

# AWS Certified Solutions Architect Professional

By Stéphane Maarek



COURSE →



EXTRA PRACTICE EXAMS →

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

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

# AWS Certified Solutions Architect Professional Course SAP-C01

# Setting the right expectations for this course

- This course is all slides based
  - I'm assuming you have experience using AWS
  - No hands-on will come with the course. You should know the basics
  - It's fast paced. Your time is valuable. Feel free to slow me down to 0.75x
- If you just passed the AWS Certified Solutions Architect cert
  - I recommend you go through AWS Certified Developer, SysOps & DevOps
  - I know you are eager to get the SAP certification, but take your time
- The AWS knowledge needed for the SA Pro exam
  - Is extremely similar to the knowledge for SAA
  - The questions are more complex, and knowing details is very important
  - It's possible that multiple answers are correct, but one is the most appropriate

# The AWS Certified Solutions Architect Professional Exam

- Is HARD
- Tests real AWS experience
- Will test you on some very subtle service features
- I have included quizzes for every single section BUT...
  - The quizzes are not “scenario based” / “exam-like”
  - They only help you extract some important notions out of what you’re learning
  - This is my optimal way of teaching you about specific topics
  - Please trust my teaching process

# Practice Exams

- This course does not come with practice exams
  - I recommend you look on Udemy for extra practice exams
  - I really want to focus this course on the knowledge needed
  - I may come up with a practice exam at some point (to be purchased separately)
- Warning:
  - This course is on the NEW CERTIFICATION (SAP-C01)
  - You may see outdated content in other practice exams, other courses, etc...
  - This course is not incomplete, it's more targeted towards the knowledge you actually need to know to pass the exam

# Identity & Federation Section

# IAM – What should you know by now

- Users: long term credentials
- Groups
- Roles: short-term credentials, uses STS
  - EC2 Instance Roles: uses the **EC2 metadata** service. One role at a time per instance
  - Service Roles: API Gateway, CodeDeploy, etc...
  - Cross Account roles
- Policies
  - AWS Managed
  - Customer Managed
  - Inline Policies
- Resource Based Policies (S3 bucket, SQS queue, etc...)

# IAM Policies Deep Dive

- Anatomy of a policy: JSON doc with Effect, Action, Resource, Conditions, Policy Variables
- Explicit DENY has precedence over ALLOW
- Best practice: use least privilege for maximum security
  - Access Advisor: See permissions granted and when last accessed
  - Access Analyzer: Analyze resources that are shared with external entity
- Navigate Examples at:  
[https://docs.aws.amazon.com/IAM/latest/UserGuide/access\\_policies\\_examples.html](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_examples.html)

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "ec2:AttachVolume",  
                "ec2:DetachVolume"  
            ],  
            "Resource": "arn:aws:ec2:*::instance/*",  
            "Condition": {  
                "StringEquals": {"ec2:ResourceTag/Department": "Development"}  
            }  
        },  
        {  
            "Effect": "Allow",  
            "Action": [  
                "ec2:AttachVolume",  
                "ec2:DetachVolume"  
            ],  
            "Resource": "arn:aws:ec2:*::volume/*",  
            "Condition": {  
                "StringEquals": {"ec2:ResourceTag/VolumeUser": "${aws:username}"}  
            }  
        }  
    ]  
}
```

# IAM AWS Managed Policies

## AdministratorAccess

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

# IAM AWS Managed Policies

## PowerUserAccess

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "NotAction": [  
        "iam:*",  
        "organizations:*",  
        "account:*"  
      ],  
      "Resource": "*"  
    },...  
    ...{  
      "Effect": "Allow",  
      "Action": [  
        "iam:CreateServiceLinkedRole",  
        "iam>DeleteServiceLinkedRole",  
        "iam>ListRoles",  
        "organizations:DescribeOrganization",  
        "account>ListRegions"  
      ],  
      "Resource": "*"  
    }  
  ]  
}
```

Note how "NotAction" is used instead of Deny

# IAM Policies Conditions

```
"Condition" : { "{condition-operator}" : { "{condition-key}" : "{condition-value}" }}
```

## Operators:

- String (StringEquals, StringNotEquals, StringLike...)
  - "Condition": {"StringEquals": {"aws:PrincipalTag/job-category": "iamuser-admin"}}
  - "Condition": {"StringLike": {"s3:prefix": [ "", "home/", "home/\${aws:username}/" ]}}
- Numeric (NumericEquals, NumericNotEquals, NumericLessThan...)
- Date (DateEquals, DateNotEquals, DateLessThan...)
- Boolean (Bool):
  - "Condition": {"Bool": {"aws:SecureTransport": "true"}}
  - "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
- (Not) IpAddress:
  - "Condition": {"IpAddress": {"aws:SourceIp": "203.0.113.0/24"}}
- ArnEquals, ArnLike
- Null: "Condition": {"Null": {"aws:TokenIssueTime": "true"}}

# IAM Policies Variables and Tags

## Example: \${aws:username}

- "Resource": ["arn:aws:s3:::mybucket/\${aws:username}/\*"]

## AWS Specific:

- aws:CurrentTime, aws:TokenIssueTime, aws:principalType, aws:SecureTransport, aws:SourceIp, aws:userId, ec2:SourceInstanceIdARN

## Service Specific:

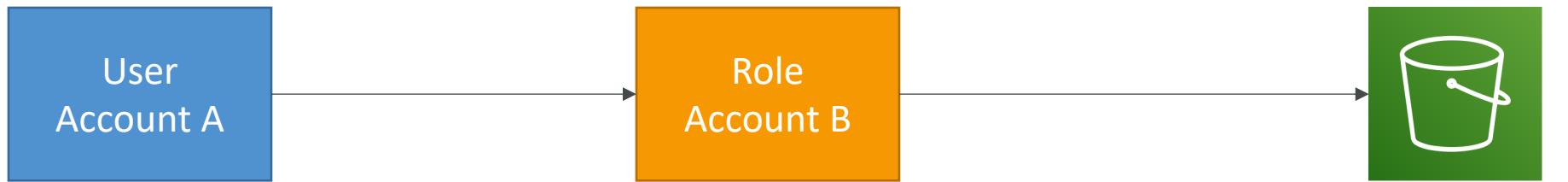
- s3:prefix, s3:max-keys, s3:x-amz-acl, sns:Endpoint, sns:Protocol...

## Tag Based:

- iam:ResourceTag/key-name, aws:PrincipalTag/key-name...

# IAM Roles vs Resource Based Policies

- Attach a policy to a resource (example: S3 bucket policy) versus attaching of a using a role as a proxy



# IAM Roles vs Resource Based Policies

- When you assume a role (user, application or service), you give up your original permissions and take the permissions assigned to the role
- When using a resource-based policy, the principal doesn't have to give up any permissions
- Example: User in account A needs to scan a DynamoDB table in Account A and dump it in an S3 bucket in Account B.
- Supported by: Amazon S3 buckets, SNS topics, SQS queues, Lambda functions, ECR, Backup, EFS, Glacier, Cloud9, AWS Artifact, Secrets Manager, ACM, KMS, CloudWatch Logs, API Gateway, EventBridge etc...

# IAM Permission Boundaries

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

**Example:**

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



**IAM Permission Boundary**

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

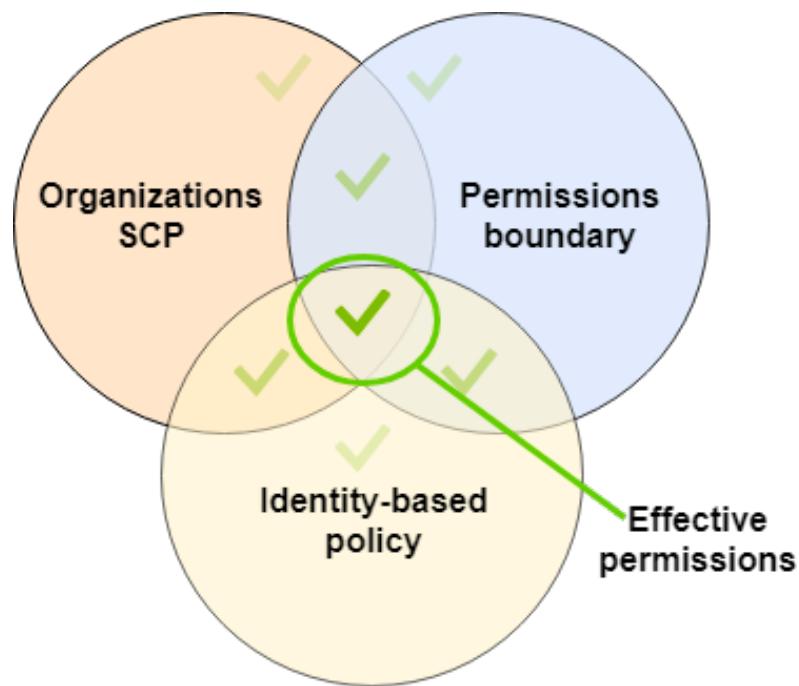
**IAM Permissions  
Through IAM Policy**



**No Permissions**

# IAM Permission Boundaries

- Can be used in combinations of AWS Organizations SCP



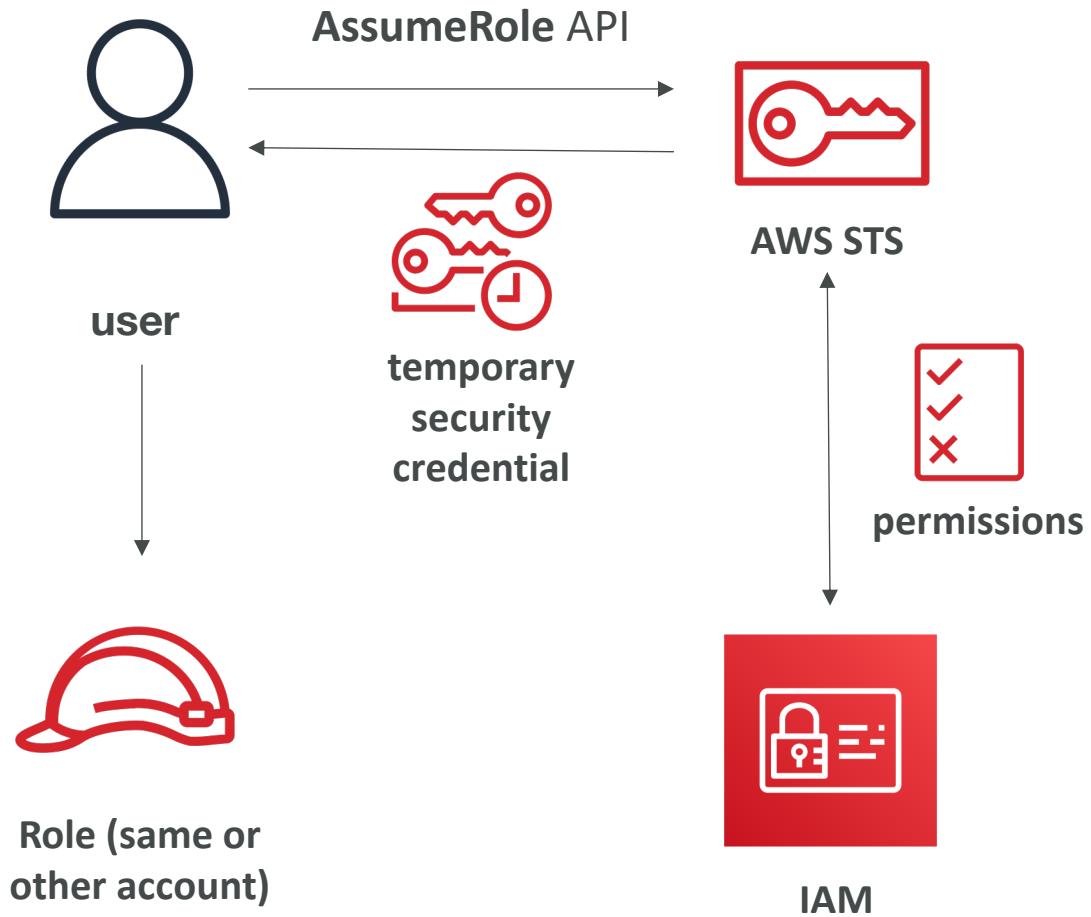
[https://docs.aws.amazon.com/IAM/latest/UserGuide/access\\_policies\\_boundaries.html](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_boundaries.html)

## Use cases

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

# Using STS to Assume a Role

- Define an IAM Role within your account or cross-account
- Define which principals can access this IAM Role
- Use AWS STS (Security Token Service) to retrieve credentials and impersonate the IAM Role you have access to (`AssumeRole API`)
- Temporary credentials can be valid between 15 minutes to 12 hour



# Assuming a Role with STS

- Provide access for an IAM user in one AWS account that you own to access resources in another account that you own
- Provide access to IAM users in AWS accounts owned by third parties
- Provide access for services offered by AWS to AWS resources
- Provide access for externally authenticated users (identity federation)
- Ability to revoke active sessions and credentials for a role  
(by adding a policy using a time statement – AWSRevokeOlderSessions)

When you assume a role (user, application or service), you give up your original permissions and take the permissions assigned to the role

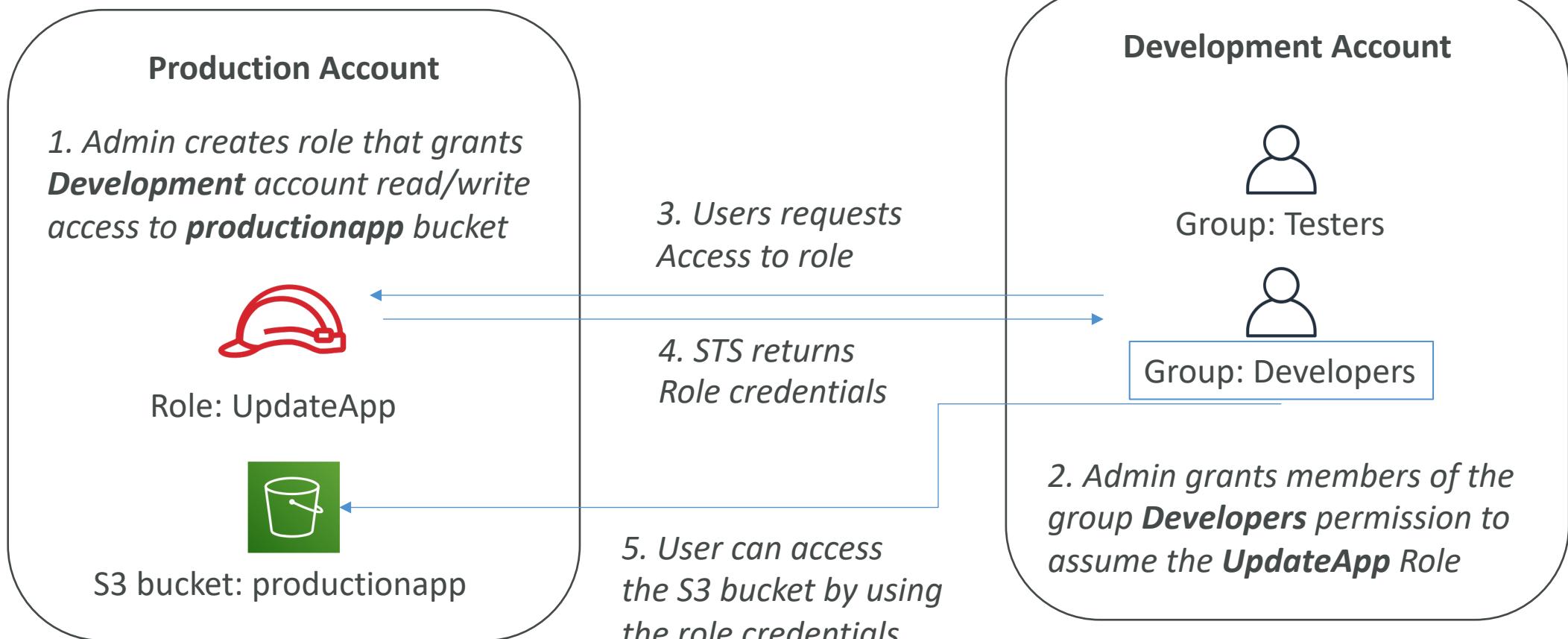
# Providing Access to an IAM User in Your or Another AWS Account That You Own

- You can grant your IAM users permission to switch to roles within your AWS account or to roles defined in other AWS accounts **that you own**.



- Benefits:
  - You must explicitly grant your users permission to assume the role.
  - Your users must actively switch to the role using the AWS Management Console or assume the role using the AWS CLI or AWS API
  - You can add multi-factor authentication (MFA) protection to the role so that only users who sign in with an MFA device can assume the role
  - Least privilege + auditing using CloudTrail

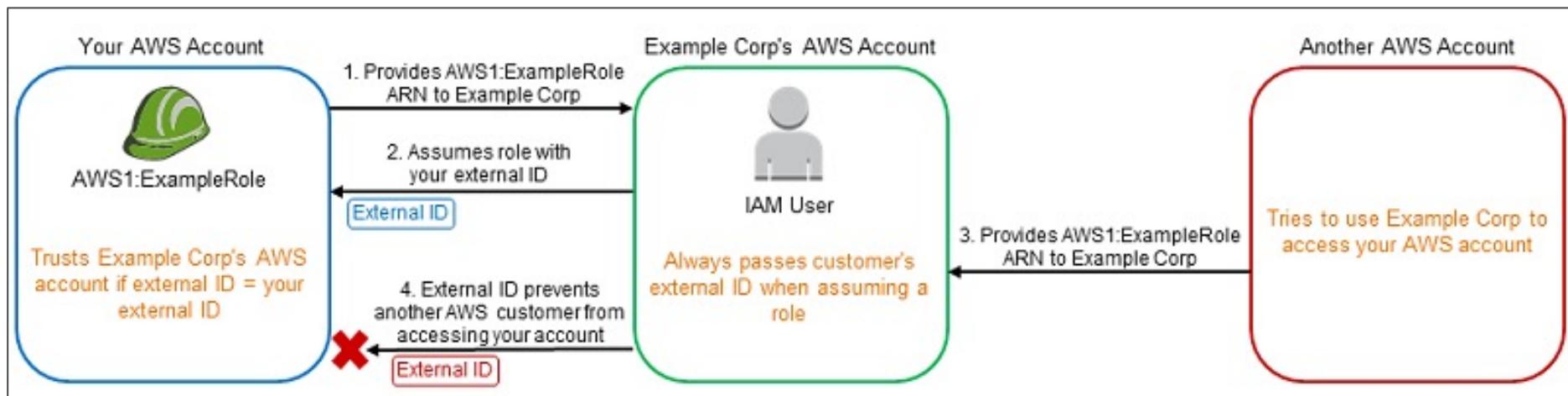
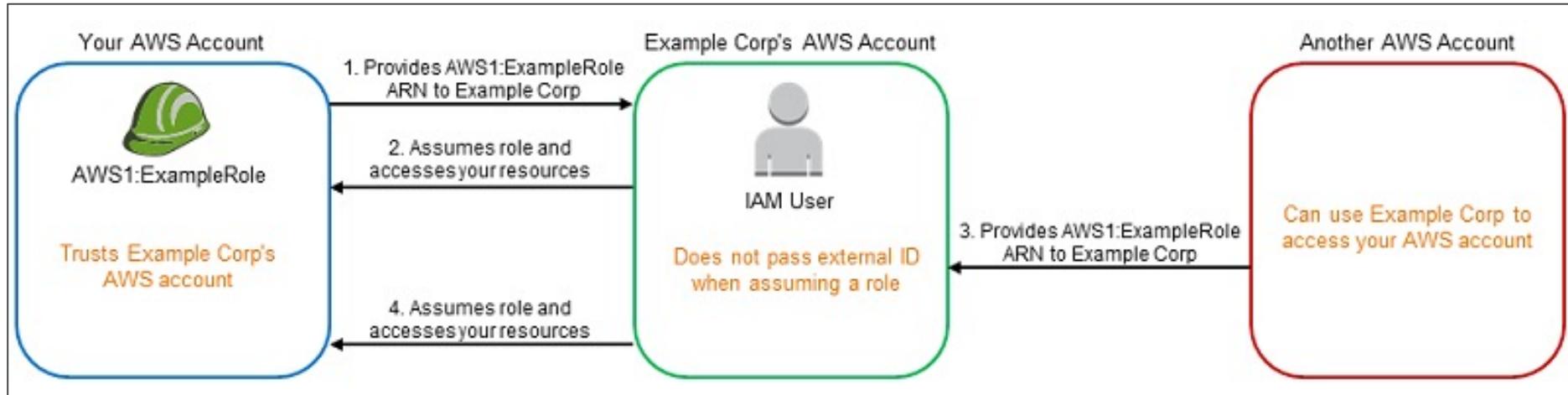
# Cross account access with STS



# Providing Access to AWS Accounts Owned by Third Parties

- Zone of trust = accounts, organizations that you own
- Outside Zone of Trust = 3<sup>rd</sup> parties
- Use IAM Access Analyzer to find out which resources are exposed
- For granting access to a 3<sup>rd</sup> party:
  - The 3<sup>rd</sup> party AWS account ID
  - An **External ID** (secret between you and the 3<sup>rd</sup> party)
    - To uniquely associate with the role between you and 3<sup>rd</sup> party
    - Must be provided when defining the trust and when assuming the role
    - Must be chosen by the 3<sup>rd</sup> party
  - Define permissions in the IAM policy

# The confused deputy

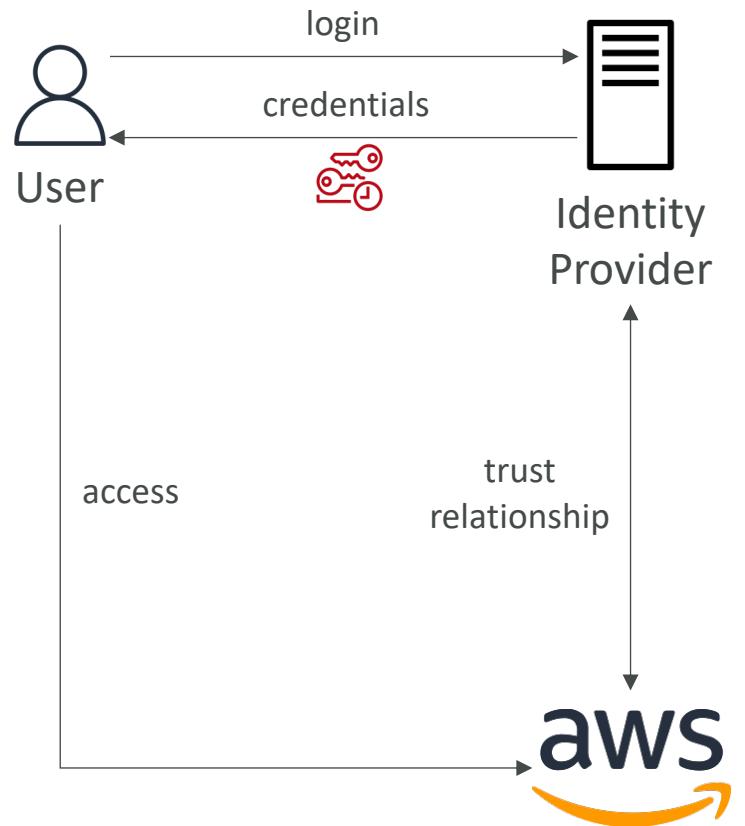


# STS Important APIs

- **AssumeRole**: access a role within your account or cross-account
- **AssumeRoleWithSAML**: return credentials for users logged with SAML
- **AssumeRoleWithWebIdentity**: return creds for users logged with an IdP
  - Example providers include Amazon Cognito, Login with Amazon, Facebook, Google, or any OpenID Connect-compatible identity provider
  - AWS recommends using Cognito instead
- **GetSessionToken**: for MFA, from a user or AWS account root user
- **GetFederationToken**: obtain temporary creds for a federated user, usually a proxy app that will give the creds to a distributed app inside a corporate network

# Identity Federation in AWS

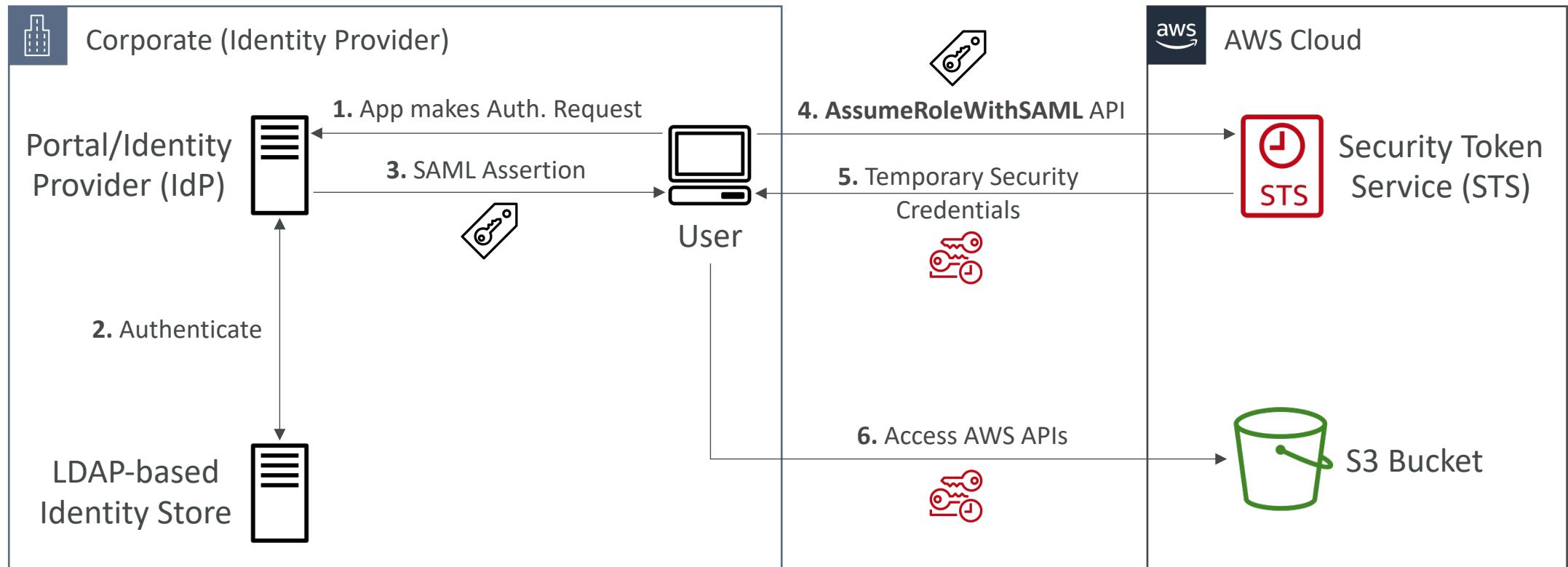
- Give users outside of AWS permissions to access AWS resources in your account
- You don't need to create IAM Users (user management is outside AWS)
- Use cases:
  - A corporate has its own identity system (e.g., Active Directory)
  - Web/Mobile application that needs access to AWS resources
- Identity Federation can have many flavors:
  - SAML 2.0
  - Custom Identity Broker
  - Web Identity Federation With(out) Amazon Cognito
  - Single Sign-On (SSO)



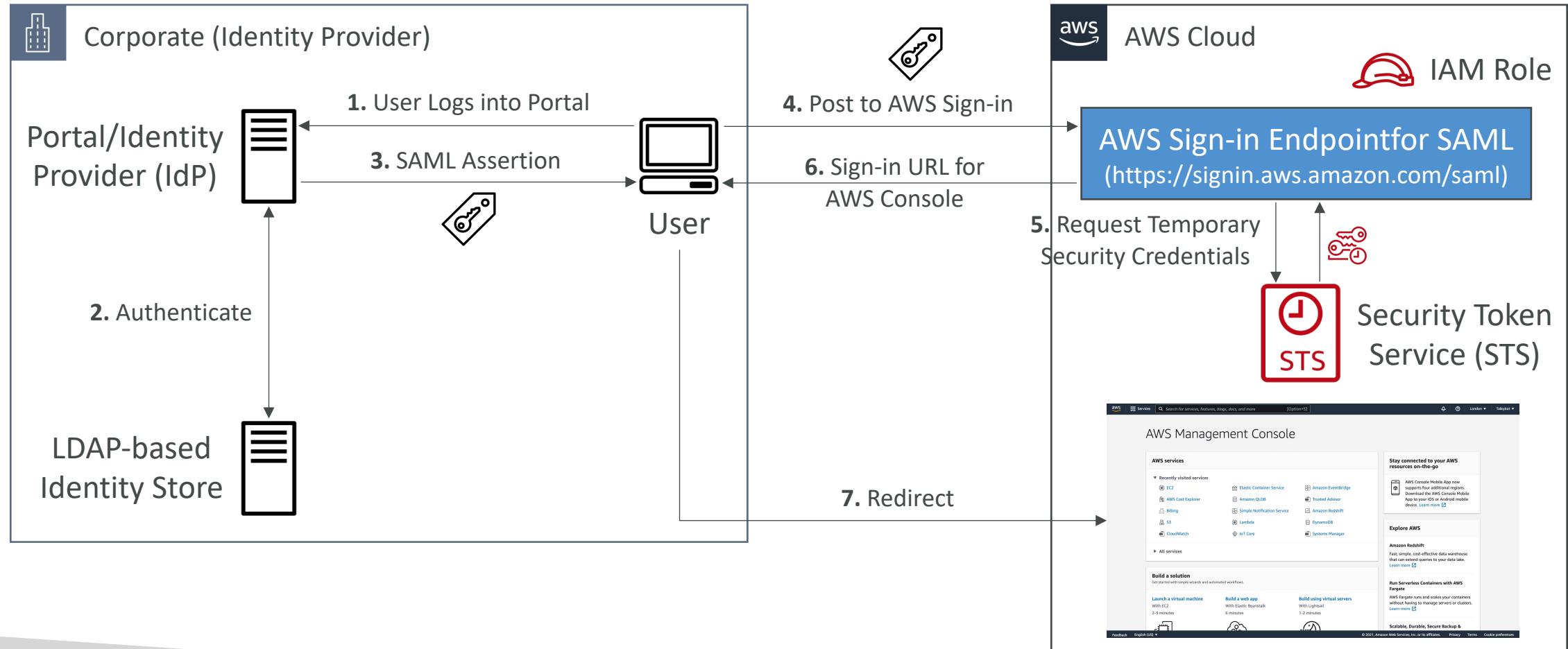
# SAML 2.0 Federation

- Security Assertion Markup Language 2.0 (SAML 2.0)
- Open standard used by many identity providers (e.g., ADFS)
  - Supports integration with Microsoft Active Directory Federations Services (ADFS)
  - Or any SAML 2.0–compatible IdPs with AWS
- Access to AWS Console, AWS CLI, or AWS API using temporary credentials
  - No need to create IAM Users for each of your employees
  - Need to setup a trust between AWS IAM and SAML 2.0 Identity Provider (both ways)
- Under-the-hood: Uses the STS API `AssumeRoleWithSAML`
- SAML 2.0 Federation is the “old way”, Amazon Single Sign-On (AWS SSO) Federation is the new managed and simpler way
  - <https://aws.amazon.com/blogs/security/enabling-federation-to-aws-using-windows-active-directory-adfs-and-saml-2-0/>

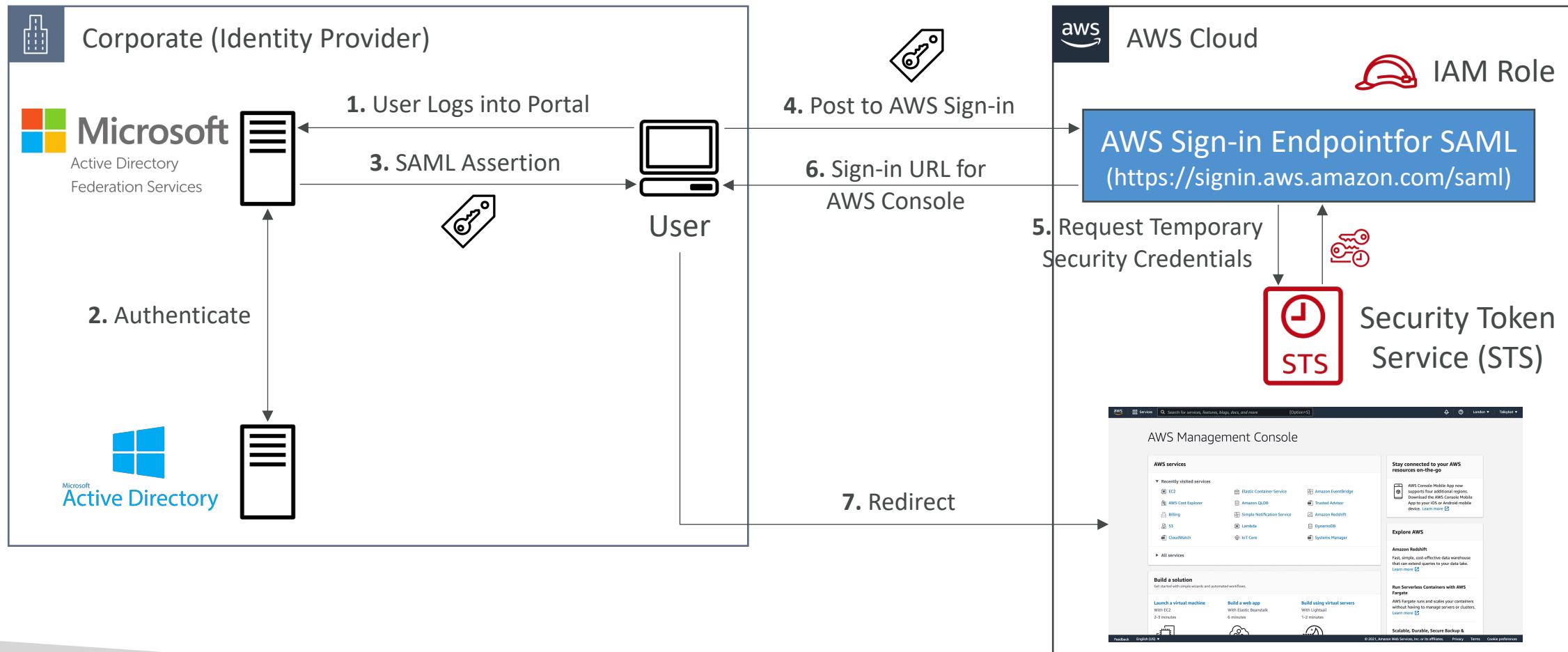
# SAML 2.0 Federation – AWS API Access



# SAML 2.0 Federation – AWS Console Access

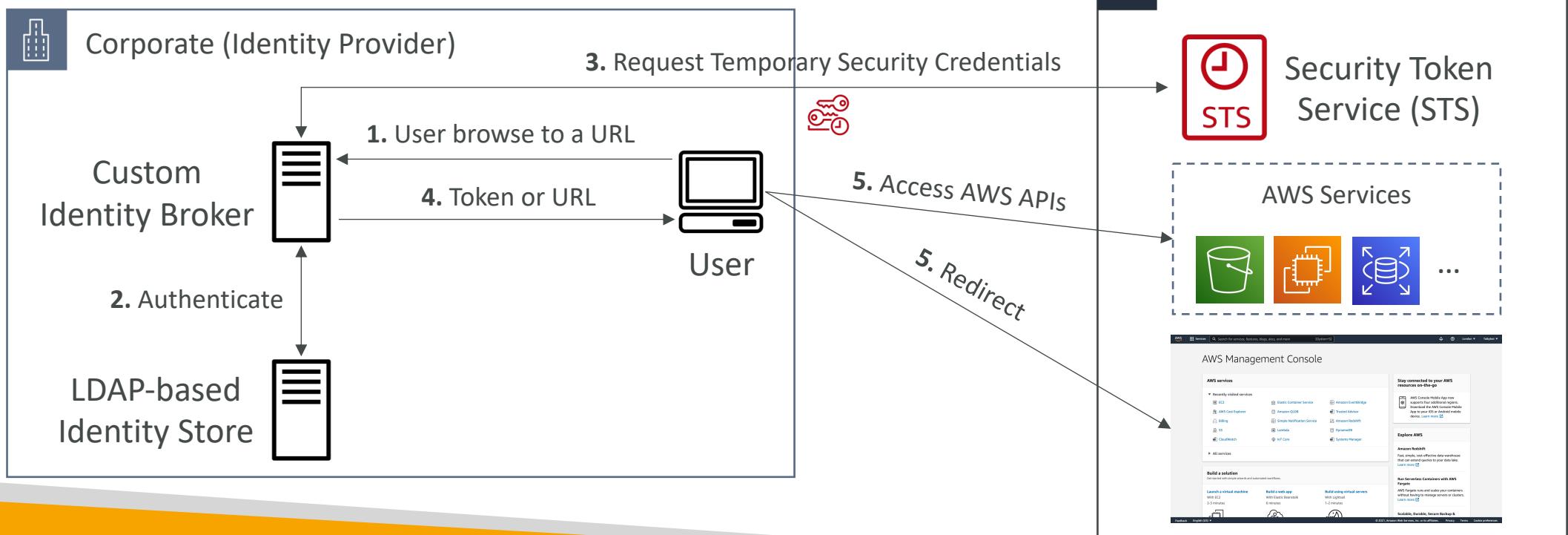


# SAML 2.0 Federation – Active Directory FS (ADFS)



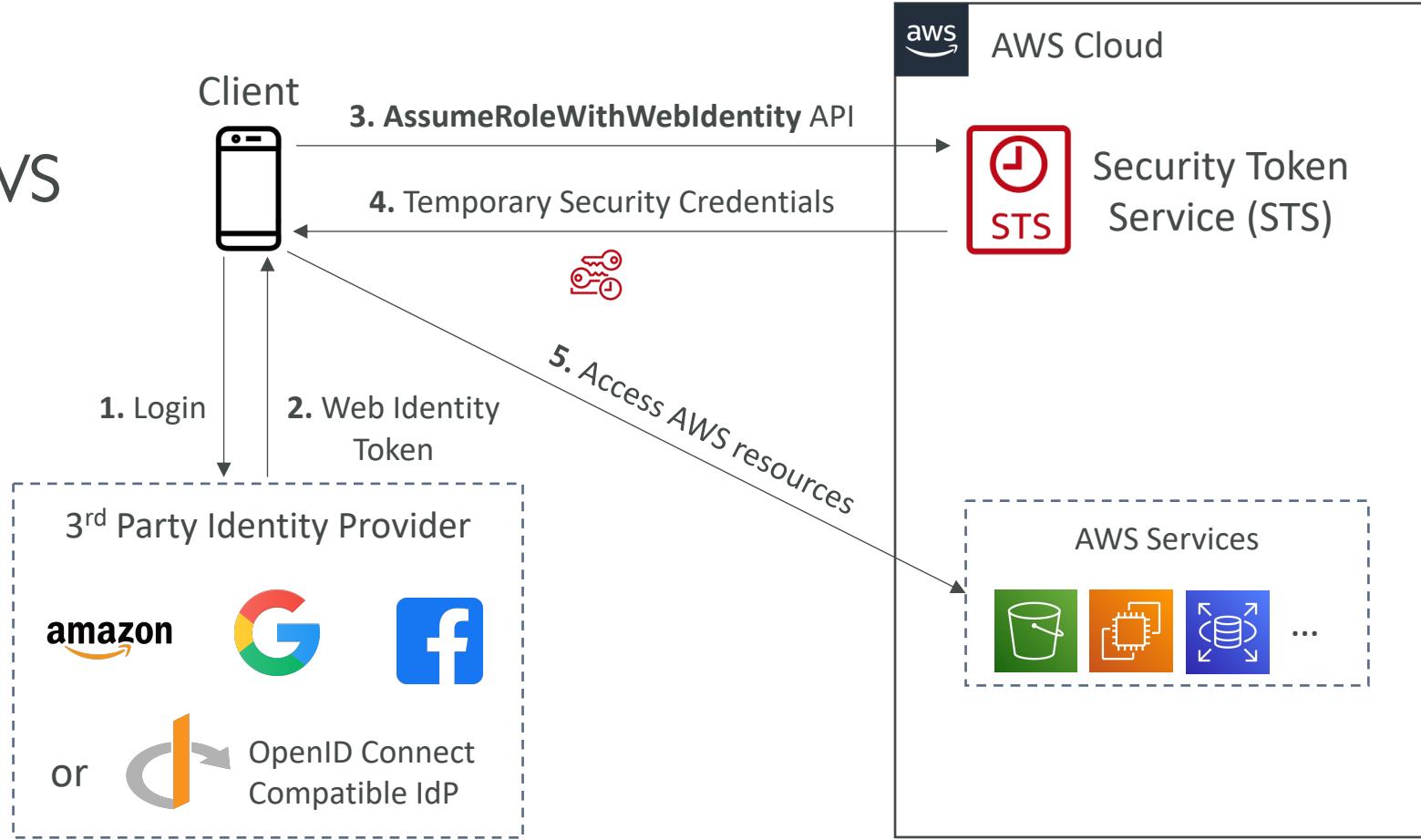
# Custom Identity Broker Application

- Use only if Identity Provider is **NOT** compatible with SAML 2.0
- The Identity Broker Authenticates users & requests temporary credentials from AWS
- The Identity Broker must determine the appropriate IAM Role
- Uses the STS API AssumeRole or GetFederationToken



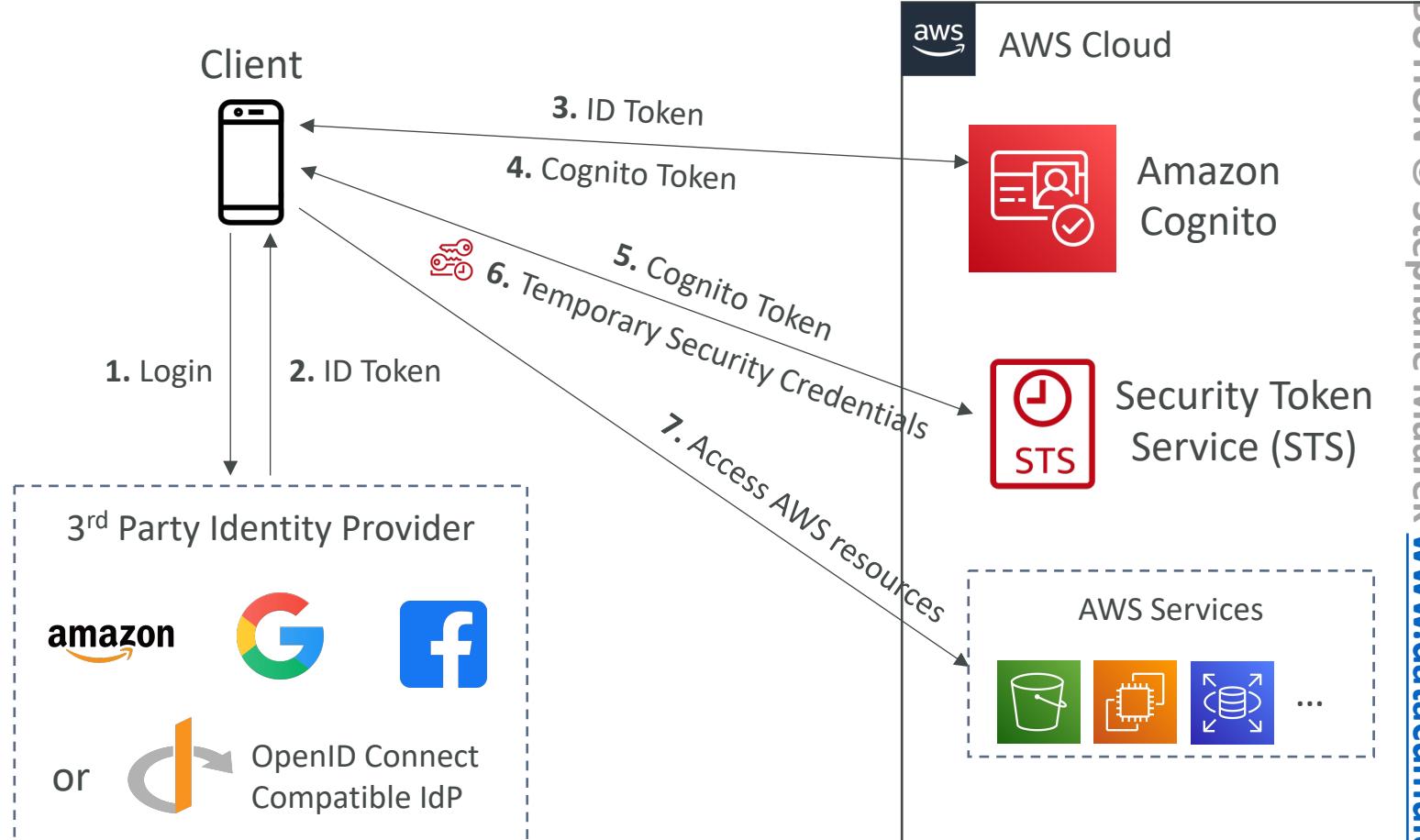
# Web Identity Federation – Without Cognito

- Not recommended by AWS  
– use Cognito instead



# Web Identity Federation – With Cognito

- Preferred over Web Identity Federation
  - Create IAM Roles using Cognito with the least privilege needed
  - Build trust between the OIDC IdP and AWS
- Cognito benefits:
  - Supports anonymous users
  - Supports MFA
  - Data Synchronization
- Cognito replaces a Token Vending Machine (TVM)



# Web Identity Federation – IAM Policy

- After being authenticated with Web Identity Federation, you can identify the user with an IAM policy variable

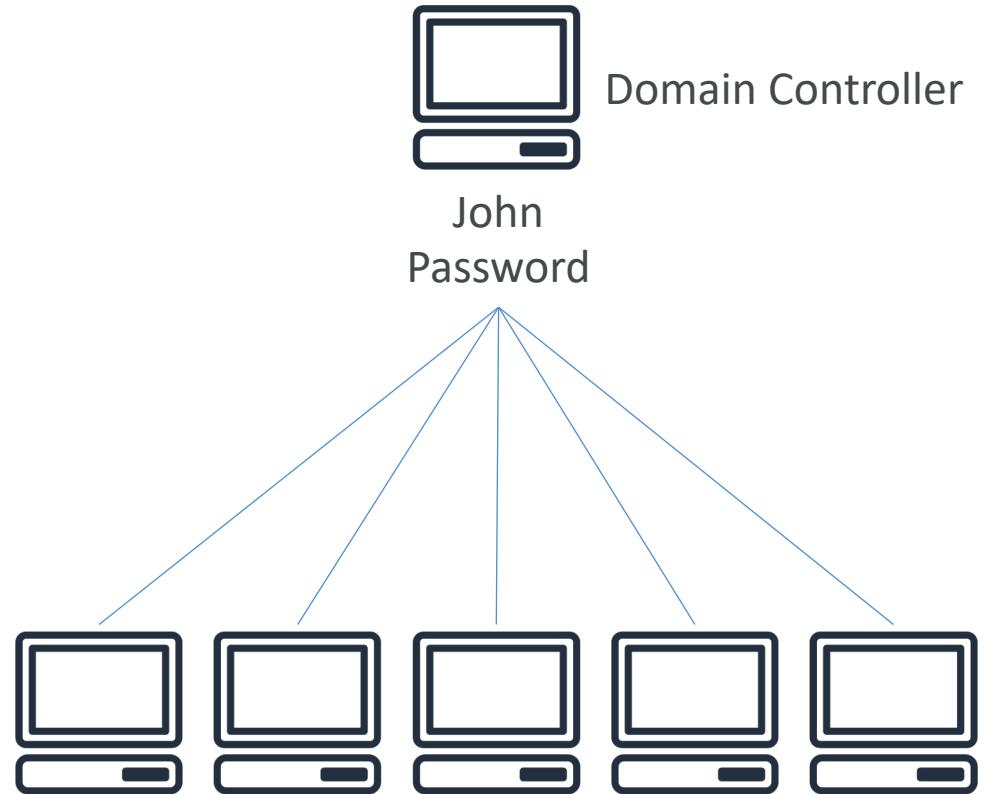
- Examples:

- cognito-identity.amazonaws.com:sub
- www.amazon.com:user\_id
- graph.facebook.com:id
- accounts.google.com:sub

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "s3>ListBucket",  
            "Resource": "arn:aws:s3:::myBucket",  
            "Condition": {  
                "StringLike": {  
                    "s3:prefix": "Amazon/mynumbersgame/${www.amazon.com:user_id}/*"  
                }  
            }  
        }, {  
            "Effect": "Allow",  
            "Action": ["s3:GetObject", "s3:PutObject", "s3>DeleteObject"],  
            "Resource": [  
                "arn:aws:s3:::myBucket/Amazon/mynumbersgame/${www.amazon.com:user_id}",  
                "arn:aws:s3:::myBucket/Amazon/mynumbersgame/${www.amazon.com:user_id}/*"  
            ]  
        }  
    ]  
}
```

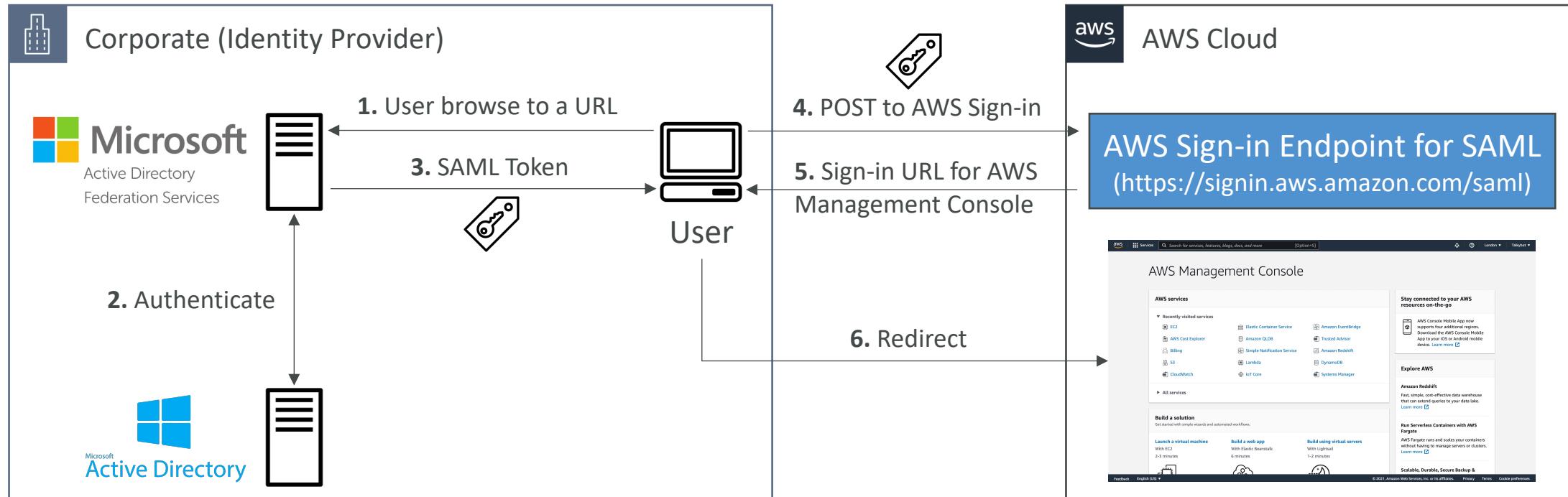
# What is Microsoft Active Directory (AD)?

- Found on any Windows Server with AD Domain Services
- Database of **objects**: User Accounts, Computers, Printers, File Shares, Security Groups
- Centralized security management, create account, assign permissions
- Objects are organized in **trees**
- A group of trees is a **forest**



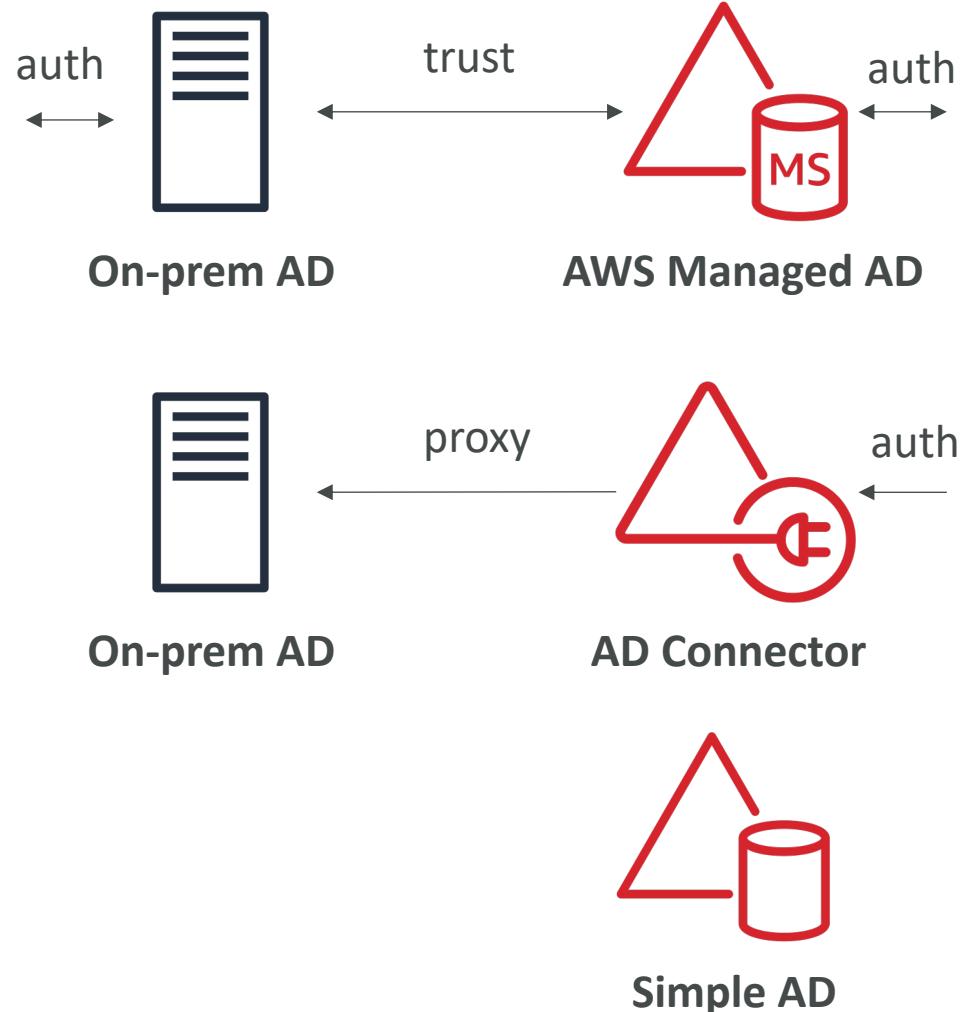
# What is ADFS (AD Federation Services)?

- ADFS provides Single Sign-On across applications
- SAML across 3<sup>rd</sup> party: AWS Console, Dropbox, Office365, etc...



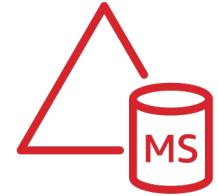
# AWS Directory Services

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

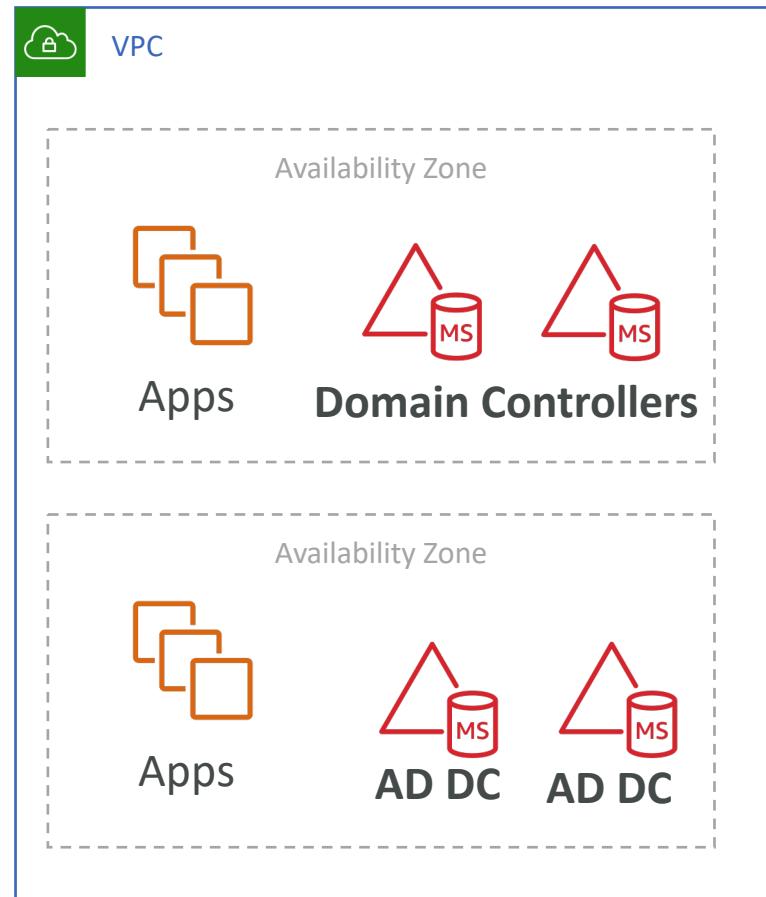


# AWS Directory Services

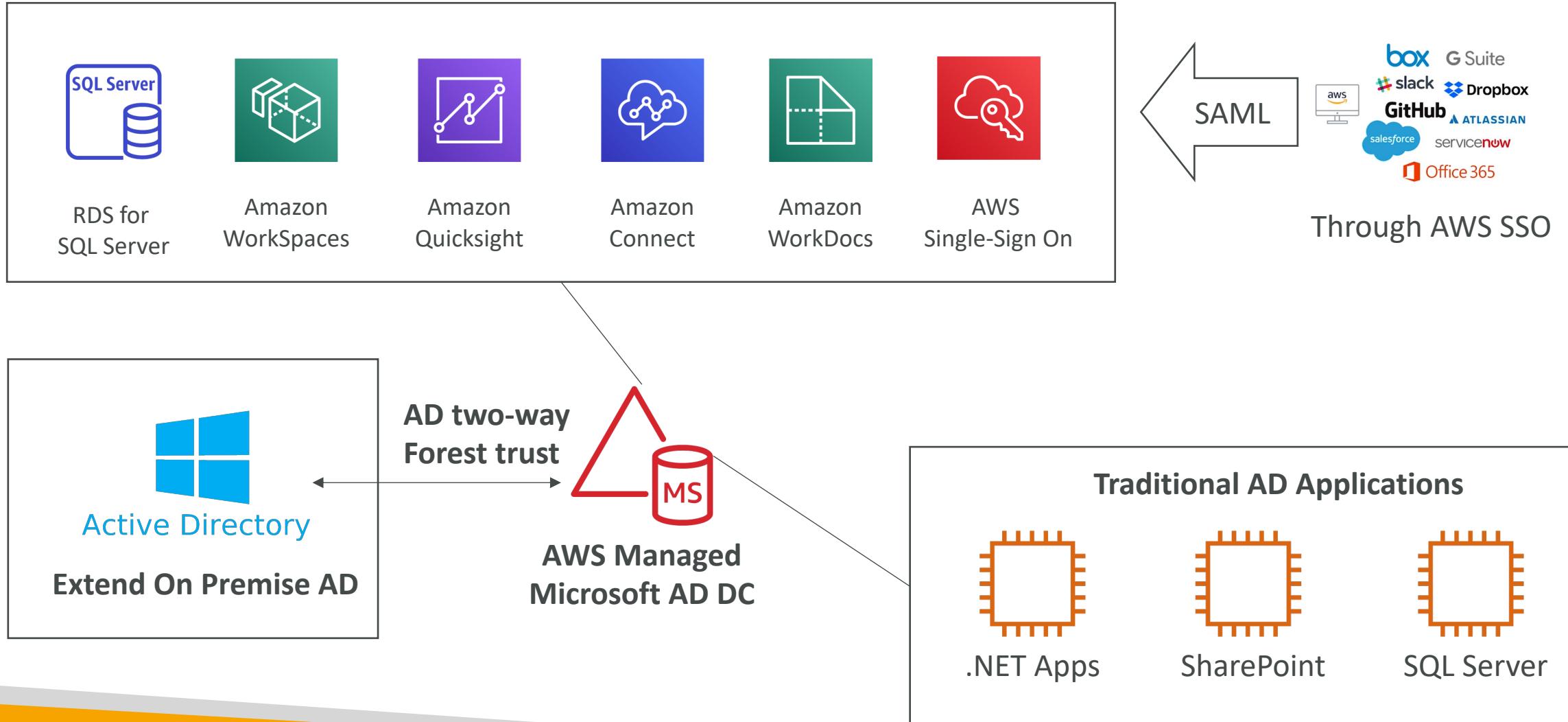
## AWS Managed Microsoft AD



- Managed Service: Microsoft AD in your AWS VPC
- EC2 Windows Instances:
  - EC2 Windows instances can join the domain and run traditional AD applications (sharepoint, etc)
  - Seamlessly Domain Join Amazon EC2 Instances from Multiple Accounts & VPCs
- Integrations:
  - RDS for SQL Server, AWS Workspaces, Quicksight...
  - AWS SSO to provide access to 3<sup>rd</sup> party applications
- Standalone repository in AWS or joined to on-premises AD
- Multi AZ deployment of AD in 2 AZ, # of DC (Domain Controllers) can be increased for scaling
- Automated backups
- Automated Multi-Region replication of your directory

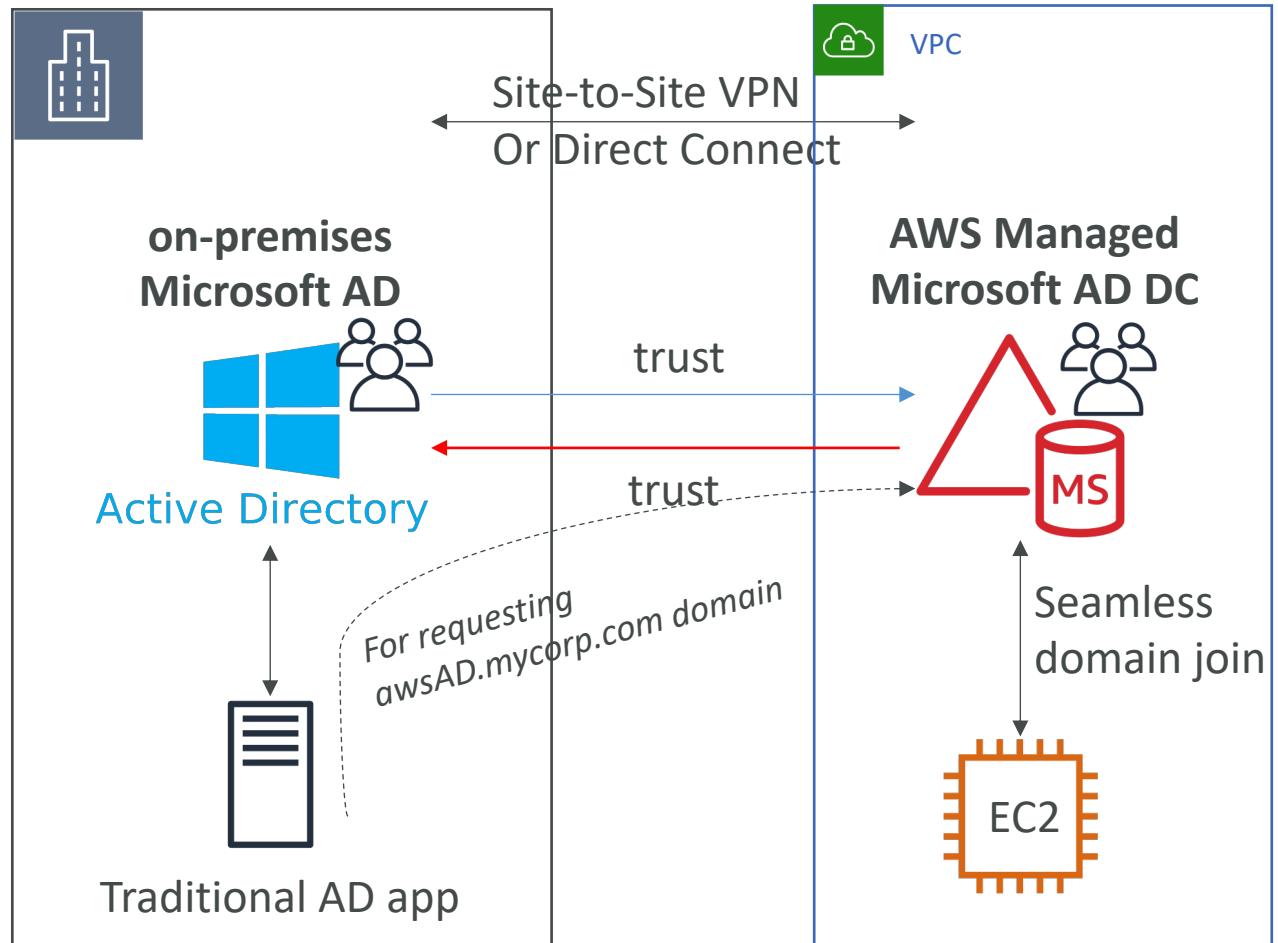


# AWS Microsoft Managed AD - Integrations



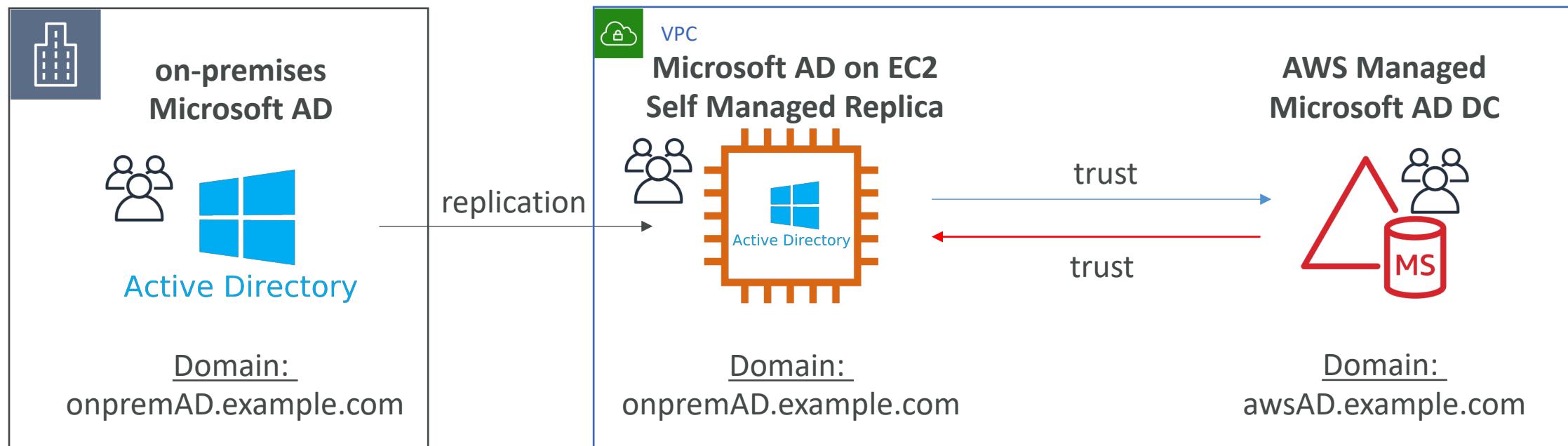
# Connect to on-premises AD

- Ability to connect your on-premises Active Directory to AWS Managed Microsoft AD
- Must establish a Direct Connect (DX) or VPN connection
- Can setup three kinds of forest trust:
  - One-way trust: AWS => on-premises
  - One-way trust: on-premises => AWS
  - Two-way forest trust: AWS ⇄ on-premises
- Forest trust is different than synchronization



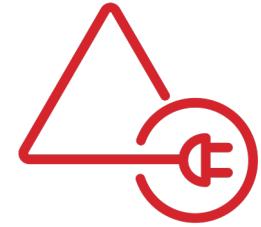
# Solution Architecture: Active Directory Replication

- You may want to create a replica of your AD on EC2 in the cloud to minimize latency of in case DX or VPN goes down
- Establish trust between the AWS Managed Microsoft AD and EC2

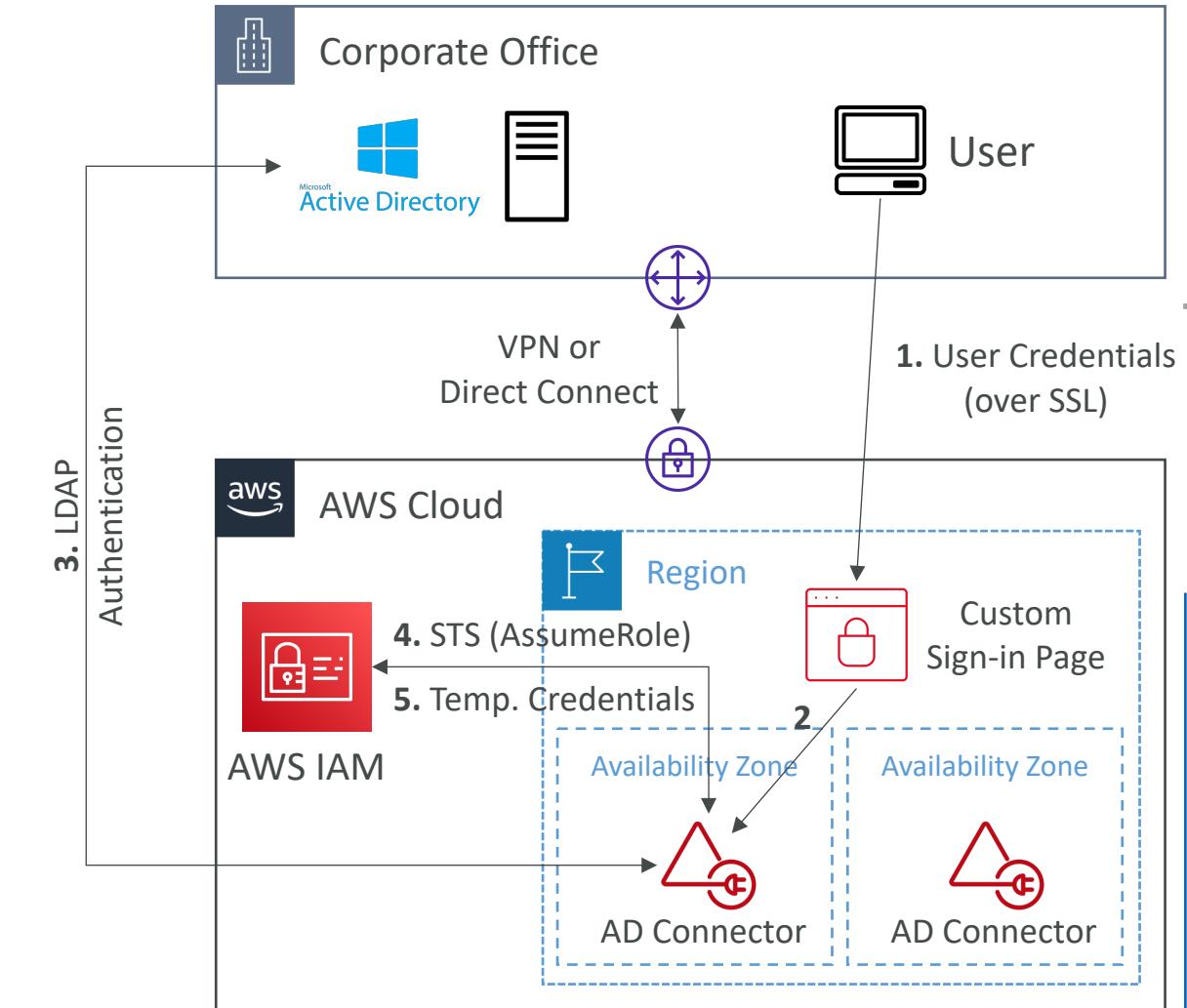


# AWS Directory Services

## AD Connector

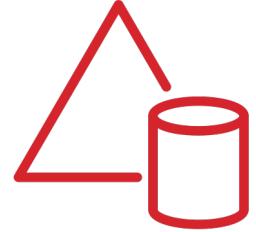


- AD Connector is a directory gateway to redirect directory requests to your on-premises Microsoft Active Directory
- No caching capability
- Manage users solely on-premises, no possibility of setting up a trust
- VPN or Direct Connect
- Doesn't work with SQL Server, doesn't do seamless joining, can't share directory



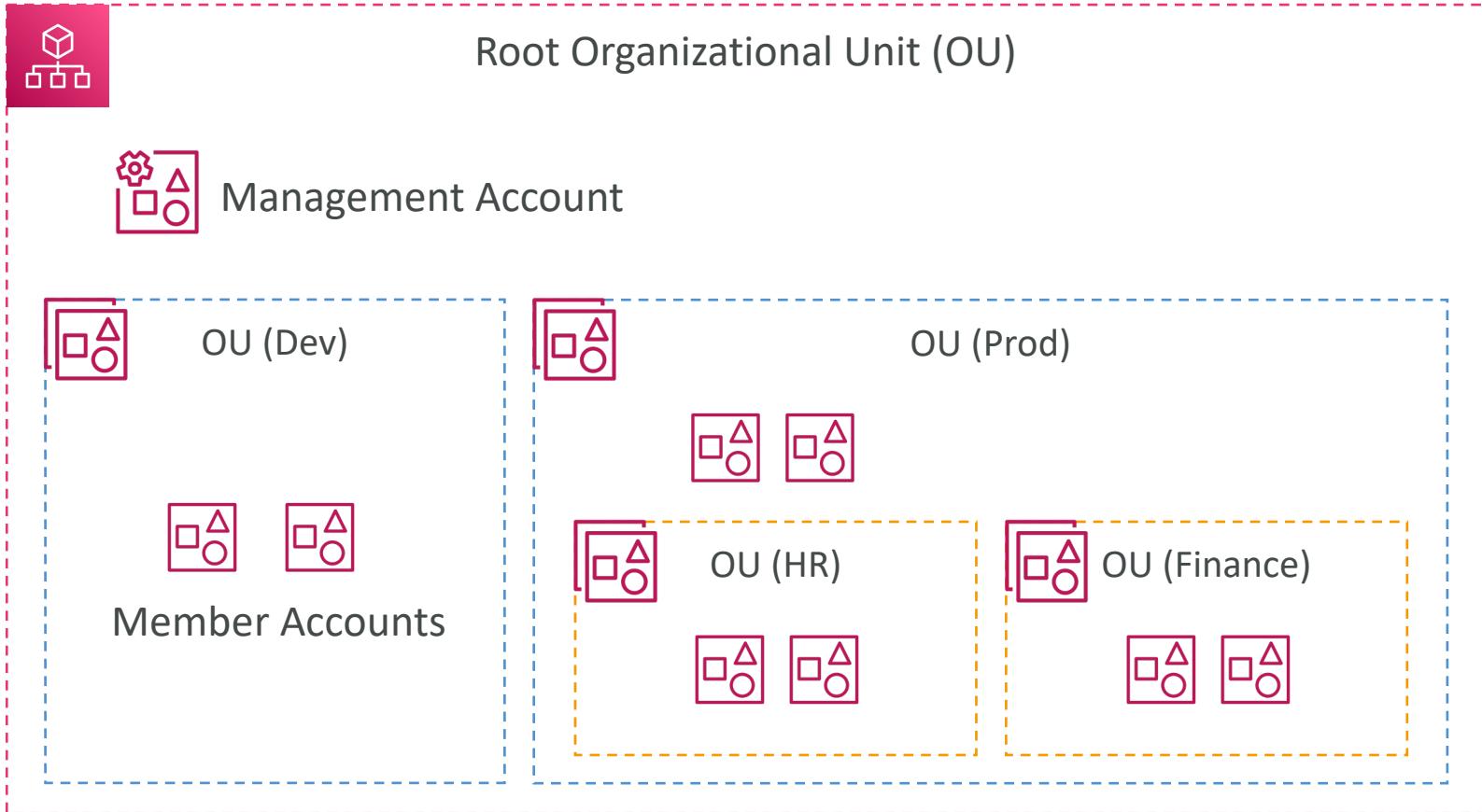
# AWS Directory Services

## Simple AD



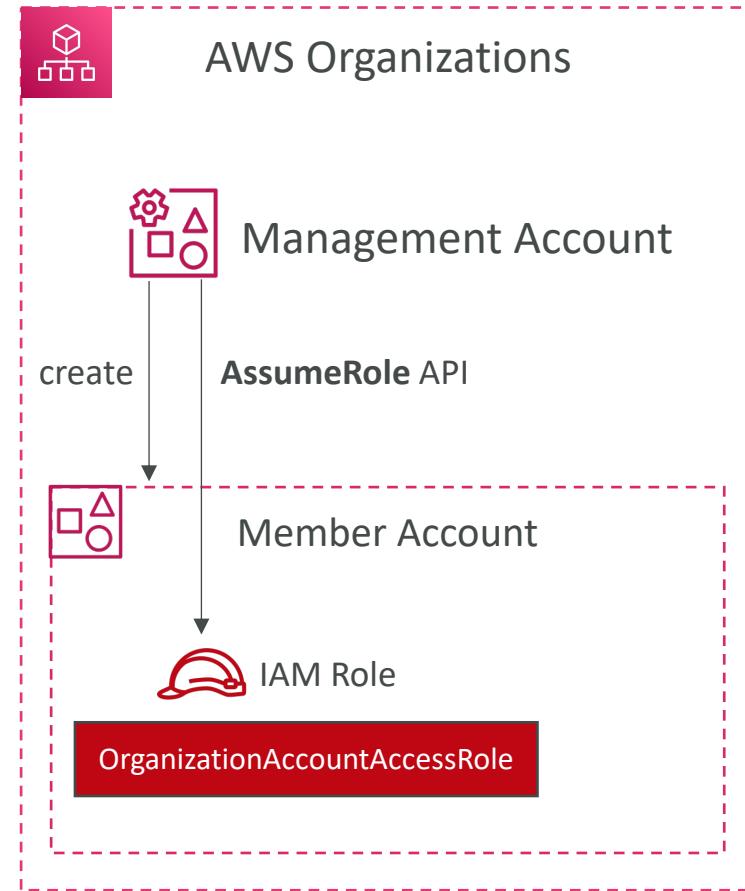
- Simple AD is an inexpensive Active Directory–compatible service with the common directory features.
- Supports joining EC2 instances, manage users and groups
- Does not support MFA, RDS SQL server, AWS SSO
- Small: 500 users, large: 5000 users
- Powered by Samba 4, compatible with Microsoft AD
- lower cost, low scale, basic AD compatible, or LDAP compatibility
- No trust relationship

# AWS Organizations



# AWS Organizations - OrganizationAccountAccessRole

- IAM role which grants full administrator permissions in the Member account to the Management account
- Used to perform admin tasks in the Member accounts (e.g., creating IAM users)
- Could be assumed by IAM users in the Management account
- Automatically added to all new Member accounts created with AWS Organizations
- Must be created manually if you invite an existing Member account

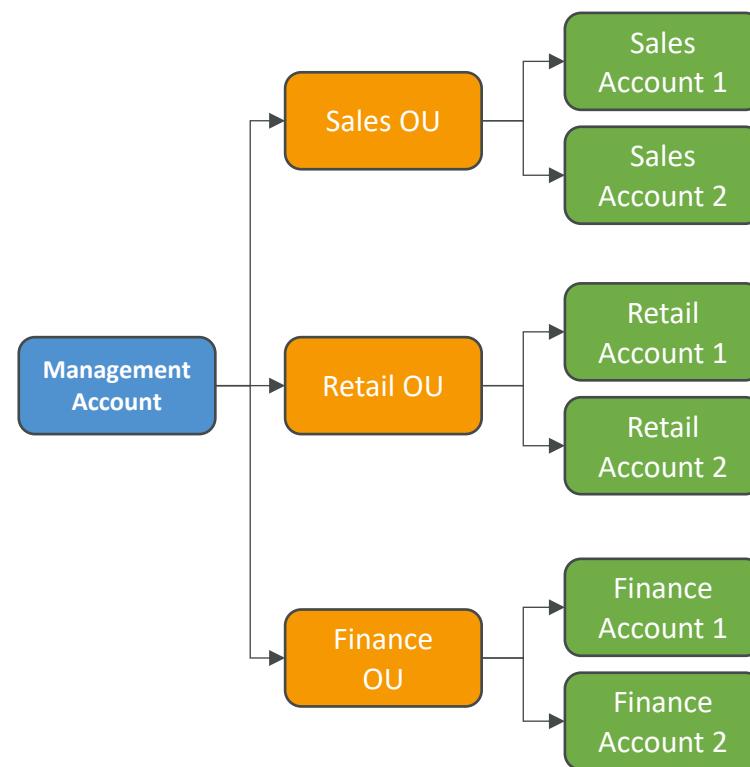


# Multi Account Strategies

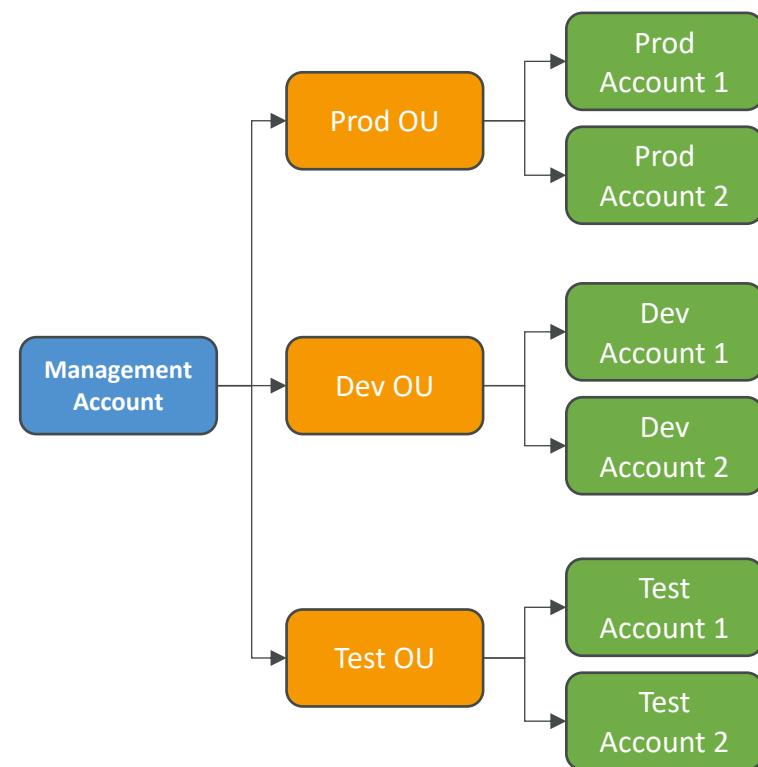
- Create accounts per **department**, per **cost center**, per **dev / test / prod**, based on **regulatory restrictions** (using SCP), for better resource **isolation** (ex:VPC), to have separate per-account service limits, isolated account for **logging**,
- Multi Account vs. One Account Multi VPC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

# Organizational Units (OU) - Examples

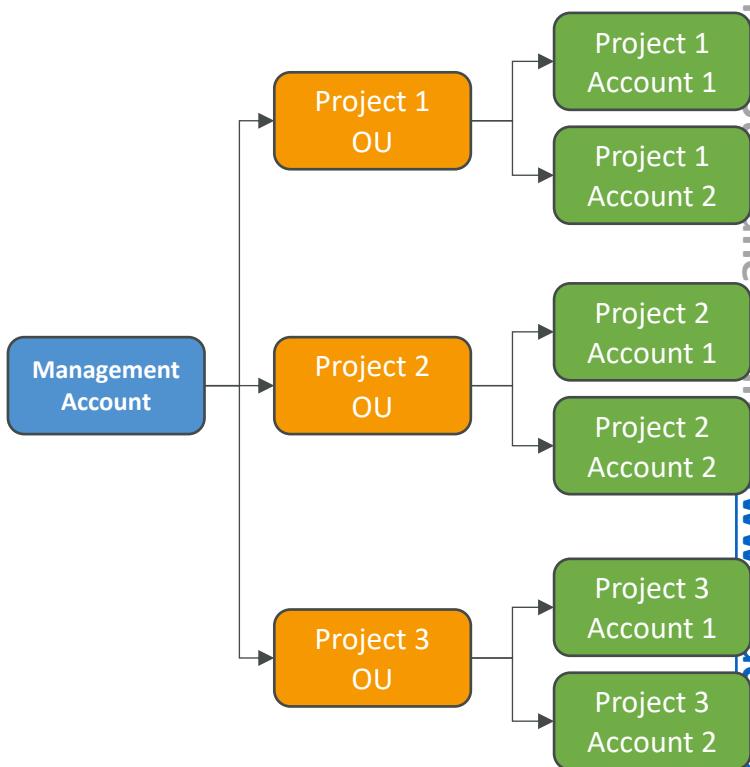
## Business Unit



## Environmental Lifecycle



## Project-Based



# AWS Organization - Feature Modes

- Consolidated billing features:
  - Consolidated Billing across all accounts - single payment method
  - Pricing benefits from aggregated usage (volume discount for EC2, S3...)
- All Features (Default):
  - Includes consolidated billing features, SCP
  - Invited accounts must approve enabling all features
  - Ability to apply an SCP to prevent member accounts from leaving the org
  - Can't switch back to Consolidated Billing Features only

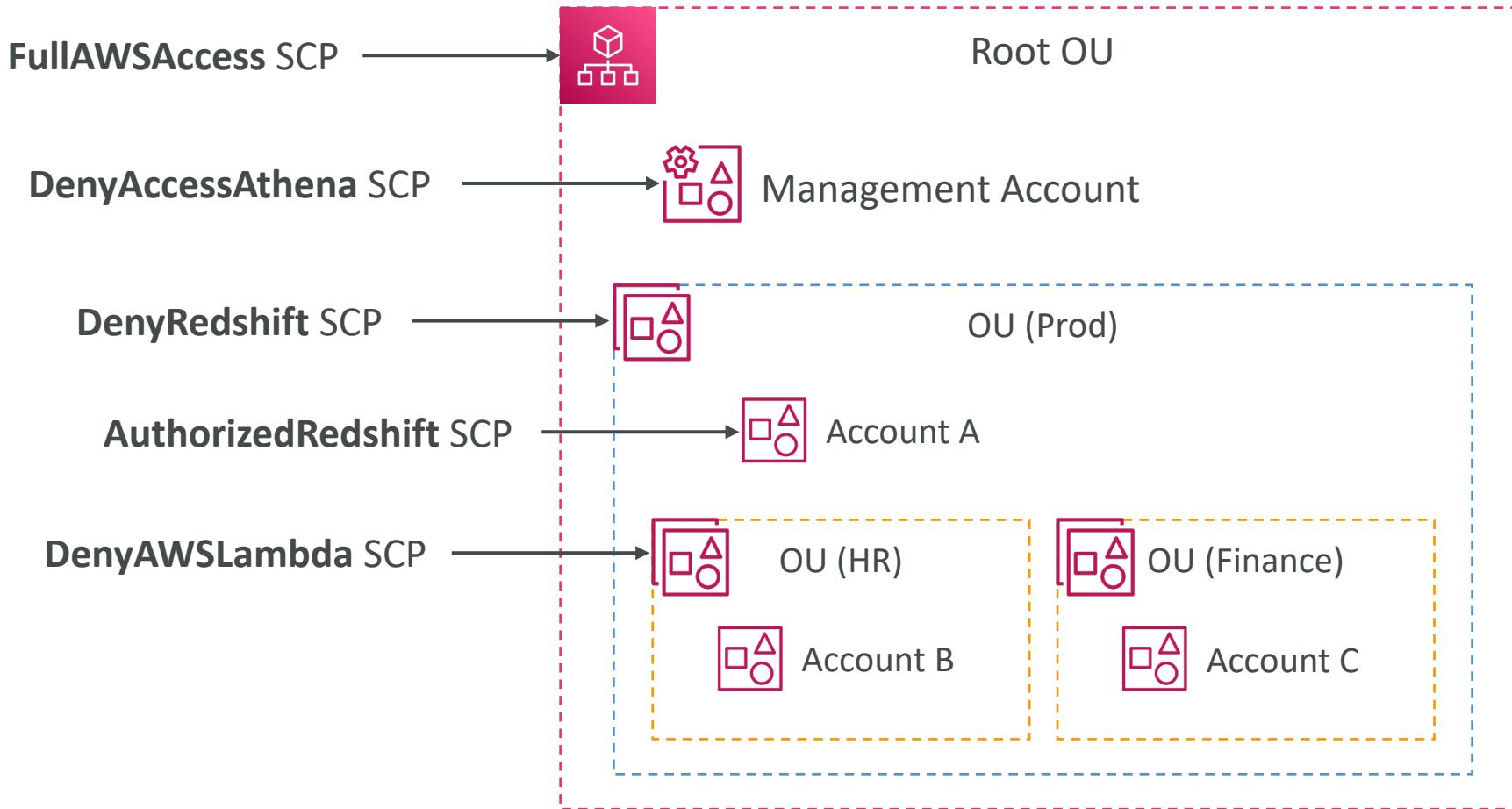
# AWS Organizations – Reserved Instances

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

# Service Control Policies (SCP)

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

# SCP Hierarchy



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

# SCP Examples

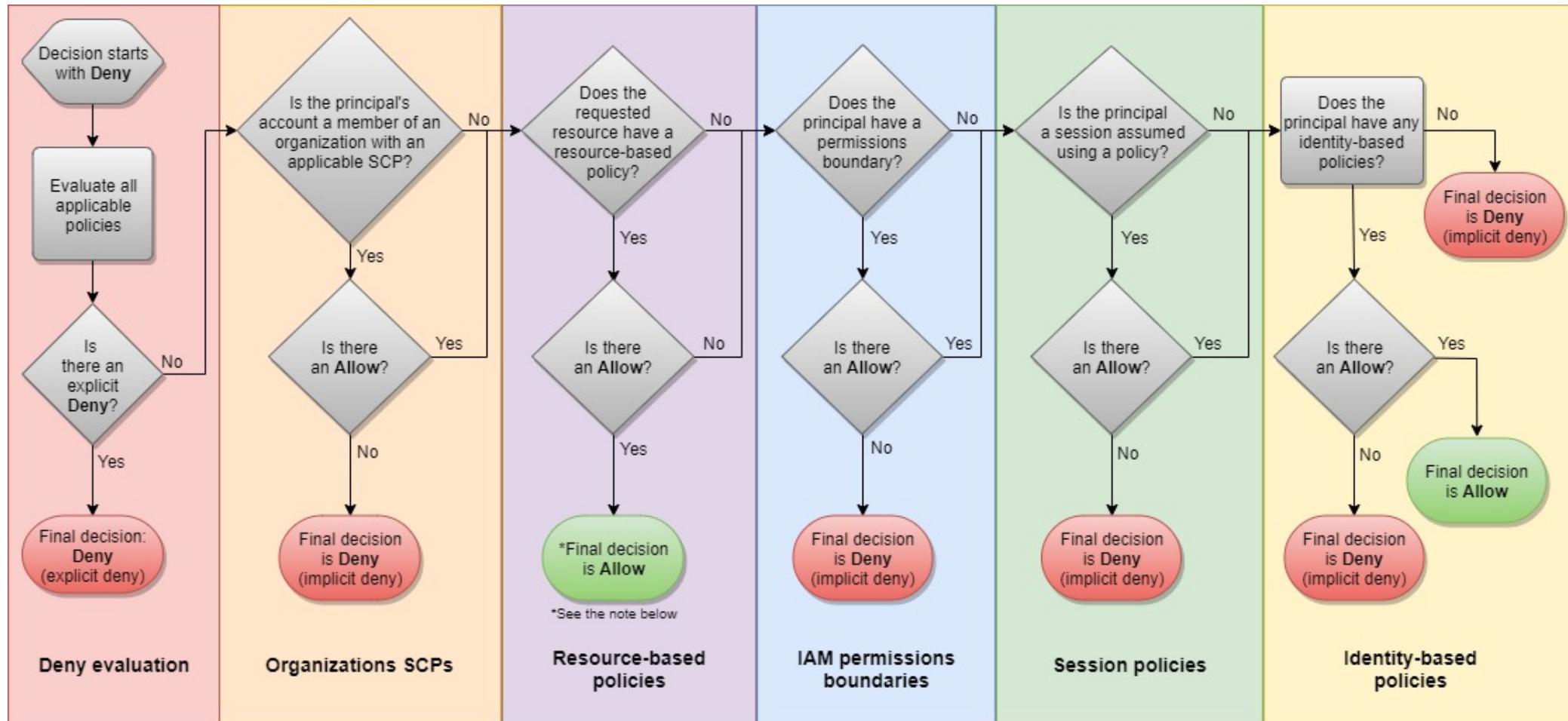
## Blocklist and Allowlist strategies

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

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

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

# IAM Policy Evaluation Logic



[https://docs.aws.amazon.com/IAM/latest/UserGuide/reference\\_policies\\_evaluation-logic.html](https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_evaluation-logic.html)

# Restricting Tags with IAM Policies

- You can restrict specific Tags on AWS resources
- Using the **aws:TagKeys** Condition Key
  - Validate the Tag Keys attached to a resource against the Tag Keys in the IAM Policy
- Example: allow IAM users to create EBS Volumes only if it has the “Env” and “CostCenter” Tags
- Use either **ForAllValues** (must have all keys) or **ForAnyValue** (must have any of these keys at a minimum)

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "ec2:CreateVolume",  
            "Resource": "arn:aws:ec2:*::volume/*",  
            "Condition": {  
                "ForAllValues:StringEquals": {  
                    "aws:TagKeys": ["Env", "CostCenter"]  
                }  
            }  
        }  
    ]  
}
```

Match All Keys

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": "ec2:CreateVolume",  
            "Resource": "arn:aws:ec2:*::volume/*",  
            "Condition": {  
                "ForAnyValue:StringEquals": {  
                    "aws:TagKeys": ["Env", "CostCenter"]  
                }  
            }  
        }  
    ]  
}
```

Match Any Keys

# Using SCP to Restrict Creating Resources without appropriate Tags

- Prevent IAM Users/Roles in the affected Member accounts from creating resources if they don't have a specific Tags
- Example: restrict launching an EC2 instance if it doesn't have the “Project” and “CostCenter” Tags

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

# AWS Organizations – Tag Policies

- Helps you standardize tags across resources in an AWS Organization
- Ensure consistent tags, audit tagged resources, maintain proper resources categorization, ...
- You define Tag keys and their allowed values
- Helps with AWS Cost Allocation Tags and Attribute-based Access Control
- Prevent any non-compliant tagging operations on specified services and resources
- Generate a report that lists all tagged/non-compliant resources
- Use CloudWatch Events to monitor non-compliant tags

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

# AWS Organizations – AI Services Opt-out Policies

- Certain AWS AI services may use your content for continuous improvement of Amazon AI/ML services
- Example: Amazon Lex, Amazon Comprehend, Amazon Polly, ...
- You can opt-out of having your content stored or used by AWS AI services
- Create an Opt-out Policy that enforces this setting across all Member accounts and AWS Regions
- You can opt-out all AI services or selected services
- Can be attached to Organization Root, specific OU, or individual Member account

```
{  
  "services": {  
    "default": {  
      "opt_out_policy": {  
        "@@assign": "optOut"  
      }  
    }  
  }  
}  
  
All Services
```

```
{  
  "services": {  
    "rekognition": {  
      "opt_out_policy": {  
        "@@assign": "optOut"  
      }  
    },  
    "lex": {  
      "opt_out_policy": {  
        "@@assign": "optOut"  
      }  
    }  
  }  
}  
  
ONLY Rekognition & Lex
```

# AWS Organizations – Backup Policies

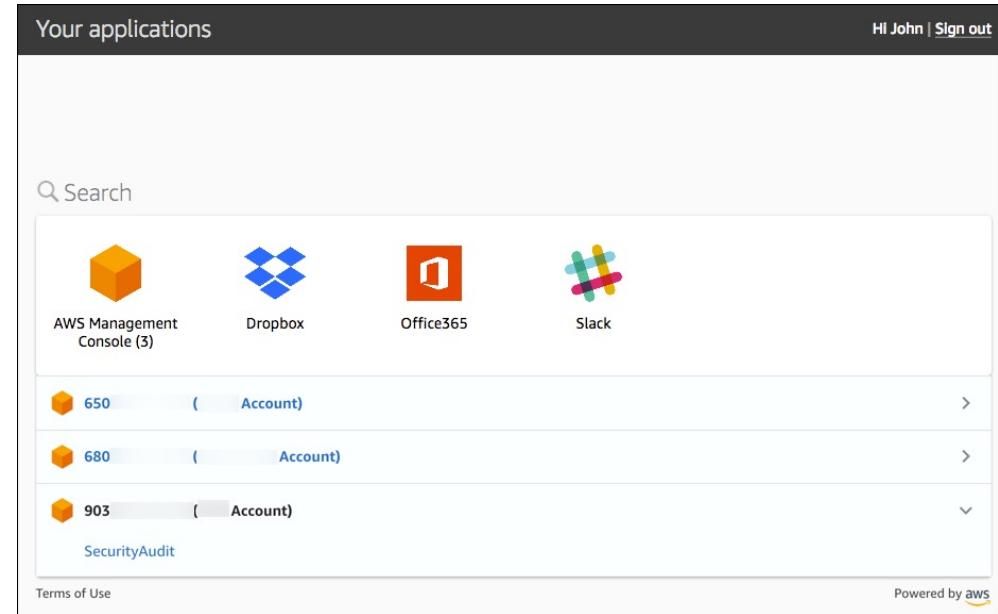
- AWS Backup enables you to create Backup Plans that define how to backup your AWS resources
- JSON documents that define Backup Plans across an AWS Organization
- Gives you granular control over backing up your resources (e.g., backup frequency, time window, backup region, ...)
- Can be attached to Organization Root, specific OU, or individual Member account
- **Immutable** Backup Plans appear in Member accounts (view ONLY)

