbAdmin	Test	80432	Widget2
3.84	DbAdmin	Prod	80432	Widget2
6.00	DbAdmin	Test	78925	Widget1
234.63	SysEng	Prod	78925	Widget1
0.73	DbAdmin	Test	78925	Widget1
0.00	DbAdmin	Prod	80432	Portal
2.47	DbAdmin	Prod	78925	Portal

# Tagging and Resource Groups

- Tags are used for organizing resources:
  - EC2: instances, images, load balancers, security groups...
  - RDS, VPC resources, Route 53, IAM users, etc...
  - Resources created by CloudFormation are all tagged the same way
- Free naming, common tags are: Name, Environment, Team ...
- Tags can be used to create **Resource Groups**
  - Create, maintain, and view a collection of resources that share common tags
  - Manage these tags using the Tag Editor



# Cost and Usage Reports

- Dive deeper into your AWS costs and usage
- The AWS Cost & Usage Report contains **the most comprehensive set of AWS cost and usage data available**, including additional metadata about AWS services, pricing, and reservations (e.g., Amazon EC2 Reserved Instances (RIs)).
- The AWS Cost & Usage Report lists AWS usage for each service category used by an account and its IAM users in hourly or daily line items, as well as any tags that you have activated for cost allocation purposes.
- Can be integrated with Athena, Redshift or QuickSight

# Cost and Usage Reports

M	N	O	P	R	S	T
lineItem/ProductCode	lineItem/UsageType	lineItem/Operation	lineItem/AvailabilityZone	lineItem/UsageAmount	lineItem/CurrencyCode	lineItem/LineItemDescription
1 AmazonEC2	CW:AlarmMonitorUsage	Unknown		0.00134409	USD	\$0.00 per alarm-month - first 10 alarms
2 AmazonS3	Requests-Tier1	ListAllMyBuckets		2	USD	\$0.00 per request - PUT, COPY, POST, or LIST requests under the monthly global free tier
4 AmazonEC2	CW:AlarmMonitorUsage	Unknown		0.00134409	USD	\$0.00 per alarm-month - first 10 alarms
5 AmazonEC2	APS2-EBS:VolumeUsage.gp2	CreateVolume-Gp2		0.01344086	USD	\$0.00 per GB-month of General Purpose (SSD) provisioned storage under monthly free tier
6 AmazonEC2	APS2-EBS:VolumeUsage.gp2	CreateVolume-Gp2		0.01344086	USD	\$0.00 per GB-month of General Purpose (SSD) provisioned storage under monthly free tier
7 AmazonEC2	USW2-BoxUsage:t2.micro	RunInstances:0002	us-west-2a	1	USD	\$0.00 per Windows t2.micro instance-hour (or partial hour) under monthly free tier
8 AmazonEC2	USW2-USE1-AWS-Out-Bytes	PublicIP-Out		0.00000174	USD	\$0.000 per GB - data transfer out under the monthly global free tier
9 AmazonEC2	USW2-USE1-AWS-In-Bytes	PublicIP-In		0.00000138	USD	\$0.00 per GB - US West (Oregon) data transfer from US East (Northern Virginia)
10 AmazonEC2	USW2-USW1-AWS-In-Bytes	PublicIP-In		0.00000149	USD	\$0.00 per GB - US West (Oregon) data transfer from US West (Northern California)
11 AmazonS3	Requests-Tier1	ListAllMyBuckets		2	USD	\$0.00 per request - PUT, COPY, POST, or LIST requests under the monthly global free tier
12 AmazonEC2	USW2-DataTransfer-Out-Bytes	RunInstances		0.00038144	USD	\$0.00 per GB - data transfer out under the monthly global free tier
13 AmazonEC2	USW2-USW1-AWS-Out-Bytes	PublicIP-Out		0.00000174	USD	\$0.000 per GB - data transfer out under the monthly global free tier
14 AmazonEC2	USW2-DataTransfer-In-Bytes	RunInstances		0.00030951	USD	\$0.00 per GB - data transfer in per month
15 AmazonEC2	USW2-BoxUsage:t2.micro	RunInstances:0002	us-west-2a	1	USD	\$0.00 per Windows t2.micro instance-hour (or partial hour) under monthly free tier
16 AmazonEC2	USW2-USW1-AWS-Out-Bytes	PublicIP-Out		0.00000349	USD	\$0.000 per GB - data transfer out under the monthly global free tier
17 AmazonEC2	USW2-USW1-AWS-In-Bytes	PublicIP-In		0.00000276	USD	\$0.00 per GB - US West (Oregon) data transfer from US West (Northern California)
18 AmazonEC2	APS2-EBS:VolumeUsage.gp2	CreateVolume-Gp2		0.01344086	USD	\$0.00 per GB-month of General Purpose (SSD) provisioned storage under monthly free tier
19 AmazonEC2	CW:AlarmMonitorUsage	Unknown		0.00134409	USD	\$0.00 per alarm-month - first 10 alarms
20 AmazonEC2	USW2-BoxUsage:t2.micro	RunInstances:0002	us-west-2a	1	USD	\$0.00 per Windows t2.micro instance-hour (or partial hour) under monthly free tier
21 AmazonEC2	USW2-DataTransfer-Regional-Bytes	PublicIP-Out		0.00000349	USD	\$0.000 per GB - regional data transfer under the monthly global free tier
22 AmazonEC2	USW2-DataTransfer-In-Bytes	RunInstances		0.00032071	USD	\$0.000 per GB - data transfer in per month
23 AmazonEC2	USW2-DataTransfer-Regional-Bytes	PublicIP-In		0.00000302	USD	\$0.000 per GB - regional data transfer under the monthly global free tier
24 AmazonEC2	USW2-USE1-AWS-Out-Bytes	PublicIP-Out		0.00000174	USD	\$0.000 per GB - data transfer out under the monthly global free tier
25 AmazonEC2	USW2-DataTransfer-Out-Bytes	RunInstances		0.00045736	USD	\$0.000 per GB - data transfer out under the monthly global free tier
26 AmazonEC2	USW2-DataTransfer-In-Bytes	RunInstances		0.00036737	USD	\$0.000 per GB - data transfer in per month
27 AmazonEC2	USW2-APN2-AWS-In-Bytes	PublicIP-In		0.00000005	USD	\$0.00 per GB - US West (Oregon) data transfer from Asia Pacific (Seoul)
28 AmazonEC2	USW2-APN2-AWS-Out-Bytes	PublicIP-Out		0.00000018	USD	\$0.000 per GB - data transfer out under the monthly global free tier
29 AmazonEC2	USW2-USE1-AWS-In-Bytes	PublicIP-In		0.00000153	USD	\$0.00 per GB - US West (Oregon) data transfer from US East (Northern Virginia)
30 AmazonEC2	USW2-DataTransfer-Out-Bytes	RunInstances		0.00039945	USD	\$0.000 per GB - data transfer out under the monthly global free tier
31 AmazonEC2	CW:AlarmMonitorUsage	Unknown		0.00134409	USD	\$0.00 per alarm-month - first 10 alarms

# Cost Explorer

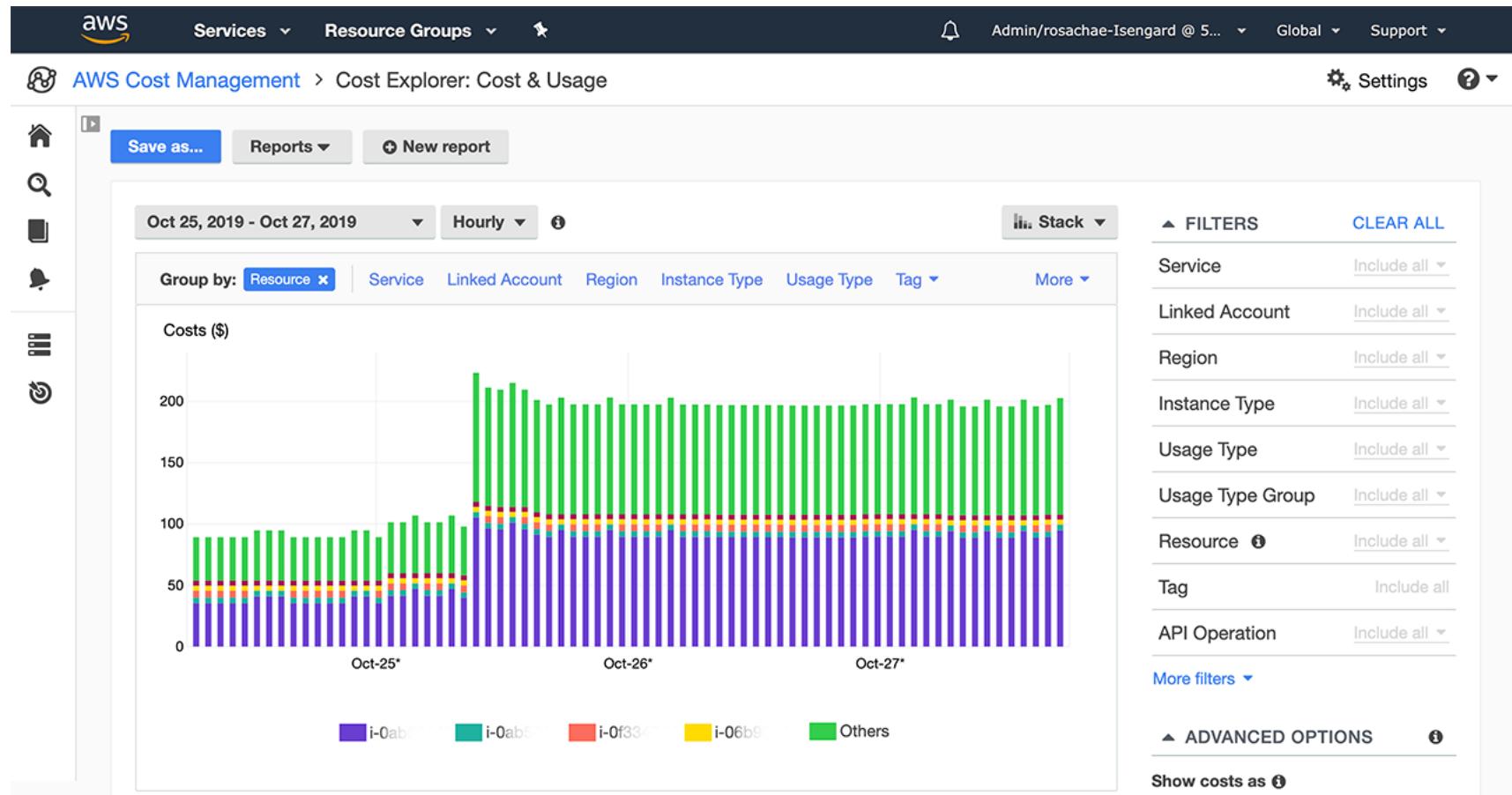


- Visualize, understand, and manage your AWS costs and usage over time
- Create custom reports that analyze cost and usage data.
- Analyze your data at a high level: total costs and usage across all accounts
- Or Monthly, hourly, resource level granularity
- Choose an optimal **Savings Plan** (to lower prices on your bill)
- Forecast usage up to 12 months based on previous usage

