

# AWS Certified CloudOps Engineer Associate

By Stéphane Maarek



COURSE →



EXTRA PRACTICE EXAMS →

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

- This document is reserved for people enrolled into the [AWS Certified CloudOps Engineer Associate course by Stephane Maarek](#)
- 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!

# Table of Contents

- [Amazon EC2 for SysOps](#)
- [Amazon Machine Image \(AMI\)](#)
- [Management of EC2 at Scale](#)
- [High Availability & Scalability](#)
- [AWS CloudFormation](#)
- [AWS Lambda](#)
- [Amazon EC2 Storage & Data Management](#)
- [Amazon S3](#)
- [Amazon S3 – Advanced](#)

# Table of Contents

- [Amazon S3 – Security](#)
- [Advanced Storage Solutions](#)
- [Amazon CloudFront](#)
- [Databases in AWS](#)
- [AWS Monitoring, Audit & Performance](#)
- [AWS Account Management](#)
- [Disaster Recovery](#)
- [Security & Compliance](#)
- [Identity](#)
- [Amazon Route 53](#)

# Table of Contents

- [Amazon VPC](#)
- [Other Services](#)
- [Exam Preparation](#)
- [Congratulations](#)

# AWS Certified CloudOps Engineer – Associate Course

## SOA-C03

# Welcome!

- We're going to prepare for the CloudOps exam **SOA-C03**
- It's a challenging certification, this course will be long and interesting 😊
- **How the course is structured:**
  - If some base knowledge is needed: videos are imported from
    - Certified Cloud Practitioner, ex: [CCP] EC2 Instances Launch Types
    - Certified Solutions Architect Associate, ex: [SAA] EC2 Hibernate Hands On
    - Certified Developer Associate, ex: [DVA] CloudFormation Drift
  - CloudOps-specific videos (not marked with [...])

# About me

- I'm Stephane!
- I'm AWS Certified
- 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>
  - Instagram: <https://Instagram.com/stephanemaarek>
  - Medium: <https://medium.com/@stephane.maarek>
  - Twitter: <https://twitter.com/stephanemaarek>
  - GitHub: <https://github.com/simplesteph>

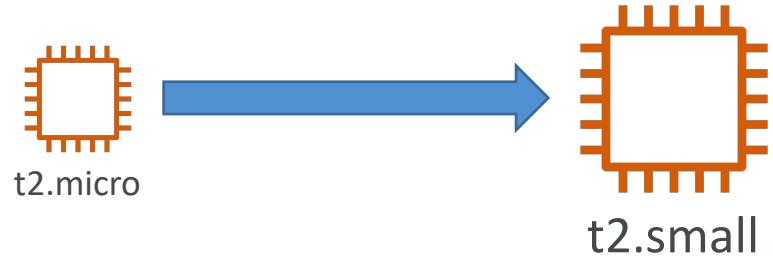


★ 4.7 Instructor Rating  
◐ 673,815 Reviews  
■ 2,173,080 Students  
▶ 56 Courses

# Amazon EC2 for CloudOps

Rocking EC2 from a CloudOps perspective

# EC2 Changing Instance Type

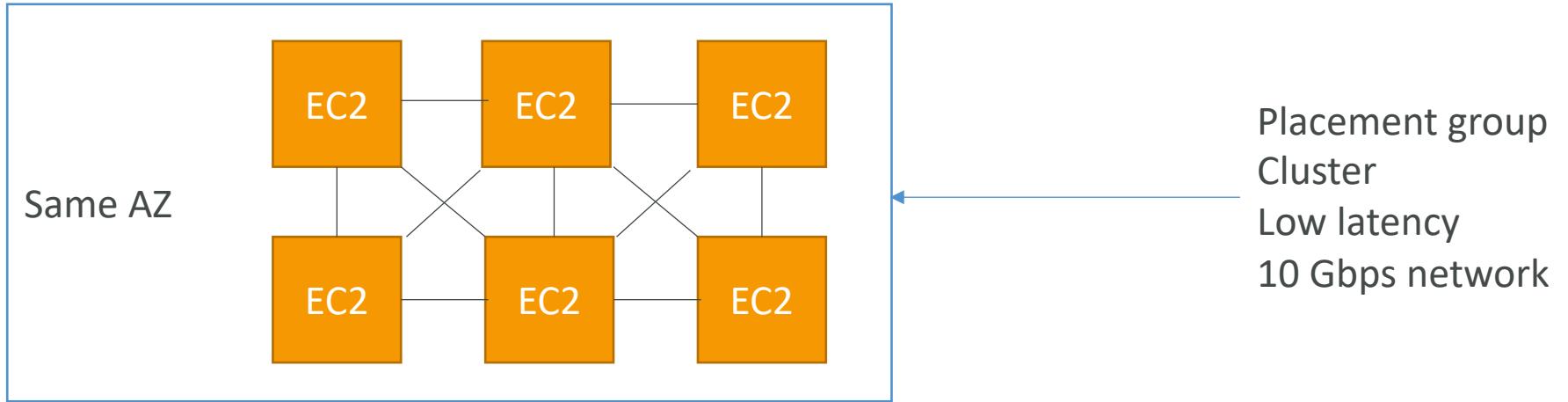


- This only works for EBS backed instances
- Stop the instance
- Instance Settings => Change Instance Type
- Start Instance

# Placement Groups

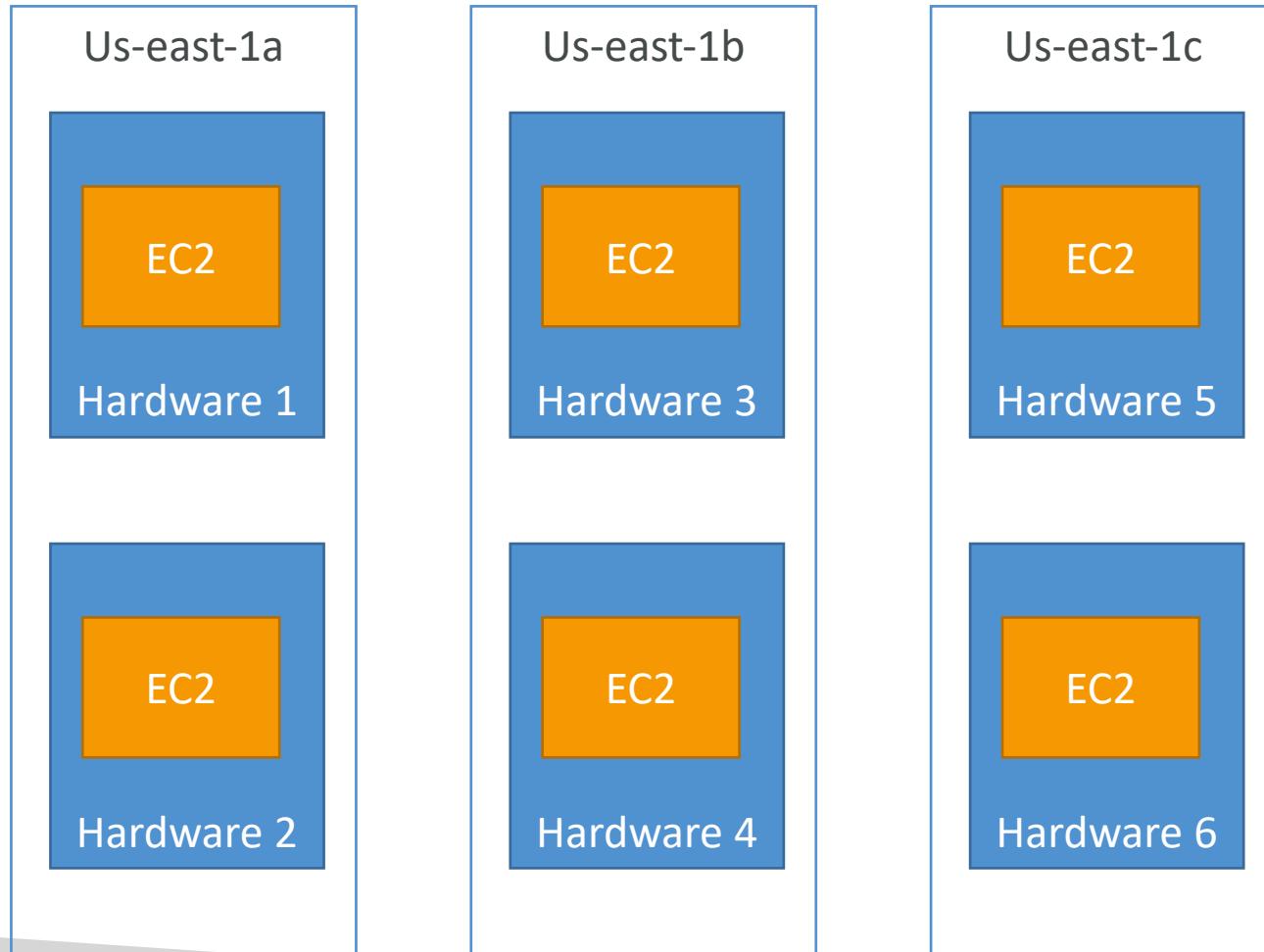
- Sometimes you want control over the EC2 Instance placement strategy
- That strategy can be defined using placement groups
- When you create a placement group, you specify one of the following strategies for the group:
  - *Cluster*—clusters instances into a low-latency group in a single Availability Zone
  - *Spread*—spreads instances across underlying hardware (max 7 instances per group per AZ) – critical applications
  - *Partition*—spreads instances across many different partitions (which rely on different sets of racks) within an AZ. Scales to 100s of EC2 instances per group (Hadoop, Cassandra, Kafka)

# Placement Groups Cluster



- Pros: Great network (10 Gbps bandwidth between instances with Enhanced Networking enabled - recommended)
- Cons: If the AZ fails, all instances fail at the same time
- Use case:
  - Big Data job that needs to complete fast
  - Application that needs extremely low latency and high network throughput

# Placement Groups Spread



- Pros:

- Can span across Availability Zones (AZ)
- Reduced risk of simultaneous failure
- EC2 Instances are on different physical hardware

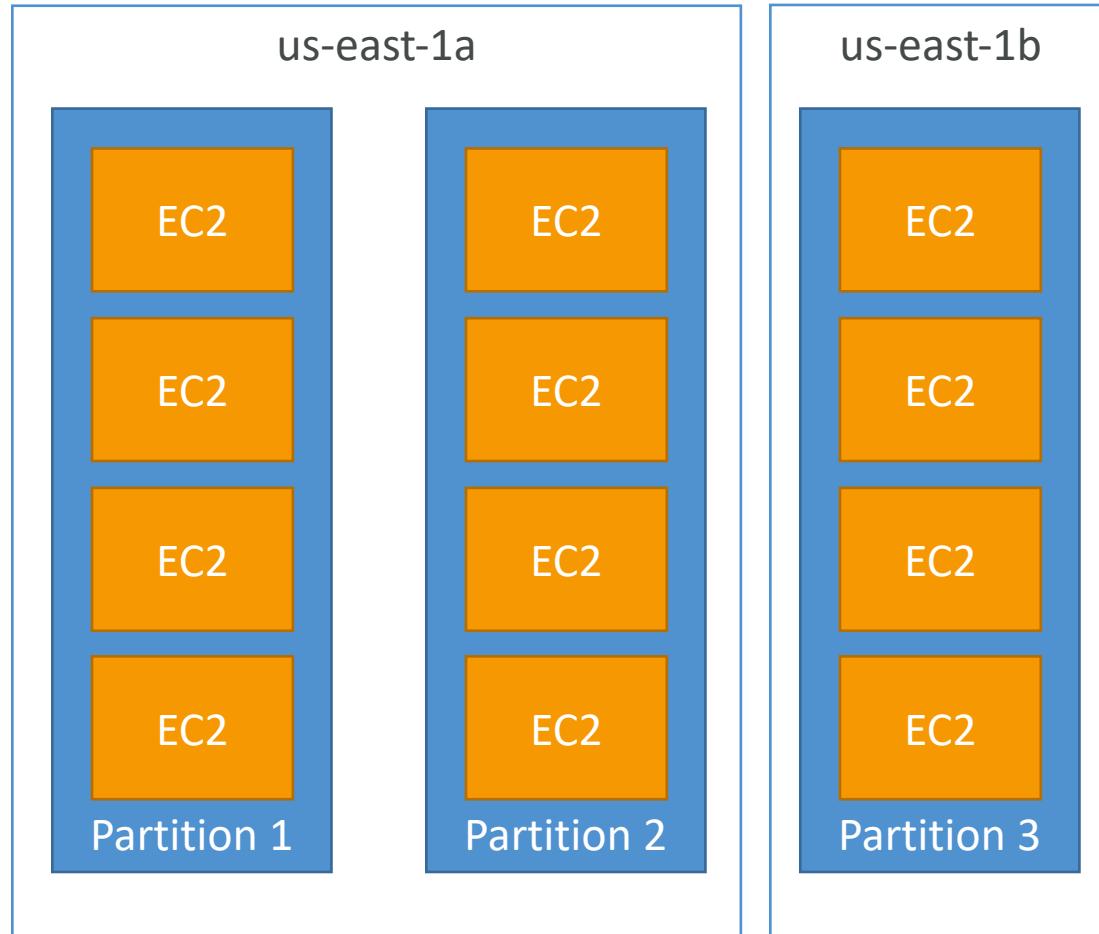
- Cons:

- Limited to 7 instances per AZ per placement group

- Use case:

- Application that needs to maximize high availability
- Critical Applications where each instance must be isolated from failure from each other

# Placements Groups Partition



- Up to 7 partitions per AZ
- Can span across multiple AZs in the same region
- Up to 100s of EC2 instances
- The instances in a partition do not share racks with the instances in the other partitions
- A partition failure can affect many EC2 but won't affect other partitions
- EC2 instances get access to the partition information as metadata
- Use cases: HDFS, HBase, Cassandra, Kafka

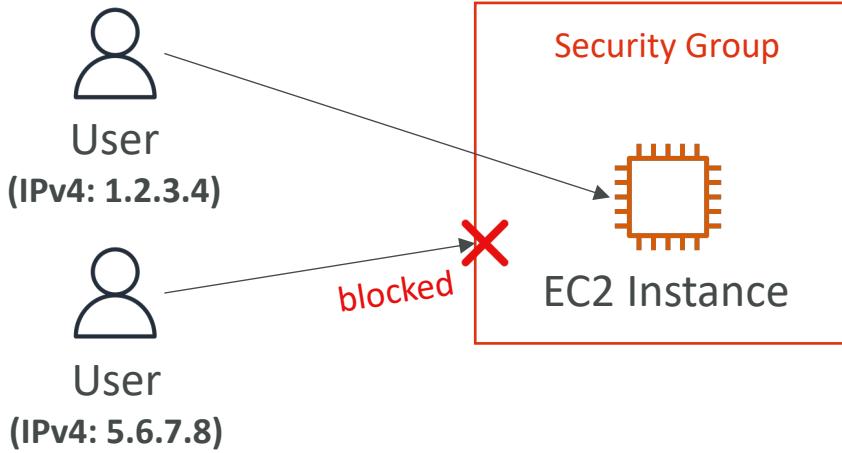
# EC2 SSH troubleshooting

- Make sure the private key (pem file) on your linux machine has 400 permissions, else you will get “**Unprotected private key file**” error
- Make sure the username for the OS is given correctly when logging via SSH, else you will get “**Host key not found**”, “**Permission denied**”, or “**Connection closed by [instance] port 22**” error
- Possible reasons for “**Connection timed out**” to EC2 instance via SSH:
  - SG is not configured correctly
  - NACL is not configured correctly
  - Check the route table for the subnet (routes traffic destined outside VPC to IGW)
  - Instance doesn’t have a public IPv4
  - CPU load of the instance is high

# SSH vs. EC2 Instance Connect

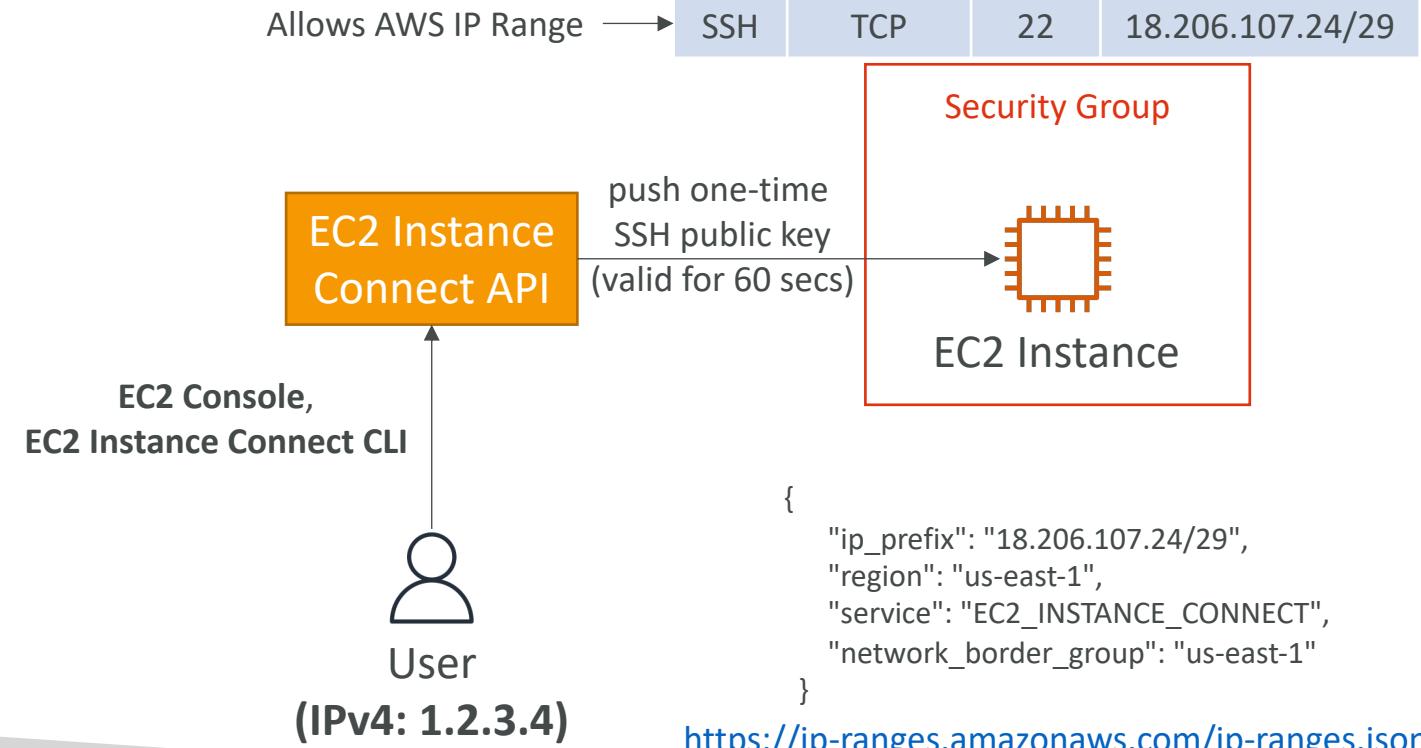
## Connect using SSH

Inbound Rules			
Type	Protocol	Port	Source
SSH	TCP	22	1.2.3.4/32



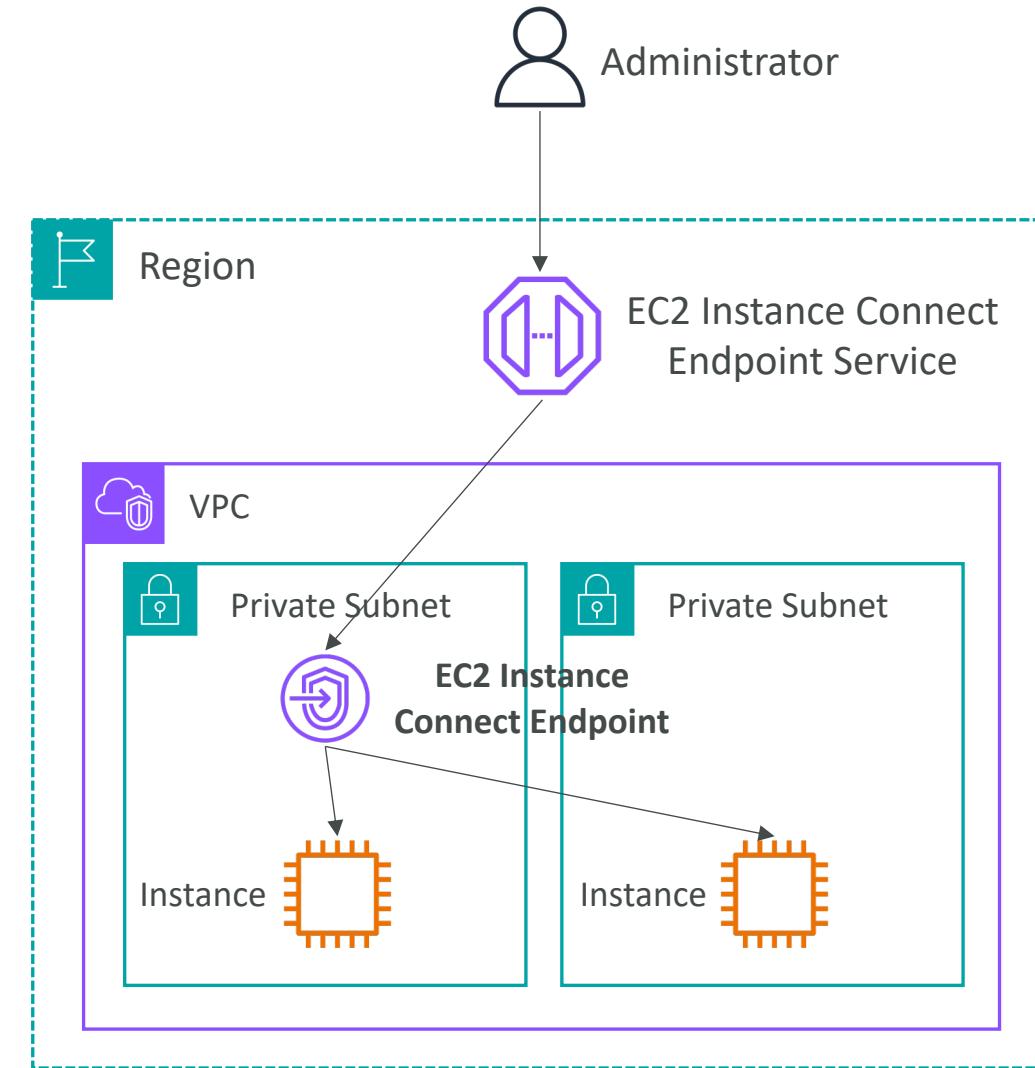
## Connect using EC2 Instance Connect

Inbound Rules			
Type	Protocol	Port	Source
SSH	TCP	22	18.206.107.24/29



# EC2 Instance Connect (EIC) Endpoint

- Allows you to connect securely to your **private** EC2 instances
- No Internet Gateway, no NAT Gateway, No Internet required
- **EIC Endpoint Security Group:** must allow outbound SSH traffic to target instances
- **EC2 Instance Security Group:** must allow inbound SSH traffic from EIC Endpoint Security Group



# CloudWatch Metrics for EC2

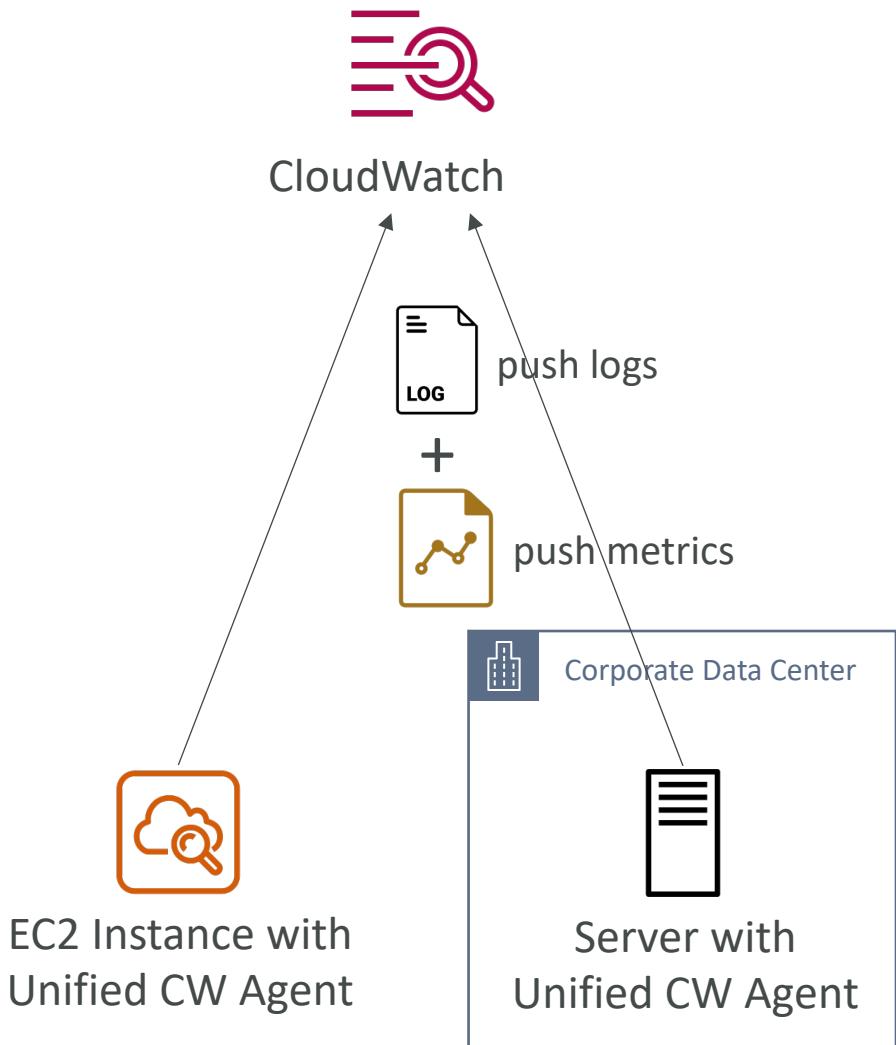
- AWS Provided metrics (AWS pushes them):
  - Basic Monitoring (default): metrics are collected at a 5 minute internal
  - Detailed Monitoring (paid): metrics are collected at a 1 minute interval
  - Includes CPU, Network, Disk and Status Check Metrics
- Custom metric (yours to push):
  - Basic Resolution: 1 minute resolution
  - High Resolution: all the way to 1 second resolution
  - Include RAM, application level metrics
  - Make sure the IAM permissions on the EC2 instance role are correct !

# EC2 included metrics

- CPU: CPU Utilization + Credit Usage / Balance
- Network: Network In / Out
- Status Check:
  - Instance status = check the EC2 VM
  - System status = check the underlying hardware
  - Attached EBS status = check attached EBS volumes
- Disk: Read / Write for Ops / Bytes (only for instance store)
- RAM is NOT included in the AWS EC2 metrics

# Unified CloudWatch Agent

- For virtual servers (EC2 instances, on-premises servers, ...)
- Collect additional system-level metrics such as RAM, processes, used disk space, etc.
- Collect logs to send to CloudWatch Logs
  - No logs from inside your EC2 instance will be sent to CloudWatch Logs without using an agent
- Centralized configuration using SSM Parameter Store
- Make sure IAM permissions are correct
- Default namespace for metrics collected by the Unified CloudWatch agent is **CWAgent** (can be configured/changed)

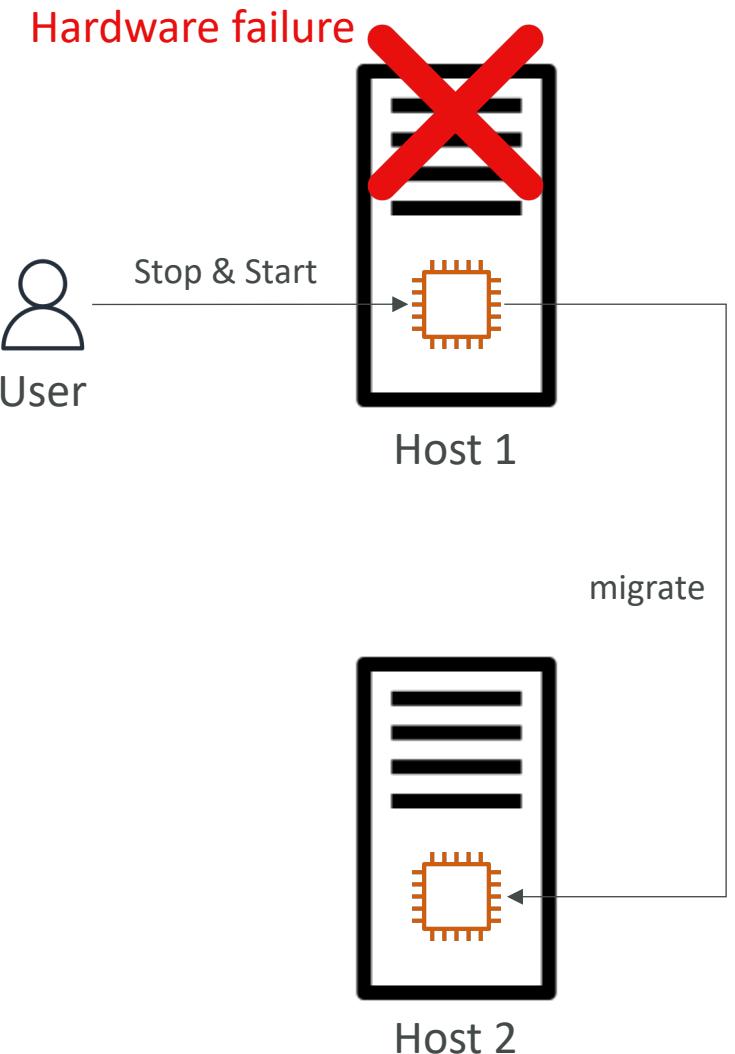


# Unified CloudWatch Agent – procstat Plugin

- Collect metrics and monitor system utilization of individual processes
- Supports both Linux and Windows servers
- Example: amount of time the process uses CPU, amount of memory the process uses, ...
- Select which processes to monitor by
  - `pid_file`: name of process identification number (PID) files they create
  - `exe`: process name that match string you specify (RegEx)
  - `pattern`: command lines used to start the processes (RegEx)
- Metrics collected by procstat plugin begins with “`procstat`” prefix (e.g., `procstat_cpu_time`, `procstat_cpu_usage`, ...)

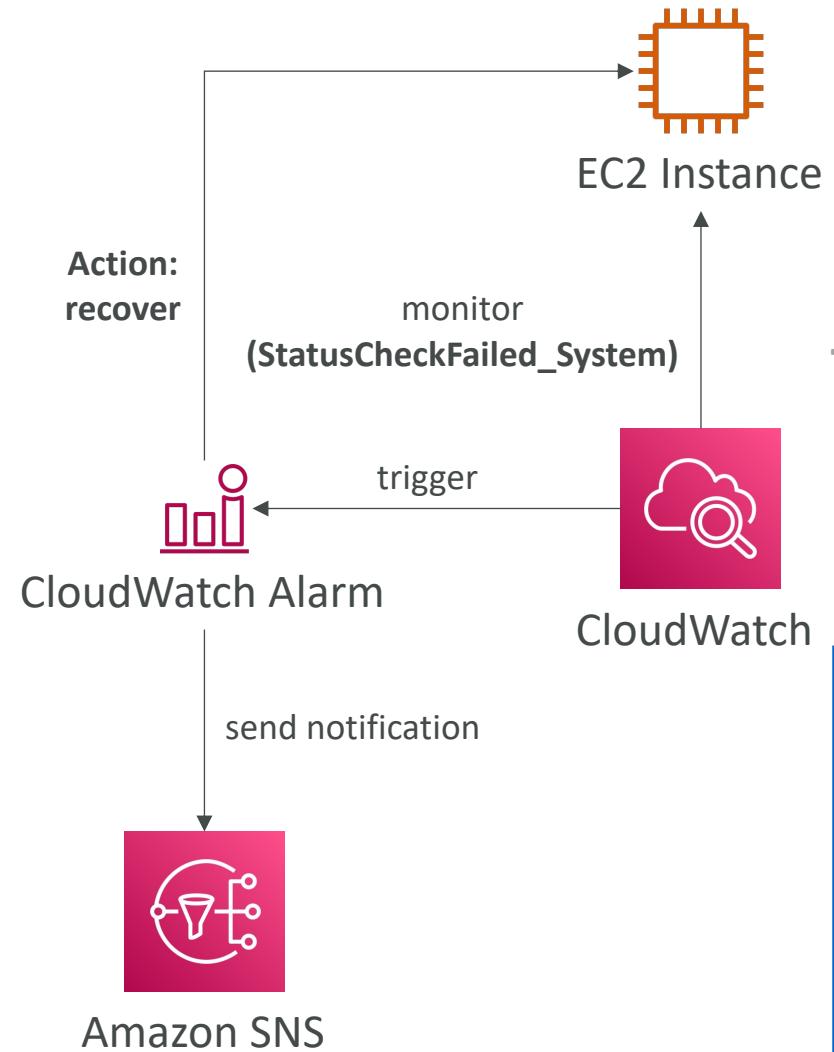
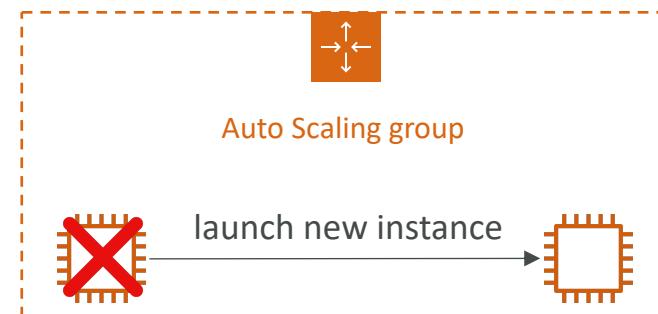
# Status Checks

- Automated checks to identify hardware and software issues
- **System Status Checks**
  - Monitors problems with AWS systems (software/hardware issues on the physical host, loss of system power, ...)
  - Check **Personal Health Dashboard** for any scheduled critical maintenance by AWS to your instance's host
  - Resolution: stop and start the instance (instance migrated to a new host)
- **Instance Status Checks**
  - Monitors software/network configuration of your instance (invalid network configuration, exhausted memory, ...)
  - Resolution: reboot the instance or change instance configuration
- **Attached EBS Status Checks**
  - Monitors EBS volumes attached to your instance (reachable & complete I/O Operations)
  - Resolution: reboot the instance or replace affected EBS volumes



# Status Checks - CW Metrics & Recovery

- CloudWatch Metrics (1 minute interval)
  - StatusCheckFailed\_System
  - StatusCheckFailed\_Instance
  - StatusCheckFailed\_AttachedEBS
  - StatusCheckFailed (for any)
- Option 1: CloudWatch Alarm
  - Recover EC2 instance with the same private/public IP, EIP, metadata, and Placement Group
  - Send notifications using SNS
- Option 2: Auto Scaling Group
  - Set min/max/desired 1 to recover an instance but won't keep the same private and elastic IP

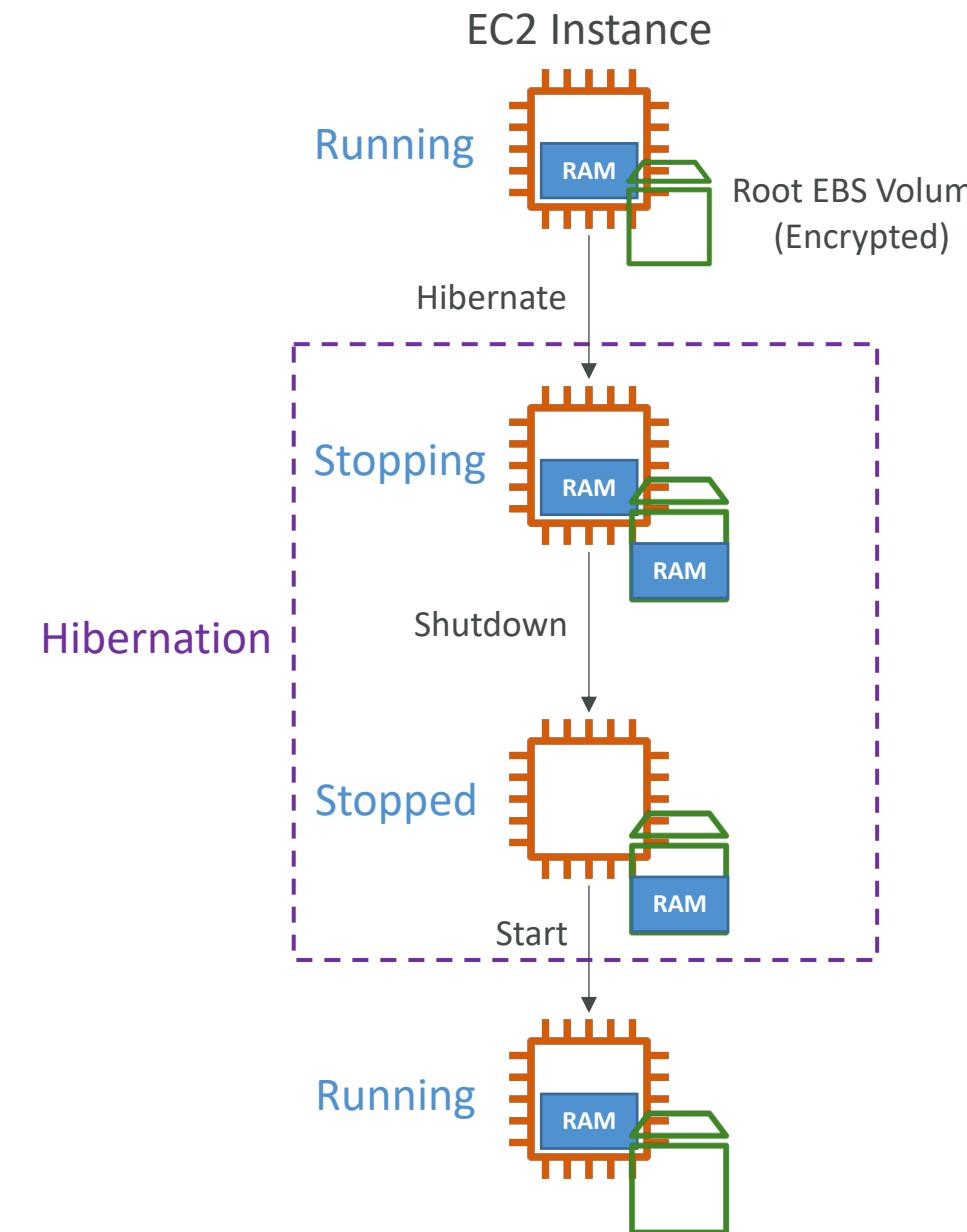


# EC2 Hibernate

- We know we can stop, terminate instances
  - Stop – s the data on disk (EBS) is kept intact in the next start
  - Terminate – any EBS volumes (root) also set-up to be destroyed is lost
- On start, the following happens:
  - First start: the OS boots & the EC2 User Data script is run
  - Following starts: the OS boots up
  - Then your application starts, caches get warmed up, and that can take time!

# EC2 Hibernate

- Introducing EC2 Hibernate:
  - The in-memory (RAM) state is preserved
  - The instance boot is much faster! (the OS is not stopped / restarted)
  - Under the hood: the RAM state is written to a file in the root EBS volume
  - The root EBS volume must be encrypted
- Use cases:
  - Long-running processing
  - Saving the RAM state
  - Services that take time to initialize

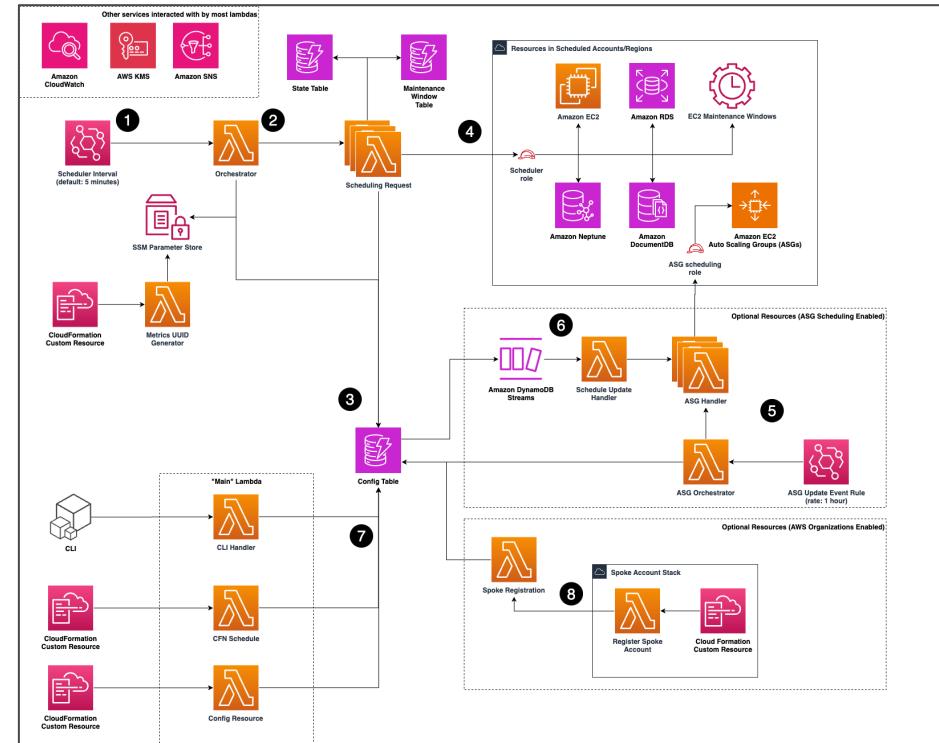


# EC2 Hibernate – Good to know

- Supported Instance Families – C3, C4, C5, I3, M3, M4, R3, R4, T2, T3, ...
- Instance RAM Size – must be less than 150 GB.
- Instance Size – not supported for bare metal instances.
- AMI – Amazon Linux 2, Linux AMI, Ubuntu, RHEL, CentOS & Windows...
- Root Volume – must be EBS, encrypted, not instance store, and large
- Available for On-Demand, Reserved and Spot Instances
- An instance can NOT be hibernated more than 60 days

# Instance Scheduler on AWS

- AWS solution deployed through CloudFormation (not a service)
- Automatically start/stop your AWS services to reduce costs (up to 70%)
- Example: stop company's EC2 instances outside business hours
- Supports EC2 instances, EC2 Auto Scaling Groups, and RDS instances
- Schedules are managed in a DynamoDB table
- Uses resources' tags and Lambda to stop/start instances
- Supports cross-account and cross-region resources
- <https://aws.amazon.com/solutions/implementations/instance-scheduler-on-aws/>



# Amazon Machine Image (AMI)

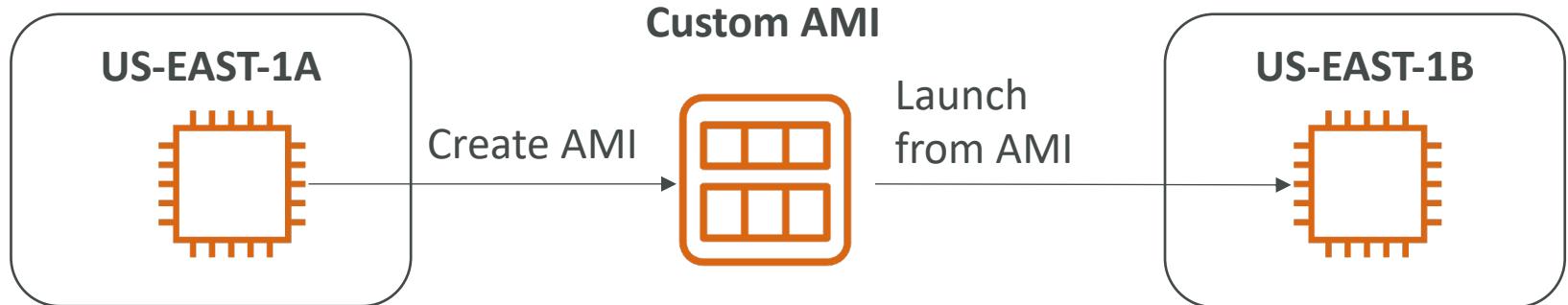


# AMI Overview

- 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)

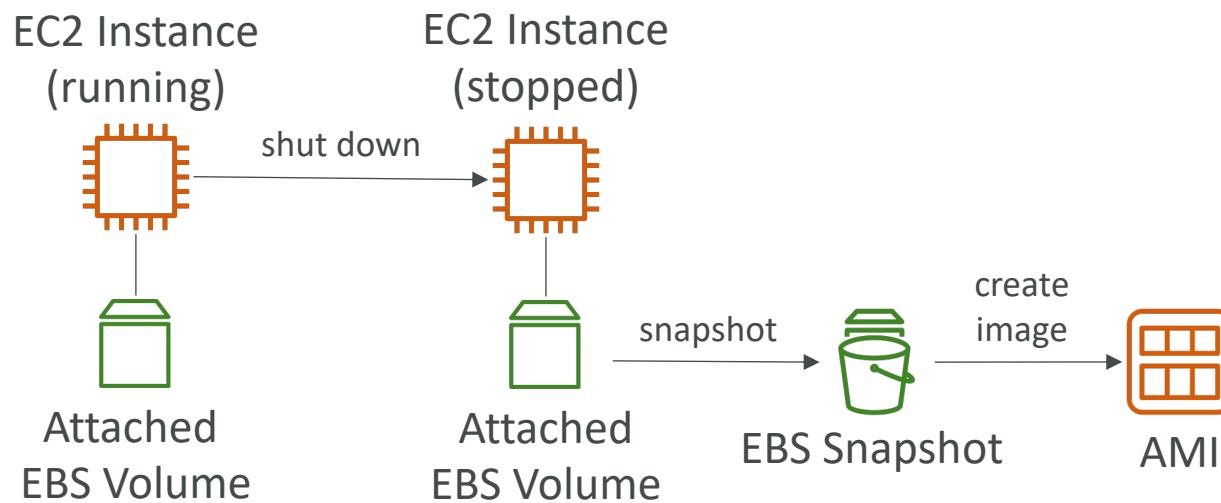
- Start an EC2 instance and customize it
- Stop the instance (for data integrity)
- Build an AMI – this will also create EBS snapshots
- Launch instances from other AMIs



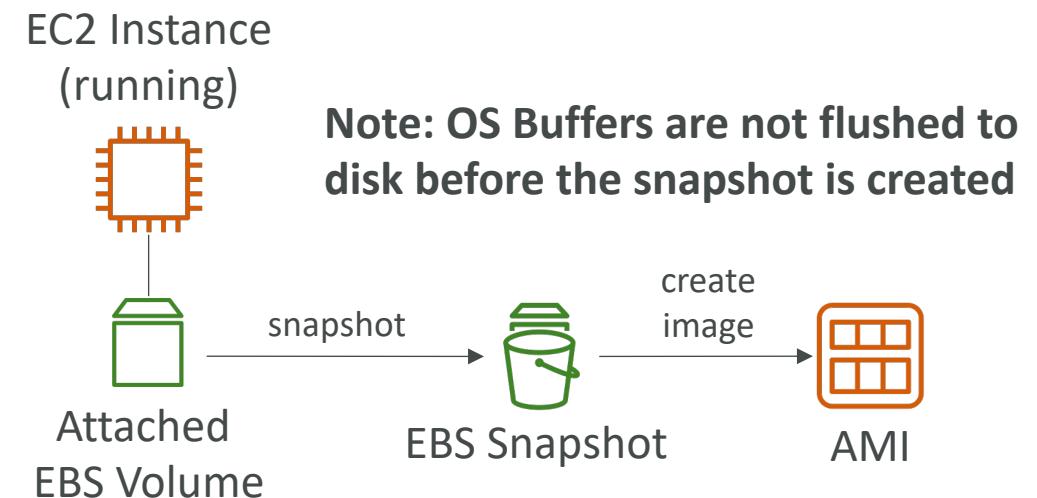
# AMI No-Reboot Option

- Enables you to create an AMI without shutting down your instance
- By default, it's not selected (AWS will shut down the instance before creating an AMI to maintain the file system integrity)

## With No-Reboot Disabled (default)

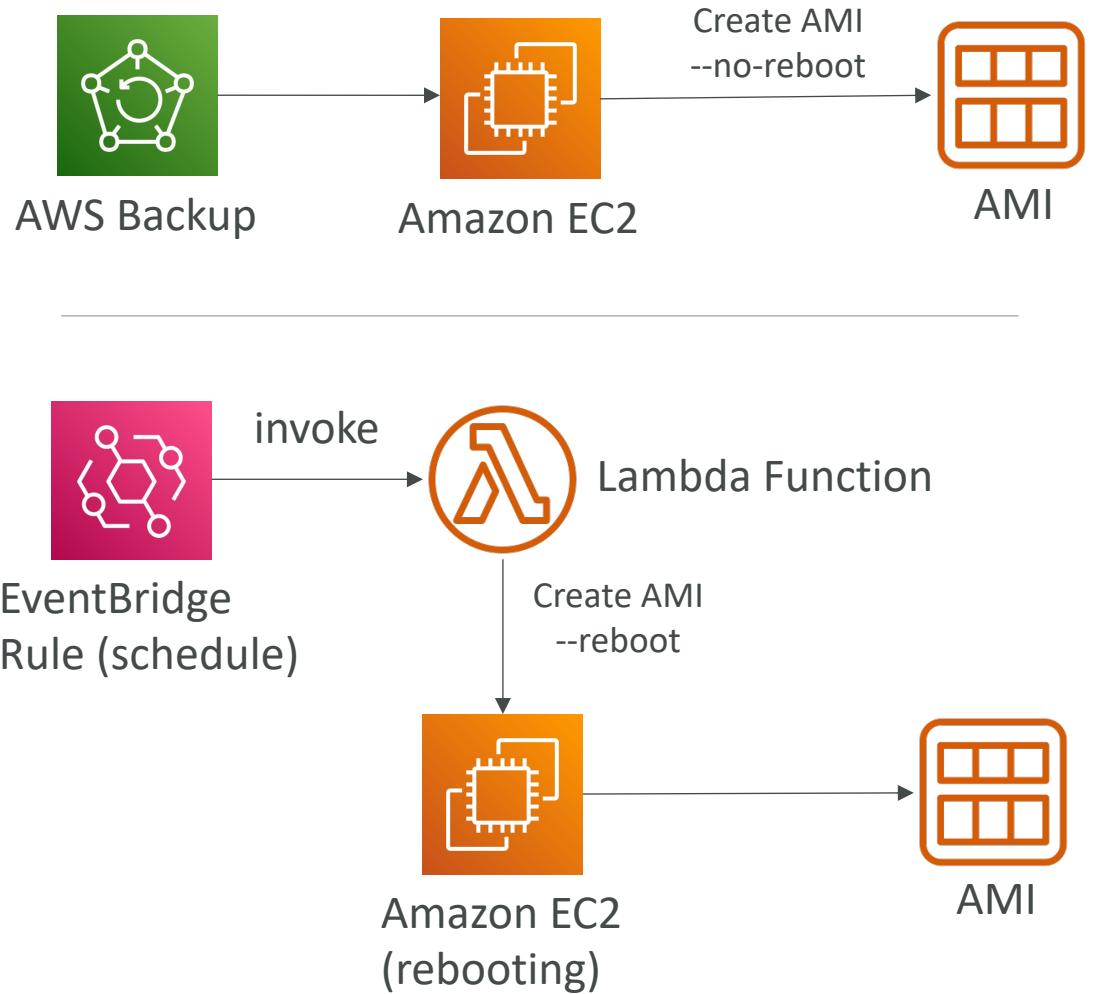


## With No-Reboot Enabled

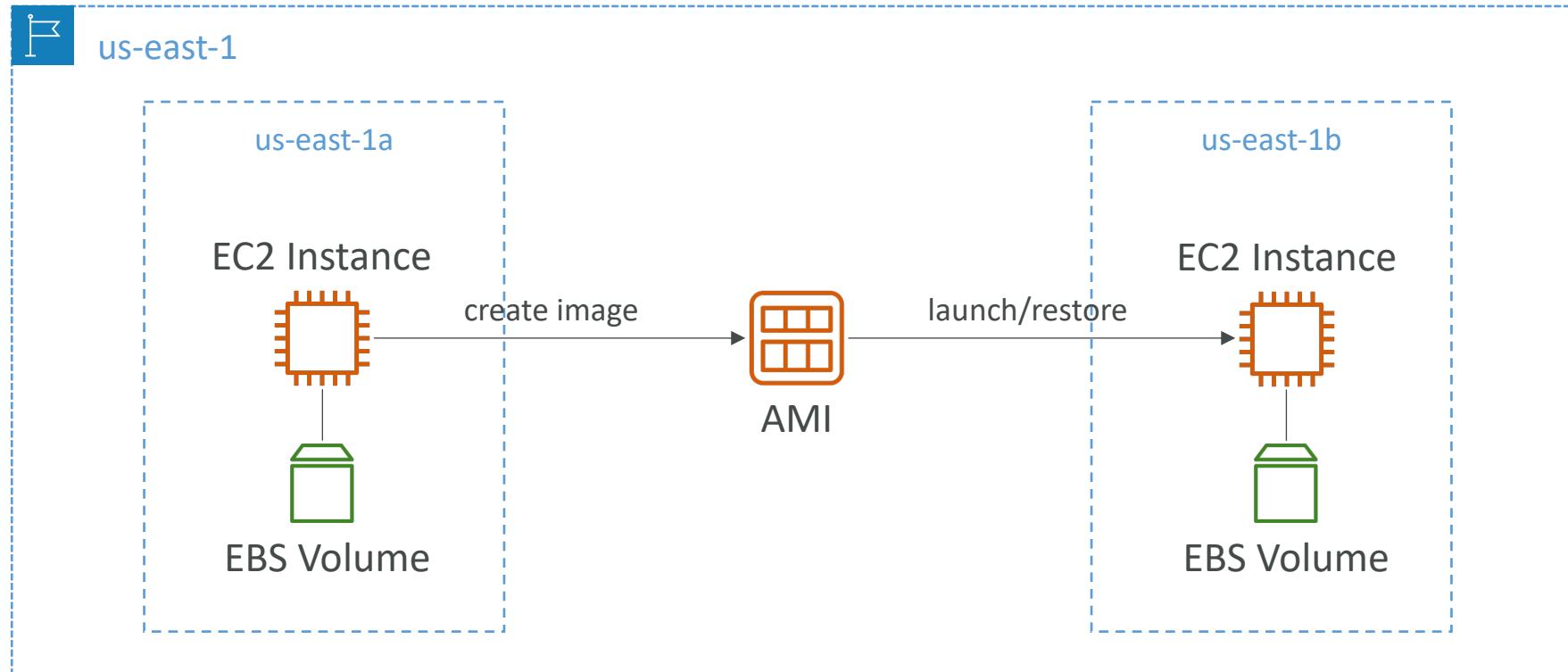


# AWS Backup Plans to create AMI

- AWS Backup doesn't reboot the instances while taking EBS snapshots (no-reboot behavior)
  - This won't help you to create an AMI that guarantees file system integrity since you need to reboot the instance
  - To maintain integrity you need to provide the reboot parameter while taking images (EventBridge + Lambda + CreateImage API with reboot)

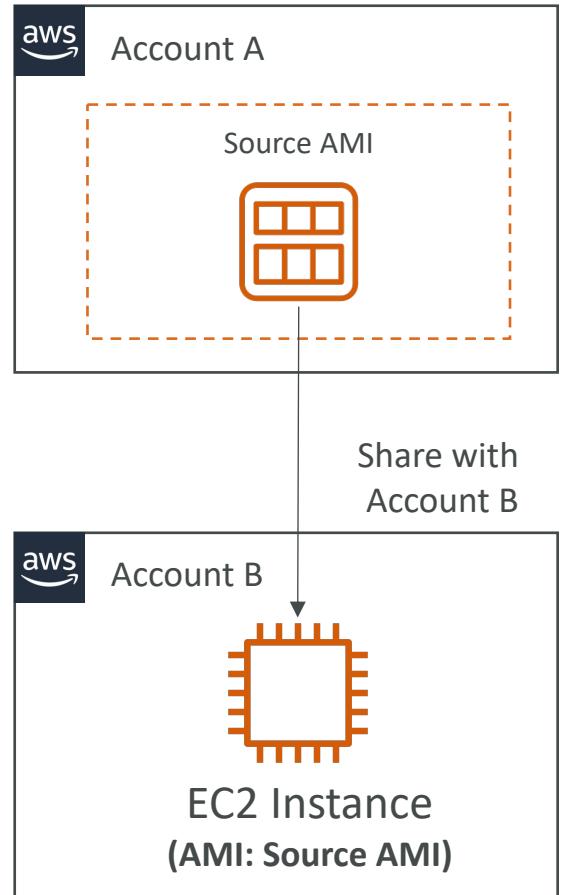


# EC2 Instance Migration between AZ

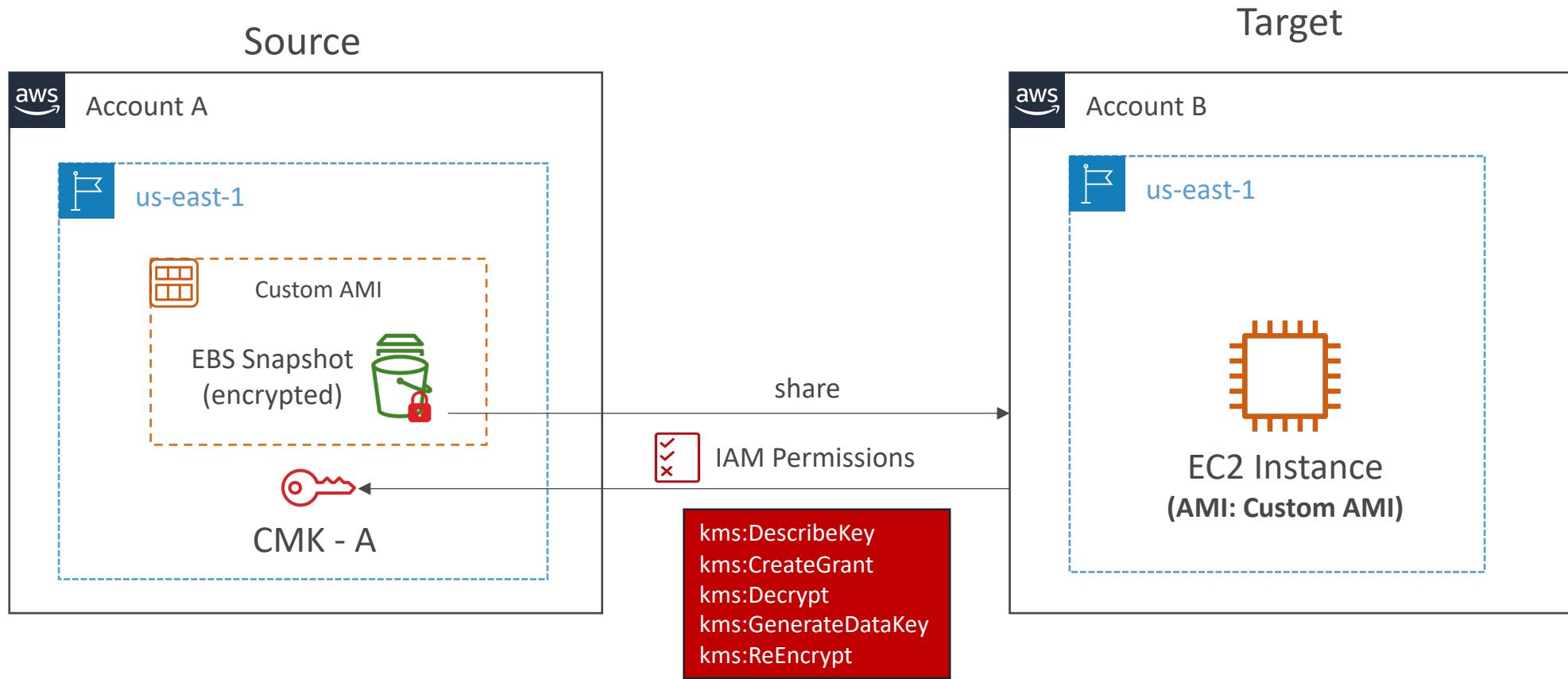


# Cross-Account AMI Sharing

- You can share an AMI with another AWS account
- Sharing an AMI does not affect the ownership of the AMI
- You can only share AMIs that have unencrypted volumes and volumes that are encrypted with a customer managed key
- If you share an AMI with encrypted volumes, you must also share any customer managed keys used to encrypt them.

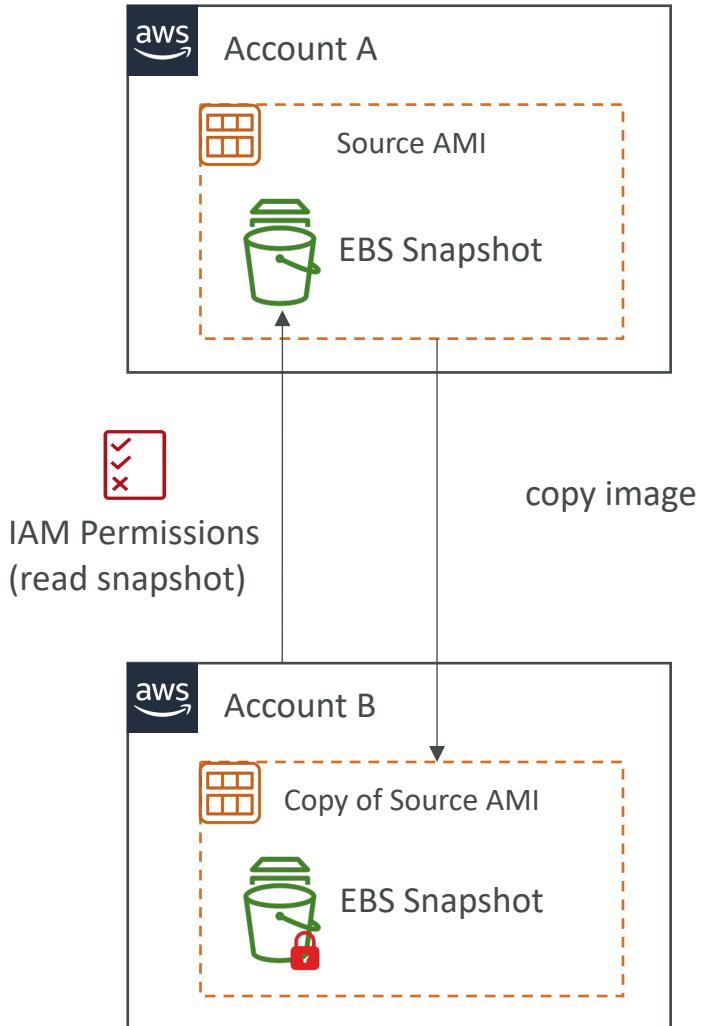


# AMI Sharing with KMS Encryption



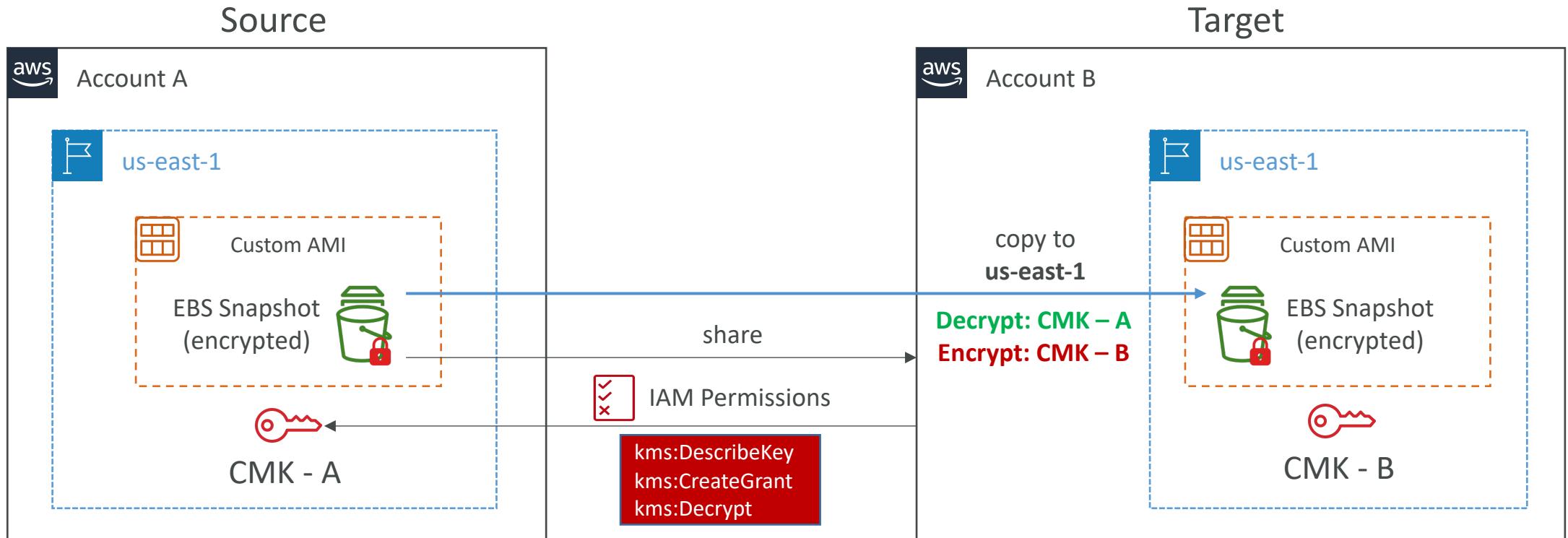
# Cross-Account AMI Copy

- If you copy an AMI that has been shared with your account, you are the owner of the target AMI in your account
- The owner of the source AMI must grant you read permissions for the storage that backs the AMI (EBS Snapshot)
- If the shared AMI has encrypted snapshots, the owner must share the key or keys with you as well
- Can encrypt the AMI with your own CMK while copying



# AMI Copy with KMS Encryption

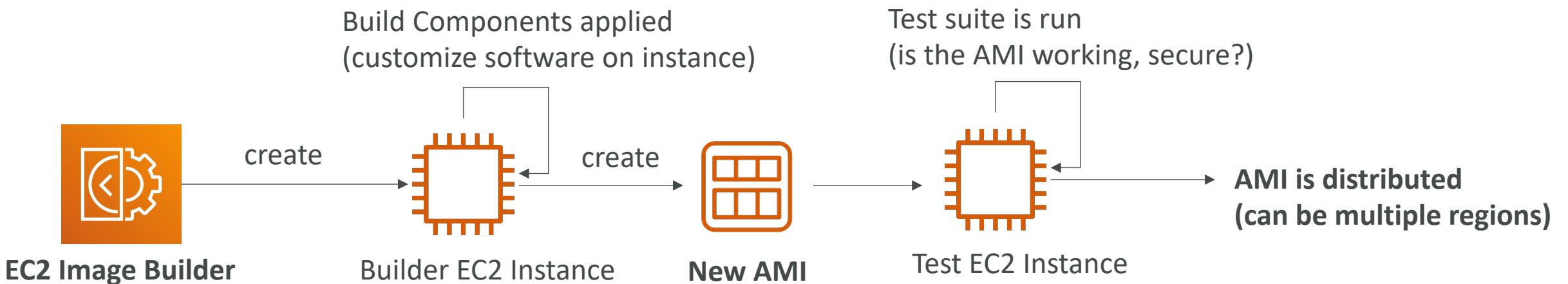
## Cross-Region / Cross-Account Encrypted AMI Copy



# EC2 Image Builder

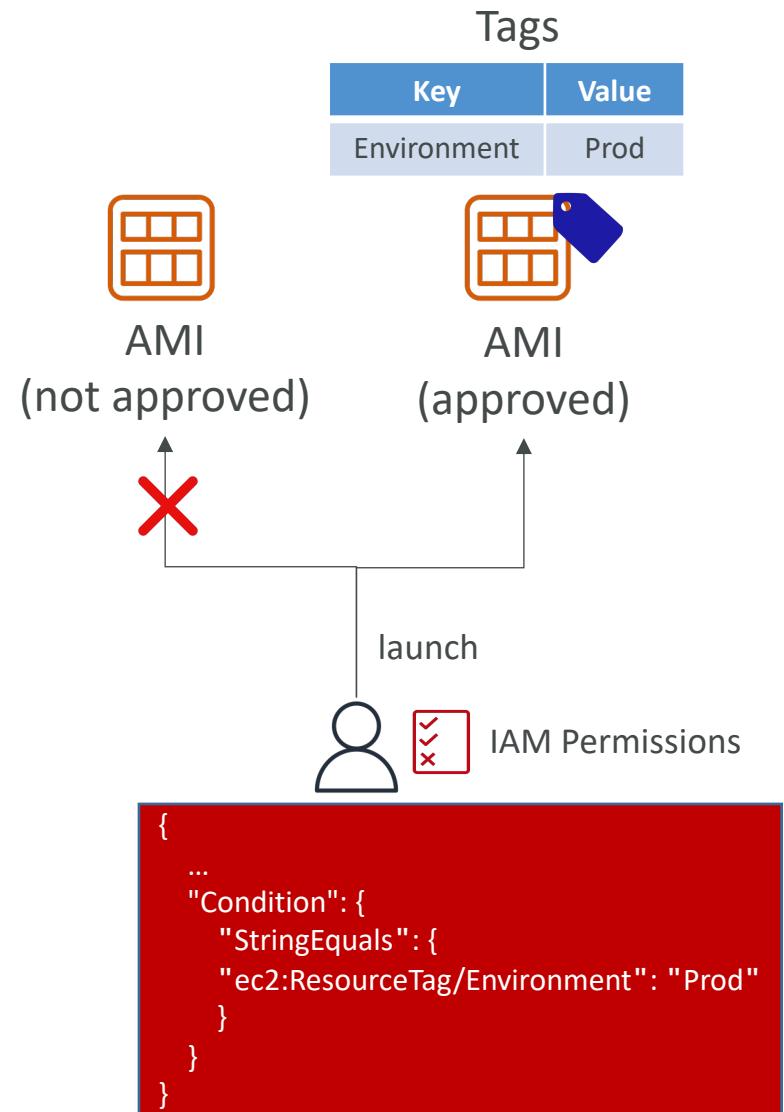
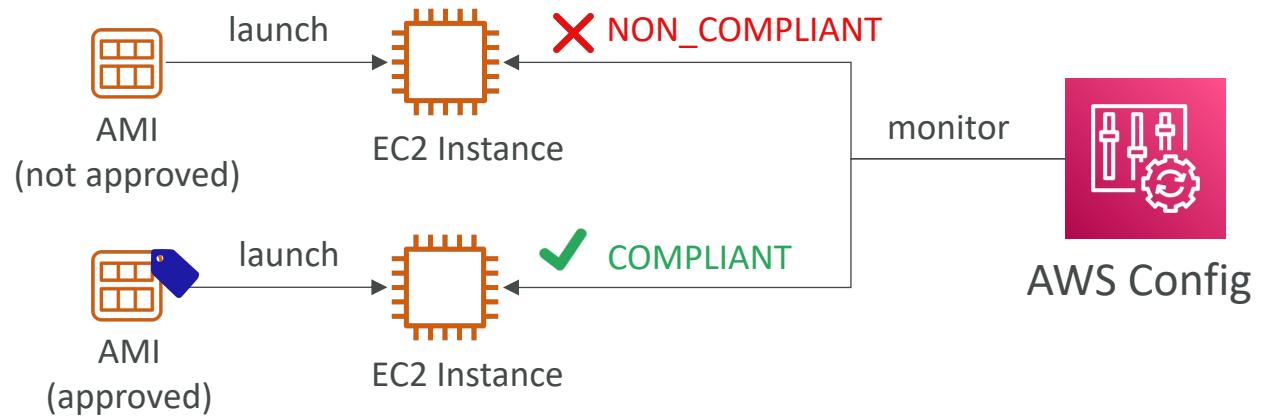


- Used to automate the creation of Virtual Machines or container images
- => Automate the creation, maintain, validate and test **EC2 AMIs**
- Can be run on a schedule (weekly, whenever packages are updated, etc...)
- Free service (only pay for the underlying resources)



# AMI in Production

- You can force users to only launch EC2 instances from pre-approved AMIs (AMIs tagged with specific tags) using IAM policies
- Combine with AWS Config to find non-compliant EC2 instance (instances launched with non-approved AMIs)



# Management of EC2 at Scale

Systems Manager

# AWS Systems Manager Overview



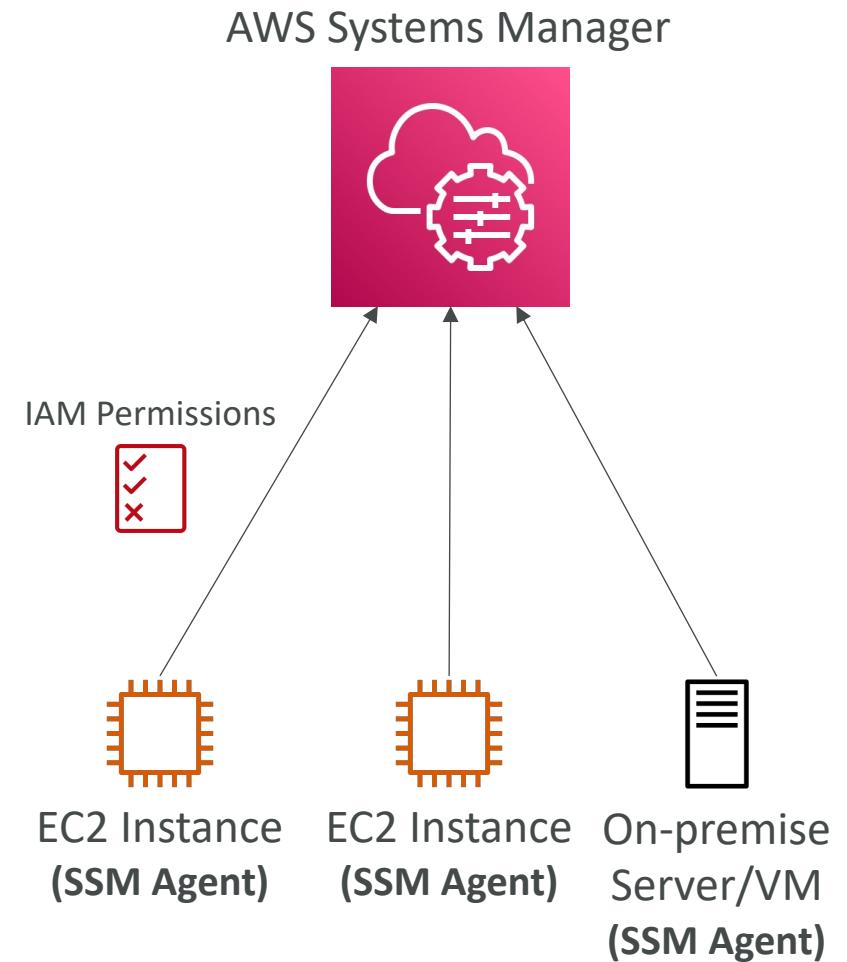
- Helps you manage your **EC2** and **On-Premises** systems at scale
- Get operational insights about the state of your infrastructure
- Easily detect problems
- **Patching automation for enhanced compliance**
- Works for both Windows and Linux OS
- Integrated with CloudWatch metrics / dashboards
- Integrated with AWS Config
- Free service

# AWS Systems Manager Features

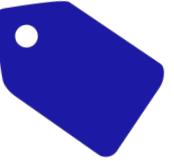
- Node Tools
  - Fleet Manager
  - Compliance
  - Inventory
  - Hybrid Activations
  - Session Manager
  - Run Command
  - State Manager
  - Patch Manager
  - Distributer
- Change Management
  - Automation
  - Change Calendar
  - Maintenance Windows
  - Documents
  - Quick Setup
- Application Tools
  - Application Manager
  - AppConfig
  - Parameter Store
- Resource Groups
- Operations Tools
  - Explorer
  - OpsCenter
  - CloudWatch Dashboard

# How Systems Manager works

- We need to install the SSM agent onto the systems we control
- Installed by default on Amazon Linux 2 AMI & some Ubuntu AMI
- If an instance can't be controlled with SSM, it's probably an issue with the SSM agent!
- Make sure the EC2 instances have a proper IAM role to allow SSM actions



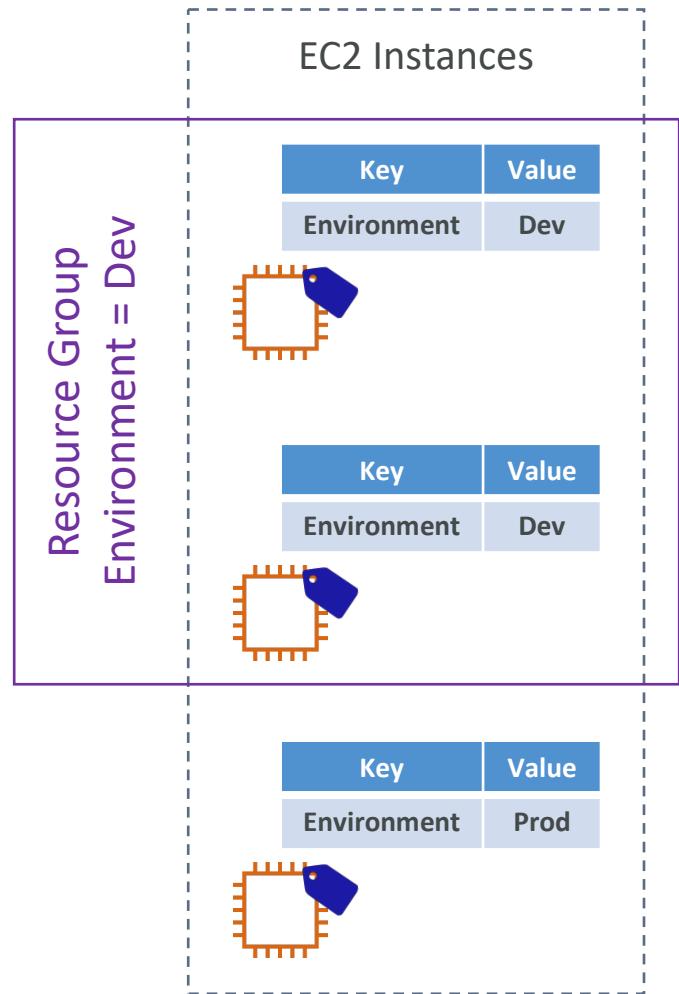
# AWS Tags



- You can add text key-value pairs called Tags to many AWS resources
- Commonly used in EC2
- Free naming, common tags are Name, Environment, Team ...
- They're used for
  - Resource grouping
  - Automation
  - Cost allocation
- Better to have too many tags than too few!

# Resource Groups

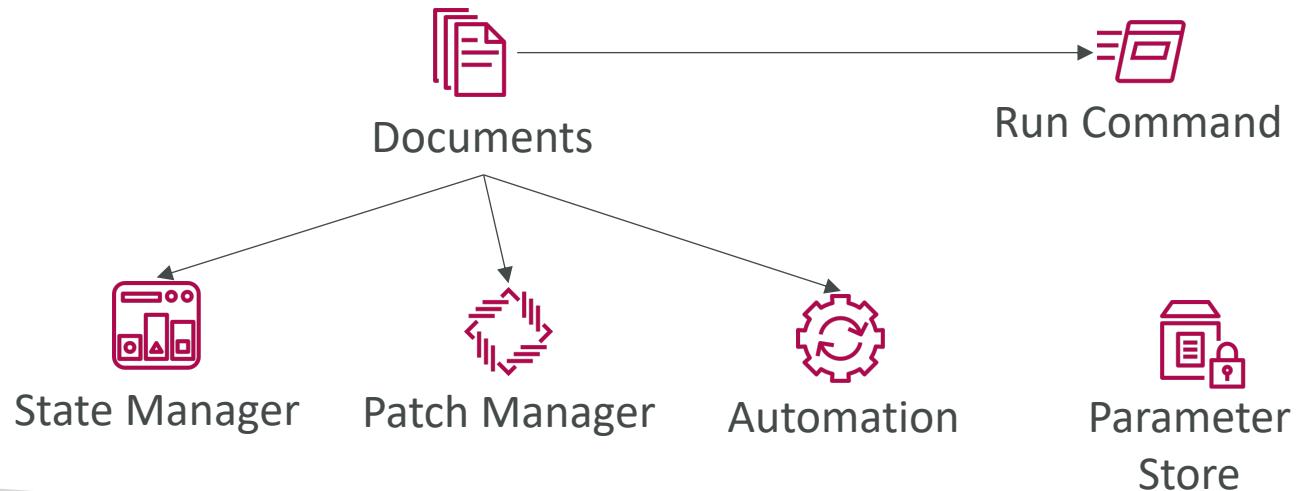
- Create, view or manage logical group of resources thanks to **tags**
- Allows creation of logical groups of resources such as
  - Applications
  - Different layers of an application stack
  - Production versus development environments
- Regional service
- Works with EC2, S3, DynamoDB, Lambda, etc...



# SSM – Documents

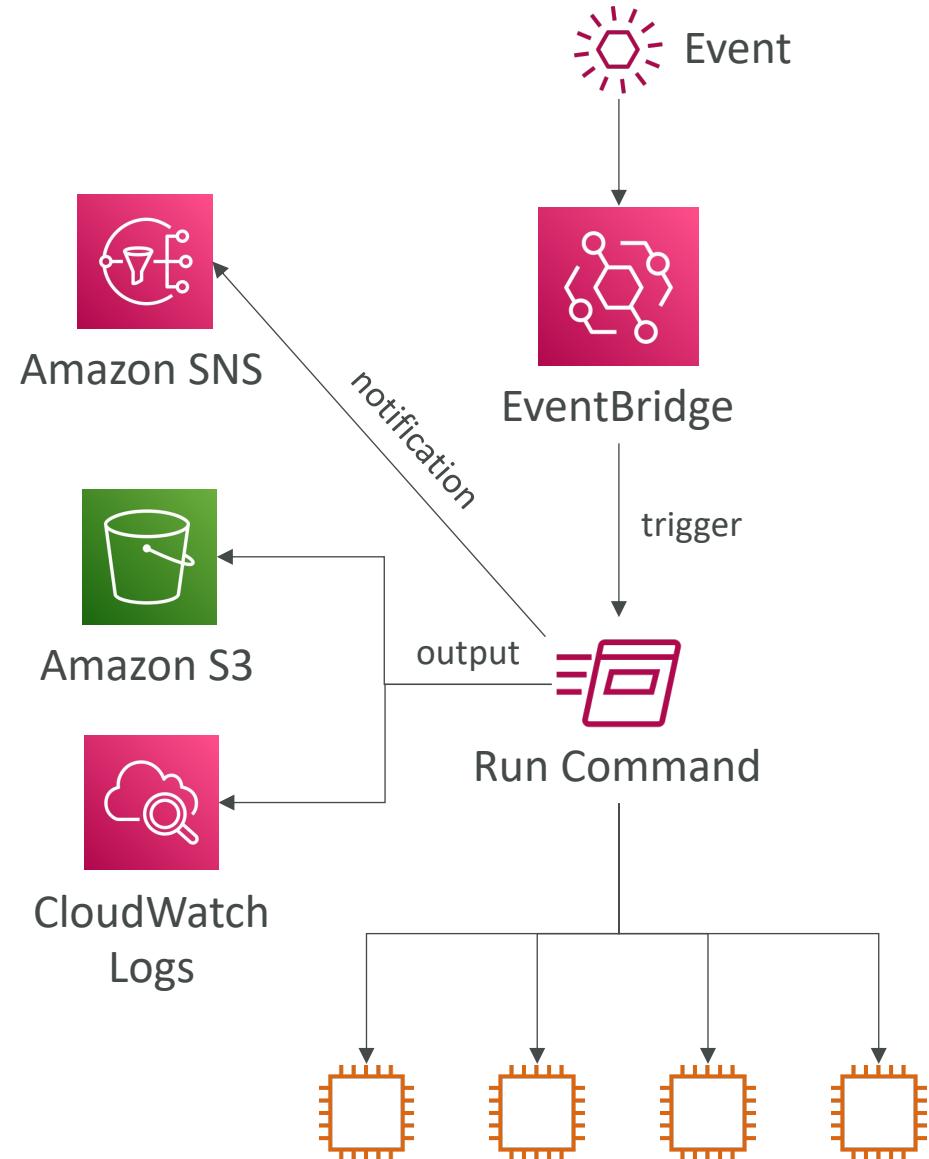
- Documents can be in JSON or YAML
- You define parameters
- You define actions
- Many documents already exist in AWS

```
---  
schemaVersion: '2.2'  
description: State Manager Bootstrap Example  
parameters: {}  
mainSteps:  
  - action: aws:runShellScript  
    name: configureServer  
    inputs:  
      runCommand:  
        - sudo yum install -y httpd  
        - sudo yum --enablerepo=epel install -y clamav
```



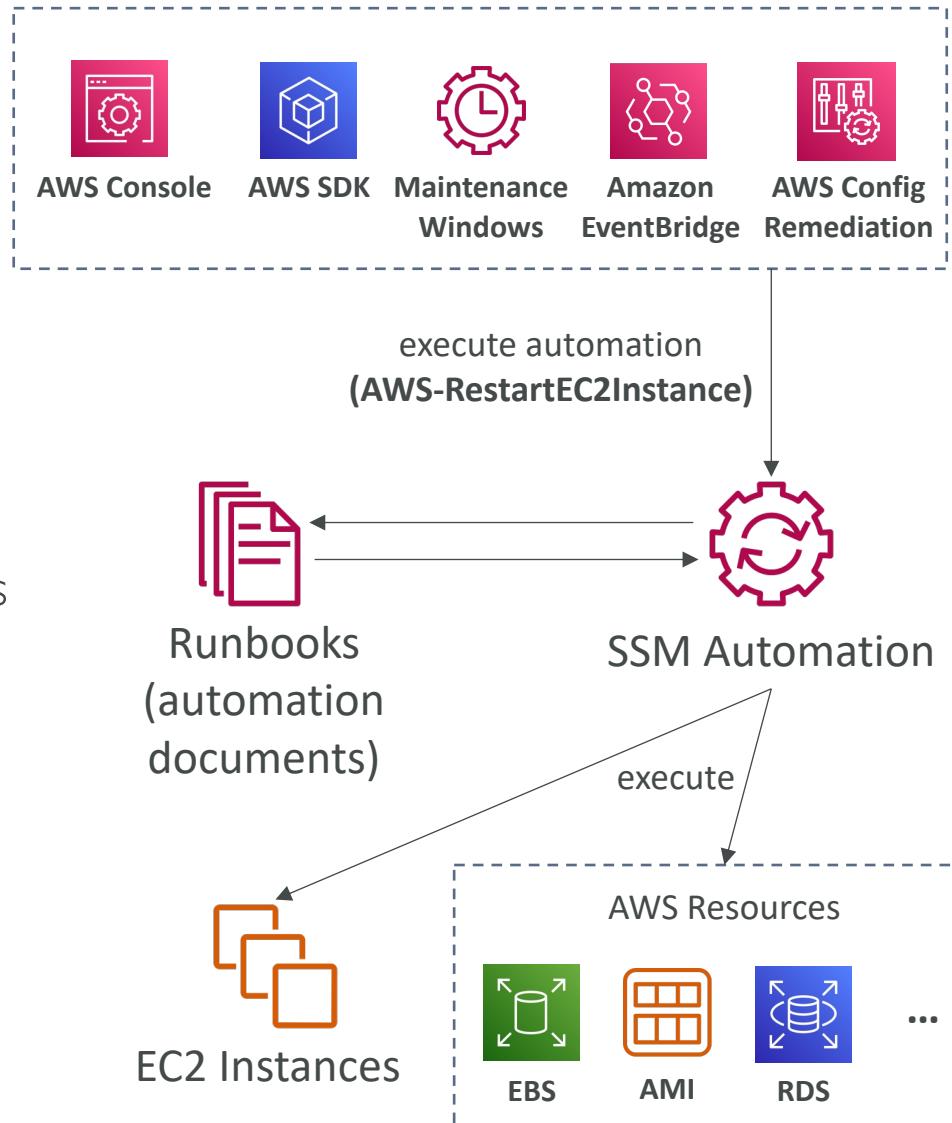
# SSM – Run Command

- Execute a document (= script) or just run a command
- Run command across multiple instances (using resource groups)
- Rate Control / Error Control
- Integrated with IAM & CloudTrail
- No need for SSH
- Command Output can be shown in the Console, sent to S3 bucket or CloudWatch Logs
- Send notifications to SNS about command statuses (In progress, Success, Failed, ...)
- Can be invoked using EventBridge

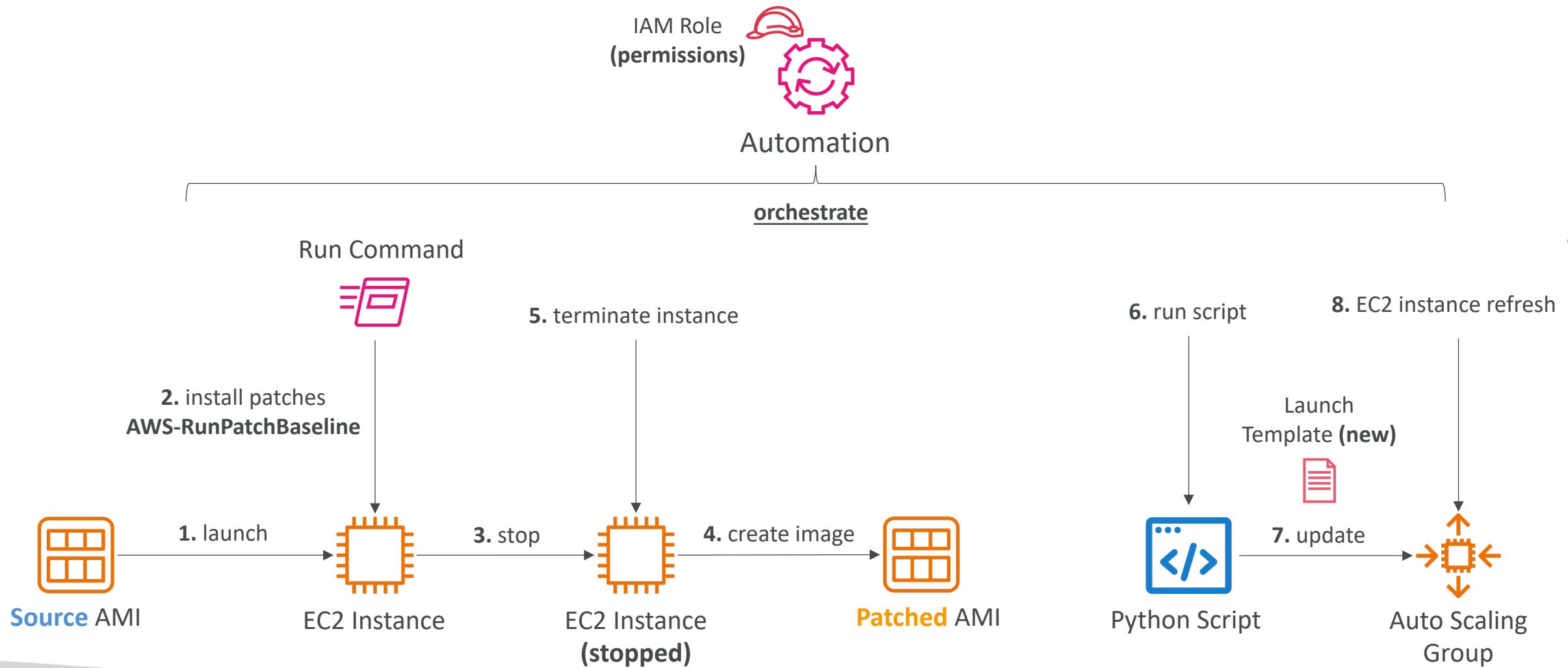


# SSM - Automation

- Simplifies common maintenance and deployment tasks of EC2 instances and other AWS resources
- Example: restart instances, create an AMI, EBS snapshot
- **Automation Runbook**
  - SSM Documents of type Automation
  - Defines actions performed on your EC2 instances or AWS resources
  - Pre-defined runbooks (AWS) or create custom runbooks
- Can be triggered
  - Manually using AWS Console, AWS CLI or SDK
  - By Amazon EventBridge
  - On a schedule using Maintenance Windows
  - By AWS Config for rules remediations

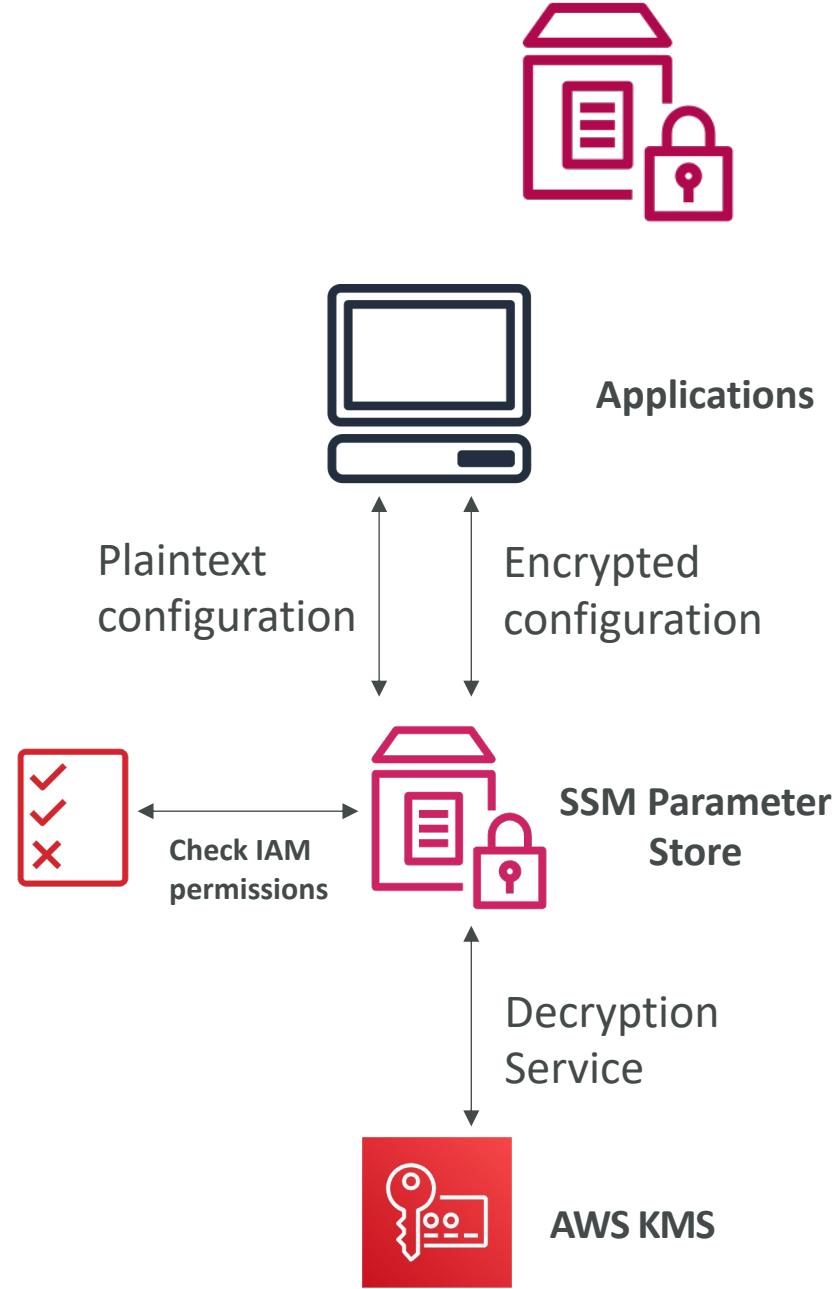


# SSM – Automation – Patch AMIs & Update ASG



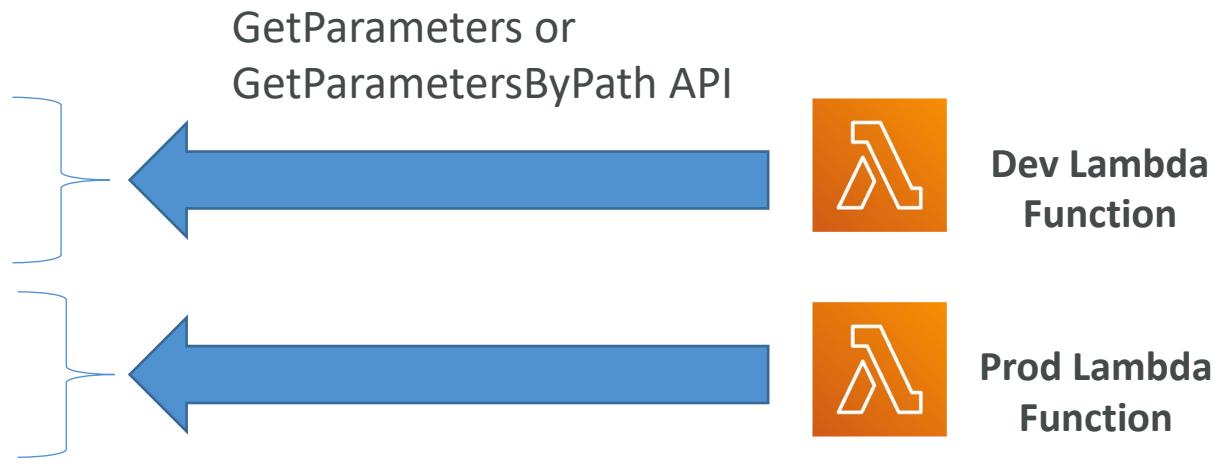
# SSM Parameter Store

- Secure storage for configuration and secrets
- Optional Seamless Encryption using KMS
- Serverless, scalable, durable, easy SDK
- Version tracking of configurations / secrets
- Security through IAM
- Notifications with Amazon EventBridge
- Integration with CloudFormation



# SSM Parameter Store Hierarchy

- /my-department/
  - my-app/
    - dev/
      - db-url
      - db-password
    - prod/
      - db-url
      - db-password
  - other-app/
  - /other-department/
  - /aws/reference/secretsmanager/secret\_ID\_in\_Secrets\_Manager
  - /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86\_64-gp2 (public)



# Standard and advanced parameter tiers

	Standard	Advanced
Total number of parameters allowed (per AWS account and Region)	10,000	100,000
Maximum size of a parameter value	4 KB	8 KB
Parameter policies available	No	Yes
Cost	No additional charge	Charges apply
Storage Pricing	Free	\$0.05 per advanced parameter per month

# Parameters Policies (for advanced parameters)

- Allow to assign a TTL to a parameter (expiration date) to force updating or deleting sensitive data such as passwords
- Can assign multiple policies at a time

## Expiration (to delete a parameter)

```
{  
  "Type": "Expiration",  
  "Version": "1.0",  
  "Attributes": {  
    "Timestamp": "2020-12-02T21:34:33.000Z"  
  }  
}
```

## ExpirationNotification (EventBridge)

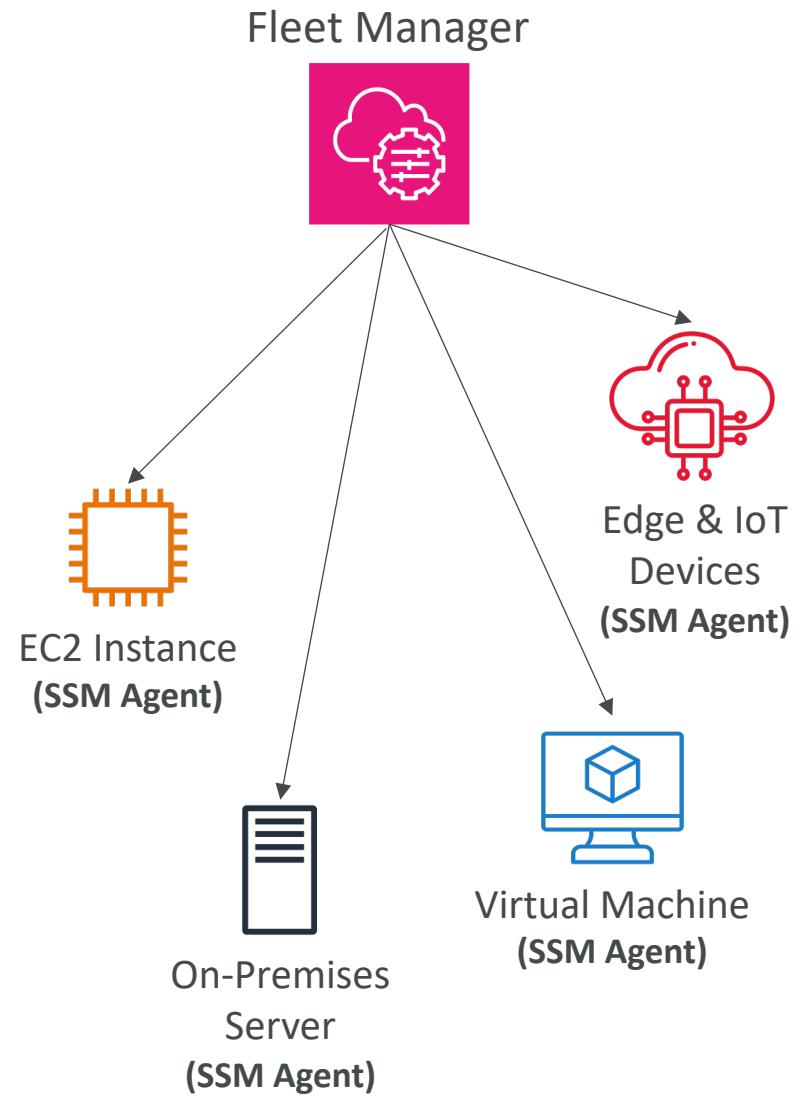
```
{  
  "Type": "ExpirationNotification",  
  "Version": "1.0",  
  "Attributes": {  
    "Before": "15",  
    "Unit": "Days"  
  }  
}
```

## NoChangeNotification (EventBridge)

```
{  
  "Type": "NoChangeNotification",  
  "Version": "1.0",  
  "Attributes": {  
    "After": "20",  
    "Unit": "Days"  
  }  
}
```

# SSM – Fleet Manager

- Centrally & remotely manage your nodes running on AWS or on-premises
- Including EC2 instances, on-premises servers / VMs, edge devices, IoT devices
- Supports different OS (Windows, Linux)
- All nodes **must be have the SSM agent installed**
- AmazonSSMManagedInstanceCore permissions necessary for the EC2 instance role, or use Default Host Management Configuration feature
- Use cases:
  - Track node's status, health, performance
  - Perform troubleshooting and management tasks
  - Perform windows RDP or use Session Manager for CLI

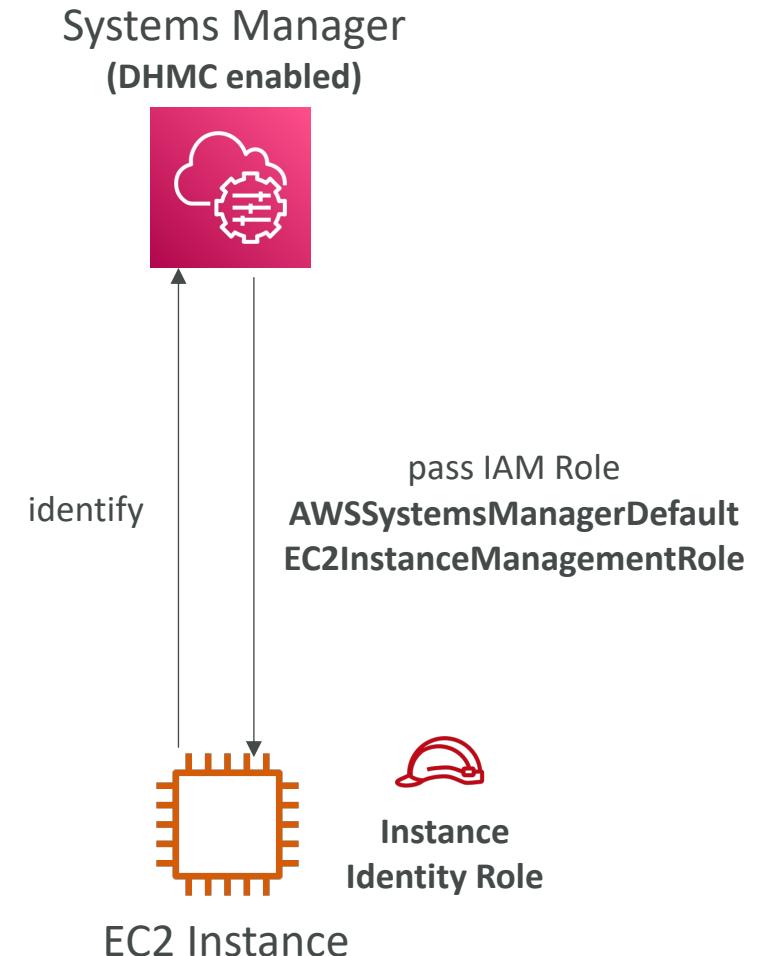


# SSM IAM Permissions

- **AmazonSSMManagedInstanceCore** policy provides the necessary permissions for:
  - Registering the instance with Systems Manager
  - Accessing Session Manager, SSM Documents, SSM Parameters
  - Receiving commands (Run Command)
  - Allow Patching operations (Patch Manager)
  - Report its data to Inventory, Compliance, and config status to SSM
  - Sending heartbeat signals

# Systems Manager Default Host Management Configuration

- When enabled, it automatically configures your EC2 instances as managed instances **without the use of EC2 Instance Profile**
- **Instance Identity Role** – a type of IAM Role with no permissions beyond identifying the EC2 instance to AWS Services (e.g., Systems Manager)
- EC2 instances must have **IMDSv2 enabled** and **SSM Agent installed** (doesn't support IMDSv1)
- Automatically enables Session Manager, Patch Manager, and Inventory
- Automatically keeps the SSM Agent up to date
- Must be enabled per AWS Region





# SSM – Inventory

- Collect metadata from your managed instances (EC2/On-premises)
- Metadata includes installed software, OS drivers, configurations, installed updates, running services ...
- View data in AWS Console or store in S3 and query and analyze using Athena and QuickSight
- Specify metadata collection interval (minutes, hours, days)
- Query data from multiple AWS accounts and regions
- Create Custom Inventory for your custom metadata (e.g., rack location of each managed instance)



# SSM – State Manager

- Automate the process of keeping your managed instances (EC2/On-premises) in a state that you define
- Use cases: bootstrap instances with software, patch OS/software updates on a schedule ...
- **State Manager Association:**
  - Defines the state that you want to maintain to your managed instances
  - Example: port 22 must be closed, antivirus must be installed ...
  - Specify a schedule when this configuration is applied
- Uses SSM Documents to create an Association (e.g., SSM Document to configure CW Agent)



# SSM – Patch Manager

- Automates the process of patching managed instances
- OS updates, applications updates, security updates, ...
- Supports both EC2 instances and on-premises servers
- Supports Linux, macOS, and Windows
- Patch on-demand or on a schedule using **Maintenance Windows**
- Scan instances and generate patch compliance report (missing patches)
- Patch compliance report can be sent to S3



# SSM – Patch Manager

- **Patch Baseline**

- Defines which patches should and shouldn't be installed on your instances
- Ability to create custom Patch Baselines (specify approved/rejected patches)
- Patches can be auto-approved within days of their release
- By default, install only critical patches and patches related to security

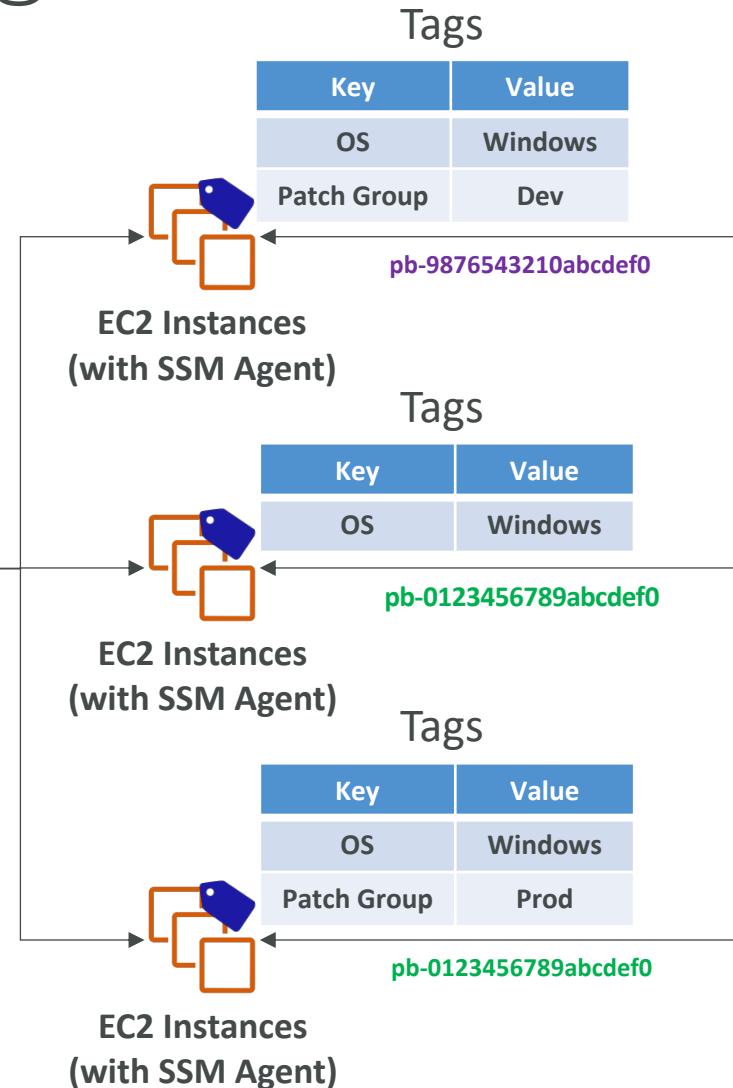
- **Patch Group**

- Associate a set of instances with a specific Patch Baseline
- Example: create Patch Groups for different environments (dev, test, prod)
- Instances should be defined with the tag key **Patch Group**
- An instance can only be in one Patch Group
- Patch Group can be registered with only one Patch Baseline

# SSM – Patch Manager Patch Baselines

- Pre-Defined Patch Baseline
  - Managed by AWS for different Operating Systems (can't be modified)
  - **AWS-RunPatchBaseline (SSM Document)** – apply both operating system and application patches (Linux, macOS, Windows Server)
- Custom Patch Baseline
  - Create your own Patch Baseline and choose which patches to auto-approve
  - Operating System, allowed patches, rejected patches, ...
  - Ability to specify custom and alternative patch repositories

# SSM – Patch Manager



Patch Baseline ID	Patch Group	Default
pb-0123456789abcdef0	Default	Yes
pb-9876543210abcdef0	Dev	No

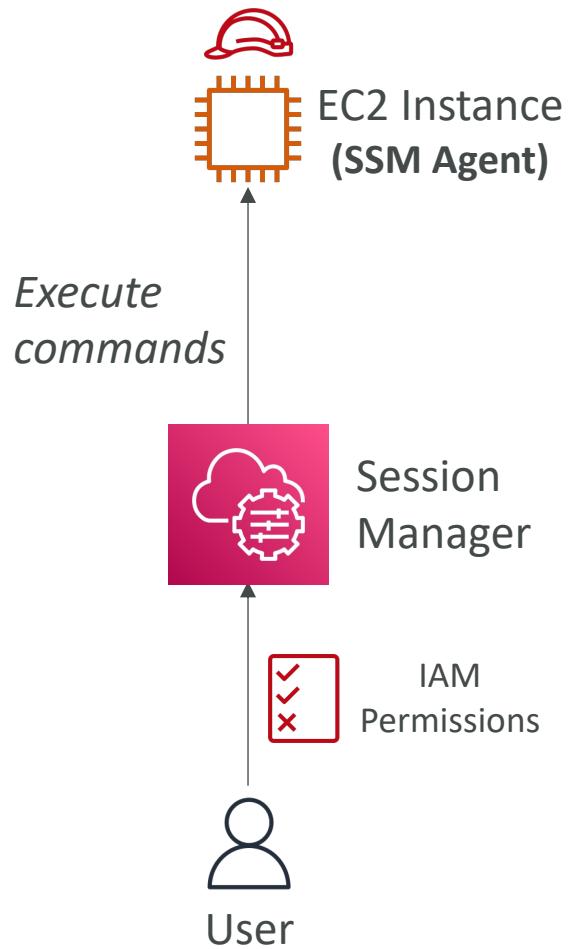


# SSM – Maintenance Windows

- Defines a schedule for when to perform actions on your instances
- Example: OS patching, updating drivers, installing software, ...
- Maintenance Window contains
  - Schedule
  - Duration
  - Set of registered instances
  - Set of registered tasks

# SSM – Session Manager

- Allows you to start a secure shell on your EC2 and on-premises servers
- Access through AWS Console, AWS CLI, or Session Manager SDK
- Does not need SSH access, bastion hosts, or SSH keys
- Supports Linux, macOS, and Windows
- Log connections to your instances and executed commands
- Session log data can be sent to S3 or CloudWatch Logs
- CloudTrail can intercept **StartSession** events



# SSM – Session Manager

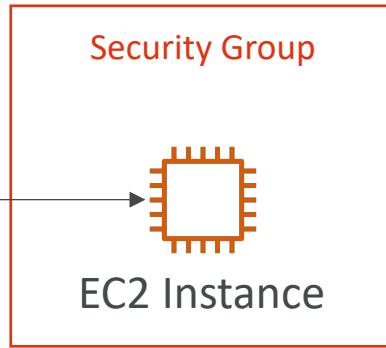
- IAM Permissions
  - Control which users/groups can access Session Manager and which instances
  - Use tags to restrict access to only specific EC2 instances
  - Access SSM + write to S3 + write to CloudWatch
- Optionally, you can restrict commands a user can run in a session