```
{  
  "plans": {  
    "PII_Backup_Plan": {  
      "regions": {  
        "@@assign": ["us-east-1"]  
      },  
      "rules": {  
        "My_Hourly_Rule": {  
          "schedule_expression": { "@@assign": "cron(0 5 * * *)" },  
          "start_backup_window_minutes": { "@@assign": "60" },  
          "complete_backup_window_minutes": { "@@assign": "604800" },  
          "enable_continuous_backup": { "@@assign": false },  
          "target_backup_vault_name": { "@@assign": "My_Backup_Vault" },  
          "lifecycle": {  
            "move_to_cold_storage_after_days": { "@@assign": "180" },  
            "delete_after_days": { "@@assign": "270" }  
          }  
        }  
      }  
    }  
  }  
}
```

# AWS Single Sign-On (SSO)



- Centrally manage SSO access to:
  - Multiple AWS accounts
  - Commonly used business applications (e.g., Salesforce, Box, Office 365, ...)
  - Custom SAML 2.0-based applications
- Integrated with AWS Organizations
- Identity source:
  - SSO-built in: manage users & groups
  - Active Directory through Directory Services (AWS Managed Microsoft AD or AD Connector)
  - External Identity Provider: any SAML 2.0 Identity Provider (e.g., Azure AD, Okta Universal Directory)
- Centralized permission management
- Centralized auditing with CloudTrail (e.g., user sign-in activities)



Your applications

Hi John | Sign out

Search

Account	Actions
AWS Management Console (3)	>
Dropbox	>
Office365	>
Slack	>
SecurityAudit (650 accounts)	>
SecurityAudit (680 accounts)	>
SecurityAudit (903 accounts)	>

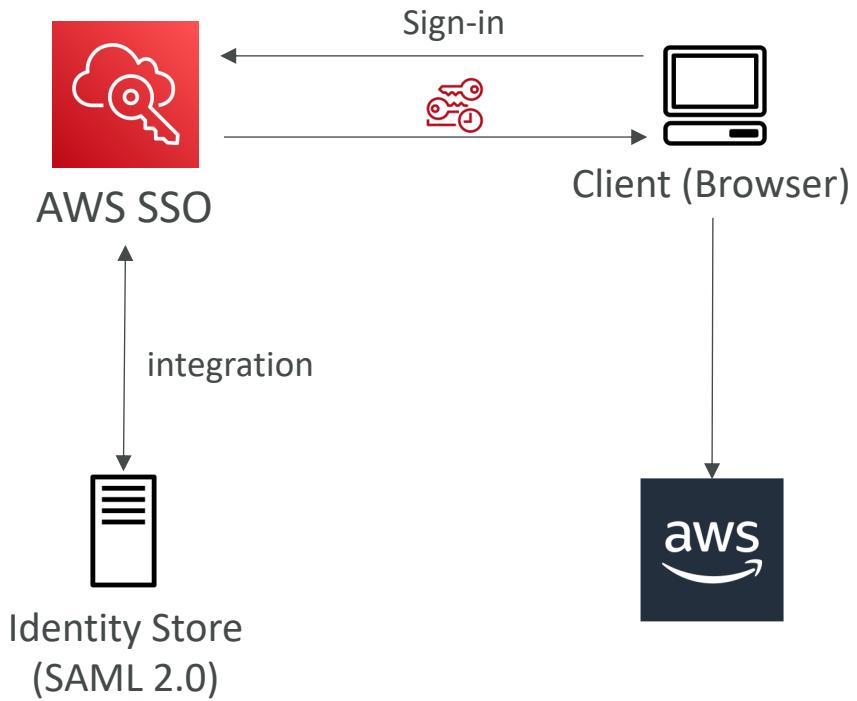
Terms of Use

Powered by aws

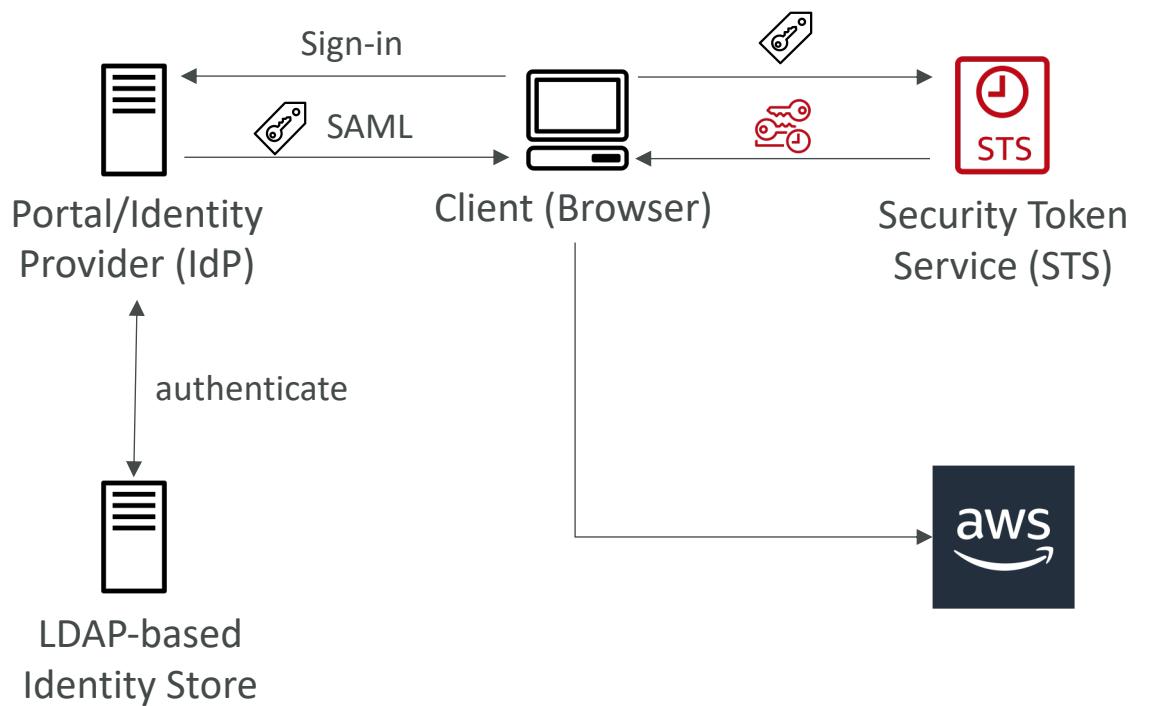
<https://aws.amazon.com/blogs/security/introducing-aws-single-sign-on/>

# SSO vs AssumeRoleWithSAML

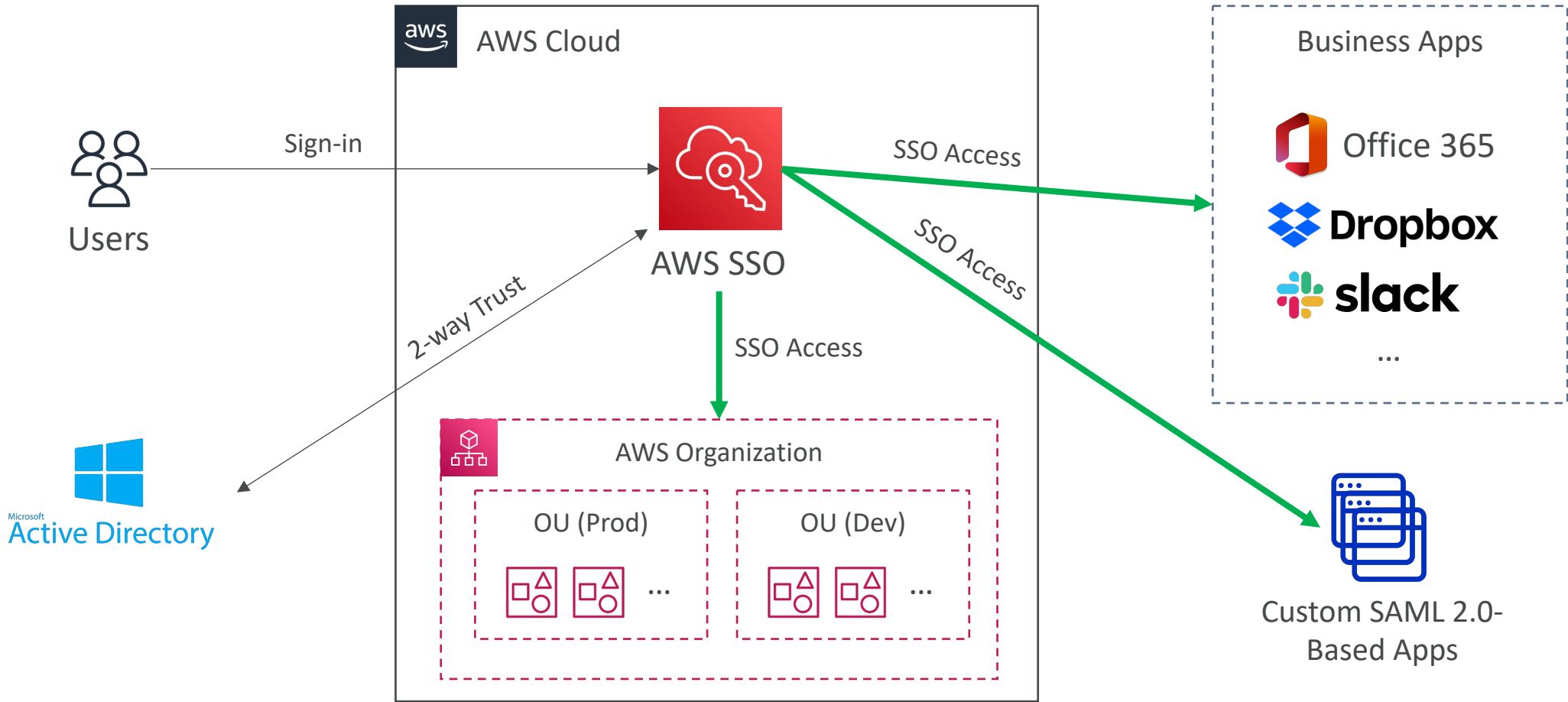
## AWS Single Sign-On



## AssumeRoleWithSAML

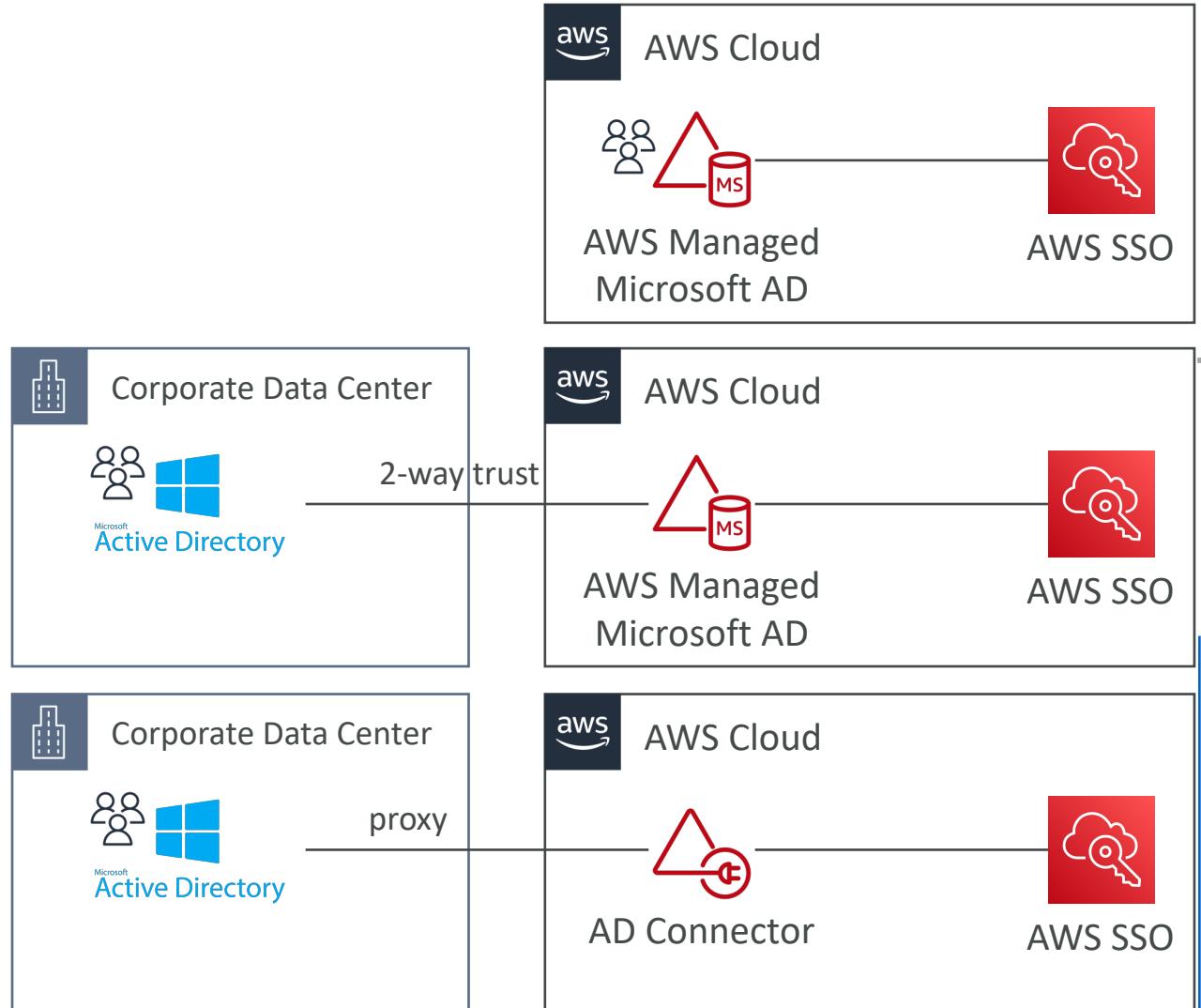


# AWS Single Sign-On (SSO) – With AD



# AWS SSO – Integration with MS AD

- AWS Managed Microsoft AD
- AWS Managed Microsoft AD with 2-way forest trust with on-premises AD
- AD Connector to on-premises AD



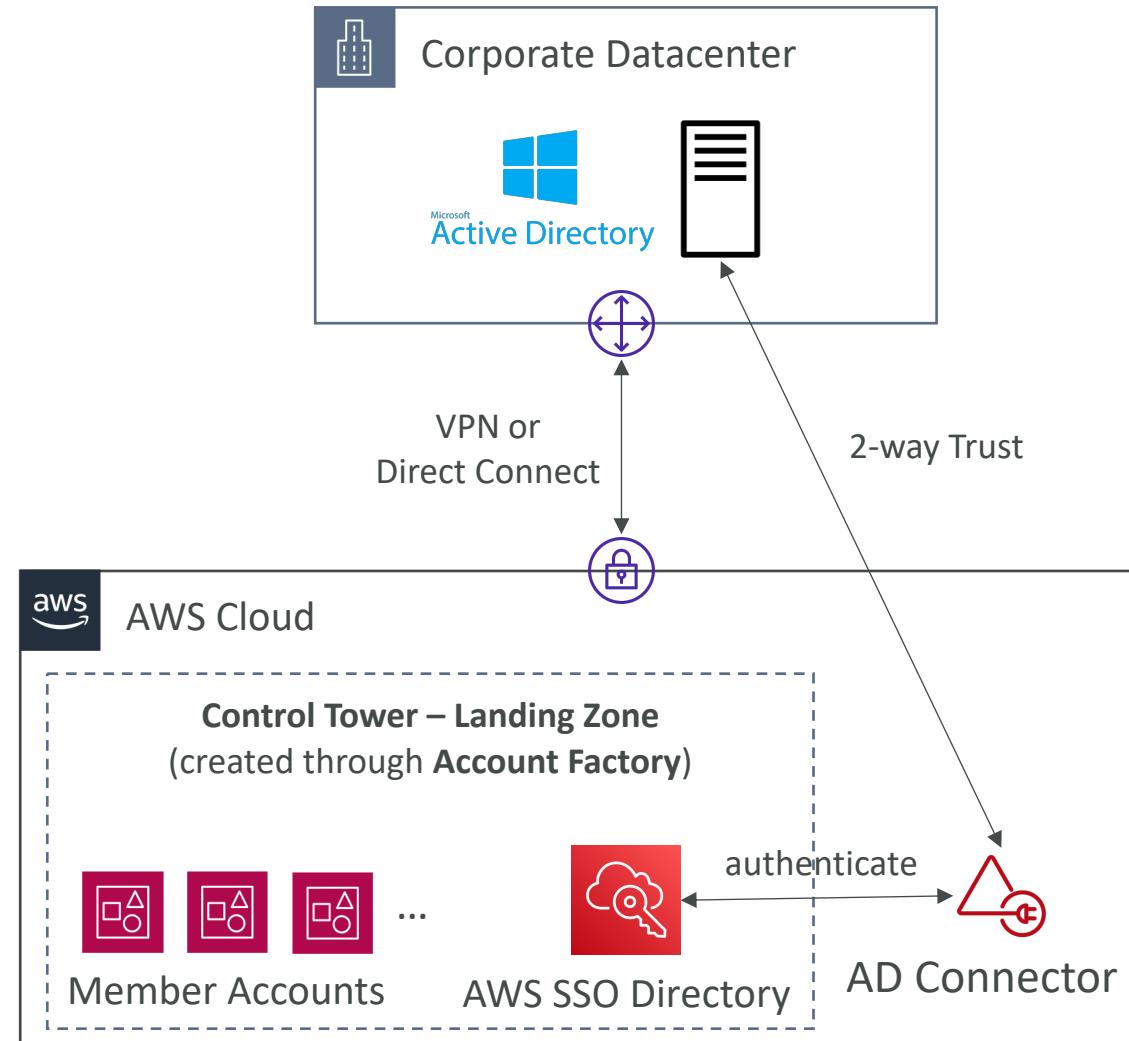
# AWS Control Tower



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

# AWS Control Tower – Account Factory

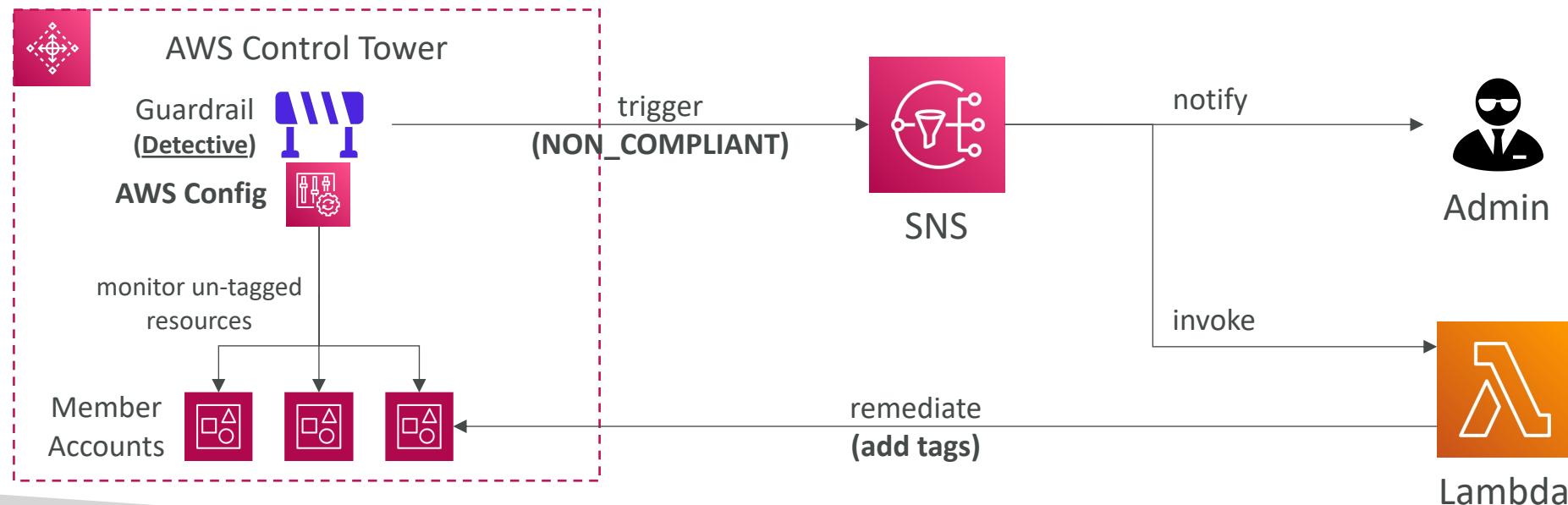
- Automates account provisioning and deployments
- Enables you to create pre-approved baselines and configuration options for AWS accounts in your organization (e.g., VPC default configuration, subnets, region, ...)
- Uses AWS Service Catalog to provision new AWS accounts



# AWS Control Tower – Detect and Remediate Policy Violations

- **Guardrail**

- Provides ongoing governance for your Control Tower environment (AWS Accounts)
- Preventive – using SCPs (e.g., Disallow Creation of Access Keys for the Root User)
- Detective – using AWS Config (e.g., Detect Whether MFA for the Root User is Enabled)
- Example: identify non-compliant resources (e.g., untagged resources)



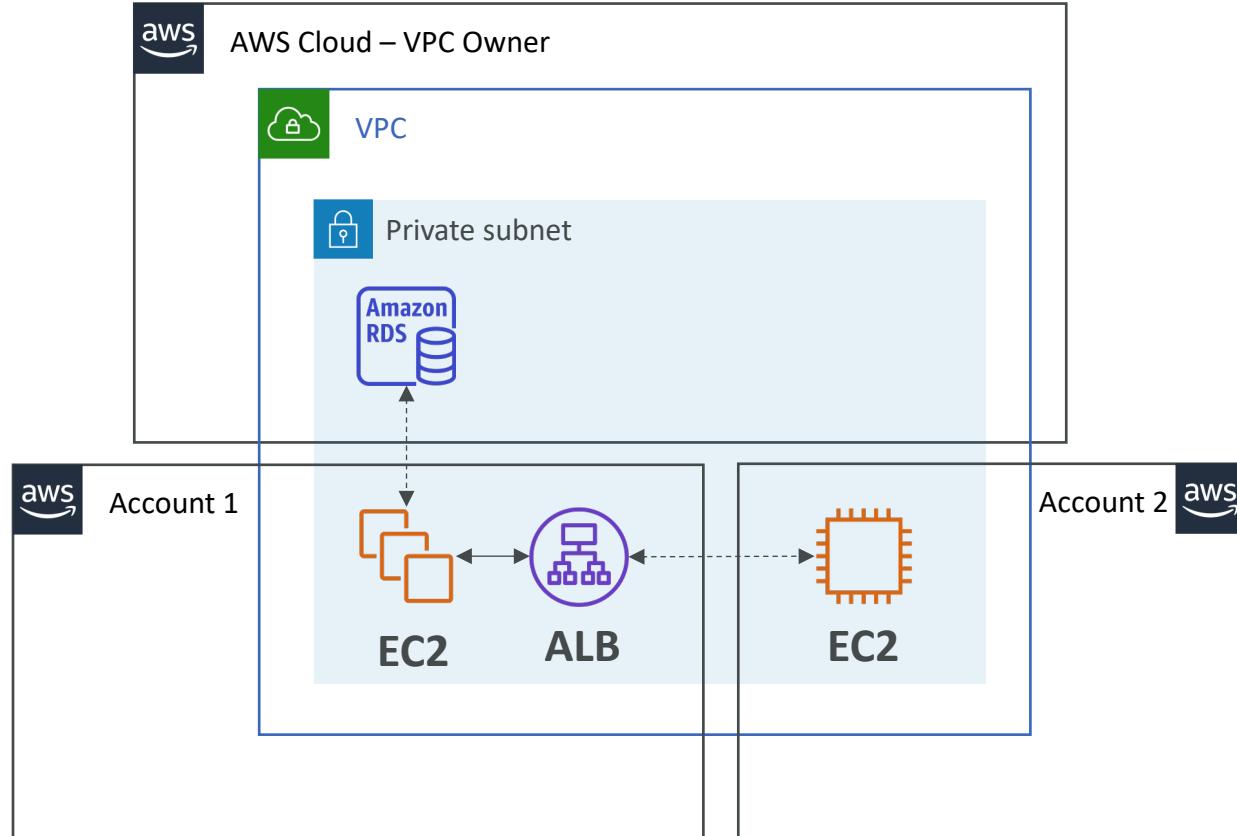
# AWS Resource Access Manager (RAM)

- Share AWS resources that you own with other AWS accounts
- Share with any account or within your Organization
- Avoid resource duplication!
- **VPC Subnets**
  - Allow to have all the resources launched in the same subnets
  - Must be from the same AWS Organizations.
  - Cannot share security groups and default VPC
  - Participants can manage their own resources in there
  - Participants can't view, modify, delete resources that belong to other participants or the owner
- **AWS Transit Gateway**
- **Route 53 (Resolver Rules, DNS Firewall Rule Groups)**
- **License Manager Configurations**

# AWS Resource Access Manager (RAM)

- Aurora DB Clusters
- ACM Private Certificate Authority
- CodeBuild Project
- EC2 (Dedicated Hosts, Capacity Reservation)
- AWS Glue (Catalog, Database, Table)
- AWS Network Firewall Policies
- AWS Resource Groups
- Systems Manager Incident Manager (Contacts, Response Plans)
- AWS Outposts (Outpost, Site)

# Resource Access Manager – VPC example



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

# Summary of Identity & Federation

- Users and Accounts all in AWS
- AWS Organizations
- AWS Control Tower to setup secure & compliant multi-account AWS environment (best practices)
- Federation with SAML
- Federation without SAML with a custom IdP ([GetFederationToken](#))
- AWS Single Sign-On to connect to multiple AWS Accounts (Organization) and SAML apps
- Web Identity Federation (not recommended)
- Cognito for most web and mobile applications (has anonymous mode, MFA)
- AWS Directory Service:
  - **Managed Microsoft AD** – standalone or setup trust AD with on-premises, has MFA, seamless join, RDS integration
  - **AD Connector** – proxy requests to on-premises
  - **Simple AD** – standalone & cheap AD-compatible with no MFA, no advanced capabilities
- AWS RAM to share resources (example VPC subnets)

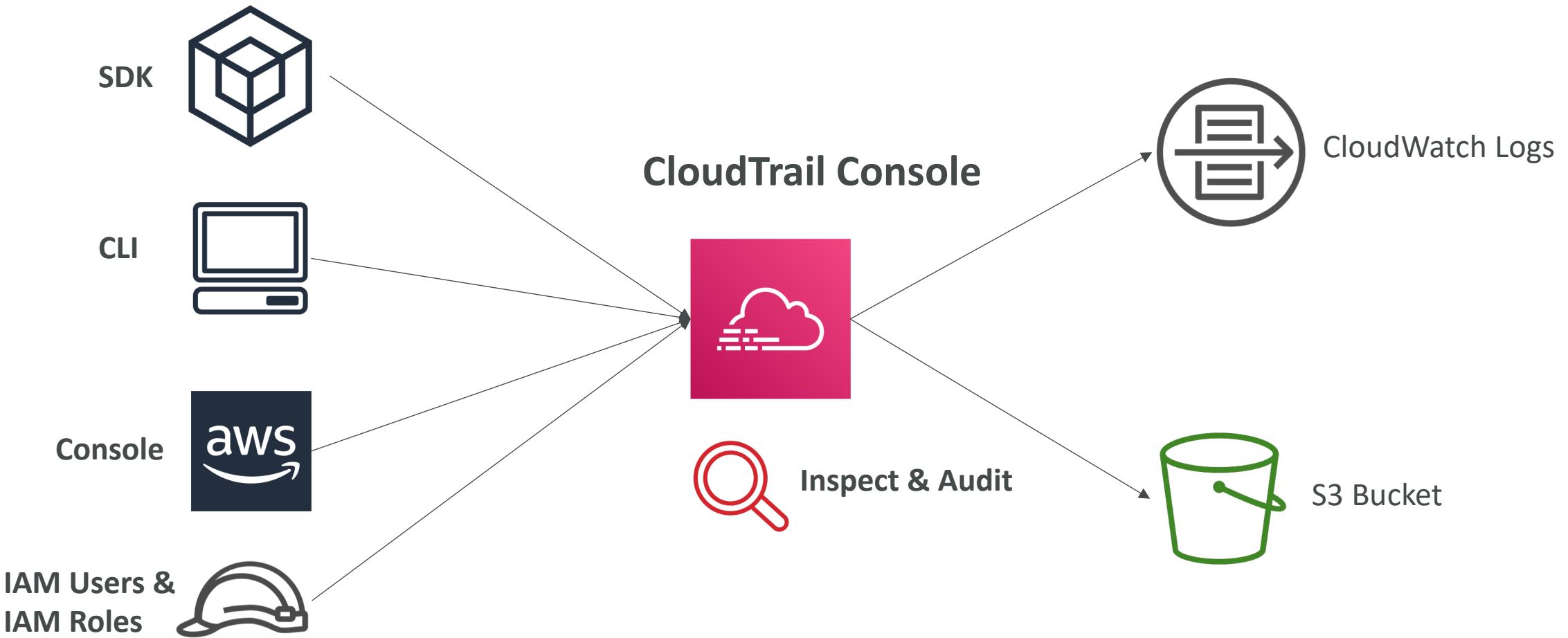
# Security Section



# AWS CloudTrail

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

# CloudTrail Diagram





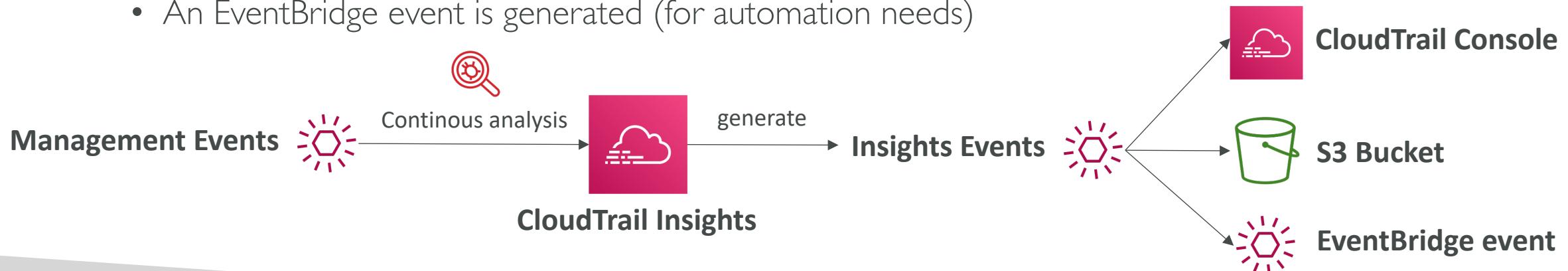
# CloudTrail Events

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



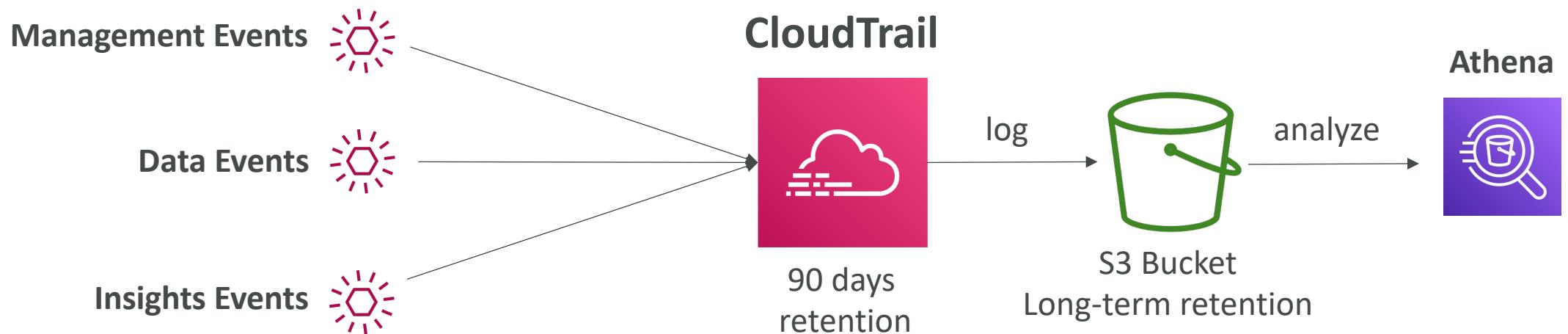
# CloudTrail Insights

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

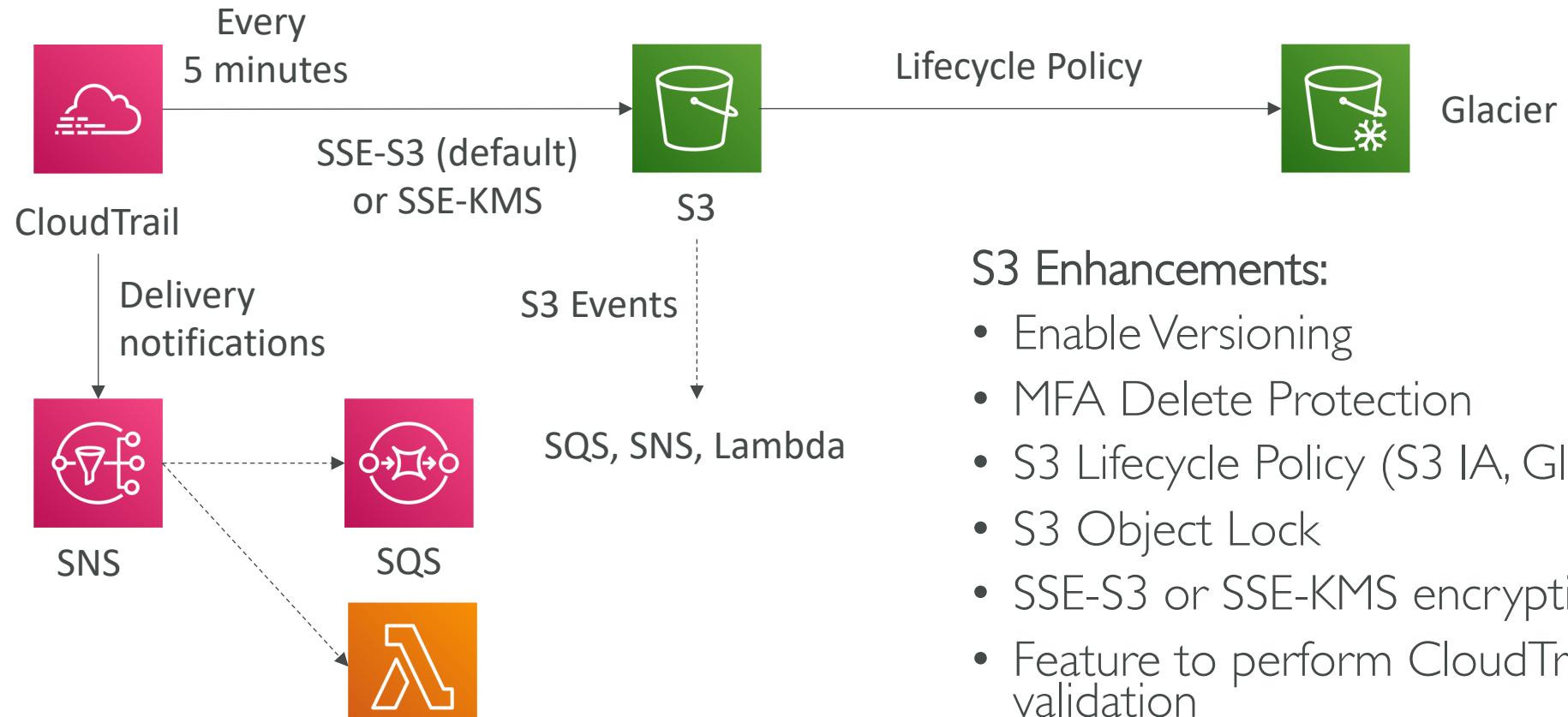


# CloudTrail Events Retention

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



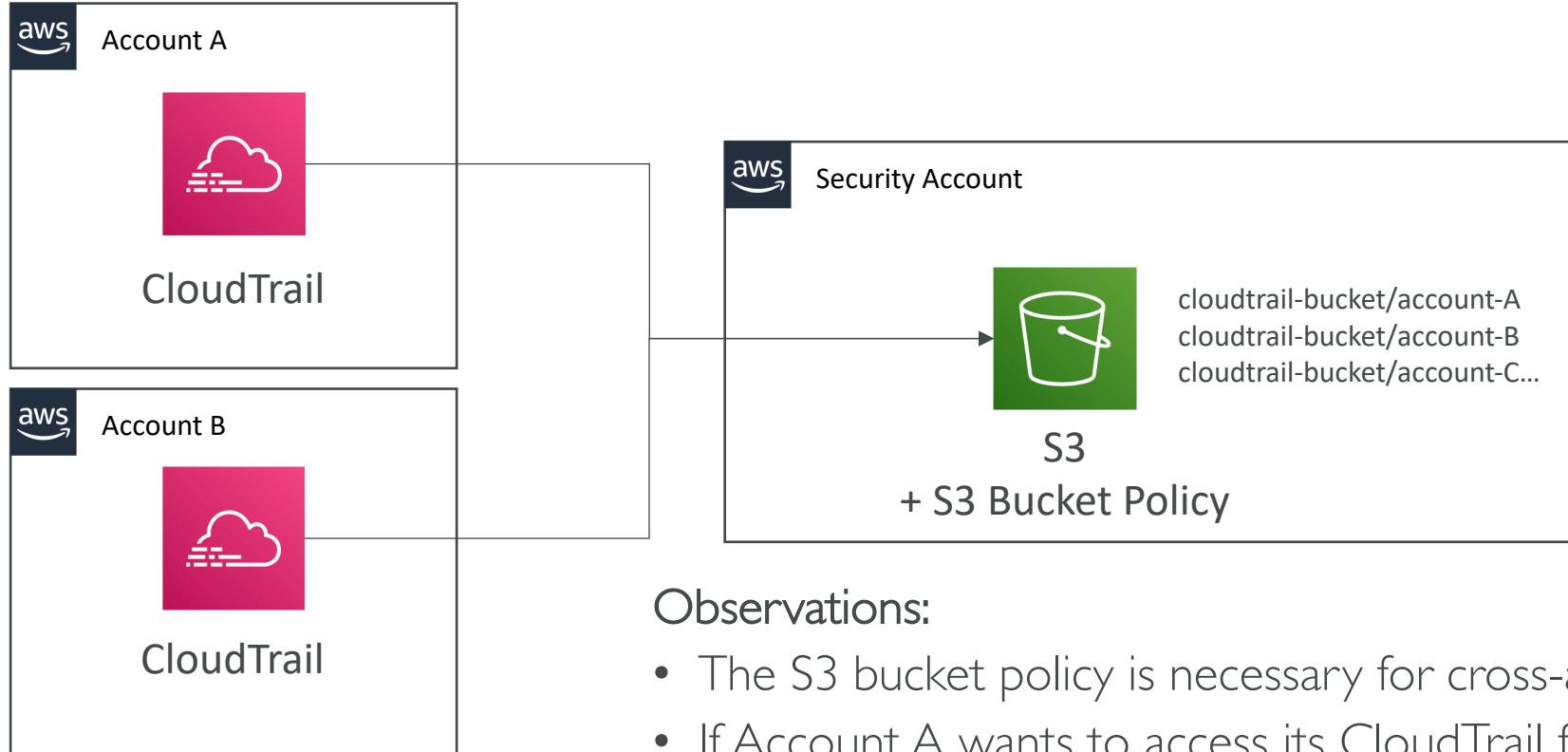
# CloudTrail – Solution Architecture: Delivery to S3



## S3 Enhancements:

- Enable Versioning
- MFA Delete Protection
- S3 Lifecycle Policy (S3 IA, Glacier...)
- S3 Object Lock
- SSE-S3 or SSE-KMS encryption
- Feature to perform CloudTrail Log File Integrity validation (SHA-256 for hashing and signing)

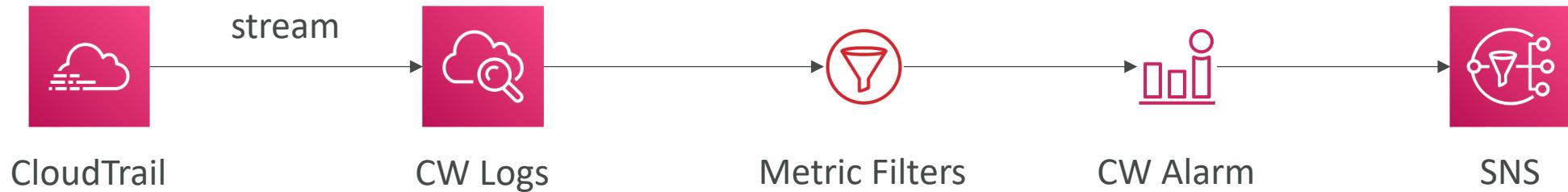
# CloudTrail - Solution Architecture: Multi Account, Multi Region Logging



## Observations:

- The S3 bucket policy is necessary for cross-account delivery
- If Account A wants to access its CloudTrail files:
  - Option 1: create a cross-account role and assume the role
  - Option 2: edit the bucket policy

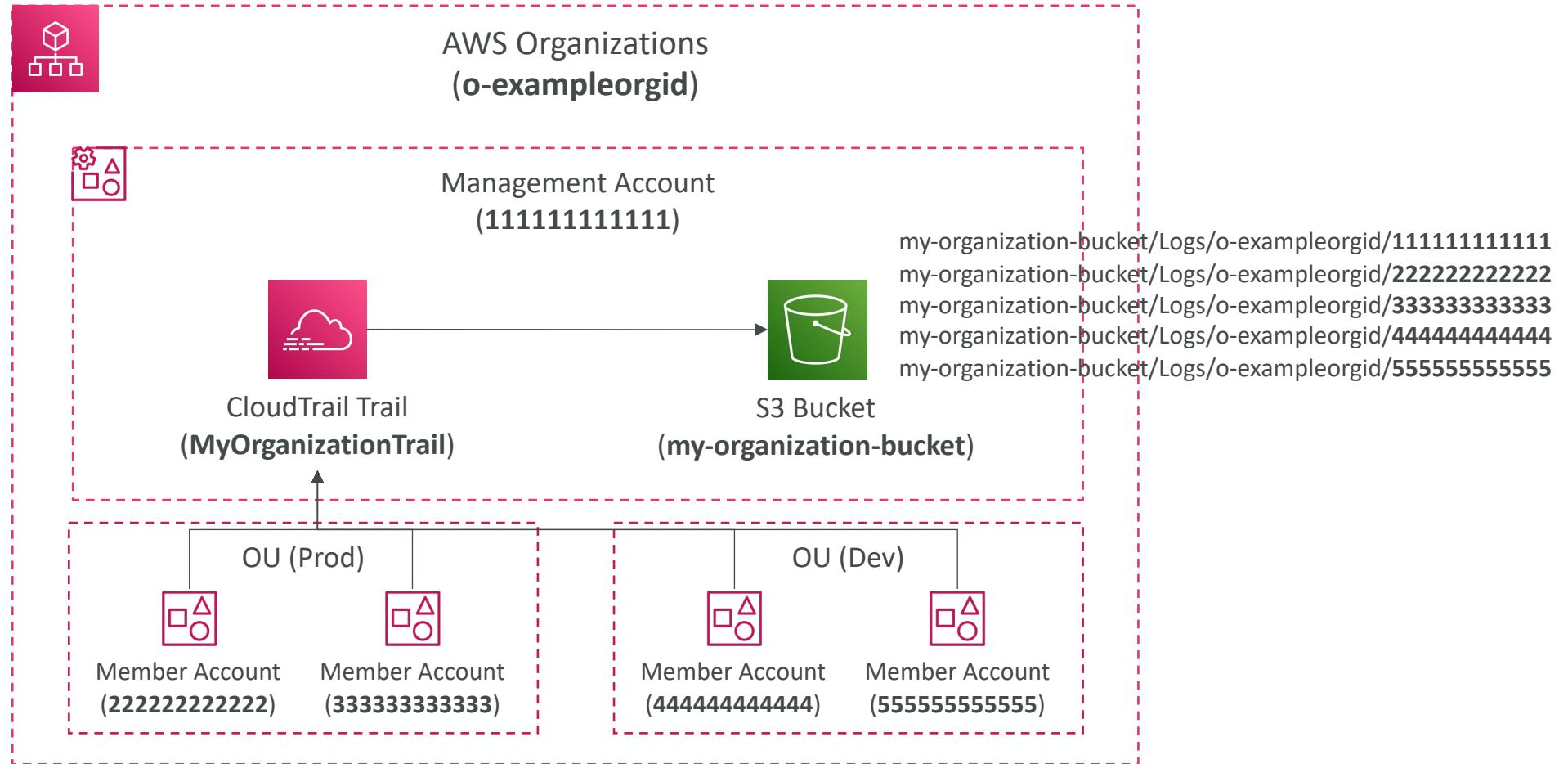
# CloudTrail - Solution Architecture: Alert for API calls



- Log filter metrics can be used to detect a high level of API happening
- Ex: Count occurrences of EC2 `TerminateInstances` API
- Ex: Count of API calls per user
- Ex: Detect high level of Denied API calls

# CloudTrail – Solution Architecture: Organizational Trail

The Organizational Trail is created in the management account.



# CloudTrail: How to react to events the fastest?

Overall, CloudTrail may take up to 15 minutes to deliver events

- **CloudWatch Events:**
  - Can be triggered for any API call in CloudTrail
  - The fastest, most reactive way
- **CloudTrail Delivery in CloudWatch Logs:**
  - Events are streamed
  - Can perform a metric filter to analyze occurrences and detect anomalies
- **CloudTrail Delivery in S3:**
  - Events are delivered every 5 minutes
  - Possibility of analyzing logs integrity, deliver cross account, long-term storage

# AWS KMS (Key Management Service)



- Anytime you hear “encryption” for an AWS service, it’s most likely KMS
- Easy way to control access to your data, AWS manages keys for us
- Fully integrated with IAM for authorization
- Seamlessly integrated into:
  - Amazon EBS: encrypt volumes
  - Amazon S3: Server-side encryption of objects
  - Amazon Redshift: encryption of data
  - Amazon RDS: encryption of data
  - Amazon SSM: Parameter store
  - Etc...
- But you can also use the CLI / SDK

# KMS – KMS Key Types

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

# Types of KMS Keys

- **Customer Managed Keys**
  - Create, manage and use, can enable or disable
  - Possibility of rotation policy (new key generated every year; old key preserved)
  - Can add a Key Policy (resource policy) & audit in CloudTrail
  - Leverage for envelope encryption
- **AWS Managed Keys**
  - Used by AWS service (aws/s3, aws/ebs, aws/redshift)
  - Managed by AWS (automatically rotated every 3 years)
  - View Key Policy & audit in CloudTrail
- **AWS Owned Keys**
  - Created and managed by AWS, use by some AWS services to protect your resources
  - Used in multiple AWS accounts, but they are not in your AWS account
  - You can't view, use, track, or audit

# Types of KMS Keys

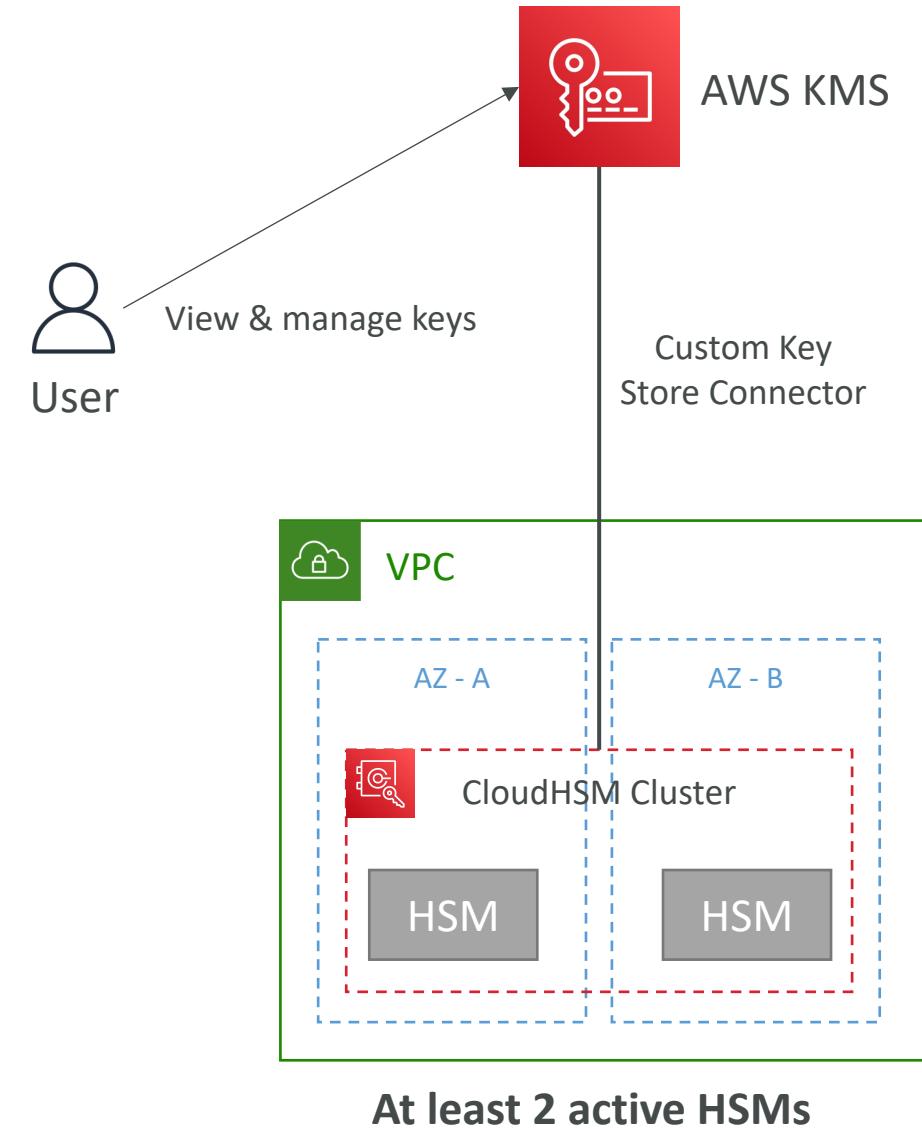
KMS Key	Customer Managed Key	AWS Managed Key	AWS Owned Key
Can view metadata?	✓	✓	✗
Can manage?	✓	✗	✗
Used only for my AWS account?	✓	✓	✗
Automatic Rotation	Optional (every 1 year)	Required (every 3 years)	Varies

# KMS Key Material Origin

- Identifies the source of the key material in the KMS key
- Can't be changed after creation
- **KMS (AWS\_KMS)** – default
  - AWS KMS creates and manages the key material in its own key store
- **External (EXTERNAL)**
  - You import the key material into the KMS key
  - You're responsible for securing and managing this key material outside of AWS
- **Custom Key Store (AWS\_CLOUDHSM)**
  - AWS KMS creates the key material in a custom key store (CloudHSM Cluster)

# KMS Key Source – Custom Key Store (CloudHSM)

- Integrate KMS with CloudHSM cluster as a Custom Key Store
- Key materials are stored in a CloudHSM cluster that you own and manage
- The cryptographic operations are performed in the HSMs
- Use cases:
  - You need direct control over the HSMs
  - KMS keys needs to be stored in a dedicated HSMs
  - HSMs must be validated at FIPS 140-2 Level 3 (KMS validated at FIPS 140-2 Level 2)

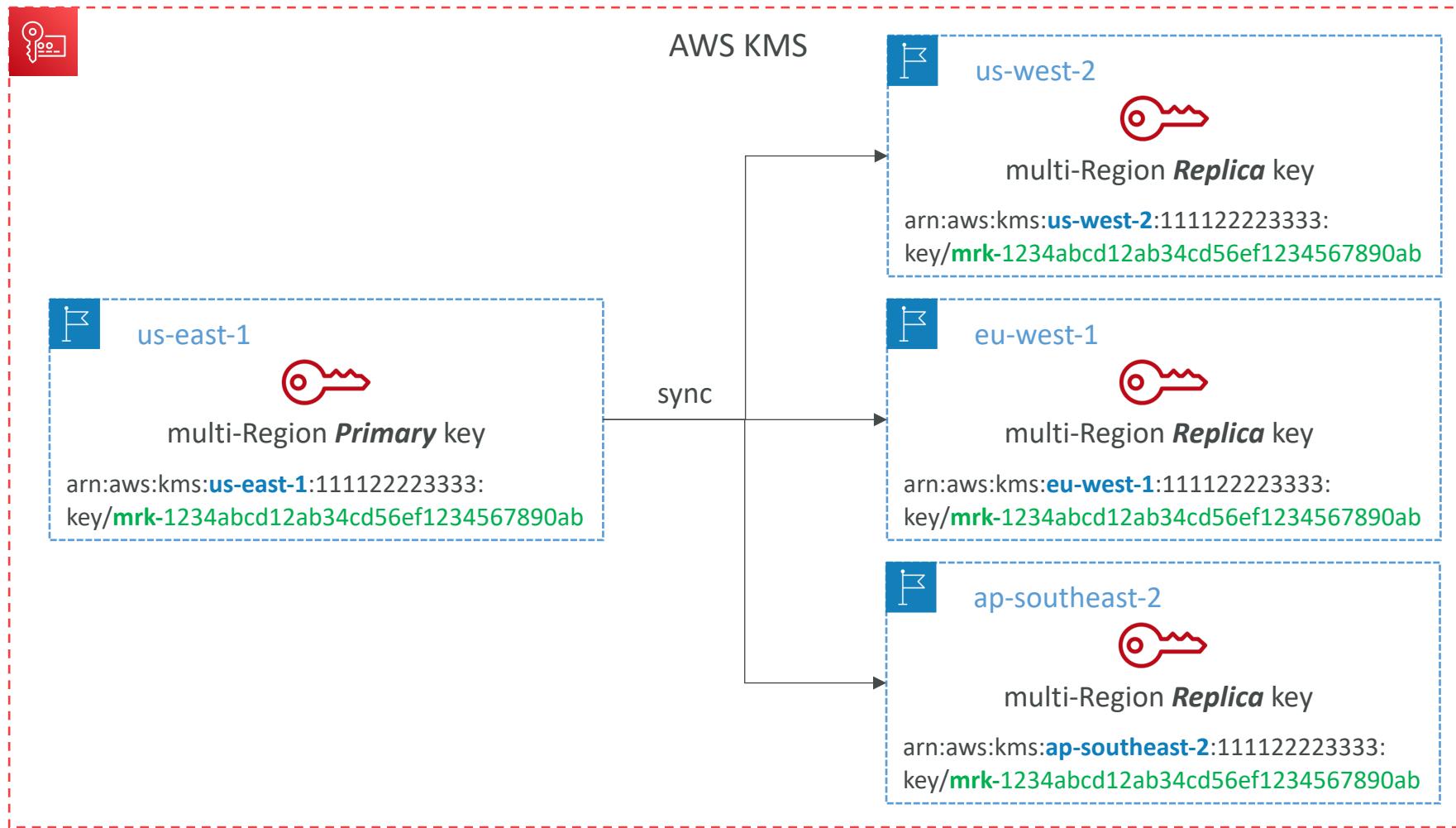


# KMS Key Source - External

- Import your own key material into KMS key, Bring Your Own Key (BYOK)
- You're responsible for key material's security, availability, and durability outside of AWS
- Must be 256-bit **Symmetric** key (Asymmetric is NOT supported)
- Can't be used with Custom Key Store (CloudHSM)
- Manually rotate your KMS key (Automatic Key Rotation is NOT supported)



# KMS Multi-Region Keys



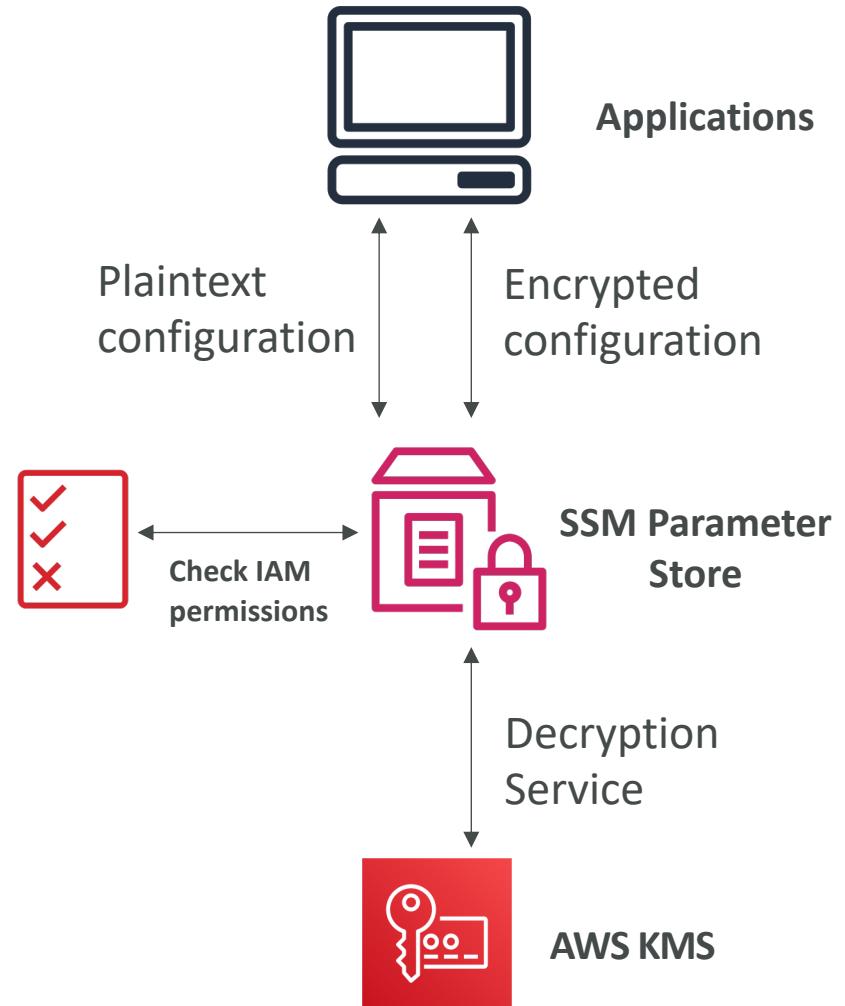
# KMS Multi-Region Keys



- A set of identical KMS keys in different AWS Regions that can be used interchangeably (~ same KMS key in multiple Regions)
- Encrypt in one Region and decrypt in other Regions (No need to re-encrypt or making cross-Region API calls)
- Multi-Region keys have the same key ID, key material, automatic rotation, ...
- KMS Multi-Region are NOT global (Primary + Replicas)
- Each Multi-Region key is managed **independently**
- Only one primary key at a time, can promote replicas into their own primary
- Use cases: Disaster Recovery, Global Data Management (e.g., DynamoDB Global Tables), Active-Active Applications that span multiple Regions, Distributed Signing applications, ...

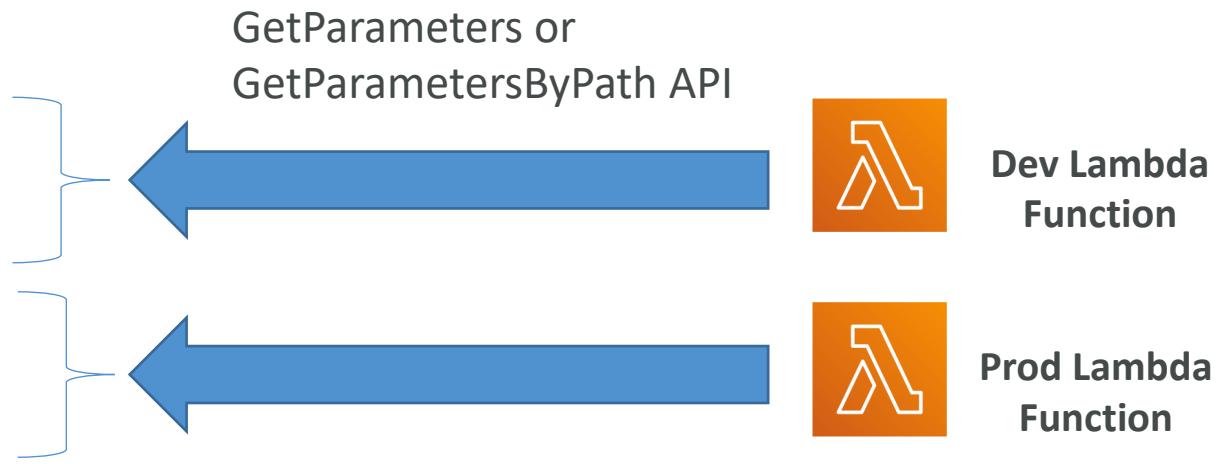
# SSM Parameter Store

- Secure storage for configuration and secrets
- Optional Seamless Encryption using KMS
- Serverless, scalable, durable, easy SDK
- Version tracking of configurations / secrets
- Configuration management using path & IAM
- Notifications with CloudWatch Events
- Integration with CloudFormation



# SSM Parameter Store Hierarchy

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



# Standard and advanced parameter tiers

	Standard	Advanced
Total number of parameters allowed (per AWS account and Region)	10,000	100,000
Maximum size of a parameter value	4 KB	8 KB
Parameter policies available	No	Yes
Cost	No additional charge	Charges apply
Storage Pricing	Free	\$0.05 per advanced parameter per month
API Interaction Pricing (higher throughput = up to 1000 Transactions per second)	Standard Throughput: free Higher Throughput: \$0.05 per 10,000 API interactions	Standard Throughput: \$0.05 per 10,000 API interactions Higher Throughput: \$0.05 per 10,000 API interactions

# Parameters Policies (for advanced parameters)

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

## Expiration (to delete a parameter)

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

## ExpirationNotification (CW Events)

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

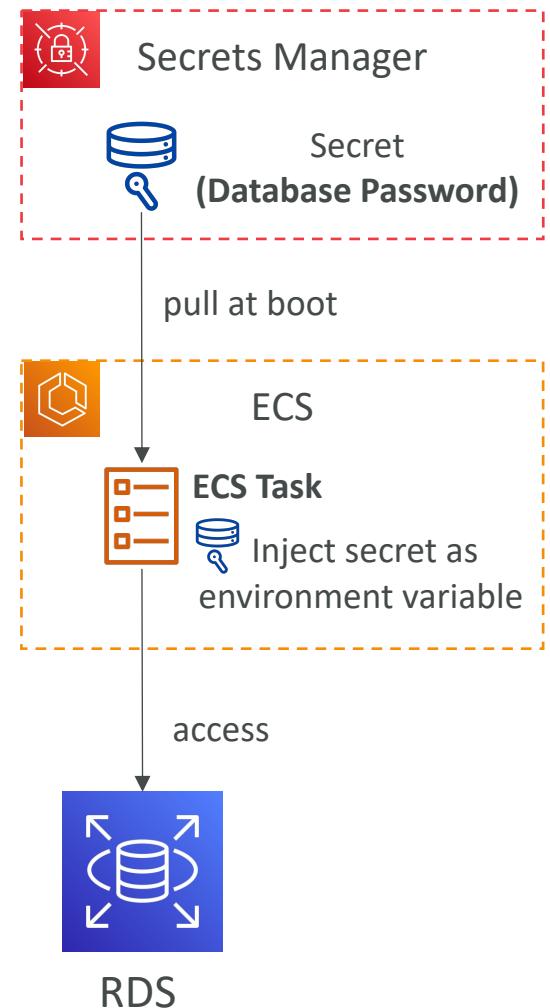
## NoChangeNotification (CW Events)

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

# AWS Secrets Manager



- Meant for storing secrets (e.g., passwords, API keys, ...)
- Capability to force **rotation of secrets** every X days
  - Automate generation of secrets on rotation (uses Lambda)
  - Natively supports Amazon RDS (all supported DB engines), Redshift, DocumentDB
  - Support other databases and services (custom Lambda function)
- Control access to secrets using Resource-based Policy
- Integration with other AWS services to natively pull secrets from Secrets Manager: *CloudFormation, CodeBuild, ECS, EMR, Fargate, EKS, Parameter Store...*



# Secrets Manager – with CloudFormation

```
Resources:  
    # Secret resource with a randomly generated password in its SecureString JSON  
    MyRDSDBInstanceRotationSecret:  
        Type: AWS::SecretsManager::Secret  
        Properties:  
            GenerateSecretString:  
                SecretStringTemplate: '{"username": "admin"}'  
                GenerateStringKey: password  
                PasswordLength: 16  
                ExcludeCharacters: "\"@/\\\""  
  
    # RDS Instance resource. Its master username and password use dynamic references  
    # to resolve values from Secrets Manager  
    MyRDSDBInstance:  
        Type: AWS::RDS::DBInstance  
        Properties:  
            DBInstanceClass: db.t2.micro  
            Engine: mysql  
            MasterUsername: !Sub "{{resolve:secretsmanager:${MyRDSDBInstanceRotationSecret}:username}}"  
            MasterUserPassword: !Sub "{{resolve:secretsmanager:${MyRDSDBInstanceRotationSecret}:password}}"  
  
    # SecretTargetAttachment resource which updates the referenced Secret with properties  
    # about the referenced RDS instance  
    SecretRDSDBInstanceAttachment:  
        Type: AWS::SecretsManager::SecretTargetAttachment  
        Properties:  
            TargetType: AWS::RDS::DBInstance  
            SecretId: !Ref MyRDSDBInstanceRotationSecret  
            TargetId: !Ref MyRDSDBInstance
```

secret is generated

reference secret in  
RDS DB instance

link the secret to  
RDS DB instance

# SSM Parameter Store vs Secrets Manager

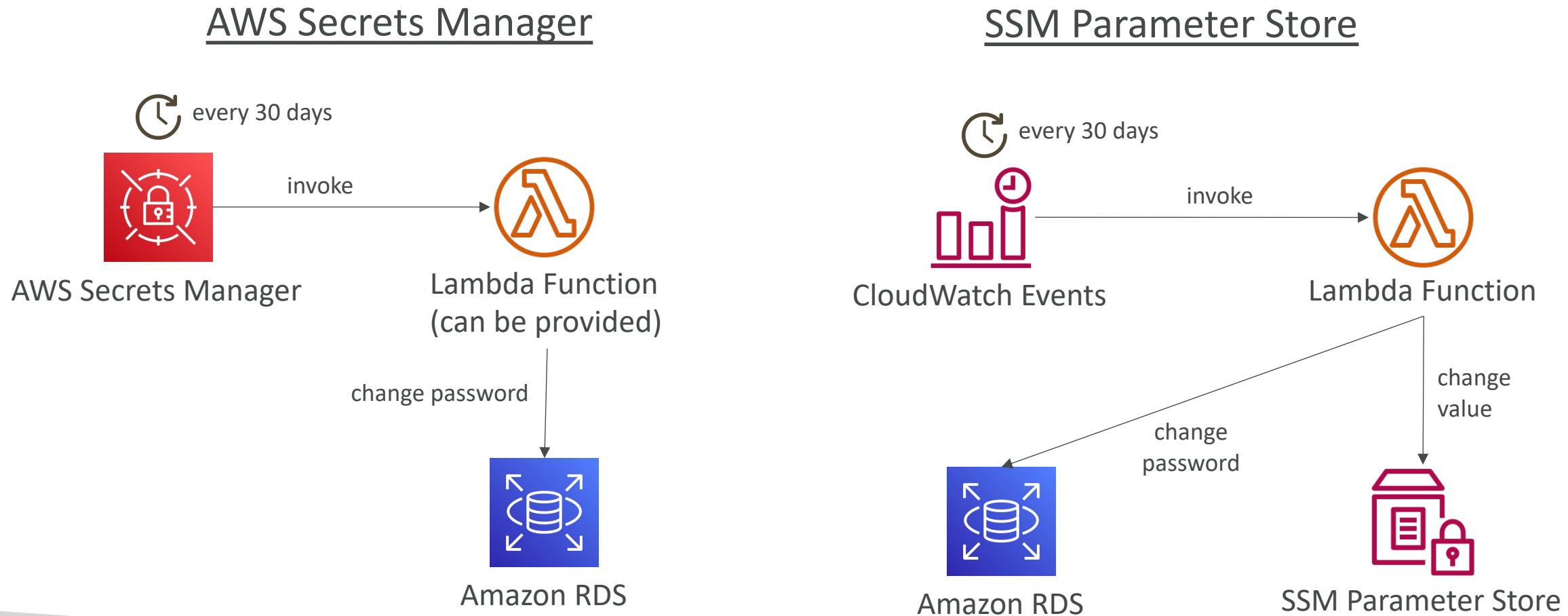
- **Secrets Manager (\$\$\$):**

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

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

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

# SSM Parameter Store vs. Secrets Manager Rotation



# RDS - Security

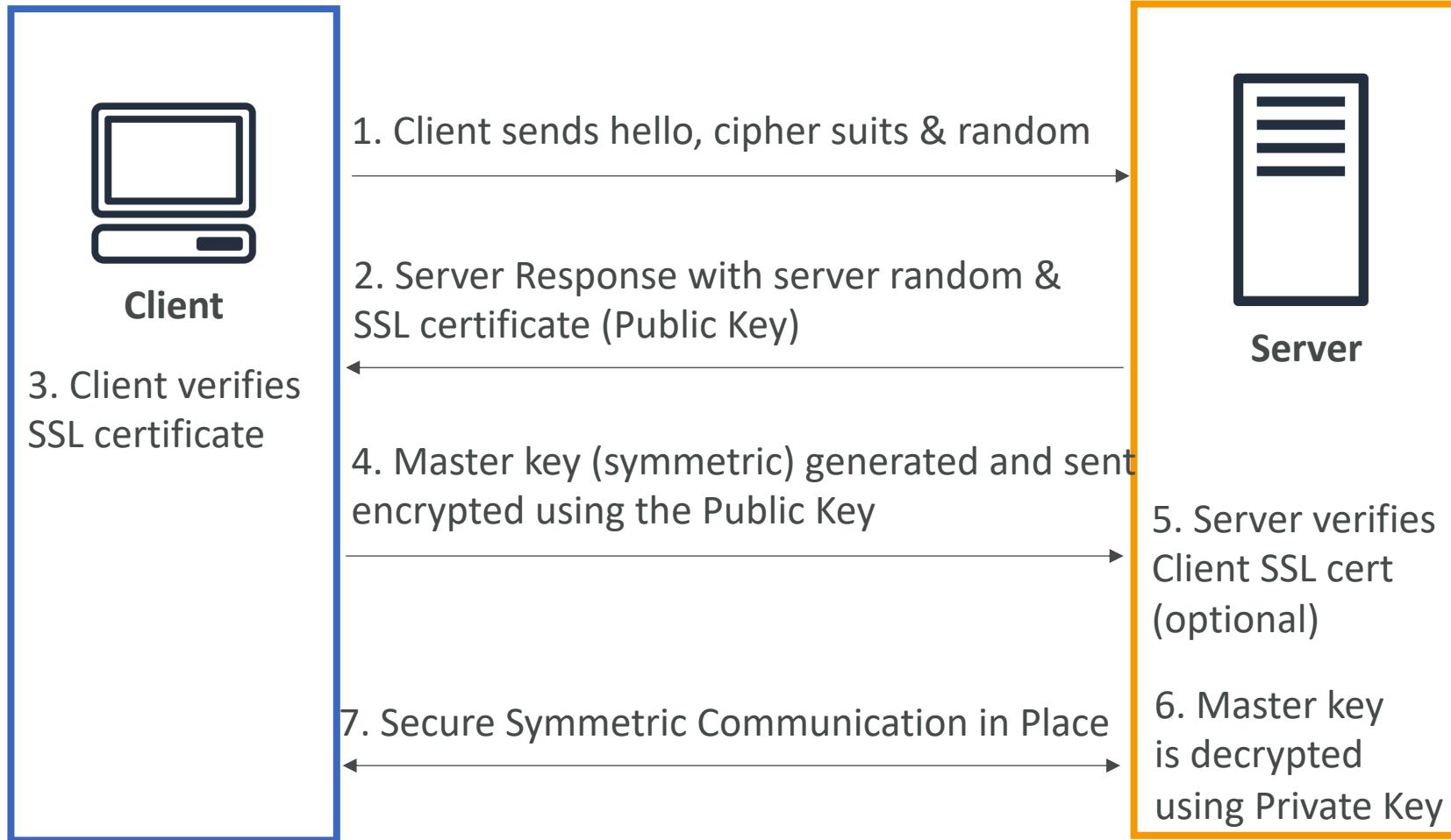


- KMS encryption at rest for underlying EBS volumes / snapshots
- Transparent Data Encryption (TDE) for Oracle and SQL Server
- SSL encryption to RDS is possible for all DB (in-flight)
- IAM authentication for MySQL and PostgreSQL
- Authorization still happens within RDS (not in IAM)
- Can copy an un-encrypted RDS snapshot into an encrypted one
- CloudTrail cannot be used to track queries made within RDS

# SSL/TLS - Basics

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

# SSL Encryption – How it works



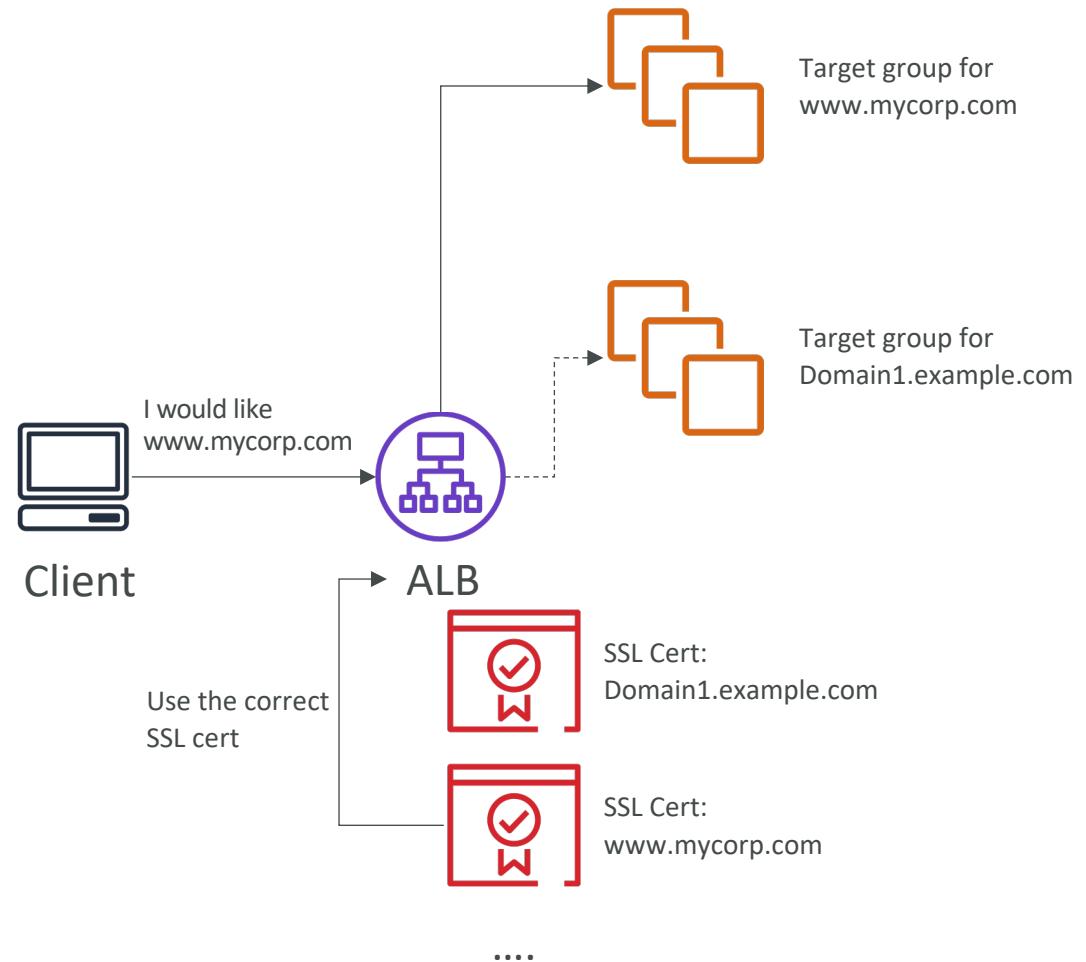
- Asymmetric Encryption is expensive (SSL)
- Symmetric encryption is cheaper
- Asymmetric handshake is used to exchange a per-client random symmetric key
- Possibility of client sending an SSL certificate as well (two-way certificate)

# SSL – Server Name Indication (SNI)

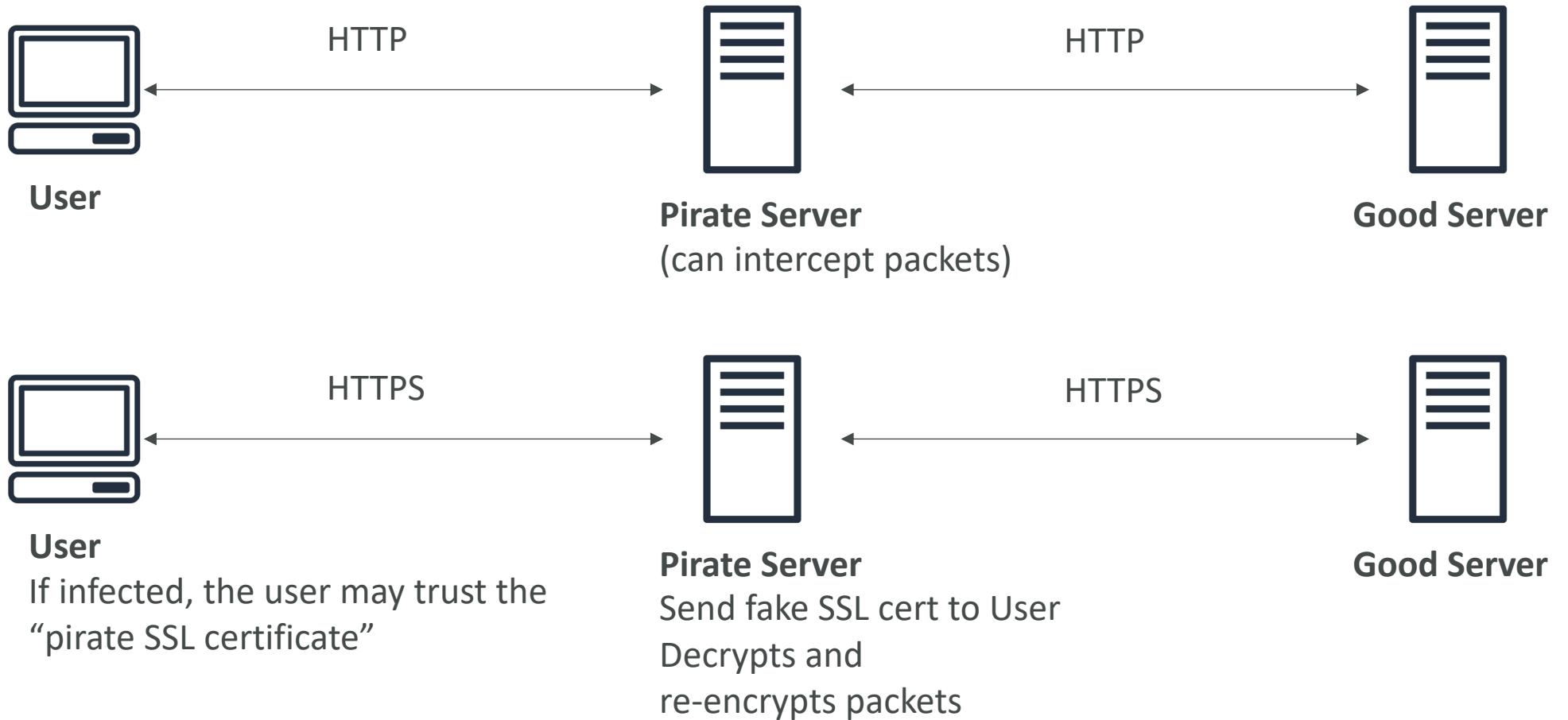
- SNI solves the problem of loading **multiple SSL certificates onto one web server** (to serve multiple websites)
- It's a “newer” protocol, and requires the client to **indicate** the hostname of the target server in the initial SSL handshake
- The server will then find the correct certificate, or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

## Note:

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



# SSL – Man in the Middle Attacks



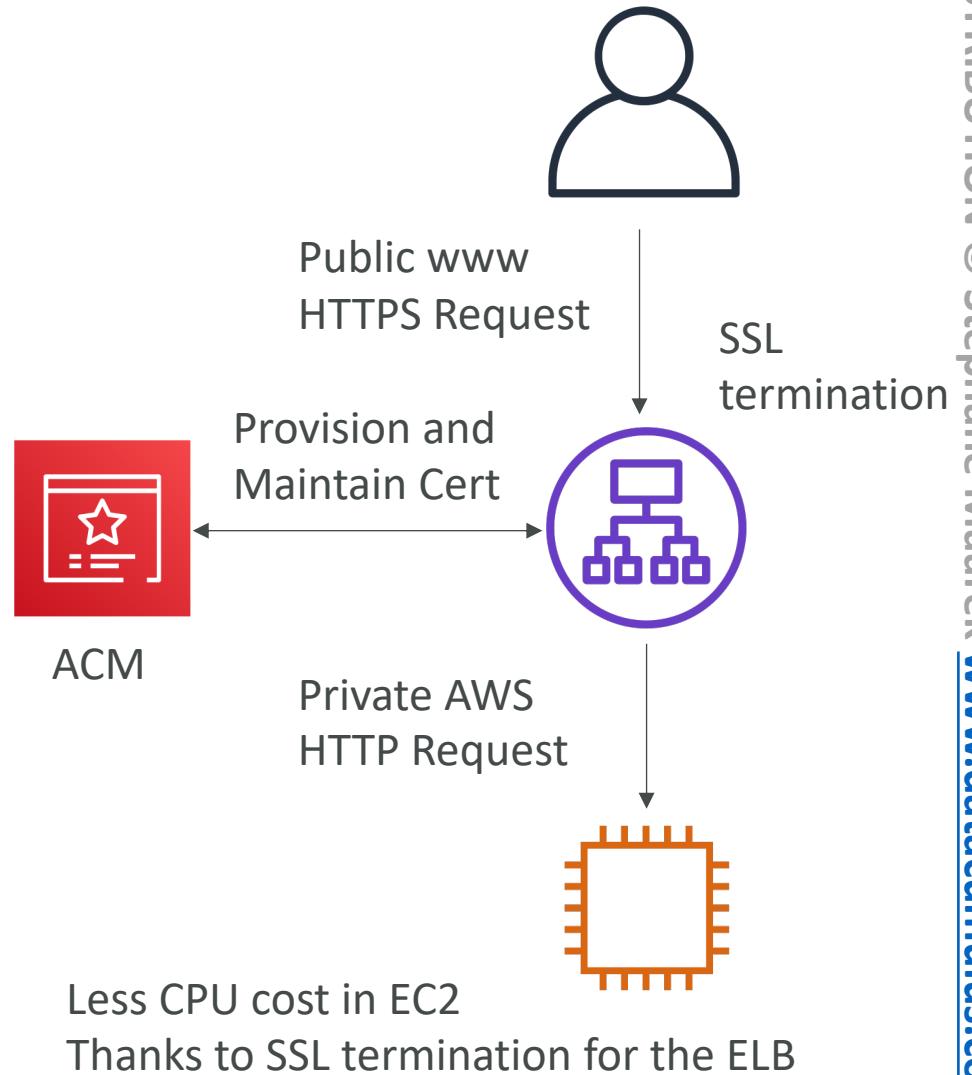
# SSL – Man in the Middle Attack

## How to prevent

1. Don't use public-facing HTTP, use HTTPS (meaning, use SSL/TLS certificates)
2. Use a DNS that has DNSSEC
  - To send a client to a pirate server, a DNS response needs to be “forged” by a server which intercepts them
  - It is possible to protect your domain name by configuring DNSSEC
  - Amazon Route 53 supports DNSSEC for domain registration.
  - Route 53 supports DNSSEC for DNS service as of December 2020 (using KMS)
  - You could also run a custom DNS server on Amazon EC2 for example (Bind is the most popular; dnsmasq, KnotDNS, PowerDNS).

# AWS Certificate Manager (ACM)

- To host public SSL certificates in AWS, you can:
  - Buy your own and upload them using the CLI
  - Have ACM provision and renew public SSL certificates for you (free of cost)
- ACM loads SSL certificates on the following integrations:
  - Load Balancers (including the ones created by EB)
  - CloudFront distributions
  - APIs on API Gateways
- SSL certificates is overall a pain to manually manage, so ACM is great to leverage in your AWS infrastructure!



# ACM – Good to know

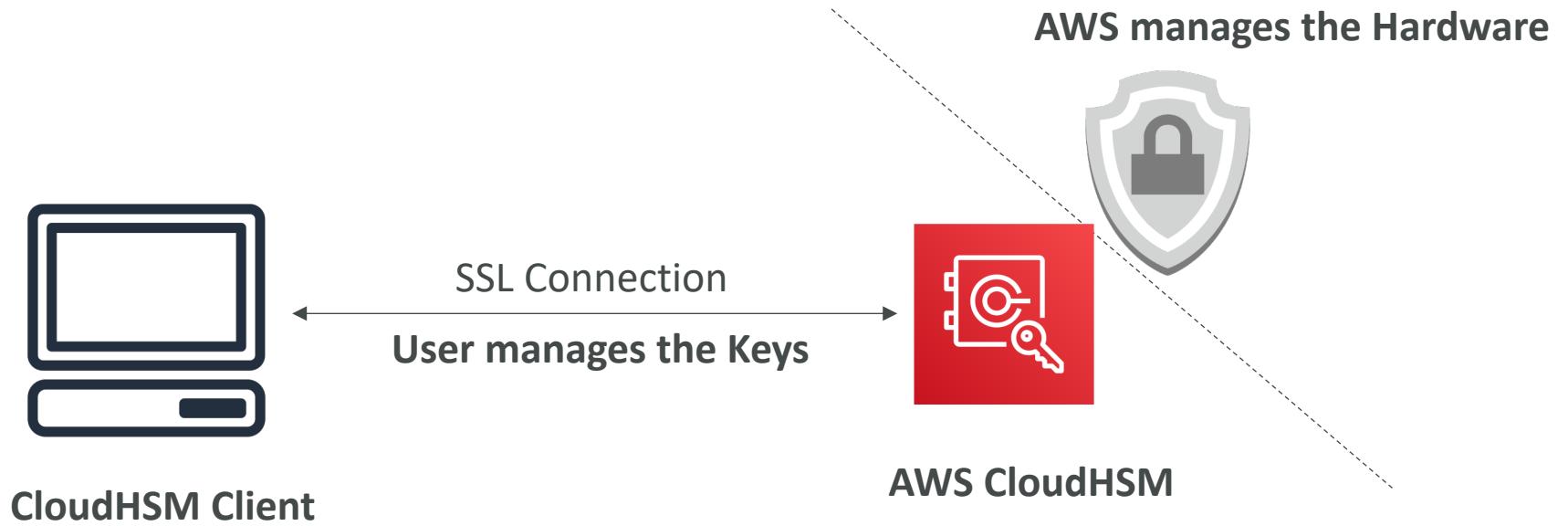
- Possibility of creating public certificates
  - Must verify public DNS
  - Must be issued by a trusted public certificate authority (CA)
- Possibility of creating private certificates
  - For your internal applications
  - You create your own private CA
  - Your applications must trust your private CA
- Certificate renewal:
  - Automatically done if generated provisioned by ACM
  - Any manually uploaded certificates must be renewed manually and re-uploaded
- ACM is a **regional** service
  - To use with a global application (multiple ALB for example), you need to issue an SSL certificate in each region where your application is deployed.
  - You cannot copy certs across regions

# CloudHSM



- KMS => AWS manages the software for encryption
- CloudHSM => AWS provisions encryption **hardware**
- Dedicated Hardware (HSM = Hardware Security Module)
- You manage your own encryption keys entirely (not AWS)
- HSM device is tamper resistant, FIPS 140-2 Level 3 compliance
- Supports both symmetric and **asymmetric** encryption (SSL/TLS keys)
- No free tier available
- Must use the CloudHSM Client Software
- Redshift supports CloudHSM for database encryption and key management
- **Good option to use with SSE-C encryption**

# CloudHSM Diagram



## IAM permissions:

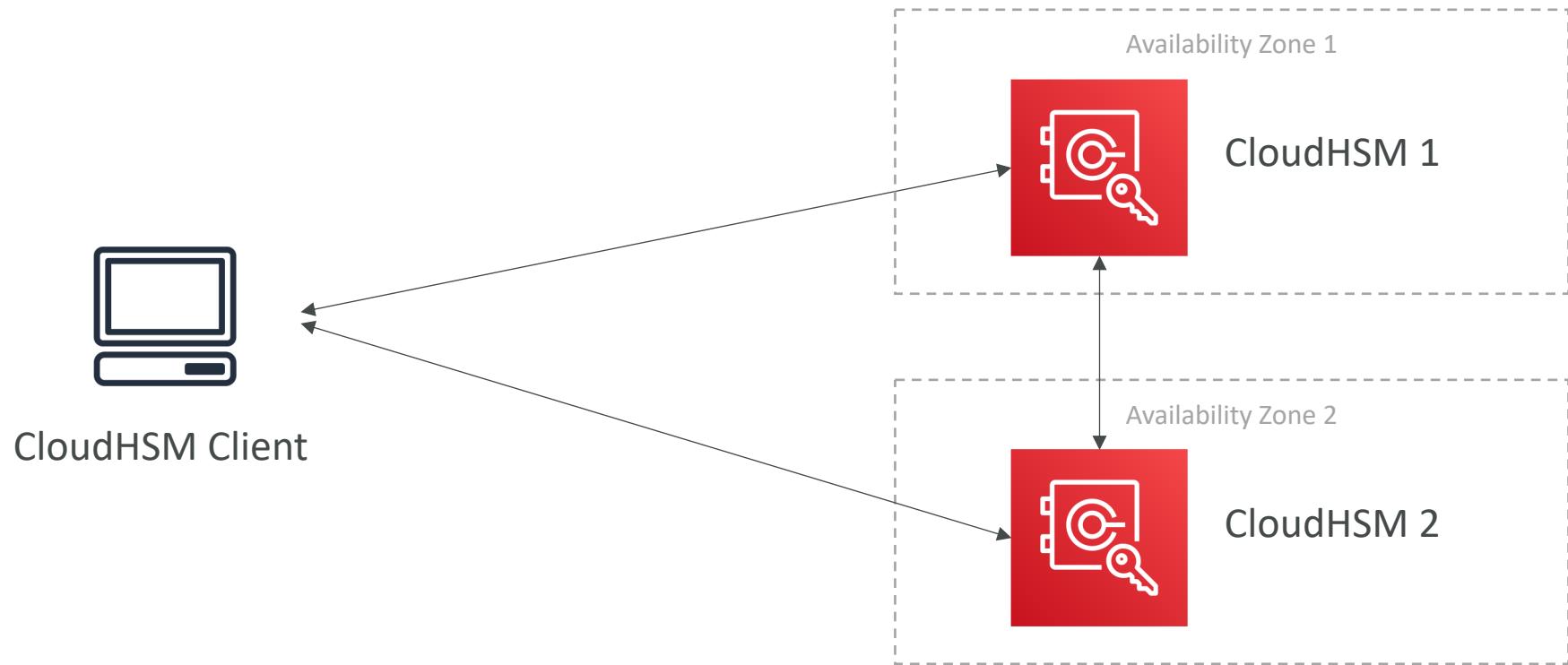
- CRUD an HSM Cluster

## CloudHSM Software:

- Manage the Keys
- Manage the Users

# CloudHSM – High Availability

- CloudHSM clusters are spread across Multi AZ (HA)
- Great for availability and durability



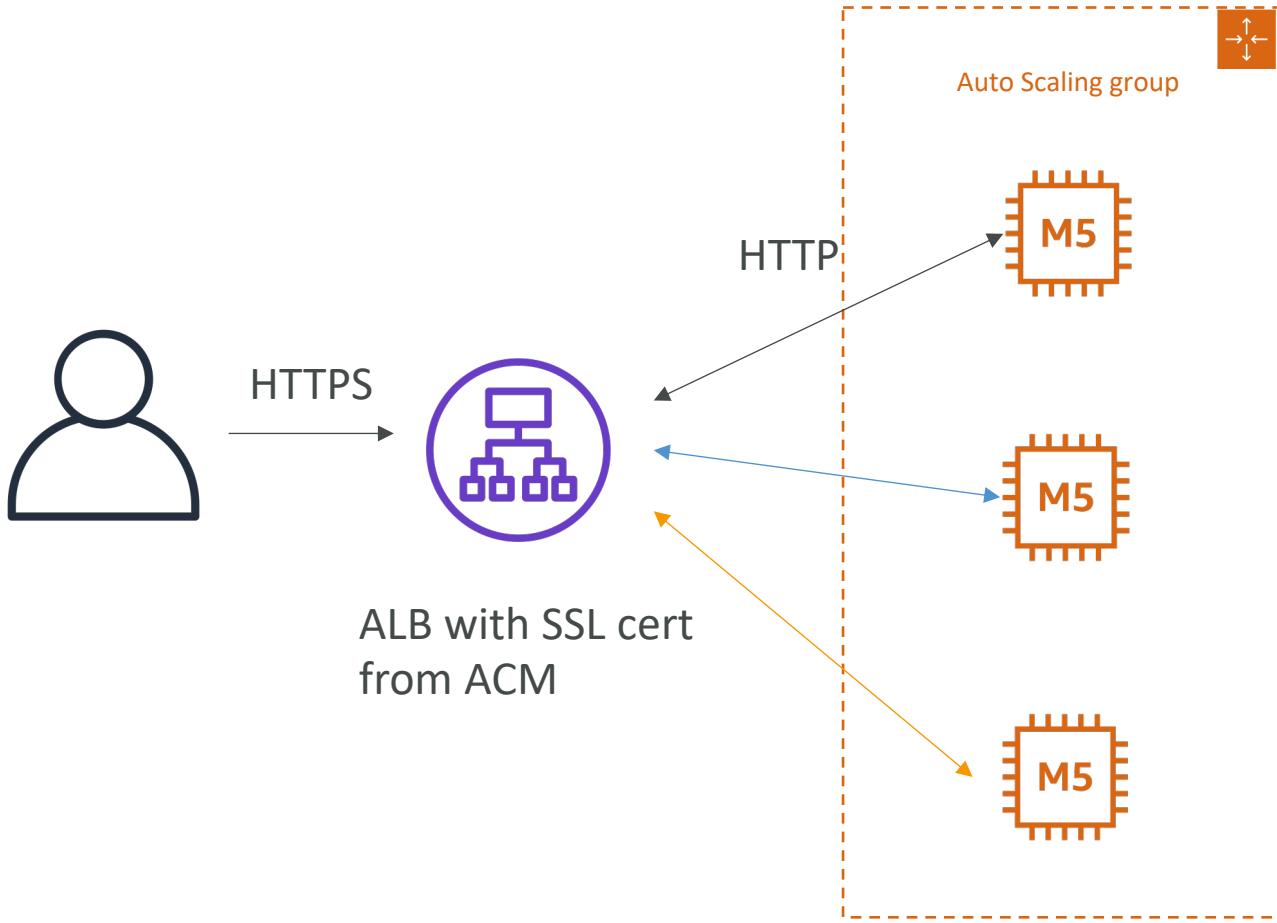
# CloudHSM vs. KMS

Feature	AWS KMS	AWS CloudHSM
Tenancy	Multi-Tenant	Single-Tenant
Standard	FIPS 140-2 Level 2	FIPS 140-2 Level 3
Master Keys	<ul style="list-style-type: none"><li>• AWS Owned Keys</li><li>• AWS Managed Keys</li><li>• Customer Managed KMS Keys</li></ul>	Customer Managed CMK
Key Types	<ul style="list-style-type: none"><li>• Symmetric</li><li>• Asymmetric</li><li>• Digital Signing</li></ul>	<ul style="list-style-type: none"><li>• Symmetric</li><li>• Asymmetric</li><li>• Digital Signing &amp; Hashing</li></ul>
Key Accessibility	Accessible in multiple AWS regions (can't access keys outside the region it's created in)	<ul style="list-style-type: none"><li>• Deployed and managed in a VPC</li><li>• Can be shared across VPCs (VPC Peering)</li></ul>
Cryptographic Acceleration	None	<ul style="list-style-type: none"><li>• SSL/TLS Acceleration</li><li>• Oracle TDE Acceleration</li></ul>
Access & Authentication	AWS IAM	You create users and manage their permissions

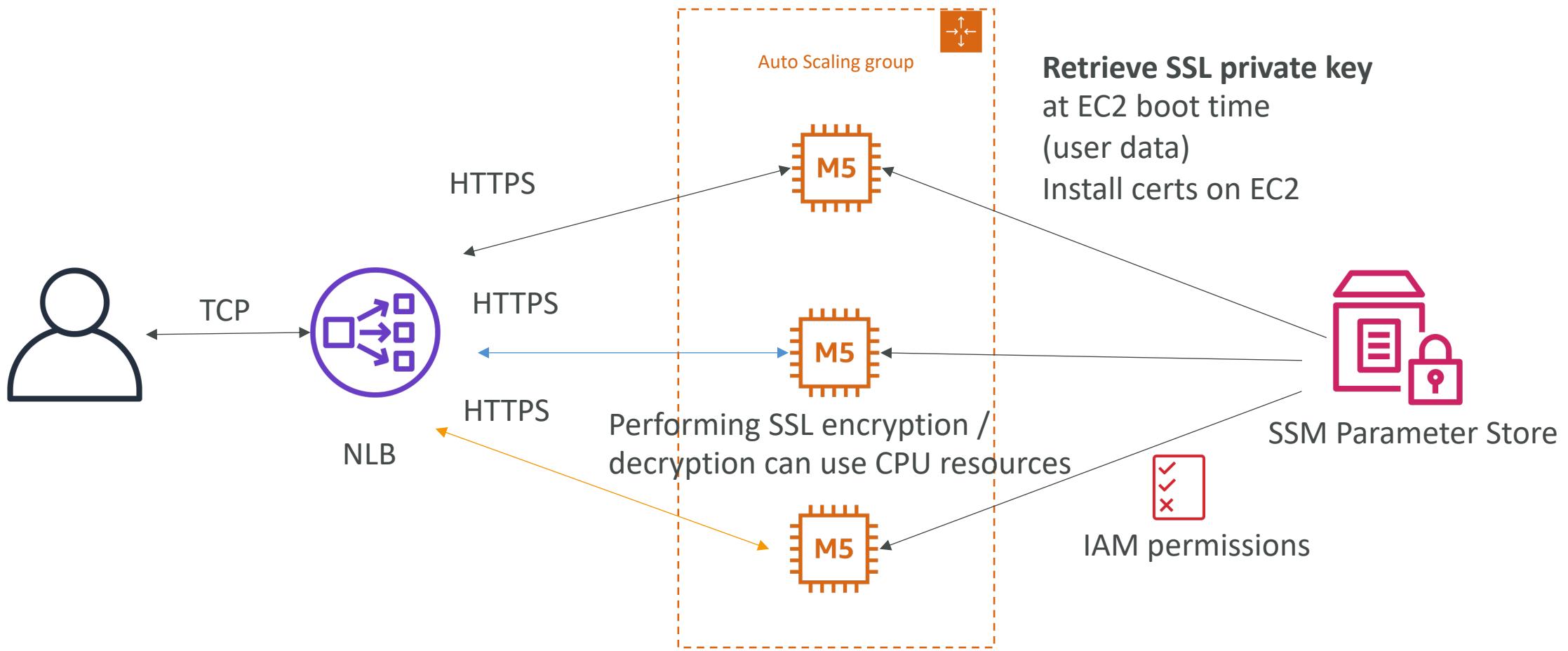
# CloudHSM vs. KMS

Feature	AWS KMS	AWS CloudHSM
High Availability	AWS Managed Service	Add multiple HSMs over different AZs
Audit Capability	<ul style="list-style-type: none"><li>• CloudTrail</li><li>• CloudWatch</li></ul>	<ul style="list-style-type: none"><li>• CloudTrail</li><li>• CloudWatch</li><li>• MFA support</li></ul>
Free Tier	Yes	No

# Solution Architecture: SSL on ALB

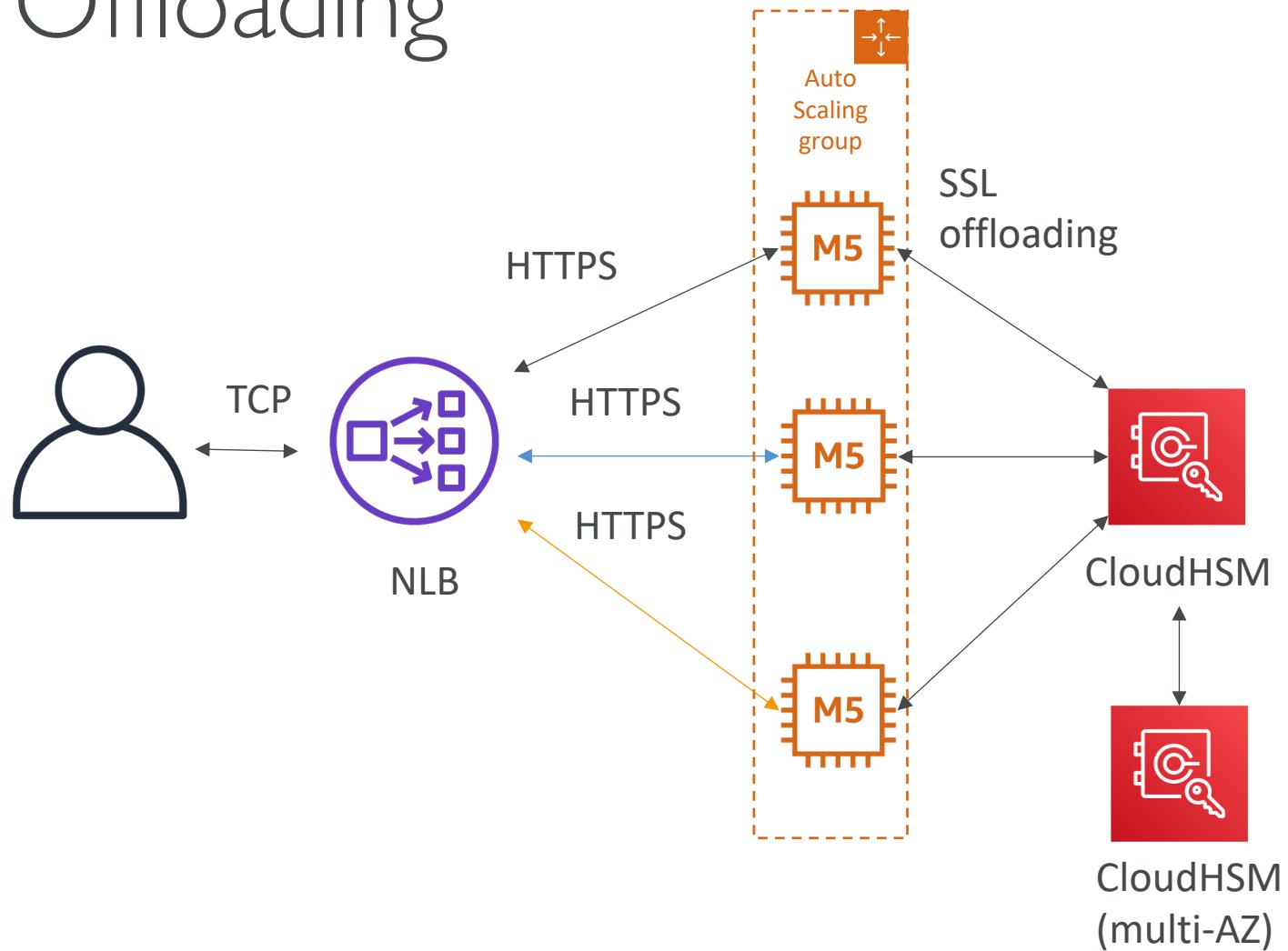


# Solution Architecture: SSL on web server EC2 instances



# Solution Architecture: CloudHSM – SSL Offloading

- You can offload SSL to CloudHSM (SSL Acceleration)
- Supported by NGINX, Apache Web servers and IIS for Windows Server
- Extra security: the SSL private key never leaves the HSM device
- Must setup a cryptographic user (CU) on the CloudHSM device



# S3 Encryption for Objects

- There are 4 methods of encrypting objects in S3
- **SSE-S3:** encrypts S3 objects using keys handled & managed by AWS
- **SSE-KMS:** leverage AWS Key Management Service to manage encryption keys
- **SSE-C:** when you want to manage your own encryption keys
- Client Side Encryption
- **Glacier:** all data is AES-256 encrypted, key under AWS control

# Encryption in transit (SSL)

- AWS S3 exposes:
  - HTTP endpoint: non encrypted
  - HTTPS endpoint: encryption in flight
- You're free to use the endpoint you want, but HTTPS is recommended
- HTTPS is mandatory for SSE-C
- Encryption in flight is also called SSL / TLS

# Events in S3 Buckets

- **S3 Access Logs:**
  - Detailed records for the requests that are made to a bucket
  - Might take hours to deliver
  - Might be incomplete (best effort)
- **S3 Events Notifications:**
  - Receive notifications when certain events happen in your bucket
  - E.g.: new objects created, object removal, restore objects, replication events
  - Destinations: SNS, SQS queue, Lambda
  - Typically delivered in seconds but can take minutes, notification for every object if versioning is enabled, else risk of one notification for two same object write done simultaneously
- **Trusted Advisor:**
  - Check the bucket permission (is the bucket public?)
- **CloudWatch Events:**
  - Need to enable CloudTrail object level logging on S3 first
  - Target can be Lambda, SQS, SNS, etc...

# S3 Security

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

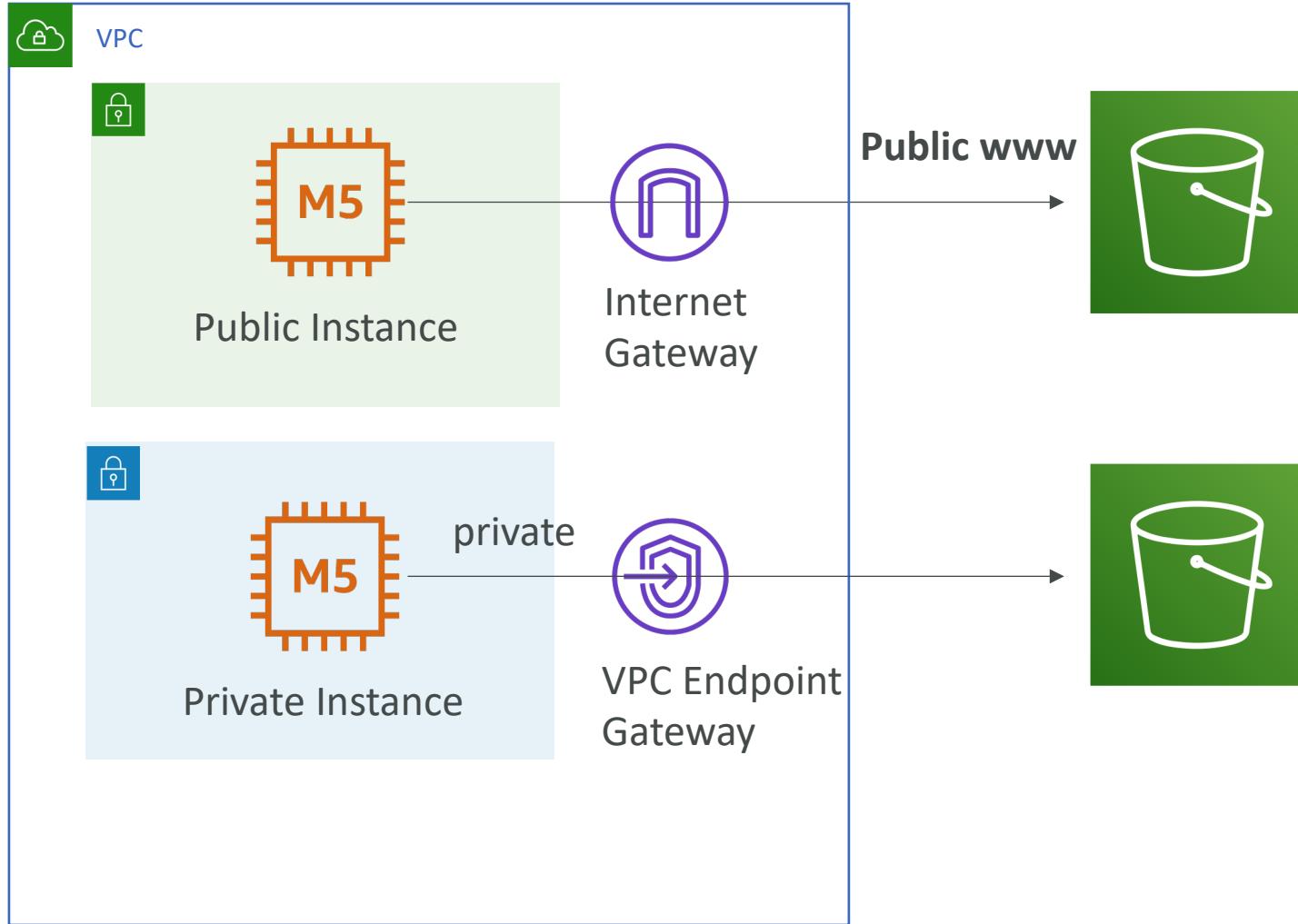
# S3 Bucket Policies

- Use S3 bucket for policy to:
  - Grant public access to the bucket
  - Force objects to be encrypted at upload
  - Grant access to another account (Cross Account)
- Optional Conditions on:
  - Public IP or Elastic IP (not on Private IP)
  - Source VPC or Source VPC Endpoint – only works with VPC Endpoints
  - CloudFront Origin Identity
  - MFA
- Examples here: <https://docs.aws.amazon.com/AmazonS3/latest/dev/example-bucket-policies.html>

# S3 pre-signed URLs

- Can generate pre-signed URLs using SDK or CLI
  - For downloads (easy, can use the CLI)
  - For uploads (harder, must use the SDK)
- Valid for a default of 3600 seconds, can change timeout with --expires-in [TIME\_BY\_SECONDS] argument
- Users given a pre-signed URL inherit the permissions of the person who generated the URL for GET / PUT
- Examples :
  - Allow only logged-in users to download a premium video on your S3 bucket
  - Allow an ever changing list of users to download files by generating URLs dynamically
  - Allow temporarily a user to upload a file to a precise location in our bucket

# VPC Endpoint Gateway for S3



S3 Bucket  
Bucket policy by **AWS:SourceIP** (public IP)

S3 Bucket  
Bucket policy by  
**AWS:SourceVpc**  
(one or few endpoints)

OR

**AWS:SourceVpc**  
(encompass all possible VPC endpoints)

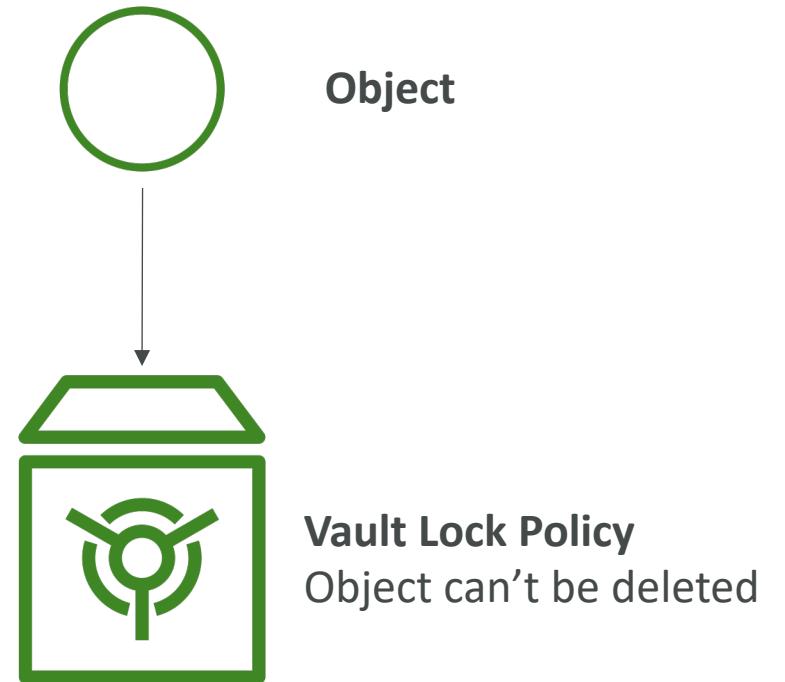
# S3 Object Lock & Glacier Vault Lock

- **S3 Object Lock**

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

- **Glacier Vault Lock**

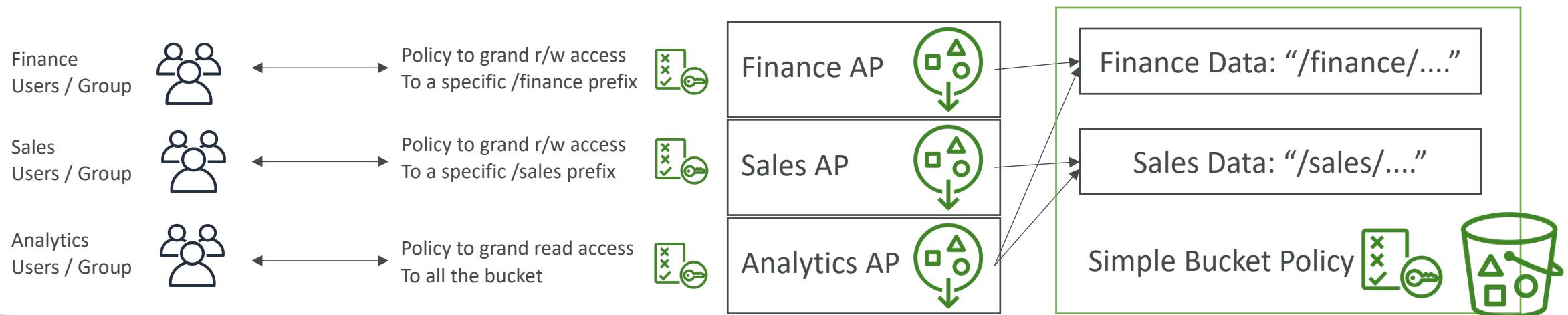
- Adopt a WORM (Write Once Read Many) model
- Lock the policy for future edits (can no longer be changed)
- Helpful for compliance and data retention



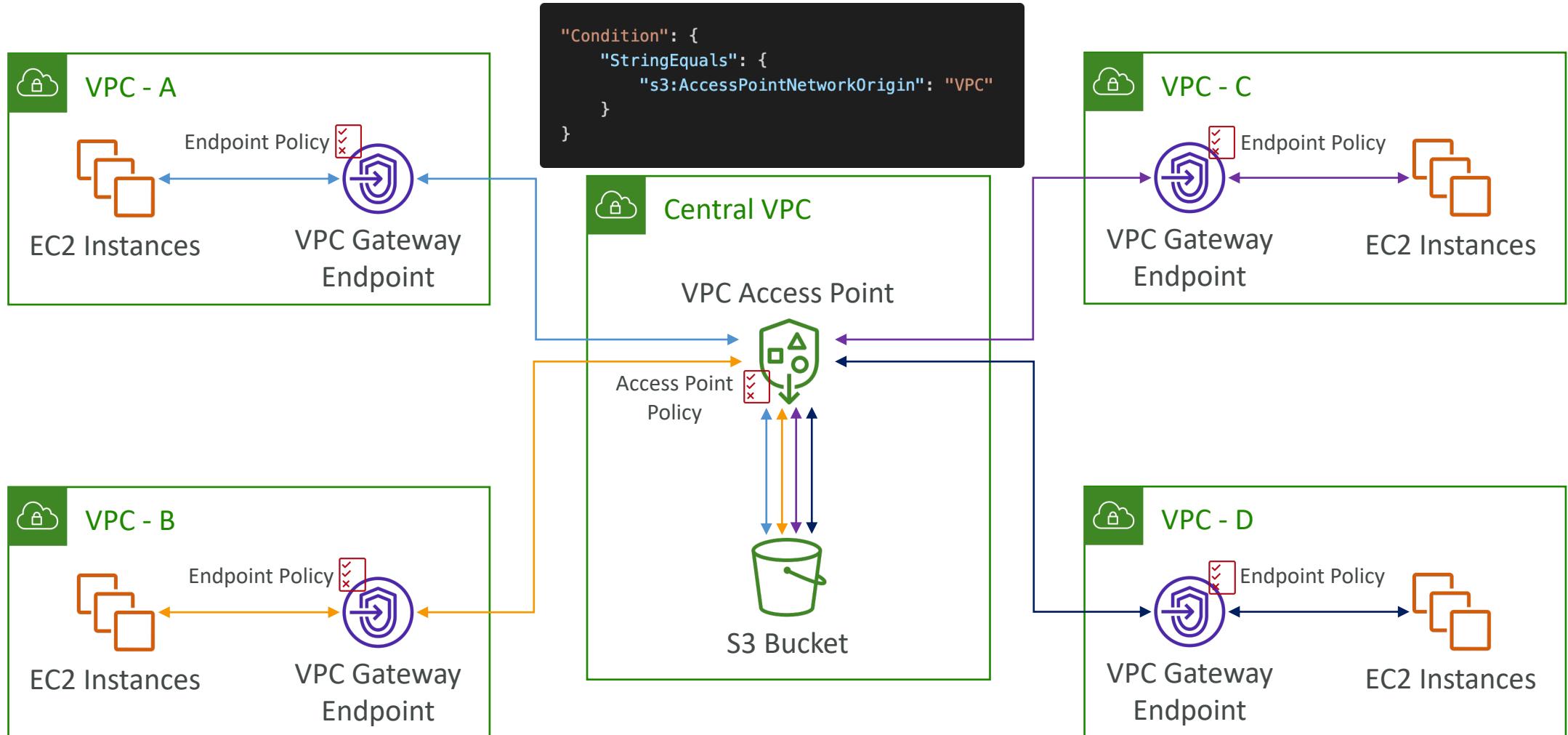


# S3 – Access Points

- Each Access Point gets its own DNS and policy to limit who can access it
  - A specific IAM user / group
  - One policy per Access Point => Easier to manage than complex bucket policies
- Can restrict to traffic from a specific VPC
- Access points are linked to a specific bucket (unique name per acct/region)

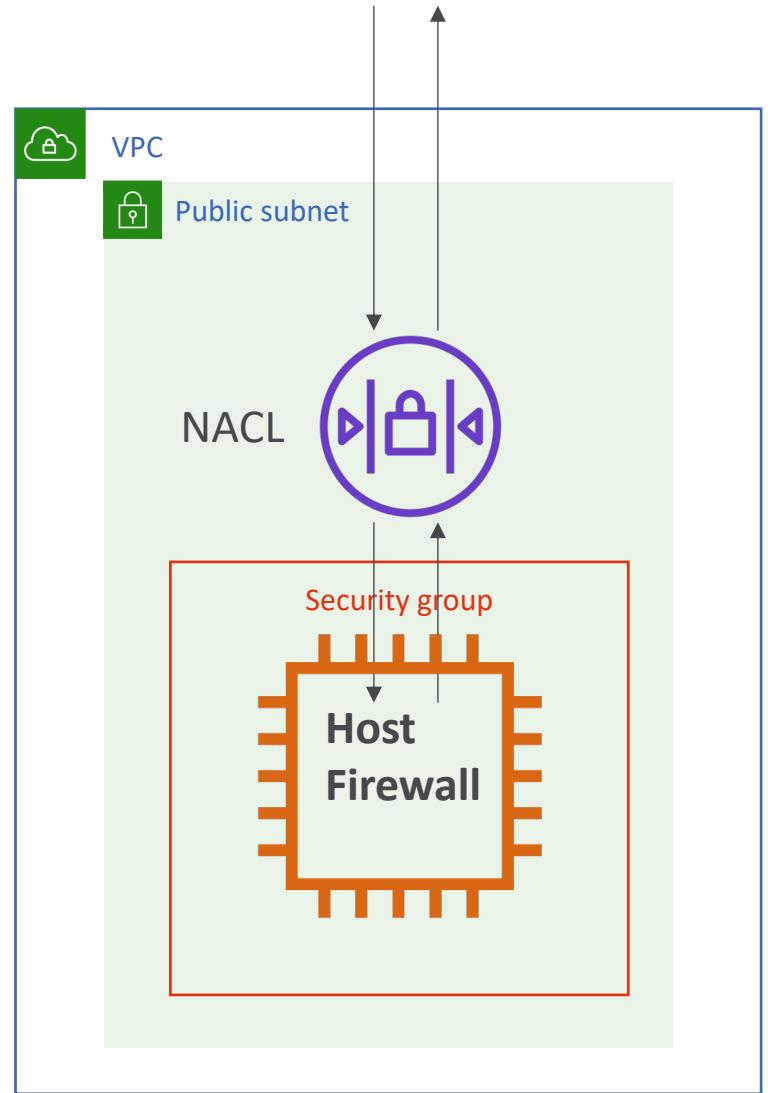


# S3 – Access Points with Shared Bucket



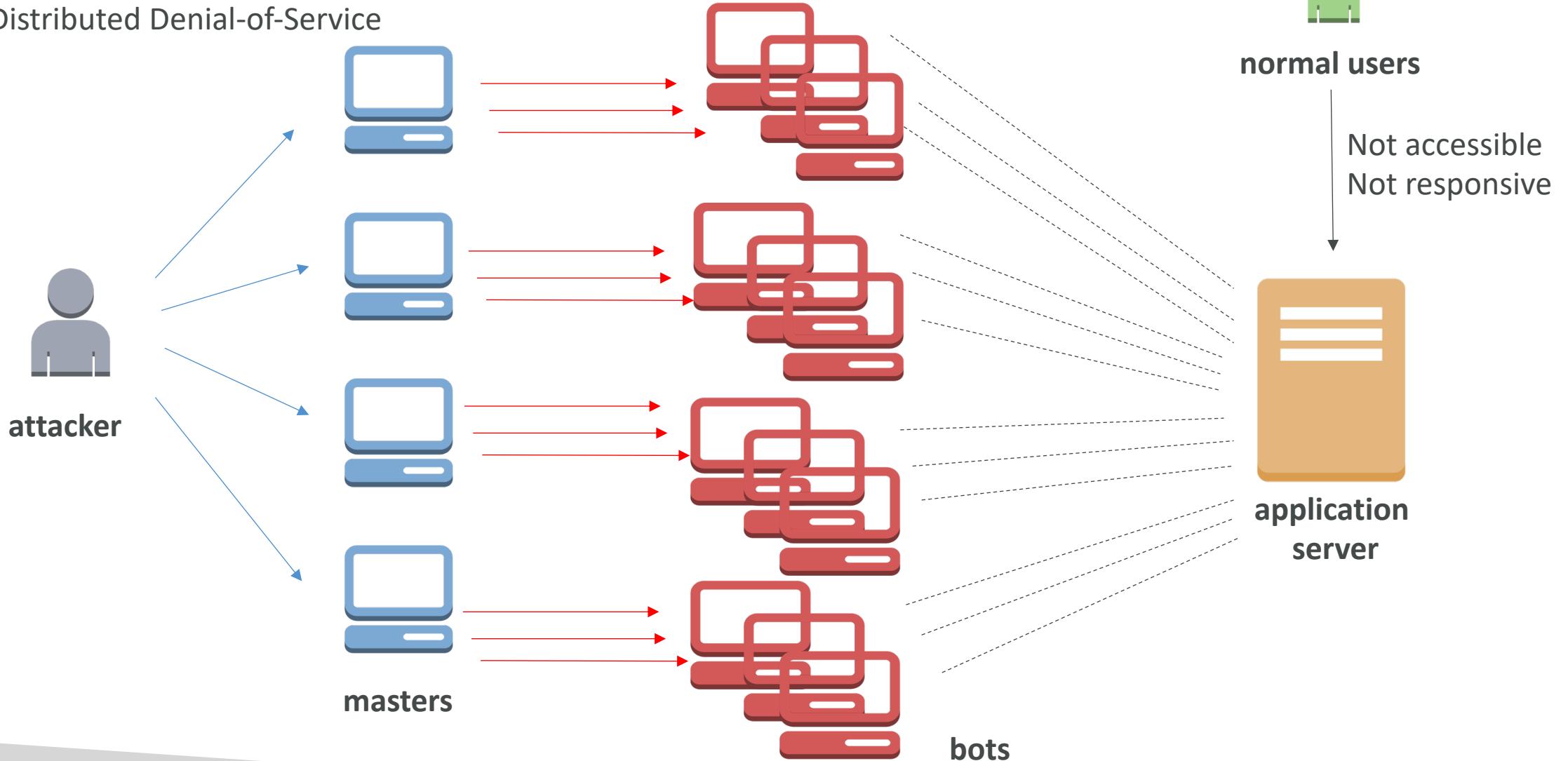
# Network Security

- **Security Groups**
  - Attached to ENI (Elastic Network Interfaces) – EC2, RDS, Lambda in VPC, etc
  - Are stateful (any traffic in is allowed to go out, any traffic out can go back in)
  - Can reference by CIDR and security group id
  - Supports security group references for VPC peering
  - Default: inbound denied, outbound all allowed