# Cost Explorer – Monthly Cost by AWS Service



# Cost Explorer– Hourly & Resource Level



# Cost Explorer – Savings Plan Alternative to Reserved Instances

Recommendation options

Savings Plans type <input checked="" type="radio"/> Compute <input type="radio"/> EC2 Instance	Savings Plans term <input type="radio"/> 1-year <input checked="" type="radio"/> 3-year	Payment option <input checked="" type="radio"/> All upfront <input type="radio"/> Partial upfront <input type="radio"/> No upfront	Based on the past <input type="radio"/> 7 days <input type="radio"/> 30 days <input checked="" type="radio"/> 60 days
--	---	---	--

Recommendation: Purchase a Compute Savings Plan at a commitment of \$2.40/hour

You could save an estimated **\$1,173** monthly by purchasing the recommended Compute Savings Plan.

Based on your past **60 days** of usage, we recommend purchasing a Savings Plan with a commitment of **\$2.40/hour** for a **3-year term**. With this commitment, we project that you could save an average of **\$1.61/hour** - representing a **40%** savings compared to On-Demand. To account for variable usage patterns, this recommendation maximizes your savings by leaving an average **\$0.04/hour** of On-Demand spend.

Before recommended purchase	After recommended purchase (based on your past 60 days of usage)
Monthly On-Demand spend <small> ⓘ</small> <b>\$2,955</b> (\$4.05/hour) Based on your On-Demand spend over the past 60 days	Estimated monthly spend <small> ⓘ</small> <b>\$1,782</b> (\$2.44/hour) Your recommended \$2.40/hour Savings Plans commitment + an average \$0.04/hour of On-Demand spend Estimated monthly savings <small> ⓘ</small> <b>\$1,173</b> (\$1.61/hour) 40% monthly savings over On-Demand \$2,955 - \$1,782 = \$1,173

This recommendation examines your usage over the past 60 days (including your existing Savings Plans and EC2 Reserved Instances) and calculates what your costs would have been had you purchased the recommended Savings Plans. See applicable rates for Savings Plans [here](#). To generate this recommendation, AWS simulates your bill for different commitment amounts and recommends the commitment amount that provides the greatest estimated savings. [Learn more](#)

Recommended Compute Savings Plans

x	Term	Payment option	Recommended commitment	Estimated hourly savings
<input checked="" type="checkbox"/>	3-year	All upfront	\$2.40/hour	\$1.61 (40%)

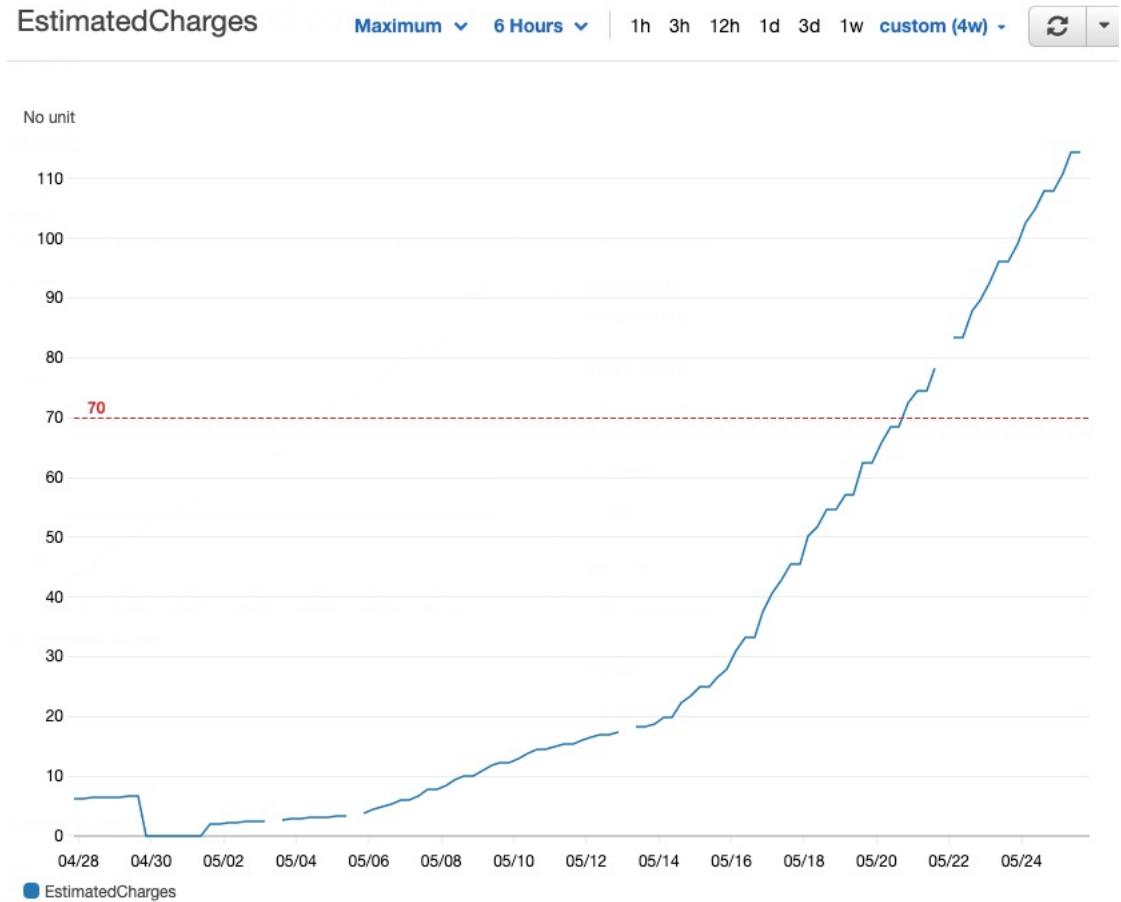
\*Average hourly spend and minimum hourly spend based on your current on-demand spend for the given instance family.

# Cost Explorer – Forecast Usage



# Billing Alarms in CloudWatch

- Billing data metric is stored in CloudWatch us-east-1
- Billing data are for overall worldwide AWS costs
- It's for actual cost, not for projected costs
- Intended a simple alarm (not as powerful as AWS Budgets)



# AWS Budgets



- Create budget and send alarms when costs exceeds the budget
- 3 types of budgets: Usage, Cost, Reservation
- For Reserved Instances (RI)
  - Track utilization
  - Supports EC2, ElastiCache, RDS, Redshift
- Up to 5 SNS notifications per budget
- Can filter by: Service, Linked Account, Tag, Purchase Option, Instance Type, Region, Availability Zone, API Operation, etc...
- Same options as AWS Cost Explorer!
- 2 budgets are free, then \$0.02/day/budget



# Trusted Advisor

- No need to install anything – high level AWS account assessment
- Analyze your AWS accounts and provides recommendation:

**Cost Optimization**



**Performance**



**Security**



**Fault Tolerance**



**Service Limits**



- Core Checks and recommendations – all customers
- Can enable weekly email notification from the console
- Full Trusted Advisor – Available for **Business & Enterprise** support plans
  - Ability to set CloudWatch alarms when reaching limits
  - Programmatic Access using AWS Support API



# Trusted Advisor Checks Examples

- Cost Optimization:
  - low utilization EC2 instances, idle load balancers, under-utilized EBS volumes...
  - Reserved instances & savings plans optimizations,
- Performance:
  - High utilization EC2 instances, CloudFront CDN optimizations
  - EC2 to EBS throughput optimizations, Alias records recommendations
- Security:
  - MFA enabled on Root Account, IAM key rotation, exposed Access Keys
  - S3 Bucket Permissions for public access, security groups with unrestricted ports
- Fault Tolerance:
  - EBS snapshots age, Availability Zone Balance
  - ASG Multi-AZ, RDS Multi-AZ, ELB configuration...
- Service Limits

# AWS Support Plans Pricing



- Basic Support: free

## Developer

Greater of \$29.00

- or -

3% of monthly AWS charges

## Business

Greater of \$100.00

- or -

10% of monthly AWS charges for the first \$0--\$10K

7% of monthly AWS charges from \$10K--\$80K

5% of monthly AWS charges from \$80K--\$250K

3% of monthly AWS charges over \$250K

## Enterprise

Greater of \$15,000.00

- or -

10% of monthly AWS charges for the first \$0--\$150K

7% of monthly AWS charges from \$150K--\$500K

5% of monthly AWS charges from \$500K--\$1M

3% of monthly AWS charges over \$1M

# AWS Basic Support Plan

- **Customer Service & Communities** - 24x7 access to customer service, documentation, whitepapers, and support forums.
- **AWS Trusted Advisor** - Access to the 7 core Trusted Advisor checks and guidance to provision your resources following best practices to increase performance and improve security.
- **AWS Personal Health Dashboard** - A personalized view of the health of AWS services, and alerts when your resources are impacted.