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "ssm:StartSession",  
            "Resource": "arn:aws:ec2:us-east-1:123456789012:instance/*",  
            "Condition": {  
                "StringLike": {  
                    "ssm:resourceTag/Environment": ["Dev"]  
                }  
            }  
        }  
    ]  
}
```

# SSH vs. SSM Session Manager

## Connect using SSH

Inbound Rules			
Type	Protocol	Port	Source
SSH	TCP	22	1.2.3.4/32

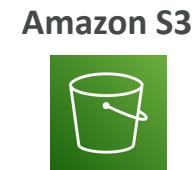


## Connect using SSM Session Manager

### CloudWatch Logs



session log data



AWS Console,  
AWS CLI,  
or Session Manager SDK

Session Manager



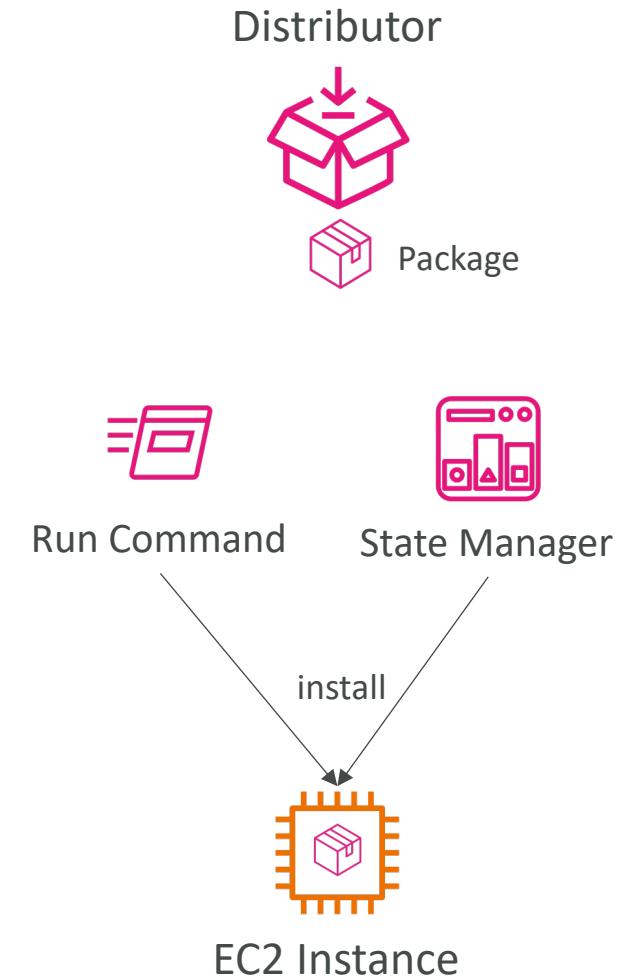
IAM Permissions

Type	Protocol	Port	Source
SSH	TCP	22	1.2.3.4/32



# SSM – Distributor

- Package and deploy software to your managed instances
- You create a **Distributor Package** (SSM Document) and deploy to different platforms (Windows, Linux)
- **Distributor Package**
  - Contents stored in S3
  - Zip file per target OS platform (install script, uninstall script, executable file)
  - JSON manifest file that describes the package content
- Use AWS-provided packages, 3rd party packages, or create your own package
- Install the package:
  - One-time – using Run Command
  - On a schedule – using State Manager (Document: [AWS-ConfigureAWSPackage](#))

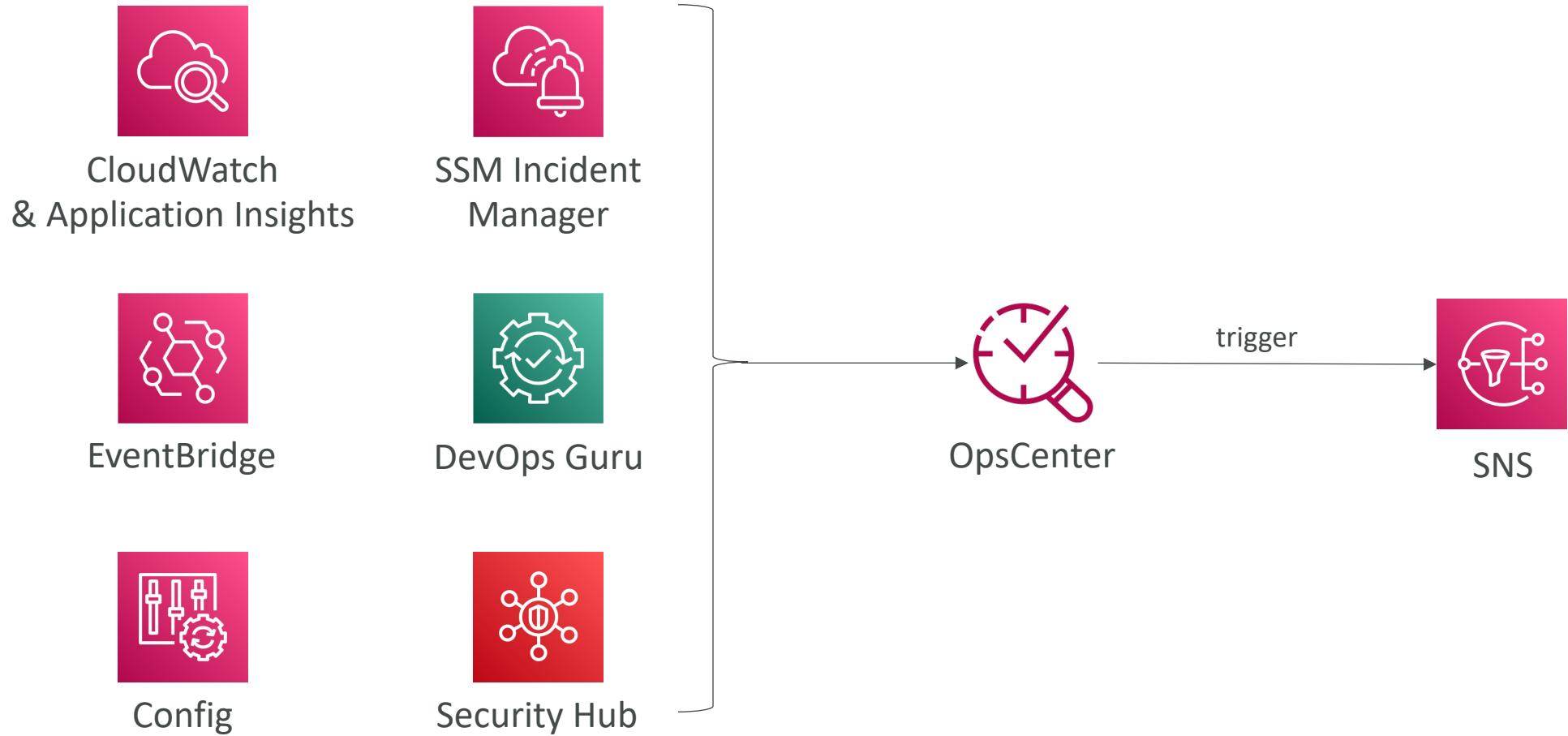




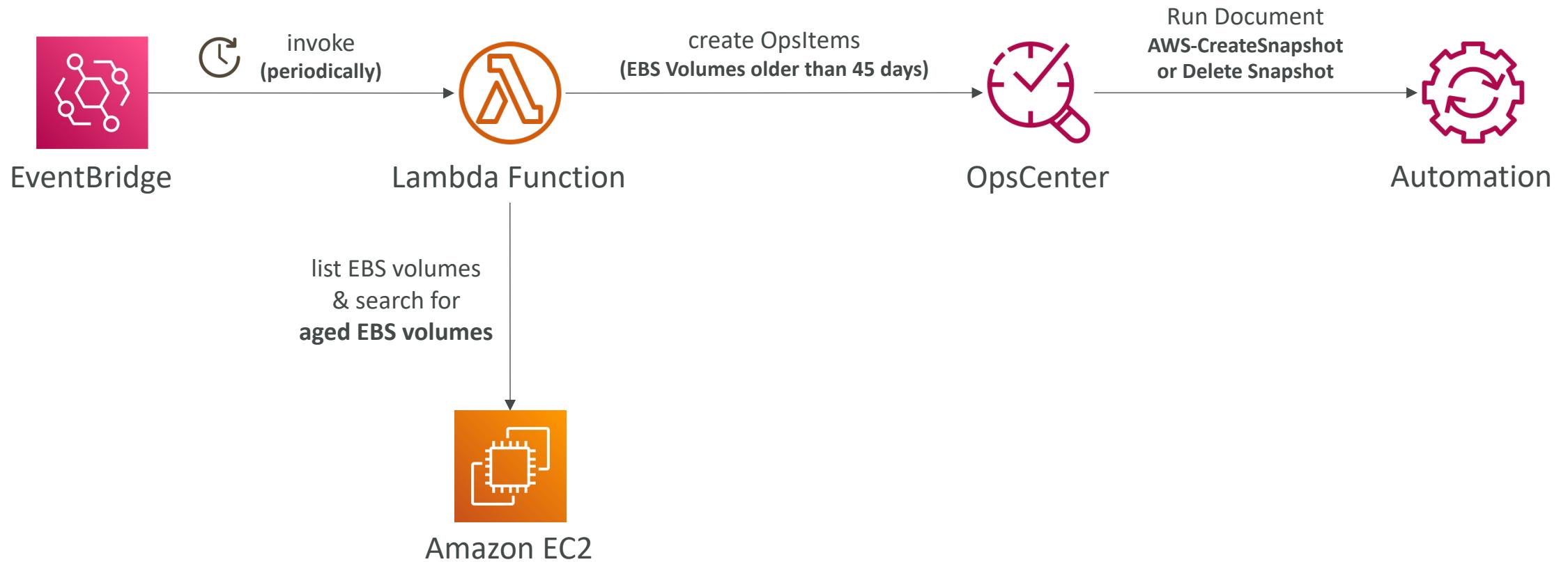
# Systems Manager – OpsCenter

- Allows you to view, investigate, and remediate issues in one place (no need to navigate across different AWS services)
- Security issues (Security Hub), performance issues (DynamoDB throttle), failures (ASG failed launch instance)...
- Reduce meantime to resolve issues
- **OpsItems**
  - Operational issue or interruption that needs investigation and remediation
  - Event, resource, AWS Config changes, CloudTrail logs, EventBridge...
  - Provides recommended Runbooks to resolve the issue
- Supports both EC2 instances and on-premises managed nodes

# Systems Manager – OpsCenter



# Systems Manager – OpsCenter – Reduce Costs by Deleting Orphaned EBS Volumes



# High Availability & Scalability

Load Balancer and Auto Scaling Groups

# Scalability and High Availability Section

- Load Balancers:
  - Troubleshooting
  - Advanced options and logging
  - CloudWatch integrations
- Auto Scaling
  - Troubleshooting
  - Advanced options and logging
  - CloudWatch integrations

# Scalability & High Availability

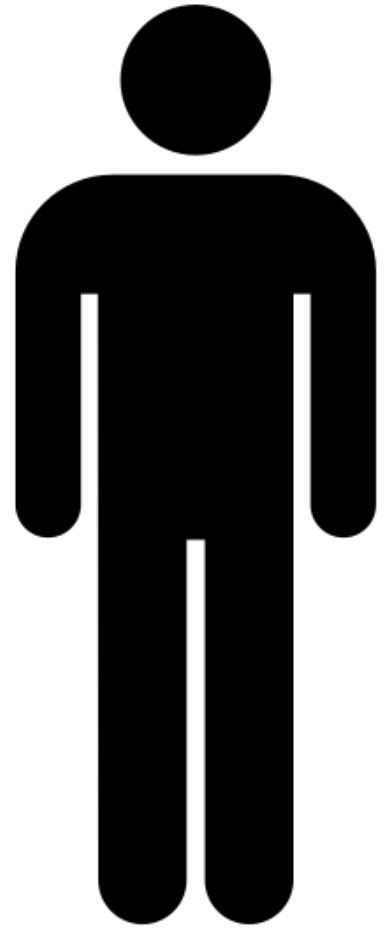
- Scalability means that an application / system can handle greater loads by adapting.
- There are two kinds of scalability:
  - Vertical Scalability
  - Horizontal Scalability (= elasticity)
- Scalability is linked but different to High Availability
- Let's deep dive into the distinction, using a call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

# Vertical Scalability

- Vertically 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.
- RDS, ElastiCache are services that can scale vertically.
- There's usually a limit to how much you can vertically scale (hardware limit)



junior operator

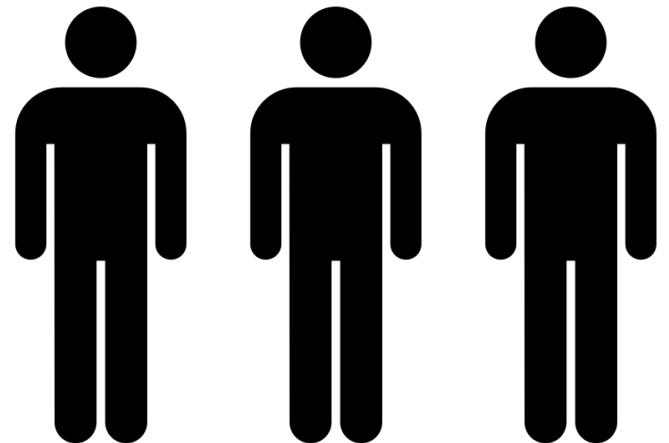
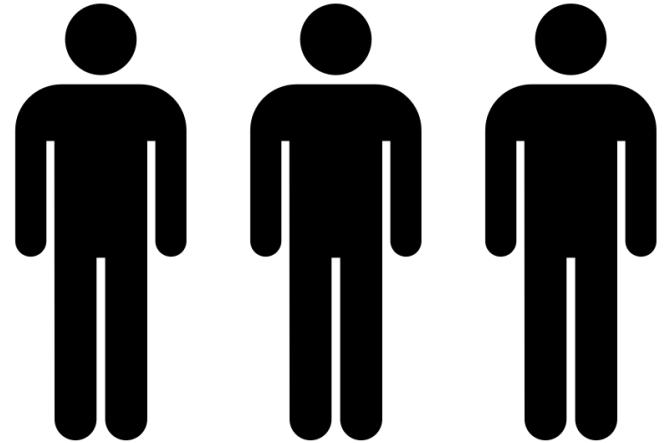


senior operator

# 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

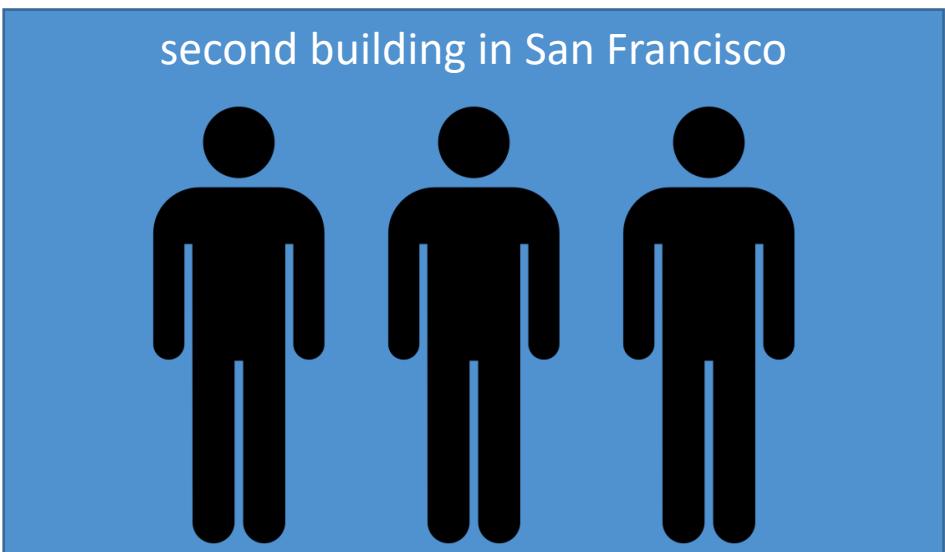
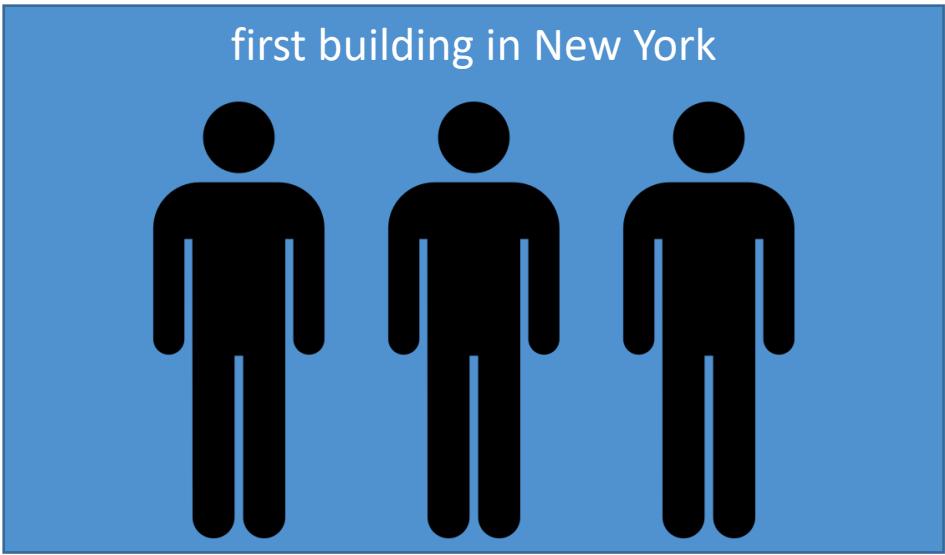
operator operator operator



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 data centers (== Availability Zones)
- The goal of high availability is to survive a data center loss
- The high availability can be passive (for RDS Multi AZ for example)
- The high availability can be active (for horizontal scaling)

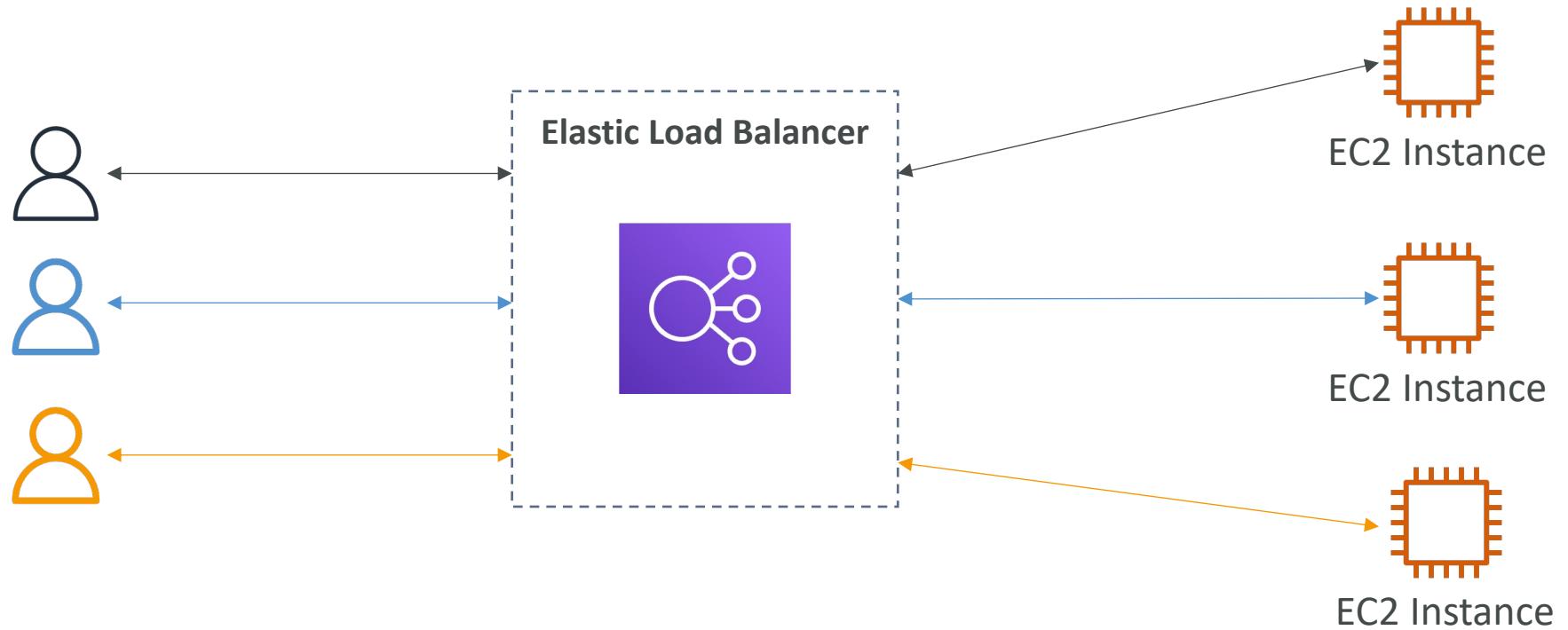


# High Availability & Scalability For EC2

- 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

# What is load balancing?

- Load Balancers are servers that forward traffic to multiple servers (e.g., EC2 instances) downstream



# Why use a load balancer?

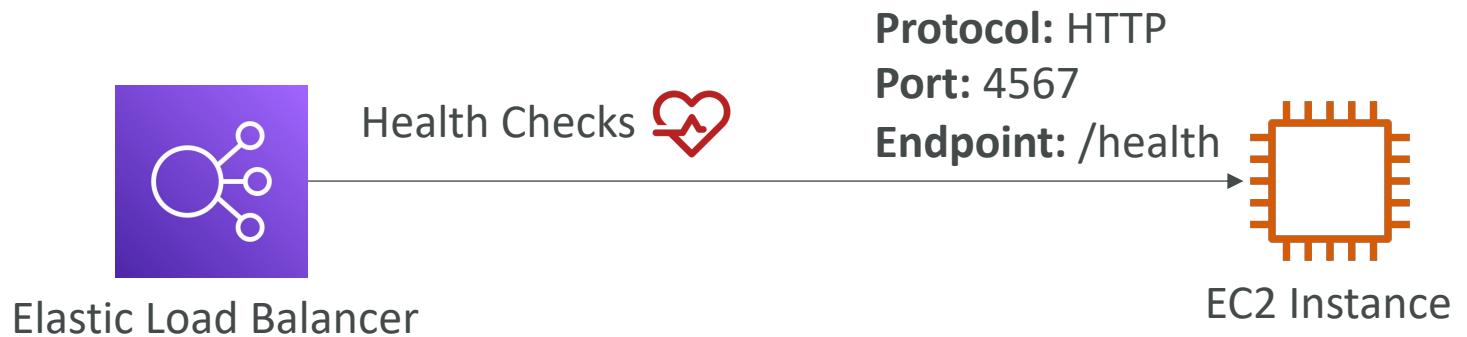
- Spread load across multiple downstream instances
- Expose a single point of access (DNS) to your application
- Seamlessly handle failures of downstream instances
- Do regular health checks to your instances
- Provide SSL termination (HTTPS) for your websites
- Enforce stickiness with cookies
- High availability across zones
- Separate public traffic from private traffic

# Why use an Elastic Load Balancer?

- An 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
- It is integrated with many AWS offerings / services
  - EC2, EC2 Auto Scaling Groups, Amazon ECS
  - AWS Certificate Manager (ACM), CloudWatch
  - Route 53, AWS WAF, AWS Global Accelerator

# Health Checks

- Health Checks are crucial for Load Balancers
- They enable the load balancer to know if instances it forwards traffic to are available to reply to requests
- The health check is done on a port and a route (/health is common)
- If the response is not 200 (OK), then the instance is unhealthy





# Types of load balancer on AWS

- AWS has **4 kinds of managed Load Balancers**
- **Classic Load Balancer** (v1 - old generation) – 2009 – CLB
  - HTTP, HTTPS, TCP, SSL (secure TCP)
- **Application Load Balancer** (v2 - new generation) – 2016 – ALB
  - HTTP, HTTPS, WebSocket
- **Network Load Balancer** (v2 - new generation) – 2017 – NLB
  - TCP, TLS (secure TCP), UDP
- **Gateway Load Balancer** – 2020 – GWLB
  - Operates at layer 3 (Network layer) – IP Protocol
- Overall, it is recommended to use the newer generation load balancers as they provide more features
- Some load balancers can be setup as **internal** (private) or **external** (public) ELBs

# Load Balancer Security Groups



## Load Balancer Security Group:

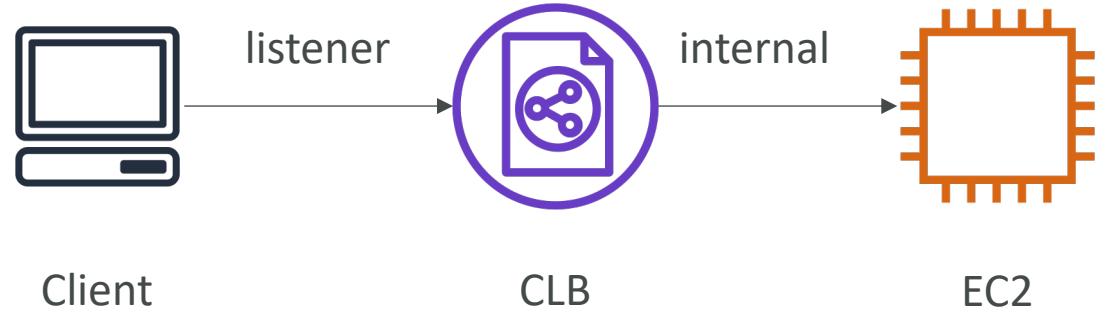
Type <small>i</small>	Protocol <small>i</small>	Port Range <small>i</small>	Source <small>i</small>	Description <small>i</small>
HTTP	TCP	80	0.0.0.0/0	Allow HTTP from an...
HTTPS	TCP	443	0.0.0.0/0	Allow HTTPS from a...

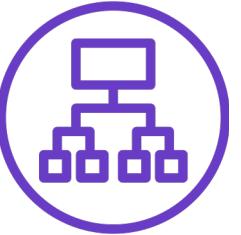
## Application Security Group: Allow traffic only from Load Balancer

Type <small>i</small>	Protocol <small>i</small>	Port Range <small>i</small>	Source <small>i</small>	Description <small>i</small>
HTTP	TCP	80	sg-054b5ff5ea02f2b6e (load-b	Allow Traffic only...

# Classic Load Balancers (v1)

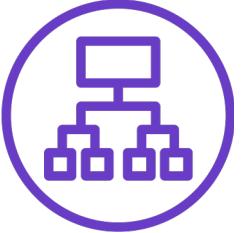
- Supports TCP (Layer 4), HTTP & HTTPS (Layer 7)
- Health checks are TCP or HTTP based
- Fixed hostname  
XXX.region.elb.amazonaws.com





# Application Load Balancer (v2)

- Application load balancers is Layer 7 (HTTP)
- Load balancing to multiple HTTP applications across machines (target groups)
- Load balancing to multiple applications on the same machine (ex: containers)
- Support for HTTP/2 and WebSocket
- Support redirects (from HTTP to HTTPS for example)

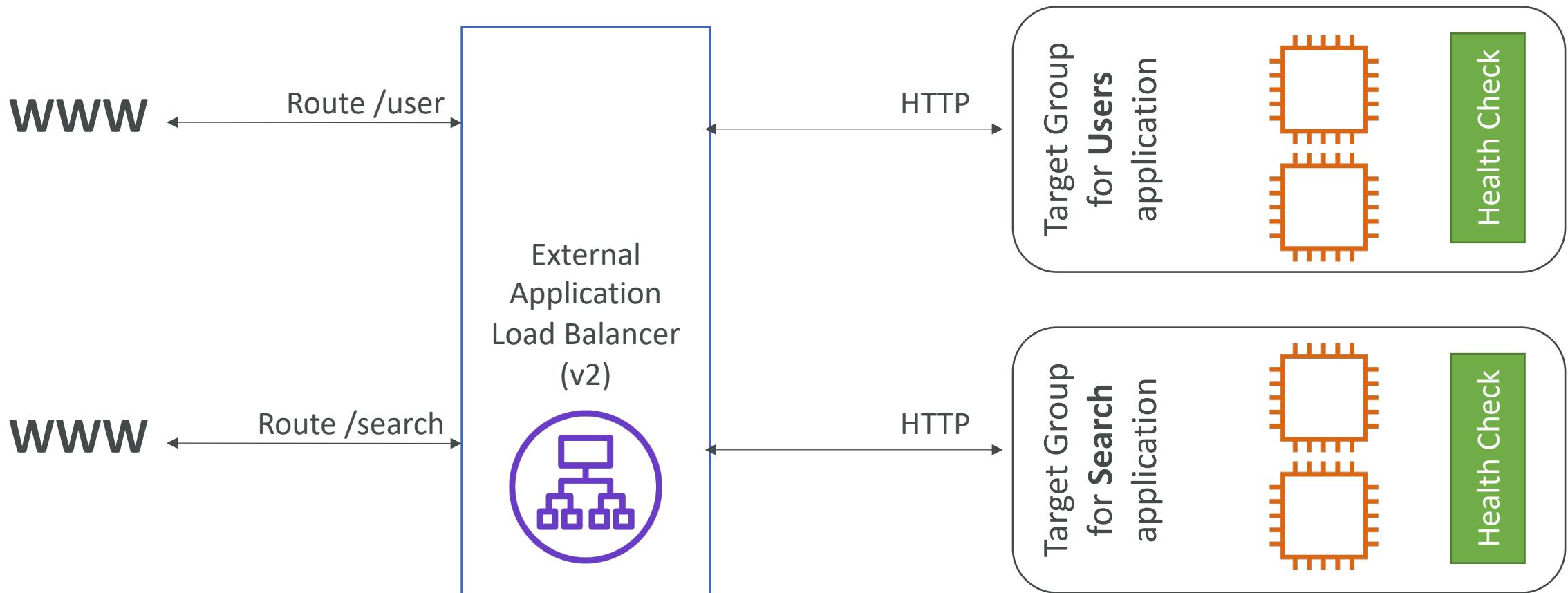


# Application Load Balancer (v2)

- Routing tables to different target groups:
  - Routing based on path in URL (example.com/**users** & example.com/**posts**)
  - Routing based on hostname in URL (**one.example.com** & **other.example.com**)
  - Routing based on Query String, Headers  
(example.com/users?id=123&order=false)
- ALB are a great fit for micro services & container-based application  
(example: Docker & Amazon ECS)
- Has a port mapping feature to redirect to a dynamic port in ECS
- In comparison, we'd need multiple Classic Load Balancer per application

# Application Load Balancer (v2)

## HTTP Based Traffic



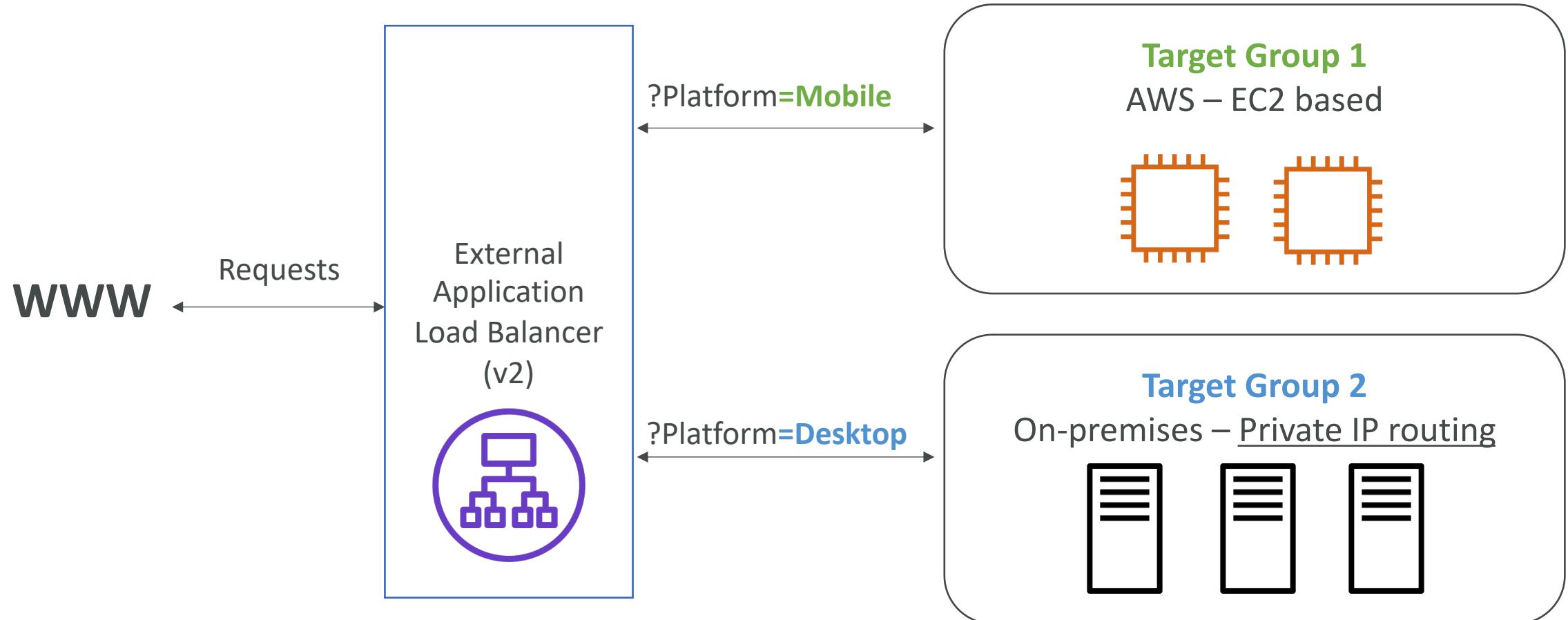
# Application Load Balancer (v2)

## Target Groups

- EC2 instances (can be managed by an Auto Scaling Group) – HTTP
  - ECS tasks (managed by ECS itself) – HTTP
  - Lambda functions – HTTP request is translated into a JSON event
  - IP Addresses – must be private IPs
- 
- ALB can route to multiple target groups
  - Health checks are at the target group level

# Application Load Balancer (v2)

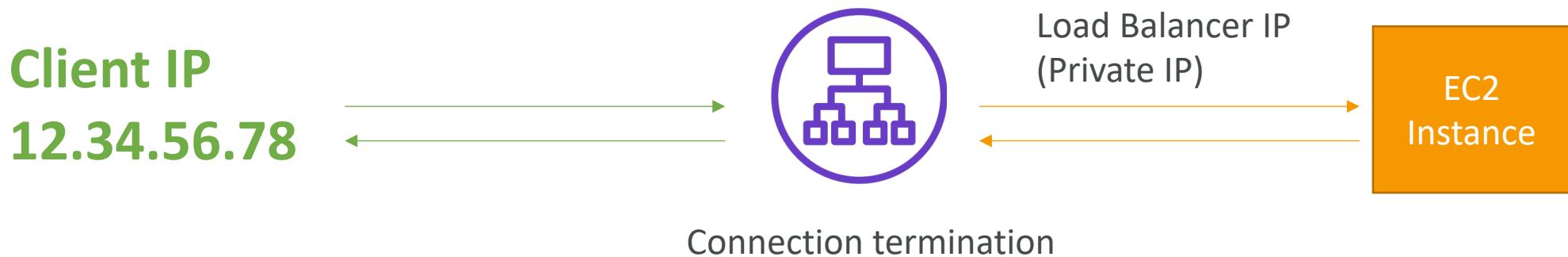
## Query Strings/Parameters Routing

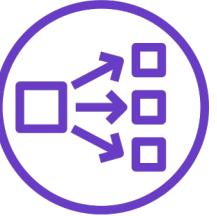


# Application Load Balancer (v2)

## Good to Know

- Fixed hostname (XXX.region.elb.amazonaws.com)
- The application servers don't see the IP of the client directly
  - The true IP of the client is inserted in the header X-Forwarded-For
  - We can also get Port (X-Forwarded-Port) and proto (X-Forwarded-Proto)



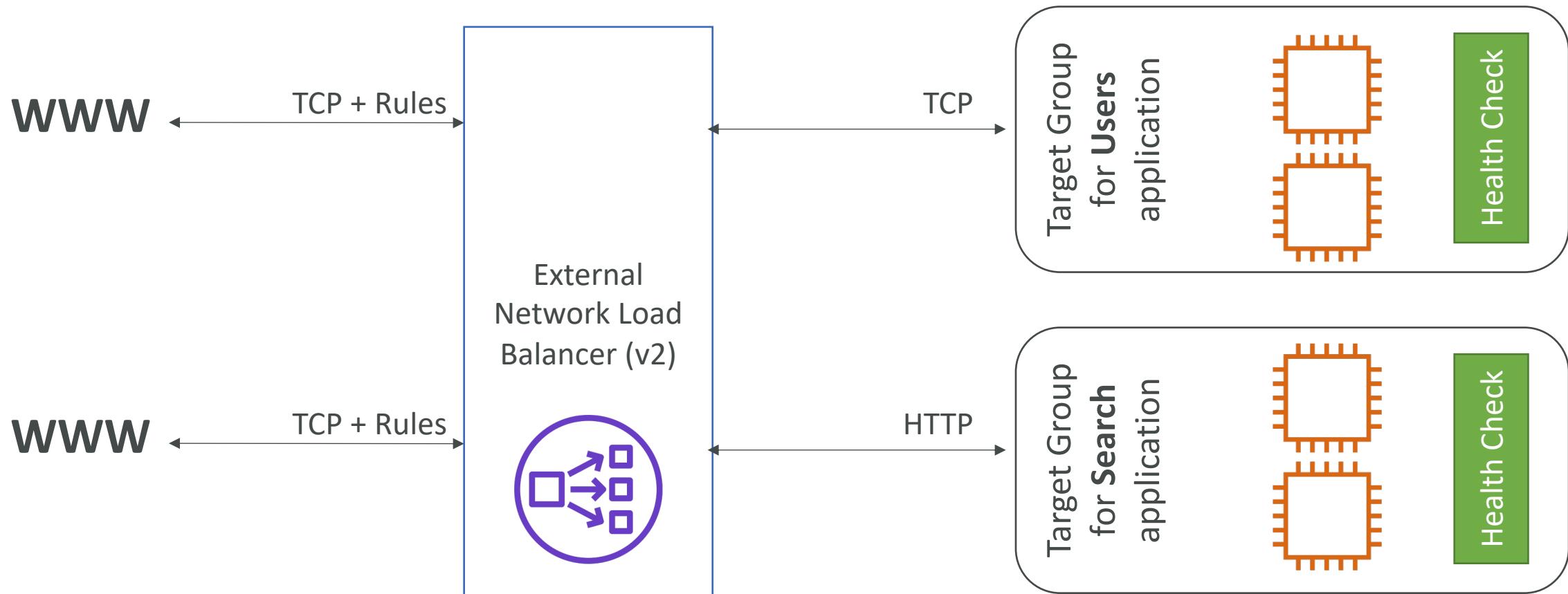


# Network Load Balancer (v2)

- Network load balancers (Layer 4) allow to:
  - Forward TCP & UDP traffic to your instances
  - Handle millions of requests per second
  - Ultra-low latency
- NLB has one static IP per AZ, and supports assigning Elastic IP (helpful for whitelisting specific IP)
- NLB are used for extreme performance, TCP or UDP traffic

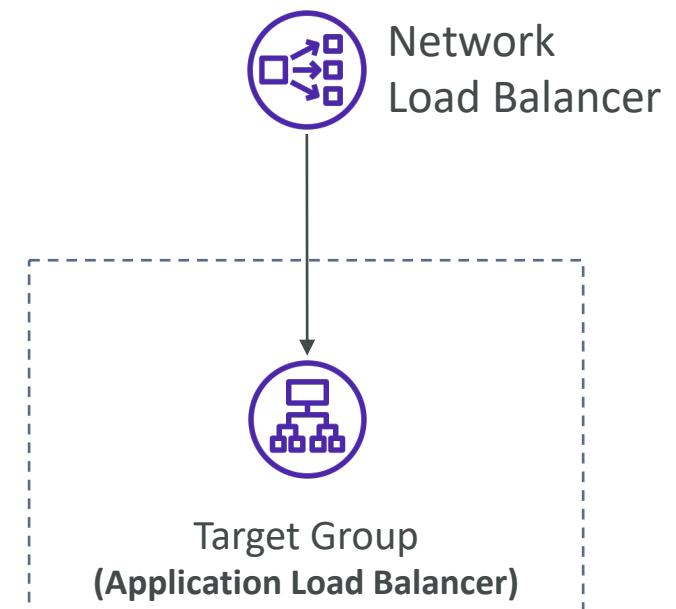
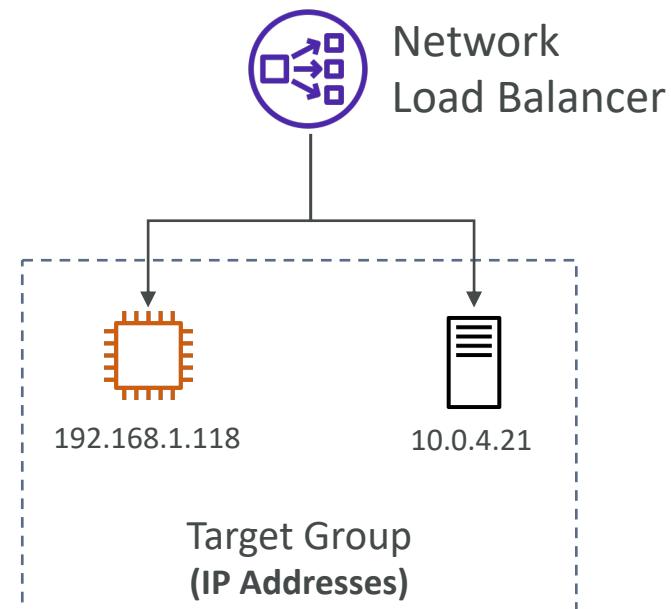
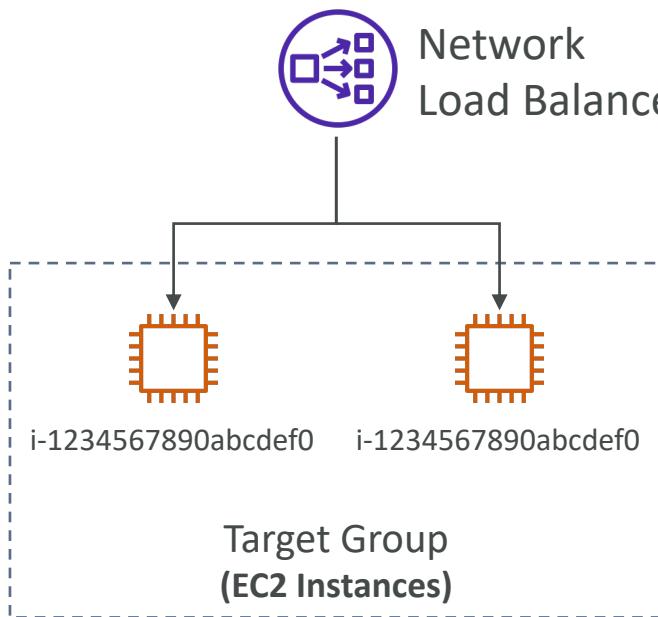
# Network Load Balancer (v2)

## TCP (Layer 4) Based Traffic



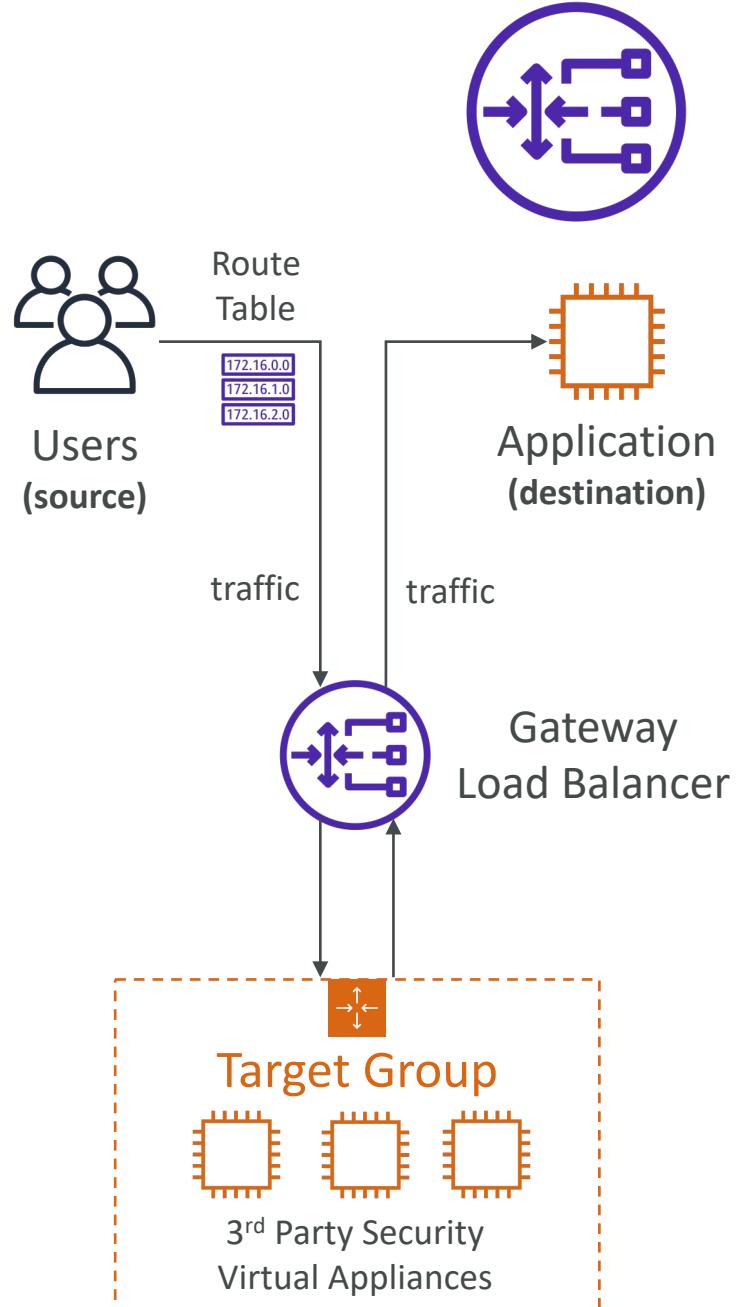
# Network Load Balancer – Target Groups

- EC2 instances
- IP Addresses – must be private IPs
- Application Load Balancer
- Health Checks support the TCP, HTTP and HTTPS Protocols



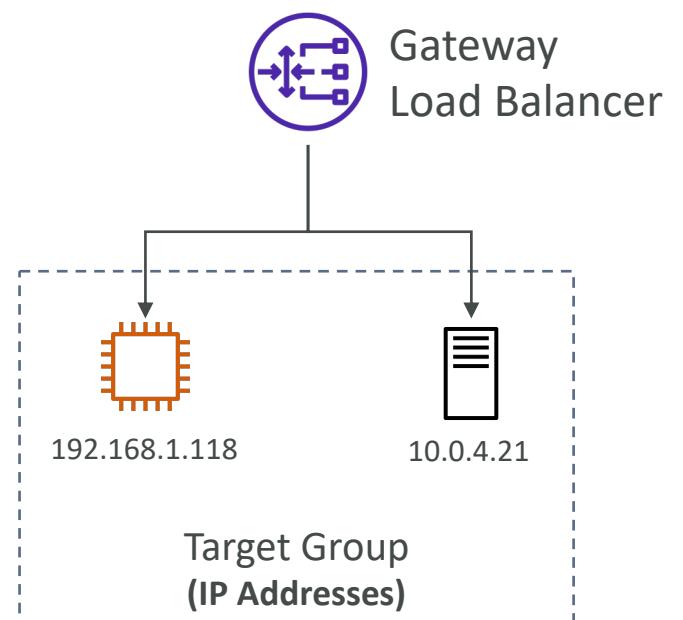
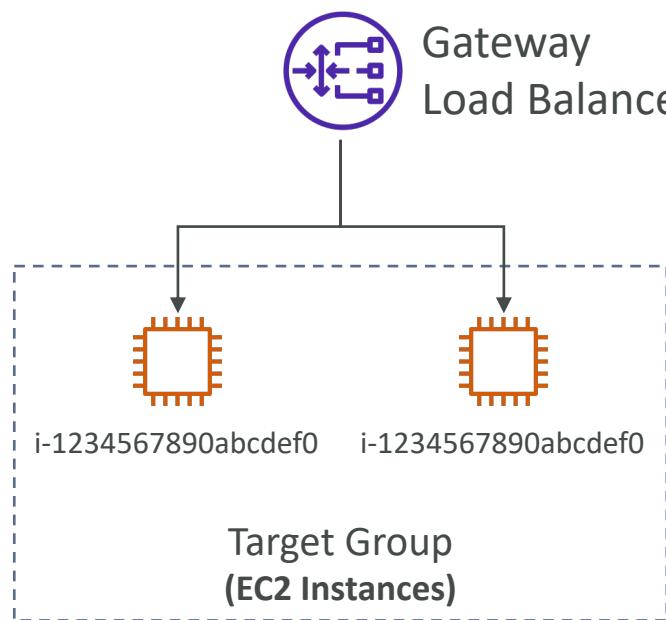
# Gateway Load Balancer

- Deploy, scale, and manage a fleet of 3<sup>rd</sup> party network virtual appliances in AWS
- Example: Firewalls, Intrusion Detection and Prevention Systems, Deep Packet Inspection Systems, payload manipulation, ...
- Operates at Layer 3 (Network Layer) – IP Packets
- Combines the following functions:
  - **Transparent Network Gateway** – single entry/exit for all traffic
  - **Load Balancer** – distributes traffic to your virtual appliances
- Uses the **GENEVE** protocol on port 6081



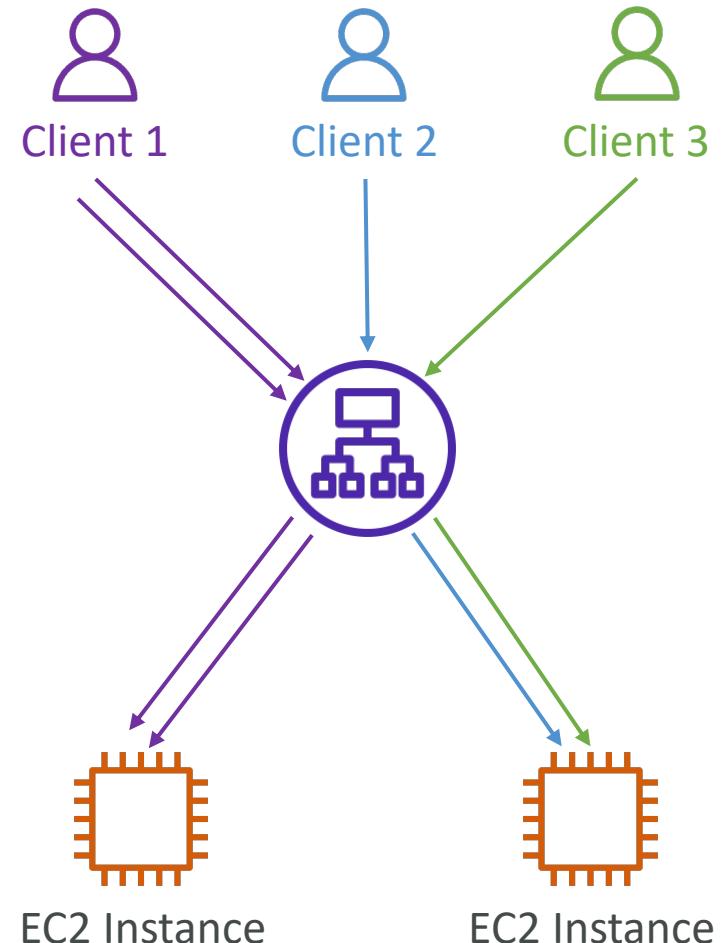
# Gateway Load Balancer – Target Groups

- EC2 instances
- IP Addresses – must be private IPs



# Sticky Sessions (Session Affinity)

- It is possible to implement stickiness so that the same client is always redirected to the same instance behind a load balancer
- This works for **Classic Load Balancer, Application Load Balancer, and Network Load Balancer**
- For both CLB & ALB, the “cookie” used for stickiness has an expiration date you control
- Use case: make sure the user doesn’t lose his session data
- Enabling stickiness may bring imbalance to the load over the backend EC2 instances



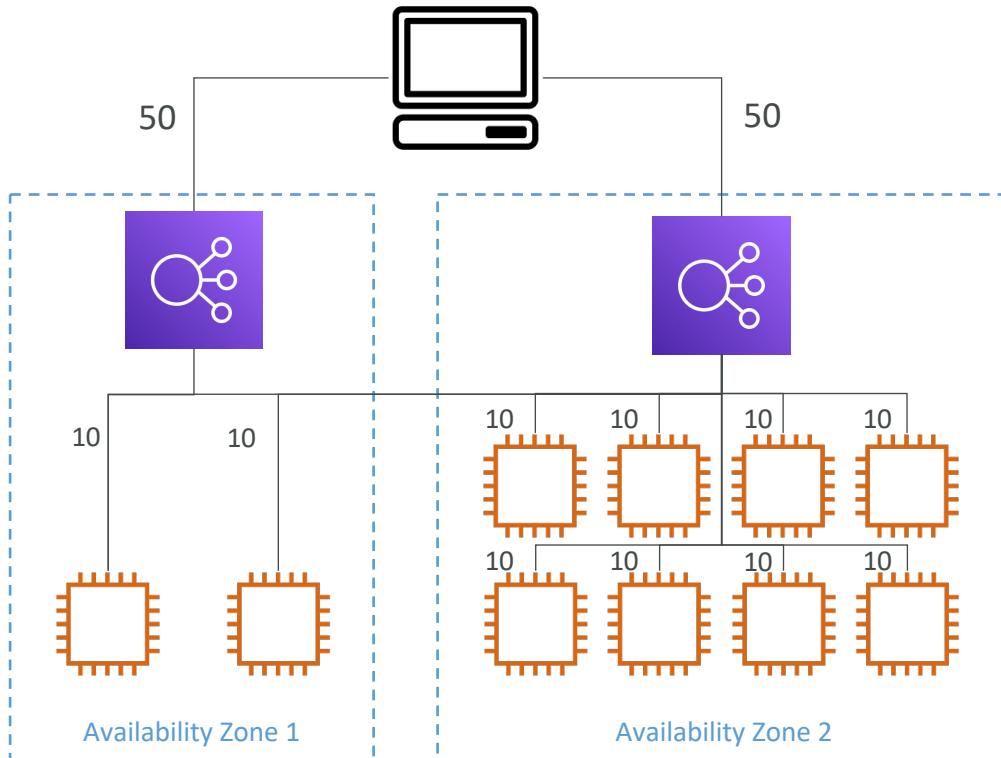
# Sticky Sessions – Cookie Names

- Application-based Cookies
  - Custom cookie
    - Generated by the target
    - Can include any custom attributes required by the application
    - Cookie name must be specified individually for each target group
    - Don't use **AWSALB**, **AWSALBAPP**, or **AWSALBTG** (reserved for use by the ELB)
  - Application cookie
    - Generated by the load balancer
    - Cookie name is **AWSALBAPP**
- Duration-based Cookies
  - Cookie generated by the load balancer
  - Cookie name is **AWSALB** for ALB, **AWSELB** for CLB

# Cross-Zone Load Balancing

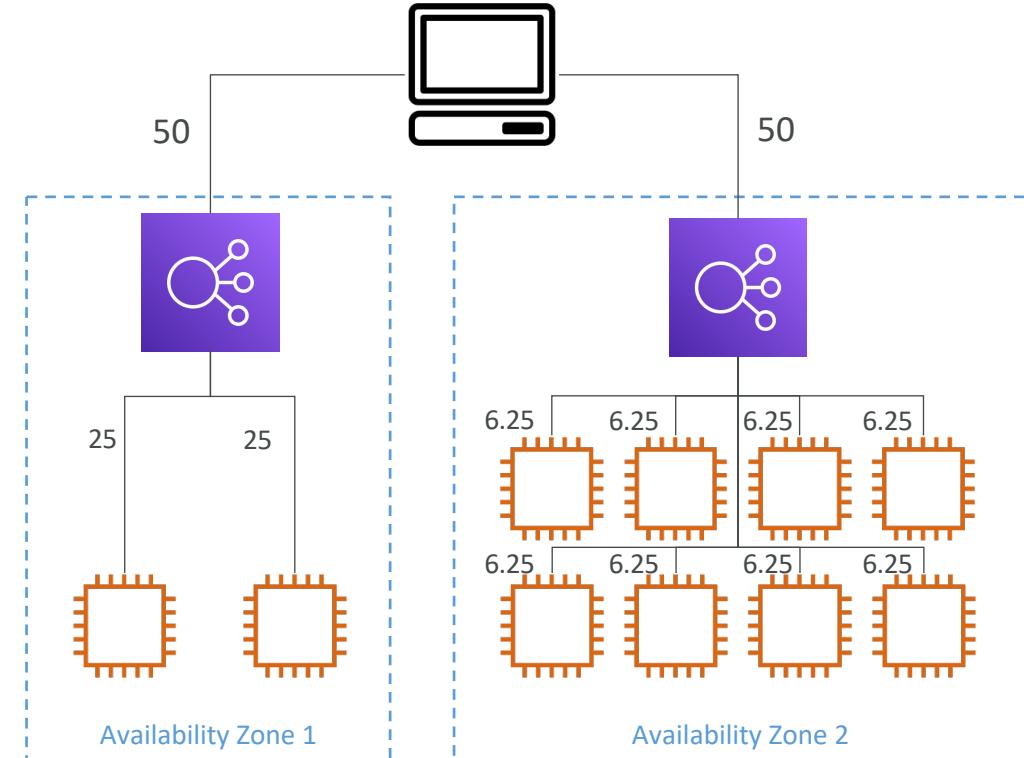
## With Cross Zone Load Balancing:

each load balancer instance distributes evenly across all registered instances in all AZ



## Without Cross Zone Load Balancing:

Requests are distributed in the instances of the node of the Elastic Load Balancer



# Cross-Zone Load Balancing

- Application Load Balancer
  - Enabled by default (can be disabled at the Target Group level)
  - No charges for inter AZ data
- Network Load Balancer & Gateway Load Balancer
  - Disabled by default
  - You pay charges (\$) for inter AZ data if enabled
- Classic Load Balancer
  - Disabled by default
  - No charges for inter AZ data if enabled

# SSL/TLS - Basics

- An SSL Certificate allows traffic between your clients and your load balancer to be encrypted in transit (in-flight encryption)
- SSL refers to Secure Sockets Layer, used to encrypt connections
- TLS refers to Transport Layer Security, which is a newer version
- Nowadays, **TLS certificates are mainly used**, but people still refer as SSL
- Public SSL certificates are issued by Certificate Authorities (CA)
- Comodo, Symantec, GoDaddy, GlobalSign, DigiCert, LetsEncrypt, etc...
- SSL certificates have an expiration date (you set) and must be renewed

# Load Balancer - SSL Certificates



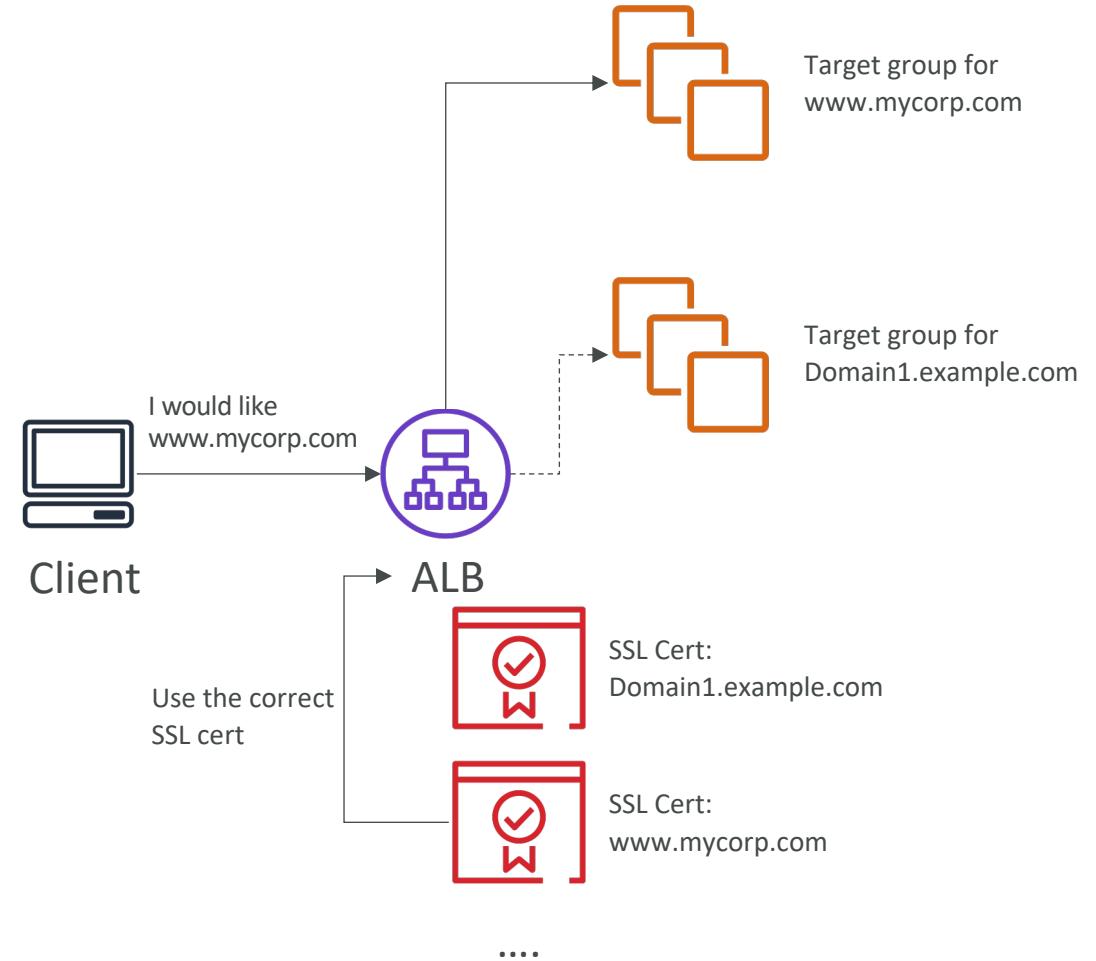
- The load balancer uses an X.509 certificate (SSL/TLS server certificate)
- You can manage certificates using ACM (AWS Certificate Manager)
- You can create/upload your own certificates alternatively
- HTTPS listener:
  - You must specify a default certificate
  - You can add an optional list of certs to support multiple domains
  - **Clients can use SNI (Server Name Indication) to specify the hostname they reach**
  - Ability to specify a security policy to support older versions of SSL / TLS (legacy clients)

# SSL – Server Name Indication (SNI)

- SNI solves the problem of loading **multiple SSL certificates onto one web server** (to serve multiple websites)
- It's a “newer” protocol, and requires the client to **indicate** the hostname of the target server in the initial SSL handshake
- The server will then find the correct certificate, or return the default one

## Note:

- Only works for ALB & NLB (newer generation), CloudFront
- Does not work for CLB (older gen)

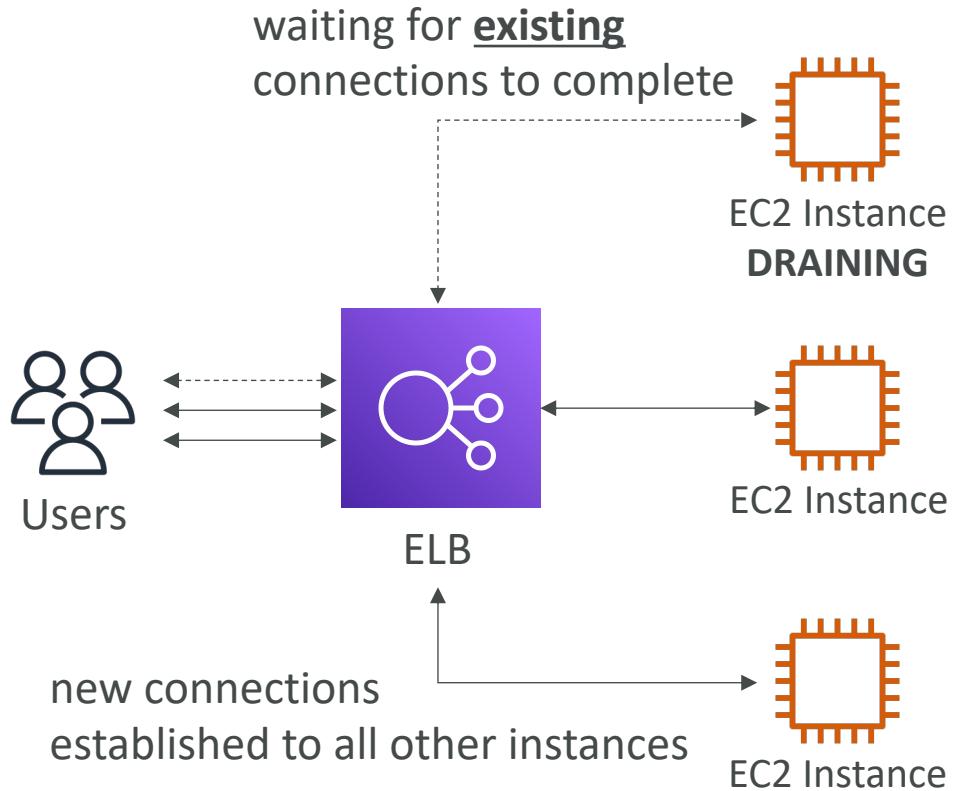


# Elastic Load Balancers – SSL Certificates

- **Classic Load Balancer (v1)**
  - Support only one SSL certificate
  - Must use multiple CLB for multiple hostname with multiple SSL certificates
- **Application Load Balancer (v2)**
  - Supports multiple listeners with multiple SSL certificates
  - Uses Server Name Indication (SNI) to make it work
- **Network Load Balancer (v2)**
  - Supports multiple listeners with multiple SSL certificates
  - Uses Server Name Indication (SNI) to make it work

# Connection Draining

- Feature naming
  - Connection Draining – for CLB
  - Deregistration Delay – for ALB & NLB
- Time to complete “in-flight requests” while the instance is de-registering or unhealthy
- Stops sending new requests to the EC2 instance which is de-registering
- Between 1 to 3600 seconds (default: 300 seconds)
- Can be disabled (set value to 0)
- Set to a low value if your requests are short



# ELB Health Checks

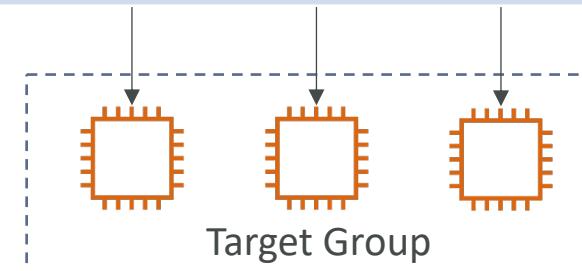
- Target Health Status
  - Initial: registering the target
  - Healthy
  - Unhealthy
  - Unused: target is not registered
  - Draining: de-registering the target
  - Unavailable: health checks disabled
- If a target group contains only unhealthy targets, ELB routes requests across its unhealthy targets

Application  
Load Balancer



Health Check

Setting	Value	Description
HealthCheckProtocol	HTTP	Protocol used to perform health checks
HealthCheckPort	80	Port used to perform health checks
HealthCheckPath	/	Destination for health checks on targets
HealthCheckTimeoutSeconds	5	Consider the health check failed if no response after 5 seconds
HealthCheckIntervalSeconds	30	Send health check every 30 seconds
HealthyThresholdCount	3	Consider the target healthy after 3 successful health checks
UnhealthyThresholdCount	5	Consider the target unhealthy after 5 failed health checks

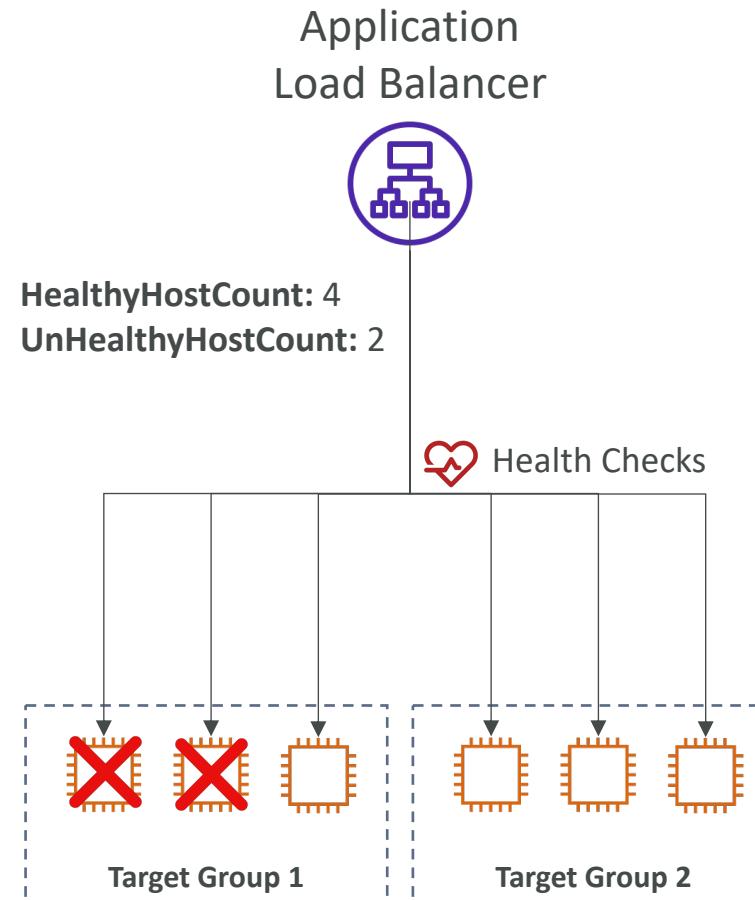


# Load Balancer Error codes

- Successful request : Code 200.
- Unsuccessful at client side : 4XX code.
  - Error 400 : Bad Request
  - Error 401 : Unauthorized
  - Error 403 : Forbidden
  - Error 460 : Client closed connection.
  - Error 463 : X-Forwarded For header with >30 IP (Similar to malformed request).
- Unsuccessful at server side : 5xx code.
  - An error 500 / Internal server error would mean some error on the ELB itself.
  - Error 502 : Bad Gateway
  - **An error 503 / Service Unavailable**
  - Error 504 / Gateway timeout : probably an issue within the server.
  - Error 561 : Unauthorized

# Load Balancers Monitoring

- All Load Balancer metrics are directly pushed to CloudWatch metrics
  - BackendConnectionErrors
  - HealthyHostCount / UnHealthyHostCount
  - HTTPCode\_Backend\_2XX: Successful request.
  - HTTPCode\_Backend\_3XX, redirected request
  - **HTTPCode\_ELB\_4XX**: Client error codes
  - **HTTPCode\_ELB\_5XX**: Server error codes generated by the load balancer.
- Latency
- RequestCount
- RequestCountPerTarget
- **SurgeQueueLength**: The total number of requests (HTTP listener) or connections (TCP listener) that are pending routing to a healthy instance. Help to scale out ASG. Max value is 1024
- **SpilloverCount**: The total number of requests that were rejected because the surge queue is full.



# Load Balancer troubleshooting using metrics

- **HTTP 400: BAD\_REQUEST =>** The client sent a malformed request that does not meet HTTP specifications.
- **HTTP 503: Service Unavailable =>** Ensure that you have healthy instances in every Availability Zone that your load balancer is configured to respond in. Look for HealthyHostCount in CloudWatch
- **HTTP 504: Gateway Timeout =>** Check if keep-alive settings on your EC2 instances are enabled and make sure that the keep-alive timeout is greater than the idle timeout settings of load balancer.
- Set alarms & look at the documentation for troubleshooting:  
<https://docs.aws.amazon.com/elasticloadbalancing/latest/classic/ts-elb-error-message.html>

# Load Balancers Access Logs

- Access logs from Load Balancers can be stored in S3 and contain:
  - Time
  - Client IP address
  - Latencies
  - Request paths
  - Server response
  - Trace Id
- Only pay for the S3 storage
- Helpful for compliance reason
- Helpful for keeping access data even after ELB or EC2 instances are terminated
- Access Logs are already encrypted

# Application Load Balancer Request Tracing

- Request tracing – Each HTTP request has an added custom header ‘X-Amzn-Trace-Id’

- Example:

X-Amzn-Trace-Id: Root=1-67891233-abcdef012345678912345678

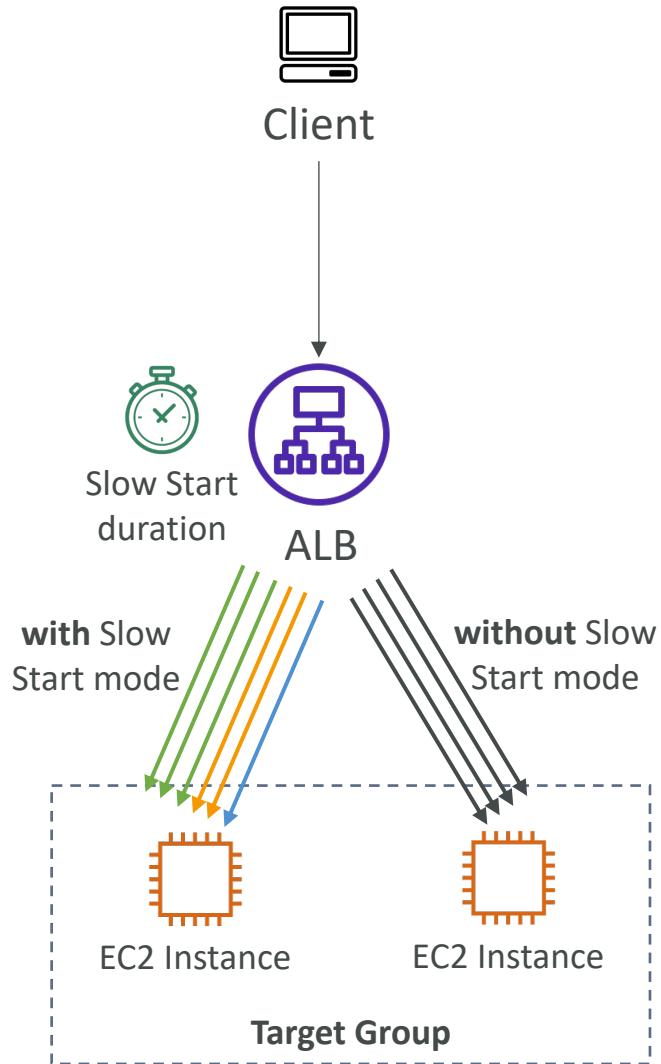
- This is useful in logs / distributed tracing platform to track a single request
- Application Load Balancer is not (yet) integrated with X-Ray

# Target Groups Settings

- `deregistration_delay.timeout_seconds`: time the load balancer waits before deregistering a target
- `slow_start.duration_seconds`: (see next slide)
- `load_balancing.algorithm.type`: how the load balancer selects targets when routing requests (Round Robin, Least Outstanding Requests)
- `stickiness.enabled`
- `stickiness.type`: application-based or duration-based cookie
- `stickiness.app_cookie.cookie_name`: name of the application cookie
- `stickiness.app_cookie.duration_seconds`: application-based cookie expiration period
- `stickiness.lb_cookie.duration_seconds`: duration-based cookie expiration period

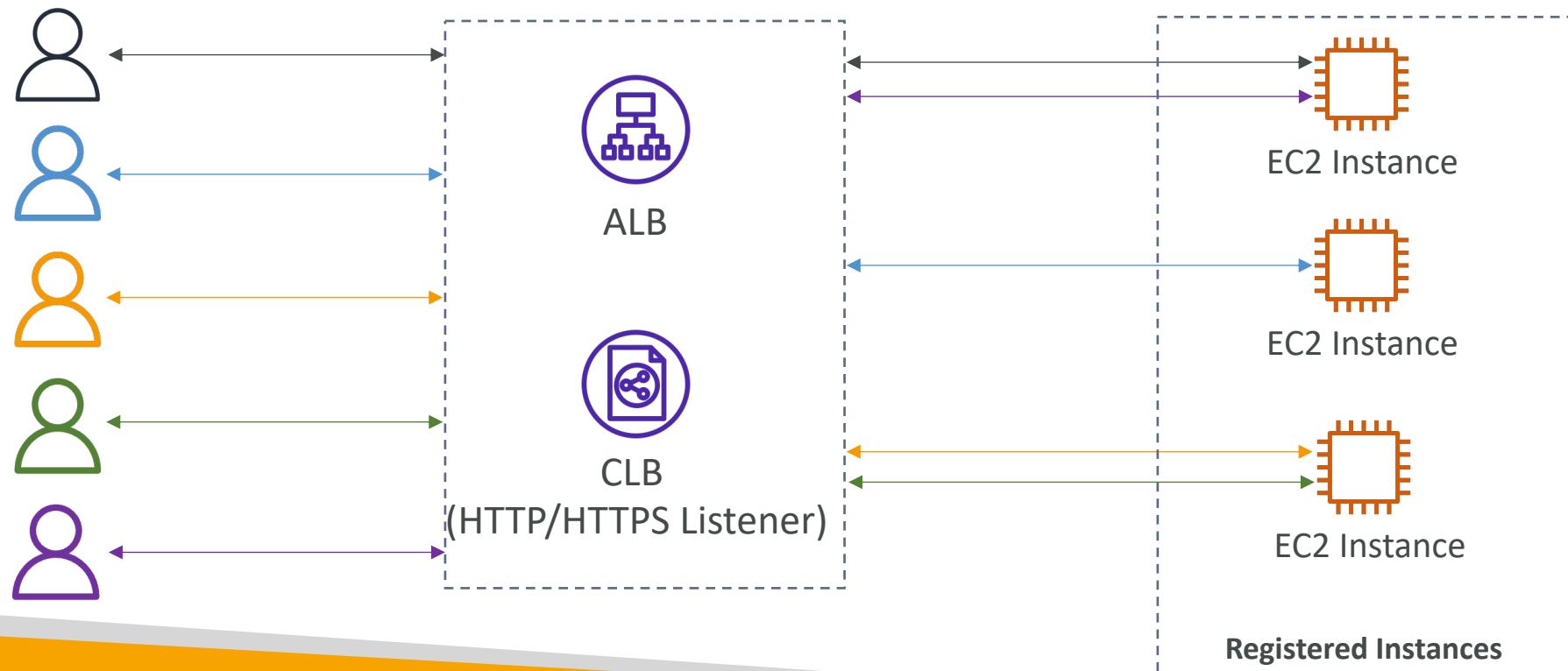
# Slow Start Mode

- By default, a target receives its full share of requests once it's registered with the target group
- **Slow Start Mode** gives healthy targets time to warm-up before the load balancer sends them a full share of requests
- The load balancer linearly increases the number of requests that it sends to the target
- A target exits Slow Start Mode when:
  - The duration period elapses
  - The target becomes unhealthy
- To disable, set **Slow start duration** value to 0



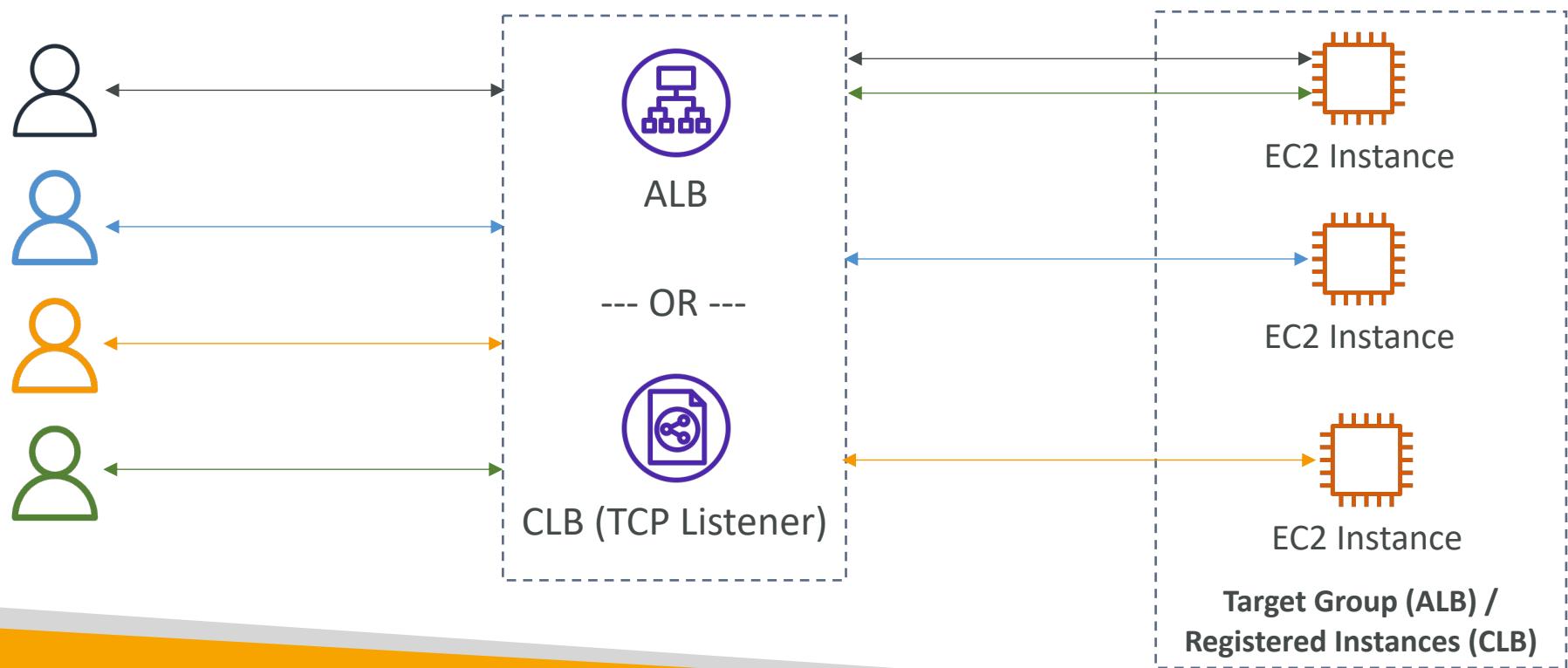
# Request Routing Algorithms – Least Outstanding Requests

- The next instance to receive the request is the instance that has the lowest number of pending/unfinished requests
- Works with Application Load Balancer and Classic Load Balancer (HTTP/HTTPS)



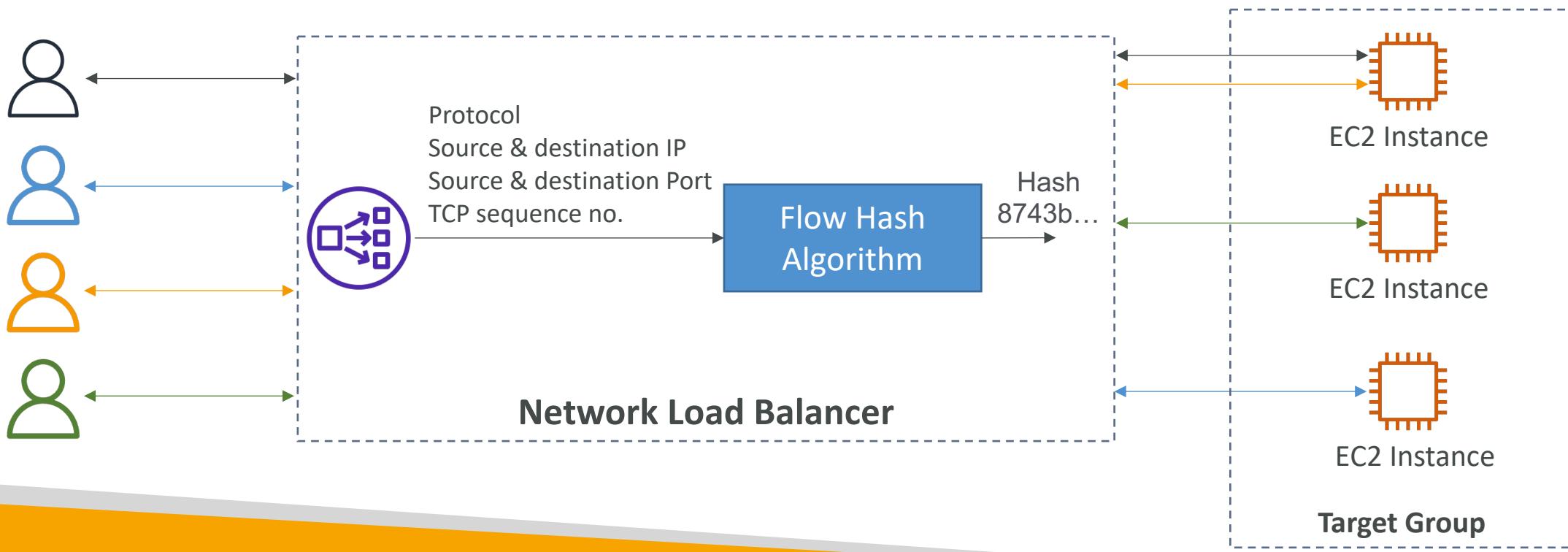
# Request Routing Algorithms – Round Robin

- Equally choose the targets from the target group
- Works with Application Load Balancer and Classic Load Balancer (TCP)



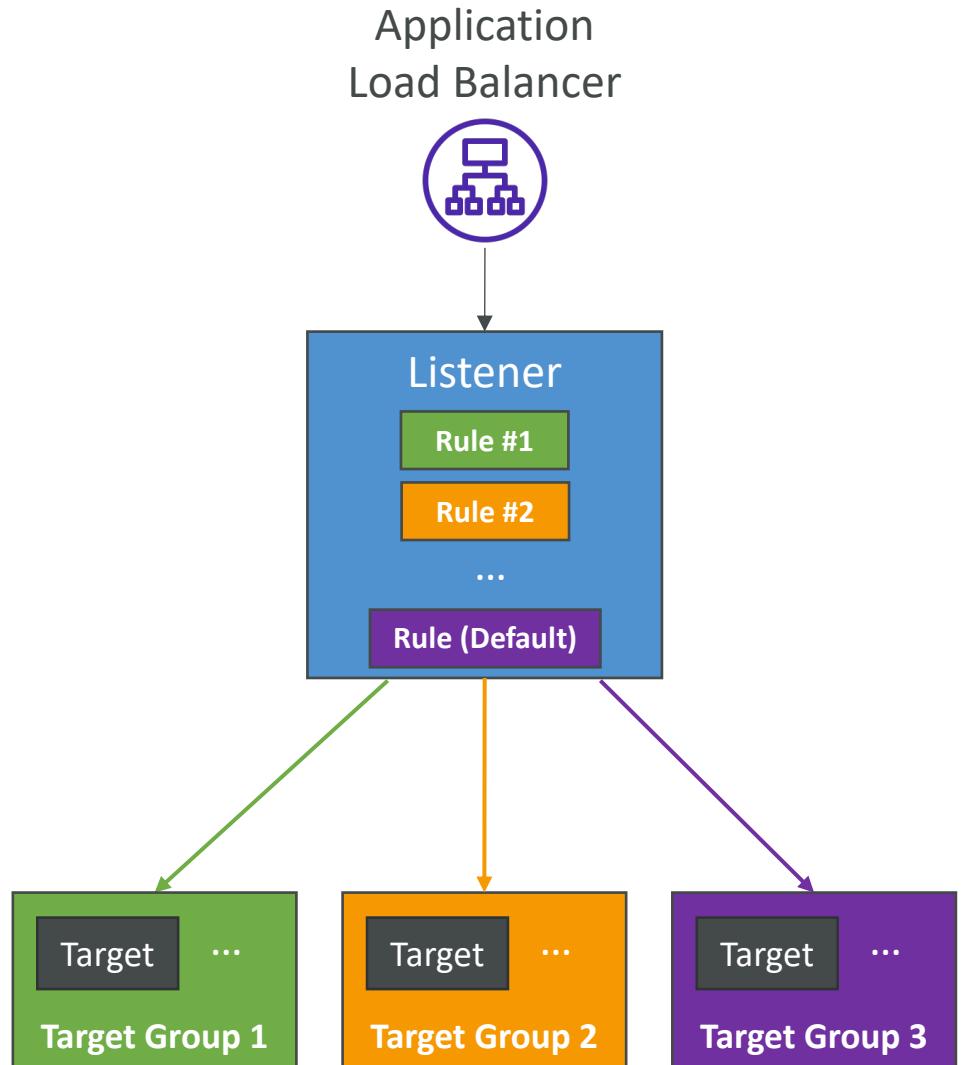
# Request Routing Algorithms – Flow Hash

- Selects a target based on the protocol, source/destination IP address, source/destination port, and TCP sequence number
- Each TCP/UDP connection is routed to a single target for the life of the connection
- Works with Network Load Balancer



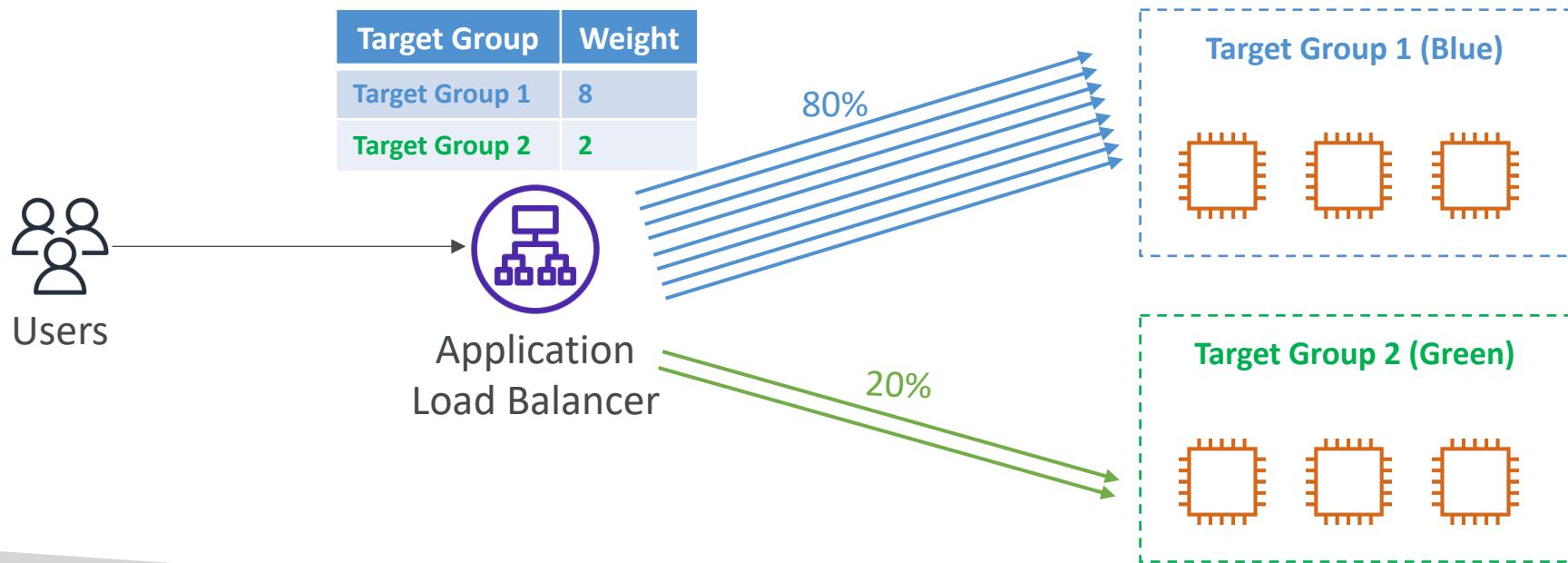
# ALB – Listener Rules

- Processed in order (with Default Rule)
- Supported Actions (forward, redirect, fixed-response)
- Rule Conditions:
  - host-header
  - http-request-method
  - path-pattern
  - source-ip
  - http-header
  - query-string

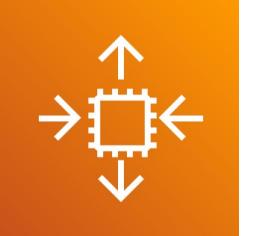


# Target Group Weighting

- Specify weight for each Target Group on a single Rule
- Example: multiple versions of your app, blue/green deployment
- Allows you to control the distribution of the traffic to your applications

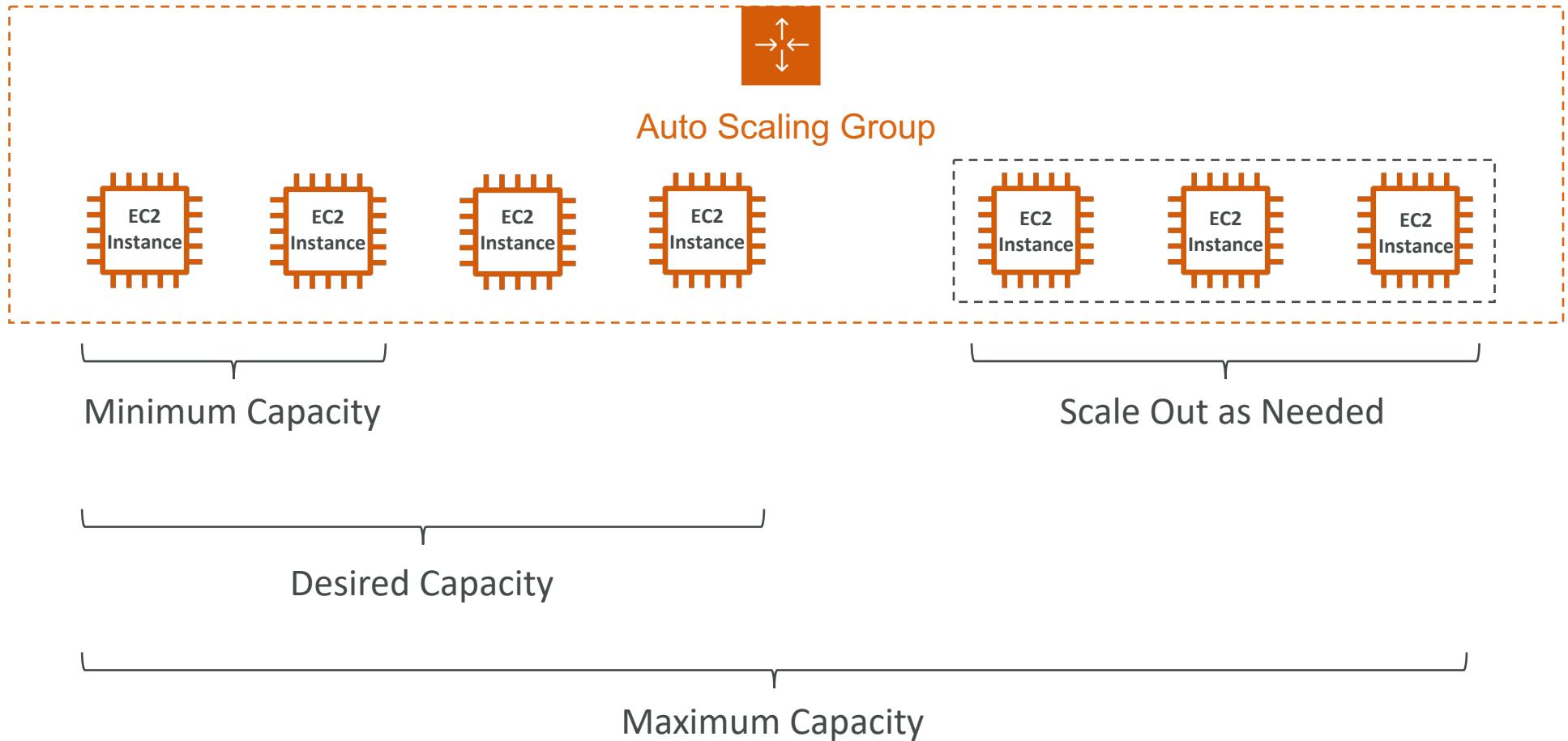


# What's an Auto Scaling Group?

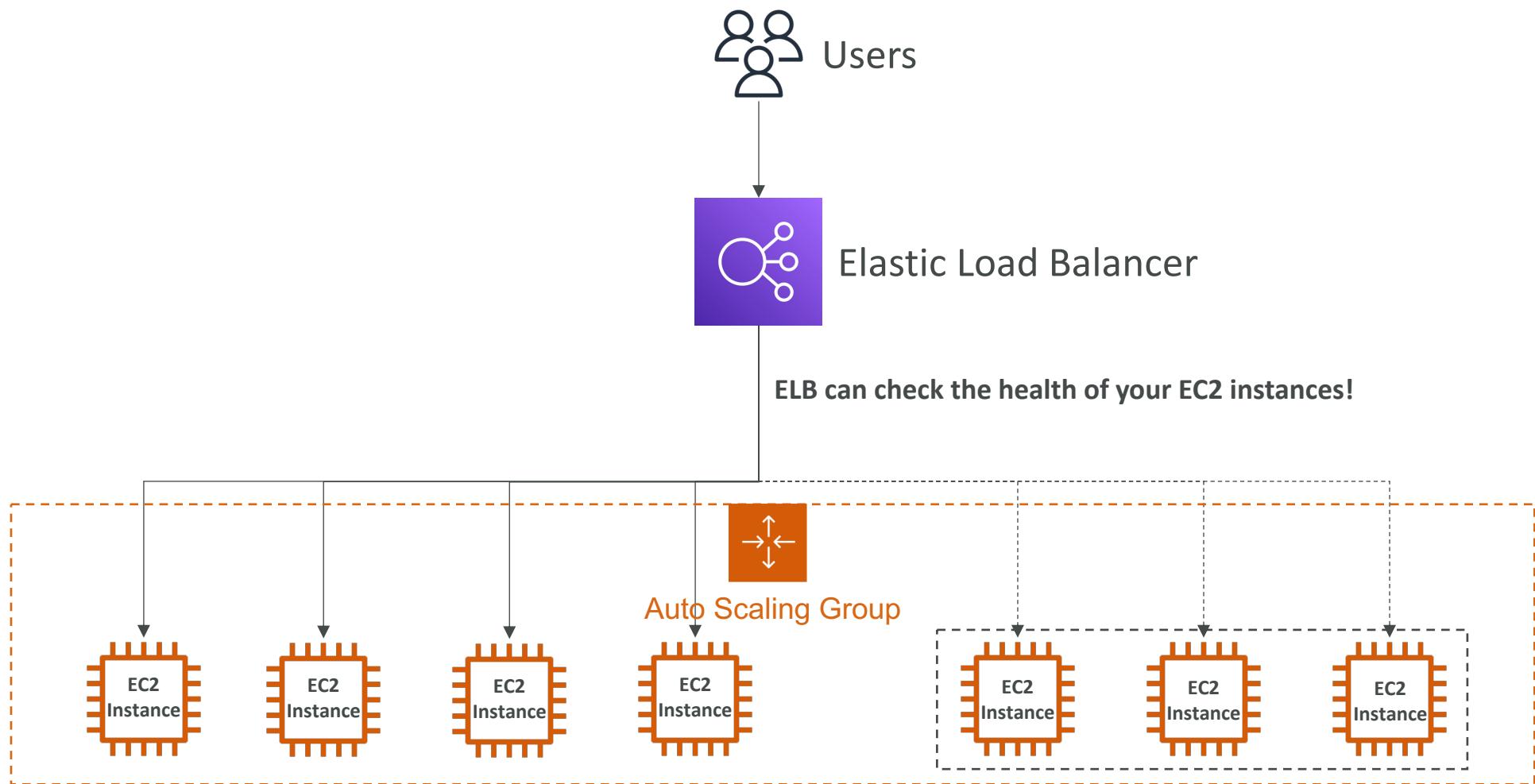


- In real-life, the load on your websites and application can change
- In the cloud, you can create and get rid of servers very quickly
- The goal of an Auto Scaling Group (ASG) is to:
  - Scale out (add EC2 instances) to match an increased load
  - Scale in (remove EC2 instances) to match a decreased load
  - Ensure we have a minimum and a maximum number of EC2 instances running
  - Automatically register new instances to a load balancer
  - Re-create an EC2 instance in case a previous one is terminated (ex: if unhealthy)
- ASG are free (you only pay for the underlying EC2 instances)

# Auto Scaling Group in AWS

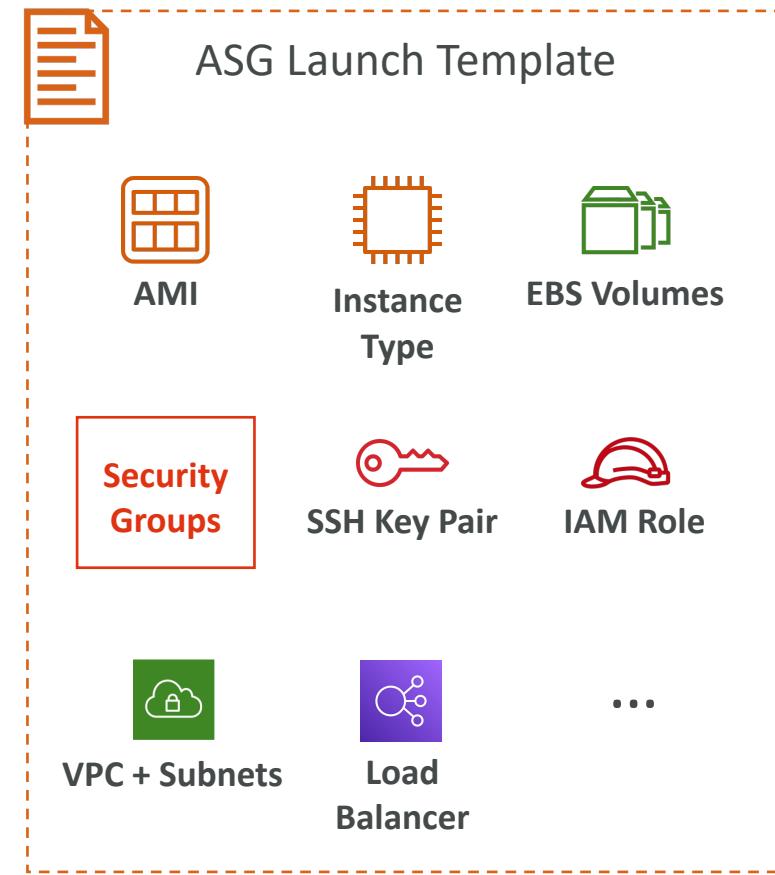


# Auto Scaling Group in AWS With Load Balancer



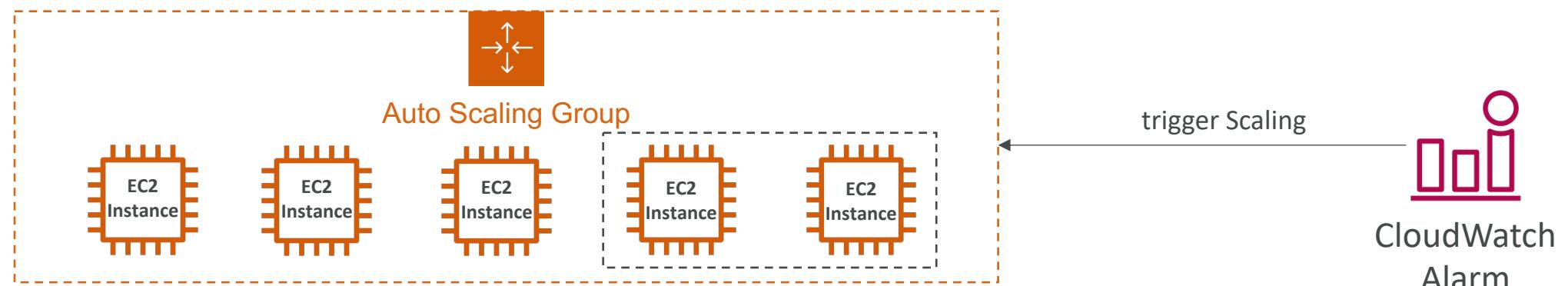
# Auto Scaling Group Attributes

- A **Launch Template** (older “Launch Configurations” are deprecated)
  - AMI + Instance Type
  - EC2 User Data
  - EBS Volumes
  - Security Groups
  - SSH Key Pair
  - IAM Roles for your EC2 Instances
  - Network + Subnets Information
  - Load Balancer Information
- Min Size / Max Size / Initial Capacity
- Scaling Policies



# Auto Scaling - CloudWatch Alarms & Scaling

- It is possible to scale an ASG based on CloudWatch alarms
- An alarm monitors a metric (such as **Average CPU**, or a **custom metric**)
- Metrics such as Average CPU are computed for the overall ASG instances
- Based on the alarm:
  - We can create scale-out policies (increase the number of instances)
  - We can create scale-in policies (decrease the number of instances)

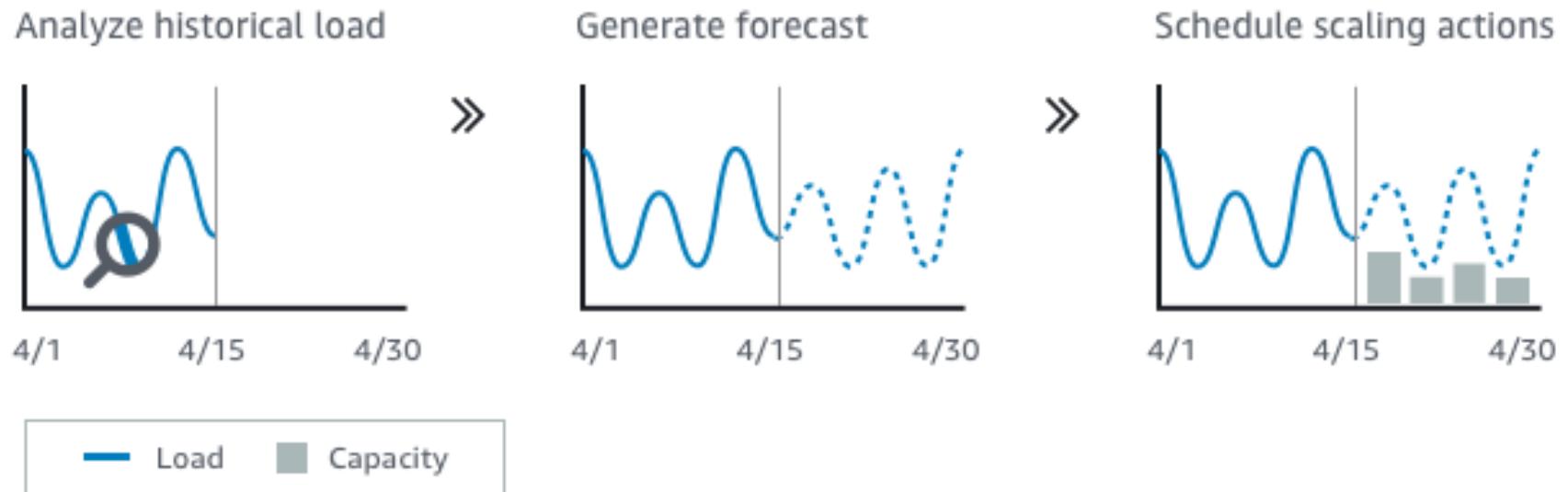


# Auto Scaling Groups – Scaling Policies

- Dynamic Scaling
  - Target Tracking Scaling
    - Simple to set-up
    - Example: I want the average ASG CPU to stay at around 40%
  - 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
- Scheduled Scaling
  - Anticipate a scaling based on known usage patterns
  - Example: increase the min capacity to 10 at 5 pm on Fridays

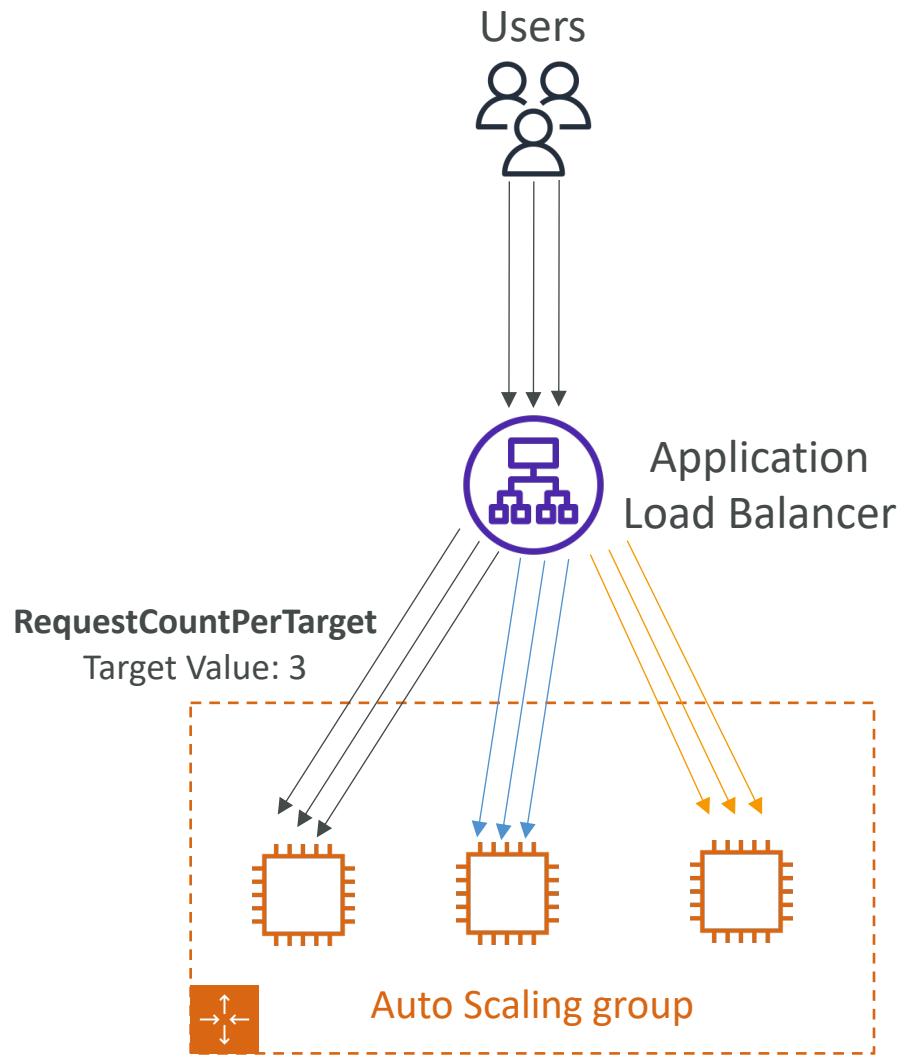
# Auto Scaling Groups – Scaling Policies

- Predictive scaling: continuously forecast load and schedule scaling ahead



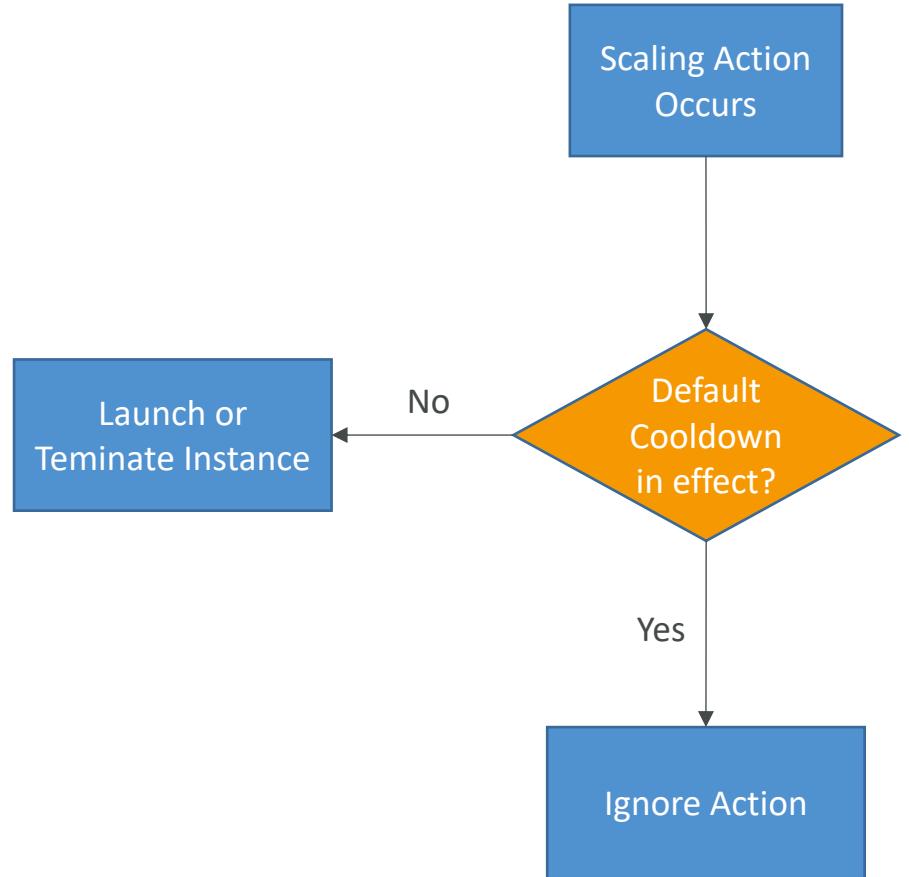
# Good metrics to scale on

- **CPUUtilization:** Average CPU utilization across your instances
- **RequestCountPerTarget:** to make sure the number of requests per EC2 instances is stable
- **Average Network In / Out** (if your application is network bound)
- Any custom metric (that you push using CloudWatch)



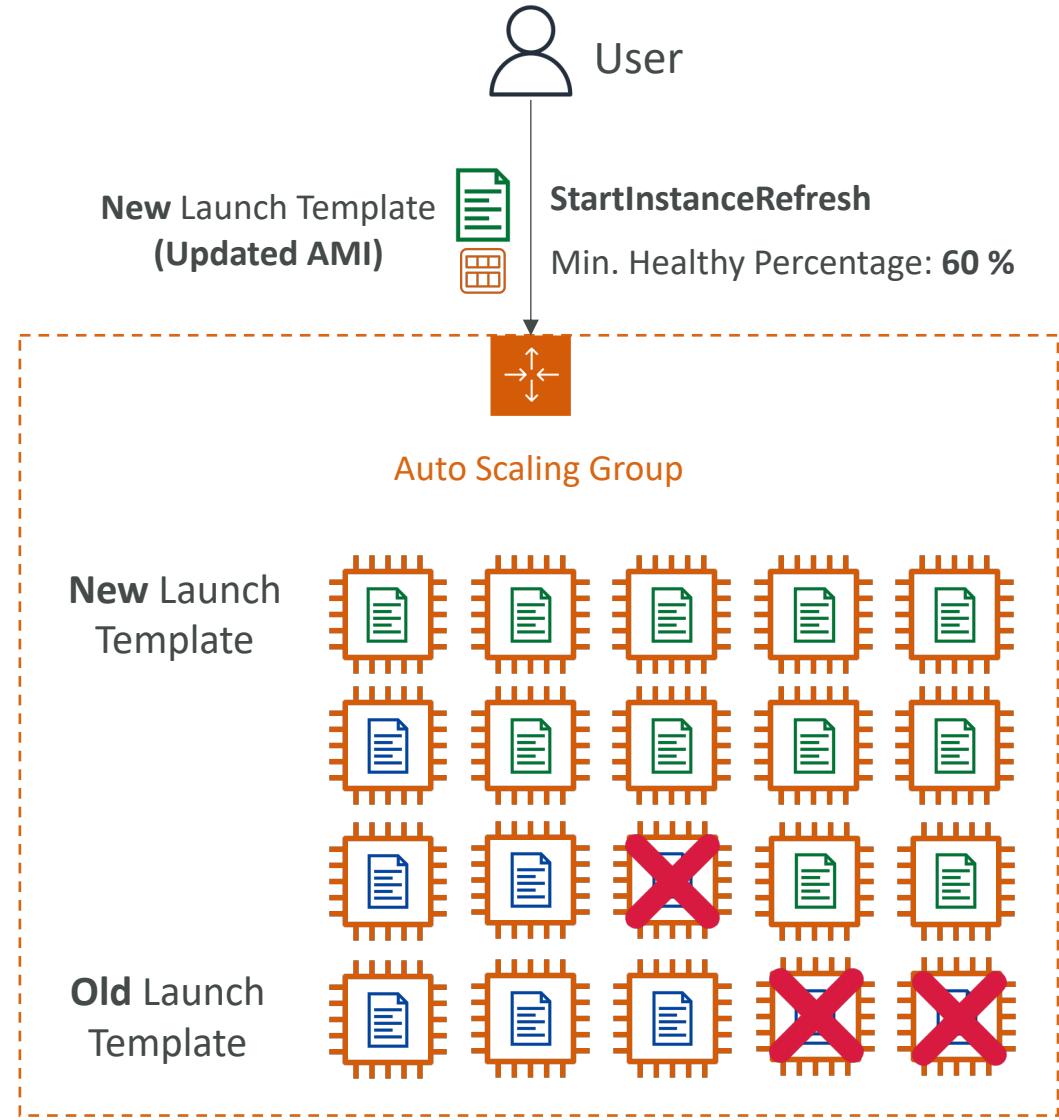
# Auto Scaling Groups - Scaling Cooldowns

- After a scaling activity happens, you are in the cooldown period (default 300 seconds)
- During the cooldown period, the ASG will not launch or terminate additional instances (to allow for metrics to stabilize)
- Advice: Use a ready-to-use AMI to reduce configuration time in order to be serving request faster and reduce the cooldown period



# Auto Scaling – Instance Refresh

- Goal: update launch template and then re-creating all EC2 instances
- For this we can use the native feature of Instance Refresh
- Setting of minimum healthy percentage
- Specify warm-up time (how long until the instance is ready to use)

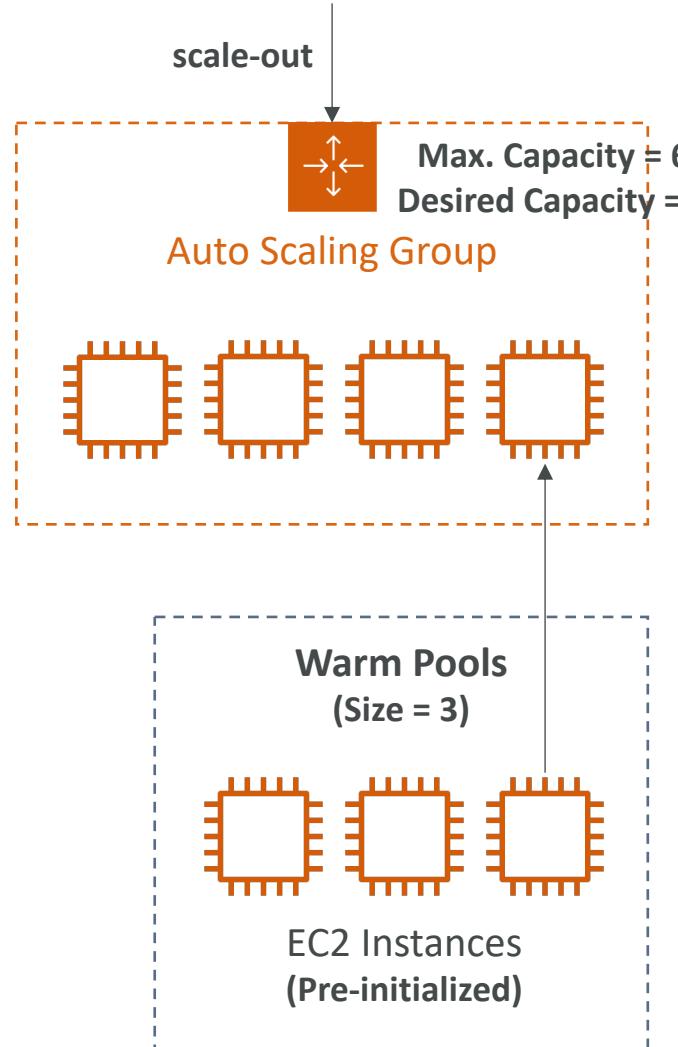


# ASG – Scale-out Latency Problem

- When an ASG scales out, it tries to launch instances as fast as possible
- Some applications contain a lengthy unavoidable latency that exists at the application initialization/bootstrap layer (several minutes or more)
- Processes that can only happen at initial boot: applying updates, data or state hydration, running configuration scripts...
- Solution was to over-provision compute resources to absorb unexpected demand increases (increased cost) or use Golden Images to try to reduce boot time
- New solution: ASG Warm Pools !!

# ASG – Warm Pools

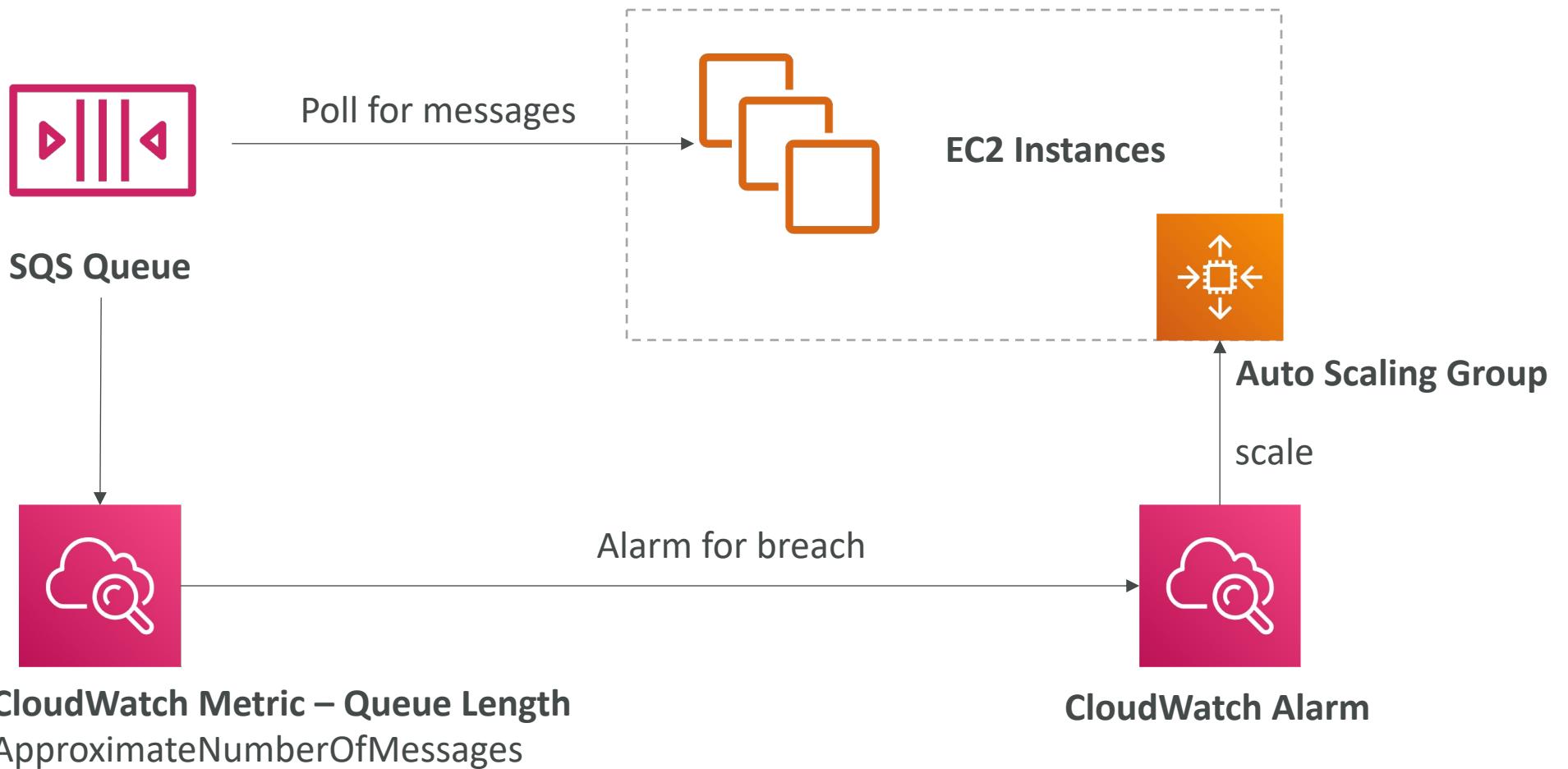
- Reduces scale-out latency by maintaining a pool of pre-initialized instances
- In a scale-out event, ASG uses the pre-initialized instances from the Warm Pool instead of launching new instances
- Warm Pool Size Settings
  - Minimum warm pool size (always in the warm pool)
  - Max prepared capacity = Max capacity of ASG (default)
  - OR Max prepared capacity = Set number of instances
- Warm Pool Instance State – what state to keep your Warm Pool instances in after initialization  
**(Running, Stopped, Hibernated)**
- Warm Pools instances don't contribute to ASG metrics that affect Scaling Policies



# Launch Configuration vs. Launch Template

- Both:
  - ID of the Amazon Machine Image (AMI), the instance type, a key pair, security groups, and the other parameters that you use to launch EC2 instances (tags, EC2 user-data...)
  - You can't edit both Launch Configurations and Launch Templates
- **Launch Configuration (legacy):**
  - Must be re-created every time
- **Launch Template (newer):**
  - Can have multiple versions
  - Create parameters subsets (partial configuration for re-use and inheritance)
  - Provision using both On-Demand and Spot instances (or a mix)
  - Supports Placement Groups, Capacity Reservations, Dedicated hosts, multiple instance types
  - Can use T2 unlimited burst feature
  - Recommended by AWS going forward

# SQS with Auto Scaling Group (ASG)



# ASG Health Checks

- To make sure you have high availability, means you have least 2 instances running across 2 AZ in your ASG (must configure multi-AZ ASG)
- Health checks available:
  - EC2 Status Checks
  - ELB Health Checks
  - Custom Health Checks: send instance's health to ASG using AWS CLI or AWS SDK
- ASG will launch a new instance after terminating an unhealthy one
- ASG will not reboot unhealthy hosts for you
- Good to know CLI:
  - set-instance-health (use with Custom Health Checks)
  - terminate-instance-in-auto-scaling-group

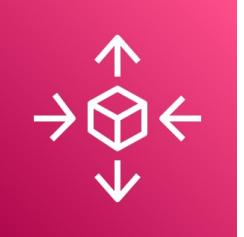
# Troubleshooting ASG issues

- <number of instances> instance(s) are already running. Launching EC2 instance failed.
  - The Auto Scaling group has reached the limit set by the MaximumCapacity parameter. Update your Auto Scaling group by providing a new value for the maximum capacity.
- Launching EC2 instances is failing:
  - The security group does not exist. SG might have been deleted
  - The key pair does not exist. The key pair might have been deleted
- If the ASG fails to launch an instance for over 24 hours, it will automatically suspend the processes (administration suspension)

# CloudWatch Metrics for ASG

- Metrics are collected every 1 minute
- ASG-level metrics: (opt-in)
  - GroupMinSize, GroupMaxSize, GroupDesiredCapacity
  - GroupInServiceInstances, GroupPendingInstances, GroupStandbyInstances
  - GroupTerminatingInstances, GroupTotalInstances
  - You should enable metric collection to see these metrics
- EC2-level metrics (enabled): CPU Utilization, etc...
  - Basic monitoring: 5 minutes granularity
  - Detailed monitoring: 1 minute granularity

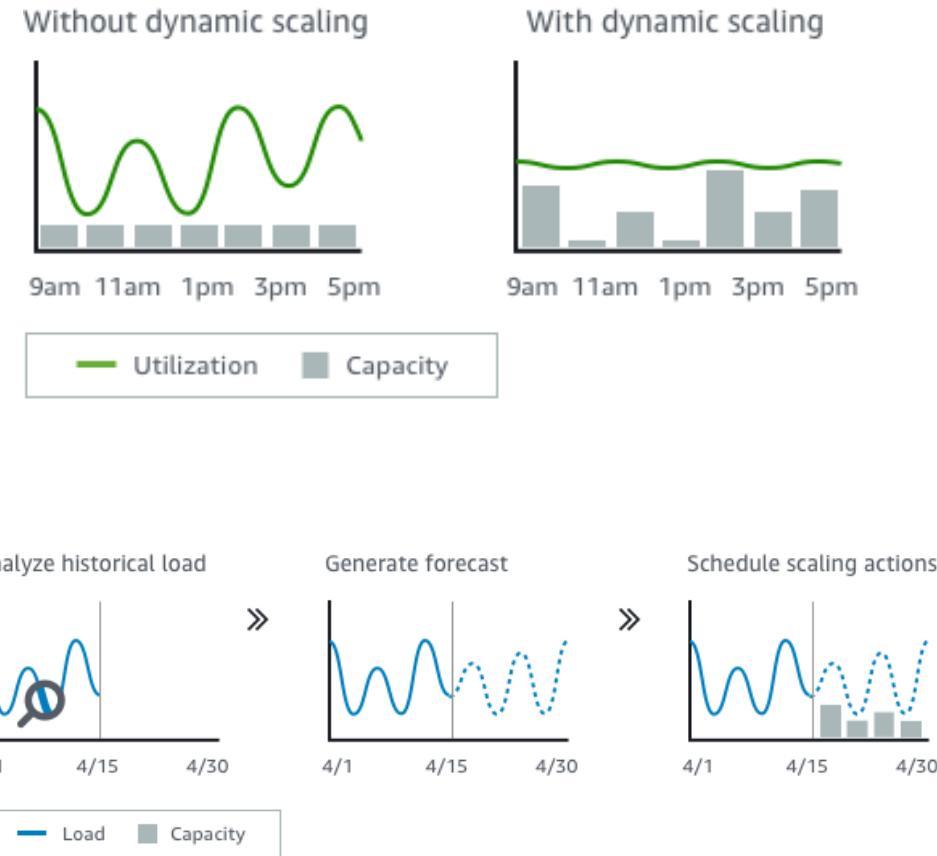
# AWS Auto Scaling



- Backbone service of auto scaling for scalable resources in AWS:
- **Amazon EC2 Auto Scaling groups:** Launch or terminate EC2 instances
- **Amazon EC2 Spot Fleet requests:** Launch or terminate instances from a Spot Fleet request, or automatically replace instances that get interrupted for price or capacity reasons.
- **Amazon ECS:** Adjust the ECS service desired count up or down
- **Amazon DynamoDB (table or global secondary index):** WCU & RCU
- **Amazon Aurora:** Dynamic Read Replicas Auto Scaling

# AWS Auto Scaling – Scaling Plans

- **Dynamic scaling:** creates a target tracking scaling policy
  - Optimize for availability => 40% of resource utilization
  - Balance availability and cost => 50% of resource utilization
  - Optimize for cost => 70% of resource utilization
  - Custom => choose own metric and target value
  - Options: Disable scale-in, cooldown period, warmup time (for ASG)
- **Predictive scaling:** continuously forecast load and schedule scaling ahead



<https://docs.aws.amazon.com/autoscaling/plans/userguide/how-it-works.html>

# AWS CloudFormation

Managing your infrastructure as code

# AWS 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 two Elastic IPs for these EC2 instances
  - I want an S3 bucket
  - I want a load balancer (ELB) in front of these EC2 instances
- Then CloudFormation creates those for you, in the **right order**, with the **exact configuration** that you specify

# CloudFormation – Template Example

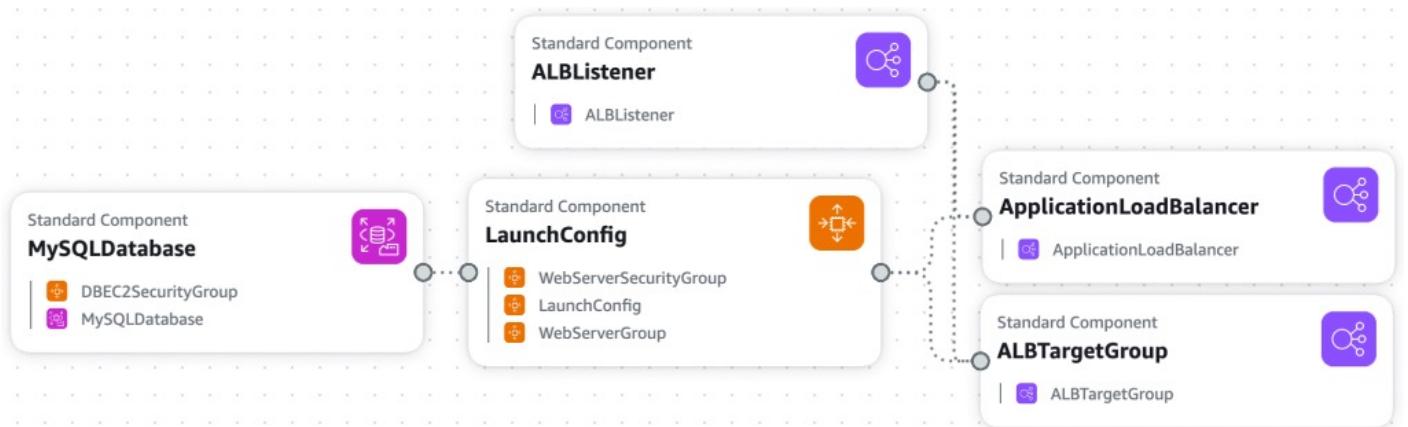
```

1 AWSTemplateFormatVersion: '2010-09-09'
2 >Description: 'AWS CloudFormation Sample Template LAMP_Multi_AZ: C
10 >Parameters:...
175 >Mappings:...
481 Resources:
482   ApplicationLoadBalancer:
483     Type: AWS::ElasticLoadBalancingV2::LoadBalancer
484     Properties:
485       Subnets: !Ref Subnets
486     ALBListener:
487       Type: AWS::ElasticLoadBalancingV2::Listener
488       Properties:
489         DefaultActions:
490           - Type: forward
491             TargetGroupArn: !Ref ALBTargetGroup
492             LoadBalancerArn: !Ref ApplicationLoadBalancer
493             Port: '80'
494             Protocol: HTTP
495     ALBTargetGroup:
496       Type: AWS::ElasticLoadBalancingV2::TargetGroup
497       Properties:
498         HealthCheckIntervalSeconds: 10
499         HealthCheckTimeoutSeconds: 5
500         HealthyThresholdCount: 2
501         Port: 80
502         Protocol: HTTP
503         UnhealthyThresholdCount: 5
504         VpcId: !Ref VpcId
505         TargetGroupAttributes:
506           - Key: stickiness.enabled
507             Value: 'true'

```



## Infrastructure Composer



# Benefits of AWS CloudFormation (1/2)

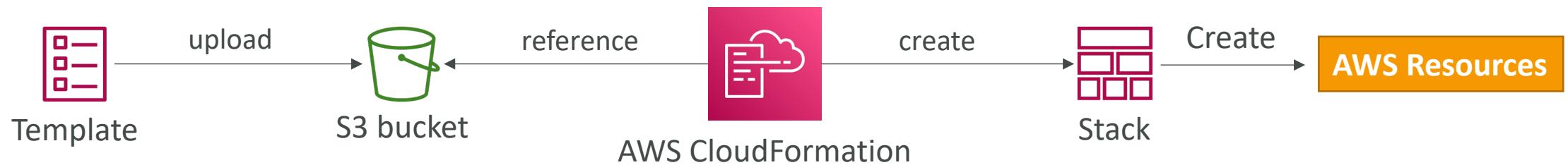
- **Infrastructure as code**
  - No resources are manually created, which is excellent for control
  - The code can be version controlled for example using Git
  - Changes to the infrastructure are reviewed through code
- **Cost**
  - Each resources within the stack is tagged with an identifier so you 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)
- **Separation of concern: create many stacks for many apps, and many layers.** Ex:
  - VPC stacks
  - Network stacks
  - App stacks
- **Don't re-invent the wheel**
  - Leverage existing templates on the web!
  - Leverage the documentation

# How CloudFormation Works

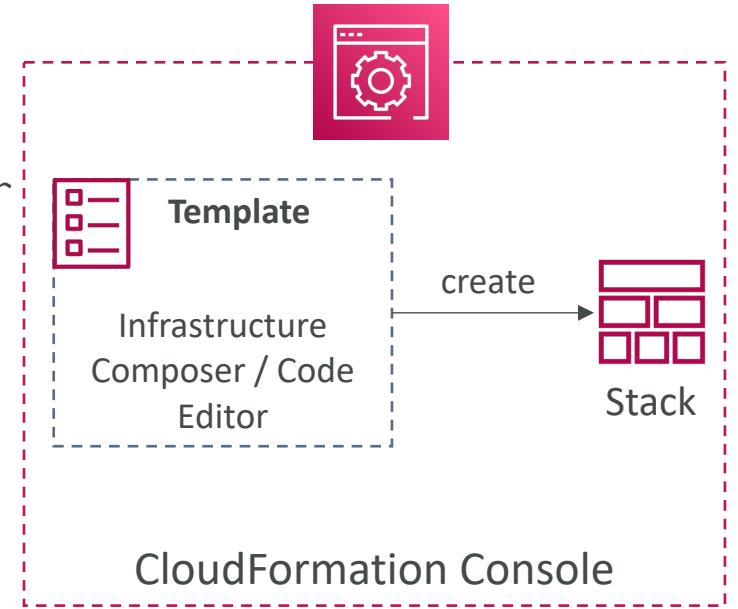
- Templates must be uploaded in S3 and then referenced in CloudFormation
- To update a template, we can't edit previous ones. We have to re-upload a new version of the template to AWS
- Stacks are identified by a name
- Deleting a stack deletes every single artifact that was created by CloudFormation.



# Deploying CloudFormation Templates

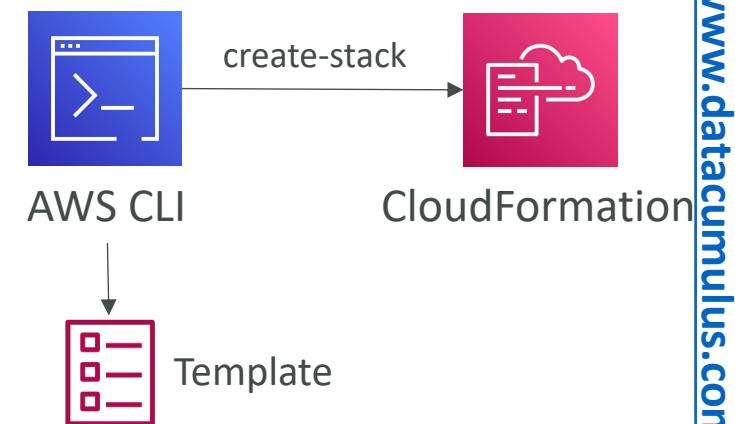
- **Manual way**

- Editing templates in Infrastructure Composer or code editor
- Using the console to input parameters, etc...
- We'll mostly do this way in the course for learning purposes



- **Automated way**

- Editing templates in a YAML file
- Using the AWS CLI (Command Line Interface) to deploy the templates, or using a Continuous Delivery (CD) tool
- Recommended way when you fully want to automate your flow

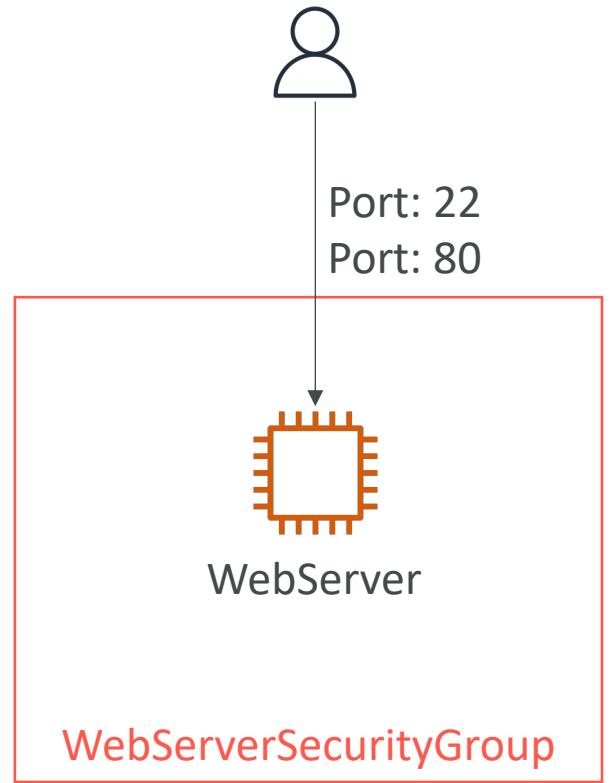


# CloudFormation – Building Blocks

- Template's Components
  - AWSTemplateFormatVersion – identifies the capabilities of the template “2010-09-09”
  - Description – comments about the template
  - Resources (**MANDATORY**) – your AWS resources declared in the template
  - Parameters – the dynamic inputs for your template
  - Mappings – the static variables for your template
  - Outputs – references to what has been created
  - Conditionals – list of conditions to perform resource creation
- Template's Helpers
  - References
  - Functions

# Introductory Example

- We're going to create a simple EC2 instance
  - And we're going to add security group to it
  - For now, forget about the code syntax
  - We'll look at the structure of the files later
- 
- We'll see how in no-time, we are able to get started with CloudFormation!



# YAML Crash Course

```
invoice: 34843
date: 2001-01-23
bill-to:
  given: Chris
  family: Dumars
  address:
    lines: |
      458 Walkman Dr.
      Suite #292
    city: Royal Oak
    state: MI
    postal: 48046
products:
  - sku: BL394D
    quantity: 4
    description: Basketball
    price: 450.00
  - sku: BL4438H
    quantity: 1
    description: Super Hoop
    price: 2392.00
```

- YAML and JSON are the languages you can use for CloudFormation
- JSON is horrible for CF
- YAML is great in so many ways
- Let's learn a bit about it!
  - Key value Pairs
  - Nested objects
  - Support Arrays
  - Multi line strings
  - Can include comments!

# CloudFormation – Resources

- Resources are the core of your CloudFormation template (**MANDATORY**)
- They represent the different AWS Components that will be created and configured
- Resources are declared and can reference each other
- AWS figures out creation, updates and deletes of resources for us
- There are over 700 types of resources (!)
- Resource types identifiers are of the form:

***service-provider::service-name::data-type-name***

# How do I find Resources documentation?

- I can't teach you all the 700+ resources, but I can teach you how to learn how to use them
- All the resources can be found here:  
<https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-template-resource-type-ref.html>
- Then, we just read the docs ☺
- Example here (for an EC2 instance):  
<https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-ec2-instance.html>

# Analysis of CloudFormation Template

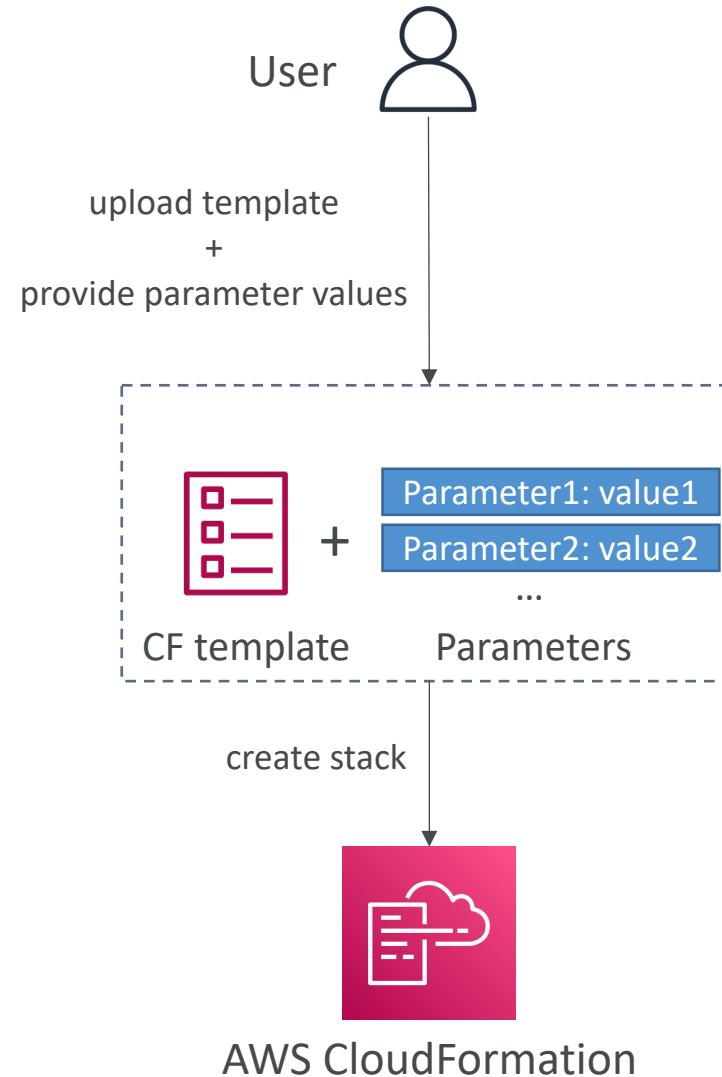
- Going back to the example of the introductory lecture, let's learn why it was written this way.
- Relevant documentation can be found here:
  - <https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-ec2-instance.html>
  - <https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-ec2-securitygroup.html>
  - <http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/aws-resource-ec2-eip.html>

# CloudFormation – Resources FAQ

- Can I create a dynamic number of resources?
  - Yes, you can by using CloudFormation Macros and Transform
  - It is not in the scope of this course
- Is every AWS Service supported?
  - Almost. Only a select few niches are not there yet
  - You can work around that using CloudFormation Custom Resources

# CloudFormation – Parameters

- Parameters are a way to provide inputs to your AWS CloudFormation template
- They're important to know about if:
  - You want to reuse your templates across the company
  - Some inputs can not be determined ahead of time
- Parameters are extremely powerful, controlled, and can prevent errors from happening in your templates, thanks to types



# When should you use a Parameter?

**Parameters:**

**SecurityGroupDescription:**

**Description:** Security Group Description

**Type:** String

- Ask yourself this:
  - Is this CloudFormation resource configuration likely to change in the future?
  - If so, make it a parameter
- You won't have to re-upload a template to change its content ☺

# CloudFormation – Parameters Settings

- Parameters can be controlled by all these settings:
  - Type:
    - String
    - Number
    - CommaDelimitedList
    - List<Number>
    - AWS-Specific Parameter (to help catch invalid values – match against existing values in the AWS account)
    - List<AWS-Specific Parameter>
    - SSM Parameter (get parameter value from SSM Parameter store)
  - Description
  - ConstraintDescription (String)
  - Min/MaxLength
  - Min/MaxValue
  - Default
  - AllowedValues (array)
  - AllowedPattern (regex)
  - NoEcho (Boolean)

# CloudFormation – Parameters Example

## AllowedValues

### Parameters:

#### InstanceType:

Description: Choose an EC2 instance type

Type: String

#### AllowedValues:

- t2.micro
- t2.small
- t2.medium

Default: t2.micro

### Resources:

#### MyEC2Instance:

Type: AWS::EC2::Instance

#### Properties:

InstanceType: !Ref InstanceType  
ImageId: ami-0c02fb55956c7d316

## NoEcho

### Parameters:

#### DBPassword:

Description: The database admin password

Type: String

NoEcho: true

### Resources:

#### MyDBInstance:

Type: AWS::RDS::DBInstance

#### Properties:

DBInstanceClass: db.t2.micro  
AllocatedStorage: 20  
Engine: mysql  
MasterUsername: admin  
MasterUserPassword: !Ref DBPassword  
DBInstanceIdentifier: mydbinstance

# How to Reference a Parameter?

**Resources:**

**DBSubnet1:**

Type: AWS::EC2::Subnet

**Properties:**

VpcId: !Ref MyVPC

- The Fn::Ref function can be leveraged to reference parameters
- Parameters can be used anywhere in a template
- The shorthand for this in YAML is !Ref
- The function can also reference other elements within the template

# CloudFormation – Pseudo Parameters

- AWS offers us Pseudo Parameters in any CloudFormation template
- These can be used at any time and are enabled by default
- Important pseudo parameters:

Reference Value	Example Returned Value
AWS::AccountId	123456789012
AWS::Region	us-east-1
AWS::StackId	arn:aws:cloudformation:us-east-1:123456789012:stack/MyStack/1c2fa620-982a-11e3-aff7-50e2416294e0
AWS::StackName	MyStack
AWS::NotificationARNs	[arn:aws:sns:us-east-1:123456789012:MyTopic]
AWS::NoValue	Doesn't return a value

# CloudFormation – Mappings

- Mappings are fixed variables within your CloudFormation template
- They're very handy to differentiate between different environments (dev vs prod), regions (AWS regions), AMI types...
- All the values are hardcoded within the template

Mappings:

Mapping01:

Key01:

Name: Value01

Key02:

Name: Value02

Key03:

Name: Value03

RegionMap:

us-east-1:

HVM64: ami-0ff8a91507f77f867

HVMG2: ami-0a584ac55a7631c0c

us-west-1:

HVM64: ami-0bdb828fd58c52235

HVMG2: ami-066ee5fd4a9ef77f1

eu-west-1:

HVM64: ami-047bb4163c506cd98

HVMG2: ami-0a7c483d527806435

# Accessing Mapping Values (Fn::FindInMap)

- We use `Fn::FindInMap` to return a named value from a specific key
- `!FindInMap [ MapName, TopLevelKey, SecondLevelKey ]`

Mappings:

RegionMap:

us-east-1:

HVM64: ami-0ff8a91507f77f867  
HVMG2: ami-0a584ac55a7631c0c

us-west-1:

HVM64: ami-0bdb828fd58c52235  
HVMG2: ami-066ee5fd4a9ef77f1

Mappings work great for AMIs  
Because AMIs are region-specific!

Resources:

MyEC2Instance:

Type: AWS::EC2::Instance

Properties:

ImageId: `!FindInMap [RegionMap, !Ref "AWS::Region", HVM64]`

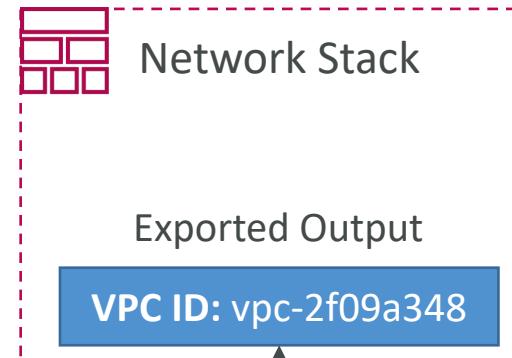
InstanceType: t2.micro

# When would you use Mappings vs. Parameters?

- Mappings are great when you know in advance all the values that can be taken and that they can be deduced from variables such as
  - Region
  - Availability Zone
  - AWS Account
  - Environment (dev vs prod)
  - etc...
- They allow safer control over the template
- Use parameters when the values are really user specific

# CloudFormation – Outputs

- The Outputs section declares *optional* outputs values that we can import into other stacks (if you export them first)!
- You can also view the outputs in the AWS Console or in using the AWS CLI
- They're very useful for example if you define a network CloudFormation, and output the variables such as VPC ID and your Subnet IDs
- It's the best way to perform some collaboration cross stack, as you let expert handle their own part of the stack



# CloudFormation – Outputs

- Creating a SSH Security Group as part of one template
- We create an output that references that security group

## Outputs:

### StackSSHSecurityGroup:

Description: The SSH Security Group for our Company

Value: !Ref MyCompanyWideSSHSecurityGroup

### Export:

Name: SSHSecurityGroup

# CloudFormation – Outputs Cross-Stack Reference

- We then create a second template that leverages that security group
- For this, we use the `Fn::ImportValue` function
- You can't delete the underlying stack until all the references are deleted

**Resources:**

**MySecureInstance:**

**Type:** AWS::EC2::Instance

**Properties:**

**ImageId:** ami-0742b4e673072066f

**InstanceType:** t2.micro

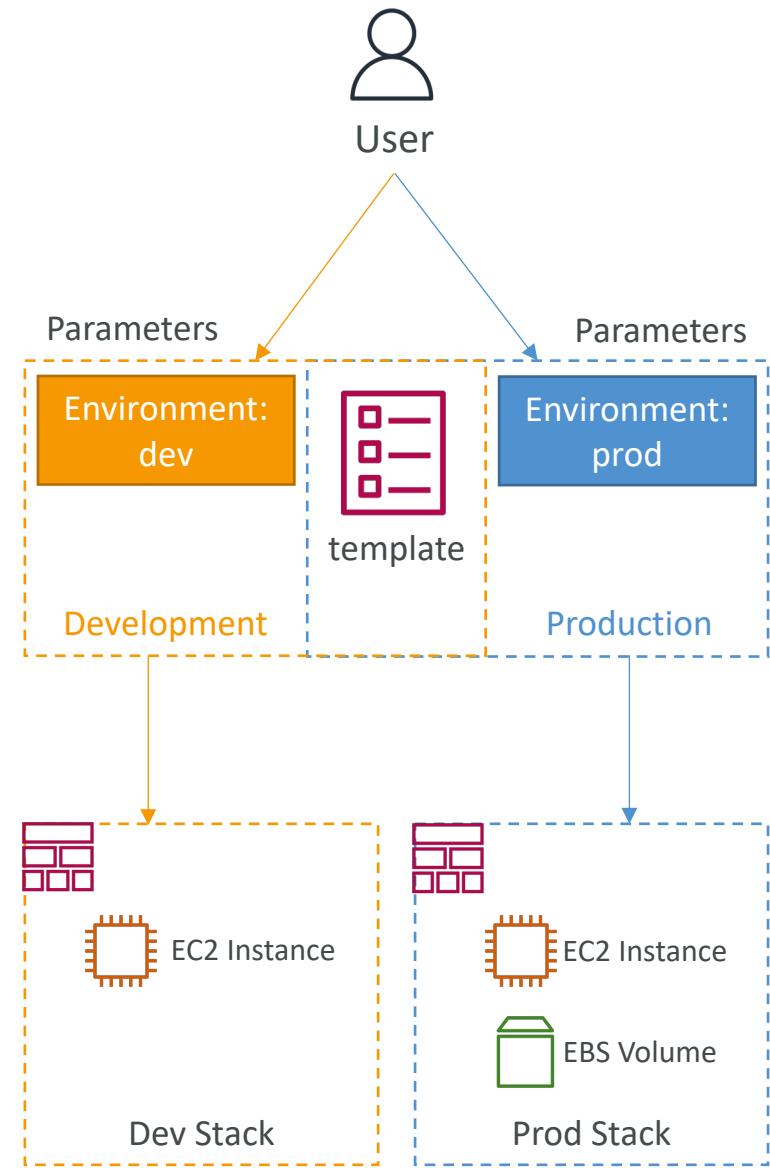
**AvailabilityZone:** us-east-1a

**SecurityGroups:**

- !ImportValue SSHSecurityGroup

# CloudFormation – Conditions

- Conditions are used to control the creation of resources or outputs based on a condition
- Conditions can be whatever you want them to be, but common ones are:
  - Environment (dev / test / prod)
  - AWS Region
  - Any parameter value
- Each condition can reference another condition, parameter value or mapping



# How to define a Condition

## Conditions:

```
CreateProdResources: !Equals [ !Ref EnvType, prod ]
```

- The logical ID is for you to choose. It's how you name condition
- The intrinsic function (logical) can be any of the following:
  - Fn::And
  - Fn::Equals
  - Fn::If
  - Fn::Not
  - Fn::Or

# How to use a Condition

- Conditions can be applied to resources / outputs / etc...

**Resources:**

**MountPoint:**

**Type:** AWS::EC2::VolumeAttachment

**Condition:** CreateProdResources

# CloudFormation – Intrinsic Functions

Blue = must know

- Ref
- Fn::GetAtt
- Fn::FindInMap
- Fn::ImportValue
- Fn::Join
- Fn::Sub
- Fn::ForEach
- Fn::ToJsonString
- Condition Functions (Fn::If, Fn::Not, Fn::Equals, etc...)
- Fn::Base64
- Fn::Cidr
- Fn::GetAZs
- Fn::Select
- Fn::Split
- Fn::Transform
- Fn::Length

# Intrinsic Functions – Fn::Ref

- The **Fn::Ref** function can be leveraged to reference
  - **Parameters** – returns the value of the parameter
  - **Resources** – returns the physical ID of the underlying resource (e.g., EC2 ID)
- The shorthand for this in YAML is **!Ref**

**Resources:**

**DBSubnet1:**

**Type: AWS::EC2::Subnet**

**Properties:**

**VpcId: !Ref MyVPC**

# Intrinsic Functions – Fn::GetAtt

- Attributes are attached to any resources you create
- To know the attributes of your resources, the best place to look at is the documentation
- Example: the AZ of an EC2 instance!

**Resources:**

**EC2Instance:**

Type: AWS::EC2::Instance

Properties:

ImageId: ami-0742b4e673072066f

InstanceType: t2.micro

**EBSVolume:**

Type: AWS::EC2::Volume

Condition: CreateProdResources

Properties:

Size: 100

AvailabilityZone: !GetAtt EC2Instance.AvailabilityZone

# Intrinsic Functions – Fn::FindInMap

- We use `Fn::FindInMap` to return a named value from a specific key
- `!FindInMap [ MapName, TopLevelKey, SecondLevelKey ]`

```
Mappings:  
RegionMap:  
    us-east-1:  
        HVM64: ami-0ff8a91507f77f867  
        HVMG2: ami-0a584ac55a7631c0c  
    us-west-1:  
        HVM64: ami-0bdb828fd58c52235  
        HVMG2: ami-066ee5fd4a9ef77f1  
  
Resources:  
MyEC2Instance:  
    Type: AWS::EC2::Instance  
    Properties:  
        ImageId: !FindInMap [RegionMap, !Ref "AWS::Region", HVM64]  
        InstanceType: t2.micro
```

# Intrinsic Functions – Fn::ImportValue

- Import values that are exported in other stacks
- For this, we use the **Fn::ImportValue** function

**Resources:**

**MySecureInstance:**

**Type:** AWS::EC2::Instance

**Properties:**

**ImageId:** ami-0742b4e673072066f

**InstanceType:** t2.micro

**AvailabilityZone:** us-east-1a

**SecurityGroups:**

- !ImportValue SSHSecurityGroup

# Intrinsic Functions – Fn::Base64

- Convert String to it's Base64 representation

```
!Base64 "ValueToEncode"
```

- Example: pass encoded data to EC2 Instance's **UserData** property

Resources:

WebServer:

Type: AWS::EC2::Instance

Properties:

...

UserData:

```
Fn::Base64: |
#!/bin/bash
dnf update -y
dnf install -y httpd
```

# Intrinsic Functions – Condition Functions

## Conditions:

`CreateProdResources: !Equals [ !Ref EnvType, prod ]`

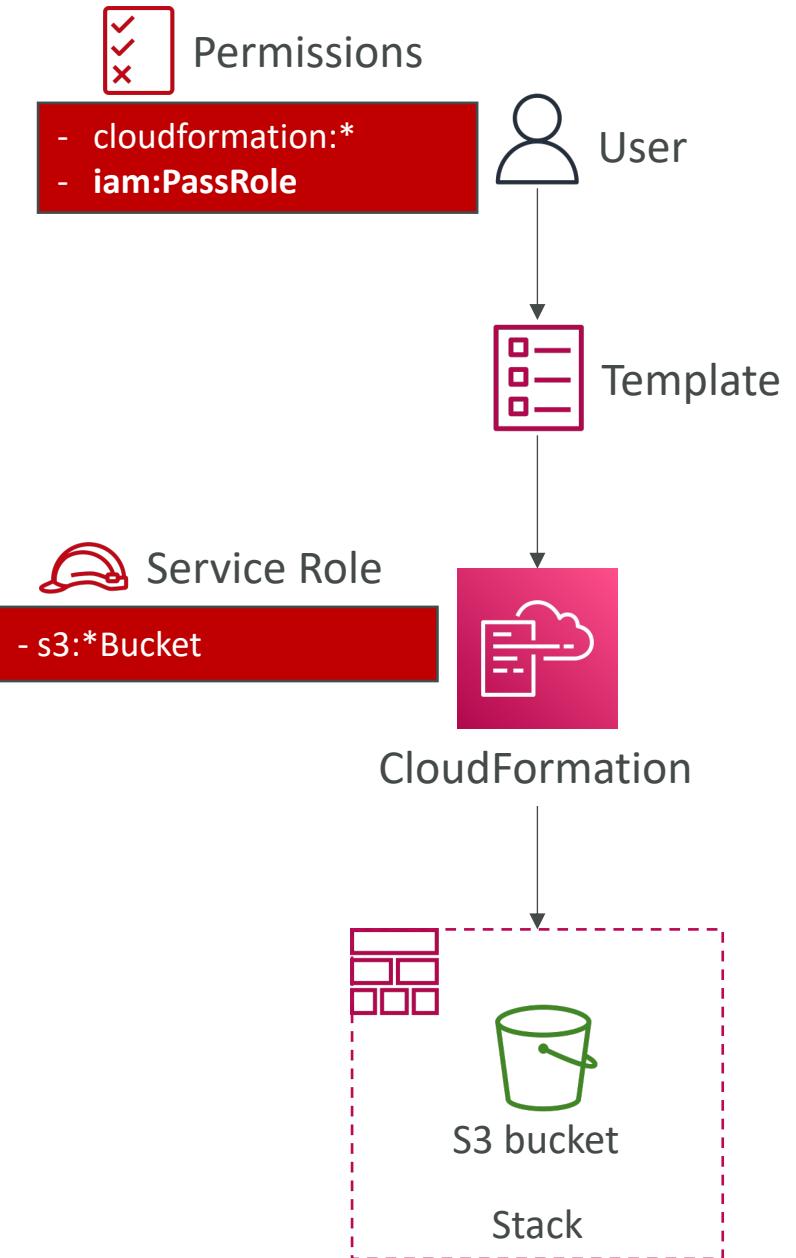
- The logical ID is for you to choose. It's how you name condition
- The intrinsic function (logical) can be any of the following:
  - Fn::And
  - Fn::Equals
  - Fn::If
  - Fn::Not
  - Fn::Or

# CloudFormation – Rollbacks

- Stack Creation Fails:
  - Default: everything rolls back (gets deleted). We can look at the log
  - Option to disable rollback and troubleshoot what happened
- Stack Update Fails:
  - The stack automatically rolls back to the previous known working state
  - Ability to see in the log what happened and error messages
- Rollback Failure? Fix resources manually then issue **ContinueUpdateRollback API** from Console
  - Or from the CLI using continue-update-rollback API call

# CloudFormation – Service Role

- IAM role that allows CloudFormation to create/update/delete stack resources on your behalf
- Give ability to users to create/update/delete the stack resources even if they don't have permissions to work with the resources in the stack
- Use cases:
  - You want to achieve the least privilege principle
  - But you don't want to give the user all the required permissions to create the stack resources
- User must have **iam:PassRole** permissions



# CloudFormation Capabilities

- **CAPABILITY\_NAMED\_IAM** and **CAPABILITY\_IAM**
  - Necessary to enable when your CloudFormation template is creating or updating IAM resources (IAM User, Role, Group, Policy, Access Keys, Instance Profile...)
  - Specify **CAPABILITY\_NAMED\_IAM** if the resources are named
- **CAPABILITY\_AUTO\_EXPAND**
  - Necessary when your CloudFormation template includes Macros or Nested Stacks (stacks within stacks) to perform dynamic transformations
  - You're acknowledging that your template may change before deploying
- **InsufficientCapabilitiesException**
  - Exception that will be thrown by CloudFormation if the capabilities haven't been acknowledged when deploying a template (security measure)

# CloudFormation – DeletionPolicy Delete

- **DeletionPolicy:**
  - Control what happens when the CloudFormation template is deleted or when a resource is removed from a CloudFormation template
  - Extra safety measure to preserve and backup resources
- Default DeletionPolicy=Delete
  - ! Delete won't work on an S3 bucket if the bucket is not empty

**Resources:**

**MyEC2Instance:**

Type: AWS::EC2::Instance

**Properties:**

ImageId: ami-12345678

InstanceType: t2.micro

KeyName: my-key-pair

SecurityGroupIds:

- sg-12345678

**DeletionPolicy: Delete**

**Resources:**

**MyS3Bucket:**

Type: AWS::S3::Bucket



**DeletionPolicy: Delete**

# CloudFormation – DeletionPolicy Retain

- `DeletionPolicy=Retain`:
  - Specify on resources to preserve in case of CloudFormation deletes
  - Works with any resources

## Resources:

`MyDynamoDBTable`:

`Type: AWS::DynamoDB::Table`

### Properties:

`TableName: MyTable`

### AttributeDefinitions:

`- AttributeName: ID`

`AttributeType: S`

### KeySchema:

`- AttributeName: ID`

`KeyType: HASH`

### ProvisionedThroughput:

`ReadCapacityUnits: 5`

`WriteCapacityUnits: 5`

`DeletionPolicy: Retain`

# CloudFormation – DeletionPolicy Snapshot

- `DeletionPolicy=Snapshot`
- Create one final snapshot before deleting the resource
- Examples of supported resources:
  - EBS Volume, ElastiCache Cluster, ElastiCache ReplicationGroup
  - RDS DBInstance, RDS DBCluster, Redshift Cluster, Neptune DBCluster, DocumentDB DBCluster

**Resources:**

`MyDBInstance`:

`Type: AWS::RDS::DBInstance`

**Properties:**

`DBInstanceClass: db.t2.micro`

`AllocatedStorage: 20`

`Engine: mysql`

`MasterUsername: admin`

`MasterUserPassword: "ExamplePassword"`

`DeletionPolicy: Snapshot`

# CloudFormation – UpdateReplacePolicy

- Controls what happens to a resource if you update a property whose update behavior is **Replacement**
- For example, updating RDS DBInstance's AvailabilityZone property
- **UpdateReplacePolicy=Delete (default)**
  - CloudFormation deletes the old resource and creates a new one with a new physical ID
- **UpdateReplacePolicy=Retain**
  - Keeps the resource (it is removed from CloudFormation's scope)
- **UpdateReplacePolicy=Snapshot**
  - EBS Volume, ElastiCache Cluster, ElastiCache ReplicationGroup
  - RDS DBInstance, RDS DBCluster, Redshift Cluster, Neptune DBCluster
  - The snapshot doesn't exist in CloudFormation's scope

# UpdateReplacePolicy vs. DeletionPolicy

```
Resources:  
  MyDB:  
    Type: AWS::RDS::DBInstance  
    DeletionPolicy: Retain  
    UpdateReplacePolicy: Retain  
    Properties: {}
```

- `UpdateReplacePolicy` only applies to resources replaced during stack updates
- `DeletionPolicy`
  - Applies to resources deleted when a stack is deleted
  - Applies when a resource definition deleted from the template as part of stack update

# CloudFormation – Stack Policies

- During a CloudFormation Stack update, all update actions are allowed on all resources (default)
- A Stack Policy is a JSON document that defines the update actions that are allowed on specific resources during Stack updates
- Protect resources from unintentional updates
- When you set a Stack Policy, all resources in the Stack are protected by default
- Specify an explicit ALLOW for the resources you want to be allowed to be updated

```
{  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": "Update:*",  
      "Principal": "*",  
      "Resource": "*"  
    },  
    {  
      "Effect": "Deny",  
      "Action": "Update:*",  
      "Principal": "*",  
      "Resource": "LogicalResourceId/ProductionDatabase"  
    }  
  ]  
}
```

Allow updates on all resources  
**except** the ProductionDatabase

# CloudFormation – Termination Protection

- To prevent accidental deletes of CloudFormation Stacks, use `TerminationProtection`

# CloudFormation – Custom Resources

- Used to
  - define resources not yet supported by CloudFormation
  - define custom provisioning logic for resources can that be outside of CloudFormation (on-premises resources, 3<sup>rd</sup> party resources...)
  - have custom scripts run during create / update / delete through Lambda functions (running a Lambda function to empty an S3 bucket before being deleted)
- Defined in the template using `AWS::CloudFormation::CustomResource` or `Custom::MyCustomResourceType` (recommended)
- Backed by a Lambda function (most common) or an SNS topic

# How to define a Custom Resource?

- **ServiceToken** specifies where CloudFormation sends requests to, such as Lambda ARN or SNS ARN (required & must be in the same region)
- Input data parameters (optional)



## Resources:

### MyCustomResourceUsingLambda:

Type: Custom::MyLambdaResource

#### Properties:

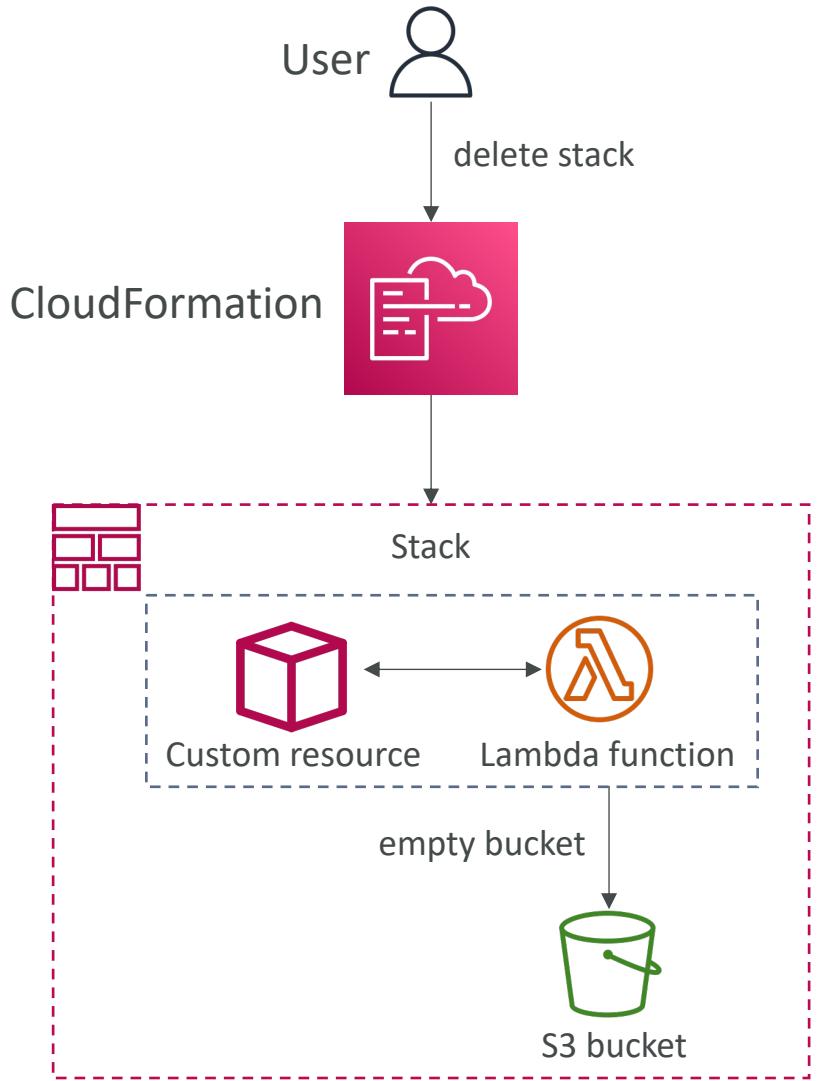
ServiceToken: arn:aws:lambda:REGION:ACCOUNT\_ID:function:FUNCTION\_NAME

# Input values (optional)

ExampleProperty: "ExampleValue"

# Use Case – Delete content from an S3 bucket

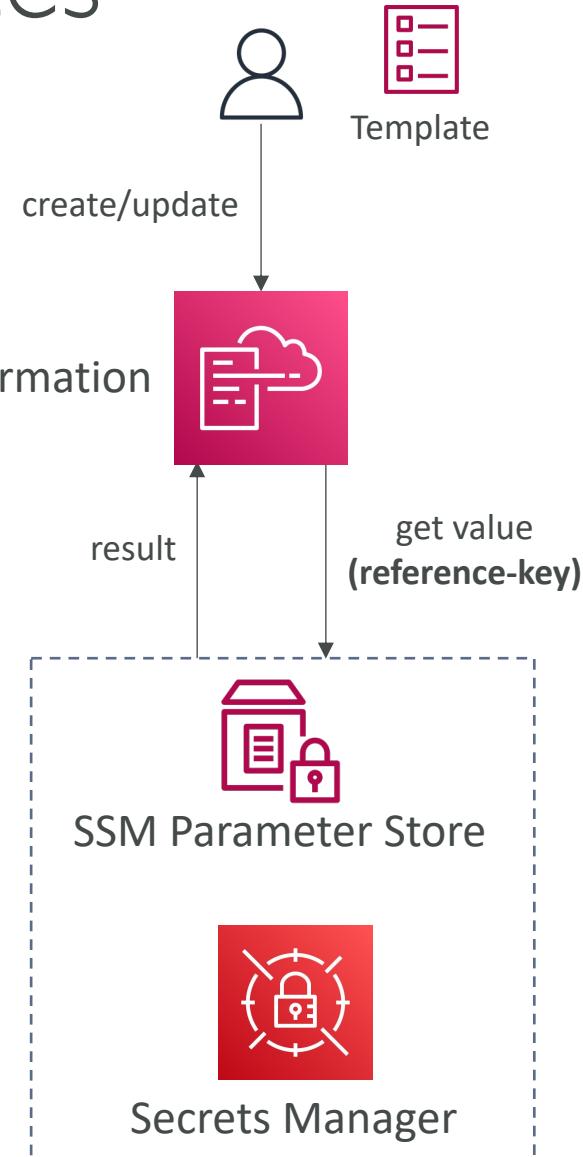
- You can't delete a non-empty S3 bucket
- To delete a non-empty S3 bucket, you must first delete all the objects inside it
- We can use a custom resource to empty an S3 bucket before it gets deleted by CloudFormation



# CloudFormation – Dynamic References

- Reference external values stored in **Systems Manager Parameter Store** and **Secrets Manager** within CloudFormation templates
- CloudFormation retrieves the value of the specified reference during **create/update/delete operations**
- For example: retrieve RDS DB Instance master password from Secrets Manager
- Supports
  - **ssm** – for plaintext values stored in SSM Parameter Store
  - **ssm-secure** – for secure strings stored in SSM Parameter Store
  - **secretsmanager** – for secret values stored in Secrets Manager

`‘{{resolve:service-name:reference-key}}’`



# CloudFormation – Dynamic References

## SSM

```
{resolve:ssm:parameter-name:version}}
```

```
Resources:  
S3Bucket:  
  Type: AWS::S3::Bucket  
Properties:  
  AccessControl: '{{resolve:ssm:S3AccessControl:2}}'
```

## SSM Secure

```
{resolve:ssm-secure:parameter-name:version}}
```

```
Resources:  
IAMUser:  
  Type: AWS::IAM::User  
Properties:  
  UserName: john  
  LoginProfile:  
    Password: '{{resolve:ssm-secure:IAMUserPassword:10}}'
```

## Secrets Manager

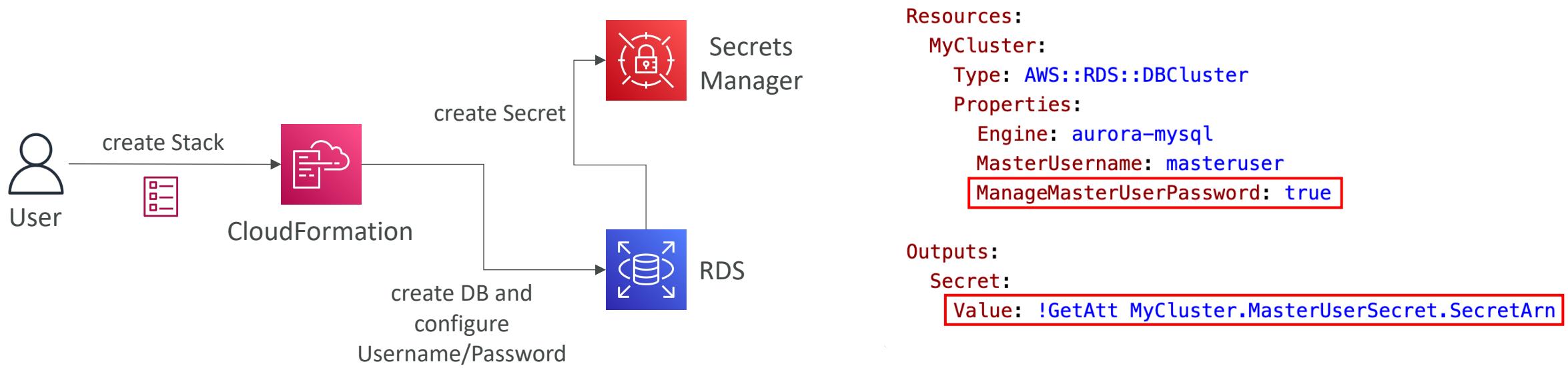
```
{resolve:secretsmanager:secret-id:secret-string:json-key:version-stage:version-id}}
```

```
Resources:  
DBInstance:  
  Type: AWS::RDS::DBInstance  
Properties:  
  DBName: MyRDSInstance  
  MasterUsername: '{{resolve:secretsmanager:MyRDSSecret:SecretString:username}}'  
  MasterUserPassword: '{{resolve:secretsmanager:MyRDSSecret:SecretString:password}}'
```

# CloudFormation, Secrets Manager & RDS

## Option I – ManageMasterUserPassword

- `ManageMasterUserPassword` – creates admin secret implicitly
- RDS, Aurora will manage the secret in Secrets Manager and its rotation



# CloudFormation, Secrets Manager & RDS

## Option 2 – Dynamic Reference

Resources:

```
MyDatabaseSecret:  
  Type: AWS::SecretsManager::Secret  
  Properties:  
    Name: MyDatabaseSecret  
    GenerateSecretString:  
      SecretStringTemplate: '{"username": "admin"}'  
      GenerateStringKey: "password"  
      PasswordLength: 16  
      ExcludeCharacters: '"@/\\'
```

1. secret is generated

```
MyDBInstance:  
  Type: AWS::RDS::DBInstance  
  Properties:  
    DBName: mydatabase  
    AllocatedStorage: 20  
    DBInstanceClass: db.t2.micro  
    Engine: mysql
```

```
  MasterUsername: '{{resolve:secretsmanager:MyDatabaseSecret:SecretString:username}}'  
  MasterUserPassword: '{{resolve:secretsmanager:MyDatabaseSecret:SecretString:password}}'
```

2. Reference secret in  
RDS DB instance

```
SecretRDSAttachment:  
  Type: AWS::SecretsManager::SecretTargetAttachment  
  Properties:  
    SecretId: !Ref MyDatabaseSecret  
    TargetId: !Ref MyDBInstance  
    TargetType: AWS::RDS::DBInstance
```

3. link the secret to  
RDS DB instance (for rotation)

# User Data in EC2 for CloudFormation

- We can have user data at EC2 instance launch through the console
- Let's learn how to write the same EC2 user-data script in our CloudFormation template
- The important thing to pass is the entire script through the function [Fn::Base64](#)
- Good to know, user data script log is in `/var/log/cloud-init-output.log`
- Let's see how to do this in CloudFormation!

# The Problems with EC2 User Data

- What if we want to have a very large instance configuration?
- What if we want to evolve the state of the EC2 instance without terminating it and creating a new one?
- How do we make EC2 user-data more readable?
- How do we know or signal that our EC2 user-data script completed successfully?
- Enter **CloudFormation Helper Scripts!**
  - Python scripts, that come directly on Amazon Linux AMIs, or can be installed using [yum](#) or [dnf](#) on non-Amazon Linux AMIs
  - `cfn-init`, `cfn-signal`, `cfn-get-metadata`, `cfn-hup`

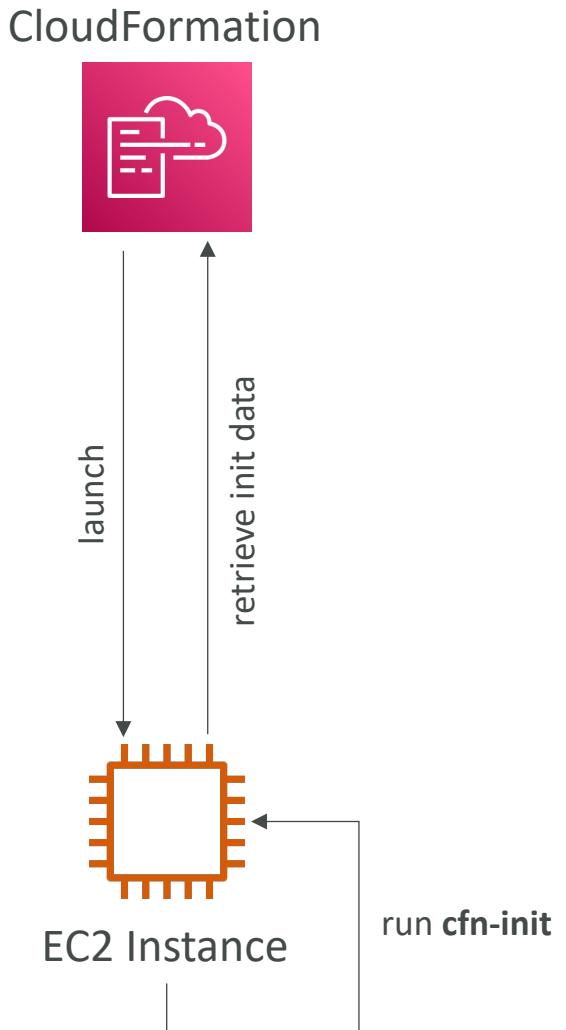
# AWS::CloudFormation::Init

- A `config` contains the following and is executed in that order
  - `Packages`: used to download and install pre-packaged apps and components on Linux/Windows (ex. MySQL, PHP, etc...)
  - `Groups`: define user groups
  - `Users`: define users, and which group they belong to
  - `Sources`: download files and archives and place them on the EC2 instance
  - `Files`: create files on the EC2 instance, using inline or can be pulled from a URL
  - `Commands`: run a series of commands
  - `Services`: launch a list of sysvinit

```
Resources:  
  EC2Instance:  
    Type: AWS::EC2::Instance  
    Properties:  
      ...  
      Metadata:  
        AWS::CloudFormation::Init:  
          config:  
            packages:  
              ...  
            groups:  
              ...  
            users:  
              ...  
            sources:  
              ...  
            files:  
              ...  
            commands:  
              ...  
            services:  
              ...
```

# CloudFormation – cfn-init

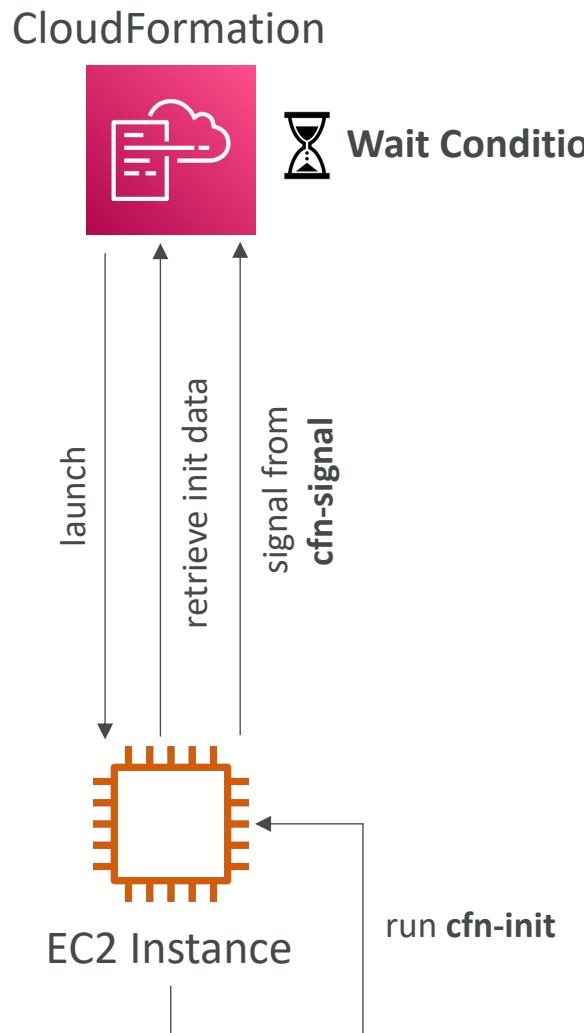
- Used to retrieve and interpret the resource metadata, installing packages, creating files and starting services
- With the **cfn-init** script, it helps make complex EC2 configurations readable
- The EC2 instance will query the CloudFormation service to get init data
- **AWS::CloudFormation::Init** must be in the Metadata of a resource
- Logs go to `/var/log/cfn-init.log`



# CloudFormation – cfn-signal & Wait Conditions

- We still don't know how to tell CloudFormation that the EC2 instance got properly configured after a `cfn-init`
- For this, we can use the `cfn-signal` script!
  - We run `cfn-signal` right after `cfn-init`
  - Tell CloudFormation service that the resource creation success/fail to keep on going or fail
- We need to define `WaitCondition`:
  - Block the template until it receives a signal from `cfn-signal`
  - We attach a `CreationPolicy` (also works on EC2, ASG)
  - We can define a `Count > 1` (in case you need more than 1 signal)

```
CreationPolicy: 
ResourceSignal: 
Timeout: PT5M
Count: 1
```

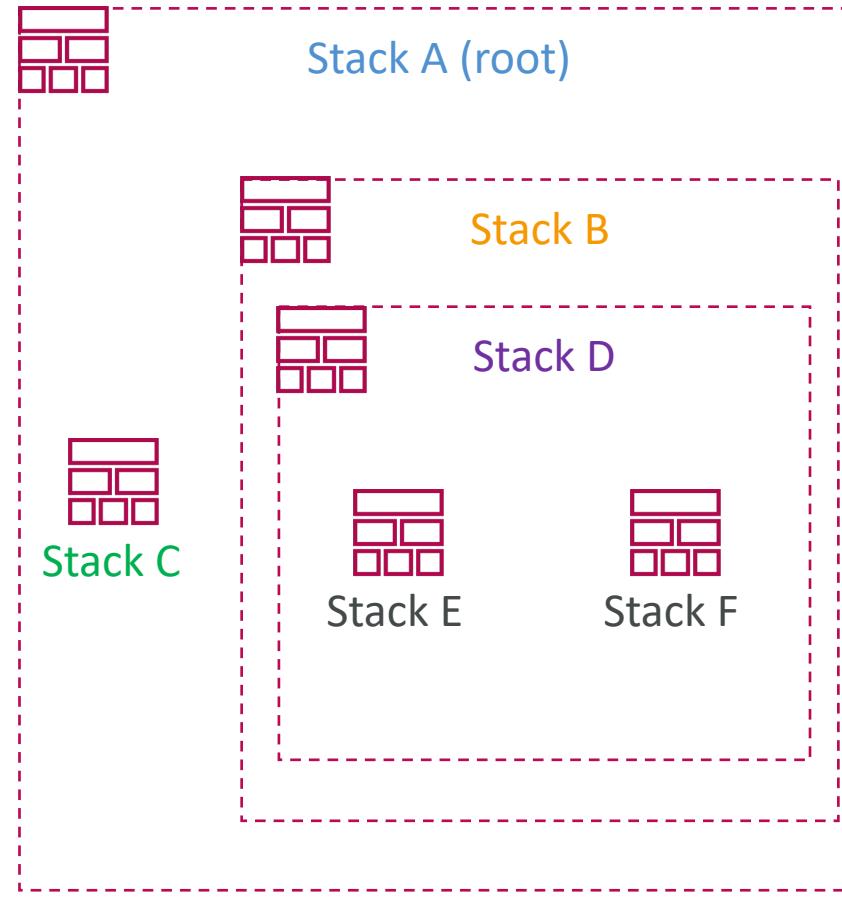


# Wait Condition Didn't Receive the Required Number of Signals from an Amazon EC2 Instance

- Ensure that the AMI you're using has the AWS CloudFormation helper scripts installed. If the AMI doesn't include the helper scripts, you can also download them to your instance
- Verify that the `cfn-init` & `cfn-signal` command was successfully run on the instance. You can view logs, such as `/var/log/cloud-init.log` or `/var/log/cfn-init.log`, to help you debug the instance launch
- You can retrieve the logs by logging in to your instance, but you must disable rollback on failure or else AWS CloudFormation deletes the instance after your stack fails to create
- Verify that the instance has a connection to the Internet. If the instance is in a VPC, the instance should be able to connect to the Internet through a NAT device if it's in a private subnet or through an Internet gateway if it's in a public subnet
- For example, run: curl -I <https://aws.amazon.com>

# CloudFormation – Nested Stacks

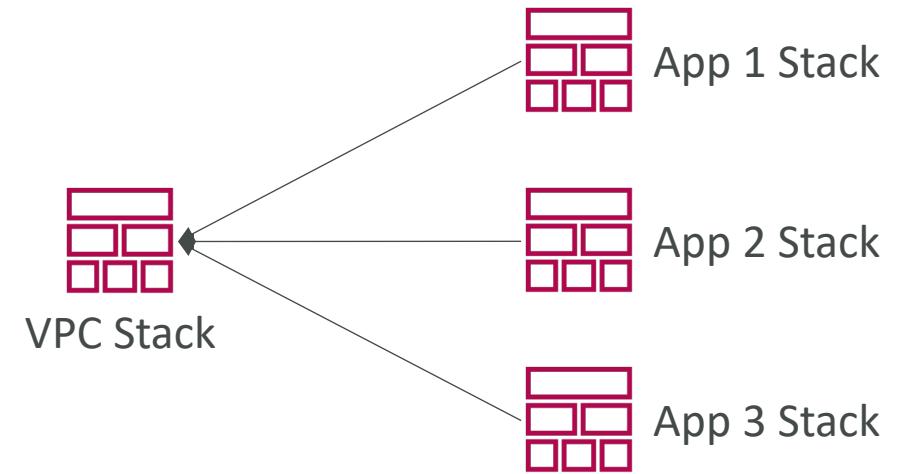
- Nested stacks are stacks as part of other stacks
- They allow you to isolate repeated patterns / common components in separate stacks and call them from other stacks
- Example:
  - Load Balancer configuration that is re-used
  - Security Group that is re-used
- Nested stacks are considered best practice
- To update a nested stack, always update the parent (root stack)
- Nested stacks can have nested stacks themselves!



# Cross Stacks vs. Nested Stacks

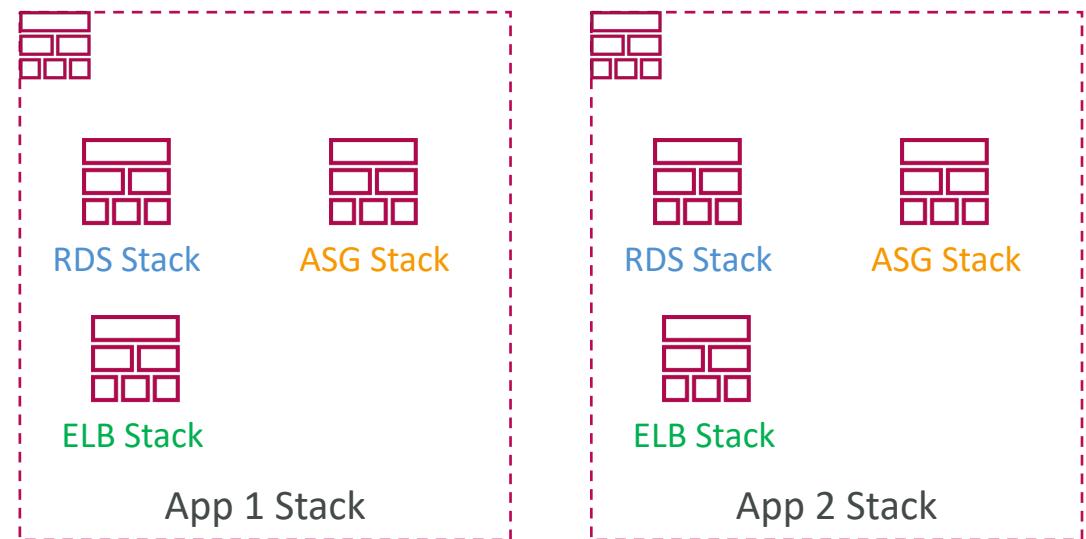
- **Cross Stacks**

- Helpful when stacks have different lifecycles
- Use Outputs Export and Fn::ImportValue
- When you need to pass export values to many stacks (VPC Id...)



- **Nested Stacks**

- Helpful when components must be re-used
- Example: re-use how to properly configure an Application Load Balancer
- The nested stack only is important to the higher-level stack (it's not shared)



# CloudFormation – DependsOn

- Specify that the creation of a specific resource follows another
- When added to a resource, that resource is created only after the creation of the resource specified in the **DependsOn** attribute
- Applied automatically when using **!Ref** and **!GetAtt**
- Use with any resource

**Resources:**

**EC2Instance:**

Type: AWS::EC2::Instance

DependsOn: DBInstance

**Properties:**

ImageId: ami-0a3c3a20c09d6f377

**DBInstance:**

Type: AWS::RDS::DBInstance

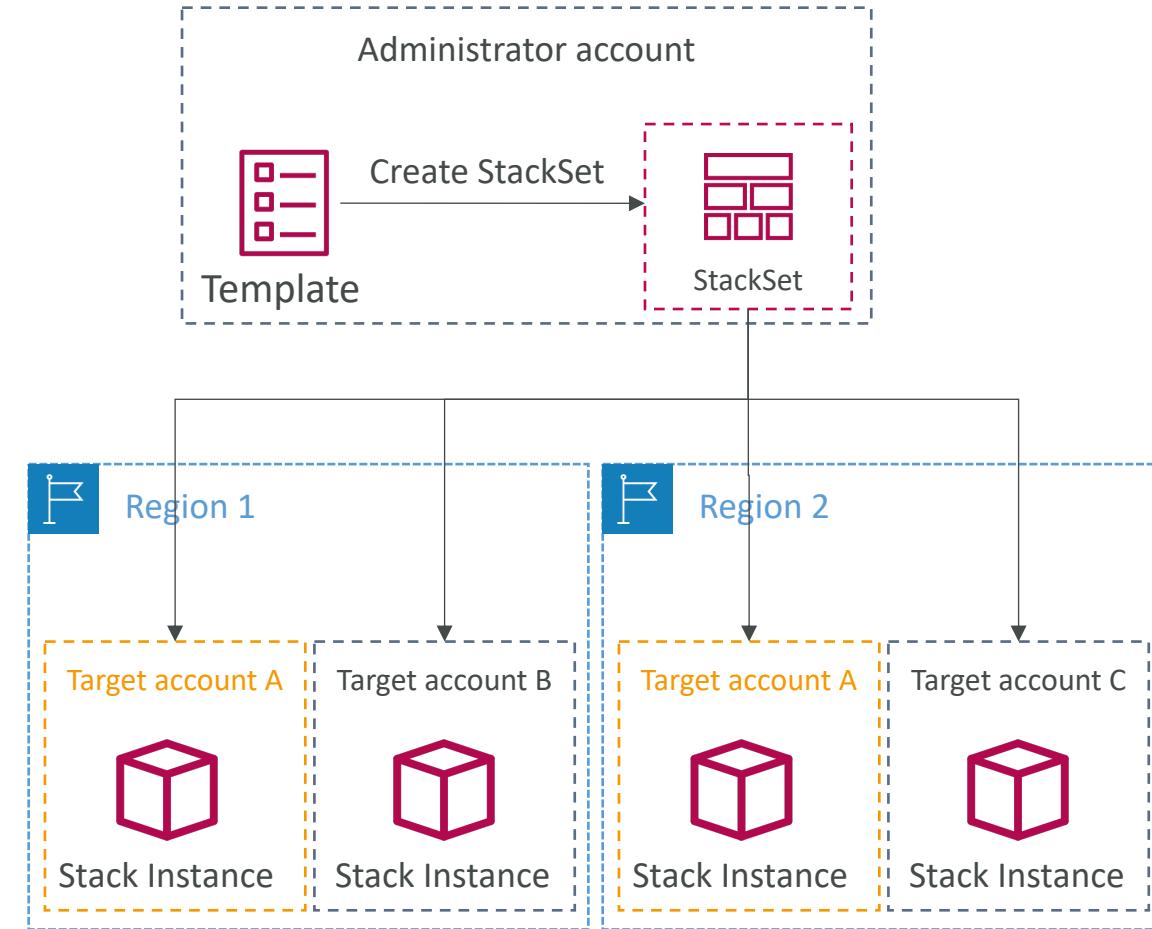
**Properties:**

Engine: MySQL

DBInstanceClass: db.t2.small

# CloudFormation – StackSets

- Create, update, or delete stacks across **multiple accounts and regions** with a single operation/template
- Administrator account to create StackSets
- Target accounts to create, update, delete stack instances from StackSets
- When you update a stack set, *all* associated stack instances are updated throughout all accounts and regions
- Can be applied into all accounts of an AWS organizations



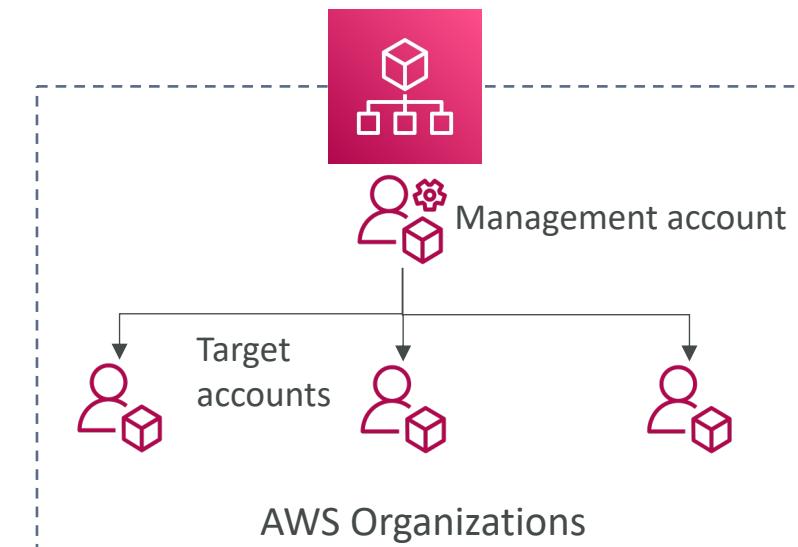
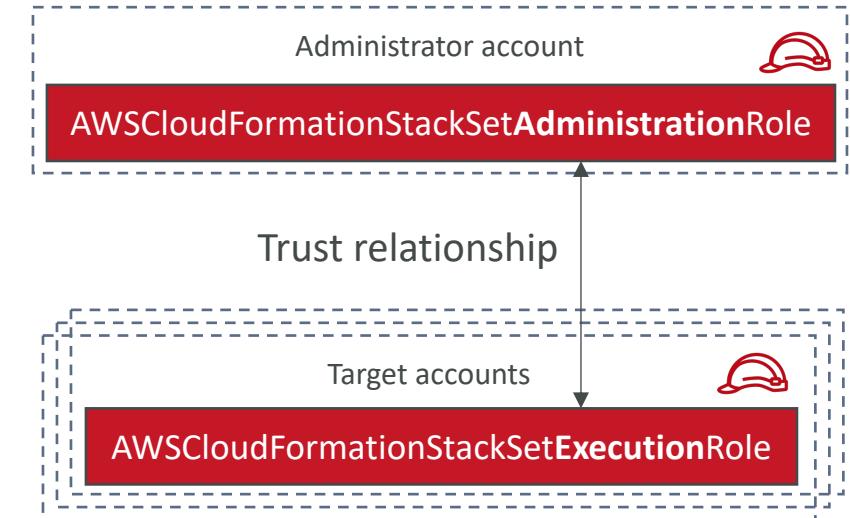
# CloudFormation – StackSets Permission Models

- **Self-managed Permissions**

- Create the IAM roles (with established trusted relationship) in both administrator and target accounts
- Deploy to any target account in which you have permissions to create IAM role

- **Service-managed Permissions**

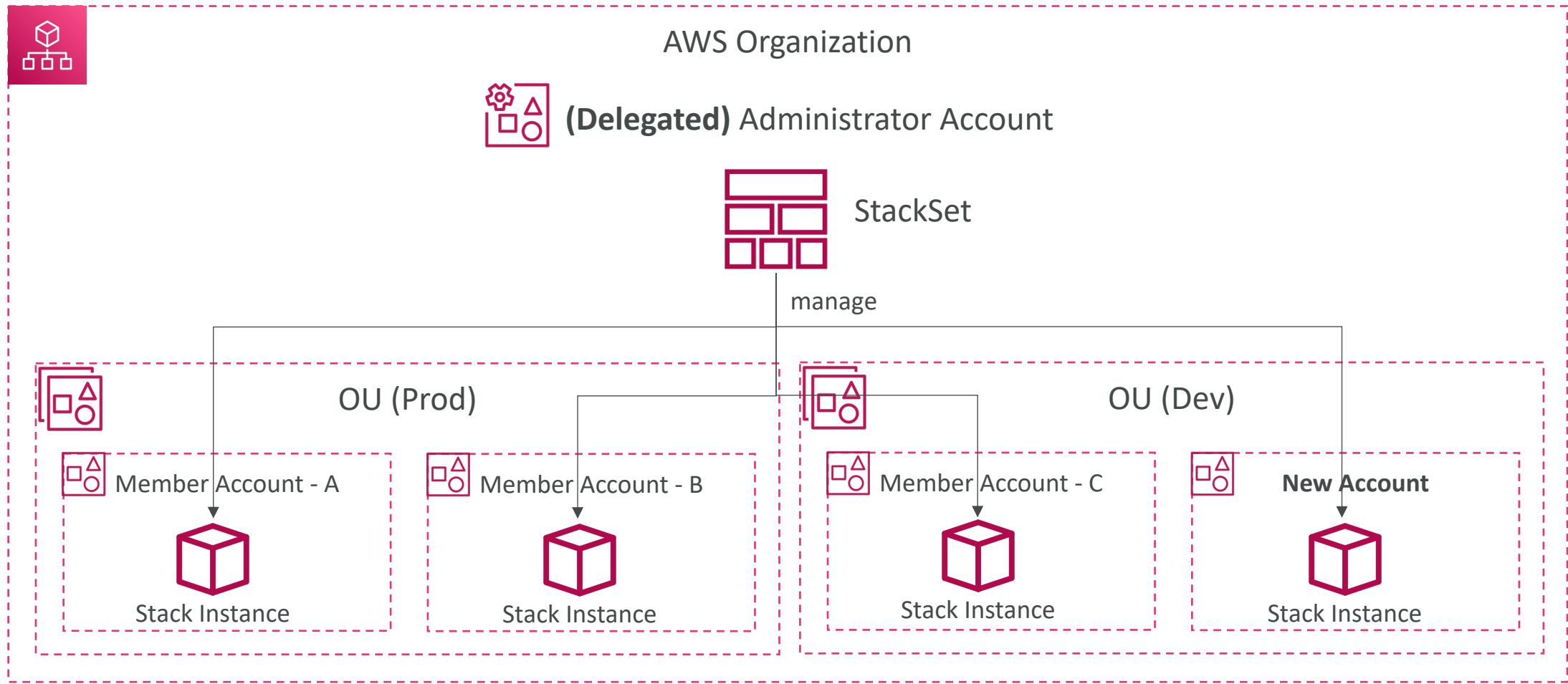
- Deploy to accounts managed by AWS Organizations
- StackSets create the IAM roles on your behalf (**enable trusted access with AWS Organizations**)
- Must **enable all features** in AWS Organizations
- Ability to deploy to accounts added to your organization in the future (Automatic Deployments)



# StackSets with AWS Organizations

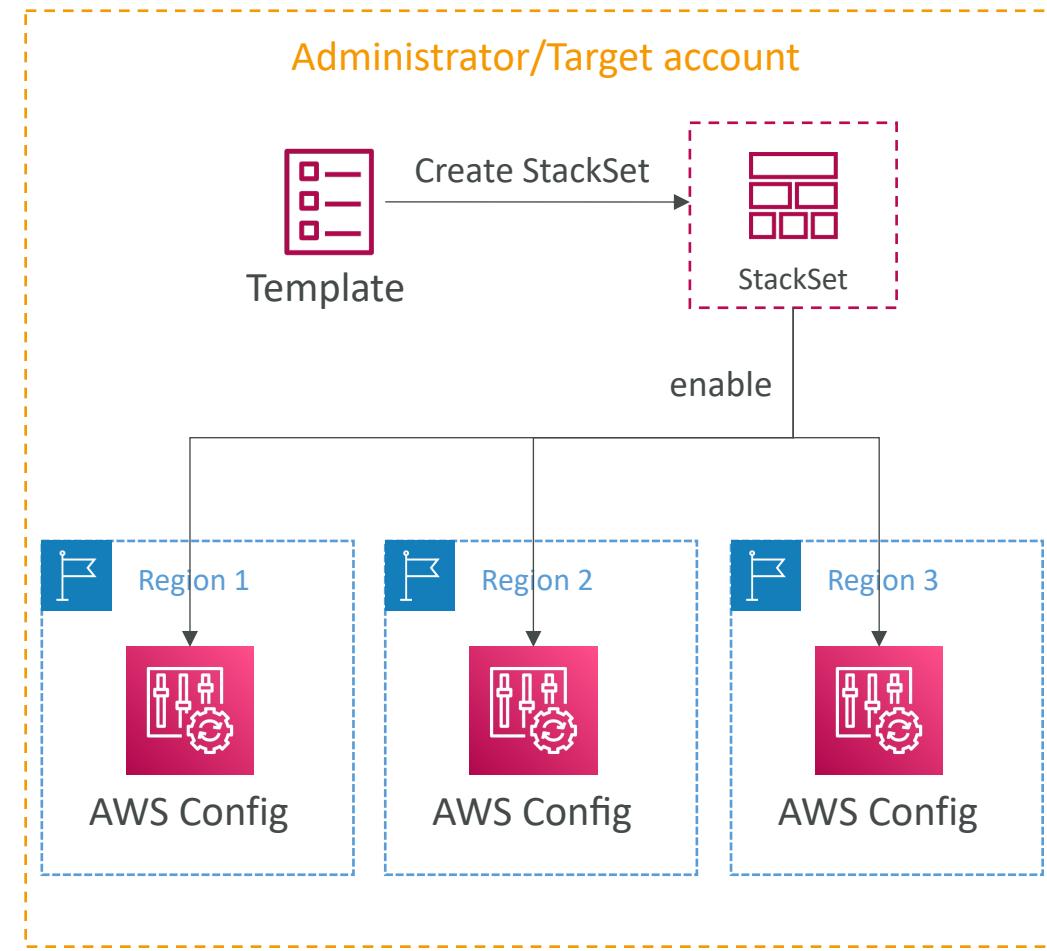
- Ability to automatically deploy Stack instances to new Accounts in an Organization
- Can delegate StackSets administration to member accounts in AWS Organization
- Trusted access with AWS Organizations must be enabled before delegated administrators can deploy to accounts managed by Organizations

# StackSets with AWS Organizations



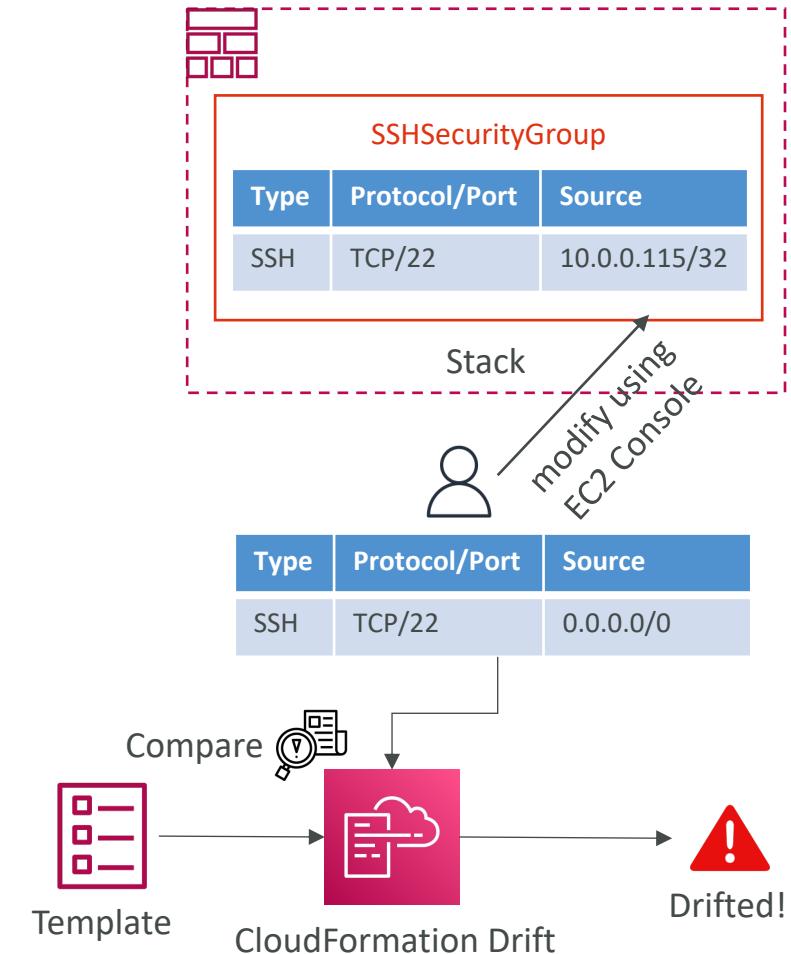
# Hands-On: StackSets

- We'll use StackSets to enable AWS Config across AWS regions with a single click
- Let's see how this works!



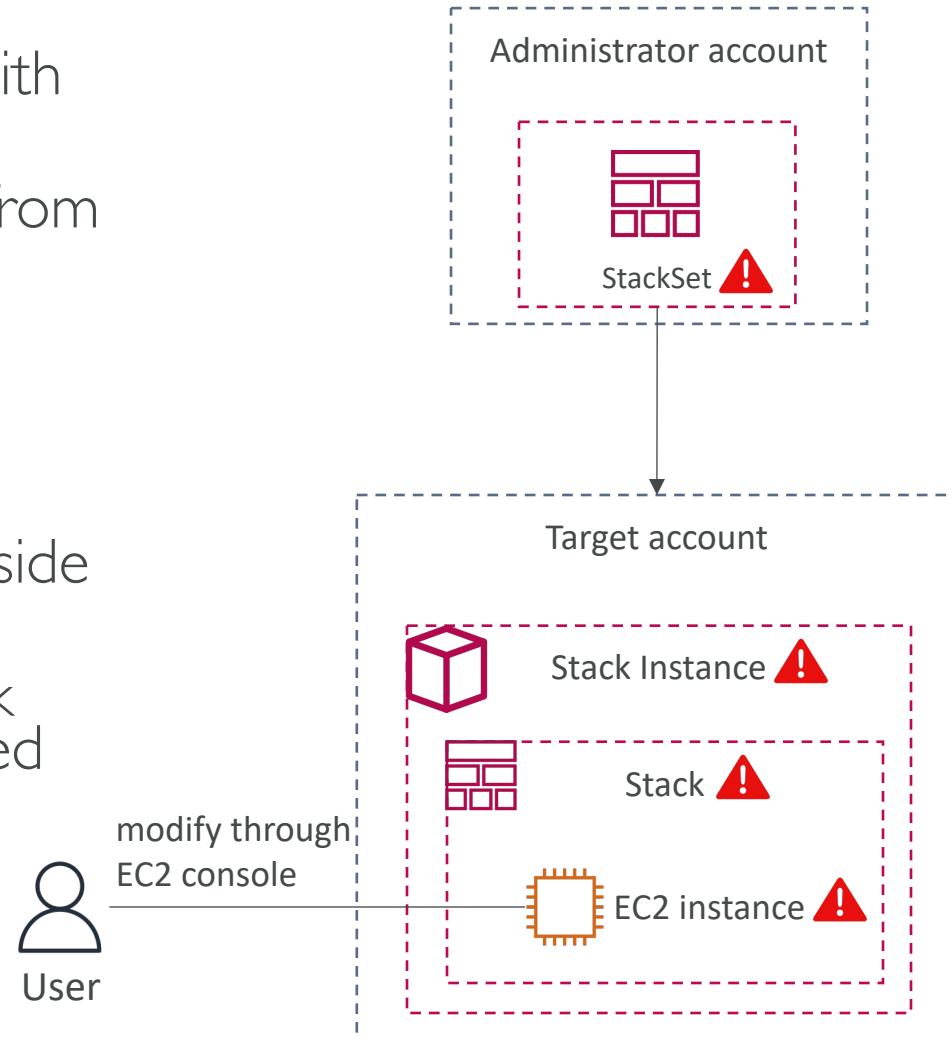
# CloudFormation – Drift

- CloudFormation allows you to create infrastructure
  - But it doesn't protect you against manual configuration changes
  - How do we know if our resources have **drifted**?
- 
- We can use CloudFormation Drift!
  - Detect drift on an entire stack or on individual resources within a stack



# StackSet Drift Detection

- Performs drift detection on the stack associated with each stack instance in the StackSet
- If the current state of a resource in a stack varies from the expected state:
  - The stack considered drifted
  - And the stack instance that the stack associated with considered drifted
  - And the StackSet is considered drifted
- Drift detection identifies unmanaged changes (outside CloudFormation)
- Changes made through CloudFormation to a stack directly (not at the StackSet level), aren't considered drifted
- You can stop drift detection on a StackSet



# CloudFormation – Troubleshooting

- DELETE\_FAILED
  - Some resources must be emptied before deleting, such as S3 buckets
  - Use Custom Resources with Lambda functions to automate some actions
  - Security Groups cannot be deleted until all EC2 instances in the group are gone
  - Think about using `DeletionPolicy=Retain` to skip deletions
- UPDATE\_ROLLBACK\_FAILED
  - Can be caused by resources changed outside of CloudFormation, insufficient permissions, Auto Scaling Group that doesn't receive enough signals...
  - Manually fix the error and then `ContinueUpdateRollback`

# CloudFormation – StackSet Troubleshooting

- A stack operation failed, and the stack instance status is **OUTDATED**.
  - Insufficient permissions in a target account for creating resources that are specified in your template.
  - The template could be trying to create global resources that must be unique but aren't, such as S3 buckets
  - The administrator account does not have a trust relationship with the target account
  - Reached a limit or a quota in the target account (too many resources)

# CloudFormation – Troubleshooting

- You can't set EC2 instance private DNS name in a CloudFormation template (*there's no PrivateDnsName attribute*)
  - Or anything else you wouldn't be able to set through the AWS Console
- If your CloudFormation template works in one Region and doesn't work in another Region:
  - Check if AWS services in the template are available in this Region
  - Check AMI IDs (AMIs are regional)
  - Check hardcoded region-specific values (e.g., ARNs)
  - Check unique names for resources that require it (e.g., S3 bucket names are globally unique)

# CloudFormation – Stack Failures

- You can specify the rollback behavior when creating a stack fails:
- `OnFailure=ROLLBACK` (default) – delete resources that were created so far
- `OnFailure=DO NOTHING` – keep all created resources, stack becomes in `CREATE FAILED` status
- `OnFailure=DELETE` – delete all resources and the stack

```
aws cloudformation create-stack --stack-name MyTestStack \  
--template-body file://template.yaml --on-failure DO NOTHING
```

# CloudFormation – Stack-Level Tags

- Tags associated with a CloudFormation Stack
- Automatically propagated to the supported resources in the Stack

## Using CloudFormation Console

### Configure stack options

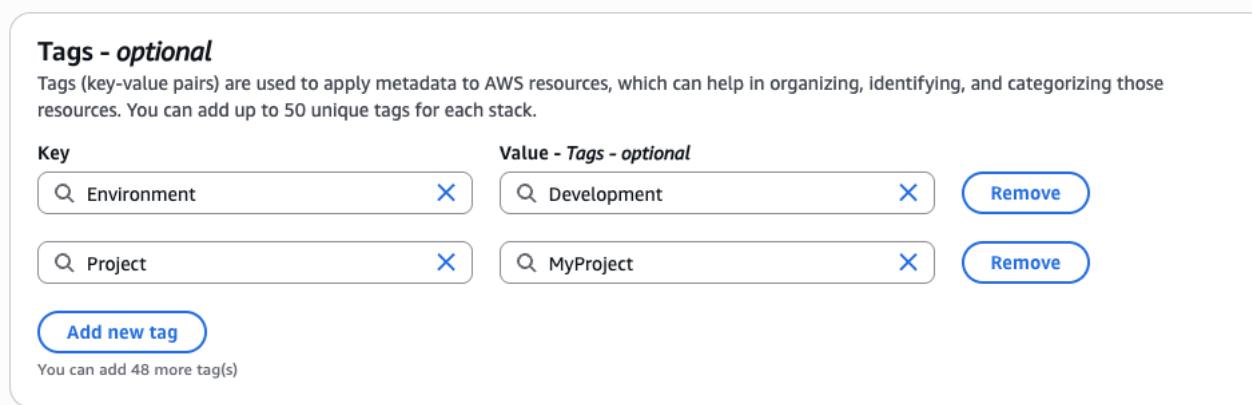
**Tags - optional**

Tags (key-value pairs) are used to apply metadata to AWS resources, which can help in organizing, identifying, and categorizing those resources. You can add up to 50 unique tags for each stack.

Key	Value - Tags - optional	Remove
<input type="text" value="Environment"/>	<input type="text" value="Development"/>	<button>Remove</button>
<input type="text" value="Project"/>	<input type="text" value="MyProject"/>	<button>Remove</button>

**Add new tag**

You can add 48 more tag(s)



## Using AWS CLI

```
aws cloudformation create-stack \
--stack-name DemoStack \
--template-body file://demo-template.yaml \
--tags Key=Environment,Value=Development \
Key=Project,Value=MyProject
```

# AutoScalingRollingUpdate

- Specify whether CloudFormation updates instances that are in an ASG in batches or all at once

```
UpdatePolicy:  
  AutoScalingRollingUpdate:  
    MaxBatchSize: Integer  
    MinInstancesInService: Integer  
    MinSuccessfulInstancesPercent: Integer  
    PauseTime: String  
    SuspendedProcesses: [ List of processes ]  
    WaitOnResourceSignals: Boolean
```

```
Resources:  
  ASG:  
    Type: AWS::AutoScaling::AutoScalingGroup  
    Properties:  
      ...  
      MinSize: 1  
      MaxSize: 4  
      DesiredCapacity: 1  
    UpdatePolicy:  
      AutoScalingRollingUpdate:  
        MaxBatchSize: 2  
        MinInstancesInService: 1  
        PauseTime: PT10M  
        WaitOnResourceSignals: true
```

# AutoScalingReplacingUpdate

- Create a new ASG as a replacement to the old one
- Make sure you have enough EC2 capacity for two ASGs
- If update fails, the new ASG is deleted and the old ASG is kept intact

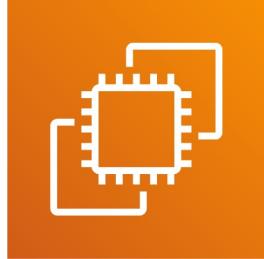
```
UpdatePolicy:  
  AutoScalingReplacingUpdate:  
    WillReplace: Boolean
```