- **NACL (Network ACL):**
  - Attached at the subnet level
  - Are stateless (inbound and outbound rules apply for all traffic)
  - Can only reference a CIDR range (no hostname)
  - Default: allow all inbound, allow all outbound
  - New NACL: denies all inbound, denies all outbound
- **Host Firewall**
  - Software based, highly customizable



# What's a DDOS\* Attack?

\*Distributed Denial-of-Service



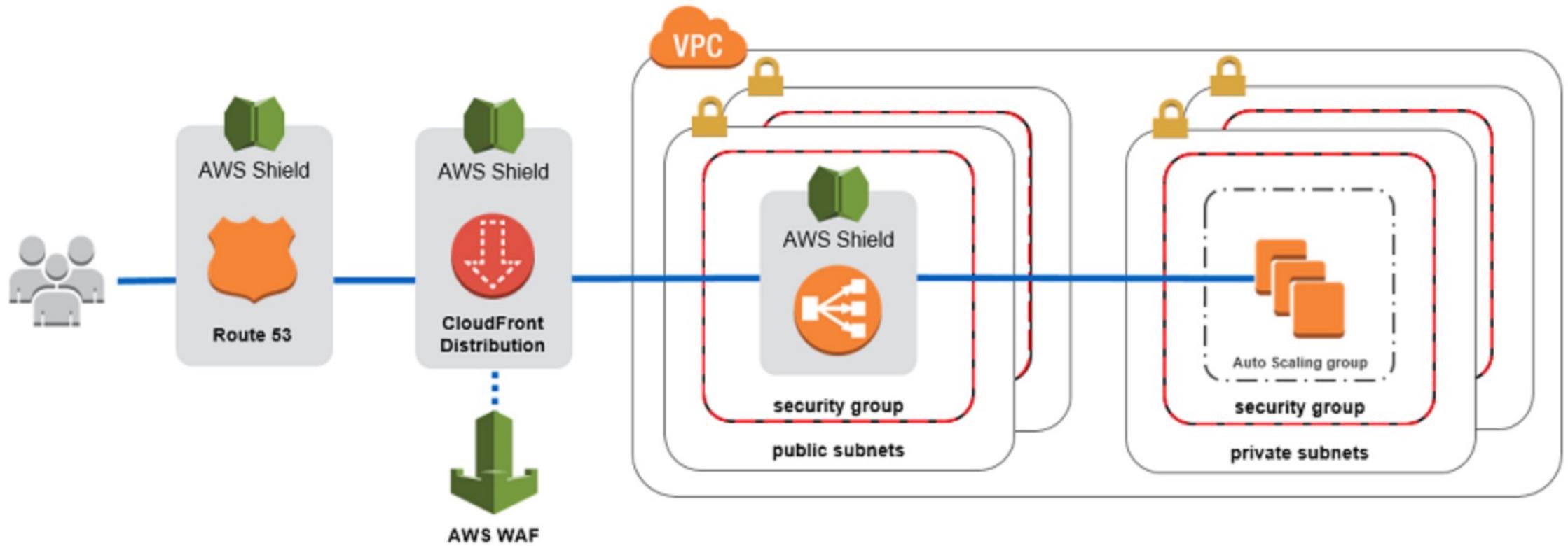
# Type of Attacks on your infrastructure

- Distributed Denial of Service (DDoS):
  - When your service is unavailable because it's receiving too many requests
  - SYN Flood (Layer 4): send too many TCP connection requests
  - UDP Reflection (Layer 4): get other servers to send many big UDP requests
  - DNS flood attack: overwhelm the DNS so legitimate users can't find the site
  - Slow Loris attack: a lot of HTTP connections are opened and maintained
- Application level attacks:
  - more complex, more specific (HTTP level)
  - Cache bursting strategies: overload the backend database by invalidating cache

# DDoS Protection on AWS

- **AWS Shield Standard:** protects against DDoS attack for your website and applications, for all customers at no additional costs
- **AWS Shield Advanced:** 24/7 premium DDoS protection
- **AWS WAF:** Filter specific requests based on rules
- **CloudFront and Route 53:**
  - Availability protection using global edge network
  - Combined with AWS Shield, provides DDoS attack mitigation at the edge
- Be ready to scale – leverage AWS Auto Scaling
- Separate static resources (S3 / CloudFront) from dynamic ones (EC2 / ALB)
- Read the whitepaper for details:  
[https://dl.awsstatic.com/whitepapers/Security/DDoS\\_White\\_Paper.pdf](https://dl.awsstatic.com/whitepapers/Security/DDoS_White_Paper.pdf)

# Sample Reference Architecture



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

# AWS Shield



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

# AWS WAF – Web Application Firewall



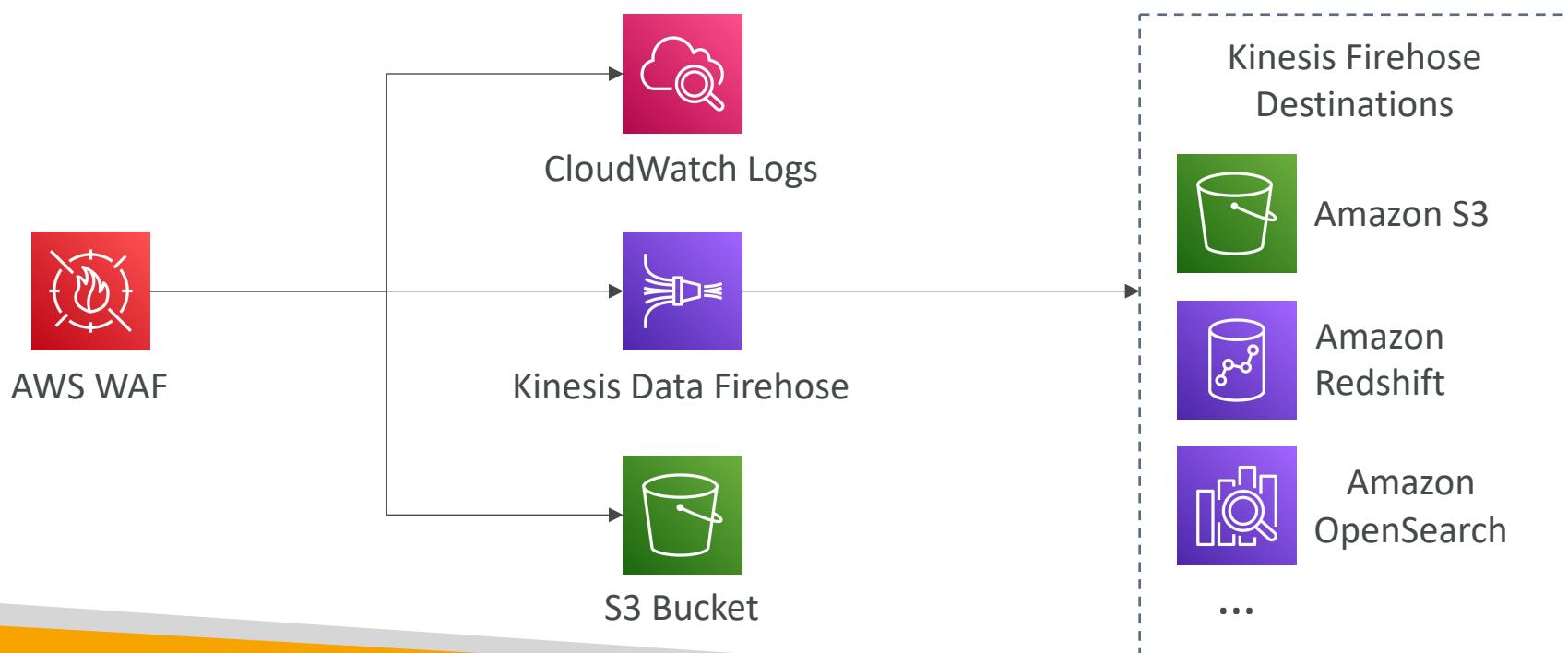
- Protects your web applications from common web exploits (Layer 7)
- Deploy on **Application Load Balancer** (localized rules)
- Deploy on **API Gateway** (rules running at the regional or edge level)
- Deploy on **CloudFront** (rules globally on edge locations)
  - Used to front other solutions: CLB, EC2 instances, custom origins, S3 websites)
- Deploy on AppSync (protect your GraphQL APIs)
- WAF is not for DDoS protection
- Define Web ACL (Web Access Control List):
  - Rules can include **IP addresses**, HTTP headers, HTTP body, or URI strings
  - Protects from common attack - **SQL injection** and Cross-Site Scripting (XSS)
  - Size constraints, Geo match
  - Rate-based rules (to count occurrences of events)

# AWS WAF – Managed Rules

- Library of over 190 managed rules
- Ready-to-use rules that are managed by AWS and AWS Marketplace Sellers
- **Baseline Rule Groups** – general protection from common threats
  - AWSManagedRulesCommonRuleSet, AWSManagedRulesAdminProtectionRuleSet, ...
- **Use-case Specific Rule Groups** – protection for many AWS WAF use cases
  - AWSManagedRulesSQLiRuleSet, AWSManagedRulesWindowsRuleSet, AWSManagedRulesPHPRuleSet, AWSManagedRulesWordPressRuleSet, ...
- **IP Reputation Rule Groups** – block requests based on source (e.g., malicious IPs)
  - AWSManagedRulesAmazonIpReputationList, AWSManagedRulesAnonymousIpList
- **Bot Control Managed Rule Group** – block and manage requests from bots
  - AWSManagedRulesBotControlRuleSet

# WAF - Web ACL – Logging

- You can send your logs to an:
  - Amazon CloudWatch Logs log group – 5 MB per second
  - Amazon Simple Storage Service (Amazon S3) bucket – 5 minutes interval
  - Amazon Kinesis Data Firehose – limited by Firehose quotas

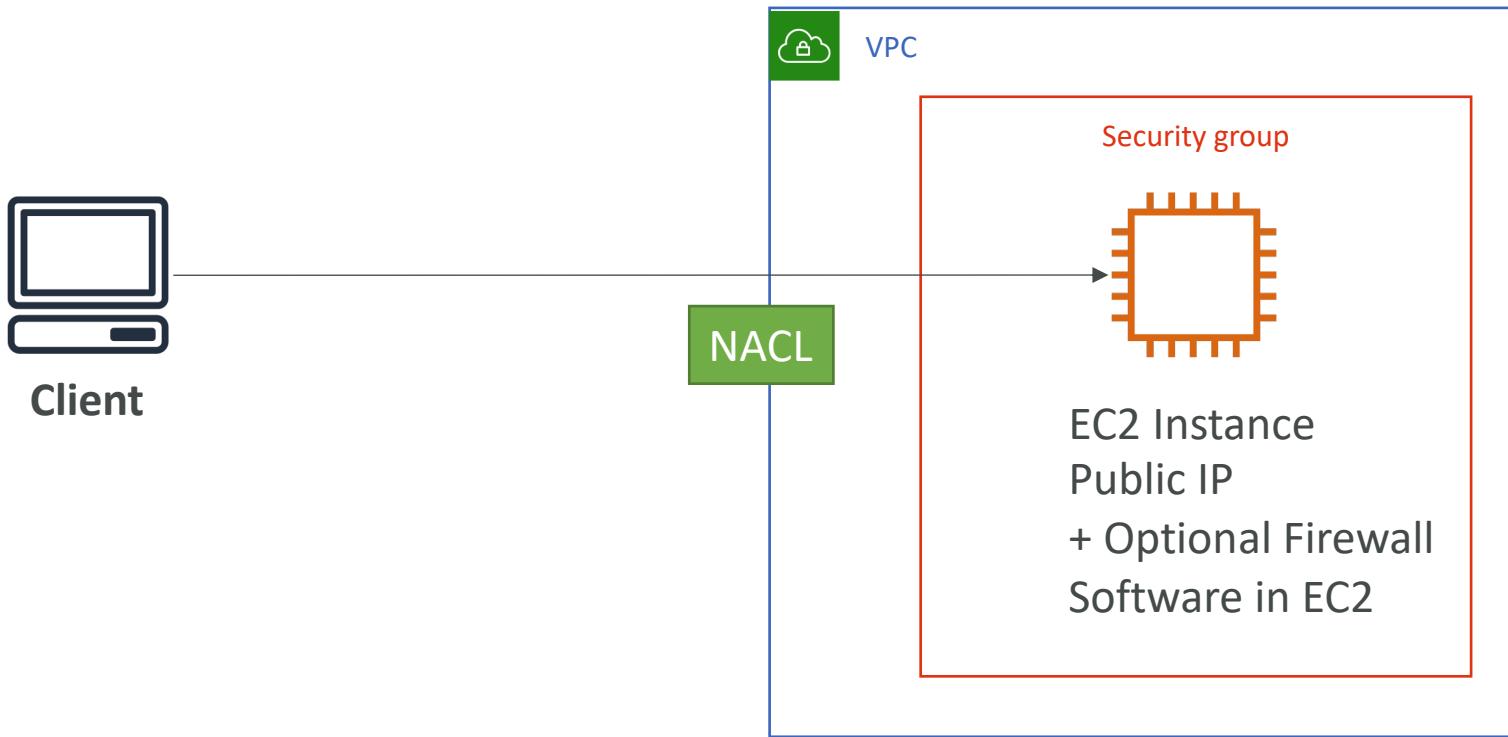


# AWS Firewall Manager

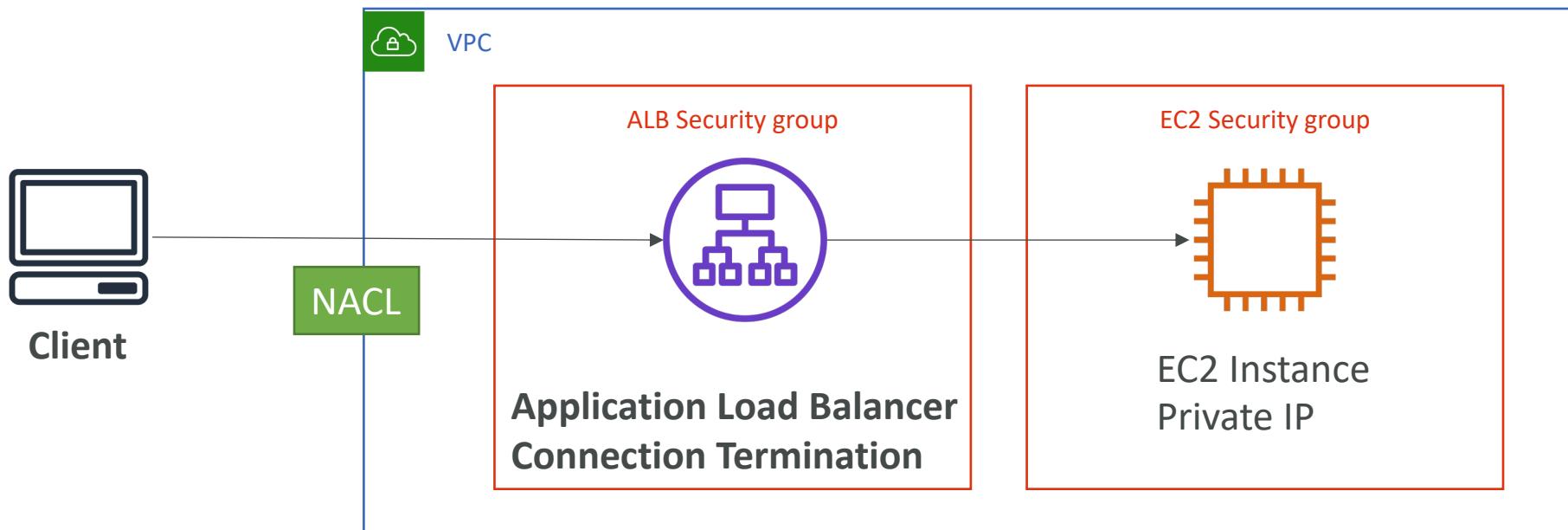


- Manage rules in all accounts of an AWS Organization
- Common set of security rules
- WAF rules (Application Load Balancer, API Gateways, CloudFront)
- AWS Shield Advanced (ALB, CLB, Elastic IP, CloudFront)
- Security Groups for EC2 and ENI resources in VPC

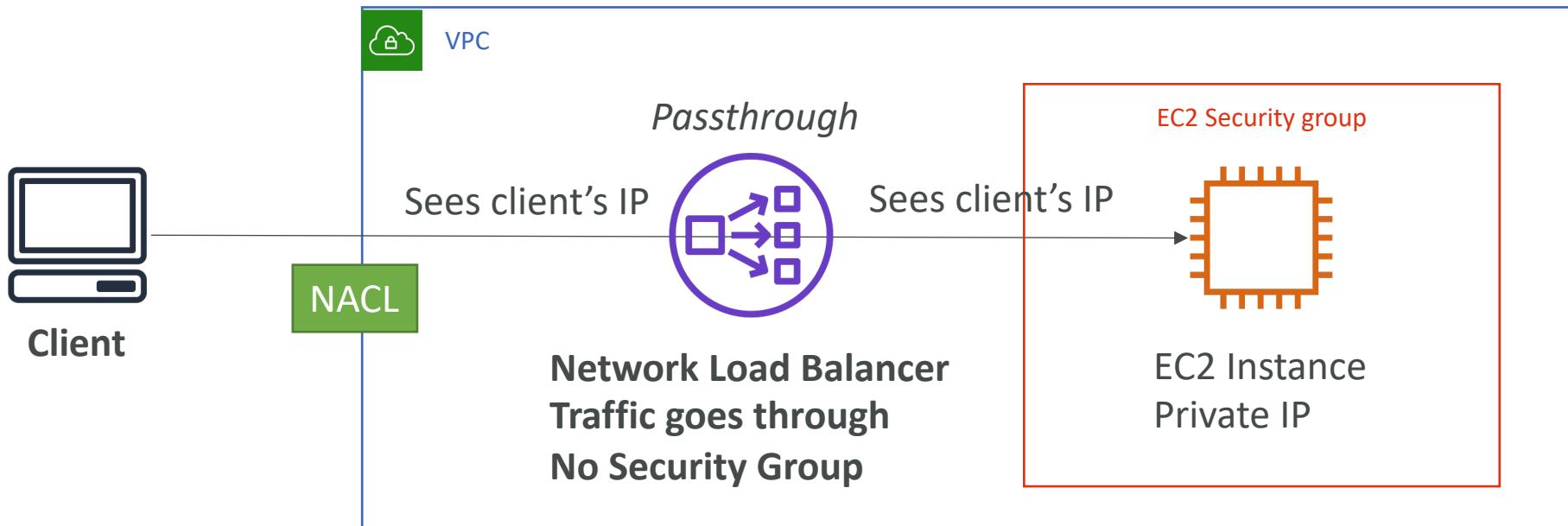
# Blocking an IP address



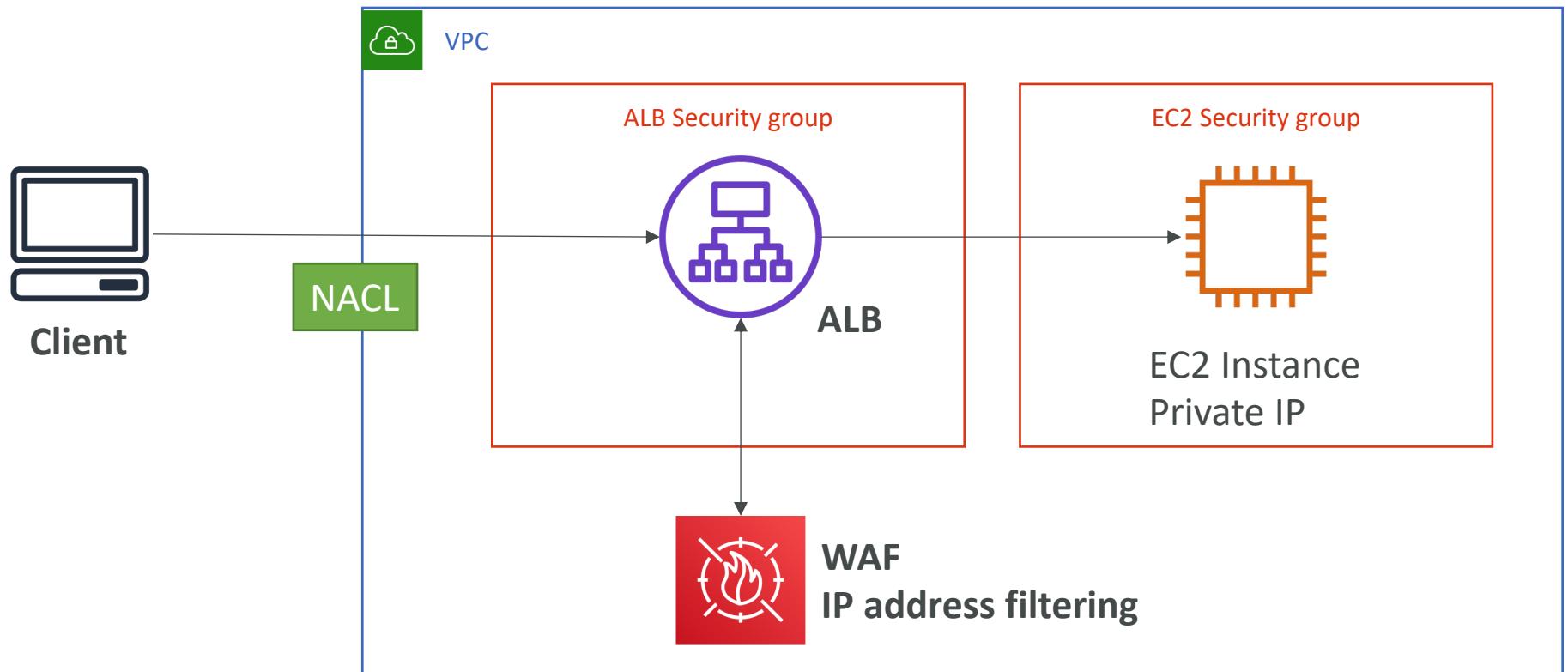
# Blocking an IP address – with an ALB



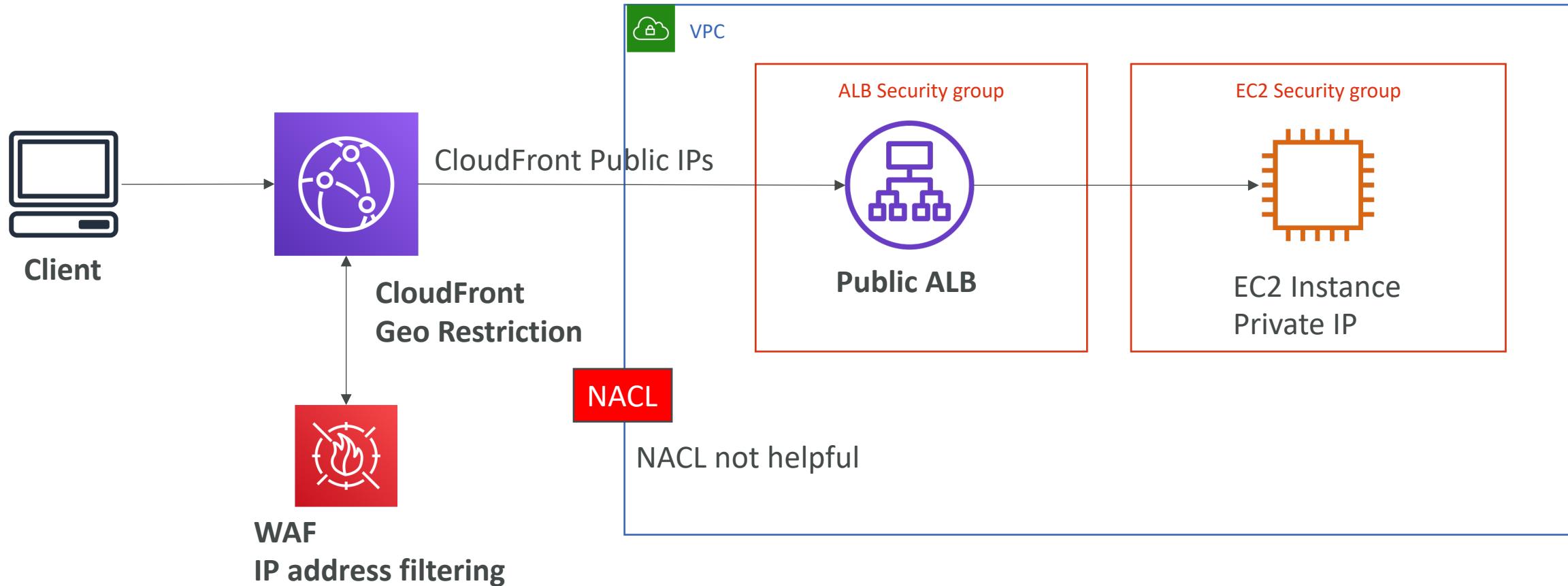
# Blocking an IP address – with an NLB



# Blocking an IP address – ALB + WAF

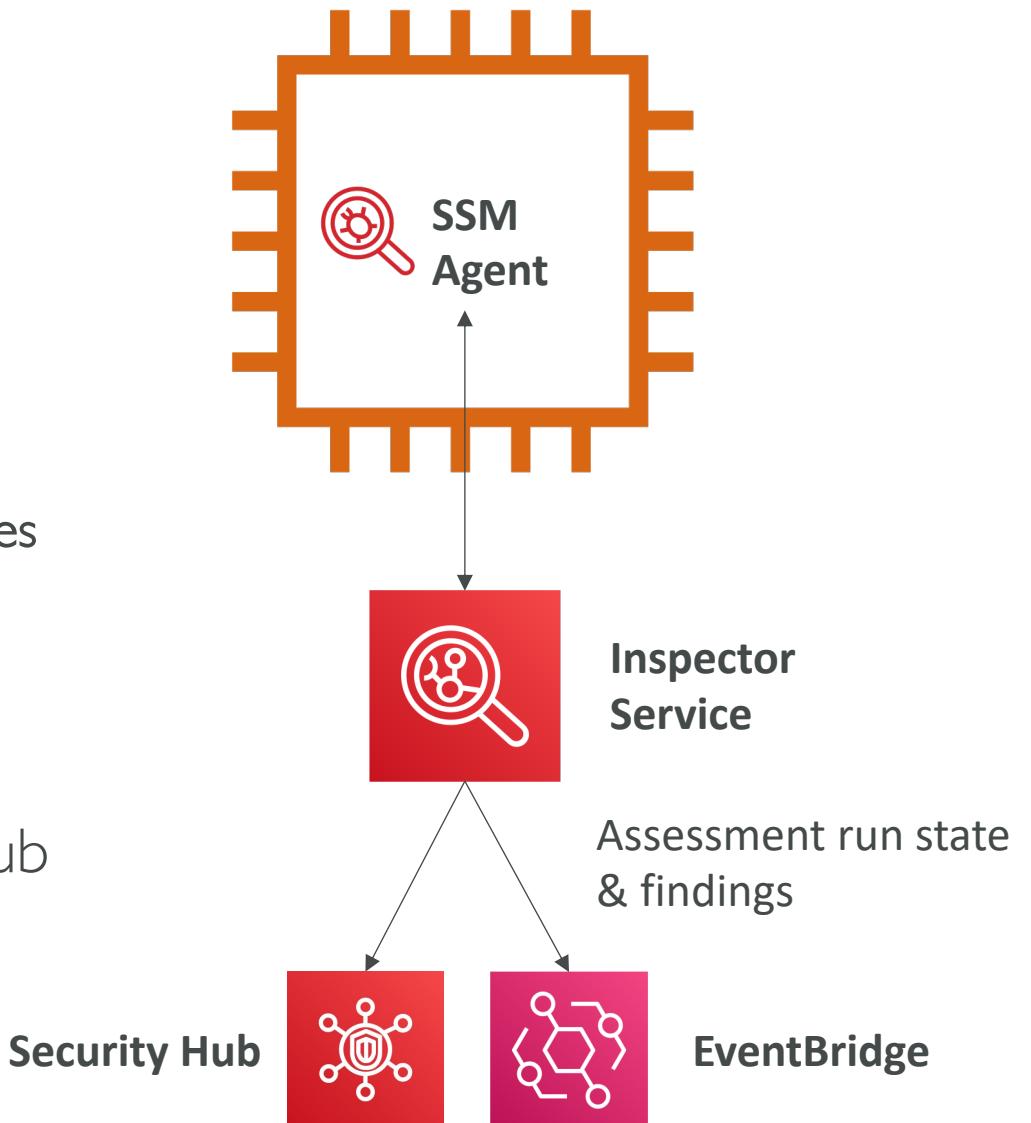


# Blocking an IP address – ALB, CloudFront WAF



# Amazon Inspector

- Automated Security Assessments
- For EC2 instances
  - Leveraging the AWS System Manager (SSM) agent
  - Analyze against unintended network accessibility
  - Analyze the running OS against known vulnerabilities
- For Containers push to Amazon ECR
  - Assessment of containers as they are pushed
- Reporting & integration with AWS Security Hub
- Send findings to Amazon Event Bridge



# What does AWS Inspector evaluate?

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

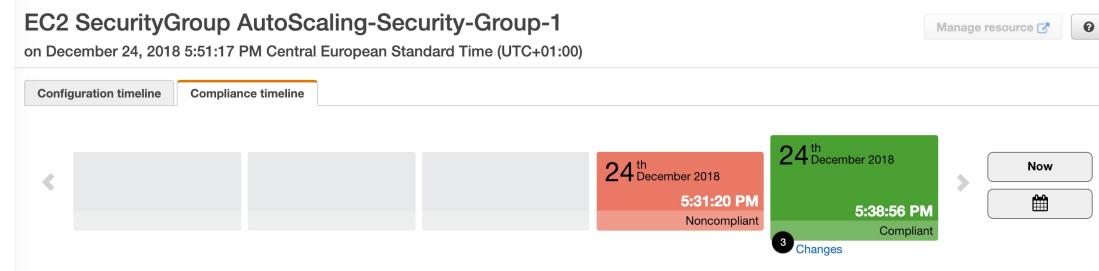


# AWS Config

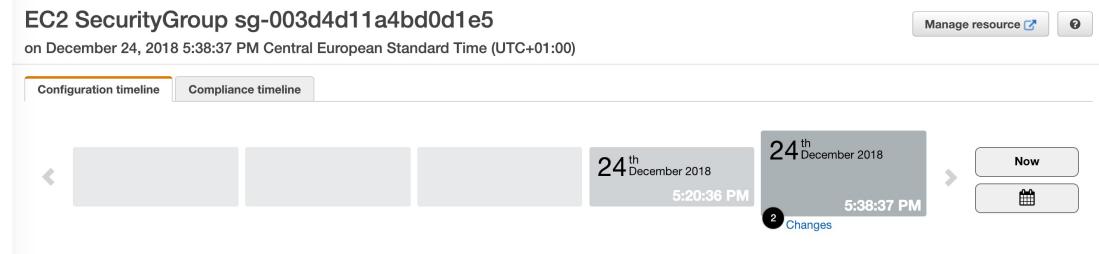
- Helps with auditing and recording **compliance** of your AWS resources
- Helps record configurations and changes over time
- **AWS Config Rules does not prevent actions from happening (no deny)**
- Questions that can be solved by AWS Config:
  - Is there unrestricted SSH access to my security groups?
  - Do my buckets have any public access?
  - How has my ALB configuration changed over time?
- You can receive alerts (SNS notifications) for any changes
- AWS Config is a per-region service
- Can be aggregated across regions and accounts

# AWS Config Resource

- View compliance of a resource over time



- View configuration of a resource over time



- View CloudTrail API calls if enabled

# AWS Config Rules

- Can use AWS managed config rules (over 75)
- Can make custom config rules (must be defined in AWS Lambda)
  - Evaluate if each EBS disk is of type gp2
  - Evaluate if each EC2 instance is t2.micro
- Rules can be evaluated / triggered:
  - For each config change
  - And / or: at regular time intervals
  - Can trigger CloudWatch Events if the rule is non-compliant (and chain with Lambda)
- Rules can have auto remediations:
  - If a resource is not compliant, you can trigger an auto remediation
  - Define the remediation through **SSM Automations**
  - Ex: remediate security group rules, stop instances with non-approved tags

# AWS Managed Logs

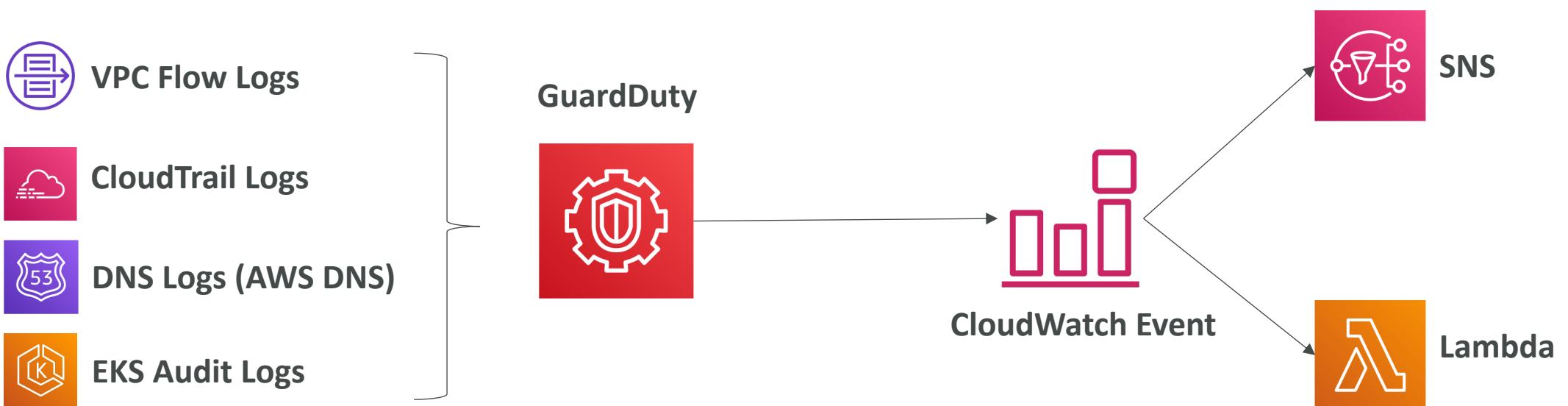
- Load Balancer Access Logs (ALB, NLB, CLB) => to S3
  - Access logs for your Load Balancers
- CloudTrail Logs => to S3 and CloudWatch Logs
  - Logs for API calls made within your account
- VPC Flow Logs => to S3 and CloudWatch Logs
  - Information about IP traffic going to and from network interfaces in your VPC
- Route 53 Access Logs => to CloudWatch Logs
  - Log information about the queries that Route 53 receives
- S3 Access Logs => to S3
  - Server access logging provides detailed records for the requests that are made to a bucket
- CloudFront Access Logs => to S3
  - Detailed information about every user request that CloudFront receives
- AWS Config => to S3



# Amazon GuardDuty

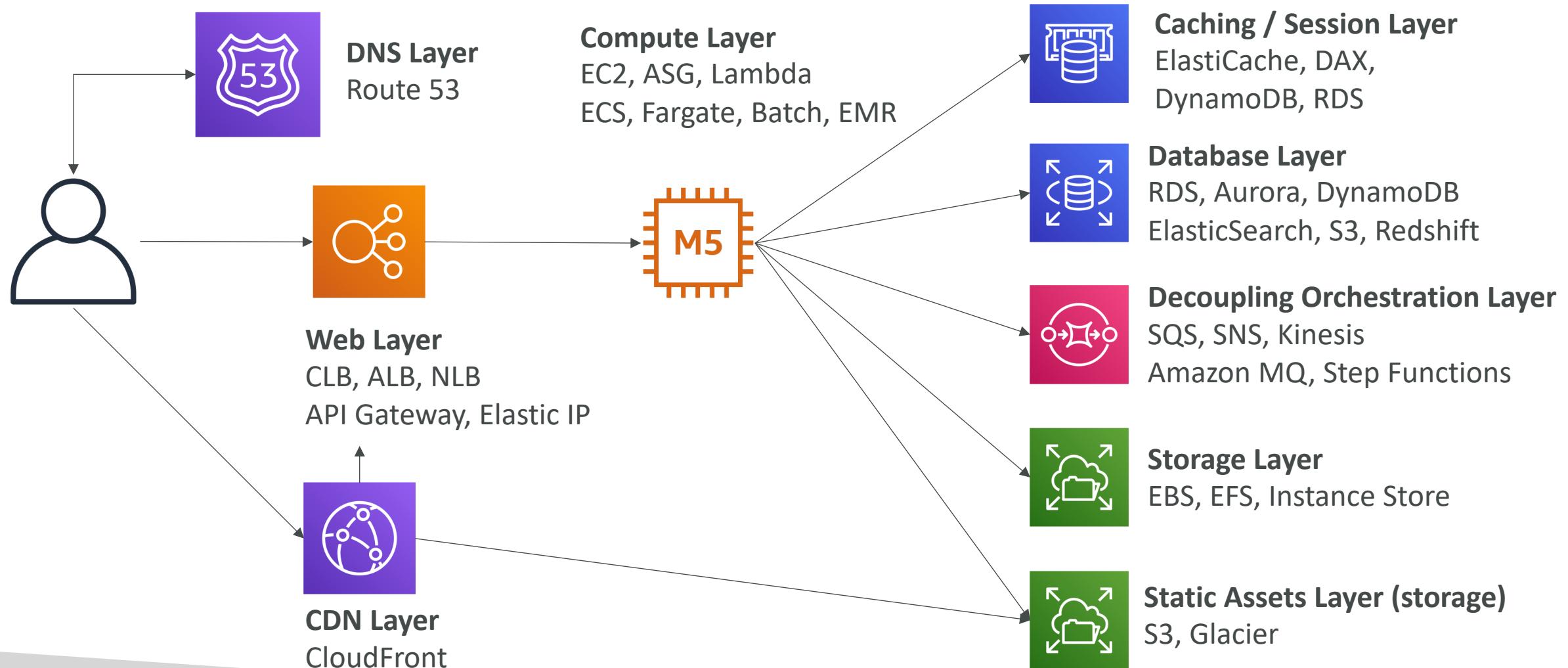
- Intelligent Threat discovery to Protect AWS Account
- Uses Machine Learning algorithms, anomaly detection, 3<sup>rd</sup> party data
- One click to enable (30 days trial), no need to install software
- Input data includes:
  - CloudTrail Events Logs – unusual API calls, unauthorized deployments
    - CloudTrail Management Events – create VPC subnet, create trail, ...
    - CloudTrail S3 Data Events – get object, list objects, delete object, ...
  - VPC Flow Logs – unusual internal traffic, unusual IP address
  - DNS Logs – compromised EC2 instances sending encoded data within DNS queries
  - Kubernetes Audit Logs – suspicious activities and potential EKS cluster compromises
- Can setup **CloudWatch Event rules** to be notified in case of findings
- CloudWatch Events rules can target AWS Lambda or SNS
- Can protect against CryptoCurrency attacks (has a dedicated “finding” for it)

# Amazon GuardDuty



# Compute and Load Balancing Section

# Solution Architecture on AWS



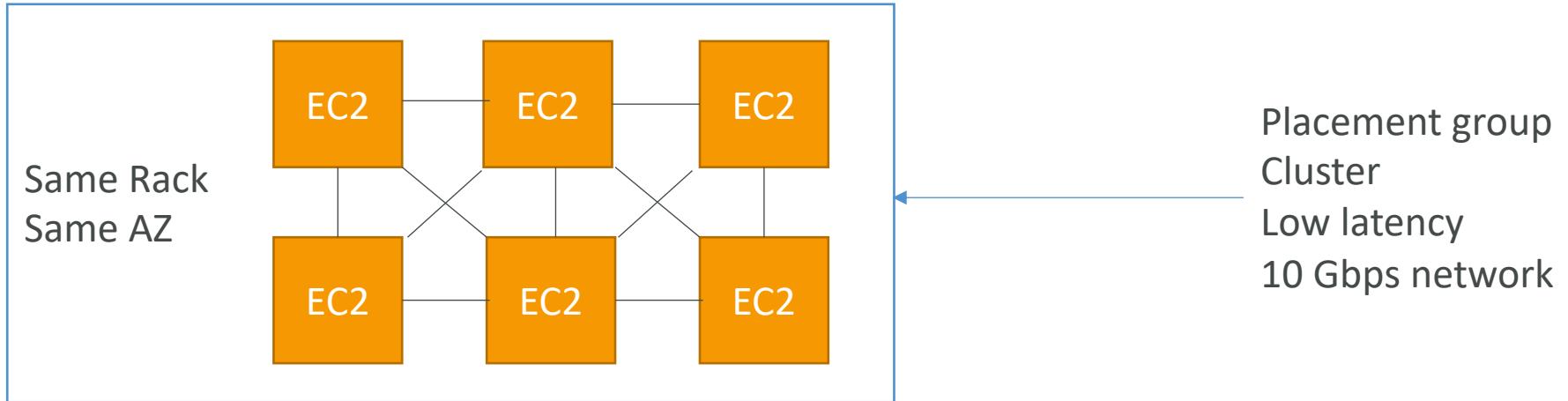
# EC2 Instance Types – Main ones

- R: applications that needs a lot of RAM – in-memory caches
- C: applications that needs good CPU – compute / databases
- M: applications that are balanced (think “medium”) – general / web app
- I: applications that need good local I/O (instance storage) – databases
- G: applications that need a GPU – video rendering / machine learning
- T2 / T3: burstable instances (up to a capacity)
- T2 / T3 - unlimited: unlimited burst
- Real-world tip: use <https://www.ec2instances.info>

# EC2 - Placement Groups

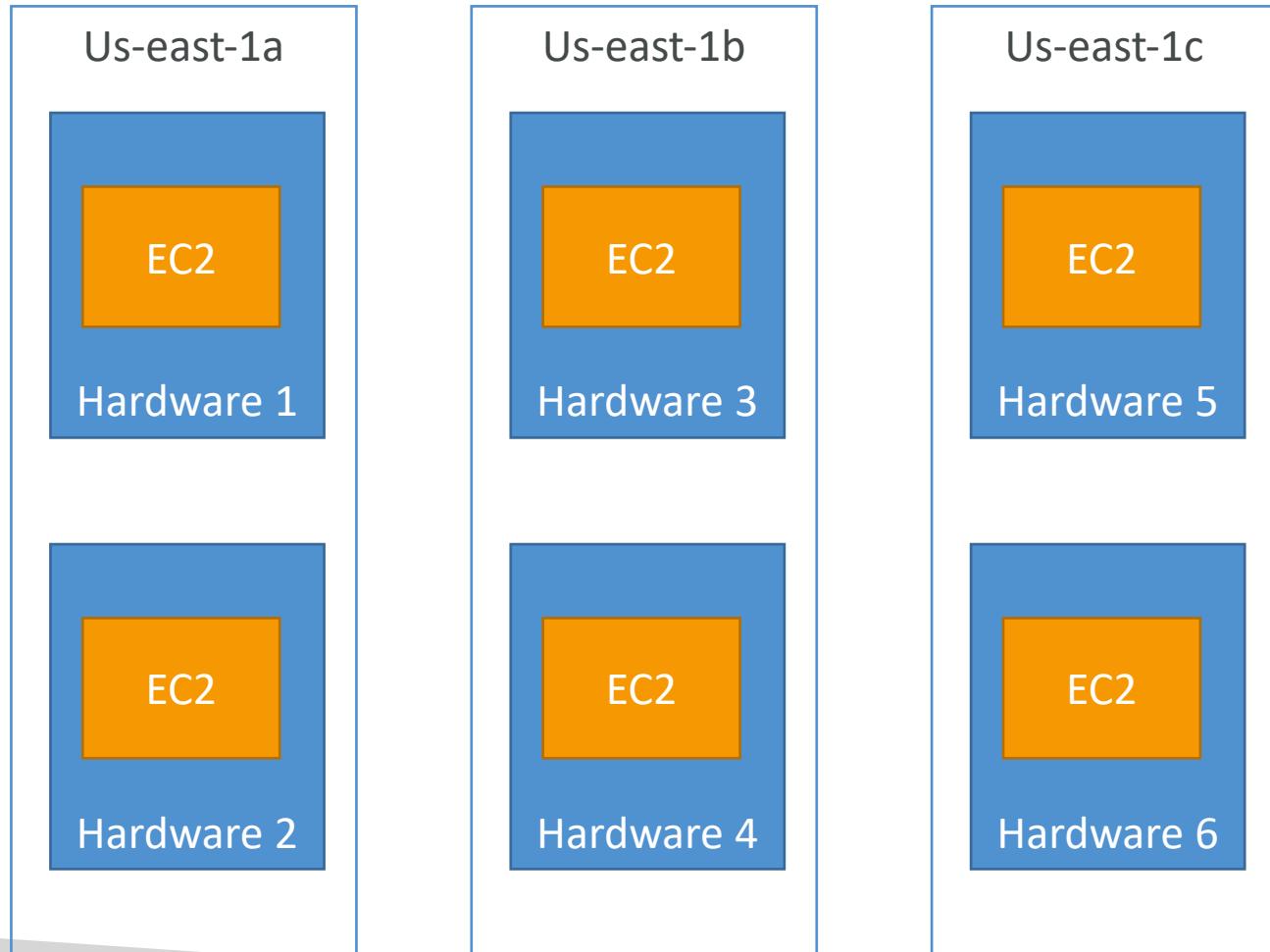
- Control the EC2 Instance placement strategy using placement groups
- Group Strategies:
  - *Cluster*—clusters instances into a low-latency group in a single Availability Zone
  - *Spread*—spreads instances across underlying hardware (max 7 instances per group per AZ) – critical applications
  - *Partition*—spreads instances across many different partitions (which rely on different sets of racks) within an AZ. Scales to 100s of EC2 instances per group (Hadoop, Cassandra, Kafka)
- You can move an instance into or out of a placement group
  - You first need to stop it
  - You then need to use the CLI (modify-instance-placement)
  - You can then start your instance

# Placement Groups Cluster



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

# Placement Groups Spread



- Pros:

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

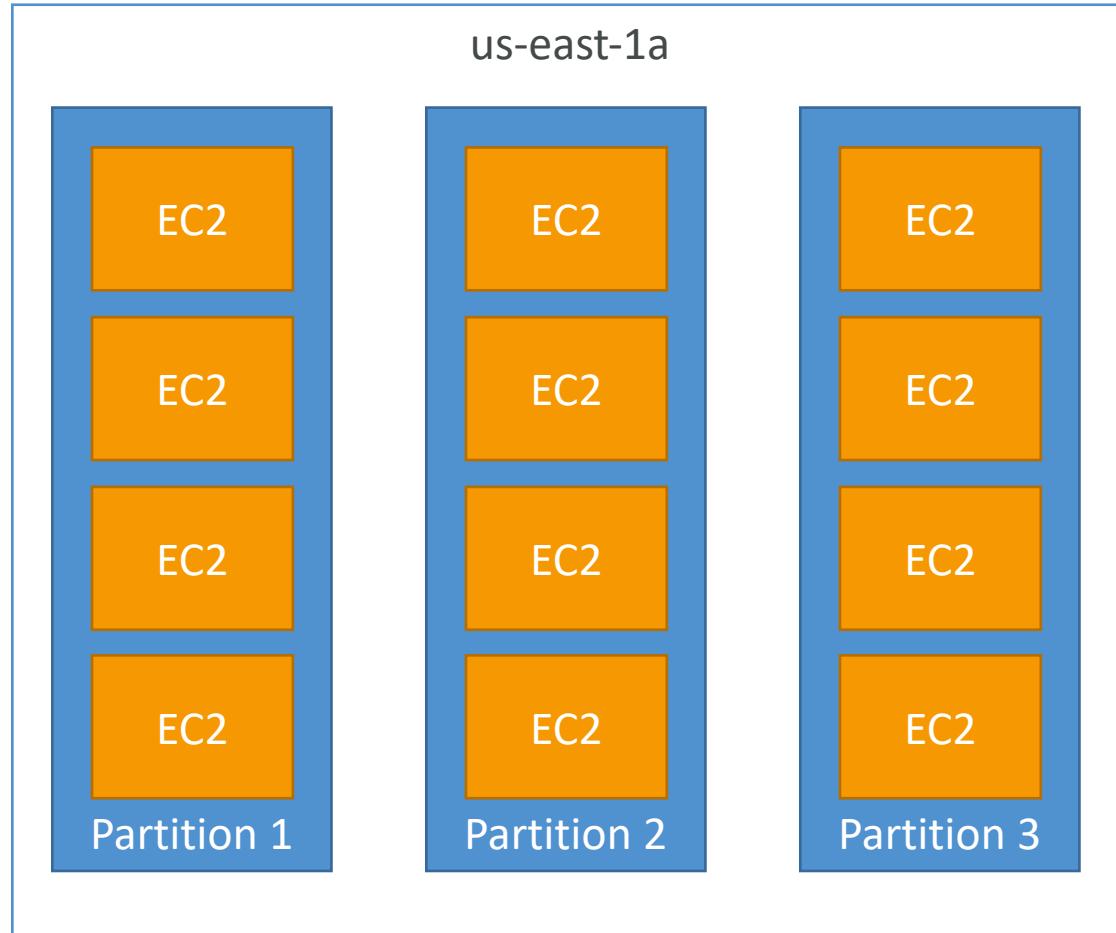
- Cons:

- Limited to 7 instances per AZ per placement group

- Use case:

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

# Placements Groups Partition



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

# EC2 Instance Launch Types

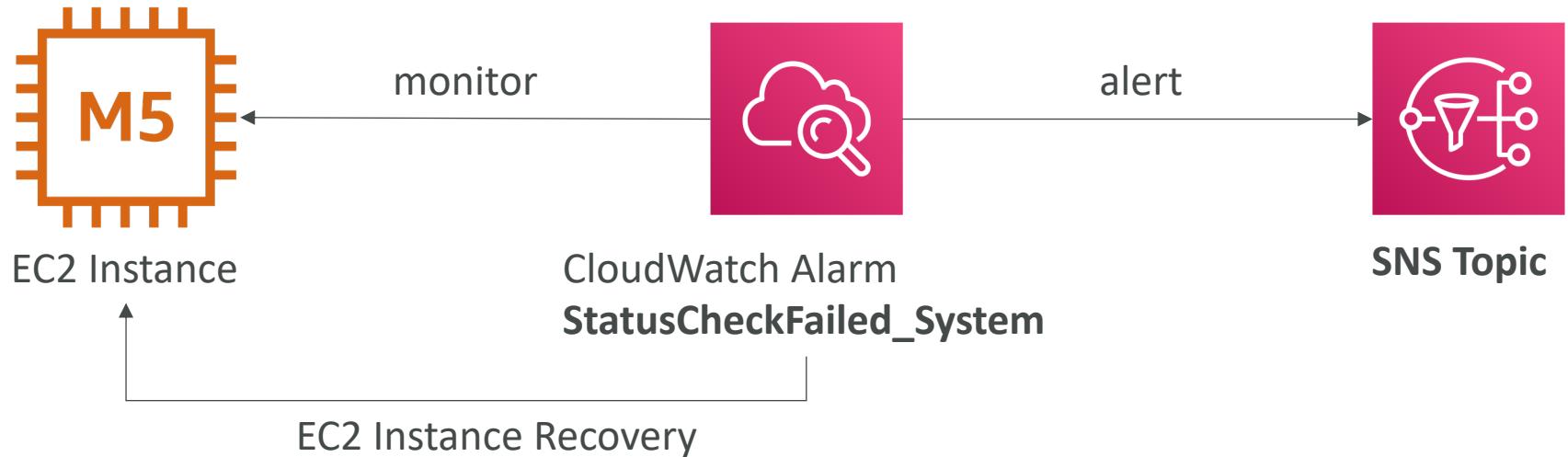
- **On Demand Instances:** short workload, predictable pricing, reliable
- **Spot Instances:** short workloads, for cheap, can lose instances (not reliable)
- **Reserved:** (MINIMUM 1 year)
  - Reserved Instances: long workloads
  - Convertible Reserved Instances: long workloads with flexible instances
- **Dedicated Instances:** no other customers will share your hardware
- **Dedicated Hosts:** book an entire physical server, control instance placement
  - Great for software licenses that operate at the core, or CPU socket level
  - Can define **host affinity** so that instance reboots are kept on the same host

# EC2 included metrics

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

# EC2 Instance Recovery

- Status Check:
  - Instance status = check the EC2 VM
  - System status = check the underlying hardware



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

# High Performance Computing (HPC)

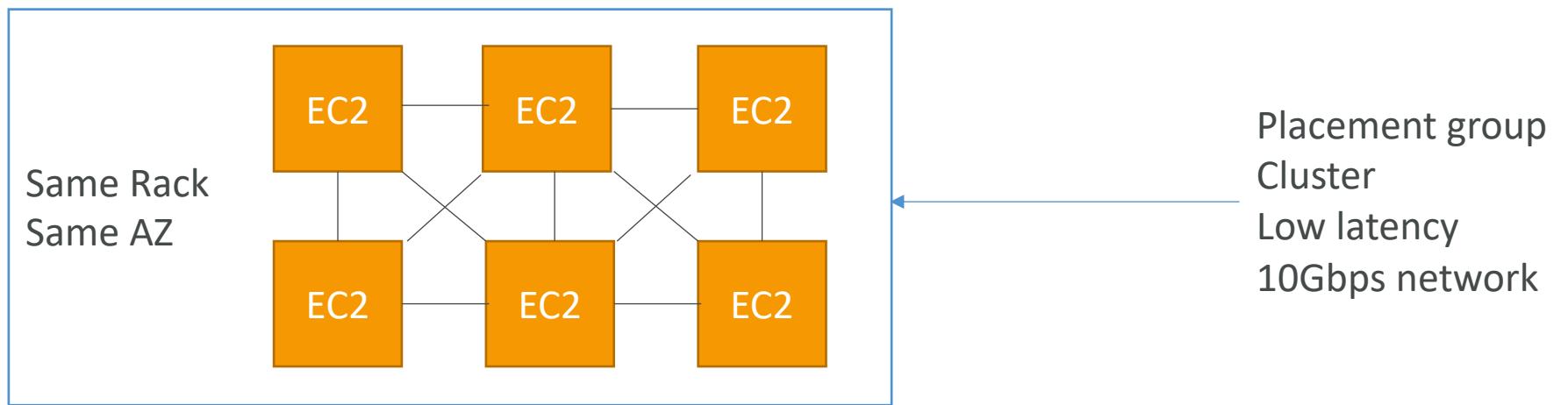
- The cloud is the perfect place to perform HPC
- You can create a very high number of resources in no time
- You can speed up time to results by adding more resources
- You can pay only for the systems you have used
- Perform genomics, computational chemistry, financial risk modeling, weather prediction, machine learning, deep learning, autonomous driving
- Which services help perform HPC?

# Data Management & Transfer

- AWS Direct Connect:
  - Move GB/s of data to the cloud, over a private secure network
- Snowball & Snowmobile
  - Move PB of data to the cloud
- AWS DataSync
  - Move large amount of data between on-premise and S3, EFS, FSx for Windows

# Compute and Networking

- EC2 Instances:
  - CPU optimized, GPU optimized
  - Spot Instances / Spot Fleets for cost savings + Auto Scaling
- EC2 Placement Groups: **Cluster** for good network performance



# Compute and Networking

- EC2 Enhanced Networking (SR-IOV)
  - Higher bandwidth, higher PPS (packet per second), lower latency
  - Option 1: Elastic Network Adapter (ENA) up to 100 Gbps
  - Option 2: Intel 82599 VF up to 10 Gbps – LEGACY
- Elastic Fabric Adapter (EFA)
  - Improved ENA for HPC, only works for Linux
  - Great for inter-node communications, **tightly coupled workloads**
  - Leverages Message Passing Interface (MPI) standard
  - Bypasses the underlying Linux OS to provide low-latency, reliable transport

# Storage

- Instance-attached storage:
  - EBS: scale up to 256,000 IOPS with io2 Block Express
  - Instance Store: scale to millions of IOPS, linked to EC2 instance, low latency
- Network storage:
  - Amazon S3: large blob, not a file system
  - Amazon EFS: scale IOPS based on total size, or use provisioned IOPS
  - Amazon FSx for Lustre:
    - HPC optimized distributed file system, millions of IOPS
    - Backed by S3

# Automation and Orchestration

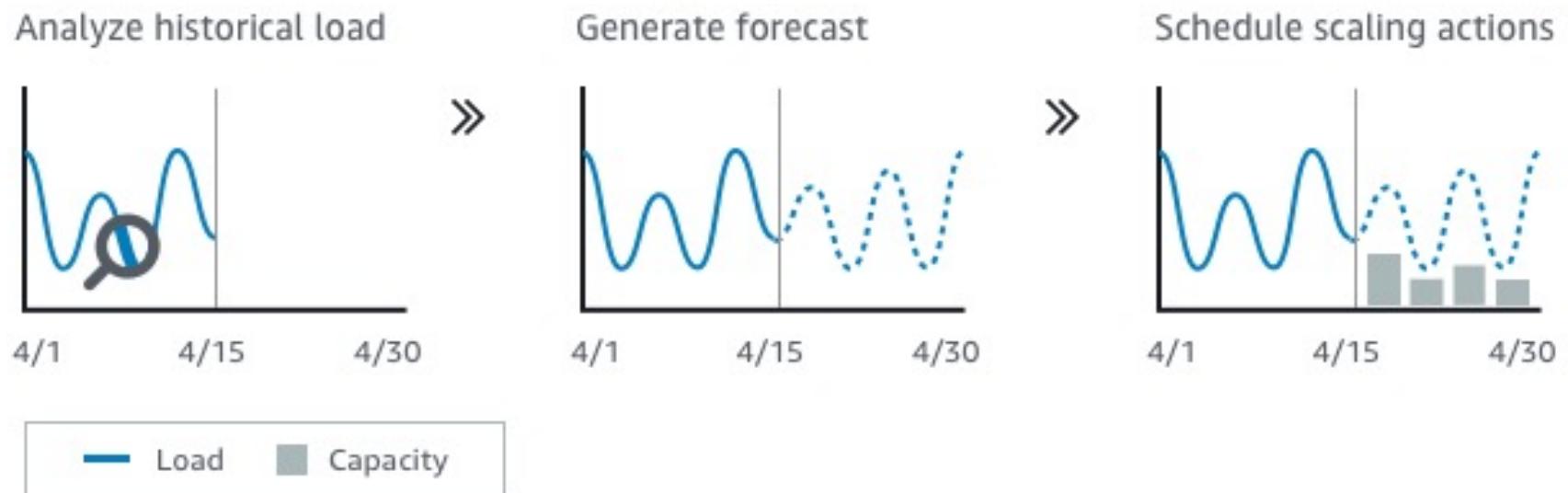
- AWS Batch
  - AWS Batch supports multi-node parallel jobs, which enables you to run single jobs that span multiple EC2 instances.
  - Easily schedule jobs and launch EC2 instances accordingly
- AWS ParallelCluster
  - Open source cluster management tool to deploy HPC on AWS
  - Configure with text files
  - Automate creation of VPC, Subnet, cluster type and instance types

# Auto Scaling Groups – Dynamic Scaling Policies

- Target Tracking Scaling
  - Most simple and easy to set-up
  - Example: I want the average ASG CPU to stay at around 40%
- Simple / Step Scaling
  - When a CloudWatch alarm is triggered (example CPU > 70%), then add 2 units
  - When a CloudWatch alarm is triggered (example CPU < 30%), then remove 1
- Scheduled Actions
  - Anticipate a scaling based on known usage patterns
  - Example: increase the min capacity to 10 at 5 pm on Fridays

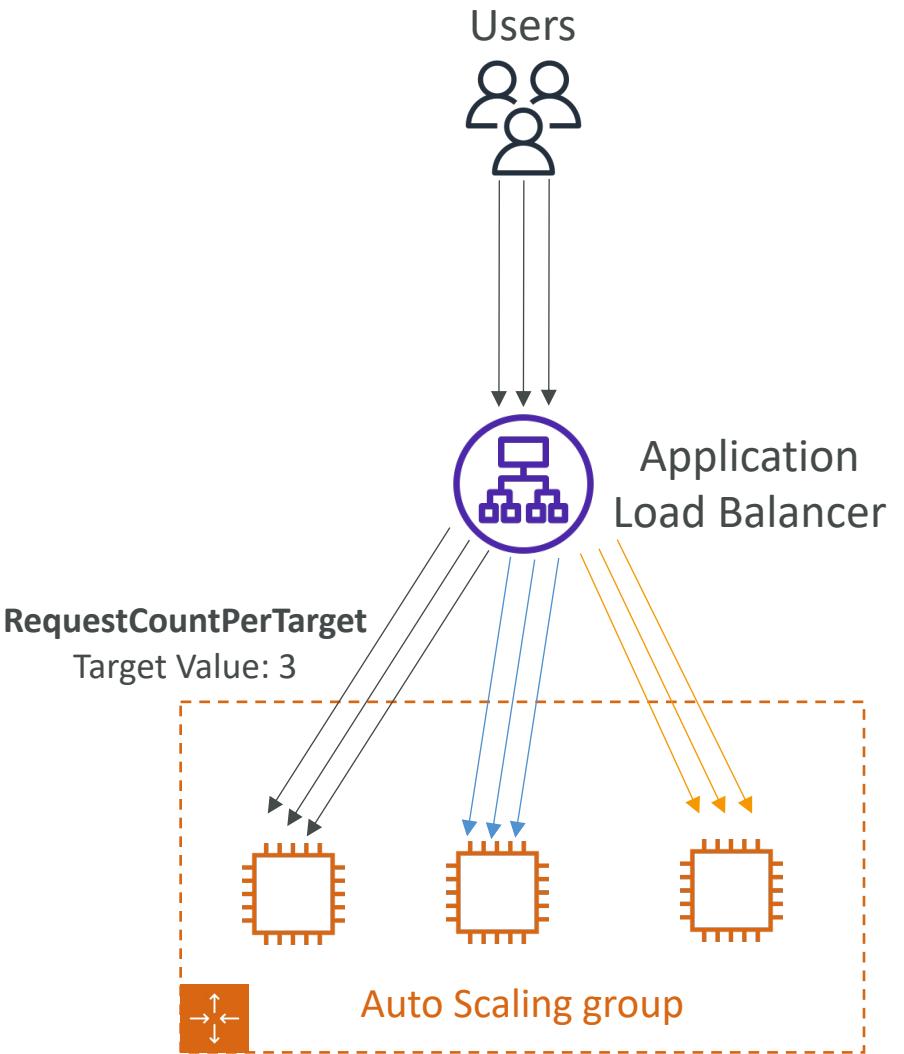
# Auto Scaling Groups – Predictive Scaling

- Predictive scaling: continuously forecast load and schedule scaling ahead



# Good metrics to scale on

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

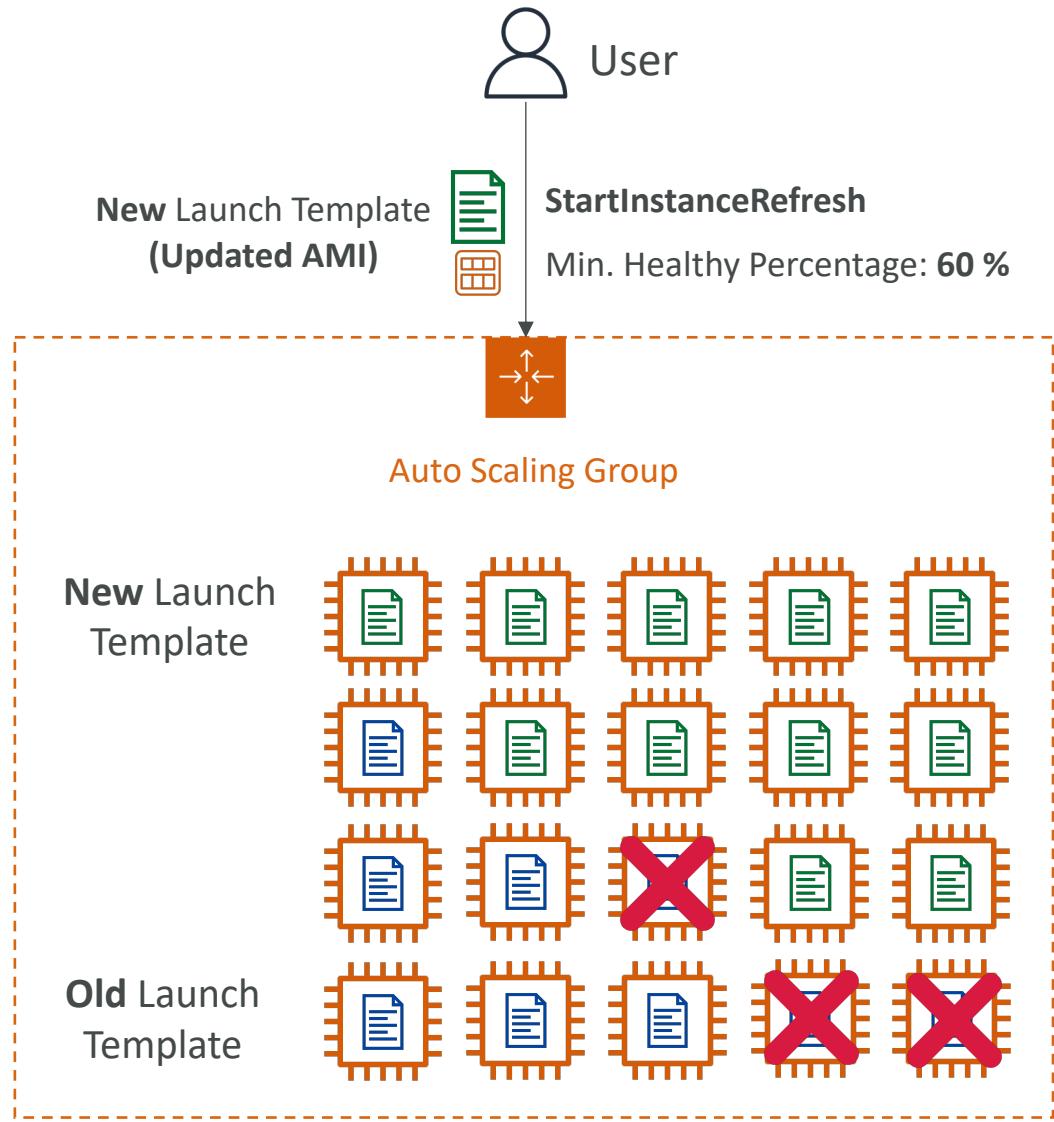


# Auto Scaling – Good to know

- Spot Fleet support (mix on Spot and On-Demand instances)
- **Lifecycle Hooks:**
  - Perform actions before an instance is in service, or before it is terminated
  - Examples: cleanup, log extraction, special health checks
- To upgrade an AMI, must update the **launch configuration / template**
  - Then terminate instances manually (CloudFormation can help)
  - Or use EC2 Instance Refresh for Auto Scaling

# Auto Scaling – Instance Refresh

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



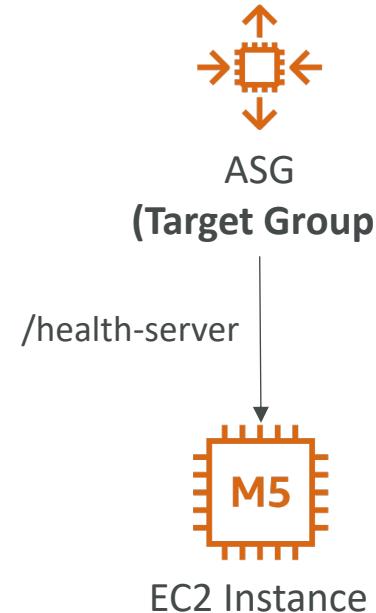
# Auto Scaling – Scaling Processes

- **Launch:** Add a new EC2 to the group, increasing the capacity
- **Terminate:** Removes an EC2 instance from the group, decreasing its capacity.
- **HealthCheck:** Checks the health of the instances
- **ReplaceUnhealthy:** Terminate unhealthy instances and re-create them
- **AZRebalance:** Balancer the number of EC2 instances across AZ
- **AlarmNotification:** Accept notification from CloudWatch
- **ScheduledActions:** Performs scheduled actions that you create.
- **AddToLoadBalancer:** Adds instances to the load balancer or target group
- **InstanceRefresh:** Perform an instance refresh
- We can suspend these processes!

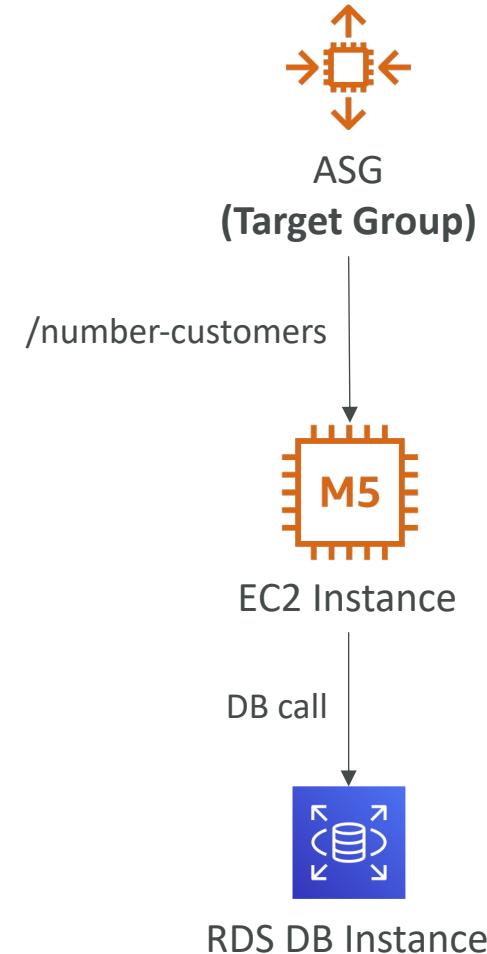
# Auto Scaling – Health Checks

- Health checks available:
  - EC2 Status Checks
  - ELB Health Checks (HTTP)
  - Custom Health Checks – send instance's health to an ASG using AWS CLI or AWS SDK (`set-instance-health`)
- ASG will launch a new instance after terminating an unhealthy one
- Make sure the health check is simple and checks the correct thing

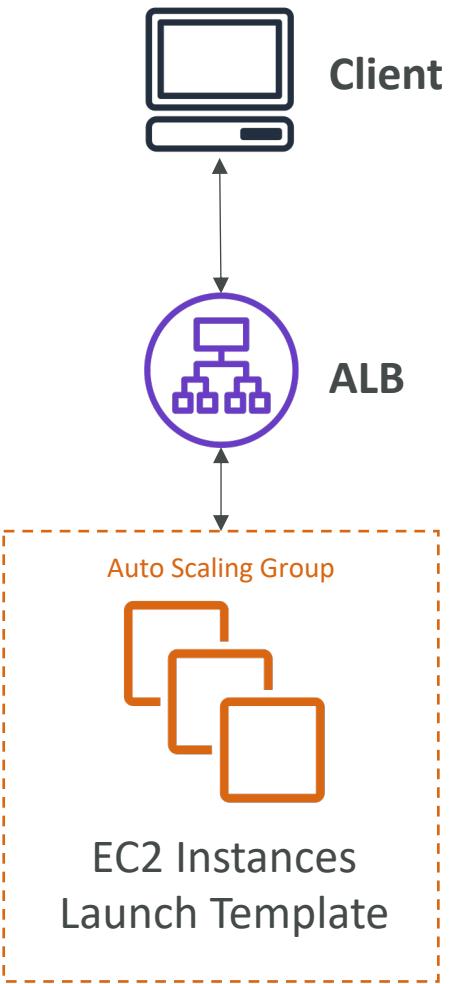
## GOOD HEALTH CHECK



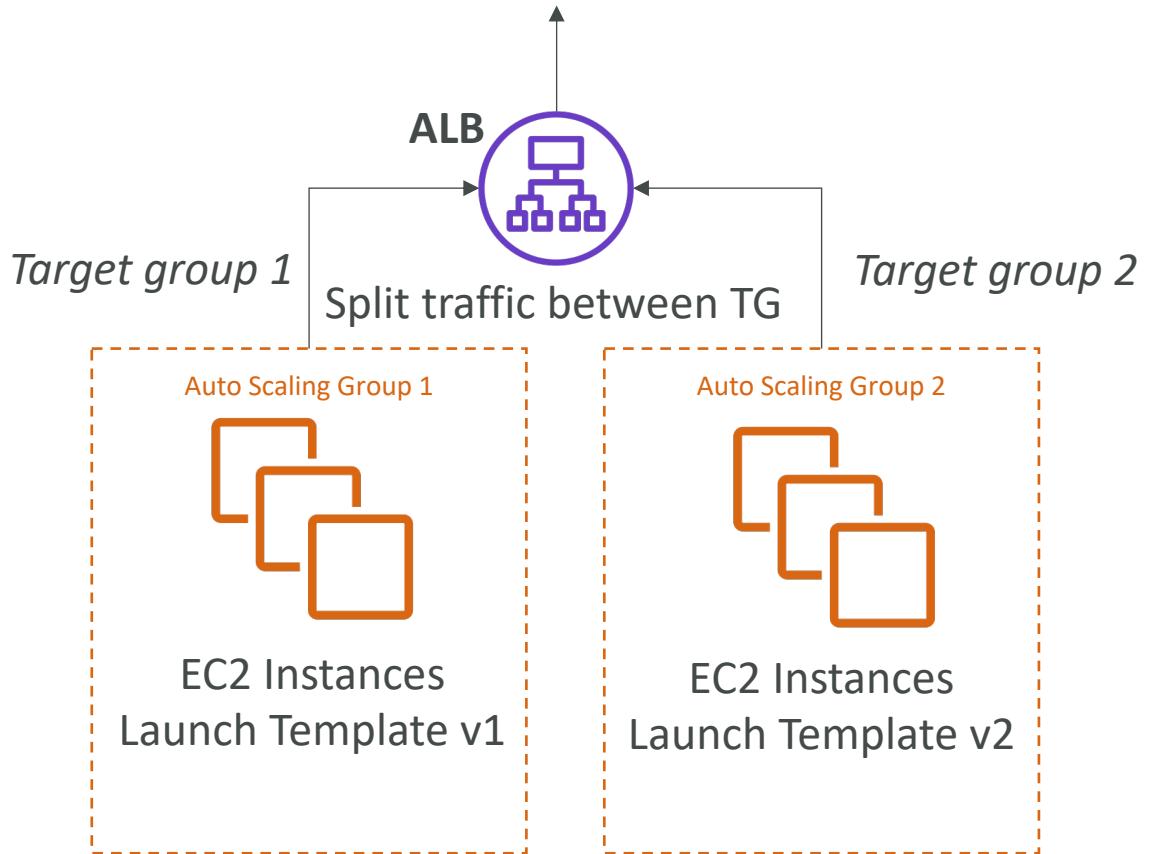
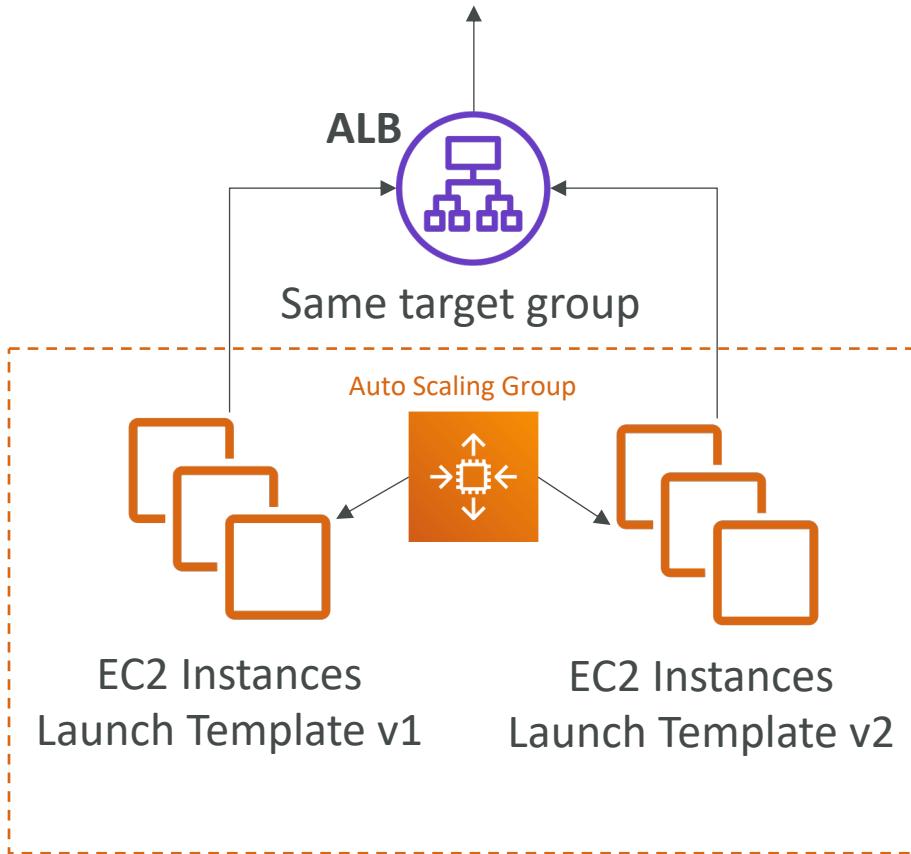
## BAD HEALTH CHECK



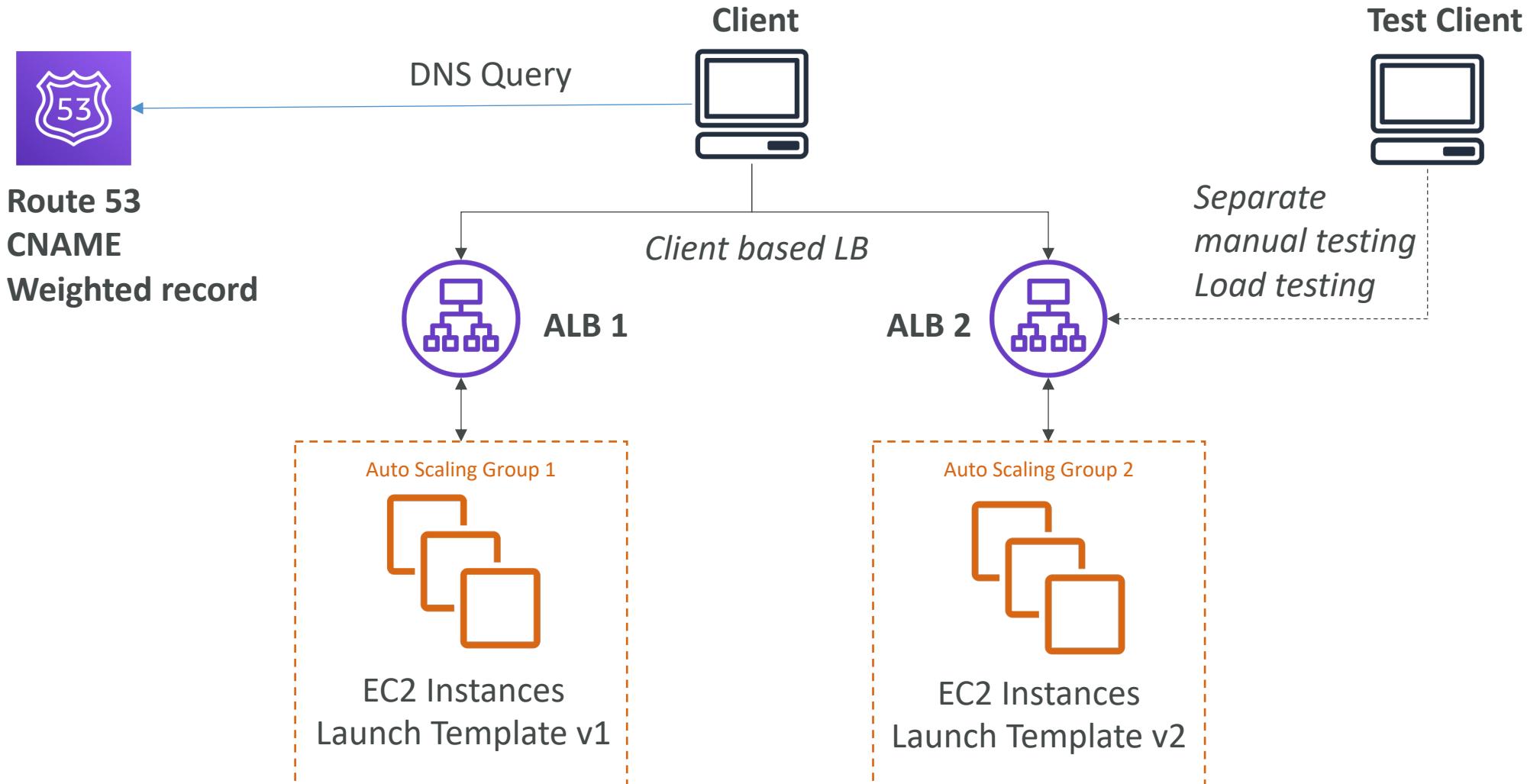
# Auto Scaling – Updating an application



# Auto Scaling – Solution Architecture



# Auto Scaling – Solution Architecture

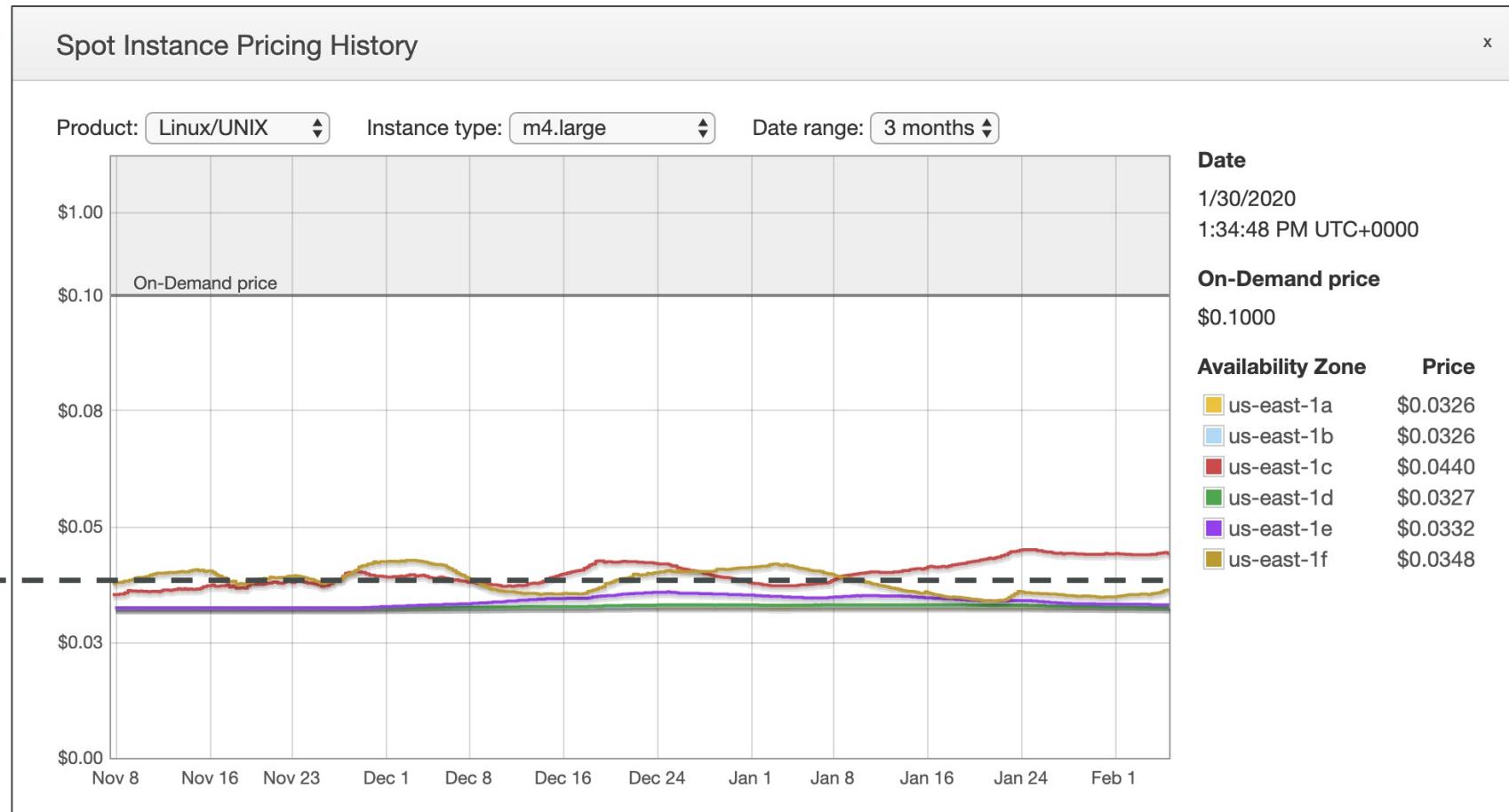




# EC2 Spot Instances

- Can get a discount of up to 90% compared to On-Demand
- Define **max spot price** and get the instance while **current spot price < max**
  - The hourly spot price varies based on offer and capacity
  - If the current spot price > your max price you can choose to **stop** or **terminate** your instance with a 2 minutes grace period.
- Used for batch jobs, data analysis, or workloads that are resilient to failures.
- Not great for critical jobs or databases

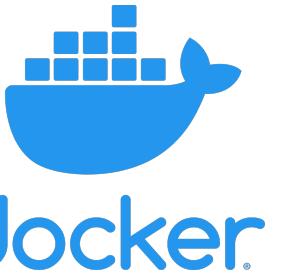
# EC2 Spot Instances



<https://console.aws.amazon.com/ec2sp/v1/spot/home?region=us-east-1#>

# Spot Fleets

- Spot Fleets = set of Spot Instances + (optional) On-Demand Instances
- The Spot Fleet will try to meet the target capacity with price constraints
  - Define possible launch pools: instance type (m5.large), OS, Availability Zone
  - Can have multiple launch pools, so that the fleet can choose
  - Spot Fleet stops launching instances when reaching capacity or max cost
- Strategies to allocate Spot Instances:
  - **lowestPrice**: from the pool with the lowest price (cost optimization, short workload)
  - **diversified**: distributed across all pools (great for availability, long workloads)
  - **capacityOptimized**: pool with the optimal capacity for the number of instances
- Spot Fleets allow us to automatically request Spot Instances with the lowest price



# What is Docker?

- Docker is a software development platform to deploy apps
- Apps are packaged in **containers** that can be run on any OS
- Apps run the same, regardless of where they're run
  - Any machine (no compatibility issues, predictable behavior)
  - Less work
  - Easier to maintain and deploy
  - Works with any language, any OS, any technology
- Control how much memory / CPU is allocated to your container
- Scale containers up and down very quickly (seconds)
- More efficient than Virtual machines

# Docker Containers Management on AWS

- To manage containers, we need a container management platform
- Amazon Elastic Container Service (Amazon ECS)
  - Amazon's own container platform
- Amazon Elastic Kubernetes Service (Amazon EKS)
  - Amazon's managed Kubernetes (open source)
- AWS Fargate
  - Amazon's own Serverless container platform
  - Works with ECS and with EKS



Amazon ECS



Amazon EKS



AWS Fargate

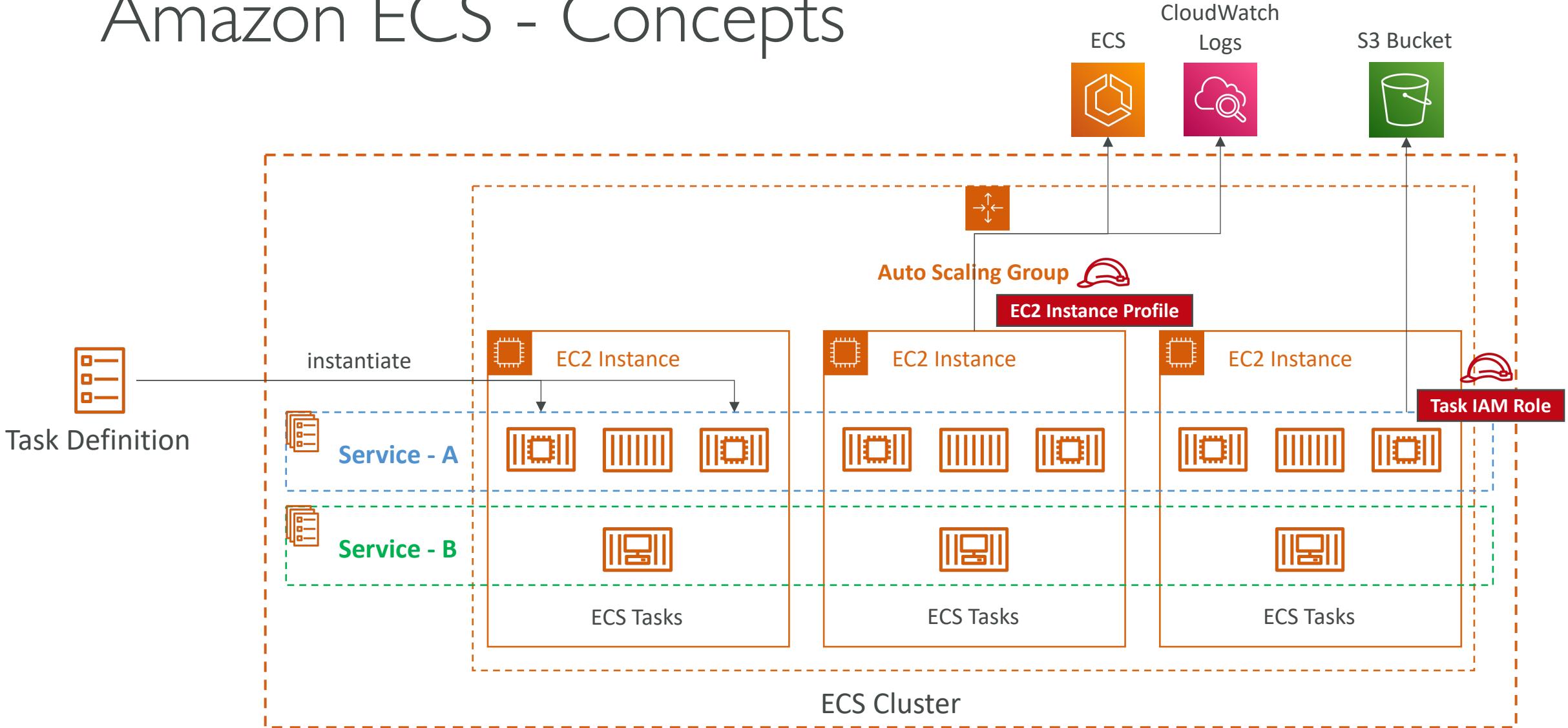
# Amazon ECS – Use cases

- Run Microservices
  - Run multiple Docker containers on the same machine
  - Easy Service Discovery features to enhance communication
  - Direct integration with Application Load Balancer and Network Load Balancer
  - Auto Scaling capability
- Run Batch Processing / Scheduled Tasks
  - Schedule ECS tasks to run on On-demand / Reserved / Spot instances
- Migrate Applications to the Cloud
  - Dockerize legacy applications running on-premises
  - Move Docker containers to run on Amazon ECS

# Amazon ECS – Concepts

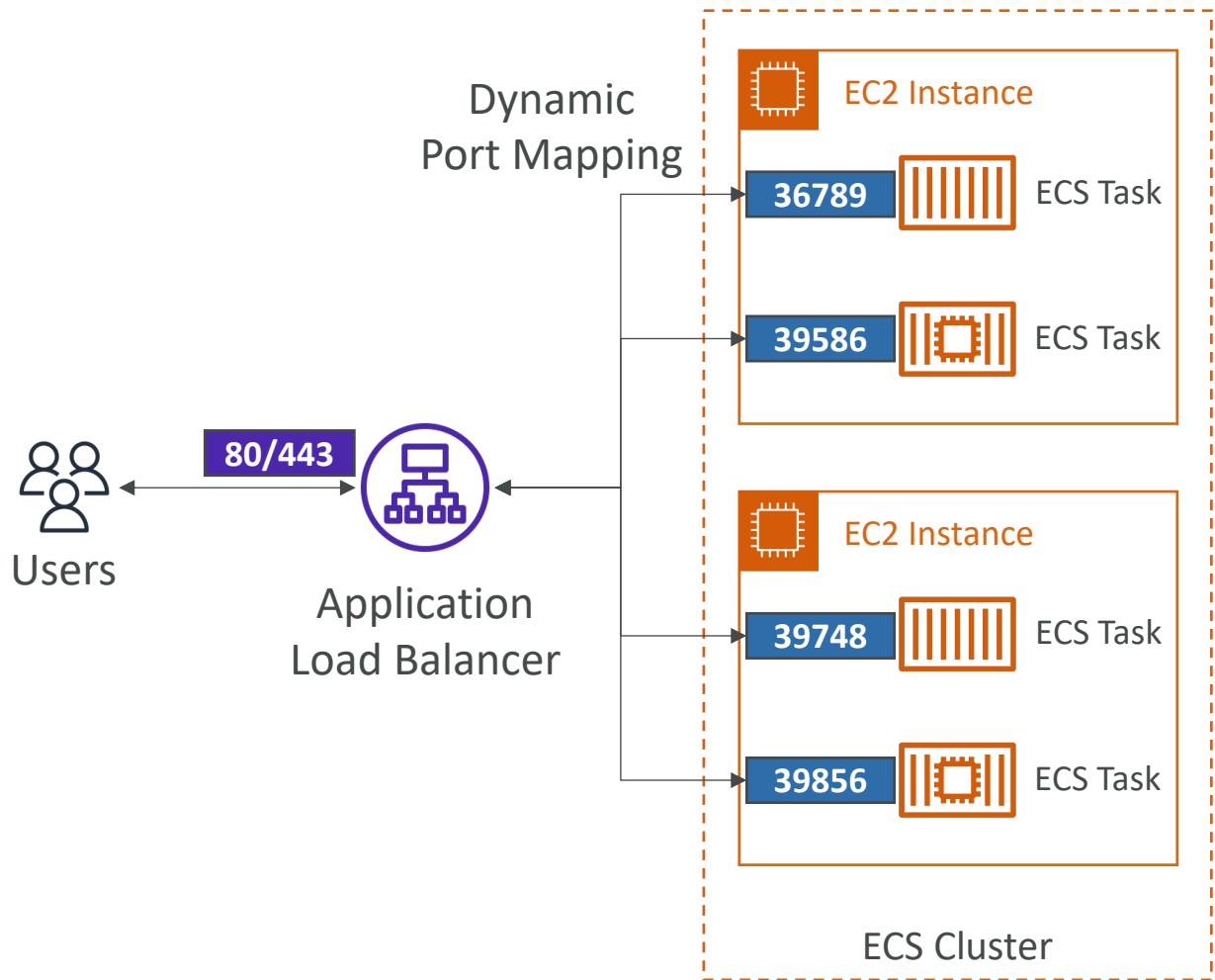
- **ECS Cluster** – logical grouping of EC2 instances
- **ECS Service** – defines how many tasks should run and how they should be run
- **Task Definitions** – metadata in **JSON form** to tell ECS how to run a Docker container (image name, CPU, RAM, ...)
- **ECS Task** – an instance of a Task Definition, a running Docker container(s)
- **ECS IAM Roles**
  - **EC2 Instance Profile** – used by the EC2 instance (e.g., make API calls to ECS, send logs, ...)
  - **ECS Task IAM Role** – allow each task to have a specific role (e.g., make API calls to S3, DynamoDB, ...)

# Amazon ECS - Concepts



# Amazon ECS – ALB Integration

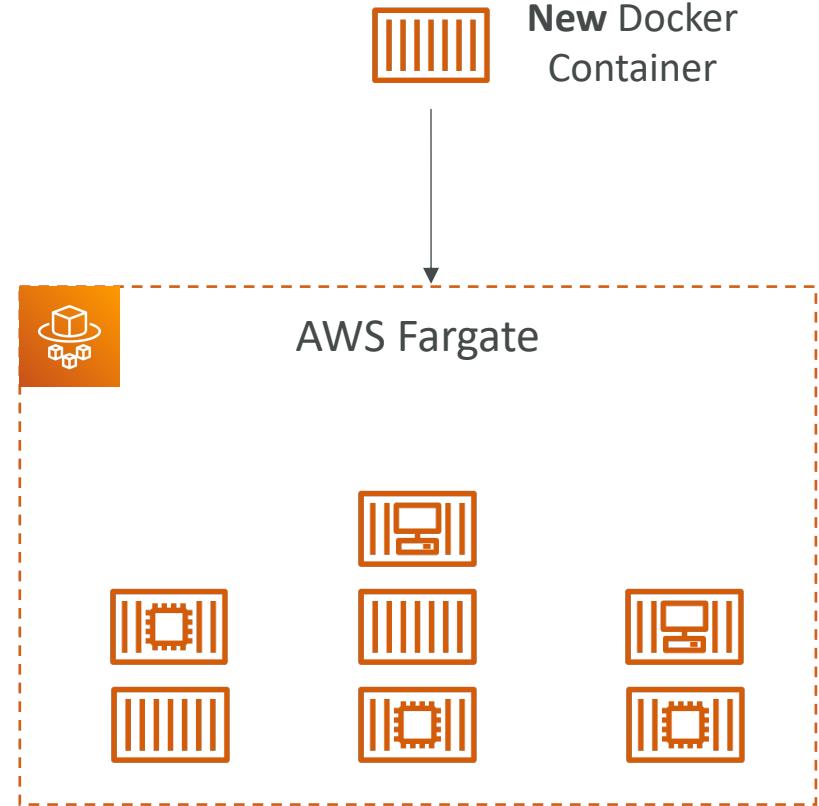
- We get Dynamic Port Mapping
- Allows you to run multiple instances of the same application on the same EC2 instance
- The ALB finds the right port on your EC2 Instances
- Use cases:
  - Increased resiliency even if running on one EC2 instance
  - Maximize utilization of CPU / cores
  - Ability to perform rolling upgrades without impacting app uptime



# AWS Fargate

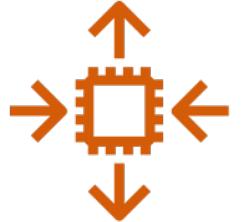


- Launch Docker containers on AWS
- You do not provision the infrastructure  
(no EC2 instances to manage)
- It's all serverless!
- You create task definitions
- AWS runs containers for you based on  
the CPU / RAM you need
- To scale, just increase the number of  
tasks. Simple! No more EC2 instances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



# Amazon ECS – Security & Networking

- You can inject secrets and configurations as Environment Variables into running Docker containers
  - Integration with **SSM Parameter Store** and **Secrets Manager**
- **ECS Tasks Networking**
  - **none** – no network connectivity, no port mappings
  - **bridge** – uses Docker's virtual container-based network
  - **host** – bypass Docker's network, uses the underlying host network interface
  - **awsvpc**
    - Every tasks launched on the instance gets its own ENI and a private IP address
    - Simplified networking, enhanced security, Security Groups, monitoring, VPC Flow Logs
    - Default mode for Fargate tasks



# Amazon ECS – Service Auto Scaling

- Automatically increase/decrease the desired number of tasks
- Amazon ECS leverages AWS Application Auto Scaling
- CPU and RAM is tracked in CloudWatch at the ECS Service level
- Target Tracking – scale based on target value for a specific CloudWatch metric
- Step Scaling – scale based on a specified CloudWatch Alarm
- Scheduled Scaling – scale based on a specified date/time (predictable changes)
- ECS Service Auto Scaling (task level) **≠** EC2 Auto Scaling (EC2 instance level)
- Fargate Auto Scaling is much easier to setup (because Serverless)



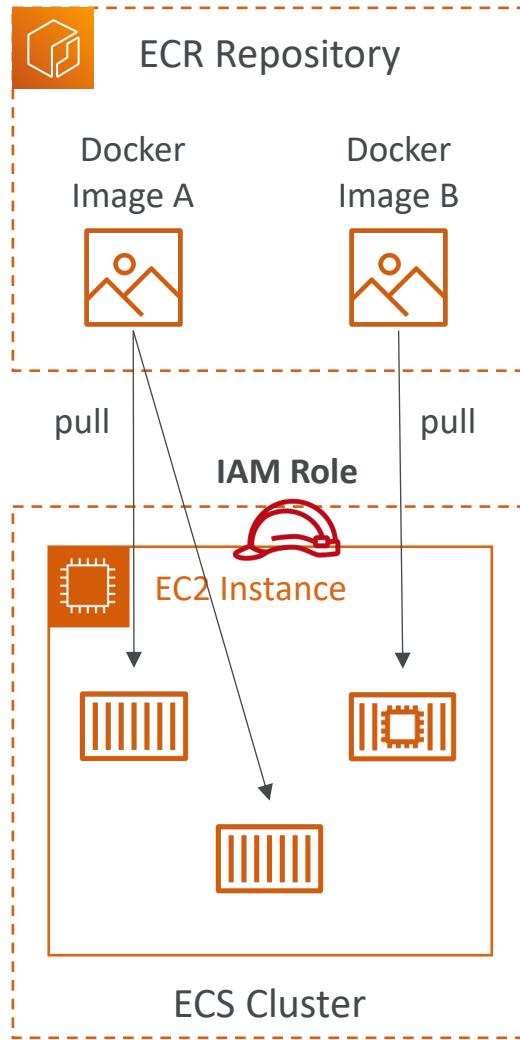
# Amazon ECS – Spot Instances

- **ECS Classic (EC2 Launch Type)**
  - Can have the underlying EC2 instances as Spot Instances (managed by an ASG)
  - Instances may go into draining mode to remove running tasks
  - Good for cost savings, but will impact reliability
- **AWS Fargate**
  - Specify minimum of tasks for on-demand baseline workload
  - Add tasks running on FARGATE\_SPOT for cost-savings (can be reclaimed by AWS)
  - Regardless of On-demand or Spot, Fargate scales well based on load

# Amazon ECR - Elastic Container Registry



- Store and manage Docker images on AWS
- Private and Public repository (Amazon ECR Public Gallery <https://gallery.ecr.aws>)
- Fully integrated with ECS
- Access is controlled through IAM (permission errors => check policy)
- Supports image vulnerability scanning, versioning, image tags, image lifecycle, ...



# AWS Lambda Integrations

## Main ones



API Gateway



Kinesis



DynamoDB



AWS S3 –  
Simple Storage Service



AWS IoT  
Internet of Things



CloudWatch Events



CloudWatch Logs



AWS SNS

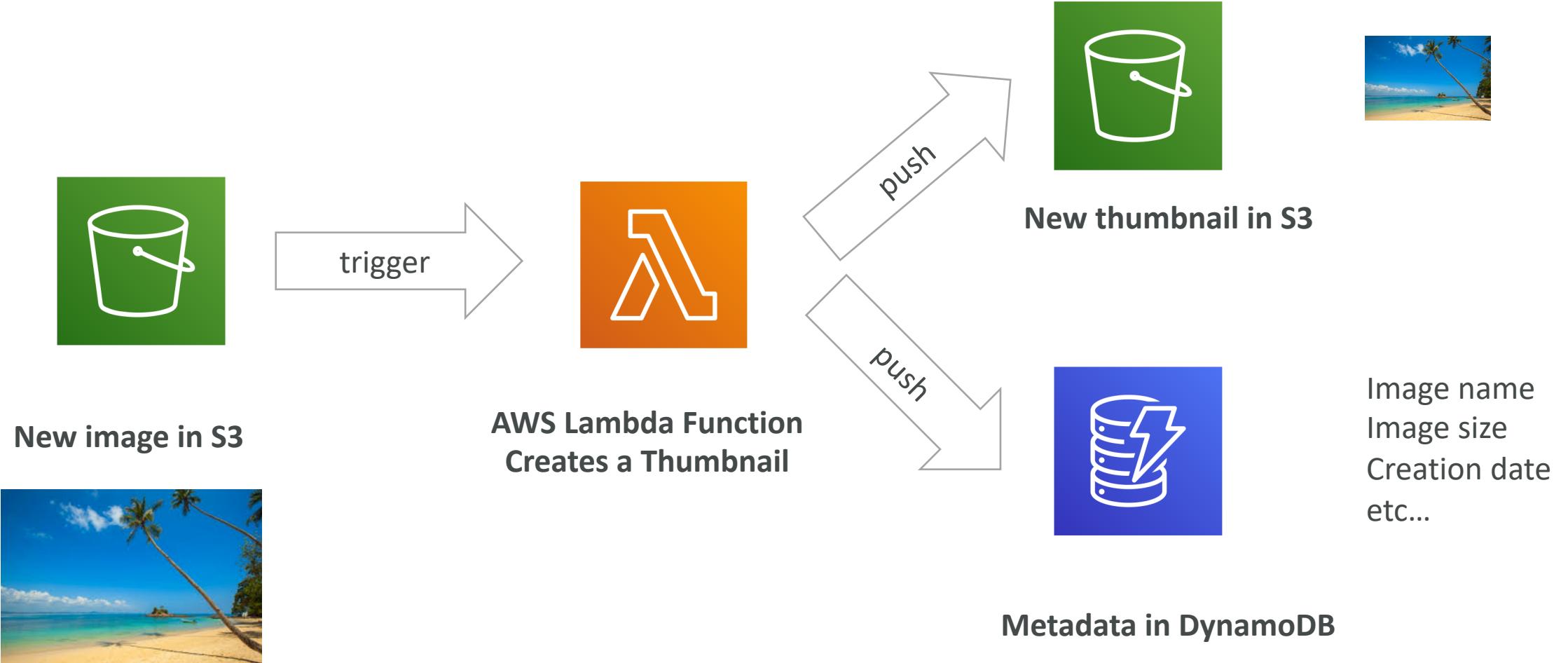


AWS Cognito

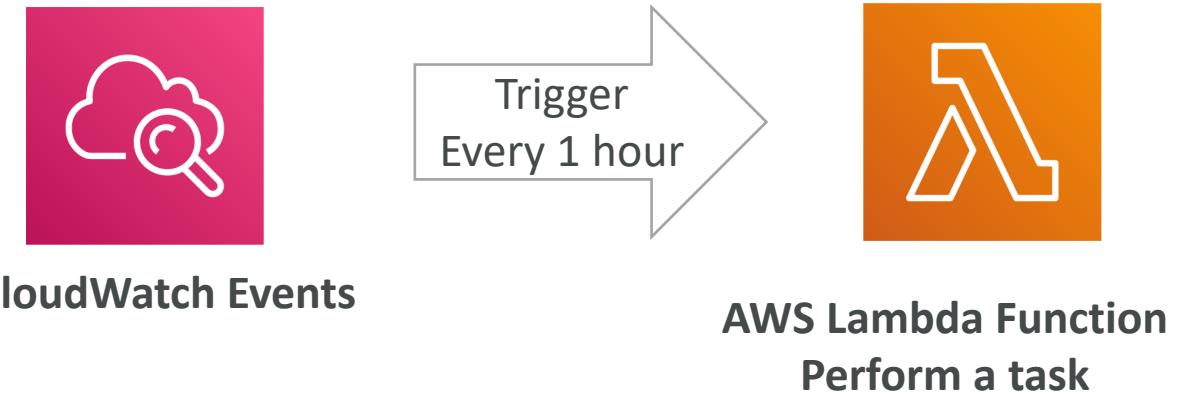


Amazon  
SQS

# Example: Serverless Thumbnail creation



# Example: Serverless CRON Job



# AWS Lambda Language Support (runtimes)

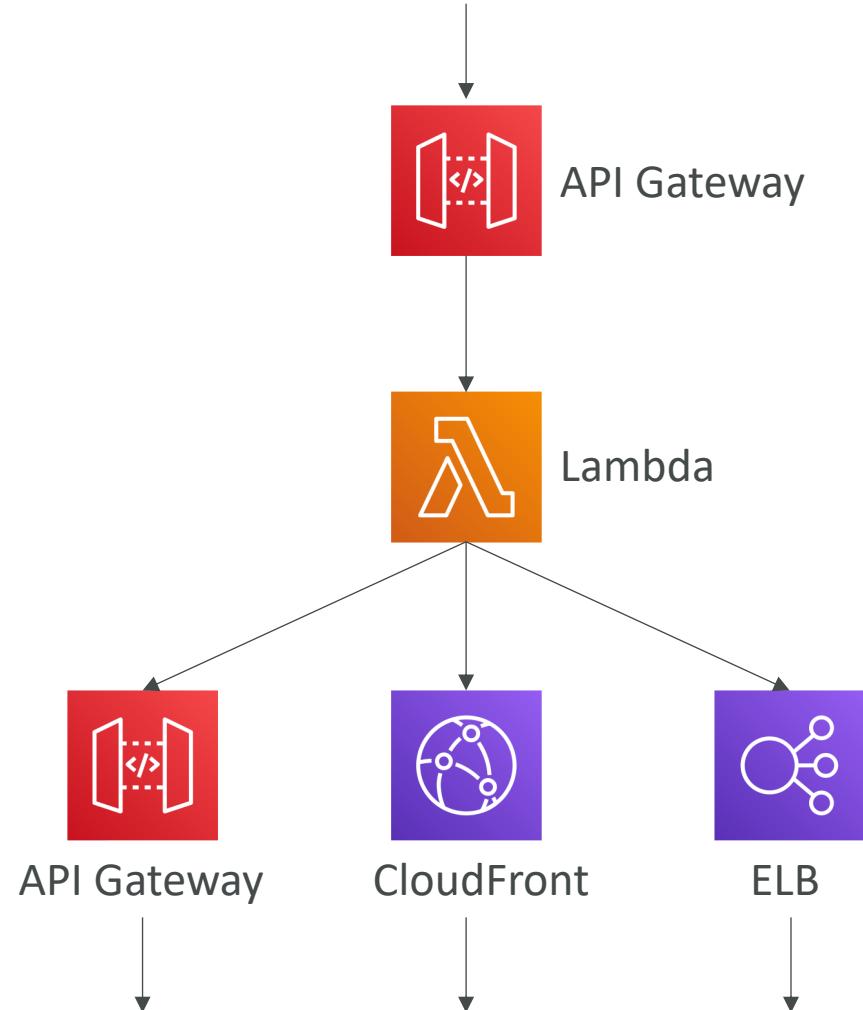
- Node.js (JavaScript)
- Python
- Java
- C# (.NET Core)
- Golang
- C# / Powershell
- Ruby
- Custom Runtime API (community supported, example Rust)
- Lambda Container Image
  - The container image must implement the Lambda Runtime API
  - ECS / Fargate is preferred for running arbitrary Docker images

# Lambda – Limits to know

- RAM – 128 MB to 10,240 MB (10 GB)
- CPU – is linked to RAM (cannot be set manually)
  - 2 vCPUs are allocated at 1,769 MB of RAM
  - 6 vCPUs are allocated at 10,240 MB of RAM
- Timeout – up to 15 minutes
- /tmp Storage – 512 MB (can't process BIG files)
- Deployment Package – 50 MB (zipped) , 250 MB (unzipped) including layers
- Concurrent Executions – 1000 (soft limit that can be increased)
- Container Image Size – 10 GB
- Invocation Payload (request/response) – 6 MB (sync), 256 KB (async)

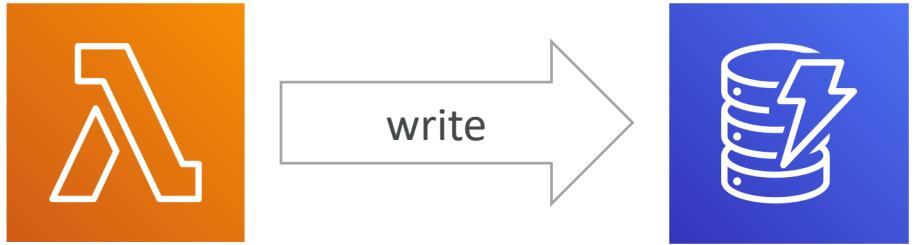
# Lambda – Latencies Considerations (approximates)

- Lambda Latency:
  - Cold Lambda Invocation: ~100ms
  - Warm Lambda Invocation: ~ms
  - New feature of “provisioned concurrency” (Dec 2019) to reduce # of cold starts
- API Gateway invocation: 100 ms
- CloudFront invocation: 100 ms
- If you chain with other services (API Gateway, CloudFront, ALB, Lambda, SQS, Step Functions...), add their latencies as well
- X-Ray can help visualize the end-to-end latency



# Lambda - Security

- IAM Roles for Lambda to grant access to other AWS services
- Resource-based Policies for Lambda (similar to S3 bucket policies):
  - Allow other accounts to invoke or manage Lambda
  - Allow other services to invoke or manage Lambda

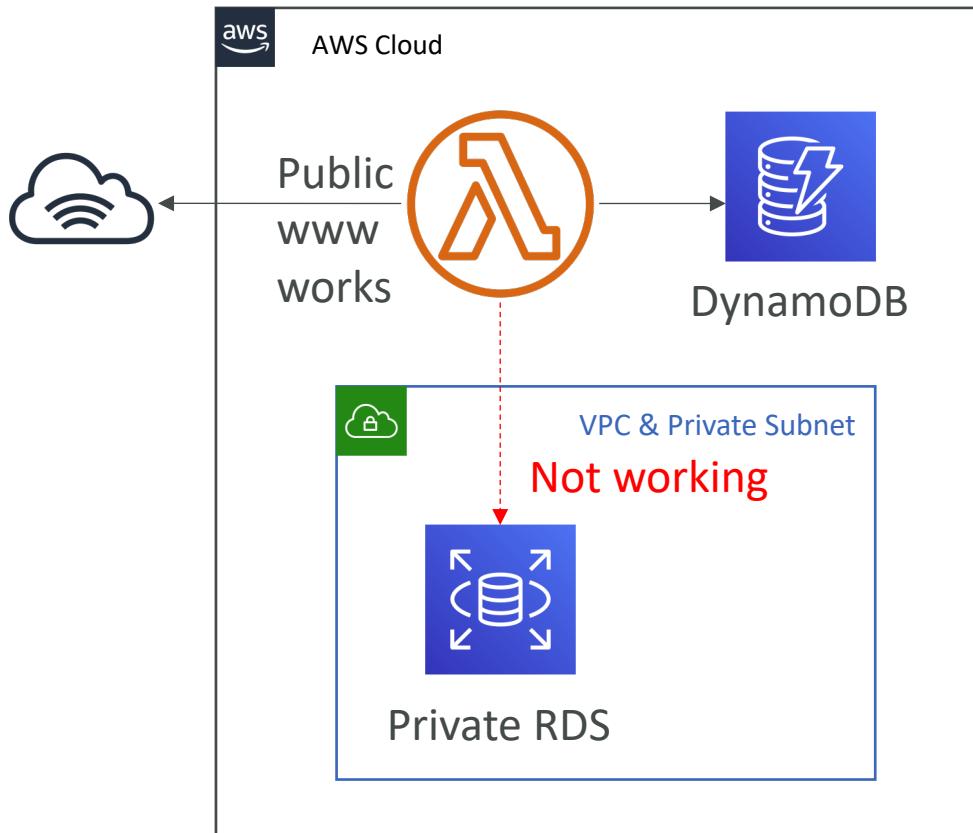