# AWS Developer Support Plan

- All Basic Support Plan +
- Business hours email access to Cloud Support Associates
- Unlimited cases / 1 primary contact
- Case severity / response times:
  - General guidance: < 24 business hours
  - System impaired: < 12 business hours

# AWS Business Support Plan (24/7)

- Intended to be used if you have production workloads
- Trusted Advisor – Full set of checks + API access
- 24x7 phone, email, and chat access to Cloud Support Engineers
- Unlimited cases / unlimited contacts
- Access to Infrastructure Event Management for additional fee.
- Case severity / response times:
  - General guidance: < 24 business hours
  - System impaired: < 12 business hours
  - Production system impaired: < 4 hours
  - Production system down: < 1 hour

# AWS Enterprise Support Plan (24/7)

- Intended to be used if you have mission critical workloads
- All of Business Support Plan +
- Access to a Technical Account Manager (TAM)
- Concierge Support Team (for billing and account best practices)
- Infrastructure Event Management, Well-Architected & Operations Reviews
- Case severity / response times:
  - ...
  - Production system impaired: < 4 hours
  - Production system down: < 1 hour
  - Business-critical system down: < 15 minutes

# Account Best Practices – Summary

- Operate multiple accounts using **Organizations**
- Use **SCP** (service control policies) to restrict account power
- Easily setup multiple accounts with best-practices with **AWS Control Tower**
- **Use Tags & Cost Allocation Tags** for easy management & billing
- **IAM guidelines:** MFA, least-privilege, password policy, password rotation
- **Config** to record all resources configurations & compliance over time
- **CloudFormation** to deploy stacks across accounts and regions
- **Trusted Advisor** to get insights, Support Plan adapted to your needs
- Send Service Logs and Access Logs to **S3** or **CloudWatch Logs**
- **CloudTrail** to record API calls made within your account
- **If your Account is compromised:** change the root password, delete and rotate all passwords / keys, contact the AWS support

# Billing and Costing Tools – Summary

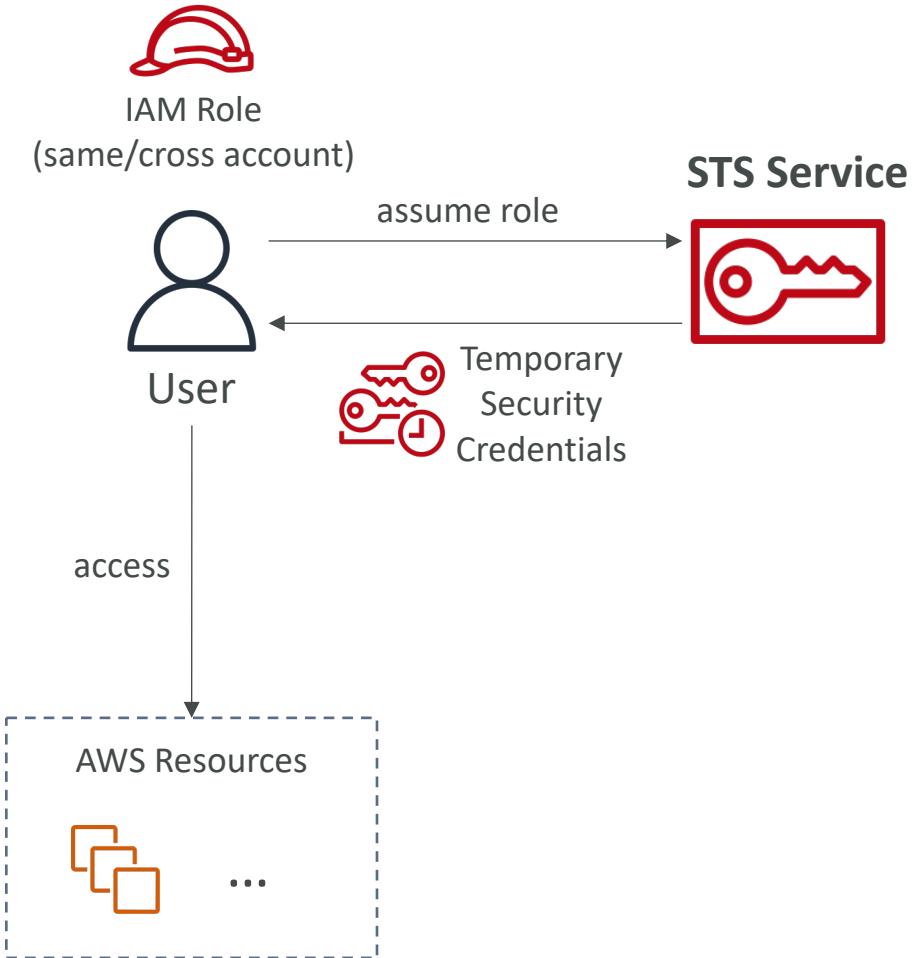


- **Compute Optimizer:** recommends resources' configurations to reduce cost
- **TCO Calculator:** from on-premises to AWS
- **Simple Monthly Calculator / Pricing Calculator:** cost of services on AWS
- **Billing Dashboard:** high level overview + free tier dashboard
- **Cost Allocation Tags:** tag resources to create detailed reports
- **Cost and Usage Reports:** most comprehensive billing dataset
- **Cost Explorer:** View current usage (detailed) and forecast usage
- **Billing Alarms:** in us-east-1 – track overall and per-service billing
- **Budgets:** more advanced – track usage, costs, RI, and get alerts
- **Savings Plans:** easy way to save based on long-term usage of AWS

# Advanced Identity Section

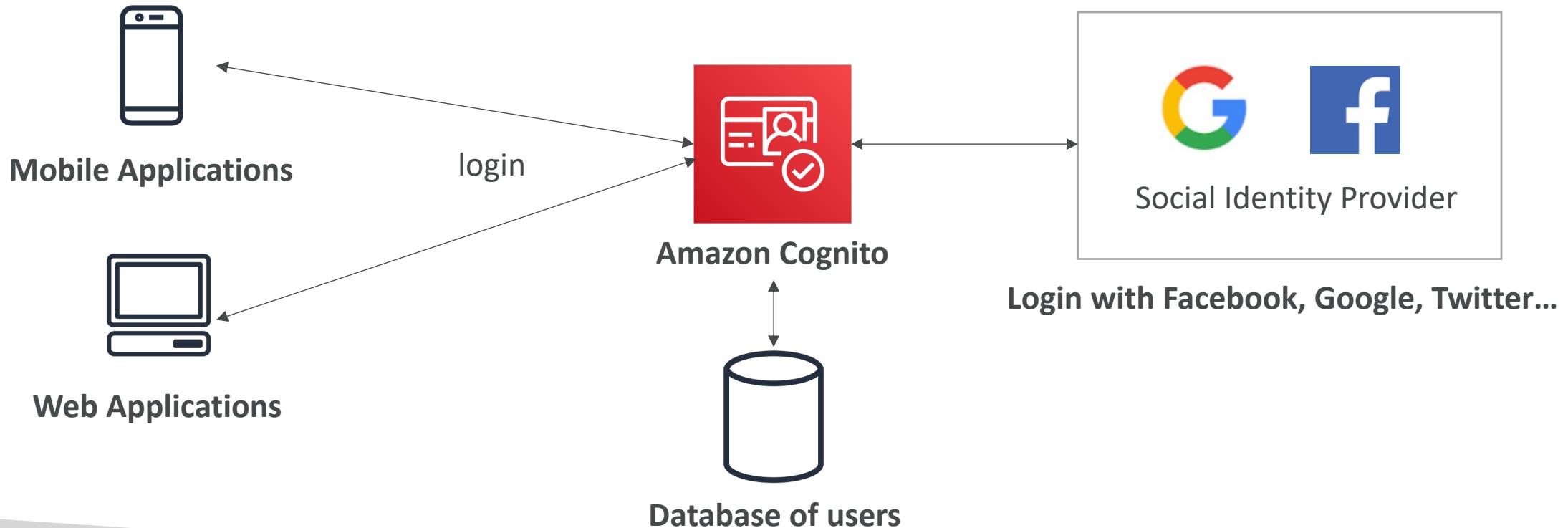
# AWS STS (Security Token Service)

- Enables you to create **temporary, limited-privileges credentials** to access your AWS resources
- Short-term credentials: you configure expiration period
- Use cases
  - **Identity federation:** manage user identities in external systems, and provide them with STS tokens to access AWS resources
  - **IAM Roles for cross/same account access**
  - **IAM Roles for Amazon EC2:** provide temporary credentials for EC2 instances to access AWS resources