```
Resources:  
  ASG:  
    Type: AWS::AutoScaling::AutoScalingGroup  
    Properties:  
      ...  
      UpdatePolicy:  
        AutoScalingReplacingUpdate:  
          WillReplace: true  
      CreationPolicy:  
        ResourceSignal:  
          Count: !Ref ResourceSignalsOnCreate  
          Timeout: PT10M  
        AutoScalingCreationPolicy:  
          MinSuccessfulInstancesPercent: !Ref MinSuccessfulPercentParameter
```

# AWS Lambda

It's a Serverless world

# 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

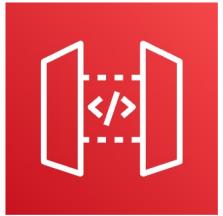
- Easy Pricing:
  - Pay per request and compute time
  - Free tier of 1,000,000 AWS Lambda requests and 400,000 GBs of compute time
- Integrated with the whole AWS suite of services
- Integrated with many programming languages
- Easy monitoring through AWS CloudWatch
- Easy to get more resources per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

# AWS Lambda language support

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

# AWS Lambda Integrations

## Main ones



API Gateway



Kinesis



DynamoDB



S3



CloudFront



CloudWatch Events  
EventBridge



CloudWatch Logs



SNS

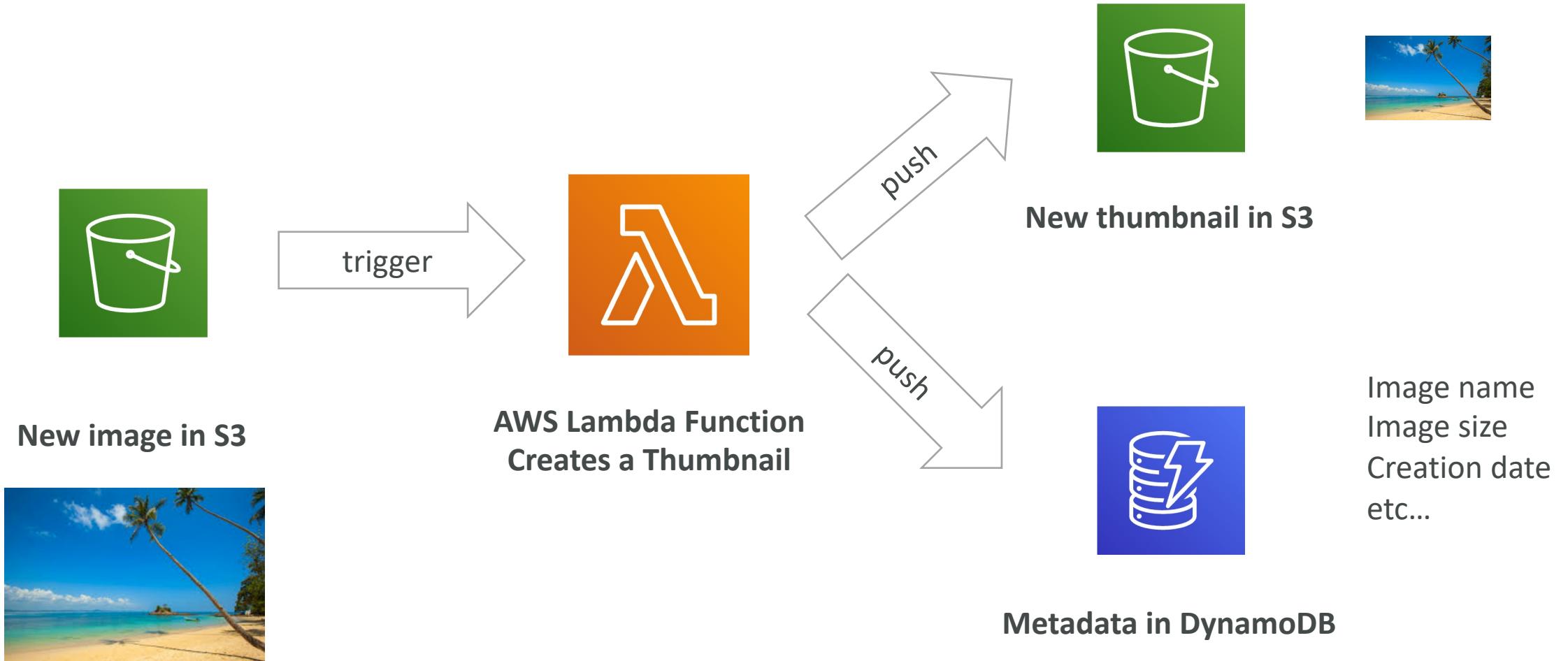


SQS

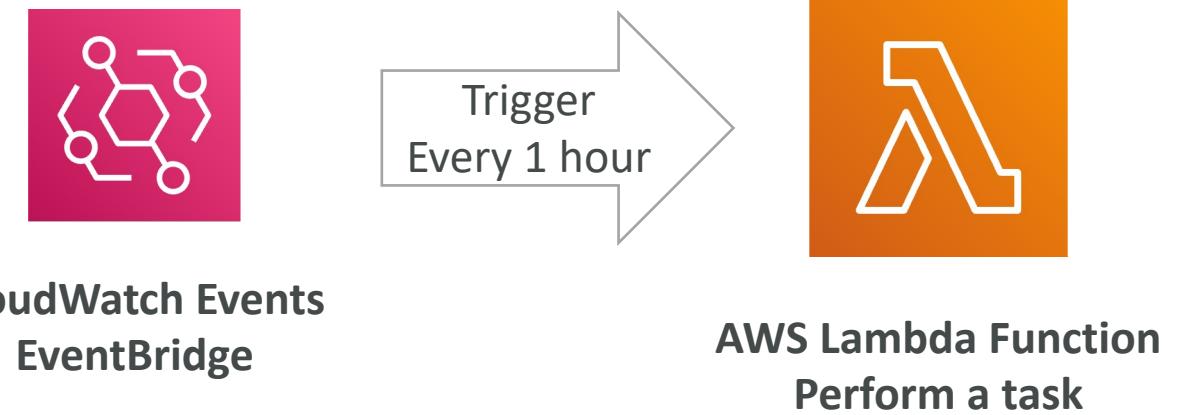


Cognito

# Example: Serverless Thumbnail creation



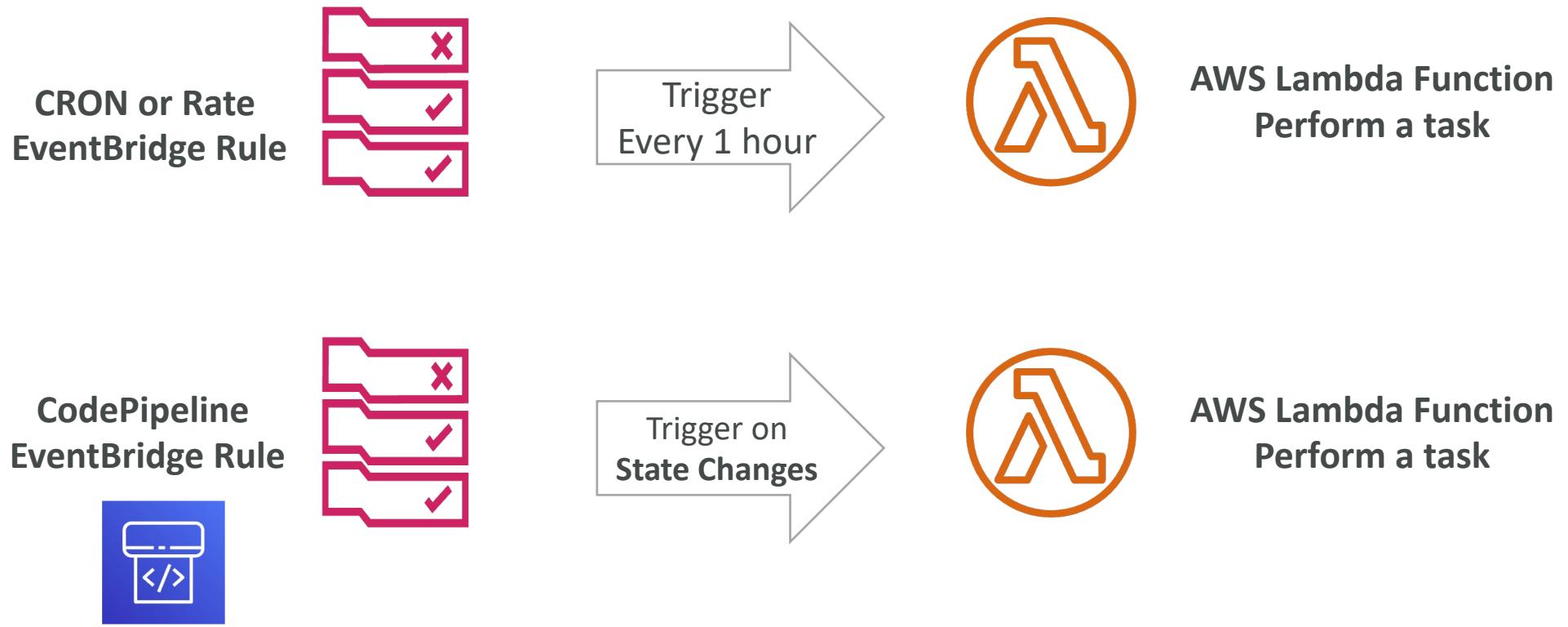
# Example: Serverless CRON Job



# AWS Lambda Pricing: example

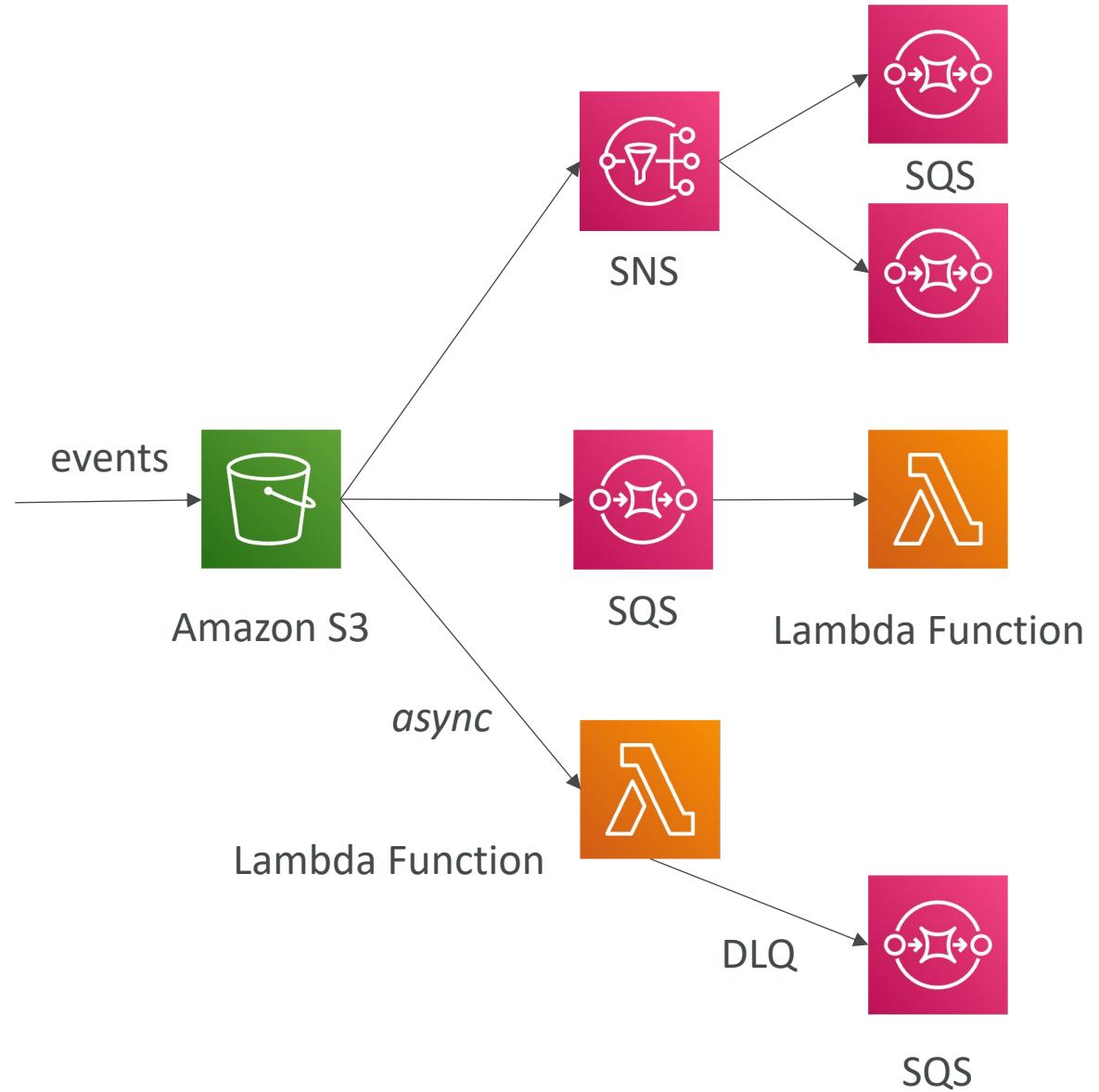
- You can find overall pricing information here:  
<https://aws.amazon.com/lambda/pricing/>
- Pay per calls:
  - First 1,000,000 requests are free
  - \$0.20 per 1 million requests thereafter (\$0.0000002 per request)
- Pay per duration: (in increment of 1 ms)
  - 400,000 GB-seconds of compute time per 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

# CloudWatch Events / EventBridge

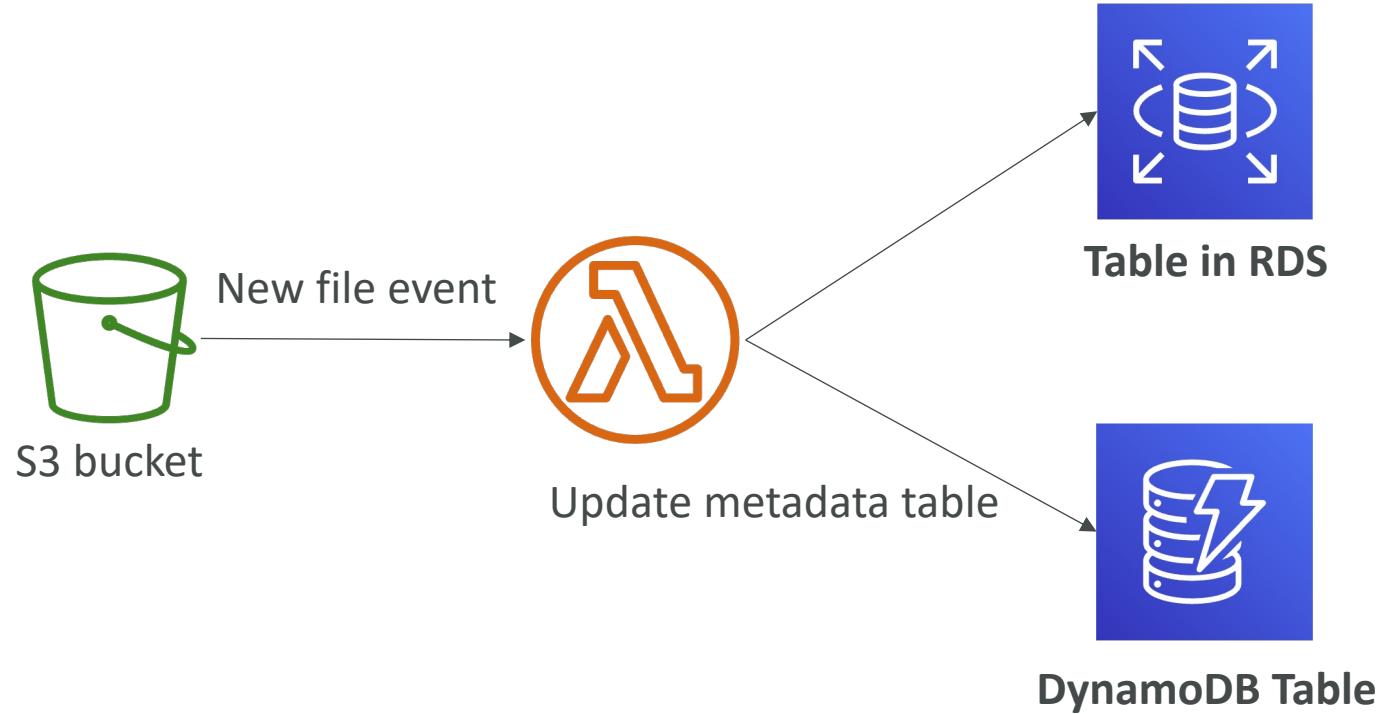


# S3 Events Notifications

- S3:ObjectCreated, S3:ObjectRemoved, S3:ObjectRestore, S3:Replication...
- Object name filtering possible (\*.jpg)
- Use case: generate thumbnails of images uploaded to S3
- S3 event notifications typically deliver events in seconds but can sometimes take a minute or longer
- If two writes are made to a single non-versioned object at the same time, it is possible that only a single event notification will be sent
- If you want to ensure that an event notification is sent for every successful write, you can enable versioning on your bucket.



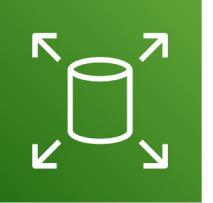
# Simple S3 Event Pattern – Metadata Sync



# Amazon EC2 Storage & Data Management

EBS, Instance Store & EFS

# What's an EBS Volume?

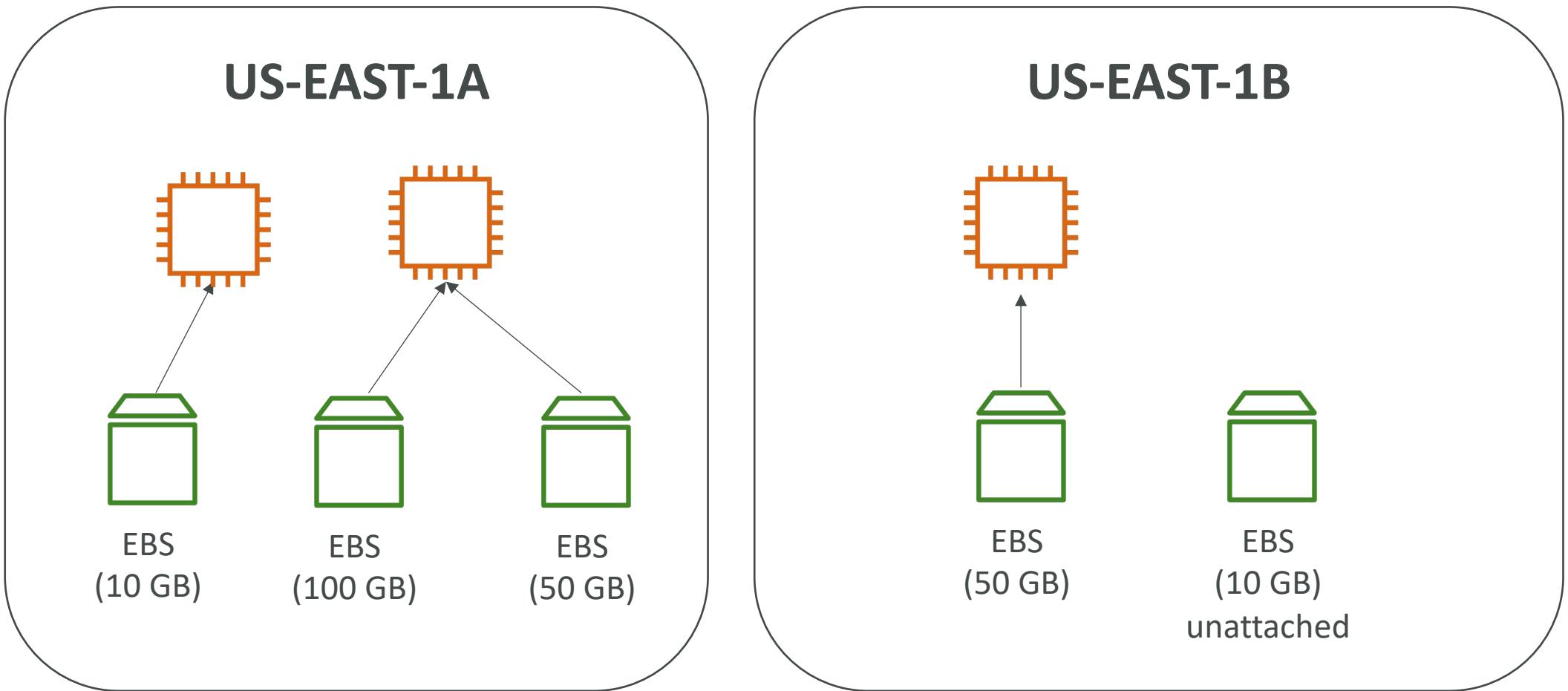


- An **EBS (Elastic Block Store) Volume** is a **network** drive you can attach to your instances while they run
- It allows your instances to persist data, even after their termination
- They can only be mounted to one instance at a time (at the CCP level)
- They are bound to a specific availability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

# 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 Volume Types

- EBS Volumes come in 6 types
  - **gp2 / gp3 (SSD)**: General purpose SSD volume that balances price and performance for a wide variety of workloads
  - **io1 / io2 Block Express (SSD)**: Highest-performance SSD volume for mission-critical low-latency or high-throughput workloads
  - **st1 (HDD)**: Low cost HDD volume designed for frequently accessed, throughput-intensive workloads
  - **sc1 (HDD)**: Lowest cost HDD volume designed for less frequently accessed workloads
- EBS Volumes are characterized in Size | Throughput | IOPS (I/O Ops Per Sec)
- When in doubt always consult the AWS documentation – it's good!
- Only gp2/gp3 and io1/io2 Block Express can be used as boot volumes

# EBS Volume Types Use cases

## General Purpose SSD

- Cost effective storage, low-latency
- System boot volumes, Virtual desktops, Development and test environments
- 1 GiB - 16 TiB
- gp3:
  - Baseline of 3,000 IOPS and throughput of 125 MiB/s
  - Can increase IOPS up to 16,000 and throughput up to 1000 MiB/s independently
- gp2:
  - Small gp2 volumes can burst IOPS to 3,000
  - Size of the volume and IOPS are linked, max IOPS is 16,000
  - 3 IOPS per GiB, means at 5,334 GiB we are at the max IOPS

# EBS Volume Types Use cases

## Provisioned IOPS (PIOPS) SSD

- Critical business applications with sustained IOPS performance
- Or applications that need more than 16,000 IOPS
- Great for **databases workloads** (sensitive to storage perf and consistency)
- io1 (4 GiB - 16 TiB):
  - Max PIOPS: 64,000 for Nitro EC2 instances & 32,000 for other
  - Can increase PIOPS independently from storage size
- io2 Block Express (4 GiB – 64 TiB):
  - Sub-millisecond latency
  - Max PIOPS: 256,000 with an IOPS:GiB ratio of 1,000:1
- Supports EBS Multi-attach

# EBS Volume Types Use cases

## Hard Disk Drives (HDD)

- Cannot be a boot volume
- 125 GiB to 16 TiB
- Throughput Optimized HDD (st1)
  - Big Data, Data Warehouses, Log Processing
  - Max throughput 500 MiB/s – max IOPS 500
- Cold HDD (sc1):
  - For data that is infrequently accessed
  - Scenarios where lowest cost is important
  - Max throughput 250 MiB/s – max IOPS 250

# EBS – Volume Types Summary

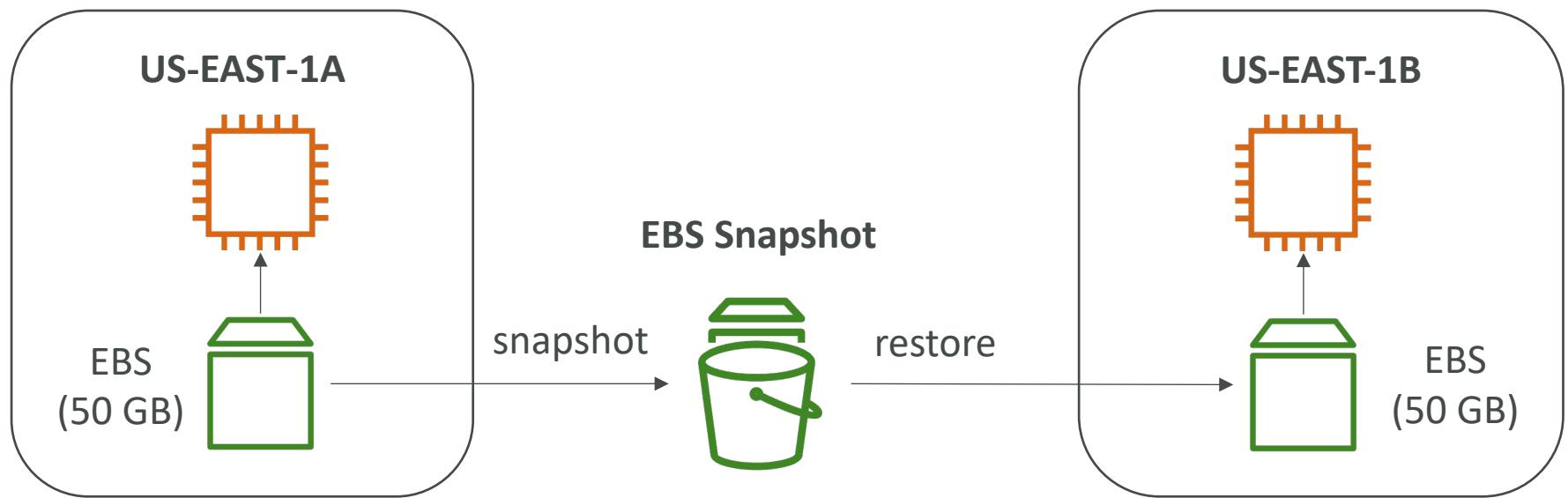
	General Purpose SSD volumes		Provisioned IOPS SSD volumes	
Volume type	gp3	gp2	io2 Block Express <sup>3</sup>	io1
Durability	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	99.999% durability (0.001% annual failure rate)	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)
Use cases	<ul style="list-style-type: none"> <li>Transactional workloads</li> <li>Virtual desktops</li> <li>Medium-sized, single-instance databases</li> <li>Low-latency interactive applications</li> <li>Boot volumes</li> <li>Development and test environments</li> </ul>	<p>Workloads that require:</p> <ul style="list-style-type: none"> <li>Sub-millisecond latency</li> <li>Sustained IOPS performance</li> <li>More than 64,000 IOPS or 1,000 MiB/s of throughput</li> </ul>	<ul style="list-style-type: none"> <li>Workloads that require sustained IOPS performance or more than 16,000 IOPS</li> <li>I/O-intensive database workloads</li> </ul>	
Volume size	1 GiB - 16 TiB	4 GiB - 64 TiB <sup>4</sup>	4 GiB - 16 TiB	
Max IOPS per volume (16 KiB I/O)	16,000	256,000 <sup>5</sup>	64,000	
Max throughput per volume	1,000 MiB/s	250 MiB/s <sup>1</sup>	4,000 MiB/s	1,000 MiB/s <sup>2</sup>
Amazon EBS Multi-attach	Not supported		Supported	
NVMe reservations	Not supported		Supported	Not supported
Boot volume	Supported			

	Throughput Optimized HDD volumes	Cold HDD volumes
Volume type	st1	sc1
Durability	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	
Use cases	<ul style="list-style-type: none"> <li>Big data</li> <li>Data warehouses</li> <li>Log processing</li> </ul>	<ul style="list-style-type: none"> <li>Throughput-oriented storage for data that is infrequently accessed</li> <li>Scenarios where the lowest storage cost is important</li> </ul>
Volume size	125 GiB - 16 TiB	
Max IOPS per volume (1 MiB I/O)	500	250
Max throughput per volume	500 MiB/s	250 MiB/s
Amazon EBS Multi-attach	Not supported	
Boot volume	Not supported	

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-volume-types.html#solid-state-drives>

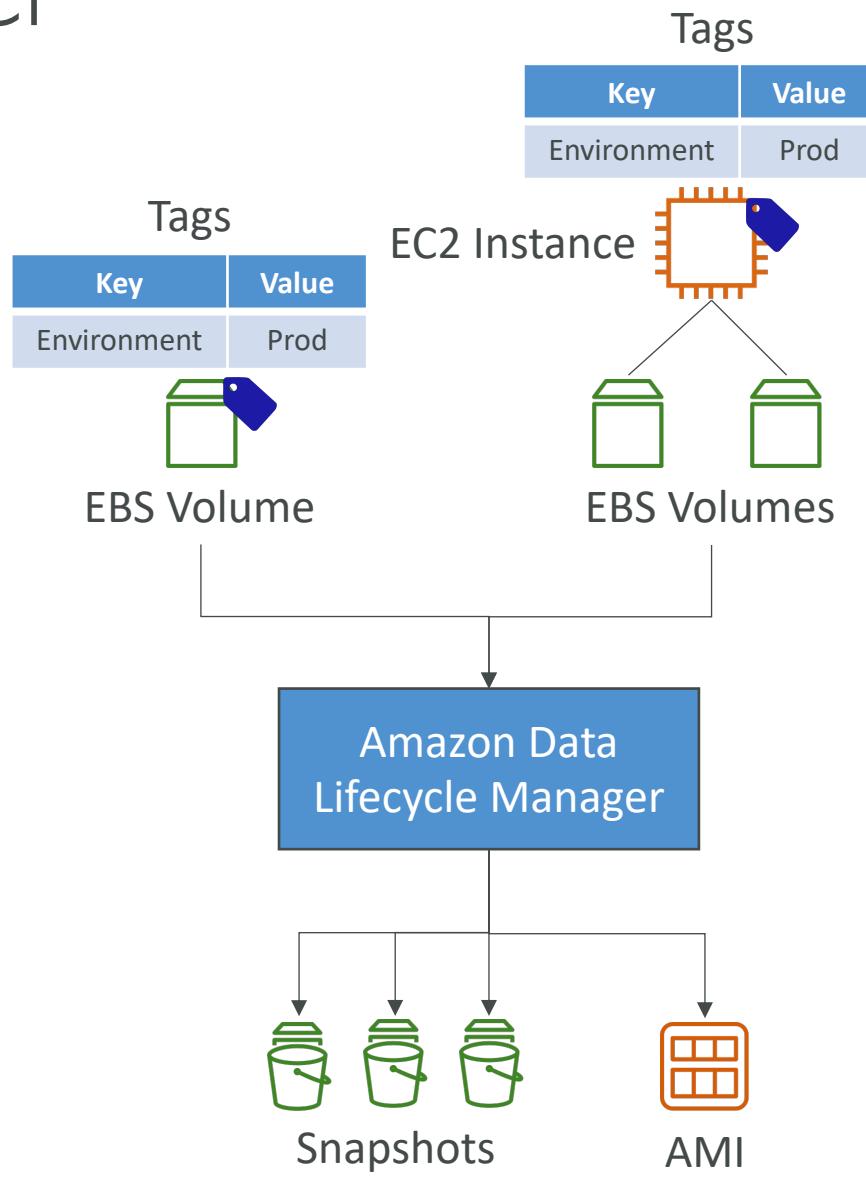
# EBS Snapshots

- 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



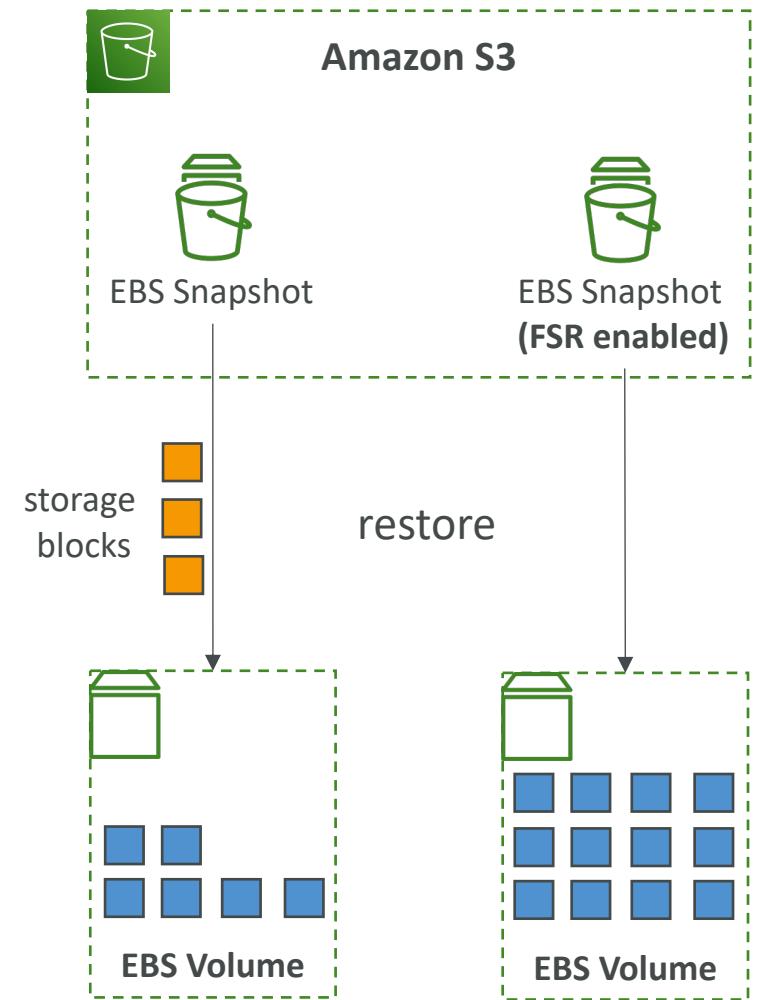
# Amazon Data Lifecycle Manager

- Automate the creation, retention, and deletion of EBS snapshots and EBS-backed AMIs
- Schedule backups, cross-account snapshot copies, delete outdated backups, ...
- Uses resource tags to identify the resources (EC2 instances, EBS volumes)
- Can't be used to manage snapshots/AMIs created outside DLM
- Can't be used to manage instance-store backed AMIs



# EBS Snapshots – Fast Snapshot Restore (FSR)

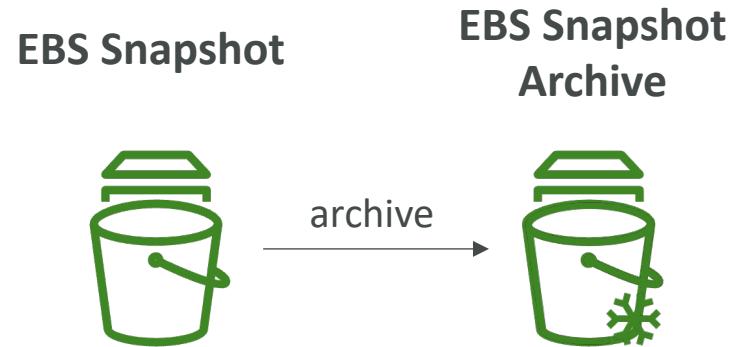
- EBS Snapshots stored in S3
- By default, there's a latency of I/O operations the first time each block is accessed (block must be pulled from S3)
- **Solution:** force the initialization of the entire volume (using the `dd` or `fio` command), or you can enable FSR
- FSR helps you to create a volume from a snapshot that is fully initialized at creation (no I/O latency)
- Enabled for a snapshot in a particular AZ (billed per minute – very expensive \$\$\$)
- Can be enabled on snapshots created by Data Lifecycle Manager



# EBS Snapshots Features

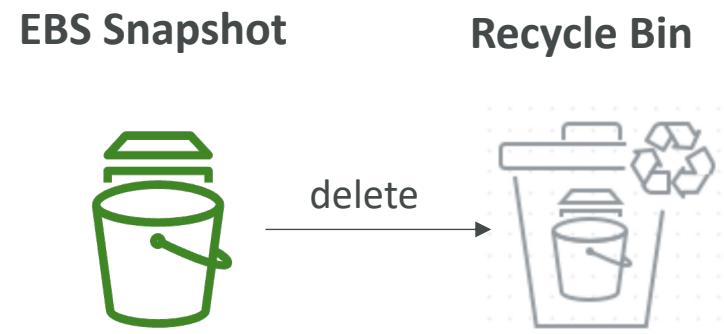
- **EBS Snapshot Archive**

- Move a Snapshot to an "archive tier" that is 75% cheaper
- Takes within 24 to 72 hours for restoring the archive



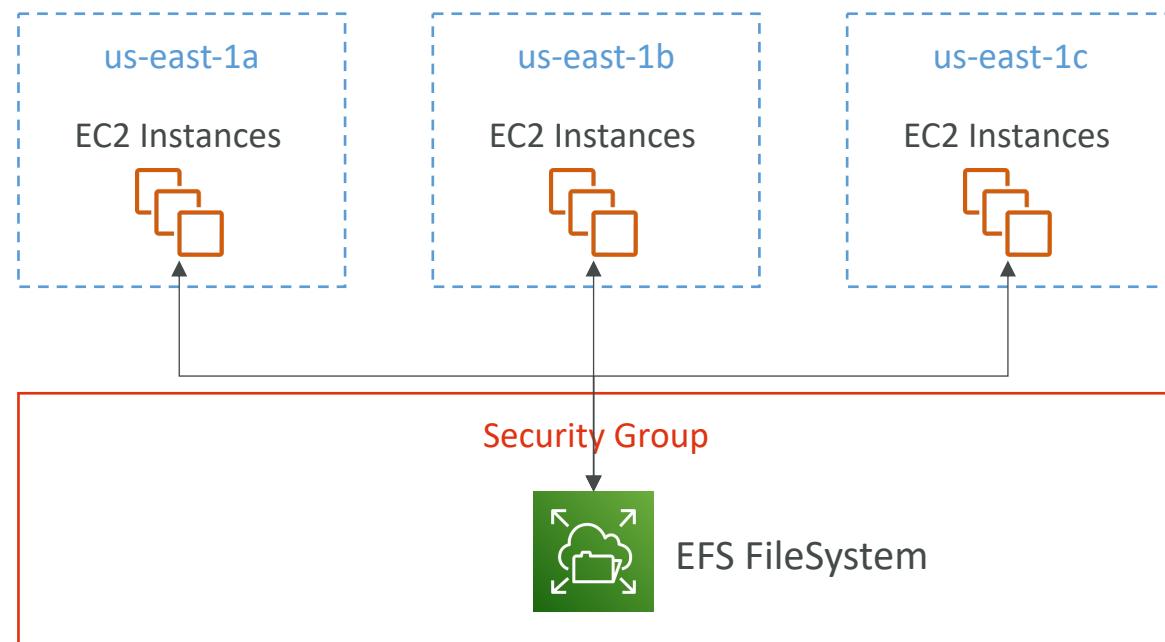
- Recycle Bin for EBS Snapshots

- Setup rules to retain deleted snapshots so you can recover them after an accidental deletion
- Specify retention (from 1 day to 1 year)



# Amazon EFS – Elastic File System

- Managed NFS (network file system) that can be mounted on many EC2 instances
- EFS works with EC2 instances in multi-AZ
- Highly available, scalable, expensive (3x gp2), pay per use



# Amazon EFS – Elastic File System

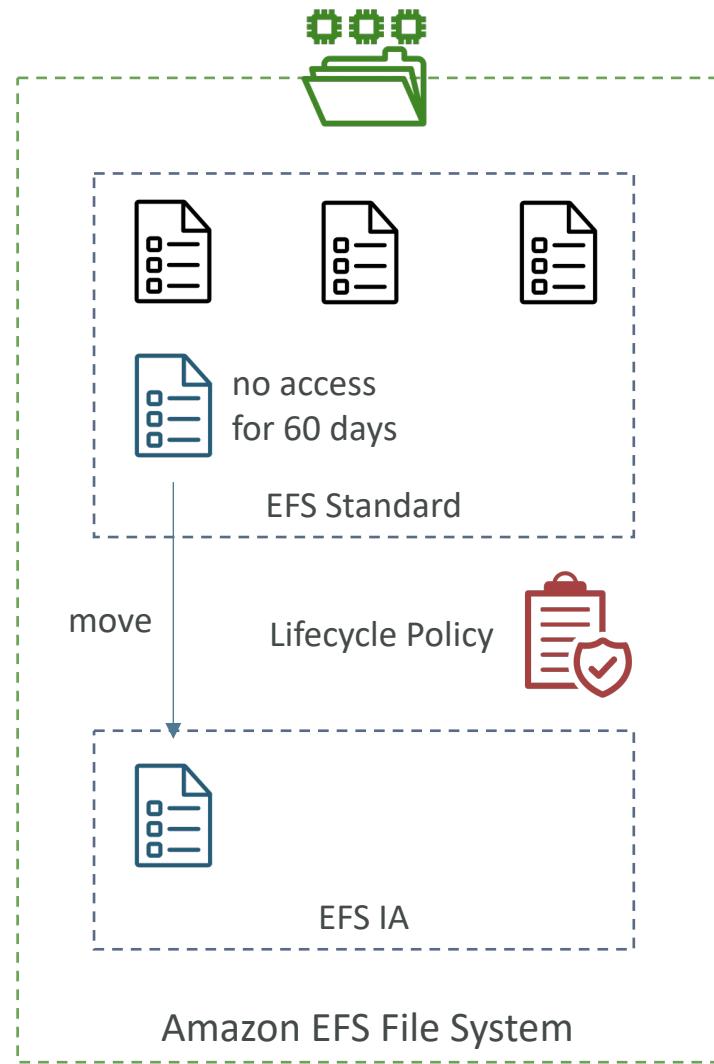
- Use cases: content management, web serving, data sharing, Wordpress
- Uses NFSv4.1 protocol
- Uses security group to control access to EFS
- **Compatible with Linux based AMI (not Windows)**
- Encryption at rest using KMS
  
- POSIX file system (~Linux) that has a standard file API
- File system scales automatically, pay-per-use, no capacity planning!

# EFS – Performance & Storage Classes

- EFS Scale
  - 1000s of concurrent NFS clients, 10 GB+ /s throughput
  - Grow to Petabyte-scale network file system, automatically
- Performance Mode (set at EFS creation time)
  - General Purpose (default) – latency-sensitive use cases (web server, CMS, etc...)
  - Max I/O – higher latency, throughput, highly parallel (big data, media processing)
- Throughput Mode
  - Bursting – 1 TB = 50MiB/s + burst of up to 100MiB/s
  - Provisioned – set your throughput regardless of storage size, ex: 1 GiB/s for 1 TB storage
  - Elastic – automatically scales throughput up or down based on your workloads
    - Up to 3GiB/s for reads and 1GiB/s for writes
    - Used for unpredictable workloads

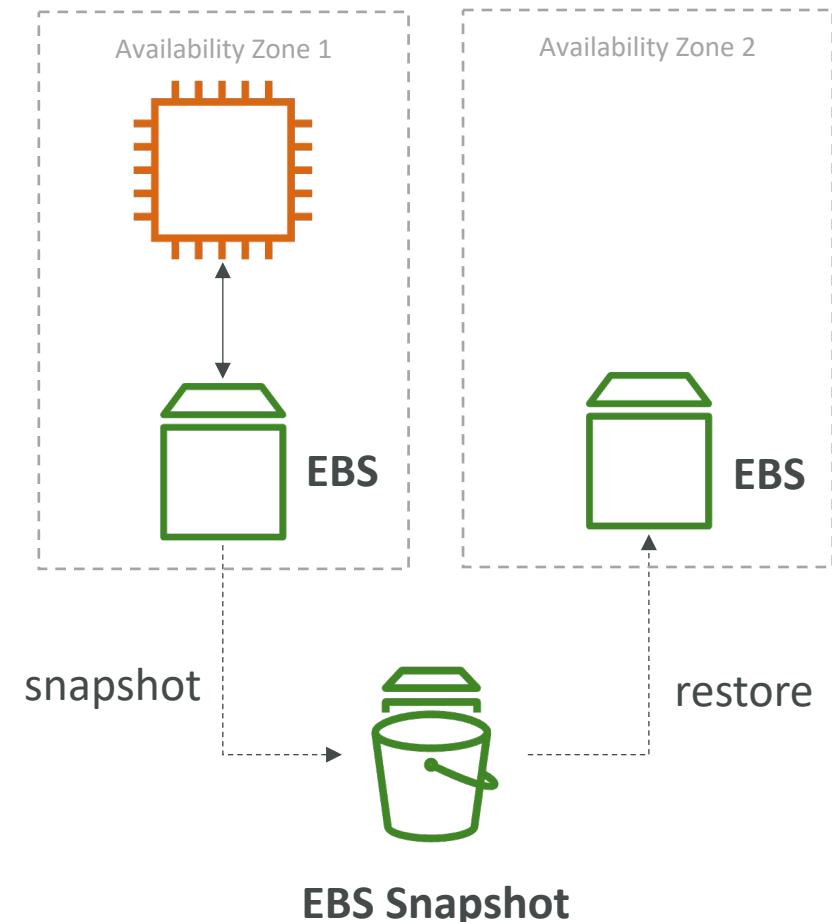
# EFS – Storage Classes

- Storage Tiers (lifecycle management feature – move file after N days)
  - Standard: for frequently accessed files
  - Infrequent access (EFS-IA): cost to retrieve files, lower price to store.
  - Archive: rarely accessed data (few times each year), 50% cheaper
  - Implement lifecycle policies to move files between storage tiers
- Availability and durability
  - Standard: Multi-AZ, great for prod
  - One Zone: One AZ, great for dev, backup enabled by default, compatible with IA (EFS One Zone-IA)
- Over 90% in cost savings



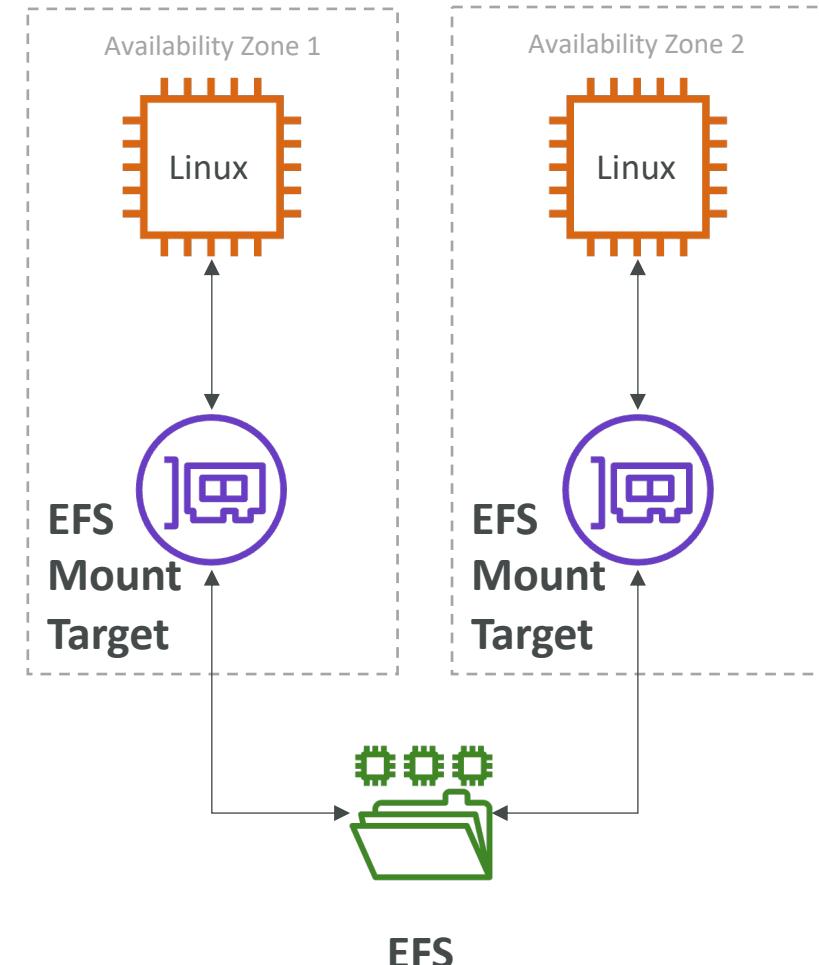
# EBS vs EFS – Elastic Block Storage

- EBS volumes...
  - one instance (except multi-attach io1/io2)
  - are locked at the Availability Zone (AZ) level
  - gp2: IO increases if the disk size increases
  - gp3 & io1: can increase IO independently
- To migrate an EBS volume across AZ
  - Take a snapshot
  - Restore the snapshot to another AZ
  - EBS backups use IO and you shouldn't run them while your application is handling a lot of traffic
- Root EBS Volumes of instances get terminated by default if the EC2 instance gets terminated. (you can disable that)



# EBS vs EFS – Elastic File System

- Mounting 100s of instances across AZ
  - EFS share website files (WordPress)
  - Only for Linux Instances (POSIX)
- 
- EFS has a higher price point than EBS
  - Can leverage Storage Tiers for cost savings
- 
- Remember: EFS vs EBS vs Instance Store



# Amazon S3



# 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

# Amazon 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 - 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
  - Must NOT start with the prefix `xn--`
  - Must NOT end with the suffix `-s3alias`



S3 Bucket

# Amazon S3 - 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 – Objects (cont.)

- 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)

# Amazon S3 – Security

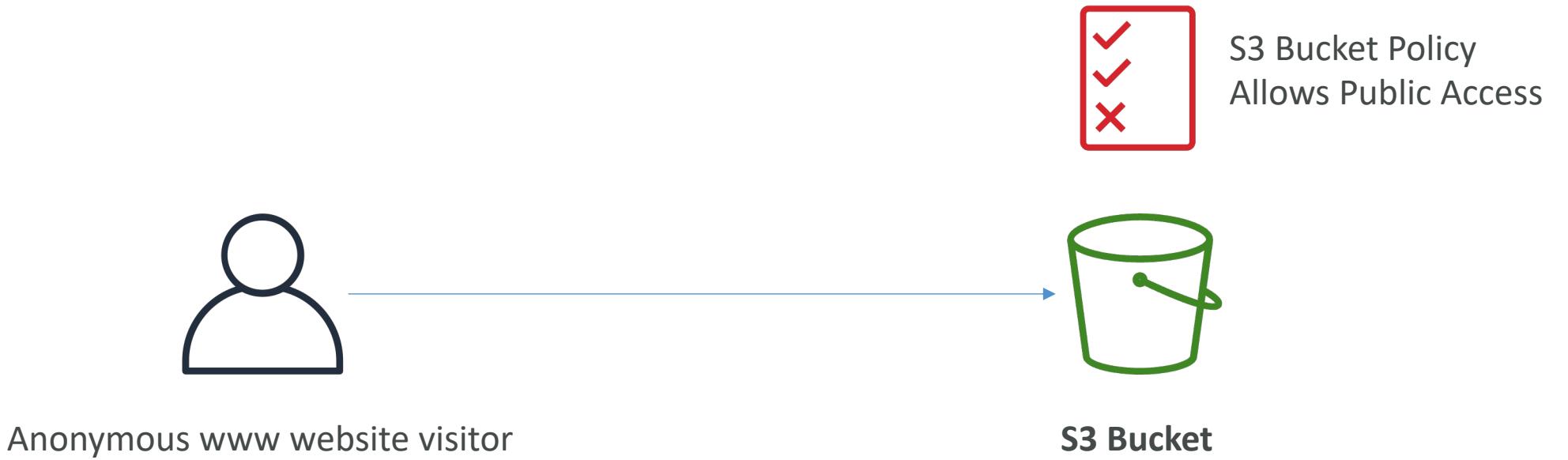
- User-Based
  - IAM Policies – which API calls should be allowed for a specific user from IAM
- Resource-Based
  - Bucket Policies – bucket wide rules from the S3 console - allows cross account
  - Object Access Control List (ACL) – finer grain (can be disabled)
  - Bucket Access Control List (ACL) – less common (can be disabled)
- **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

# S3 Bucket Policies

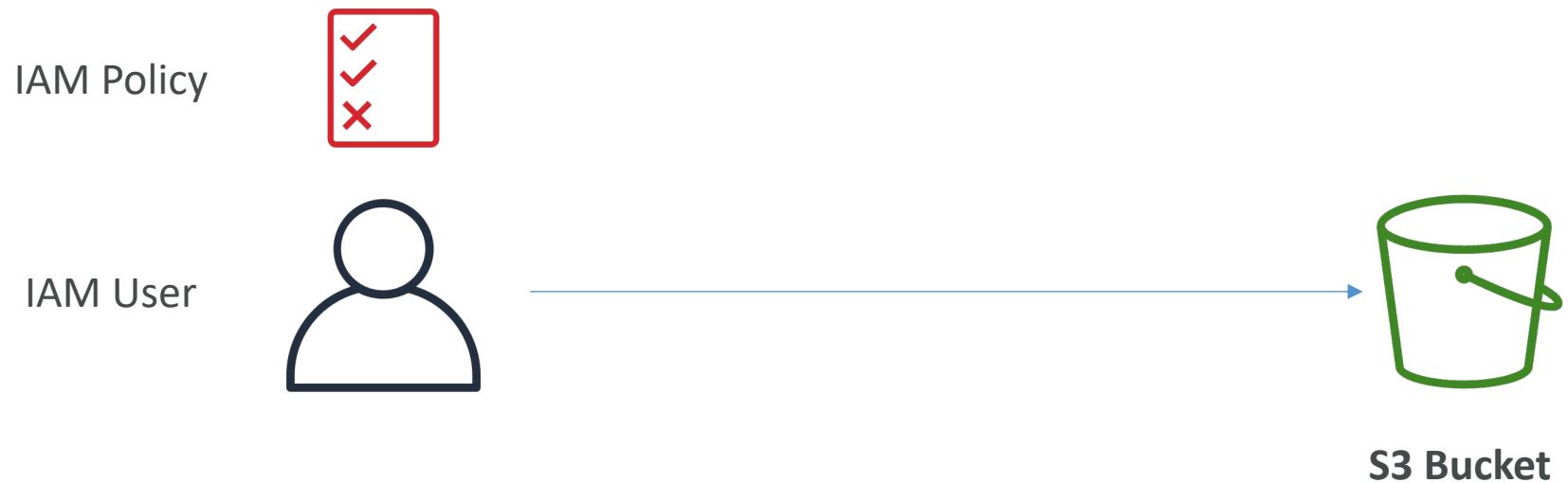
- JSON based policies
  - Resources: buckets and objects
  - Effect: Allow / Deny
  - Actions: Set of API to Allow or Deny
  - Principal: The account or user to apply the policy to
- Use S3 bucket for policy to:
  - Grant public access to the bucket
  - Force objects to be encrypted at upload
  - Grant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

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

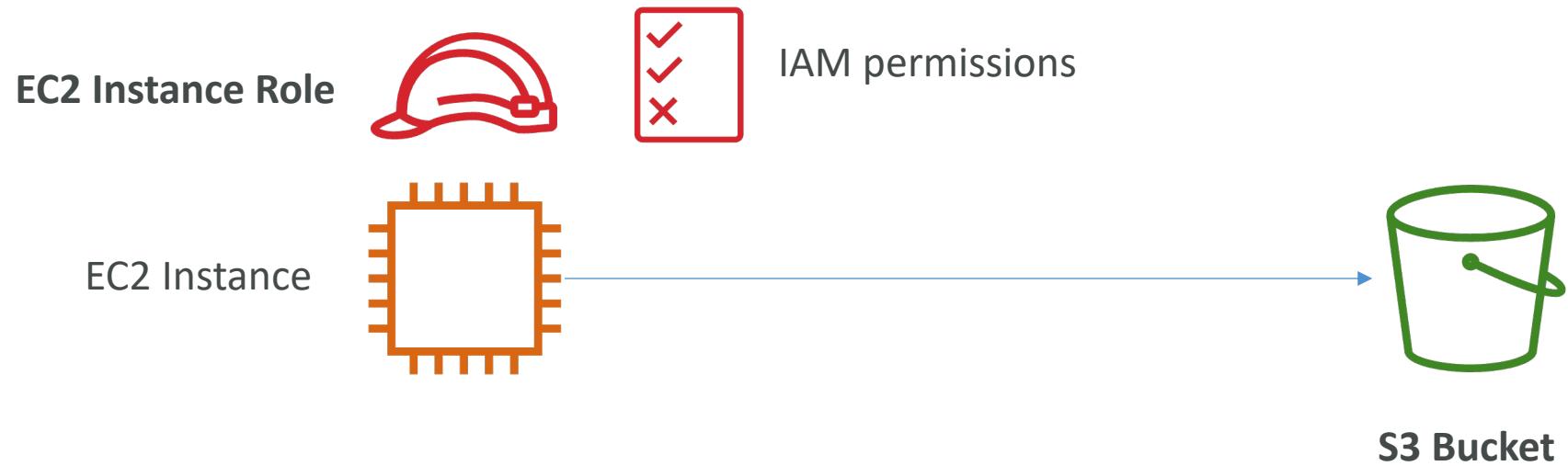
# Example: Public Access - Use Bucket Policy



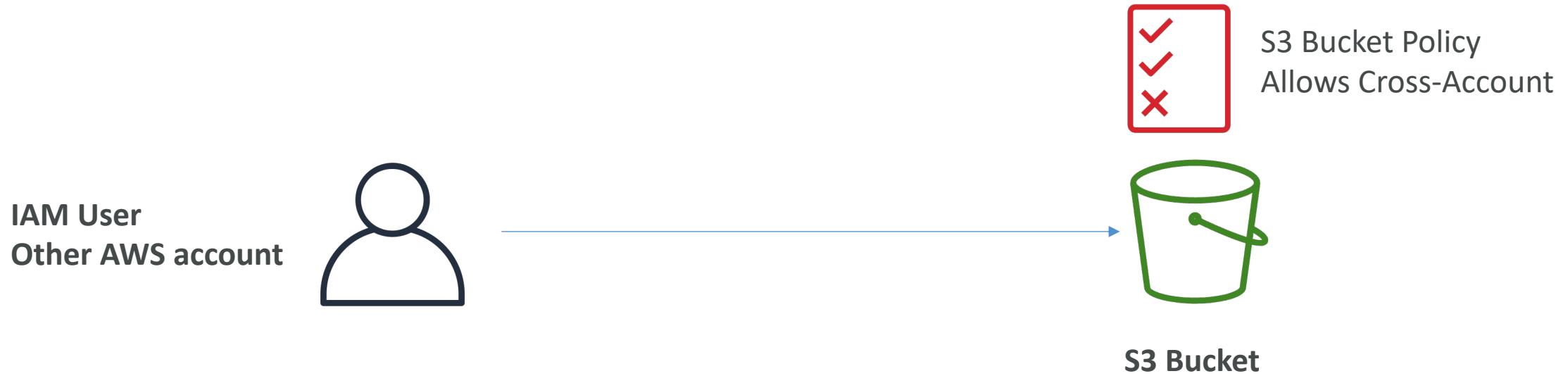
# Example: User Access to S3 – IAM permissions



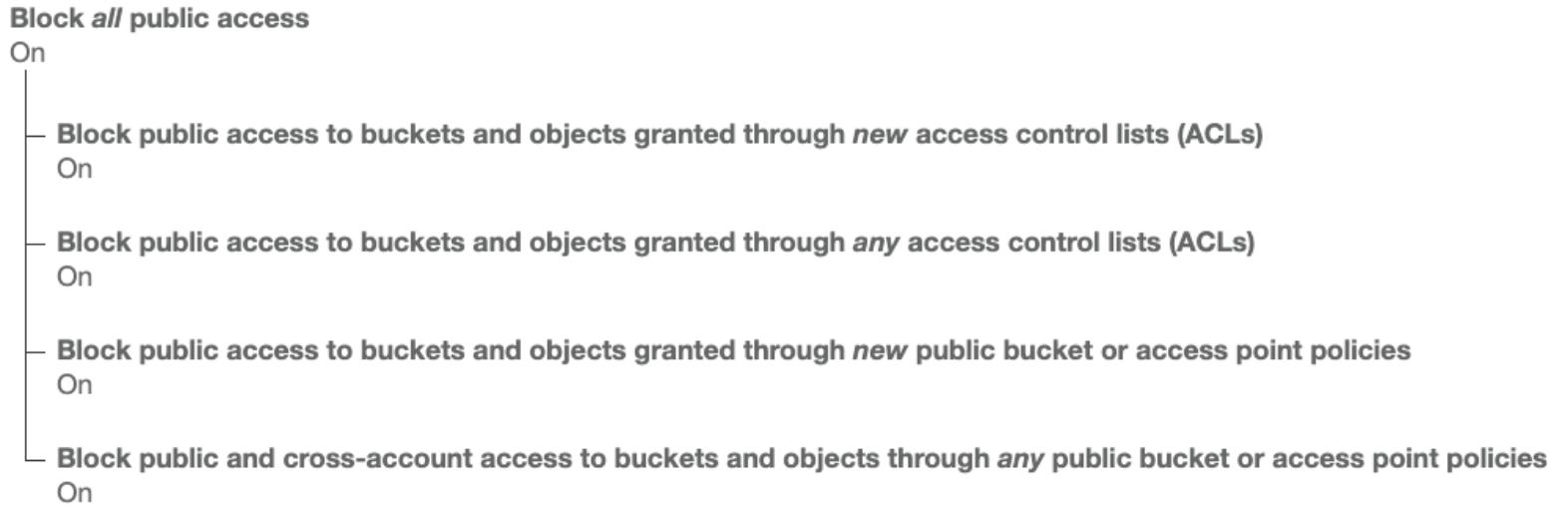
# Example: EC2 instance access - Use IAM Roles



# Advanced: Cross-Account Access – Use Bucket Policy



# 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 Bucket Policies

- 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)
- Optional Conditions on:
  - Public IP or Elastic IP (not on Private IP)
  - Source VPC or Source VPC Endpoint – only works with VPC Endpoints
  - CloudFront Origin Identity
  - MFA
- Examples here: <https://docs.aws.amazon.com/AmazonS3/latest/dev/example-bucket-policies.html>

# Bucket Policies – Advanced Examples

- Restrict access to only principals from AWS accounts inside an AWS Organization using `aws:PrincipalOrgID` condition key
- Prevent uploads of unencrypted objects to an S3 bucket using `s3:x-amz-server-side-encryption` condition key

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": "*",  
            "Action": "s3:GetObject",  
            "Resource": "arn:aws:s3:::mybucket/*",  
            "Condition": {  
                "StringEquals": {  
                    "aws:PrincipalOrgID": ["o-exampleorgid"]  
                }  
            }  
        }  
    ]  
}
```

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Deny",  
            "Principal": "*",  
            "Action": "s3:PutObject",  
            "Resource": "arn:aws:s3:::mybucket/*",  
            "Condition": {  
                "Null": {  
                    "s3:x-amz-server-side-encryption": true  
                }  
            }  
        }  
    ]  
}
```

# Bucket Policies – Advanced Examples

- Restrict access to specific IP addresses using `NotIpAddress` condition key

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Deny",  
            "Principal": "*",  
            "Action": "s3:*",  
            "Resource": [  
                "arn:aws:s3:::mybucket",  
                "arn:aws:s3:::mybucket/*"  
            ],  
            "Condition": {  
                "NotIpAddress": {  
                    "aws:SourceIp": "54.240.143.0/24"  
                }  
            }  
        }  
    ]  
}
```

# Bucket Policies – Advanced Examples

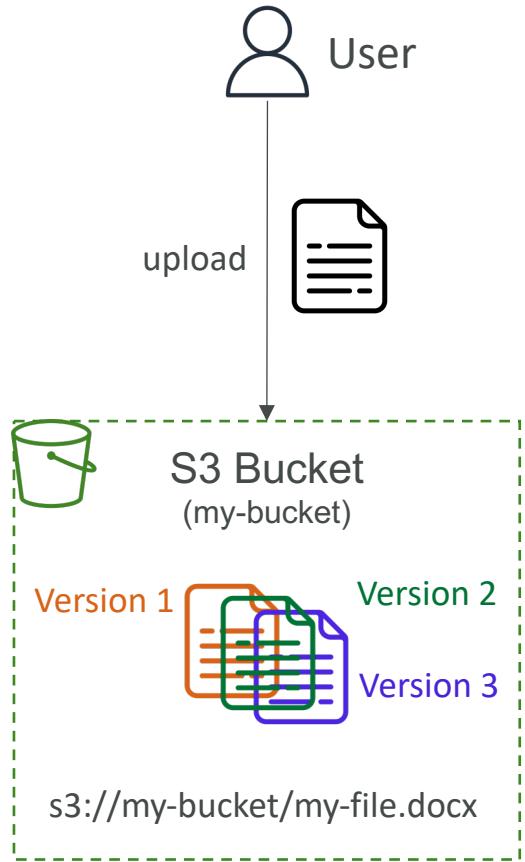
- Grant user access to list and download all objects in an S3 bucket
- Restrict access to users authenticated using MFA using `MultiFactorAuthPresent` condition key

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

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": "*",  
      "Action": "s3:GetObject",  
      "Resource": "arn:aws:s3:::mybucket/*",  
      "Condition": {  
        "Bool": {  
          "aws:MultiFactorAuthPresent": "true"  
        }  
      }  
    }  
  ]  
}
```

# Amazon S3 - Versioning

- You can version your files in Amazon S3
- It is enabled at the **bucket level**
- Same key overwrite will change 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



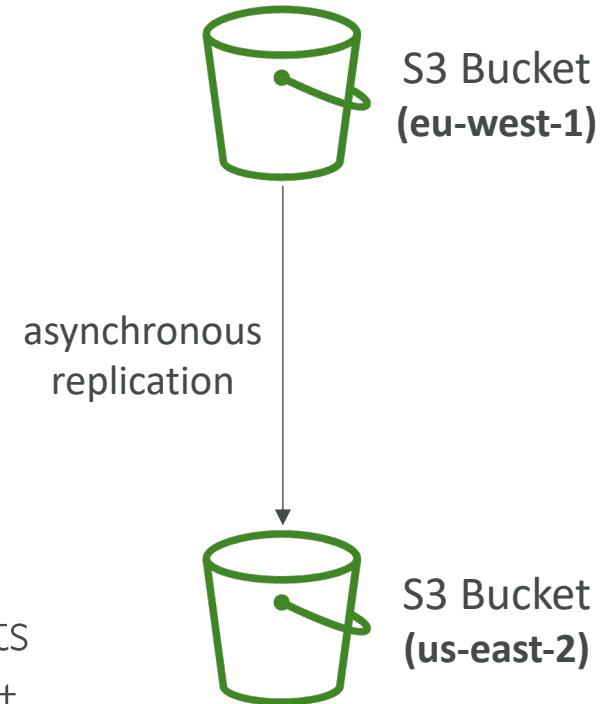
# Amazon S3 – Versioning – Troubleshooting

- When you enable Versioning for the first time, you have to wait at least 15 minutes to be fully propagated
- After enable Versioning, HTTP 404 NoSuchKey error when trying to get Objects created or updated
  - Wait 15 minutes after enable Versioning before issuing write operations
- <https://docs.aws.amazon.com/AmazonS3/latest/userguide/versioning-workflows.html>

# Amazon S3 – Replication (CRR & SRR)



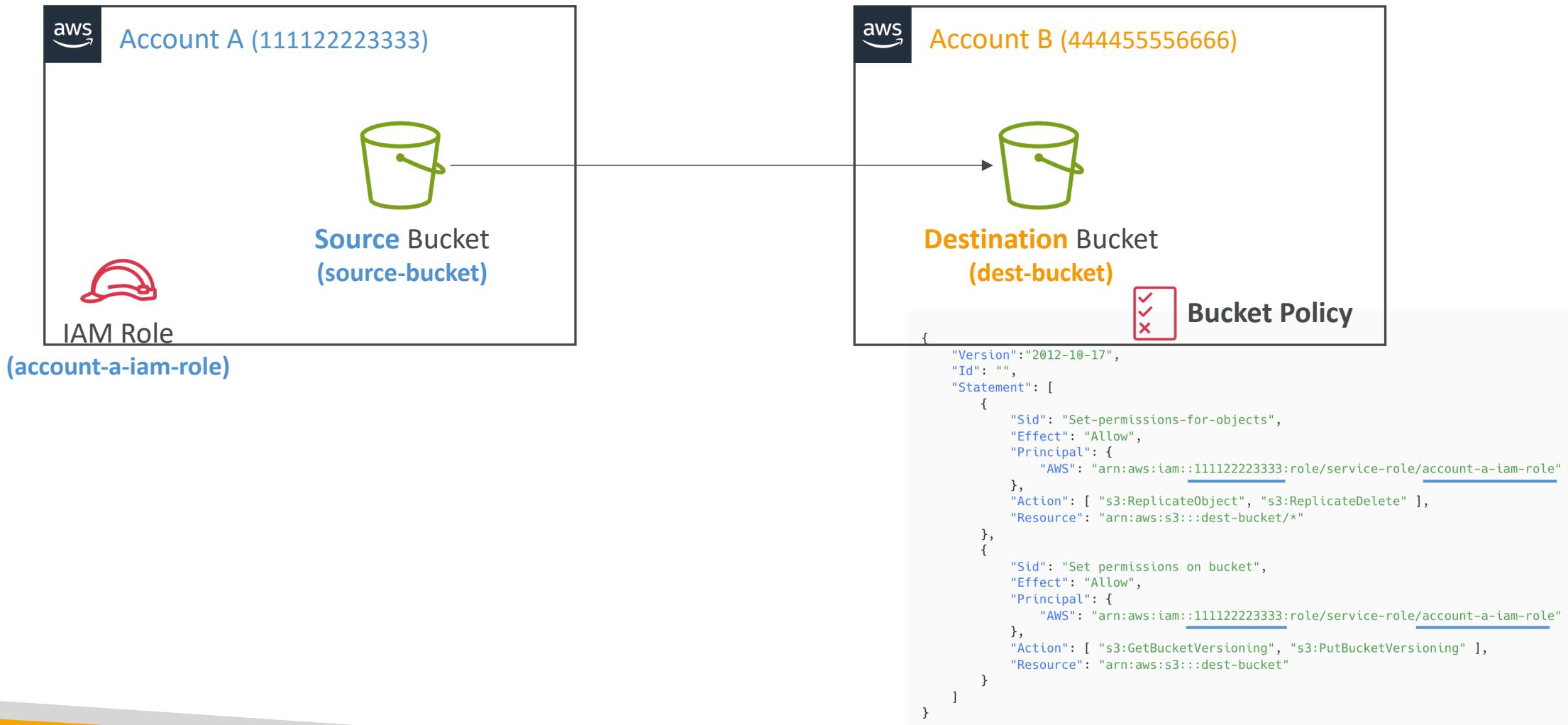
- Must enable Versioning in source and destination buckets
- Cross-Region Replication (CRR)
- Same-Region Replication (SRR)
- Buckets can be in different AWS accounts
- Copying is asynchronous
- Must give proper IAM permissions to S3
- Use cases:
  - CRR – compliance, lower latency access, replication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



# Amazon S3 – Replication (Notes)

- After you enable Replication, only new objects are replicated
- Optionally, you can replicate existing objects using **S3 Batch Replication**
  - Replicates existing objects and objects that failed replication
- For DELETE operations
  - Can replicate **delete markers** from source to target (optional setting)
  - Deletions with a version ID are not replicated (to avoid malicious deletes)
- **There is no “chaining” of replication**
  - If bucket 1 has replication into bucket 2, which has replication into bucket 3
  - Then objects created in bucket 1 are not replicated to bucket 3

# Amazon S3 – Cross-account S3 Replication



# Amazon S3 – Cross-account S3 Replication

- By default, the owner of the source object (**source** bucket) is the owner of the replica object (**destination** bucket)
- To change the replica object owner to the **destination** account:
  - Add the *owner override* option to the replication rule config
  - In **source** account, grant Amazon S3 permission to change replica ownership (*s3:ObjectOwnerOverrideToBucketOwner* in the IAM role)
  - In **destination** account, update the **destination** bucket policy to add the permission to allow change replica ownership (*s3:ObjectOwnerOverrideToBucketOwner*)

Source IAM Role

```
{...  
  "Effect": "Allow",  
  "Action": [ "s3:ObjectOwnerOverrideToBucketOwner" ],  
  "Resource": "arn:aws:s3:::dest-bucket/*"  
}  
...
```

Destination Bucket Policy

```
...  
{  
  "Effect": "Allow",  
  "Principal": { "AWS": "source-account-id" },  
  "Action": [ "s3:ObjectOwnerOverrideToBucketOwner" ],  
  "Resource": "arn:aws:s3:::dest-bucket/*"  
}  
...
```

# Amazon S3 – Replication Time Control (RTC)

- Feature of Amazon S3 Replication that guarantees replication speed
- 99.99% of new objects replicated within 15 minutes
- Provides predictable, auditable replication time
- Publishes CloudWatch metrics to monitoring
- Alerts you if replication falls behind
- Can replicate to same or different AWS Region
- Helps you meet compliance or business requirements
- Extra cost per GB for the replication

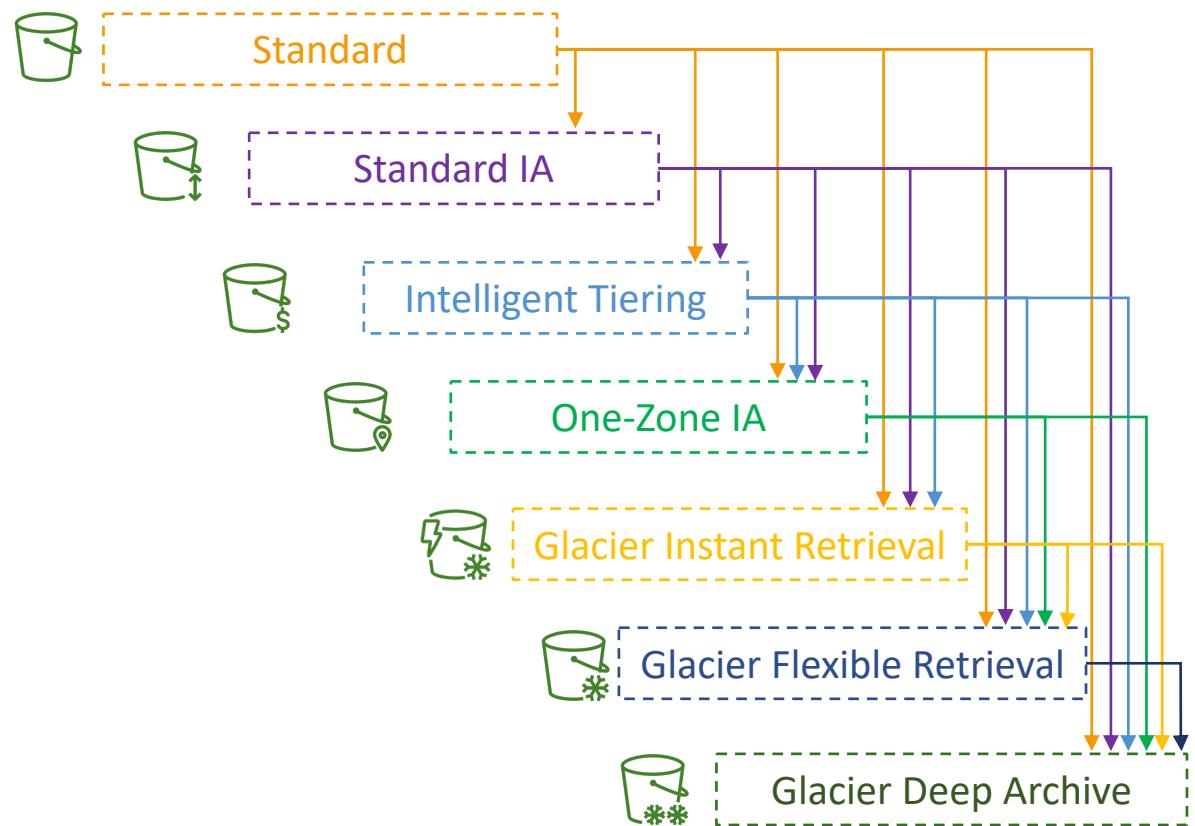


S3 RTC

# Amazon S3 – Advanced

# Amazon S3 – Moving between Storage Classes

- You can transition objects between storage classes
- For infrequently accessed object, move them to **Standard IA**
- For archive objects that you don't need fast access to, move them to **Glacier or Glacier Deep Archive**
- Moving objects can be automated using a **Lifecycle Rules**





# Amazon S3 – Lifecycle Rules

- **Transition Actions** – configure objects to transition to another storage class
  - Move objects to Standard IA class 60 days after creation
  - Move to Glacier for archiving after 6 months
- **Expiration actions** – configure objects to expire (delete) after some time
  - Access log files can be set to delete after a 365 days
  - **Can be used to delete old versions of files (if versioning is enabled)**
  - Can be used to delete incomplete Multi-Part uploads
- Rules can be created for a certain prefix (example: s3://mybucket/mp3/\*)
- Rules can be created for certain objects Tags (example: Department: Finance)

# Amazon S3 – Lifecycle Rules (Scenario I)

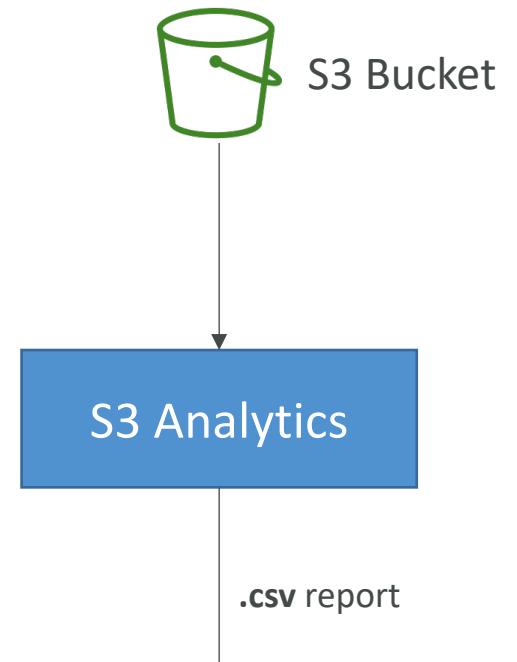
- Your application on EC2 creates images thumbnails after profile photos are uploaded to Amazon S3. These thumbnails can be easily recreated, and only need to be kept for 60 days. The source images should be able to be immediately retrieved for these 60 days, and afterwards, the user can wait up to 6 hours. How would you design this?
- S3 source images can be on **Standard**, with a lifecycle configuration to transition them to **Glacier** after 60 days
- S3 thumbnails can be on **One-Zone IA**, with a lifecycle configuration to expire them (delete them) after 60 days

# Amazon S3 – Lifecycle Rules (Scenario 2)

- A rule in your company states that you should be able to recover your deleted S3 objects immediately for 30 days, although this may happen rarely. After this time, and for up to 365 days, deleted objects should be recoverable within 48 hours.
- Enable **S3 Versioning** in order to have object versions, so that “deleted objects” are in fact hidden by a “delete marker” and can be recovered
- Transition the “noncurrent versions” of the object to **Standard IA**
- Transition afterwards the “noncurrent versions” to **Glacier Deep Archive**

# Amazon S3 Analytics – Storage Class Analysis

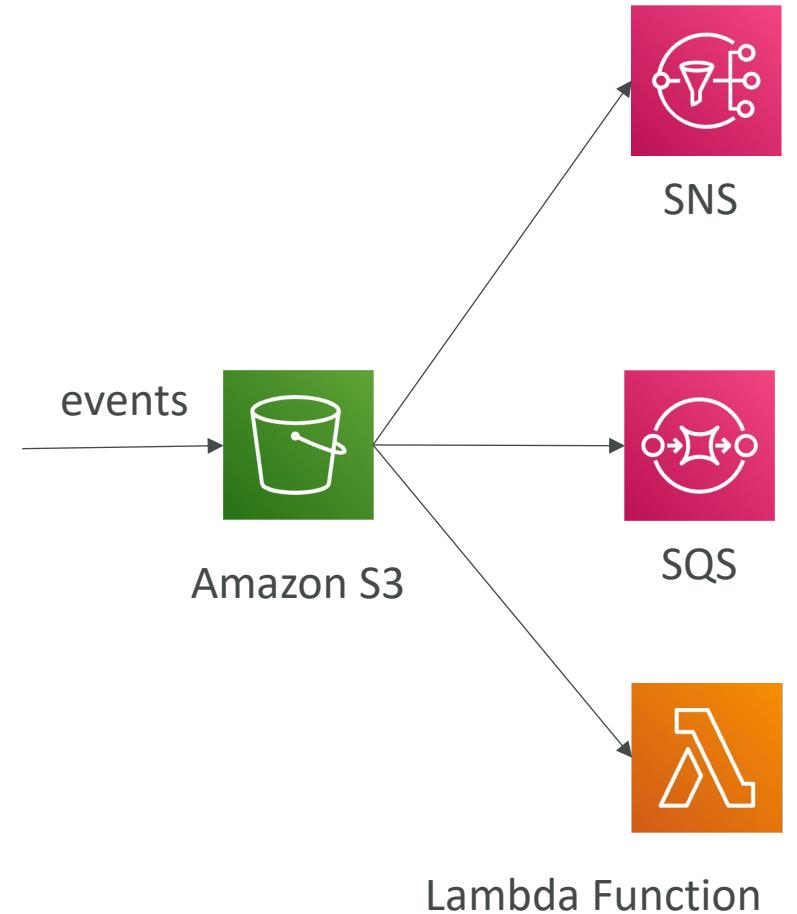
- Help you decide when to transition objects to the right storage class
- Recommendations for **Standard** and **Standard IA**
  - Does NOT work for One-Zone IA or Glacier
- Report is updated daily
- 24 to 48 hours to start seeing data analysis
- Good first step to put together Lifecycle Rules (or improve them)!



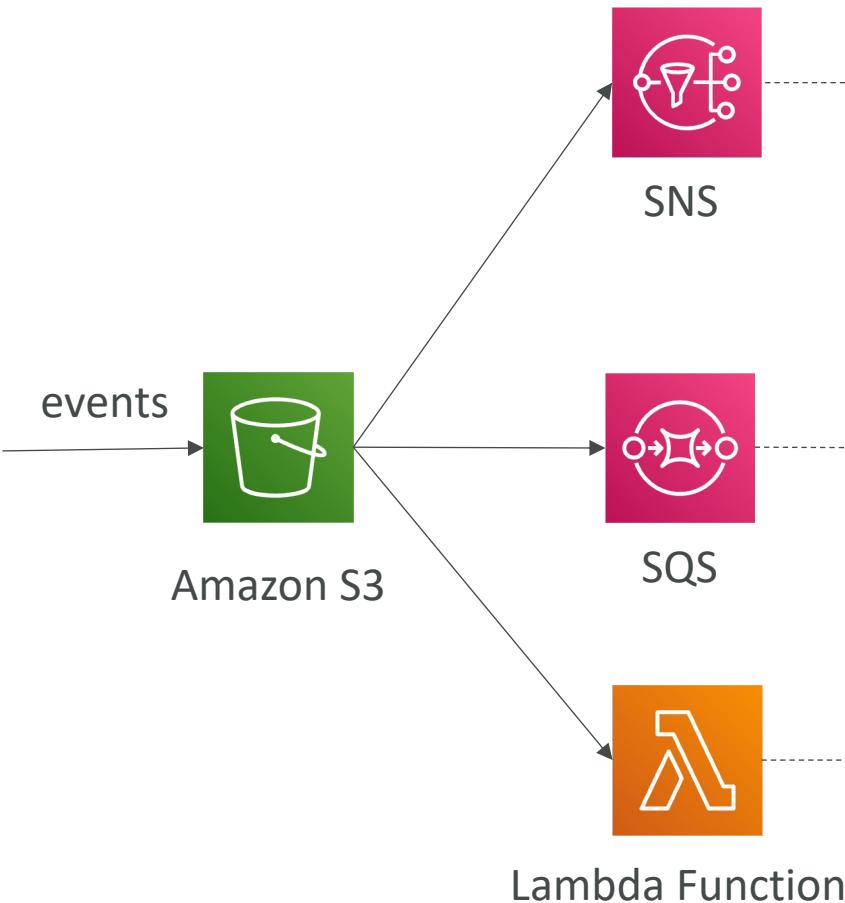
Date	StorageClass	ObjectAge
8/22/2022	STANDARD	000-014
8/25/2022	STANDARD	030-044
9/6/2022	STANDARD	120-149

# S3 Event Notifications

- S3:ObjectCreated, S3:ObjectRemoved, S3:ObjectRestore, S3:Replication...
- Object name filtering possible (\*.jpg)
- Use case: generate thumbnails of images uploaded to S3
- Can create as many “S3 events” as desired
- S3 event notifications typically deliver events in seconds but can sometimes take a minute or longer



# S3 Event Notifications – IAM Permissions



{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": "SNS:Publish",  
        "Principal": {  
            "Service": "s3.amazonaws.com"  
        },  
        "Resource": "arn:aws:sns:us-east-1:123456789012:MyTopic",  
        "Condition": {  
            "ArnLike": {  
                "aws:SourceArn": "arn:aws:s3:::MyBucket"  
            }  
        }  
    }  
}

**SNS Resource (Access) Policy**

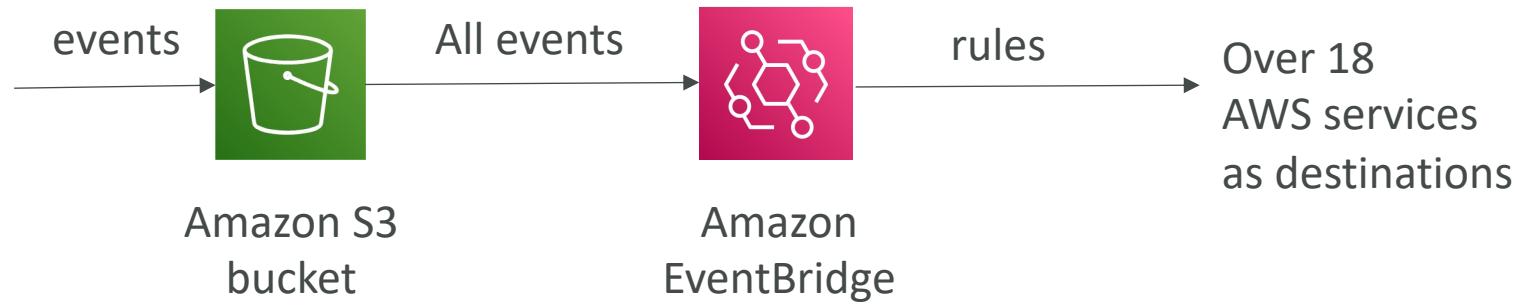
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": "SQS:SendMessage",  
        "Principal": {  
            "Service": "s3.amazonaws.com"  
        },  
        "Resource": "arn:aws:sqs:us-east-1:123456789012:MyQueue",  
        "Condition": {  
            "ArnLike": {  
                "aws:SourceArn": "arn:aws:s3:::MyBucket"  
            }  
        }  
    }  
}

**SQS Resource (Access) Policy**

{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": "lambda:InvokeFunction",  
        "Principal": {  
            "Service": "s3.amazonaws.com"  
        },  
        "Resource": "arn:aws:lambda:us-east-1:123456789012:function:MyFunction",  
        "Condition": {  
            "ArnLike": {  
                "AWS:SourceArn": "arn:aws:s3:::MyBucket"  
            }  
        }  
    }  
}

**Lambda Resource Policy**

# S3 Event Notifications with Amazon EventBridge



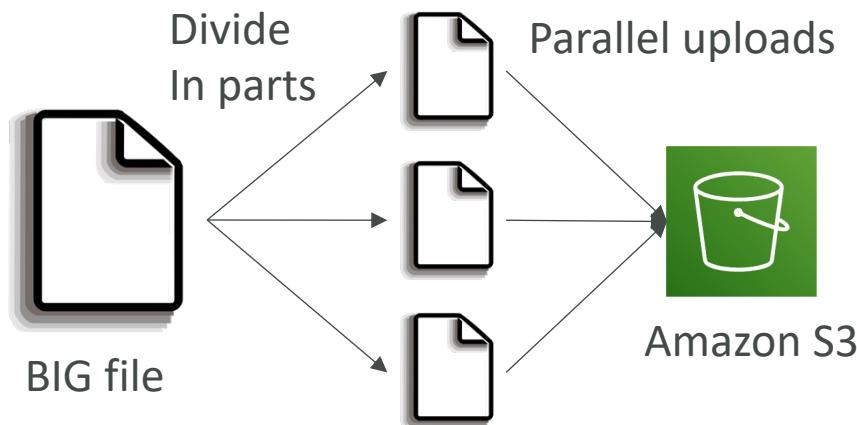
- Advanced filtering options with JSON rules (metadata, object size, name...)
- Multiple Destinations – ex Step Functions, Kinesis Streams / Firehose...
- EventBridge Capabilities – Archive, Replay Events, Reliable delivery

# S3 – Baseline Performance

- Amazon S3 automatically scales to high request rates, latency 100-200 ms
- Your application can achieve at least 3,500 PUT/COPY/POST/DELETE or 5,500 GET/HEAD requests per second per prefix in a bucket.
- There are no limits to the number of prefixes in a bucket.
- Example (object path => prefix):
  - bucket/folder1/sub1/file => /folder1/sub1/
  - bucket/folder1/sub2/file => /folder1/sub2/
  - bucket/1/file => /1/
  - bucket/2/file => /2/
- If you spread reads across all four prefixes evenly, you can achieve 22,000 requests per second for GET and HEAD

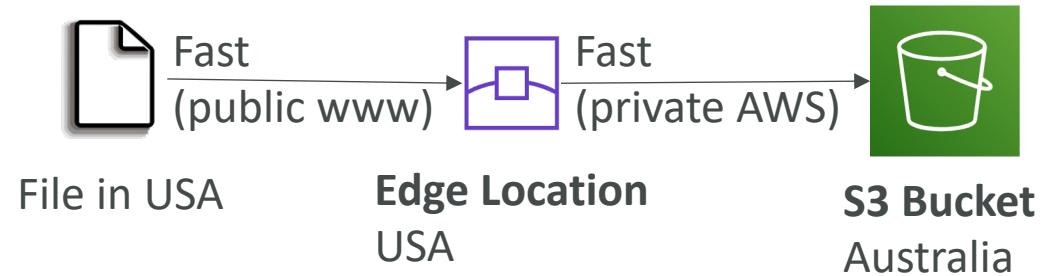
# S3 Performance

- Multi-Part upload:
  - recommended for files > 100MB, must use for files > 5GB
  - Can help parallelize uploads (speed up transfers)



- S3 Transfer Acceleration

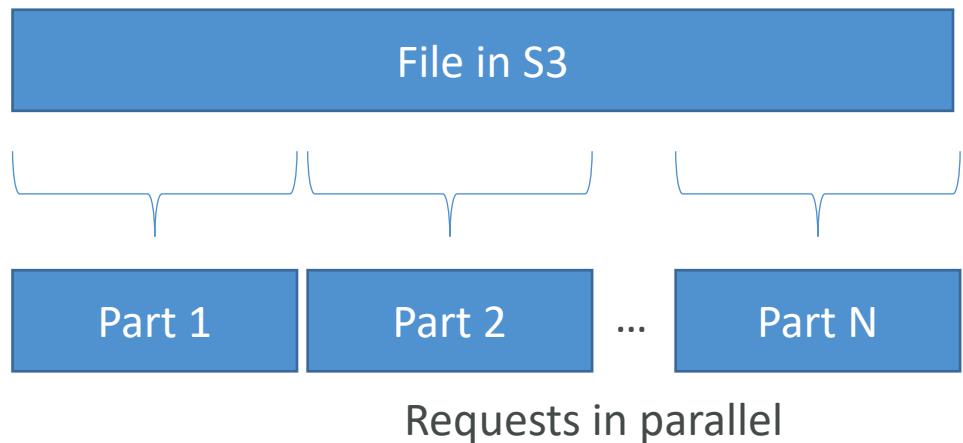
- Increase transfer speed by transferring file to an AWS edge location which will forward the data to the S3 bucket in the target region
- Compatible with multi-part upload



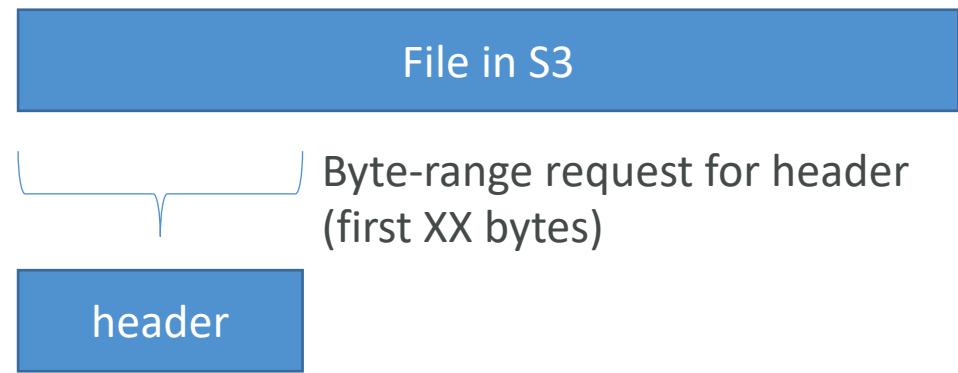
# S3 Performance – S3 Byte-Range Fetches

- Parallelize GETs by requesting specific byte ranges
- Better resilience in case of failures

Can be used to speed up downloads

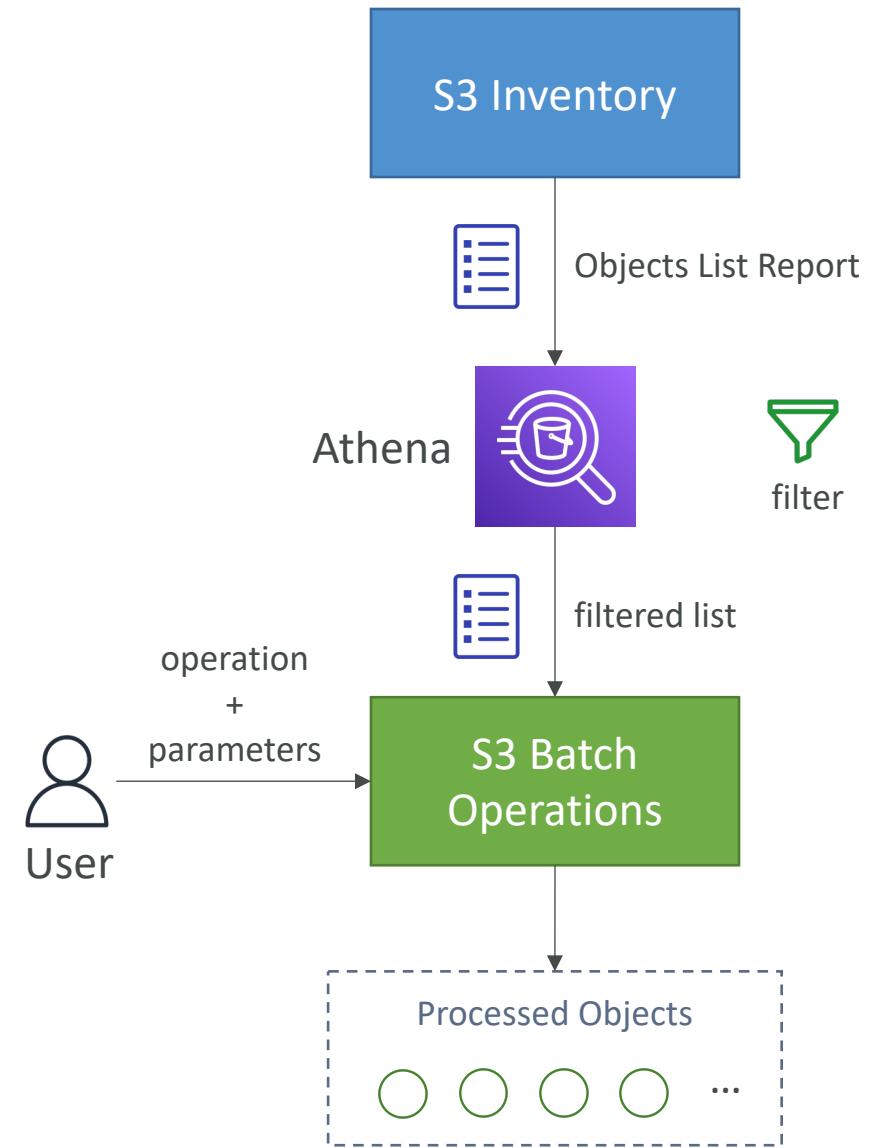


Can be used to retrieve only partial data (for example the head of a file)



# S3 Batch Operations

- Perform bulk operations on existing S3 objects with a single request, example:
  - Modify object metadata & properties
  - Copy objects between S3 buckets
  - **Encrypt un-encrypted objects**
  - Modify ACLs, tags
  - Restore objects from S3 Glacier
  - Invoke Lambda function to perform custom action on each object
- A job consists of a list of objects, the action to perform, and optional parameters
- S3 Batch Operations manages retries, tracks progress, sends completion notifications, generate reports ...
- You can use S3 Inventory to get object list and use Athena to query and filter your objects

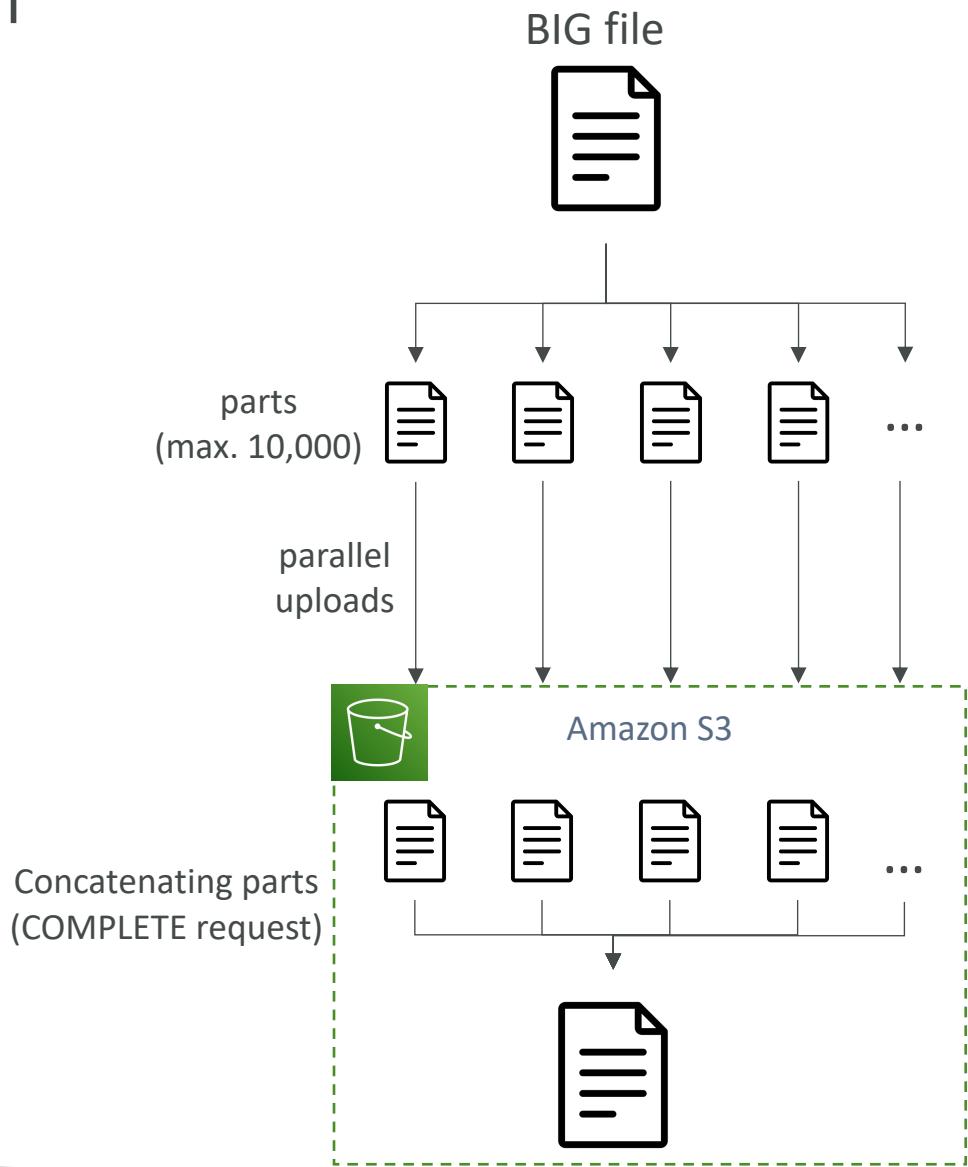


# S3 Inventory

- List objects and their corresponding metadata (alternative to S3 List API operation)
- Usage examples:
  - Audit and report on the replication and encryption status of your objects
  - Get the number of objects in an S3 bucket
  - Identify the total storage of previous object versions
- Generate daily or weekly reports
- Output files: CSV, ORC, or Apache Parquet
- You can query all the data using Amazon Athena, Redshift, Presto, Hive, Spark...
- Use cases: Business, Compliance, Regulatory needs, ...

# S3 Multi Part Upload – Deep Dive

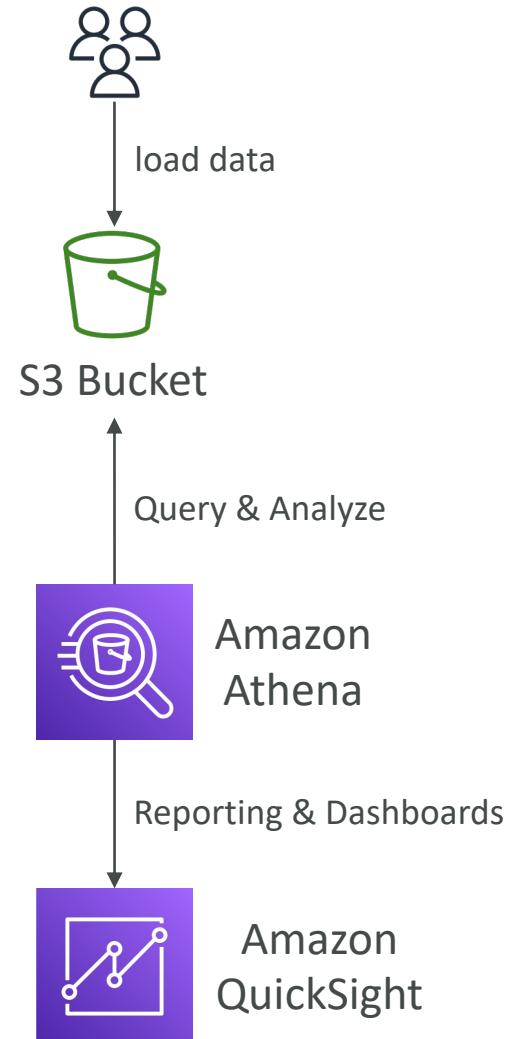
- Upload large objects in parts (any order)
- Recommended for files > 100MB, must use for files > 5GB
- Can help parallelize uploads (speed up transfers)
- Max. parts: 10,000
- **Failures:** restart uploading ONLY failed parts (improves performance)
- Use **Lifecycle Policy** to automate old parts deletion of unfinished upload after x days (e.g., network outage)
- Upload using AWS CLI or AWS SDK



# Amazon Athena



- Serverless query service to analyze data stored in Amazon S3
- Uses standard SQL language to query the files (built on Presto)
- Supports CSV, JSON, ORC, Avro, and Parquet
- Pricing: \$5.00 per TB of data scanned
- Commonly used with Amazon Quicksight for reporting/dashboards
- **Use cases:** Business intelligence / analytics / reporting, analyze & query VPC Flow Logs, ELB Logs, CloudTrail trails, etc...
- **Exam Tip:** analyze data in S3 using serverless SQL, use Athena

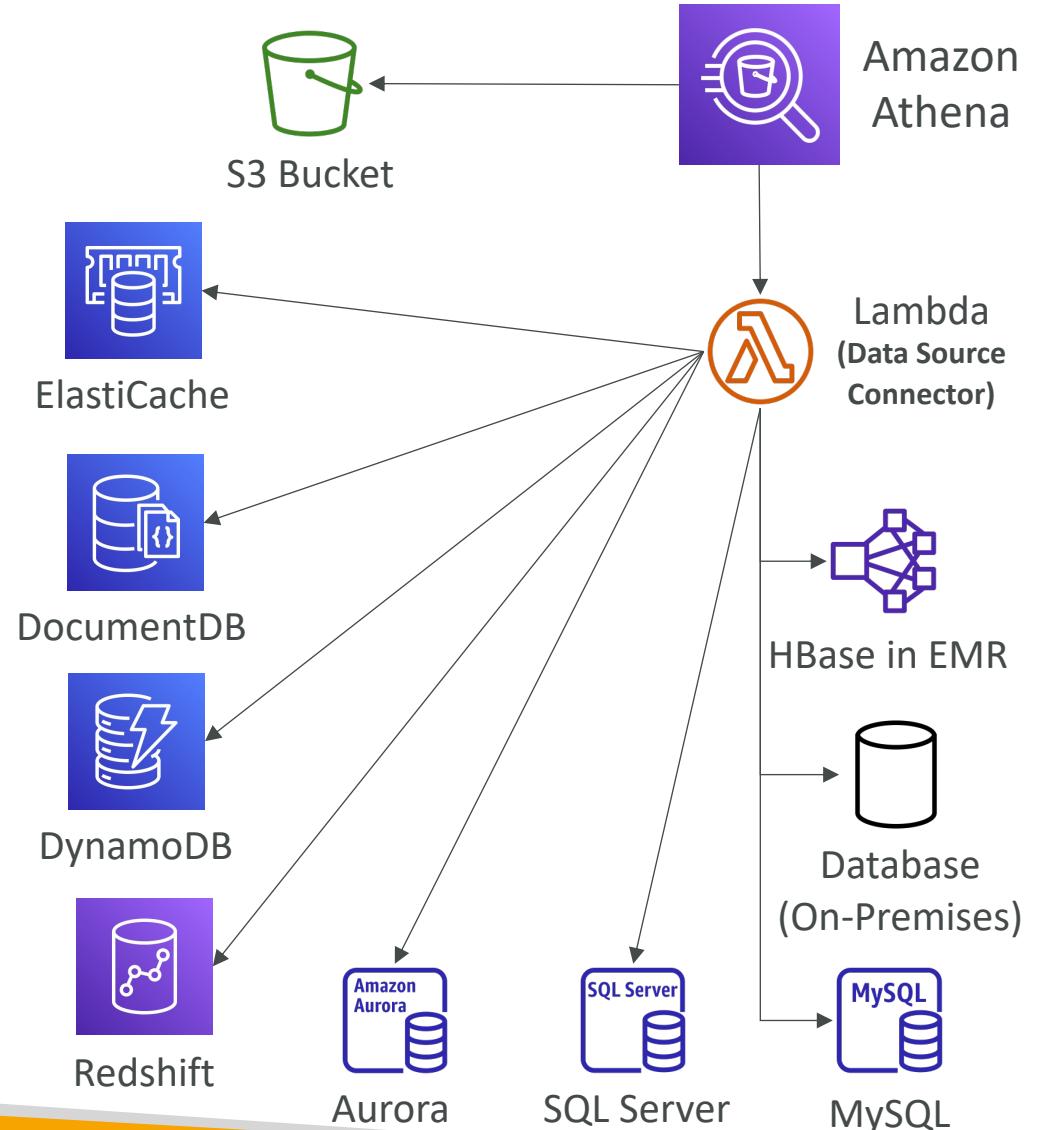


# Amazon Athena – Performance Improvement

- Use **columnar data** for cost-savings (less scan)
  - Apache Parquet or ORC is recommended
  - Huge performance improvement
  - Use Glue to convert your data to Parquet or ORC
- **Compress data** for smaller retrievals (bzip2, gzip, lz4, snappy, zlip, zstd...)
- **Partition** datasets in S3 for easy querying on virtual columns
  - s3://yourBucket/pathToTable  
  / <PARTITION\_COLUMN\_NAME>=<VALUE>  
  / <PARTITION\_COLUMN\_NAME>=<VALUE>  
  / <PARTITION\_COLUMN\_NAME>=<VALUE>  
  /etc...
  - Example: s3://athena-examples/flight/parquet/year=1991/month=1/day=1/
- Use **larger files** (> 128 MB) to minimize overhead

# Amazon Athena – Federated Query

- Allows you to run SQL queries across data stored in relational, non-relational, object, and custom data sources (AWS or on-premises)
- Uses Data Source Connectors that run on AWS Lambda to run Federated Queries (e.g., CloudWatch Logs, DynamoDB, RDS, ...)
- Store the results back in Amazon S3



# Amazon S3 – Security

# Amazon S3 – MFA Delete

- **MFA (Multi-Factor Authentication)** – force users to generate a code on a device (usually a mobile phone or hardware) before doing important operations on S3
- MFA will be required to:
  - Permanently delete an object version
  - Suspend Versioning on the bucket
- MFA won't be required to:
  - Enable Versioning
  - List deleted versions
- To use MFA Delete, Versioning must be enabled on the bucket
- Only the bucket owner (root account) can enable/disable MFA Delete



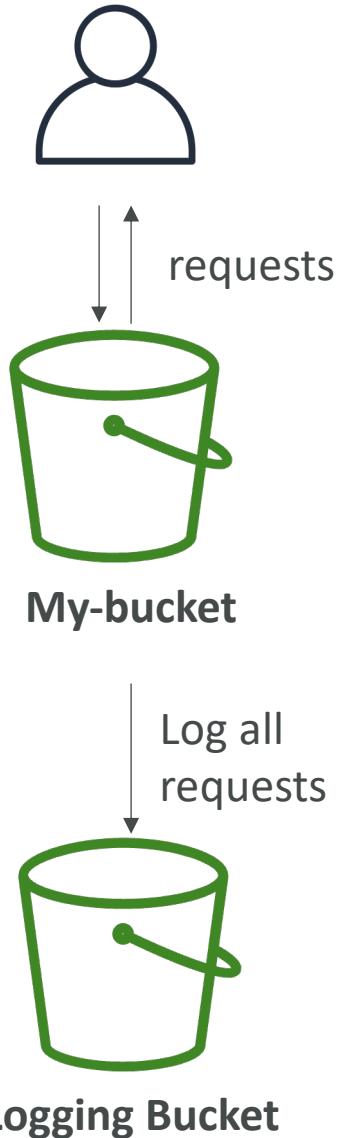
Google Authenticator



MFA Hardware Device

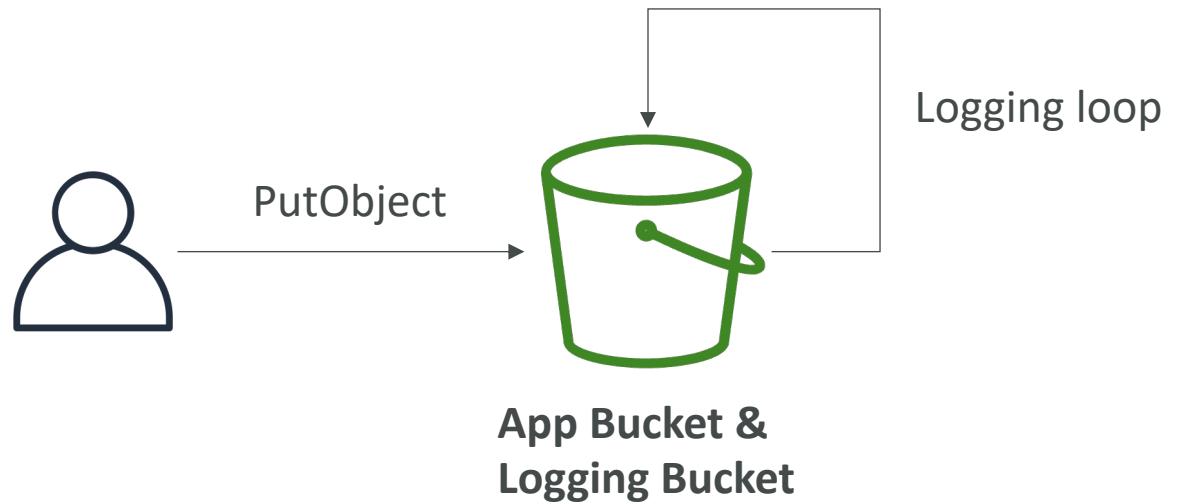
# S3 Access Logs

- 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...
  - The target logging bucket must be in the same AWS region
- 
- The log format is at:  
<https://docs.aws.amazon.com/AmazonS3/latest/dev/LogFormat.html>



# S3 Access Logs: Warning

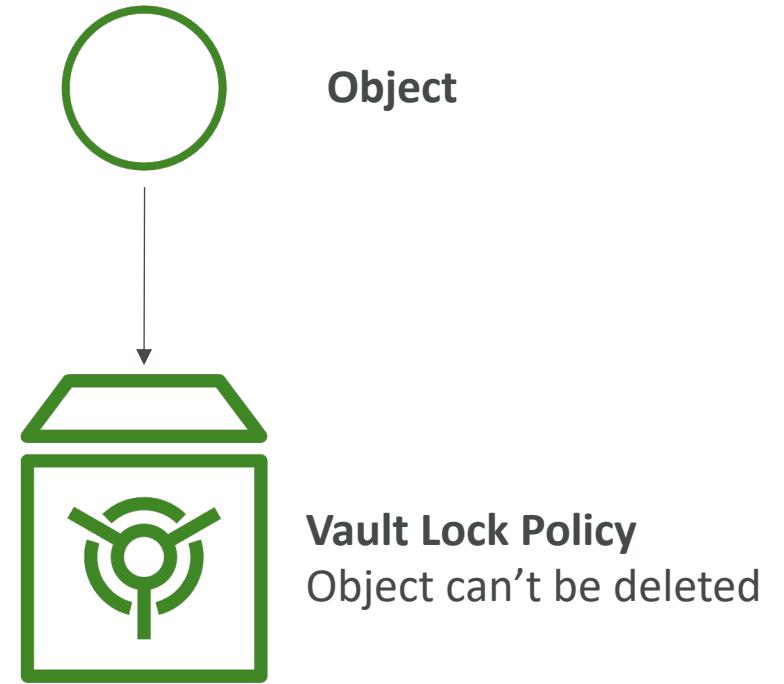
- Do not set your logging bucket to be the monitored bucket
- It will create a logging loop, and **your bucket will grow exponentially**



Do not try this at home 😊

# S3 Glacier Vault Lock

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



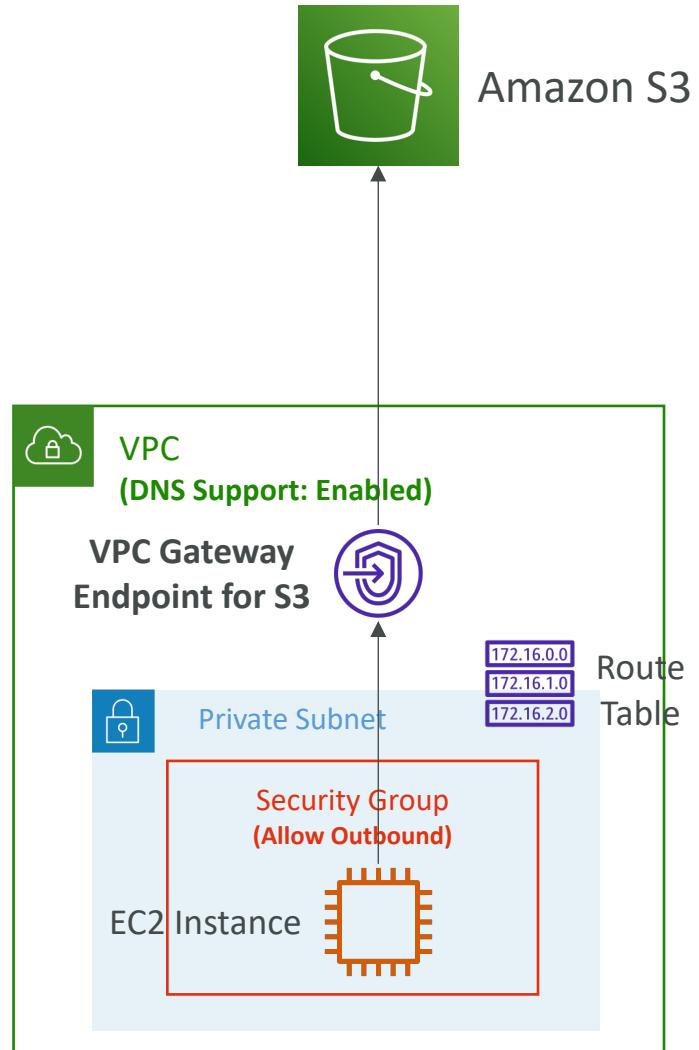
# S3 Object Lock (versioning must be enabled)

- Adopt a WORM (Write Once Read Many) model
- Block an object version deletion for a specified amount of time
- **Retention mode - Compliance:**
  - Object versions can't be overwritten or deleted by any user, including the root user
  - Objects retention modes can't be changed, and retention periods can't be shortened
- **Retention mode - Governance:**
  - Most users can't overwrite or delete an object version or alter its lock settings
  - Some users have special permissions to change the retention or delete the object
- **Retention Period:** protect the object for a fixed period, it can be extended
- **Legal Hold:**
  - protect the object indefinitely, independent from retention period
  - can be freely placed and removed using the `s3:PutObjectLegalHold` IAM permission

# VPC Gateway Endpoint for Amazon S3

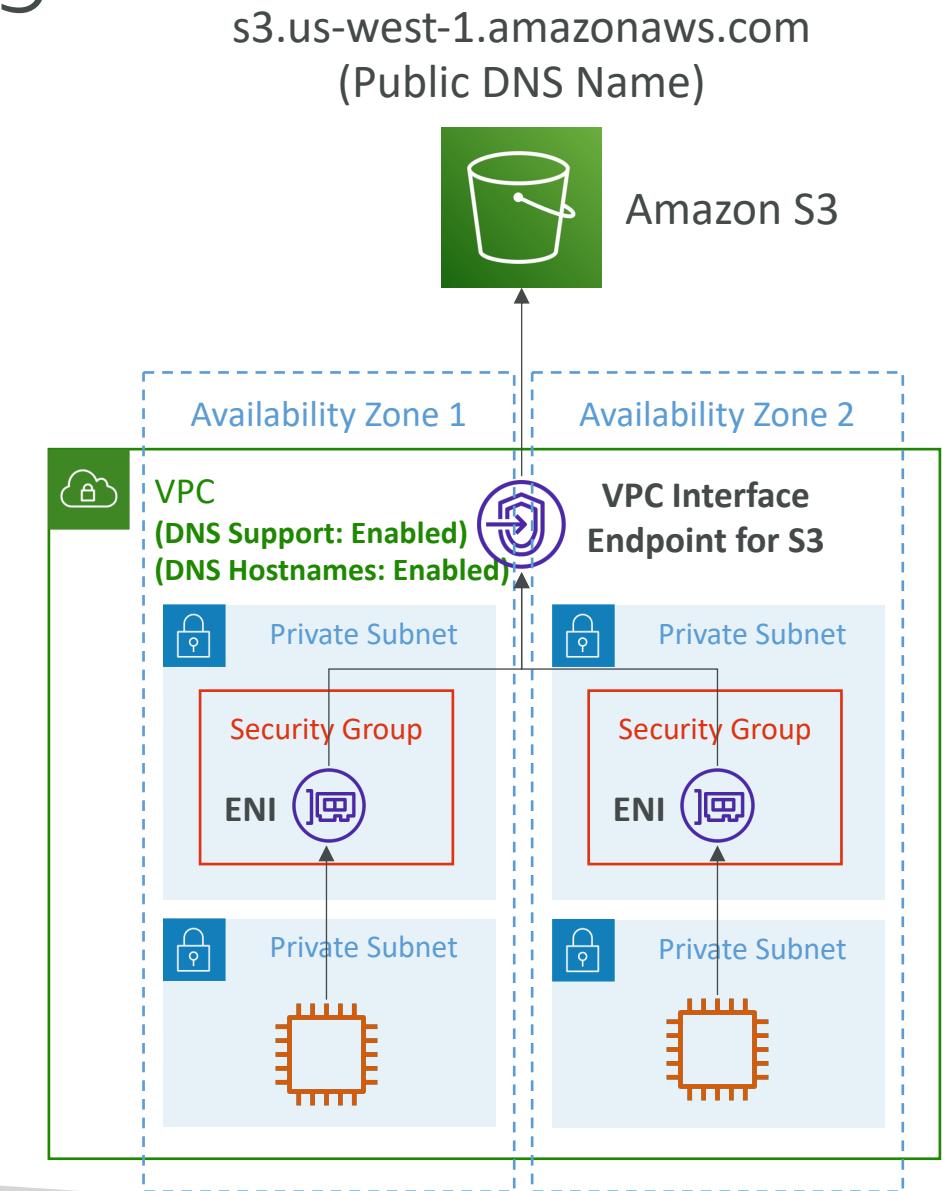
- No cost
- Only accessed by resources in the VPC where it's created
- Make sure “DNS Support” is Enabled
- Keep on using the public DNS of Amazon S3
- Make sure Outbound rules of SG of EC2 instance allows traffic to S3

Outbound rules (1/1)						
<input type="button" value="Filter security group rules"/>						
<input checked="" type="checkbox"/>	Name	Security group rule...	IP version	Type	Protocol	Port range
<input checked="" type="checkbox"/>	-	sgr-0ee382d4b7d379...	-	HTTPS	TCP	443



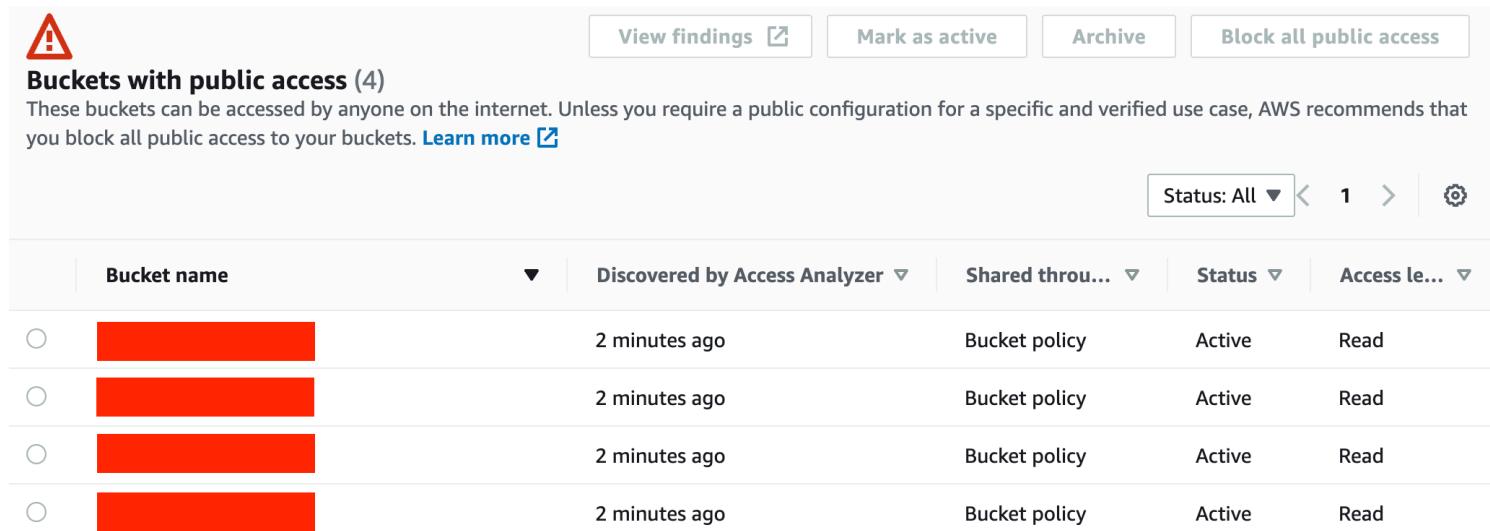
# VPC Interface Endpoint for S3

- ENI(s) are deployed in your Subnets (Security Groups can be attached to ENIs)
- Can access from on-premises (VPN or Direct Connect)
- Costs \$0.01 per hour per AZ
- Both VPC settings “Enable DNS hostnames” and “Enable DNS Support” must be ‘true’



# IAM Access Analyzer for S3

- Ensures that only intended people have access to your S3 buckets
- Example: publicly accessible bucket, bucket shared with other AWS account...
- Evaluates S3 Bucket Policies, S3 ACLs, S3 Access Point Policies
- Powered by IAM Access Analyzer



The screenshot shows a warning message for buckets with public access. It includes buttons for 'View findings' and 'Block all public access', and links for 'Mark as active' and 'Archive'. A table lists four buckets, each with a redacted name, discovered 2 minutes ago via a bucket policy, and marked as Active with Read access.

Bucket name	Discovered by Access Analyzer	Shared through	Status	Access level
[REDACTED]	2 minutes ago	Bucket policy	Active	Read
[REDACTED]	2 minutes ago	Bucket policy	Active	Read
[REDACTED]	2 minutes ago	Bucket policy	Active	Read
[REDACTED]	2 minutes ago	Bucket policy	Active	Read

# Advanced Storage Solutions

# Amazon FSx – Overview



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



FSx for Lustre



FSx for  
Windows  
File Server



FSx for  
NetApp ONTAP



FSx for  
OpenZFS

# Amazon FSx for Windows (File Server)



- FSx for Windows is a fully managed Windows file system share drive
- Supports SMB protocol & Windows NTFS
- Microsoft Active Directory integration, ACLs, user quotas
- Can be mounted on Linux EC2 instances
- Supports Microsoft's Distributed File System (DFS) Namespaces (group files across multiple FS)
- Scale up to 10s of GB/s, millions of IOPS, 100s PB of data
- Storage Options:
  - SSD – latency sensitive workloads (databases, media processing, data analytics, ...)
  - HDD – broad spectrum of workloads (home directory, CMS, ...)
- Can be accessed from your on-premises infrastructure (VPN or Direct Connect)
- Can be configured to be Multi-AZ (high availability)
- Data is backed-up daily to S3

# Amazon FSx for Lustre



- Lustre is a type of parallel distributed file system, for large-scale computing
- The name Lustre is derived from “Linux” and “cluster”
- Machine Learning, **High Performance Computing (HPC)**
- Video Processing, Financial Modeling, Electronic Design Automation
- Scales up to 100s GB/s, millions of IOPS, sub-ms latencies
- Storage Options:
  - SSD – low-latency, IOPS intensive workloads, small & random file operations
  - HDD – throughput-intensive workloads, large & sequential file operations
- **Seamless integration with S3**
  - Can “read S3” as a file system (through FSx)
  - Can write the output of the computations back to S3 (through FSx)
- **Can be used from on-premises servers (VPN or Direct Connect)**

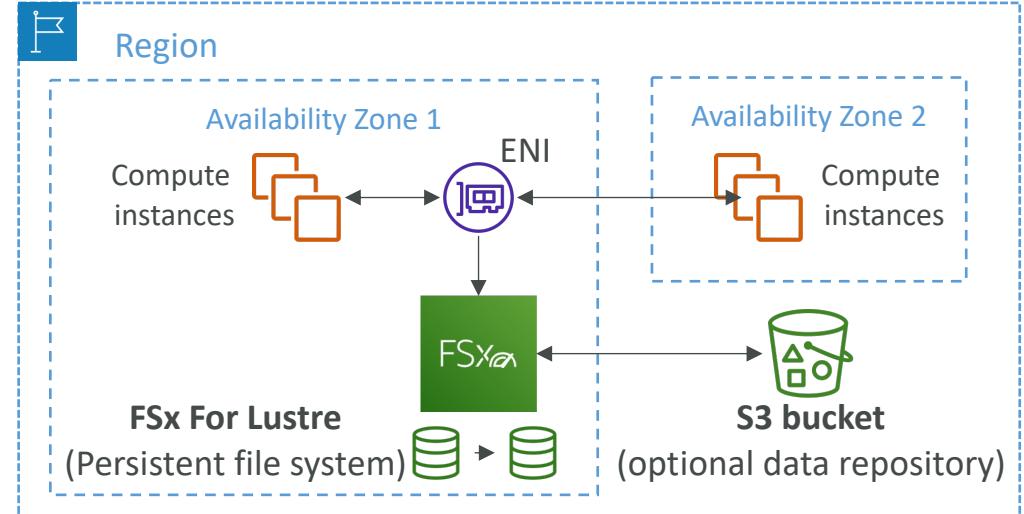
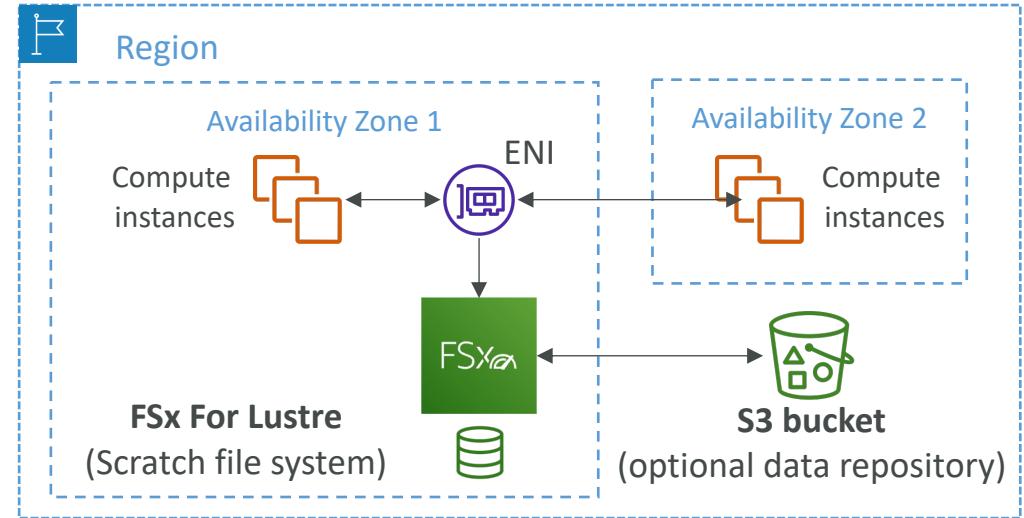
# FSx Lustre - File System Deployment Options

## • Scratch File System

- Temporary storage
- Data is not replicated (doesn't persist if file server fails)
- High burst (6x faster, 200MBps per TiB)
- Usage: short-term processing, optimize costs

## • Persistent File System

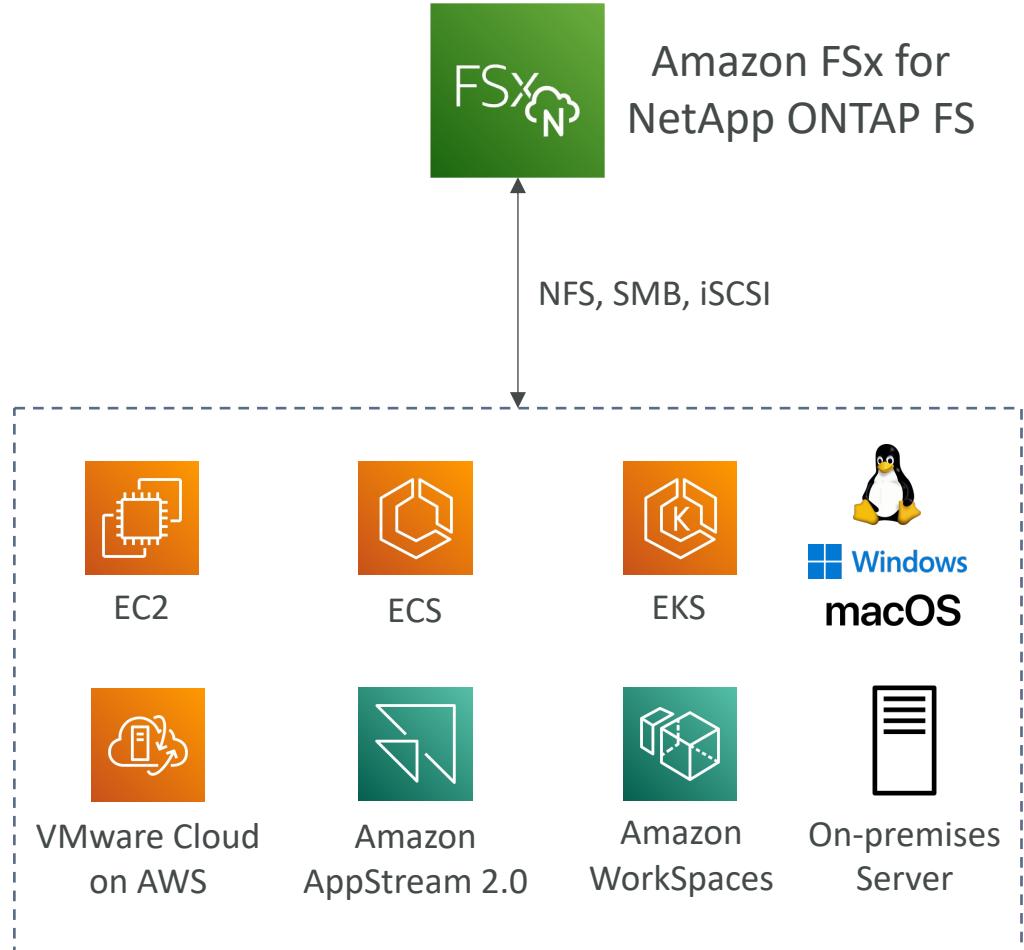
- Long-term storage
- Data is replicated within same AZ
- Replace failed files within minutes
- Usage: long-term processing, sensitive data



# Amazon FSx for NetApp ONTAP



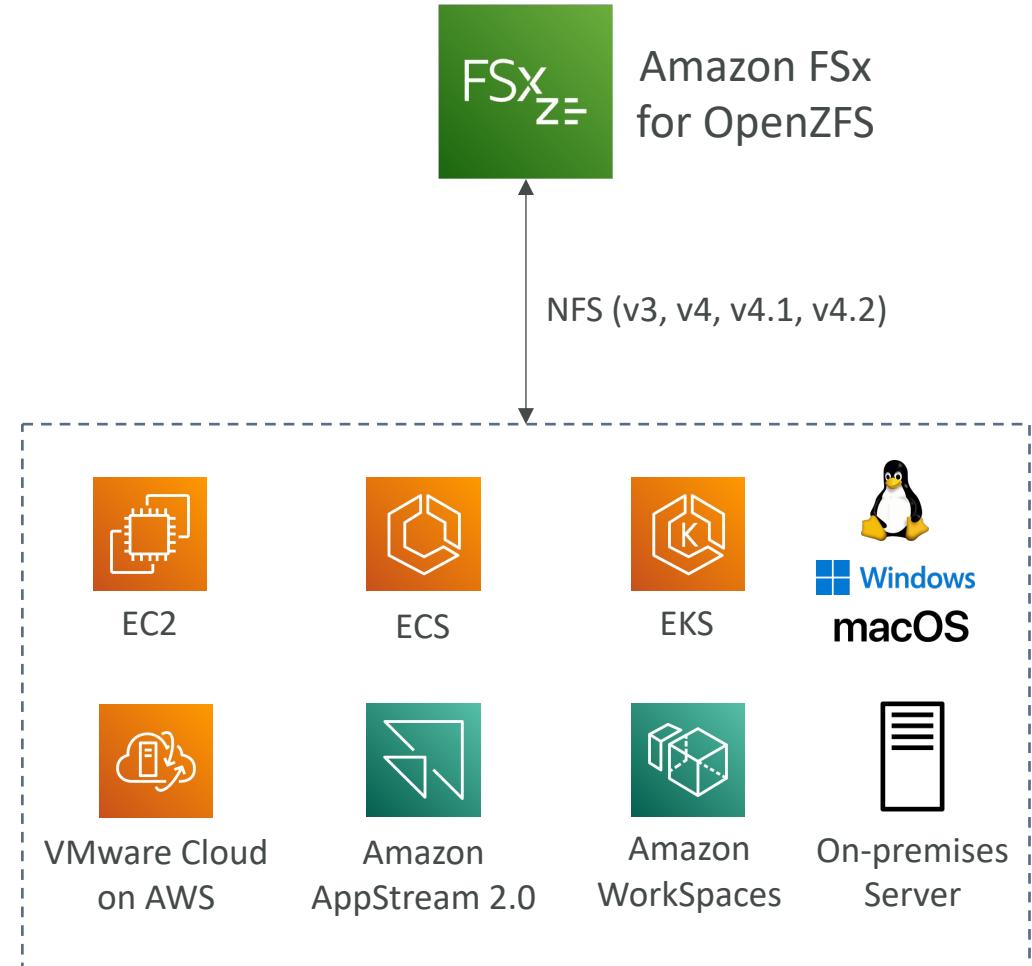
- Managed NetApp ONTAP on AWS
- File System compatible with NFS, SMB, iSCSI protocol
- Move workloads running on ONTAP or NAS to AWS
- Works with:
  - Linux
  - Windows
  - MacOS
  - VMware Cloud on AWS
  - Amazon Workspaces & AppStream 2.0
  - Amazon EC2, ECS and EKS
- Storage shrinks or grows automatically
- Snapshots, replication, low-cost, compression and data de-duplication
- Point-in-time instantaneous cloning (helpful for testing new workloads)



# Amazon FSx for OpenZFS



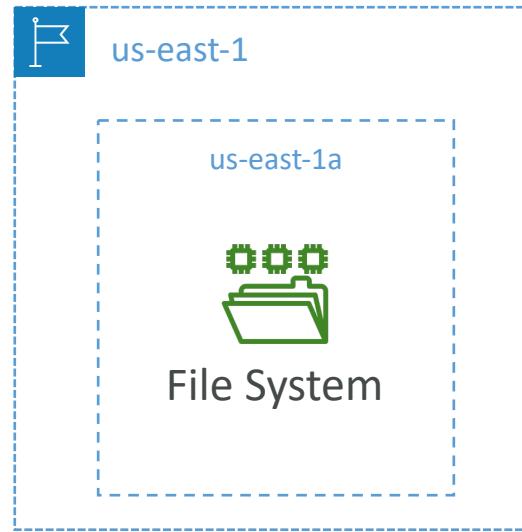
- Managed OpenZFS file system on AWS
- File System compatible with NFS (v3, v4, v4.1, v4.2)
- Move workloads running on ZFS to AWS
- Works with:
  - Linux
  - Windows
  - MacOS
  - VMware Cloud on AWS
  - Amazon Workspaces & AppStream 2.0
  - Amazon EC2, ECS and EKS
- Up to 1,000,000 IOPS with < 0.5ms latency
- Snapshots, compression and low-cost
- **Point-in-time instantaneous cloning (helpful for testing new workloads)**



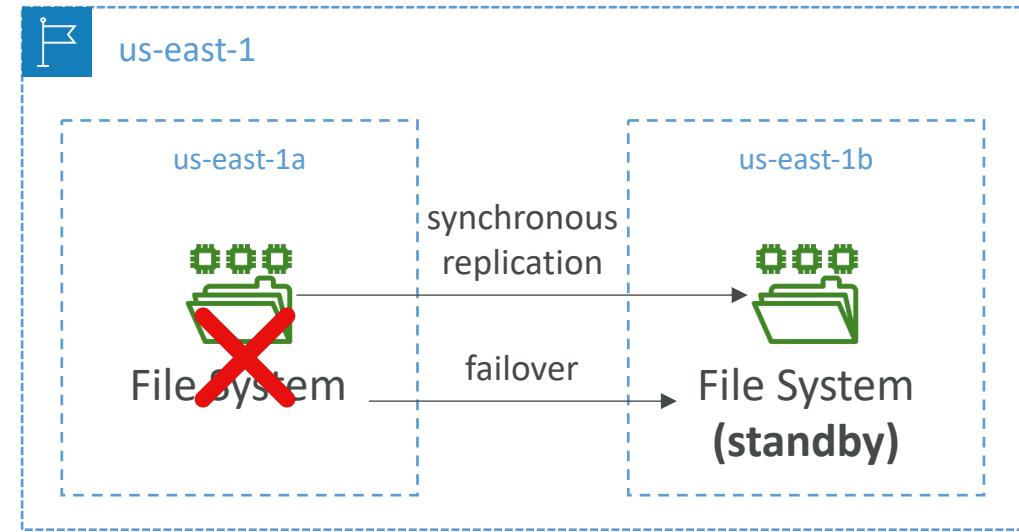
# FSx for SysOps



- **FSx for Windows – Single-AZ**
  - Automatically replicates data within an AZ
  - Two generations: Single-AZ 1 (SSD), Single-AZ 2 (SSD & HDD)



- **FSx for Windows – Multi-AZ**
  - Automatically replicates data across AZs (synchronous)
  - Standby file server in a different AZ (automatic failover)



# Hybrid Cloud for Storage

- AWS is pushing for "hybrid cloud"
  - Part of your infrastructure is on the cloud
  - Part of your infrastructure is on-premises
- 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-premises?
- AWS Storage Gateway!

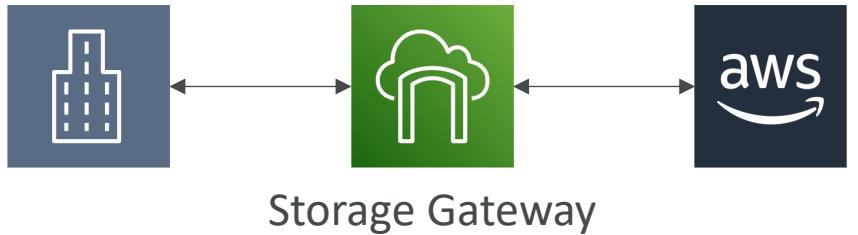
# AWS Storage Cloud Native Options



# AWS Storage Gateway

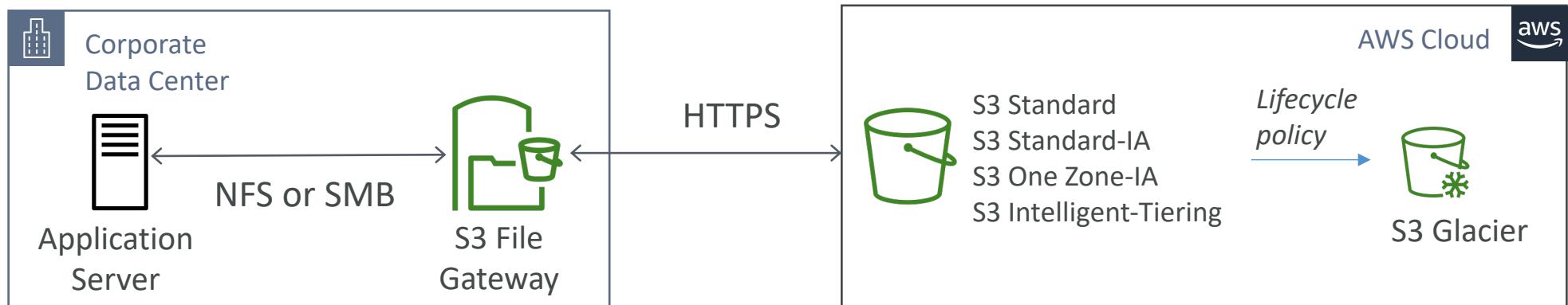


- Bridge between on-premises data and cloud data
- **Use cases:**
  - disaster recovery
  - backup & restore
  - tiered storage
  - on-premises cache & low-latency files access
- Types of Storage Gateway:
  - S3 File Gateway
  - Volume Gateway
  - Tape Gateway



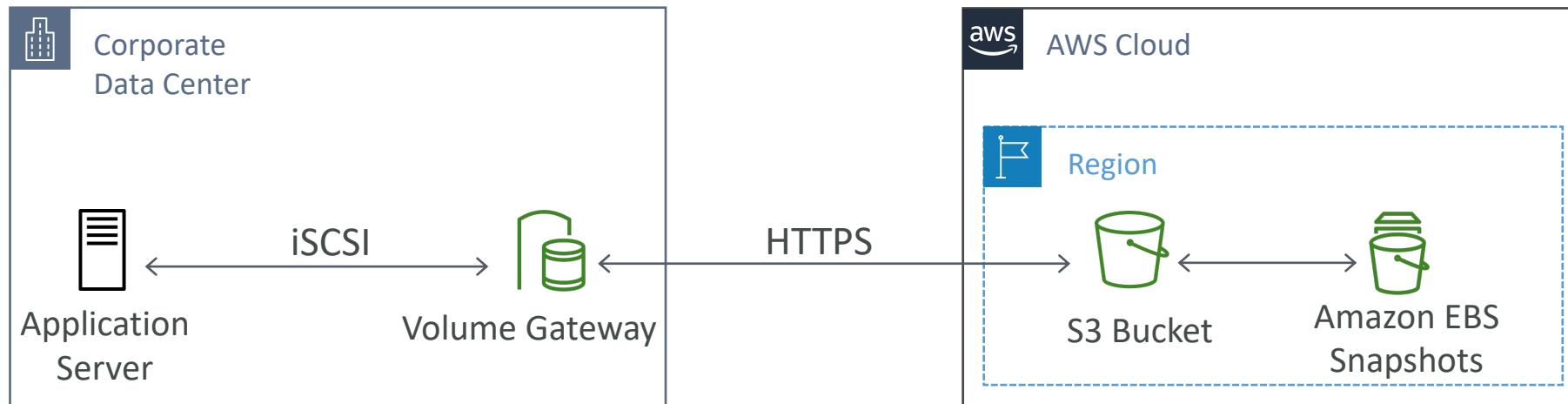
# Amazon S3 File Gateway

- Configured S3 buckets are accessible using the NFS and SMB protocol
- Most recently used data is cached in the file gateway
- Supports S3 Standard, S3 Standard IA, S3 One Zone A, S3 Intelligent Tiering
- Transition to S3 Glacier using a Lifecycle Policy
- Bucket access using IAM roles for each File Gateway
- SMB Protocol has integration with Active Directory (AD) for user authentication



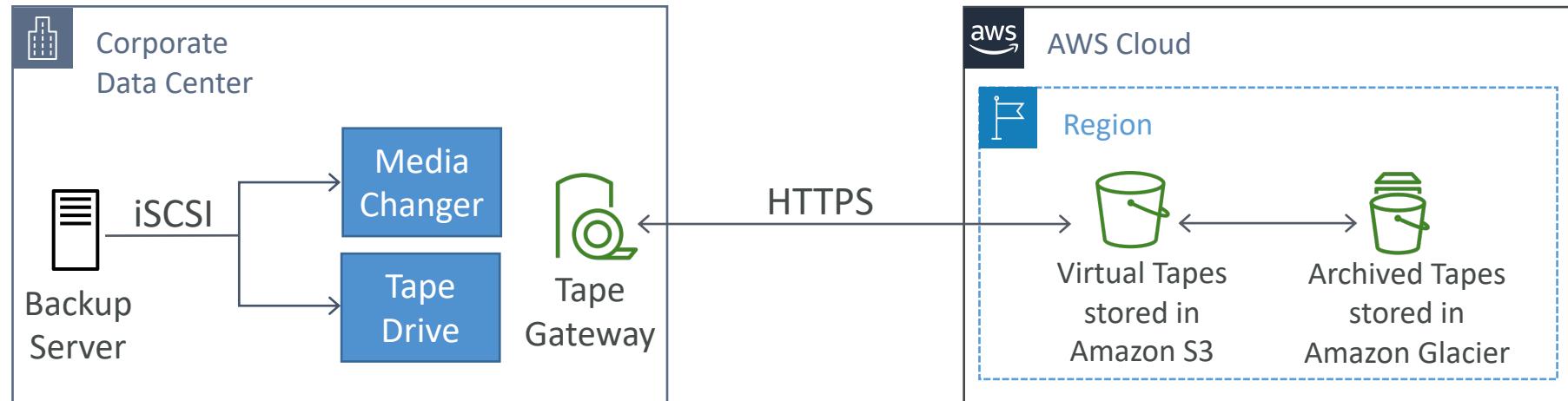
# Volume Gateway

- Block storage using iSCSI protocol backed by S3
- Backed by EBS snapshots which can help restore on-premises volumes!
- **Cached volumes:** low latency access to most recent data
- **Stored volumes:** entire dataset is on premise, scheduled backups to S3

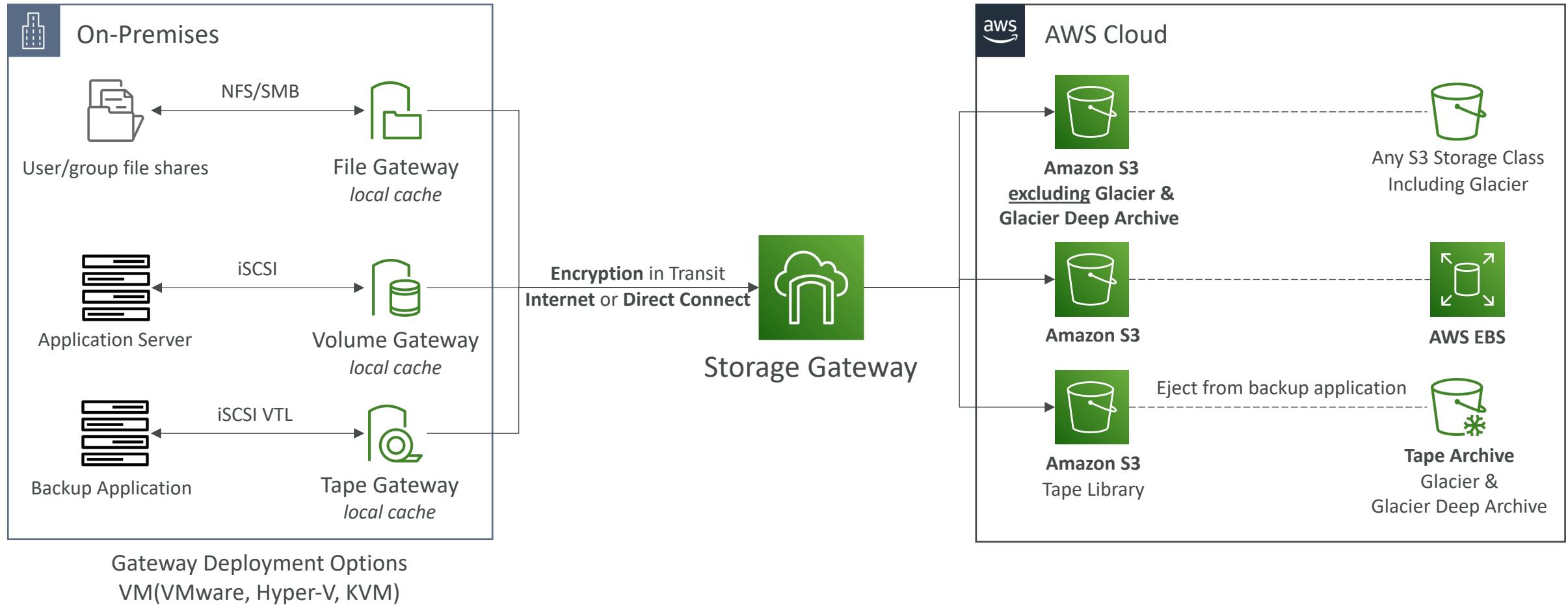


# Tape Gateway

- Some companies have backup processes using physical tapes (!)
- With Tape Gateway, companies use the same processes but, in the cloud
- Virtual Tape Library (VTL) backed by Amazon S3 and Glacier
- Back up data using existing tape-based processes (and iSCSI interface)
- Works with leading backup software vendors



# AWS Storage Gateway



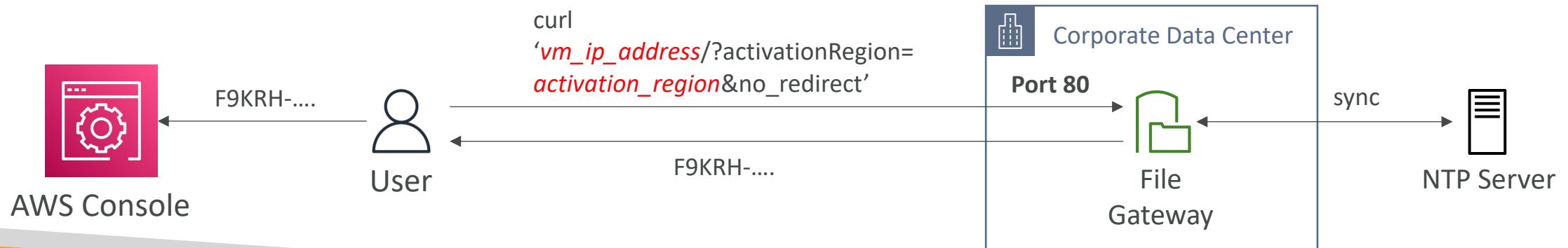
# Storage Gateway – SysOps

- File Gateway is POSIX compliant (Linux file system)
  - POSIX metadata ownership, permissions, and timestamps stored in the object's metadata in S3
- Reboot Storage Gateway VM: (e.g., maintenance)
  - **File Gateway:** simply restart the Storage Gateway VM
  - **Volume and Tape Gateway:**
    - Stop Storage Gateway Service (AWS Console, VM local Console, Storage Gateway API)
    - Reboot the Storage Gateway VM
    - Start Storage Gateway Service (AWS Console, VM local Console, Storage Gateway API)

# Storage Gateway – Activations

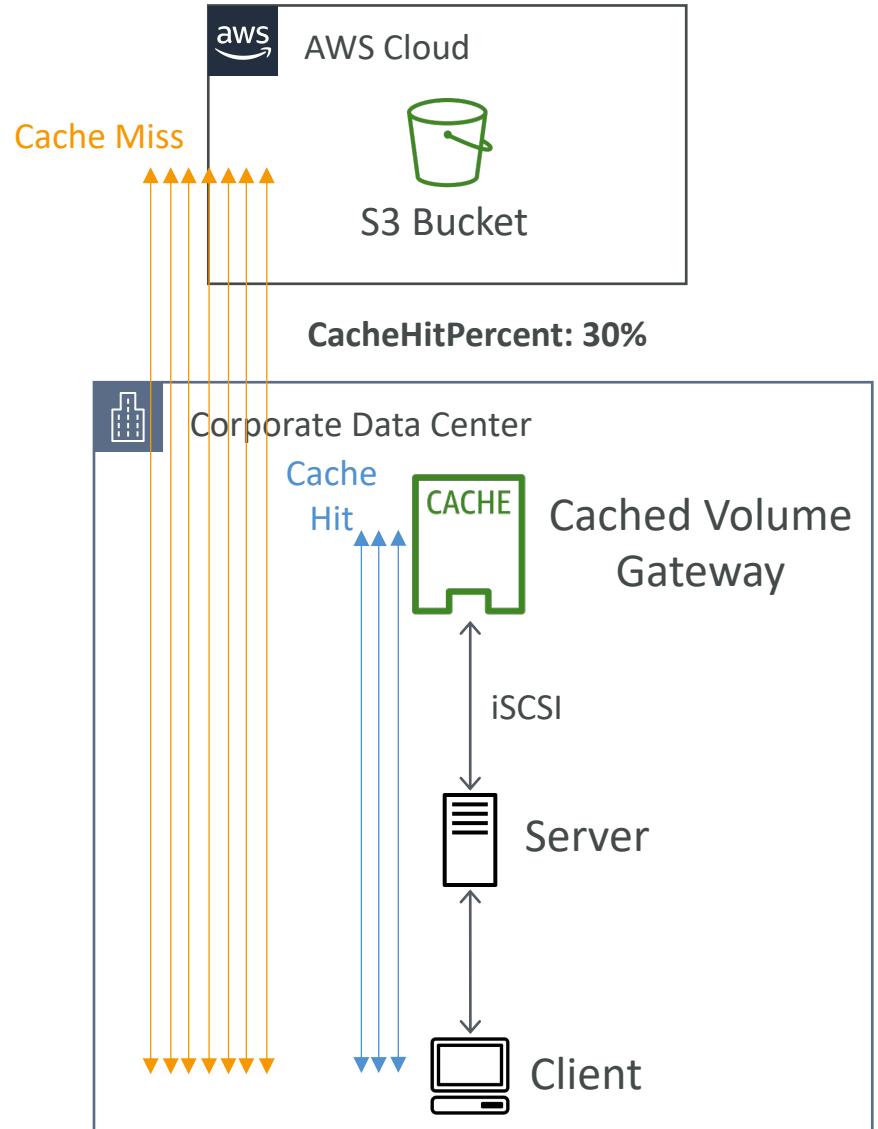
- Two ways to get Activation Key:
  - Using the Gateway VM CLI
  - Make a web request to the Gateway VM (Port 80)
- Troubleshooting Activation Failures:
  - Make sure the Gateway VM has port 80 opened
  - Check that the Gateway VM has the correct time and synchronizing its time automatically to a Network Time Protocol (NTP) server

```
AWS Appliance Activation - Configuration
#####
##  Currently connected network adapters:
##
##  eth0:
#####
1: HTTP/SOCKS Proxy Configuration
2: Network Configuration
3: Test Network Connectivity
4: View System Resource Check (0 Errors)
5: License Information
6: Command Prompt
0: Get activation key
Press "x" to exit session
Enter command: █
```



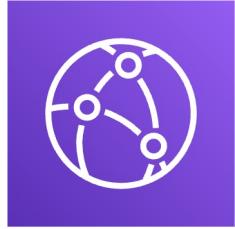
# Storage Gateway – Volume Gateway Cache

- Cached mode: only the most recent data is stored
- Looking at cache efficiency
  - Look at the CacheHitPercent metric (you want it to be high)
  - Look at the CachePercentUsed (you don't want it to be too high)
- Create a larger cache disk
  - Use the cached volume to clone a new volume of a larger size
  - Select the new disk as the cached volume

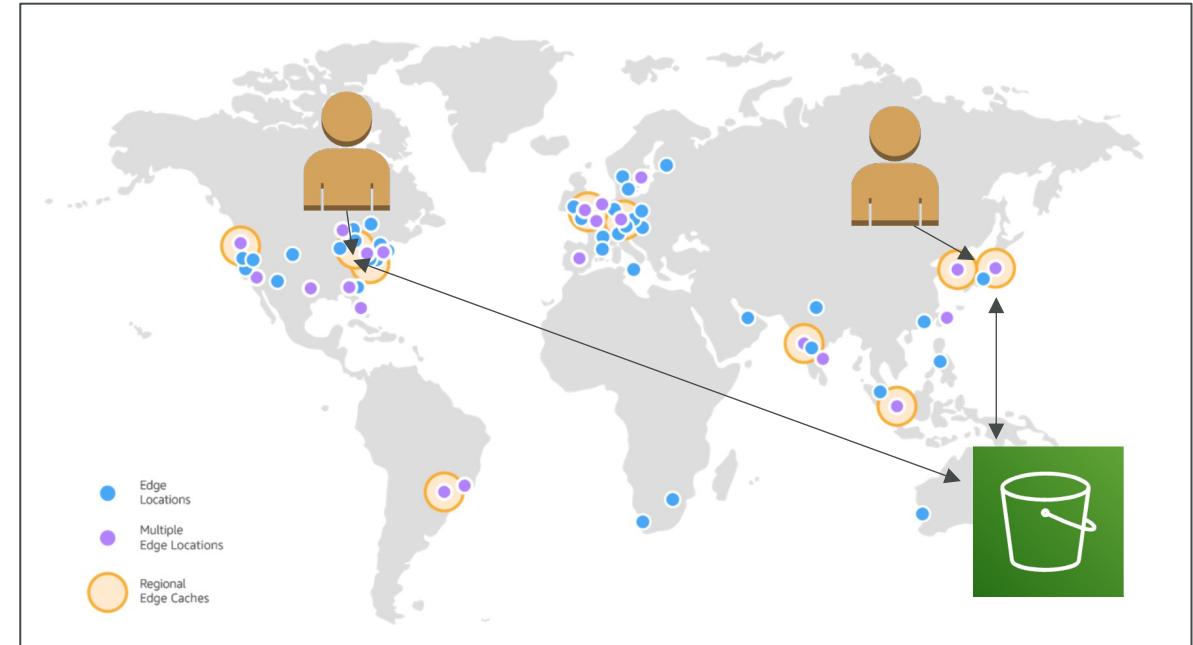


# Amazon CloudFront

# Amazon CloudFront



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

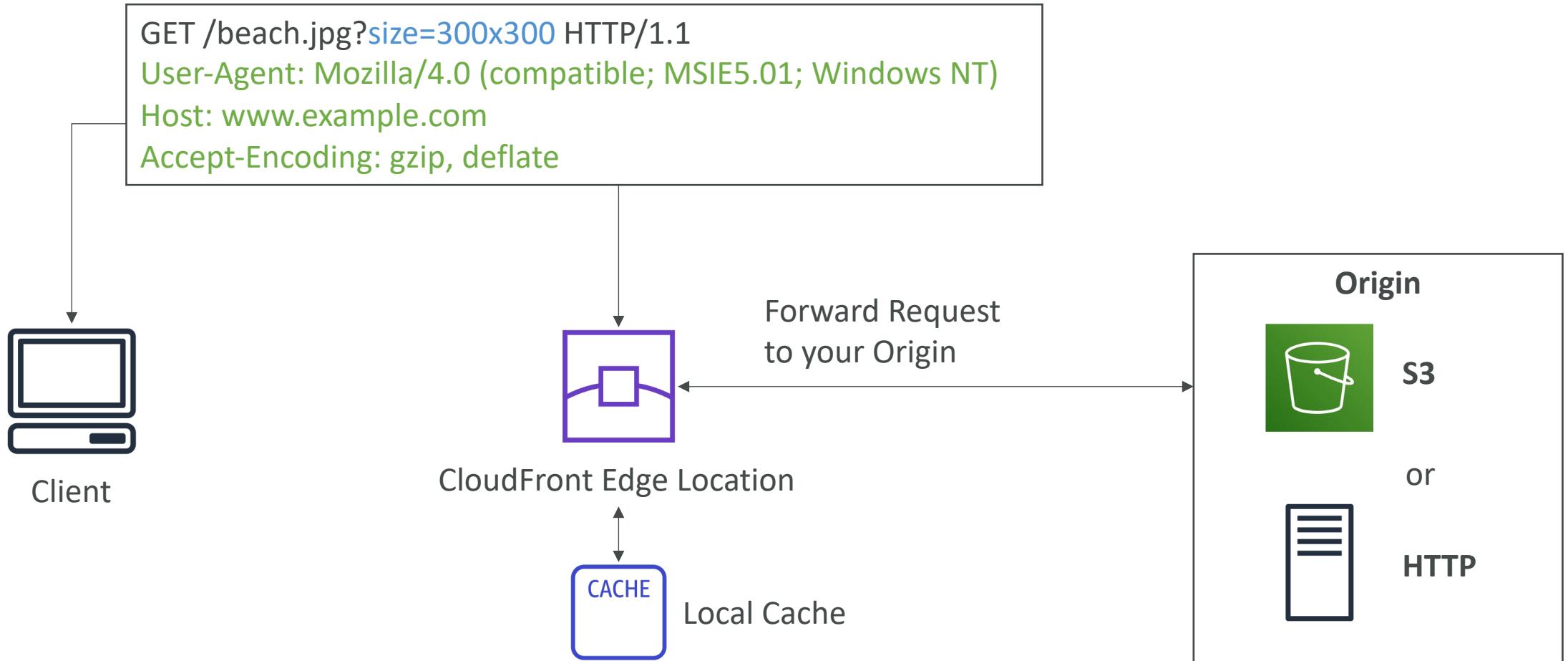


Source: <https://aws.amazon.com/cloudfront/features/?nc=sn&loc=2>

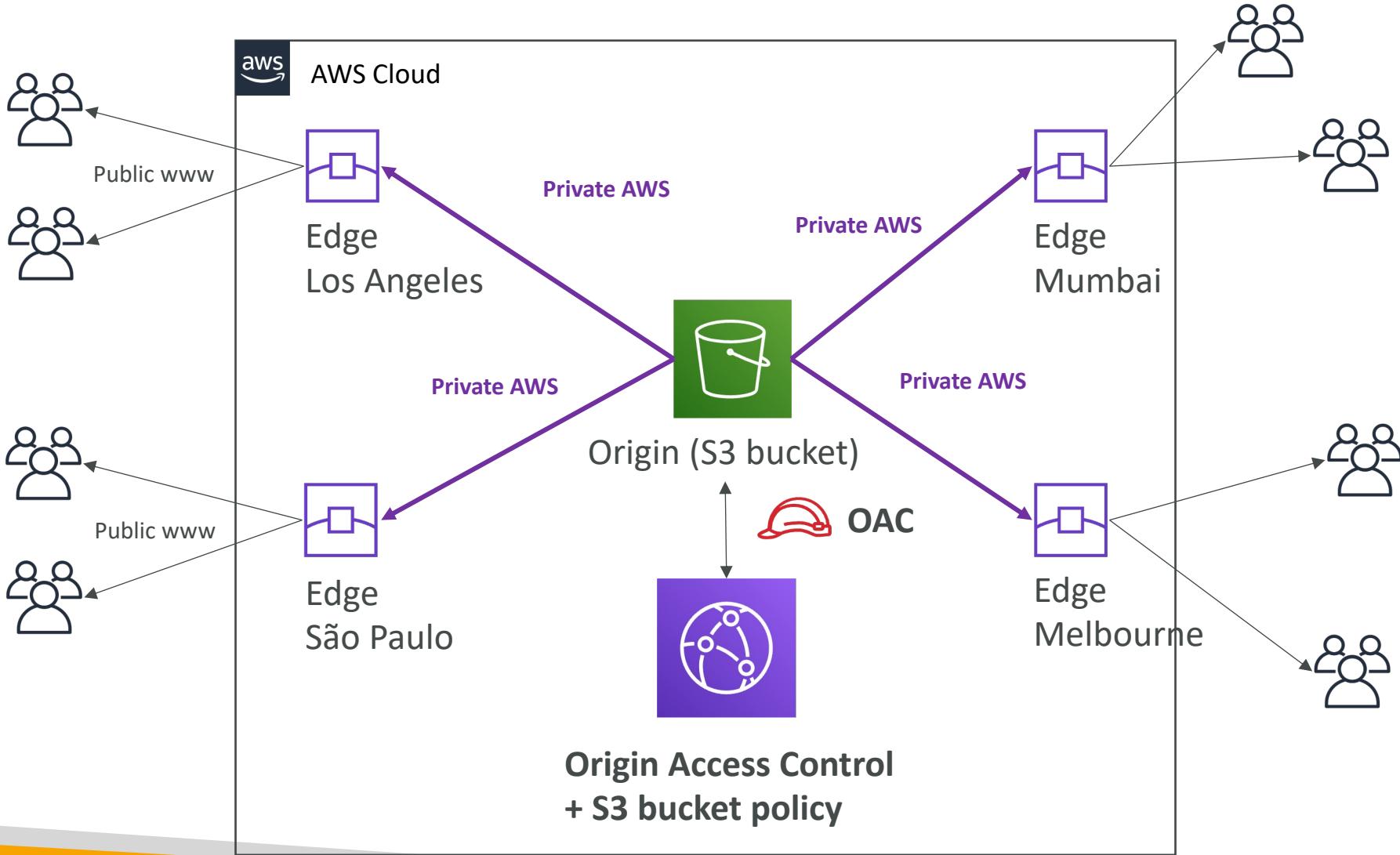
# CloudFront – Origins

- **S3 bucket**
  - For distributing files and caching them at the edge
  - For uploading files to S3 through CloudFront
  - Secured using Origin Access Control (OAC)
- **VPC Origin**
  - For applications hosted in VPC private subnets
  - Application Load Balancer / Network Load Balancer / EC2 Instances
- **Custom Origin (HTTP)**
  - S3 website (must first enable the bucket as a static S3 website)
  - Any public HTTP backend you want

# CloudFront at a high level



# CloudFront – S3 as an Origin

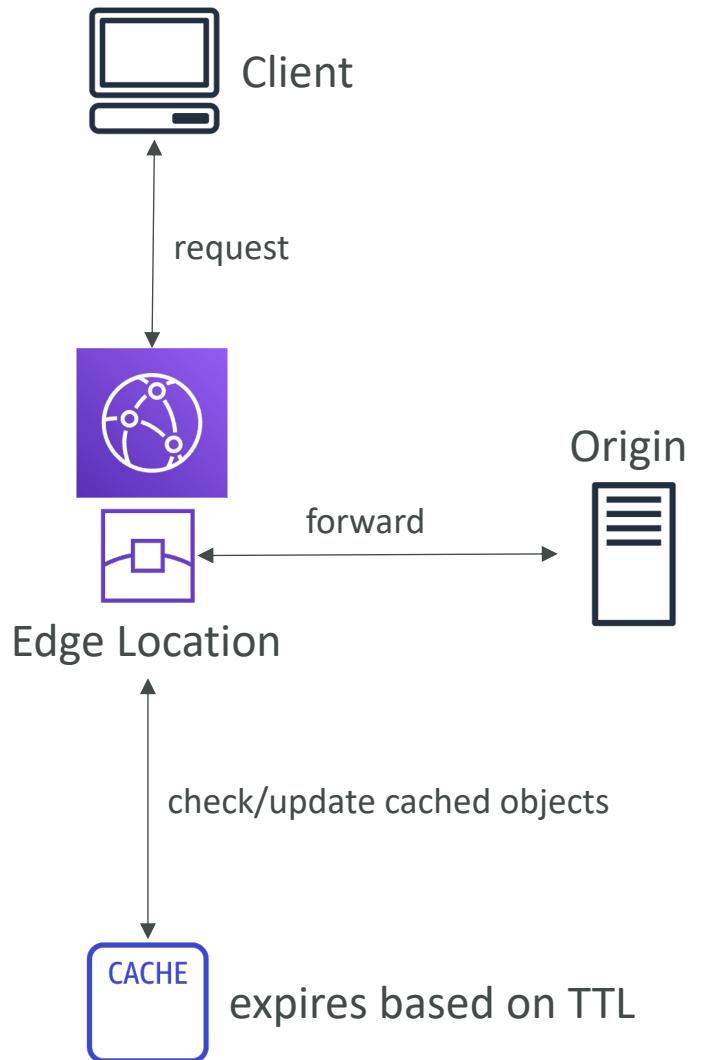


# CloudFront vs S3 Cross Region Replication

- 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

# CloudFront Caching

- The cache lives at each CloudFront Edge Location
- CloudFront identifies each object in the cache using the Cache Key
  - A unique identifier for each object in the cache
- You want to maximize the Cache Hit ratio to minimize requests to the origin
- You can invalidate part of the cache using the CreateInvalidation API

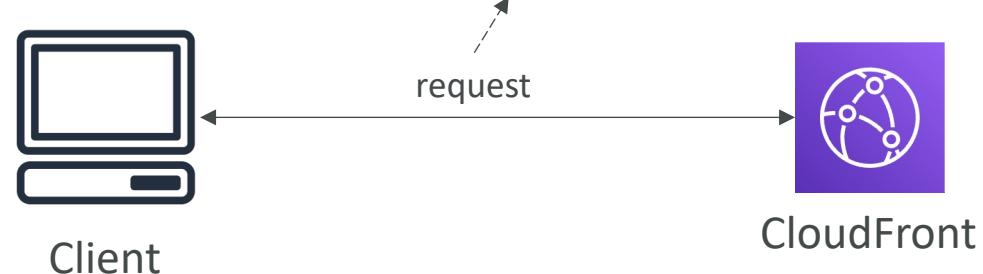


# CloudFront Caching – Cache Policy

- Cache based on:
  - **HTTP Headers:** None – Whitelist
  - **Cookies:** None – Whitelist – Include All – Except – All
  - **Query Strings:** None – Whitelist – Include All-Except – All
- Control the TTL (0 seconds to 1 year), can be set by the origin using the **Cache-Control** header, **Expires** header...
- Create your own policy or use Predefined Managed Policies

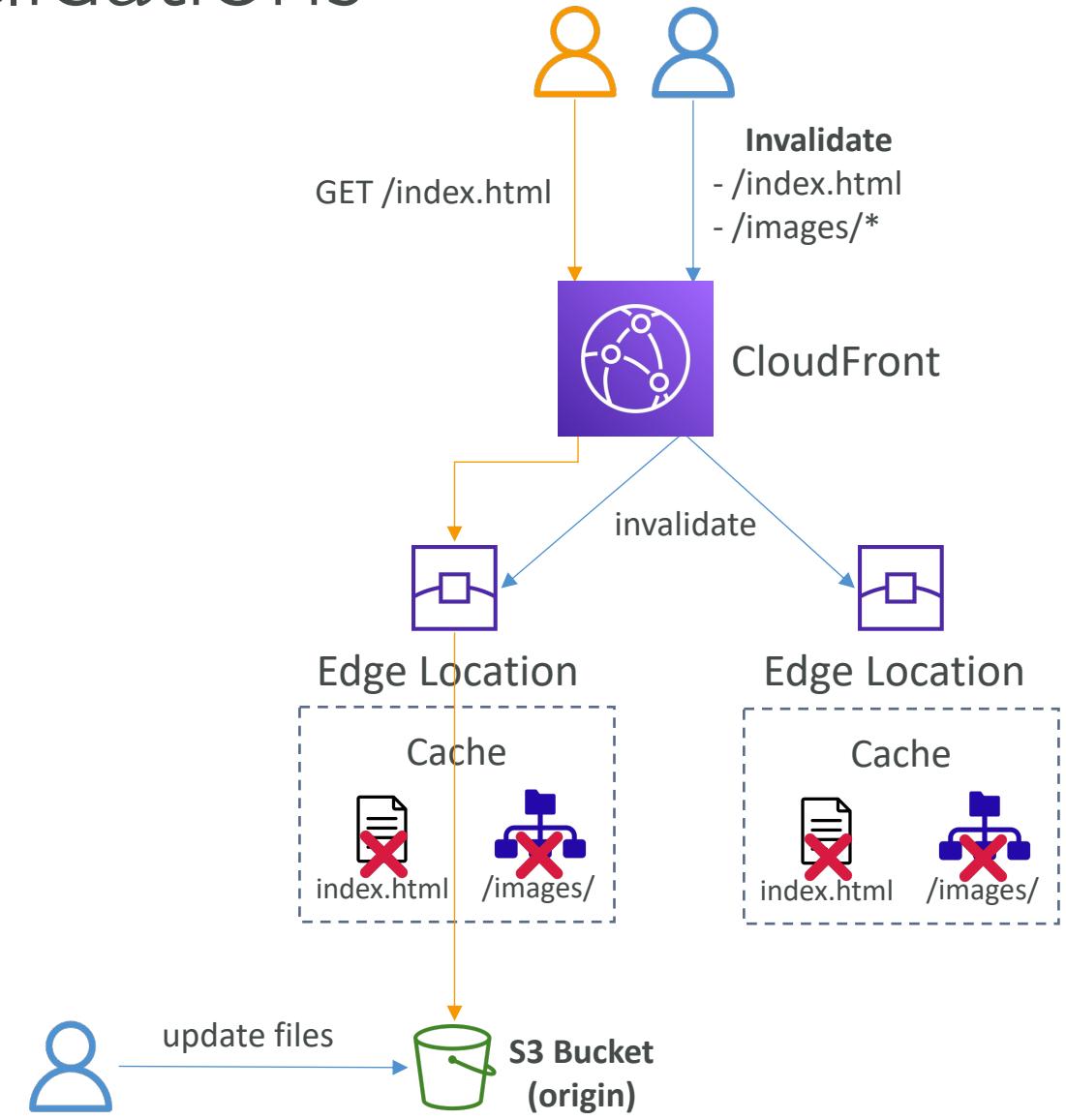
## Cache Policy using HTTP Headers

