

SECTION 1

Let's Get Started!

The SAA-C03 Exam





The SAA-C03 Exam

Level: Associate

Length: 130 minutes

Format: 65 questions

Cost: \$150 USD

Delivery Method: Testing center or online

Scoring:

- Scaled score between 100 – 1000
- Minimum passing score of 720



The SAA-C03 Exam

Question format:

- **Multiple-choice:** Has one correct response and three incorrect responses
- **Multiple-response:** Has two or more correct responses out of five or more options



The SAA-C03 Exam

Domain 1: Design Secure Architectures

- Task Statement 1.1: Design secure access to AWS resources
- Task Statement 1.2: Design secure workloads and applications
- Task Statement 1.3: Determine appropriate data security controls

Domain 2: Design Resilient Architectures

- Task Statement 2.1: Design scalable and loosely coupled architectures
- Task Statement 2.2: Design highly available and/or fault-tolerant architectures



The SAA-C03 Exam

Domain 3: Design High-Performing Architectures

- Task Statement 3.1: Determine high-performing and/or scalable storage solutions
- Task Statement 3.2: Design high-performing and elastic compute solutions
- Task Statement 3.3: Determine high-performing database solutions
- Task Statement 3.4: Determine high-performing and/or scalable network architectures
- Task Statement 3.5: Determine high-performing data ingestion and transformation solutions



The SAA-C03 Exam

Domain 4: Design Cost-Optimized Architectures

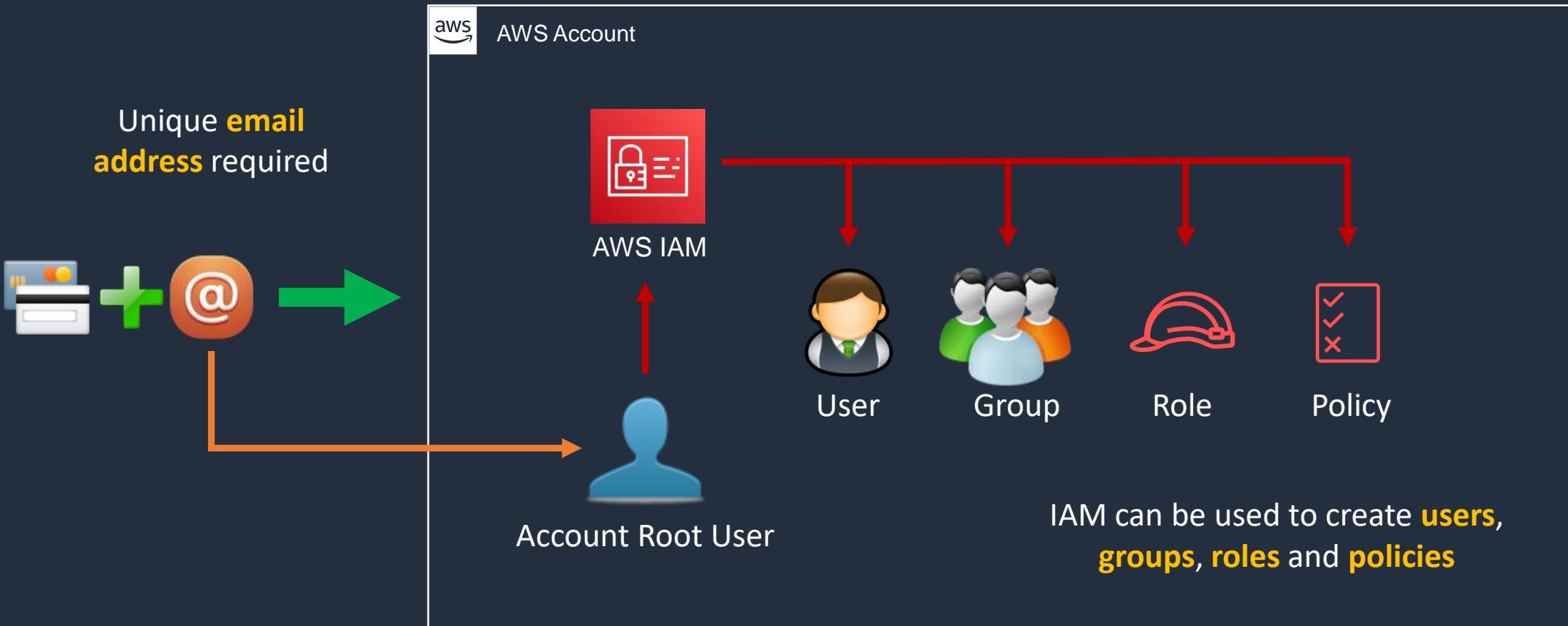
- Task Statement 4.1: Design cost-optimized storage solutions
- Task Statement 4.2: Design cost-optimized compute solutions
- Task Statement 4.3: Design cost-optimized database solutions
- Task Statement 4.4: Design cost-optimized network architectures

AWS Account Overview



AWS Account Overview

It's an IAM best practice to create **individual users**
and to avoid using the **Root** account

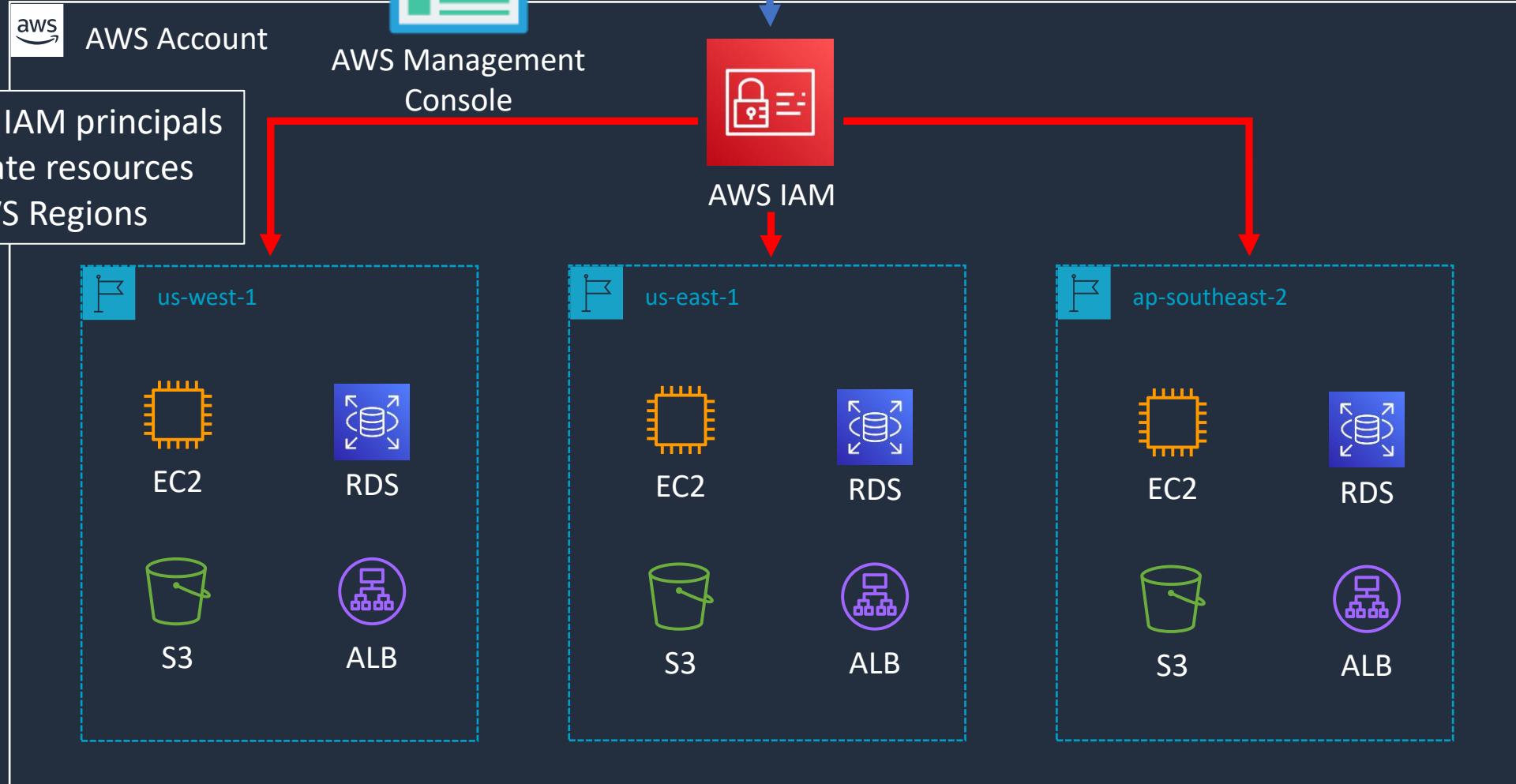


The Root user has **full control**
over the account



AWS Account Overview

Authentication: IAM principals authenticate to IAM using the console, API, or CLI



All AWS **identities** and **resources** are created
within the AWS account

Create your AWS Free Tier Account



aws What you need...



Credit card for setting up the account and paying any bills



Unique email address for this account

john@gmail.com



Check if you can use a **dynamic alias** with an existing email address



john+ACCOUNT-ALIAS-1@gmail.com

john+ACCOUNT-ALIAS-2@gmail.com



AWS account name / alias



Phone to receive an **SMS** verification code

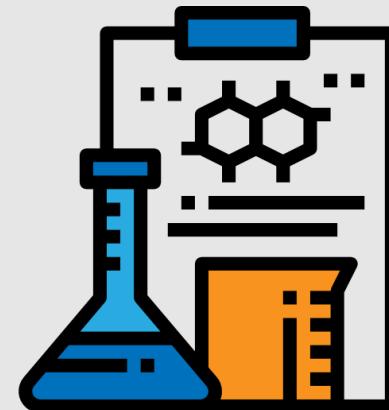
Configure Account and Create a Budget



Account Configuration

- Configure **Account Alias**
- Enable access to billing for **IAM users**
- Update **billing preferences**
- Create a **budget and alarm**

Install Tools and Configure AWS CLI





Install Tools and Configure AWS CLI

- ✓ Download the code (from the *next* lesson)
- ✓ Install Visual Studio Code
- ✓ Install and Configure the AWS CLI
- ✓ Access AWS CloudShell

SECTION 2

AWS Identity and Access Management (IAM)

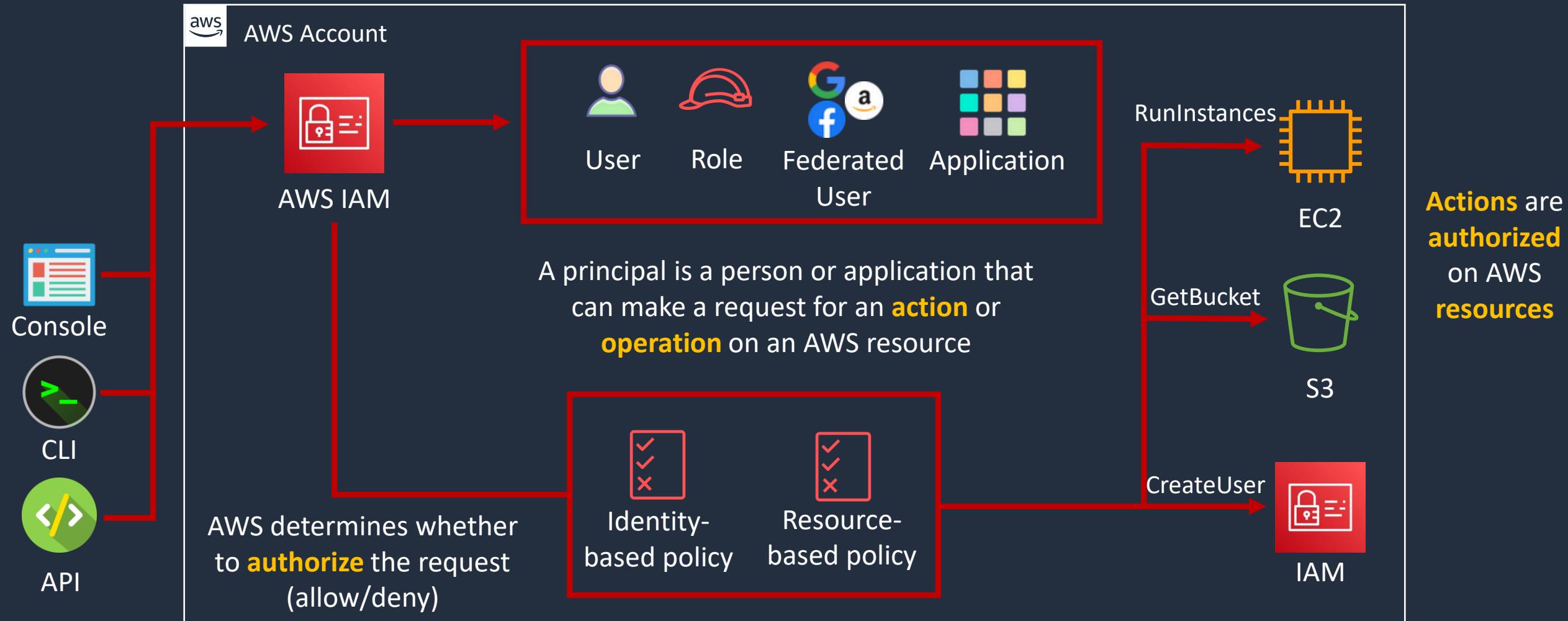
AWS Identity and Access Management (IAM)





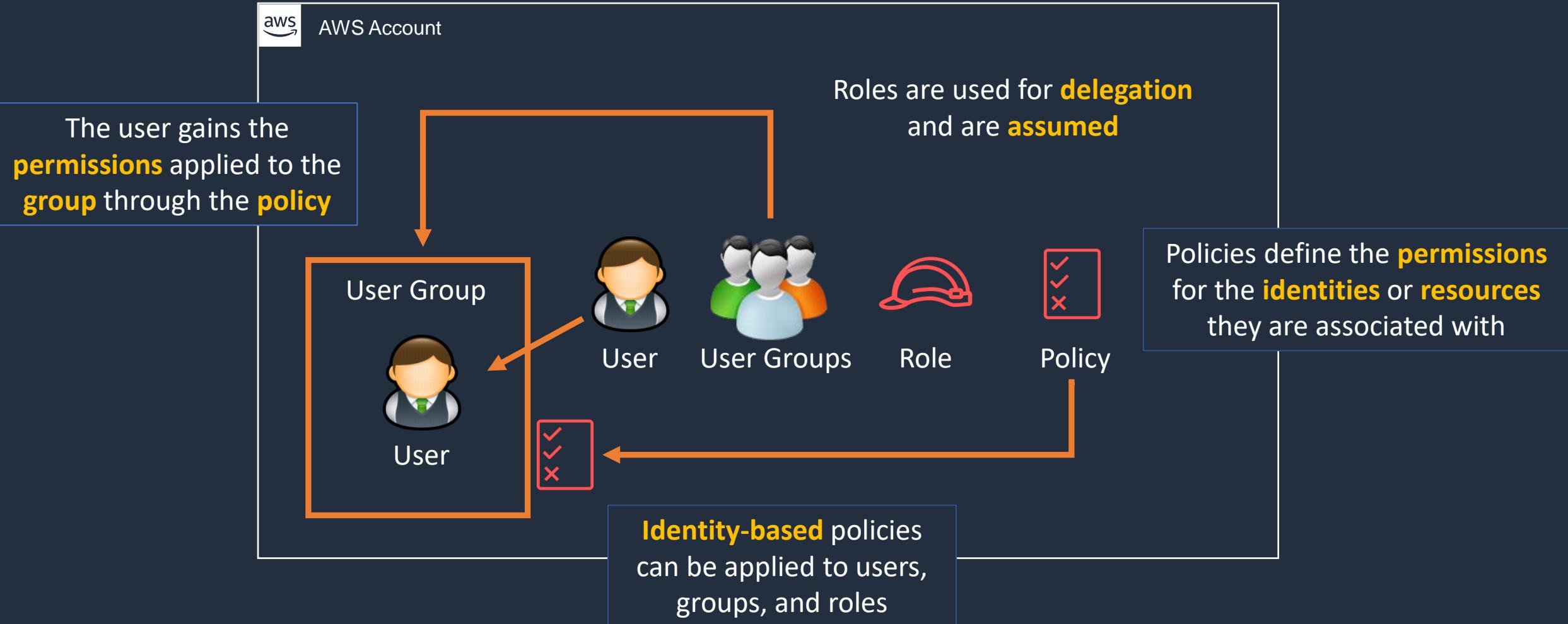
AWS Identity and Access Management (IAM)

IAM Principals must be **authenticated** to send requests (with a few exceptions)



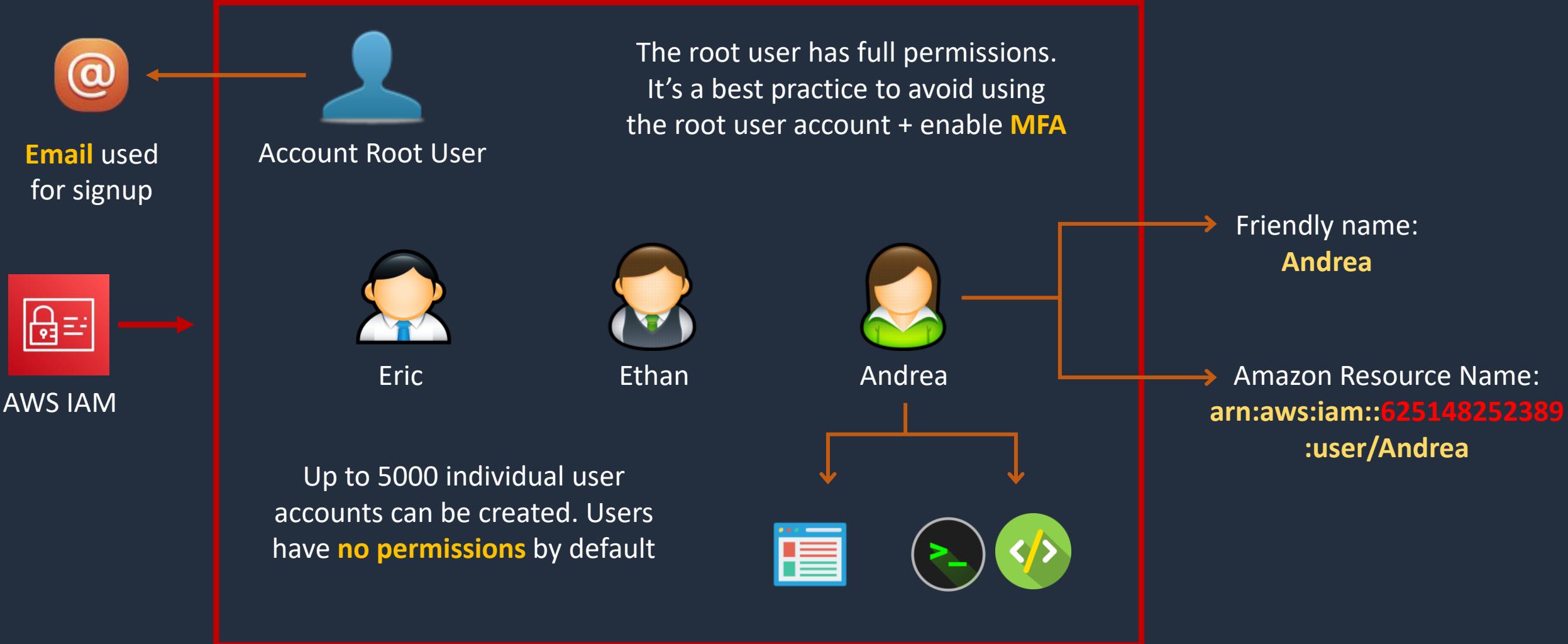


Users, User Groups, Roles and Policies



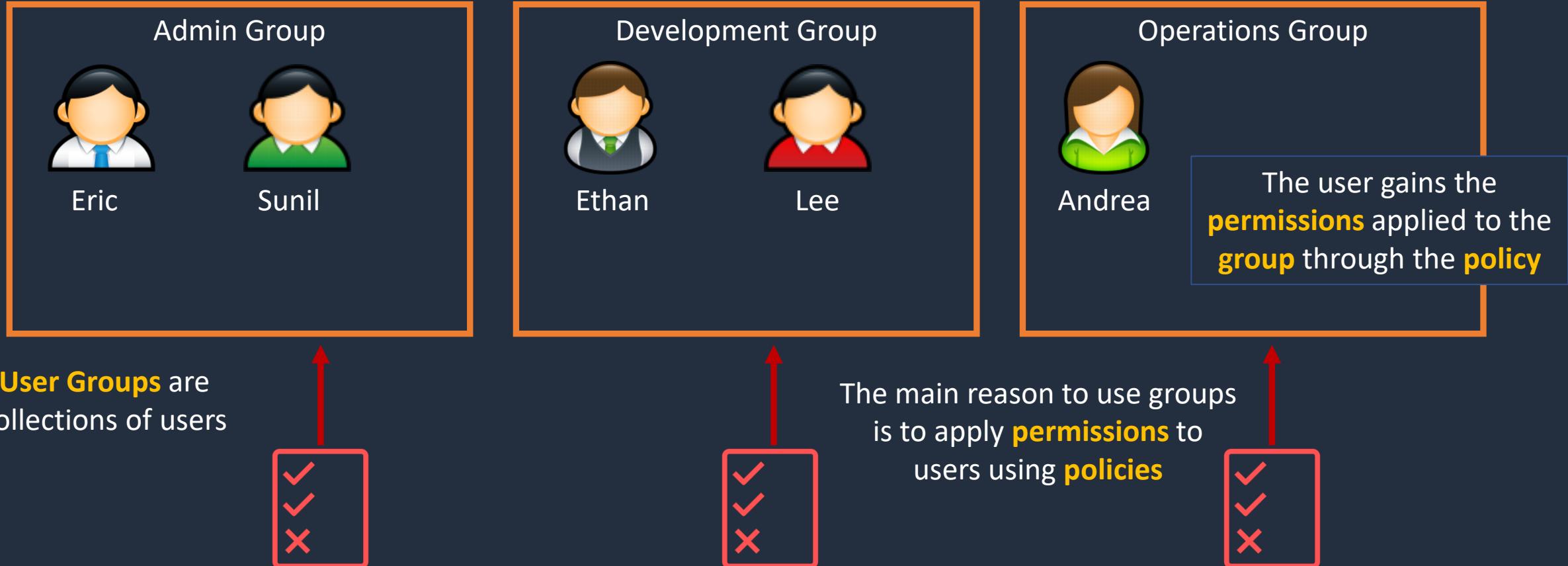


IAM Users



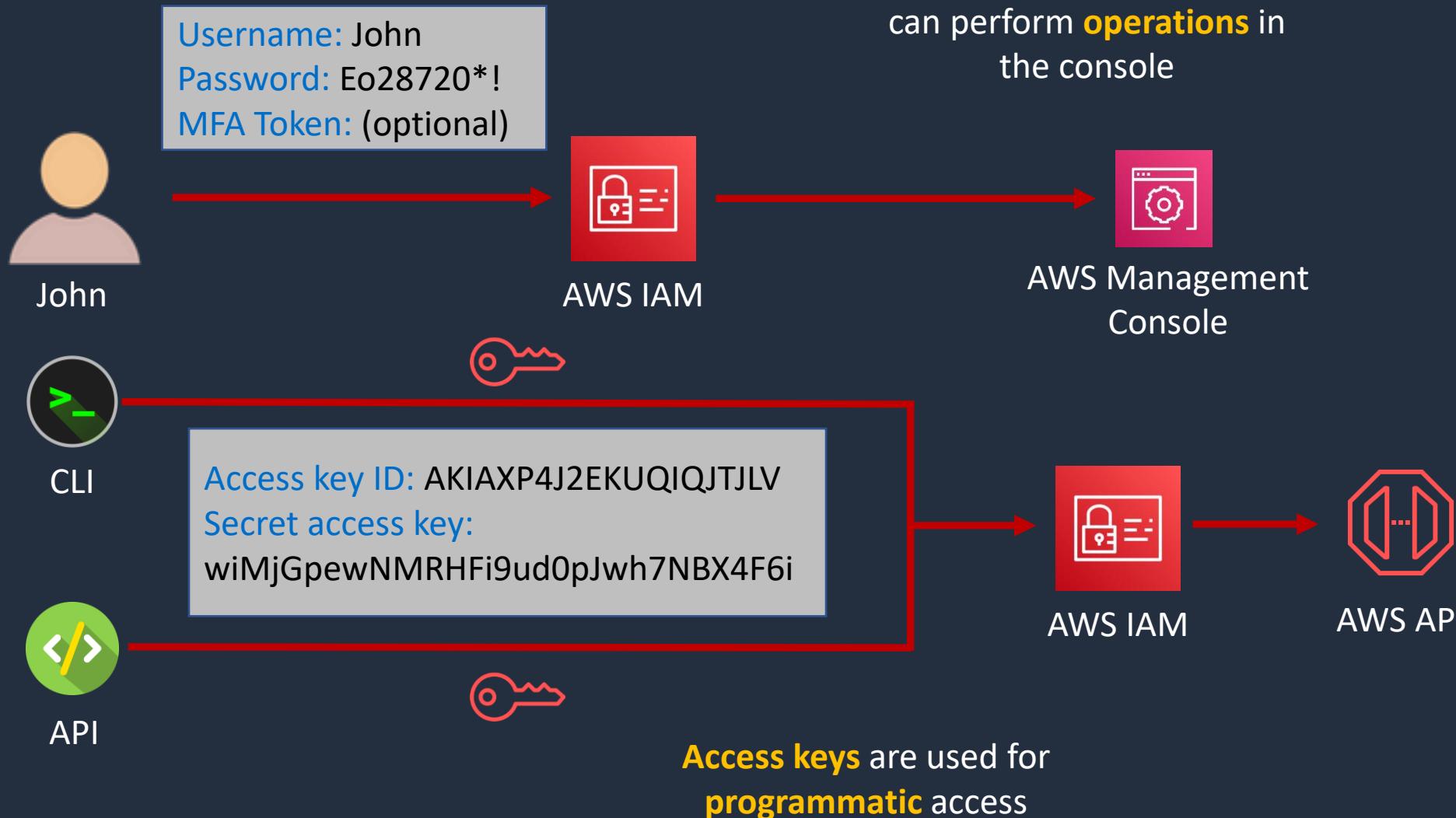


IAM User Groups





IAM Authentication Methods





Root User vs IAM User

User	Login Details	Permissions
 Root User	 Email address	 Full - Unrestricted
 IAM User	Friendly name: John + AWS account ID or Alias	 IAM Permissions Policy

Creating IAM Users and Groups

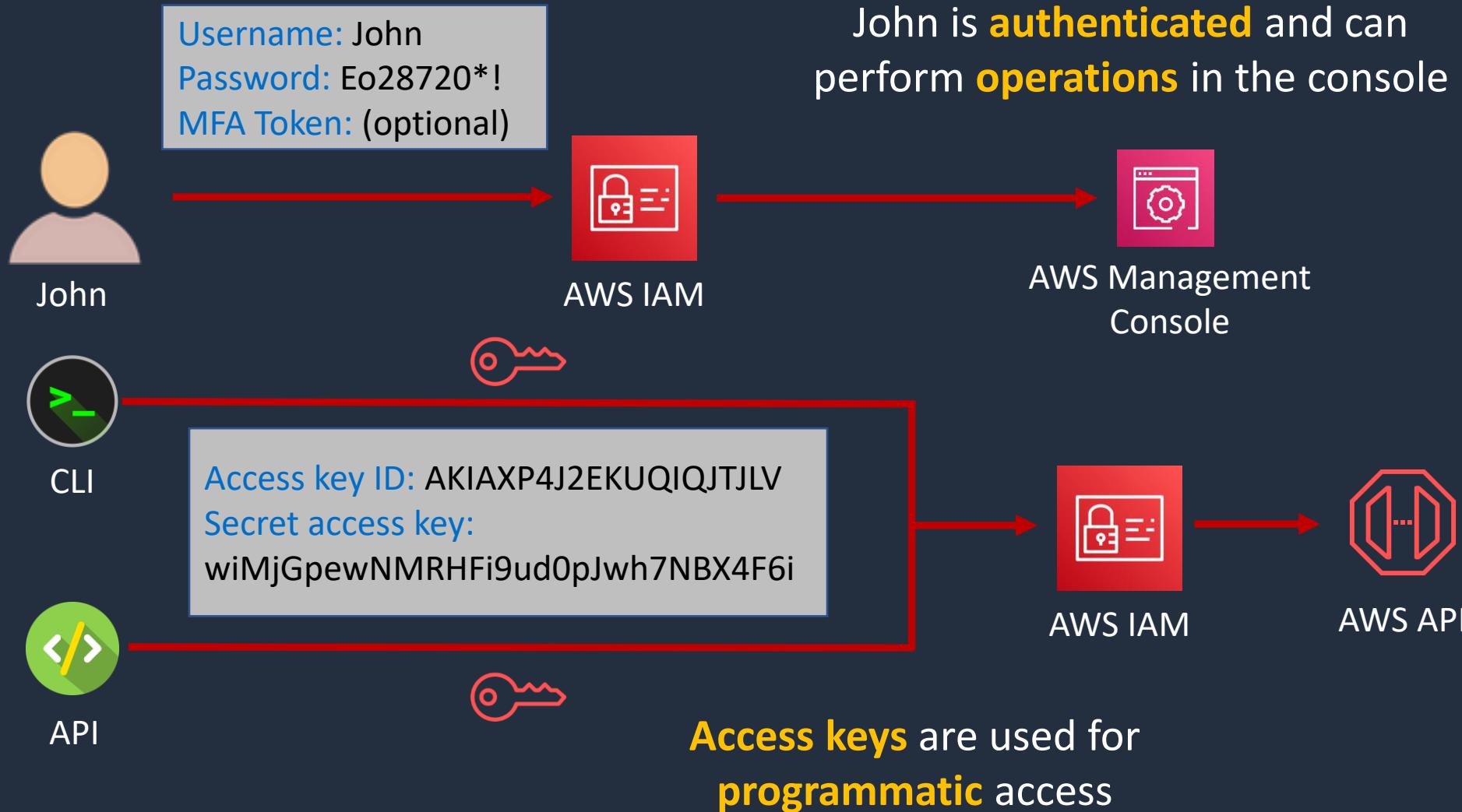


IAM Authentication and MFA





IAM Authentication Methods





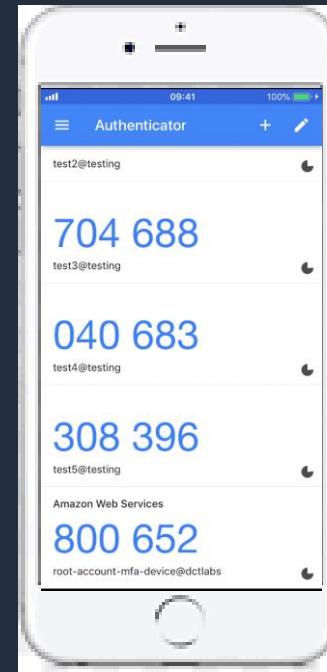
Multi-Factor Authentication

Something you **know**:

EJPx!*21p9%

Password

Something you **have**:



Something you **are**:





Multi-Factor Authentication

Something you **know**:



IAM User

EJPx!*21p9%

Password

Something you **have**:



MFA



MFA

Hardware device

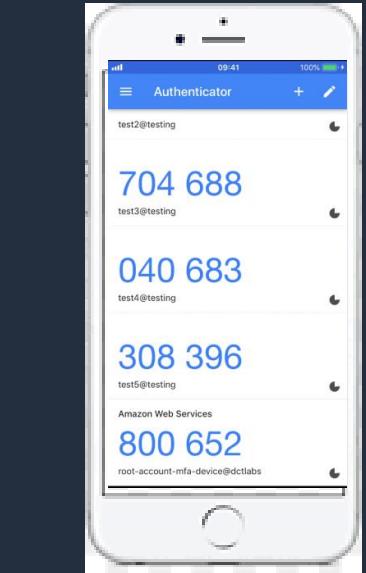


e.g. Google Authenticator on
your smart phone



MFA

Virtual MFA

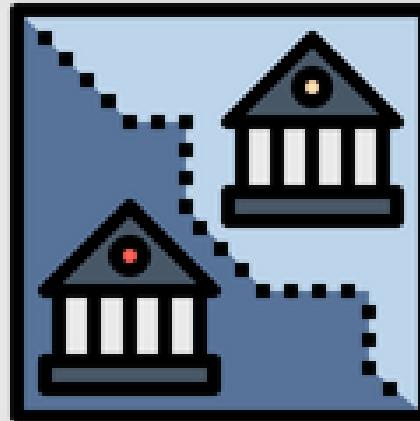


Security keys and **time-based
one-time password (TOTP)**
tokens

Setup Multi-Factor Authentication (MFA)

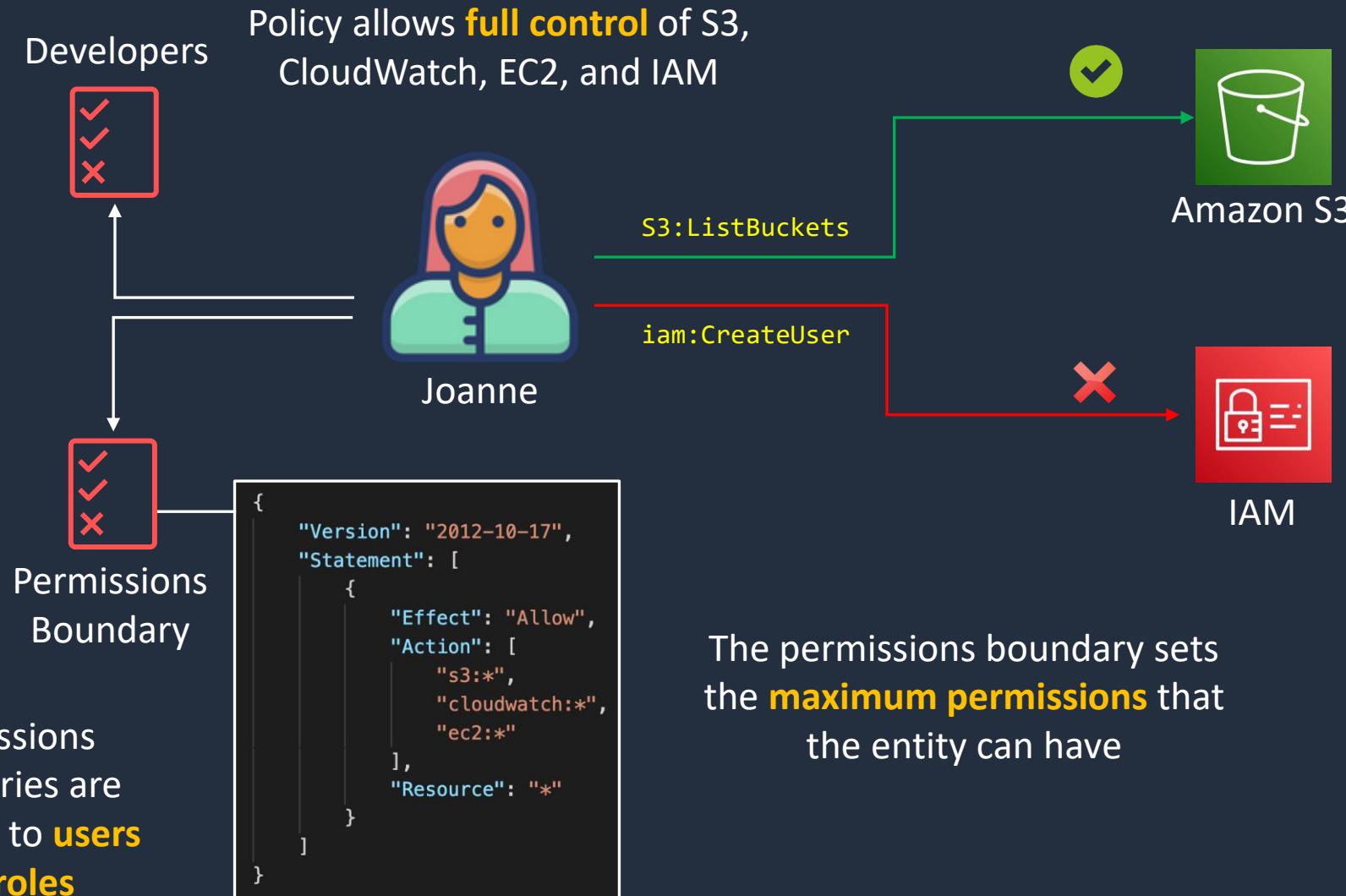


Permissions Boundaries





Permissions Boundaries



The **operation fails** because the permissions boundary does not allow it

The permissions boundary sets the **maximum permissions** that the entity can have



Privilege Escalation

IAMFullAccess



Lindsay

Lindsay is assigned permissions to AWS IAM only and **cannot** launch AWS resources

`iam:CreateUser`



IAM

Lindsay applies the **AdministratorAccess** policy to the X-User account

AdministratorAccess



X-User

Lindsay is now able to login with the X-User account and gain full privileges to the AWS account



AWS Batch



Lindsay mines bitcoins



DigitalCloud
TRAINING



Preventing Privilege Escalation

IAMFullAccess



Permissions
Boundary

The permissions boundary ensures that users created by Lindsay have the **same** or **fewer permissions**

Lindsay is assigned permissions to AWS IAM only and **cannot** launch AWS resources



Lindsay

iam:CreateUser



IAM

Lindsay applies the **AdministratorAccess** policy to the X-User account



X-User

AdministratorAccess



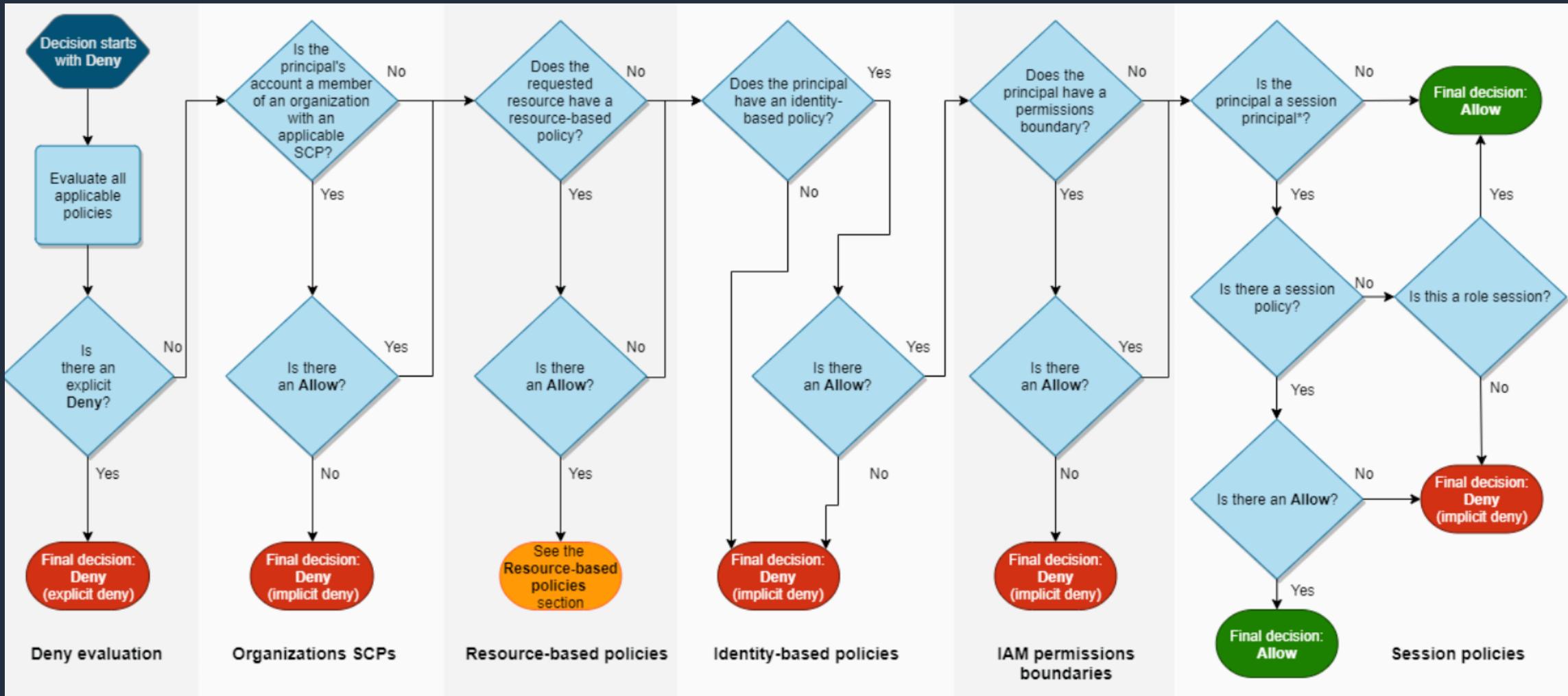
Lindsay does not have more privileges when logging in as X-User and cannot launch AWS resources

IAM Policy Evaluation





Evaluation Logic





Steps for Authorizing Requests to AWS

1. Authentication – AWS authenticates the principal that makes the request



Console



CLI



API



- Request context:**
- **Actions** – the actions or operations the principal wants to perform
 - **Resources** – The AWS resource object upon which actions are performed
 - **Principal** – The user, role, federated user, or application that sent the request
 - **Environment data** – Information about the IP address, user agent, SSL status, or time of day
 - **Resource data** – Data related to the resource that is being requested

2. Processing the request context

3. Evaluating all policies within the account



s3:GetObject



S3 Bucket

4. Determining whether a request is **allowed** or **denied**

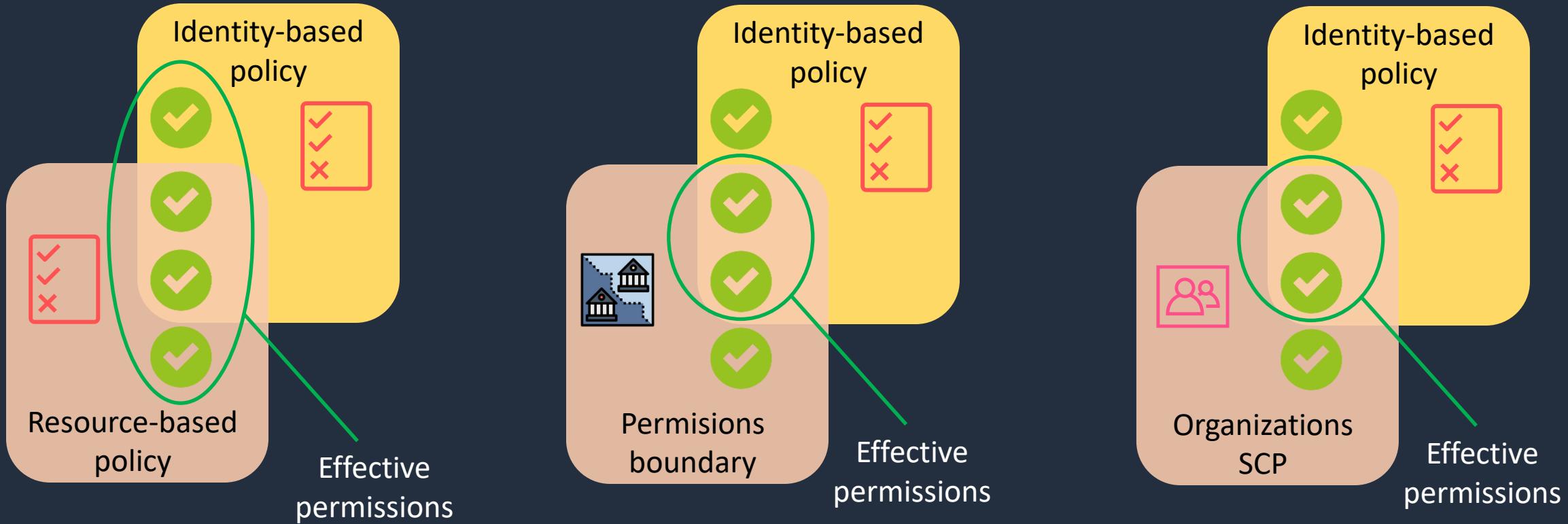


Types of Policy

- **Identity-based policies** – attached to users, groups, or roles
- **Resource-based policies** – attached to a resource; define permissions for a principal accessing the resource
- **IAM permissions boundaries** – set the maximum permissions an identity-based policy can grant an IAM entity
- **AWS Organizations service control policies (SCP)** – specify the maximum permissions for an organization or OU
- **Session policies** – used with AssumeRole* API actions



Evaluating Policies within an AWS Account

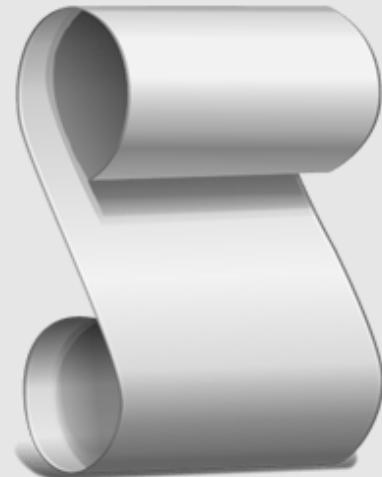




Determination Rules

1. By default, all requests are implicitly denied (though the root user has full access)
2. An explicit allow in an identity-based or resource-based policy overrides this default
3. If a permissions boundary, Organizations SCP, or session policy is present, it might override the allow with an implicit deny
4. An explicit deny in any policy overrides any allows

IAM Policy Structure





API Actions

Each AWS service has its own set of **actions** that describe **tasks** you can perform with that service

Amazon EC2

```
" "Action": "ec2:RunInstances"
```

Amazon RDS

```
" "Action": "rds:StopDBInstance"
```

AWS IAM

```
" "Action": "iam:ChangePassword"
```

Amazon S3

```
" "Action": "s3:GetObject"
```



Reading IAM Policies

All properties in a single statement block are evaluated together

A policy may contain more than one permission statement

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "s3:*",  
        "dynamodb:Describe*",  
        "dynamodb>List*",  
        "dynamodb:GetItem"  
      ],  
      "Resource": [  
        "arn:aws:s3:::mys3bucket",  
        "arn:aws:s3:::mys3bucket/*",  
        "arn:aws:dynamodb:us-east-1:111122223333:table/mytable"  
      ]  
    }  
  ]  
}
```



Reading IAM Policies

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "s3:*",  
                "dynamodb:Describe*",  
                "dynamodb>List*",  
                "dynamodb:GetItem"  
            ],  
            "Resource": [  
                "arn:aws:s3:::mys3bucket",  
                "arn:aws:s3:::mys3bucket/*",  
                "arn:aws:dynamodb:us-east-1:111122223333:table/mytable"  
            ]  
        }  
    ]  
}
```

A * is a wildcard

The **effect** is either *allow* or *deny*

Action lists the specific resource operations that the policy affects

Resource lists the specific resources that the policy applies to



Permission Statements - Conditions

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Principal": "*",  
      "Action": "s3:GetObject",  
      "Resource": "arn:aws:s3:::example-bucket/*",  
      "Condition": {  
        "DateGreaterThan": {"aws:CurrentTime": "2025-01-01T09:00:00Z"},  
        "DateLessThan": {"aws:CurrentTime": "2025-12-31T17:00:00Z"},  
        "StringEquals": {  
          "aws:DayOfWeek": ["Mon", "Tue", "Wed", "Thu", "Fri"]  
        }  
      }  
    }  
  ]  
}
```

Conditions allow for context-based decisions. In this example, access is allowed Mon-Fri during business hours



IAM Policy Example – Source IP Address Condition

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": ["ec2:TerminateInstances"],  
            "Resource": ["*"]  
        },  
        {  
            "Effect": "Deny",  
            "Action": ["ec2:TerminateInstances"],  
            "Condition": {  
                "NotIpAddress": {  
                    "aws:SourceIp": [  
                        "192.0.2.0/24",  
                        "203.0.113.0/24"  
                    ]  
                }  
            },  
            "Resource": ["*"]  
        }  
    ]  
}
```

The specific **API action** is defined

The effect is to deny the API action if the **IP address** is not in the specified range



IAM Policy Example – Encryption Condition

```
{  
  "Version": "2012-10-17",  
  "Id": "ExamplePolicy01",  
  "Statement": [  
    {  
      "Sid": "ExampleStatement01",  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": "*"  
      },  
      "Action": [  
        "elasticfilesystem:ClientRootAccess",  
        "elasticfilesystem:ClientMount",  
        "elasticfilesystem:ClientWrite"  
      ],  
      "Condition": {  
        "Bool": {  
          "aws:SecureTransport": "true"  
        }  
      }  
    }  
  ]  
}
```

You can tell this is a **resource-based policy** as it has a **principal element** defined

The policy grants **read and write** access to an EFS file systems to all **IAM principals** ("AWS ":"*")

the policy **condition** element requires that **SSL/TLS encryption** is used



IAM Policy Example – Prefix Condition

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Action": ["s3>ListBucket"],  
            "Effect": "Allow",  
            "Resource": ["arn:aws:s3:::mybucket"],  
            "Condition": {"StringLike": {"s3:prefix": ["${aws:username}/*"]}}  
        },  
        {  
            "Action": [  
                "s3:GetObject",  
                "s3:PutObject"  
            ],  
            "Effect": "Allow",  
            "Resource": ["arn:aws:s3:::mybucket/${aws:username}/*"]  
        }  
    ]  
}
```

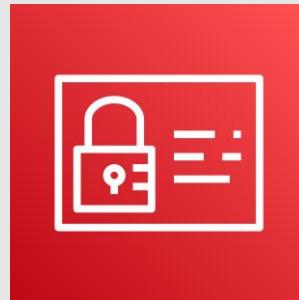
A variable is used for the **s3:prefix** that is replaced with the user's friendly name

The actions are allowed only within the **user's folder** within the bucket

Access Evaluation Tools



IAM Best Practices





AWS IAM Best Practices

- Require human users to use federation with an identity provider to access AWS using temporary credentials
- Require workloads to use temporary credentials with IAM roles to access AWS
- Require multi-factor authentication (MFA)
- Rotate access keys regularly for use cases that require long-term credentials
- Safeguard your root user credentials and don't use them for everyday tasks
- Apply least-privilege permissions
- Get started with AWS managed policies and move toward least-privilege permissions



AWS IAM Best Practices

- Use IAM Access Analyzer to generate least-privilege policies based on access activity
- Regularly review and remove unused users, roles, permissions, policies, and credentials
- Use conditions in IAM policies to further restrict access
- Verify public and cross-account access to resources with IAM Access Analyzer
- Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions
- Establish permissions guardrails across multiple accounts
- Use permissions boundaries to delegate permissions management within an account

Architecture Patterns – AWS IAM





Architecture Patterns – AWS IAM

Requirement

A select group of users only should be allowed to change their IAM passwords

An Amazon EC2 instance must be delegated with permissions to an Amazon DynamoDB table

A company has created their first AWS account. They need to assign permissions to users based on job function

Solution

Create a group for the users and apply a permissions policy that grants the iam:ChangePassword API permission

Create a role and assign a permissions policy to the role that grants access to the database service

Use AWS managed policies that are aligned with common job functions



Architecture Patterns – AWS IAM

Requirement

A solutions architect needs to restrict access to an AWS service based on the source IP address of the requester

Solution

Create an IAM permissions policy and use the Condition element to control access based on source IP address

A developer needs to make programmatic API calls from the AWS CLI

Instruct the developer to create a set of access keys and use those for programmatic access

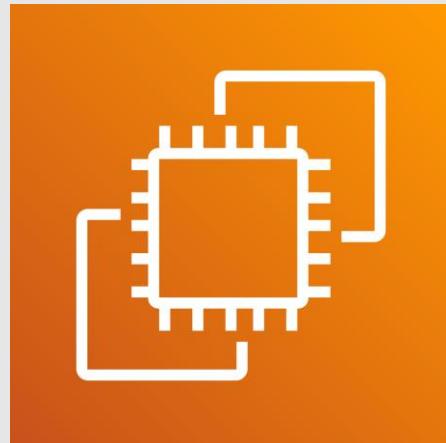
A group of users require full access to all Amazon EC2 API actions

Create a permissions policy that uses a wildcard for the Action element relating to EC2 (ec2:*)

SECTION 3

Amazon Elastic Compute Cloud (EC2)

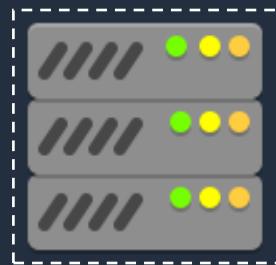
Amazon EC2 Overview



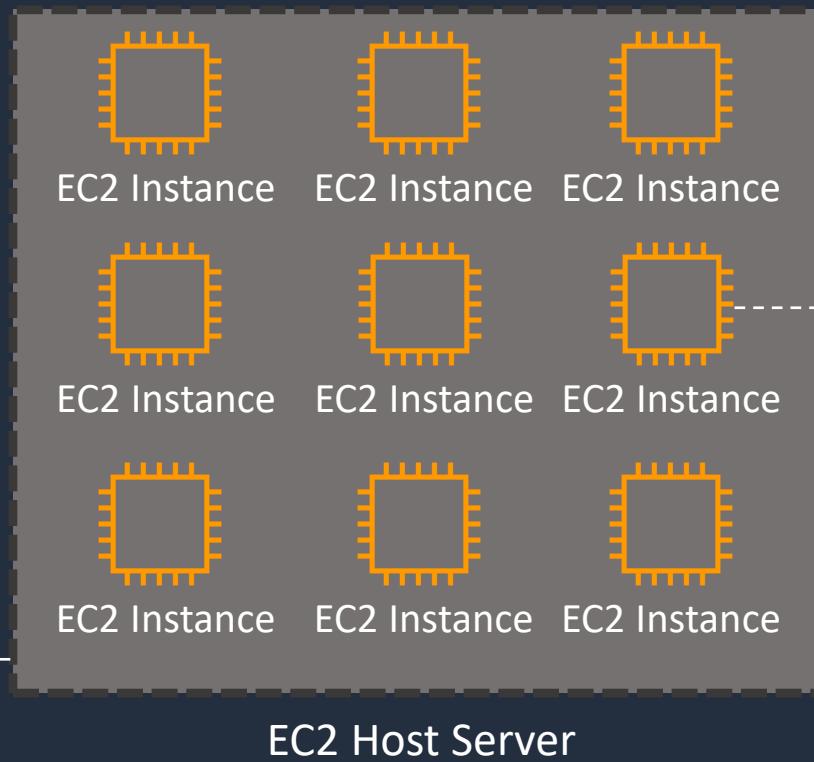


Amazon Elastic Compute Cloud (EC2)

EC2 hosts are
managed by AWS



EC2 instances run Windows,
Linux, or MacOS



An **EC2 instance** is
a virtual server



A selection of **instance types**
come with varying combinations
of CPU, memory, storage and
networking

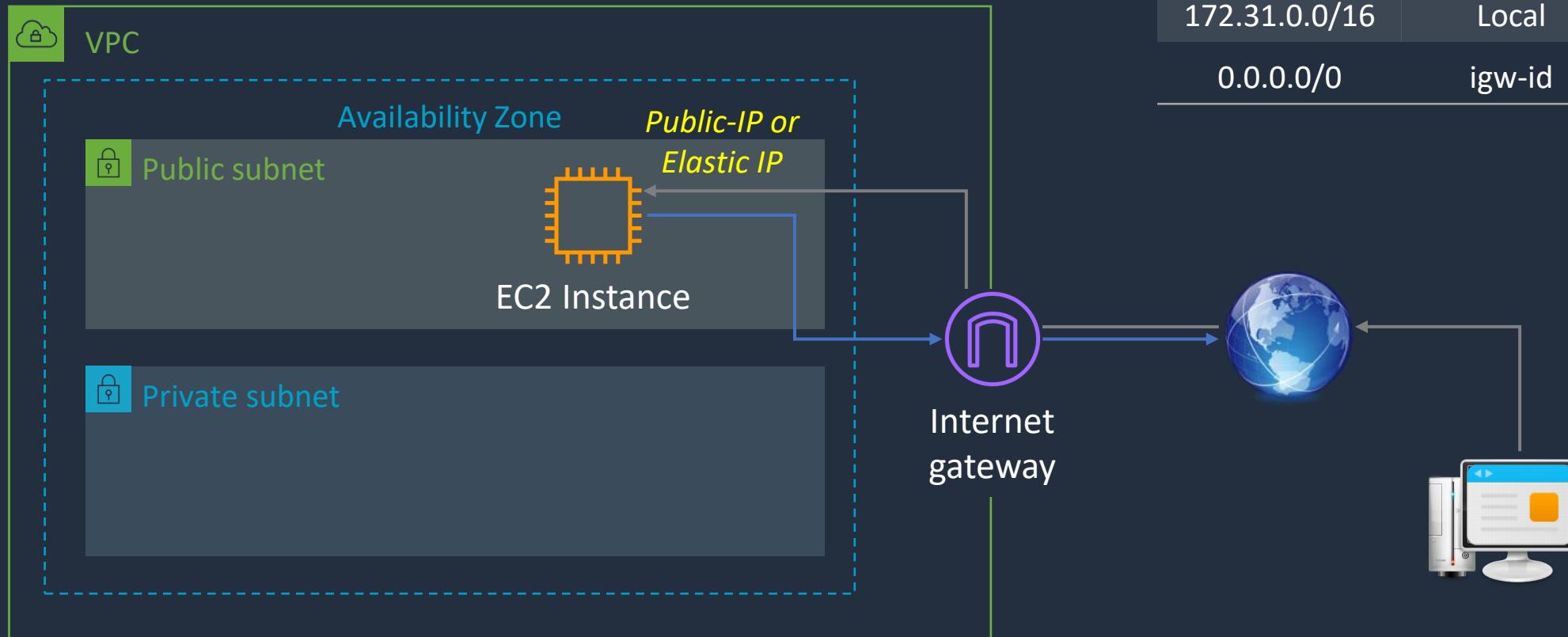


Public, Private, and Elastic IP addresses

Type	Description
Public IP address	<p>Lost when the instance is stopped</p> <p>Used in Public Subnets</p> <p>No charge</p> <p>Associated with a private IP address on the instance</p> <p>Cannot be moved between instances</p>
Private IP address	<p>Retained when the instance is stopped</p> <p>Used in Public and Private Subnets</p>
Elastic IP address	<p>Static Public IP address</p> <p>You are charged if not used</p> <p>Associated with a private IP address on the instance</p> <p>Can be moved between instances and Elastic Network Adapters</p>

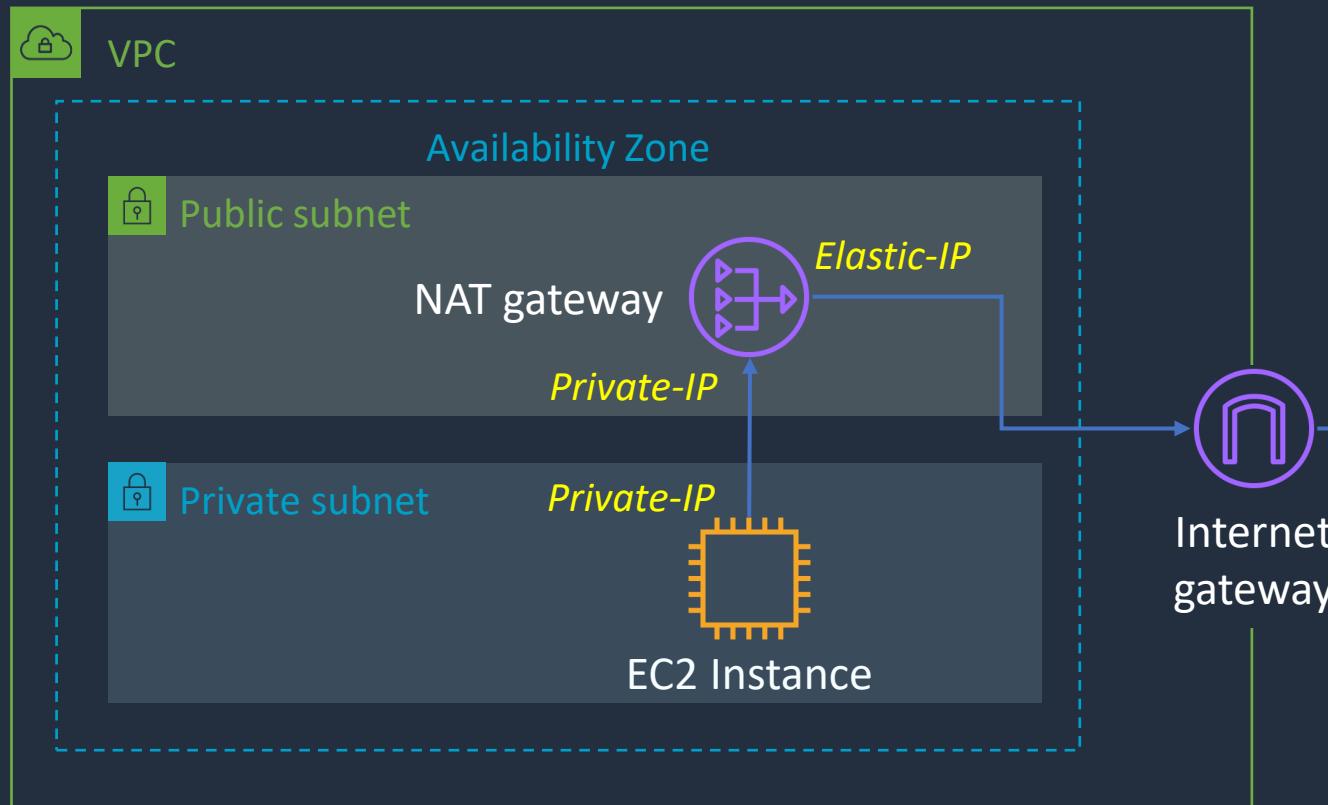


Public Subnets





Public Subnets



Public Subnet Route Table

Destination	Target
172.31.0.0/16	Local
0.0.0.0/0	igw-id

Private Subnet Route Table

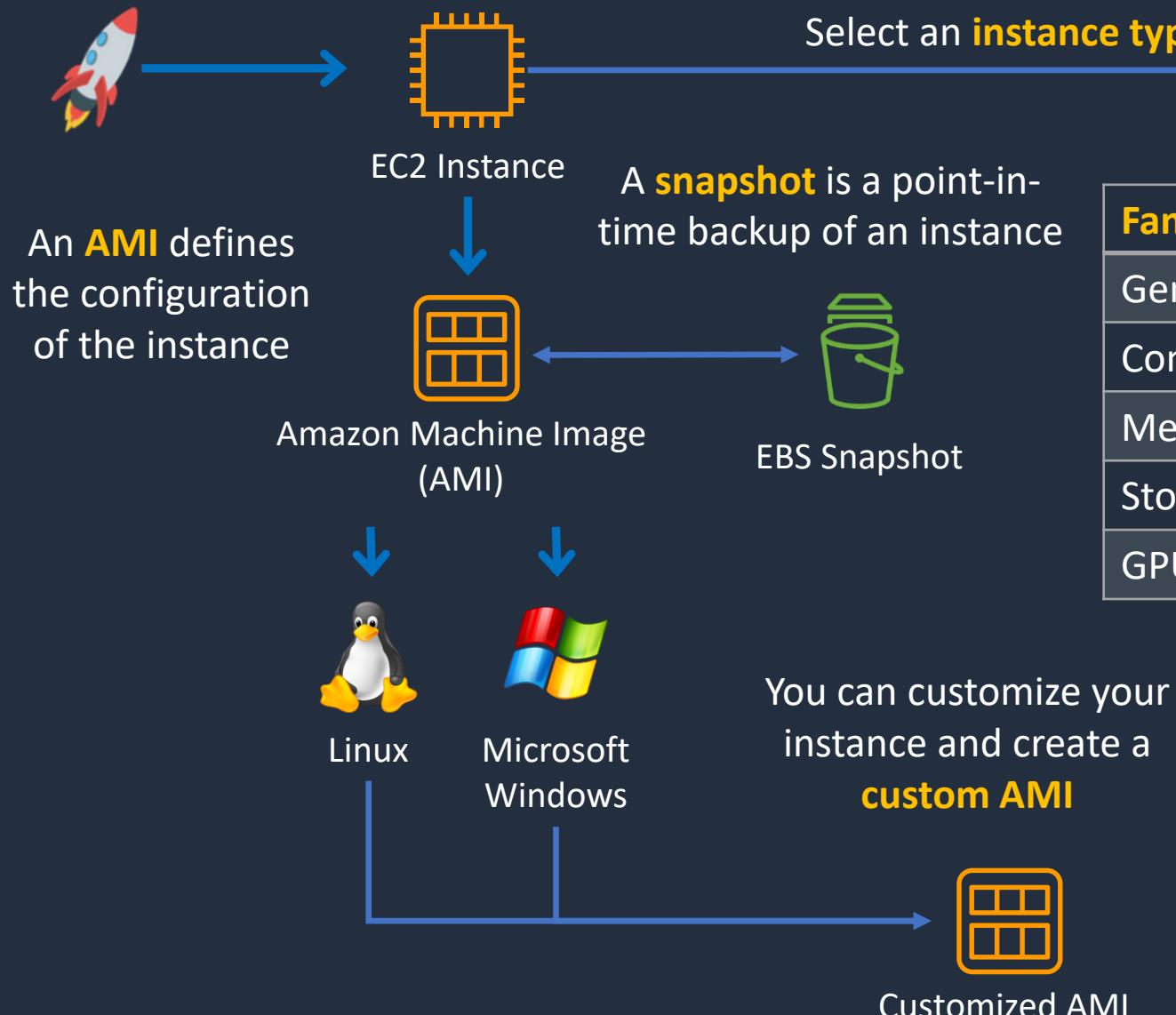
Destination	Target
172.31.0.0/16	Local
0.0.0.0/0	nat-gateway-id

Launching Amazon EC2 Instances





Launching an EC2 Instance



Connecting to Amazon EC2



Amazon EC2 User Data and Metadata





Amazon EC2 Metadata

- Instance metadata is data about your EC2 instance
- Instance metadata is available at <http://169.254.169.254/latest/meta-data>
- Examples:



```
[ec2-user@ip-172-31-42-248 ~]$ curl http://169.254.169.254/latest/meta-data
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hibernation/
hostname
identity-credentials/
instance-action
instance-id
instance-life-cycle
instance-type
local-hostname
local-ipv4
```



Amazon EC2 Metadata

- Examples ctd.:

```
[ec2-user@ip-172-31-42-248 ~]$ curl http://169.254.169.254/latest/meta-data/local-ipv4  
172.31.42.248[ec2-user@ip-172-31-42-248 ~]$
```

```
[ec2-user@ip-172-31-42-248 ~]$ curl http://169.254.169.254/latest/meta-data/public-ipv4  
3.26.54.18[ec2-user@ip-172-31-42-248 ~]$
```



IMDSv1 vs IMDSv2

Instance Metadata Service (IMDS) comes in two versions:

- **IMDSv1** – is older and less secure
- **IMDSv2** – is newer, more secure, and requires a session token for authorization
- Default EC2 launch settings may **disable IMDSv1**

Metadata accessible | [Info](#)

Enabled

Metadata transport

Select

Metadata version | [Info](#)

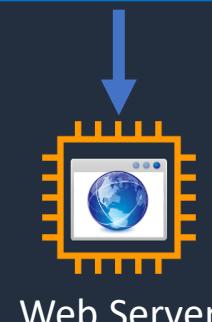
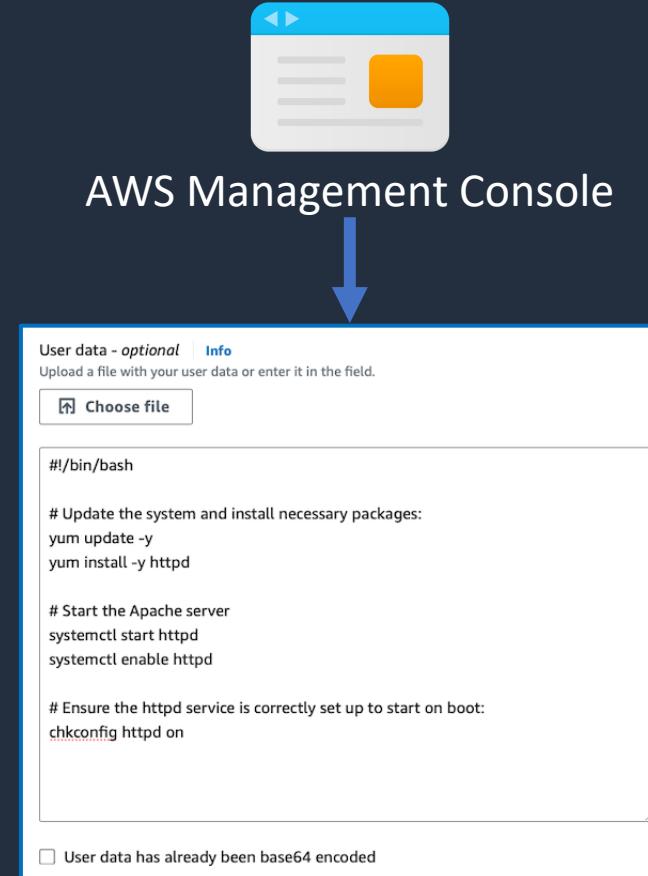
V2 only (token required)

⚠️ For V2 requests, you must include a session token in all instance metadata requests. Applications or agents that use V1 for instance metadata access will break.



Amazon EC2 User Data

The code is run when
the instance starts for
the **first time**



Batch and PowerShell
scripts can be run on
Windows



Amazon EC2 User Data via the AWS CLI

```
aws ec2 run-instances --instance-type t2.micro --image-id  
ami-0440d3b780d96b29d --user-data file://user_data.sh
```

The user data is supplied by
specifying the file

```
#!/bin/bash  
yum update -y  
yum install -y httpd  
systemctl start httpd  
systemctl enable httpd  
chkconfig httpd on
```



Amazon EC2 User Data

- User data must be **base64-encoded**
- Encoding is automatic with the console and AWS CLI
- User data is limited to **16 KB**, in raw form, before it is base64-encoded
- User data only runs the **first time** you launch your instance

Launch Instance with User Data and Metadata

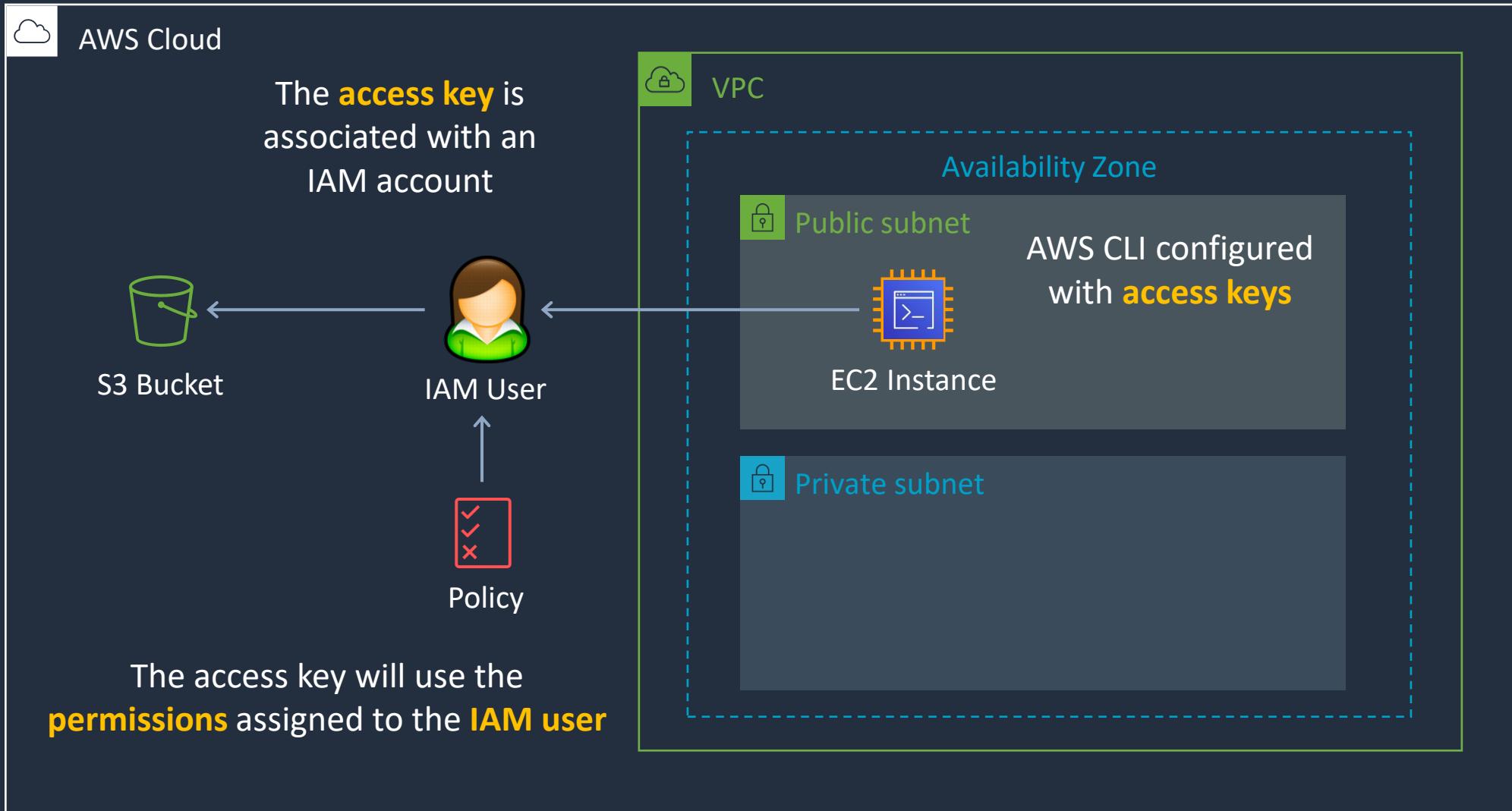


Access Keys and IAM Roles with EC2



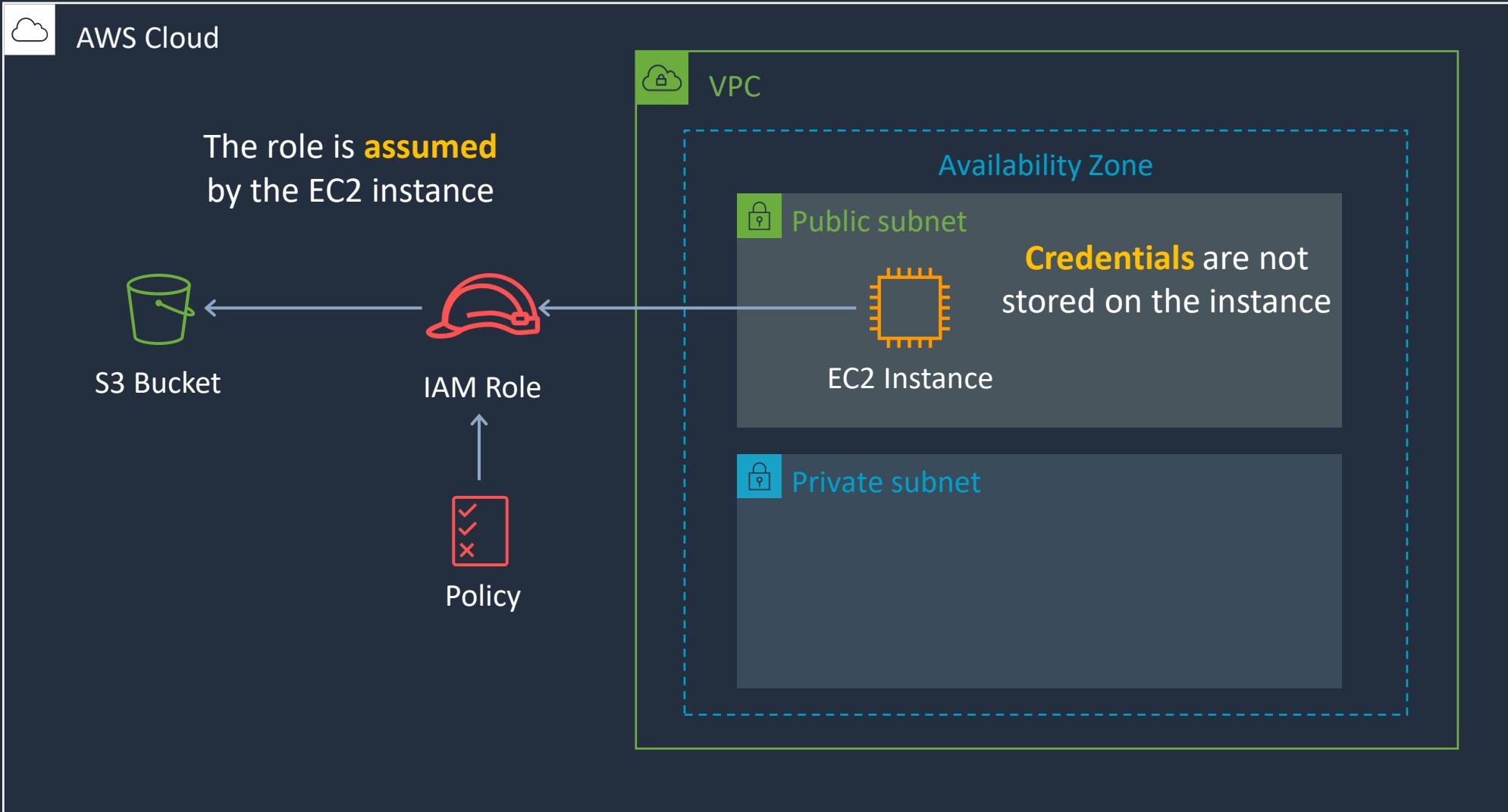


Using Access Keys with Amazon EC2





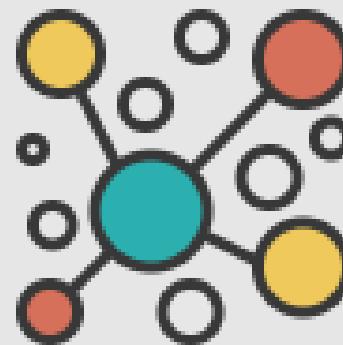
Using Access Keys with Amazon EC2



Practice with Access Keys and IAM Roles



EC2 Placement Groups



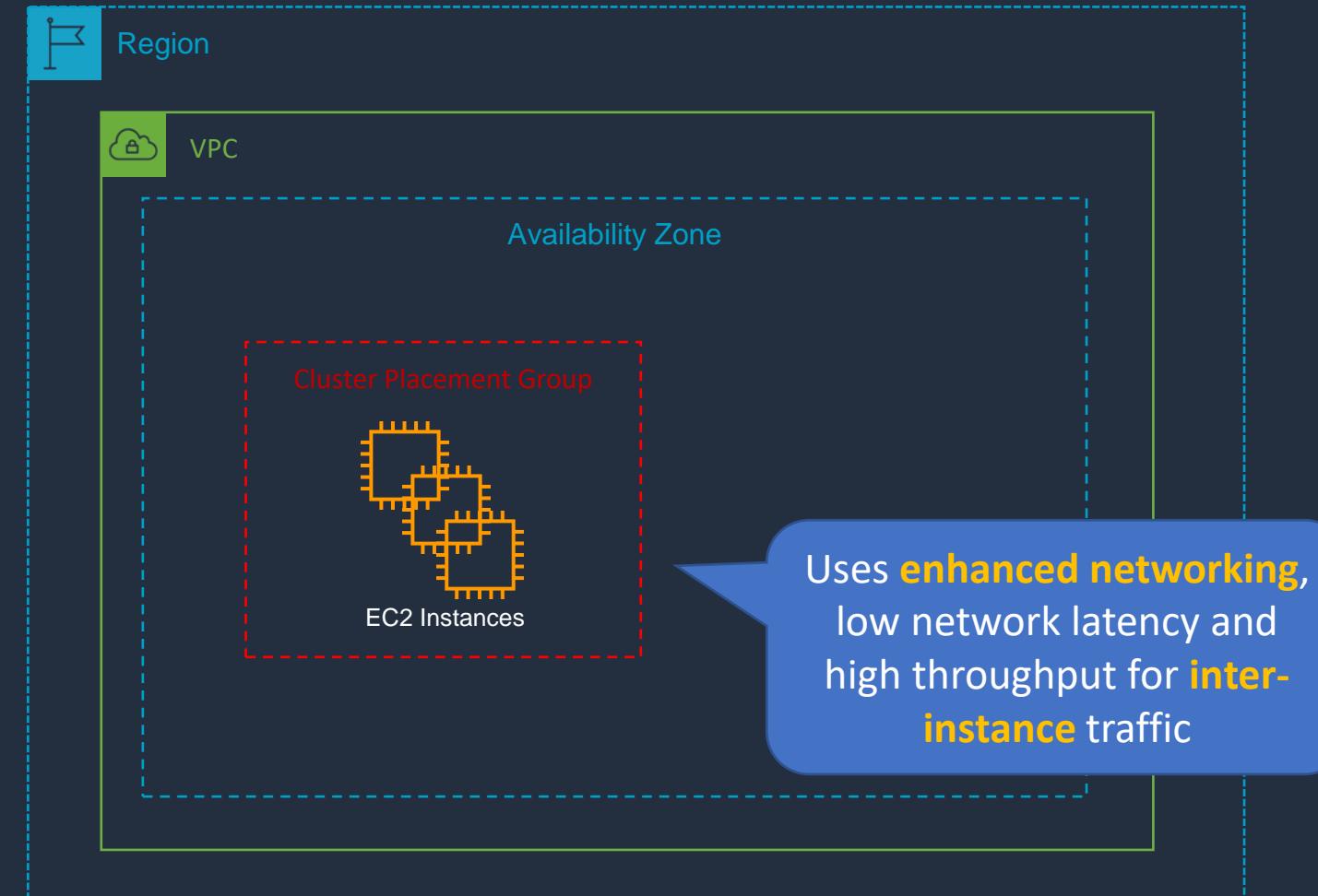


EC2 Placement Groups

- **Cluster** – packs instances close together inside an Availability Zone. This strategy enables workloads to achieve the **low-latency** network performance necessary for **tightly-coupled** node-to-node communication that is typical of **HPC applications**
- **Partition** – spreads your instances across logical partitions such that groups of instances in one partition **do not share the underlying hardware** with groups of instances in different partitions. This strategy is typically used by large **distributed and replicated workloads**, such as Hadoop, Cassandra, and Kafka
- **Spread** – strictly places a small group of instances across **distinct underlying hardware** to reduce correlated failures



Cluster Placement Group





Partition Placement Group



Region



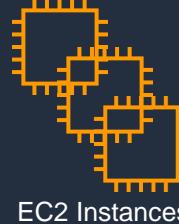
VPC

Availability Zone

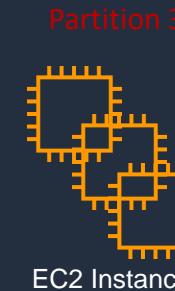
Each partition is located on
a **separate AWS rack**



Partition 1



Availability Zone

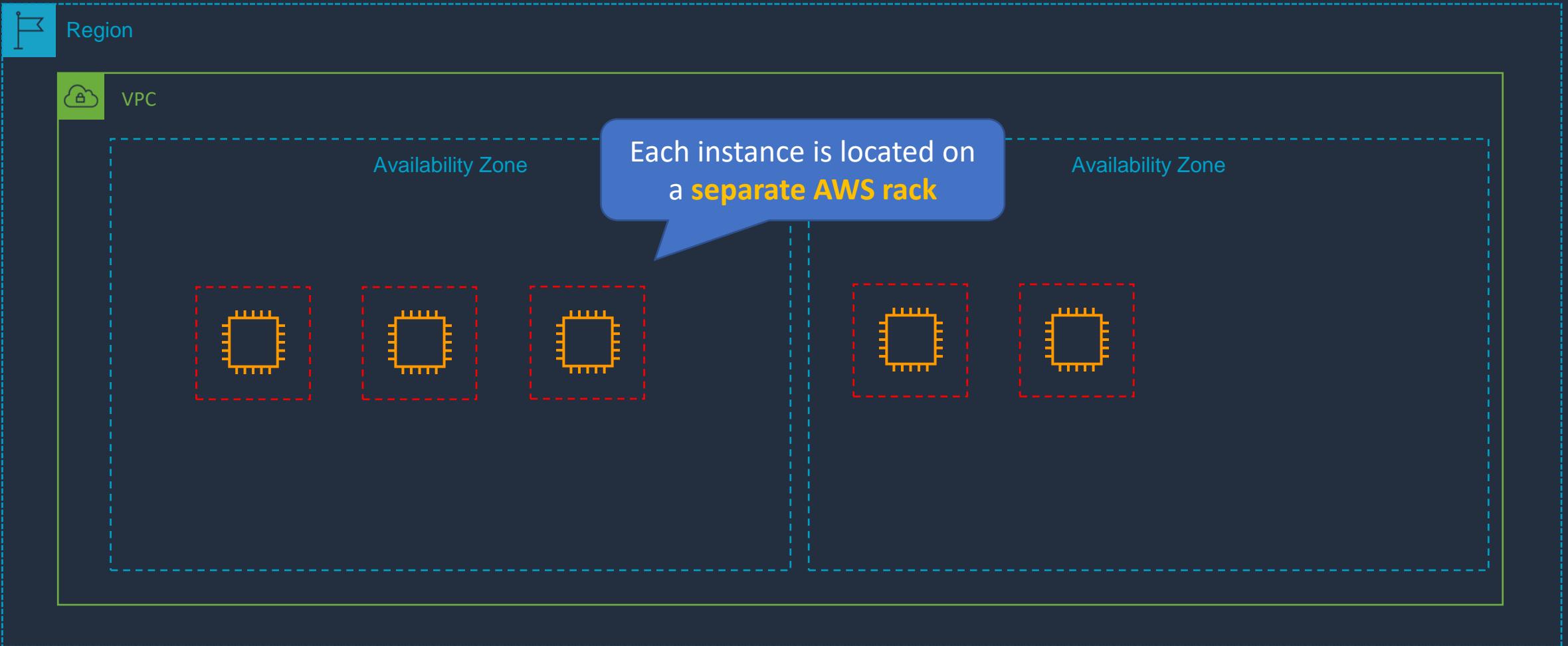


Partition 3

Partitions can be
in **multiple AZs**
(up to 7 per AZ)

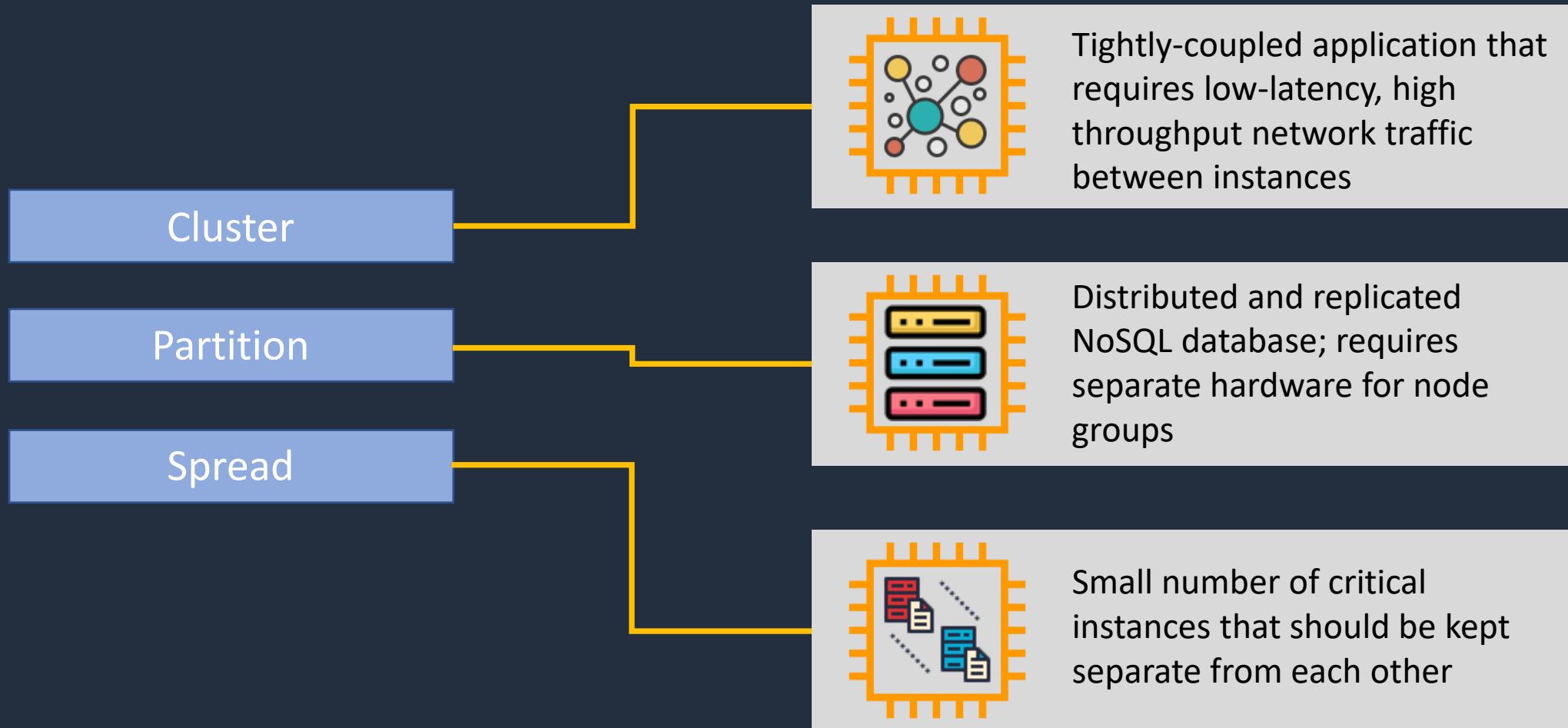


Spread Placement Group





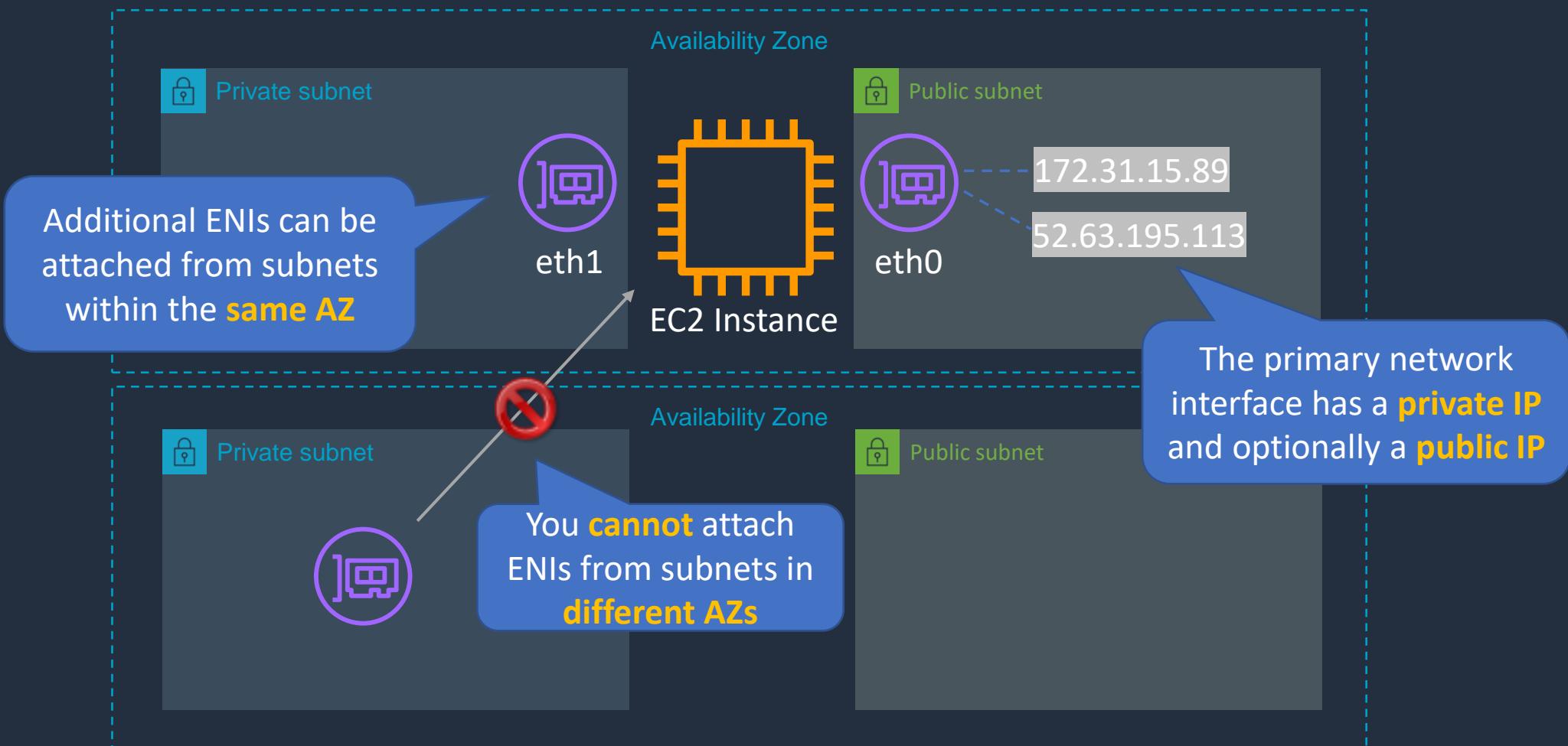
EC2 Placement Group Use Cases



Network Interfaces (ENI, ENA, EFA)



Network Interfaces (ENI, ENA, EFA)





Network Interfaces (ENI, ENA, EFA)



Elastic network
interface

- Basic adapter type for when you don't have any high-performance requirements
- Can use with all instance types



Elastic network
adapter

- Enhanced networking performance
- Higher bandwidth and lower inter-instance latency
- Must choose supported instance type



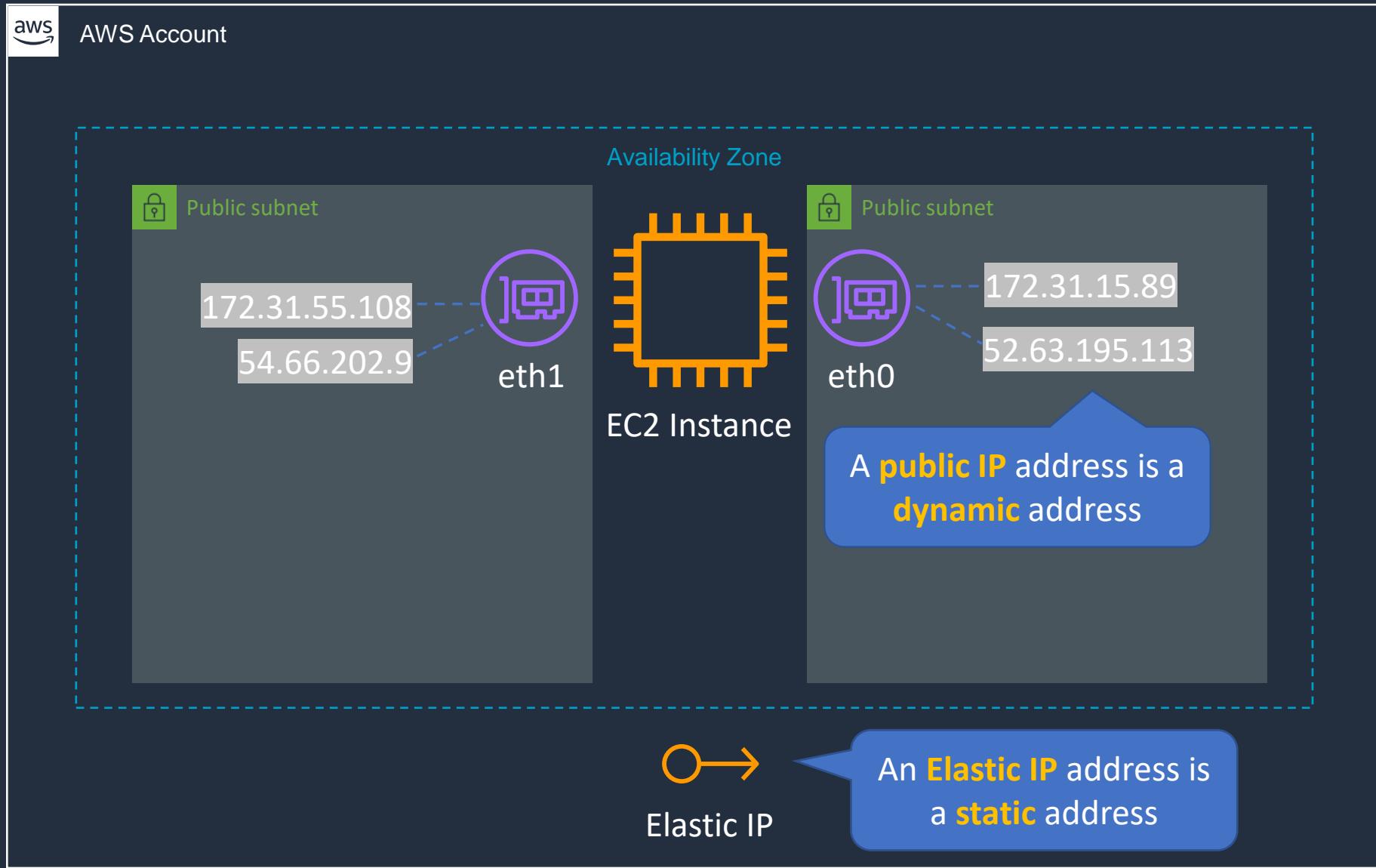
Elastic Fabric
Adapter

- Use with High Performance Computing and MPI and ML use cases
- Tightly coupled applications
- Can use with all instance types

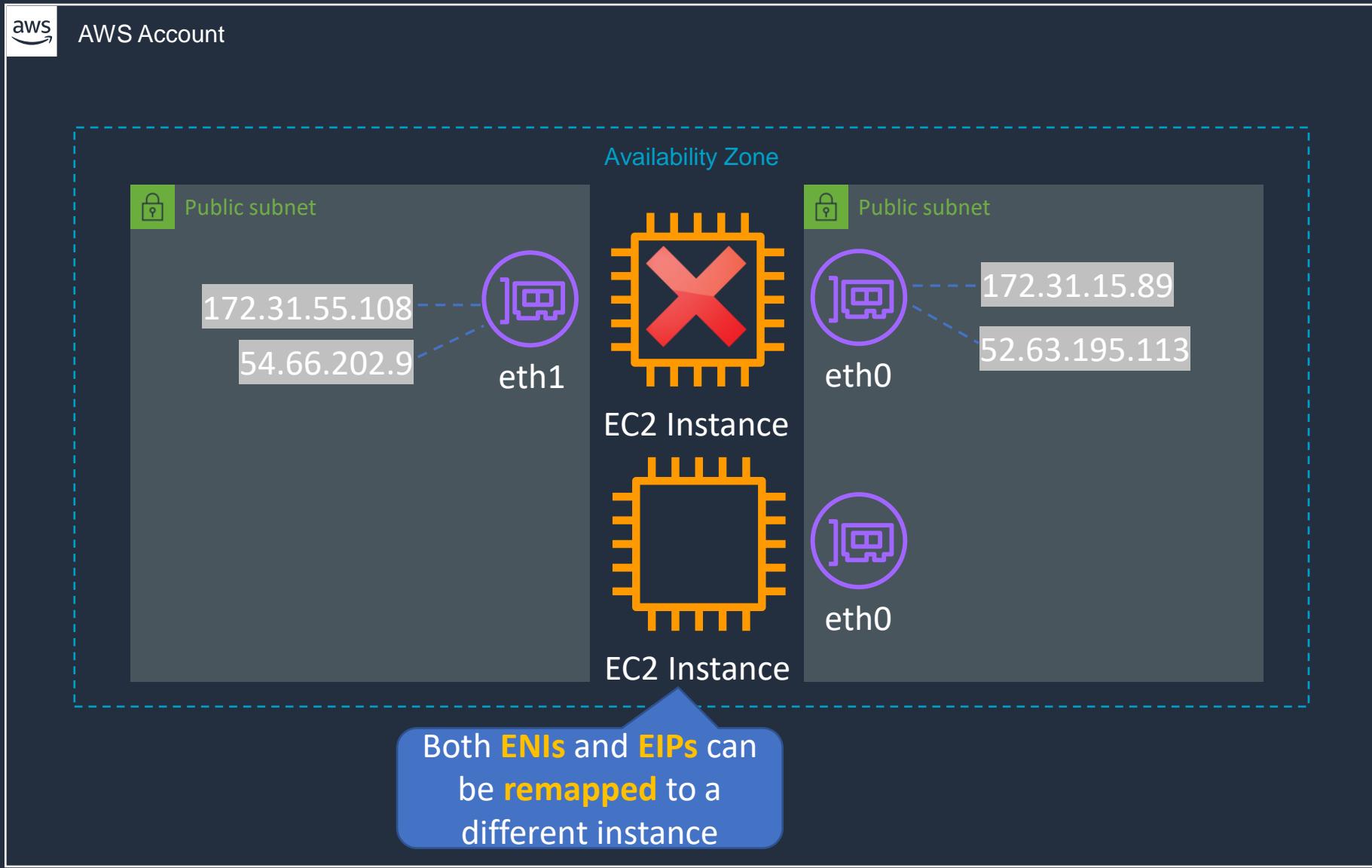
Public, Private and Elastic IP Addresses



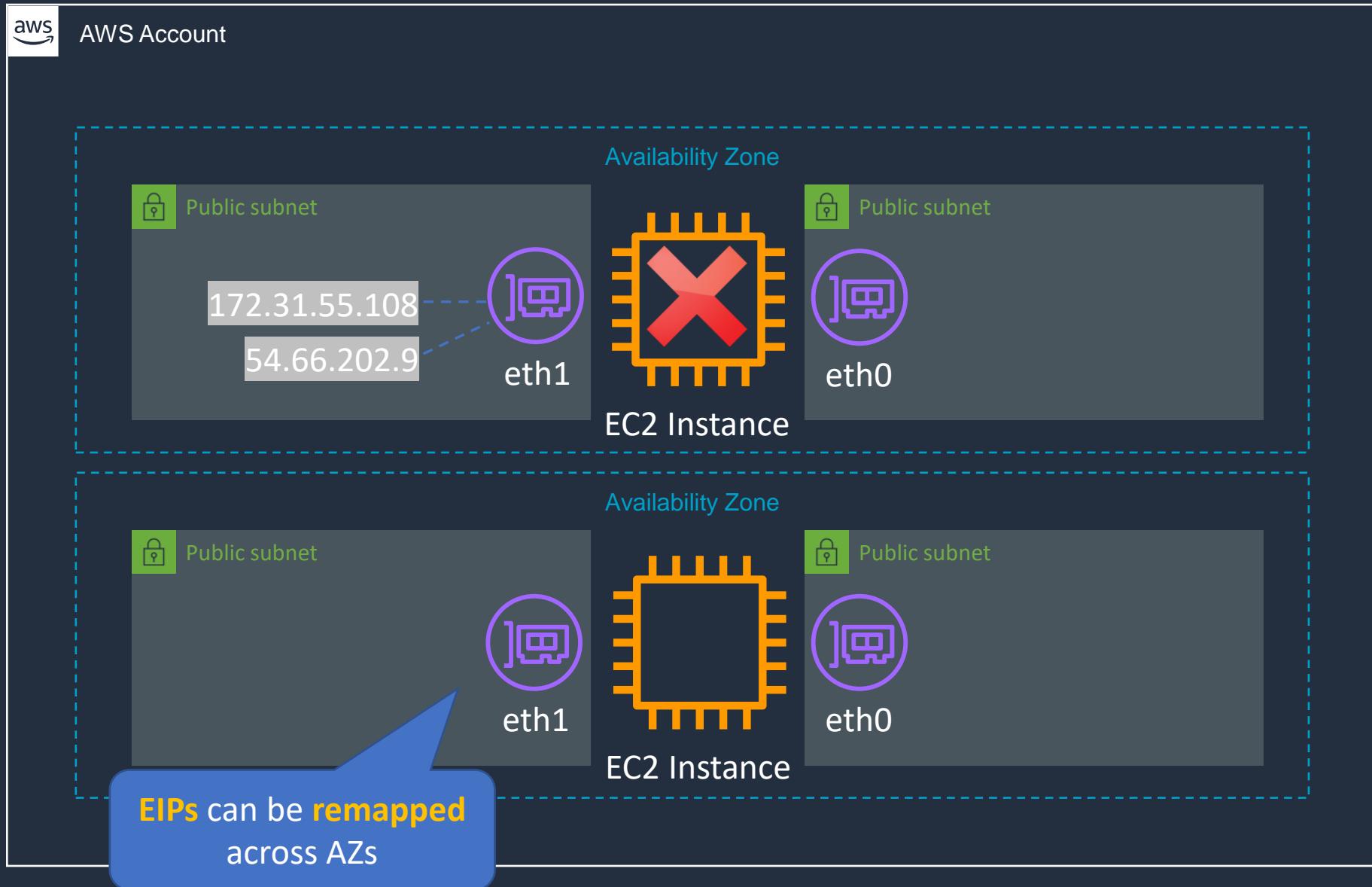
→ Public, Private and Elastic IP Addresses



→ Public, Private and Elastic IP Addresses



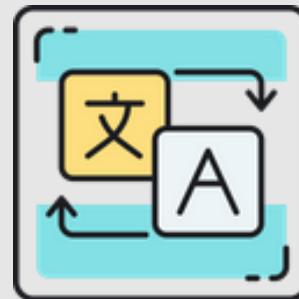
→ Public, Private and Elastic IP Addresses



→ Public, Private and Elastic IP addresses

Name	Description
Public IP address	<p>Lost when the instance is stopped</p> <p>Used in Public Subnets</p> <p>No charge</p> <p>Associated with a private IP address on the instance</p> <p>Cannot be moved between instances</p>
Private IP address	<p>Retained when the instance is stopped</p> <p>Used in Public and Private Subnets</p>
Elastic IP address	<p>Static Public IP address</p> <p>You are charged if not used</p> <p>Associated with a private IP address on the instance</p> <p>Can be moved between instances and Elastic Network Adapters</p>

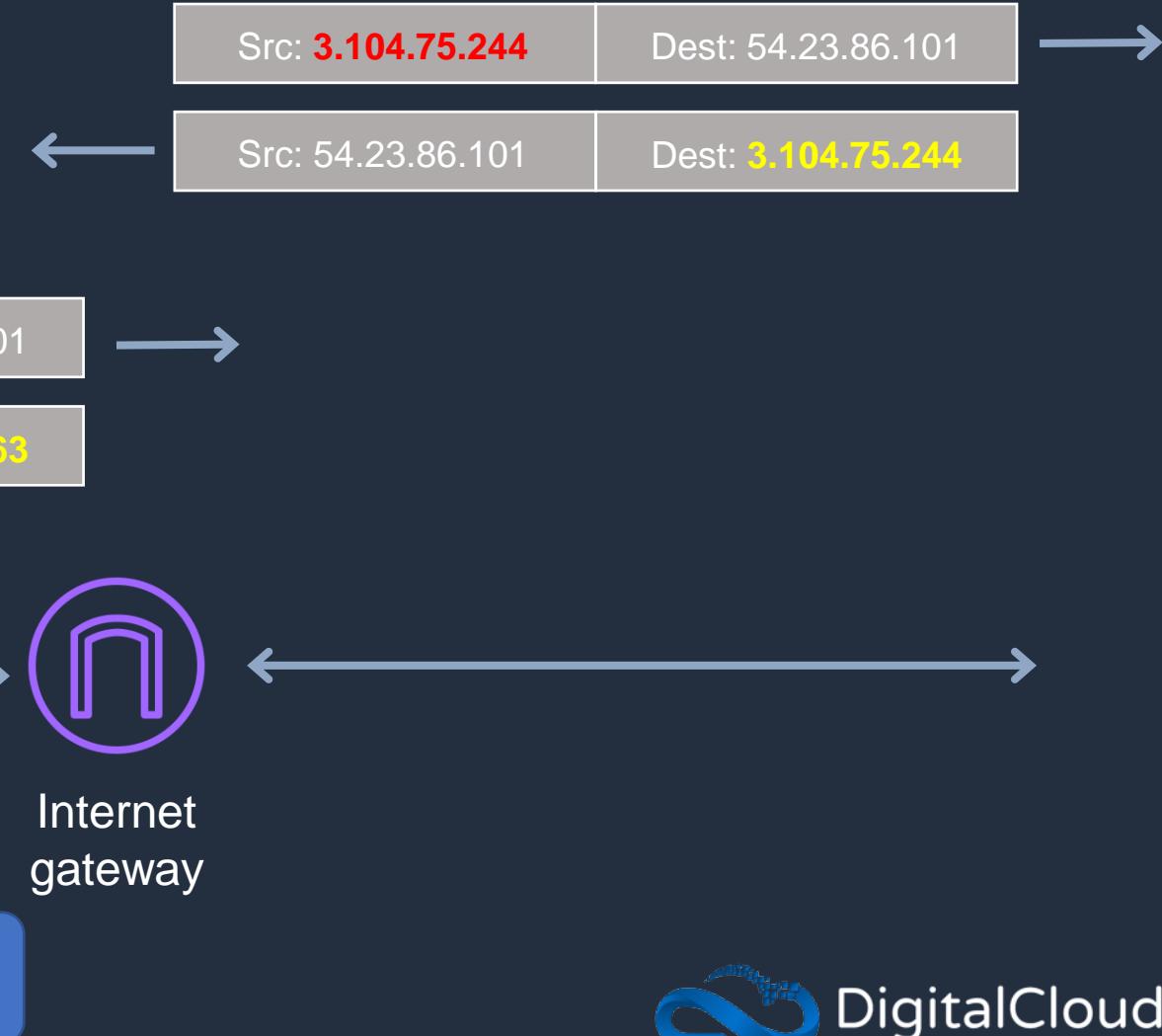
NAT for Public Addresses



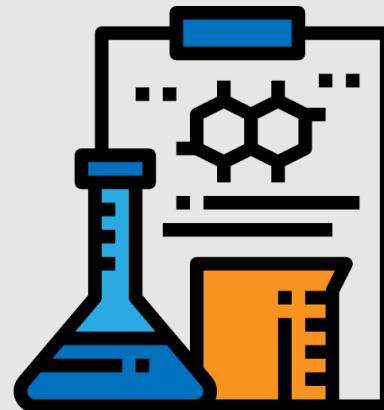


NAT for Public Addresses

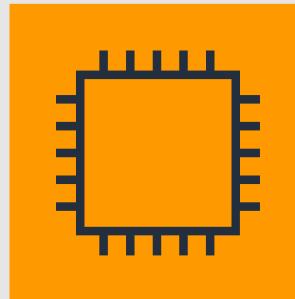
```
[ec2-user@ip-172-31-32-63 ~]$ ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 06:06:db:ad:56:28 brd ff:ff:ff:ff:ff:ff
    inet 172.31.32.63/20 brd 172.31.47.255 scope global dynamic eth0
        valid_lft 3330sec preferred_lft 3330sec
    inet6 fe80::406:dbff:fead:5628/64 scope link
        valid_lft forever preferred_lft forever
```

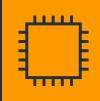


Working with ENIs and IP Addresses

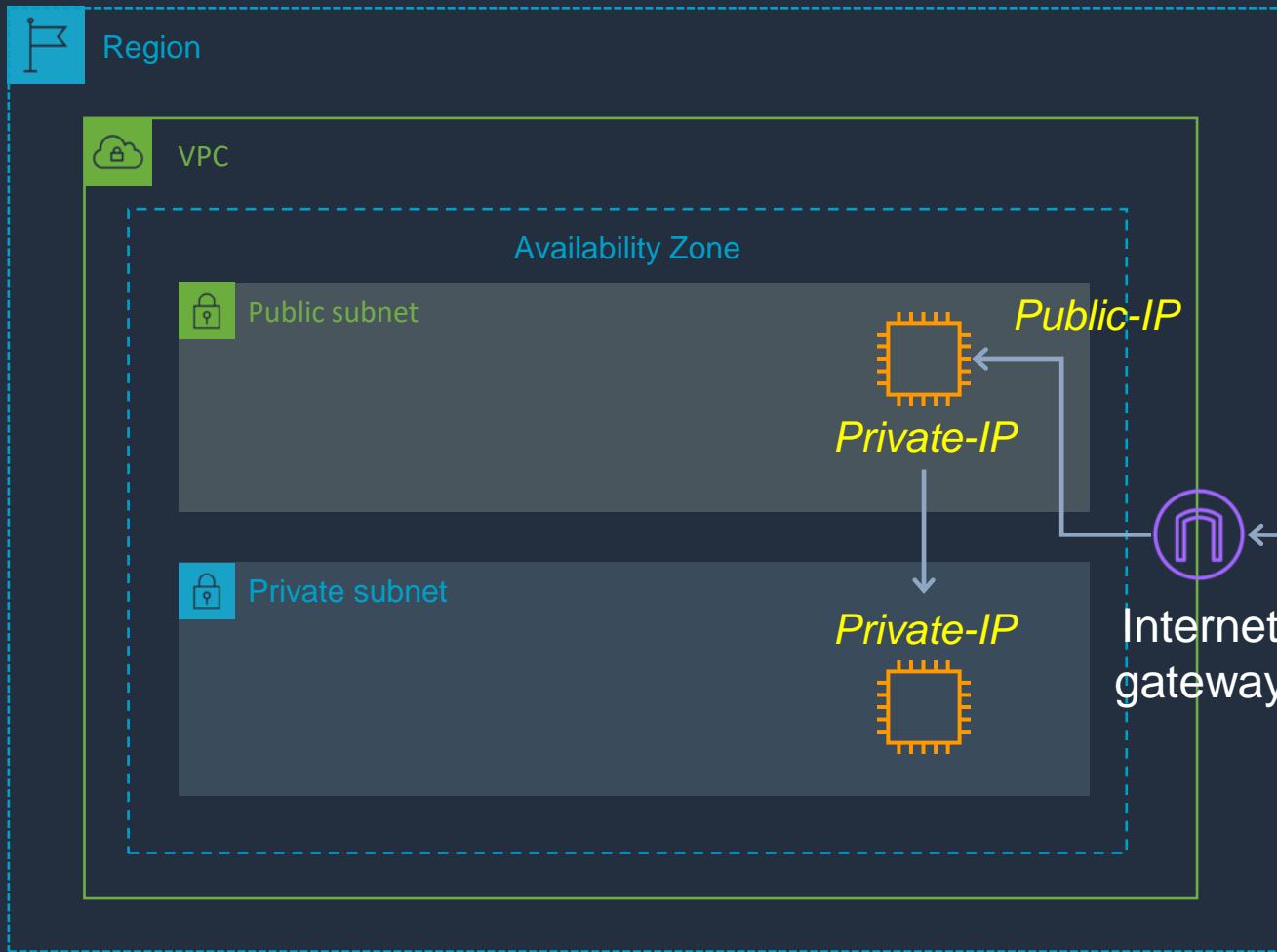


Private Subnets and Bastion Hosts





Private Subnets and Bastion Hosts



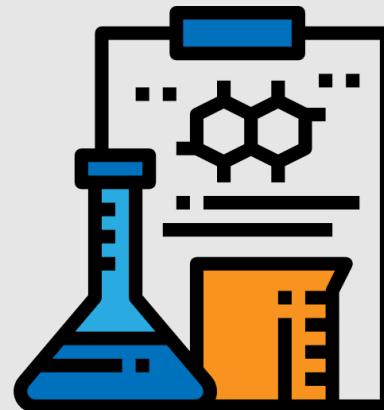
Public Subnet Route Table

Destination	Target
172.31.0.0/16	Local
0.0.0.0/0	igw-id

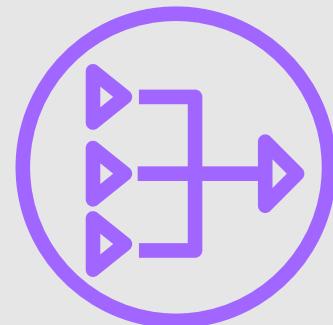
Private Subnet Route Table

Destination	Target
172.31.0.0/16	Local

Private Subnets and Bastion Hosts

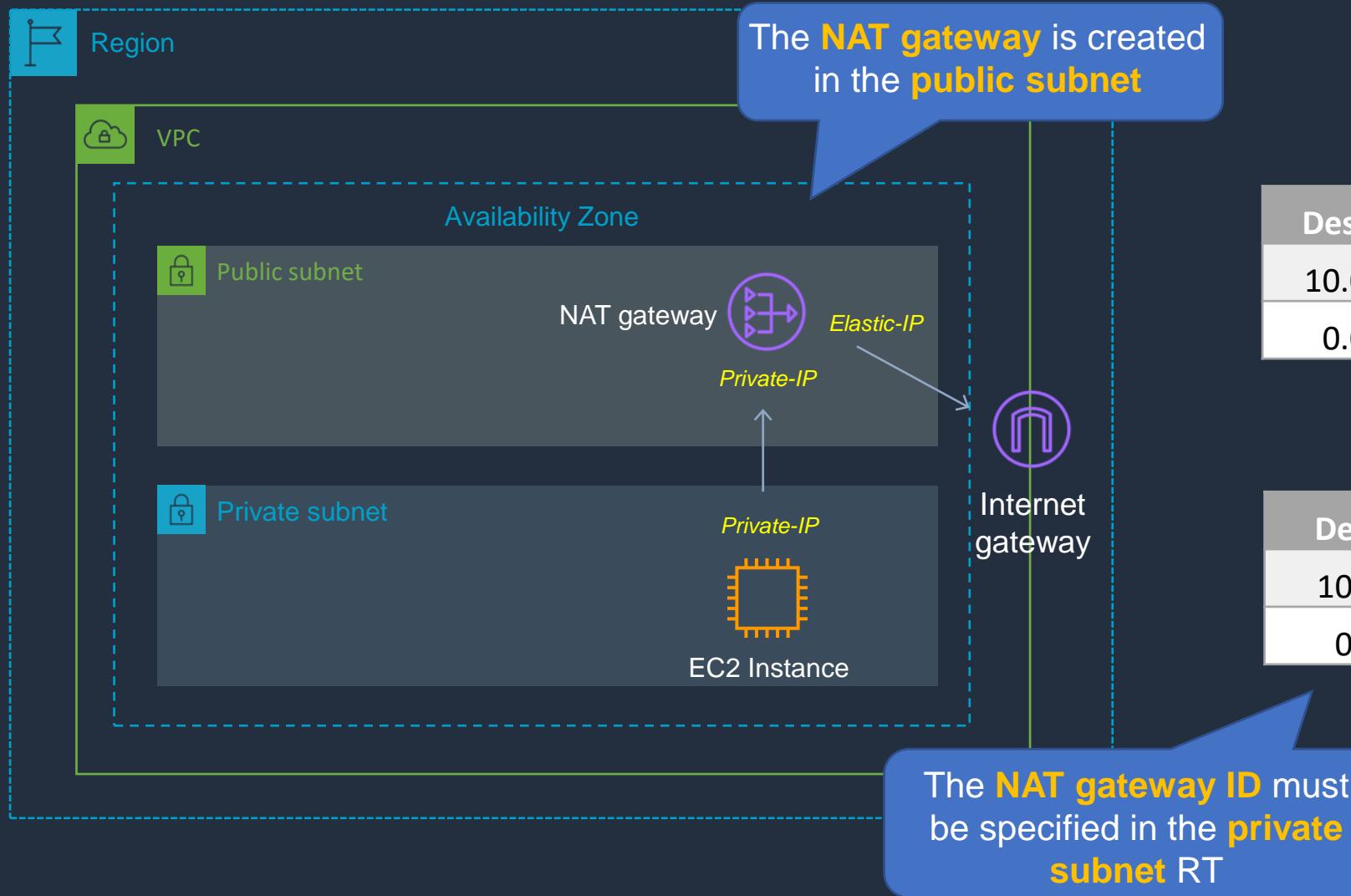


NAT Gateways and NAT Instances Overview



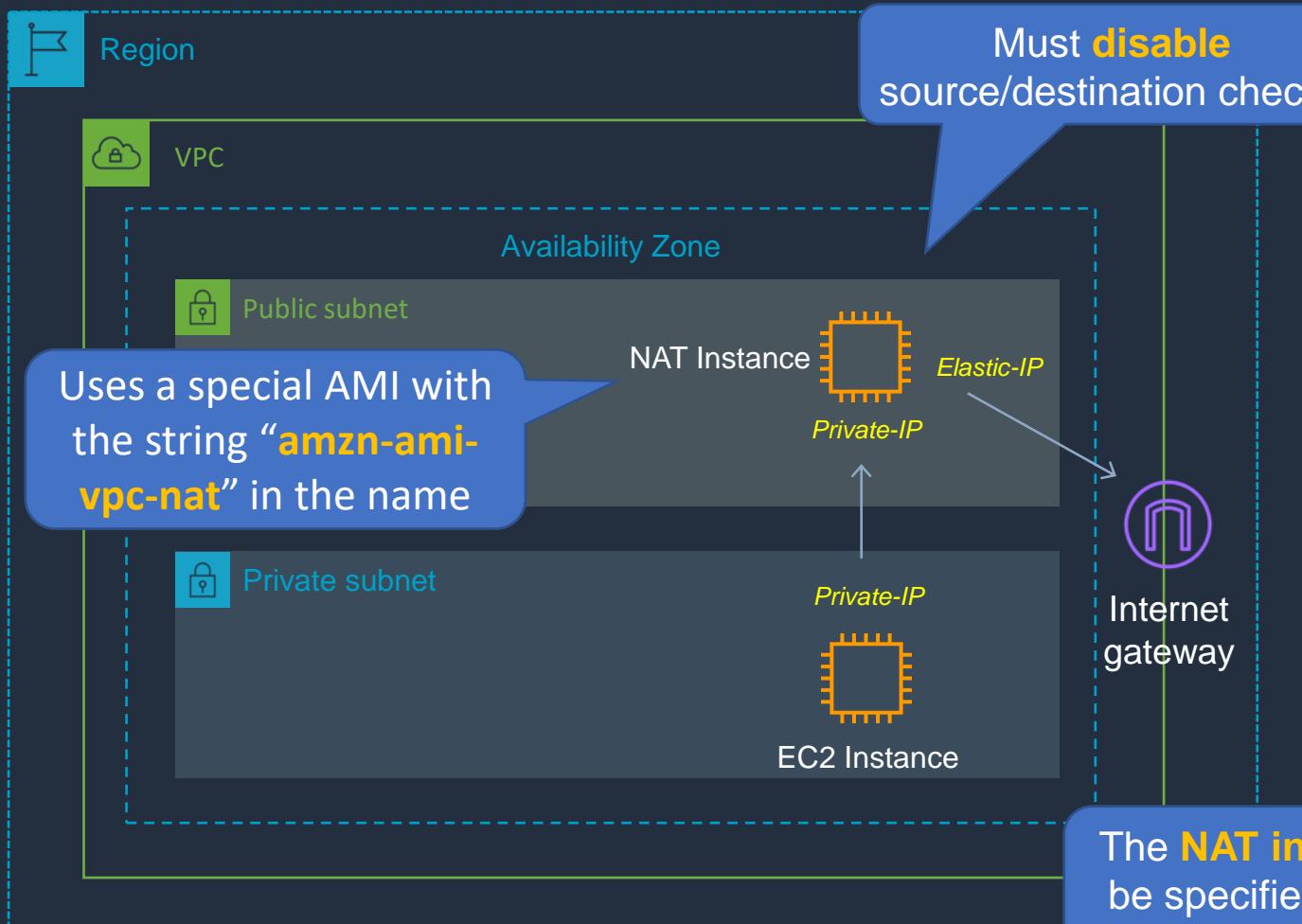


NAT Gateways





NAT Instances



Main Route Table

Destination	Target
10.0.0.0/16	Local
0.0.0.0/0	igw-id

Private Route Table

Destination	Target
10.0.0.0/16	Local
0.0.0.0/0	nat-instance-id



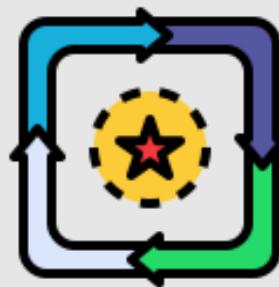
NAT Instance vs NAT Gateway

NAT Instance	NAT Gateway
Managed by you (e.g. software updates)	Managed by AWS
Scale up (instance type) manually and use enhanced networking	Elastic scalability up to 45 Gbps
No high availability – scripted/auto-scaled HA possible using multiple NATs in multiple subnets	Provides automatic high availability within an AZ and can be placed in multiple AZs
Need to assign Security Group	No Security Groups
Can use as a bastion host	Cannot access through SSH
Use an Elastic IP address or a public IP address with a NAT instance	Choose the Elastic IP address to associate with a NAT gateway at creation
Can implement port forwarding through manual customisation	Does not support port forwarding

Private Subnet with NAT Gateway

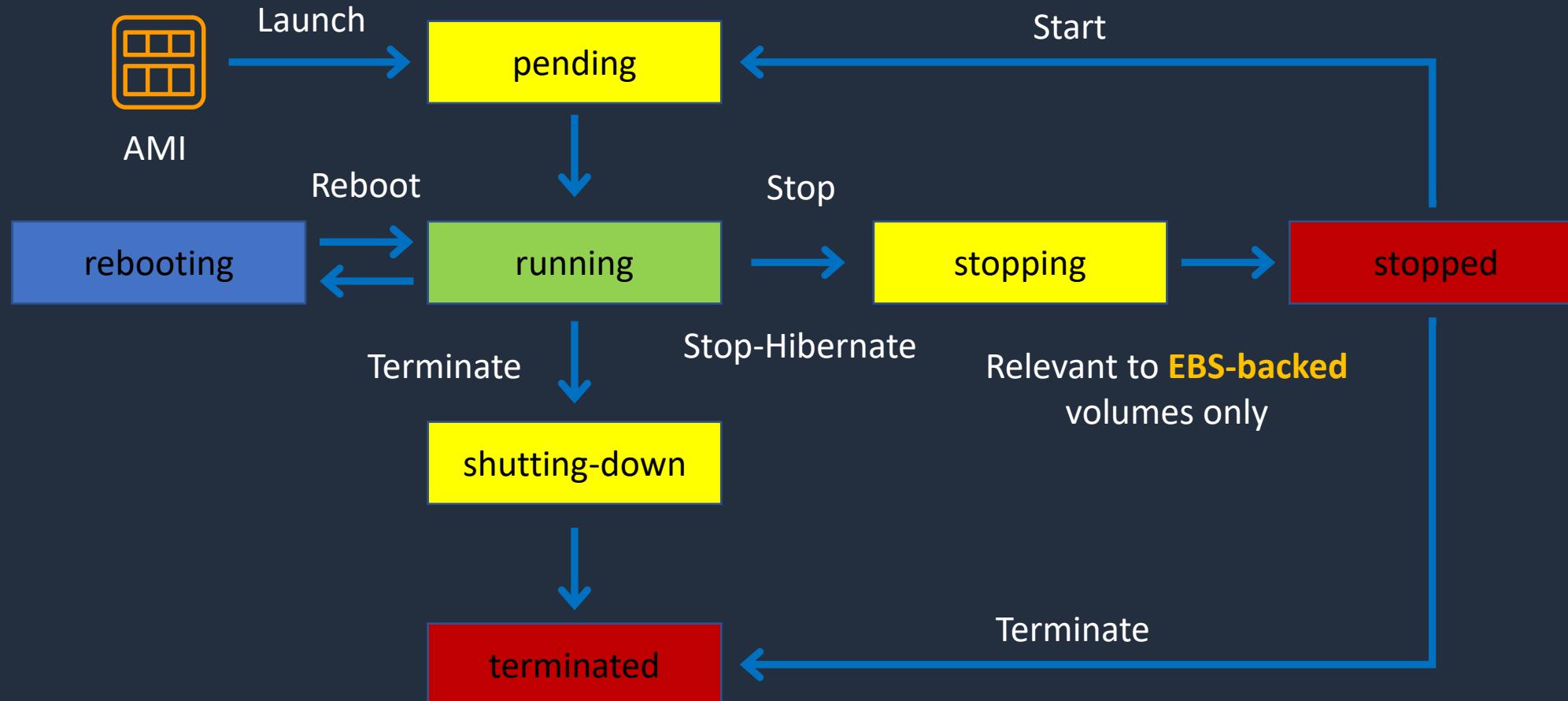


EC2 Instance Lifecycle





EC2 Instance Lifecycle





EC2 Instance Lifecycle

Stopping EC2 instances

- EBS backed instances only
- No charge for stopped instances
- EBS volumes remain attached (chargeable)
- Data in RAM is lost
- Instance is migrated to a different host
- Private IPv4 addresses and IPv6 addresses retained; public IPv4 addresses released
- Associated Elastic IPs retained



EC2 Instance Lifecycle

Hibernating EC2 instances

- Applies to supported AMIs
- Contents of RAM saved to EBS volume
- Must be enabled for hibernation when launched
- Specific prerequisites apply
- When started (after hibernation):
 - The EBS root volume is restored to its previous state
 - The RAM contents are reloaded
 - The processes that were previously running on the instance are resumed
 - Previously attached data volumes are reattached and the instance retains its instance ID



EC2 Instance Lifecycle

Rebooting EC2 instances

- Equivalent to an OS reboot
- DNS name and all IPv4 and IPv6 addresses retained
- Does not affect billing

Retiring EC2 instances

- Instances may be retired if AWS detects irreparable failure of the underlying hardware that hosts the instance
- When an instance reaches its scheduled retirement date, it is stopped or terminated by AWS



EC2 Instance Lifecycle

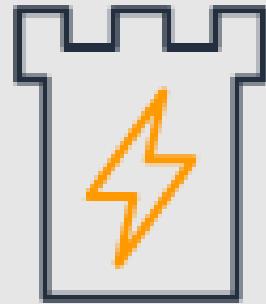
Terminating EC2 instances

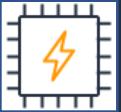
- Means deleting the EC2 instance
- Cannot recover a terminated instance
- By default, root EBS volumes are deleted

Recovering EC2 instances

- CloudWatch can be used to monitor system status checks and recover instance if needed
- Applies if the instance becomes impaired due to underlying hardware / platform issues
- Recovered instance is identical to original instance

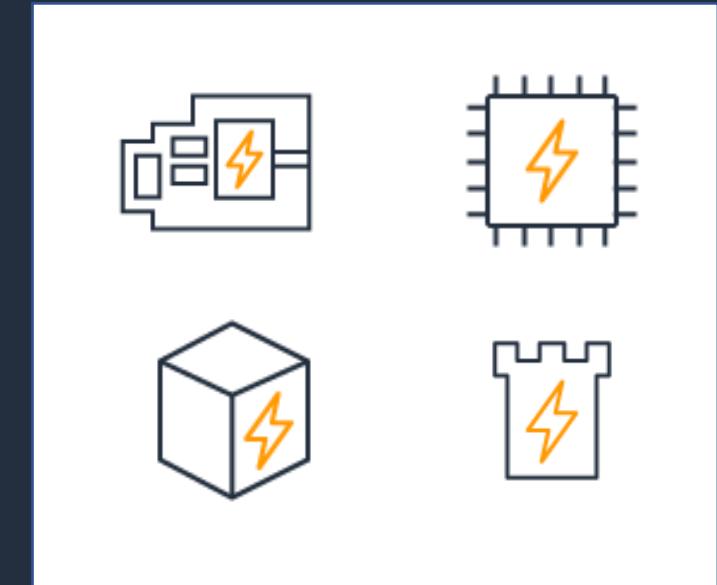
Nitro Instances and Nitro Enclaves

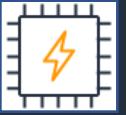




AWS Nitro System

- Nitro is the underlying platform for the next generation of EC2 instances
- Support for many virtualized and bare metal instance types
- Breaks functions into specialized hardware with a Nitro Hypervisor
- Specialized hardware includes:
 - Nitro cards for VPC
 - Nitro cards for EBS
 - Nitro for Instance Storage
 - Nitro card controller
 - Nitro security chip
 - Nitro hypervisor
 - Nitro Enclaves





AWS Nitro System

- Improves performance, security and innovation:
 - Performance close to bare metal for virtualized instances
 - Elastic Network Adapter and Elastic Fabric Adapter
 - More bare metal instance types
 - Higher network performance (e.g. 100 Gbps)
 - High Performance Computing (HPC) optimizations
 - Dense storage instances (e.g. 60 TB)



AWS Nitro Enclaves

- Isolated compute environments
- Runs on isolated and hardened virtual machines
- No persistent storage, interactive access, or external networking
- Uses cryptographic attestation to ensure only authorized code is running
- Integrates with AWS Key Management Service (KMS)
- Protect and securely process highly sensitive data:
 - Personally identifiable information (PII)
 - Healthcare data
 - Financial data
 - Intellectual Property data

Amazon EC2 Pricing Options





Amazon EC2 Pricing Options

On-Demand

Standard rate - no discount; no commitments; dev/test, short-term, or unpredictable workloads

Spot Instances

Get discounts of up to 90% for unused capacity. Can be terminated at any time

Dedicated Hosts

Physical server dedicated for your use; Socket/core visibility, host affinity; pay per host; workloads with server-bound software licenses

Reserved

1 or 3-year commitment; up to 75% discount; steady-state, predictable workloads and reserved capacity

Dedicated Instances

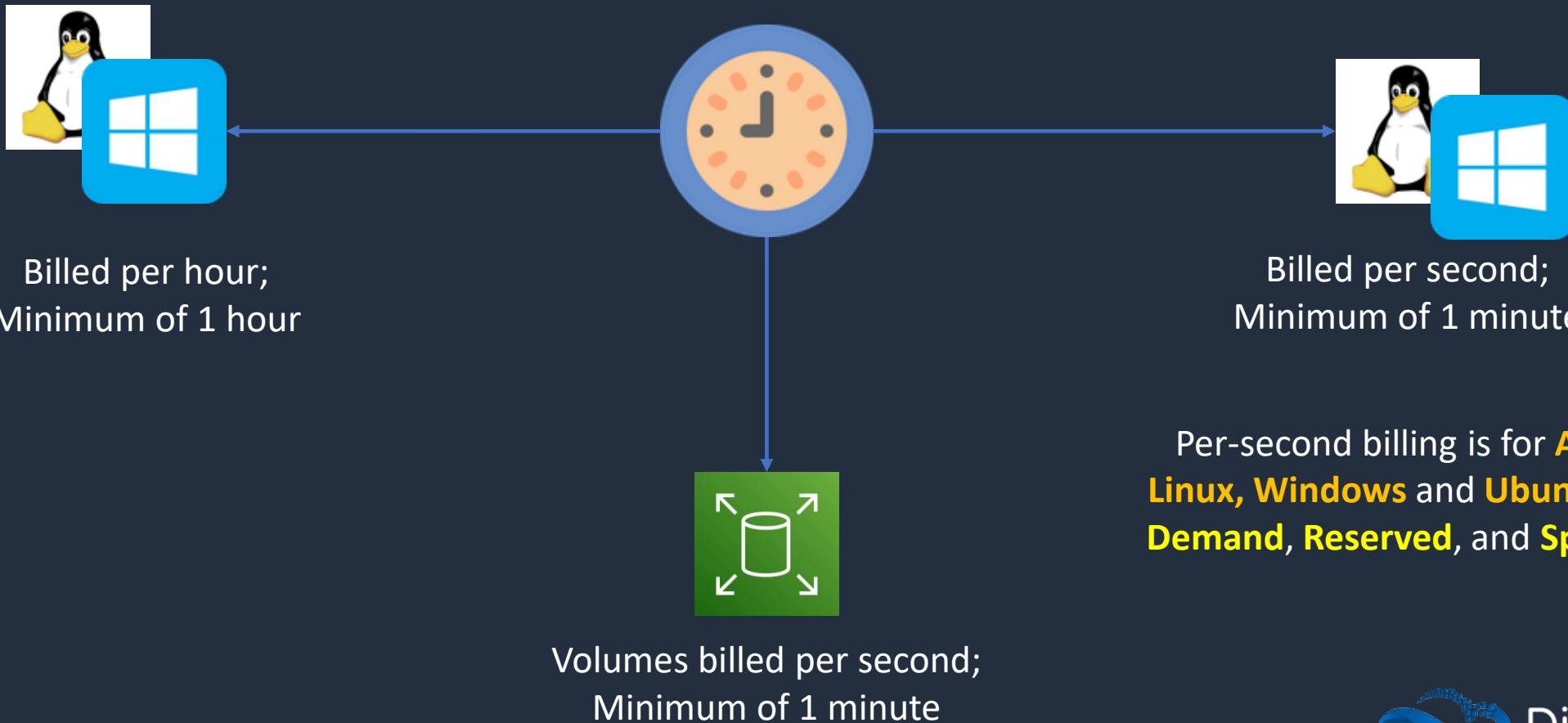
Physical isolation at the host hardware level from instances belonging to other customers; pay per instance

Savings Plans

Commitment to a consistent amount of usage (EC2 + Fargate + Lambda); Pay by \$/hour; 1 or 3-year commitment

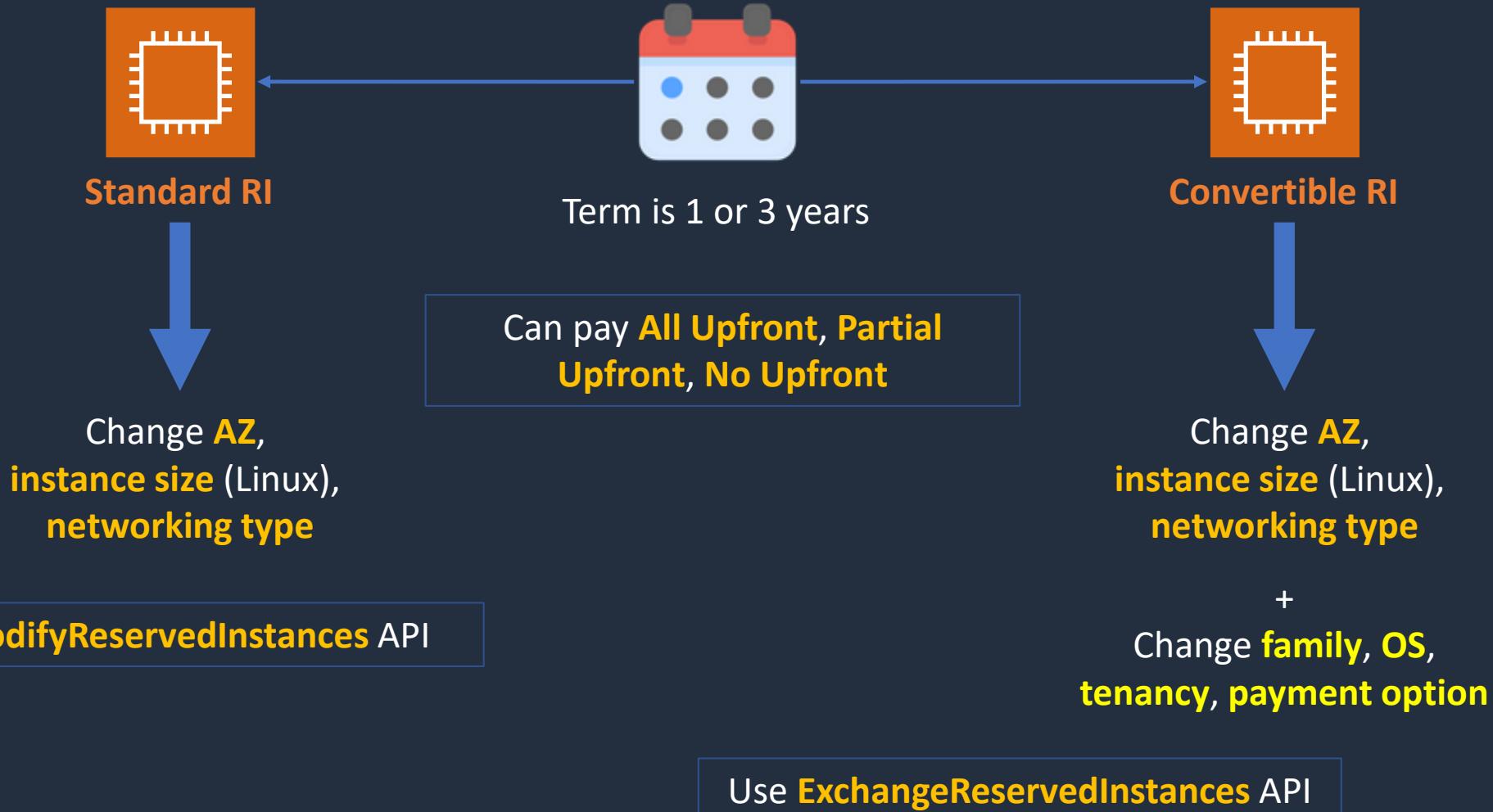
\$ Amazon EC2 Billing

Commercial Linux distros such as **Red Hat EL** and **SUSE ES** use **hourly** pricing



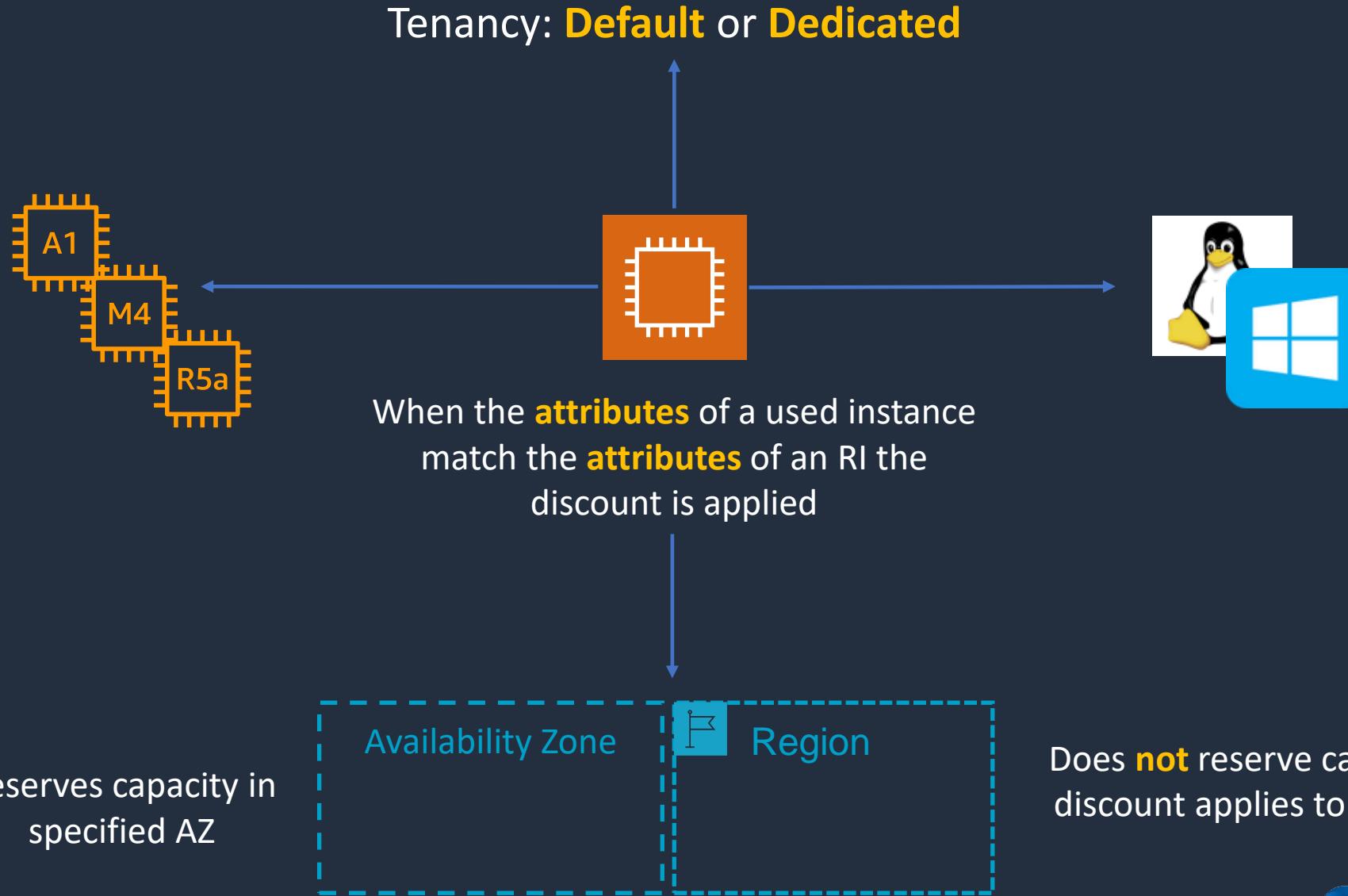


Amazon EC2 Reserved Instances (RIs)





Amazon EC2 Reserved Instances (RIs)

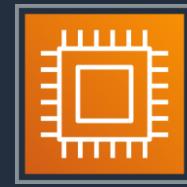




Amazon EC2 On-Demand Capacity Reservations

- Reserve compute capacity for your Amazon EC2 instances in a specific Availability Zone
- Any duration can be specified
- Mitigates against the risk of being unable to get On-Demand capacity
- Does not require any term commitments and can be cancelled at any time
- When you create a Capacity Reservation, you specify:
 - The **Availability Zone** in which to reserve the capacity
 - The **number of instances** for which to reserve capacity
 - The **instance attributes**, including the instance type, tenancy, and platform/OS

\$ AWS Savings Plans



Compute Savings Plan



1 or 3-year; hourly commitment to usage of **Fargate**, **Lambda**, and **EC2**; Any Region, family, size, tenancy, and OS



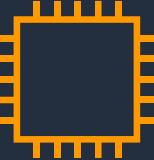
EC2 Savings Plan



1 or 3-year; hourly commitment to usage of **EC2** within a **selected Region** and **Instance Family**; Any size, tenancy and OS



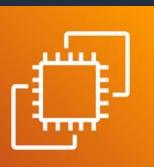
Amazon EC2 Spot Instances



Spot Instance: One or more EC2 instances



Spot Fleet: launches and maintains the number of Spot / On-Demand instances to meet specified target capacity



EC2 Fleet: launches and maintains specified number of Spot / On-Demand / Reserved instances in a **single API call**

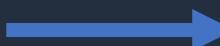


2-minute warning if AWS need to reclaim capacity – available via **instance metadata** and **CloudWatch Events**

Can define separate OD/Spot **capacity targets**, **Spot price**, **instance types**, and **AZs**



Spot Block



Requirement:
Uninterrupted for
1-6 hours

Pricing is **30% - 45%** less
than On-Demand



Solution: **Spot Block**