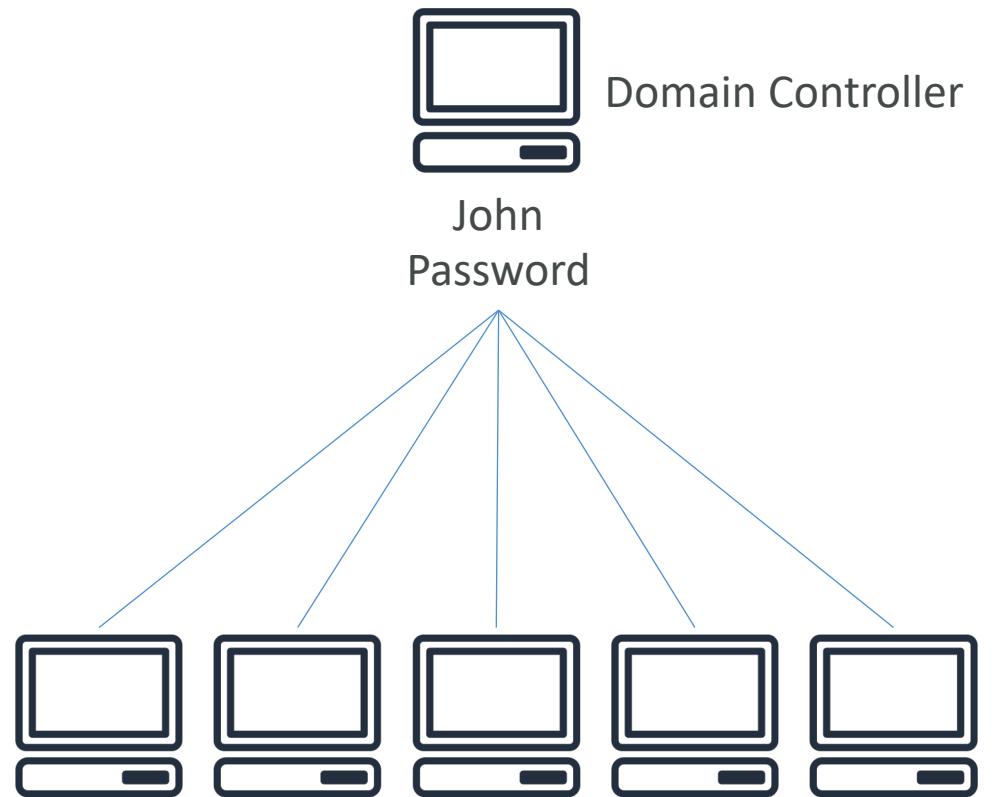
# Amazon Cognito (simplified)

- Identity for your Web and Mobile applications users (potentially millions)
- Instead of creating them an IAM user; you create a user in Cognito



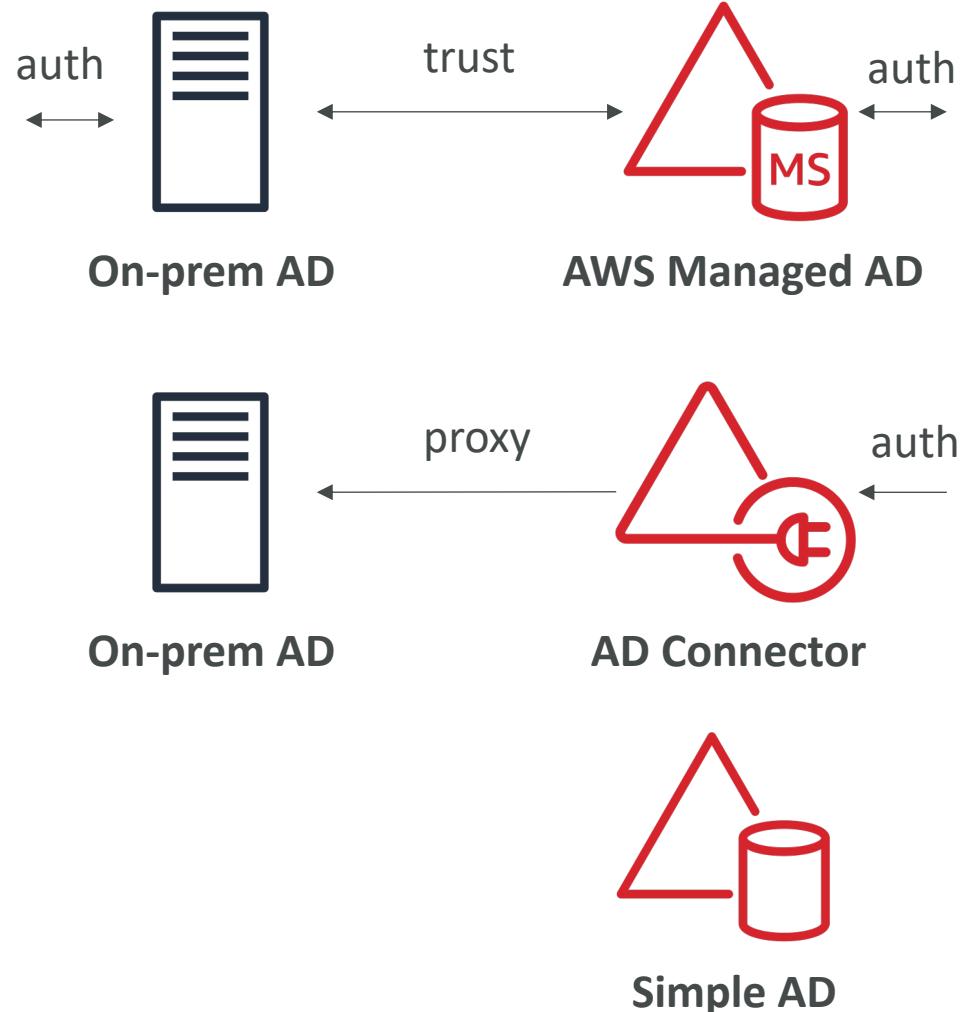
# What is Microsoft Active Directory (AD)?

- Found on any Windows Server with AD Domain Services
- Database of **objects**: User Accounts, Computers, Printers, File Shares, Security Groups
- Centralized security management, create account, assign permissions



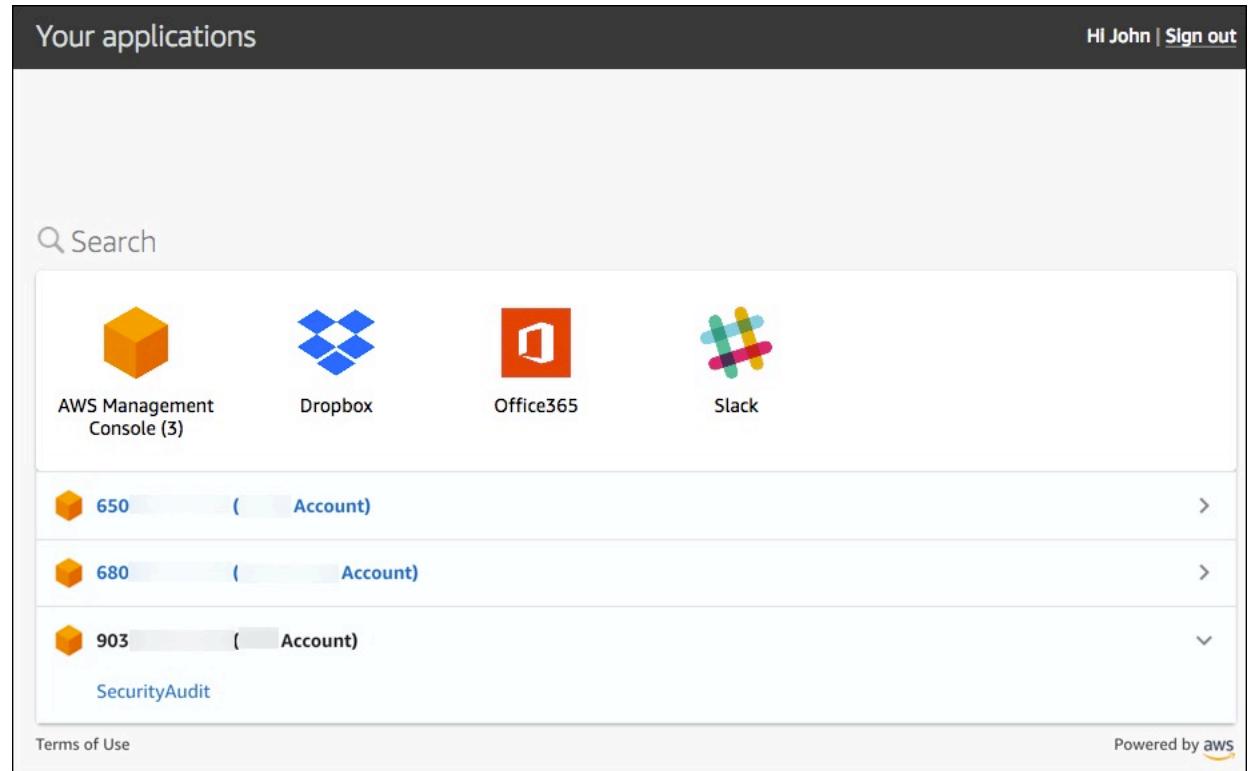
# AWS Directory Services

- AWS Managed Microsoft AD
  - Create your own AD in AWS, manage users locally, supports MFA
  - Establish “trust” connections with your on-premise AD
- AD Connector
  - Directory Gateway (proxy) to redirect to on-premise AD
  - Users are managed on the on-premise AD
- Simple AD
  - AD-compatible managed directory on AWS
  - Cannot be joined with on-premise AD