```
GET /blogs/myblog.html HTTP/1.1
Host: mywebsite.com
User-Agent: Mozilla/5.0 (Mac OS X 10_15_2....)
Date: Tue, 28 Jan 2021 17:01:57 GMT
Authorization: SAPSIDHASH fdd00ecee39fe....
Keep-Alive: 300
Language: fr-fr
```



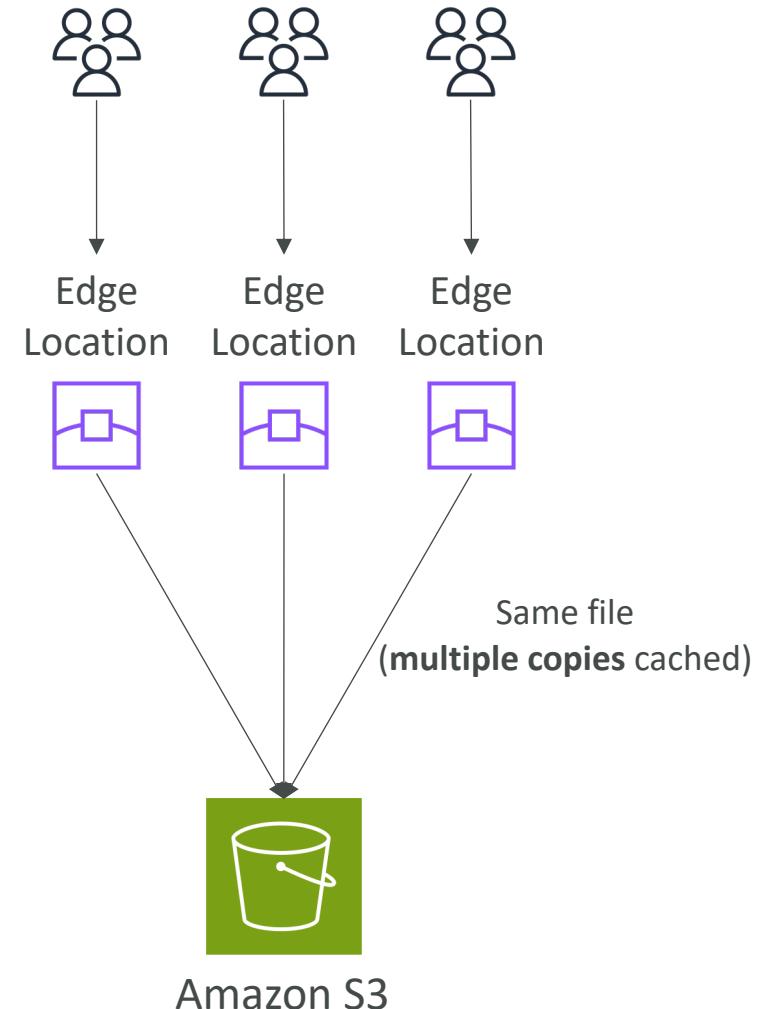
# CloudFront – Cache Invalidations

- In case you update the back-end origin, CloudFront doesn't know about it and will only get the refreshed content after the TTL has expired
- However, you can force an entire or partial cache refresh (thus bypassing the TTL) by performing a **CloudFront Invalidation**
- You can invalidate all files (\*) or a special path (/images/\*)



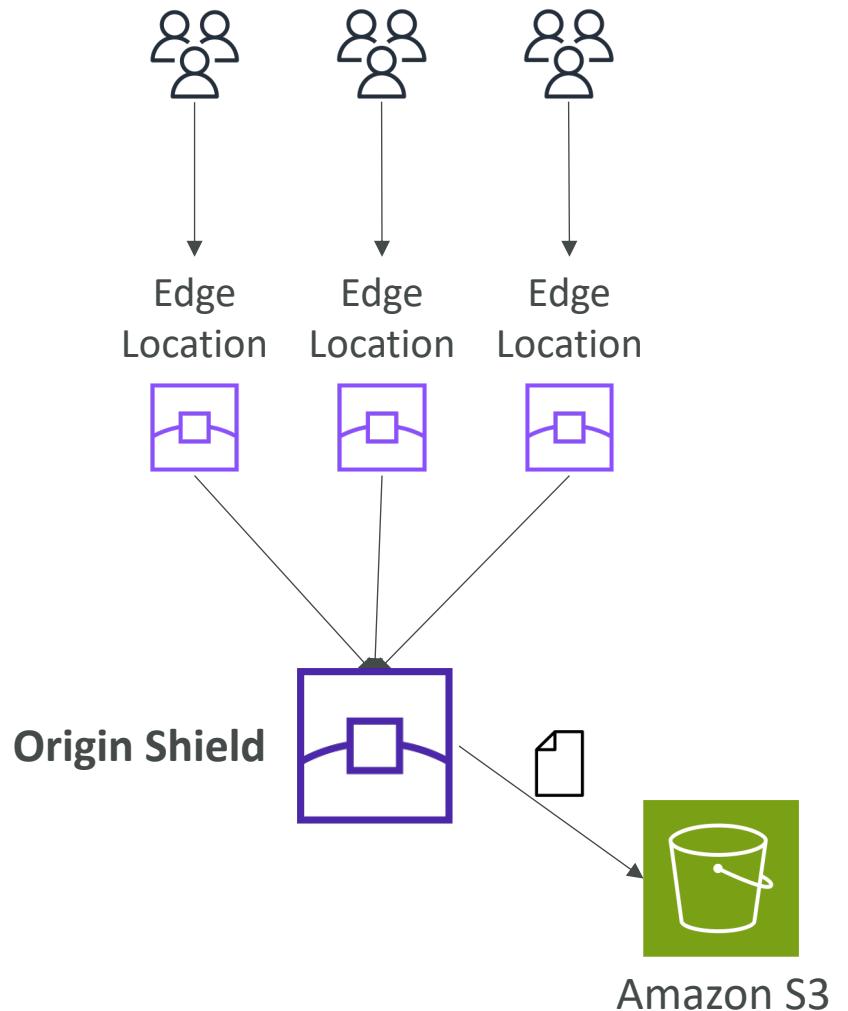
# CloudFront – without Origin Shield

- CloudFront distributions are accessed through multiple Edge Locations
- These Edge Locations may perform the same requests to the Origin
- Your Origin can be overload and its costs increase



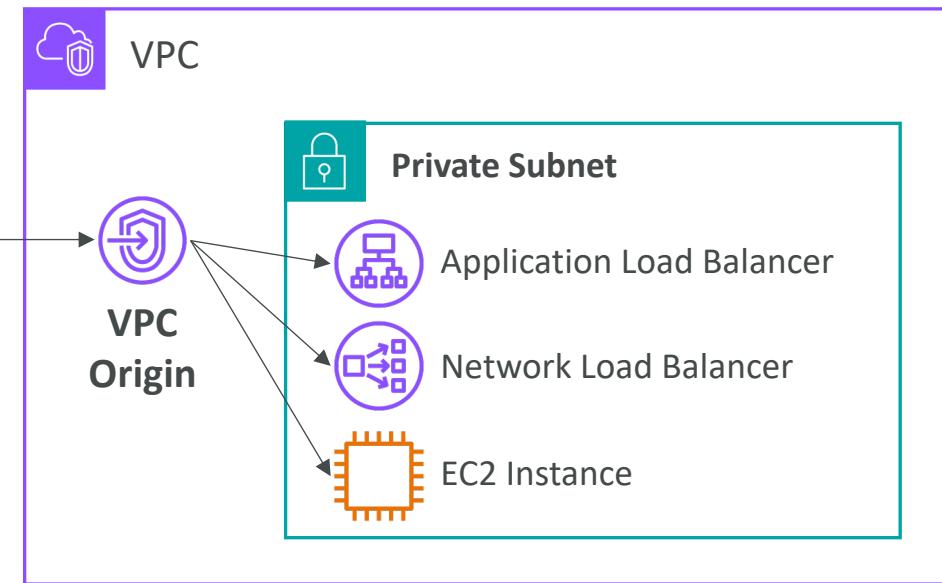
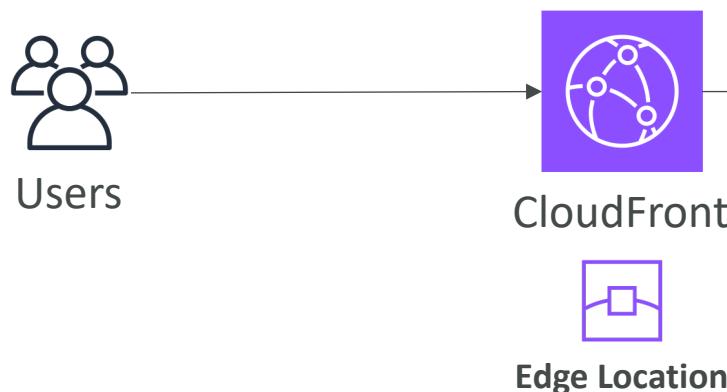
# CloudFront – with Origin Shield

- **Origin Shield** – extra caching layer that helps minimize your origin's load and improve its availability
- Resides between Regional Edge Location and your Origin
- Combine multiple requests for the same object into a single request (reduce origin load)

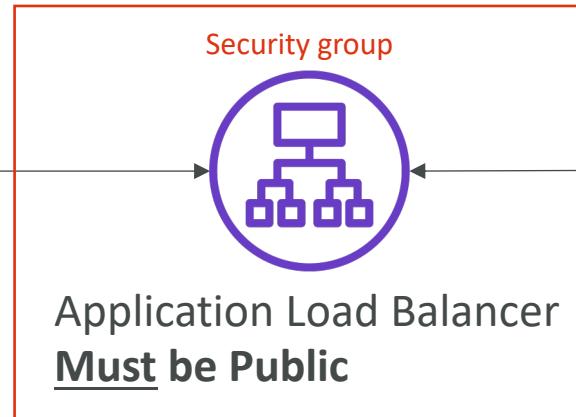
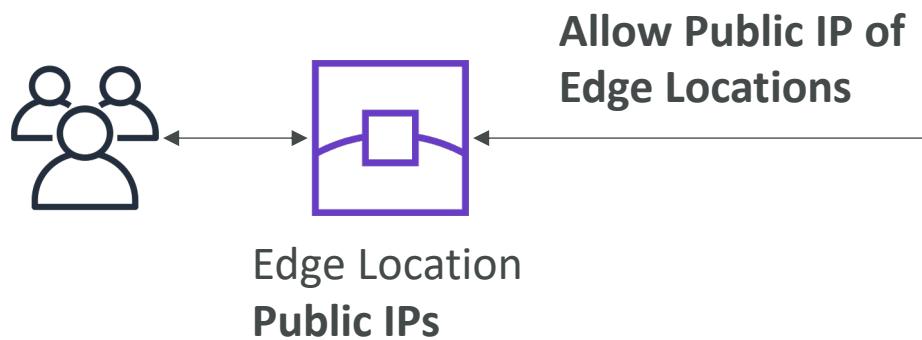
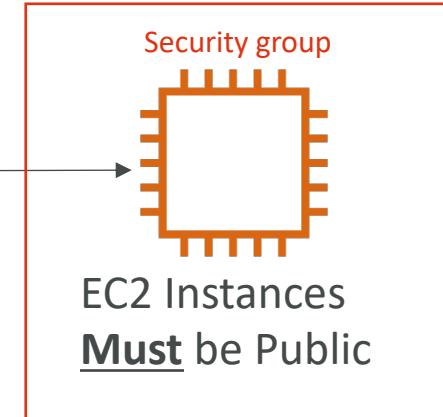
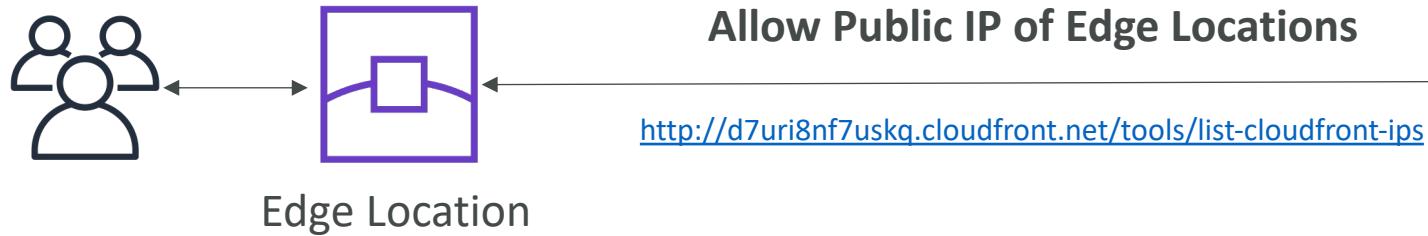


# CloudFront – ALB or EC2 as an origin Using VPC Origins

- Allows you to deliver content from your applications hosted in your VPC private subnets (no need to expose them on the Internet)
- Deliver traffic to **private**:
  - Application Load Balancer
  - Network Load Balancer
  - EC2 Instances



# CloudFront – ALB or EC2 as an origin Using Public Network

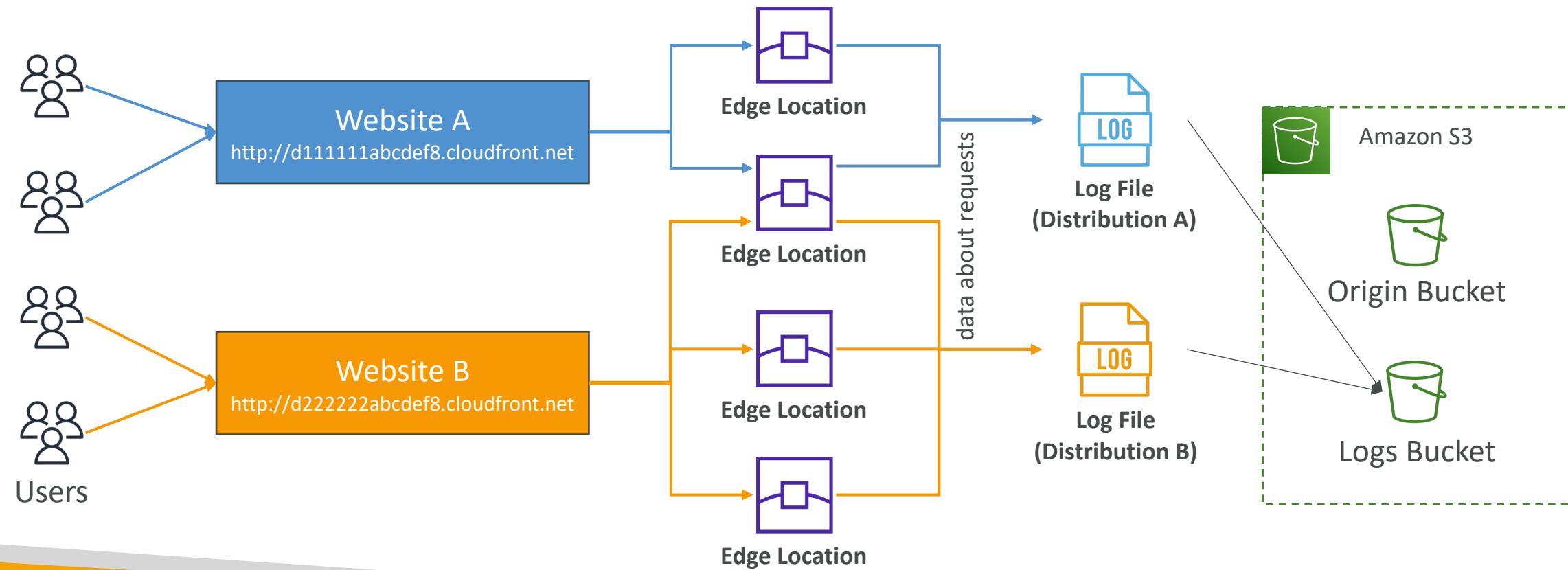


# CloudFront Geo Restriction

- You can restrict who can access your distribution
  - **Allowlist:** Allow your users to access your content only if they're in one of the countries on a list of approved countries.
  - **Blocklist:** Prevent your users from accessing your content if they're in one of the countries on a list of banned countries.
- The “country” is determined using a 3<sup>rd</sup> party Geo-IP database
- Use case: Copyright Laws to control access to content

# CloudFront Access Logs

- Logs every request made to CloudFront into a logging S3 bucket
- Can send logs to CloudWatch Logs and Kinesis Data Firehose



# CloudFront Reports

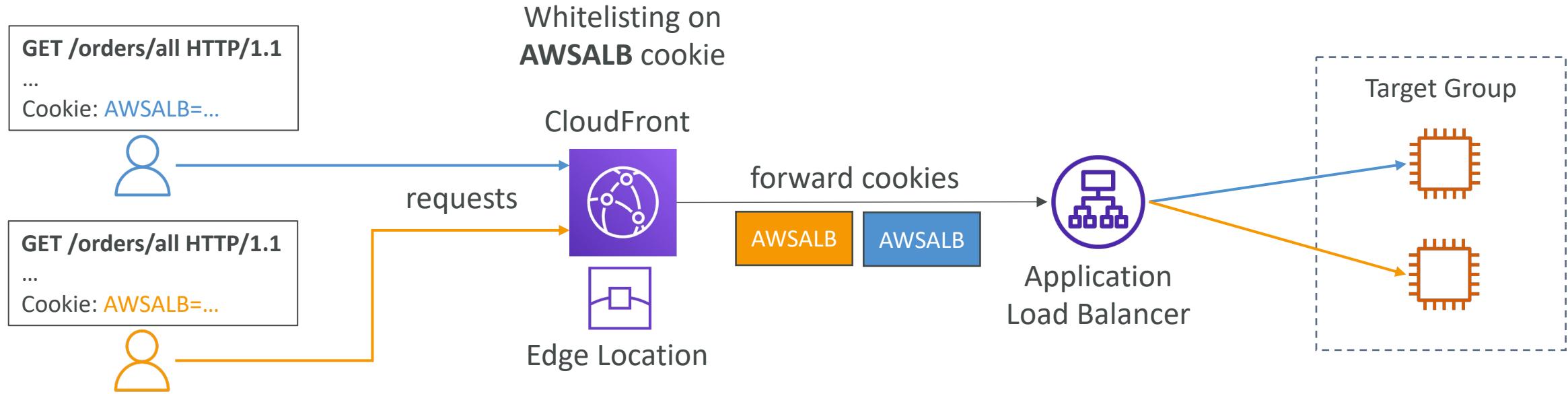
- It's possible to generate reports on:
  - Cache Statistics Report
  - Popular Objects Report
  - Top Referrers Report
  - Usage Reports
  - Viewers Report
- These reports are based on the data from the Access Logs.

# CloudFront troubleshooting

- CloudFront caches HTTP 4xx and 5xx status codes returned by S3 ( or the origin server)
- 4xx error code indicates that user doesn't have access to the underlying bucket (403) or the object user is requesting is not found (404)
- 5xx error codes indicates Gateway issues

# CloudFront with ALB sticky sessions

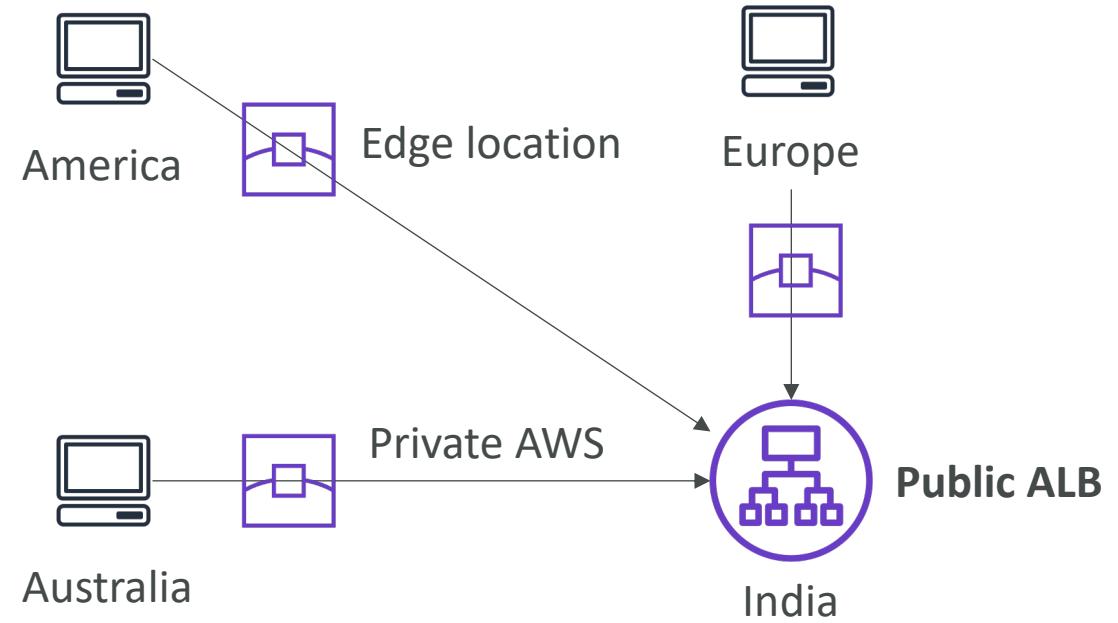
- Must forward / whitelist the cookie that controls the session affinity to the origin to allow the session affinity to work
- Set a TTL to a value lesser than when the authentication cookie expires



# AWS Global Accelerator



- Leverage the AWS internal network to route to your application
- 2 Anycast IP are created for your application
- The Anycast IP send traffic directly to Edge Locations
- The Edge locations send the traffic to your application



# AWS Global Accelerator

- Works with Elastic IP, EC2 instances, ALB, NLB, public or private
- Consistent Performance
  - Intelligent routing to lowest latency and fast regional failover
  - No issue with client cache (because the IP doesn't change)
  - Internal AWS network
- Health Checks
  - Global Accelerator performs a health check of your applications
  - Helps make your application global (failover less than 1 minute for unhealthy)
  - Great for disaster recovery (thanks to the health checks)
- Security
  - only 2 external IP need to be whitelisted
  - DDoS protection thanks to AWS Shield

# 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**
  - Improves performance for both cacheable content (such as images and videos)
  - Dynamic content (such as API acceleration and dynamic site delivery)
  - Content is served at the edge
- **Global Accelerator**
  - Improves performance for a wide range of applications over TCP or UDP
  - Proxying packets at the edge to applications running in one or more AWS Regions.
  - Good fit for non-HTTP use cases, such as gaming (UDP), IoT (MQTT), or Voice over IP
  - Good for HTTP use cases that require static IP addresses
  - Good for HTTP use cases that required deterministic, fast regional failover

# Databases in AWS

# Amazon RDS Overview



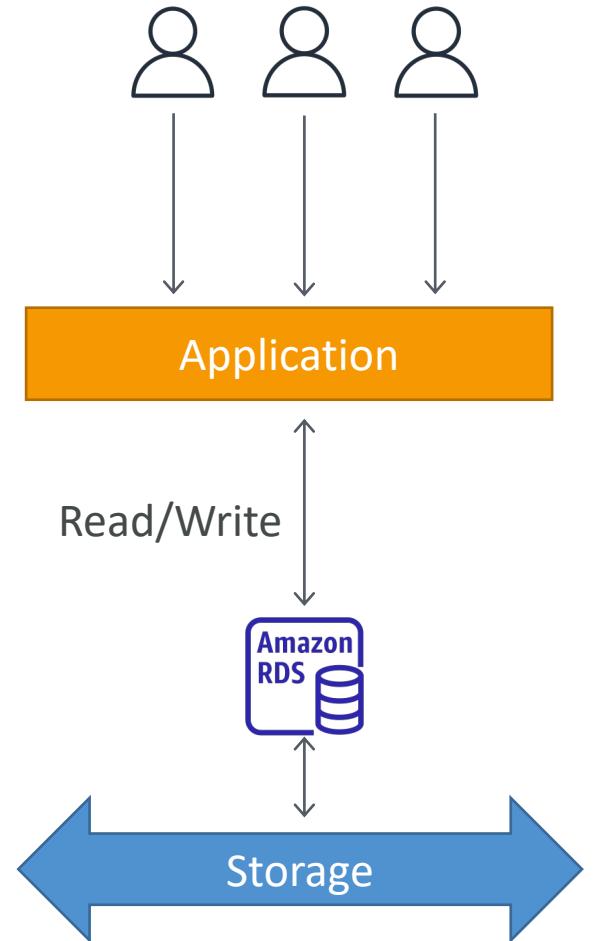
- RDS stands for Relational Database Service
- It's a managed DB service for DB use SQL as a query language.
- It allows you to create databases in the cloud that are managed by AWS
  - Postgres
  - MySQL
  - MariaDB
  - Oracle
  - Microsoft SQL Server
  - IBM DB2
  - Aurora (AWS Proprietary database)

# 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
- BUT you can't SSH into your instances

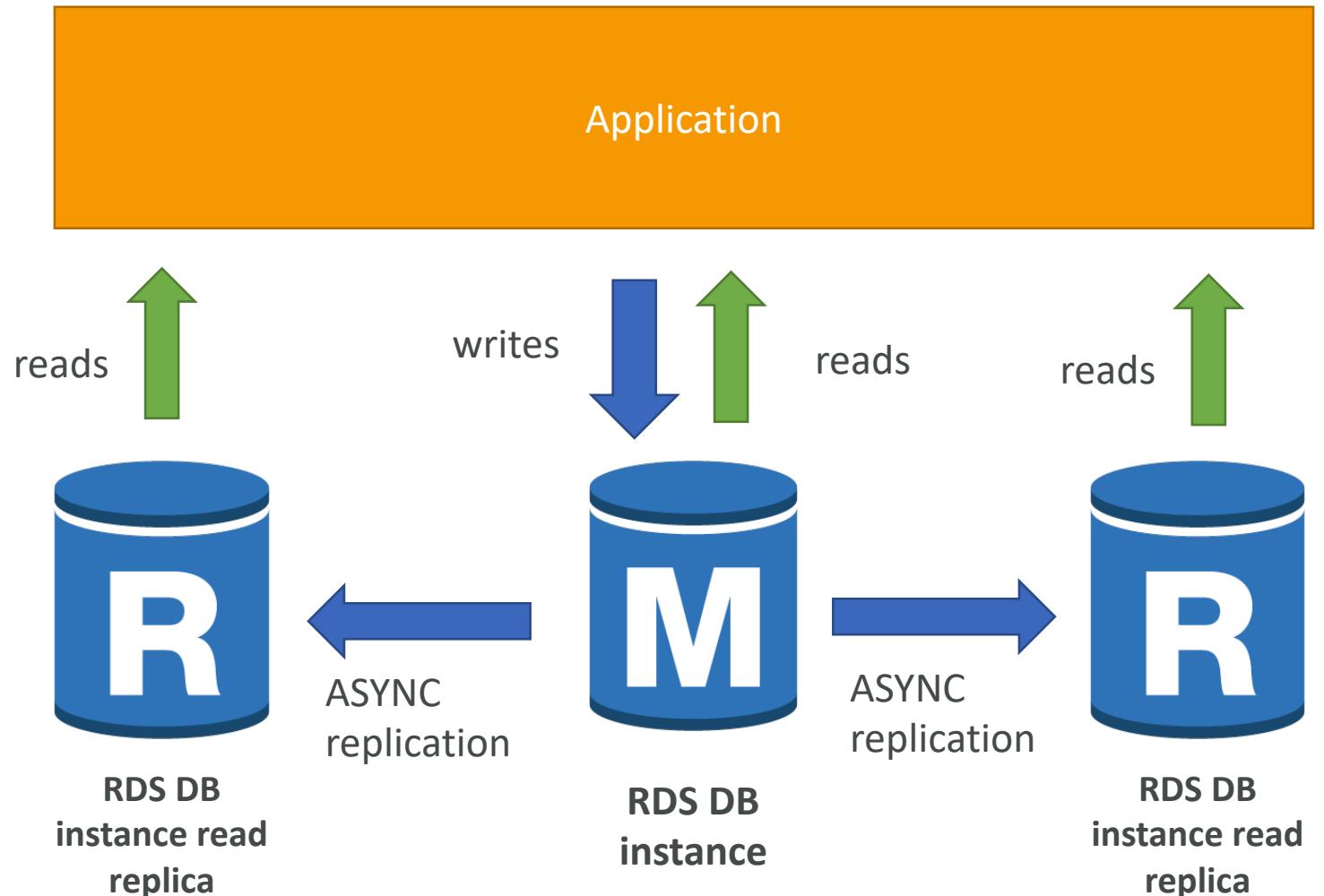
# RDS – Storage Auto Scaling

- Helps you increase storage on your RDS DB instance dynamically
- When RDS detects you are running out of free database storage, it scales automatically
- Avoid manually scaling your database storage
- You have to set **Maximum Storage Threshold** (maximum limit for DB storage)
- Automatically modify storage if:
  - Free storage is less than 10% of allocated storage
  - Low-storage lasts at least 5 minutes
  - 6 hours have passed since last modification
- Useful for applications with **unpredictable workloads**
- Supports all RDS database engines



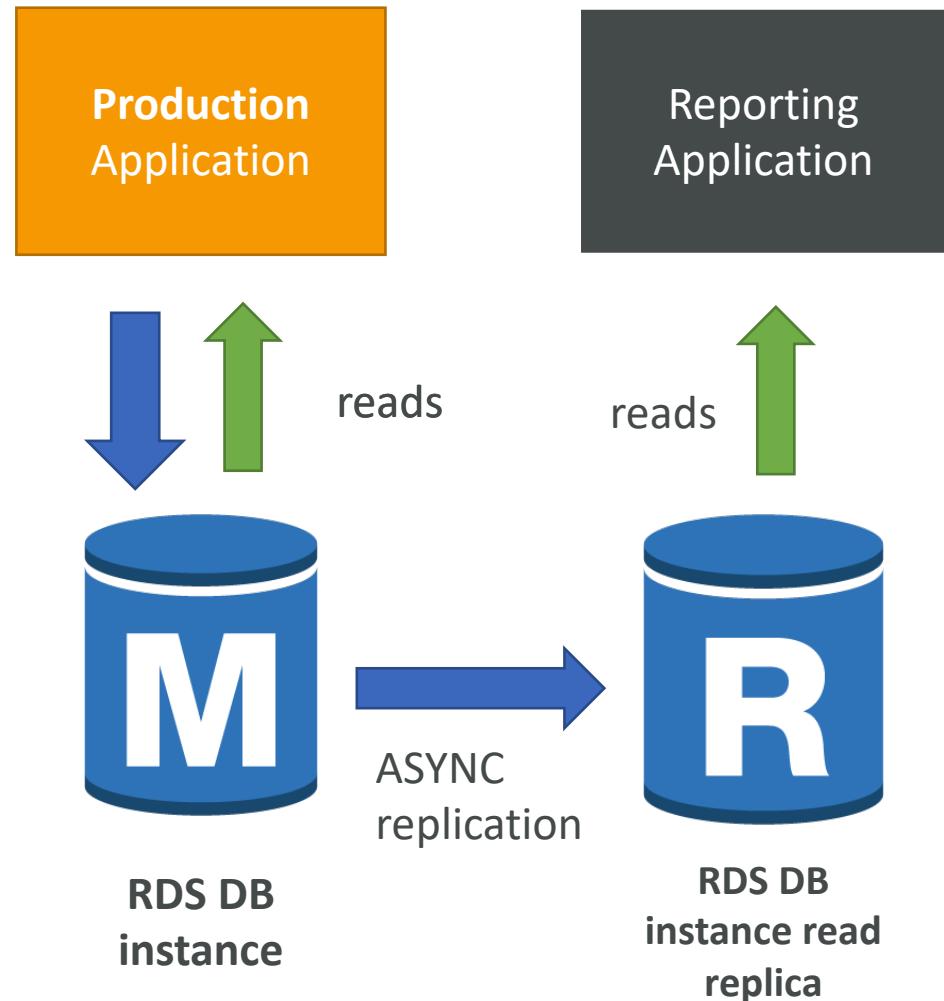
# RDS Read Replicas for read scalability

- Up to 15 Read Replicas
- Within AZ, Cross AZ or Cross Region
- Replication is **ASYNC**, so reads are eventually consistent
- Replicas can be promoted to their own DB
- Applications must update the connection string to leverage read replicas



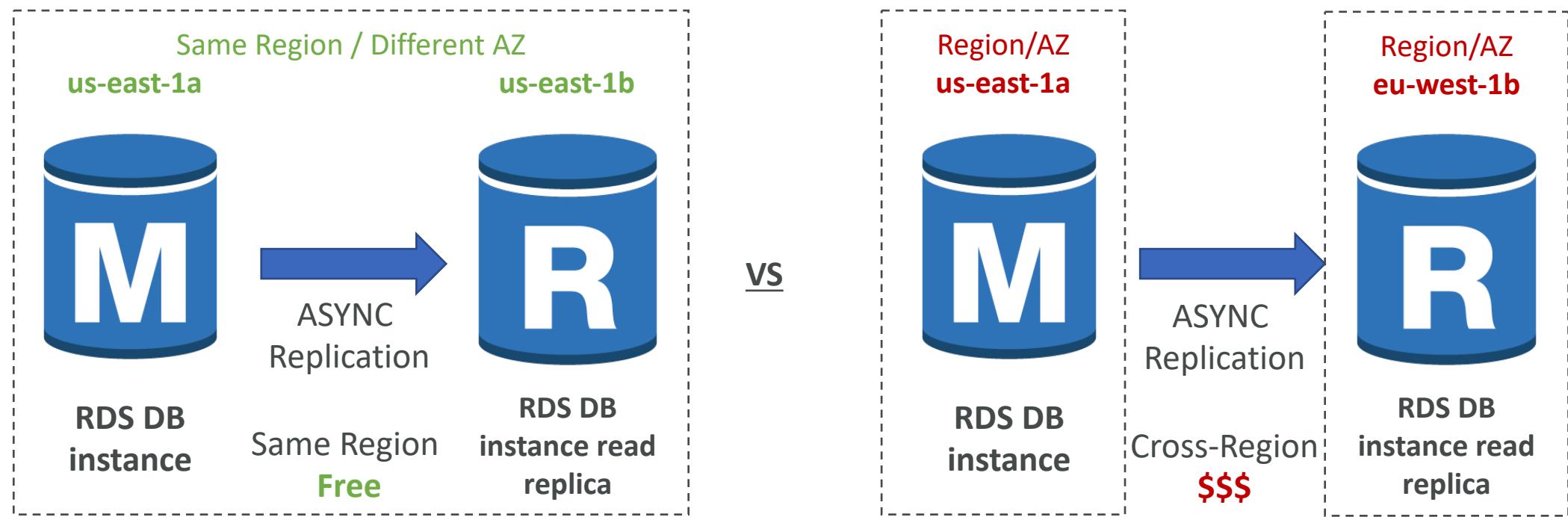
# RDS Read Replicas – Use Cases

- You have a production database that is taking on normal load
- You want to run a reporting application to run some analytics
- You create a Read Replica to run the new workload there
- The production application is unaffected
- Read replicas are used for SELECT (=read) only kind of statements (not INSERT, UPDATE, DELETE)



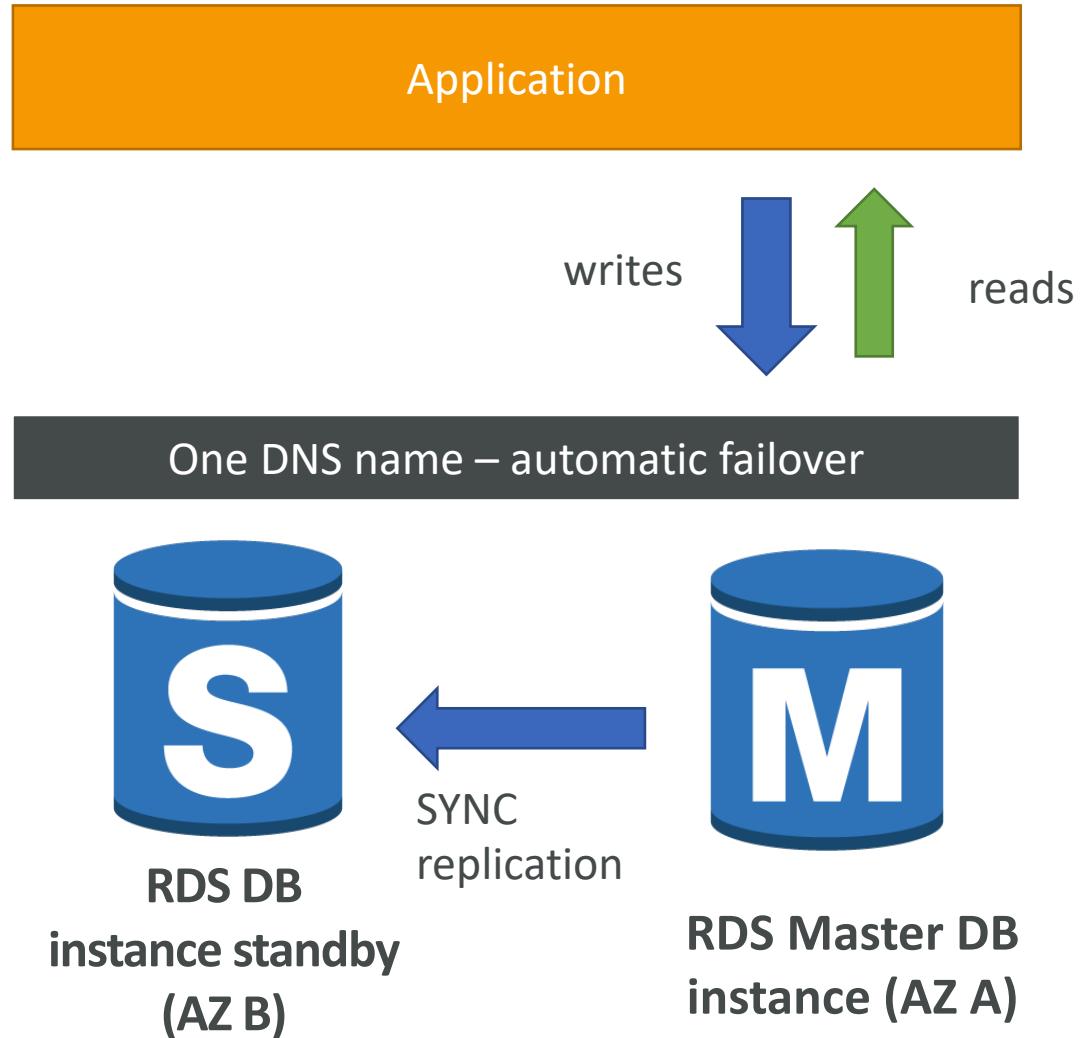
# RDS Read Replicas – Network Cost

- In AWS there's a network cost when data goes from one AZ to another
- For RDS Read Replicas within the same region, you don't pay that fee



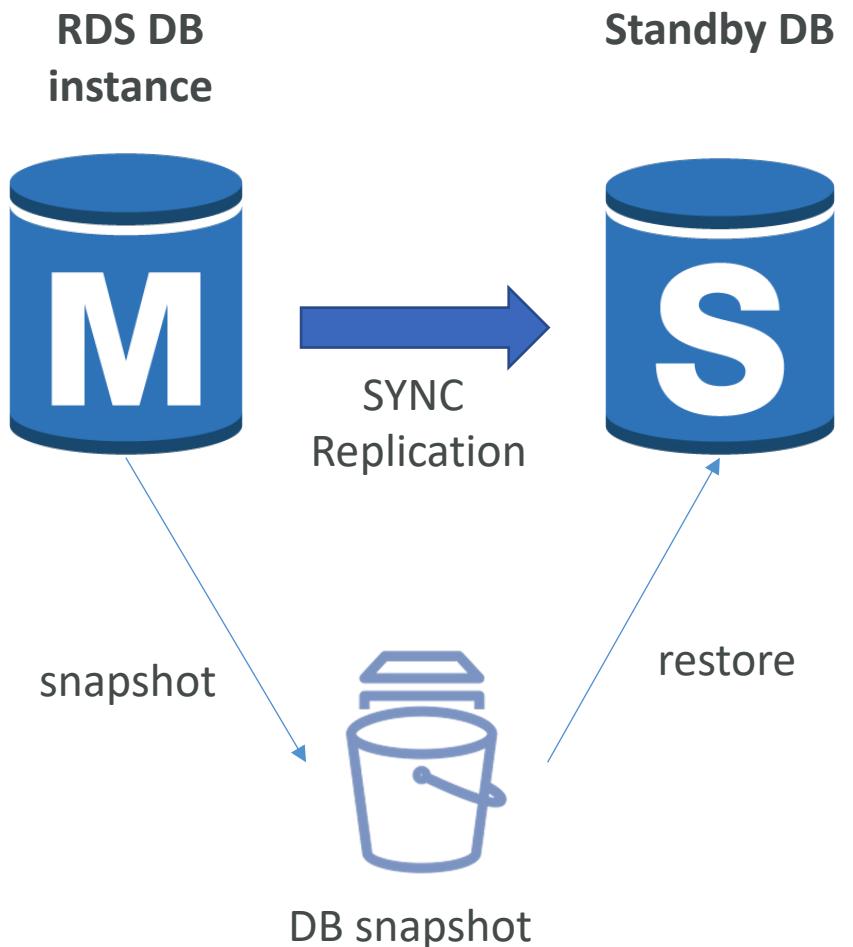
# RDS Multi AZ (Disaster Recovery)

- SYNC replication
- One DNS name – automatic app failover to standby
- Increase availability
- Failover in case of loss of AZ, loss of network, instance or storage failure
- No manual intervention in apps
- Not used for scaling
- Note: The Read Replicas be setup as Multi AZ for Disaster Recovery (DR)



# RDS – From Single-AZ to Multi-AZ

- Zero downtime operation (no need to stop the DB)
- Just click on “modify” for the database
- The following happens internally:
  - A snapshot is taken
  - A new DB is restored from the snapshot in a new AZ
  - Synchronization is established between the two databases



# RDS Multi AZ – Failover Conditions

- The primary DB instance
  - Failed
  - OS is undergoing software patches
  - Unreachable due to loss of network connectivity
  - Modified (e.g., DB instance type changed)
  - Busy and unresponsive
  - Underlying storage failure
- An Availability Zone outage
- A manual failover of the DB instance was initiated using **Reboot with failover**

# RDS Backup vs. Snapshots

## Backups

- Backups are “continuous” and allow point in time recovery
- Backups happen during maintenance windows
- When you delete a DB instance, you can retain automated backups
- Backups have a retention period you set between 0 and 35 days
- To disable backups, set retention period to 0

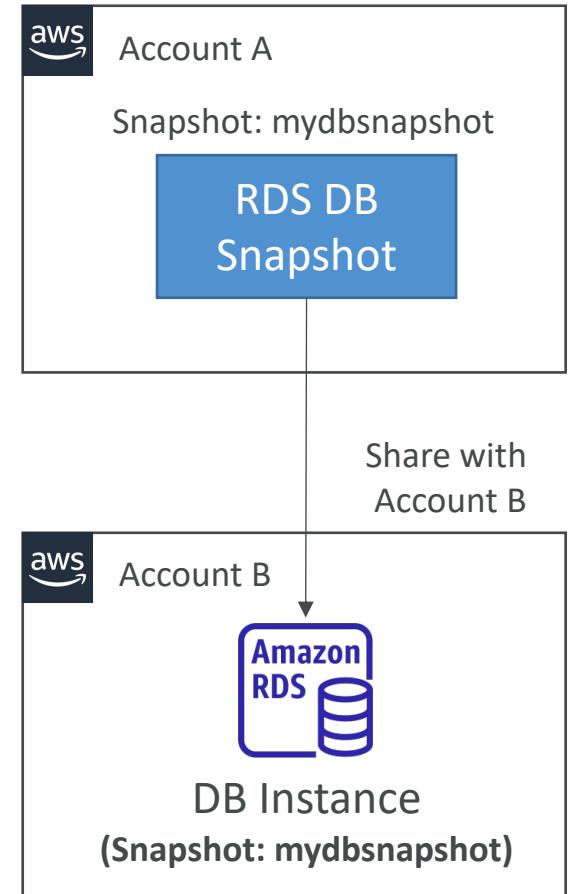
## Snapshots

- Snapshots takes IO operations and can stop the database from seconds to minutes
- Snapshots taken on Multi AZ DB don’t impact the master – just the standby
- Snapshots are incremental after the first snapshot (which is full)
- You can copy & share DB Snapshots
- Manual Snapshots don’t expire
- You can take a “final snapshot” when you delete your DB

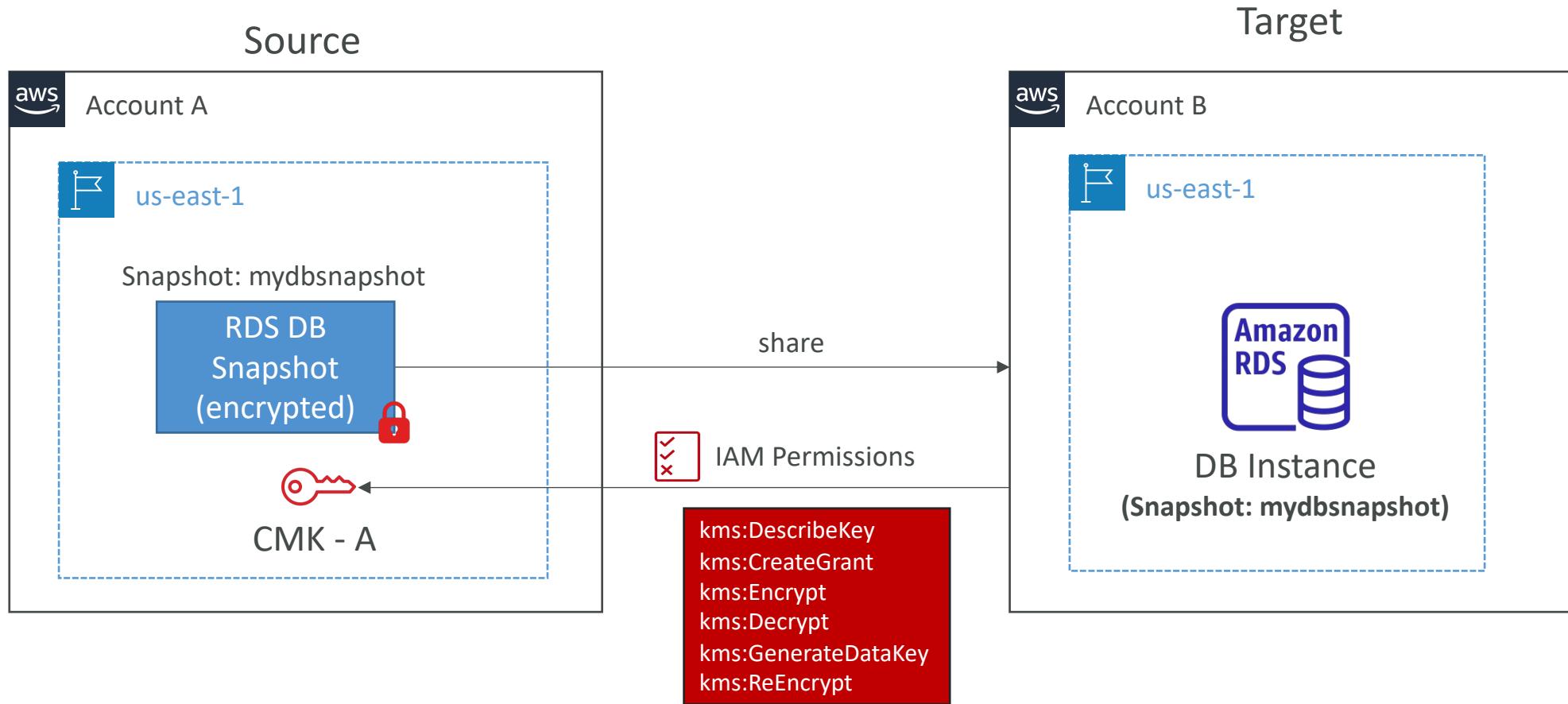
Restoring from Automated Backups or DB Snapshots creates a new DB Instance

# RDS Snapshots Sharing

- **Manual snapshots:** can be shared with AWS accounts
- **Automated snapshots:** can't be shared, copy first
- You can only share unencrypted snapshots and snapshots encrypted with a customer managed key
- If you share an encrypted snapshots, you must also share any customer managed keys used to encrypt them

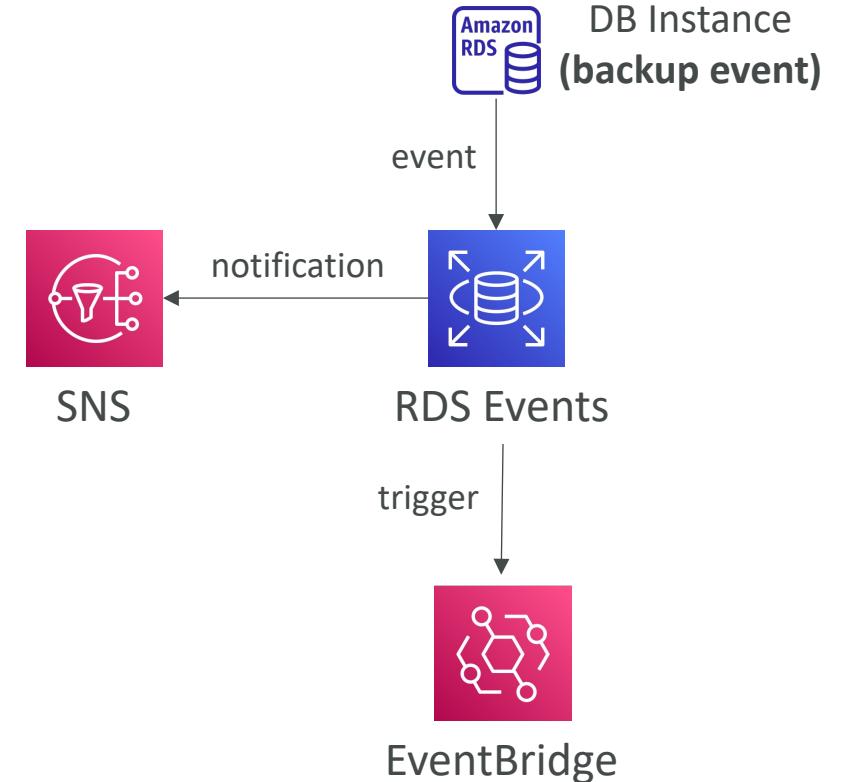


# RDS Snapshots Sharing with KMS Encryption

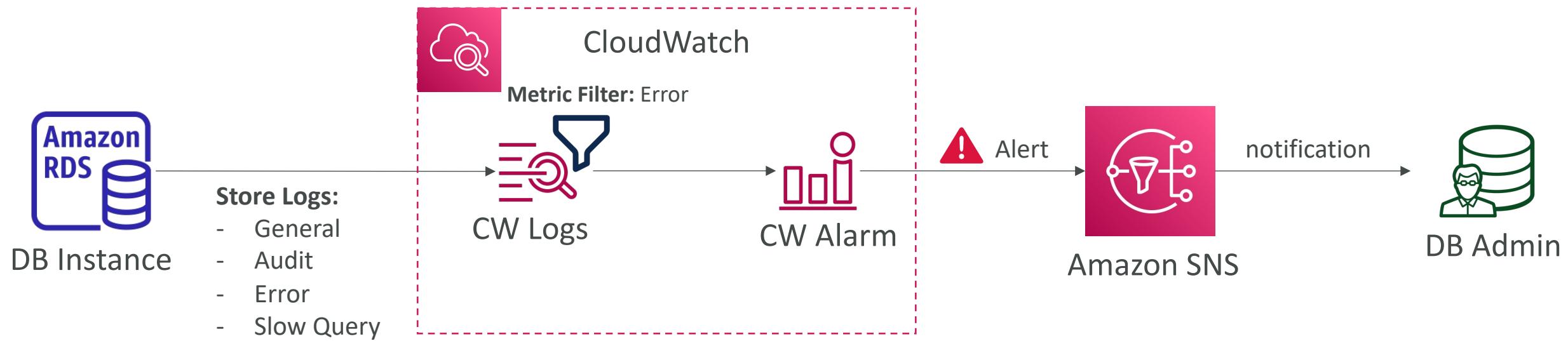


# RDS Events & Event Subscriptions

- RDS keeps record of events related to:
  - DB instances
  - Snapshots
  - Parameter groups, security groups ...
- Example: DB state changed from pending to running
- **RDS Event Subscriptions**
  - Subscribe to events to be notified when an event occurs using SNS
  - Specify the Event Source (instances, SGs, ...) and the Event Category (creation, failover, ...)
- RDS delivers events to EventBridge



# RDS Database Log Files



# RDS with CloudWatch

- CloudWatch metrics associated with RDS (gathered from the hypervisor):
  - DatabaseConnections
  - SwapUsage
  - ReadIOPS / WriteIOPS
  - ReadLatency / WriteLatency
  - ReadThroughPut / WriteThroughPut
  - DiskQueueDepth
  - FreeStorageSpace
- Enhanced Monitoring (gathered from an agent on the DB instance)
  - Useful when you need to see how different processes or threads use the CPU
  - Access to over 50 new CPU, memory, file system, and disk I/O metrics

# RDS Performance Insights

- Visualize your database performance and analyze any issues that affect it
- With the Performance Insights dashboard, you can visualize the database load and filter the load:
  - By Waits => find the resource that is the bottleneck (CPU, IO, lock, etc...)
  - By SQL statements => find the SQL statement that is the problem
  - By Hosts => find the server that is using the most our DB
  - By Users => find the user that is using the most our DB
- DBLoad = the number of active sessions for the DB engine
- You can view the SQL queries that are putting load on your database
- Generates **Proactive Recommendation** to help you prevent future database performance impact (**Paid Tier**)

# Performance Insights Screenshots

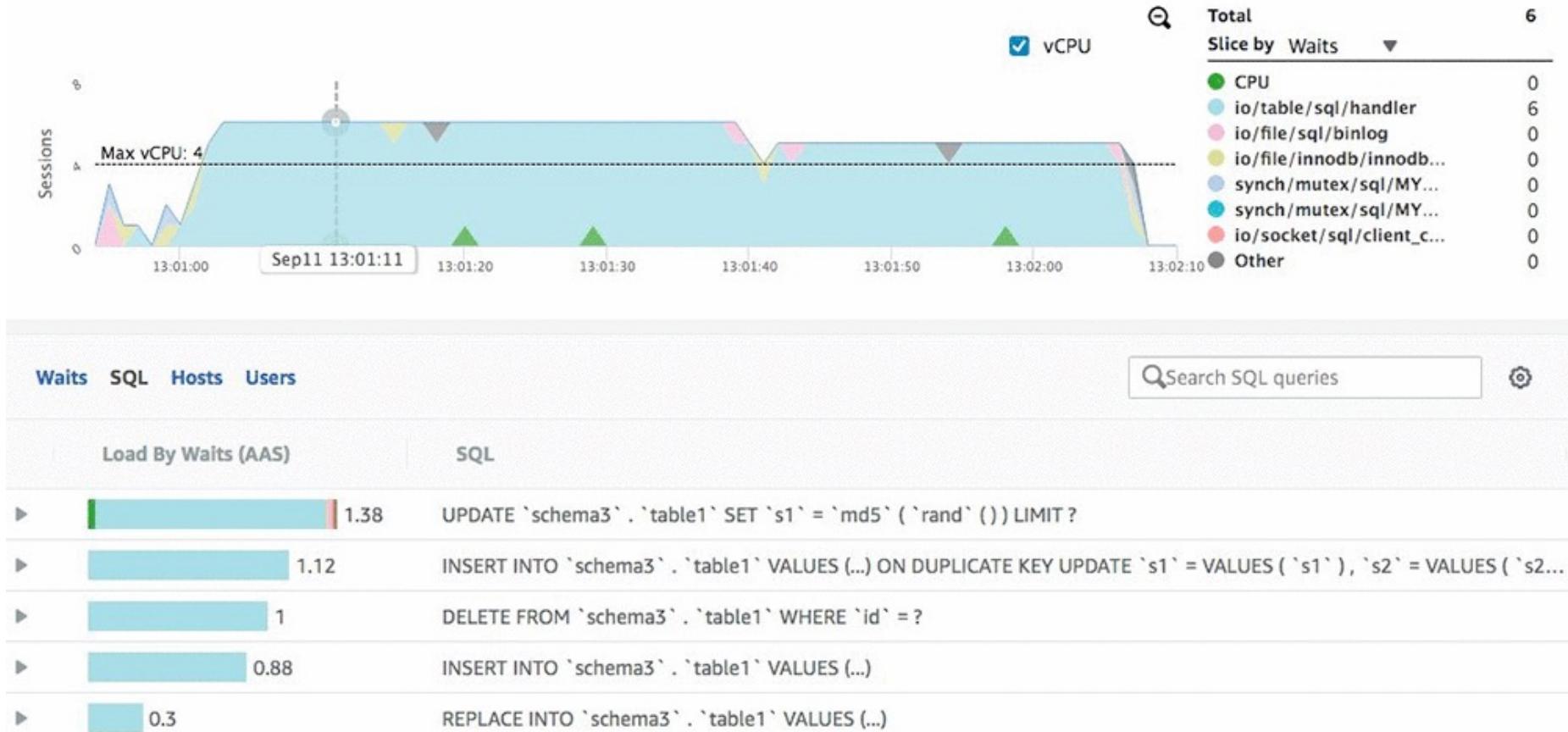
## DB Waits



From: <https://aws.amazon.com/blogs/database/tuning-amazon-rds-for-mysql-with-performance-insights/>

# Performance Insights Screenshots

## SQL



From: <https://aws.amazon.com/blogs/database/tuning-amazon-rds-for-mysql-with-performance-insights/>

# Performance Insights Screenshots

## Users



From: <https://aws.amazon.com/blogs/database/tuning-amazon-rds-for-mysql-with-performance-insights/>

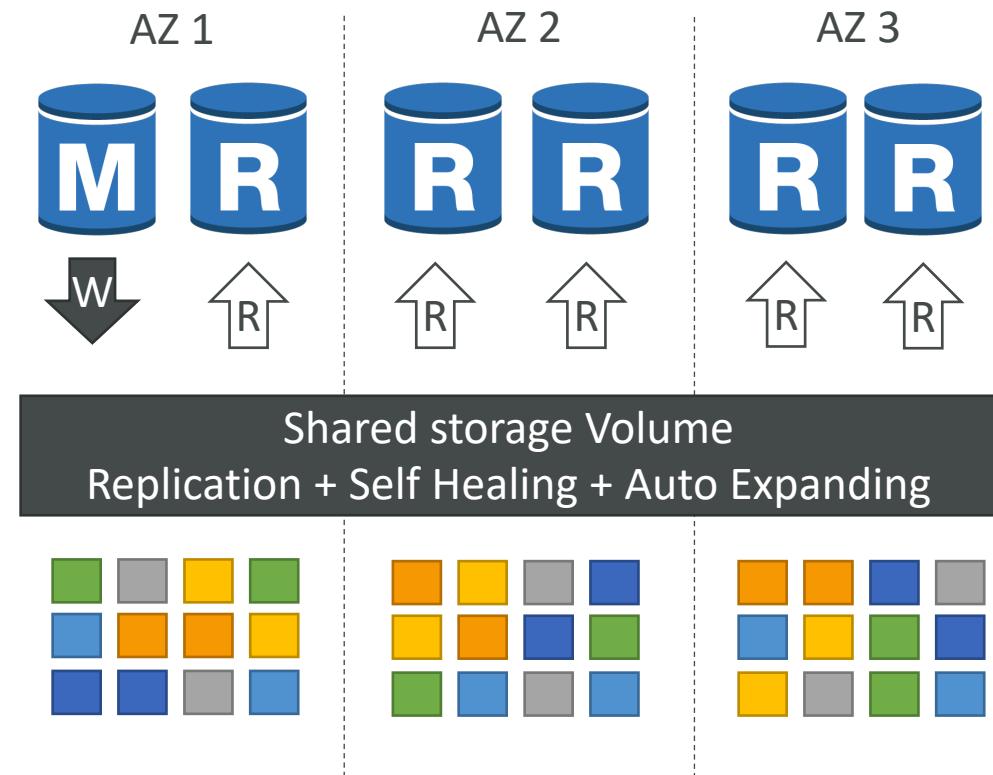


# Amazon Aurora

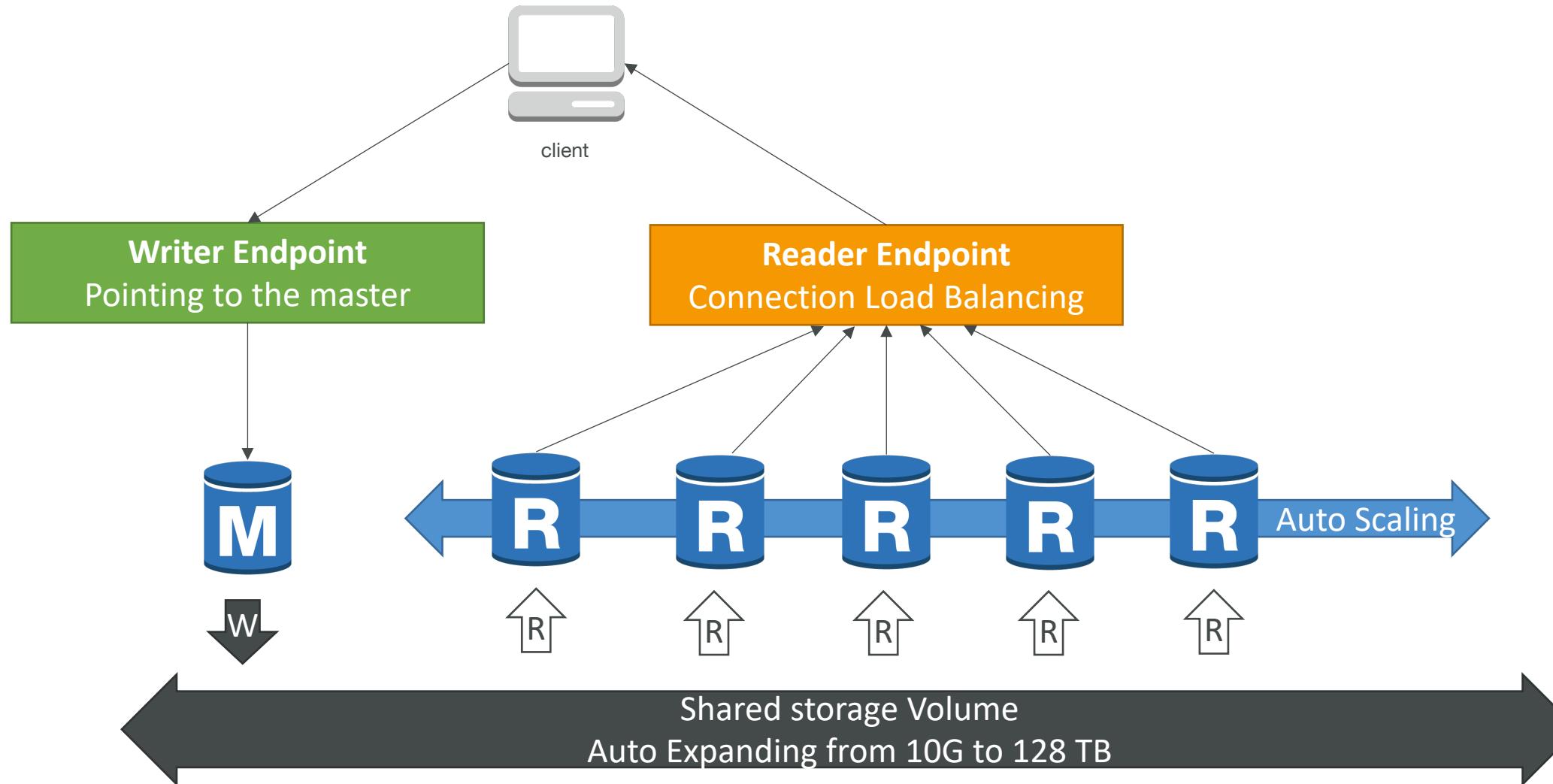
- Aurora is a proprietary technology from AWS (not open sourced)
- Postgres and MySQL are both supported as Aurora DB (that means your drivers will work as if Aurora was a Postgres or MySQL database)
- Aurora is “AWS cloud optimized” and claims 5x performance improvement over MySQL on RDS, over 3x the performance of Postgres on RDS
- Aurora storage automatically grows in increments of 10GB, up to 256 TB.
- Aurora can have up to 15 replicas and the replication process is faster than MySQL (sub 10 ms replica lag)
- Failover in Aurora is instantaneous. It’s HA (High Availability) native.
- Aurora costs more than RDS (20% more) – but is more efficient

# Aurora High Availability and Read Scaling

- 6 copies of your data across 3 AZ:
  - 4 copies out of 6 needed for writes
  - 3 copies out of 6 need for reads
  - Self healing with peer-to-peer replication
  - Storage is striped across 100s of volumes
- One Aurora Instance takes writes (master)
- Automated failover for master in less than 30 seconds
- Master + up to 15 Aurora Read Replicas serve reads
- Support for Cross Region Replication



# Aurora DB Cluster



# Features of Aurora

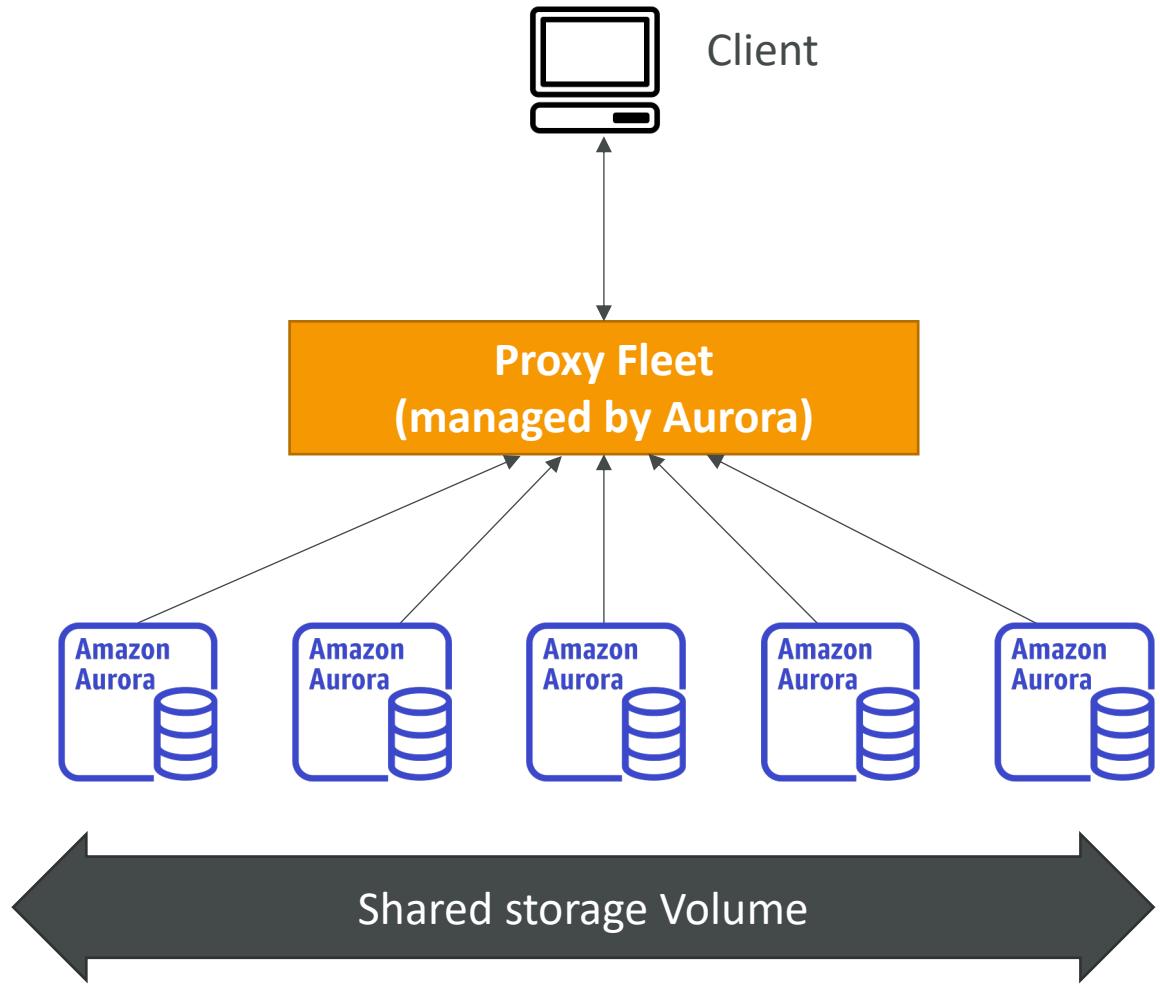
- Automatic fail-over
- Backup and Recovery
- Isolation and security
- Industry compliance
- Push-button scaling
- Automated Patching with Zero Downtime
- Advanced Monitoring
- Routine Maintenance
- Backtrack: restore data at any point of time without using backups

# Backups, Backtracking & Restores in Aurora

- **Automatic Backups**
  - Retention period 1-35 days (can't be disabled)
  - PITR, restore your DB cluster within 5 minutes of the current time
  - Restore to a new DB cluster
- **Aurora Backtracking**
  - Rewind the DB cluster back and forth in time (up to 72 hours)
  - Doesn't create a new DB cluster (in-place restore)
  - Supports Aurora MySQL only
- **Aurora Database Cloning**
  - Creates a new DB cluster that uses the same DB cluster volume as the original cluster
  - Uses copy-on-write protocol (use the original/single copy of the data and allocate storage only when changes made to the data)
  - Example: create a test environment using your production data

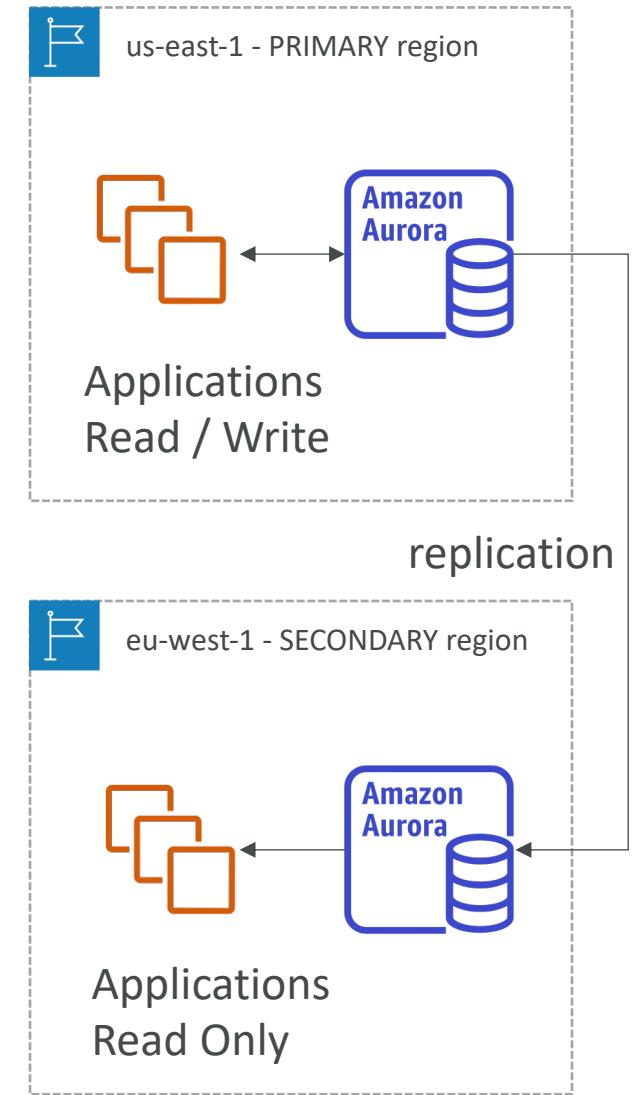
# Aurora Serverless

- Automated database instantiation and auto-scaling based on actual usage
- Good for infrequent, intermittent or unpredictable workloads
- No capacity planning needed
- Pay per second, can be more cost-effective



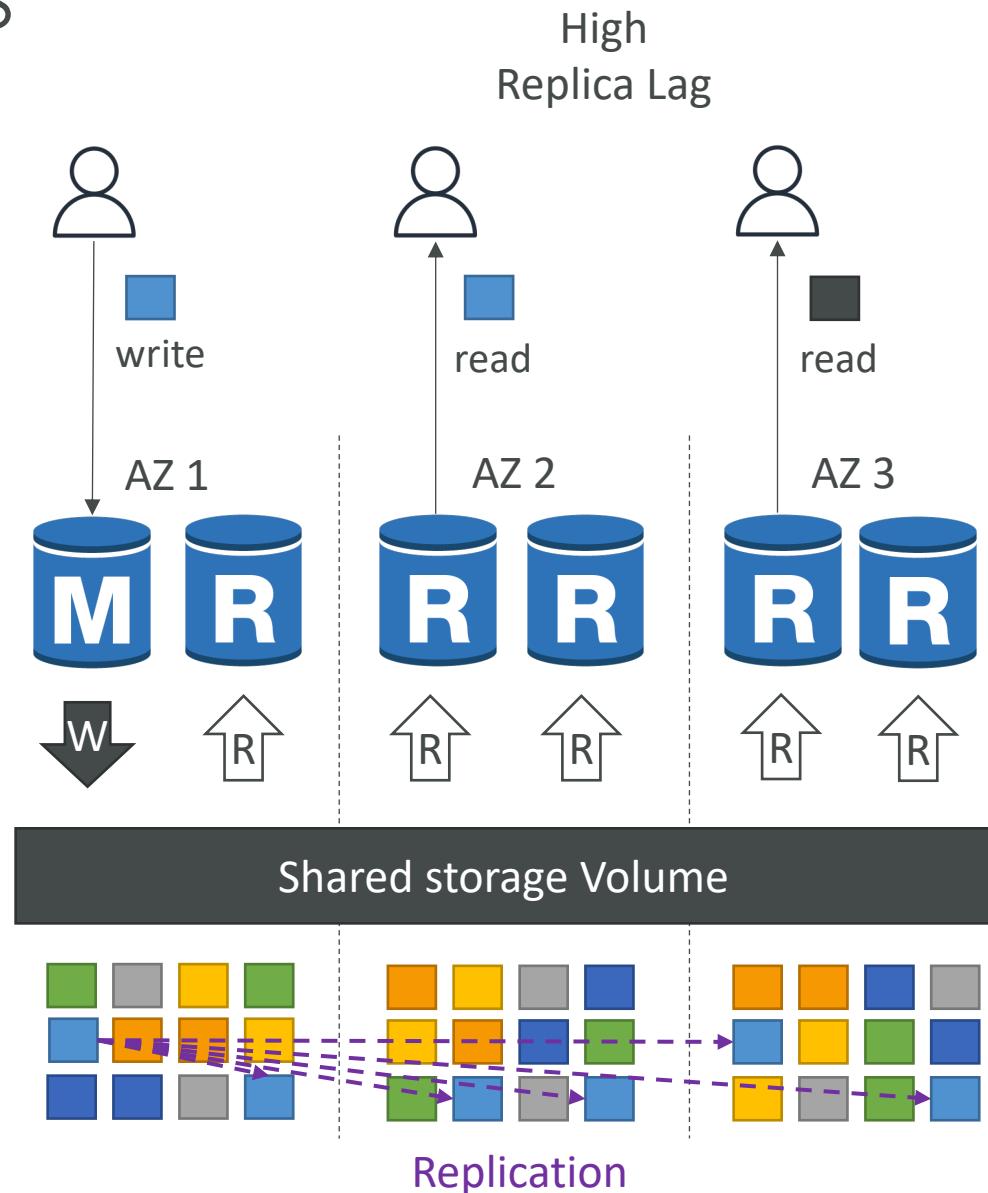
# Aurora Global Database

- 1 Primary Region (read / write)
- Up to 10 secondary (read-only) regions, replication lag is less than 1 second
- Up to 16 Read Replicas per secondary region
- Helps for decreasing latency
- Promoting another region (for disaster recovery) has an RTO of < 1 minute
- Typical cross-region replication takes less than 1 second



# Aurora: CloudWatch metrics

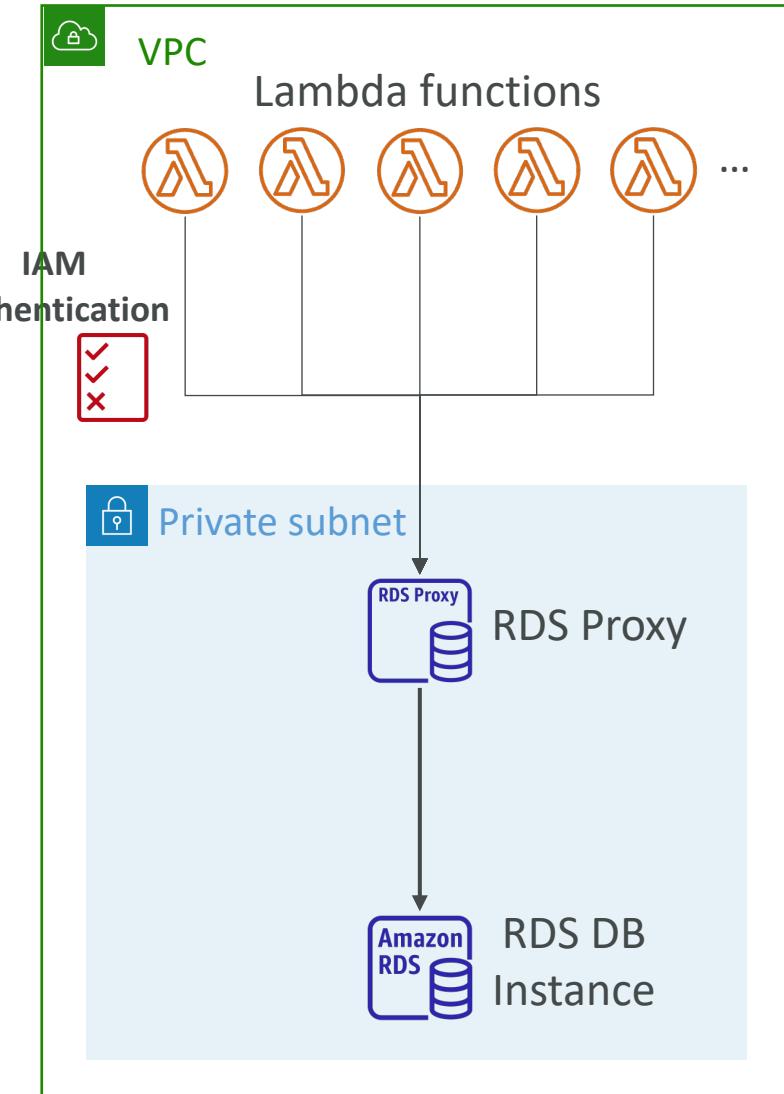
- **AuroraReplicaLag**: amount of lag when replicating updates from the primary instance
- **AuroraReplicaLagMaximum**: max. amount of lag across all DB instances in the cluster
- **AuroraReplicaLagMinimum**: min. amount of lag across all DB instances in the cluster
- If replica lag is high, that means the users will have a different experience based on which replica they get the data from (eventual consistency)
- **DatabaseConnections**: current number of connections to a DB instance
- **InsertLatency**: average duration of insert operations





# Amazon RDS Proxy

- Fully managed database proxy for RDS
- Allows apps to pool and share DB connections established with the database
- Improving database efficiency by reducing the stress on database resources (e.g., CPU, RAM) and minimize open connections (and timeouts)
- Serverless, autoscaling, highly available (multi-AZ)
- Reduced RDS & Aurora failover time by up 66%
- Supports RDS (MySQL, PostgreSQL, MariaDB, MS SQL Server) and Aurora (MySQL, PostgreSQL)
- No code changes required for most apps
- Enforce IAM Authentication for DB, and securely store credentials in AWS Secrets Manager
- RDS Proxy is never publicly accessible (must be accessed from VPC)





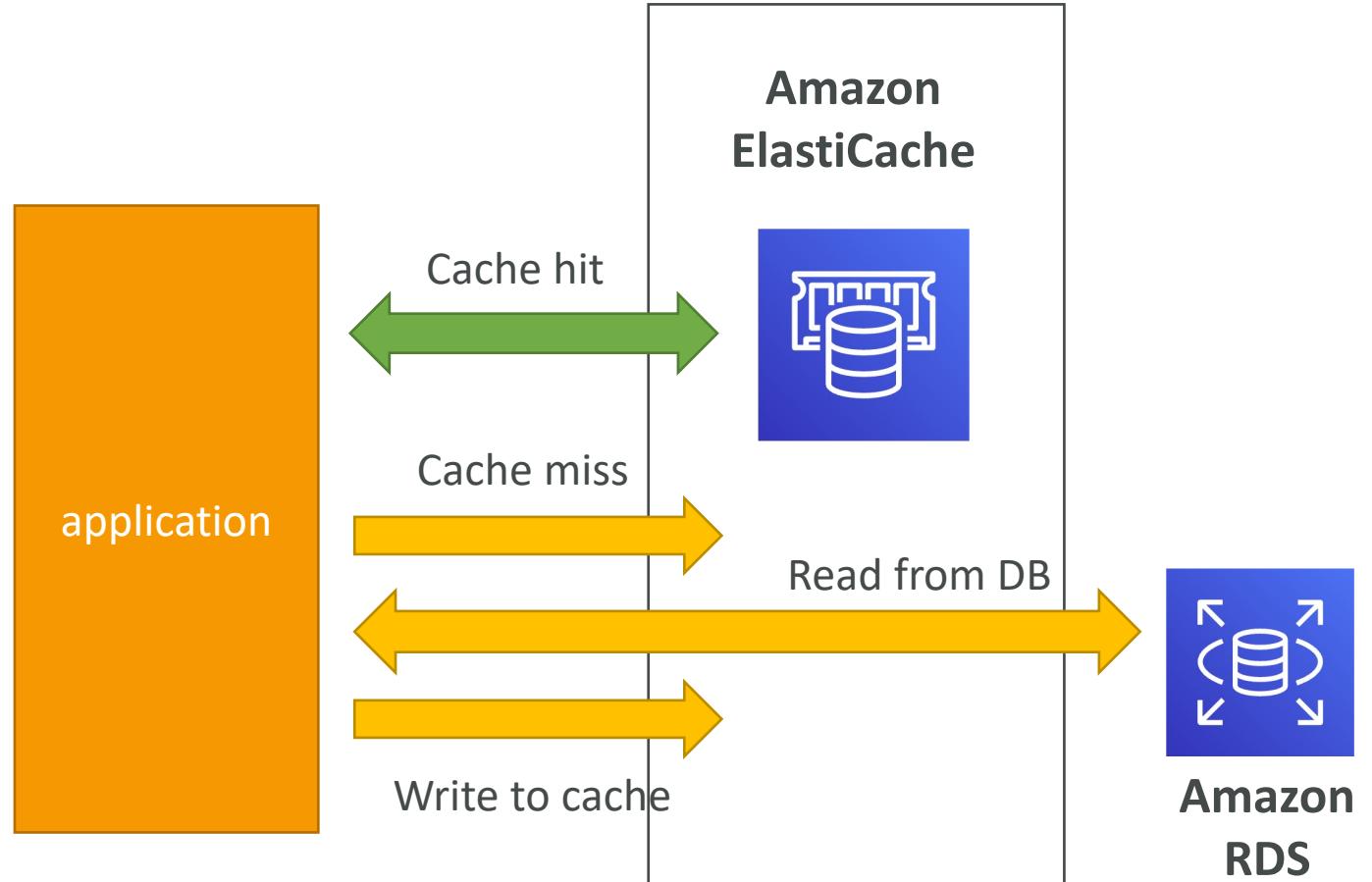
# Amazon ElastiCache Overview

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

# ElastiCache

## Solution Architecture - DB Cache

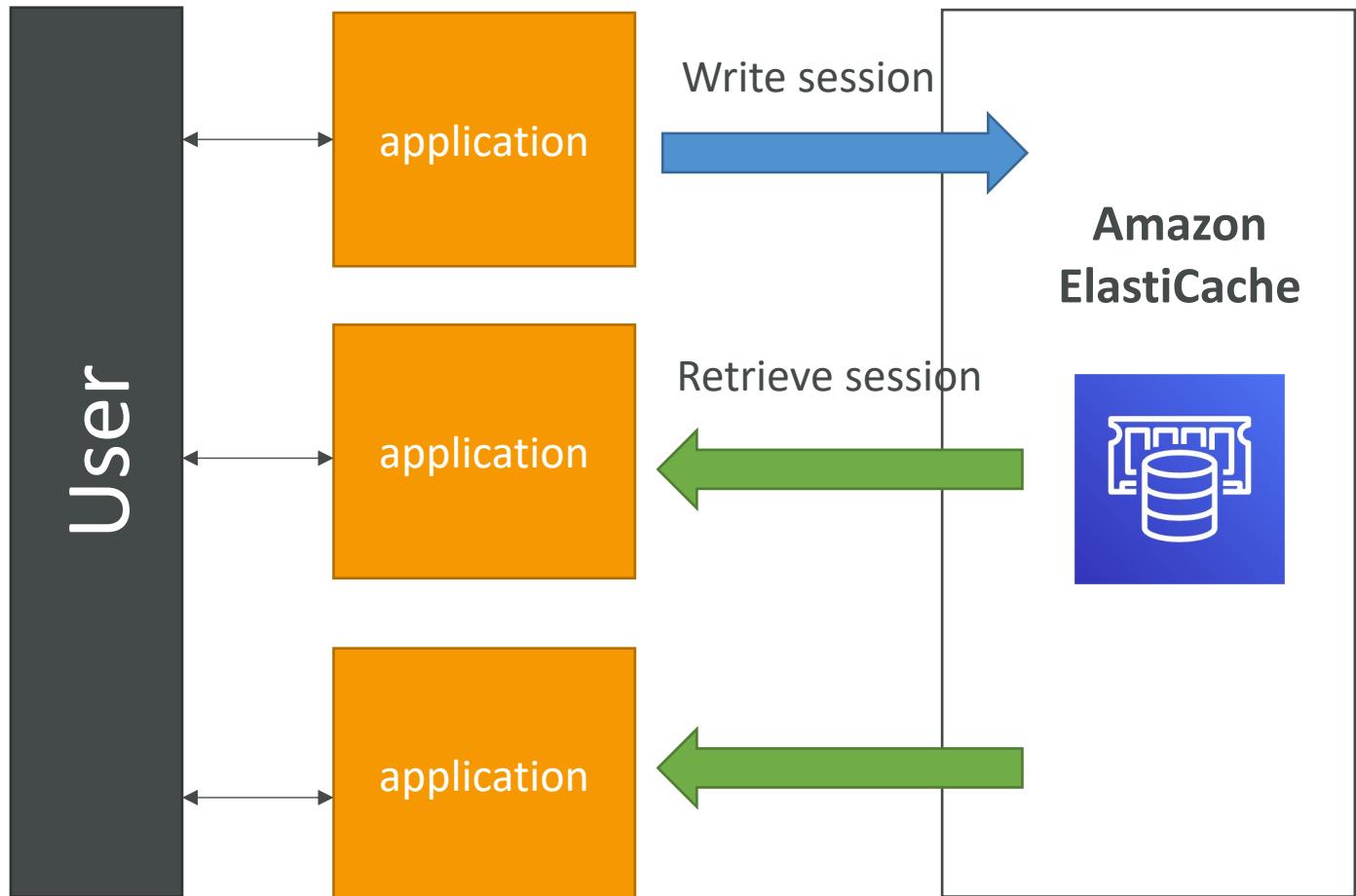
- Applications queries ElastiCache, if not available, get from RDS and store in ElastiCache.
- Helps relieve load in RDS
- Cache must have an invalidation strategy to make sure only the most current data is used in there.



# ElastiCache

## Solution Architecture – User Session Store

- User logs into any of the application
- The application writes the session data into ElastiCache
- The user hits another instance of our application
- The instance retrieves the data and the user is already logged in



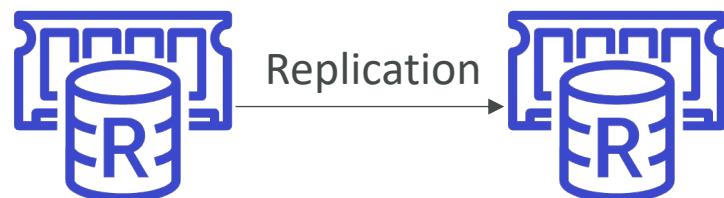
# ElastiCache – Redis vs Memcached

## REDIS

- Multi AZ with Auto-Failover
- Read Replicas to scale reads and have high availability
- Data Durability using AOF persistence
- Backup and restore features
- Supports Sets and Sorted Sets

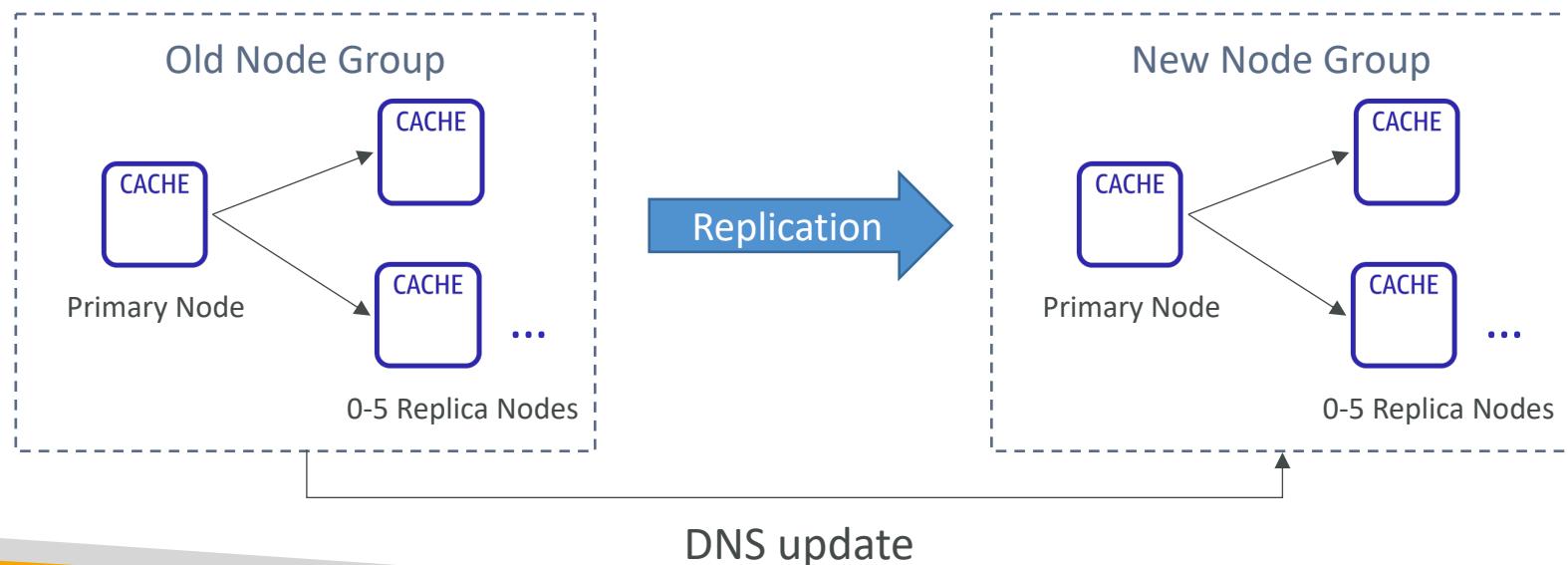
## MEMCACHED

- Multi-node for partitioning of data (sharding)
- No high availability (replication)
- Non persistent
- Backup and restore (Serverless)
- Multi-threaded architecture



# Redis Scaling – Cluster Mode Disabled

- Horizontal:
  - Scale out/in by adding/removing read replicas (max. 5 replicas)
- Vertical:
  - Scale up/down to larger/smaller node type
  - ElastiCache will internally create a new node group, then data replication and DNS update



# Redis Scaling – Cluster Mode Enabled

- Two Modes:
  - **Online Scaling:** continue serving requests during the scaling process (no downtime, some degradation in performance)
  - **Offline Scaling:** unable to serve requests during the scaling process (backup and restore). Additional configurations supported (change node type, upgrade engine version, ...)
- **Horizontal:** (Resharding and Shard Rebalancing)
  - Resharding: scale out/in by adding/removing shards
  - Shard Rebalancing: equally distribute the keystpaces among the shards as possible
  - Supports **Online** and **Offline Scaling**
- **Vertical:** (change read/write capacity)
  - Scale up/down to larger/smaller node type
  - Supports **Online Scaling**

# Redis Metrics to Monitor

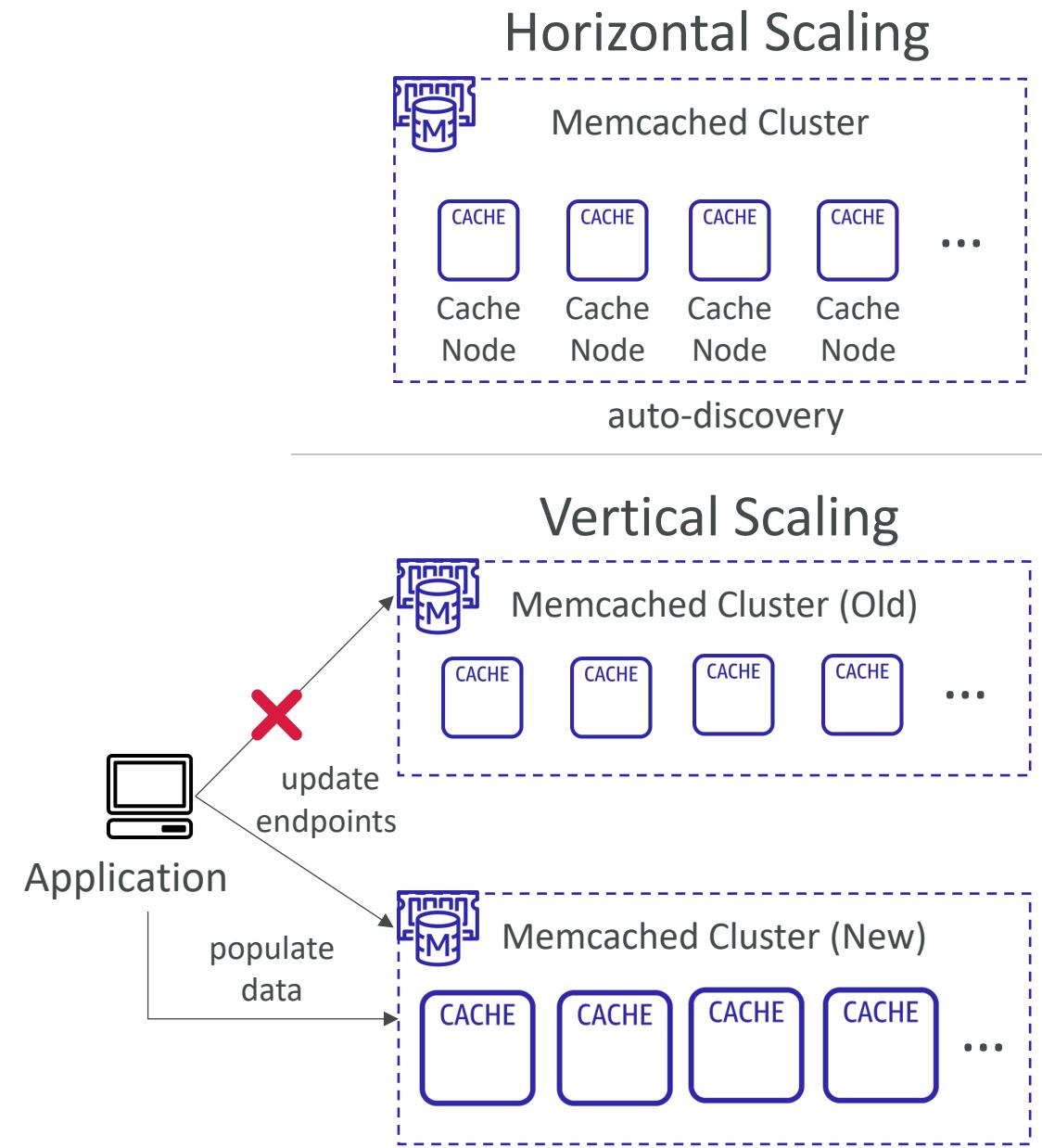
- **Evictions:** the number of non-expired items the cache evicted to allow space for new writes (memory is overfilled). Solution:
  - Choose an eviction policy to evict expired items (e.g., evict least recently used (LRU) items)
  - Scale up to larger node type (more memory) or scale out by adding more nodes
- **CPUUtilization:** monitor CPU utilization for the entire host
  - Solution: scale up to larger node type (more memory) or scale out by adding more nodes
- **SwapUsage:** should not exceed 50 MB
  - Solution: verify that you have configured enough reserved memory

# Redis Metrics to Monitor

- **CurrConnections:** the number of concurrent and active connections
  - Solution: investigate application behavior to address the issue
- **DatabaseMemoryUsagePercentage:** the percentage of memory utilization
- **NetworkBytesIn/Out & NetworkPacketsIn/Out**
- **ReplicationBytes:** the volume of data being replicated
- **ReplicationLag:** how far behind the replica is from the primary node

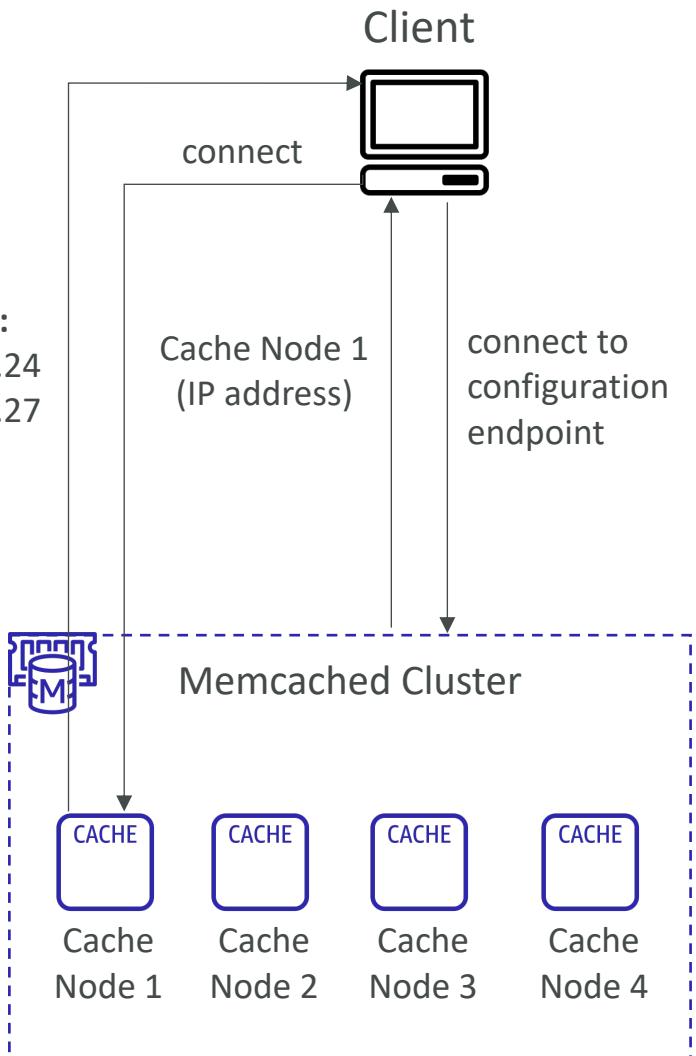
# Memcached Scaling

- Memcached Clusters can have 1-40 nodes (soft limit)
- **Horizontal:**
  - Add/remove nodes from the cluster
  - **Auto-discovery** allow your app to find nodes
- **Vertical:**
  - Scale up/down to larger/smaller node type
  - Scale up process:
    - Create a new cluster with the new node type
    - Update your application to use the new cluster's endpoints
    - Delete the old cluster
  - Memcached clusters/nodes start out empty



# Memcached Auto Discovery

- Typically you need to manually connect to individual cache nodes in the cluster using its DNS endpoints
- Auto Discovery automatically identifies all of the nodes
- All the cache nodes in the cluster maintain a list of metadata about all other nodes
- This is seamless from a client perspective



# Memcached Metrics to Monitor

- **Evictions:** the number of non-expired items the cache evicted to allow space for new writes (memory is overfilled). Solution:
  - Choose an eviction policy to evict expired items (e.g., LRU items)
  - Scale up to larger node type (more memory) or scale out by adding more nodes
- **CPUUtilization**
  - Solution: scale up to larger node type or scale out by adding more nodes
- **SwapUsage:** should not exceed 50 MB
- **CurrConnections:** the number of concurrent and active connections
  - Solution: investigate application behavior to address the issue
- **FreeableMemory:** amount of free memory on the host

# AWS Monitoring, Audit & Performance

CloudWatch, CloudTrail & AWS Config

# AWS CloudWatch Metrics



- CloudWatch provides metrics for every services in AWS
- **Metric** is a variable to monitor (CPUUtilization, NetworkIn...)
- Metrics belong to **namespaces**
- **Dimension** is an attribute of a metric (instance id, environment, etc....).
- Up to 30 dimensions per metric
- Metrics have **timestamps**
- Can create CloudWatch dashboards of metrics

# EC2 Detailed monitoring

- EC2 instance metrics have metrics “every 5 minutes”
- With detailed monitoring (for a cost), you get data “every 1 minute”
- Use detailed monitoring if you want to scale faster for your ASG!
- The AWS Free Tier allows us to have 10 detailed monitoring metrics
- Note: EC2 Memory usage is by default not pushed (must be pushed from inside the instance as a custom metric)

# CloudWatch Custom Metrics

- Possibility to define and send your own custom metrics to CloudWatch
- Example: memory (RAM) usage, disk space, number of logged in users ...
- Use API call `PutMetricData`
- Ability to use dimensions (attributes) to segment metrics
  - `Instance.id`
  - `Environment.name`
- Metric resolution (`StorageResolution` API parameter – two possible value):
  - Standard: 1 minute (60 seconds)
  - High Resolution: 1/5/10/30 second(s) – Higher cost
- **Important:** Accepts metric data points two weeks in the past and two hours in the future (make sure to configure your EC2 instance time correctly)

# CloudWatch Anomaly Detection

- Also called CloudWatch outlier detection
- Continuously analyze metrics to determine normal baselines and surface anomalies using ML algorithms
- It creates a model of the metric's expected values (based on metric's past data)
- Shows you which values in the graph are out of the normal range
- Allows you to create Alarms based on metric's expected value (instead of Static Threshold)
- Ability to exclude specified time periods or events from being trained



[https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch\\_Anomaly\\_Detection.html](https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch_Anomaly_Detection.html)

# CloudWatch Dashboards

- Great way to setup custom dashboards for quick access to key metrics and alarms
- Dashboards are global
- Dashboards can include graphs from different AWS accounts and regions
- You can change the time zone & time range of the dashboards
- You can setup automatic refresh (10s, 1m, 2m, 5m, 15m)
- Dashboards can be shared with people who don't have an AWS account (public, email address, 3<sup>rd</sup> party SSO provider through Amazon Cognito)
- Pricing:
  - 3 dashboards (up to 50 metrics) for free
  - \$3/dashboard/month afterwards



# CloudWatch Logs

- **Log groups:** arbitrary name, usually representing an application
- **Log stream:** instances within application / log files / containers
- Can define log expiration policies (never expire, 1 day to 10 years...)
- **CloudWatch Logs can send logs to:**
  - Amazon S3 (exports)
  - Kinesis Data Streams
  - Kinesis Data Firehose
  - AWS Lambda
  - OpenSearch
- Logs are encrypted by default
- Can setup KMS-based encryption with your own keys

# CloudWatch Logs - Sources

- SDK, CloudWatch Logs Agent, CloudWatch Unified Agent
- Elastic Beanstalk: collection of logs from application
- ECS: collection from containers
- AWS Lambda: collection from function logs
- VPC Flow Logs: VPC specific logs
- API Gateway
- CloudTrail based on filter
- Route53: Log DNS queries

# CloudWatch Logs Insights

The screenshot shows the CloudWatch Logs Insights interface. At the top, there's a navigation bar with 'CloudWatch > Logs Insights'. Below it is a search bar labeled 'Select log group(s)' with 'application.log' selected. To the right of the search bar are time range controls set to '2021-11-09 (06:40:02) > 2021-11-09 (06:55:17)'. A large orange box highlights the search bar area with the text 'Change the time range here.' An arrow points from this box to the time range controls.

The main query editor area contains a code snippet:

```
1 fields @timestamp, @message
2 | sort @timestamp desc
3 | limit 20
```

An orange box highlights the query editor with the text 'Write your query here.' An arrow points from this box to the code snippet. Below the editor are three buttons: 'Run query', 'Save', and 'History'. A note below says 'Queries are allowed to run for up to 15 minutes.'

To the right of the query editor is a sidebar with 'Fields', 'Queries', and 'Help' sections. An orange box highlights the 'Fields' section with the text 'Discovered Fields in your log groups.' An arrow points from this box to the 'Fields' icon in the sidebar.

Below the query editor is a histogram showing log record counts over time. The x-axis represents time from 06:40 to 06:55, and the y-axis represents count from 0 to 400. The histogram shows several peaks, with the highest peak around 06:45. An orange box highlights the histogram area with the text 'Export the results, or add to a dashboard.' Two arrows point from this box to the 'Export results' and 'Add to dashboard' buttons.

At the bottom, there are tabs for 'Logs' (which is selected) and 'Visualization'. An orange box highlights the 'Logs' tab with the text 'Tabs for query results, and visualization options.' An arrow points from this box to the 'Logs' tab.

The bottom section displays the results of the query, showing 20 of 10,197 records matched. It includes a header table with columns '#', '@timestamp', and '@message'. Two sample log entries are shown:

#	@timestamp	@message
► 1	2021-11-09T06:54:17.62...	{"Severity": "INFO", "message": "This is where the message detail would go", "IP Address": "10.30.86.98", "Timestamp": "2021-11-09T11:54:17.620Z"}
► 2	2021-11-09T06:54:13.38...	{"Severity": "INFO", "message": "This is where the message detail would go", "IP Address": "192.168.0.43", "Timestamp": "2021-11-09T11:54:13.380Z"}

<https://mng.workshop.aws/operations-2022/detect/cwlogs.html>

# CloudWatch Logs Insights

- Search and analyze log data stored in CloudWatch Logs
- Example: find a specific IP inside a log, count occurrences of “ERROR” in your logs...
- Provides a purpose-built query language
  - Automatically discovers fields from AWS services and JSON log events
  - Fetch desired event fields, filter based on conditions, calculate aggregate statistics, sort events, limit number of events...
  - Can save queries and add them to CloudWatch Dashboards
- Can query multiple Log Groups in different AWS accounts
- It's a query engine, not a real-time engine

Sample queries [Learn more ↗](#)

- ▶ Lambda
- ▶ VPC Flow Logs
- ▶ CloudTrail
- ▼ Common queries

▼ 25 most recently added log events

```
fields @timestamp, @message
| sort @timestamp desc
| limit 25
```

[Apply](#)

▼ Number of exceptions logged every 5 minutes

```
filter @message like /Exception/
| stats count(*) as exceptionCount by
bin(5m)
| sort exceptionCount desc
```

[Apply](#)

▼ List of log events that are not exceptions

```
fields @message
| filter @message not like /Exception/
```

[Apply](#)

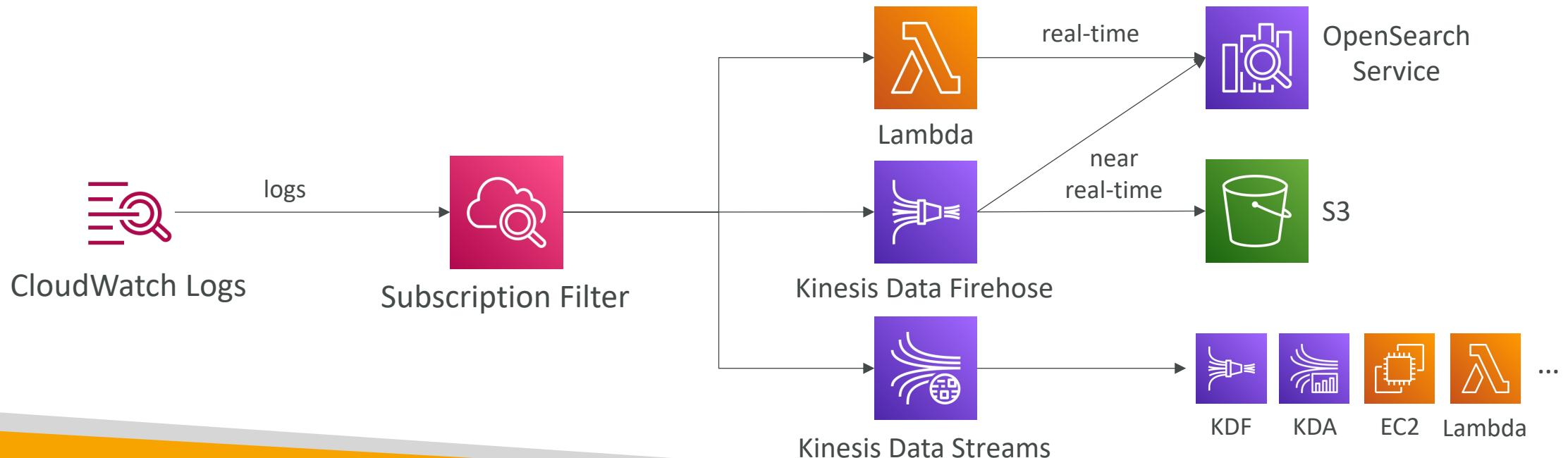
# CloudWatch Logs – S3 Export



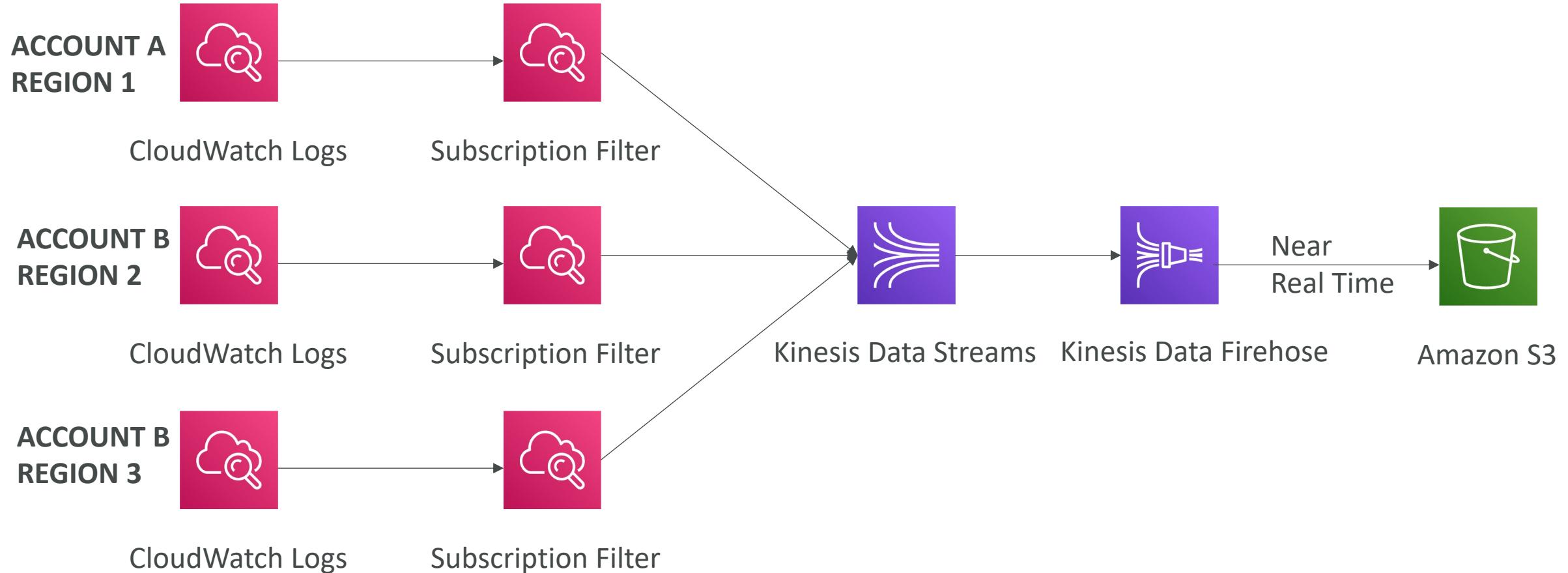
- Log data can take up to 12 hours to become available for export
- The API call is `CreateExportTask`
- Not near-real time or real-time... use Logs Subscriptions instead

# CloudWatch Logs Subscriptions

- Get a real-time log events from CloudWatch Logs for processing and analysis
- Send to Kinesis Data Streams, Kinesis Data Firehose, or Lambda
- **Subscription Filter** – filter which logs are events delivered to your destination

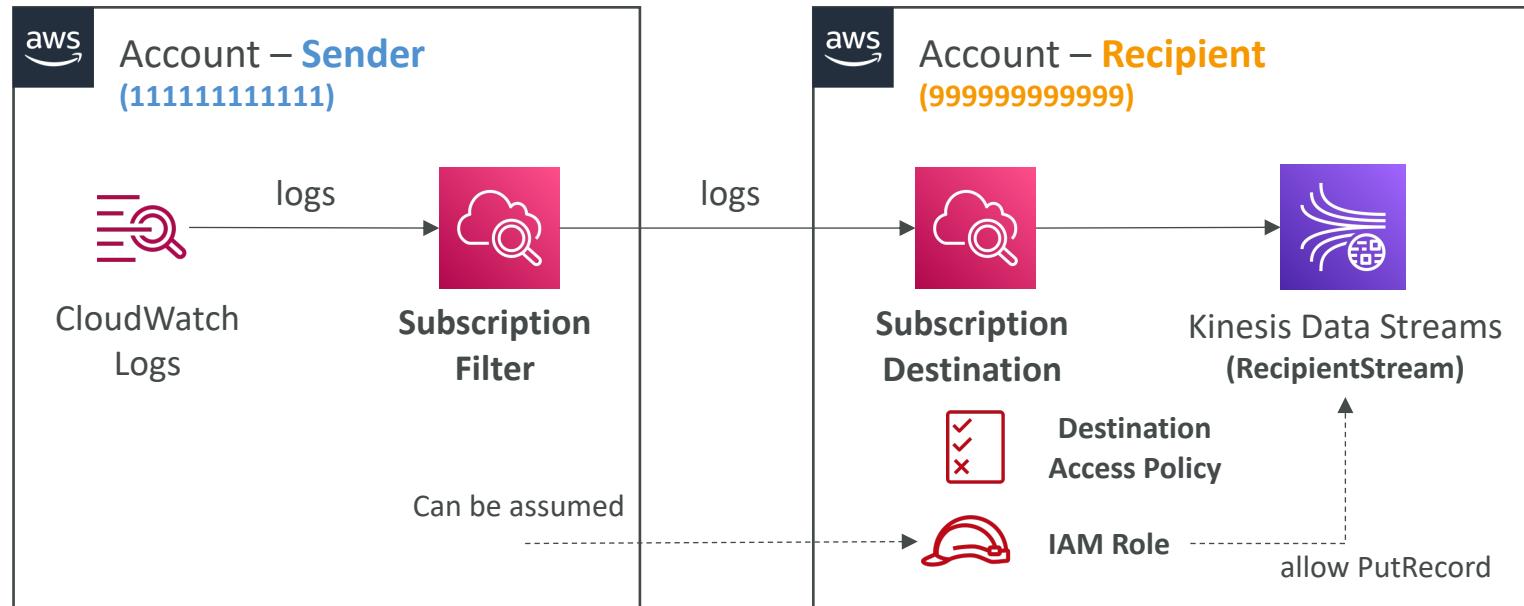


# CloudWatch Logs Aggregation Multi-Account & Multi Region



# CloudWatch Logs Subscriptions

- Cross-Account Subscription – send log events to resources in a different AWS account (KDS, KDF)



```

{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "kinesis:PutRecord",
            "Resource": "arn:aws:kinesis:us-east-1:999999999999:stream/RecipientStream"
        }
    ]
}

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "AWS": "111111111111"
            },
            "Action": "logs:PutSubscriptionFilter",
            "Resource": "arn:aws:logs:us-east-1:999999999999:destination:testDestination"
        }
    ]
}

```

**IAM Role (Cross-Account)**

**Destination Access Policy**

# CloudWatch Logs Data Protection



- Detect and mask sensitive log data ingested into CloudWatch Logs (at-rest and in transit)
- Uses ML to detect and protect sensitive log
- **Data Protection Policy**
  - Specify the Data Identifier(s) (e.g., email address)
  - 100+ Data Identifiers of common sensitive data patterns (e.g., emails, passwords, credit card details, social security number...)
  - Can create a Custom Data Identifier
- Can send Data Protection audit reports to another CloudWatch Log Group, Amazon S3, and Kinesis Data Firehose

Lambda Function



send logs

[88140ce4-5bfc-4b54-86e4-de18f1a2dee8]  
INFO : User has signed in with  
email address: [foobar@example.com](mailto:foobar@example.com)



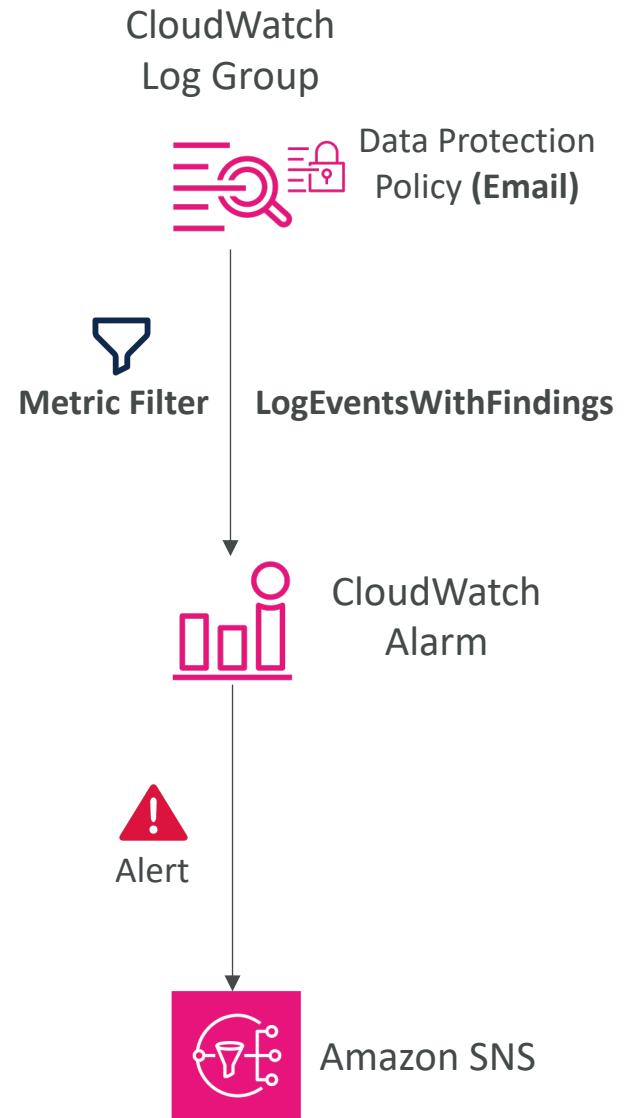
Data Protection Policy (Email)

CloudWatch Log Group

[88140ce4-5bfc-4b54-86e4-de18f1a2dee8]  
INFO : User has signed in with  
email address: \*\*\*\*\*

# CloudWatch Logs Data Protection

- You can get notified when sensitive data is detected using metric `LogEventsWithFindings`
- Sensitive data is masked at:
  - CloudWatch Logs Insights
  - Metric Filters
  - CloudWatch Logs Subscription Filters
- Only users with `logs:Unmask` permissions can view unmasked data



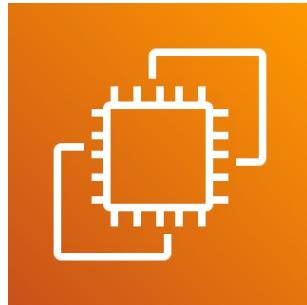
# CloudWatch Alarms



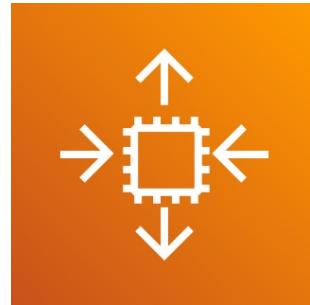
- Alarms are used to trigger notifications for any metric
- Various options (sampling, %, max, min, etc...)
- Alarm States:
  - OK
  - INSUFFICIENT\_DATA
  - ALARM
- Period:
  - Length of time in seconds to evaluate the metric
  - High resolution custom metrics: 10 sec, 30 sec or multiples of 60 sec

# CloudWatch Alarm Targets

- Stop, Terminate, Reboot, or Recover an EC2 Instance
- Trigger Auto Scaling Action
- Send notification to SNS (from which you can do pretty much anything)



Amazon EC2



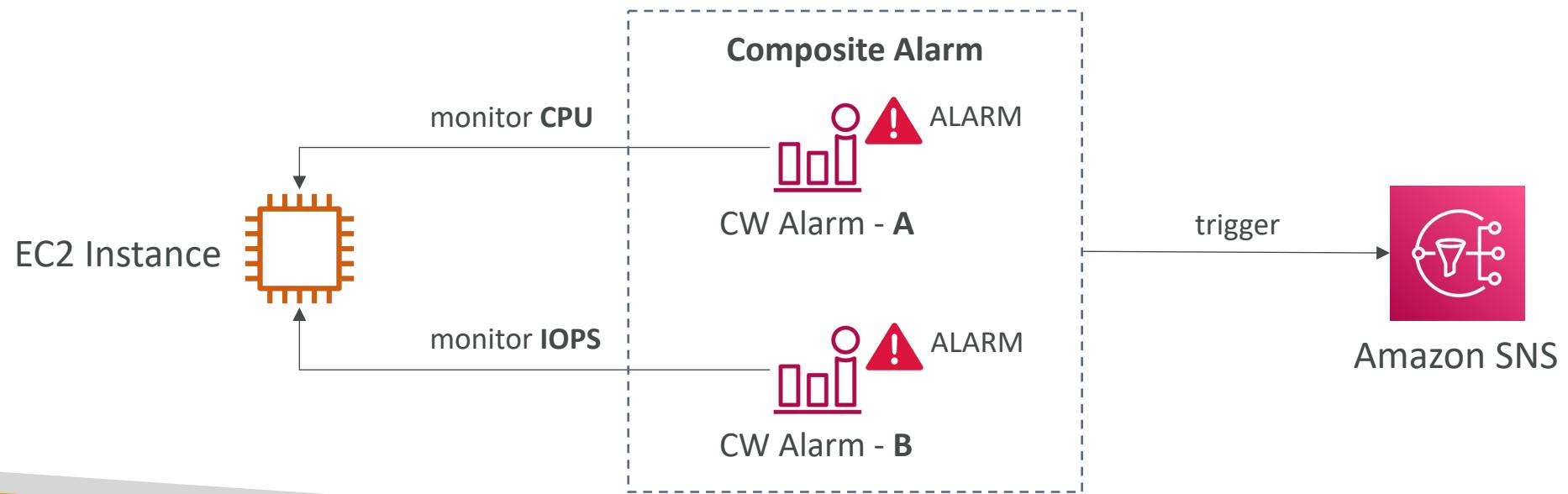
EC2 Auto Scaling



Amazon SNS

# CloudWatch Alarms – Composite Alarms

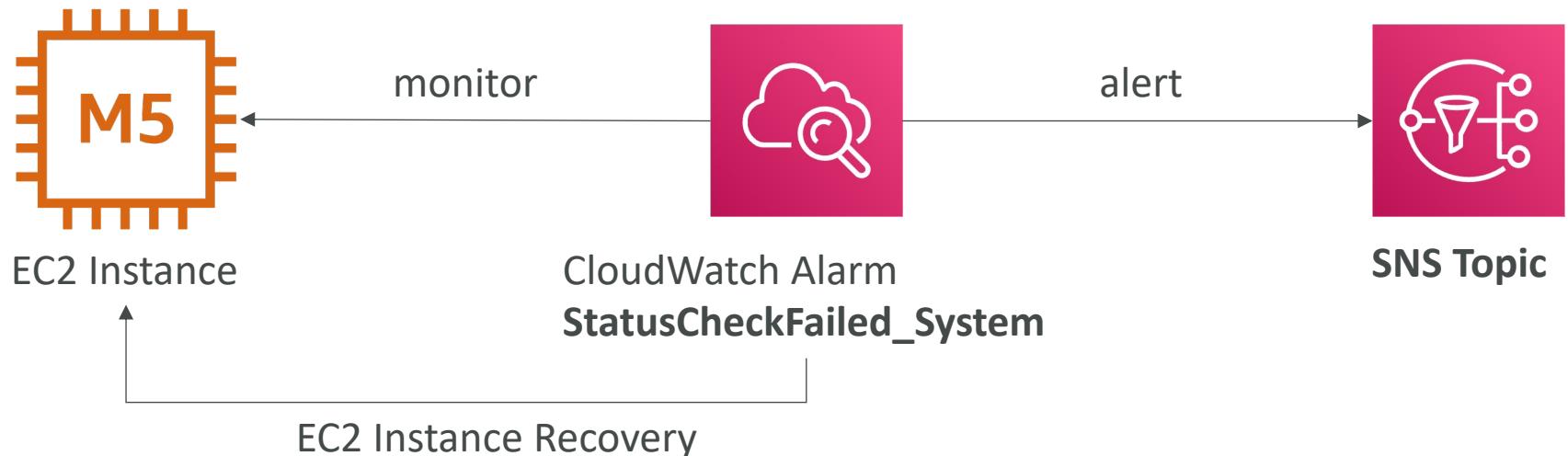
- CloudWatch Alarms are on a single metric
- Composite Alarms are monitoring the states of multiple other alarms
- AND and OR conditions
- Helpful to reduce “alarm noise” by creating complex composite alarms



# EC2 Instance Recovery

- Status Check:

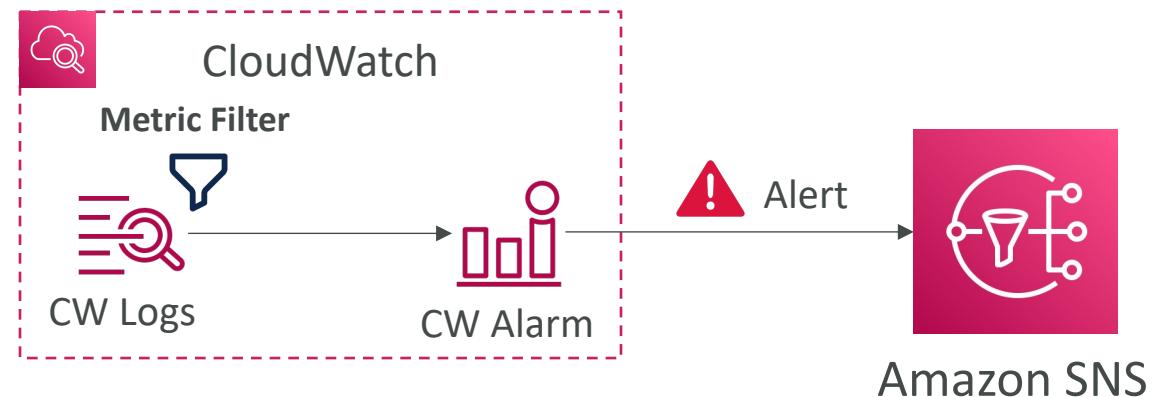
- Instance status = check the EC2 VM
- System status = check the underlying hardware
- Attached EBS status = check attached EBS volumes



- Recovery: Same Private, Public, Elastic IP, metadata, placement group

# CloudWatch Alarm: good to know

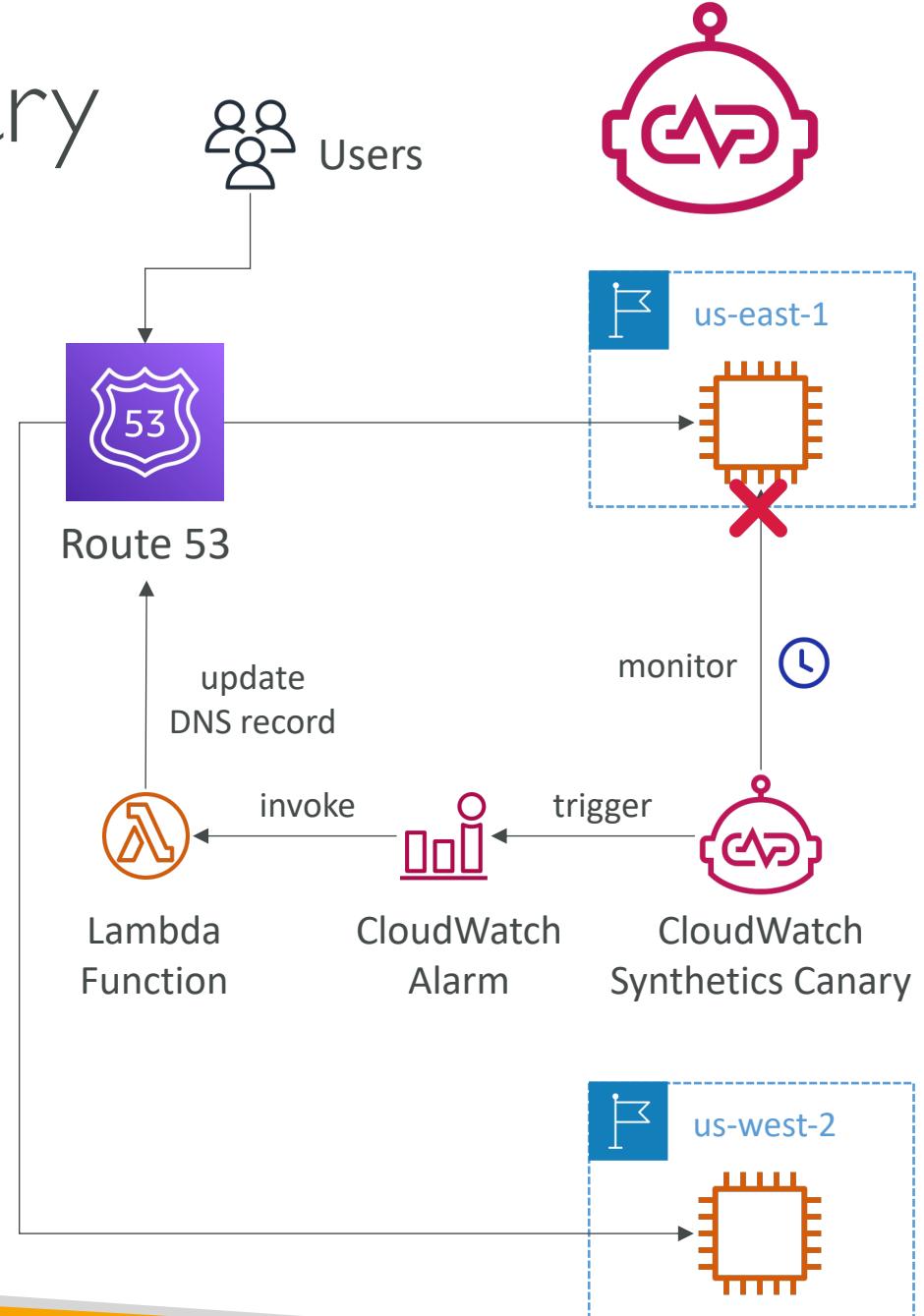
- Alarms can be created based on CloudWatch Logs Metrics Filters



- To test alarms and notifications, set the alarm state to Alarm using CLI  
`aws cloudwatch set-alarm-state --alarm-name "myalarm" --state-value ALARM --state-reason "testing purposes"`

# CloudWatch Synthetics Canary

- Configurable script that monitor your APIs, URLs, Websites, ...
- Reproduce what your customers do programmatically to find issues before customers are impacted
- Checks the availability and latency of your endpoints and can store load time data and screenshots of the UI
- Integration with CloudWatch Alarms
- Scripts written in Node.js or Python
- Programmatic access to a headless Google Chrome browser
- Can run once or on a regular schedule



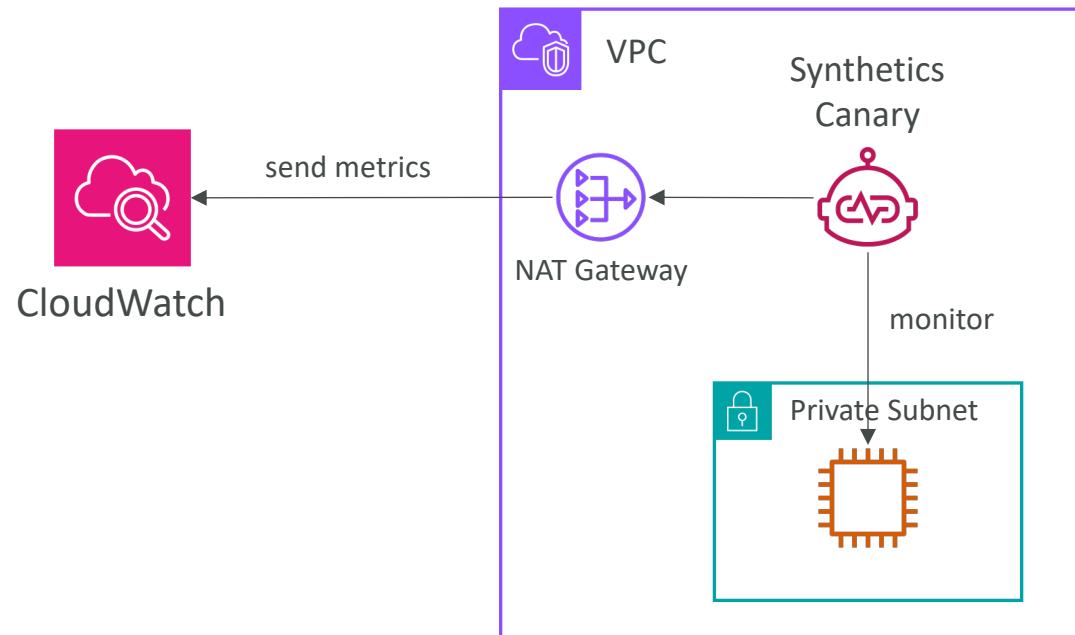
# CloudWatch Synthetics Canary Blueprints

- **Heartbeat Monitor** – load URL, store screenshot and an HTTP archive file
- **API Canary** – test basic read and write functions of REST APIs
- **Broken Link Checker** – check all links inside the URL that you are testing
- **Visual Monitoring** – compare a screenshot taken during a canary run with a baseline screenshot
- **Canary Recorder** – used with CloudWatch Synthetics Recorder (record your actions on a website and automatically generates a script for that)
- **GUI Workflow Builder** – verifies that actions can be taken on your webpage (e.g., test a webpage with a login form)

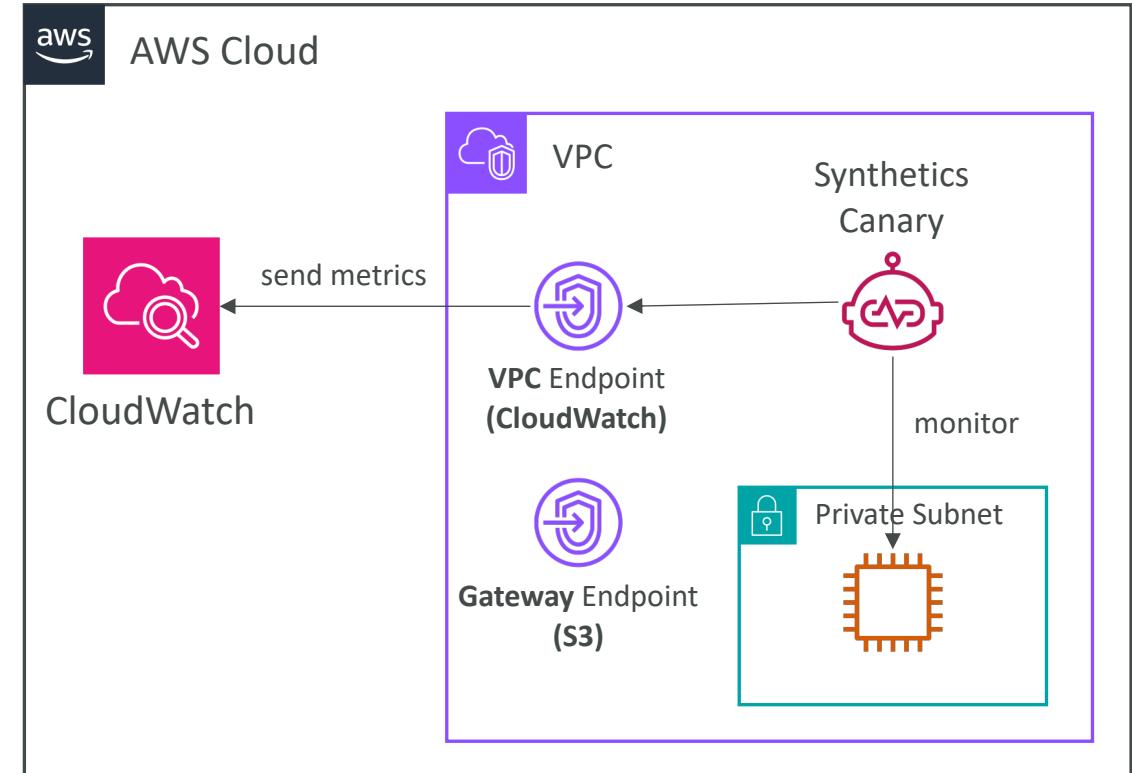
# CloudWatch Synthetics Canary in a VPC

- You can setup canaries on endpoints in a VPC
- Must have **DNS Resolution** and **DNS Hostnames** enabled in the VPC

Traffic Goes through Public Internet

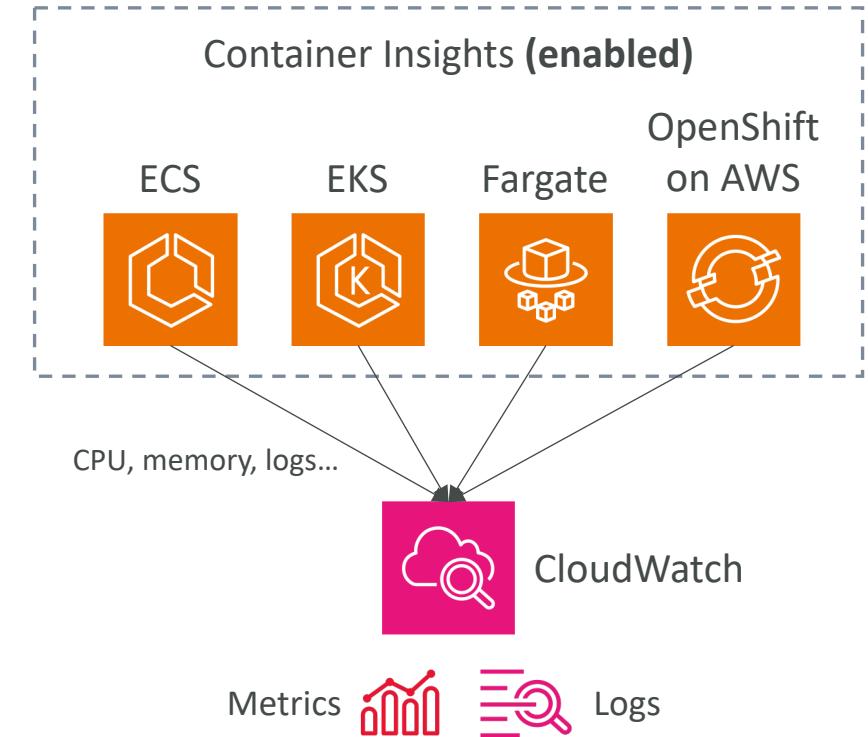


Traffic Goes internally within AWS



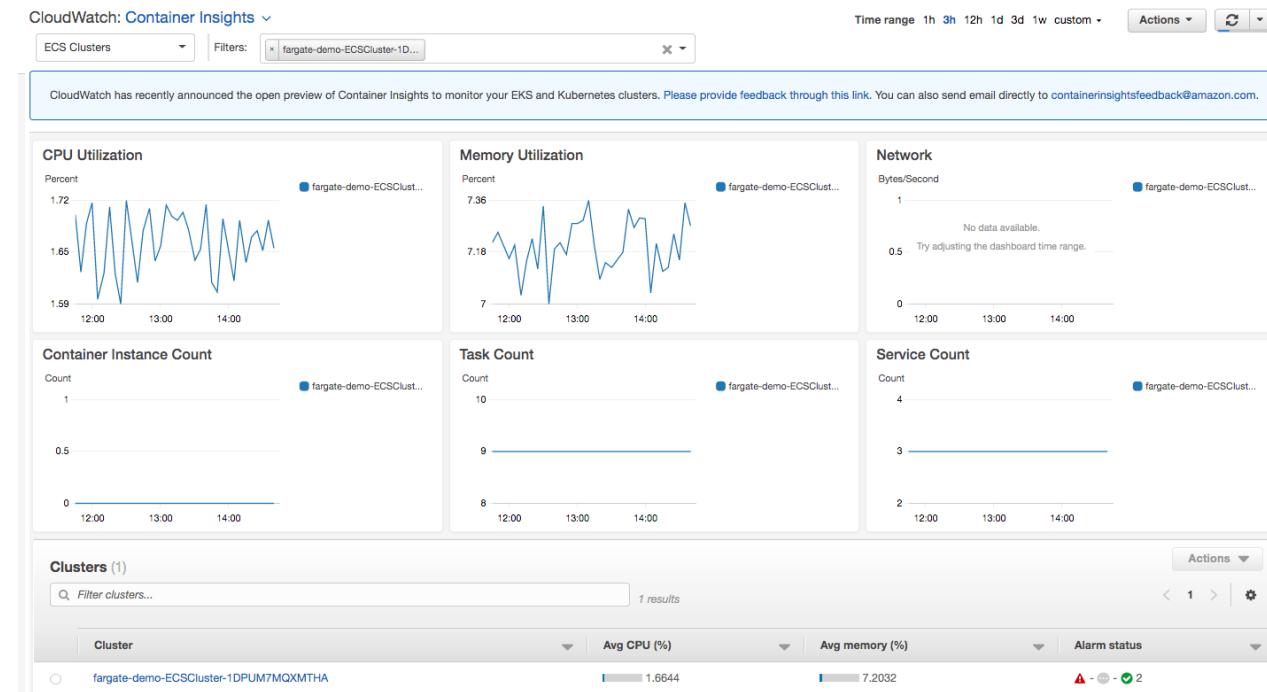
# CloudWatch Container Insights

- Collects, aggregates, summarizes container metrics and logs from your containerized apps and microservices
- Supports ECS, Fargate, EKS, and Redhat OpenShift on AWS (ROSA)
- CPU, memory, tasks, services, network utilization, disk usage...
- No need to use sidecars to monitor your containers
- Can be enabled at account level or when creating a Cluster (cluster-level)



# CloudWatch Container Insights

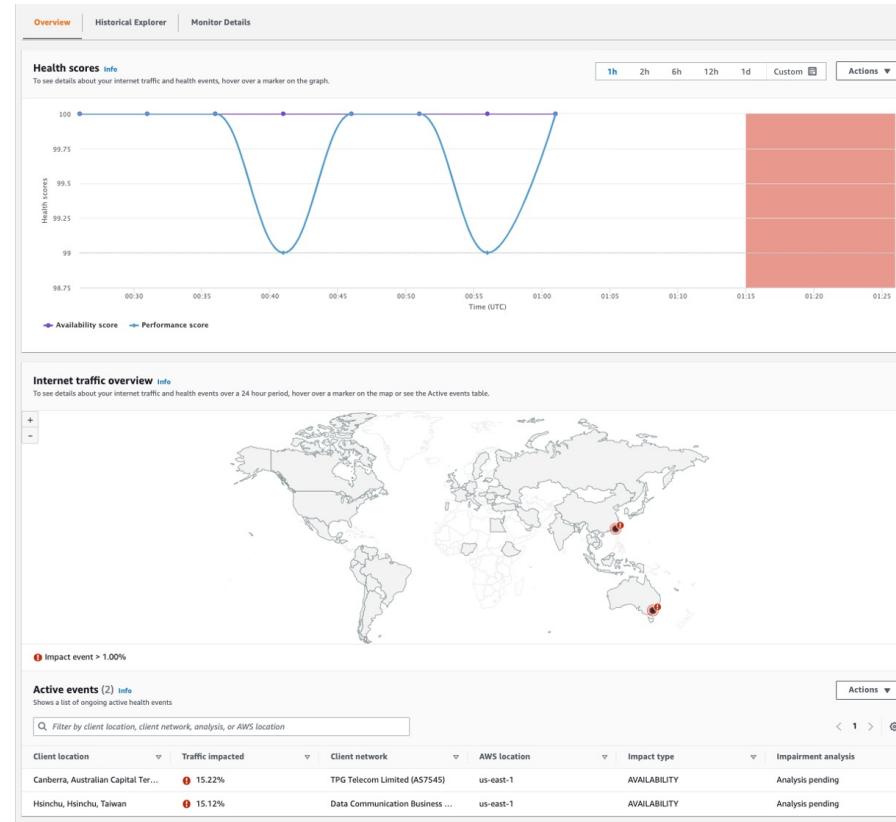
- By default, Container Insights provide metrics at cluster and service level
- Enhanced Visibility – provides metrics at task and container level
- Use cases: high resource usage, throttling issues, performance issues...



<https://aws.amazon.com/blogs/mt/introducing-container-insights-for-amazon-ecs/>

# CloudWatch Internet Monitor

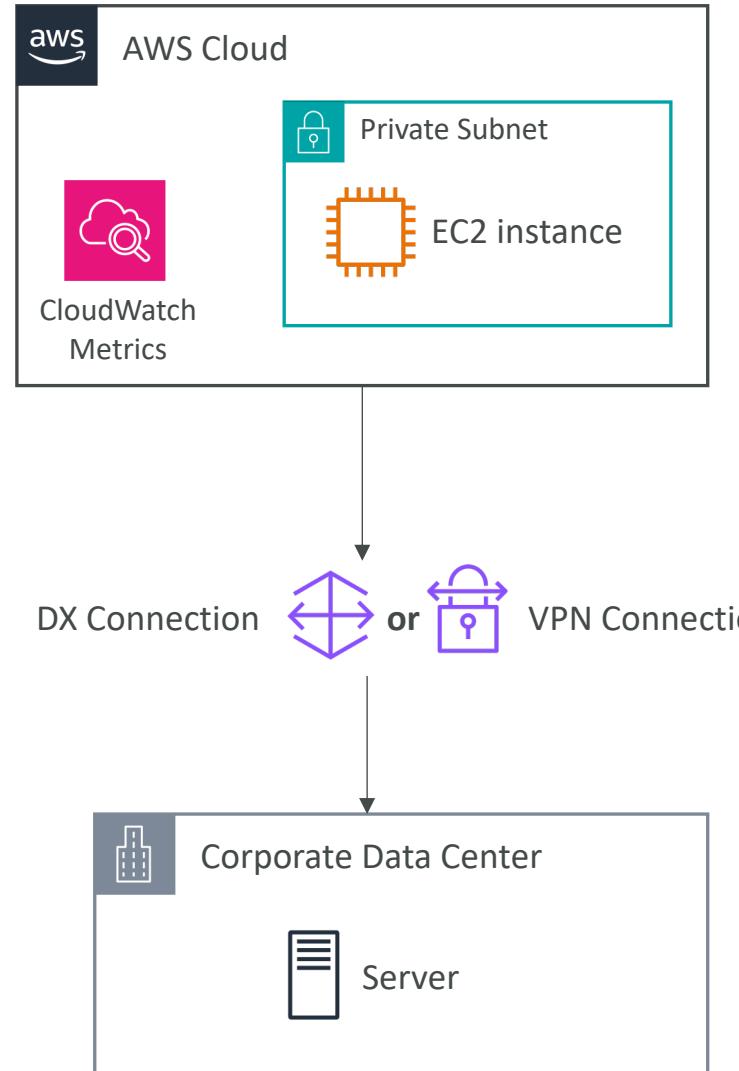
- Monitor and detects how internet issues impact your apps hosted on AWS and end users (city-networks, ASNs, client-locations...)
- Uses Internet data that AWS captures using its global network footprint
- Global view of traffic patterns and health events
- Suggests recommendations to improve end-users experience (e.g., latency)
- Publishes data to CloudWatch Logs & Metrics
- Sends global health events to EventBridge



<https://aws.amazon.com/blogs/networking-and-content-delivery/introducing-amazon-cloudwatch-internet-monitor/>

# CloudWatch Network Synthetic Monitor

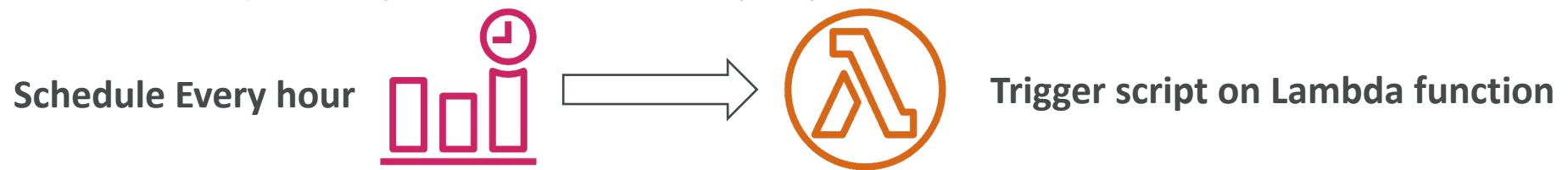
- Monitor and detects network issues between your apps hosted on AWS and your on-premises data center
- Identify any network performance degradation (e.g., packet loss, latency, jitter...)
- No agents required to be installed
- Tests ICMP or TCP traffic to IPv4/IPv4 on-premises destinations through **Direct Connect** or **S2S VPN** connections
- Publishes data to CloudWatch Metrics



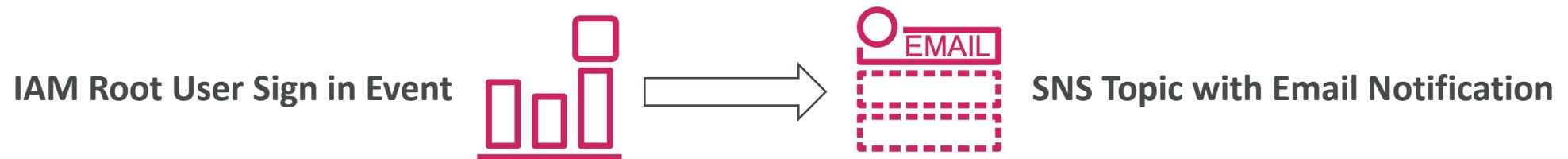
# Amazon EventBridge (formerly CloudWatch Events)



- Schedule: Cron jobs (scheduled scripts)

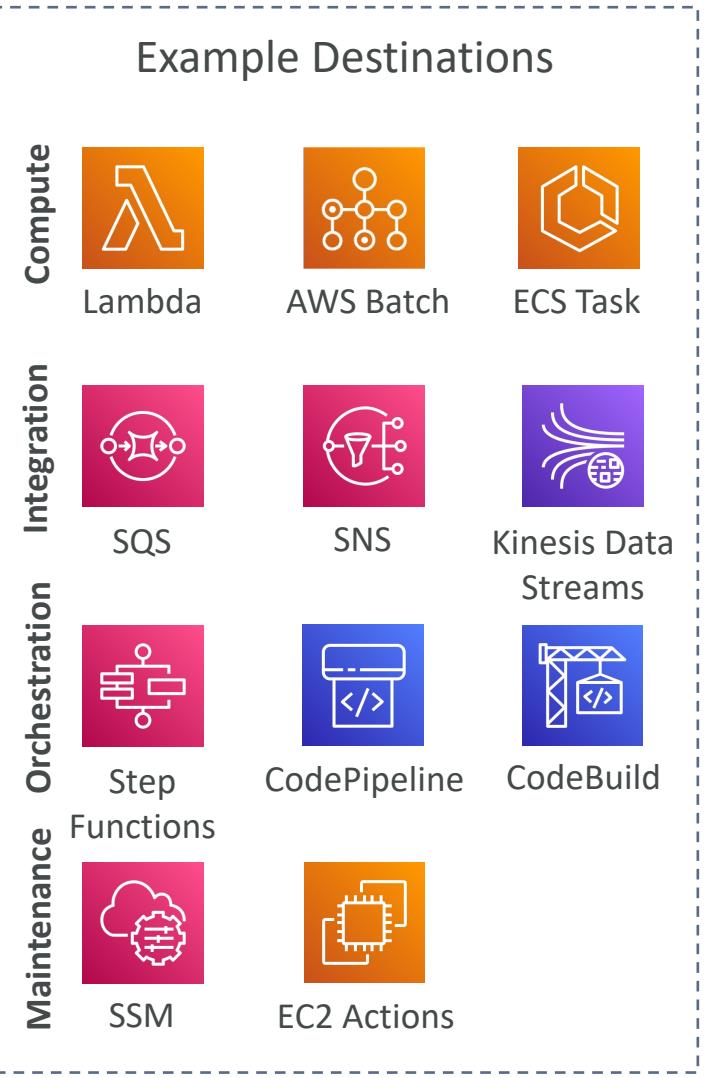
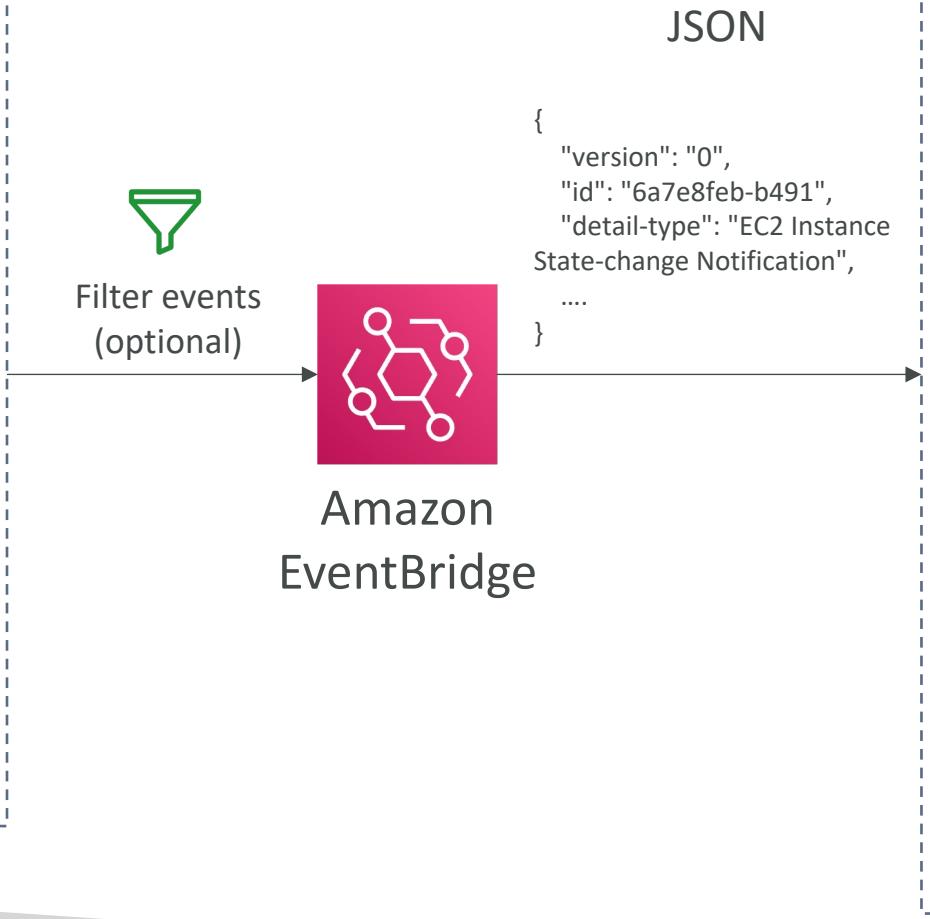
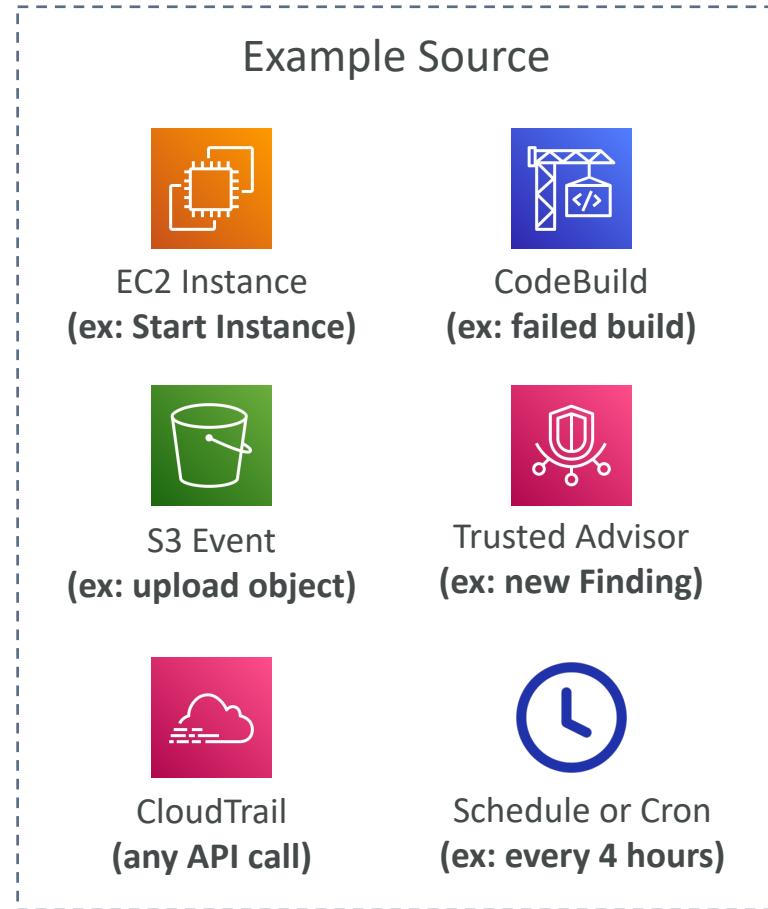


- Event Pattern: Event rules to react to a service doing something

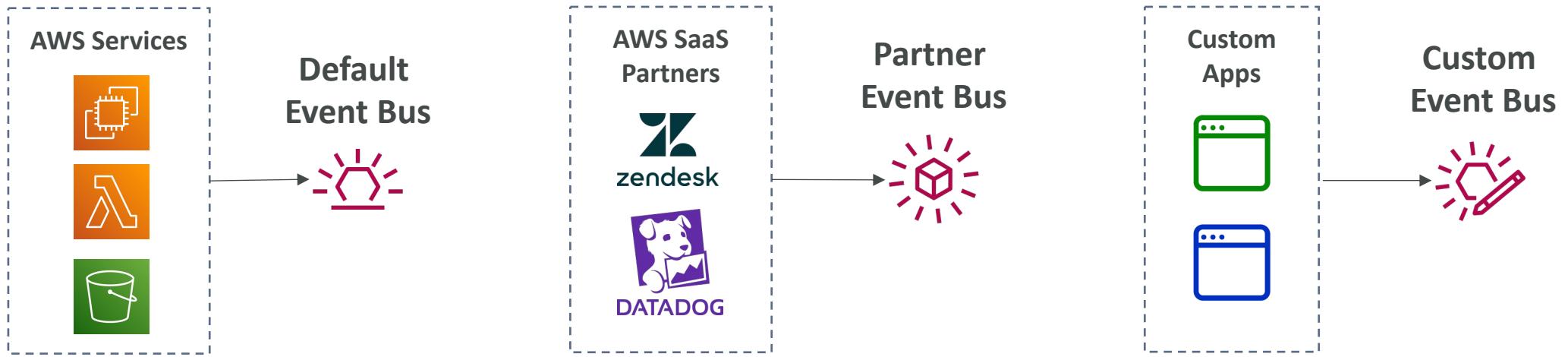


- Trigger Lambda functions, send SQS/SNS messages...

# Amazon EventBridge Rules



# Amazon EventBridge



- Event buses can be accessed by other AWS accounts using Resource-based Policies
- You can archive events (all/filter) sent to an event bus (indefinitely or set period)
- Ability to replay archived events

# Amazon EventBridge – Schema Registry

- EventBridge can analyze the events in your bus and infer the **schema**
- The **Schema Registry** allows you to generate code for your application, that will know in advance how data is structured in the event bus
- Schema can be versioned

The screenshot shows the AWS Schema Registry interface. At the top, there's a header with the URL "aws.codepipeline@CodePipelineActionExecutionStateChange". Below it, a section titled "Schema details" provides the following information:

Schema name	Last modified	Schema ARN
aws.codepipeline@CodePipelineActionExecutionStateChange	Dec 1, 2019, 12:11 AM GMT	-
Description	Schema for event type CodePipelineActionExecutionStateChange, published by AWS service aws.codepipeline	Schema registry aws.events Number of versions 1 Schema type OpenAPI 3.0

Below this, a section titled "Version 1" shows the schema definition:

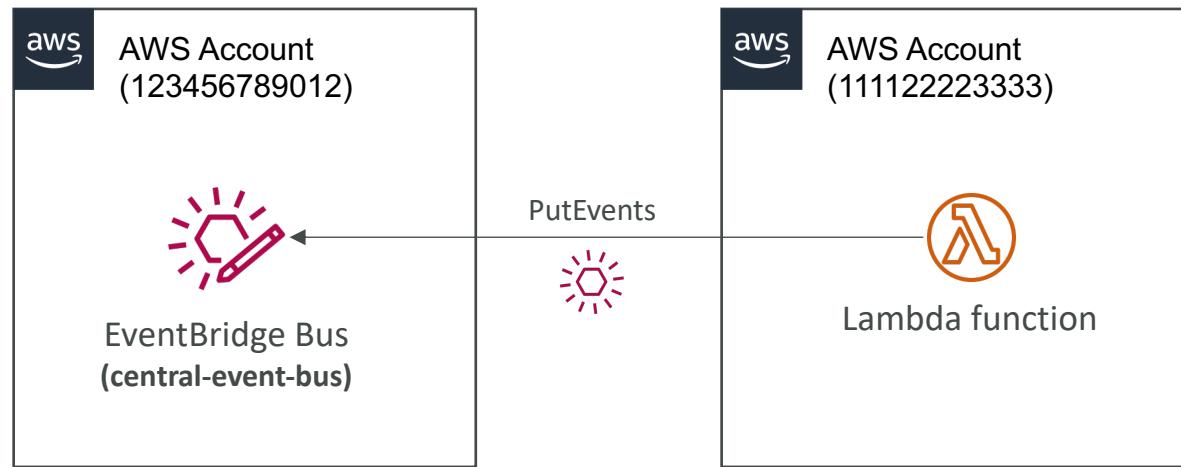
```
1 {
2   "openapi": "3.0.0",
3   "info": {
4     "version": "1.0.0",
5     "title": "CodePipelineActionExecutionStateChange"
6   },
7   "paths": {},
8   "components": {
9     "schemas": {
10       "AWSEvent": {
```

# Amazon EventBridge – Resource-based Policy

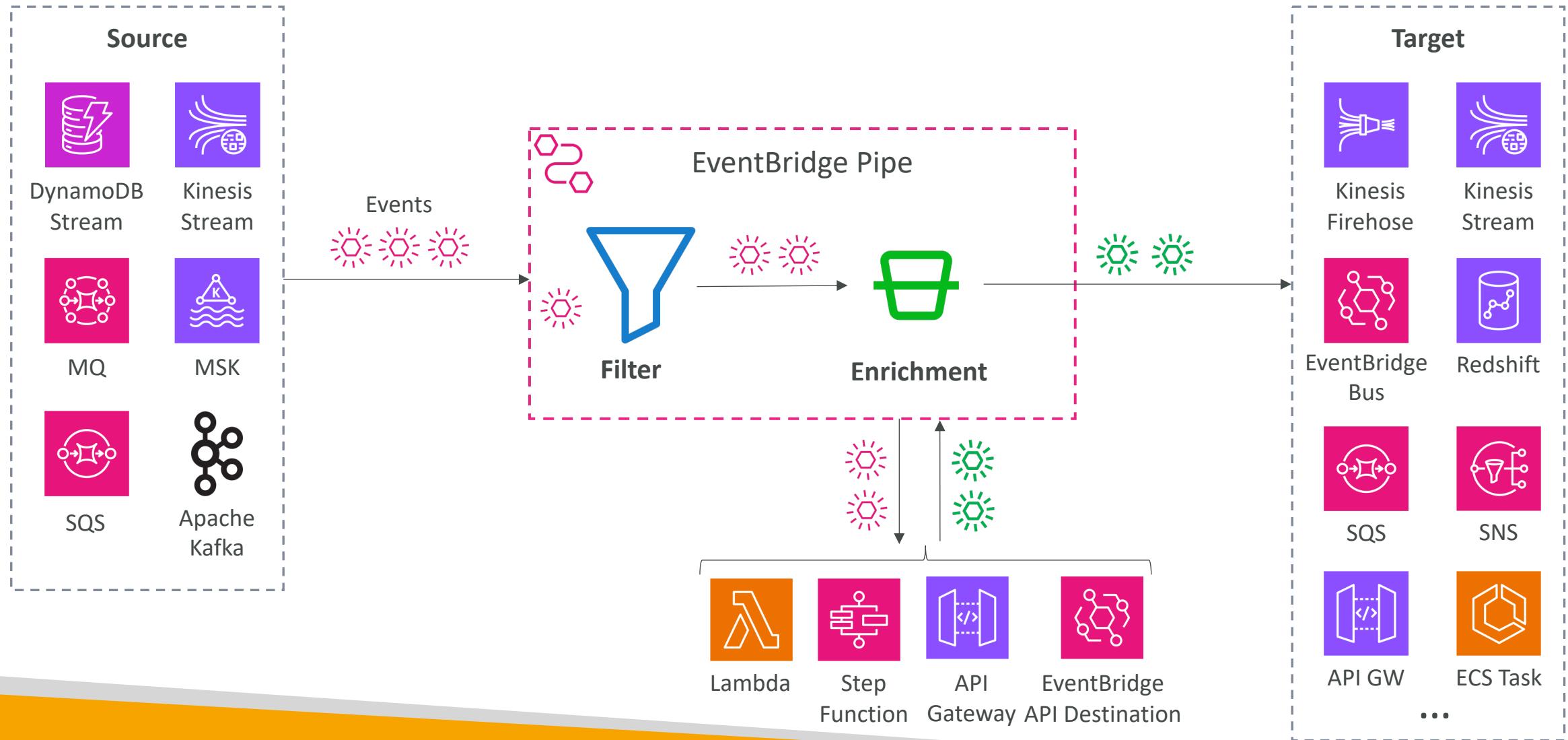
- Manage permissions for a specific Event Bus
- Example: allow/deny events from another AWS account or AWS region
- Use case: aggregate all events from your AWS Organization in a single AWS account or AWS region

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "events:PutEvents",  
            "Principal": { "AWS": "111122223333" },  
            "Resource": "arn:aws:events:us-east-1:123456789012:  
event-bus/central-event-bus"  
        }  
    ]  
}
```

Allow **events** from another AWS account



# Amazon EventBridge – Pipes



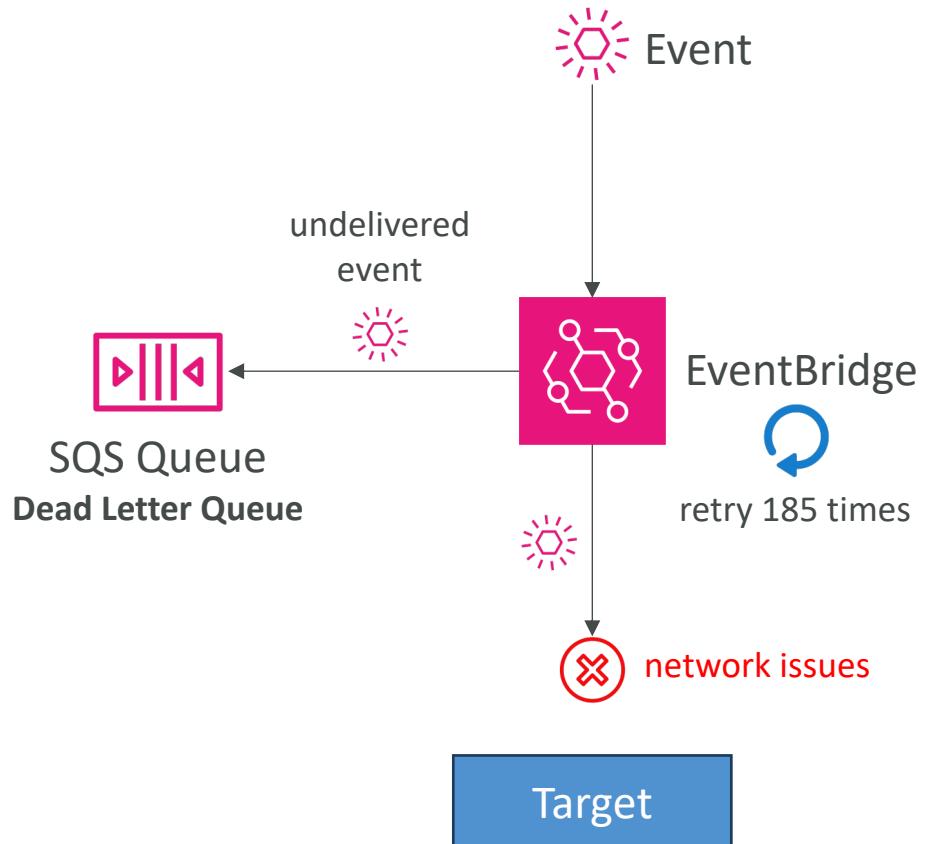


# Amazon EventBridge – Pipes

- Out-of-the box integration between data sources and targets in AWS
- No-code solution
- **Filtering** – filter selected subset of events
- **Enrichment** – process & enhance data before sending events to the target
- You can process events using:
  - Lambda
  - Step Functions
  - API Gateway
  - EventBridge API Destination

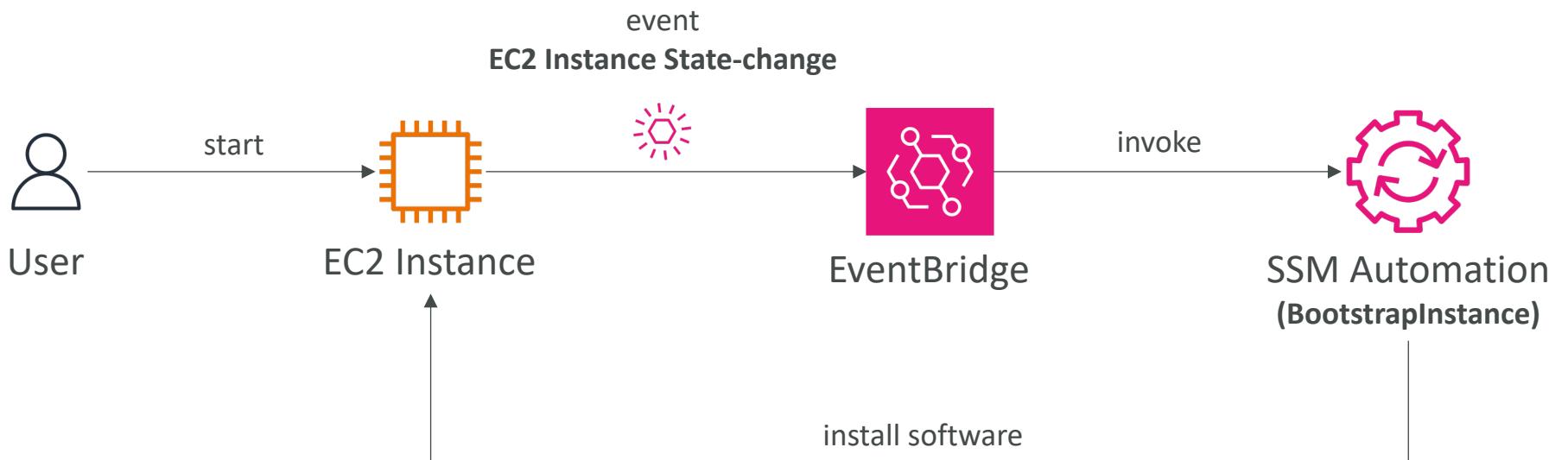
# Amazon EventBridge – Retries & DLQs

- Sometimes events can't be delivered to a target due to:
  - Target not being available
  - Network issues
- You define a **Retry Policy** by configuring:
  - Length of Time (default: 24 hours)
  - Retry Attempts (default: 185)
- You can send undelivered events to **Dead Letter Queue (DLQ)** using SQS for later processing



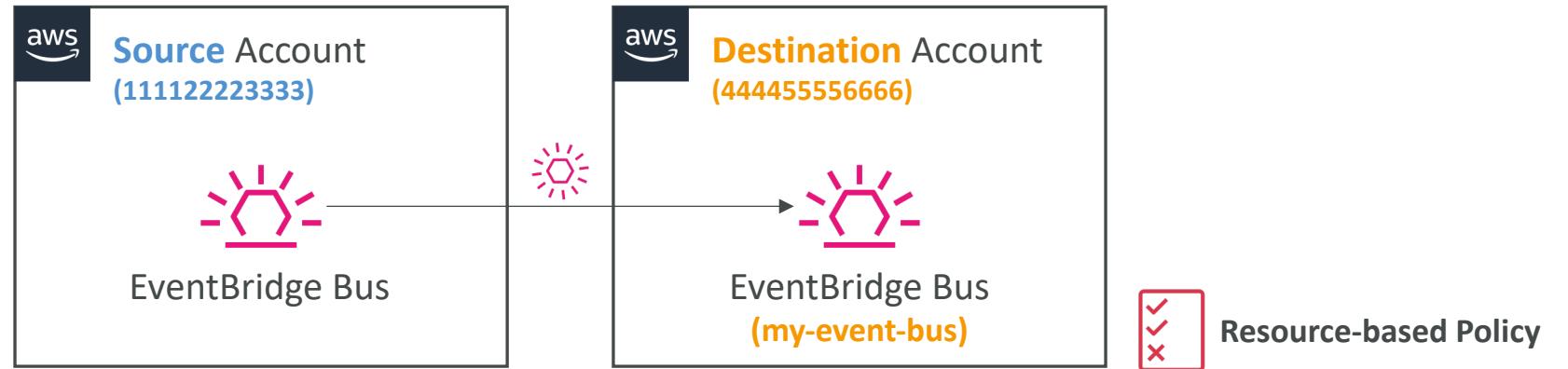
# Amazon EventBridge – Trigger SSM Automation

- You can specify SSM Automations as a target of EventBridge event
- Use on a schedule or when a specific event occurs



# Amazon EventBridge – Cross-account Targets

## EventBridge Bus as Target

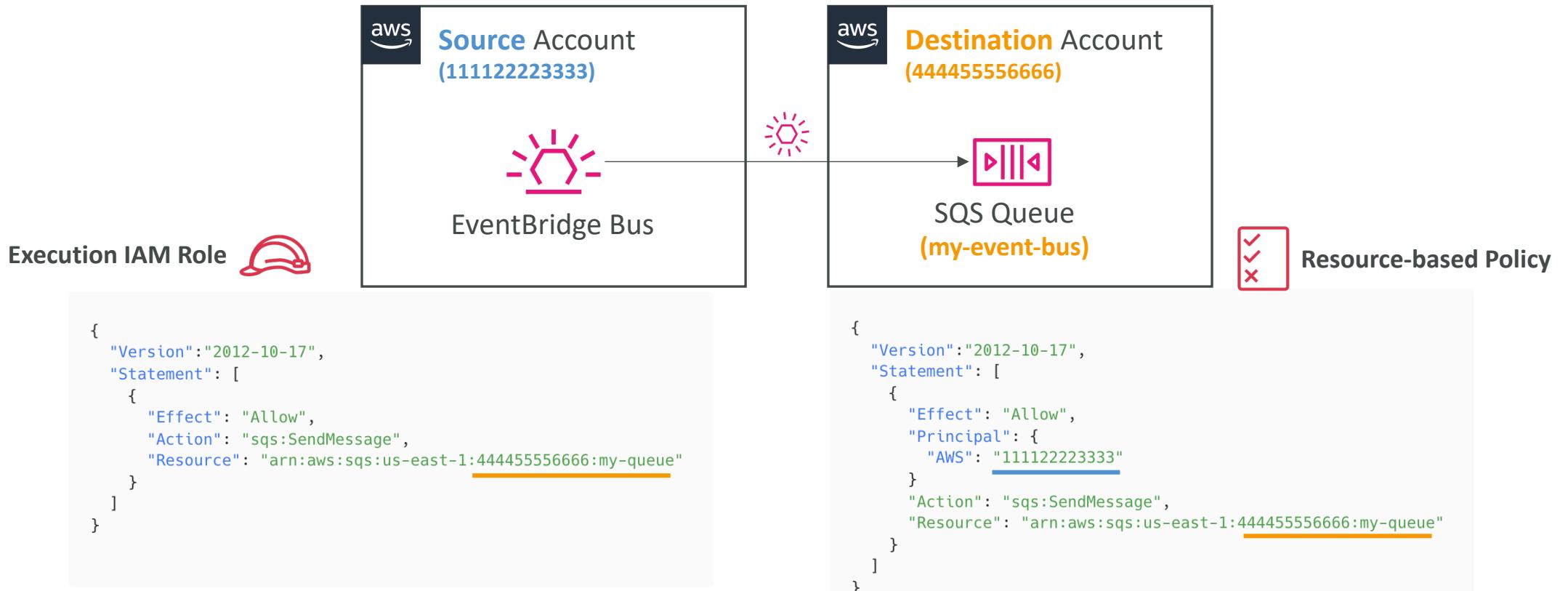


```
{  
  "Version": "2012-10-17",  
  "Statement": [{  
    "Effect": "Allow",  
    "Action": "events:PutEvents",  
    "Resource": "arn:aws:events:eu-west-1:444455556666:event-bus/my-event-bus"  
  }]  
}
```

```
{  
  "Version": "2012-10-17",  
  "Statement": [{  
    "Effect": "Allow",  
    "Principal": {  
      "AWS": "arn:aws:iam::111122223333:root"  
    },  
    "Action": "events:PutEvents",  
    "Resource": "arn:aws:events:eu-west-1:444455556666:event-bus/my-event-bus"  
  }]  
}
```

# Amazon EventBridge – Cross-account Targets

## Other Services as Targets

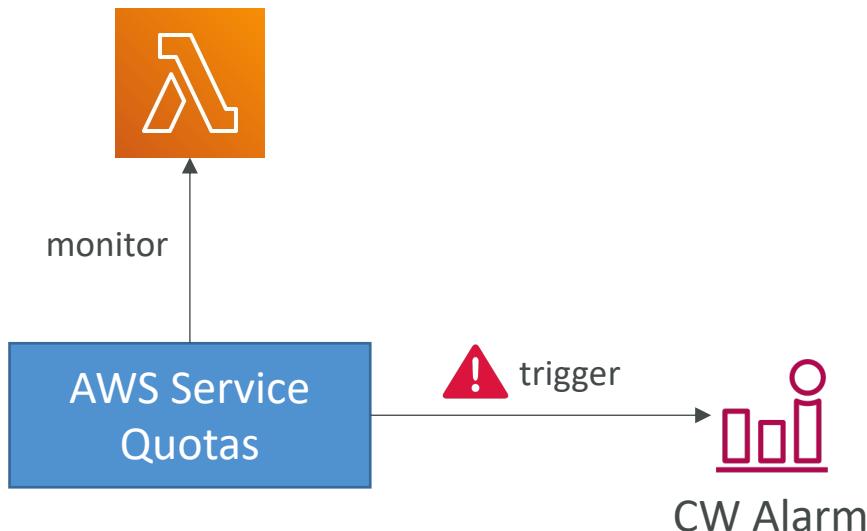


**Supported Targets:** SQS, SNS, Lambda, API Gateway, Kinesis Data Streams

# Service Quotas CloudWatch Alarms

- Notify you when you're close to a service quota value threshold
- Create CloudWatch Alarms on the Service Quotas console
- Example: Lambda concurrent executions
- Helps you know if you need to request a quota increase or shutdown resources before limit is reached

AWS Lambda Quota



**Create a CloudWatch alarm: Concurrent executions**

Description  
The maximum number of events that functions can process simultaneously in the current Region.