```
$ aws ec2 request-spot-instances \  
  --block-duration-minutes 360 \  
  --instance-count 5 \  
  --spot-price "0.25" ...
```



Dedicated Instances and Dedicated Hosts

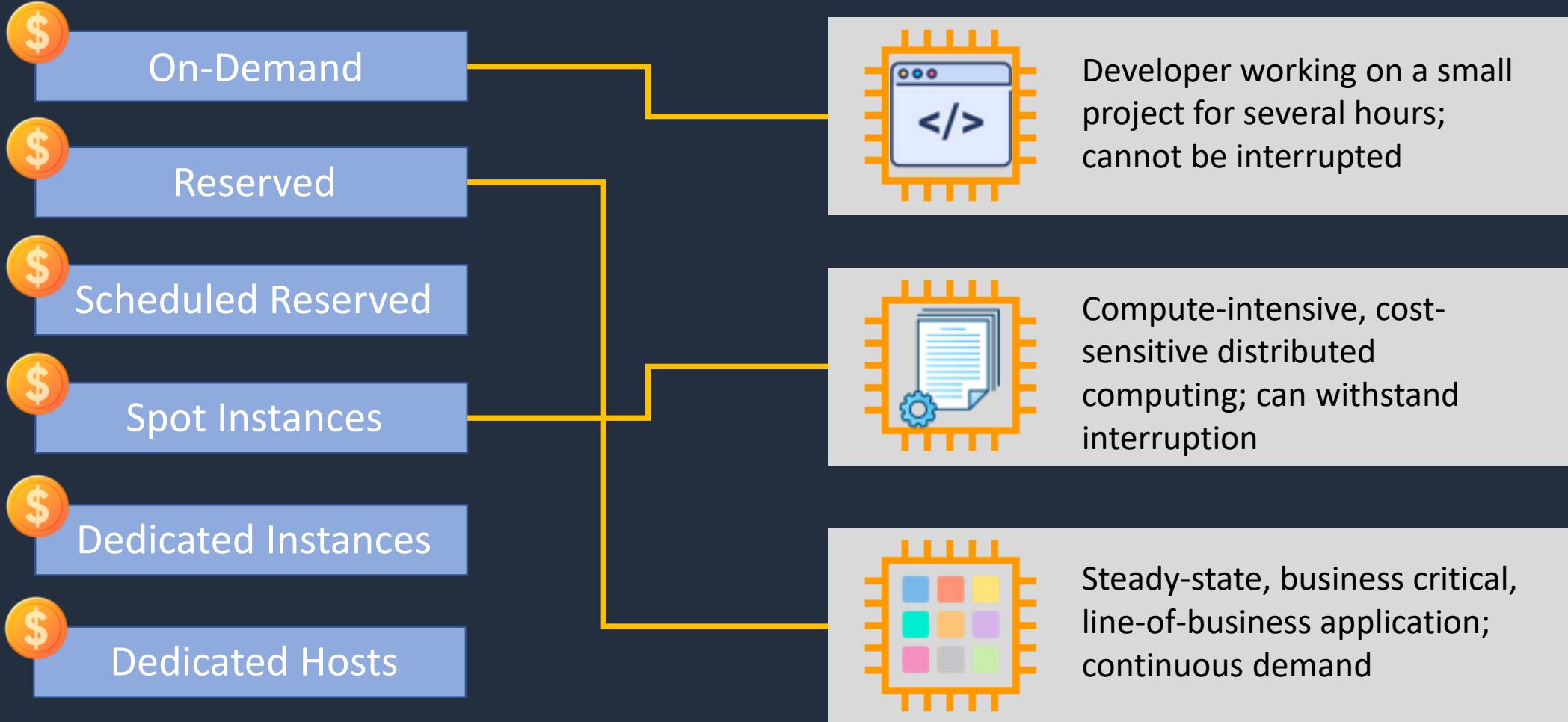
Characteristic	Dedicated Instances	Dedicated Hosts
Enables the use of dedicated physical servers	X	X
Per instance billing (subject to a \$2 per region fee)	X	
Per host billing		X
Visibility of sockets, cores, host ID		X
Affinity between a host and instance		X
Targeted instance placement		X
Automatic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

Amazon EC2 Pricing Use Cases



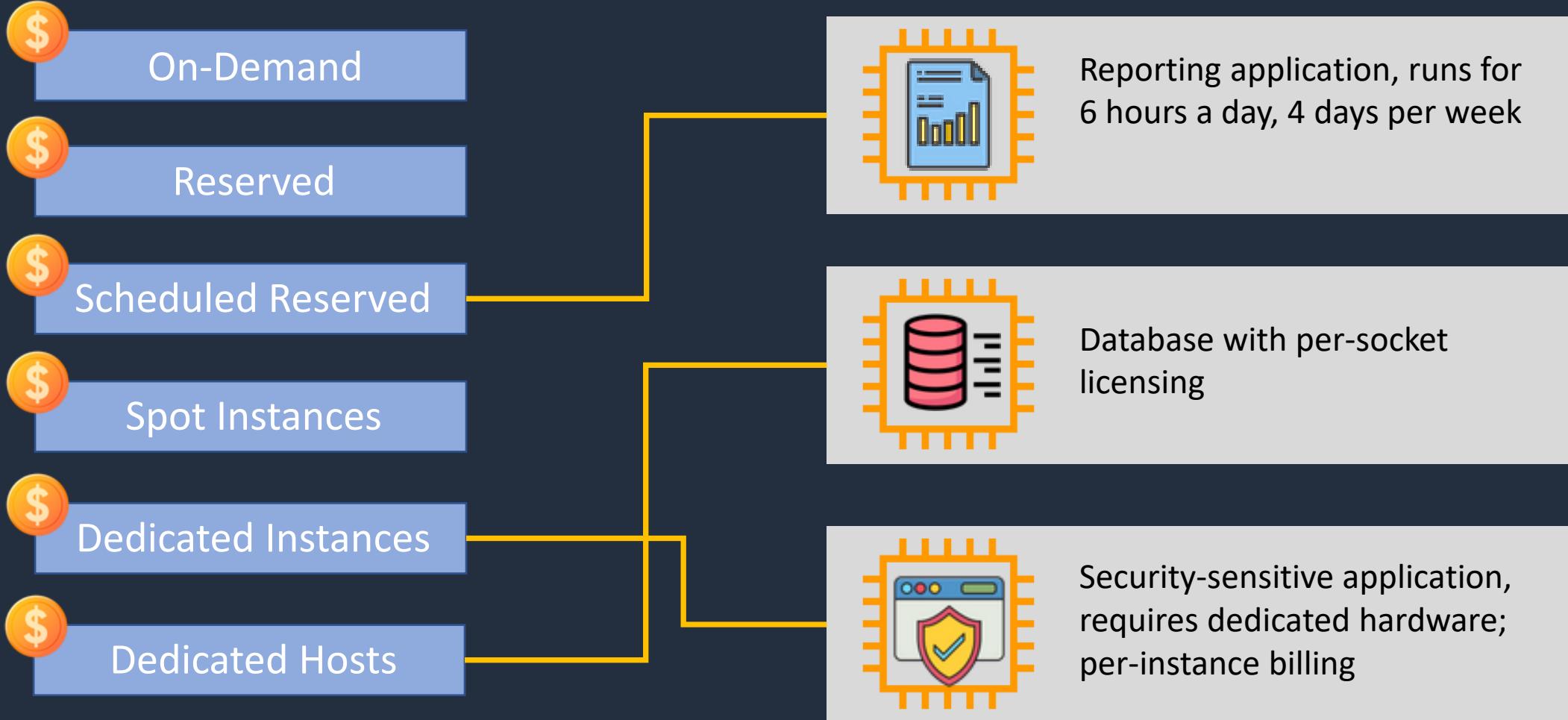


Amazon EC2 Pricing Use Cases

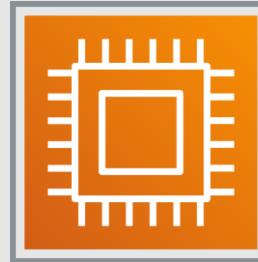


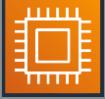


Amazon EC2 Pricing Use Cases



Architecture Patterns – Amazon EC2





Architecture Patterns – Amazon EC2

Requirement

Company needs to run a short batch script to configure Amazon EC2 Linux instances after they are launched

A tightly coupled High Performance Computing (HPC) workload requires low-latency between nodes and optimum network performance

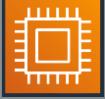
LoB application receives weekly bursts of traffic and must scale for short periods – need the most cost-effective solution

Solution

Add the bash script to the user data of the EC2 instances

Launch EC2 instances in a single AZ in a cluster placement group and use an Elastic Fabric Adapter (EFA)

Use reserved instances for minimum required workload and then use Spot instances for the bursts in traffic



Architecture Patterns - Amazon EC2

Requirement

A single instance application uses a static public IP address. In the event of failure, the address must be remapped to a failover instance

A fleet of Amazon EC2 instances run in private subnets across multiple AZs. Company needs a redundant path to the internet

A team of engineers must administer EC2 instances in private subnets from remote locations using SSH

Solution

Attach an Elastic IP address to the EC2 instance. Remap the EIP in the event of failure

Deploy NAT Gateways into multiple AZs and update route tables

Deploy a bastion host in a public subnet and instruct the engineers to use the bastion host to “jump” to the instances in private subnets



Architecture Patterns - Amazon EC2

Requirement

An application uses several EC2 instances. Architect must eliminate the risk of correlated hardware failures

Solution

Launch the instances in a spread placement group across distinct underlying hardware

Application requires enhanced networking capabilities

Choose an instance type that supports enhanced networking and ensure the ENA module is installed and ENA support is enabled

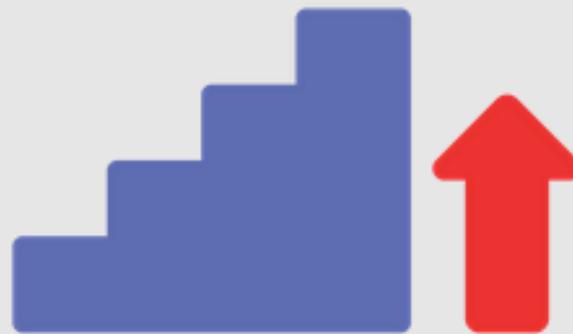
Instance needs close to bare metal performance, EFA, and high performance networking

Use an AWS Nitro instance type

SECTION 4

Elastic Load Balancing, and Auto Scaling

Scaling Up vs Scaling Out





Stateful vs Stateless Applications

Stateless

No “state” is recorded about the user's session



Person checks a weather website

Stateful

Amazon stores information about **activity**

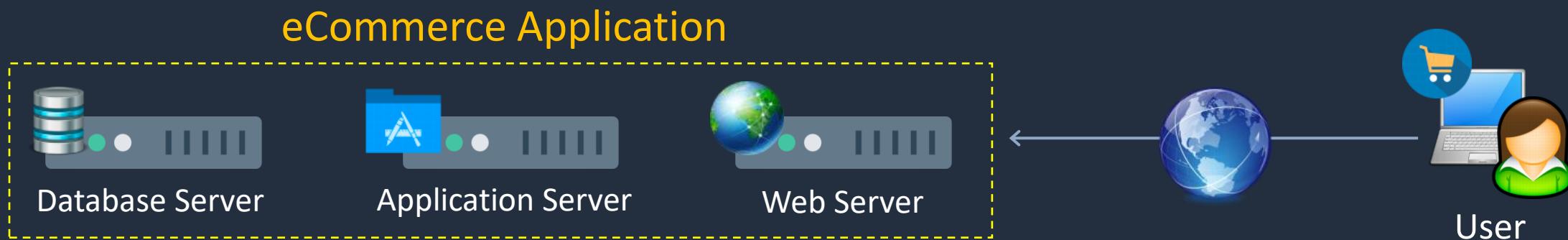


Person browses / purchases on Amazon



Stateful vs Stateless Applications

No data is stored on the web server, it is **stateless**



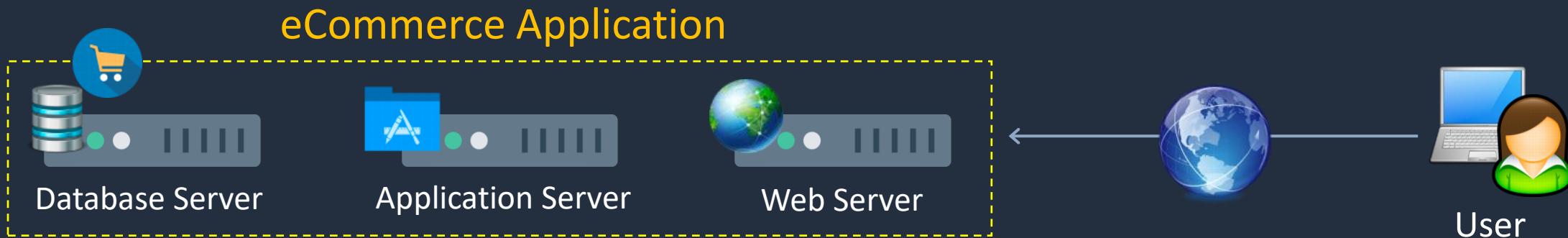
When the user purchases, the application layer processes the order and records the data in the database. This is **stateful**

The cart items are stored in cookies on the computer



Stateful vs Stateless Applications

No data is stored on the web server, it is **stateless**



When the user purchases, the application layer processes the order and records the data in the database. This is **stateful**

The cart items are stored in cookies on the computer



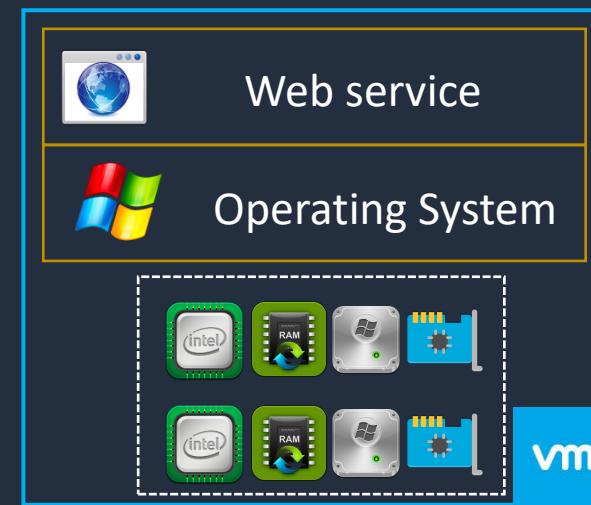
Scalability and Elasticity: Scaling Up





Scalability and Elasticity: Scaling Up

Scaling up means adding resources to the server



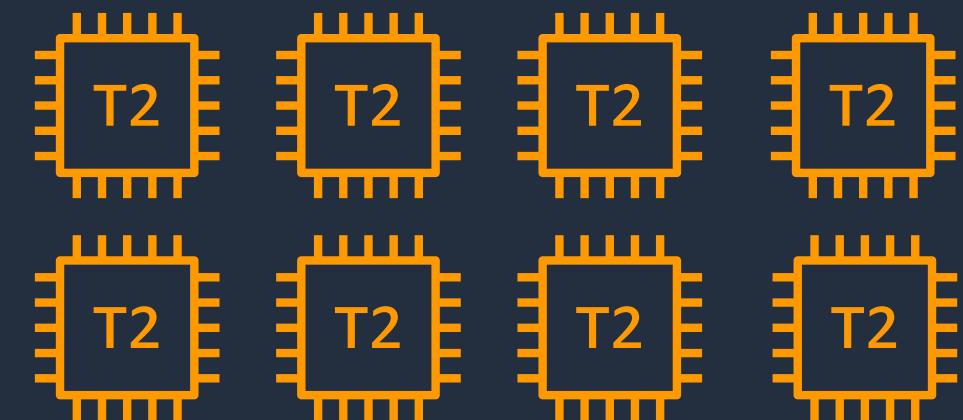
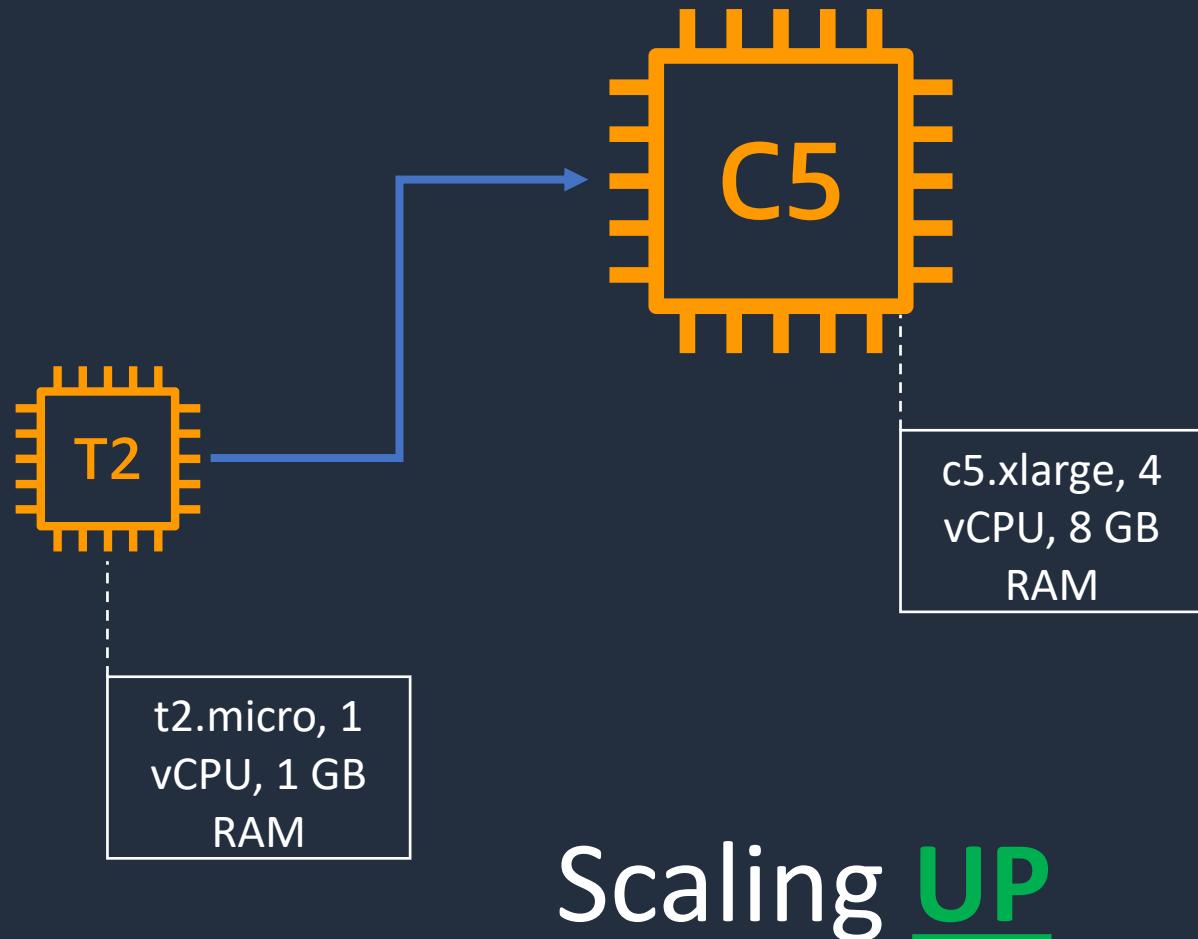


Scalability and Elasticity: Scaling Out





Scaling Up vs Out



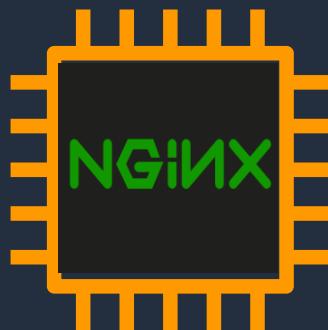


Which scaling model should be used?



Scale UP

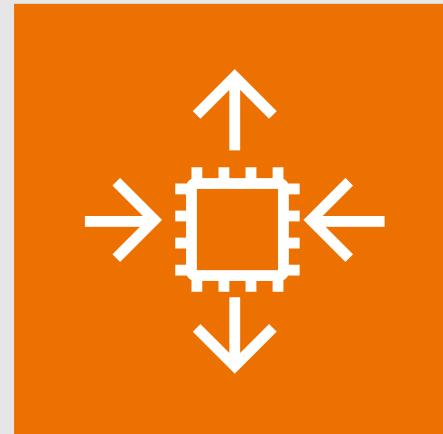
EC2 with MySQL DB



Scale OUT

EC2 with **Static** Website

Amazon EC2 Auto Scaling





Amazon EC2 Auto Scaling

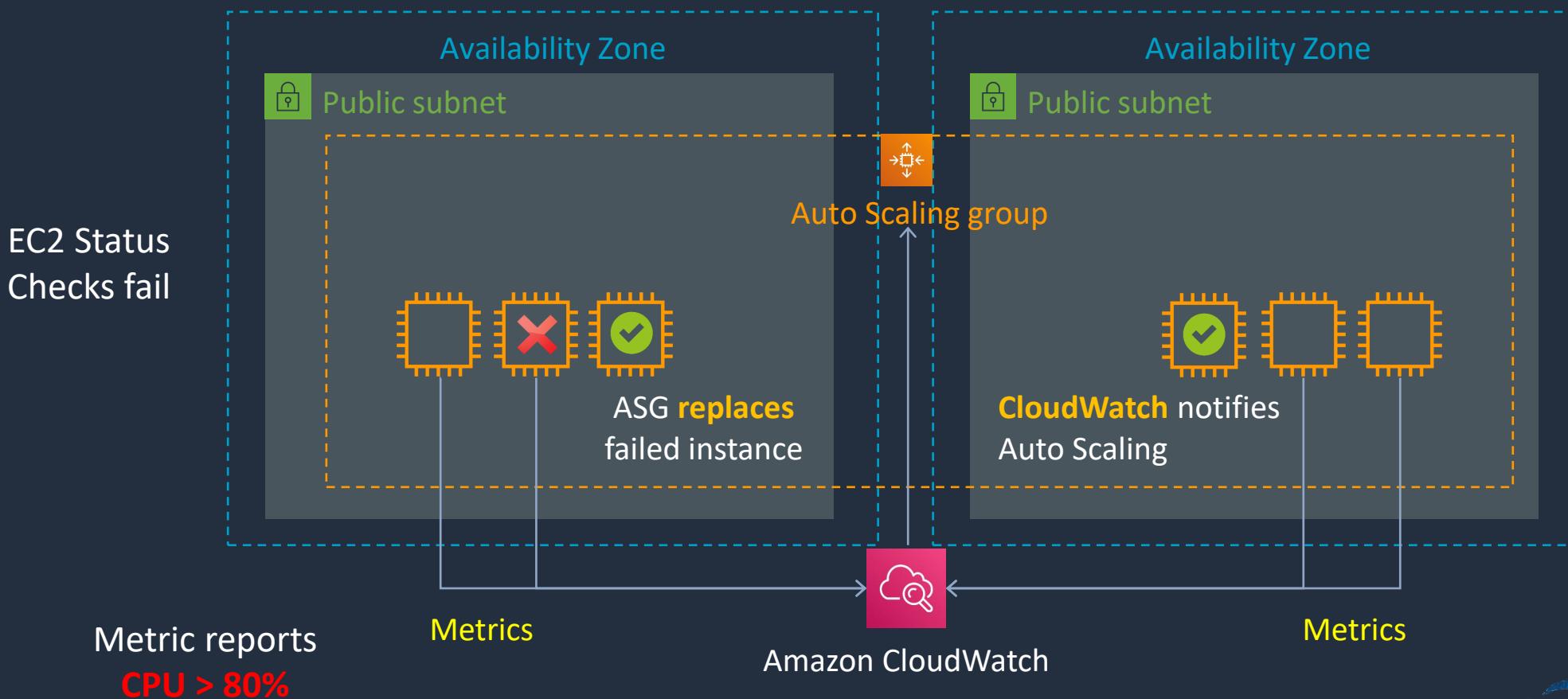
- Automatically **launches** and **terminates** instances
- Maintain **availability** and **scale** capacity
- Works with EC2, ECS, and EKS
- Integrates with many AWS services, including:
 - CloudWatch for monitoring and scaling
 - Elastic Load Balancing for distributing connections
 - EC2 Spot Instances for cost optimization
 - Amazon VPC for deploying instances across AZs



Amazon EC2 Auto Scaling

1. Automatic scaling
2. Maintaining availability

Auto Scaling launches
an extra instance





Amazon EC2 Auto Scaling

- Scaling is horizontal (scales out)
- Provides **elasticity** and **scalability**
- Responds to EC2 status checks and CloudWatch metrics
- Can scale based on demand (performance) or on a schedule
- Scaling policies define how to respond to changes in demand



Configuration of an Auto Scaling Group

A **Launch Template**

specifies the EC2 instance configuration



Launch Template

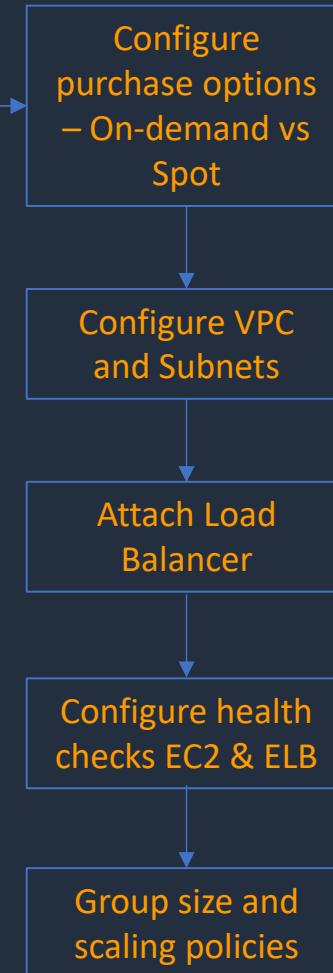
- AMI and instance type
- EBS volumes
- Security groups
- Key pair
- IAM instance profile
- User data
- Shutdown behavior
- Termination protection
- Placement group name
- Capacity reservation
- Tenancy
- Purchasing option (e.g. Spot)



Launch Config

- AMI and instance type
- EBS volumes
- Security groups
- Key pair
- Purchasing option (e.g. Spot)
- IAM instance profile
- User data

Launch Configurations are replaced by launch templates and have fewer features





Amazon EC2 Auto Scaling

- **Health checks**
 - EC2 = EC2 status checks
 - ELB = Uses the ELB health checks **in addition** to EC2 status checks
- **Health check grace period**
 - How long to wait before checking the health status of the instance
 - Auto Scaling does not act on health checks until grace period expires



Amazon EC2 Auto Scaling

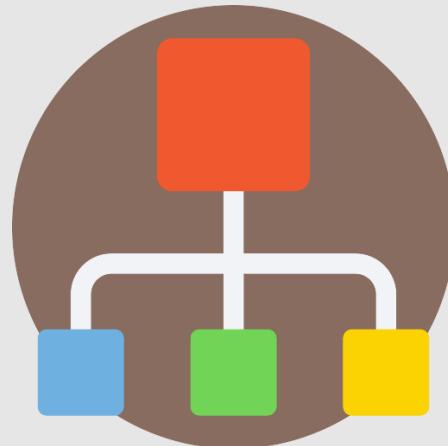
Types of Auto Scaling:

- **Manual** – make changes to ASG size manually
- **Dynamic** – automatically scales based on demand
- **Predictive** – uses Machine Learning to predict
- **Scheduled** – scales based on a schedule

Create an Auto Scaling Group



High Availability and Fault Tolerance





High Availability vs Fault Tolerance

High Availability

- Minimal service interruption
- Designed with no single point of failure (redundancy)
- Uptime measured in %, e.g. 99.99%
- Synchronous or asynchronous replication
- Lower cost compared to FT
- Services that create HA:
 - Elastic Load Balancing
 - EC2 Auto Scaling
 - Amazon Route 53

Fault Tolerance

- No service interruption
- Specialized hardware with instantaneous failover
- No downtime
- Synchronous replication
- Higher cost compared to HA
- Examples of FT:
 - Fault tolerant NICs
 - Disk mirroring (RAID 1)
 - Synchronous DB replication
 - Redundant power



Fault Tolerance

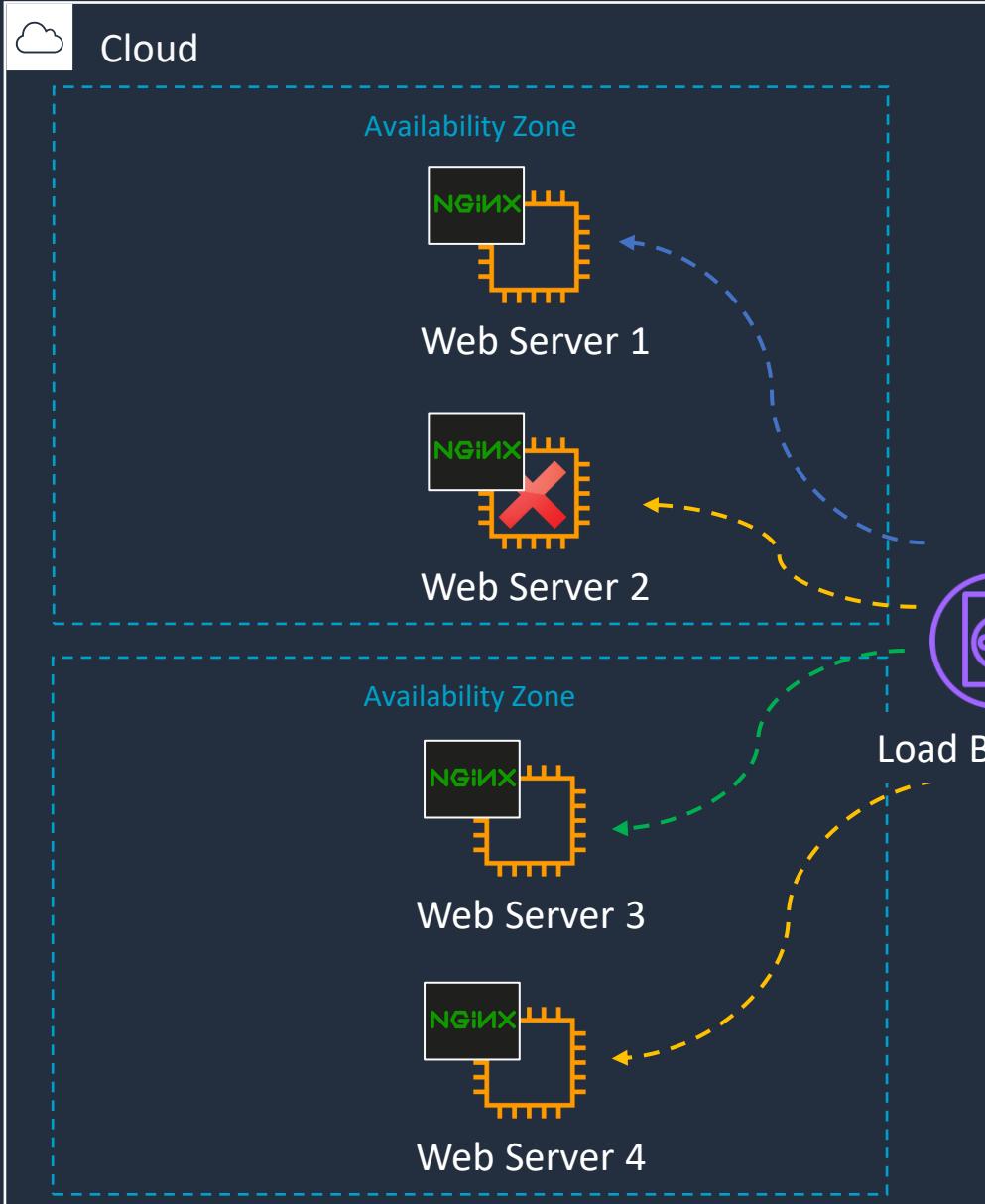
Redundant components
allow the system to
continue to operate



The system may fail if there is
no built-in redundancy



High Availability and Fault Tolerance



Think of an **availability zone** as a separate data center



User 1



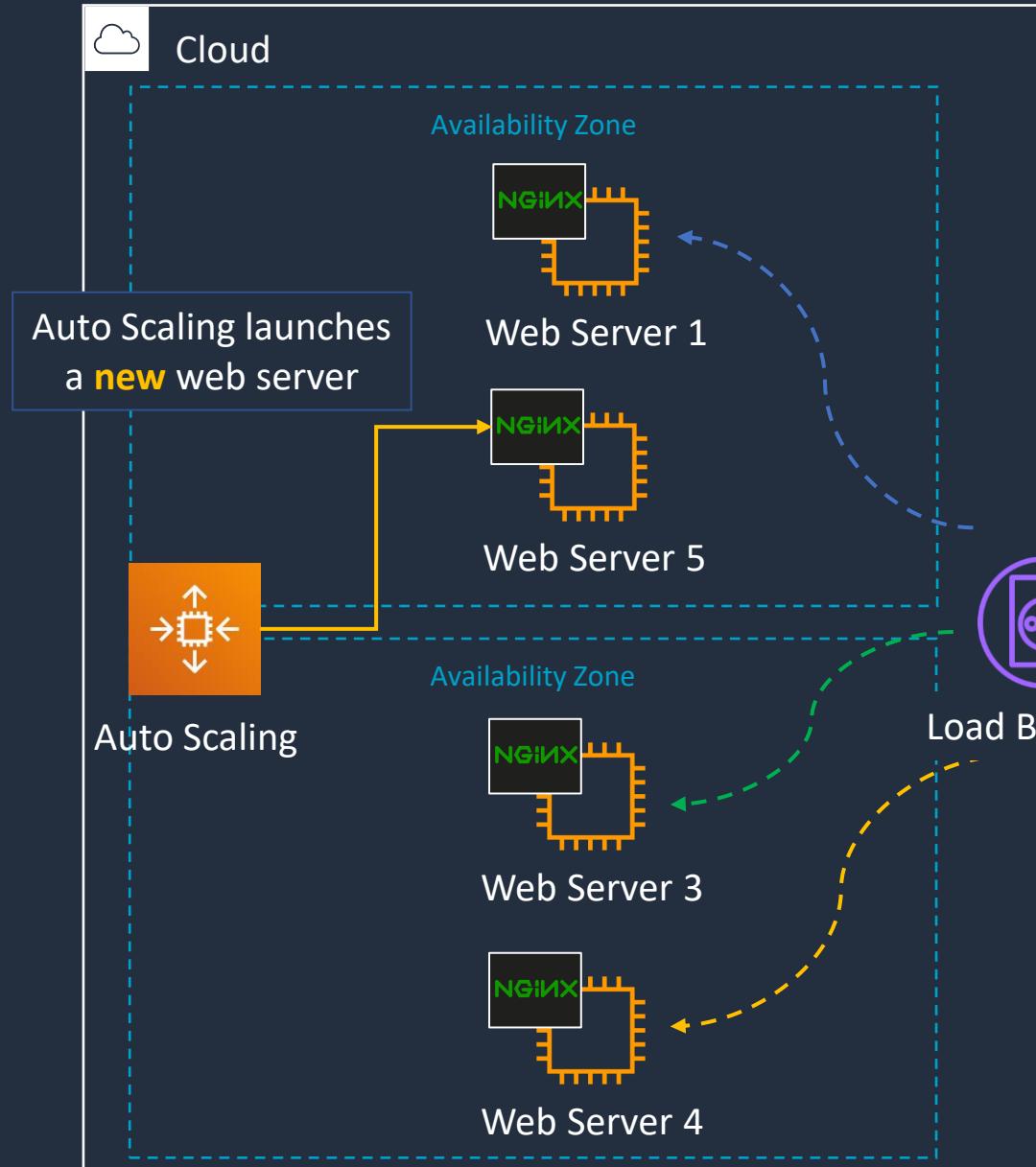
User 2



User 3



High Availability and Fault Tolerance

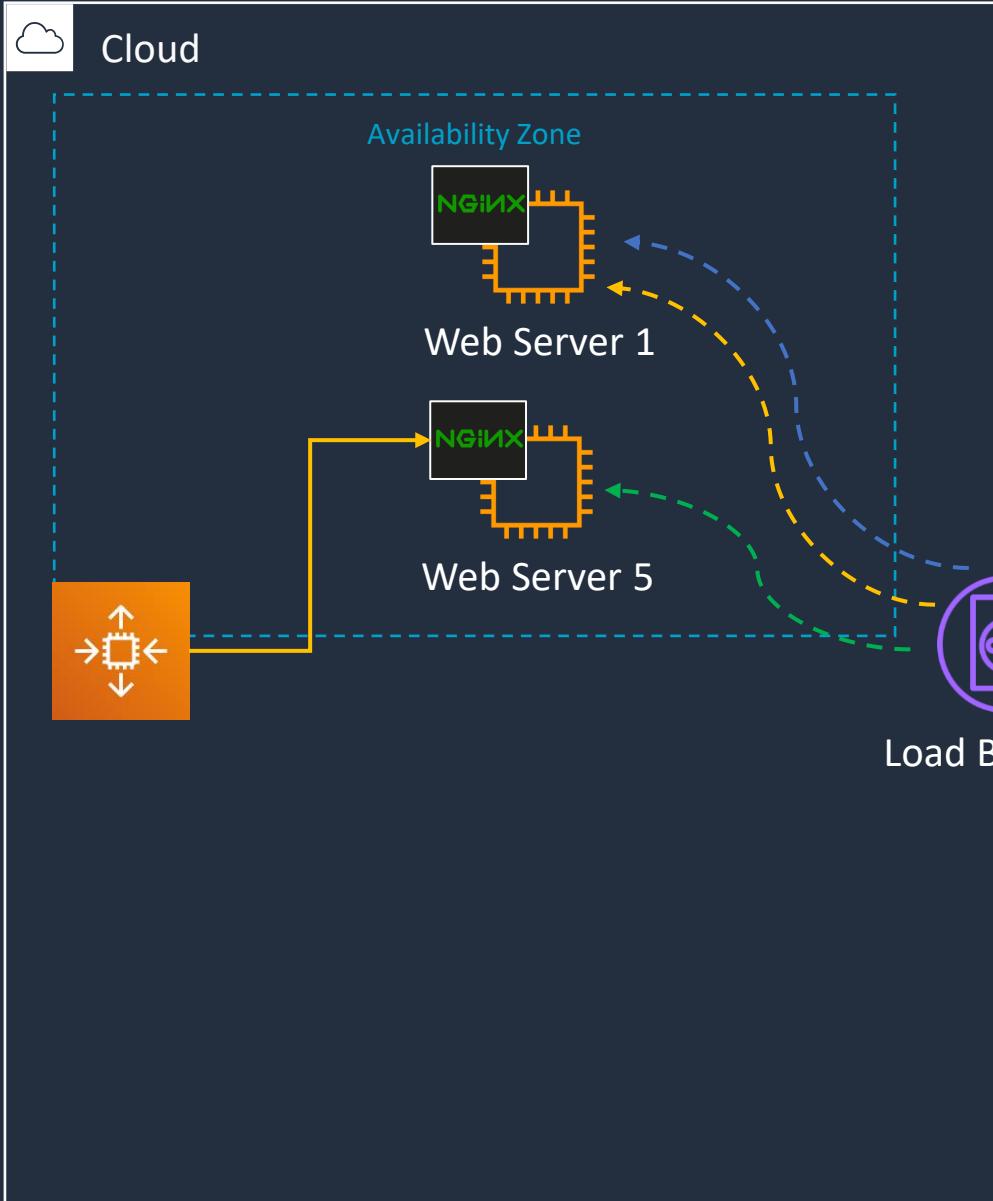


Think of an **availability zone** as a separate data center





High Availability and Fault Tolerance



Think of an **availability zone** as
a separate data center



User 1



User 2



User 3



Durability and Availability

Durability

Durability is protection against:

- Data loss
- Data corruption
- S3 offers 11 9s durability (99.99999999)

If you store 10 million objects in S3, then you can expect to lose one object every 10,000 years!

Availability

Availability is a measurement of:

- The amount of time the data is available to access
- Expressed as a percent of time per year
- E.g. 99.99%

Amazon Elastic Load Balancing





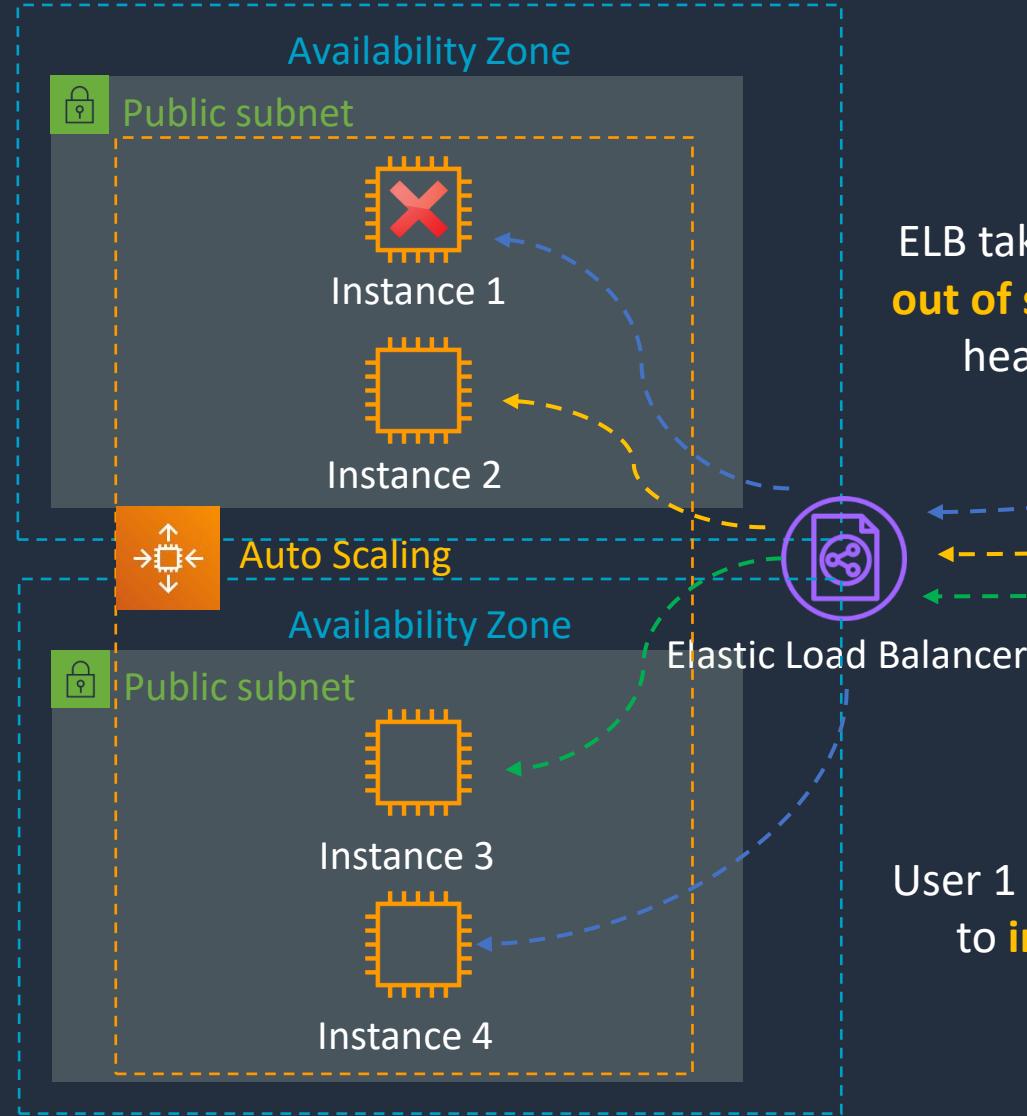
Amazon Elastic Load Balancing

- Provides high availability and fault tolerance
- Targets include:
 - Amazon EC2 instances
 - Amazon ECS containers
 - IP addresses
 - Lambda functions
 - Other load balancers



Amazon Elastic Load Balancing

EC2 Auto Scaling
terminates
instance 1



ELB takes instance 1
out of service (failed
health check)

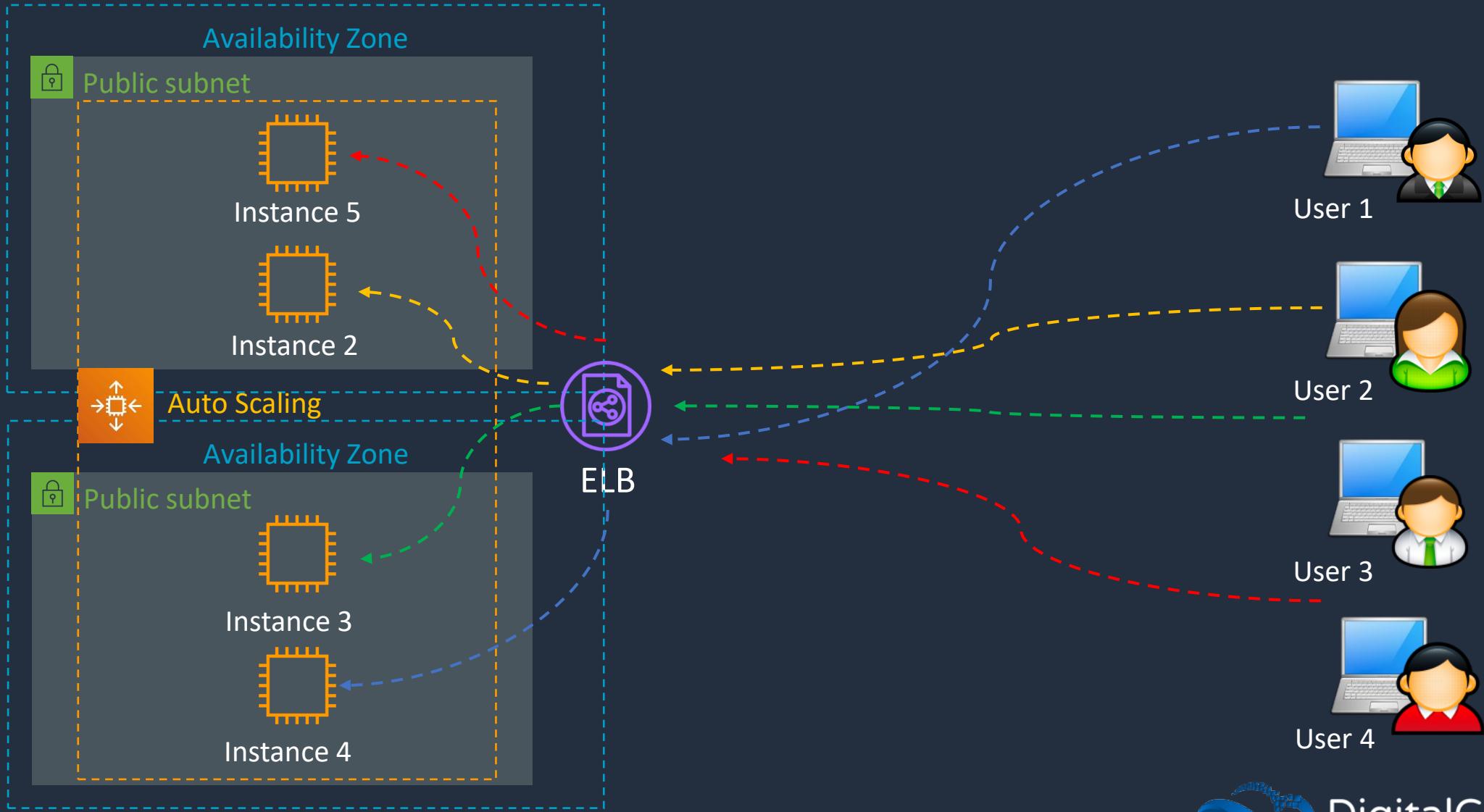
User 1 is connected
to **instance 4**





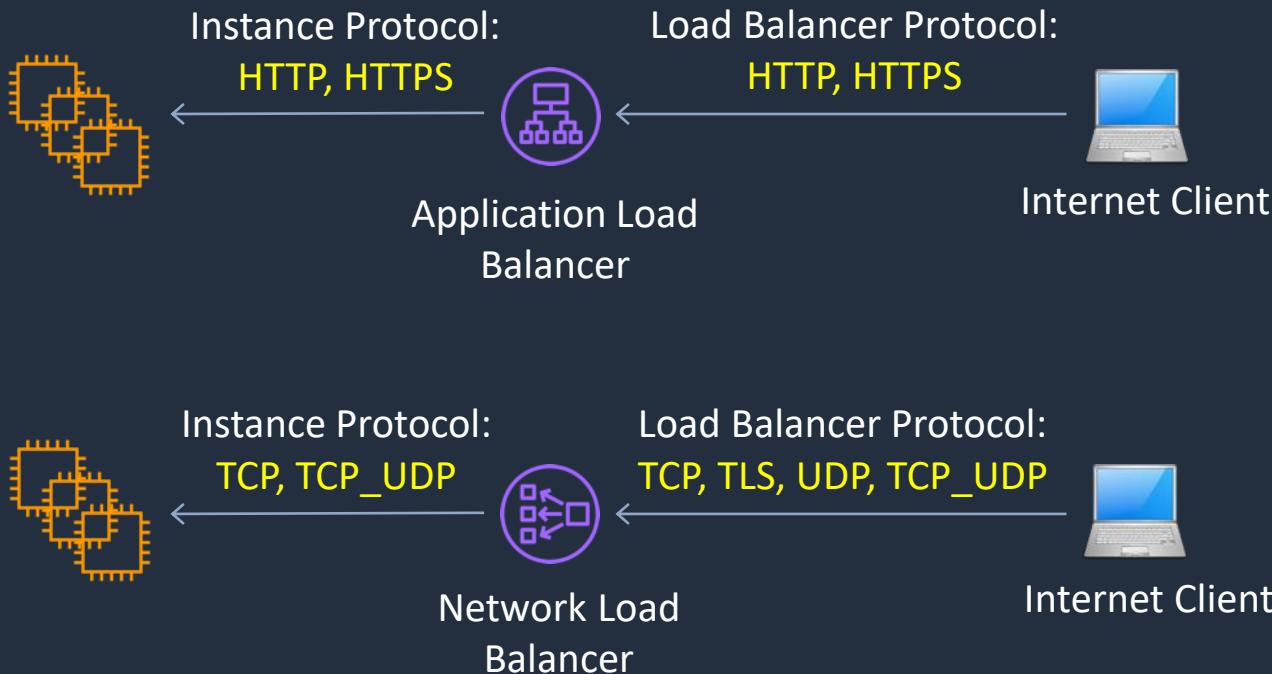
Amazon Elastic Load Balancing

EC2 Auto Scaling
launches
instance 5





Types of Elastic Load Balancer (ELB)



Application Load Balancer

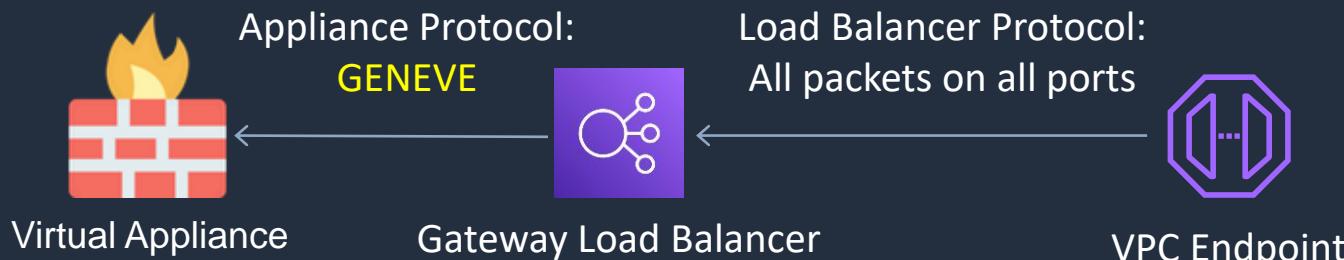
- Operates at the request level
- Routes based on the content of the request (L7)
- Supports path-based routing, host-based routing, query string parameter-based routing, and source IP address-based routing
- Supports instances, IP addresses, Lambda functions and containers as targets

Network Load Balancer

- Operates at the connection level
- Routes connections based on IP protocol data (L4)
- Offers ultra high performance, low latency and TLS offloading at scale
- Can have a static IP / Elastic IP
- Supports UDP and static IP addresses as targets



Types of Elastic Load Balancer (ELB)



Gateway Load Balancer

- Used in front of virtual appliances such as firewalls, IDS/IPS, and deep packet inspection systems
- Operates at Layer 3 – listens for all packets on all ports
- Forwards traffic to the TG specified in the listener rules
- Exchanges traffic with appliances using the GENEVE protocol on port 6081



ELB Use Cases

Application Load Balancer

- Web applications with L7 routing (HTTP/HTTPS)
- Microservices architectures (e.g. Docker containers)
- Lambda targets

Network Load Balancer

- TCP and UDP based applications
- Ultra-low latency
- Static IP addresses
- VPC endpoint services



ELB Use Cases

Gateway Load Balancer

- Deploy, scale and manage 3rd party virtual network appliances
- Centralized inspection and monitoring
- Firewalls, intrusion detection and prevention systems, and deep packet inspection systems

ALB and NLB Deployments

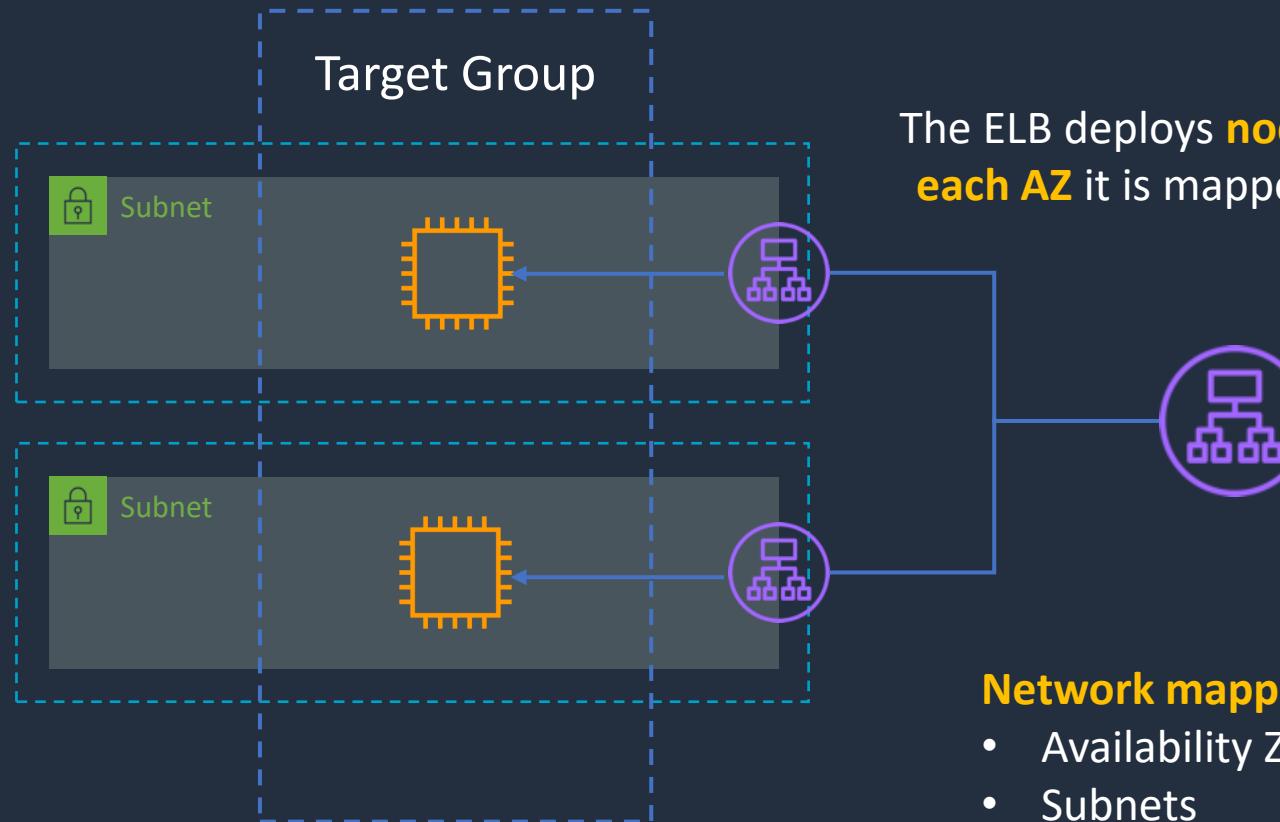




ELB Deployments

Target groups define:

- Target type
- Target protocol / port
- VPC
- Health checks
- Registered targets



The ELB deploys **nodes** in
each AZ it is mapped to

The ELB listener defines:

- Listener protocol /port
- Routing rules
- SSL/TLS certificate

Network mappings define:

- Availability Zones
- Subnets



ELB Supported Configurations

Application Load Balancer (ALB)

- Target type can be:
 - instance
 - ip
 - lambda
- Target group protocol must be HTTP/HTTPS
- Health check protocol must be HTTP/HTTPS
- Can define rules for advanced request routing

Network Load Balancer (NLB)

- Target type can be:
 - instance
 - ip
 - alb
- Target group protocol must be TCP/UDP
- Any health check protocol is supported
- Can define elastic IP per subnet



Routing with Application Load Balancer (ALB)

Application Load Balancer (ALB)



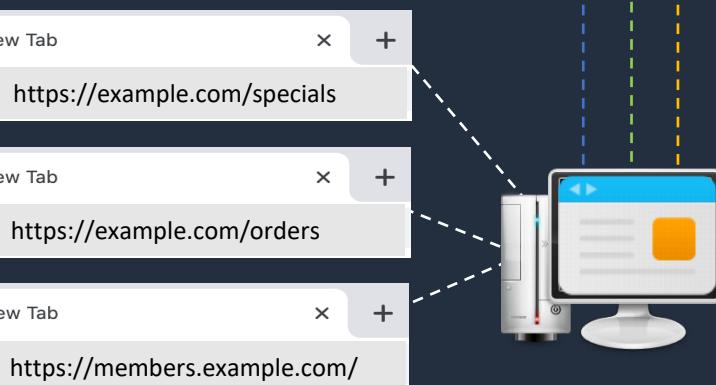
A **rule** is configured on the **listener**.

ALBs listen on **HTTP/HTTPS**

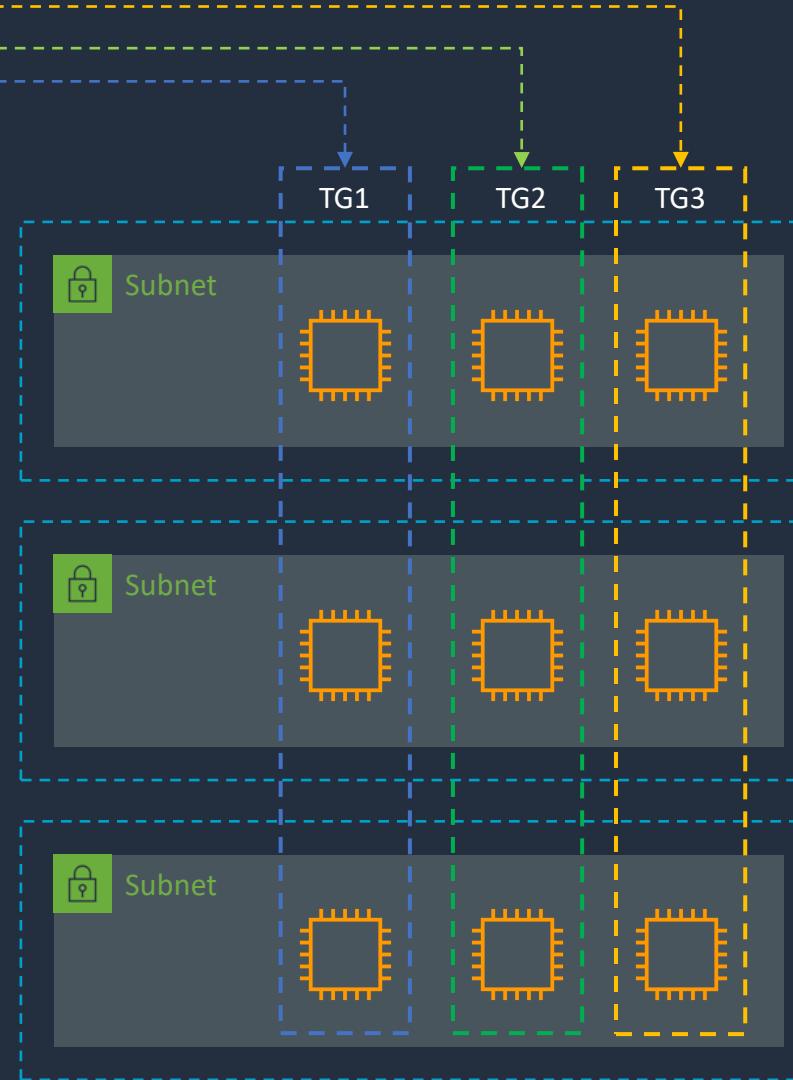
Path-based routing

Requests can be routed based on the **path** in the **URL**

Requests can also be routed based on the **host** field in the **HTTP header**



Host-based routing



Target groups are used to route requests to registered targets

Targets can be EC2 instances, IP addresses, and Lambda functions



Routing with Network Load Balancer (NLB)

Network Load Balancer (NLB)



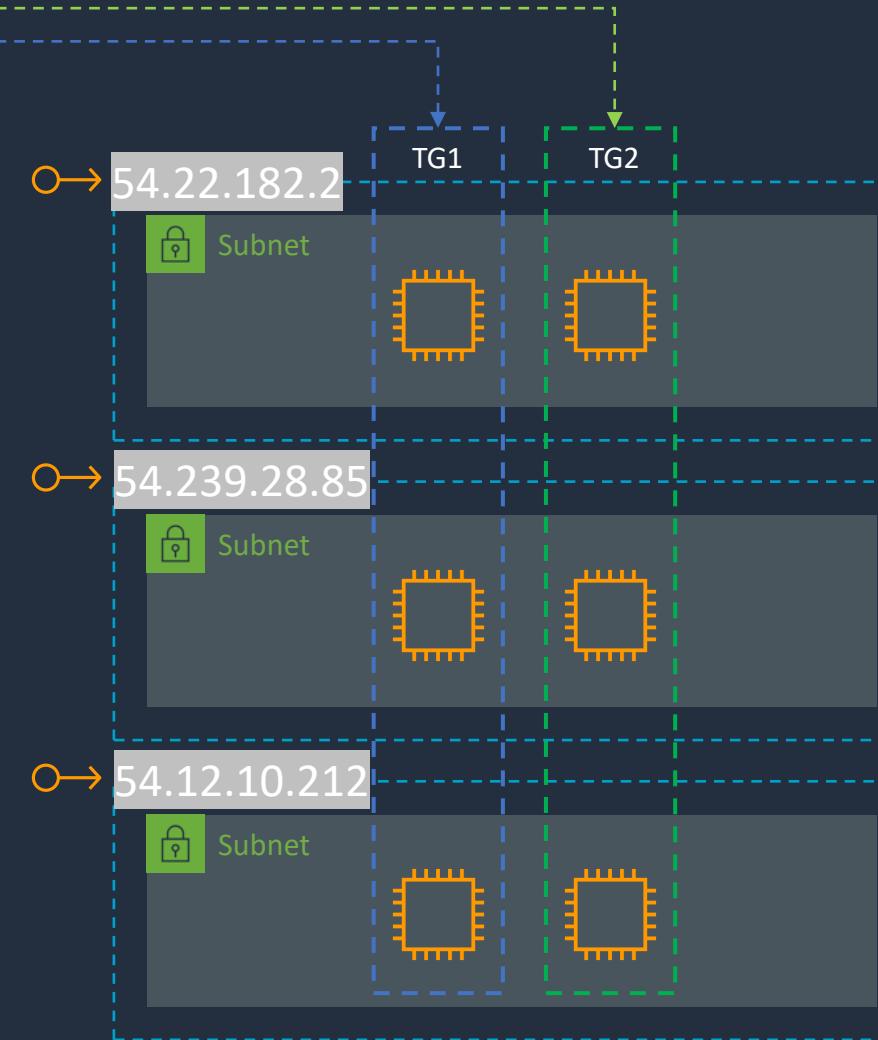
NLBs listen on
TCP, TLS, UDP or
TCP_UDP

Requests are routed based
on **IP protocol** data

NLB **nodes** can
have **elastic IPs** in
each subnet



A **separate** listener on a
unique port is required
for routing



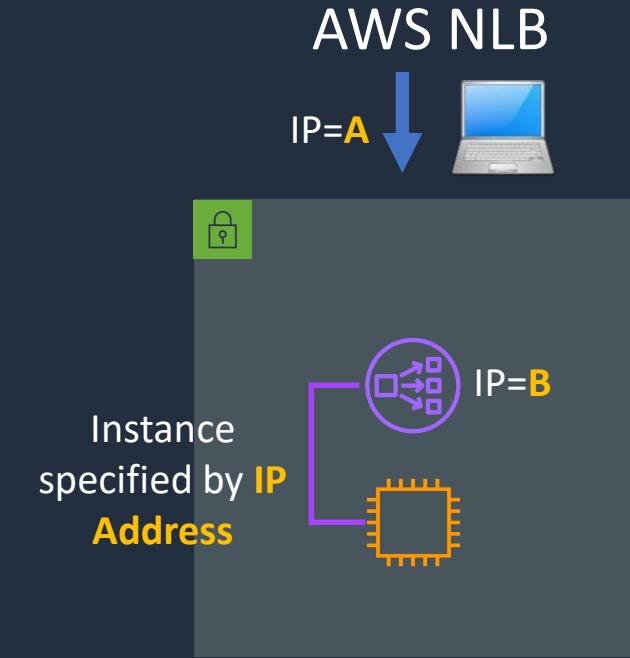
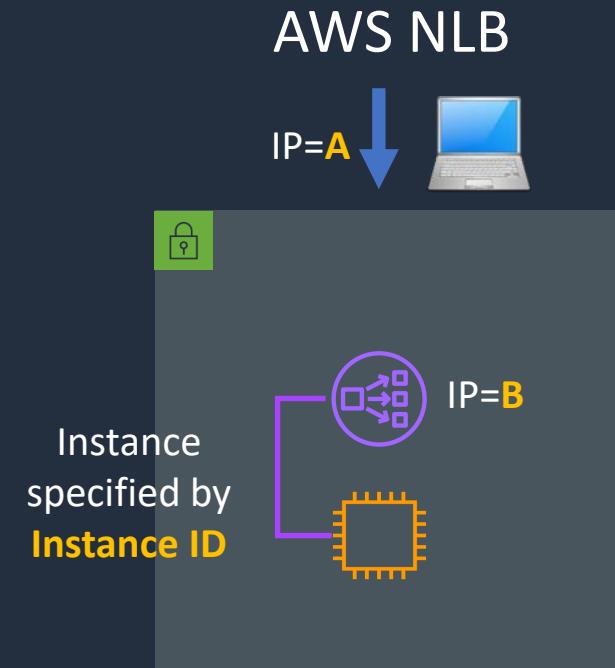
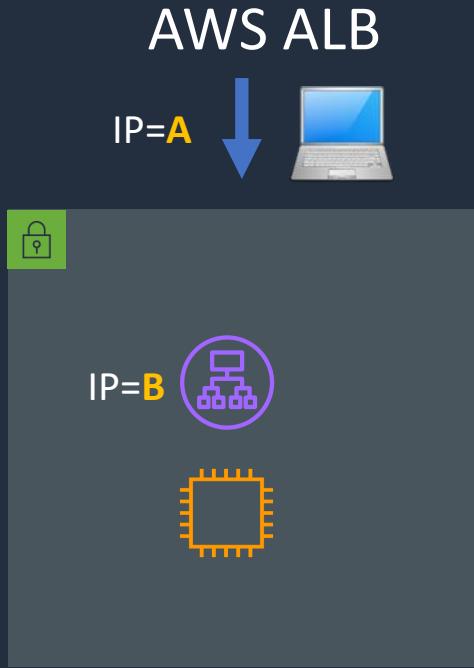
Targets can be EC2
instances, IP addresses,
or ALBs

Targets can be
outside a VPC
(e.g. on-premises)



What's the Source IP Address the App sees?

Note: X-forwarded-for can be used with ALB to capture client IPs



Applicable to TCP and TLS – for UDP and TCP_UDP should be IP=A

CLB and ALB use private IP of their ENIs as source address

Source	Protocol	Port
IP=B	TCP	80

Source	Protocol	Port
IP=A	TCP	80

Source	Protocol	Port
IP=B	TCP	80

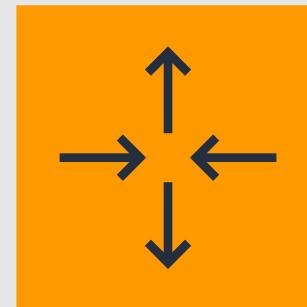
When using an NLB with a VPC Endpoint or AWS GA source IPs are private IPs of NLB nodes

<https://aws.amazon.com/premiumsupport/knowledge-center/elb-capture-client-ip-addresses/>

Create an Application Load Balancer



Amazon EC2 Scaling Policies

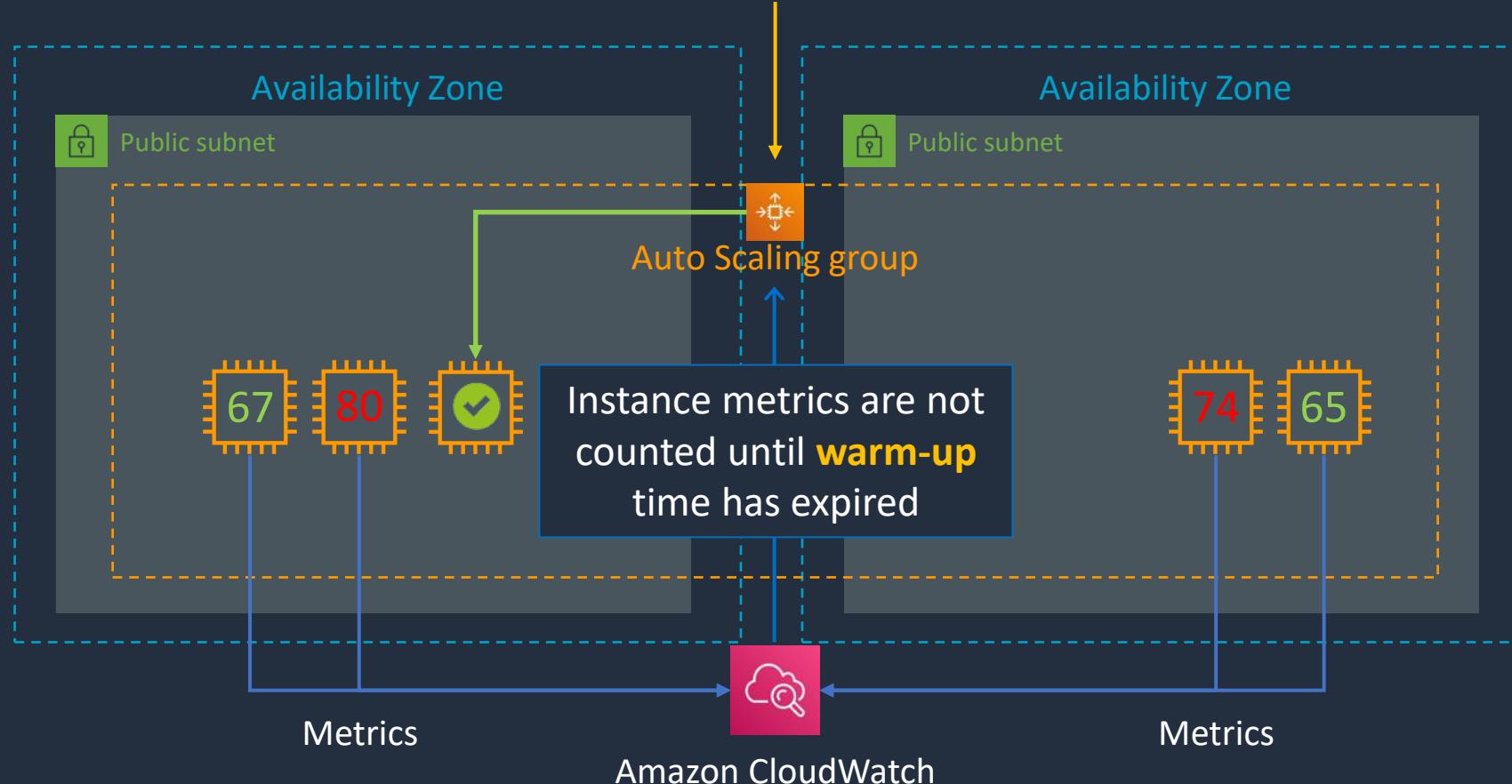




Dynamic Scaling – Target Tracking

AWS recommend scaling on metrics
with a **1-minute** frequency

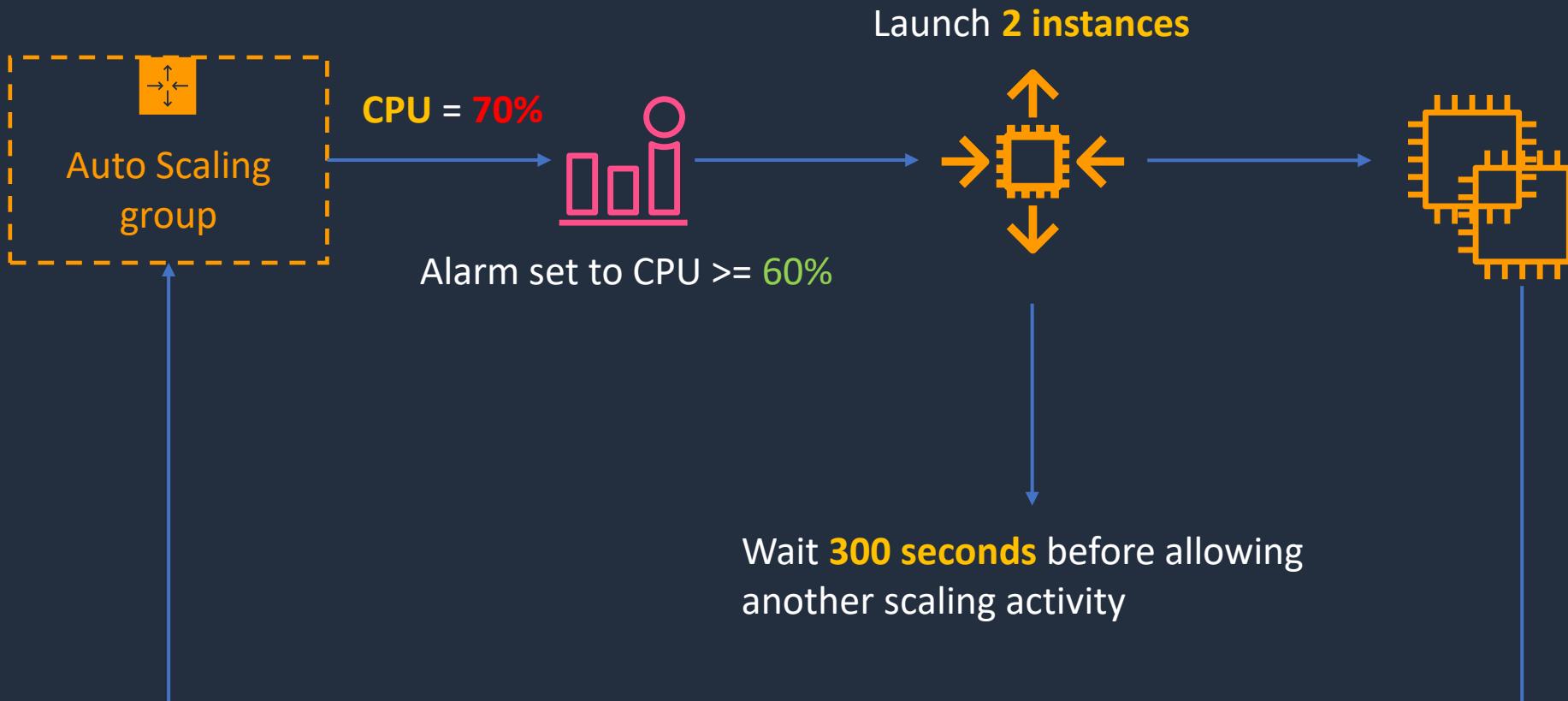
$$\text{ASGAverageCPUUtilization} = 60\%$$



$$\text{Average CPU} = 71.5\%$$

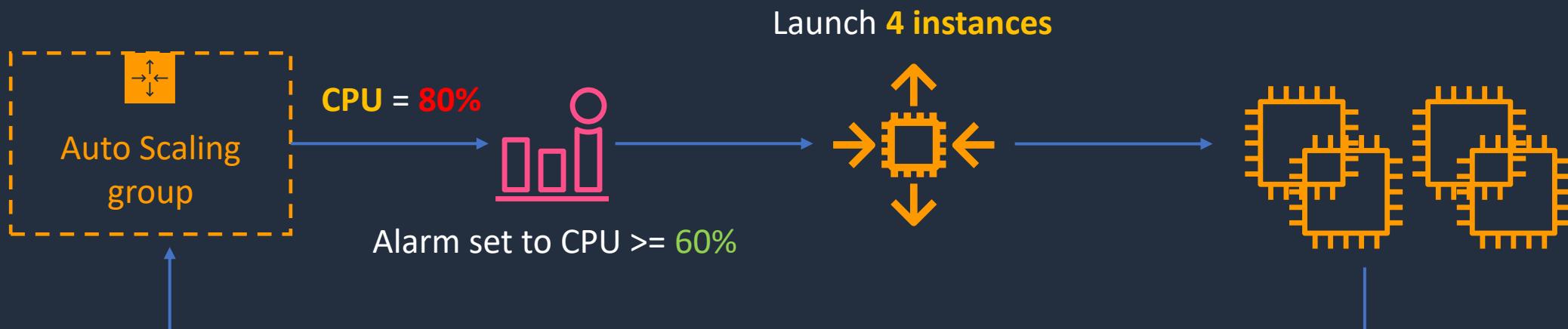
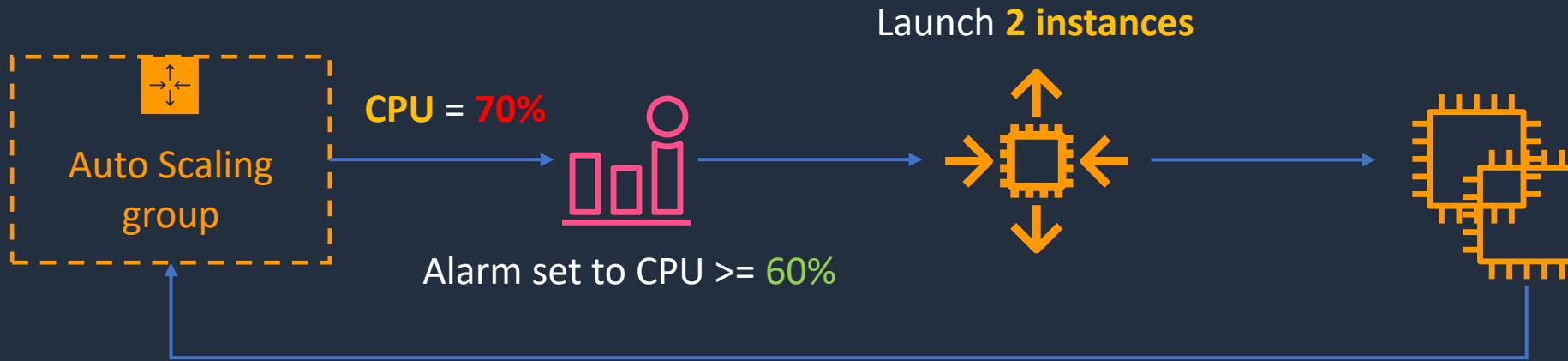


Dynamic Scaling – Simple Scaling



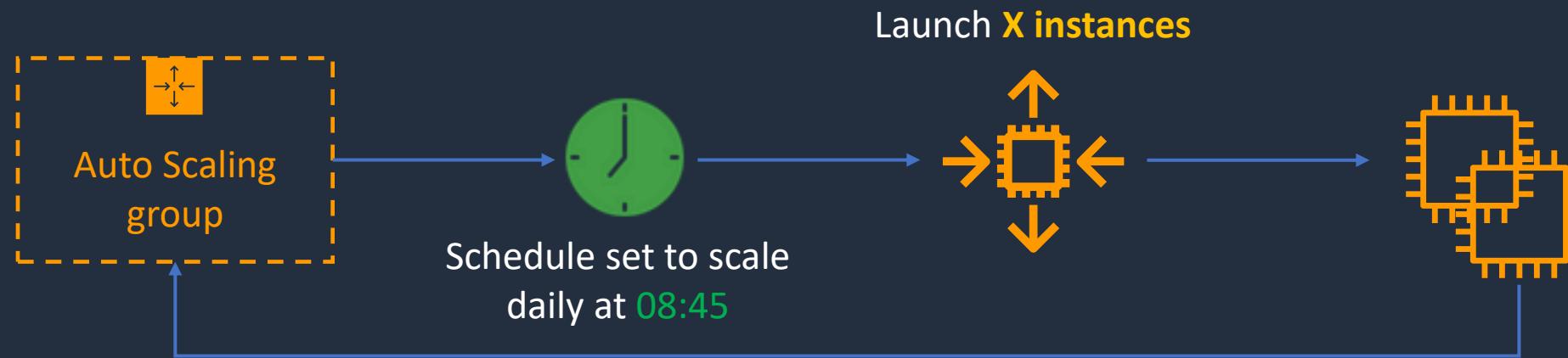


Dynamic Scaling – Step Scaling

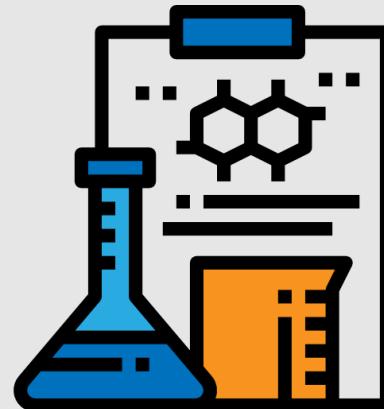




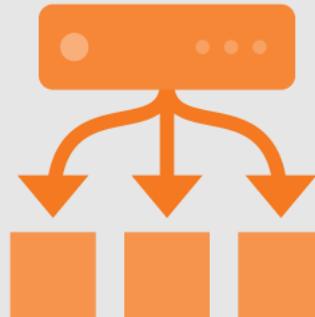
Scheduled Scaling



Create a Scaling Policy



Cross-Zone Load Balancing





Cross-Zone Load Balancing

When cross-zone load balancing is enabled:

- Each load balancer node distributes traffic across the registered targets in all enabled Availability Zones

When cross-zone load balancing is disabled:

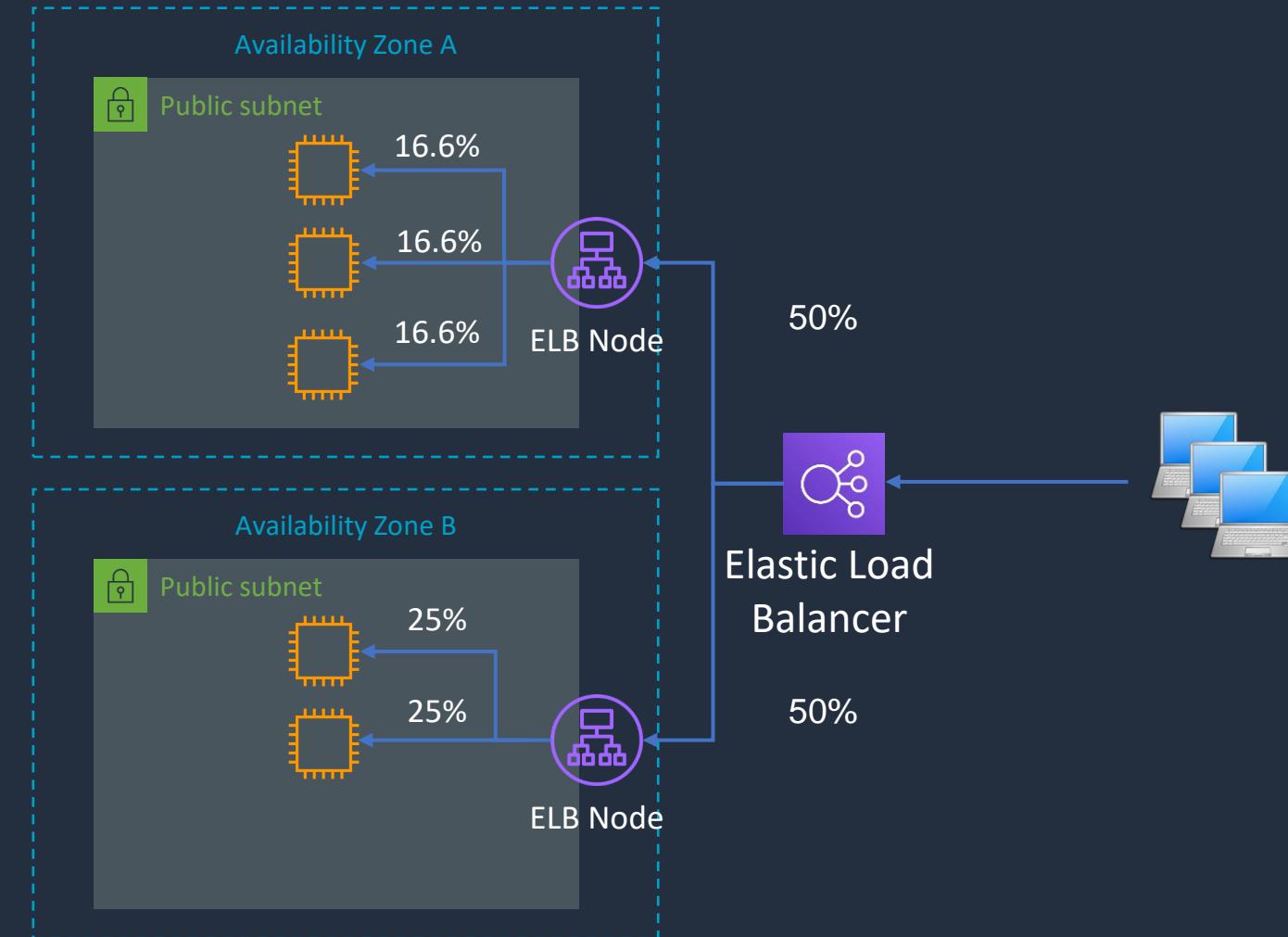
- Each load balancer node distributes traffic only across the registered targets in its Availability Zone
- With Application Load Balancers, cross-zone load balancing is **always enabled**
- With Network Load Balancers and Gateway Load Balancers, cross-zone load balancing is **disabled by default**



Cross-Zone Load Balancing - Disabled

If cross-zone load balancing is disabled:

- Each of the three targets in Availability Zone A receives 16.6% of the traffic
- Each of the two targets in Availability Zone B receives 25% of the traffic

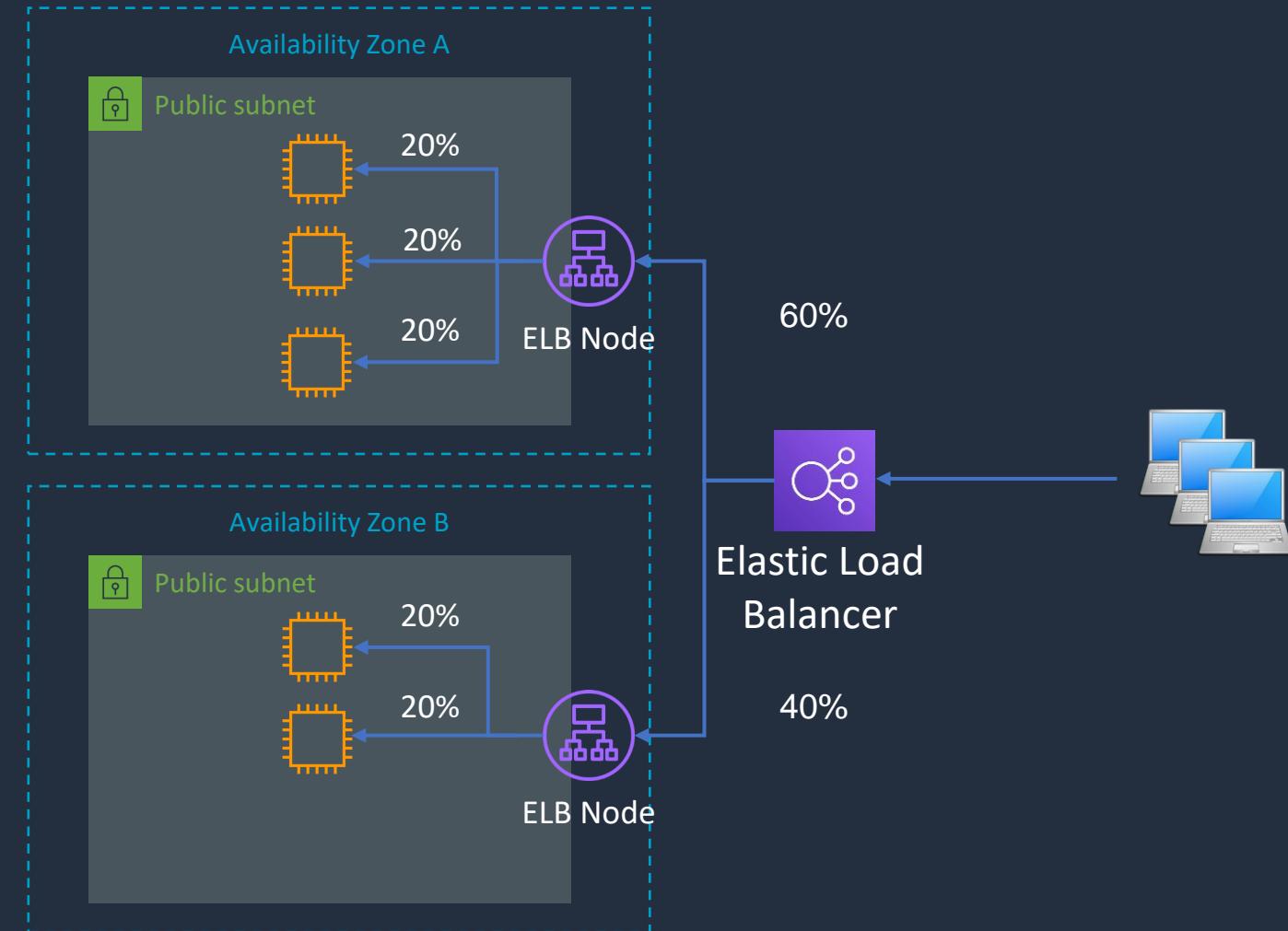




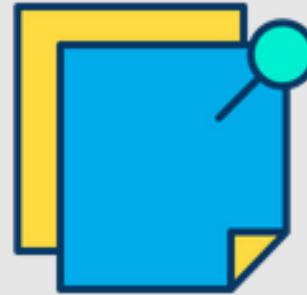
Cross-Zone Load Balancing - Enabled

If cross-zone load balancing is enabled:

- Each of the five targets receives 20% of the traffic



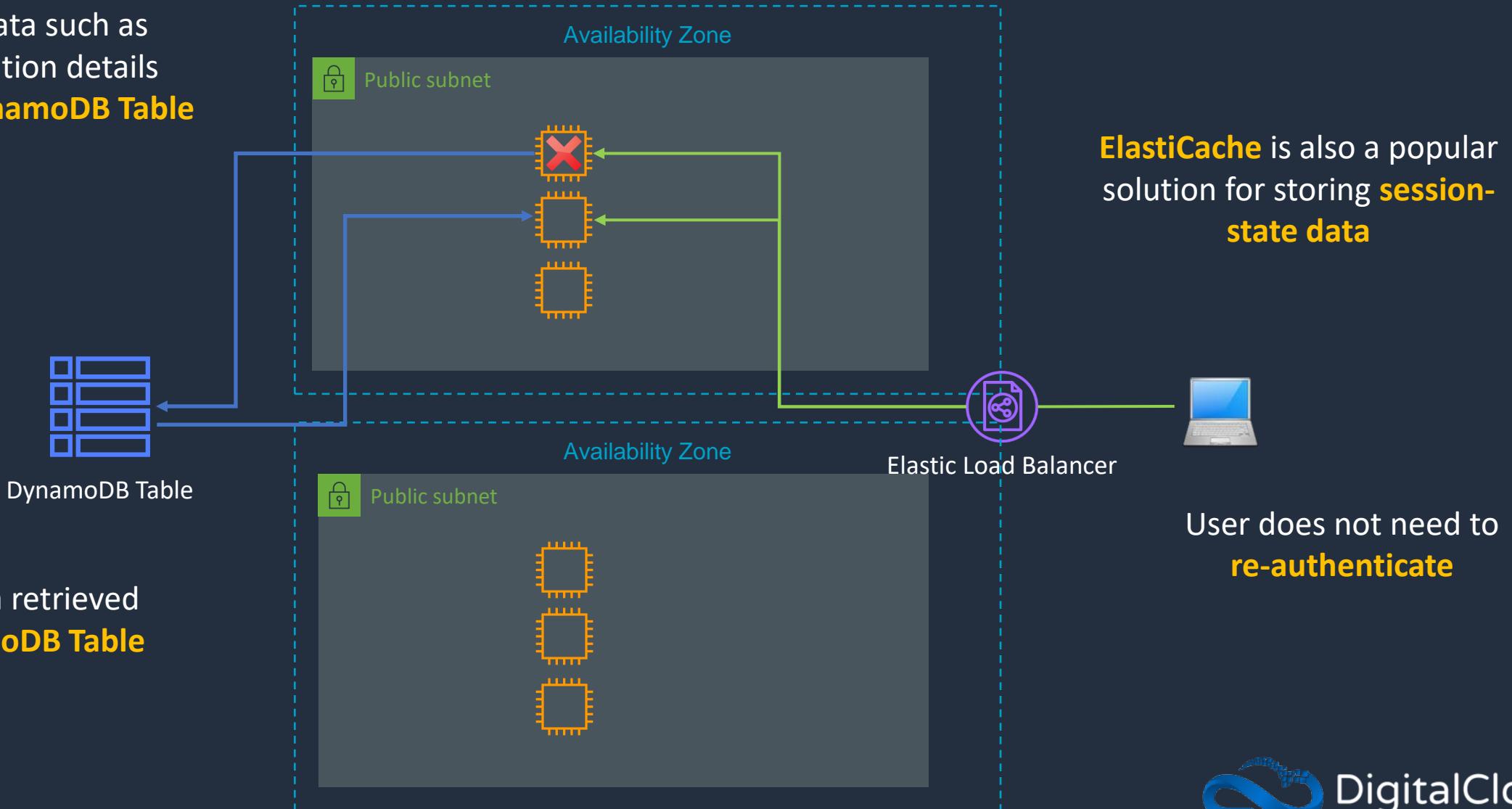
Session State and Session Stickiness





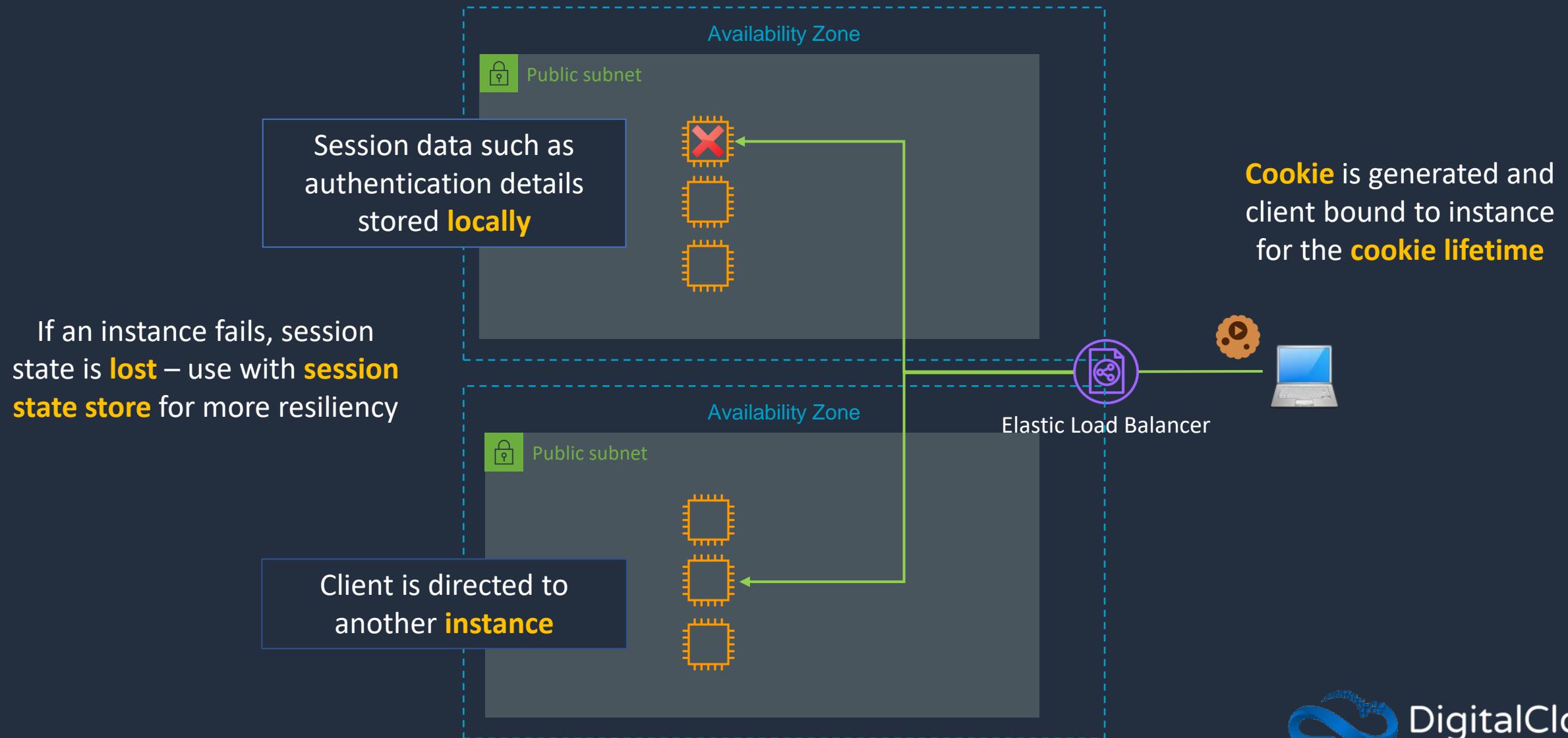
Storing Session State

Session data such as authentication details stored in **DynamoDB Table**





Sticky Sessions



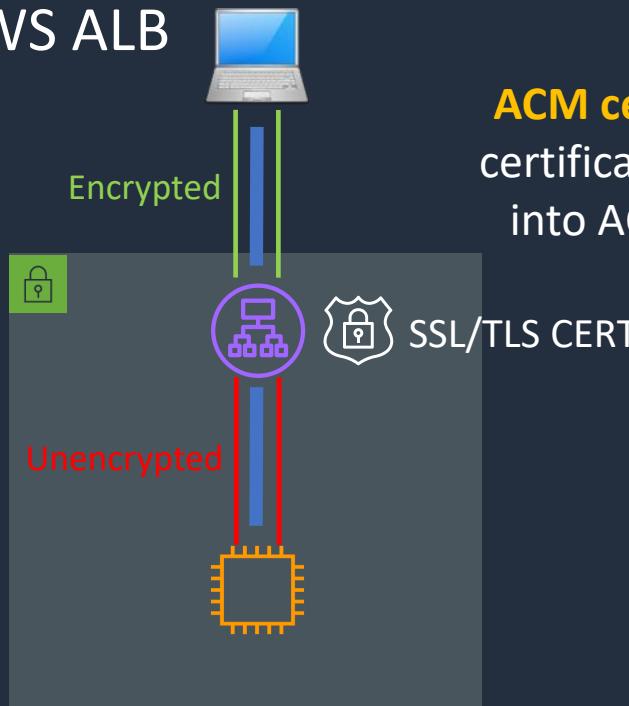
Secure Listeners for ELB





SSL/TLS Termination

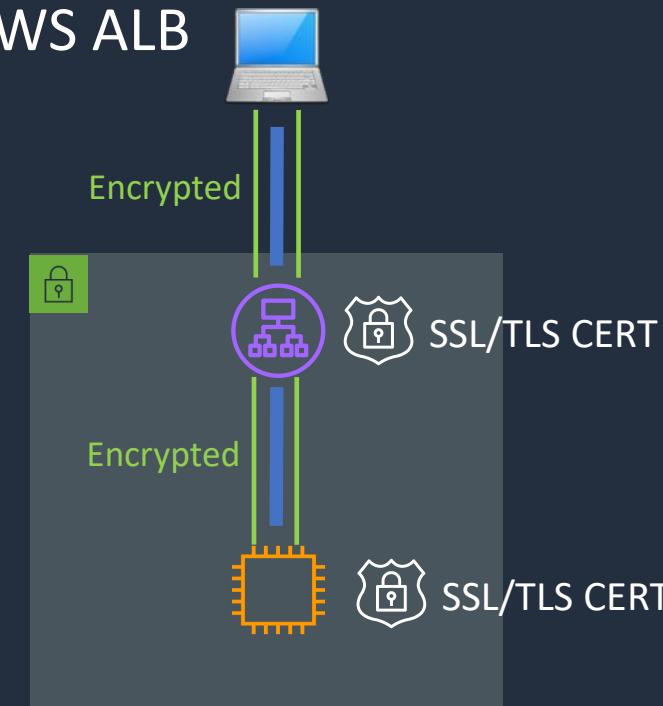
AWS ALB



ACM certificate or
certificate imported
into ACM or IAM

With a **L7 ELB** a **new connection**
is established with the instance

AWS ALB



Self-signed certificate can be
used, or a certificate from a
private certificate authority

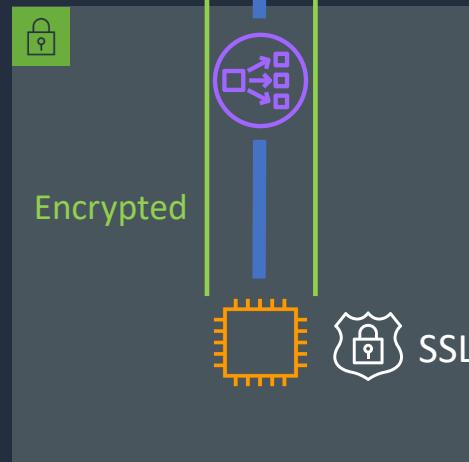


SSL/TLS Termination

AWS NLB



Single encrypted connection



AWS NLB



Encrypted



Encrypted



SSL/TLS CERT



SSL/TLS CERT

Public certificate
must be used

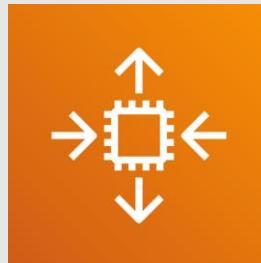
Create a Secure Listener



Network Load Balancer (NLB)



Architecture Patterns – Auto Scaling and ELB





Architecture Patterns - Auto Scaling and ELB

Requirement

High availability and elastic scalability for web servers

Solution

Use Amazon EC2 Auto Scaling and an Application Load Balancer across multiple AZs

Low-latency connections over UDP to a pool of instances running a gaming application

Use a Network Load Balancer with a UDP listener

Clients need to whitelist static IP addresses for a highly available load balanced application in an AWS Region

Use an NLB and create static IP addresses in each AZ



Architecture Patterns - Auto Scaling and ELB

Requirement

Application on EC2 in an Auto Scaling group requires disaster recovery across Regions

Application on EC2 must scale in larger increments if a big increase in traffic occurs, compared to small increases in traffic

Need to scale EC2 instances behind an ALB based on the number of requests completed by each instance

Solution

Create an ASG in a second Region with the capacity set to 0. Take snapshots and copy them across Regions (Lambda or DLM)

Use Auto Scaling with a Step Scaling policy and configure a larger capacity increase

Configure a target tracking policy using the ALBRequestCountPerTarget metric



Architecture Patterns - Auto Scaling and ELB

Requirement

Application runs on EC2 behind an ALB.
Once authenticated users should not
need to reauthenticate if an instance
fails

Solution

Use session state store such as
DynamoDB or ElastiCache

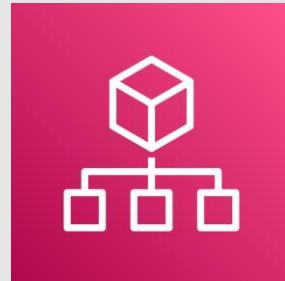
Company is deploying an IDS/IPS
system using virtual appliances and
needs to scale horizontally

Deploy a Gateway Load Balancer in
front of the virtual appliances

SECTION 5

AWS Organizations

AWS Organizations





AWS Organizations

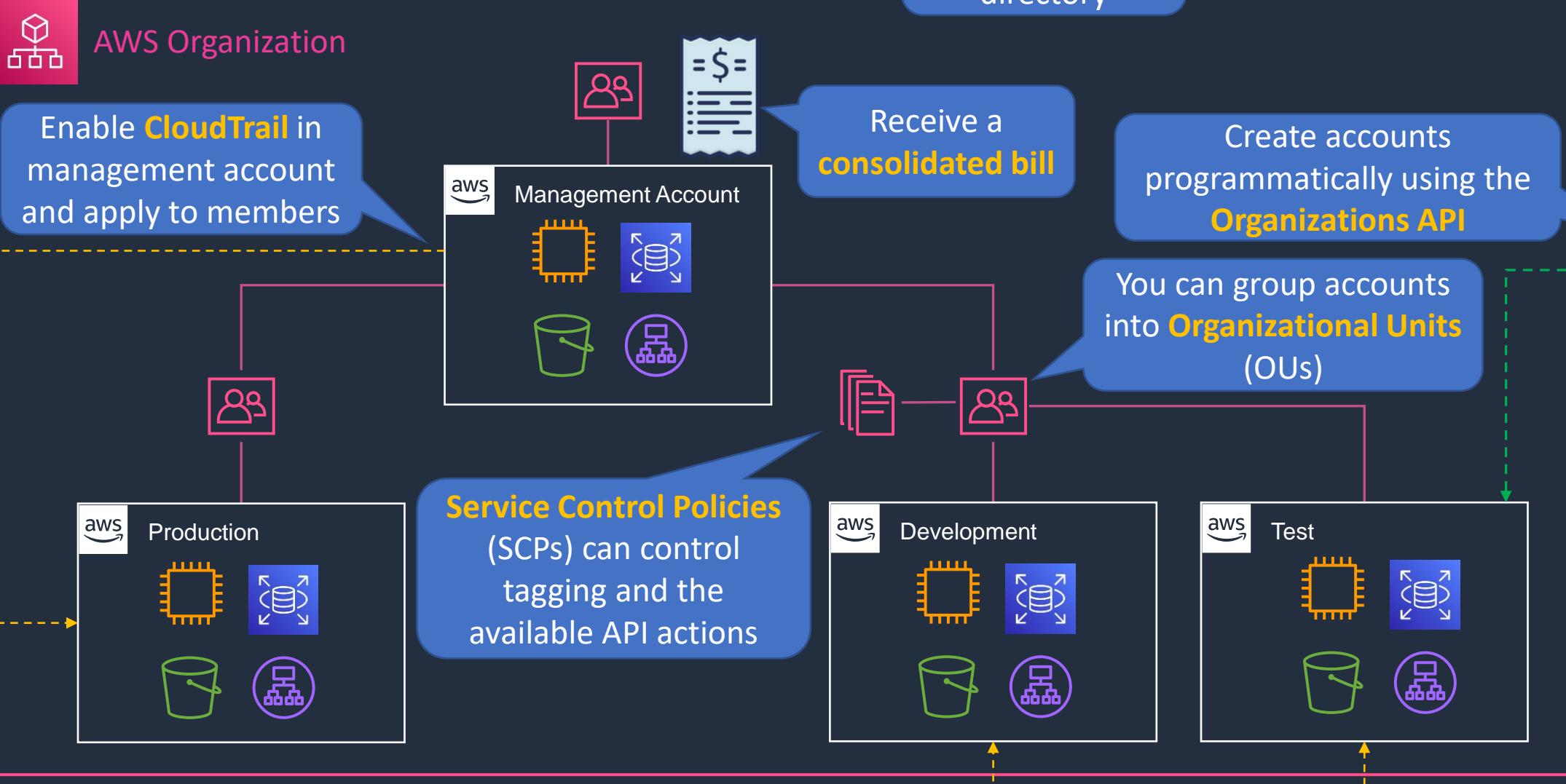
- AWS organizations allows you to consolidate multiple AWS accounts into an organization that you create and centrally manage
- Available in two feature sets:
 - **Consolidated Billing**
 - **All features**
- Includes root accounts and organizational units
- Policies are applied to root accounts or OUs
- Consolidated billing includes:
 - **Paying Account** – independent and cannot access resources of other accounts
 - **Linked Accounts** – all linked accounts are independent



AWS Organizations



Enable AWS SSO
using on-prem
directory

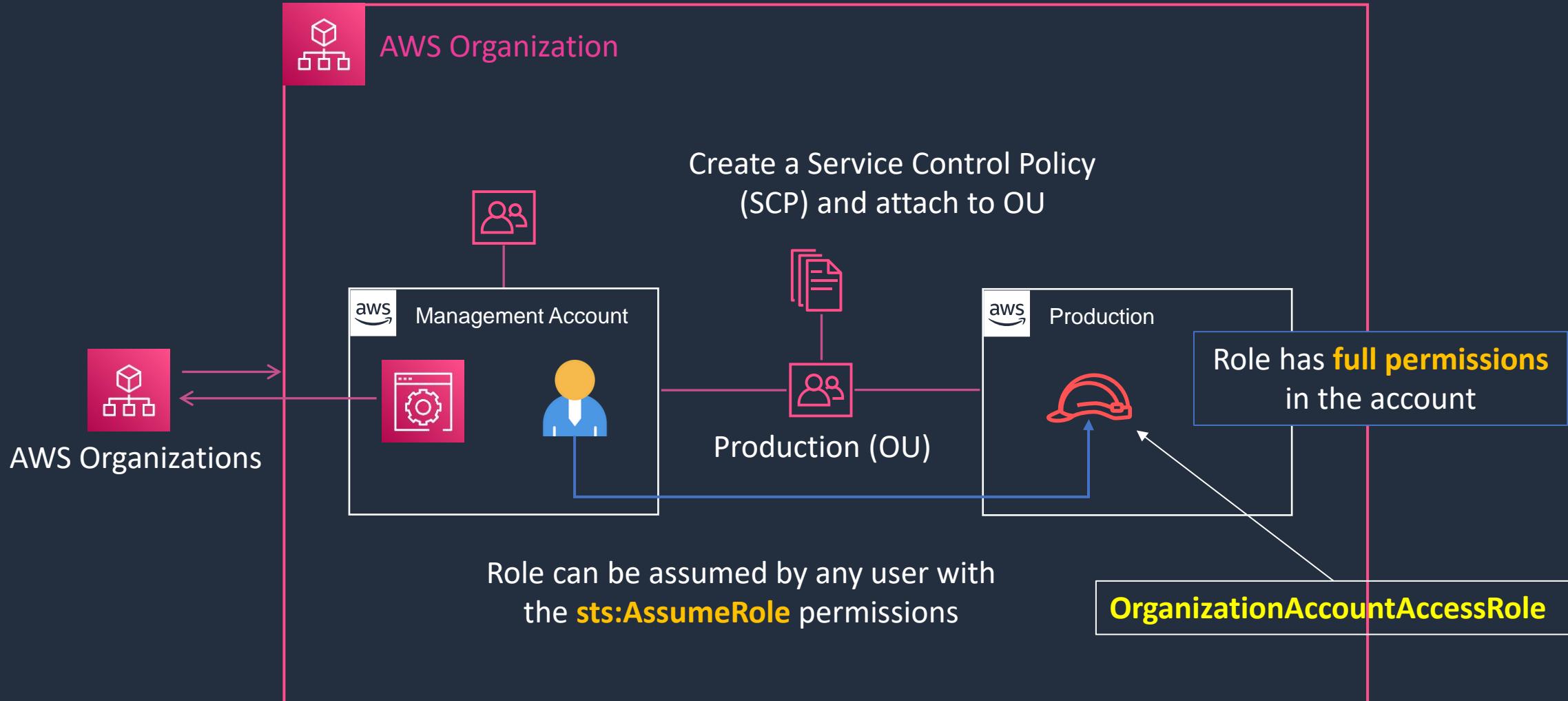


Create AWS Organization and Add Account





Account Configuration

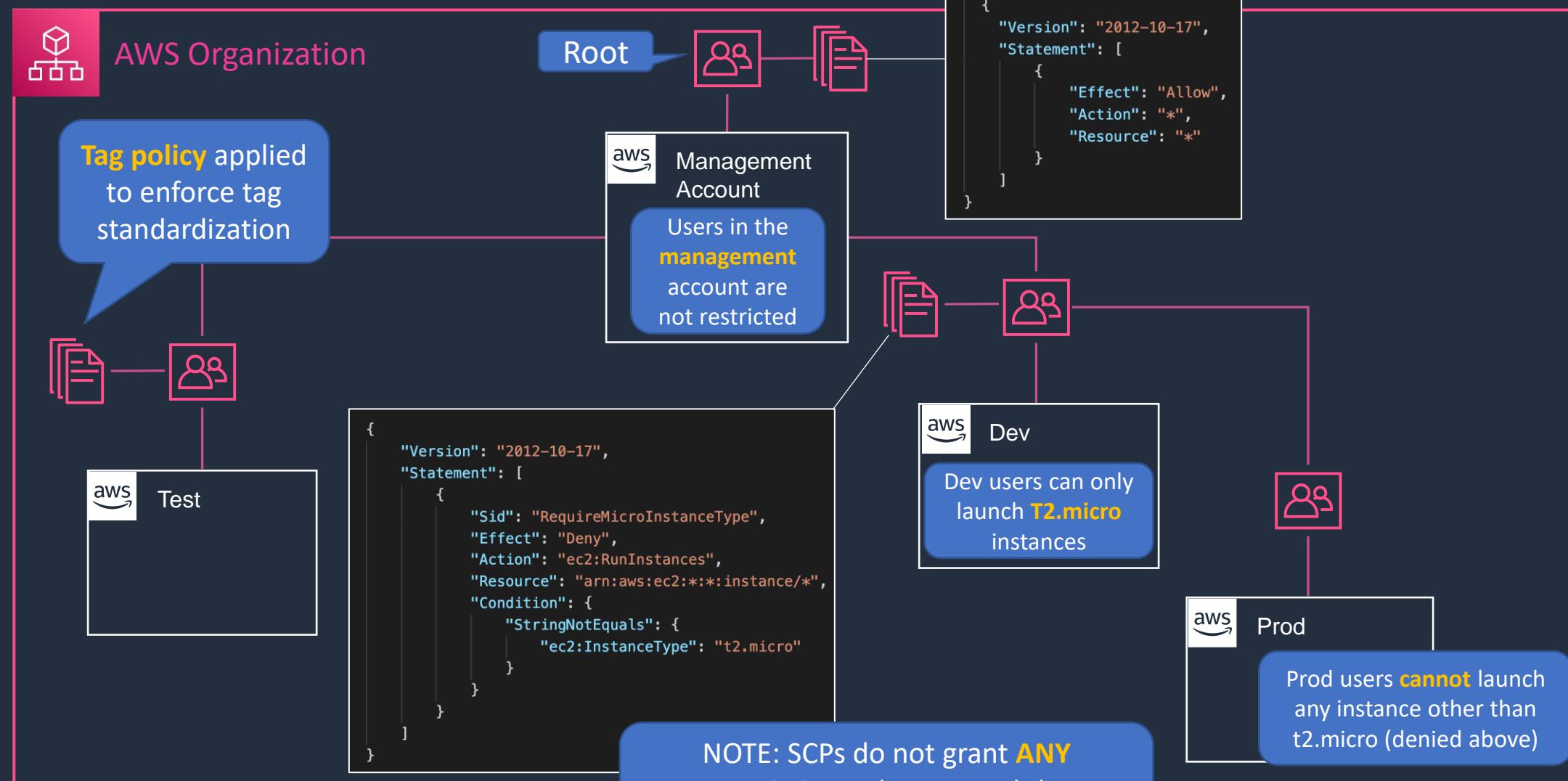


Service Control Policies (SCPs)



Service Control Policies

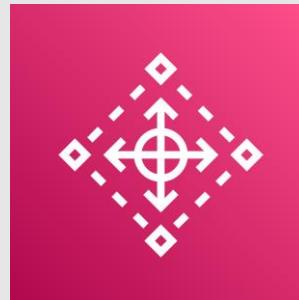
SCPs control the maximum available permissions



Create Service Control Policy (SCP)



AWS Control Tower



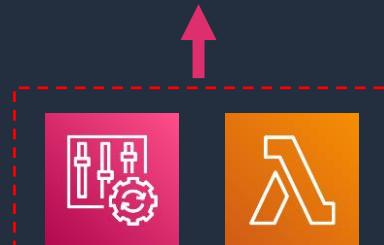
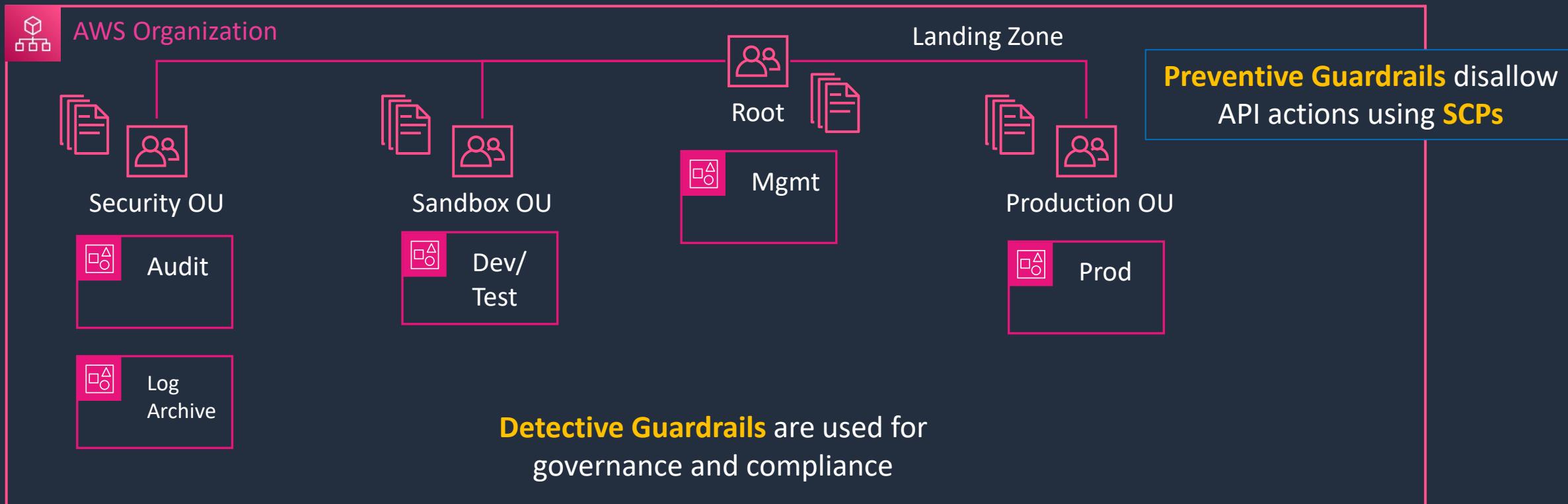


AWS Control Tower

Directory source can be
IAM Identity Center, SAML
2.0 IdP, or Microsoft AD



A **landing zone** is a well-architected multi-account baseline





AWS Control Tower – Shared Accounts

- **Management account** – The AWS account used to launch AWS Control Tower. The root user and IAM administrator have full access to all resources in the landing zone
- **Log archive account** – Contains a central Amazon S3 bucket for storing a copy of all AWS CloudTrail and AWS Config log files for all other accounts in the landing zone
- **Audit account** – Aggregates and stores logs collected from all other accounts in the landing zone. Secure account with restricted access



Preventive guardrails

- **Purpose:** Preventive guardrails are designed to proactively prevent policy violations before they occur. They enforce compliance and security best practices by restricting certain actions or configurations
- **How They Work:** These guardrails are implemented using AWS **Service Control Policies (SCPs)**. They effectively limit what actions users and roles can perform in the AWS accounts within the organization
- **Examples:**
 - Disallow Deletion of CloudTrail Logs and S3 Logging Buckets
 - Disallow Public Read Access to S3 Buckets
 - Require Encryption on EBS Volumes
 - Disallow RDP/SSH Access from 0.0.0.0/0



Detective Guardrails

- **Purpose:** Detective guardrails are designed to monitor and report on policy violations or non-compliant activities that have already occurred. They help in identifying misconfigurations or activities that do not adhere to the organization's policies
- **How They Work:** These guardrails are implemented using **AWS Config rules** and **Lambda functions**. They continuously evaluate the configuration of AWS resources and generate alerts or reports when non-compliance is detected
- **Examples:**
 - Detecting publicly accessible Amazon S3 buckets
 - Detect whether versioning is enabled for Amazon S3 buckets
 - Detect whether encryption is enabled for Amazon RDS instances
 - Monitoring for IAM policies that grant overly broad permissions



AWS Organizations vs Control Tower

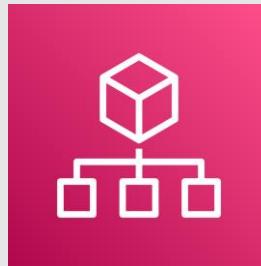
AWS Organizations

- Manage multiple accounts
- Consolidated billing
- Organizational Units
- Service Control Policies
- Tag Policies
- Backup Policies

AWS Control Tower

- Extends capabilities of AWS Organizations
- Landing Zones (best practices)
- Federated Access (IAM Identity Center)
- Centralized Logging
- Account Factory (automation)
- Guardrails (governance rules)

Architecture Patterns – AWS Organizations





Architecture Patterns – AWS Organizations

Requirement

A company needs a method of quickly creating AWS accounts programmatically

Solution

Use the Organizations API to create the accounts programmatically

Users in a member account in AWS Organizations should be restricted from making changes in IAM

User a Service Control Policy (SCP) to deny access to IAM actions

An AWS account must be moved between Organizations

Migrate the account using the AWS Organizations console



Architecture Patterns – AWS Organizations

Requirement

A solutions architect created a new account through the Organizations console and needs to login to launch resources

Multiple member accounts in AWS Organizations require the same permissions to be restricted using SCPs

The developers in a company each have their own AWS accounts for testing. The security team wish to enable central governance

Solution

The architect should switch roles to access the new account

Create an OU and add the member accounts then attach the SCP to the OU

Create an AWS Organization and send an invite to each developer's AWS account to join the Organization

SECTION 6

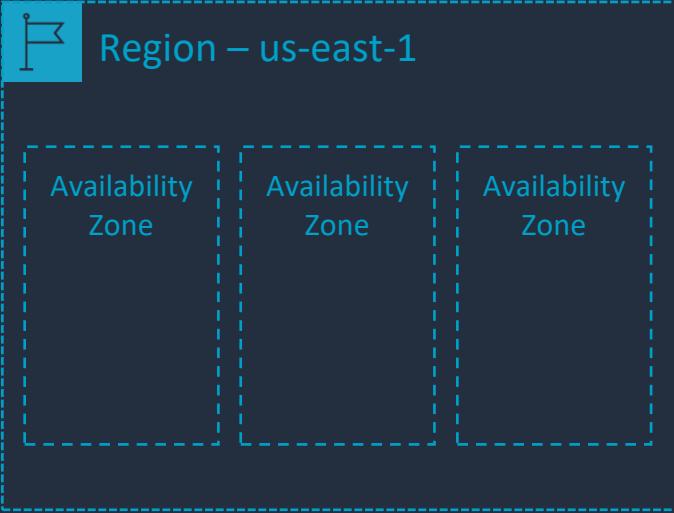
Amazon Virtual Private Cloud (VPC)

The AWS Global Infrastructure





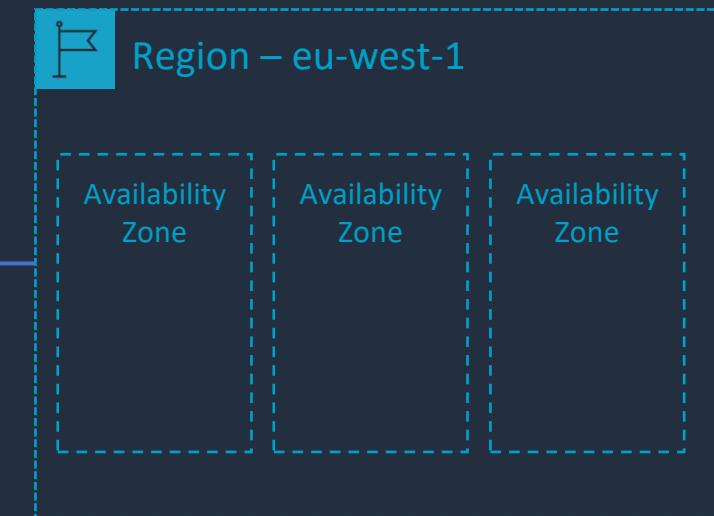
AWS Global Infrastructure



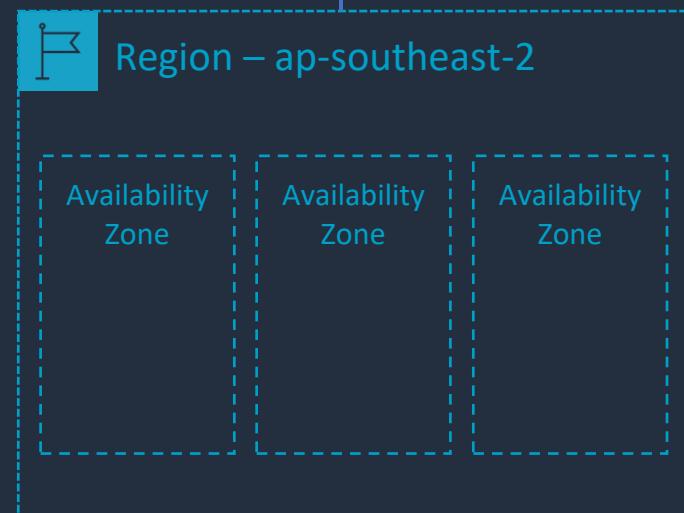
A **Region** is a separate physical location in the world



There are many **Regions** around the world



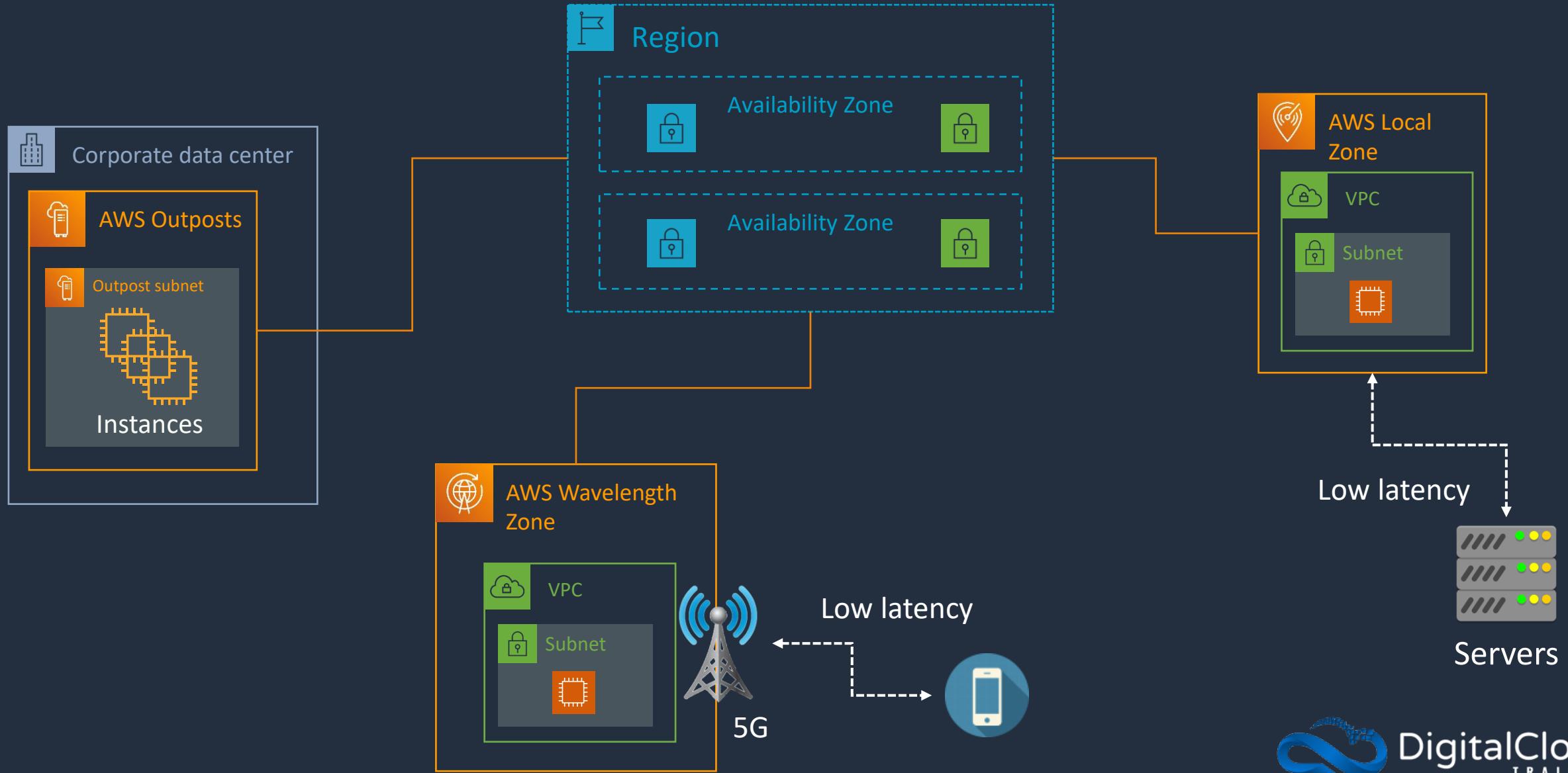
An Availability Zone is composed of **one** or **more** data centers



Each region consists of multiple **Availability Zones**

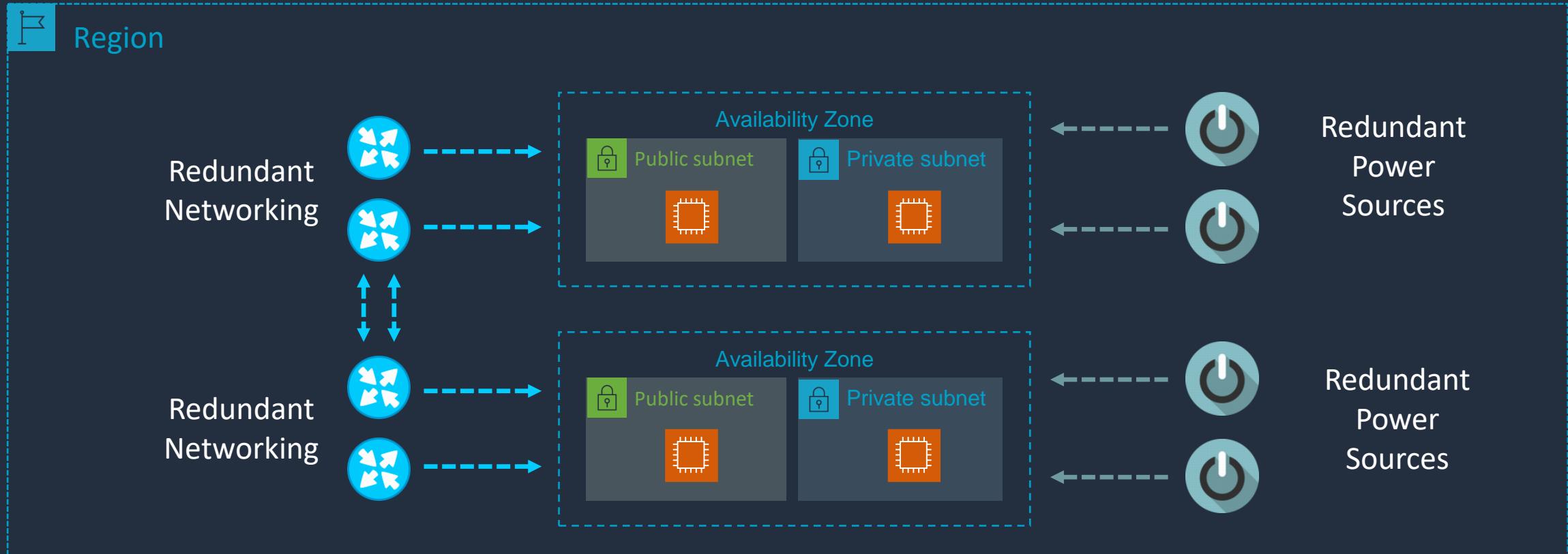


AWS Global Infrastructure



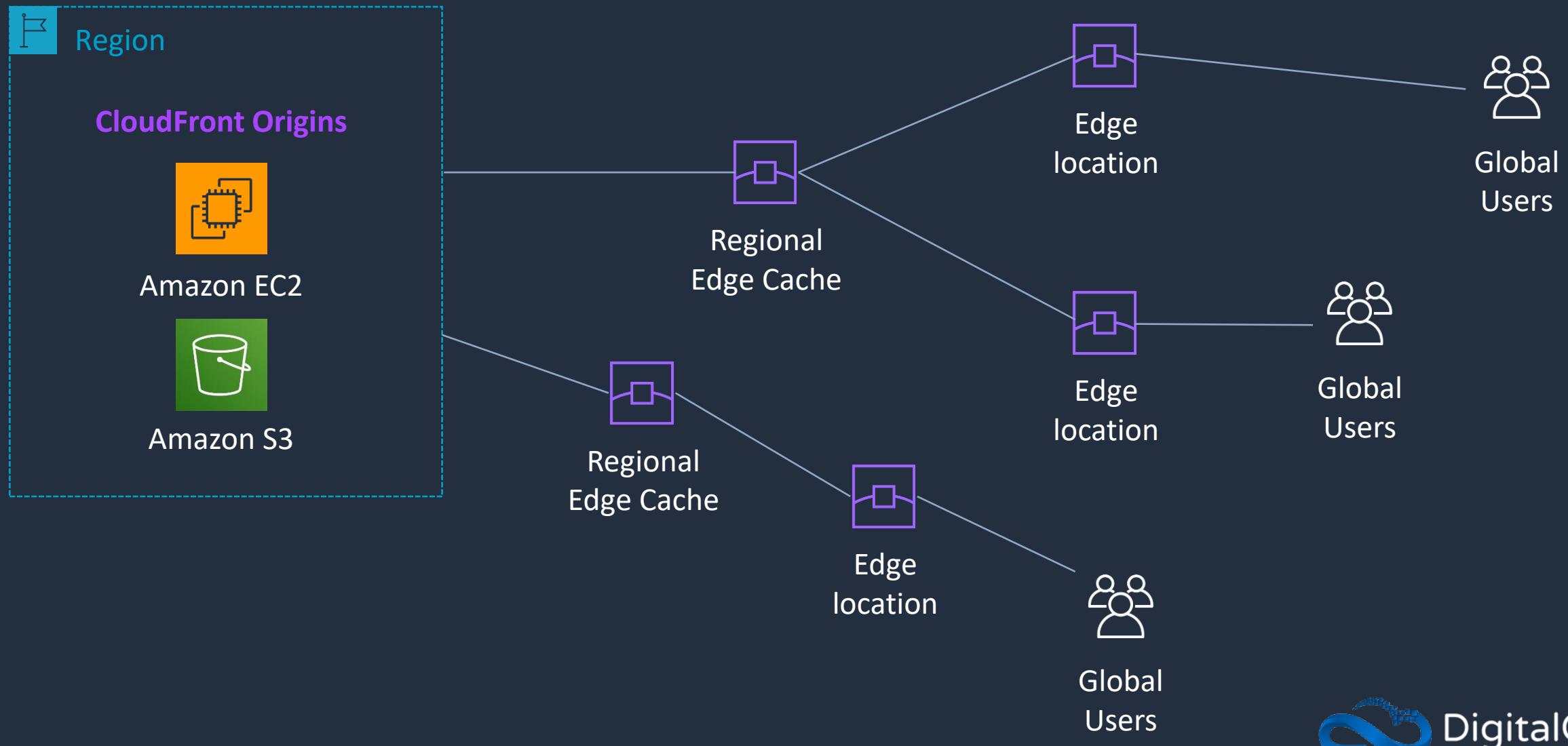


AWS Global Infrastructure



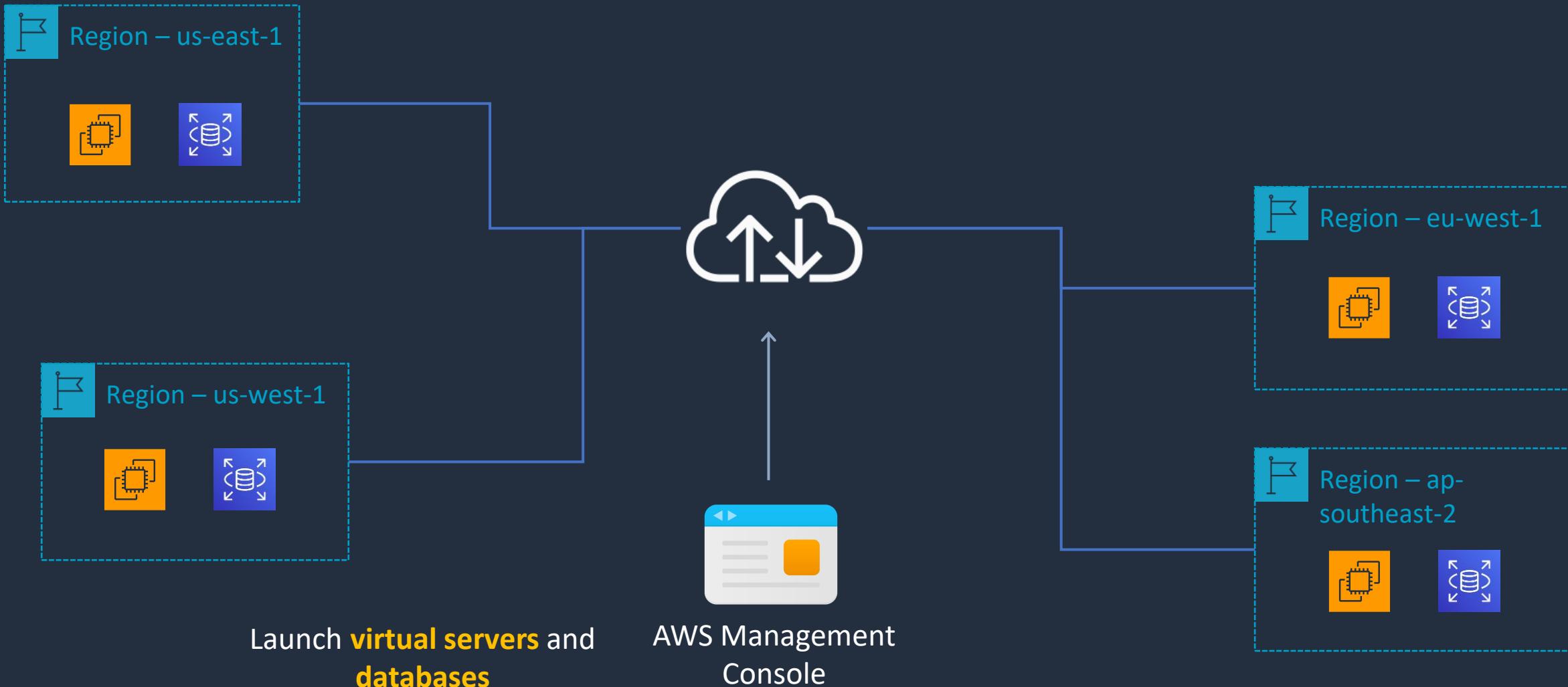


Amazon CloudFront





Deploying Services Globally



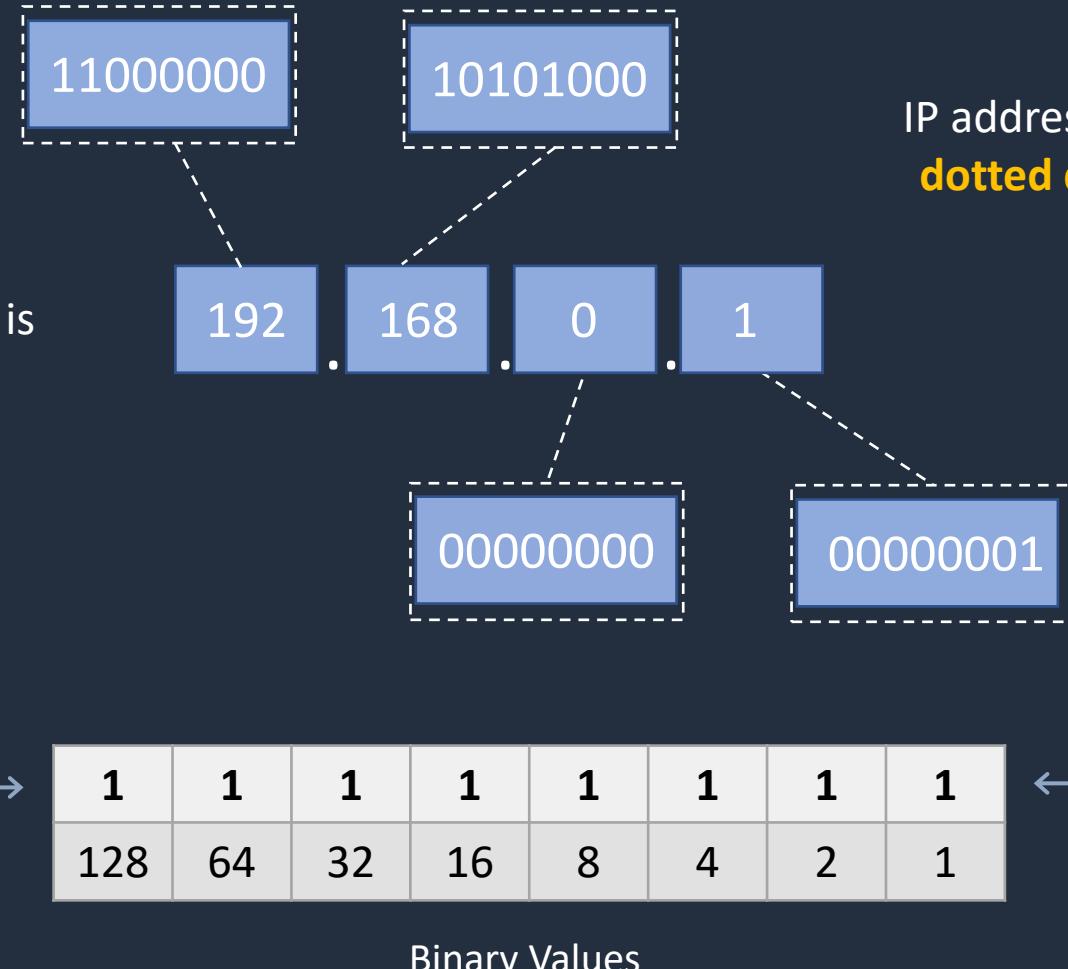
IP Addressing





Structure of an IPv4 Address

Each part of the address is a **binary octet**

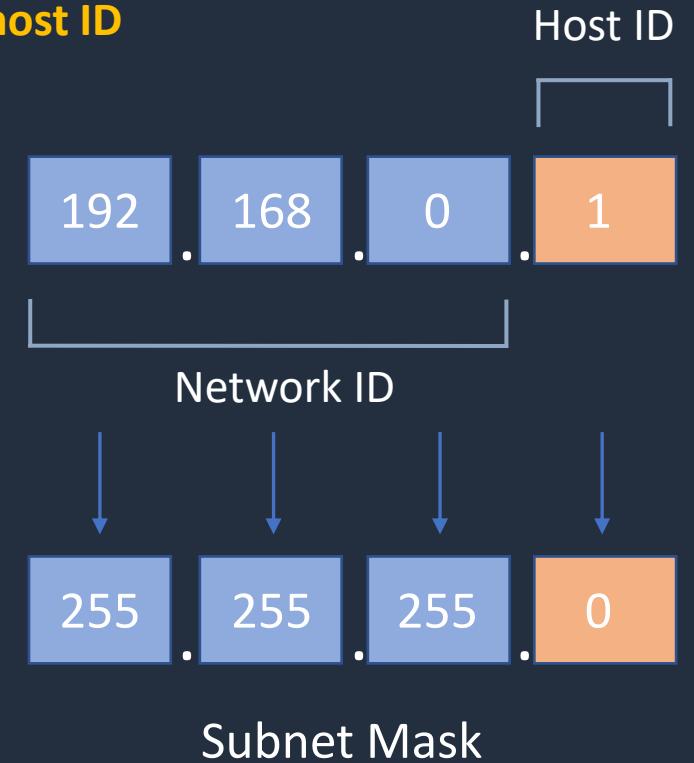


IP addresses are written in
dotted decimal notation



Networks and Hosts

An IPv4 address has a **network** and **host ID**



The **subnet mask** is used to define the **network** and **host ID**



Networks and Hosts

Network

192	168	0	0
-----	-----	---	---

Subnet Mask

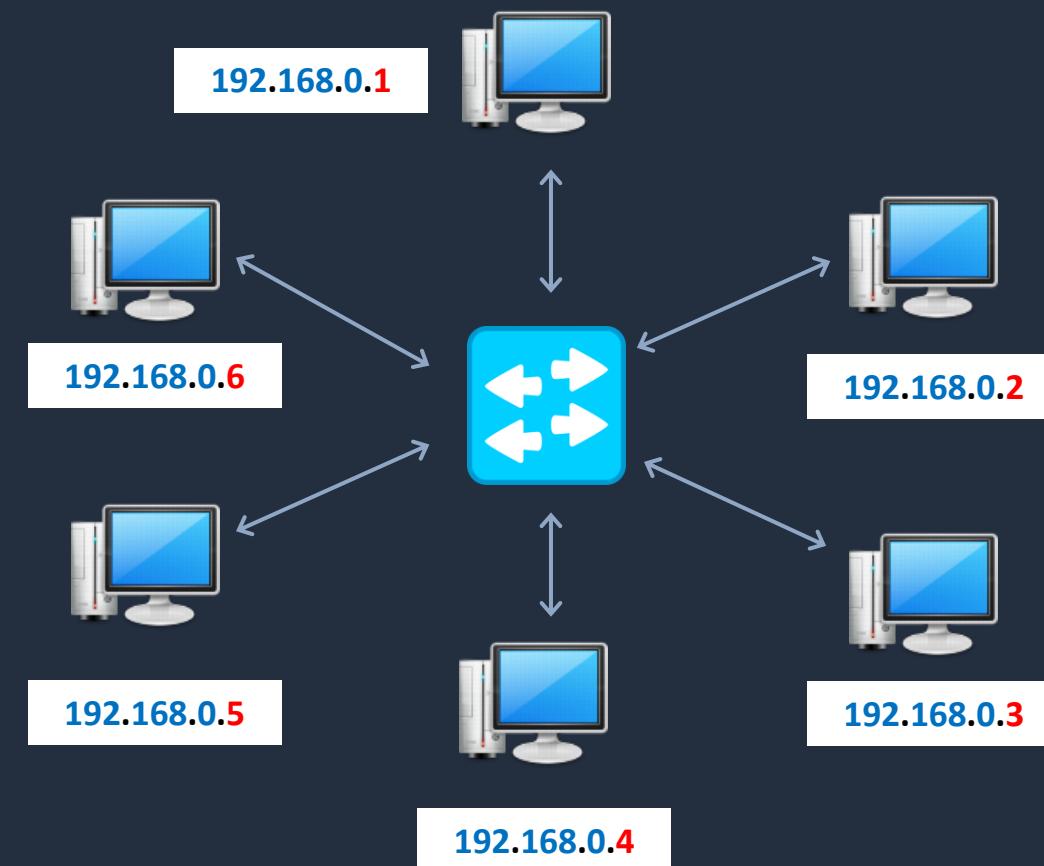
255	255	255	0
-----	-----	-----	---

24 bits

192.168.0.0/24

A **network** and **subnet mask** can also be written in this format

All computers share the same **network ID** and have a unique **host ID**





Private IP Address Ranges



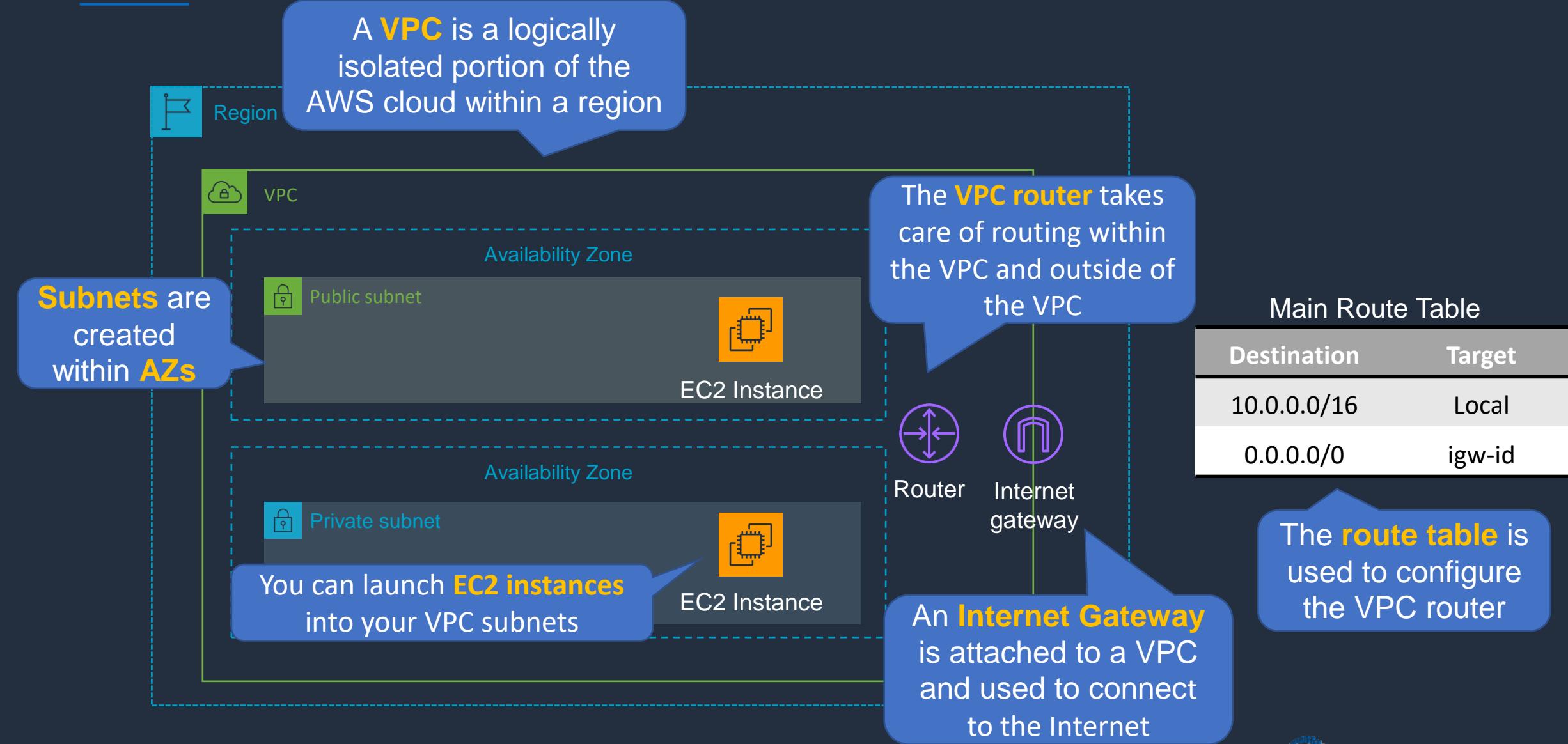
These addresses are reserved
for **private use** according to
IETF RFC-1918

Amazon VPC Overview





Amazon Virtual Private Cloud (VPC)





Amazon VPC



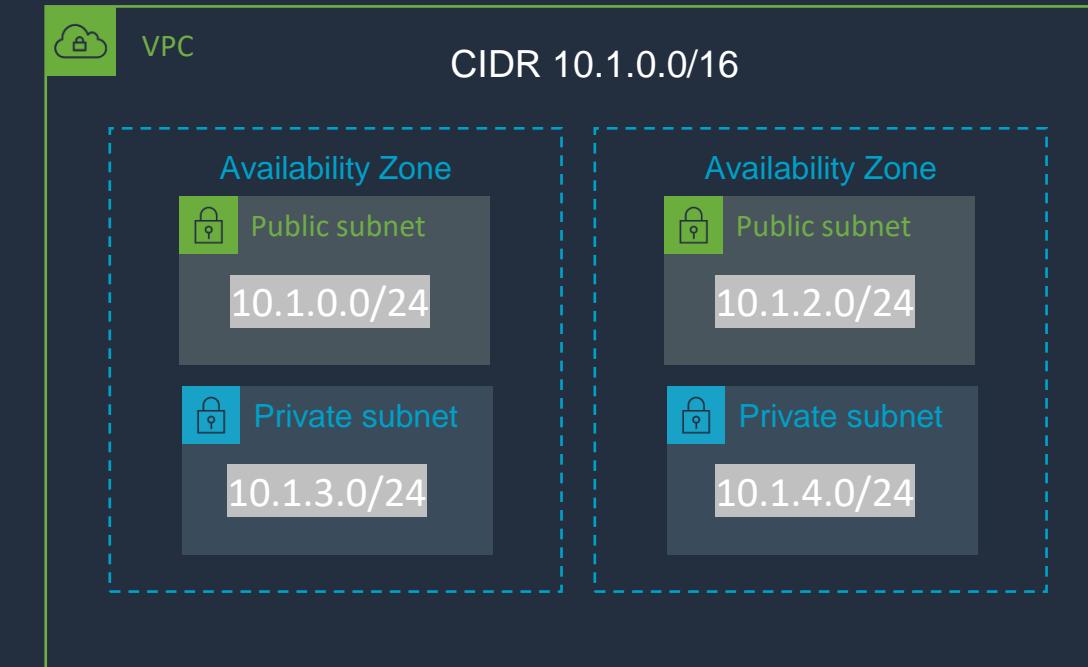
Region

Each **VPC** has a different block of IP addresses

CIDR stands for Classless
Interdomain Routing



Each subnet has a
block of **IP addresses**
from the CIDR block



You can create **multiple**
VPCs within each region



Amazon VPC Components

VPC Component	What it is
Virtual Private Cloud (VPC)	A logically isolated virtual network in the AWS cloud
Subnet	A segment of a VPC's IP address range where you can place groups of isolated resources
Internet Gateway/Egress-only Internet Gateway	The Amazon VPC side of a connection to the public Internet for IPv4/IPv6
Router	Routers interconnect subnets and direct traffic between Internet gateways, virtual private gateways, NAT gateways, and subnets
Peering Connection	Direct connection between two VPCs
VPC Endpoints	Private connection to public AWS services
NAT Instance	Enables Internet access for EC2 instances in private subnets managed by you
NAT Gateway	Enables Internet access for EC2 instances in private subnets (managed by AWS)
Virtual Private Gateway	The Amazon VPC side of a Virtual Private Network (VPN) connection
Customer Gateway	Customer side of a VPN connection
AWS Direct Connect	High speed, high bandwidth, private network connection from customer to aws
Security Group	Instance-level firewall
Network ACL	Subnet-level firewall



Amazon VPC Core Knowledge

- A virtual private cloud (VPC) is a virtual network dedicated to your AWS account
- Analogous to having your own data center inside AWS
- It is logically isolated from other virtual networks in the AWS Cloud
- Provides complete control over the virtual networking environment including selection of IP ranges, creation of subnets, and configuration of route tables and gateways
- You can launch your AWS resources, such as Amazon EC2 instances, into your VPC



Amazon VPC Core Knowledge

- When you create a VPC, you must specify a range of IPv4 addresses for the VPC in the form of a Classless Inter-Domain Routing (CIDR) block; for example, 10.0.0.0/16
- A VPC spans all the Availability Zones in the region
- You have full control over who has access to the AWS resources inside your VPC
- By default you can create up to 5 VPCs per region
- A default VPC is created in each region with a subnet in each AZ

Defining VPC CIDR Blocks





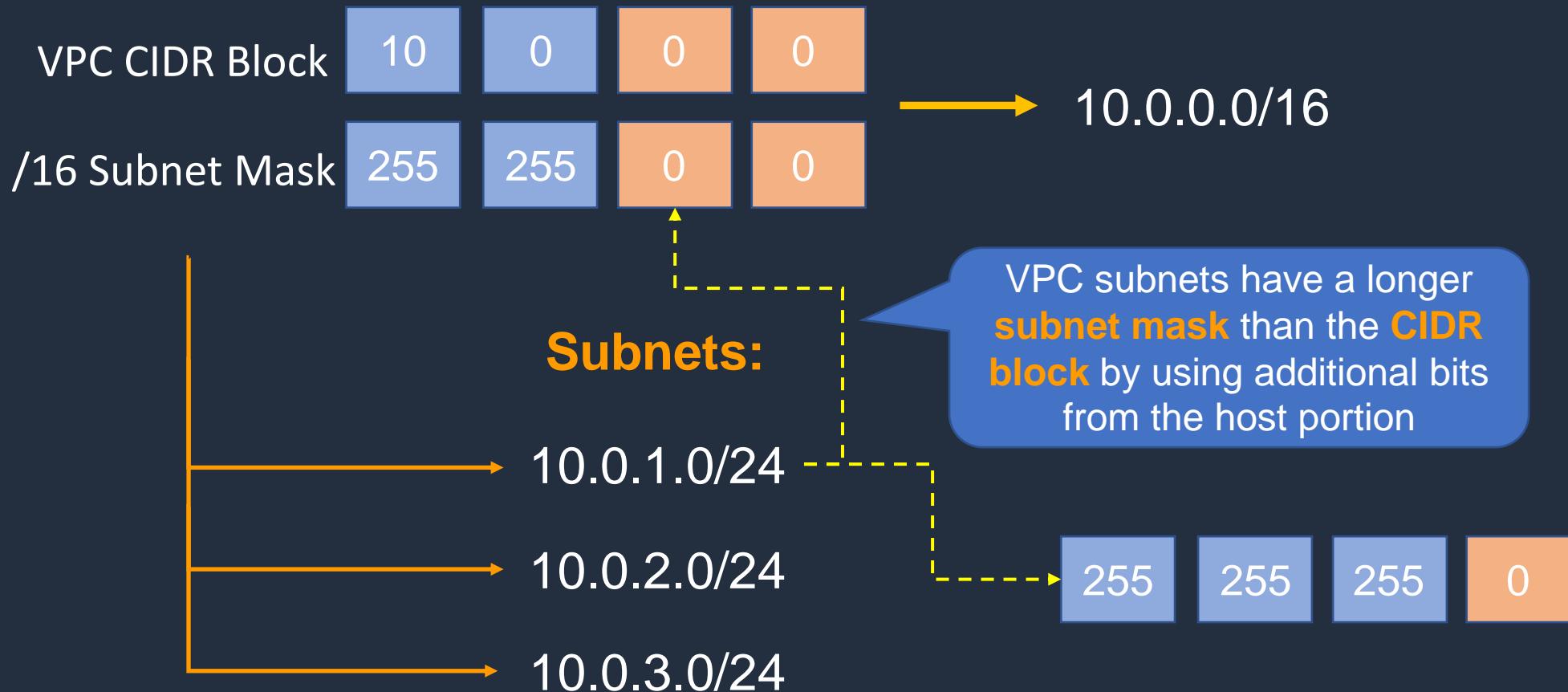
Rules and Guidelines

- CIDR block size can be between /16 and /28
- The CIDR block must not overlap with any existing CIDR block that's associated with the VPC
- You cannot increase or decrease the size of an existing CIDR block
- The first four and last IP address are not available for use
- AWS recommend you use CIDR blocks from the RFC 1918 ranges:

RFC 1918 Range	Example CIDR Block
10.0.0.0 - 10.255.255.255 (10/8 prefix)	Your VPC must be /16 or smaller, for example, 10.0.0.0/16
172.16.0.0 - 172.31.255.255 (172.16/12 prefix)	Your VPC must be /16 or smaller, for example, 172.31.0.0/16
192.168.0.0 - 192.168.255.255 (192.168/16 prefix)	Your VPC can be smaller, for example 192.168.0.0/20



VPC CIDR Blocks and Subnets



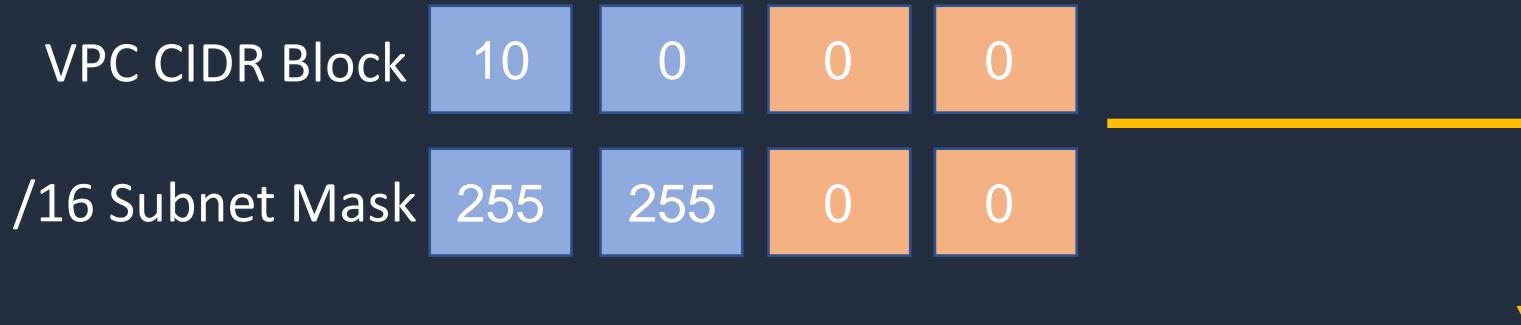


Additional Considerations

- Ensure you have enough networks and hosts
- **Bigger CIDR blocks** are typically better (more flexibility)
- **Smaller subnets** are OK for most use cases
- Consider deploying application tiers per subnet
- Split your HA resources across subnets in different AZs
- VPC Peering requires non-overlapping CIDR blocks
 - This is across all VPCs in all Regions / accounts you want to connect
- **Avoid overlapping CIDR blocks** as much as possible!



Example VPC CIDR Block and Subnets



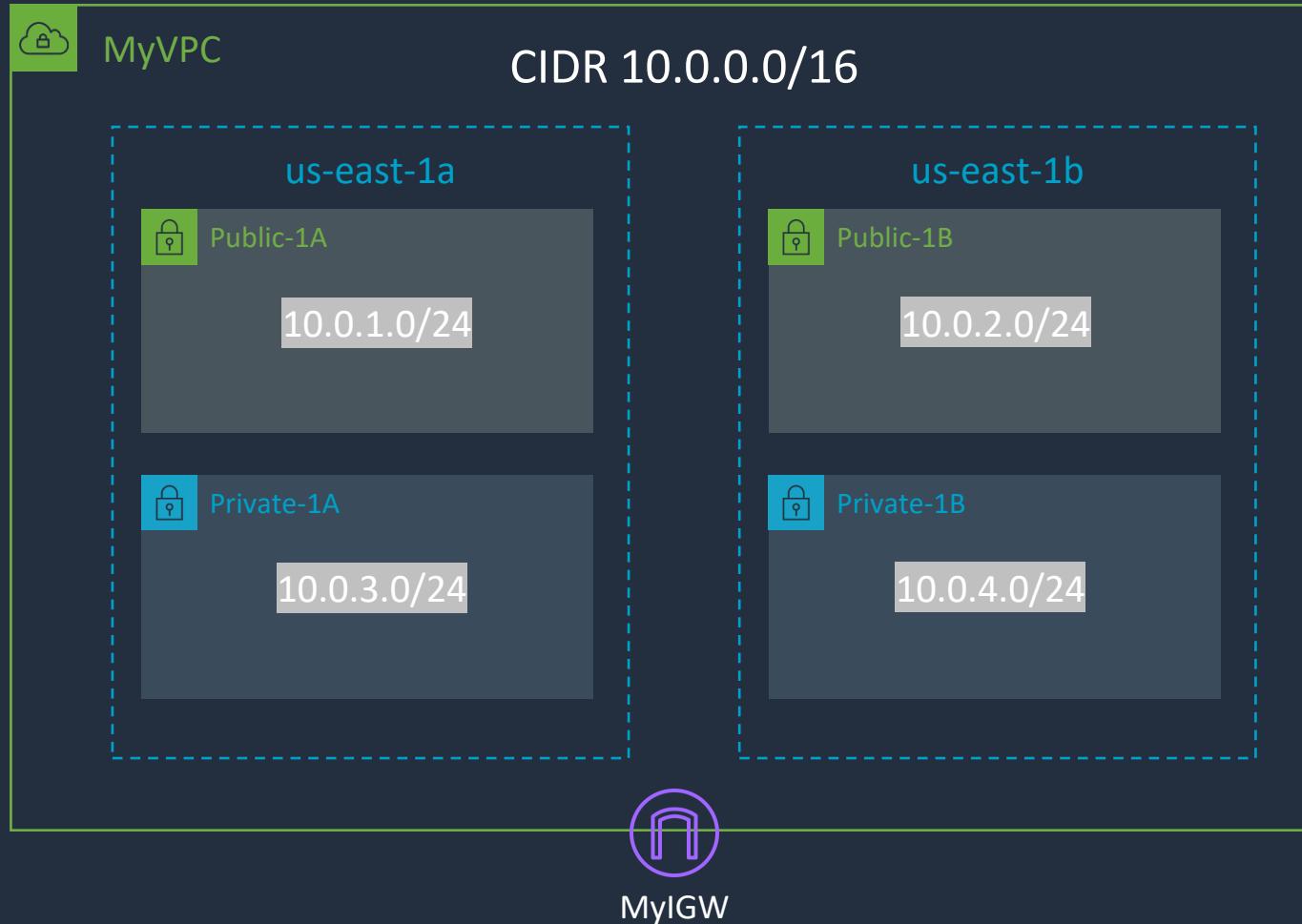
Subnet Name	IPv4 CIDR block	Availability Zone	Route Table	Auto-assign Public IP v4
private-1a	10.0.0.0/24	us-east-1a	Private-RT	No
private-1b	10.0.1.0/24	us-east-1b	Private-RT	No
private-1c	10.0.2.0/24	us-east-1c	Private-RT	No
public-1a	10.0.3.0/24	us-east-1a	MAIN	Yes
public-1b	10.0.4.0/24	us-east-1b	MAIN	Yes
public-1c	10.0.5.0/24	us-east-1c	MAIN	Yes

Create a Custom VPC





Custom VPC



Main Route Table

Destination	Target
10.0.0.0/16	Local
0.0.0.0/0	igw-id

Private-RT Route Table

Destination	Target
10.0.0.0/16	Local

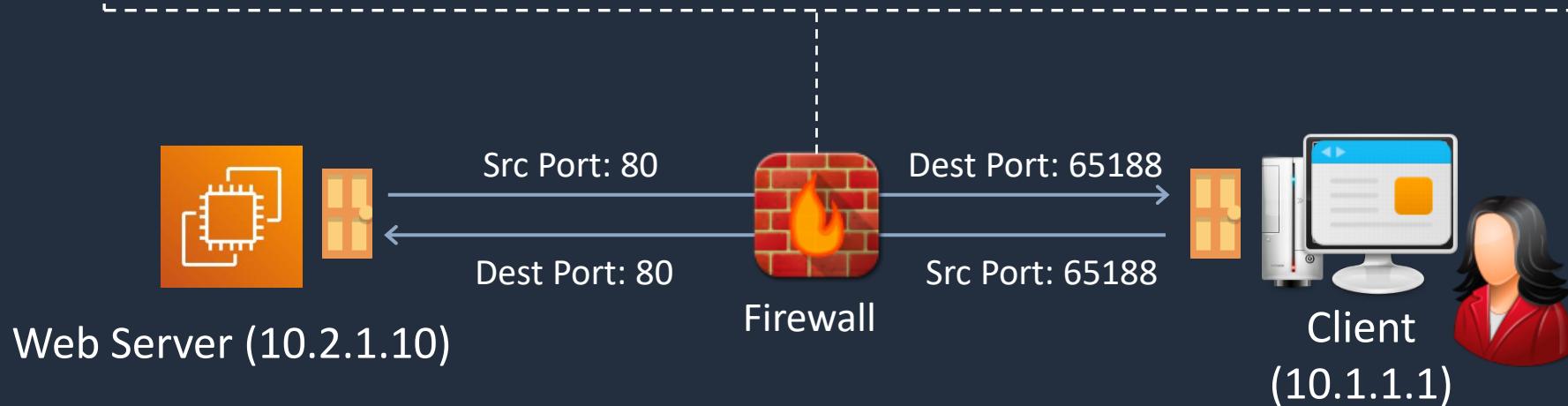
Security Groups and Network ACLs





Stateful vs Stateless Firewalls

PROTOCOL	SOURCE IP	DESTINATION IP	SOURCE PORT	DESTINATION PORT
HTTP	10.1.1.1	10.2.1.10	65188	80
HTTP	10.2.1.10	10.1.1.1	80	65188

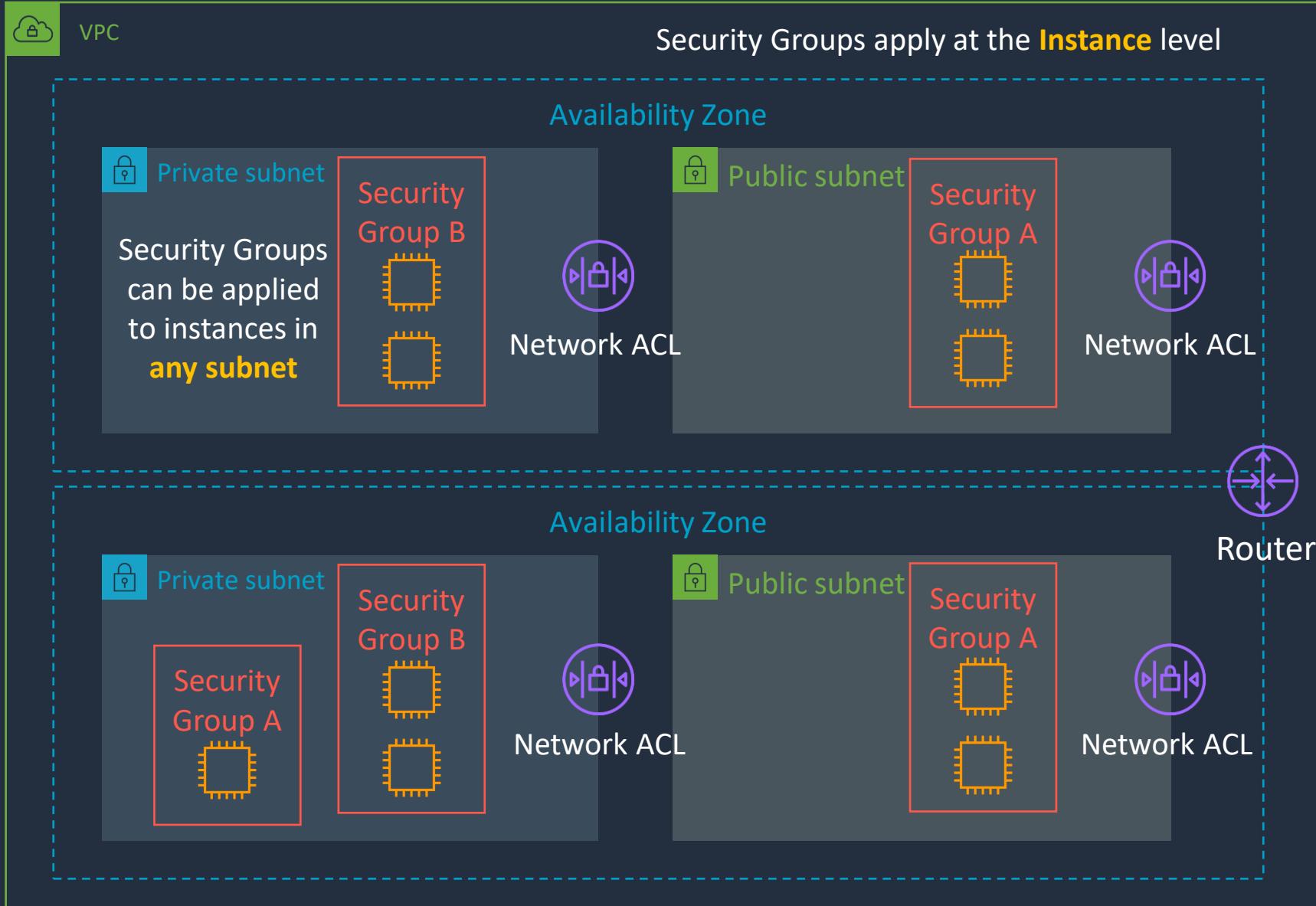


A **stateful** firewall
allows the return
traffic automatically

A **stateless** firewall
checks for an allow
rule for **both**
connections



Security Groups and Network ACLs





Security Groups & Network Access Control Lists (NACLs)

Security Group	Network ACL
Operates at the instance (interface) level	Operates at the subnet level
Supports allow rules only	Supports allow and deny rules
Stateful	Stateless
Evaluates all rules	Processes rules in order
Applies to an instance only if associated with a group	Automatically applies to all instances in the subnets its associated with

Using Security Groups and NACLs

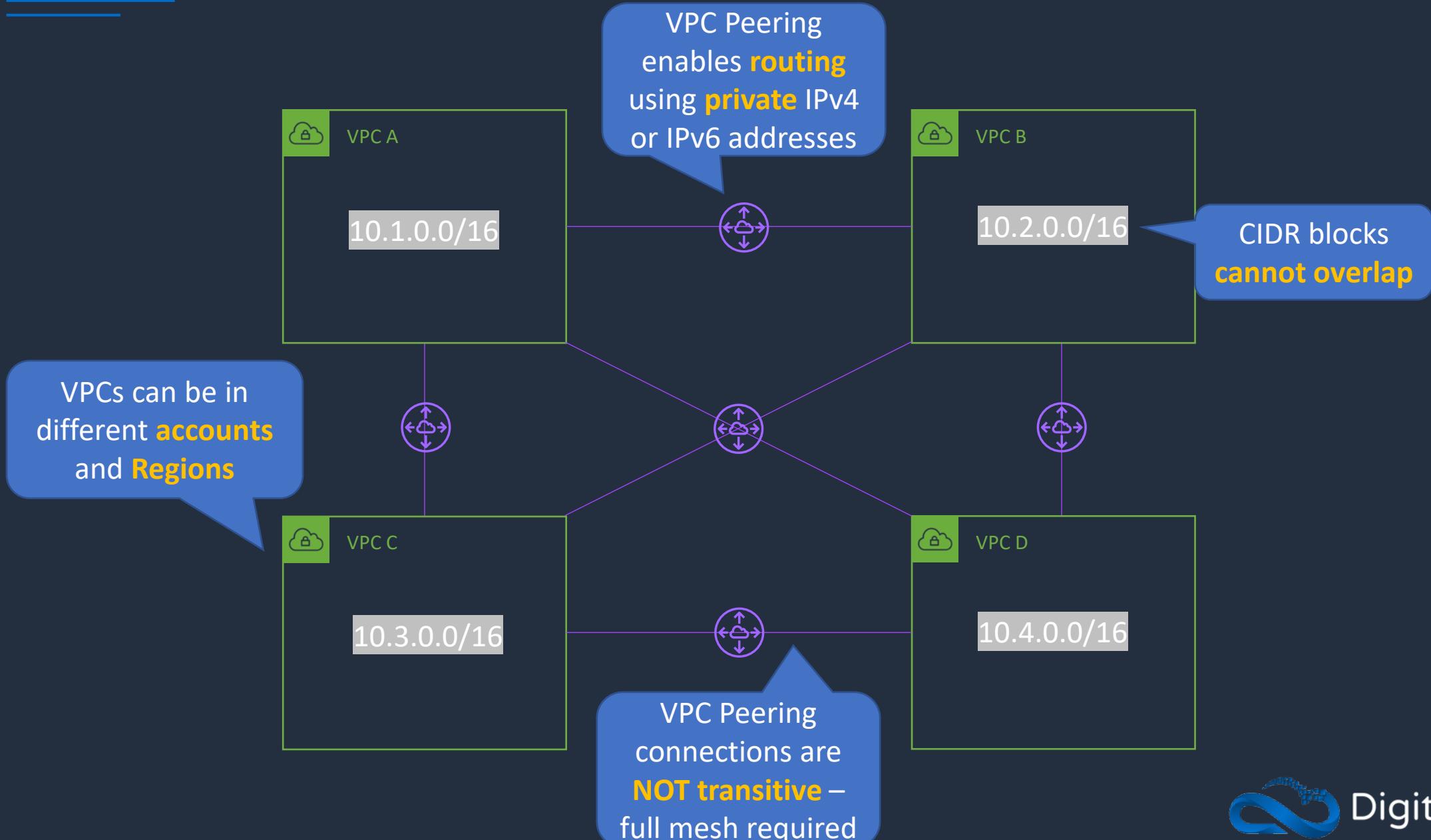


VPC Peering





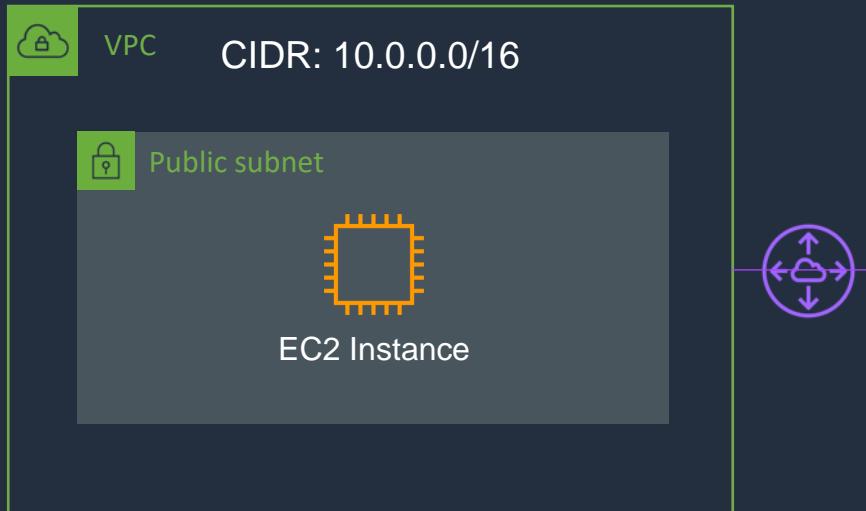
VPC Peering



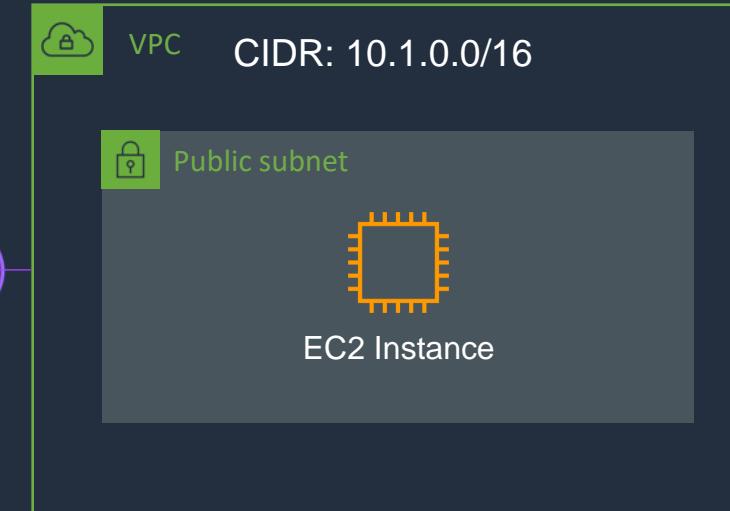


VPC Peering

Management Account



Production Account



Security group (VPCPEER-MGMT)

Protocol	Port	Source
ICMP	All	10.1.0.0/16
TCP	22	0.0.0.0/0

Security group (VPCPEER-PROD)

Protocol	Port	Source
ICMP	All	10.0.0.0/16
TCP	22	0.0.0.0/0

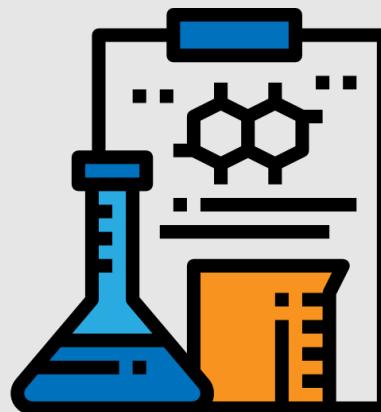
Route Table

Destination	Target
10.1.0.0/16	peering-id

Route Table

Destination	Target
10.0.0.0/16	peering-id

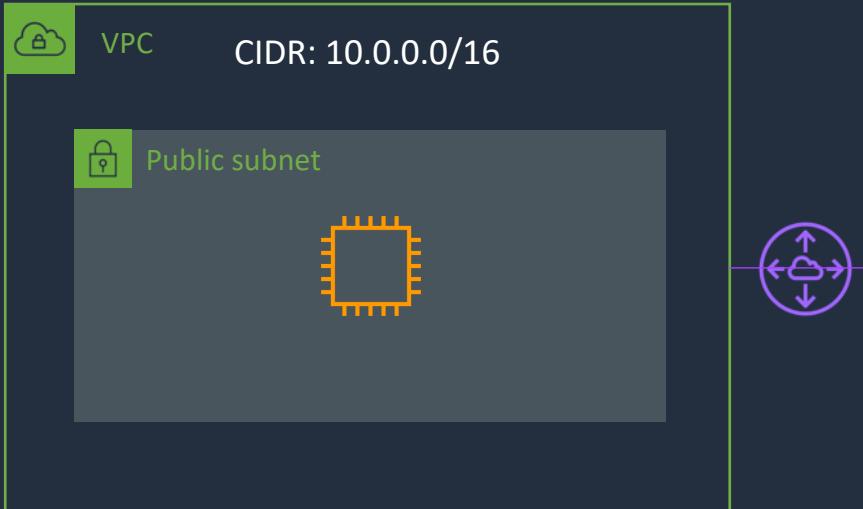
Configure VPC Peering



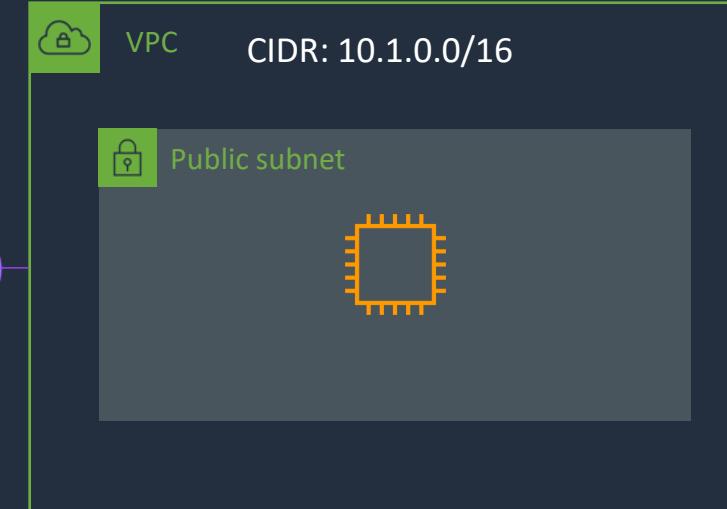


VPC Peering

US-EAST-1



US-WEST-1



Security group

Protocol	Port	Source
ICMP	All	10.1.0.0/16
TCP	22	0.0.0.0/0

Security group

Protocol	Port	Source
ICMP	All	10.0.0.0/16
TCP	22	0.0.0.0/0

Route Table

Destination	Target
10.1.0.0/16	peering-id

Route Table

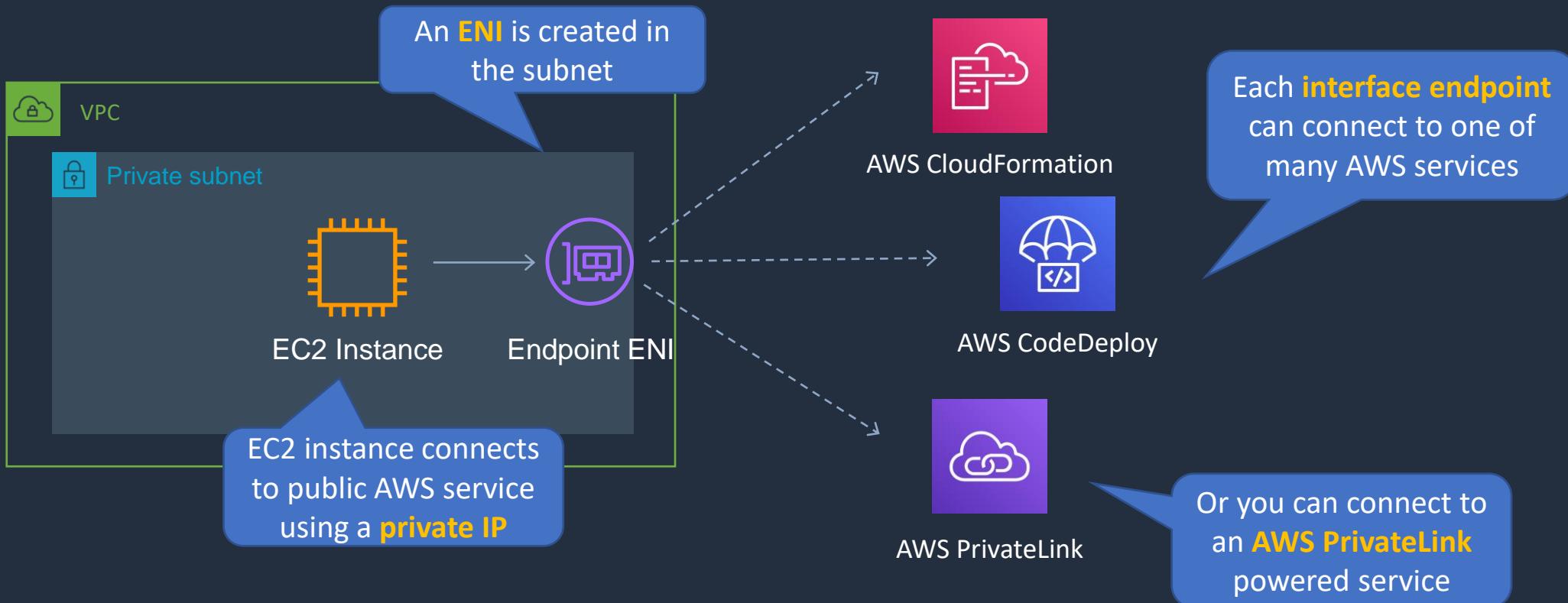
Destination	Target
10.0.0.0/16	peering-id

VPC Endpoints



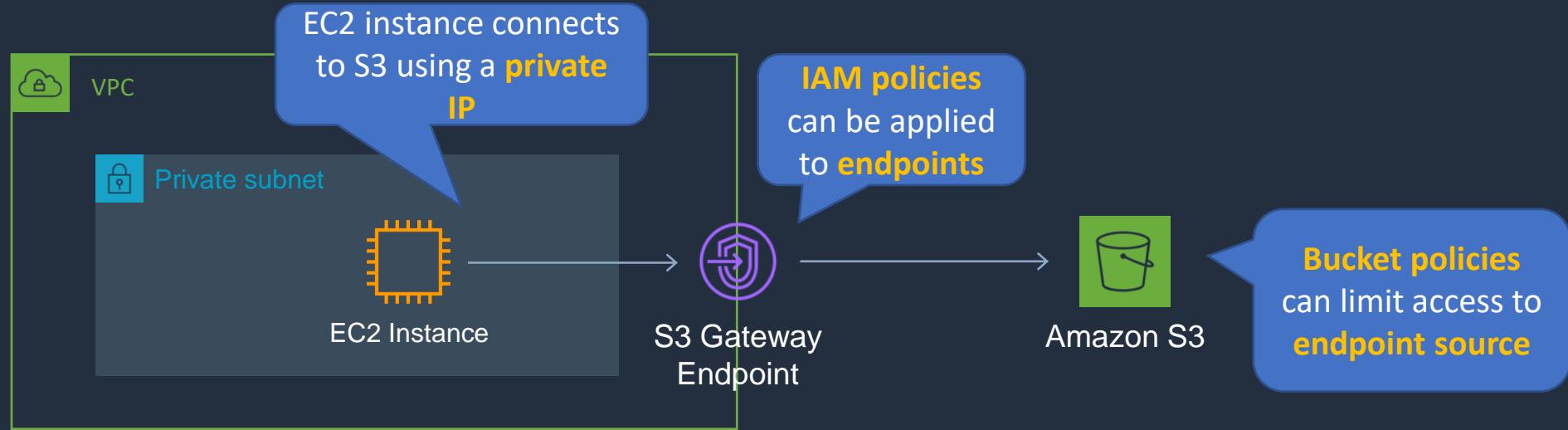


VPC Interface Endpoints





VPC Gateway Endpoints



Route Table

Destination	Target
<code>pl-6ca54005 (com.amazonaws.ap-southeast-2.s3, 54.231.248.0/22, 54.231.252.0/24, 52.95.128.0/21)</code>	<code>vpce-ID</code>

A **route table** entry is required with the prefix list for S3 and the **gateway ID**

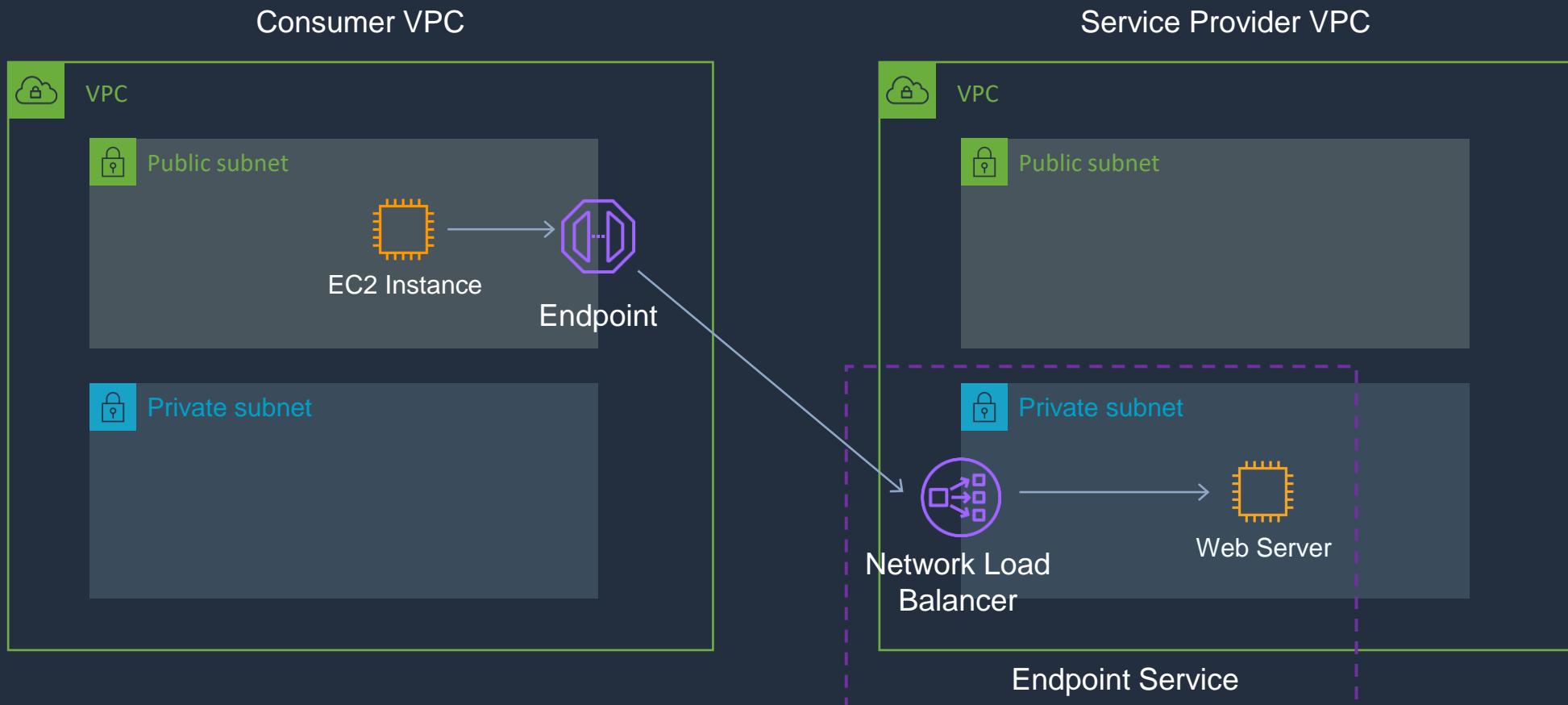


VPC Endpoints

	Interface Endpoint	Gateway Endpoint
What	Elastic Network Interface with a Private IP	A gateway that is a target for a specific route
How	Uses DNS entries to redirect traffic	Uses prefix lists in the route table to redirect traffic
Which services	API Gateway, CloudFormation, CloudWatch etc.	Amazon S3, DynamoDB
Security	Security Groups	VPC Endpoint Policies



Service Provider Model



Create VPC Endpoint





VPC Gateway Endpoints

EC2 instance connects to S3 using a **private IP**

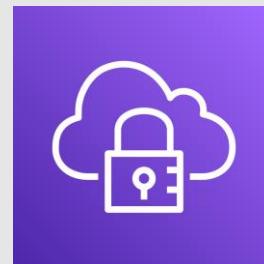


Route Table

Destination	Target
<code>pl-6ca54005 (com.amazonaws.ap-southeast-2.s3, 54.231.248.0/22, 54.231.252.0/24, 52.95.128.0/21)</code>	<code>vpce-ID</code>

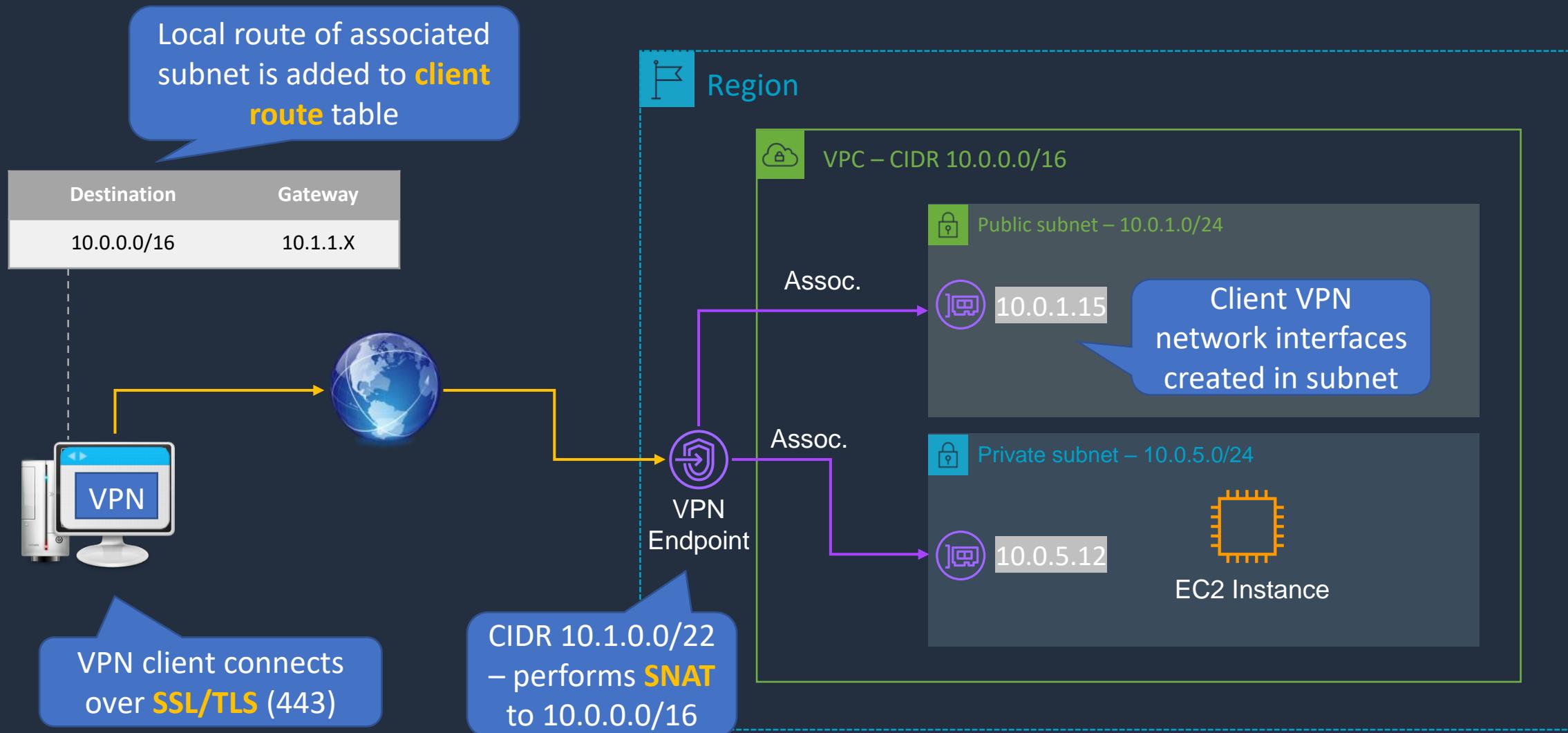
A **route table** entry is required with the prefix list for S3 and the **gateway ID**

AWS Client VPN

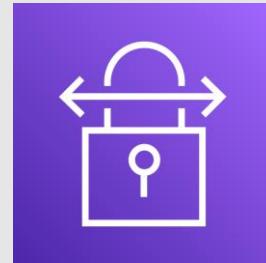




AWS Client VPN

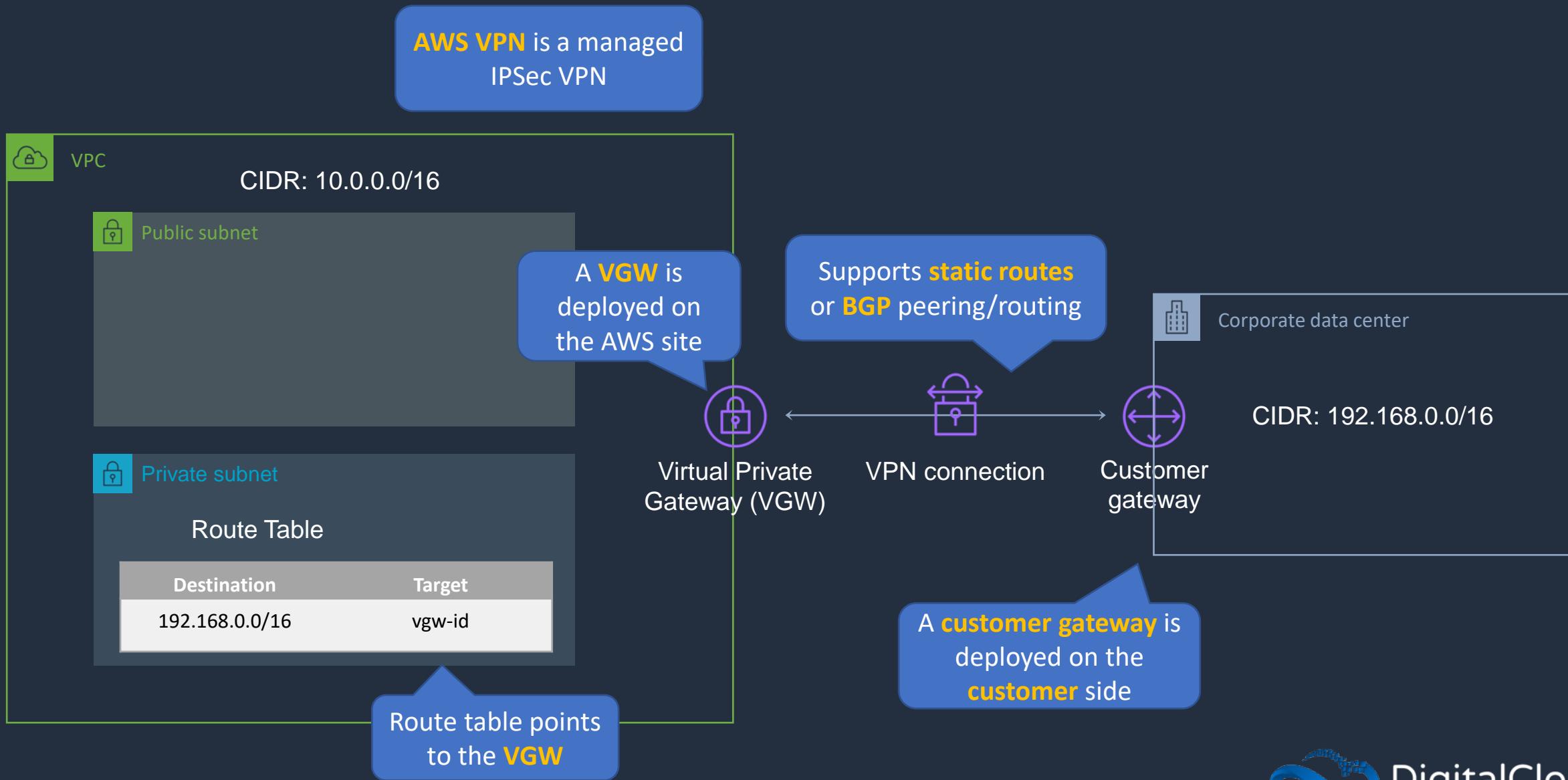


AWS Site-to-Site VPN





AWS Site-to-Site VPN

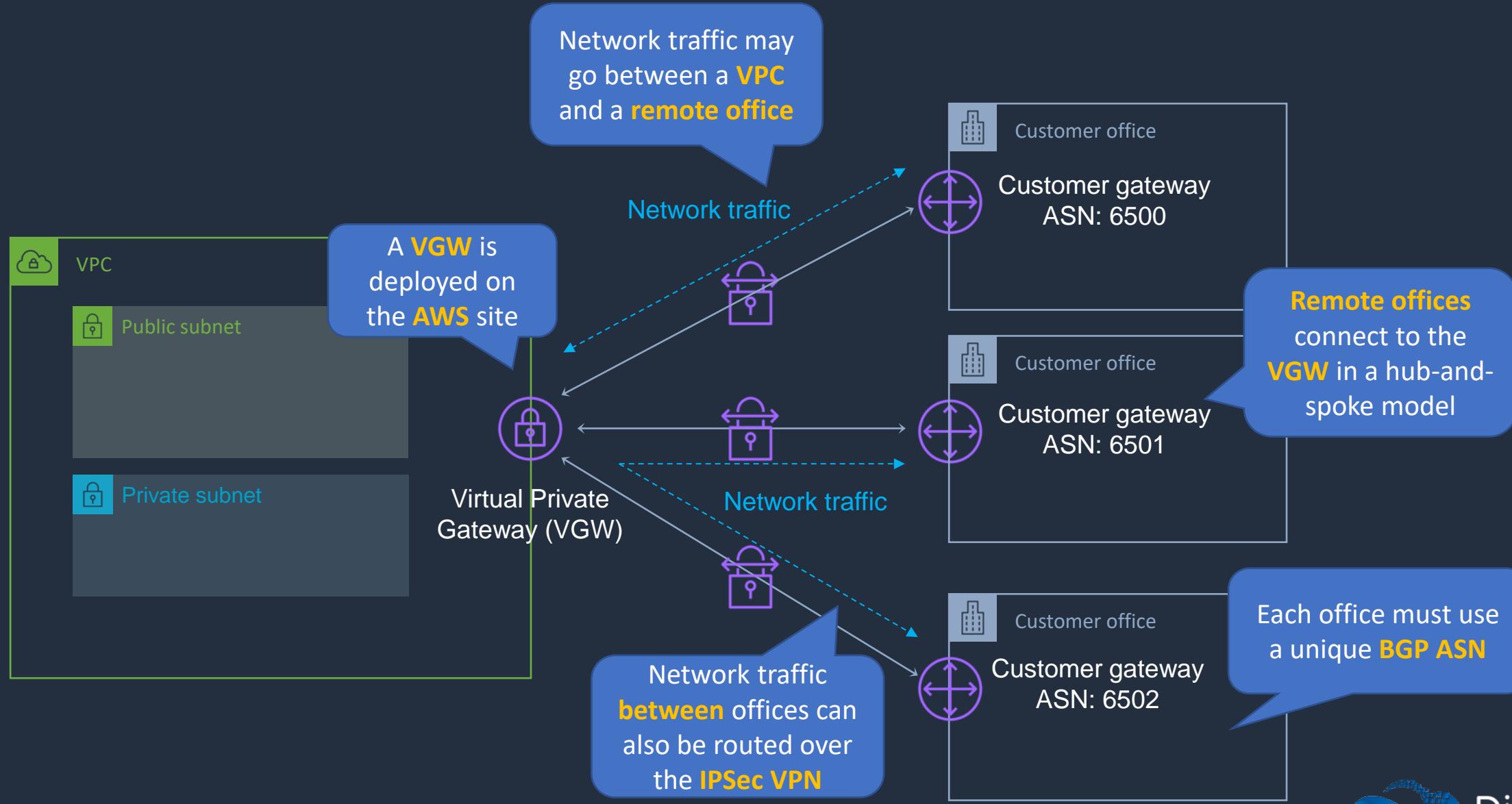


AWS VPN CloudHub





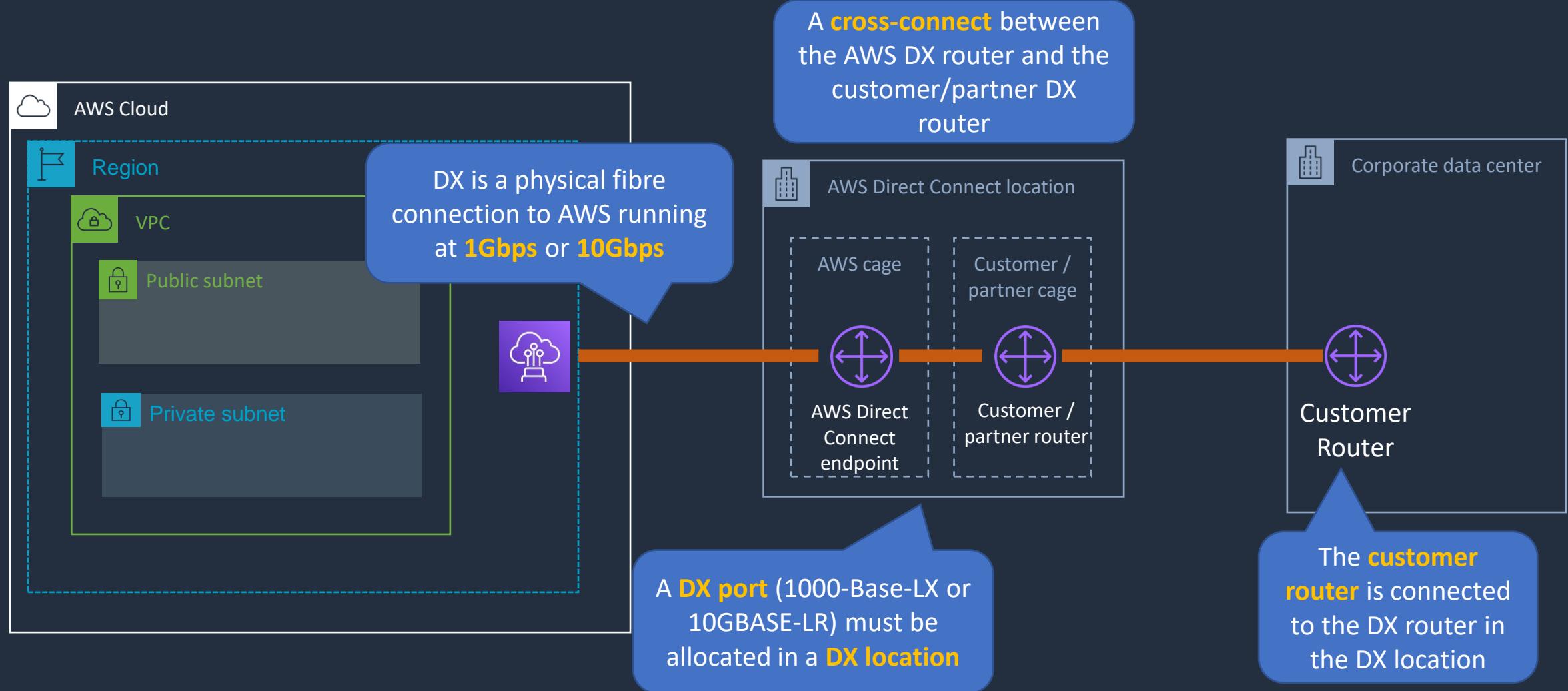
AWS VPN CloudHub



AWS Direct Connect (DX)



AWS Direct Connect (DX)

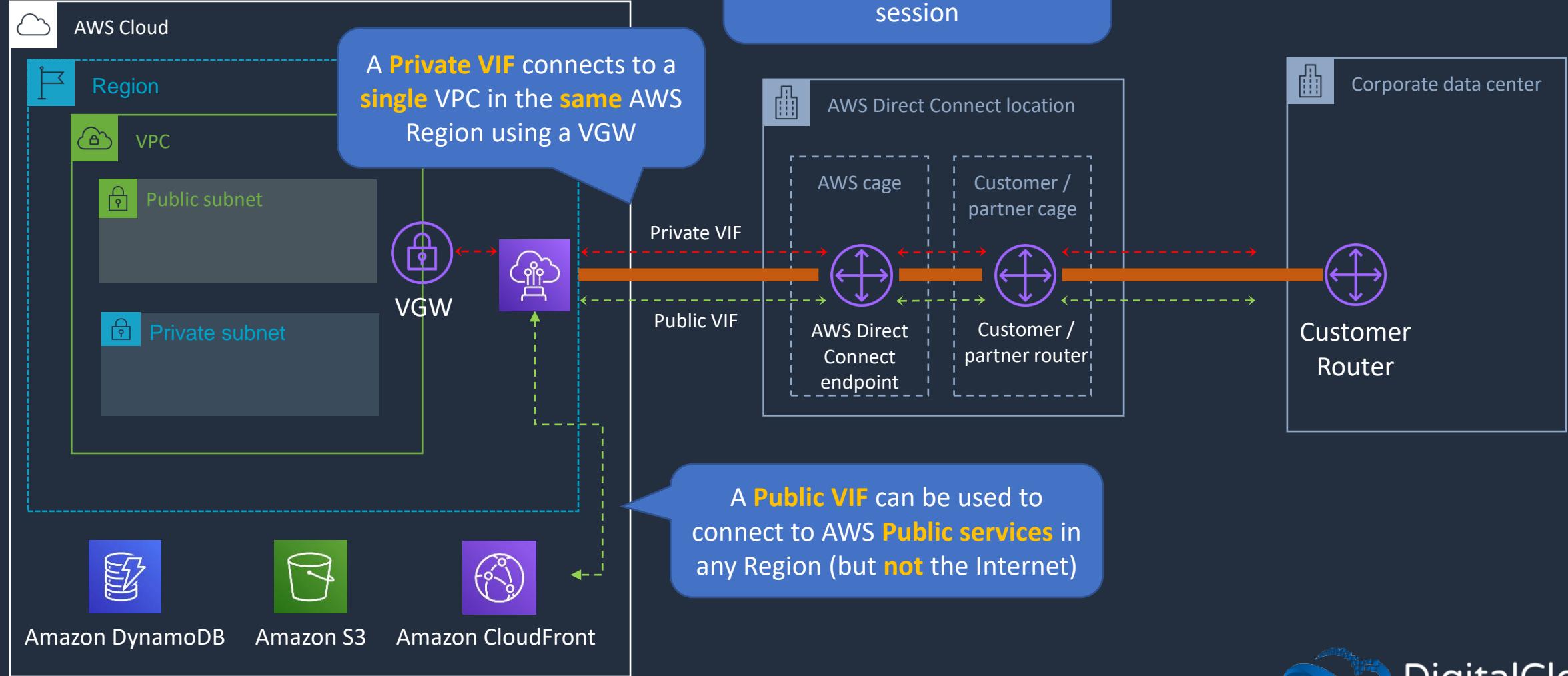




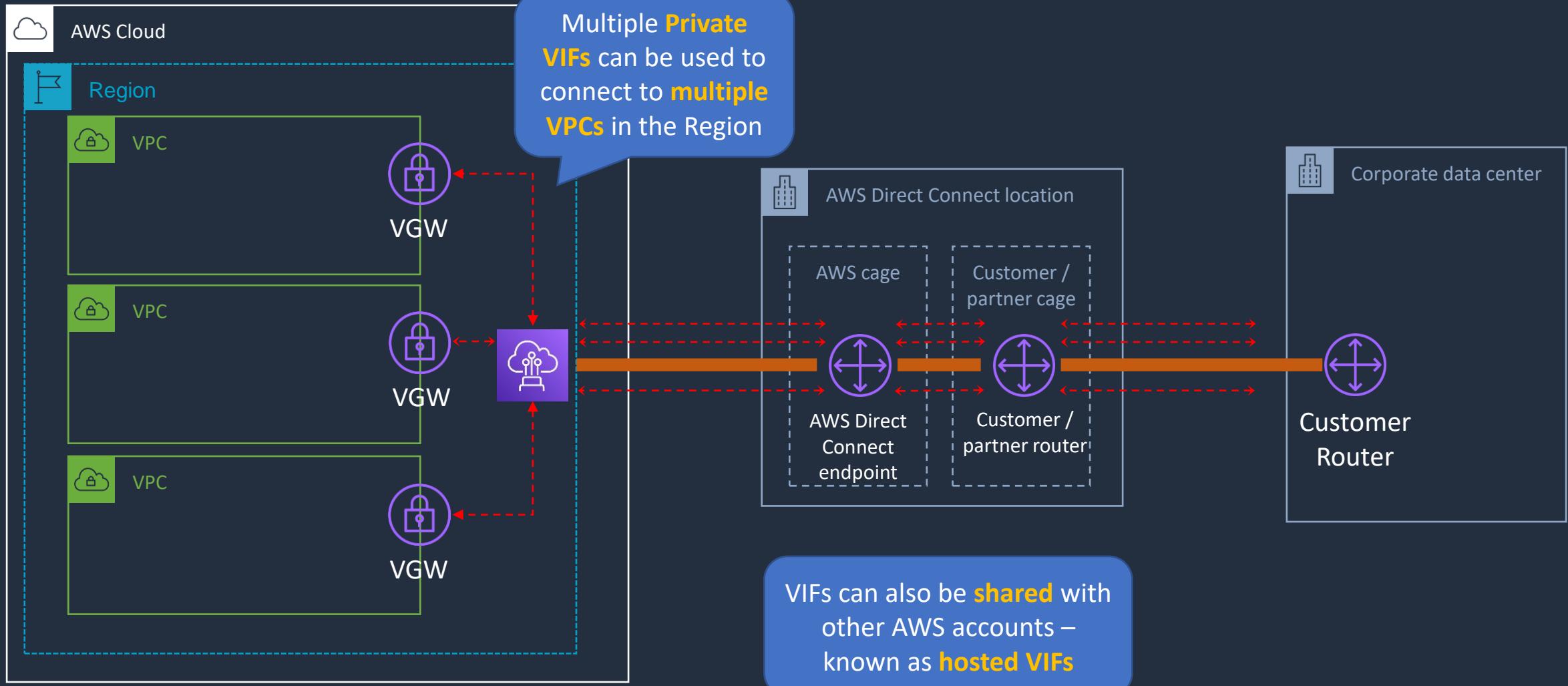
AWS Direct Connect Benefits

- **Private** connectivity between AWS and your data center / office
- Consistent network experience – increased **speed/latency** & **bandwidth/throughput**
- Lower costs for organizations that transfer **large** volumes of data

AWS Direct Connect (DX)



AWS Direct Connect (DX)





AWS Direct Connect (DX)

- Speeds from 50Mbps to 500Mbps can also be accessed via an APN partner
- 100 Gbps is now featuring in select locations
- DX Connections are **NOT** encrypted!
- Use an **IPSec S2S VPN** connection over a VIF to add encryption in transit

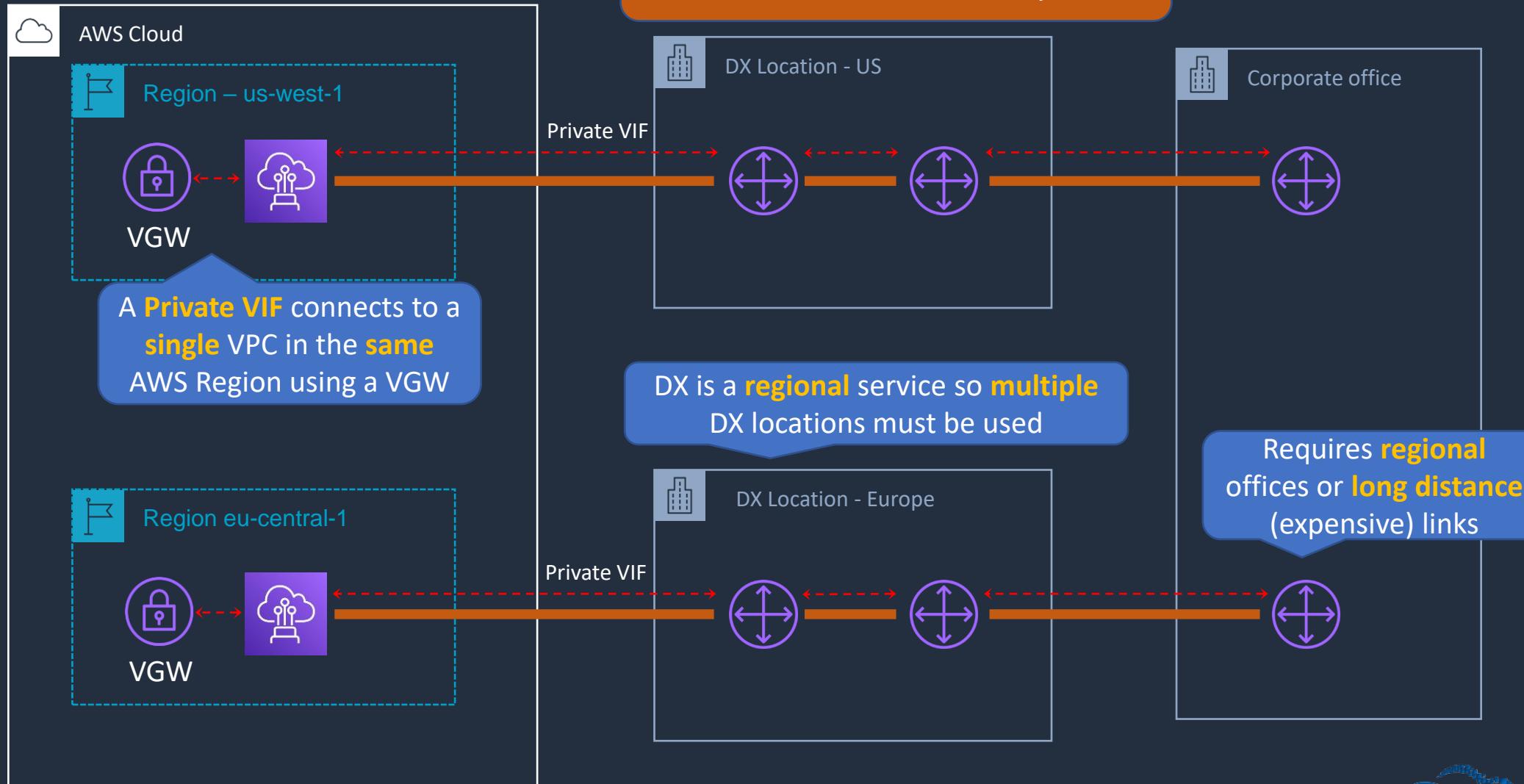
AWS Direct Connect Gateway





Direct Connect - Multiple Regions

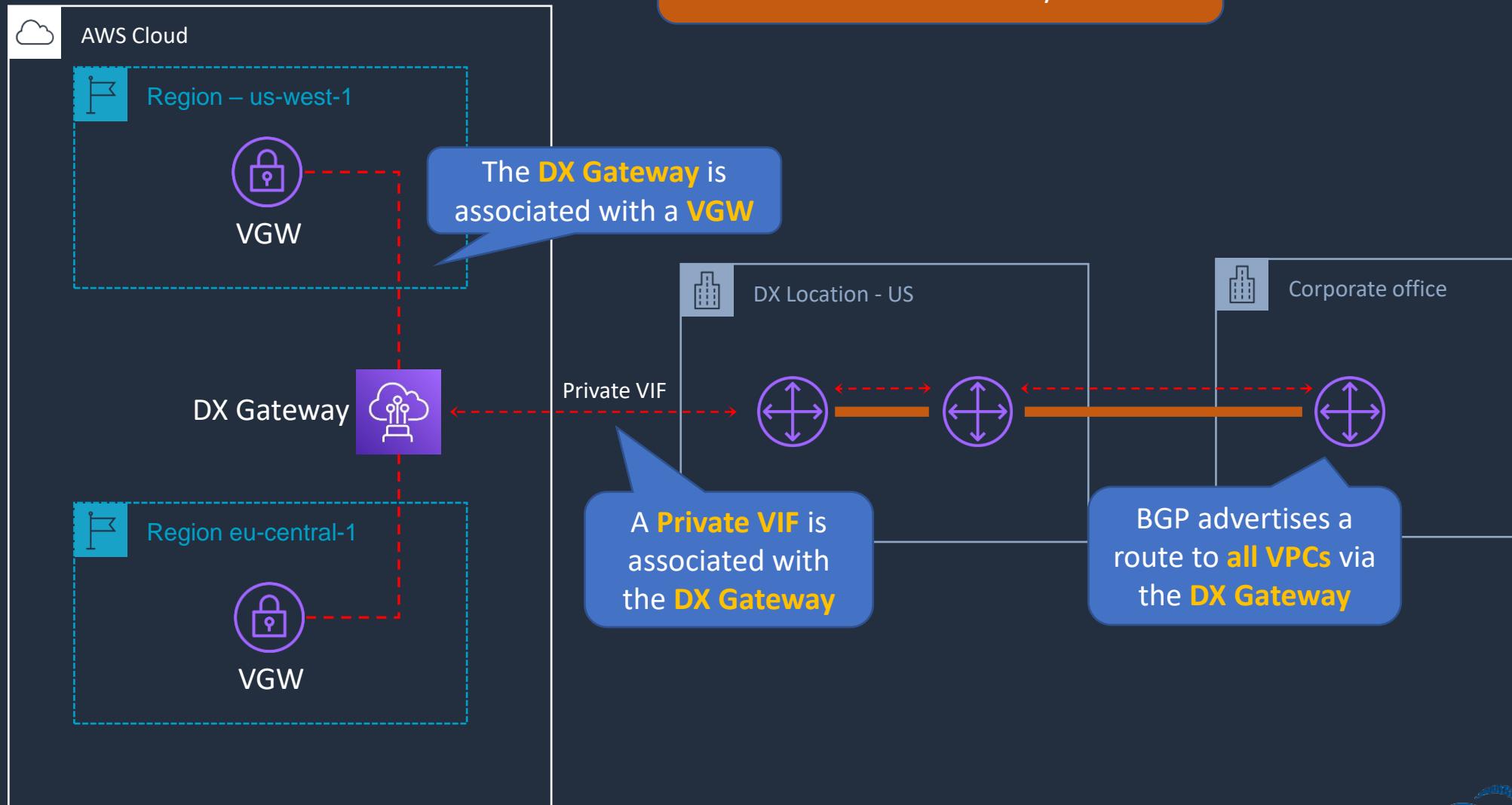
Example architecture **without** AWS Direct Connect Gateway





Direct Connect - Multiple Regions

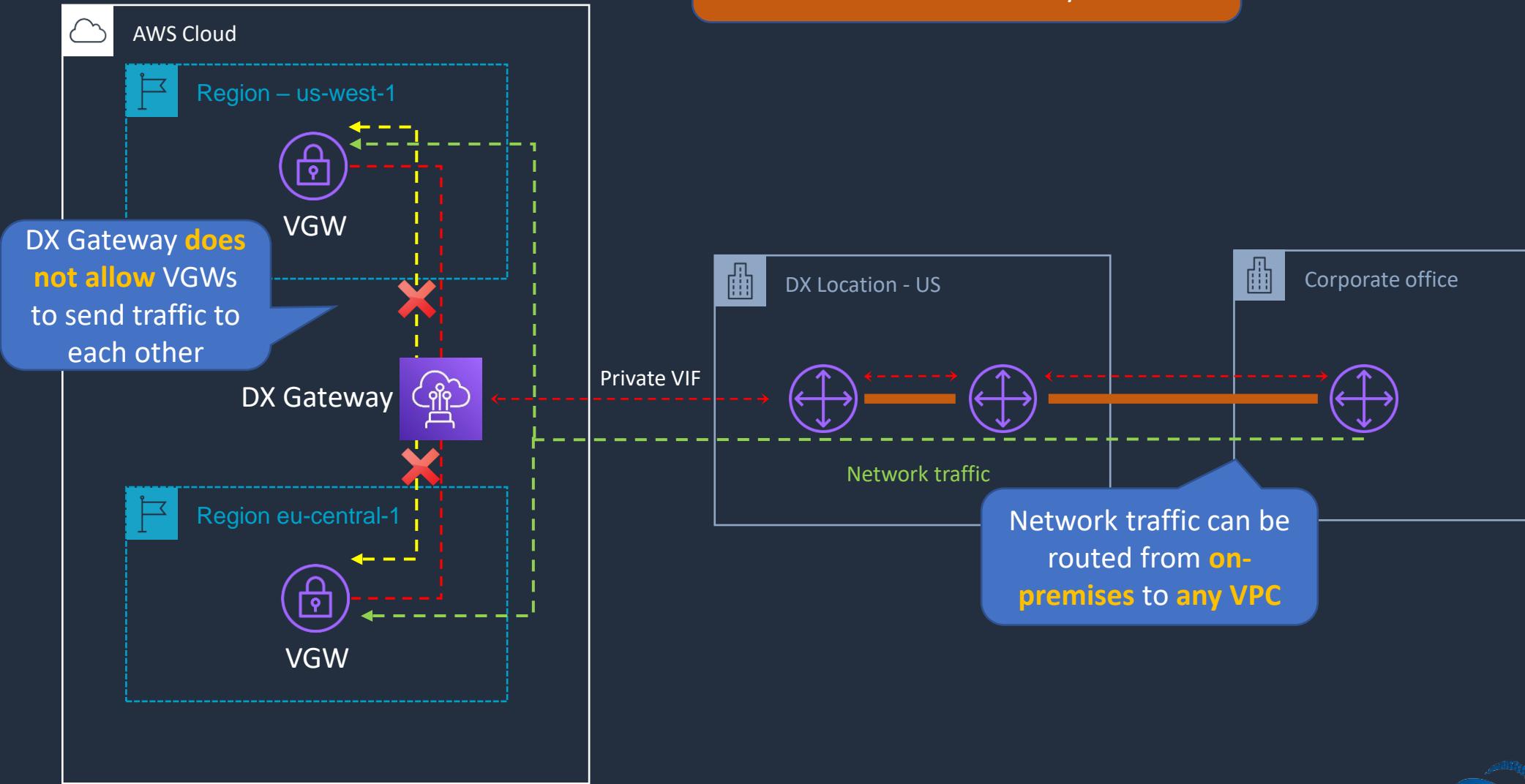
Example architecture **with** AWS Direct Connect Gateway



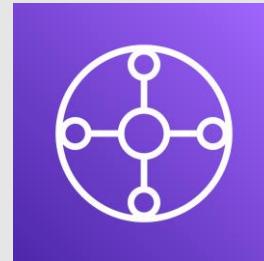


Direct Connect - Multiple Regions

Example architecture **with** AWS Direct Connect Gateway

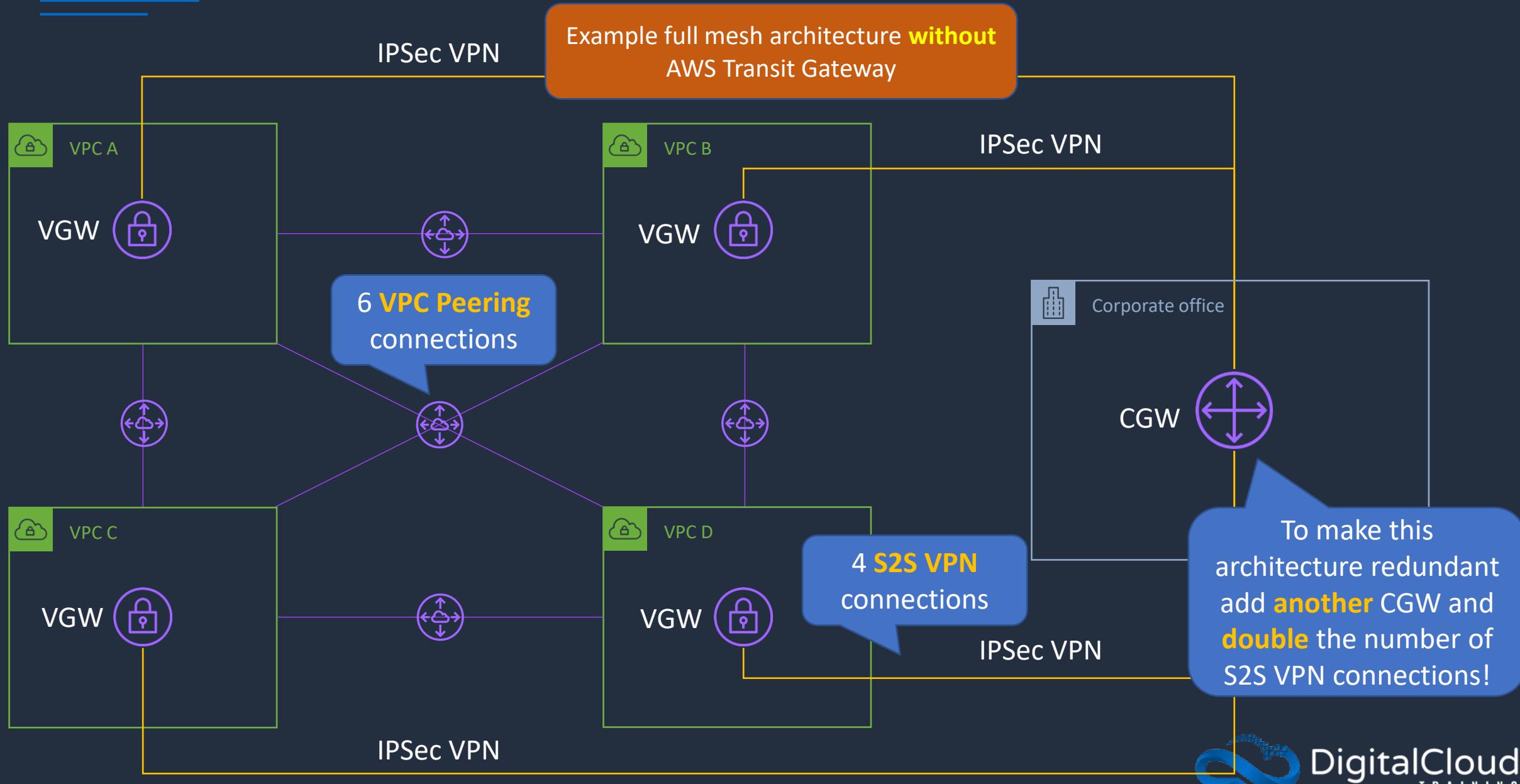


AWS Transit Gateway



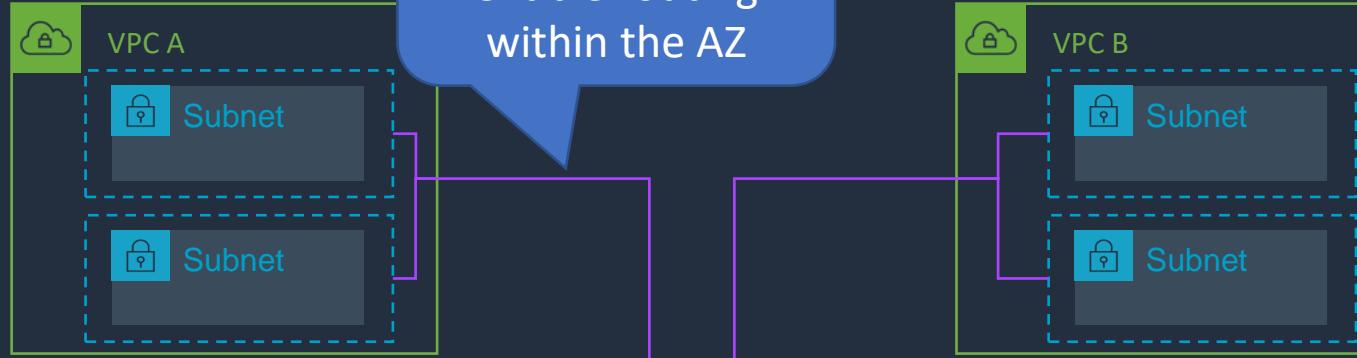


AWS Transit Gateway

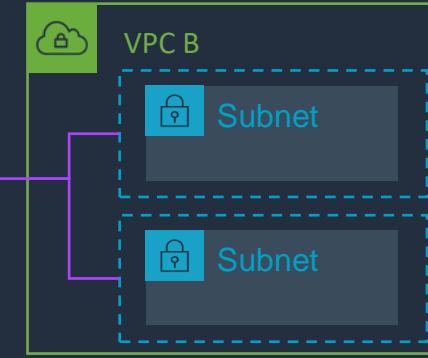




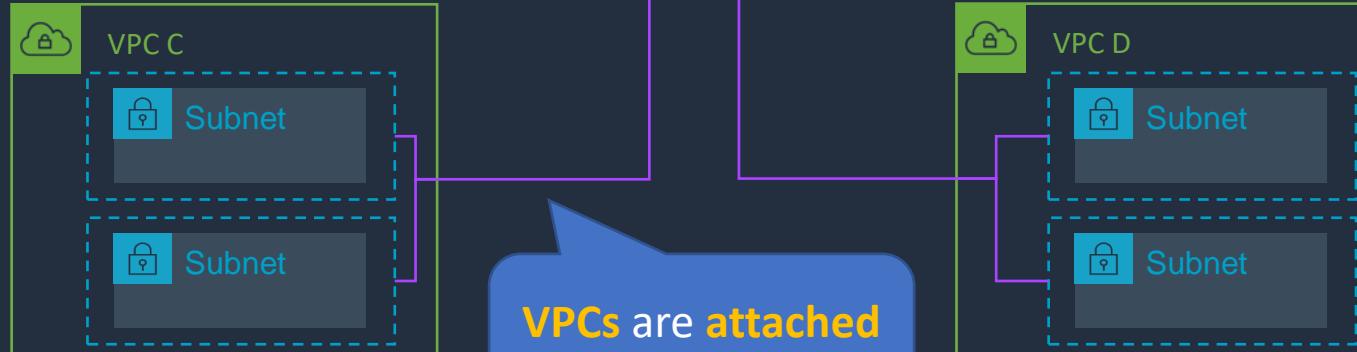
AWS Transit Gateway



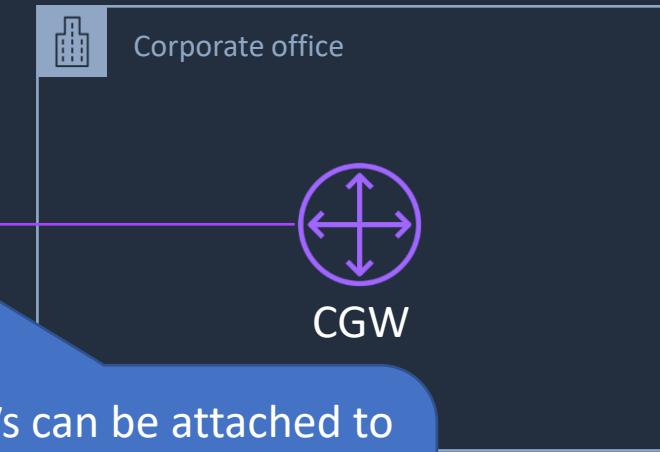
Example full mesh architecture **with** AWS Transit Gateway



Transit Gateway is a network transit hub that interconnects **VPCs** and **on-premises** networks



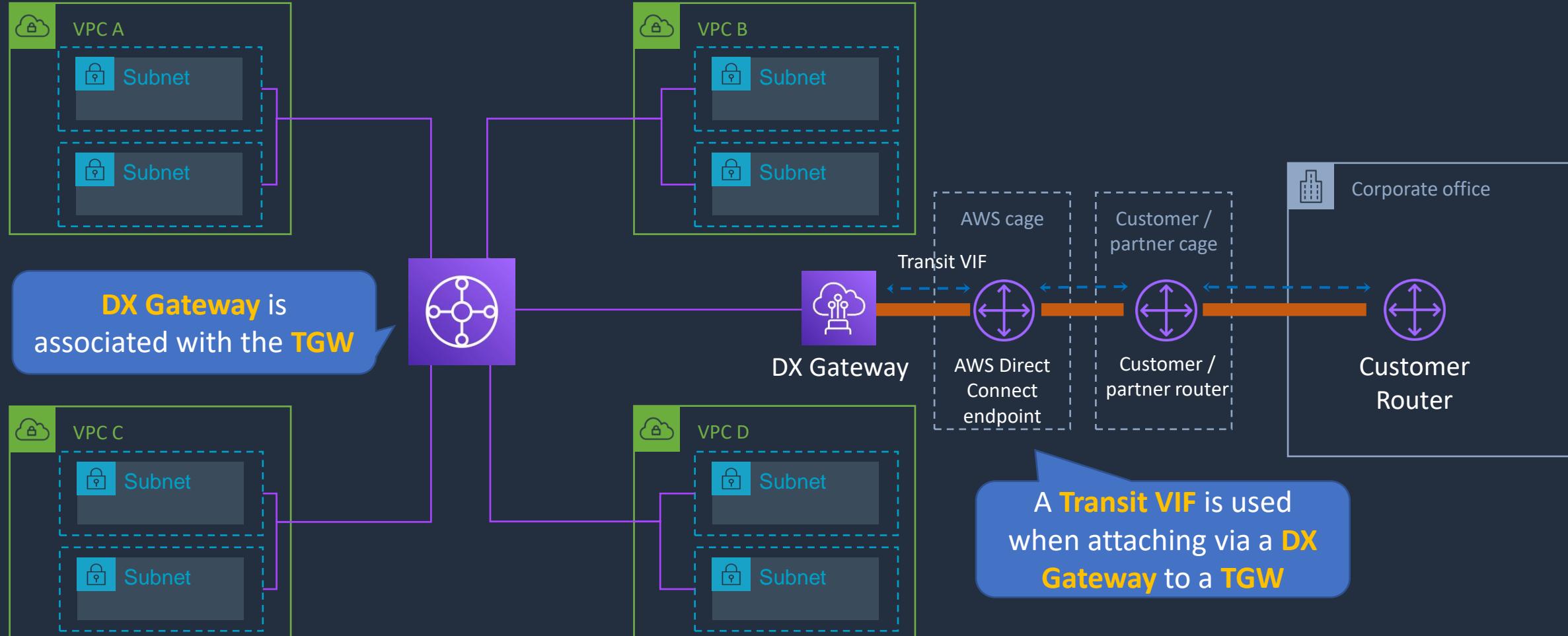
VPCs are **attached** to Transit Gateway



TGWs can be attached to **VPNs, Direct Connect Gateways, 3rd party appliances** and **TGWs** in other Regions/accounts

AWS TGW + DX Gateway

This architecture supports **full transitive** routing between **on-premises**, **TGW** and **VPCs**



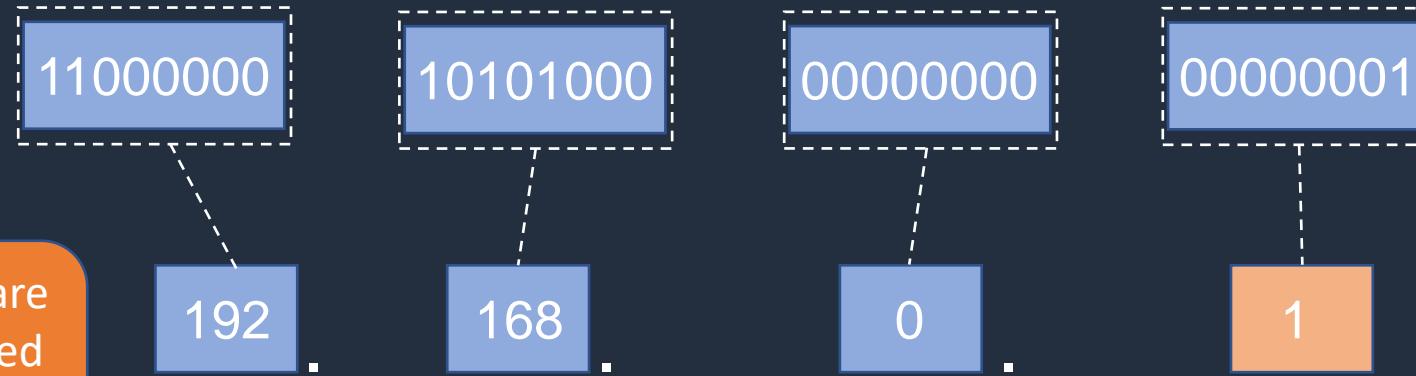
Using IPv6 in a VPC





Using IPv6 in a VPC

An IPv4 address is **32 bits** long



Public **IPv4** addresses are close to being exhausted and **NAT** must be used extensively



IPv4 provides approximately **4.3 billion** addresses



Using IPv6 in a VPC

An IPv6 address is **128 bits** long

2020 : 0001 : 9d32 : 5bc2 : 1c48 : 32c1 : a93b : b12c

Network Part

Node Part

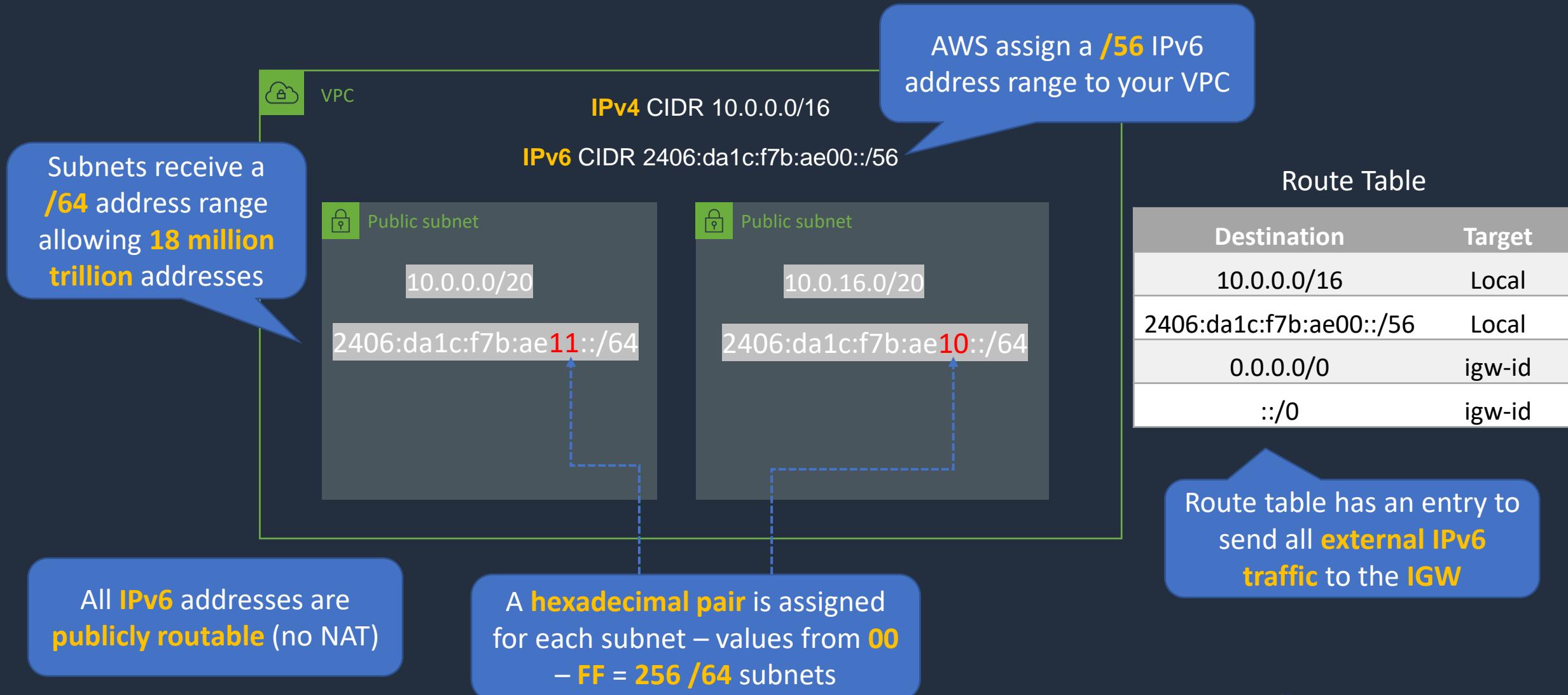
An **IPv6** addresses use **hexadecimal** whereas **IPv4** addresses use **dotted decimal**

That's enough to assign more than **100 IPv6 addresses** to **every atom** on earth!!!

IPv6 provides **340,282,366,920,938,463,463,374,607,431,768,211,456** addresses

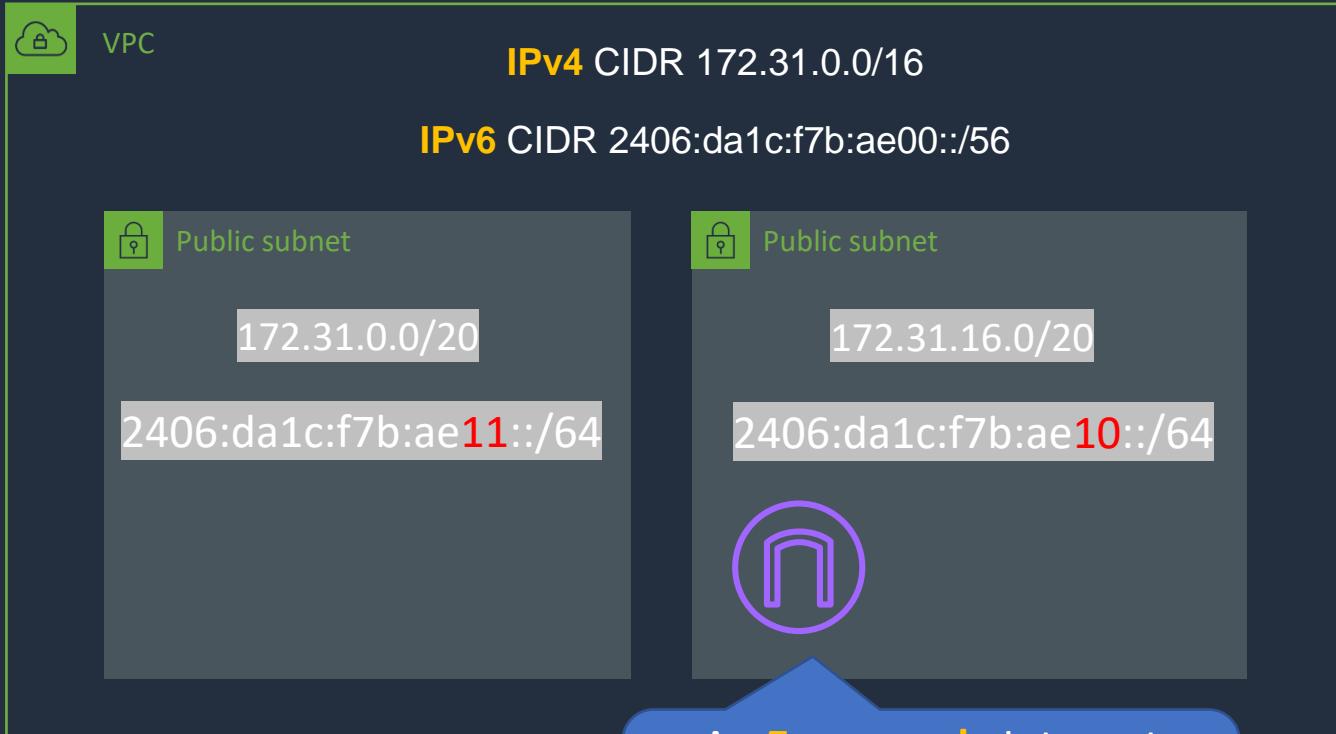


Using IPv6 in a VPC





Using IPv6 in a VPC



All **IPv6** addresses are
publicly routable (no NAT)

An **Egress-only** Internet
Gateway allows IPv6 traffic
outbound but **not inbound**

Route Table	
Destination	Target
172.31.0.0/16	Local
0.0.0.0/0	igw-id
::/0	eo-igw-id

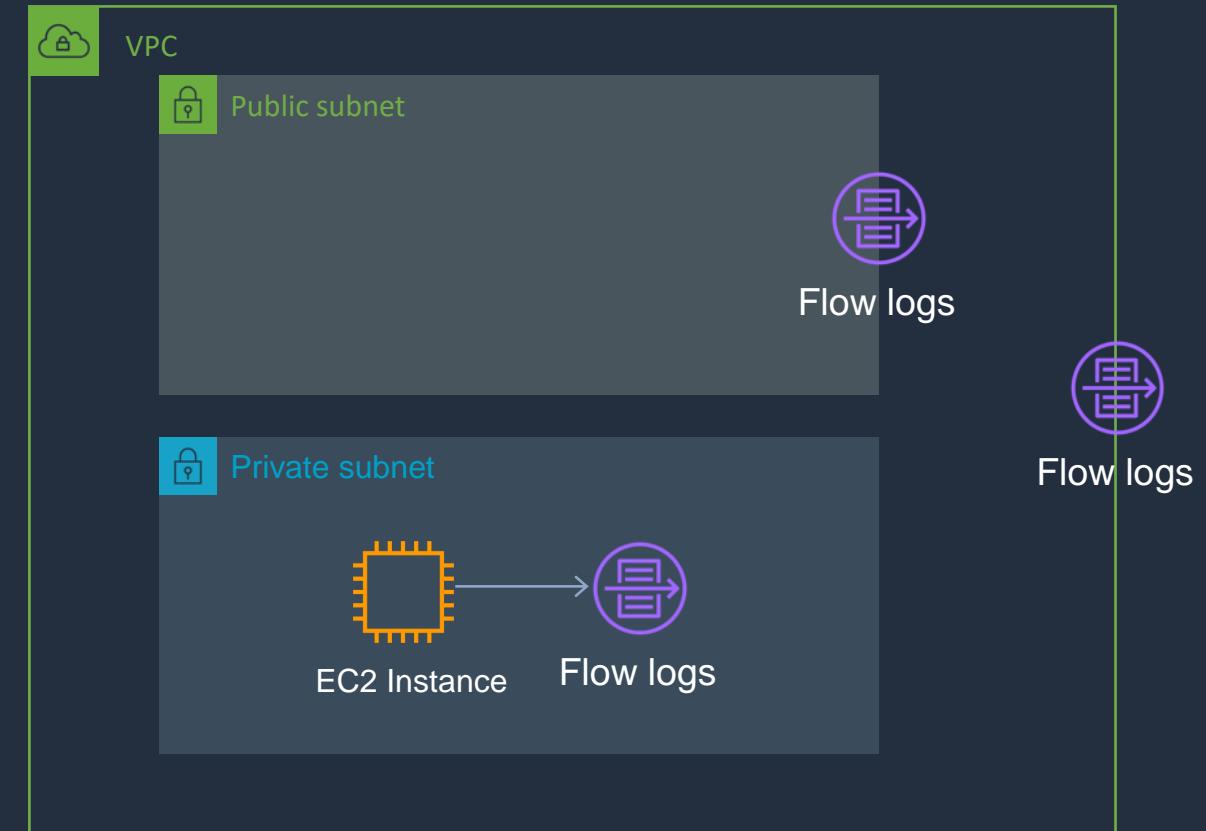
VPC Flow Logs



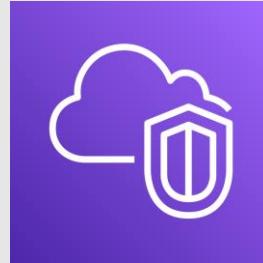


VPC Flow Logs

- Flow Logs capture information about the IP traffic going to and from network interfaces in a VPC
- Flow log data is stored using Amazon CloudWatch Logs or S3
- Flow logs can be created at the following levels:
 - VPC
 - Subnet
 - Network interface



Architecture Patterns – Amazon VPC





Architecture Patterns – Amazon VPC

Requirement

An Amazon S3 bucket must only allow access from EC2 instances in a private subnet using private IPs

Malicious traffic is reaching some EC2 instances in a public subnet from a few identified public IP addresses

A company wants to connect their on-premises data center to AWS and requires consistent performance and encryption

Solution

Create a VPC endpoint and configure a bucket policy that restricts access to the VPC endpoint ID using the Condition element

Use a Network ACL to deny access based on the source IP addresses

Create an AWS Direct Connect connection and run an AWS S2S VPN over the DX connection to enable encryption



Architecture Patterns – Amazon VPC

Requirement

A company requires private connectivity between VPCs in different Regions will full redundancy

Solution

Create VPC peering connections between the VPCs

Several remote office locations should be connected to an Amazon VPC and to each other over the internet with full encryption

Create VPG and attach multiple remote locations in a hub and spoke topology using AWS CloudHub

Microservices app requires instance-level firewall with different rules per application component

Create separate security groups for each application component and configure the appropriate rules



Architecture Patterns – Amazon VPC

Requirement

A company is using IPv6 addresses with Amazon EC2 and needs to enable outbound internet connectivity

Subnet must be configured that allows internet connectivity using IPv4 with auto-address assignment

An on-premises data center needs to establish S2S VPN connections to several VPCs in a full mesh architecture

Solution

Configure an egress-only Internet Gateway

Attach an internet gateway to the VPC and update the route table for the new subnet. Enable the auto-assign public IPv4 setting for the subnet

Deploy an AWS Transit Gateway and attach the VPN connection from on-premises and each VPC

SECTION 7

Amazon Simple Storage Service (S3)

Amazon Simple Storage Service (S3)





Amazon Simple Storage Service (S3)



S3 Bucket



A **bucket** is a container for objects

An **object** is a file you upload

You can store millions of **objects** in a **bucket**



Accessing objects in a bucket:

`https://bucket.s3.aws-region.amazonaws.com/key`
`https://s3.aws-region.amazonaws.com/bucket/key`

The **HTTP protocol** is used with a **REST API** (e.g. GET, PUT, POST, SELECT, DELETE)



Amazon Simple Storage Service (S3)



Bucket

A **bucket** is a container for objects

<http://bucket.s3.aws-region.amazonaws.com>

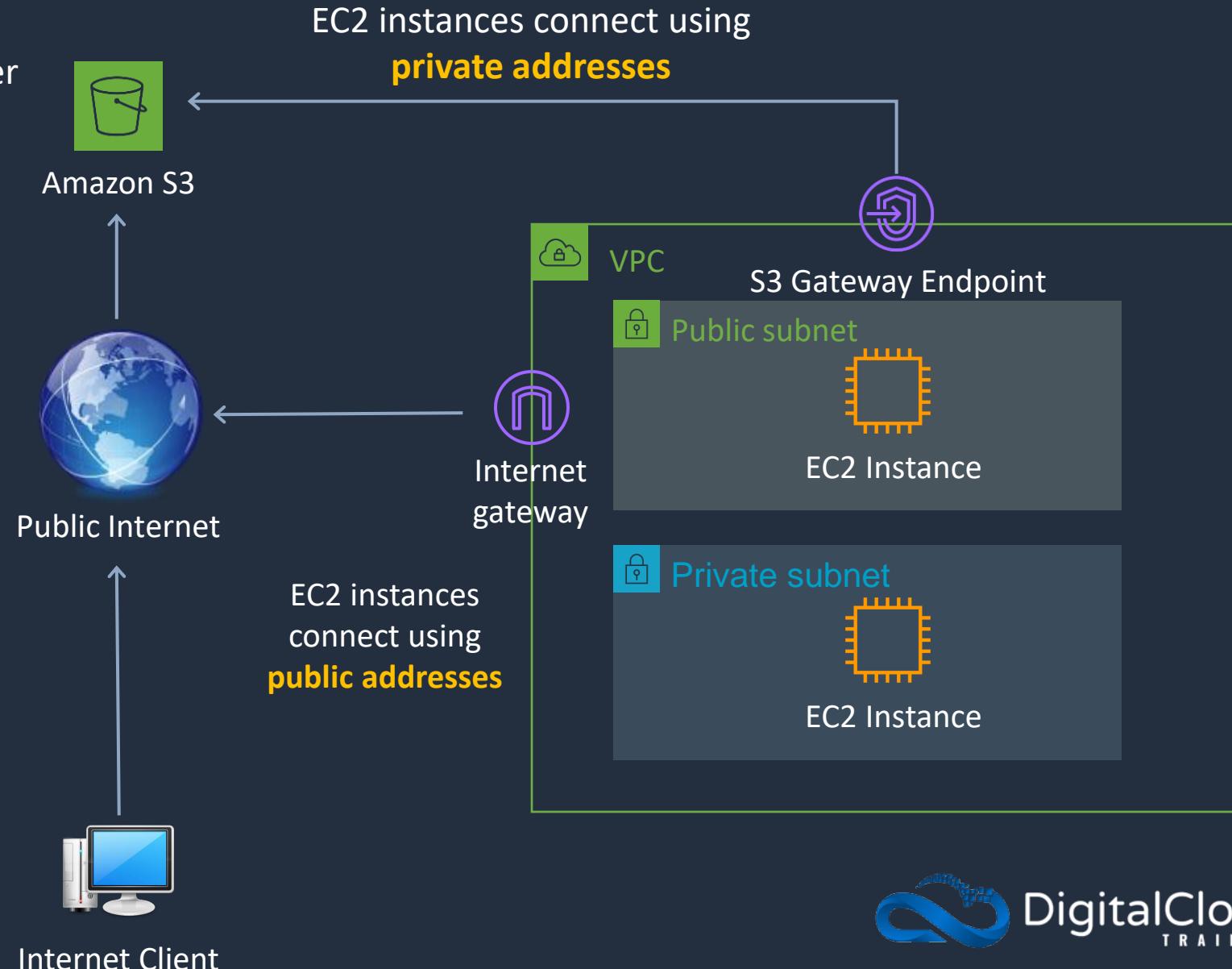
<http://s3.aws-region.amazonaws.com/bucket>



Object

An object consists of:

- Key (name of objects)
- Version ID
- Value (actual data)
- Metadata
- Subresources
- Access control information





File Storage vs Object Storage

File Share



- Data stored in directories
- A hierarchy of directories can be formed
- File systems are mounted to an operating system
- Function like local storage
- Network connection is maintained
- Example is Amazon EFS

Object Store



`http://bucket.s3.aws-region.amazonaws.com`

- Data stored in buckets
- Flat namespace (no hierarchy)
- Hierarchy can be mimicked with prefixes
- Accessed by **REST API** and cannot be mounted
- Network connection is completed after each request
- Example is Amazon S3

Amazon S3 Storage Classes





Durability and Availability in S3

Durability

Durability is protection against:

- Data loss
- Data corruption
- S3 offers 11 9s durability (99.99999999)

If you store 10 million objects, then you expect to lose one object every 10,000 years!

Availability

Availability is a measurement of:

- The amount of time the data is available to you
- Expressed as a percent of time per year
- E.g. 99.99%



S3 Storage Classes

	S3 Standard	S3 Intelligent Tiering	S3 Standard-IA	S3 One Zone-IA	S3 Glacier Instant Retrieval	S3 Glacier Flexible Retrieval	S3 Glacier Deep Archive
Designed for durability	99.999999999%	99.999999999%	99.999999999%	99.999999999%	99.999999999%	99.999999999%	99.999999999%
Designed for availability	99.99%	99.9%	99.9%	99.5%	99.9%	99.99%	99.99%
Availability SLA	99.9%	99%	99%	99%	99%	99.9%	99.9%
Availability Zones	≥3	≥3	≥3	1	≥3	≥3	≥3
Minimum capacity charge per object	N/A	N/A	128KB	128KB	128KB	40KB	40KB
Minimum storage duration charge	N/A	N/A	30 days	30 days	90 days	90 days	180 days
Retrieval fee	N/A	N/A	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved
First byte latency	milliseconds	milliseconds	milliseconds	milliseconds	milliseconds	minutes or hours	hours
Storage type	Object	Object	Object	Object	Object	Object	Object
Lifecycle transitions	Yes	Yes	Yes	Yes	Yes	Yes	Yes

Working with S3 Buckets and Objects



IAM Policies, Bucket Policies and ACLs





IAM Policies

- IAM Policies are identity-based policies
- Specify what actions are allowed on what AWS resources

IAM Policies are attached to IAM **users, groups, or roles**



IAM Policies are written in JSON using the AWS access policy language

The Principal element is not required in the policy



User



Group



Role



Bucket Policies

- Bucket Policies are resource-based policies
- Can only be attached to Amazon S3 buckets
- Also use the AWS access policy language