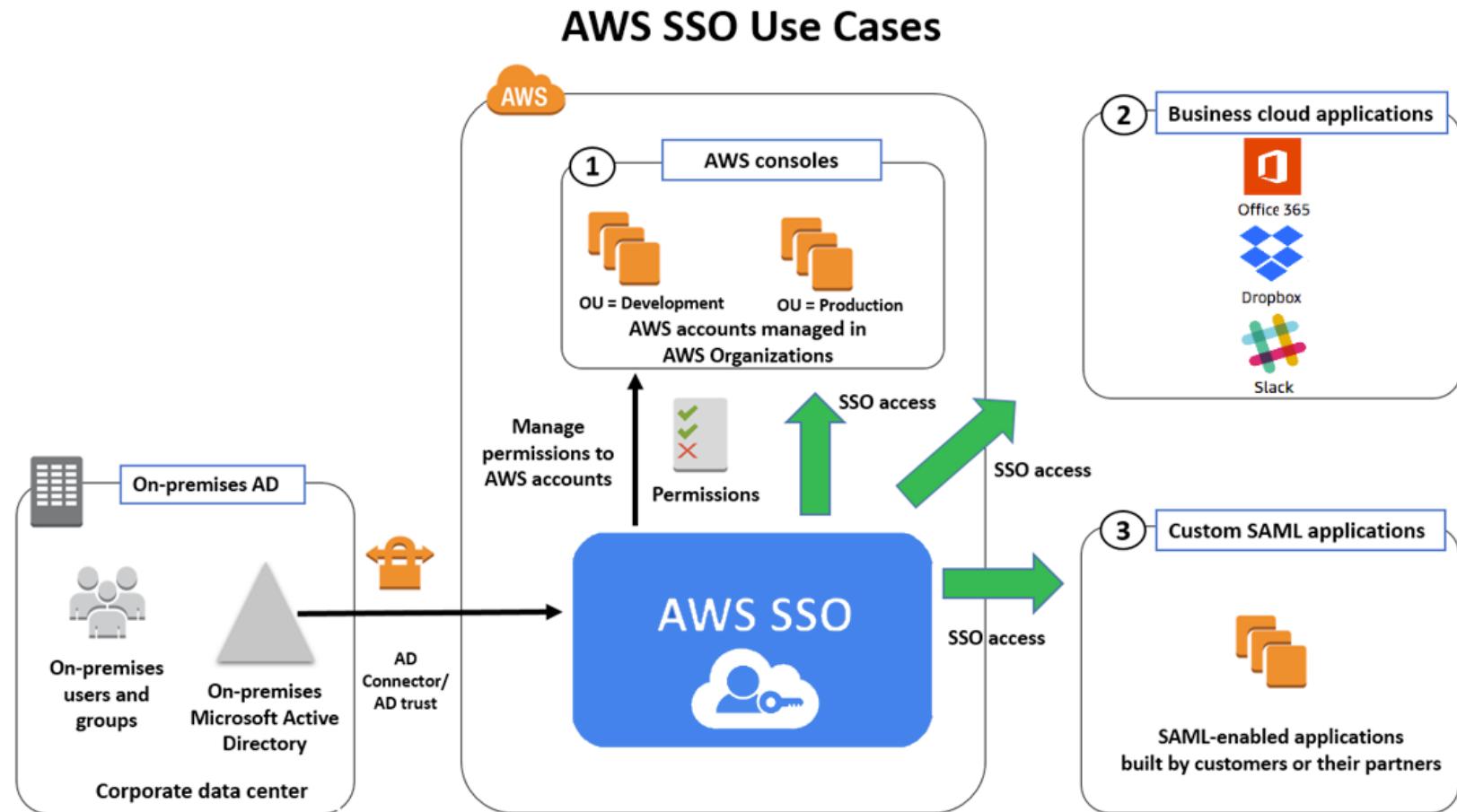
# AWS Single Sign-On (SSO)

- Centrally manage Single Sign-On to access multiple accounts and 3<sup>rd</sup>-party business applications.
- Integrated with AWS Organizations
- Supports SAML 2.0 markup
- Integration with on-premise Active Directory



<https://aws.amazon.com/blogs/security/introducing-aws-single-sign-on/>

# AWS Single Sign-On (SSO) – Setup with AD



# Advanced Identity - Summary

- IAM
  - Identity and Access Management inside your AWS account
  - For users that you trust and belong to your company
- Organizations: manage multiple AWS accounts
- Security Token Service (STS): temporary, limited-privileges credentials to access AWS resources
- Cognito: create a database of users for your mobile & web applications
- Directory Services: integrate Microsoft Active Directory in AWS
- Single Sign-On (SSO): one login for multiple AWS accounts & applications

# Other AWS Services

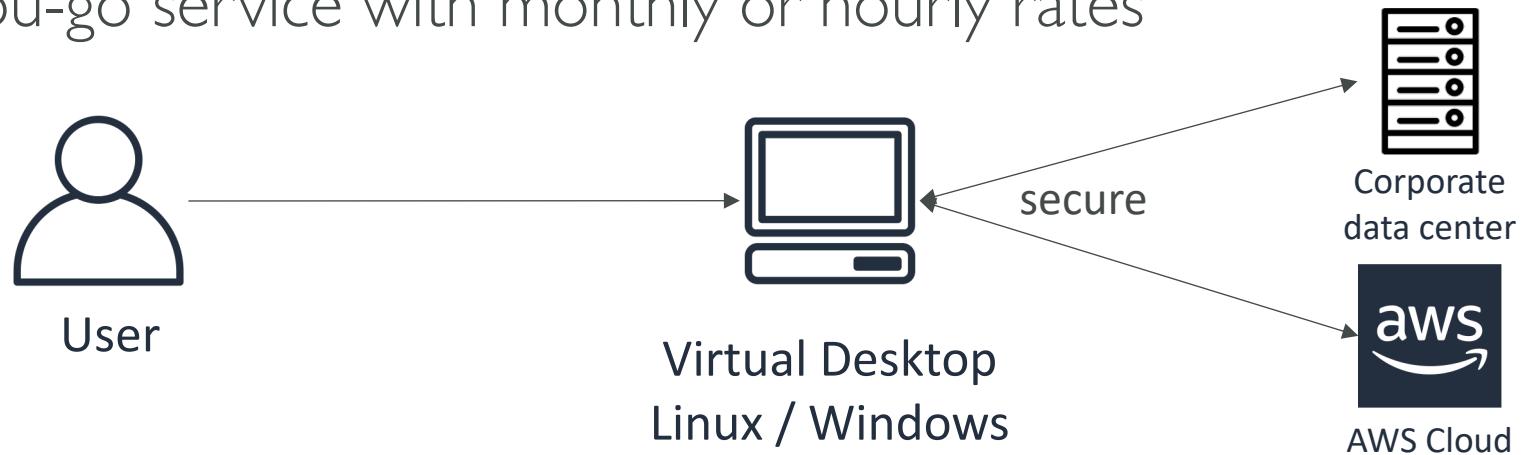
# Other AWS services section

- Other services represent services I couldn't group with the other ones
- They are services reported by students as **sometimes, but rarely**, appearing on the AWS exam
- The lectures are short and brief and most likely without hands-on
- No summary lecture at the end of the section to keep things flexible!

# Amazon WorkSpaces



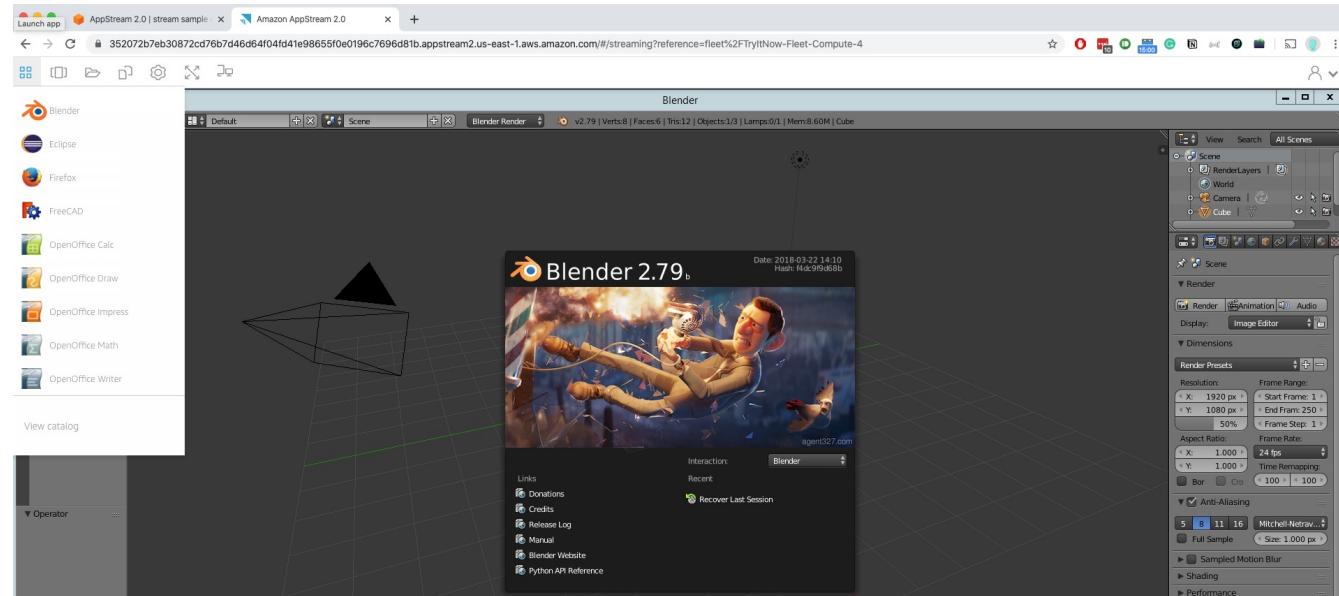
- Managed Desktop as a Service (DaaS) solution to easily provision Windows or Linux desktops
- Great to eliminate management of on-premise VDI (Virtual Desktop Infrastructure)
- Fast and quickly scalable to thousands of users
- Secured data – integrates with KMS
- Pay-as-you-go service with monthly or hourly rates





# Amazon AppStream 2.0

- Desktop Application Streaming Service
- Deliver to any computer; without acquiring, provisioning infrastructure
- The application is delivered from within a web browser



# Amazon AppStream 2.0 vs WorkSpaces

- **Workspaces**

- Fully managed VDI and desktop available
- The users connect to the VDI and open native or WAM applications
- Workspaces are on-demand or always on

- **AppStream 2.0**

- Stream a desktop application to web browsers (no need to connect to a VDI)
- Works with any device (that has a web browser)
- Allow to configure an instance type per application type (CPU, RAM, GPU)