```
{  
  "Sid": "sns",  
  "Effect": "Allow",  
  "Principal": {  
    "Service": "sns.amazonaws.com"  
  },  
  "Action": "lambda:InvokeFunction",  
  "Resource":  
    "arn:aws:lambda:us-east-2:123456789012:function:my-function"  
}
```

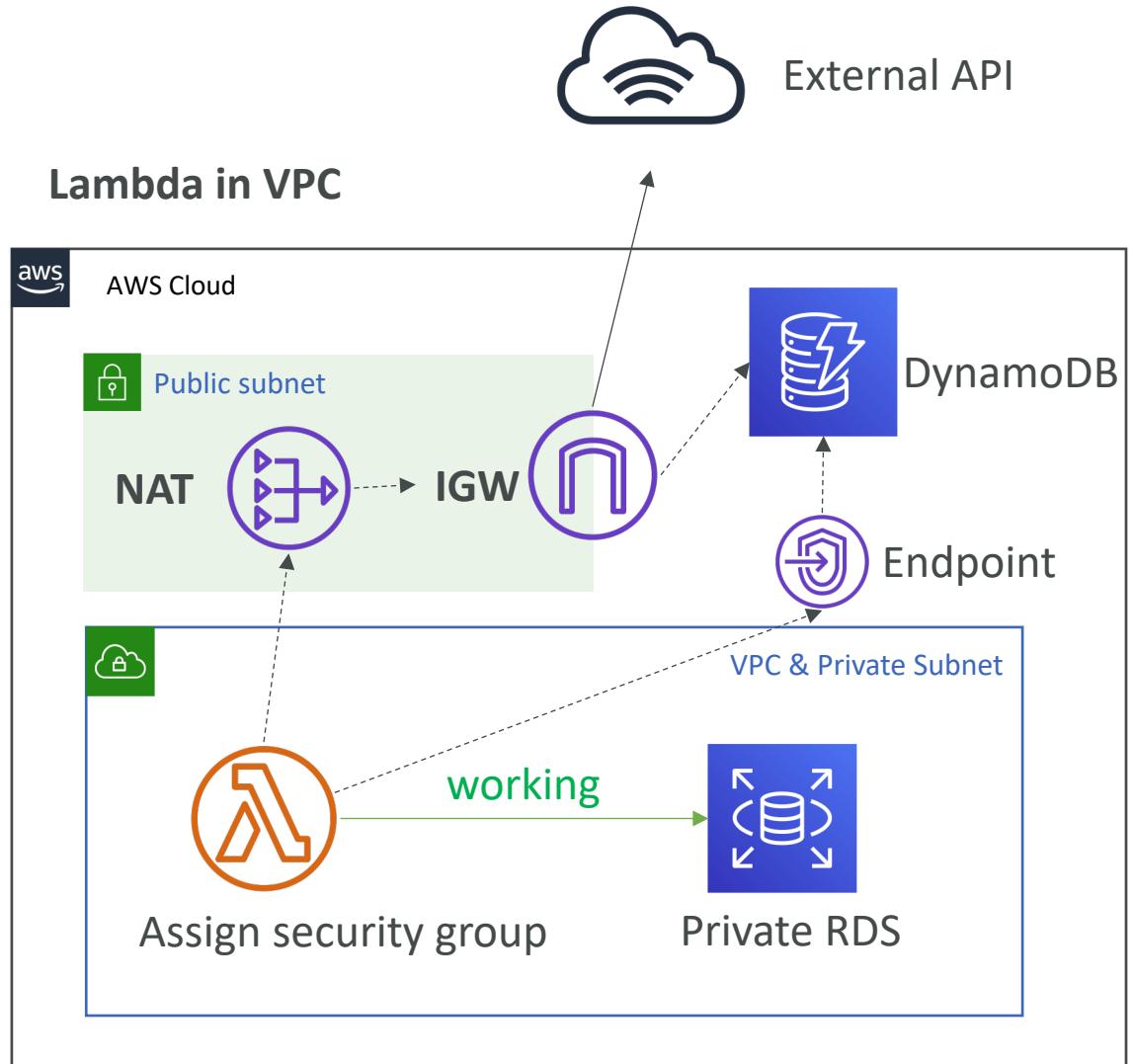
(define through the CLI)

# Lambda in a VPC

## Default Lambda Deployment



## Lambda in VPC



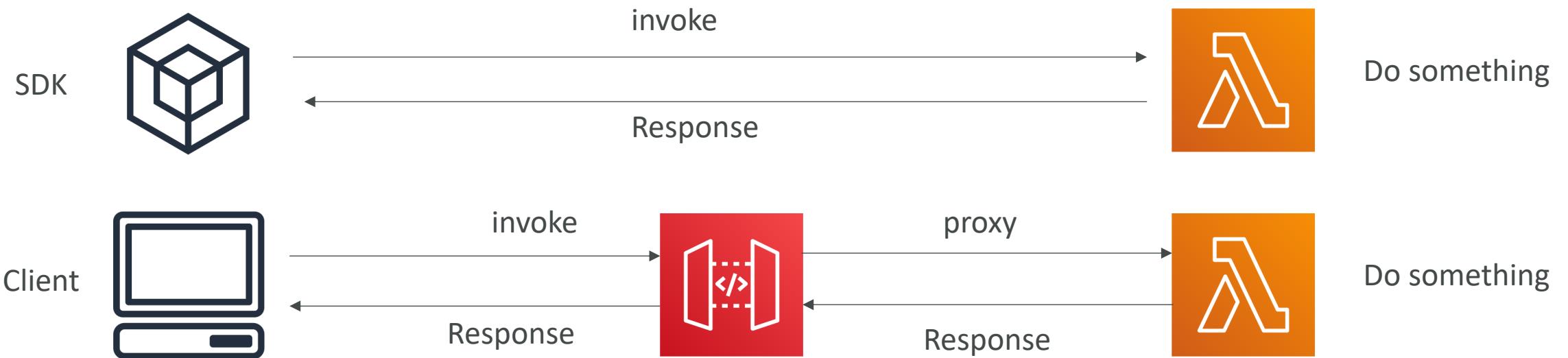
Note: Lambda - CloudWatch Logs works even without endpoint or NAT Gateway

# AWS Lambda Logging, Monitoring and Tracing

- CloudWatch:
  - AWS Lambda execution logs are stored in AWS CloudWatch Logs
  - AWS Lambda metrics are displayed in AWS CloudWatch Metrics (successful invocations, error rates, latency, timeouts, etc...)
  - Make sure your AWS Lambda function has an execution role with an IAM policy that authorizes writes to CloudWatch Logs
- X-Ray:
  - It's possible to trace Lambda with X-Ray
  - Enable in Lambda configuration (runs the X-Ray daemon for you)
  - Use AWS SDK in Code
  - Ensure Lambda Function has correct IAM Execution Role

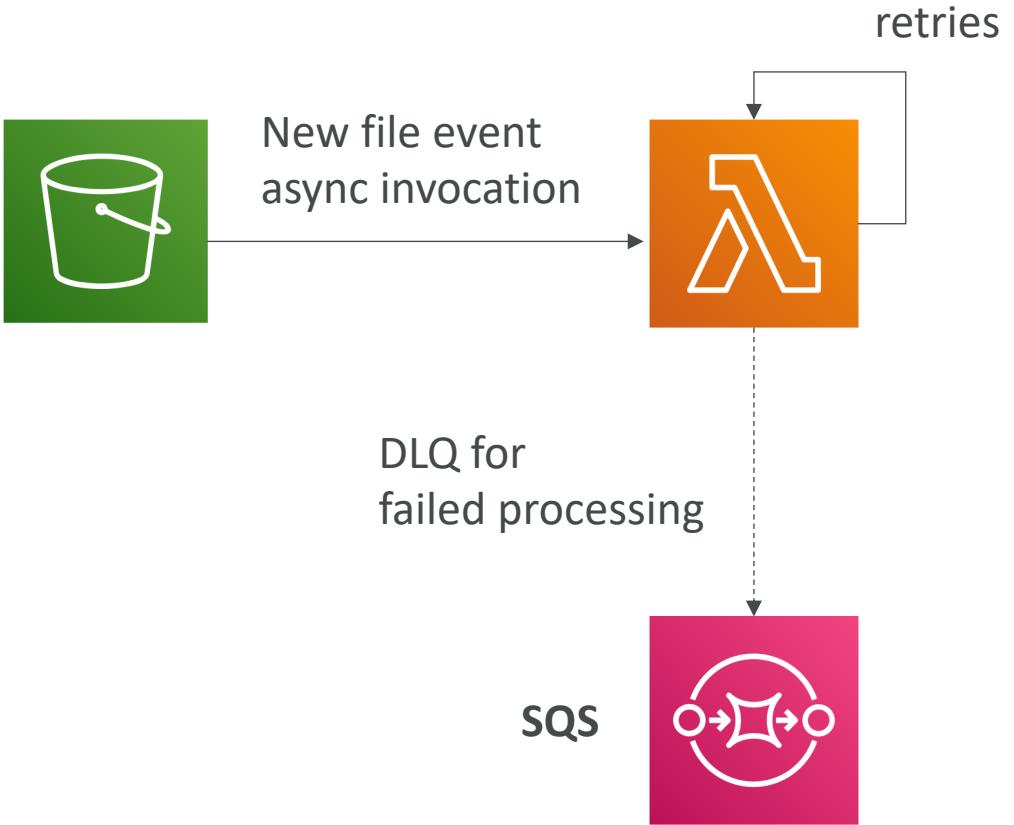
# Lambda – Synchronous Invocations

- Synchronous: CLI, SDK, API Gateway
  - Results is returned right away
  - Error handling must happen client side (retries, exponential backoff, etc...)



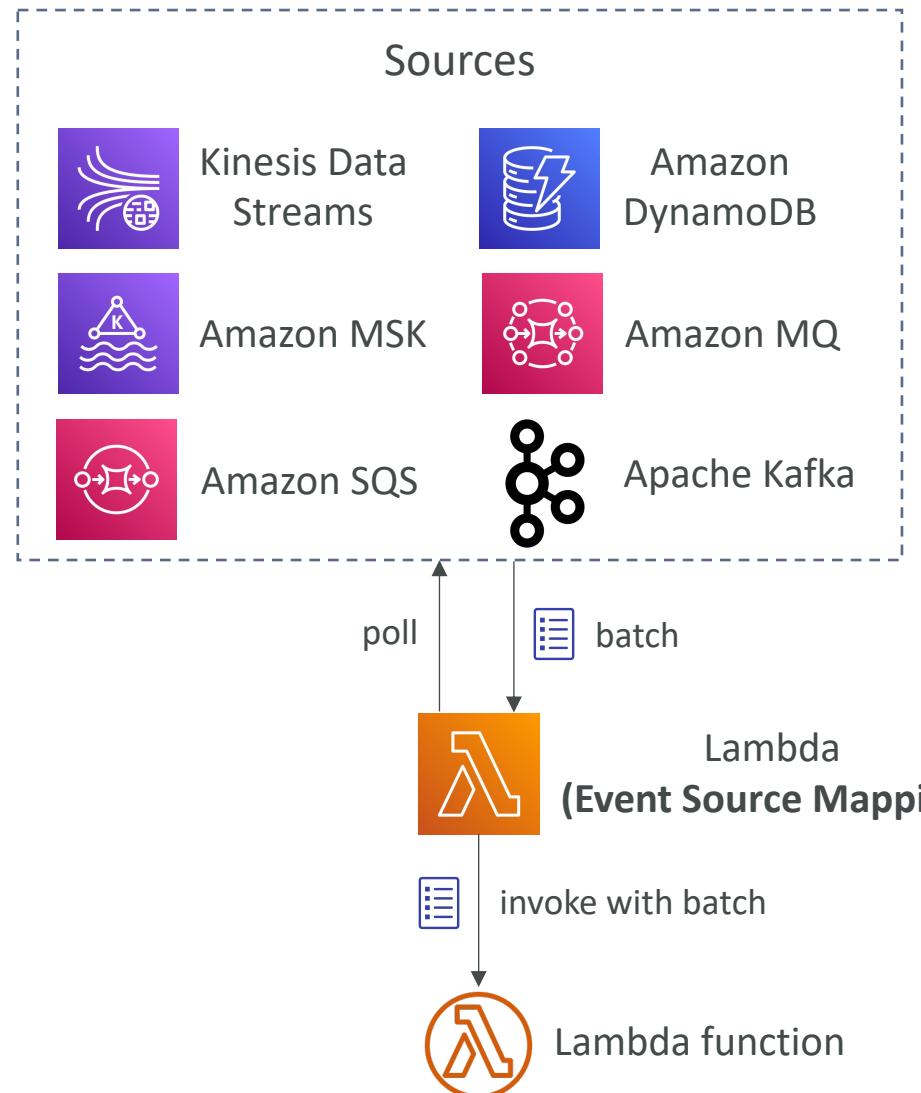
# Lambda – Asynchronous Invocation

- S3, SNS, CloudWatch Events...
- Lambda attempts to retry on errors (3 tries total)
- Make sure the processing is **idempotent** (in case of retries)
- Can define a DLQ (dead-letter queue) – SNS or SQS – for failed processing



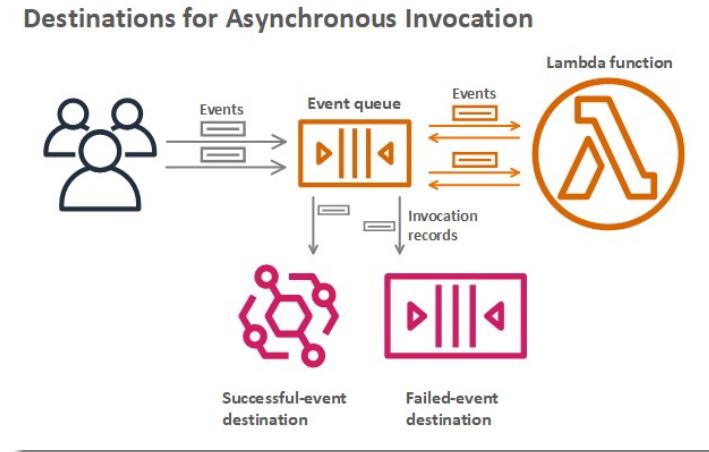
# Lambda – Event Source Mapping

- Sources:
  - Kinesis Data Streams, SQS, SQS FIFO
  - DynamoDB Streams, Amazon MQ, Apache Kafka
- Records need to be polled from the source (common denominator)
- All records are respect ordering properties EXCEPT for SQS standard
- If your function returns an error, **the entire batch is reprocessed** until success
  - Kinesis, DynamoDB Stream: stop shard processing
  - SQS FIFO: stop, unless a SQS DLQ has been defined
  - Need to make sure your Lambda function is idempotent

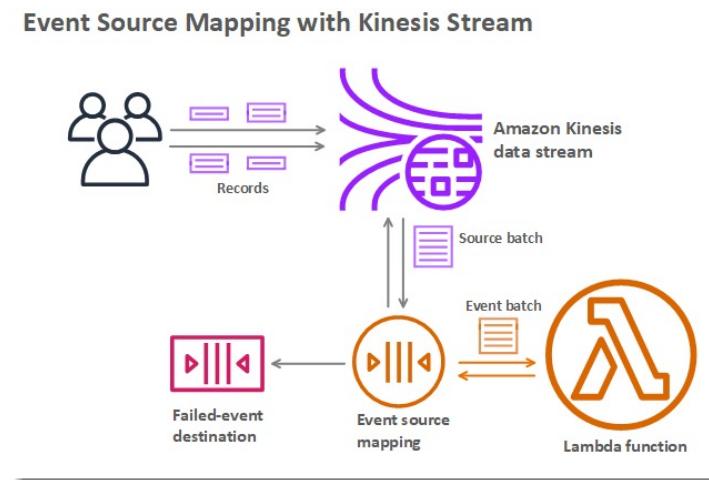


# Lambda – Destinations

- Nov 2019: Can configure to send result to a destination
- **Asynchronous invocations** - can define destinations for successful and failed event:
  - Amazon SQS
  - Amazon SNS
  - AWS Lambda
  - Amazon EventBridge bus
- Note: AWS recommends you use destinations instead of DLQ now (but both can be used at the same time)
- **Event Source mapping:** for discarded event batches
  - Amazon SQS
  - Amazon SNS
- Note: you can send events to a DLQ directly from SQS



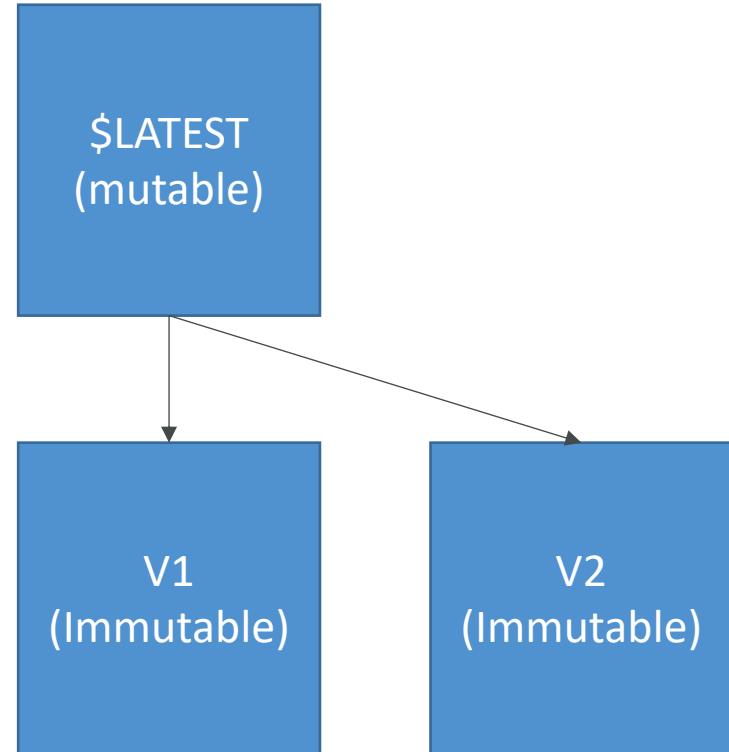
<https://docs.aws.amazon.com/lambda/latest/dg/invocation-async.html>



<https://docs.aws.amazon.com/lambda/latest/dg/invocation-eventsourcemapping.html>

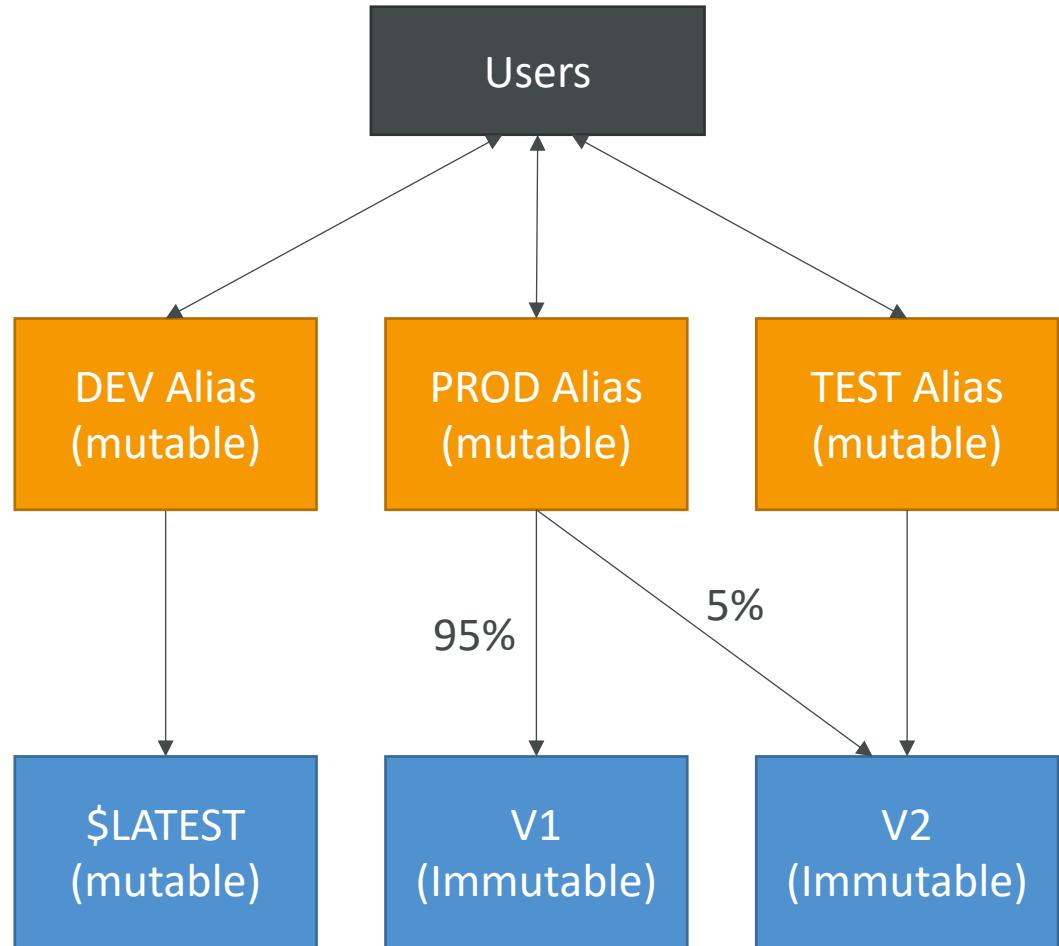
# AWS Lambda Versions

- When you work on a Lambda function, we work on **\$LATEST**
- When we're ready to publish a Lambda function, we create a version
- Versions are immutable
- Versions have increasing version numbers
- Versions get their own ARN (Amazon Resource Name)
- Version = code + configuration (nothing can be changed - immutable)
- Each version of the lambda function can be accessed

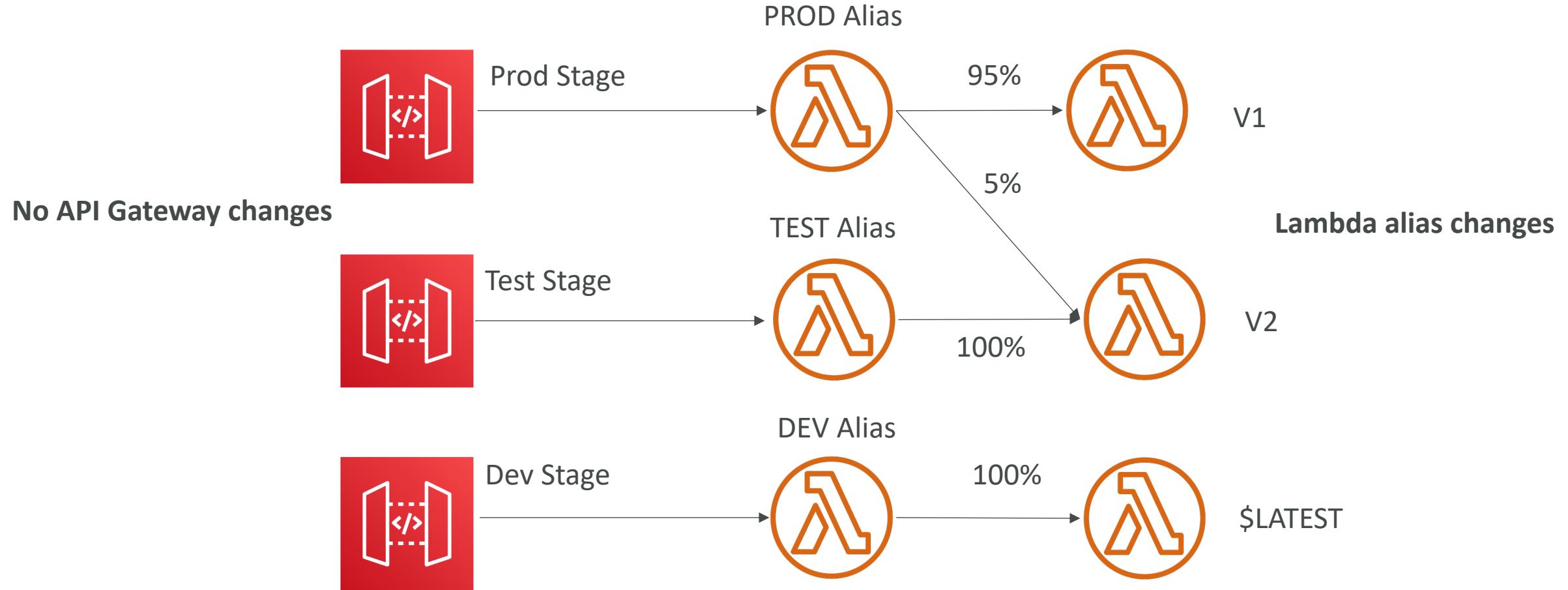


# AWS Lambda Aliases

- Aliases are "pointers" to Lambda function versions
- We can define a "dev", "test", "prod" aliases and have them point at different lambda versions
- Aliases are mutable
- Aliases enable Blue / Green deployment by assigning weights to lambda functions
- Aliases enable stable configuration of our event triggers / destinations
- Aliases have their own ARNs
- Aliases cannot reference aliases

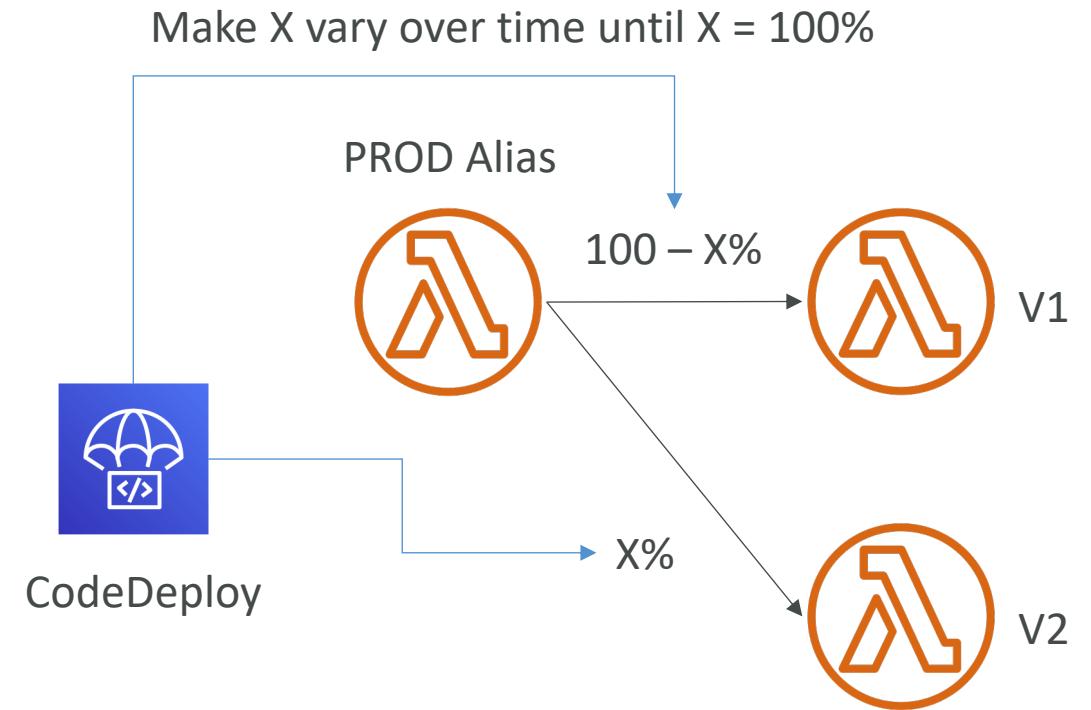


# AWS Lambda Aliases with API Gateway



# Lambda & CodeDeploy

- CodeDeploy can help you automate traffic shift for Lambda aliases
- Feature is integrated within the SAM framework
- **Linear:** grow traffic every N minutes until 100%
  - Linear10PercentEvery3Minutes
  - Linear10PercentEvery10Minutes
- **Canary:** try X percent then 100%
  - Canary10Percent5Minutes
  - Canary10Percent30Minutes
- **AllAtOnce:** immediate
- Can create Pre & Post Traffic hooks to check the health of the Lambda function

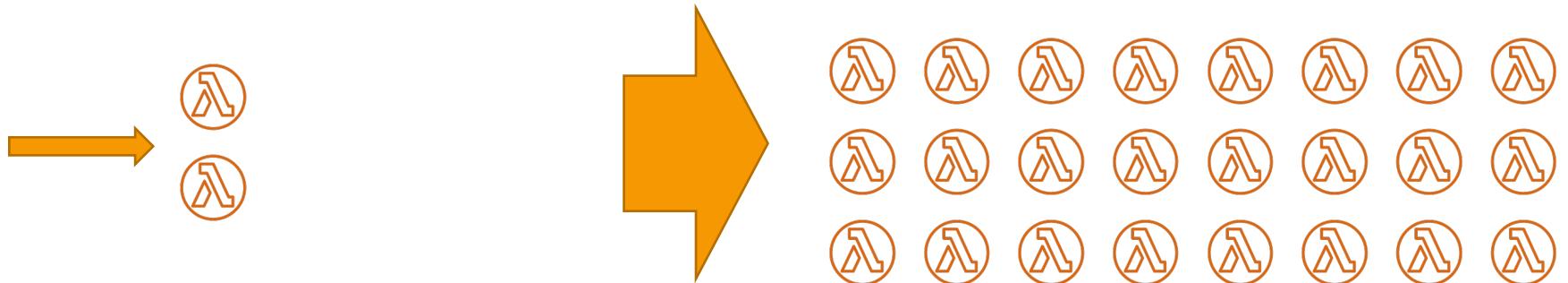


# Lambda Environment Variables

- Environment variable = key / value pair in “String” form
  - Adjust the function behavior without updating code
  - The environment variables are available to your code
  - Lambda Service adds its own system environment variables as well
- 
- Helpful to store secrets (encrypted by KMS)
  - Secrets can be encrypted by the Lambda service key, or your own KMS Key

# Lambda Concurrency and Throttling

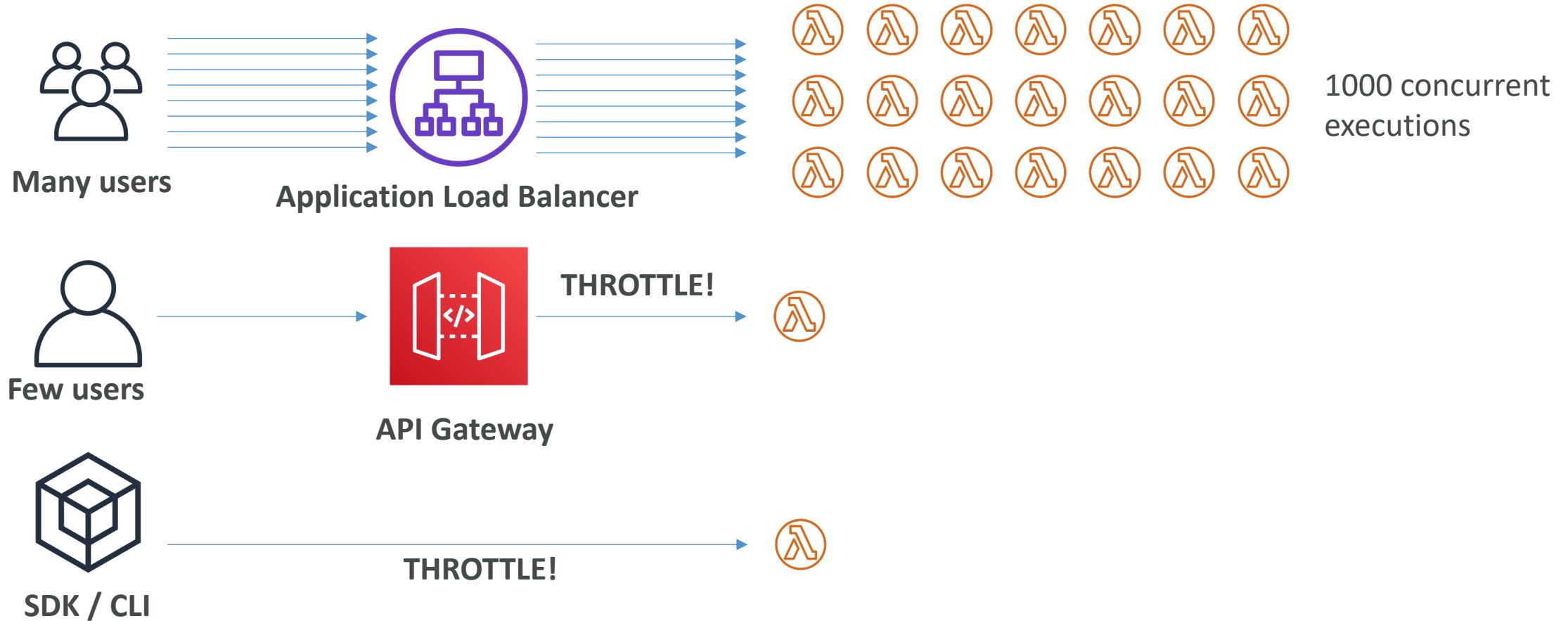
- Concurrency limit: up to 1000 concurrent executions



- Can set a “reserved concurrency” at the function level (=limit)
- Each invocation over the concurrency limit will trigger a “Throttle”

# Lambda Concurrency Issue

- If you don't reserve (=limit) concurrency, the following can happen:

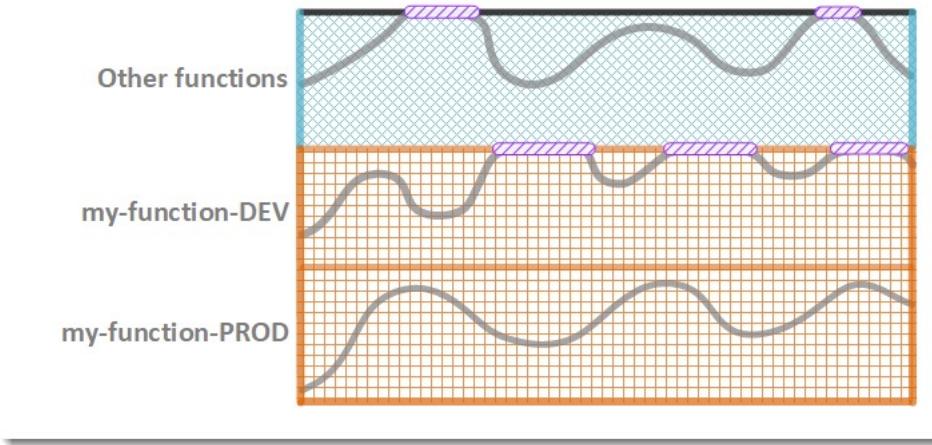


# Cold Starts & Provisioned Concurrency

- **Cold Start:**
  - New instance => code is loaded and code outside the handler run (init)
  - If the init is large (code, dependencies, SDK...) this process can take some time.
  - First request served by new instances has higher latency than the rest
- **Provisioned Concurrency:**
  - Concurrency is allocated before the function is invoked (in advance)
  - So, the cold start never happens, and all invocations have low latency
  - Application Auto Scaling can manage concurrency (schedule or target utilization)

# Reserved and Provisioned Concurrency

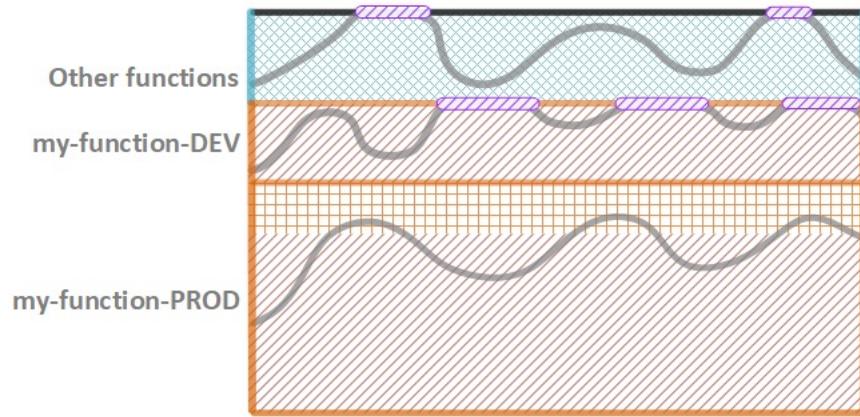
Reserved Concurrency



Legend

- Function concurrency
- Reserved concurrency
- Unreserved concurrency
- Throttling

Provisioned Concurrency with Reserved Concurrency



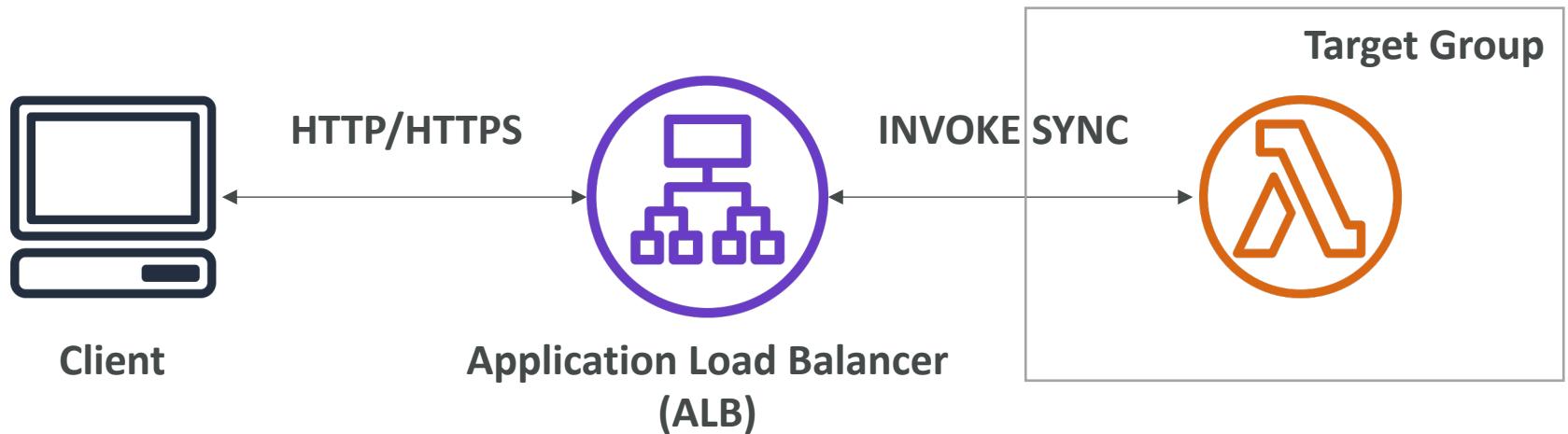
Legend

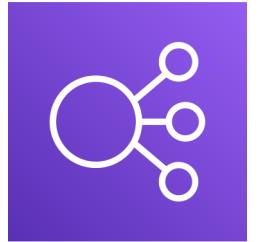
- Function concurrency
- Reserved concurrency
- Provisioned concurrency
- Unreserved concurrency
- Throttling

<https://docs.aws.amazon.com/lambda/latest/dg/configuration-concurrency.html>

# Lambda Integration with ALB

- To expose a Lambda function as an HTTP(S) endpoint...
- You can use the Application Load Balancer (or an API Gateway)
- The Lambda function must be registered in a target group



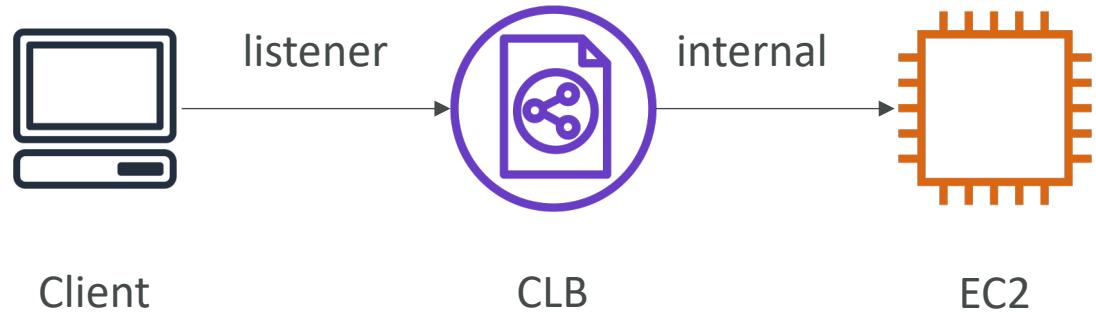


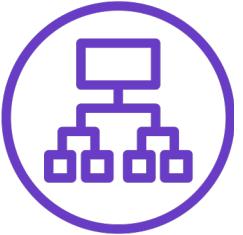
# Types of load balancer on AWS

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

# Classic Load Balancers (v1)

- Health Checks can be HTTP (L7) or TCP (L4) based including with SSL
- Supports only one SSL certificate
  - The SSL certificate can have many SAN (Subject Alternate Name), but the SSL certificate must be changed anytime a SAN is added / edited / removed
  - Better to use ALB with SNI (Server Name Indication) if possible
  - Can use multiple CLB if you want distinct SSL certificates
- TCP => TCP passes all the traffic to the EC2 instance
  - Only way to use 2-way SSL authentication



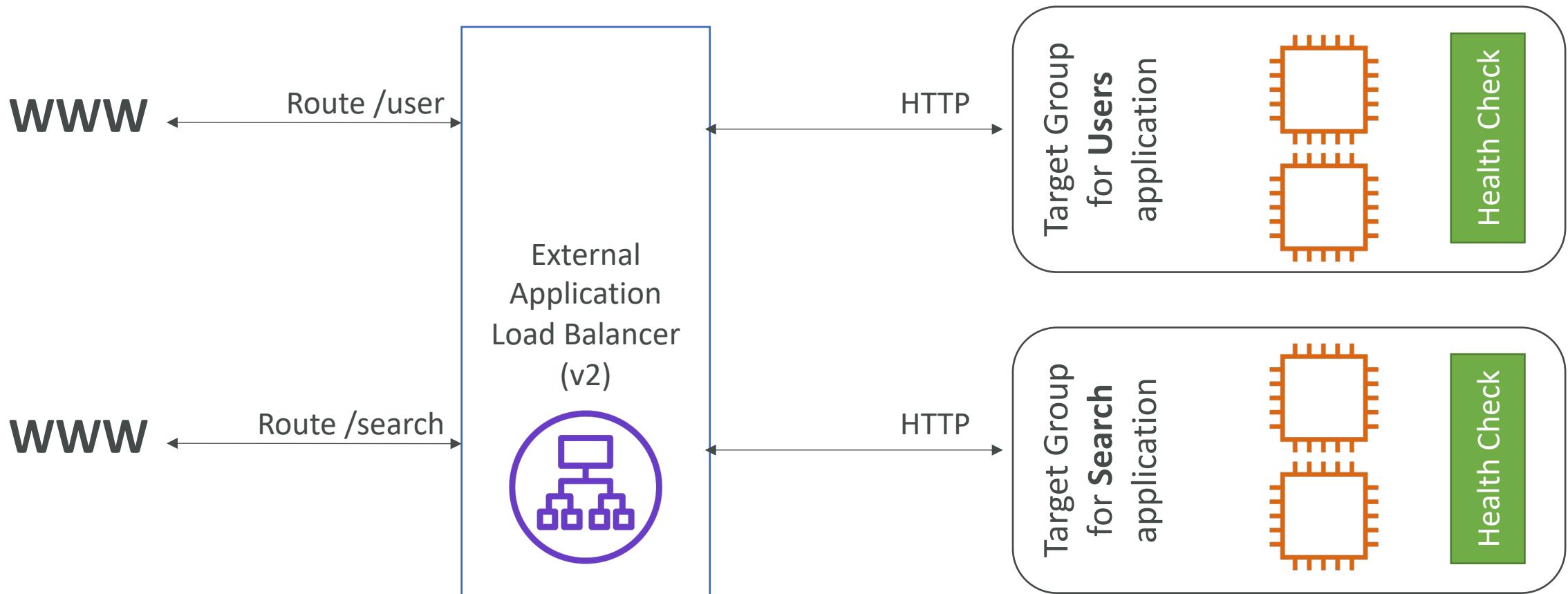


# Application Load Balancer (v2)

- Application load balancers is Layer 7 (HTTP)
- Load balancing to multiple HTTP applications across machines (target groups)
- Load balancing to multiple applications on the same machine (ex: containers) – great fit with ECS, has dynamic port mapping
- Support for HTTP/2 and WebSocket
- Support redirects (from HTTP to HTTPS for example)
- Routing Rules for path, headers, query string

# Application Load Balancer (v2)

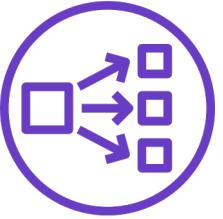
## HTTP Based Traffic



# Application Load Balancer (v2)

## Target Groups

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

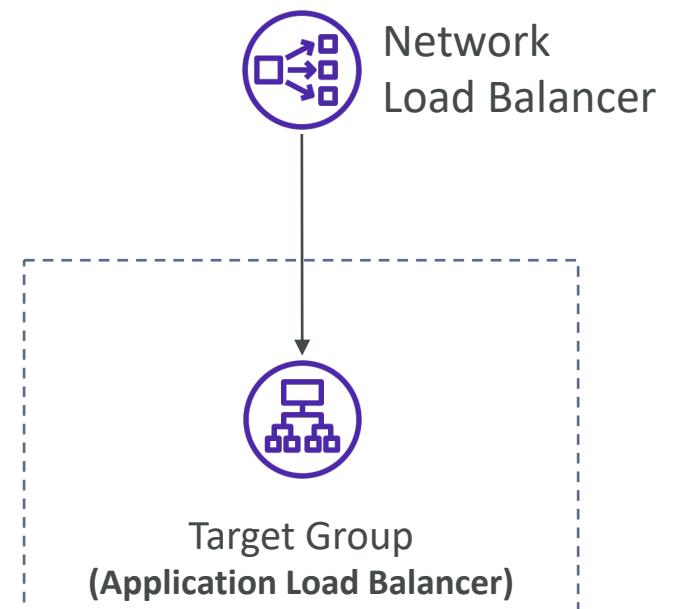
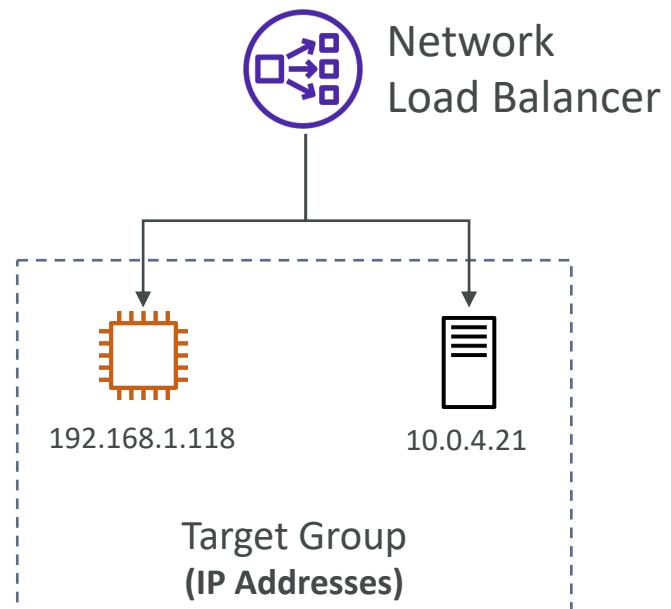
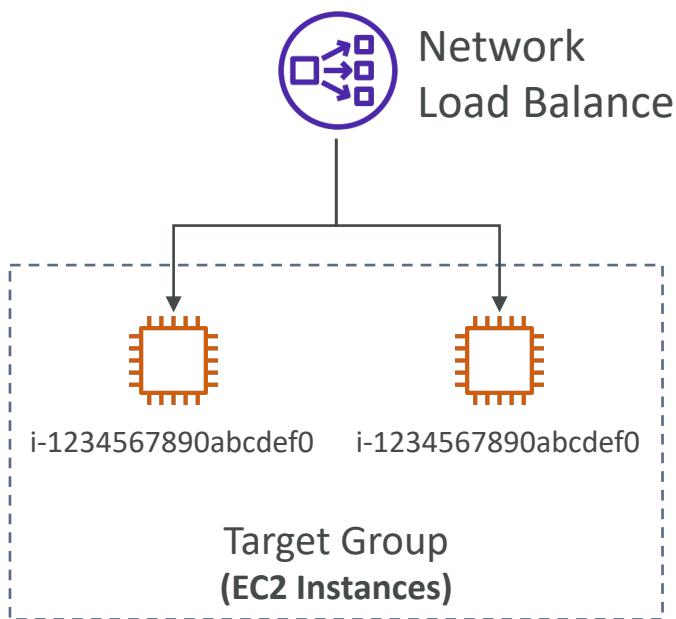


# Network Load Balancer (v2)

- Network load balancers (Layer 4) allow to:
  - Forward TCP & UDP traffic to your instances
  - Handle millions of requests per second
  - Less latency ~100 ms (vs 400 ms for ALB)
- NLB has one static IP per AZ, and supports assigning Elastic IP (helpful for whitelisting specific IP)
- NLB are used for extreme performance, TCP or UDP traffic
- Not included in the AWS free tier

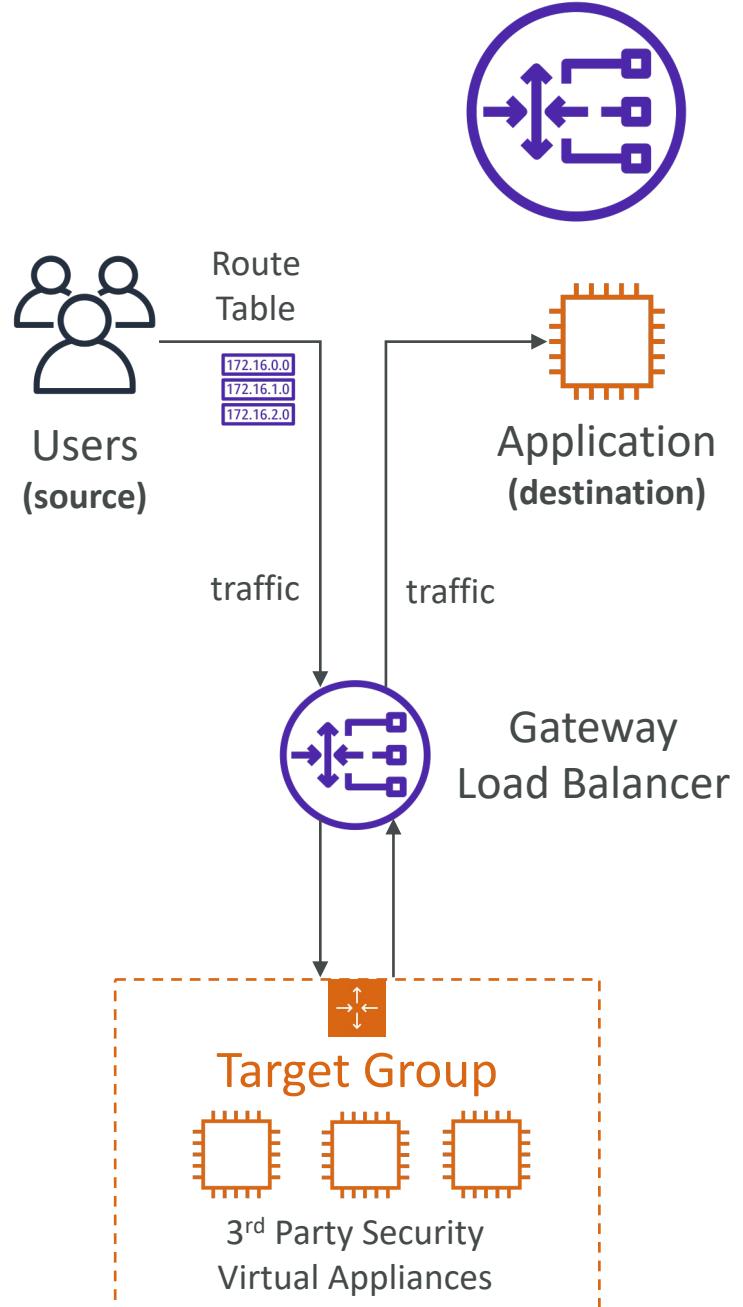
# Network Load Balancer – Target Groups

- EC2 instances
- IP Addresses – must be private IPs
- Application Load Balancer



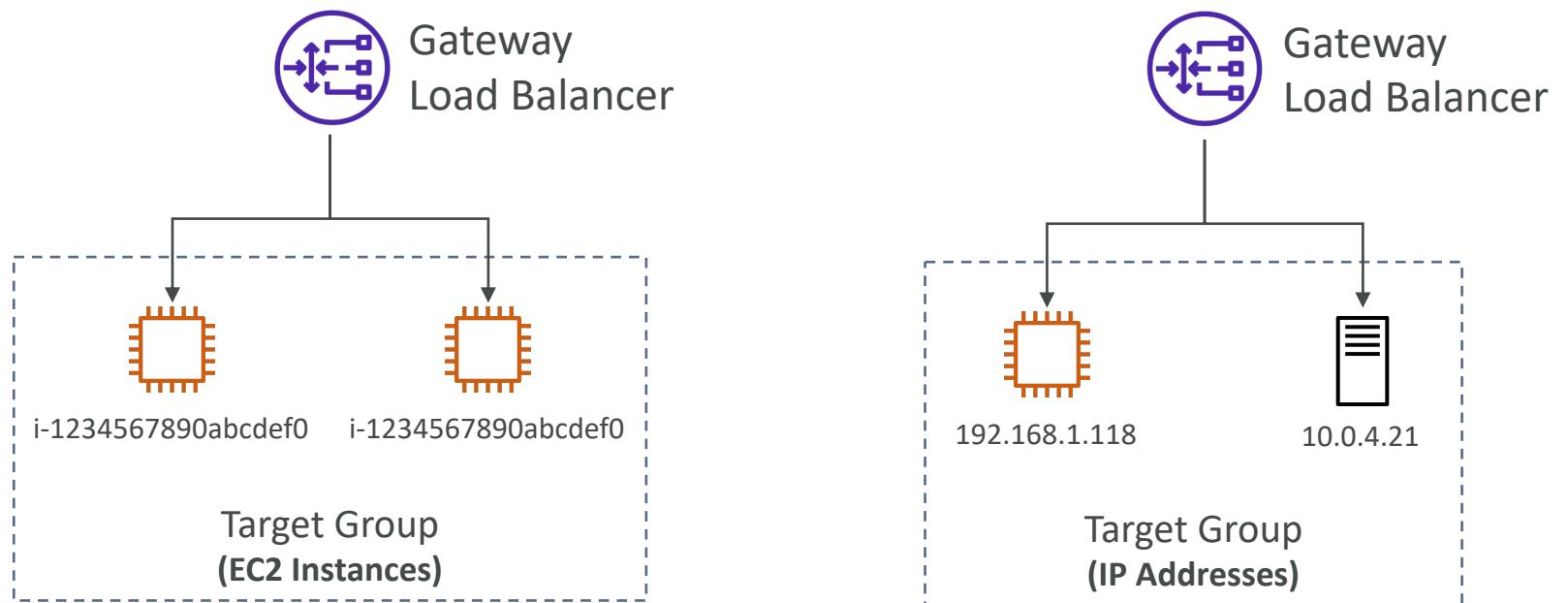
# Gateway Load Balancer

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



# Gateway Load Balancer – Target Groups

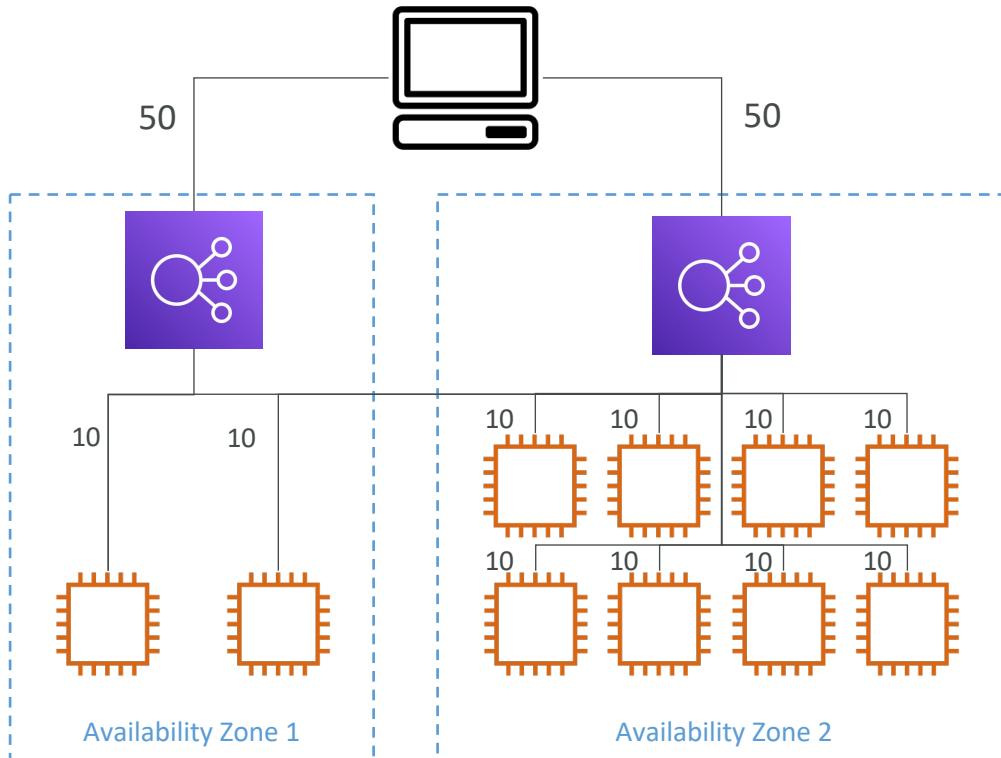
- EC2 instances
- IP Addresses – must be private IPs



# Cross-Zone Load Balancing

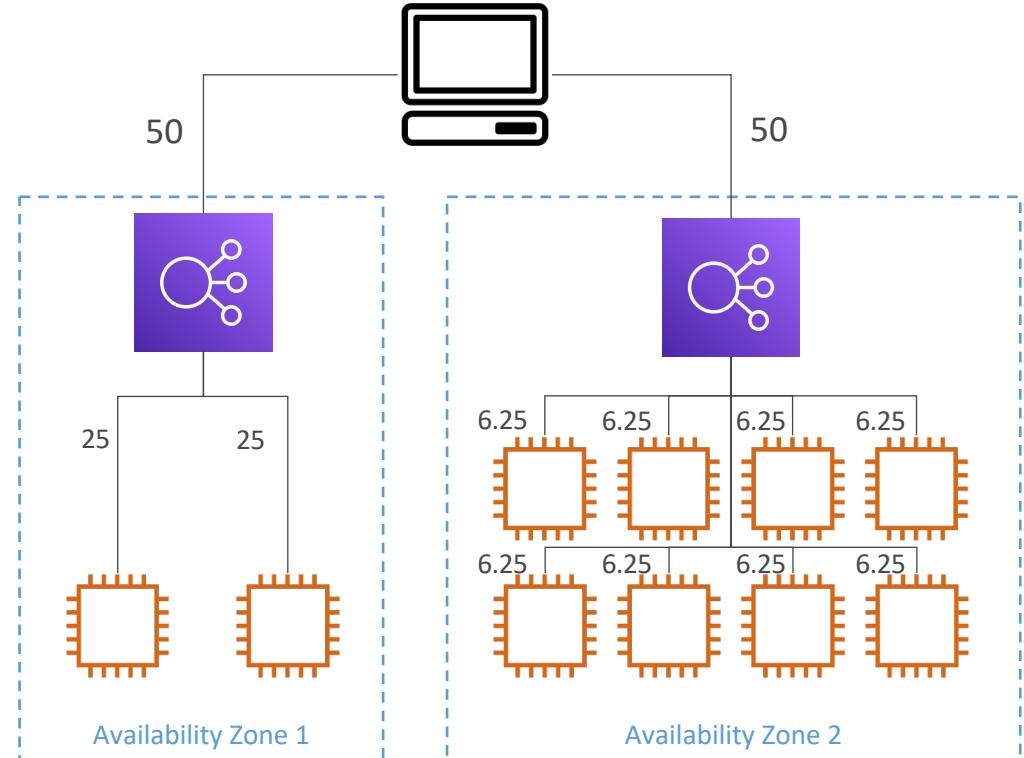
## With Cross Zone Load Balancing:

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



## Without Cross Zone Load Balancing:

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

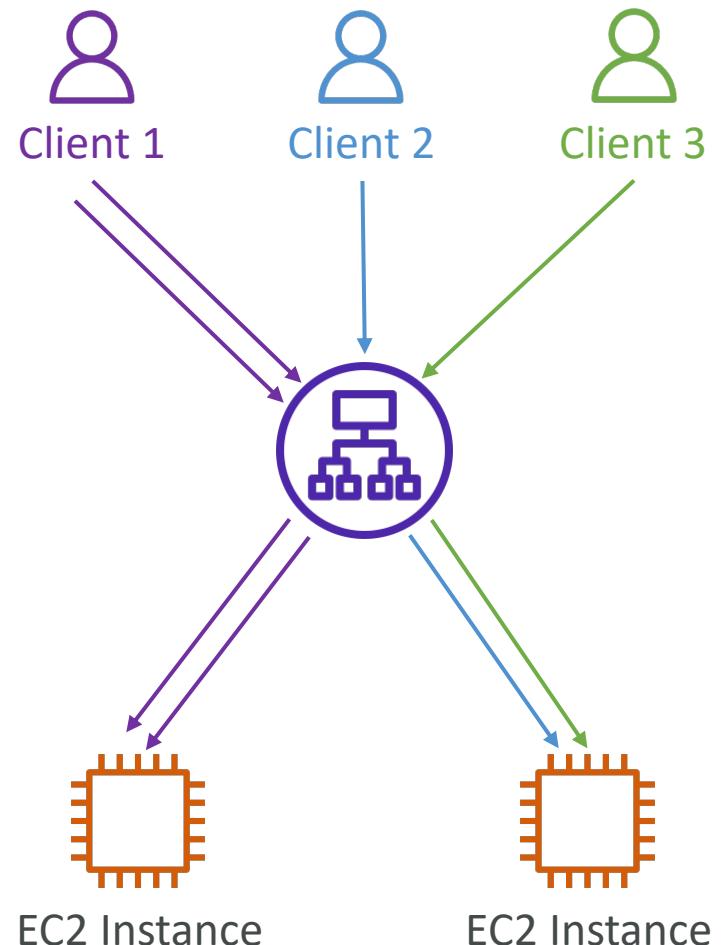


# Cross-Zone Load Balancing

- Classic Load Balancer
  - Disabled by default
  - No charges for inter AZ data if enabled
- Application Load Balancer
  - Always on (can't be disabled)
  - No charges for inter AZ data
- Network Load Balancer
  - Disabled by default
  - You pay charges (\$) for inter AZ data if enabled
- Gateway Load Balancer
  - Disabled by default
  - You pay charges (\$) for inter AZ data if enabled

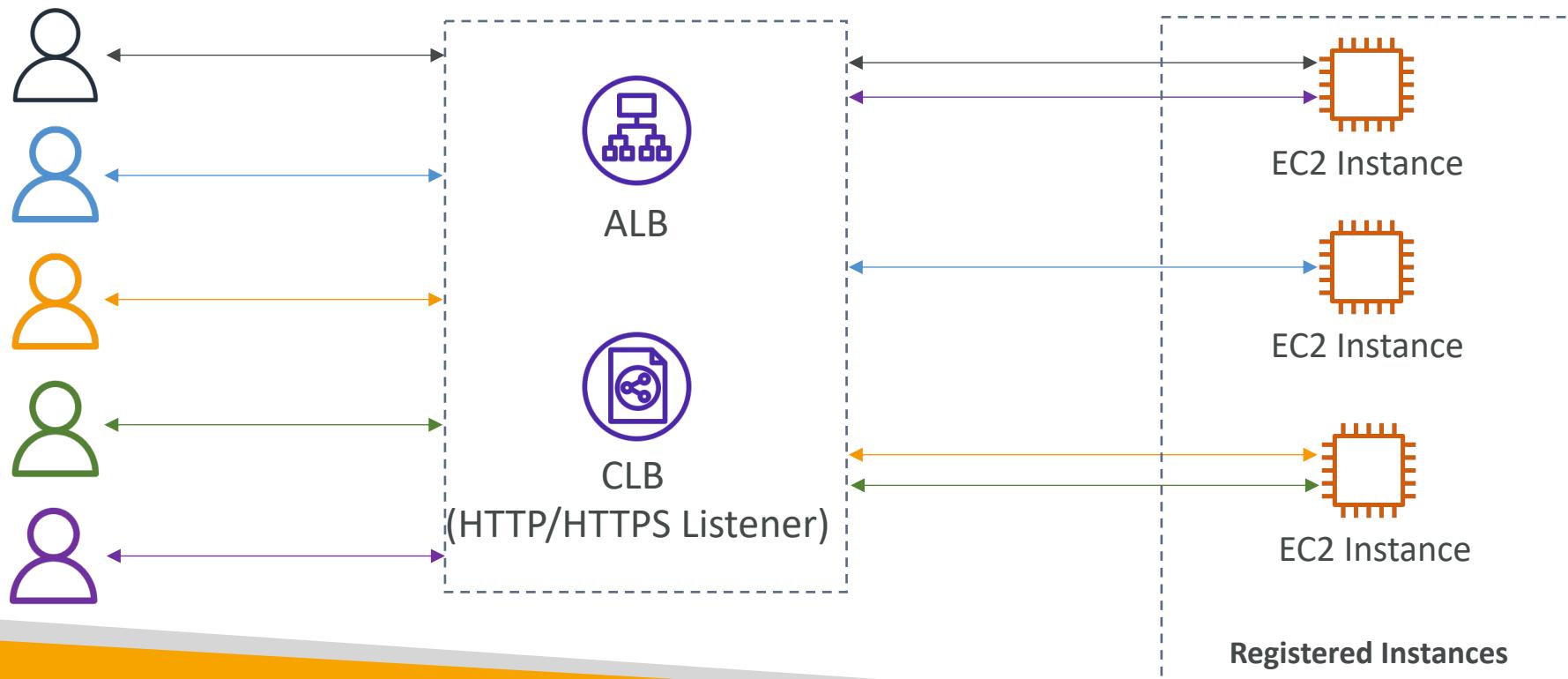
# Sticky Sessions (Session Affinity)

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



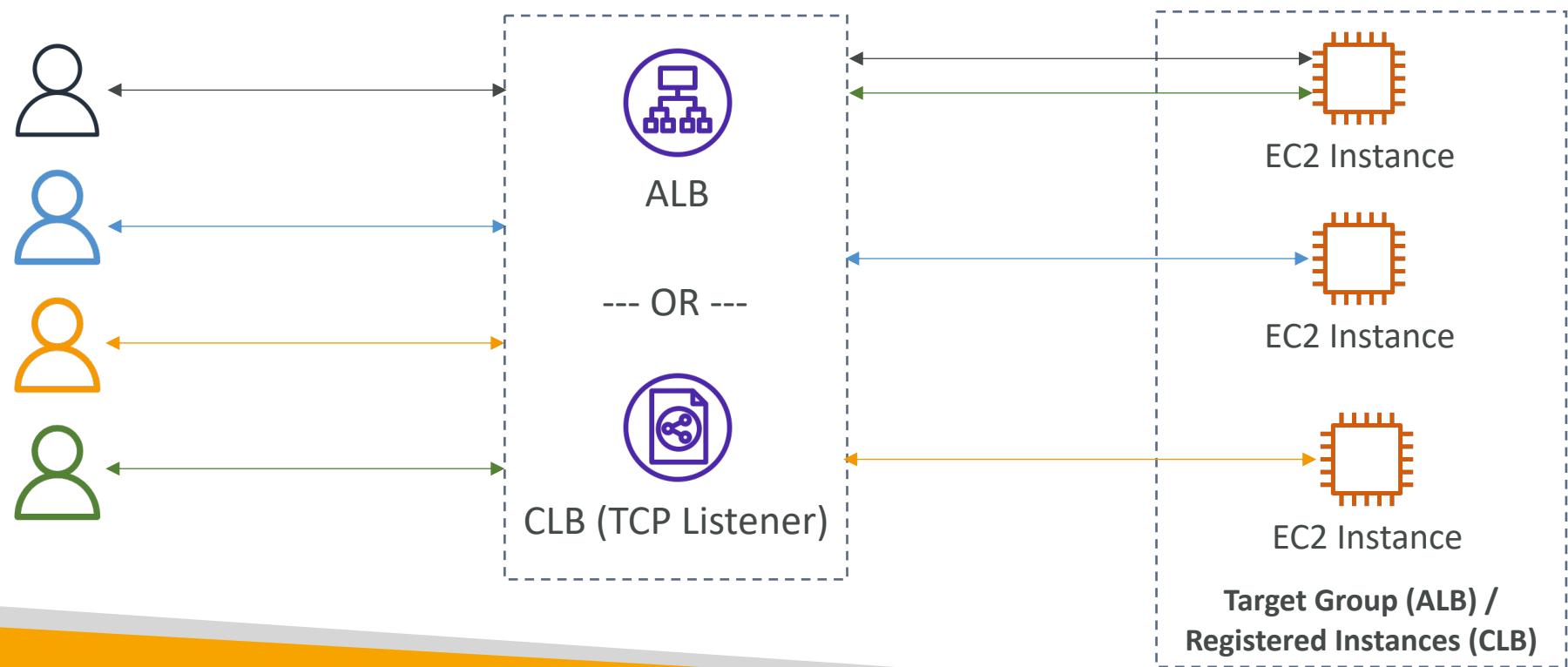
# Request Routing Algorithms – Least Outstanding Requests

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



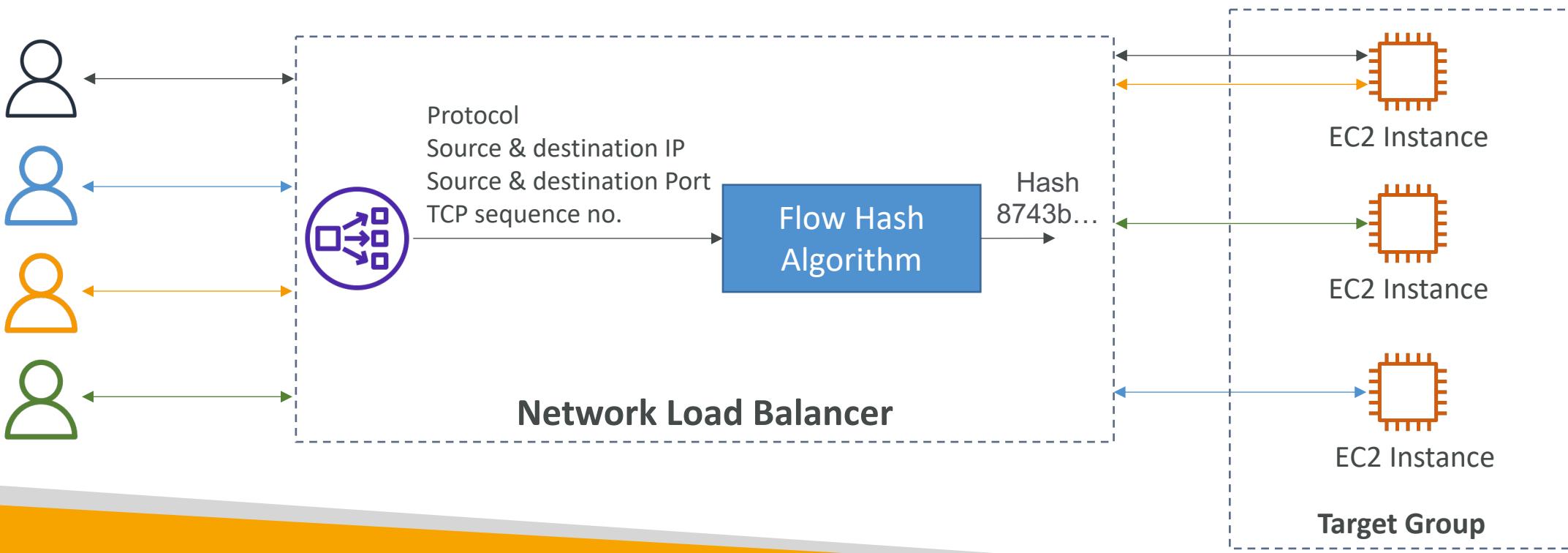
# Request Routing Algorithms – Round Robin

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

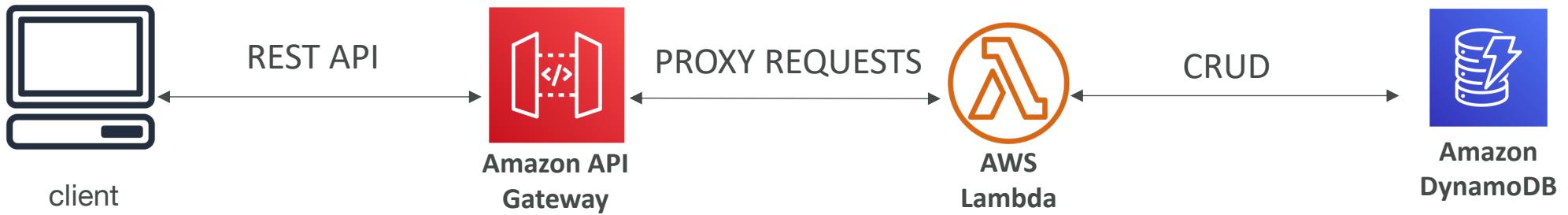


# Request Routing Algorithms – Flow Hash

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



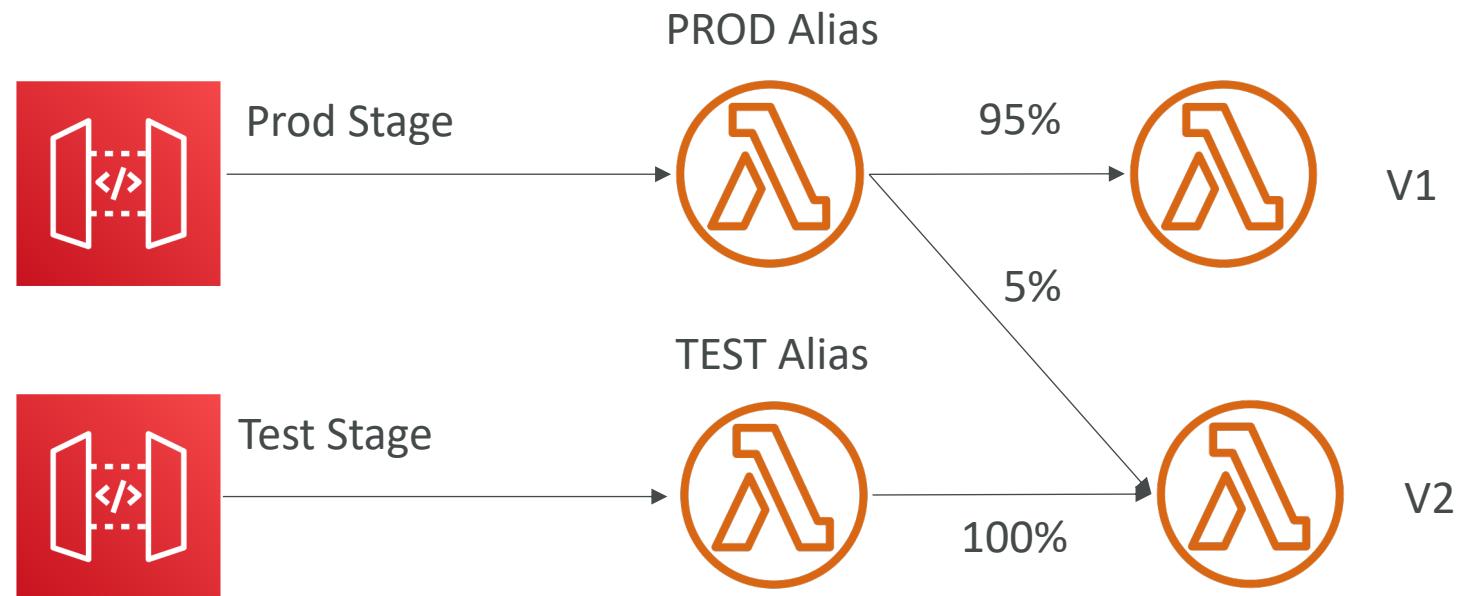
# API Gateway – Overview



- Helps expose Lambda, HTTP & AWS Services as an API
- API versioning, authorization, traffic management (API keys, throttles), huge scale, serverless, req/resp transformations, OpenAPI spec, CORS
- **Limits to know:**
  - 29 seconds timeout
  - 10 MB max payload size

# API Gateway – Deployment Stages

- API changes are deployed to “Stages” (as many as you want)
- Use the naming you like for stages (dev, test, prod)
- Stages can be rolled back as a history of deployments is kept

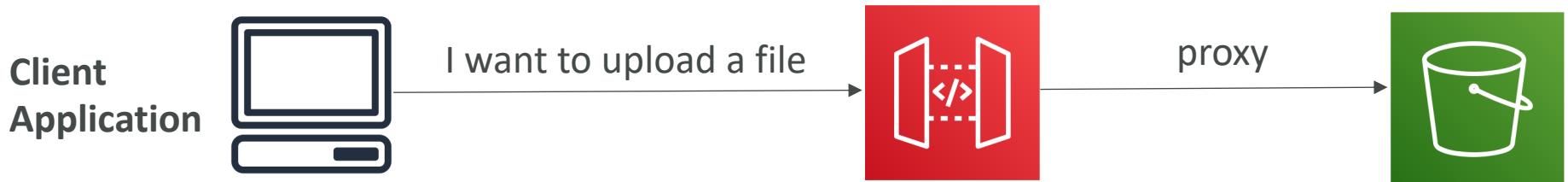


# API Gateway – Integrations

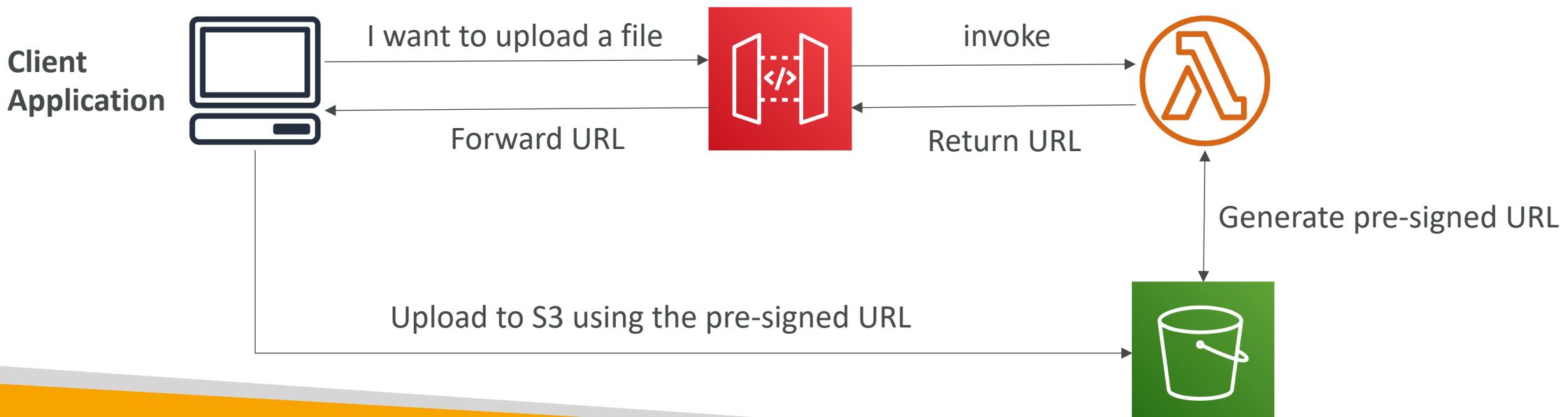
- **HTTP**
  - Expose HTTP endpoints in the backend
  - Example: internal HTTP API on premise, Application Load Balancer...
  - Why? Add rate limiting, caching, user authentications, API keys, etc...
- **Lambda Function**
  - Invoke Lambda function
  - Easy way to expose REST API backed by AWS Lambda
- **AWS Service**
  - Expose any AWS API through the API Gateway?
  - Example: start an AWS Step Function workflow, post a message to SQS
  - Why? Add authentication, deploy publicly, rate control...

# Solution Architecture Discussion: API Gateway in front of S3

- You will be impacted by the 10 MB payload size limit



- Better architecture:

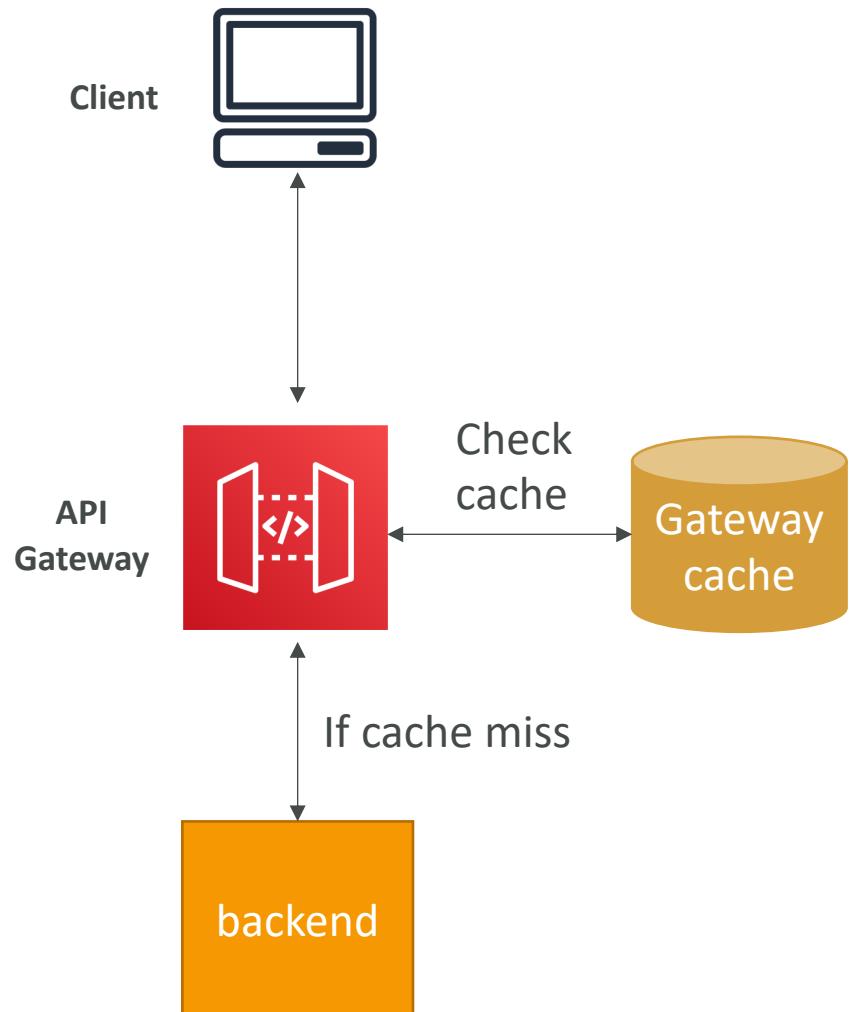


# API Gateway - Endpoint Types

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

# Caching API responses

- Caching reduces the number of calls made to the backend
- Default TTL (time to live) is 300 seconds (min: 0s, max: 3600s)
- Caches are defined **per stage**
- Possible to override cache settings **per method**
- Clients can invalidate the cache with header: **Cache-Control: max-age=0** (with proper IAM authorization)
- Able to flush the entire cache (invalidate it) immediately
- Cache encryption option
- Cache capacity between 0.5GB to 237GB



# API Gateway - Errors

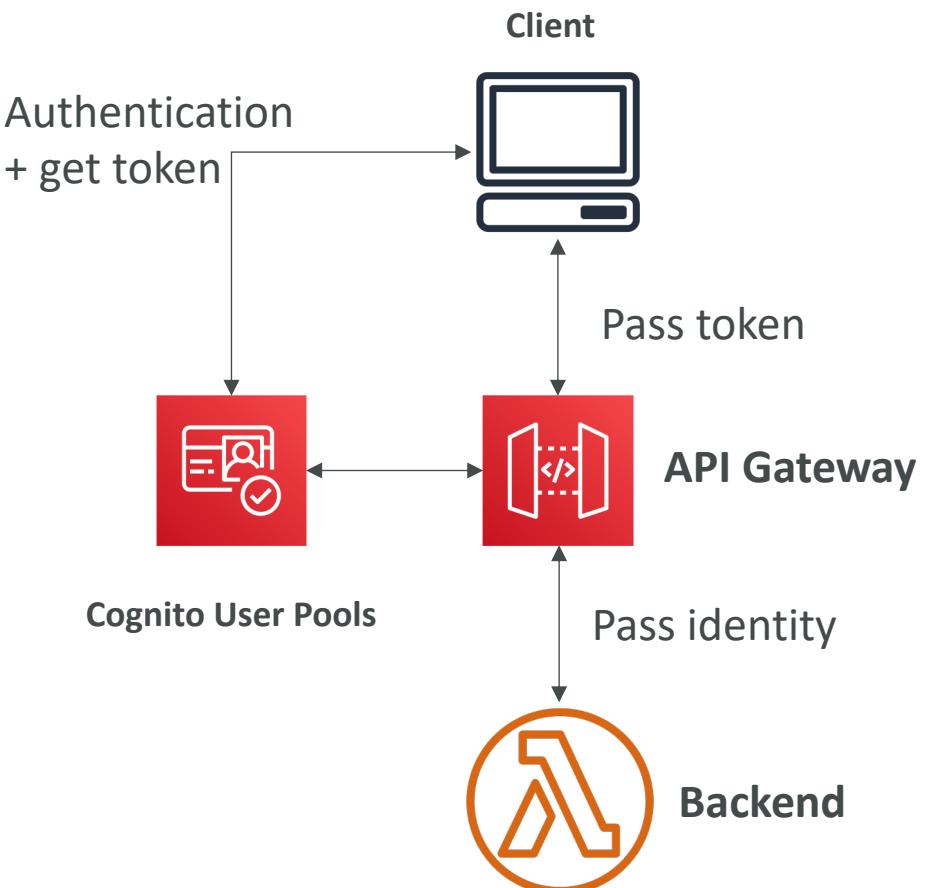
- 4xx means Client errors
  - 400: Bad Request
  - 403: Access Denied, WAF filtered
  - 429: Quota exceeded, Throttle
- 5xx means Server errors
  - 502: Bad Gateway Exception, usually for an incompatible output returned from a Lambda proxy integration backend and occasionally for out-of-order invocations due to heavy loads.
  - 503: Service Unavailable Exception
  - 504: Integration Failure – ex Endpoint Request Timed-out Exception  
**API Gateway requests time out after 29 second maximum**

# API Gateway – Security

- Load SSL certificates and use Route53 to define a CNAME
- Resource Policy (~S3 Bucket Policy):
  - control who can access the API
  - Users from AWS accounts, IP or CIDR blocks, VPC or VPC Endpoints
- IAM Execution Roles for API Gateway at the API level
  - To invoke a Lambda Function, an AWS service...
- CORS (Cross-origin resource sharing):
  - Browser based security
  - Control which domains can call your API

# API Gateway – Authentication

- IAM based access (**AWS\_IAM**)
  - Good for providing access within your infrastructure
  - Pass IAM credentials in headers through SigV4
- Lambda Authorizer (formerly Custom Authorizer)
  - Use Lambda to verify a custom OAuth / SAML / 3<sup>rd</sup> party authentication
- Cognito User Pools
  - Client authenticates with Cognito
  - Client passes the token to API Gateway
  - API Gateway knows out-of-the-box how to verify to token



# API Gateway – Logging, Monitoring, Tracing

- **CloudWatch Logs:**

- Enable CloudWatch logging at the Stage level (with Log Level – ERROR, INFO)
- Can log full requests / responses data
- Can send API Gateway Access Logs (customizable)
- Can send logs directly into Kinesis Data Firehose (as an alternative to CW logs)

- **CloudWatch Metrics:**

- Metrics are by stage, possibility to enable detailed metrics
- *IntegrationLatency, Latency, CacheHitCount, CacheMissCount*

- **X-Ray:**

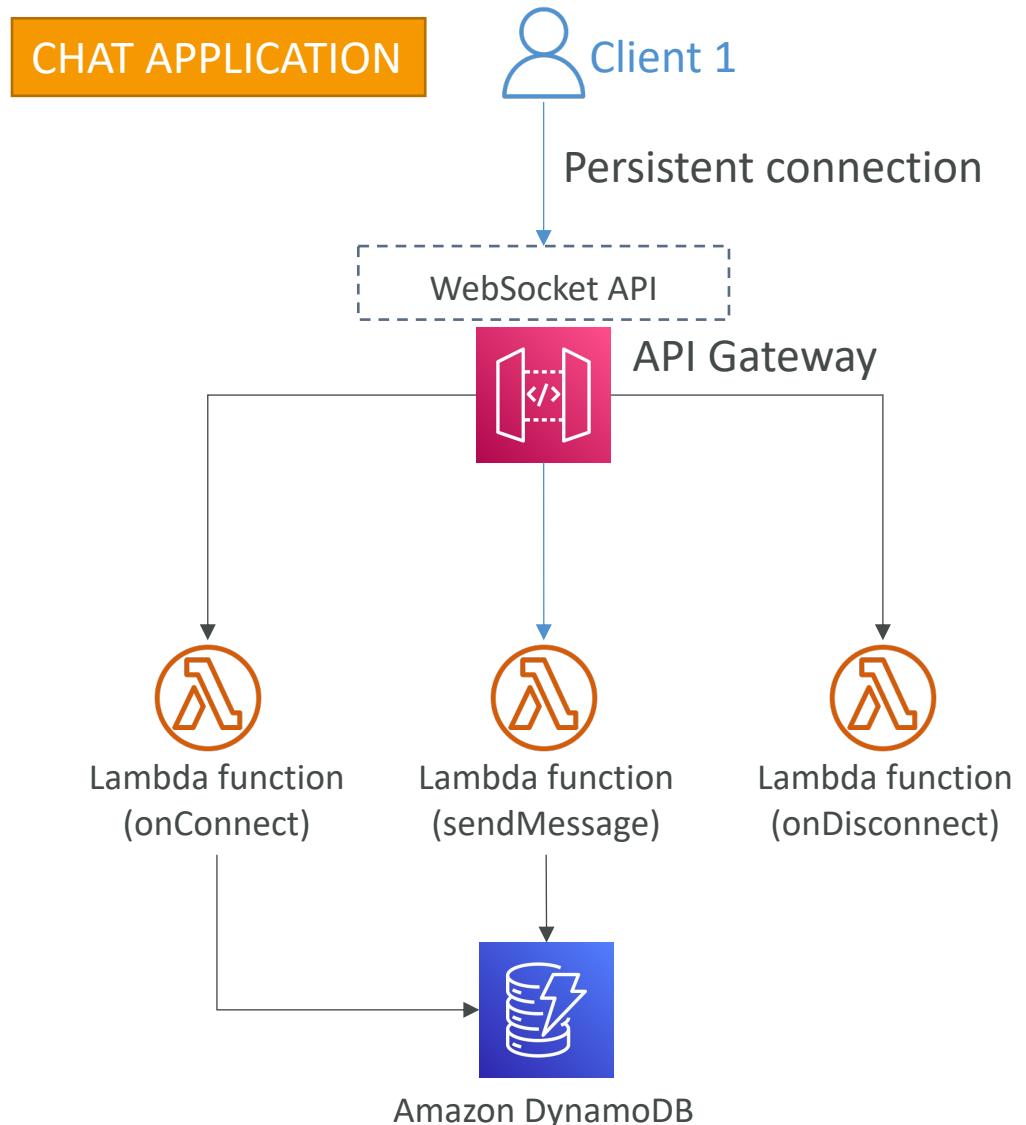
- Enable tracing to get extra information about requests in API Gateway
- X-Ray API Gateway + AWS Lambda gives you the full picture

# API Gateway – Usage Plans & API Keys

- If you want to make an API available as an offering (\$) to your customers
- **Usage Plan:**
  - who can access one or more deployed API stages and methods
  - how much and how fast they can access them
  - uses API keys to identify API clients and meter access
  - configure throttling limits and quota limits that are enforced on individual client
- **API Keys:**
  - alphanumeric string values to distribute to your customers
  - Ex: WBjHxNtoAb4WPKBC7cGm64CBiblb24b4jt8jjHo9
  - Can use with usage plans to control access
  - Throttling limits are applied to the API keys
  - Quotas limits is the overall number of maximum requests

# API Gateway – WebSocket API – Overview

- What's WebSocket?
  - Two-way interactive communication between a user's browser and a server
  - Server can push information to the client
  - This enables **stateful** application use cases
- WebSocket APIs are often used in **real-time applications** such as chat applications, collaboration platforms, multiplayer games, and financial trading platforms.
- Works with AWS Services (Lambda, DynamoDB) or HTTP endpoints

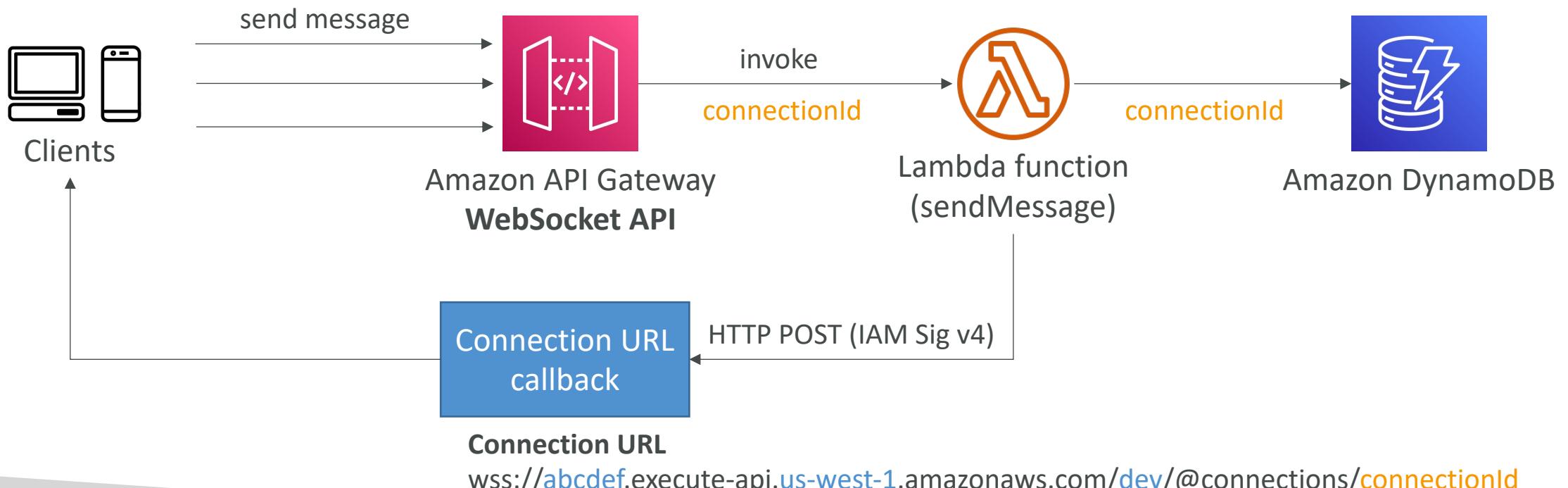


# Server to Client Messaging

## @connections used for replies to clients

### WebSocket URL

wss://abcdef.execute-api.us-west-1.amazonaws.com/dev

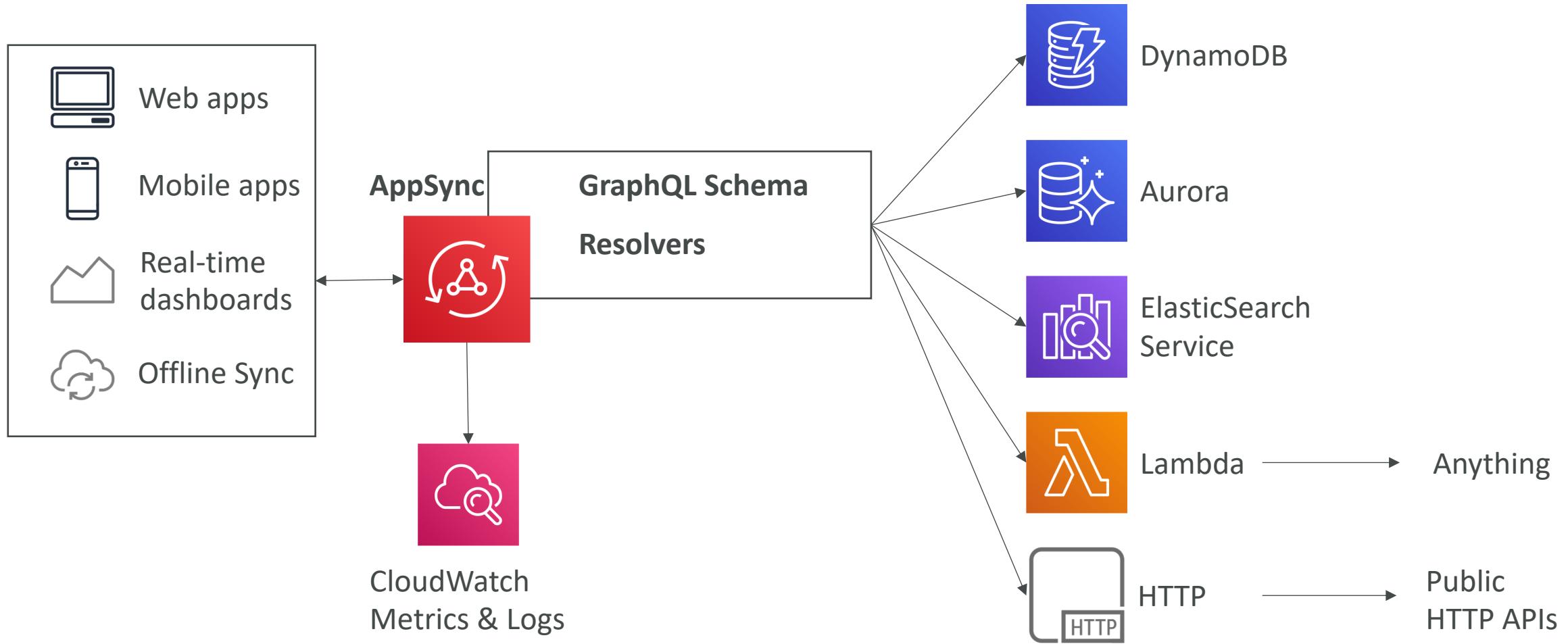


# AWS AppSync - Overview



- AppSync is a managed service that uses **GraphQL**
- **GraphQL** makes it easy for applications to get exactly the data they need.
- This includes combining data from **one or more sources**
  - NoSQL data stores, Relational databases, HTTP APIs...
  - Integrates with DynamoDB, Aurora, Elasticsearch & others
  - Custom sources with AWS Lambda
- Retrieve data in **real-time** with **WebSocket** or **MQTT** on **WebSocket**
- For mobile apps: local data access & data synchronization
- It all starts with uploading one **GraphQL schema**

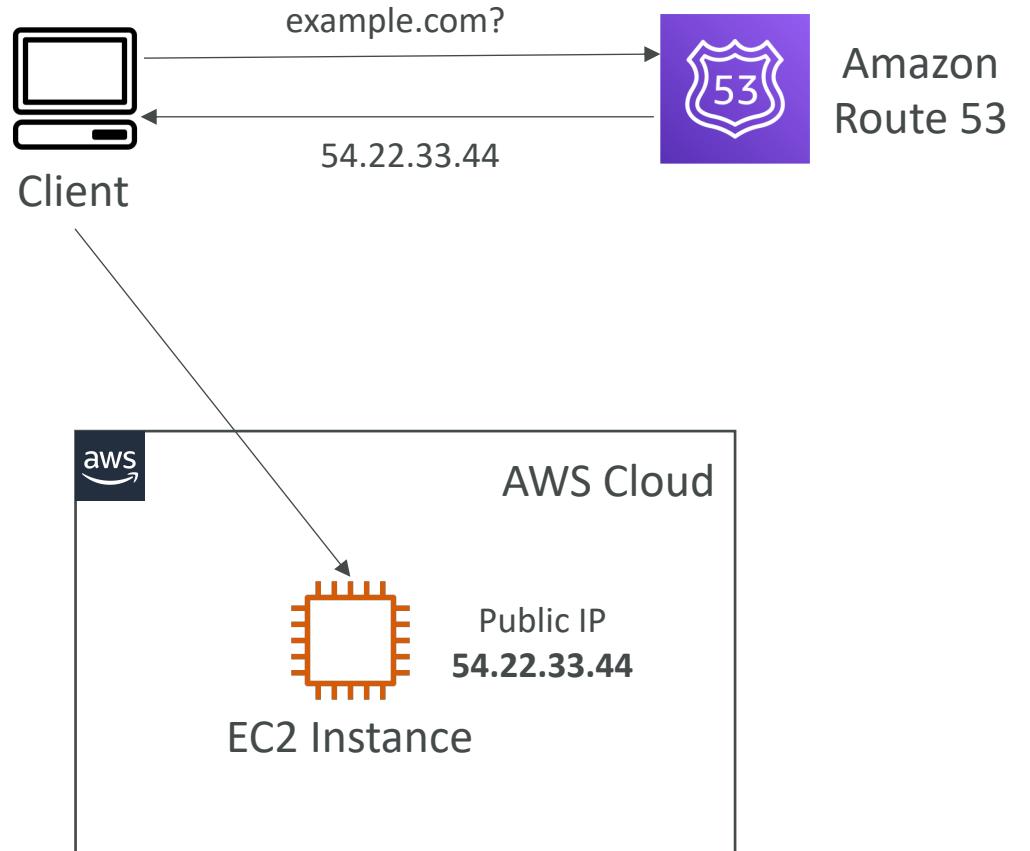
# AppSync Diagram



# Route 53 – Record Types

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

# Route 53 – Diagram for A record



# Route 53 – CNAME vs. Alias

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

# Route 53 – Alias Records Targets

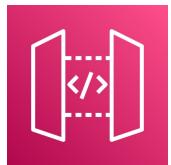
- Elastic Load Balancers
- CloudFront Distributions
- API Gateway
- Elastic Beanstalk environments
- S3 Websites
- VPC Interface Endpoints
- Global Accelerator accelerator
- Route 53 record in the same hosted zone
- You cannot set an ALIAS record for an EC2 DNS name



Elastic  
Load Balancer



Amazon  
CloudFront



Amazon  
API Gateway



Elastic Beanstalk



S3 Websites



VPC Interface  
Endpoints



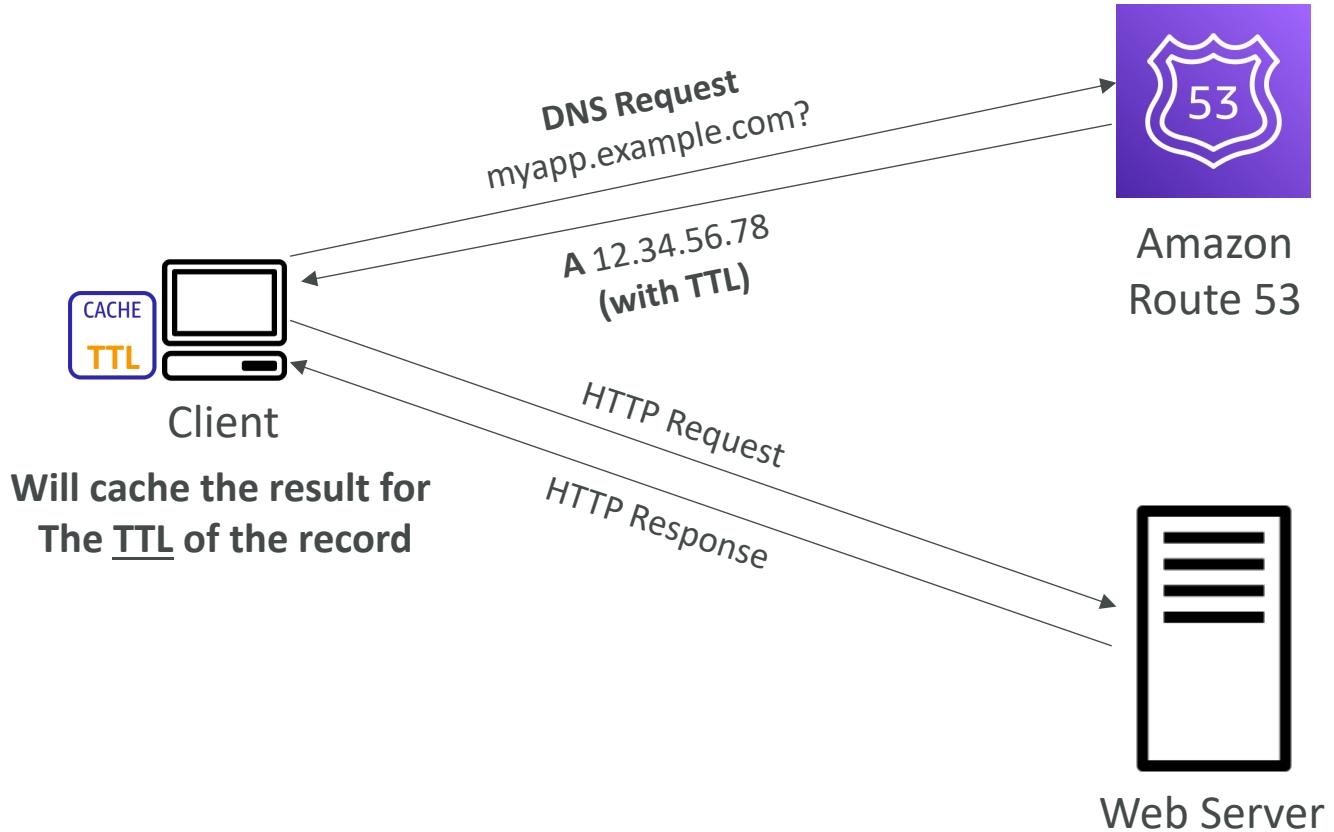
Global Accelerator



Route 53 Record  
(same Hosted Zone)

# Route 53 – Records TTL (Time To Live)

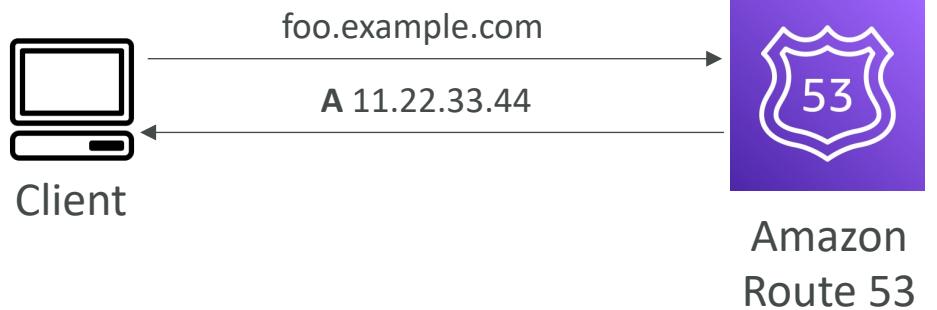
- High TTL – e.g., 24 hr
  - Less traffic on Route 53
  - Possibly outdated records
- Low TTL – e.g., 60 sec.
  - More traffic on Route 53 (\$\$)
  - Records are outdated for less time
  - Easy to change records
- Except for Alias records, TTL is mandatory for each DNS record



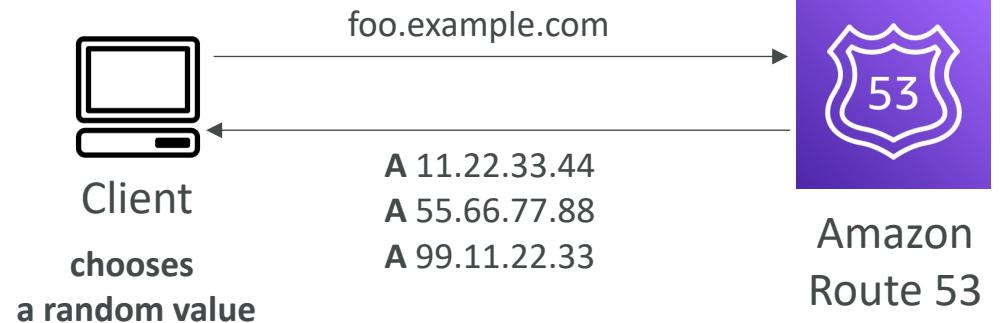
# Routing Policies – Simple

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

## Single Value

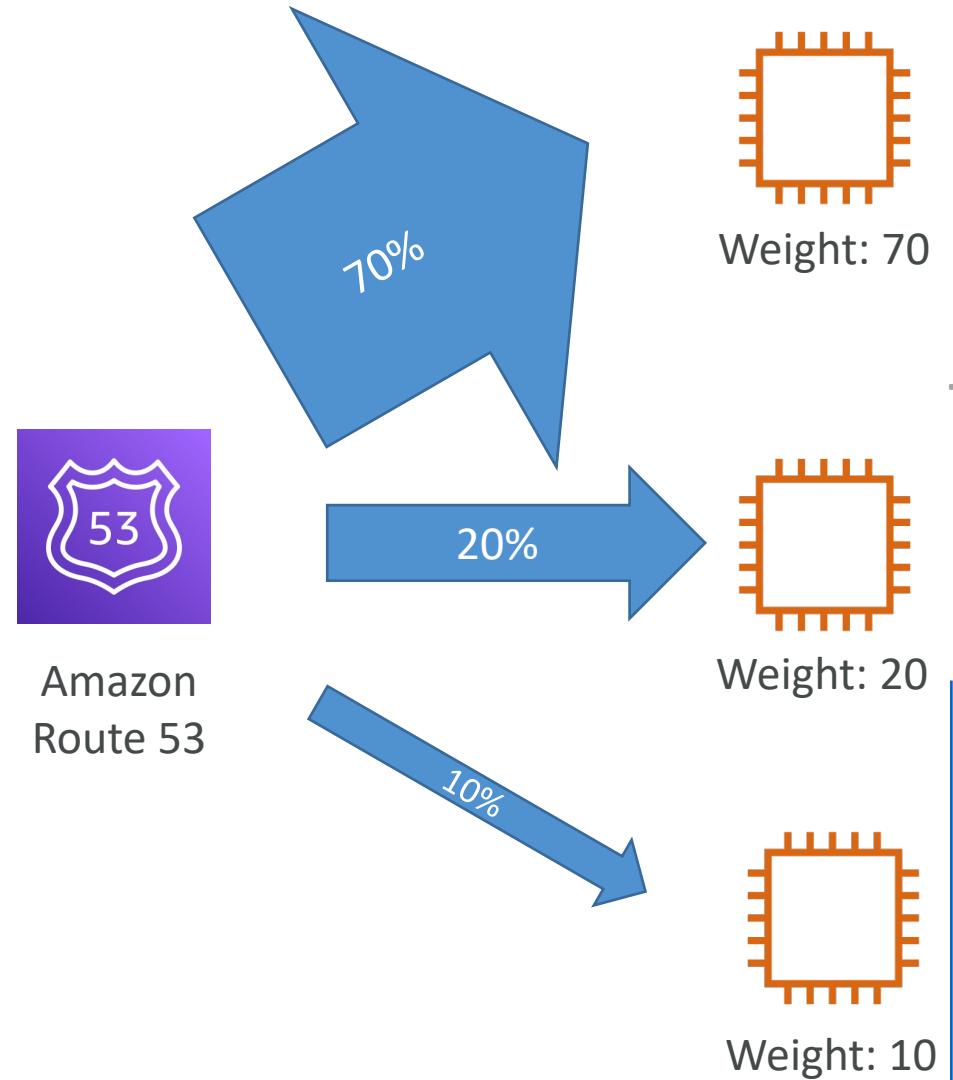


## Multiple Value



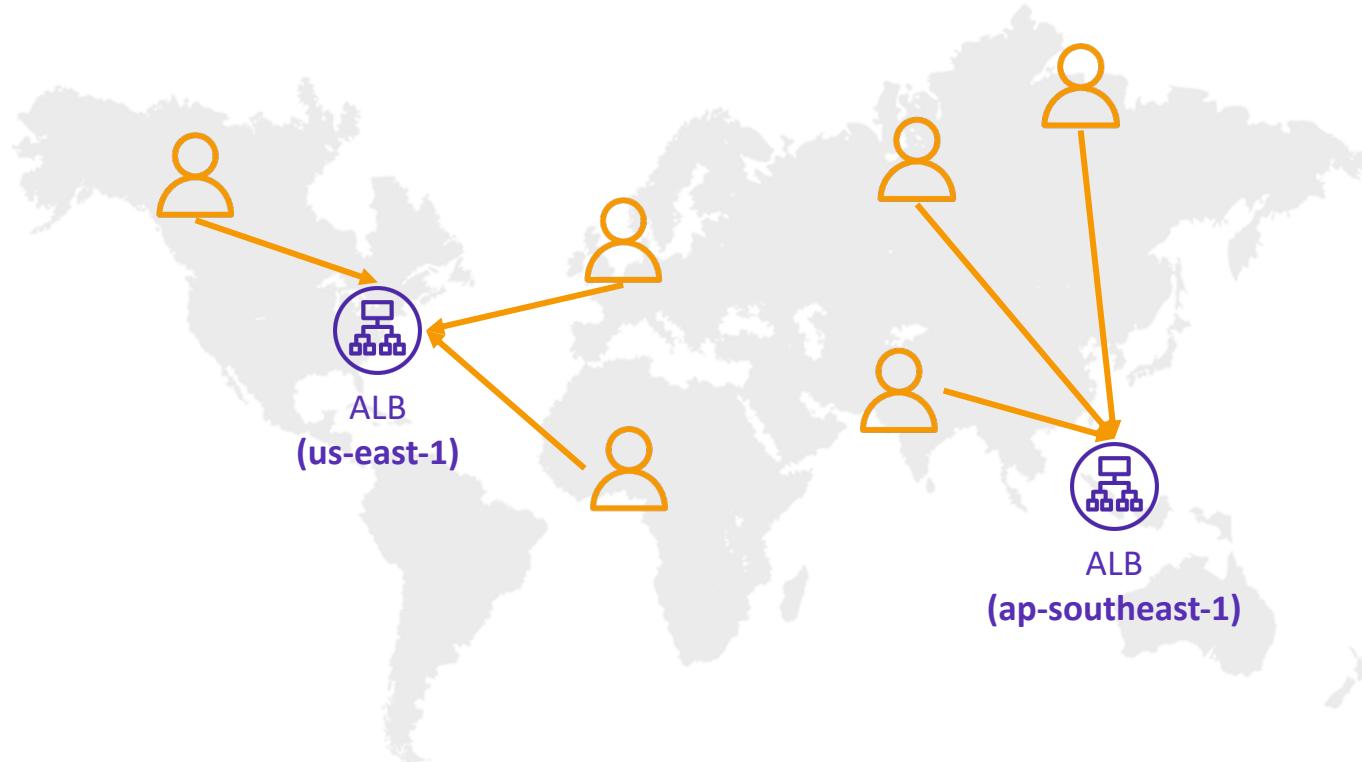
# Routing Policies – Weighted

- Control the % of the requests that go to each specific resource
- Can be associated with Health Checks
- Use cases: load balancing between regions, testing new application versions...

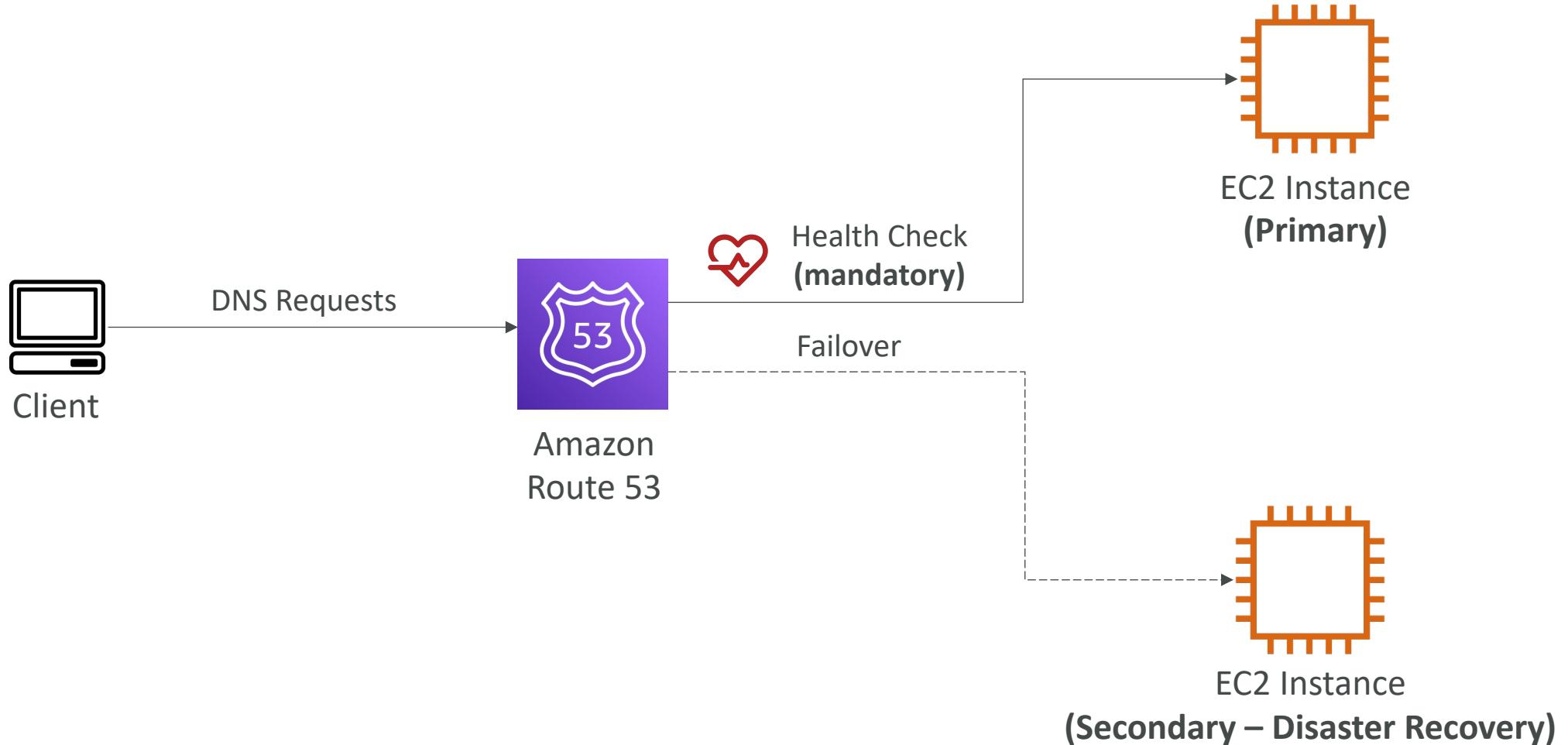


# Routing Policies – Latency-based

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

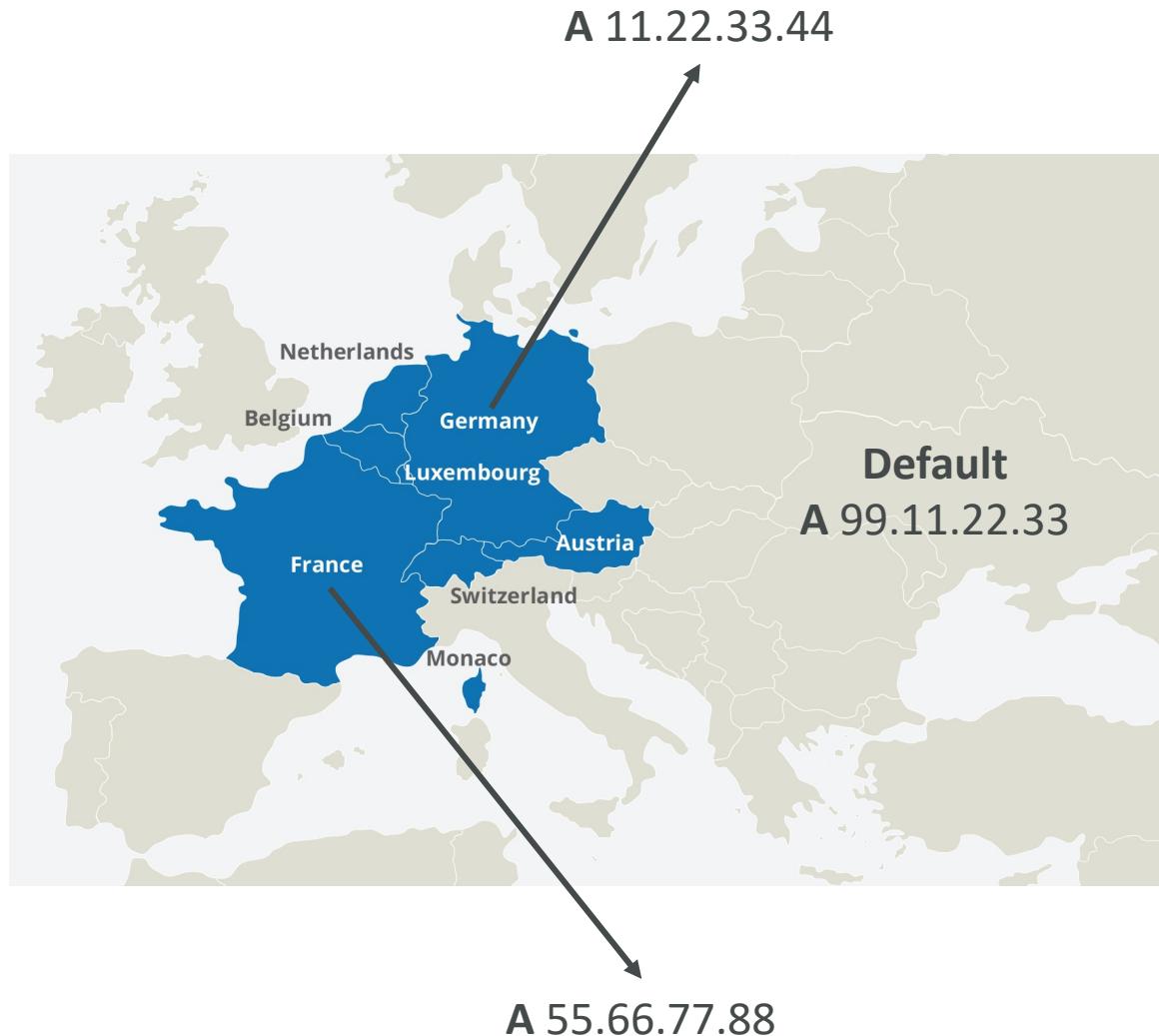


# Routing Policies – Failover (Active-Passive)



# Routing Policies – Geolocation

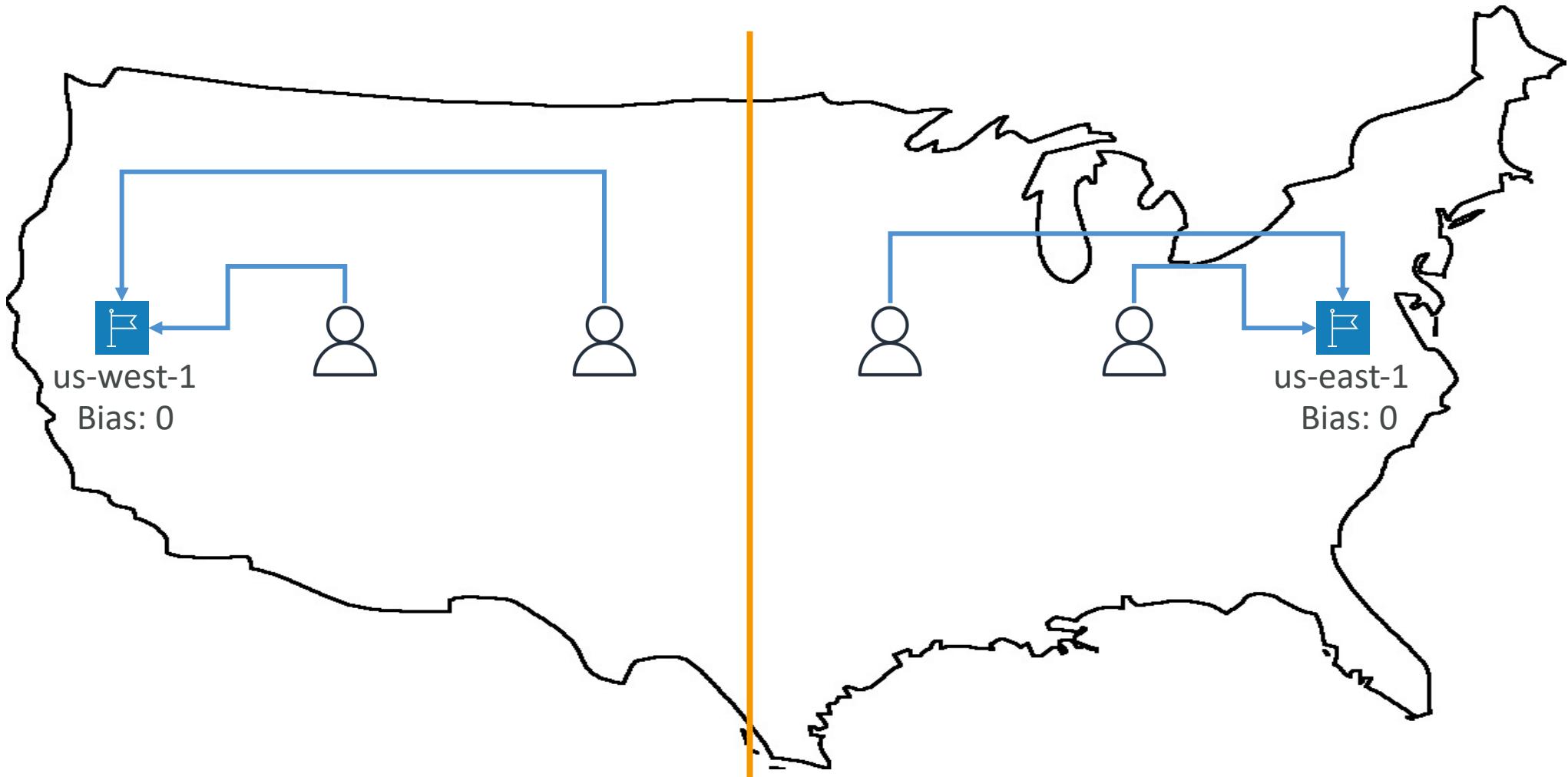
- Different from Latency-based!
- This routing is based on user location
- Specify location by Continent, Country or by US State (if there's overlapping, most precise location selected)
- Should create a “Default” record (in case there's no match on location)
- Use cases: website localization, restrict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



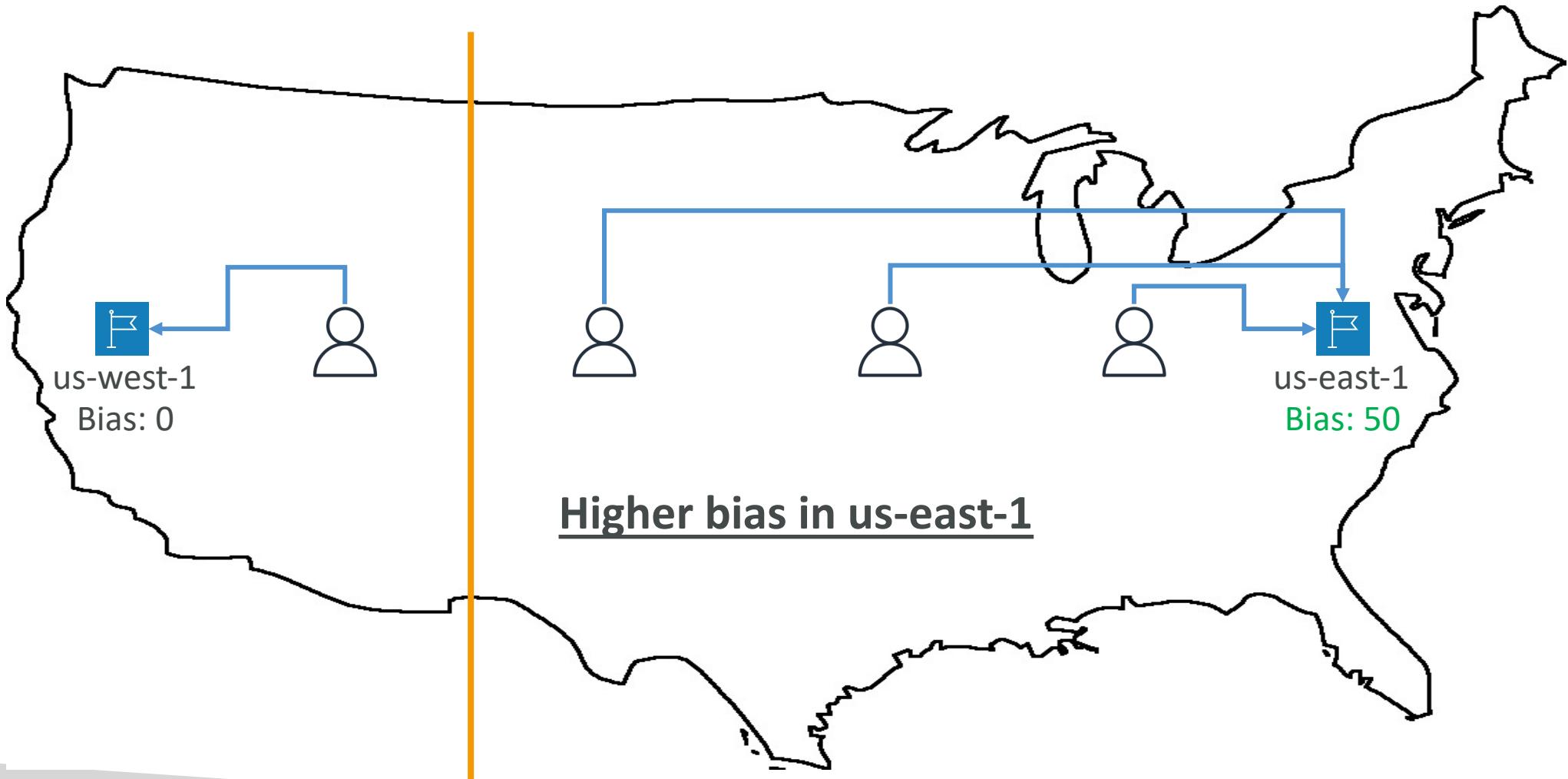
# Routing Policies – Geoproximity

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

# Routing Policies – Geoproximity

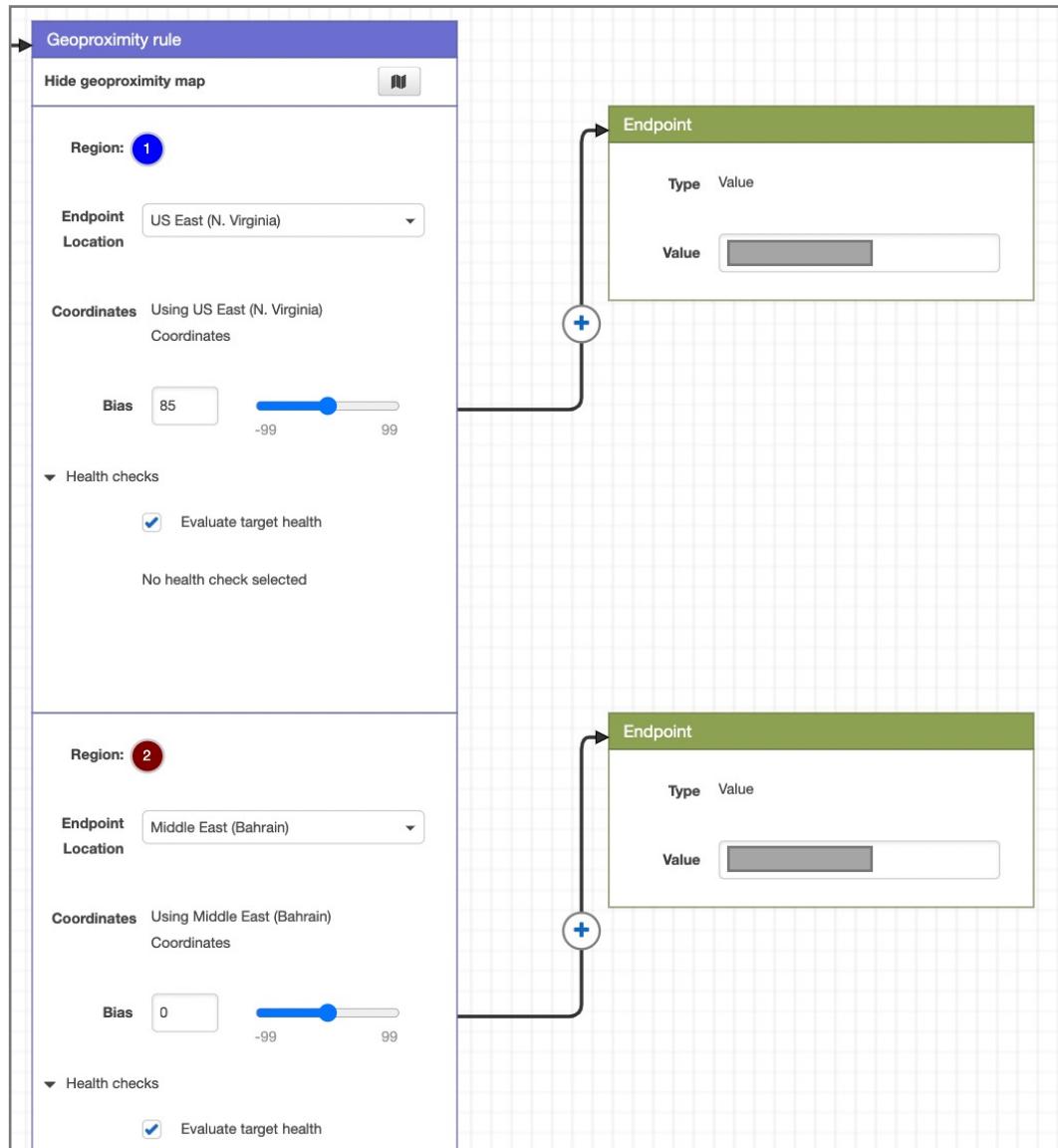


# Routing Policies – Geoproximity



# Route 53 – Traffic flow

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



# Routing Policies – Multi-Value

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

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

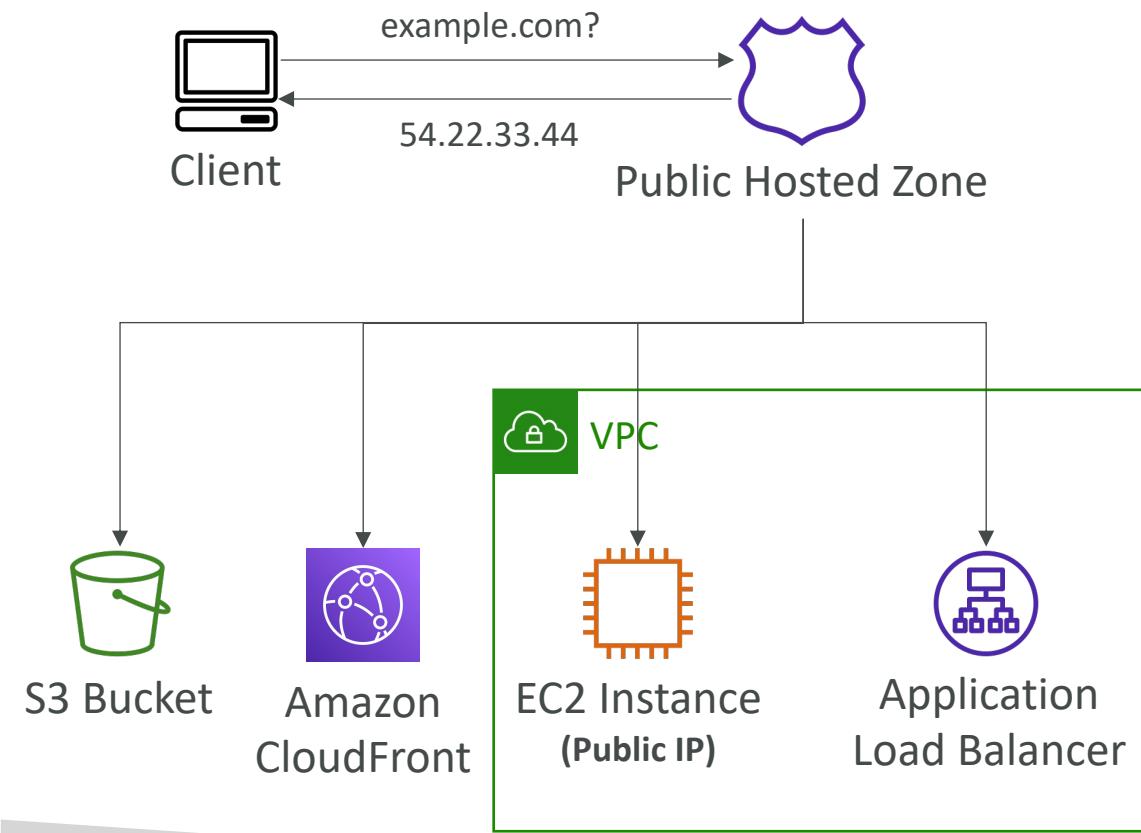


# Route 53 – Hosted Zones

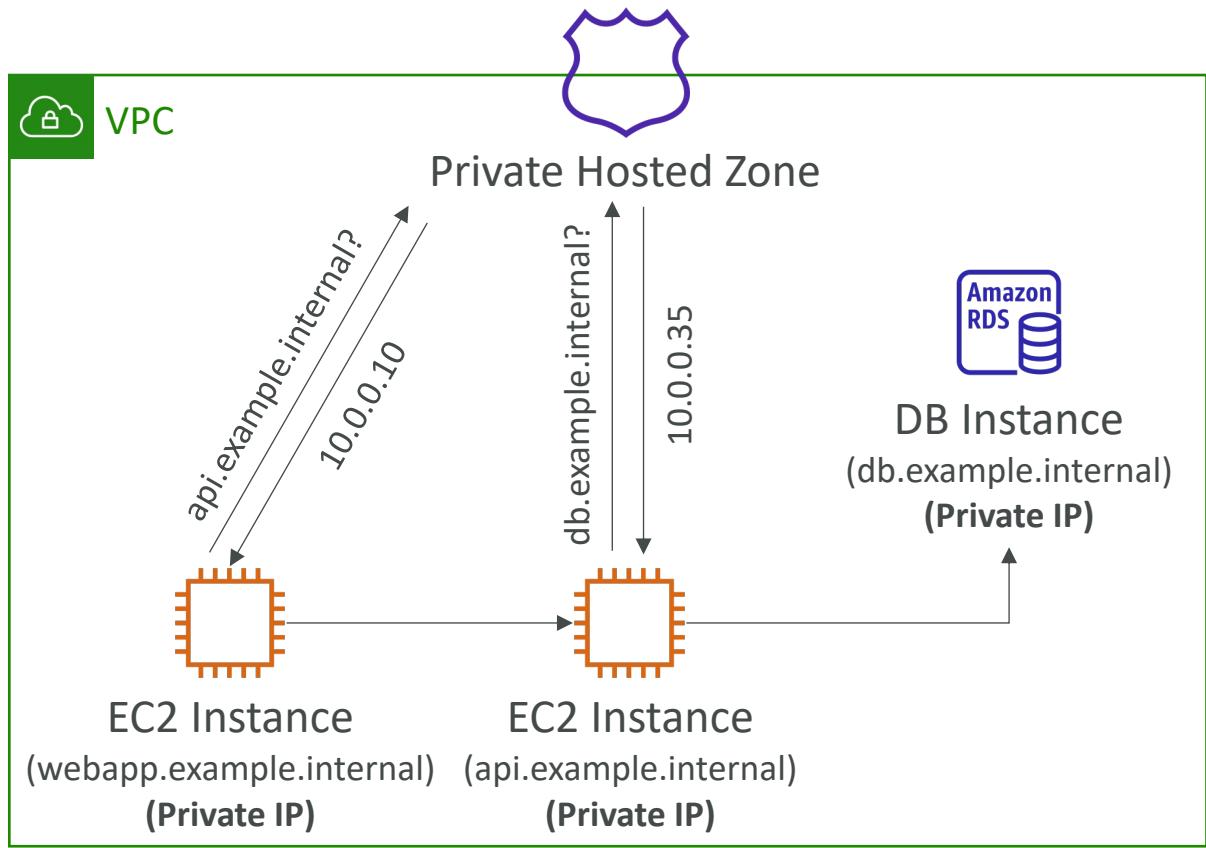
- A container for records that define how to route traffic to a domain and its subdomains
- **Public Hosted Zones** – contains records that specify how to route traffic on the Internet (public domain names)  
[application1.mypublicdomain.com](http://application1.mypublicdomain.com)
- **Private Hosted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

# Route 53 – Public vs. Private Hosted Zones

## Public Hosted Zone



## Private Hosted Zone

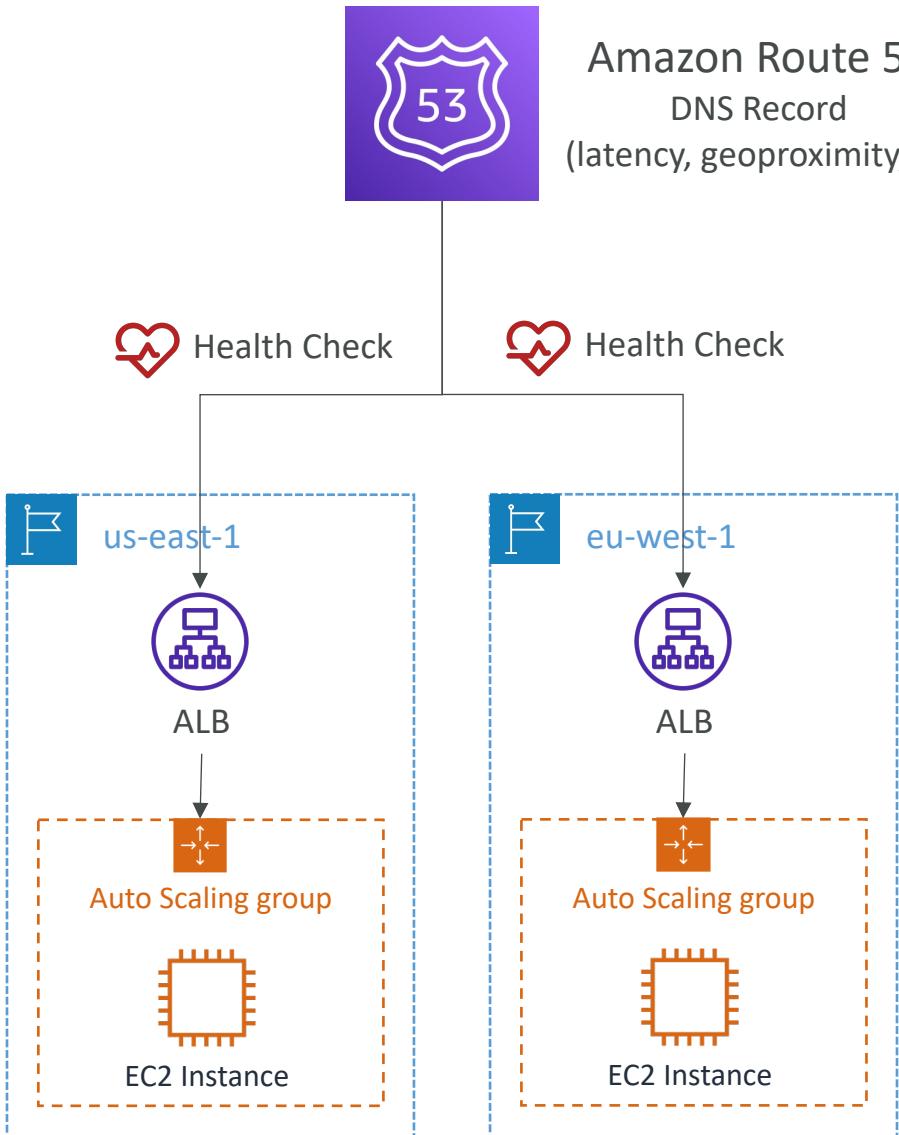


# Route 53 – Good to Know

- For internal private DNS (Private Hosted Zone), you must enable the VPC settings `enableDnsHostnames` and `enableDnsSupport`
- **DNS Security Extensions (DNSSEC)**
  - A protocol for securing DNS traffic, verifies DNS data integrity and origin
  - Protects against Man in the Middle (MITM) attacks
  - Route 53 supports both DNSSEC for Domain Registration and DNSSEC Signing
  - Works only with **Public Hosted Zones**
- **Route 53 with 3<sup>rd</sup> Registrar**
  - You can buy the domain out of AWS and use Route 53 as the DNS provider
  - Update the NS records on the 3<sup>rd</sup> party Registrar

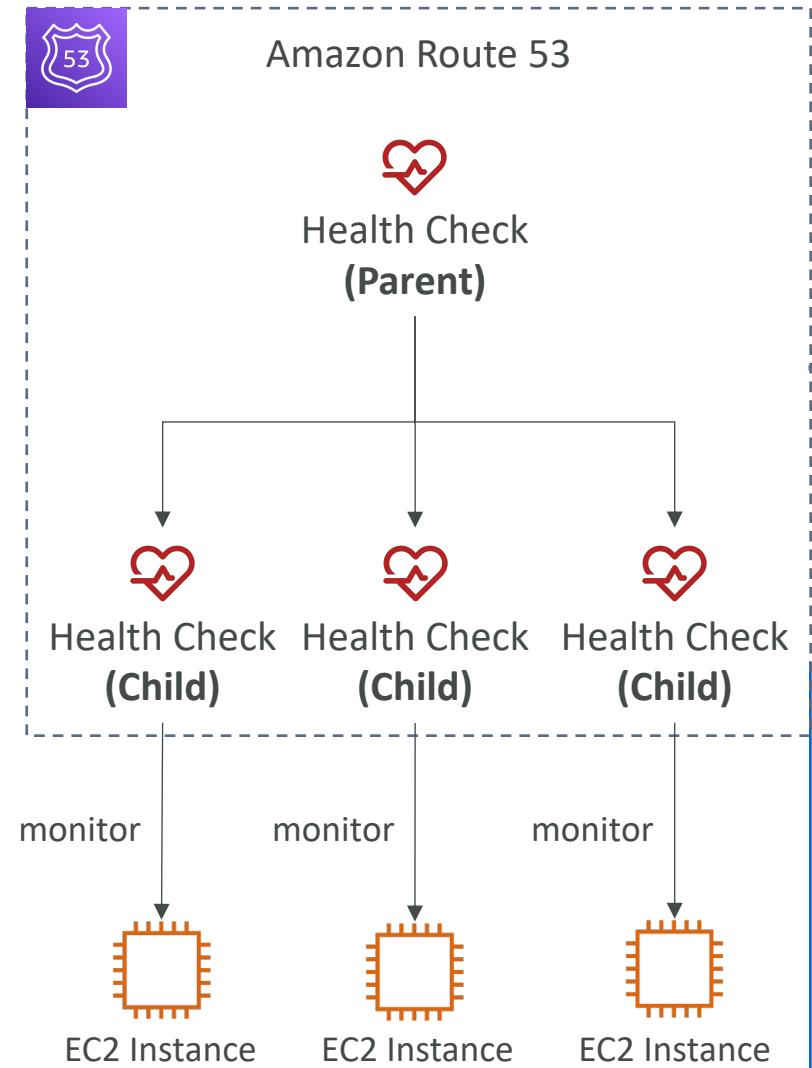
# Route 53 – Health Checks

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



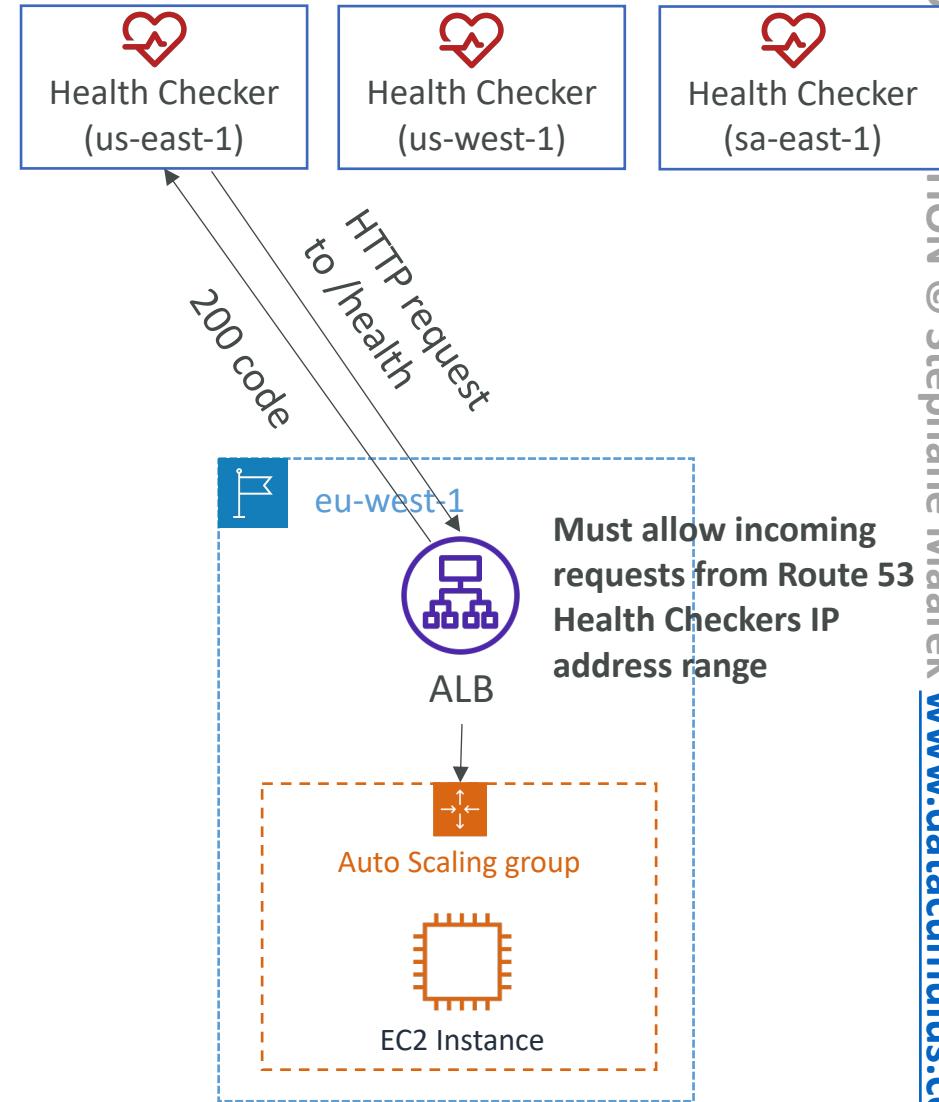
# Route 53 – Calculated Health Checks

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



# Health Checks – Monitor an Endpoint

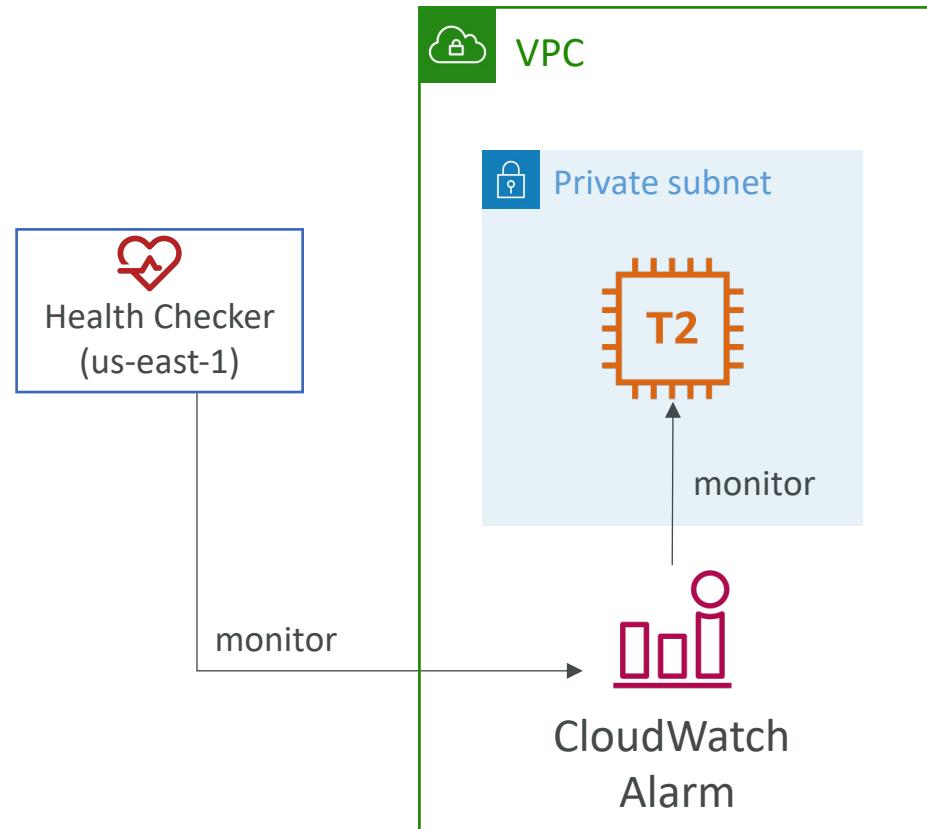
- About 15 global health checkers will check the endpoint health
- Health Checks pass only when the endpoint responds with the 2xx and 3xx status codes
- Health Checks can be setup to pass / fail based on the text in the first 5120 bytes of the response



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

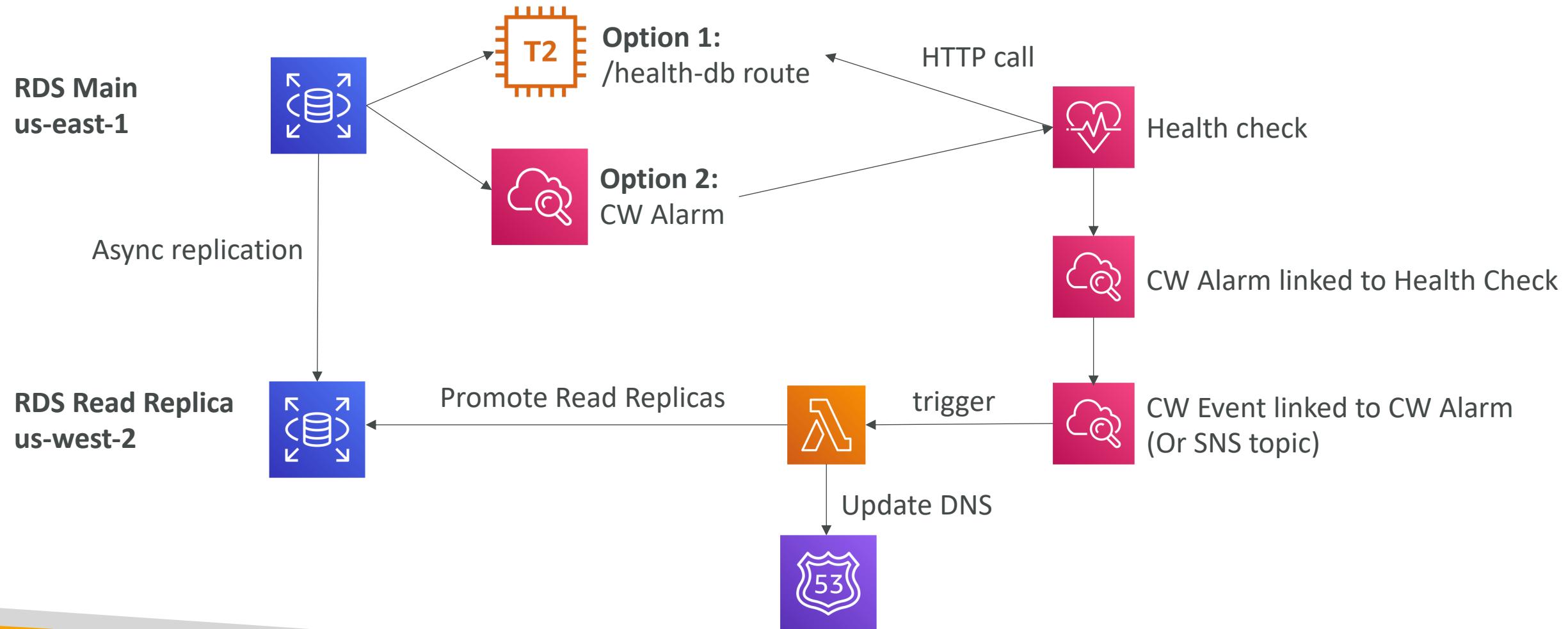
# Health Checks – Private Hosted Zones

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



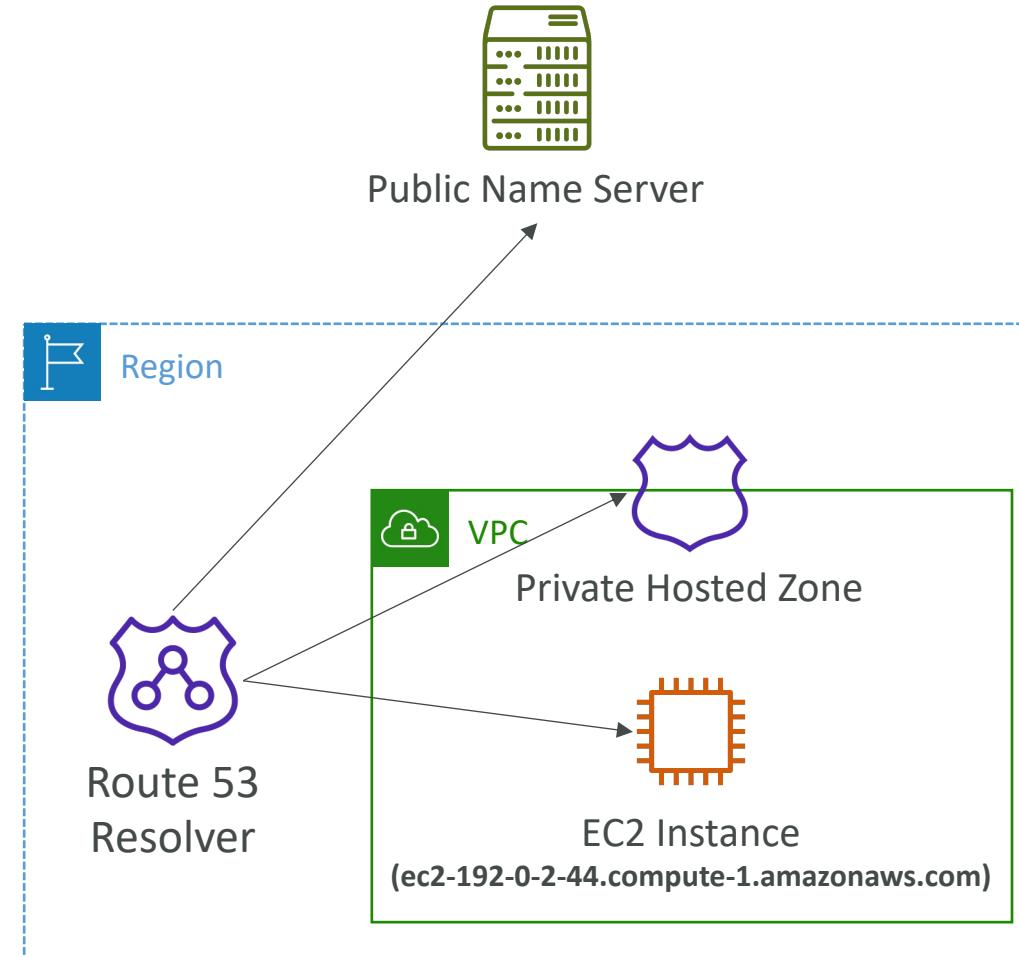
# Health Checks Solution Architecture

## RDS multi-region failover



# Route 53 – Hybrid DNS

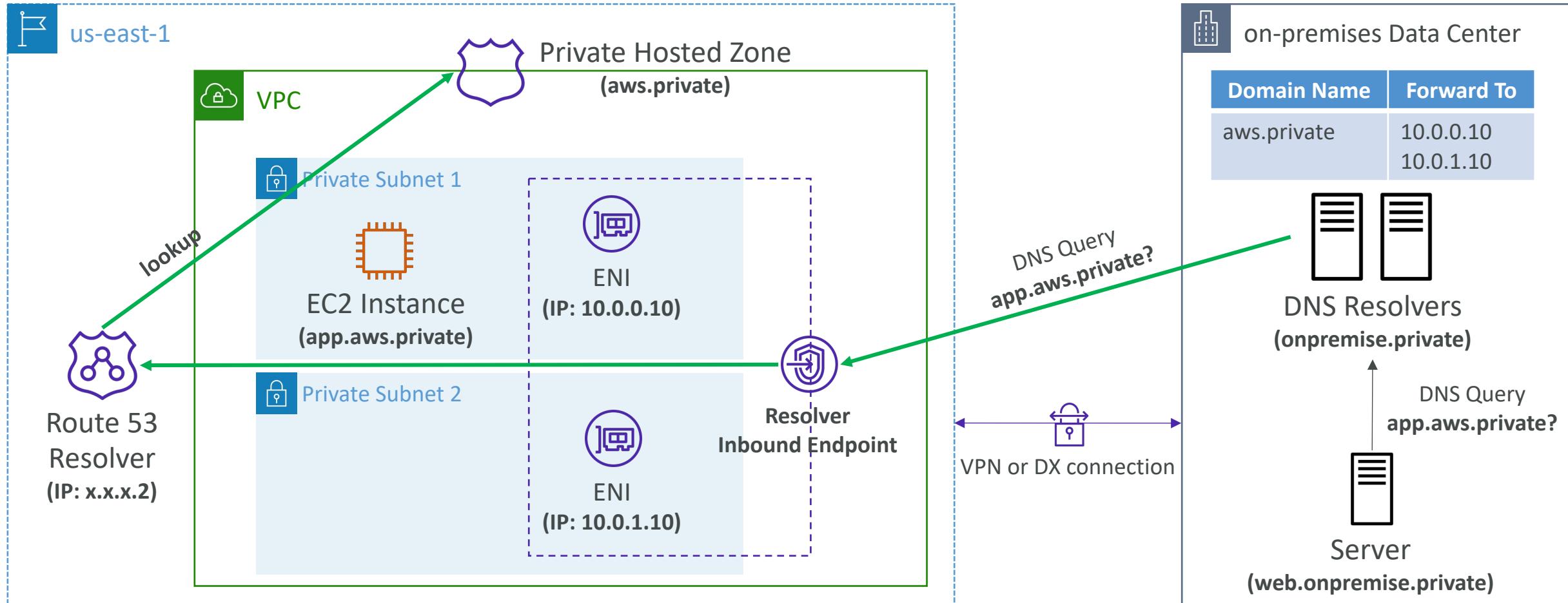
- By default, Route 53 Resolver automatically answers DNS queries for:
  - Local domain names for EC2 instances
  - Records in Private Hosted Zones
  - Records in public Name Servers
- **Hybrid DNS** – resolving DNS queries between VPC (Route 53 Resolver) and your networks (other DNS Resolvers)
- Networks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



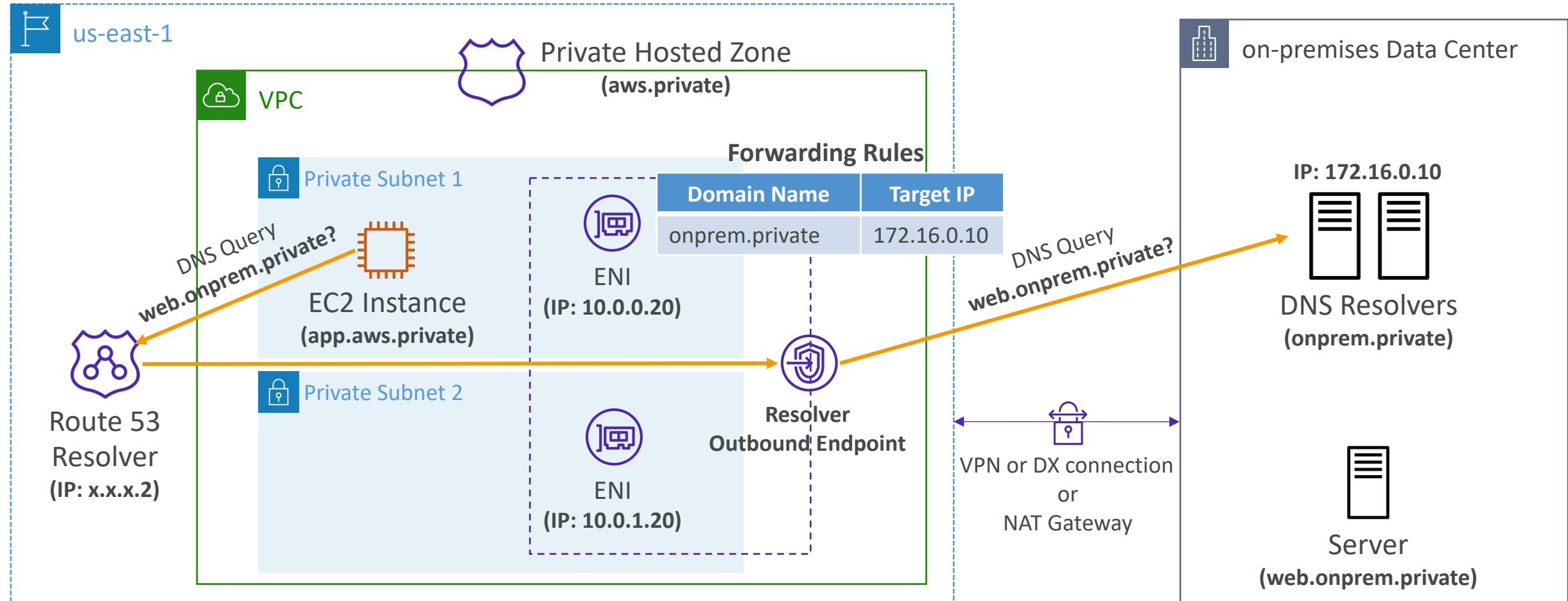
# Route 53 – Resolver Endpoints

- **Inbound Endpoint**
  - DNS Resolvers on your network can forward DNS queries to Route 53 Resolver
  - Allows your DNS Resolvers to resolve domain names for AWS resources (e.g., EC2 instances) and records in Route 53 Private Hosted Zones
- **Outbound Endpoint**
  - Route 53 Resolver conditionally forwards DNS queries to your DNS Resolvers
  - Use **Resolver Rules** to forward DNS queries to your DNS Resolvers
- Associated with one or more VPCs in the same AWS Region
- Create in two AZs for high availability
- Each Endpoint supports 10,000 queries per second per IP address

# Route 53 – Resolver Inbound Endpoints

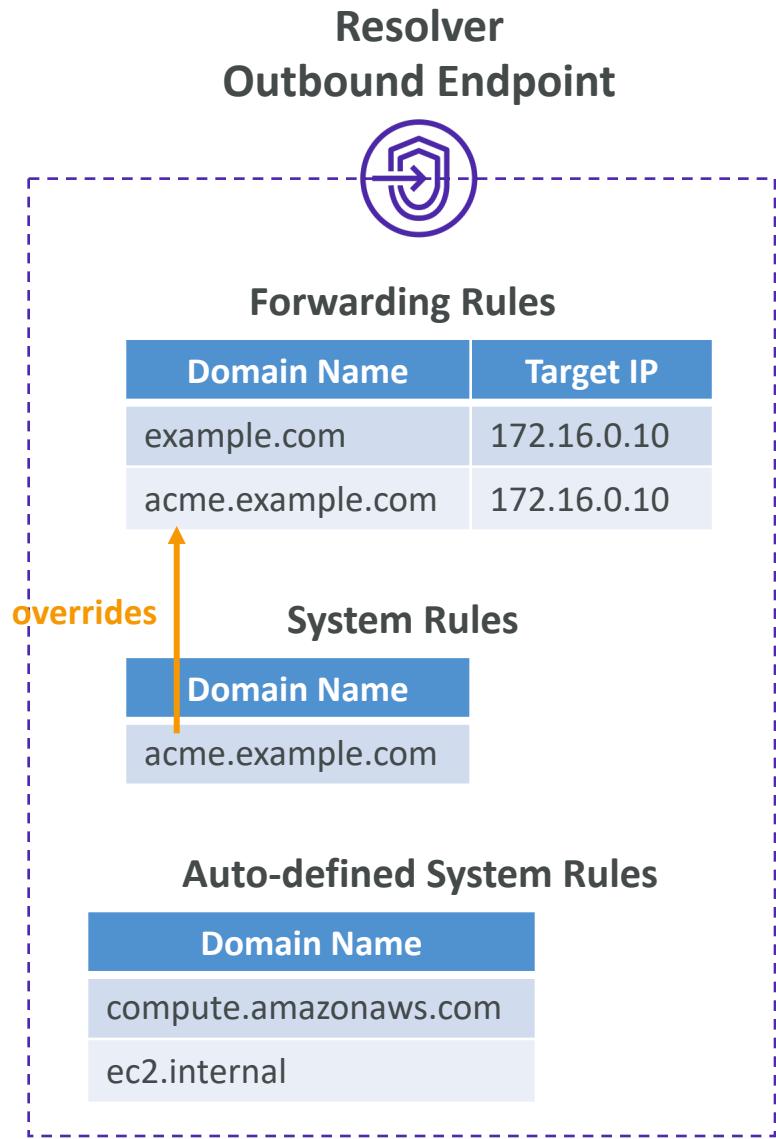


# Route 53 – Resolver Outbound Endpoints



# Route 53 – Resolver Rules

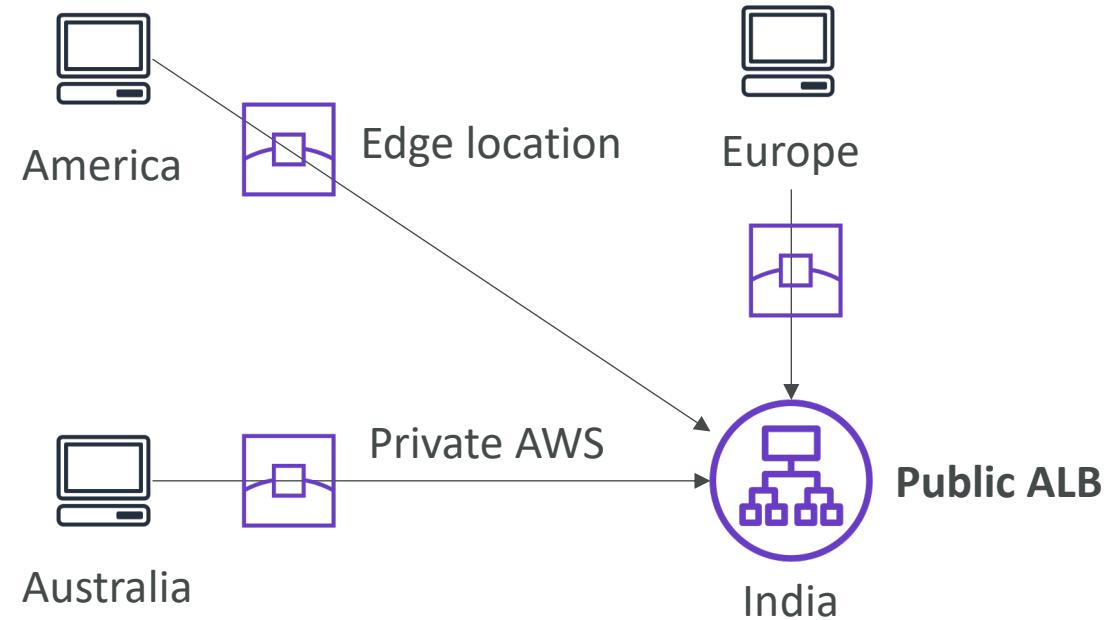
- Control which DNS queries are forwarded to DNS Resolvers on your network
- **Conditional Forwarding Rules (Forwarding Rules)**
  - Forward DNS queries for a specified domain and all its subdomains to target IP addresses
- **System Rules**
  - Selectively overriding the behavior defined in Forwarding Rules (e.g., don't forward DNS queries for a subdomain acme.example.com)
- **Auto-defined System Rules**
  - Defines how DNS queries for selected domains are resolved (e.g., AWS internal domain names, Private Hosted Zones)
  - If multiple rules matched, Route 53 Resolver chooses the most specific match



# AWS Global Accelerator



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



# AWS Global Accelerator

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

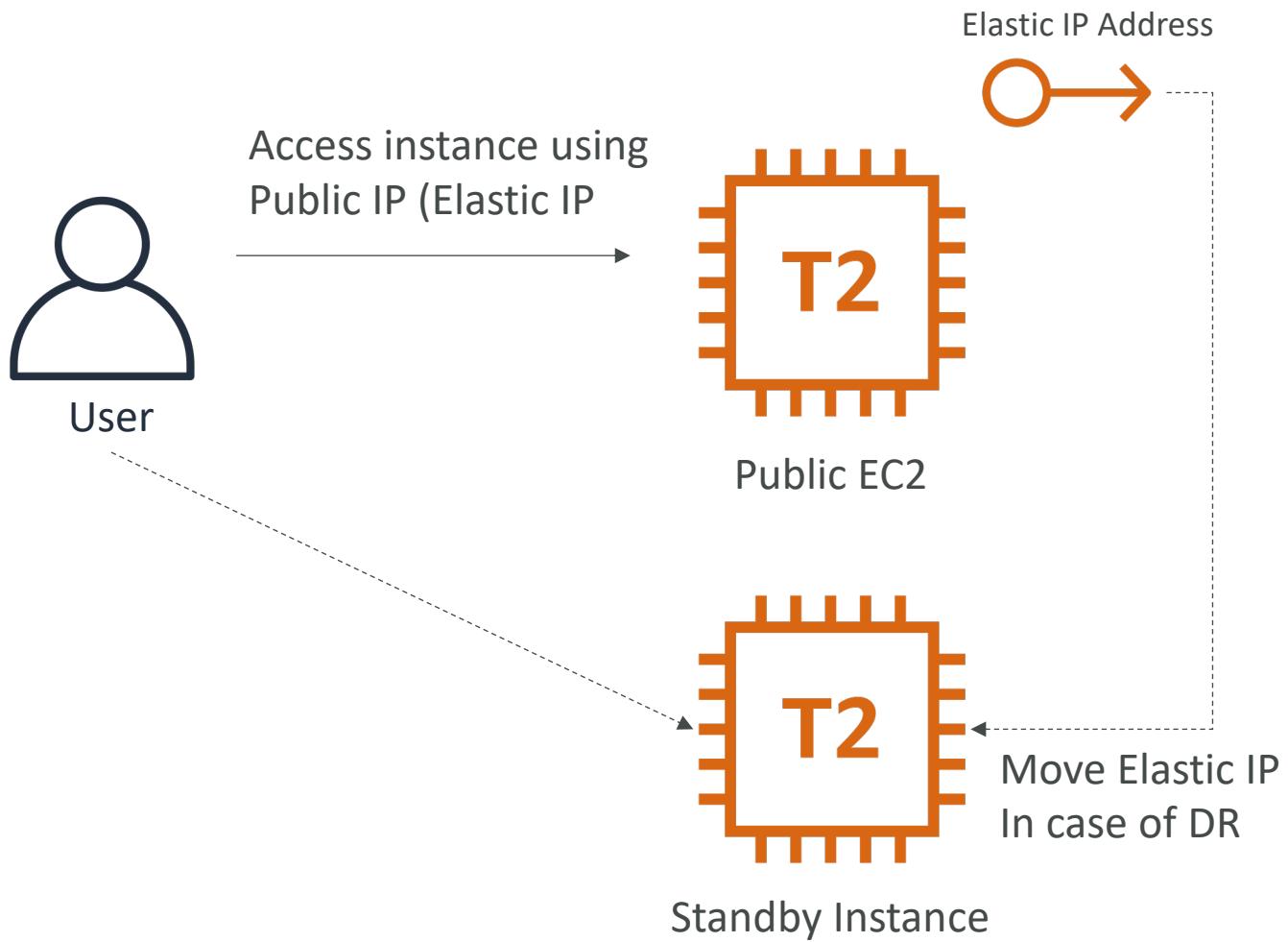
# AWS Global Accelerator vs CloudFront

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

# Solution Architecture Comparisons

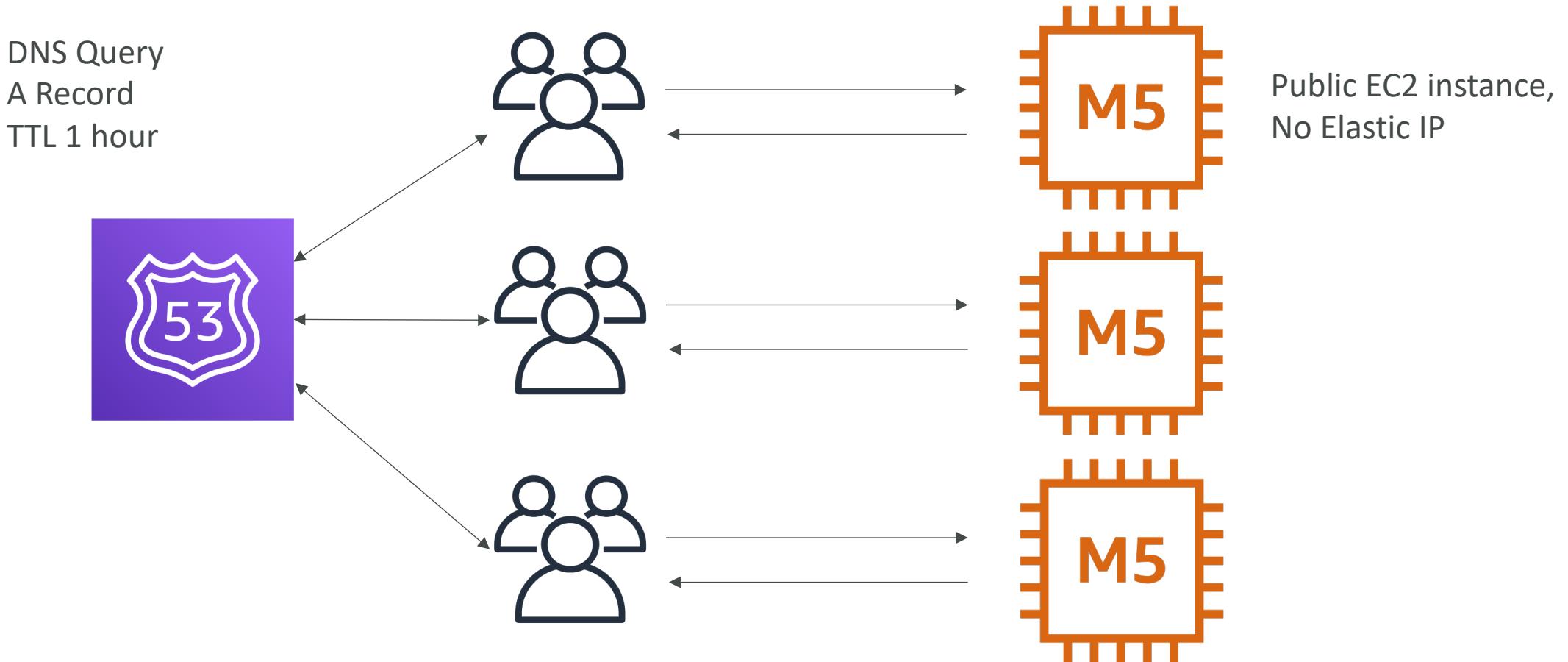
- EC2 on its own with Elastic IP
- EC2 with Route53
- ALB + ASG
- ALB + ECS on EC2
- ALB + ECS on Fargate
- ALB + Lambda
- API Gateway + Lambda
- API Gateway + AWS Service
- API Gateway + HTTP backend (ex: ALB)

# EC2 with Elastic IP

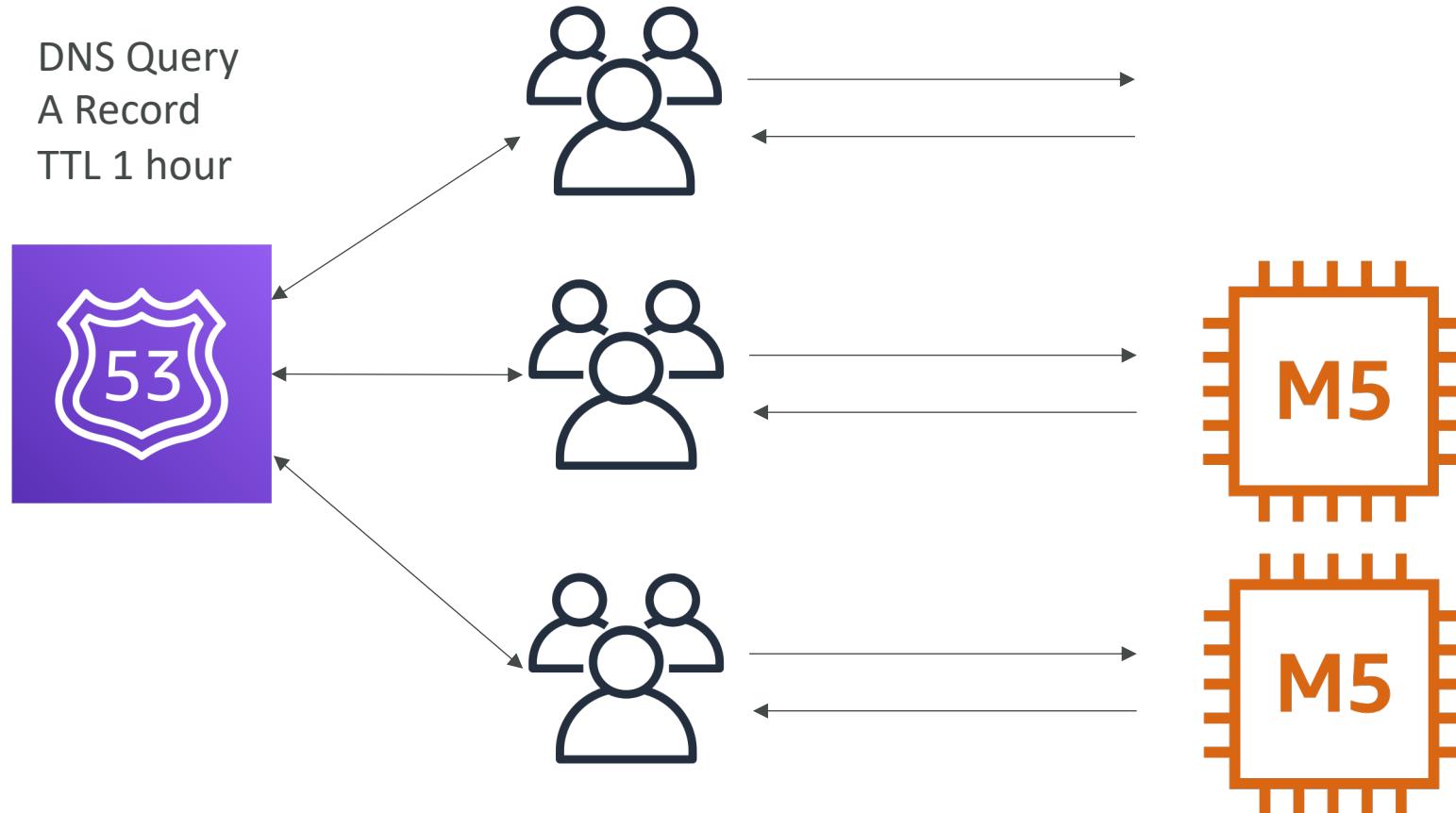


- Quick failover
- The client should not see the change happen
- Helpful if the client needs to resolve by static Public IP address
- Does not scale
- Cheap

# Stateless web app - scaling horizontally

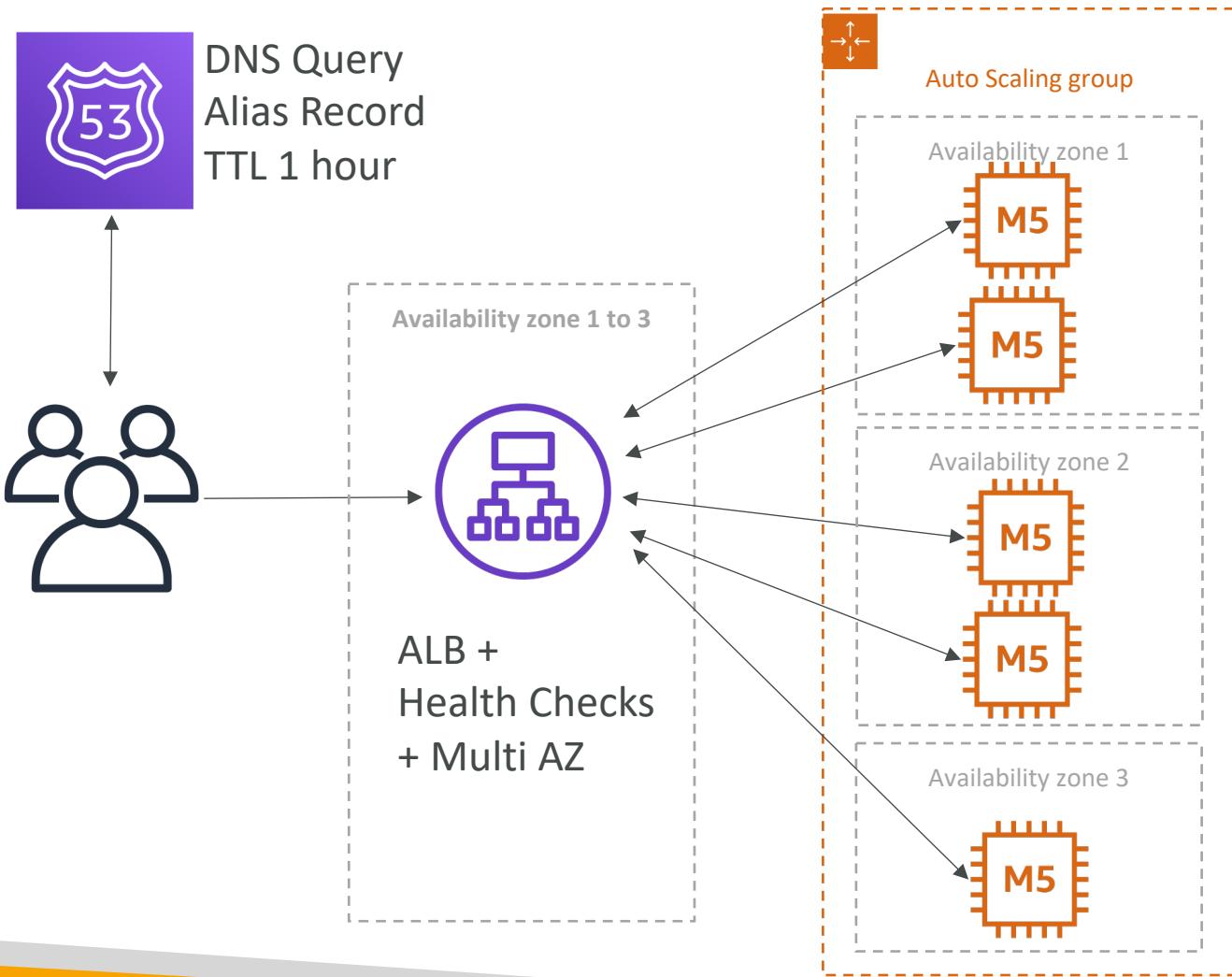


# Stateless web app - scaling horizontally



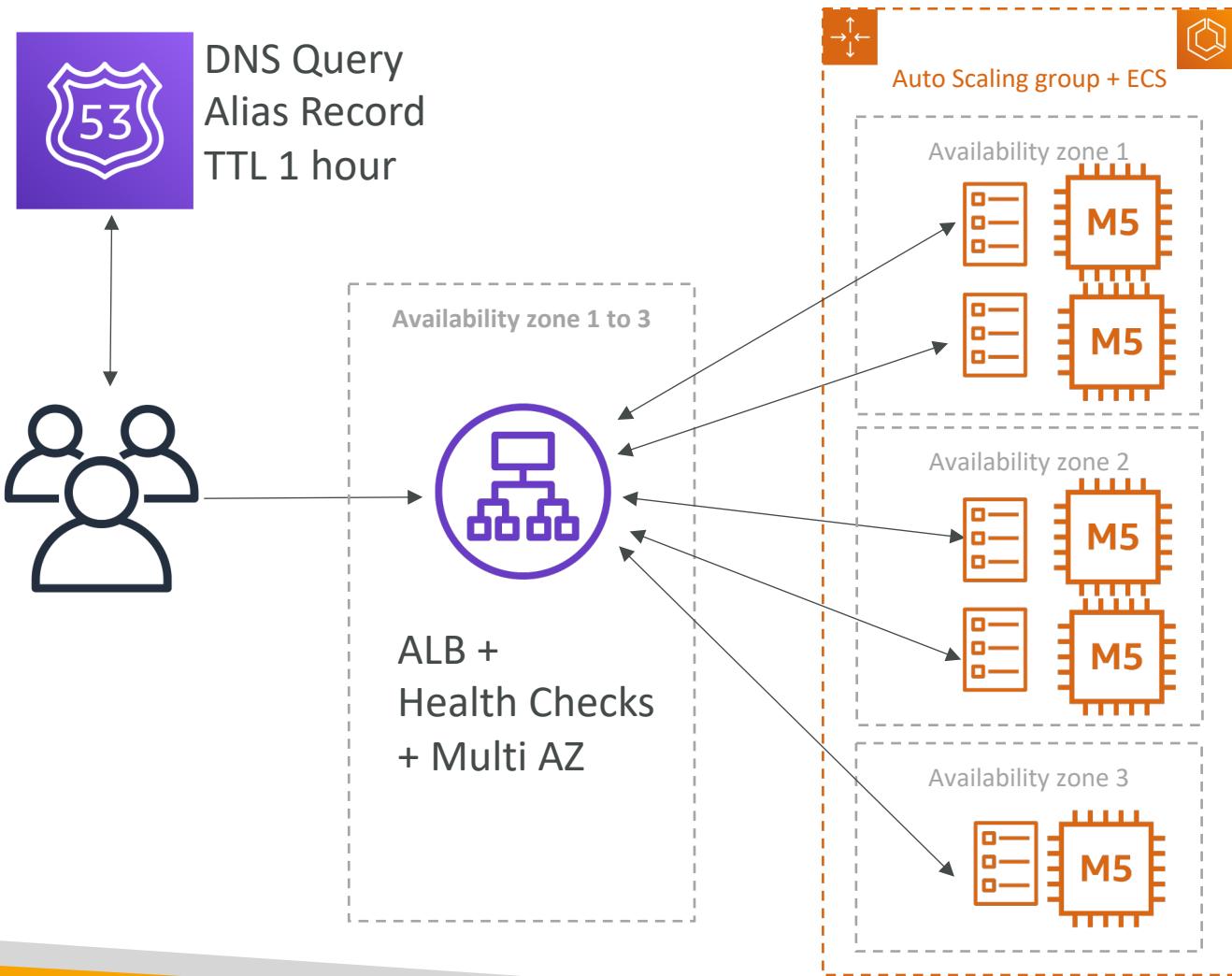
- “DNS-based load balancing”
- Ability to use multiple instances
- Route53 TTL implies client may get outdated information
- Clients must have logic to deal with hostname resolution failures
- Adding an instance may not receive full traffic right away due to DNS TTL

# ALB + ASG



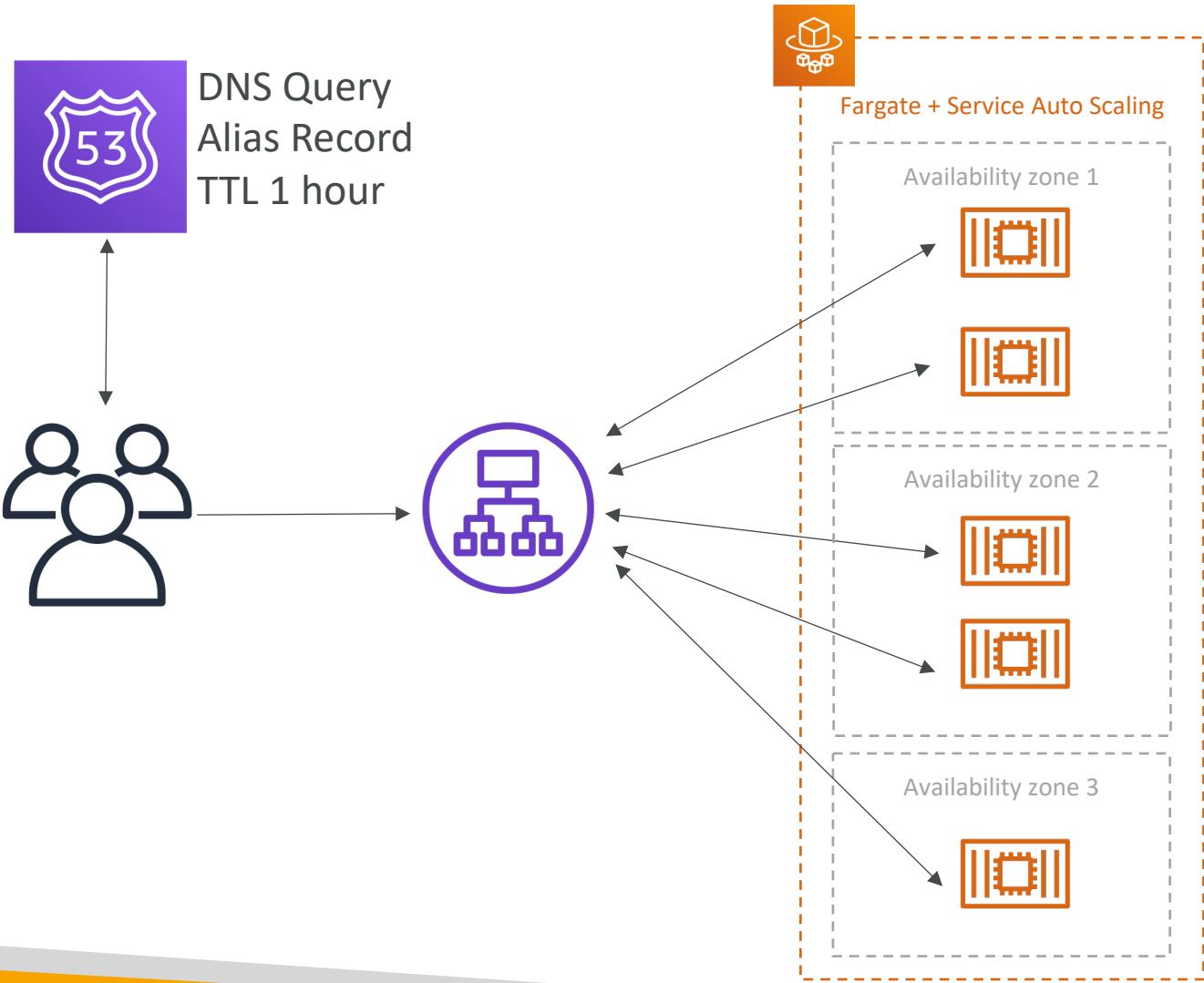
- Scales well, classic architecture
  - New instances are in service right away.
  - Users are not sent to instances that are out-of-service
  - Time to scale is slow (EC2 instance startup + bootstrap) – AMI can help
  - ALB is elastic but can't handle sudden, huge peak of demand (pre-warm)
  - Could lose a few requests if instances are overloaded
  - CloudWatch used for scaling
  - Cross-Zone balancing for even traffic distribution
- 
- Target utilization should be between 40% and 70%

# ALB + ECS on EC2 (backed by ASG)



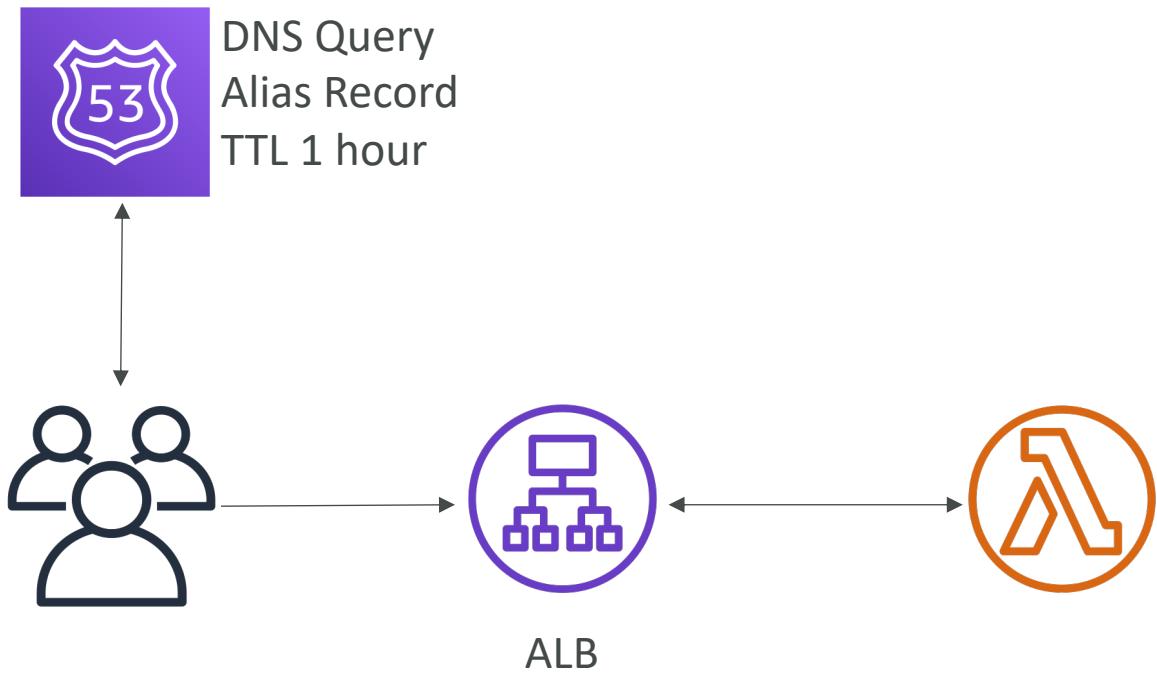
- Same properties as ALB + ASG
- Application is run on Docker
- ASG + ECS allows to have dynamic port mappings
- Tough to orchestrate ECS service auto-scaling + ASG auto-scaling

# ALB + ECS on Fargate



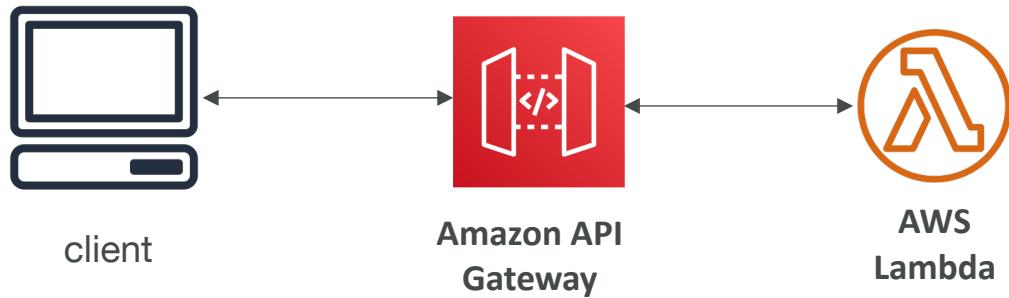
- Application is run on Docker
- Service Auto Scaling is easy
- Time to be in-service is quick (no need to launch an EC2 instance in advance)
- Still limited by the ALB in case of sudden peaks
- “serverless” application tier
- “managed” load balancer

# ALB + Lambda



- Limited to Lambda's runtimes
- Seamless scaling thanks to Lambda
- Simple way to expose Lambda functions as HTTP/S without all the features from API Gateway
- Can combine with WAF (Web Application Firewall)
- Good for hybrid microservices
- Example: use ECS for some requests, use Lambda for others

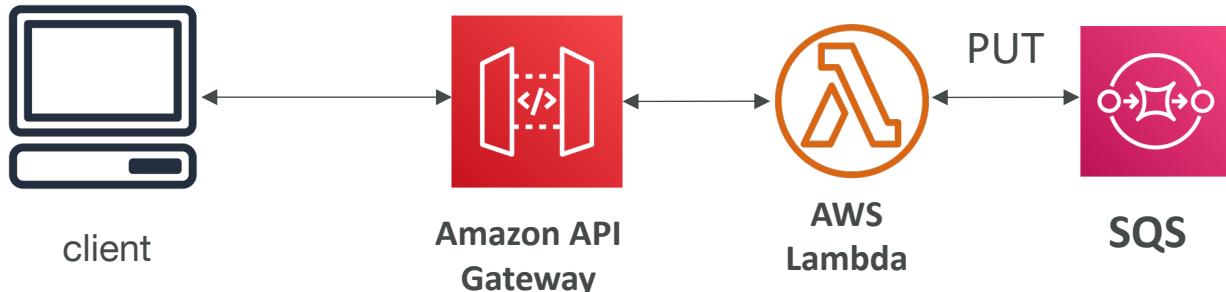
# API Gateway + Lambda



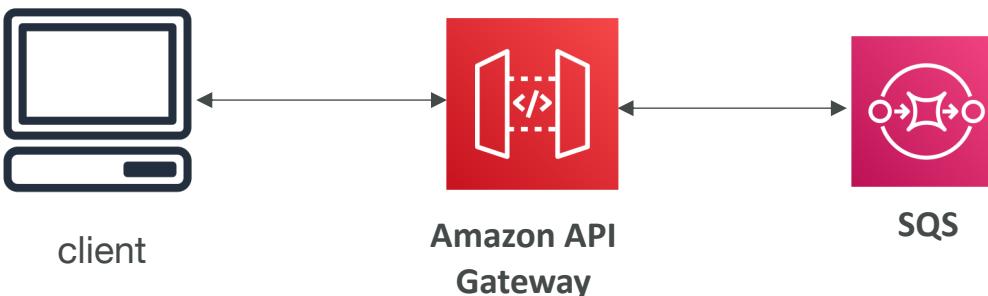
- Pay per request, seamless scaling, fully serverless
- Soft limits: 10000/s API Gateway, 1000 concurrent Lambda
- API Gateway features: authentication, rate limiting, caching, etc...
- Lambda Cold Start time may increase latency for some requests
- Fully integrated with X-Ray

# API Gateway + AWS Service (as a proxy)

OK

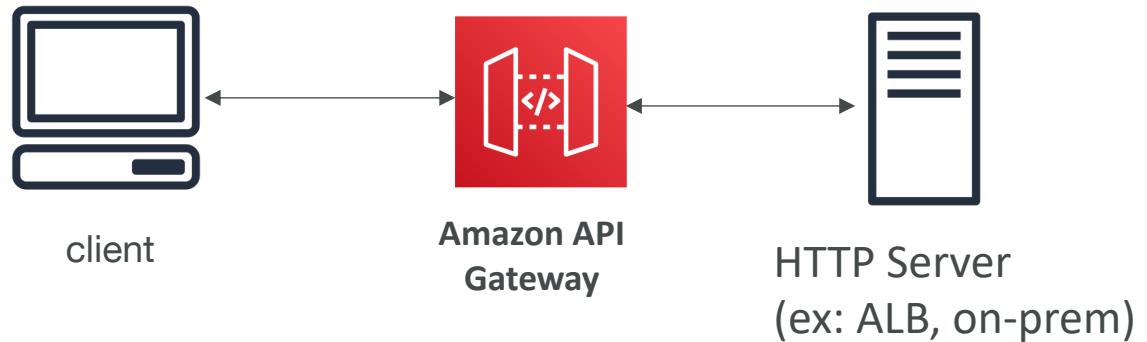


BETTER



- Lower latency, cheaper
- Not using Lambda concurrent capacity, no custom code
- Expose AWS APIs securely through API Gateway
- SQS, SNS, Step Functions...
- Remember API Gateway has a payload limit of 10 MB (can be a problem for S3 proxy)

# API Gateway + HTTP backend (ex: ALB)



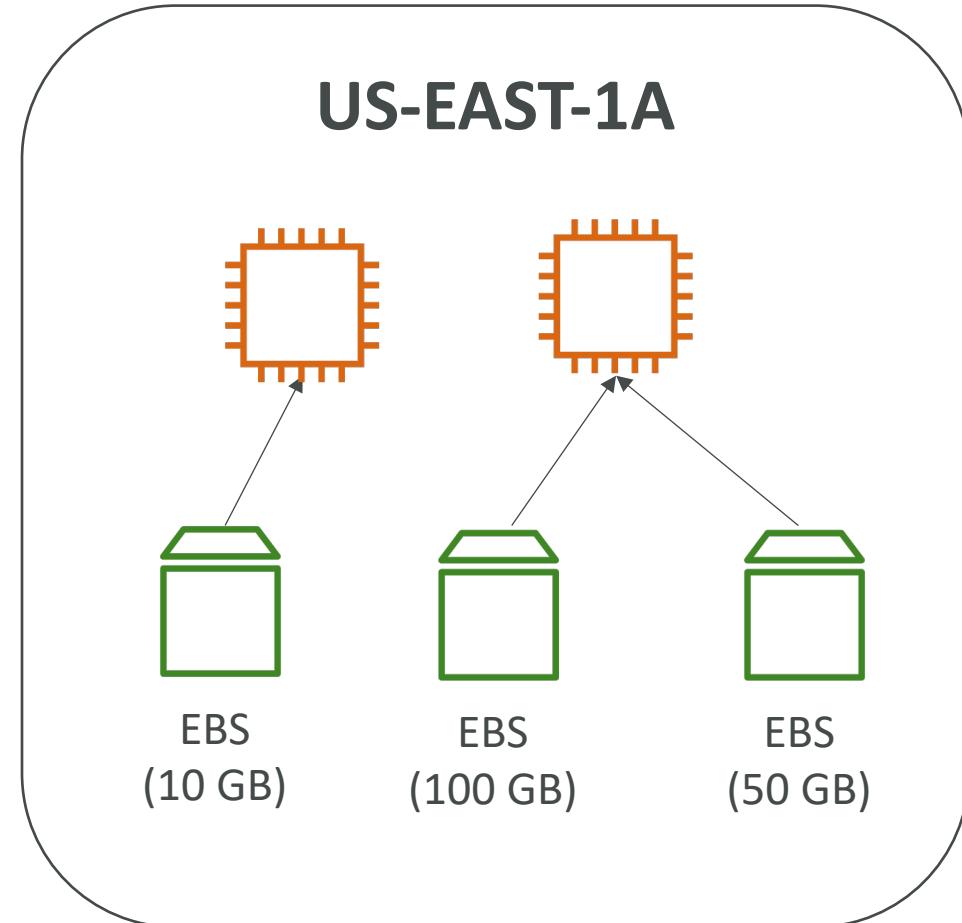
- Use API Gateway features on top of custom HTTP backend (authentication, rate control, API keys, caching...)
- Can connect to...
  - on-premises service
  - Application Load Balancer
  - 3<sup>rd</sup> party HTTP service

# Storage Section

# EBS



- Network drive you attach to ONE instance only
- Linked to a specific availability zone (transfer: snapshot => restore)
- Volumes can be resized
- Make sure you choose an instance type that is EBS optimized to enjoy maximum throughput



# EBS Volume Types

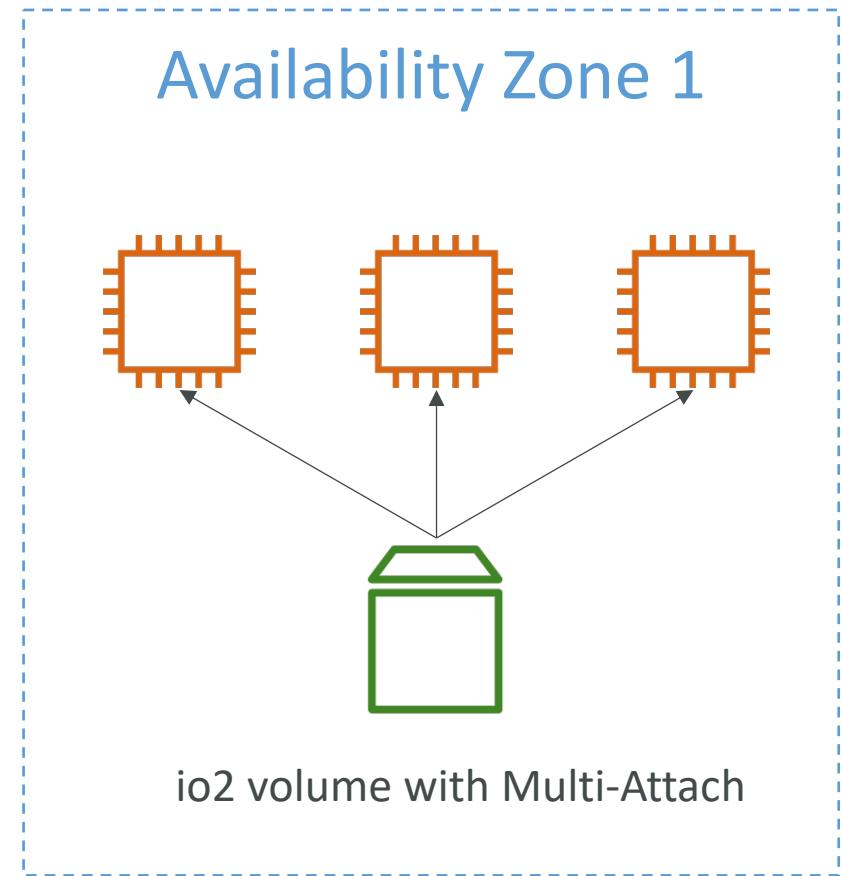
- EBS Volumes come in 6 types
  - [gp2 / gp3 \(SSD\)](#): General purpose SSD volume that balances price and performance for a wide variety of workloads
  - [io1 / io2 \(SSD\) / io2 Block Express](#): Highest-performance SSD volume for mission-critical low-latency or high-throughput workloads
  - [st1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

# EBS Snapshots

- Incremental – only backup changed blocks
- EBS backups use IO, and you shouldn't run them while your application is handling a lot of traffic
- Snapshots will be stored in S3 (but you won't directly see them)
- Not necessary to detach volume to do snapshot, but recommended
- Can copy snapshots across region (for DR)
- Can make Image (AMI) from Snapshot
- EBS volumes restored by snapshots need to be pre-warmed (use the Fast Snapshot Restore FSR feature or fio/dd command to read the entire volume)
- Snapshots can be automated using Amazon Data Lifecycle Manager

# EBS Multi-Attach – io1/io2 family

- Attach the same EBS volume to multiple EC2 instances in the same AZ
- Each instance has full read & write permissions to the volume
- Use case:
  - Achieve higher application availability in clustered Linux applications (ex:Teradata)
  - Applications must manage concurrent write operations
- Must use a file system that's cluster-aware (not XFS, EX4, etc...)



# Local EC2 Instance Store

Very high IOPS

- Physical disk attached to the physical server where your EC2 is
- Very High IOPS (because physical)
- Disks up to 7.5 TiB (can change over time), striped to reach 60 TiB (can change over time...)
- Block Storage (just like EBS)
- Cannot be increased in size
- Risk of data loss if hardware fails

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

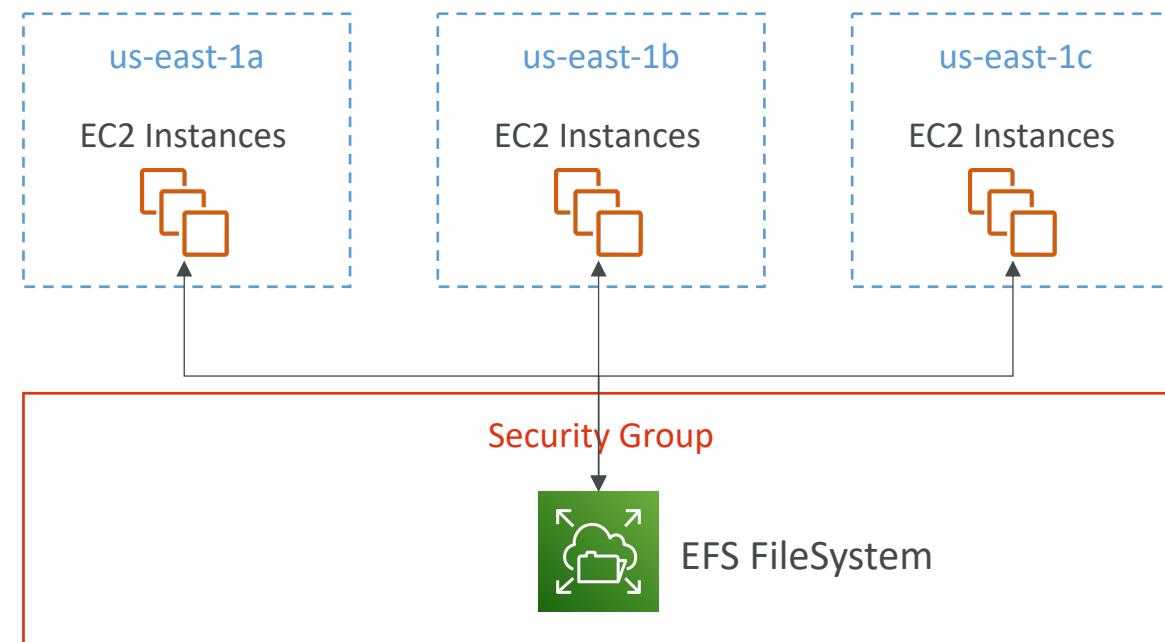
# Instance Store vs EBS

- Instance store is physically attached to the machine (ephemeral storage)
- EBS is a network drive (persistent)
- Pros:
  - Better I/O performance (EBS gp2 has a max IOPS of 16000, io1 of 64000, io2 Block Express of 256000)
  - Good for buffer / cache / scratch data / temporary content
  - Data survives reboots
- Cons:
  - On stop or termination, the instance store is lost
  - You can't resize the instance store
  - Backups must be operated by the user

# EFS – Elastic File System



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



# EFS – Elastic File System

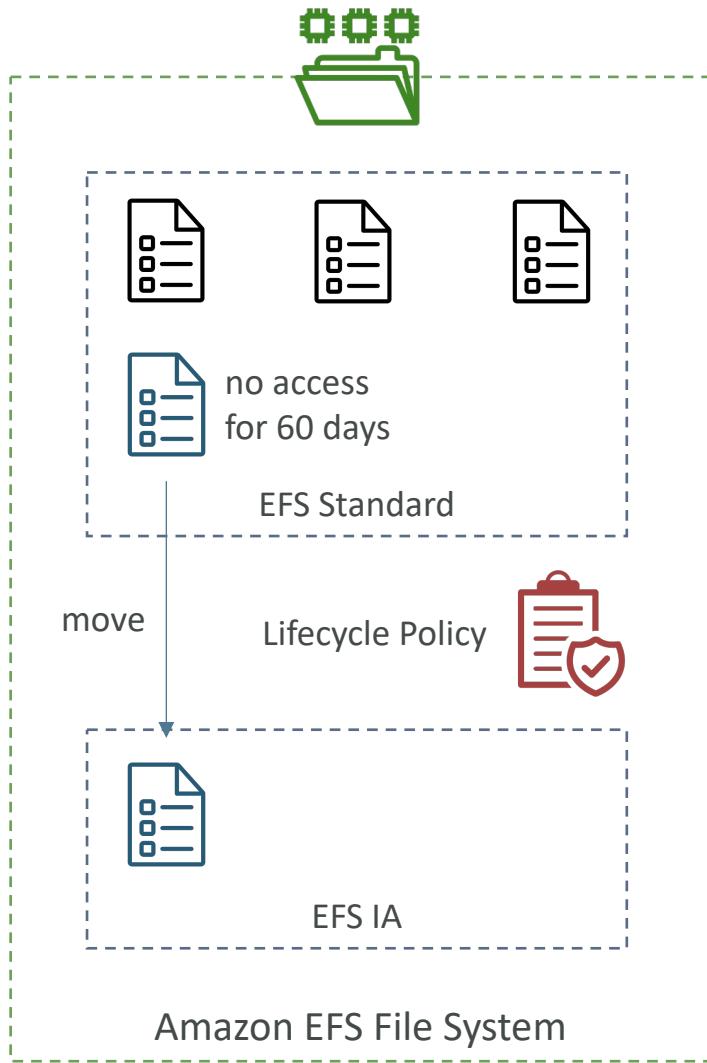
- Use cases: content management, web serving, data sharing, WordPress
- Compatible with Linux based AMI (not Windows), **POSIX**-compliant
- Uses NFSv4.1 protocol
- Uses security group to control access to EFS
- Encryption at rest using KMS
- Can only attach to one VPC, create one ENI (mount target)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

# EFS – Performance & Storage Classes

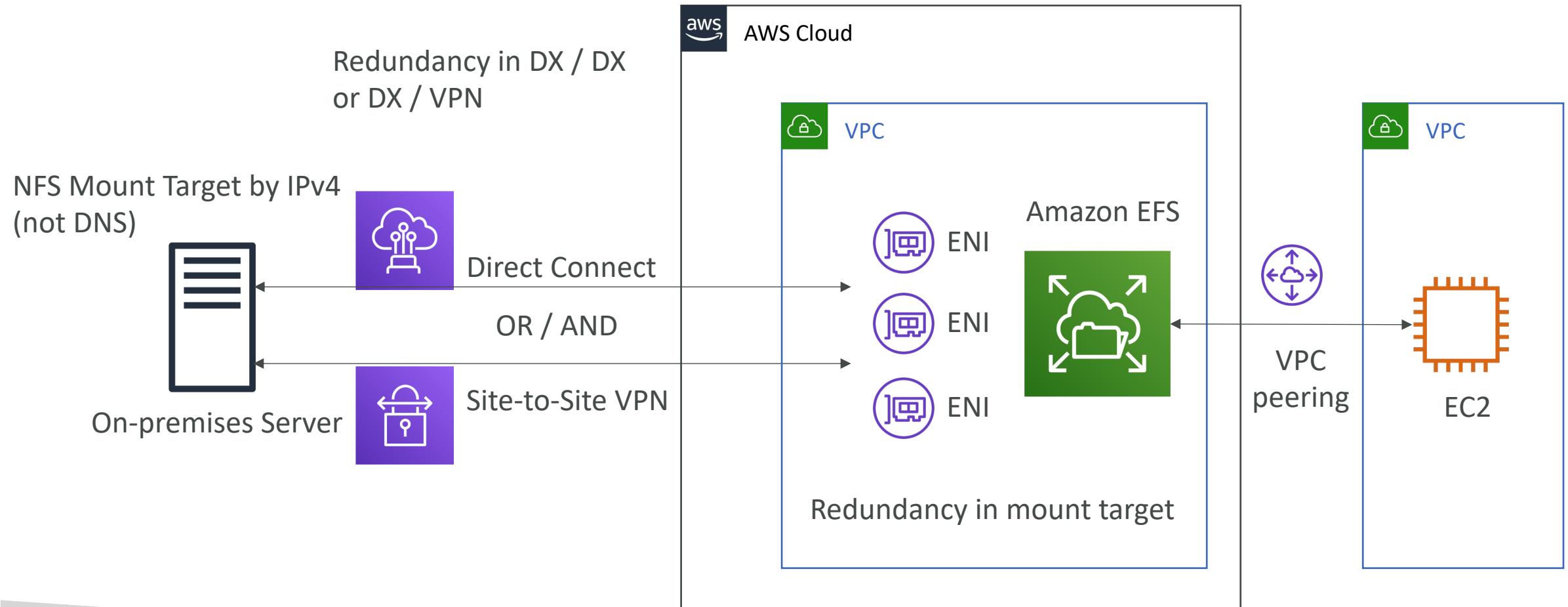
- EFS Scale
  - 1000s of concurrent NFS clients, 10 GB+ /s throughput
  - Grow to Petabyte-scale network file system, automatically
- Performance mode (set at EFS creation time)
  - General purpose (default): latency-sensitive use cases (web server, CMS, etc...)
  - Max I/O – higher latency, throughput, highly parallel (big data, media processing)
- Throughput mode
  - Bursting (1 TB = 50MiB/s + burst of up to 100MiB/s)
  - Provisioned: set your throughput regardless of storage size, ex: 1 GiB/s for 1 TB storage

# EFS – Storage Classes

- Storage Tiers (lifecycle management feature – move file after N days)
  - Standard: for frequently accessed files
  - Infrequent access (EFS-IA): cost to retrieve files, lower price to store. Enable EFS-IA with a Lifecycle Policy
- Availability and durability
  - Regional: Multi-AZ, great for prod
  - One Zone: One AZ, great for dev, backup enabled by default, compatible with IA (EFS One Zone-IA)

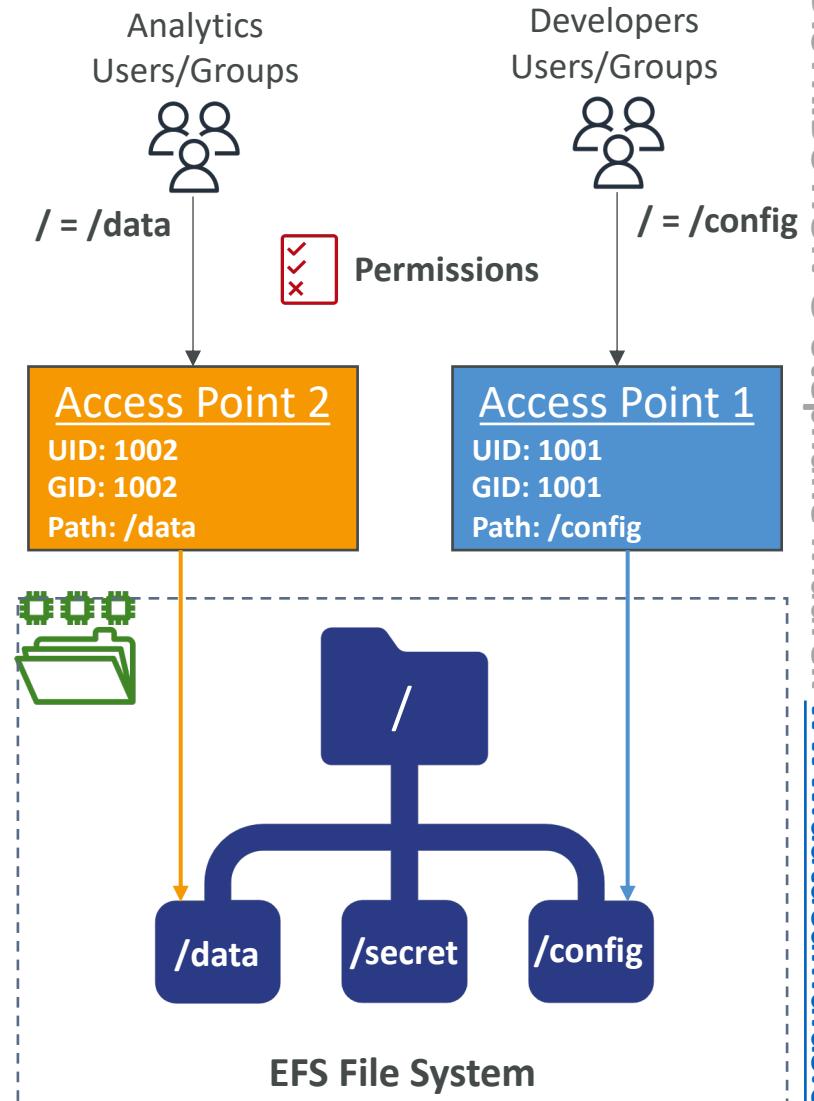


# EFS - On-premises & VPC Peering



# EFS – Access Points

- Easily manage applications access to NFS environments
- Enforce a POSIX user and group to use when accessing the file system
- Restrict access to a directory within the file system and optionally specify a different root directory
- Can restrict access from NFS clients using IAM policies



# EFS – File System Policies

- Resource-based policy to control access to EFS File Systems (same as S3 bucket policy)
- By default, it grants full access to all clients

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": { "AWS": "arn:aws:iam::123456789012:user/Stephane" },  
            "Action": [  
                "elasticfilesystem:ClientMount",  
                "elasticfilesystem:ClientWrite"  
            ],  
            "Condition": {  
                "Bool": { "aws:SecureTransport": "true" }  
            }  
        }  
    ]  
}
```