```
{  
  "Version": "2012-10-17",  
  "Id": "Policy1561964929358",  
  "Statement": [  
    {  
      "Sid": "Stmt1561964454052",  
      "Effect": "Allow",  
      "Principal": {  
        "AWS": "arn:aws:iam::515148227241:user/Paul"  
      },  
      "Action": "s3:*",  
      "Resource": "arn:aws:s3:::dctcompany"  
    }  
  ]  
}
```



S3 Access Control Lists (ACLs)

- Legacy access control mechanism that predates IAM
- AWS generally recommends using **S3 bucket policies** or **IAM policies** rather than ACLs
- Can be attached to a **bucket** or directly to an **object**
- Limited options for grantees and permissions



When to use each access control mechanism

- **Use IAM policies if:**

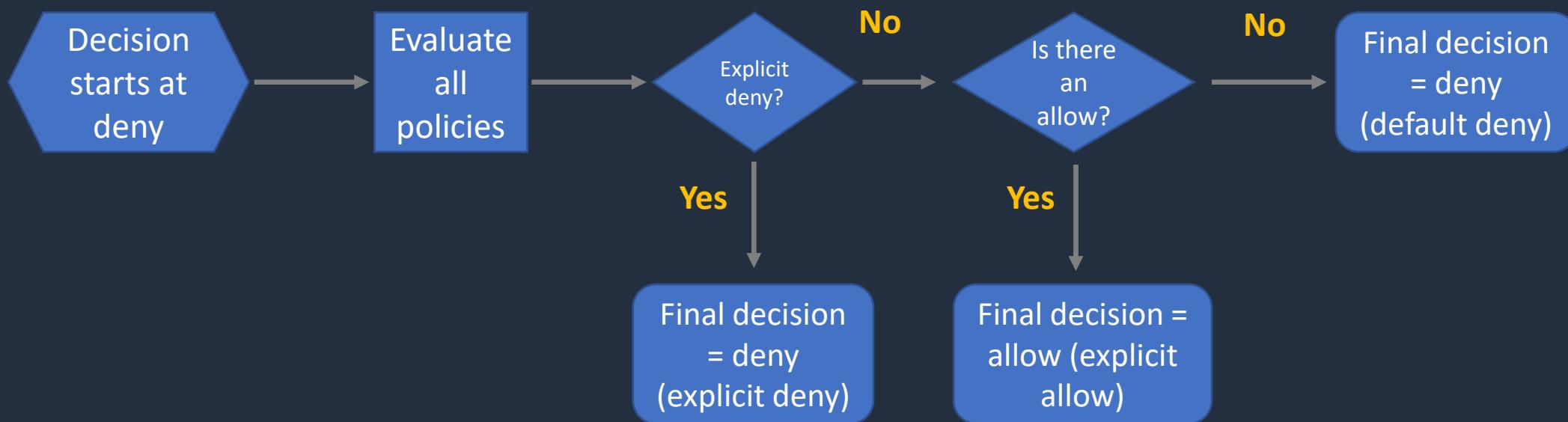
- You need to control access to AWS services other than S3
- You have numerous S3 buckets each with different permissions requirements (IAM policies will be easier to manage)
- You prefer to keep access control policies in the IAM environment

- **Use S3 bucket policies if:**

- You want a simple way to grant cross-account access to your S3 environment, without using IAM roles
- Your IAM policies are reaching the size limits
- You prefer to keep access control policies in the S3 environment



Authorization Process



S3 Permissions and Bucket Policies



S3 Versioning, Replication and Lifecycle Rules





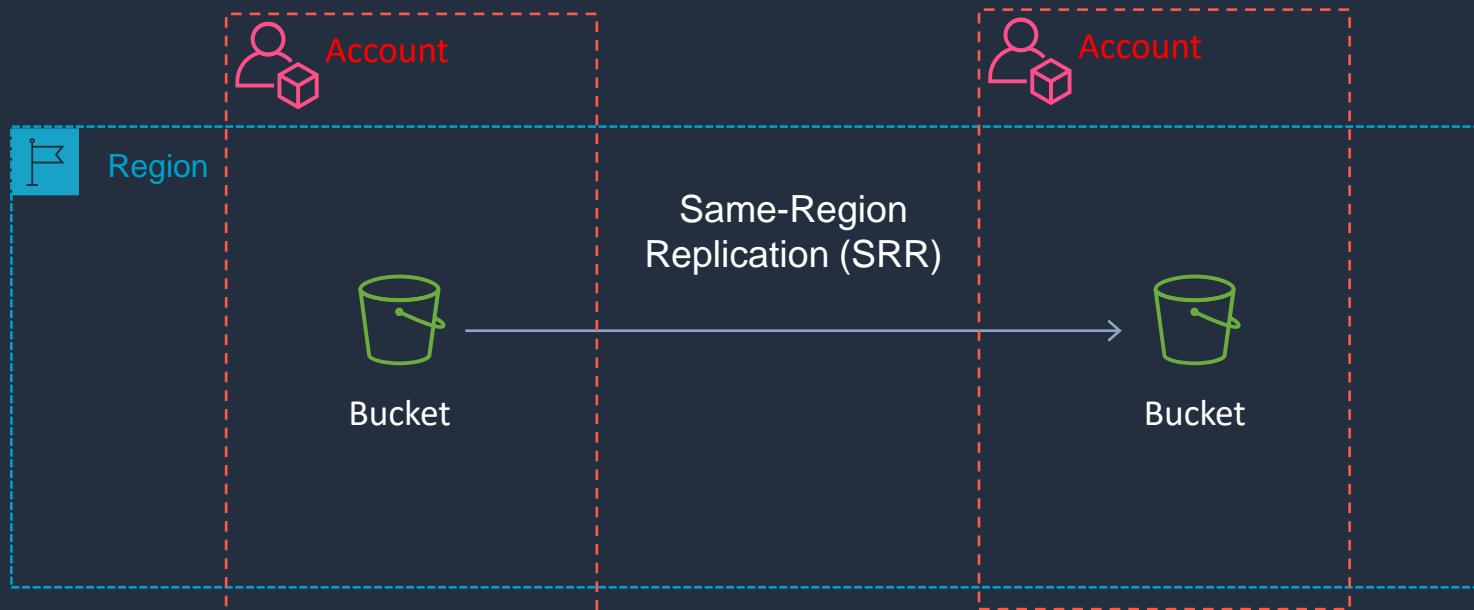
S3 Versioning

- Versioning is a means of keeping multiple variants of an object in the same bucket
- Use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket
- Versioning-enabled buckets enable you to recover objects from **accidental deletion** or **overwrite**



S3 Replication

Cross-Region Replication (CRR)



Buckets must have
versioning enabled



S3 Lifecycle Management

There are two types of actions:

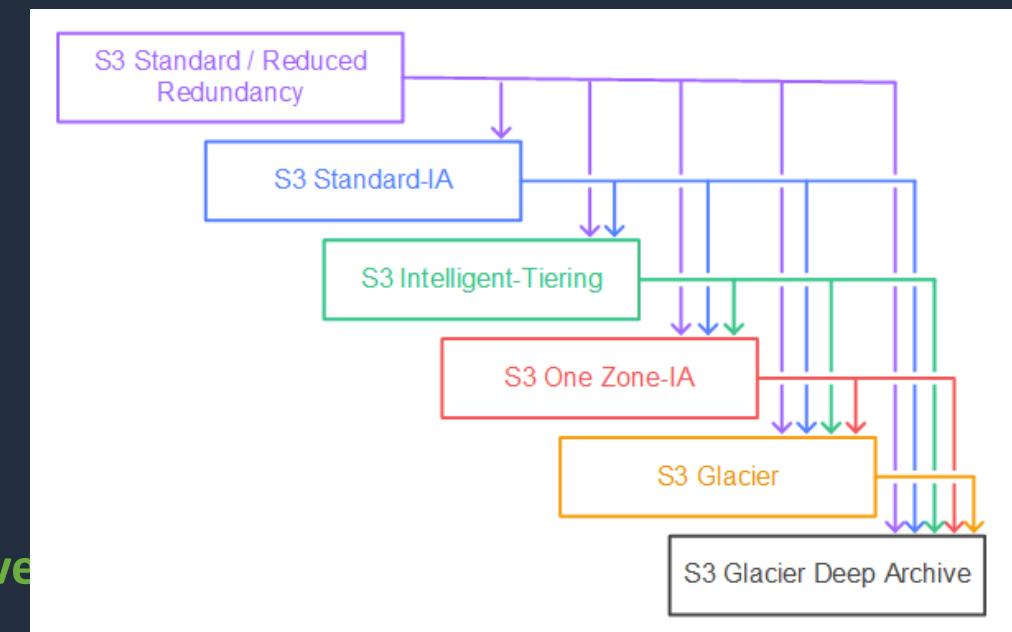
- **Transition actions** - Define when objects transition to another storage class
- **Expiration actions** - Define when objects expire (deleted by S3)



S3 LM: Supported Transitions

You can transition from the following:

- The **S3 Standard** storage class to any other storage class
- Any storage class to the **S3 Glacier** or **S3 Glacier Deep Archive** storage classes
- The **S3 Standard-IA** storage class to the **S3 Intelligent-Tiering** or **S3 One Zone-IA** storage classes
- The **S3 Intelligent-Tiering** storage class to the **S3 One Zone-IA** storage class
- The **S3 Glacier** storage class to the **S3 Glacier Deep Archive** storage class

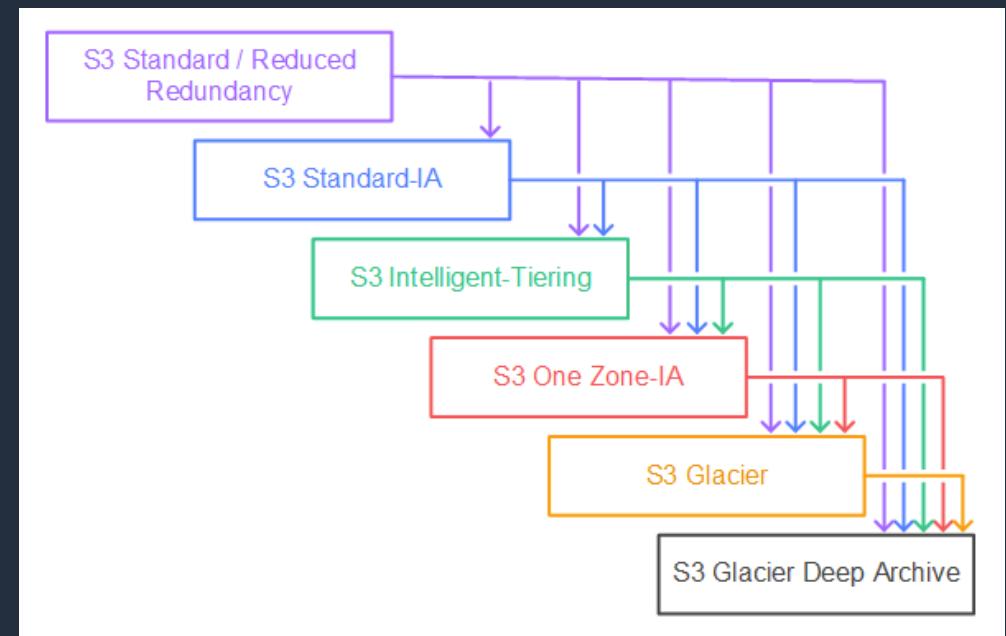




S3 LM: Unsupported Transitions

You can't transition from the following:

- Any storage class to the **S3 Standard** storage class
- Any storage class to the **Reduced Redundancy** storage class
- The **S3 Intelligent-Tiering** storage class to the **S3 Standard-IA** storage class
- The **S3 One Zone-IA** storage class to the **S3 Standard-IA** or **S3 Intelligent-Tiering** storage classes



Configure Replication and Lifecycle

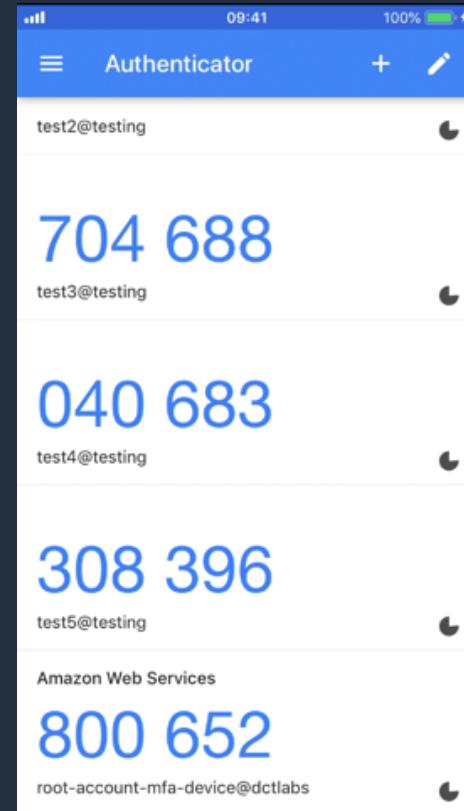


MFA with Amazon S3



S3 Multi-Factor Authentication Delete (MFA Delete)

- Adds MFA requirement for bucket owners to the following operations:
 - Changing the versioning state of a bucket
 - Permanently deleting an object version
- The **x-amz-mfa** request header must be included in the above requests
- The second factor is a token generated by a hardware device or software program
- Requires **versioning** to be enabled on the bucket



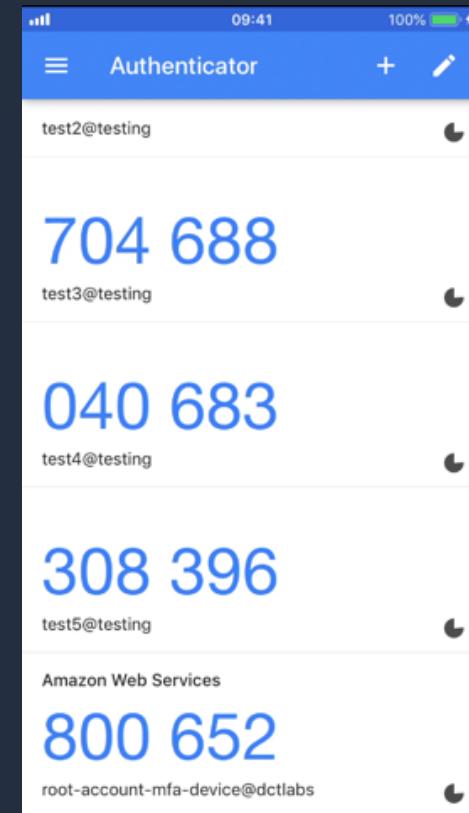
S3 Multi-Factor Authentication Delete (MFA Delete)

- **Versioning** can be enabled by:

- Bucket owners (root account)
- AWS account that created the bucket
- Authorized IAM users

- **MFA delete** can be enabled by:

- Bucket owner (root account)



MFA-Protected API Access

- Used to enforce another authentication factor (MFA code) when accessing AWS resources (not just S3)
- Enforced using the `aws:MultiFactorAuthAge` key in a bucket policy:

```
{  
    "Version": "2012-10-17",  
    "Id": "123",  
    "Statement": [  
        {  
            "Sid": "",  
            "Effect": "Deny",  
            "Principal": "*",  
            "Action": "s3:*",  
            "Resource": "arn:aws:s3:::examplebucket/securedocuments/*",  
            "Condition": { "Null": { "aws:MultiFactorAuthAge": true } }  
        }  
    ]  
}
```

Denies any API operation
that is not authenticated
using MFA

Amazon S3 Encryption





Amazon S3 Encryption

Server-side encryption with S3 managed keys (SSE-S3)



- S3 managed keys
- Unique object keys
- Master key
- AES 256



Encryption / decryption



User

Server-side encryption with AWS KMS managed keys (SSE-KMS)



- KMS managed keys
- Can be AWS managed keys
- Or customer managed KMS keys



Encryption / decryption

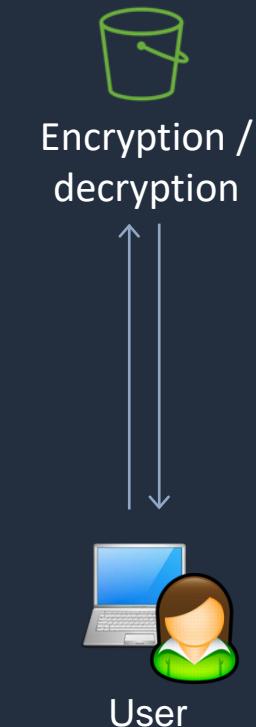


User



Amazon S3 Encryption

Server-side encryption with client provided keys (SSE-C)



- Client managed keys
- Not stored on AWS

Client-side encryption



- Client managed keys
- Not stored on AWS
- Or you can use a KMS Key



Amazon S3 Default Encryption

- All Amazon S3 buckets have encryption configured by default
- All new object uploads to Amazon S3 are automatically encrypted
- There is no additional cost and no impact on performance
- Objects are automatically encrypted by using server-side encryption with Amazon S3 managed keys (SSE-S3)
- To encrypt existing unencrypted Amazon S3 objects, you can use Amazon S3 Batch Operations
- You can also encrypt existing objects by using the **CopyObject** API operation or the **copy-object** AWS CLI command



Enforce Encryption with Bucket Policy

```
{  
    "Version": "2012-10-17",  
    "Id": "PutObjPolicy",  
    "Statement": [  
        {  
            "Sid": "DenyIncorrectEncryptionHeader",  
            "Effect": "Deny",  
            "Principal": "*",  
            "Action": "s3:PutObject",  
            "Resource": "arn:aws:s3:::<bucket_name>/*",  
            "Condition": {  
                "StringNotEquals": {  
                    "s3:x-amz-server-side-encryption": "aws:kms"  
                }  
            },  
            {  
                "Sid": "DenyUnEncryptedObjectUploads",  
                "Effect": "Deny",  
                "Principal": "*",  
                "Action": "s3:PutObject",  
                "Resource": "arn:aws:s3:::<bucket_name>/*",  
                "Condition": {  
                    "Null": {  
                        "s3:x-amz-server-side-encryption": true  
                    }  
                }  
            }  
        ]  
    ]  
}
```

Enforces encryption
using SSE-KMS

Example PUT request

```
PUT /example-object HTTP/1.1  
Host: myBucket.s3.amazonaws.com  
Date: Wed, 8 Jun 2016 17:50:00 GMT  
Authorization: authorization string  
Content-Type: text/plain  
Content-Length: 11434  
x-amz-meta-author: Janet  
Expect: 100-continue  
x-amz-server-side-encryption: aws:kms  
[11434 bytes of object data]
```

Enforce Encryption with AWS KMS





Enforce Encryption with Bucket Policy

```
{  
    "Version": "2012-10-17",  
    "Id": "PutObjPolicy",  
    "Statement": [  
        {  
            "Sid": "DenyIncorrectEncryptionHeader",  
            "Effect": "Deny",  
            "Principal": "*",  
            "Action": "s3:PutObject",  
            "Resource": "arn:aws:s3:::<bucket_name>/*",  
            "Condition": {  
                "StringNotEquals": {  
                    "s3:x-amz-server-side-encryption": "aws:kms"  
                }  
            },  
            {  
                "Sid": "DenyUnEncryptedObjectUploads",  
                "Effect": "Deny",  
                "Principal": "*",  
                "Action": "s3:PutObject",  
                "Resource": "arn:aws:s3:::<bucket_name>/*",  
                "Condition": {  
                    "Null": {  
                        "s3:x-amz-server-side-encryption": true  
                    }  
                }  
            }  
        ]  
    ]  
}
```

Enforces encryption
using SSE-KMS

Example PUT request

```
PUT /example-object HTTP/1.1  
Host: myBucket.s3.amazonaws.com  
Date: Wed, 8 Jun 2016 17:50:00 GMT  
Authorization: authorization string  
Content-Type: text/plain  
Content-Length: 11434  
x-amz-meta-author: Janet  
Expect: 100-continue  
x-amz-server-side-encryption: aws:kms  
[11434 bytes of object data]
```

S3 Event Notifications





S3 Event Notifications

- Sends notifications when events happen in buckets
- Destinations include:
 - Amazon Simple Notification Service (SNS) topics
 - Amazon Simple Queue Service (SQS) queues
 - AWS Lambda functions

Event types
Specify at least one event for which you want to receive notifications. For each group, you can choose an event type for all events, or you can choose one or more individual events.

Object creation

All object create events
`s3:ObjectCreated:*`

Put
`s3:ObjectCreated:Put`

Post
`s3:ObjectCreated:Post`

Copy
`s3:ObjectCreated:Copy`

Multipart upload completed
`s3:ObjectCreated:CompleteMultipartUpload`

Object removal

All object removal events
`s3:ObjectRemoved:*`

Permanently deleted
`s3:ObjectRemoved:Delete`

Delete marker created
`s3:ObjectRemoved:DeleteMarkerCreated`

S3 Presigned URLs





S3 Presigned URLs



```
aws s3 presign s3://dct-data-bucket/presigned_index.html
```



```
https://mywebsite-3eqd2a.s3.us-east-1.amazonaws.com/index.html?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=ASIA6GBMFBGVJIKXMWK%2F20240322%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-
```

Multipart Upload & Transfer Acceleration





S3 Multipart Upload

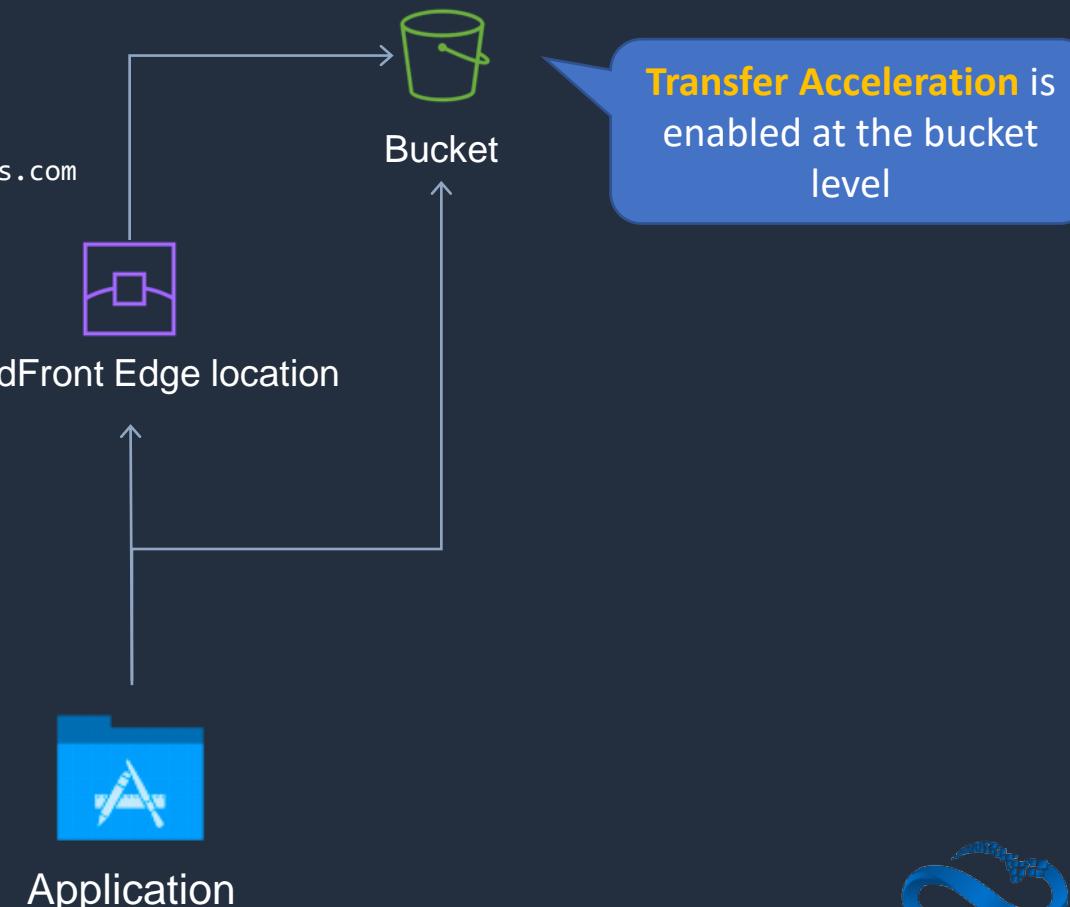
- Multipart upload uploads objects in parts independently, in parallel and in any order
- Performed using the S3 Multipart upload API
- It is recommended for objects of 100 MB or larger
- Can be used for objects from 5 MB up to 5 TB
- Must be used for objects larger than 5 GB

S3 Transfer Acceleration

- Uses CloudFront edge locations to improve performance of transfers from client to S3 bucket

`http://bucketname.s3-accelerate.amazonaws.com`
`http://bucketname.s3-accelerate.dualstack.amazonaws.com`

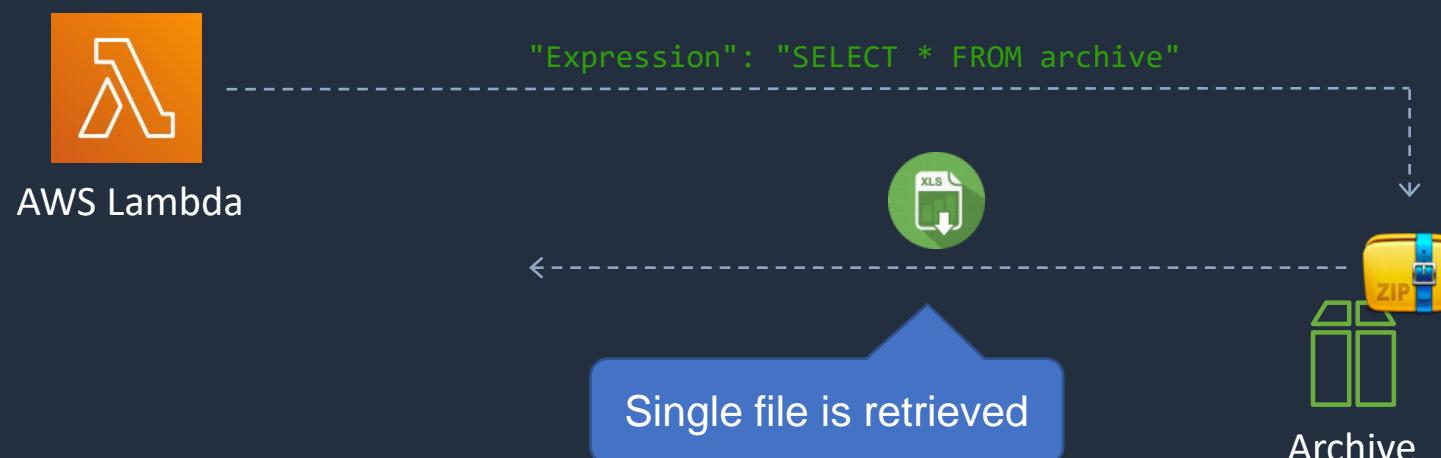
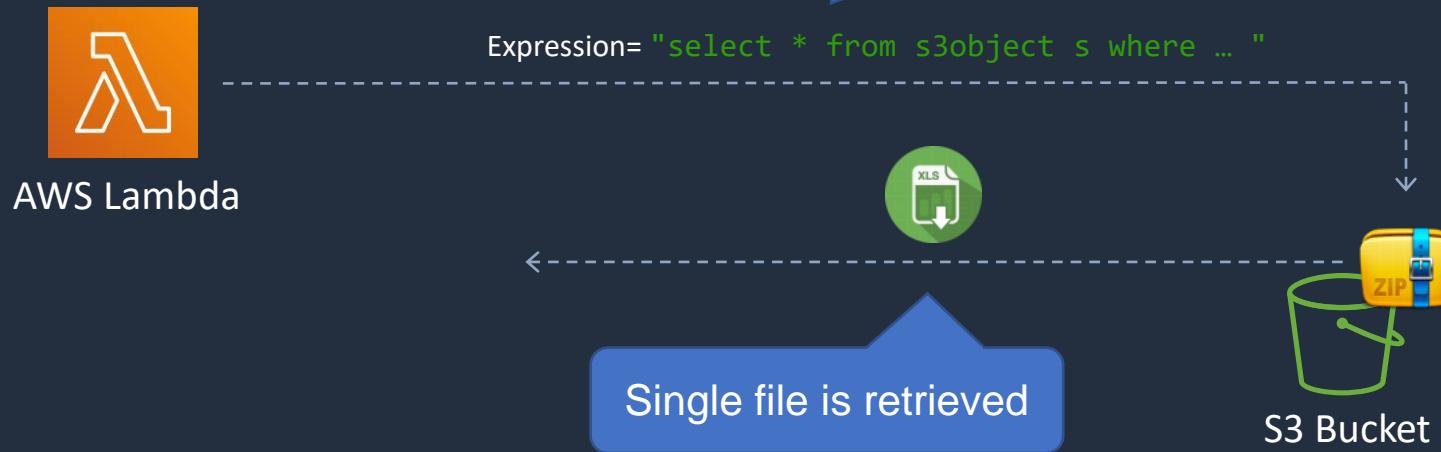
AWS only charges if
there's a performance
improvement



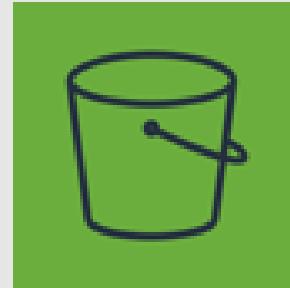
S3 Select and Glacier Select



S3 Select and Glacier Select



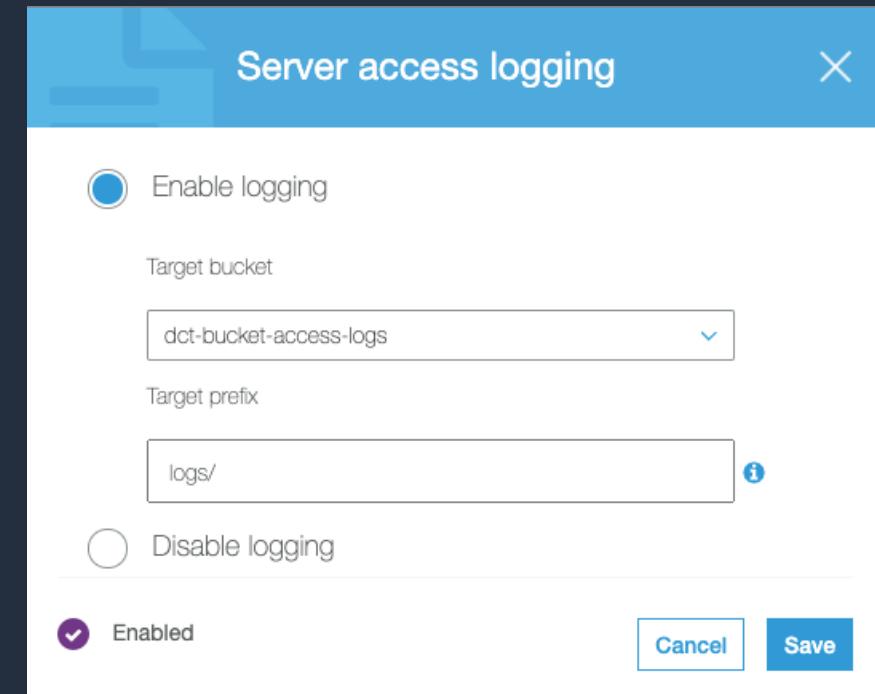
Server Access Logging





Server Access Logging

- Provides detailed records for the requests that are made to a bucket
- Details include the requester, bucket name, request time, request action, response status, and error code (if applicable)
- Disabled by default
- Only pay for the storage space used
- Must configure a separate bucket as the destination (can specify a prefix)
- Must grant write permissions to the Amazon S3 Log Delivery group on destination bucket



Create an Amazon S3 Static Website



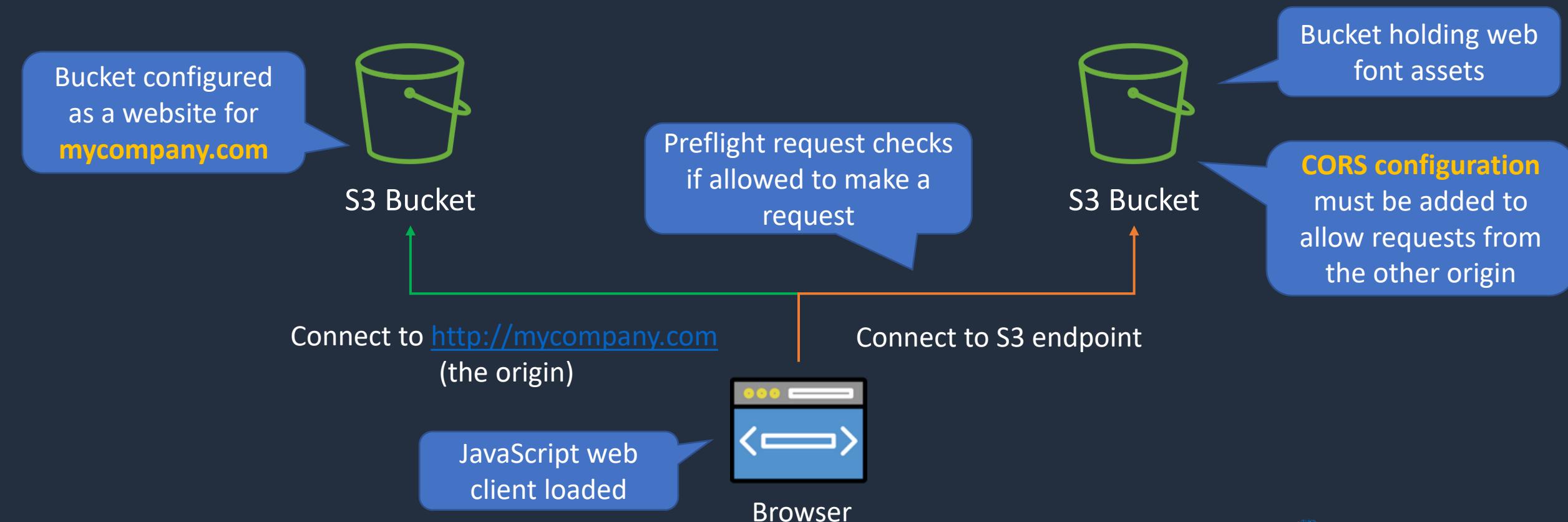
Cross-Origin Resource Sharing (CORS)





CORS with Amazon S3

- Allows requests from an origin to another origin
- Origin is defined by DNS name, protocol, and port





CORS with Amazon S3

- Enabled through setting:
 - Access-Control-Allow-Origin
 - Access-Control-Allow-Methods
 - Access-Control-Allow-Headers
- These settings are defined using rules
- Rules are added using JSON files in S3



Example CORS Rule

```
[  
  {  
    "AllowedHeaders": [  
      "*"  
    ],  
    "AllowedMethods": [  
      "PUT",  
      "POST",  
      "DELETE"  
    ],  
    "AllowedOrigins": [  
      "http://www.mycompany.com"  
    ],  
    "ExposeHeaders": []  
  }  
]
```

Cross-Account Access

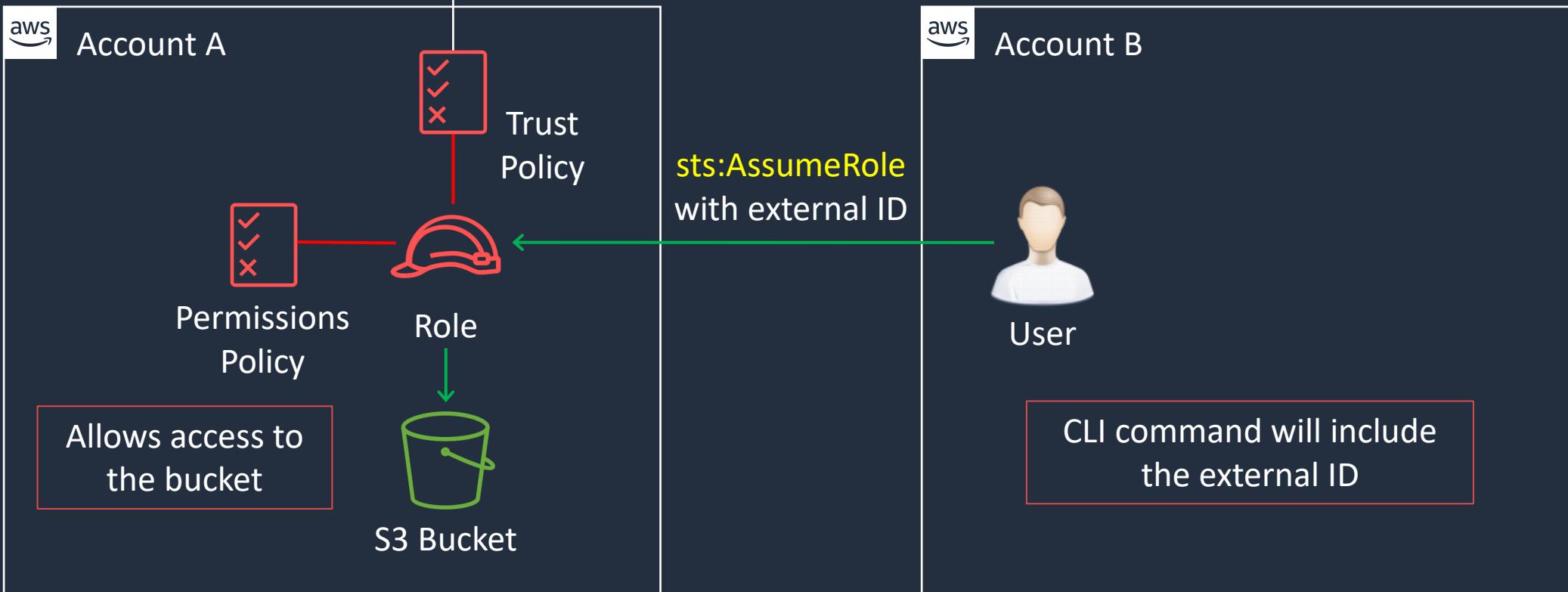




Cross Account Access (IAM Role)

The trust policy condition requires the external ID

```
"Statement": {  
    "Effect": "Allow",  
    "Action": "sts:AssumeRole",  
    "Principal": {"AWS": "3rd party AWS Account ID"},  
    "Condition": {"StringEquals": {"sts:ExternalId": "12345"}}}
```



S3 Object Lambda





S3 Object Lambda

- S3 Object Lambda uses Lambda functions to process the output of S3 GET requests
- You can use your own functions or use the AWS pre-built functions

GetObject requests trigger Lambda to process and modify data before returning it to the application





S3 Object Lambda – Prebuilt Functions

- Prebuilt Lambda functions that detect personally identifiable information (PII)
- PII includes names, addresses, dates, credit card numbers, and social security numbers
- Prebuilt functions include:
 - PII Access Control – detects PII and restricts access
`arn:aws:serverlessrepo:us-east-1:839782855223:applications/ComprehendPiiAccessControlS3ObjectLambda`
 - PII Redaction – detects PII and returns documents with the PII redacted
`arn:aws:serverlessrepo:us-east-1:839782855223:applications/ComprehendPiiRedactionS3ObjectLambda`
 - Decompression – decrypts objects compressed with bzip2, gzip, snappy, zlib, zstandard and ZIP
`arn:aws:serverlessrepo:eu-west-1:123065155563:applications/S3ObjectLambdaDecompression`

Architecture Patterns – Amazon S3





Architecture Patterns – Amazon S3

Requirement

Company is concerned about accidental deletion of Amazon S3 objects

Data stored in S3 is frequently accessed for 30 days then is rarely accessed but must be immediately retrievable

A backup of S3 objects within a specific folder in a bucket must be replicated to another Region

Solution

Enable S3 versioning

Use a lifecycle policy to transition objects from S3 standard to S3 Standard-IA after 30 days

Configure cross-region replication and specify the folder name as a prefix



Architecture Patterns – Amazon S3

Requirement

Previous versions of objects in a versioning-enabled S3 bucket must be stored long term at the lowest cost

A company wishes to manage all encryption of S3 objects through their application with their own encryption keys

Unencrypted objects in an Amazon S3 bucket must be encrypted

Solution

Create a lifecycle rule that transitions previous versions to S3 Glacier Deep Archive

Use client-side encryption with client managed keys

Re-upload the objects and specify the encryption an encryption key



Architecture Patterns – Amazon S3

Requirement

An administrator requires a notification when objects are deleted from an Amazon S3 bucket

Solution

Configure an event notification that uses the SNS service

A group of customers without AWS credentials must be granted time-limited access to a software update that is stored in an Amazon S3 bucket

Generate a presigned URL

Solutions architects require both programmatic and console access across AWS accounts

Configure cross-account access using IAM roles

SECTION 8

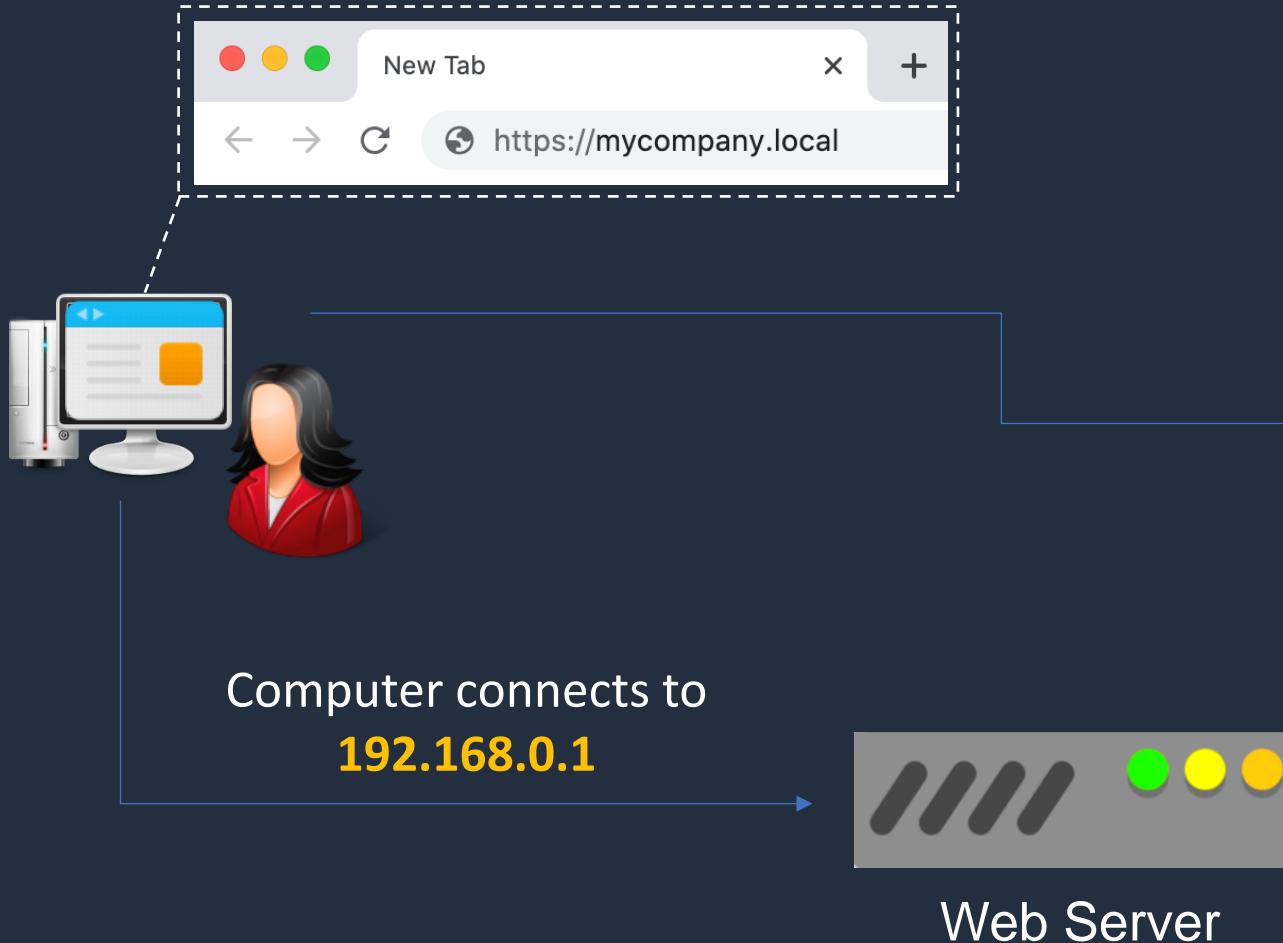
DNS, Caching, and Performance Optimization

DNS and Amazon Route 53



The Domain Name System (DNS)

User enters **website address** in browser



Name	Type	Value
mycompany.local	A	192.168.0.1
emailserver.local	A	192.168.0.2

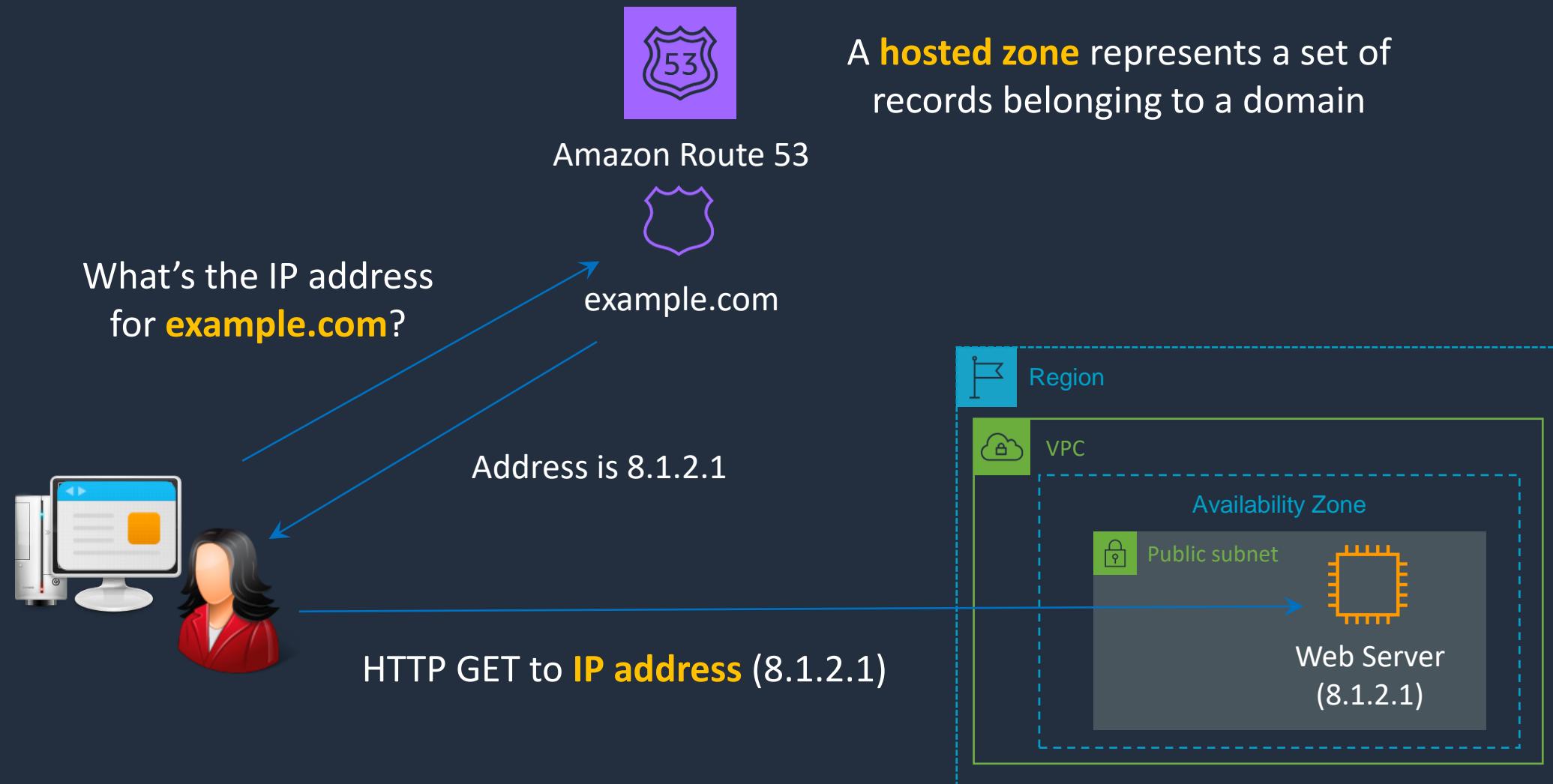


DNS Server

Domain name is resolved to the **IP address** of the webserver



Amazon Route 53



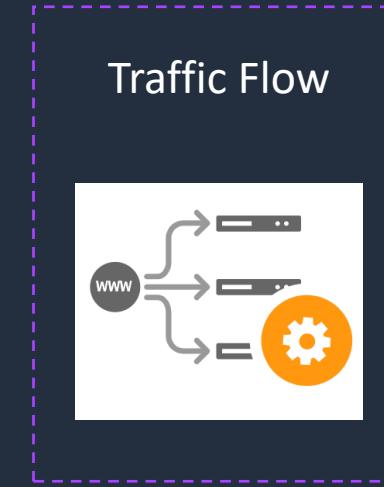
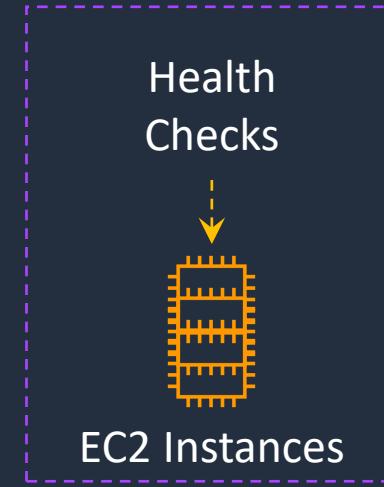
Amazon Route 53 Routing Policies

Routing Policy	What it does
Simple	Simple DNS response providing the IP address associated with a name
Failover	If primary is down (based on health checks), routes to secondary destination
Geolocation	Uses geographic location you're in (e.g. Europe) to route you to the closest region
Geoproximity	Routes you to the closest region within a geographic area
Latency	Directs you based on the lowest latency route to resources
Multivalue answer	Returns several IP addresses and functions as a basic load balancer
Weighted	Uses the relative weights assigned to resources to determine which to route to
IP-Based	Uses the IP addresses of clients to make routing decisions

Amazon Route Features



Amazon Route 53



Register Domain with Route 53 (Optional)

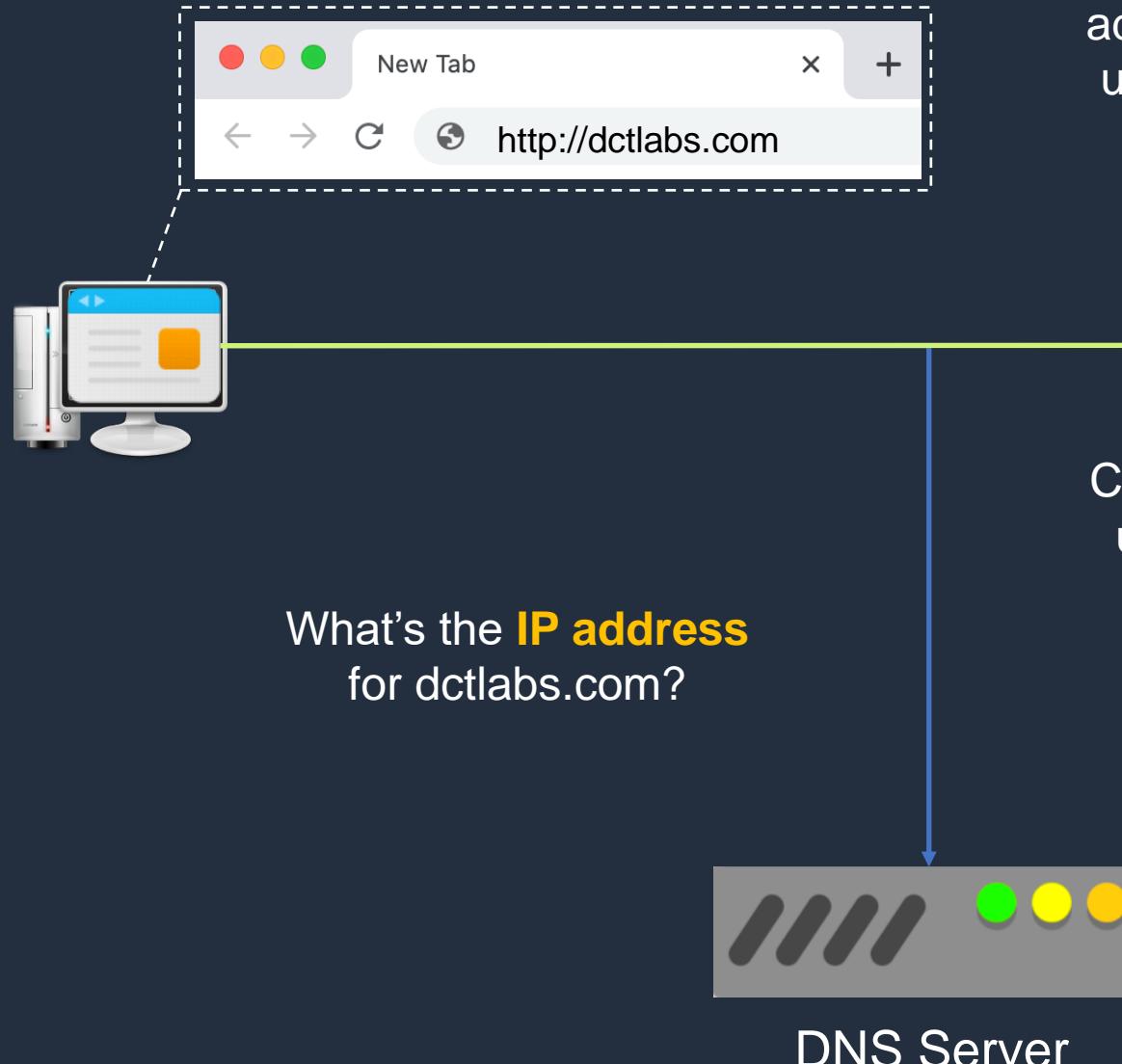


DNS and Amazon Route 53





The Domain Name System



IP addresses are the addresses computers use to communicate

Connection is made using **IP address**

52.134.26.12



Web Server

Name	Type	Value
<code>dctlabs.com</code>	A	52.134.26.12
<code>www.dctlabs.com</code>	A	52.134.26.12

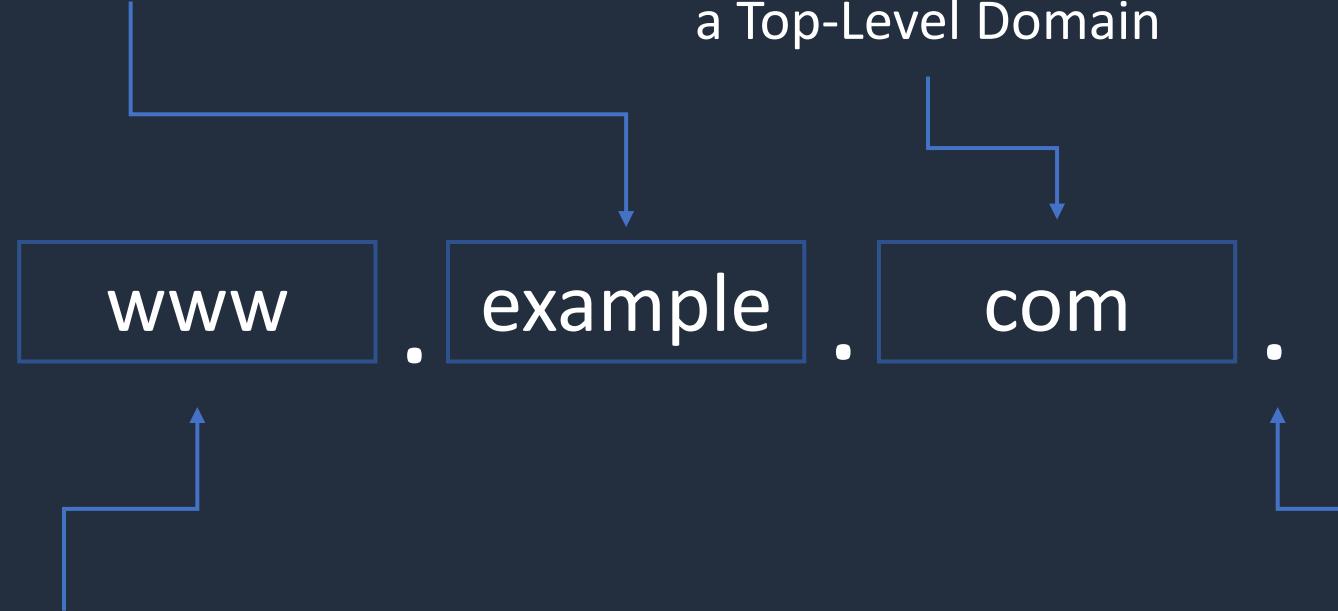
DNS Zone File



Fully Qualified Domain Names (FQDNs)

Example is a subdomain

com is an example of
a Top-Level Domain



www is a hostname within
the example subdomain

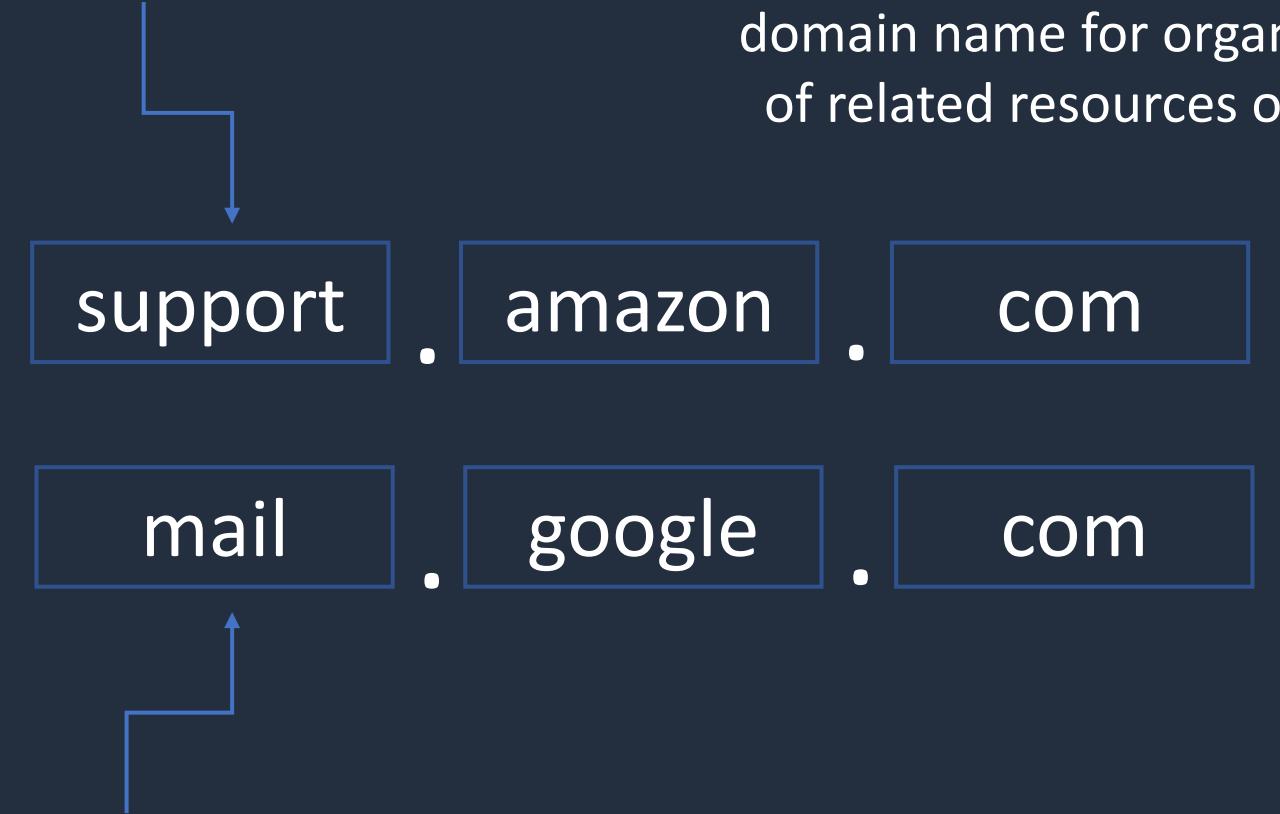
The **root domain** is represented by a
“.” And is not usually visible in a DNS
name



Subdomains

support is a subdomain
of amazon.com

A **subdomain** is subdivision of a domain name for organizing a set of related resources or services



mail is a subdomain
of google.com



DNS Zones and Records

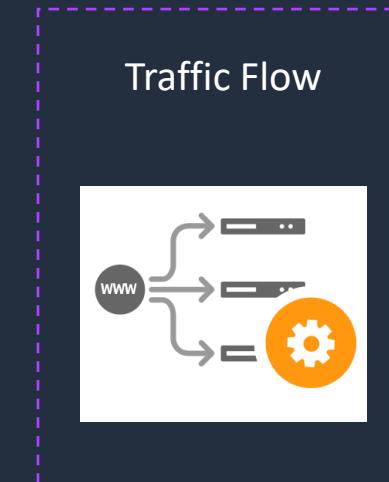
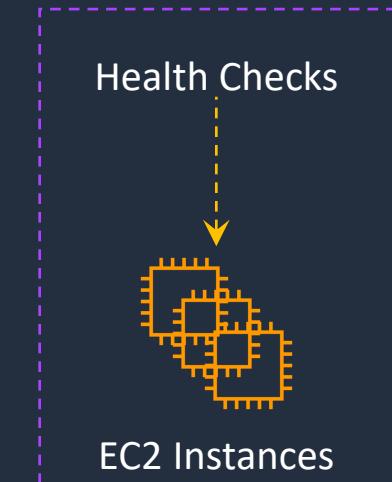
Record Type	Description
A	Maps a domain name to an IP address (e.g. dctlabs.com to 52.23.21.43)
CNAME	Maps a domain name to another domain name (e.g. mail.dctlabs.com to mailserver1.net)
MX	Returns the mail servers for a domain name
TXT	Associates text with a domain name (used for verification, authorization etc.)
SRV	Maps a domain name to a specific service or protocol (e.g. a Kerberos server)
NS	Specifies the authoritative DNS servers for a particular domain
SOA	Start of Authority record stores important information about the domain



Amazon Route 53

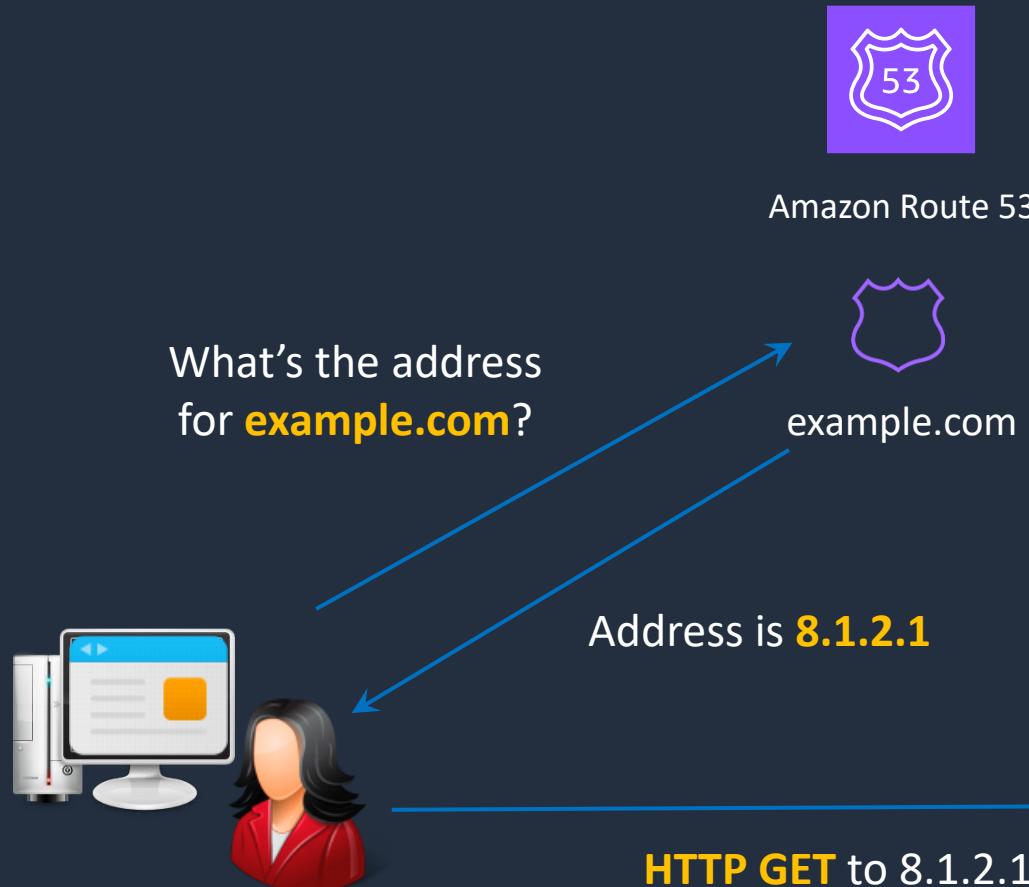


Amazon Route 53

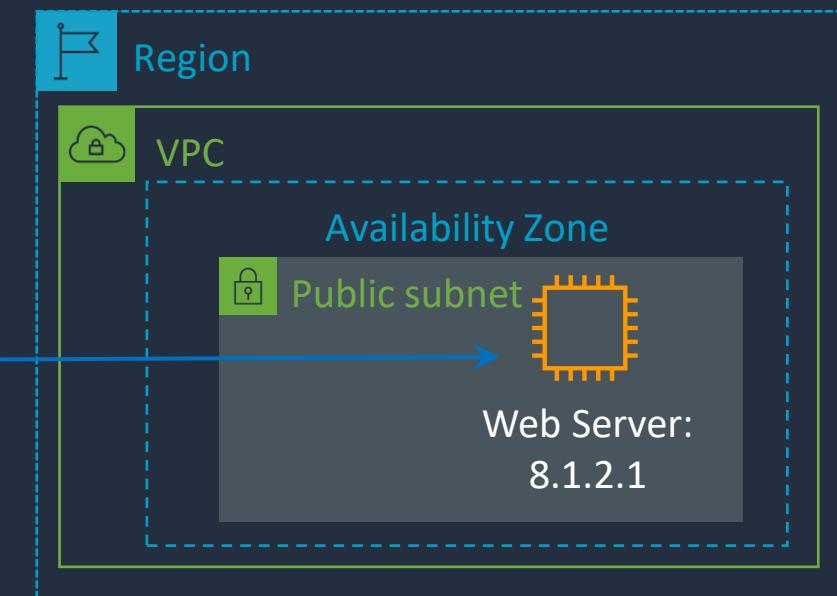




DNS Resolution with Route 53



A **hosted zone** represents a set of records belonging to a domain



Route 53 Routing Policies

Routing Policy	What it does
Simple	Simple DNS response providing the IP address associated with a name
Failover	If primary is down (based on health checks), routes to secondary destination
Geolocation	Uses geographic location client is in (e.g. Europe) to route to the closest region
Geoproximity	Routes to the closest region within a geographic area
Latency	Directs based on the lowest latency route to resources
Multivalue answer	Returns several IP addresses and functions as a basic load balancer
Weighted	Uses the relative weights assigned to resources
IP Based	Route based on the originating IP address of the traffic

Route 53 Routing Policies



Amazon Route 53 Routing Policies

Routing Policy	What it does
Simple	Simple DNS response providing the IP address associated with a name
Failover	If primary is down (based on health checks), routes to secondary destination
Geolocation	Uses geographic location you're in (e.g. Europe) to route you to the closest region
Geoproximity	Routes you to the closest region within a geographic area
Latency	Directs you based on the lowest latency route to resources
Multivalue answer	Returns several IP addresses and functions as a basic load balancer
Weighted	Uses the relative weights assigned to resources to determine which to route to
IP-Based	Uses the IP addresses of clients to make routing decisions



Amazon Route 53 – Simple Routing Policy

Name	Type	Value	TTL
simple.dctlabs.com	A	1.1.1.1	60
		2.2.2.2	
simple2.dctlabs.com	A	3.3.3.3	60



Amazon Route 53

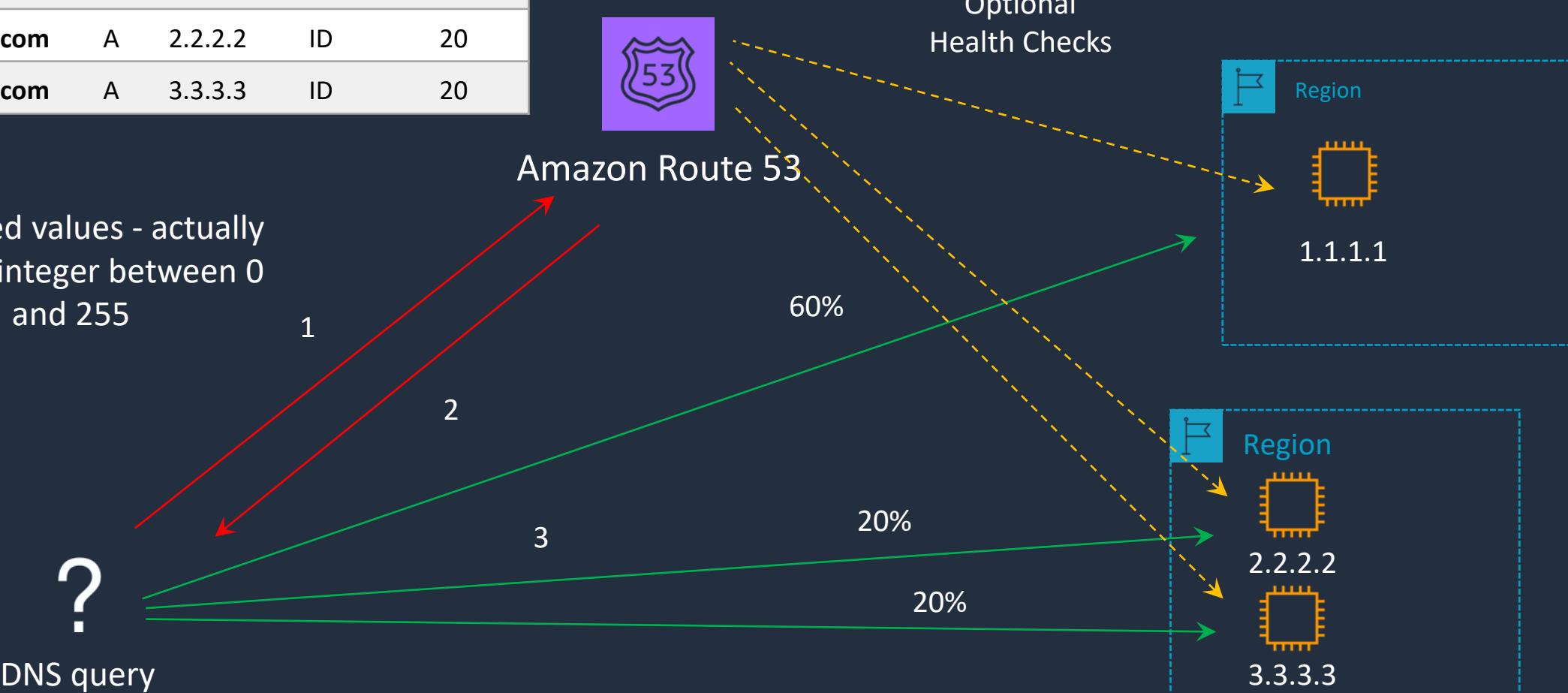




Amazon Route 53 – Weighted Routing Policy

Name	Type	Value	Health	Weight
weighted.dctlabs.com	A	1.1.1.1	ID	60
weighted.dctlabs.com	A	2.2.2.2	ID	20
weighted.dctlabs.com	A	3.3.3.3	ID	20

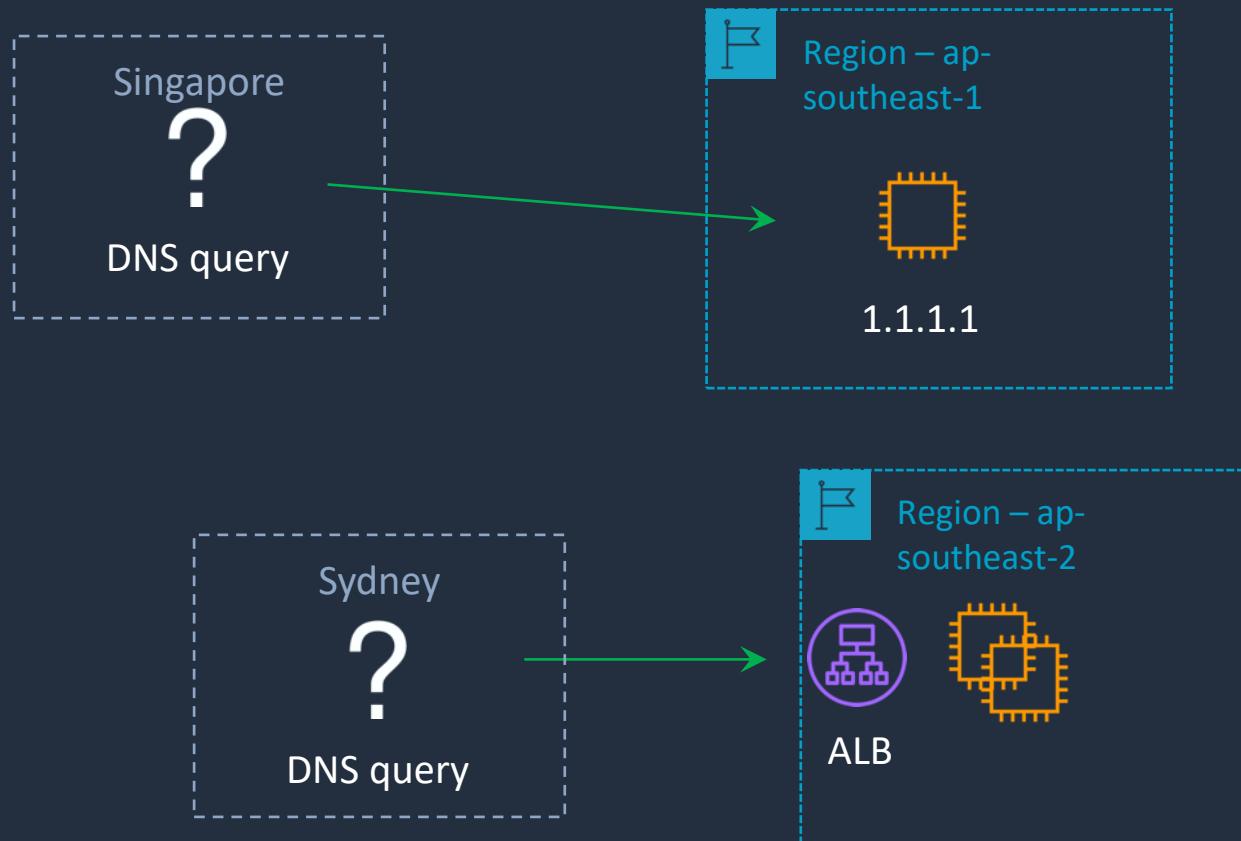
Simplified values - actually uses an integer between 0 and 255





Amazon Route 53 – Latency Routing Policy

Name	Type	Value	Health	Region
latency.dctlabs.com	A	1.1.1.1	ID	ap-southeast-1
latency.dctlabs.com	A	2.2.2.2	ID	us-east-1
latency.dctlabs.com	A	alb-id	ID	ap-southeast-2



Amazon Route 53



Optional
Health Checks





Amazon Route 53 – Failover Routing Policy

Name	Type	Value	Health	Record Type
failover.dctlabs.com	A	1.1.1.1	ID	Primary
failover.dctlabs.com	A	alb-id		Secondary



Health check is
required on Primary

Amazon Route 53

?

DNS query

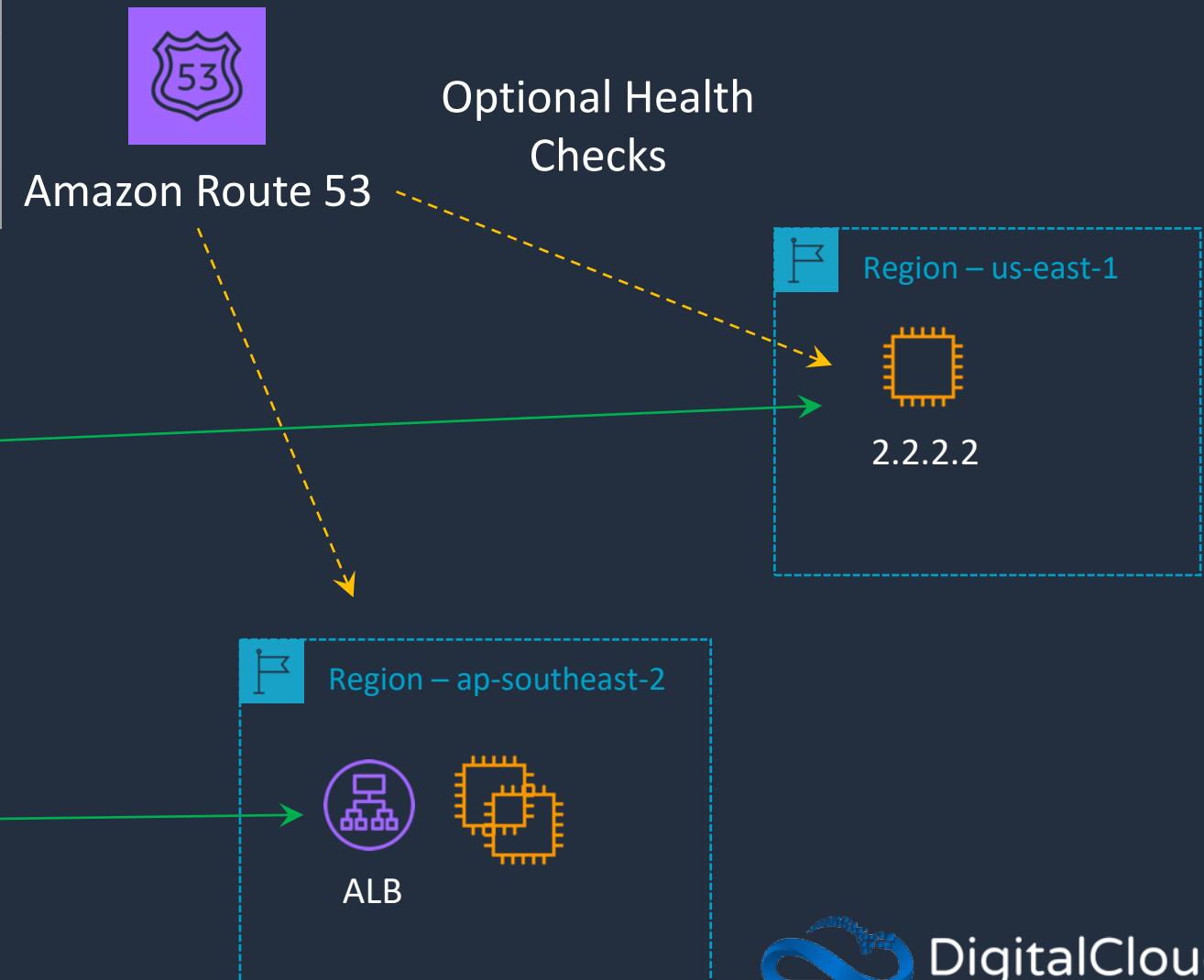


ap-southeast-2 is the
secondary Region



Amazon Route 53 – Geolocation Routing Policy

Name	Type	Value	Health	Geolocation
geolocation.dctlabs.com	A	1.1.1.1	ID	Singapore
geolocation.dctlabs.com	A	2.2.2.2	ID	Default
geolocation.dctlabs.com	A	alb-id	ID	Oceania





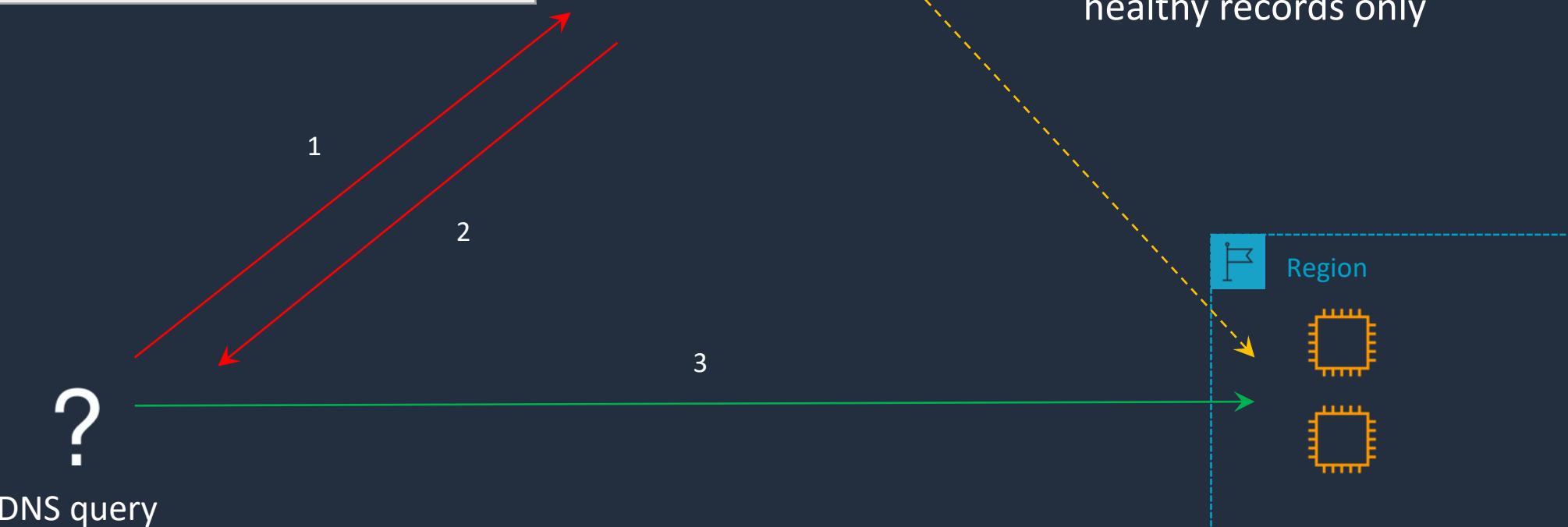
Amazon Route 53 – Multivalue Routing Policy

Name	Type	Value	Health	Multi Value
multivalue.dctlabs.com	A	1.1.1.1	ID	Yes
multivalue.dctlabs.com	A	2.2.2.2	ID	Yes
multivalue.dctlabs.com	A	3.3.3.3	ID	Yes



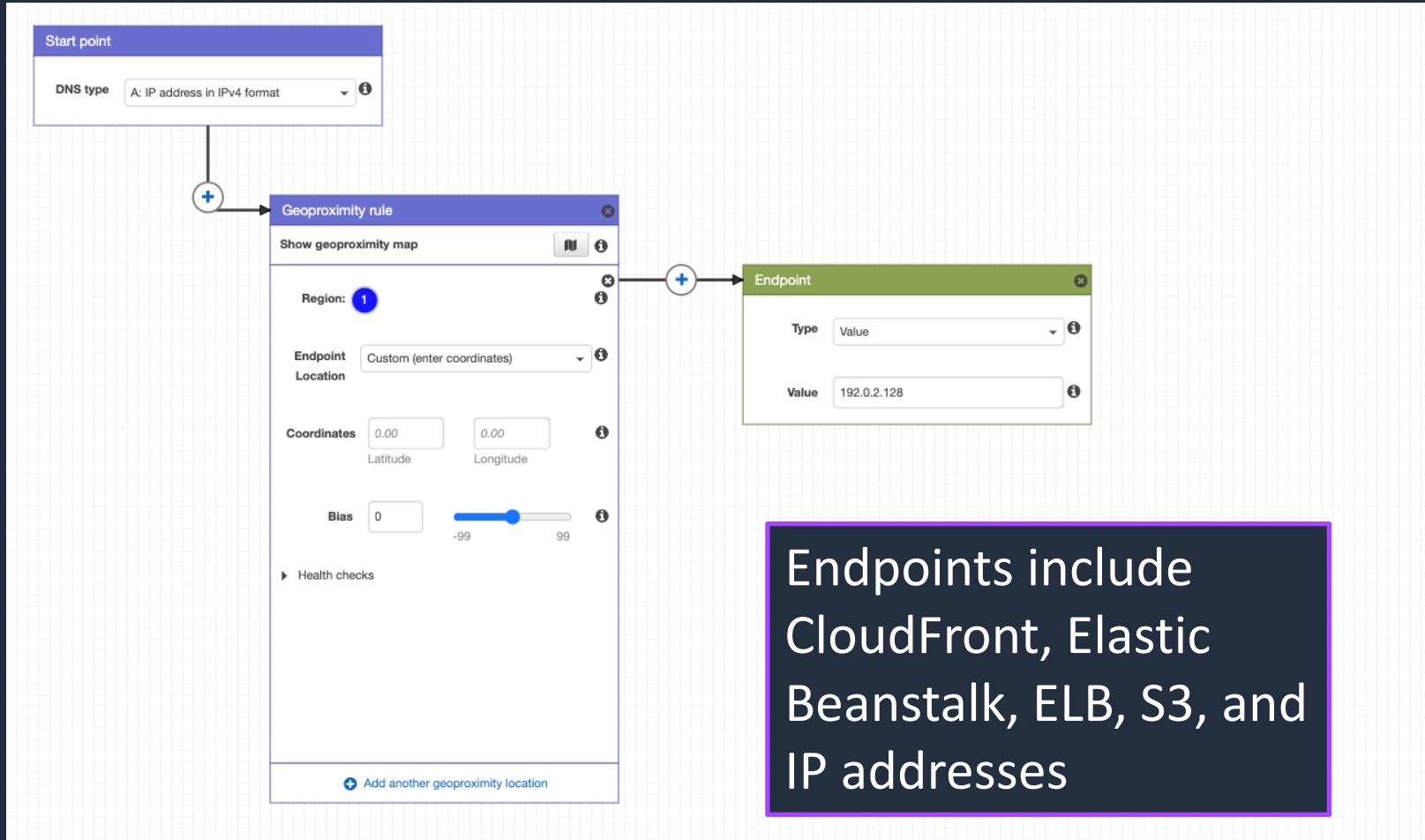
Amazon Route 53

Health Checks: returns healthy records only



Amazon Route 53 – Geoproximity Routing Policy

Must create a **policy** in Traffic Flow



Amazon Route 53 – IP-Based Routing Policy

CIDR collections (2) [Info](#)
A CIDR collection contains locations that hold IP address ranges in CIDR notation.

Collection name	ARN
Europe-CIDR	arn:aws:route53:::cidrcollection/f0c2f216-ef1a-2ab7-7a66-218503ddfe1b
US-CIDR	arn:aws:route53:::cidrcollection/92c3d75b-2a22-7502-cf83-cd0115a7f760

CIDR collections are used to define IP addresses and CIDR blocks

Routing rules can then be created to route based on a **CIDR collection**

▼ Record 1

Record name | [Info](#)
 nealawslabs.link
Keep blank to create a record for the root domain.

Alias

Route traffic to | [Info](#)

Record type | [Info](#)
A – Routes traffic to an IPv4 address and some AWS resources

Routing policy | [Info](#)
 IP-based

USA

Failover Routing Policy with ALB

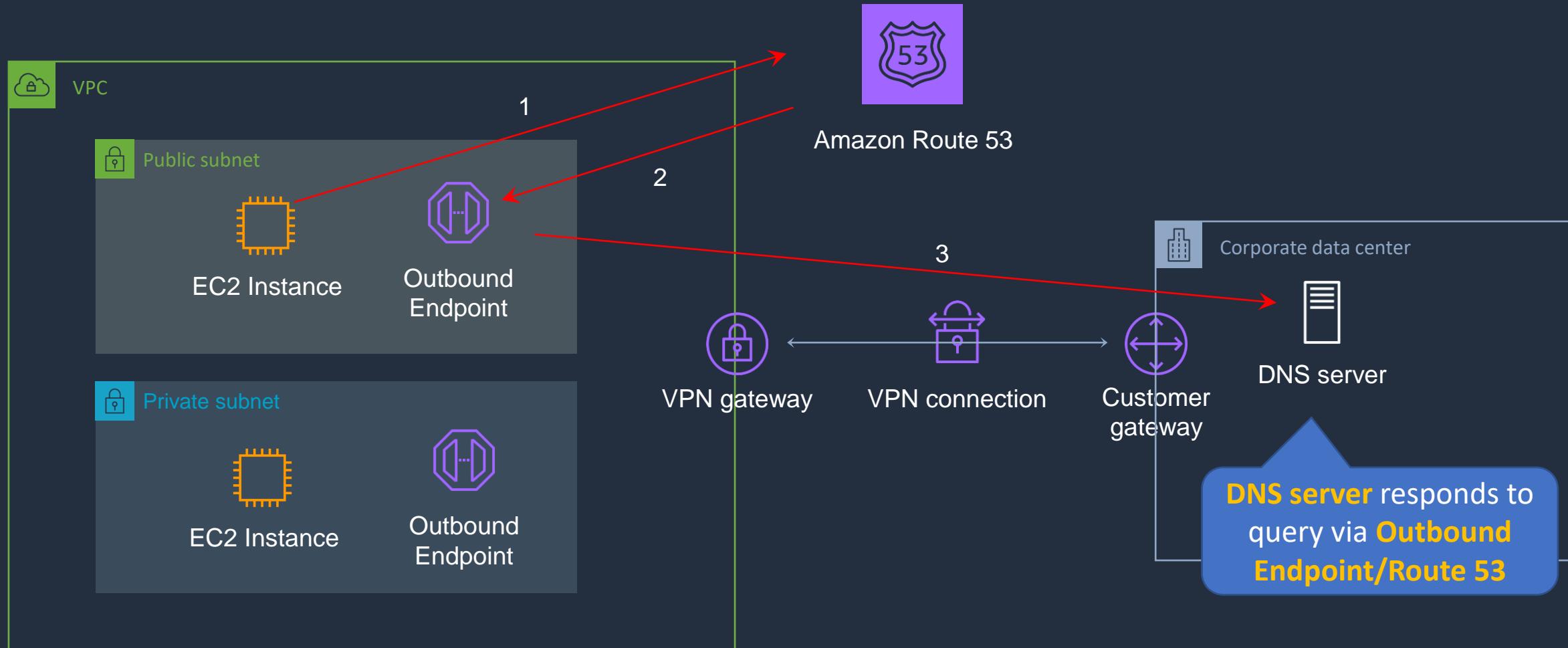


Amazon Route 53 Resolver



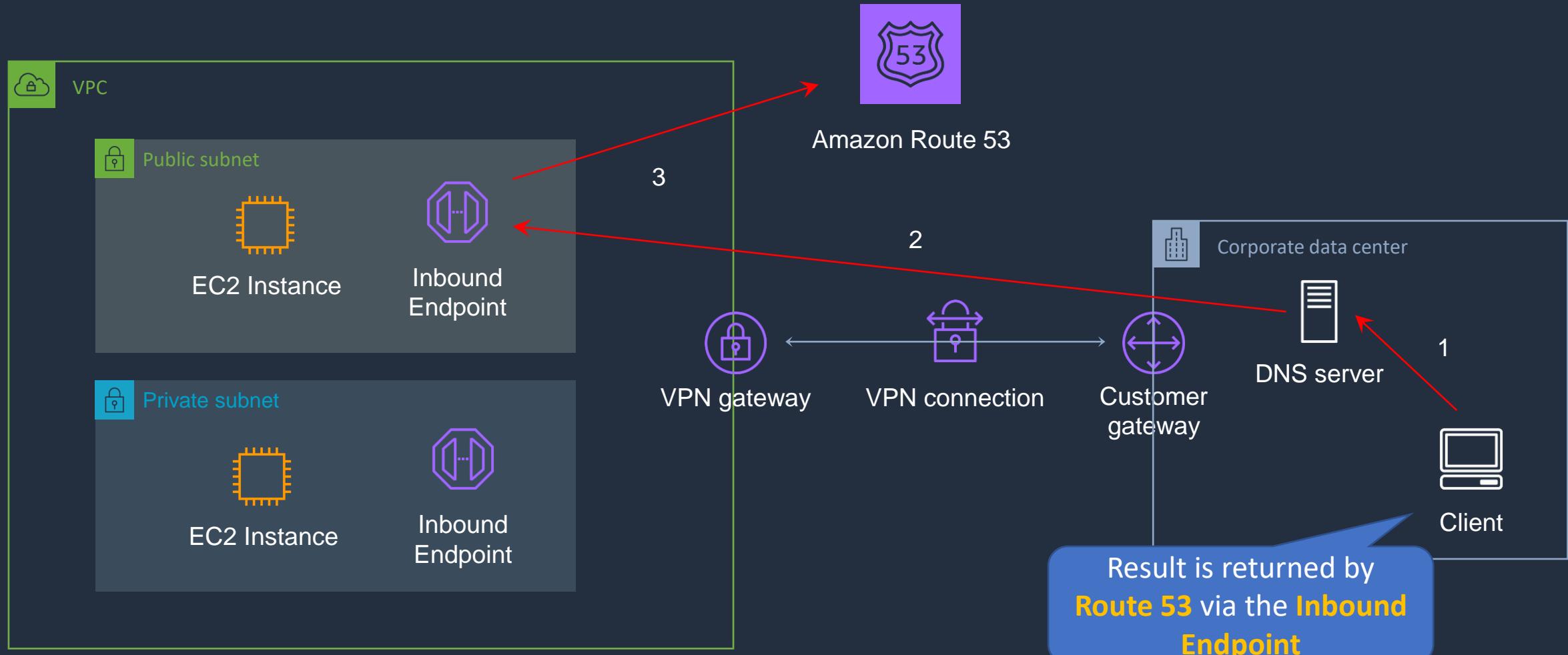


Route 53 Resolver – Outbound Endpoints





Route 53 Resolver – Inbound Endpoints

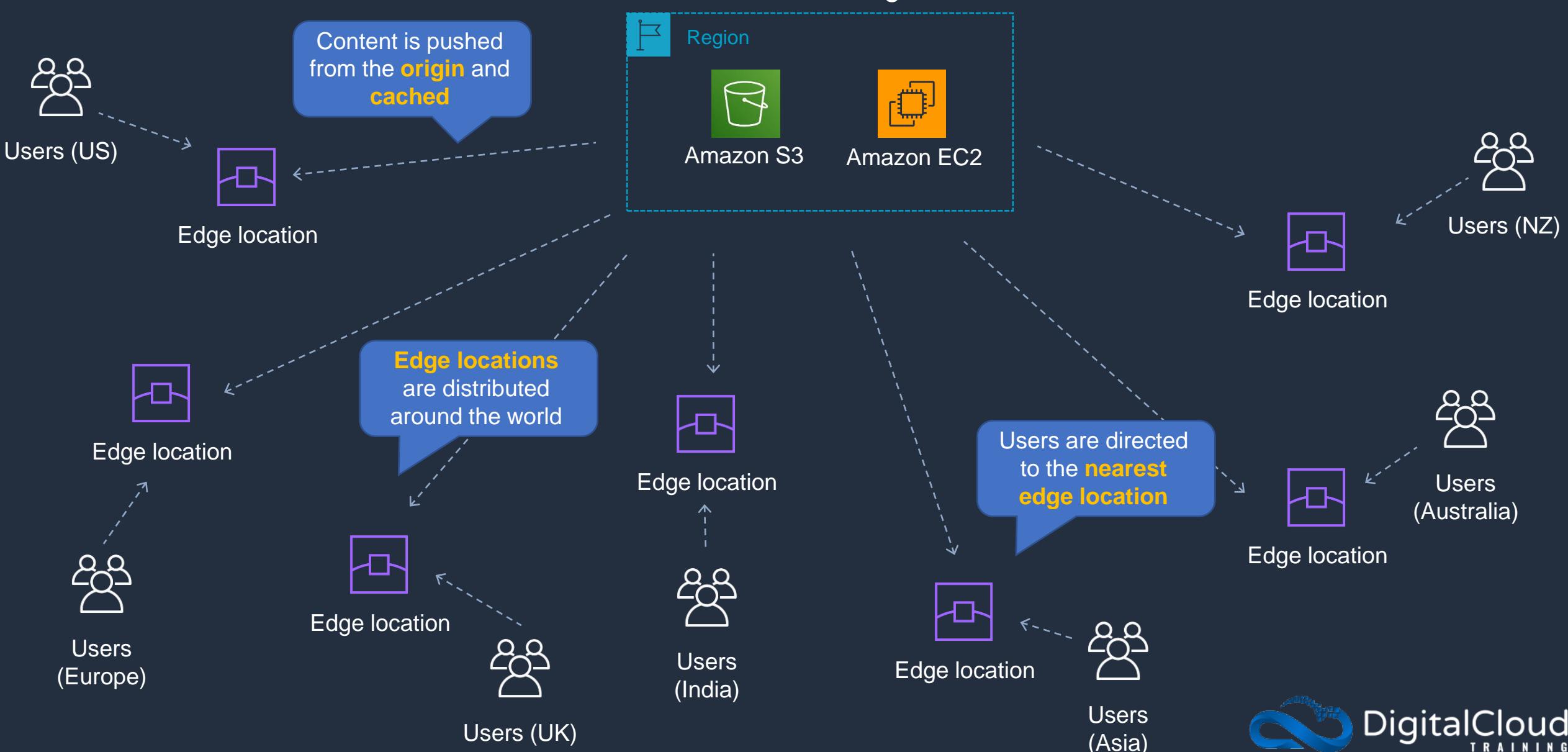


Amazon CloudFront Origins and Distributions





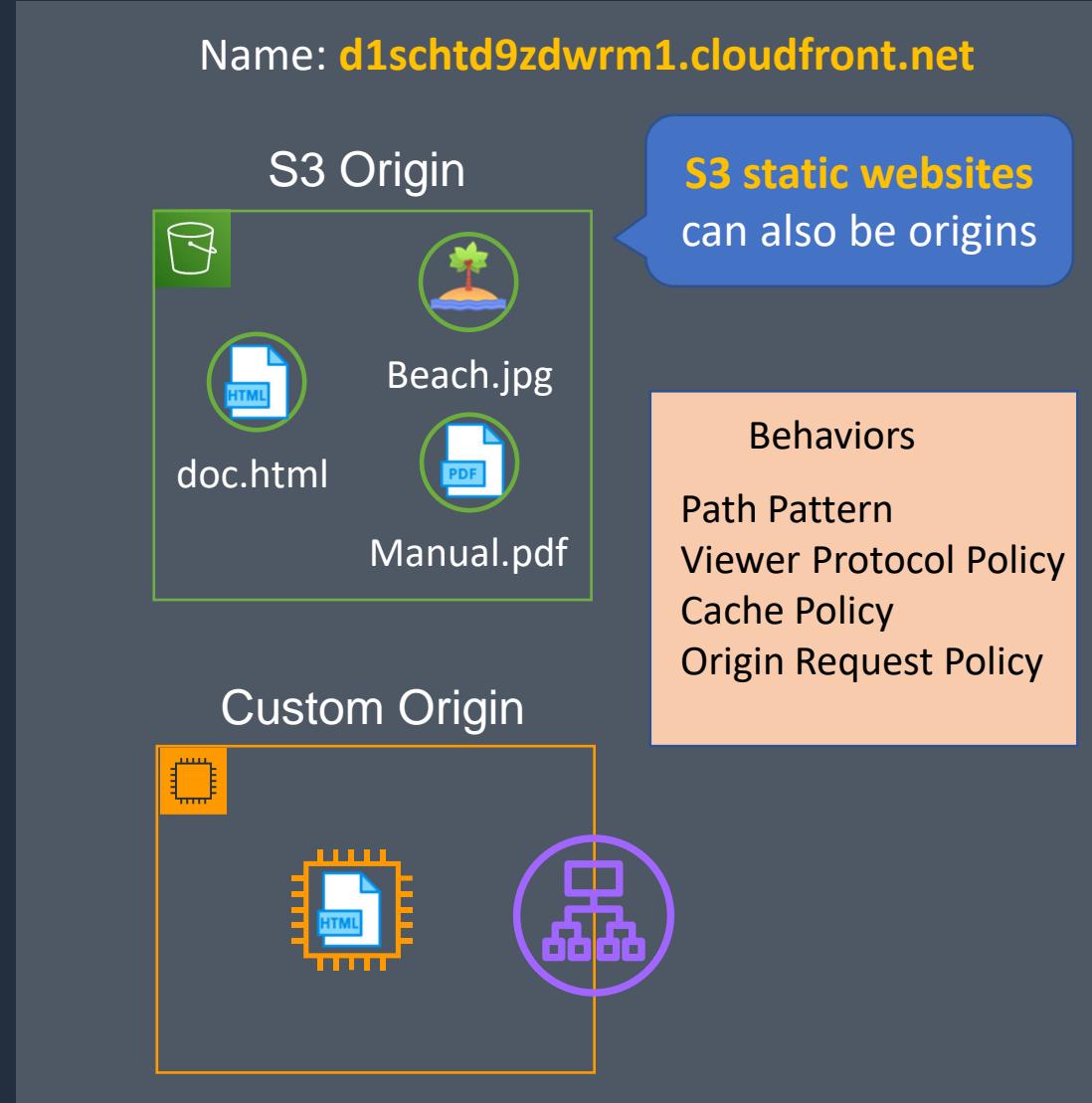
Amazon CloudFront





CloudFront Origins and Distributions

CloudFront Distribution



RTMP distributions were discontinued so only **web distributions** are currently available

CloudFront Web Distribution:

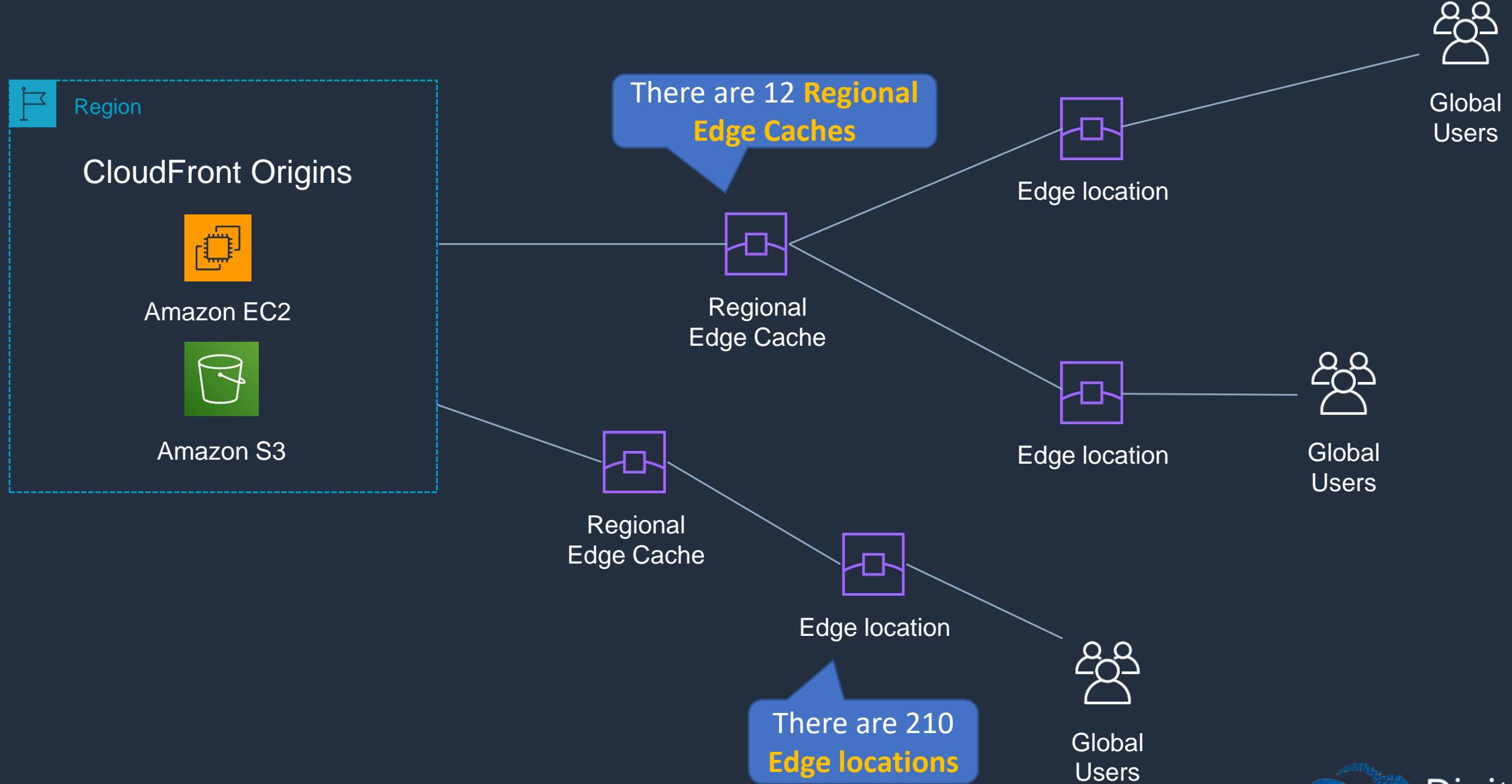
- Speed up distribution of static and dynamic content, for example, .html, .css, .php, and graphics files
- Distribute media files using HTTP or HTTPS
- Add, update, or delete objects, and submit data from web forms
- Use live streaming to stream an event in real time

Amazon CloudFront Caching and Behaviors



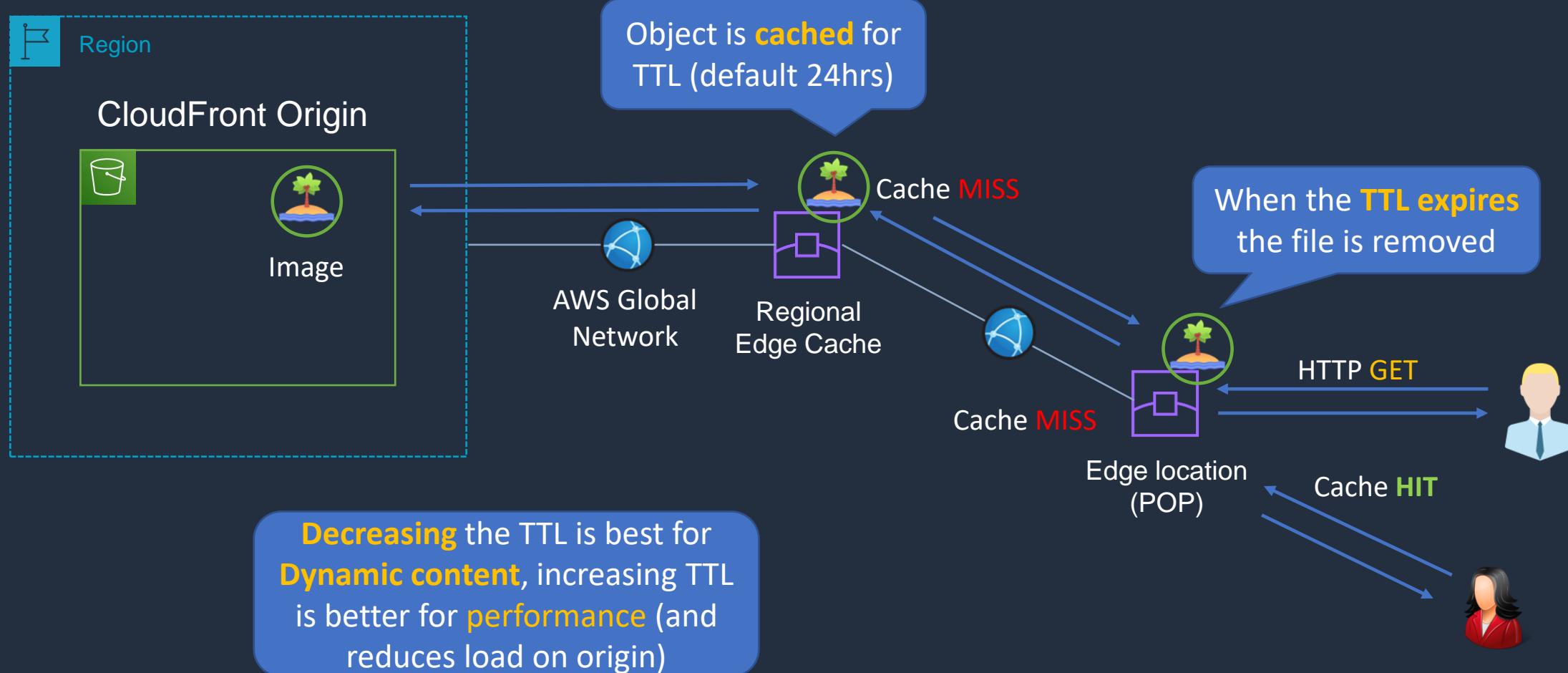


Amazon CloudFront Caching





Amazon CloudFront Caching





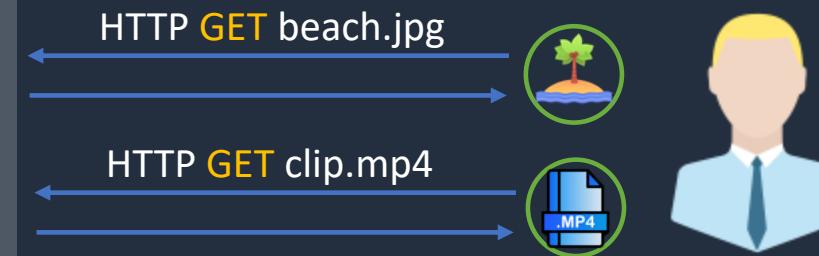
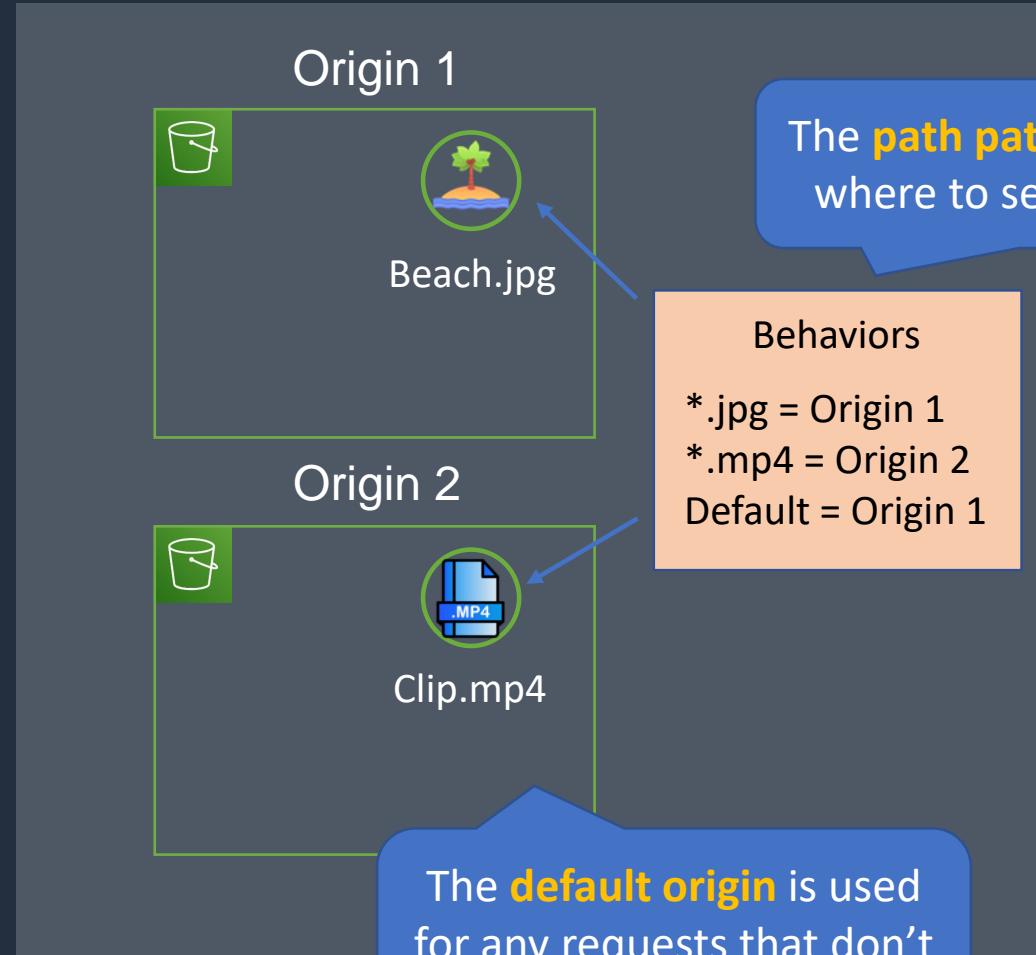
Amazon CloudFront Caching

- You can define a maximum **Time To Live** (TTL) and a default TTL
- TTL is defined at the **behavior** level
- This can be used to define different TTLs for different file types (e.g. png vs jpg)
- After expiration, CloudFront checks the origin for any new requests (check the file is the latest version)
- Headers can be used to control the cache:
 - **Cache-Control max-age=(seconds)** - specify how long before CloudFront gets the object again from the origin server
 - **Expires** – specify an expiration date and time



CloudFront Path Patterns

CloudFront Distribution



Precedence	Path Pattern	Origin or Origin Group	Viewer Protocol Policy	Cache Policy Name
0	*.jpg	Origin 1	HTTP and HTTPS	Managed-CachingOptimized
1	*.mp4	Origin 2	HTTP and HTTPS	Managed-CachingOptimized
2	Default (*)	Origin 1	HTTP and HTTPS	Managed-CachingOptimized



Caching Based on Request Headers

- You can configure CloudFront to forward **headers** in the **viewer request** to the origin
- CloudFront can then cache multiple versions of an object based on the values in one or more request headers
- Controlled in a behavior to do one of the following:
 - Forward all headers to your origin (objects are **not cached**)
 - Forward a whitelist of headers that you specify
 - Forward only the default headers (doesn't cache objects based on values in request headers)

CloudFront Signed URLs and OAI/OAC



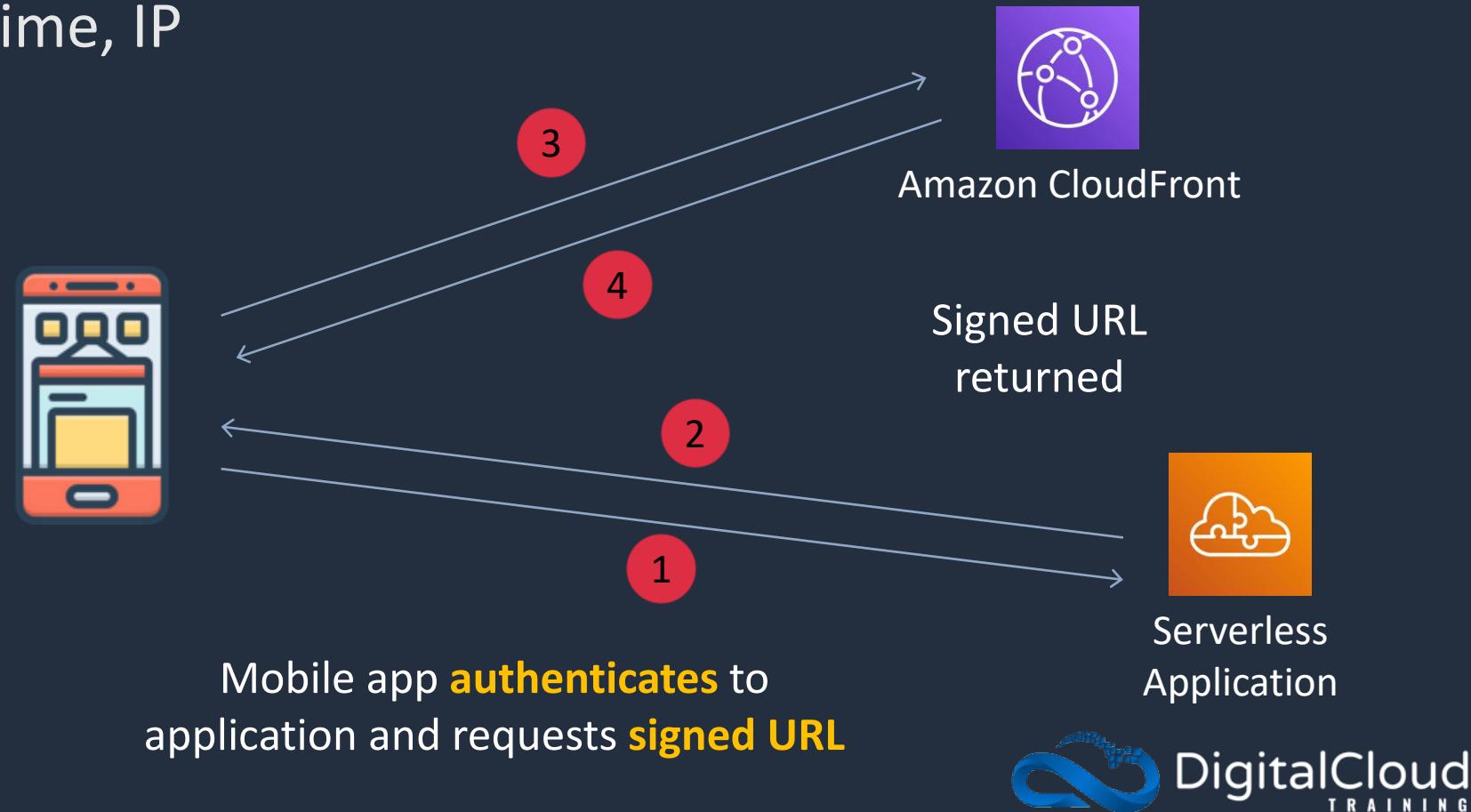


CloudFront Signed URLs

- Signed URLs provide more control over access to content
- Can specify beginning and expiration date and time, IP addresses

Mobile app uses **signed URL** to access distribution

Signed URLs should be used for **individual files** and clients that don't support **cookies**





CloudFront Signed Cookies

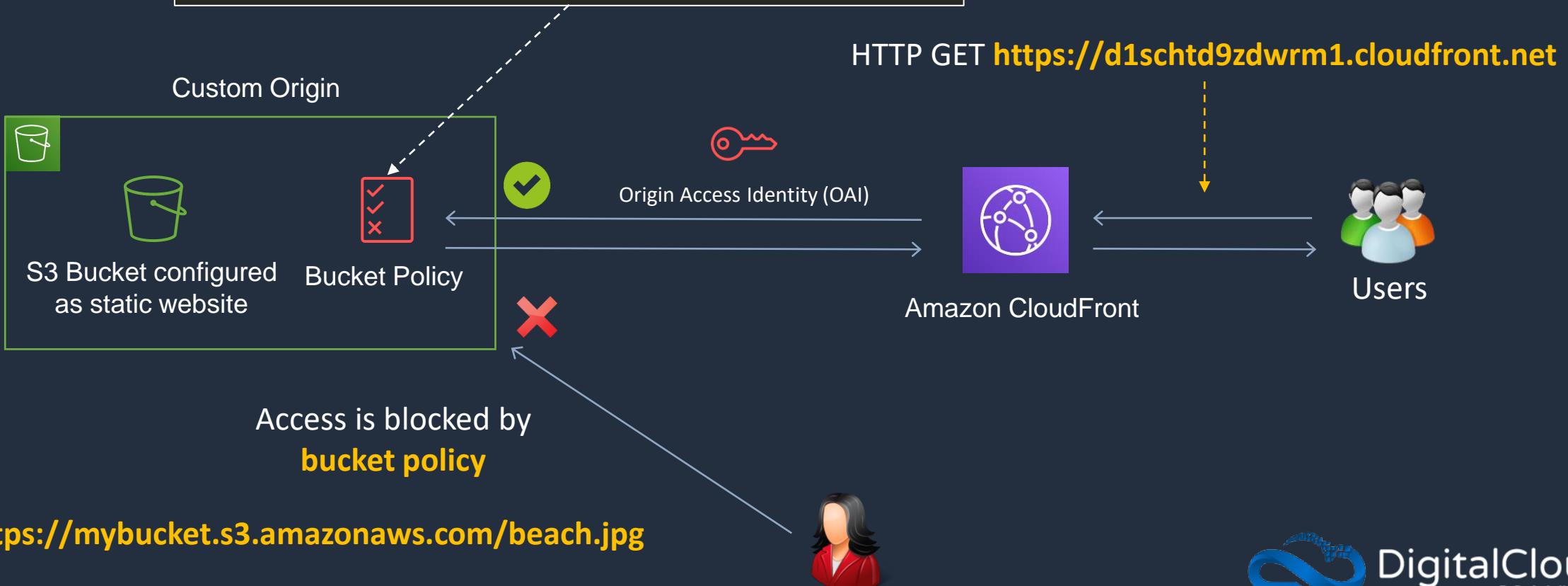
- Similar to Signed URLs
- Use signed cookies when you don't want to change URLs
- Can also be used when you want to provide access to **multiple restricted files** (Signed URLs are for individual files)



CloudFront Origin Access Identity (OAI)

```
{  
    "Version": "2008-10-17",  
    "Id": "PolicyForCloudFrontPrivateContent",  
    "Statement": [  
        {  
            "Sid": "1",  
            "Effect": "Allow",  
            "Principal": {  
                "AWS": "arn:aws:iam::cloudfront:user/CloudFront Origin Access Identity E11A2JL2H306JJ"  
            },  
            "Action": "s3:GetObject",  
            "Resource": "arn:aws:s3:::mybucket/*"  
        }  
    ]  
}
```

The policy restricts access to the **OAI**





CloudFront Origin Access Control (OAC)

- Like a OAI but supports additional use cases
- AWS recommend using the OAC instead of an OAI
- Requires an S3 bucket policy that allows the CloudFront service principal

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowCloudFrontServicePrincipalReadOnly",  
            "Effect": "Allow",  
            "Principal": {  
                "Service": "cloudfront.amazonaws.com"  
            },  
            "Action": "s3:GetObject",  
            "Resource": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*",  
            "Condition": {  
                "StringEquals": {  
                    "AWS:SourceArn": "arn:aws:cloudfront::111122223333:distribution/EDFDVBD6EXAMPLE"  
                }  
            }  
        }  
    ]  
}
```

CloudFront Cache and Behavior Settings



CloudFront SSL/TLS and SNI

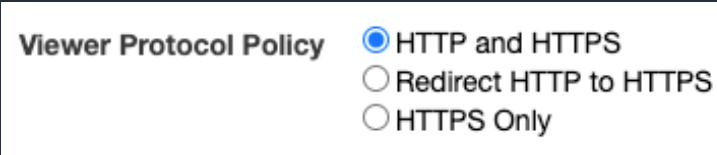




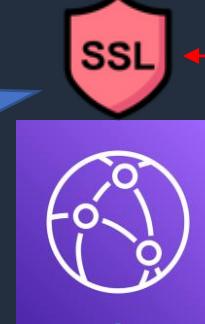
CloudFront SSL/TLS



Viewer Protocol



Certificate can be **ACM** or a trusted **third-party CA**



For CloudFront certificate must be issued in **us-east-1**



AWS Certificate Manager

Default CF **domain name** can be changed using **CNAMEs**

S3 has its **own** certificate (can't be changed)



Origin Protocol

Origin certificates must be **public certificates**



Certificate can be **ACM** (ALB) or **third-party** (EC2)



CloudFront Server Name Indication (SNI)



HTTP GET: <https://mypublicdomain.com>



HTTP GET: <https://myotherdomain.com>

Note: SNI works with
browsers/clients released
after 2010 – otherwise
need **dedicated IP**

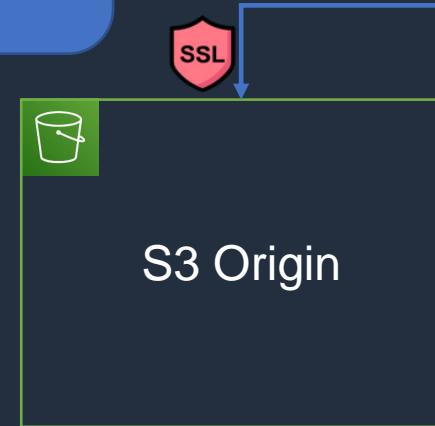
Name: myotherdomain.com



Request URL includes
domain name which
matches certificate

Name: mypublicdomain.com

Multiple certificates
share the **same IP**
with SNI



S3 Origin



Custom Origin

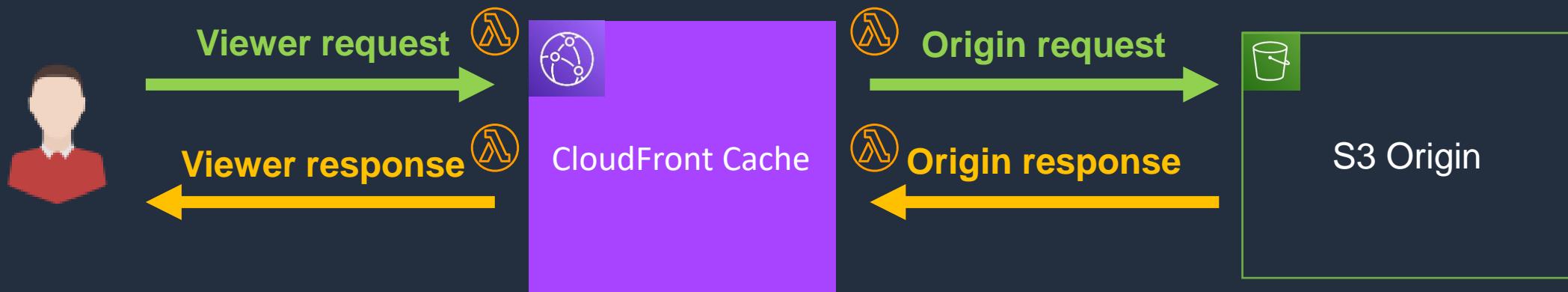
Lambda@Edge





Lambda@Edge

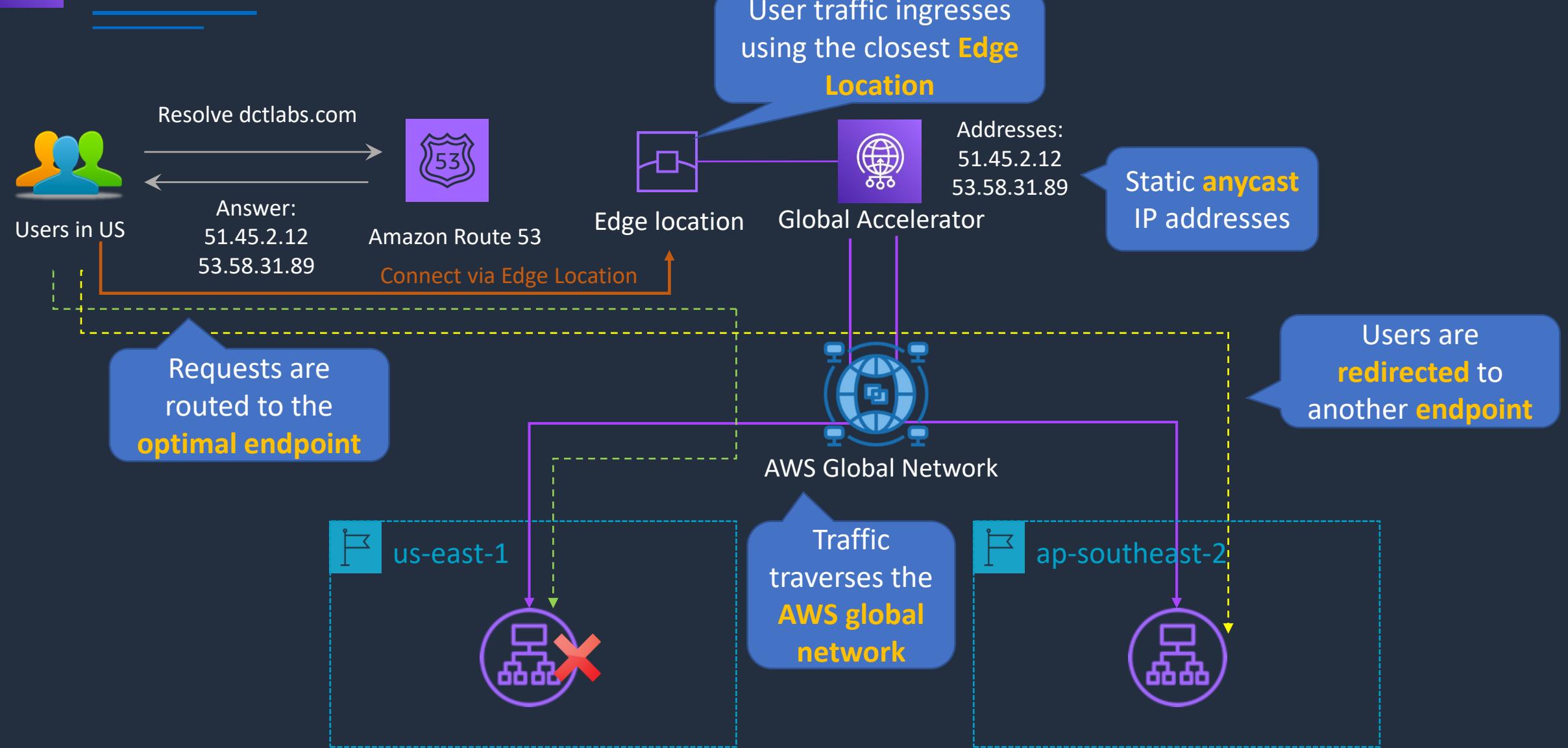
- Run Node.js and Python Lambda functions to customize the content CloudFront delivers
- Executes functions closer to the viewer
- Can be run at the following points
 - After CloudFront receives a request from a viewer (viewer request)
 - Before CloudFront forwards the request to the origin (origin request)
 - After CloudFront receives the response from the origin (origin response)
 - Before CloudFront forwards the response to the viewer (viewer response)



AWS Global Accelerator



AWS Global Accelerator



Architecture Patterns – DNS, Caching and Performance Optimization





Requirement

An Elastic Load Balancer must be resolvable using a company's public domain name. A Route 53 hosted zone exists

A website runs across two AWS Regions. All traffic goes to one Region and should be redirected only if the website is unavailable

Websites run in several countries and distribution rights require restricting access to content based on the geographic source of the connection

Solution

Create an Alias record that maps the domain name to the ELB

Create a failover routing policy in AWS Route 53 and configure health checks on the primary

Use AWS Route 53 geolocation routing and restrict distribution based on geographic location



Requirement

A CloudFront distribution has multiple S3 origins. Requests should be served from different origins based on file type being requested

Content is accessed using an application and CloudFront distribution. Need to control access to multiple files on the distribution

Application runs behind an Application Load Balancer in multiple Regions. Need to intelligently route traffic based on latency and availability

Solution

Modify the CloudFront behavior and configure a path pattern

Configure signed cookies and update the application

Create an AWS Global Accelerator and add the ALBs

SECTION 9

Block and File Storage

Block vs File vs Object Storage





Block Storage



Hard Disk Drive (HDD)

- Also known as magnetic drives
- Older technology
- Much slower than SSD
- Much cheaper than SSD



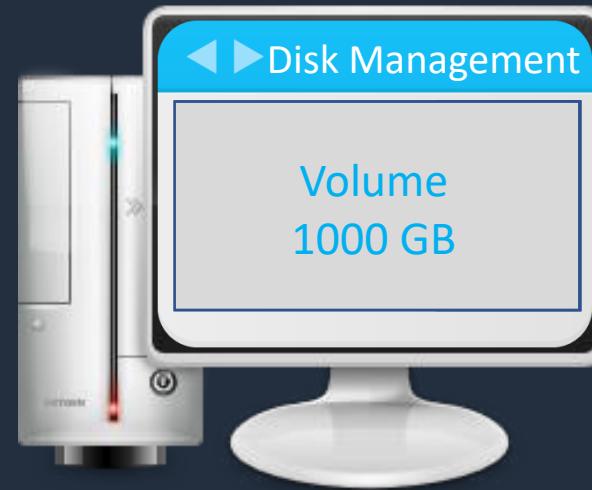
Solid State Drive (SSD)

- Uses flash memory
- Newer technology
- MUCH faster than HDD
- More expensive than HDD



Block Storage

Hard drives are
block-based
storage systems



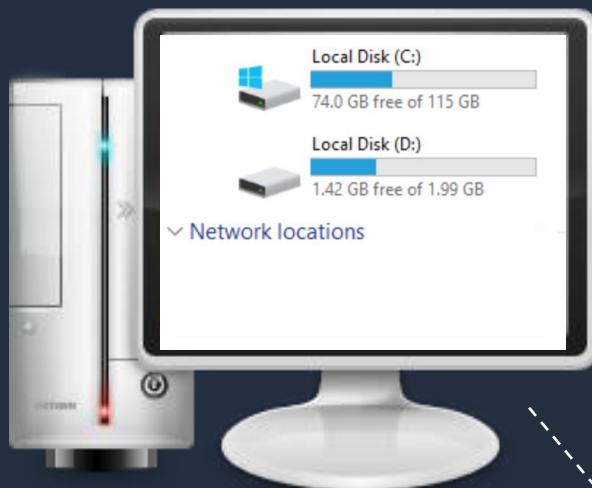
Hard drives are block-based storage systems

The Operating System
(OS) sees a volume. A
volume can be
partitioned and
formatted

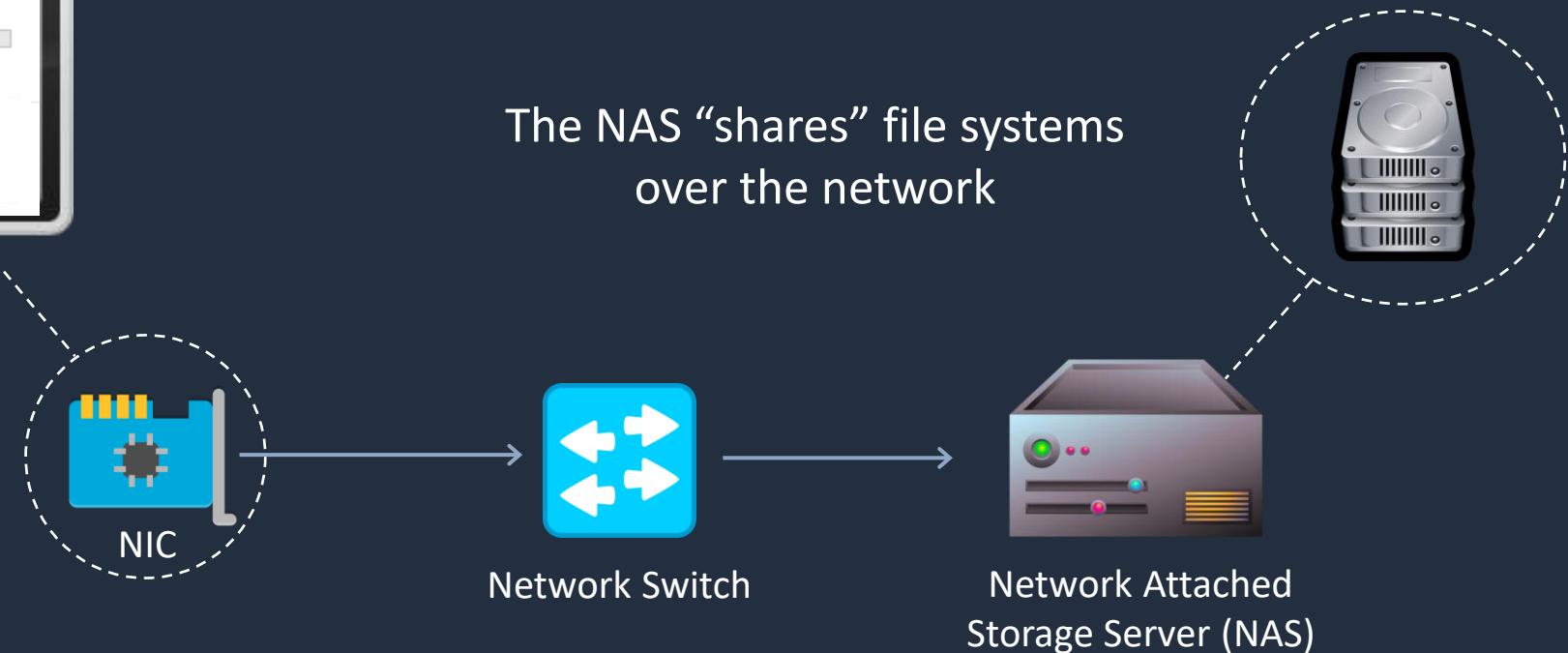


File Storage

The Operating System (OS) sees a **file system** that is mapped to a local drive letter



The NAS “shares” file systems over the network



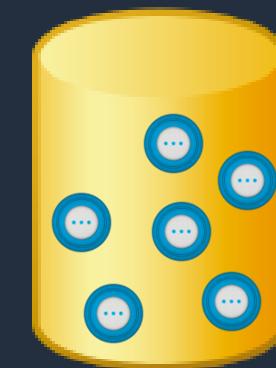
Object Storage Systems

User uploads **objects** using a web browser



The **HTTP protocol** is used with a **REST API** (e.g. GET, PUT, POST, SELECT, DELETE)

There is no hierarchy of objects in the container



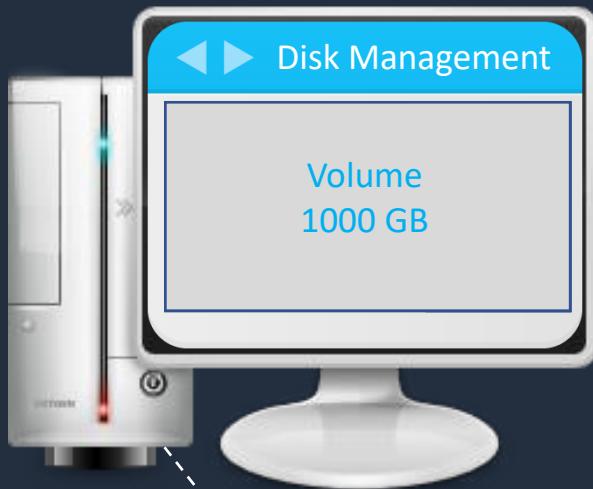
Object Storage Container

Objects can be files, videos, images etc.