# Amazon Sumerian



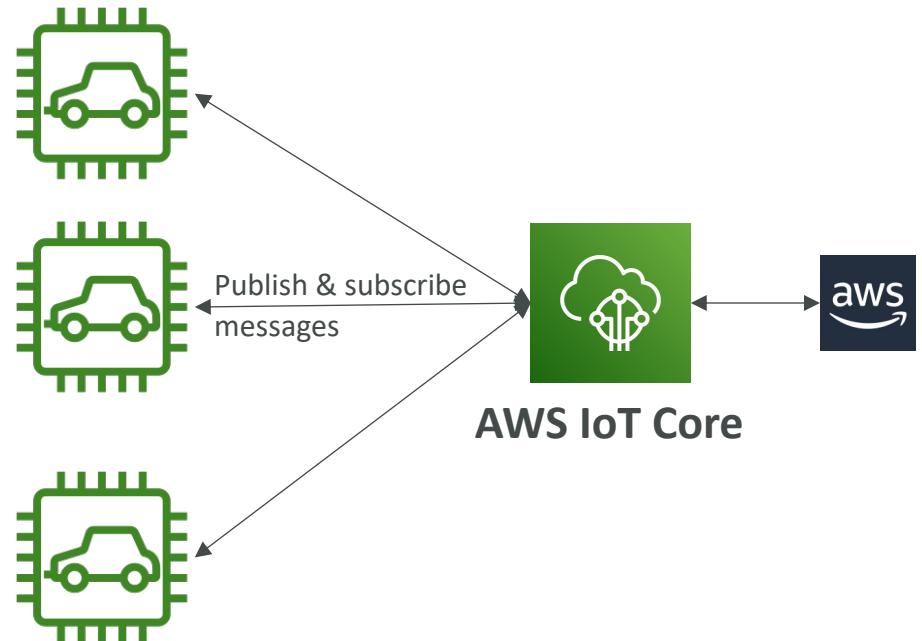
- Create and run virtual reality (VR), augmented reality (AR), and 3D applications
- Can be used to quickly **create** 3D models with animations
- Ready-to-use templates and assets - no programming or 3D expertise required
- Accessible via a web-browser URLs or on popular hardware for AR/VR
- Example: <https://docs.aws.amazon.com/sumerian/latest/userguide/gettingstarted-showcase.html>



# AWS IoT Core



- IoT stands for “Internet of Things” – the network of internet-connected devices that are able to collect and transfer data
- AWS IoT Core allows you to **easily connect IoT devices to the AWS Cloud**
- **Serverless, secure & scalable** to billions of devices and trillions of messages
- Your applications can communicate with your devices even when they aren't connected
- Integrates with a lot of AWS services (Lambda, S3, SageMaker, etc.)
- Build IoT applications that gather, process, analyze, and act on data



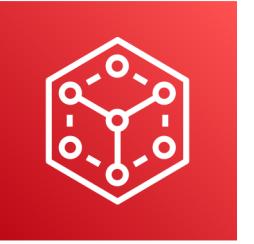
# Amazon Elastic Transcoder



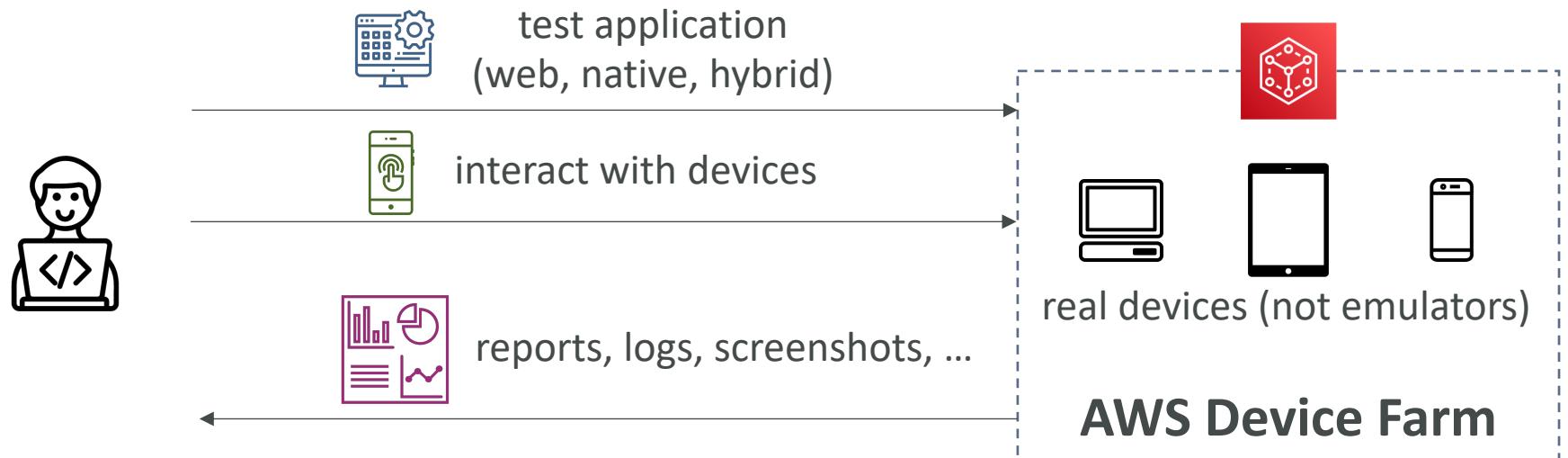
- Elastic Transcoder is used to convert media files stored in S3 into media files in the formats required by consumer playback devices (phones etc..)
- Benefits:
  - Easy to use
  - Highly scalable – can handle large volumes of media files and large file sizes
  - Cost effective – duration-based pricing model
  - Fully managed & secure, pay for what you use



# AWS Device Farm



- Fully-managed service that tests your web and mobile apps against desktop browsers, real mobile devices, and tablets
- Run tests concurrently on multiple devices (speed up execution)
- Ability to configure device settings (GPS, language, Wi-Fi, Bluetooth, ...)

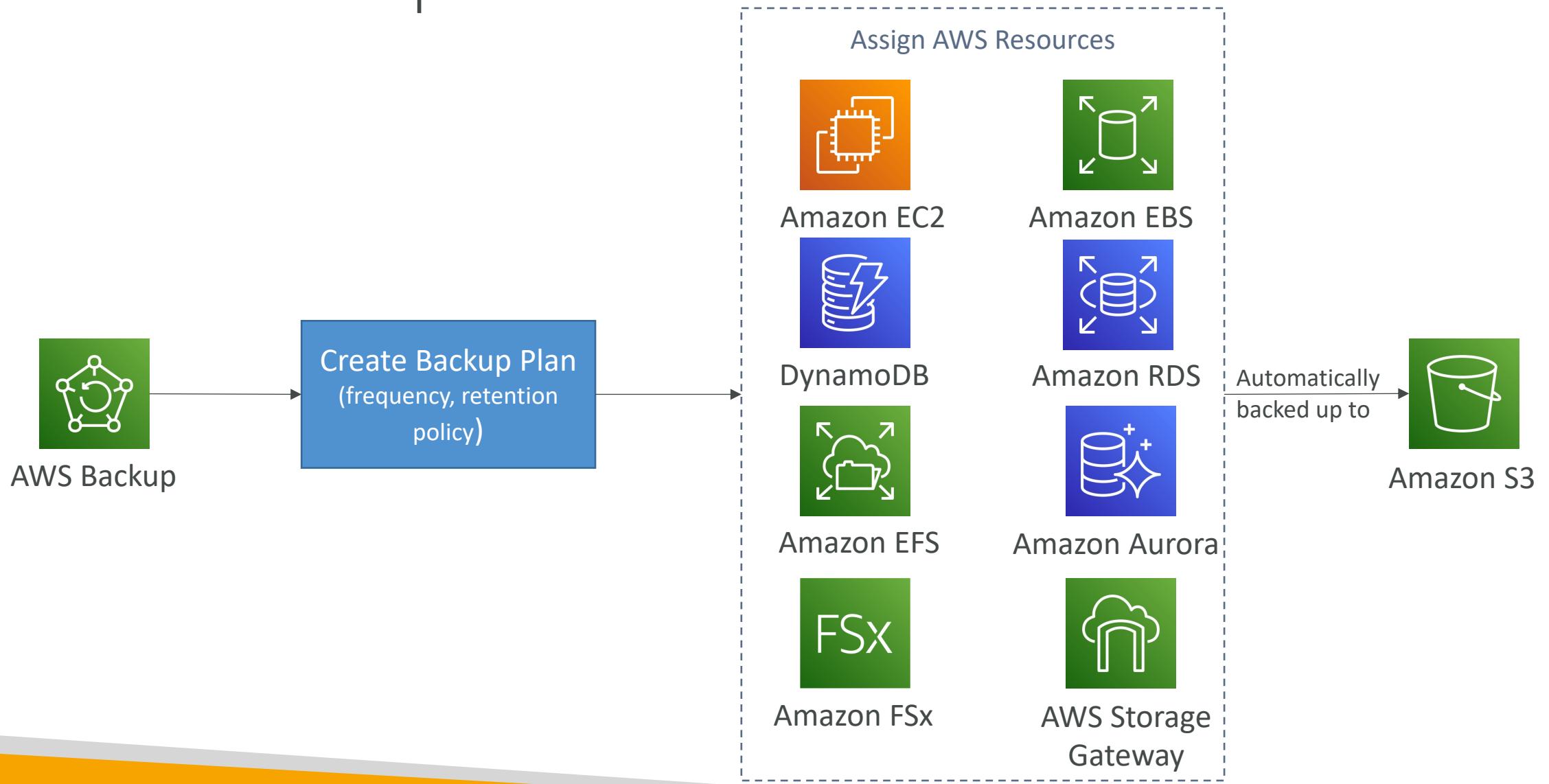


# AWS Backup



- Fully-managed service to centrally manage and automate backups across AWS services
- On-demand and scheduled backups
- Supports PITR (Point-in-time Recovery)
- Retention Periods, Lifecycle Management, Backup Policies, ...
- Cross-Region Backup
- Cross-Account Backup (using AWS Organizations)