Grant Read & Write Access to A specific IAM User

# S3 – Overview

- Object storage, serverless, unlimited storage, pay-as-you-go
- Good to store static content (image, video files)
- Access objects by key, no indexing facility
- Not a filesystem, cannot be mounted natively on EC2
- Anti patterns:
  - Lots of small files
  - POSIX file system (use EFS instead), file locks
  - Search features, queries, rapidly changing data
  - Website with dynamic content

# S3 Storage Classes Comparison

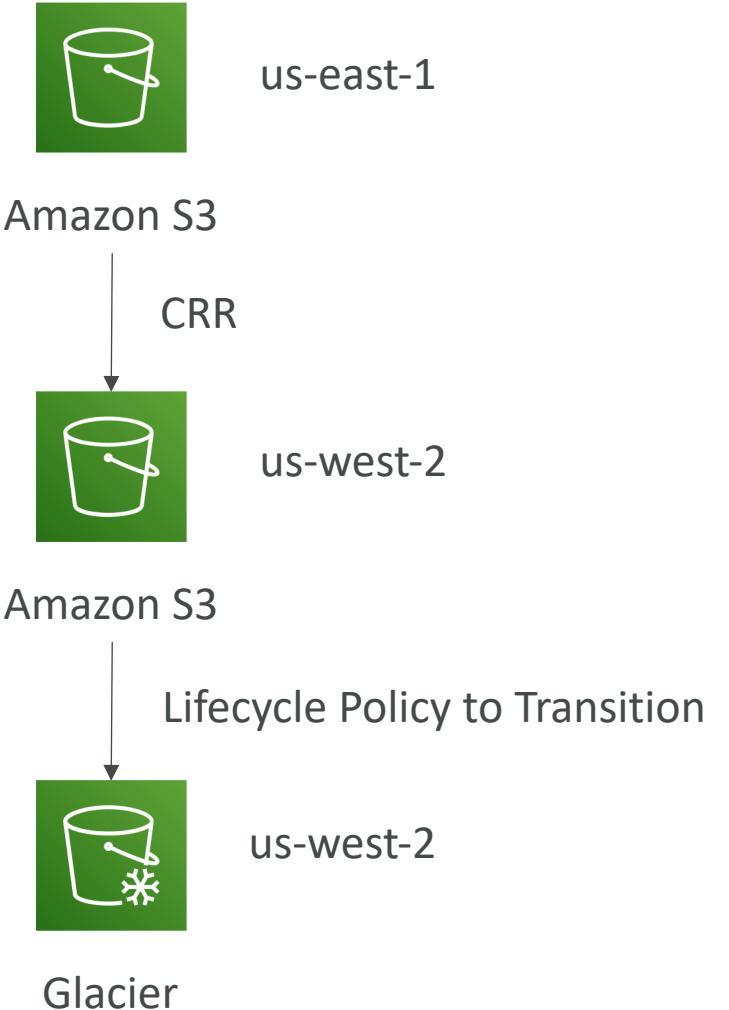
	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Durability	99.999999999% == (11 9's)						
Availability	99.99%	99.9%	99.9%	99.5%	99.9%	99.99%	99.99%
Availability Zones	>= 3	>= 3	>= 3	1	>= 3	>= 3	>= 3
Min. Storage Duration Charge	None	None	30 Days	30 Days	90 Days	90 Days	180 Days
Min. Billable Object Size	None	None	128 KB	128 KB	128 KB	40 KB	40 KB
Retrieval Fee	None	None	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved

- You can transition objects between tiers (or delete) using S3 Lifecycle Policies

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

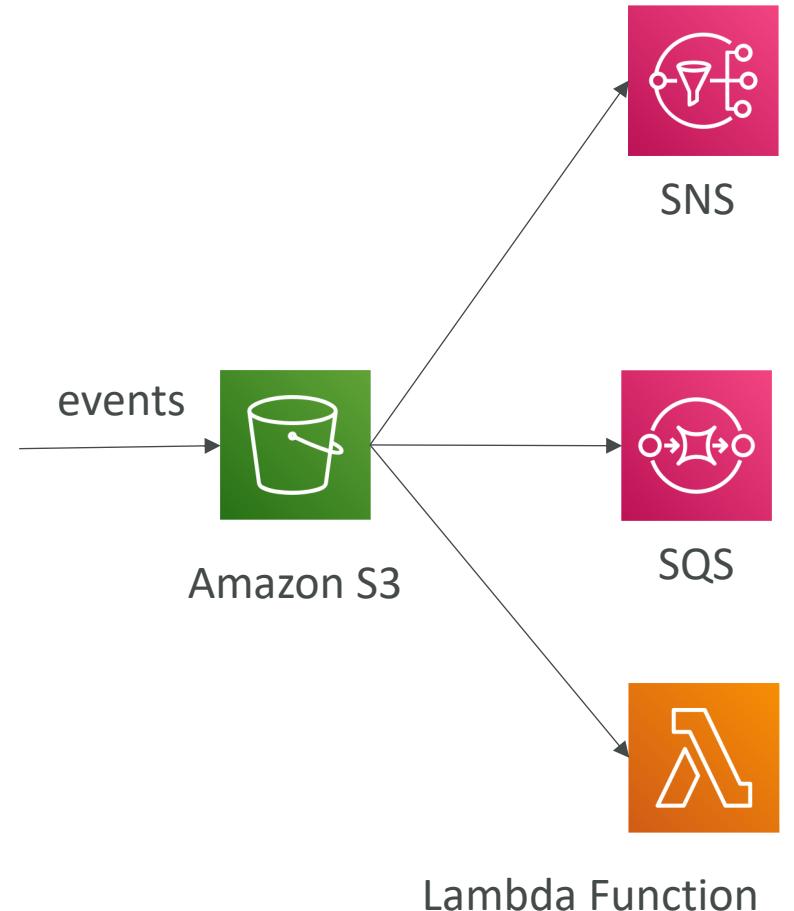
# S3 – Replication

- Cross Region Replication (CRR)
- Same Region Replication (SRR)
- Combine with Lifecycle Policies
  
- Helpful to reduce latency
- Helpful for disaster recovery
- Helpful for security
  
- S3 bucket versioning must be enabled

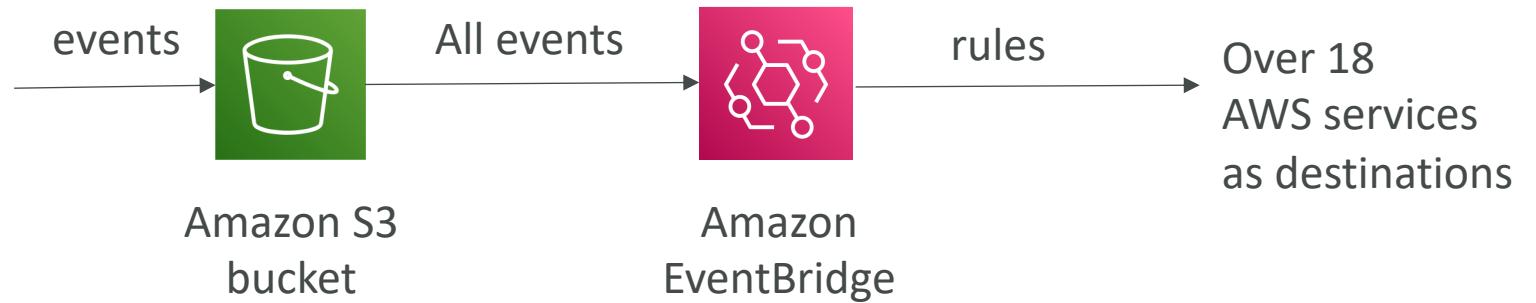


# S3 Event Notifications

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



# S3 Event Notifications with Amazon EventBridge



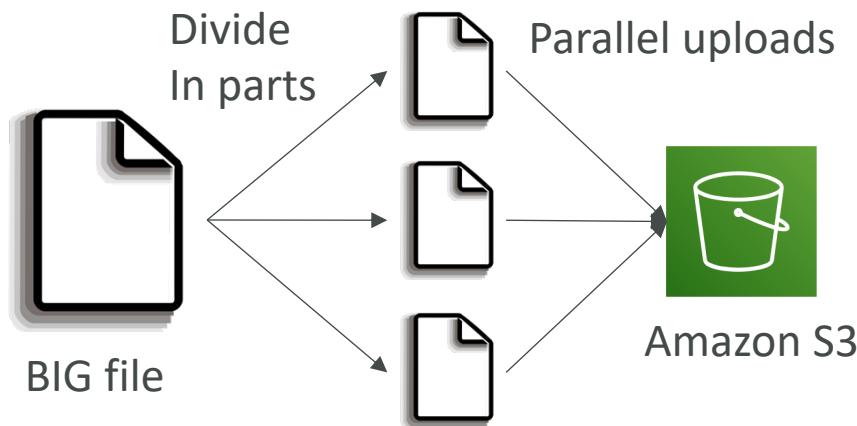
- Advanced filtering options with JSON rules (metadata, object size, name...)
- Multiple Destinations – ex Step Functions, Kinesis Streams / Firehose...
- EventBridge Capabilities – Archive, Replay Events, Reliable delivery

# S3 – Baseline Performance

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

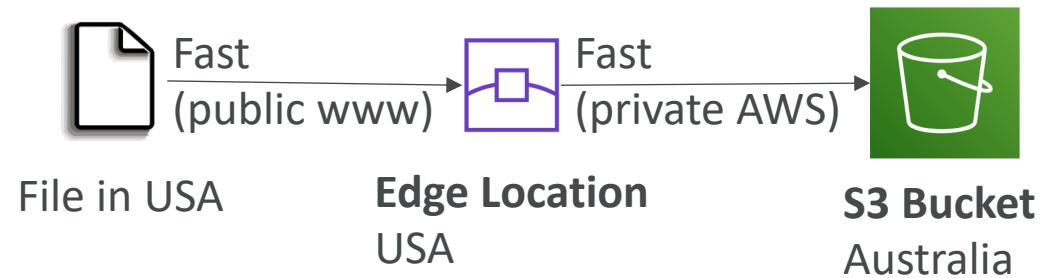
# S3 Performance

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



- S3 Transfer Acceleration

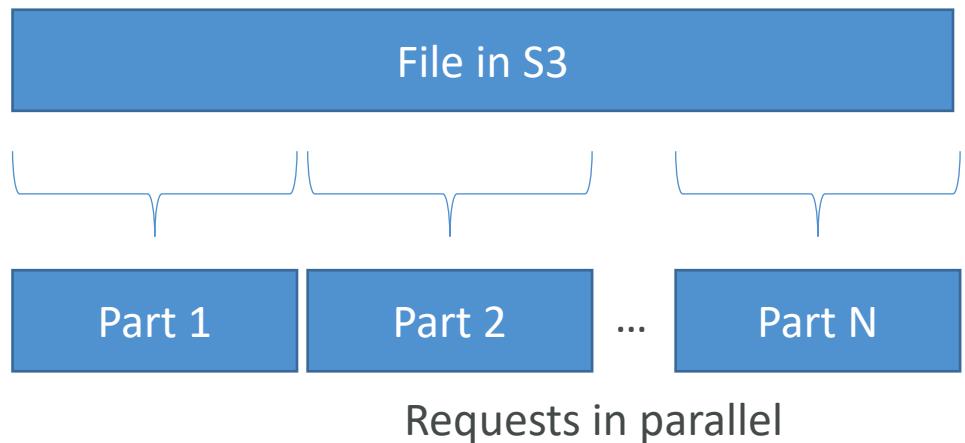
- Increase transfer speed by transferring file to an AWS edge location which will forward the data to the S3 bucket in the target region
- Compatible with multi-part upload



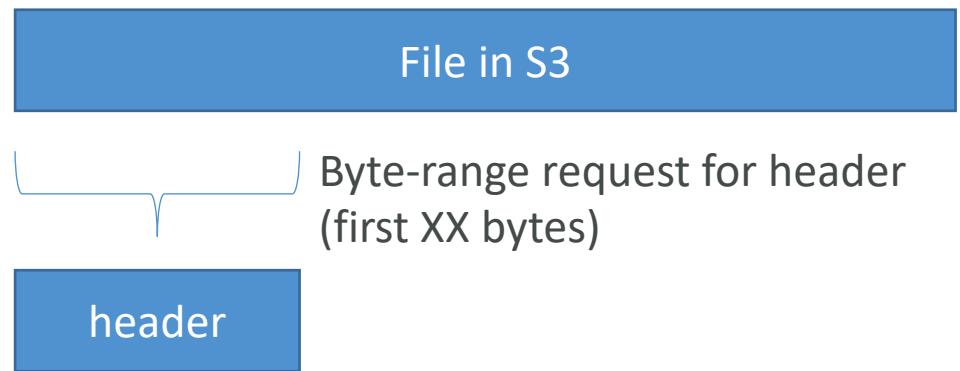
# S3 Performance – S3 Byte-Range Fetches

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

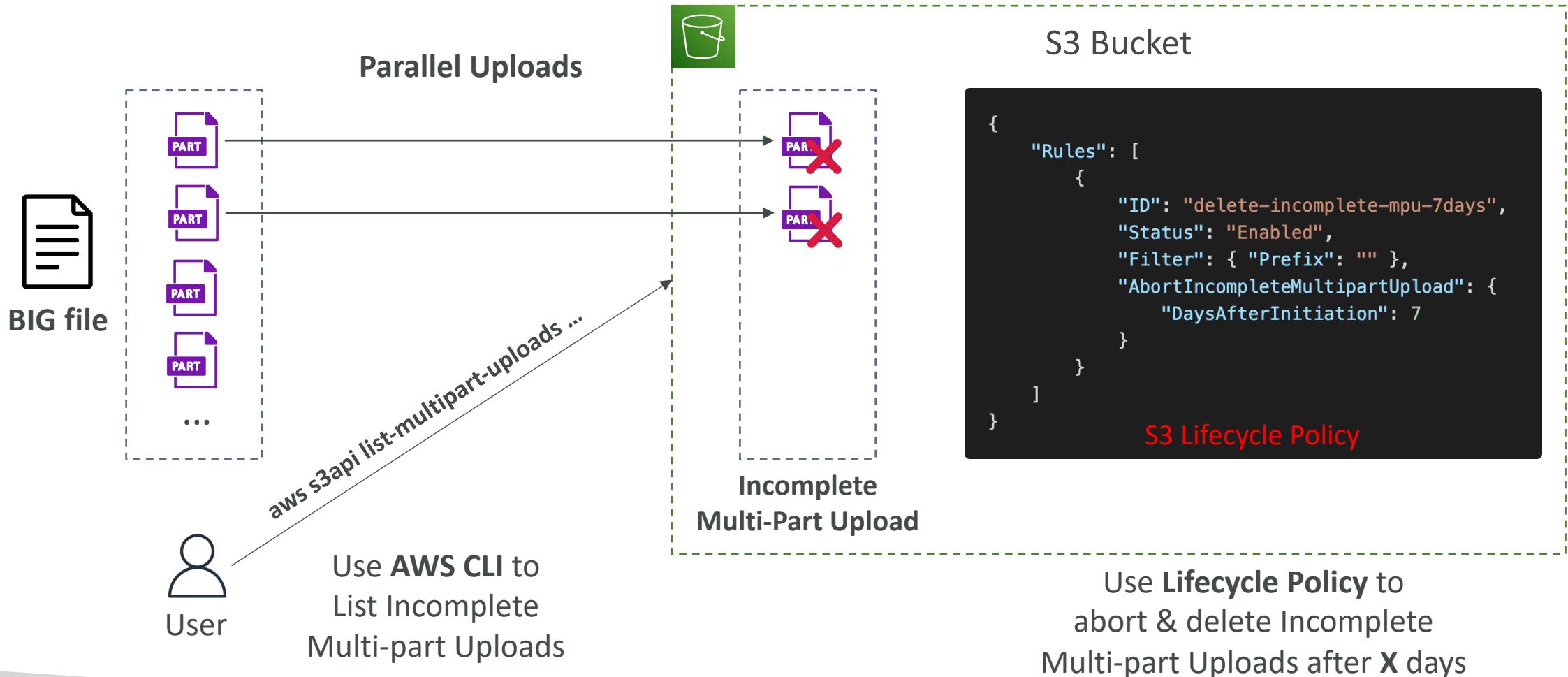
Can be used to speed up downloads



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

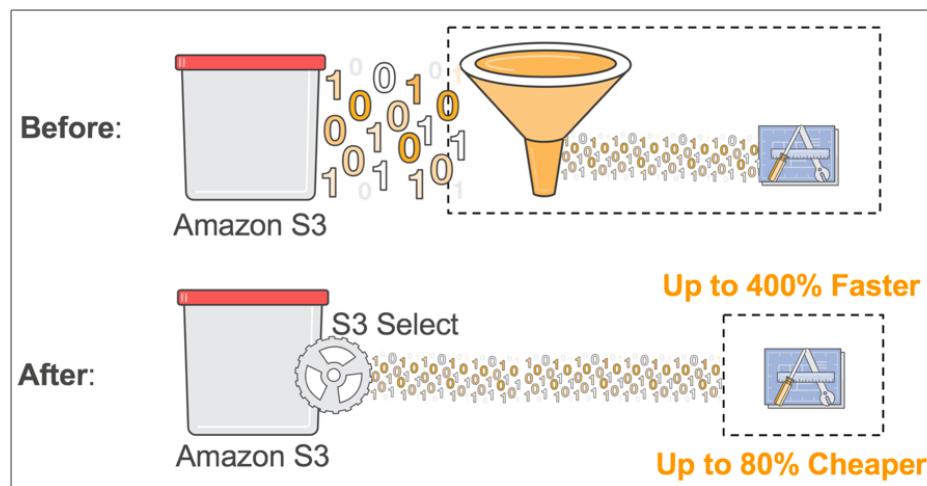


# S3 Multi-Part Upload – Remove Incomplete Parts



# S3 Select & Glacier Select

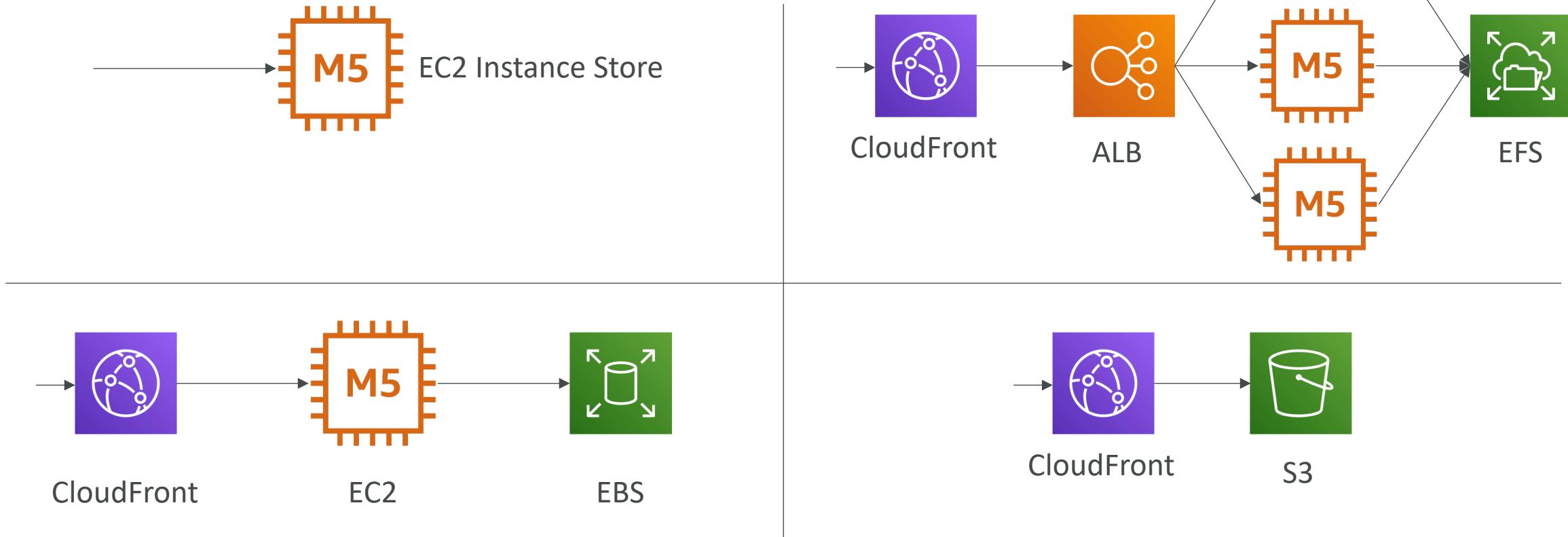
- Retrieve less data using SQL by performing **server side filtering**
- Can filter by rows & columns (simple SQL statements)
- Less network transfer, less CPU cost client-side



<https://aws.amazon.com/blogs/aws/s3-glacier-select/>

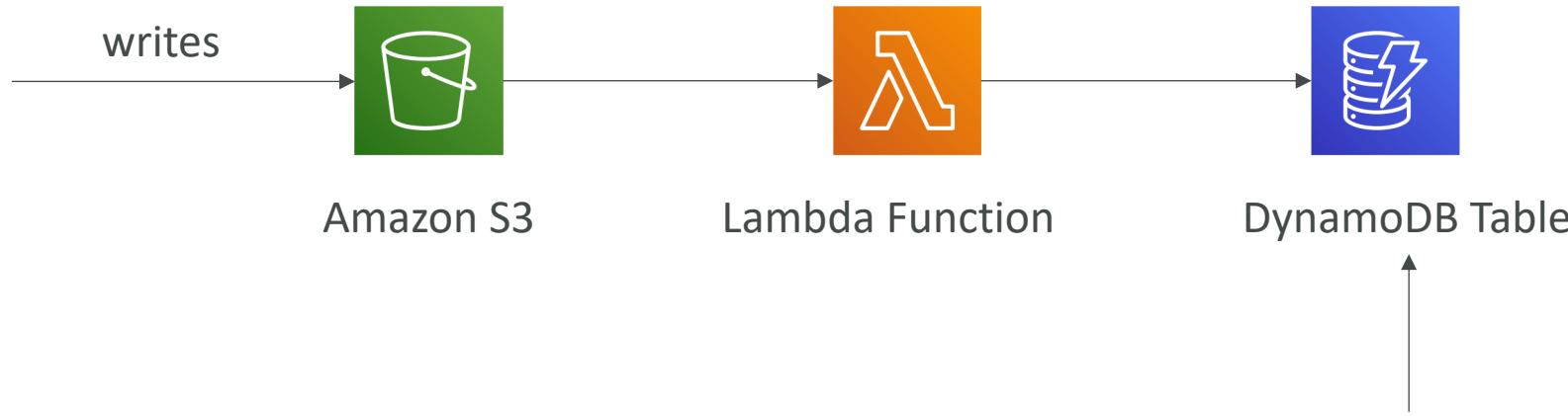


# S3 Solution Architecture Exposing Static Objects



# S3 Solution Architecture

## Indexing objects in DynamoDB

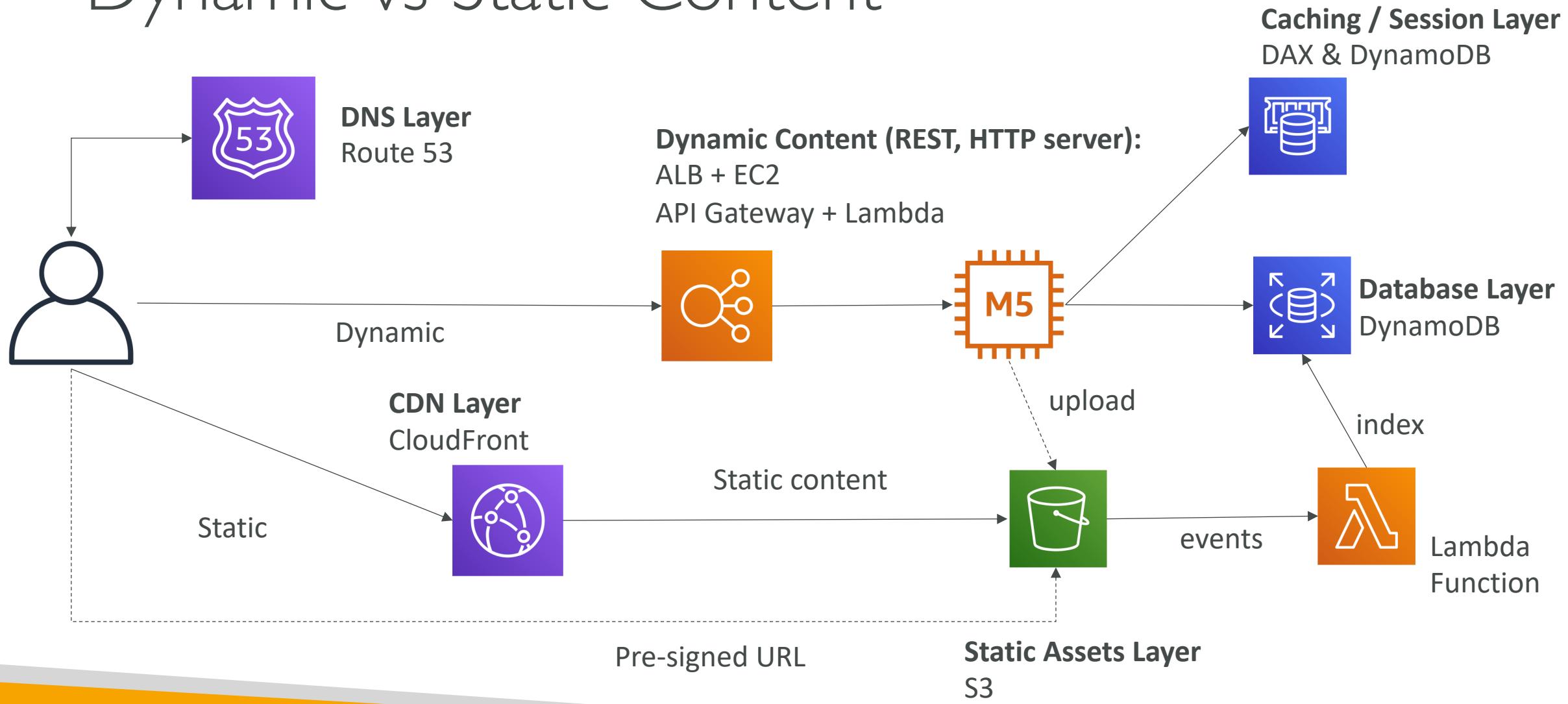


### API for object metadata

- Search by date
- Total storage used by a customer
- List of all objects with certain attributes
- Find all objects uploaded within a date range

# Solution Architecture on AWS

## Dynamic vs Static Content



# Amazon FSx – Overview



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



**FSx for Lustre**



**FSx for  
Windows File  
Server**



**FSx for  
NetApp ONTAP**

# Amazon FSx for Windows (File Server)



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

# Amazon FSx for Lustre



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

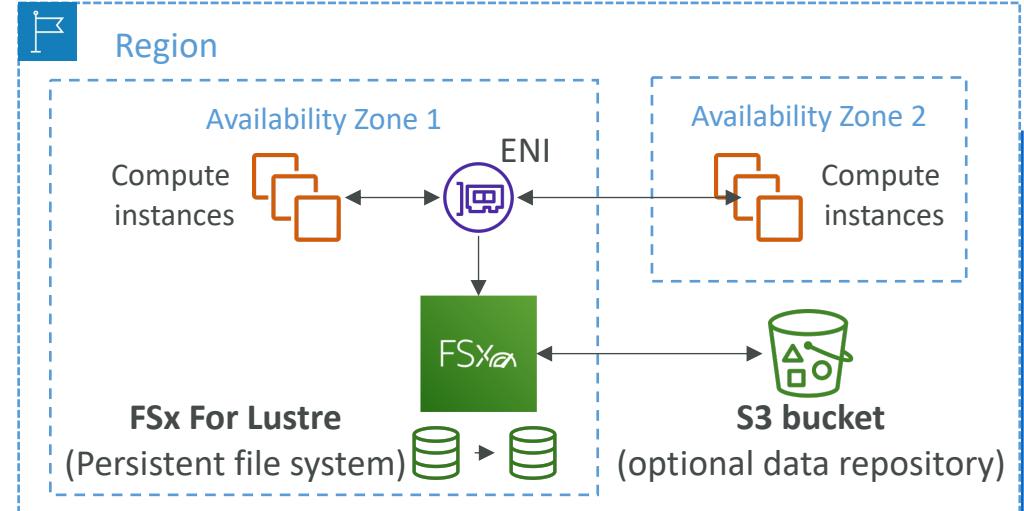
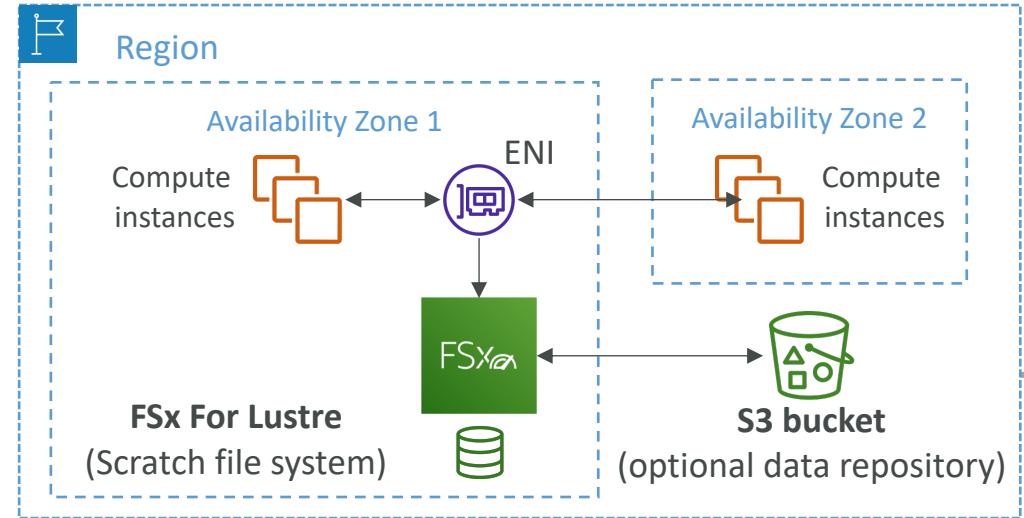
# FSx File System Deployment Options

## • Scratch File System

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

## • Persistent File System

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



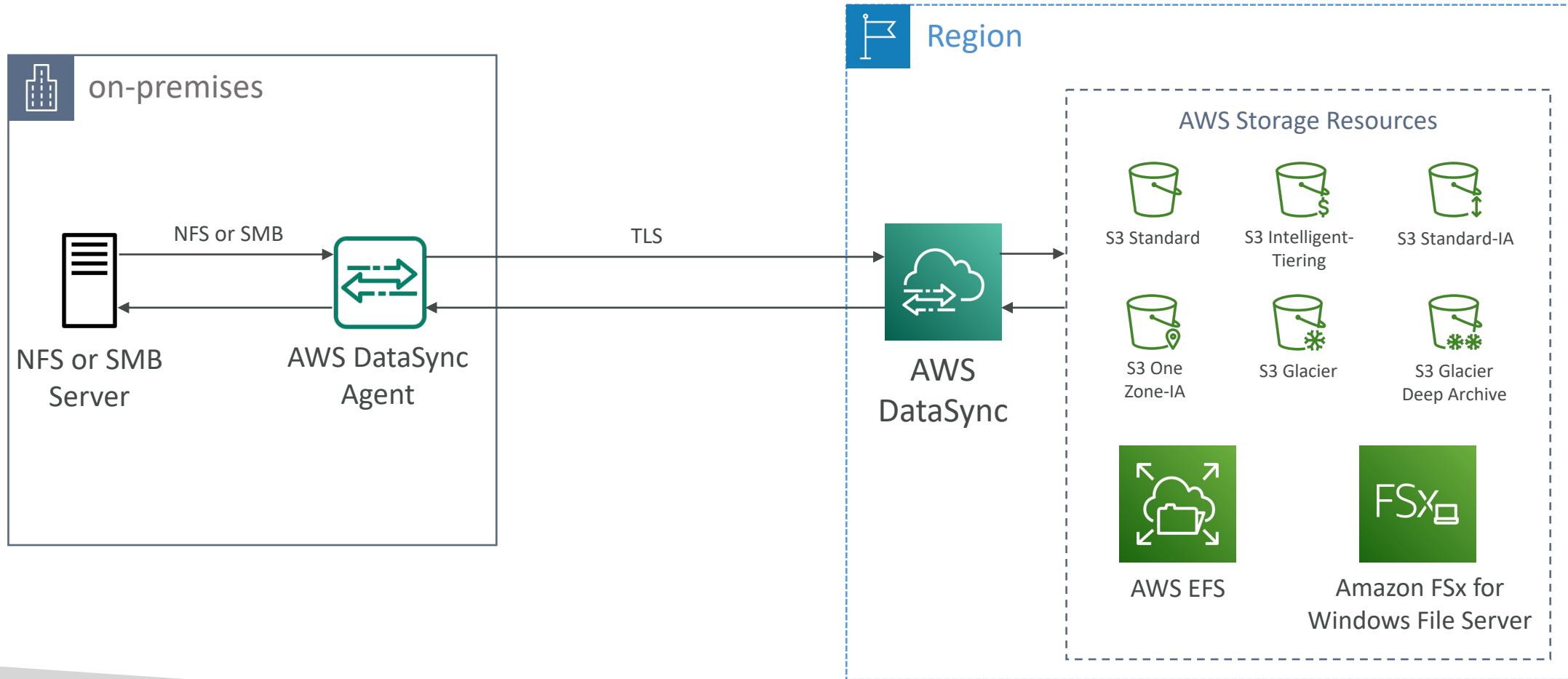


# AWS DataSync

- Move large amount of data from on-premises to AWS
- Can synchronize to: Amazon S3 (any storage classes – including Glacier), Amazon EFS, Amazon FSx for Windows
- Move data from your NAS or file system via NFS or SMB
- Replication tasks can be scheduled hourly, daily, weekly
- Leverage the DataSync agent to connect to your systems
- Can setup a bandwidth limit

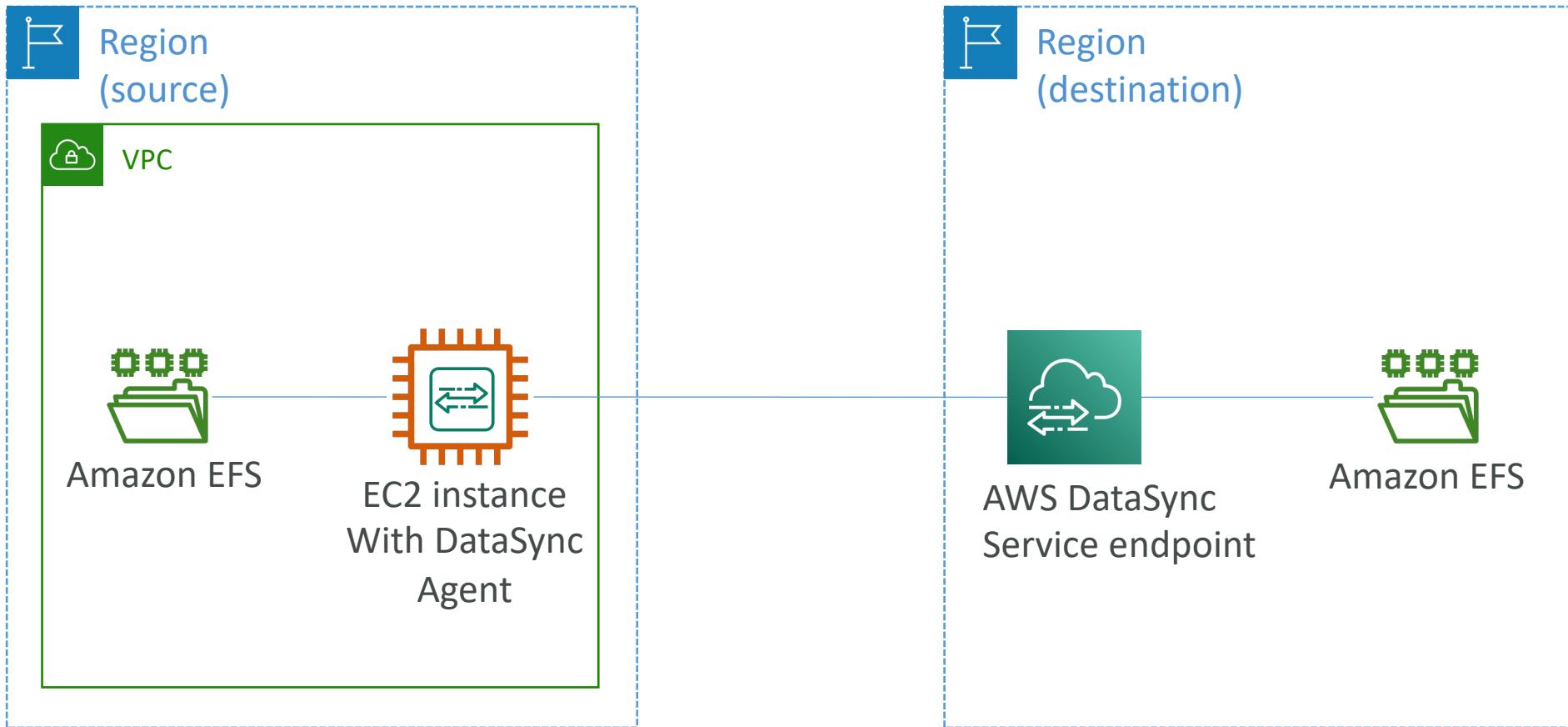
# AWS DataSync

NFS / SMB to AWS (S3, EFS, FSx for Windows)

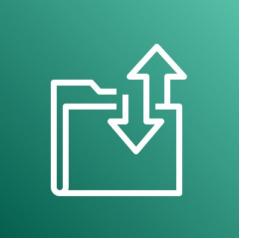


# AWS DataSync

## EFS to EFS

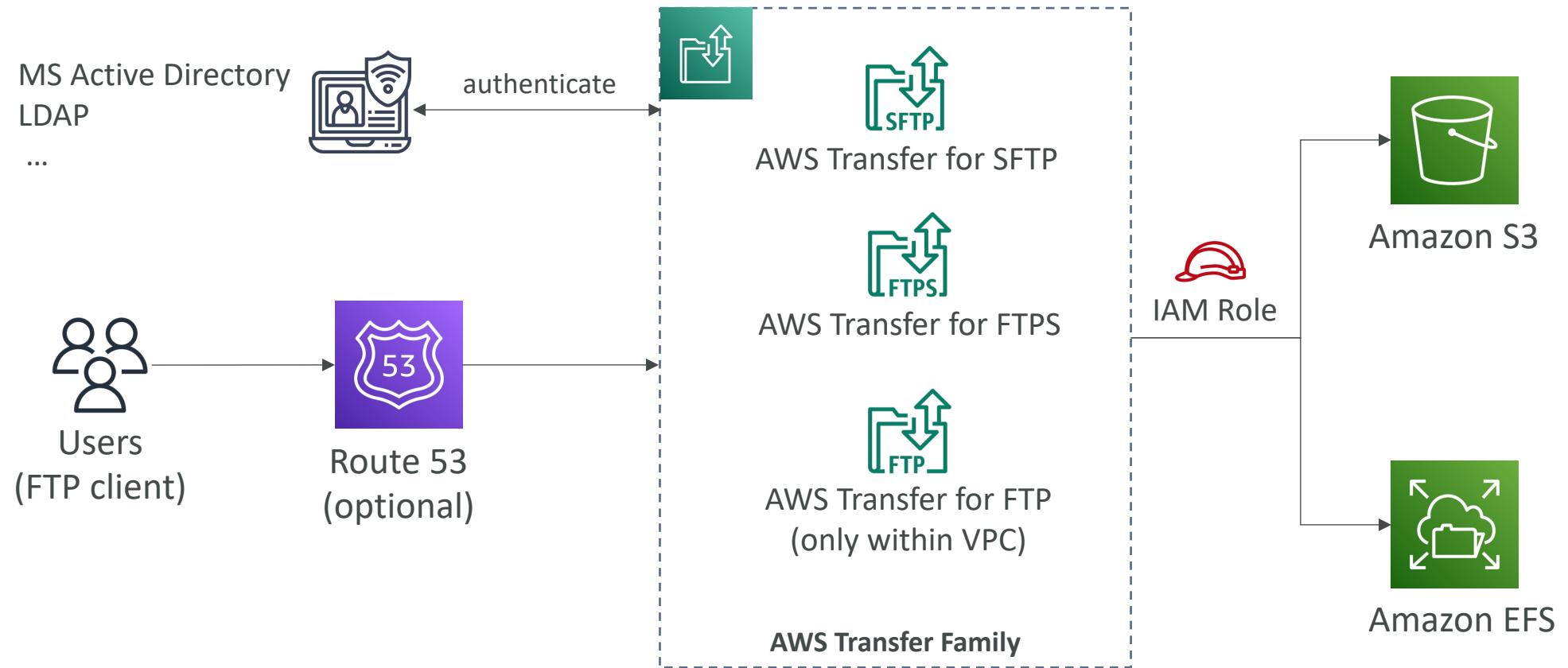


# AWS Transfer Family



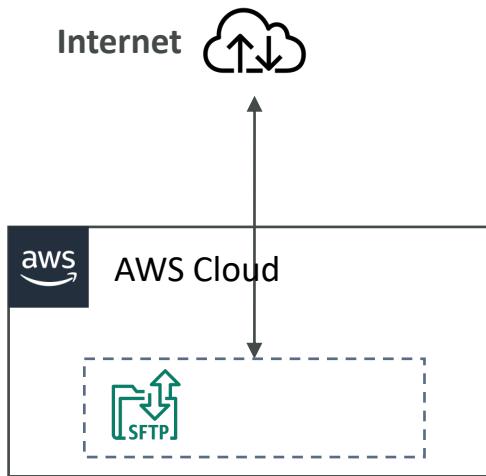
- A fully-managed service for file transfers into and out of Amazon S3 or Amazon EFS using the FTP protocol
- Supported Protocols
  - AWS Transfer for FTP (File Transfer Protocol (FTP))
  - AWS Transfer for FTPS (File Transfer Protocol over SSL (FTPS))
  - AWS Transfer for SFTP (Secure File Transfer Protocol (SFTP))
- Managed infrastructure, Scalable, Reliable, Highly Available (multi-AZ)
- Pay per provisioned endpoint per hour + data transfers in GB
- Store and manage users' credentials within the service
- Integrate with existing authentication systems (Microsoft Active Directory, LDAP, Okta, Amazon Cognito, custom)
- Usage: sharing files, public datasets, CRM, ERP, ...

# AWS Transfer Family



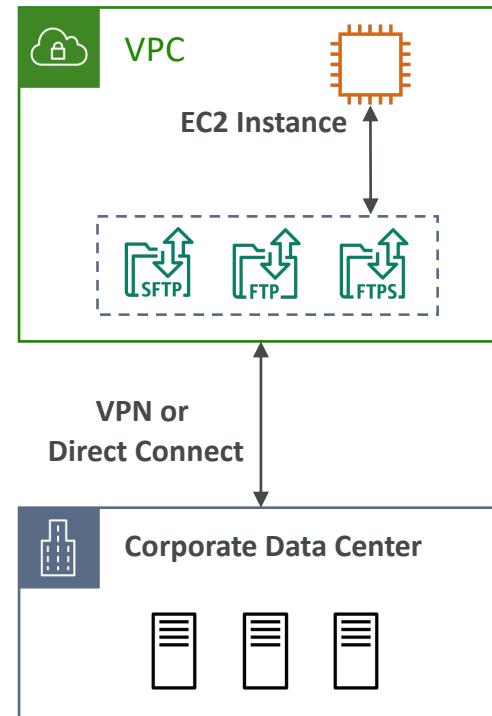
# AWS Transfer Family – Endpoint Types

## Public Endpoint



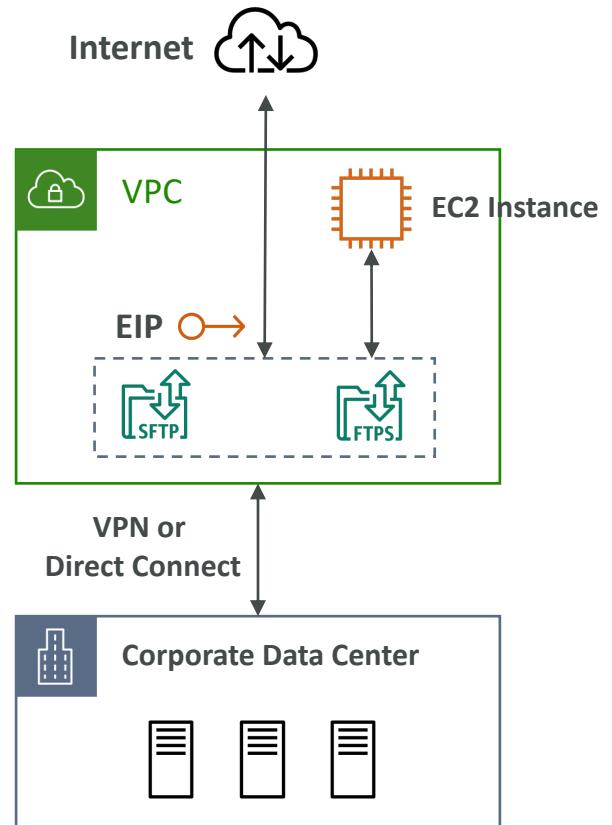
- IPs managed by AWS (subject to change, use DNS names)
- Can't setup allow lists by source IP addresses

## VPC Endpoint with Internal Access



- Static private IPs
- Setup allow lists (SGs & NACLs)

## VPC Endpoint with Internet-facing Access



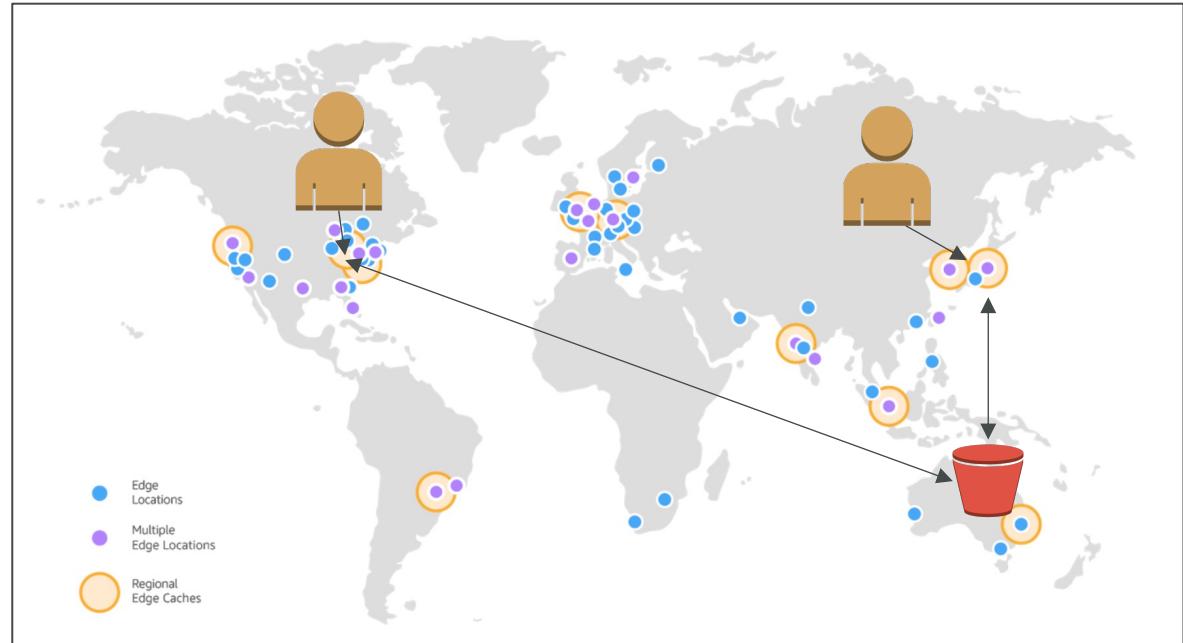
- Static private IPs
- Static public IPs (EIPs)
- Setup Security Groups

# Caching Section

# AWS CloudFront



- Content Delivery Network (CDN)
- Improves read performance, content is cached at the edge
- 300+ Point of Presence globally (edge locations)
- DDoS protection, integration with Shield, AWS Web Application Firewall
- Can expose external HTTPS and can talk to internal HTTPS backends

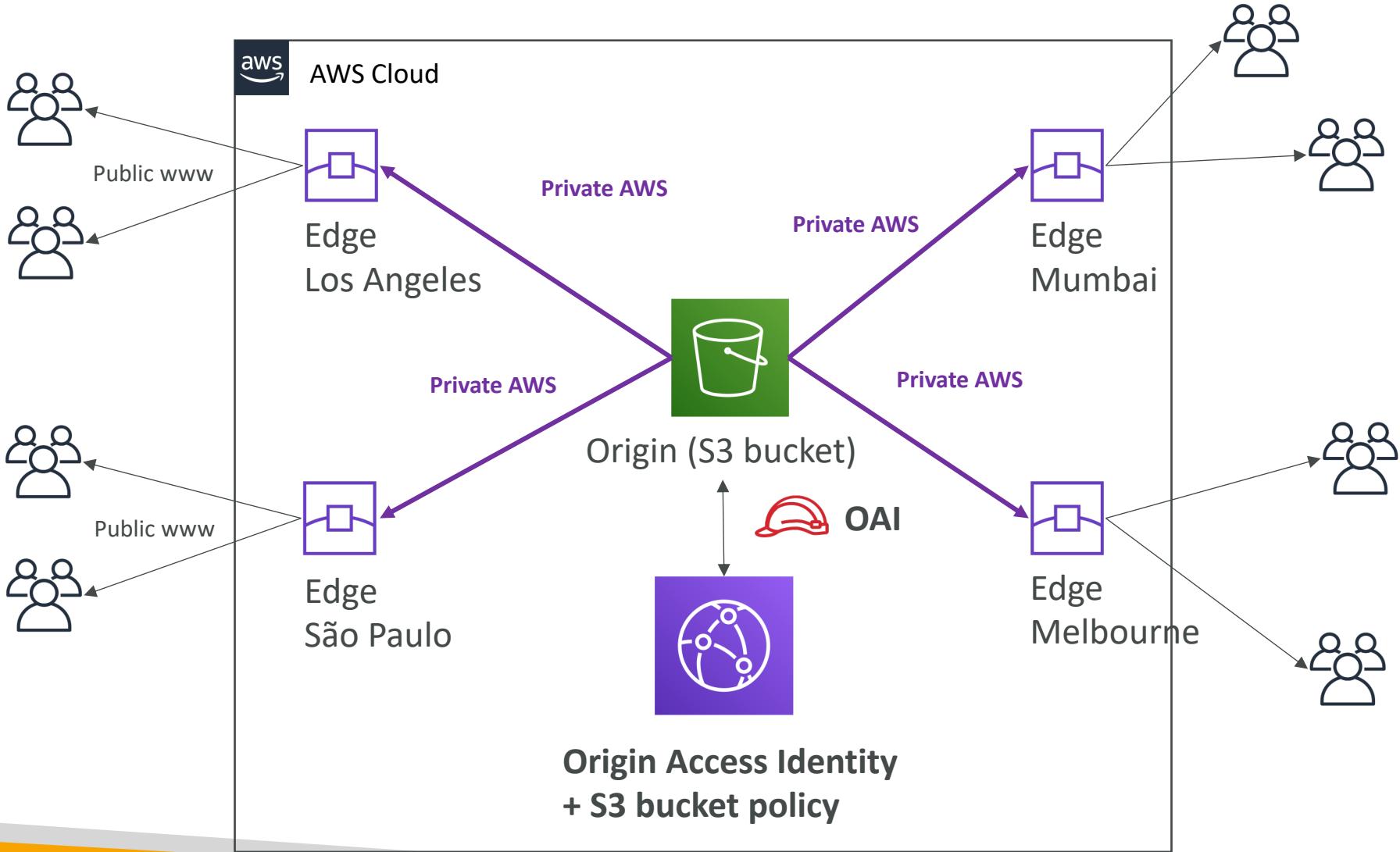


Source: <https://aws.amazon.com/cloudfront/features/?nc=sn&loc=2>

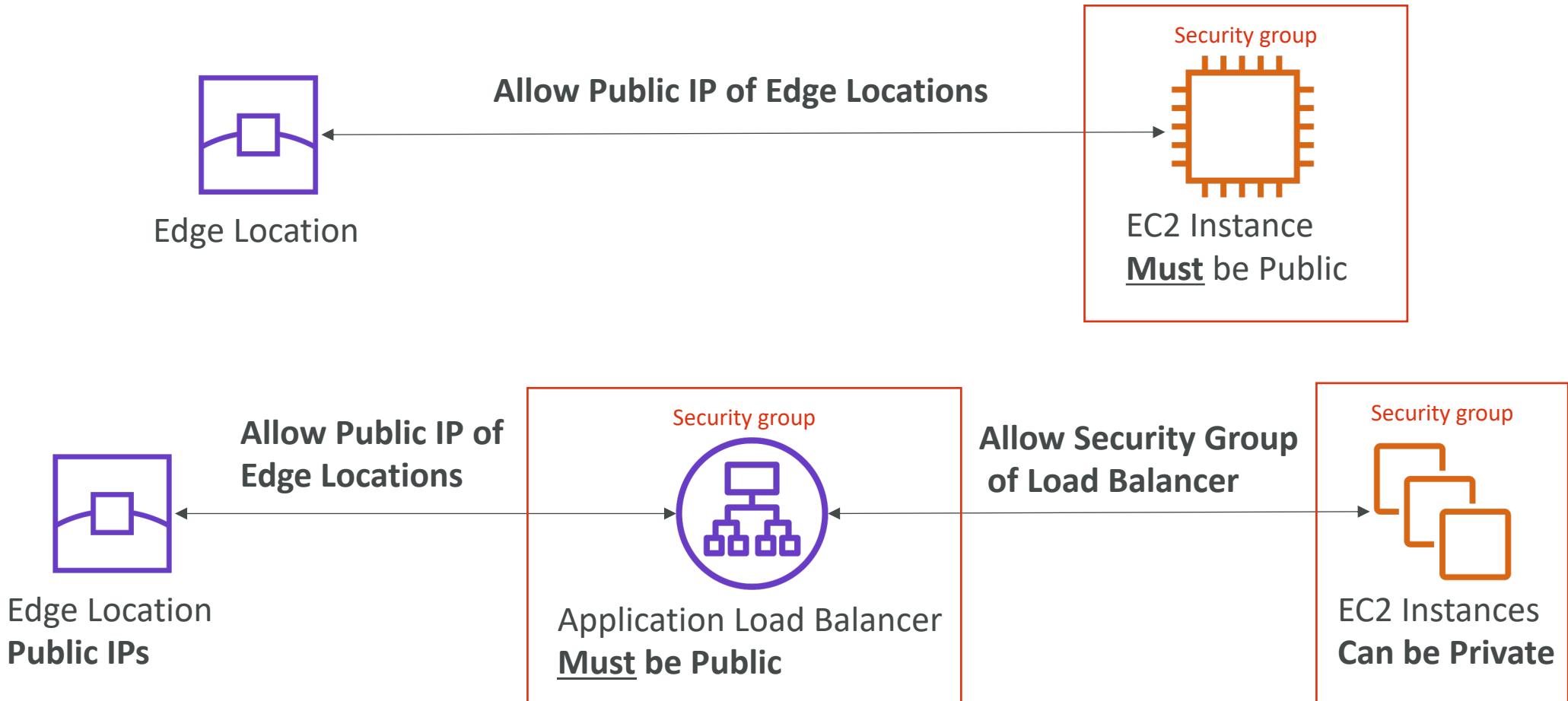
# CloudFront – Origins

- S3 Bucket
  - For distributing files
  - Enhanced security with CloudFront Origin Access Identity (OAI)
  - CloudFront can be used as an ingress (to upload files to S3)
- S3 Bucket configured as a website
  - First, enable Static Website hosting on the bucket
- MediaStore Container & MediaPackage Endpoint
  - To deliver Video On Demand (VOD) or live streaming video using AWS Media Services
- Custom Origin (HTTP)
  - EC2 instance
  - Elastic Load Balancer (CLB or ALB)
  - API Gateway (for more control... otherwise use API Gateway Edge)
  - Any HTTP backend you want

# CloudFront – S3 as an Origin



# CloudFront – EC2 or ALB as an origin



<http://d7uri8nf7uskq.cloudfront.net/tools/list-cloudfront-ips>

# CloudFront vs S3 Cross Region Replication

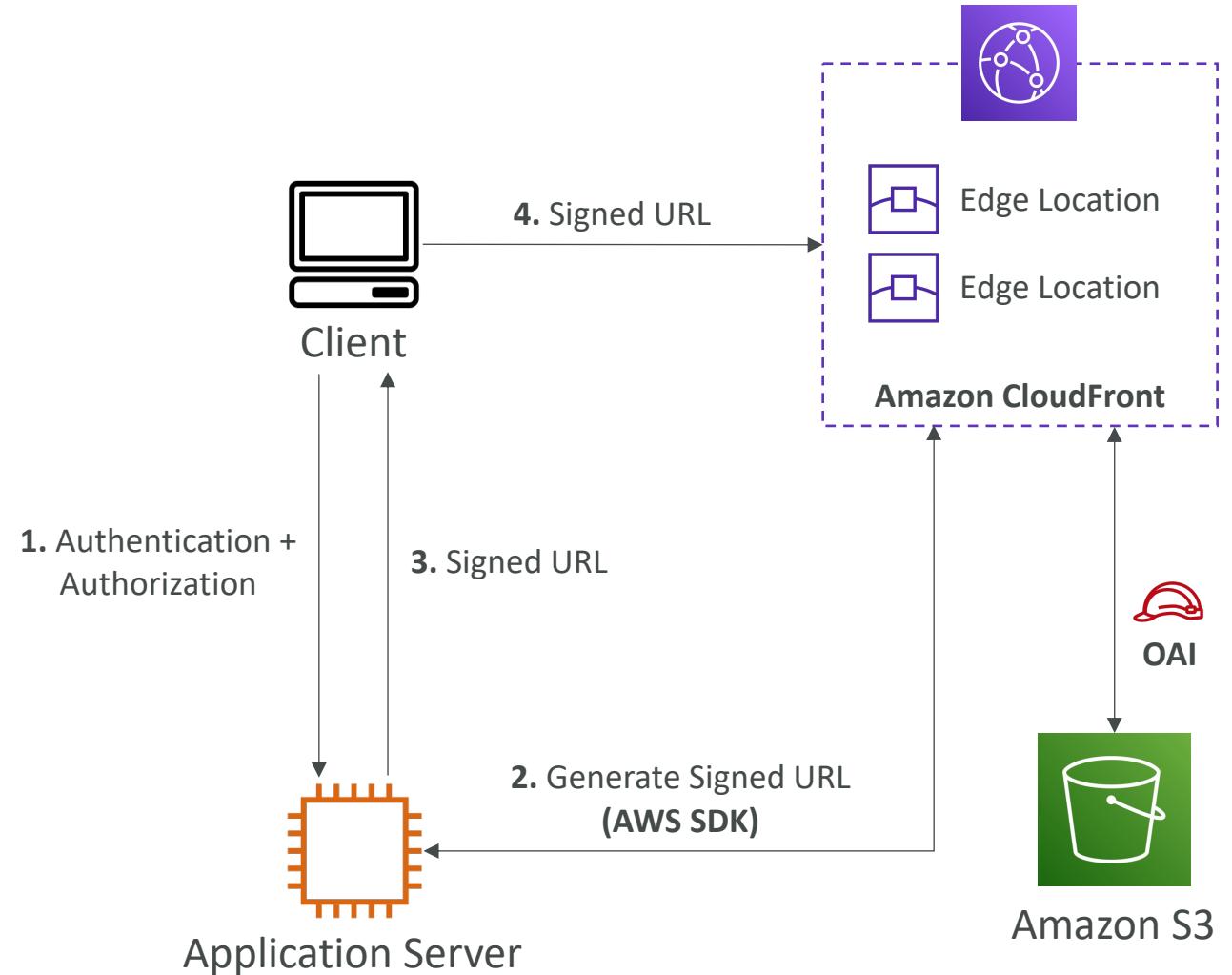
- CloudFront:
  - Global Edge network
  - Files are cached for a TTL (maybe a day)
  - Great for static content that must be available everywhere
- S3 Cross Region Replication:
  - Must be setup for each region you want replication to happen
  - Files are updated in near real-time
  - Read only
  - Great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

# CloudFront Geo Restriction

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

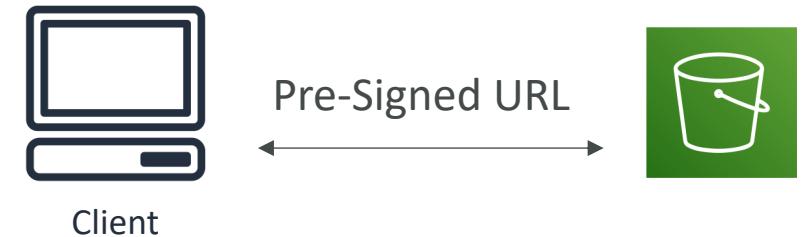
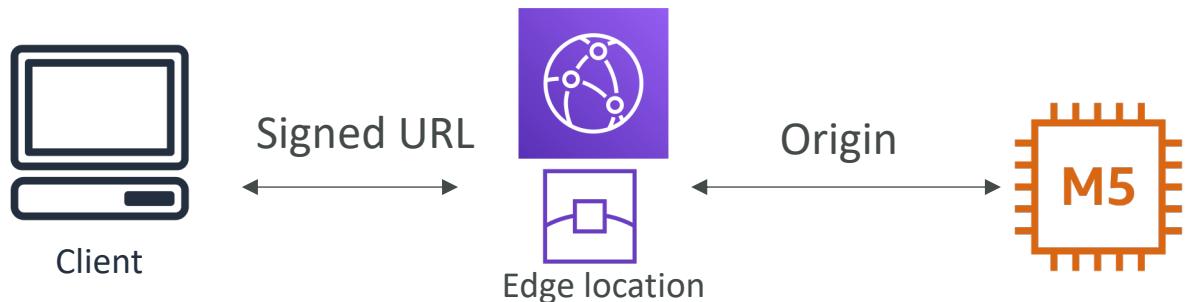
# CloudFront Signed URL Diagram

- Signed URL with expiration to control access to content in CloudFront
- The Signed URLs are generated by an API call into CloudFront as a trusted signer



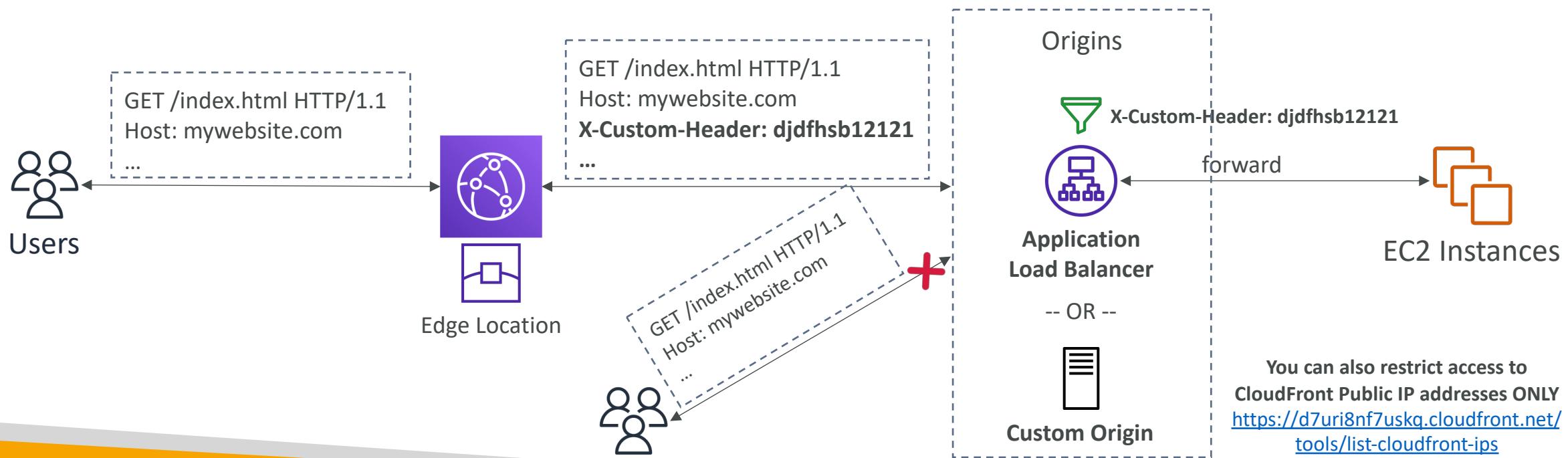
# CloudFront Signed URL vs S3 Pre-Signed URL

- CloudFront Signed URL:
  - Allow access to a path, no matter the origin
  - Account wide key-pair, only the root can manage it
  - Can filter by IP, path, date, expiration
  - Can leverage caching features
- S3 Pre-Signed URL:
  - Issue a request as the person who pre-signed the URL
  - Uses the IAM key of the signing IAM principal
  - Limited lifetime



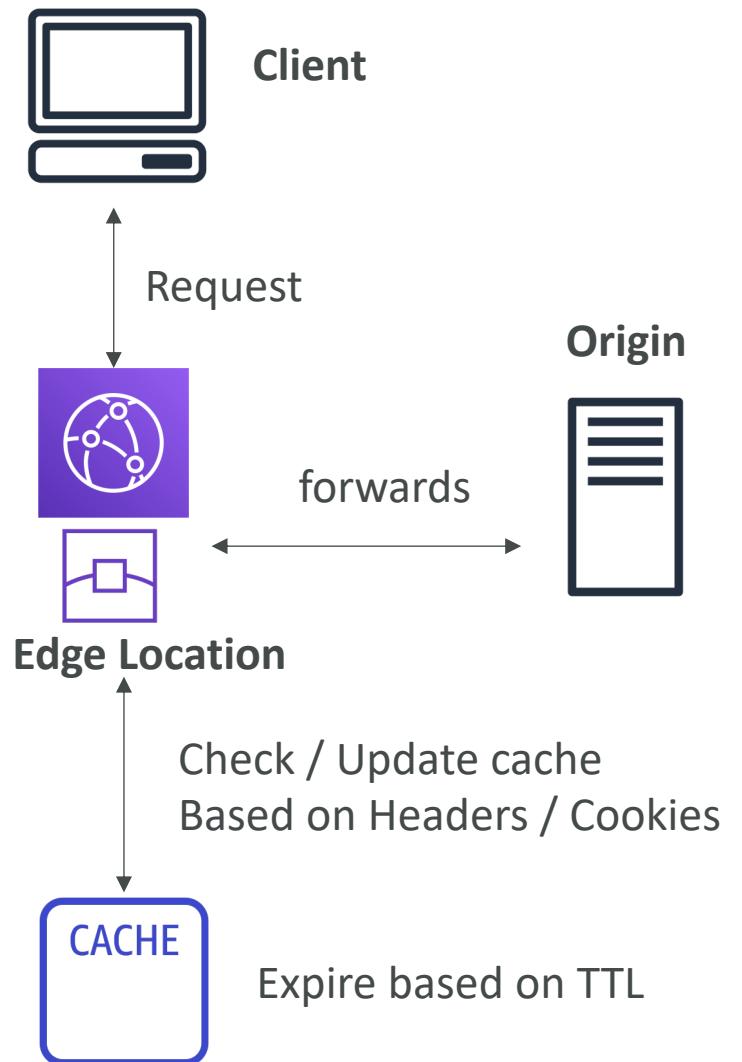
# CloudFront – Restrict Access to Application Load Balancers and Custom Origins

- Prevent direct access to your ALB or Custom Origins (only access through CloudFront)
- First, configure CloudFront to add a **Custom HTTP Header** to requests it sends to the ALB
- Second, configure the ALB to only forward requests that contain that Custom HTTP Header
- Keep the custom header name and value secret!



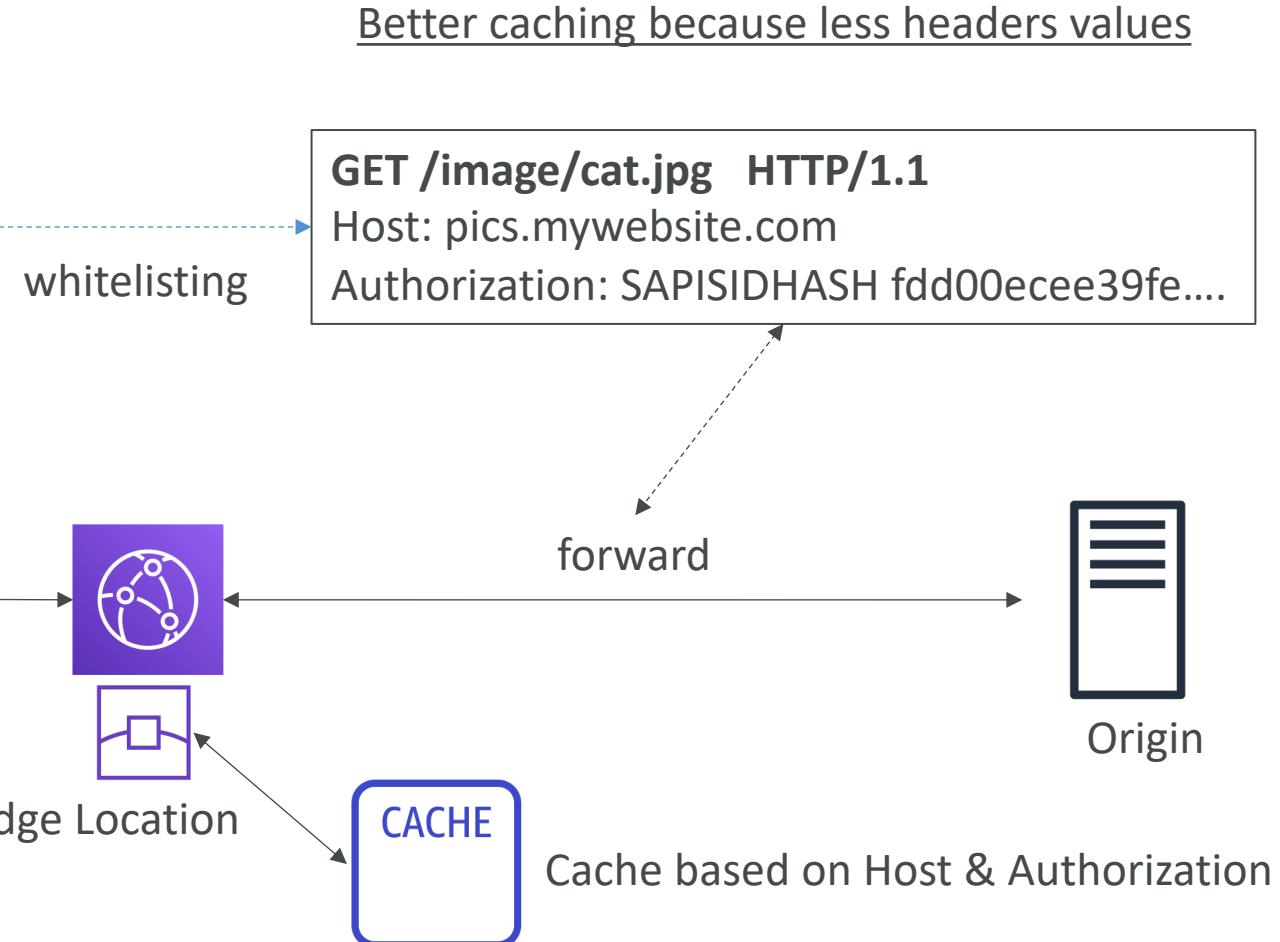
# CloudFront Caching

- Cache based on
  - Headers
  - Session Cookies
  - Query String Parameters
- The cache lives at each CloudFront Edge Location
- You want to maximize the cache hit rate to minimize requests on the origin
- Control the TTL (0 seconds to 1 year), can be set by the origin using the Cache-Control header, Expires header...