Block vs File vs Object Storage

The OS sees **volumes** that can be partitioned and formatted

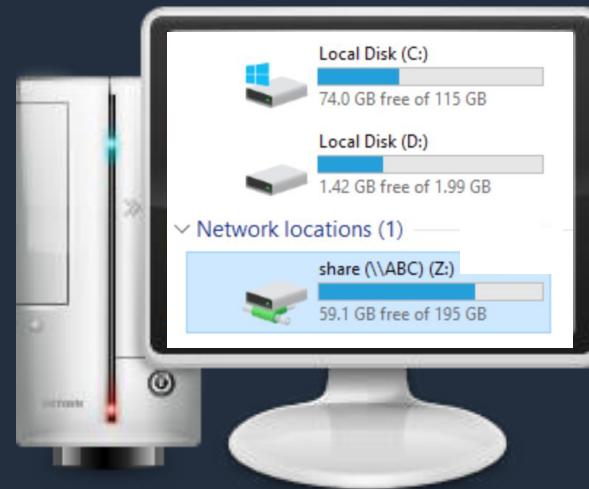
Block Storage



The OS reads/writes at the **block level**. Disks can be internal, or network attached

A filesystem can be shared by many users

File Storage



A filesystem is “**mounted**” to the OS using a network share

Massively scalable and low cost

Object Storage

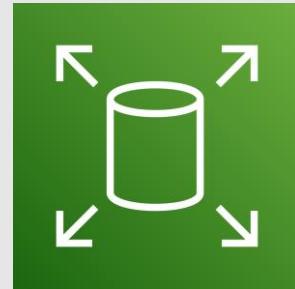


Object Storage Container

There is **no hierarchy** of objects in the container

Cannot be mounted but can be accessed programmatically using the **REST API**

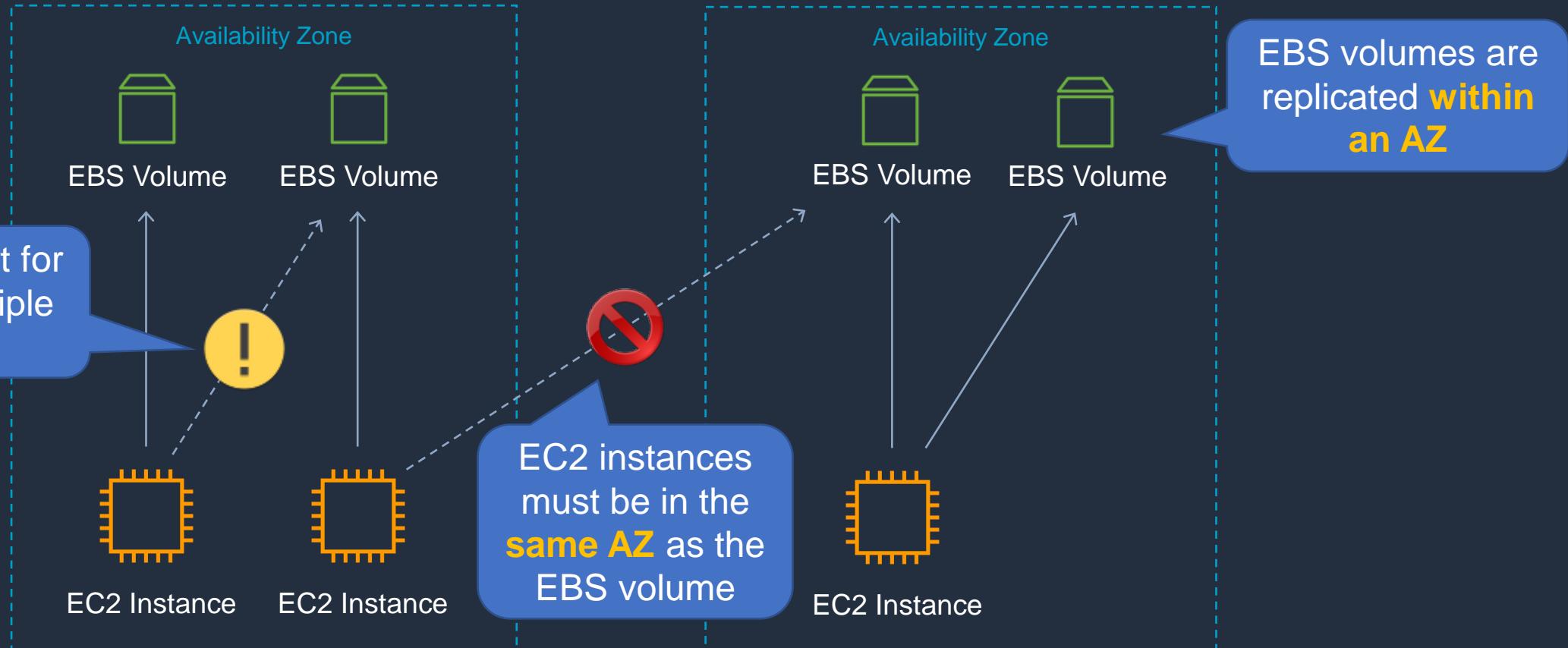
Amazon EBS Deployment and Volume Types



Amazon EBS Deployment



Amazon Elastic Block Store (EBS)





Amazon EBS Multi-Attach



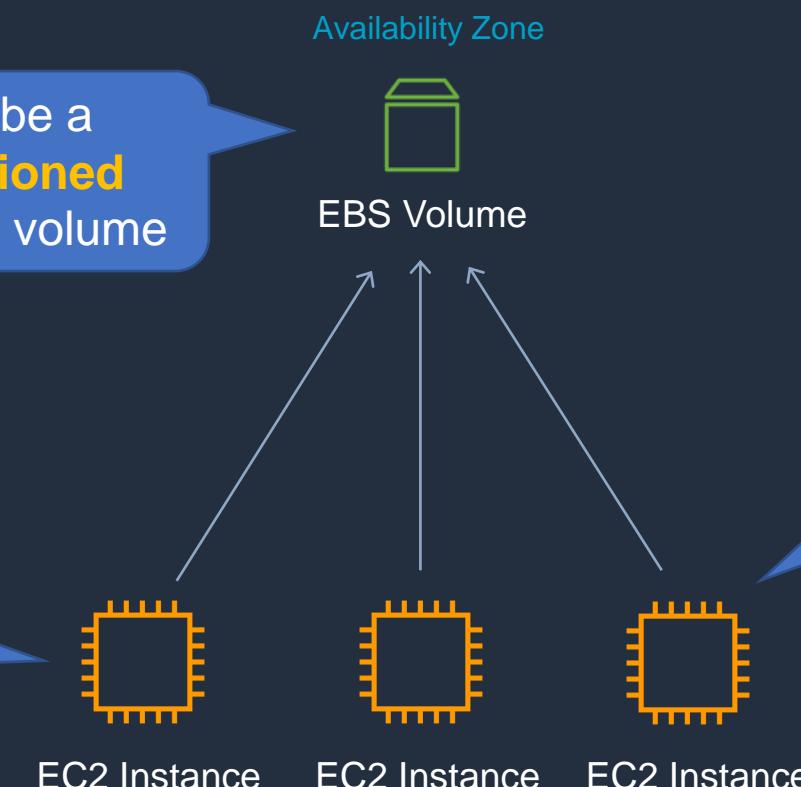
May not be on the exam yet

Must be within a **single AZ**

Must be a **Provisioned IOPS io1 volume**

Available for **Nitro system-based EC2 instances**

Up to **16 instances** can be attached to a single volume





Amazon EBS SSD-Backed Volumes

New and **may not** be on the exam yet

New and **may not** be on the exam yet

	General Purpose SSD		Provisioned IOPS SSD		
Volume type	gp3	gp2	io2 Block Express ‡	io2	io1
Durability	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	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)
Use cases	<ul style="list-style-type: none">Low-latency interactive appsDevelopment and test environments		Workloads that require sub-millisecond latency, and sustained IOPS performance or more than 64,000 IOPS or 1,000 MiB/s of throughput		<ul style="list-style-type: none">Workloads that require sustained IOPS performance or more than 16,000 IOPSI/O-intensive database workloads
Volume size	1 GiB - 16 TiB		4 GiB - 64 TiB	4 GiB - 16 TiB	
Max IOPS per volume (16 KiB I/O)	16,000		256,000	64,000 †	
Max throughput per volume	1,000 MiB/s	250 MiB/s *	4,000 MiB/s	1,000 MiB/s †	
Amazon EBS Multi-attach	Not supported		Supported		
Boot volume	Supported				



Amazon EBS HDD-Backed Volumes

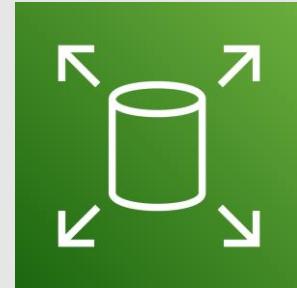
	Throughput Optimized HDD	Cold HDD
Volume type	st1	sc1
Durability	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)
Use cases	<ul style="list-style-type: none">• Big data• Data warehouses• Log processing	<ul style="list-style-type: none">• Throughput-oriented storage for data that is infrequently accessed• Scenarios where the lowest storage cost is important
Volume size	125 GiB - 16 TiB	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	Not supported
Boot volume	Not supported	Not supported



Amazon EBS

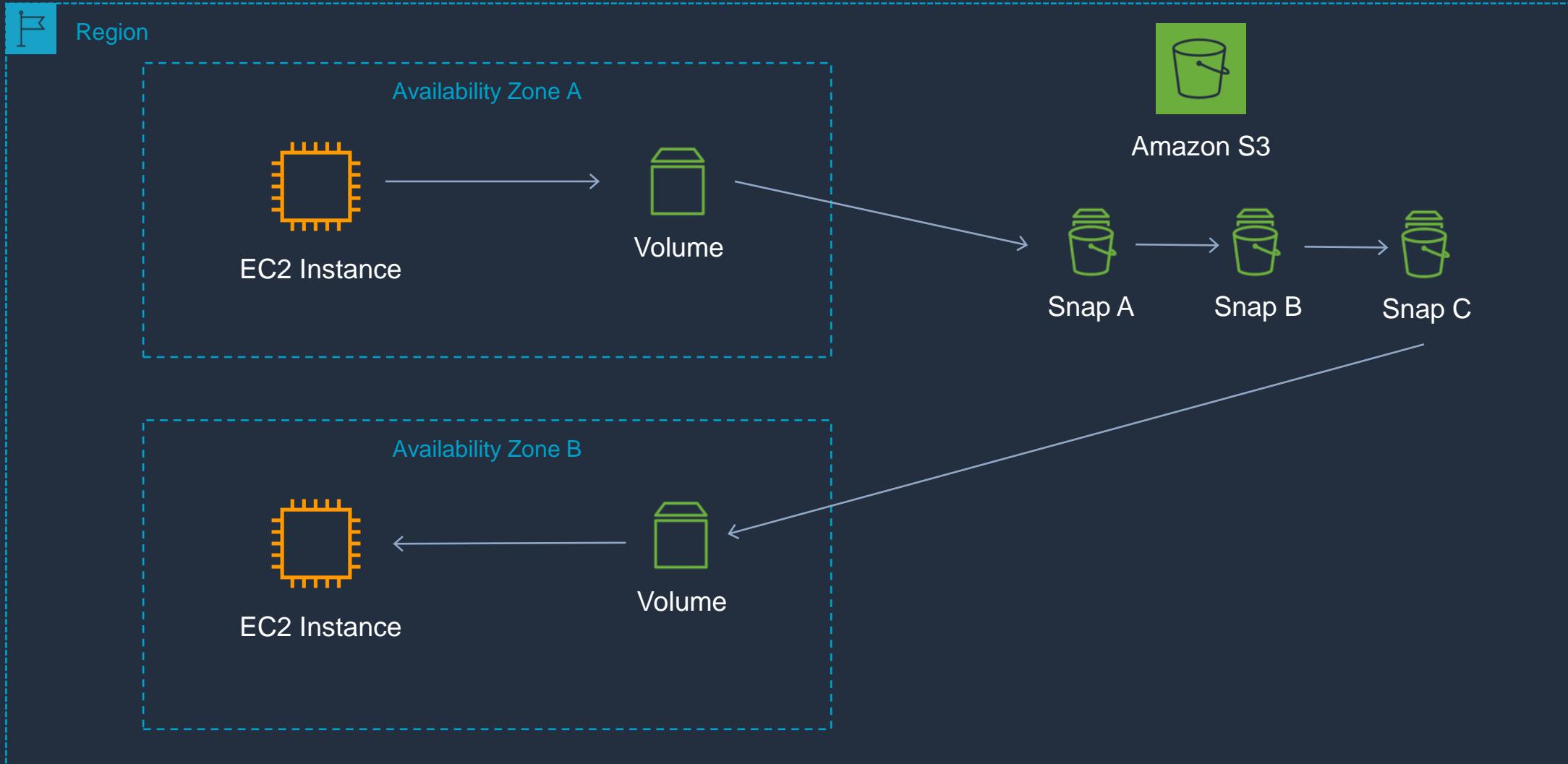
- EBS volume data persists **independently** of the life of the instance
- EBS volumes do not need to be attached to an instance
- You can attach multiple EBS volumes to an instance
- You can use multi-attach to attach a volume to multiple instances but with some constraints
- EBS volumes must be in the **same AZ** as the instances they are attached to
- Root EBS volumes **are deleted** on termination by default
- Extra non-boot volumes **are not deleted** on termination by default

Amazon EBS Copying, Sharing and Encryption



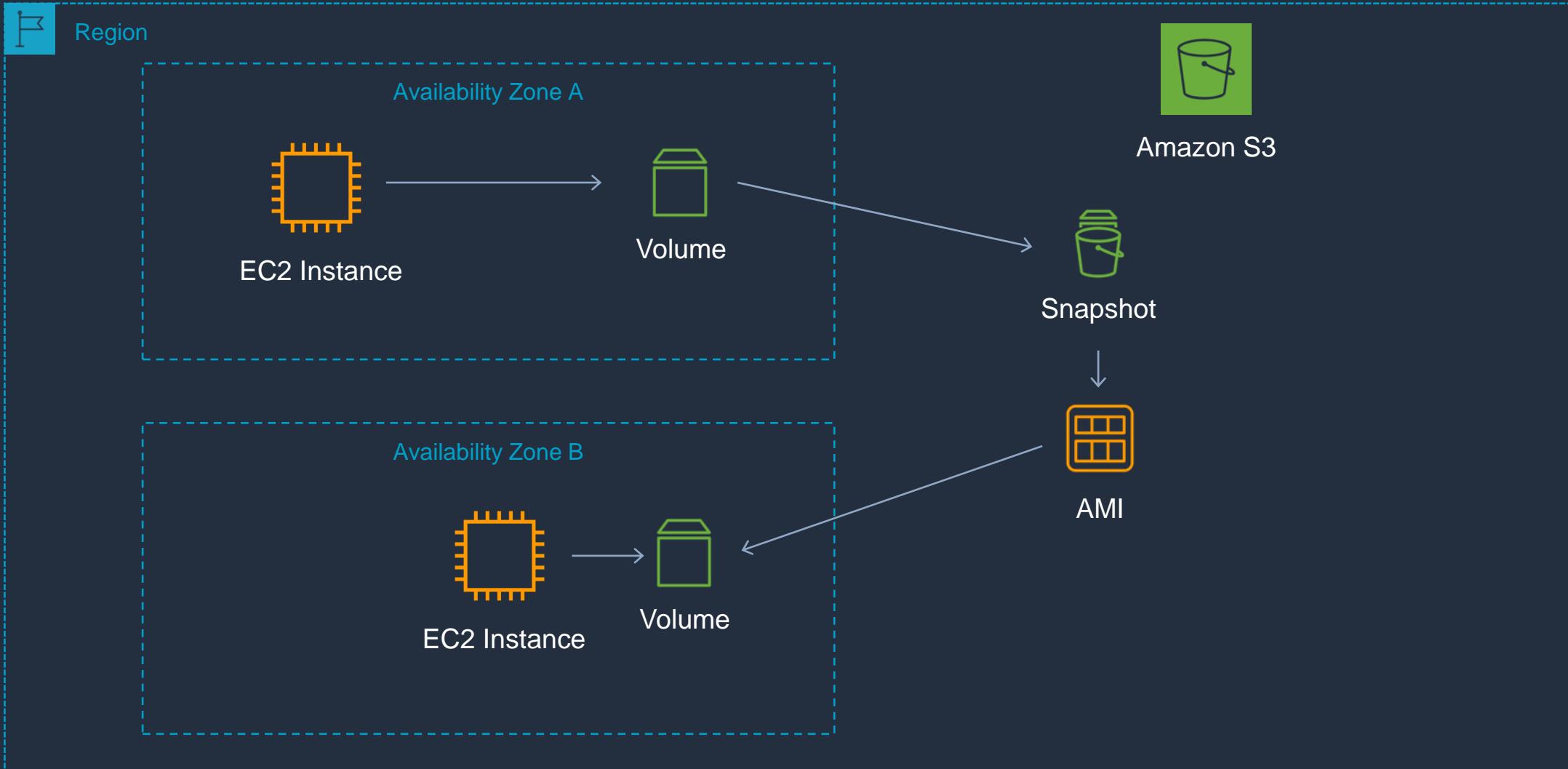


Amazon EBS Copying, Sharing and Encryption





Take Snapshot, Create AMI, Launch New Instance

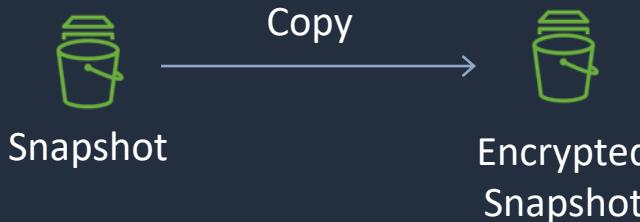




Copying and Sharing AMIs and Snapshots



- Encryption state retained
- Same region



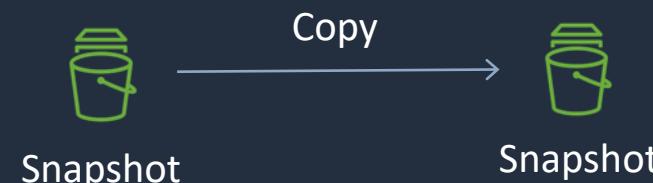
- Can be encrypted
- Can change regions



- Can be encrypted
- Can change AZ



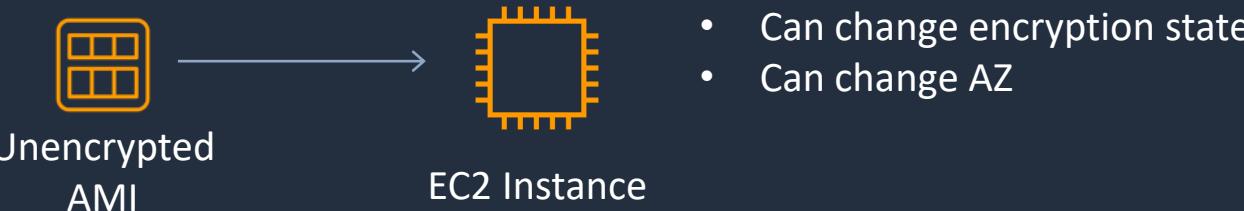
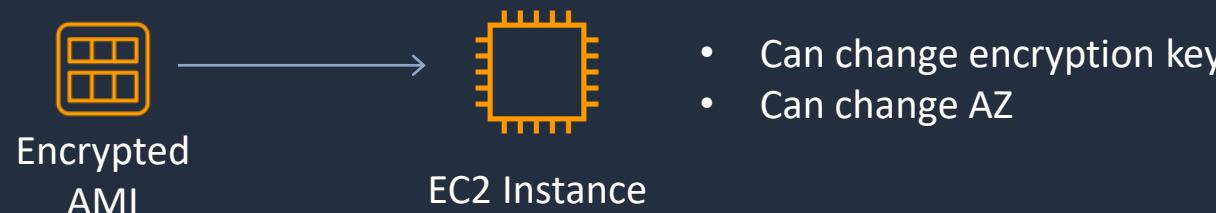
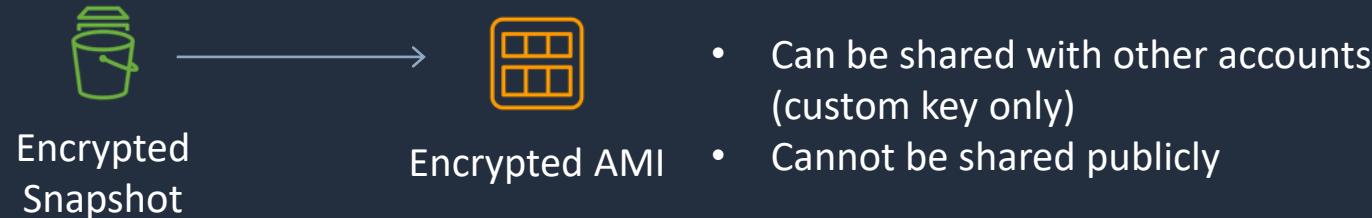
- Cannot be encrypted
- Can be shared with other accounts
- Can be shared publicly



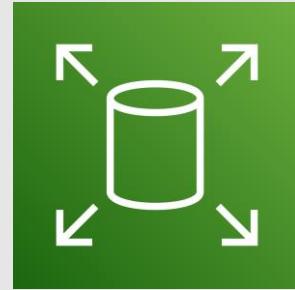
- Can change encryption key
- Can change regions



Copying and Sharing AMIs and Snapshots

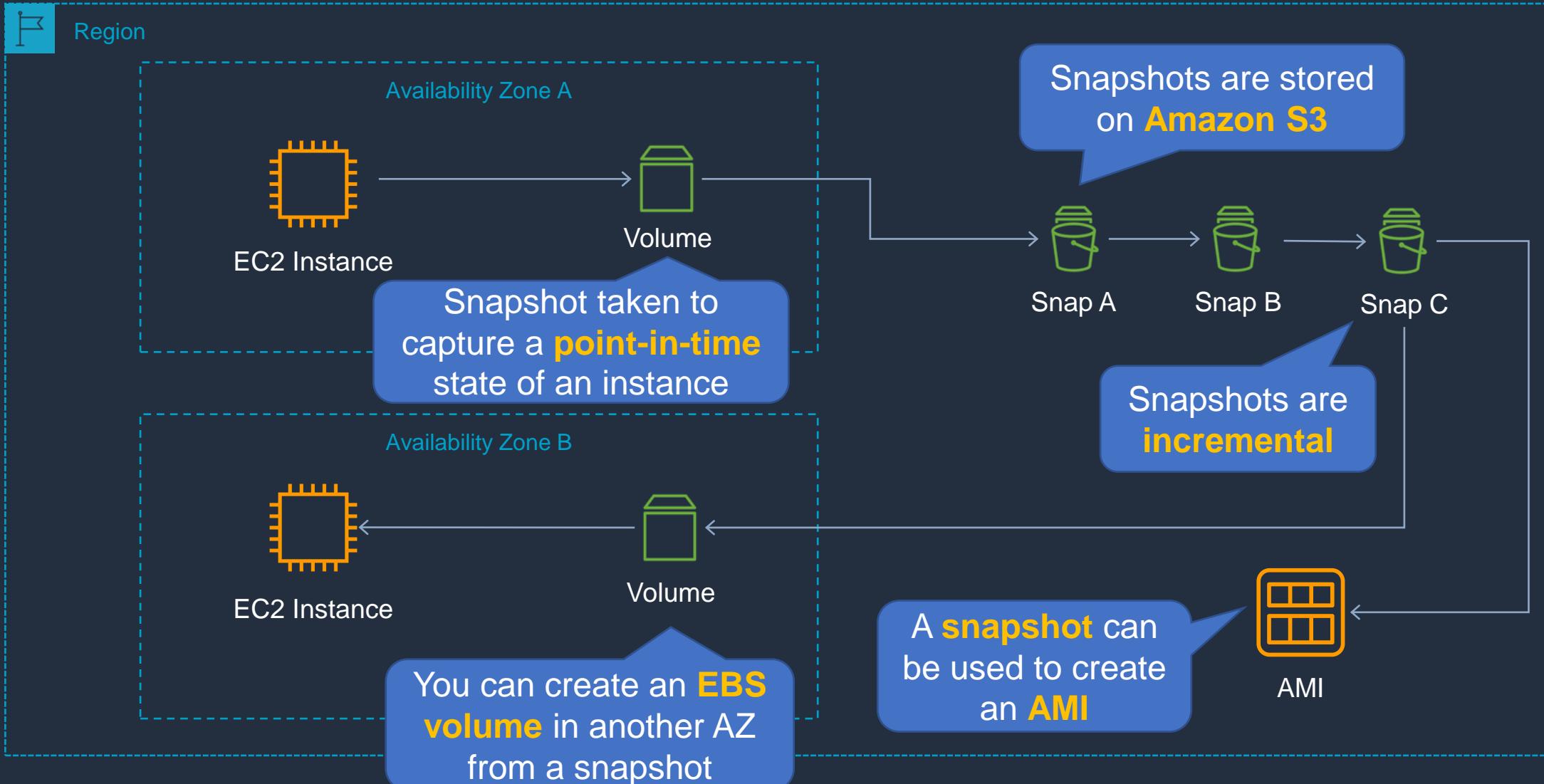


Amazon EBS Snapshots and DLM





Amazon EBS Snapshots





Amazon Data Lifecycle Manager (DLM)

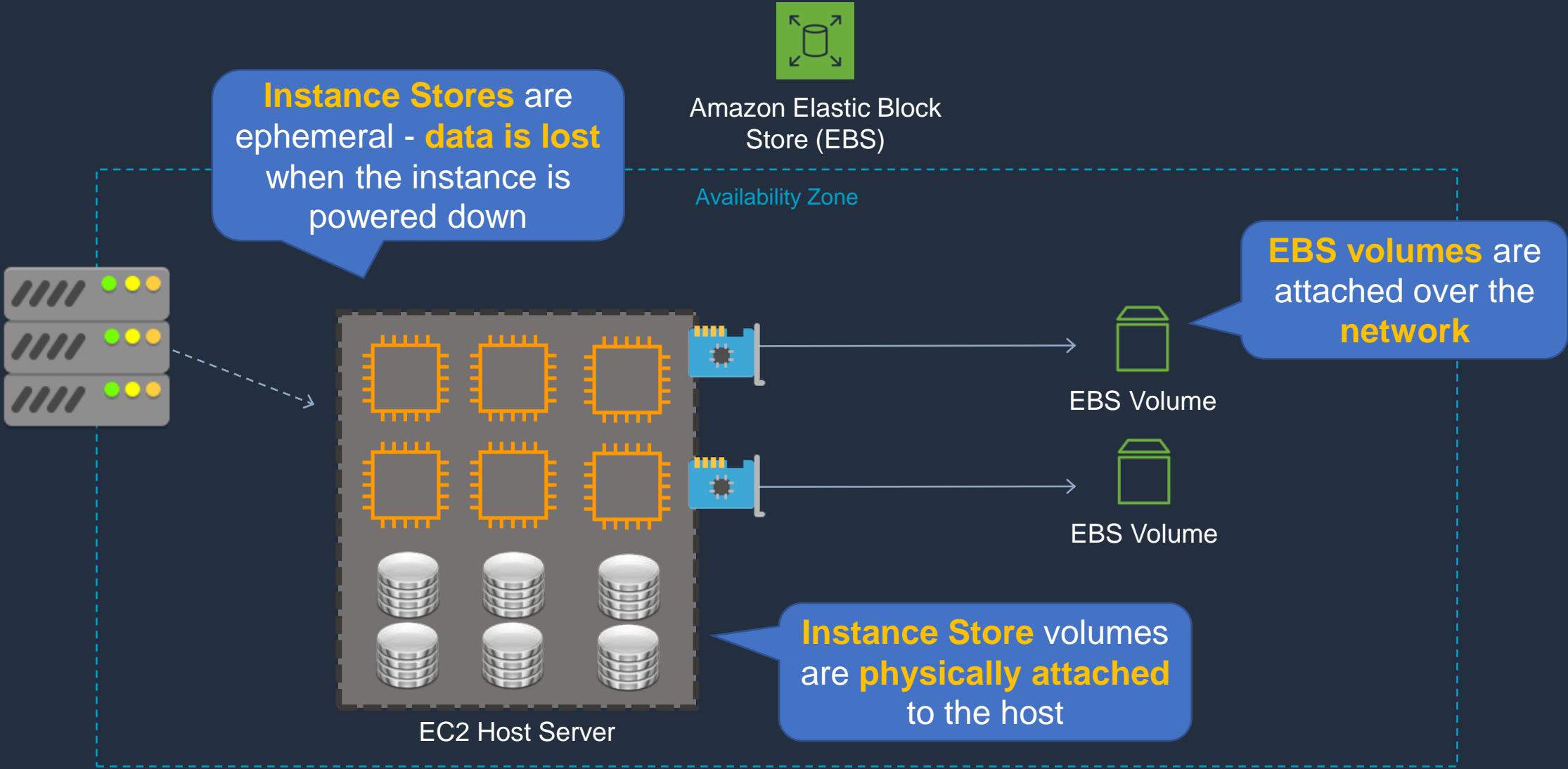
- DLM automates the creation, retention, and deletion of EBS snapshots and EBS-backed AMIs
- DLM helps with the following:
 - Protects valuable data by enforcing a regular backup schedule
 - Create standardized AMIs that can be refreshed at regular intervals
 - Retain backups as required by auditors or internal compliance
 - Reduce storage costs by deleting outdated backups
 - Create disaster recovery backup policies that back up data to isolated accounts

EC2 Instance Store Volumes





EBS vs instance store

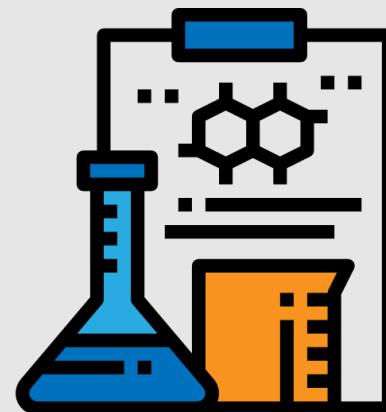




EBS vs instance store

- Instance store volumes are high performance local disks that are physically attached to the host computer on which an EC2 instance runs
- Instance stores are ephemeral which means the data is lost when powered off (non-persistent)
- Instance stores are ideal for temporary storage of information that changes frequently, such as buffers, caches, or scratch data
- Instance store volume root devices are created from AMI templates stored on S3
- Instance store volumes cannot be detached/reattached

Create and Attach an EBS Volume



EBS Snapshots and AMIs



Using RAID with EBS





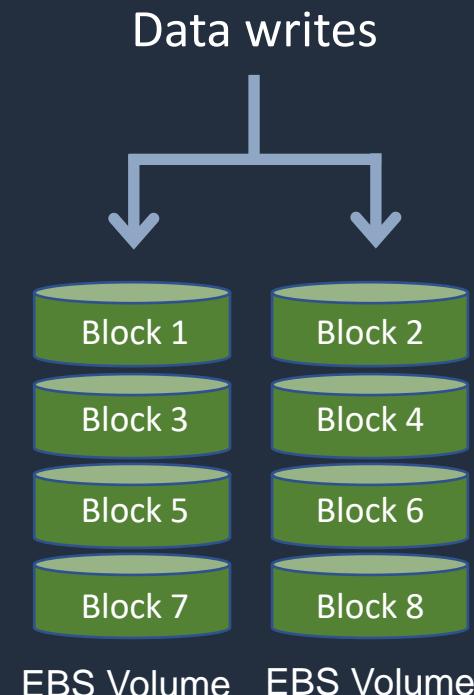
Using RAID with EBS

- RAID stands for Redundant Array of Independent disks
- Not provided by AWS, you must configure through your operating system
- RAID 0 and RAID 1 are potential options on EBS
- RAID 5 and RAID 6 are not recommended by AWS



Using RAID with EBS

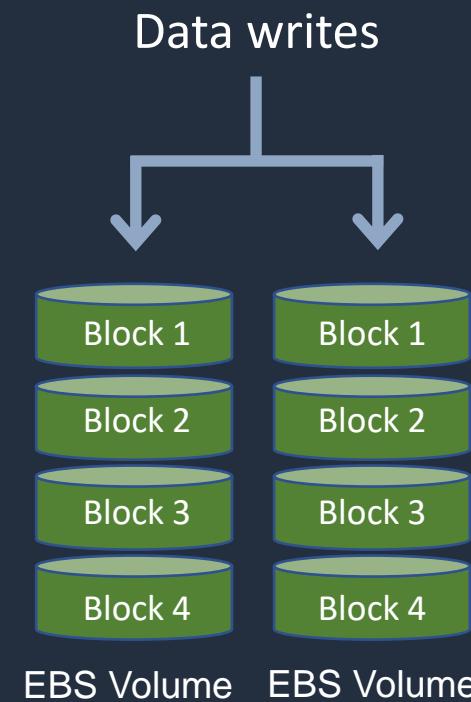
- RAID 0 is used for striping data across disks (performance):
 - Use 2 or more disks
 - If one disk fails, the entire RAID set fails





Using RAID with EBS

- RAID 1 is used for mirroring data across disks (redundancy / fault tolerance):
 - If one disk fails, the other disk is still working
 - Data gets sent to 2 EBS volumes at the same time



Amazon Elastic File System (EFS)

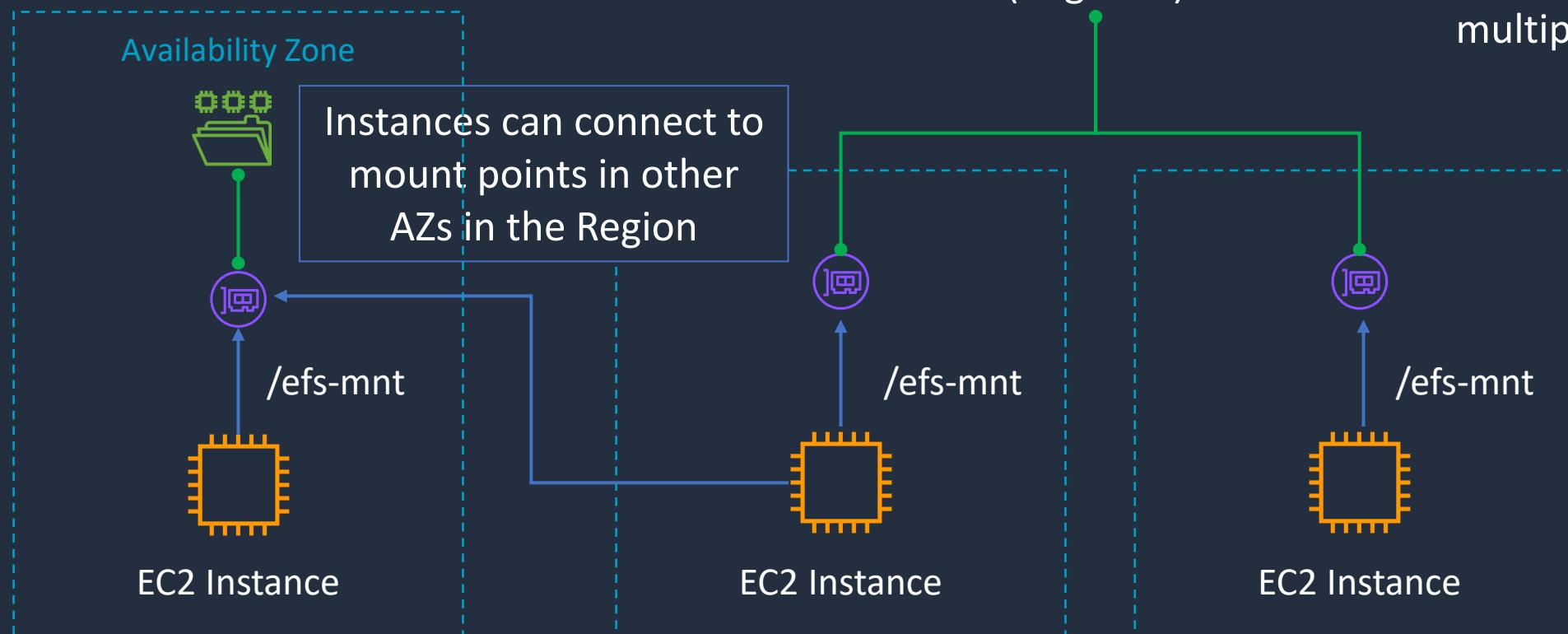




Amazon Elastic File System (EFS)

Note: EFS supports **Linux** only

One Zone file systems have mount targets in a single AZ



Regional file systems have mount targets in multiple AZs

The connection protocol is **NFS**

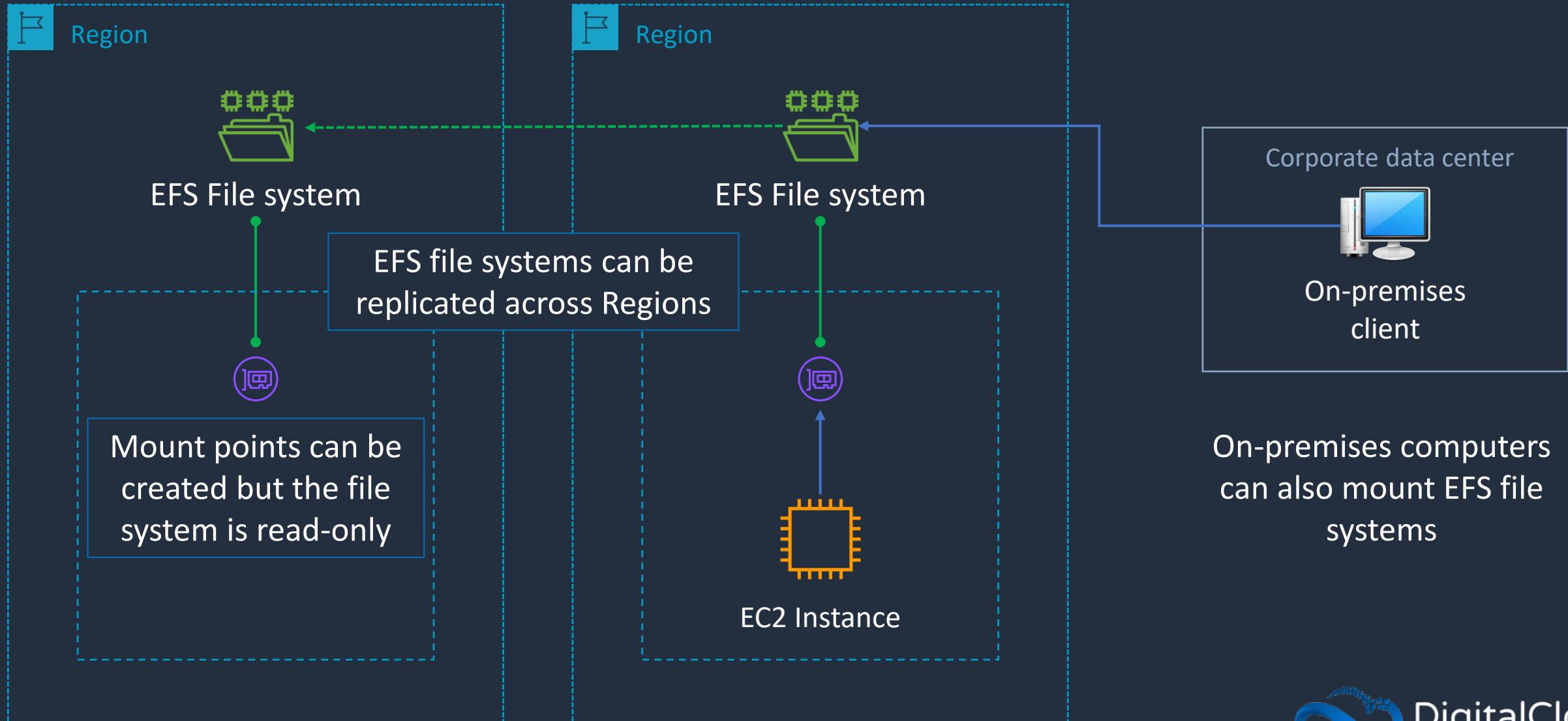


Amazon Elastic File System (EFS)

- **Data consistency** – write operations for Regional file systems are durably stored across Availability Zones
- **File locking** – NFS client applications can use NFS v4 file locking for read and write operations on EFS files
- **Storage classes** – there are three options:
 - **EFS Standard** – uses SSDs for low latency performance
 - **EFS Infrequent Access (IA)** – cost effective option
 - **EFS Archive** – even cheaper for less active data (archival)
- **Durability** – all storage classes offer 11 9s of durability



Amazon Elastic File System (EFS)





Amazon Elastic File System (EFS)

- **EFS Replication** – data is replicated across Regions for disaster recovery purposes with RPO/RTO in the minutes
- **Automatic Backup** – EFS integrates with AWS Backup for automatic file system backups
- **Performance options** – there are two options:
 - **Provisioned throughput** – Specify a level of throughput that the file system can drive independent of the file system's size
 - **Bursting throughput** – Throughput scales with the amount of storage and supports bursting to higher levels

Create an Amazon EFS File System



Amazon FSx



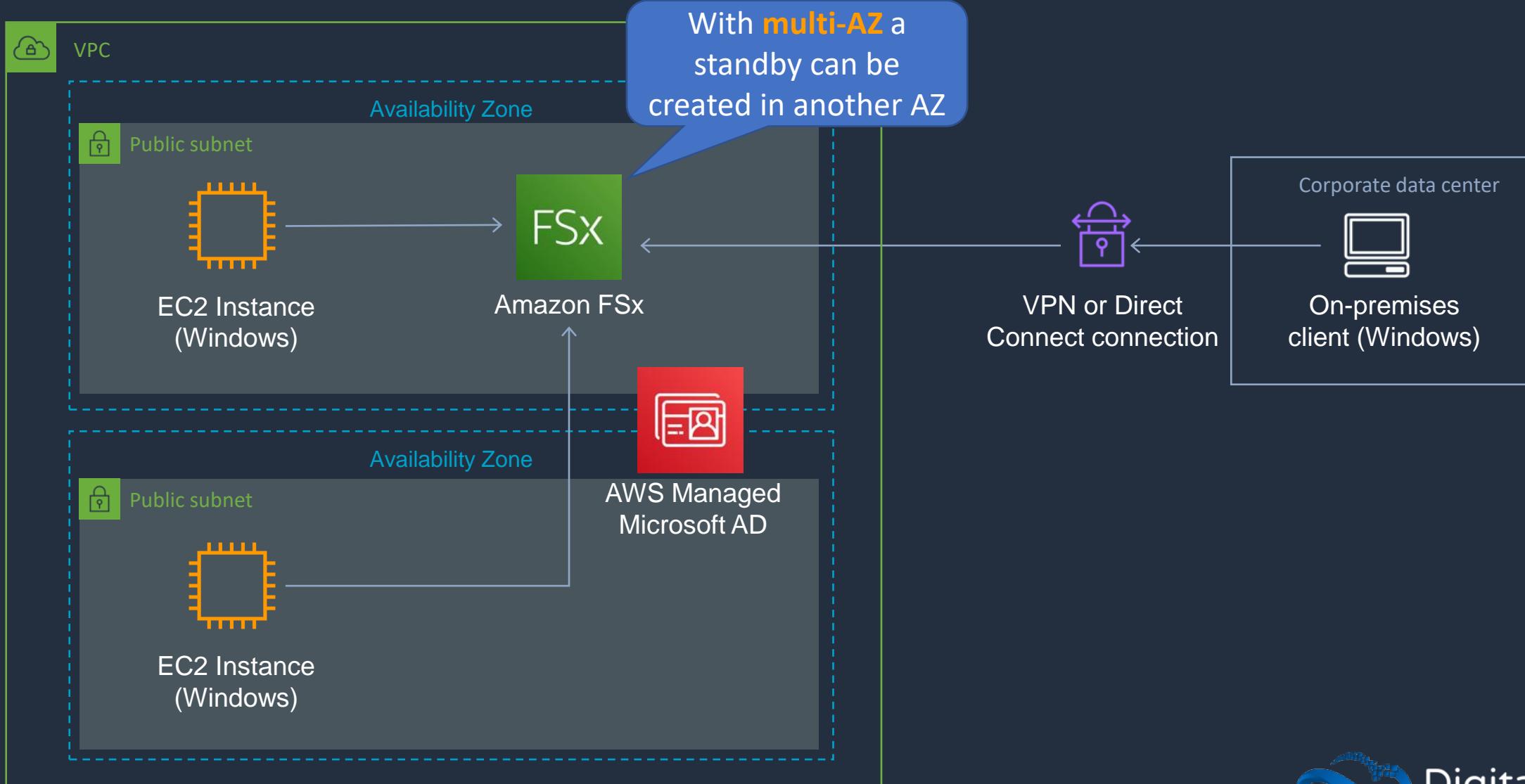
Amazon FSx

- Amazon FSx provides fully managed third-party file systems
- Amazon FSx provides you with two file systems to choose from:
 - **Amazon FSx for Windows File Server** for Windows-based applications
 - **Amazon FSx for Lustre** for compute-intensive workloads

Amazon FSx for Windows File Server

- Provides a fully managed native Microsoft Windows file system
- Full support for the SMB protocol, Windows NTFS, and Microsoft Active Directory (AD) integration
- Supports Windows-native file system features:
 - Access Control Lists (ACLs), shadow copies, and user quotas.
 - NTFS file systems that can be accessed from up to thousands of compute instances using the SMB protocol
- **High availability:** replicates data within an Availability Zone (AZ)
- **Multi-AZ:** file systems include an active and standby file server in separate AZs

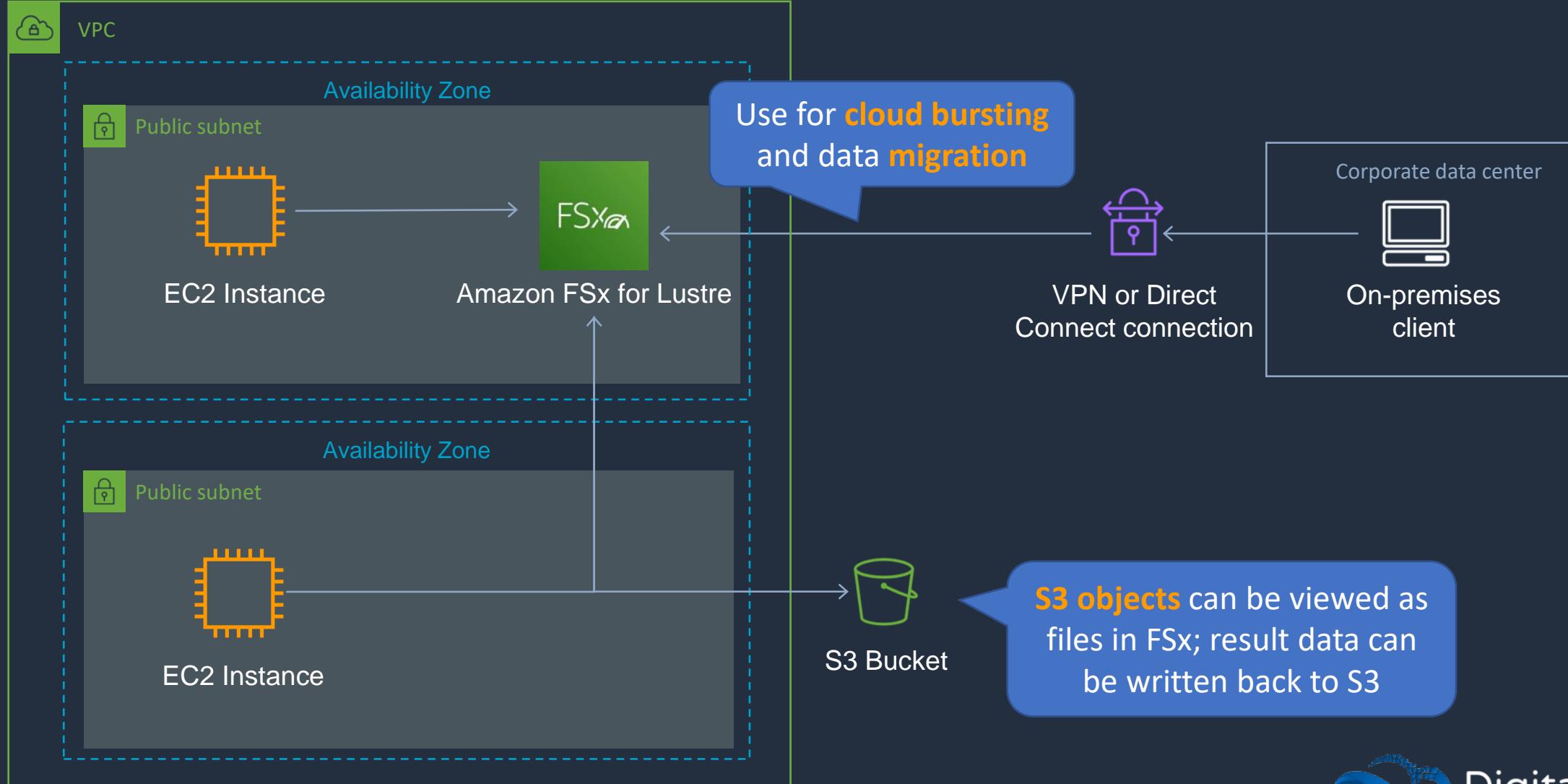
Amazon FSx for Windows File Server



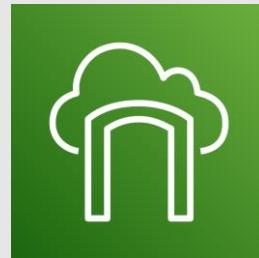
Amazon FSx for Lustre

- High-performance file system optimized for fast processing of workloads such as:
 - Machine learning
 - High performance computing (HPC)
 - Video processing
 - Financial modeling
 - Electronic design automation (EDA)
- Works natively with S3, letting you transparently access your S3 objects as files
- Your S3 objects are presented as files in your file system, and you can write your results back to S3
- Provides a POSIX-compliant file system interface

Amazon FSx for Lustre

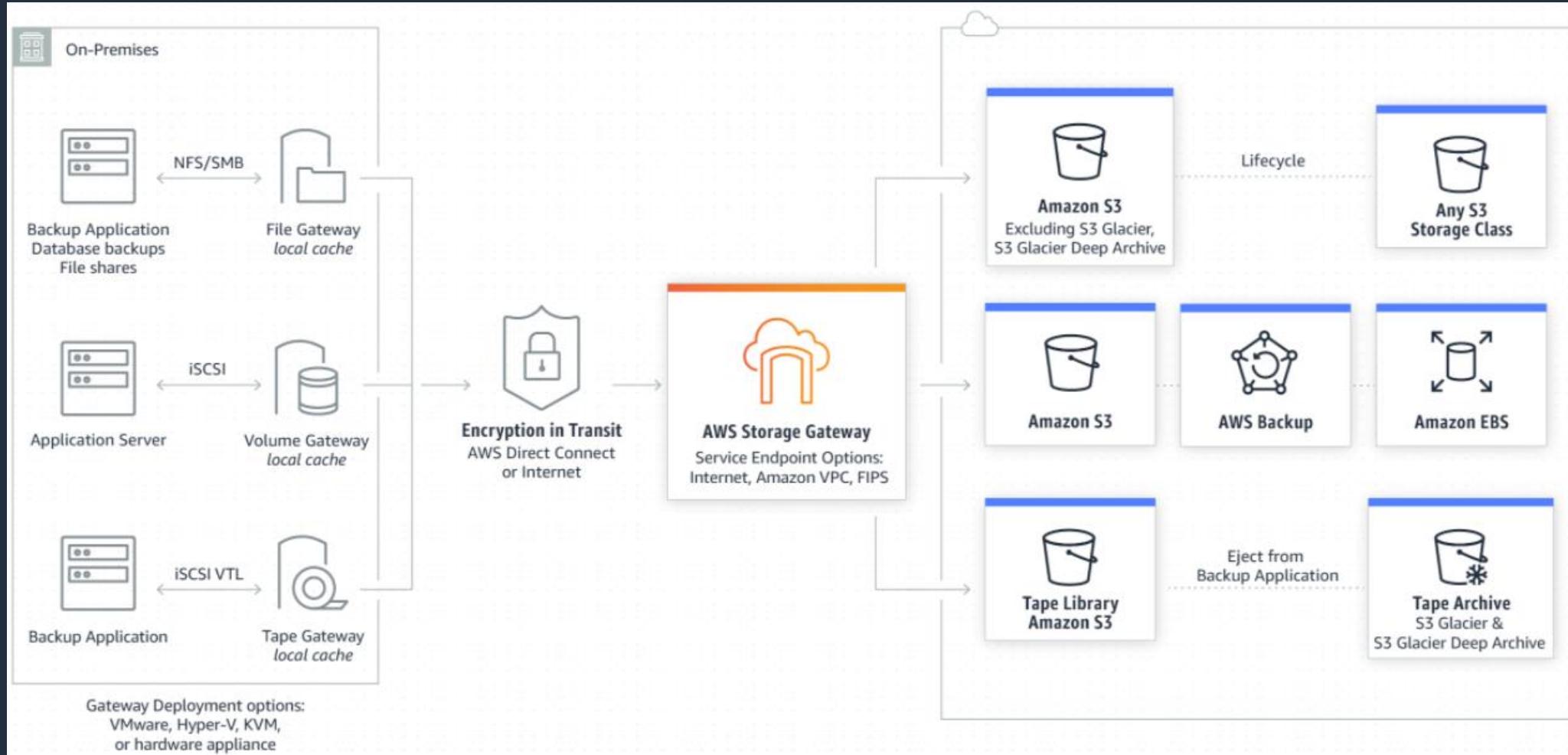


AWS Storage Gateway



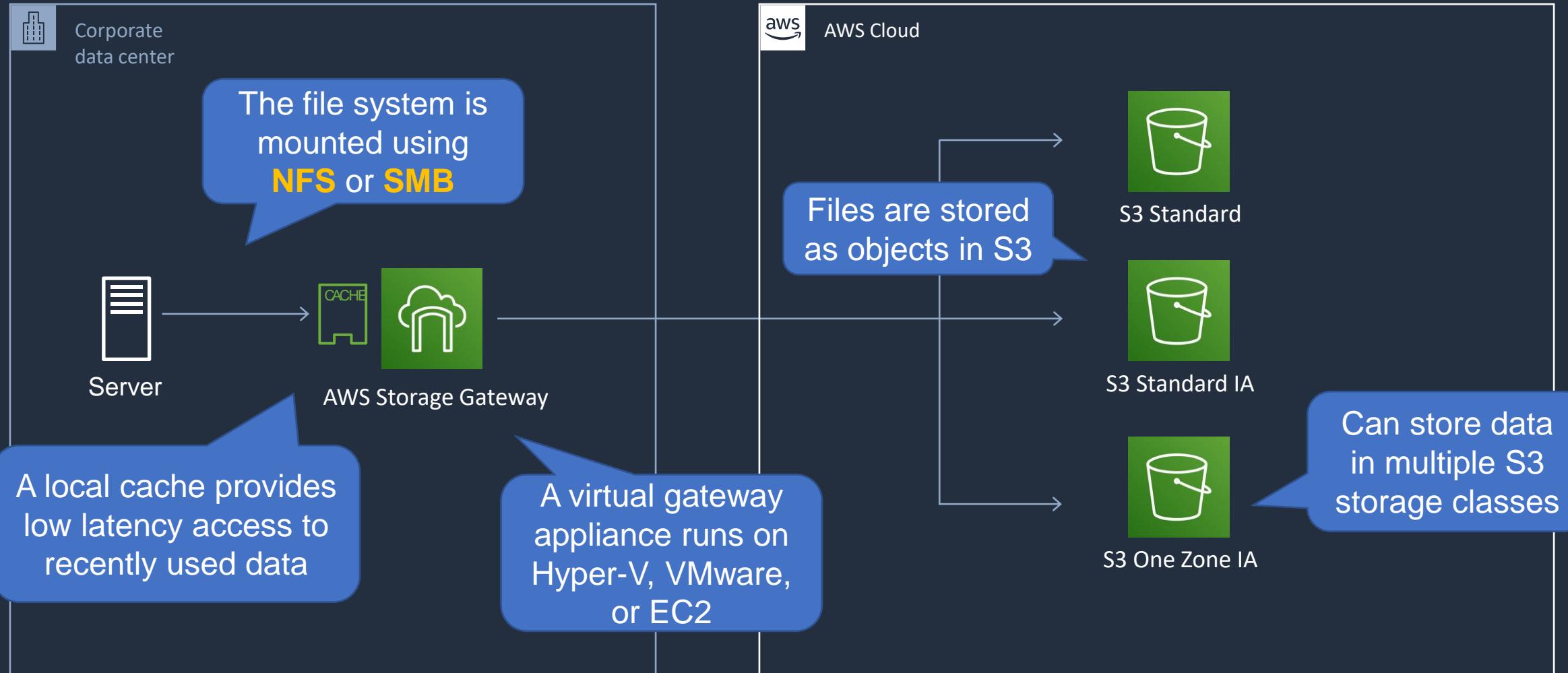


AWS Storage Gateway





AWS Storage Gateway – File Gateway



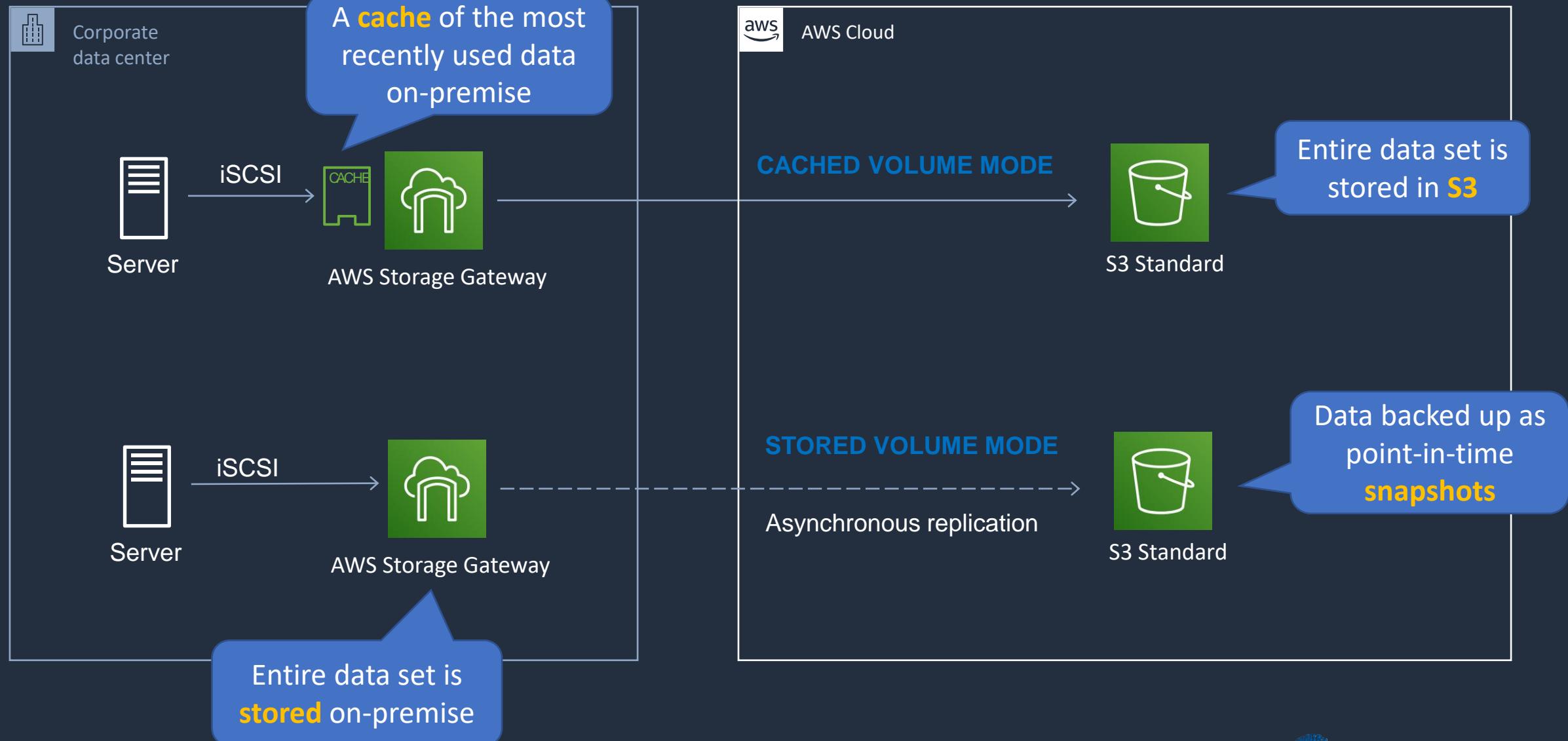


AWS Storage Gateway – File Gateway

- File gateway provides a virtual **on-premises file server**
- Store and retrieve files as objects in Amazon S3
- Use with on-premises applications, and EC2-based applications that need file storage in S3 for object-based workloads
- File gateway offers **SMB** or **NFS**-based access to data in Amazon S3 with local caching



AWS Storage Gateway - Volume Gateway



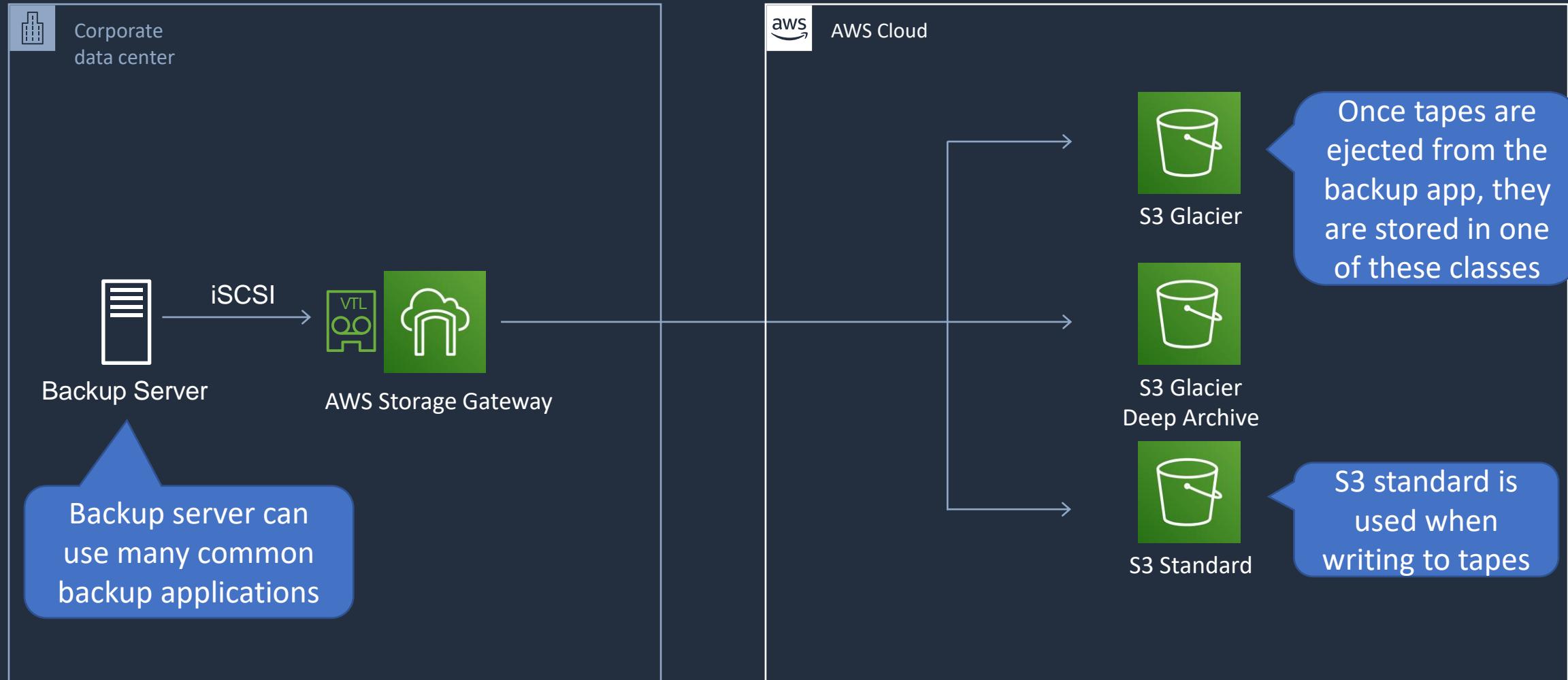


AWS Storage Gateway - Volume Gateway

- The volume gateway supports block-based volumes
- Block storage – iSCSI protocol
- **Cached Volume mode** – the entire dataset is stored on S3 and a cache of the most frequently accessed data is cached on-site
- **Stored Volume mode** – the entire dataset is stored on-site and is asynchronously backed up to S3 (EBS point-in-time snapshots).
Solutions are incremental and compressed



AWS Storage Gateway - Tape Gateway





AWS Storage Gateway - Tape Gateway

- Used for backup with popular backup software
- Each gateway is preconfigured with a media changer and tape drives. Supported by NetBackup, Backup Exec, Veeam etc.
- When creating virtual tapes, you select one of the following sizes: 100 GB, 200 GB, 400 GB, 800 GB, 1.5 TB, and 2.5 TB
- A tape gateway can have up to 1,500 virtual tapes with a maximum aggregate capacity of 1 PB
- All data transferred between the gateway and AWS storage is encrypted using SSL
- All data stored by tape gateway in S3 is encrypted server-side with Amazon S3-Managed Encryption Keys (SSE-S3)

Architecture Patterns – Block and File Storage





Architecture Patterns – Block and File Storage

Requirement

Simple method of backing up Amazon EBS volumes is required that is fully automated

Solution

Use Data Lifecycle Manager to create a backup schedule

A distributed application has many nodes that each hold a copy of data that is synchronized between them. Need the best performance

Use instance stores for storing the data with the best performance

Application must startup quickly when launched by ASG but requires app dependencies and code to be installed

Create an AMI that includes the application dependencies and code



Architecture Patterns – Block and File Storage

Requirement

Many Linux instances must be attached to a shared filesystem that scales elastically

Company requires a managed file system that uses the NTFS file system

On-premises servers must be able to attach a block storage system locally. Data should be backed up to S3 as snapshots

Solution

Create an Amazon EFS file system and mount from each instance

Use Amazon FSx for Windows File Server

Deploy an AWS Storage Gateway volume gateway in stored volume mode



Architecture Patterns – Block and File Storage

Requirement

An Amazon EBS volume must be moved between Regions

Root EBS volumes for a critical application must not be deleted on termination

On-premises servers use NFS to attach a file system. The file system should be replaced with an AWS service that uses Amazon S3 with a local cache

Solution

Take a snapshot and copy the snapshot between Regions

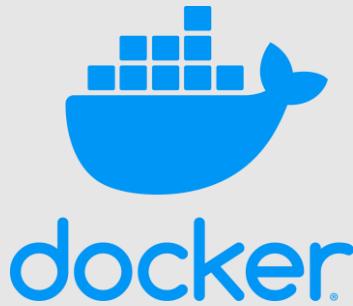
Modify the delete on termination attribute when launching the EC2 instances

Deploy an AWS Storage Gateway file gateway

SECTION 10

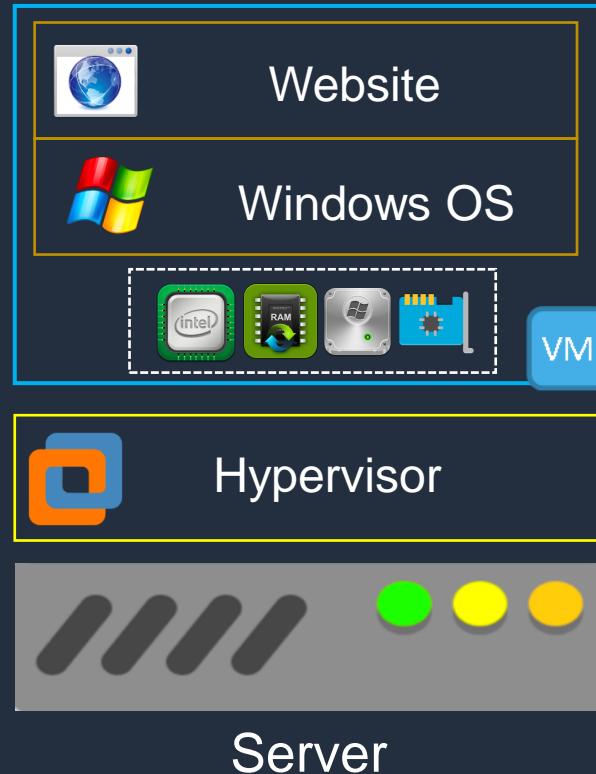
Docker Containers and ECS

Docker Containers and Microservices



Server Virtualization vs Containers

Every VM-instance needs an **operating system** which uses significant resources





Docker Containers

A **container** includes all the code, settings, and dependencies for running the application

Containers **start up** very **quickly**



Containers are very resource **efficient**

Each container is **isolated** from other containers



Docker Containers

- Docker utilizes containerization to package an application and its dependencies into a single **container image**
- Docker provides **Docker Hub**, a cloud-based registry service for sharing container images and automating workflows
- Containers are lightweight because they share the host system's kernel
- Docker is ideal for **microservices** architectures and building **cloud-native** applications



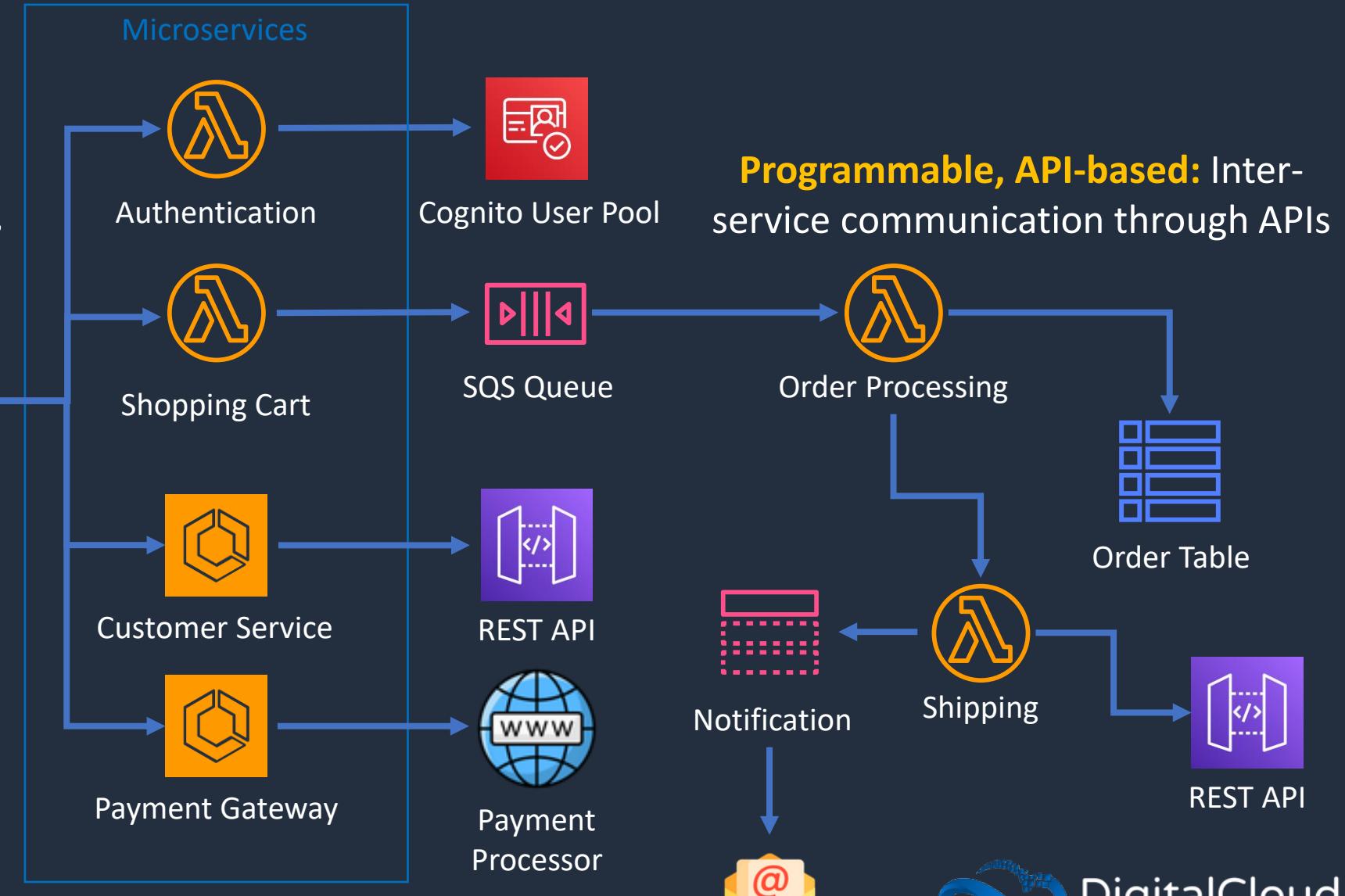
Cloud-Native Applications

Microservices architecture:

Applications are structured as a collection of **loosely coupled**, independently deployable services, each running its own process



Containers and Functions: Code runs in Docker containers and Lambda functions for isolation, elasticity, and cost-efficiency

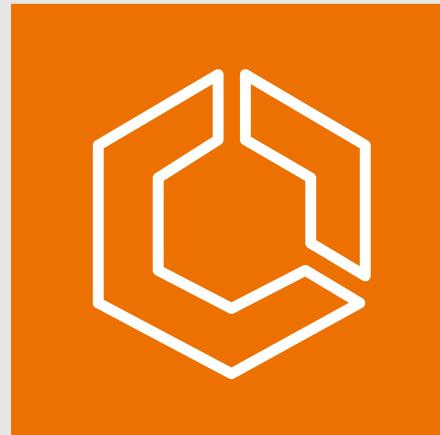




Microservices: Attributes and Benefits

Microservices Attribute	Microservices Benefit
Use of Application Programming Interfaces (APIs)	Easier integrations between application components; assists with loose coupling
Independently deployable blocks of code	Can be scaled and maintained independently
Business-oriented architecture	Development organized around business capabilities; teams may be cross-functional and services may be reused
Flexible use of technologies	Each microservice can be written using different technologies (e.g. programming languages)
Speed and agility	Fast to deploy and update. Easy to include high availability and fault tolerance for each microservice

Amazon Elastic Container Service (ECS)





Amazon ECS

ECS **Services** are used to maintain a **desired count** of tasks

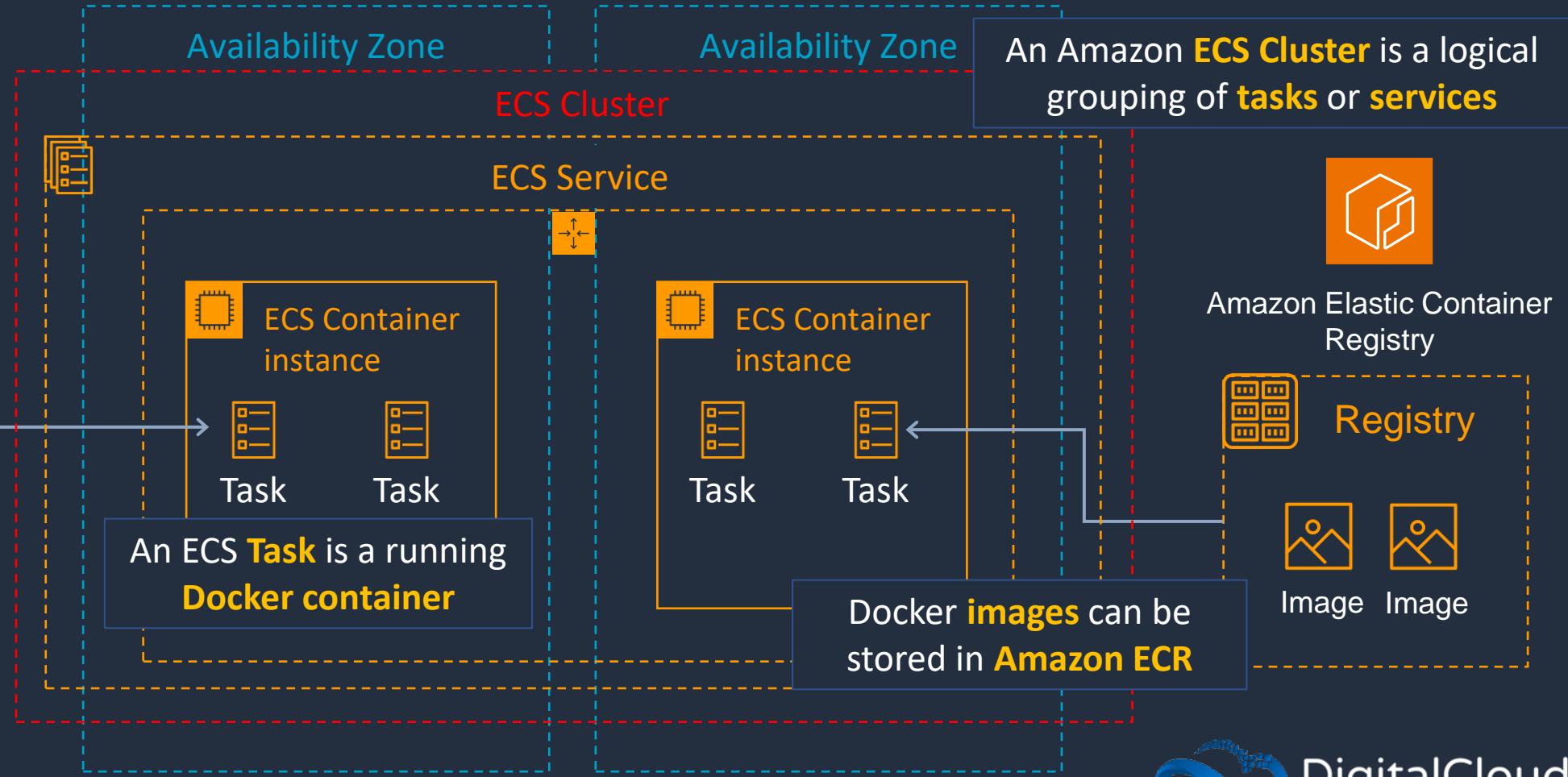
An ECS **Task** is created from a **Task Definition**

Task Definition

```
{
  "containerDefinitions": [
    {
      "name": "wordpress",
      "links": [
        "mysql"
      ],
      "image": "wordpress",
      "essential": true,
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80
        }
      ],
      "memory": 500,
      "cpu": 10
    }
  ]
}
```



Amazon Elastic Container Service





Amazon ECS Key Features

- **Serverless with AWS Fargate** – managed for you and fully scalable
- **Fully managed container orchestration** – control plane is managed for you
- **Docker support** – run and manage Docker containers with integration into the Docker Compose CLI
- **Windows container support** – ECS supports management of Windows containers
- **Elastic Load Balancing integration** – distribute traffic across containers using ALB or NLB
- **Amazon ECS Anywhere** – enables the use of Amazon ECS control plane to manage on-premises implementations



Amazon ECS Components

Elastic Container Service (ECS)	Description
Cluster	Logical grouping of tasks or services
Container instance	EC2 instance running the the ECS agent
Task Definition	Blueprint that describes how a docker container should launch
Task	A running container using settings in a task definition
Image	A Docker image referenced in the task definition
Service	Defines long running tasks – can control task count with Auto Scaling and attach an ELB



Amazon ECS Images

- Containers are created from a read-only template called an **image** which has the instructions for creating a Docker container
- Images are built from a **Dockerfile**
- Only Docker containers are supported on ECS
- Images are stored in a registry such as DockerHub or Amazon Elastic Container Registry (ECR)
- ECR is a managed AWS Docker registry service that is secure, scalable and reliable
- ECR supports private Docker repositories with resource-based permissions using AWS IAM in order to access repositories and images
- You can use the Docker CLI to push, pull and manage images



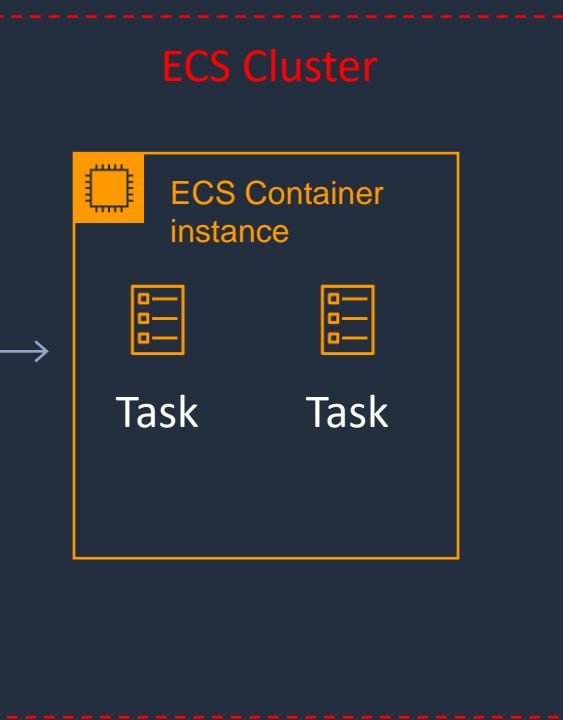


Amazon ECS Tasks and Task Definitions

- A task definition is required to run Docker containers in Amazon ECS
- A task definition is a text file in JSON format that describes one or more containers, up to a maximum of 10
- Task definitions use Docker images to launch containers

Task Definition

```
{  
  "containerDefinitions": [  
    {  
      "name": "wordpress",  
      "links": [  
        "mysql"  
      ],  
      "image": "wordpress",  
      "essential": true,  
      "portMappings": [  
        {  
          "containerPort": 80,  
          "hostPort": 80  
        }  
      ],  
      "memory": 500,  
      "cpu": 10  
    }  
  ]  
}
```

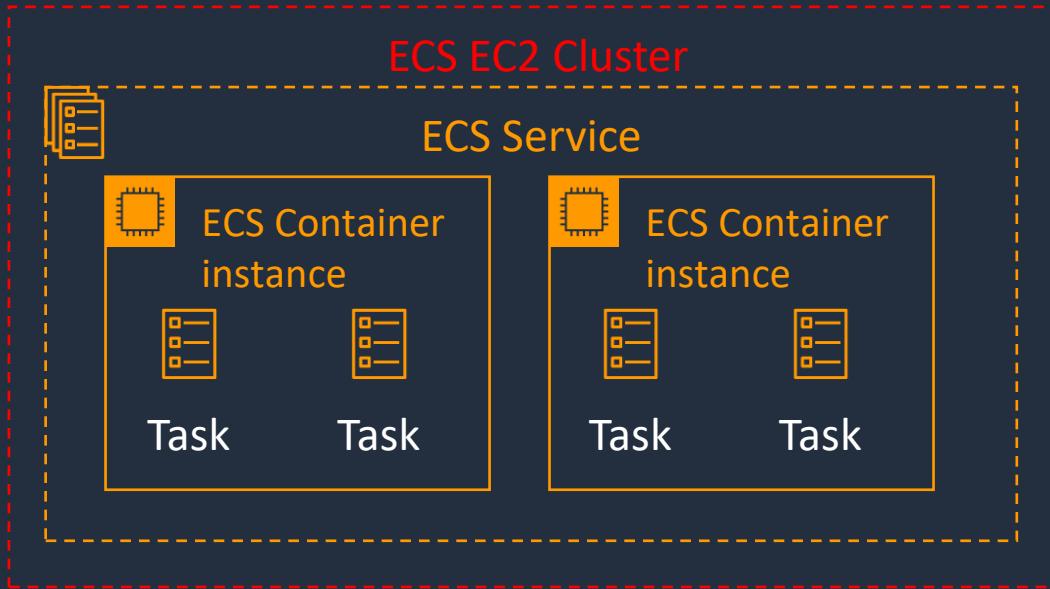




Launch Types – EC2 and Fargate



Registry:
ECR, Docker Hub, Self-hosted

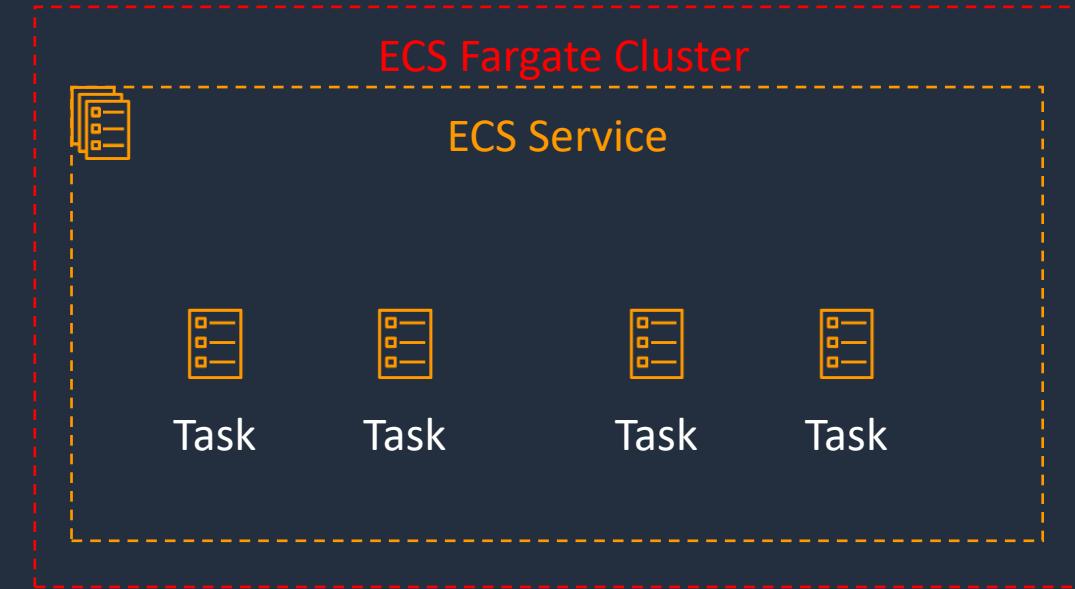


EC2 Launch Type:

- You explicitly provision EC2 instances
- You're responsible for managing EC2 instances
- Charged per running EC2 instance
- EFS, FSx, and EBS integration
- You handle cluster optimization
- More granular control over infrastructure



Registry:
ECR, Docker Hub

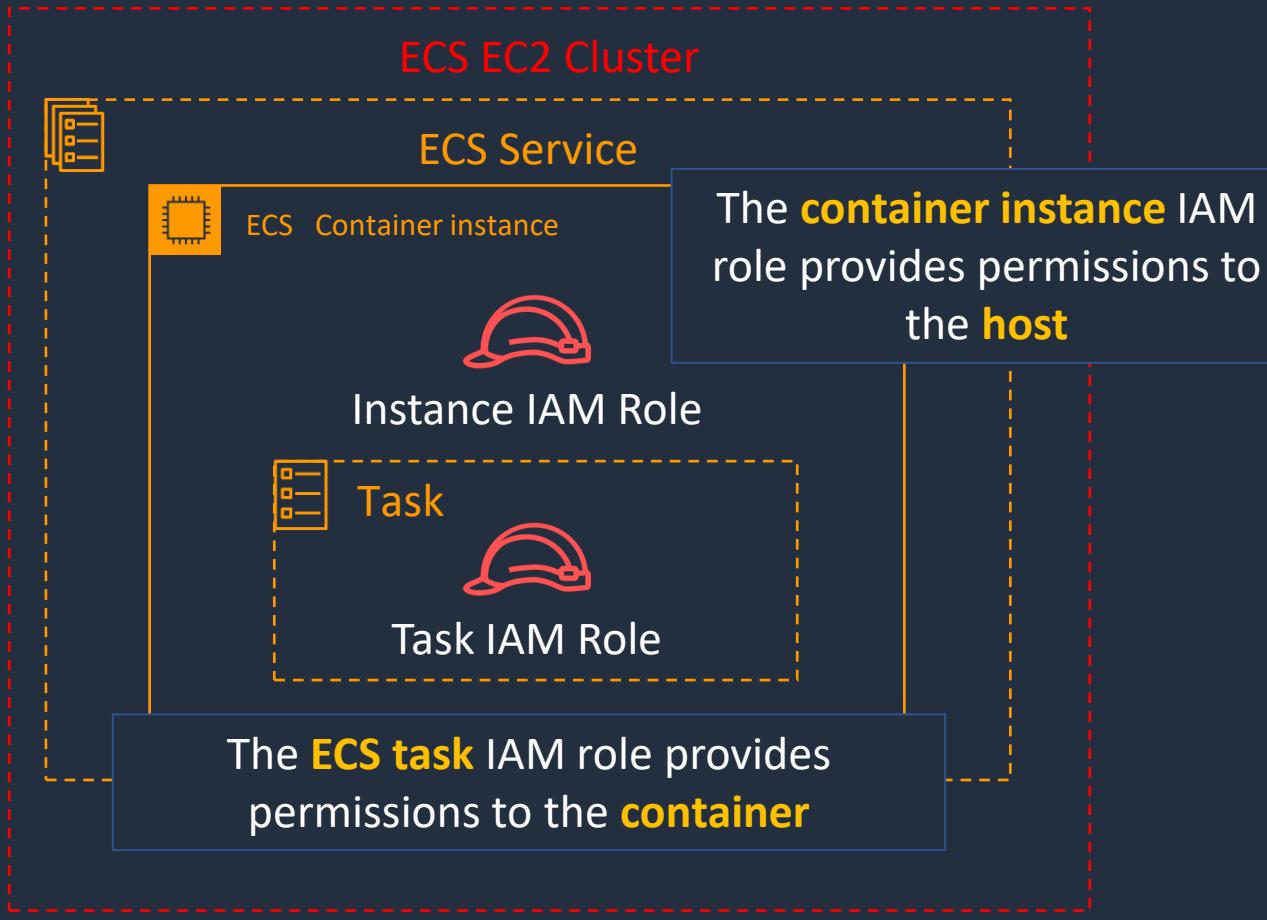


Fargate Launch Type:

- Fargate automatically provisions resources
- Fargate provisions and manages compute
- Charged for running tasks
- EFS integration only
- Fargate handles cluster optimization
- Limited control



ECS and IAM Roles



NOTE: With the Fargate launch type the container instance role is replaced with the **Task Execution Role**

AmazonEC2ContainerServiceforEC2Role

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "ec2:DescribeTags",  
        "ecs>CreateCluster",  
        "ecs>DeregisterContainerInstance",  
        "ecs>DiscoverPollEndpoint",  
        "ecs>Poll",  
        "ecs>RegisterContainerInstance",  
        "ecs>StartTelemetrySession",  
        "ecs>UpdateContainerInstancesState",  
        "ecs>Submit*",  
        "ecr>GetAuthorizationToken",  
        "ecr>BatchCheckLayerAvailability",  
        "ecr>GetDownloadUrlForLayer",  
        "ecr>BatchGetImage",  
        "logs>CreateLogStream",  
        "logs>PutLogEvents"  
      ],  
      "Resource": "*"  
    }  
  ]  
}
```

Amazon ECS and IAM Roles





ECS and IAM Roles

There are several roles used with ECS:

- **Amazon ECS container instance IAM role** – used by EC2 and external instances to provide permissions to the container agent to call AWS APIs
- **Task IAM role** – the permissions granted in the IAM role are assumed by the containers running in the task
- **Amazon ECS task execution IAM role** – the task execution role grants the Amazon ECS container and Fargate agents permissions to make AWS API calls on your behalf
- **Amazon ECS infrastructure IAM role** – allows Amazon ECS to manage infrastructure resources in your clusters on your behalf (e.g. EBS volumes)



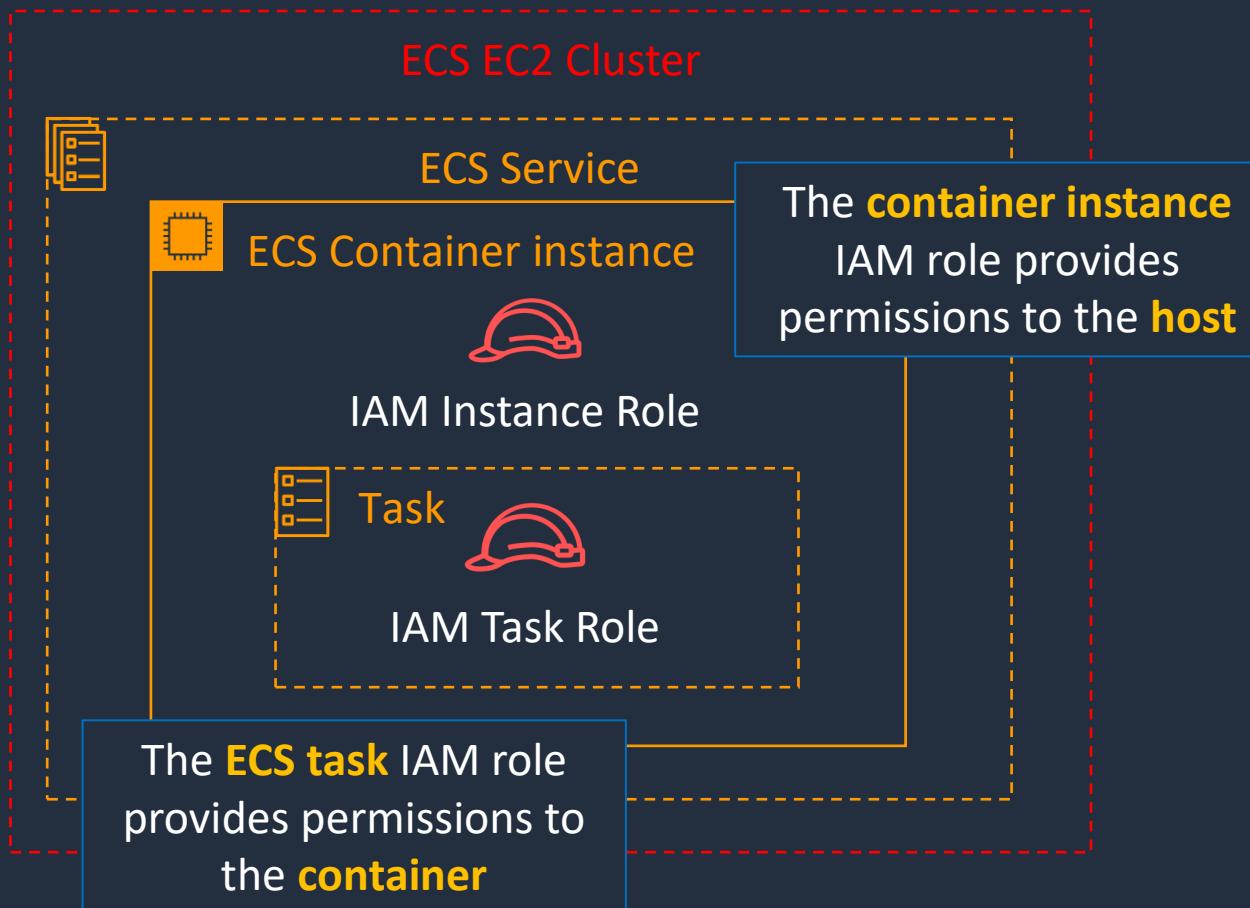
ECS and IAM Roles

There are several roles used with ECS:

- **ECS Anywhere IAM role** – On-premises servers or virtual machines (VM) require an IAM role to communicate with AWS APIs
- **Amazon ECS CodeDeploy IAM Role** – the CodeDeploy service needs permissions to update ECS when performing blue/green deployments
- **Amazon ECS EventBridge IAM Role** – to use Amazon ECS scheduled tasks with EventBridge rules and targets, the EventBridge service needs permissions



ECS and IAM Roles

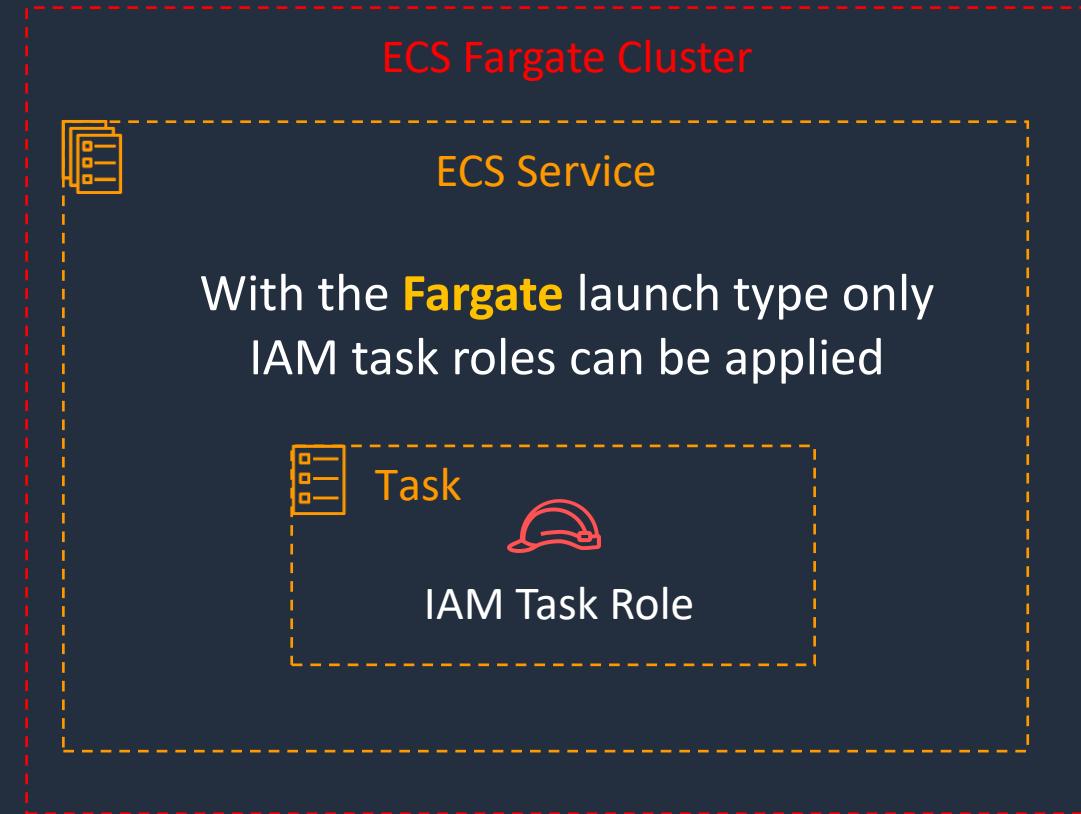


AmazonEC2ContainerServiceforEC2Role

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "ec2:DescribeTags",  
        "ecs>CreateCluster",  
        "ecs>DeregisterContainerInstance",  
        "ecs>DiscoverPollEndpoint",  
        "ecs>Poll",  
        "ecs>RegisterContainerInstance",  
        "ecs>StartTelemetrySession",  
        "ecs>UpdateContainerInstancesState",  
        "ecs>Submit*",  
        "ecr>GetAuthorizationToken",  
        "ecr>BatchCheckLayerAvailability",  
        "ecr>GetDownloadUrlForLayer",  
        "ecr>BatchGetImage",  
        "logs>CreateLogStream",  
        "logs>PutLogEvents"  
      ],  
      "Resource": "*"  
    }  
  ]  
}
```



ECS and IAM Roles





ECS and IAM Roles

- A container can only retrieve credentials for the IAM role that is defined in the task definition to which it belongs
- A container never has access to credentials that are intended for another container that belongs to another task
- When using EC2 instances, tasks are not prevented from accessing credentials supplied to the IAM instance role

Scaling Amazon ECS





Auto Scaling for ECS

Two types of scaling:

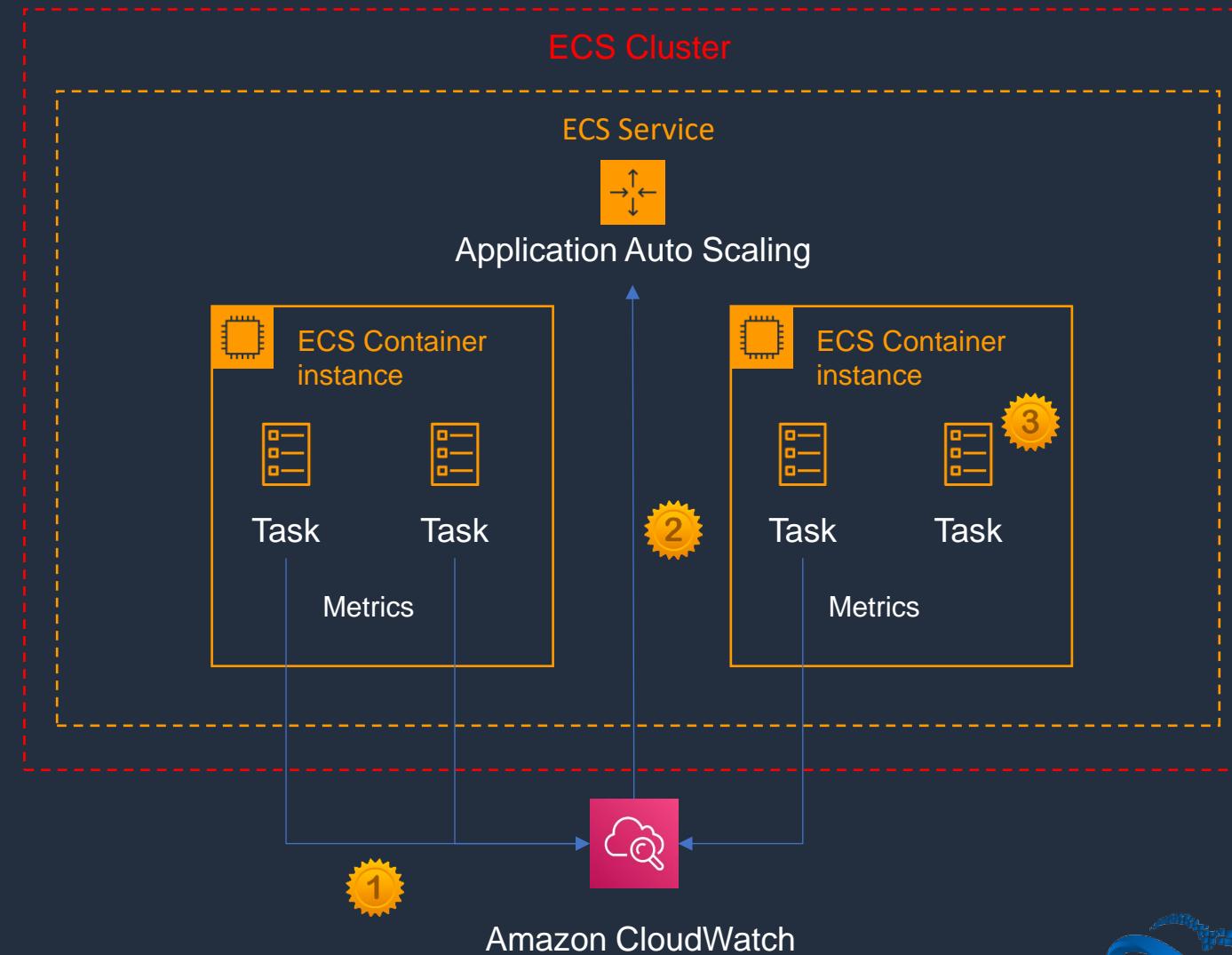
1. Service auto scaling
2. Cluster auto scaling

- **Service auto scaling** automatically adjusts the desired task count up or down using the Application Auto Scaling service
- **Service auto scaling** supports target tracking, step, and scheduled scaling policies
- **Cluster auto scaling** uses a Capacity Provider to scale the number of EC2 cluster instances using EC2 Auto Scaling



Service Auto Scaling

- 1. Metric reports CPU > 80%
- 2. CloudWatch notifies Application Auto Scaling
- 3. ECS launches additional task





Service Auto Scaling

- Amazon ECS Service Auto Scaling supports the following types of scaling policies:
 - **Target Tracking Scaling Policies**—Increase or decrease the number of tasks that your service runs based on a target value for a specific CloudWatch metric
 - **Step Scaling Policies**—Increase or decrease the number of tasks that your service runs in response to CloudWatch alarms. Step scaling is based on a set of scaling adjustments, known as step adjustments, which vary based on the size of the alarm breach
 - **Scheduled Scaling**—Increase or decrease the number of tasks that your service runs based on the date and time



Cluster Auto Scaling

- 1. Metric reports target capacity > 80%
- 2. CloudWatch notifies ASG
- 3. AWS launches additional container instance

ASG is linked to ECS using a **Capacity Provider**

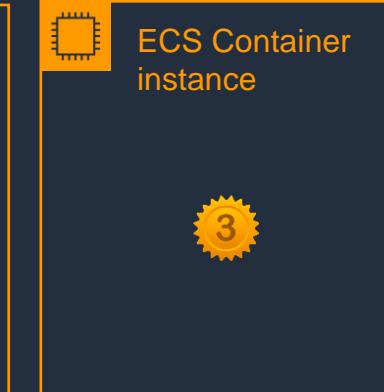
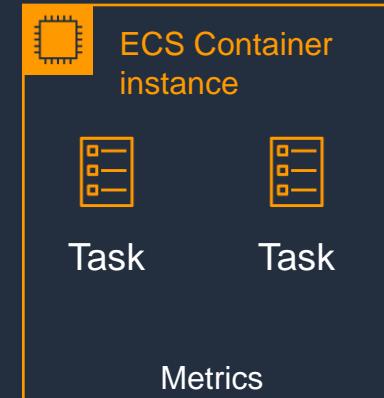


Amazon Elastic Container Service

ECS Cluster

ECS Service

Auto Scaling group



A **Capacity provider reservation**

metric measures the total percentage of cluster resources needed by all ECS workloads in the cluster

1



Amazon CloudWatch



Cluster Auto Scaling

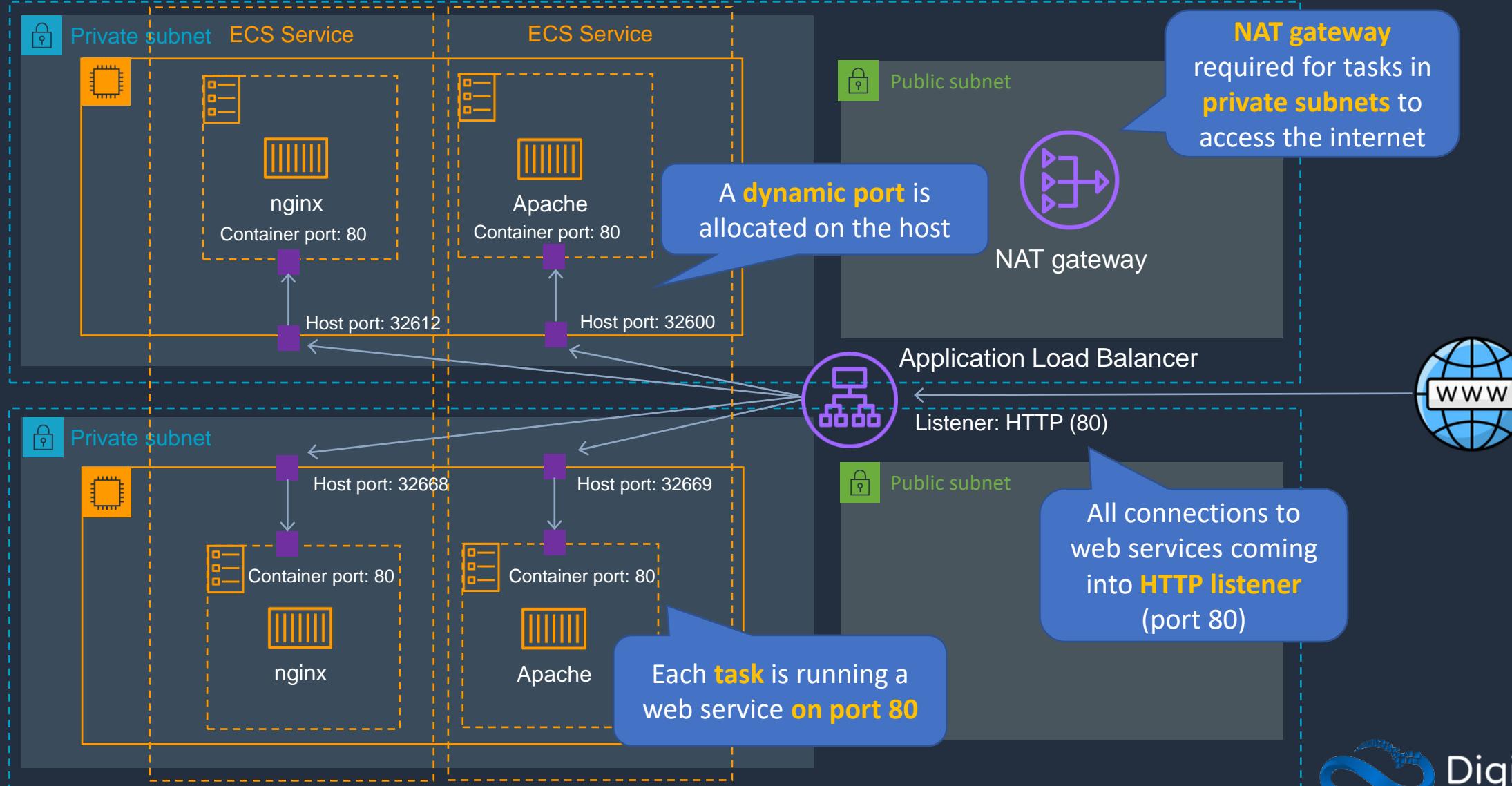
- Uses an ECS resource type called a **Capacity Provider**
- A Capacity Provider can be associated with an EC2 **Auto Scaling Group** (ASG)
- ASG can automatically scale using:
 - **Managed scaling** - with an automatically-created scaling policy on your ASG
 - **Managed instance termination protection** - which enables container-aware termination of instances in the ASG when scale-in happens

Amazon ECS with ALB





Amazon ECS with ALB



Launch Docker Containers on AWS Fargate



Amazon Elastic Kubernetes Service (EKS)



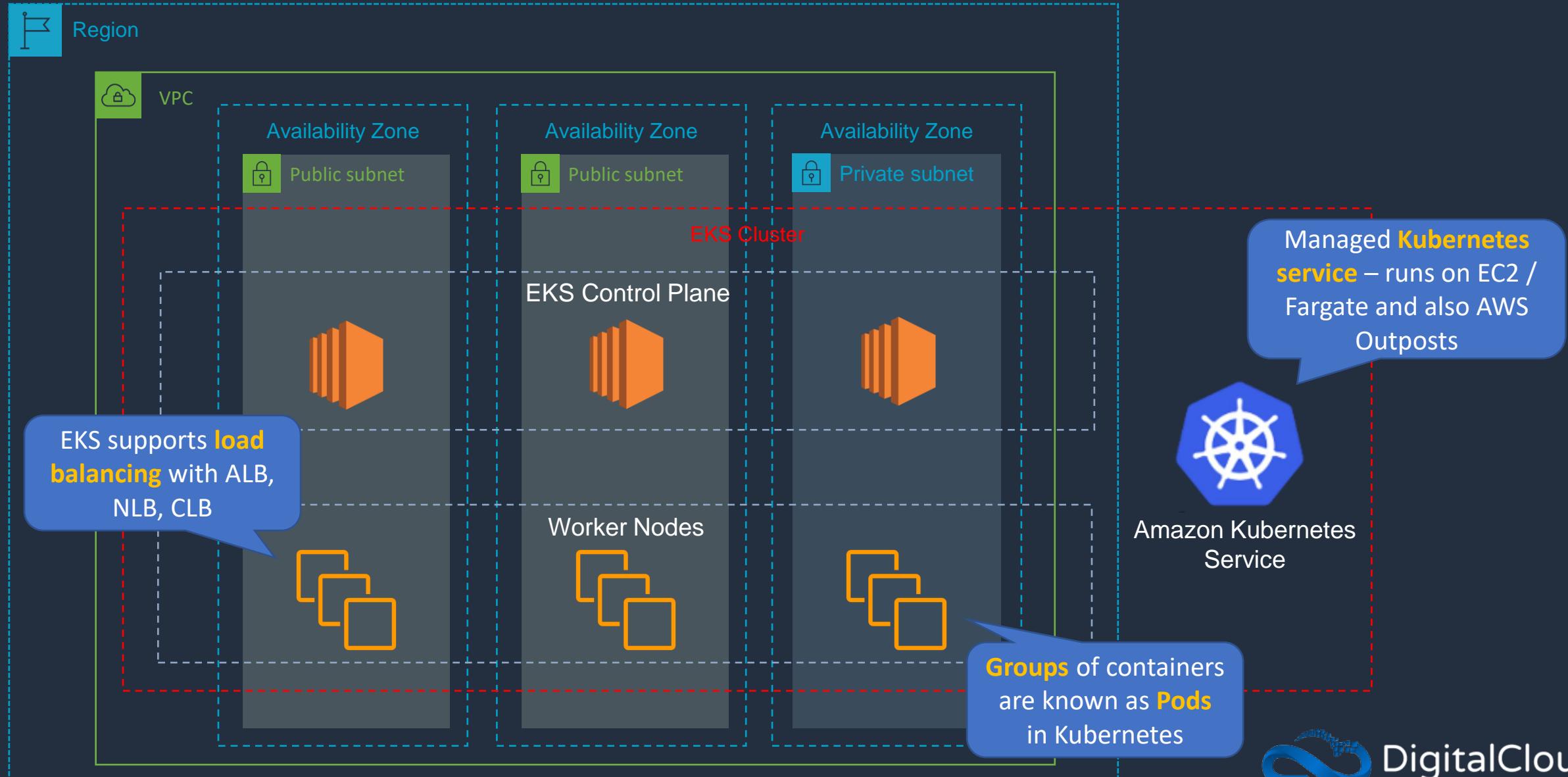


Amazon EKS

- Amazon Elastic Kubernetes Service (Amazon EKS) is a managed service for running Kubernetes applications in the cloud or on-premises
- Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications
- Use when you need to **standardize** container orchestration across multiple environments using a **managed Kubernetes** implementation
- Features:
 - **Hybrid Deployment** - manage Kubernetes clusters and applications across hybrid environments (AWS + On-premises)
 - **Batch Processing** - run sequential or parallel batch workloads on your EKS cluster using the Kubernetes Jobs API. Plan, schedule and execute batch workloads
 - **Machine Learning** - use Kubeflow with EKS to model your machine learning workflows and efficiently run distributed training jobs using the latest EC2 GPU-powered instances, including Inferentia
 - **Web Applications** - build web applications that automatically scale up and down and run in a highly available configuration across multiple Availability Zones



Amazon EKS





Amazon EKS Auto Scaling

Cluster Auto Scaling:

- **Vertical Pod Autoscaler** - automatically adjusts the CPU and memory reservations for your pods to help "right size" your applications
- **Horizontal Pod Autoscaler** - automatically scales the number of pods in a deployment, replication controller, or replica set based on that resource's CPU utilization

Workload Auto Scaling:

- Amazon EKS supports two **autoscaling** products:
 - Kubernetes Cluster Autoscaler
 - Karpenter open source autoscaling project
- The cluster autoscaler uses AWS scaling groups, while Karpenter works directly with the Amazon EC2 fleet



Amazon EKS and Elastic Load Balancing

- Amazon EKS supports Network Load Balancers and Application Load Balancers
- The AWS Load Balancer Controller manages AWS Elastic Load Balancers for a Kubernetes cluster
- Install the AWS Load Balancer Controller using Helm V3 or later or by applying a Kubernetes manifest
- The controller provisions the following resources:
 - An AWS Application Load Balancer (ALB) when you create a Kubernetes Ingress
 - An AWS Network Load Balancer (NLB) when you create a Kubernetes service of type **LoadBalancer**
- In the past, the Kubernetes network load balancer was used for instance targets, but the AWS Load balancer Controller was used for IP targets
- With the AWS Load Balancer Controller version 2.3.0 or later, you can create NLBs using either target type



Amazon EKS Distro

- Amazon EKS Distro is a distribution of Kubernetes with the same dependencies as Amazon EKS
- Allows you to manually run Kubernetes clusters anywhere
- EKS Distro includes binaries and containers of open-source Kubernetes, etcd, networking, and storage plugins, tested for compatibility
- You can securely access EKS Distro releases as open source on GitHub or within AWS via Amazon S3 and Amazon ECR
- Amazon EKS Distro alleviates the need to track updates, determine compatibility, and standardize on a common Kubernetes version across distributed teams
- You can create Amazon EKS Distro clusters in AWS on Amazon EC2 and on your own on-premises hardware using the tooling of your choice



Amazon ECS and EKS Anywhere

- Run ECS or EKS on customer-managed infrastructure, supported by AWS
- Customers can run Amazon ECS/EKS Anywhere on their own on-premises infrastructure on bare metal servers
- You can also deploy ECS/EKS Anywhere using VMware vSphere

Amazon Elastic Container Registry (ECR)





Amazon Elastic Container Registry (ECR)

- Amazon ECR is a fully-managed container registry
- Integrated with Amazon ECS and Amazon EKS
- Supports Open Container Initiative (OCI) and Docker Registry HTTP API V2 standards
- You can use Docker tools and Docker CLI commands such as `push`, `pull`, `list`, and `tag`
- Can be accessed from any Docker environment – in the cloud, on-premises, or on your machine



Amazon Elastic Container Registry (ECR)

- Container images and artifacts are stored in S3
- You can use namespaces to organize repositories
- Public repositories allow everyone to access container images
- Access control applies to private repositories:
 - **IAM access control** - Set policies to define access to container images in private repositories
 - **Resource-based policies** - Access control down to the individual API action such as `create`, `list`, `describe`, `delete`, and `get`



Amazon ECR Components

ECR Component	Description
Registry	An Amazon ECR private registry is provided to each AWS account; you can create one or more repositories in your registry and store images in them
Authorization token	Your client must authenticate to Amazon ECR registries as an AWS user before it can push and pull images
Repository	An Amazon ECR repository contains your Docker images, OCI images, and OCI compatible artifacts
Repository policy	You can control access to your repositories and the images within them with repository policies
Image	You can push and pull container images to your repositories

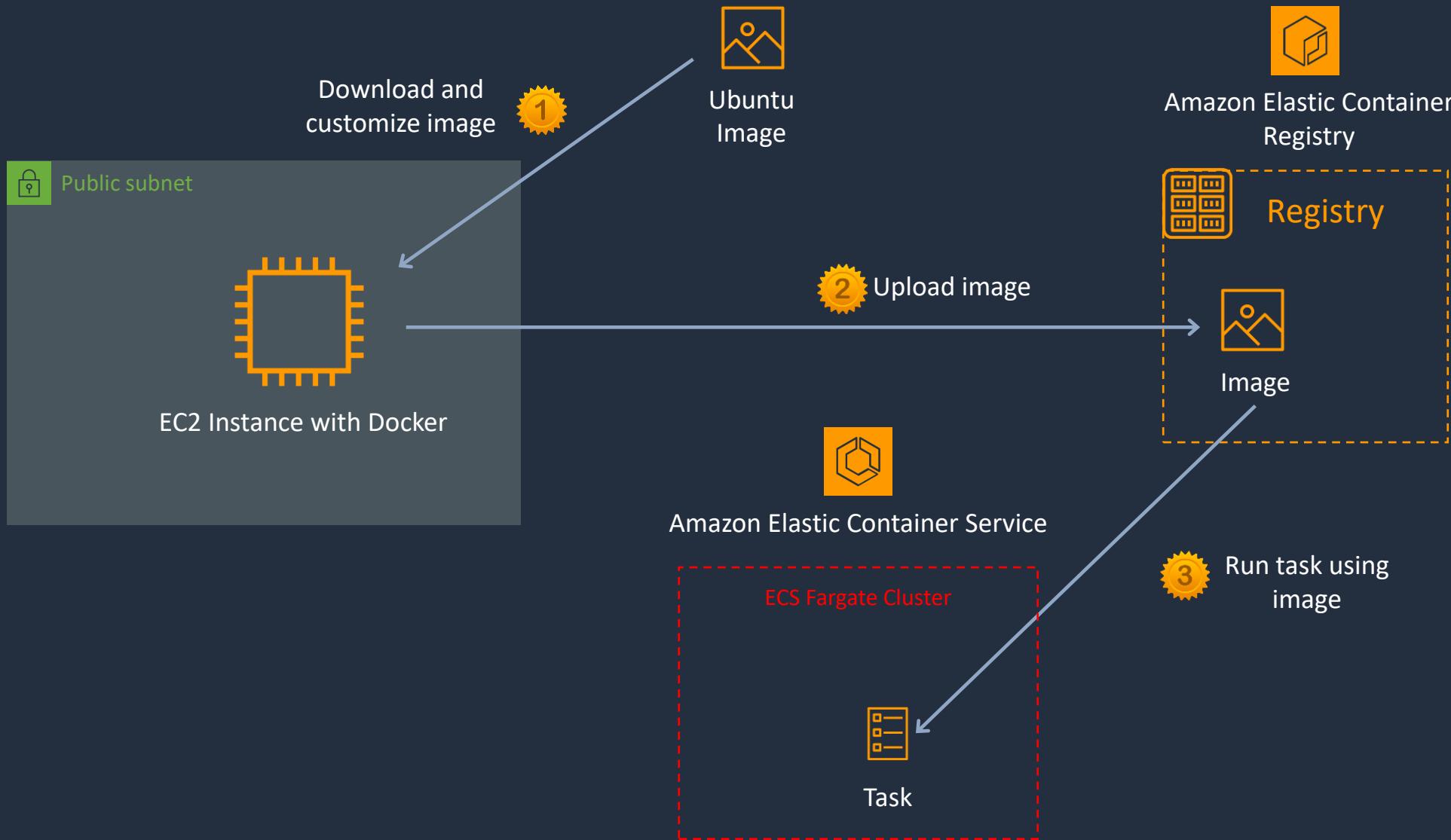


Amazon Elastic Container Registry (ECR)

- **Lifecycle policies** - manage the lifecycle of the images in your repositories
- **Image scanning** - identify software vulnerabilities in your container images
- **Cross-Region and cross-account replication** – replicate images across accounts/Region
- **Pull through cache rules** - cache repositories in remote public registries in your private Amazon ECR registry



Amazon Elastic Container Registry (ECR)





Pushing an Image to a Private Repository

- Users must have the following IAM permissions:

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "ecr:CompleteLayerUpload",  
                "ecr:GetAuthorizationToken",  
                "ecr:UploadLayerPart",  
                "ecr:InitiateLayerUpload",  
                "ecr:BatchCheckLayerAvailability",  
                "ecr:PutImage"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

The **Resource** element can also be used to scope to a specific repository ARN



Pushing an Image to a Private Repository

- First, authenticate the Docker client to ECR:

```
aws ecr get-login-password --region region | docker login --username AWS --password-stdin  
aws_account_id.dkr.ecr.region.amazonaws.com
```

- Tag your image with the Amazon ECR registry, repository, and image tag name to use:

```
docker tag e9ae3c220b23 aws_account_id.dkr.ecr.region.amazonaws.com/my-repository:tag
```

- Push the image using the docker push command:

```
docker push aws_account_id.dkr.ecr.region.amazonaws.com/my-repository:tag
```

AWS App Runner

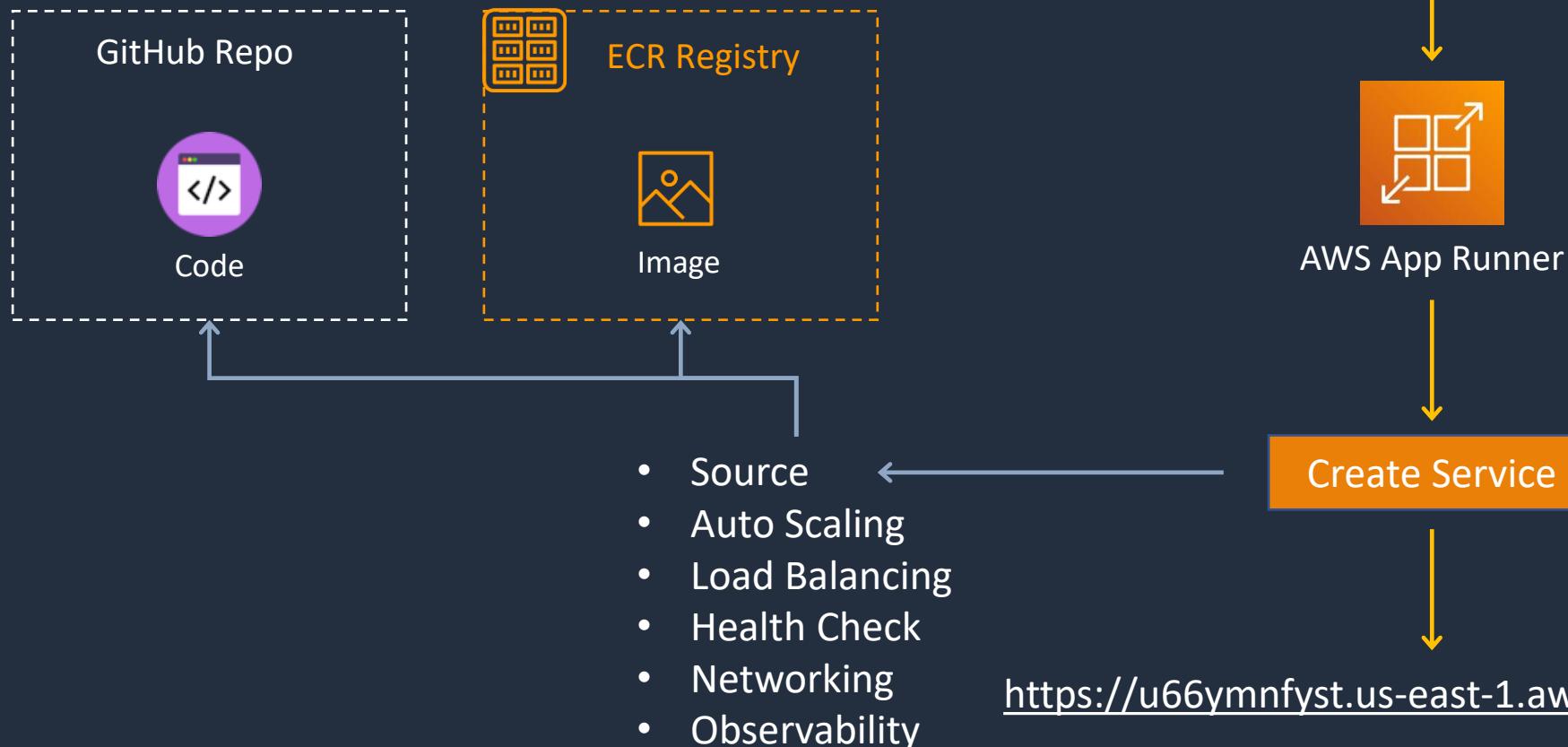




AWS App Runner

Fully managed service for deploying **containerized** web apps and APIs

```
aws apprunner create-service --service-name my-simple-service --source-configuration '{"ImageRepository": {"ImageIdentifier": "111111111111.dkr.ecr.us-east-1.amazonaws.com/nginx:latest", "ImageRepositoryType": "ECR_PUBLIC"}'}
```



PaaS solution with all components managed – just bring your code and container image!

Architecture Patterns – Amazon ECS





Architecture Patterns – Amazon ECS

Requirement

Application will be deployed on Amazon ECS and must scale based on memory

Application will run on Amazon ECS tasks across multiple hosts and needs access to an Amazon S3 bucket

Company requires standard Docker container automation and management service to be used across multiple environments

Solution

Use service auto-scaling and use the memory utilization

Use a task execution IAM role to provide permissions to S3 bucket.

Deploy Amazon EKS



Architecture Patterns – Amazon ECS

Requirement

Company plans to deploy Docker containers on AWS at the lowest cost

Company plans to migrate Docker containers to AWS and does not want to manage operating systems

Multiple microservices applications running on Amazon ECS. Need to route based on information in the HTTP header

Solution

Use Amazon ECS with a cluster of Spot instances and enable Spot instance draining

Migrate to Amazon ECS using the Fargate launch type

Deploy an Application Load Balancer in front of ECS and use query string parameter based routing



Requirement

Containerized app runs on Amazon EKS. Need to collect and centrally view metrics and logs including EKS namespaces and EKS services

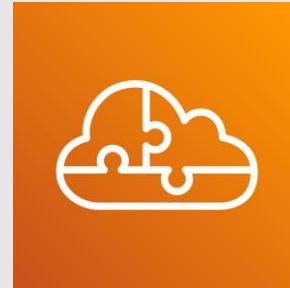
Solution

Configure CloudWatch Container Insights and view data in CloudWatch console

SECTION 11

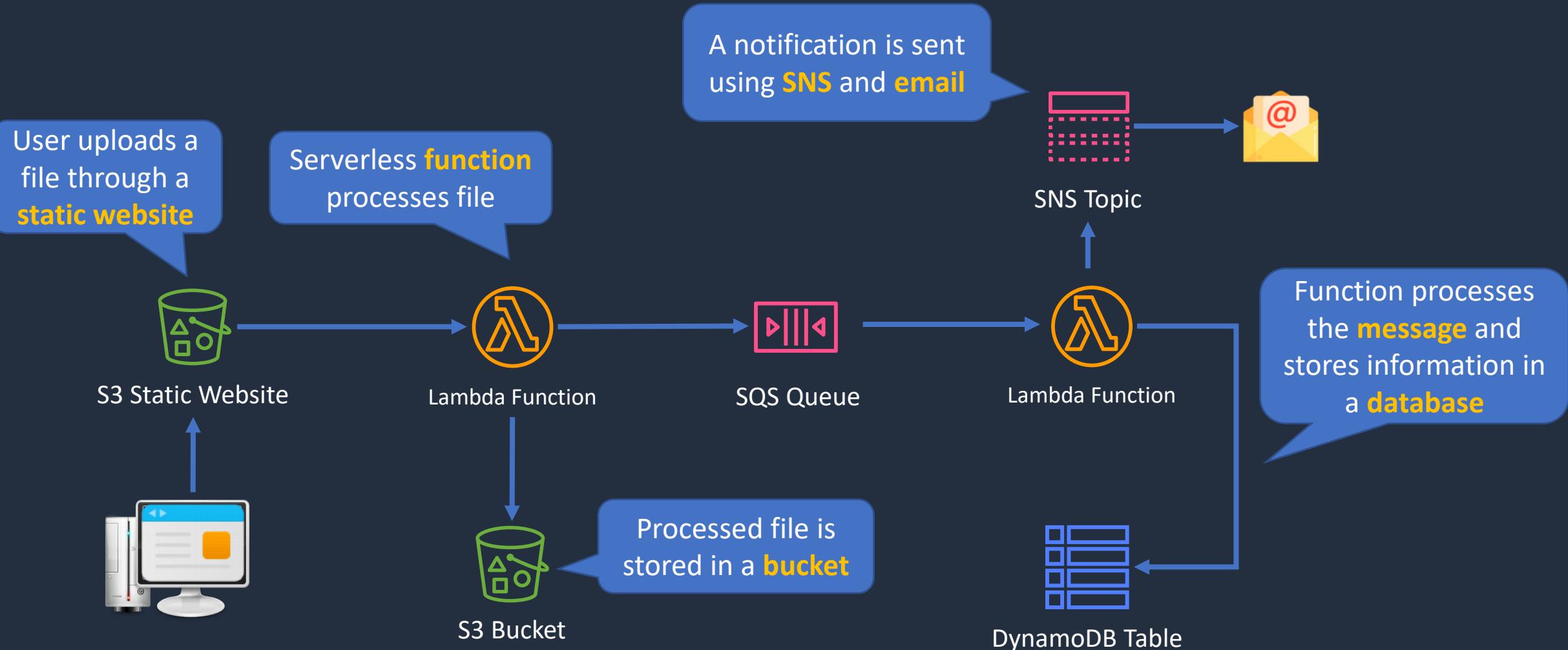
Serverless Applications

Serverless Services and Event-Driven Architecture





Serverless Services and Event-Driven Architecture





Serverless Services

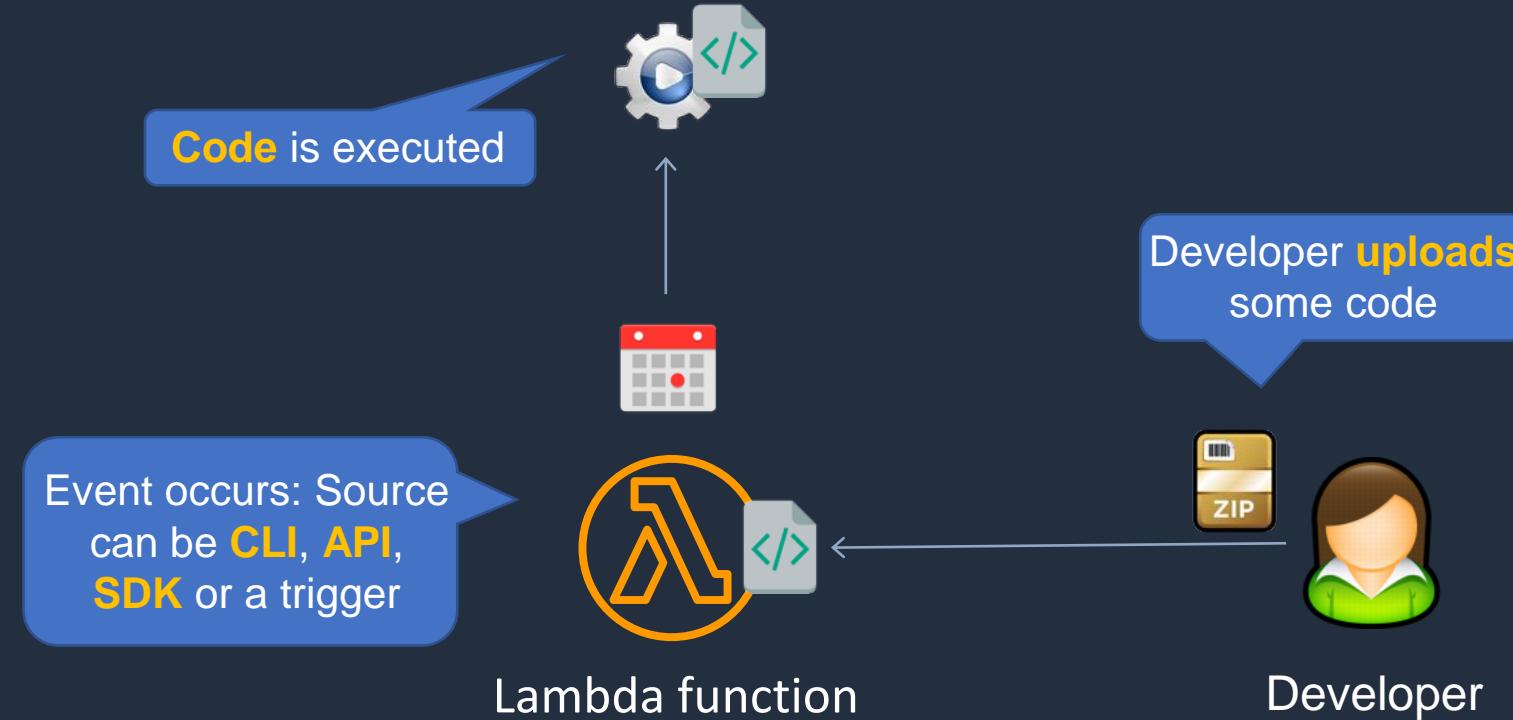
- With serverless there are **no instances** to manage
- You don't need to provision hardware
- There is no management of operating systems or software
- Capacity provisioning and patching is handled automatically
- Provides automatic scaling and high availability
- Can be very cheap!