Alarm threshold  
This alarm will notify you based on the threshold you choose.

Alarm name  
  
Required. Alarm names must be unique within an AWS account.

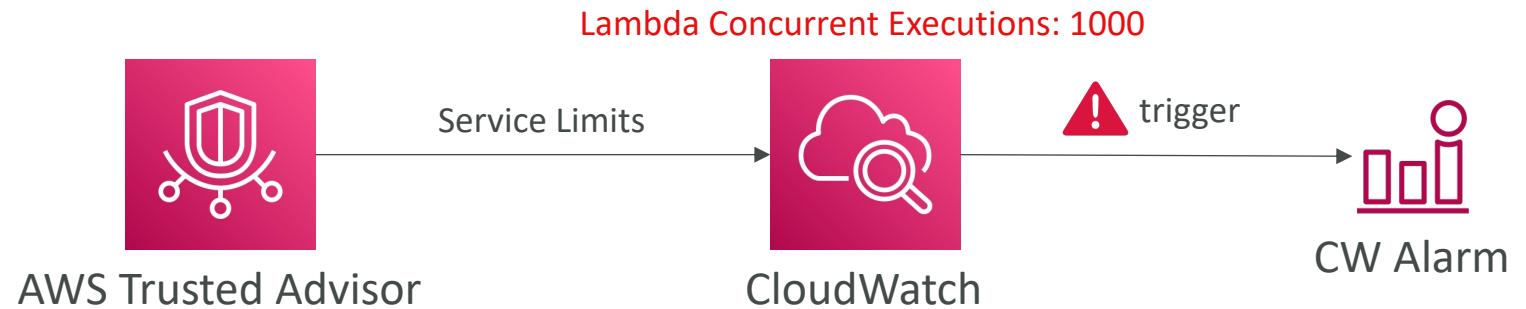
Region  
US East (N. Virginia) us-east-1

Pricing  
Using CloudWatch can incur costs. [CloudWatch pricing](#)

[Cancel](#) [Create](#)

# Alternative: Trusted Advisor + CW Alarms

- Limited number of Service Limits checks in Trusted Advisor (~50)
- Trusted Advisor publishes its check results to CloudWatch
- You can create CloudWatch Alarms on service quota usage (Service Limits)

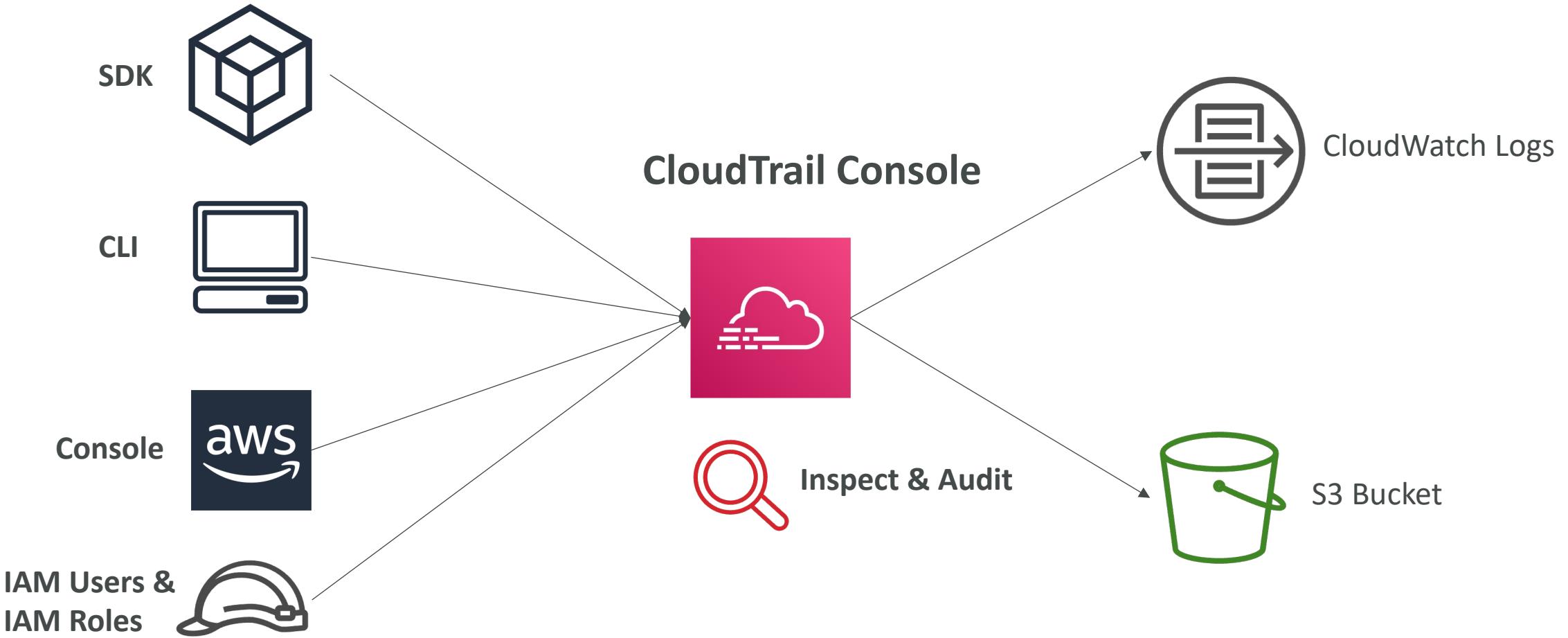




# AWS CloudTrail

- 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





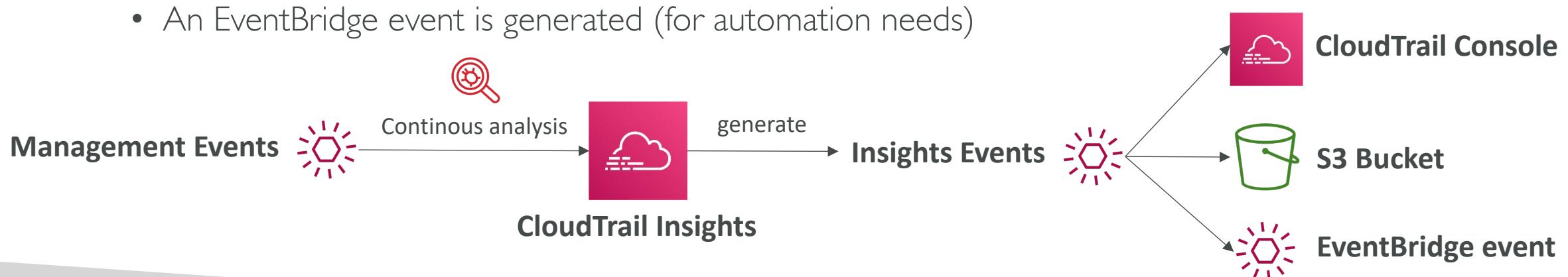
# CloudTrail Events

- 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 ☺



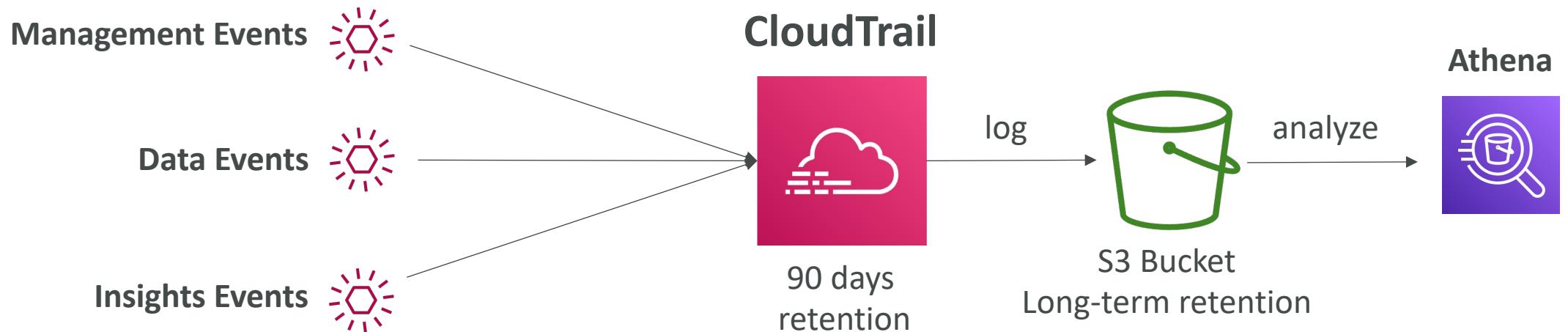
# CloudTrail Insights

- 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)

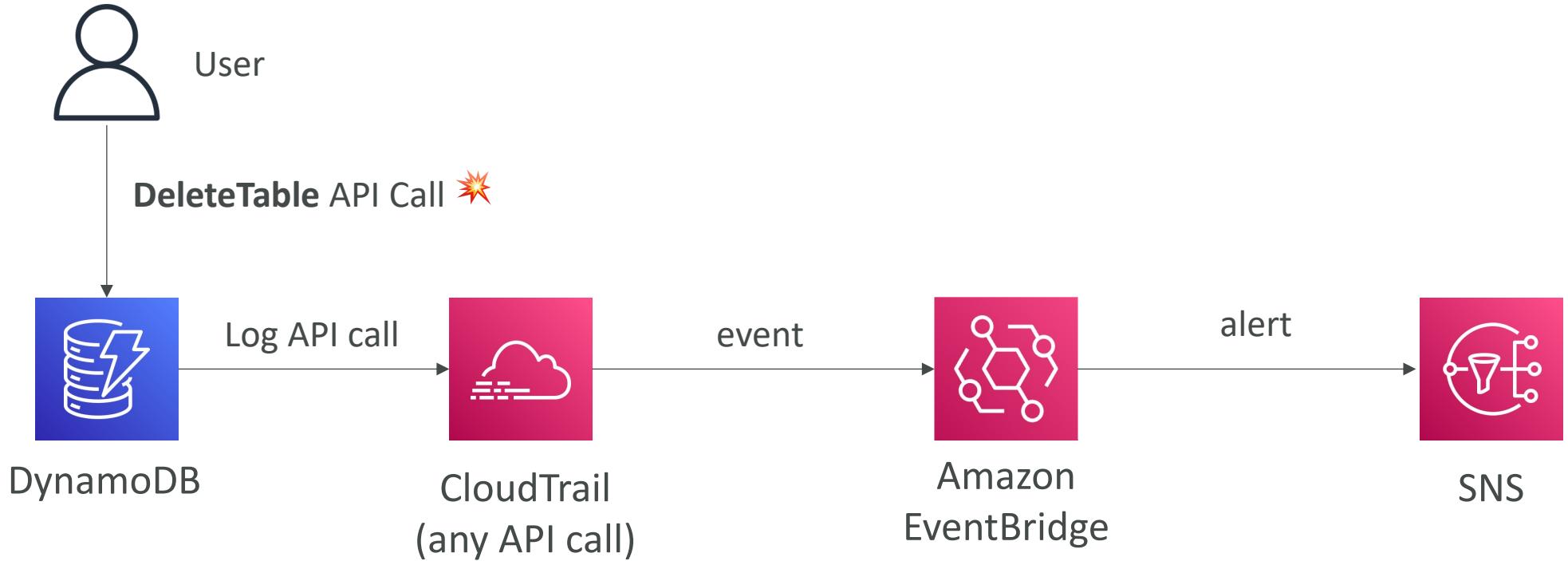


# CloudTrail Events Retention

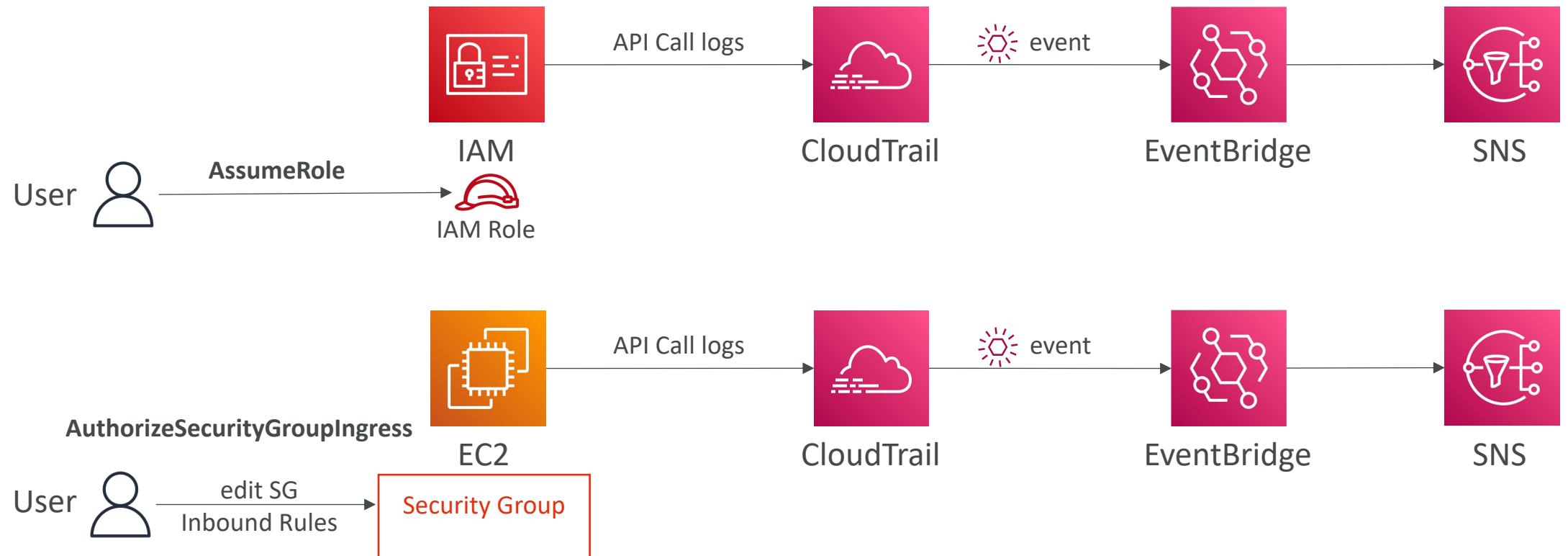
- Events are stored for 90 days in CloudTrail
- To keep events beyond this period, log them to S3 and use Athena



# Amazon EventBridge – Intercept API Calls

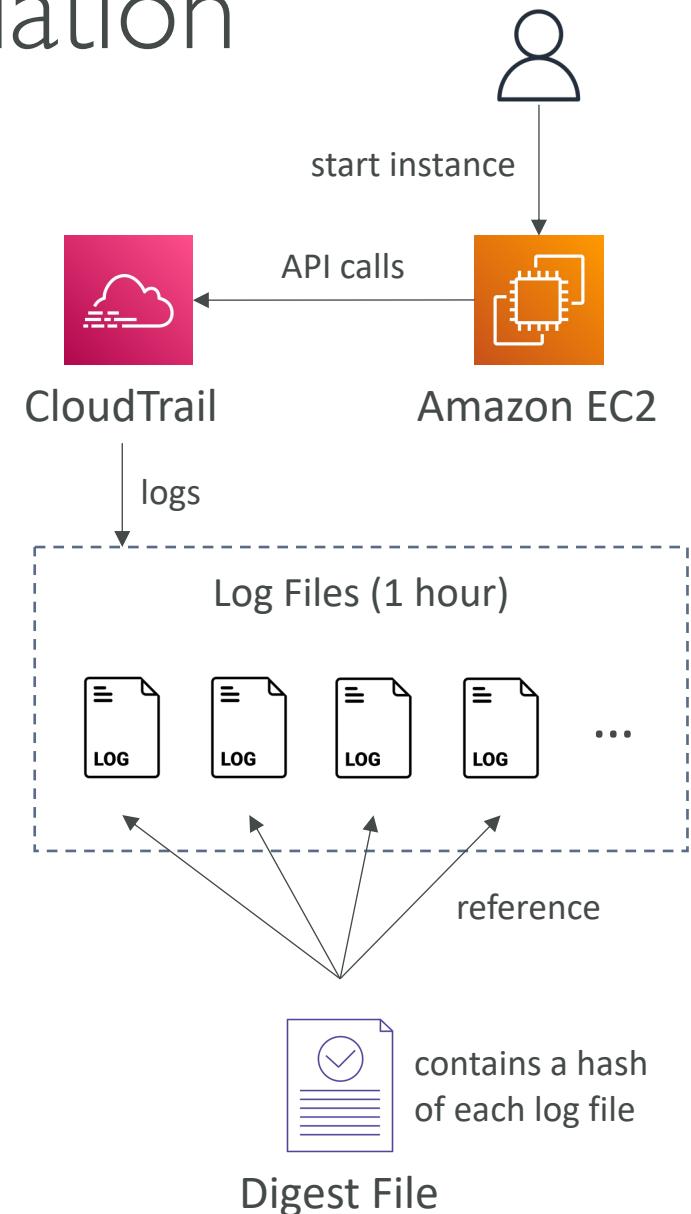


# Amazon EventBridge + CloudTrail

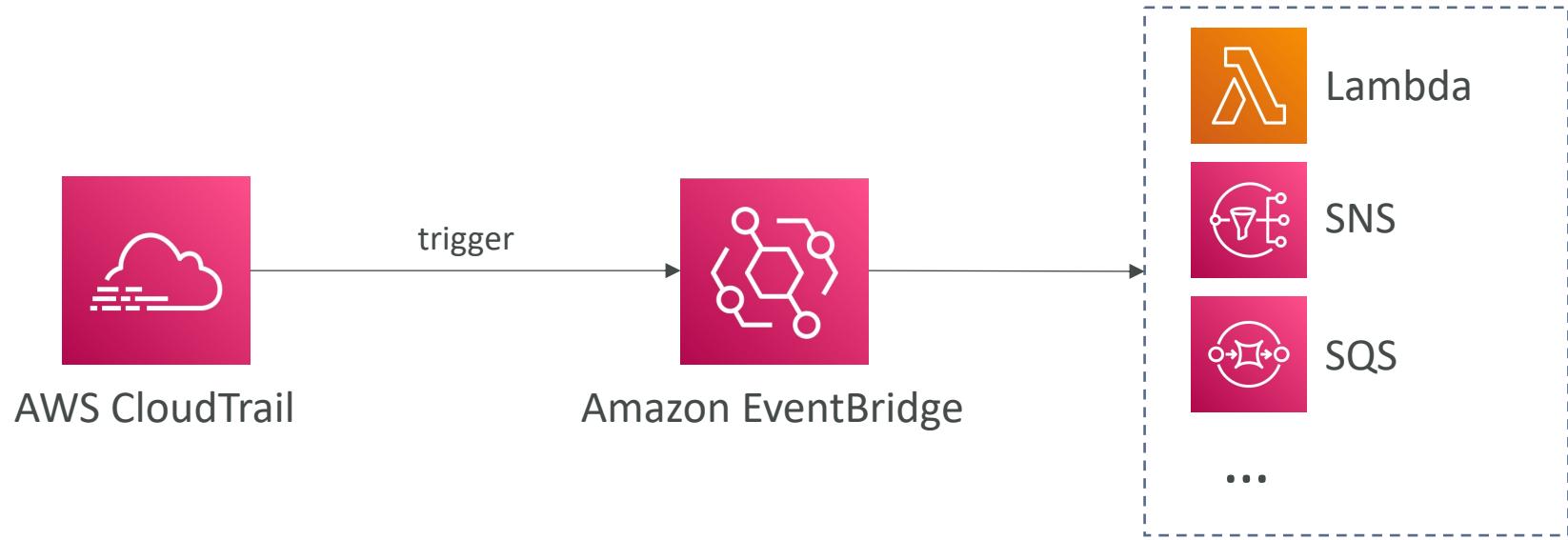


# CloudTrail – Log File Integrity Validation

- **Digest Files:**
  - References the log files for the last hour and contains a hash of each
  - Stored in the same S3 bucket as log files (different folder)
- Helps you determine whether a log file was modified/deleted after CloudTrail delivered it
- Hashing using SHA-256, Digital Signing using SHA-256 with RSA
- Protect the S3 bucket using bucket policy, versioning, MFA Delete protection, encryption, object lock
- Protect CloudTrail using IAM



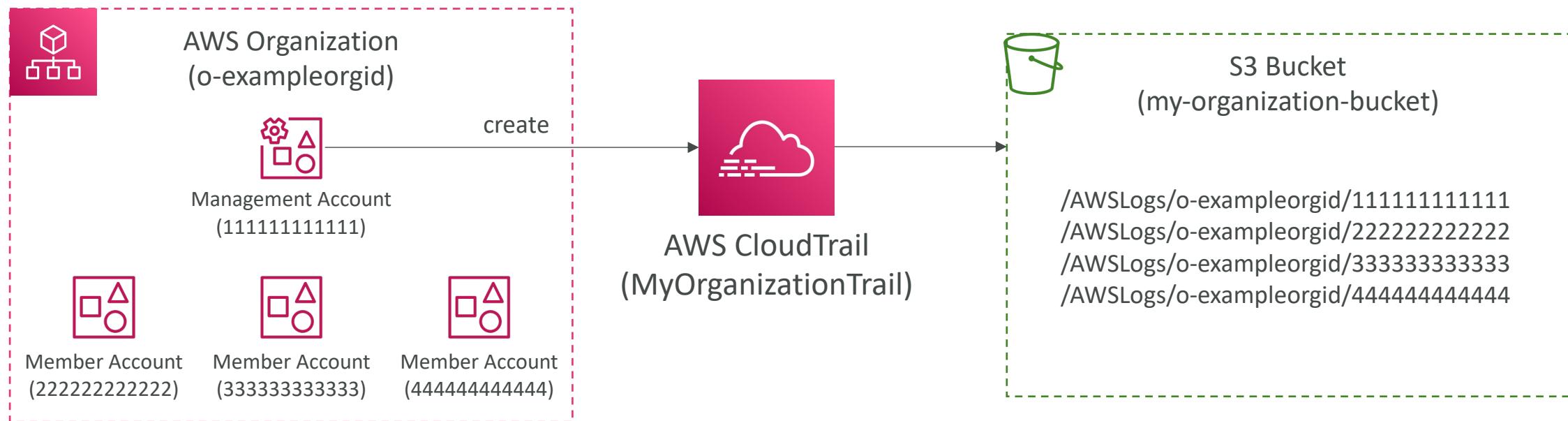
# CloudTrail – Integration with EventBridge

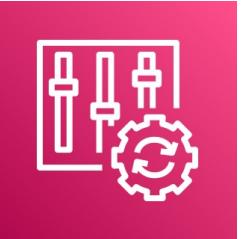


- Used to react to any API call being made in your account
- CloudTrail is not “real-time”:
  - Delivers an event within 15 minutes of an API call
  - Delivers log files to an S3 bucket every 5 minutes

# CloudTrail – Organizations Trails

- A trail that will log all events for all AWS accounts in an AWS Organization
- Log events for management and member accounts
- Trail with the same name will be created in every AWS account (IAM permissions)
- Member accounts can't remove or modify the organization trail (view only)





# AWS Config

- Helps with auditing and recording **compliance** of your AWS resources
- Helps record configurations and changes over time
- 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
- Possibility of storing the configuration data into S3 (analyzed by Athena)

# Config Rules

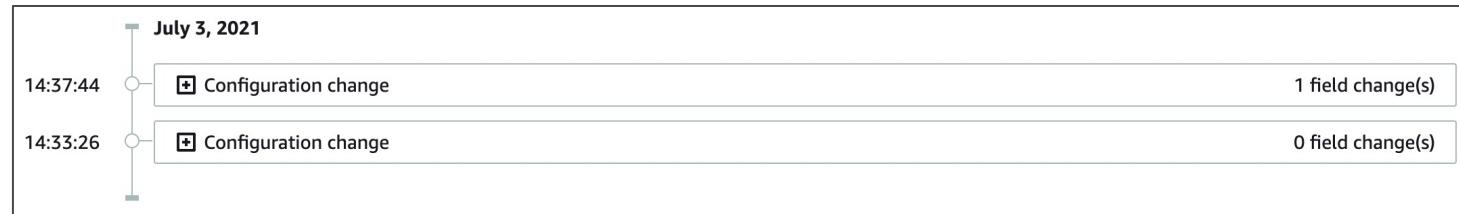
- Can use AWS managed config rules (over 75)
- Can make custom config rules (must be defined in AWS Lambda)
  - Ex: evaluate if each EBS disk is of type gp2
  - Ex: evaluate if each EC2 instance is t2.micro
- Rules can be evaluated / triggered:
  - For each config change
  - And / or: at regular time intervals
- AWS Config Rules does not prevent actions from happening (no deny)
- Pricing: no free tier, \$0.003 per configuration item recorded per region, \$0.001 per config rule evaluation per region

# AWS Config Resource

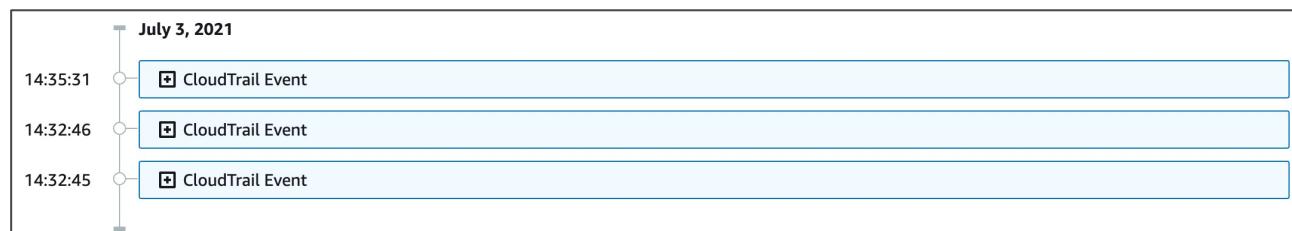
- View compliance of a resource over time

<input type="radio"/> sg-077b425b1649da83e	EC2 SecurityGroup	 Compliant
<input type="radio"/> sg-0831434f1876c0c74	EC2 SecurityGroup	 Noncompliant
<input type="radio"/> sg-09f10ed254d464f30	EC2 SecurityGroup	 Compliant

- View configuration of a resource over time

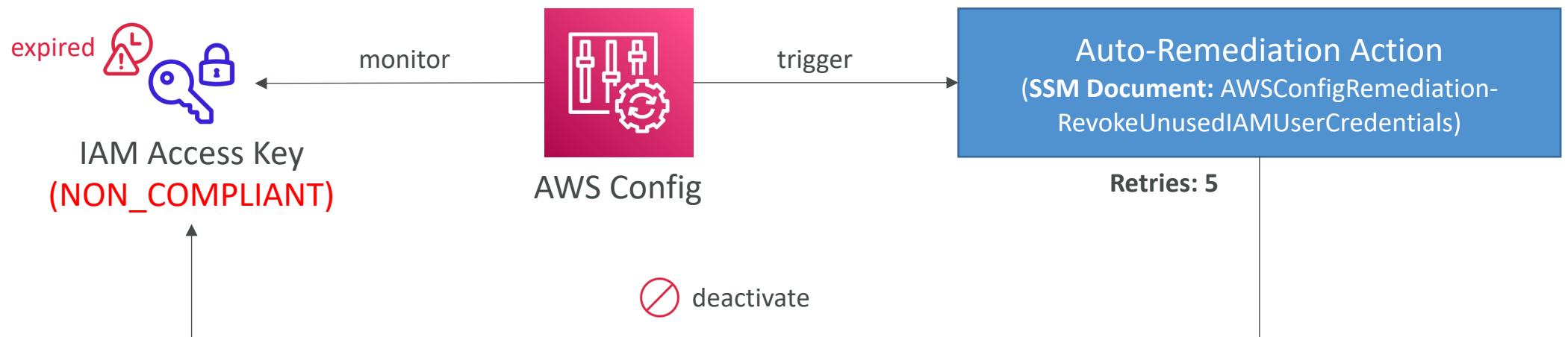


- View CloudTrail API calls of a resource over time



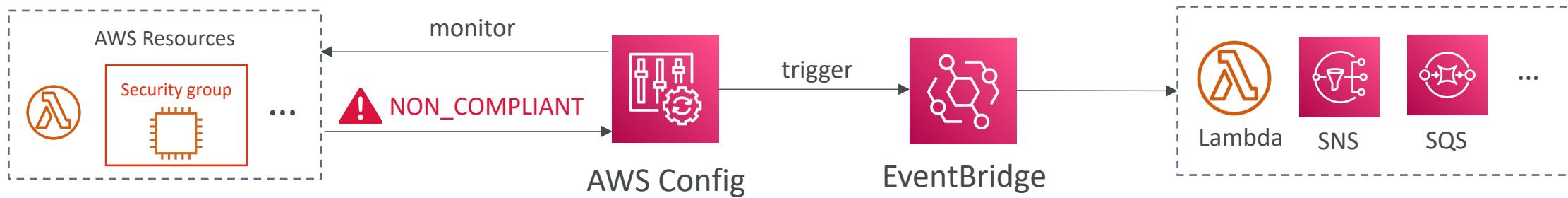
# Config Rules – Remediations

- Automate remediation of non-compliant resources using SSM Automation Documents
- Use AWS-Managed Automation Documents or create custom Automation Documents
  - Tip: you can create custom Automation Documents that invokes Lambda function
- You can set **Remediation Retries** if the resource is still non-compliant after auto-remediation

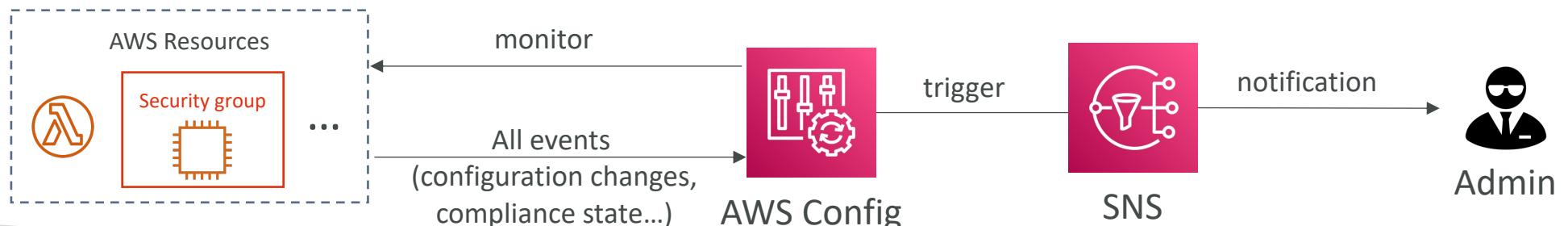


# Config Rules – Notifications

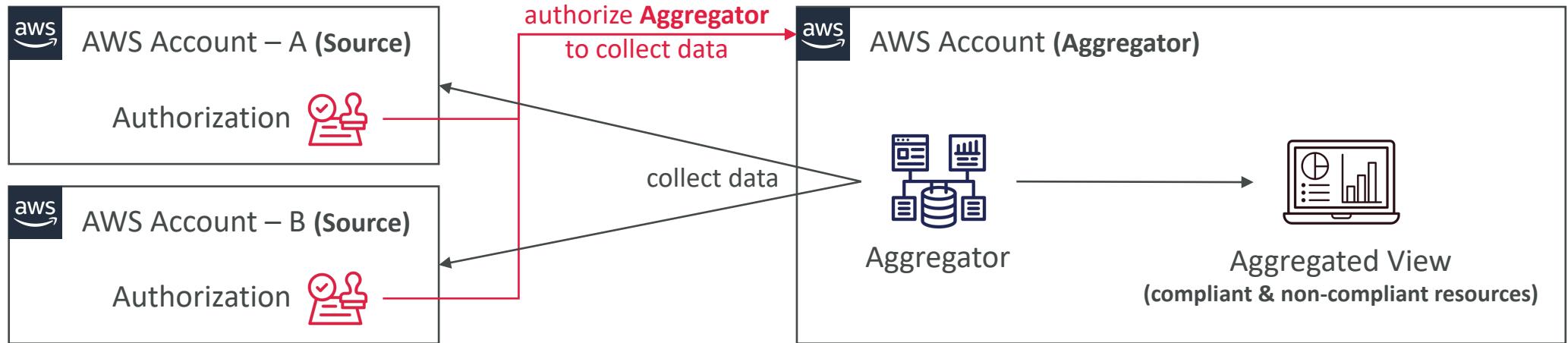
- Use EventBridge to trigger notifications when AWS resources are non-compliant



- Ability to send configuration changes and compliance state notifications to SNS (all events – use SNS Filtering or filter at client-side)



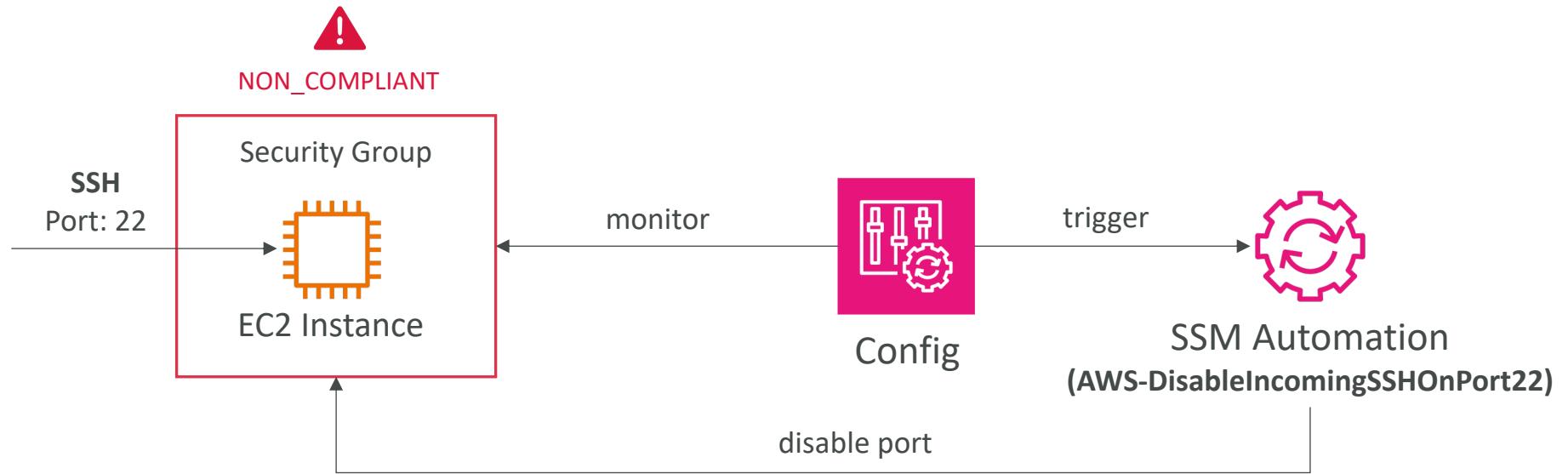
# AWS Config – Aggregators



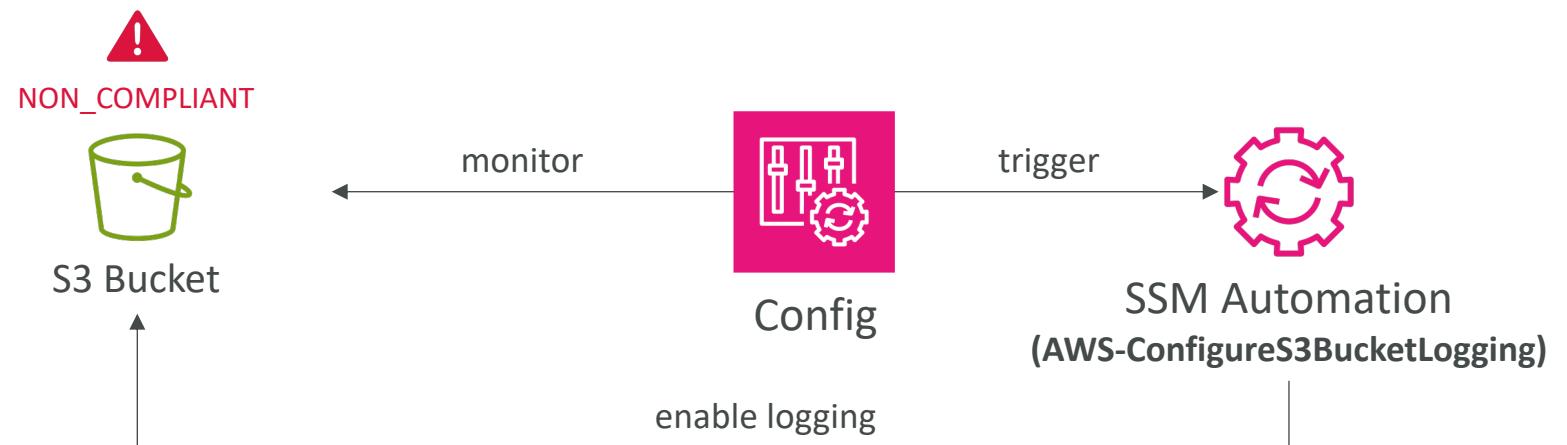
- The aggregator is created in one central aggregator account
- Aggregates rules, resources, etc... across multiple accounts & regions
- If using AWS Organizations, no need for individual Authorization
- Rules are created in each individual source AWS account
- Can deploy rules to multiple target accounts using CloudFormation StackSets

# AWS Config – Remediation Examples

## Disabling Incoming SSH Traffic



## Enable Logging on S3 Bucket



# CloudWatch vs CloudTrail vs Config

- CloudWatch
  - Performance monitoring (metrics, CPU, network, etc...) & dashboards
  - Events & Alerting
  - Log Aggregation & Analysis
- CloudTrail
  - Record API calls made within your Account by everyone
  - Can define trails for specific resources
  - Global Service
- Config
  - Record configuration changes
  - Evaluate resources against compliance rules
  - Get timeline of changes and compliance

# For an Elastic Load Balancer

- CloudWatch:
  - Monitoring Incoming connections metric
  - Visualize error codes as a % over time
  - Make a dashboard to get an idea of your load balancer performance
- Config:
  - Track security group rules for the Load Balancer
  - Track configuration changes for the Load Balancer
  - Ensure an SSL certificate is always assigned to the Load Balancer (compliance)
- CloudTrail:
  - Track who made any changes to the Load Balancer with API calls

# AWS Account Management

Health Dashboards, AWS Organizations and Billing Console

# AWS Health Dashboard - Service History



- Shows all regions, all services health
- Shows historical information for each day
- Has an RSS feed you can subscribe to
- Previously called AWS Service Health Dashboard

Service history									
The following table is a running log of AWS service interruptions for the past 12 months. Choose a status icon to see status updates for that service. All dates and times are reported in UTC. To update your time zone, see <a href="#">Time zone settings</a> .									
Find an AWS service or Region		2023/01/10							
North America	South America	Europe	Africa	Asia Pacific	Middle East				
Alexa for Business (N. Virginia)	RSS	Today	9 Jan	8 Jan	7 Jan	6 Jan	5	4	3
Amazon EventBridge Scheduler (N. Virginia)	RSS	Today	9 Jan	8 Jan	7 Jan	6 Jan	5	4	3
Amazon EventBridge Scheduler (Ohio)	RSS	Today	9 Jan	8 Jan	7 Jan	6 Jan	5	4	3
Amazon EventBridge Scheduler (Oregon)	RSS	Today	9 Jan	8 Jan	7 Jan	6 Jan	5	4	3
Amazon API Gateway (Montreal)	RSS	Today	9 Jan	8 Jan	7 Jan	6 Jan	5	4	3
Amazon API Gateway (N. California)	RSS	Today	9 Jan	8 Jan	7 Jan	6 Jan	5	4	3
Amazon API Gateway (N. Virginia)	RSS	Today	9 Jan	8 Jan	7 Jan	6 Jan	5	4	3
Amazon API Gateway (Ohio)	RSS	Today	9 Jan	8 Jan	7 Jan	6 Jan	5	4	3

# AWS Health Dashboard – Your Account



- Previously called AWS Personal Health Dashboard (PHD)
- AWS Account 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, Account 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.
- Can aggregate data from an entire AWS Organization

# AWS Health Dashboard – Your Account



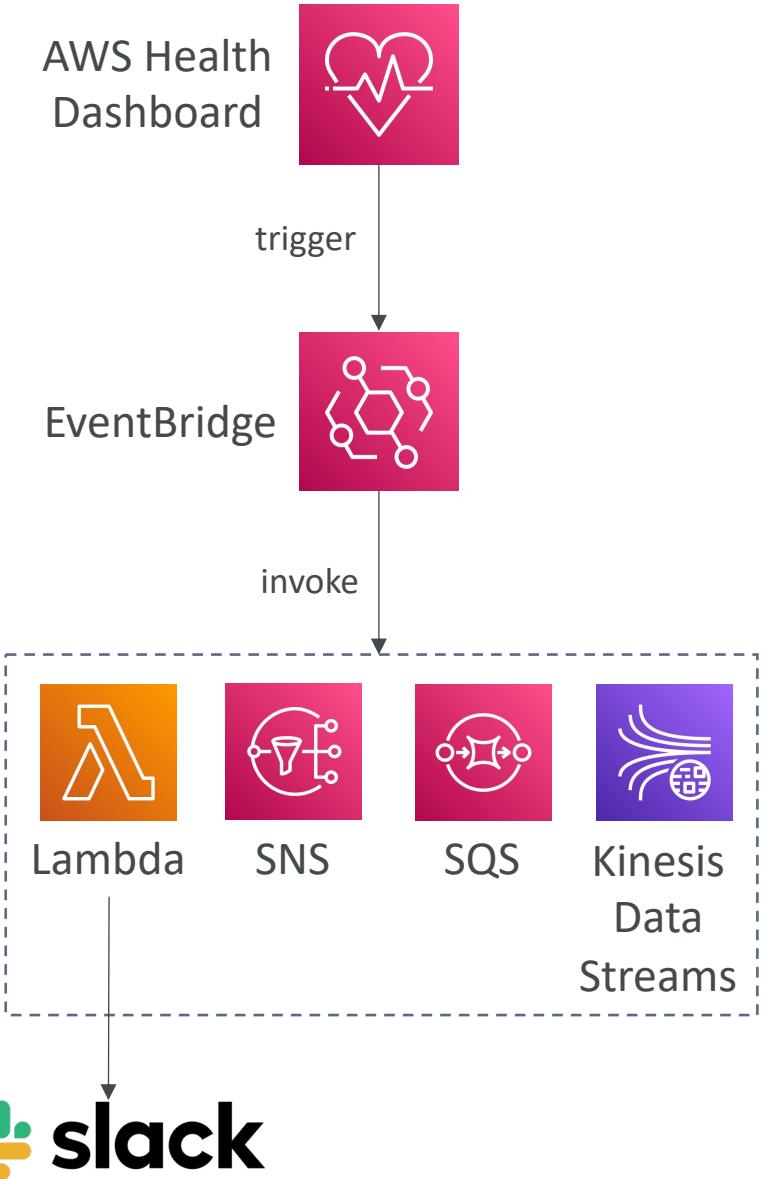
- Global service
- Shows how AWS outages directly impact you & your AWS resources
- Alert, remediation, proactive, scheduled activities

Open issues	0
Scheduled changes	0
Other notifications	0
Event log	

Event log						
Event	Status	Event category	Region / Zone	Start time	Last update time	Affected resources
Operational issue - EC2 (Ohio)	Closed	Issue	us-east-2	December 24, 2022 at 2:25:00 AM UTC	December 24, 2022 at 2:38:53 AM UTC	-
Operational issue - Codestar (Oregon)	Closed	Issue	us-west-2	December 21, 2022 at 3:03:57 PM UTC	December 21, 2022 at 4:50:47 PM UTC	-
Operational issue - Amplify (N. Virginia)	Closed	Issue	us-east-1	December 17, 2022 at 2:24:17 PM UTC	December 17, 2022 at 2:43:21 PM UTC	-
Operational issue - Multiple services (Singapore)	Closed	Issue	ap-southeast-1	December 13, 2022 at 10:00:55 PM UTC	December 14, 2022 at 1:01:16 AM UTC	-

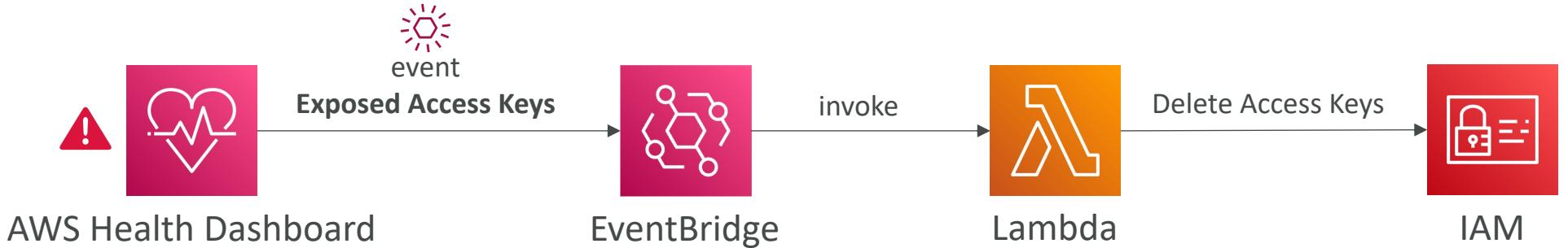
# Health Event Notifications

- Use EventBridge to react to changes for AWS Health events in your AWS account
- Example: receive email notifications when EC2 instances in your AWS account are scheduled for updates
- This is possible for Account events (resources that are affected in your account) and Public Events (Regional availability of a service)
- Use cases: send notifications, capture event information, take corrective action...

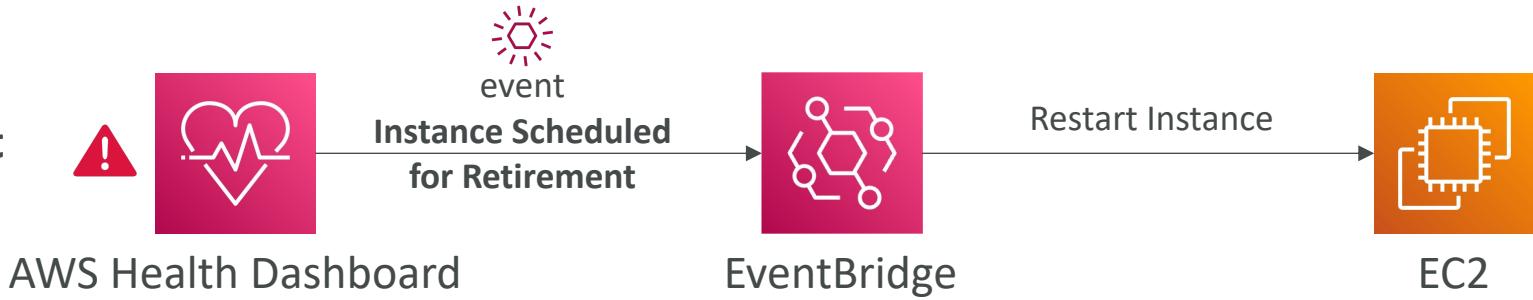


# AWS Health Dashboard – Examples

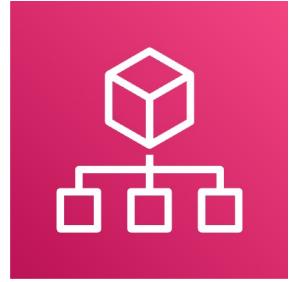
Remediating Exposed IAM Access Keys



Restarting Instances That are Scheduled for Retirement

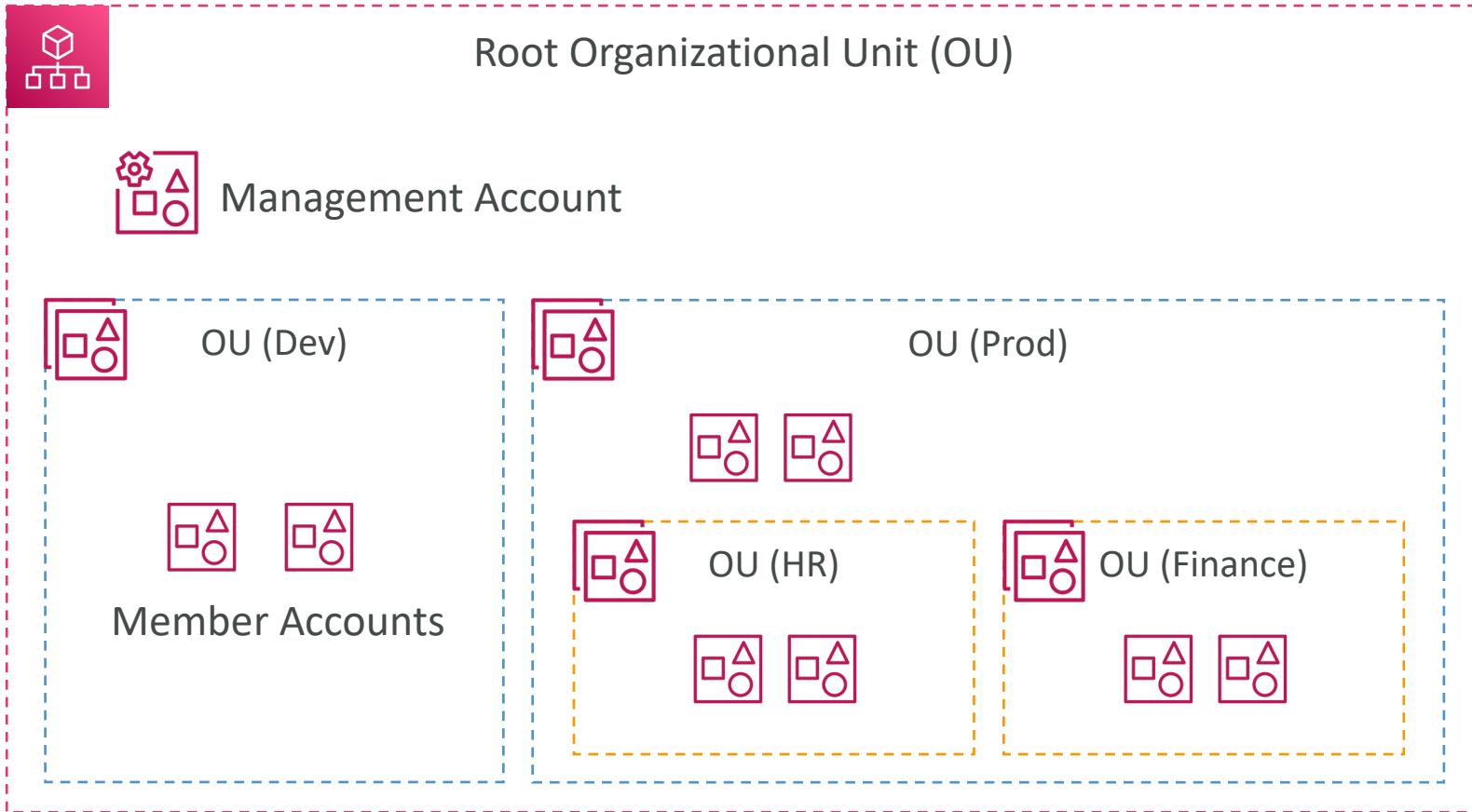


# AWS Organizations



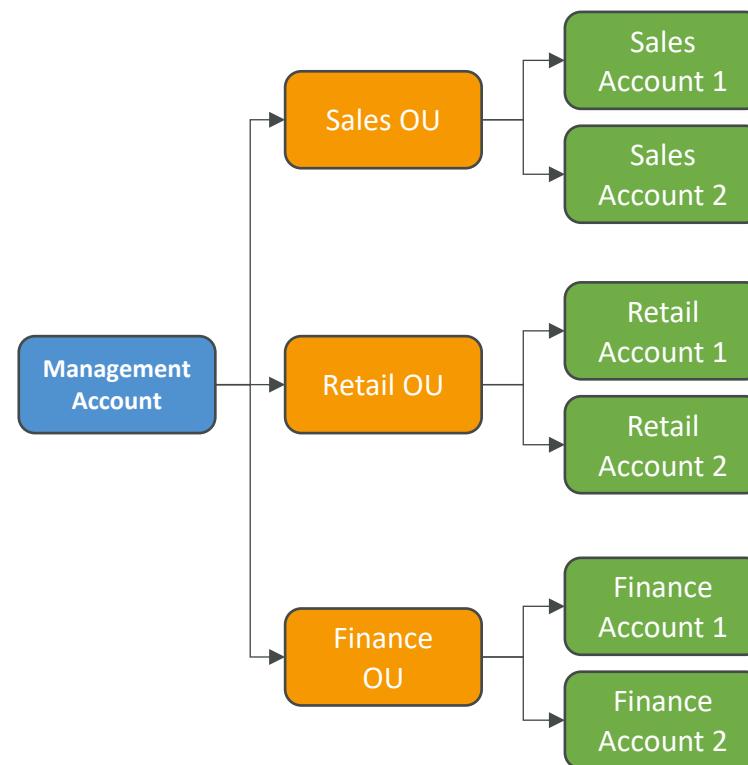
- Global service
- Allows to manage multiple AWS accounts
- The main account is the management account
- Other accounts are member accounts
- Member accounts can only be part of one organization
- Consolidated Billing across all accounts - single payment method
- Pricing benefits from aggregated usage (volume discount for EC2, S3...)
- Shared reserved instances and Savings Plans discounts across accounts
- API is available to automate AWS account creation

# AWS Organizations

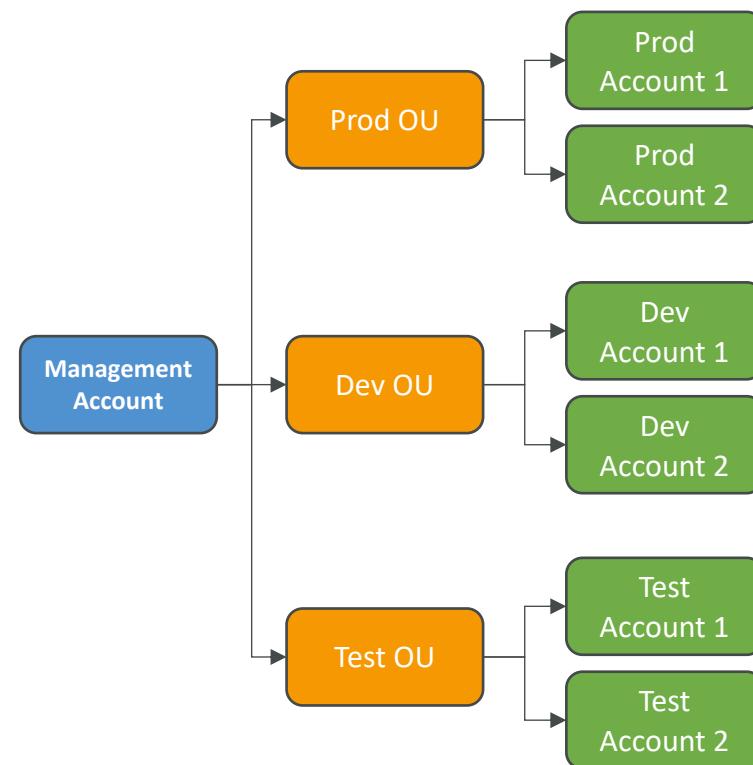


# Organizational Units (OU) - Examples

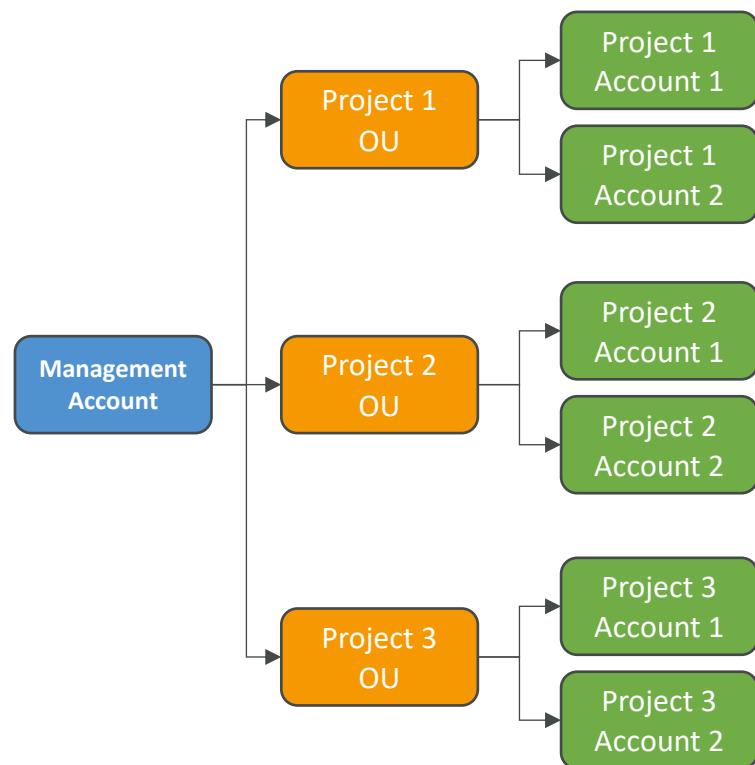
## Business Unit



## Environmental Lifecycle



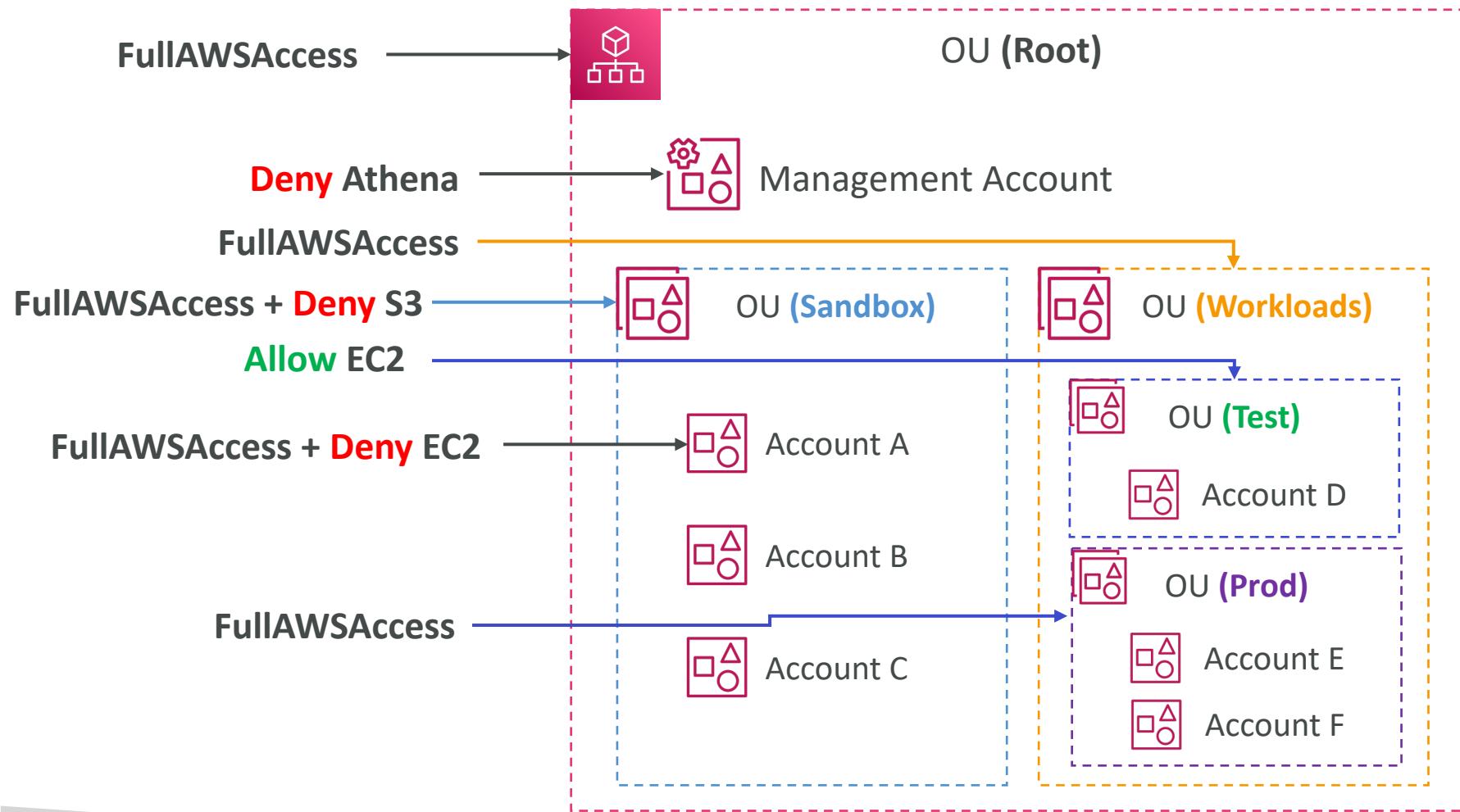
## Project-Based



# AWS Organizations

- Advantages
  - 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
  - Establish Cross Account Roles for Admin purposes
- Security: Service Control Policies (SCP)
  - IAM policies applied to OU or Accounts to restrict Users and Roles
  - They do not apply to the management account (full admin power)
  - Must have an explicit allow from the root through each OU in the direct path to the target account (does not allow anything by default – like IAM)

# SCP Hierarchy



- Management Account
  - Can do anything (no SCP apply)
- Account A
  - Can do anything
  - EXCEPT S3 (explicit Deny from Sandbox OU)
  - EXCEPT EC2 (explicit Deny)
- Account B & C
  - Can do anything
  - EXCEPT S3 (explicit Deny from Sandbox OU)
- Account D
  - Can access EC2
- Prod OU & Account E & F
  - Can do anything

# SCP Examples

## Blocklist and Allowlist 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": "*"  
    }  
  ]  
}
```

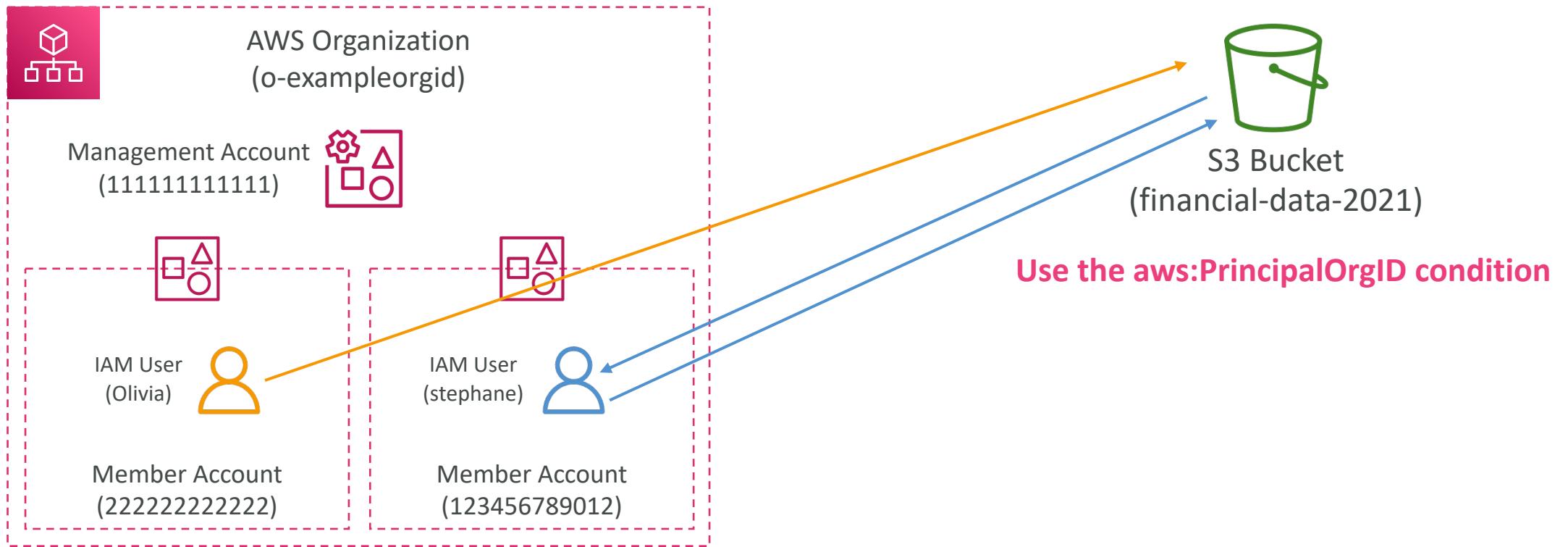
More examples: [https://docs.aws.amazon.com/organizations/latest/userguide/orgs\\_manage\\_policies\\_example-scps.html](https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_example-scps.html)

# AWS Organizations – Reserved Instances

- For billing purposes, the consolidated billing feature of AWS Organizations treats all the accounts in the organization as one account.
- This means that **all accounts** in the organization can receive the hourly cost benefit of Reserved Instances that are purchased by **any other account**.
- **The payer account (master account) of an organization** can turn off Reserved Instance (RI) discount and Savings Plans discount sharing for any accounts in that organization, including the payer account
- This means that RIs and Savings Plans discounts aren't shared between any accounts that have sharing turned off.
- To share an RI or Savings Plans discount with an account, both accounts must have sharing turned on.

# AWS Organizations – IAM Policies

- Use `aws:PrincipalOrgID` condition key in your resource-based policies to restrict access to IAM principals from accounts in an AWS Organization



# AWS Organizations – Tag Policies

- Helps you standardize tags across resources in an AWS Organization
- Ensure consistent tags, audit tagged resources, maintain proper resources categorization, ...
- You define tag keys and their allowed values
- Helps with AWS Cost Allocation Tags and Attribute-based Access Control
- Prevent any non-compliant tagging operations on specified services and resources (has no effect on resources without tags)
- Generate a report that lists all tagged/non-compliant resources
- Use CloudWatch Events to monitor non-compliant tags

```
{  
  "tags": {  
    "costcenter": {  
      "tag_key": {  
        "@@assign": "CostCenter"  
      },  
      "tag_value": {  
        "@@assign": ["100", "200"]  
      },  
      "enforced_for": {  
        "@@assign": ["secretsmanager:*"]  
      }  
    }  
  }  
}
```

# AWS Organizations – SCP Examples

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Deny",  
      "Action": "ec2:RunInstances",  
      "Resource": "arn:aws:ec2:*::instance/*",  
      "Condition": {  
        "Null": {  
          "aws:RequestTag/Department": "true"  
        }  
      }  
    }  
  ]  
}
```

Deny Run Instances  
If **Department** Tag is missing

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Deny",  
      "Action": "ec2:RunInstances",  
      "Resource": "arn:aws:ec2:*::instance/*",  
      "Condition": {  
        "StringNotLike": {  
          "aws:RequestTag/BusinessUnit": "infra-*"  
        }  
      }  
    }  
  ]  
}
```

Deny Run Instances  
If **BusinessUnit** Tag doesn't  
start with "infra-"

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Deny",  
      "Action": "ec2:RunInstances",  
      "Resource": "arn:aws:ec2:*::instance/*",  
      "Condition": {  
        "StringNotEquals": {  
          "aws:RequestTag/DeploymentType": ["In-region", "Edge"]  
        }  
      }  
    }  
  ]  
}
```

Deny Run Instances  
If **DeploymentType** Tag isn't  
set to either "In-region" or "Edge"

# AWS Organizations – SCP Examples

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Deny",  
        "Action": "*",  
        "Resource": "*",  
        "Condition": {  
            "StringNotEquals": {  
                "aws:RequestedRegion": ["eu-central-1", "eu-west-1"]  
            }  
        }  
    }  
}
```

Allow only actions in specific Regions  
“eu-central-1” and “eu-west-1”

# 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)

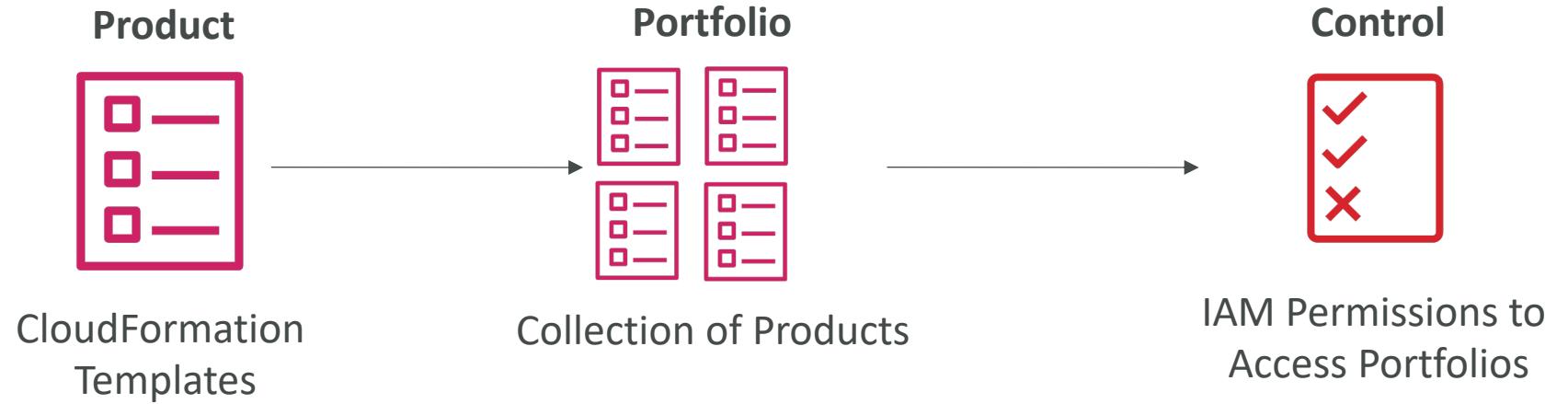
# AWS Service Catalog



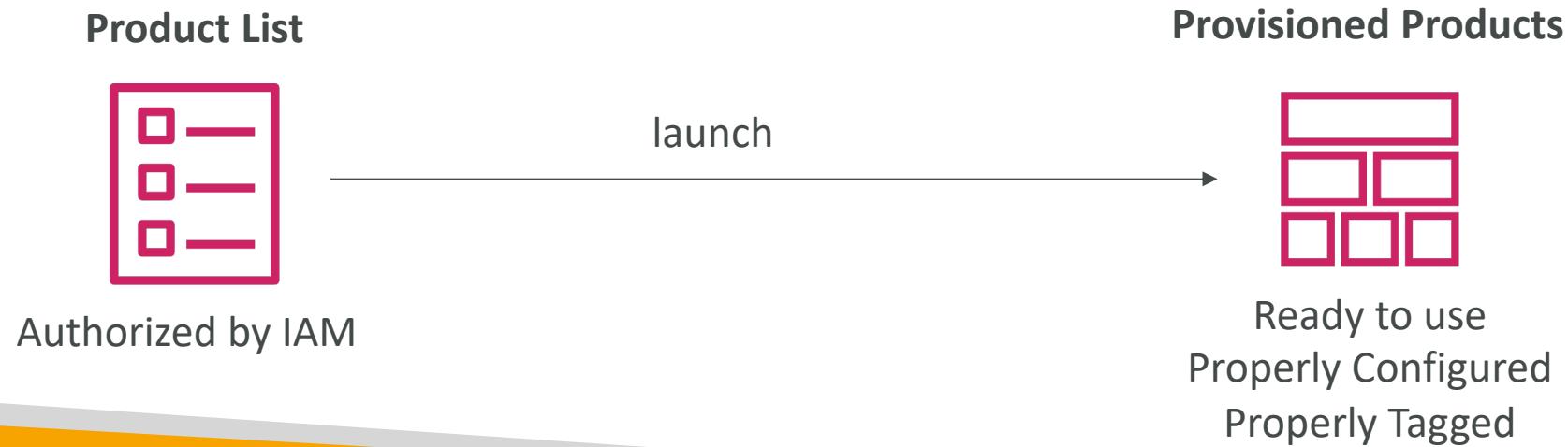
- Users that are new to AWS have too many options, and may create stacks that are not compliant / in line with the rest of the organization
- Some users just want a quick **self-service portal** to launch a set of authorized products pre-defined by admins
- Includes: virtual machines, databases, storage options, etc...
- Enter AWS Service Catalog!

# Service Catalog diagram

## ADMIN TASKS

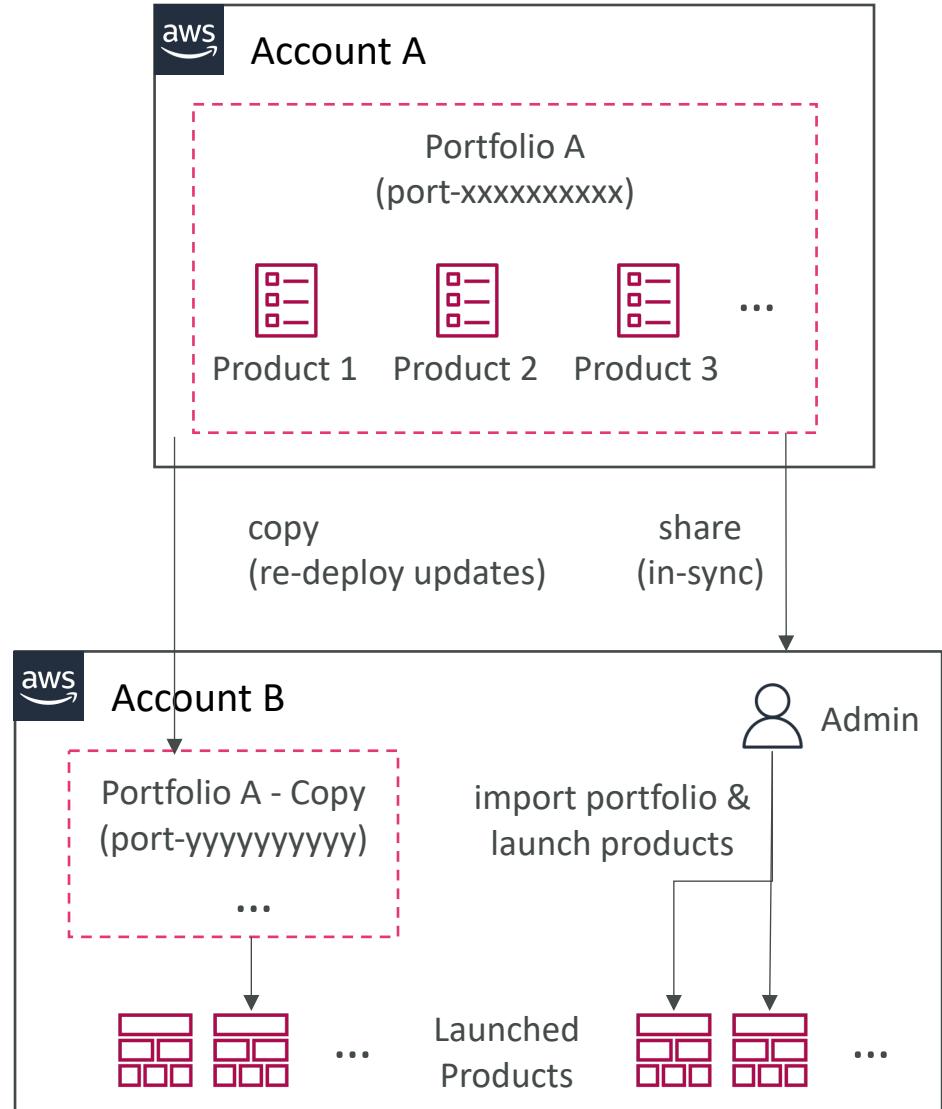


## USER TASKS



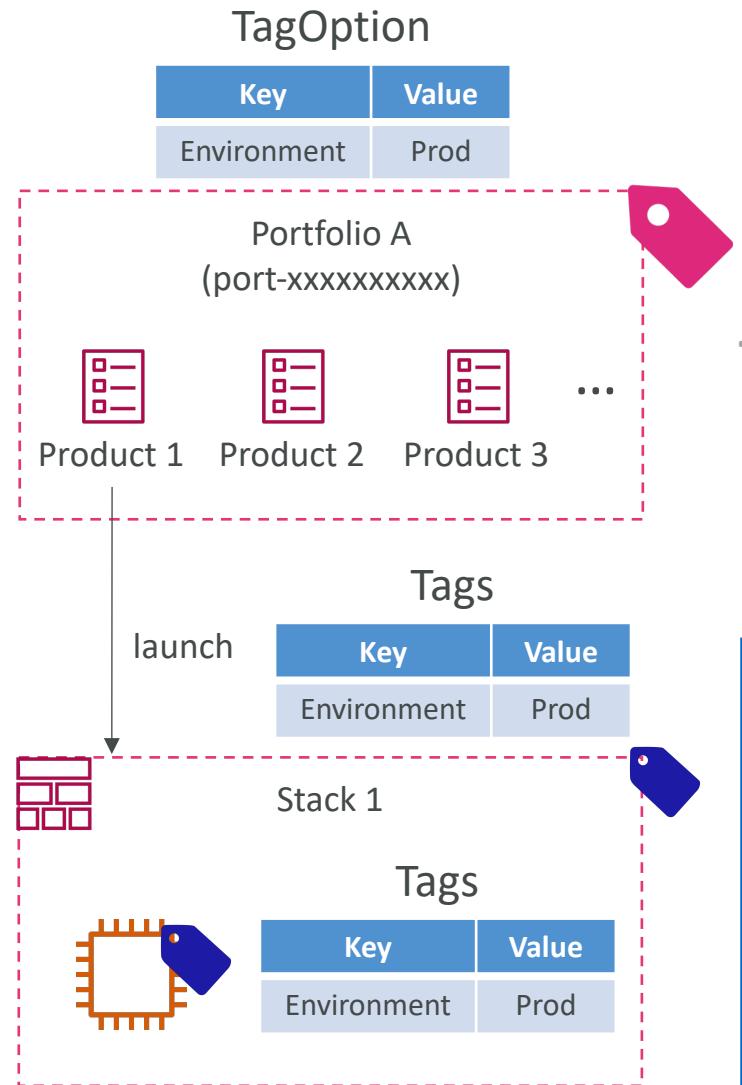
# AWS Service Catalog – Sharing Catalogs

- Share portfolios with individual AWS accounts or AWS Organizations
- Sharing options:
  - Share a reference of the portfolio, then import the shared portfolio in the recipient account (stays in-sync with the original portfolio)
  - Deploy a copy of the portfolio into the recipient account (must re-deploy any updates)
- Ability to add products from the imported portfolio to the local portfolio



# AWS Service Catalog – TagOptions Library

- Easily manage tags on provisioned products
- **TagOption:**
  - Key-value pair managed in AWS Service Catalog
  - Used to create an AWS Tag
- Can be associated with Portfolios and Products
- Use cases: proper resources tagging, defined allowed tags, ...
- Can be shared with other AWS accounts and AWS Organizations



# AWS Billing Alarms



- 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 project costs
  
- Let's create a billing alarm together!

# Cost Explorer

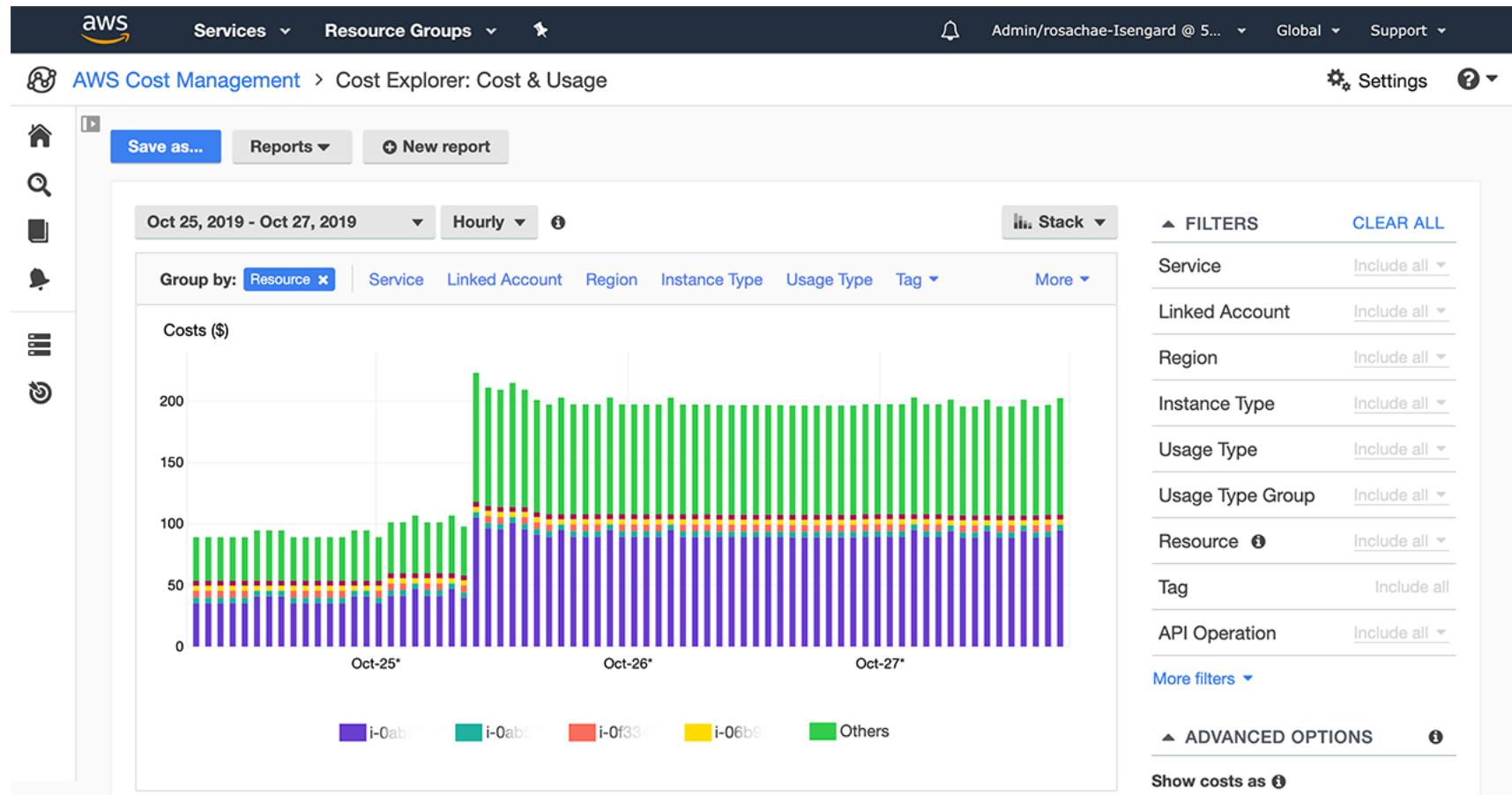


- Visualize, understand, and manage your AWS costs and usage over time
- Create custom reports that analyze cost and usage data.
- Analyze your data at a high level: total costs and usage across all accounts
- Or Monthly, hourly, resource level granularity
- Choose an optimal **Savings Plan** (to lower 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 [Download CSV](#) [Add selected Savings Plan\(s\) to cart](#)

x	Term	Payment option	Recommended commitment	Estimated hourly savings <small> ⓘ</small>
<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



# AWS Budgets



- Create budget and send alarms when costs exceeds the budget
- 4 types of budgets: Usage, Cost, Reservation, Savings Plans
- 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

# 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

# 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**
- Includes 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
  - in hourly or daily line items
  - any tags that you have activated for cost allocation purposes
- Can be configured for daily exports to S3
- Can be integrated with Athena, Redshift or QuickSight



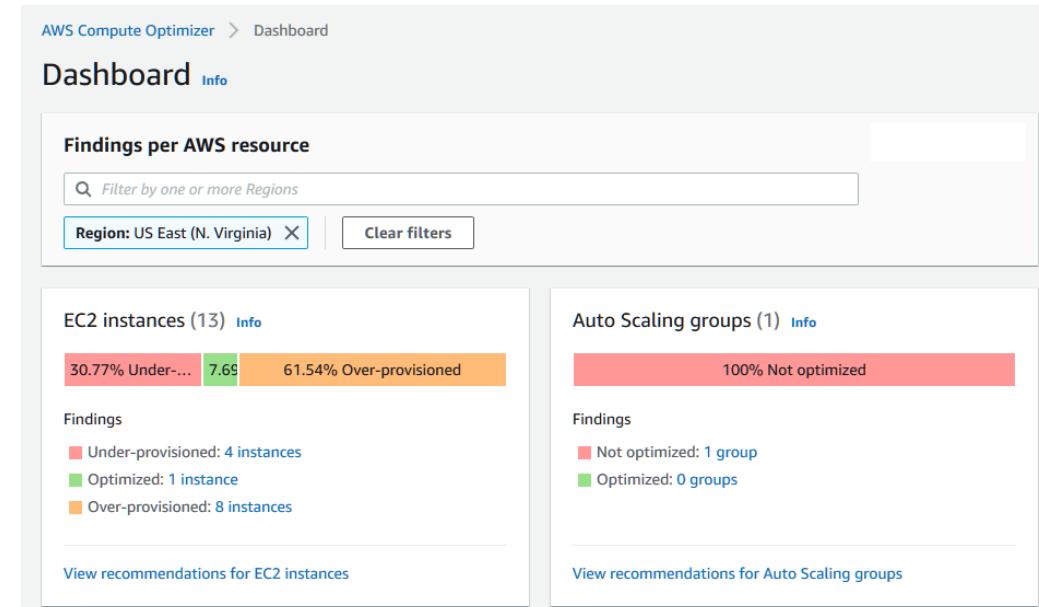
# Cost and Usage Reports

M	N	O	P	R	S	T
lineItem/ProductCode	lineItem/UsageType	lineItem/Operation	lineItem/AvailabilityZone	lineItem/UsageAmount	lineItem/CurrencyCode	lineItem/LineItemDescription
AmazonEC2	CW:AlarmMonitorUsage	Unknown		0.00134409	USD	\$0.00 per alarm-month - first 10 alarms
AmazonS3	Requests-Tier1	ListAllMyBuckets		2	USD	\$0.00 per request - PUT, COPY, POST, or LIST requests under the monthly global free tier
AmazonEC2	CW:AlarmMonitorUsage	Unknown		0.00134409	USD	\$0.00 per alarm-month - first 10 alarms
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
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
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
AmazonEC2	USW2-USE1-AWS-Out-Bytes	PublicIP-Out		0.00000174	USD	\$0.000 per GB - data transfer out under the monthly global free tier
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)
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)
AmazonS3	Requests-Tier1	ListAllMyBuckets		2	USD	\$0.00 per request - PUT, COPY, POST, or LIST requests under the monthly global free tier
AmazonEC2	USW2-DataTransfer-Out-Bytes	RunInstances		0.00038144	USD	\$0.00 per GB - data transfer out under the monthly global free tier
AmazonEC2	USW2-USW1-AWS-Out-Bytes	PublicIP-Out		0.00000174	USD	\$0.000 per GB - data transfer out under the monthly global free tier
AmazonEC2	USW2-DataTransfer-In-Bytes	RunInstances		0.00030951	USD	\$0.00 per GB - data transfer in per month
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
AmazonEC2	USW2-USW1-AWS-Out-Bytes	PublicIP-Out		0.00000349	USD	\$0.000 per GB - data transfer out under the monthly global free tier
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)
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
AmazonEC2	CW:AlarmMonitorUsage	Unknown		0.00134409	USD	\$0.00 per alarm-month - first 10 alarms
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
AmazonEC2	USW2-DataTransfer-Regional-Bytes	PublicIP-Out		0.00000349	USD	\$0.000 per GB - regional data transfer under the monthly global free tier
AmazonEC2	USW2-DataTransfer-In-Bytes	RunInstances		0.00032071	USD	\$0.000 per GB - data transfer in per month
AmazonEC2	USW2-DataTransfer-Regional-Bytes	PublicIP-In		0.00000302	USD	\$0.000 per GB - regional data transfer under the monthly global free tier
AmazonEC2	USW2-USE1-AWS-Out-Bytes	PublicIP-Out		0.00000174	USD	\$0.000 per GB - data transfer out under the monthly global free tier
AmazonEC2	USW2-DataTransfer-Out-Bytes	RunInstances		0.00045736	USD	\$0.000 per GB - data transfer out under the monthly global free tier
AmazonEC2	USW2-DataTransfer-In-Bytes	RunInstances		0.00036737	USD	\$0.000 per GB - data transfer in per month
AmazonEC2	USW2-APN2-AWS-In-Bytes	PublicIP-In		0.00000005	USD	\$0.00 per GB - US West (Oregon) data transfer from Asia Pacific (Seoul)
AmazonEC2	USW2-APN2-AWS-Out-Bytes	PublicIP-Out		0.00000018	USD	\$0.000 per GB - data transfer out under the monthly global free tier
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)
AmazonEC2	USW2-DataTransfer-Out-Bytes	RunInstances		0.00039945	USD	\$0.000 per GB - data transfer out under the monthly global free tier
AmazonEC2	CW:AlarmMonitorUsage	Unknown		0.00134409	USD	\$0.00 per alarm-month - first 10 alarms

# 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
  - ECS on Fargate, Aurora & RDS databases
  - Commercial software licenses
- Lower your costs by up to 25%
- Recommendations can be exported to S3



# AWS Compute Optimizer for CloudOps

- To view AWS Compute Optimizer, users need access to the IAM AWS-managed policy **ComputeOptimizerReadOnlyAccess**
- Issue: your EC2 instance does NOT appear in Compute Optimizer
  - Your EC2 instance is new and has insufficient metrics data
  - **Solution:** keep the instance running for at least 30 hours to allow Compute Optimizer to collect enough metrics data and generate recommendations

# AWS Billing Conductor



- Service for customizing and presenting AWS billing data
- Group accounts, apply markups/discounts, and generate pro forma bills
- Does not affect your actual AWS bill — only how it's displayed and allocated
- Common Use Cases
  - Enterprises: split bill by department or cost center
  - MSPs: apply custom pricing per customer
  - Finance teams: build chargeback/showback models
  - Subsidiaries: create transparent internal invoices
- Helpful for big organizations with big accounting needs

# Disaster Recovery

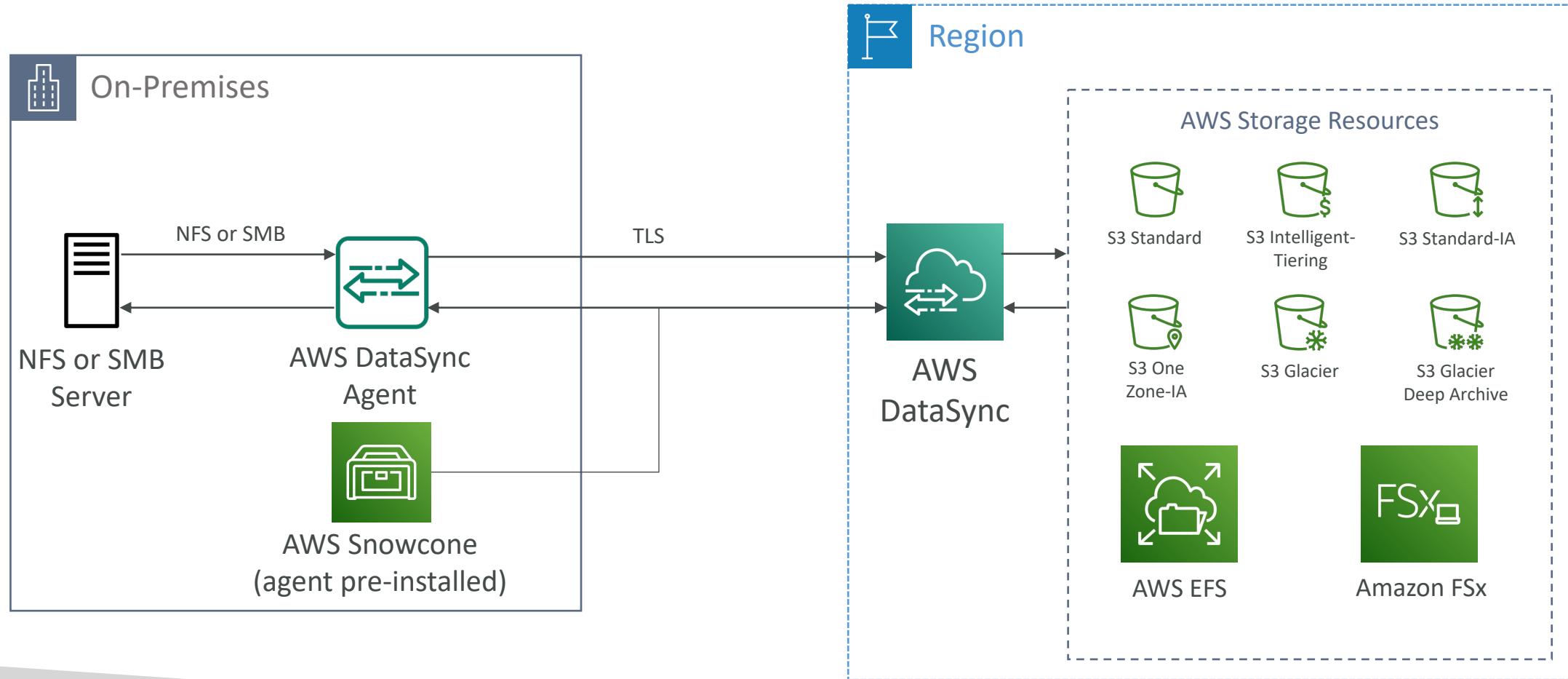


# AWS DataSync

- Move large amount of data to and from
  - On-premises / other cloud to AWS (NFS, SMB, HDFS, S3 API...) – needs agent
  - AWS to AWS (different storage services) – no agent needed
- Can synchronize to:
  - Amazon S3 (any storage classes – including Glacier)
  - Amazon EFS
  - Amazon FSx (Windows, Lustre, NetApp, OpenZFS...)
- Replication tasks can be scheduled hourly, daily, weekly
- File permissions and metadata are preserved (NFS POSIX, SMB...)
- One agent task can use 10 Gbps, can setup a bandwidth limit

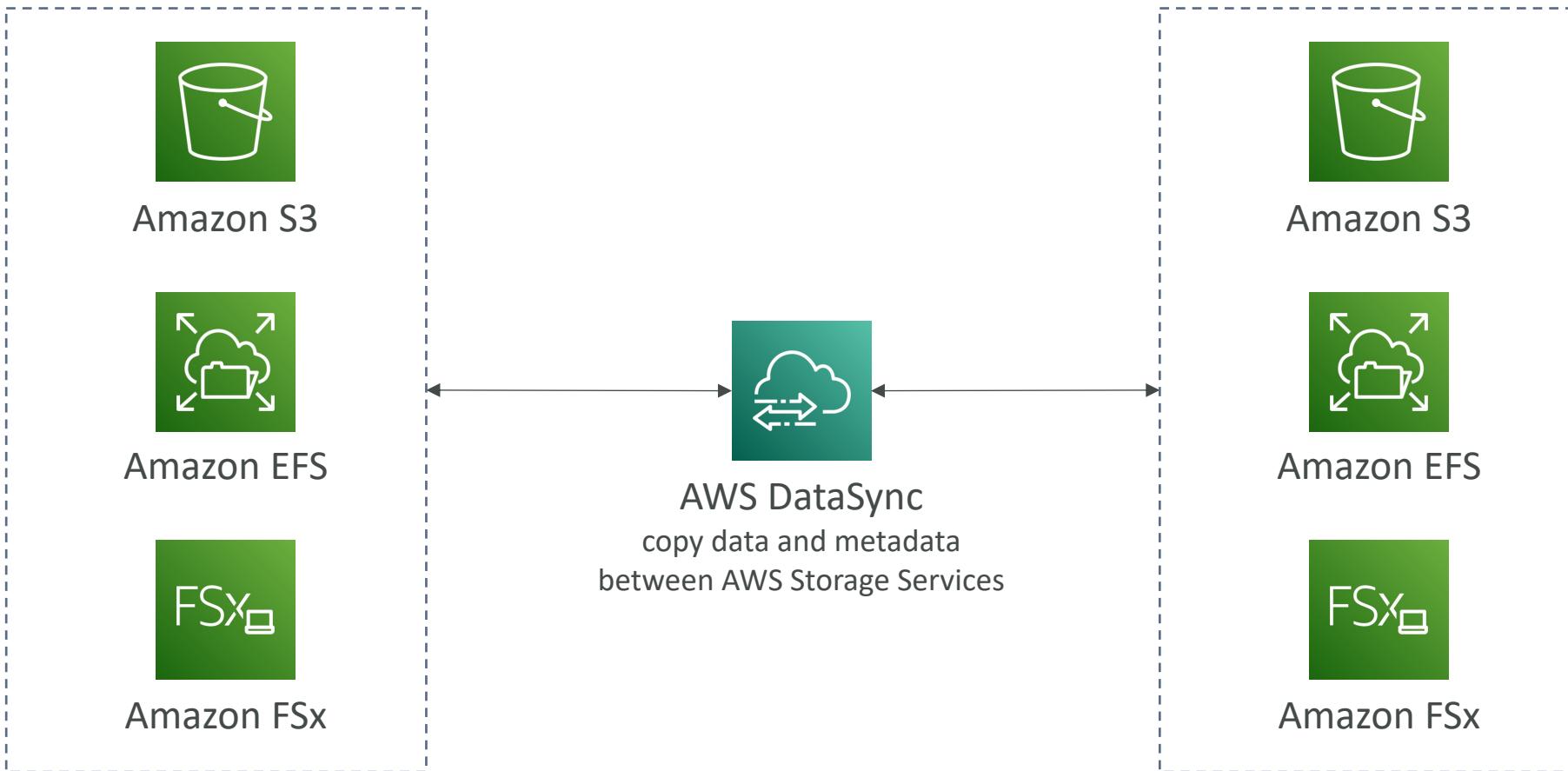
# AWS DataSync

## NFS / SMB to AWS (S3, EFS, FSx...)



# AWS DataSync

## Transfer between AWS storage services





# AWS Backup

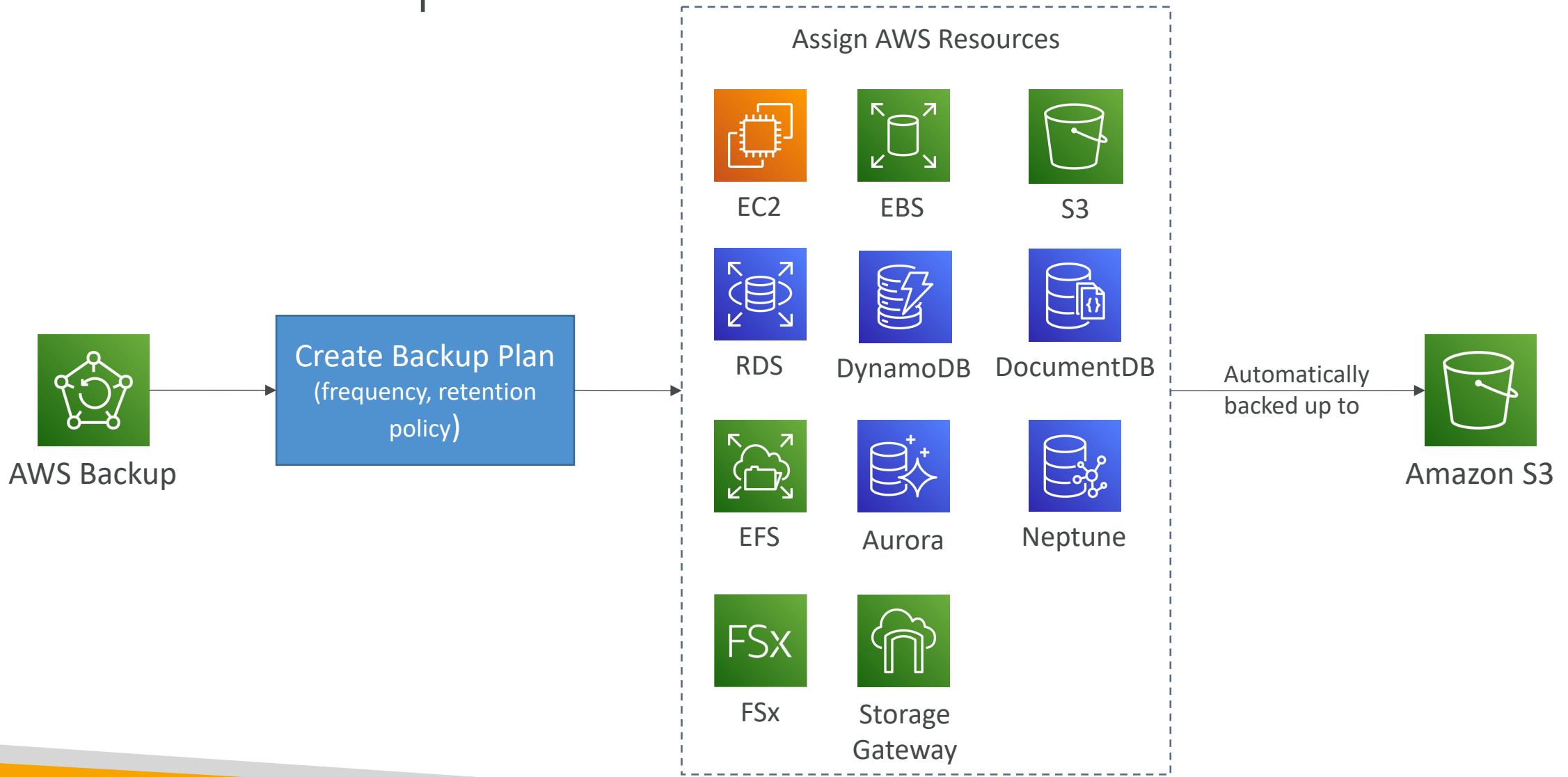
- Fully managed service
- Centrally manage and automate backups across AWS services
- No need to create custom scripts and manual processes
- Supported services:
  - Amazon EC2 / Amazon EBS
  - Amazon S3
  - Amazon RDS (all DBs engines) / Amazon Aurora / Amazon DynamoDB
  - Amazon DocumentDB / Amazon Neptune
  - Amazon EFS / Amazon FSx (Lustre & Windows File Server)
  - AWS Storage Gateway (Volume Gateway)
- Supports cross-region backups
- Supports cross-account backups

# AWS Backup



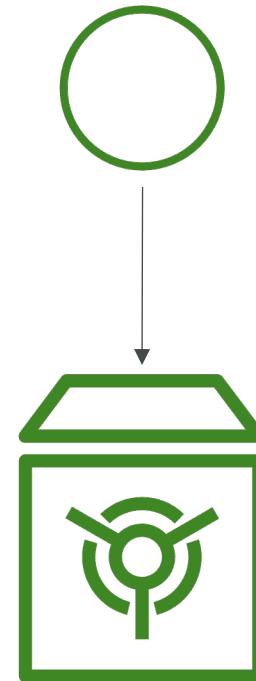
- Supports PITR for supported services
- On-Demand and Scheduled backups
- Tag-based backup policies
- You create backup policies known as **Backup Plans**
  - Backup frequency (every 12 hours, daily, weekly, monthly, cron expression)
  - Backup window
  - Transition to Cold Storage (Never, Days, Weeks, Months, Years)
  - Retention Period (Always, Days, Weeks, Months, Years)

# AWS Backup



# AWS Backup Vault Lock

- Enforce a WORM (Write Once Read Many) state for all the backups that you store in your AWS Backup Vault
- Additional layer of defense to protect your backups against:
  - Inadvertent or malicious delete operations
  - Updates that shorten or alter retention periods
- Even the root user cannot delete backups when enabled



backup

**Backup Vault Lock Policy**  
Backups can't be deleted

# Security & Compliance

# AWS WAF – Web Application Firewall



- Protects your web applications from common web exploits (Layer 7)
- Layer 7 is HTTP (vs Layer 4 is TCP/UDP)
- Deploy on
  - Application Load Balancer
  - API Gateway
  - CloudFront
  - AppSync GraphQL API
  - Cognito User Pool

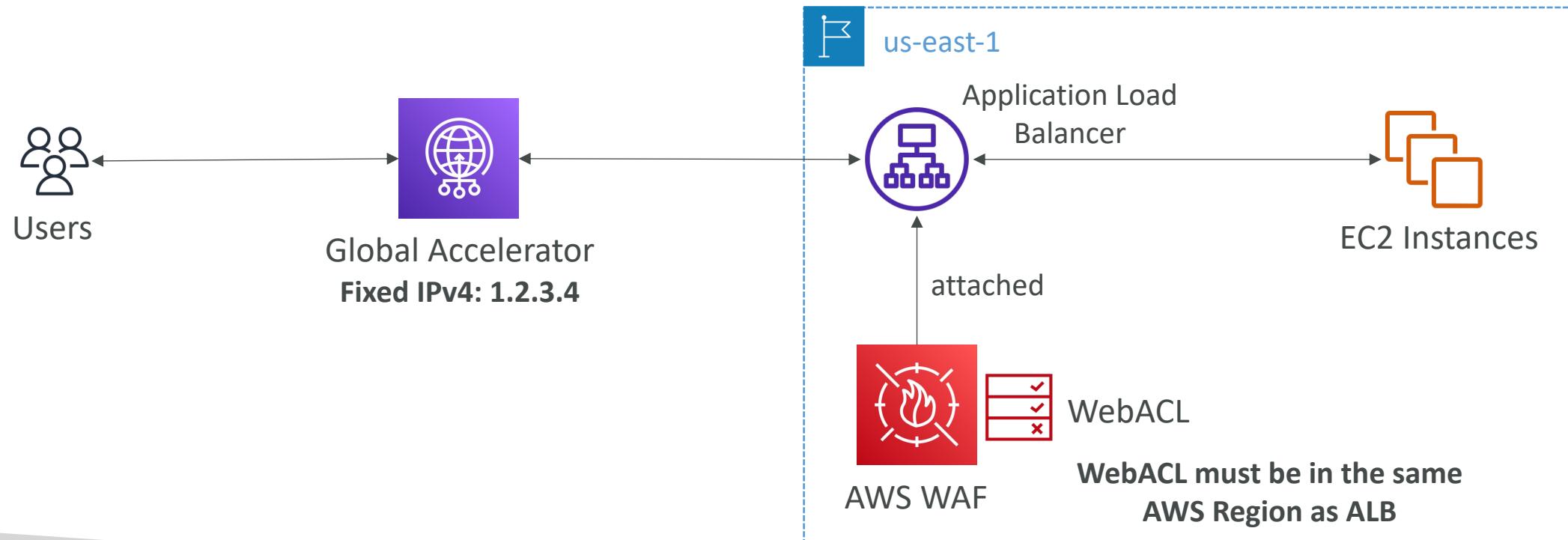
# AWS WAF – Web Application Firewall



- Define Web ACL (Web Access Control List) Rules:
  - IP Set: up to 10,000 IP addresses – use multiple Rules for more IPs
  - 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
- Web ACL are Regional except for CloudFront
- A rule group is a reusable set of rules that you can add to a web ACL

# WAF – Fixed IP while using WAF with a Load Balancer

- WAF does not support the Network Load Balancer (Layer 4)
- We can use Global Accelerator for fixed IP and WAF on the ALB



# AWS Shield: protect from DDoS attack



- DDoS: Distributed Denial of Service – many requests at the same time
- 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
  - Shield Advanced automatic application layer DDoS mitigation automatically creates, evaluates and deploys AWS WAF rules to mitigate layer 7 attacks

# AWS Firewall Manager



- Manage rules in all accounts of an AWS Organization
- Security policy: common set of security rules
  - WAF rules (Application Load Balancer, API Gateways, CloudFront)
  - AWS Shield Advanced (ALB, CLB, NLB, Elastic IP, CloudFront)
  - Security Groups for EC2, Application Load Balancer and ENI resources in VPC
  - AWS Network Firewall (VPC Level)
  - Amazon Route 53 Resolver DNS Firewall
  - Policies are created at the region level
- Rules are applied to new resources as they are created (good for compliance) across all and future accounts in your Organization

# WAF vs. Firewall Manager vs. Shield



AWS WAF



AWS Firewall Manager

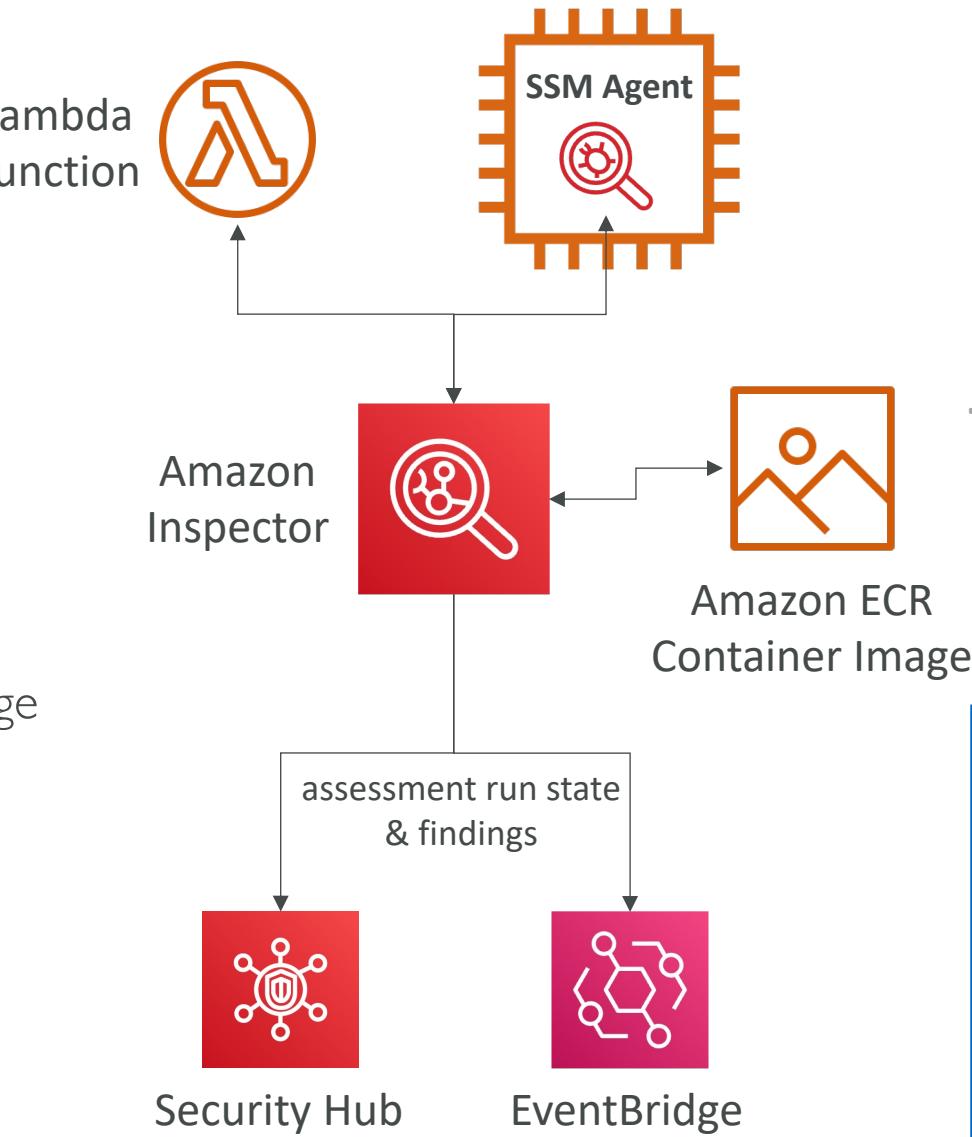


AWS Shield

- WAF, Shield and Firewall Manager are used together for comprehensive protection
- Define your Web ACL rules in WAF
- For granular protection of your resources, WAF alone is the correct choice
- If you want to use AWS WAF across accounts, accelerate WAF configuration, automate the protection of new resources, use Firewall Manager with AWS WAF
- Shield Advanced adds additional features on top of AWS WAF, such as dedicated support from the Shield Response Team (SRT) and advanced reporting.
- If you're prone to frequent DDoS attacks, consider purchasing Shield Advanced

# Amazon Inspector

- Automated Security Assessments
- For EC2 instances
  - Leveraging the AWS System Manager (SSM) agent
  - Analyze against unintended network accessibility
  - Analyze the running OS against known vulnerabilities
- For Container Images push to Amazon ECR
  - Assessment of Container Images as they are pushed
- For Lambda Functions
  - Identifies software vulnerabilities in function code and package dependencies
  - Assessment of functions as they are deployed
- Reporting & integration with AWS Security Hub
- Send findings to Amazon Event Bridge



# What does Amazon Inspector evaluate?



- Remember: only for EC2 instances, Container Images & Lambda functions
- Continuous scanning of the infrastructure, only when needed
- Package vulnerabilities (EC2, ECR & Lambda) – database of CVE
- Network reachability (EC2)
- A risk score is associated with all vulnerabilities for prioritization

# Logging in AWS for security and compliance

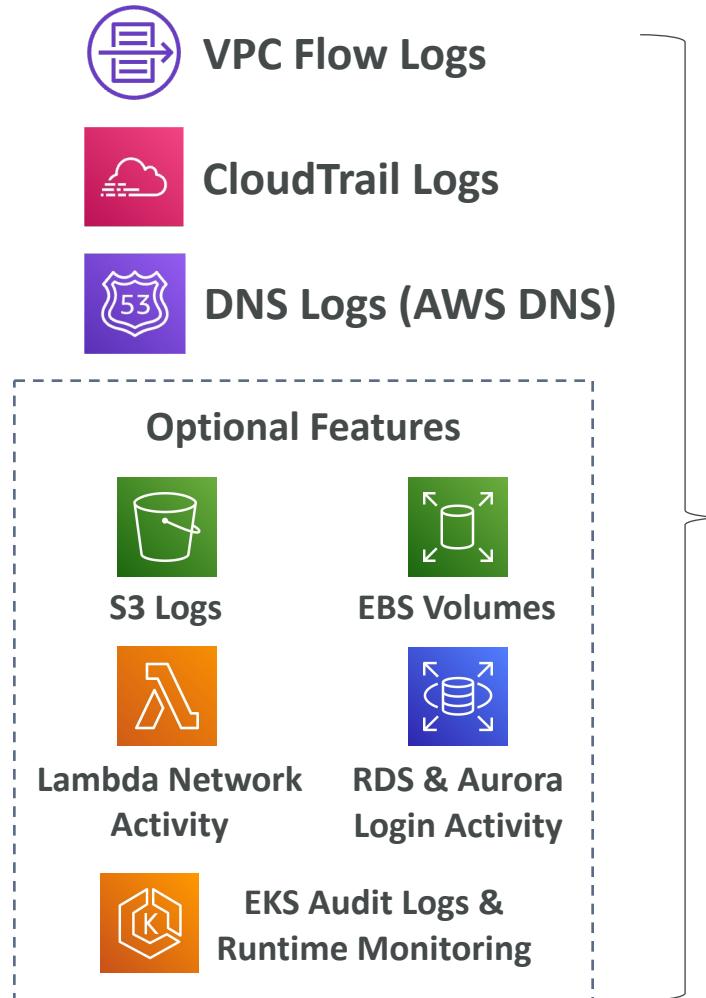
- To help compliance requirements, AWS provides many service-specific security and audit logs
- Service Logs include:
  - CloudTrail trails - trace all API calls
  - Config Rules - for config & compliance over time
  - CloudWatch Logs - for full data retention
  - VPC Flow Logs - IP traffic within your VPC
  - ELB Access Logs - metadata of requests made to your load balancers
  - CloudFront Logs - web distribution access logs
  - WAF Logs - full logging of all requests analyzed by the service
- Logs can be analyzed using AWS Athena if they're stored in S3
- You should encrypt logs in S3, control access using IAM & Bucket Policies, MFA
- Move Logs to Glacier for cost savings
- Read whitepaper if interested at:  
[https://d0.awsstatic.com/whitepapers/compliance/AWS\\_Security\\_at\\_Scale\\_Logging\\_in\\_AWS\\_Whitepaper.pdf](https://d0.awsstatic.com/whitepapers/compliance/AWS_Security_at_Scale_Logging_in_AWS_Whitepaper.pdf)



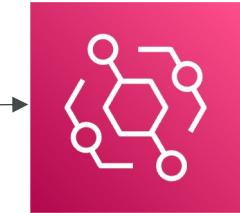
# Amazon GuardDuty

- Intelligent Threat discovery to protect your 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 Events Logs – unusual API calls, unauthorized deployments
    - CloudTrail Management Events – create VPC subnet, create trail, ...
    - CloudTrail S3 Data Events – get object, list objects, delete object, ...
  - VPC Flow Logs – unusual internal traffic, unusual IP address
  - DNS Logs – compromised EC2 instances sending encoded data within DNS queries
  - Optional Feature – EKS Audit Logs, RDS & Aurora, EBS, Lambda, S3 Data Events...
- Can setup **EventBridge rules** to be notified in case of findings
- EventBridge rules can target AWS Lambda or SNS
- Can protect against CryptoCurrency attacks (has a dedicated “finding” for it)

# Amazon GuardDuty



GuardDuty



EventBridge



SNS

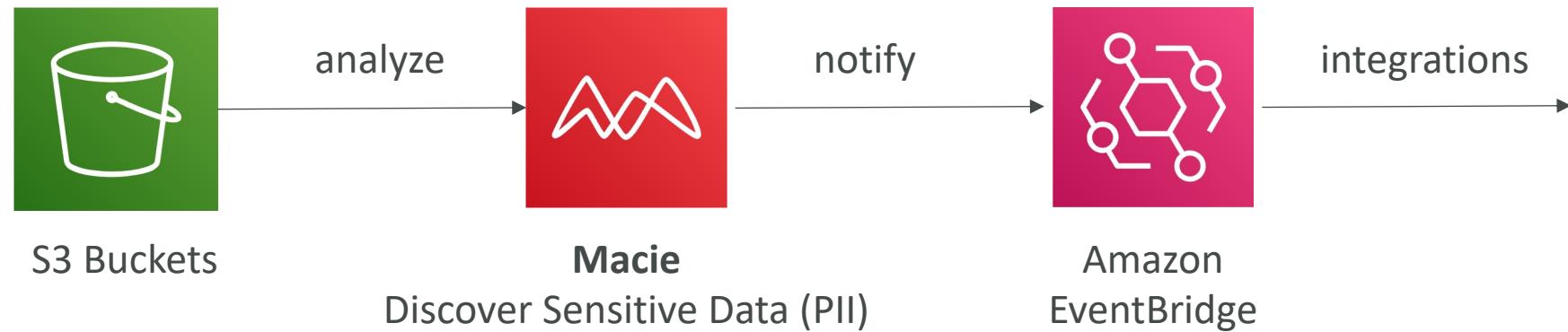


Lambda

# AWS Macie



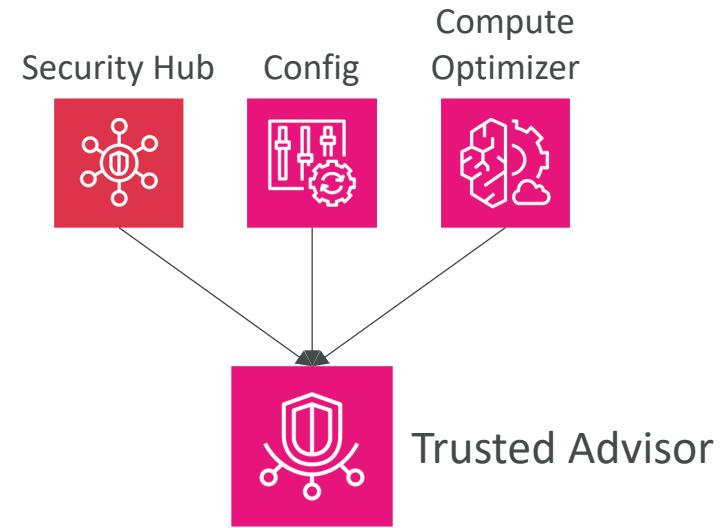
- Amazon Macie is a fully managed data security and data privacy service that uses machine learning and pattern matching to discover and protect your sensitive data in AWS.
- Macie helps identify and alert you to sensitive data, such as personally identifiable information (PII)



# Trusted Advisor

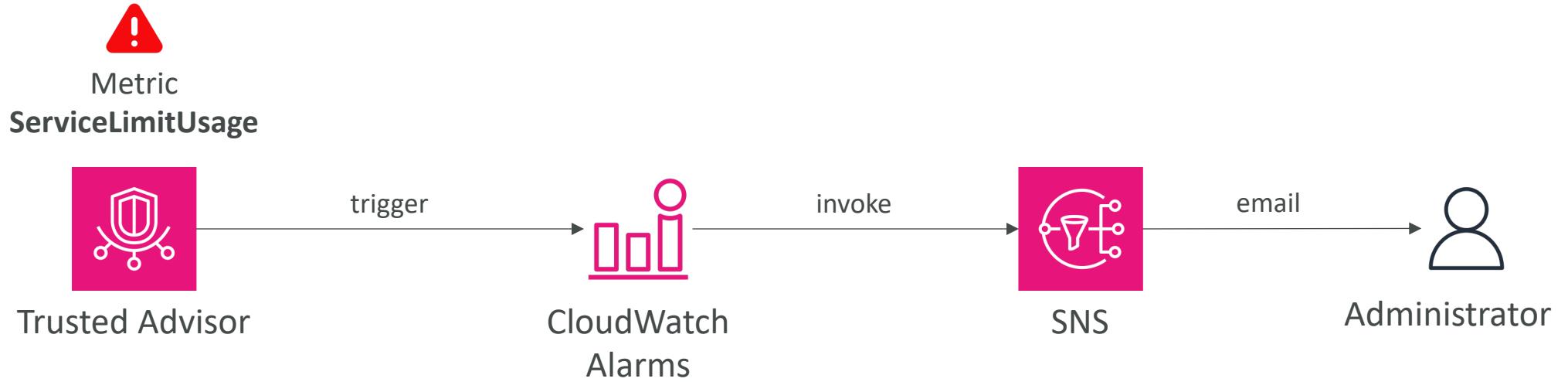


- No need to install anything – high level AWS account assessment
- Analyze your AWS accounts and provides recommendation on 6 categories:
  - Cost optimization
  - Performance
  - Security
  - Fault tolerance
  - Service limits
  - Operational Excellence
- **Business & Enterprise Support plan**
  - Full Set of Checks
  - Programmatic Access using AWS Support API
- Integrated with Security Hub, Config, Compute Optimizer



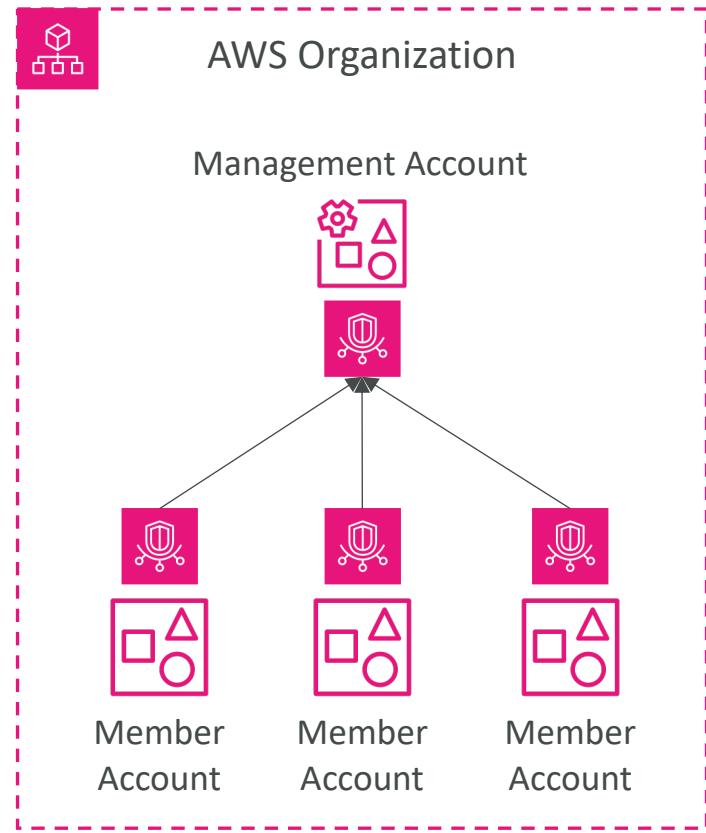
Checks	
▶	<span style="color: green;">✔</span> <b>Amazon EBS Public Snapshots</b> Checks the permission settings for your Amazon Elastic Block Store snapshots. 0 EBS snapshots are marked as public.
▶	<span style="color: green;">✔</span> <b>Amazon RDS Public Snapshots</b> Checks the permission settings for your Amazon Relational Database Service snapshots. 0 RDS snapshots are marked as public.
▶	<span style="color: green;">✔</span> <b>IAM Use</b> This check is intended to discourage the use of root access keys. At least one IAM user has been created for this account.

# Trusted Advisor – Automate Notifications



# Trusted Advisor – Organizational View

- You can view all Trusted Advisor checks across all accounts in an AWS Organization
- Organization must have all Features enabled and have Business, Enterprise On-Ramp, or Enterprise Support Plan
- In Trusted Advisor, enable **Trusted Access with AWS Organization** in the Management account (or Delegated Administrator account)

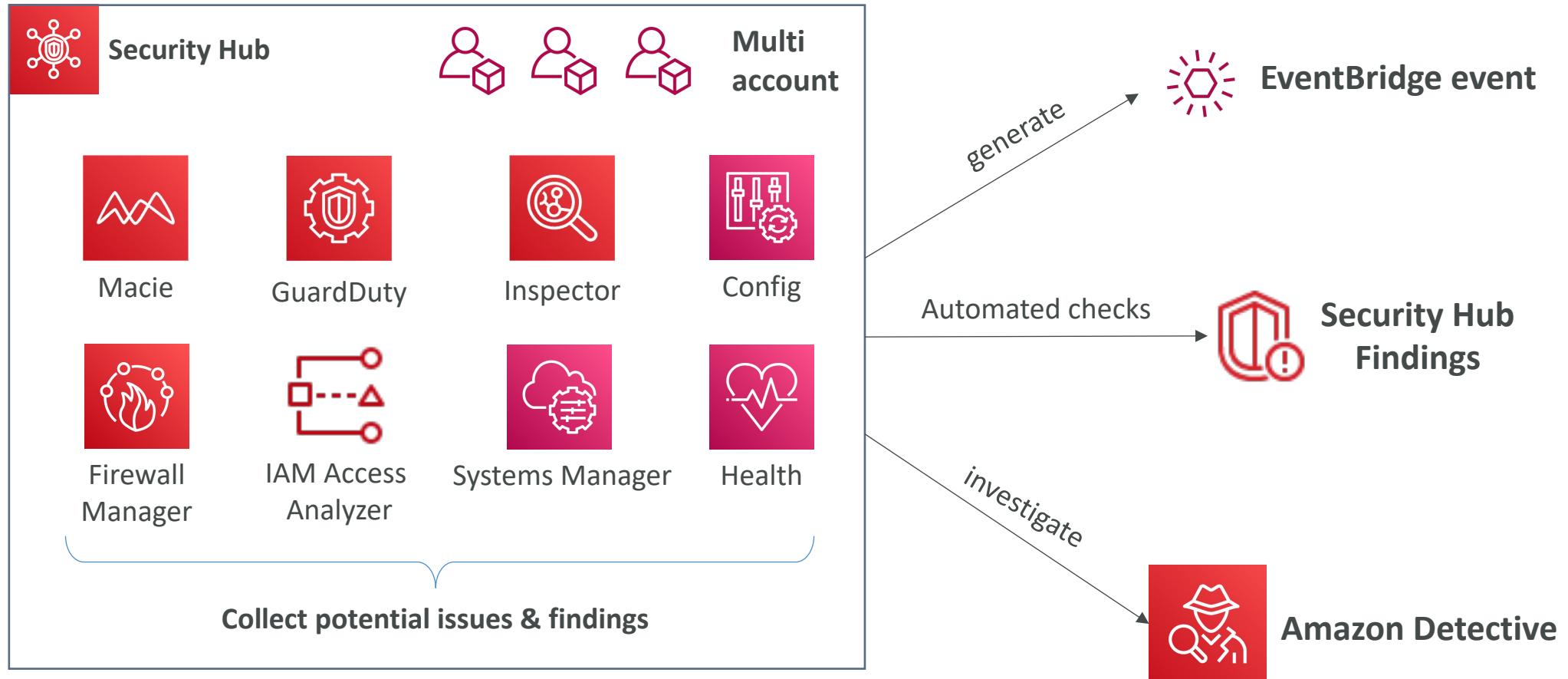


# AWS Security Hub



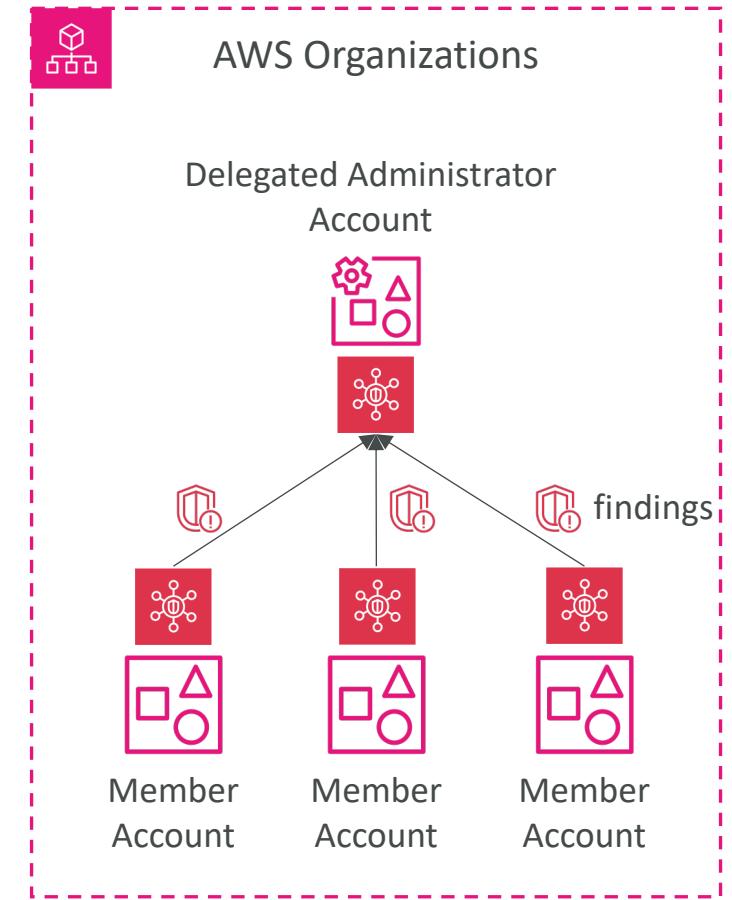
- Central security tool to manage security across several AWS accounts and automate security checks
- Integrated dashboards showing current security and compliance status to quickly take actions
- Automatically aggregates alerts in predefined or personal findings formats from various AWS services & AWS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

# AWS Security Hub



# Security Hub – Managing Multiple Accounts

- Integrate Security Hub with AWS Organizations
- You can manage Security Hub in all your member accounts from an Administrator Account
- You designate a Security Hub Delegated Administrator & Security Hub is automatically enabled
- Manage Security Hub with centralized config
- Example: run CIS AWS Benchmark to scan all accounts and monitor from a central Security Hub

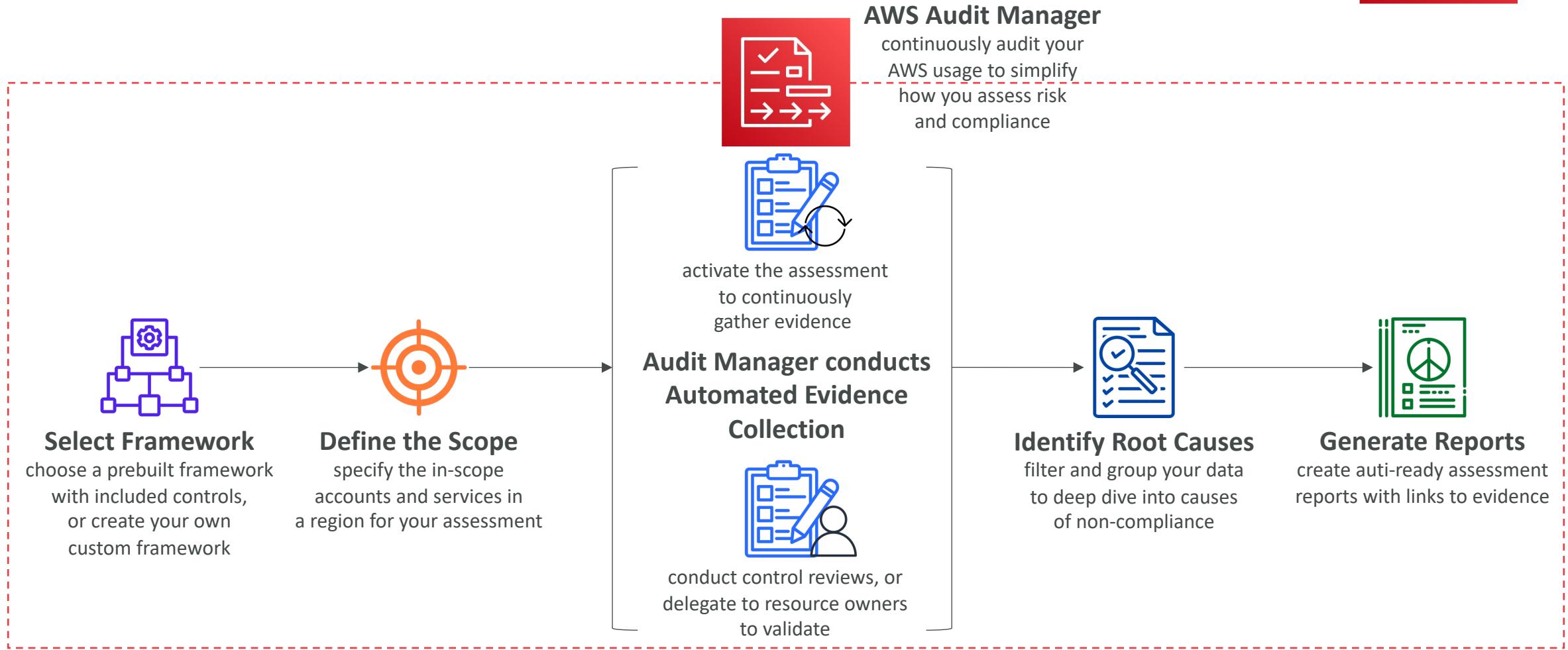




# AWS Audit Manager

- Assess risk and compliance of your AWS workloads
- Continuously audit AWS services usage and prepare audits
- Prebuilt frameworks include:
  - CIS AWS Foundations Benchmark 1.2.0 & 1.3.0
  - General Data Protection Regulation (GDPR),
  - Health Insurance Portability and Accountability Act (HIPAA)
  - Payment Card Industry Data Security Standard (PCI DSS) v3.2.1
  - Service Organization Control 2 (SOC 2)
- Generates reports of compliance alongside evidence folders

# AWS Audit Manager



# AWS KMS (Key Management Service)



- Anytime you hear “encryption” for an AWS service, it’s most likely KMS
- AWS manages encryption keys for us
- Fully integrated with IAM for authorization
- Easy way to control access to your data
- Able to audit KMS Key usage using CloudTrail
- Seamlessly integrated into most AWS services (EBS, S3, RDS, SSM...)
- **Never ever store your secrets in plaintext, especially in your code!**
  - KMS Key Encryption also available through API calls (SDK, CLI)
  - Encrypted secrets can be stored in the code / environment variables

# KMS Keys Types

- KMS Keys is the new name of KMS Customer Master Key
- Symmetric (AES-256 keys)
  - Single encryption key that is used to Encrypt and Decrypt
  - AWS services that are integrated with KMS use Symmetric CMKs
  - You never get access to the KMS Key unencrypted (must call KMS API to use)
- Asymmetric (RSA & ECC key pairs)
  - Public (Encrypt) and Private Key (Decrypt) pair
  - Used for Encrypt/Decrypt, or Sign/Verify operations
  - The public key is downloadable, but you can't access the Private Key unencrypted
  - Use case: encryption outside of AWS by users who can't call the KMS API

# AWS KMS (Key Management Service)



- Types of KMS Keys:

- AWS Owned Keys (free): SSE-S3, SSE-SQS, SSE-DDB (default key)
- AWS Managed Key: **free** (aws/service-name, example: aws/rds or aws/ebs)
- Customer managed keys created in KMS: \$1 / month
- Customer managed keys imported: \$1 / month
- + pay for API call to KMS (\$0.03 / 10000 calls)

## Encryption key management

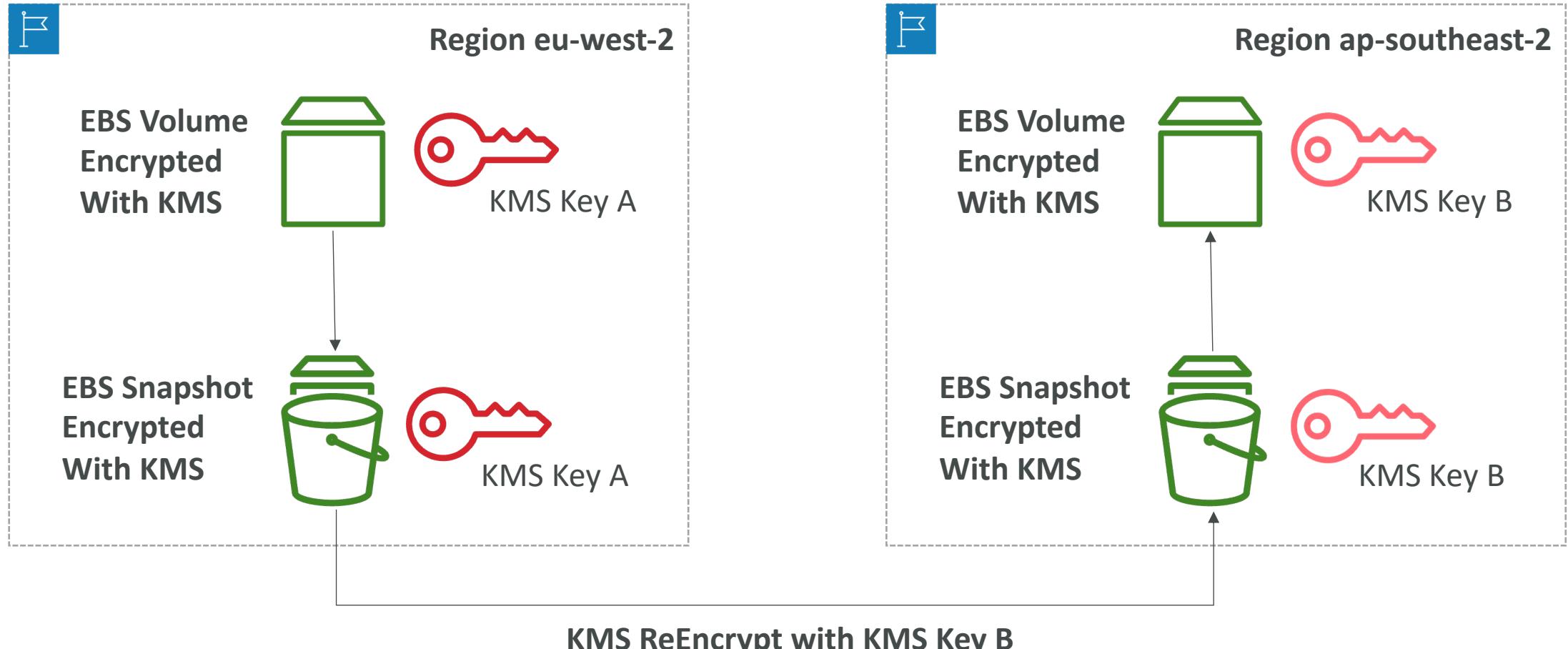
- Owned by Amazon DynamoDB
- AWS managed key **Lea**  
Key alias: aws/dynamodb.
- Stored in your account,  
and owned and managed by you



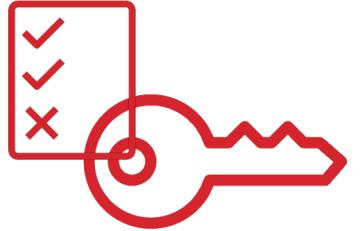
- Automatic Key rotation:

- AWS-managed KMS Key: automatic every 1 year
- Customer-managed KMS Key: (must be enabled) automatic & on-demand
- Imported KMS Key: only manual rotation possible using alias

# Copying Snapshots across regions



# KMS Key Policies



- Control access to KMS keys, “similar” to S3 bucket policies
- Difference: you cannot control access without them
- **Default KMS Key Policy:**
  - Created if you don't provide a specific KMS Key Policy
  - Complete access to the key to the root user = entire AWS account
- **Custom KMS Key Policy:**
  - Define users, roles that can access the KMS key
  - Define who can administer the key
  - Useful for cross-account access of your KMS key

# Copying Snapshots across accounts

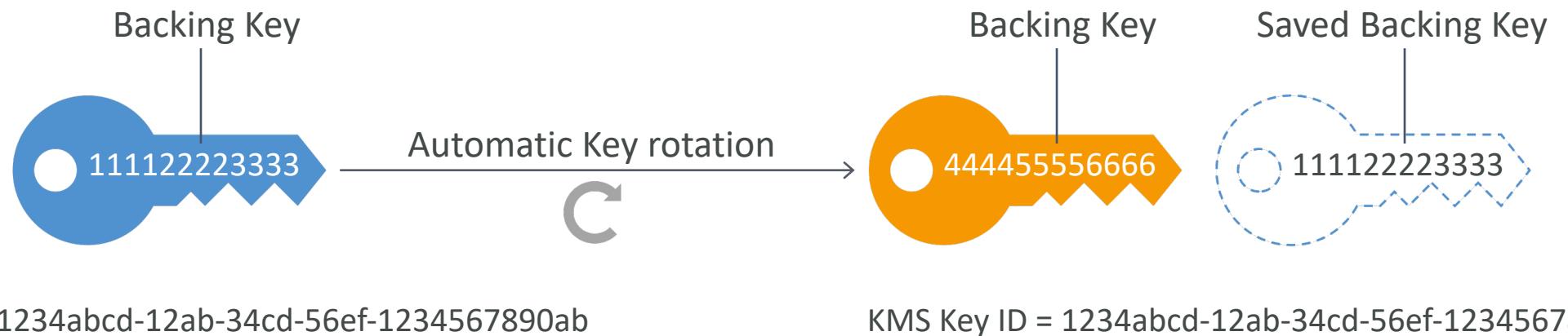
1. Create a Snapshot, encrypted with your own KMS Key (Customer Managed Key)
2. Attach a KMS Key Policy to authorize cross-account access
3. Share the encrypted snapshot
4. (in target) Create a copy of the Snapshot, encrypt it with a CMK in your account
5. Create a volume from the snapshot

```
{  
  "Sid": "Allow use of the key with destination account",  
  "Effect": "Allow",  
  "Principal": {  
    "AWS": "arn:aws:iam::TARGET-ACCOUNT-ID:role/ROLENAMESPACE"  
  },  
  "Action": [  
    "kms:Decrypt",  
    "kms>CreateGrant"  
  ],  
  "Resource": "*",  
  "Condition": {  
    "StringEquals": {  
      "kms:ViaService": "ec2.REGION.amazonaws.com",  
      "kms:CallerAccount": "TARGET-ACCOUNT-ID"  
    }  
  }  
}
```

KMS Key Policy

# KMS Automatic Key Rotation

- AWS-managed KMS Keys: automatically rotated every 1 year
- For Customer-Managed Symmetric KMS Key
  - Automatic key rotation is optionally enabled
  - **Customize Rotation Period** between 90 and 2560 days (default: 365 days)
  - Previous key is kept active so you can decrypt old data
  - New Key has the same KMS Key ID (only the backing key is changed)

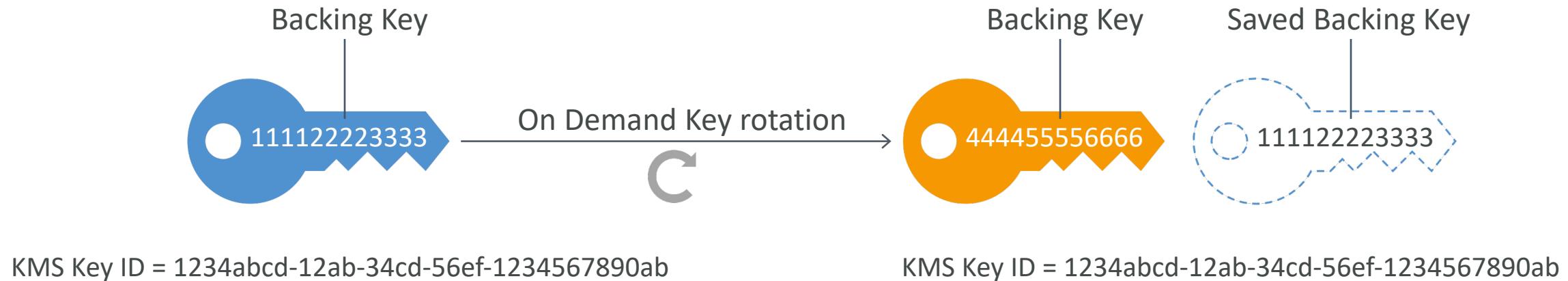


KMS Key ID = 1234abcd-12ab-34cd-56ef-1234567890ab

KMS Key ID = 1234abcd-12ab-34cd-56ef-1234567890ab

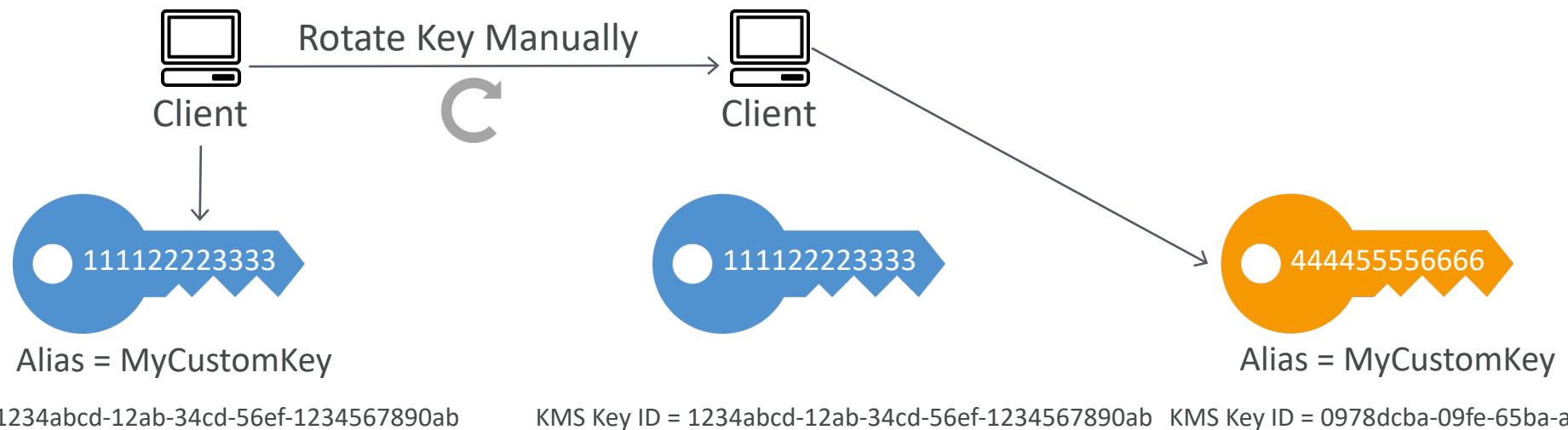
# KMS On-Demand Key Rotation

- For Customer-Managed Symmetric KMS Key (not AWS managed CMK)
- Does NOT require Automatic Key Rotation to be enabled
- Does NOT change existing Automatic Rotation schedules
- Limit to how many times you can trigger an on-demand key rotation



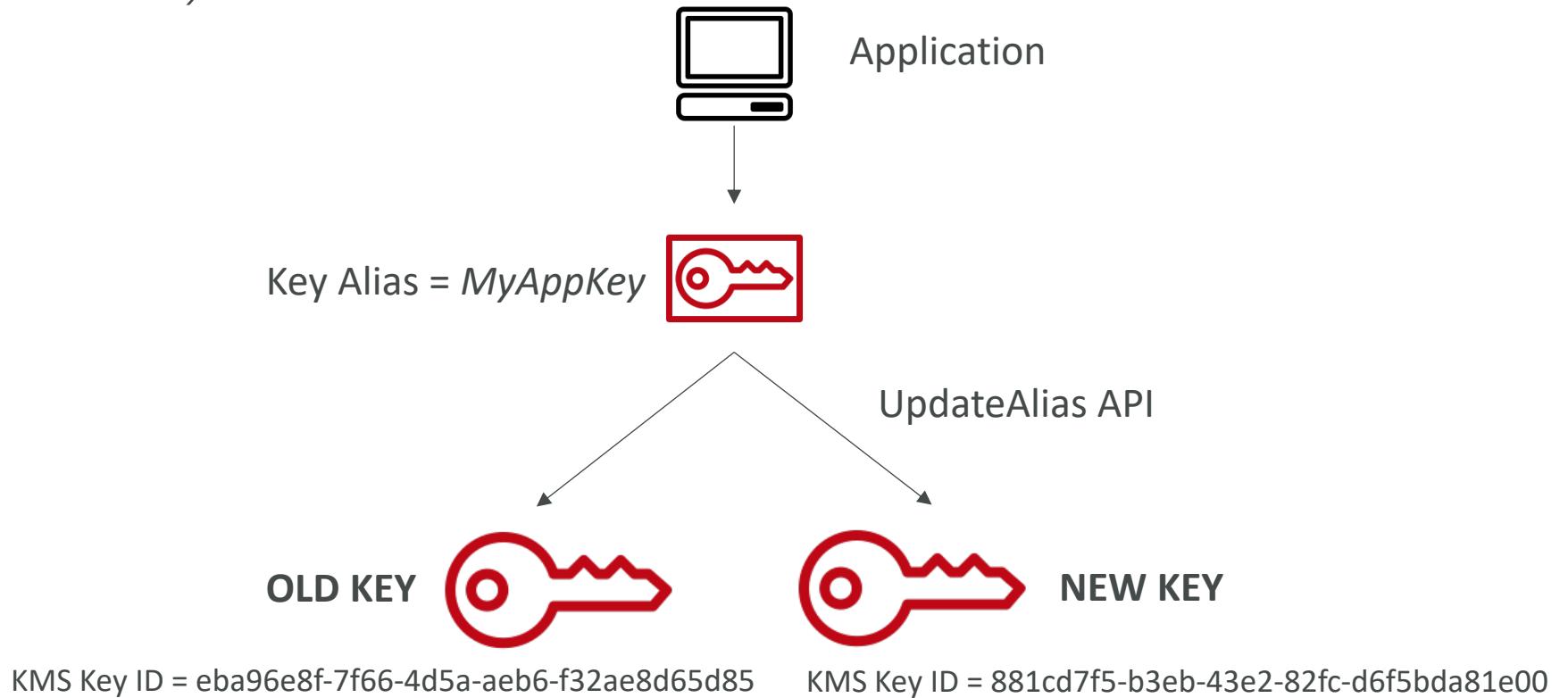
# KMS Manual Key Rotation (Customer-Managed Symmetric KMS Key & Imports)

- When you want to rotate key (example: every month)
- New Key has a different KMS Key ID
- Keep the previous key active so you can decrypt old data
- Better to use aliases in this case (to hide the change of key for the application)
- Good solution to rotate KMS Key that are not eligible for automatic rotation (like **asymmetric KMS Key**)



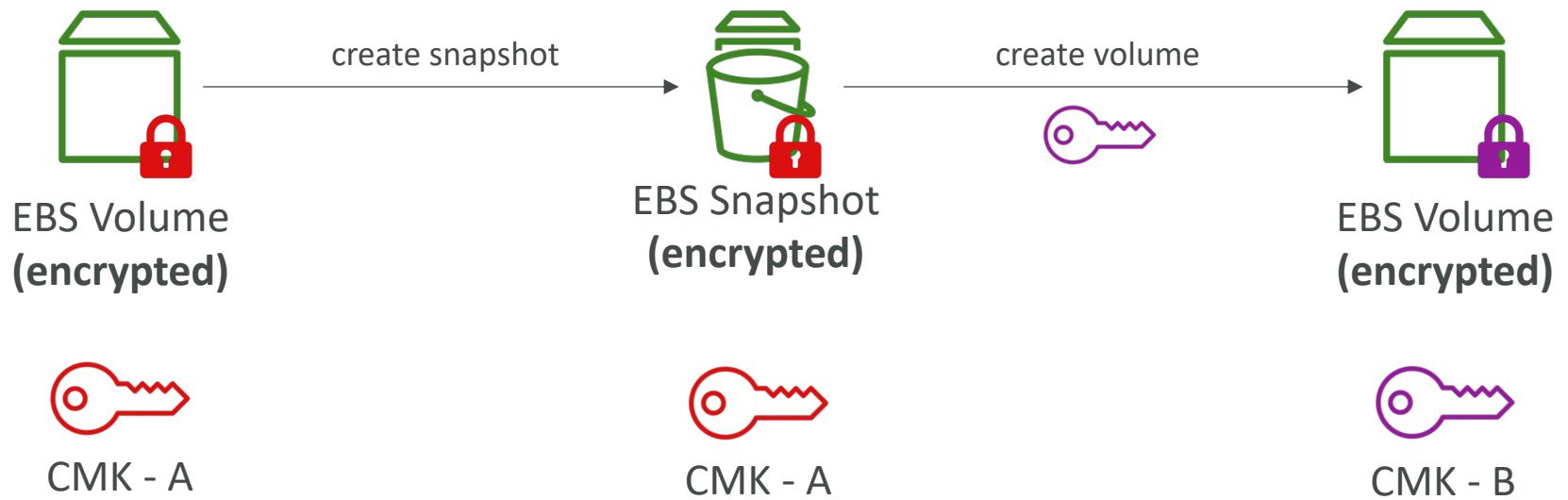
# KMS Alias Updating

- Better to use aliases in this case (to hide the change of key for the application)



# Changing The KMS Key For An Encrypted EBS Volume

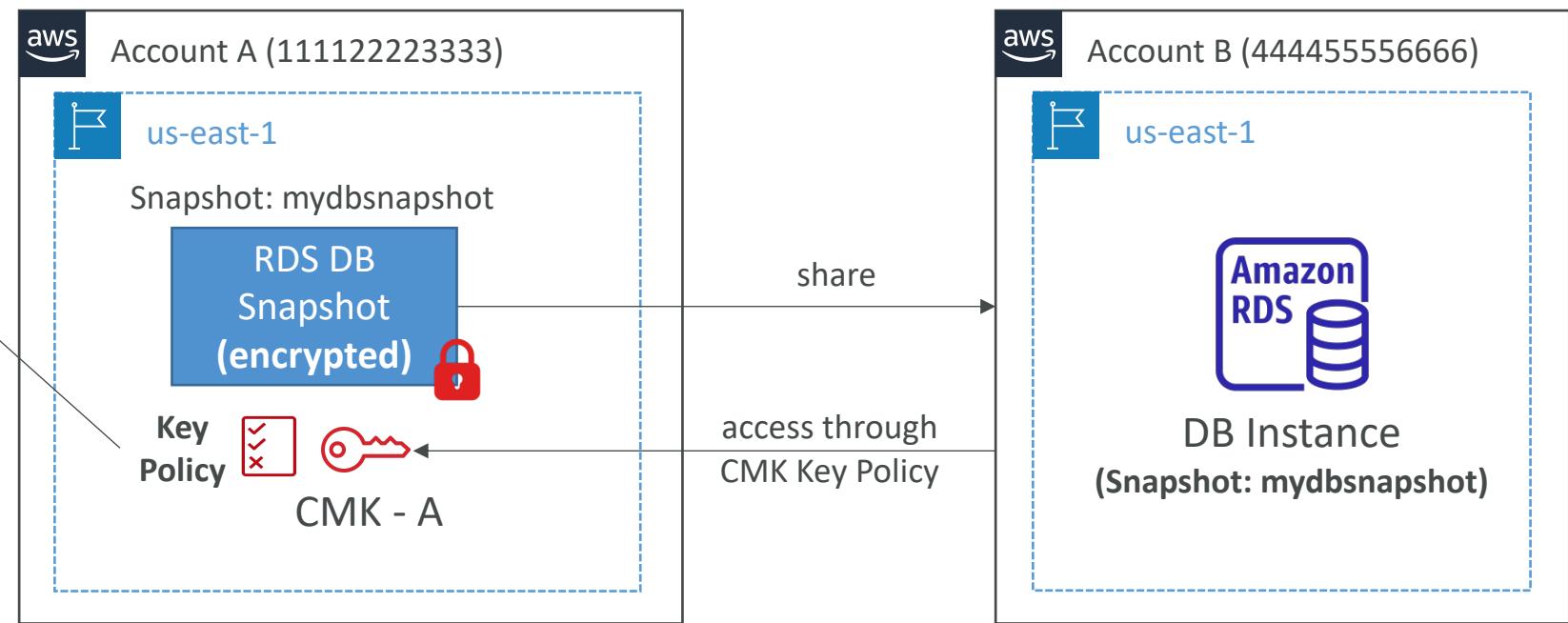
- You can't change the encryption keys used by an EBS volume
- Create an EBS snapshot and create a new EBS volume and specify the new KMS key



# Sharing KMS Encrypted RDS DB Snapshots

- You can share RDS DB snapshots encrypted with KMS CMK with other accounts, but must first share the KMS CMK with the target account using Key Policy

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": ["arn:aws:iam::444455556666:root"]
      },
      "Action": [
        "kms:CreateGrant",
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
  ]
}
```

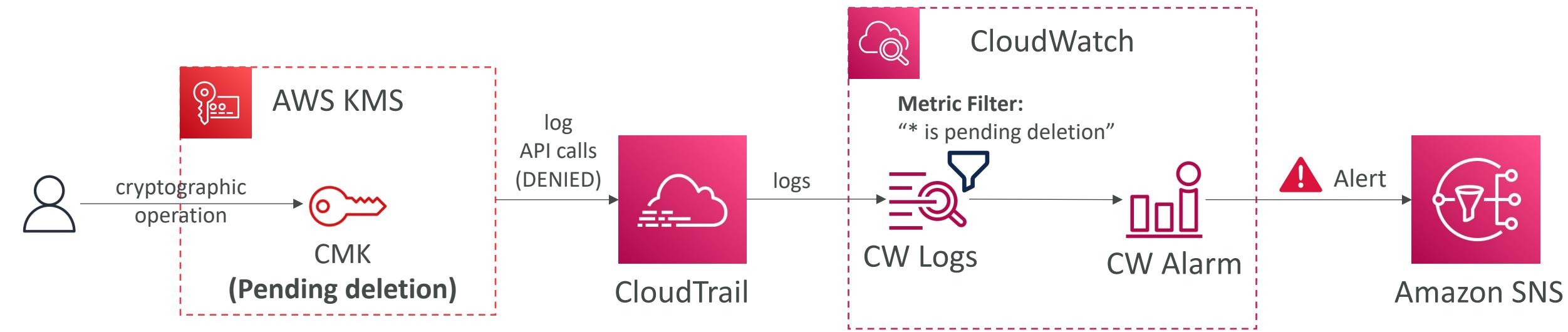


# KMS Key Deletion Considerations

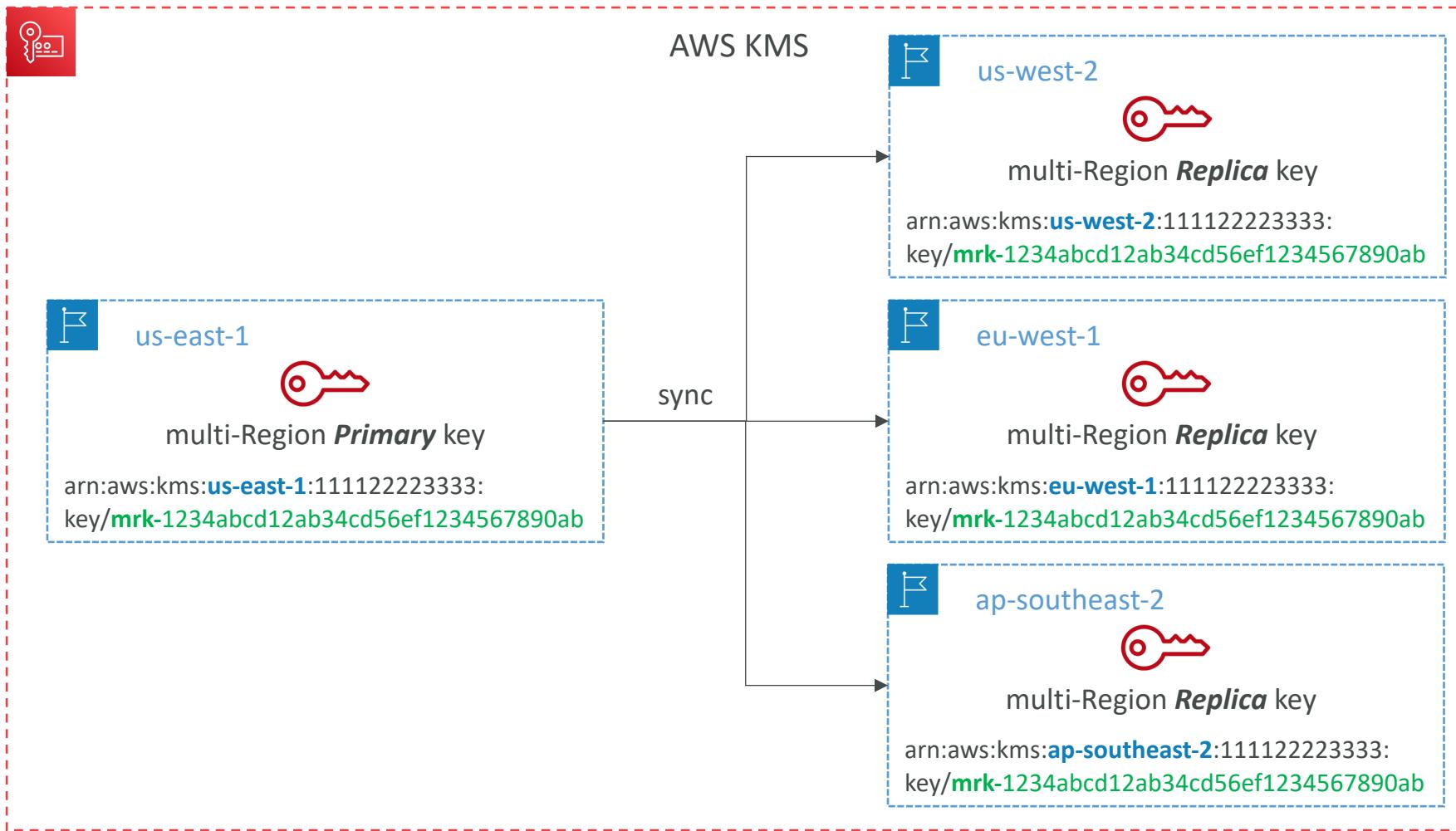
- Schedule CMK for deletion with a waiting period of 7 to 30 days
  - CMK's status is “Pending deletion” during the waiting period
- During the CMK's deletion waiting period:
  - The CMK can't be used for cryptographic operations (e.g., can't decrypt KMS-encrypted objects in S3 – SSE-KMS)
  - The key is not rotated even if planned
- You can cancel the key deletion during the waiting period
- Consider disabling your key instead of deleting it if you're not sure!