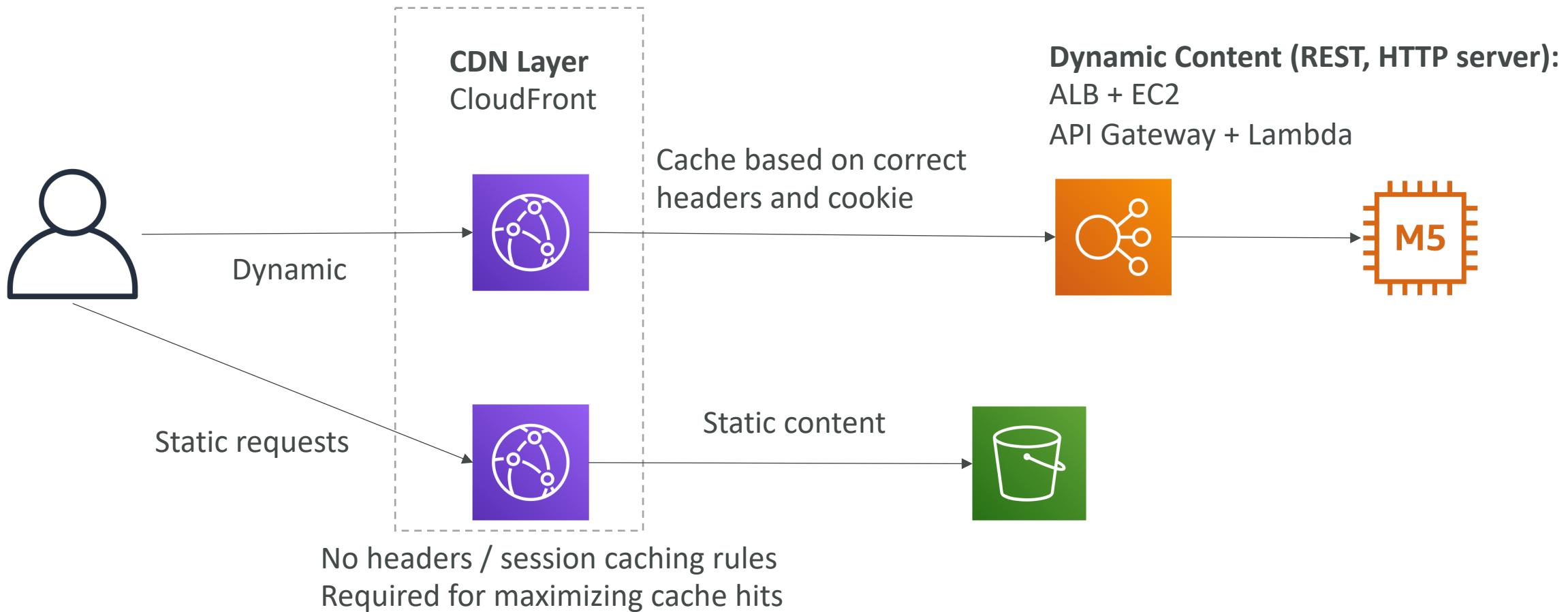


# CloudFront Caching – Whitelist Headers

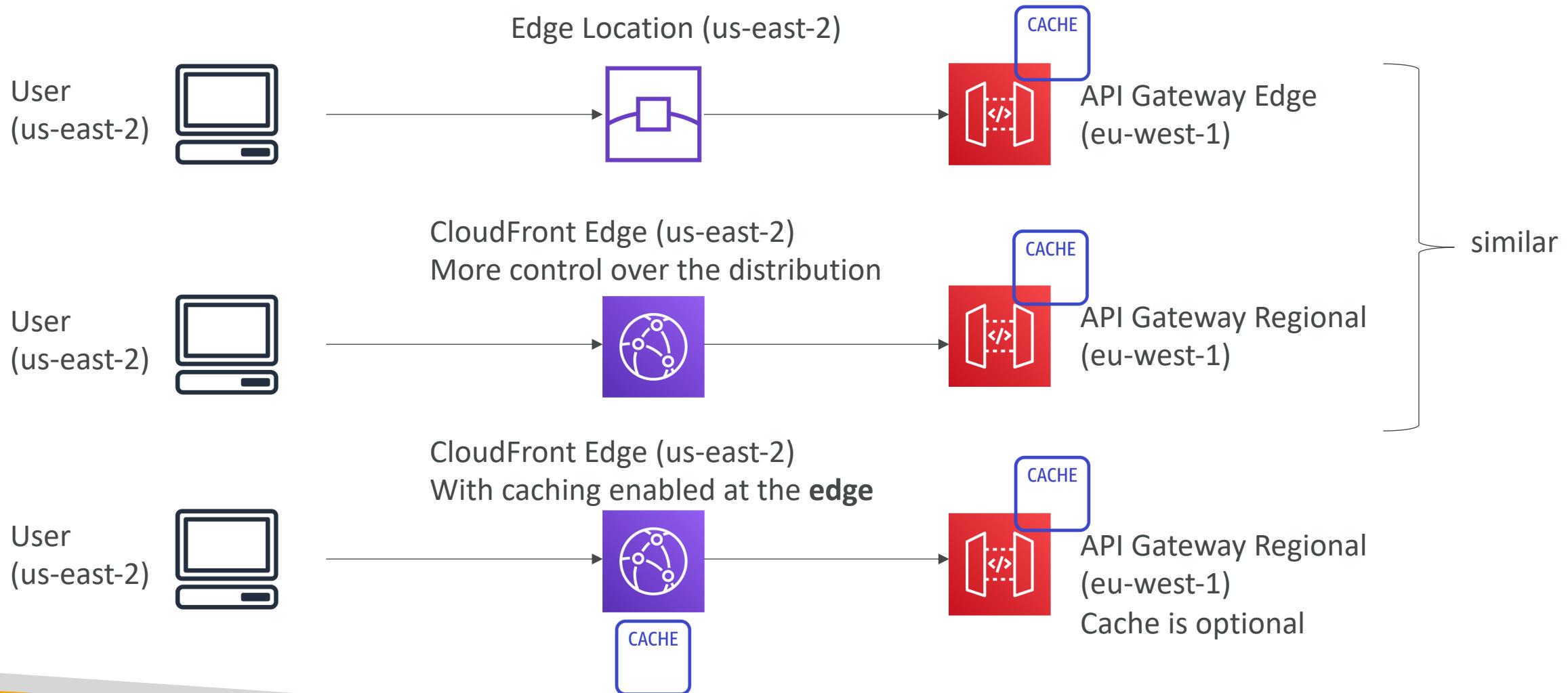
```
GET /image/cat.jpg HTTP/1.1
Host: pics.mywebsite.com
User-Agent: Mozilla/5.0 (Mac OS X 10_15_2....)
Date: Tue, 28 Jan 2020 17:01:57 GMT
Authorization: SAPISIDHASH fdd00ecee39fe....
Keep-Alive: 300
Accept-Ranges: bytes
```



# CloudFront – Maximize cache hits by separating static and dynamic distributions



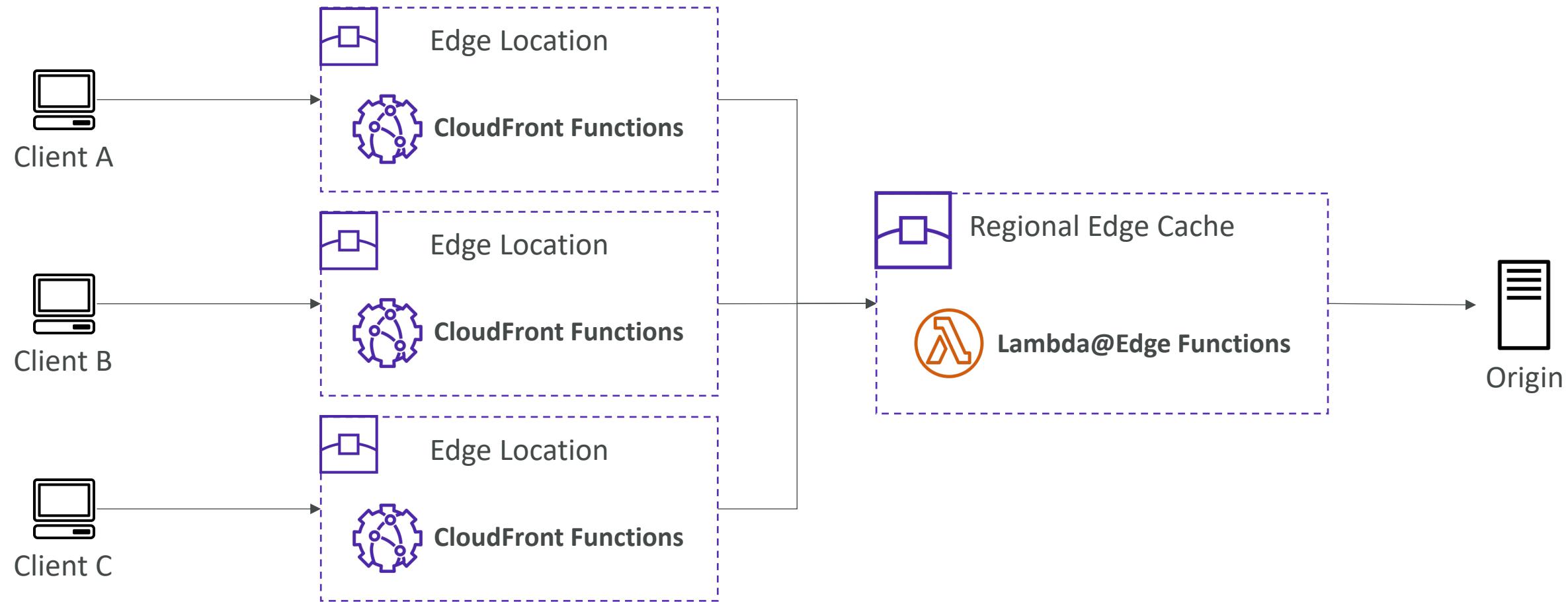
# CloudFront Caching vs API Gateway Caching



# CloudFront – Customization At The Edge

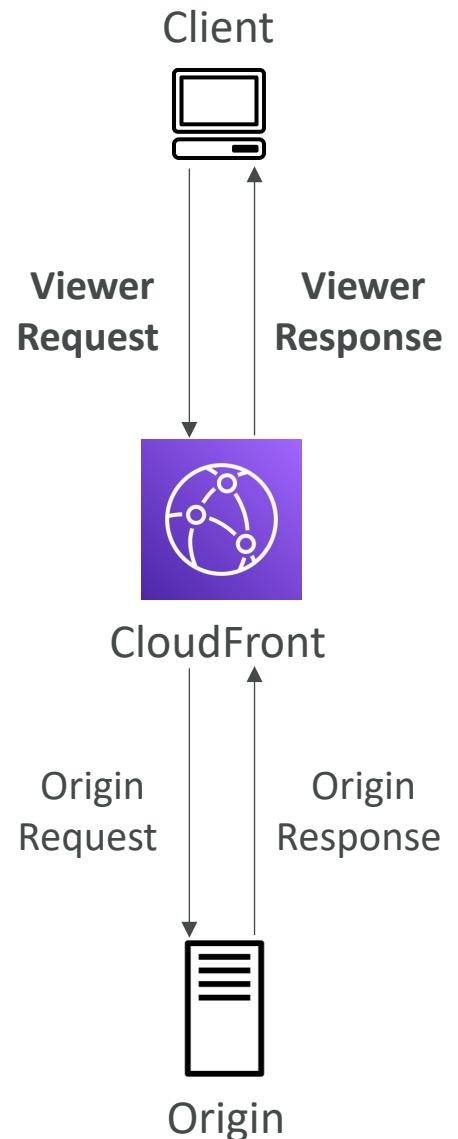
- Many modern applications execute some form of the logic at the edge
- **Edge Function:**
  - A code that you write and attach to CloudFront distributions
  - Runs close to your users to minimize latency
  - Doesn't have any cache, only to change requests/responses
  - CloudFront provides two types: **CloudFront Functions & Lambda@Edge**
- Use cases:
  - Manipulate HTTP requests and responses
  - Implement request filtering before reaching your application
  - User authentication and authorization
  - Generate HTTP responses at the edge
  - A/B Testing
  - Bot mitigation at the edge
- You don't have to manage any servers, deployed globally

# CloudFront Functions & Lambda@Edge



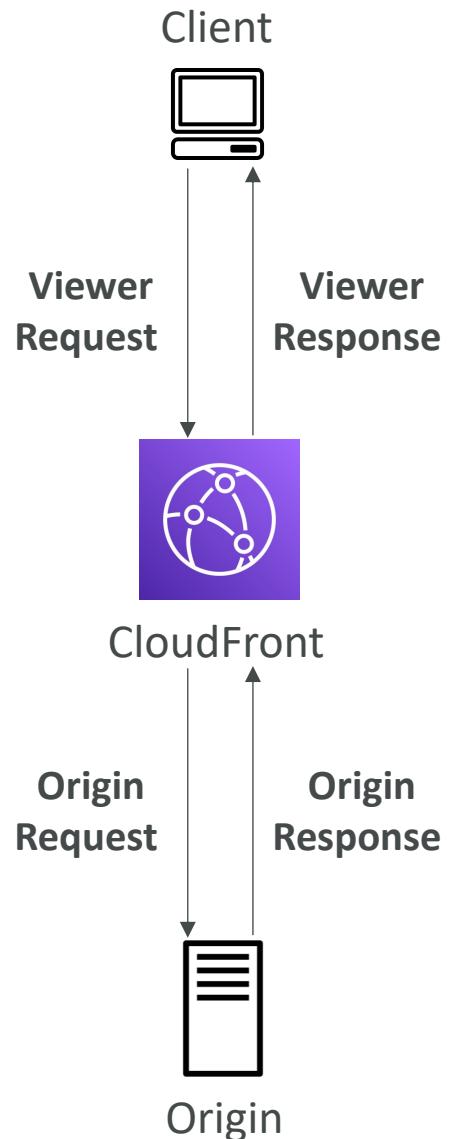
# CloudFront – CloudFront Functions

- Lightweight functions written in JavaScript
- For high-scale, latency-sensitive CDN customizations
- Sub-ms startup times, millions of requests/second
- Run at Edge Locations
- Process-based isolation
- Used to change Viewer requests and responses:
  - **Viewer Request:** after CloudFront receives a request from a viewer
  - **Viewer Response:** before CloudFront forwards the response to the viewer
- Native feature of CloudFront (manage code entirely within CloudFront)



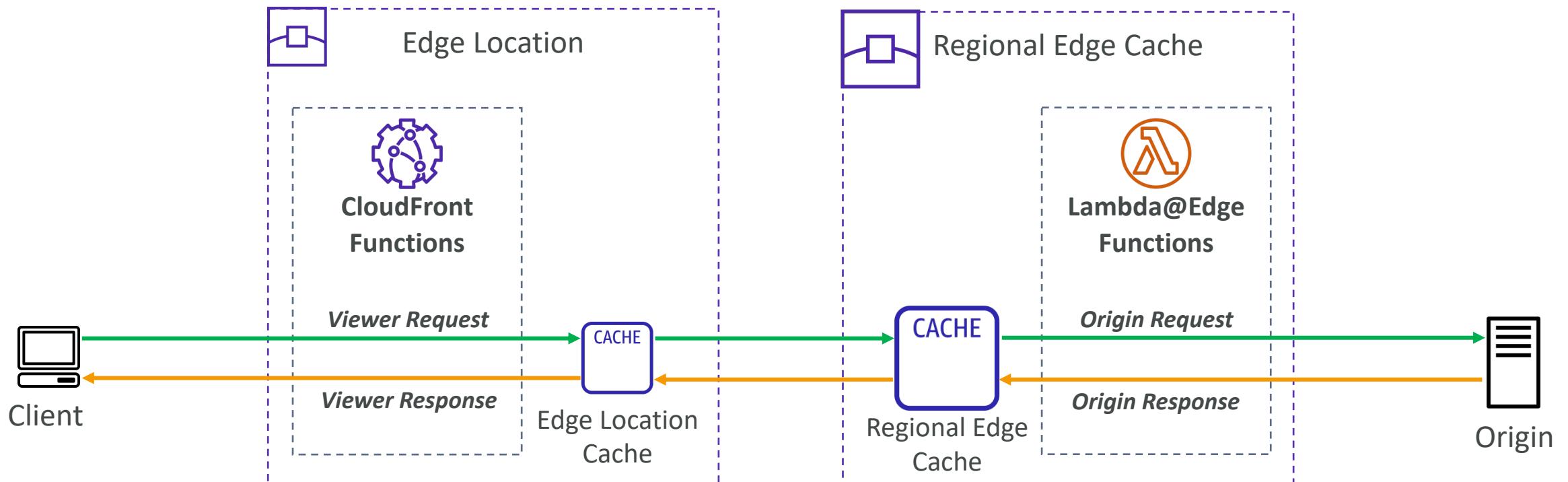
# CloudFront – Lambda@Edge

- Lambda functions written in NodeJS or Python
- Scales to 1000s of requests/second
- Runs at the nearest Regional Edge Cache
- VM-based isolation
- Used to change CloudFront requests and responses:
  - **Viewer Request** – after CloudFront receives a request from a viewer
  - **Origin Request** – before CloudFront forwards the request to the origin
  - **Origin Response** – after CloudFront receives the response from the origin
  - **Viewer Response** – before CloudFront forwards the response to the viewer
- Author your functions in one AWS Region (us-east-1), then CloudFront replicates to its locations



# CloudFront Functions with Lambda@Edge

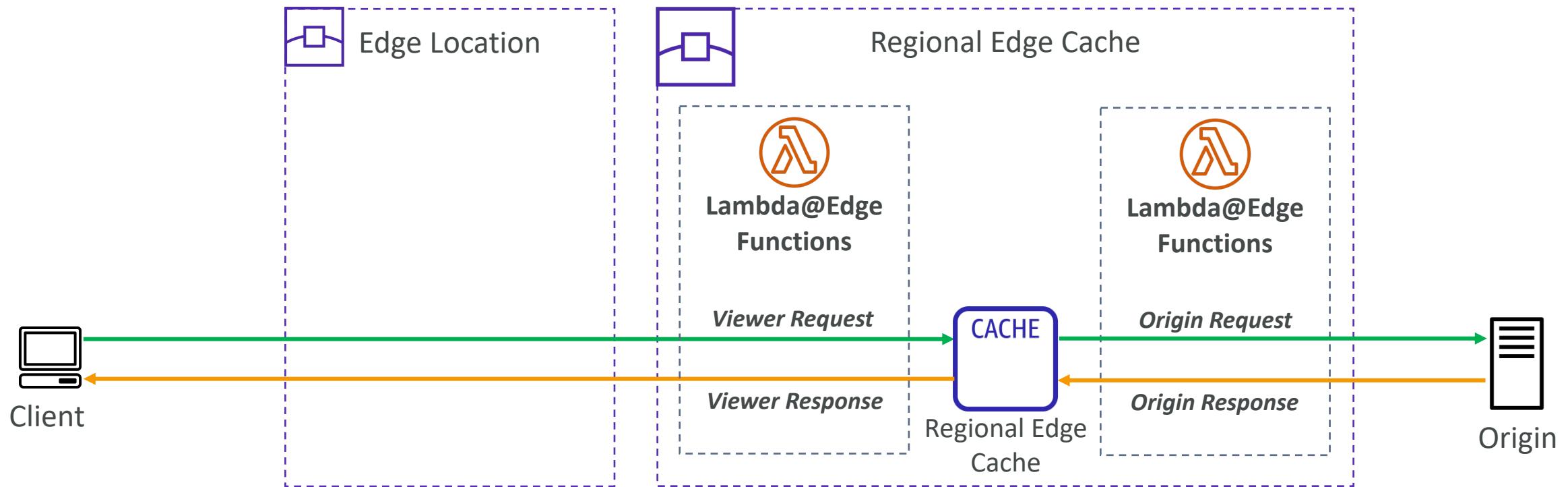
**CloudFront Functions and Lambda@Edge can be used together**



**NOTE: You can't combine CloudFront Functions and Lambda@Edge in viewer events (viewer request & viewer response)**

# Using Lambda@Edge Only

Use when you need some of the capabilities of Lambda@Edge that aren't available with CloudFront Functions (e.g., longer execution time, network access, ...)



# CloudFront Functions vs. Lambda@Edge

	CloudFront Functions	Lambda@Edge
Runtime Support	JavaScript	Node.js, Python
Execution Location	Edge Locations	Regional Edge Caches
CloudFront Triggers	- Viewer Request/Response	- Viewer Request/Response - Origin Request/Response
Isolation	Process-based	VM-based
Max. Execution Time	< 1 ms	- 5 seconds (viewers triggers) - 30 seconds (origin triggers)
Max. Memory	2 MB	- 128 MB (viewer triggers) - 10 GB (origin triggers)
Total Package Size	10 KB	- 1 MB (viewer triggers) - 50 MB (origin t
Network Access, File System Access	No	Yes
Access to the Request Body	No	Yes
Pricing	Free tier available, 1/6 <sup>th</sup> price of @Edge	No free tier, charged per request & duration

# CloudFront Functions vs. Lambda@Edge – Use Cases

## CloudFront Functions

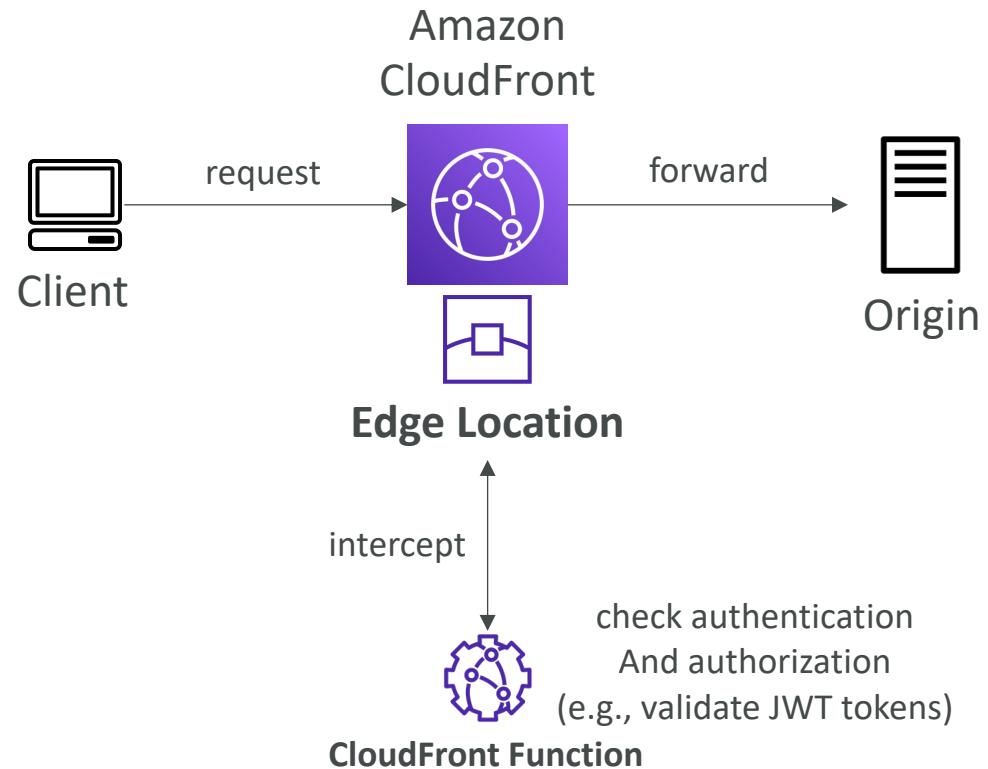
- Cache key normalization
  - Transform request attributes (headers, cookies, query strings, URL) to create an optimal Cache Key
- Header manipulation
  - Insert/modify/delete HTTP headers in the request or response
- URL rewrites or redirects
- Request authentication & authorization
  - Create and validate user-generated tokens (e.g., JWT) to allow/deny requests

## Lambda@Edge

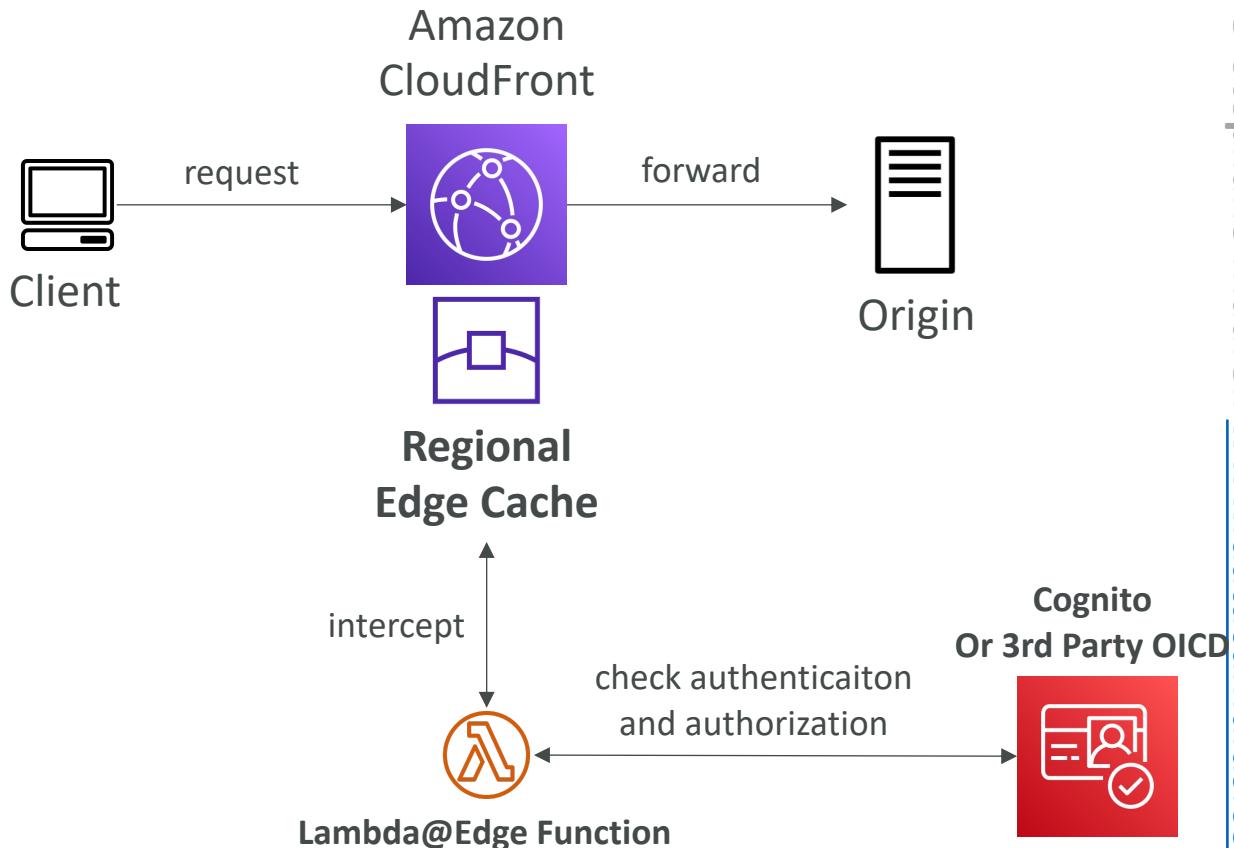
- Longer execution time (several ms)
- Adjustable CPU or memory
- Your code depends on a 3rd libraries (e.g., AWS SDK to access other AWS services)
- Network access to use external services for processing
- File system access or access to the body of HTTP requests

# CloudFront Functions vs. Lambda@Edge – Authentication and Authorization

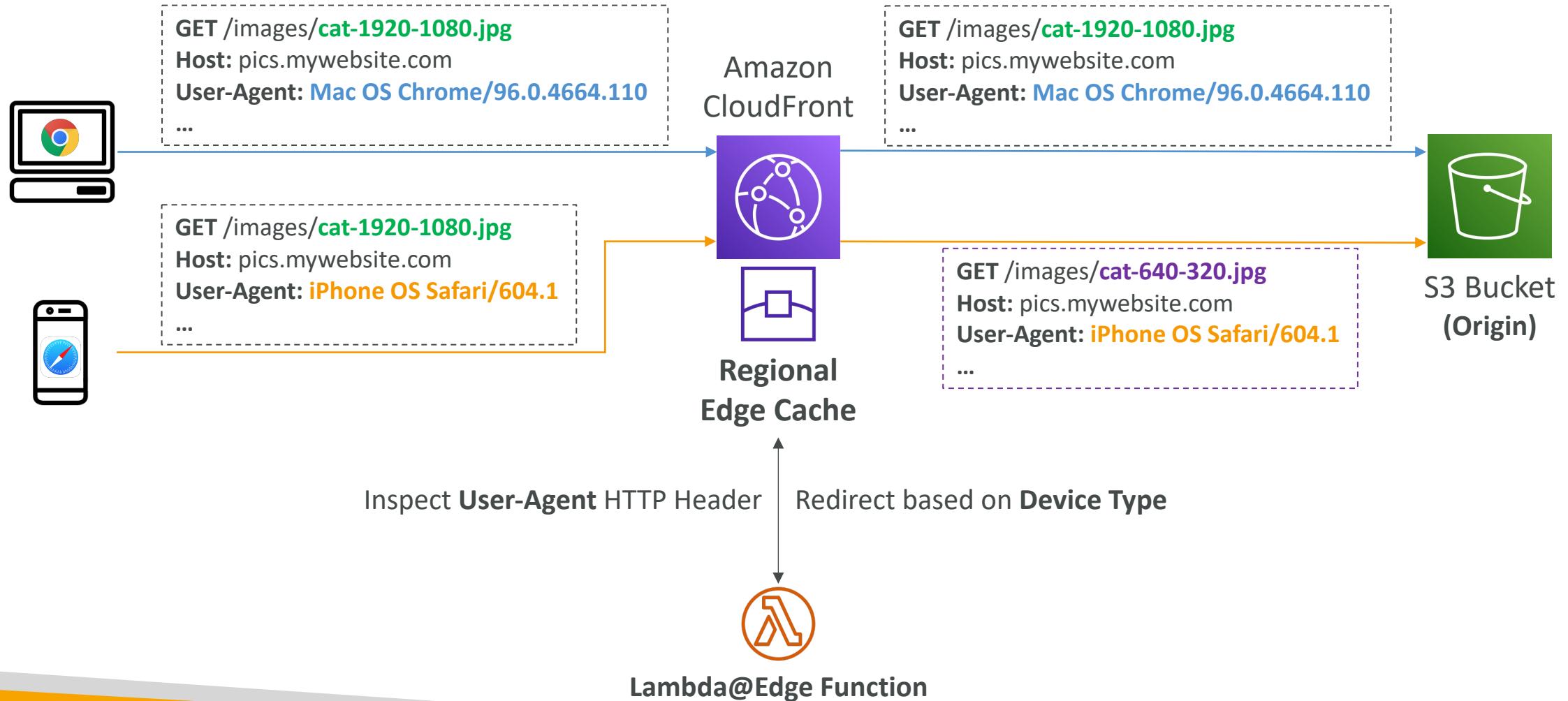
## CloudFront Functions



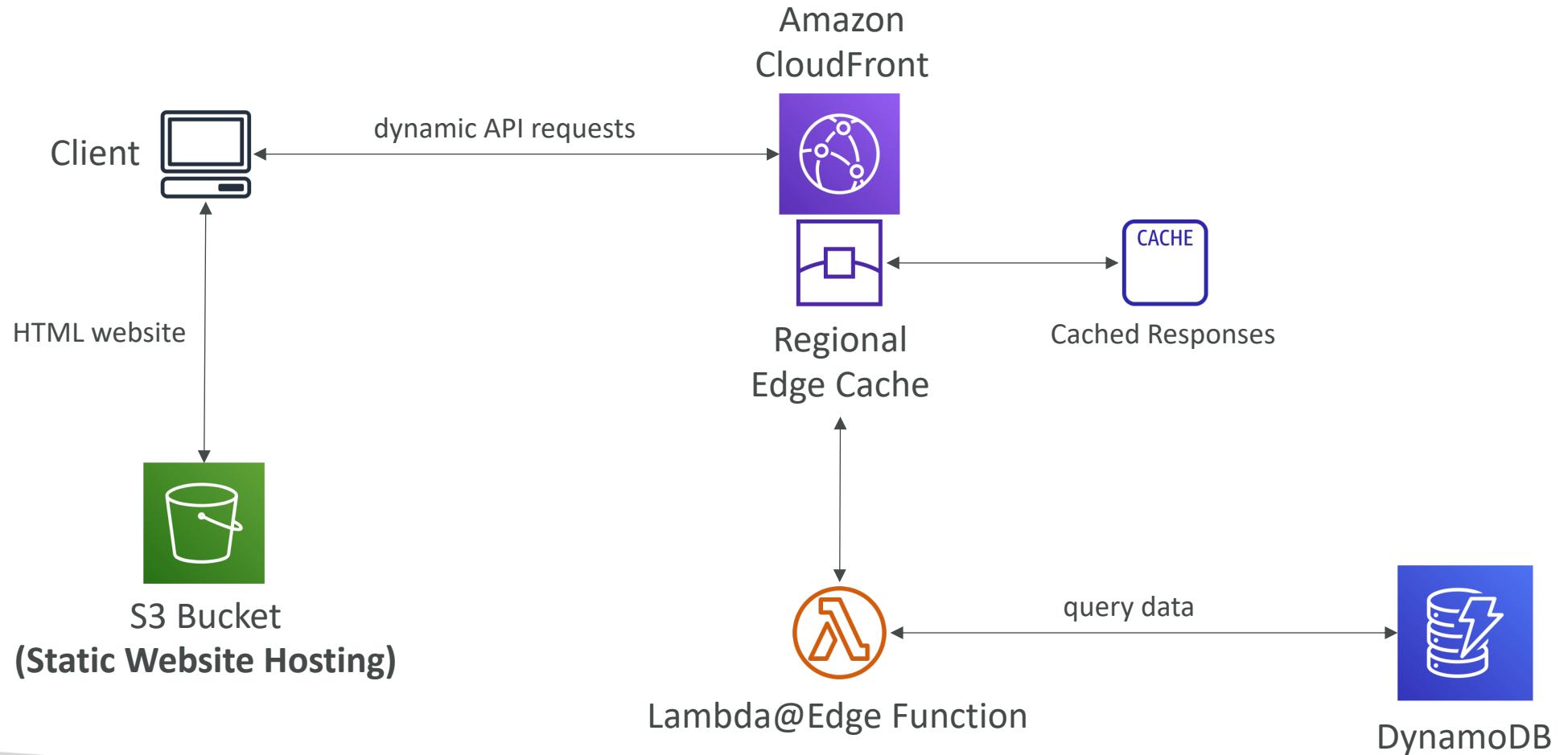
## Lambda@Edge



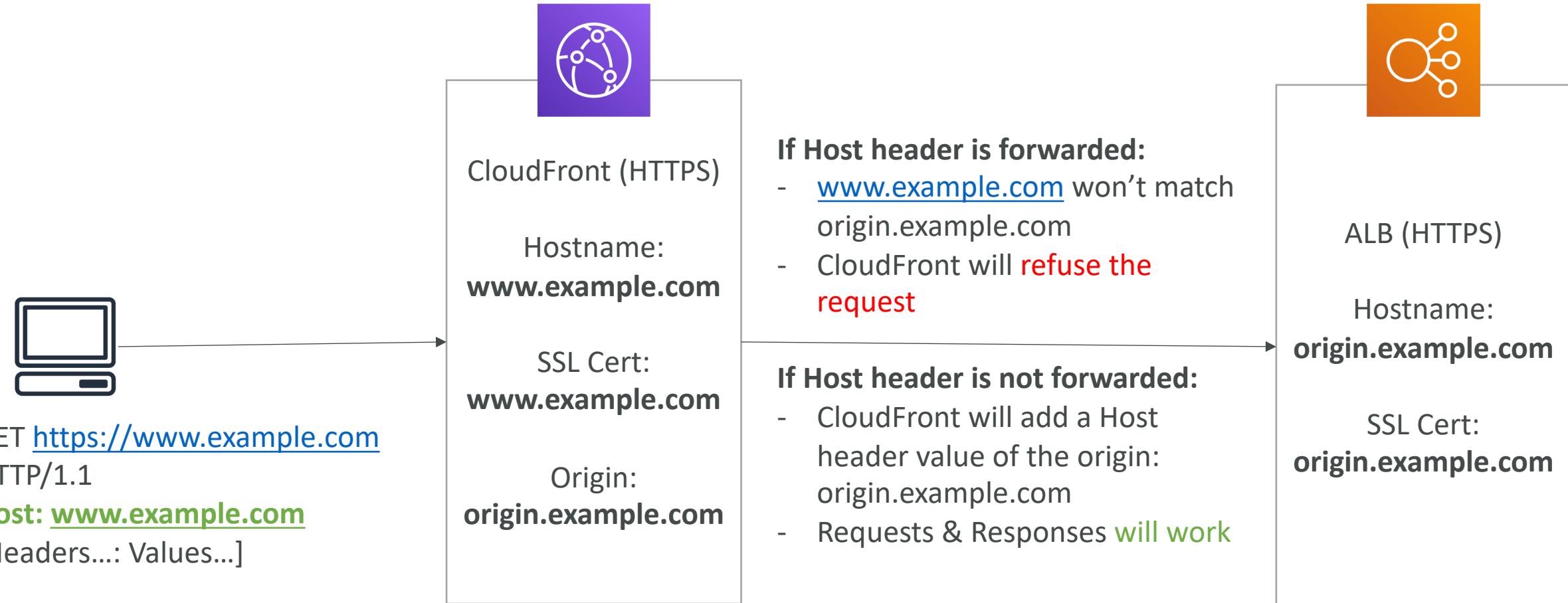
# Lambda@Edge: Loading content based on User-Agent



# Lambda@Edge – Global Application

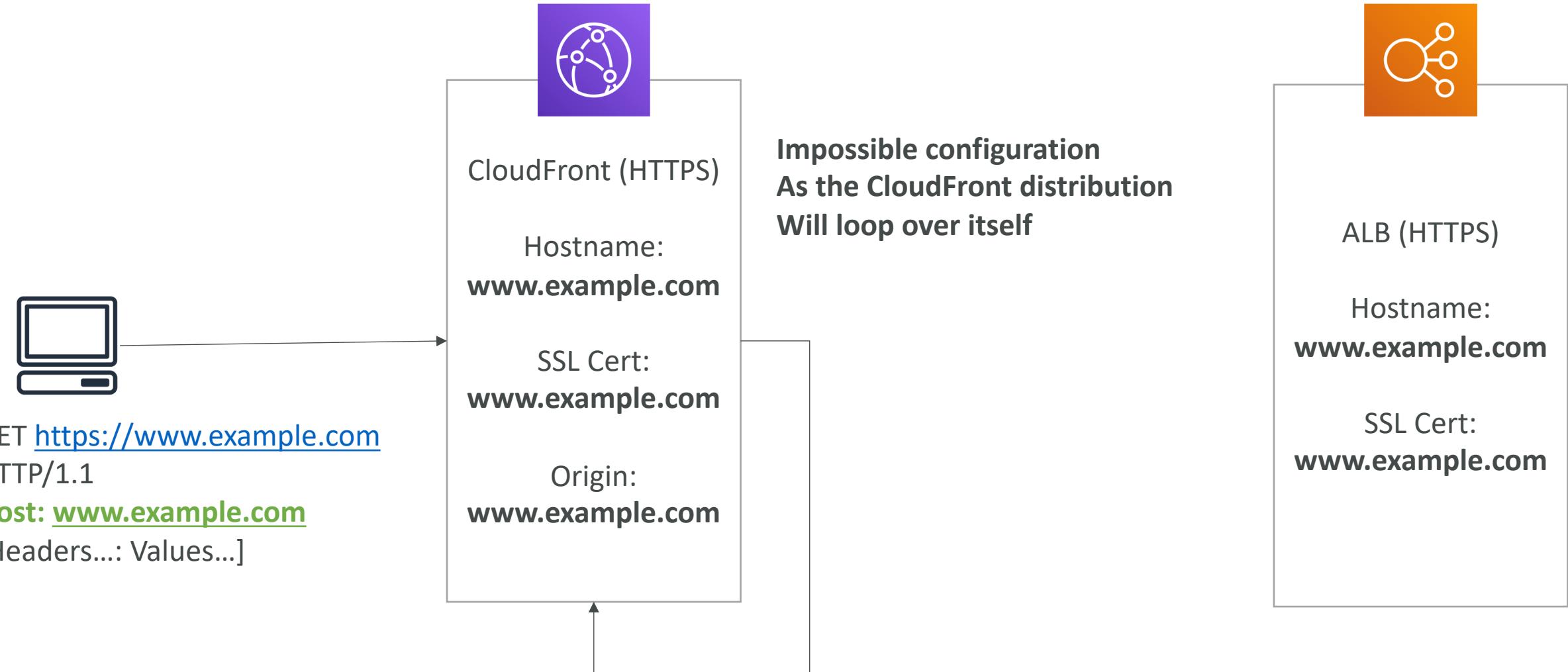


# CloudFront – HTTPS configuration and Host



Note: there are 2 SSL certificates to manage

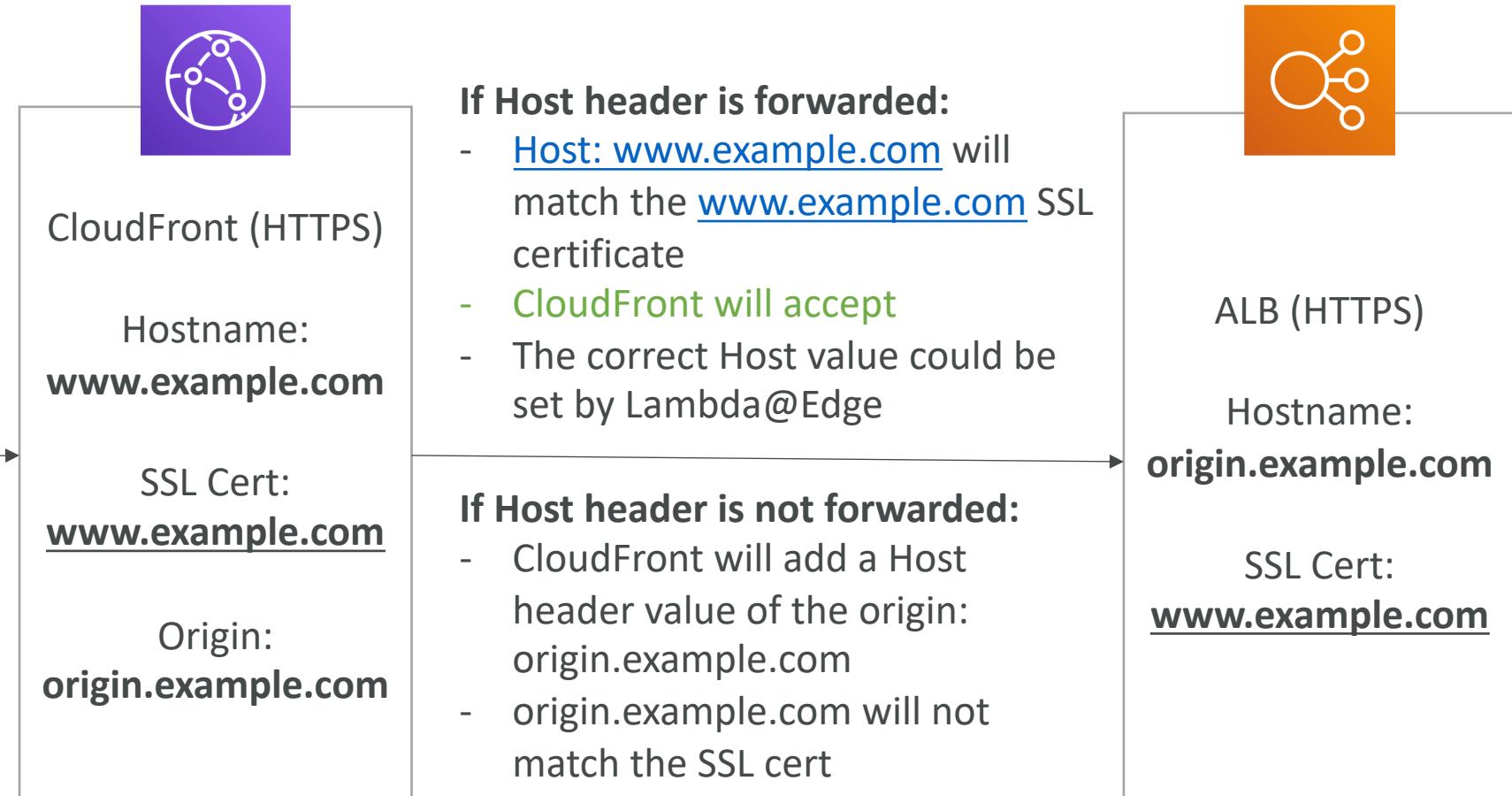
# CloudFront – HTTPS configuration and Host



# CloudFront – HTTPS configuration and Host



GET <https://www.example.com>  
HTTP/1.1  
**Host: www.example.com**  
[Headers...: Values...]



Note: there is 1 SSL certificates to manage



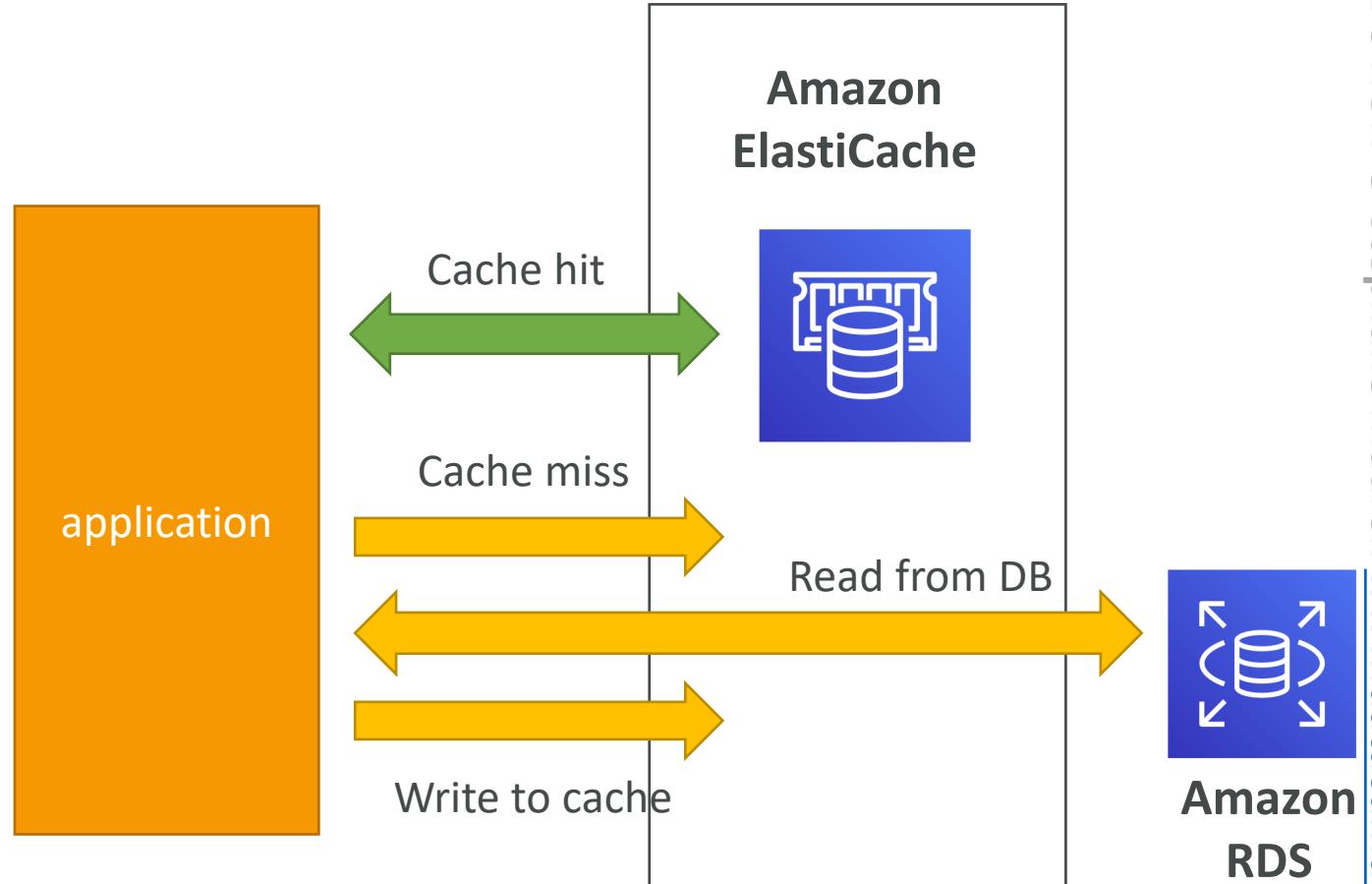
# Amazon ElastiCache Overview

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

# ElastiCache

## Solution Architecture - DB Cache

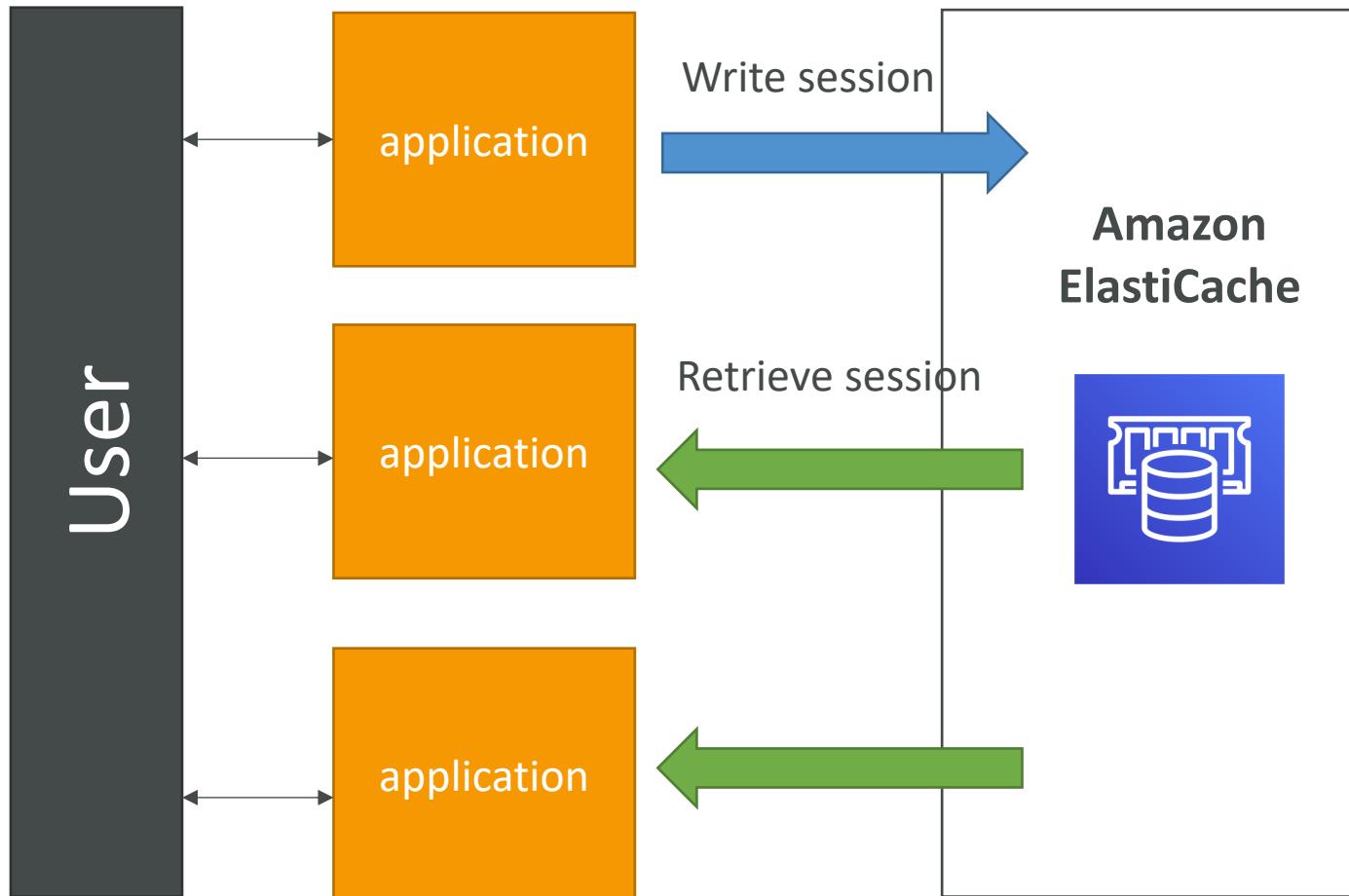
- Applications queries ElastiCache, if not available, get from RDS and store in ElastiCache.
- Helps relieve load in RDS
- Cache must have an invalidation strategy to make sure only the most current data is used in there.



# ElastiCache

## Solution Architecture – User Session Store

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



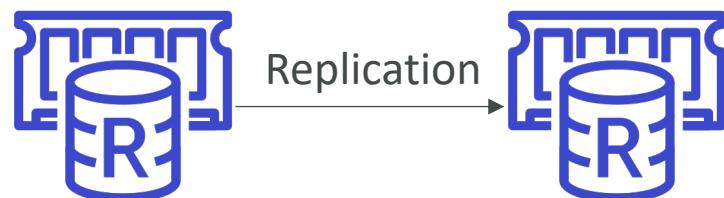
# ElastiCache – Redis vs Memcached

## REDIS

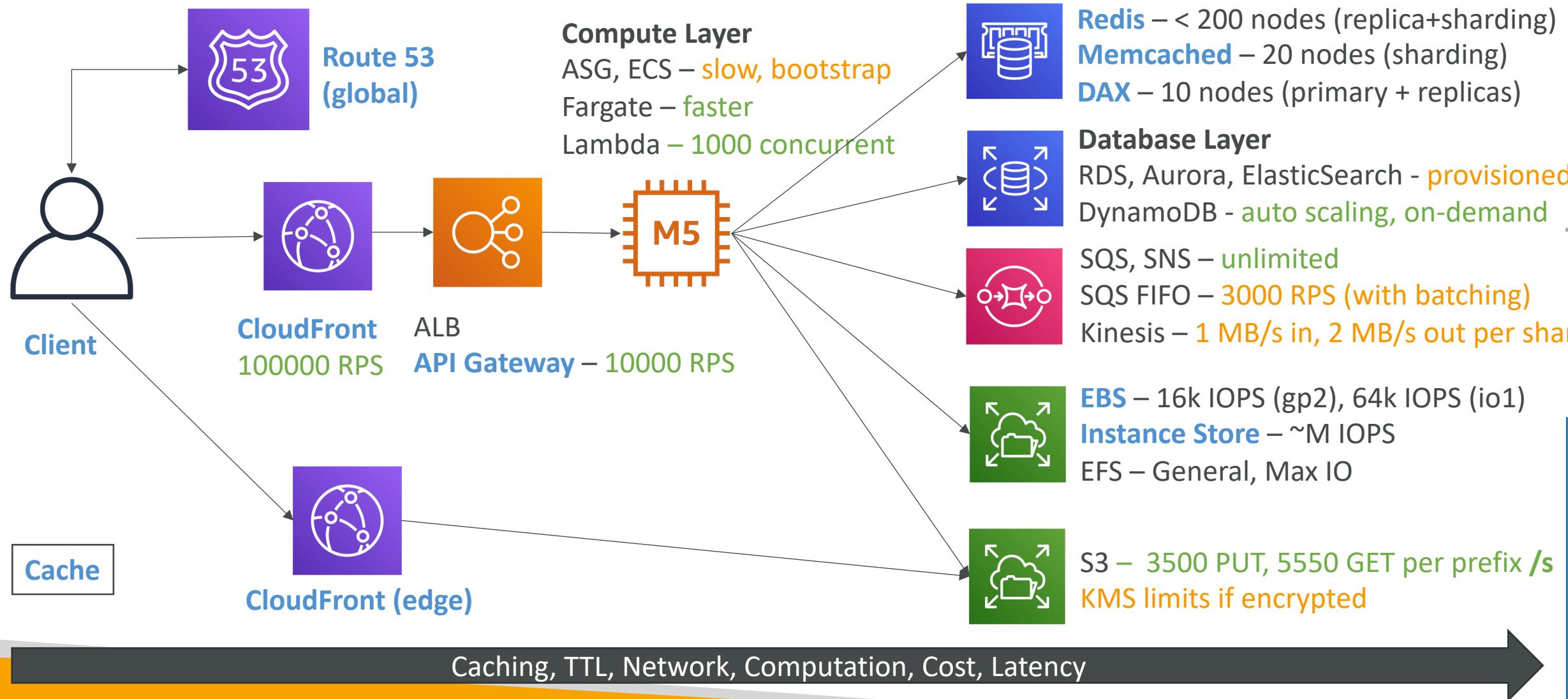
- Multi AZ with Auto-Failover
- Read Replicas to scale reads and have high availability
- Persistent, Data Durability: Append Only File (AOF), backup and restore features

## MEMCACHED

- Multi-node for partitioning of data (sharding)
- Non persistent
- No backup and restore
- Multi-threaded architecture



# Handling Extreme Rates



# Databases Section

# DynamoDB – in short



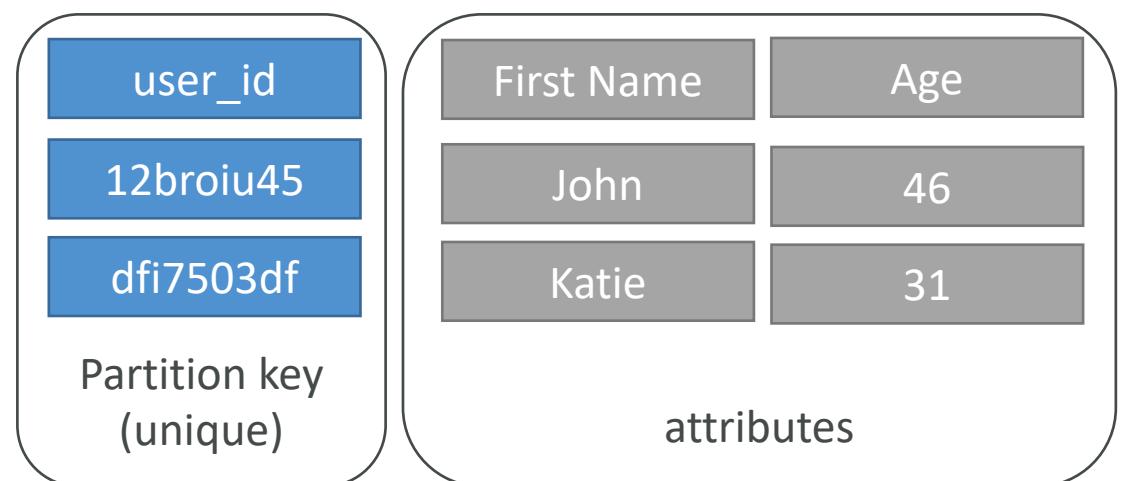
- NoSQL database, fully managed, massive scale (1,000,000 rps)
- Similar to Apache Cassandra (can migrate to DynamoDB)
- No disk space to provision, max object size is 400 KB
- Capacity: provisioned (WCU, RCU, & Auto Scaling) or on-demand
- Supports CRUD (Create Read Update Delete)
- Read: eventually or strong consistency
- Supports transactions across multiple tables (ACID support)
- Backups available, point in time recovery
- Table classes: Standard and Infrequent Access (IA)

# DynamoDB - Basics

- DynamoDB is made of **tables**
- Each table has a **primary key** (must be decided at creation time)
- Each table can have an infinite number of items (= rows)
- Each item has **attributes** (can be added over time – can be null)
- Maximum size of a item is 400KB
- Data types supported are:
  - Scalar Types: String, Number, Binary, Boolean, Null
  - Document Types: List, Map
  - Set Types: String Set, Number Set, Binary Set

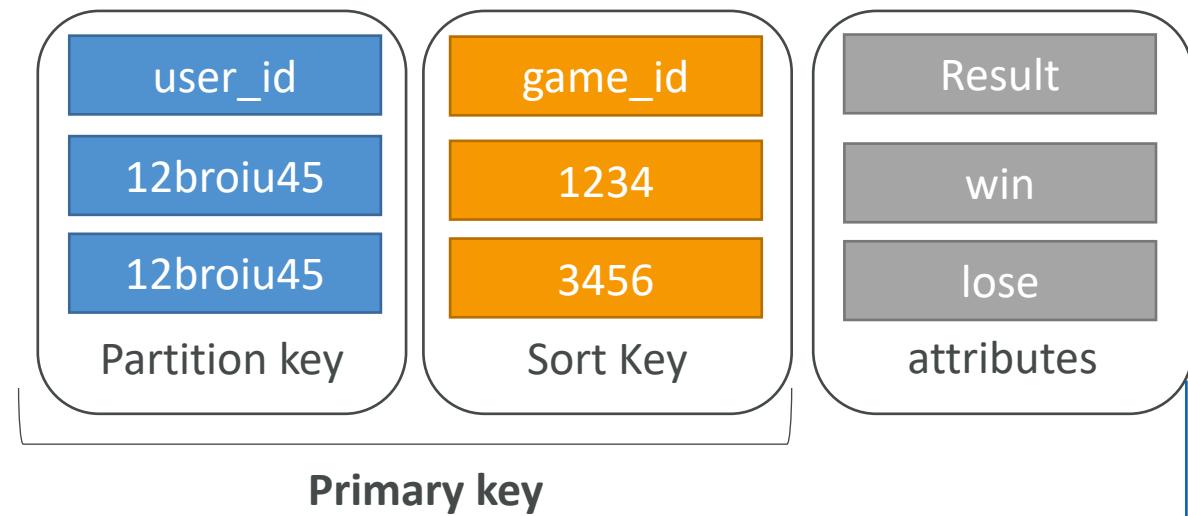
# DynamoDB – Primary Keys

- Option 1: Partition key only (HASH)
- Partition key must be unique for each item
- Partition key must be “diverse” so that the data is distributed
- Example: user\_id for a users table



# DynamoDB – Primary Keys

- Option 2: Partition key + Sort Key
- The combination must be unique
- Data is grouped by partition key
- Sort key == range key
- Example: users-games table
  - user\_id for the partition key
  - game\_id for the sort key
- Example good sort key: timestamp

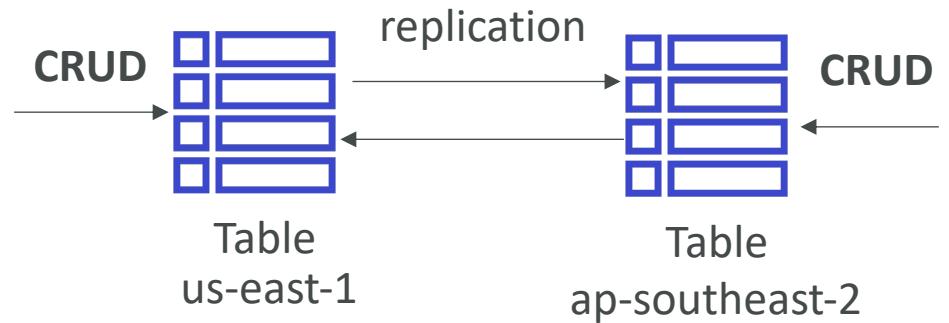
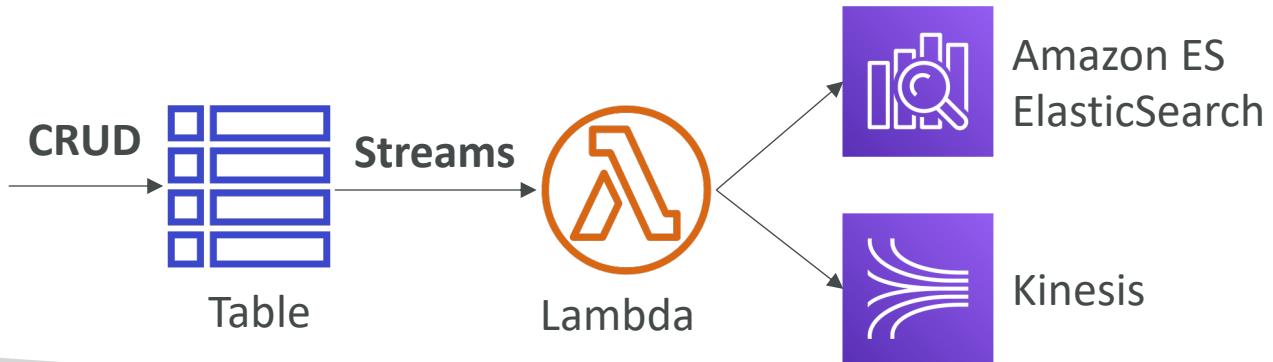


# DynamoDB – Indexes

- Object = primary key + optional sort key + attributes
- LSI – Local Secondary Index
  - Keep the same primary key
  - Select an alternative sort key
  - Must be defined at table creation time
- GSI – Global Secondary Index
  - Change the primary key and optional sort sort
  - Can be defined after the table is created
- You can only query by PK + sort key on the main table & indexes (**≠ RDS**)

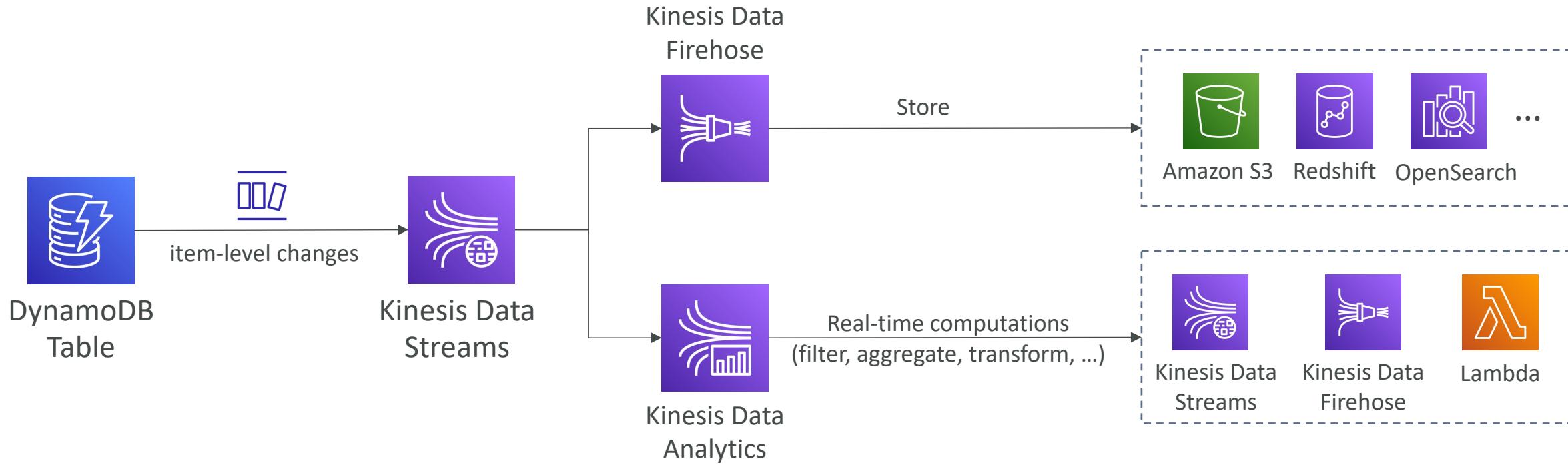
# DynamoDB – Important Features

- TTL: automatically expire row after a specified epoch date
- **DynamoDB Streams:**
  - react to changes to DynamoDB tables in real time
  - Can be read by AWS Lambda, EC2...
  - 24 hours retention of data
- **Global Tables:** (cross region replication)
  - Active Active replication, many regions
  - Must enable DynamoDB Streams
  - Useful for low latency, DR purposes



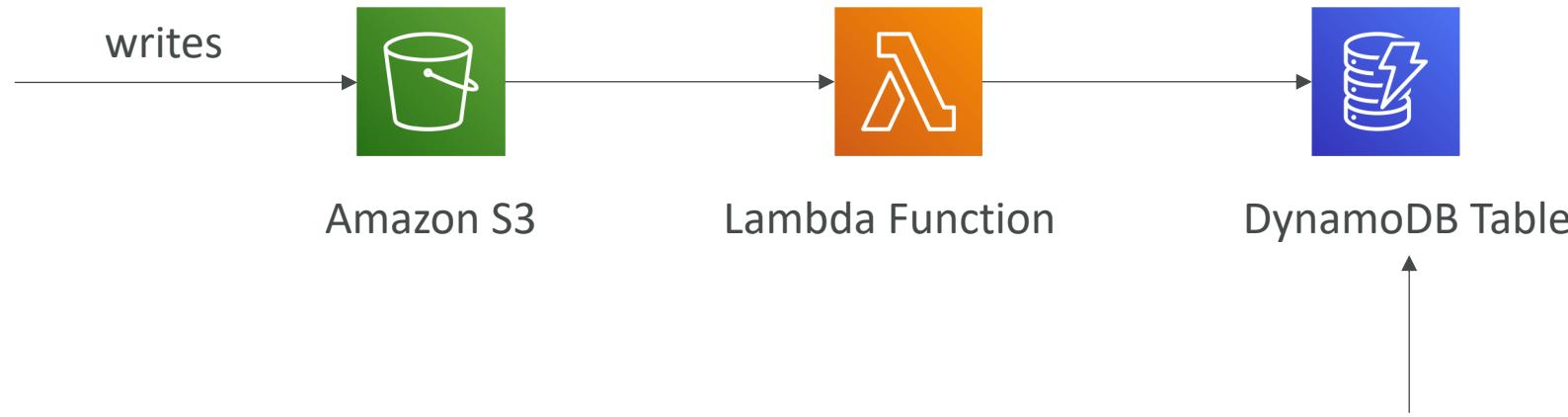
# Amazon Kinesis Data Streams for DynamoDB

- You can use Kinesis Data Streams to capture item-level changes in DynamoDB
- Custom and longer data retention period (> 24 hours in DynamoDB Streams)



# DynamoDB Solution Architecture

## Indexing objects in DynamoDB

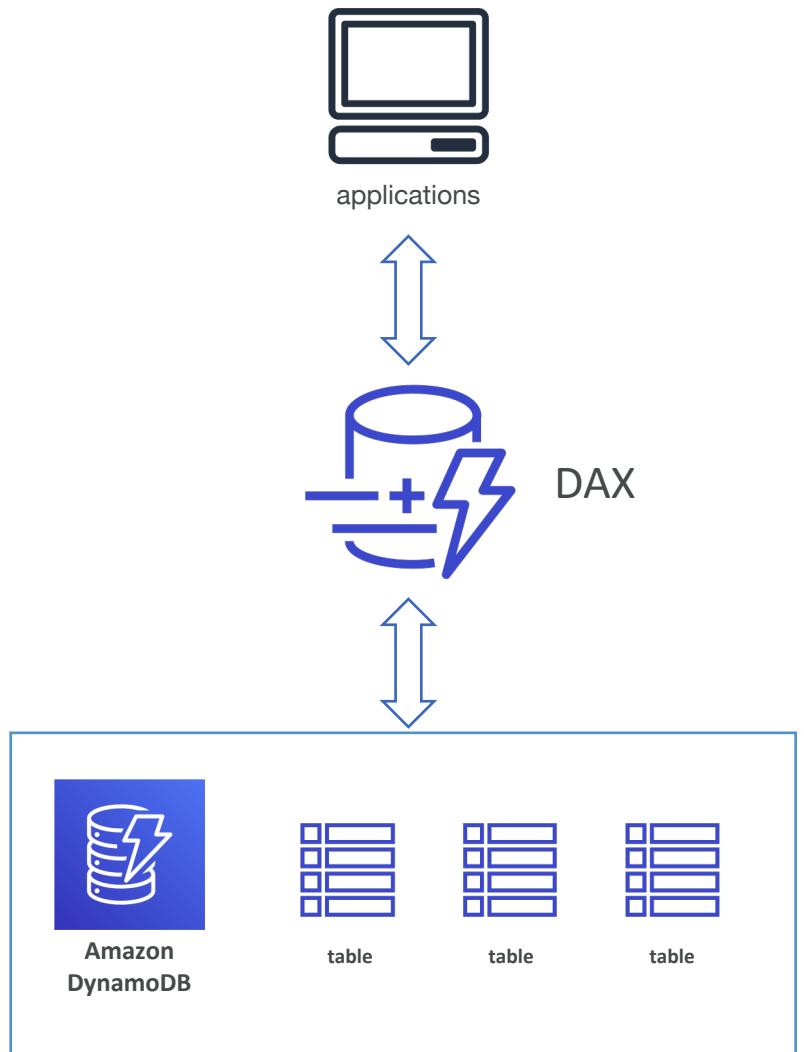


### API for object metadata

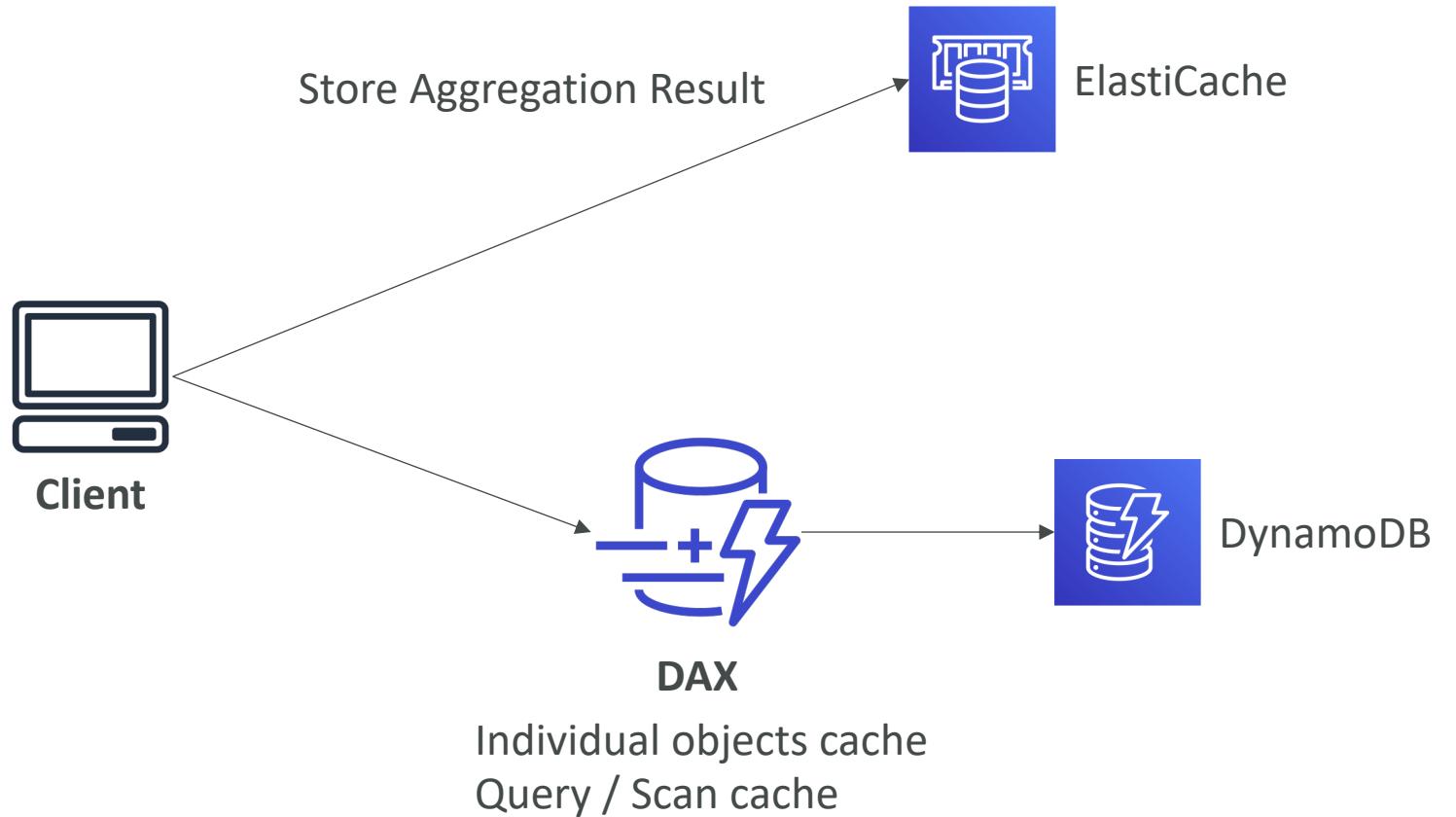
- Search by date
- Total storage used by a customer
- List of all objects with certain attributes
- Find all objects uploaded within a date range

# DynamoDB - DAX

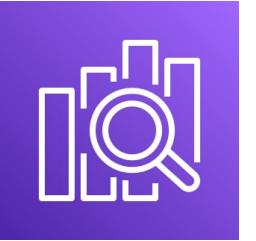
- DAX = DynamoDB Accelerator
- Seamless cache for DynamoDB, no application re-write
- Writes go through DAX to DynamoDB
- Micro second latency for cached reads & queries
- Solves the Hot Key problem (too many reads)
- 5 minutes TTL for cache by default
- Up to 10 nodes in the cluster
- Multi AZ (3 nodes minimum recommended for production)
- Secure (Encryption at rest with KMS,VPC, IAM, CloudTrail...)



# DynamoDB – DAX vs ElastiCache



# Amazon OpenSearch (ex ElasticSearch)



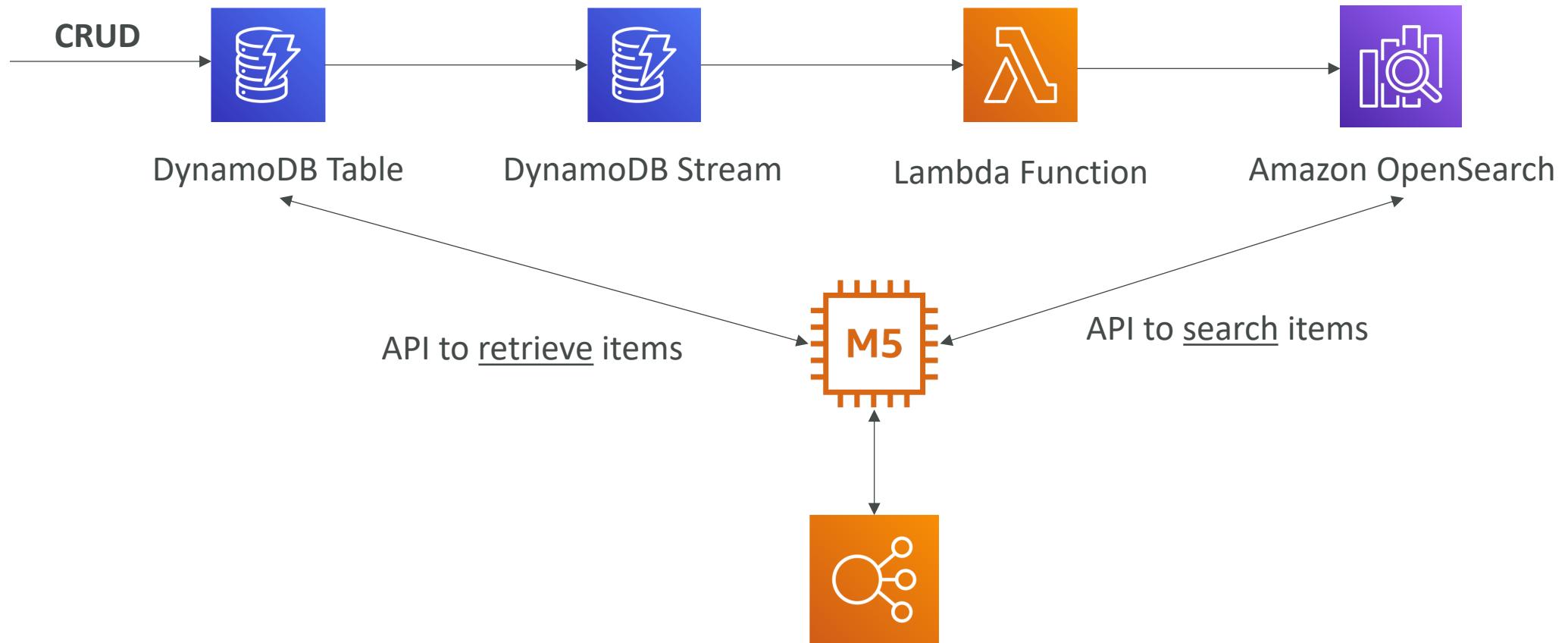
- New name is Amazon OpenSearch
- ElasticSearch => OpenSearch
- Kibana => OpenSearch Dashboards
- Managed version of OpenSearch (open-source project, fork of ElasticSearch)
- Needs to run on servers (not a serverless offering)
- Use cases:
  - Log Analytics
  - Real Time application monitoring
  - Security Analytics
  - Full Text Search
  - Clickstream Analytics
  - Indexing

# OpenSearch + OS Dashboards + Logstash

- OpenSearch (ex ElasticSearch): provide search and indexing capability
  - You must specify instance types, multi-AZ, etc
- OpenSearch Dashboards (ex Kibana):
  - Provide real-time dashboards on top of the data that sits in OpenSearch
  - Alternative to CloudWatch dashboards (more advanced capabilities)
- Logstash:
  - Log ingestion mechanism, use the “Logstash Agent”
  - Alternative to CloudWatch Logs (you decide on retention and granularity)

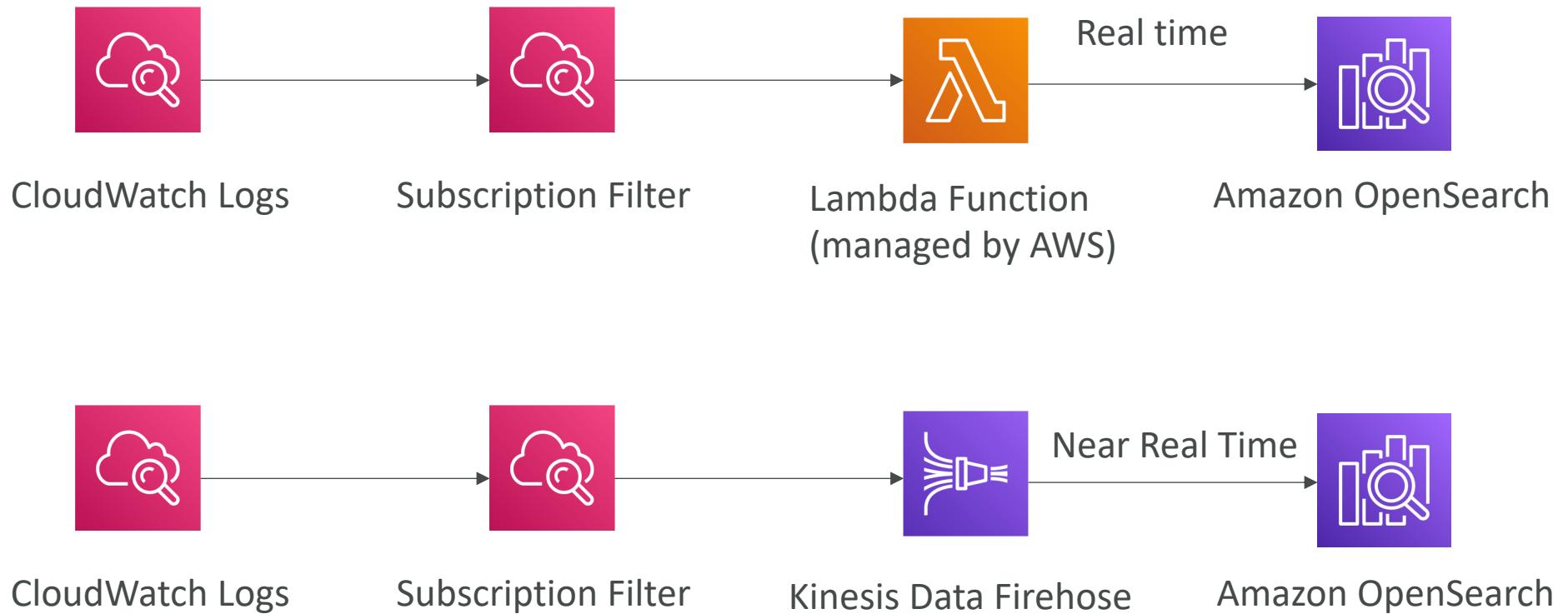
# OpenSearch patterns

## DynamoDB



# OpenSearch patterns

## CloudWatch Logs



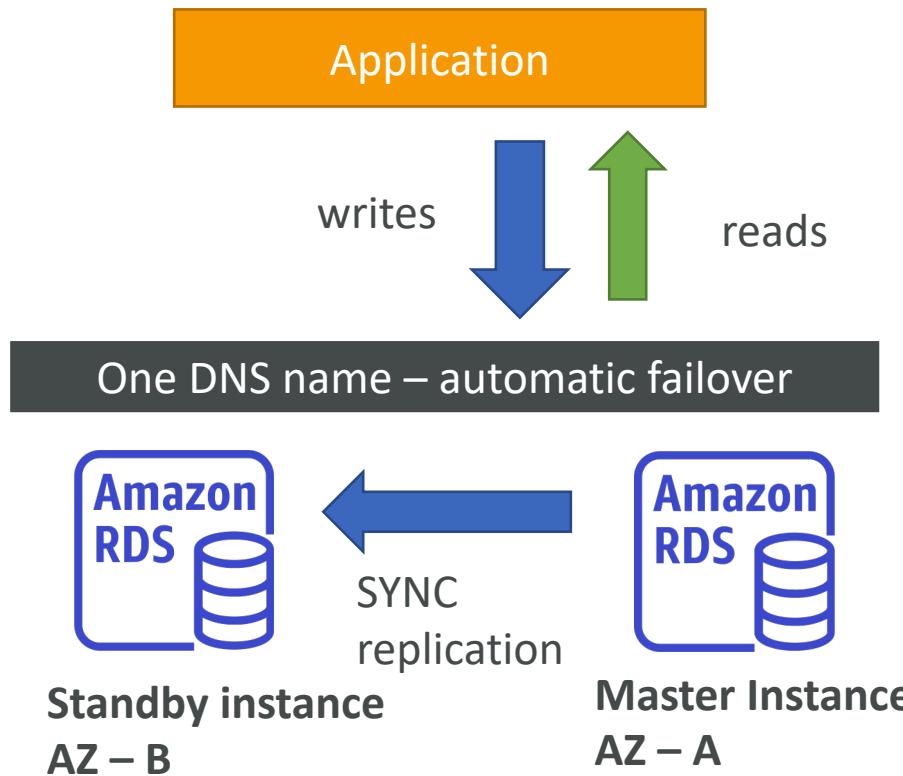
# RDS



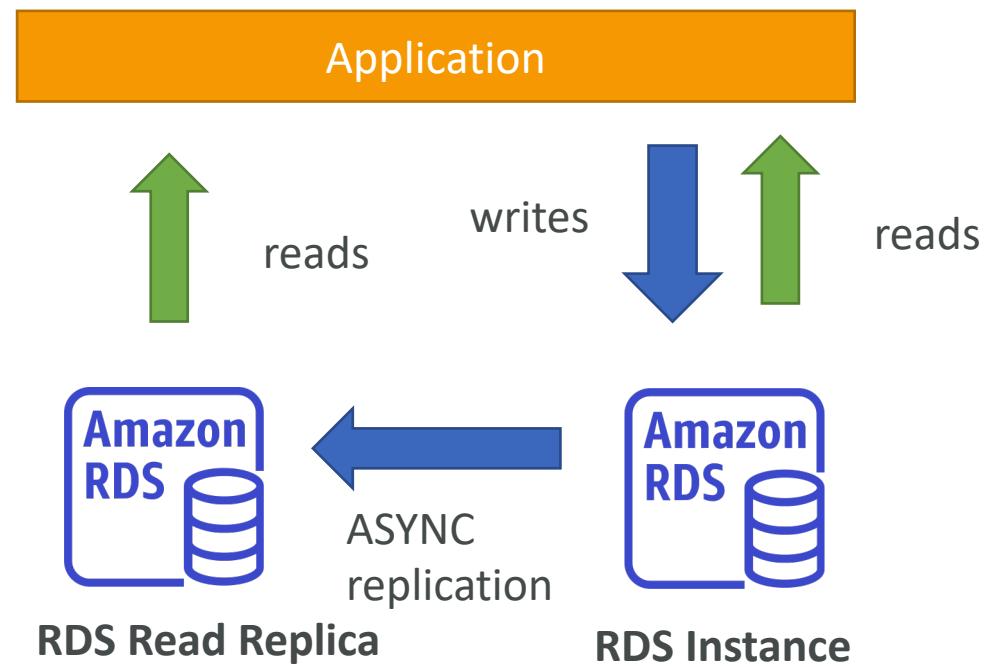
- **Engines:** PostgreSQL, MySQL, MariaDB, Oracle, Microsoft SQL Server
- **Managed DB:** provisioning, backups, patching, monitoring
- **Launched within a VPC,** usually in private subnet, control network access using security groups (important when using Lambda)
- **Storage by EBS** (gp2 or io1), can increase volume size with auto-scaling
- **Backups:** automated with point-in-time recovery. Backups expire
- **Snapshots:** manual, can make copies of snapshots cross region
- **RDS Events:** get notified via SNS for events (operations, outages...)

# RDS – Multi AZ & Read Replicas

- Multi-AZ: Standby instance for failover in case of outage



- Read Replicas: Increase read throughput. Eventual consistency. Can be cross-region



# RDS – Security (reminder)

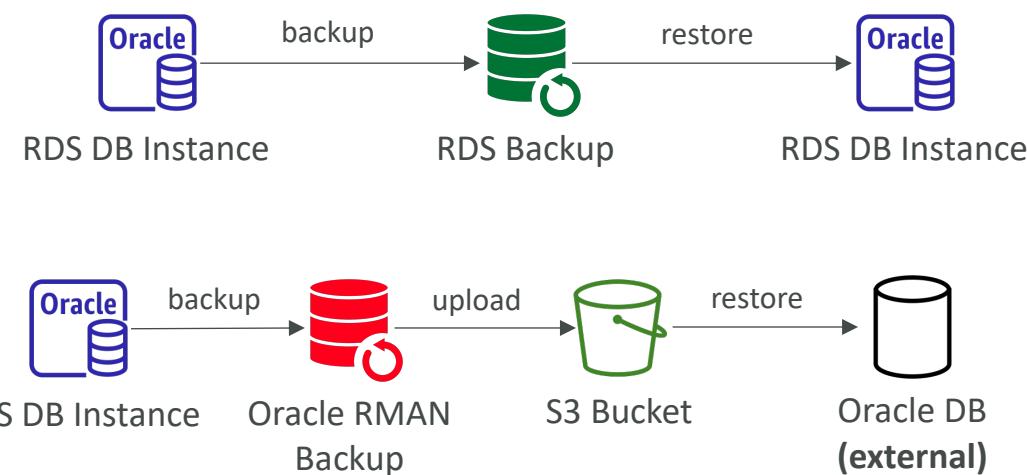


- KMS encryption at rest for underlying EBS volumes / snapshots
- Transparent Data Encryption (TDE) for Oracle and SQL Server
- SSL encryption to RDS is possible for all DB (in-flight)
- IAM authentication for MySQL and PostgreSQL
- Authorization still happens within RDS (not in IAM)
- Can copy an un-encrypted RDS snapshot into an encrypted one
- CloudTrail cannot be used to track queries made within RDS

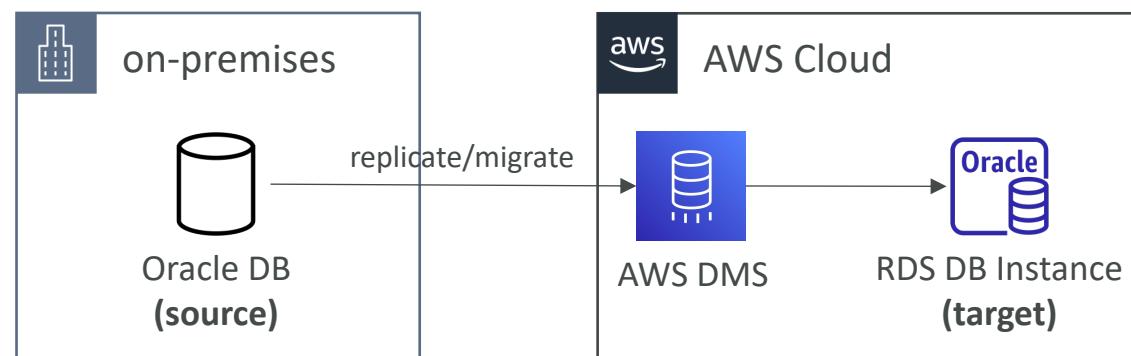
# About RDS for Oracle – Exam Tips

## Backups

- Use RDS Backups for backups & restore to Amazon RDS for Oracle
- Use Oracle RMAN (Recovery Manager) for backups & restore to non-RDS (RDS not supported)

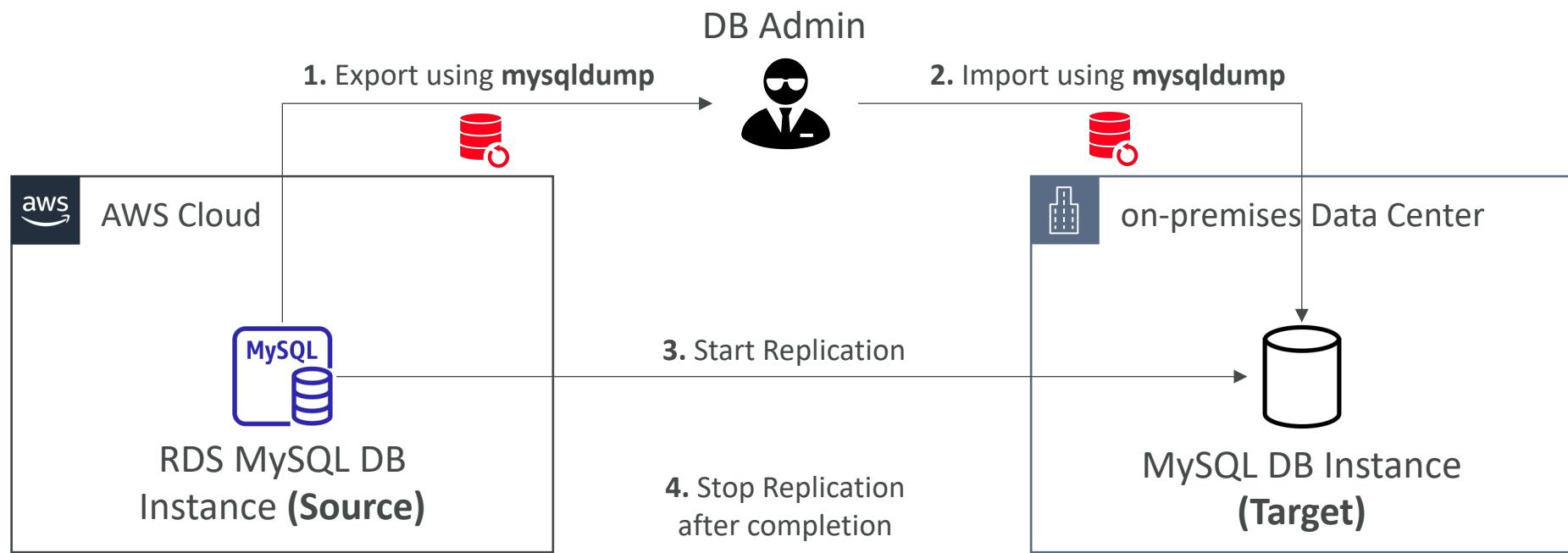


- Real Application Clusters (RAC)
  - RDS for Oracle does NOT support RAC
  - RAC is working on Oracle on EC2 Instances because you have full control
- RDS for Oracle supports Transparent Data Encryption (TDE) to encrypt data before it's written to storage
- DMS works on Oracle RDS



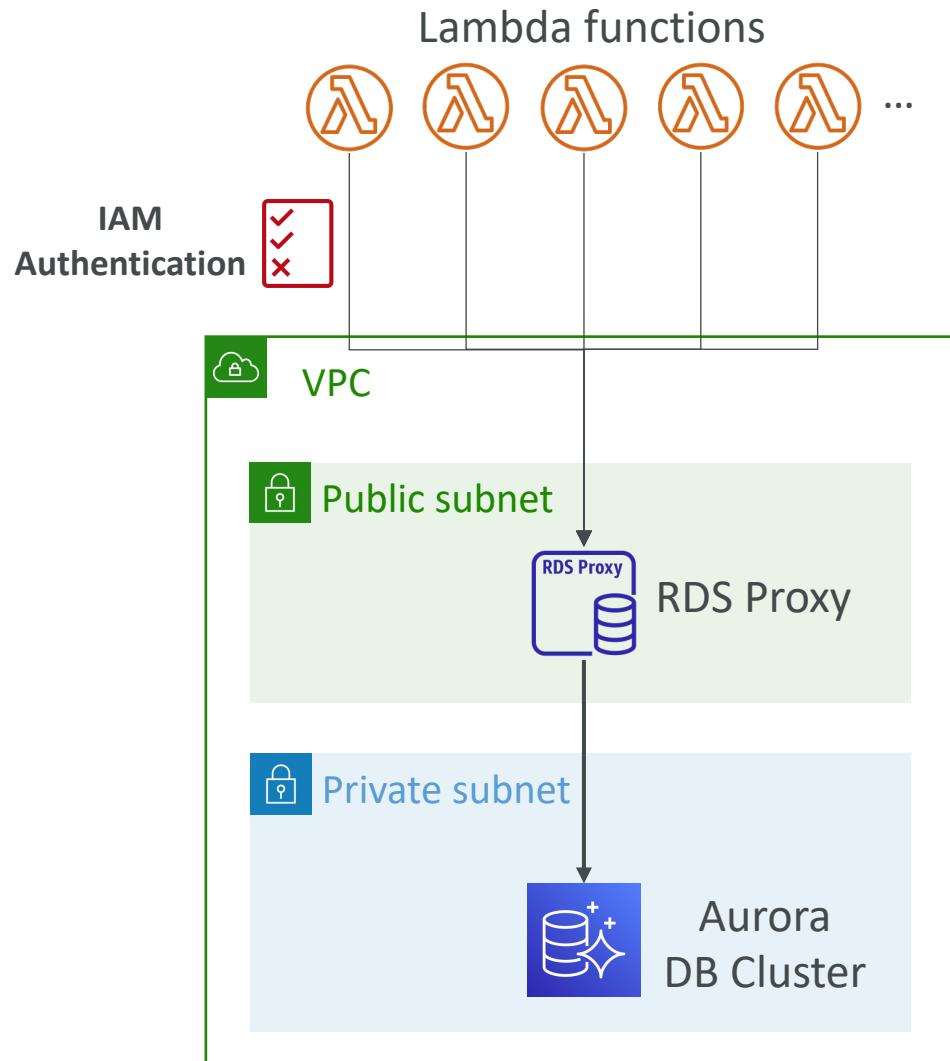
# About RDS for MySQL

- You can use the native `mysqldump` to migrate a MySQL RDS DB to non-RDS
- The external MySQL database can run either on-premises in your data center, or on an Amazon EC2 instance



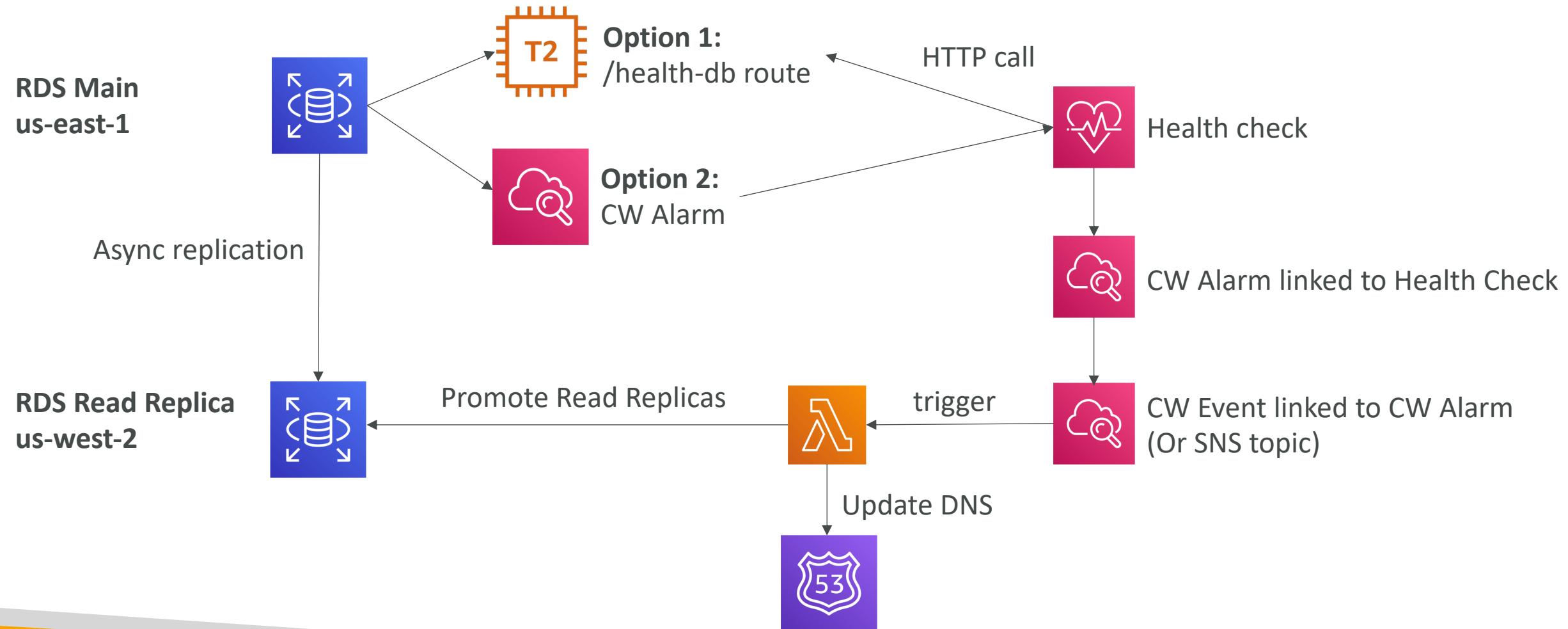
# RDS Proxy for AWS Lambda

- When using Lambda functions with RDS, it opens and maintains a database connection
- This can result in a “TooManyConnections” exception
- With **RDS Proxy**, you no longer need code that handles cleaning up idle connections and managing connection pools
- Supports IAM authentication or DB authentication, auto-scaling
- The Lambda function must have connectivity to the Proxy (public proxy => public Lambda, private proxy => Lambda in VPC)



# RDS Solution Architecture

## Cross Region Failover



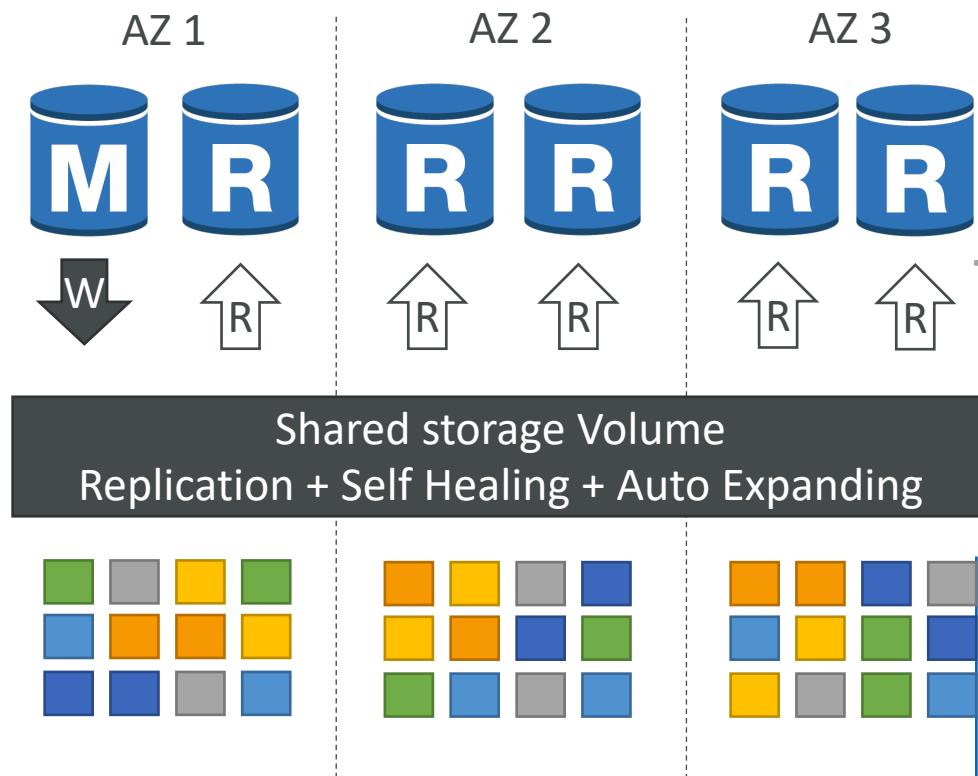
# Aurora



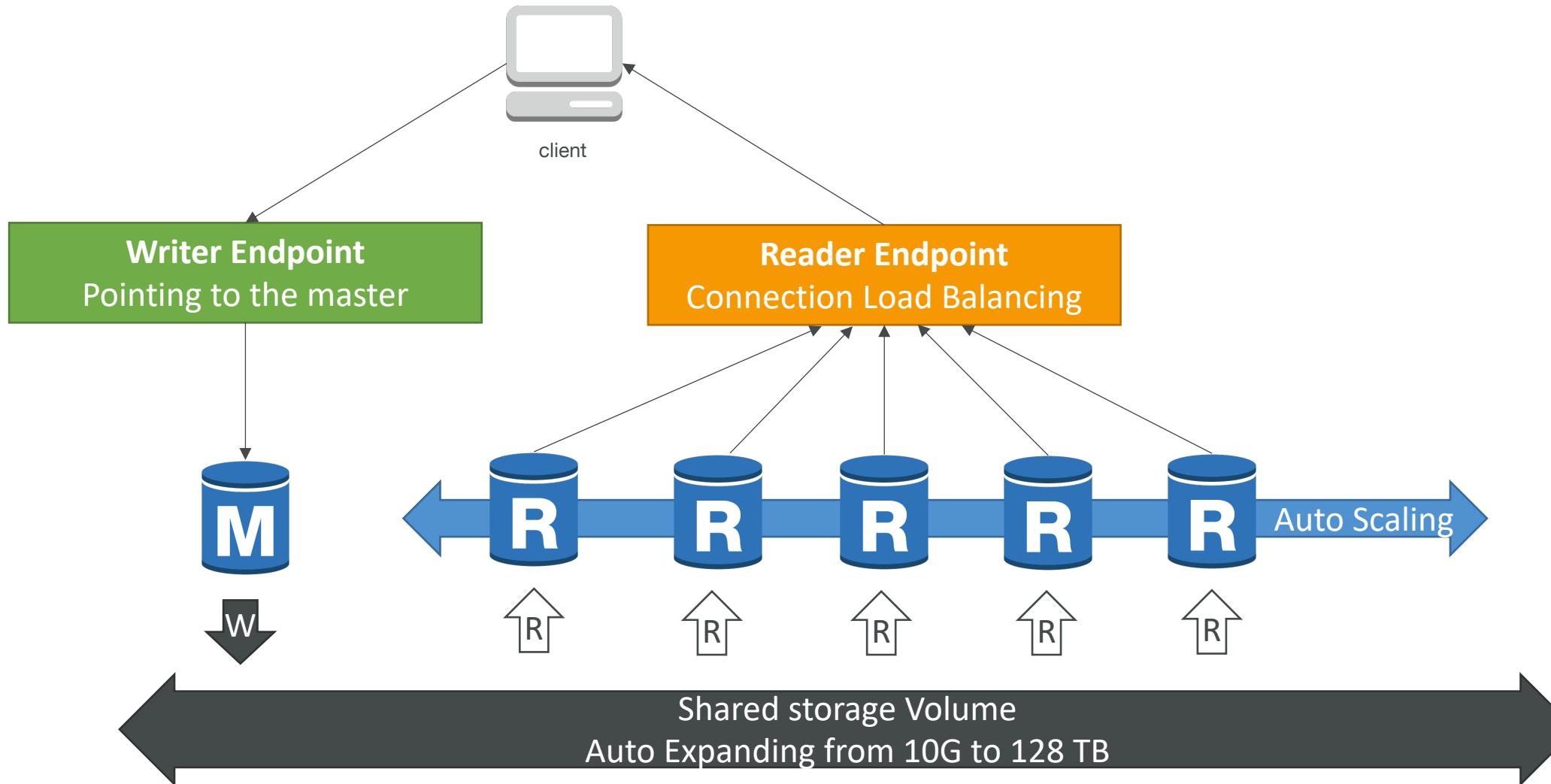
- **DB Engines:** PostgreSQL-compatible & MySQL-compatible
- **Storage:** automatically grows up to 128 TB, 6 copies of data, multi-AZ
- **Read Replicas:** up to 15 RR, reader endpoint to access them all
- **Cross Region RR:** entire database is copied (not select tables)
- **Load / Offload data directly from / to S3:** efficient use of resources
- **Backup, Snapshots & Restore:** same as RDS

# Aurora High Availability and Read Scaling

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

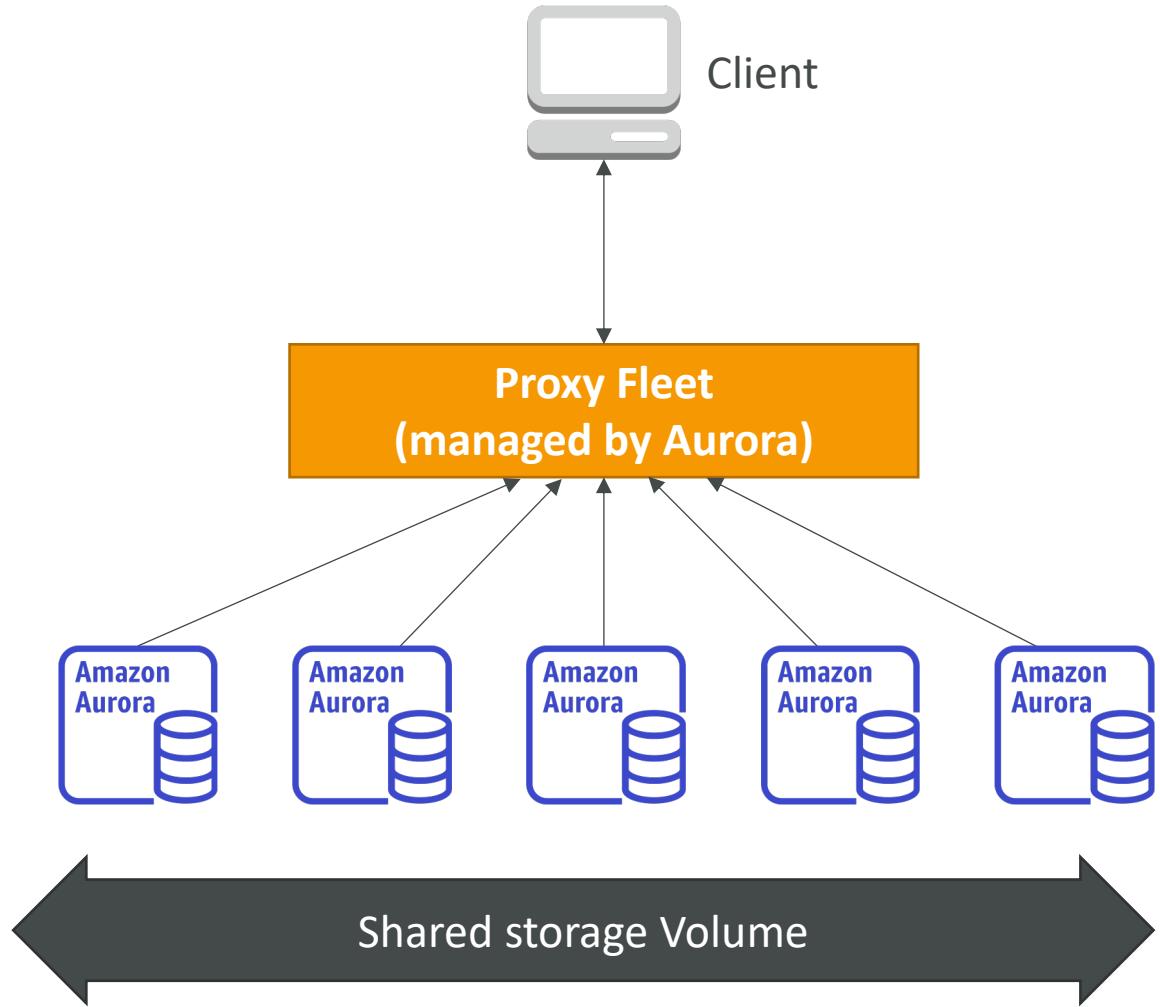


# Aurora DB Cluster



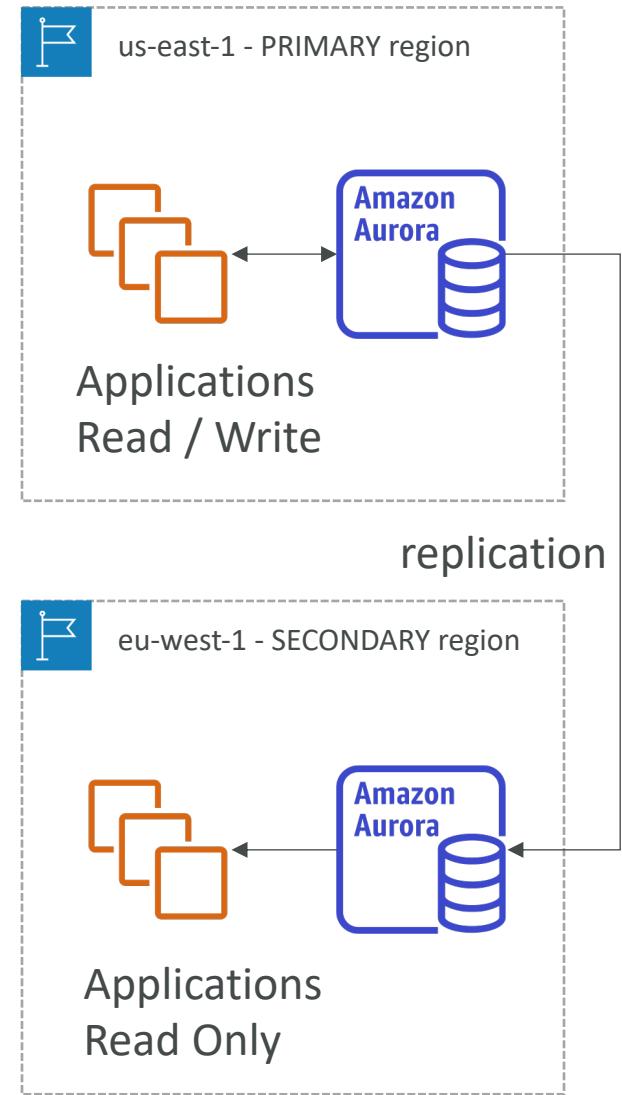
# Aurora Serverless

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



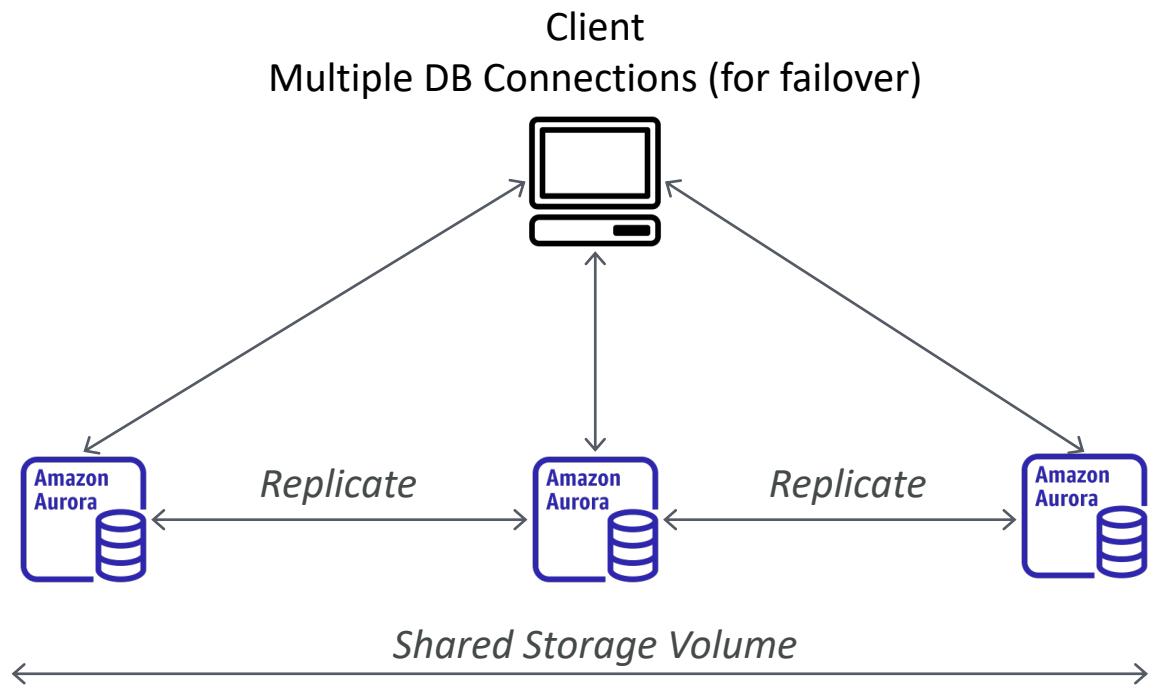
# Global Aurora

- Aurora Cross Region Read Replicas:
  - Useful for disaster recovery
  - Simple to put in place
- Aurora Global Database (recommended):
  - 1 Primary Region (read / write)
  - Up to 5 secondary (read-only) regions, replication lag is less than 1 second
  - Up to 16 Read Replicas per secondary region
  - Helps for decreasing latency
  - Promoting another region (for disaster recovery) has an RTO of < 1 minute



# Aurora Multi-Master

- In case you want immediate failover for write node (HA) –
- Every node does R/W - vs promoting a RR as the new master

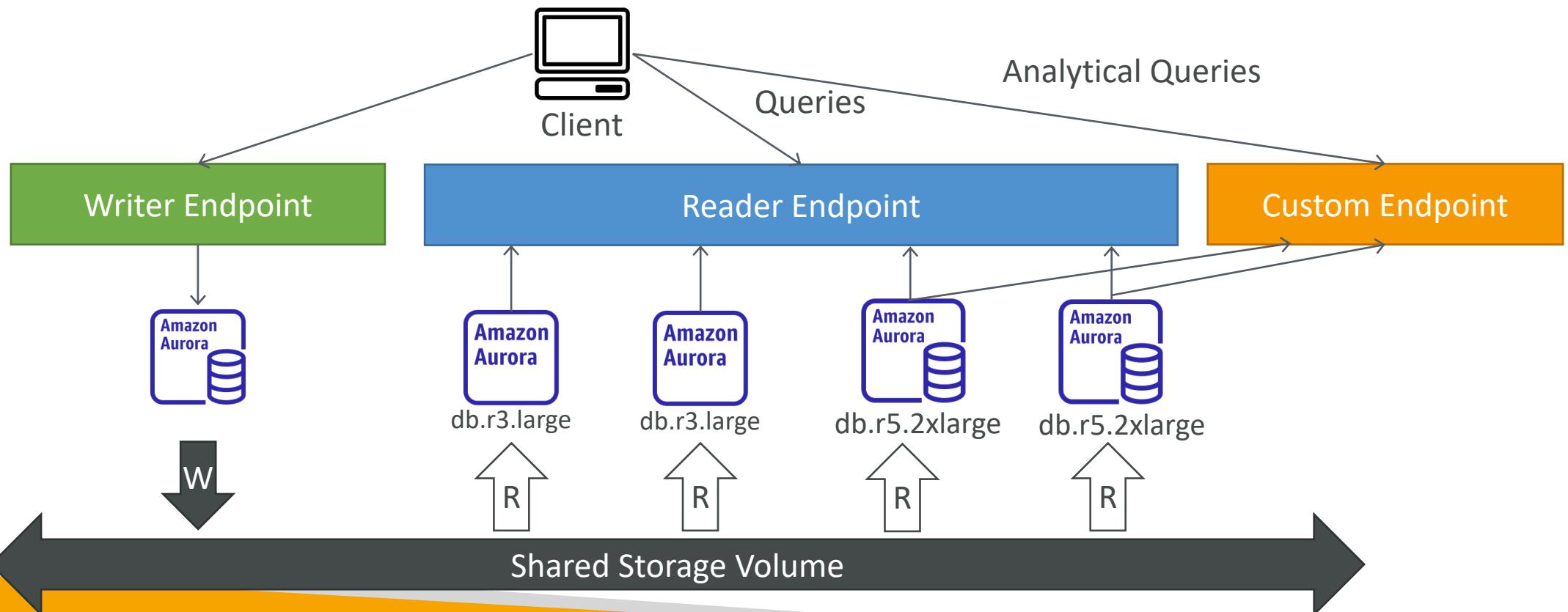


# Aurora Endpoints

- Endpoint = Host Address + Port
- Cluster Endpoint (Writer Endpoint)
  - Connects to the current primary DB instance in the Aurora cluster
  - Used for all write operations in the DB cluster (inserts, updates, deletes, and queries)
- Reader Endpoint
  - Provides load-balancing for read only connections to all Aurora Replicas in the Aurora cluster
  - Used only for read operations (queries)
- Custom Endpoint
  - Represents a set on DB instances that you choose in the Aurora cluster
  - Used when you want to connect to different subsets of DB instances with different capacities and configurations (e.g., different DB parameter group)
- Instance Endpoint
  - Connects to a specific DB instance in the Aurora cluster
  - Used when you want to diagnosis and fine tune a specific DB instance

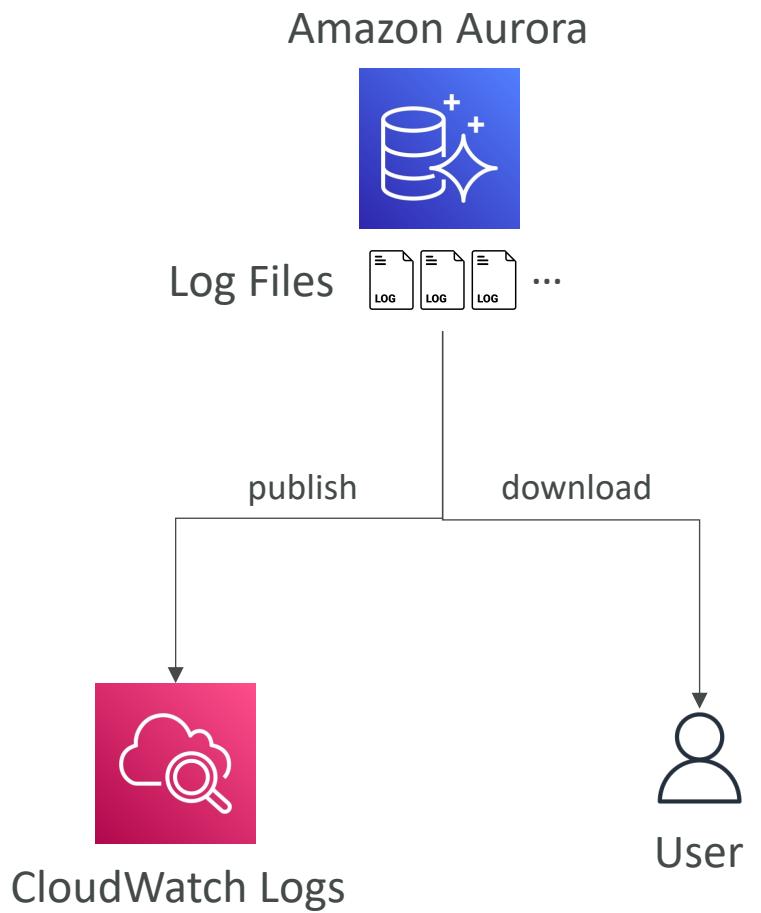
# Aurora – Custom Endpoints

- Define a subset of Aurora Instances as a Custom Endpoint
- Example: Run analytical queries on specific replicas
- The Reader Endpoint is generally not used after defining Custom Endpoints



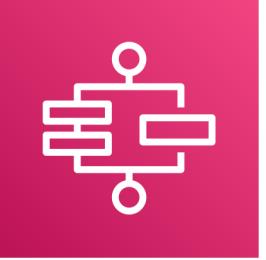
# Aurora Logs

- You can monitor the following types of Aurora MySQL log files:
  - Error log
  - Slow query log
  - General log
  - The audit log
- These log files are either downloaded or published to CloudWatch Logs



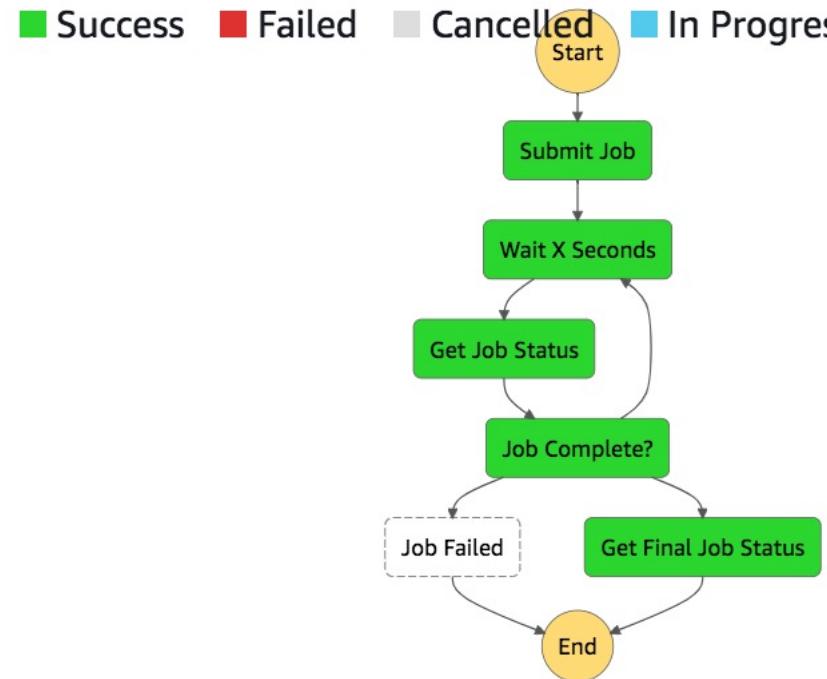
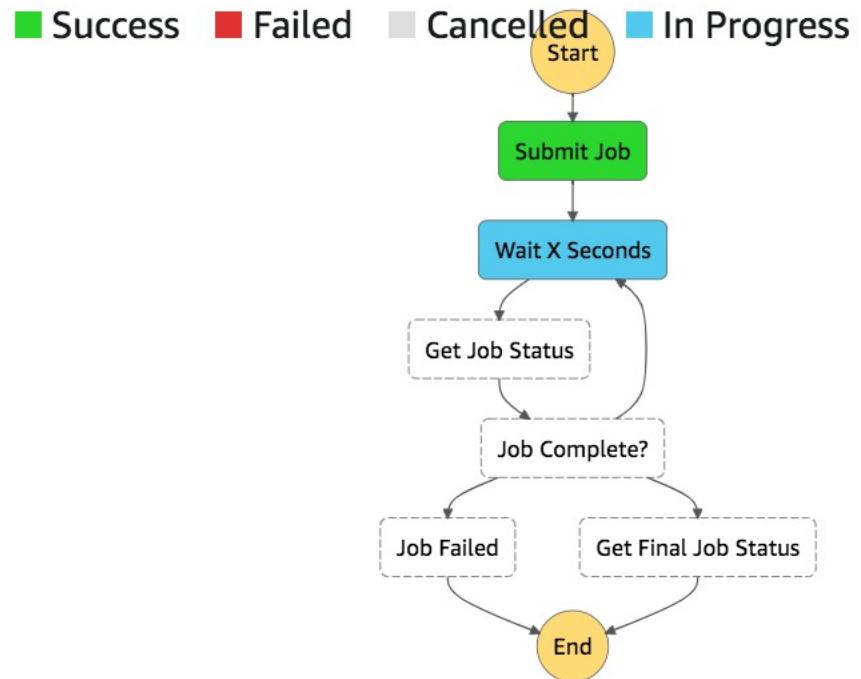
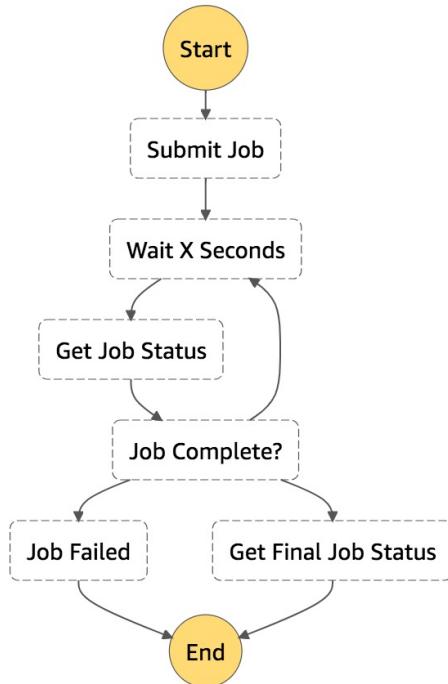
# Service Communications Section

# AWS Step Functions



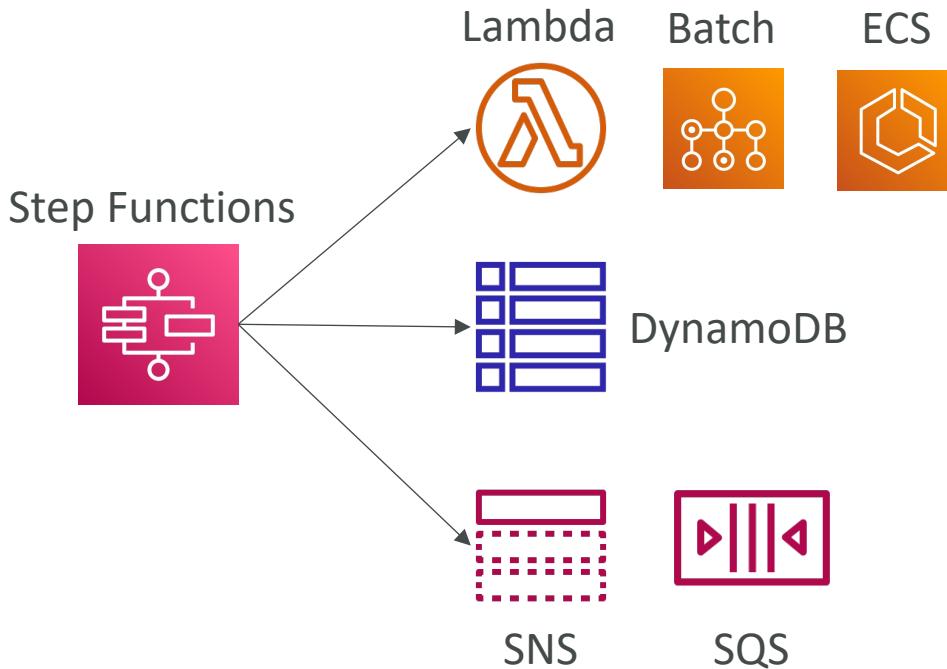
- Build serverless visual workflow to orchestrate your Lambda functions
  - Represent flow as a **JSON state machine**
  - Features: sequence, parallel, conditions, timeouts, error handling...
  - Maximum execution time of 1 year
  - Possibility to implement human approval feature
- 
- If you chain Lambda functions using Step Functions, be mindful of the added latency to pass the calls.