AWS Lambda





AWS Lambda





AWS Lambda

- AWS Lambda executes code only when needed and scales automatically
- You pay only for the compute time you consume (you pay nothing when your code is not running)
- Benefits of AWS Lambda:
 - No servers to manage
 - Continuous scaling
 - Millisecond billing
 - Integrates with almost all other AWS services



AWS Lambda

- Primary use cases for AWS Lambda:
 - Data processing
 - Real-time file processing
 - Real-time stream processing
 - Build serverless backends for web, mobile, IOT, and 3rd party API requests



Lambda Function Invocations

Synchronous:

- CLI, SDK, API Gateway
- Wait for the function to process the event and return a response
- Error handling happens client side (retries, exponential backoff etc.)

Asynchronous:

- S3, SNS, CloudWatch Events etc.
- Event is queued for processing and a response is returned immediately
- Lambda retries up to 3 times
- Processing must be idempotent (due to retries)

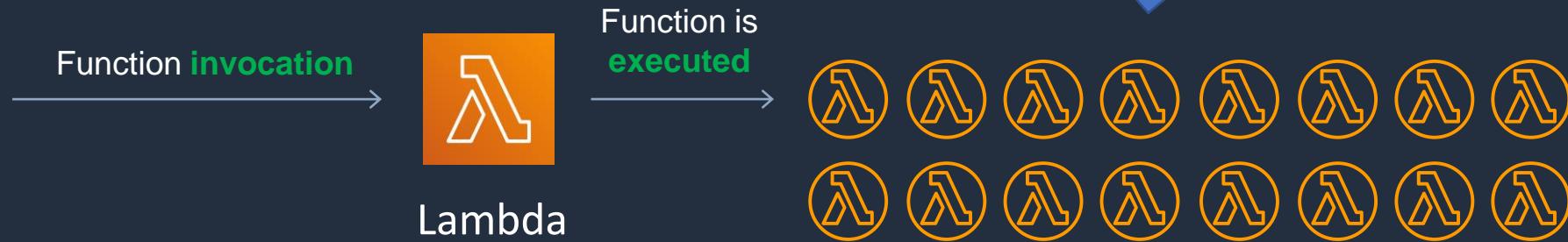
Event source mapping:

- SQS, Kinesis Data Streams, DynamoDB Streams
- Lambda does the polling (polls the source)
- Records are processed in order (except for SQS standard)

SQS can also trigger
Lambda



Lambda Function Concurrency

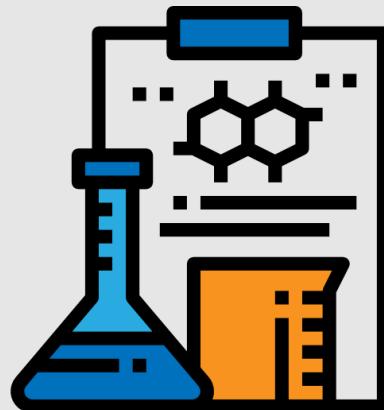


Burst **concurrency** quotas:

- 3000 – US West (Oregon), US East (N. Virginia), Europe (Ireland)
- 1000 – Asia Pacific (Tokyo), Europe (Frankfurt), US East (Ohio)
- 500 – Other Regions

If the concurrency **limit** is **exceeded** throttling occurs with error "**Rate exceeded**" and a 429 "**TooManyRequestsException**"

Create Function to Resize Instance



Application Integration Services Overview





Application Integration Services Overview

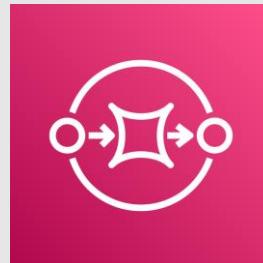
Service	What it does	Example use cases
Simple Queue Service	Messaging queue; store and forward patterns	Building distributed / decoupled applications
Simple Notification Service	Set up, operate, and send notifications from the cloud	Send email notification when CloudWatch alarm is triggered
Step Functions	Out-of-the-box coordination of AWS service components with visual workflow	Order processing workflow
Simple Workflow Service	Need to support external processes or specialized execution logic	Human-enabled workflows like an order fulfilment system or for procedural requests Note: AWS recommends that for new applications customers consider Step Functions instead of SWF
Amazon MQ	Message broker service for Apache Active MQ and RabbitMQ	Need a message queue that supports industry standard APIs and protocols; migrate queues to AWS
Amazon Kinesis	Collect, process, and analyze streaming data.	Collect data from IoT devices for later processing



Kinesis vs SQS vs SNS

Amazon Kinesis	Amazon SQS	Amazon SNS
Consumers pull data	Consumers pull data	Push data to many subscribers
As many consumers as you need	Data is deleted after being consumed	Publisher / subscriber model
Routes related records to same record processor	Can have as many workers (consumers) as you need	Integrates with SQS for fan-out architecture pattern
Multiple applications can access stream concurrently	No ordering guarantee (except with FIFO queues)	Up to 10,000,000 subscribers
Ordering at the shard level	Provides messaging semantics	Up to 100,000 topics
Can consume records in correct order at later time	Individual message delay	Data is not persisted
Must provision throughput	Dynamically scales	No need to provision throughput

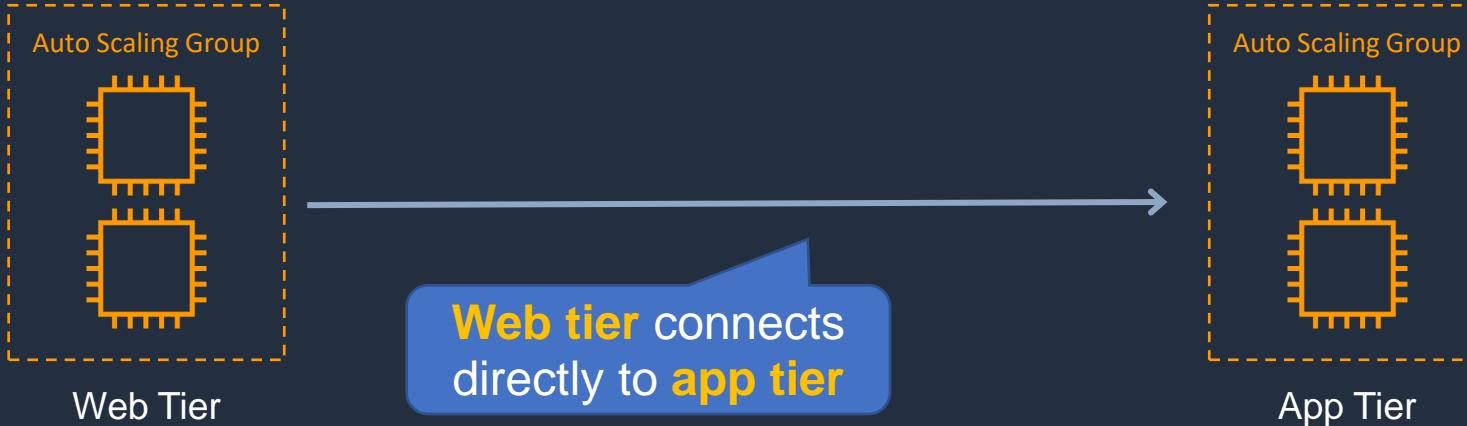
Amazon SQS



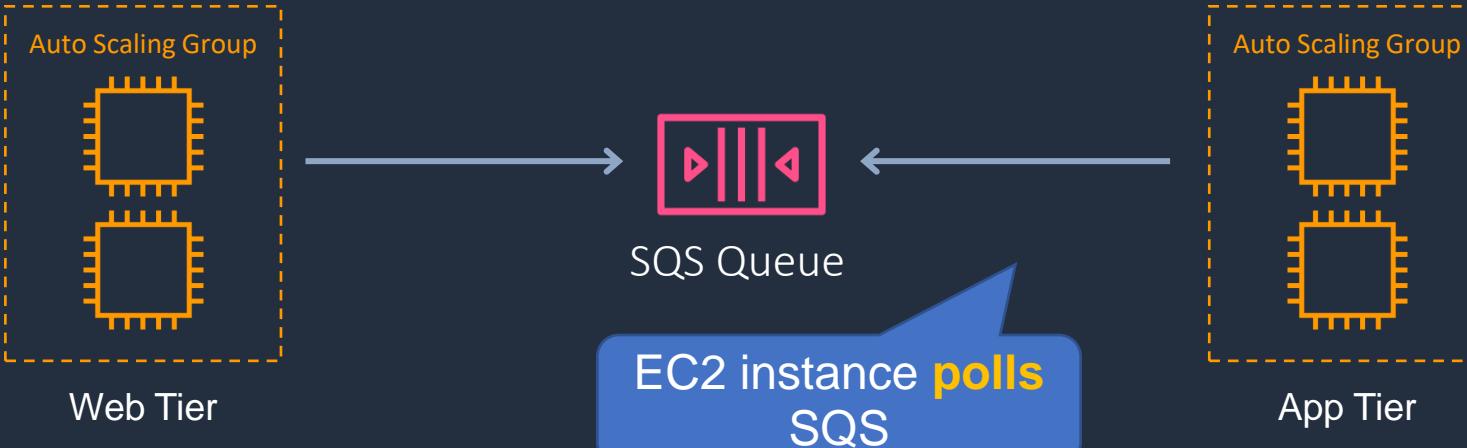


Decoupling with SQS Queues

Direct integration

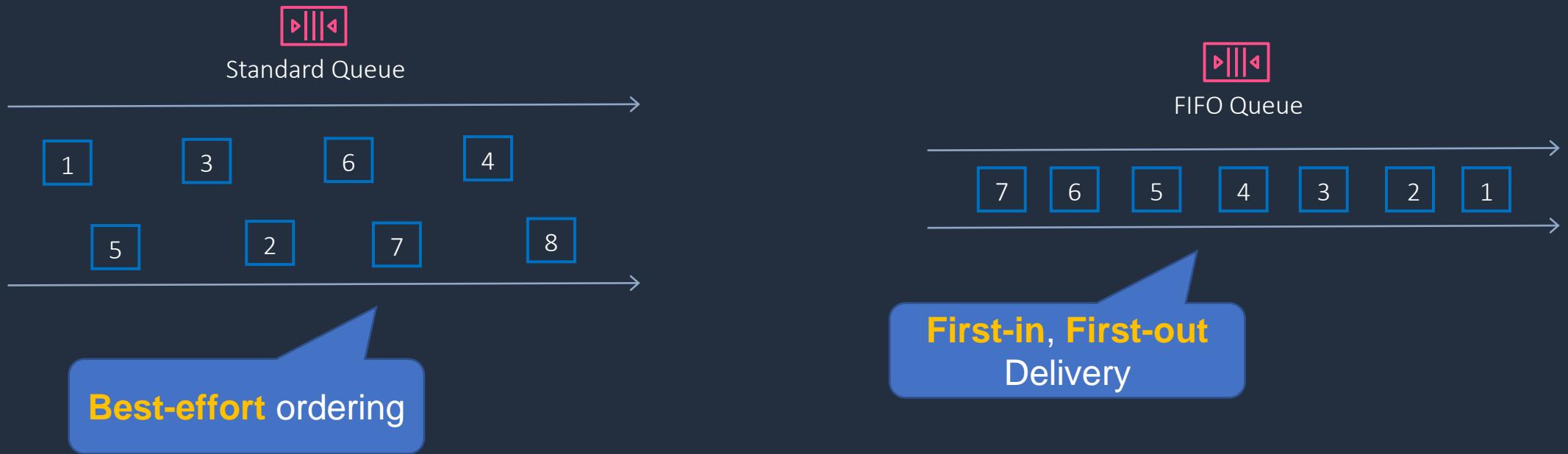


Decoupled integration





SQS Queue Types





SQS Queue Types

Standard Queue	FIFO Queue
Unlimited Throughput: Standard queues support a nearly unlimited number of transactions per second (TPS) per API action.	High Throughput: FIFO queues support up to 300 messages per second (300 send, receive, or delete operations per second). When you batch 10 messages per operation (maximum), FIFO queues can support up to 3,000 messages per second
Best-Effort Ordering: Occasionally, messages might be delivered in an order different from which they were sent	First-In-First-out Delivery: The order in which messages are sent and received is strictly preserved
At-Least-Once Delivery: A message is delivered at least once, but occasionally more than one copy of a message is delivered	Exactly-Once Processing: A message is delivered once and remains available until a consumer processes and deletes it. Duplicates are not introduced into the queue

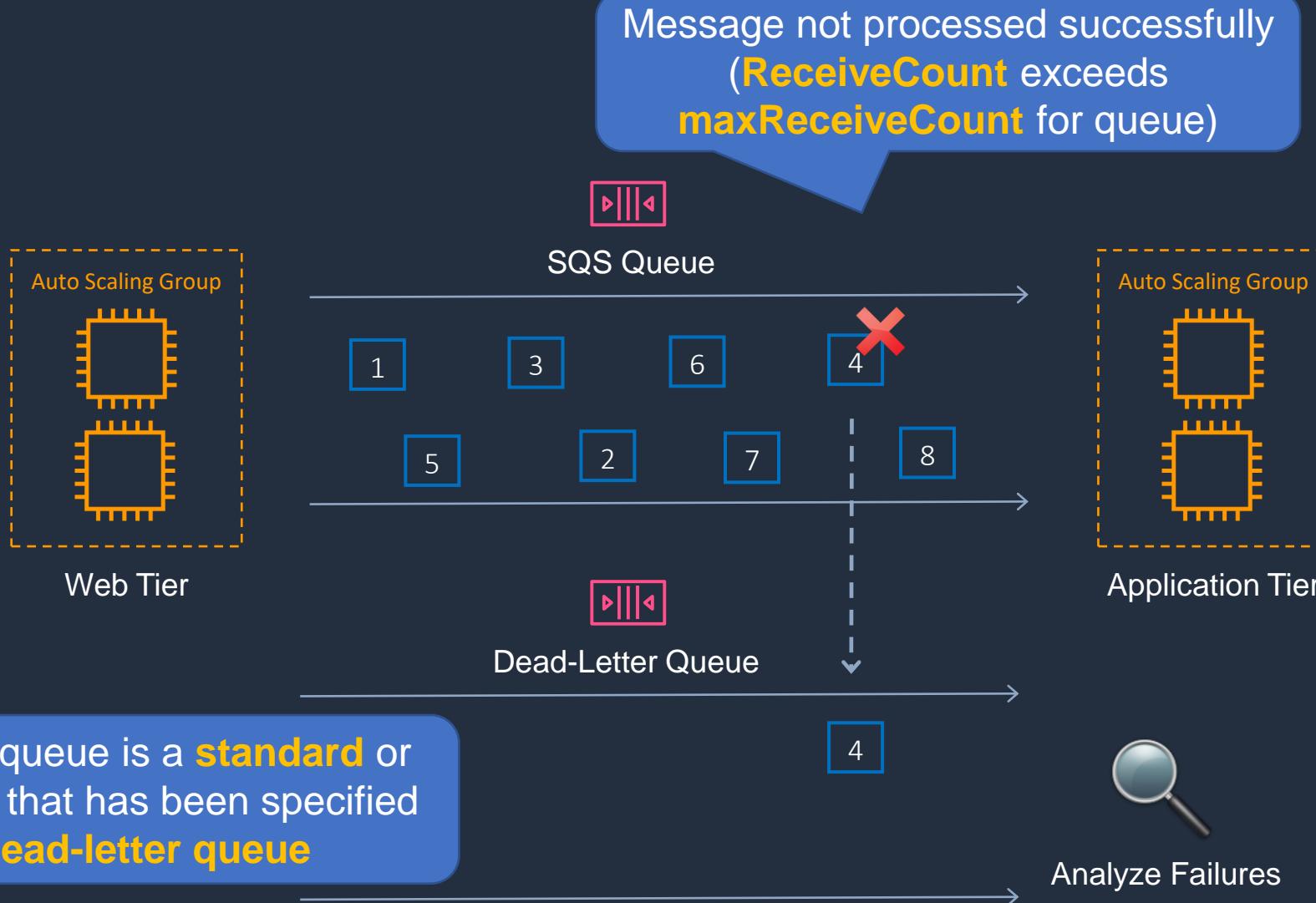


SQS Queue Types

- FIFO queues require the **Message Group ID** and **Message Deduplication ID** parameters to be added to messages
- **Message Group ID:**
 - The tag that specifies that a message belongs to a specific message group. Messages that belong to the same message group are guaranteed to be processed in a FIFO manner.
- **Message Deduplication ID:**
 - The token used for deduplication of messages within the deduplication interval.



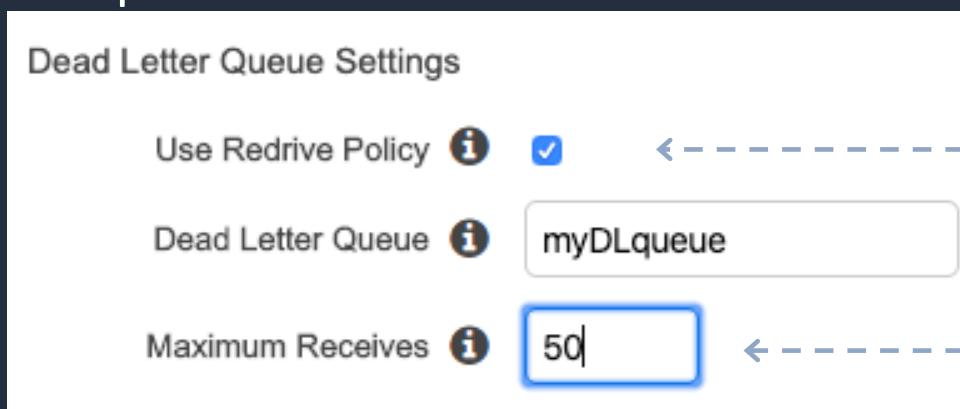
SQS – Dead Letter Queue





SQS – Dead Letter Queue

- The main task of a dead-letter queue is handling message failure
- A dead-letter queue lets you set aside and isolate messages that can't be processed correctly to determine why their processing didn't succeed
- It is not a queue type, it is a **standard** or **FIFO** queue that has been specified as a dead-letter queue in the configuration of **another** standard or FIFO queue



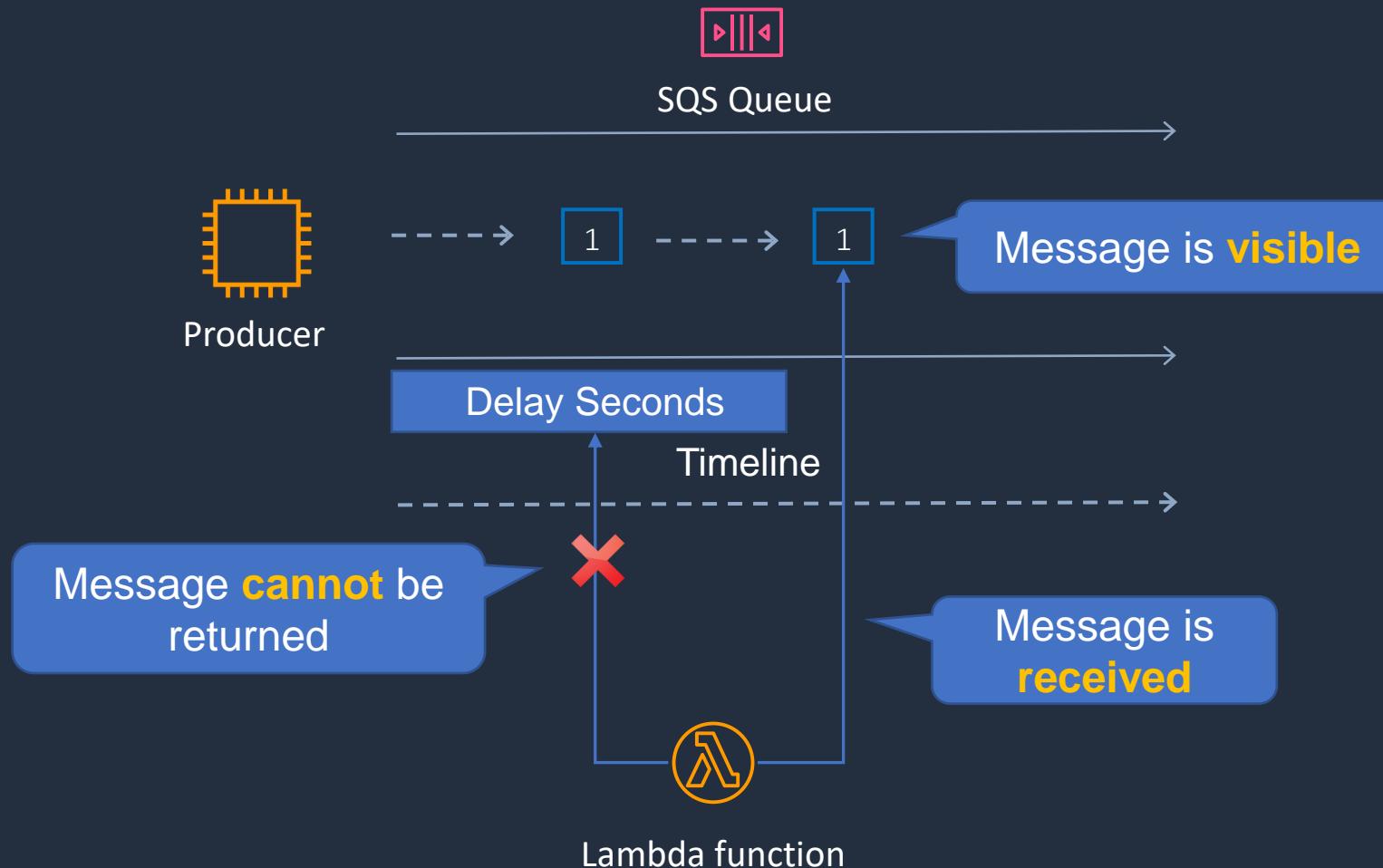
Enable Redrive Policy

Specify the queue to use as a dead-letter queue

Specify the maximum receives before a message is sent to the dead-letter queue



SQS – Delay Queue



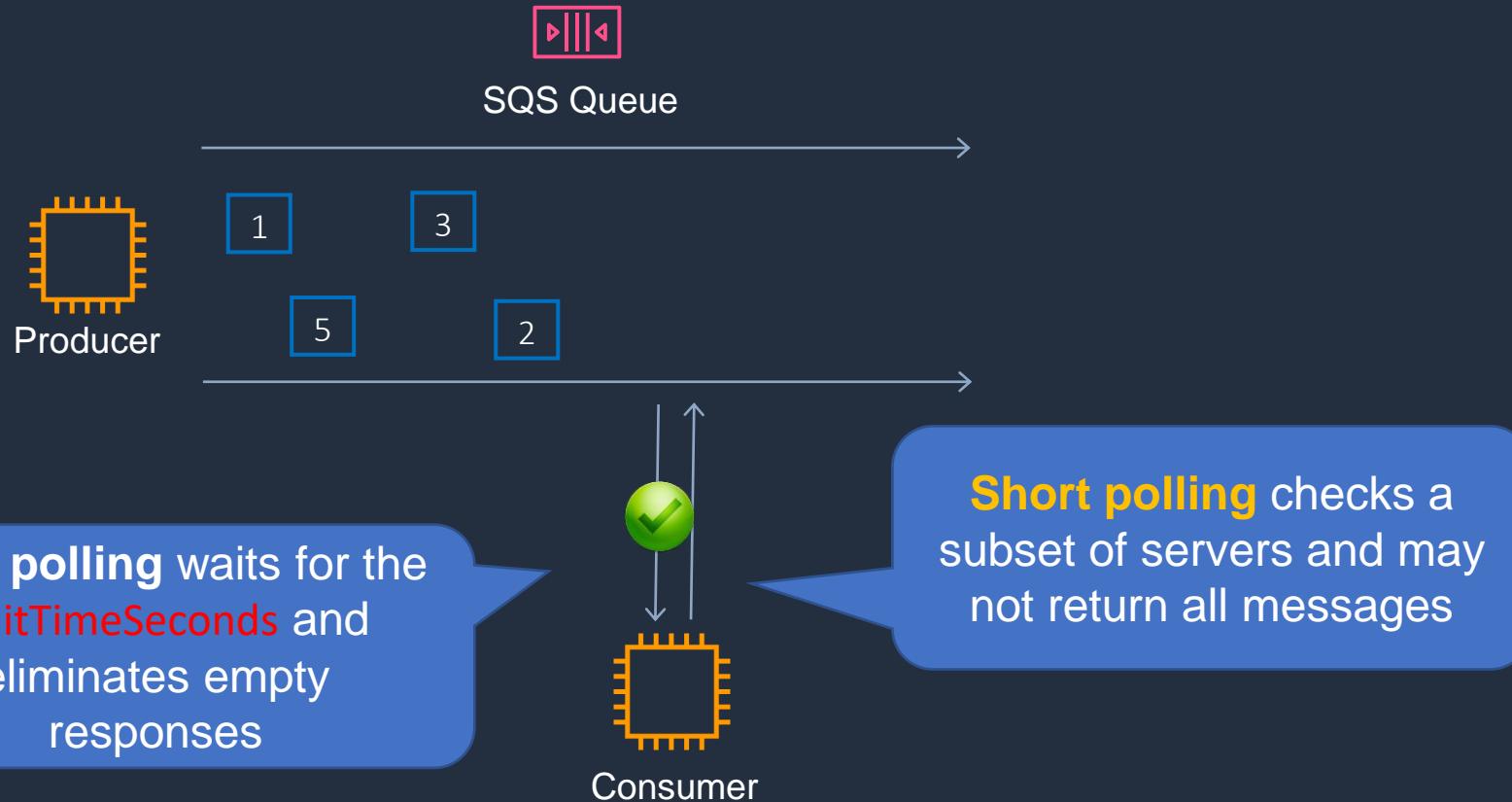
Delivery delay [Info](#)

Default 0s, max 15mins

0 Seconds



SQS Long Polling vs Short Polling



Long polling waits for the `WaitTimeSeconds` and eliminates empty responses

Short polling checks a subset of servers and may not return all messages



SQS Long Polling vs Short Polling

- SQS **Long polling** is a way to retrieve messages from SQS queues – waits for messages to arrive
- SQS **Short polling** returns immediately (even if the message queue is empty)
- SQS Long polling can lower costs
- SQS Long polling can be enabled at the queue level or at the API level using **WaitTimeSeconds**
- SQS Long polling is in effect when the Receive Message Wait Time is a value greater than 0 seconds and up to 20 seconds

Receive Message Wait Time i 20 seconds

←-----

The maximum amount of time that a long polling receive call will wait for a message to become available before returning an empty response.

Amazon SNS





Amazon SNS Notifications



Event **producer** sends one message to one SNS **topic**

Transport Protocols



Subscribers



Lambda



Amazon Simple Queue Service



Web Application



Email



Text

Many subscribers listen for notifications

Each **subscriber** will get all the messages



Amazon SNS

- Amazon SNS is a highly available, durable, secure, fully managed pub/sub messaging service
- Amazon SNS provides topics for high-throughput, push-based, many-to-many messaging
- Publisher systems can fan out messages to a large number of subscriber endpoints:
- Endpoints include:
 - Amazon SQS queues
 - AWS Lambda functions
 - HTTP/S webhooks
 - Mobile push
 - SMS
 - Email



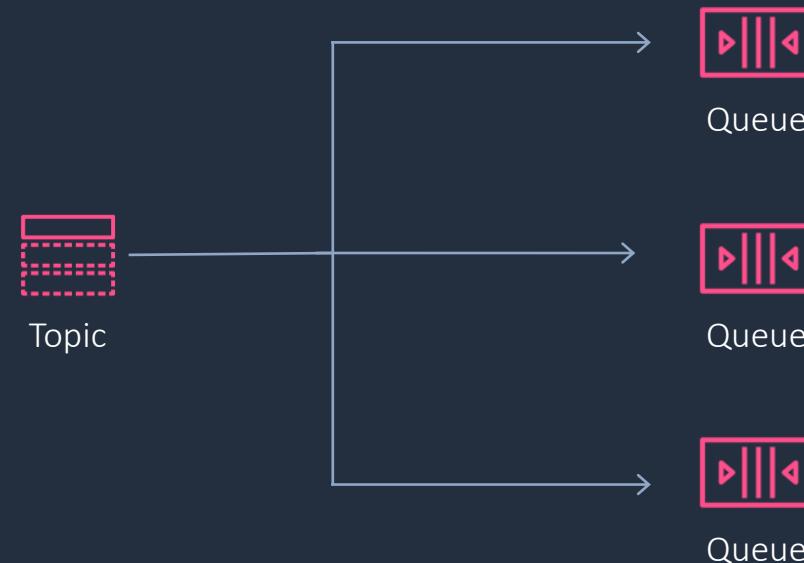
Amazon SNS

- Multiple recipients can be grouped using Topics
- A topic is an “access point” for allowing recipients to dynamically subscribe for identical copies of the same notification
- One topic can support deliveries to multiple endpoint types
- Simple APIs and easy integration with applications
- Flexible message delivery over multiple transport protocols



Amazon SNS + Amazon SQS Fan-Out

- You can subscribe one or more Amazon SQS queues to an Amazon SNS topic
- Amazon SQS manages the subscription and any necessary permissions
- When you publish a message to a topic, Amazon SNS sends the message to every subscribed queue



Simple Event-Driven App



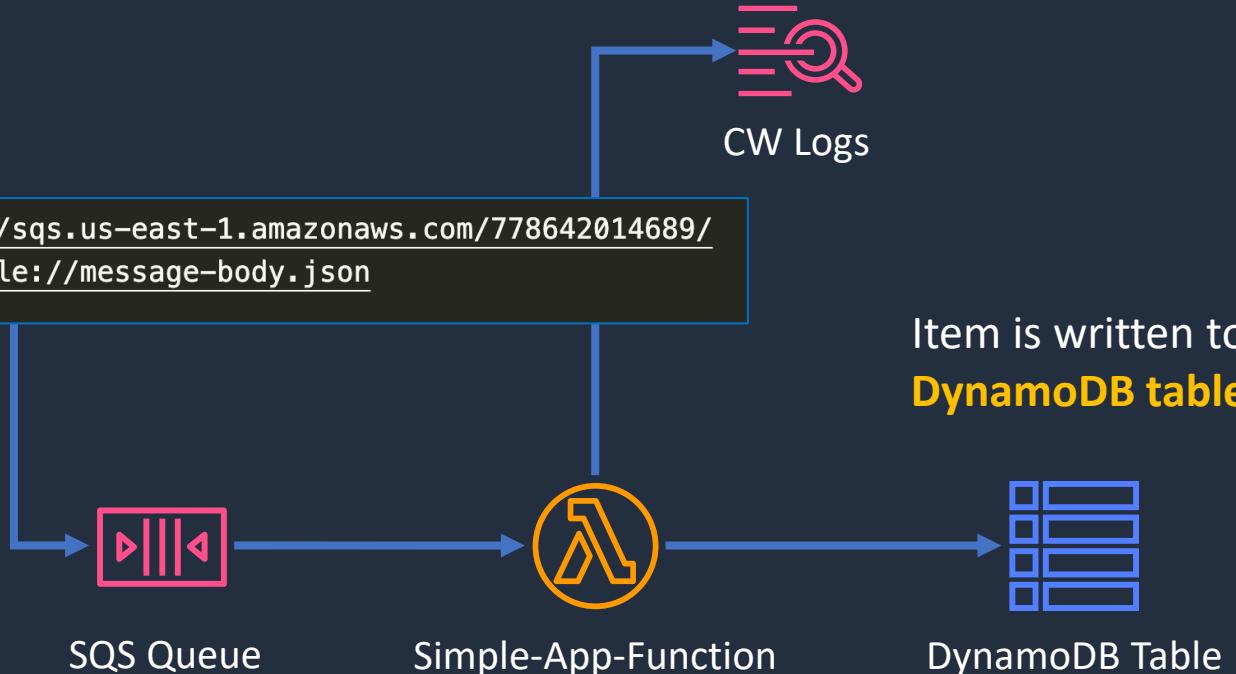


Simple Event-Driven App



```
aws sqs send-message --queue-url https://sns.us-east-1.amazonaws.com/778642014689/  
ProductVisitsDataQueue --message-body file://message-body.json
```

Manually add message to queue with **AWS CLI**



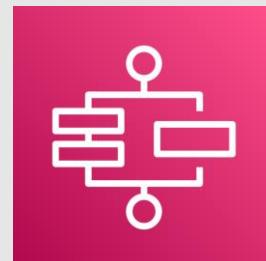
SQS triggers Lambda

JSON

```
{  
    "ProductId": "c96b49bb-c378-4a15-b2e3-842a9850b23d",  
    "ProductName": "Gloves",  
    "Category": "Accessories",  
    "PricePerUnit": "10",  
    "CustomerId": "be44af0a-74f9-438e-a3ac-e3e21d84259f",  
    "CustomerName": "John Doe",  
    "TimeOfVisit": "2021-02-21T16:23:42.389Z"  
}
```

This is the **message body**

AWS Step Functions





AWS Step Functions

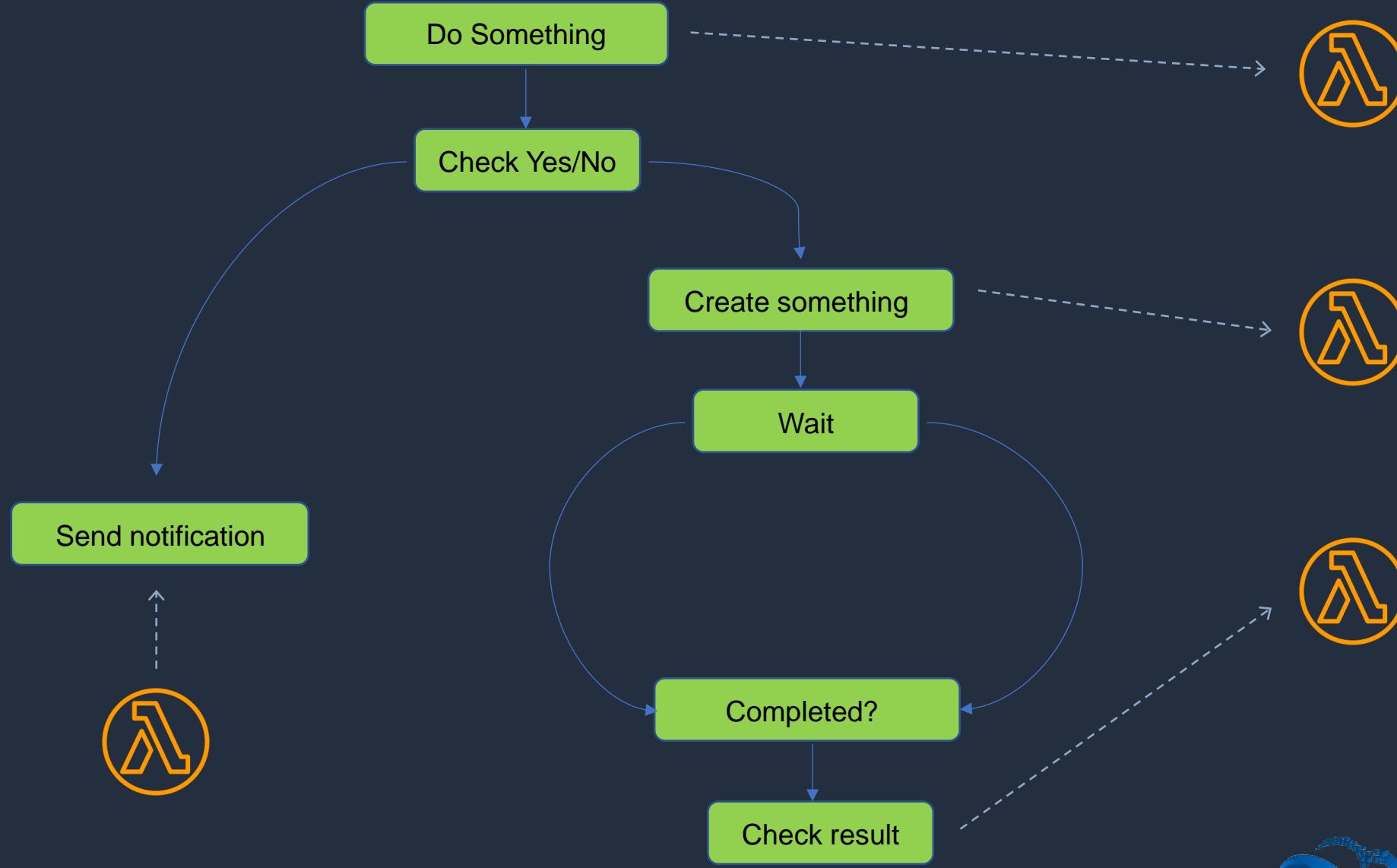
- AWS Step Functions is used to build distributed applications as a series of steps in a visual workflow
- You can quickly build and run state machines to execute the steps of your application

How it works:

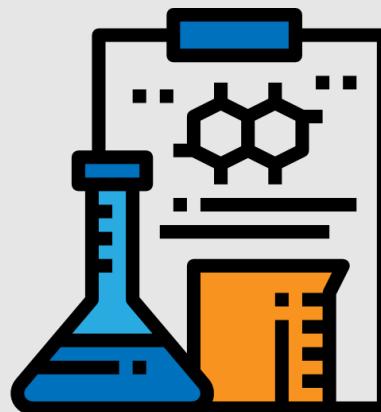
1. Define the steps of your workflow in the **JSON-based Amazon States Language**.
The visual console automatically graphs each step in the order of execution
2. Start an execution to visualize and verify the steps of your application are operating as intended. AWS Step Functions **operates and scales** the steps of your **application** and **underlying compute** for you to help ensure your application executes reliably under increasing demand



AWS Step Functions



Create a State Machine



Amazon EventBridge

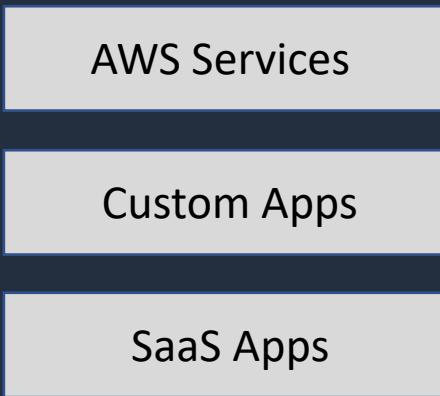




Amazon EventBridge

EventBridge used to be known as **CloudWatch Events**

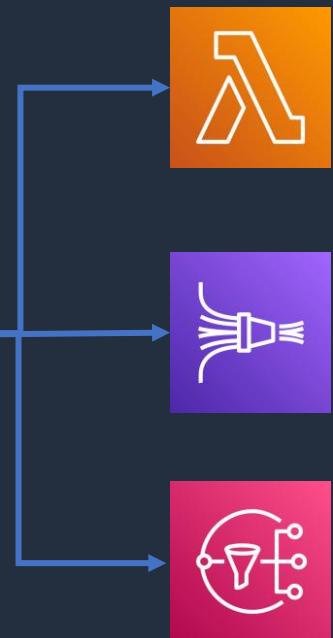
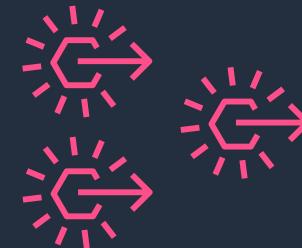
Event Sources



Events

EventBridge
event bus

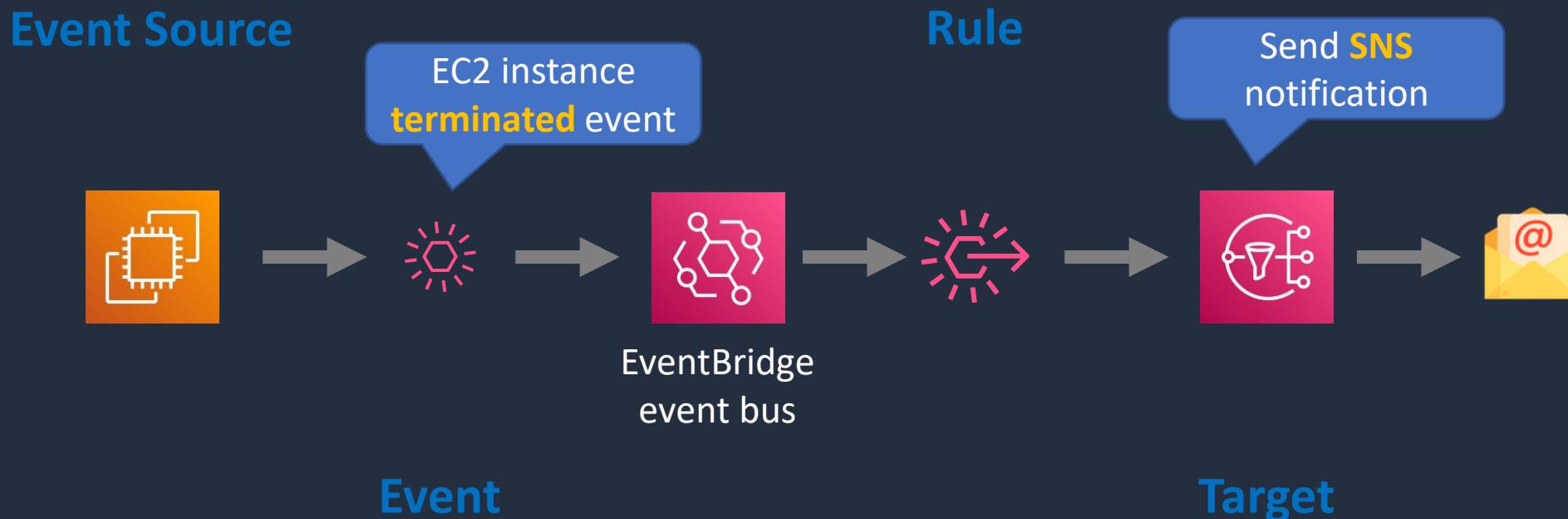
Rules



Targets



Amazon EventBridge Example 1





Amazon EventBridge Example 1

Event matching pattern
You can use pre-defined pattern provided by a service or create a custom pattern

Pre-defined pattern by service
 Custom pattern

Service provider
AWS services or custom/partner services

AWS

Service name
The name of partner service selected as the event source

EC2

Event type
The type of events as the source of the matching pattern

EC2 Instance State-change Notification

Any state
 Specific state(s)

terminated X

Any instance
 Specific instance Id(s)

i-1234567890abcdef0

Event pattern

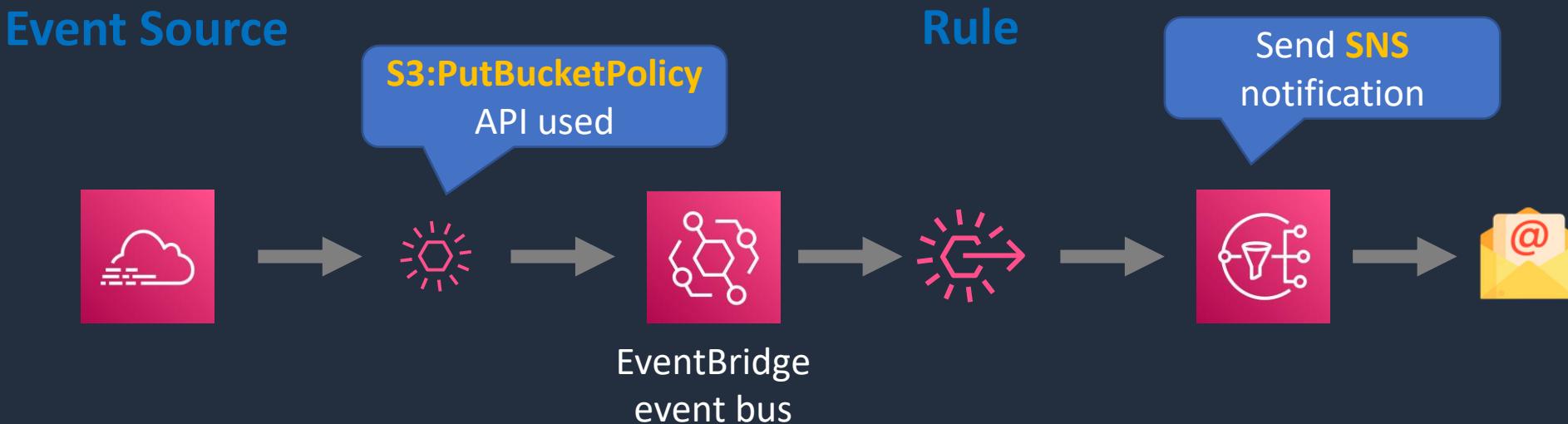
Copy Edit

```
1 {
2     "source": ["aws.ec2"],
3     "detail-type": ["EC2 Instance State-change Notification"]
4     "detail": {
5         "state": ["terminated"],
6         "instance-id": ["i-1234567890abcdef0"]
7     }
8 }
```

```
{
    "version": "0",
    "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
    "detail-type": "EC2 Instance State-change Notification",
    "source": "aws.ec2",
    "account": "111122223333",
    "time": "2017-12-22T18:43:48Z",
    "region": "us-west-1",
    "resources": [
        "arn:aws:ec2:us-west-1:123456789012:instance/i-1234567890abcdef0"
    ],
    "detail": {
        "instance-id": "i-1234567890abcdef0",
        "state": "terminated"
    }
}
```



Amazon EventBridge Example 2



Event

Target

```
{  
  "source": ["aws.cloudtrail"],  
  "detail-type": ["AWS API Call via CloudTrail"],  
  "detail": {  
    "eventSource": ["clouptrail.amazonaws.com"],  
    "eventName": ["s3:PutBucketPolicy"]  
  }  
}
```

Create Event Bus and Rule





Amazon EventBridge Example 1

Event matching pattern
You can use pre-defined pattern provided by a service or create a custom pattern

Pre-defined pattern by service
 Custom pattern

Service provider
AWS services or custom/partner services

AWS

Service name
The name of partner service selected as the event source

EC2

Event type
The type of events as the source of the matching pattern

EC2 Instance State-change Notification

Any state
 Specific state(s)

terminated X

Any instance
 Specific instance Id(s)

i-1234567890abcdef0

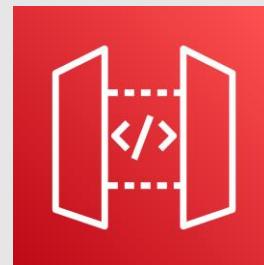
Event pattern

Copy Edit

```
1 {
2     "source": ["aws.ec2"],
3     "detail-type": ["EC2 Instance State-change Notification"]
4     "detail": {
5         "state": ["terminated"],
6         "instance-id": ["i-1234567890abcdef0"]
7     }
8 }
```

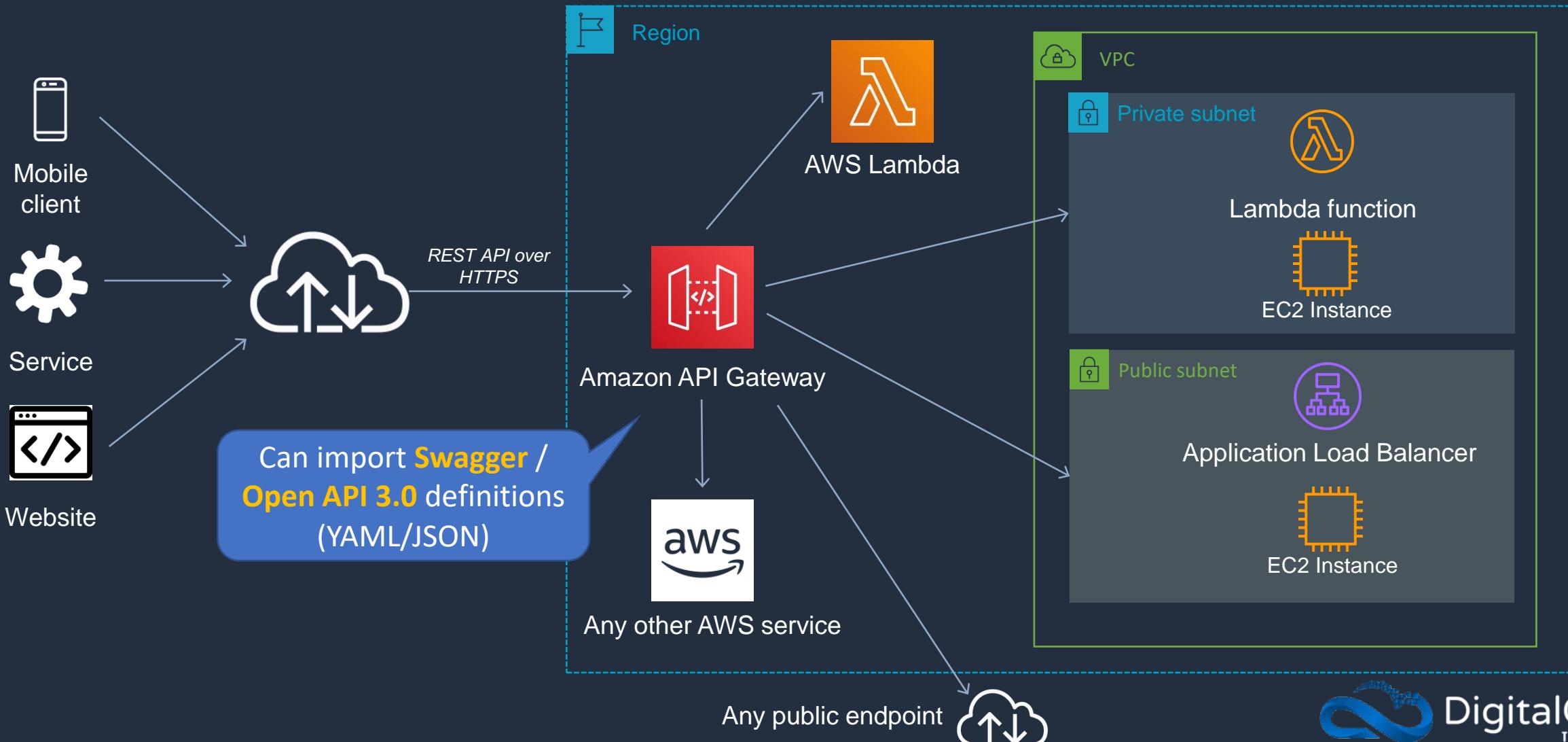
```
{
    "version": "0",
    "id": "6a7e8feb-b491-4cf7-a9f1-bf3703467718",
    "detail-type": "EC2 Instance State-change Notification",
    "source": "aws.ec2",
    "account": "111122223333",
    "time": "2017-12-22T18:43:48Z",
    "region": "us-west-1",
    "resources": [
        "arn:aws:ec2:us-west-1:123456789012:instance/i-1234567890abcdef0"
    ],
    "detail": {
        "instance-id": "i-1234567890abcdef0",
        "state": "terminated"
    }
}
```

Amazon API Gateway





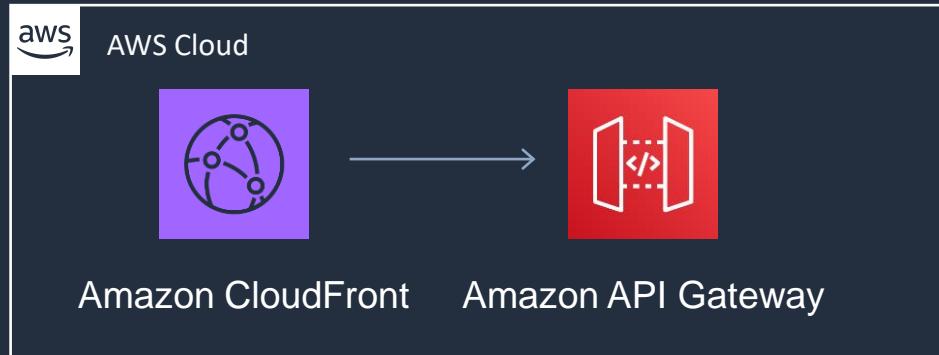
Amazon API Gateway Overview





Amazon API Gateway Deployment Types

Edge-optimized endpoint



Regional endpoint



Private endpoint



Key benefits:

- Reduced latency for requests from around the world

Key benefits:

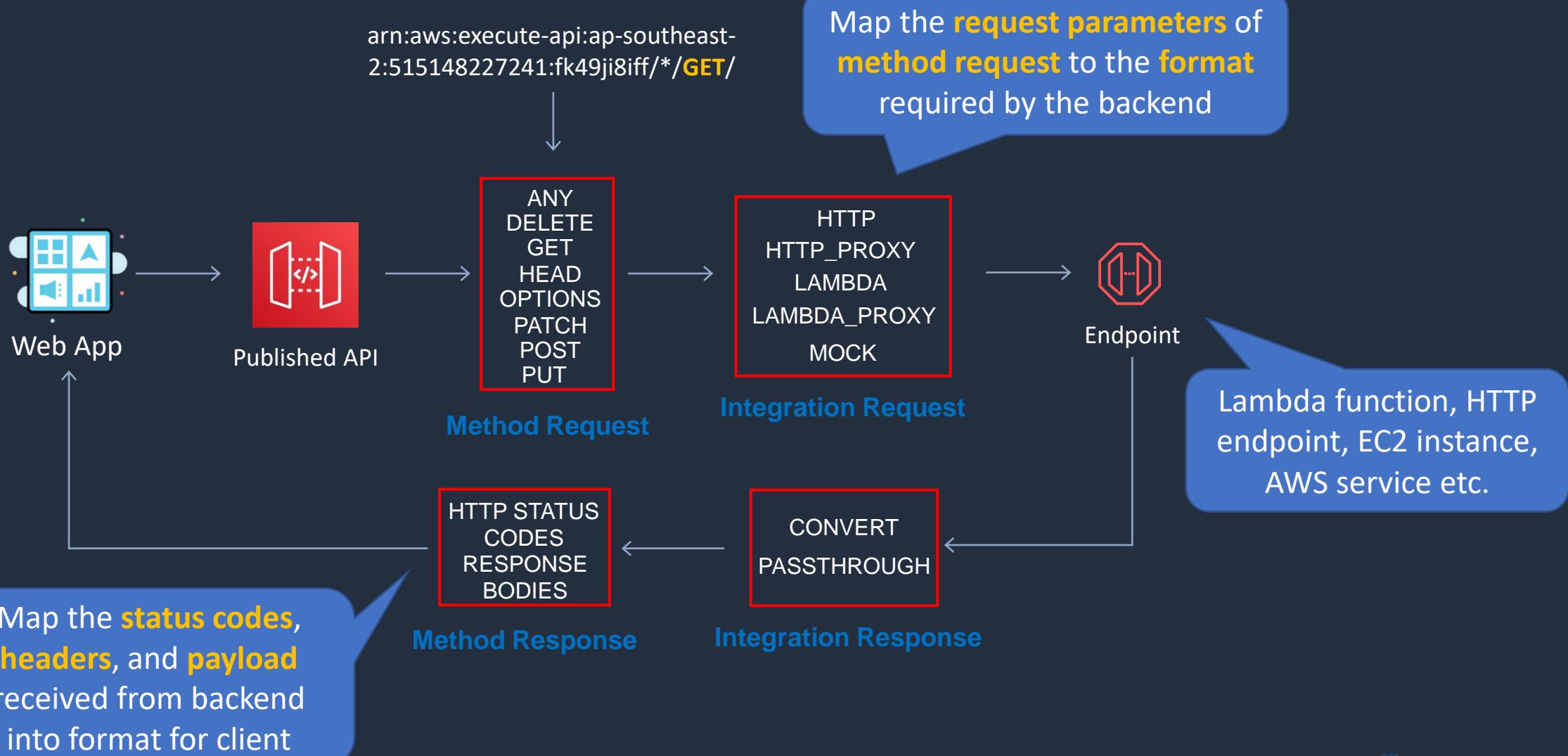
- Reduced latency for requests that originate in the same region
- Can also configure your own CDN and protect with WAF

Key benefits:

- Securely expose your REST APIs only to other services within your VPC or connect via Direct Connect



Structure of a REST API





API Gateway Integrations

For a **Lambda function** you can have:

- Lambda proxy integration
- Lambda custom integration

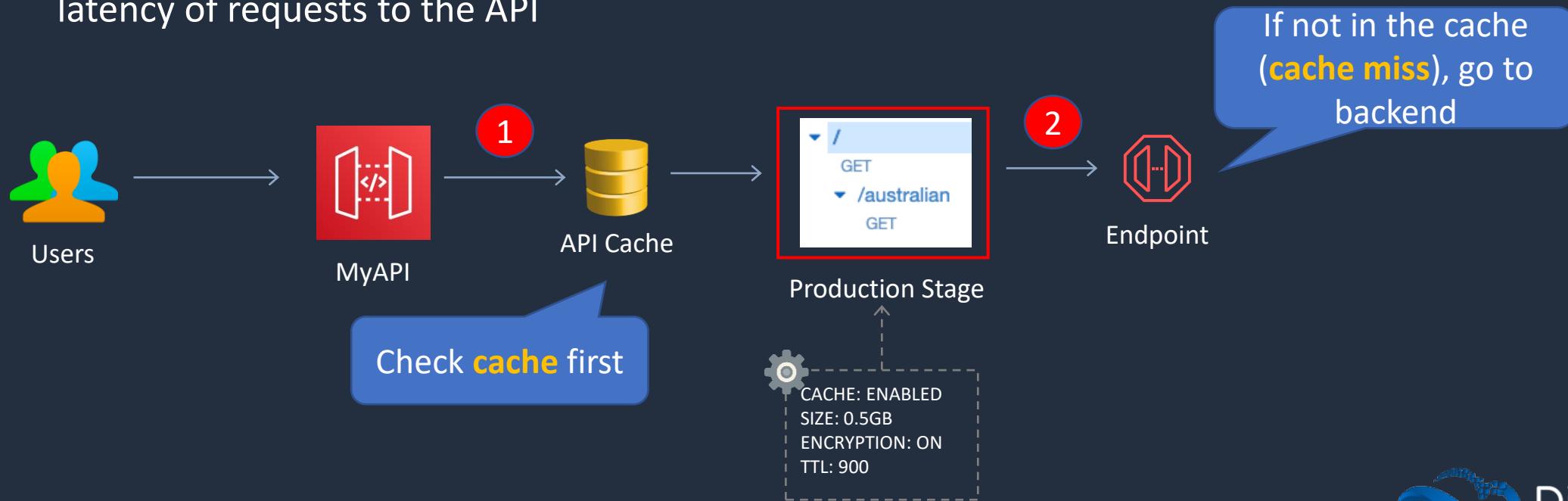
For an **HTTP endpoint** you can have:

- HTTP proxy integration
- HTTP custom integration
- For an **AWS service action** you have the AWS integration of the non-proxy type only



API Gateway - Caching

- You can add caching to API calls by provisioning an Amazon API Gateway cache and specifying its size in gigabytes
- Caching allows you to cache the endpoint's response
- Caching can reduce number of calls to the backend and improve latency of requests to the API



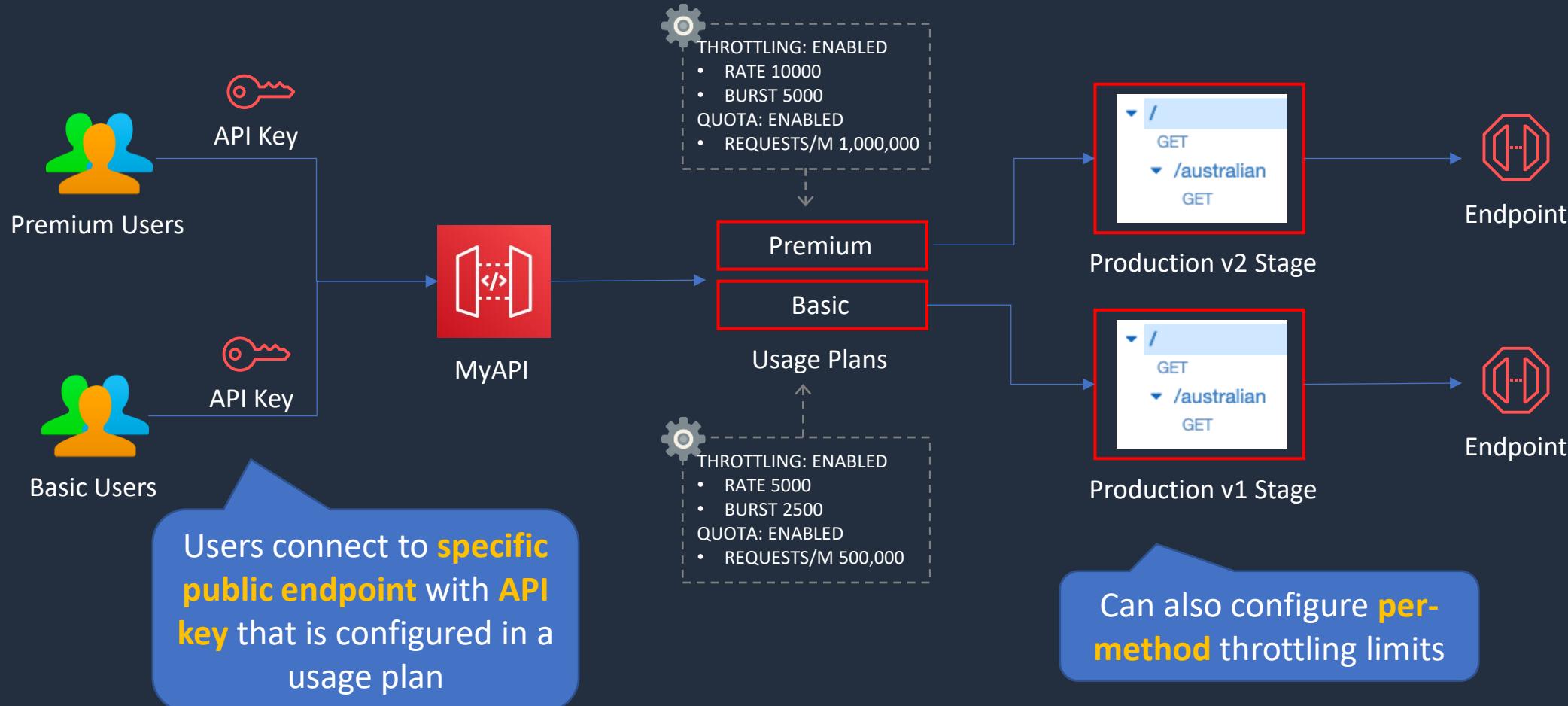


API Gateway - Throttling

- API Gateway sets a limit on a steady-state rate and a burst of request submissions against all APIs in your account
- Limits:
 - By default API Gateway limits the steady-state request rate to 10,000 requests per second
 - The maximum concurrent requests is 5,000 requests across all APIs within an AWS account
 - If you go over 10,000 requests per second or 5,000 concurrent requests you will receive a **429 Too Many Requests** error response
- Upon catching such exceptions, the client can resubmit the failed requests in a way that is rate limiting, while complying with the API Gateway throttling limits



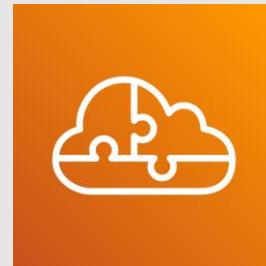
API Gateway – Usage Plans and API Keys



Simple HTTP API



Architecture Patterns - Serverless





Architecture Patterns - Serverless

Requirement

Application includes EC2 and RDS.
Spikes in traffic causing writes to be dropped by RDS

Solution

Decouple EC2 and RDS database with an SQS queue; use Lambda to process records in the queue

Web app includes a web tier and processing tier. Must be decoupled and processing tier should dynamically scale based on number of jobs

Decouple the web tier and processing tier with SQS and scale with Auto Scaling based on the queue length

Lambda function execution time has increased significantly as the payload size increased.

Optimize execution time by increasing memory available to the function which will proportionally increase CPU



Architecture Patterns - Serverless

Requirement

Statistical data is stored in RDS and will be accessed using a REST API. Demand will range from no traffic to sudden bursts of traffic and is unpredictable

New application processes customer orders and consists of multiple decoupled tiers. Orders must be processed in the order they are received

App uses API Gateway and Lambda. During busy periods, many requests fail multiple times before succeeding. No errors reported in Lambda

Solution

Create a REST API using Amazon API Gateway and integrate with an AWS Lambda function for connecting to RDS

Implement an Amazon SQS FIFO queue to preserve the record order

Throttle limit could be configured with a value that is too low. Increase the throttle limit



Architecture Patterns - Serverless

Requirement

EC2 instance processes images using JavaScript code and stores results in S3. Load is highly variable. Need a more cost-effective solution

App uses API Gateway regional REST API. Just gone global and performance has suffered

Legacy application uses many batch scripts that process data and pass on to next script. Complex and difficult to maintain

Solution

Replace EC2 with AWS Lambda function

Convert API to an edge-optimized API to optimize for the global user base

Migrate scripts to AWS Lambda functions and use AWS Step Functions to coordinate components



Architecture Patterns - Serverless

Requirement

Objects uploaded to an S3 bucket must be processed by AWS Lambda

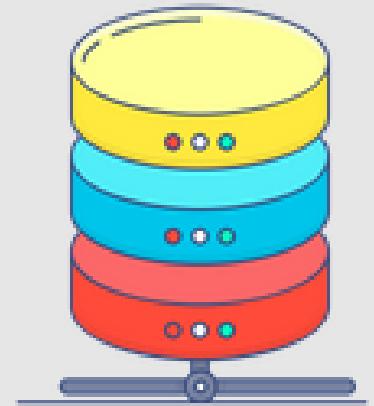
Solution

Create an event source notification to notify Lambda function to process new objects

SECTION 12

Databases and Analytics

Database Types and Use Cases





Relational vs Non-Relational

Key differences are how data are *managed* and how data are *stored*

Relational	Non-Relational
Organized by tables, rows and columns	Varied data storage models
Rigid schema (SQL)	Flexible schema (NoSQL) – data stored in key-value pairs, columns, documents or graphs
Rules enforced within database	Rules can be defined in application code (outside database)
Typically scaled vertically	Scales horizontally
Supports complex queries and joins	Unstructured, simple language that supports any kind of schema
Amazon RDS, Oracle, MySQL, IBM DB2, PostgreSQL	Amazon DynamoDB, MongoDB, Redis, Neo4j



Relational Databases

EmployeeID	FirstName	LastName	JobRole	Location
00001	Paul	Peterson	Senior Developer	Atlanta
00002	Kaleigh	Annette	Assistant Manager	Miami
00003	Carl	Wood	Sales Support	New York
00004	Vinni	Jones	Customer Service	Dallas
00005	Stefanie	Howard	IT Architect	Los Angeles

SQL is used for defining the structure of the database and its elements

SQL provides the tools for inserting, updating, deleting, and querying data within the database table

Example **Structured Query Language** (SQL) query:

```
SELECT FirstName  
FROM employees  
WHERE Location = Atlanta
```



Non-Relational Databases

NoSQL databases can be **key/value** and **document** stores

This is an example of a **key/value** store

Primary Key		Attributes				
Partition Key	Sort Key	sku	category	size	color	weight
john@example.com	1583975308	SKU-S523	T-Shirt	Small	Red	Light
chris@example.com	1583975613	SKU-J091	Pen		Blue	
chris@example.com	1583975449	SKU-A234	Mug			
sarah@example.com	1583976311	SKU-R873	Chair			
jenny@example.com	1583976323	SKU-I019	Plate	30		

There is no rigid schema so attributes can be missing or have different data types



Graph Databases

Graph databases like Amazon Neptune are designed to store, manage, and navigate relationships in data

Graph databases use:

- **Nodes** to represent entities
- **Edges** to represent relationships
- **Properties** to store information about nodes and edges





Operational vs Analytical

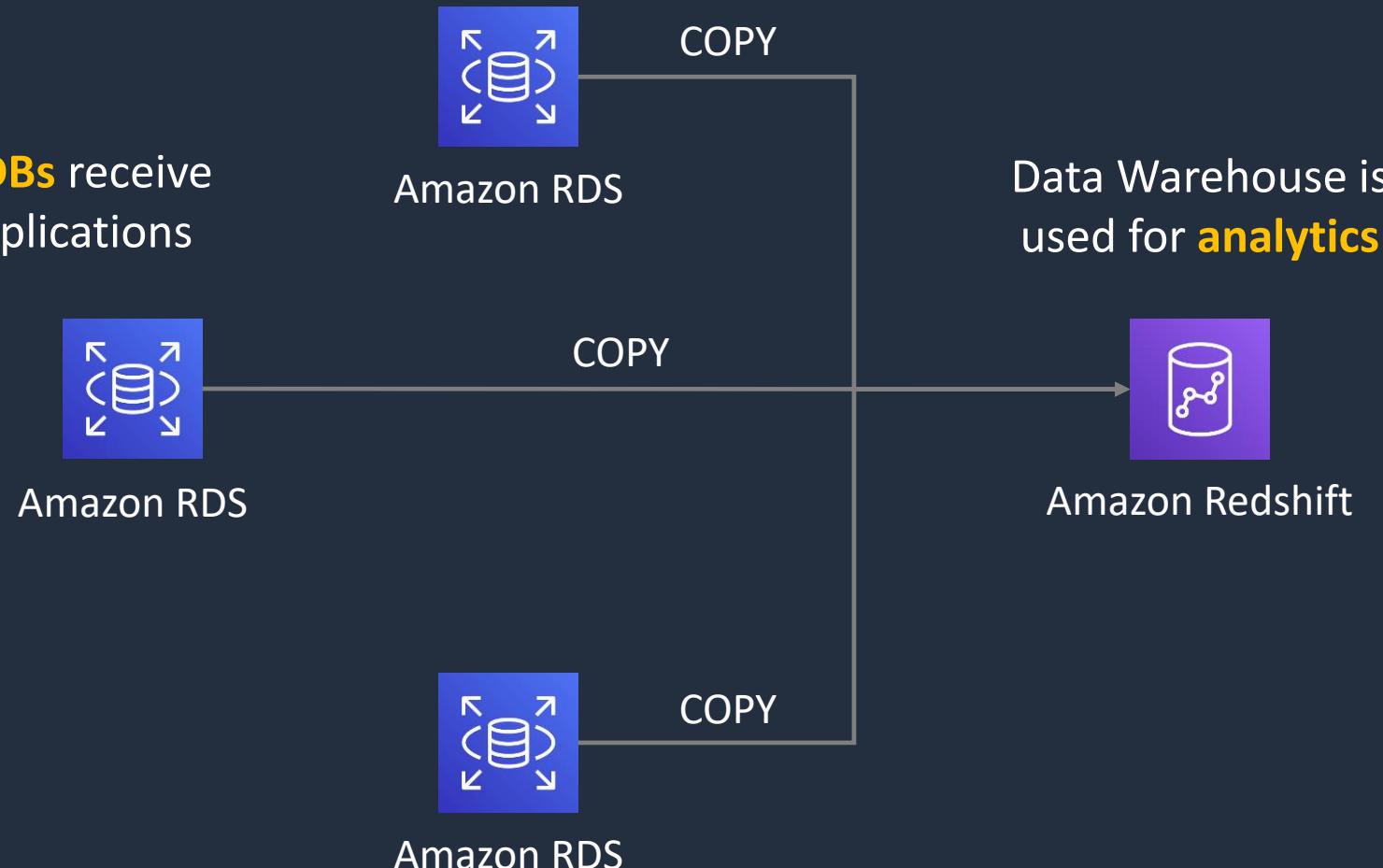
Key differences are **use cases** and how the database is **optimized**

Operational / transactional	Analytical
Online Transaction Processing (OLTP)	Online Analytics Processing (OLAP) – the source data comes from OLTP DBs
Production DBs that process transactions. E.g. adding customer records, checking stock availability (INSERT, UPDATE, DELETE)	Data warehouse. Typically, separated from the customer facing DBs. Data is extracted for decision making
Short transactions and simple queries	Long transactions and complex queries
Relational examples: Amazon RDS, Oracle, IBM DB2, MySQL	Relational examples: Amazon RedShift, Teradata, HP Vertica
Non-relational examples: MongoDB, Cassandra, Neo4j, HBase	Non-relational examples: Amazon EMR, MapReduce



Operational vs Analytical

Operational DBs receive data from applications





AWS Databases

Data Store	Use Case
Database on EC2	<ul style="list-style-type: none">• Need full control over instance and database• Third-party database engine (not available in RDS)
Amazon RDS	<ul style="list-style-type: none">• Need traditional relational database• e.g. Oracle, PostgreSQL, Microsoft SQL, MariaDB, MySQL• Data is well-formed and structured
Amazon DynamoDB	<ul style="list-style-type: none">• NoSQL database• In-memory performance• High I/O needs• Dynamic scaling
Amazon RedShift	<ul style="list-style-type: none">• Data warehouse for large volumes of aggregated data
Amazon ElastiCache	<ul style="list-style-type: none">• Fast temporary storage for small amounts of data• In-memory database

Amazon Relational Database Service (RDS)

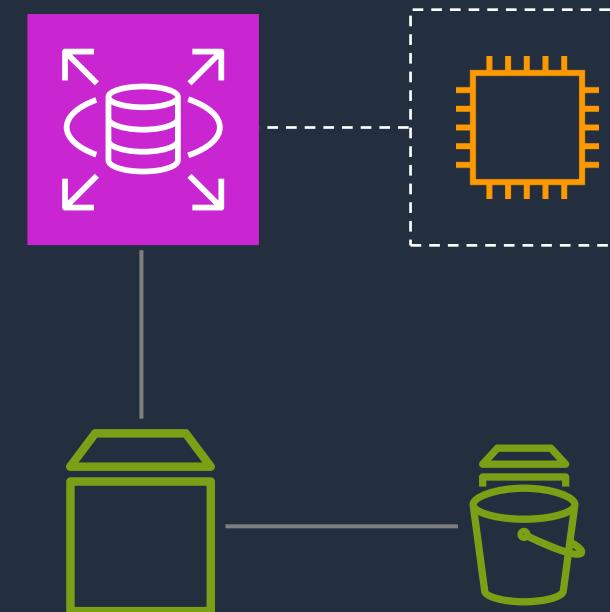




Amazon RDS

- Managed **relational** database service
- Used for online transaction processing (OLTP) use cases
- Runs on Amazon EC2 instances

A DB instance can contain
multiple user-created databases



You must choose the
DB instance type

RDS uses **Amazon EBS** volumes
for storage

Backups can be taken using
EBS snapshots



Amazon RDS

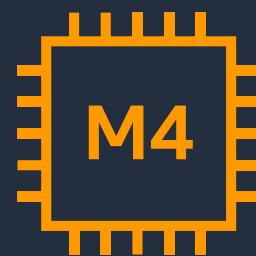
Amazon RDS offers a choice of database engines:

- **Amazon Aurora** – (MySQL and PostgreSQL compatible)
- **MySQL** – One of the most popular open-source relational database management systems
- **PostgreSQL** – An advanced open-source relational database that supports both SQL (relational) and JSON (non-relational) querying
- **Oracle** – Offers support for Oracle Database under two licensing models: "License Included" and "Bring Your Own License (BYOL)"
- **Microsoft SQL Server** – Supports several editions of Microsoft SQL Server
- **MariaDB** – A community-developed fork of MySQL intended to remain free under the GNU GPL



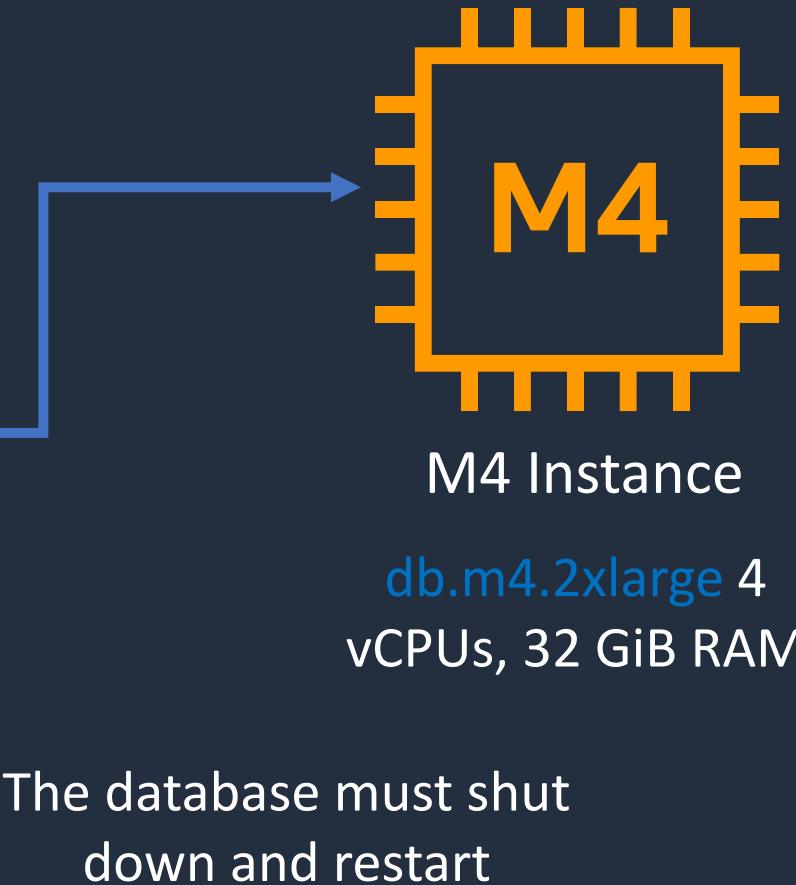
Amazon RDS – Scaling up (vertically)

RDS scales up by changing
the **instance type**



M4 instance

db.m4.large 2 vCPUs,
8 GiB RAM

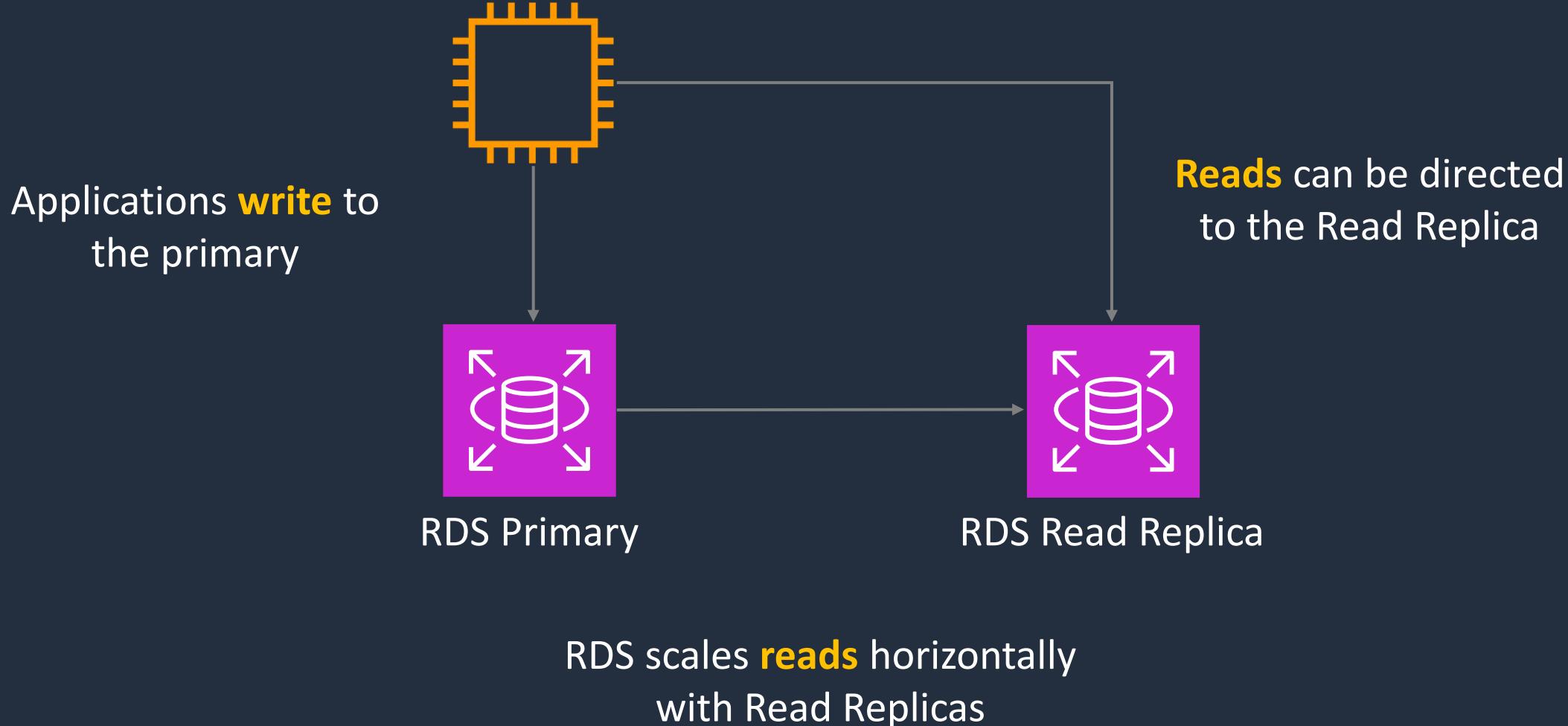


Scaling up is required for
better **write** performance

The database must shut
down and restart



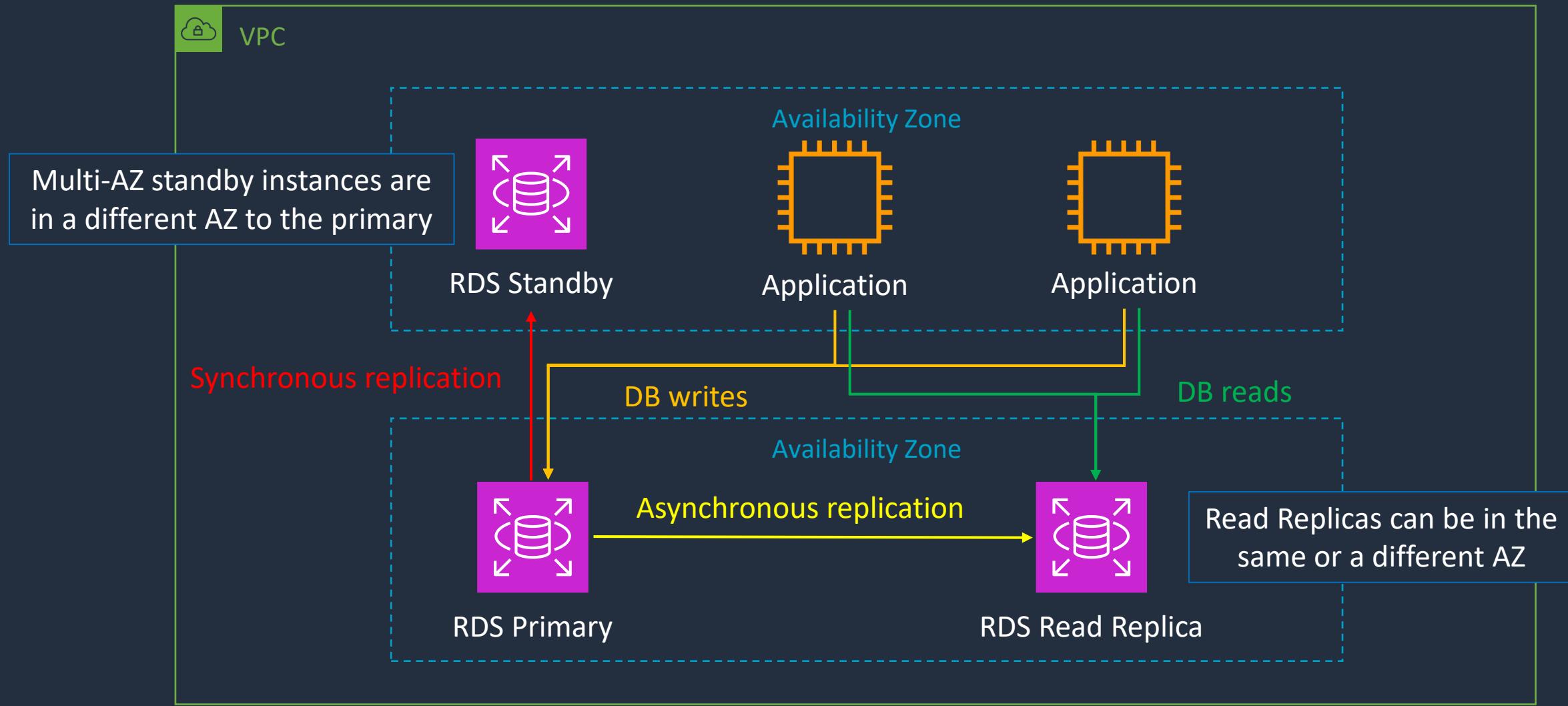
Amazon RDS – Scaling out (horizontally)





Amazon RDS Multi AZ

Multi-AZ deployments enable automatic **disaster recovery (DR)**



Amazon RDS Backup and Recovery





Amazon RDS Automated Backups

Backup window [Info](#)

Select the period you want automated backups of the database to be created by Amazon RDS.

Select window

No preference

Start time
00 : 00 UTC

Duration
0.5 hours



New DB Instance



DB Instance

Restore

Backup



Snapshot

Retention is
0–35 days



Amazon RDS Manual Backups (Snapshot)

- Backs up the entire DB instance, not just individual databases
- For single-AZ DB instances there is a brief suspension of I/O
- For Multi-AZ SQL Server, I/O activity is briefly suspended on primary
- For Multi-AZ MariaDB, MySQL, Oracle and PostgreSQL the snapshot is taken from the standby
- Snapshots do not expire (no retention period)



Amazon RDS Maintenance Windows

- Operating system and DB patching can require taking the database offline
- These tasks take place during a maintenance window
- By default a weekly maintenance window is configured
- You can choose your own maintenance window

Maintenance window [Info](#)

Select the period you want pending modifications or maintenance applied to the database by Amazon RDS.

Select window

No preference

Start day: Monday ▾

Start time: 00 ▾ : 00 ▾ UTC

Duration: 0.5 ▾ hours

Create Amazon RDS Database



Create a Read Replica

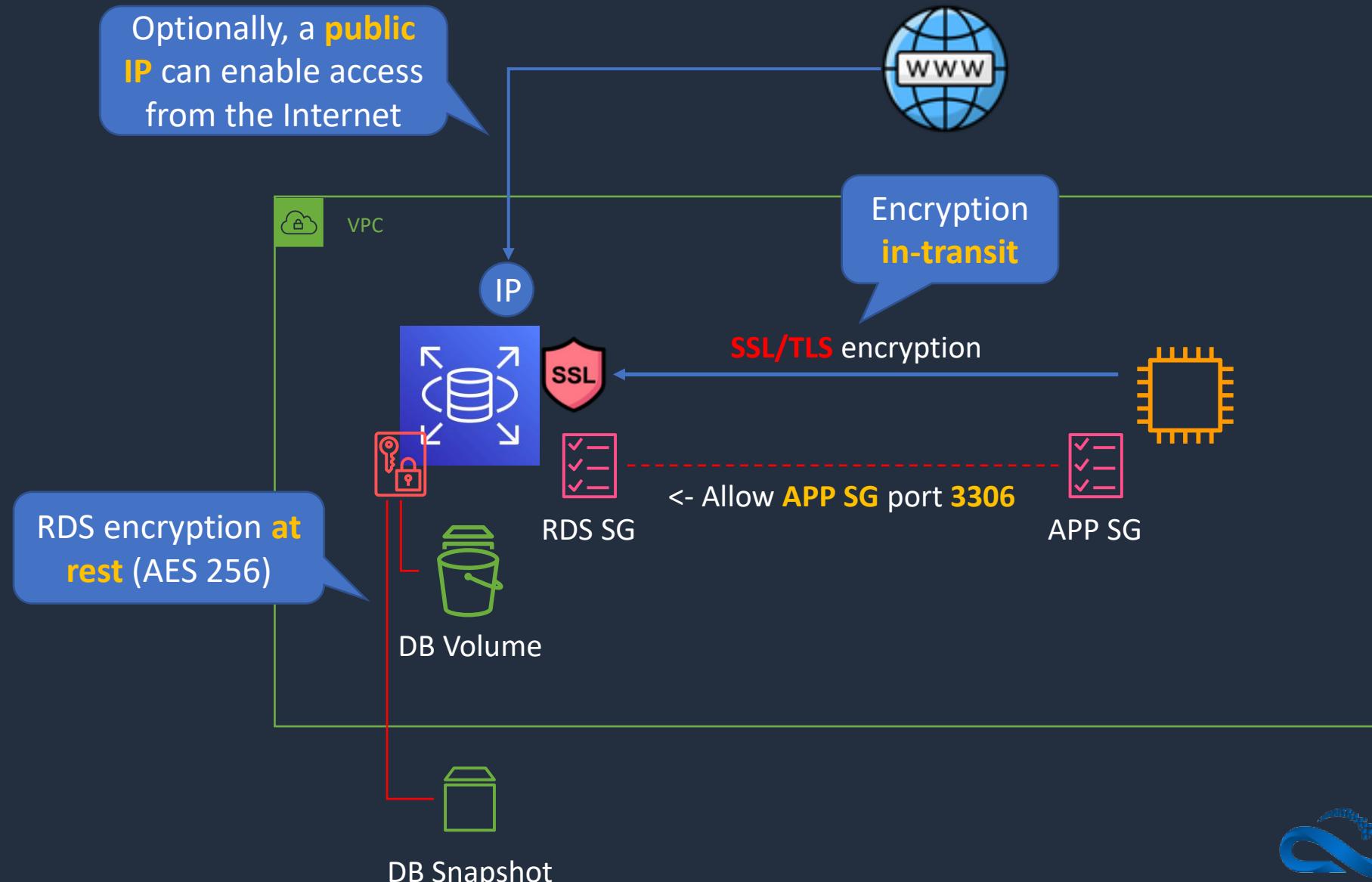


Amazon RDS Security





Amazon RDS Security





Amazon RDS Security

- Encryption **at rest** can be enabled – includes DB storage, backups, read replicas and snapshots
- You can only enable encryption for an Amazon RDS DB instance when you create it, not after the DB instance is created
- DB instances that are encrypted can't be modified to disable encryption
- Uses AES 256 encryption and encryption is transparent with minimal performance impact
- RDS for Oracle and SQL Server is also supported using Transparent Data Encryption (TDE) (may have performance impact)
- AWS KMS is used for managing encryption keys

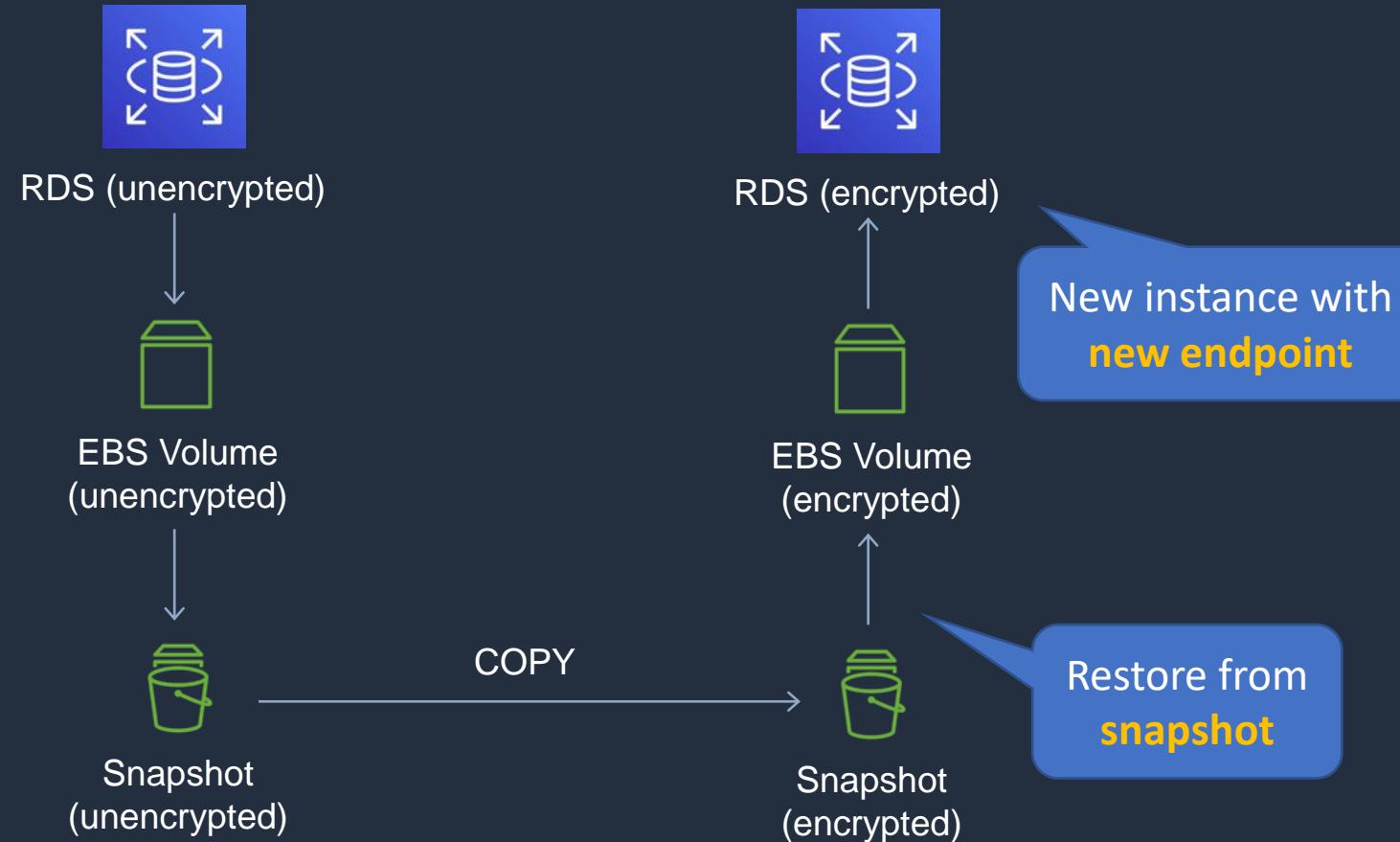


Amazon RDS Security

- You can't have:
 - An **encrypted** read replica of an **unencrypted** DB instance
 - An **unencrypted** read replica of an **encrypted** DB instance
- Read replicas of encrypted primary instances are encrypted
- The same KMS key is used if in the same Region as the primary
- If the read replica is in a different Region, a different KMS key is used
- You can't restore an unencrypted backup or snapshot to an encrypted DB instance



Amazon RDS Security

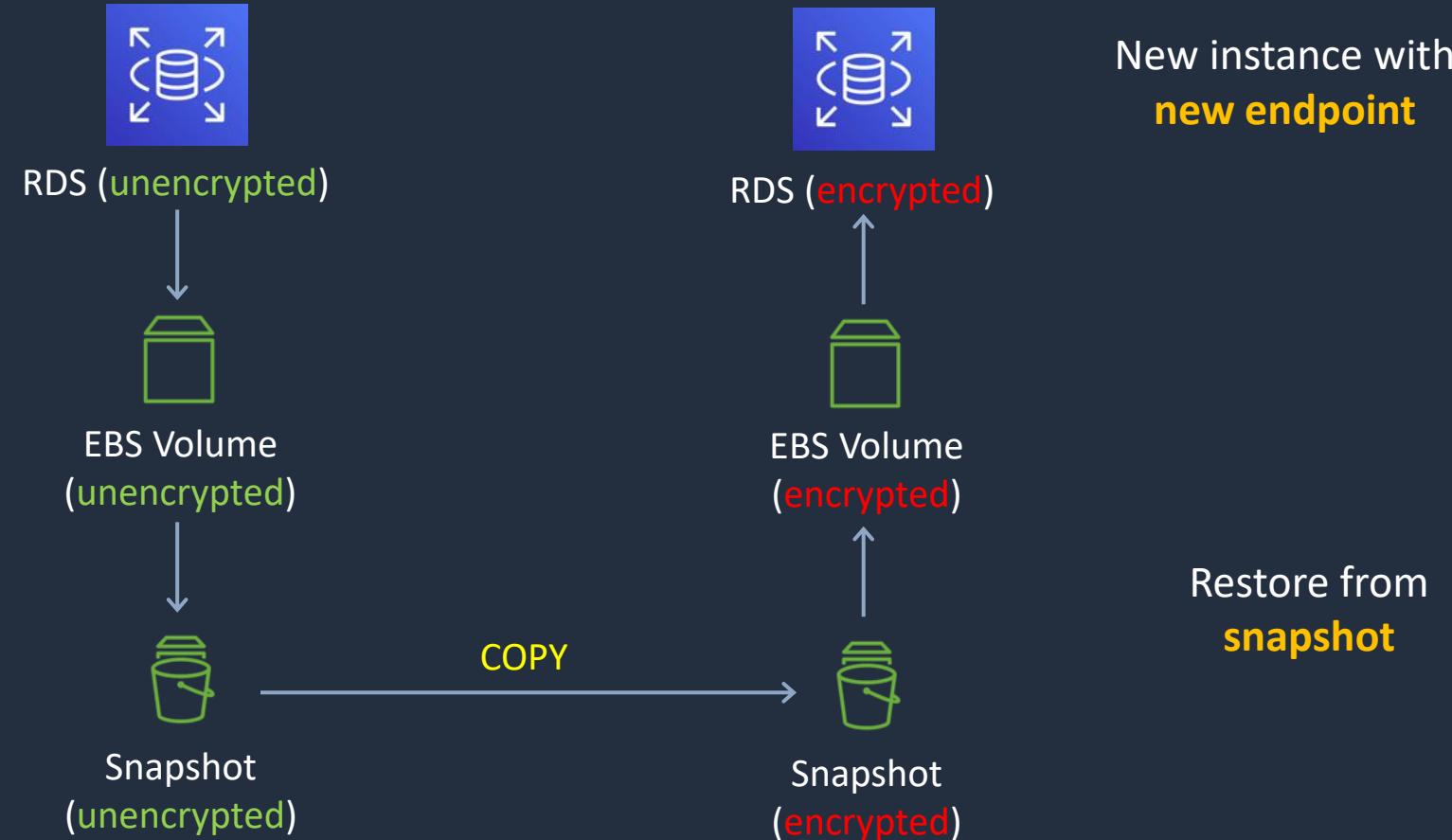


Create Encrypted Copy of RDS Database





Amazon RDS Security



Amazon Aurora





Amazon Aurora

- Amazon Aurora is an AWS database offering in the RDS family
- Amazon Aurora is a MySQL and PostgreSQL-compatible relational database built for the cloud
- Amazon Aurora is up to five times faster than standard MySQL databases and three times faster than standard PostgreSQL databases
- Amazon Aurora features a distributed, fault-tolerant, self-healing storage system that auto-scales up to 128TB per database instance

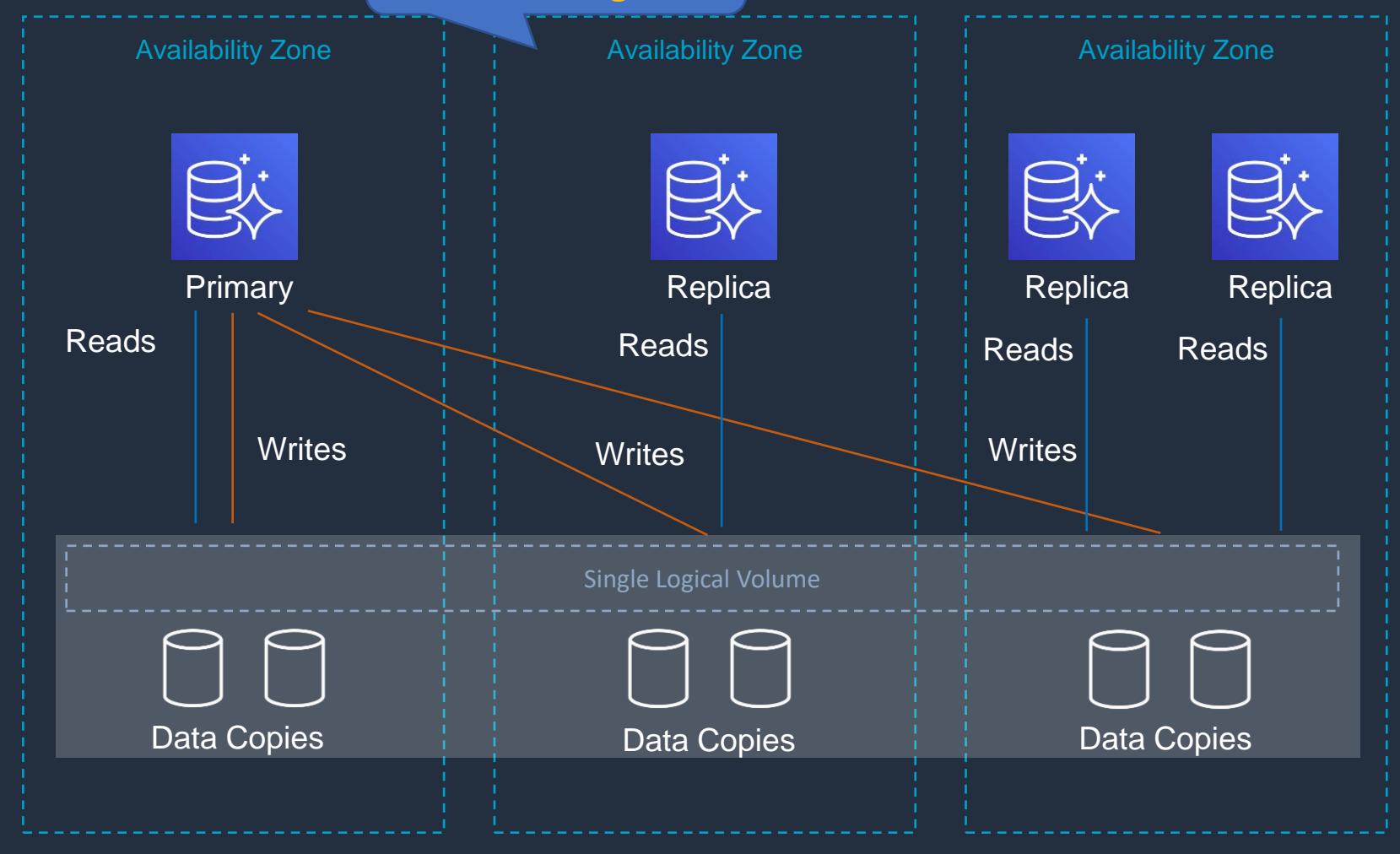


Amazon Aurora



Region

Aurora Replicas are
within a region



Aurora Fault Tolerance

- Fault tolerance across 3 AZs
- Single logical volume
- Aurora Replicas scale-out read requests
- Can **promote** Aurora Replica to be a new primary or create new primary
- Can use **Auto Scaling** to add replicas



Amazon Aurora Key Features

Aurora Feature	Benefit
High performance and scalability	Offers high performance, self-healing storage that scales up to 128TB, point-in-time recovery and continuous backup to S3
DB compatibility	Compatible with existing MySQL and PostgreSQL open source databases
Aurora Replicas	In-region read scaling and failover target – up to 15 (can use Auto Scaling)
MySQL Read Replicas	Cross-region cluster with read scaling and failover target – up to 5 (each can have up to 15 Aurora Replicas)
Global Database	Cross-region cluster with read scaling (fast replication / low latency reads). Can remove secondary and promote
Multi-Master	Scales out writes within a region. In preview currently and will not appear on the exam
Serverless	On-demand, autoscaling configuration for Amazon Aurora - does not support read replicas or public IPs (can only access through VPC or Direct Connect - not VPN)



Amazon Aurora Replicas

Feature	Aurora Replica	MySQL Replica
Number of replicas	Up to 15	Up to 5
Replication type	Asynchronous (milliseconds)	Asynchronous (seconds)
Performance impact on primary	Low	High
Replica location	In-region	Cross-region
Act as failover target	Yes (no data loss)	Yes (potentially minutes of data loss)
Automated failover	Yes	No
Support for user-defined replication delay	No	Yes
Support for different data or schema vs. primary	No	Yes

Amazon Aurora Deployment Options



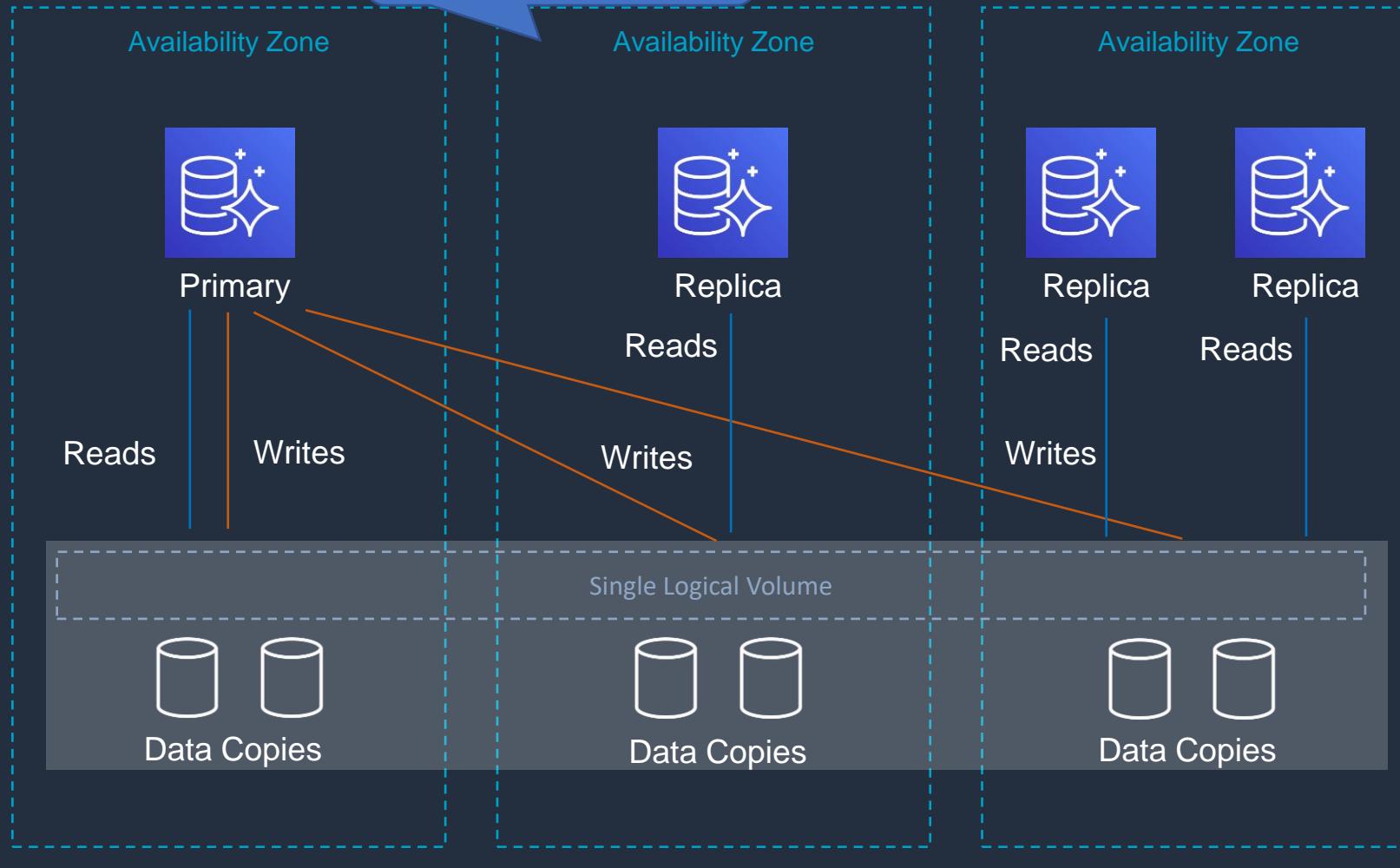


Aurora Fault Tolerance and Aurora Replicas



Region

Aurora Replicas are
within a region

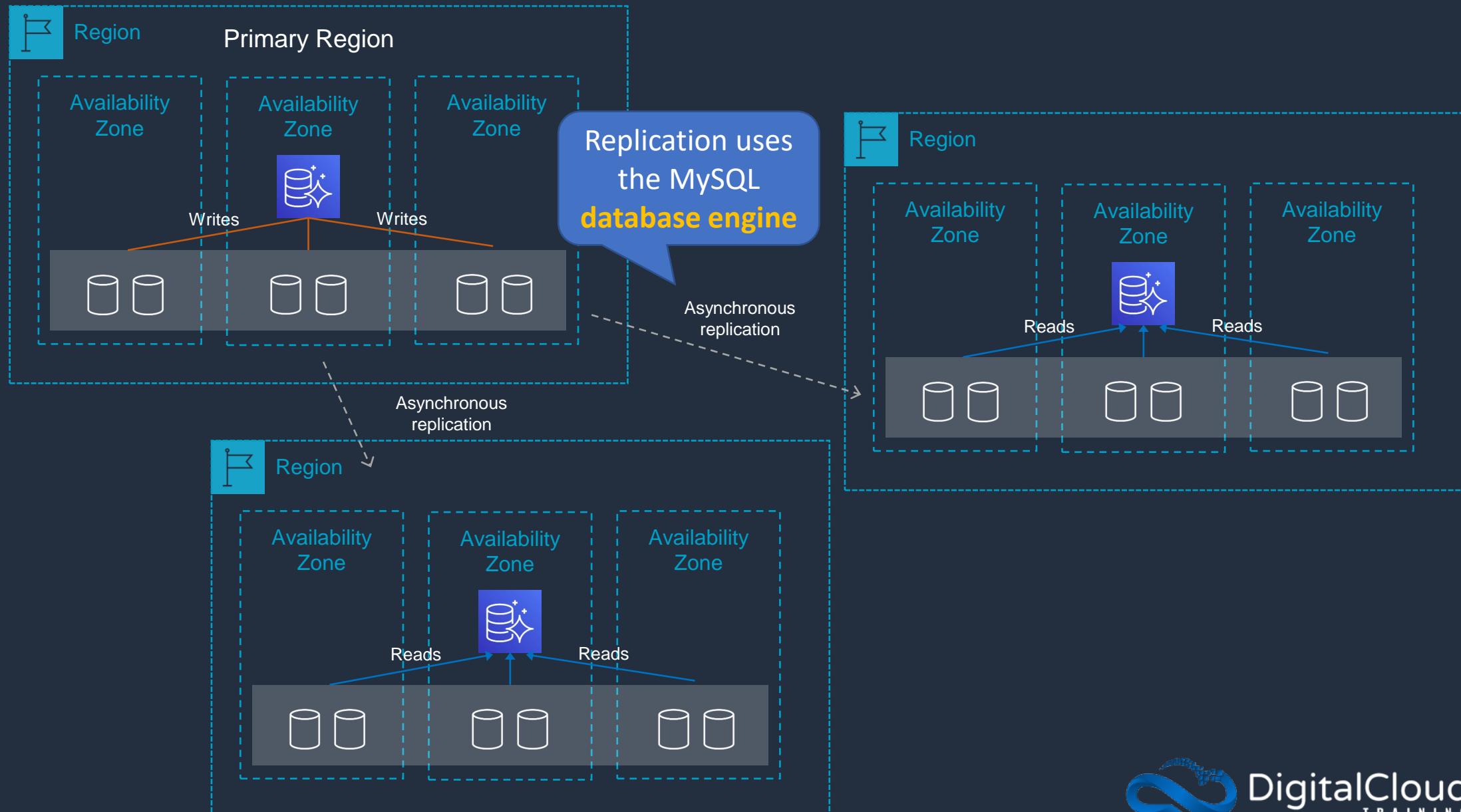


Aurora Fault Tolerance

- Fault tolerance across 3 AZs
- Single logical volume
- Aurora Replicas scale-out read requests
- Up to 15 Aurora Replicas with **sub-10ms** replica lag
- Aurora Replicas are independent endpoints
- Can **promote** Aurora Replica to be a new primary or create new primary
- Set priority (tiers) on Aurora Replicas to control order of promotion
- Can use **Auto Scaling** to add replicas

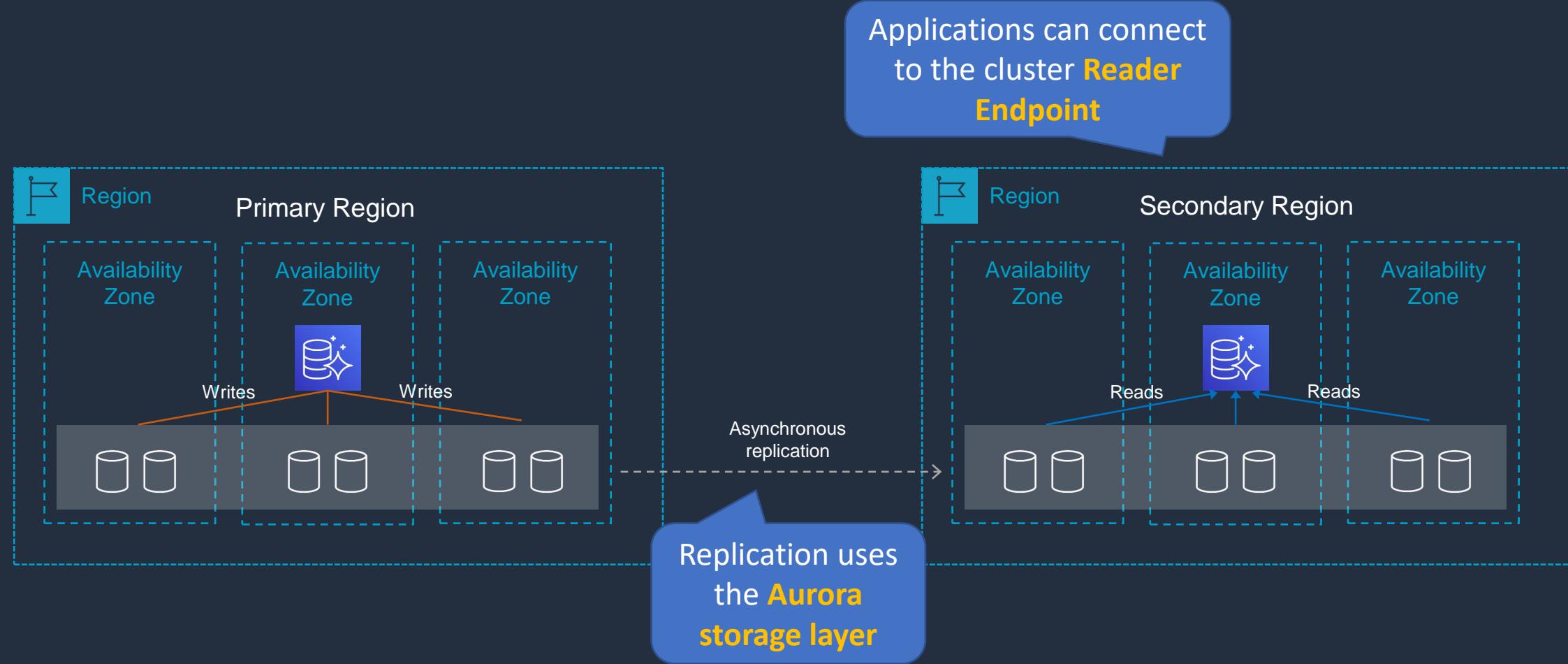


Cross-Region Replica with Aurora MySQL





Aurora Global Database

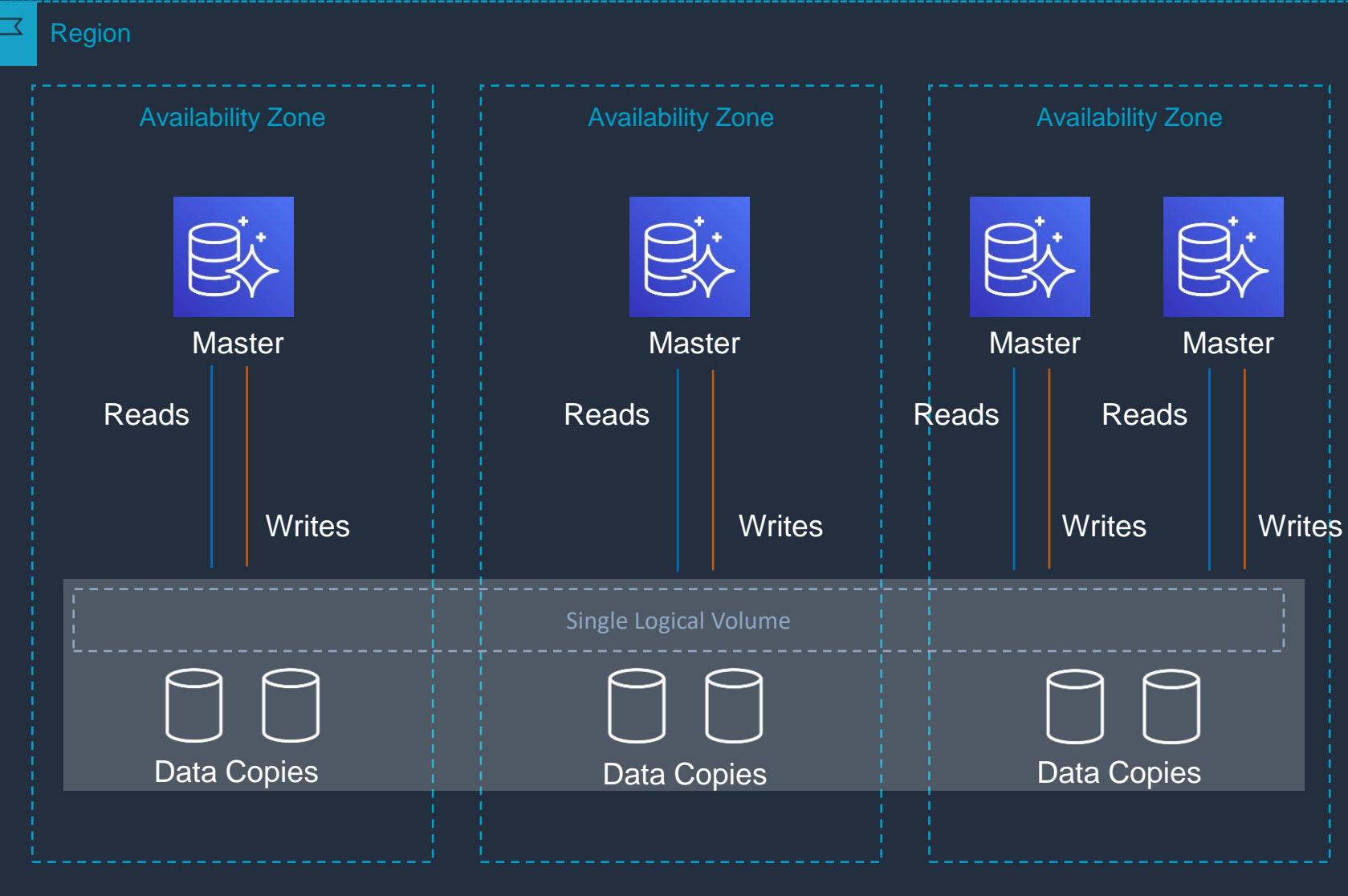




Aurora Multi-Master



Region

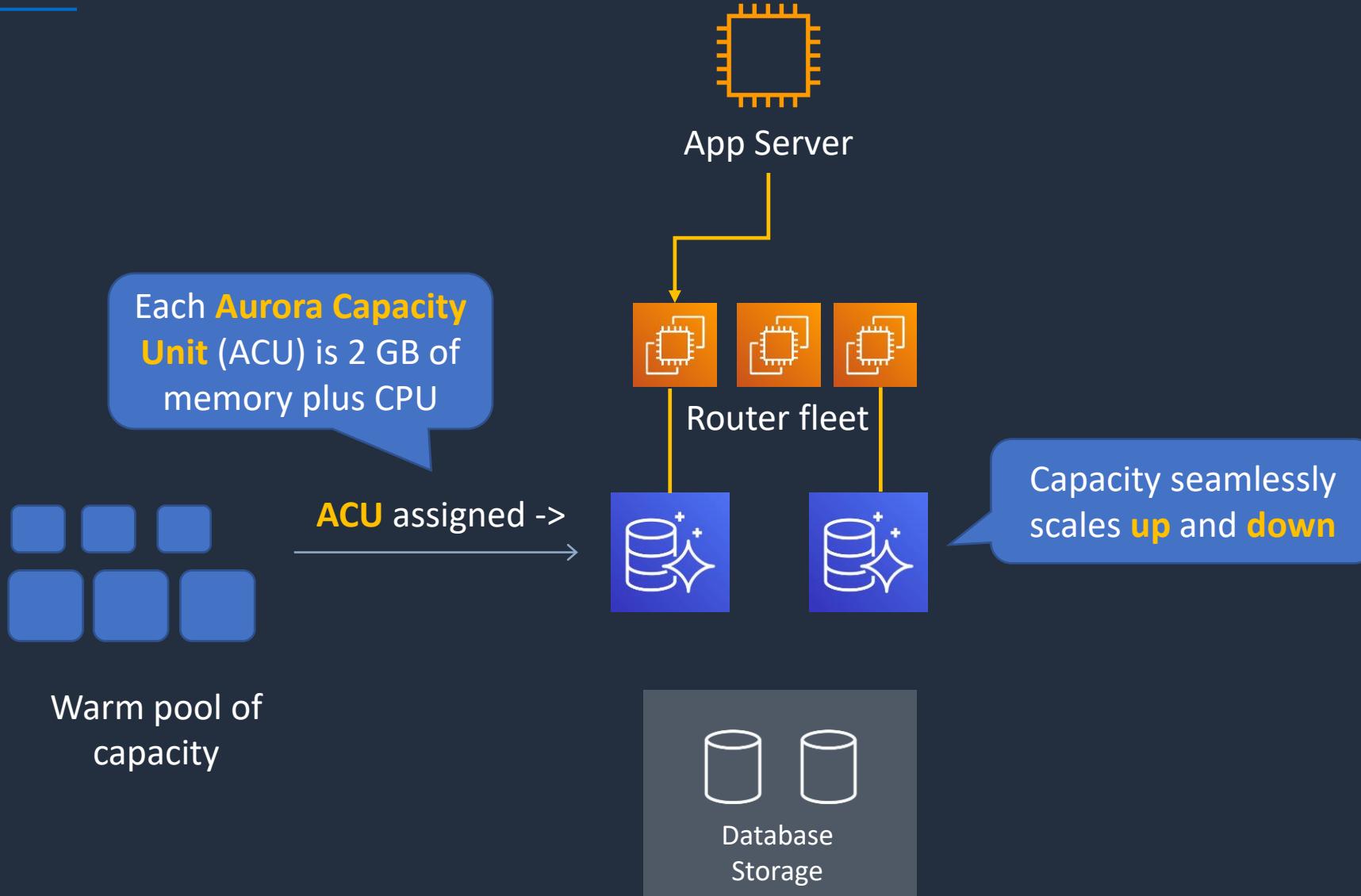


Aurora Multi-Master

- All nodes allow reads/writes
- Available for MySQL only
- Up to four read/write nodes
- Single Region only
- Cannot have cross-Region replicas
- Can work with active-active and active-passive workloads
- Can restart read/write DB instance without impacting other instances



Aurora Serverless

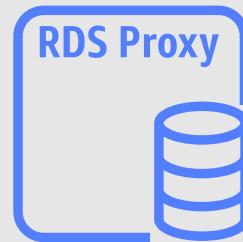




Aurora Serverless Use Cases

- Infrequently used applications
- New applications
- Variable workloads
- Unpredictable workloads
- Development and test databases
- Multi-tenant applications

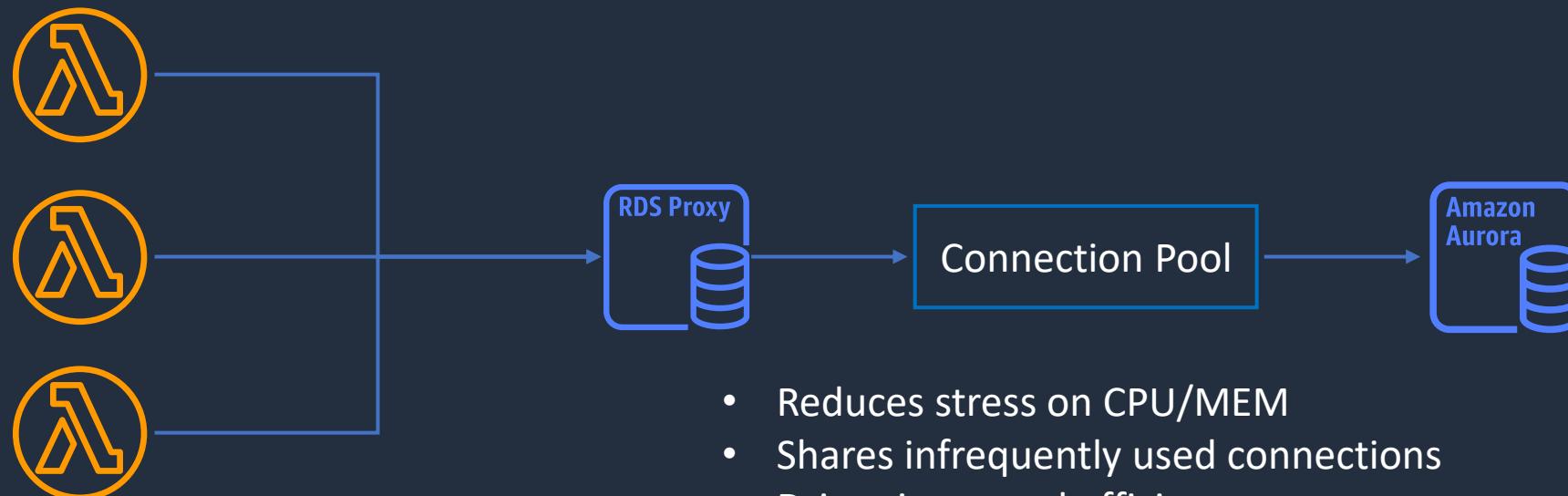
Amazon RDS Proxy





Amazon RDS Proxy

- RDS Proxy is a fully managed database proxy for RDS
- Highly available across multiple AZs
- Increases scalability, fault tolerance, and security



- Reduces stress on CPU/MEM
- Shares infrequently used connections
- Drives increased efficiency
- High availability with failover
- Control authentication methods

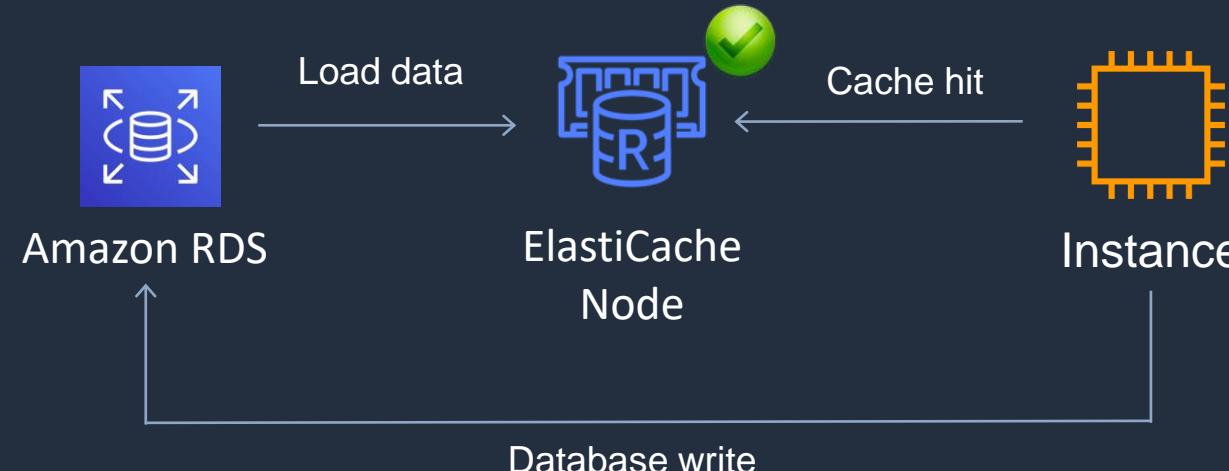
Amazon ElastiCache





Amazon ElastiCache

- Fully managed implementations **Redis** and **Memcached**
- ElastiCache is a **key/value** store
- In-memory database offering high performance and low latency
- Can be put in front of databases such as RDS and DynamoDB
- ElastiCache nodes run on Amazon EC2 instances, so you must choose an instance family/type





Amazon ElastiCache

Feature	Memcached	Redis (cluster mode disabled)	Redis (cluster mode enabled)
Data persistence	No	Yes	Yes
Data types	Simple	Complex	Complex
Data partitioning	Yes	No	Yes
Encryption	No	Yes	Yes
High availability (replication)	No	Yes	Yes
Multi-AZ	Yes, place nodes in multiple AZs. No failover or replication	Yes, with auto-failover. Uses read replicas (0-5 per shard)	Yes, with auto-failover. Uses read replicas (0-5 per shard)
Scaling	Up (node type); out (add nodes)	Up (node type); out (add replica)	Up (node type); out (add shards)
Multithreaded	Yes	No	No
Backup and restore	No (and no snapshots)	Yes, automatic and manual snapshots	Yes, automatic and manual snapshots



Amazon ElastiCache Use Cases

- Data that is relatively **static** and **frequently accessed**
- Applications that are tolerant of stale data
- Data is slow and expensive to get compared to cache retrieval
- Require push-button scalability for memory, writes and reads
- Often used for storing session state



Amazon ElastiCache Examples

Use Case	Benefit
Web session store	In cases with load-balanced web servers, store web session information in Redis so if a server is lost, the session info is not lost, and another web server can pick it up
Database caching	Use Memcached in front of AWS RDS to cache popular queries to offload work from RDS and return results faster to users
Leaderboards	Use Redis to provide a live leaderboard for millions of users of your mobile app
Streaming data dashboards	Provide a landing spot for streaming sensor data on the factory floor, providing live real-time dashboard displays

Scaling ElastiCache





Amazon ElastiCache - Scalability

Memcached

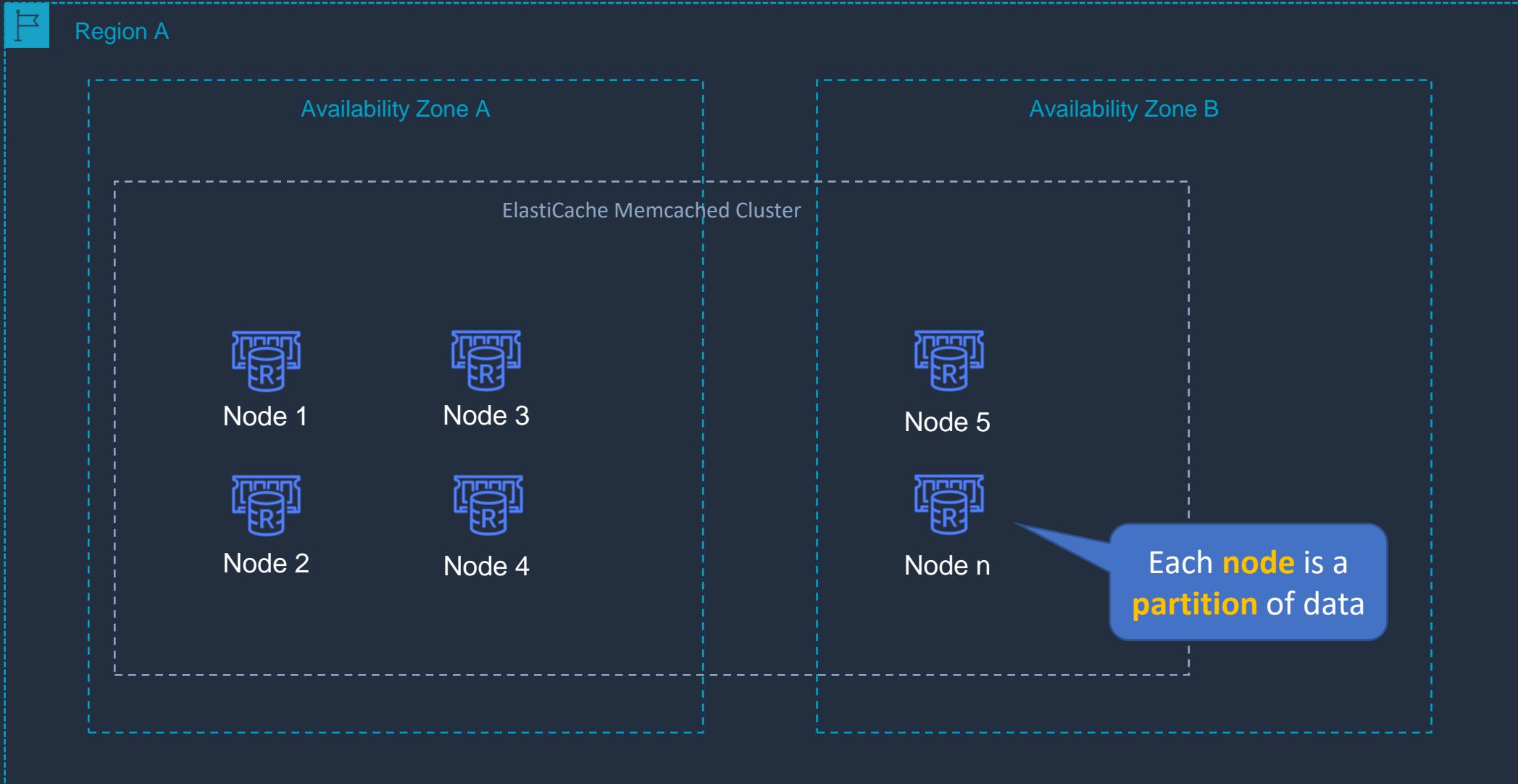
- Add nodes to a cluster
- Scale vertically (node type) – must create a **new cluster** manually

Redis

- Cluster mode **disabled**:
 - Add replica or change node type – creates a new cluster and migrates data
- Cluster mode **enabled**:
 - Online resharding to add or remove shards; vertical scaling to change node type
 - Offline resharding to add or remove shards change node type or upgrade engine (more flexible than online)

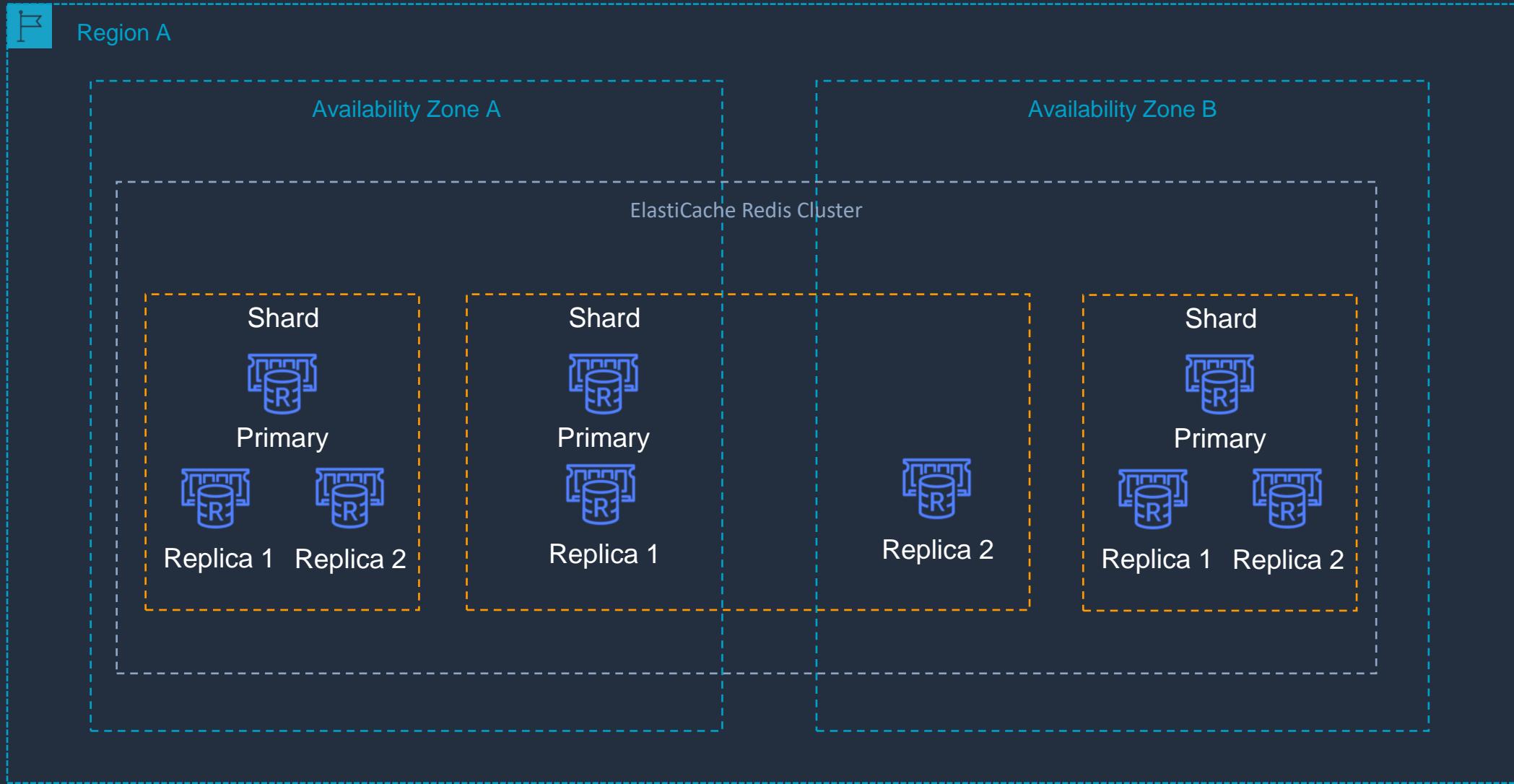


Amazon ElastiCache Memcached



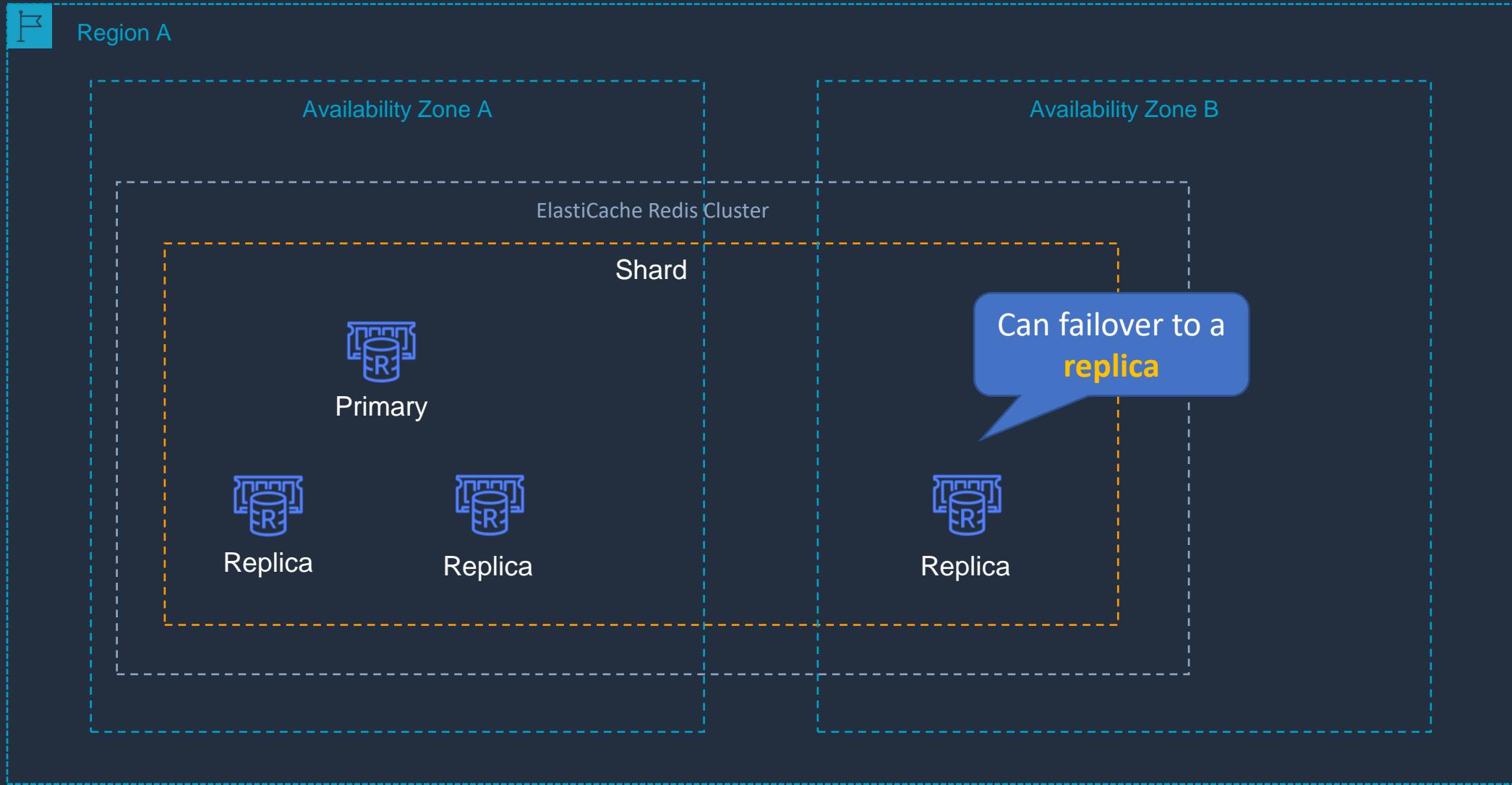


Amazon ElastiCache Redis (Cluster mode enabled)





Amazon ElastiCache Redis (Cluster mode disabled)



Create ElastiCache Cluster



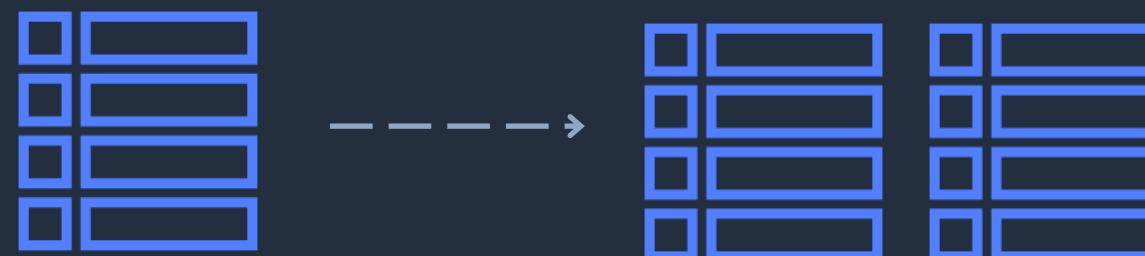
Amazon DynamoDB





Amazon DynamoDB

- Fully managed NoSQL database service
- Key/value store and document store
- It is a non-relational, key-value type of database
- Fully serverless service
- Push button scaling



DynamoDB Table



Amazon DynamoDB

- DynamoDB is made up of:

- Tables
- Items
- Attributes

userid	orderid	book	price	date
user001	1000092	ISBN100..	9.99	2020.04..
user002	1000102	ISBN100..	24.99	2020.03..
user003	1000168	ISBN2X0..	12.50	2020.04..