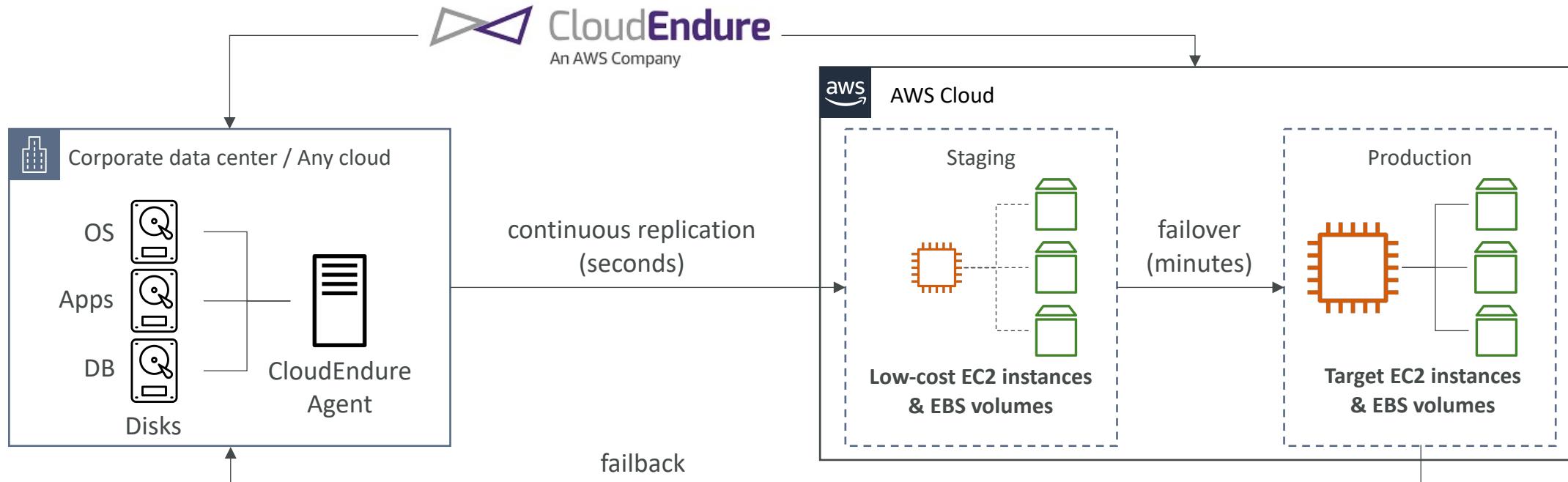
# AWS Backup



# CloudEndure Disaster Recovery



- Quickly and easily recover your physical, virtual, and cloud-based servers into AWS
- Example: protect your most critical databases (including Oracle, MySQL, and SQL Server), enterprise apps (SAP), protect your data from ransomware attacks, ...
- Continuous block-level replication for your servers



# AWS Architecting & Ecosystem Section

# Well Architected Framework

## General Guiding Principles

- Stop guessing your capacity needs
- Test systems at production scale
- Automate to make architectural experimentation easier
- Allow for evolutionary architectures
  - Design based on changing requirements
- Drive architectures using data
- Improve through game days
  - Simulate applications for flash sale days

# AWS Cloud Best Practices – Design Principles

- **Scalability:** vertical & horizontal
- **Disposable Resources:** servers should be disposable & easily configured
- **Automation:** Serverless, Infrastructure as a Service, Auto Scaling...
- **Loose Coupling:**
  - Monolith are applications that do more and more over time, become bigger
  - Break it down into smaller, loosely coupled components
  - A change or a failure in one component should not cascade to other components
- **Services, not Servers:**
  - Don't use just EC2
  - Use managed services, databases, serverless, etc !

# Well Architected Framework

## 5 Pillars

- 1) Operational Excellence
  - 2) Security
  - 3) Reliability
  - 4) Performance Efficiency
  - 5) Cost Optimization
- 
- They are not something to balance, or trade-offs, they're a synergy

# I) Operational Excellence

- Includes the ability to run and monitor systems to deliver business value and to continually improve supporting processes and procedures
- Design Principles
  - Perform operations as code - Infrastructure as code
  - Annotate documentation - Automate the creation of annotated documentation after every build
  - Make frequent, small, reversible changes - So that in case of any failure, you can reverse it
  - Refine operations procedures frequently - And ensure that team members are familiar with it
  - Anticipate failure
  - Learn from all operational failures

# Operational Excellence AWS Services

- Prepare



AWS CloudFormation



AWS Config

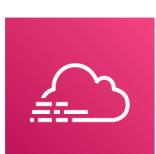
- Operate



AWS CloudFormation



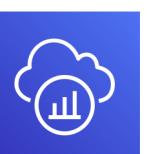
AWS Config



AWS CloudTrail



Amazon CloudWatch



AWS X-Ray

- Evolve



AWS CloudFormation



AWS CodeBuild



AWS CodeCommit



AWS CodeDeploy



AWS CodePipeline

## 2) Security

- Includes the ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies
- Design Principles
  - **Implement a strong identity foundation** - Centralize privilege management and reduce (or even eliminate) reliance on long-term credentials - Principle of least privilege - IAM
  - **Enable traceability** - Integrate logs and metrics with systems to automatically respond and take action
  - **Apply security at all layers** - Like edge network, VPC, subnet, load balancer, every instance, operating system, and application
  - **Automate security best practices**
  - **Protect data in transit and at rest** - Encryption, tokenization, and access control
  - **Keep people away from data** - Reduce or eliminate the need for direct access or manual processing of data
  - **Prepare for security events** - Run incident response simulations and use tools with automation to increase your speed for detection, investigation, and recovery

# Security AWS Services

- Identity and Access Management



IAM



AWS STS



MFA token



AWS Organizations

- Detective Controls



AWS Config



AWS CloudTrail



Amazon CloudWatch

- Infrastructure Protection



Amazon CloudFront



Amazon VPC



AWS Shield



AWS WAF



Amazon Inspector

- Data Protection:



KMS



S3



Elastic Load Balancing (ELB)



Amazon EBS



Amazon RDS

- Incident Response



IAM



AWS CloudFormation



Amazon CloudWatch Events

# 3) Reliability

- Ability of a system to recover from infrastructure or service disruptions, dynamically acquire computing resources to meet demand, and mitigate disruptions such as misconfigurations or transient network issues
- Design Principles
  - **Test recovery procedures** - Use automation to simulate different failures or to recreate scenarios that led to failures before
  - **Automatically recover from failure** - Anticipate and remediate failures before they occur
  - **Scale horizontally to increase aggregate system availability** - Distribute requests across multiple, smaller resources to ensure that they don't share a common point of failure
  - **Stop guessing capacity** - Maintain the optimal level to satisfy demand without over or under provisioning - Use Auto Scaling
  - **Manage change in automation** - Use automation to make changes to infrastructure

# Reliability AWS Services

- Foundations



IAM



Amazon VPC



Service Quotas



AWS Trusted Advisor

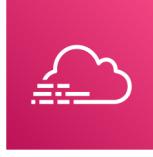
- Change Management



AWS Auto Scaling



Amazon CloudWatch



AWS CloudTrail



AWS Config

- Failure Management



Backups



AWS CloudFormation



Amazon S3



Amazon S3 Glacier



Amazon Route 53

# 4) Performance Efficiency

- Includes the ability to use computing resources efficiently to meet system requirements, and to maintain that efficiency as demand changes and technologies evolve
- Design Principles
  - **Democratize advanced technologies** - Advance technologies become services and hence you can focus more on product development
  - **Go global in minutes** - Easy deployment in multiple regions
  - **Use serverless architectures** - Avoid burden of managing servers
  - **Experiment more often** - Easy to carry out comparative testing
  - **Mechanical sympathy** - Be aware of all AWS services

# Performance Efficiency AWS Services

- Selection



AWS Auto Scaling



AWS Lambda



Amazon Elastic Block Store  
(EBS)



Amazon Simple Storage  
Service (S3)



Amazon RDS

- Review



AWS CloudFormation



AWS Lambda

- Monitoring



Amazon CloudWatch



AWS Lambda

- Tradeoffs



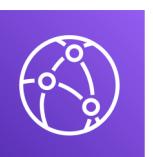
Amazon RDS



Amazon ElastiCache



AWS Snowball



Amazon CloudFront

## AWS News Blog

# 5) Cost Optimization

- Includes the ability to run systems to deliver business value at the lowest price point
- Design Principles
  - Adopt a consumption mode - Pay only for what you use
  - Measure overall efficiency - Use CloudWatch
  - Stop spending money on data center operations - AWS does the infrastructure part and enables customer to focus on organization projects
  - Analyze and attribute expenditure - Accurate identification of system usage and costs, helps measure return on investment (ROI) - Make sure to use tags
  - Use managed and application level services to reduce cost of ownership - As managed services operate at cloud scale, they can offer a lower cost per transaction or service

# Cost Optimization AWS Services

- Expenditure Awareness



AWS Budgets



AWS Cost and Usage Report



AWS Cost Explorer



Reserved Instance Reporting

- Cost-Effective Resources



Spot instance



Reserved instance



Amazon S3 Glacier

- Matching supply and demand



AWS Auto Scaling



AWS Lambda

- Optimizing Over Time



AWS Trusted Advisor



AWS Cost and Usage Report

**AWS News Blog**

# AWS Well-Architected Tool



- Free tool to **review your architectures** against the 5 pillars Well-Architected Framework and **adopt architectural best practices**
- How does it work?
  - Select your workload and answer questions
  - Review your answers against the 5 pillars
  - Obtain advice: get videos and documentations, generate a report, see the results in a dashboard
- Let's have a look: <https://console.aws.amazon.com/wellarchitected>