# Visual workflow in Step Functions



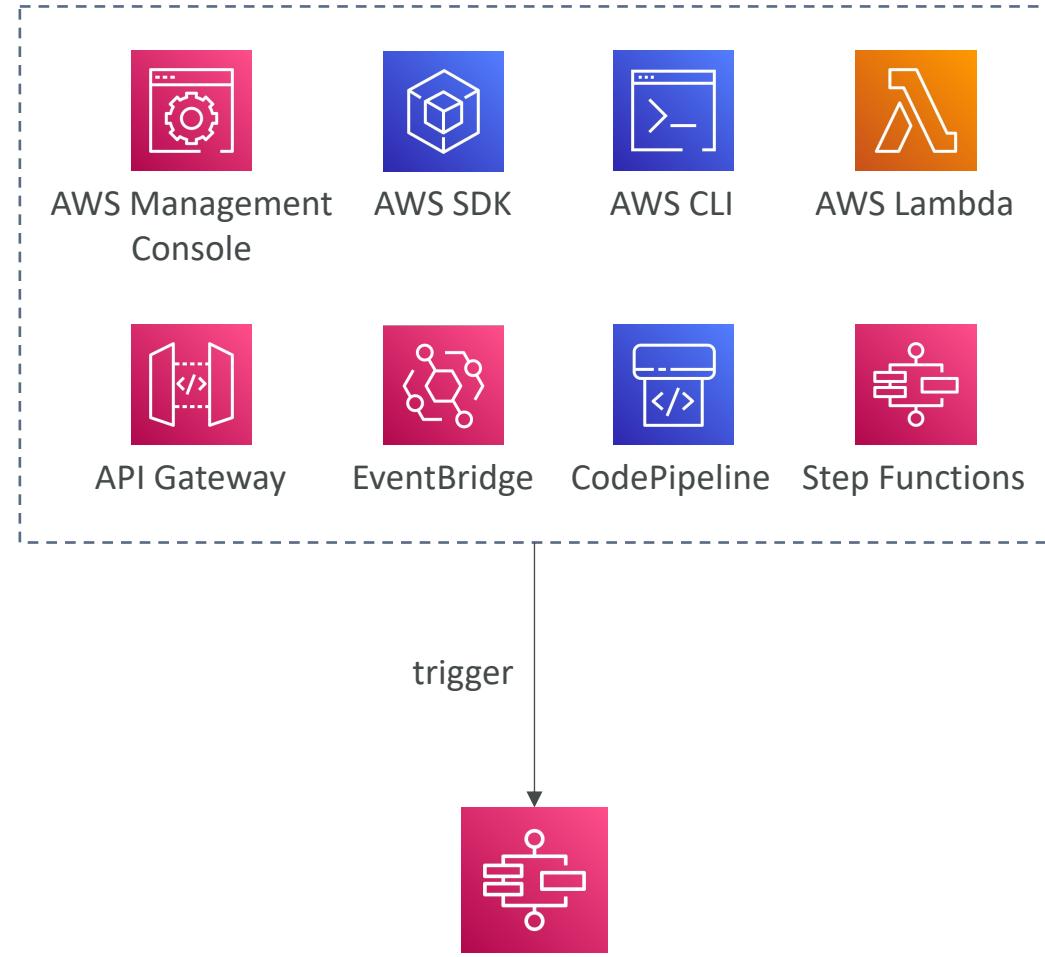
# Step Function Integrations

- Optimized Integrations
  - Can invoke a Lambda function
  - Run an AWS Batch job
  - Run an ECS task and wait for it to complete
  - Insert an item from DynamoDB
  - Publish message to SNS, SQS
  - Launch an EMR, Glue, or SageMaker jobs
  - Launch another Step Function workflow...
- AWS SDK Integrations
  - Access 200+ AWS services from your State Machine
  - Works like standard AWS SDK API call



# Step Functions Workflow Triggers

- You can invoke a Step Function Workflow (State Machine) using:
  - AWS Management Console
  - AWS SDK (`StartExecution` API call)
  - AWS CLI (`start-execution`)
  - AWS Lambda (`StartExecution` API call)
  - API Gateway
  - EventBridge
  - CodePipeline
  - Step Functions



# Step Functions – Sample Projects

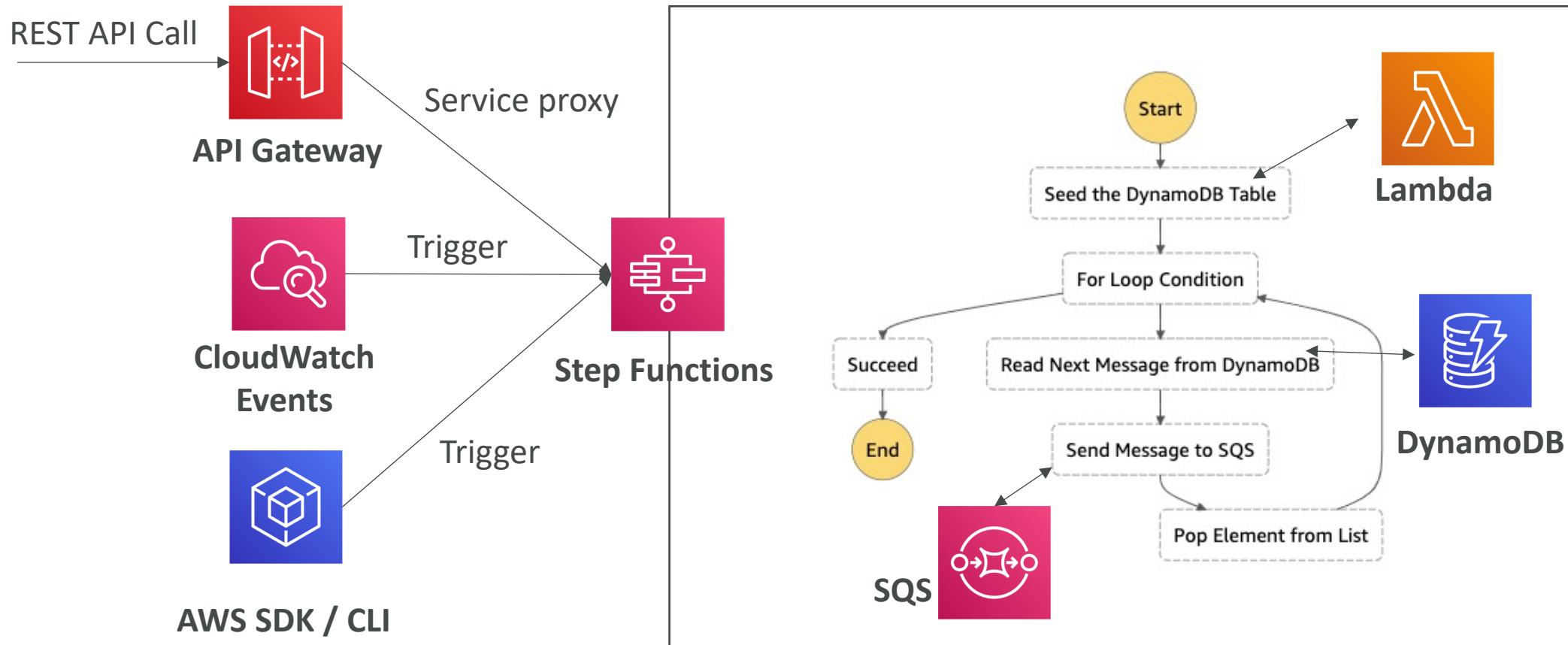
- <https://console.aws.amazon.com/states/home?region=us-east-1#/sampleProjects>

<ul style="list-style-type: none"><li>○ Process high-volume messages from SQS <span style="border: 1px solid green; border-radius: 50%; padding: 2px;">New</span> Use an Express state machine to process SQS messages at a very high rate. (Amazon SQS, AWS Lambda)</li></ul>	<ul style="list-style-type: none"><li>○ Selective checkpointing example <span style="border: 1px solid green; border-radius: 50%; padding: 2px;">New</span> Combine Standard and Express state machines to run an e-commerce workflow that does selective checkpointing. (Amazon SQS, AWS Lambda)</li></ul>	<ul style="list-style-type: none"><li>○ Job Poller Manage an asynchronous job using a serverless polling loop (AWS Lambda, AWS Batch)</li></ul>	<ul style="list-style-type: none"><li>○ Start a workflow within a workflow Combine different workflows together by triggering the execution of one state machine from within another.</li></ul>
<ul style="list-style-type: none"><li>○ Dynamically process data with a Map state Dynamically iterate a series of steps by passing an array to a Map state (AWS Lambda, Amazon SQS, Amazon SNS, Amazon DynamoDB).</li></ul>	<ul style="list-style-type: none"><li>○ Train a machine learning model Train a machine learning model and batch transform a test dataset (Amazon SageMaker, AWS Lambda, Amazon S3)</li></ul>	<ul style="list-style-type: none"><li>○ Tune a machine learning model Tune hyperparameters of a machine learning model and batch transform a test dataset (Amazon SageMaker, AWS Lambda, Amazon S3)</li></ul>	<ul style="list-style-type: none"><li>○ Callback pattern example Send a message to SQS, pausing the workflow until it receives a callback from an external service (in this case Lambda) with success or failure (Amazon SQS, Amazon SNS, AWS Lambda)</li></ul>
<ul style="list-style-type: none"><li><b>○ Manage a batch job</b> Trigger a notification on success or failure of a batch job (AWS Batch, Amazon SNS)</li></ul>	<ul style="list-style-type: none"><li>○ Manage a container task Trigger a notification on success or failure of a container task (AWS Fargate, Amazon SNS)</li></ul>	<ul style="list-style-type: none"><li>○ Transfer data records Iteratively read items from a database table and send each item as messages to a queue (Amazon DynamoDB, Amazon SQS)</li></ul>	<ul style="list-style-type: none"><li>○ Task Timer Trigger a notification after a certain period of time (Amazon SNS)</li></ul>
<ul style="list-style-type: none"><li>○ Manage an EMR job Create an EMR cluster, add multiple steps and run them, then terminate the cluster (Amazon EMR)</li></ul>			

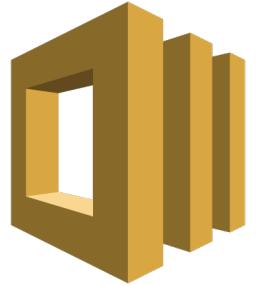
# Step Functions – Tasks

- Lambda Tasks:
  - Invoke a Lambda function
- Activity Tasks:
  - Activity worker (HTTP), EC2 Instances, mobile device, on premise DC
  - They poll the Step functions service
- Service Tasks:
  - Connect to a supported AWS service
  - Lambda function, ECS Task, Fargate, DynamoDB, Batch job, SNS topic, SQS queue
- Wait Task:
  - To wait for a duration or until a timestamp
- Note: Step Functions does not integrate natively with AWS Mechanical Turk

# Step Functions – Solution Architecture



# AWS SWF – Simple Workflow Service



- Coordinate work amongst applications
- Code runs on EC2 (not serverless)
- 1 year max runtime
- Concept of “activity step” and “decision step”
- Has built-in “human intervention” step
- Example: order fulfilment from web to warehouse to delivery
- **Step Functions is recommended to be used for new applications, except:**
  - If you need external signals to intervene in the processes
  - If you need child processes that return values to parent processes
  - If you need to use Amazon Mechanical Turk

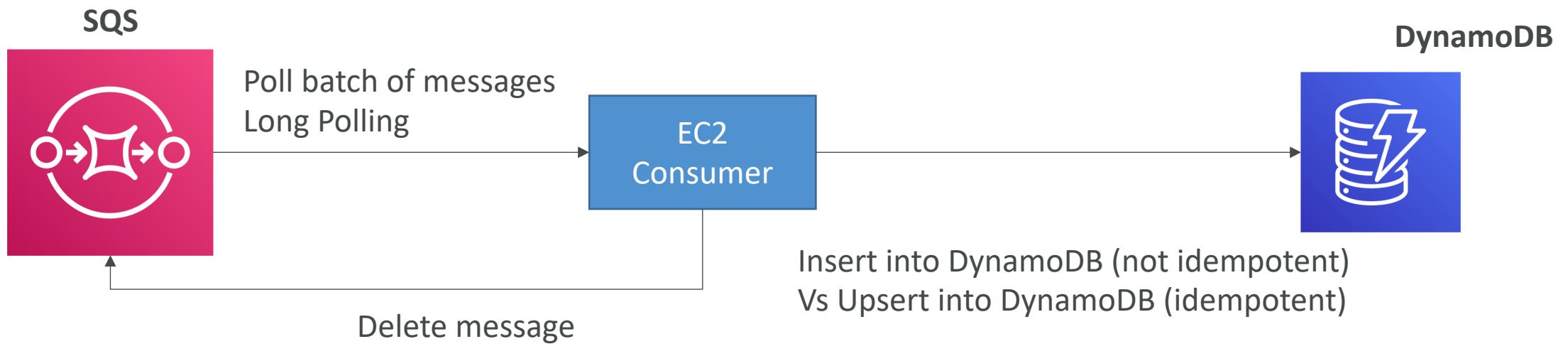
# SQS



- Serverless, managed queue, integrated with IAM
- Can handle extreme scale, no provisioning required
- Used to **decouple** services
- Message size of max 256 KB (use a pointer to S3 for large messages)
- Can be read from EC2 (optional ASG), Lambda
- SQS could be used as a write buffer for DynamoDB
- **SQS FIFO:**
  - receive messages in order they were sent
  - 300 messages/s without batching, 3000 /s with batching

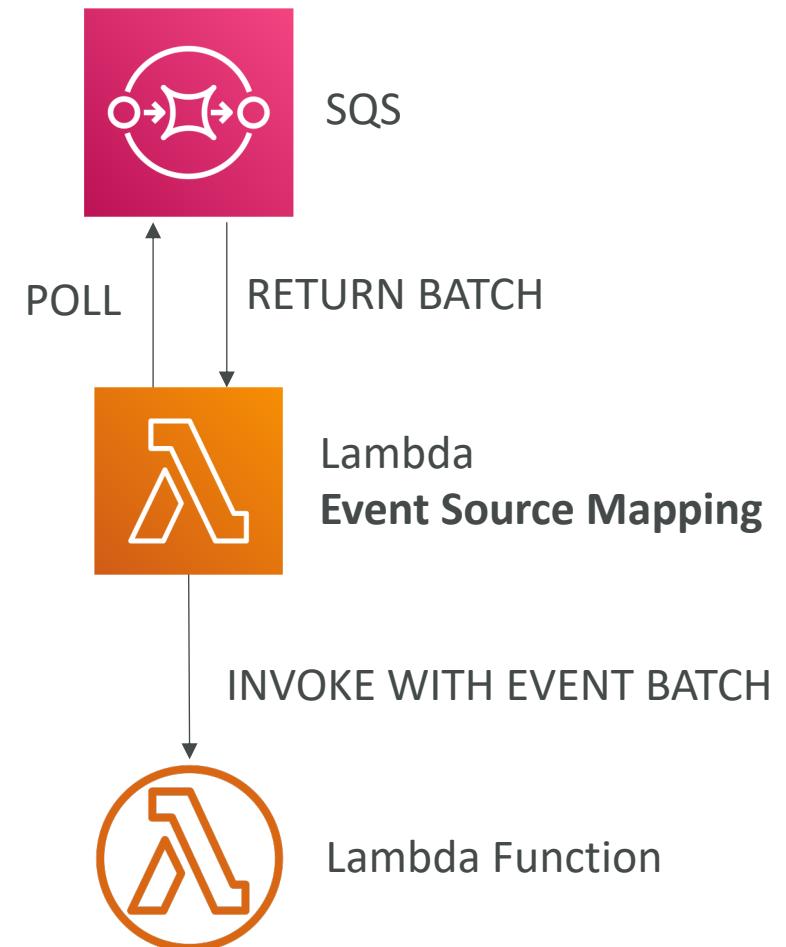
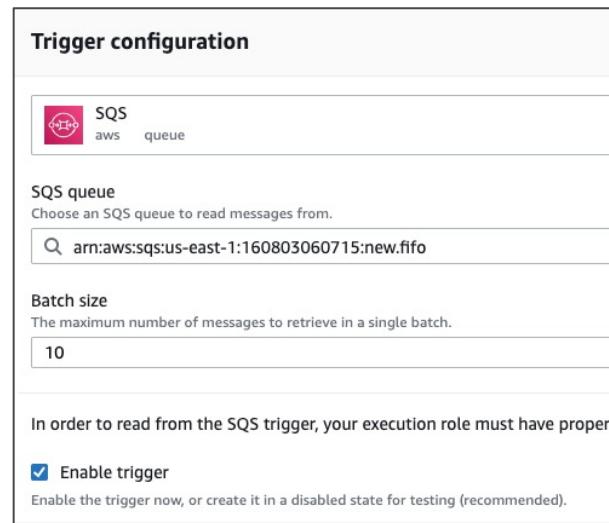
# SQS – Solution Architecture Idempotency

- Messages can be processed twice by consumer (in case of failures, timeouts, etc)
- To hedge against that problem, implement **idempotency** at the consumer level
- Means the same action done twice by the consumer won't duplicate the effect



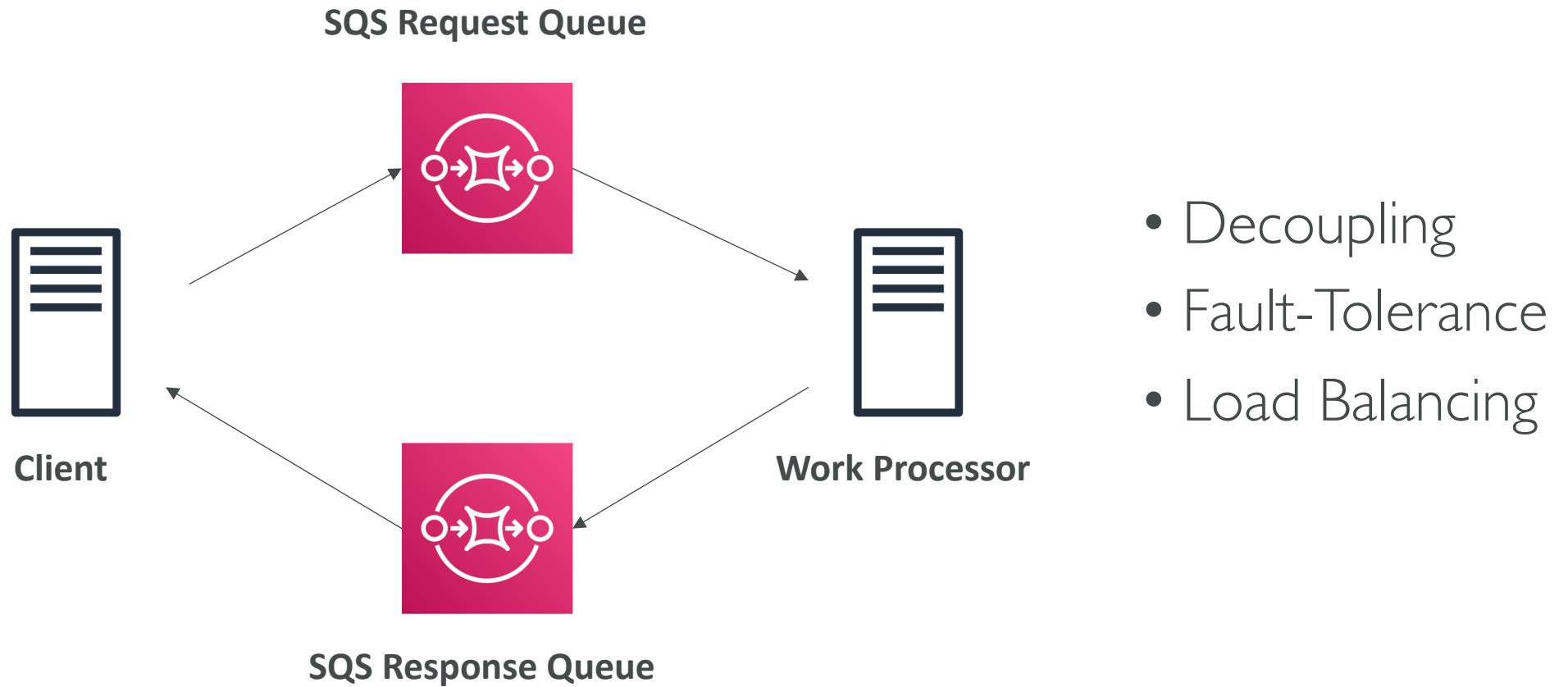
# Lambda – Event Source Mapping SQS & SQS FIFO

- Event Source Mapping will poll SQS (Long Polling)
- Specify batch size (1-10 messages)
- Recommended: Set the queue visibility timeout to 6x the timeout of your Lambda function
- To use a DLQ
  - set-up on the SQS queue, not Lambda (DLQ for Lambda is only for async invocations)
  - Or use a Lambda destination for failures

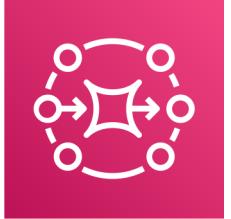


# SQS - Solution Architecture

## Request / Response queue (async)



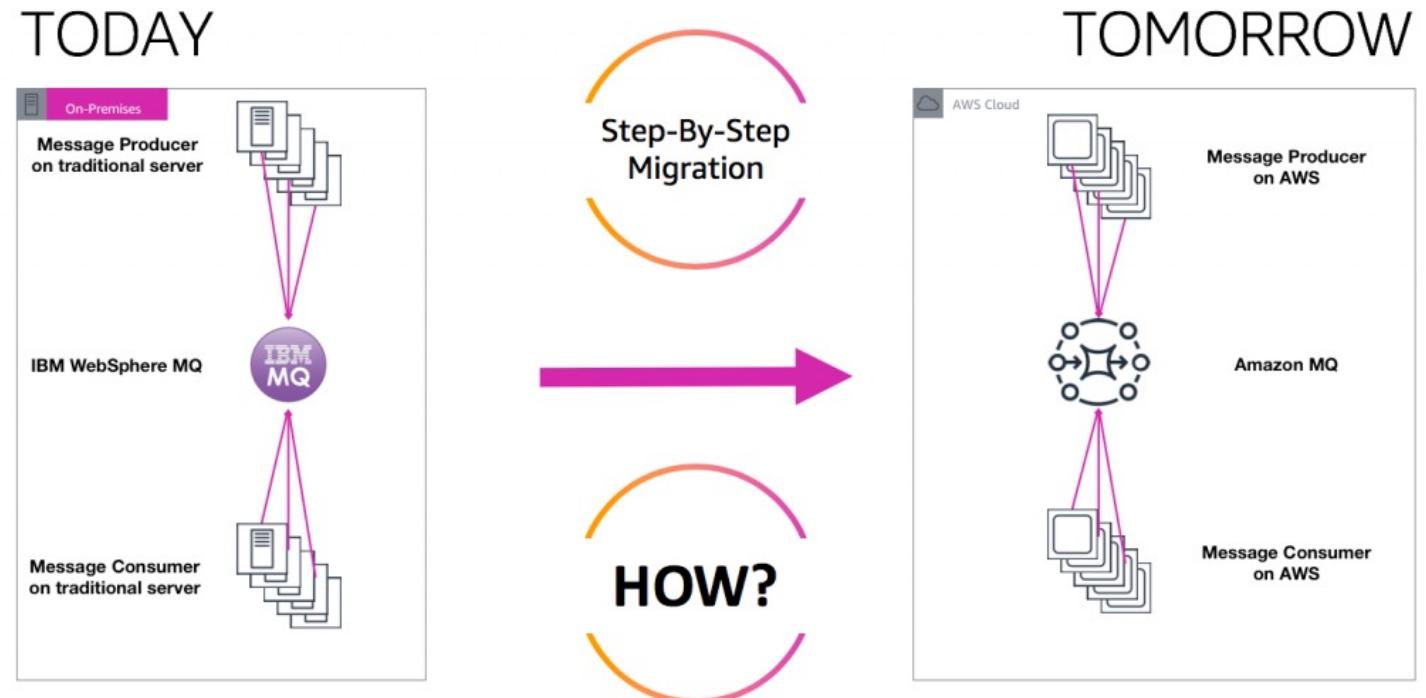
# Amazon MQ



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

# Amazon MQ – Re-platform

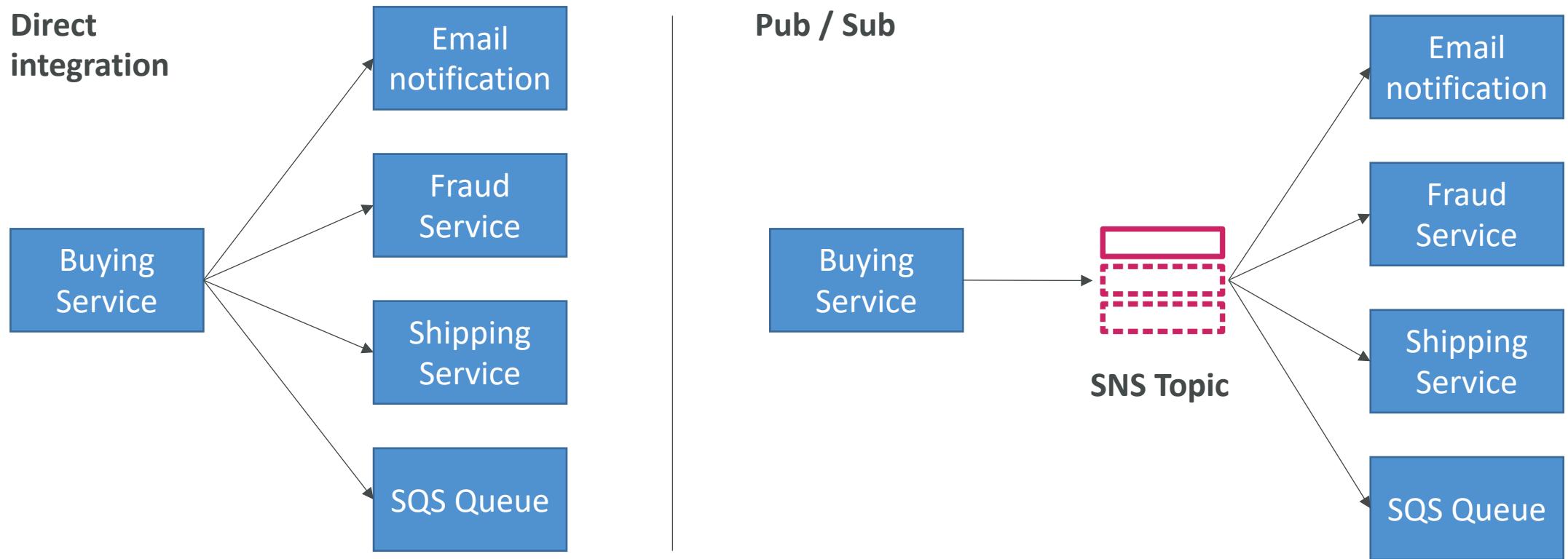
- IBM MQ, TIBCO EMS, Rabbit MQ, and Apache ActiveMQ can be migrated to Amazon MQ



<https://aws.amazon.com/blogs/compute/migrating-from-ibm-mq-to-amazon-mq-using-a-phased-approach/>

# Amazon SNS

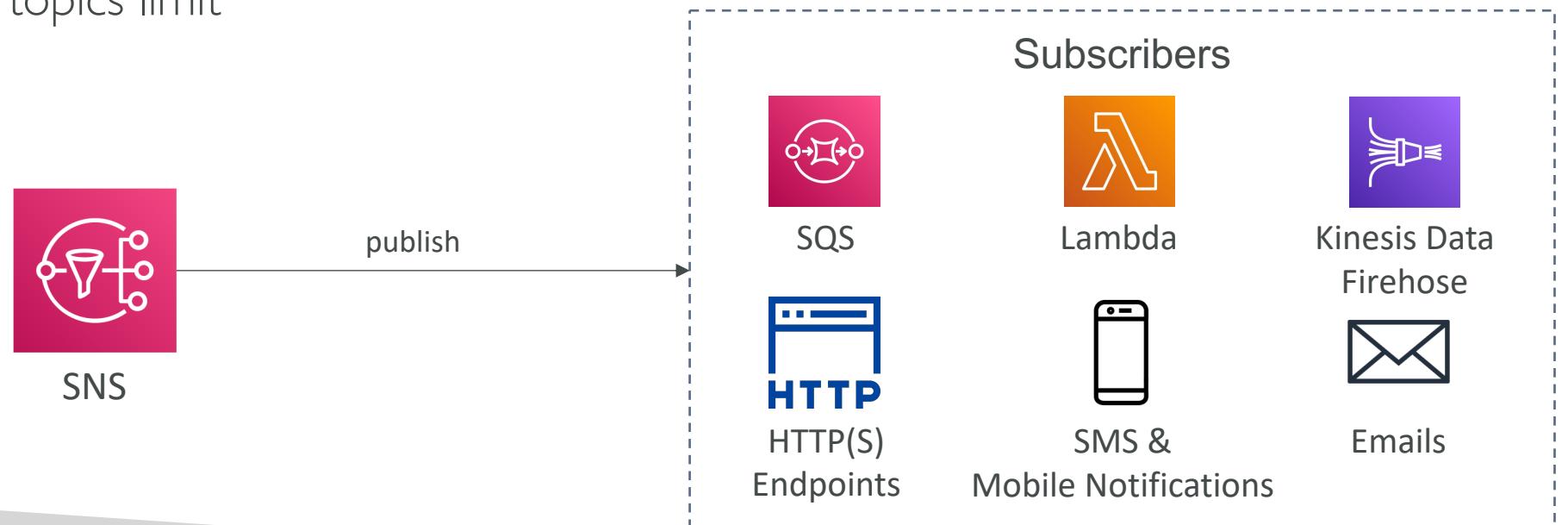
- What if you want to send one message to many receivers?



# Amazon SNS

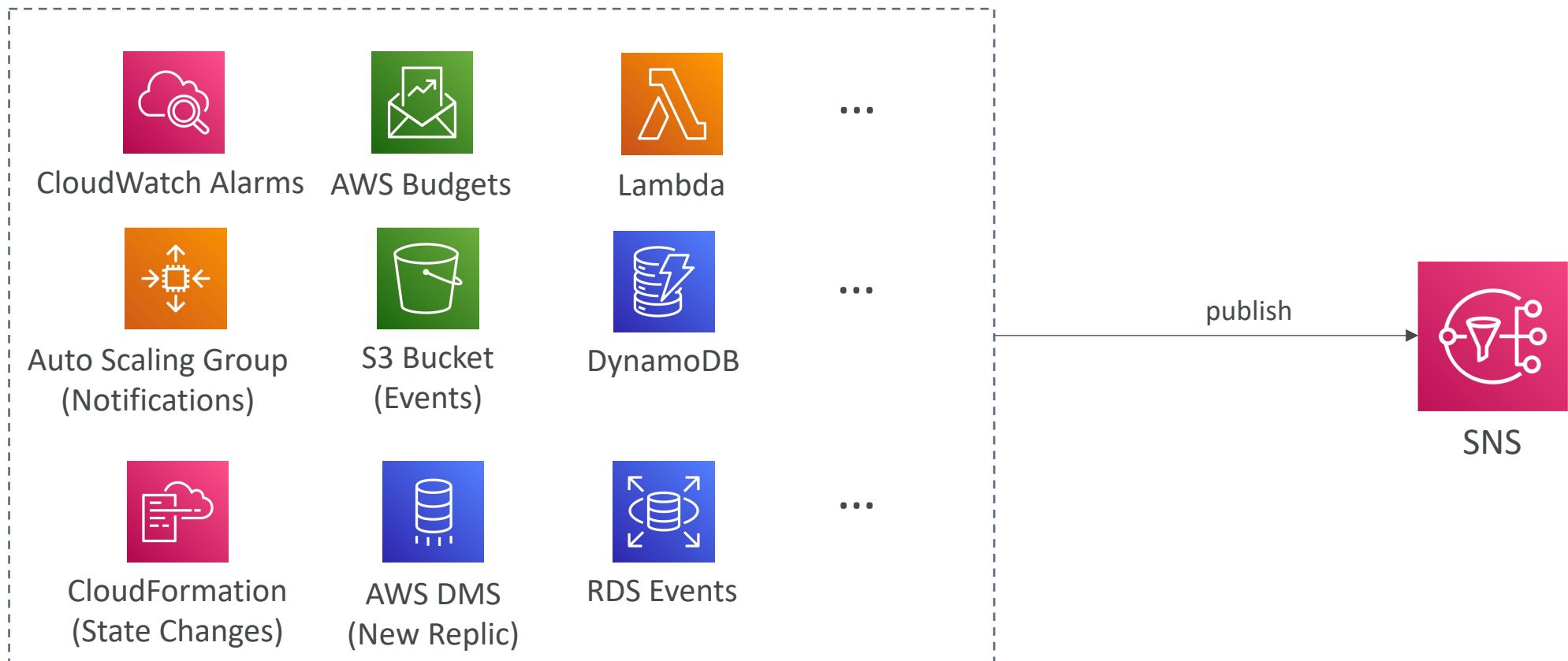


- The “event producer” only sends message to one SNS topic
- As many “event receivers” (subscriptions) as we want to listen to the SNS topic notifications
- Each subscriber to the topic will get all the messages (note: new feature to filter messages)
- Up to 12,500,000 subscriptions per topic
- 100,000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



# SNS integrates with a lot of AWS services

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



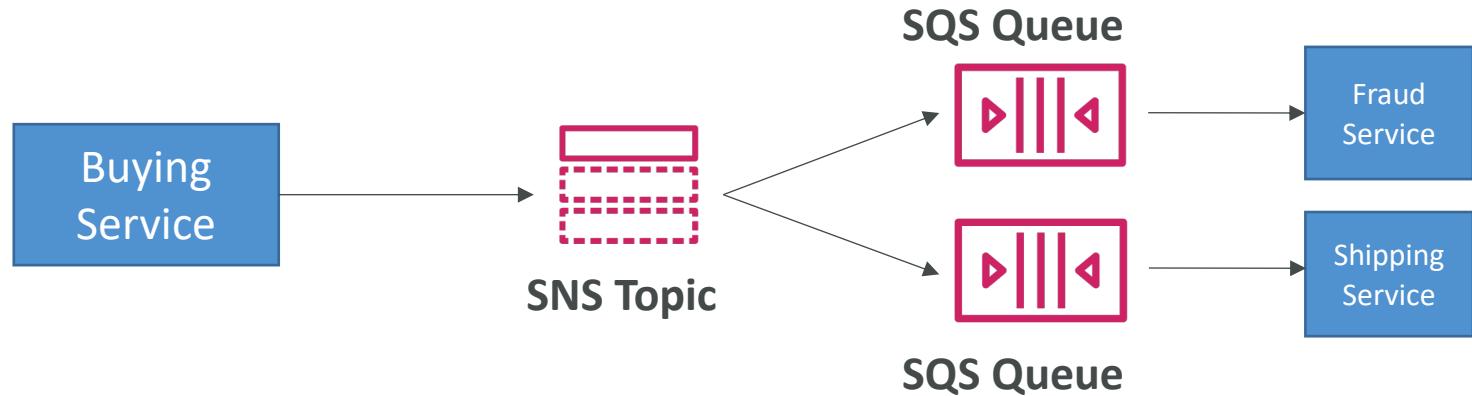
# Amazon SNS – How to publish

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

# Amazon SNS – Security

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

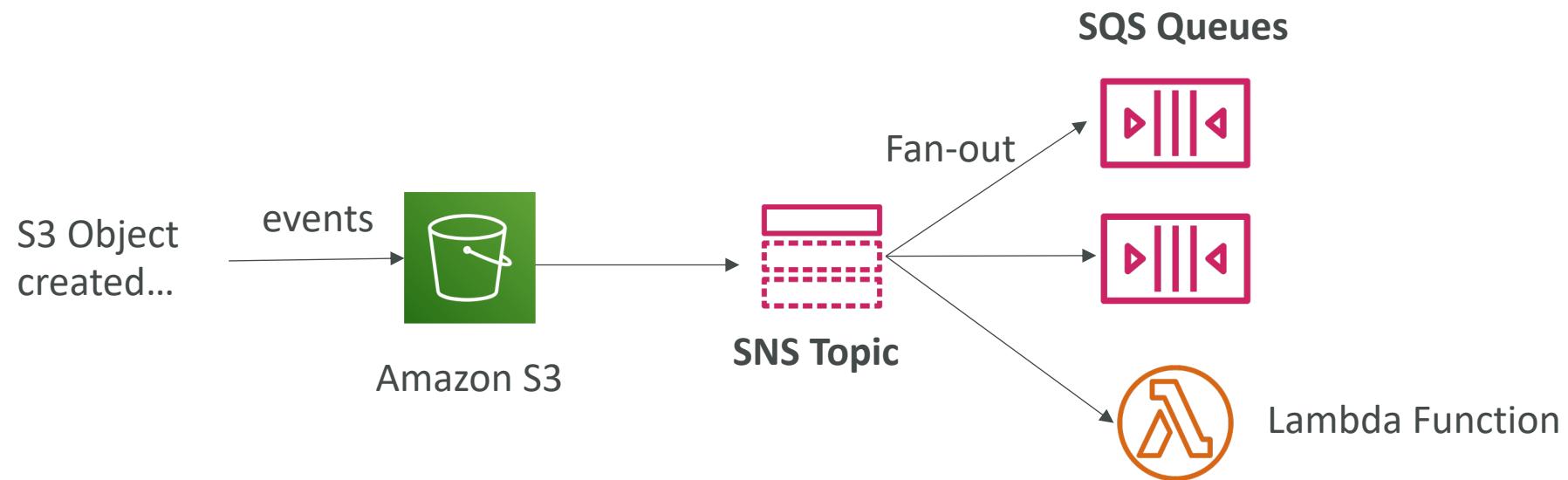
# SNS + SQS: Fan Out



- Push once in SNS, receive in all SQS queues that are subscribers
- Fully decoupled, no data loss
- SQS allows for: data persistence, delayed processing and retries of work
- Ability to add more SQS subscribers over time
- Make sure your SQS queue access policy allows for SNS to write

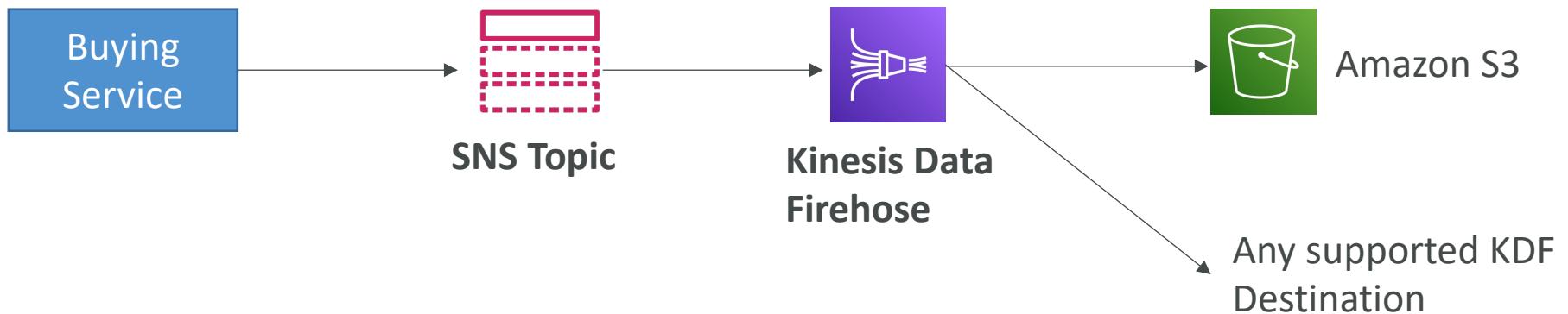
# Application: S3 Events to multiple queues

- For the same combination of: **event type** (e.g. object create) and **prefix** (e.g. images/) you can only have one S3 Event rule
- If you want to send the same S3 event to many SQS queues, use fan-out



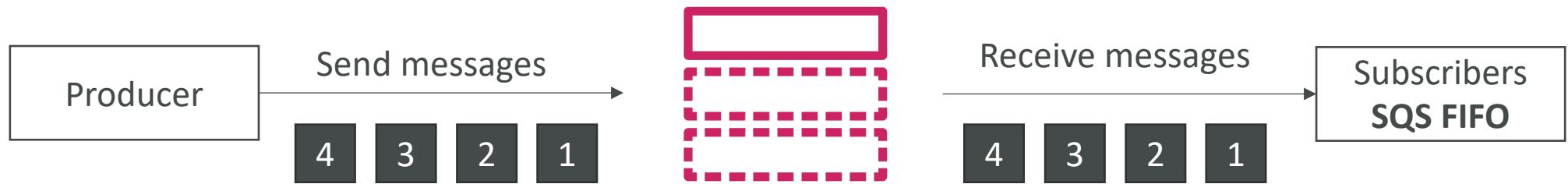
# Application: SNS to Amazon S3 through Kinesis Data Firehose

- SNS can send to Kinesis and therefore we can have the following solutions architecture:



# Amazon SNS – FIFO Topic

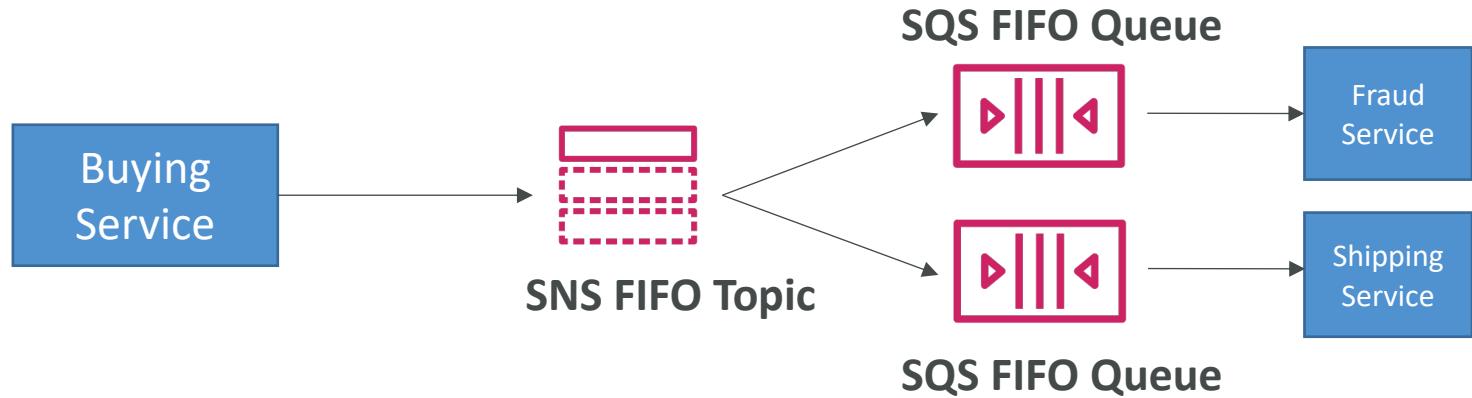
- FIFO = First In First Out (ordering of messages in the topic)



- Similar features as SQS FIFO:
  - Ordering by Message Group ID (all messages in the same group are ordered)
  - Deduplication using a Deduplication ID or Content Based Deduplication
- Can only have SQS FIFO queues as subscribers
- Limited throughput (same throughput as SQS FIFO)

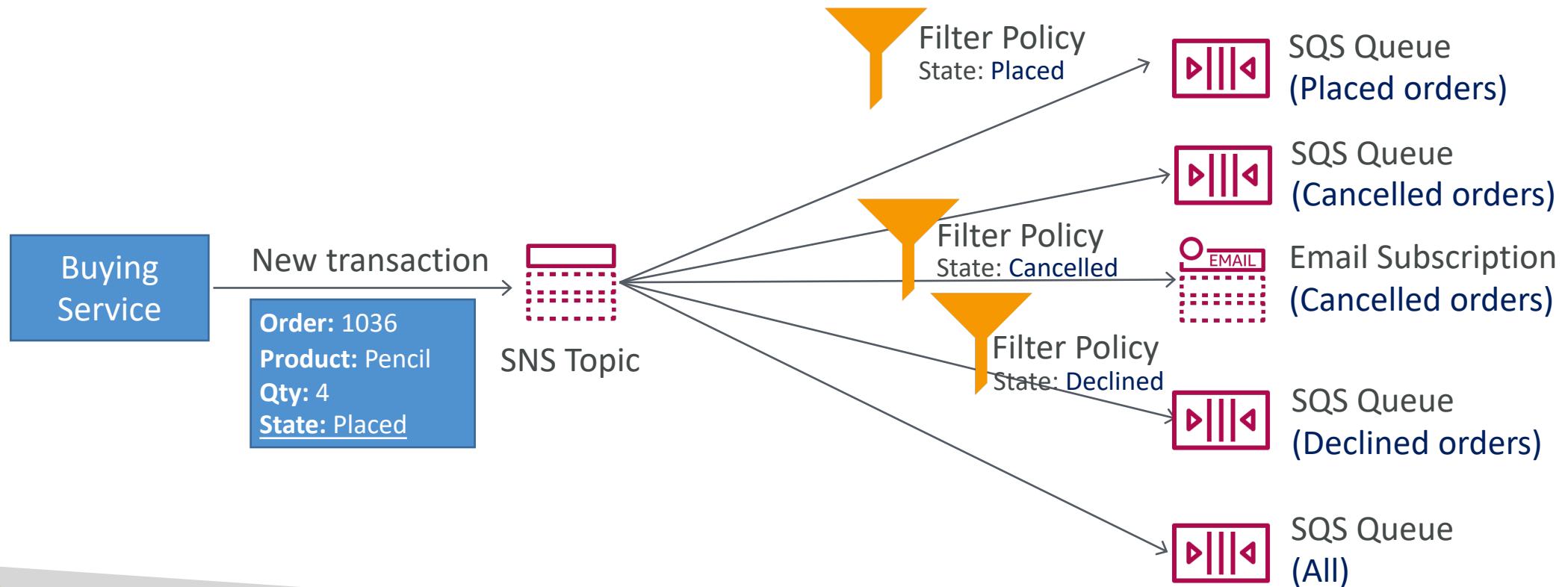
# SNS FIFO + SQS FIFO: Fan Out

- In case you need fan out + ordering + deduplication



# SNS – Message Filtering

- JSON policy used to filter messages sent to SNS topic's subscriptions
- If a subscription doesn't have a filter policy, it receives every message



# SQS vs SNS vs Kinesis

## SQS:

- Consumer “pull data”
- Data is deleted after being consumed
- Can have as many workers (consumers) as we want
- No need to provision throughput
- Ordering guarantees only on FIFO queues
- Individual message delay capability



## SNS:

- Push data to many subscribers
- Up to 12,500,000 subscribers
- Data is not persisted (lost if not delivered)
- Pub/Sub
- Up to 100,000 topics
- No need to provision throughput
- Integrates with SQS for fan-out architecture pattern
- FIFO capability for SQS FIFO



## Kinesis:

- Standard: pull data
  - 2 MB per shard
- Enhanced-fan out: push data
  - 2 MB per shard per consumer
- Possibility to replay data
- Meant for real-time big data, analytics and ETL
- Ordering at the shard level
- Data expires after X days
- Provisioned mode or on-demand capacity mode



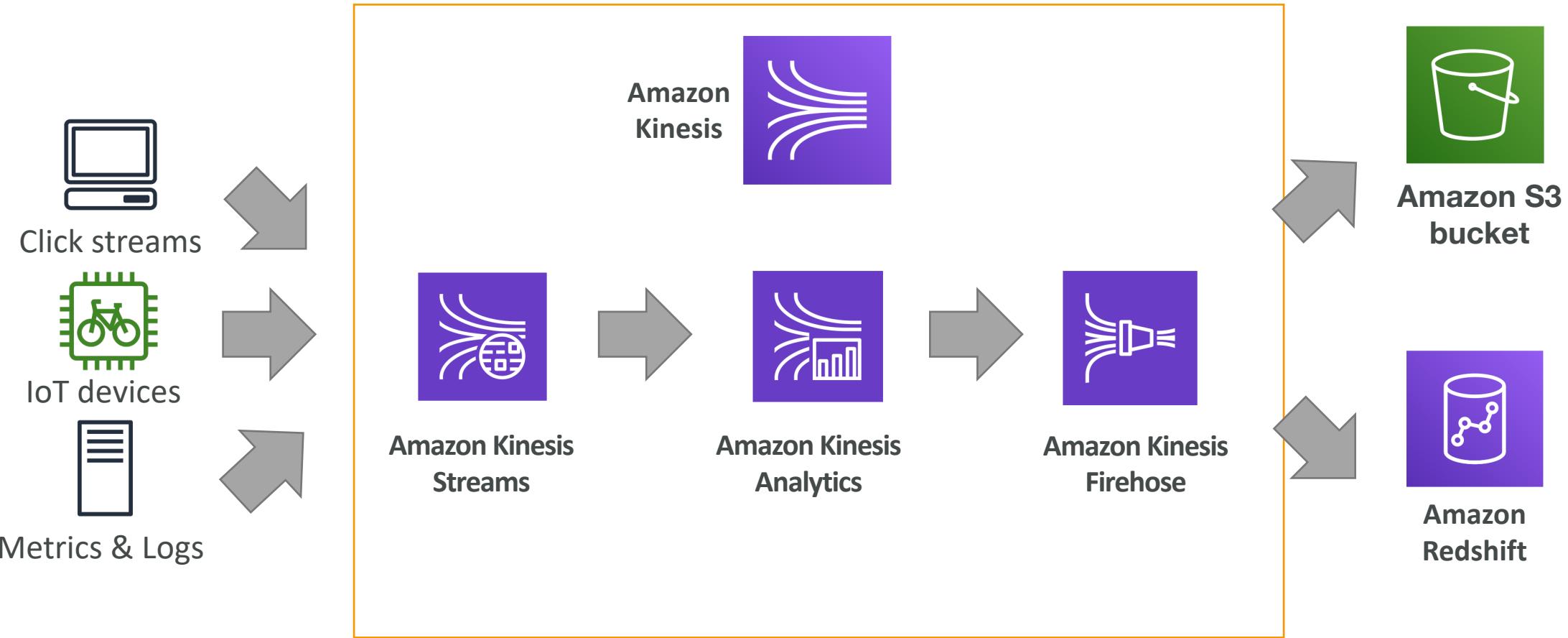
# Data Engineering Section

# AWS Kinesis Overview



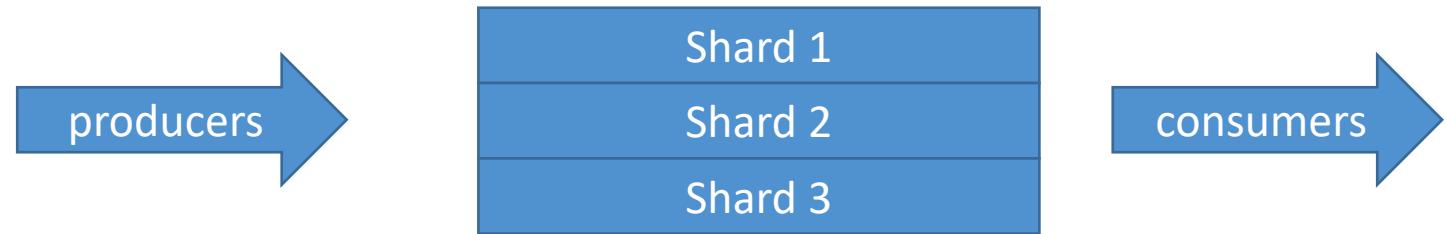
- Kinesis is a managed “data streaming” service
  - Great for application logs, metrics, IoT, clickstreams
  - Great for “real-time” big data
  - Great for streaming processing frameworks (Spark, NiFi, etc...)
  - Data is automatically replicated synchronously to 3 AZ
- 
- **Kinesis Streams:** low latency streaming ingest at scale
  - **Kinesis Analytics:** perform real-time analytics on streams using SQL
  - **Kinesis Firehose:** load streams into S3, Redshift, ElasticSearch & Splunk

# Kinesis



# Kinesis Streams Overview

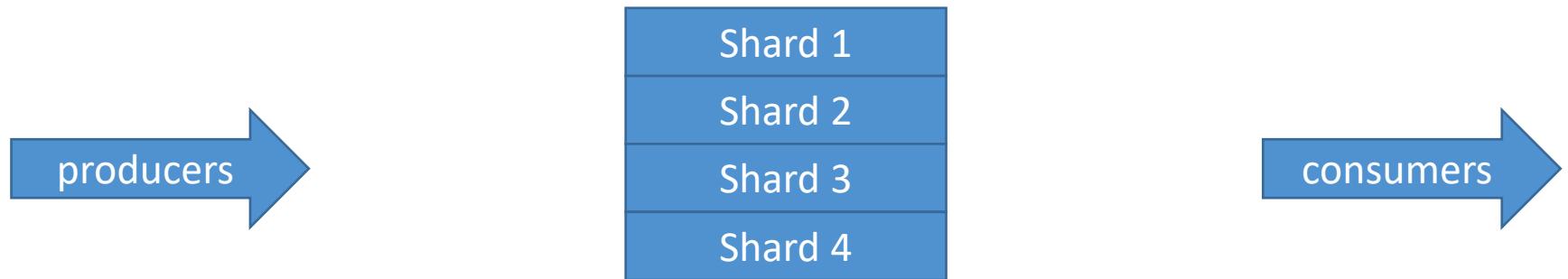
- Streams are divided in ordered Shards / Partitions



- Data retention is 24 hours by default, can go up to 365 days
- Ability to reprocess / replay data
- Multiple applications can consume the same stream
- Real-time processing with scale of throughput
- Once data is inserted in Kinesis, it can't be deleted (immutability)

# Kinesis Streams Shards

- Two modes for capacity:
  - On-demand: no capacity planning, Kinesis scales shards automatically
  - Provisioned: you manage the shards over time
- Batching available or per message calls.
- The number of shards can evolve over time (reshard / merge)
- Records are ordered per shard



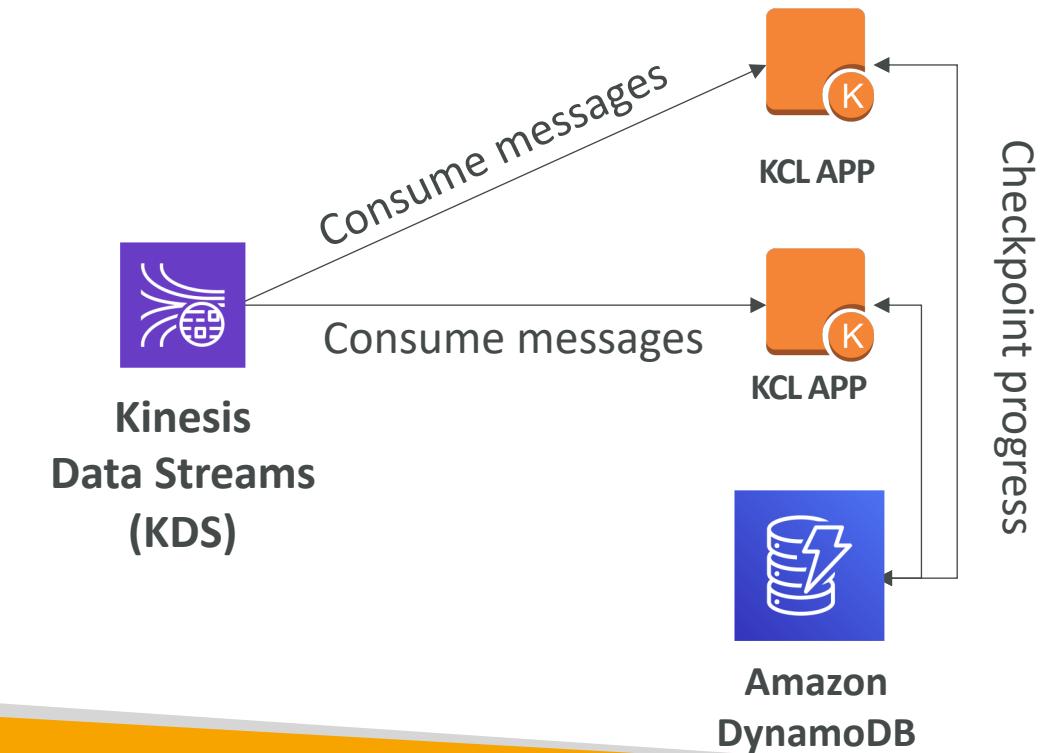
# Kinesis Producers & Consumers

## KINESIS PRODUCERS

- AWS SDK: simple producer
- Kinesis Producer Library (KPL): batch, compression, retries, C++, Java
- Kinesis Agent:
  - Monitor log files and sends them to Kinesis directly
  - can write to Kinesis Data Streams AND Kinesis Data Firehose

## KINESIS CONSUMERS

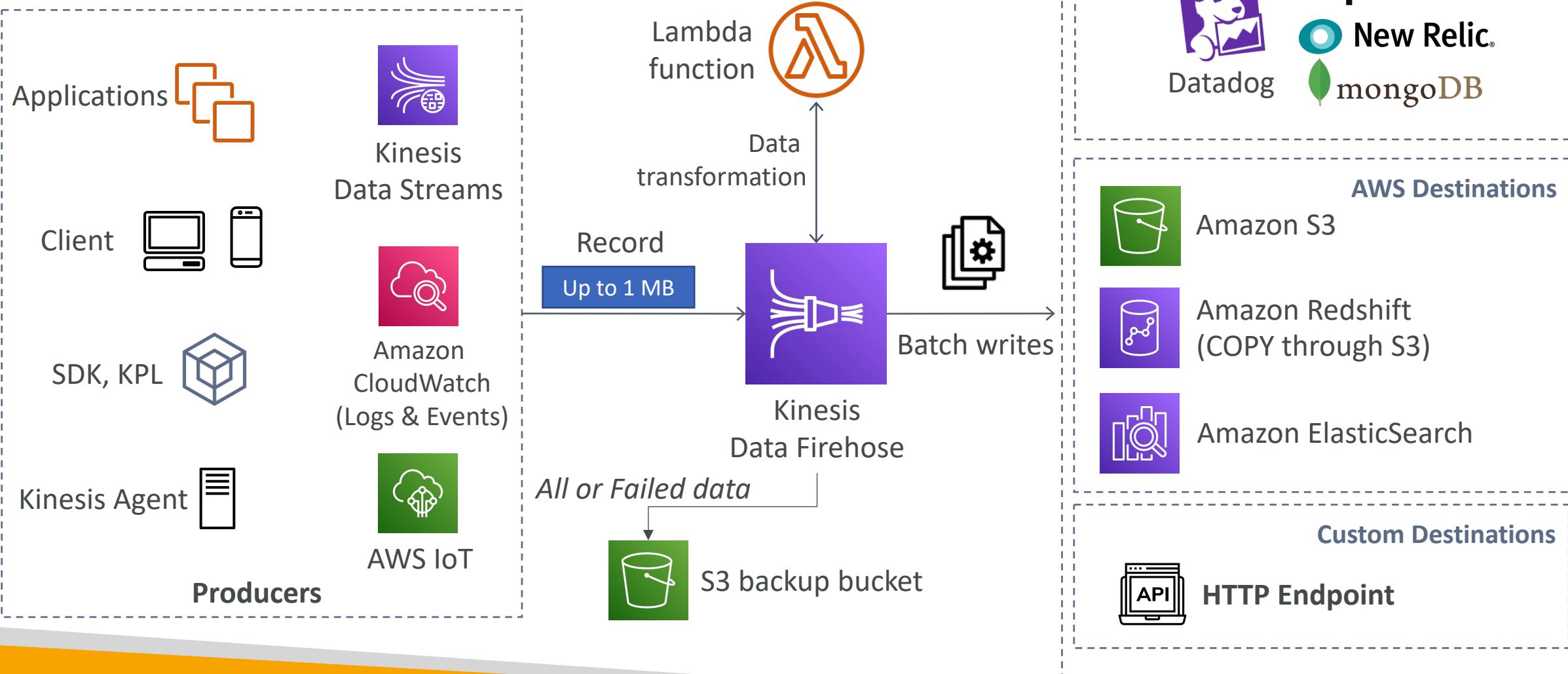
- AWS SDK: simple consumer
- Lambda: (through Event source mapping)
- KCL: checkpointing, coordinated reads



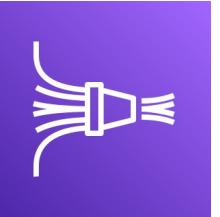
# Kinesis Data Streams Limits to know

- Producer:
  - 1MB/s or 1000 messages/s at write PER SHARD
  - “ProvisionedThroughputException” otherwise
- Consumer Classic:
  - 2MB/s at read PER SHARD across all consumers
  - 5 API calls per second PER SHARD across all consumers
- Consumer Enhanced Fan-Out:
  - 2MB/s at read PER SHARD, PER ENHANCED CONSUMER
  - No API calls needed (push model)
- Data Retention:
  - 24 hours data retention by default
  - Can be extended to 7 days

# Kinesis Data Firehose

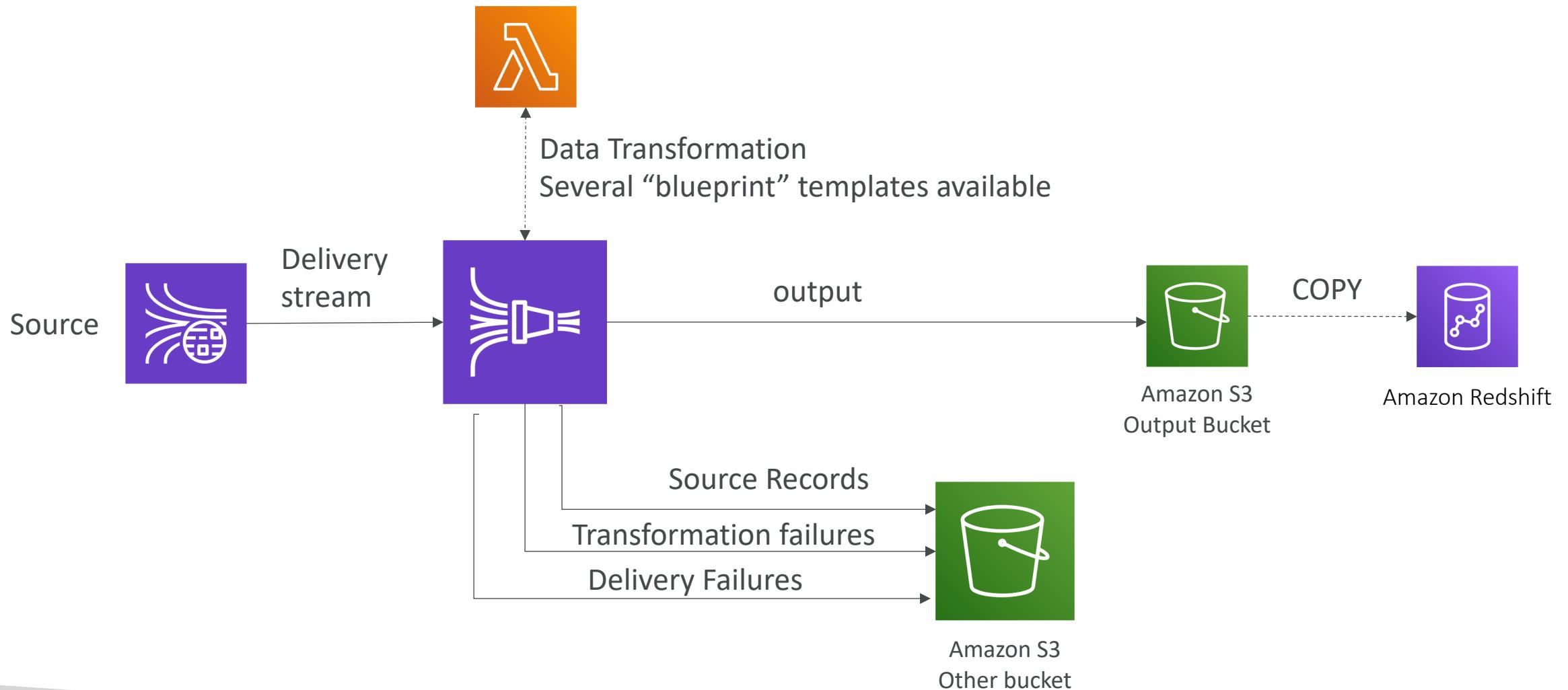


# Kinesis Data Firehose



- Fully Managed Service, no administration, automatic scaling, serverless
  - AWS: Redshift / Amazon S3 / ElasticSearch
  - 3rd party partner: Splunk / MongoDB / DataDog / NewRelic / ...
  - Custom: send to any HTTP endpoint
- Pay for data going through Firehose
- **Near Real Time**
  - 60 seconds latency minimum for non full batches
  - Or minimum 1 MB of data at a time
- Supports many data formats, conversions, transformations, compression
- Supports custom data transformations using AWS Lambda
- Can send failed or all data to a backup S3 bucket

# Kinesis Data Firehose Delivery Diagram



# Firehose Buffer Sizing

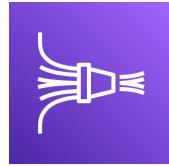
- Firehose accumulates records in a buffer
- The buffer is flushed based on time and size rules
- **Buffer Size (ex: 32MB)**: if that buffer size is reached, it's flushed
- **Buffer Time (ex: 1 minute)**: if that time is reached, it's flushed
- Firehose can automatically increase the buffer size to increase throughput
- High throughput => Buffer Size will be hit
- Low throughput => Buffer Time will be hit
- If real-time flush from Kinesis Data Streams to S3 is needed, use Lambda

# Kinesis Data Streams vs Firehose



## Kinesis Data Streams

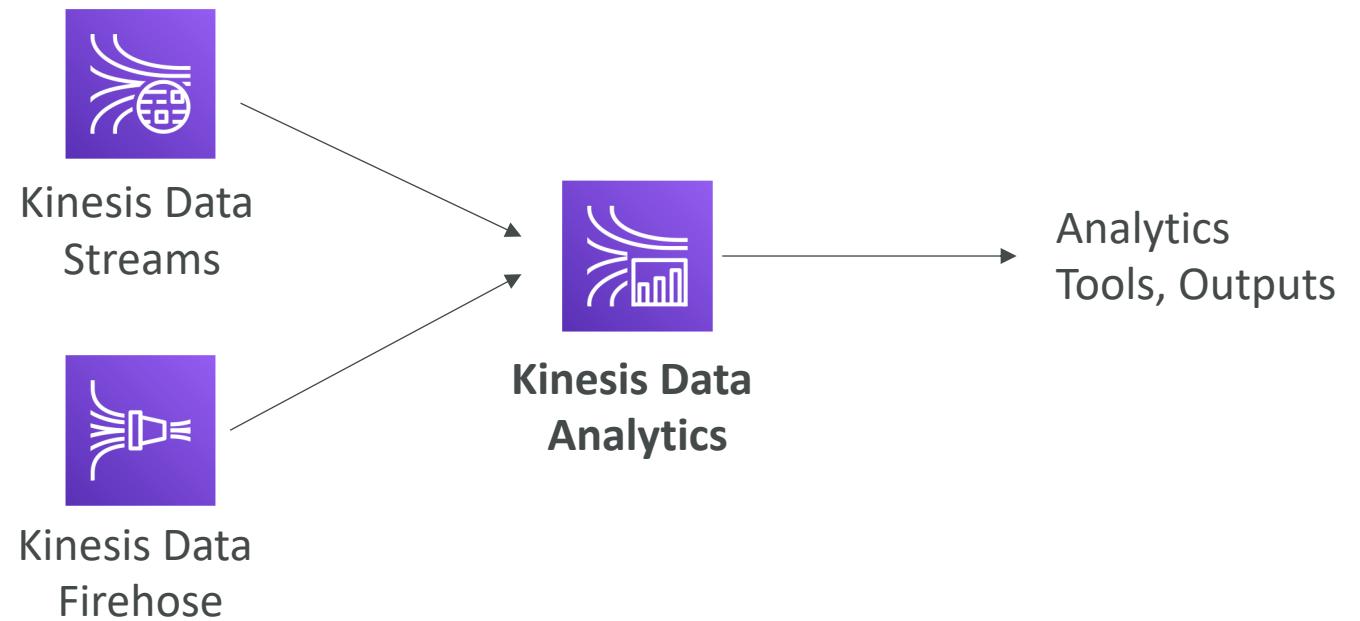
- Streaming service for ingest at scale
- Write custom code (producer / consumer)
- Real-time (~200 ms)
- Manage scaling (shard splitting / merging)
- Data storage for 1 to 365 days
- Supports replay capability



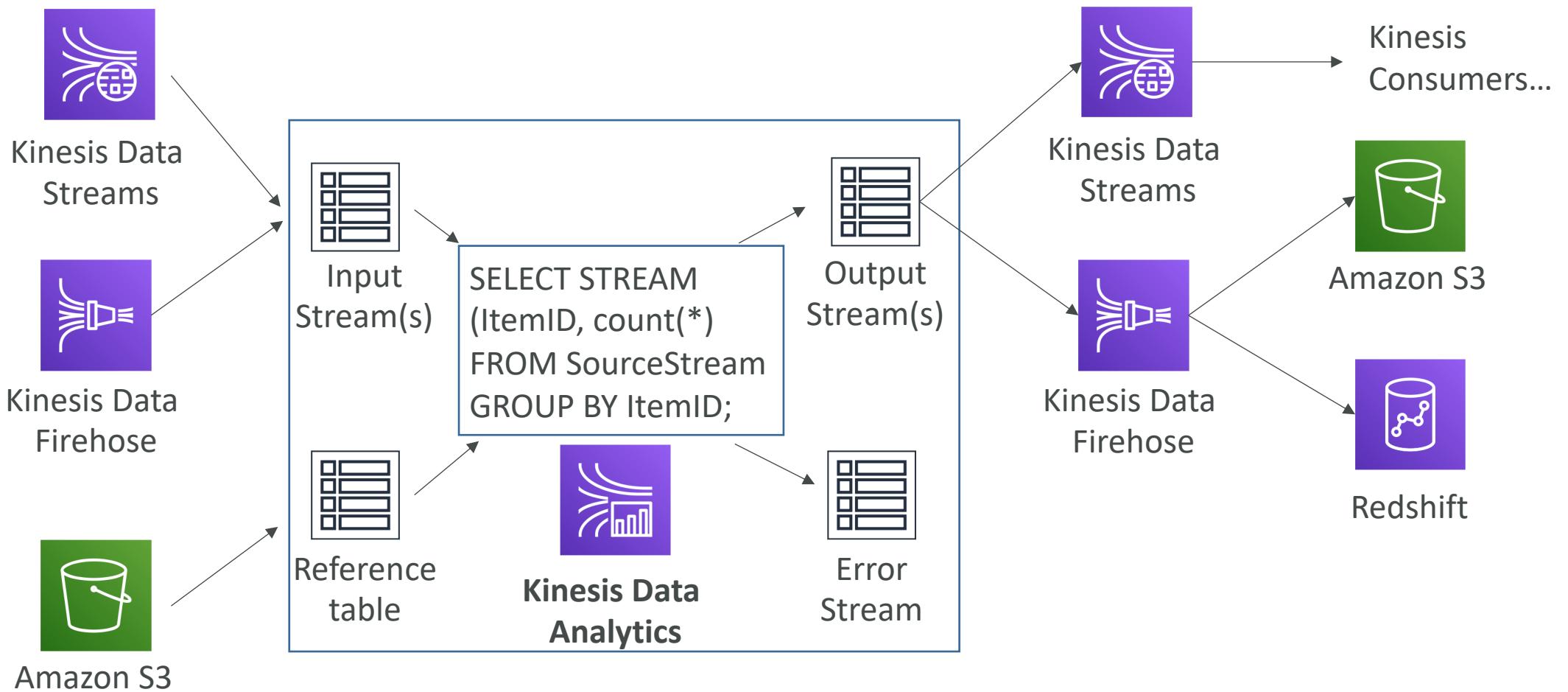
## Kinesis Data Firehose

- Load streaming data into S3 / Redshift / ES / 3<sup>rd</sup> party / custom HTTP
- Fully managed
- Near real-time (buffer time min. 60 sec)
- Automatic scaling
- No data storage
- Doesn't support replay capability

# Kinesis Analytics, Conceptually...



# Kinesis Analytics, In more depth...





# Kinesis Data Analytics

- Use cases
  - Streaming ETL: select columns, make simple transformations, on streaming data
  - Continuous metric generation: live leaderboard for a mobile game
  - Responsive analytics: look for certain criteria and build alerting (filtering)
- Features
  - Pay only for resources consumed (but it's not cheap)
  - Serverless; scales automatically
  - Use IAM permissions to access streaming source and destination(s)
  - SQL or Flink to write the computation
  - Schema discovery
  - Lambda can be used for pre-processing

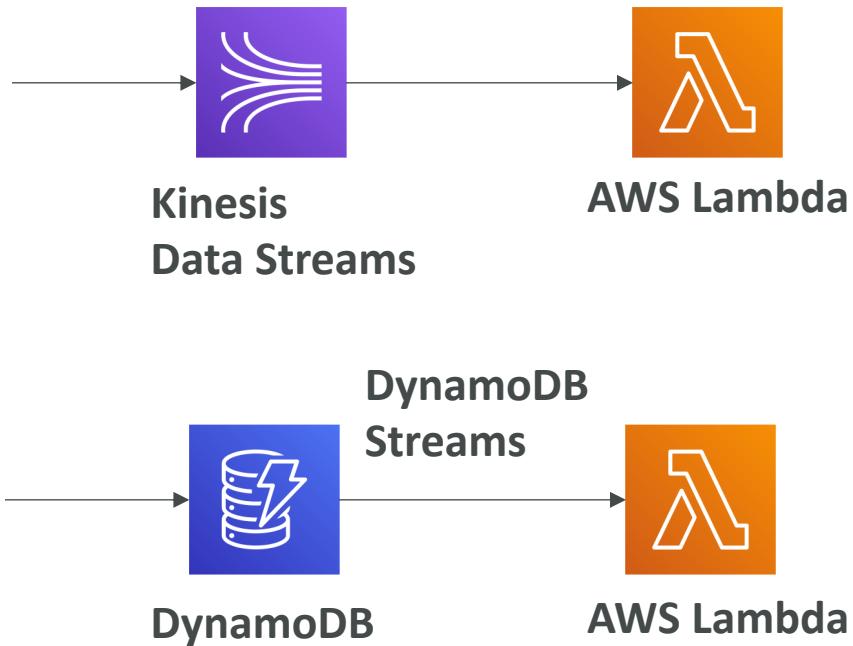
# Full Data Engineering Pipeline

## Real-Time Layer



# Streaming Architectures

## 3000 messages of 1 KB per second



### Kinesis

- 3 shards: 3MB/s in
- $3 * \$0.015/\text{hr} = \$32.4/\text{mth}$
- Must use KDF for output to S3

### DynamoDB + Streams

- 3000 WCU = 3 MB/s
- = \$1,450.90 / month
- Storage in DynamoDB

# Comparison Charts

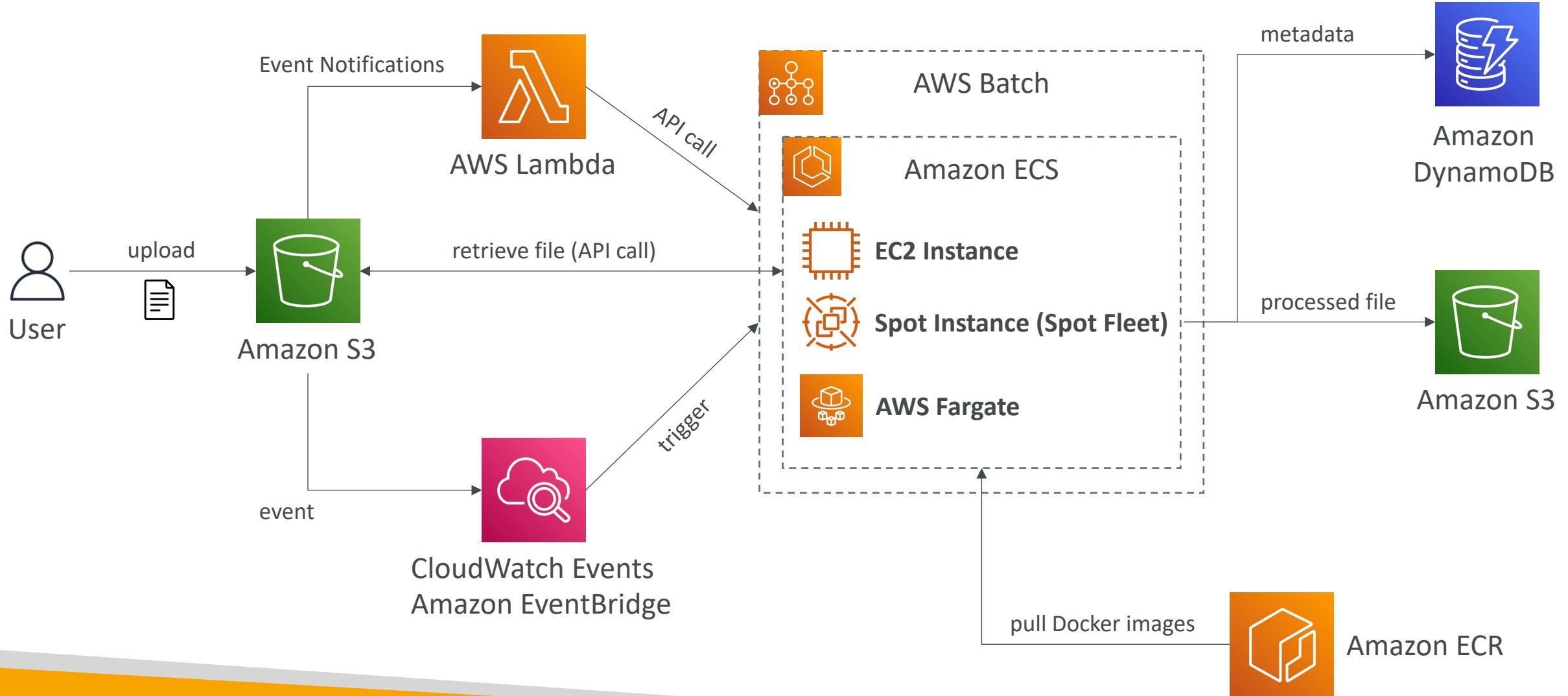
	Kinesis Data Streams	SQS	SQS FIFO	SNS	DynamoDB	S3
Data	Immutable	Immutable	Immutable	Immutable	Mutable	Mutable
Retention	1-7 days, export to S3 using KDF	1-14 days	1-14 days	No retention	Infinite or can implement TTL	Infinite, can setup lifecycle policies
Ordering	Per shard	No ordering	Per group-id	No ordering	No ordering	No ordering
Scalability	Provision shards	Soft limit	300 msg/s Or 3000 if batch	Soft limit	WCU & RCU On-demand	Infinite 3500 PUT 5500 GET / prefix
Readers	EC2, Lambda, KDF, KDA, KCL (checkpoint)	EC2, Lambda	EC2, Lambda	HTTP, Lambda, Email, SQS...	DynamoDB Streams	SDK, S3 Events
Latency	KDS (200 ms) KDF (1 min)	Low (10-100ms)	Low (10-100ms)	Low (10-100ms)	Low (10-100ms)	Low (10-100ms)

# AWS Batch



- Run batch jobs as Docker images
- Two options:
  1. Run on AWS Fargate (fully serverless offering)
  2. Dynamic provisioning of the instances (EC2 & Spot Instances) – in VPC
- Optimal quantity and type based on volume and requirements
- No need to manage clusters, fully **serverless**
- You just pay for the underlying resources used
- Example: batch process of images, running thousands of concurrent jobs
- Schedule Batch Jobs using CloudWatch Events
- Orchestrate Batch Jobs using AWS Step Functions

# AWS Batch – Solution Architecture



# Batch vs. Lambda

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

# AWS Batch – Compute Environments

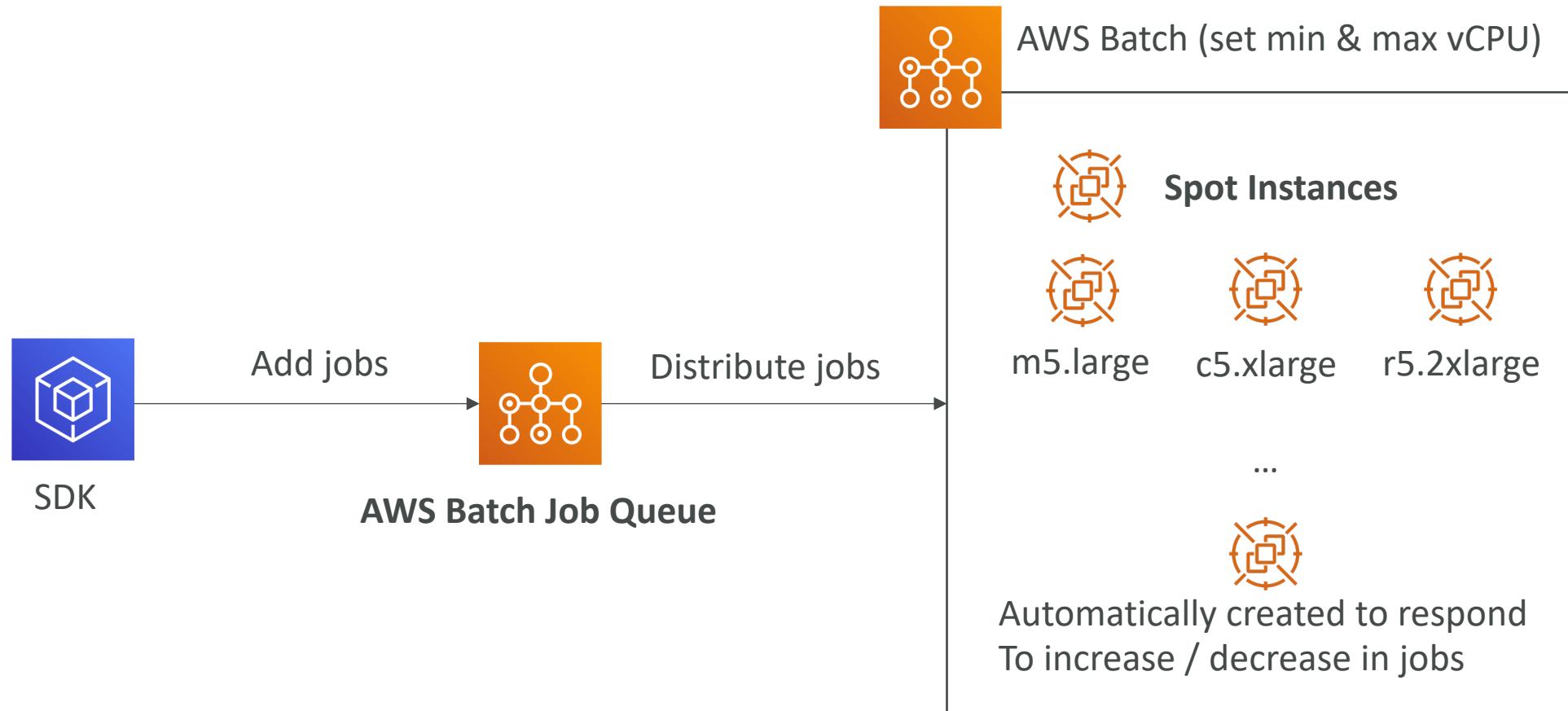
- Managed Compute Environment:

- AWS Batch managed the capacity and instance types within the environment
- You can choose EC2 On-Demand or Spot Instances
- You can choose Fargate On-Demand or Fargate Spot Instances
- You can set a maximum price for Spot Instances
- Launched within your own VPC
  - If you launch within your own private subnet, make sure it has access to the ECS service
  - Either using a NAT gateway / instance or using VPC Endpoints for ECS

- Unmanaged Compute Environment

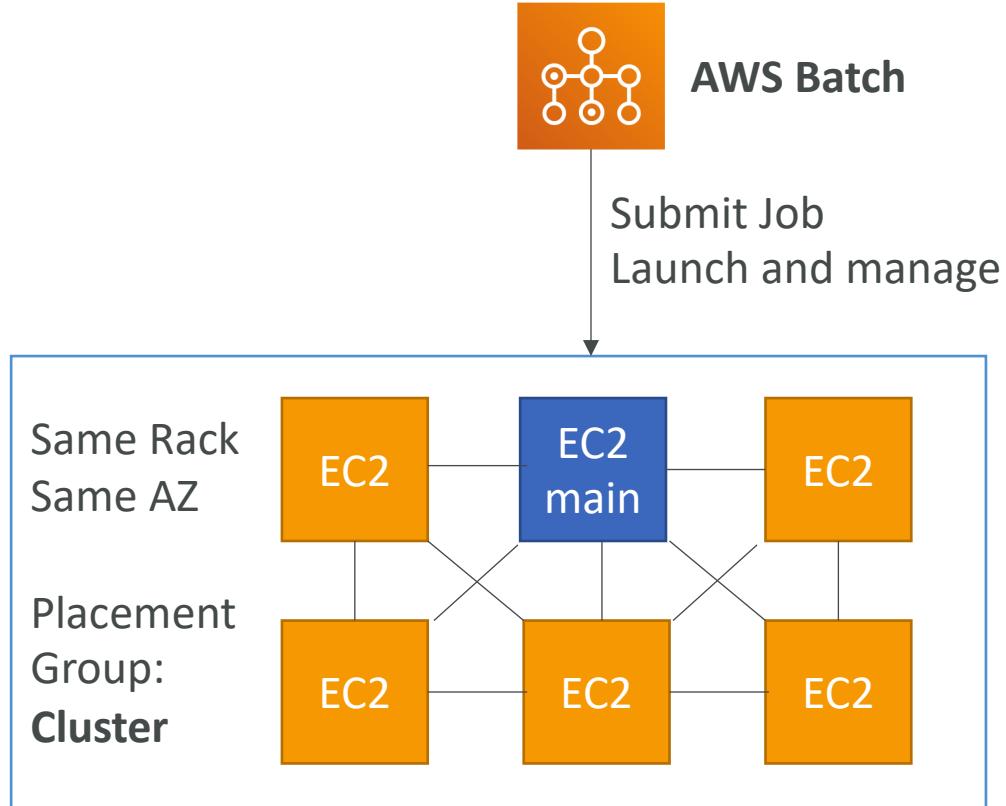
- You control and manage EC2 instance configuration, provisioning and scaling

# AWS Batch – Managed Compute Environment



# AWS Batch – Multi Node Mode

- **Multi Node:** large scale, good for HPC (high performance computing)
  - Leverages multiple EC2 / ECS instances at the same time
  - Good for tightly coupled workloads
  - Represents a single job, and specified how many nodes to create for the job
  - 1 main node, and many child node.
  - **Does not work with Spot Instances**
  - Works better if your EC2 launch mode is a placement group "cluster"

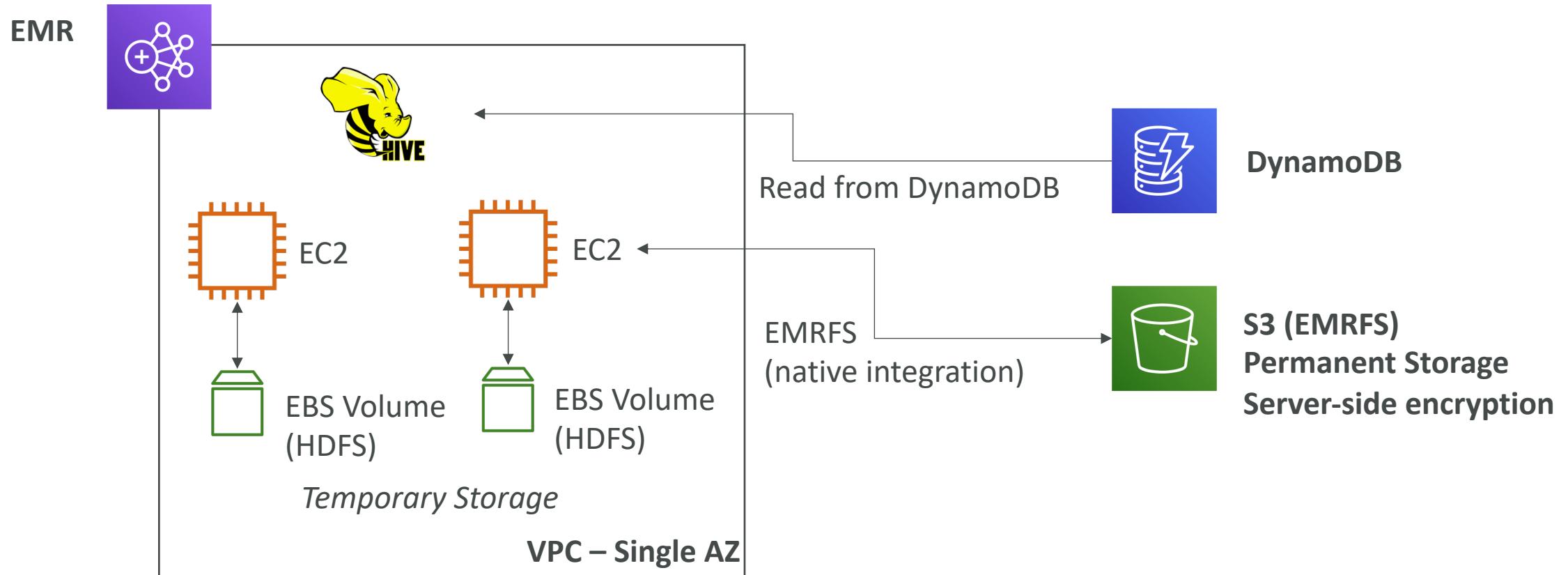


# Amazon EMR



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

# EMR – Integrations



# Amazon EMR – Node types & purchasing

- **Master Node:** Manage the cluster, coordinate, manage health
- **Core Node:** Run tasks and store data
- **Task Node (optional):** Just to run tasks
- **Purchasing options:**
  - On-demand: reliable, predictable, won't be terminated
  - Reserved (min 1 year): cost savings (EMR will automatically use if available)
  - Spot Instances: cheaper, can be terminated, less reliable
- Can have long-running cluster, or transient (temporary) cluster
- One big cluster vs many smaller ones? Long running vs transient?

# Amazon EMR – Instance Configuration

- Uniform instance groups: select a single instance type and purchasing option for each node (has auto scaling)

Node type	Instance type	Instance count	Purchasing option
Master Master - 1	m5.xlarge 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB	1 Instances	<input checked="" type="radio"/> On-demand <input type="radio"/> Spot Use on-demand as max price
Core Core - 2	m5.xlarge 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB	2 Instances	<input checked="" type="radio"/> On-demand <input type="radio"/> Spot Use on-demand as max price
Task X Task - 3	c4.xlarge 4 vCore, 7.5 GiB memory, EBS only storage EBS Storage: 64 GiB	4 Instances	<input checked="" type="radio"/> On-demand <input type="radio"/> Spot Use on-demand as max price
Task X Task - 4	d2.xlarge 8 vCore, 30.5 GiB memory, 6144 SSD GB storage EBS Storage: none	3 Instances	<input type="radio"/> On-demand <input checked="" type="radio"/> Spot Use on-demand as max price

- Instance fleet: select target capacity, mix instance types and purchasing options (no Auto Scaling)

Node type	Fleet instance types	Target capacity
Master Master - 1	m5.xlarge 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB Maximum Spot price: % On-Demand 100 Add / remove instance types to fleet	<input checked="" type="radio"/> On-demand <input type="radio"/> Spot The master fleet consists of one EC2 instance
Core Core - 2	c5d.xlarge 4 vCore, 8 GiB memory, 100 SSD GB storage EBS Storage: none Maximum Spot price: % On-Demand 100 Each instance counts as 4 units	6 On-demand units 10 Spot units 16 Total units
	m5.xlarge 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB Maximum Spot price: % On-Demand 100 Each instance counts as 4 units	
Task X Task - 3	m5.xlarge 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 64 GiB Maximum Spot price: % On-Demand 100 Each instance counts as 4 units Add / remove instance types to fleet	2 On-demand units 20 Spot units 22 Total units

# Running Jobs on AWS

**Strategy 1: Provision EC2 instance  
(long running - CRON jobs)**



EC2

**Strategy 3: Reactive Workflow**

CW Events

S3 Events

API Gateway

SQS, SNS

Etc...



**Strategy 2: CloudWatch Events + Lambda  
(cron)**



cron schedule



CW Events

**Strategy 4: use AWS Batch**



CW Events



Batch

**Strategy 5: use Fargate**



CW Events

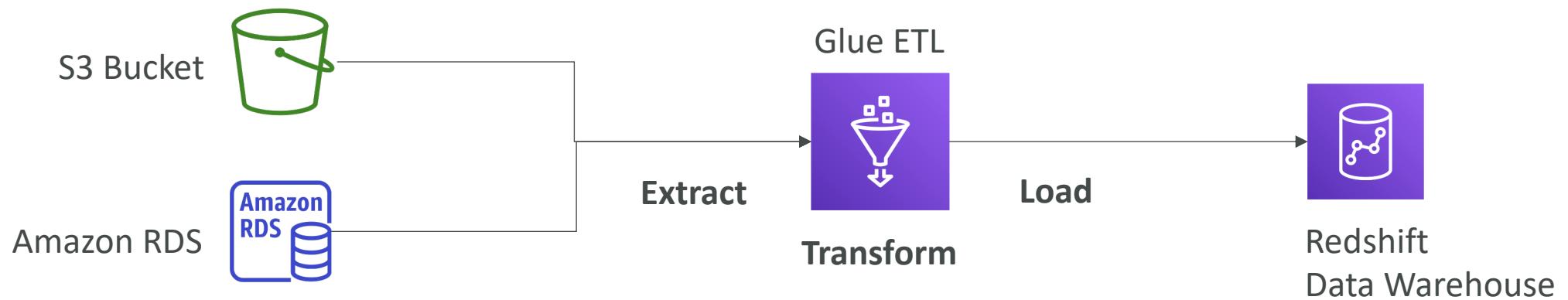


Fargate

# AWS Glue



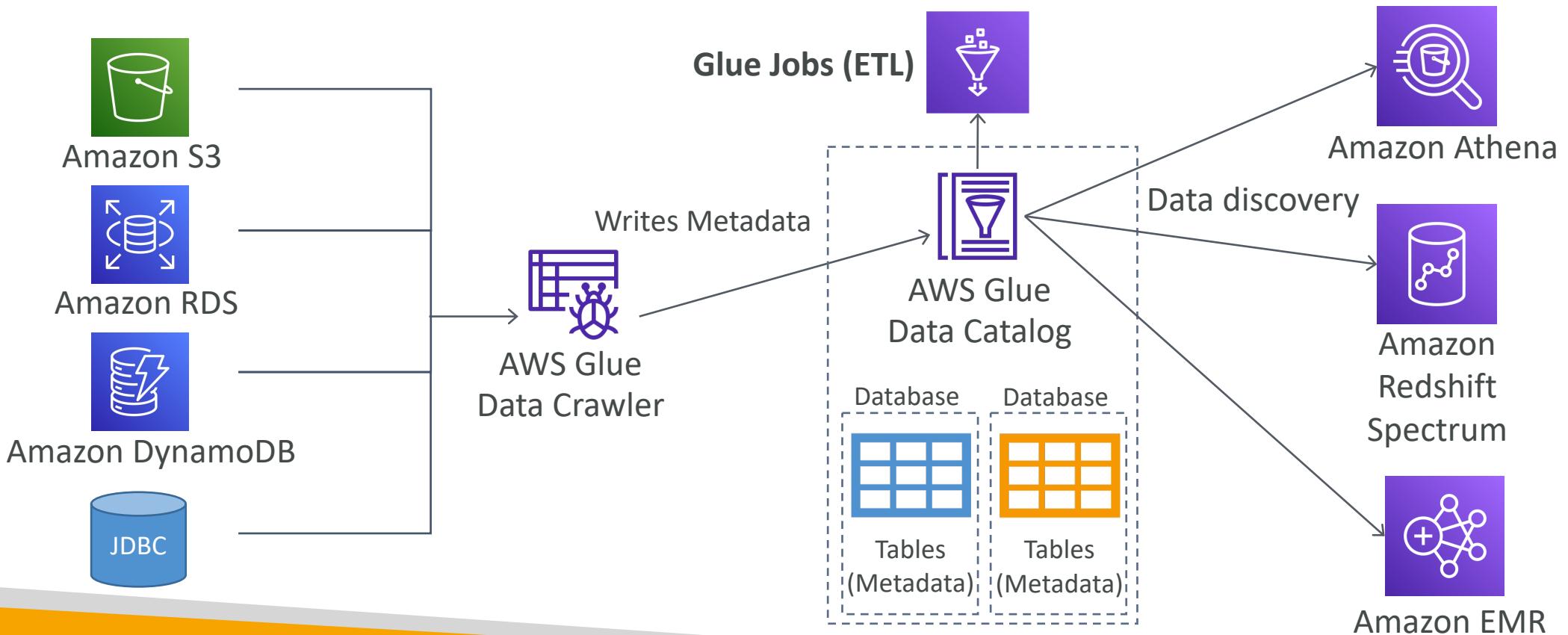
- Managed **extract, transform, and load (ETL)** service
- Useful to prepare and transform data for analytics
- Fully **serverless** service



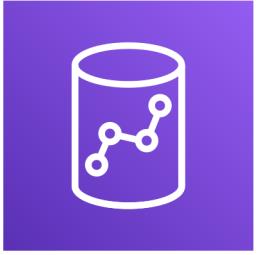
# Glue Data Catalog



- Glue Data Catalog: catalog of datasets



# Redshift Overview



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

# Redshift Continued...



- Data is loaded from S3, Kinesis Firehose, DynamoDB, DMS...
- Based on node type: up to 100+ nodes, up to 16 TB of space per node
- Can provision multiple nodes, but it's not Multi-AZ
- Leader node: for query planning, results aggregation
- Compute node: for performing the queries, send results to leader
- Backup & Restore, Security VPC / IAM / KMS, Monitoring
- Redshift Enhanced VPC Routing: COPY / UNLOAD goes through VPC
- Redshift is provisioned, so it's worth it when you have a sustained usage (use Athena if the queries are sporadic instead)

# Redshift – Snapshots & DR

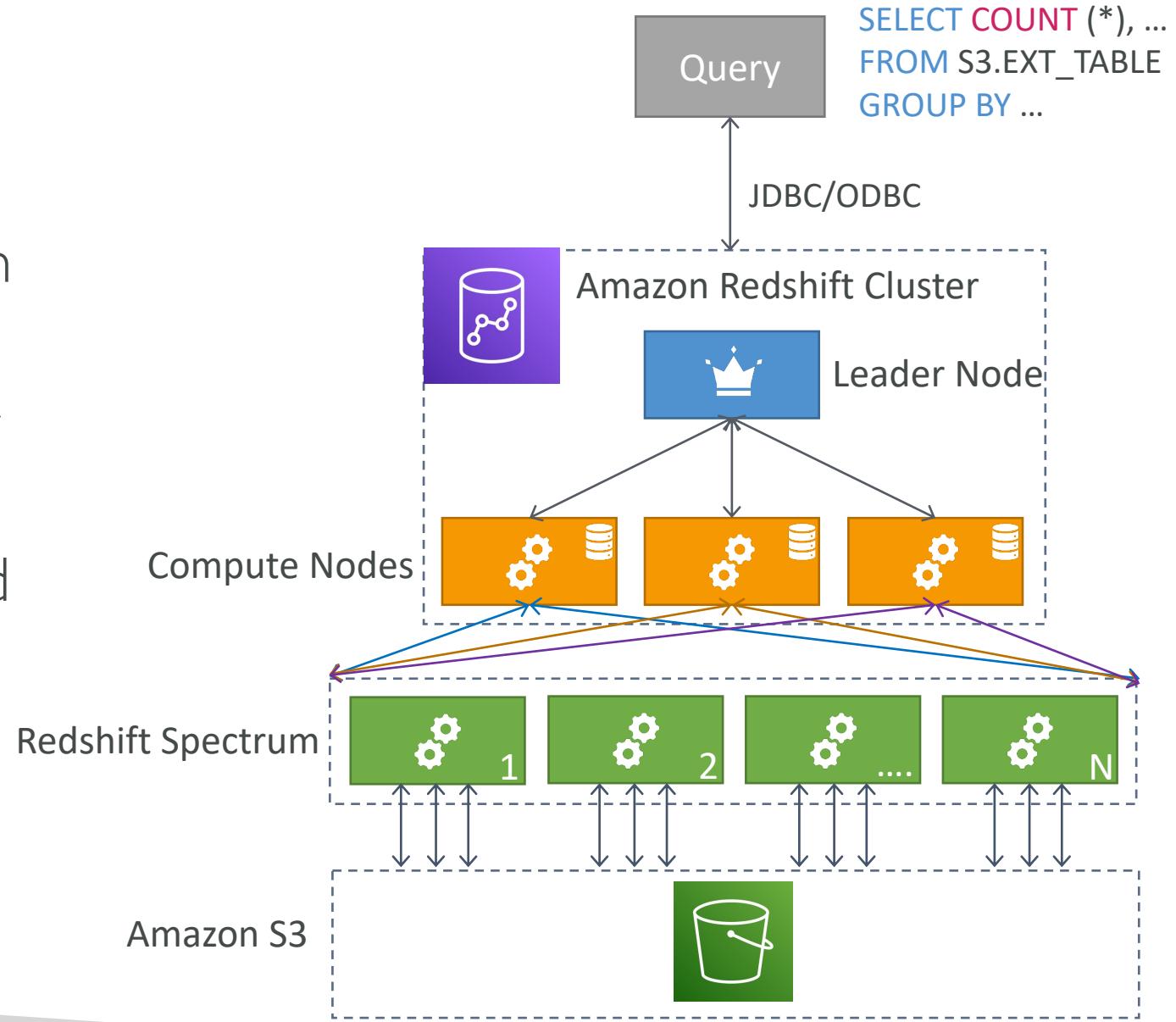
- Snapshots are point-in-time backups of a cluster, stored internally in S3
  - Snapshots are incremental (only what has changed is saved)
  - You can restore a snapshot into a **new cluster**
  - Automated: every 8 hours, every 5 GB, or on a schedule. Set retention
  - Manual: snapshot is retained until you delete it
- 
- You can configure Amazon Redshift to automatically copy snapshots (automated or manual) of a cluster to another AWS Region

# Cross-Region Snapshot Copy for an KMS-Encrypted Redshift



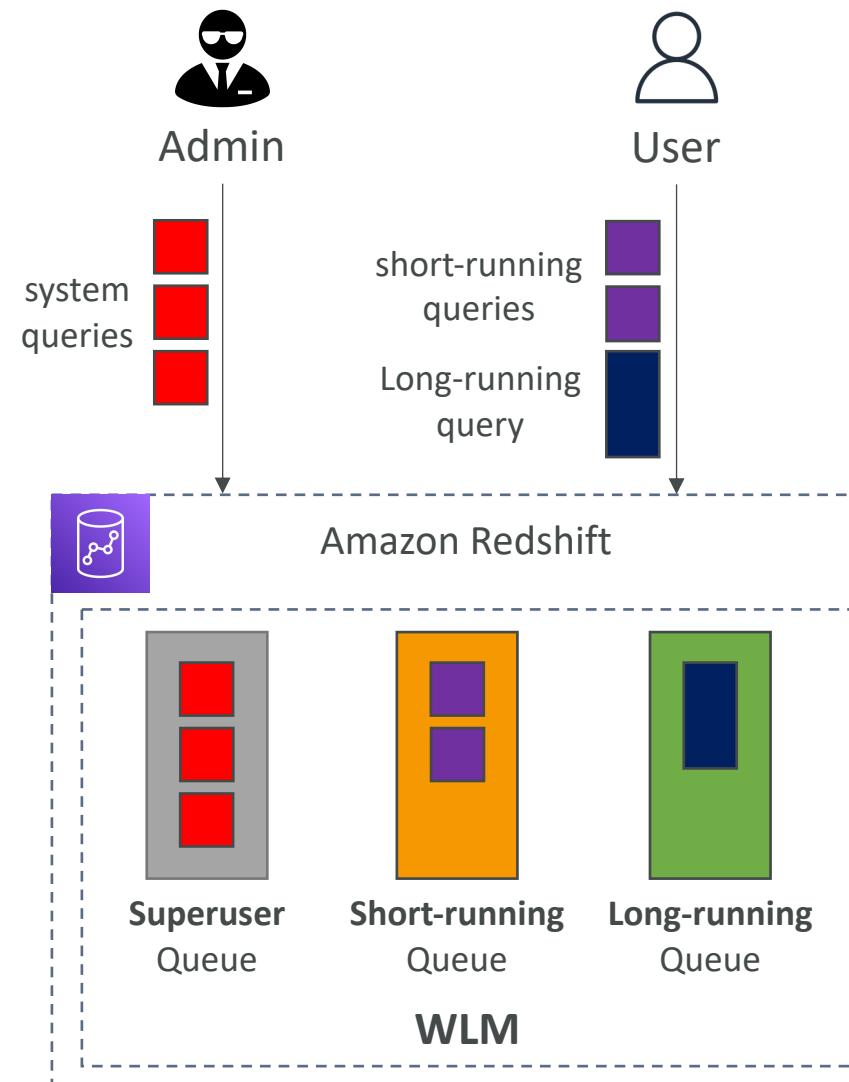
# Redshift Spectrum

- Query data that is already in S3 without loading it
- Must have a Redshift cluster available to start the query
- The query is then submitted to thousands of Redshift Spectrum nodes



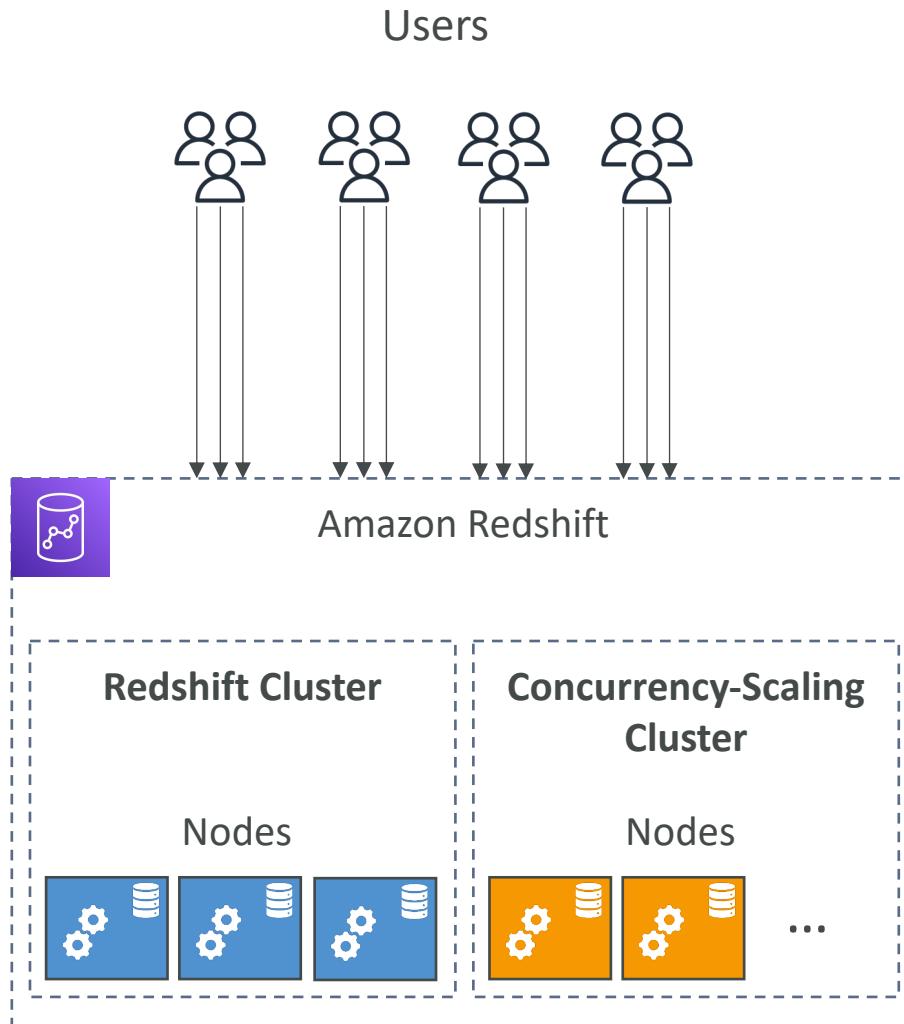
# Redshift Workload Management (WLM)

- Enables you to flexibly manage queries' priorities within workloads
- Example: prevent short, fast-running queries from getting stuck behind long-running queries
- Define multiple query queues (Superuser queue, User-defined queues)
- Route queries to the appropriate queues at runtime
- Automatic WLM – queues and resources managed by Redshift
- Manual WLM – queues and resources managed by you



# Redshift Concurrency Scaling

- Enables you to provide consistently fast performance with virtually unlimited concurrent users and queries
- Redshift automatically adds additional cluster capacity (**Concurrency-Scaling Cluster**) to process an increase in requests
- Ability to decide which queries sent to the concurrency-Scaling Cluster using WLM
- Charged per second



# DocumentDB



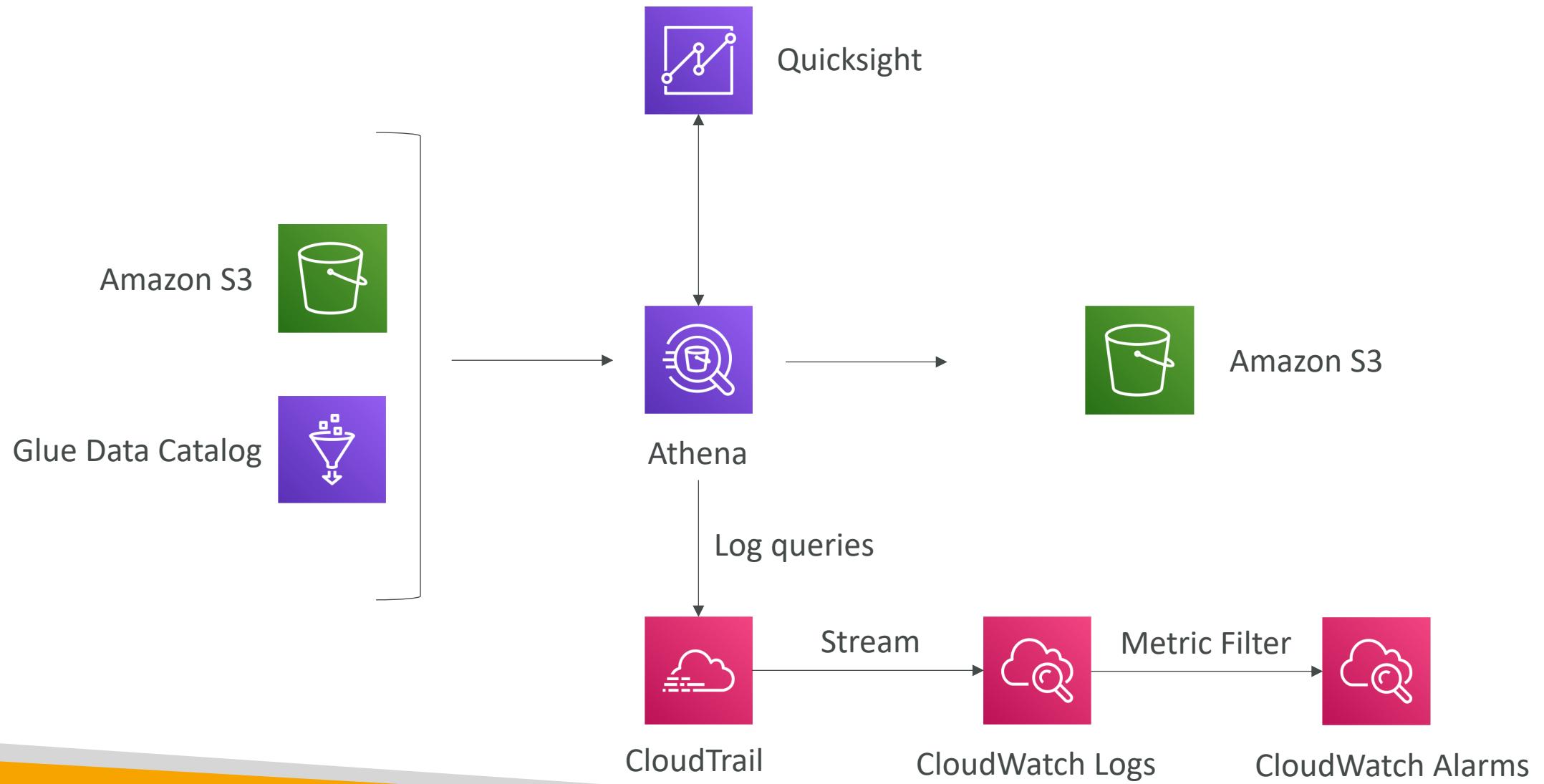
- Aurora is an “AWS-implementation” of PostgreSQL / MySQL ...
  - **DocumentDB is the same for MongoDB (which is a NoSQL database)**
  - MongoDB is used to store, query, and index JSON data
- 
- Similar “deployment concepts” as Aurora
  - Fully Managed, highly available with replication across 3 AZ
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

# Athena & Quicksight



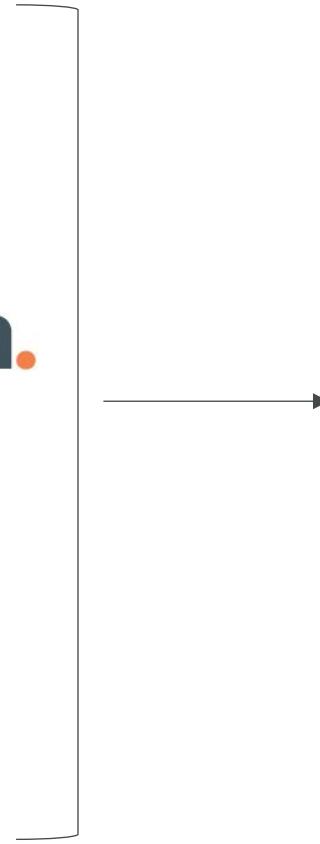
- **Athena:**
  - Serverless SQL queries on top of your data in S3, pay per query, output to S3
  - Supports CSV, JSON, Parquet, ORC, etc...
  - Queries are logged in CloudTrail (which can be chained with CloudWatch logs)
  - Great for sporadic queries
  - Ready-to-use queries for VPC Flow Logs, CloudTrail, ALB Access Logs, Cost and Usage reports (billing), etc...
- **Quicksight:**
  - Business Intelligence tool for data visualization, creating dashboards
  - Integrates with Athena, Redshift, EMR, RDS...