DynamoDB Time to Live (TTL)

- TTL lets you define when items in a table expire so that they can be automatically deleted from the database
- With TTL enabled on a table, you can set a timestamp for deletion on a per-item basis
- No extra cost and does not use WCU / RCU
- Helps reduce storage and manage the table size over time



Amazon DynamoDB

Primary Key		Attributes				
Partition Key	Sort Key	sku	category	size	colour	weight
clientid	created	SKU-S523	T-Shirt	Small	Red	Light
john@example.com	1583975308	SKU-J091	Pen		Blue	
chris@example.com	1583975613	SKU-A234	Mug			
chris@example.com	1583975449	SKU-R873	Chair		4011	
sarah@example.com	1583976311	SKU-I019	Plate	30		
jenny@example.com	1583976323					

This is known as a
composite key as it has a
partition key + sort key

Useful when the data
structure is **unpredictable**



Amazon DynamoDB

DynamoDB Feature	Benefit
Serverless	Fully managed, fault tolerant, service
Highly available	99.99% availability SLA – 99.999% for Global Tables!
NoSQL type of database with Name / Value structure	Flexible schema, good for when data is not well structured or unpredictable
Horizontal scaling	Seamless scalability to any scale with push button scaling or Auto Scaling
DynamoDB Streams	Captures a time-ordered sequence of item-level modifications in a DynamoDB table and durably stores the information for up to 24 hours. Often used with Lambda and the Kinesis Client Library (KCL)
DynamoDB Accelerator (DAX)	Fully managed in-memory cache for DynamoDB that increases performance (microsecond latency)
Transaction options	Strongly consistent or eventually consistent reads, support for ACID transactions
Backup	Point-in-time recovery down to the second in last 35 days; On-demand backup and restore
Global Tables	Fully managed multi-region, multi-master solution

Practice Creating DynamoDB Tables



Practice Creating DynamoDB Tables

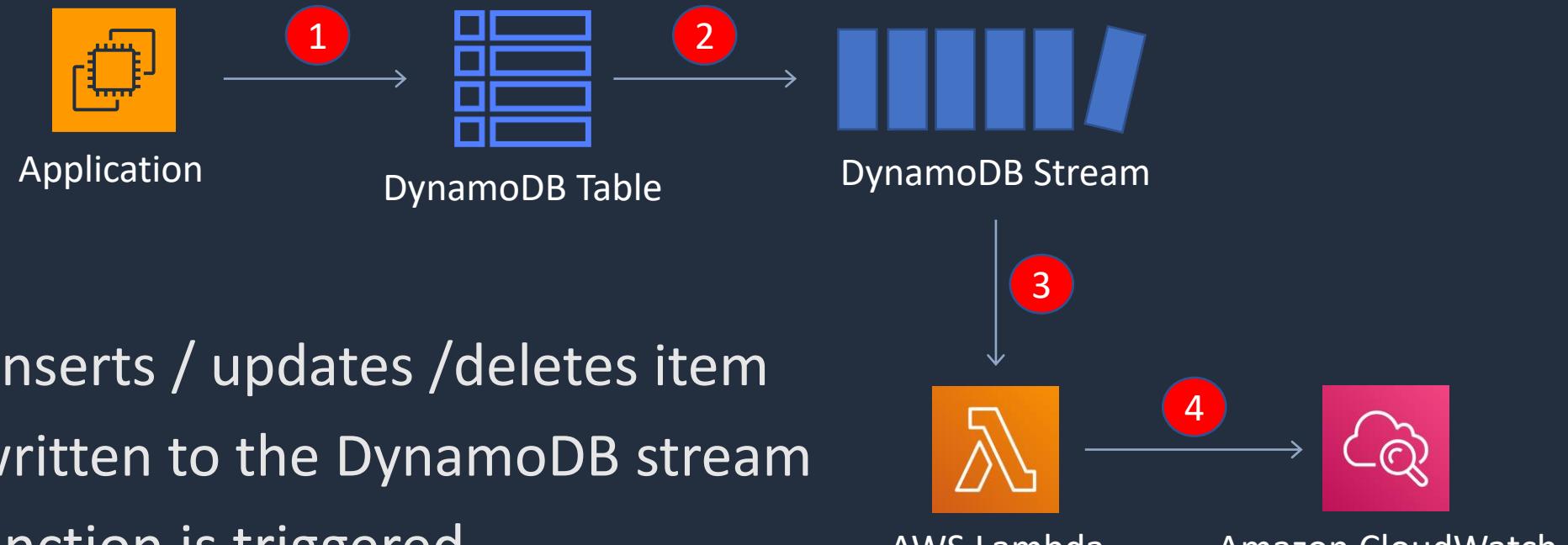


DynamoDB Streams





DynamoDB Streams



1. Application inserts / updates /deletes item
2. A record is written to the DynamoDB stream
3. A Lambda function is triggered
4. The Lambda function writes to CloudWatch Logs



DynamoDB Streams

- Captures a **time-ordered** sequence of **item-level** modifications in any DynamoDB table and stores this information in a log for up to **24 hours**
- Can configure the information that is written to the stream:
 - **KEYS_ONLY** — Only the key attributes of the modified item
 - **NEW_IMAGE** — The entire item, as it appears after it was modified
 - **OLD_IMAGE** — The entire item, as it appeared before it was modified
 - **NEW_AND_OLD_IMAGES** — Both the new and the old images of the item

DynamoDB Accelerator (DAX)

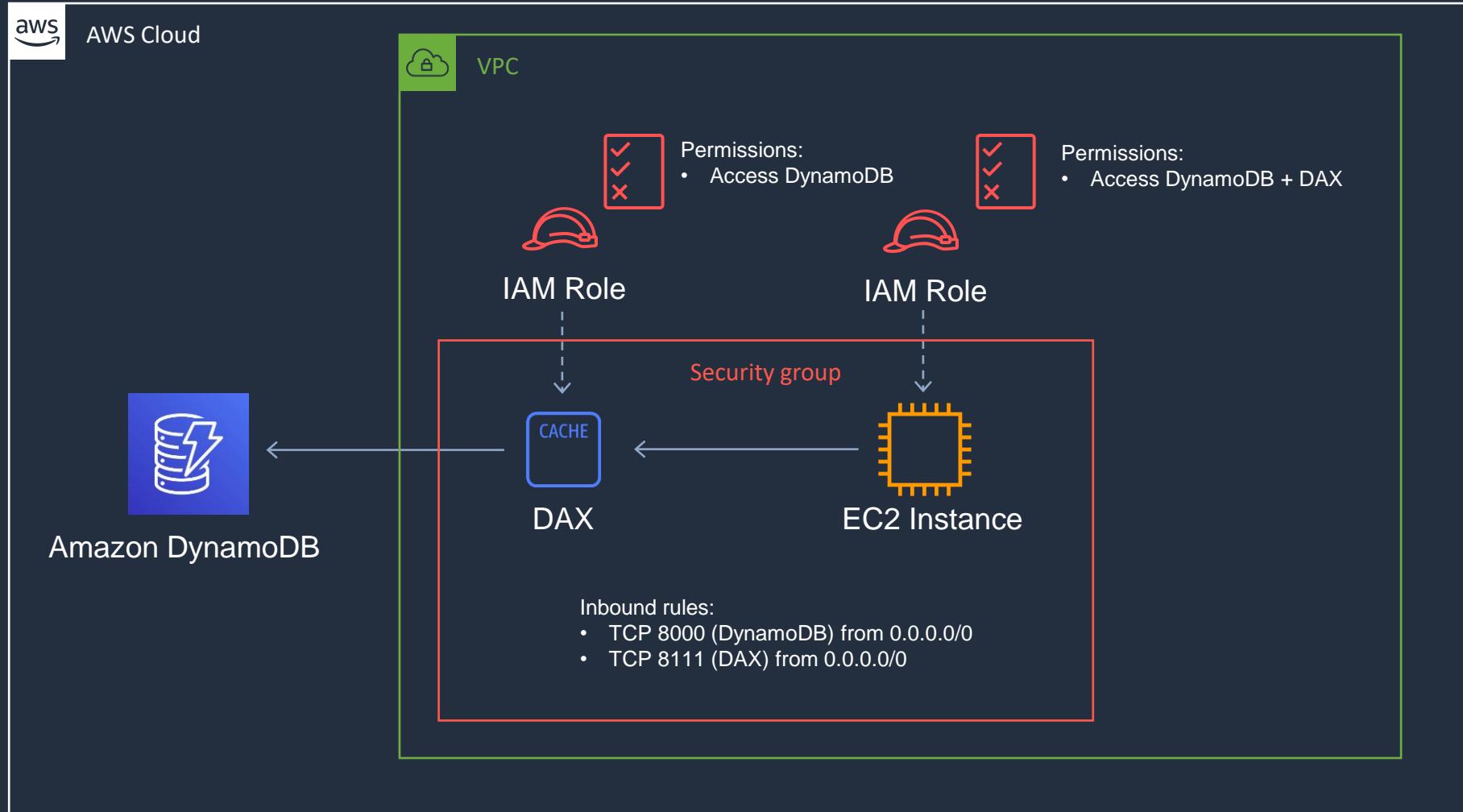




DynamoDB Accelerator (DAX)

- DAX is a fully managed, highly available, **in-memory** cache for DynamoDB
- Improves performance from milliseconds to microseconds
- Can be a read-through cache and a write-through cache
- Used to improve **READ** and **WRITE** performance
- You do not need to modify application logic, since DAX is compatible with existing DynamoDB API calls

DynamoDB Accelerator (DAX)





DAX vs ElastiCache

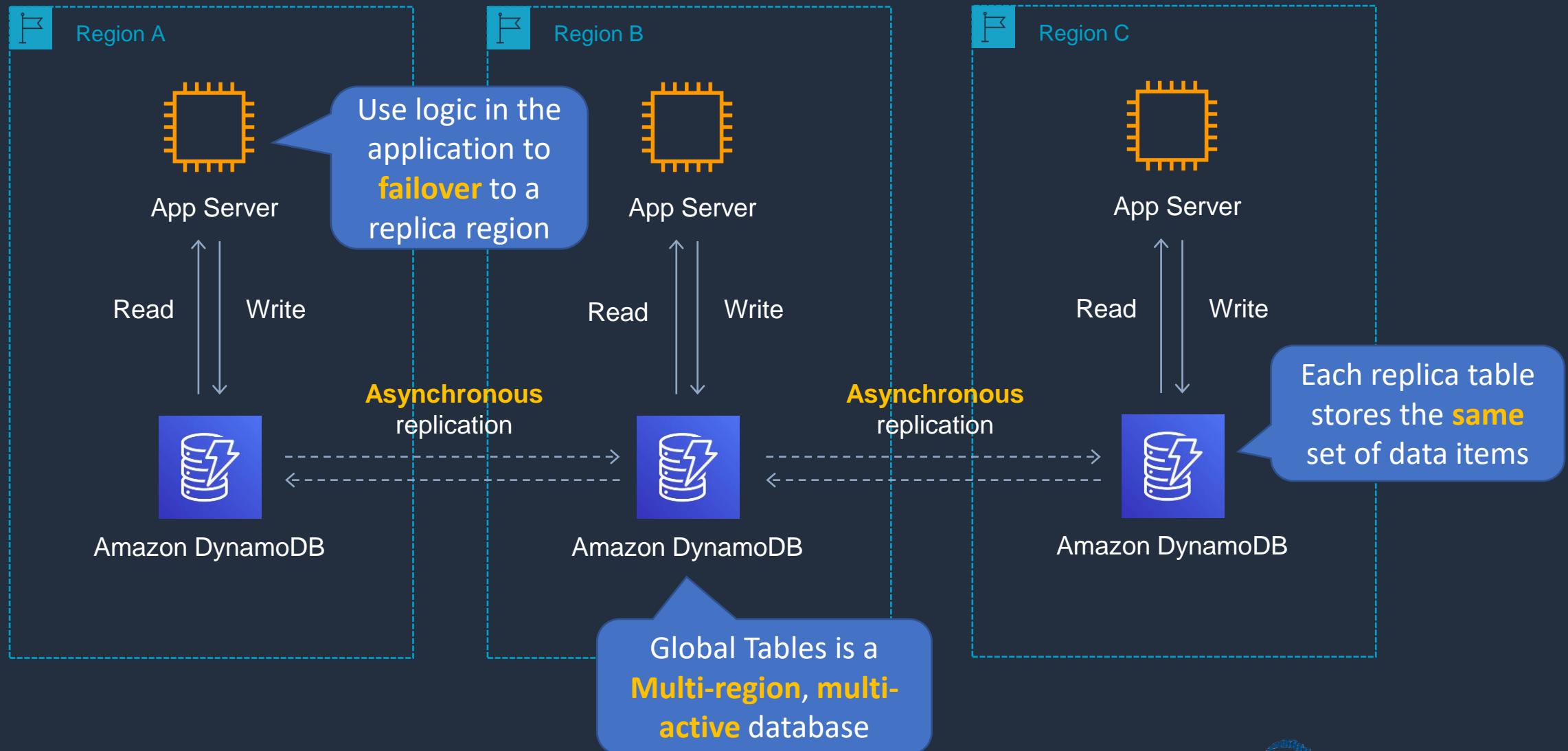
- DAX is **optimized** for DynamoDB
- With ElastiCache you have more **management overhead** (e.g. invalidation)
- With ElastiCache you need to **modify application code** to point to cache
- ElastiCache supports more datastores

DynamoDB Global Tables





DynamoDB Global Tables



Enable Global Table



Amazon RedShift





Amazon RedShift

- Amazon Redshift is a fast, fully managed data warehouse
- Analyze data using standard SQL and existing Business Intelligence (BI) tools
- RedShift is a SQL based data warehouse used for analytics applications
- RedShift is a relational database that is used for Online Analytics Processing (OLAP) use cases
- RedShift uses Amazon EC2 instances, so you must choose an instance family/type
- RedShift always keeps three copies of your data
- RedShift provides continuous/incremental backups



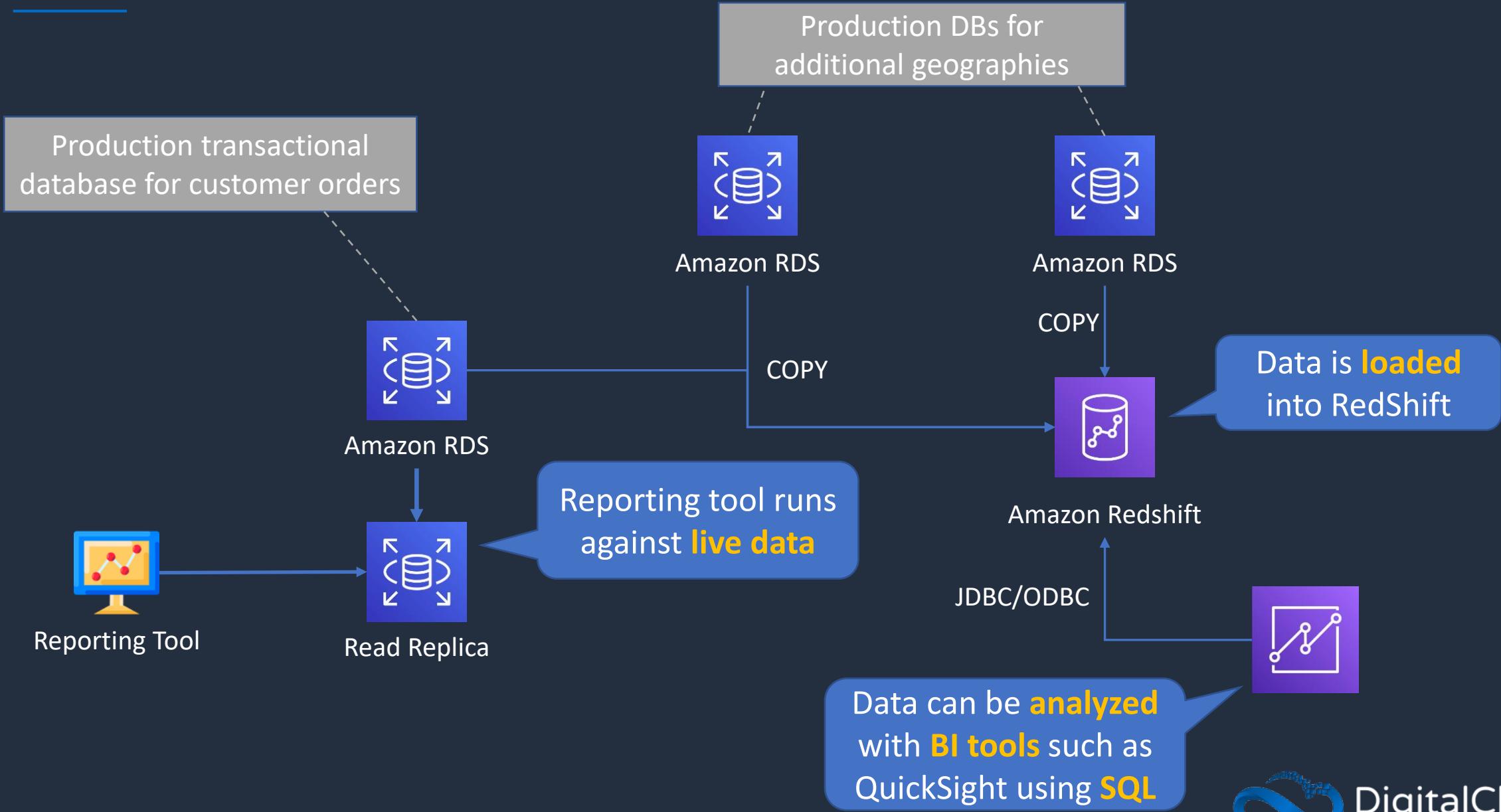
OLTP vs OLAP (refresher)

Key differences are *use cases* and how the database is *optimized*

Operational / transactional	Analytical
Online Transaction Processing (OLTP)	Online Analytics Processing (OLAP) – the source data comes from OLTP DBs
Production DBs that process transactions. E.g. adding customer records, checking stock availability (INSERT, UPDATE, DELETE)	Data warehouse. Typically, separated from the customer facing DBs. Data is extracted for decision making
Short transactions and simple queries	Long transactions and complex queries
Examples: Amazon RDS, DynamoDB	Examples: Amazon RedShift, Amazon EMR

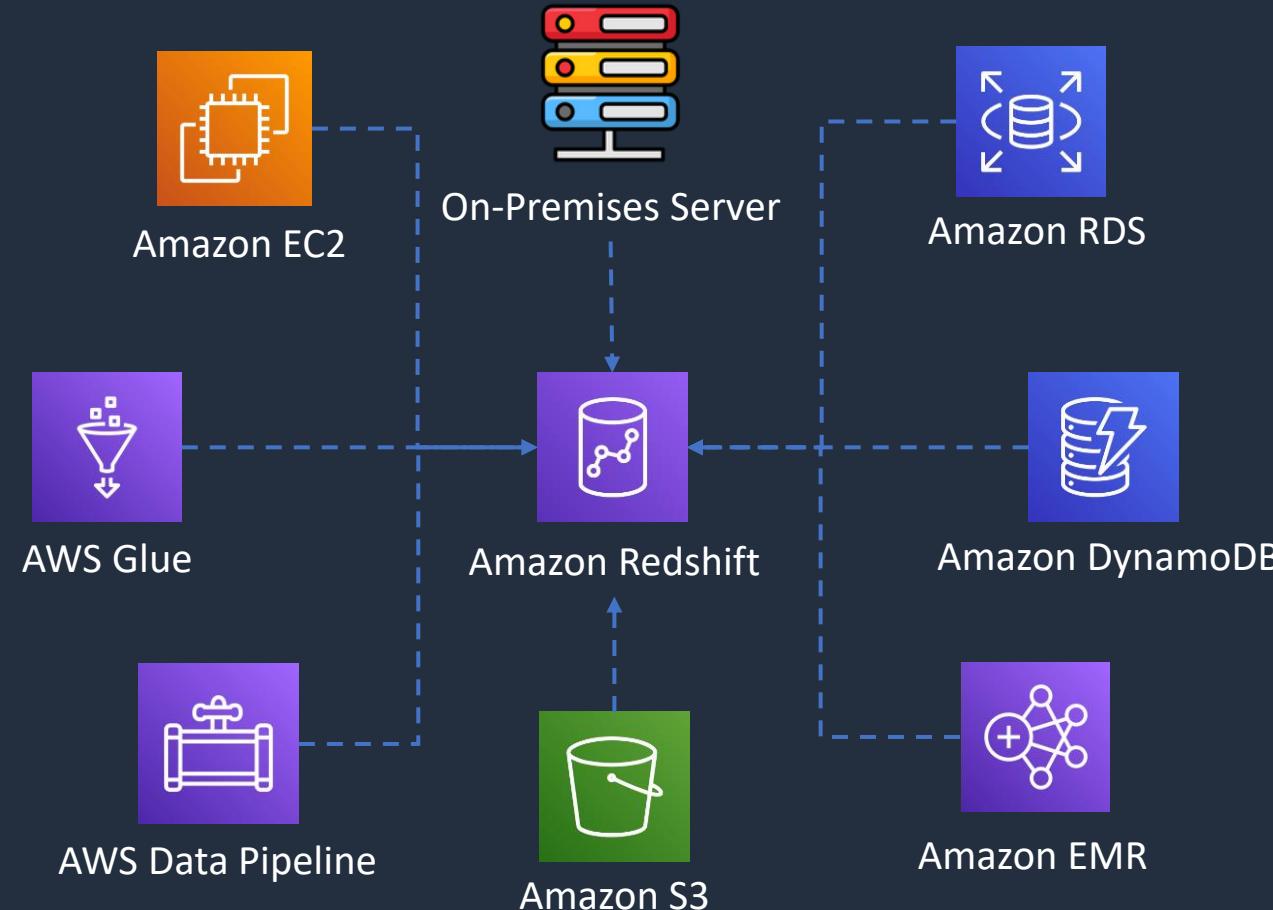


Reporting and Analytics Use Cases





Redshift Data Sources



RedShift Spectrum can
run SQL queries on data
directly in **S3**



RedShift Use Cases

- Perform **complex queries** on massive collections of **structured** and **semi-structured** data and get fast performance
- Frequently accessed data that needs a consistent, highly structured format
- Use **Spectrum** for direct access of **S3 objects** in a data lake
- Managed data warehouse solution with:
 - Automated provisioning, configuration and patching
 - Data durability with continuous backup to S3
 - Scales with simple API calls
 - Exabyte scale query capability

Amazon Elastic Map Reduce (EMR)



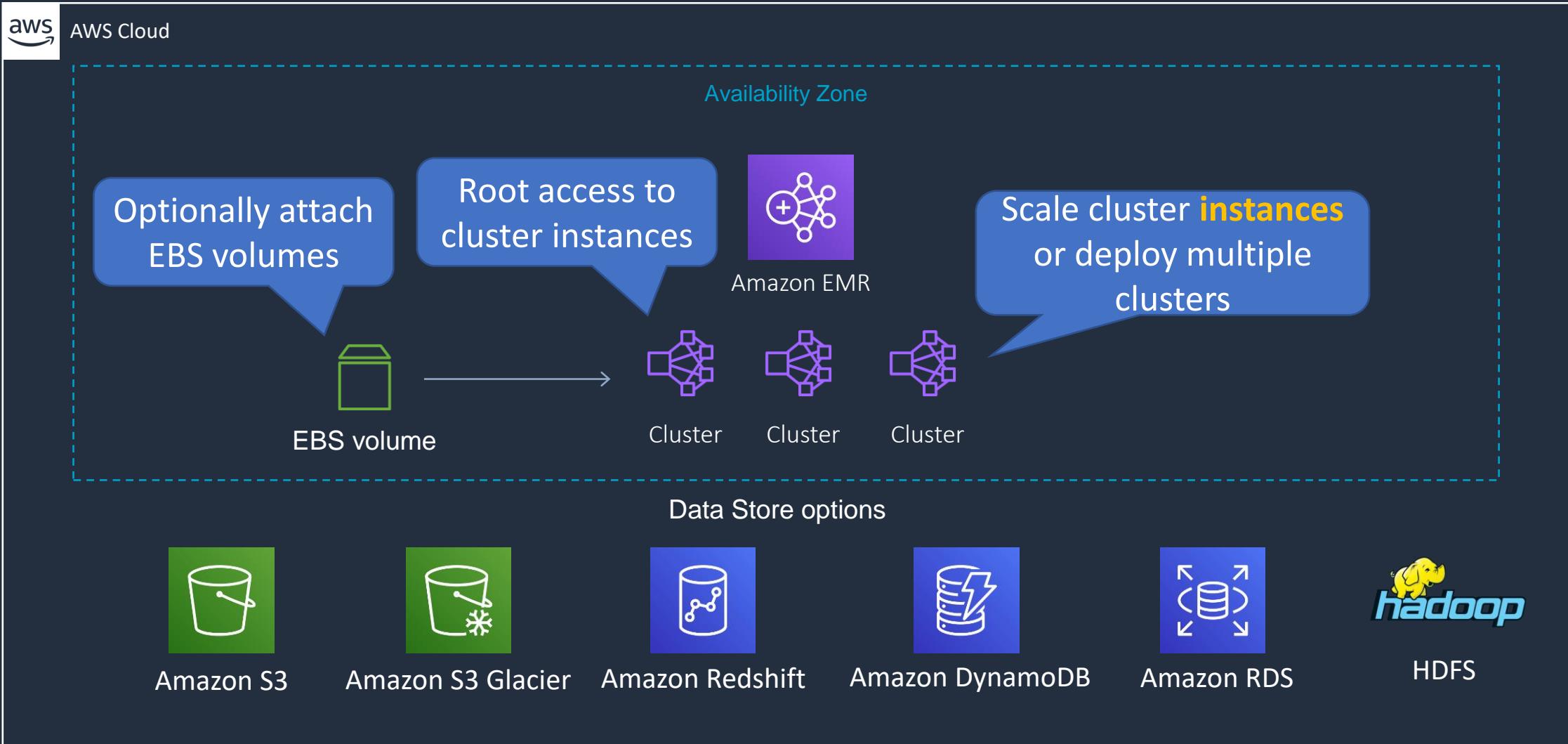


Amazon EMR

- Managed cluster platform that simplifies running big data frameworks including **Apache Hadoop** and **Apache Spark**
- Used for processing data for analytics and business intelligence
- Can also be used for transforming and moving large amounts of data
- Performs extract, transform, and load (ETL) functions



Amazon EMR



Amazon Kinesis





Amazon Kinesis Services

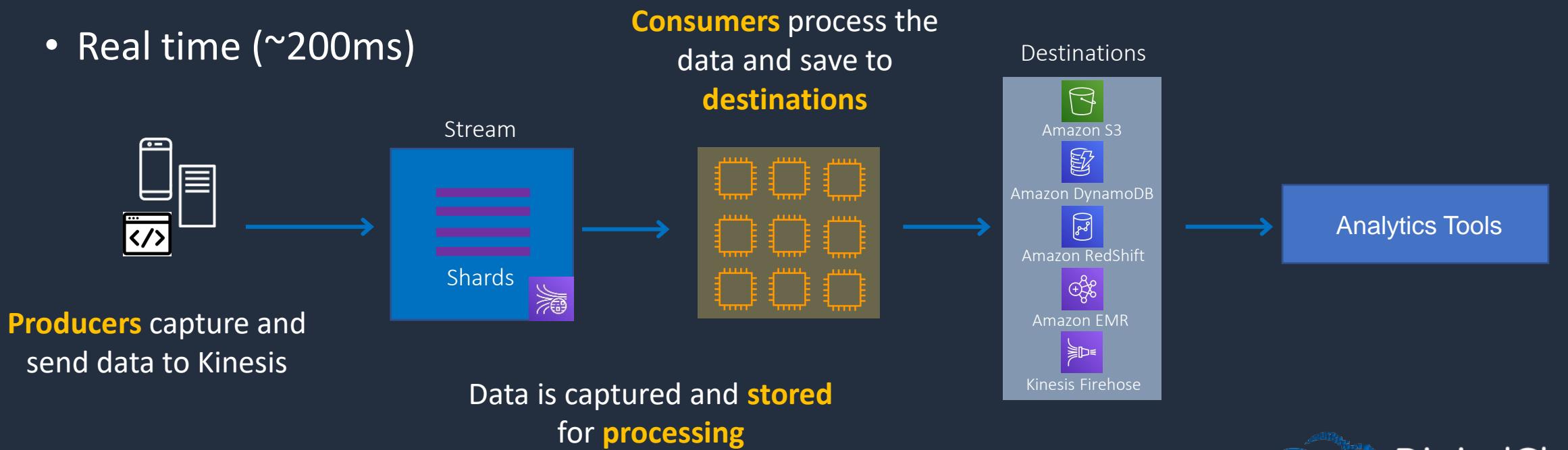
Data Analytics uses **Apache Flink** for **processing** data streams





Amazon Kinesis Data Streams

- Producers send data to Kinesis, data is stored in Shards for 24 hours (by default, up to 365 days)
- Consumers then take the data and process it - data can then be saved into another AWS service
- Real time (~200ms)

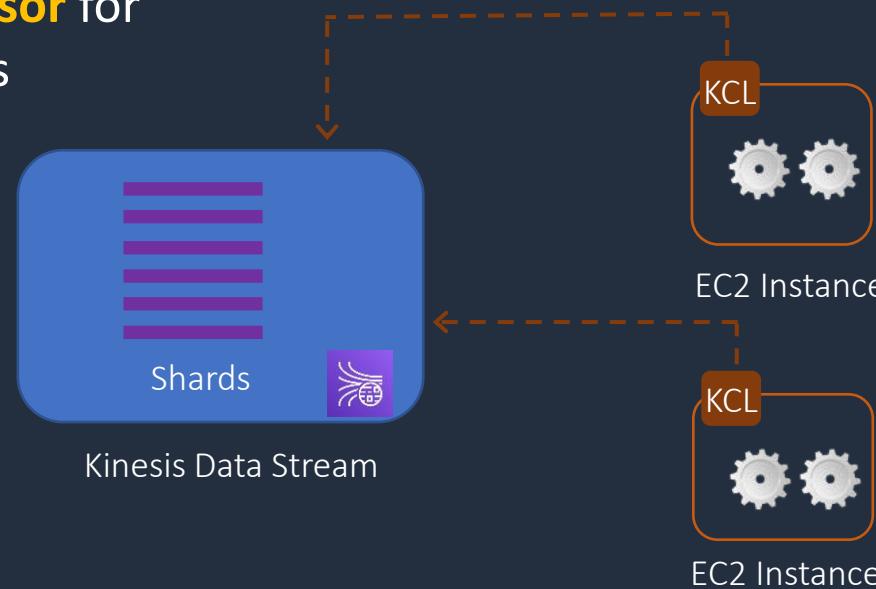




Kinesis Client Library (KCL)

- The Kinesis Client Library (KCL) helps you consume and process data from a Kinesis data stream

KCL **enumerates shards** and instantiates a **record processor** for each shard it manages



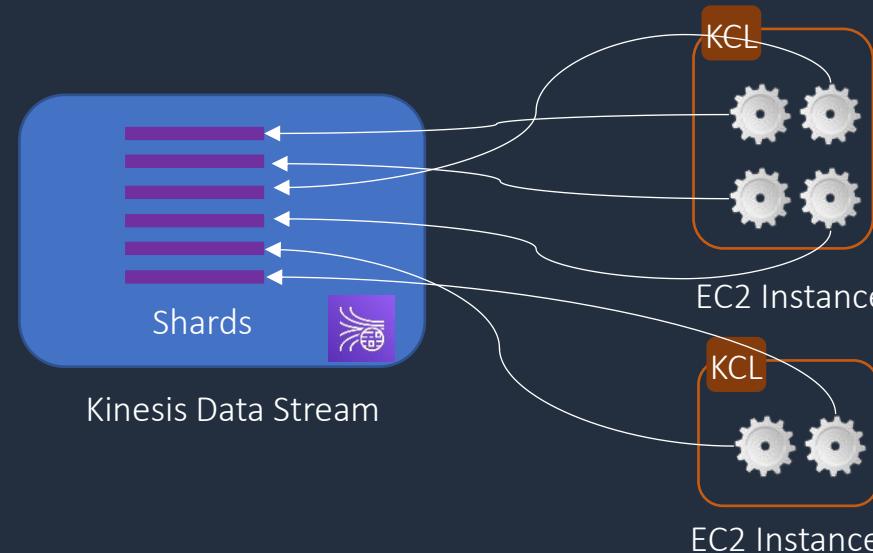
KCL Worker with multiple record processors



Kinesis Client Library (KCL)

- Each shard is processed by exactly one KCL worker and has exactly one corresponding record processor
- One worker can process any number of shards, so it's fine if the number of shards exceeds the number of instances

Each shard is **processed** by exactly **one** KCL worker



A **record processor** maps to exactly one **shard**



Amazon Kinesis Data Streams

Order is maintained for records **within a shard**



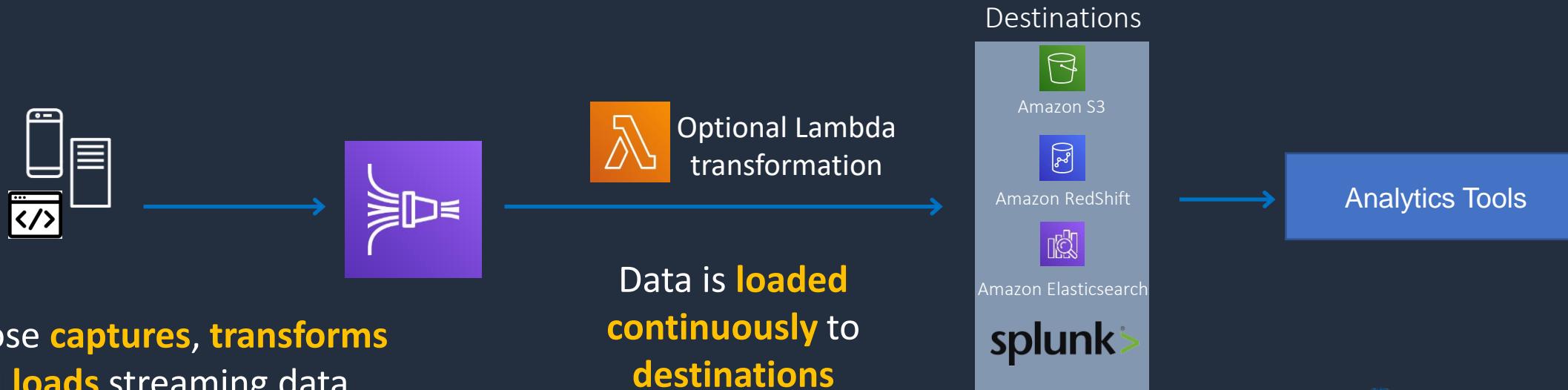
Producers
(EC2)

A partition key can be specified
with **PutRecord** to group data by
shard



Kinesis Data Firehose

- Producers send data to Firehose
- There are no Shards, completely automated (scalability is elastic)
- Firehose data is sent to another AWS service for storing, data can be optionally processed/transformed using AWS Lambda
- Near real-time delivery (~60 seconds latency)





Kinesis Data Firehose

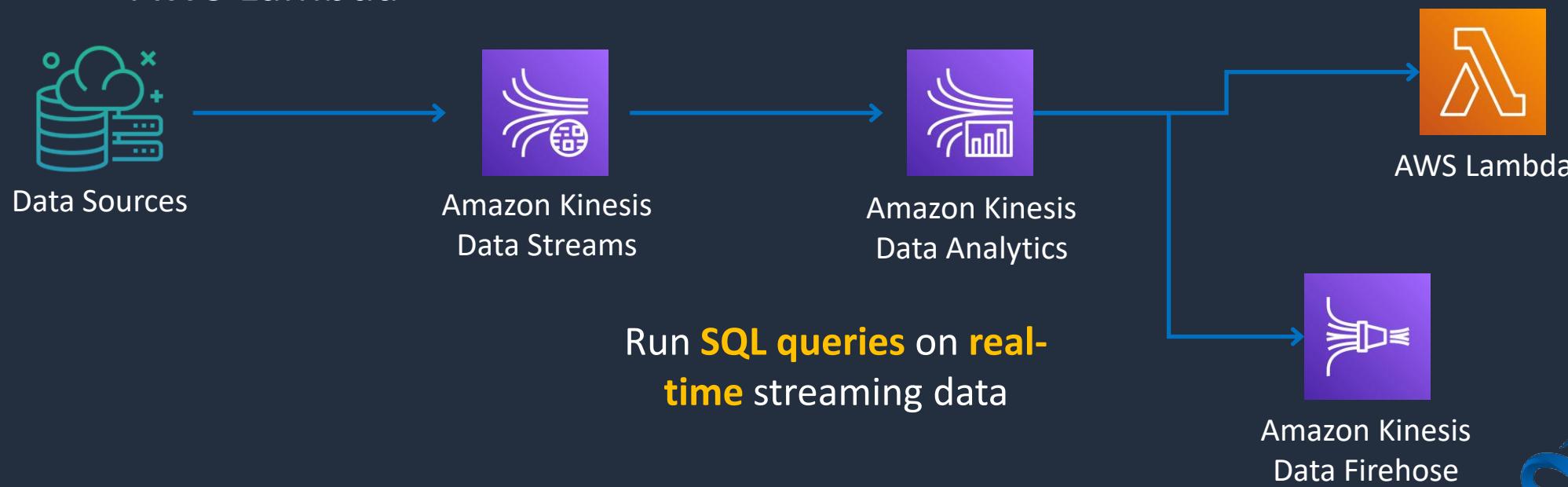
Kinesis Data Firehose destinations (changes over time):

- RedShift (via an intermediate S3 bucket)
- OpenSearch
- Amazon S3
- Splunk
- Datadog
- Honeycomb
- Coralogix
- LogicMonitor
- Logz.io
- MongoDB
- New Relic
- Sumo Logic
- HTTP Endpoint



Kinesis Data Analytics

- Provides real-time SQL processing for streaming data
- Provides analytics for data coming in from Kinesis Data Streams and Kinesis Data Firehose
- Destinations can be Kinesis Data Streams, Kinesis Data Firehose, or AWS Lambda

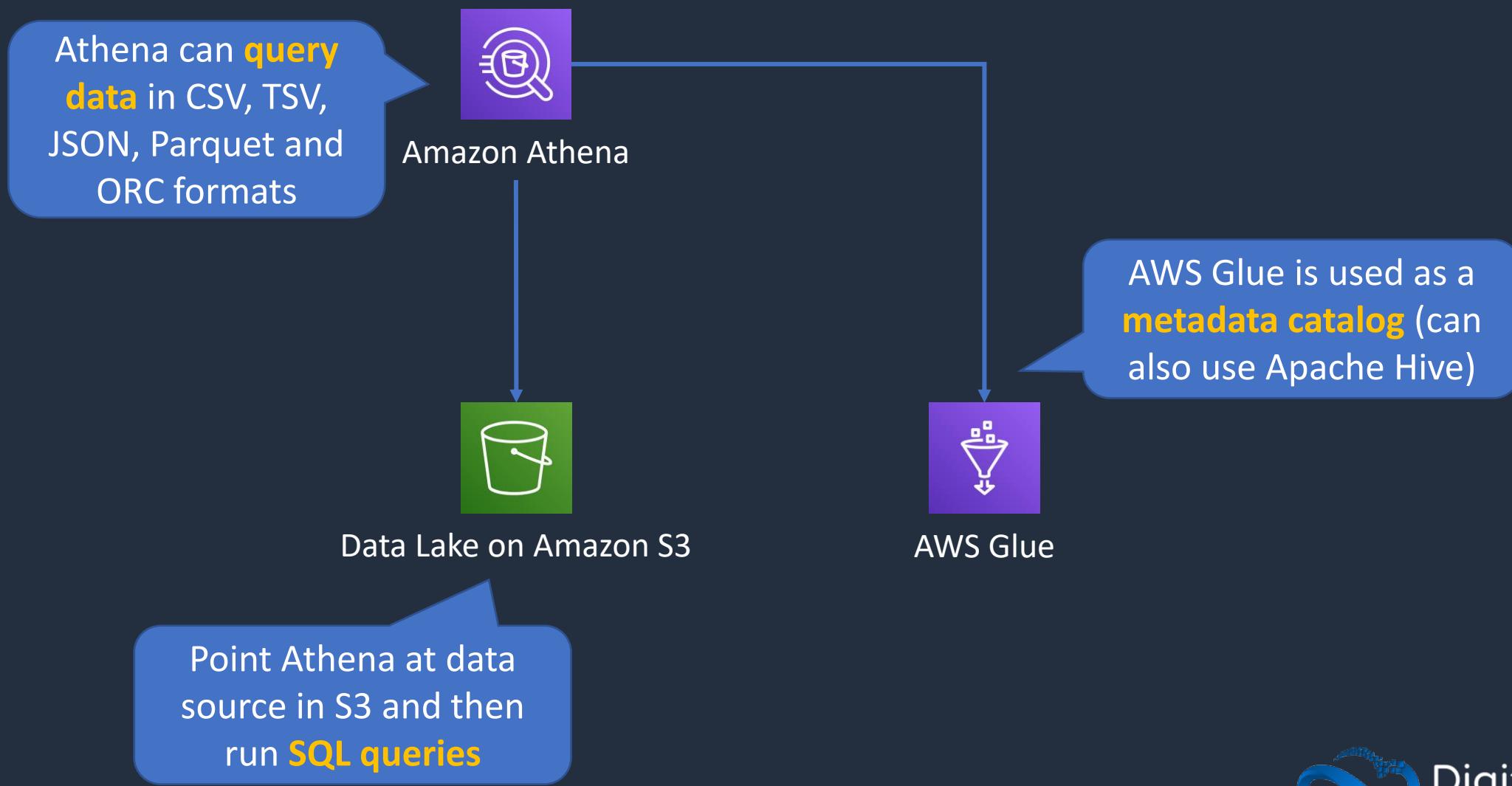


Amazon Athena and AWS Glue



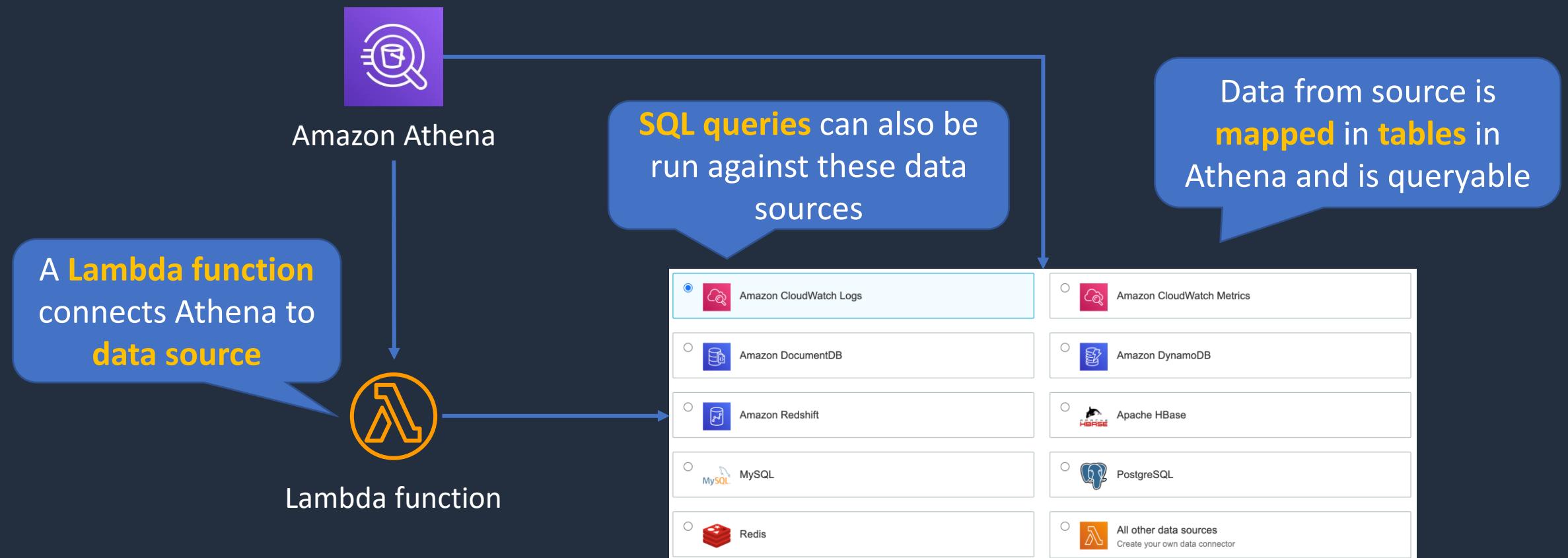


Amazon Athena and AWS Glue





Amazon Athena and AWS Glue





Amazon Athena

- Athena queries data in S3 using SQL
- Can be connected to other data sources with Lambda
- Data can be in CSV, TSV, JSON, Parquet and ORC formats
- Uses a managed Data Catalog (AWS Glue) to store information and schemas about the databases and tables



Optimizing Athena for Performance

- **Partition your data**
- **Bucket your data** – bucket the data within a single partition
- **Use Compression** – AWS recommend using either Apache Parquet or Apache ORC
- **Optimize file sizes**
- **Optimize columnar data store generation** – Apache Parquet and Apache ORC are popular columnar data stores
- **Optimize ORDER BY and Optimize GROUP BY**
- **Use approximate functions**
- **Only include the columns that you need**



AWS Glue

- Fully managed extract, transform and load (ETL) service
- Used for preparing data for analytics
- AWS Glue runs the ETL jobs on a fully managed, scale-out Apache Spark environment
- AWS Glue discovers data and stores the associated metadata (e.g. table definition and schema) in the AWS Glue Data Catalog
- Works with data lakes (e.g. data on S3), data warehouses (including RedShift), and data stores (including RDS or EC2 databases)



AWS Glue

- You can use a **crawler** to populate the AWS Glue Data Catalog with tables
- A crawler can crawl multiple data stores in a single run
- Upon completion, the crawler creates or updates one or more tables in your Data Catalog.
- ETL jobs that you define in AWS Glue use the Data Catalog tables as sources and targets

Query S3 ALB Access Logs with Athena



Amazon OpenSearch Service (Elasticsearch)





Amazon OpenSearch Service



Successor to **Amazon Elasticsearch Service**

Search, visualize, and analyze **text and unstructured data**

Amazon OpenSearch Service

Deploy **nodes** and **replicas** across AZs



Fully Managed



Petabyte Scale



Secure



Highly Available



Scalable

Deploy to **Amazon VPC** and integrates with **IAM**



Amazon OpenSearch Service

- Distributed search and analytics suite
- Based on the popular open source Elasticsearch
- Supports queries using SQL syntax
- Integrates with open-source tools
- Scale by adding or removing instances
- Availability in up to three Availability Zones
- Backup using snapshots
- Encryption at-rest and in-transit

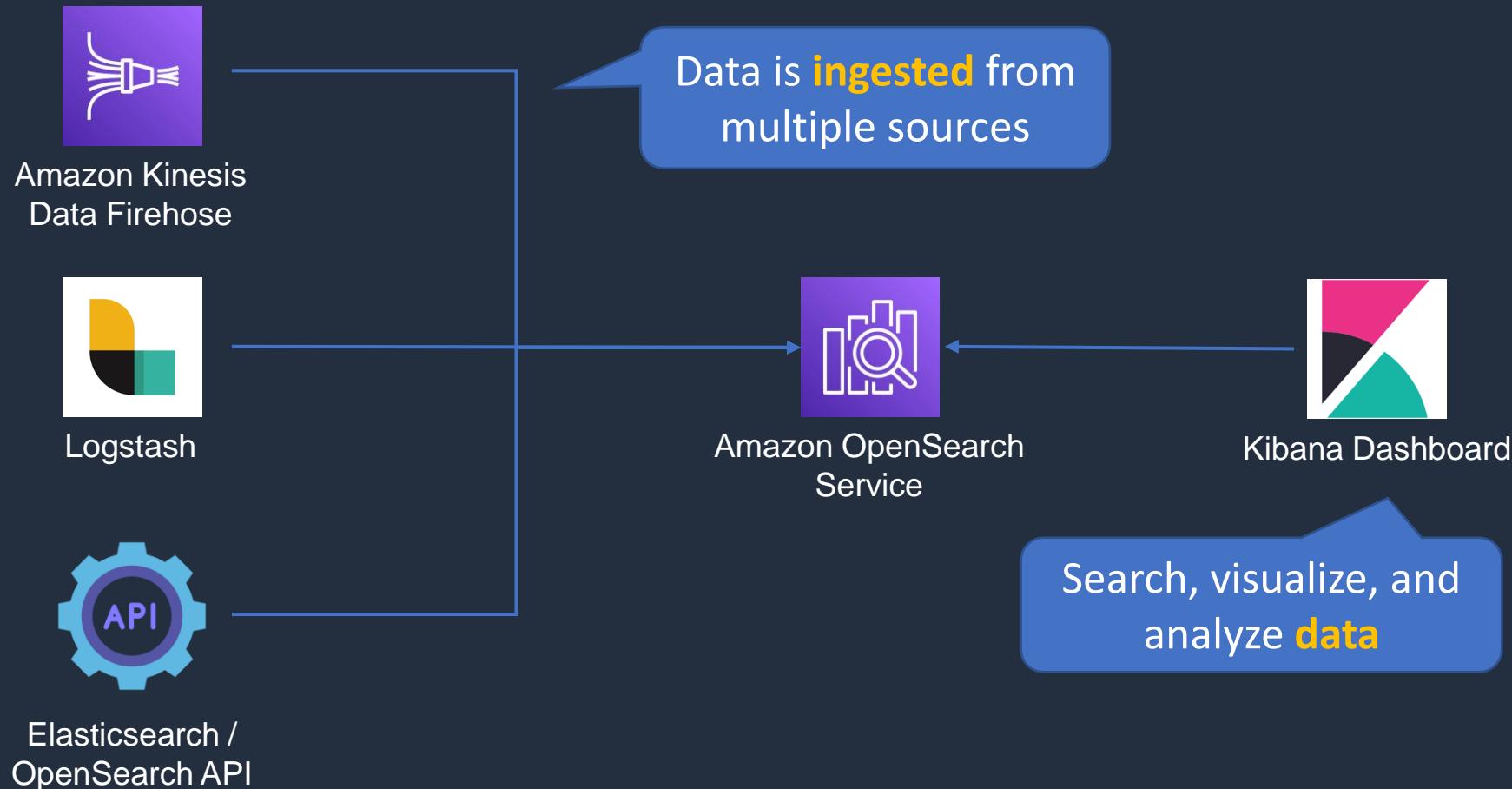


OpenSearch Service Deployment

- Clusters are created (Management Console, API, or CLI)
- Clusters are also known as OpenSearch Service domains
- You specify the number of instances and instance types
- Storage options include UltraWarm or Cold storage



Ingesting Data into OpenSearch Service Domains





OpenSearch in an Amazon VPC

- Clusters can be deployed in a VPC for secure intra-VPC communications
- VPN or proxy required to connect from the internet (public domains are directly accessible)
- Cannot use IP-based access policies



OpenSearch in an Amazon VPC

- Limitations of VPC deployments:
 - You can't switch from VPC to a public endpoint. The reverse is also true
 - You can't launch your domain within a VPC that uses dedicated tenancy
 - After you place a domain within a VPC, you can't move it to a different VPC, but you can change the subnets and security group settings



The ELK Stack

- ELK stands for Elasticsearch, Logstash, and Kibana



- This is a popular combination of projects
- Aggregate logs from systems and applications, analyze these logs, and create visualizations
- Use cases include:
 - Visualizing application and infrastructure monitoring data
 - Troubleshooting
 - Security analytics



OpenSearch Access Control

- **Resource-based policies** – often called a domain access policy
- **Identity-based policies** – attached to users or roles (principals)
- **IP-based policies** – Restrict access to one or more IP addresses or CIDR blocks
- **Fine-grained access control** – Provides:
 - Role-based access control
 - Security at the index, document, and field level
 - OpenSearch Dashboards multi-tenancy
 - HTTP basic authentication for OpenSearch and OpenSearch Dashboards



OpenSearch Access Control

- Authentication options include:
 - Federation using SAML to on-premises directories
 - Amazon Cognito and social identity providers



OpenSearch Best Practices

- Deploy OpenSearch data instances across three Availability Zones (AZs) for the best availability
- Provision instances in multiples of three for equal distribution across AZs
- If three AZs are not available use two AZs with equal numbers of instances



OpenSearch Best Practices

- Use three dedicated master nodes
- Configure at least one replica for each index
- Apply restrictive resource-based access policies to the domain (or use fine-grained access control)
- Create the domain within an Amazon VPC
- For sensitive data enable node-to-node encryption and encryption at rest

AWS Batch





AWS Batch



Launch a **Batch Job**



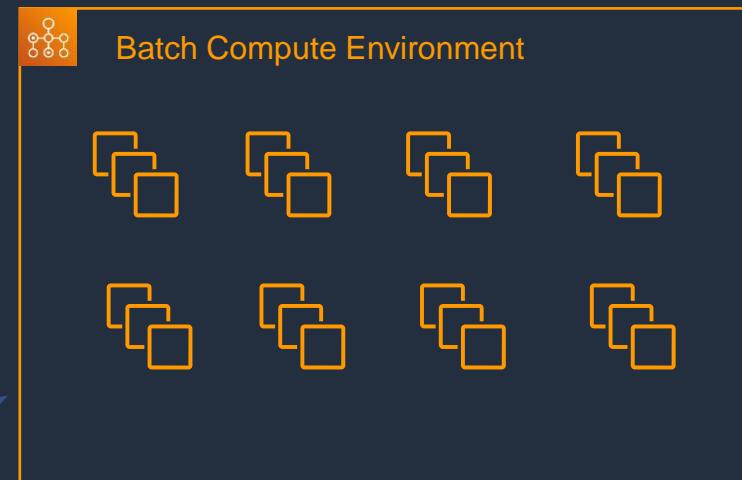
Job **Definition**



A job is submitted to a **queue** until **scheduled** onto a compute environment

A job is a unit of work such as a **shell script**, **executable** or **Docker container image**

Batch **launches**, **manages**, and **terminates** resources as required (EC2 and ECS/Fargate)



Managed or **unmanaged** resources used to run the job

Other Database Services





Amazon DocumentDB

- Amazon DocumentDB provides MongoDB compatibility
- It is a database service that is purpose-built for JSON data management at scale

```
JSON
1  [
2    {
3      "year" : 2013,
4      "title" : "Turn It Down, Or Else!",
5      "info" : {
6        "directors" : [ "Alice Smith", "Bob Jones"],
7        "release_date" : "2013-01-18T00:00:00Z",
8        "rating" : 6.2,
9        "genres" : ["Comedy", "Drama"],
10       "image_url" : "http://ia.media-imdb.com/images/N/09ERWAU7FS797AJ7LU8HN09AMUP908RLlo5JF90EWR7LJKQ7@._V1_SX400_.jpg",
11       "plot" : "A rock band plays their music at high volumes, annoying the neighbors.",
12       "actors" : ["David Matthewman", "Jonathan G. Neff"]
13     }
14   },
15   {
16     "year": 2015,
17     "title": "The Big New Movie",
18     "info": {
19       "plot": "Nothing happens at all.",
20       "rating": 0
21     }
22   }
23 ]
```



Amazon DocumentDB

- Fully managed service
- Storage scales automatically up to 64 TB without any impact to your application
- Supports millions of requests per second with up to 15 low latency read replicas
- Designed for 99.99% availability and replicates six copies of your data across three AZs
- Can migrate from MongoDB using the AWS Database Migration Service (AWS DMS)



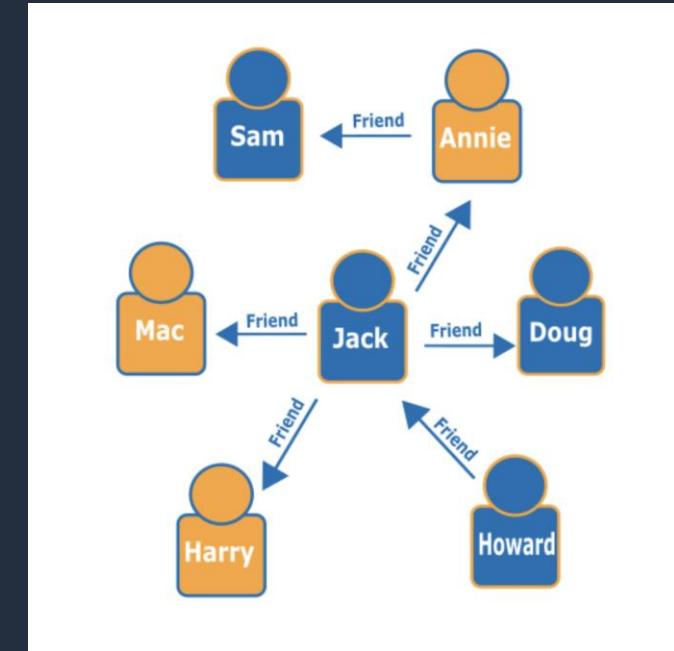
Amazon Keyspaces (for Apache Cassandra)

- A scalable, highly available, and managed Apache Cassandra-compatible database service
- Keyspaces enables you to use the Cassandra Query Language (CQL) API code
- Keyspaces is serverless and fully managed
- Scales automatically in response to application traffic
- Can serve thousands of requests per second with virtually unlimited throughput and storage
- Consistent, single-digit-millisecond response times at any scale
- 99.99% availability SLA within an AWS Region



Amazon Neptune

- Fully managed graph database service
- Build and run identity, knowledge, fraud graph, and other applications
- Deploy high performance graph applications using popular open-source APIs including:
 - Gremlin
 - openCypher
 - SPARQL
- Offers greater than 99.99% availability
- Storage is fault-tolerant and self-healing
- DB volumes grow in increments of 10 GB up to a maximum of 64 TB
- Create up to 15 database read replicas





Amazon Quantum Ledger Database

- Amazon QLDB is a fully managed ledger database that provides a transparent, immutable, and cryptographically verifiable transaction log
- Amazon QLDB has a built-in immutable journal that stores an accurate and sequenced entry of every data change
- The journal is append-only, meaning that data can only be added to a journal, and it cannot be overwritten or deleted
- Amazon QLDB uses cryptography to create a concise summary of your change history
- Generated using a cryptographic hash function (SHA-256)
- Serverless and offers automatic scalability

Other Analytics Services





Amazon Timestream

- Time series database service for IoT and operational applications
- Faster and cheaper than relational databases
- Keeps recent data in memory and moves historical data to a cost optimized storage tier based upon user defined policies
- Serverless and scales automatically



AWS Data Exchange

- AWS Data Exchange is a data marketplace with over 3,000 products from 250+ providers
- AWS Data Exchange supports Data Files, Data Tables, and Data APIs
- Consume directly into data lakes, applications, analytics, and machine learning models
- Automatically export new or updated data sets to Amazon S3
- Query data tables with AWS Data Exchange for Amazon Redshift
- Use AWS-native authentication and governance, AWS SDKs, and consistent API documentation



AWS Data Exchange - Benefits



Extensive Data Set Selection

- 3,000+ data sets from 250+ data providers
- Over 1,000 free data products and custom data products
- Automatic access to new data



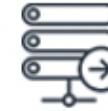
Streamlined Data Procurement & Governance

- One place to exchange data publicly or privately
- Migrate existing subscriptions at no additional cost
- Simplified contracts and consolidated secure billing



Better Data Technology

- Native integration of data into AWS
- Data encrypted at rest and in transit
- Integration with AWS Identity & Access Management (IAM)



Ease of Use for Data Analytics & Machine Learning

- Cloud native support for data files, tables, and APIs
- Self-service options
- Secure and compliant solutions



AWS Data Pipeline

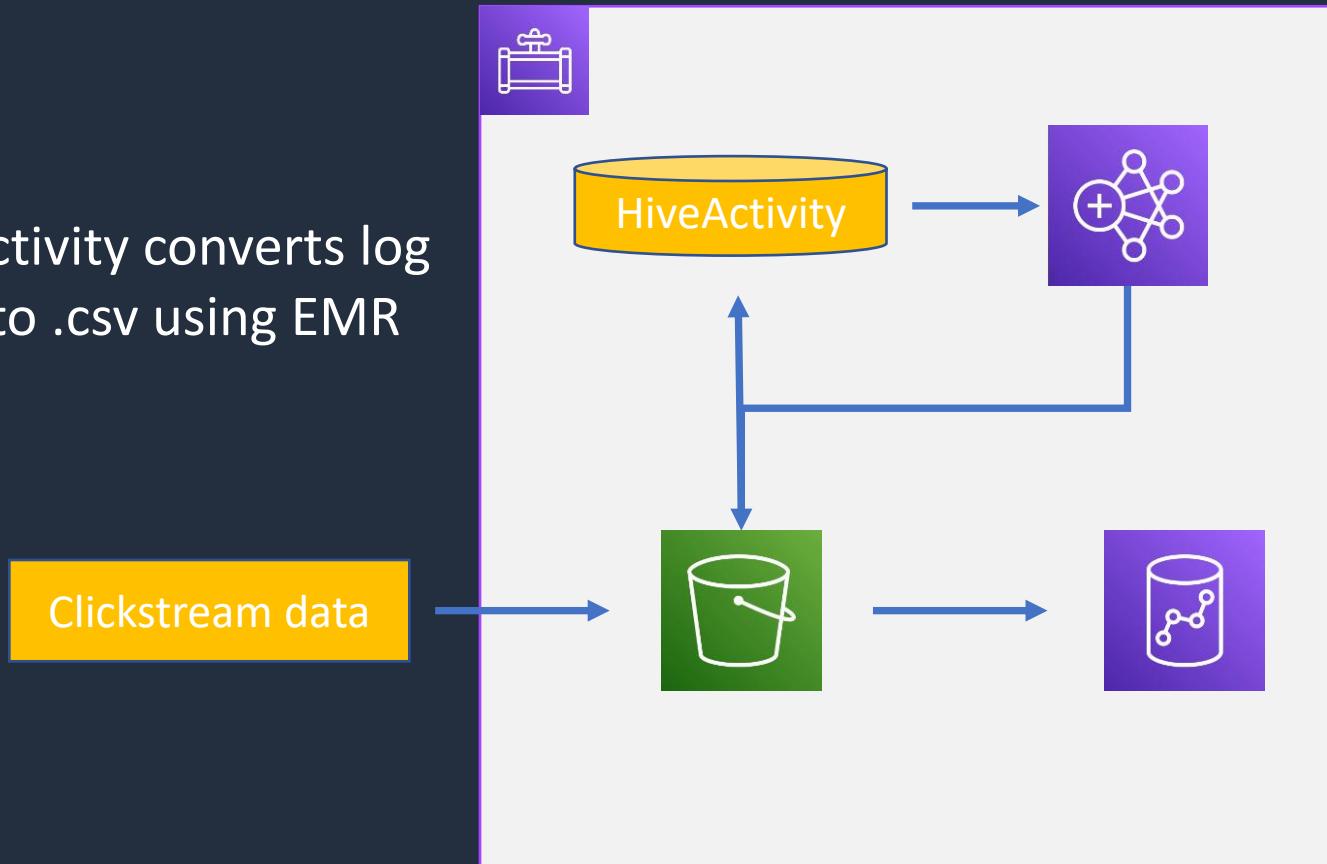
- AWS Data Pipeline is a managed ETL (Extract-Transform-Load) service
- Process and move data between different AWS compute and storage services
- Data sources can also be on-premises
- Data can be processed and transformed
- Results can be loaded to services such as Amazon S3, Amazon RDS, Amazon DynamoDB, and Amazon EMR



AWS Data Pipeline – Clickstream Data Example

Pipeline

HiveActivity converts log files to .csv using EMR



Results saved to S3

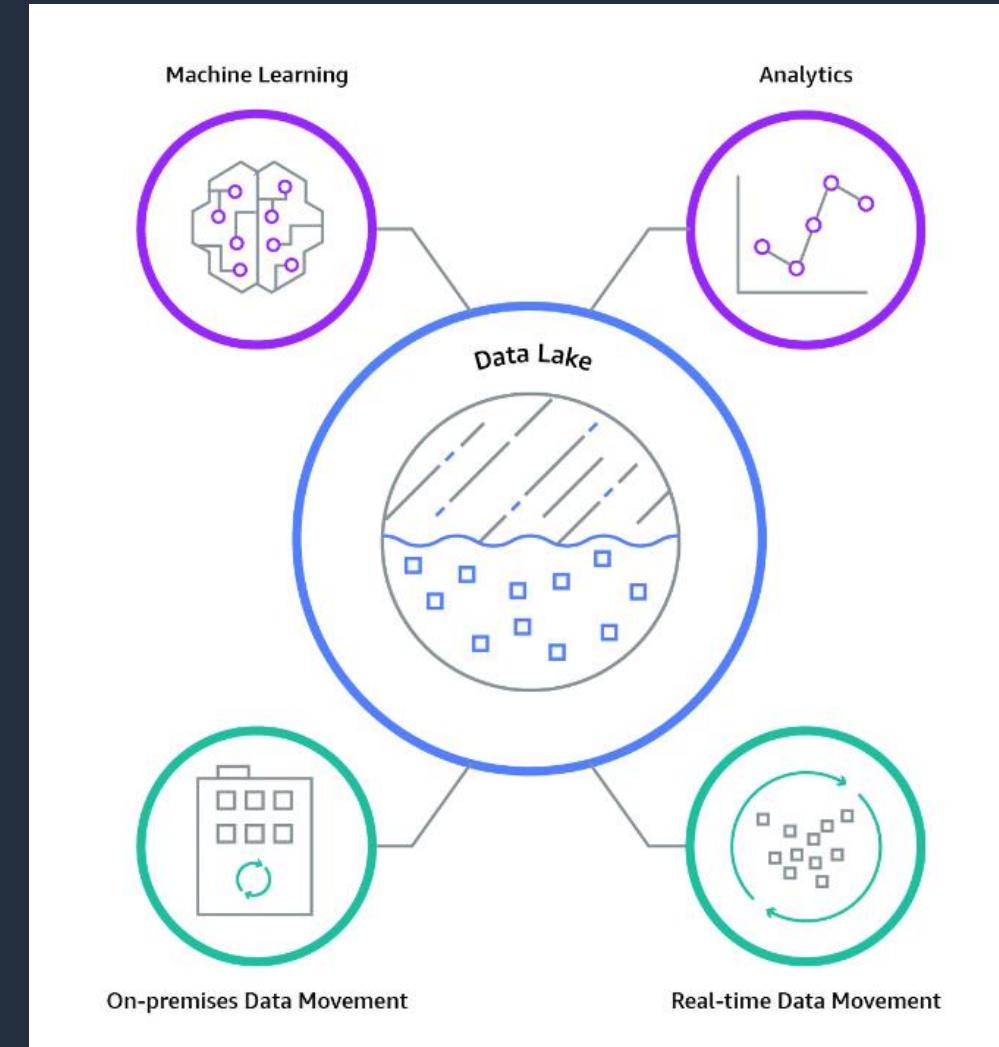
RedShiftCopyActivity loads to RedShift



What is a Data Lake?

A data lake is a centralized repository that allows you to store all your structured and unstructured data at any scale

Can create dashboards, and visualizations



You can store your data as-is, without having to first structure the data

Big data processing, real-time analytics, ML



Data Warehouse vs Data Lake

Characteristics	Data Warehouse	Data Lake
Data	Relational from transactional systems, operational databases, and line of business applications	Non-relational and relational from IoT devices, web sites, mobile apps, social media, and corporate applications
Schema	Designed prior to the DW implementation (schema-on-write)	Written at the time of analysis (schema-on-read)
Price/Performance	Fastest query results using higher cost storage	Query results getting faster using low-cost storage
Data Quality	Highly curated data that serves as the central version of the truth	Any data that may or may not be curated (ie. raw data)
Users	Business analysts	Data scientists, Data developers, and Business analysts (using curated data)
Analytics	Batch reporting, BI and visualizations	Machine Learning, Predictive analytics, data discovery and profiling



AWS Lake Formation

- AWS Lake Formation enables you to set up secure data lakes in days
- Data can be collected from databases and object storage
- It is saved to the Amazon S3 data lake
- You can then clean and classify data using ML algorithms
- Security can be applied at column, row, and cell-levels
- The data sets can then be used through services such as Amazon Redshift, Amazon Athena, Amazon EMR for Apache Spark, and Amazon QuickSight
- Lake Formation builds on the capabilities available in AWS Glue



Amazon Managed Streaming for Apache Kafka (MSK)

- Amazon MSK is used for ingesting and processing streaming data in real-time
- Build and run Apache Kafka applications
- It is a fully managed service
- Provisions, configures, and maintains Apache Kafka clusters and Apache ZooKeeper nodes
- Security levels include:
 - VPC network isolation
 - AWS IAM for control-plane API authorization
 - Encryption at rest
 - TLS encryption in-transit
 - TLS based certificate authentication
 - SASL/SCRAM authentication secured by AWS Secrets Manager

Architecture Patterns – Databases and Analytics





Architecture Patterns – Databases & Analytics

Requirement

Relational database running on MySQL must be migrated to AWS and must be highly available

Solution

Use Amazon RDS MySQL and configure a Multi-AZ standby node for HA

Amazon RDS DB has high query traffic that is causing performance degradation

Create a Read Replica and configure the application to use the reader endpoint for database queries

Amazon RDS DB is approaching its storage capacity limits and/or is suffering from high write latency

Scale up the DB instance to an instance type that has more storage / CPU



Architecture Patterns – Databases & Analytics

Requirement

Amazon RDS database is unencrypted and a cross-Region read replica must be created with encryption

Solution

Encrypt a snapshot of the main DB and create a new encrypted DB instance from the encrypted snapshot. Create a encrypted cross-Region read replica

Amazon Aurora DB deployed and requires a cross-Region replica

Deploy an Aurora MySQL Replica in the second Region

Amazon Aurora DB deployed and requires a read replica in the same Region with minimal synchronization latency

Deploy an Aurora Replica in the Region in a different Availability Zone



Architecture Patterns – Databases & Analytics

Requirement

Aurora deployed and app in another Region requires read-only access with low latency – synchronization latency must also be minimized

Solution

Use Aurora Global Database and configure the app in the second Region to use the reader endpoint

Application and DB migrated to Aurora and requires the ability to write to the DB across multiple nodes

Use Aurora Multi-Master for an in-Region multi-master database

Application requires a session-state data store that provides low-latency

Use either Amazon ElastiCache or DynamoDB



Architecture Patterns – Databases & Analytics

Requirement

Multi-threaded in-memory datastore required for unstructured data

In-memory datastore required that offers microsecond performance for unstructured data

In-memory datastore required that supports data persistence and high availability

Solution

Use Amazon ElastiCache Memcached

Use Amazon DynamoDB DAX (DAX)

Use Amazon ElastiCache Redis



Architecture Patterns – Databases & Analytics

Requirement

Serverless database required that supports No-SQL key-value store workload

Solution

Use Amazon DynamoDB

Serverless database required that supports MySQL or PostgreSQL

Use Amazon Aurora Serverless

Relational database required for a workload with an unknown usage pattern (usage expected to be low and variable)

Use Amazon Aurora Serverless



Architecture Patterns – Databases & Analytics

Requirement

Application requires a key-value database that can be written to from multiple AWS Regions

Solution

Use DynamoDB Global Tables

Athena is being used to analyze a large volume of data based on date ranges. Performance must be optimized

Store data using Apache Hive partitioning with a key based on the data. Use the Apache Parquet and ORC storage formats

Lambda is processing streaming data from API Gateway and is generating a TooManyRequestsException as volume increases

Stream the data into a Kinesis Data Stream from API Gateway and process in batches



Architecture Patterns – Databases & Analytics

Requirement

Lambda function is processing streaming data that must be analyzed with SQL

Solution

Load data into a Kinesis Data Stream and then analyze with Kinesis Data Analytics

Security logs generated by AWS WAF must be sent to a third-party auditing application

Send logs to Kinesis Data Firehose and configure the auditing application using an HTTP endpoint

Real-time streaming data must be stored for future analysis

Ingest data into a Kinesis Data Stream and then use Firehose to load to a data store for later analysis



Architecture Patterns – Databases & Analytics

Requirement

Company runs several production databases and must run complex queries across consolidated data set for business forecasting

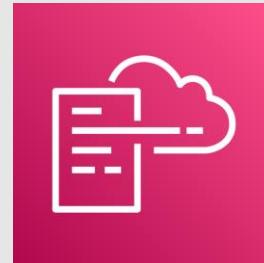
Solution

Load the data from the OLTP databases into a RedShift data warehouse for OLAP

SECTION 13

Deployment and Management

Infrastructure as Code with AWS CloudFormation





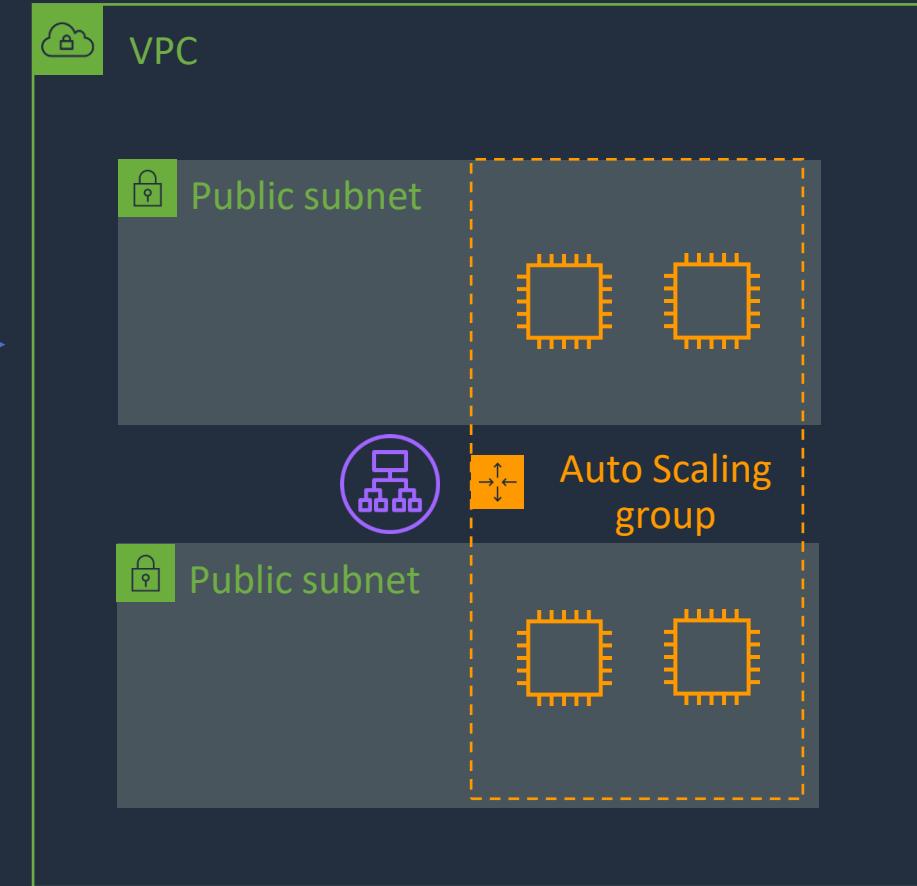
AWS CloudFormation

Infrastructure patterns are defined in a **template** file using **code**



CloudFormation **builds** your infrastructure according to the **template**

```
1 "AWSTemplateFormatVersion": "2010-09-09",
2
3 "Description": "AWS CloudFormation Sample Template WordPress_Multi_AZ: WordPress is web
4
5 "Parameters": {
6   "VpcId": {
7     "Type": "AWS::EC2::VPC::Id",
8     "Description": "VpcId of your existing Virtual Private Cloud (VPC)",
9     "ConstraintDescription": "must be the VPC Id of an existing Virtual Private Cloud."
10 },
11
12 "Subnets": {
13   "Type": "List<AWS::EC2::Subnet::Id>",
14   "Description": "The list of SubnetIds in your Virtual Private Cloud (VPC)",
15   "ConstraintDescription": "must be a list of at least two existing subnets associated
16 },
```





AWS CloudFormation

Component	Description
Templates	The JSON or YAML text file that contains the instructions for building out the AWS environment
Stacks	The entire environment described by the template and created, updated, and deleted as a single unit
StackSets	AWS CloudFormation StackSets extends the functionality of stacks by enabling you to create, update, or delete stacks across multiple accounts and regions with a single operation
Change Sets	A summary of proposed changes to your stack that will allow you to see how those changes might impact your existing resources before implementing them

Creating and Updating Stacks



Create Nested Stack using the AWS CLI

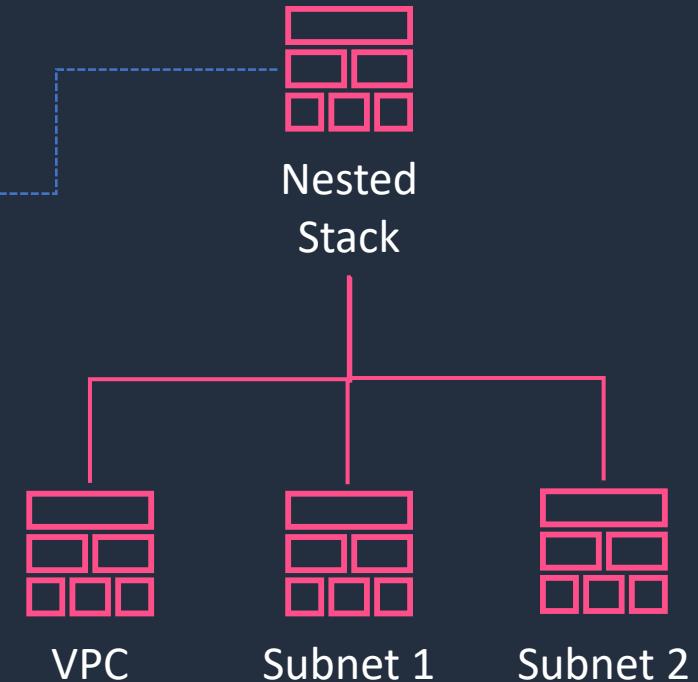




CloudFormation Nested Stacks

The `AWS::CloudFormation::Stack` resource nests a stack as a resource in a top-level template

```
AWSTemplateFormatVersion: '2010-09-09'
Resources:
  VPCStack:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: https://my-cloudformation-s3-bucket-3121s2.s3.amazonaws.com/vpc.yaml
  Subnet1Stack:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: https://my-cloudformation-s3-bucket-3121s2.s3.amazonaws.com/subnet1.yaml
      Parameters:
        VpcId: !GetAtt VPCStack.Outputs.VpcId
  Subnet2Stack:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: https://my-cloudformation-s3-bucket-3121s2.s3.amazonaws.com/subnet2.yaml
      Parameters:
        VpcId: !GetAtt VPCStack.Outputs.VpcId
```



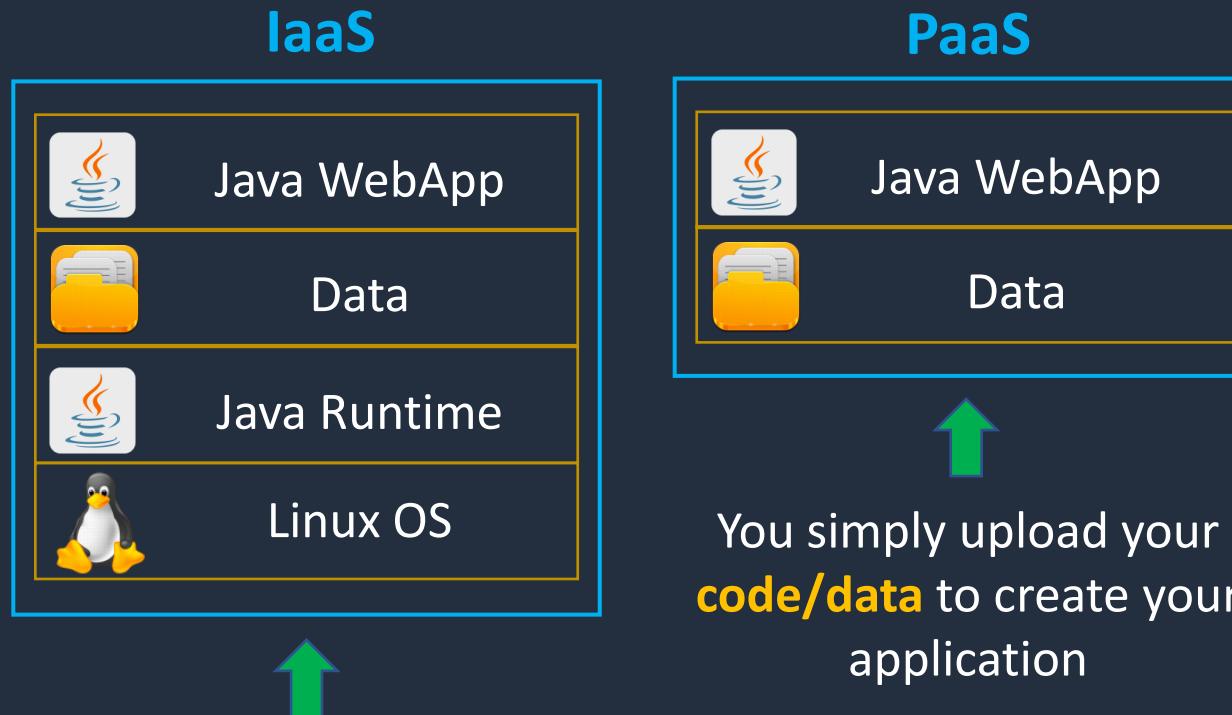
Platform as a Service with AWS Elastic Beanstalk





Cloud Service Models: Comparison

Example is
Amazon EC2

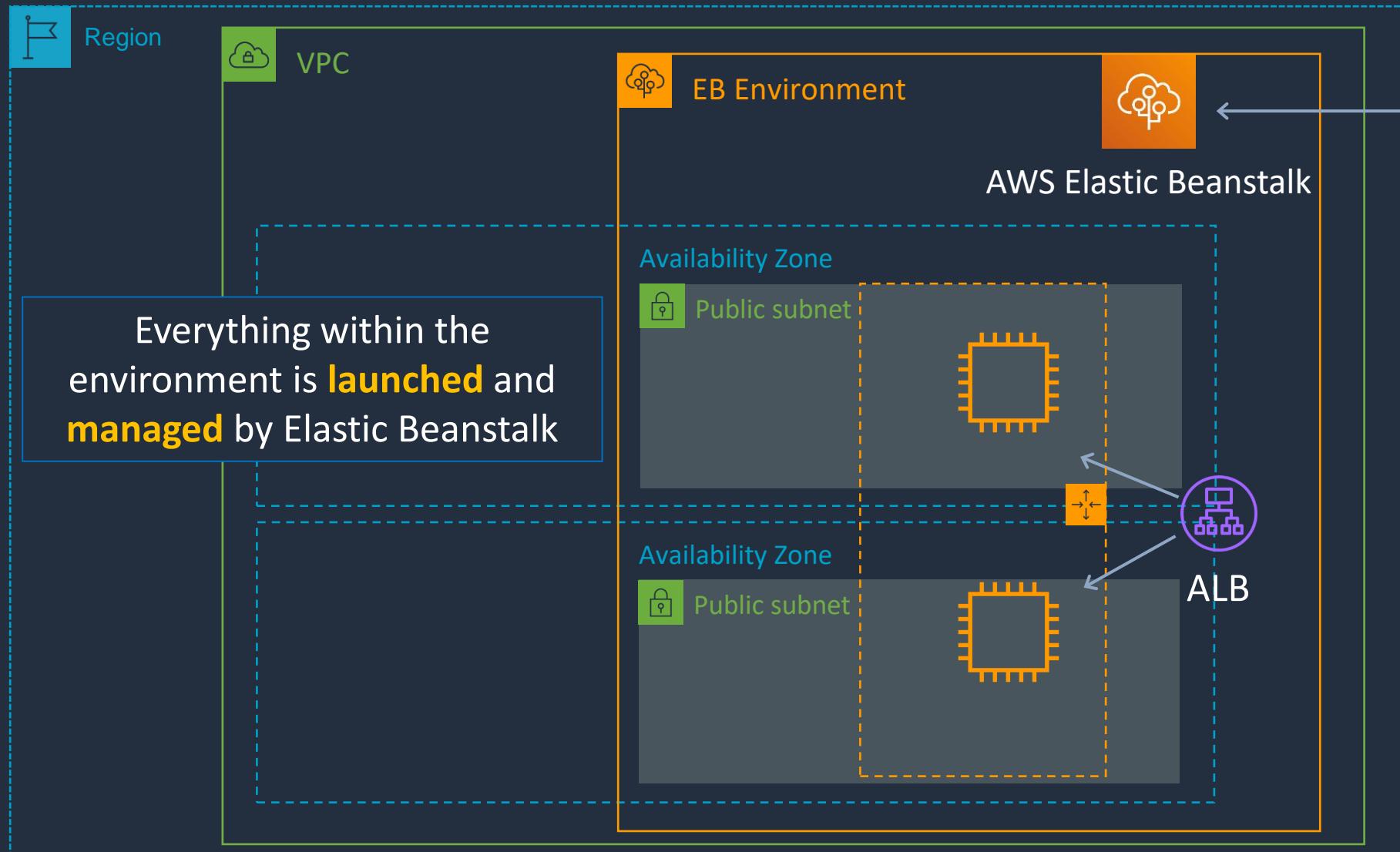


You manage from the
virtual server upwards

Example is **AWS Elastic Beanstalk**



AWS Elastic Beanstalk



Upload **source code** in ZIP file



Developer Client



AWS Elastic Beanstalk

- Supports many application platforms including:
 - Java, .NET, Node.js, PHP, Ruby, Python, Go, and Docker
- Uses core AWS services including EC2, ECS, Auto Scaling, and Elastic Load Balancing
- Elastic Beanstalk provides a UI to monitor and manage the health of applications
- Managed platform updates deploy the latest versions of software and patches



AWS Elastic Beanstalk

There are several **layers**

Applications:

- Contain environments, environment configurations, and application versions
- You can have multiple application versions held within an application

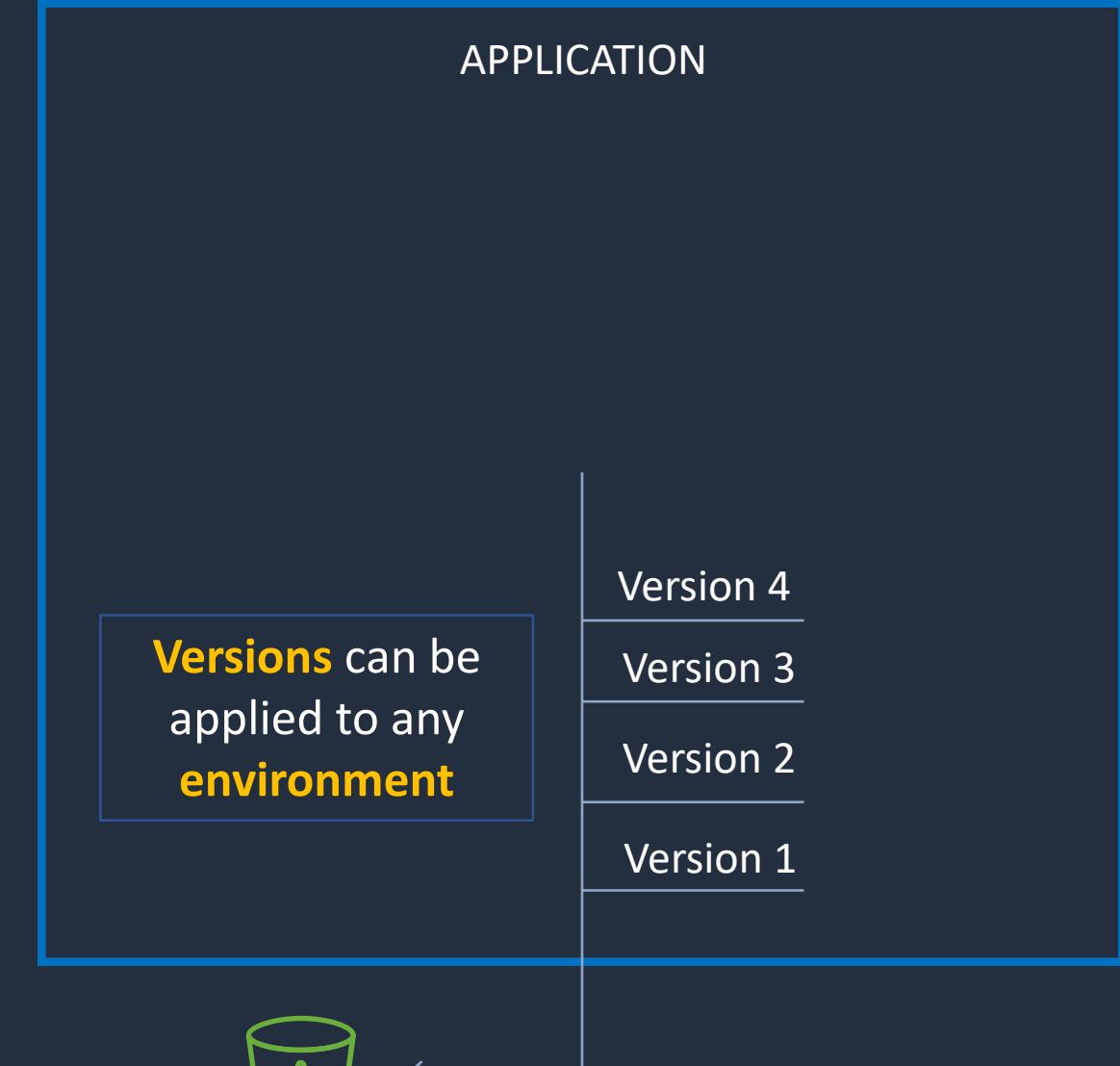
APPLICATION



AWS Elastic Beanstalk

Application version

- A specific reference to a section of deployable code
- The application version will point typically to an Amazon S3 bucket containing the code

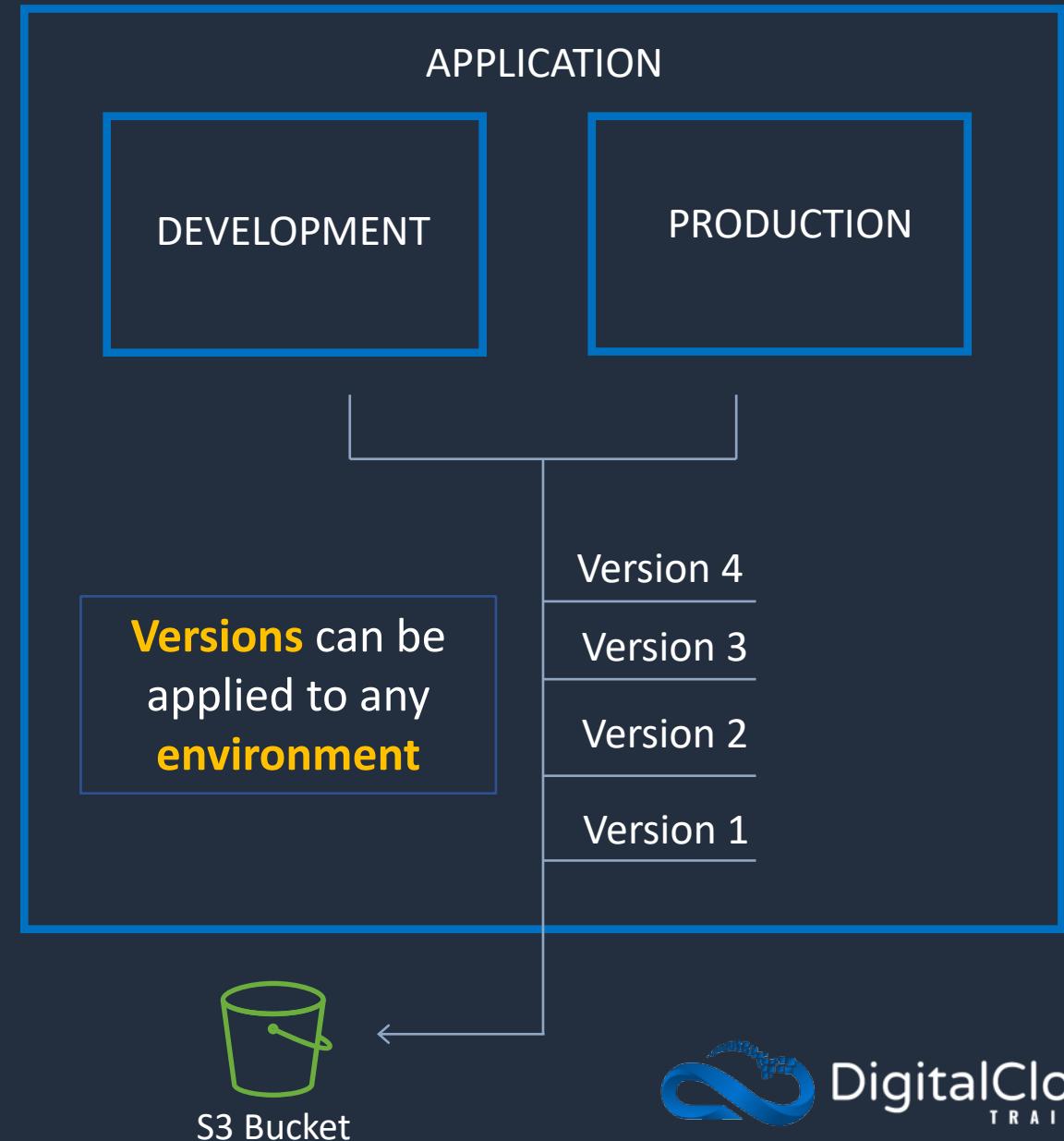




AWS Elastic Beanstalk

Environments:

- An application version that has been deployed on AWS resources
- The resources are configured and provisioned by AWS Elastic Beanstalk
- The environment is comprised of all the resources created by Elastic Beanstalk and not just an EC2 instance with your uploaded code



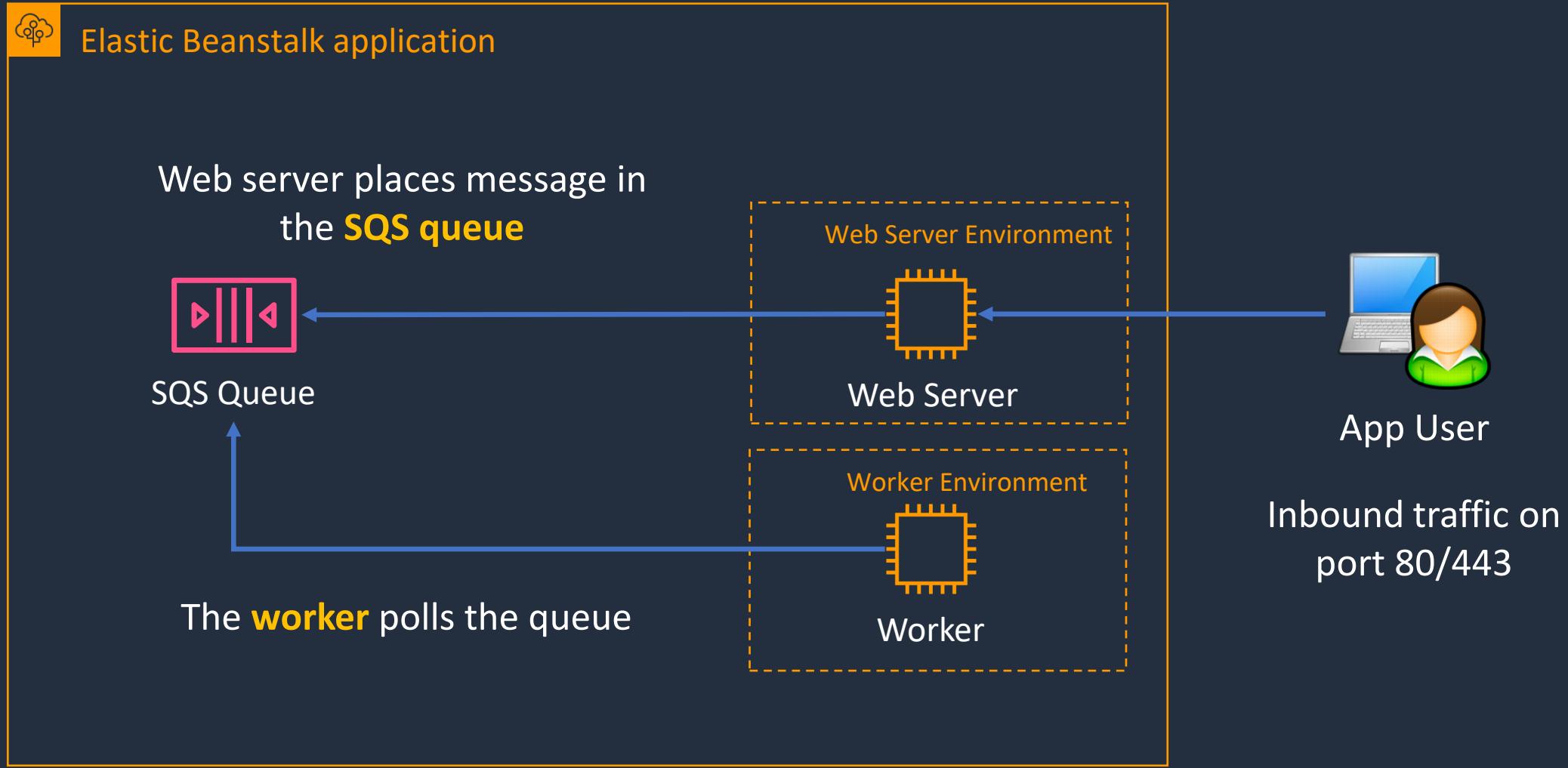


Web Servers and Workers

- **Web servers** are standard applications that listen for and then process HTTP requests, typically over port 80
- **Workers** are specialized applications that have a background processing task that listens for messages on an Amazon SQS queue
- **Workers** should be used for long-running tasks



AWS Elastic Beanstalk



Create an Elastic Beanstalk Application



Creating and Updating Environments



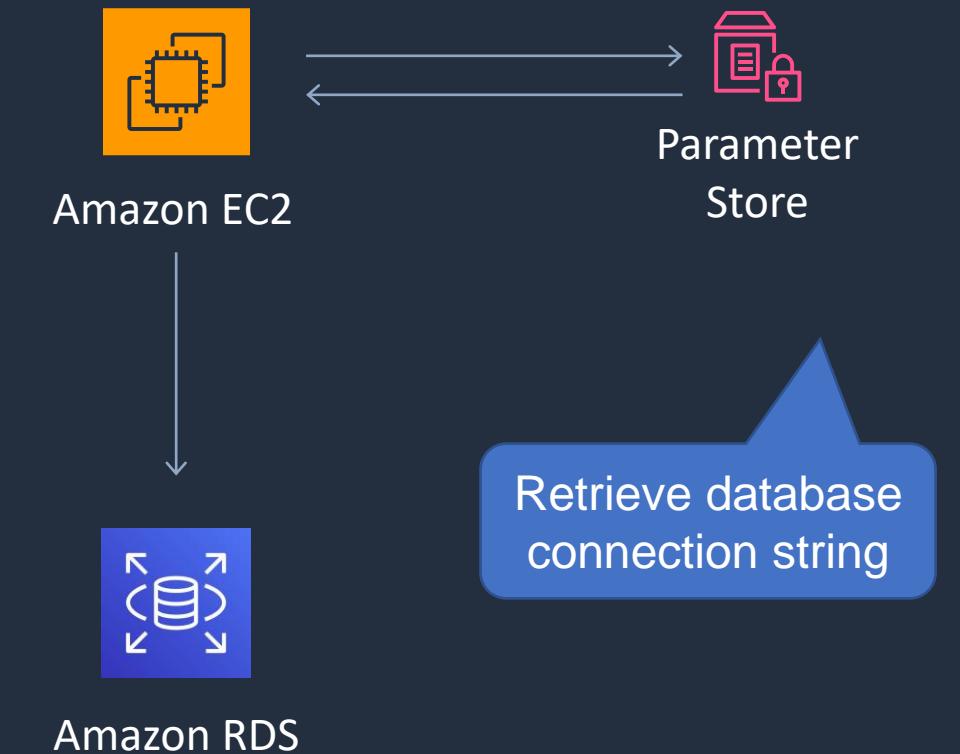
AWS Systems Manager Parameter Store



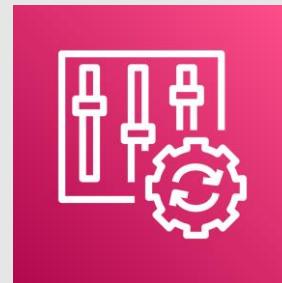


AWS SSM Parameter Store

- Parameter Store provides secure, hierarchical storage for configuration data and secrets
- Highly scalable, available, and durable
- Store data such as passwords, database strings, and license codes as parameter values
- Store values as plaintext (unencrypted data) or ciphertext (encrypted data)
- Reference values by using the unique name that you specified when you created the parameter
- No native rotation of keys (difference with AWS Secrets Manager which does it automatically)



AWS Config





AWS Config

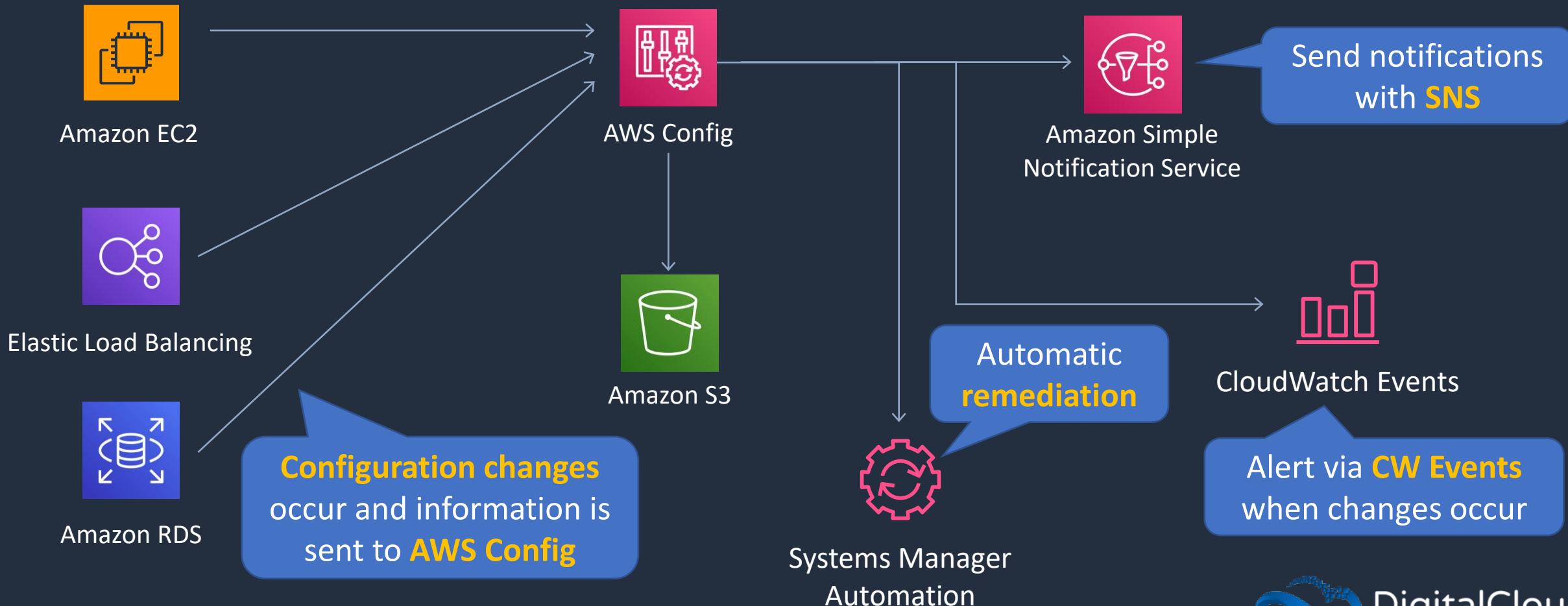
- Evaluate your AWS resource configurations for desired settings
- Get a snapshot of the current configurations of resources that are associated with your AWS account
- Retrieve configurations of resources that exist in your account
- Retrieve historical configurations of one or more resources
- Receive a notification whenever a resource is created, modified, or deleted
- View relationships between resources



AWS Config

AWS Config evaluates the **configuration** against desired configurations

Example Services:





AWS Config

Example Rule	Description
s3-bucket-server-side-encryption-enabled	Checks that your Amazon S3 bucket either has S3 default encryption enabled or that the S3 bucket policy explicitly denies put-object requests without server side encryption
restricted-ssh	Checks whether security groups that are in use disallow unrestricted incoming SSH traffic
rds-instance-public-access-check	Checks whether the Amazon Relational Database Service (RDS) instances are not publicly accessible
cloudtrail-enabled	Checks whether AWS CloudTrail is enabled in your AWS account

SSM Automation and Config Rules



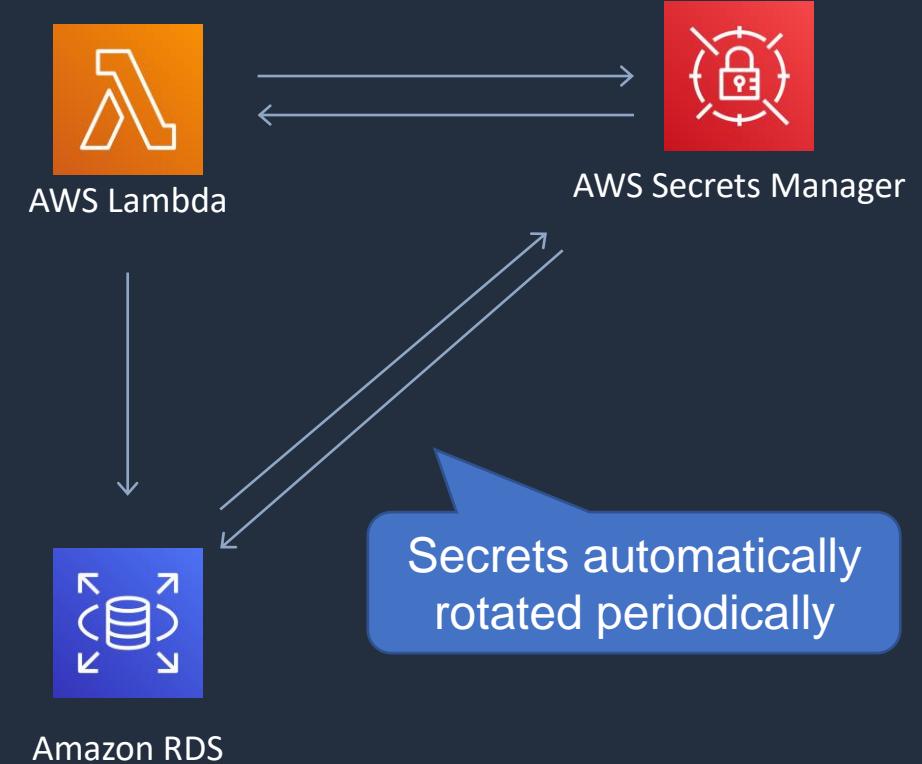
AWS Secrets Manager





AWS Secrets Manager

- Stores and rotate secrets safely without the need for code deployments
- Secrets Manager offers automatic rotation of credentials (built-in) for:
 - Amazon RDS (MySQL, PostgreSQL, and Amazon Aurora)
 - Amazon Redshift
 - Amazon DocumentDB
- For other services you can write your own AWS Lambda function for automatic rotation

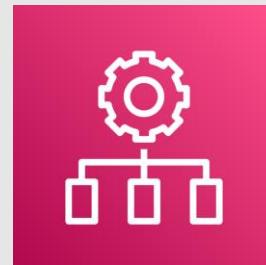




AWS Secrets Manager vs SSM Parameter Store

	Secrets Manager	SSM Parameter Store
Automatic Key Rotation	Yes, built-in for some services, use Lambda for others	No native key rotation; can use custom Lambda
Key/Value Type	String or Binary (encrypted)	String, StringList, SecureString (encrypted)
Hierarchical Keys	No	Yes
Price	Charges apply per secret	Free for standard, charges for advanced

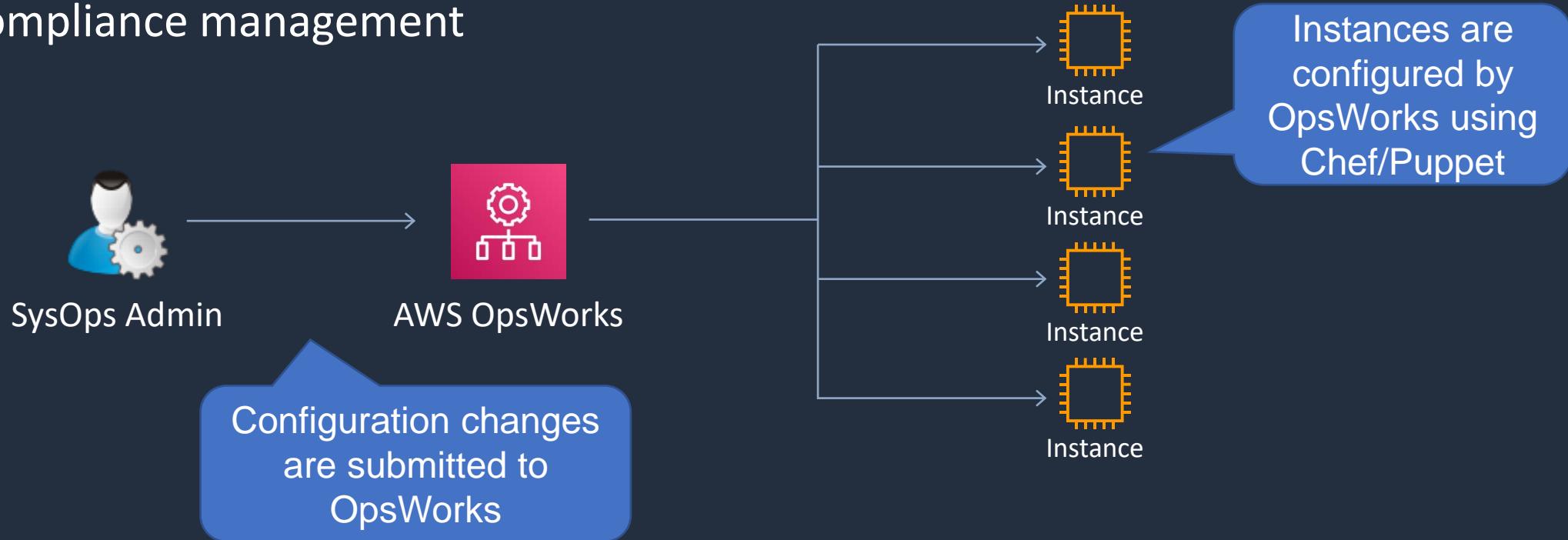
AWS OpsWorks





AWS OpsWorks

- AWS OpsWorks is a configuration management service that provides managed instances of Chef and Puppet
- Updates include patching, updating, backup, configuration and compliance management



AWS Resource Access Manager (RAM)





AWS RAM

- Shares resources:
 - Across AWS accounts
 - Within AWS Organizations or OUs
 - IAM roles and IAM users
- Resource shares are created with:
 - The AWS RAM Console
 - AWS RAM APIs
 - AWS CLI
 - AWS SDKs



AWS RAM

RAM can be used to share:

- AWS App Mesh
- Amazon Aurora
- AWS Certificate Manager Private Certificate Authority
- AWS CodeBuild
- Amazon EC2
- EC2 Image Builder
- AWS Glue
- AWS License Manager
- AWS Network Firewall
- AWS Outposts
- Amazon S3 on Outposts
- AWS Resource Groups
- Amazon Route 53
- AWS Systems Manager Incident Manager
- Amazon VPC

RPO, RTO, and DR Strategies





Recovery Point Objective (RPO)

- Measurement of the amount of data that can be acceptably lost
- Measured in seconds, minutes, or hours
- Example:
 - You can acceptably lose 2 hours of data in a database (2hr RPO)
 - This means backups must be taken every 2 hours

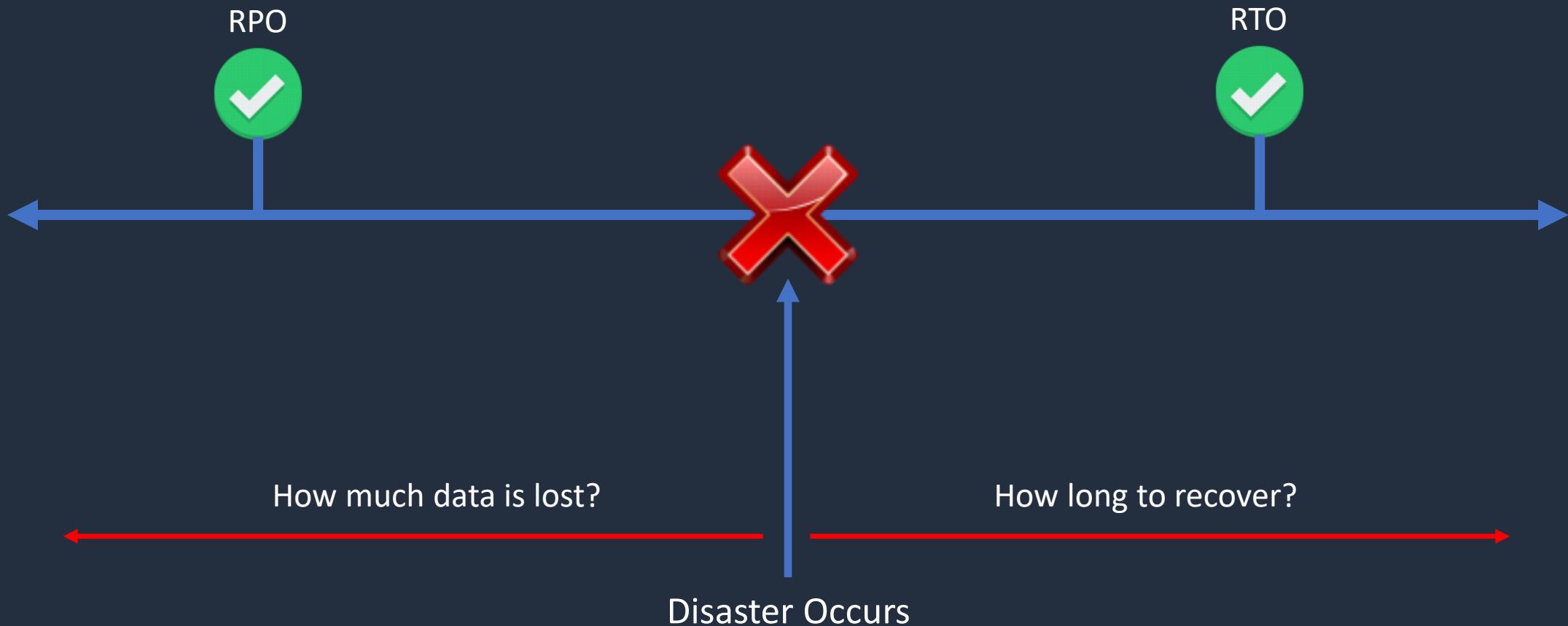


Recovery Time Objective (RTO)

- Measurement of the amount of time it takes to restore after a disaster event
- Measured in seconds, minutes, or hours
- Example:
 - The IT department expect it to take 4 hours to bring applications online after a disaster
 - This would be an RTO of 4 hours



RPO and RTO





Achievable RPOs

Recovery Point Objective	Technique
Milliseconds - Seconds	Synchronous replication
Seconds - Minutes	Asynchronous replication
Minutes to hours	Snapshots, cloud backup, D2D
Hours to days	Offsite / traditional backups / tape backups

The **RPO** is determined by how you take a backup of data



Achievable RTOs

Recovery Time Objective	Technique
Milliseconds / Seconds	Fault tolerance
Seconds - Minutes	High availability, load balancing, auto scaling
Minutes to hours	Cross-site recovery (cloud), automated recovery
Hours to days	Cross-site recovery (cloud/on-premises), manual recovery

The **RTO** is determined by how quickly you can recover



DR Strategies

Backup and Restore

- Low priority workloads
- Provision/restore after event
- Cost \$

Pilot Light

- Data replicated
- Services idle/off
- Resources activated after event
- Cost \$\$

Warm standby

- Minimum resources always running
- Business critical workloads
- Scale up/out after event
- Cost \$\$\$

Multi-site active/active

- Zero downtime
- Near zero data loss
- Mission critical workloads
- Cost\$\$\$\$

Architecture Patterns – Deployment and Management





Architecture Patterns – Deployment and Management

Requirement

Application must authenticate to Amazon Aurora and need to securely store credentials. Automatic credential rotation is required on a monthly basis

Solution

Use AWS Secrets Manager to store the credentials. Update the app to retrieve credentials from Secrets Manager; enable automatic monthly rotation

Company currently uses Chef cookbooks to manage infrastructure and is moving to the cloud. Need to minimize migration complexity

Use AWS OpsWorks for Chef Automate

Need a managed environment for running a simple web application. App processes incoming data which can take several minutes per task

Use an Elastic Beanstalk environment with a web server for the app front-end and a decoupled worker tier for the long running process



Architecture Patterns – Deployment and Management

Requirement

Systems integrator deploys standardized Amazon VPC configuration for many customers. Need to increase efficiency and reduce errors

Solution

Create an AWS CloudFormation template based on the standard config and use CloudFormation to deploy to new customers

Company has experienced issues rolling out updates to a CloudFormation Stack and needs to preview changes before the next update

Preview the change by creating a change set with the updated template before updating the stack

Manager wishes to monitor and enforce configuration compliance for AWS resources including S3 buckets and security groups

Use AWS Config to create rules to monitor compliance and use auto remediation to enforce compliance

SECTION 14

Monitoring, Logging, and Auditing

Amazon CloudWatch Overview





Amazon CloudWatch

CloudWatch is used for performance monitoring, alarms, log collection and automated actions

Use cases / benefits include:

- Collect performance metrics from AWS and on-premises systems
- Automate responses to operational changes
- Improve operational performance and resource optimization
- Derive actionable insights from logs
- Get operational visibility and insight



Amazon CloudWatch

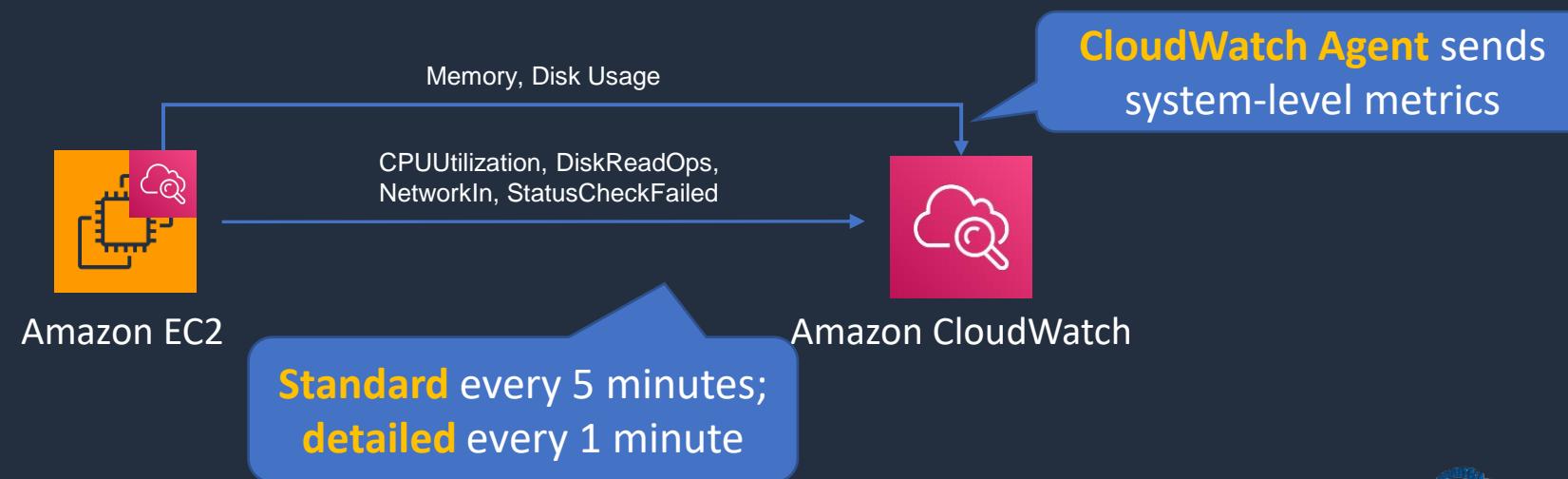
CloudWatch Core Features:

- **CloudWatch Metrics** – services send time-ordered data points to CloudWatch
- **CloudWatch Alarms** – monitor metrics and initiate actions
- **CloudWatch Logs** – centralized collection of system and application logs
- **CloudWatch Events** – stream of system events describing changes to AWS resources and can trigger actions



Amazon CloudWatch Metrics

- Metrics are sent to CloudWatch for many AWS services
- EC2 metrics are sent every **5 minutes** by default (free)
- Detailed EC2 monitoring sends every **1 minute** (chargeable)
- Unified CloudWatch Agent sends system-level metrics for EC2 and on-premises servers
- System-level metrics include memory and disk usage





Amazon CloudWatch Metrics

- Can publish custom metrics using CLI or API
- Custom metrics are one of the following resolutions:
 - **Standard resolution** – data having a one-minute granularity
 - **High resolution** – data at a granularity of one second
- AWS metrics are standard resolution by default

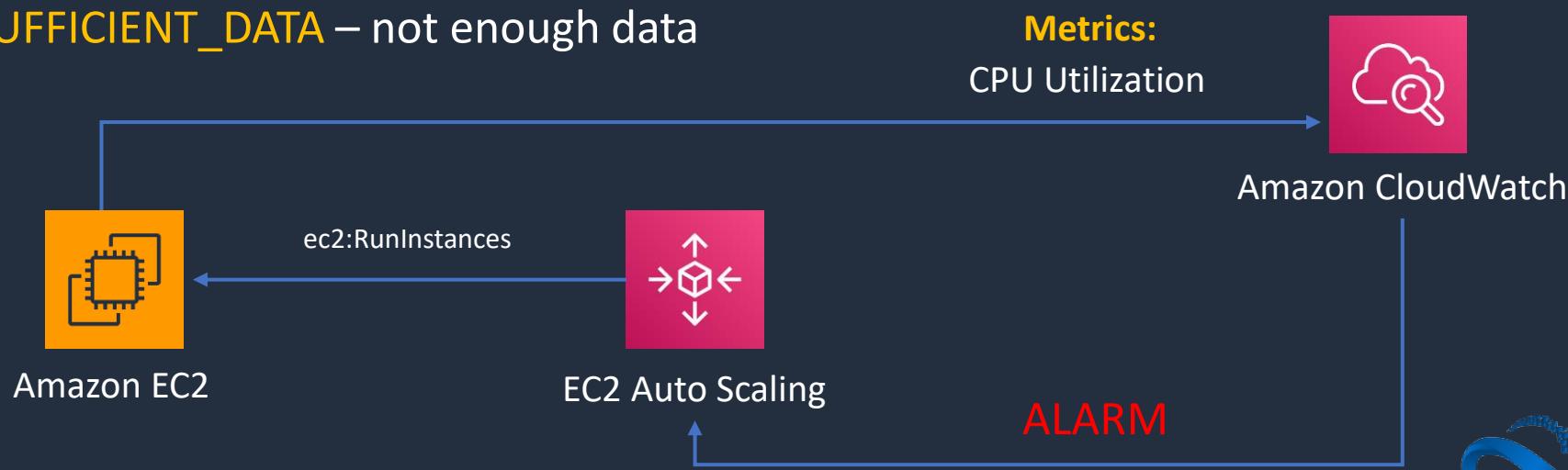




Amazon CloudWatch Alarms

Two types of alarms

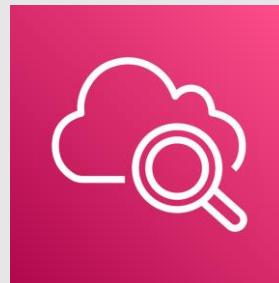
- **Metric alarm** – performs one or more actions based on a single metric
- **Composite alarm** – uses a rule expression and takes into account multiple alarms
- Metric alarm states:
 - **OK** – Metric is within a threshold
 - **ALARM** – Metric is outside a threshold
 - **INSUFFICIENT_DATA** – not enough data



Create a Custom Metric and Alarm



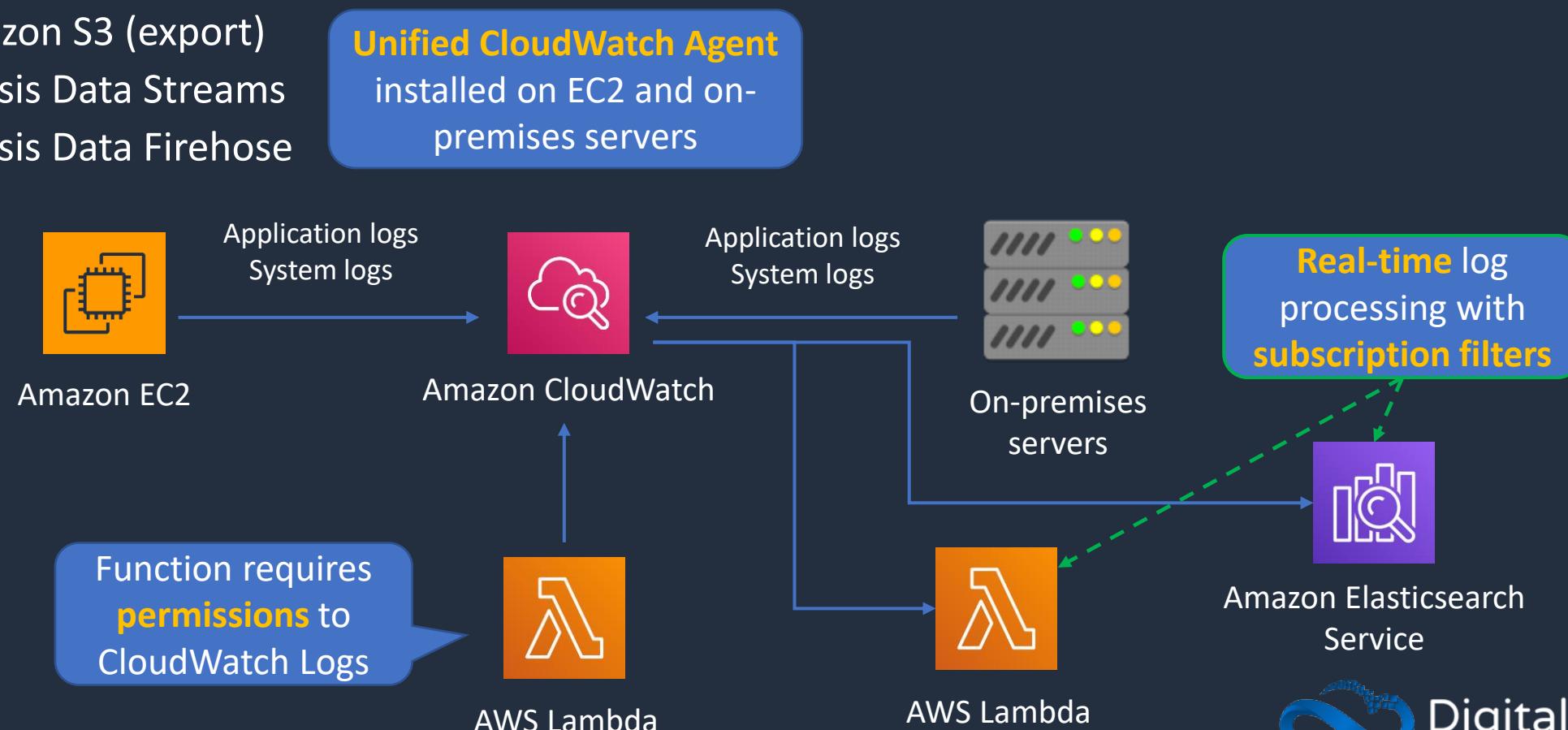
Amazon CloudWatch Logs





Amazon CloudWatch Logs

- Gather application and system logs in CloudWatch
- Defined expiration policies and KMS encryption
- Send to:
 - Amazon S3 (export)
 - Kinesis Data Streams
 - Kinesis Data Firehose



The Unified CloudWatch Agent





The Unified CloudWatch Agent

The unified CloudWatch agent enables you to do the following:

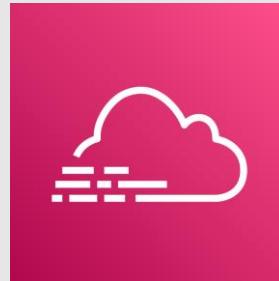
- Collect internal system-level metrics from Amazon **EC2 instances** across operating systems
- Collect system-level metrics from **on-premises servers**
- Retrieve custom metrics from your applications or services using the StatsD and collectd protocols
- Collect logs from Amazon EC2 instances and on-premises servers (Windows / Linux)



The Unified CloudWatch Agent

- Agent must be installed on the server
- Can be installed on:
 - Amazon EC2 instances
 - On-premises servers
 - Linux, Windows Server, or macOS