Name	Overall status	High risks	Medium risks	Improvement status	Last updated
Internal Employee Portal	Answered	13	2	None	Nov 24, 2018 3:40 PM UTC-8
Mobile app - Android	Answered	9	1	None	Nov 24, 2018 3:43 PM UTC-8
Mobile app - iOS	Answered	0	1	None	Nov 24, 2018 3:49 PM UTC-8
Retail Website- EU	Unanswered	0	0	None	Nov 24, 2018 3:52 PM UTC-8
Retail Website- North America	Unanswered	0	0	None	Nov 24, 2018 3:19 PM UTC-8

<https://aws.amazon.com/blogs/aws/new-aws-well-architected-tool-review-workloads-against-best-practices/>

# AWS Ecosystem – Free resources

- AWS Blogs: <https://aws.amazon.com/blogs/aws/>
- AWS Forums (community): <https://forums.aws.amazon.com/index.jspa>
- AWS Whitepapers & Guides: <https://aws.amazon.com/whitepapers>
- AWS Quick Starts: <https://aws.amazon.com/quickstart/>
  - Automated, gold-standard deployments in the AWS Cloud
  - Build your production environment quickly with templates
  - Example: WordPress on AWS [https://fwd.aws/P3yyv?did=qs\\_card&trk=qs\\_card](https://fwd.aws/P3yyv?did=qs_card&trk=qs_card)
  - Leverages CloudFormation
- AWS Solutions: <https://aws.amazon.com/solutions/>
  - Vetted Technology Solutions for the AWS Cloud
  - Example - AWS Landing Zone: secure, multi-account AWS environment
    - <https://aws.amazon.com/solutions/implementations/aws-landing-zone/>
    - “Replaced” by AWS Control Tower

# AWS Ecosystem - AWS Support

<b>DEVELOPER</b>	<ul style="list-style-type: none"><li>• Business hours email access to Cloud Support Associates</li><li>• General guidance: &lt; 24 business hours</li><li>• System impaired: &lt; 12 business hours</li></ul>
<b>BUSINESS</b>	<ul style="list-style-type: none"><li>• 24x7 phone, email, and chat access to Cloud Support Engineers</li><li>• Production system impaired: &lt; 4 hours</li><li>• Production system down: &lt; 1 hour</li></ul>
<b>ENTERPRISE</b>	<ul style="list-style-type: none"><li>• Access to a Technical Account Manager (TAM)</li><li>• Concierge Support Team (for billing and account best practices)</li><li>• Business-critical system down: &lt; 15 minutes</li></ul>

# AWS Marketplace



- Digital catalog with thousands of software listings from **independent software vendors** (3<sup>rd</sup> party)
- Example:
  - Custom AMI (custom OS, firewalls, technical solutions...)
  - CloudFormation templates
  - Software as a Service
  - Containers
- If you buy through the AWS Marketplace, it goes into your AWS bill
- You can **sell your own solutions** on the AWS Marketplace

# AWS Training

- AWS Digital (online) and Classroom Training (in-person or virtual)
- AWS Private Training (for your organization)
- Training and Certification for the U.S Government
- Training and Certification for the Enterprise
- AWS Academy: helps universities teach AWS
- And your favorite online teacher...  
teaching you all about AWS Certifications and more!

# AWS Professional Services & Partner Network

- The AWS Professional Services organization is a global team of experts
- They work alongside your team and a chosen member of the APN
- APN = AWS Partner Network
- APN Technology Partners: providing hardware, connectivity, and software
- APN Consulting Partners: professional services firm to help build on AWS
- APN Training Partners: find who can help you learn AWS
- AWS Competency Program: AWS Competencies are granted to APN Partners who have demonstrated technical proficiency and proven customer success in specialized solution areas.
- AWS Navigate Program: help Partners become better Partners

# Exam Preparation Section

# Quick note on Distractors

- There are many services you will find in questions that are **distractors**
- There are **over 200 AWS services**, and we can't cover them all
  - Quicksight, Cognito, AppStreams, Server Migration Service, etc...
- I have covered all services that from my research and experience, people get questions for at the exam.
- If you see a service not covered by my course, but in someone else's practice exam, don't panic, I must have intentionally left it out
- If you see a service that is an answer at the exam but not covered in my course, please let me know!

# State of learning checkpoint

- Let's look how far we've gone on our learning journey
- <https://aws.amazon.com/certification/certified-cloud-practitioner/>

# Practice makes perfect

- If you're new to AWS, take a bit of AWS practice thanks to this course before rushing to the exam
  - The exam recommends you have 6 months or more of hands-on experience on AWS
  - Practice makes perfect!
- 
- If you feel overwhelmed by the amount of knowledge you just learned, just go through it one more time

# Exam content

- Two types of questions:
  - **Multiple choice:** has one correct answer and three incorrect responses
  - **Multiple response:** has two or more correct responses out of five or more options – CAREFUL: the exam software does not tell you if you selected the right number of answers (but the required number is mentioned)
- **Always try to answer the question**
  - Unanswered questions are considered as incorrect
  - No penalty for a wrong answer → guess!
- If you need to review a question for later (when you're done answering all questions), you can **flag it**

## MULTIPLE CHOICE QUESTION

Which AWS service does rock...?

- Option 1
- Option 2
- Option 3
- Option 4

## MULTI REONSE QUESTION

Blabla question...? (SELECT TWO)

- Option 1
- Option 2
- Option 3
- Option 4
- Option 5

# Proceed by elimination

- Most questions are going to be high-level “pick-a-service” questions
  - For all the questions, rule out answers that you know for sure are wrong
  - For the remaining answers, understand which one makes the most sense
- 
- There are very few trick questions
  - Don’t over-think it
  - If a solution seems feasible but highly complicated, it’s probably wrong

# Read each service's Overview

- Example: <https://aws.amazon.com/s3/>
- Overviews cover a lot of the questions asked at the exam
- They help confirm your understanding of a service

# Get into the AWS Community

- Help and discuss with other people in the course Q&A
- Review questions asked by other people in the Q&A
- Do the practice test in this section
- Do extra practice exams (for example from my practice exam course)
  
- Read forums online
- Read online blogs
- Attend local meetups and discuss with other AWS engineers
- Watch re-invent videos on YouTube (AWS Conference)

# How will the exam work?

- You'll have to register online at <https://www.aws.training/>
- Fee for the exam is 100 USD
- Provide two identity documents (ID, Credit Card, details are in emails sent to you)
- No notes are allowed, no pen is allowed, no speaking
- 65 questions will be asked in 90 minutes
- At the end you can optionally review all the questions / answers
  
- You would know right away if you passed / failed the exams
- You will not know which answers were right / wrong
- You will know the overall score a few days later (email notification)
- To pass you need a score of **at least 700 out of 1000**
- If you fail, you can retake the exam again 14 days later

# Congratulations!

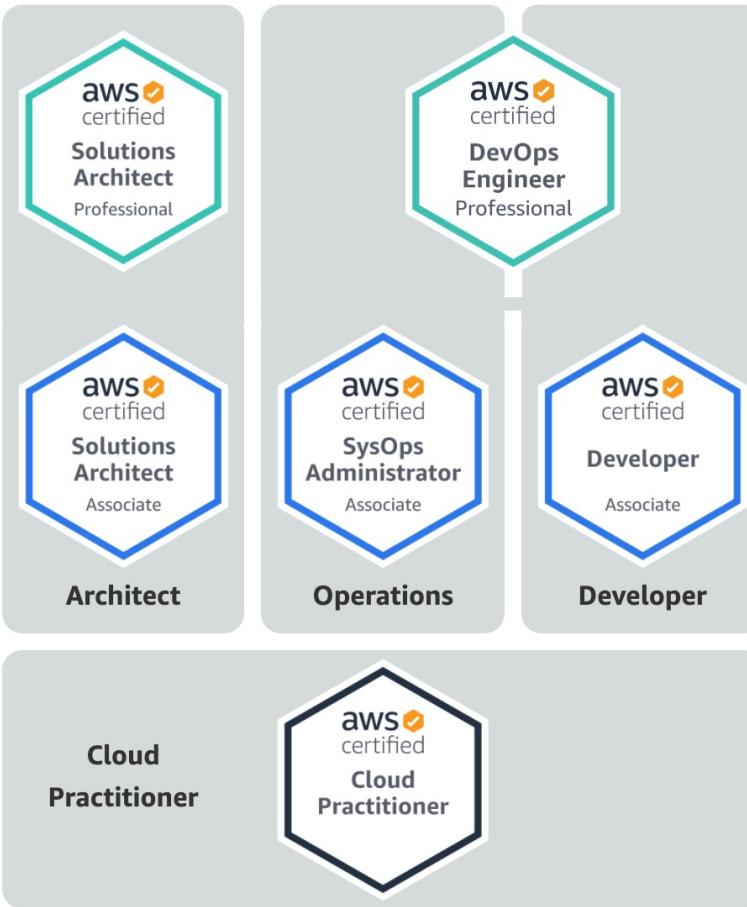
# Congratulations!

- Congrats on finishing the course!
- I hope you will pass the exam without a hitch ☺
- If you haven't done so yet, I'd love a review from you!
- If you passed, I'll be more than happy to know I've helped
  - Post it in the Q&A to help & motivate other students. Share your tips!
  - Post it on LinkedIn and tag me!
- Overall, I hope you learned how to use AWS and that you will be a tremendously good AWS Cloud Practitioner

# Your AWS Certification journey

## Professional

**Two years** of comprehensive experience designing, operating, and troubleshooting solutions using the AWS Cloud



## Associate

**One year** of experience solving problems and implementing solutions using the AWS Cloud

## Foundational

**Six months** of fundamental AWS Cloud and industry knowledge

## Specialty

Technical AWS Cloud experience in the Specialty domain as specified in the **exam guide**