# KMS Key Deletion – CloudWatch Alarm

- Use CloudTrail, CloudWatch Logs, CloudWatch Alarms and SNS to be notified when someone tries to use a CMK that's "Pending deletion" in a cryptographic operation (Encrypt, Decrypt, ...)



# KMS Multi-Region Keys



# KMS Multi-Region Keys

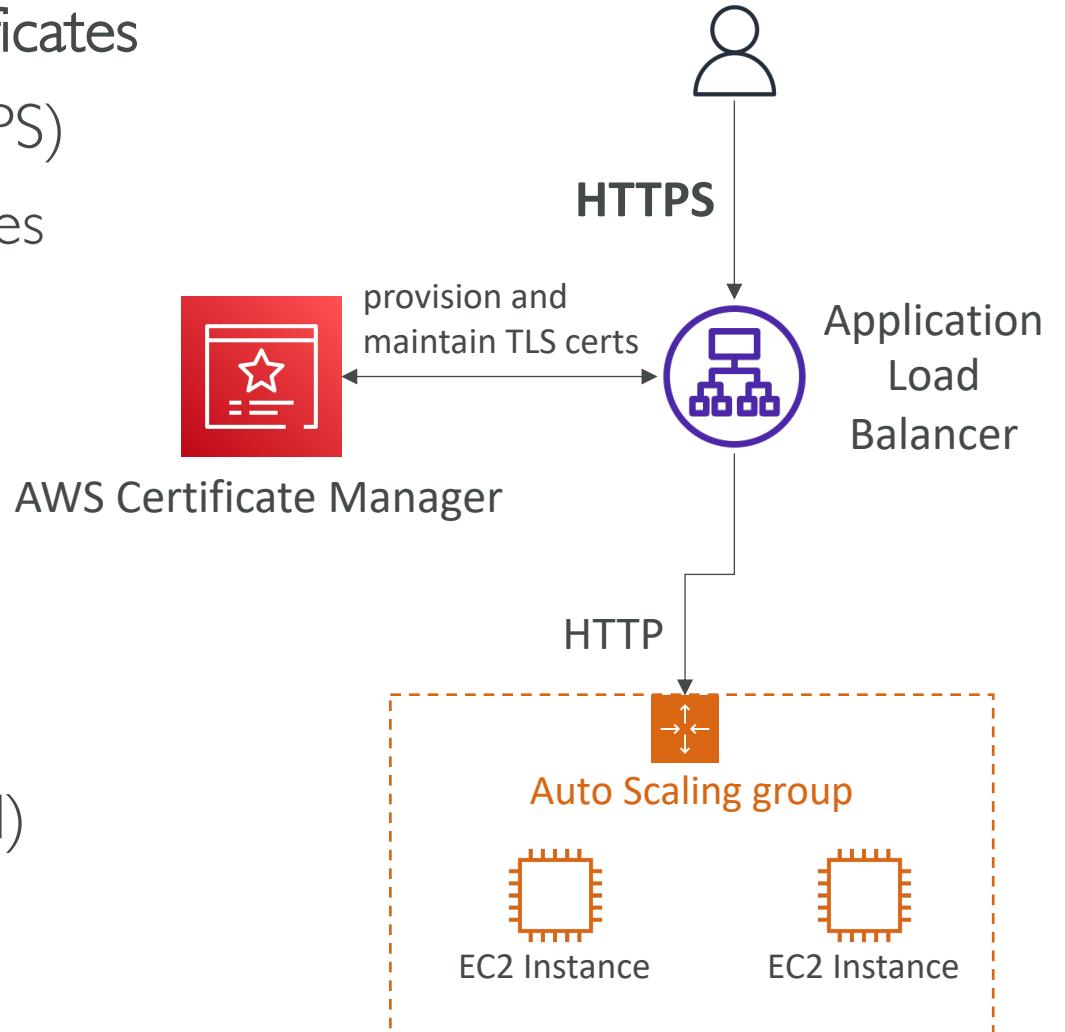


- Identical KMS keys in different AWS Regions that can be used interchangeably
- Multi-Region keys have the same key ID, key material, automatic rotation...
- Encrypt in one Region and decrypt in other Regions
- No need to re-encrypt or making cross-Region API calls
- KMS Multi-Region are NOT global (Primary + Replicas)
- Each Multi-Region key is managed **independently**
- **Use cases:** global client-side encryption, encryption on Global DynamoDB, Global Aurora

# AWS Certificate Manager (ACM)



- Easily provision, manage, and deploy **TLS Certificates**
- 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 (CLB, ALB, NLB)
  - CloudFront Distributions
  - APIs on API Gateway
- Cannot use ACM with EC2 (can't be extracted)

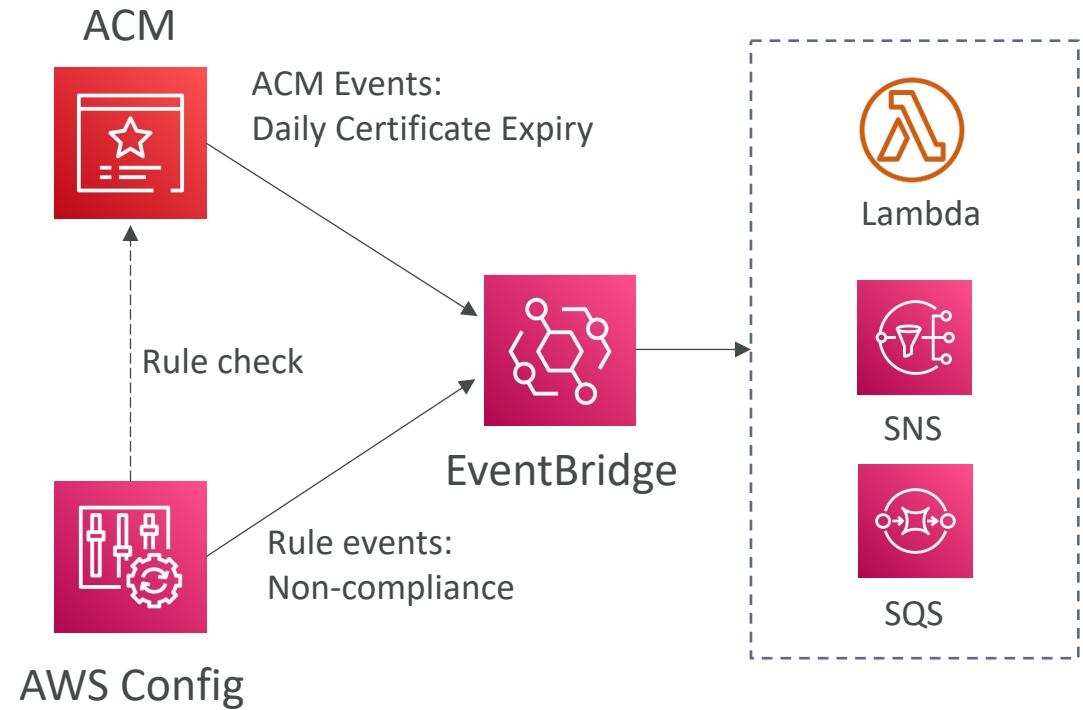


# ACM – Requesting Public Certificates

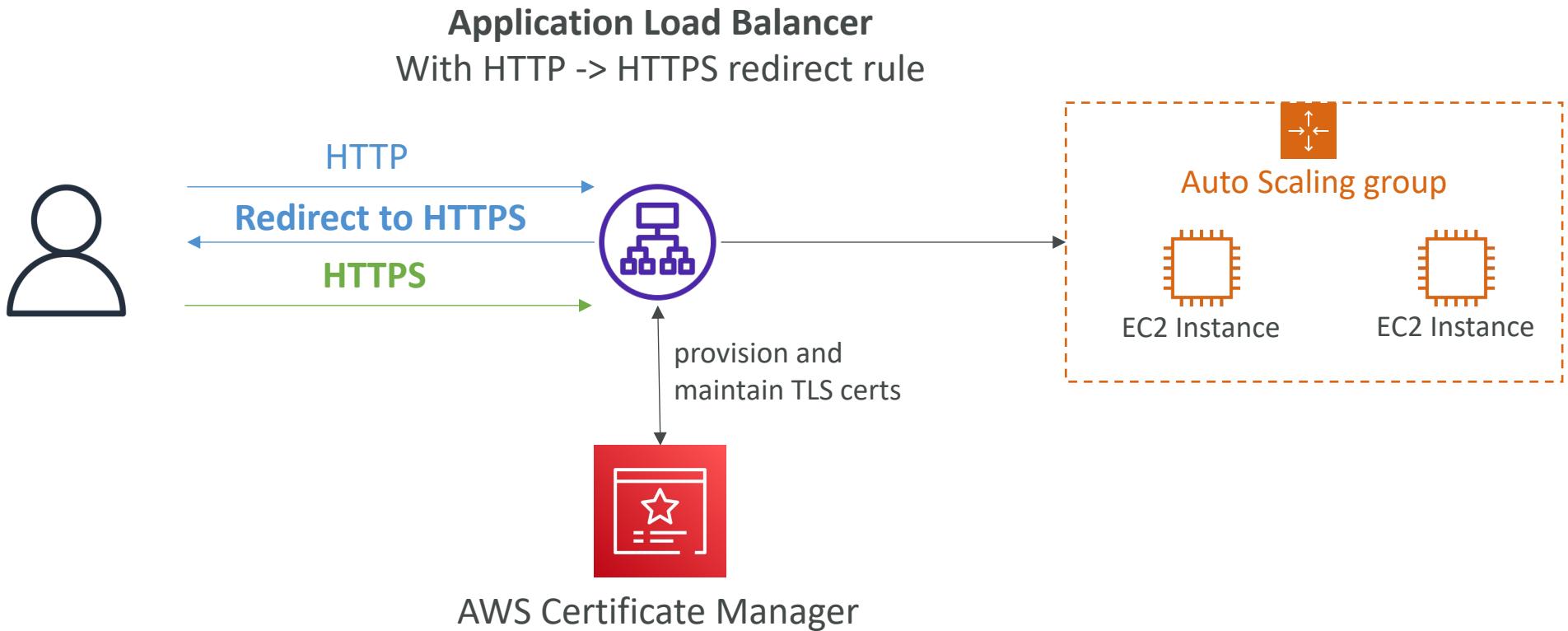
1. List domain names to be included in the certificate
  - Fully Qualified Domain Name (FQDN): corp.example.com
  - Wildcard Domain: \*.example.com
2. Select Validation Method: DNS Validation or Email validation
  - DNS Validation is preferred for automation purposes
  - Email validation will send emails to contact addresses in the WHOIS database
  - DNS Validation will leverage a CNAME record to DNS config (ex: Route 53)
3. It will take a few hours to get verified
4. The Public Certificate will be enrolled for automatic renewal
  - ACM automatically renews ACM-generated certificates 60 days before expiry

# ACM – Importing Public Certificates

- Option to generate the certificate outside of ACM and then import it
- No automatic renewal, must import a new certificate before expiry
- ACM sends daily expiration events starting 45 days prior to expiration
  - The # of days can be configured
  - Events are appearing in EventBridge
- AWS Config has a managed rule named `acm-certificate-expiration-check` to check for expiring certificates (configurable number of days)



# ACM – Integration with ALB

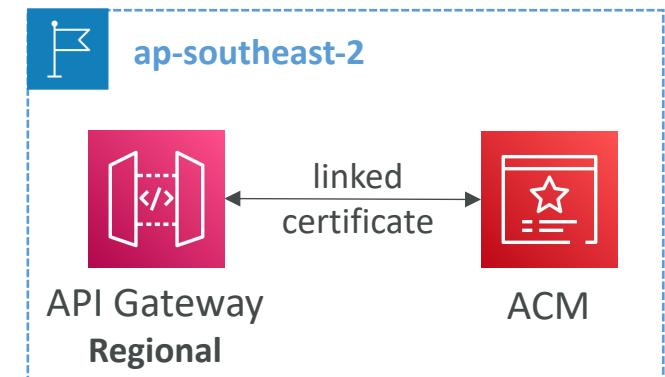
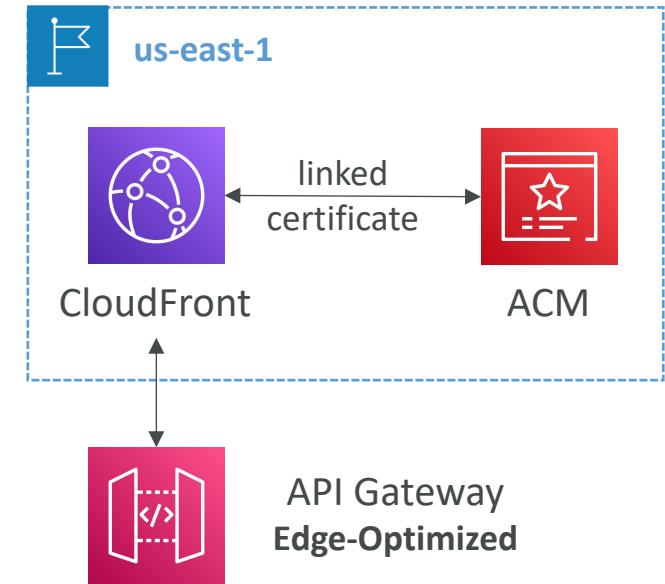


# API Gateway - Endpoint Types

- **Edge-Optimized (default):** For global clients
  - Requests are routed through the CloudFront Edge locations (improves latency)
  - The API Gateway still lives in only one region
- **Regional:**
  - For clients within the same region
  - Could manually combine with CloudFront (more control over the caching strategies and the distribution)
- **Private:**
  - Can only be accessed from your VPC using an interface VPC endpoint (ENI)
  - Use a resource policy to define access

# ACM – Integration with API Gateway

- Create a **Custom Domain Name** in API Gateway
- **Edge-Optimized (default):** For global clients
  - Requests are routed through the CloudFront Edge locations (improves latency)
  - The API Gateway still lives in only one region
  - The TLS Certificate must be in the same region as CloudFront, in us-east-1
  - Then setup CNAME or (better) A-Alias record in Route 53
- **Regional:**
  - For clients within the same region
  - The TLS Certificate must be imported on API Gateway, in the same region as the API Stage
  - Then setup CNAME or (better) A-Alias record in Route 53



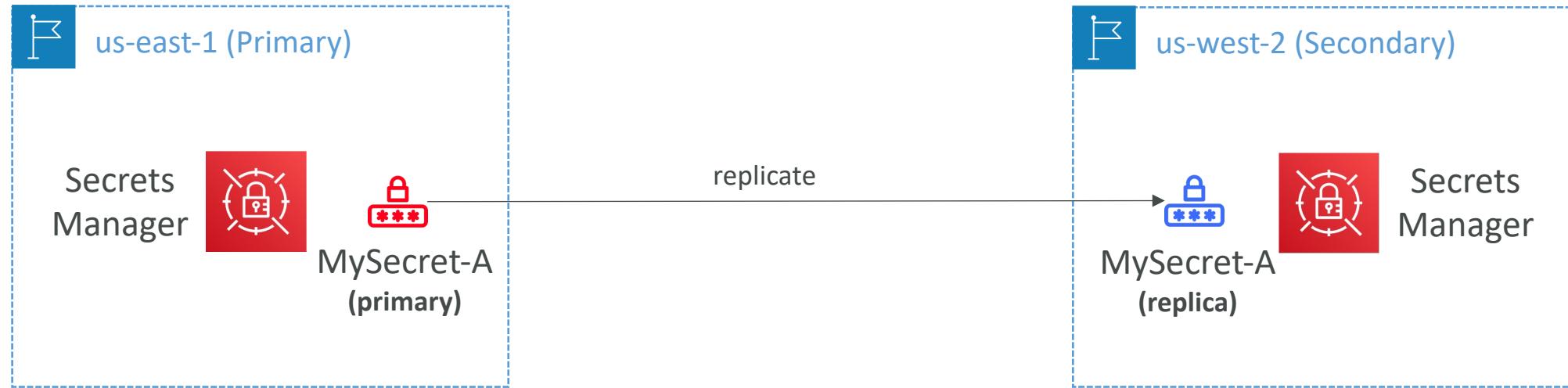
# AWS Secrets Manager



- 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 Secrets Manager – Multi-Region Secrets

- Replicate Secrets across multiple AWS Regions
- Secrets Manager keeps read replicas in sync with the primary Secret
- Ability to promote a read replica Secret to a standalone Secret
- Use cases: multi-region apps, disaster recovery strategies, multi-region DB...



# SSM Parameter Store vs Secrets Manager

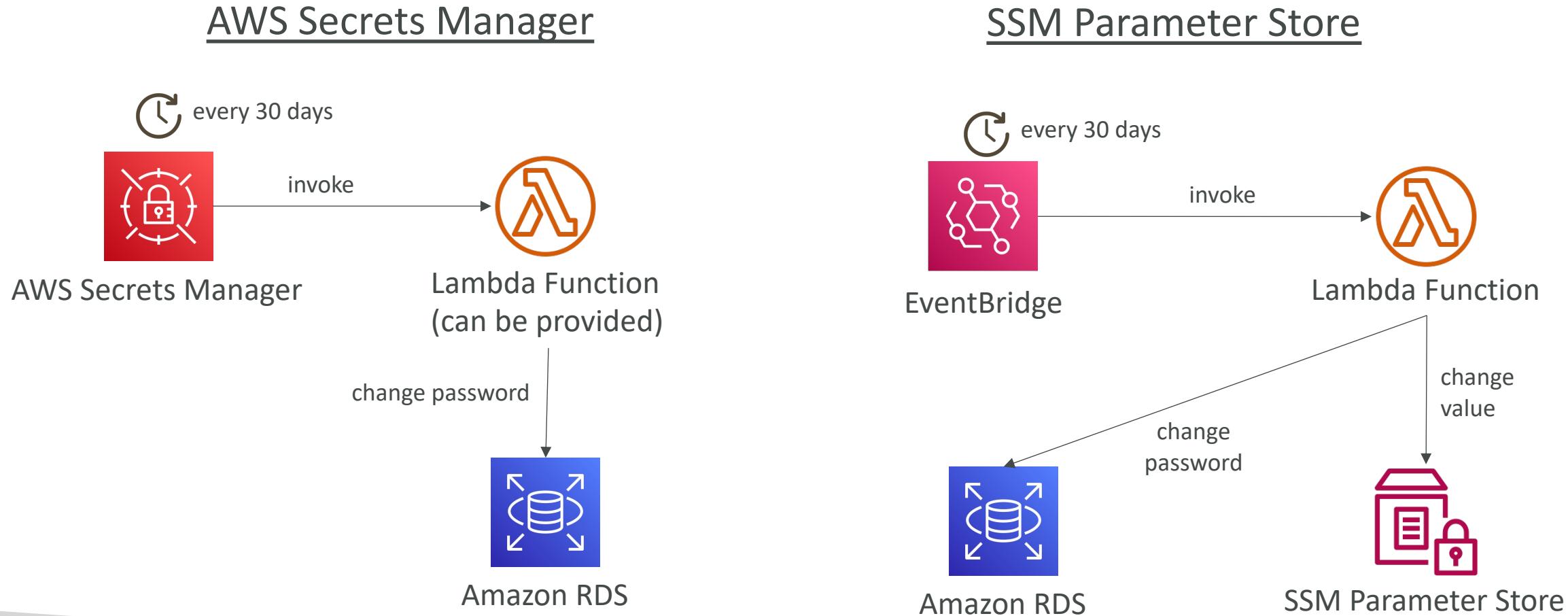
- **Secrets Manager (\$\$\$):**

- Automatic rotation of secrets with AWS Lambda
- Lambda function is provided for RDS, Redshift, DocumentDB
- KMS encryption is mandatory
- Can integration with CloudFormation

- **SSM Parameter Store (\$):**

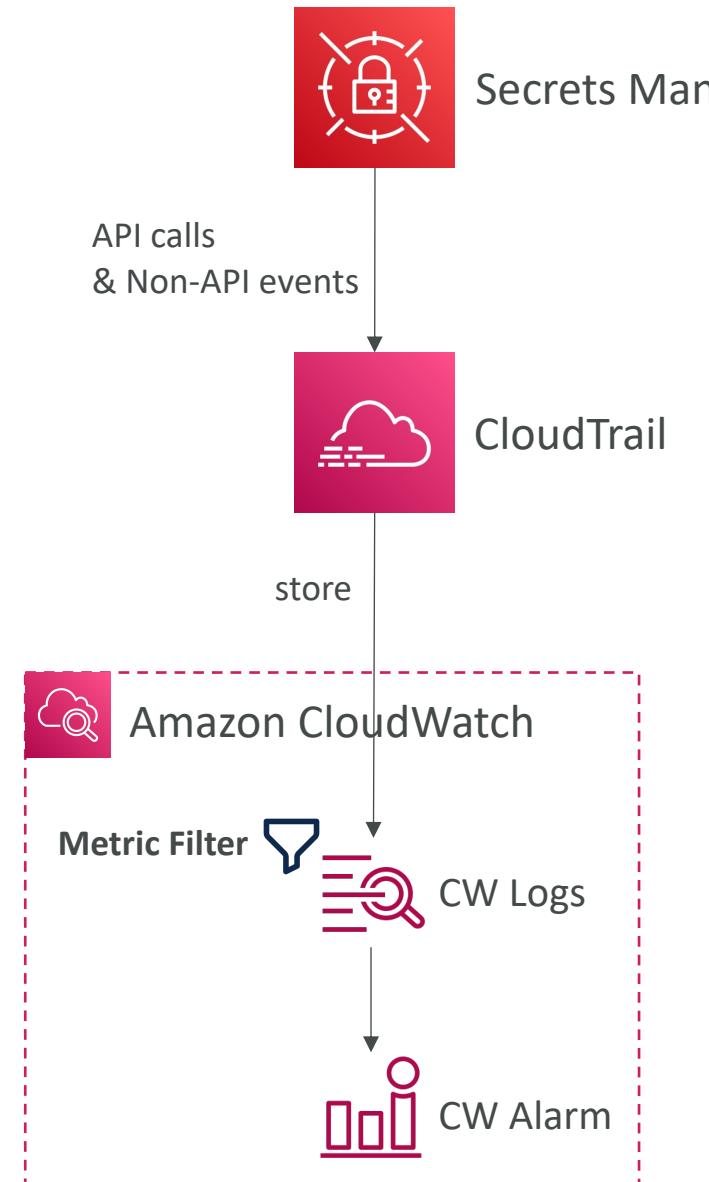
- Simple API
- No secret rotation (can enable rotation using Lambda triggered by EventBridge)
- KMS encryption is optional
- Can integration with CloudFormation
- Can pull a Secrets Manager secret using the SSM Parameter Store API

# SSM Parameter Store vs. Secrets Manager Rotation

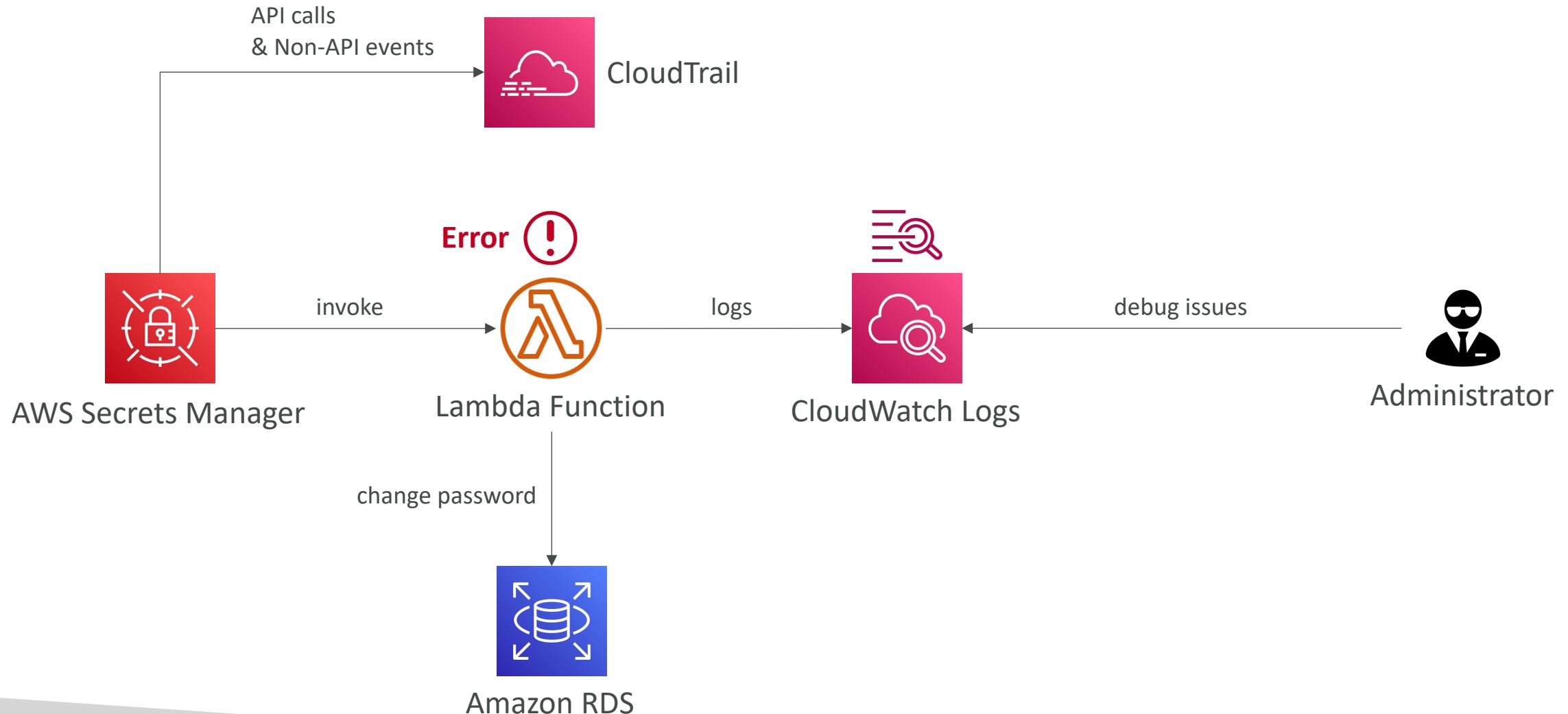


# Secrets Manager – Monitoring

- CloudTrail captures API calls to the Secrets Manager API
- CloudTrail captures other related events that might have a security or compliance impact on your AWS account or might help you troubleshoot operational problems.
- **CloudTrail records these events as non-API service events.**
  - RotationStarted event
  - RotationSucceeded event
  - RotationFailed event
  - RotationAbandoned event – a manual change to a secret instead of automated rotation
  - StartSecretVersionDelete event
  - CancelSecretVersionDelete event
  - EndSecretVersionDelete event
- Combine with CloudWatch Logs and CloudWatch alarms for automations



# Secrets Manager – Troubleshooting Rotation



# Identity

# IAM Permission Boundaries

- IAM Permission Boundaries are supported for users and roles (not groups)
- Advanced feature to use a managed policy to set the maximum permissions an IAM entity can get.

**Example:**

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "s3:*",  
                "cloudwatch:*",  
                "ec2:*"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```



**IAM Permission Boundary**

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "iam>CreateUser",  
            "Resource": "*"  
        }  
    ]  
}
```

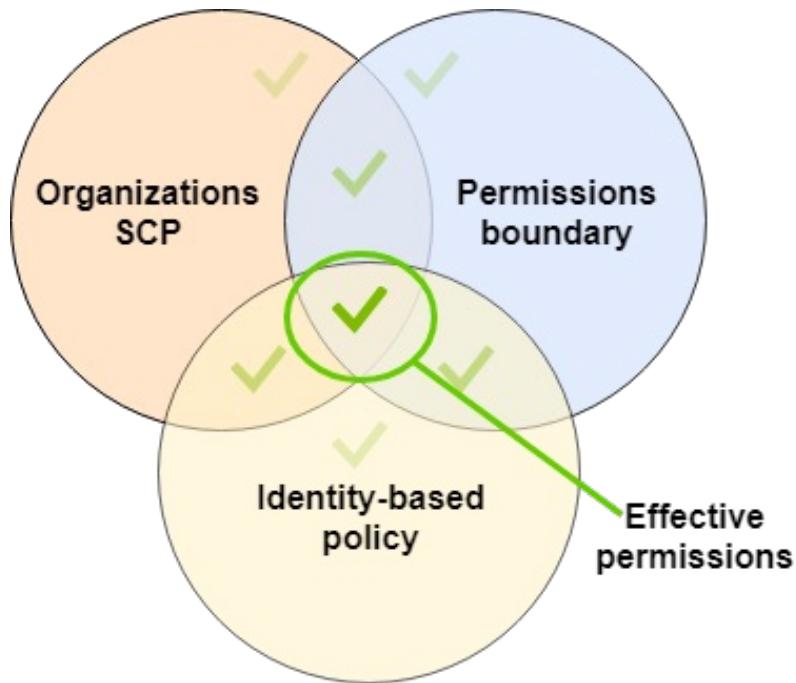
**IAM Permissions  
Through IAM Policy**



**No Permissions**

# IAM Permission Boundaries

- Can be used in combinations of AWS Organizations SCP



[https://docs.aws.amazon.com/IAM/latest/UserGuide/access\\_policies\\_boundaries.html](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_boundaries.html)

## Use cases

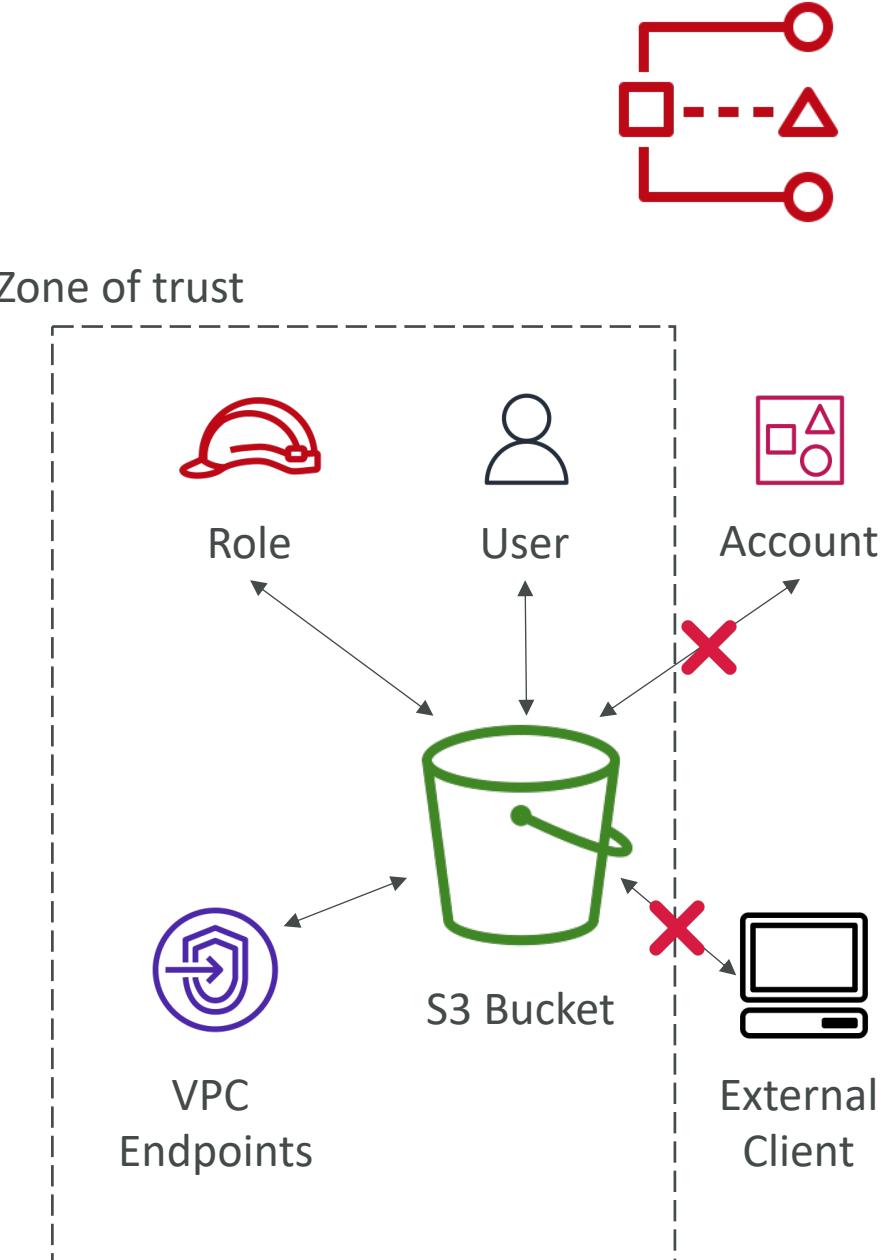
- Delegate responsibilities to non administrators within their permission boundaries, for example create new IAM users
- Allow developers to self-assign policies and manage their own permissions, while making sure they can't "escalate" their privileges (= make themselves admin)
- Useful to restrict one specific user (instead of a whole account using Organizations & SCP)

# 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 Access Analyzer

- Find out which resources are shared externally
  - S3 Buckets
  - IAM Roles
  - KMS Keys
  - Lambda Functions and Layers
  - SQS queues
  - Secrets Manager Secrets
- Define **Zone of Trust** = AWS Account or AWS Organization
- Access outside zone of trusts => findings



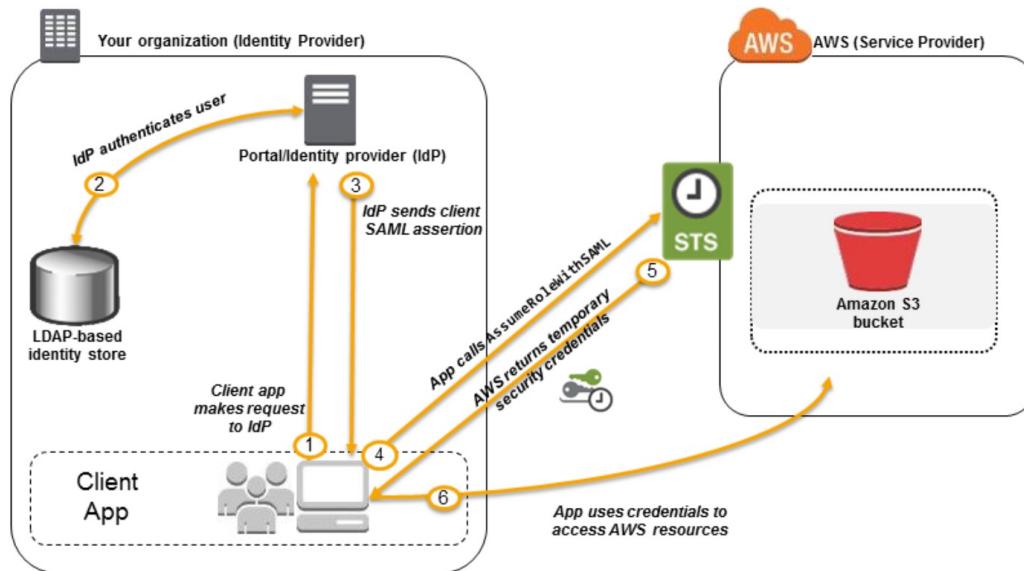
# What's Identity Federation?

- Federation lets users outside of AWS to assume temporary role for accessing AWS resources.
- These users assume identity provided access role.
- Federation assumes a form of 3rd party authentication
  - LDAP
  - Microsoft Active Directory ( $\sim$ = SAML)
  - Single Sign On
  - Open ID
  - Cognito
- Using federation, you don't need to create IAM users (user management is outside of AWS)

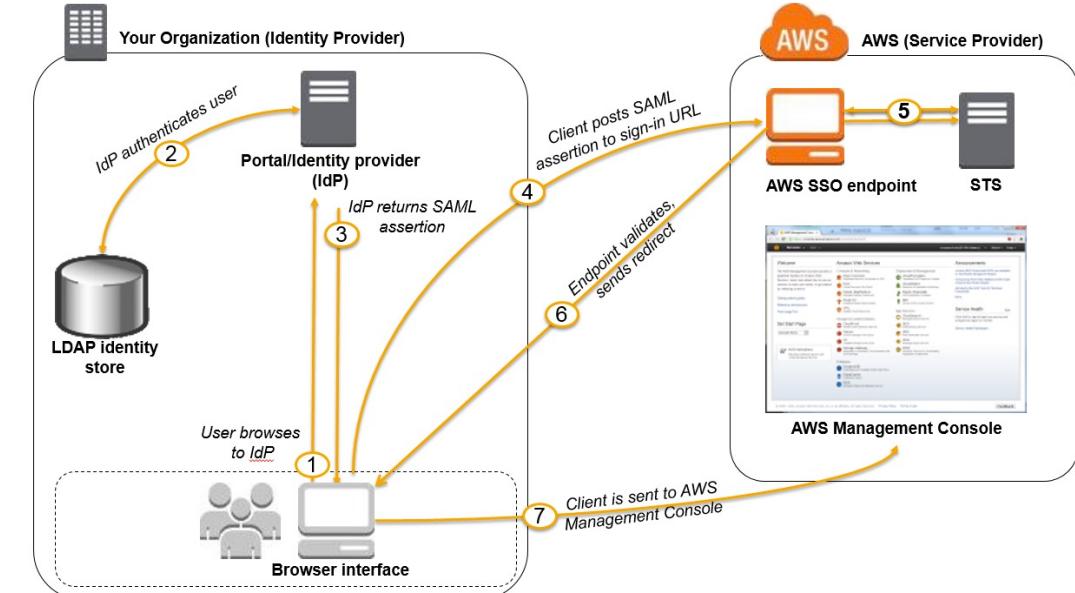


# SAML Federation For Enterprises

- To integrate Active Directory / ADFS with AWS (or any SAML 2.0)
- Provides access to AWS Console or CLI (through temporary creds)
- No need to create an IAM user for each of your employees



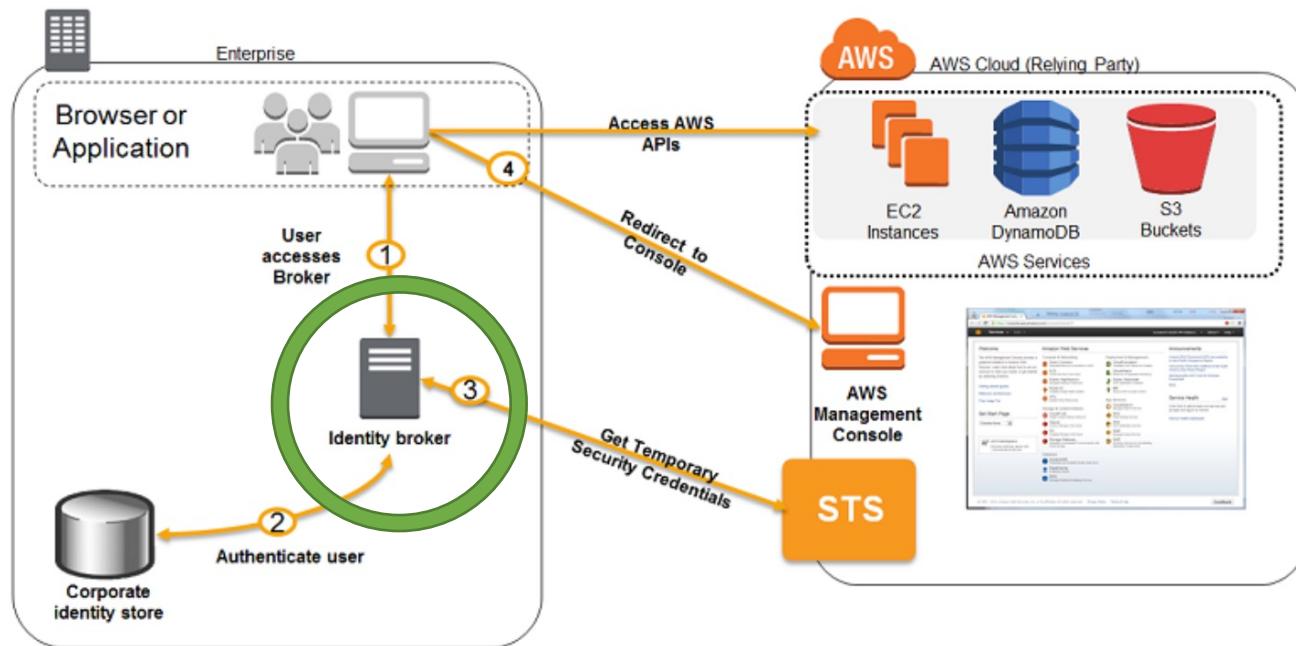
[https://docs.aws.amazon.com/IAM/latest/UserGuide/id\\_roles\\_providers\\_saml.html](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers_saml.html)



[https://docs.aws.amazon.com/IAM/latest/UserGuide/id\\_roles\\_providers\\_enable-console-saml.html](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_providers_enable-console-saml.html)

# Custom Identity Broker Application For Enterprises

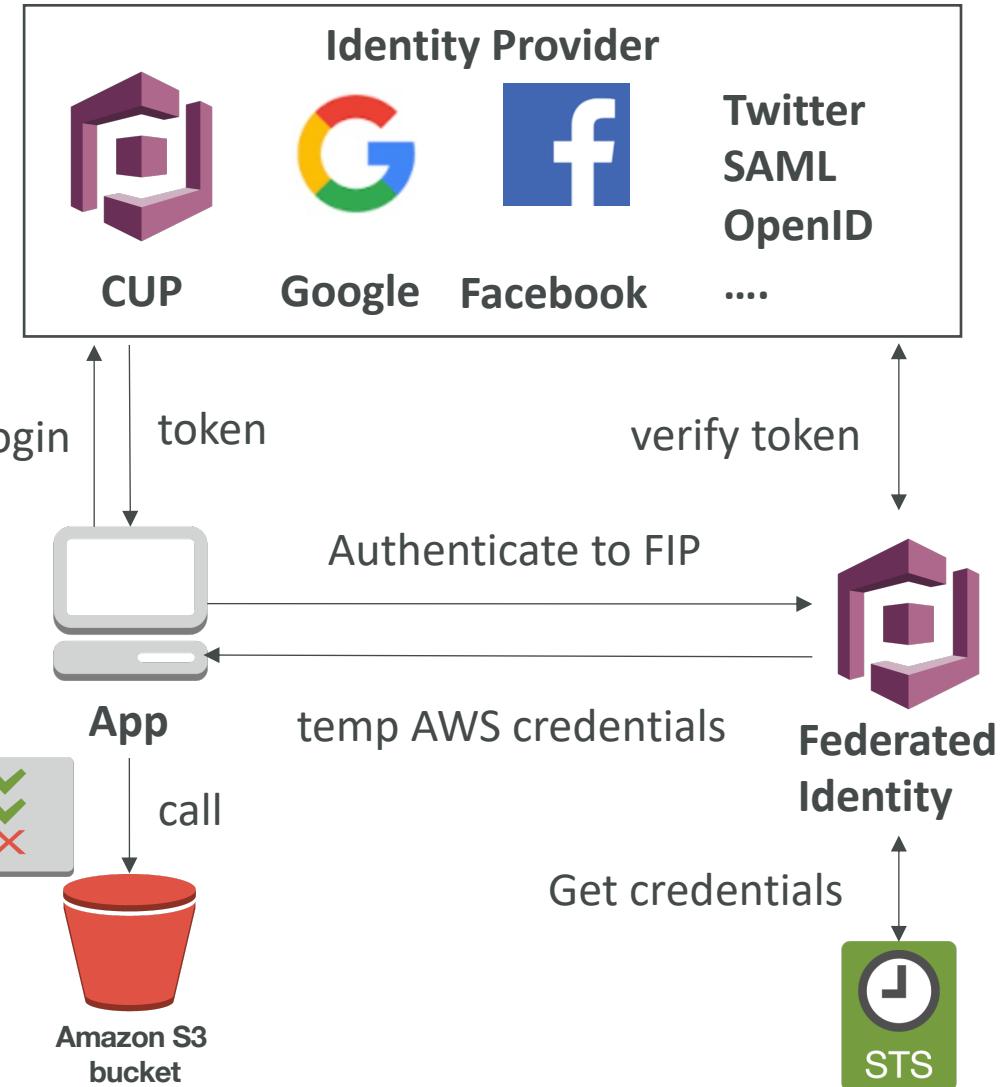
- Use only if identity provider is not compatible with SAML 2.0
- The identity broker must determine the appropriate IAM policy



[https://docs.aws.amazon.com/IAM/latest/UserGuide/id\\_roles\\_common-scenarios\\_federated-users.html](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_common-scenarios_federated-users.html)

# AWS Cognito - Federated Identity Pools For Public Applications

- Goal:
  - Provide direct access to AWS Resources from the Client Side
- How:
  - Log in to federated identity provider – or remain anonymous
  - Get temporary AWS credentials back from the Federated Identity Pool
  - These credentials come with a pre-defined IAM policy stating their permissions
- Example:
  - provide (temporary) access to write to S3 bucket using Facebook Login
- Note:
  - Web Identity Federation is an alternative to using Cognito but AWS recommends against it



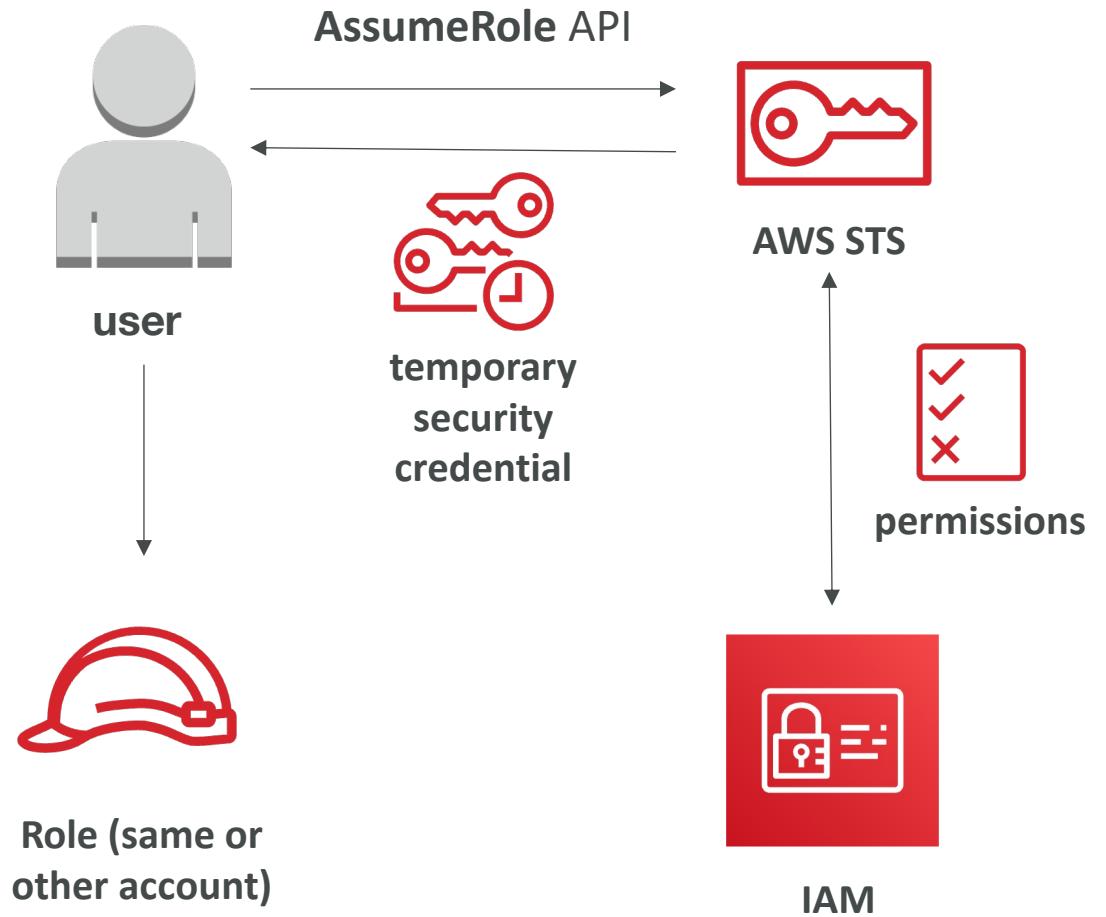


# AWS STS – Security Token Service

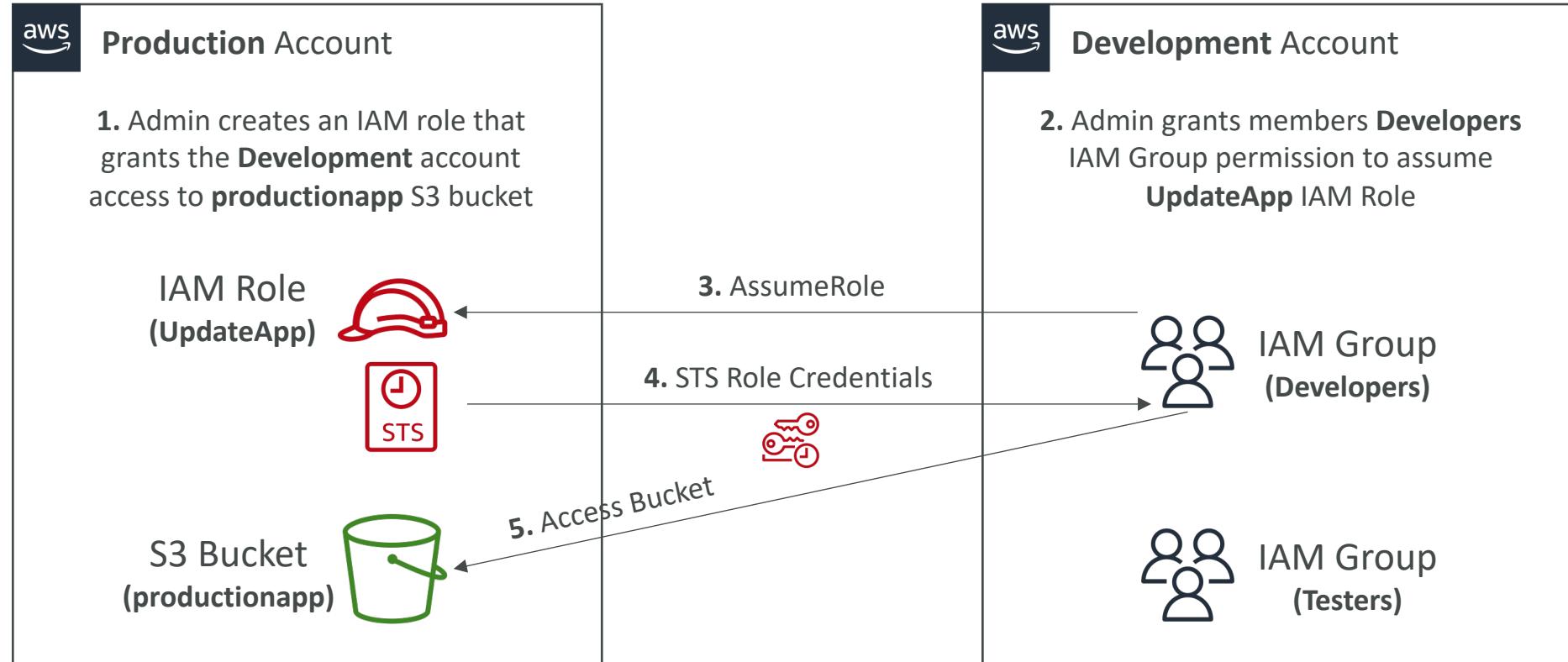
- Allows to grant limited and temporary access to AWS resources.
- Token is valid for up to one hour (must be refreshed)
- **AssumeRole**
  - Within your own account: for enhanced security
  - Cross Account Access: assume role in target account to perform actions there
- **AssumeRoleWithSAML**
  - return credentials for users logged with SAML
- **AssumeRoleWithWebIdentity**
  - return creds for users logged with an IdP (Facebook Login, Google Login, OIDC compatible...)
  - AWS recommends against using this, and using **Cognito** instead
- **GetSessionToken**
  - for MFA, from a user or AWS account root user

# Using STS to Assume a Role

- Define an IAM Role within your account or cross-account
- Define which principals can access this IAM Role
- Use AWS STS (Security Token Service) to retrieve credentials and impersonate the IAM Role you have access to (`AssumeRole API`)
- Temporary credentials can be valid between 15 minutes to 1 hour



# Cross-Account Access with STS



# IAM Policy Simulator

- Allows you to test and troubleshoot IAM policies before applying them in an AWS environment
- Helps you understand what permissions a user, group, or role has
- Example: test if a user can perform action `s3:PutObject` on S3 bucket
- Works with **Identity-based Policies, Resource-based Policies, Permission Boundaries, SCPs**
- Use Cases:
  - Test Identity-based Policies attached to IAM Users, Groups, or Roles
  - Test all policies or select one or more policies attached to a User
  - Test the effect of Resource-based Policy on an IAM User
  - Test the impact of SCP on your Identity-based Policies

# Amazon Route 53

# What is DNS?

- Domain Name System which translates the human friendly hostnames into the machine IP addresses
- www.google.com => 172.217.18.36
- DNS is the backbone of the Internet
- DNS uses hierarchical naming structure

.com

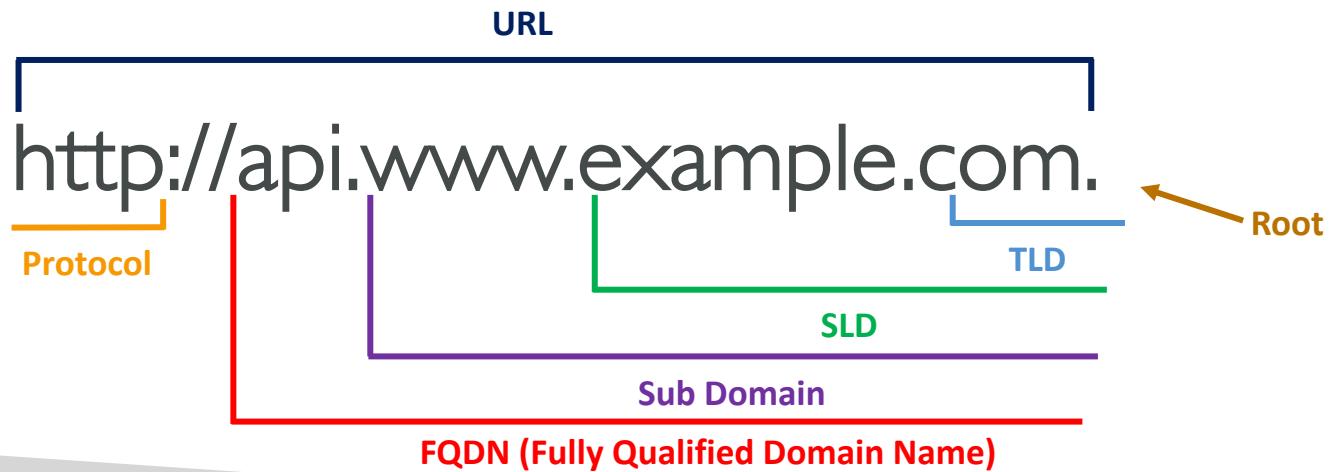
example.com

www.example.com

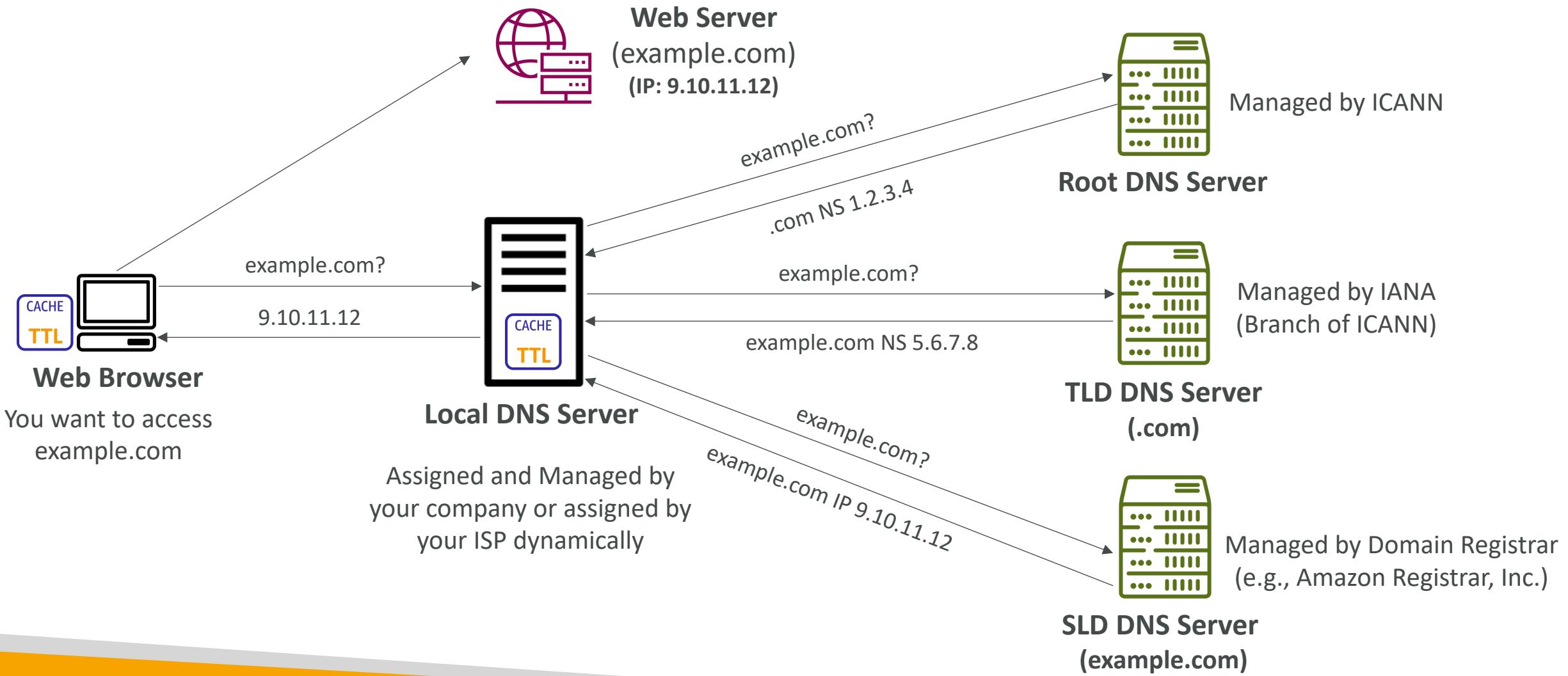
api.example.com

# DNS Terminologies

- Domain Registrar: Amazon Route 53, GoDaddy, ...
- DNS Records: A, AAAA, CNAME, NS, ...
- Zone File: contains DNS records
- Name Server: resolves DNS queries (Authoritative or Non-Authoritative)
- Top Level Domain (TLD): .com, .us, .in, .gov, .org, ...
- Second Level Domain (SLD): amazon.com, google.com, ...

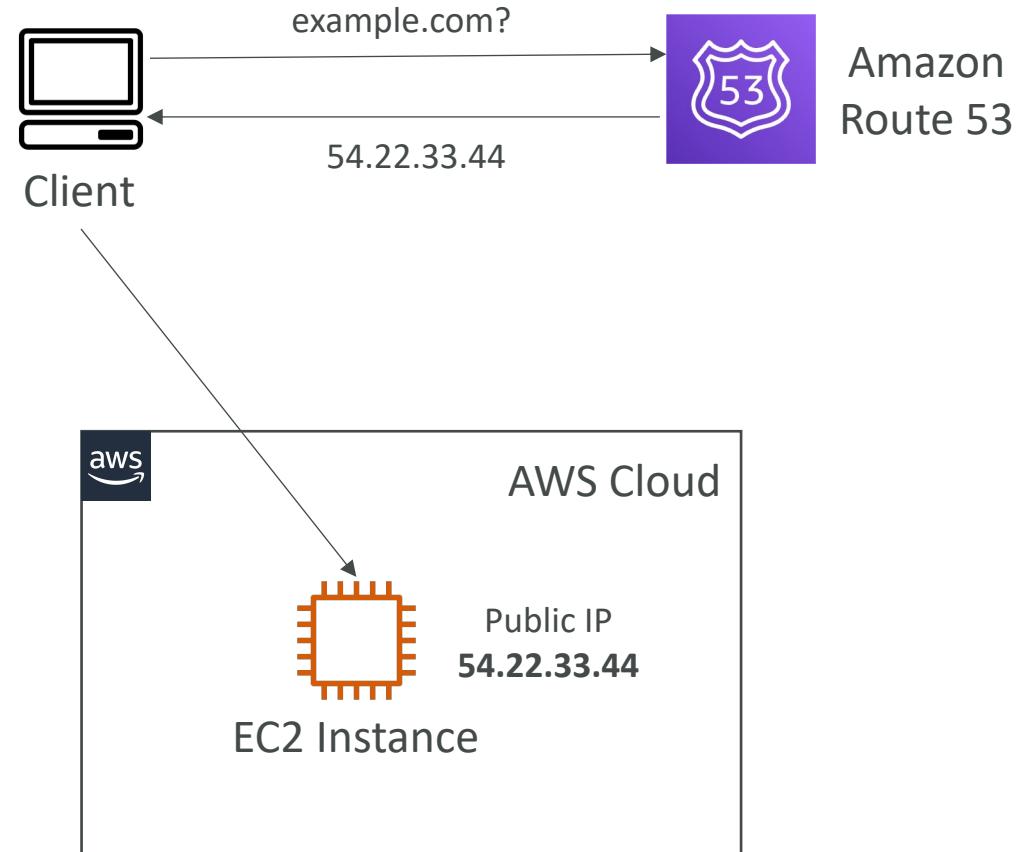


# How DNS Works



# Amazon Route 53

- A highly available, scalable, fully managed and Authoritative DNS
  - Authoritative = the customer (you) can update the DNS records
- Route 53 is also a Domain Registrar
- Ability to check the health of your resources
- The only AWS service which provides 100% availability SLA
- Why Route 53? 53 is a reference to the traditional DNS port



# Route 53 – Records

- How you want to route traffic for a domain
- Each record contains:
  - Domain/subdomain Name – e.g., example.com
  - Record Type – e.g., A or AAAA
  - Value – e.g., 12.34.56.78
  - Routing Policy – how Route 53 responds to queries
  - TTL – amount of time the record cached at DNS Resolvers
- Route 53 supports the following DNS record types:
  - (must know) A / AAAA / CNAME / NS
  - (advanced) CAA / DS / MX / NAPTR / PTR / SOA / TXT / SPF / SRV

# Route 53 – Record Types

- A – maps a hostname to IPv4
- AAAA – maps a hostname to IPv6
- CNAME – maps a hostname to another hostname
  - The target is a domain name which must have an A or AAAA record
  - Can't create a CNAME record for the top node of a DNS namespace (Zone Apex)
  - Example: you can't create for example.com, but you can create for www.example.com
- NS – Name Servers for the Hosted Zone
  - Control how traffic is routed for a domain

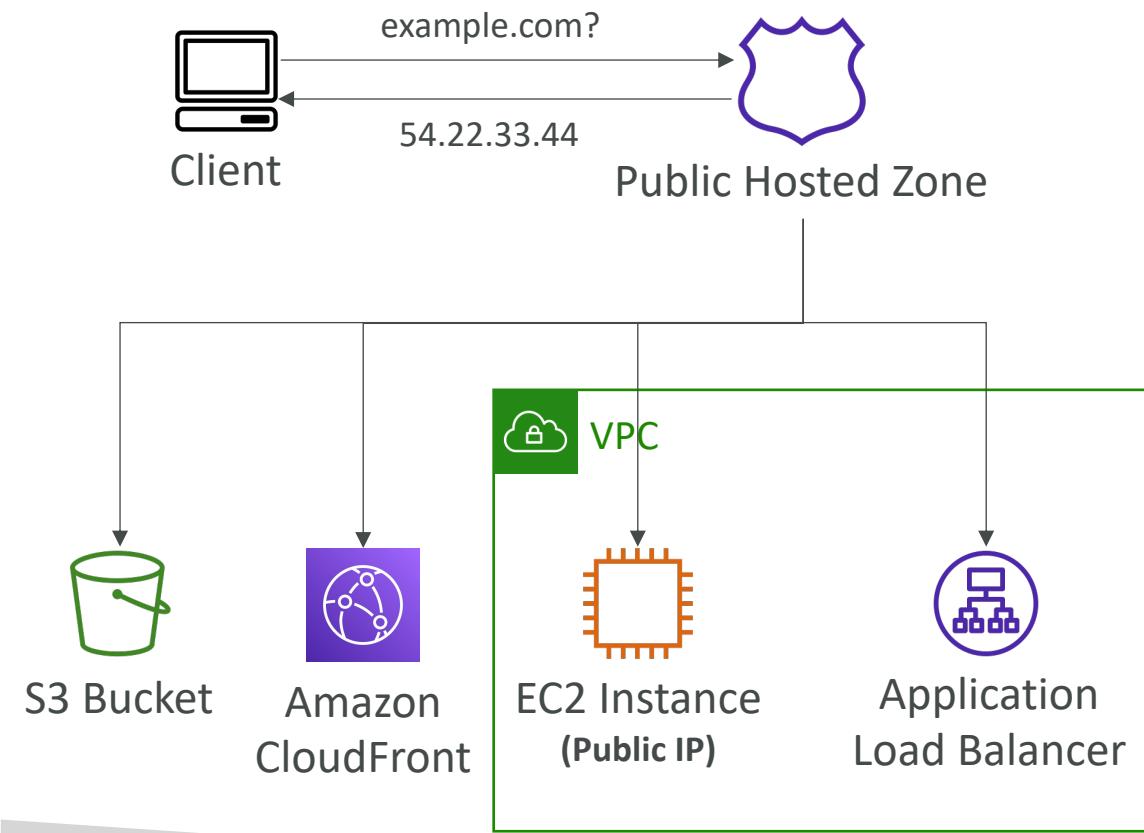


# Route 53 – Hosted Zones

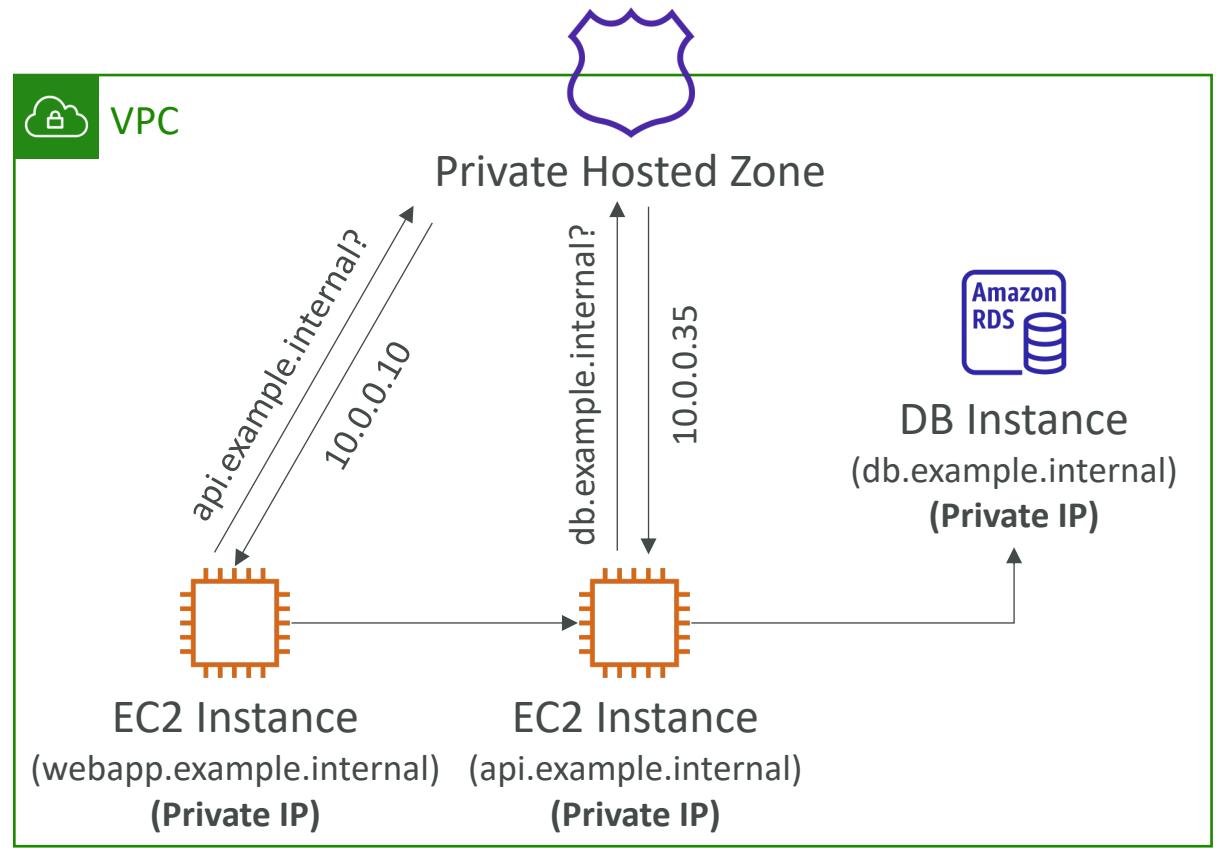
- A container for records that define how to route traffic to a domain and its subdomains
- **Public Hosted Zones** – contains records that specify how to route traffic on the Internet (public domain names)  
[application1.mypublicdomain.com](http://application1.mypublicdomain.com)
- **Private Hosted Zones** – contain records that specify how you route traffic within one or more VPCs (private domain names)  
[application1.company.internal](http://application1.company.internal)
- You pay \$0.50 per month per hosted zone

# Route 53 – Public vs. Private Hosted Zones

## Public Hosted Zone



## Private Hosted Zone



# Common DNS Record Types for Email

Record	Purpose	Example
MX	Mail Exchange – tells where to deliver inbound mail	10 inbound-smtp.us-east-1.amazonaws.com
TXT (SPF)	Sender Policy Framework – authorizes servers allowed to send mail	v=spf1 include:amazonses.com ~all
TXT (DKIM)	DomainKeys Identified Mail – cryptographic signature verifying message integrity	abcdefg12345._domainkey.example.com
TXT (DMARC)	Defines handling policy for messages failing SPF/DKIM	v=DMARC1; p=quarantine; rua=mailto:dmarc@example.com

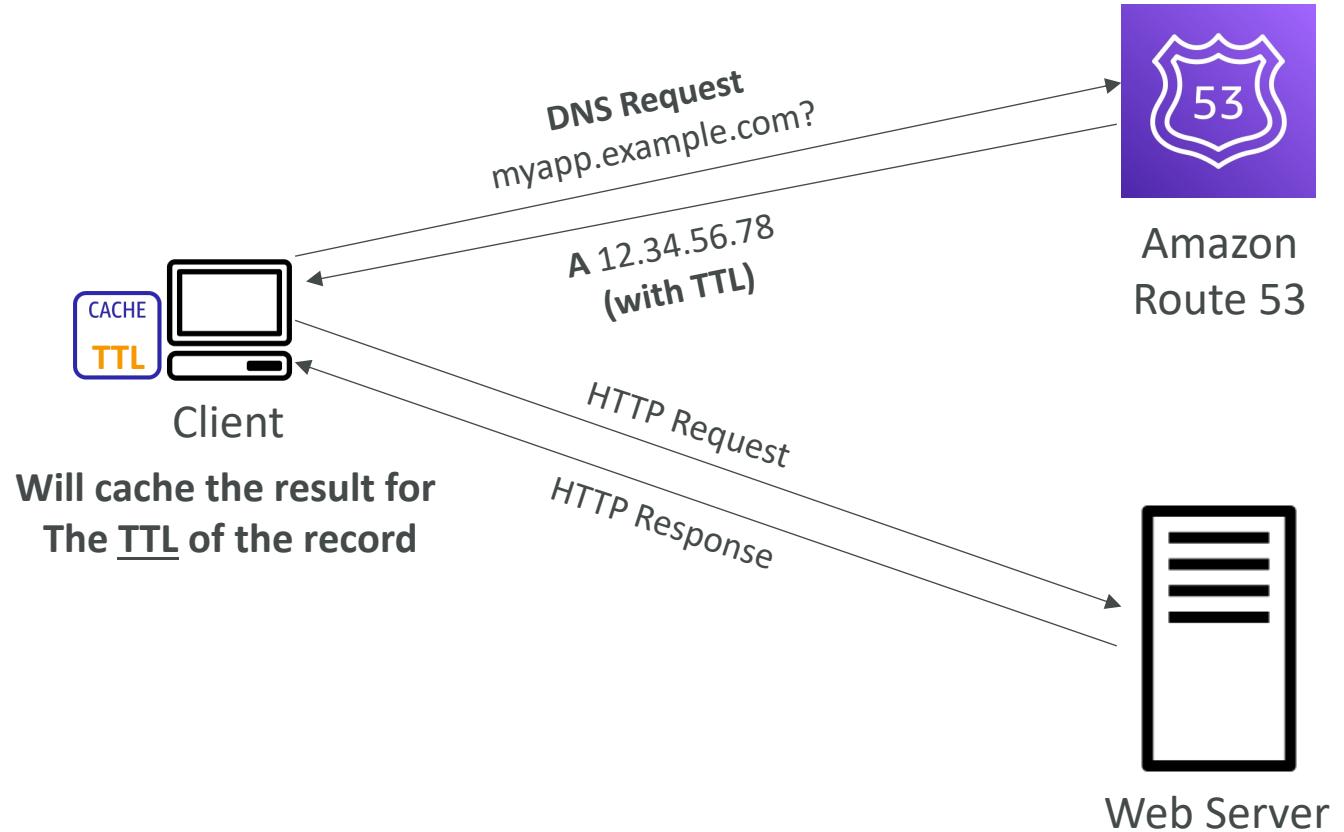
# Amazon SES Integration Flow in Route 53

1. Verify your domain in Amazon SES  
→ SES provides TXT record for verification.
2. Set up DKIM (recommended)  
→ SES provides 3 CNAME records for DKIM signatures.
3. Update Route 53 hosted zone  
→ Add SES verification + DKIM + SPF records.
4. Optional: Add MX record if receiving mail through SES.
5. Wait for propagation → SES status changes to “verified.”

Use the “Create Record in Route 53” button directly from SES → auto-creates verification, DKIM, and SPF records.

# Route 53 – Records TTL (Time To Live)

- High TTL – e.g., 24 hr
  - Less traffic on Route 53
  - Possibly outdated records
- Low TTL – e.g., 60 sec.
  - More traffic on Route 53 (\$\$)
  - Records are outdated for less time
  - Easy to change records
- Except for Alias records, TTL is mandatory for each DNS record



# CNAME vs Alias

- AWS Resources (Load Balancer, CloudFront...) expose an AWS hostname:
  - [lb-1234.us-east-2.elb.amazonaws.com](#) and you want [myapp.mydomain.com](#)
- CNAME:
  - Points a hostname to any other hostname. (app.mydomain.com => blabla.anything.com)
  - ONLY FOR NON ROOT DOMAIN (aka. something.mydomain.com)
- Alias:
  - Points a hostname to an AWS Resource (app.mydomain.com => blabla.amazonaws.com)
  - Works for ROOT DOMAIN and NON ROOT DOMAIN (aka mydomain.com)
  - Free of charge
  - Native health check

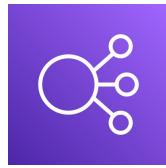
# Route 53 – Alias Records

- Maps a hostname to an AWS resource
- An extension to DNS functionality
- Automatically recognizes changes in the resource's IP addresses
- Unlike CNAME, it can be used for the top node of a DNS namespace (Zone Apex), e.g.: example.com
- Alias Record is always of type A/AAAA for AWS resources (IPv4 / IPv6)
- You can't set the TTL



# Route 53 – Alias Records Targets

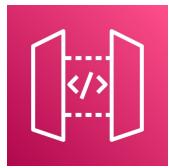
- Elastic Load Balancers
- CloudFront Distributions
- API Gateway
- Elastic Beanstalk environments
- S3 Websites
- VPC Interface Endpoints
- Global Accelerator accelerator
- Route 53 record in the same hosted zone
- You cannot set an ALIAS record for an EC2 DNS name



Elastic  
Load Balancer



Amazon  
CloudFront



Amazon  
API Gateway



Elastic Beanstalk



S3 Websites



VPC Interface  
Endpoints



Global Accelerator



Route 53 Record  
(same Hosted Zone)

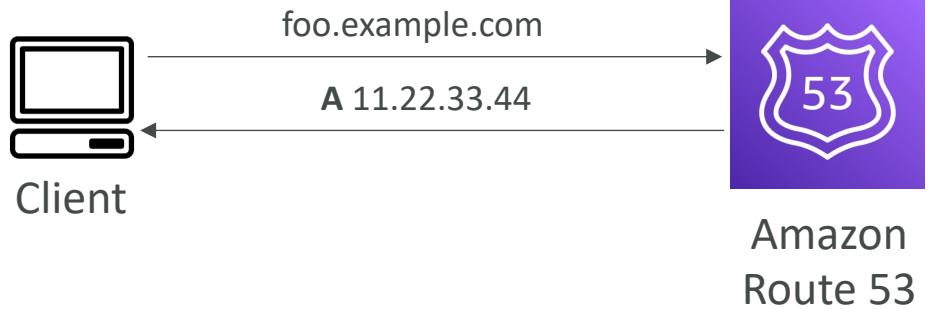
# Route 53 – Routing Policies

- Define how Route 53 responds to DNS queries
- Don't get confused by the word "Routing"
  - It's not the same as Load balancer routing which routes the traffic
  - DNS does not route any traffic, it only responds to the DNS queries
- Route 53 Supports the following Routing Policies
  - Simple
  - Weighted
  - Failover
  - Latency based
  - Geolocation
  - Multi-Value Answer
  - Geoproximity (using Route 53 Traffic Flow feature)

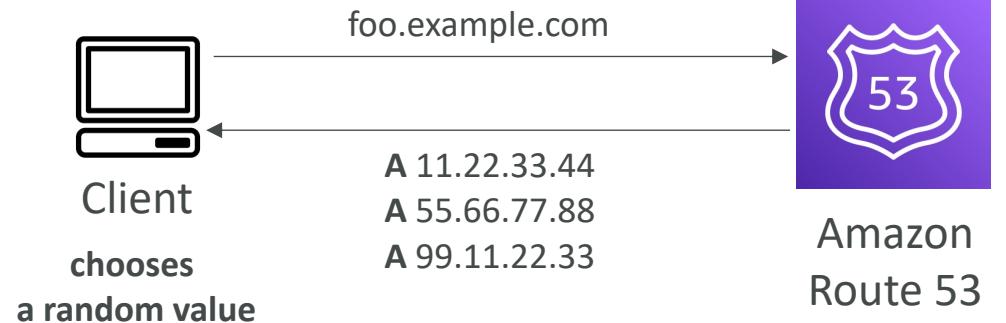
# Routing Policies – Simple

- Typically, route traffic to a single resource
- Can specify multiple values in the same record
- If multiple values are returned, a random one is chosen by the client
- When Alias enabled, specify only one AWS resource
- Can't be associated with Health Checks

## Single Value

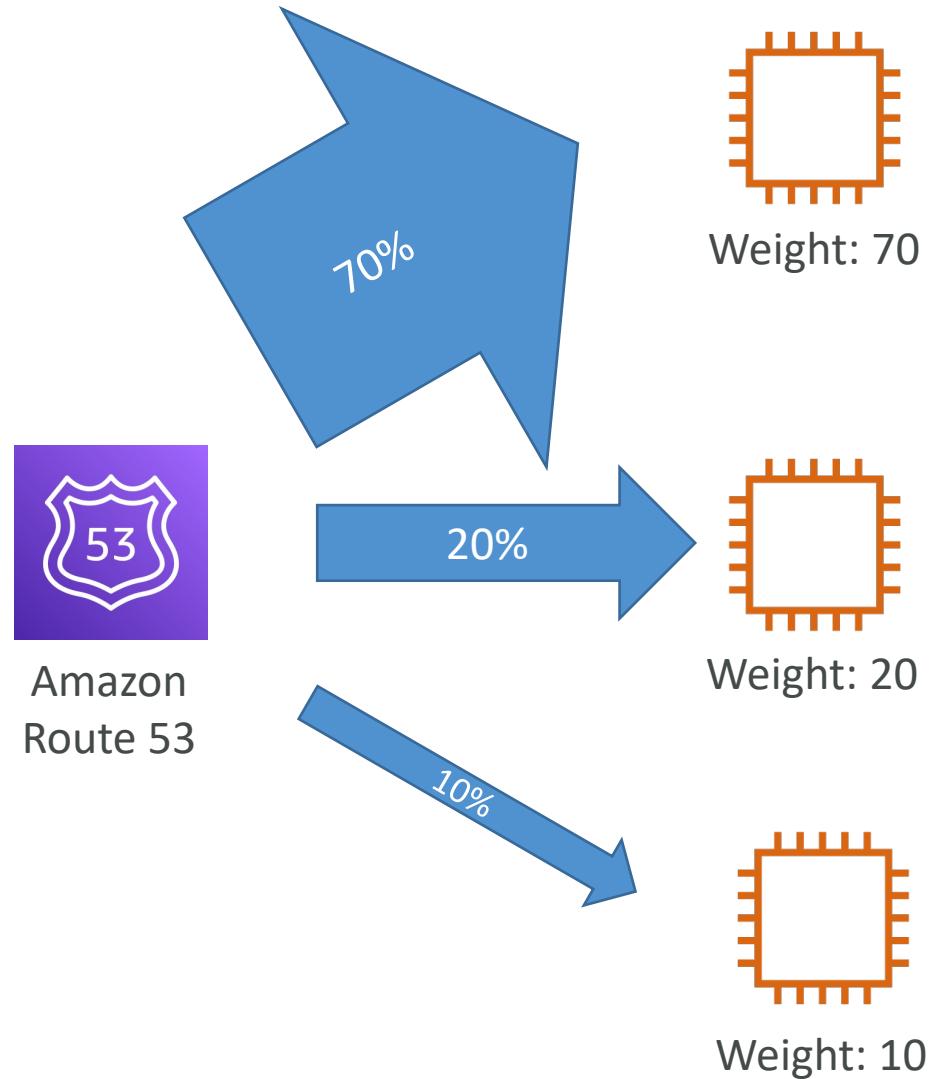


## Multiple Value



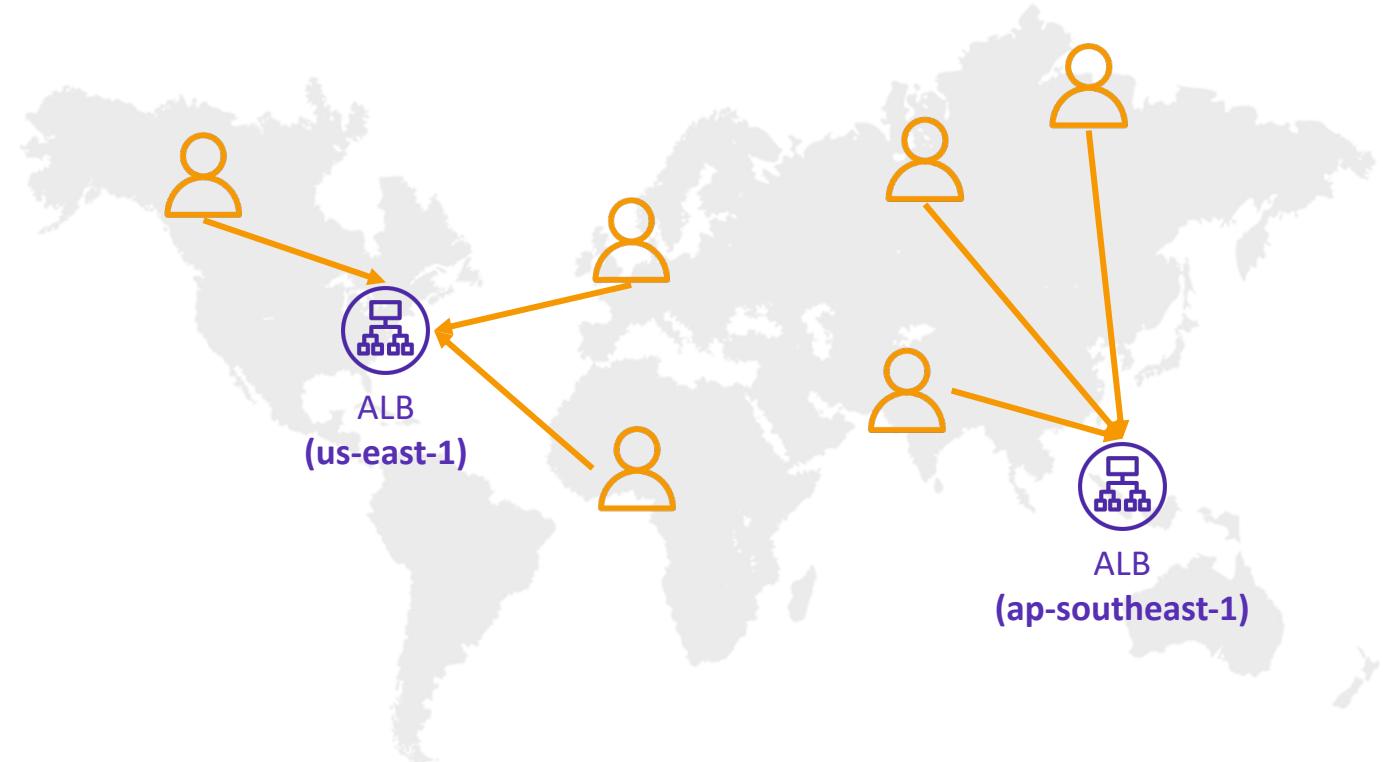
# Routing Policies – Weighted

- Control the % of the requests that go to each specific resource
- Assign each record a relative weight:
  - $traffic\ (%) = \frac{Weight\ for\ a\ specific\ record}{Sum\ of\ all\ the\ weights\ for\ all\ records}$
  - Weights don't need to sum up to 100
- DNS records must have the same name and type
- Can be associated with Health Checks
- Use cases: load balancing between regions, testing new application versions...
- Assign a weight of 0 to a record to stop sending traffic to a resource
- If all records have weight of 0, then all records will be returned equally



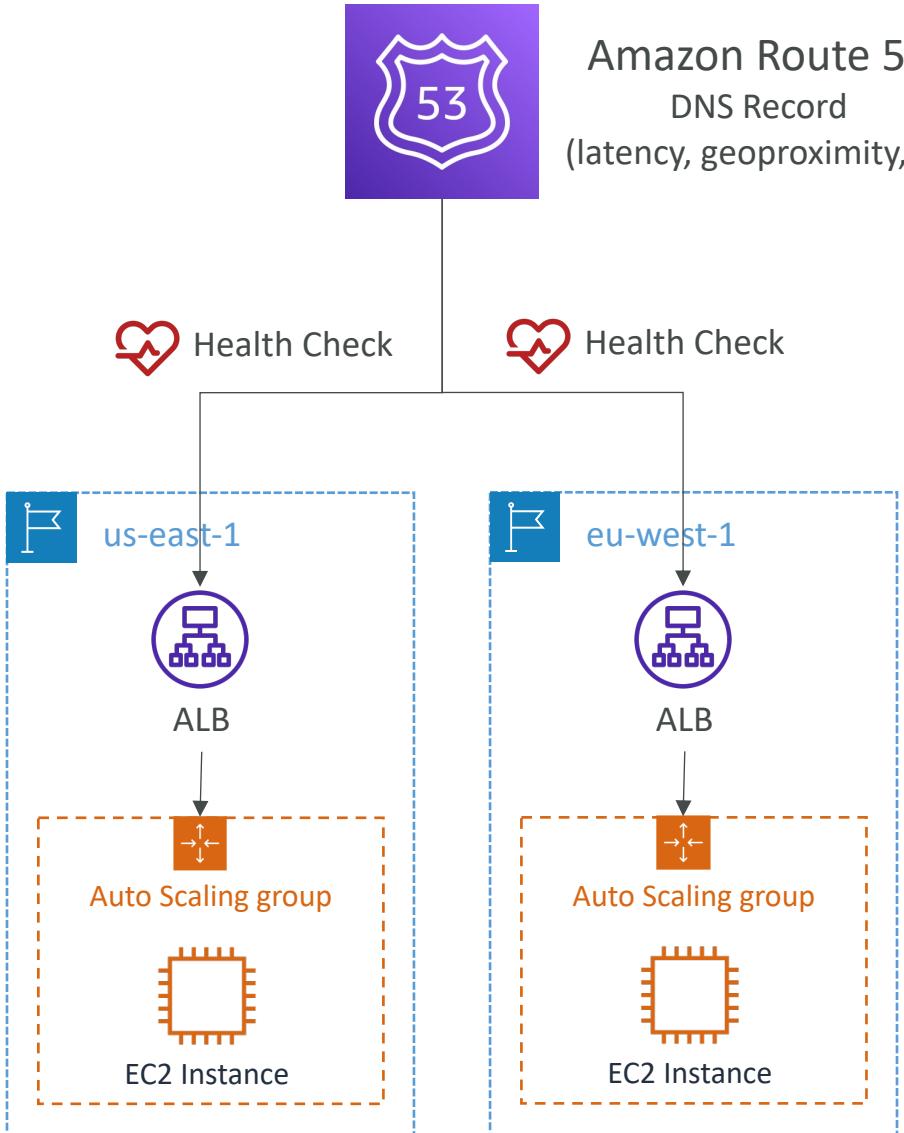
# Routing Policies – Latency-based

- Redirect to the resource that has the least latency close to us
- Super helpful when latency for users is a priority
- Latency is based on traffic between users and AWS Regions
- Germany users may be directed to the US (if that's the lowest latency)
- Can be associated with Health Checks (has a failover capability)



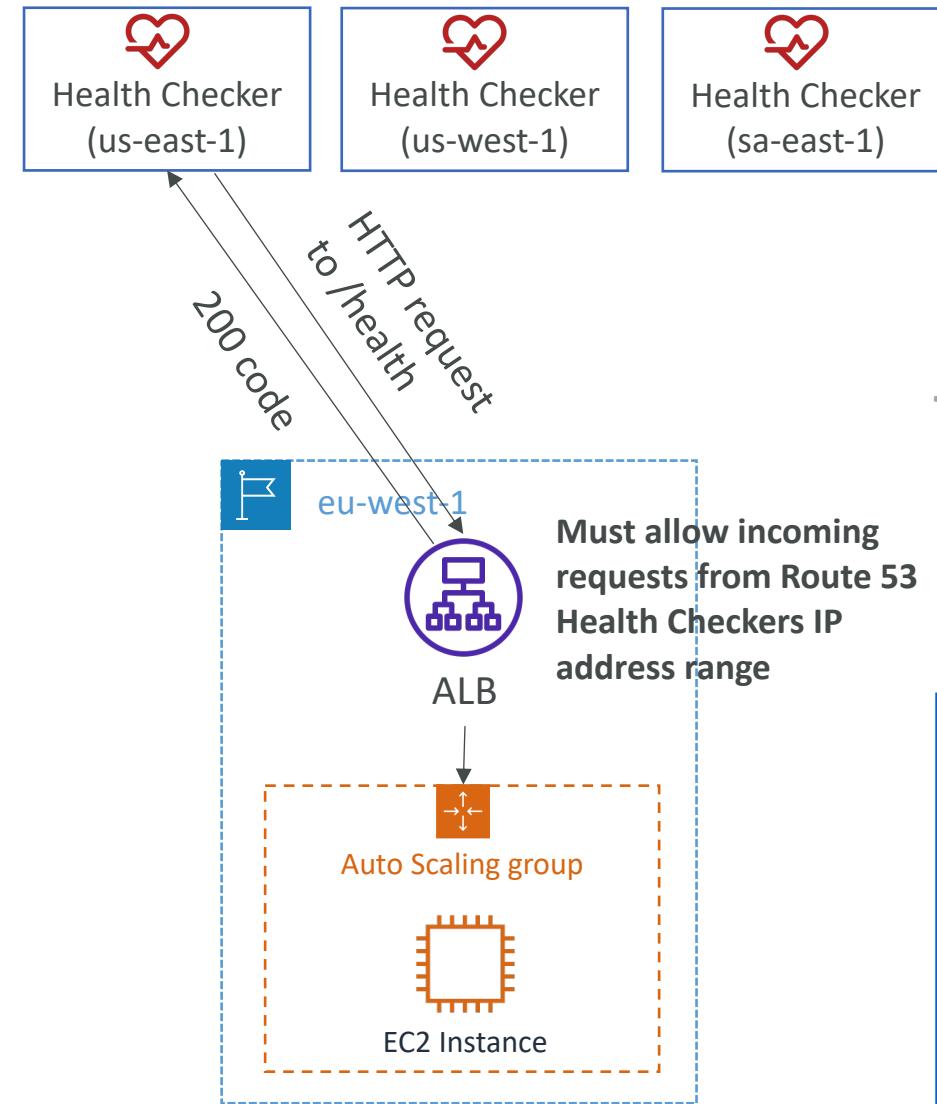
# Route 53 – Health Checks

- HTTP Health Checks are only for **public resources**
- Health Check => Automated DNS Failover:
  1. Health checks that monitor an endpoint (application, server, other AWS resource)
  2. Health checks that monitor other health checks (Calculated Health Checks)
  3. Health checks that monitor CloudWatch Alarms (full control !!) – e.g., throttles of DynamoDB, alarms on RDS, custom metrics, ... (helpful for private resources)
- Health Checks are integrated with CW metrics



# Health Checks – Monitor an Endpoint

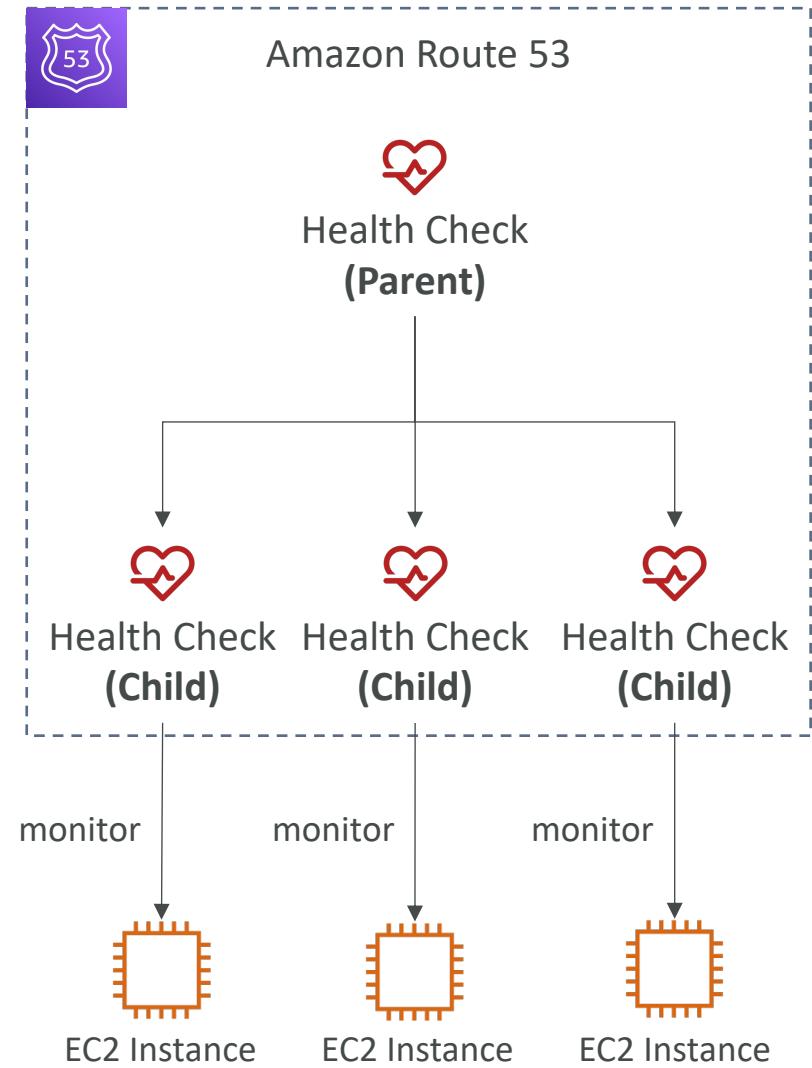
- About 15 global health checkers will check the endpoint health
  - Healthy/Unhealthy Threshold – 3 (default)
  - Interval – 30 sec (can set to 10 sec – higher cost)
  - Supported protocol: HTTP, HTTPS and TCP
  - If > 18% of health checkers report the endpoint is healthy, Route 53 considers it **Healthy**. Otherwise, it's **Unhealthy**
  - Ability to choose which locations you want Route 53 to use
- Health Checks pass only when the endpoint responds with the 2xx and 3xx status codes
- Health Checks can be setup to pass / fail based on the text in the first **5120 bytes** of the response
- Configure your router/firewall to allow incoming requests from Route 53 Health Checkers



<https://ip-ranges.amazonaws.com/ip-ranges.json>

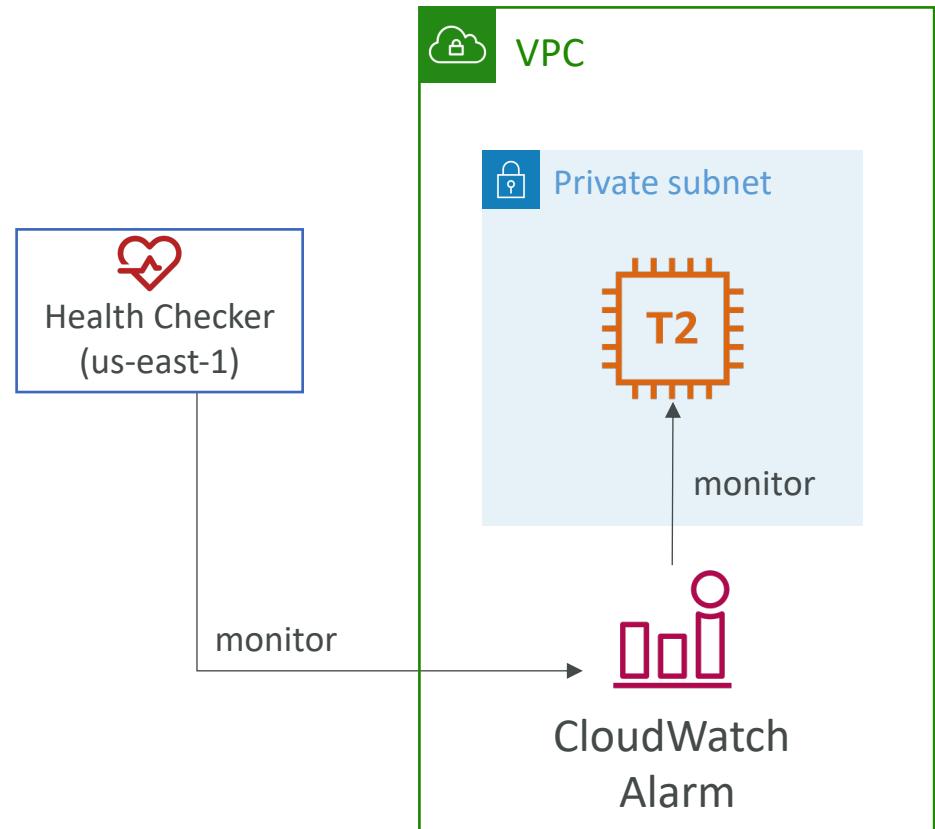
# Route 53 – Calculated Health Checks

- Combine the results of multiple Health Checks into a single Health Check
- You can use **OR**, **AND**, or **NOT**
- Can monitor up to 256 Child Health Checks
- Specify how many of the health checks need to pass to make the parent pass
- Usage: perform maintenance to your website without causing all health checks to fail

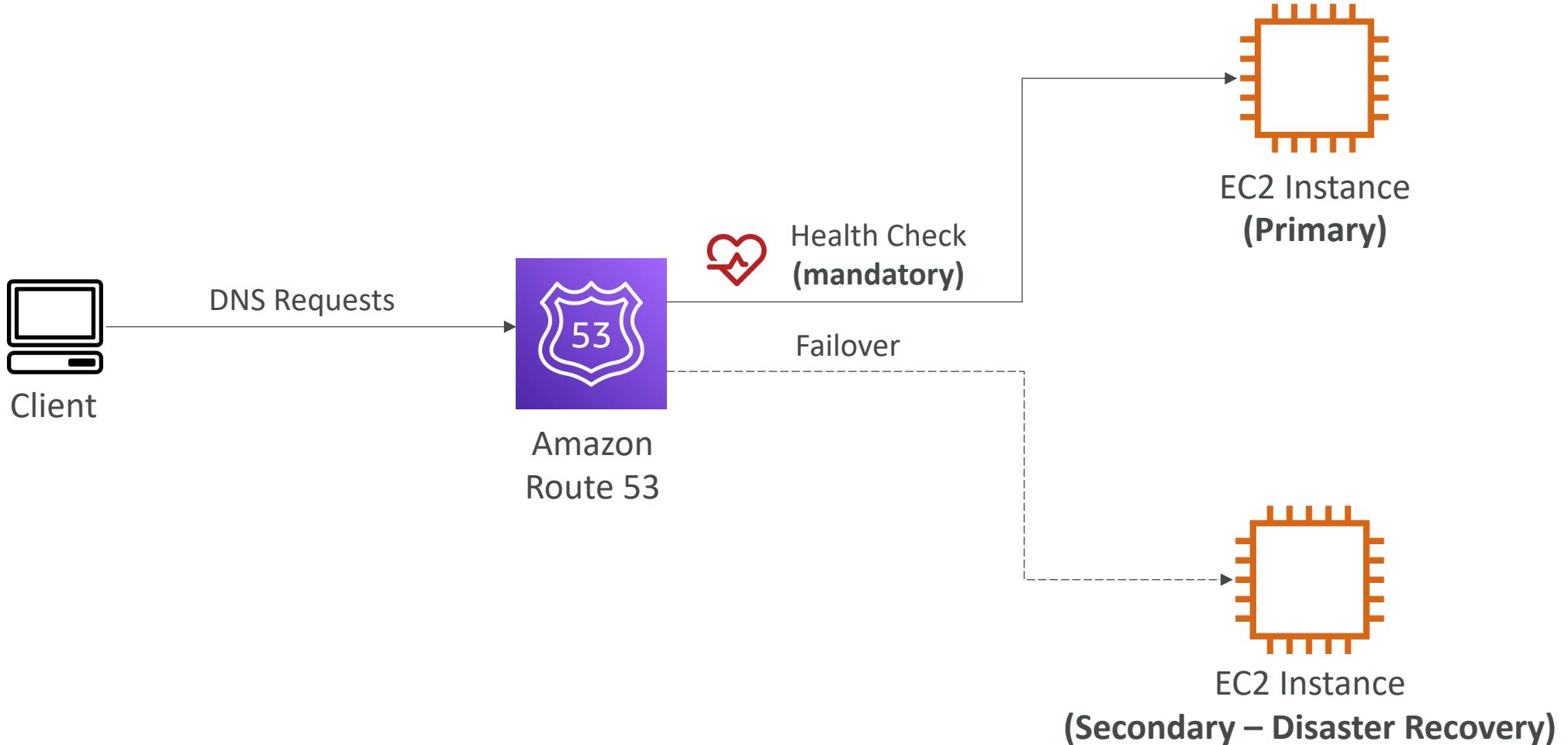


# Health Checks – Private Hosted Zones

- Route 53 health checkers are outside the VPC
- They can't access **private** endpoints (private VPC or on-premises resource)
- You can create a CloudWatch Metric and associate a CloudWatch Alarm, then create a Health Check that checks the alarm itself

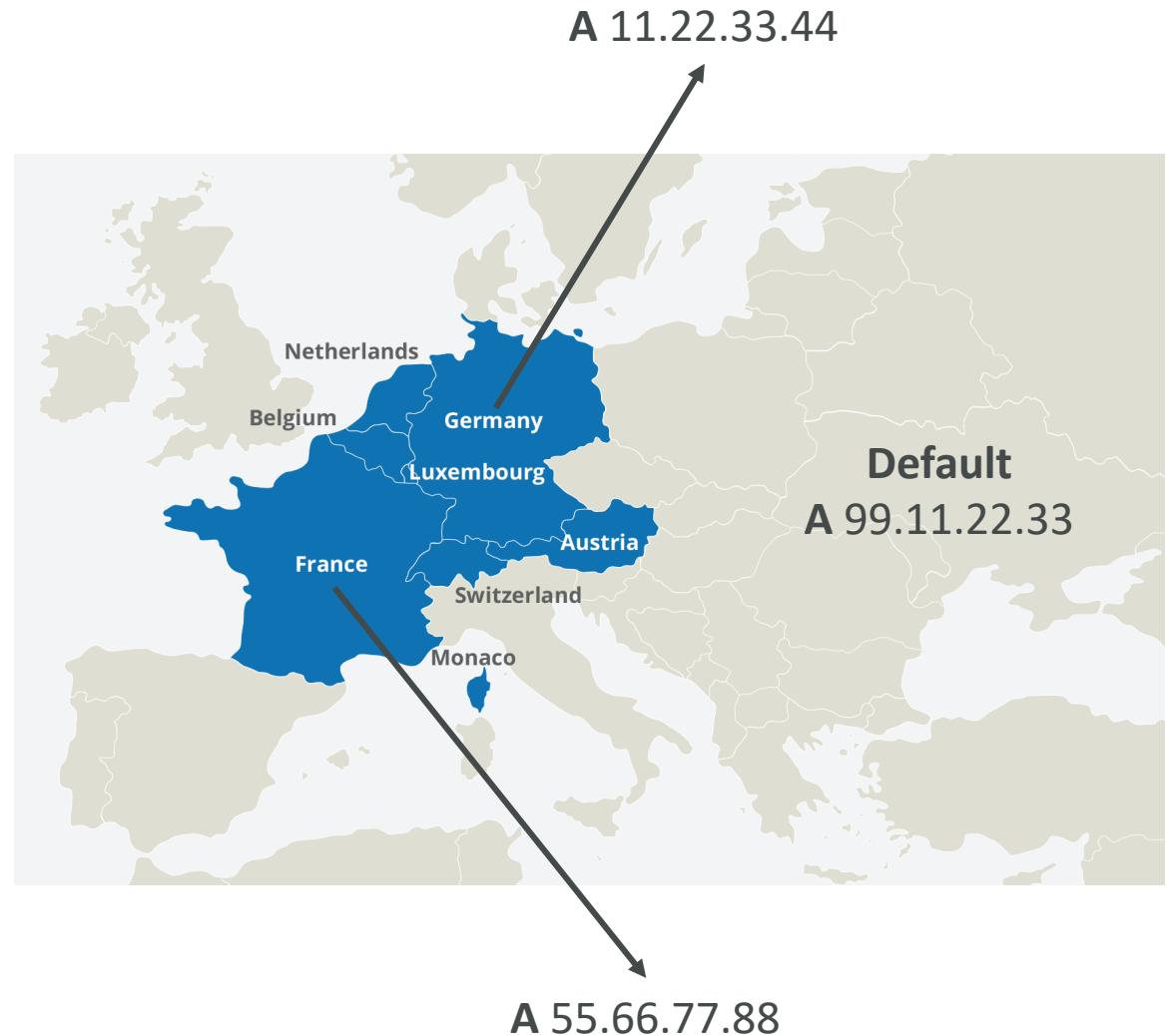


# Routing Policies – Failover (Active-Passive)



# Routing Policies – Geolocation

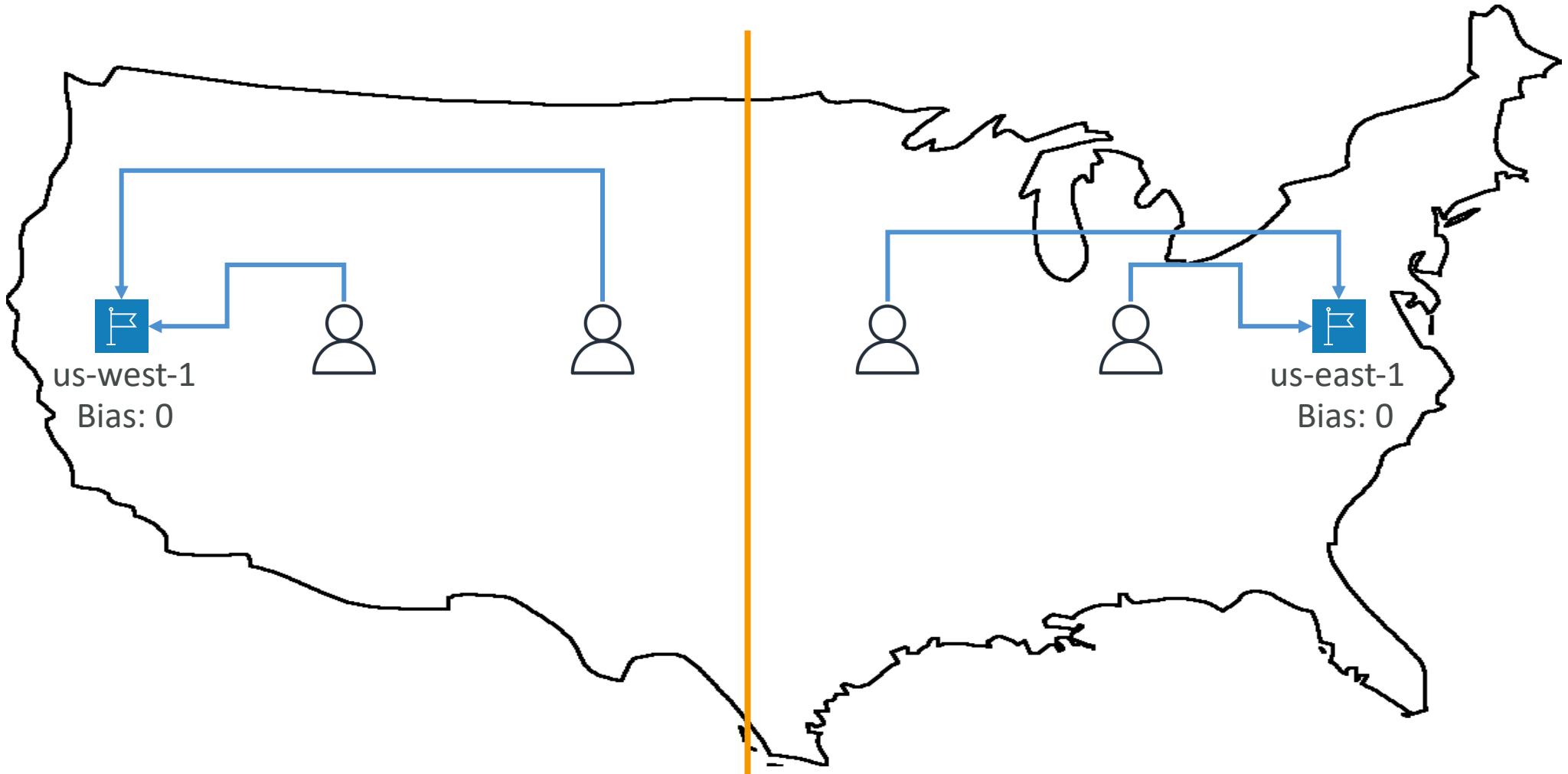
- Different from Latency-based!
- This routing is based on user location
- Specify location by Continent, Country or by US State (if there's overlapping, most precise location selected)
- Should create a “Default” record (in case there's no match on location)
- Use cases: website localization, restrict content distribution, load balancing, ...
- Can be associated with Health Checks



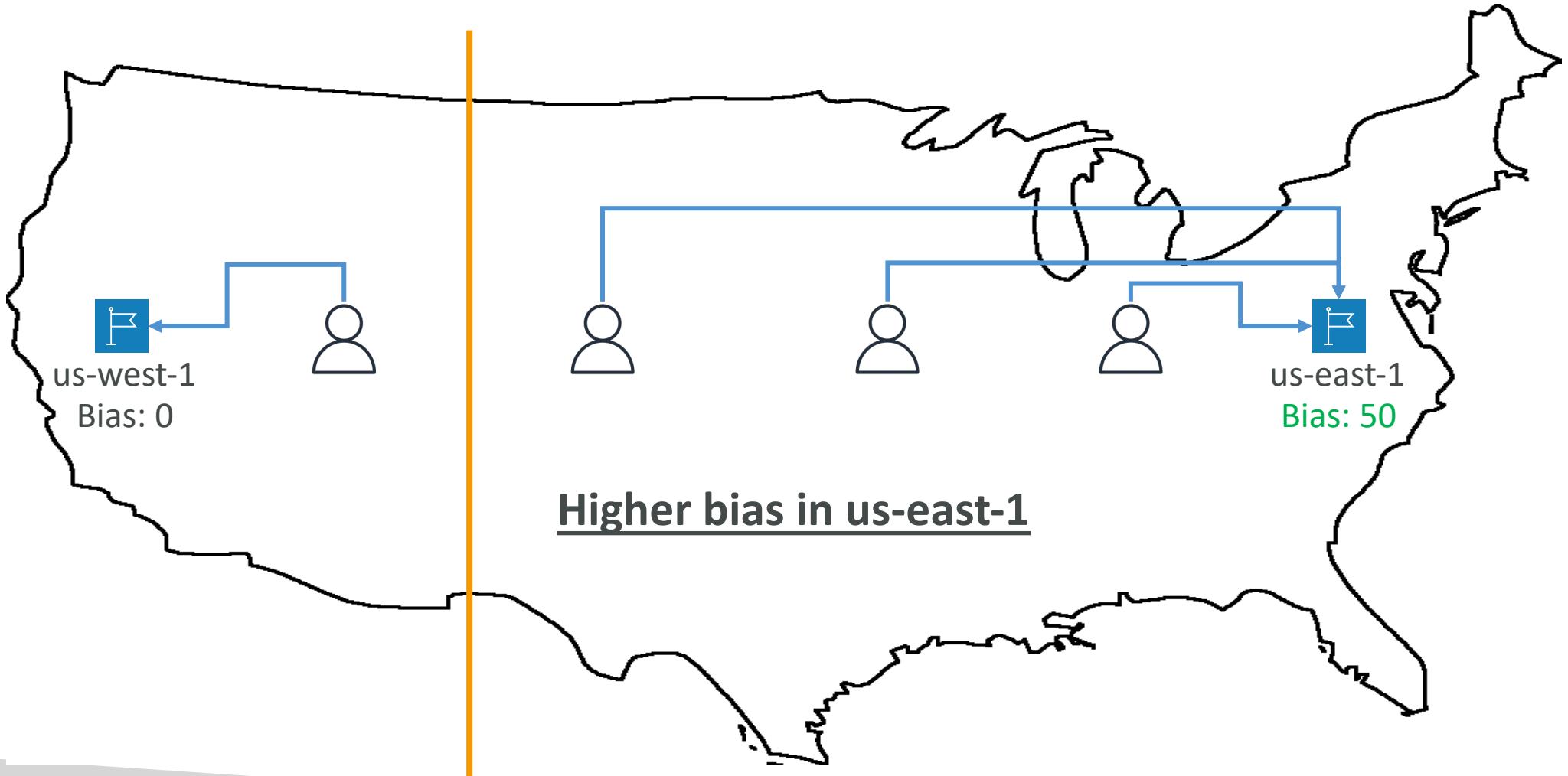
# Routing Policies – Geoproximity

- Route traffic to your resources based on the geographic location of users and resources
- Ability **to shift more traffic to resources based** on the defined bias
- To change the size of the geographic region, specify **bias** values:
  - To expand (1 to 99) – more traffic to the resource
  - To shrink (-1 to -99) – less traffic to the resource
- Resources can be:
  - AWS resources (specify AWS region)
  - Non-AWS resources (specify Latitude and Longitude)
- You must use Route 53 Traffic Flow to use this feature

# Routing Policies – Geoproximity

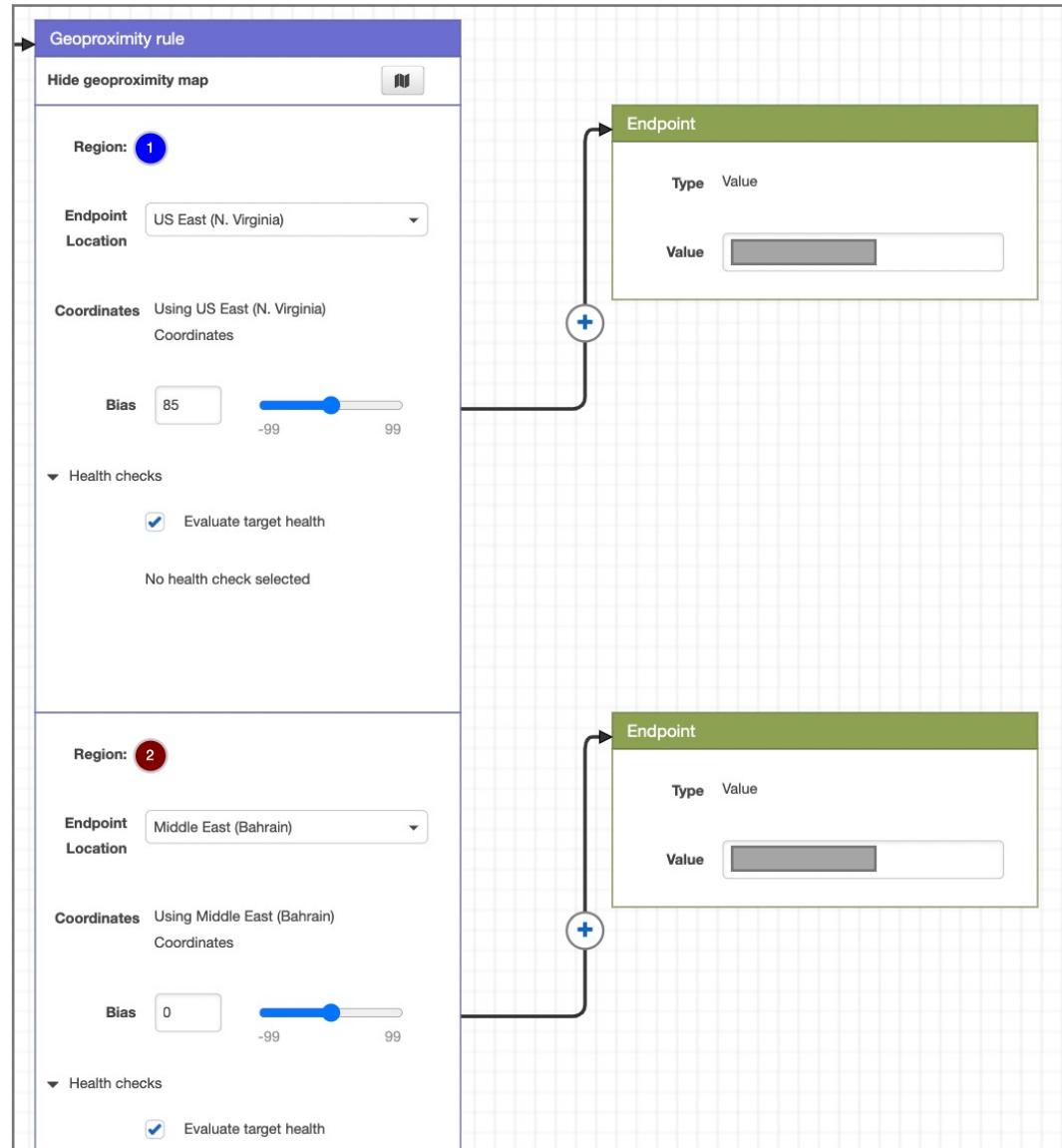


# Routing Policies – Geoproximity



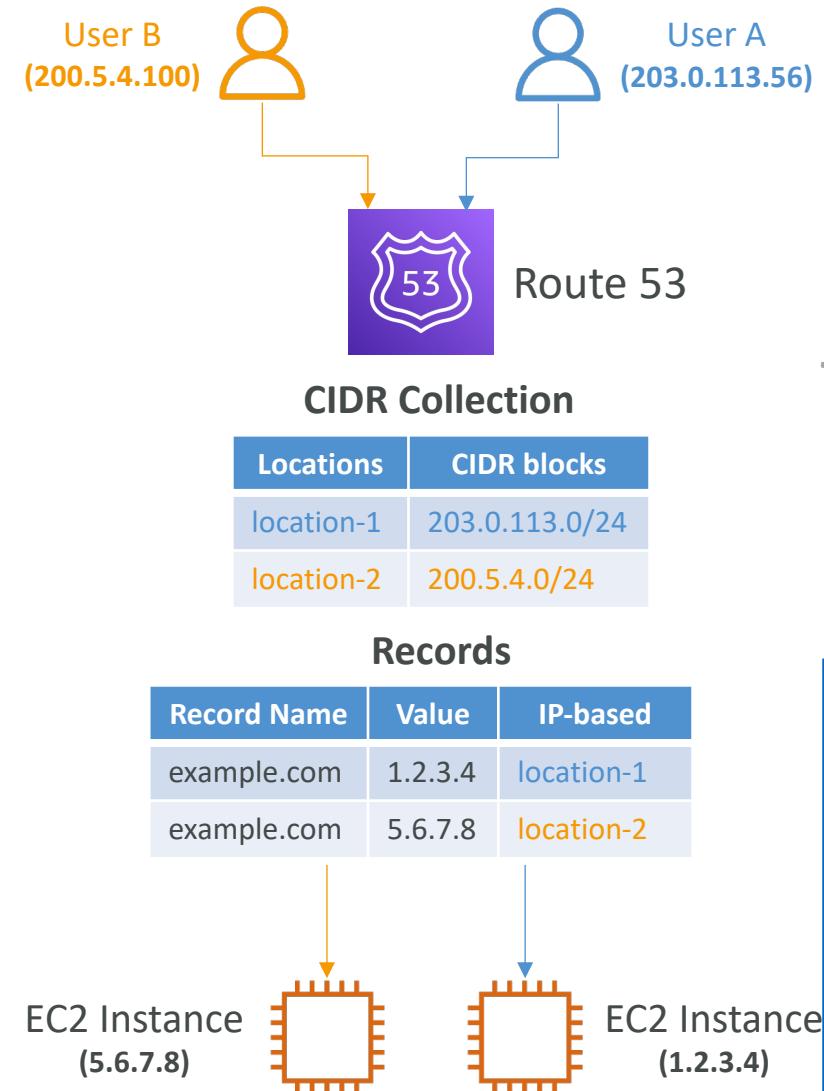
# Route 53 – Traffic flow

- Simplify the process of creating and maintaining records in large and complex configurations
- Visual editor to manage complex routing decision trees
- Configurations can be saved as **Traffic Flow Policy**
  - Can be applied to different Route 53 Hosted Zones (different domain names)
  - Supports versioning



# Routing Policies – IP-based Routing

- Routing is based on clients' IP addresses
- You provide a list of CIDRs for your clients and the corresponding endpoints/locations (user-IP-to-endpoint mappings)
- Use cases: Optimize performance, reduce network costs...
- Example: route end users from a particular ISP to a specific endpoint



# Routing Policies – Multi-Value

- Use when routing traffic to multiple resources
- Route 53 return multiple values/resources
- Can be associated with Health Checks (return only values for healthy resources)
- Up to 8 healthy records are returned for each Multi-Value query
- Multi-Value is not a substitute for having an ELB

Name	Type	Value	TTL	Set ID	Health Check
www.example.com	A Record	192.0.2.2	60	Web1	A
www.example.com	A Record	198.51.100.2	60	Web2	B
www.example.com	A Record	203.0.113.2	60	Web3	C

# Domain Registrar vs. DNS Service

- You buy or register your domain name with a Domain Registrar typically by paying annual charges (e.g., GoDaddy, Amazon Registrar Inc., ...)
- The Domain Registrar usually provides you with a DNS service to manage your DNS records
- But you can use another DNS service to manage your DNS records
- Example: purchase the domain from GoDaddy and use Route 53 to manage your DNS records



# GoDaddy as Registrar & Route 53 as DNS Service



## Records

We can't display your DNS information because your nameservers aren't managed by us.

## Nameservers

Using custom nameservers [Change](#)

Nameserver
ns-1083.awsdns-07.org
ns-932.awsdns-52.net
ns-1911.awsdns-46.co.uk
ns-481.awsdns-60.com



Amazon  
Route 53

**Public Hosted Zone**  
stephanetheteacher.com

▼ Hosted zone details [Edit hosted zone](#)

Hosted zone ID	Type	Name servers
Z30IJCCWPKUV	Public hosted zone	ns-252.awsdns-31.com ns-1468.awsdns-55.org ns-633.awsdns-15.net ns-1800.awsdns-33.co.uk
Description	Record count	
HostedZone created by Route53 Registrar	22	
Query log		

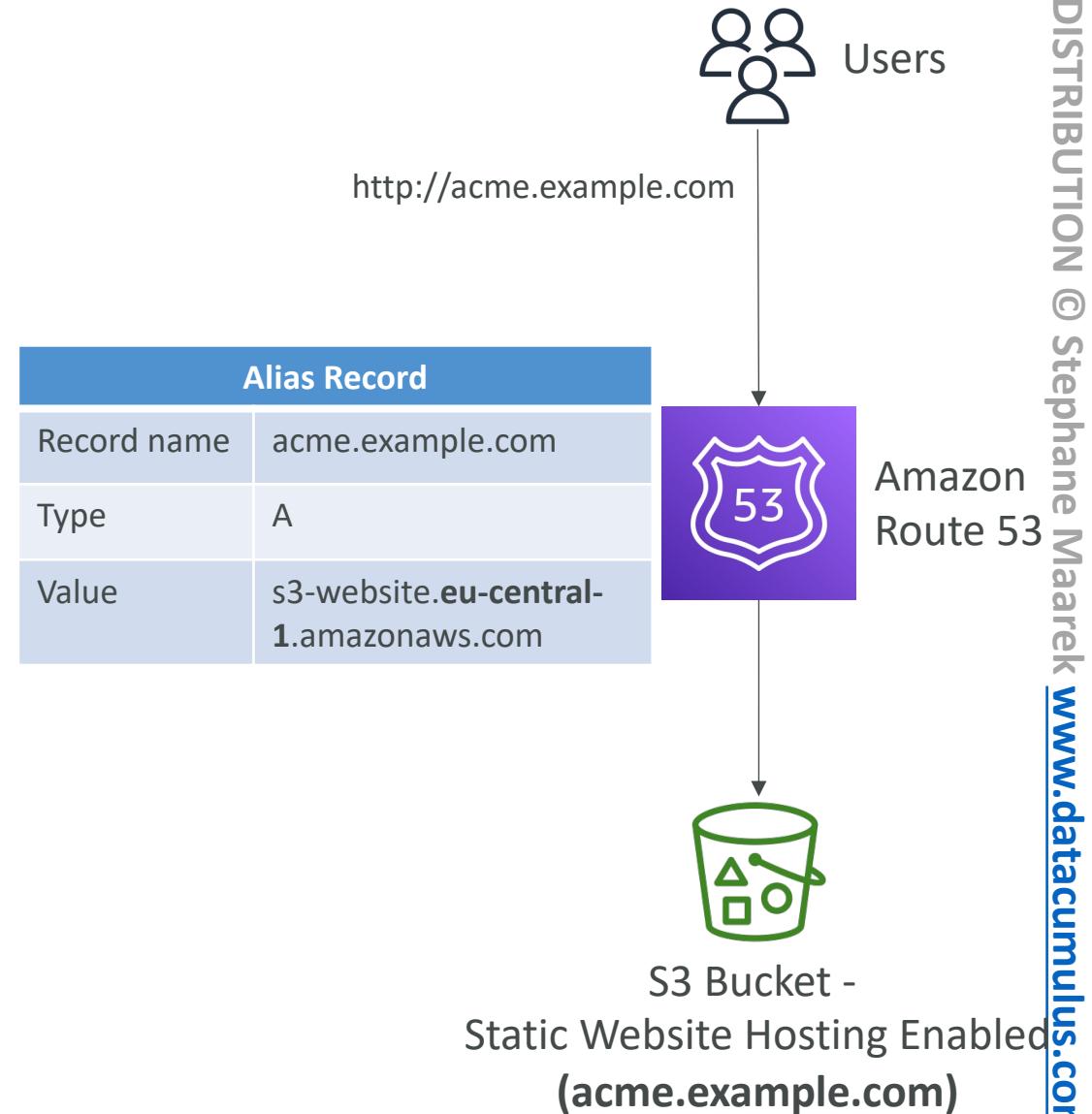
# 3<sup>rd</sup> Party Registrar with Amazon Route 53

- If you buy your domain on a 3<sup>rd</sup> party registrar, you can still use Route 53 as the DNS Service provider
  - 1. Create a Hosted Zone in Route 53
  - 2. Update NS Records on 3<sup>rd</sup> party website to use Route 53 Name Servers
- Domain Registrar != DNS Service
- But every Domain Registrar usually comes with some DNS features

# S3 Website with Route 53

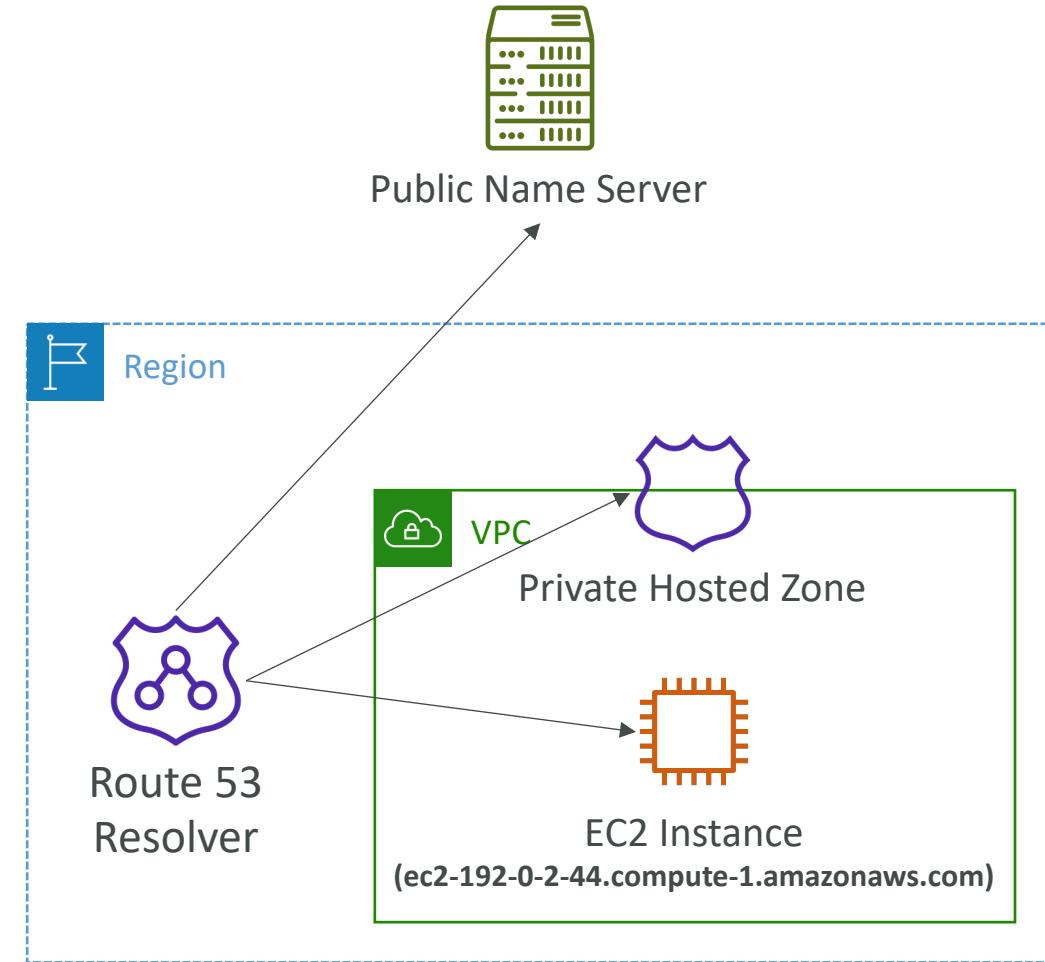
For acme.example.com:

- Create an S3 bucket with the **same name** as the target record (acme.example.com)
- Enable S3 website on the bucket (and enable S3 bucket public settings)
- Create a Route 53 **Alias record** to the S3 website endpoint or type A – IPv4 address
- This only works for HTTP traffic (for HTTPS, use CloudFront)



# Route 53 – Hybrid DNS

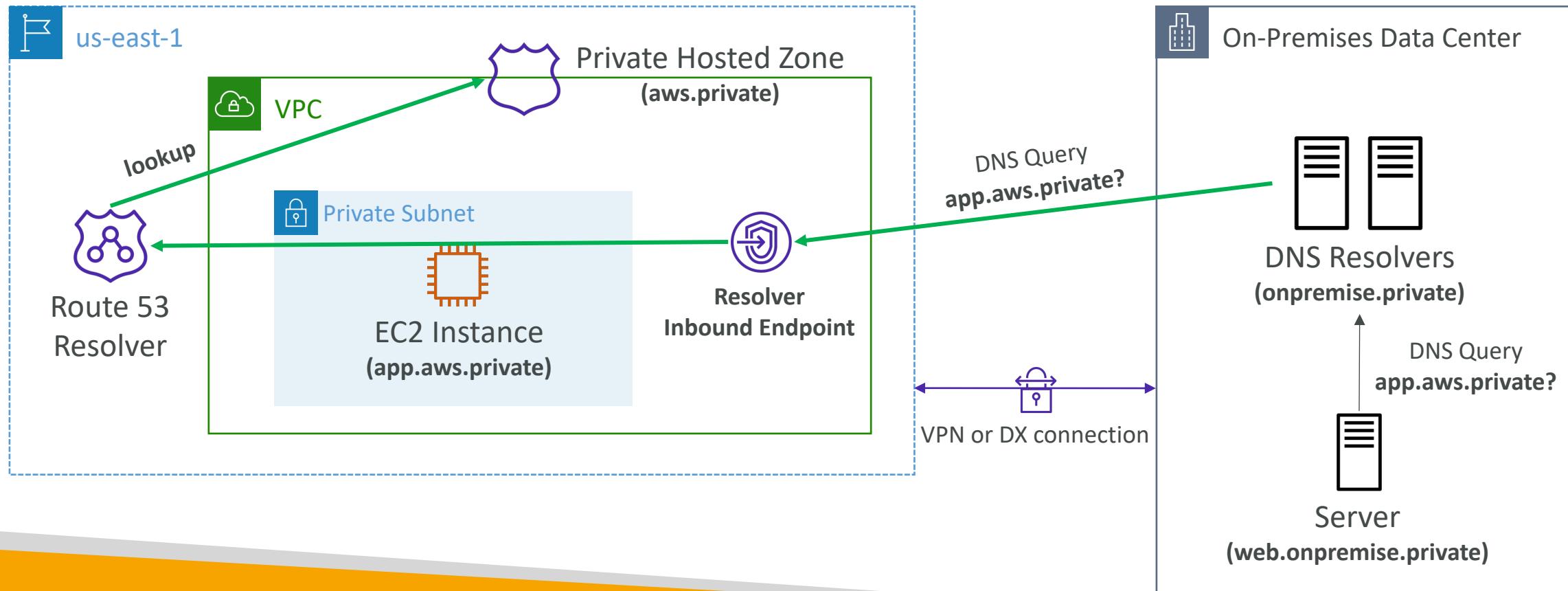
- By default, Route 53 Resolver automatically answers DNS queries for:
  - Local domain names for EC2 instances
  - Records in Private Hosted Zones
  - Records in public Name Servers
- **Hybrid DNS** – resolving DNS queries between VPC (Route 53 Resolver) and your networks (other DNS Resolvers)
- Networks can be:
  - VPC itself / Peered VPC
  - On-premises Network (connected through Direct Connect or AWS VPN)





# Route 53 – Resolver Endpoints

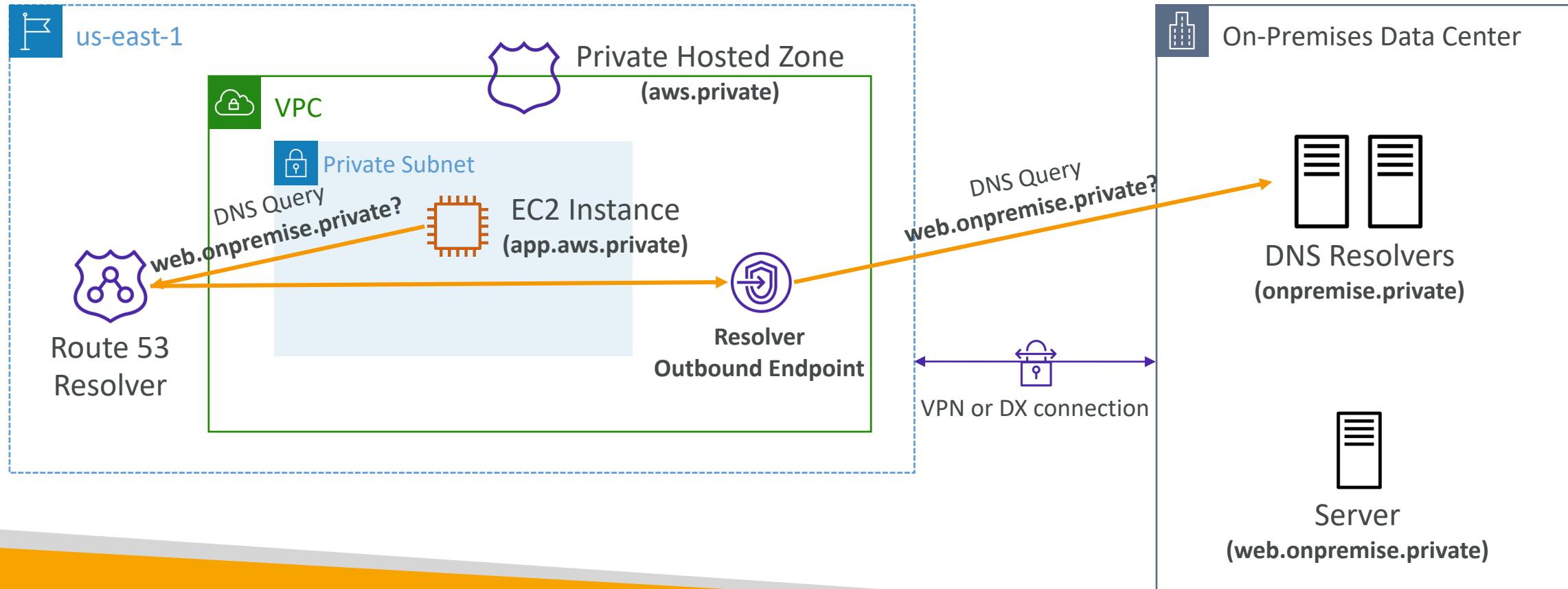
- **Inbound Endpoint** – allows your DNS Resolvers to resolve domain names for AWS resources (e.g., EC2 instances) and records in Private Hosted Zones





# Route 53 – Resolver Endpoints

- Outbound Endpoint
  - Route 53 Resolver forwards DNS queries to your DNS Resolvers



# Route 53 – DNS Query Logging

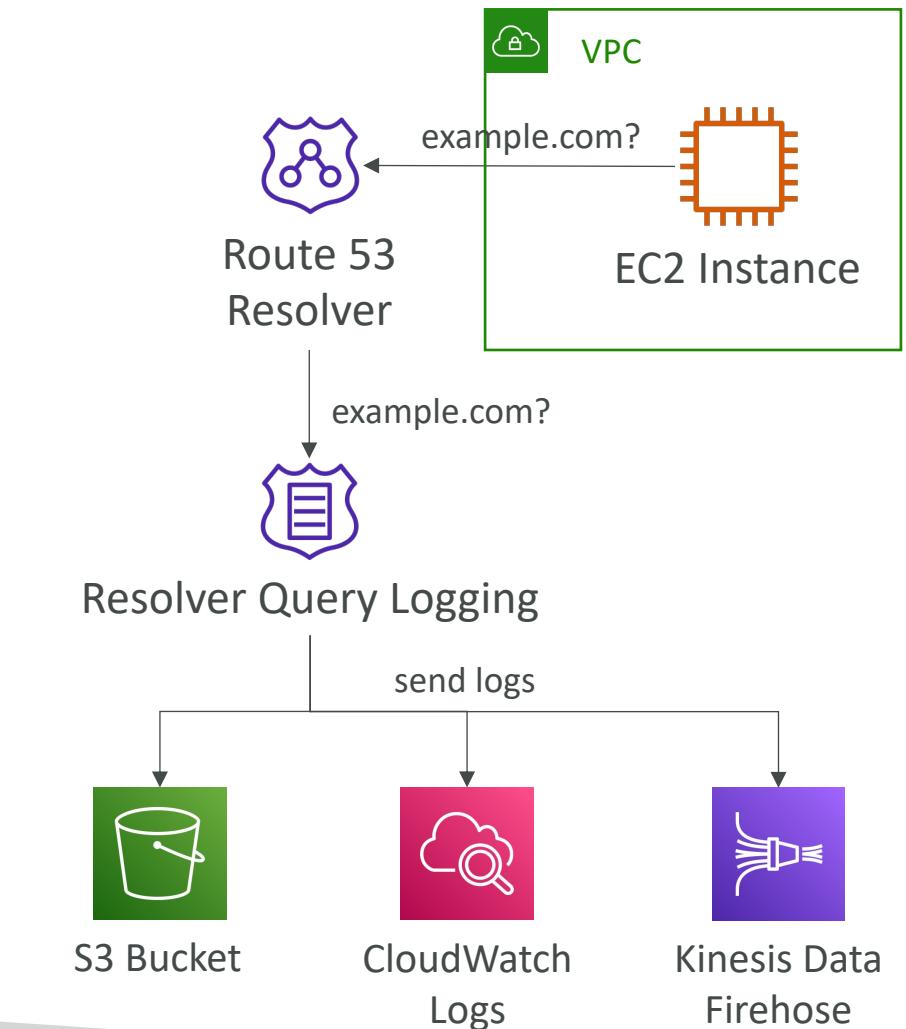
- Log information about public DNS queries Route 53 Resolver receives
- Only for Public Hosted Zones
- Can send logs to CloudWatch Logs (can export to S3)

Log Format	Version	Hosted Zone ID	Query Type	Query Protocol	Resolver IP Address
1.0 2017-12-13T08:16:02.130Z	Z123412341234	example.com	A	NOERROR	UDP FRA6 192.168.1.1 -
1.0 2017-12-13T08:15:50.235Z	Z123412341234	example.com	AAAA	NOERROR	TCP IAD12 192.168.3.1 192.168.222.0/24
1.0 2017-12-13T08:16:03.983Z	Z123412341234	example.com	ANY	NOERROR	UDP FRA6 2001:db8::1234 2001:db8:abcd::/48
1.0 2017-12-13T08:15:50.342Z	Z123412341234	bad.example.com	A	NXDOMAIN	UDP IAD12 192.168.3.1 192.168.111.0/24
1.0 2017-12-13T08:16:05.744Z	Z123412341234	txt.example.com	TXT	NOERROR	UDP JFK5 192.168.1.2 -

# Route 53 – Resolver Query Logging



- Logs all DNS queries made by resources within a VPC
  - Private Hosted Zones
  - Resolver Inbound & Outbound Endpoints
  - Resolver DNS Firewall
- Can send logs to CloudWatch Logs, S3 bucket, or Kinesis Data Firehose
- Configurations can be shared with other AWS Accounts using AWS Resource Access Manager (AWS RAM)





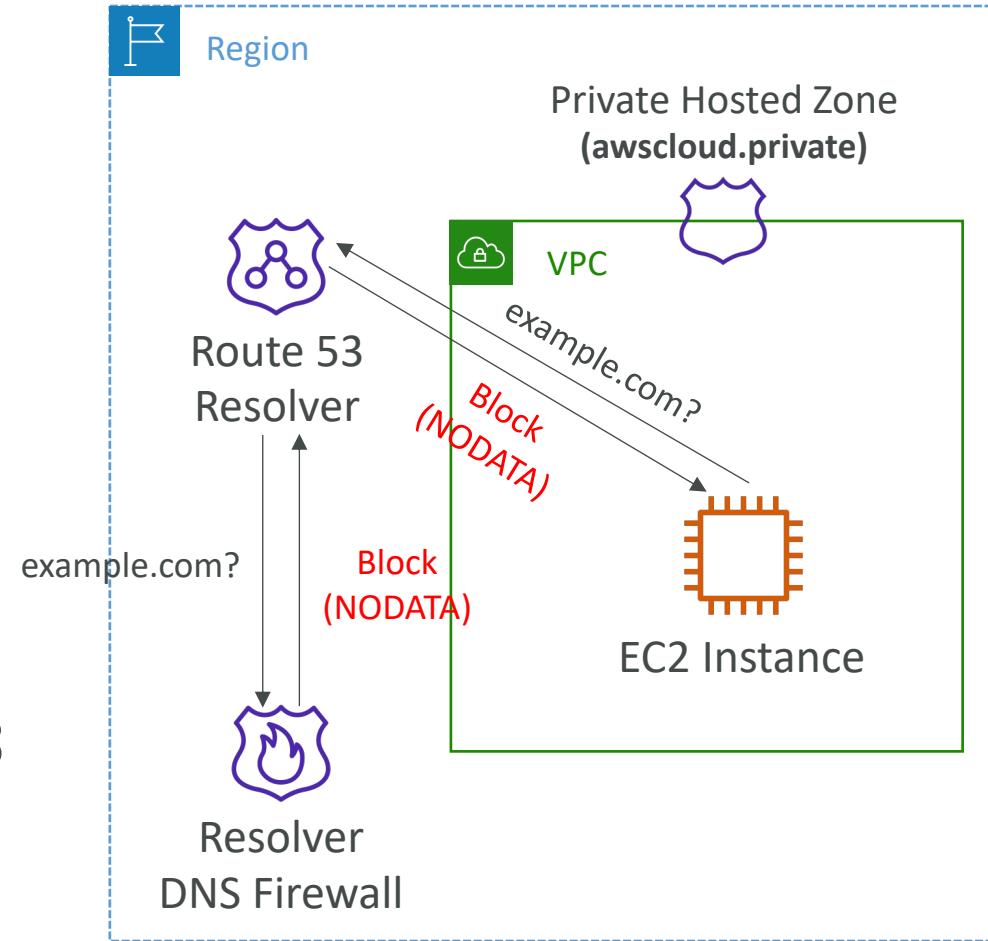
# Route 53 – Resolver Query Logging