AWS CloudTrail



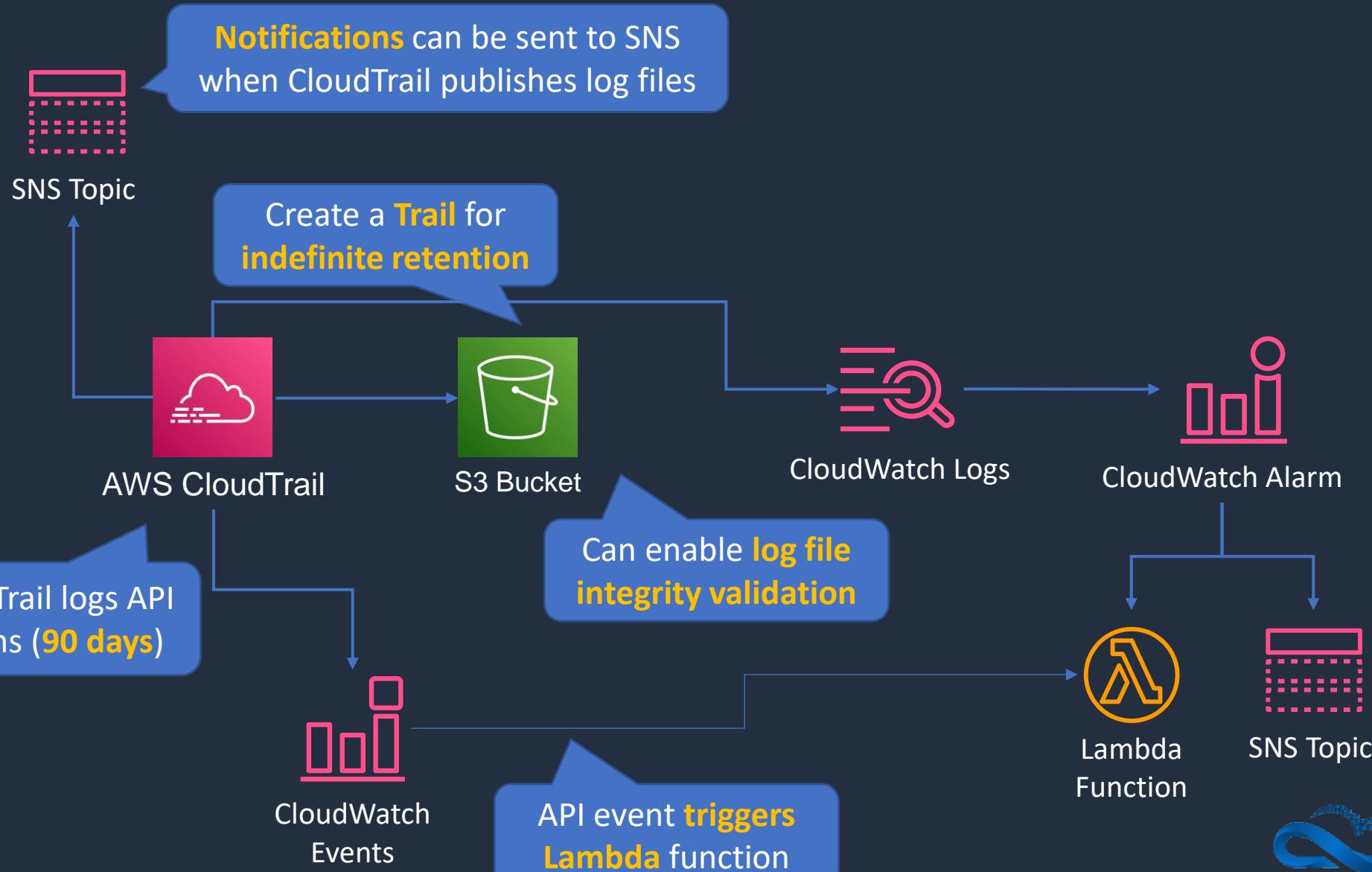


AWS CloudTrail

- CloudTrail logs **API activity** for auditing
- By default, management events are logged and retained for 90 days
- A **CloudTrail Trail** logs any events to S3 for indefinite retention
- Trail can be within Region or all Regions
- CloudWatch Events can be triggered based on API calls in CloudTrail
- Events can be streamed to CloudWatch Logs



AWS CloudTrail





CloudTrail – Types of Events

- **Management events** provide information about management operations that are performed on resources in your AWS account
- **Data events** provide information about the resource operations performed on or in a resource
- **Insights events** identify and respond to unusual activity associated with write API calls by continuously analyzing CloudTrail management events

Create a Trail in AWS CloudTrail

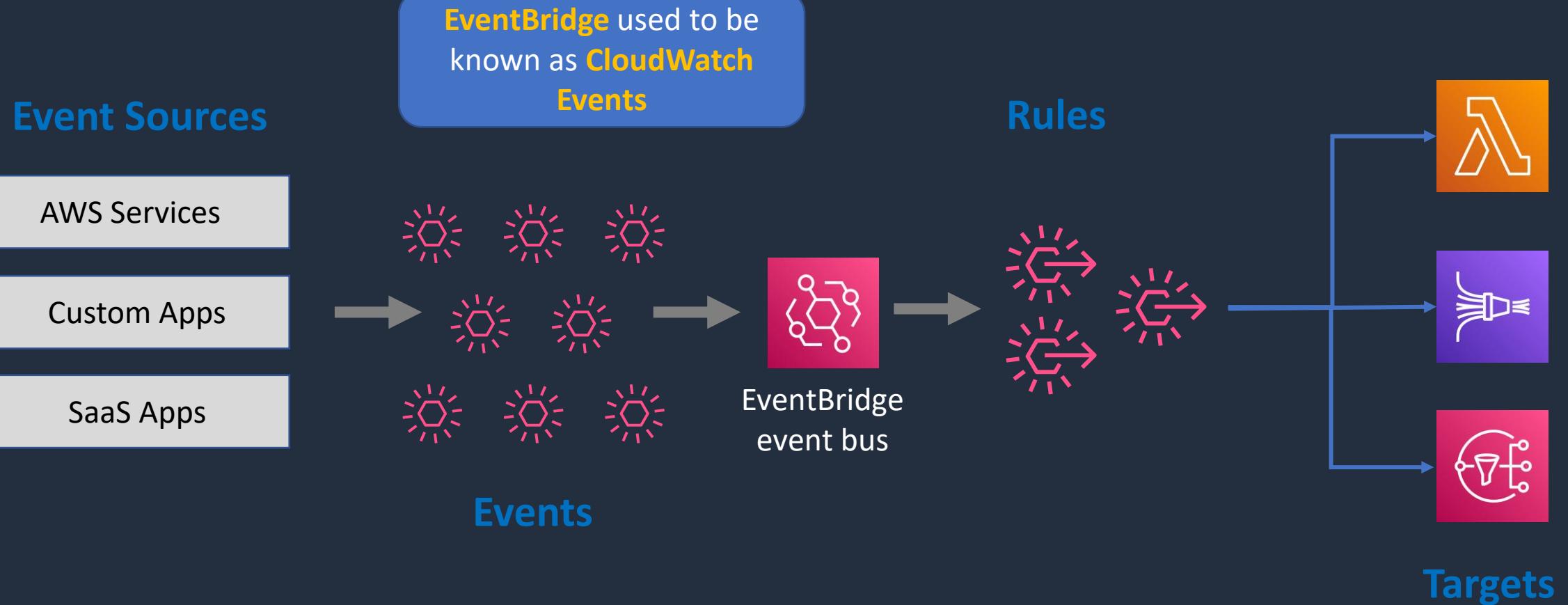


Amazon EventBridge (Refresher)





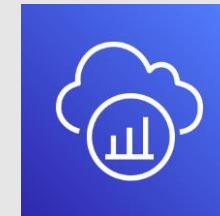
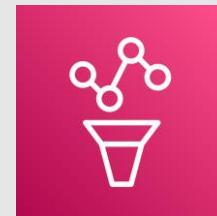
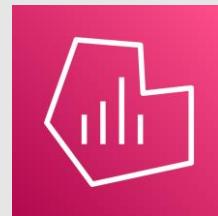
Amazon CloudWatch Events / EventBridge



Create EventBridge rule for CloudTrail API calls



Metric Analysis and Tracing (Grafana, Prometheus, X-Ray)





Tracing with AWS X-Ray

- You can use AWS X-Ray to visualize the components of your application, identify performance bottlenecks, and troubleshoot requests that resulted in an error
- AWS services send trace data to X-Ray, and X-Ray processes the data to generate a service map and searchable trace summaries





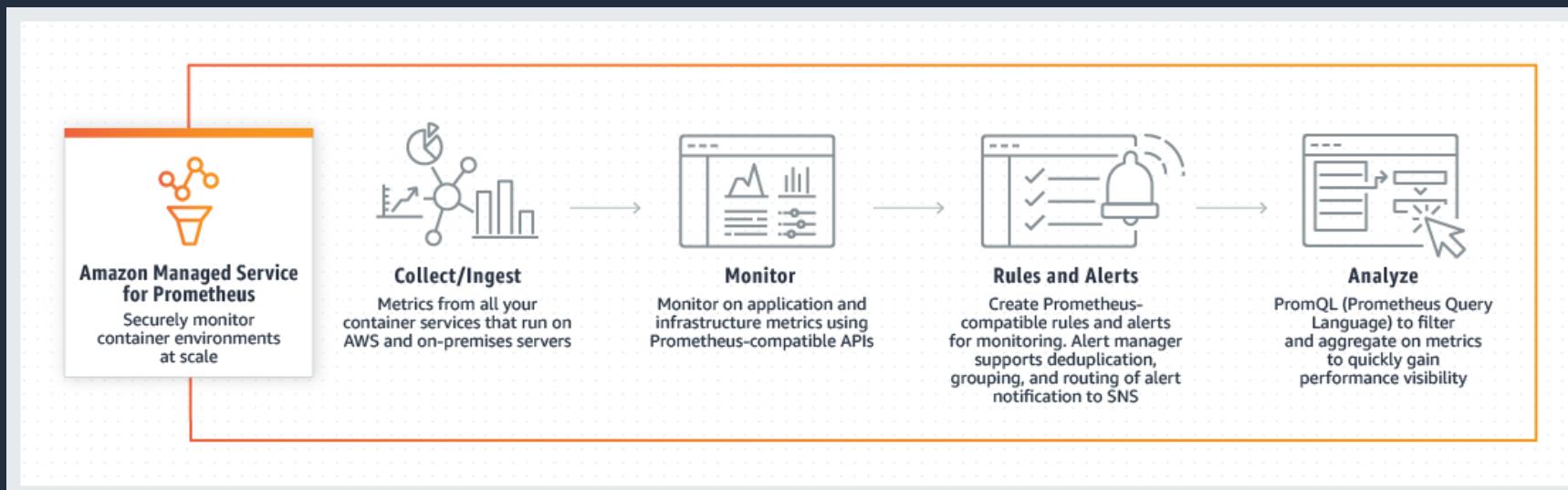
Tracing with AWS X-Ray

- X-Ray can be used with applications running on EC2, ECS, Lambda, and Elastic Beanstalk
- You must integrate the X-Ray SDK with your application and install the X-Ray agent
- The AWS X-Ray agent is a software application that gathers raw segment data and relays it to the AWS X-Ray service
- The agent works in conjunction with the AWS X-Ray SDKs so that data sent by the SDKs can reach the X-Ray service
- The X-Ray SDK captures metadata for requests made to MySQL and PostgreSQL databases and Amazon DynamoDB
- It also captures metadata for requests made to Amazon SQS and Amazon SNS



Amazon Managed Service for Prometheus

- Prometheus is an open-source monitoring system and time series database
- Use the open-source Prometheus query language (PromQL) to monitor and alert on the performance of containerized workloads
- Automatically scales the ingestion, storage, alerting, and querying of operational metrics as workloads grow or shrink
- Integrated with Amazon EKS, Amazon ECS, and AWS Distro for OpenTelemetry

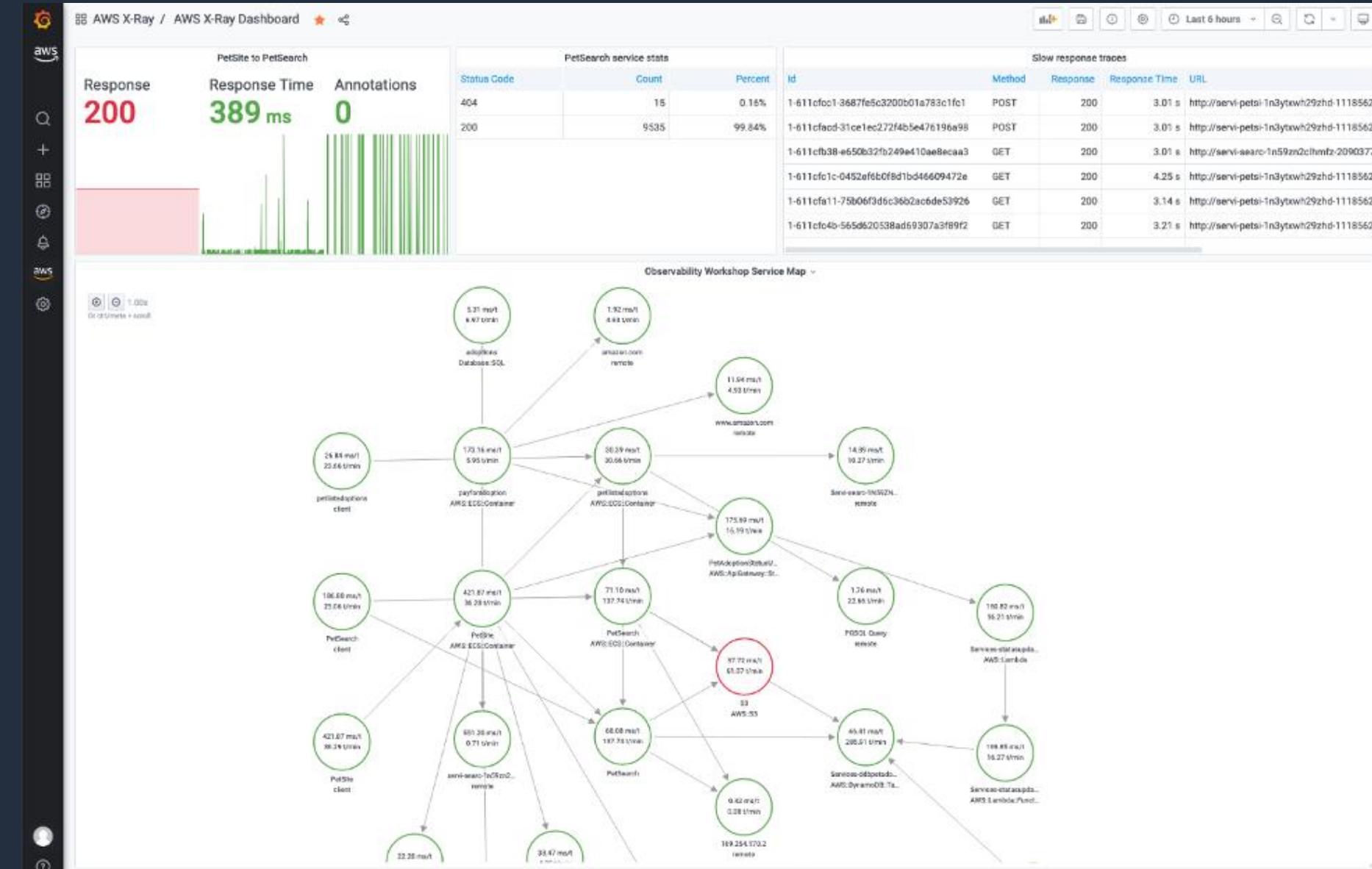




Amazon Managed Grafana

- Grafana is an open-source analytics and monitoring solution for databases
- Highly scalable, highly available, and fully managed service Grafana
- Provides interactive data visualization for your monitoring and operational data
- Visualize, analyze, and alarm on your metrics, logs, and traces collected from multiple data sources
- Integrates with AWS SSO and SAML

Amazon Managed Grafana



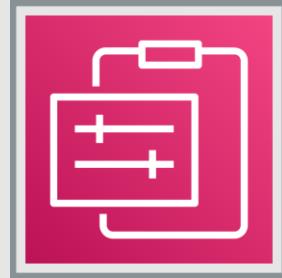
Example of an Amazon Managed Grafana dashboard visualizing data from AWS X-Ray as a data source

Amazon Managed Grafana



Example visualizing
an imported MySQL
dashboard

Architecture Patterns - Monitoring, Logging and Auditing





Architecture Patterns – Monitoring, Logging and Auditing

Requirement

Need to stream logs from Amazon EC2 instances in an Auto Scaling Group

Solution

Install the unified CloudWatch Agent and collect log files in Amazon CloudWatch

Need to collect metrics from EC2 instances with a 1 second granularity

Create a custom metric with high resolution

The application logs from on-premises servers must be processed by AWS Lambda in real time

Install the unified CloudWatch Agent on the servers and use a subscription filter in CloudWatch to connect to a Lambda function



Architecture Patterns – Monitoring, Logging and Auditing

Requirement

CloudWatch Logs entries must be transformed with Lambda and then loaded into Amazon S3

Access auditing must be enabled, and records must be stored for a minimum of 5 years. Any attempts to modify the log files must be identified

API activity must be captured from all accounts in an Organization. Admins in member accounts must not be able to modify or delete the trail

Solution

Configure a Kinesis Firehose destination, transform with Lambda and then load into an S3 bucket

Create a trail in CloudTrail that stores the data in an S3 bucket and enable log file integrity validation

Create an Organization trail in AWS CloudTrail that applies to the management account and all member accounts



Architecture Patterns – Monitoring, Logging and Auditing

Requirement

Company requires API events that involve the root user account to generate a notification

Solution

Create a CloudTrail trail and an EventBridge rule that looks for API events that involve root and configure an SNS notification

For compliance reasons all S3 buckets must have encryption enabled and any non-compliant buckets must be auto remediated

Use AWS Config to check the encryption status of the buckets and use auto remediation to enable encryption as required

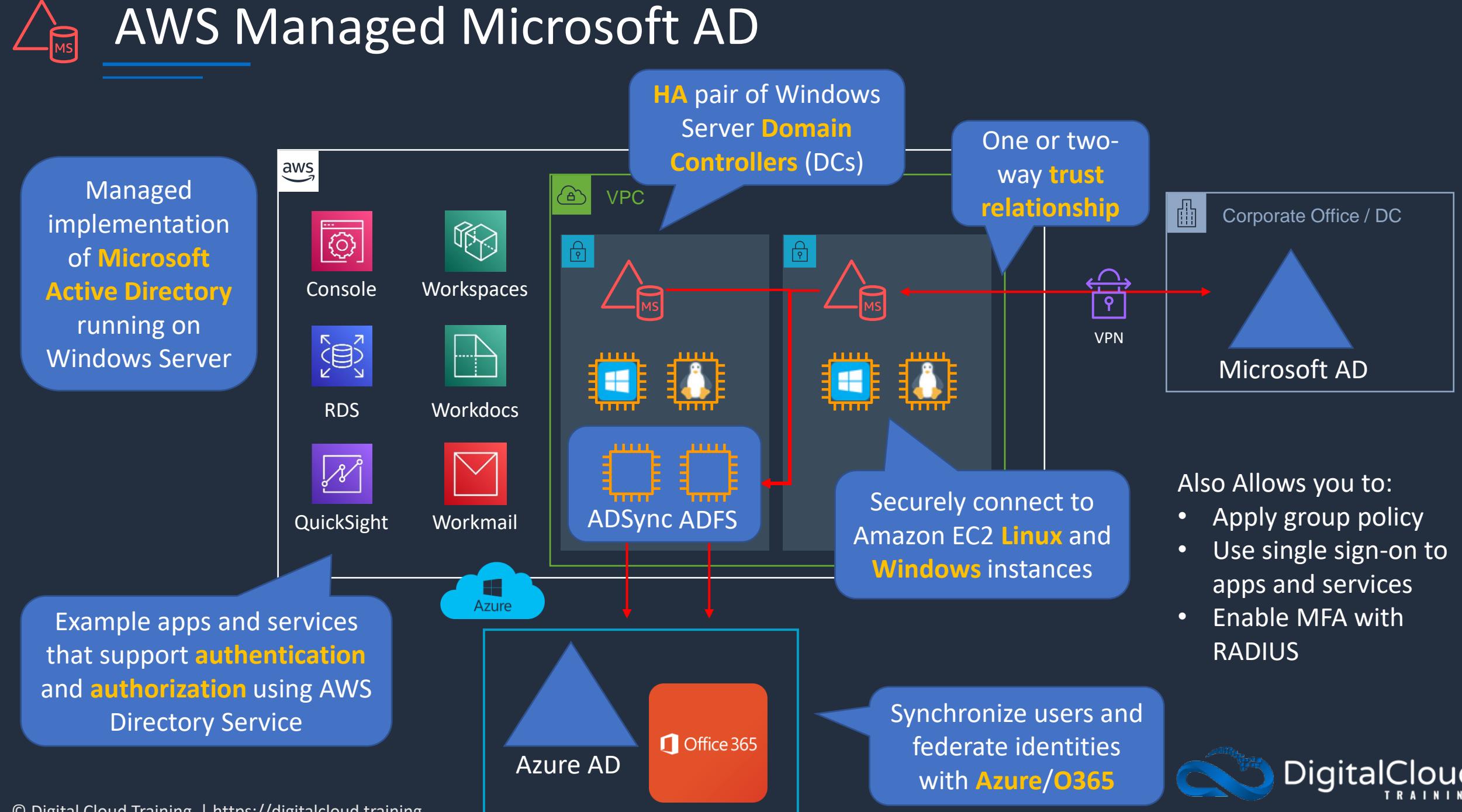
SECTION 15

Security in the Cloud

AWS Directory Service



AWS Managed Microsoft AD

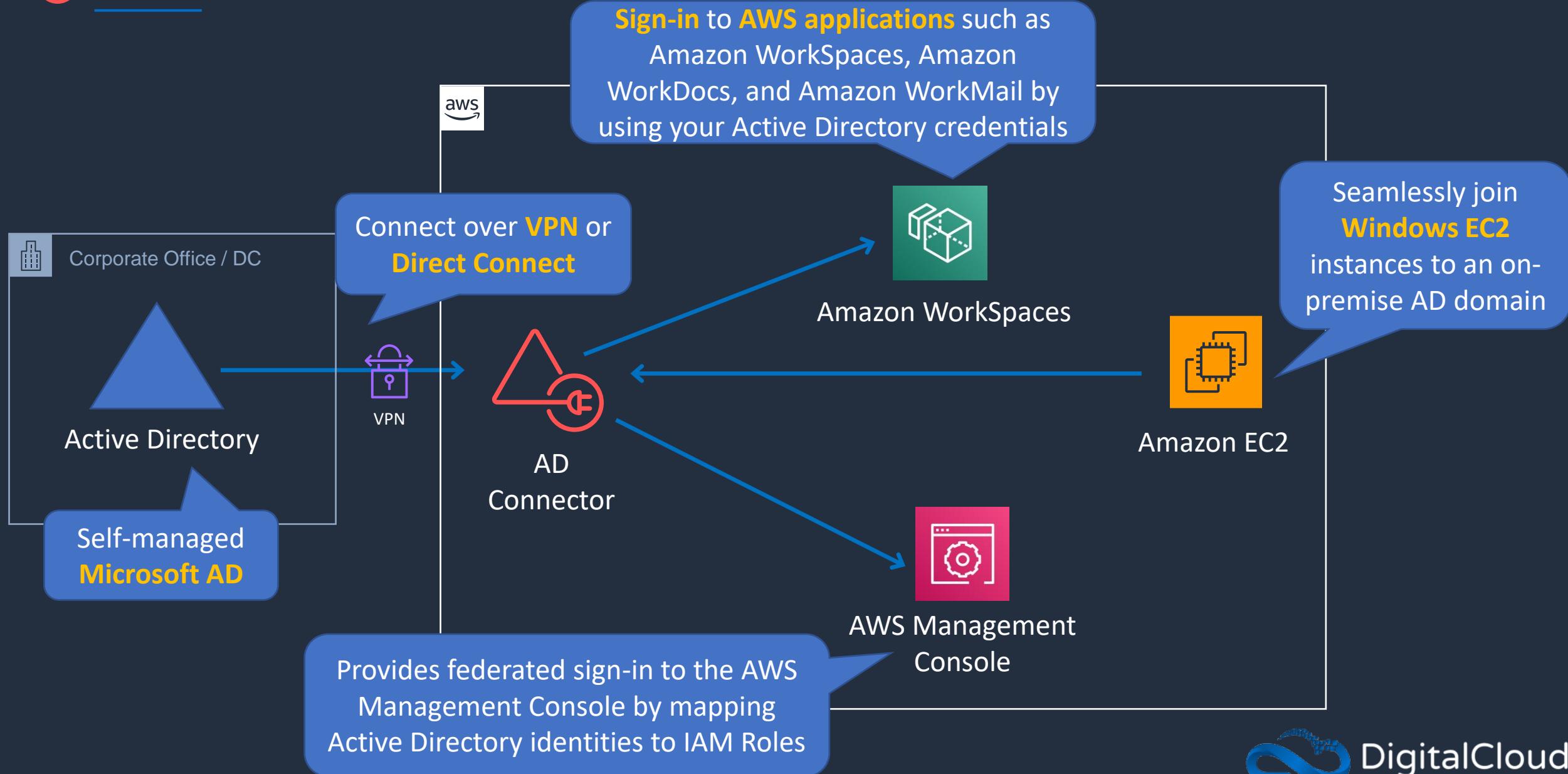




AWS Managed Microsoft AD

- Fully managed AWS service
- Best choice if you have more than 5000 users and/or need a trust relationship set up
- Can perform schema extensions
- Can setup trust relationships to with on-premises Active Directories:
 - On-premise users and groups can access resources in either domain using SSO
 - Requires a VPN or Direct Connect connection
- Can be used as a standalone AD in the AWS cloud

AD Connector





AD Connector

- Redirects directory requests to your on-premise Active Directory
- Best choice when you want to use an existing Active Directory with AWS services
- AD Connector comes in two sizes:
 - Small – designed for organizations up to 500 users
 - Large – designed for organizations up to 5000 users
- Requires a VPN or Direct Connect connection
- Join EC2 instances to your on-premise AD through AD Connector
- Login to the AWS Management Console using your on-premise AD DCs for authentication



Simple AD

- Inexpensive Active Directory-compatible service with common directory features
- Standalone, fully managed, directory on the AWS cloud
- Simple AD is generally the least expensive option
- Best choice for less than 5000 users and don't need advanced AD features
- Features include:
 - Manage user accounts / groups
 - Apply group policies
 - Kerberos-based SSO
 - Supports joining Linux or Windows based EC2 instances

Identity Providers and Federation

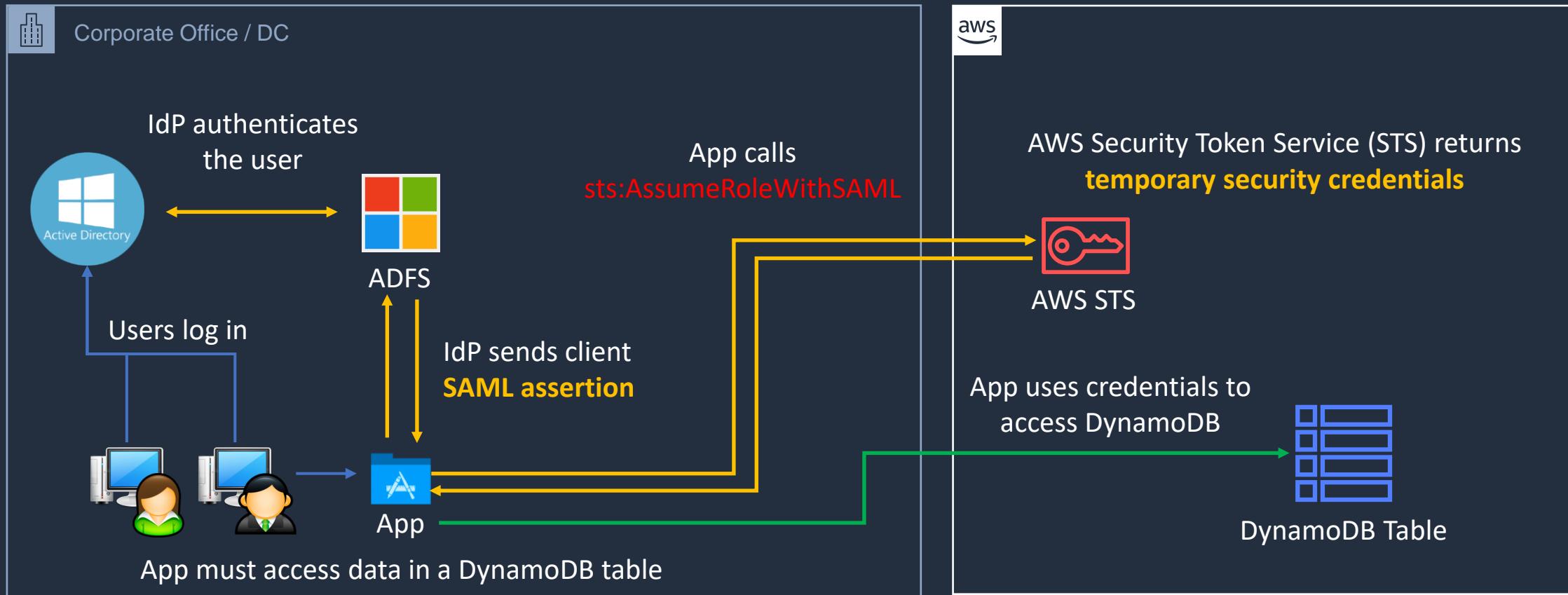




IAM – SAML 2.0 Identity Federation

Active Directory is an
LDAP **Identity Store**

Active Directory Federation Services
is an **Identity Provider** (IdP)





IAM – Web Identity Federation

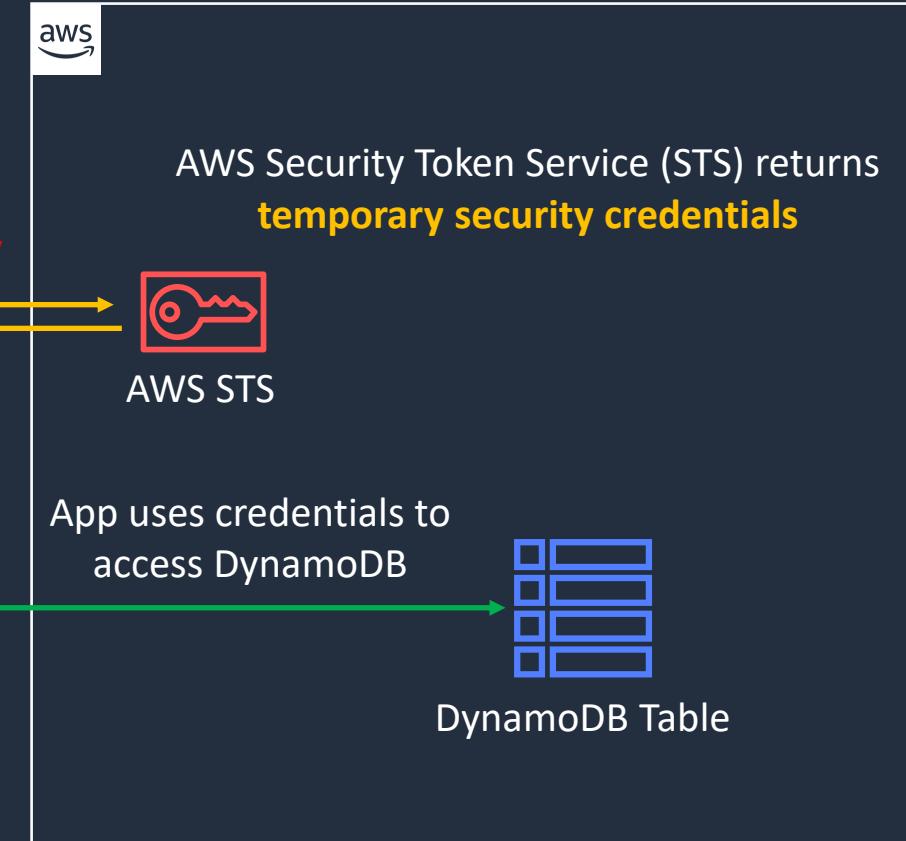
Any **Open ID Connect (OIDC)**
compatible IdP supported

Social identity providers (IdPs)



Mobile App

App calls
`sts:AssumeRoleWithWebIdentity`



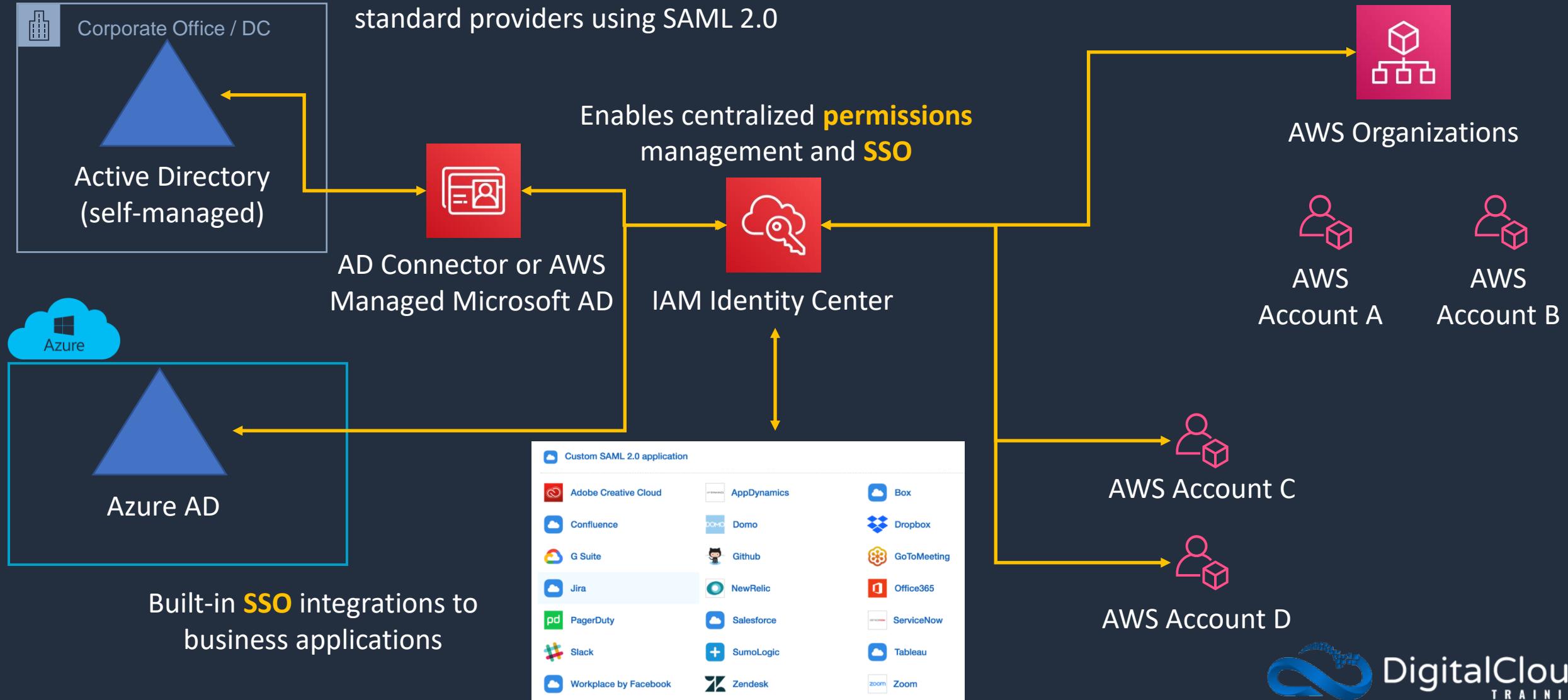
AWS recommend to use **Cognito** for **web identity federation** in most cases



IAM Identity Center

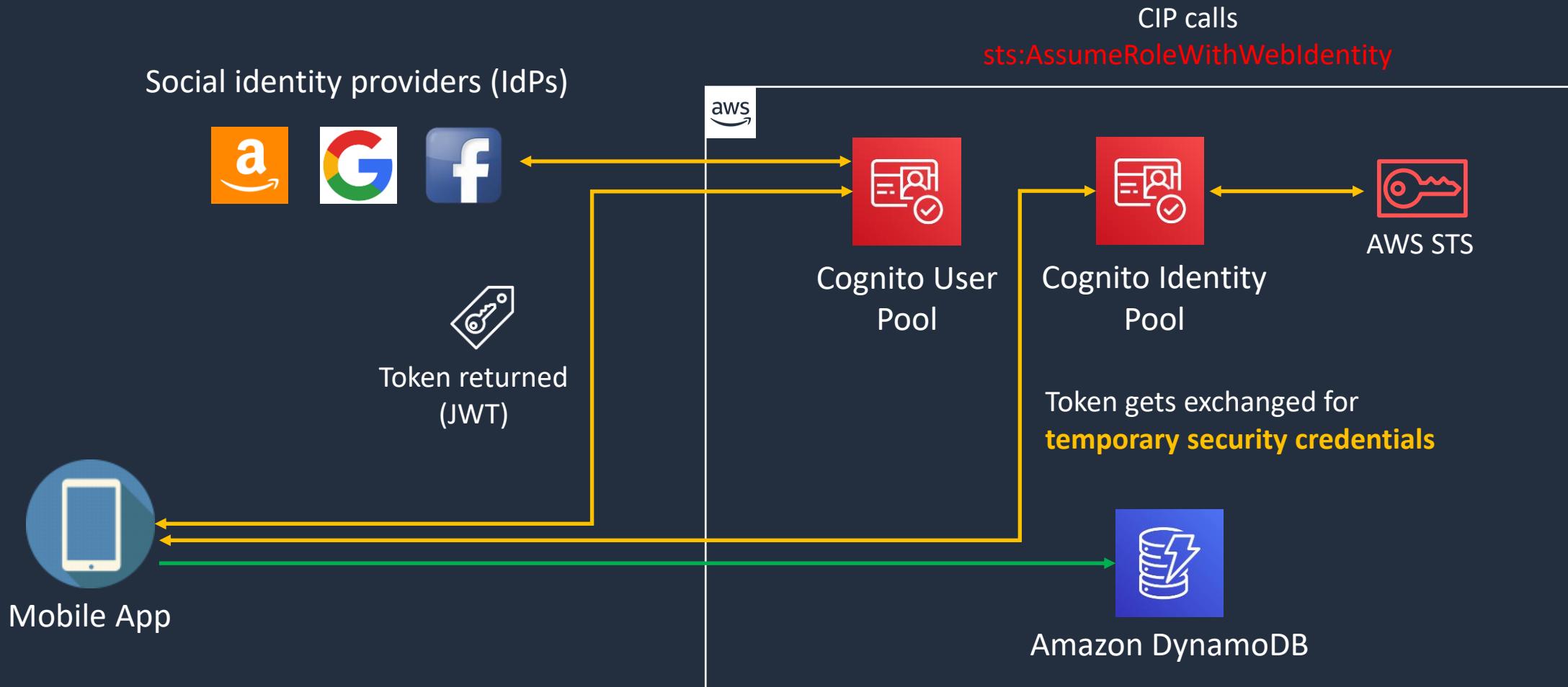
Identity sources can be Identity Center directory, Active Directory and standard providers using SAML 2.0

IAM Identity Center is the successor to **AWS Single Sign-On (SSO)**

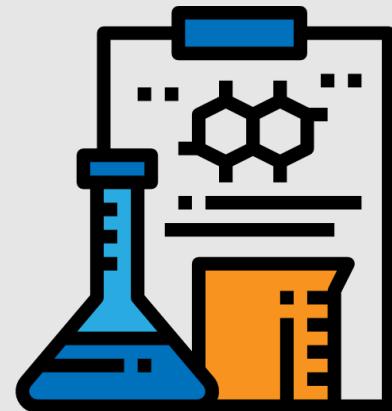




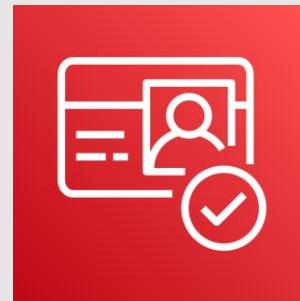
Amazon Cognito



IAM Identity Center in Action



Amazon Cognito

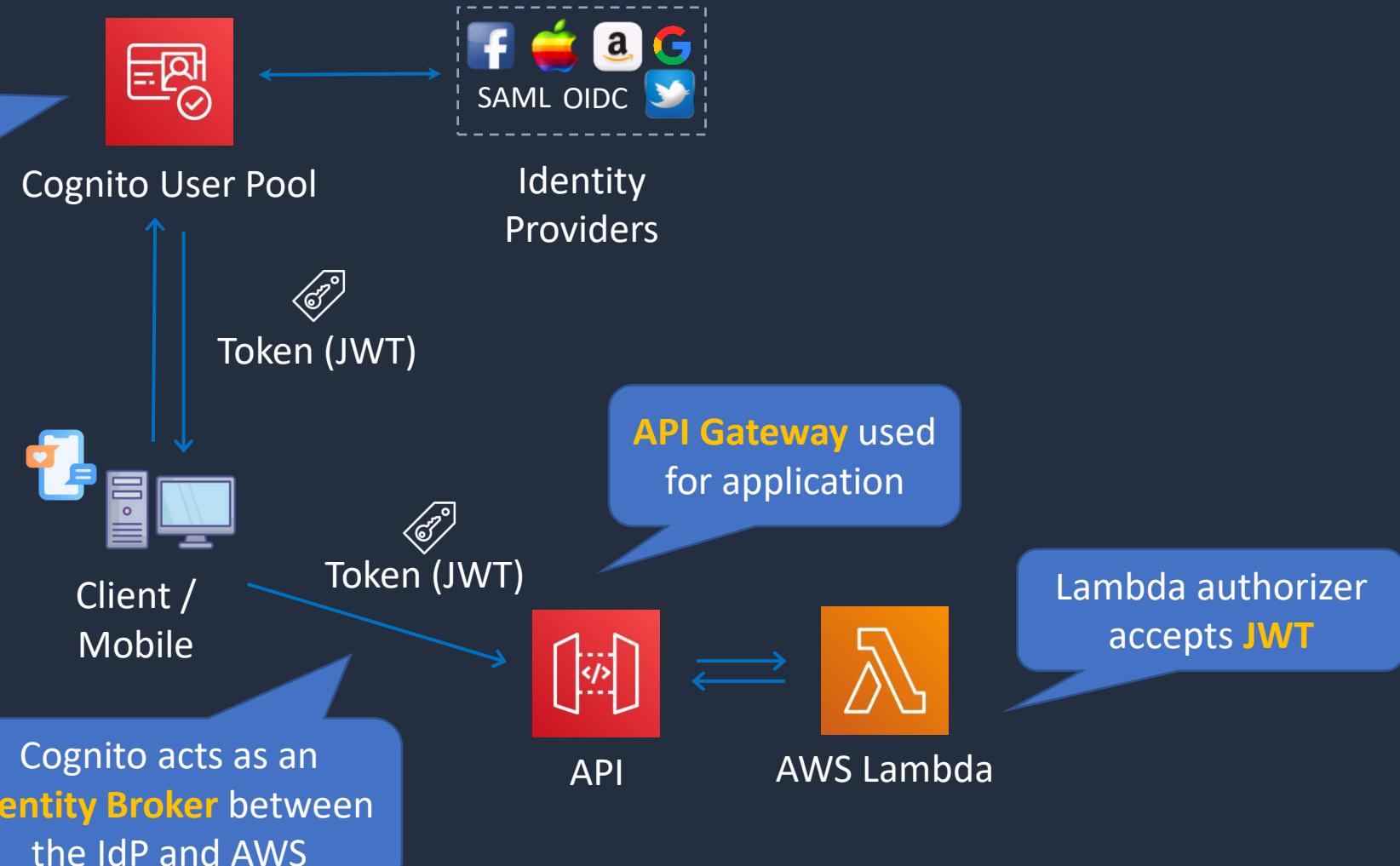




Cognito User Pools

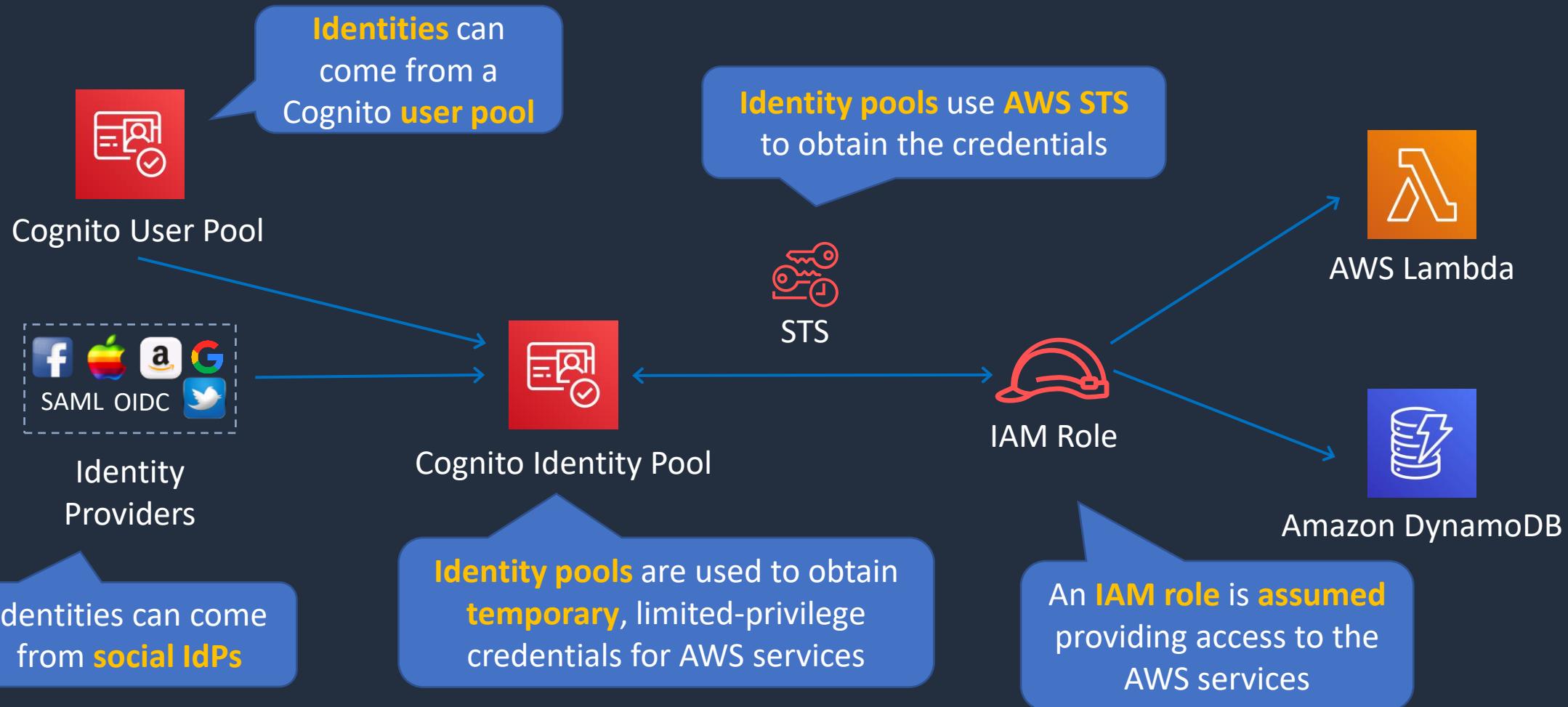
A **User Pool** is a directory for managing **sign-in** and **sign-up** for mobile applications

Users can also sign in using **social IdPs**

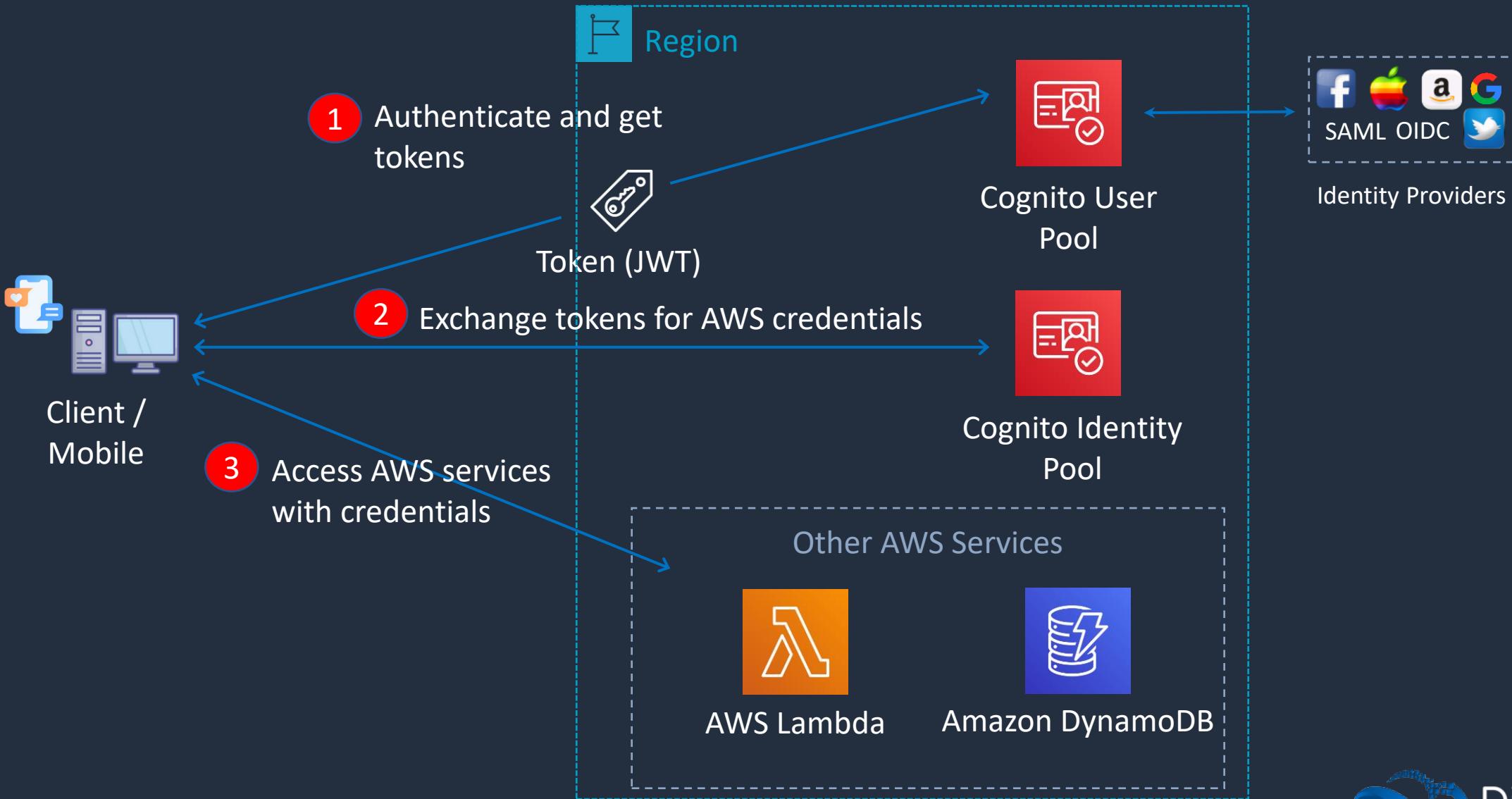




Cognito Identity Pool



User Pools + Identity Pools



Encryption Primer





Encryption In Transit vs At Rest



User

Encryption In Transit

HTTPS Connection

Data is protected by
SSL/TLS in transit



ALB

Encryption At Rest

Amazon S3 **encrypts** the object as it is **written** to the bucket it



Unencrypted
Object



Data encryption key



Encryption process

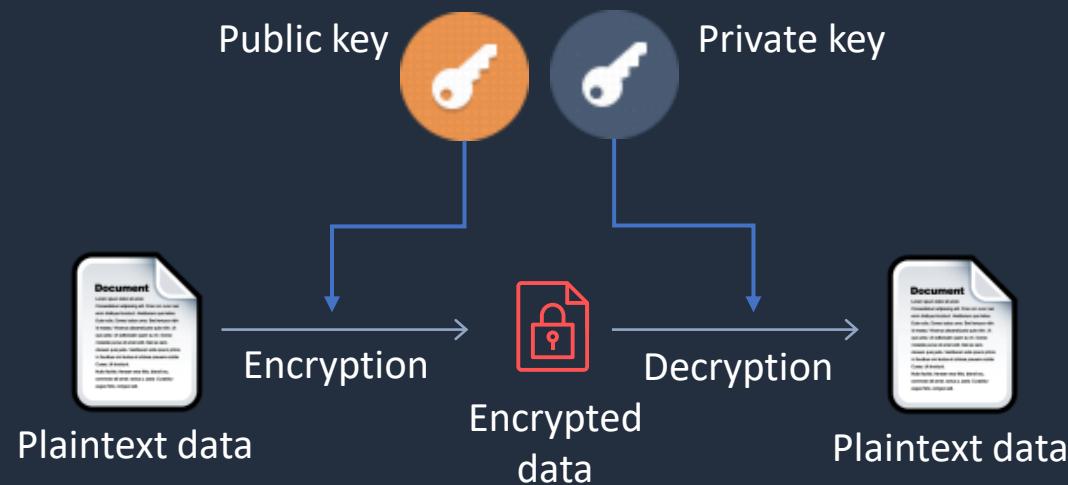


Encrypted
bucket



Asymmetric Encryption

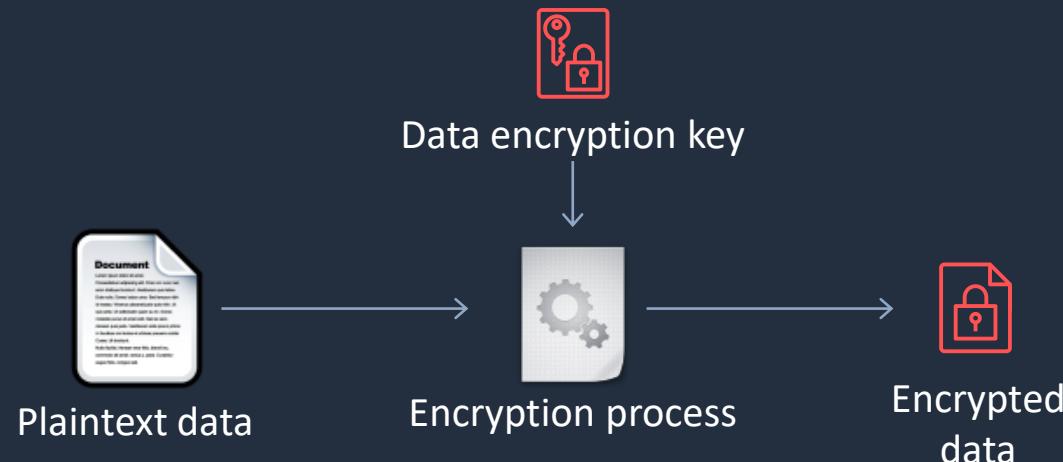
- Asymmetric encryption is also known as public key cryptography
- Messages encrypted with the public key can only be decrypted with the private key
- Messages encrypted with the private key can be decrypted with the public key
- Examples include SSL/TLS and SSH



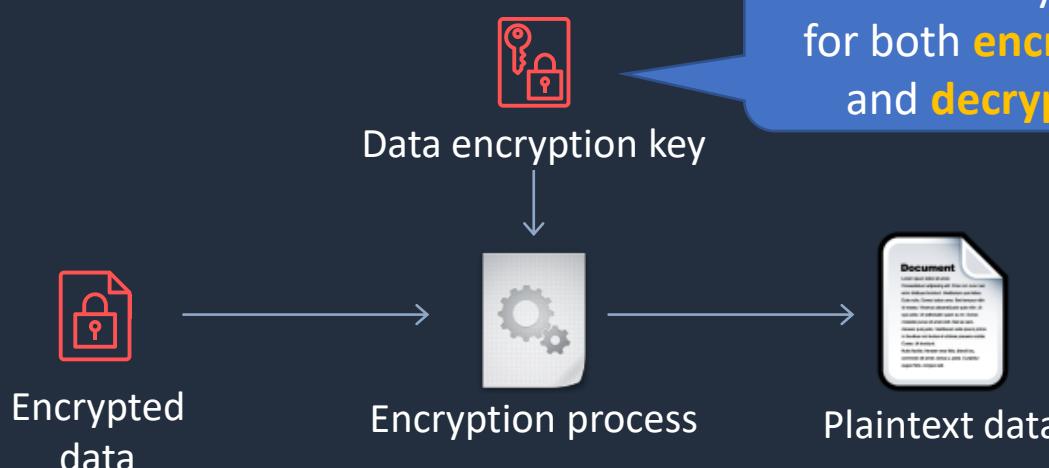


Symmetric Encryption

Encryption



Decryption



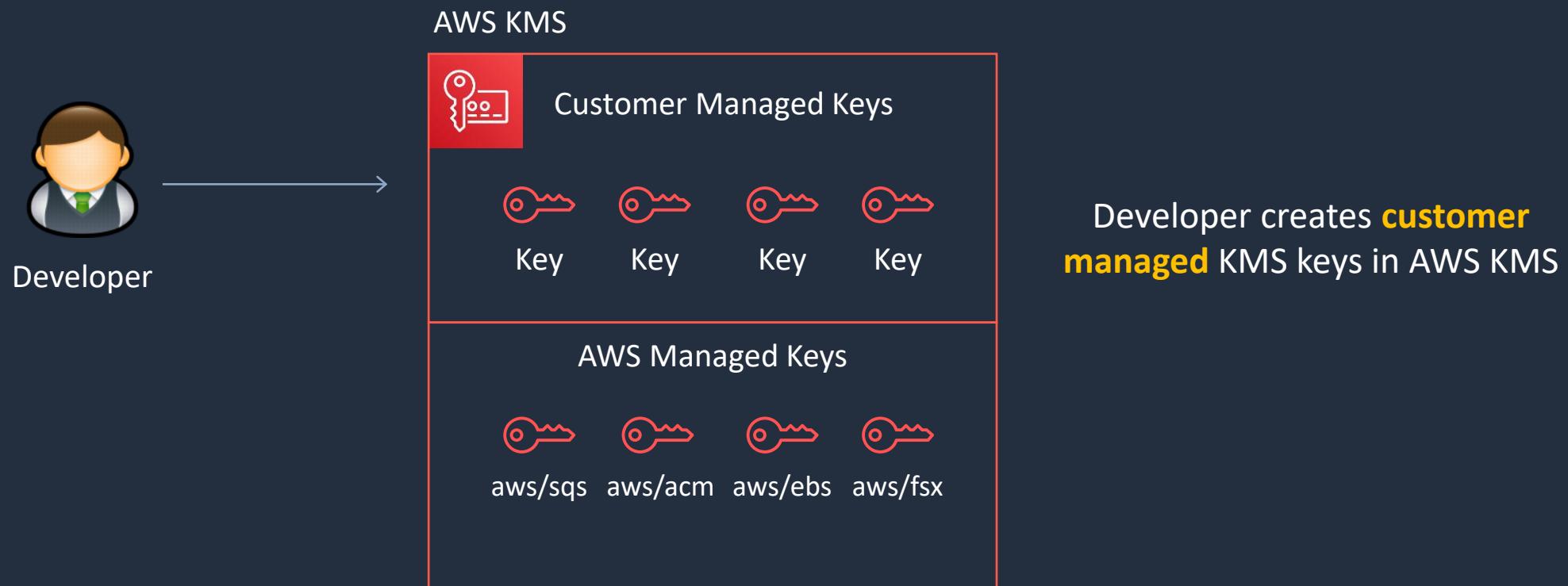
AWS Key Management Service (KMS)





AWS Key Management Service (KMS)

- Create and managed **symmetric** and **asymmetric** encryption keys
- The **KMS keys** are protected by hardware security modules (HSMs)





KMS Keys

- KMS keys are the primary resources in AWS KMS
- Used to be known as “customer master keys” or CMKs
- The KMS key also contains the key material used to encrypt and decrypt data
- By default, AWS KMS creates the key material for a KMS key
- You can also import your own key material
- A KMS key can encrypt data up to 4KB in size
- A KMS key can generate, encrypt and decrypt Data Encryption Keys (DEKs)





Alternative Key Stores

External Key Store

- Keys can be stored outside of AWS to meet regulatory requirements
- You can create a KMS key in an AWS KMS external key store (XKS)
- All keys are generated and stored in an external key manager
- When using an XKS, key material never leaves your HSM

Custom Key Store

- You can create KMS keys in an AWS CloudHSM custom key store
- All keys are generated and stored in an AWS CloudHSM cluster that you own and manage
- Cryptographic operations are performed solely in the AWS CloudHSM cluster you own and manage
- Custom key stores are not available for asymmetric KMS keys



AWS Managed KMS Keys

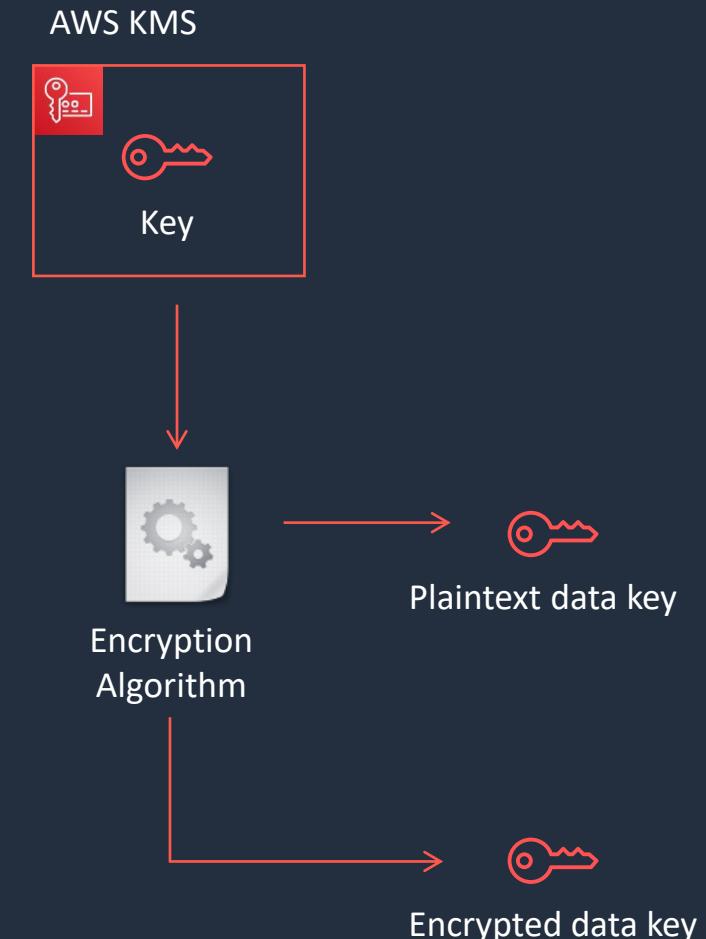
- Created, managed, and used on your behalf by an AWS service that is integrated with AWS KMS
- You cannot manage these KMS keys, rotate them, or change their key policies
- You also cannot use AWS managed KMS keys in cryptographic operations directly; the service that creates them uses them on your behalf

Alias	Key ID
aws/sqs	025b9386-b1f8-4fa9-84e2-ac3220b1de59
aws/acm	2d604e85-c2d4-42dc-ab0b-0b356f5fe26e
aws/codecommit	41fea9df-e447-4992-8af7-6ddec6d81175
aws/elasticfilesystem	460c4f05-fe98-4a35-b940-3e1992f04314
aws/glue	617516fe-bf19-4da4-a743-2b13c41973e1
aws/lambda	7f513d01-784b-41b9-9c51-51621db7b5e1
aws/ebs	b9baa4f6-3e87-4256-af6a-d181940df286
aws/lightsail	bc7ba666-8e17-444a-800c-d3d4be303a97
aws/fsx	cebc434c-ee2b-4a61-9b5a-f63be9fdb068
aws/kinesis	d99014b5-09d4-480d-9b2a-3a7d7e3e9c5b



Data Encryption Keys

- Data keys are encryption keys that you can use to encrypt large amounts of data
- You can use AWS KMS keys to generate, encrypt, and decrypt data keys
- AWS KMS does not store, manage, or track your data keys, or perform cryptographic operations with data keys
- You must use and manage data keys outside of AWS KMS





KMS Keys and Automatic Rotation

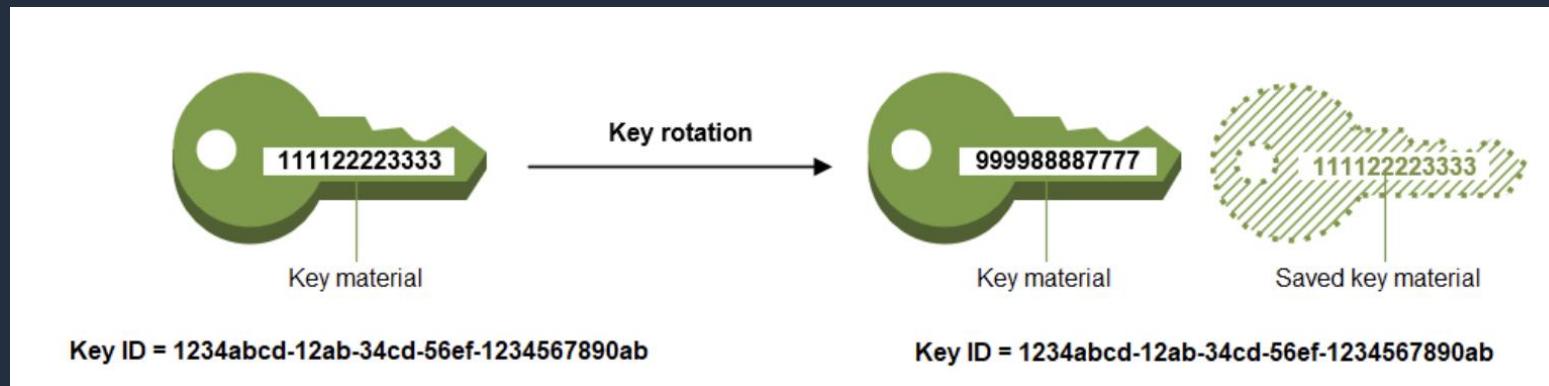
Type of KMS Key	Can view	Can manage	Used only for my AWS account	Automatic rotation
Customer managed key	Yes	Yes	Yes	Optional. Every 365 days
AWS managed key	Yes	No	Yes	Required. Every 365 days
AWS owned key	No	No	No	Varies

- You cannot enable or disable key rotation for AWS owned keys
- Automatic key rotation is supported only on symmetric encryption KMS keys with key material that AWS KMS generates (**Origin = AWS_KMS**)



KMS Keys and Automatic Rotation

- Automatic rotation generates new key material every year
(optional for customer managed keys)



Rotation only changes the **key material** used for encryption, the KMS key remains the same



KMS Keys and Automatic Rotation

With automatic key rotation:

- The properties of the KMS key, including its key ID, key ARN, region, policies, and permissions, do not change when the key is rotated
- You do not need to change applications or aliases that refer to the key ID or key ARN of the KMS key
- After you enable key rotation, AWS KMS rotates the KMS key automatically every year

Automatic key rotation is not supported on the following types of KMS keys:

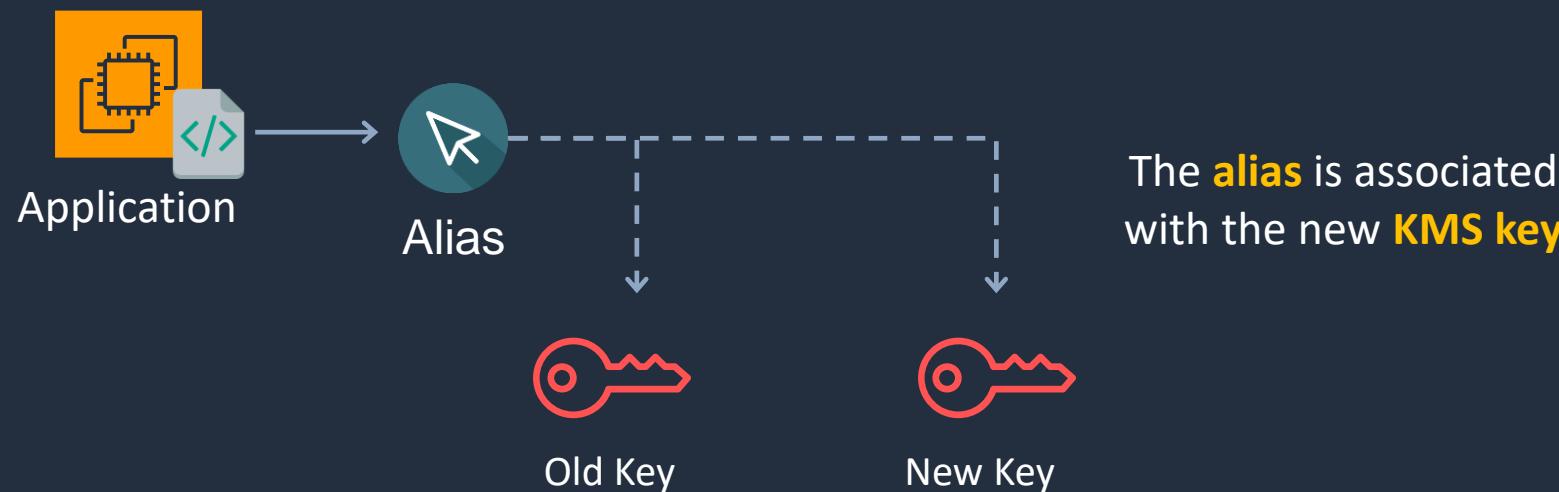
- Asymmetric KMS keys
- HMAC KMS keys
- KMS keys in custom key stores
- KMS keys with imported key material

Note: You can rotate these KMS keys **manually**



Manual Rotation

- Manual rotation is creating a new KMS key with a different key ID
- You must then update your applications with the new key ID
- You can use an **alias** to represent a KMS key so you don't need to modify your application code





KMS Key Policies

- Key policies define management and usage permissions for KMS keys

```
{  
    "Sid": "Allow access for Key Administrators",  
    "Effect": "Allow",  
    "Principal": {"AWS": "arn:aws:iam::111122223333:user/KMSKeyAdmin"},  
    "Action": [  
        "kms:Describe*",  
        "kms:Put*",  
        "kms>Create*",  
        "kms:Update*",  
        "kms:Enable*",  
        "kms:Revoke*",  
        "kms>List*",  
        "kms:Disable*",  
        "kms:Get*",  
        "kms>Delete*",  
        "kms:ScheduleKeyDeletion",  
        "kms:CancelKeyDeletion"  
    ],  
    "Resource": "*"
```

This key policy defines the **administrative actions** that are permitted for a key administrator



KMS Key Policies

- Multiple policy statements can be combined to specify separate administrative and usage permissions

```
{  
  "Sid": "Allow use of the key",  
  "Effect": "Allow",  
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/EncryptionApp"},  
  "Action": [  
    "kms:DescribeKey",  
    "kms:GenerateDataKey*",  
    "kms:Encrypt",  
    "kms:ReEncrypt*",  
    "kms:Decrypt"  
,  
  "Resource": "*"  
}
```

This key policy defines the **cryptographic** actions for encrypting and decrypting data with the KMS key



KMS Key Policies

- Permissions can be specified for delegating use of the key to AWS services

```
{  
  "Sid": "Allow attachment of persistent resources",  
  "Effect": "Allow",  
  "Principal": {"AWS": "arn:aws:iam::111122223333:role/EncryptionApp"},  
  "Action": [  
    "kms>ListGrants",  
    "kms>CreateGrant",  
    "kms>RevokeGrant"  
  ],  
  "Resource": "*",  
  "Condition": {"Bool": {"kms:GrantIsForAWSResource": true}}  
}
```

Grants are useful for **temporary permissions** as they can be used without modifying key policies or IAM policies



Additional Exam Tips

- To share snapshots with another account you must specify Decrypt and CreateGrant permissions
- The kms:ViaService condition key can be used to limit key usage to specific AWS services
- For example:

```
"Condition": {  
    "StringEquals": {  
        "kms:ViaService": [  
            "ec2.us-west-2.amazonaws.com",  
            "rds.us-west-2.amazonaws.com"  
        ]  
    }  
}
```



Additional Exam Tips

- Cryptographic erasure means removing the ability to decrypt data and can be achieved when using **imported key material** and deleting that key material
- You must use the **DeleteImportedKeyMaterial** API to remove the key material
- An **InvalidKeyId** exception when using SSM Parameter Store indicates the KMS key is not enabled
- Make sure you know the differences between AWS managed and customer managed KMS keys and automatic vs manual rotation

Encrypt and Decrypt Data with AWS KMS



AWS CloudHSM





AWS CloudHSM

- AWS CloudHSM is a cloud-based hardware security module (HSM)
- Generate and use your own encryption keys on the AWS Cloud
- CloudHSM runs in your Amazon VPC
- Uses FIPS 140-2 level 3 validated HSMs
- Managed service and automatically scales
- Retain control of your encryption keys - you control access (and AWS has no visibility of your encryption keys)



AWS CloudHSM Use Cases

- Offload SSL/TLS processing from web servers
- Protect private keys for an issuing certificate authority (CA)
- Store the master key for Oracle DB Transparent Data Encryption
- Custom key store for AWS KMS – retain control of the HSM that protects the master keys



AWS CloudHSM vs KMS

	CloudHSM	AWS KMS
Tenancy	Single-tenant HSM	Multi-tenant AWS service
Availability	Customer-managed durability and available	Highly available and durable key storage and management
Root of Trust	Customer managed root of trust	AWS managed root of trust
FIPS 140-2	Level 3	Level 2 / Level 3
3rd Party Support	Broad 3 rd Party Support	Broad AWS service support

AWS Certificate Manager (ACM)





AWS Certificate Manager (ACM)

- Create, store and renew SSL/TLS X.509 certificates
- Single domains, multiple domain names and wildcards
- Integrates with several AWS services including:
 - **Elastic Load Balancing**
 - **Amazon CloudFront**
 - **AWS Elastic Beanstalk**
 - **AWS Nitro Enclaves**
 - **AWS CloudFormation**



AWS Certificate Manager (ACM)

- **Public certificates** are signed by the AWS public Certificate Authority
- You can also create a Private CA with ACM
- Can then issue private certificates
- You can also import certificates from third-party issuers

SSL/TLS Certificate in ACM



AWS Web Application Firewall (WAF)



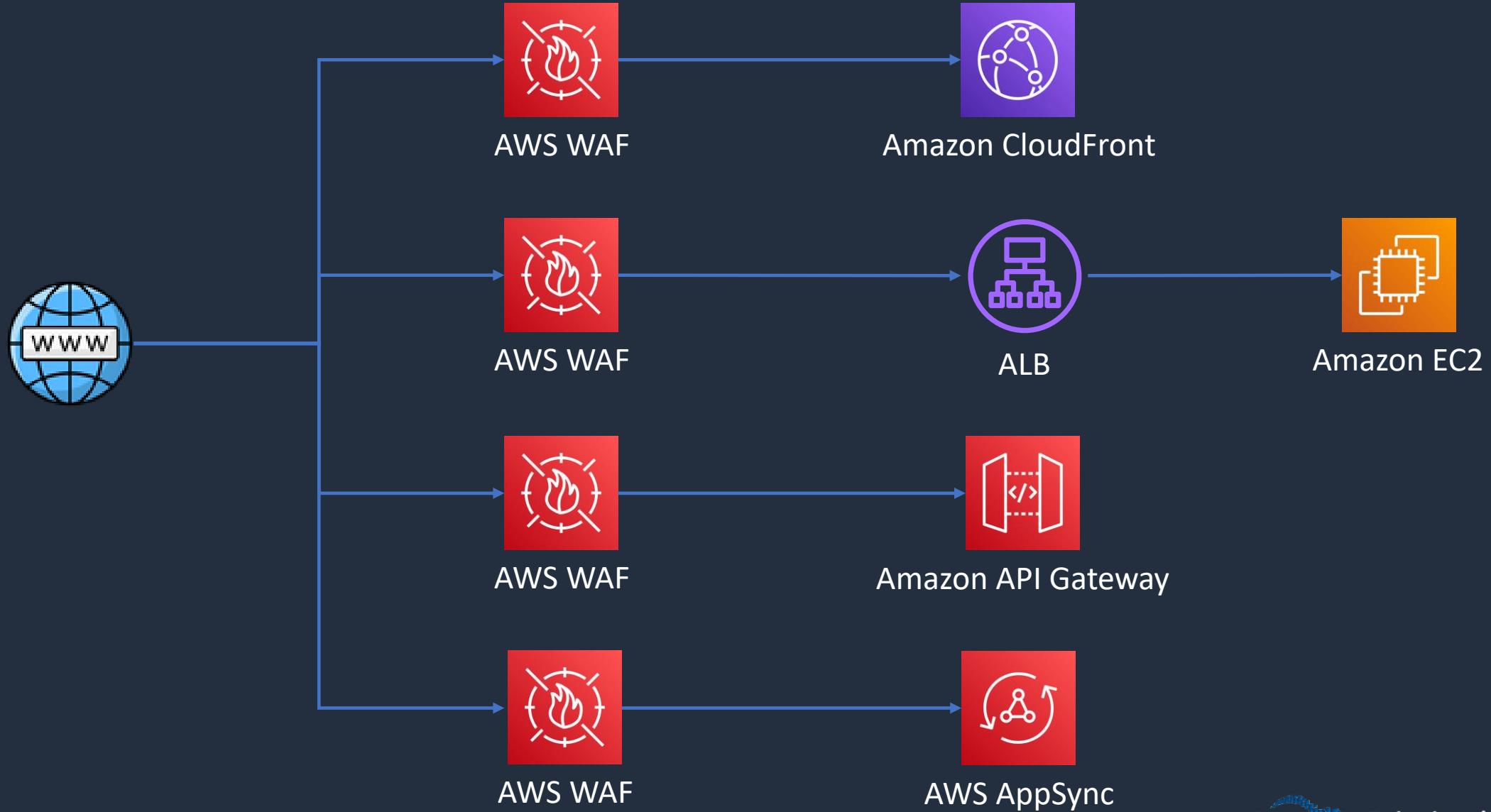


AWS WAF

- AWS WAF is a web application firewall
- WAF lets you create rules to filter web traffic based on conditions that include IP addresses, HTTP headers and body, or custom URIs
- WAF makes it easy to create rules that block common web exploits like SQL injection and cross site scripting



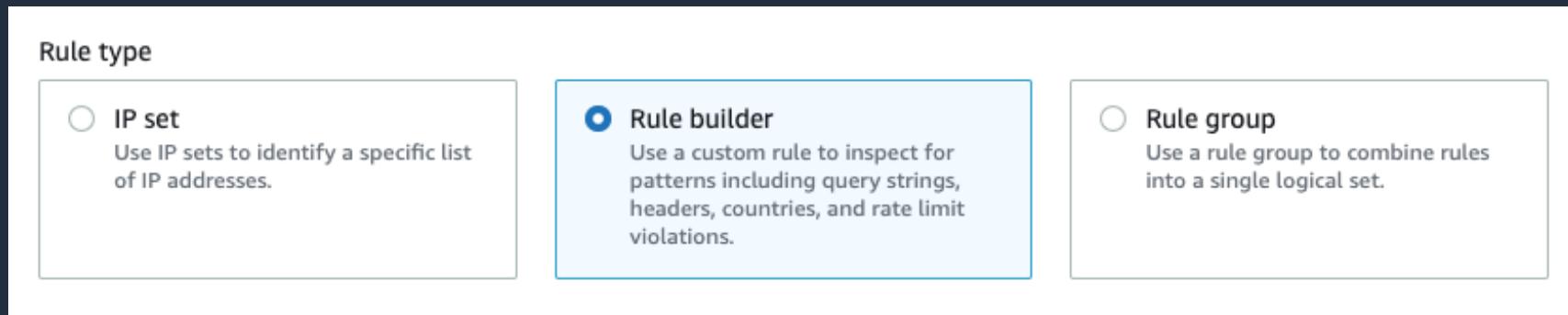
AWS WAF





AWS WAF

- **Web ACLs** – You use a web access control list (ACL) to protect a set of AWS resources
- **Rules** – Each rule contains a statement that defines the inspection criteria, and an action to take if a web request meets the criteria
- **Rule groups** – You can use rules individually or in reusable rule groups





AWS WAF

- **IP Sets** - An IP set provides a collection of IP addresses and IP address ranges that you want to use together in a rule statement
- **Regex pattern set** - A regex pattern set provides a collection of regular expressions that you want to use together in a rule statement



AWS WAF

A **rule action** tells AWS WAF what to do with a web request when it **matches** the criteria defined in the rule:

- **Count** – AWS WAF counts the request but doesn't determine whether to allow it or block it. With this action, AWS WAF continues processing the remaining rules in the web ACL
- **Allow** – AWS WAF allows the request to be forwarded to the AWS resource for processing and response
- **Block** – AWS WAF blocks the request and the AWS resource responds with an HTTP 403 (Forbidden) status code



Match statements compare the web request or its origin against conditions that you provide

Match Statement	Description
Geographic match	Inspects the request's country of origin
IP set match	Inspects the request against a set of IP addresses and address ranges
Regex pattern set	Compares regex patterns against a specified request component
Size constraint	Checks size constraints against a specified request component
SQLi attack	Inspects for malicious SQL code in a specified request component
String match	Compares a string to a specified request component
XSS scripting attack	Inspects for cross-site scripting attacks in a specified request component

AWS Shield





AWS Shield

- AWS Shield is a managed Distributed Denial of Service (DDoS) protection service
- Safeguards web application running on AWS with always-on detection and automatic inline mitigations
- Helps to minimize application downtime and latency
- Two tiers –
 - **Standard** – no cost
 - **Advanced** - \$3k USD per month and 1 year commitment
- Integrated with Amazon CloudFront (standard included by default)

Amazon Macie



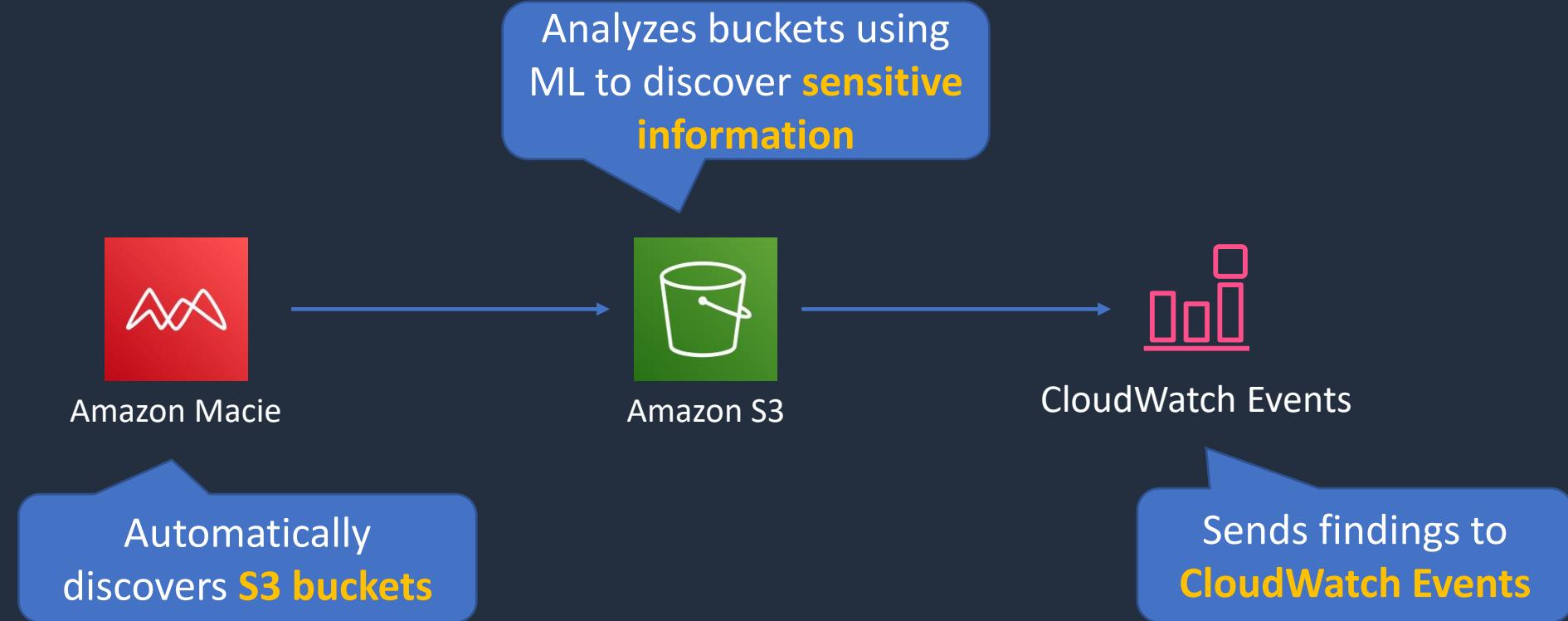


Amazon Macie

- Macie is a fully managed data security and data privacy service
- Uses machine learning and pattern matching to discover, monitor, and help you protect your sensitive data on Amazon S3
- Macie enables security compliance and preventive security as follows:
 - Identify a variety of data types, including PII, Protected Health Information (PHI), regulatory documents, API keys, and secret keys
 - Identify changes to policy and access control lists
 - Continuously monitor the security posture of Amazon S3
 - Generate security findings that you can view using the Macie console, AWS Security Hub, or Amazon EventBridge
 - Manage multiple AWS accounts using AWS Organizations



Amazon Macie



Amazon Inspector





Amazon Inspector

- Runs assessments that check for security exposures and vulnerabilities in EC2 instances
- Can be configured to run on a schedule
- Agent must be installed on EC2 for host assessments
- Network assessments do not require an agent



Network Assessments

- Assessments: Network configuration analysis to check for ports reachable from outside the VPC
- If the Inspector Agent is installed on your EC2 instances, the assessment also finds processes reachable on port
- Price based on the number of instance assessments



Host Assessments

- Assessments: Vulnerable software (CVE), host hardening (CIS benchmarks), and security best practices
- Requires an agent (auto-install with SSM Run Command)
- Price based on the number of instance assessments

AWS GuardDuty





AWS GuardDuty

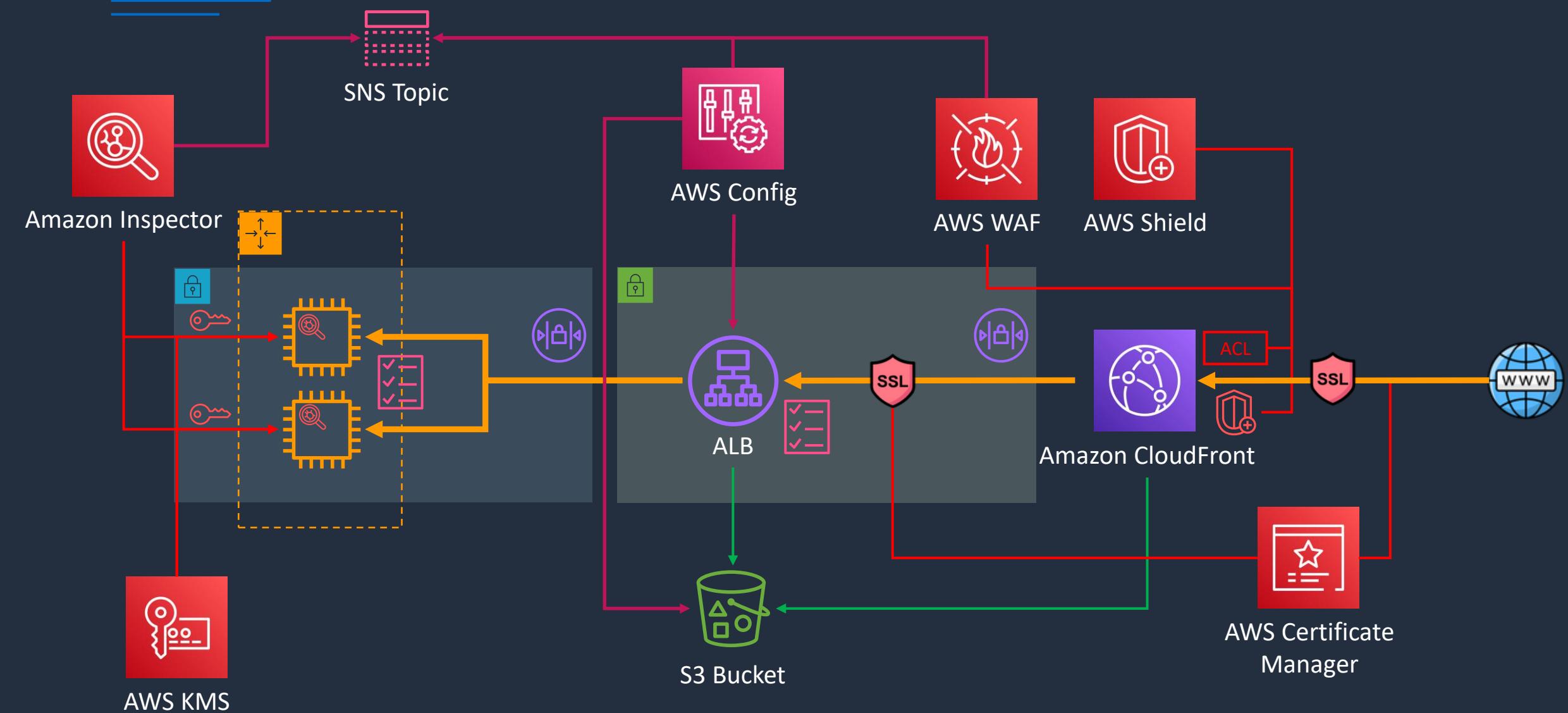
- Intelligent threat detection service
- Detects account compromise, instance compromise, malicious reconnaissance, and bucket compromise
- Continuous monitoring for events across:
 - **AWS CloudTrail Management Events**
 - **AWS CloudTrail S3 Data Events**
 - **Amazon VPC Flow Logs**
 - **DNS Logs**

Defense In-Depth





Secure Multi-Tier Architecture



Architecture Patterns - Security





Architecture Patterns – Security

Requirement

Need to enable custom domain name and encryption in transit for an application running behind an Application Load Balancer

Website running on EC2 instances behind and ALB must be protected against well known web exploits

Need to block access to an application running on an ALB from connections originating in a specific list of countries

Solution

Use AWS Route 53 to create an Alias record to the ALB's DNS name and attach an SSL/TLS certificate issued by Amazon Certificate Manager (ACM)

Create a Web ACL in AWS WAF to protect against web exploits and attach to the ALB

Create a Web ACL in AWS WAF with a geographic match and block traffic that matches the list of countries



Requirement

Company needs to encrypt large volumes of data using a CMK in AWS KMS

Mobile app requires authorized access to AWS services. Users authenticate using social IdPs and a preconfigured Web UI is required for logging in

Company has an on-premises Microsoft AD and AWS DX connection. Requires joining EC2 instances to on-premises domain

Solution

Create a data encryption key using the CMK to encrypt large volumes of data

Create an Amazon Cognito User Pool that leverages the social IdPs and an Identity Pool for gaining temporary credentials for AWS

Configure an AD Connector that uses the on-premises AD

SECTION 16

Migration and Transfer Services

AWS Migration Tools Overview





AWS Migration Tools



Region

Collect data

about servers in
on-premises DC



AWS Application
Discovery Service



AWS Migration Hub

**Monitor migrations that use
AWS or partner tools**



Amazon S3



Amazon RDS



EC2 Instances



EFS File system



AWS Application
Migration Service



AWS Database Migration
Service



AWS DataSync



Corporate data center



Servers



Database



NAS / File
Server



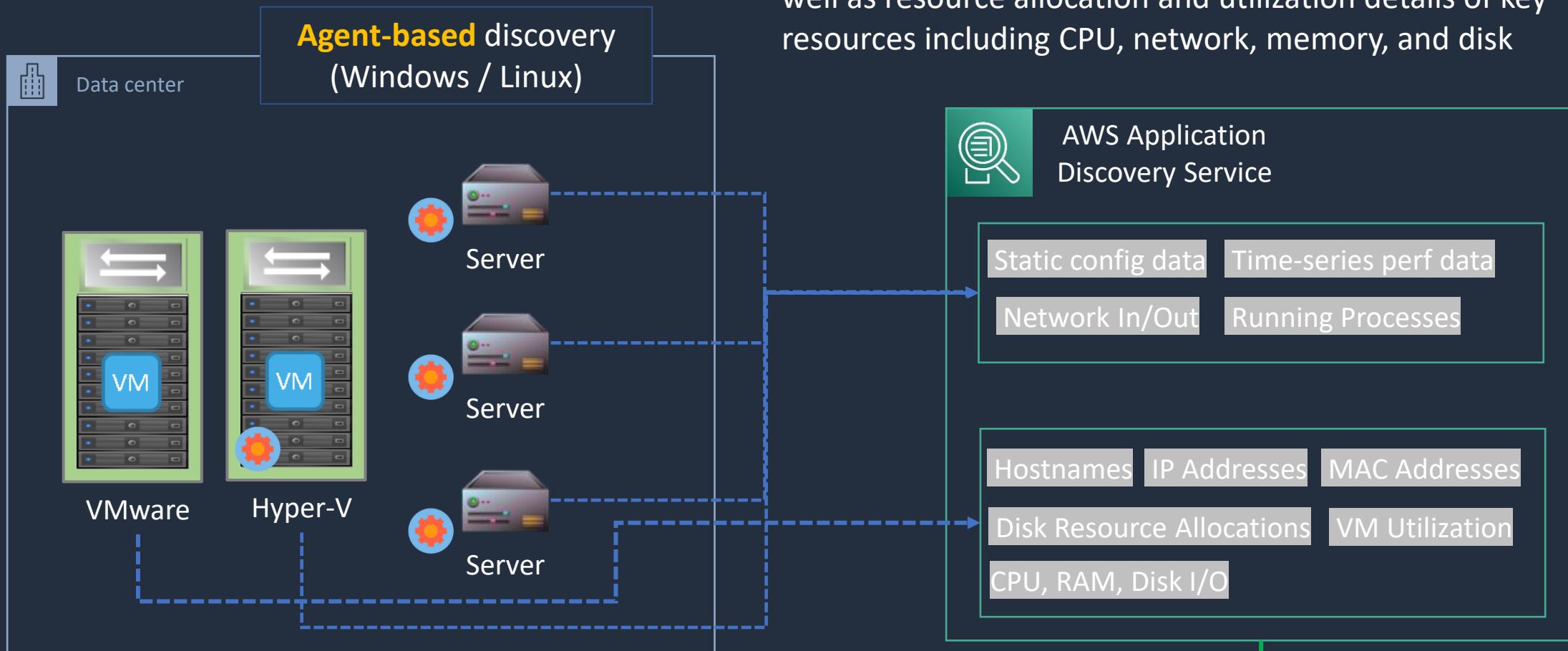
VPN, Direct
Connect or Internet



DigitalCloud
TRAINING



AWS Application Discovery Service



Agentless discovery
(OVA) file in vCenter

Data can be saved to S3
and **queried** with
Athena / QuickSight

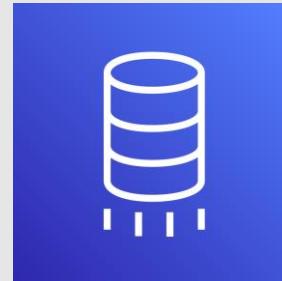




AWS Application Discovery Service

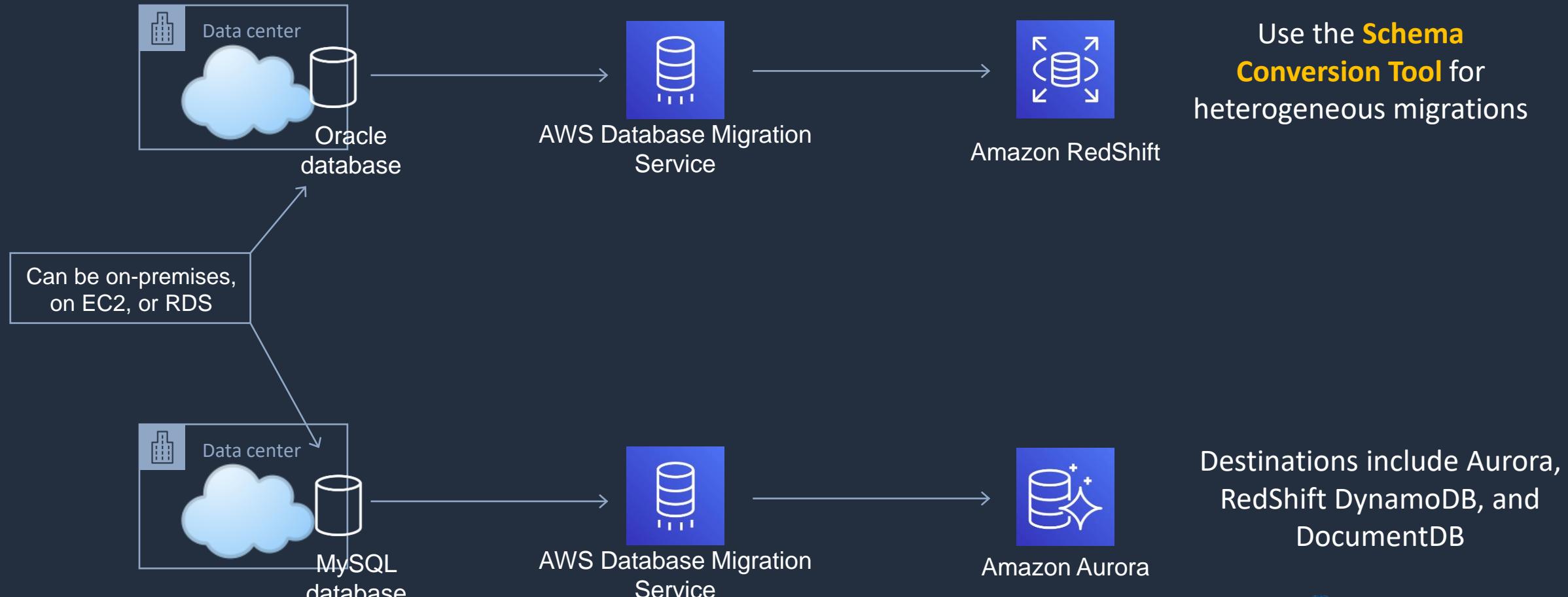
	Discovery Connector	Discovery Agent
Supported Servers	VMware	VMware, Physical
Deployment	Per vCenter	Per Server
Collected data	VM inventory, configuration, and performance history such as CPU, memory, and disk usage	System configuration System performance Running processes Network connections between systems
Supported OS	Any OS running in VMware vCenter (v5.5, v6, & v6.5)	Amazon Linux, Ubuntu, RHEL, CentOS, SUSE, Windows Server

AWS Database Migration Service (DMS)





AWS Database Migration Service (DMS)





AWS DMS Use Cases

- **Cloud to Cloud** – EC2 to RDS, RDS to RDS, RDS to Aurora
- **On-Premises to Cloud**
- **Homogeneous migrations** – Oracle to Oracle, MySQL to RDS MySQL, Microsoft SQL to RDS for SQL Server
- **Heterogeneous migrations** – Oracle to Aurora, Oracle to PostgreSQL, Microsoft SQL to RDS MySQL (must convert schema first with the **Schema Conversion Tool (SCT)**)



AWS DMS Use Cases

- **Development and Test** – use the cloud for dev/test workloads
- **Database consolidation** – consolidate multiple source DBs to a single target DB
- **Continuous Data Replication** – use for DR, dev/test, single source multi-target or multi-source single target

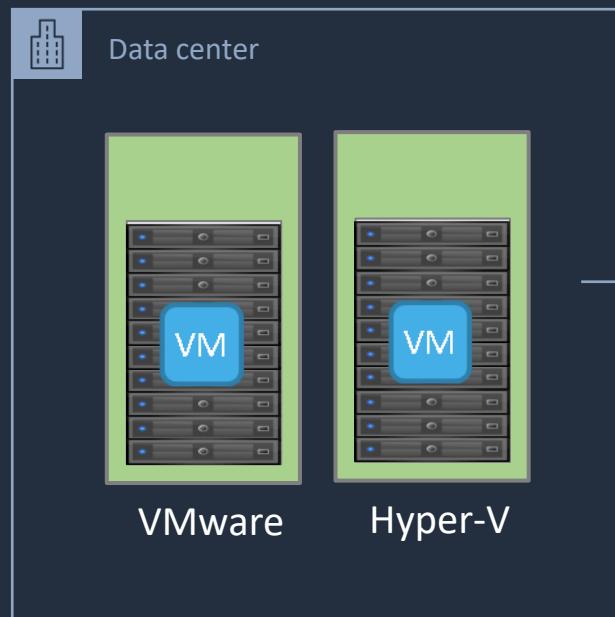
AWS Application Migration Service (MGN)



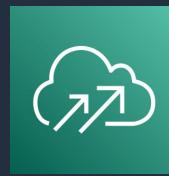


AWS Application Migration Service (MGN)

AWS recommend **agent-based** replication when possible as it support continuous data protection (CDP)

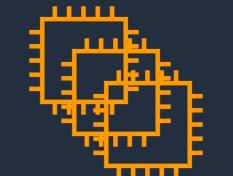


Automated, **incremental** and **scheduled** migrations



AWS Application
Migration Service

Agentless snapshot-based
replication possible with the **AWS**
MGN vCenter Client installed



EC2 Instances

EC2 instances are launched from launch templates



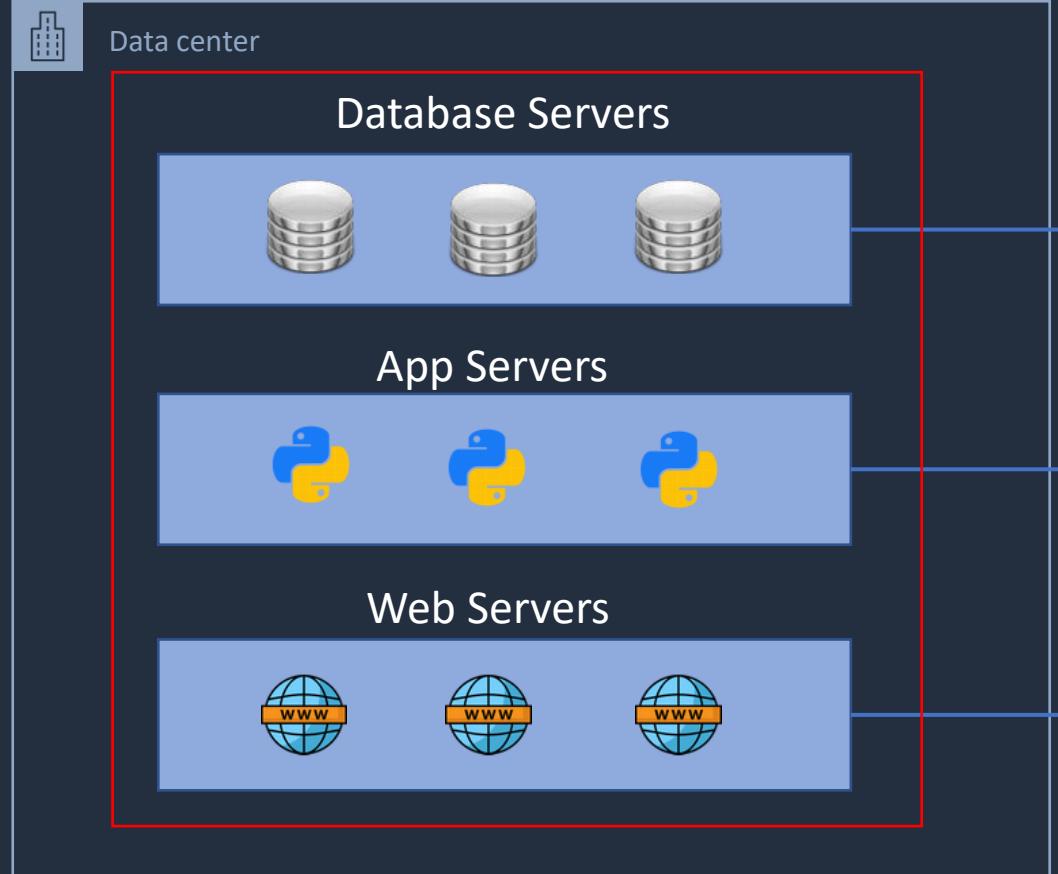
Launch Template



CloudWatch Events
and Lambda can
automate actions



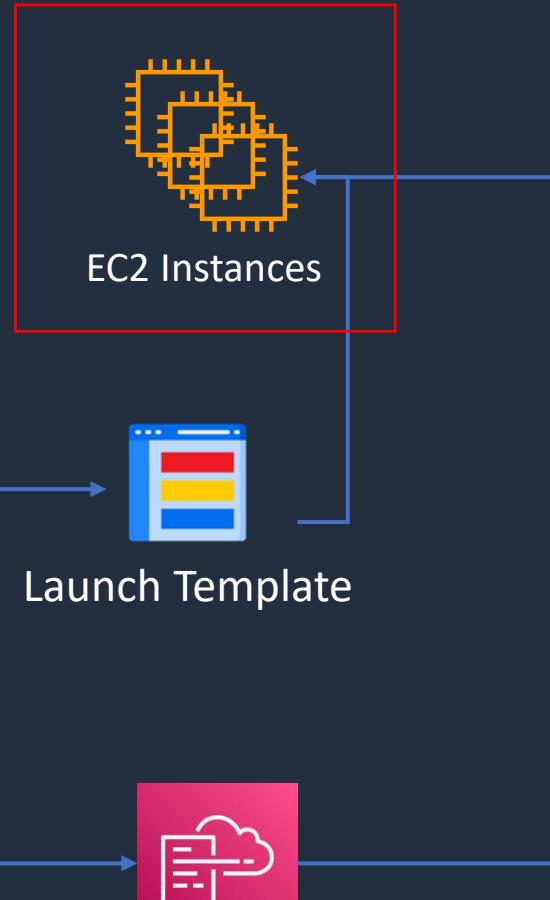
AWS Application Migration Service (MGN)



Entire application group
is launched in a
migration wave

AWS Application
Migration Service

CloudFormation
Template



CloudFormation



AWS Application Migration Service (MGN)

- Highly automated lift-and-shift (rehost) solution for migrating applications to AWS
- Utilizes continuous, block-level replication and enables short cutover windows measured in minutes
- Server Migration Service (SMS) utilizes incremental, snapshot-based replication and enables cutover windows measured in hours
- AWS recommend using AWS MGN instead of SMS
- Integrates with the Cloud Migration Factory for orchestrating manual processes
- Can migrate virtual and physical servers

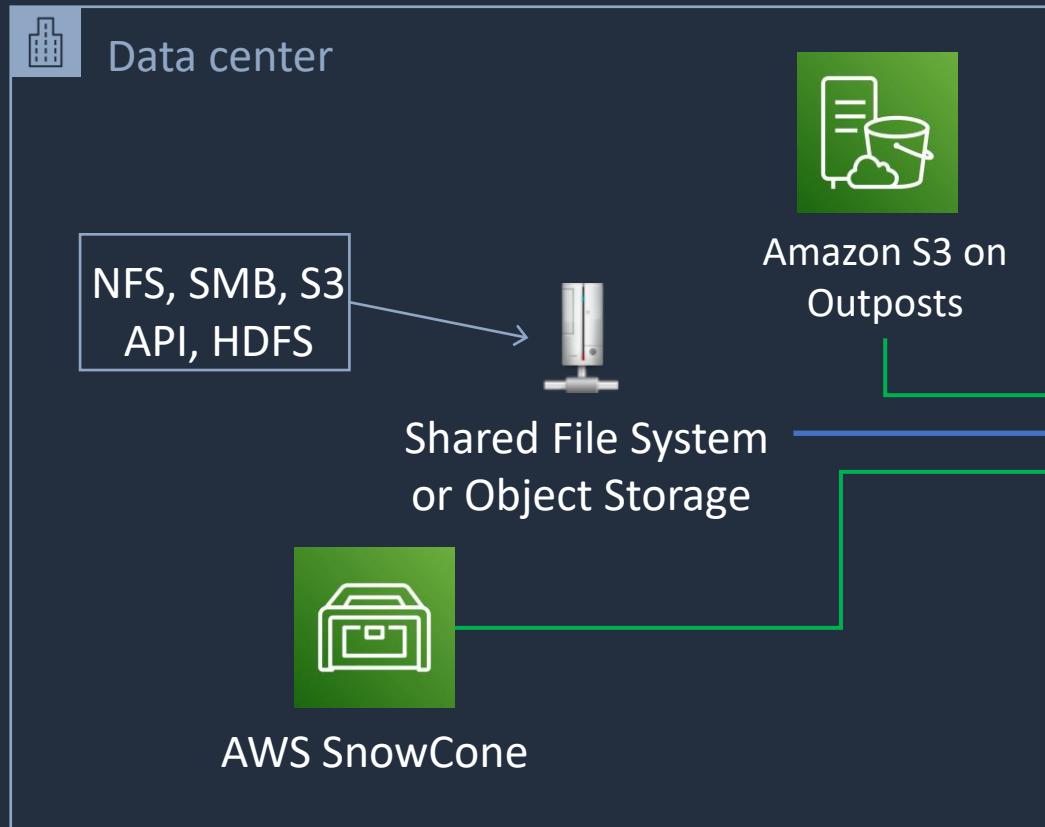
AWS DataSync





AWS DataSync

DataSync **software agent**
connects to storage system



DataSync agent installed
on Snowcone

Data is encrypted in transit with **TLS**
Scheduled, automated data transfer





AWS DataSync

- Secure, online service that automates and accelerates moving data between on premises and AWS Storage services
- DataSync can copy data between:
 - Network File System (NFS) shares
 - Server Message Block (SMB) shares
 - Hadoop Distributed File Systems (HDFS)
 - Self-managed object storage
 - AWS Snowcone
 - Amazon S3 buckets
 - Amazon Elastic File System (Amazon EFS) file systems
 - Amazon FSx (Windows, Lustre, OpenZFS, and NetApp ONTAP)

AWS Snow Family





AWS Snowball Family

- **AWS Snowball and Snowmobile** are used for migrating large volumes of data to AWS
- **Snowball Edge Compute Optimized**
 - Provides block and object storage and optional GPU
 - Use for data collection, machine learning and processing, and storage in environments with intermittent connectivity (edge use cases)
- **Snowball Edge Storage Optimized**
 - Provides block storage and Amazon S3-compatible object storage
 - Use for local storage and large-scale data transfer
- **Snowcone**
 - Small device used for edge computing, storage and data transfer
 - Can transfer data offline or online with AWS DataSync agent





AWS Snowball Family

- Uses a secure storage device for physical transportation
- Snowball Client is software that is installed on a local computer and is used to identify, compress, encrypt, and transfer data
- Uses 256-bit encryption (managed with the AWS KMS) and tamper-resistant enclosures with TPM
- Snowball (80TB) (50TB) “petabyte scale”
- Snowball Edge (100TB) “petabyte scale”
- Snowmobile – “exabyte scale” with up to 100PB per Snowmobile





AWS Snowball Family

Ways to optimize the performance of Snowball transfers:

1. Use the latest Mac or Linux Snowball client
2. Batch small files together
3. Perform multiple copy operations at one time
4. Copy from multiple workstations
5. Transfer directories, not files





AWS Snowball Use Cases

- **Cloud data migration** – migrate data to the cloud
- **Content distribution** – send data to clients or customers
- **Tactical Edge Computing** – collect data and compute
- **Machine learning** – run ML directly on the device
- **Manufacturing** – data collection and analysis in the factory
- **Remote locations with simple data** – pre-processing, tagging, compression etc.

The 7 Rs of Migration





The 7 Rs of Migration

Most Effort



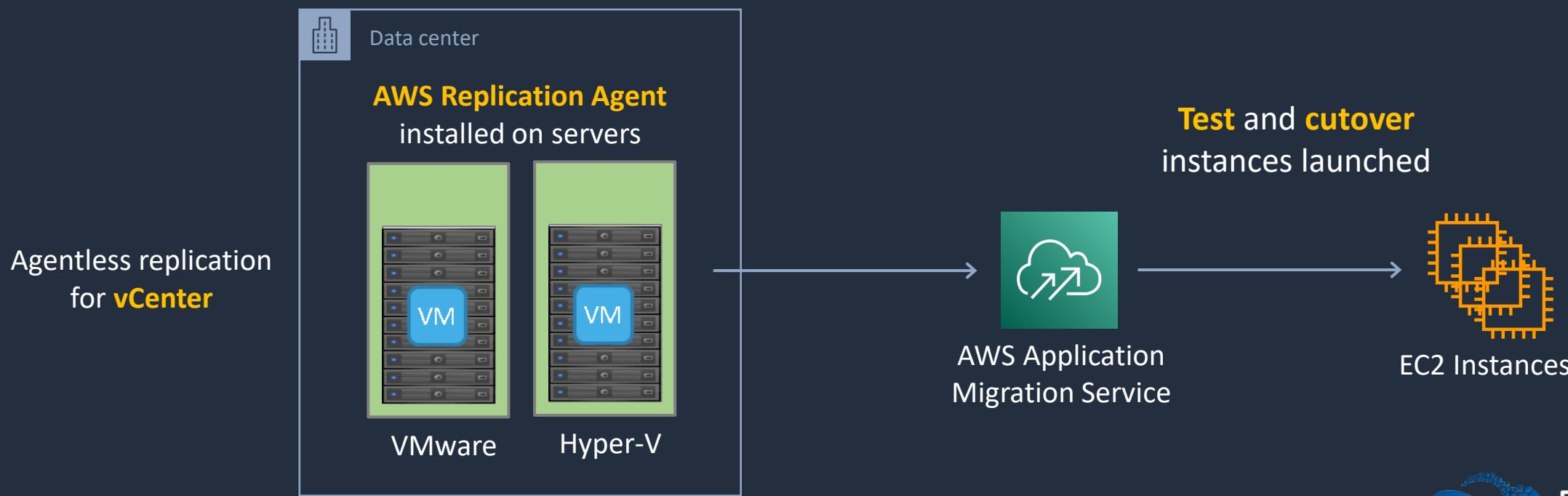
Least Effort

- **Refactor** – Re-architect to a cloud-native serverless architecture
- **Replatform** – Ex. database to RDS; server to Elastic Beanstalk
- **Repurchase** – Use a different solution (e.g. SaaS)
- **Rehost** – OS/App moved to another host system (lift & shift)
- **Relocate** – Move without modification (lift & Shift)
- **Retain** – Leave as is (revisit at a later date)
- **Retire** – No longer needed, get rid of it



The 7 Rs of Migration

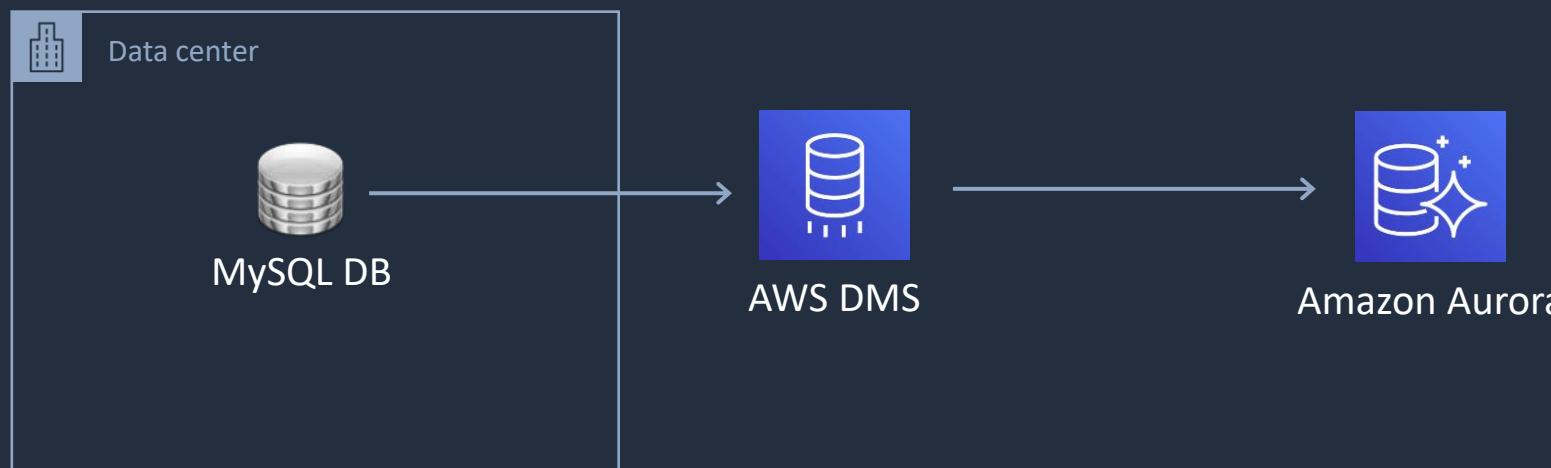
- **Rehost** – OS/App moved to another host system
 - AWS recommend the **AWS Application Migration Service** (AWS MGN) for lift & shift
 - Can also use AWS SMS and AWS VM Import / Export





The 7 Rs of Migration

- **Replatform** – Ex. database to RDS; server to Elastic Beanstalk
 - AWS Database Migration Service (AWS DMS) and Schema Conversion Tool (SCT)
 - Migrate to EB using custom AMI, or code-level migration
 - Moderate development and migration effort





The 7 Rs of Migration

- **Refactor** – Re-architect to a cloud-native serverless architecture
 - Leverage serverless services and event-driven architecture patterns
 - Migrate servers to serverless functions or containers
 - Migrate databases to managed DBs or serverless NoSQL DBs
 - Migrate file stores to object stores or elastic file systems
 - Decouple with queues, notification services, and orchestration tools
 - May involve a large amount of development and migration effort as well as expense

Architecture Patterns – Migration and Transfer





Architecture Patterns – Migration and Transfer

Requirement

Company is migrating Linux and Windows VMs in VMware to the cloud. Need to determine performance requirements for right-sizing

Solution

Install the Application Discovery Service discovery connector in VMware vCenter to gather data

Company has a mixture of VMware VMs and physical servers to migrate to AWS and dependencies exist between application components

Install the AWS MGN replication agent and create Application groups to migrate in waves

Need to migrate an Oracle data warehouse to AWS

Migrate using AWS DMS and SCT to a RedShift data warehouse



Architecture Patterns – Migration and Transfer

Requirement

Snowball Edge used to transfer millions of small files using a shell script.
Transfer times are very slow

Solution

Perform multiple copy operations at one time by running each command from a separate terminal, in separate instances of the Snowball client

Need to minimize downtime for servers that must be migrated to AWS

Use AWS MGN and perform a final synchronization before cutting over in a short outage window

Need to migrate 50TB of data and company only has a 1Gbps internet link. Requirement is urgent

Use AWS Snowball to transfer the data

SECTION 17

Web, Mobile, ML, and Cost Management

AWS Amplify and AppSync





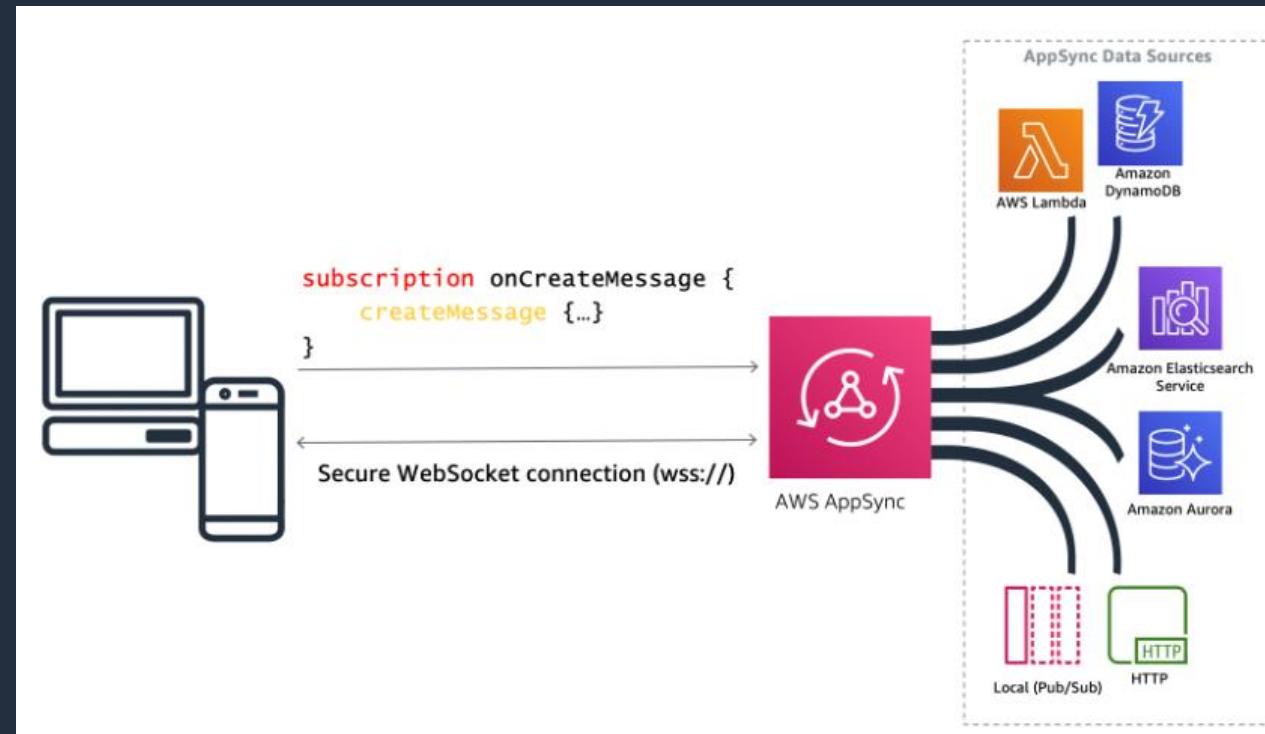
AWS Amplify

- Tools and features for building full-stack applications on AWS
- Build web and mobile backends, and web frontend UIs
- AWS Amplify Studio is a visual interface for building web and mobile apps:
 - Use the visual interface to define a data model, user authentication, and file storage without backend expertise
 - Easily add AWS services not available within Amplify Studio using the AWS Cloud Development Kit (CDK)
 - Connect mobile and web apps using Amplify Libraries for iOS, Android, Flutter, React Native, and web (JavaScript)
- AWS Amplify Hosting is a fully managed CI/CD and hosting service for fast, secure, and reliable static and server-side rendered apps



AWS AppSync

- AWS AppSync is a fully managed service that makes it easy to develop GraphQL APIs
- Applications can securely access, manipulate, and receive real-time updates from multiple data sources such as databases or APIs





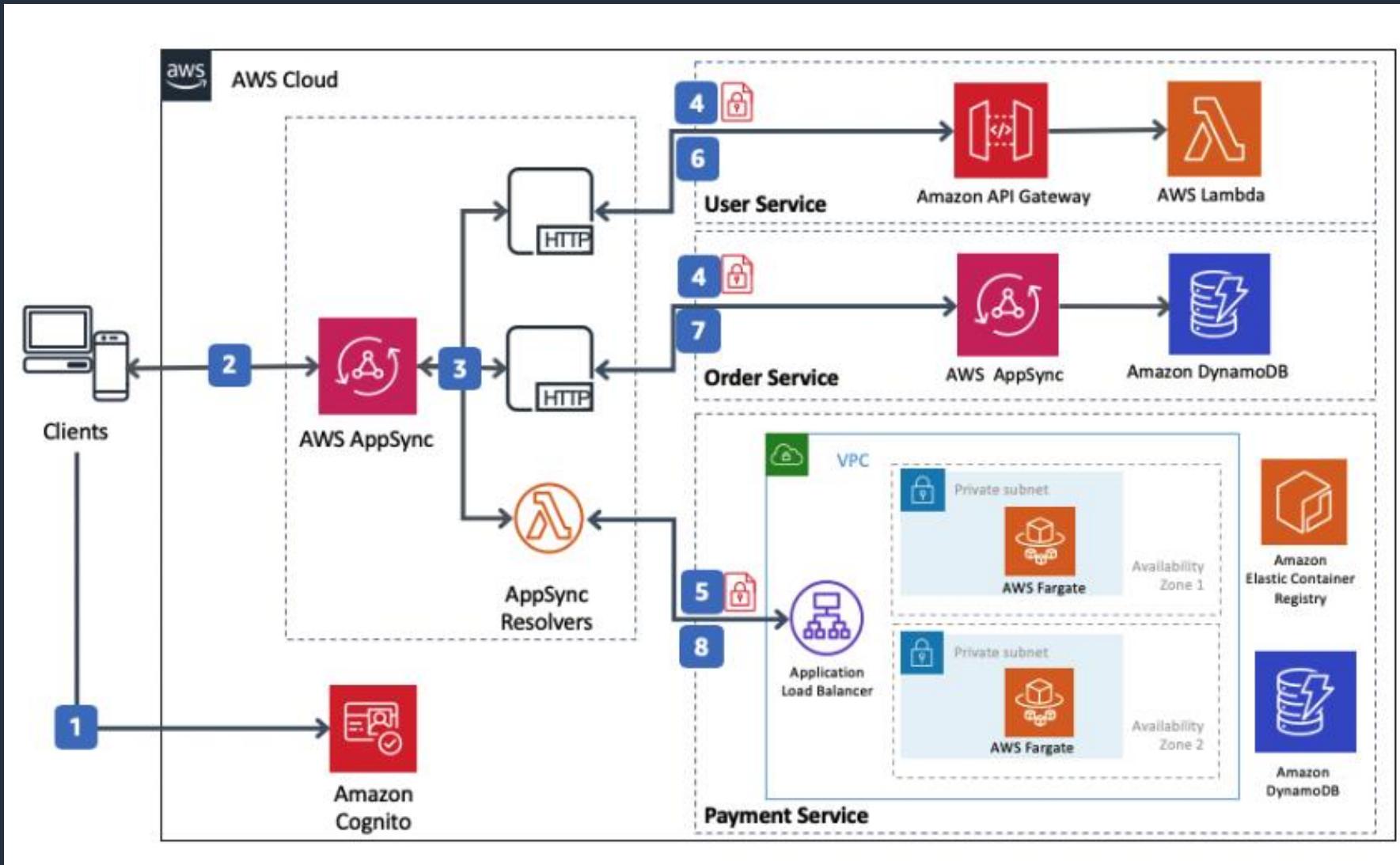
AWS AppSync

- AWS AppSync automatically scales a GraphQL API execution engine up and down to meet API request volumes
- Uses GraphQL, a data language that enables client apps to fetch, change and subscribe to data from servers
- AWS AppSync lets you specify which portions of your data should be available in a real-time manner using GraphQL Subscriptions
- AWS AppSync supports AWS Lambda, Amazon DynamoDB, and Amazon Elasticsearch
- Server-side data caching capabilities reduce the need to directly access data sources
- AppSync is fully managed and eliminates the operational overhead of managing cache clusters



AWS AppSync

Example of using **AppSync** and **Amplify** to simplify access to microservices



Amplify is used to build and host the WebStore application and create backend services

AppSync creates a unified API layer for integrating the microservices

AWS Device Farm



AWS Machine Learning and AI Services





AWS Rekognition

Identify objects



Perform facial analysis



Celebrity recognition





AWS Rekognition in Event-Driven Architecture





AWS Rekognition

- Add image and video analysis to your applications
- Identify objects, people, text, scenes, and activities in images and videos
- Processes videos stored in an Amazon S3 bucket
- Publish completion status to Amazon SNS Topic



Amazon Transcribe

- Add speech to text capabilities to applications
- Recorded speech can be converted to text before it can be used in applications
- Uses a deep learning process called automatic speech recognition (ASR) to convert speech to text quickly and accurately



Amazon Translate

- Neural machine translation service that delivers fast, high-quality, and affordable language translation
- Uses deep learning models to deliver more accurate and more natural sounding translation
- Localize content such as websites and applications for your diverse users



Amazon Comprehend

- Natural-language processing (NLP) service
- Uses machine learning to uncover information in unstructured data
- Can identify critical elements in data, including references to language, people, and places, and the text files can be categorized by relevant topics
- In real time, you can automatically and accurately detect customer sentiment in your content



Amazon Lex

- Conversational AI for Chatbots
- Build conversational interfaces into any application using voice and text
- Build bots to increase contact center productivity, automate simple tasks, and drive operational efficiencies across the enterprise



Amazon DevOps Guru

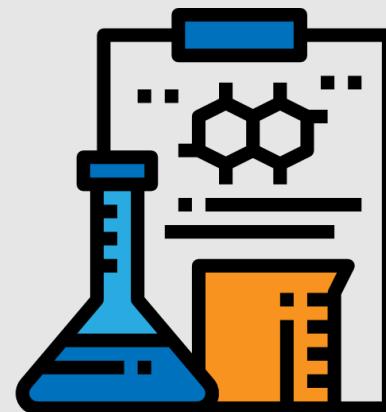
- Cloud operations service for improving **application operational performance and availability**
- Detect behaviors that deviate from normal operating patterns
- Benefits:
 - Automatically detect operational issues
 - Resolve issues with ML-powered insights
 - Elastically scale operational analytics
 - Uses ML to reduce alarm noise



Amazon CodeGuru Security

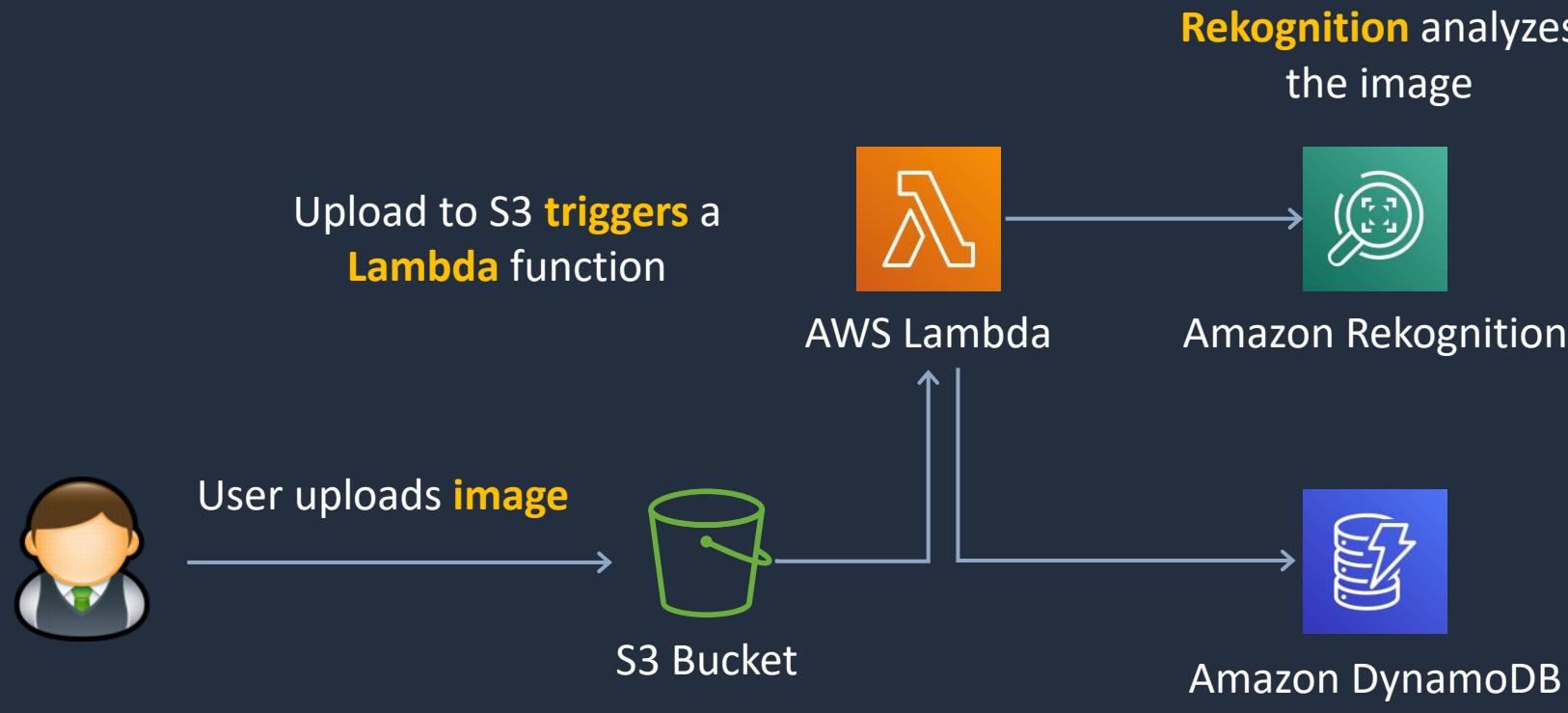
- Detect, track, and fix code security vulnerabilities anywhere in the development cycle using ML and automated reasoning
- Integrates with IDEs and CI/CD tools
- Automated bug tracking
- Assisted remediation through suggested code fixes
- Offers performance optimization recommendations
- Detects anomalies in application profiles

Process and Analyze Images





Process and Analyze Images



The **Lambda function** receives the results and creates an item in **DynamoDB**

AWS License Manager





AWS License Manager

- Used to manage licenses from software vendors
- For example, manage your Microsoft, Oracle, SAP and IBM licenses
- Centralized management of software licenses for AWS and on-premises resources
- Can track license usage including when licensed based on virtual cores (vCPUs), sockets, or number of machines
- Distribute, activate, and track software licenses across accounts and throughout an organization
- Enforce limits across multiple Regions

AWS Compute Optimizer





AWS Compute Optimizer

- Recommends optimal AWS resources for your workloads to reduce costs and improve performance
- Uses machine learning to analyze historical utilization metrics
- Offers optimization guidance for:
 - Amazon EC2 instances
 - Amazon EBS volumes
 - AWS Lambda functions
- Results can be viewed in the console or via the CLI



AWS Compute Optimizer

AWS Compute Optimizer > Dashboard

Dashboard Info

Findings per AWS resource

090765505187 ▾

Filter by one or more Regions

Region: US East (N. Virginia)

EC2 instances (13) Info

30.77% Under-provisioned 7.... 61.54% Over-provisioned

Findings

- Under-provisioned: 4 instances
- Optimized: 1 instance
- Over-provisioned: 8 instances

[View recommendations for EC2 instances](#)

Auto Scaling groups (1) Info

100% Not optimized

Findings

- Not optimized: 1 group
- Optimized: 0 groups

[View recommendations for Auto Scaling groups](#)



AWS Compute Optimizer

AWS Compute Optimizer > Dashboard > Recommendations for EC2 instances

Recommendations for EC2 instances (8) Info

Recommendations for modifying current resources for better cost and performance.

Action ▾

View detail

Filter by one or more Regions

090765505187 ▾

Over-provisioned ▾

< 1 >



Region: US East (N. Virginia) X

Clear filters

Instance ID ▾	Instance name ▾	Finding ▾	Current Instance type ▾	Current On-Demand price ▾	Recommended Instance type ▾	Recommended On-Demand price
i-0fb9323080785de1e	-	Over-provisioned	c5.xlarge	\$0.17 per hour	t3.large	\$0.0832 per hour
i-0f4f4c06ad8afe81a	-	Over-provisioned	m5.2xlarge	\$0.384 per hour	r5.xlarge	\$0.252 per hour
i-0f277818dfef522e9	-	Over-provisioned	c5.xlarge	\$0.17 per hour	t3.large	\$0.0832 per hour
i-0ceb95ed248026d24	-	Over-provisioned	m5.xlarge	\$0.192 per hour	r5.large	\$0.126 per hour
i-0af9322ff627d7e8f	-	Over-provisioned	m5.xlarge	\$0.192 per hour	r5.large	\$0.126 per hour
i-07084b94d1bcf391b	-	Over-provisioned	c5.xlarge	\$0.17 per hour	t3.large	\$0.0832 per hour
i-069f6e837890db127	-	Over-provisioned	c5.xlarge	\$0.17 per hour	t3.large	\$0.0832 per hour
i-0218a45abd8b53658	-	Over-provisioned	m5.xlarge	\$0.192 per hour	r5.large	\$0.126 per hour

AWS Cost Explorer



AWS Cost Allocation Tags



AWS Cost Management Tools





AWS Cost Explorer

- The **AWS Cost Explorer** is a free tool that allows you to view charts of your costs
- You can view cost data for the past 13 months and forecast how much you are likely to spend over the next three months
- Cost Explorer can be used to discover patterns in how much you spend on AWS resources over time and to identify cost problem areas
- Cost Explorer can help you to identify service usage statistics such as:
 - Which services you use the most
 - View metrics for which AZ has the most traffic
 - Which linked account is used the most



AWS Cost & Usage Report

- Publish AWS billing reports to an Amazon S3 bucket
- Reports break down costs by:
 - Hour, day, month, product, product resource, tags
- Can update the report up to three times a day
- Create, retrieve, and delete your reports using the AWS CUR API Reference



AWS Price List API

- Query the prices of AWS services
- **Price List Service API** (AKA the Query API) – query with JSON
- **AWS Price List API** (AKA the Bulk API) – query with HTML
- Alerts via Amazon SNS when prices change