```
{  
    "srcaddr": "4.5.64.102",  
    "vpc_id": "vpc-7example",  
    "answers": [  
        {  
            "Rdata": "203.0.113.9",  
            "Type": "PTR",  
            "Class": "IN"  
        }  
    ],  
    "firewall_rule_group_id": "rslvr-frg-01234567890abcdef",  
    "firewall_rule_action": "BLOCK",  
    "query_name": "15.3.4.32.in-addr.arpa.",  
    "firewall_domain_list_id": "rslvr-fdl-01234567890abcdef",  
    "query_class": "IN",  
    "srcids": {  
        "instance": "i-0d15cd0d3example"  
    },  
    "rcode": "NOERROR",  
    "query_type": "PTR",  
    "transport": "UDP",  
    "version": "1.100000",  
    "account_id": "111122223333",  
    "srcport": "56067",  
    "query_timestamp": "2021-02-04T17:51:55Z",  
    "region": "us-east-1"  
}
```



# Route 53 – Resolver DNS Firewall

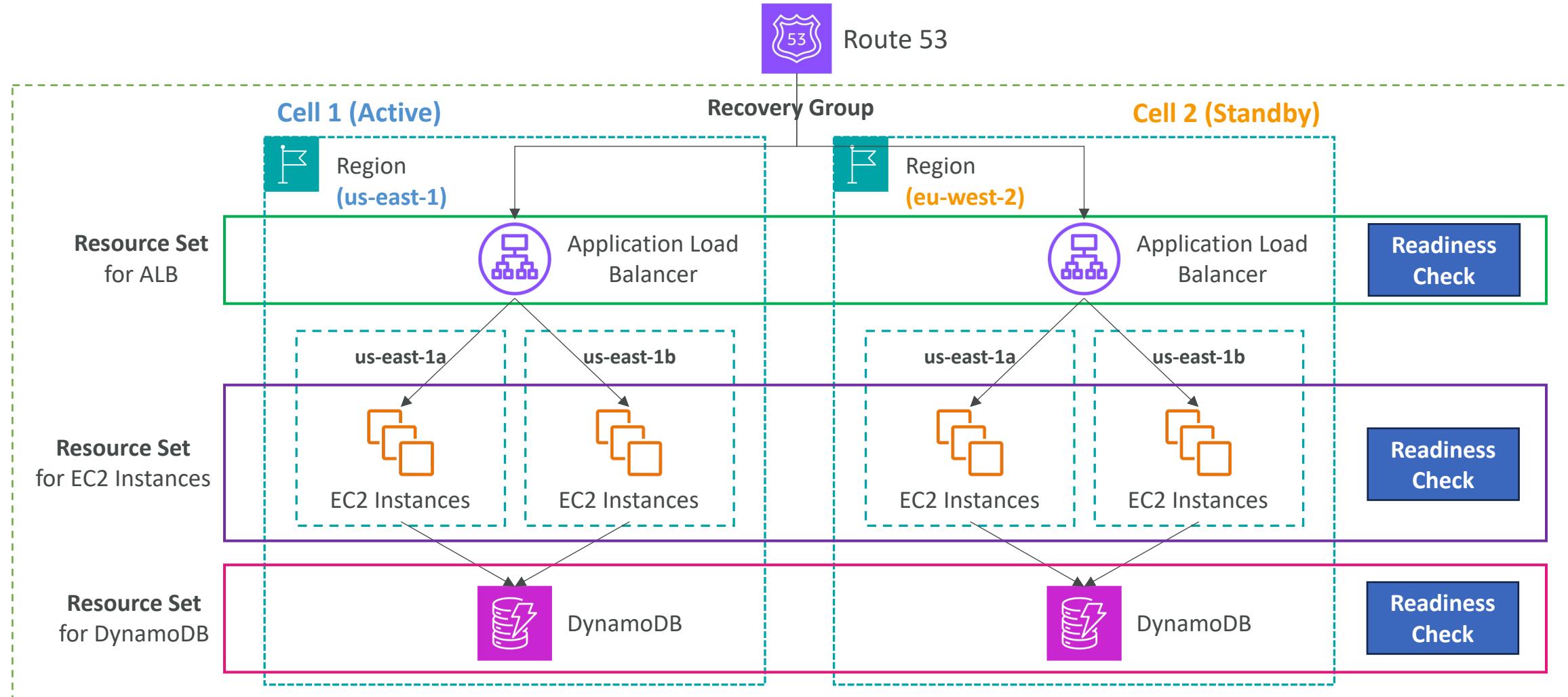
- A managed firewall enables you to filter outbound DNS requests going out through Route 53 Resolver
- Blacklist malicious domains or Whitelist trusted domains
- Example: prevent a compromised application within your VPC to send data out through DNS to a malicious domain (**DNS exfiltration**)
- Can be managed/configured from AWS Firewall Manager
- Send logs to CloudWatch Logs and Route 53 Resolver Query Logs



# Route 53 – Application Recovery Controller

- Automate and orchestrate application recovery across AZs and Regions
- Mission-critical applications needing *high availability* and *minimal downtime*
- Multi-Region active/standby or active/active setups
- Environments with regulatory or business continuity requirements
- Readiness checks – continuously validate standby/replica infra is ready
- Routing controls + DNS integration – turn “traffic switches” on/off to reroute users between AZs/Regions via DNS & health-checks
- Zonal Shift / Region Switch – shift traffic away from impaired AZs or orchestrate full Region-level failover

# Route 53 – Application Recovery Controller

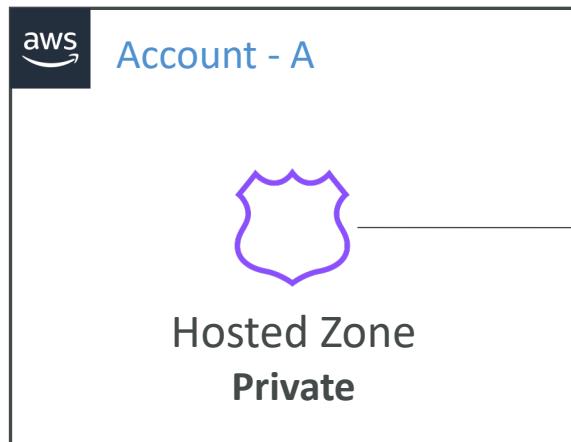


# Route 53 – Profiles

- Centrally manage Route 53 configuration across many VPCs and AWS accounts
- Helps you standardize DNS configuration across your VPCs
- Supports resources:
  - Private Hosted Zones
  - Route 53 Resolver Rules
  - Resolver DNS Firewall Rule Groups
  - VPC Interface Endpoints
- Route 53 Profiles can be shared across account using AWS RAM

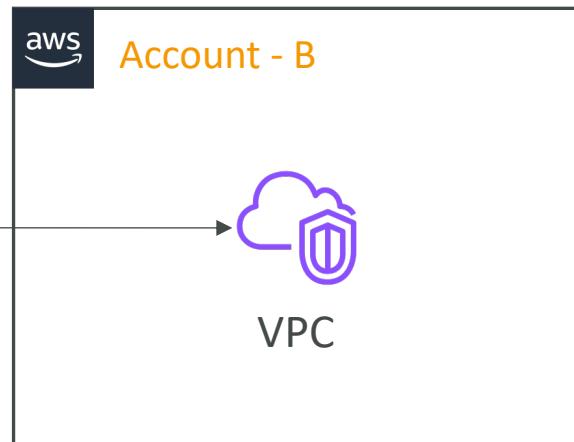
# Route 53 – Cross-Account Private Hosted Zone

- You can associate Private Hosted Zone with a VPC in another Account



1. Create VPC Association Authorization

```
aws route53 create-vpc-association-authorization --hosted-zone-id hosted-zone-id --vpc VPCRegion=region,VPCId=vpc-id
```

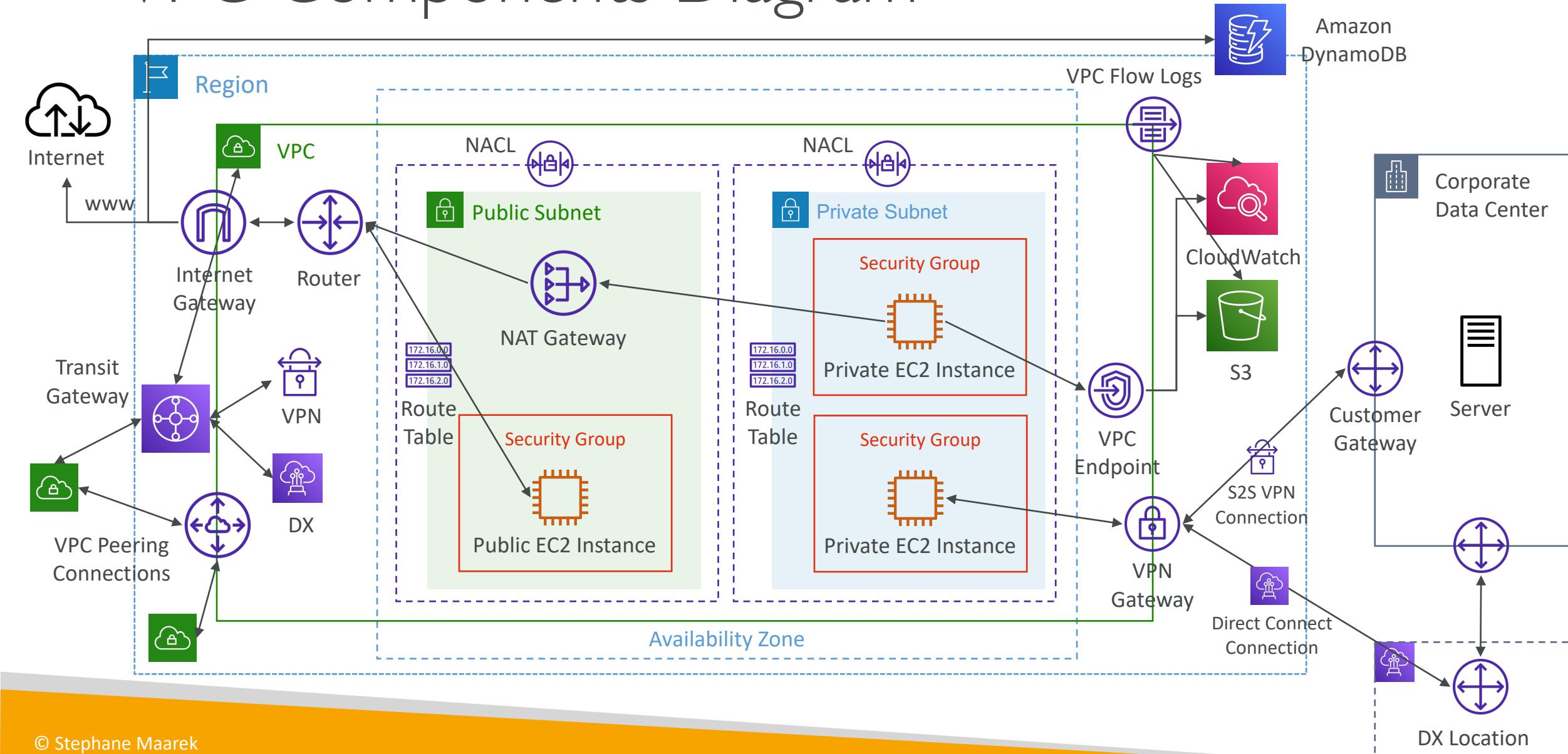


2. Create the VPC Association

```
aws route53 associate-vpc-with-hosted-zone --hosted-zone-id hosted-zone-id --vpc VPCRegion=region,VPCId=vpc-id
```

# Amazon VPC

# VPC Components Diagram



# Understanding CIDR – IPv4

- Classless Inter-Domain Routing – a method for allocating IP addresses
- Used in **Security Groups** rules and AWS networking in general

IP version	Type	Protocol	Port range	Source	Description
IPv4	SSH	TCP	22	122.149.196.85/32	-
IPv4	HTTP	TCP	80	0.0.0.0/0	-

- They help to define an IP address range:
  - We've seen WW.XX.YY.ZZ/32 => one IP
  - We've seen 0.0.0.0/0 => all IPs
  - But we can define: 192.168.0.0/26 => 192.168.0.0 – 192.168.0.63 (64 IP addresses)

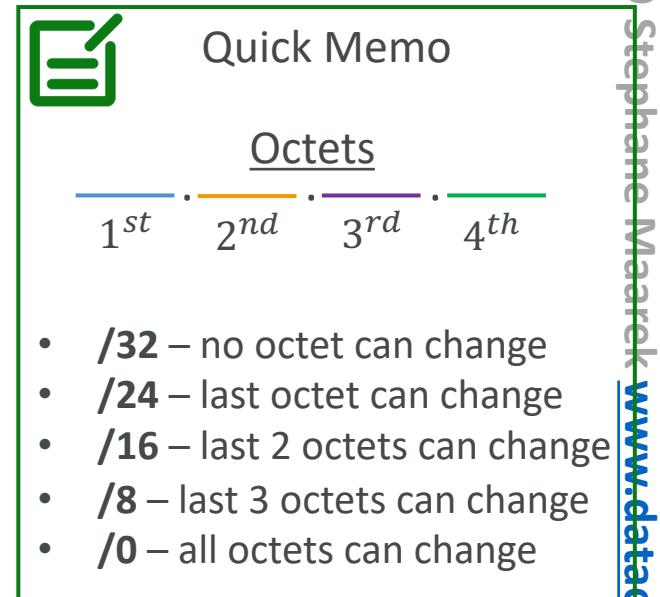
# Understanding CIDR – IPv4

- A CIDR consists of two components
- Base IP
  - Represents an IP contained in the range (XX.XX.XX.XX)
  - Example: 10.0.0.0, 192.168.0.0, ...
- Subnet Mask
  - Defines how many bits can change in the IP
  - Example: /0, /24, /32
  - Can take two forms:
    - /8  $\Leftrightarrow$  255.0.0.0
    - /16  $\Leftrightarrow$  255.255.0.0
    - /24  $\Leftrightarrow$  255.255.255.0
    - /32  $\Leftrightarrow$  255.255.255.255

# Understanding CIDR – Subnet Mask

- The Subnet Mask basically allows part of the underlying IP to get additional next values from the base IP

192	. 168 . 0 . 0 /32 => allows for <b>1</b> IP ( $2^0$ )	→ 192.168.0.0
192	. 168 . 0 . 0 /31 => allows for <b>2</b> IP ( $2^1$ )	→ 192.168.0.0 -> 192.168.0.1
192	. 168 . 0 . 0 /30 => allows for <b>4</b> IP ( $2^2$ )	→ 192.168.0.0 -> 192.168.0.3
192	. 168 . 0 . 0 /29 => allows for <b>8</b> IP ( $2^3$ )	→ 192.168.0.0 -> 192.168.0.7
192	. 168 . 0 . 0 /28 => allows for <b>16</b> IP ( $2^4$ )	→ 192.168.0.0 -> 192.168.0.15
192	. 168 . 0 . 0 /27 => allows for <b>32</b> IP ( $2^5$ )	→ 192.168.0.0 -> 192.168.0.31
192	. 168 . 0 . 0 /26 => allows for <b>64</b> IP ( $2^6$ )	→ 192.168.0.0 -> 192.168.0.63
192	. 168 . 0 . 0 /25 => allows for <b>128</b> IP ( $2^7$ )	→ 192.168.0.0 -> 192.168.0.127
192	. 168 . 0 . 0 /24 => allows for <b>256</b> IP ( $2^8$ )	→ 192.168.0.0 -> 192.168.0.255
...		
192	. 168 . 0 . 0 /16 => allows for <b>65,536</b> IP ( $2^{16}$ )	→ 192.168.0.0 -> 192.168.255.255
...		
192	. 168 . 0 . 0 /0 => allows for All IPs	→ 0.0.0.0 -> 255.255.255.255



# Understanding CIDR – Little Exercise

- $192.168.0.0/24 = \dots ?$ 
  - $192.168.0.0 - 192.168.0.255$  (256 IPs)
- $192.168.0.0/16 = \dots ?$ 
  - $192.168.0.0 - 192.168.255.255$  (65,536 IPs)
- $134.56.78.123/32 = \dots ?$ 
  - Just  $134.56.78.123$
- $0.0.0.0/0$ 
  - All IPs!
- When in doubt, use this website <https://www.ipaddressguide.com/cidr>

# Public vs. Private IP (IPv4)

- The Internet Assigned Numbers Authority (IANA) established certain blocks of IPv4 addresses for the use of private (LAN) and public (Internet) addresses
- **Private IP** can only allow certain values:
  - 10.0.0 – 10.255.255.255 (10.0.0/8) ↵ in big networks
  - 172.16.0.0 – 172.31.255.255 (172.16.0.0/12) ↵ AWS default VPC in that range
  - 192.168.0.0 – 192.168.255.255 (192.168.0.0/16) ↵ e.g., home networks
- All the rest of the IP addresses on the Internet are Public

# Default VPC Walkthrough

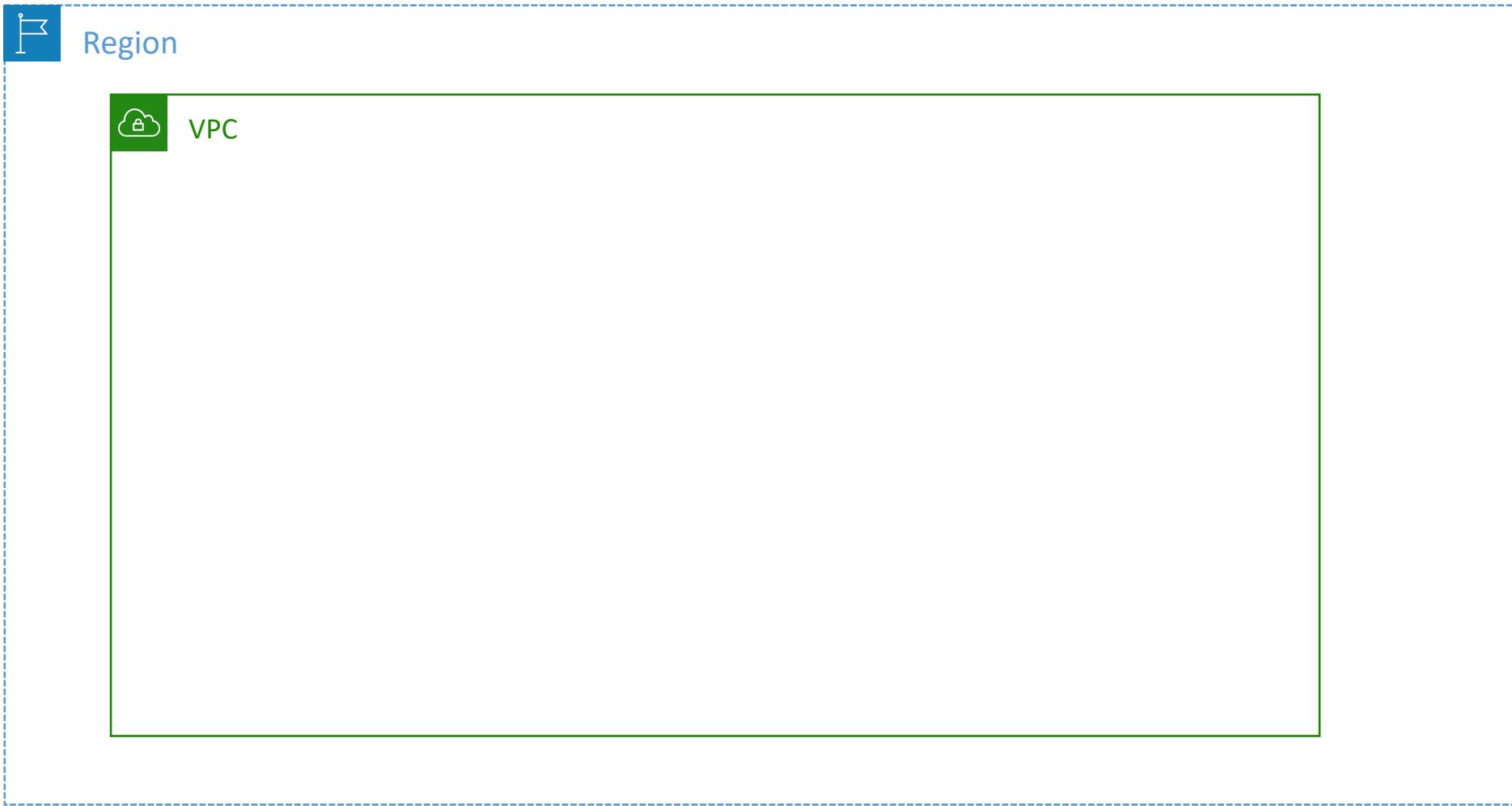
- All new AWS accounts have a default VPC
- New EC2 instances are launched into the default VPC if no subnet is specified
- Default VPC has Internet connectivity and all EC2 instances inside it have public IPv4 addresses
- We also get a public and a private IPv4 DNS names

# VPC in AWS – IPv4

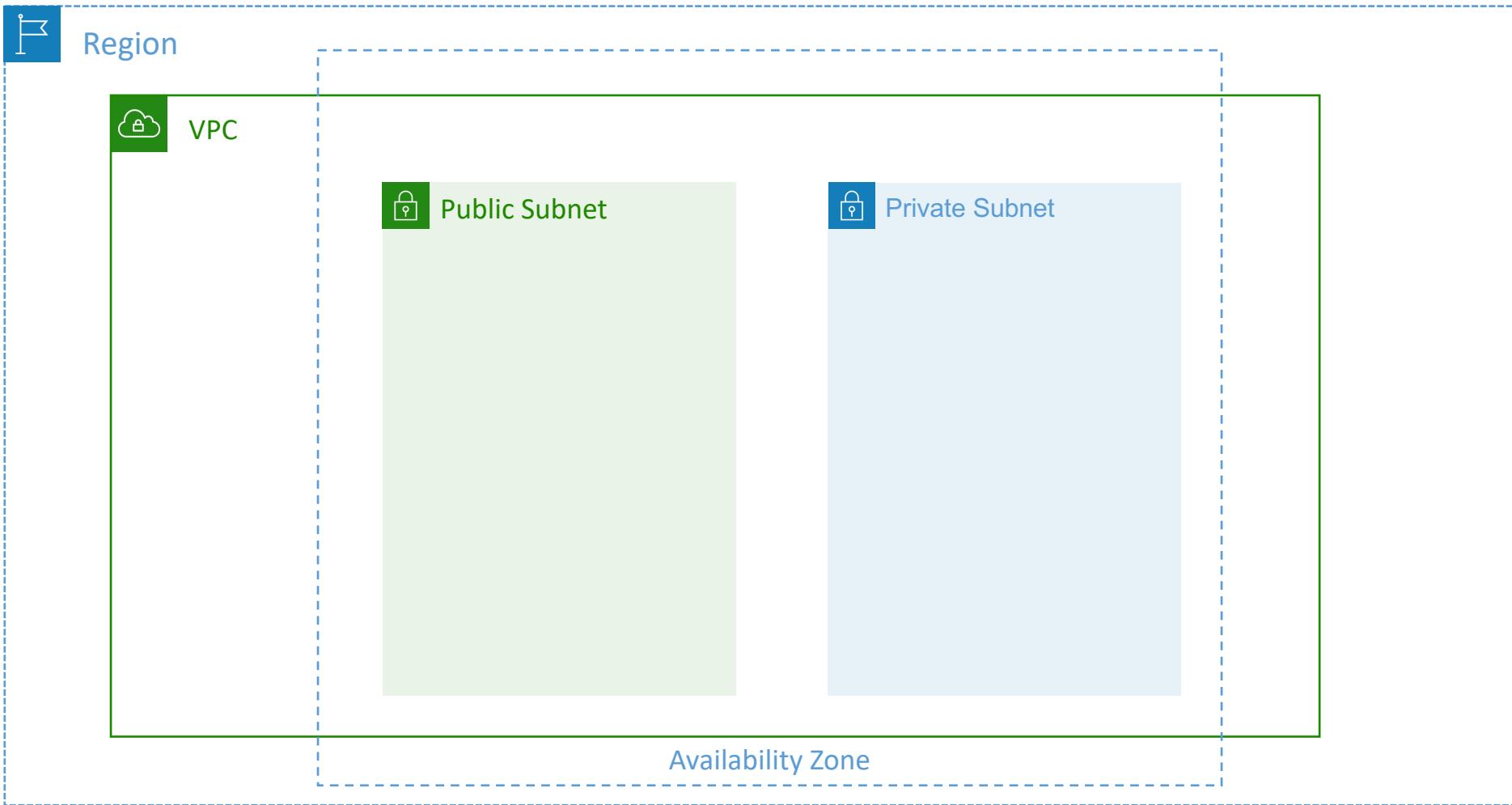


- VPC = Virtual Private Cloud
- You can have multiple VPCs in an AWS region (max. 5 per region – soft limit)
- Max. CIDR per VPC is 5, for each CIDR:
  - Min. size is /28 (16 IP addresses)
  - Max. size is /16 (65536 IP addresses)
- Because VPC is private, only the Private IPv4 ranges are allowed:
  - 10.0.0.0 – 10.255.255.255 (10.0.0.0/8)
  - 172.16.0.0 – 172.31.255.255 (172.16.0.0/12)
  - 192.168.0.0 – 192.168.255.255 (192.168.0.0/16)
- Your VPC CIDR should NOT overlap with your other networks (e.g., corporate)

# State of Hands-on



# Adding Subnets



# VPC – Subnet (IPv4)



- AWS reserves **5 IP addresses (first 4 & last 1)** in each subnet
- These 5 IP addresses are not available for use and can't be assigned to an EC2 instance
- Example: if CIDR block 10.0.0.0/24, then reserved IP addresses are:
  - 10.0.0.0 – Network Address
  - 10.0.0.1 – reserved by AWS for the VPC router
  - 10.0.0.2 – reserved by AWS for mapping to Amazon-provided DNS
  - 10.0.0.3 – reserved by AWS for future use
  - 10.0.0.255 – Network Broadcast Address. AWS does not support broadcast in a VPC, therefore the address is reserved
- **Exam Tip**, if you need 29 IP addresses for EC2 instances:
  - You can't choose a subnet of size /27 (32 IP addresses,  $32 - 5 = 27 < 29$ )
  - You need to choose a subnet of size /26 (64 IP addresses,  $64 - 5 = 59 > 29$ )

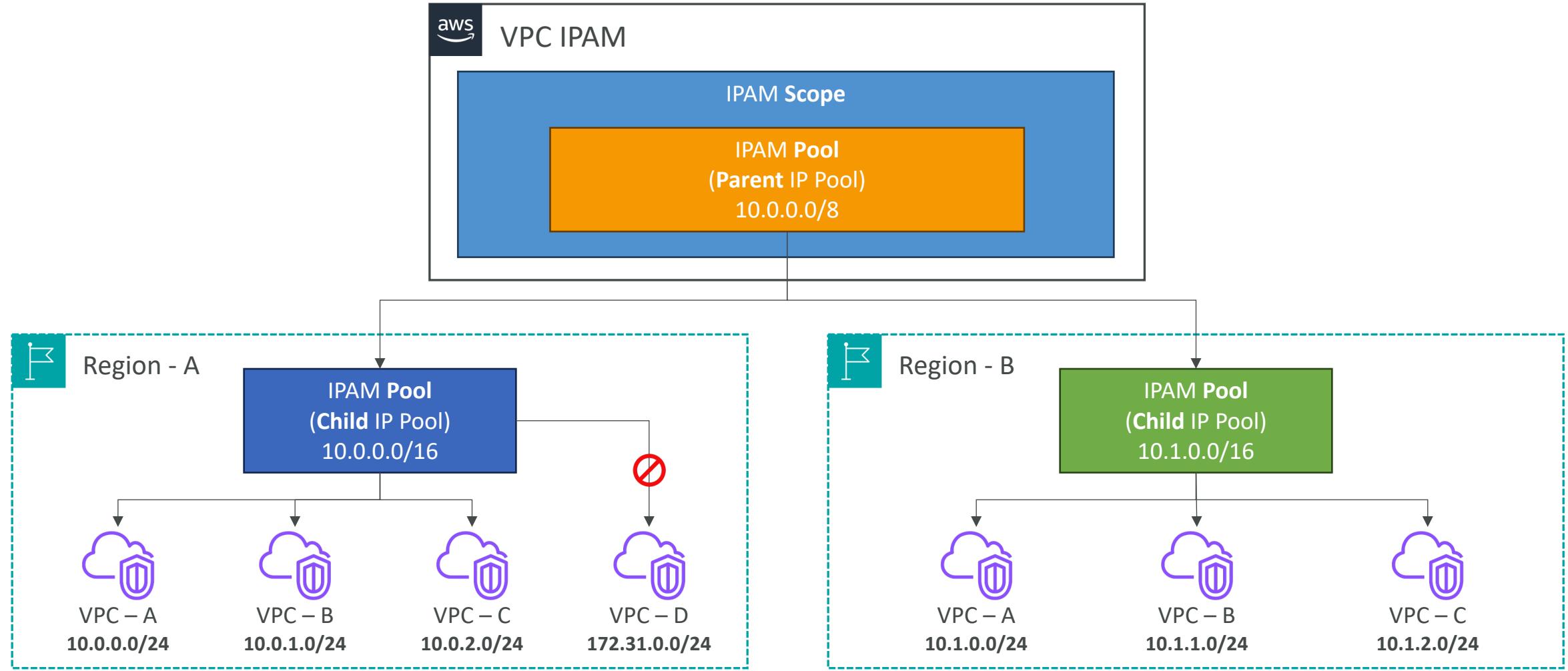
# VPC Troubleshooting

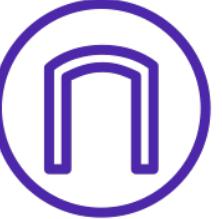
- You can **NOT** change or modify the IPv4 CIDR block of an existing VPC
  - Remove and add a new IPv4 CIDR block
  - Add a secondary IPv4 CIDR block
  - Create a new VPC with a different IPv4 CIDR block
  - Example: if your VPC CIDR conflicts with another VPC or on-premises CIDR block
- You can **NOT** change or modify the IPv4 CIDR block of an existing Subnet
  - Create a new Subnet with a new IPv4 CIDR block
  - Example: you don't have enough IPv4 addresses in your Subnet, must create a new Subnet with larger CIDR block

# VPC IP Address Manager (IPAM)

- Centrally plan, track, and monitor your IP address spaces
- Single source-of-truth for all IP address information
- Automate IP address allocations across 100s of accounts and VPCs
- **IPAM Scope** – top-level container that encapsulates the IP address space (two default scopes public & private)
- **IP Pool** – collection of IP address ranges (CIDRs) that can be divided into sub-pools for Regional IP Pools
- Monitor your IP addresses and provides alerts when it detects potential issues (e.g., overlapping IP addresses)
- Use cases: audit public IP usage, identify unused IP addresses, monitor CIDR allocations, reclaim unused IP addresses...

# VPC IP Address Manager (IPAM)

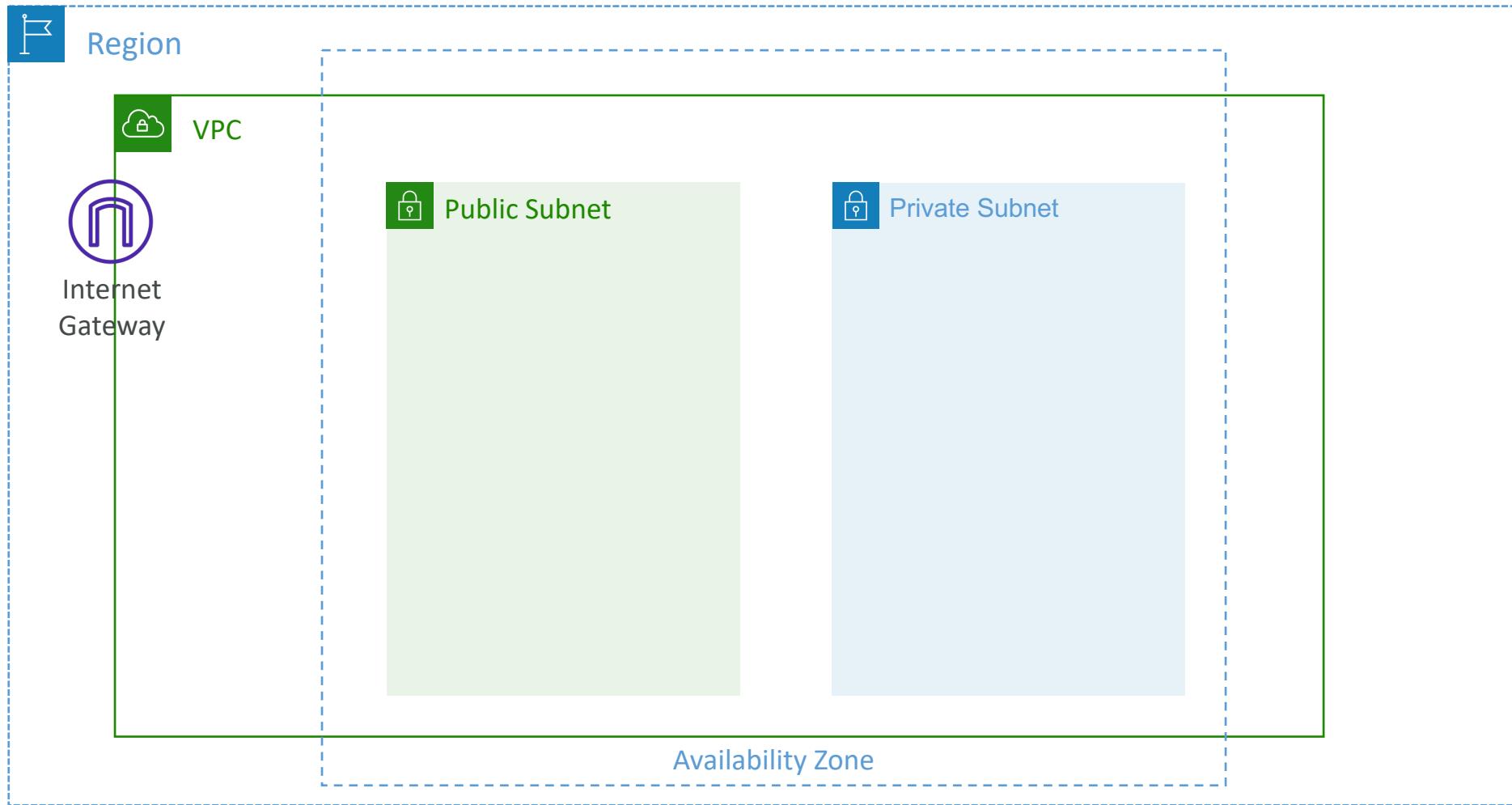




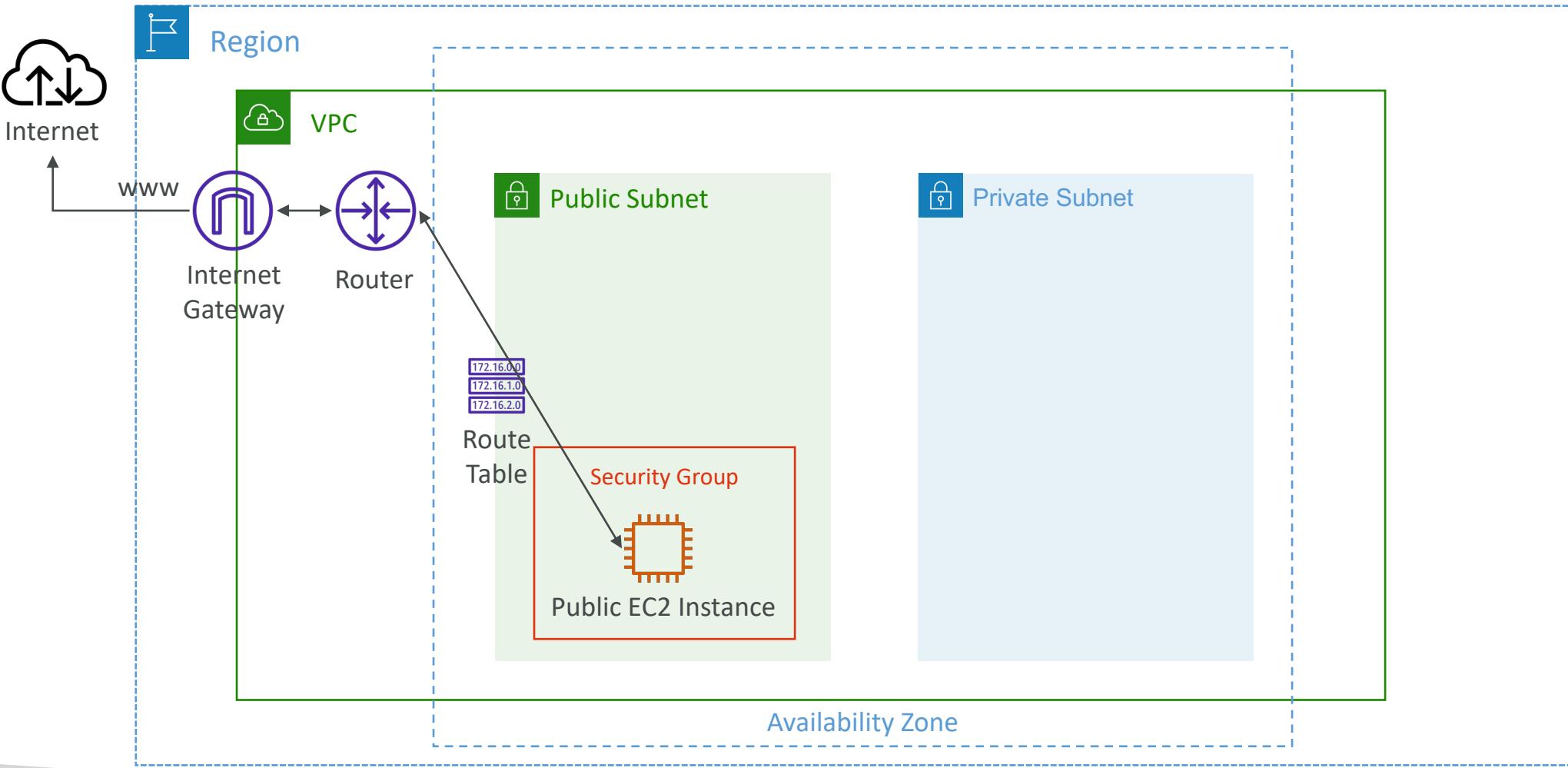
# Internet Gateway (IGW)

- Allows resources (e.g., EC2 instances) in a VPC connect to the Internet
- It scales horizontally and is highly available and redundant
- Must be created separately from a VPC
- One VPC can only be attached to one IGW and vice versa
  
- Internet Gateways on their own do not allow Internet access...
- Route tables must also be edited!

# Adding Internet Gateway

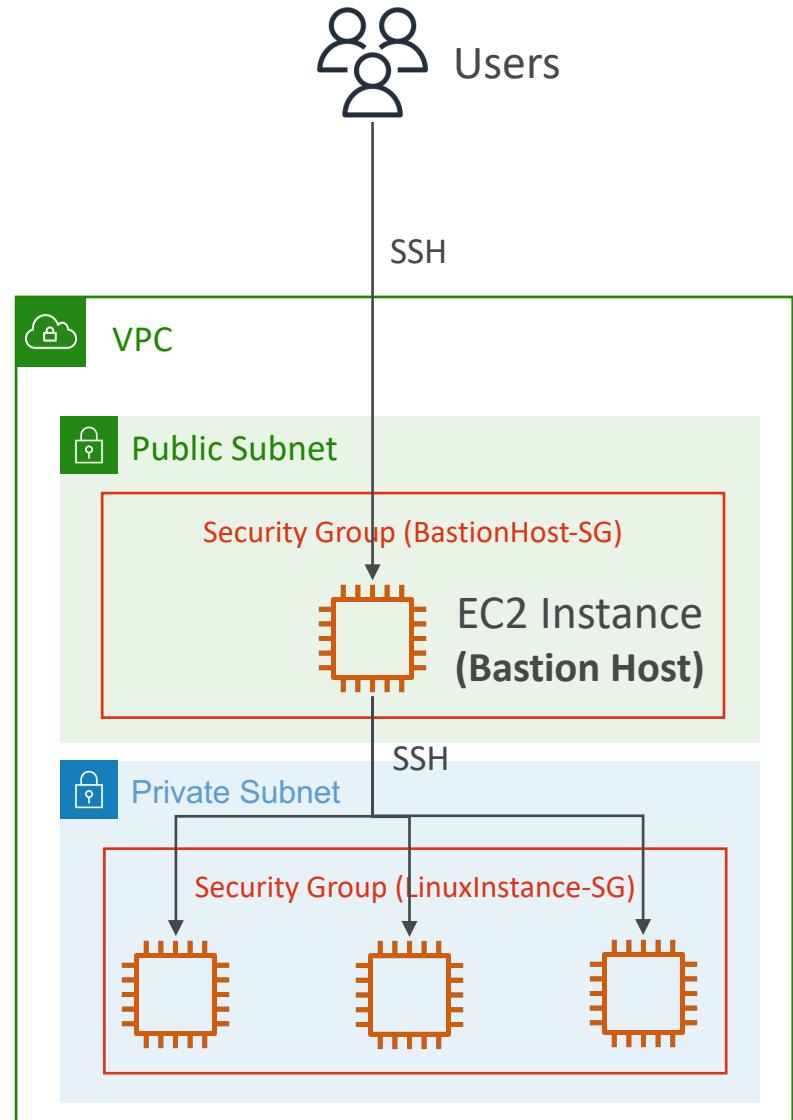


# Editing Route Tables



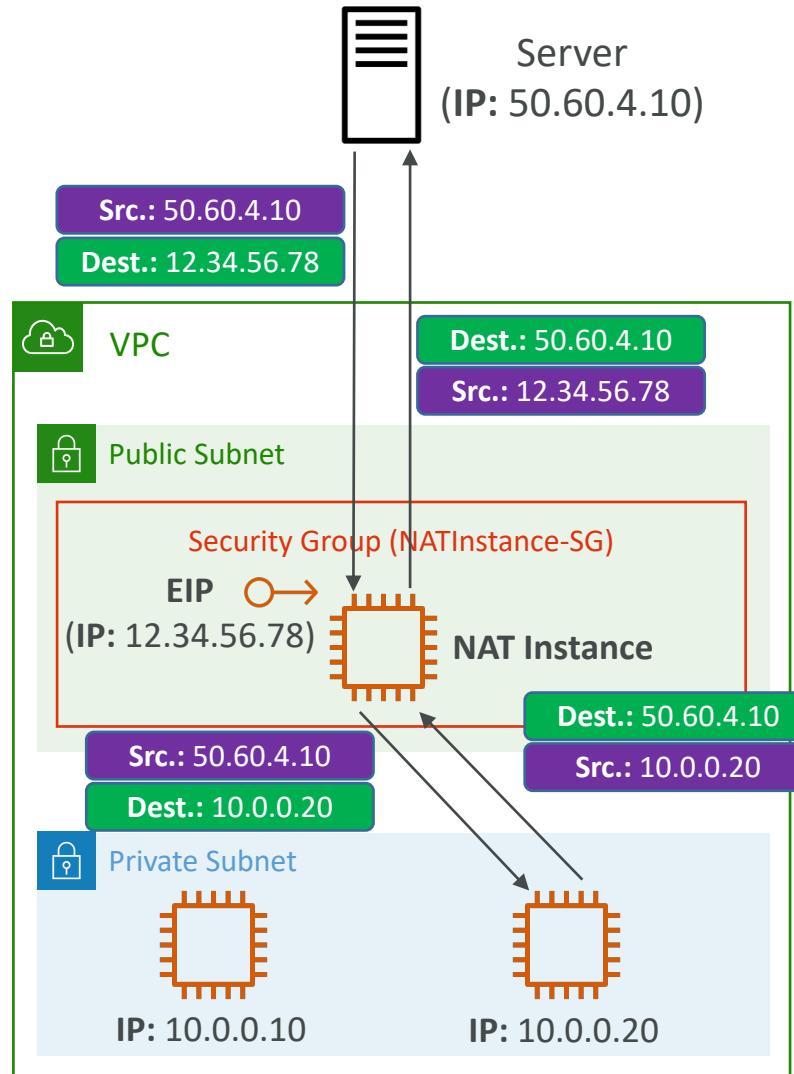
# Bastion Hosts

- We can use a Bastion Host to SSH into our private EC2 instances
- The bastion is in the public subnet which is then connected to all other private subnets
- **Bastion Host security group must allow** inbound from the internet on port 22 from restricted CIDR, for example the public CIDR of your corporation
- **Security Group of the EC2 Instances** must allow the Security Group of the Bastion Host, or the private IP of the Bastion host

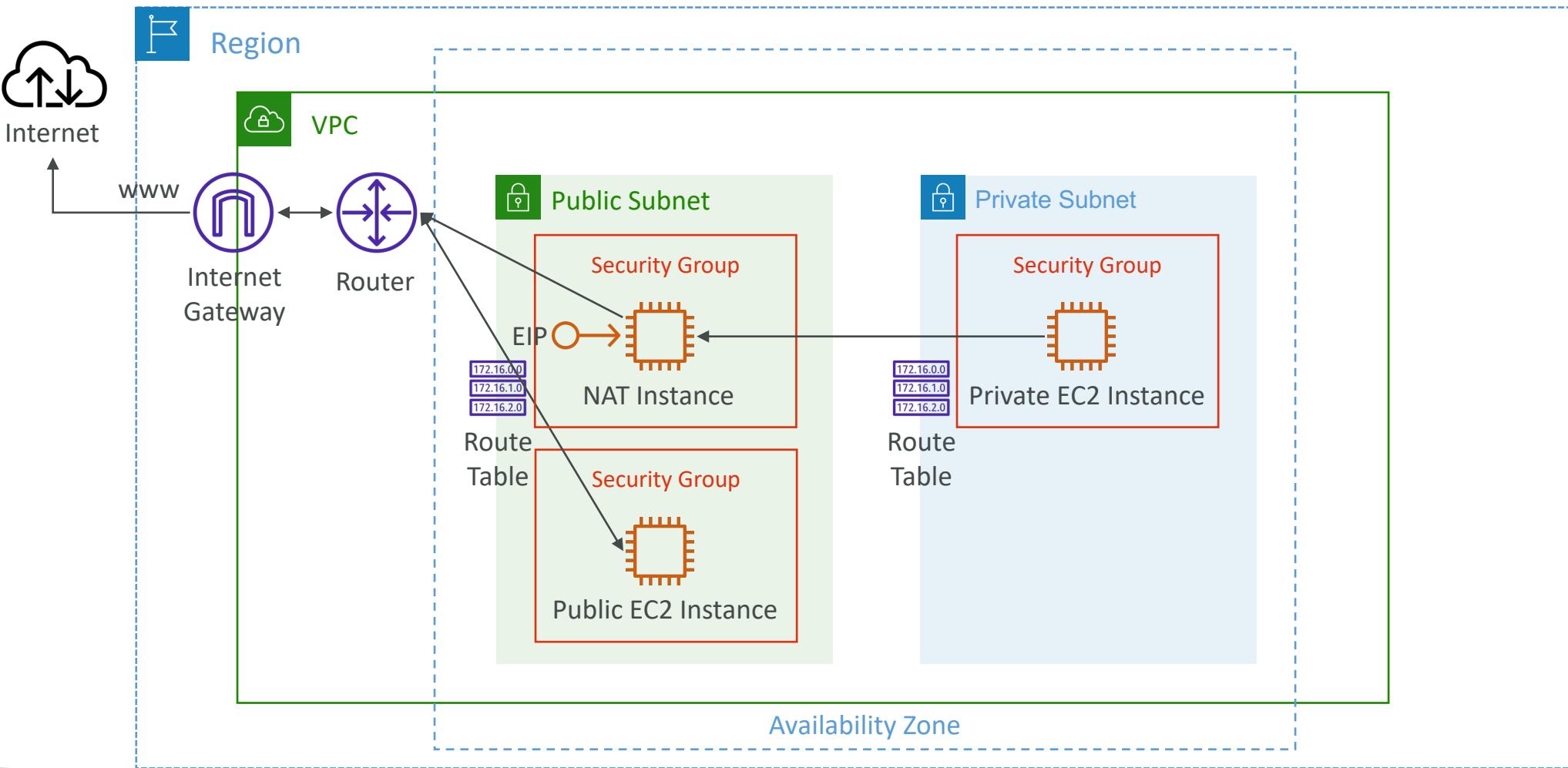


# NAT Instance (outdated, but still at the exam)

- NAT = Network Address Translation
- Allows EC2 instances in private subnets to connect to the Internet
- Must be launched in a public subnet
- Must disable EC2 setting: **Source / destination Check**
- Must have Elastic IP attached to it
- Route Tables must be configured to route traffic from private subnets to the NAT Instance

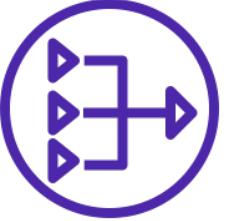


# NAT Instance



# NAT Instance – Comments

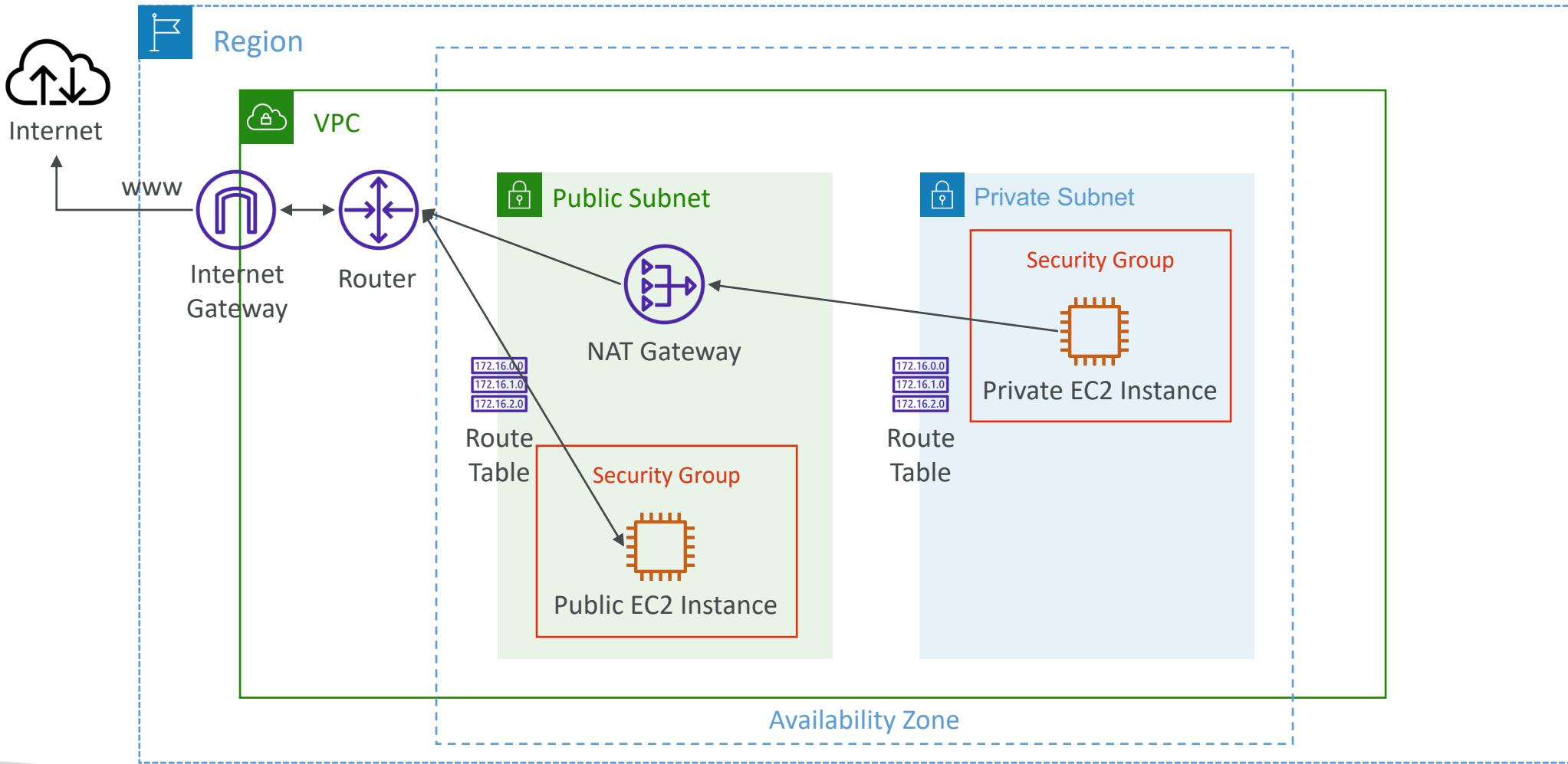
- Pre-configured Amazon Linux AMI is available
  - Reached the end of standard support on December 31, 2020
- Not highly available / resilient setup out of the box
  - You need to create an ASG in multi-AZ + resilient user-data script
- Internet traffic bandwidth depends on EC2 instance type
- You must manage Security Groups & rules:
  - Inbound:
    - Allow HTTP / HTTPS traffic coming from Private Subnets
    - Allow SSH from your home network (access is provided through Internet Gateway)
  - Outbound:
    - Allow HTTP / HTTPS traffic to the Internet



# NAT Gateway

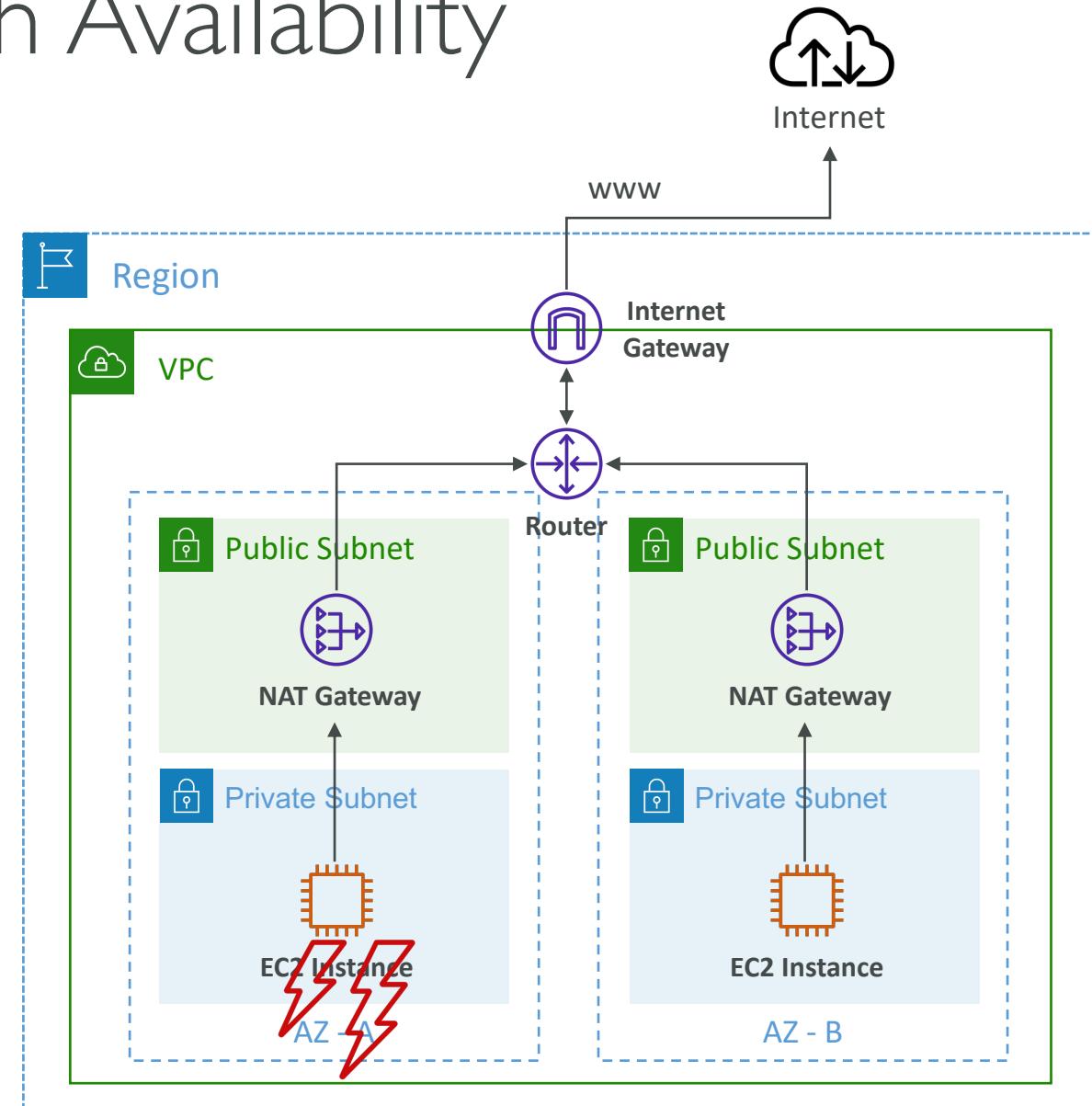
- AWS-managed NAT, higher bandwidth, high availability, no administration
- Pay per hour for usage and bandwidth
- NATGW is created in a specific Availability Zone, uses an Elastic IP
- Can't be used by EC2 instance in the same subnet (only from other subnets)
- Requires an IGW (Private Subnet => NATGW => IGW)
- 5 Gbps of bandwidth with automatic scaling up to 100 Gbps
- No Security Groups to manage / required

# NAT Gateway



# NAT Gateway with High Availability

- NAT Gateway is resilient within a single Availability Zone
- Must create multiple NAT Gateways in multiple AZs for fault-tolerance
- There is no cross-AZ failover needed because if an AZ goes down it doesn't need NAT



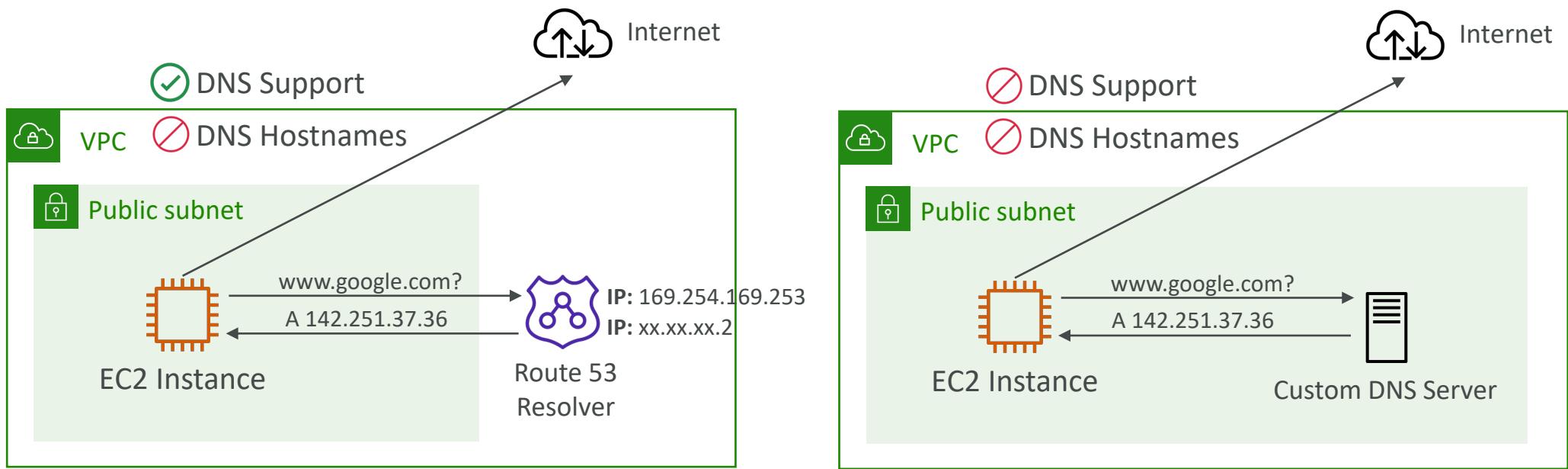
# NAT Gateway vs. NAT Instance

	NAT Gateway	NAT Instance
<b>Availability</b>	Highly available within AZ (create in another AZ)	Use a script to manage failover between instances
<b>Bandwidth</b>	Up to 100 Gbps	Depends on EC2 instance type
<b>Maintenance</b>	Managed by AWS	Managed by you (e.g., software, OS patches, ...)
<b>Cost</b>	Per hour & amount of data transferred	Per hour, EC2 instance type and size, + network \$
<b>Public IPv4</b>	✓	✓
<b>Private IPv4</b>	✓	✓
<b>Security Groups</b>	✗	✓
<b>Use as Bastion Host?</b>	✗	✓

More at: <https://docs.aws.amazon.com/vpc/latest/userguide/vpc-nat-comparison.html>

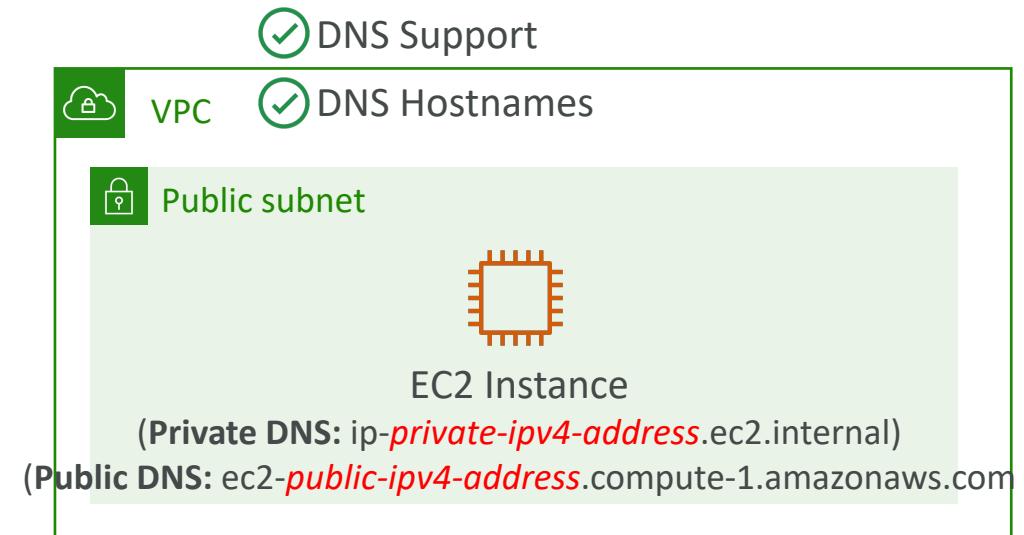
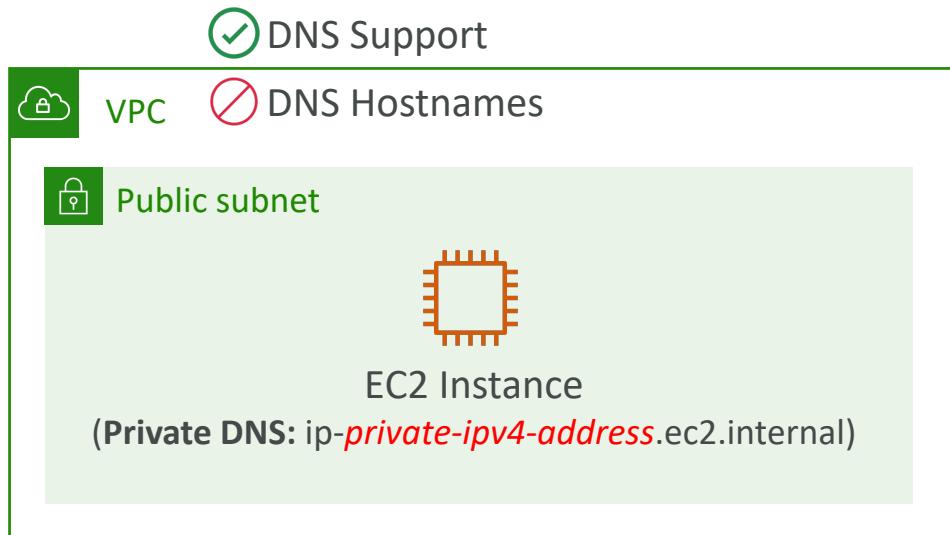
# DNS Resolution in VPC

- DNS Resolution (`enableDnsSupport`)
  - Decides if DNS resolution from Route 53 Resolver server is supported for the VPC
  - True (default): it queries the Amazon Provider DNS Server at 169.254.169.253 or the reserved IP address at the base of the VPC IPv4 network range plus two (.2)



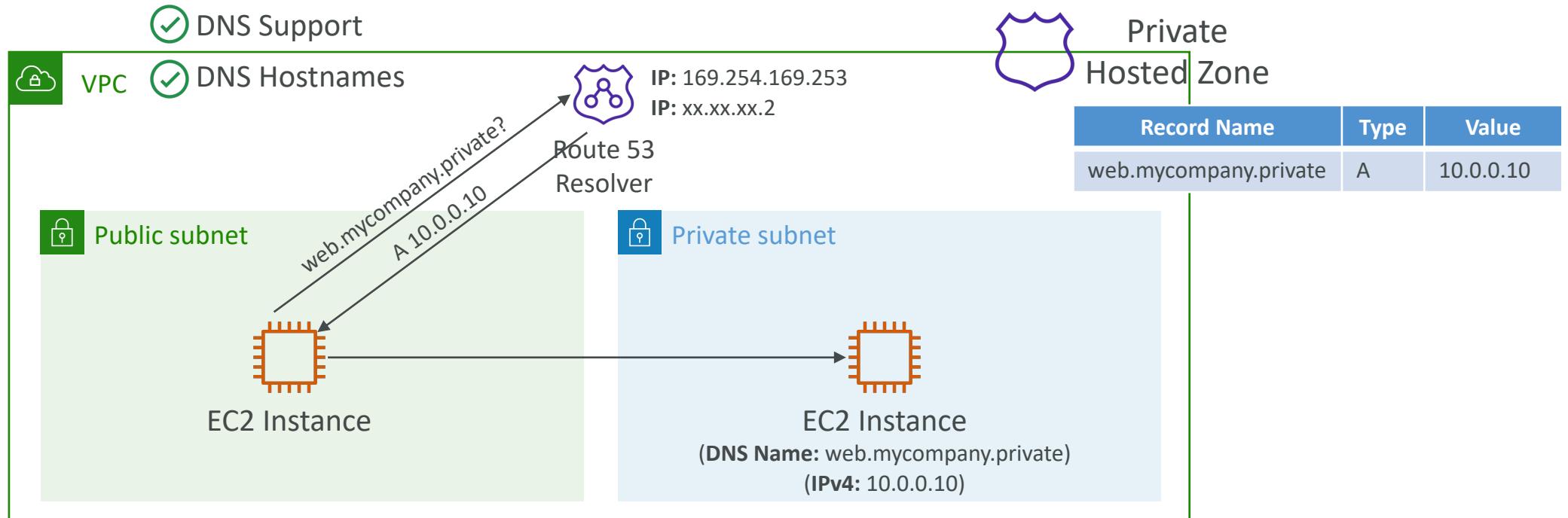
# DNS Resolution in VPC

- DNS Hostnames (enableDnsHostnames)
  - By default,
    - True => default VPC
    - False => newly created VPCs
  - Won't do anything unless enableDnsSupport=true
  - IfTrue, assigns public hostname to EC2 instance if it has a public IPv4



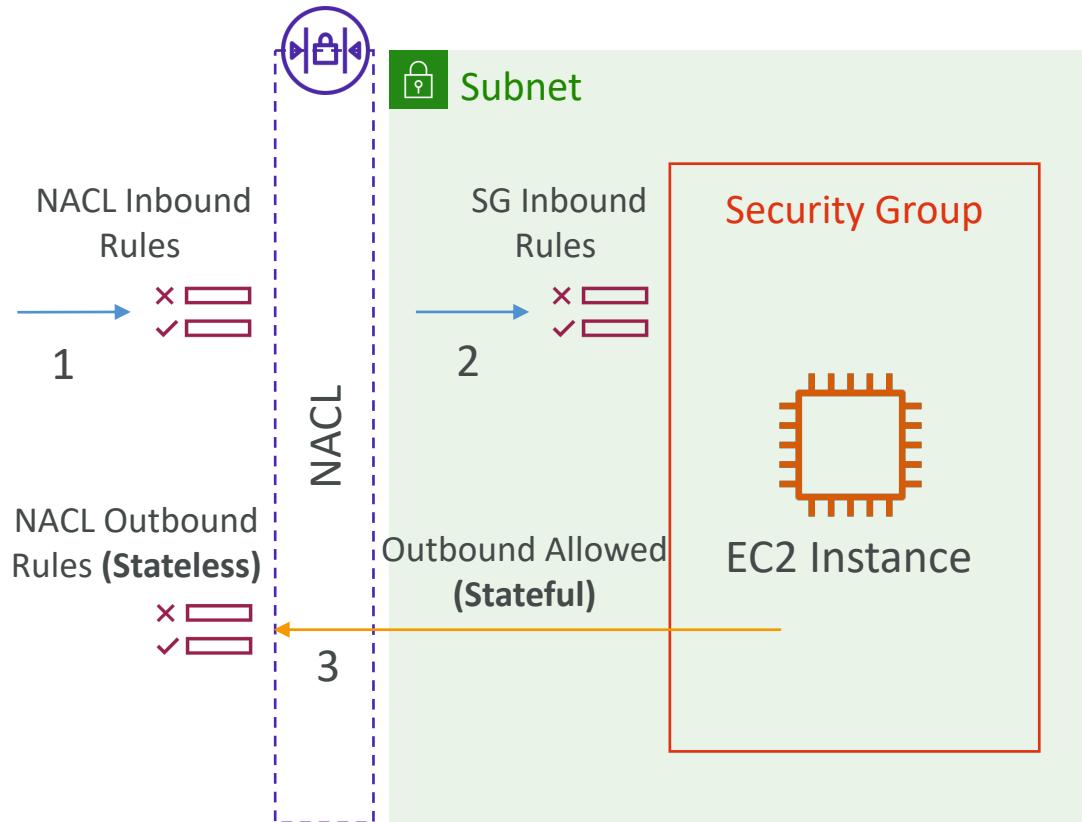
# DNS Resolution in VPC

- If you use custom DNS domain names in a Private Hosted Zone in Route 53, you must set both these attributes (enableDnsSupport & enableDnsHostname) to true

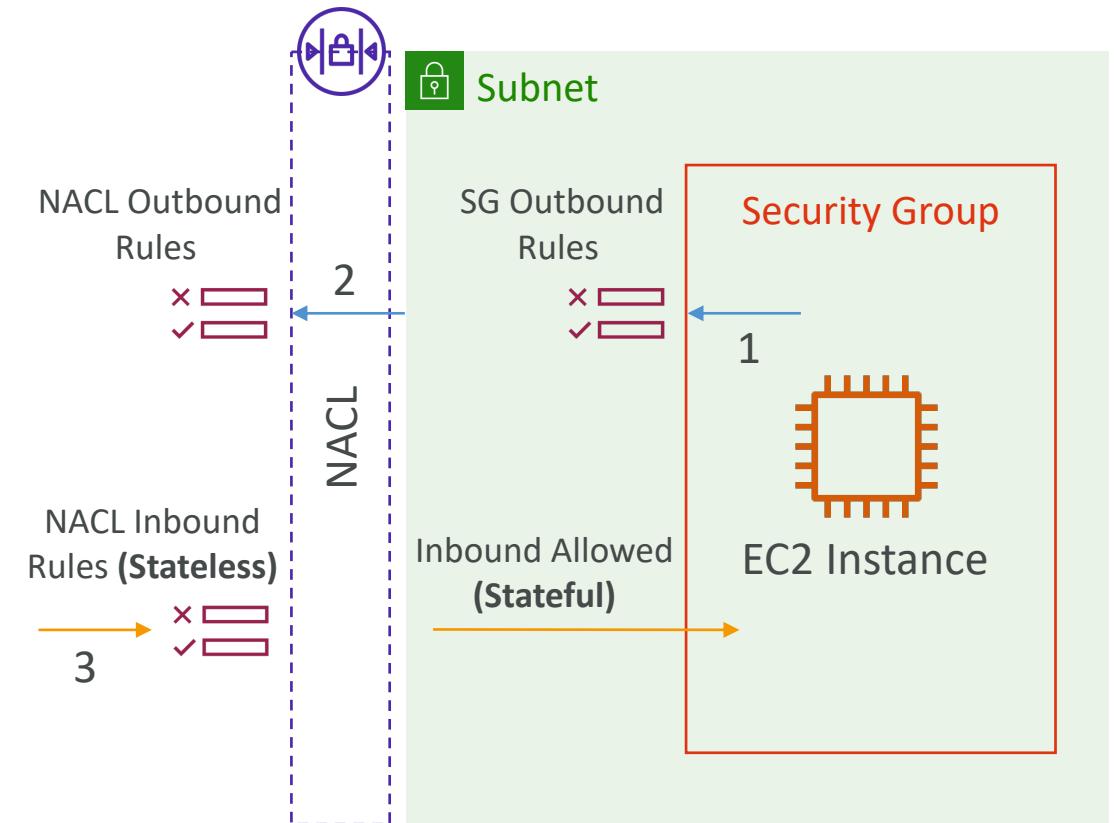


# Security Groups & NACLs

## Incoming Request



## Outgoing Request

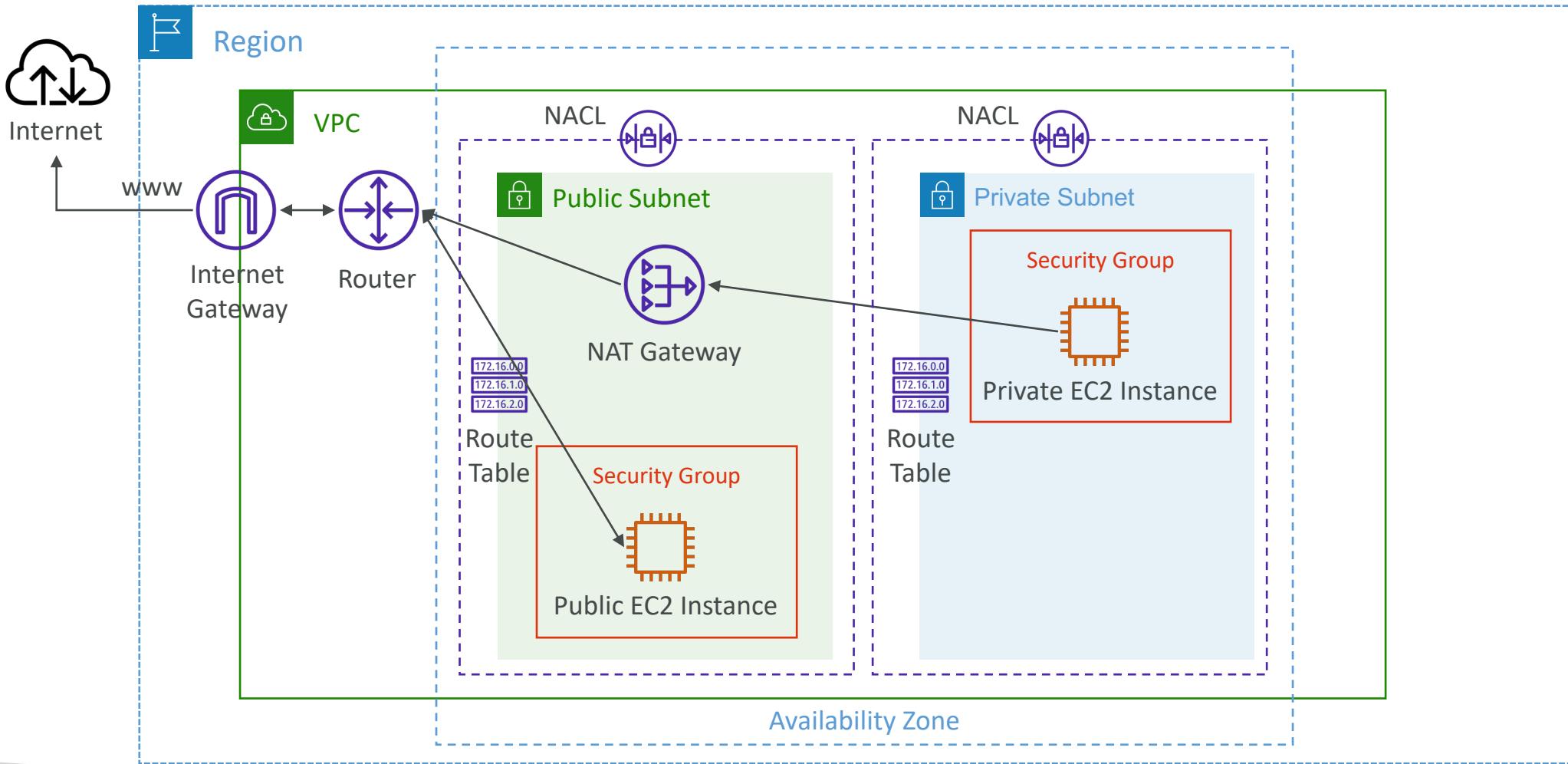




# Network Access Control List (NACL)

- NACL are like a firewall which control traffic from and to subnets
- One NACL per subnet, new subnets are assigned the Default NACL
- You define NACL Rules:
  - Rules have a number (1-32766), higher precedence with a lower number
  - First rule match will drive the decision
  - Example: if you define #100 ALLOW 10.0.0.10/32 and #200 DENY 10.0.0.10/32, the IP address will be allowed because 100 has a higher precedence over 200
  - The last rule is an asterisk (\*) and denies a request in case of no rule match
  - AWS recommends adding rules by increment of 100
- Newly created NACLs will deny everything
- NACL are a great way of blocking a specific IP address at the subnet level

# NAACLs



# Default NACL

- Accepts everything inbound/outbound with the subnets it's associated with
- Do NOT modify the Default NACL, instead create custom NACLs



Default NACL for a VPC that supports IPv4

## Inbound Rules

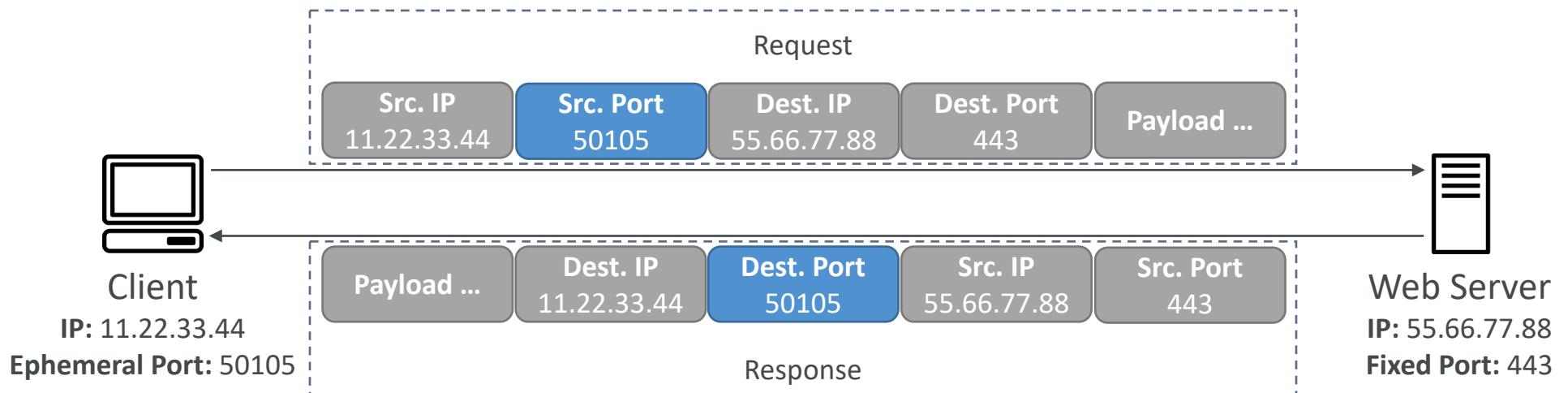
Rule #	Type	Protocol	Port Range	Source	Allow/Deny
100	All IPv4 Traffic	All	All	0.0.0.0/0	ALLOW
*	All IPv4 Traffic	All	All	0.0.0.0/0	DENY

## Outbound Rules

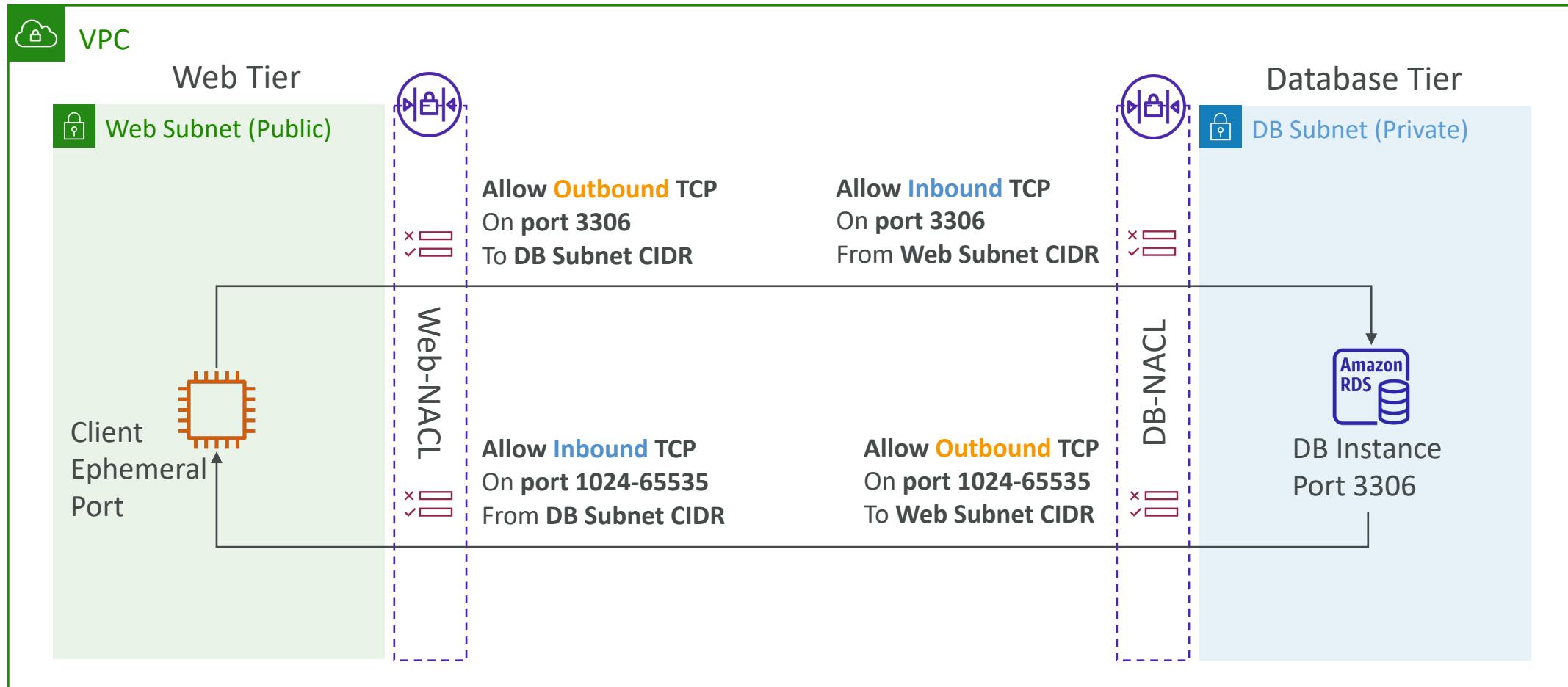
Rule #	Type	Protocol	Port Range	Destination	Allow/Deny
100	All IPv4 Traffic	All	All	0.0.0.0/0	ALLOW
*	All IPv4 Traffic	All	All	0.0.0.0/0	DENY

# Ephemeral Ports

- For any two endpoints to establish a connection, they must use ports
- Clients connect to a **defined port**, and expect a response on an **ephemeral port**
- Different Operating Systems use different port ranges, examples:
  - IANA & MS Windows 10 → 49152 – 65535
  - Many Linux Kernels → 32768 – 60999

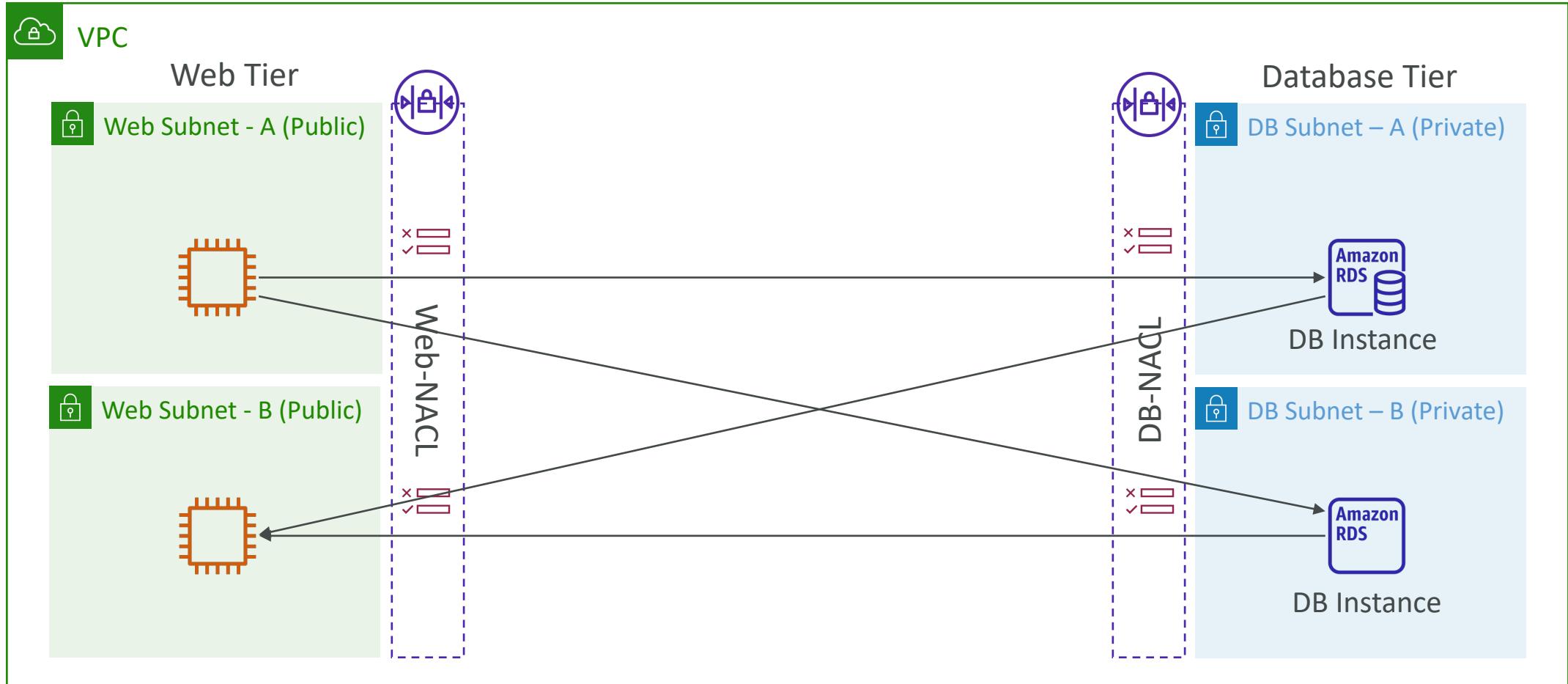


# NACL with Ephemeral Ports



<https://docs.aws.amazon.com/vpc/latest/userguide/vpc-network-acls.html#nacl-ephemeral-ports>

# Create NACL rules for each target subnets CIDR



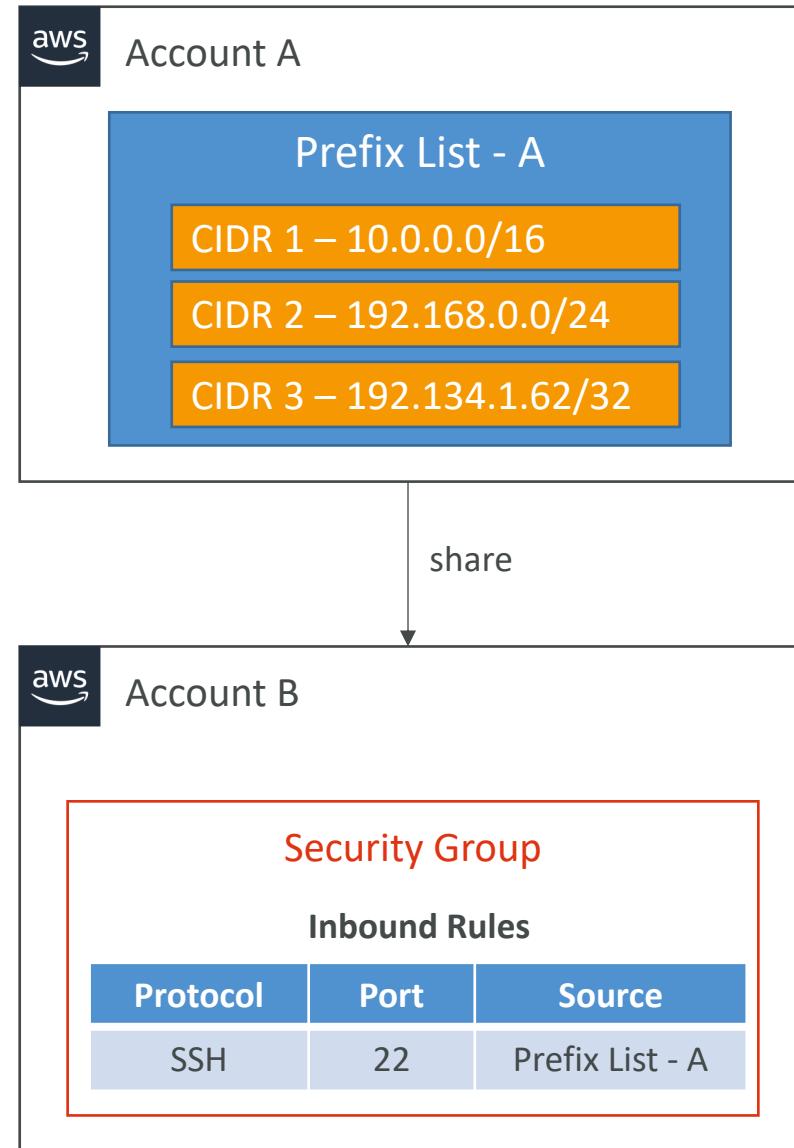
# Security Group vs. NACLs

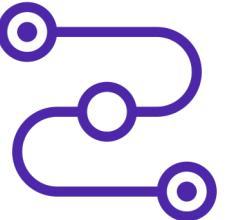
Security Group	NACL
Operates at the instance level	Operates at the subnet level
Supports allow rules only	Supports allow rules and deny rules
<b>Stateful:</b> return traffic is automatically allowed, regardless of any rules	<b>Stateless:</b> return traffic must be explicitly allowed by rules (think of ephemeral ports)
All rules are evaluated before deciding whether to allow traffic	Rules are evaluated in order (lowest to highest) when deciding whether to allow traffic, first match wins
Applies to an EC2 instance when specified by someone	Automatically applies to all EC2 instances in the subnet that it's associated with

NACL Examples: <https://docs.aws.amazon.com/vpc/latest/userguide/vpc-network-acls.html>

# Managed Prefix List

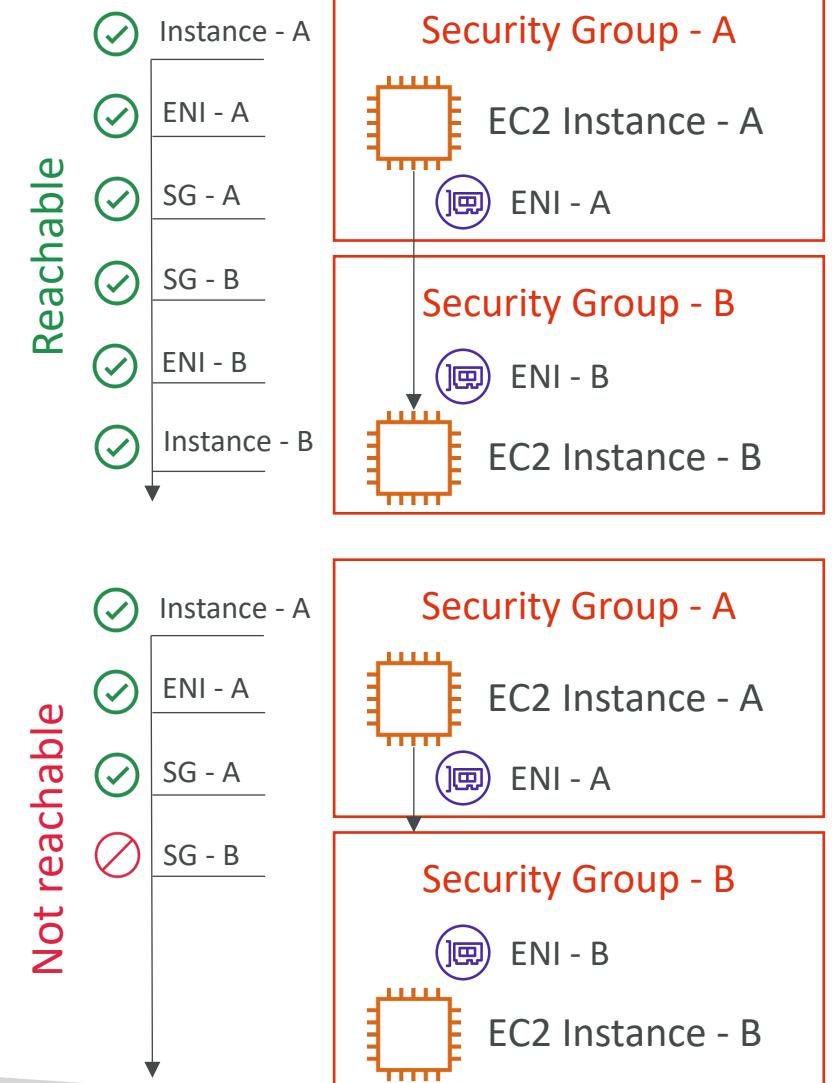
- A set of one or more CIDR blocks
- Makes it easier to configure and maintain Security Groups and Route Tables
- **Customer-Managed Prefix List**
  - Set of CIDRs that you define and managed by you
  - Can be shared with other AWS accounts or AWS Organization
  - Modify to update many Security Groups at once
- **AWS-Managed Prefix List**
  - Set of CIDRs for AWS services
  - You can't create, modify, share, or delete them
  - S3, CloudFront, DynamoDB, Ground Station...





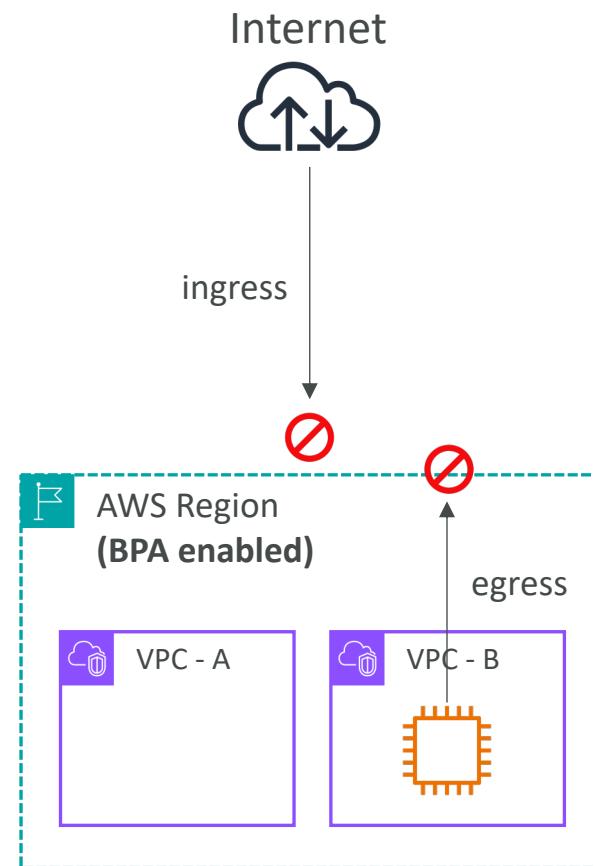
# VPC – Reachability Analyzer

- A network diagnostics tool that troubleshoots network connectivity between two endpoints in your VPC(s)
- It builds a model of the network configuration, then checks the reachability based on these configurations (**it doesn't send packets**)
- When the destination is
  - **Reachable** – it produces hop-by-hop details of the virtual network path
  - **Not reachable** – it identifies the blocking component(s) (e.g., configuration issues in SGs, NACLs, Route Tables, ...)
- Use cases: troubleshoot connectivity issues, ensure network configuration is as intended, ...



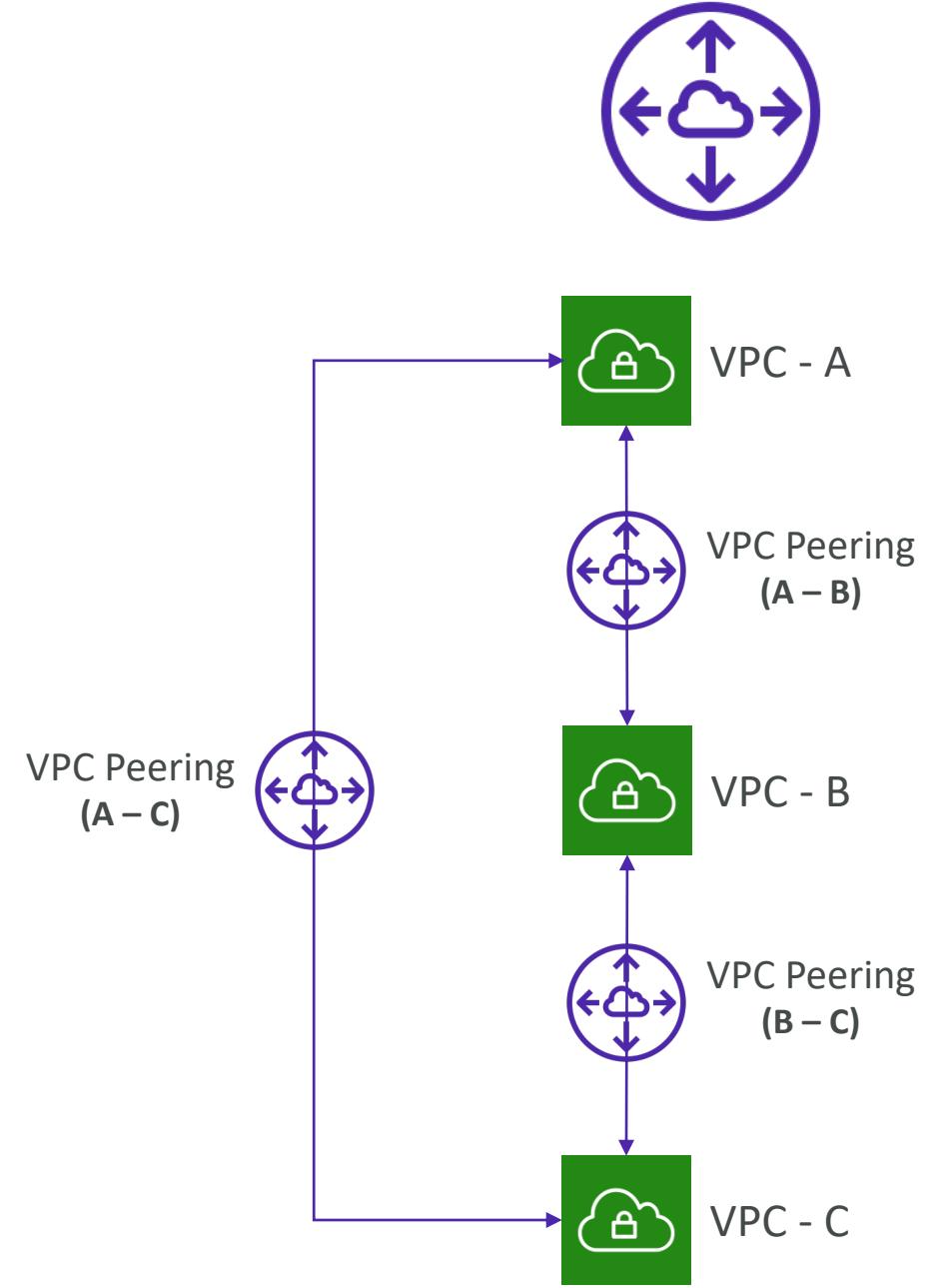
# VPC Block Public Access (BPA)

- Centrally block ingress/egress Internet access to VPCs & Subnets
- Helps you ensure compliance and security requirements
- Two Modes:
  - Bidirectional – block all Internet traffic to/from VPCs
  - Ingress-Only – block all Internet traffic to/from VPCs **except** for NAT Gateways and Egress-only Internet Gateways
- Ability to exclude single VPCs and Subnets, with two **Exclusion** modes:
  - Bidirectional – allow all Internet traffic to/from excluded VPCs & Subnets
  - Egress-Only – allow only outbound Internet from excluded VPCs & Subnets
- Can be enabled per AWS Region
- Verify if traffic blocked using VPC Reachability Analyzer



# VPC Peering

- Privately connect two VPCs using AWS' network
- Make them behave as if they were in the same network
- Must not have overlapping CIDRs
- VPC Peering connection is **NOT** transitive (must be established for each VPC that need to communicate with one another)
- You must update route tables in each VPC's subnets to ensure EC2 instances can communicate with each other



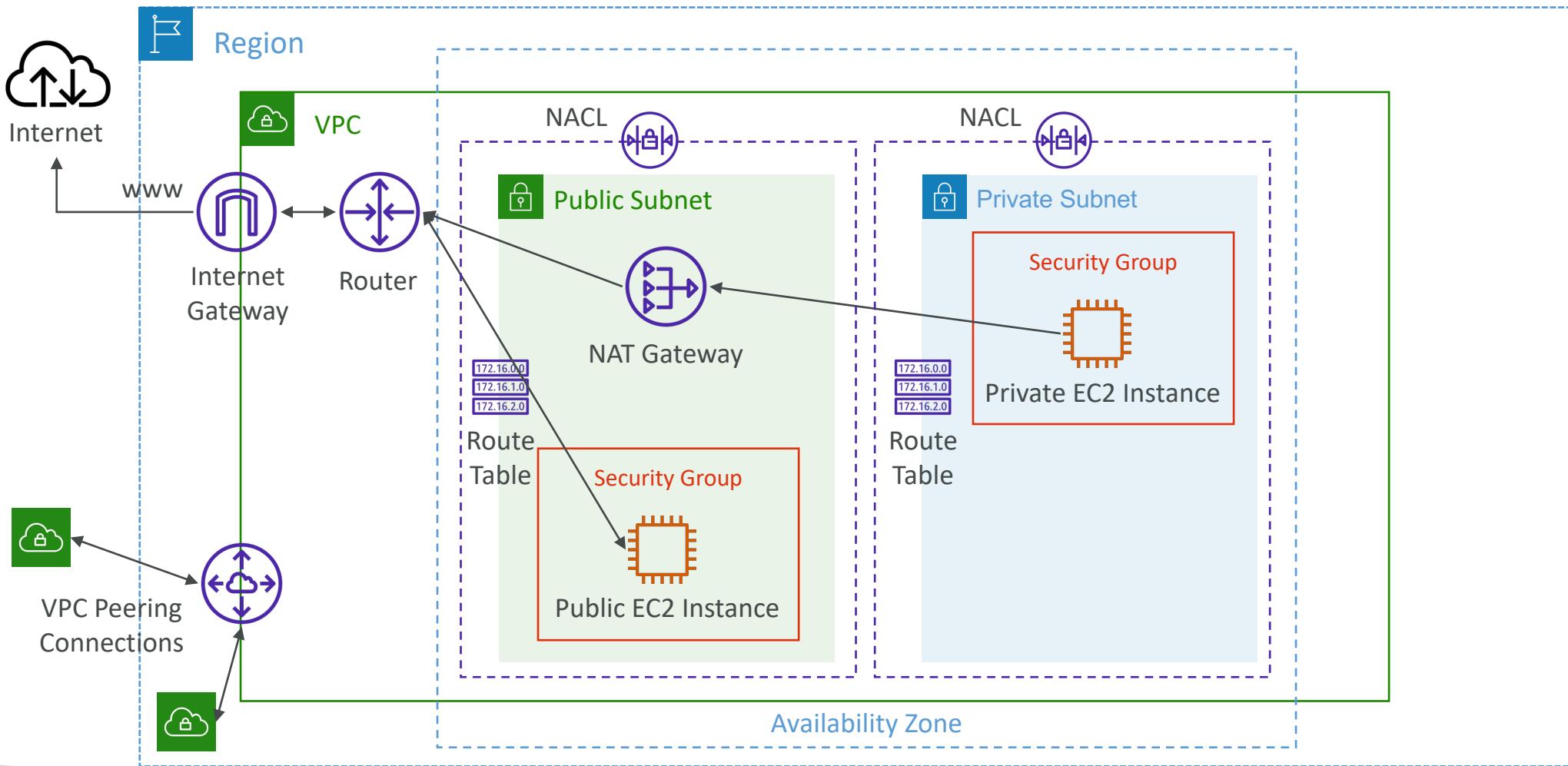
# VPC Peering – Good to know

- You can create VPC Peering connection between VPCs in different AWS accounts/regions
- You can reference a security group in a peered VPC (works cross accounts – same region)

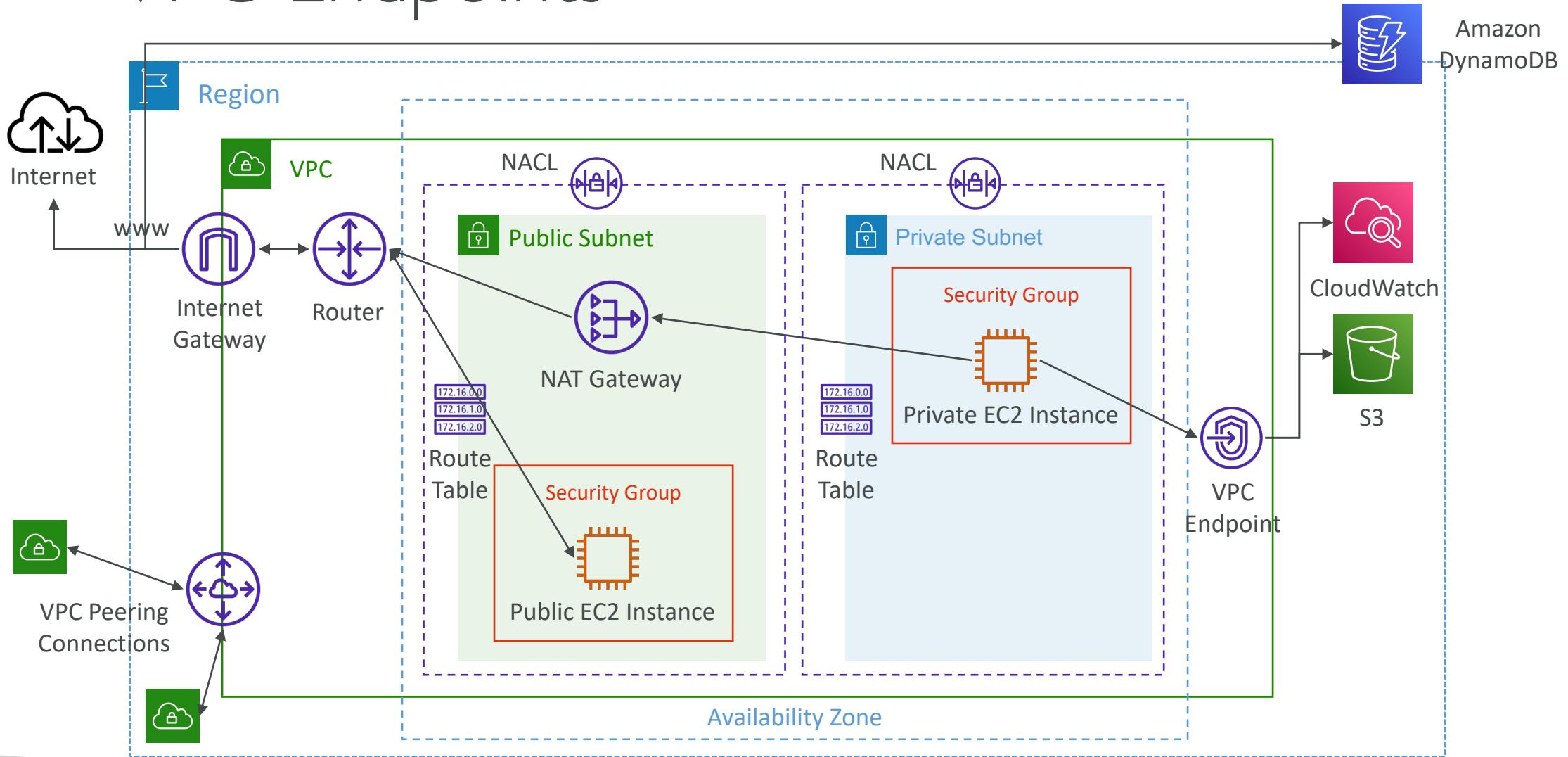
Type	Protocol	Port range	Source
HTTP	TCP	80	sg-04991f9af3473b939 / default
HTTP	TCP	80	[REDACTED] / sg-027ad1f7865d4be76

↑  
Account ID

# VPC Peering



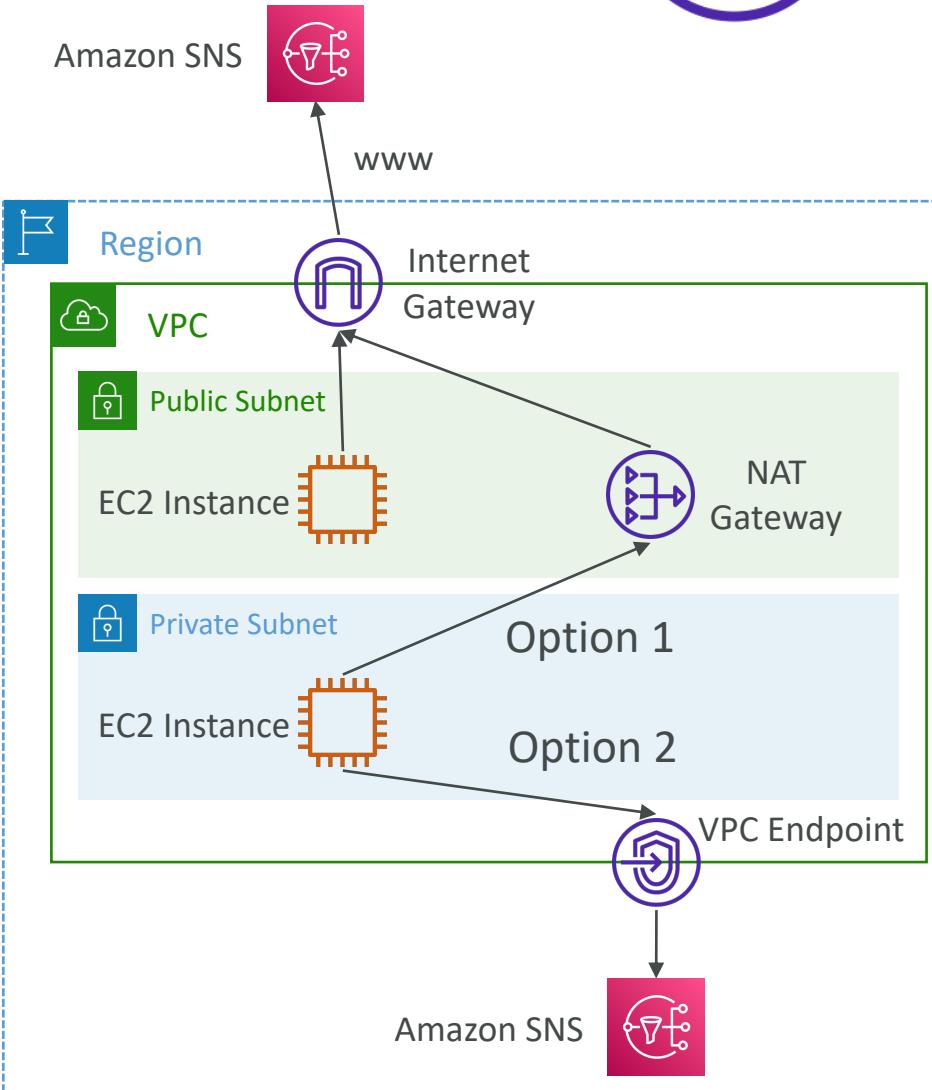
# VPC Endpoints



# VPC Endpoints (AWS PrivateLink)



- Every AWS service is publicly exposed (public URL)
- VPC Endpoints (powered by AWS PrivateLink) allows you to connect to AWS services using a **private network** instead of using the public Internet
- They're redundant and scale horizontally
- They remove the need of IGW, NATGW, ... to access AWS Services
- In case of issues:
  - Check DNS Setting Resolution in your VPC
  - Check Route Tables



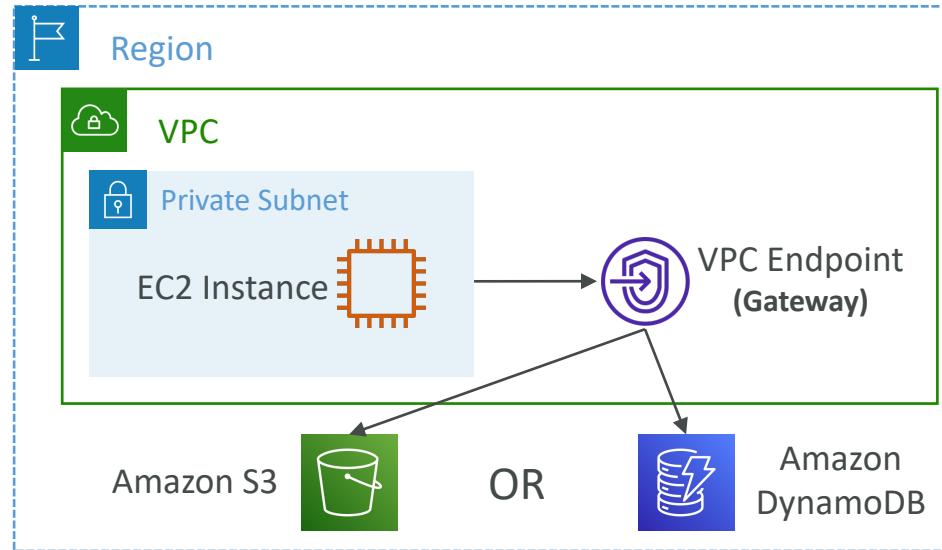
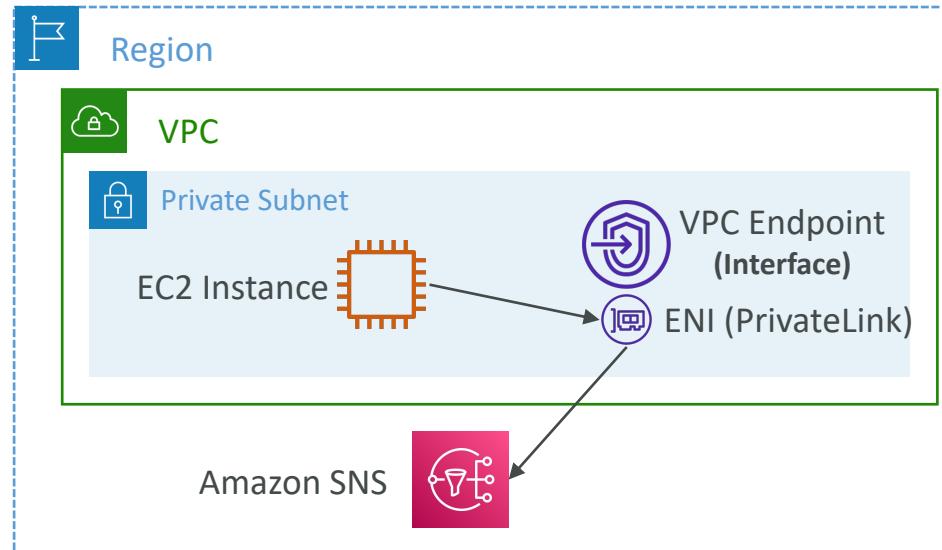
# Types of Endpoints

- Interface Endpoints (powered by PrivateLink)

- Provisions an ENI (private IP address) as an entry point (must attach a Security Group)
- Supports most AWS services
- \$ per hour + \$ per GB of data processed

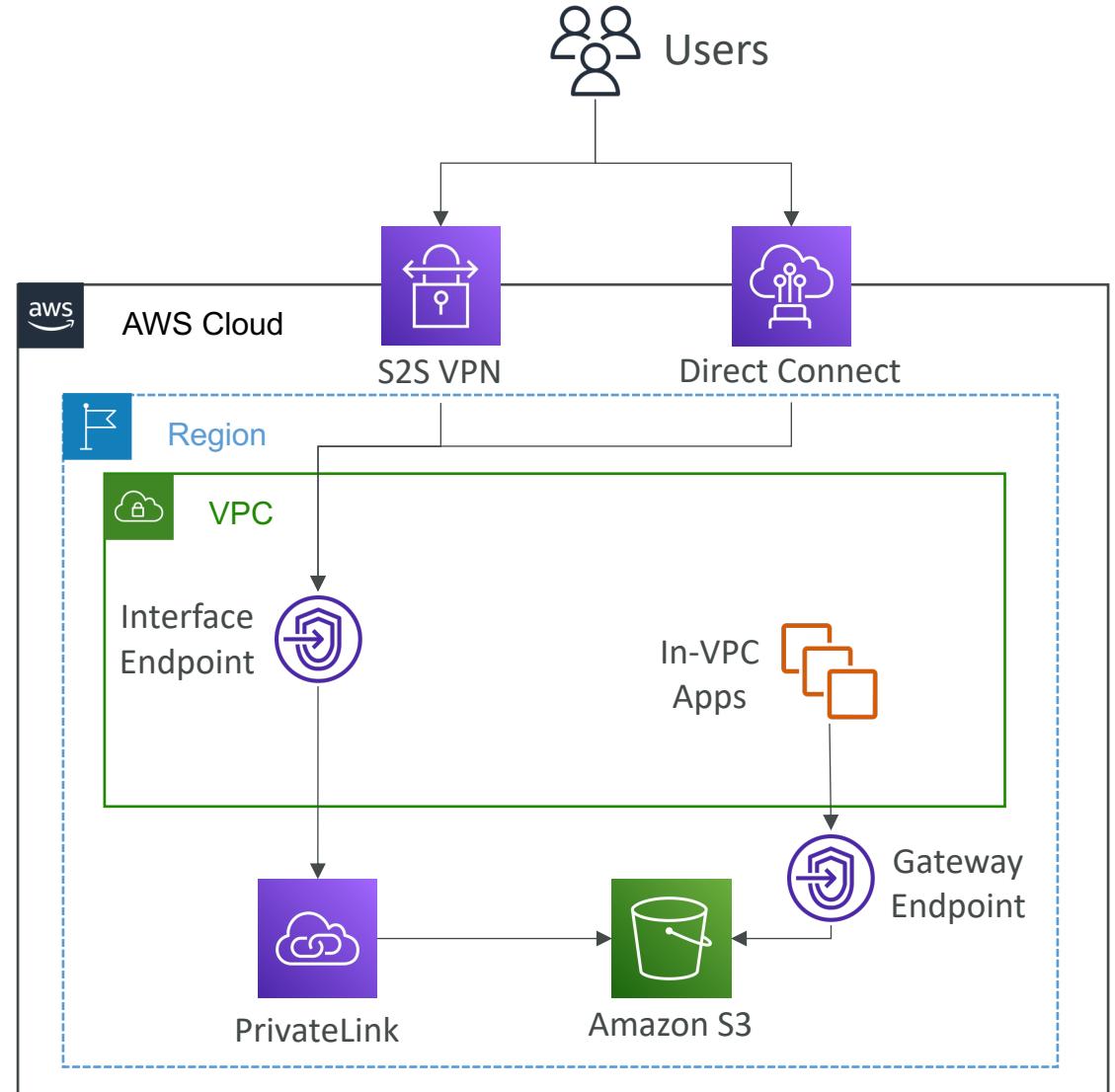
- Gateway Endpoints

- Provisions a gateway and must be used as a target in a route table (does not use security groups)
- Supports both S3 and DynamoDB
- Free



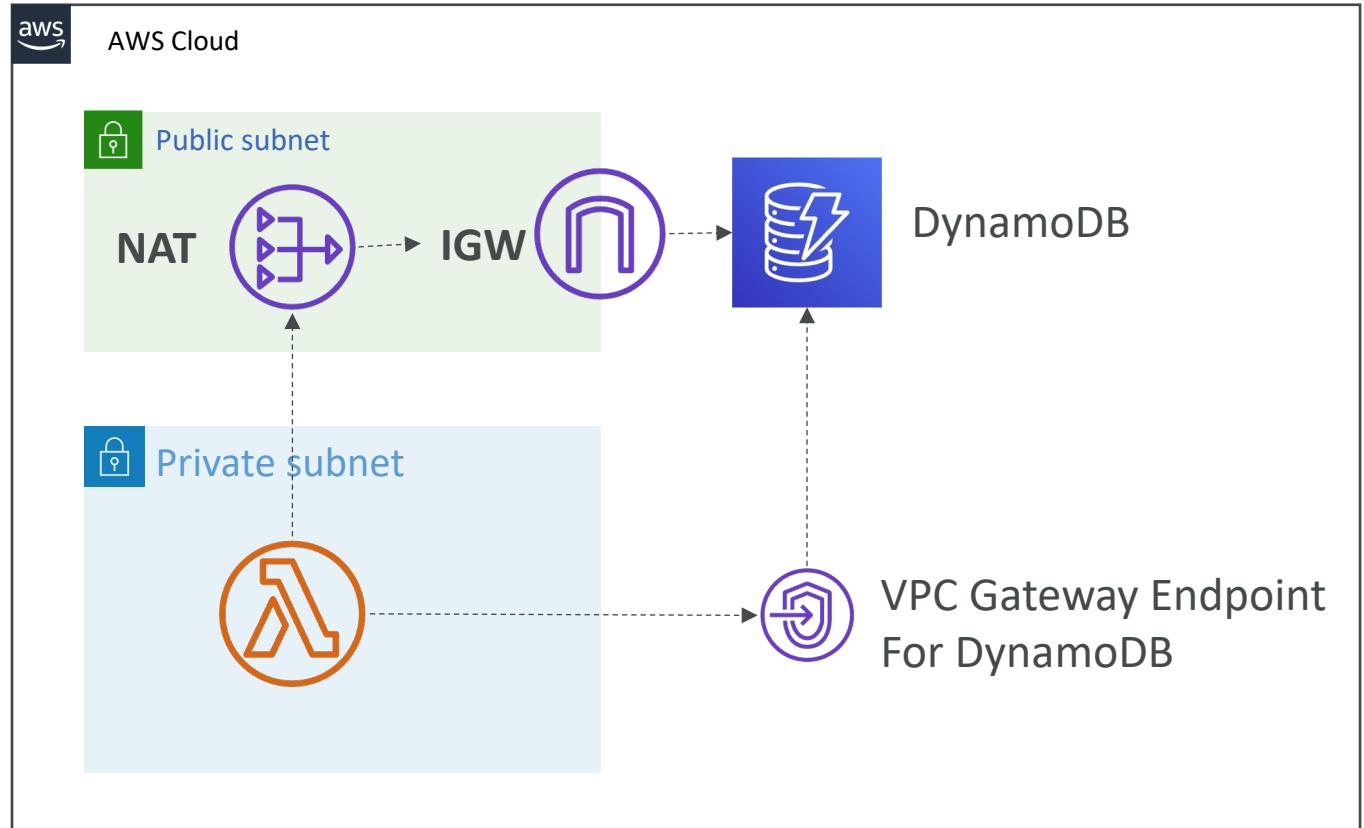
# Gateway or Interface Endpoint for S3?

- Gateway is most likely going to be preferred all the time at the exam
- Cost: free for Gateway, \$ for interface endpoint
- Interface Endpoint is preferred access is required from on-premises (Site to Site VPN or Direct Connect), a different VPC or a different region



# Lambda in VPC accessing DynamoDB

- DynamoDB is a public service from AWS
- Option 1: Access from the public internet
  - Because Lambda is in a VPC, it needs a NAT Gateway in a public subnet and an internet gateway
- Option 2 (better & free): Access from the private VPC network
  - Deploy a VPC Gateway endpoint for DynamoDB
  - Change the Route Tables

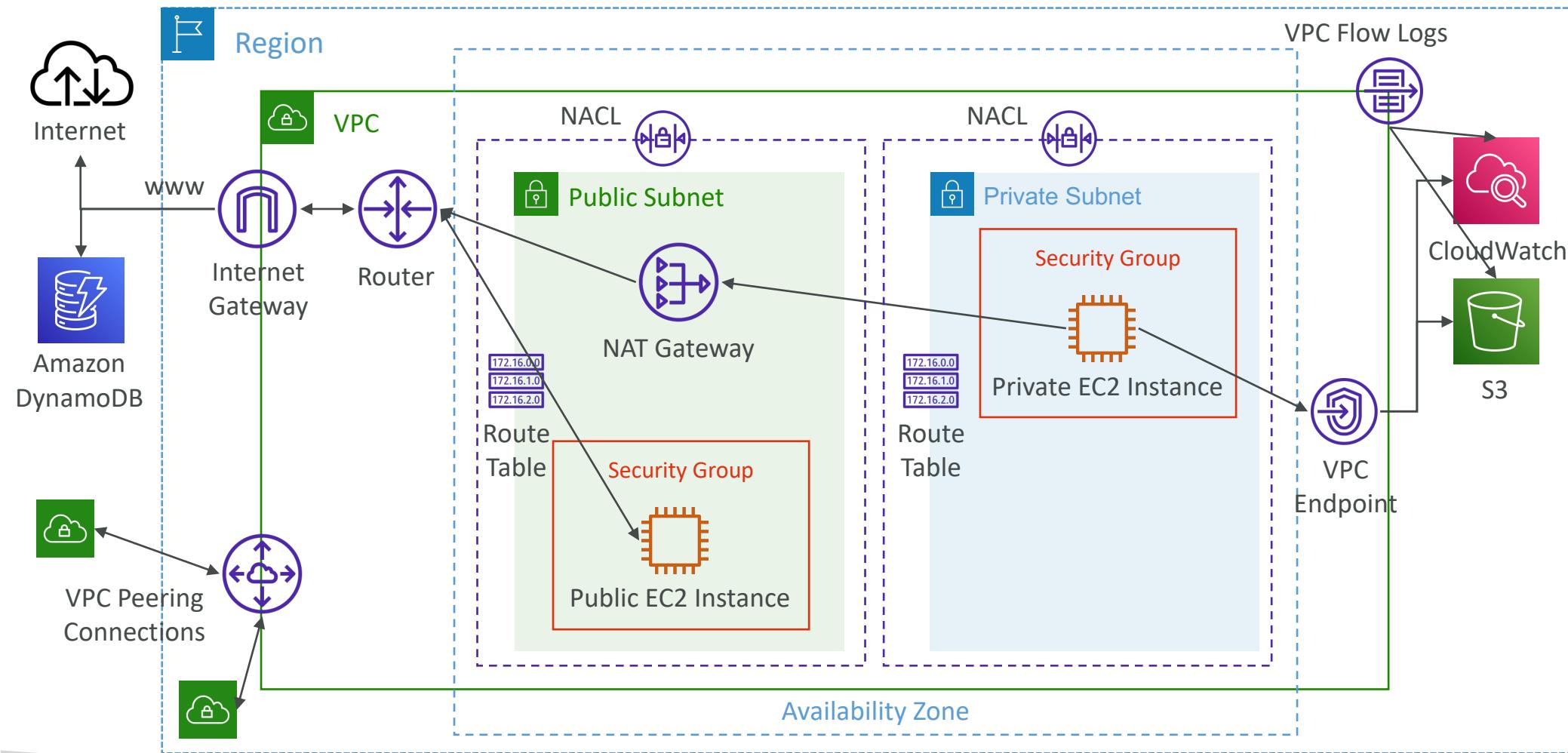




# VPC Flow Logs

- Capture information about IP traffic going into your interfaces:
  - VPC Flow Logs
  - Subnet Flow Logs
  - Elastic Network Interface (ENI) Flow Logs
- Helps to monitor & troubleshoot connectivity issues
- Flow logs data can go to S3, CloudWatch Logs, and Kinesis Data Firehose
- Captures network information from AWS managed interfaces too: ELB, RDS, ElastiCache, Redshift, WorkSpaces, NATGW, Transit Gateway...

# VPC Flow Logs



# VPC Flow Logs Syntax

version	interface-id	dstaddr	dstport	packets	start	action
2	123456789010	eni-1235b8ca123456789	172.31.16.139	172.31.16.21	20641	ACCEPT OK
2	123456789010	eni-1235b8ca123456789	172.31.9.69	172.31.9.12	49761	REJECT OK
account-id	srcaddr	srcport	protocol	bytes	end	log-status

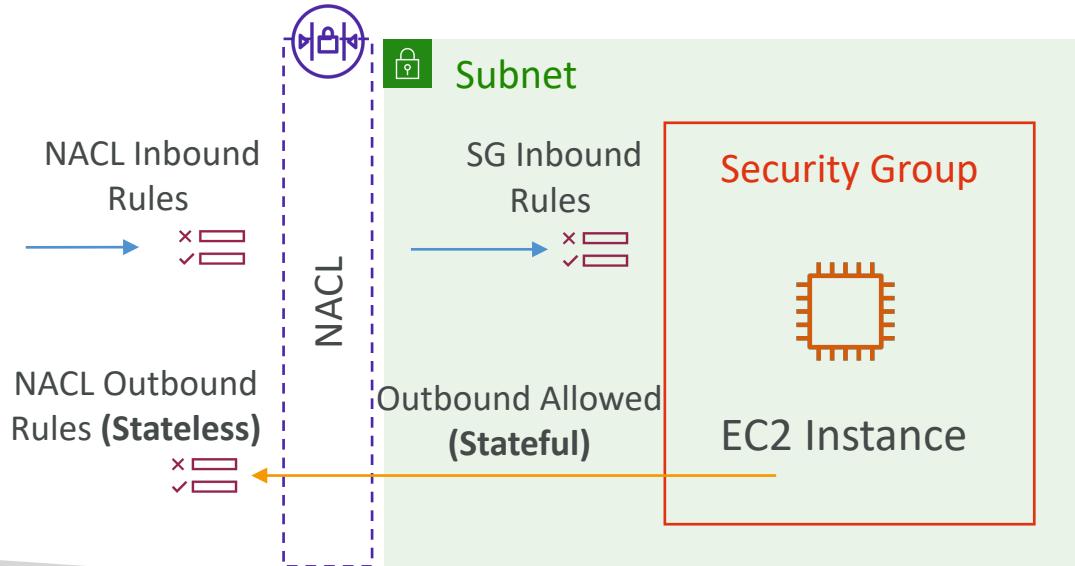
- srcaddr & dstaddr – help identify problematic IP
- srcport & dstport – help identify problematic ports
- Action – success or failure of the request due to Security Group / NACL
- Can be used for analytics on usage patterns, or malicious behavior
- Query VPC flow logs using Athena on S3 or CloudWatch Logs Insights
- Flow Logs examples: <https://docs.aws.amazon.com/vpc/latest/userguide/flow-logs-records-examples.html>

# VPC Flow Logs – Troubleshoot SG & NACL issues

Look at the “ACTION” field

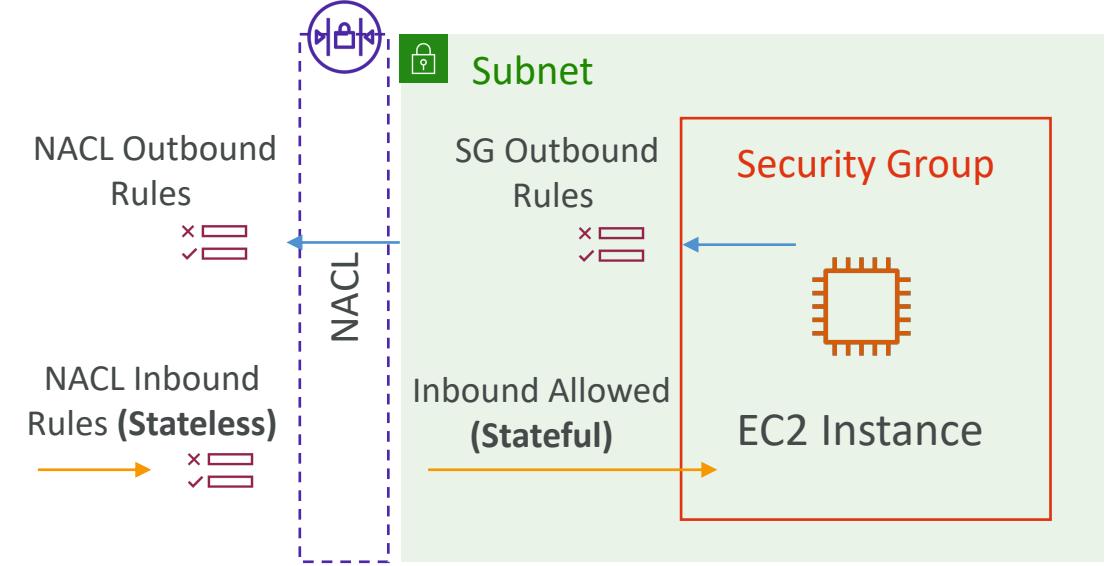
## Incoming Requests

- Inbound REJECT => NACL or SG
- Inbound ACCEPT, Outbound REJECT => NACL

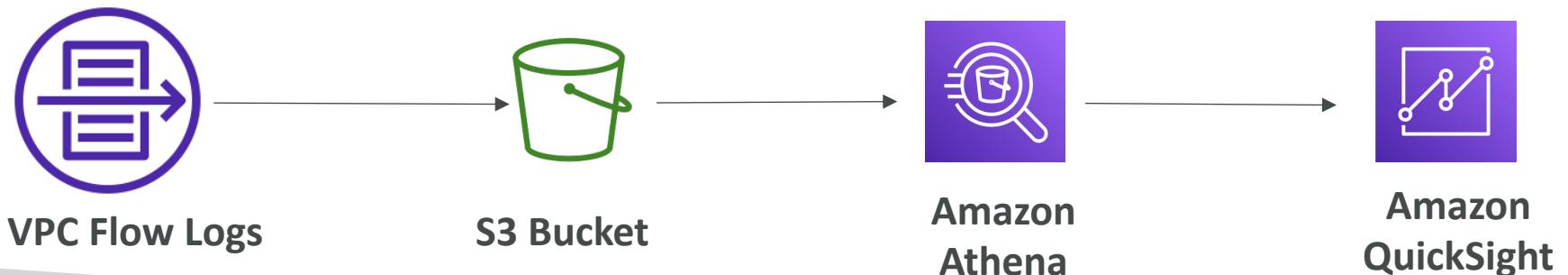
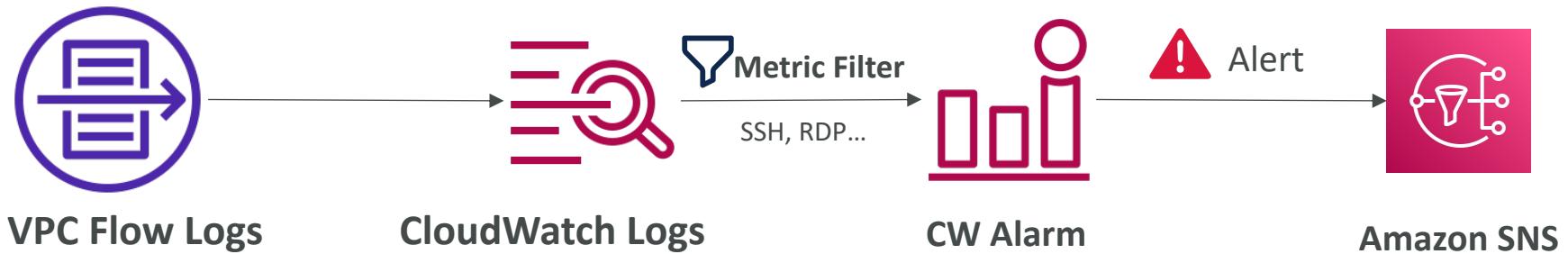


## Outgoing Requests

- Outbound REJECT => NACL or SG
- Outbound ACCEPT, Inbound REJECT => NACL



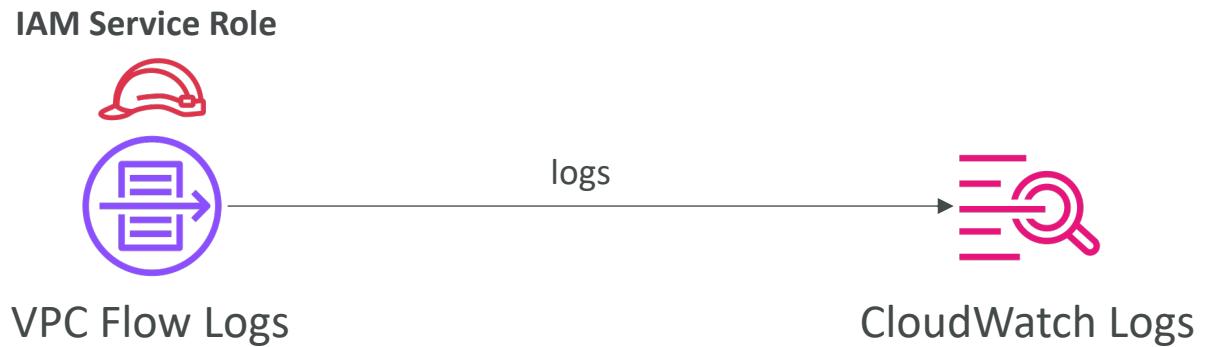
# VPC Flow Logs – Architectures



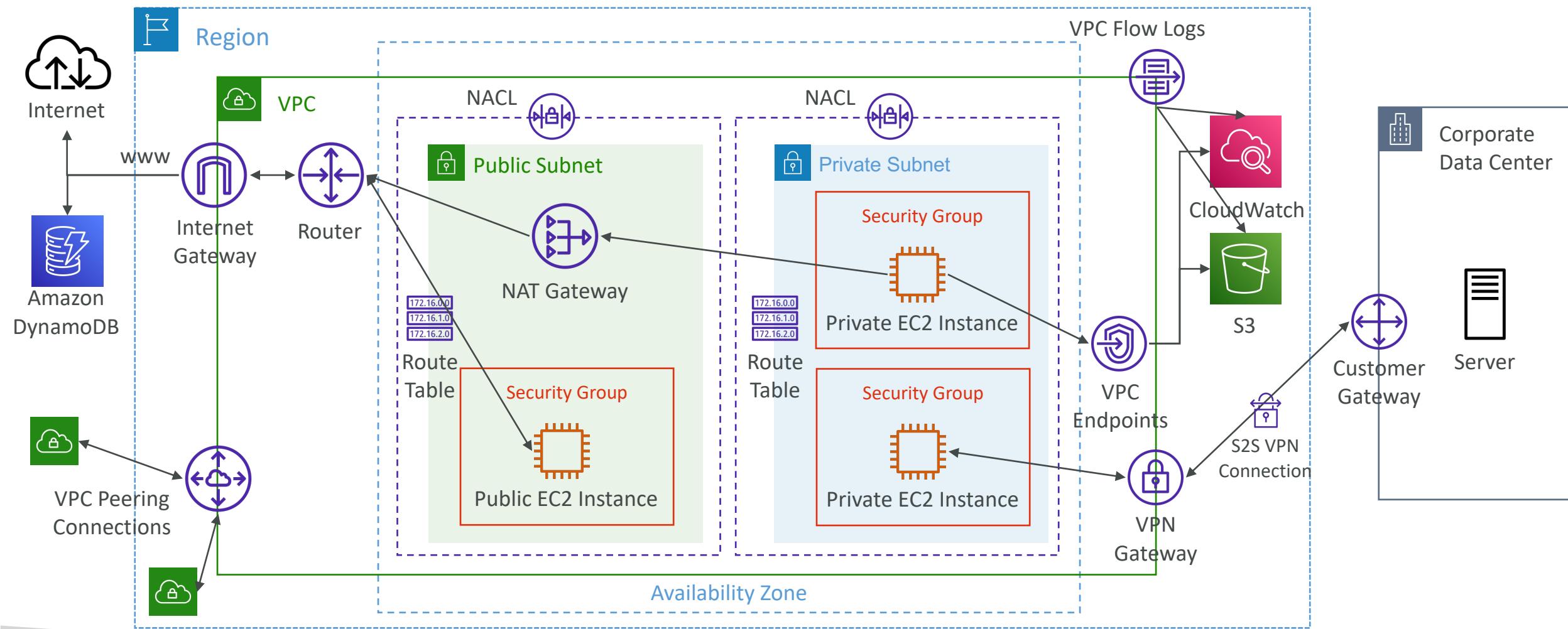
# VPC Flow Logs – CloudWatch Permissions

- IAM Service Role associated with VPC Flow Logs must have the required permissions to publish logs to CloudWatch Logs
- logs:CreateLogGroup, logs:CreateLogStream, or logs:PutLogEvents

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "logs:CreateLogGroup",  
        "logs:CreateLogStream",  
        "logs:PutLogEvents",  
        "logs:DescribeLogGroups",  
        "logs:DescribeLogStreams"  
      ],  
      "Resource": "*"  
    }  
  ]  
}
```



# AWS Site-to-Site VPN



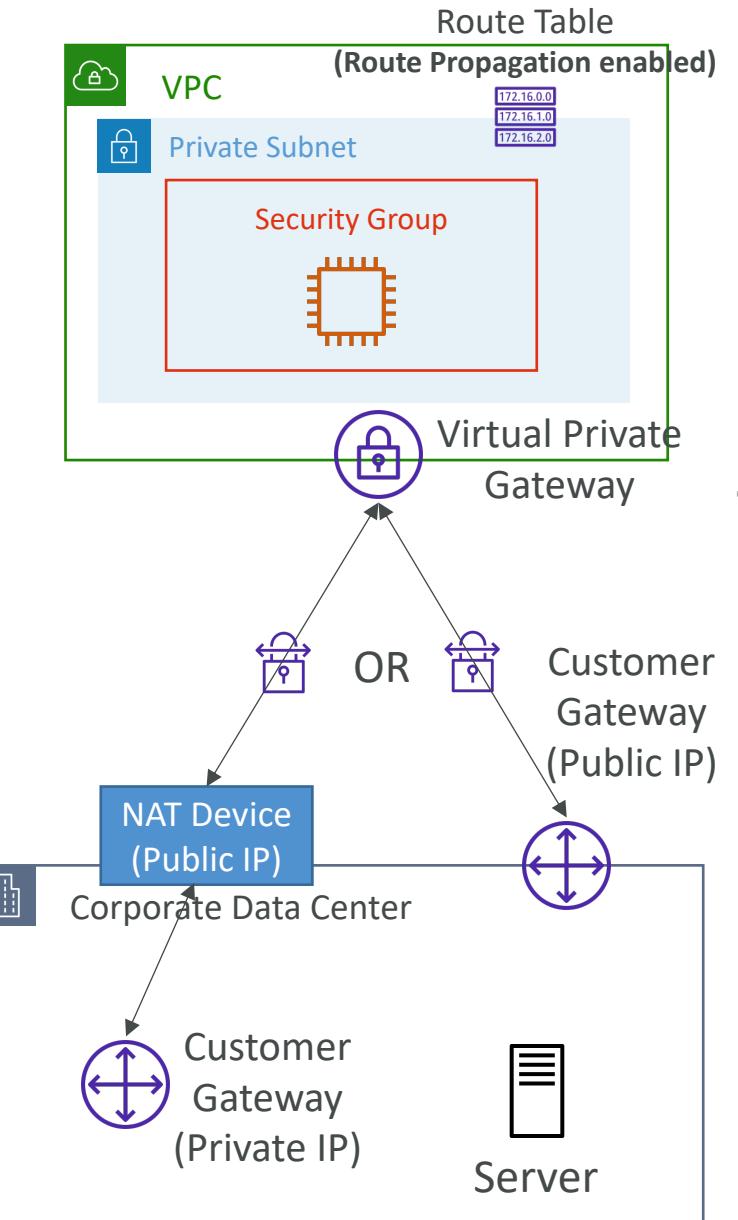
# AWS Site-to-Site VPN



- **Virtual Private Gateway (VGW)**
  - VPN concentrator on the AWS side of the VPN connection
  - VGW is created and attached to the VPC from which you want to create the Site-to-Site VPN connection
  - Possibility to customize the ASN (Autonomous System Number)
- **Customer Gateway (CGW)**
  - Software application or physical device on customer side of the VPN connection
  - <https://docs.aws.amazon.com/vpn/latest/s2svpn/your-cgw.html#DevicesTested>

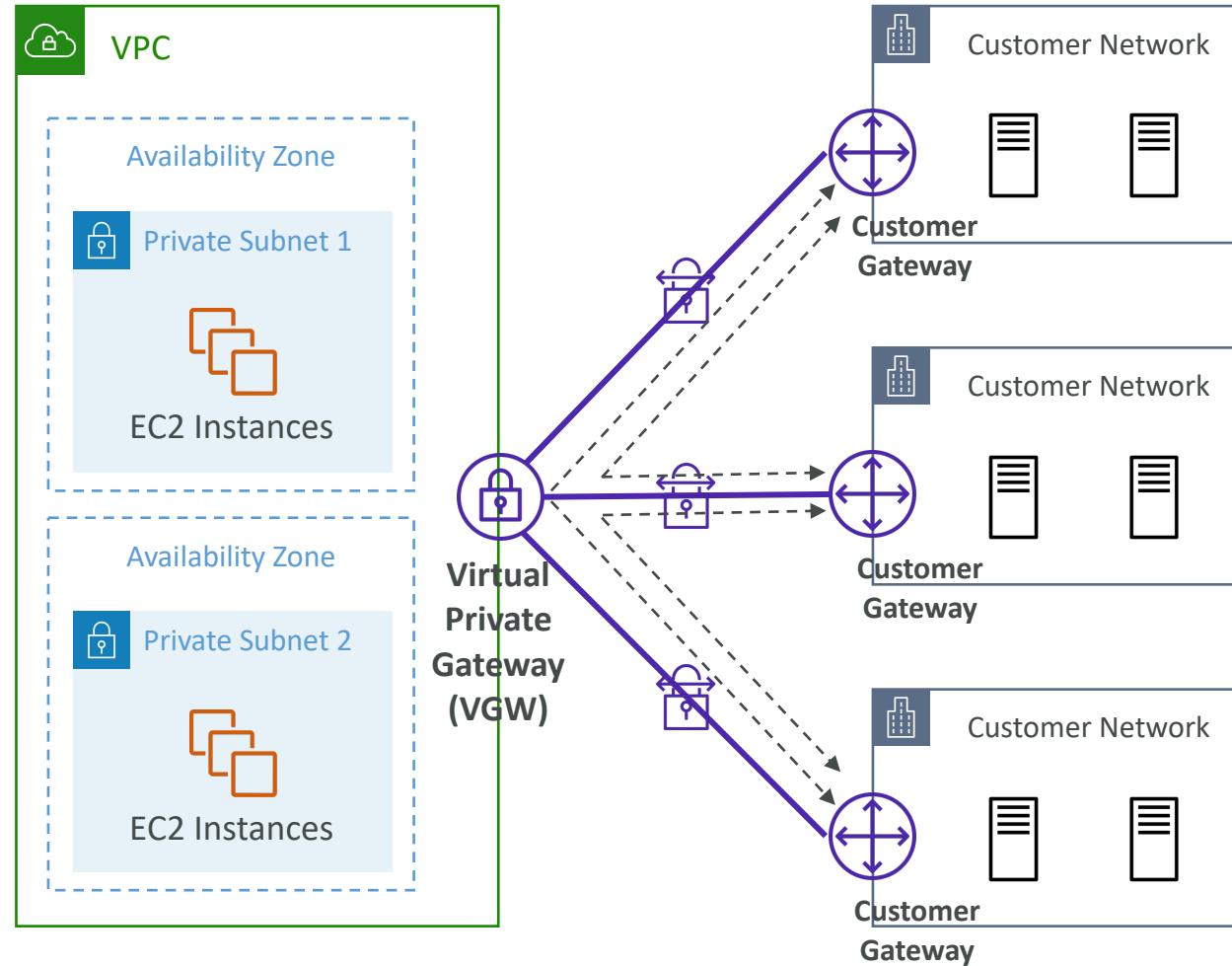
# Site-to-Site VPN Connections

- Customer Gateway Device (On-premises)
  - What IP address to use?
    - Public Internet-routable IP address for your Customer Gateway device
    - If it's behind a NAT device that's enabled for NAT traversal (NAT-T), use the public IP address of the NAT device
- Important step: enable Route Propagation for the Virtual Private Gateway in the route table that is associated with your subnets
- If you need to ping your EC2 instances from on-premises, make sure you add the ICMP protocol on the inbound of your security groups



# AWS VPN CloudHub

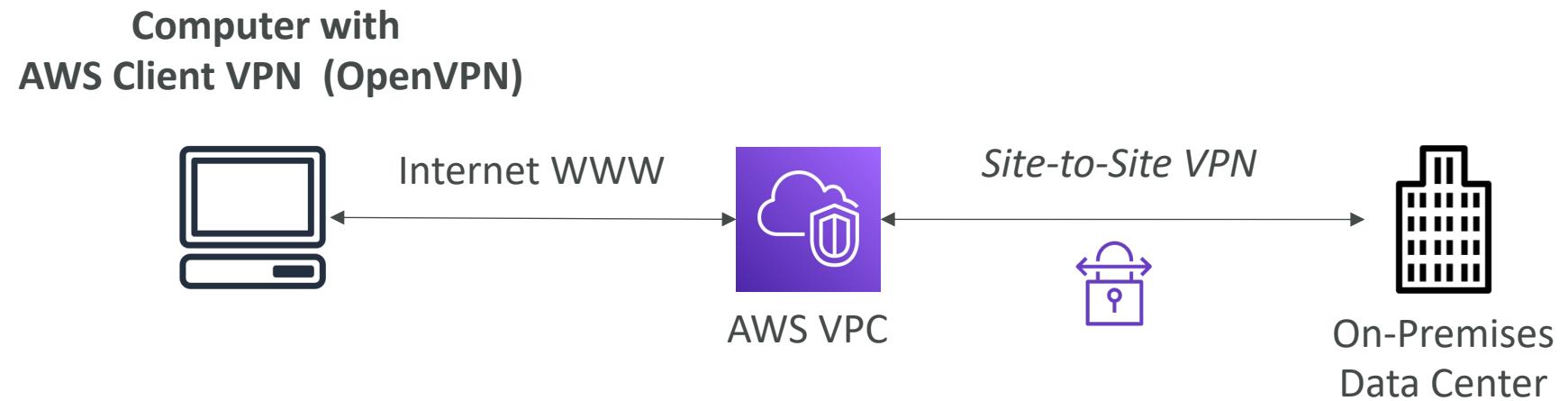
- Provide secure communication between multiple sites, if you have multiple VPN connections
- Low-cost hub-and-spoke model for primary or secondary network connectivity between different locations (VPN only)
- It's a VPN connection so it goes over the public Internet
- To set it up, connect multiple VPN connections on the same VGW, setup dynamic routing and configure route tables



# AWS Client VPN



- Connect from your computer using OpenVPN to your private network in AWS and on-premises
- Allow you to connect to your EC2 instances over a private IP (just as if you were in the private VPC network)
- Goes over public Internet

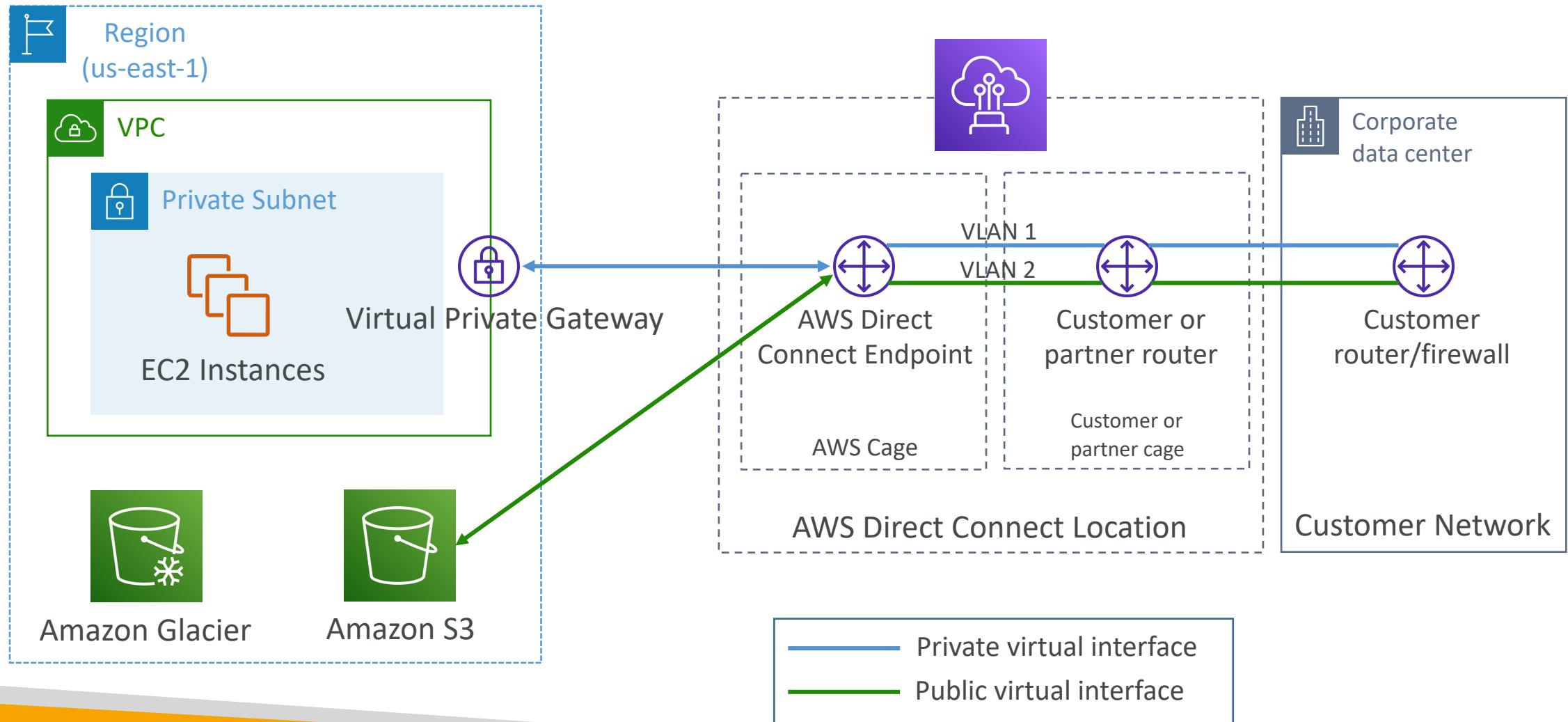




# Direct Connect (DX)

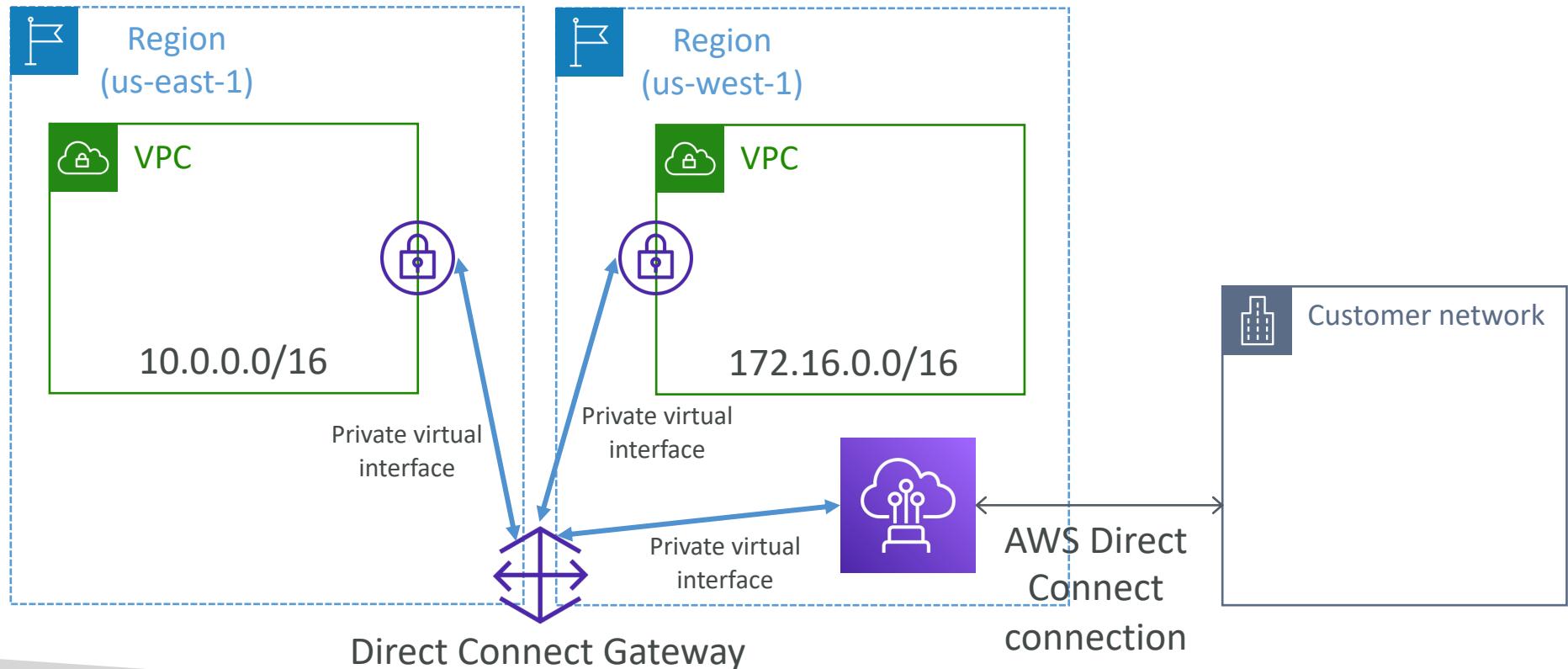
- Provides a dedicated private connection from a remote network to your VPC
- Dedicated connection must be setup between your DC and AWS Direct Connect locations
- You need to setup a Virtual Private Gateway on your VPC
- Access public resources (S3) and private (EC2) on same connection
- Use Cases:
  - Increase bandwidth throughput - working with large data sets – lower cost
  - More consistent network experience - applications using real-time data feeds
  - Hybrid Environments (on prem + cloud)
- Supports both IPv4 and IPv6

# Direct Connect Diagram



# Direct Connect Gateway

- If you want to setup a Direct Connect to one or more VPC in many different regions (same account), you must use a Direct Connect Gateway

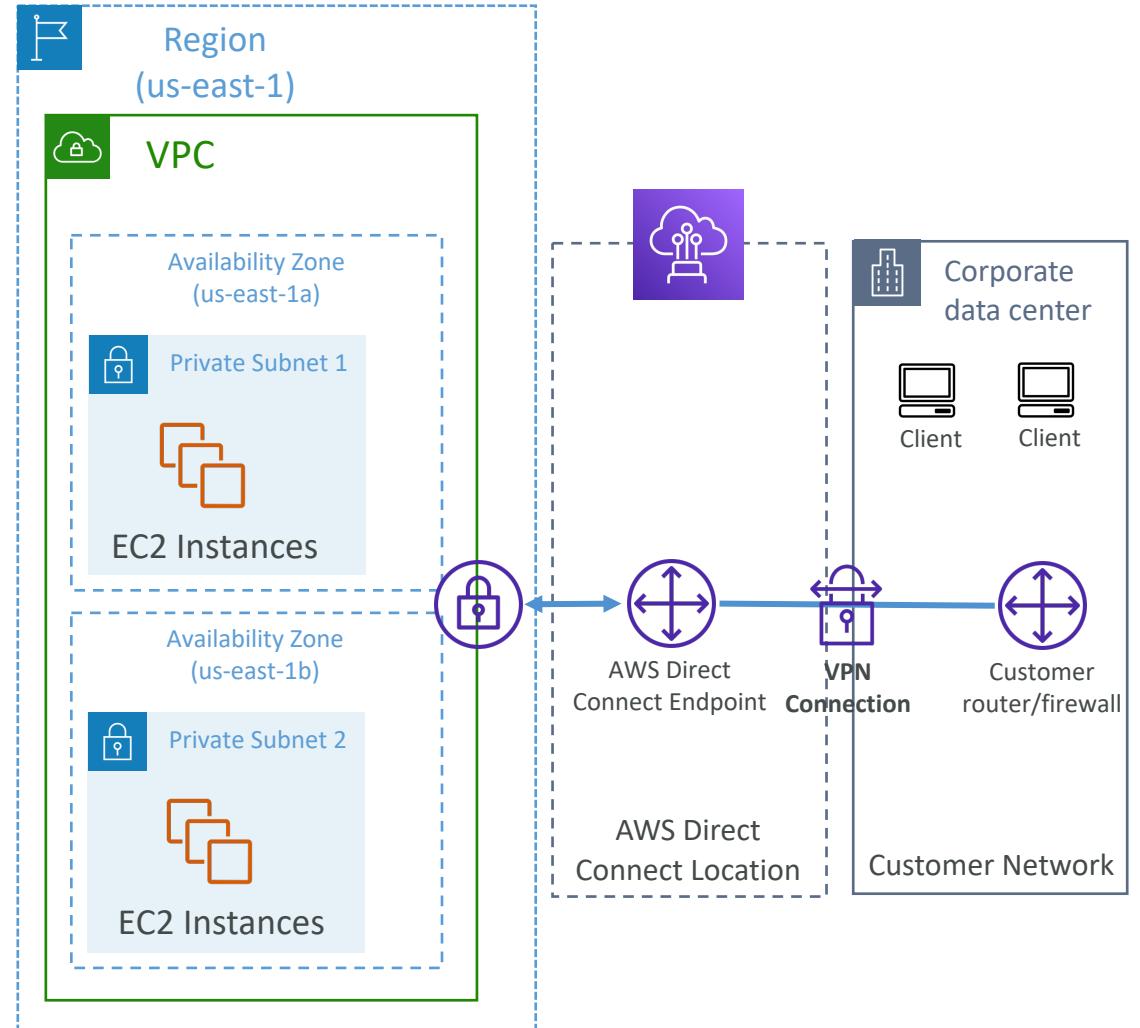


# Direct Connect – Connection Types

- **Dedicated Connections:** 1 Gbps, 10 Gbps and 100 Gbps capacity
  - Physical ethernet port dedicated to a customer
  - Request made to AWS first, then completed by AWS Direct Connect Partners
- **Hosted Connections:** 50Mbps, 500 Mbps, to 10 Gbps
  - Connection requests are made via AWS Direct Connect Partners
  - Capacity can be **added or removed on demand**
  - 1, 2, 5, 10 Gbps available at select AWS Direct Connect Partners
- Lead times are often longer than 1 month to establish a new connection

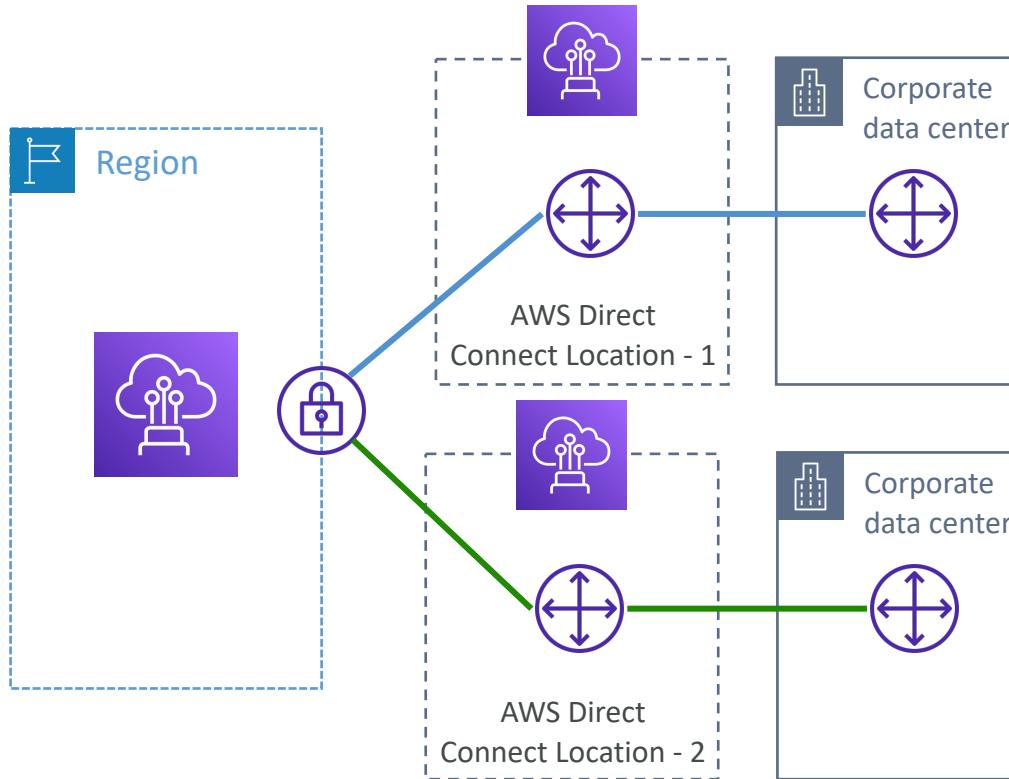
# Direct Connect – Encryption

- Data in transit is not encrypted but is private
- AWS Direct Connect + VPN provides an IPsec-encrypted private connection
- Good for an extra level of security, but slightly more complex to put in place



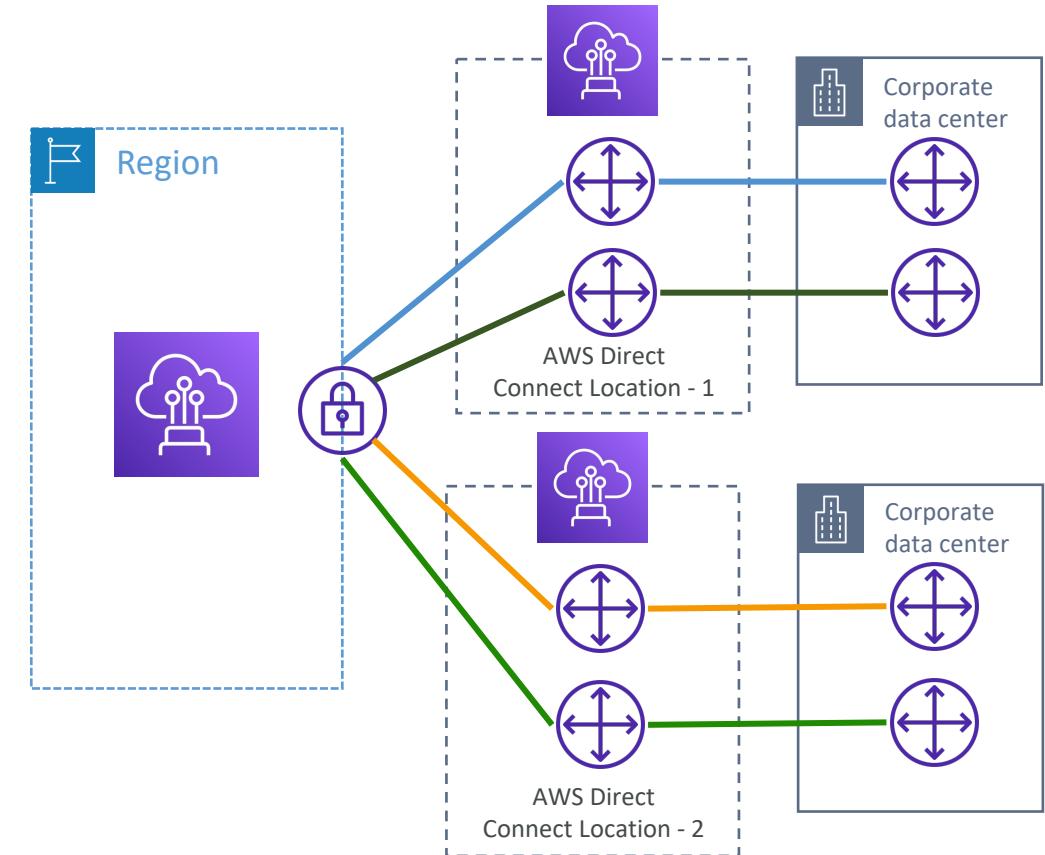
# Direct Connect - Resiliency

High Resiliency for Critical Workloads



One connection at multiple locations

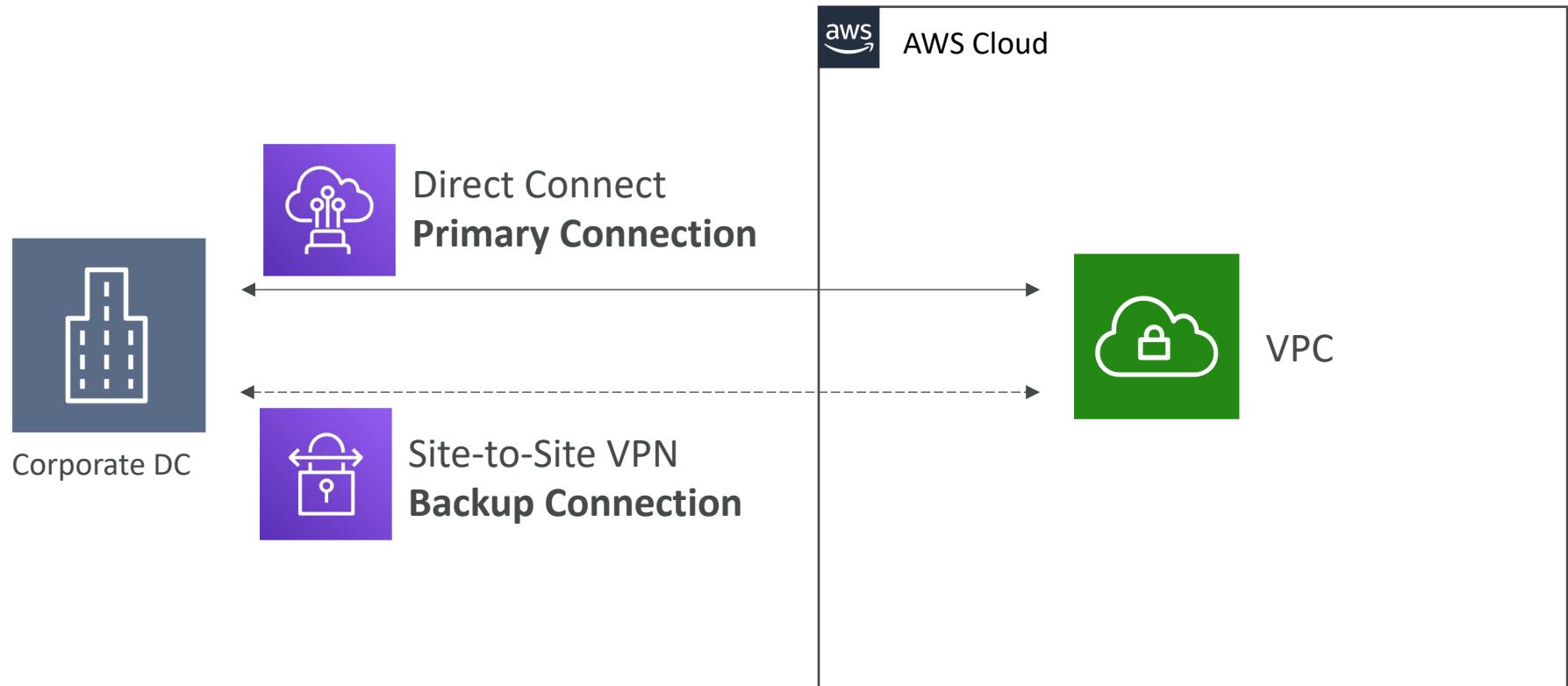
Maximum Resiliency for Critical Workloads



Maximum resilience is achieved by separate connections terminating on separate devices in more than one location.

# Site-to-Site VPN connection as a backup

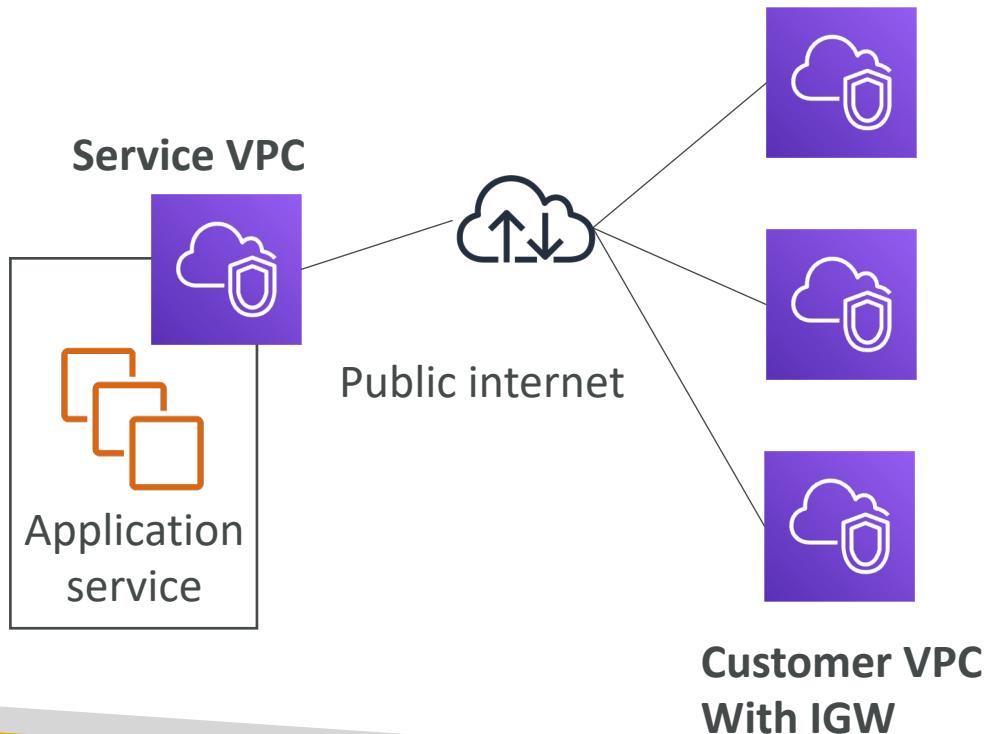
- In case Direct Connect fails, you can set up a backup Direct Connect connection (expensive), or a Site-to-Site VPN connection



# Exposing services in your VPC to other VPC

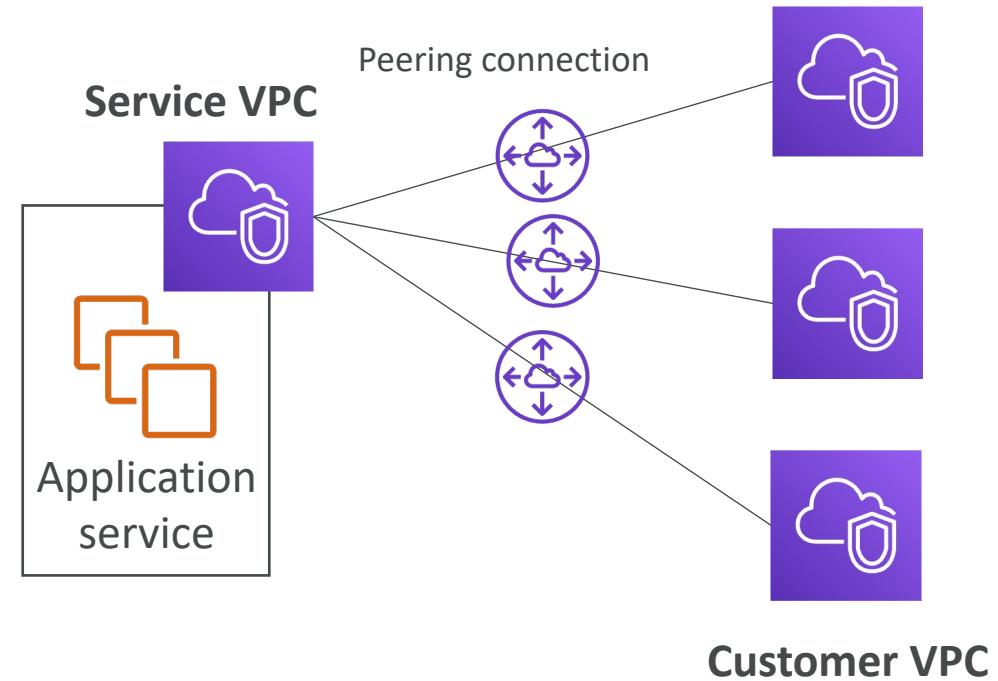
- Option 1: make it public

- Goes through the public www
- Tough to manage access



- Option 2: VPC peering

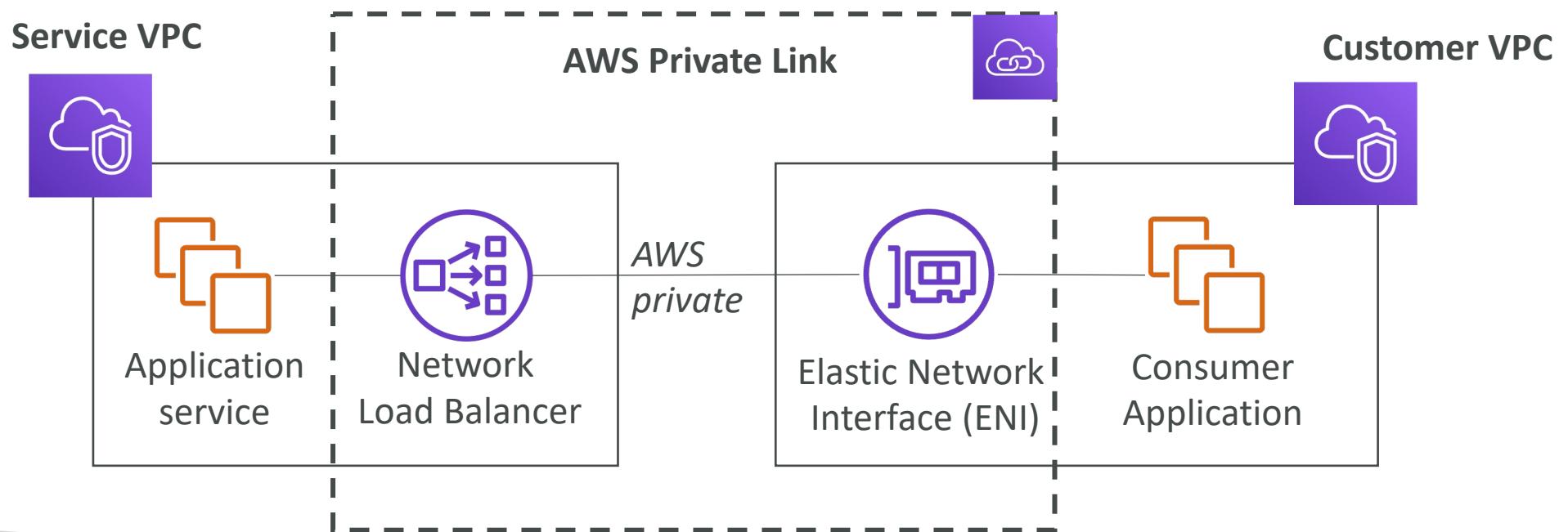
- Must create many peering relations
- Opens the **whole** network



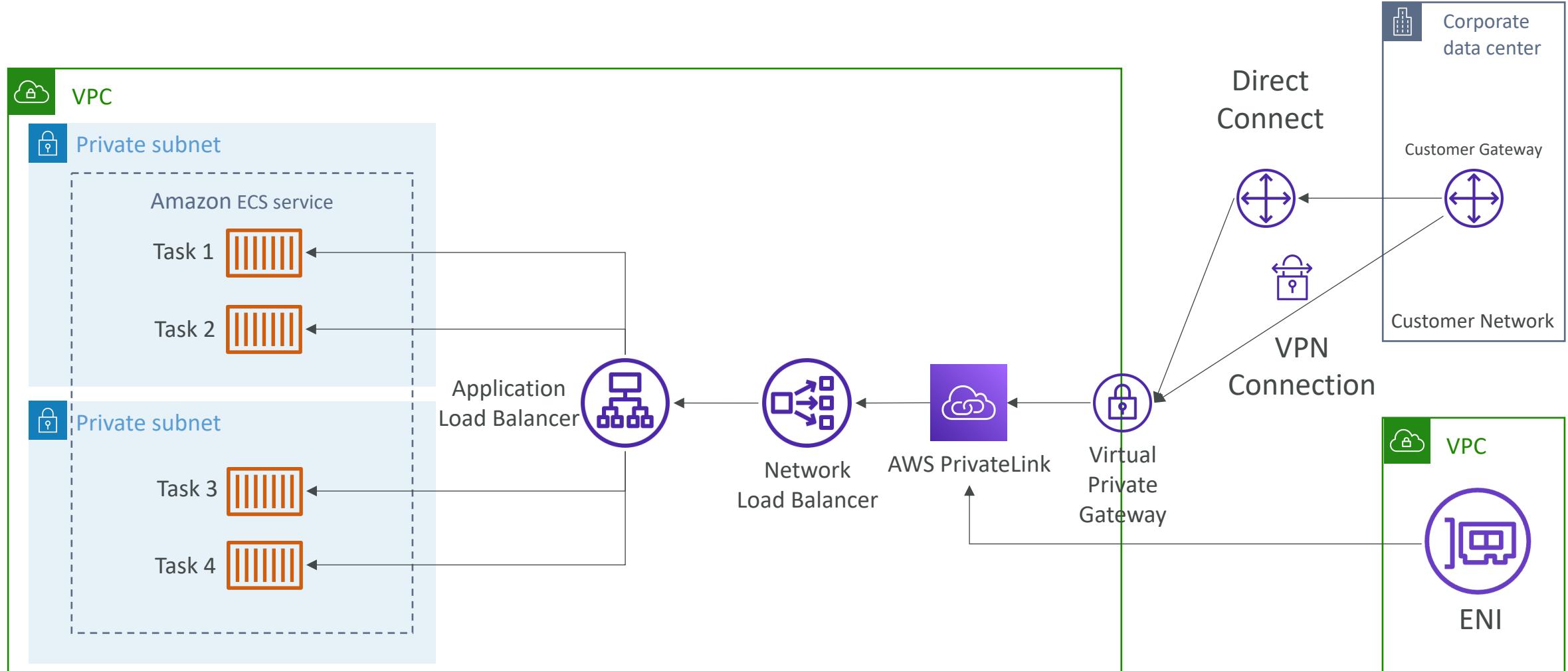
# AWS PrivateLink (VPC Endpoint Services)



- Most secure & scalable way to expose a service to 1000s of VPC (own or other accounts)
- Does not require VPC peering, internet gateway, NAT, route tables...
- Requires a network load balancer (Service VPC) and ENI (Customer VPC) or GWLB
- If the NLB is in multiple AZ, and the ENIs in multiple AZ, the solution is fault tolerant!



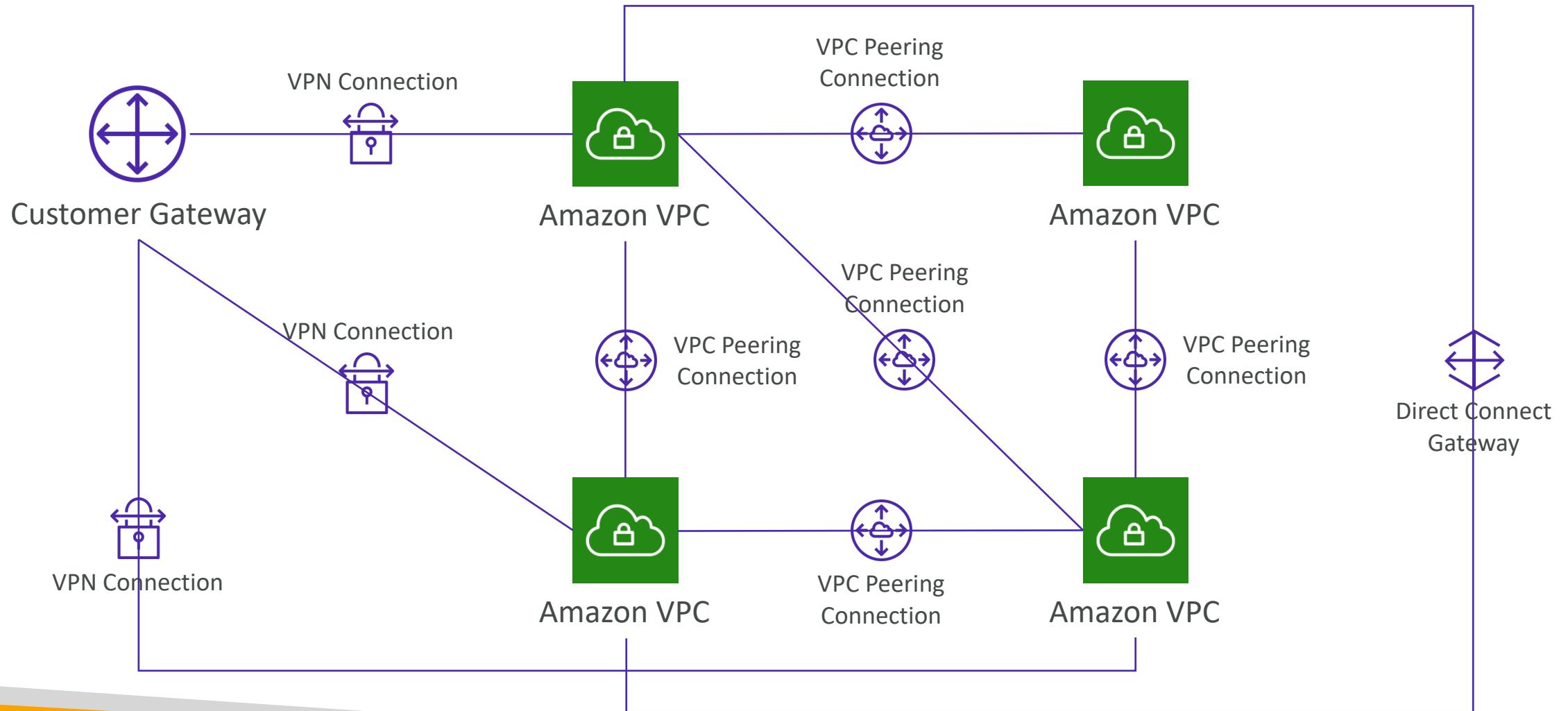
# AWS Private Link & ECS



# EC2-Classic & AWS ClassicLink (deprecated)

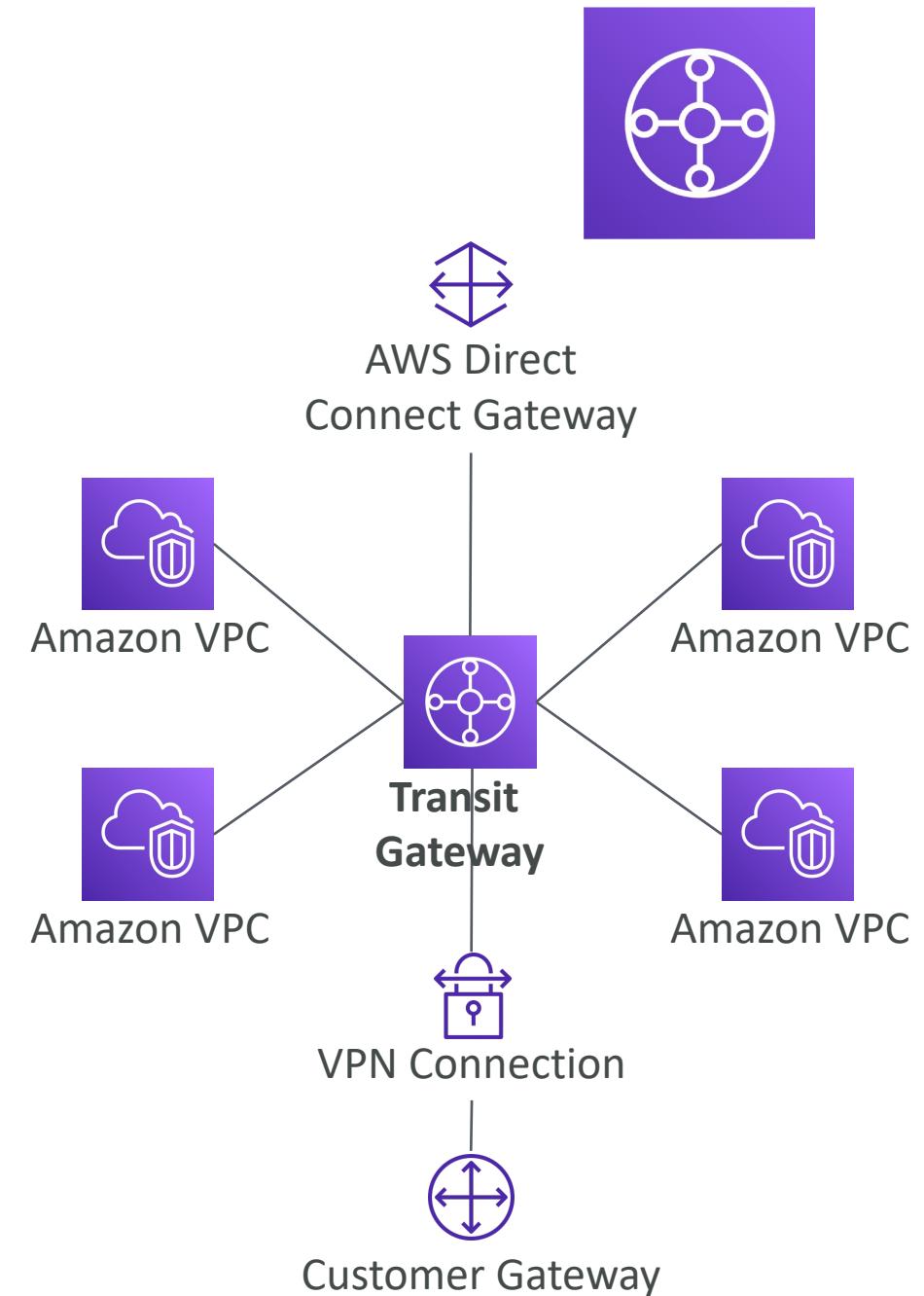
- **EC2-Classic:** instances run in a single network shared with other customers
- **Amazon VPC:** your instances run logically isolated to your AWS account
- **ClassicLink** allows you to link EC2-Classic instances to a VPC in your account
  - Must associate a security group
  - Enables communication using private IPv4 addresses
  - Removes the need to make use of public IPv4 addresses or Elastic IP addresses
- Likely to be distractors at the exam

# Network topologies can become complicated



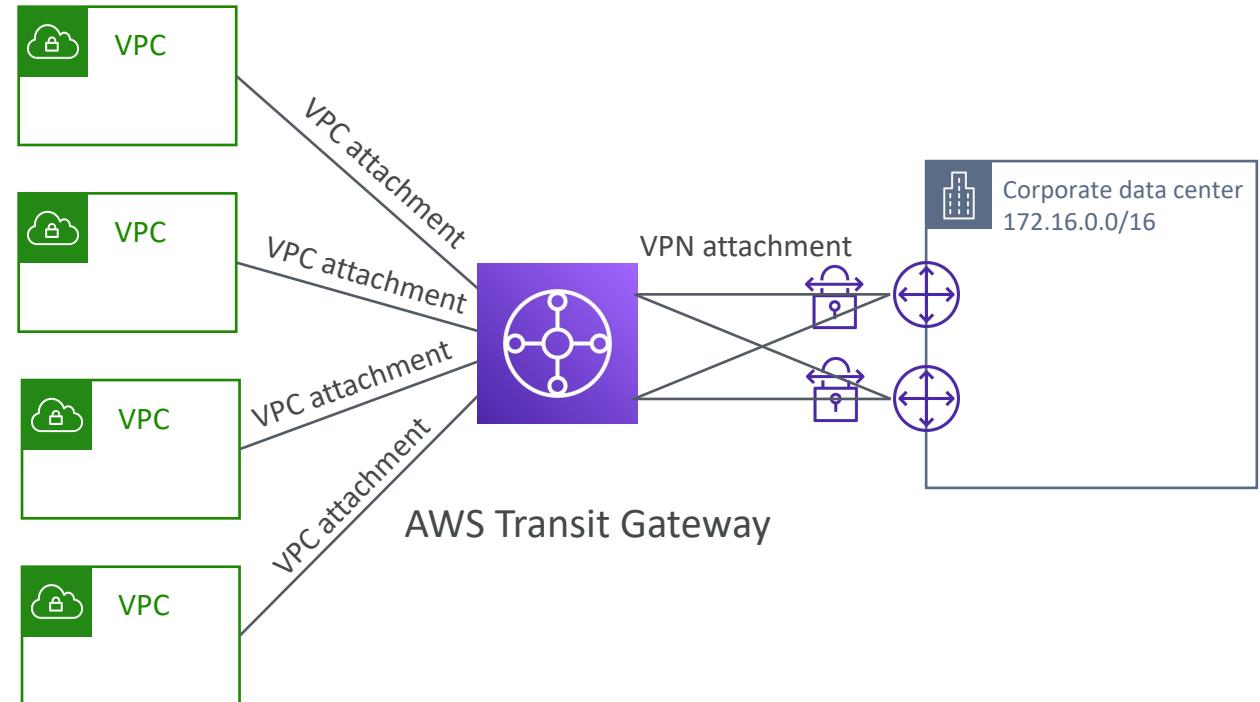
# Transit Gateway

- For having transitive peering between thousands of VPC and on-premises, hub-and-spoke (star) connection
- Regional resource, can work cross-region
- Share cross-account using Resource Access Manager (RAM)
- You can peer Transit Gateways across regions
- Route Tables: limit which VPC can talk with other VPC
- Works with Direct Connect Gateway, VPN connections
- Supports IP Multicast (not supported by any other AWS service)



# Transit Gateway: Site-to-Site VPN ECMP

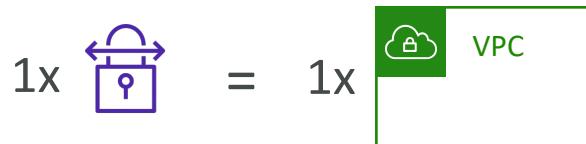
- ECMP = Equal-cost multi-path routing
- Routing strategy to allow to forward a packet over multiple best path
- Use case: create multiple Site-to-Site VPN connections to increase the bandwidth of your connection to AWS



# Transit Gateway: throughput with ECMP



VPN to virtual private gateway



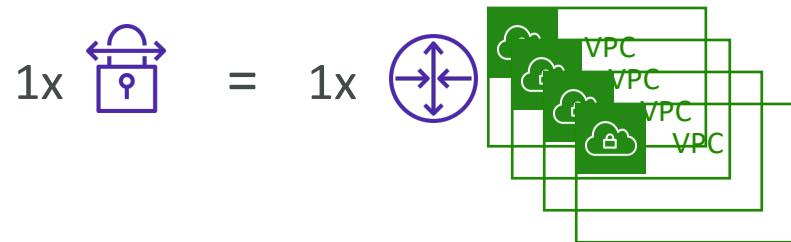
1x = 1.25 Gbps



VPN connection  
(2 tunnels)



VPN to transit gateway



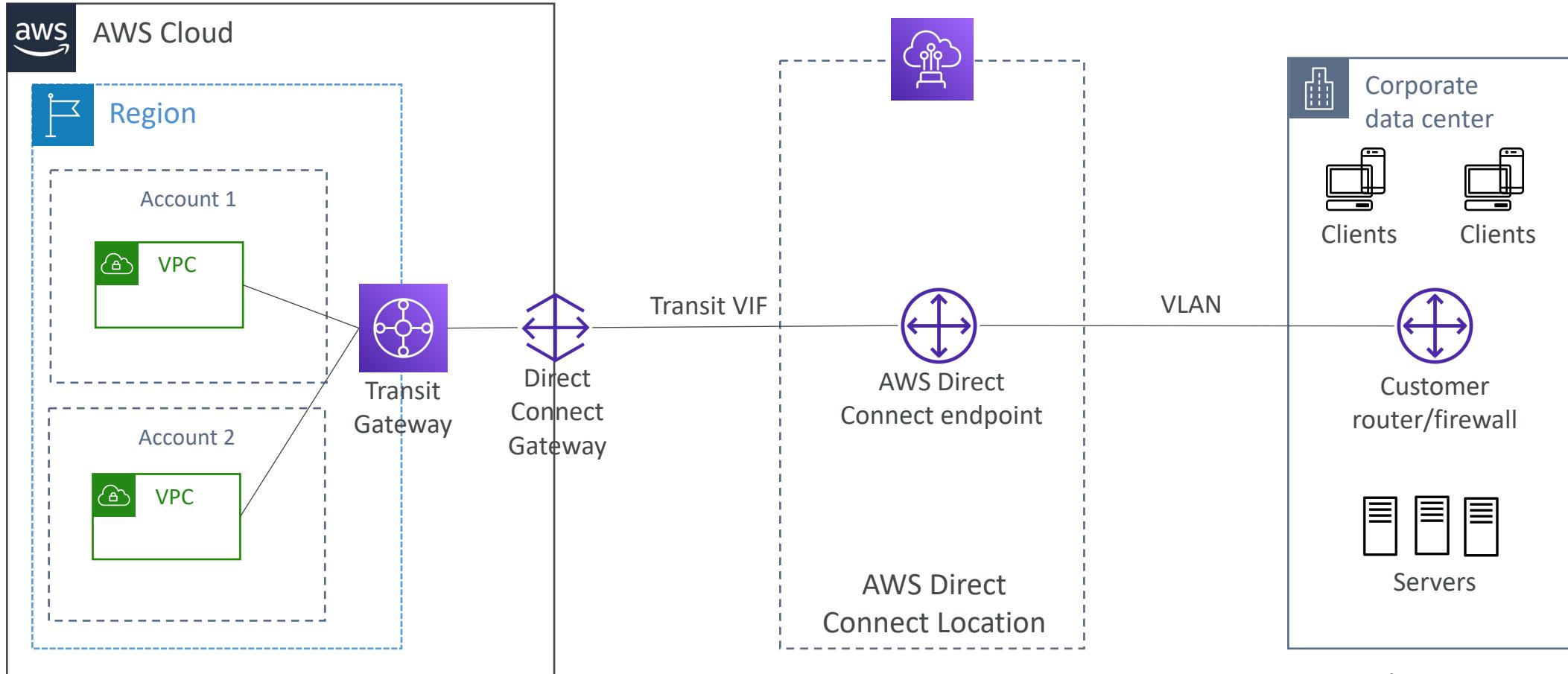
1x = 2.5 Gbps (ECMP) – 2 tunnels used

2x = 5.0 Gbps (ECMP)

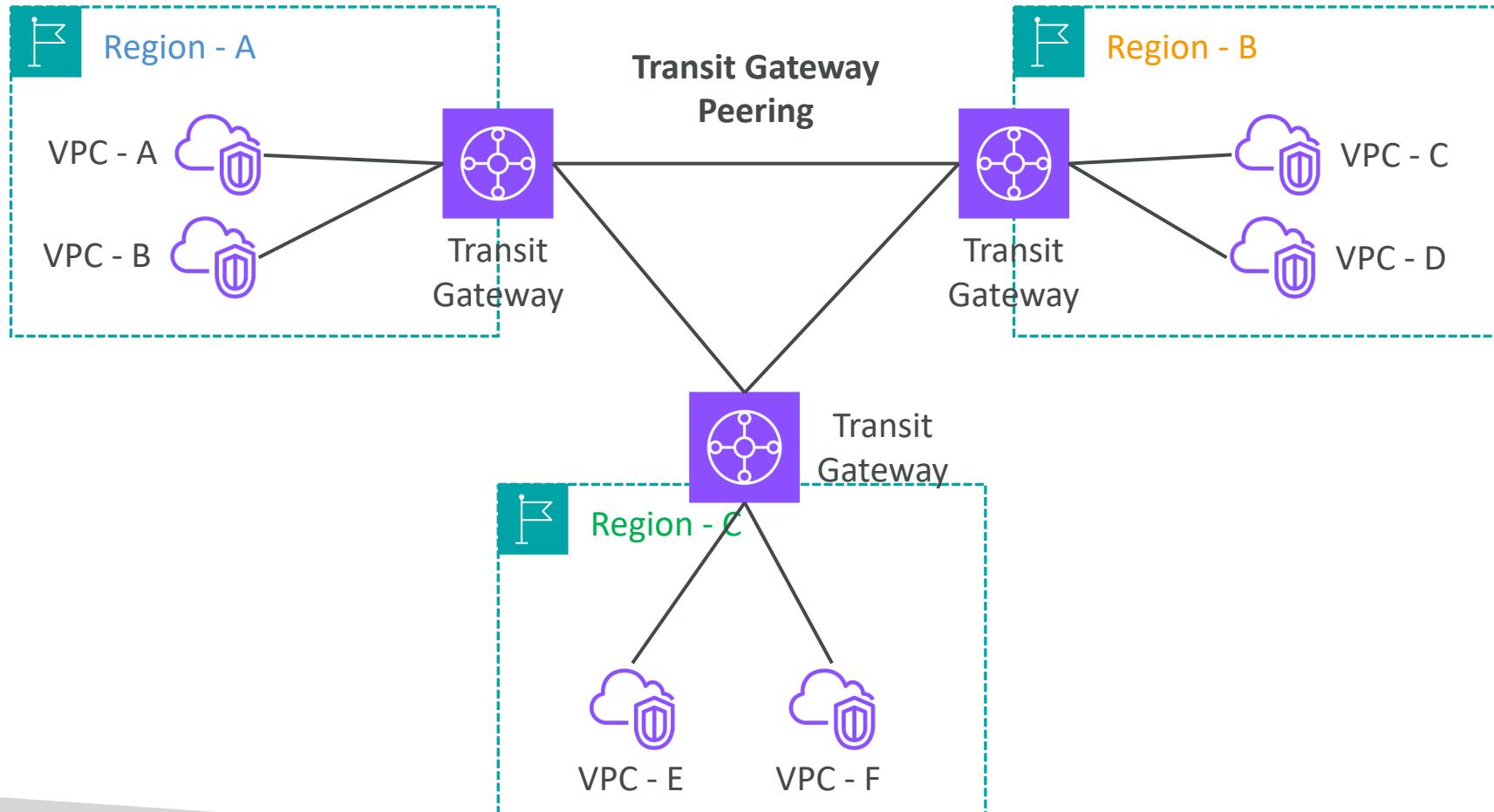
3x = 7.5 Gbps (ECMP)

+\$\$ per GB of TGW  
processed data

# Transit Gateway – Share Direct Connect between multiple accounts

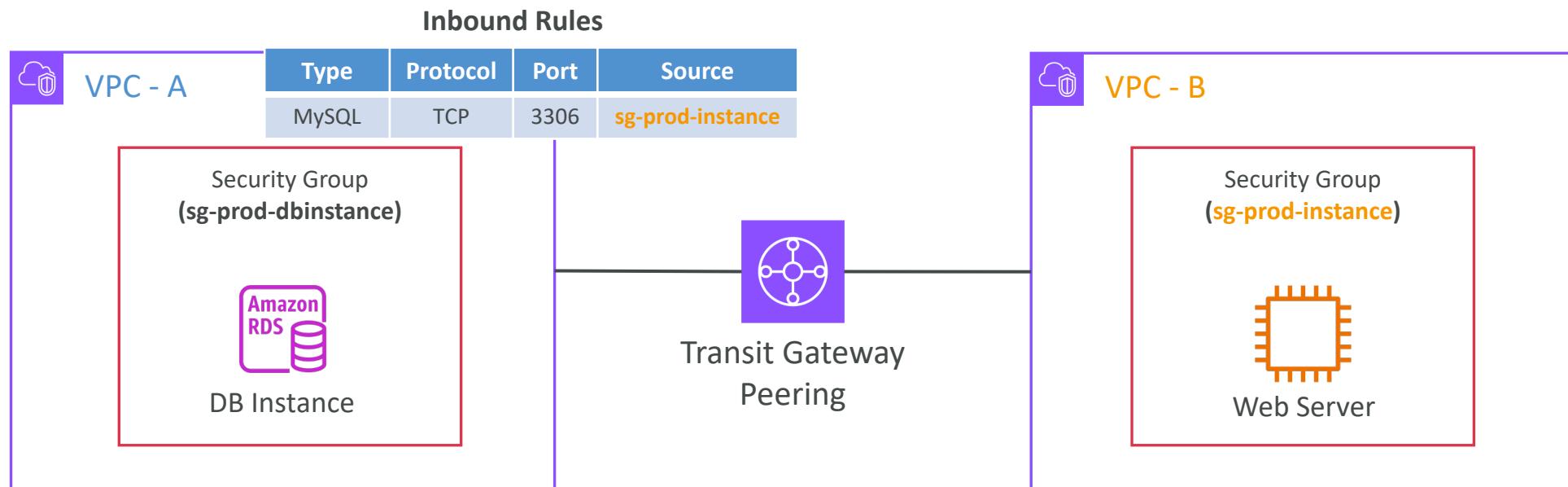


# Transit Gateway – Inter-Region Communication



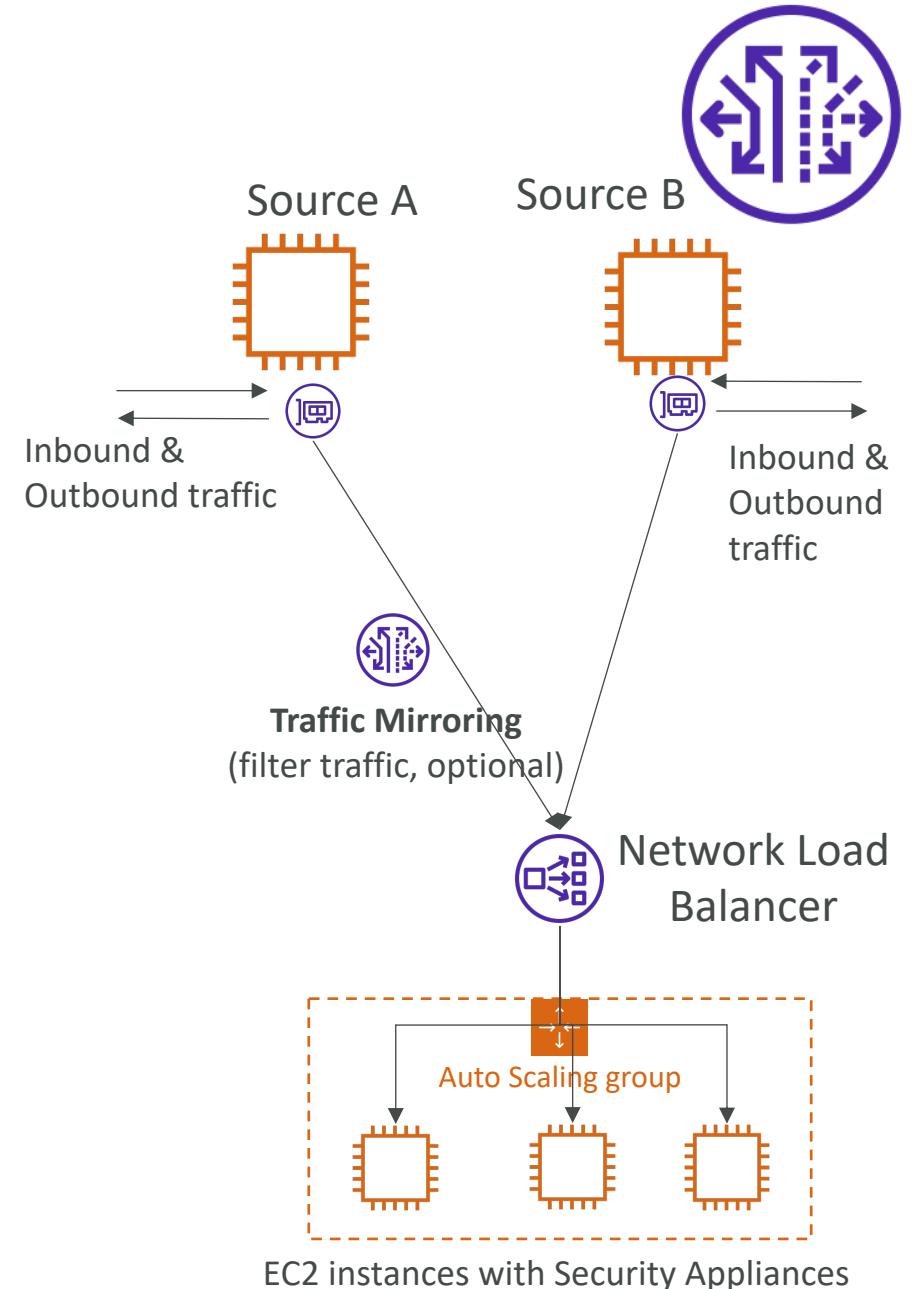
# Transit Gateway – Security Group Referencing

- You can reference a Security Group in another VPC connected using Transit Gateway
- Must enable the setting **Security Group Referencing Support** when creating the Transit Gateway Attachment



# VPC – Traffic Mirroring

- Allows you to capture and inspect network traffic in your VPC
- Route the traffic to security appliances that you manage
- Capture the traffic
  - From (Source) – ENIs
  - To (Targets) – an ENI or a Network Load Balancer
- Capture all packets or capture the packets of your interest (optionally, truncate packets)
- Source and Target can be in the same VPC or different VPCs (VPC Peering)
- Use cases: content inspection, threat monitoring, troubleshooting, ...

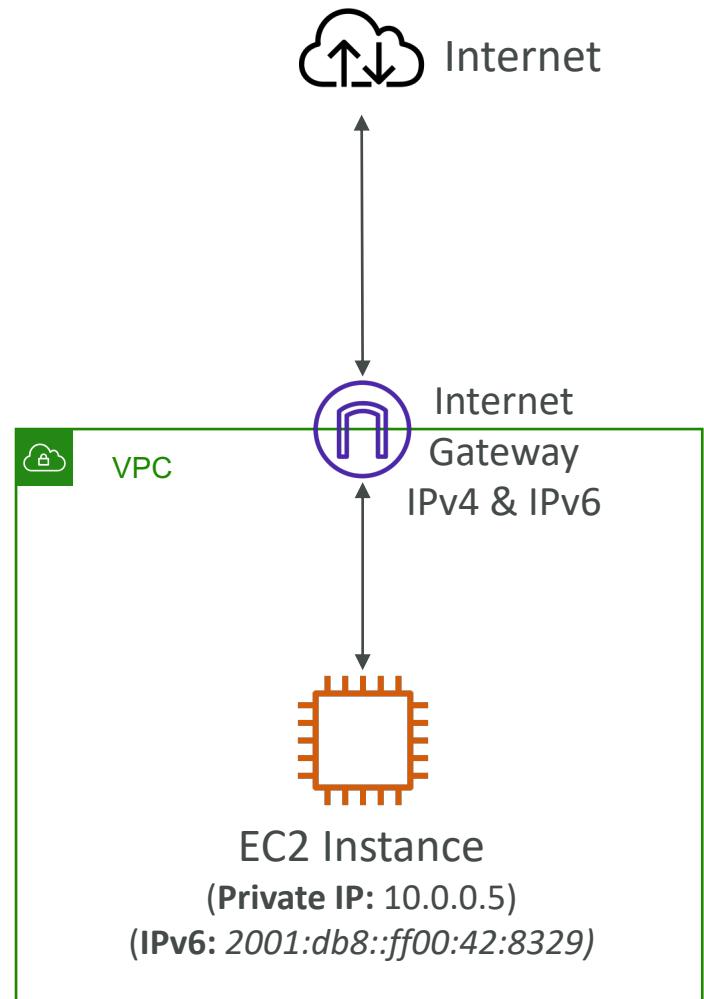


# What is IPv6?

- IPv4 designed to provide 4.3 Billion addresses (they'll be exhausted soon)
- IPv6 is the successor of IPv4
- IPv6 is designed to provide  $3.4 \times 10^{38}$  unique IP addresses
- Every IPv6 address in AWS is public and Internet-routable (no private range)
- Format → x.x.x.x.x.x.x.x (x is hexadecimal, range can be from 0000 to ffff)
- Examples:
  - 2001:db8:3333:4444:5555:6666:7777:8888
  - 2001:db8:3333:4444:cccc:dddd:eeee:ffff
  - :: → all 8 segments are zero
  - 2001:db8:: → the last 6 segments are zero
  - ::1234:5678 → the first 6 segments are zero
  - 2001:db8::1234:5678 → the middle 4 segments are zero

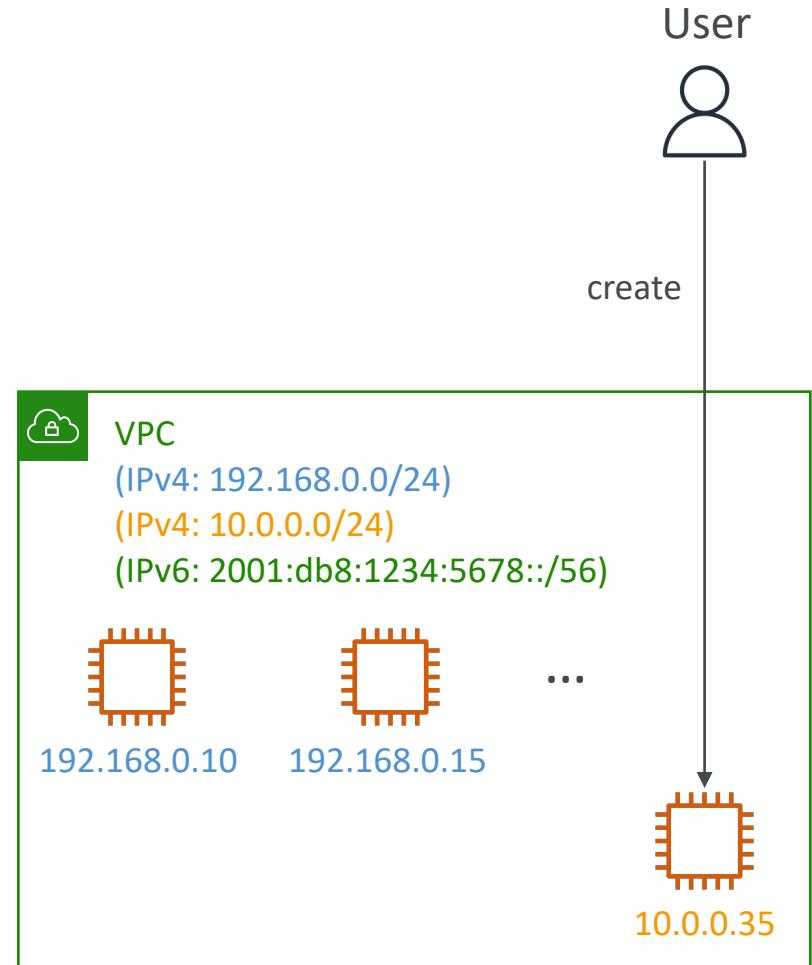
# IPv6 in VPC

- IPv4 cannot be disabled for your VPC and subnets
- You can enable IPv6 (they're public IP addresses) to operate in dual-stack mode
- Your EC2 instances will get at least a private internal IPv4 and a public IPv6
- They can communicate using either IPv4 or IPv6 to the internet through an Internet Gateway

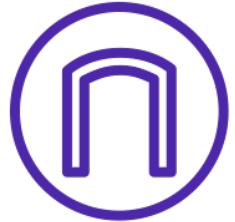


# IPv4 Troubleshooting

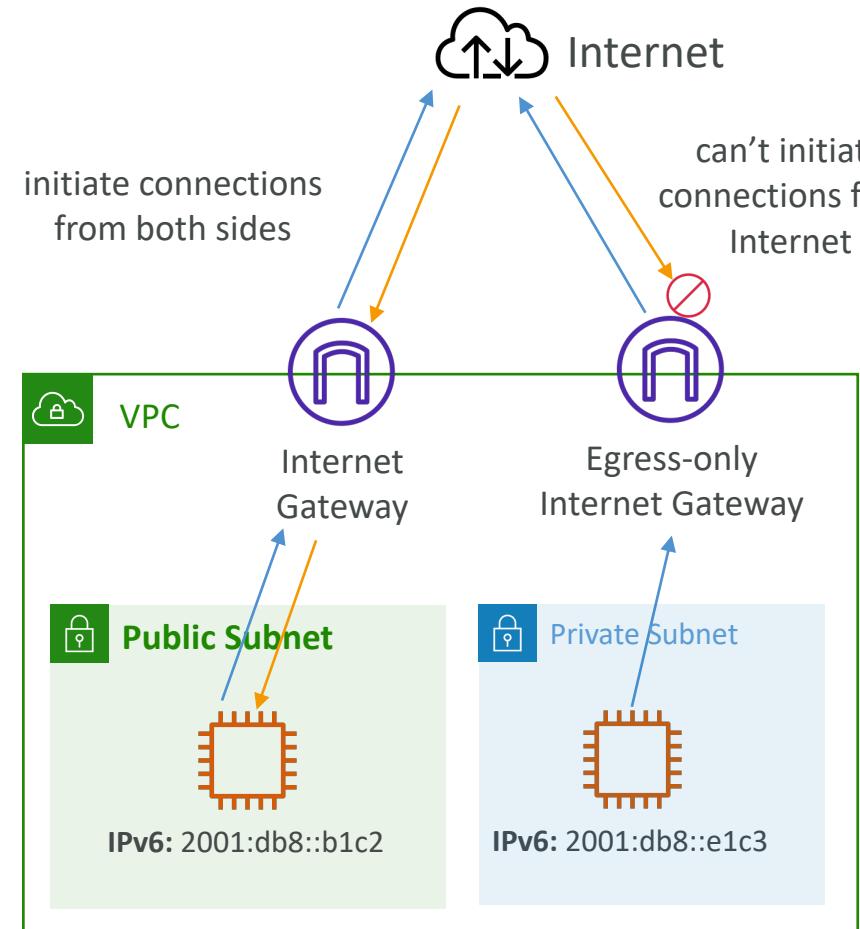
- IPv4 cannot be disabled for your VPC and subnets
- So, if you cannot launch an EC2 instance in your subnet
  - It's not because it cannot acquire an IPv6 (the space is very large)
  - It's because there are no available IPv4 in your subnet
- Solution: create a new IPv4 CIDR in your subnet



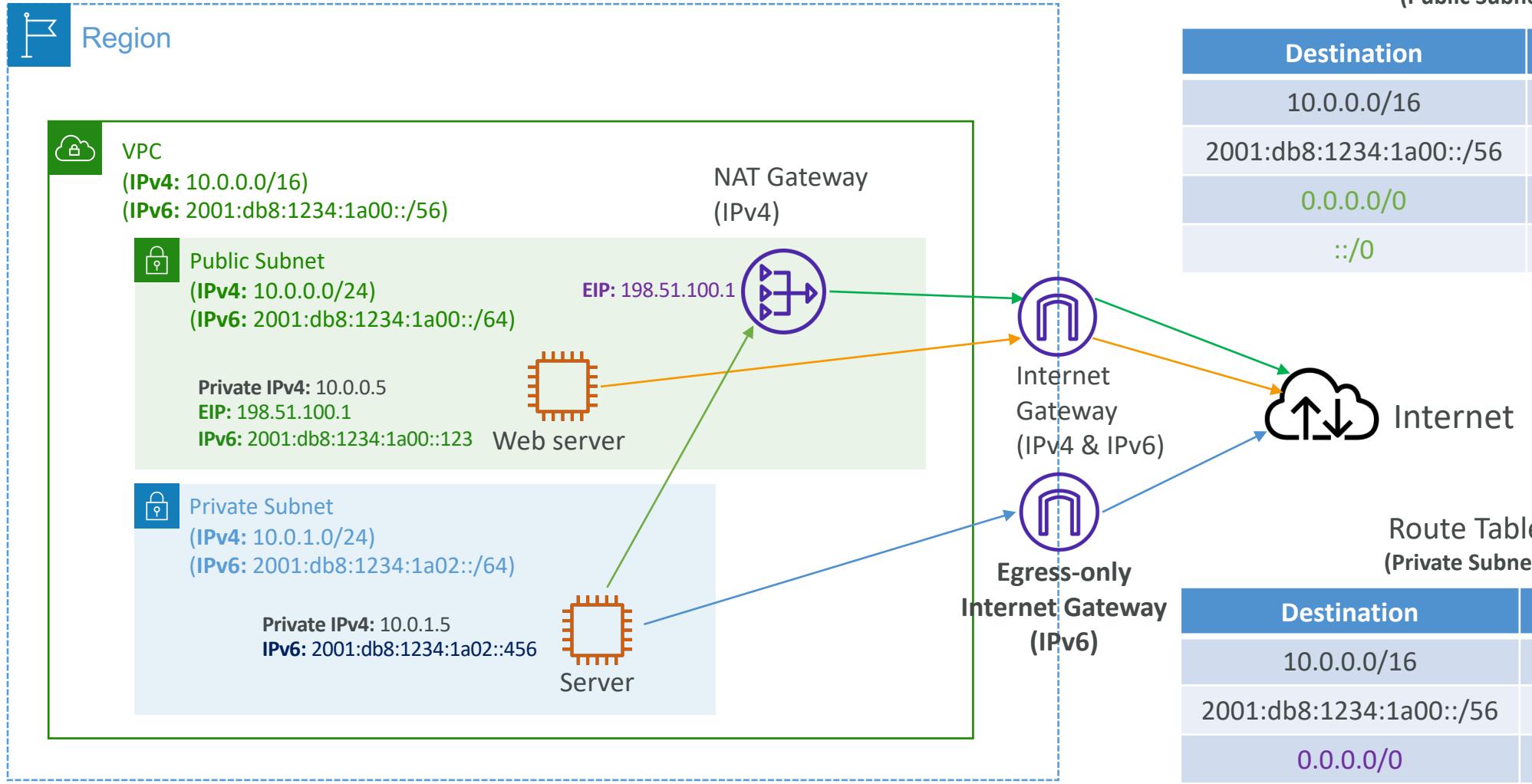
# Egress-only Internet Gateway



- Used for IPv6 only
- (similar to a NAT Gateway but for IPv6)
- Allows instances in your VPC outbound connections over IPv6 while preventing the internet to initiate an IPv6 connection to your instances
- You must update the Route Tables



# IPv6 Routing

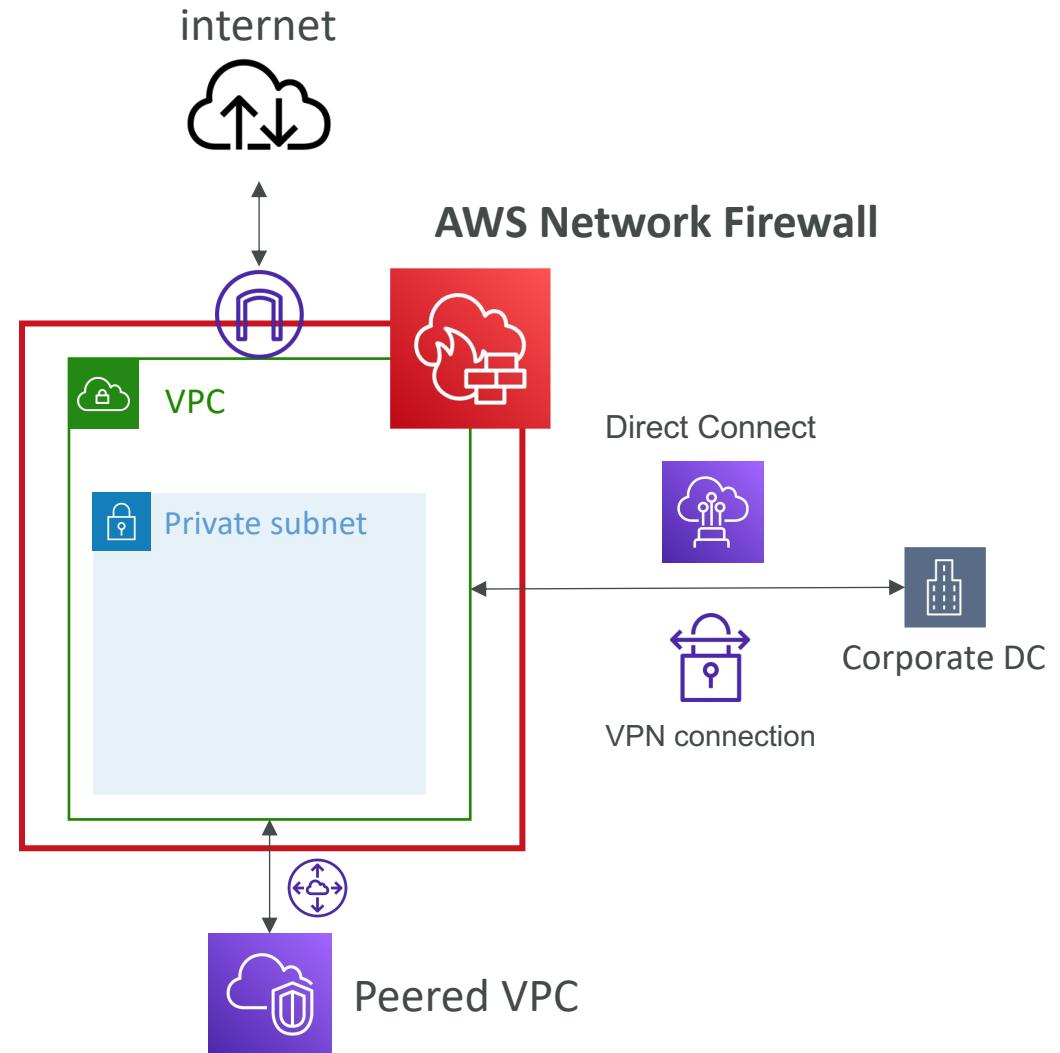


# Network Protection on AWS

- To protect network on AWS, we've seen
  - Network Access Control Lists (NACLs)
  - Amazon VPC security groups
  - AWS WAF (protect against malicious requests)
  - AWS Shield & AWS Shield Advanced
  - AWS Firewall Manager (to manage them across accounts)
- But what if we want to protect in a sophisticated way our entire VPC?

# AWS Network Firewall

- Protect your entire Amazon VPC
- From Layer 3 to Layer 7 protection
- Any direction, you can inspect
  - VPC to VPC traffic
  - Outbound to internet
  - Inbound from internet
  - To / from Direct Connect & Site-to-Site VPN
- Internally, the AWS Network Firewall uses the AWS Gateway Load Balancer
- Rules can be centrally managed cross-account by AWS Firewall Manager to apply to many VPCs





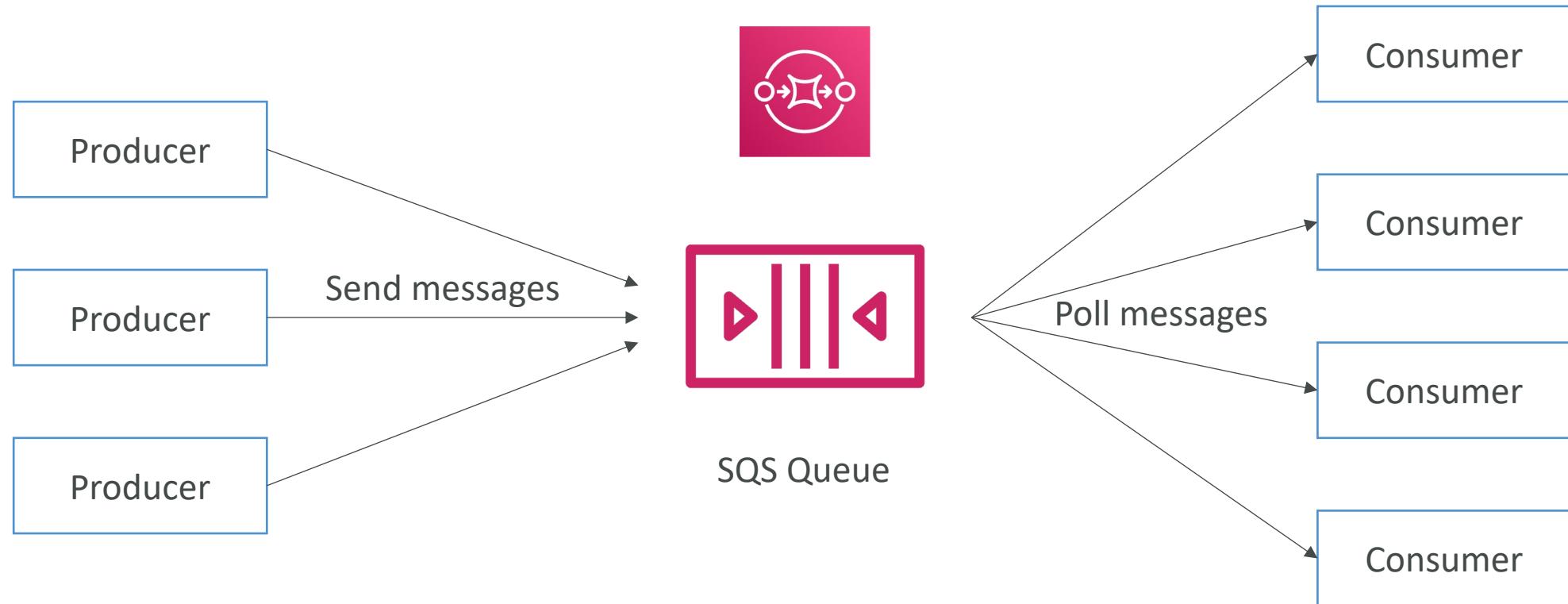
# Network Firewall – Fine Grained Controls

- Supports 1000s of rules
  - IP & port - example: 10,000s of IPs filtering
  - Protocol – example: block the SMB protocol for outbound communications
  - Stateful domain list rule groups: only allow outbound traffic to \*.mycorp.com or third-party software repo
  - General pattern matching using regex
- **Traffic filtering:** Allow, drop, or alert for the traffic that matches the rules
- Active flow inspection to protect against network threats with intrusion-prevention capabilities (like Gateway Load Balancer, but all managed by AWS)
- Send logs of rule matches to Amazon S3, CloudWatch Logs, Kinesis Data Firehose

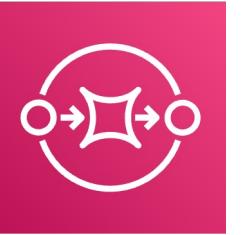
# Other Services

# Amazon SQS

## What's a queue?



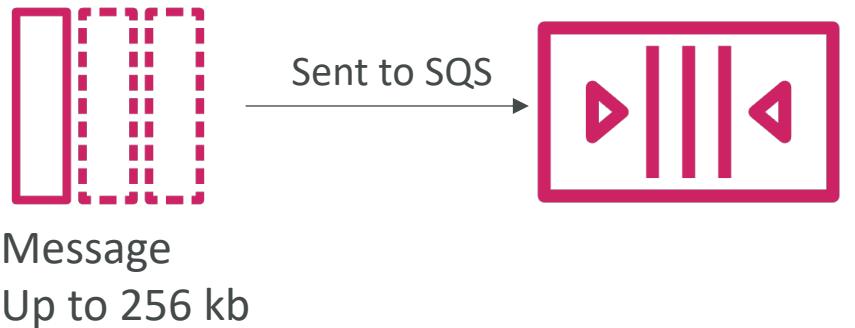
# Amazon SQS – Standard Queue



- Oldest offering (over 10 years old)
- Fully managed service, used to **decouple applications**
- Attributes:
  - Unlimited throughput, unlimited number of messages in queue
  - Default retention of messages: 4 days, maximum of 14 days
  - Low latency (<10 ms on publish and receive)
  - Limitation of 256KB per message sent
- Can have duplicate messages (at least once delivery, occasionally)
- Can have out of order messages (best effort ordering)

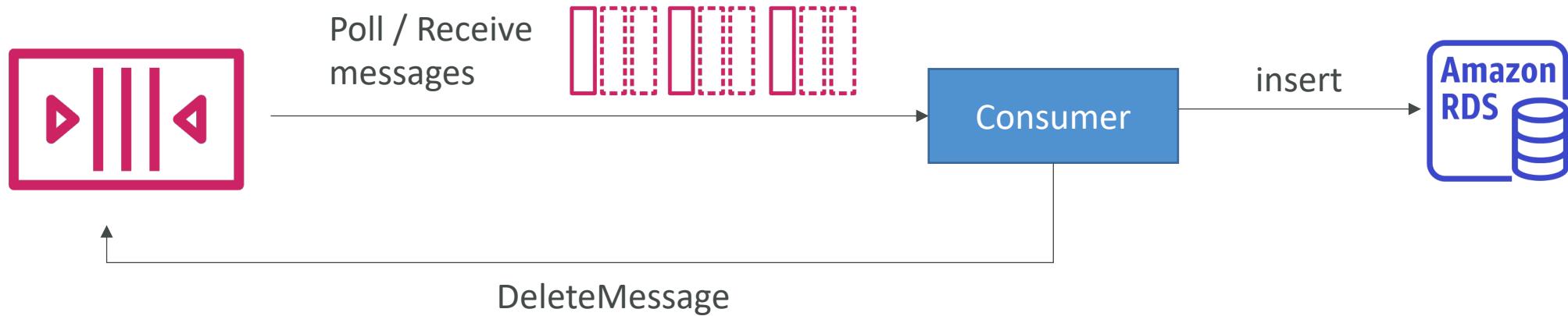
# SQS – Producing Messages

- Produced to SQS using the SDK (SendMessage API)
- The message is **persisted** in SQS until a consumer deletes it
- Message retention: default 4 days, up to 14 days
- Example: send an order to be processed
  - Order id
  - Customer id
  - Any attributes you want
- SQS standard: unlimited throughput

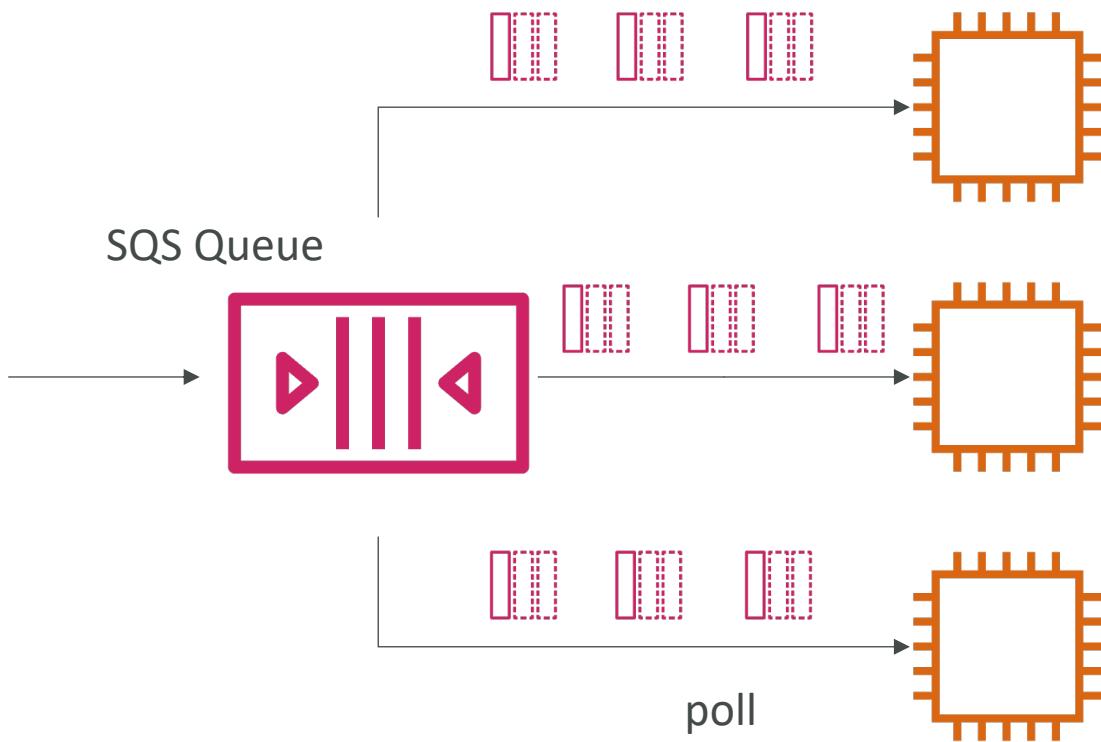


# SQS – Consuming Messages

- Consumers (running on EC2 instances, servers, or AWS Lambda)...
- Poll SQS for messages (receive up to 10 messages at a time)
- Process the messages (example: insert the message into an RDS database)
- Delete the messages using the DeleteMessage API

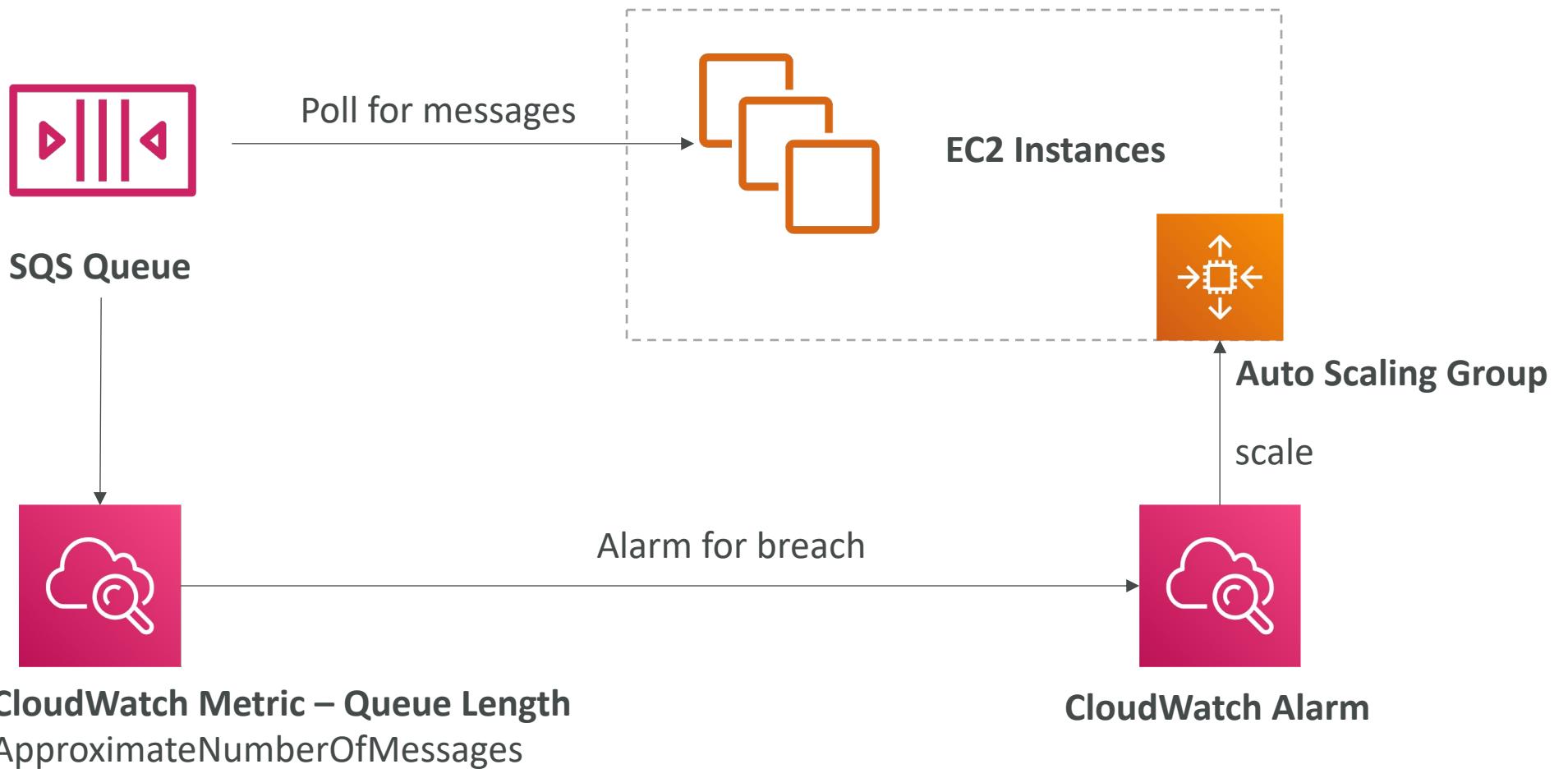


# SQS – Multiple EC2 Instances Consumers

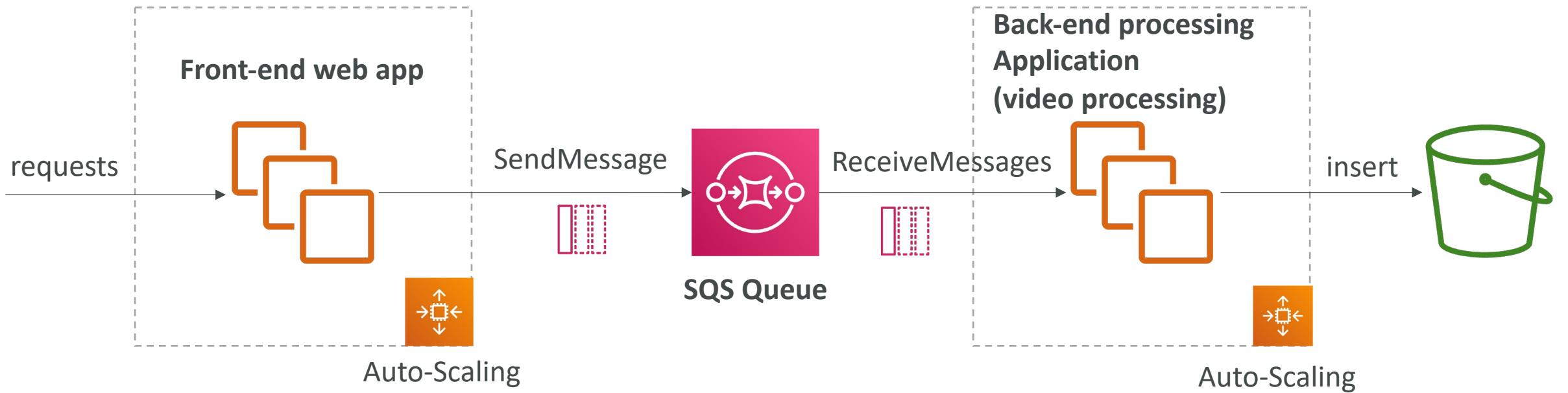


- Consumers receive and process messages in parallel
- At least once delivery
- Best-effort message ordering
- Consumers delete messages after processing them
- We can scale consumers horizontally to improve throughput of processing

# SQS with Auto Scaling Group (ASG)



# SQS to decouple between application tiers



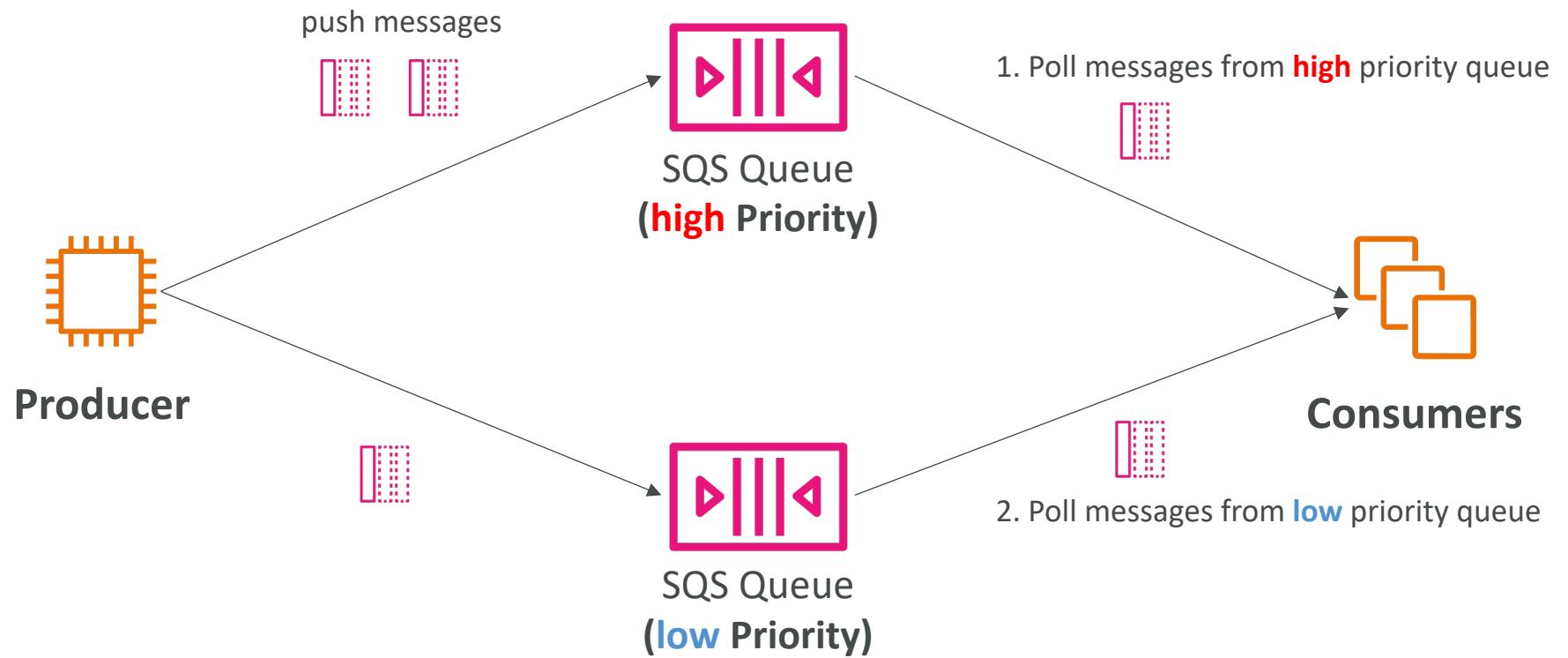
# Amazon SQS - Security

- **Encryption:**
  - In-flight encryption using HTTPS API
  - At-rest encryption using KMS keys
  - Client-side encryption if the client wants to perform encryption/decryption itself
- **Access Controls:** IAM policies to regulate access to the SQS API
- **SQS Access Policies** (similar to S3 bucket policies)
  - Useful for cross-account access to SQS queues
  - Useful for allowing other services (SNS, S3...) to write to an SQS queue

# Amazon SQS for CloudOps

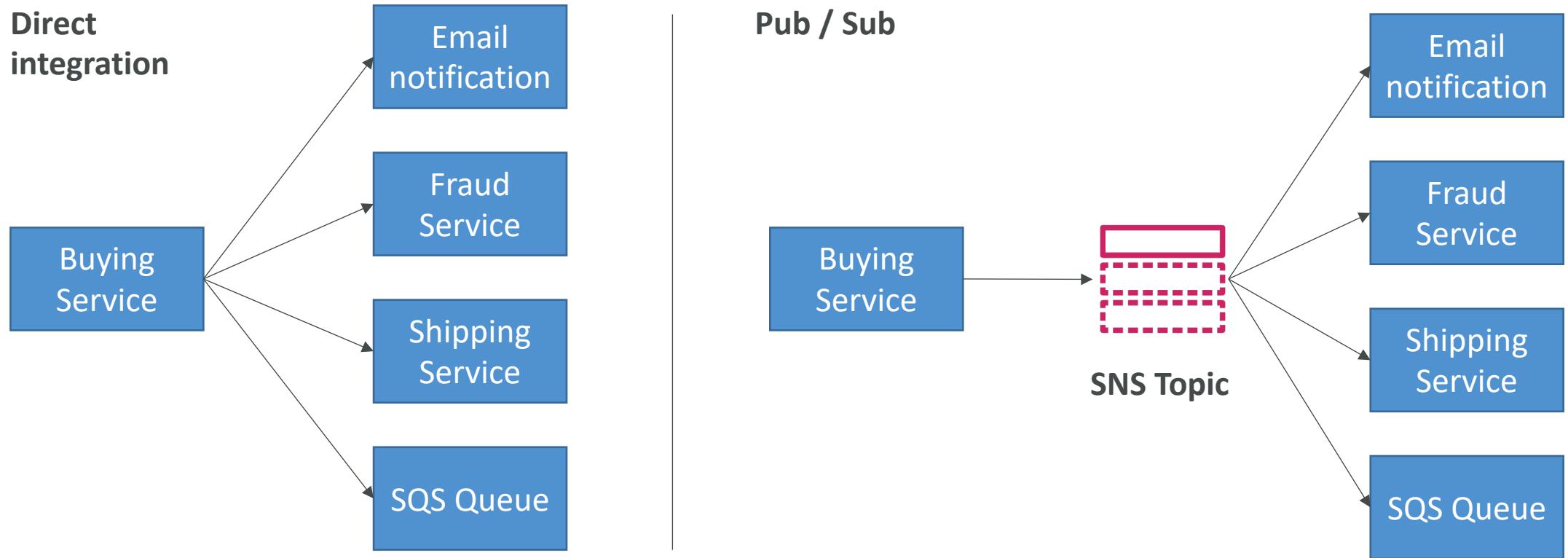
- **Message Visibility Timeout** – after a message is polled by a consumer, it becomes invisible to other consumers (default: 30 seconds)
- **Dead Letter Queue (DLQ)** – if the consumer fails to process the message for **MaximumReceives** threshold, it goes to the DLQ
- **Message Retention Period** – SQS automatically deletes messages from the Queue after that time, can be set from 1 minute to 14 days (default: 4 days)
- **SQS Access Policy** – is a Resource-based Policy which you can define permissions to access this Queue (similar to S3 Bucket Policy)

# SQS – Prioritization Pattern



# Amazon SNS

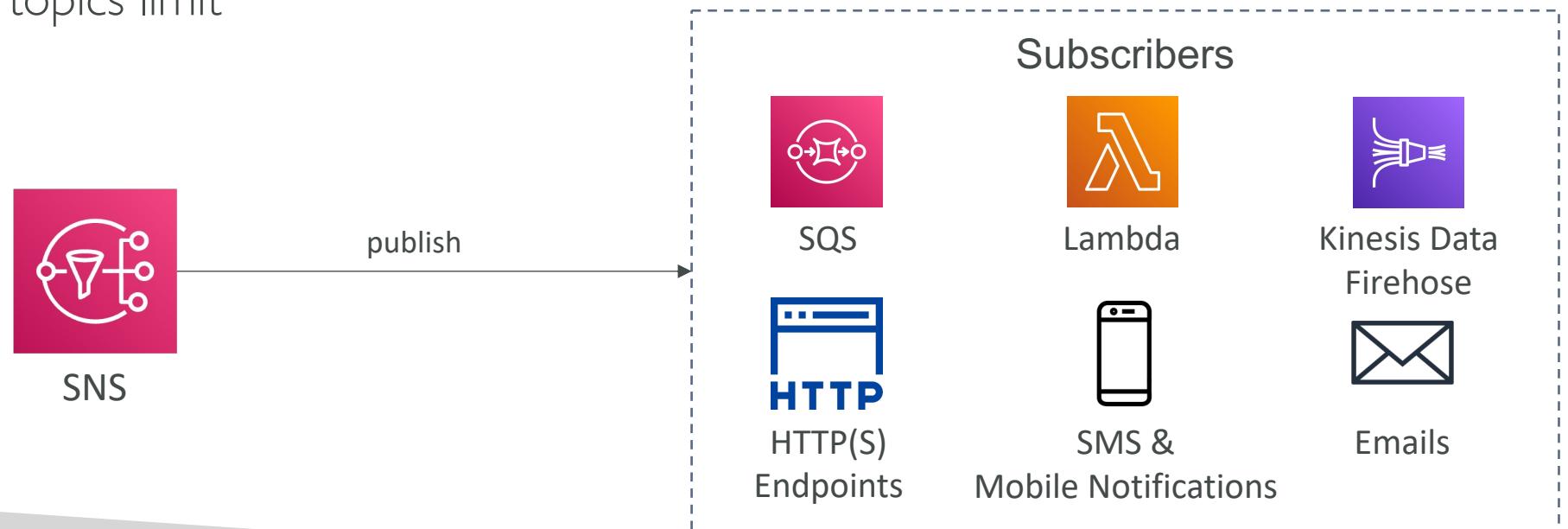
- What if you want to send one message to many receivers?



# Amazon SNS

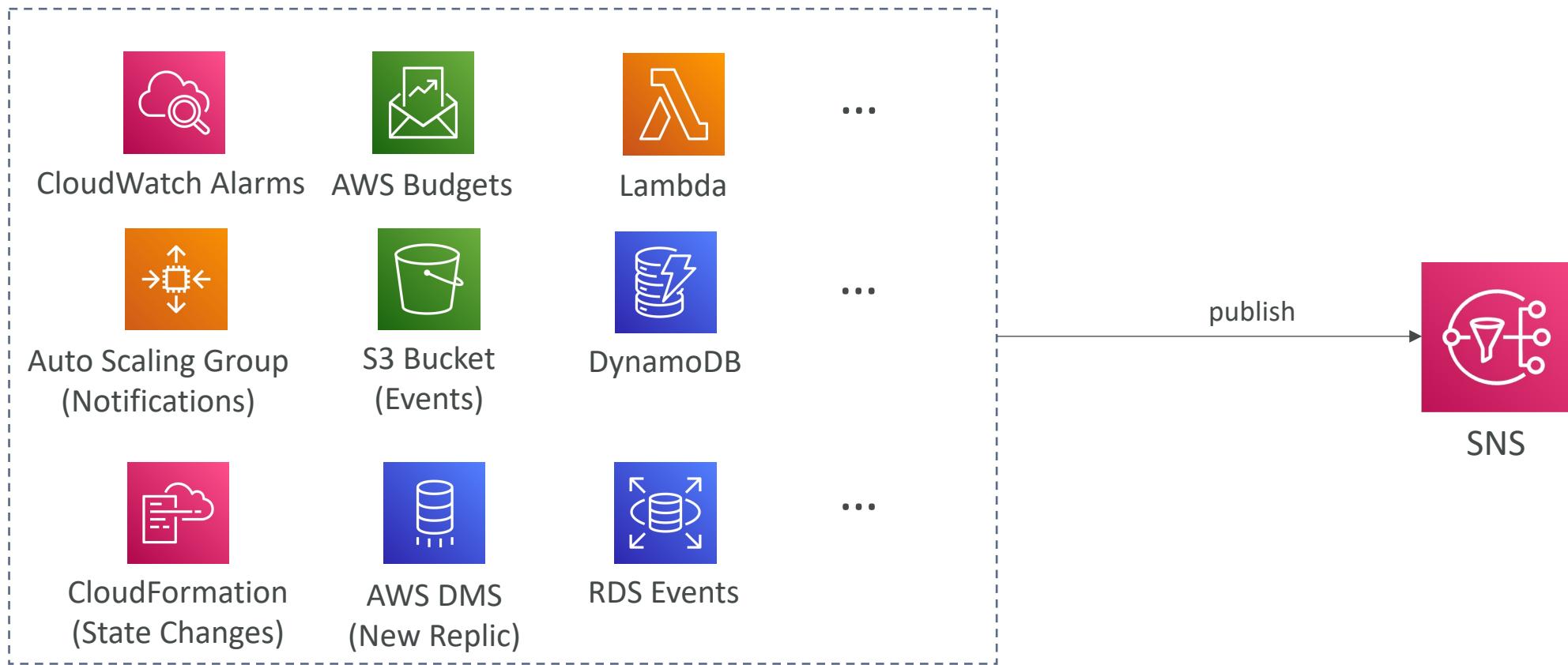


- The “event producer” only sends message to one SNS topic
- As many “event receivers” (subscriptions) as we want to listen to the SNS topic notifications
- Each subscriber to the topic will get all the messages (note: new feature to filter messages)
- Up to 12,500,000 subscriptions per topic
- 100,000 topics limit



# SNS integrates with a lot of AWS services

- Many AWS services can send data directly to SNS for notifications



# Amazon SNS – How to publish

- Topic Publish (using the SDK)
  - Create a topic
  - Create a subscription (or many)
  - Publish to the topic
- Direct Publish (for mobile apps SDK)
  - Create a platform application
  - Create a platform endpoint
  - Publish to the platform endpoint
  - Works with Google GCM, Apple APNS, Amazon ADM...

# Amazon SNS – Security

- **Encryption:**
  - In-flight encryption using HTTPS API
  - At-rest encryption using KMS keys
  - Client-side encryption if the client wants to perform encryption/decryption itself
- **Access Controls:** IAM policies to regulate access to the SNS API
- **SNS Access Policies** (similar to S3 bucket policies)
  - Useful for cross-account access to SNS topics
  - Useful for allowing other services ( S3...) to write to an SNS topic

# Amazon SNS – Filtering Messages

- By default, SNS Topic Subscriber receives every message published
- You can filter messages sent to the subscribers using a Filter Policy
- **Filter Policy** – JSON object that defines which messages the subscriber receives
  - attached to the Subscriber

SNS Message

```
{  
  "Type": "Notification",  
  "MessageId": "a1b2c34d-567e-8f90-g1h2-i345j67klmn8",  
  "TopicArn": "arn:aws:sns:us-east-2:123456789012:MyTopic",  
  "Message": "{  
    \"customer_interests\": [\"soccer\", \"rugby\", \"hockey\"],  
    \"store\": \"example_corp\",  
    \"event\": \"order_placed\",  
    \"price_usd\": 210.75  
  }",  
  ...  
}
```

Filter Policy (**accept**)

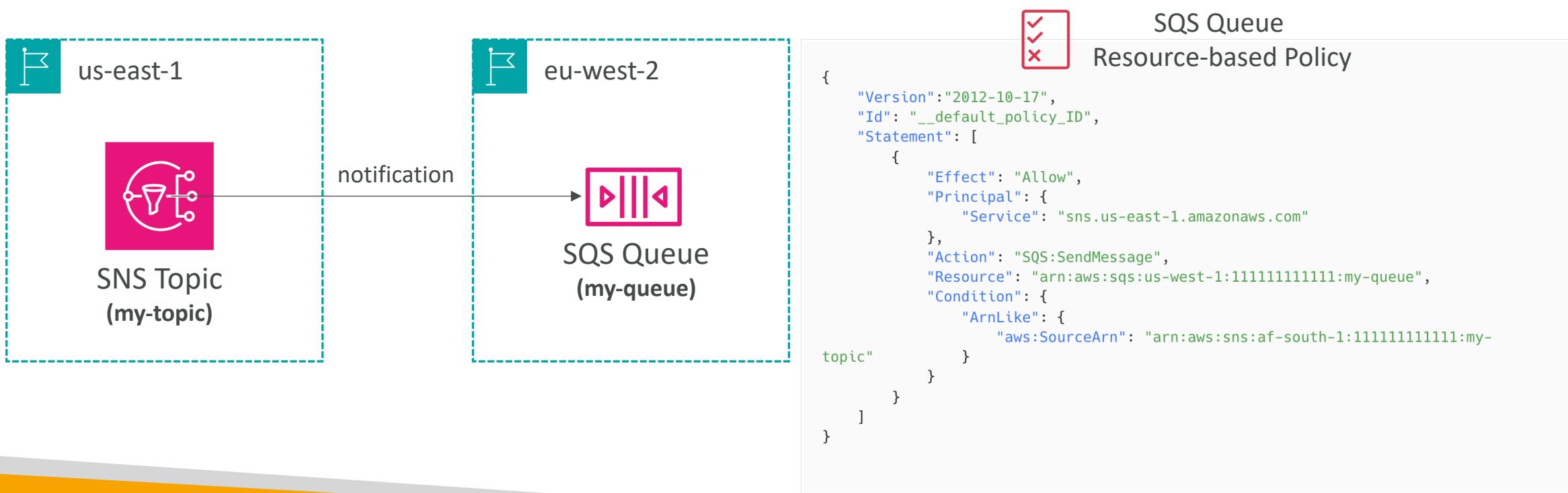
```
{  
  "store": ["example_corp"],  
  "event": [{ "anything-but": "order_cancelled" }],  
  "customer_interests": [ "rugby", "football", "baseball" ],  
  "price_usd": [{ "numeric": [ ">=", 100 ] }]  
}
```

Filter Policy (**deny**)

```
{  
  "store": ["example_corp"],  
  "event": [ "order_cancelled" ],  
  "encrypted": [ false ],  
  "customer_interests": [ "basketball", "baseball" ]  
}
```

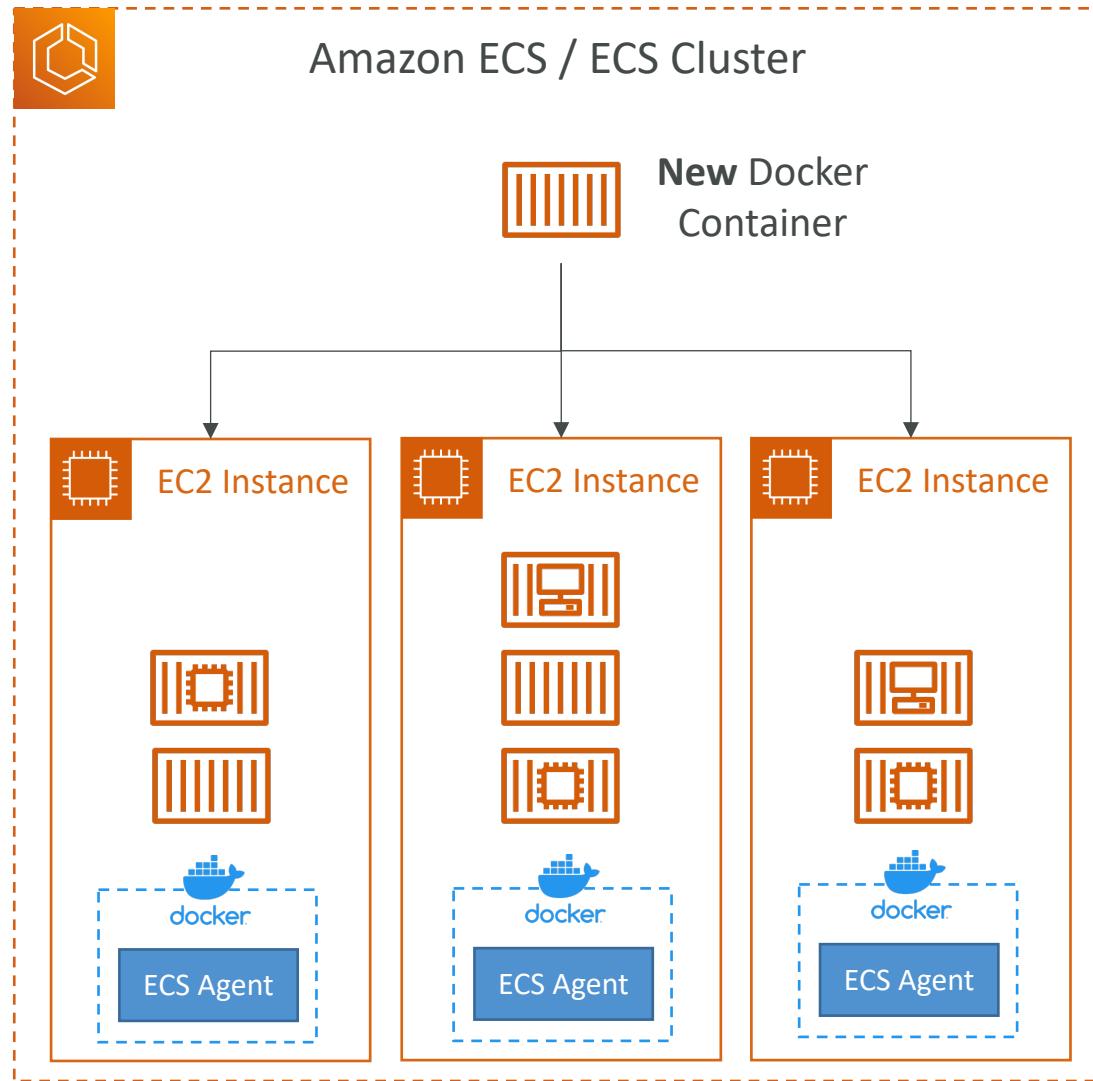
# Amazon SNS – Cross-Region Subscriber

- SNS can deliver notifications to subscribers in different regions (SQS Queues & Lambda functions only)
- Subscribers must have the required permissions to get notifications from SNS



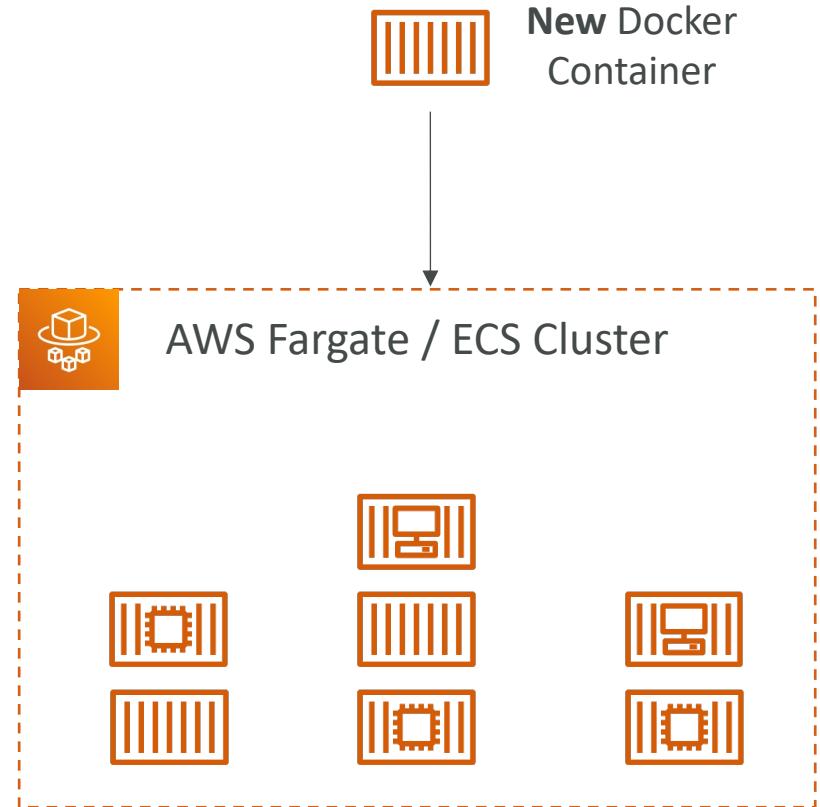
# Amazon ECS - EC2 Launch Type

- ECS = Elastic Container Service
- Launch Docker containers on AWS = Launch **ECS Tasks** on ECS Clusters
- EC2 Launch Type: you must provision & maintain the infrastructure (the EC2 instances)
- Each EC2 Instance must run the ECS Agent to register in the ECS Cluster
- AWS takes care of starting / stopping containers



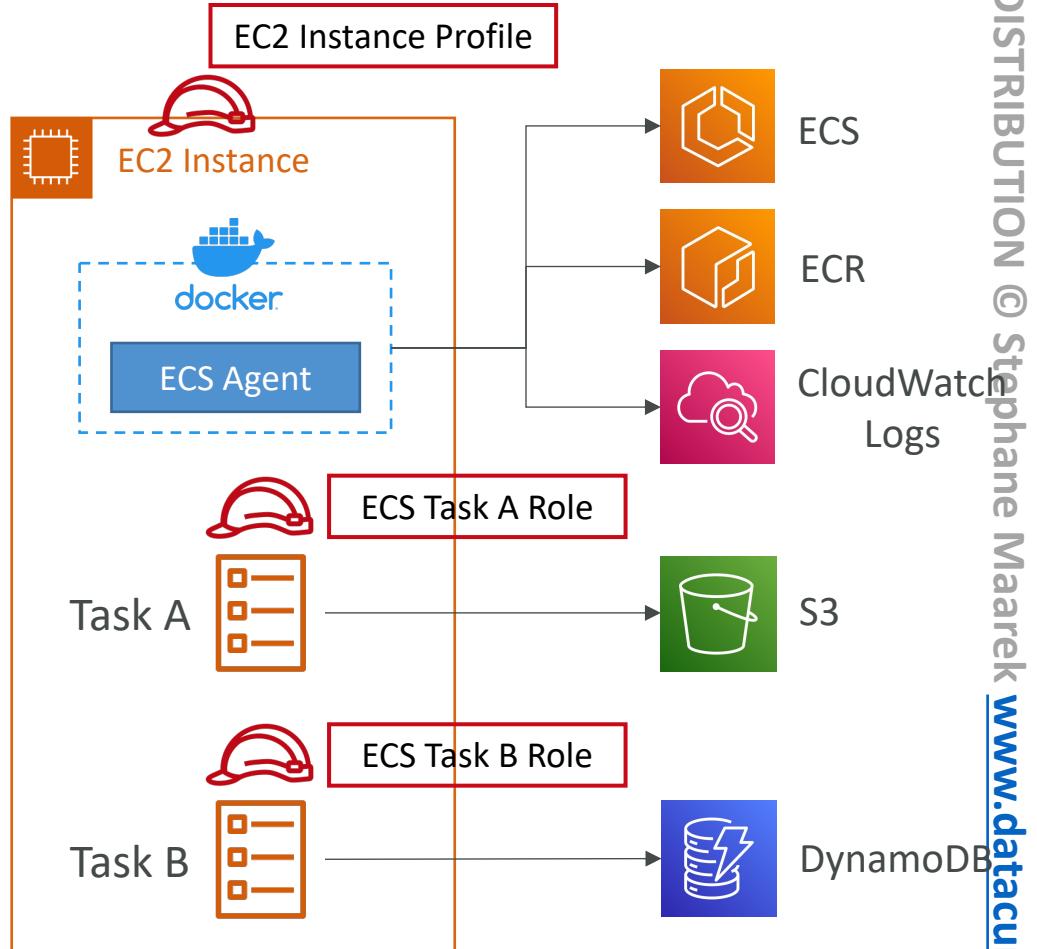
# Amazon ECS – Fargate Launch Type

- Launch Docker containers on AWS
- You do not provision the infrastructure  
(no EC2 instances to manage)
- It's all Serverless!
- You just create task definitions
- AWS just runs ECS Tasks for you based  
on the CPU / RAM you need
- To scale, just increase the number of  
tasks. Simple - no more EC2 instances



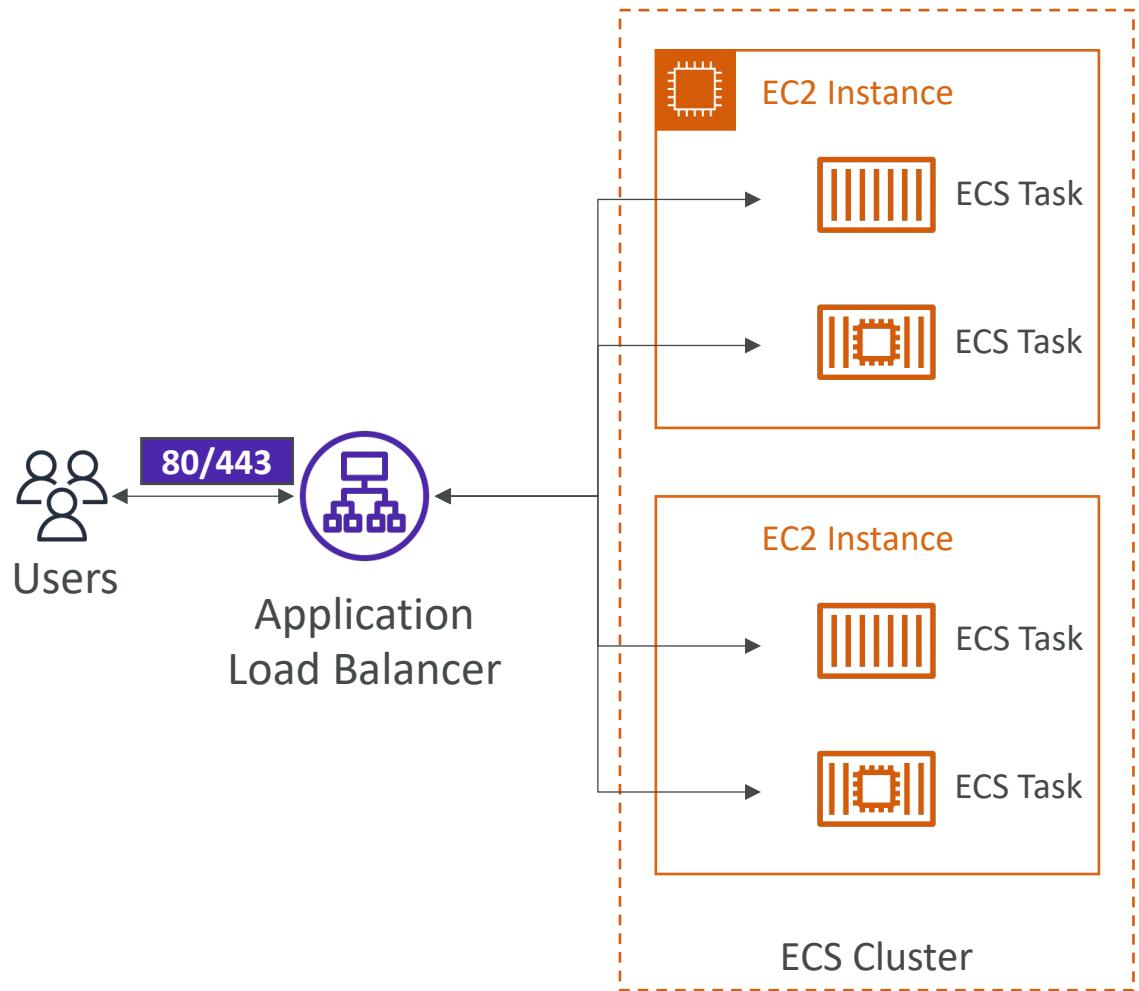
# Amazon ECS – IAM Roles for ECS

- **EC2 Instance Profile (EC2 Launch Type only):**
  - Used by the ECS agent
  - Makes API calls to ECS service
  - Send container logs to CloudWatch Logs
  - Pull Docker image from ECR
  - Reference sensitive data in Secrets Manager or SSM Parameter Store
- **ECS Task Role:**
  - Allows each task to have a specific role
  - Use different roles for the different ECS Services you run
  - Task Role is defined in the task definition



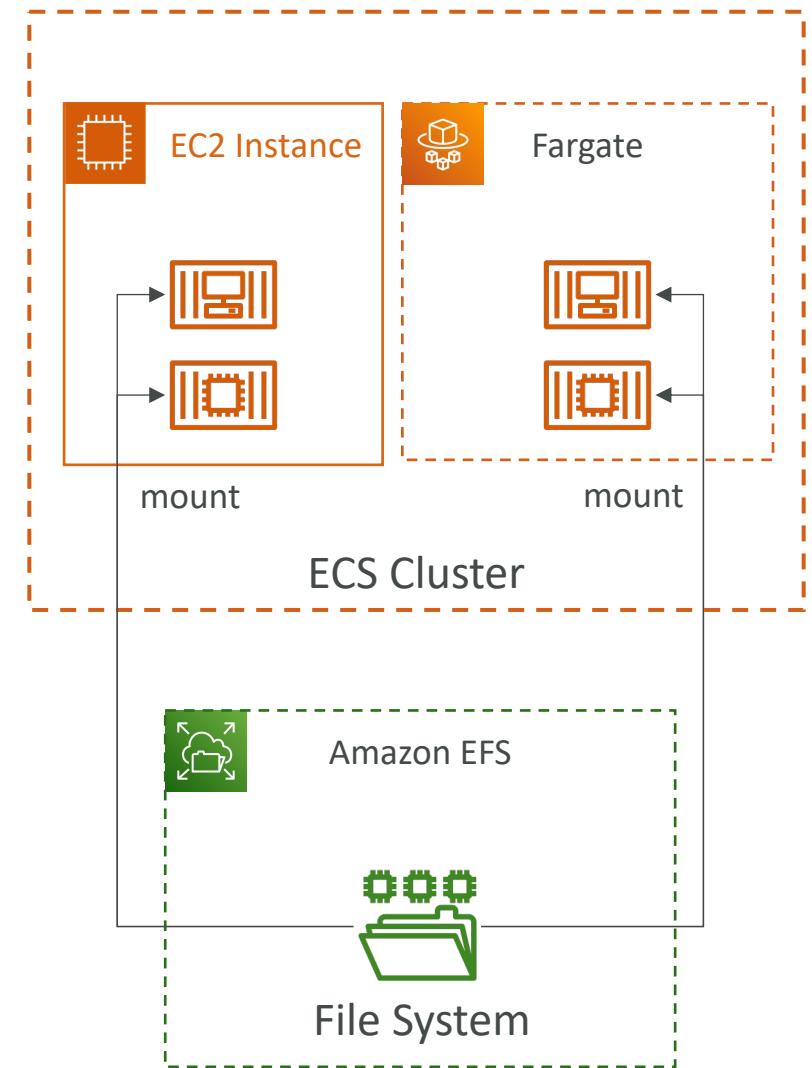
# Amazon ECS – Load Balancer Integrations

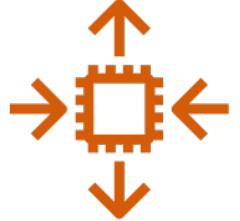
- **Application Load Balancer** supported and works for most use cases
- **Network Load Balancer** recommended only for high throughput / high performance use cases, or to pair it with AWS Private Link
- **Classic Load Balancer** supported but not recommended (no advanced features – no Fargate)



# Amazon ECS – Data Volumes (EFS)

- Mount EFS file systems onto ECS tasks
- Works for both **EC2** and **Fargate** launch types
- Tasks running in any AZ will share the same data in the EFS file system
- **Fargate + EFS = Serverless**
- Use cases: persistent multi-AZ shared storage for your containers
- Note:
  - Amazon S3 cannot be mounted as a file system





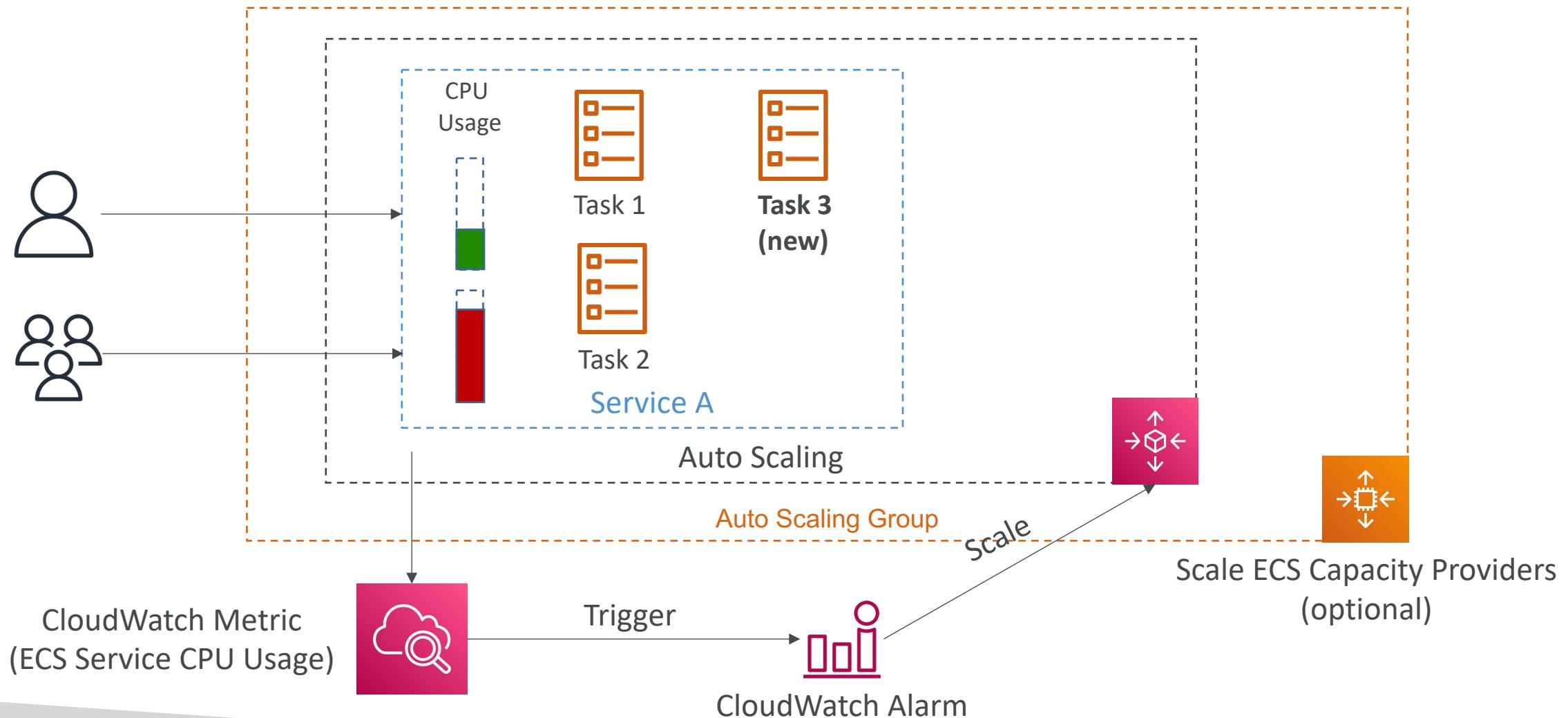
# ECS Service Auto Scaling

- Automatically increase/decrease the desired number of ECS tasks
- Amazon ECS Auto Scaling uses **AWS Application Auto Scaling**
  - ECS Service Average CPU Utilization
  - ECS Service Average Memory Utilization - Scale on RAM
  - ALB Request Count Per Target – metric coming from the ALB
- **Target Tracking** – scale based on target value for a specific CloudWatch metric
- **Step Scaling** – scale based on a specified CloudWatch Alarm
- **Scheduled Scaling** – scale based on a specified date/time (predictable changes)
- ECS Service Auto Scaling (task level) **≠** EC2 Auto Scaling (EC2 instance level)
- Fargate Auto Scaling is much easier to setup (because **Serverless**)

# EC2 Launch Type – Auto Scaling EC2 Instances

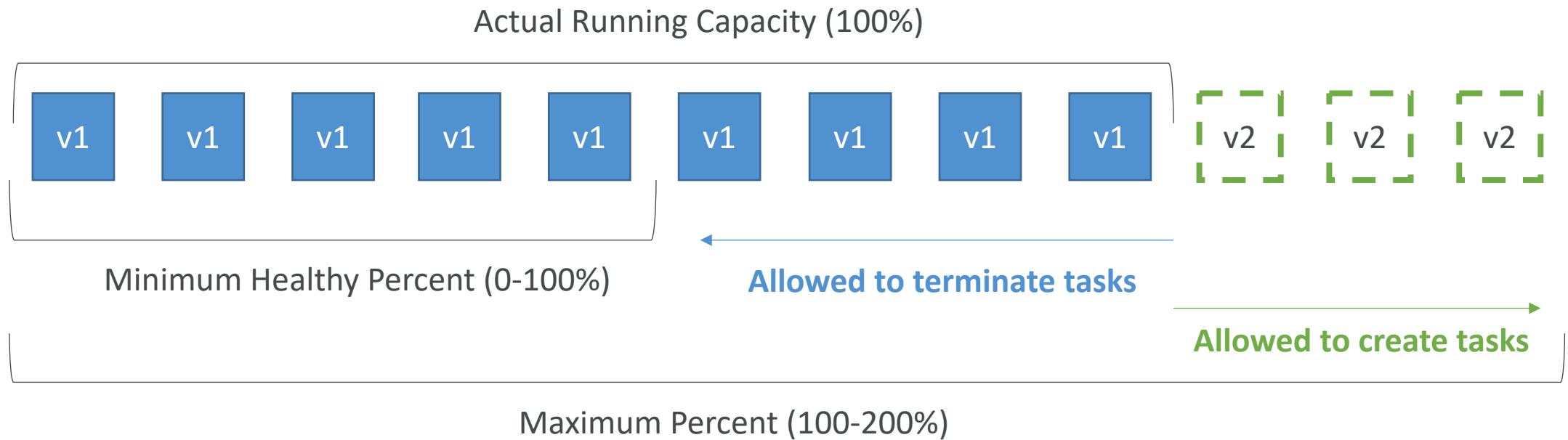
- Accommodate ECS Service Scaling by adding underlying EC2 Instances
- **Auto Scaling Group Scaling**
  - Scale your ASG based on CPU Utilization
  - Add EC2 instances over time
- **ECS Cluster Capacity Provider**
  - Used to automatically provision and scale the infrastructure for your ECS Tasks
  - Capacity Provider paired with an Auto Scaling Group
  - Add EC2 Instances when you're missing capacity (CPU, RAM...)

# ECS Scaling – Service CPU Usage Example



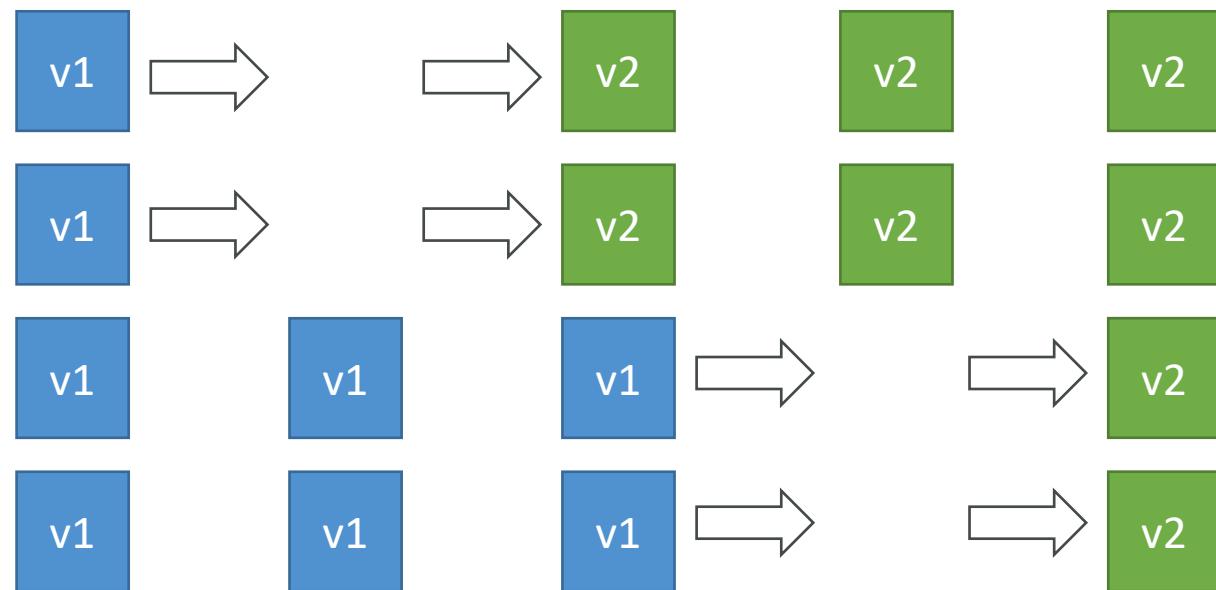
# ECS Rolling Updates

- When updating from v1 to v2, we can control how many tasks can be started and stopped, and in which order



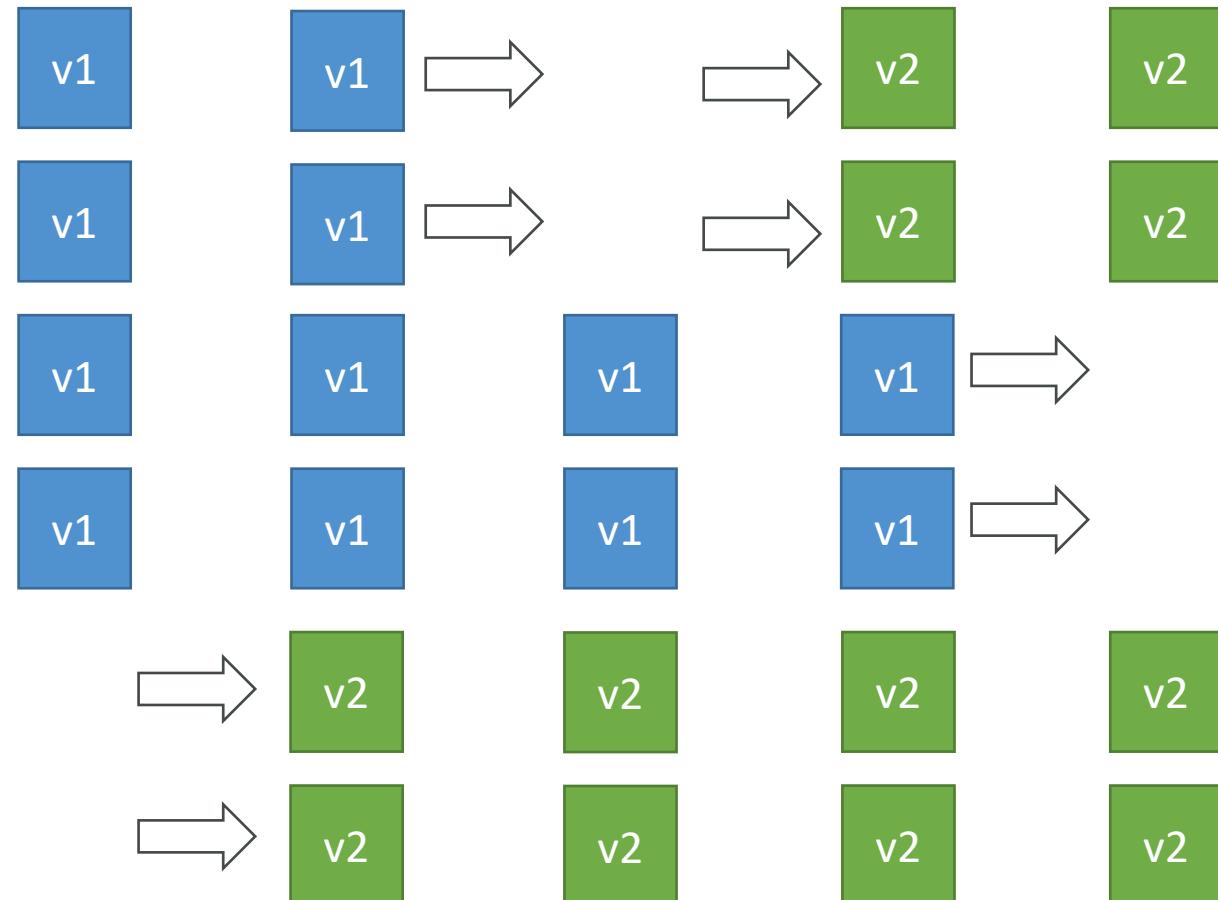
# ECS Rolling Update – Min 50%, Max 100%

- Starting number of tasks: 4



# ECS Rolling Update – Min 100%, Max 150%

- Starting number of tasks: 4



# Amazon ECS for CloudOps

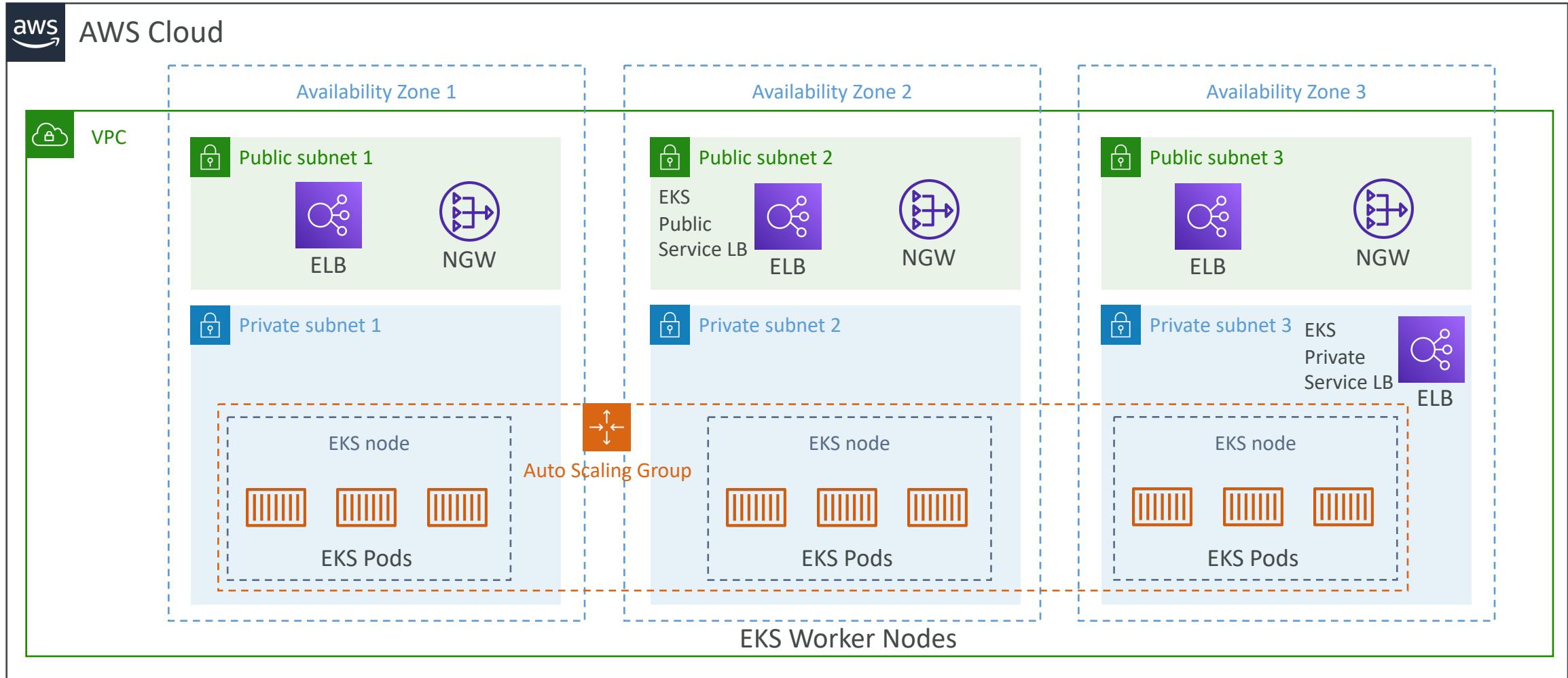
- **Ephemeral Storage for Fargate Tasks**
  - Fargate Tasks has ephemeral storage that can be shared among other Tasks
  - Configurable from 20 GB to max. of 200 GB
  - Encrypted using AES-256 encryption
- **Send ECS Tasks Logs to CloudWatch or AWS Partner**
  - Add the FireLens (Fluent Bit / Fluentd) container to your ECS Task Definition
  - Runs as a sidecar container & acts as a log router
  - Forwards logs to CloudWatch or 3rd party services

# Amazon EKS Overview



- Amazon EKS = Amazon Elastic **Kubernetes** Service
- It is a way to launch **managed Kubernetes clusters** on AWS
- Kubernetes is an **open-source system** for automatic deployment, scaling and management of containerized (usually Docker) application
- It's an alternative to ECS, similar goal but different API
- EKS supports **EC2** if you want to deploy worker nodes or **Fargate** to deploy serverless containers
- **Use case:** if your company is already using Kubernetes on-premises or in another cloud, and wants to migrate to AWS using Kubernetes
- **Kubernetes is cloud-agnostic** (can be used in any cloud – Azure, GCP...)
- For multiple regions, deploy one EKS cluster per region
- Collect logs and metrics using **CloudWatch Container Insights**

# Amazon EKS - Diagram

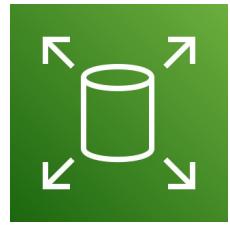


# Amazon EKS – Node Types

- **Managed Node Groups**
  - Creates and manages Nodes (EC2 instances) for you
  - Nodes are part of an ASG managed by EKS
  - Supports On-Demand or Spot Instances
- **Self-Managed Nodes**
  - Nodes created by you and registered to the EKS cluster and managed by an ASG
  - You can use prebuilt AMI - Amazon EKS Optimized AMI
  - Supports On-Demand or Spot Instances
- **AWS Fargate**
  - No maintenance required; no nodes managed

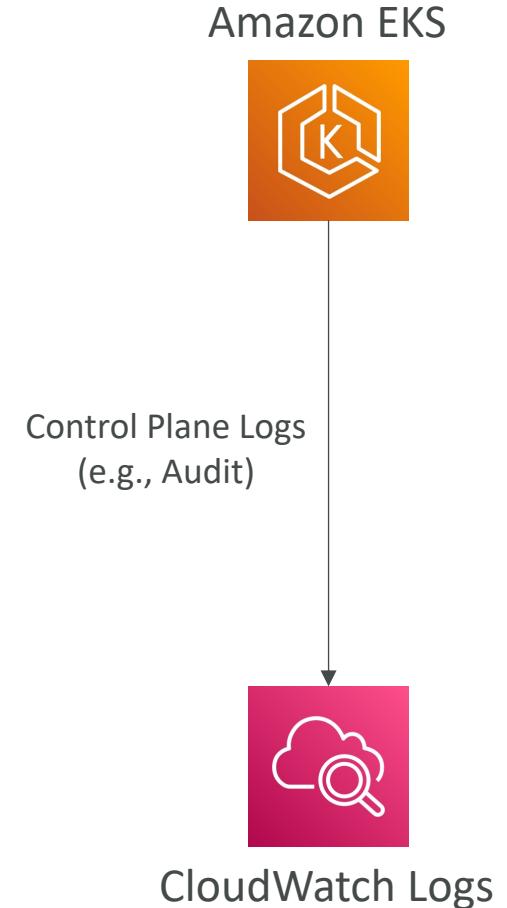
# Amazon EKS – Data Volumes

- Need to specify **StorageClass** manifest on your EKS cluster
- Leverages a **Container Storage Interface (CSI)** compliant driver
- Support for...
- Amazon EBS
- Amazon EFS (works with Fargate)
- Amazon FSx for Lustre
- Amazon FSx for NetApp ONTAP



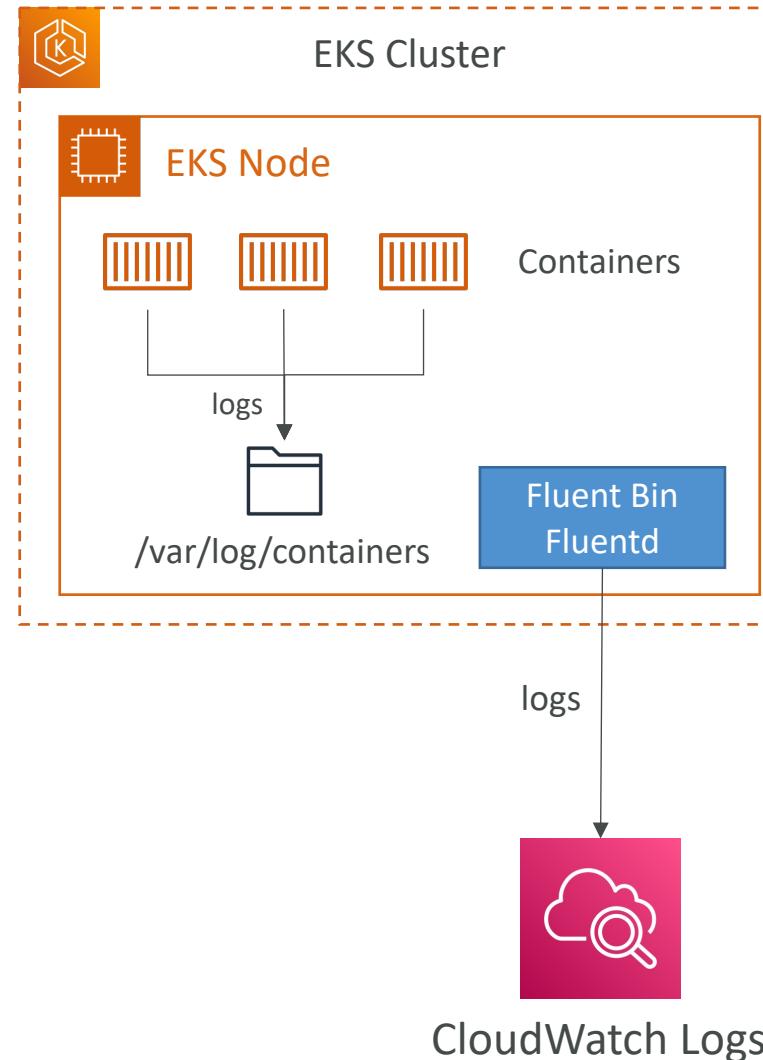
# Amazon EKS – Control Plane Logging

- Send EKS Control Plane audit and diagnostic logs to CloudWatch Logs
- EKS Control Plane Log Types
  - API Server (api)
  - Audit (audit)
  - Authenticator (authenticator)
  - Controller Manager (controllerManager)
  - Scheduler (scheduler)
- Ability to select the exact log types to send to CloudWatch Logs



# Amazon EKS – Nodes & Containers Logging

- You can capture node, pod, and containers logs and send them to CloudWatch Logs
- Use **CloudWatch Agent** to send metrics to CloudWatch
- Use the **Fluent Bit**, or **Fluentd** log drivers to send logs to CloudWatch Logs
- Container logs are stored on a Node directory `/var/log/containers`
- Use **CloudWatch Container Insights** to get a dashboarding monitoring solution for nodes, pods, tasks, and services



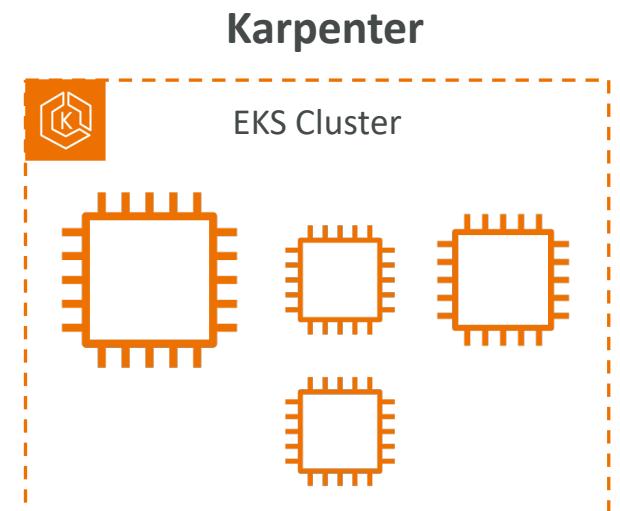
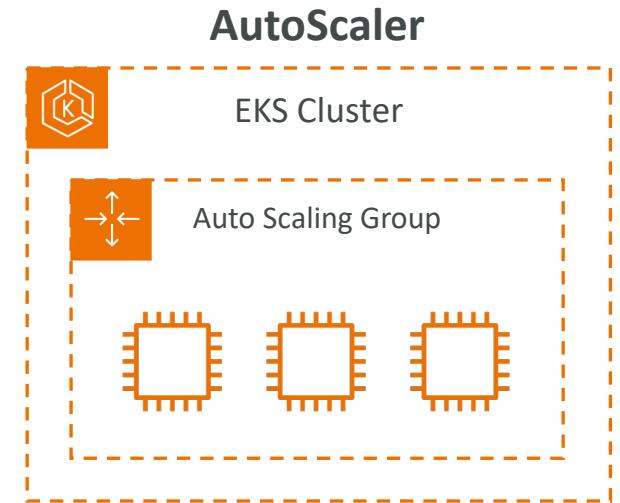
# Amazon EKS for CloudOps

- EKS Cluster Insights – detect issues and provide recommendations
  - Configuration Insights – identifies misconfiguration in your ECS Cluster (hybrid)
  - Upgrade Insights – identifies issues that could impact your ability to upgrade to new Kubernetes version
- To upgrade your EKS Cluster:
  1. Review Upgrade Insights in EKS Cluster Insights to identify any issues might occur while upgrading
  2. Update Cluster Control Plane
  3. Update Cluster Components (e.g. Nodes)

```
aws eks update-cluster-version --name <cluster-name> \
--kubernetes-version <verion-number> --region <region-code>
```

# Amazon EKS – AutoScaling Modes

- **Cluster AutoScaler** – automatically adjusts the number of nodes (**same size**) in your cluster (uses Auto Scaling Groups)
- **Karpenter** – launching **right-sized** compute resources (EC2 instances, Fargate) in response to load changes in under a minute
- **EKS Auto Mode (AWS-managed Karpenter)** – automatically scales cluster compute resources with ability to consolidate workloads and delete nodes



# 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



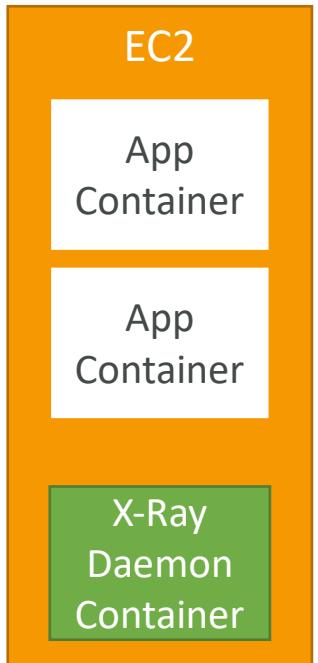
- Troubleshooting performance (bottlenecks)
- Understand dependencies in a microservice architecture
- Pinpoint service issues
- Review request behavior
- Find errors and exceptions
- Are we meeting time SLA?
- Where I am throttled?
- Identify users that are impacted

# ECS + X-Ray integration options



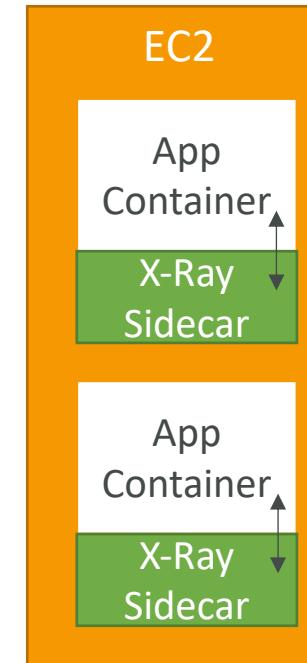
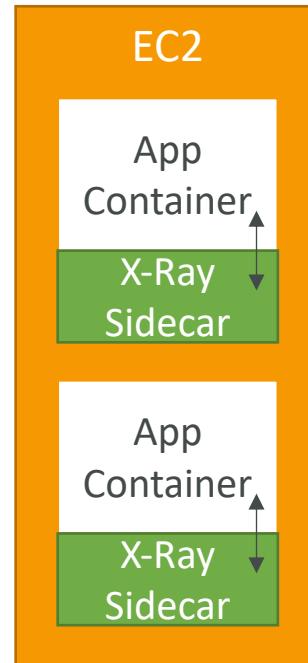
ECS Cluster

X-Ray Container as a Daemon



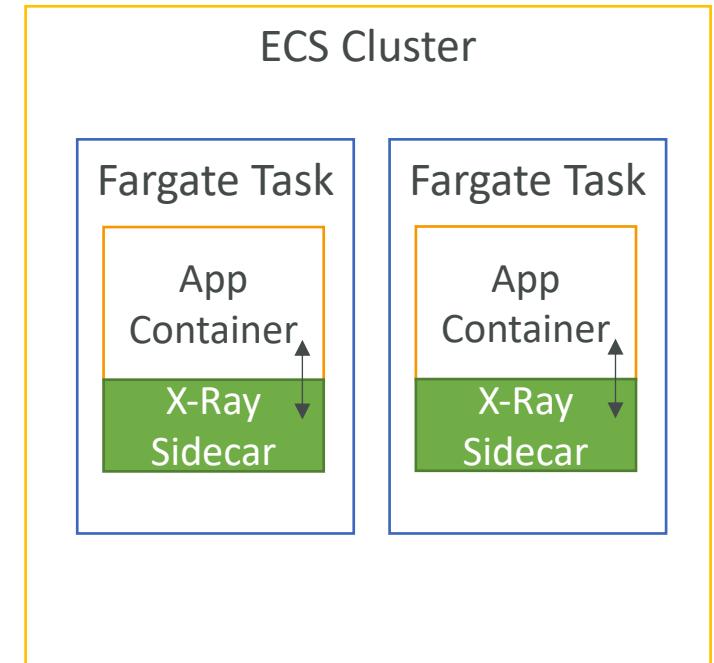
ECS Cluster

X-Ray Container as a “Side Car”



Fargate Cluster

X-Ray Container as a “Side Car”



# ECS + X-Ray: Example Task Definition

```
{  
    "name": "xray-daemon",  
    "image": "123456789012.dkr.ecr.us-east-2.amazonaws.com/xray-daemon",  
    "cpu": 32,  
    "memoryReservation": 256,  
    "portMappings" : [  
        {  
            "hostPort": 0,  
            "containerPort": 2000,  
            "protocol": "udp"  
        },  
    ],  
},  
{  
    "name": "scorekeep-api",  
    "image": "123456789012.dkr.ecr.us-east-2.amazonaws.com/scorekeep-api",  
    "cpu": 192,  
    "memoryReservation": 512,  
    "environment": [  
        { "name" : "AWS_REGION", "value" : "us-east-2" },  
        { "name" : "NOTIFICATION_TOPIC", "value" : "arn:aws:sns:us-east-2:123456789012:scorekeep-notifications" },  
        { "name" : "AWS_XRAY_DAEMON_ADDRESS", "value" : "xray-daemon:2000" }  
    ],  
    "portMappings" : [  
        {  
            "hostPort": 5000,  
            "containerPort": 5000  
        }  
    ],  
    "links": [  
        "xray-daemon"  
    ]  
}
```

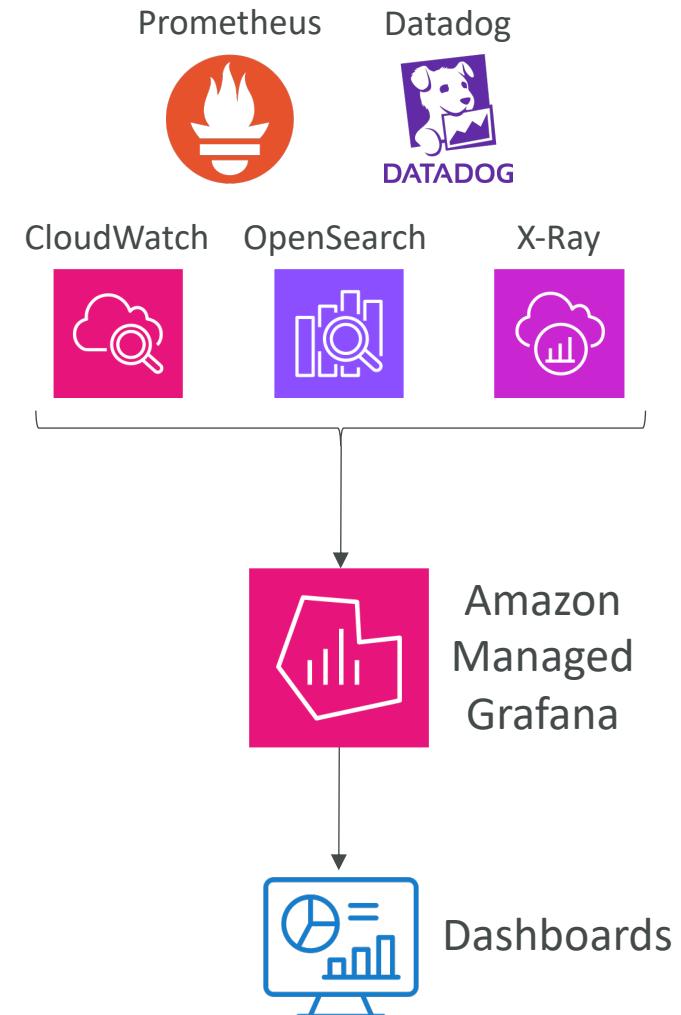
<https://docs.aws.amazon.com/xray/latest/devguide/xray-daemon-ecs.html#xray-daemon-ecs-build>

# Amazon Managed Service for Grafana



# Amazon Managed Service for Grafana

- Fully managed Grafana service that lets you create dashboards and visualize metrics, logs, and traces
- You create **Workspaces** in which you build your Grafana dashboards
- AWS handles provisioning, scaling, and maintenance
- Integrates with CloudWatch, Amazon Managed Service for Prometheus, X-Ray, OpenSearch...
- Supports many third-party and open-source data sources (e.g., Prometheus, Datadog...)

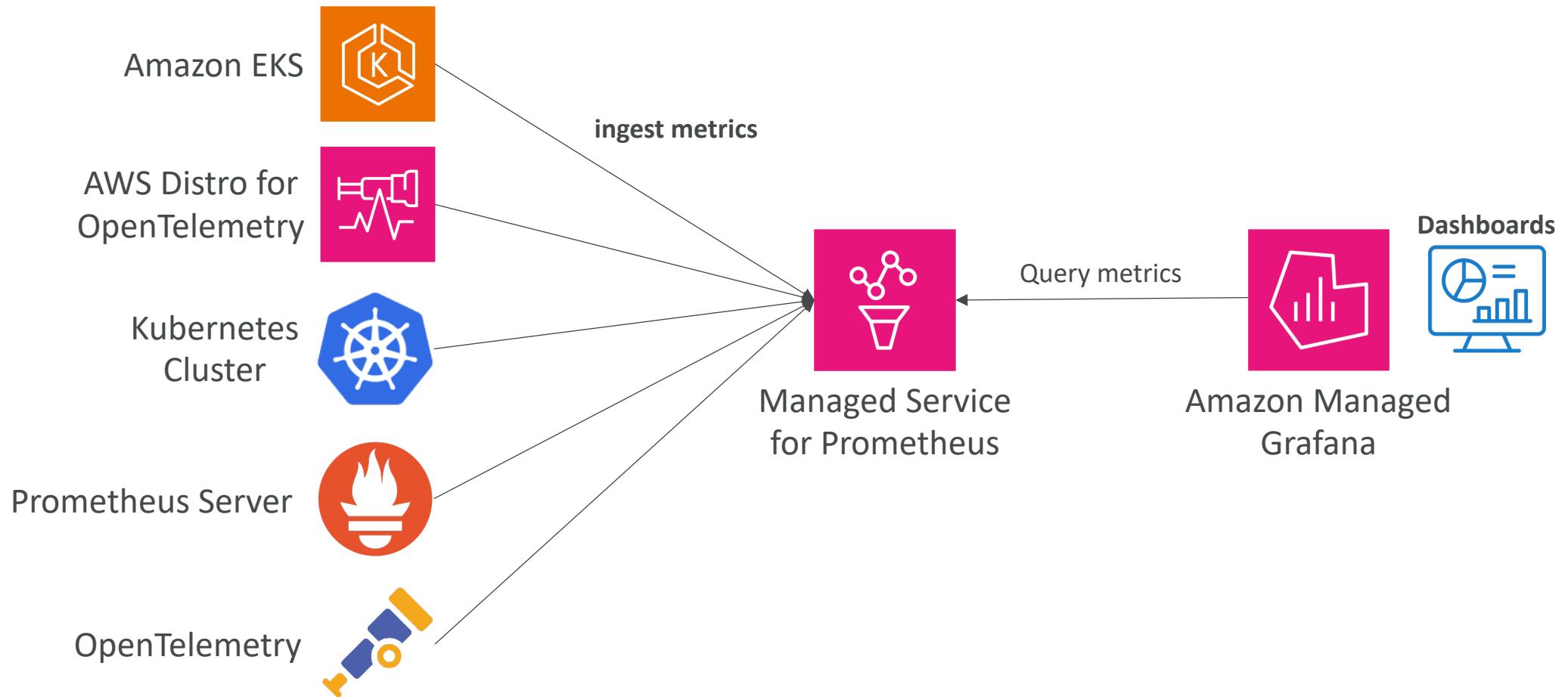




# Amazon Managed Service for Prometheus

- Serverless, Prometheus-compatible service for monitoring container metrics
- Automatically scales ingestion, storage, query operations
- Data is replicated across 3 Availability Zones (highly available)
- Same open-source Prometheus data model and PromQL Query language
  - PromQL – a query language used to filter and aggregate metrics
- Works with Amazon EKS and self-managed Kubernetes clusters
- Metrics can be stored up to 3 years (default: 150 days Retention Period)

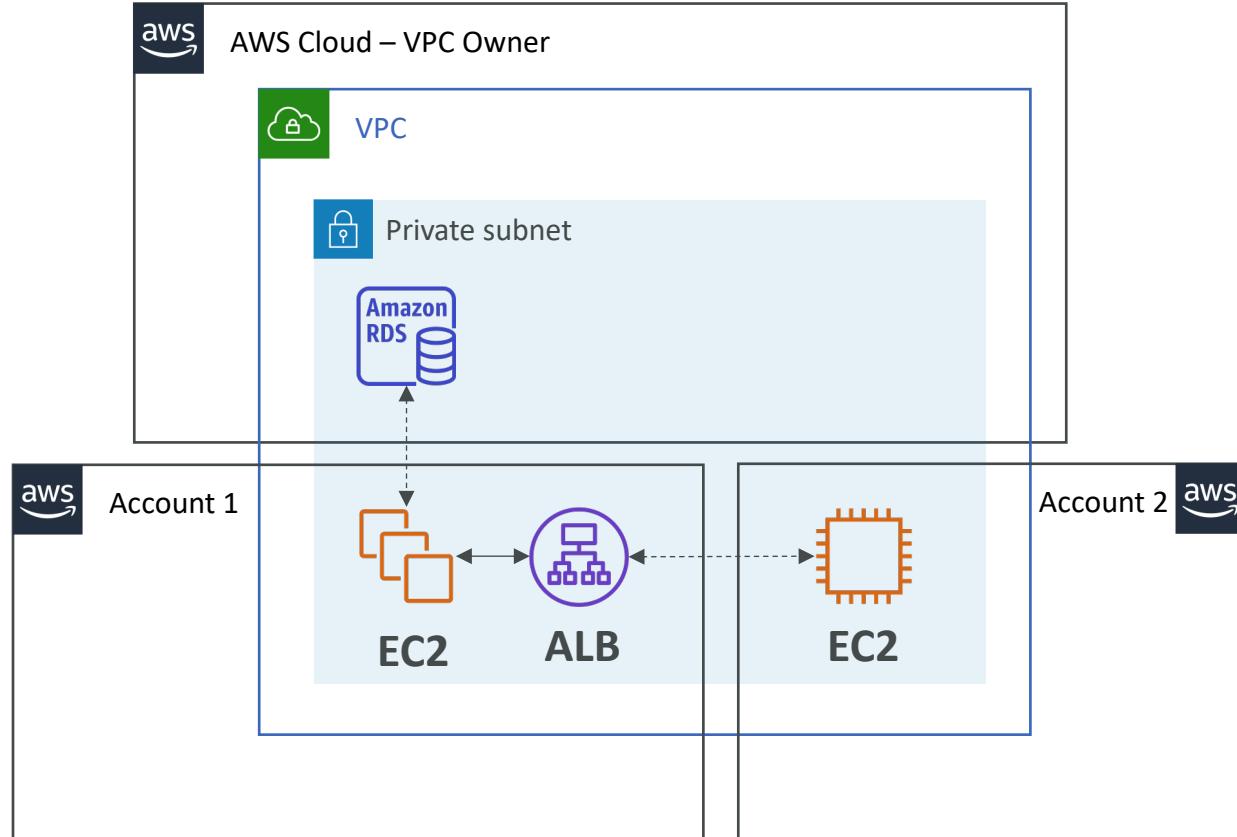
# Amazon Managed Service for Prometheus



# AWS Resource Access Manager (RAM)

- Share AWS resources that you own with other AWS accounts
- Share with any account or within your Organization
- Avoid resource duplication!
- **VPC Subnets:**
  - allow to have all the resources launched in the same subnets
  - must be from the same AWS Organizations.
  - Cannot share security groups and default VPC
  - Participants can manage their own resources in there
  - Participants can't view, modify, delete resources that belong to other participants or the owner
- AWS Transit Gateway
- Route53 Resolver Rules
- License Manager Configurations

# Resource Access Manager – VPC example



- Each account...
  - is responsible for its own resources
  - cannot view, modify or delete other resources in other accounts
- Network is shared so...
  - Anything deployed in the VPC can talk to other resources in the VPC
  - Applications are accessed easily across accounts, using private IP!
  - Security groups from other accounts can be referenced for maximum security

# Exam Review & Tips

# State of learning checkpoint

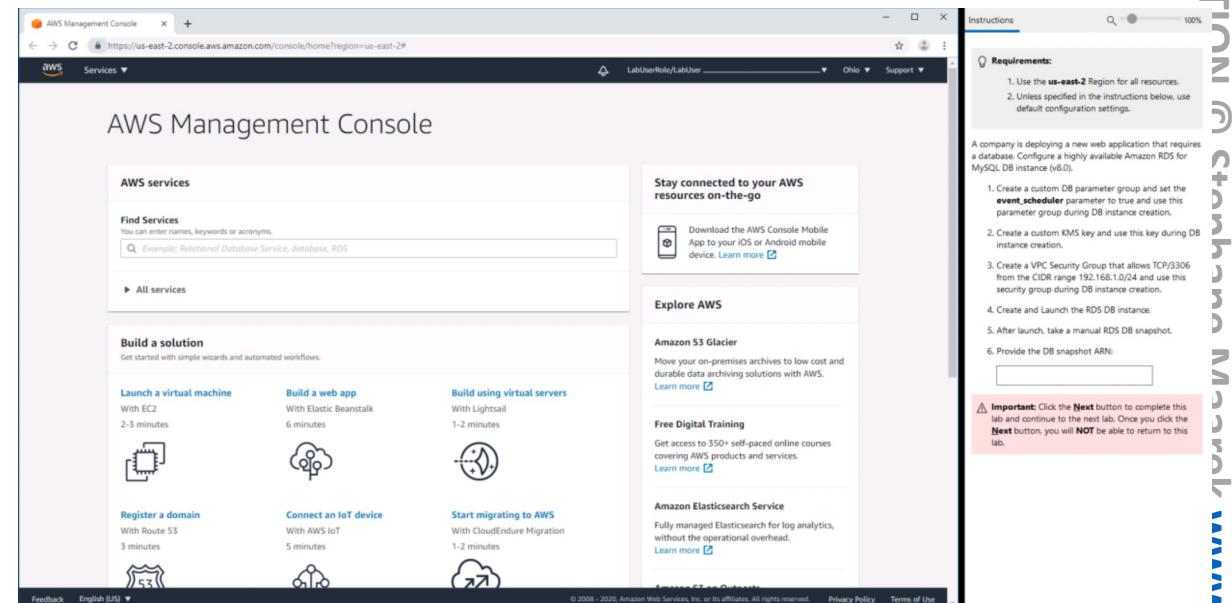
- Let's look how far we've gone on our learning journey
- <https://aws.amazon.com/certification/certified-sysops-admin-associate/>

# AWS Certified SysOps Exam Labs

- Starting with SOA-C02, the exam will contain **3 labs**
- Each exam lab consists of several different tasks
- AWS recommends to **allocate 20 minutes for each exam lab**
- First, you'll need to answer the multiple-choice/multiple-responses questions (you can't go back once you have completed this section)
- Second, you'll need to answer the exam labs
- You must complete all the work on the exam lab, before you move to the next one (you won't be able to go back to a lab once you have completed it)

# AWS Certified SysOps Exam Sample Lab

- A company is deploying a new web application. Configure a highly available MySQL 8.0 database with the following:
  1. Create a custom DB parametergroup and set the `event_scheduler` parameter to true and use this parameter during DBinstance creation.
  2. Create a custom AWS Key Management Service (AWS KMS) key and use this key during DBinstance creation.
  3. Create a VPC security group that allows TCP port 3306 from the CIDR block 192.168.1.0/24. Use this security group during DB instance creation.
  4. Launch the Amazon RDS DB instance.
  5. After launch, take a manual RDS DB snapshot.



# Your AWS Certification journey

## Foundational

Knowledge-based certification for foundational understanding of AWS Cloud.

**No prior experience needed.**



## Associate

Role-based certifications that showcase your knowledge and skills on AWS and build your credibility as an AWS Cloud professional.

**Prior cloud and/or strong on-premises IT experience recommended.**



## Professional

Role-based certifications that validate advanced skills and knowledge required to design secure, optimized, and modernized applications and to automate processes on AWS.

**2 years of prior AWS Cloud experience recommended.**



## Specialty

Dive deeper and position yourself as a trusted advisor to your stakeholders and/or customers in these strategic areas.

**Refer to the exam guides on the exam pages for recommended experience.**



# AWS Certification Paths – Architecture

## Architecture

### Solutions Architect

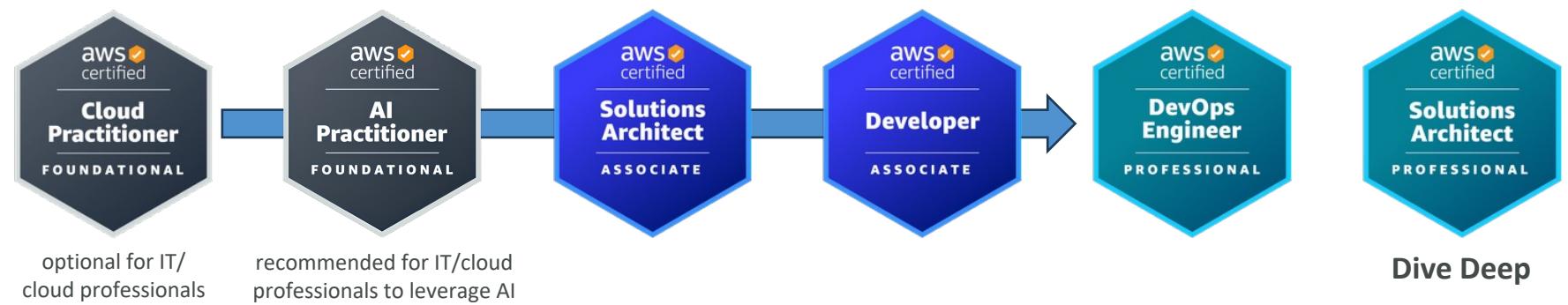
Design, develop, and manage cloud infrastructure and assets, work with DevOps to migrate applications to the cloud



## Architecture

### Application Architect

Design significant aspects of application architecture including user interface, middleware, and infrastructure, and ensure enterprise-wide scalable, reliable, and manageable systems



[https://d1.awsstatic.com/training-and-certification/docs/AWS\\_certification\\_paths.pdf](https://d1.awsstatic.com/training-and-certification/docs/AWS_certification_paths.pdf)

# AWS Certification Paths – Operations

## Operations

### Systems Administrator

Install, upgrade, and maintain computer components and software, and integrate automation processes



## Operations

### Cloud Engineer

Implement and operate an organization's networked computing infrastructure and Implement security systems to maintain data safety



# AWS Certification Paths – DevOps

## DevOps Test Engineer

Embed testing and quality best practices for software development from design to release, throughout the product life cycle



optional for IT/  
cloud professionals



Optional



recommended for IT/cloud  
professionals working on  
AI/ML projects



Dive Deep

## DevOps Cloud DevOps Engineer

Design, deployment, and operations of large-scale global hybrid cloud computing environment, advocating for end-to-end automated CI/CD DevOps pipelines



optional for IT/  
cloud professionals



recommended for IT/cloud  
professionals working on  
AI/ML projects

## DevOps DevSecOps Engineer

Accelerate enterprise cloud adoption while enabling rapid and stable delivery of capabilities using CI/CD principles, methodologies, and technologies

# AWS Certification Paths – Security

## Security Cloud Security Engineer

Design computer security architecture and develop detailed cyber security designs.  
Develop, execute, and track performance of security measures to protect information



## Security Cloud Security Architect

Design and implement enterprise cloud solutions applying governance to identify, communicate, and minimize business and technical risks

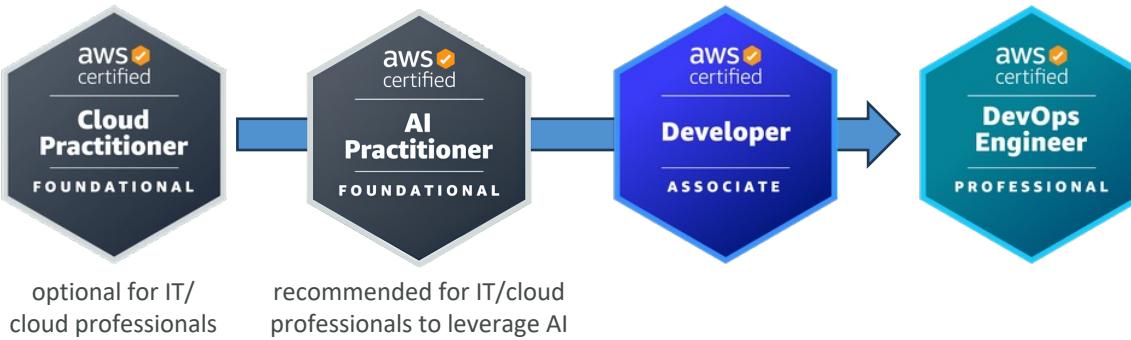


# AWS Certification Paths – Development & Networking

## Development

### Software Development Engineer

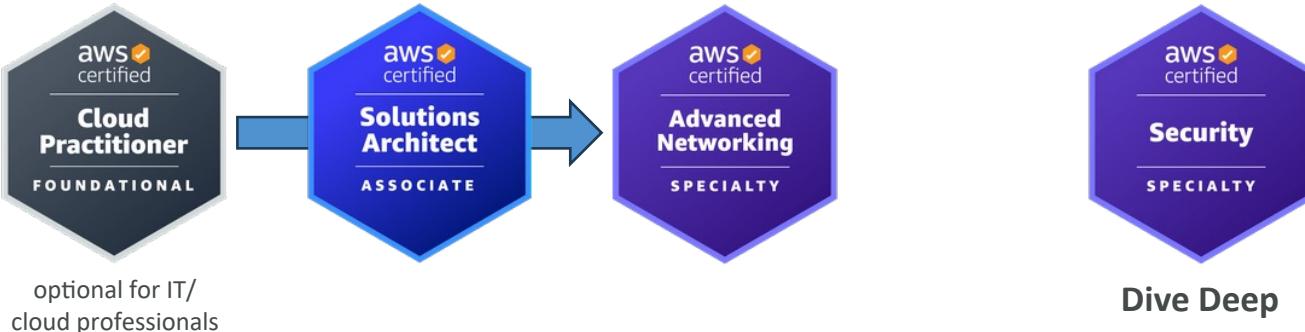
Develop, construct, and maintain software across platforms and devices



## Networking

### Network Engineer

Design and implement computer and information networks, such as local area networks (LAN), wide area networks (WAN), intranets, extranets, etc.



# AWS Certification Paths – Data Analytics & AI/ML

## Data Analytics

### Cloud Data Engineer

Automate collection and processing of structured/semi-structured data and monitor data pipeline performance



optional for IT/  
cloud professionals



recommended for IT/cloud  
professionals working on  
AI/ML projects



Dive Deep

## AI/ML

### Machine Learning Engineer

Research, build, and design artificial intelligence (AI) systems to automate predictive models, and design machine learning systems, models, and schemes



optional for IT/  
cloud professionals

optional for AI/ML  
professionals



Dive Deep



# AWS Certification Paths – AI/ML

## AI/ML

### Prompt Engineer

Design, test, and refine text prompts to optimize the performance of AI language models



optional for IT/  
cloud professionals



Dive Deep

## AI/ML

### Machine Learning Ops Engineer

Build and maintain AI and ML platforms and infrastructure. Design, implement, and operationally support AI/ML model activity and deployment infrastructure



optional for IT/  
cloud professionals

optional for AI/ML  
professionals



## AI/ML

### Data Scientist

Develop and maintain AI/ML models to solve business problems. Train and fine tune models and evaluate their performance



optional for IT/  
cloud professionals

optional for AI/ML  
professionals



# 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 SysOps