# Athena



# Quicksight

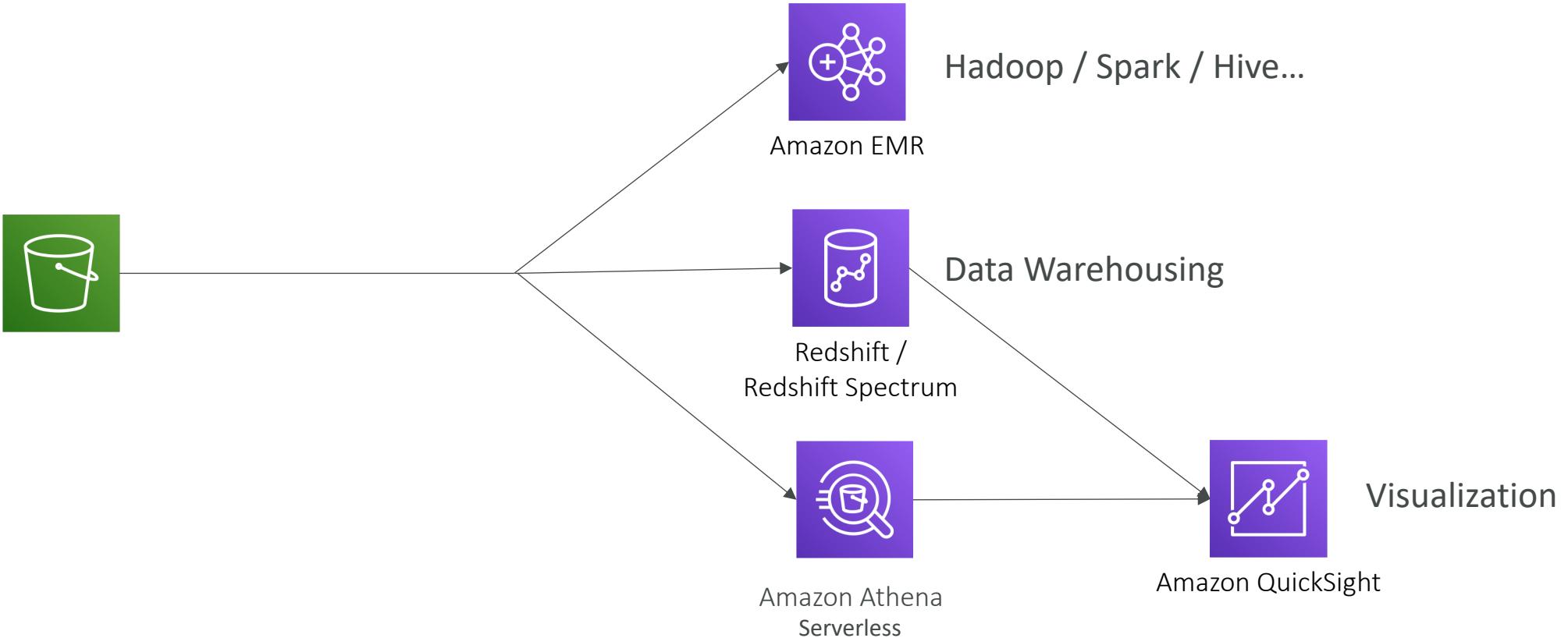
RDS / JDBC	
Redshift	
Athena	
Amazon S3	



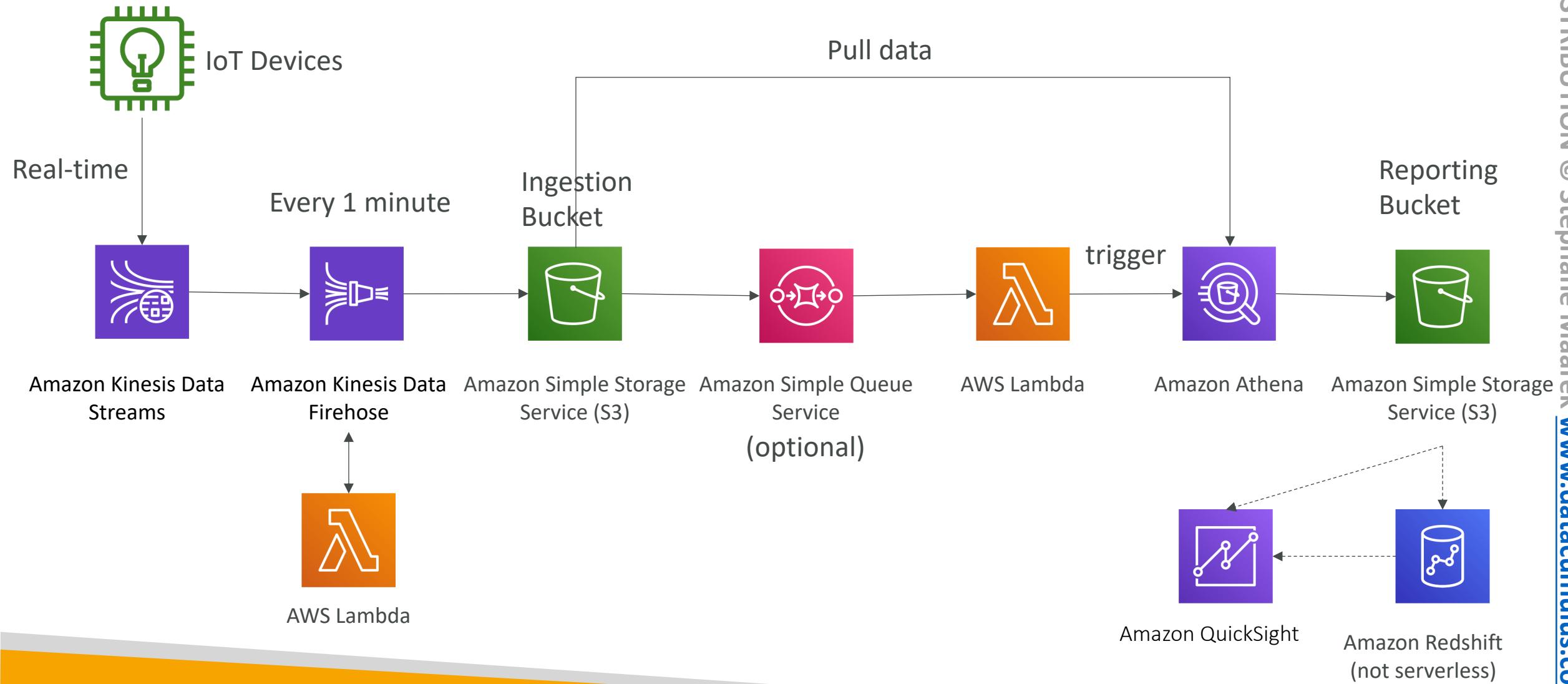
Quicksight

# Full Data Engineering Pipeline

## Analytics layer



# Big Data Ingestion Pipeline



# Comparison of warehousing technologies

- **EMR**
  - Need to use Big Data tools such as Apache Hive, Spark
  - One long-running cluster, many jobs, with auto-scaling, or one cluster per job?
  - Purchasing options – Spot, On Demand, Reserved Instances
  - Can access data in DynamoDB and / or S3
  - Scratch data on EBS disks (HDFS) and long term storage in S3 (EMRFS)
- **Athena**
  - Simple queries and aggregations, data must live in S3
  - Serverless, simple SQL queries, out-of-the-box queries for many services (cost & billing..)
  - Audit queries through CloudTrail
- **Redshift**
  - Advanced SQL queries, must provision servers
  - Can leverage Redshift Spectrum for serverless queries on S3

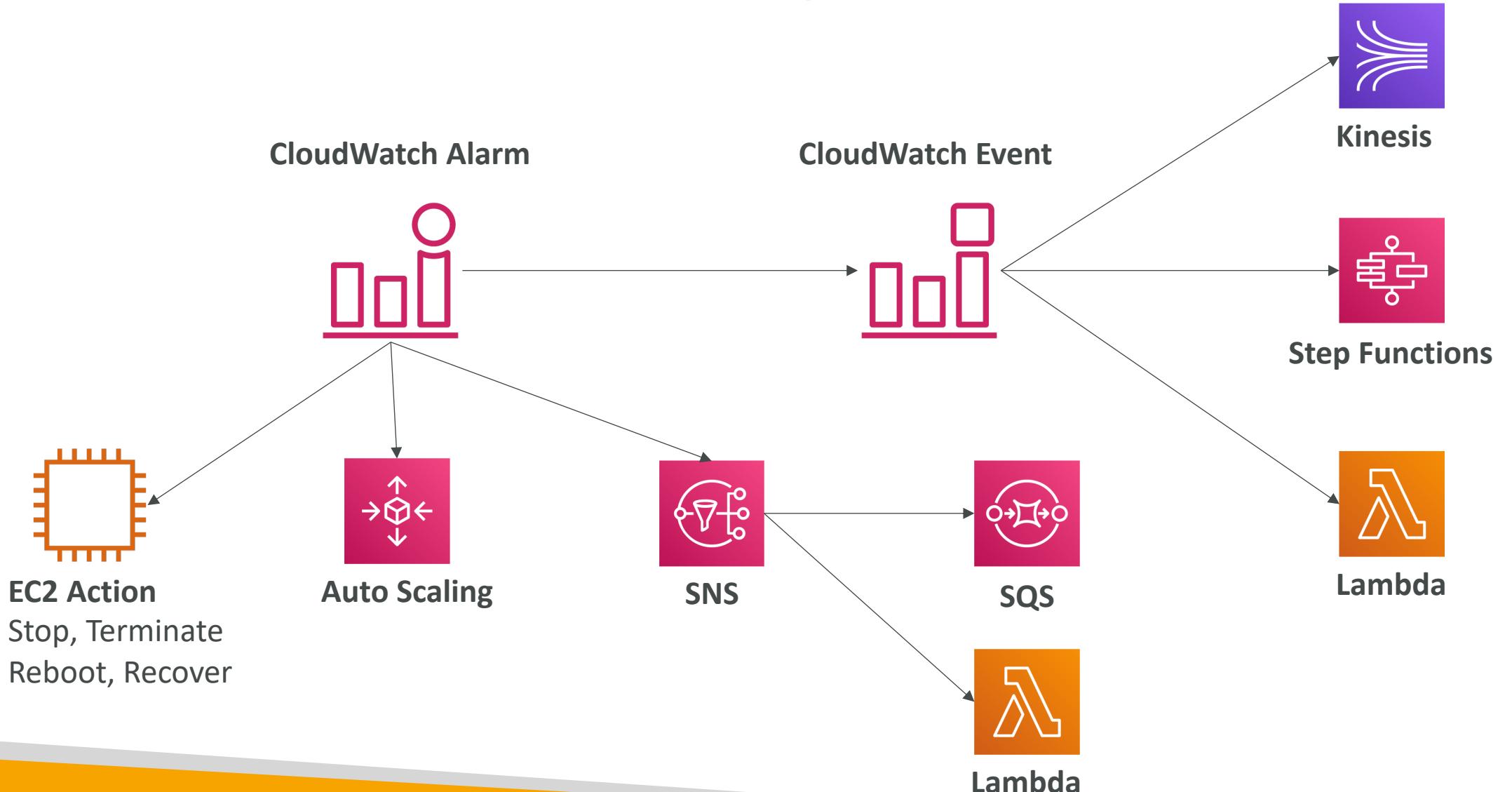
# Monitoring Section



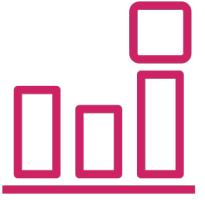
# CloudWatch

- **CloudWatch Metrics**
  - Provided by many AWS services
  - EC2 standard: 5 minutes, detailed monitoring: 1 minute
  - EC2 RAM is not a built-in metric
  - Can create custom metrics: standard resolution 1 minute, high resolution 1 sec
- **CloudWatch Alarms**
  - Can trigger actions: EC2 action (reboot, stop, terminate, recover), Auto Scaling, SNS
  - Alarm events can be intercepted by CloudWatch Events
- **CloudWatch Dashboards**
  - Display metrics and alarms
  - Can show metrics of multiple regions

# CloudWatch Alarms integrations



# CloudWatch Events



- Intercept events from AWS services
- Example: EC2 Instance Start, CodeBuild Failure, S3, Trusted Advisor
- Can intercept any API call with CloudTrail integration
- Notable targets:
  - **Compute:** Lambda, Batch, ECS task
  - **Orchestration:** Step Functions, CodePipeline, CodeBuild
  - **Integration:** SQS, SNS, Kinesis Data Streams, Kinesis Data Firehose
  - **Maintenance:** SSM, EC2 Actions

# AWS CloudWatch Logs - Sources

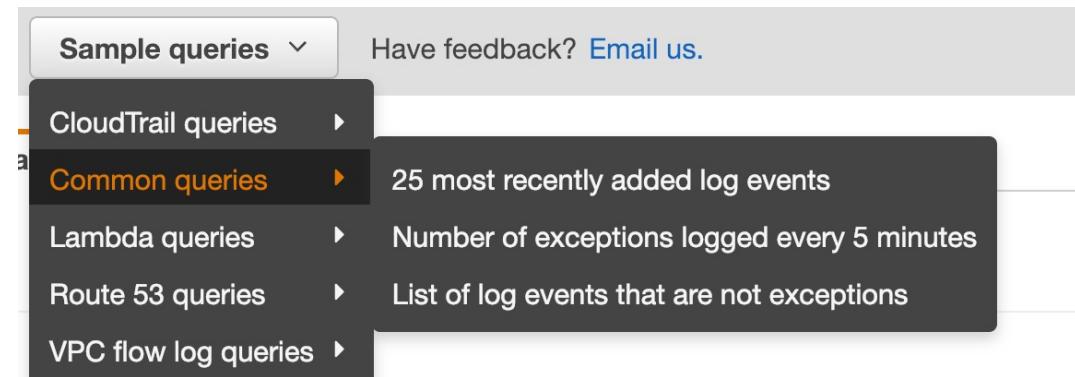
- SDK, CloudWatch Logs Agent, CloudWatch Unified Agent
- Elastic Beanstalk: collection of logs from application
- ECS: collection from containers
- AWS Lambda: collection from function logs
- VPC Flow Logs: VPC specific logs
- API Gateway
- CloudTrail based on filter
- CloudWatch log agents: for example on EC2 machines
- Route53: Log DNS queries

# CloudWatch Logs

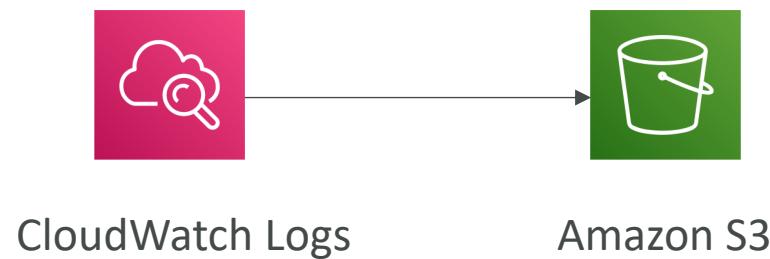
- **Log groups:** arbitrary name, usually representing an application
- **Log stream:** instances within application / log files / containers
- Can define log expiration policies (never expire, 30 days, etc..)
- Optional KMS encryption
- **CloudWatch Logs can send logs to:**
  - Amazon S3 (exports)
  - Kinesis Data Streams
  - Kinesis Data Firehose
  - AWS Lambda
  - ElasticSearch

# CloudWatch Logs Metric Filter & Insights

- CloudWatch Logs can use filter expressions
  - For example, find a specific IP inside of a log
  - Or count occurrences of “ERROR” in your logs
  - Metric filters can be used to trigger alarms
- CloudWatch Logs Insights can be used to query logs and add queries to CloudWatch Dashboards

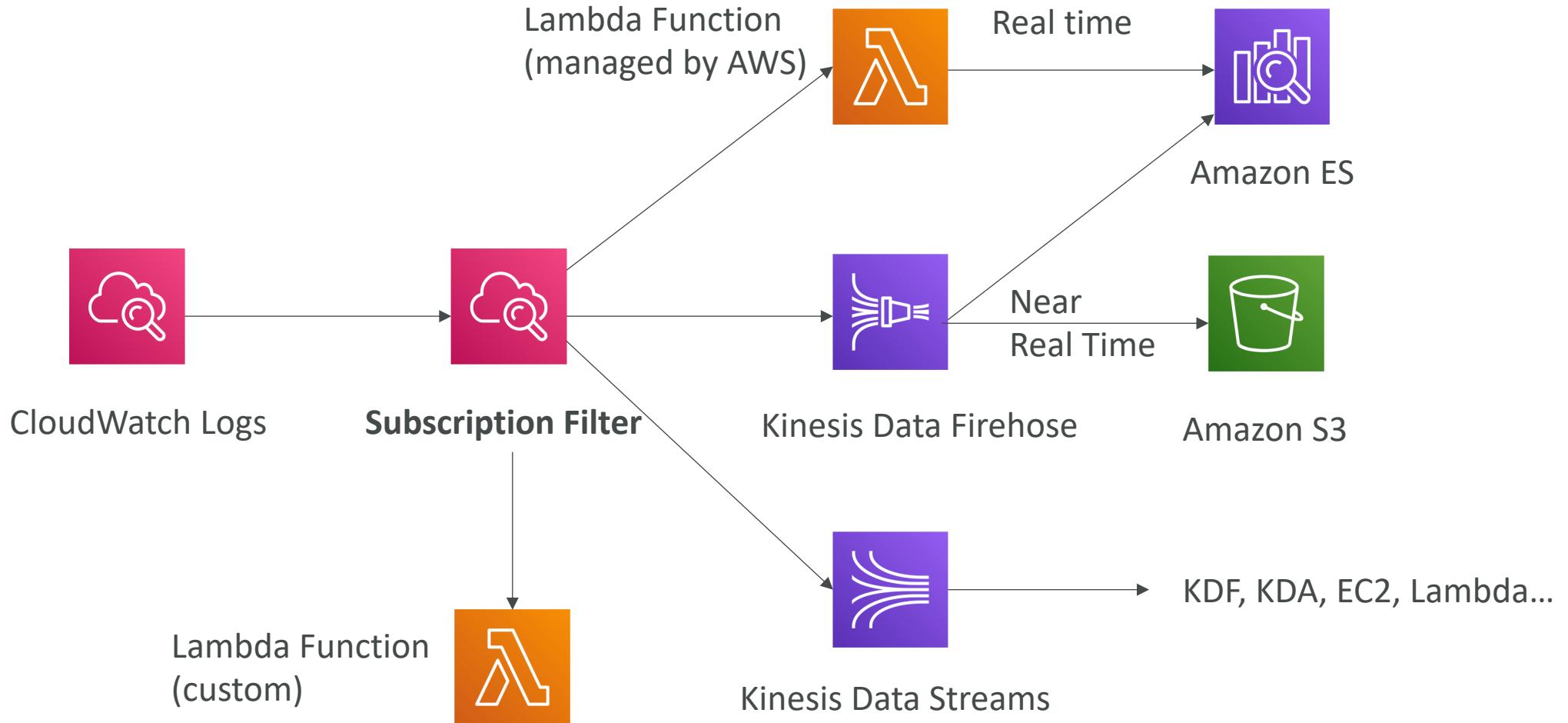


# CloudWatch Logs – S3 Export

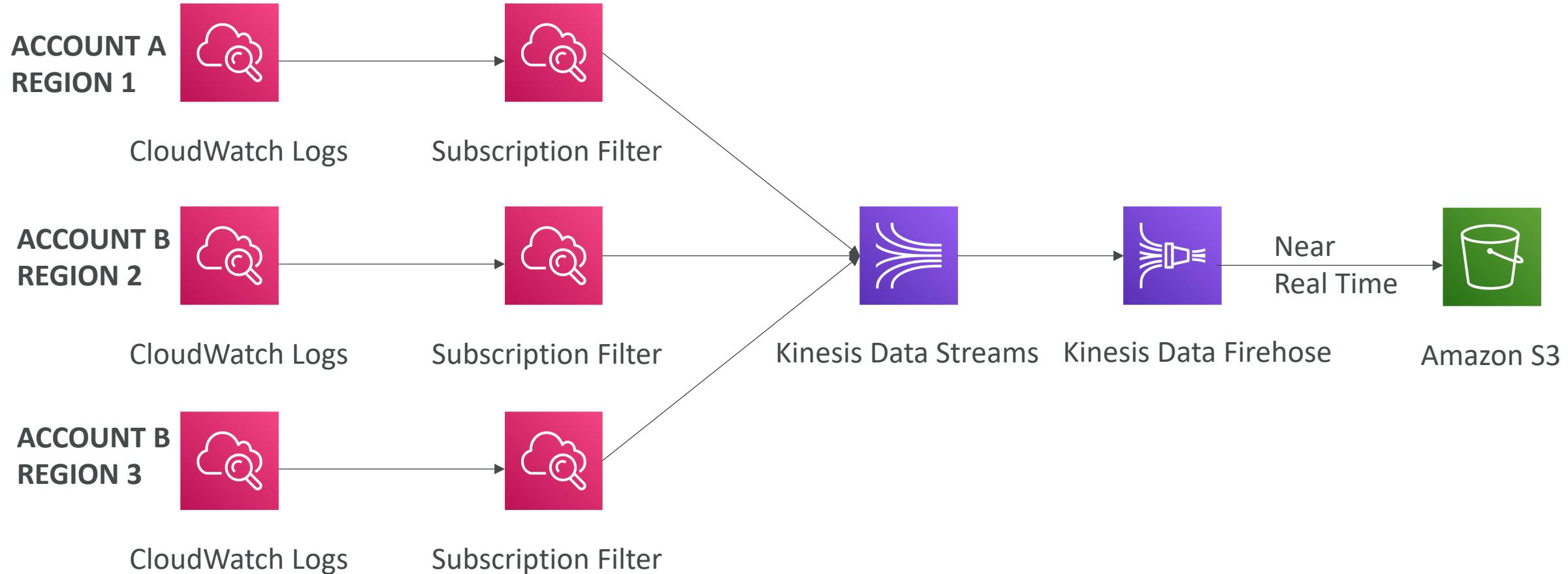


- S3 buckets must be encrypted with AES-256 (SSE-S3), not SSE-KMS
- Log data can take up to 12 hours to become available for export
- The API call is `CreateExportTask`
  
- Not near-real time or real-time... use Logs Subscriptions instead

# CloudWatch Logs Subscriptions



# CloudWatch Logs Aggregation Multi-Account & Multi Region



# CloudWatch Logs Agent & Unified Agent

- For virtual servers (EC2 instances, on-premises servers...)
- **CloudWatch Logs Agent**
  - Old version of the agent
  - Can only send to CloudWatch Logs
- **CloudWatch Unified Agent**
  - Collect additional system-level metrics such as RAM, processes, etc...
  - Collect logs to send to CloudWatch Logs
  - Centralized configuration using SSM Parameter Store
- **Batch Sends**
  - batch\_count: number of log events to send (default 10000, min 1)
  - batch\_duration: duration of batching for log events (default & min is 5000ms)
  - batch\_size: max size of log events in a batch (default & max is 1 MB)
- **Both agents cannot send logs to Kinesis**

# AWS X-Ray

## Visual analysis of our applications





# X-Ray

- Tracing requests across your microservices (distributed systems)
- Integrations with:
  - EC2 – install the X-Ray agent
  - ECS – install the X-Ray agent or Docker container
  - Lambda
  - Beanstalk - agent is automatically installed
  - API Gateway – helpful to debug errors (such as 504)
- The X-Ray agent or services need IAM permissions to X-Ray

# Deployment and Instance Management Section

# AWS Elastic Beanstalk Overview



- Elastic Beanstalk is a developer centric view of deploying an application on AWS
- It uses all the component's we've seen before: EC2, Auto Scaling Group, Elastic Load Balancers, RDS, etc...
- But it's all in one view that's easy to make sense of!
- We still have full control over the configuration of each component
- Beanstalk is free but you pay for the underlying instances

# Elastic Beanstalk

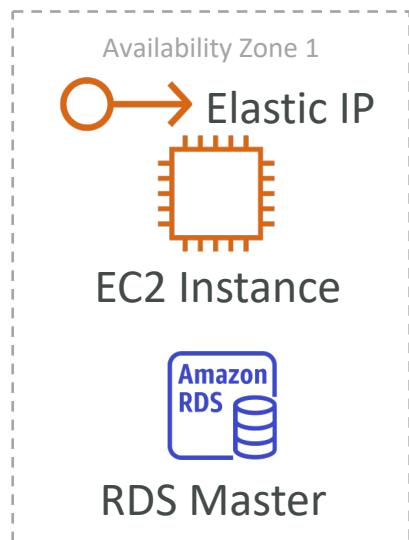
- Support for many platforms:
  - Go
  - Java SE
  - Java with Tomcat
  - .NET on Windows Server with IIS
  - Node.js
  - PHP
  - Python
  - Ruby
  - Packer Builder
  - Single Container Docker
- Multicontainer Docker
- Preconfigured Docker
- If not supported, you can write your custom platform (advanced)
- Beanstalk is great to “Replatform” your application from on-premises to the cloud

# Elastic Beanstalk

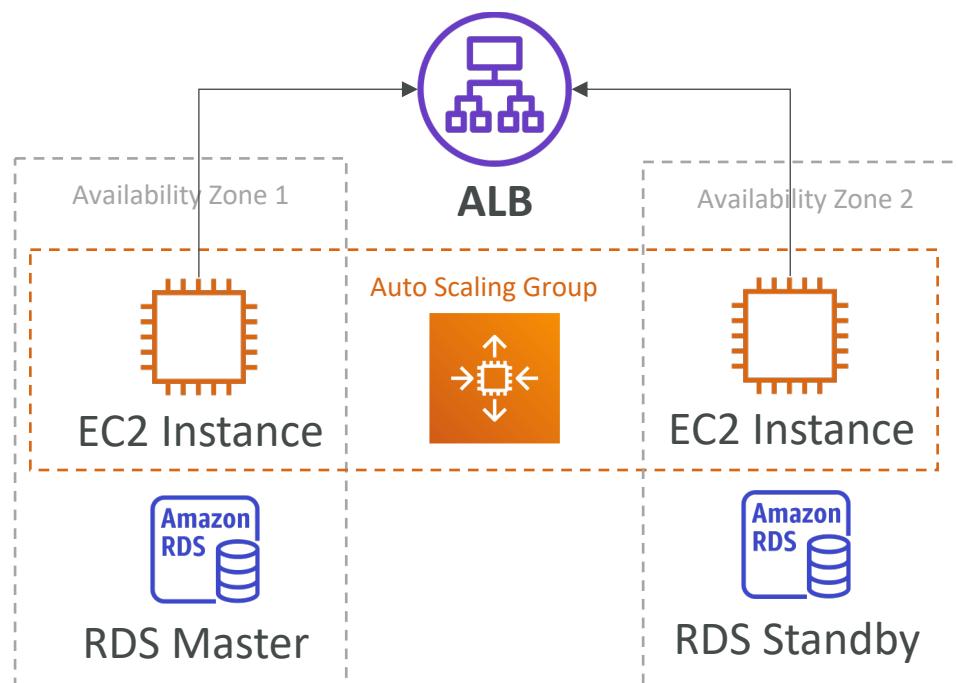
- Managed service
  - Instance configuration / OS is handled by Beanstalk
  - Deployment strategy is configurable but performed by Elastic Beanstalk
- Just the application code is the responsibility of the developer
- Three architecture models:
  - Single Instance deployment: good for dev
  - LB + ASG: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

# Beanstalk Environments

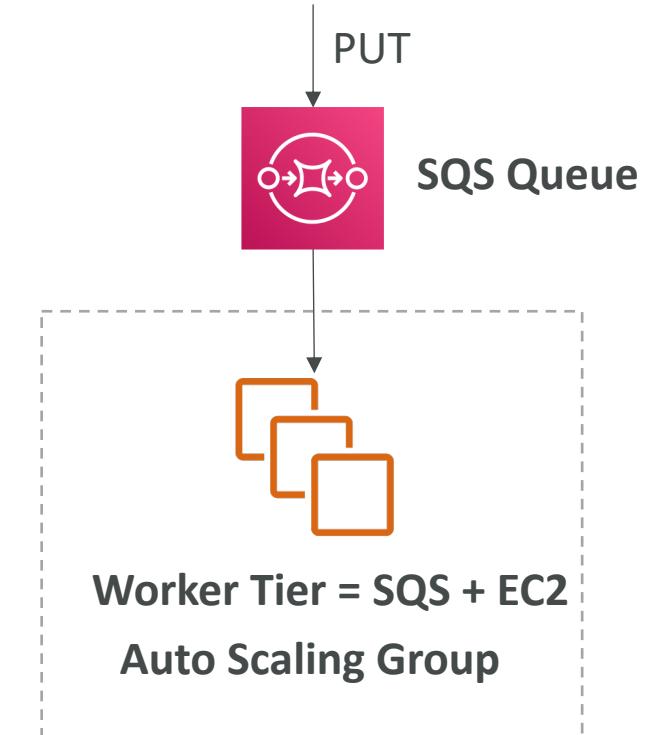
**Single Instance**  
Great for dev



**High Availability with Load Balancer**  
Great for prod

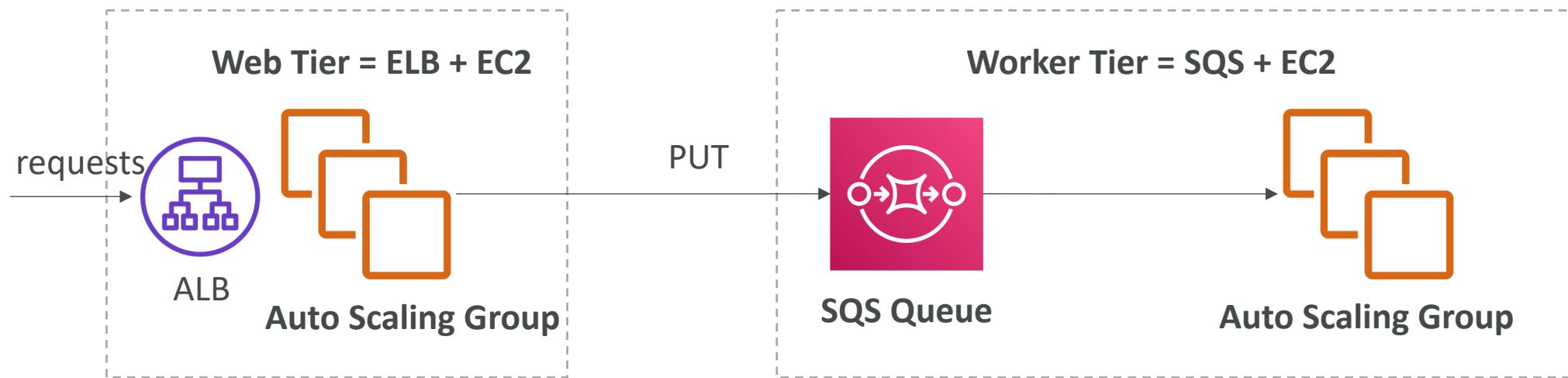


**Worker Tier**



# Web Server vs Worker Environment

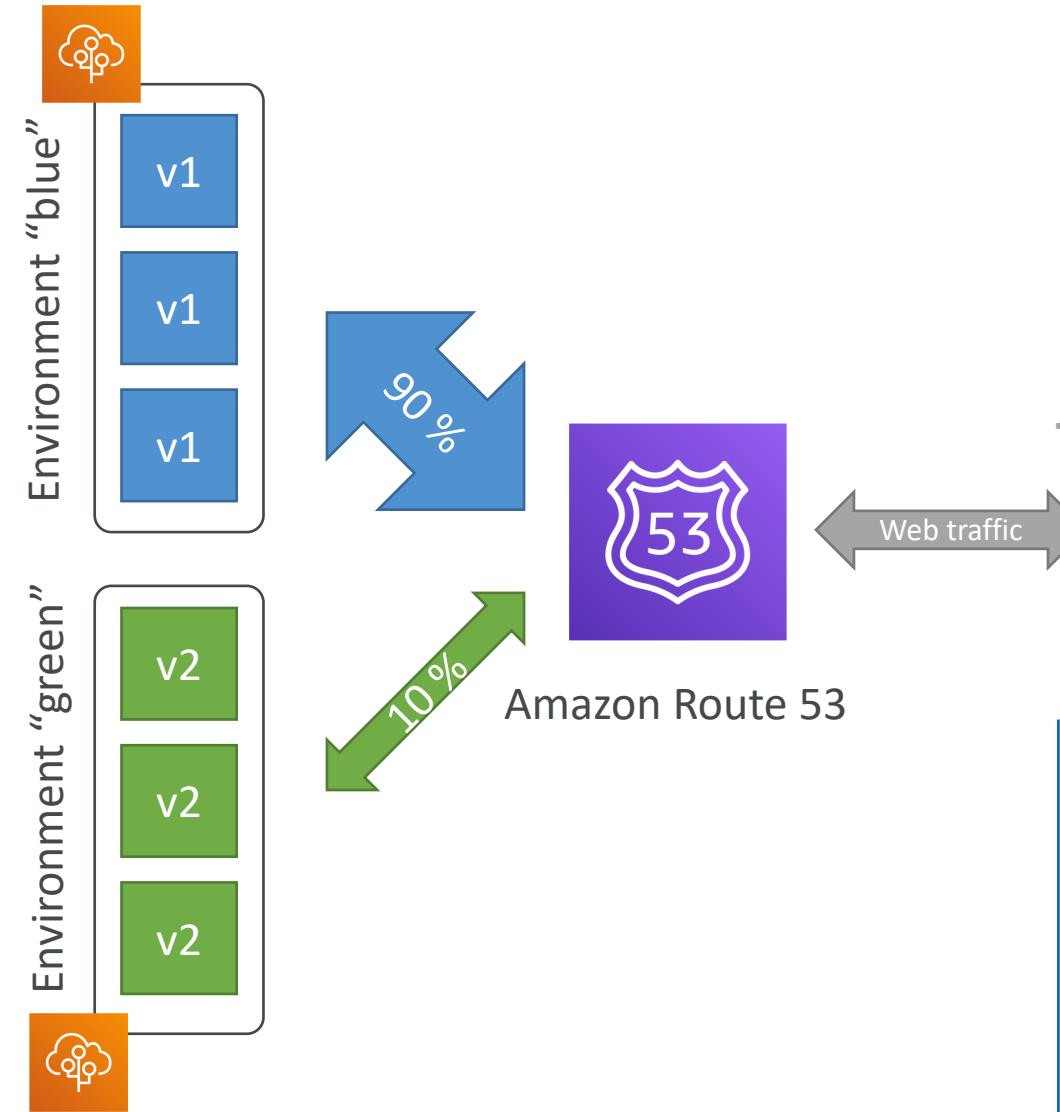
- If your application performs tasks that are long to complete, offload these tasks to a dedicated **worker environment**
- Decoupling your application into two tiers is common
- Example: processing a video, generating a zip file, etc
- You can define periodic tasks in a file `cron.yaml`



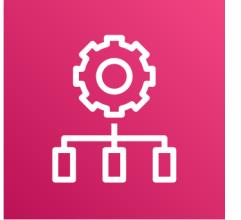
# Elastic Beanstalk Deployment

## Blue / Green

- Not a “direct feature” of Elastic Beanstalk
- Zero downtime and release facility
- Create a new “stage” environment and deploy v2 there
- The new environment (green) can be validated independently and roll back if issues
- Route 53 can be setup using weighted policies to redirect a little bit of traffic to the stage environment
- Using Beanstalk, “swap URLs” (DNS swap) when done with the environment test



# AWS OpsWorks



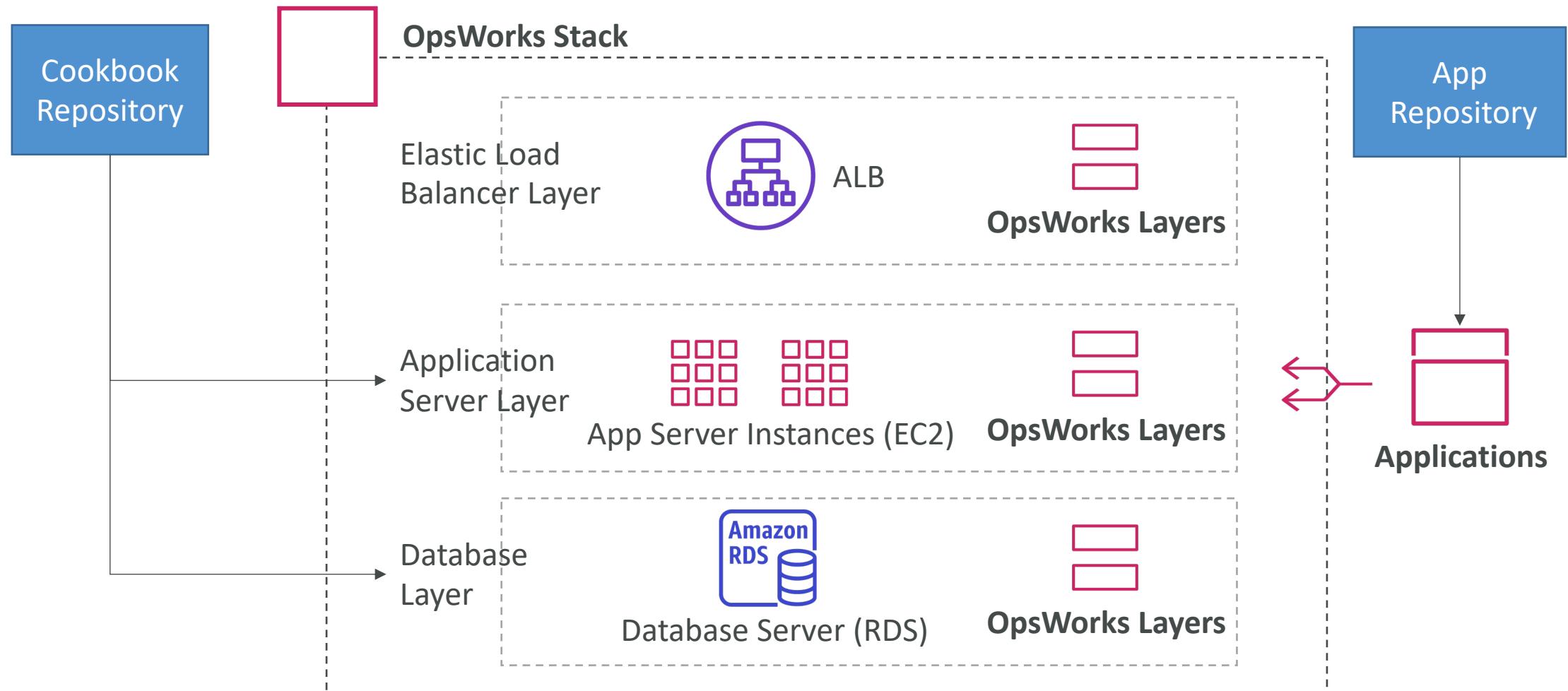
- Chef & Puppet help you perform server configuration automatically, or repetitive actions
  - They work great with EC2 & On Premise VM
  - AWS OpsWorks = Managed Chef & Puppet
  - It's an alternative to AWS SSM
- 
- If you're already using cookbooks (chef) on premise, OpsWorks is good
  - Migrating from other tech to OpsWorks or vice-versa is not easy



# Quick word on Chef / Puppet

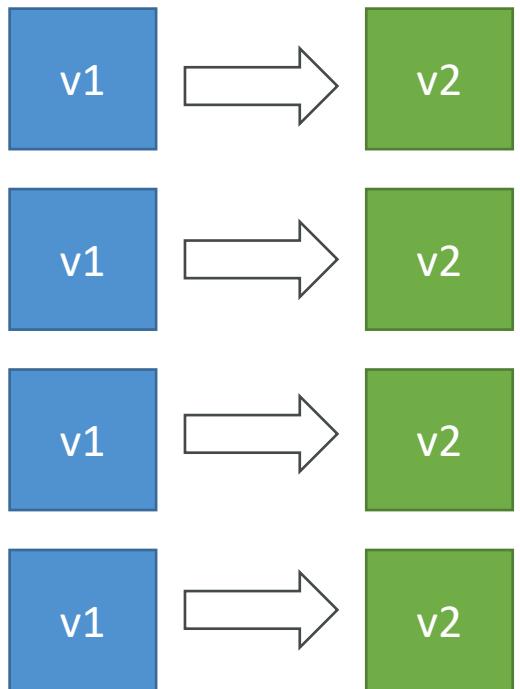
- They help with managing configuration as code
- Helps in having consistent deployments
- Works with Linux / Windows
- Can automate: user accounts, cron, ntp, packages, services...
- They leverage “Recipes”, “Cookbooks” or “Manifests”
- Chef / Puppet have similarities with SSM / Beanstalk / CloudFormation but they’re open-source tools that work cross-cloud

# OpsWorks Architecture



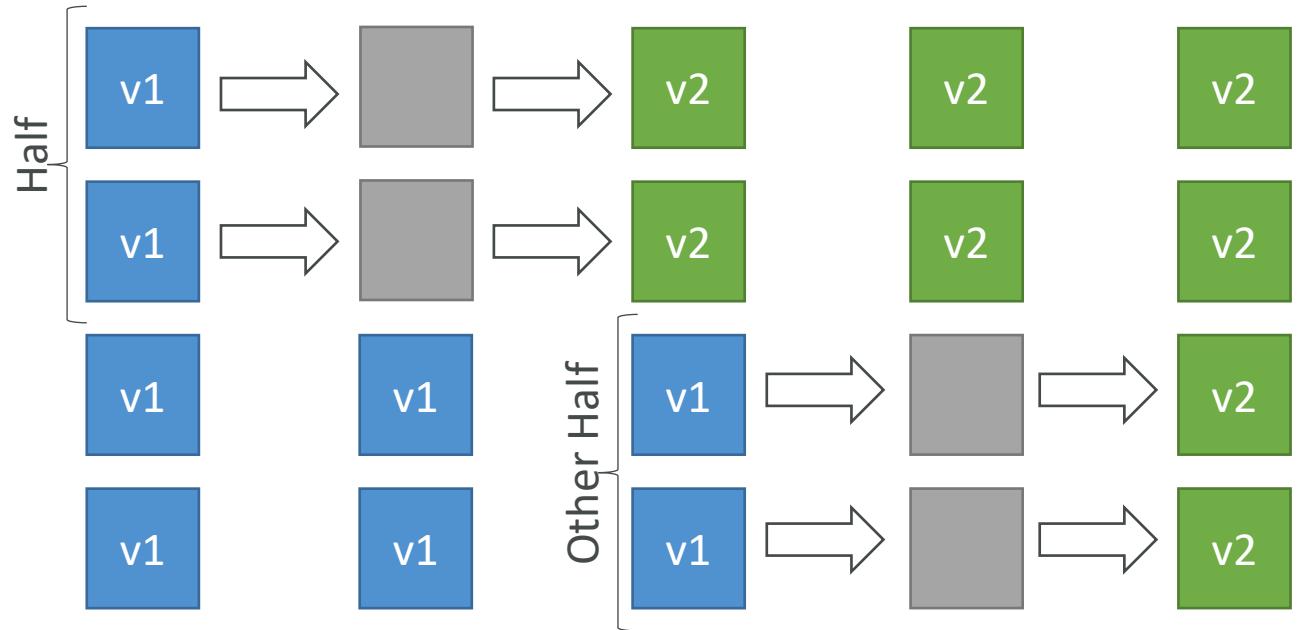
# AWS CodeDeploy

- We want to deploy our application automatically to many EC2 instances
- These instances are not managed by Elastic Beanstalk
- There are several ways to handle deployments using open source tools (Ansible, Terraform, Chef, Puppet, etc...)
- We can use the managed Service AWS CodeDeploy
- CodeDeploy can deploy to: **EC2, ASG, ECS & Lambda**



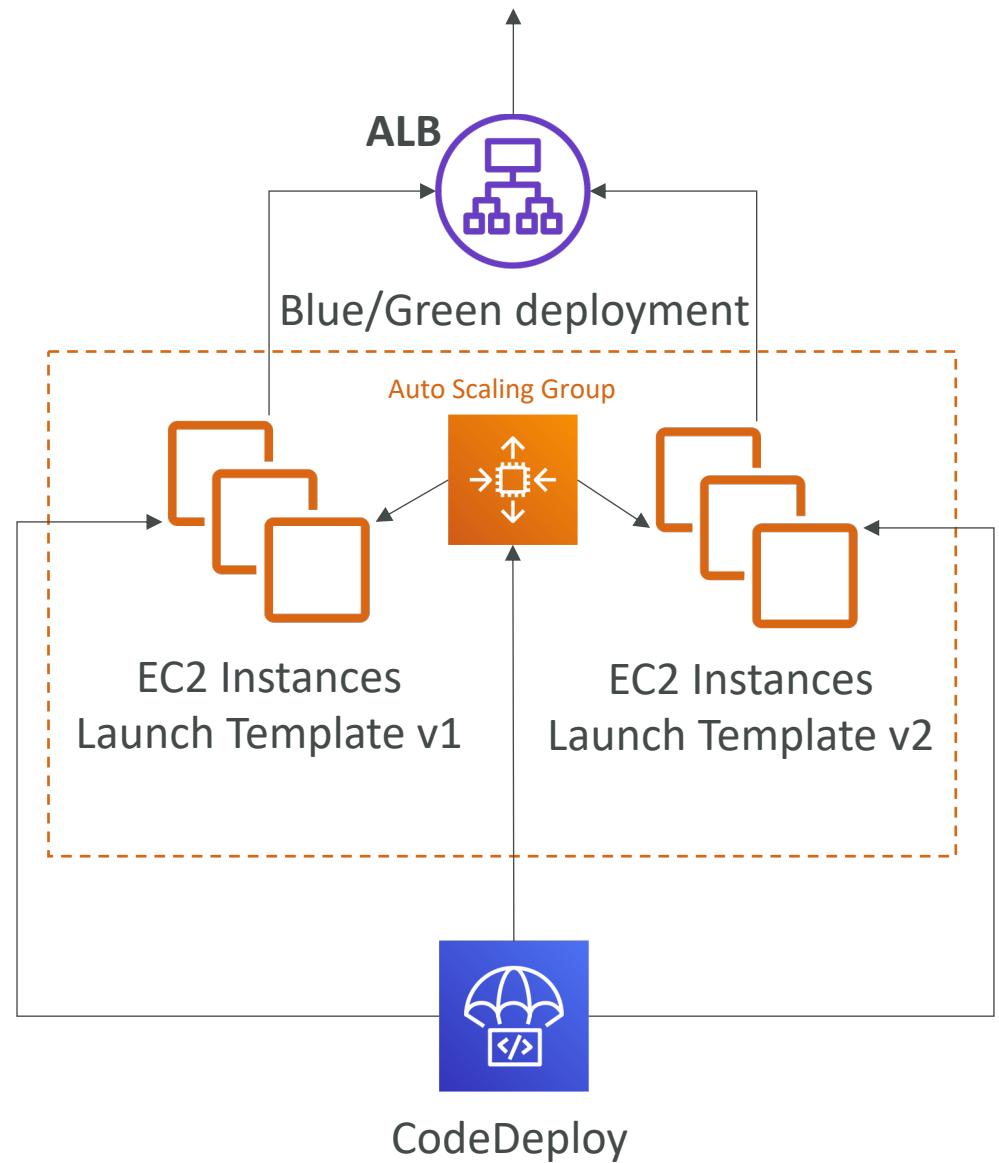
# CodeDeploy to EC2

- Define how to deploy the application using appspec.yml + deployment strategy
- Will do in-place update to your fleet of EC2 instances
- Can use hooks to verify the deployment after each deployment phase



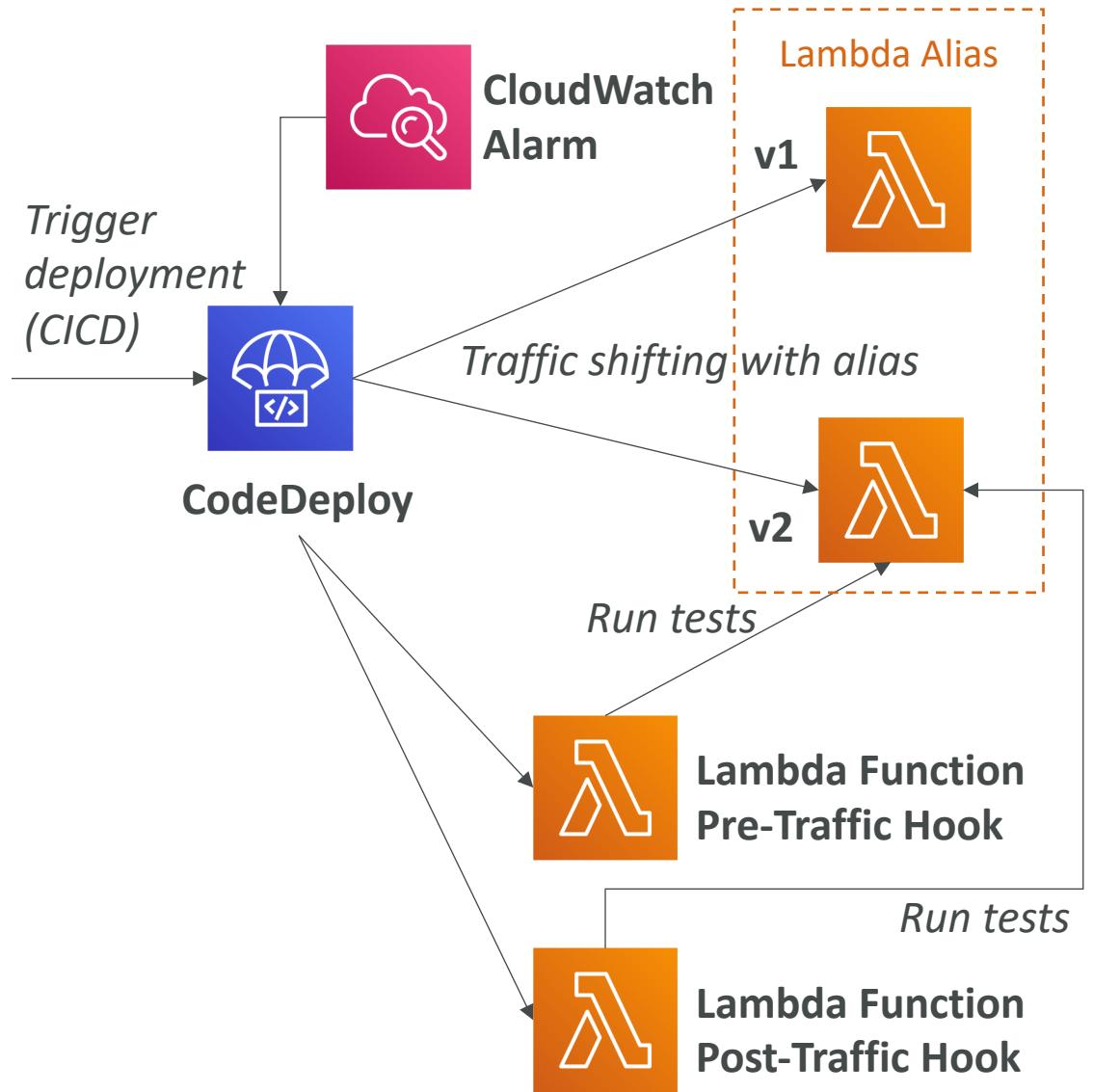
# CodeDeploy to ASG

- In place updates:
  - Updates current existing EC2 instances
  - Instances newly created by an ASG will also get automated deployments
- Blue / green deployment:
  - A new auto-scaling group is created (settings are copied)
  - Choose how long to keep the old instances
  - Must be using an ELB



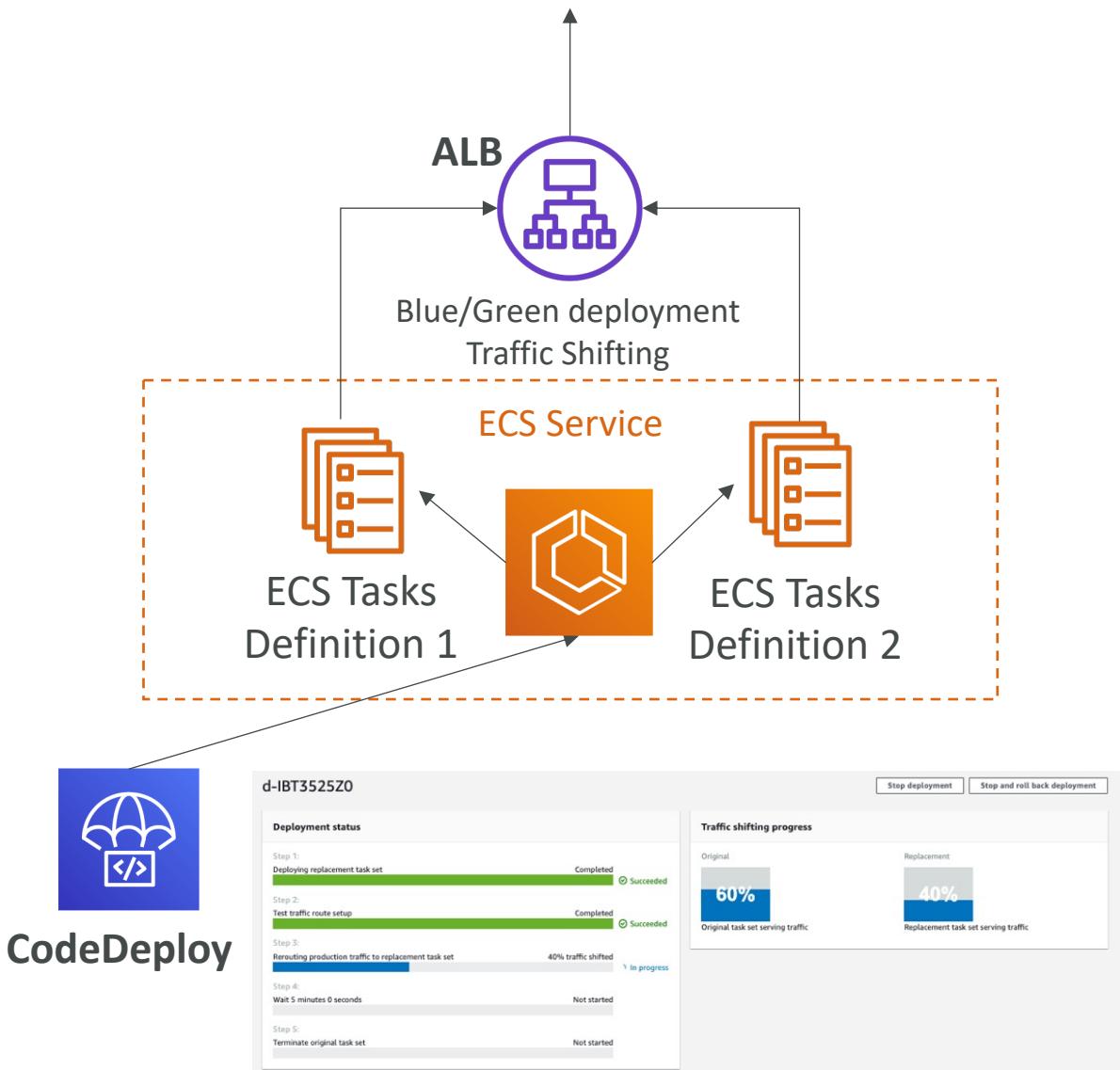
# CodeDeploy to AWS Lambda

- Traffic Shifting feature
- Pre and Post traffic hooks  
features to validate deployment  
(before the traffic shift starts  
and after it ends)
- Easy & automated rollback  
using CloudWatch Alarms
- SAM framework natively uses  
CodeDeploy



# CodeDeploy to ECS

- Support for Blue/Green deployments for Amazon ECS and AWS Fargate
- Setup is done within the ECS service definition
- A new task set is created, and traffic is re-routed to the new task test.
- Then if everything is stable for X minutes, the old task set is terminated (so you have time to notice issues)



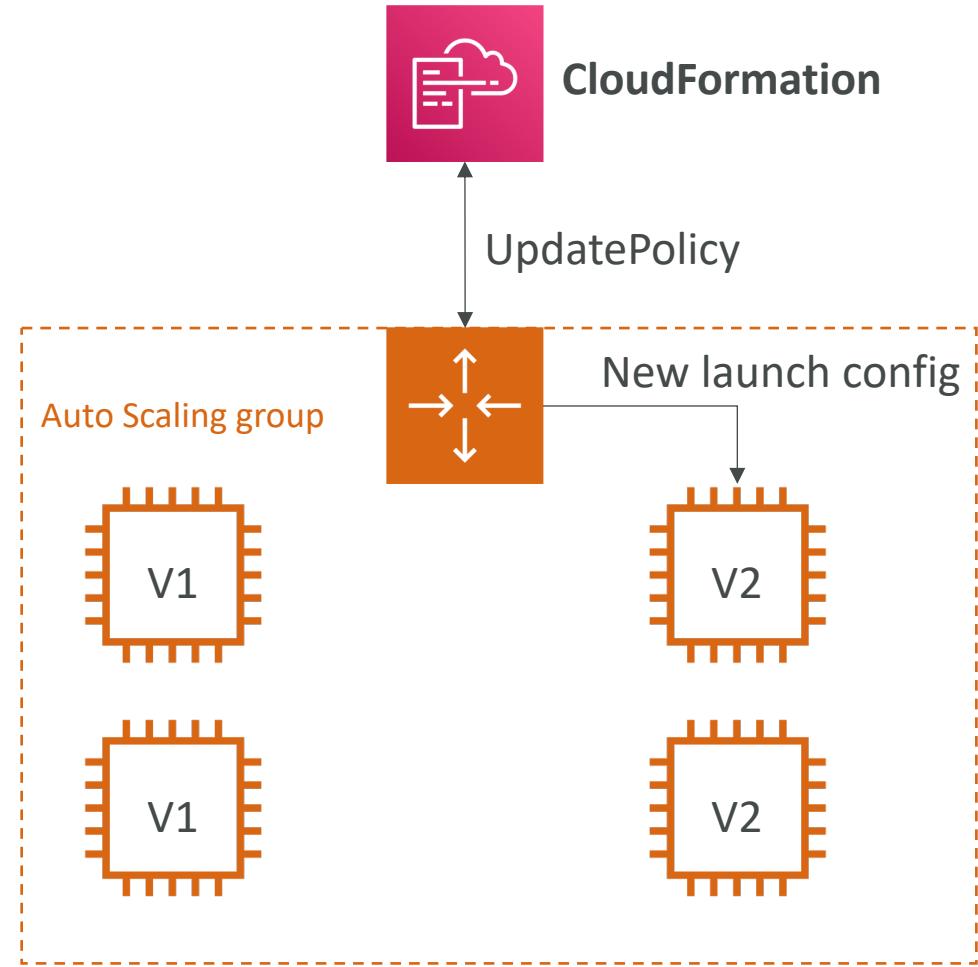
# AWS CloudFormation



- Infrastructure as code (IaC) in AWS
- Portability of stacks across multiple accounts and regions
- Backbone of the Elastic Beanstalk service
- Backbone of the Service Catalog service
- Backbone of the SAM (Serverless Application Model) framework
- Must-know service as a developer / sysops / devops

# CloudFormation & ASG

- CloudFormation manages the ASG, not the underlying EC2
- You can define “success conditions” for the launch of your EC2 instances using a **CreationPolicy**
- You can define “update strategies” for the update of your EC2 instances using an **UpdatePolicy**
- To update the underlying EC2 in an ASG, you have to create a new launch configuration / launch template & use an **UpdatePolicy**



# Retaining Data on Deletes

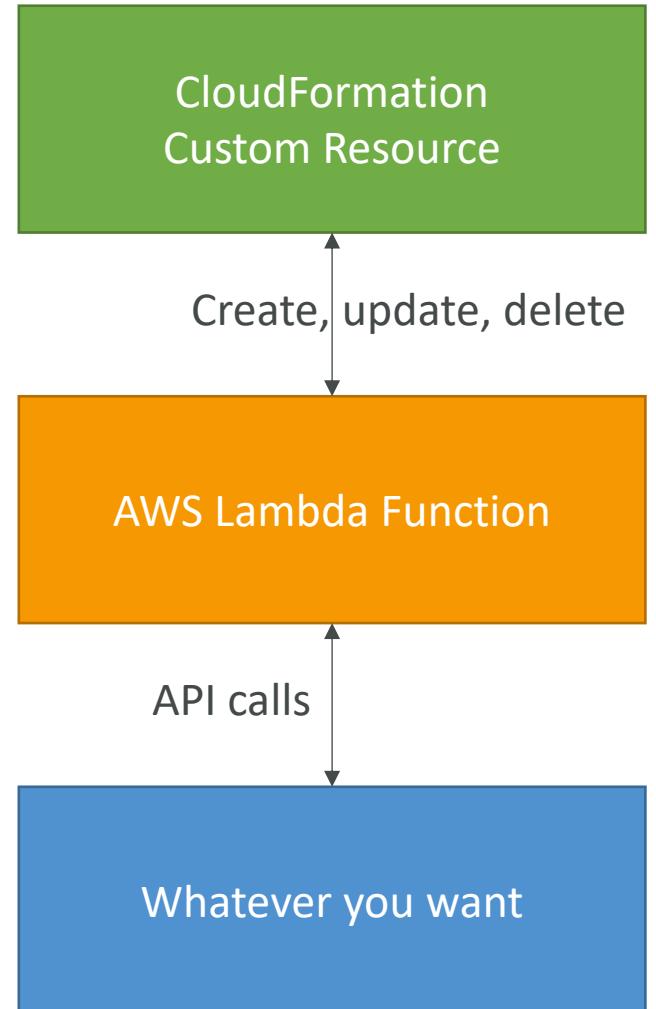
- You can put a `DeletionPolicy` on any resource to control what happens when the CloudFormation template is deleted
- `DeletionPolicy=Retain`:
  - Specify on resources to preserve / backup in case of CloudFormation deletes
  - To keep a resource, specify `Retain` (works for any resource / nested stack)
- `DeletionPolicy=Snapshot`:
  - EBS Volume, ElastiCache Cluster, ElastiCache ReplicationGroup
  - RDS DBInstance, RDS DBCluster, Redshift Cluster
- `DeletePolicy=Delete` (default behavior):
  - Note: for `AWS::RDS::DBCluster` resources, the default policy is Snapshot
  - Note: to delete an S3 bucket, you need to first empty the bucket of its content

# CloudFormation and IAM

- When deploying a CloudFormation stack
  1. it uses the permissions of our own IAM principal
  2. OR assign an IAM role to the stack that can perform the actions
- If you create IAM resources, you need to explicitly provide a “capability” to CloudFormation **CAPABILITY\_IAM** and **CAPABILITY\_NAMED\_IAM**

# CloudFormation Custom Resources (Lambda)

- You can define a Custom Resource in CloudFormation to address any of these use cases:
- An AWS resource is **not yet supported** (new service for example)
- An **on-premises resource**
- Emptying an S3 bucket before being deleted
- Fetch an AMI id
- Anything you want...!



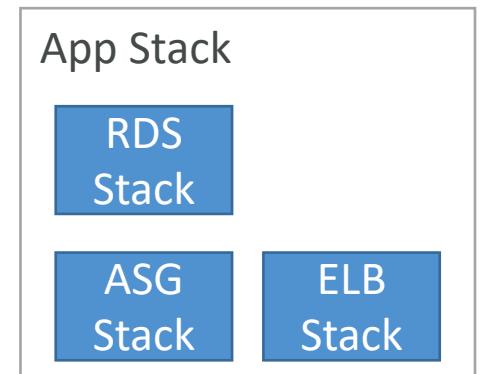
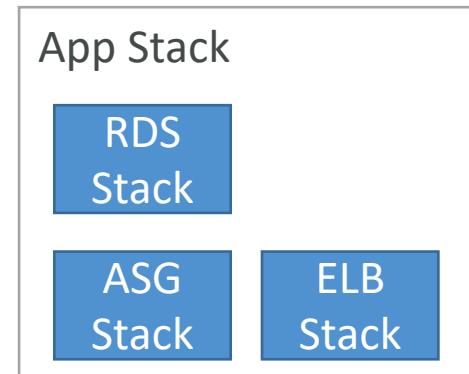
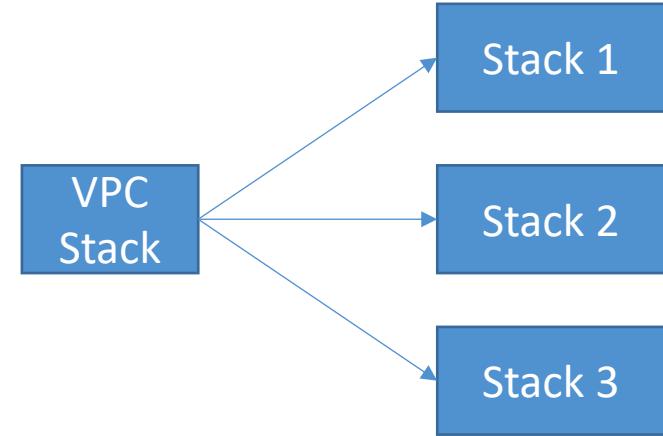
# CloudFormation – Cross vs Nested Stacks

- **Cross Stacks**

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

- **Nested Stacks**

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



# CloudFormation – Others Concepts

- **CloudFormer**
  - Create an AWS CloudFormation template from existing AWS resources
- **ChangeSets**
  - Generate & Preview the CloudFormation changes before they get applied
- **StackSets**
  - Deploy a CloudFormation stack across multiple accounts and regions
- **Stack Policies**
  - Prevent accidental updates / deletes to stack resources

# CloudFormation – Integration with Secrets Manager

```
Resources:  
  # Secret resource with a randomly generated password in its SecureString JSON  
  MyRDSDBInstanceRotationSecret:  
    Type: AWS::SecretsManager::Secret  
    Properties:  
      GenerateSecretString:  
        SecretStringTemplate: '{"username": "admin"}'  
        GenerateStringKey: password  
        PasswordLength: 16  
        ExcludeCharacters: "\"@/\\\""  
  
  # RDS Instance resource. Its master username and password use dynamic references  
  # to resolve values from Secrets Manager  
  MyRDSDBInstance:  
    Type: AWS::RDS::DBInstance  
    Properties:  
      DBInstanceClass: db.t2.micro  
      Engine: mysql  
      MasterUsername: !Sub "{{resolve:secretsmanager:${MyRDSDBInstanceRotationSecret}:username}}"  
      MasterUserPassword: !Sub "{{resolve:secretsmanager:${MyRDSDBInstanceRotationSecret}:password}}"  
  
  # SecretTargetAttachment resource which updates the referenced Secret with properties  
  # about the referenced RDS instance  
  SecretRDSDBInstanceAttachment:  
    Type: AWS::SecretsManager::SecretTargetAttachment  
    Properties:  
      TargetType: AWS::RDS::DBInstance  
      SecretId: !Ref MyRDSDBInstanceRotationSecret  
      TargetId: !Ref MyRDSDBInstance
```

secret is generated

reference secret in RDS DB instance

link the secret to RDS DB instance

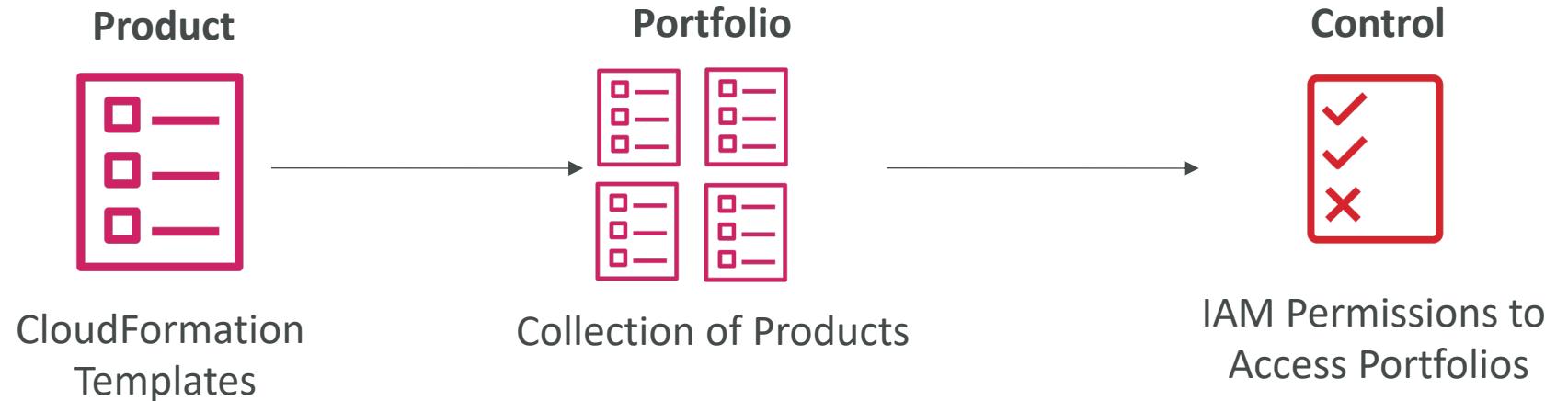
# AWS Service Catalog



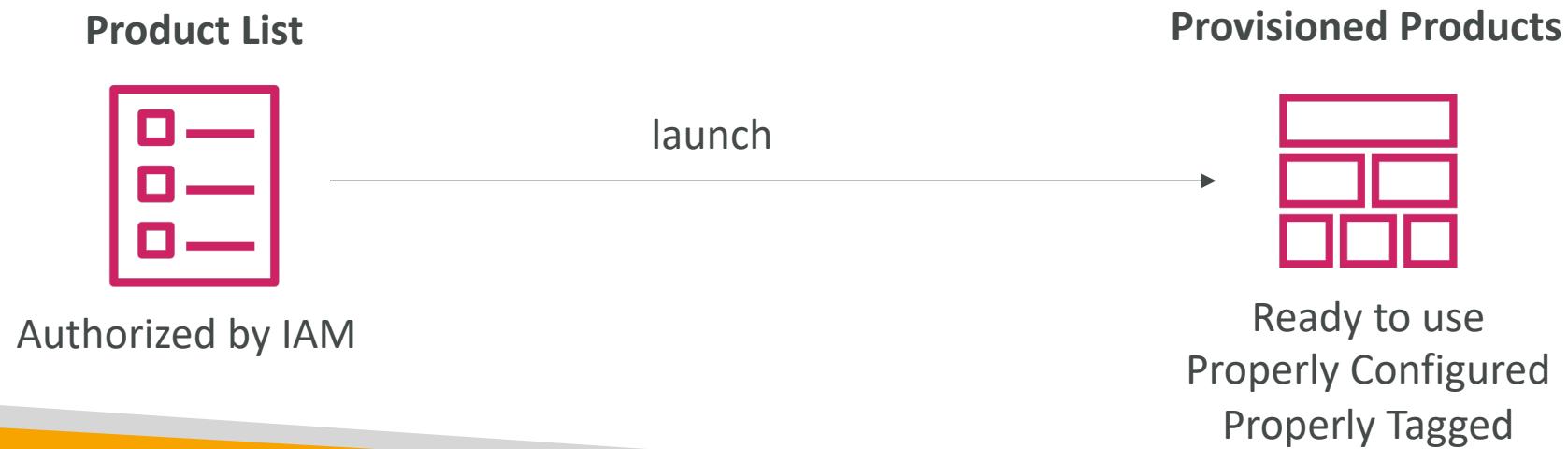
- Users that are new to AWS have too many options, and may create stacks that are not compliant / in line with the rest of the organization
- Some users just want a quick **self-service portal** to launch a set of authorized products pre-defined by admins
- Includes: virtual machines, databases, storage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

# Service Catalog diagram

## ADMIN TASKS



## USER TASKS



# AWS Service Catalog



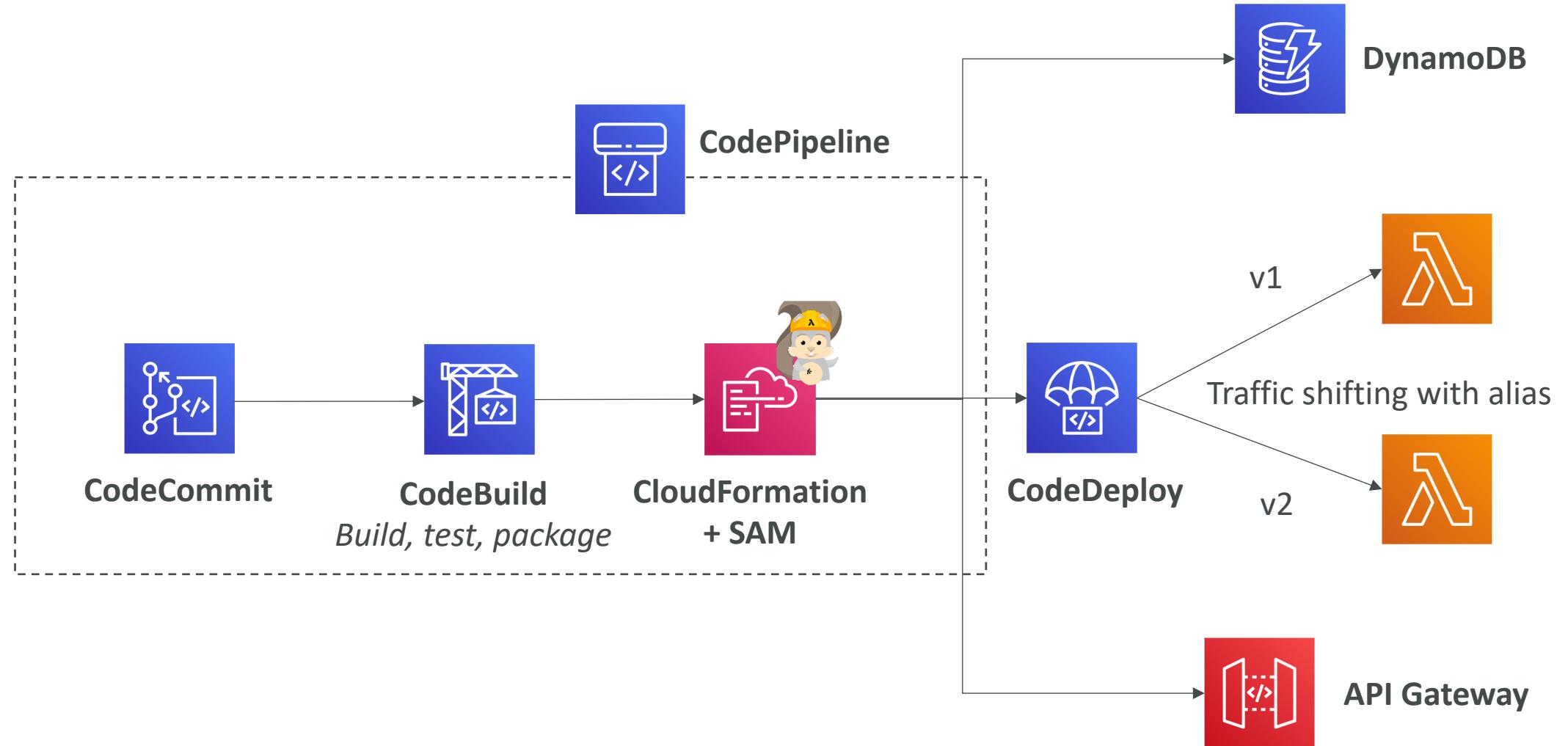
- Create and manage catalogs of IT services that are approved on AWS
- The “products” are CloudFormation templates
- Ex: Virtual machine images, Servers, Software, Databases, Regions, IP address ranges
- **CloudFormation helps ensure consistency, and standardization by Admins**
- They are assigned to Portfolios (teams)
- Teams are presented a self-service portal where they can launch the products
- All the deployed products are centrally managed deployed services
- **Helps with governance, compliance, and consistency**
- Can give user access to launching products without requiring deep AWS knowledge
- Integrations with “self-service portals” such as ServiceNow

# AWS SAM - Serverless Application Model



- SAM = Serverless Application Model
- Framework for developing and deploying serverless applications
- All the configuration is YAML code. Examples:
  - Lambda Functions (AWS::Serverless::Function)
  - DynamoDB tables (AWS::Serverless::SimpleTable)
  - API Gateway (AWS::Serverless::API)
  - StepFunction - State Machine (AWS::Serverless::StateMachine)
- SAM can help you to run Lambda, API Gateway, DynamoDB locally
- SAM can use CodeDeploy to deploy Lambda functions (traffic shifting)
- Leverages CloudFormation in the backend

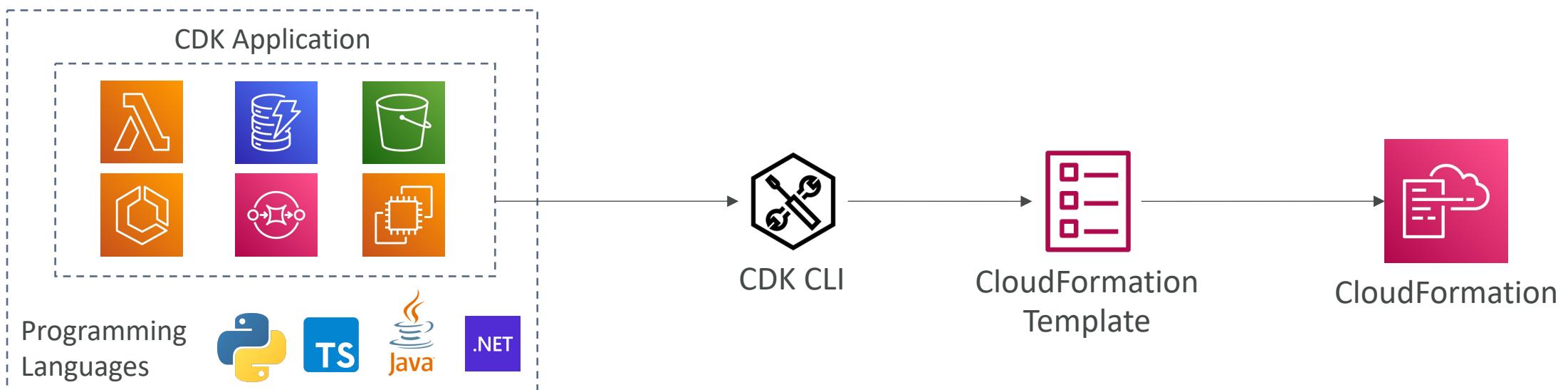
# CICD Architecture for SAM





# AWS Cloud Development Kit (CDK)

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



# CDK Example

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

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

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

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

# Deployment Options

- Vanilla EC2 with User Data (just for the first launch)
- Build an AMI for things that are slow to install (runtimes, updates, tools), and use EC2 user data for quick runtime setup
- Auto Scaling Group with launch template (AMI)
- **CodeDeploy** (no new AMI – application deployments)
  - In-place on EC2
  - In-place on ASG
  - New instances on ASG
  - Traffic shifting for AWS Lambda
  - New task set for ECS + traffic shifting
- **Elastic Beanstalk**
  - In-place all at once upgrades
  - Rolling upgrades (with or without additional instances)
  - Immutable upgrades (new instances)
  - Blue / Green (entirely new stack)
- **OpsWorks**
  - For chef / puppet stacks only
  - Can manage ELB and EC2 instances
  - Cannot manage an ASG
- **SAM Framework**
  - Leverages CloudFormation & CodeDeploy
- **CDK**
  - Manage infra with a programming language
  - Leverages CloudFormation

# AWS Systems Manager Overview



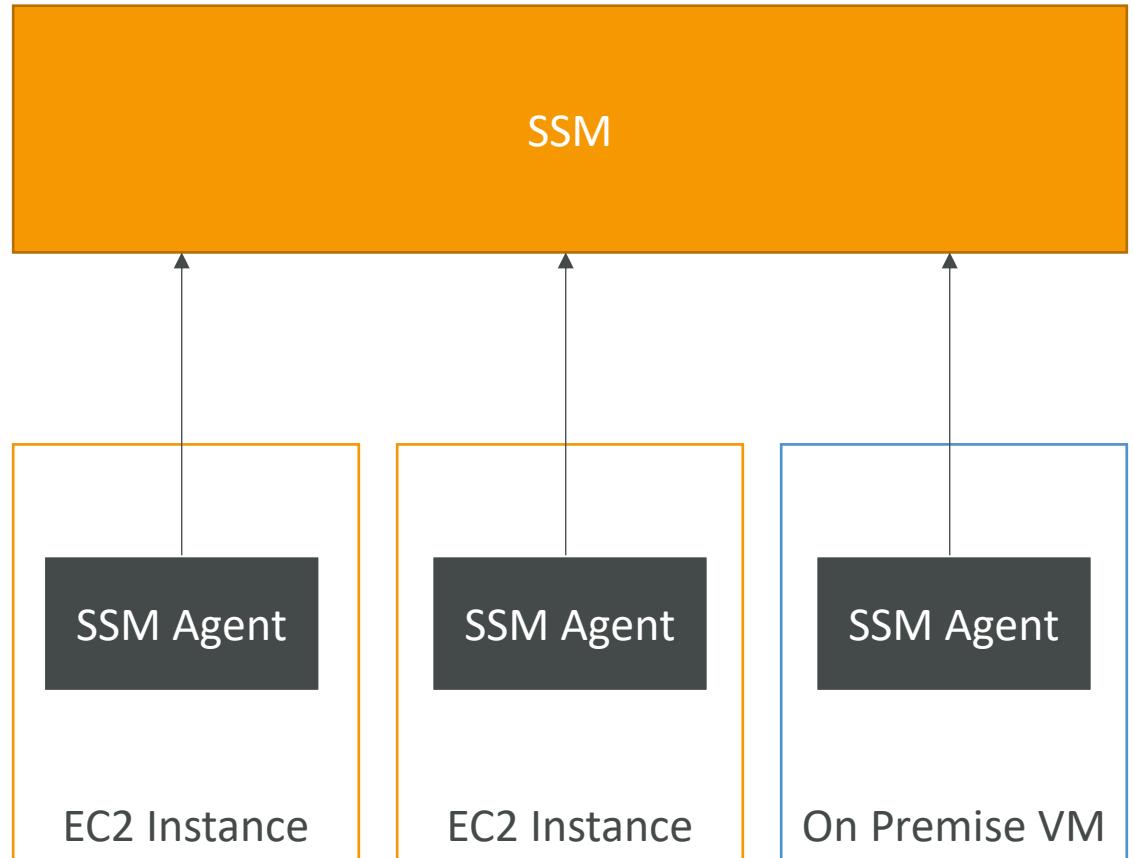
- Helps you manage your EC2 and on-premises systems at scale
- Get operational insights about the state of your infrastructure
- Easily detect problems
- Patching automation for enhanced compliance
- Works for both Windows and Linux OS
- Integrated with CloudWatch metrics / dashboards
- Integrated with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

# AWS Systems Manager Features

- Resource Groups
  - Insights:
    - Insights Dashboard
    - Inventory: discover and audit the software installed
    - Compliance
  - Parameter Store
- Action:
- Automation (shut down EC2, create AMIs)
  - Run Command
  - Session Manager
  - Patch Manager
  - Maintenance Windows
  - State Manager: define and maintaining configuration of OS and applications

# How Systems Manager works

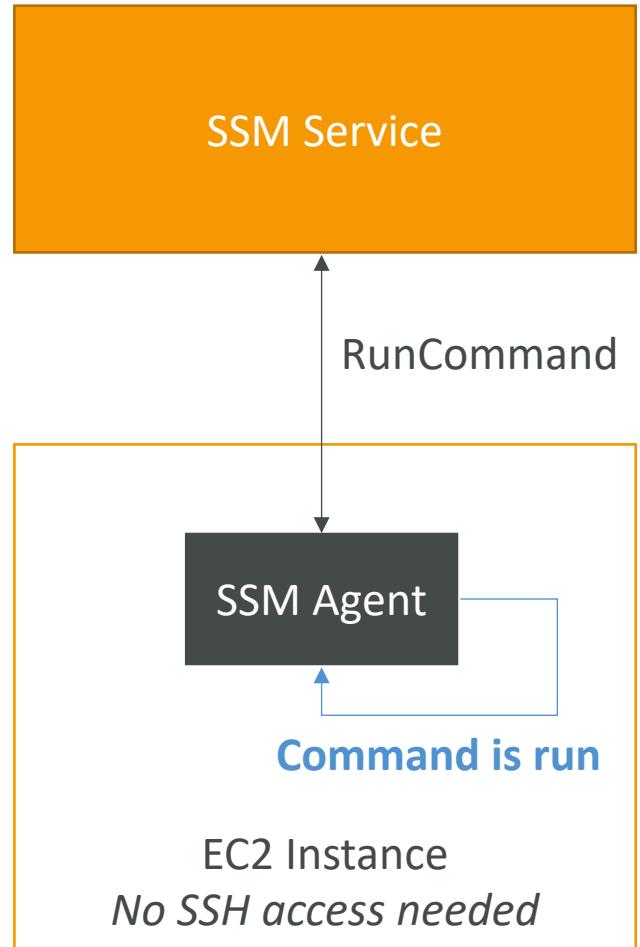
- We need to install the SSM agent onto the systems we control
- Installed by default on Amazon Linux AMI & some Ubuntu AMI
- If an instance can't be controlled with SSM, it's probably an issue with the SSM agent!
- Make sure the EC2 instances have a proper IAM role to allow SSM actions



# AWS Systems Manager

## Run Command

- Execute a document (= script) or just run a command
- Run command across multiple instances (using resource groups)
- Rate Control / Error Control
- Integrated with IAM & CloudTrail
- No need for SSH
- Results in the console

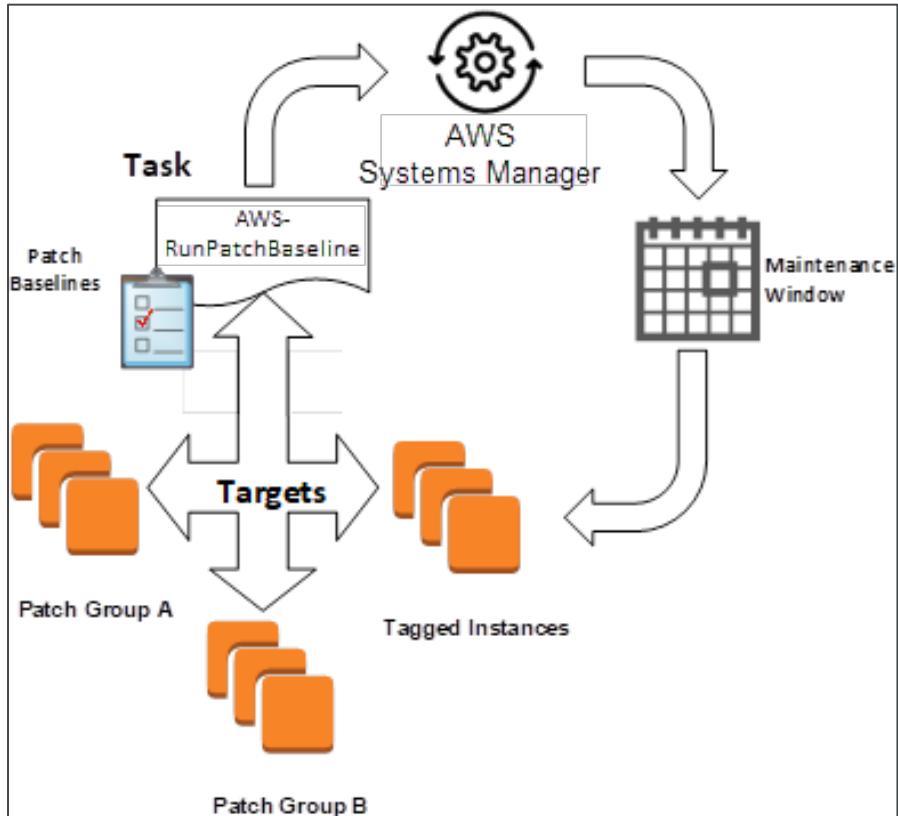


# SSM Patch Manager – Predefined Patch Baselines

- Defines which patches should or shouldn't be installed on your instances
- Linux:
  - AWS-AmazonLinux2DefaultPatchBaseline
  - AWS-CentOSDefaultPatchBaseline
  - AWS-RedHatDefaultPatchBaseline
  - AWS-SuseDefaultPatchBaseline
  - AWS-UbuntuDefaultPatchBaseline
- Windows: (patches are auto-approved 7 days after the release)
  - **AWS-DefaultPatchBaseline**: install OS patch CriticalUpdates & SecurityUpdates
  - AWS-WindowsPredefinedPatchBaseline-OS: same as “AWS-DefaultPatchBaseline”
  - AWS-WindowsPredefinedPatchBaseline-OS-Applications: also updates Microsoft applications
- Can define your own custom patch baselines as well (OS, classification, severity...)

# SSM Patch Managers – Steps

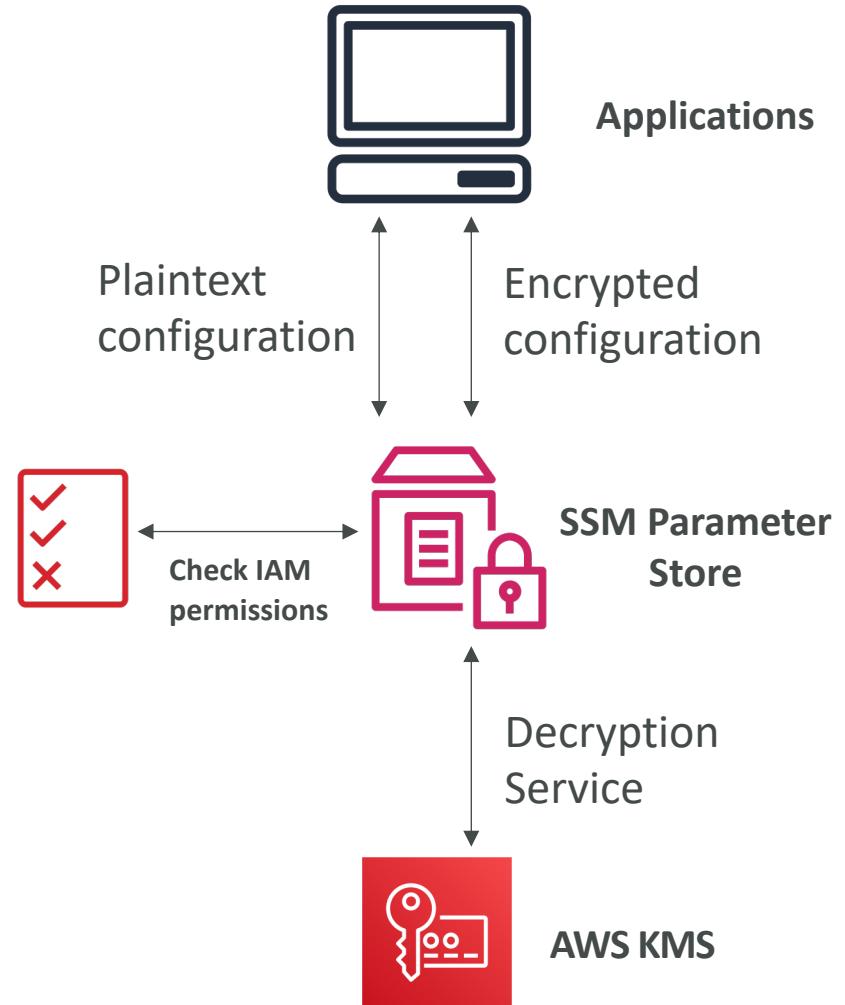
1. Define a **patch baseline** to use (or multiple if you have multiple environments)
2. Define patch groups: define based on tags, example different environments (dev, test, prod) – use tag **Patch Group**
3. Define **Maintenance Windows** (schedule, duration, registered targets/patch groups and registered tasks)
4. Add the **AWS-RunPatchBaseline Run Command** as part of the registered tasks of the Maintenance Window (works cross platform Linux & Windows)
5. Define **Rate Control** (concurrency & error threshold) for the task
6. Monitor Patch Compliance using SSM Inventory



<https://aws.amazon.com/blogs/mt/patching-your-windows-ec2-instances-using-aws-systems-manager-patch-manager/>

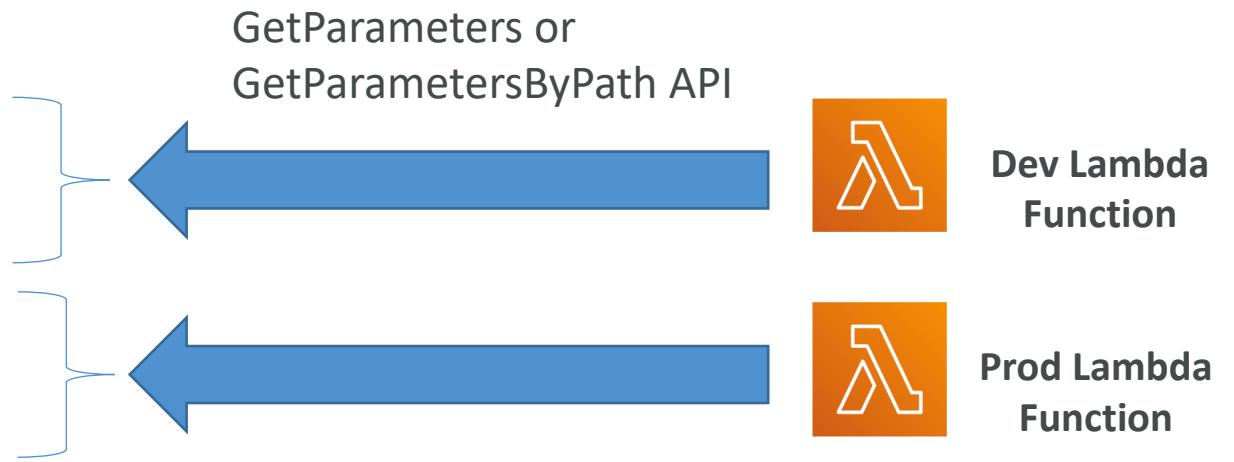
# AWS Parameter Store

- Secure storage for configuration and secrets
- Optional Seamless Encryption using KMS
- Serverless, scalable, durable, easy SDK, free
- Version tracking of configurations / secrets
- Configuration management using path & IAM
- Notifications with CloudWatch Events
- Integration with CloudFormation
- Can retrieve secrets from Secrets Manager using the SSM Parameter Store API



# AWS Parameter Store Hierarchy

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



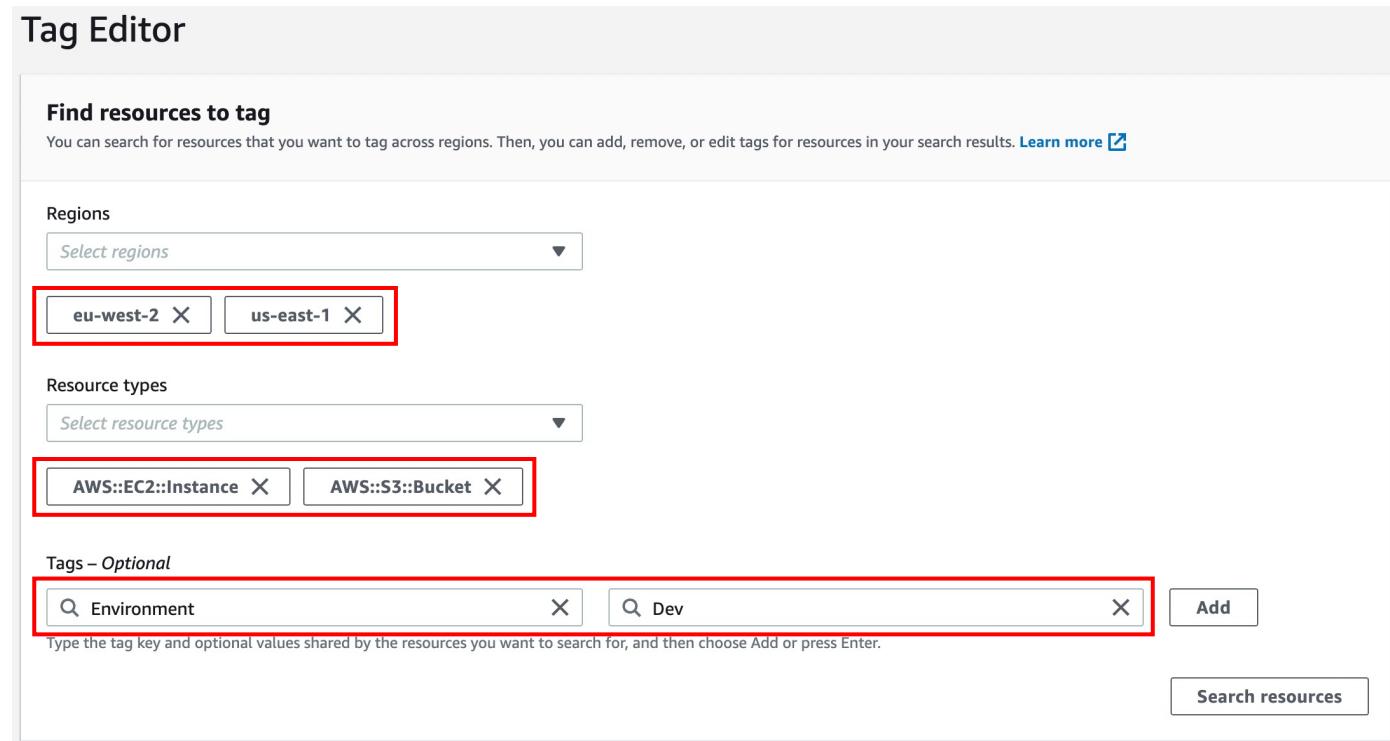
# Cost Control Section

# AWS Cost Allocation Tags

- With Tags we can track resources that relate to each other
- With Cost Allocation Tags we can enable detailed costing reports
- Just like Tags, but they show up as columns in Reports
- AWS Generated Cost Allocation Tags
  - Automatically applied to the resource you create
  - Starts with Prefix **aws:** (e.g. `aws: createdBy`)
  - They're not applied to resources created before the activation
- User tags
  - Defined by the user
  - Starts with Prefix **user:**
- Cost Allocation Tags just appear in the Billing Console
- Takes up to 24 hours for the tags to show up in the report

# AWS Tag Editor

- Allows you to manage tags of multiple resources at once
- You can add/update/delete tags
- Search tagged/untagged resources in all AWS Regions





# Trusted Advisor

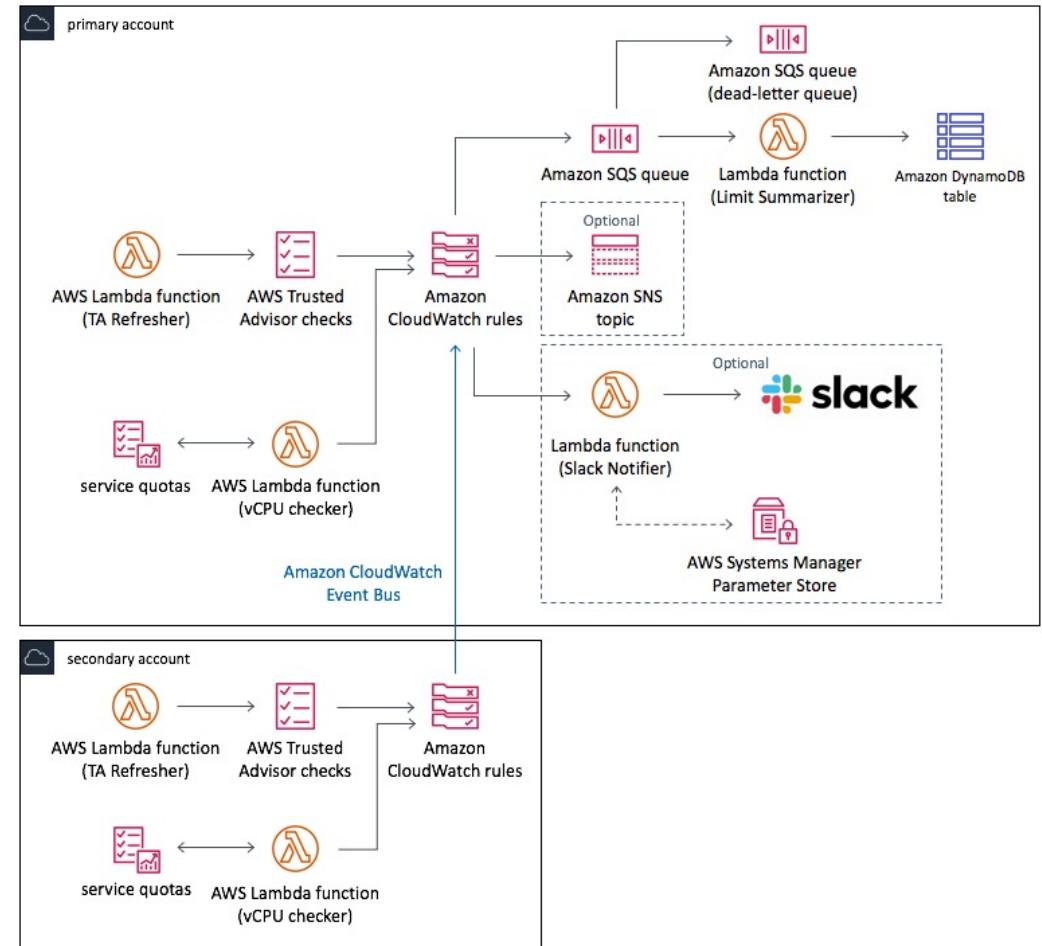
- No need to install anything – high level AWS account assessment
- Analyze your AWS accounts and provides recommendation:
  - Cost Optimization & Recommendations
  - Performance
  - Security
  - Fault Tolerance
  - Service Limits
- Core Checks and recommendations – all customers
- Can enable weekly email notification from the console
- Full Trusted Advisor – Available for Business & Enterprise support plans
  - Ability to set CloudWatch alarms when reaching limits
  - Programmatic Access using AWS Support API

# AWS Support Plans

	<u>Basic Support</u>	<u>Developer</u>	<u>Business</u>	<u>Enterprise</u>
	<i>included for all AWS customers and free</i>	<i>Recommended if you are experimenting or testing in AWS.</i>	<i>Recommended if you have production workloads in AWS.</i>	<i>Recommended if you have business and/or mission critical workloads in AWS.</i>
<b>AWS Trusted Advisor Best Practice Checks</b>	7 Core checks	7 Core checks	<b>Full set of checks + Programmatic Access using AWS Support API</b>	<b>Full set of checks + Programmatic Access using AWS Support API</b>
<b>Enhanced Technical Support</b>	24x7 access to customer service, documentation, whitepapers, and support forums.	Business hours email access to Cloud Support Associates  Unlimited cases / 1 primary contact	24x7 phone, email, and chat access to Cloud Support Engineers  Unlimited cases / unlimited contacts (IAM supported)	24x7 phone, email, and chat access to Cloud Support Engineers  Unlimited cases / unlimited contacts (IAM supported)
<b>Case Severity / Response Times</b>		General guidance: < 24 business hours**  System impaired: < 12 business hours**	General guidance: < 24 hours System impaired: < 12 hours Production system impaired: < 4 hours Production system down: < 1 hour	General guidance: < 24 hours System impaired: < 12 hours Production system impaired: < 4 hours Production system down: < 1 hour Business-critical system down: < 15 minutes

# Trusted Advisor – Good to know

- Can check if an S3 bucket is made public
  - But cannot check for S3 objects that are public inside of your bucket!
  - Use CloudWatch Events / S3 Events instead
- Service Limits
  - Limits can only be monitored in Trusted Advisor (cannot be changed)
  - Cases have to be created manually in **AWS Support Centre** to increase limits
  - OR use the new **AWS Service Quotas** service (new service - has an API)



# EC2 Instance Launch Types

- **On Demand Instances:** short workload, predictable pricing, reliable
- **Spot Instances:** short workloads, for cheap, can lose instances (not reliable)
- **Reserved:** (MINIMUM 1 year)
  - Reserved Instances: long workloads
  - Convertible Reserved Instances: long workloads with flexible instances
- **Dedicated Instances:** no other customers will share your hardware
- **Dedicated Hosts:** book an entire physical server, control instance placement
  - Great for software licenses that operate at the core, or socket level
  - Can define **host affinity** so that instance reboots are kept on the same host

# AWS Savings Plan



- New pricing model to get a discount based on long-term usage
- Commit to a certain type of usage: ex \$10 per hour for 1 to 3 years
- Any usage beyond the savings plan is billed at the on-demand price
- **EC2 Instance Savings plan** (up to 72% - same discount as Standard RIs)
  - Select instance family (e.g. M5, C5...), and locked to a specific region
  - Flexible across size (m5.large to m5.4xlarge), OS (Windows to Linux), tenancy (dedicated or default)
- **Compute Savings plan** (up to 66% - same discount as Convertible RIs)
  - Ability to move between instance family (move from C5 to M5), region (Ireland to US), compute type (EC2, Fargate, Lambda), OS & tenancy
- **SageMaker Savings plan** (up to 64% off)

# S3 Storage Classes

- Amazon S3 Standard - General Purpose
- Amazon S3 Standard-Infrequent Access (IA)
- Amazon S3 One Zone-Infrequent Access
- Amazon S3 Glacier Instant Retrieval
- Amazon S3 Glacier Flexible Retrieval
- Amazon S3 Glacier Deep Archive
- Amazon S3 Intelligent Tiering
- Can move between classes manually or using S3 Lifecycle configurations

# S3 – Other Cost Savings

- S3 Select & Glacier Select: save in network and CPU cost
- S3 Lifecycle Rules: transition objects between tiers
- Compress objects to save space
- S3 Requester Pays:
  - In general, bucket owners pay for all Amazon S3 storage and data transfer costs associated with their bucket
  - With Requester Pays buckets, the requester instead of the bucket owner pays the cost of the request and the data download from the bucket
  - The bucket owner always pays the cost of storing data
  - Helpful when you want to share large datasets with other accounts
  - If an IAM role is assumed, the owner account of that role pays for the request

# S3 Storage Classes

- Amazon S3 Standard - General Purpose
- Amazon S3 Standard-Infrequent Access (IA)
- Amazon S3 One Zone-Infrequent Access
- Amazon S3 Glacier Instant Retrieval
- Amazon S3 Glacier Flexible Retrieval
- Amazon S3 Glacier Deep Archive
- Amazon S3 Intelligent Tiering
- Can move between classes manually or using S3 Lifecycle configurations

# S3 Durability and Availability

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

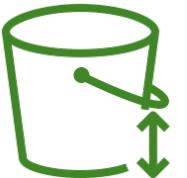


# S3 Standard – General Purpose

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

# S3 Storage Classes – Infrequent Access

- For data that is less frequently accessed, but requires rapid access when needed
- Lower cost than S3 Standard
- Amazon S3 Standard-Infrequent Access (S3 Standard-IA)
  - 99.9% Availability
  - Use cases: Disaster Recovery, backups
- Amazon S3 One Zone-Infrequent Access (S3 One Zone-IA)
  - High durability (99.99999999%) in a single AZ; data lost when AZ is destroyed
  - 99.5% Availability
  - Use Cases: Storing secondary backup copies of on-premise data, or data you can recreate



# Amazon S3 Glacier Storage Classes

- Low-cost object storage meant for archiving / backup
- Pricing: price for storage + object retrieval cost
- **Amazon S3 Glacier Instant Retrieval**
  - Millisecond retrieval, great for data accessed once a quarter
  - Minimum storage duration of 90 days
- **Amazon S3 Glacier Flexible Retrieval** (formerly Amazon S3 Glacier):
  - Expedited (1 to 5 minutes), Standard (3 to 5 hours), Bulk (5 to 12 hours) – free
  - Minimum storage duration of 90 days
- **Amazon S3 Glacier Deep Archive** – for long term storage:
  - Standard (12 hours), Bulk (48 hours)
  - Minimum storage duration of 180 days





# S3 Intelligent-Tiering

- Small monthly monitoring and auto-tiering fee
  - Moves objects automatically between Access Tiers based on usage
  - There are no retrieval charges in S3 Intelligent-Tiering
- 
- *Frequent Access tier (automatic)*: default tier
  - *Infrequent Access tier (automatic)*: objects not accessed for 30 days
  - *Archive Instant Access tier (automatic)*: objects not accessed for 90 days
  - *Archive Access tier (optional)*: configurable from 90 days to 700+ days
  - *Deep Archive Access tier (optional)*: config. from 180 days to 700+ days

# S3 Storage Classes Comparison

	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Durability	99.999999999% == (11 9's)						
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
Min. Storage Duration Charge	None	None	30 Days	30 Days	90 Days	90 Days	180 Days
Min. Billable Object Size	None	None	128 KB	128 KB	128 KB	40 KB	40 KB
Retrieval Fee	None	None	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved	Per GB retrieved

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

# S3 Storage Classes – Price Comparison

Example: us-east-1

	Standard	Intelligent-Tiering	Standard-IA	One Zone-IA	Glacier Instant Retrieval	Glacier Flexible Retrieval	Glacier Deep Archive
Storage Cost (per GB per month)	\$0.023	\$0.0025 - \$0.023	%0.0125	\$0.01	\$0.004	\$0.0036	\$0.00099
Retrieval Cost (per 1000 request)	<b>GET:</b> \$0.0004 <b>POST:</b> \$0.005	<b>GET:</b> \$0.0004 <b>POST:</b> \$0.005	<b>GET:</b> \$0.001 <b>POST:</b> \$0.01	<b>GET:</b> \$0.001 <b>POST:</b> \$0.01	<b>GET:</b> \$0.01 <b>POST:</b> \$0.02	<b>GET:</b> \$0.0004 <b>POST:</b> \$0.03  <b>Expedited:</b> \$10 <b>Standard:</b> \$0.05 <b>Bulk:</b> free	<b>GET:</b> \$0.0004 <b>POST:</b> \$0.05  <b>Standard:</b> \$0.10 <b>Bulk:</b> \$0.025
Retrieval Time	Instantaneous						<b>Expedited</b> (1 – 5 mins) <b>Standard</b> (3 – 5 hours) <b>Bulk</b> (5 – 12 hours)
Monitoring Cost (pet 1000 objects)		\$0.0025					

<https://aws.amazon.com/s3/pricing/>

# AWS Budgets



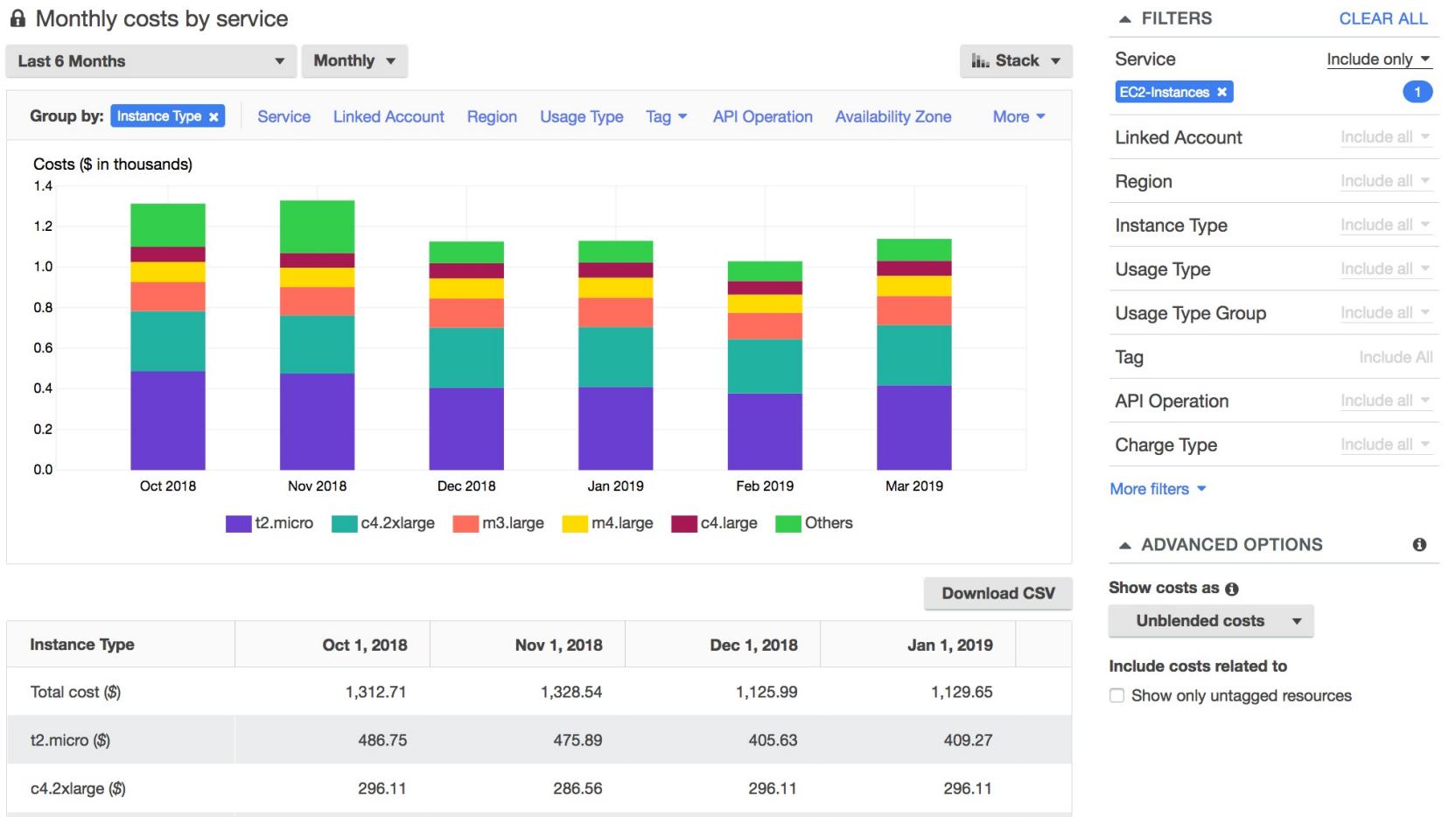
- Create budget and send alarms when costs exceeds the budget
- 4 types of budgets: Usage, Cost, Reservation, Savings Plans
- For Reserved Instances (RI)
  - Track utilization
  - Supports EC2, ElastiCache, RDS, Redshift
- Up to 5 SNS notifications per budget
- Can filter by: Service, Linked Account,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

# Cost Explorer



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

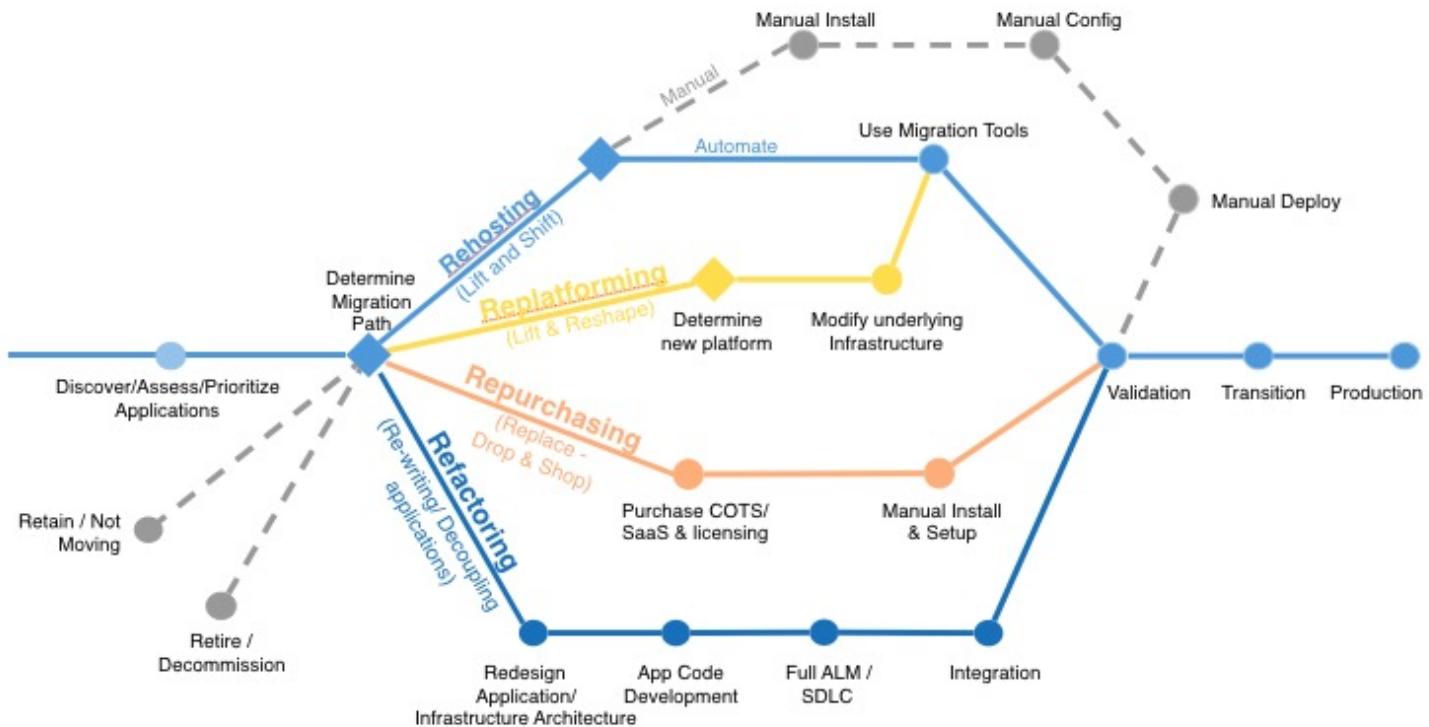
# Cost Explorer – Example



# Migrations Section

# Cloud Migration: The 6R

- From: <https://aws.amazon.com/blogs/enterprise-strategy/6-strategies-for-migrating-applications-to-the-cloud/>



# Cloud Migration: The 6R

- **Rehosting:** “lift and shift”
  - Simple migrations by re-hosting on AWS (applications, databases, data...)
  - No cloud optimizations being done, application is migrated as is
  - Could save as much as 30% on cost
  - Example: Migrate using AWS VM Import/Export, AWS Server Migration Service
- **Replatforming:**
  - Example: migrate your database to RDS
  - Example: migrate your application to Elastic Beanstalk (Java with Tomcat)
  - Not changing the core architecture, but leverage some cloud optimizations

# Cloud Migration: The 6R

- **Repurchase:** “drop and shop”
  - Moving to a different product while moving to the cloud
  - Often you move to a SaaS platform
  - Expensive in the short term, but quick to deploy
  - Example: CRM to Salesforce.com, HR to Workday, CMS to Drupal
- **Refactoring / Re-architecting:**
  - Reimagining how the application is architected using Cloud Native features
  - Driven by the need of the business to add features, scale, performance
  - Example: move an application to Serverless architectures, use AWS S3

# Cloud Migration: The 6R

- Retire
  - Turn off things you don't need (maybe as a result of Re-architecting)
  - Helps with reducing the surface areas for attacks (more security)
  - Save cost, maybe up to 10% to 20%
  - Focus your attention on resources that must be maintained
- Retain
  - Do nothing for now (for simplicity, cost reason, importance...)
  - It's still a decision to make in a Cloud Migration

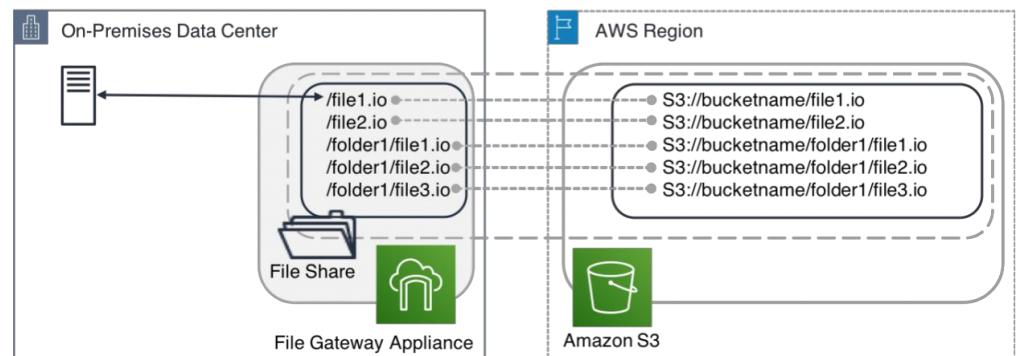
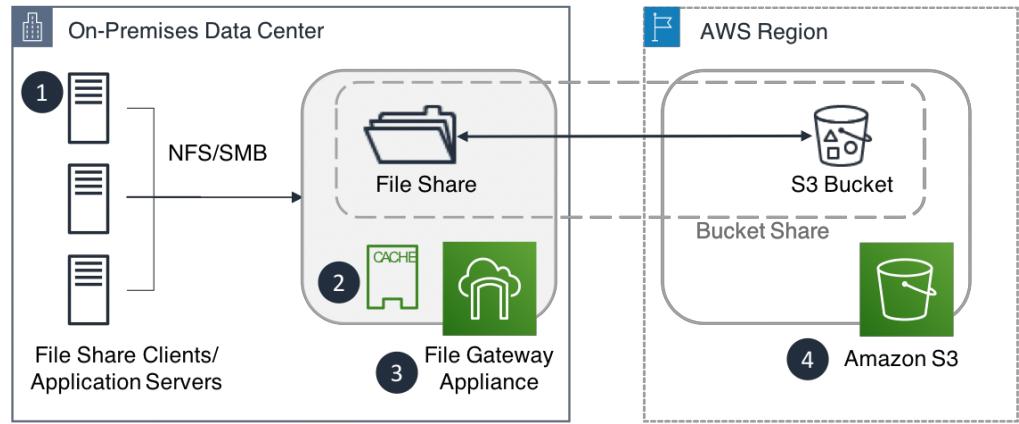
# AWS Storage Gateway

- Bridge between on-premises data and cloud data in S3
- Use cases: disaster recovery, backup & restore, tiered storage
- 4 types of Storage Gateway:
  - File Gateway
  - Volume Gateway
  - Tape Gateway
  - Amazon FSx File Gateway
- Exam Tip: You need to know the differences between all 4!

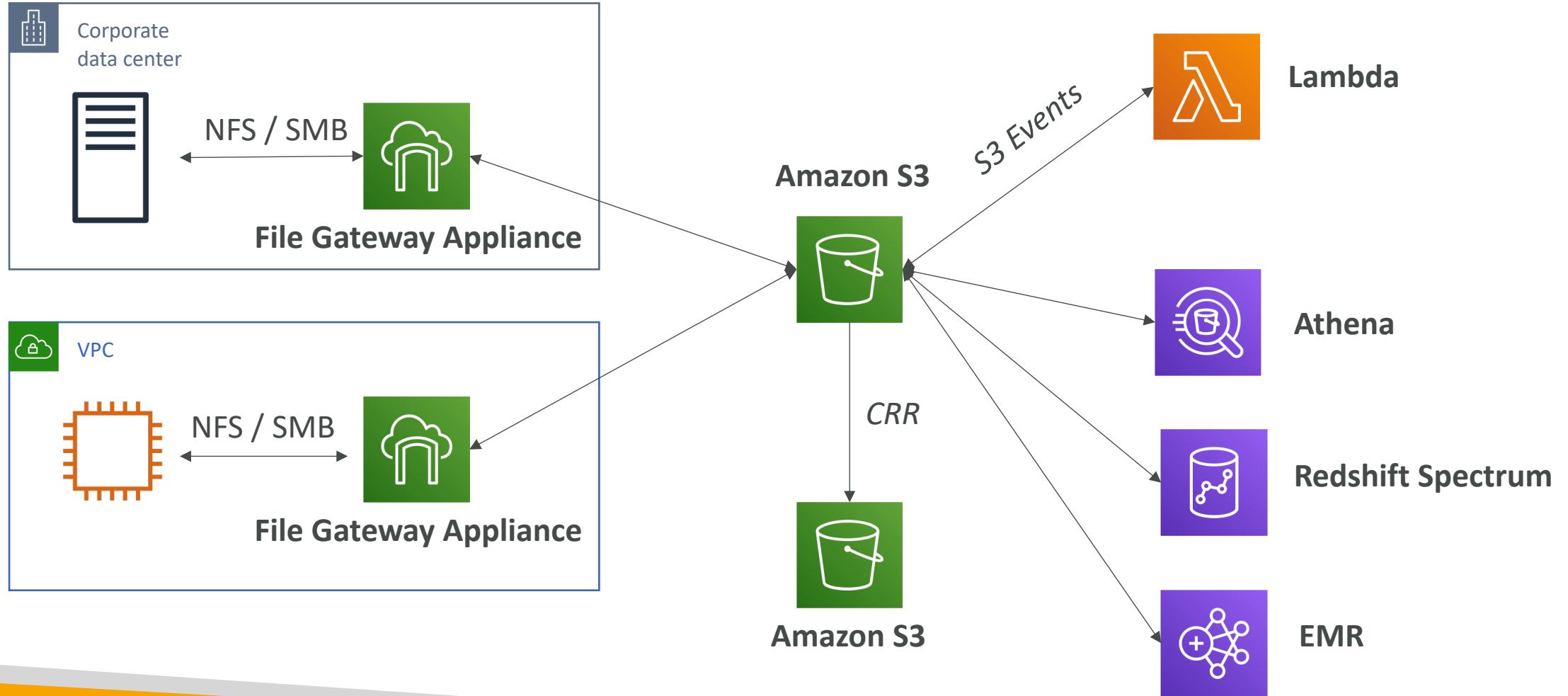


# File Gateway

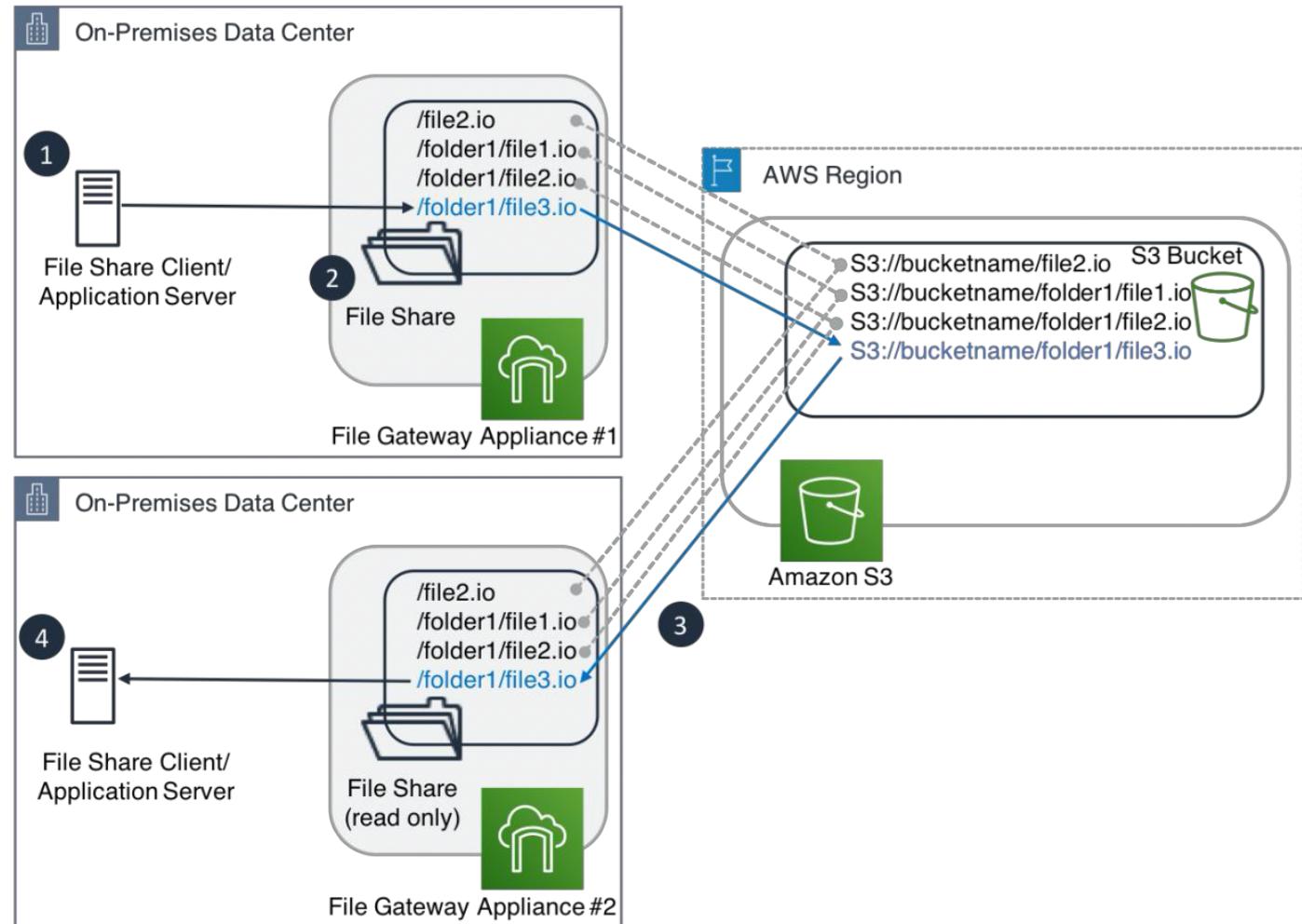
- File Gateway appliance is a virtual machine to bridge between your NFS and S3
- Metadata and directory structure are preserved
- Configured S3 buckets are accessible using the **NFS and SMB protocol**
- Each File Gateway should have an IAM role to access S3
- Most recently used data is **cached** in the file gateway
- Can be mounted on many servers
- Whitepaper:  
<https://d0.awsstatic.com/whitepapers/aws-storage-gateway-file-gateway-for-hybrid-architectures.pdf>



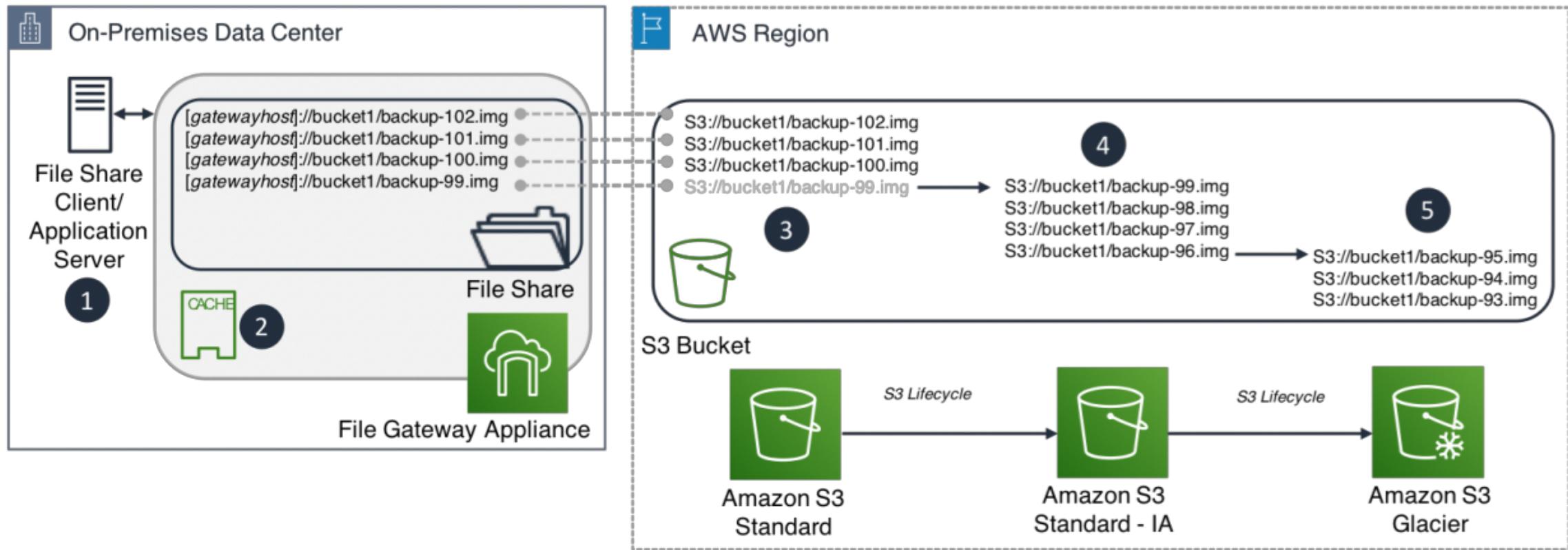
# File Gateway: Extensions



# File Gateway: Read Only Replicas



# File Gateway: Backup and Lifecycle Policies



# File Architectures: Other possibilities

- **Amazon S3 Object Versioning**

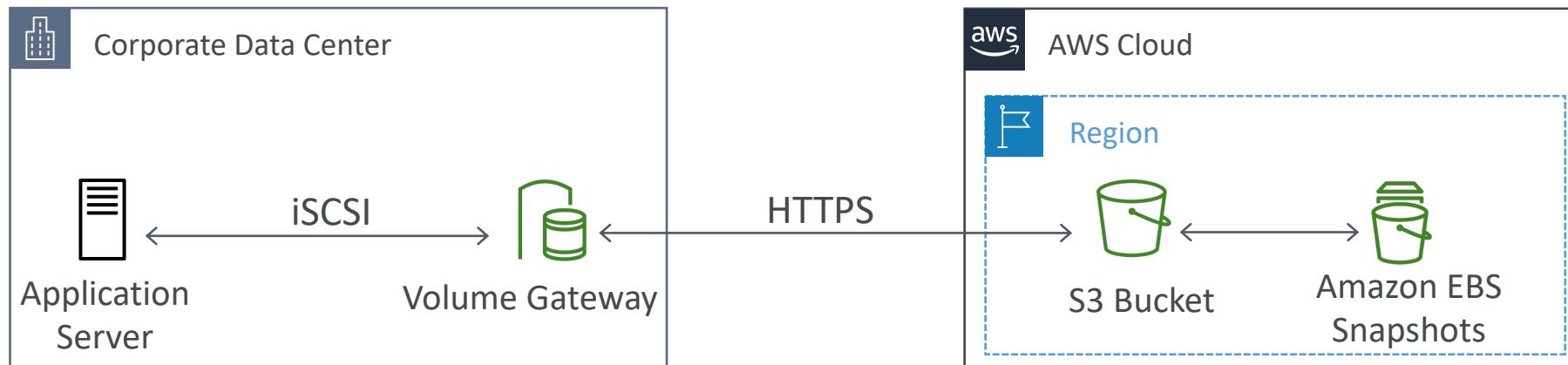
- Ability to store multiple object versions as they are modified
- Helpful to restore a file to a previous version
- Could restore an entire file system to a previous version
- Must use the “RefreshCache” API on the Gateway to be notified of restore

- **Amazon S3 Object Lock**

- Enables to have the File Gateway for Write Once Read Many (WORM) data
- If there are file modifications or renames in the file share clients, the file gateway creates a new version of the object without affecting priori versions, and the original locked version will remain unchanged

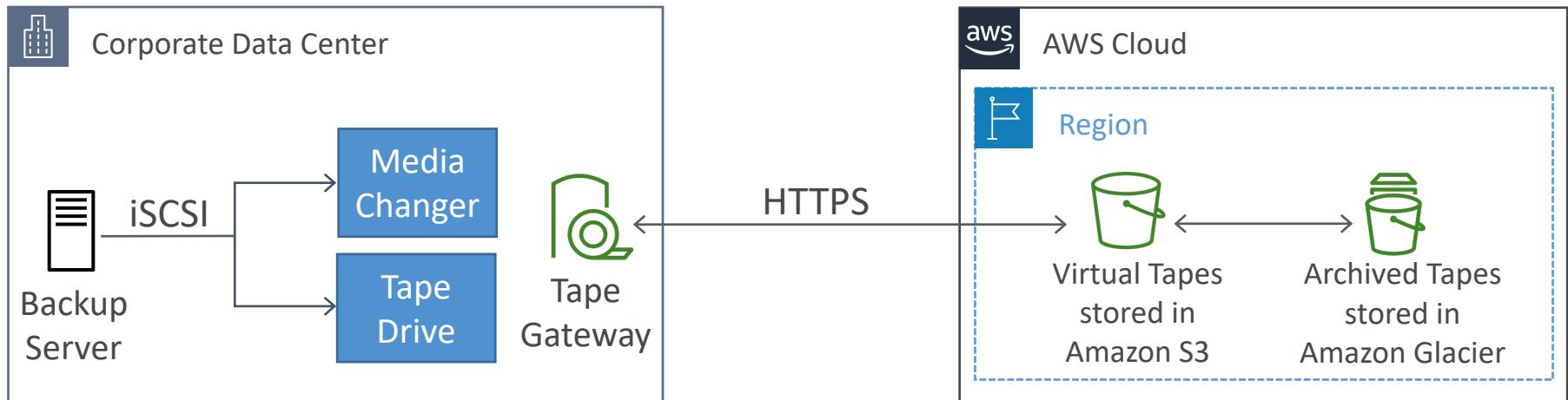
# Volume Gateway

- Block storage using **iSCSI** protocol backed by S3
- **Cached volumes:** low latency access to most recent data, full data on S3
- **Stored volumes:** entire dataset is on premise, scheduled backups to S3
- Can create EBS snapshots from the volumes and restore as EBS!
- Up to 32 volumes per gateway
  - Each volume up to 32TB in cached mode (1PB per Gateway)
  - Each volume up to 16 TB in stored mode (512TB per Gateway)



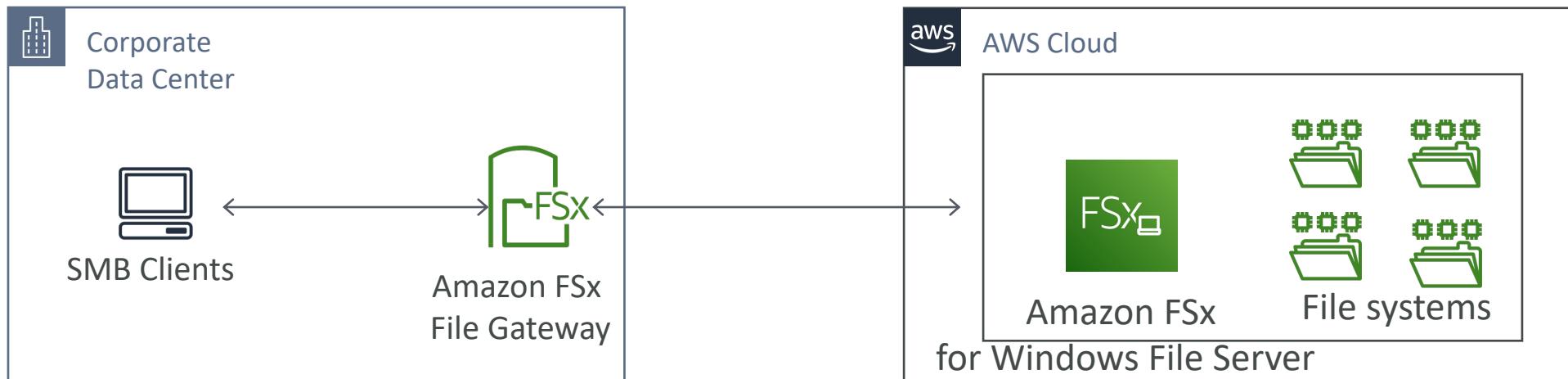
# Tape Gateway

- Some companies have backup processes using physical tapes (!)
- With Tape Gateway, companies use the same processes but in the cloud
- Virtual Tape Library (VTL) backed by Amazon S3 and Glacier
- Back up data using existing tape-based processes (and iSCSI interface)
- Works with leading backup software vendors
- You can't access single file within tapes. You need to restore the tape entirely



# Amazon FSx File Gateway

- Native access to Amazon FSx for Windows File Server
- Local cache for frequently accessed data
- Windows native compatibility (SMB, NTFS, Active Directory...)
- Useful for group file shares and home directories



# AWS Snow Family

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

- Data migration:



Snowcone



Snowball Edge



Snowmobile

- Edge computing:



Snowcone



Snowball Edge

# Data Migrations with AWS Snow Family

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

## Challenges:

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

AWS Snow Family: offline devices to perform data migrations

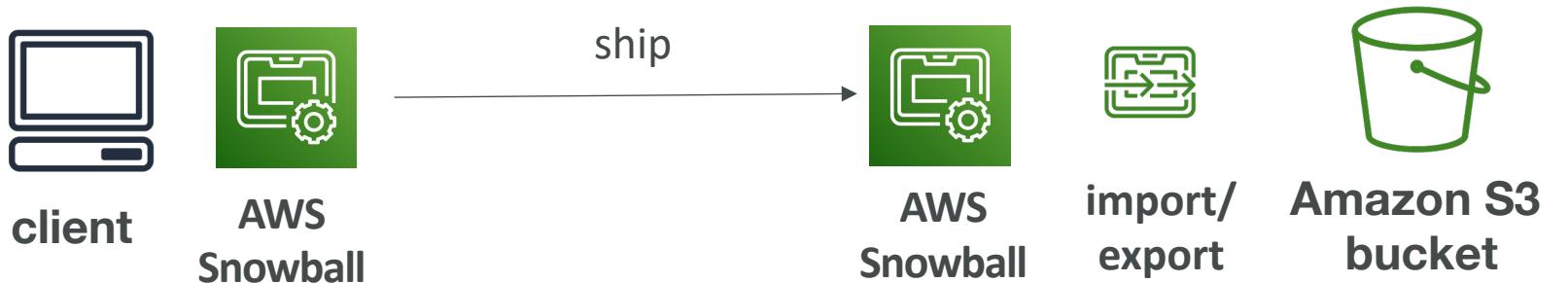
If it takes more than a week to transfer over the network, use Snowball devices!

# Diagrams

- Direct upload to S3:



- With Snow Family:



# Snowball Edge (for data transfers)



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



# AWS Snowcone



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



# AWS Snowmobile



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

# AWS Snow Family for Data Migrations



**Snowcone**



**Snowball Edge**



**Snowmobile**

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

# Snow Family – Usage Process

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

# What is Edge Computing?

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



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

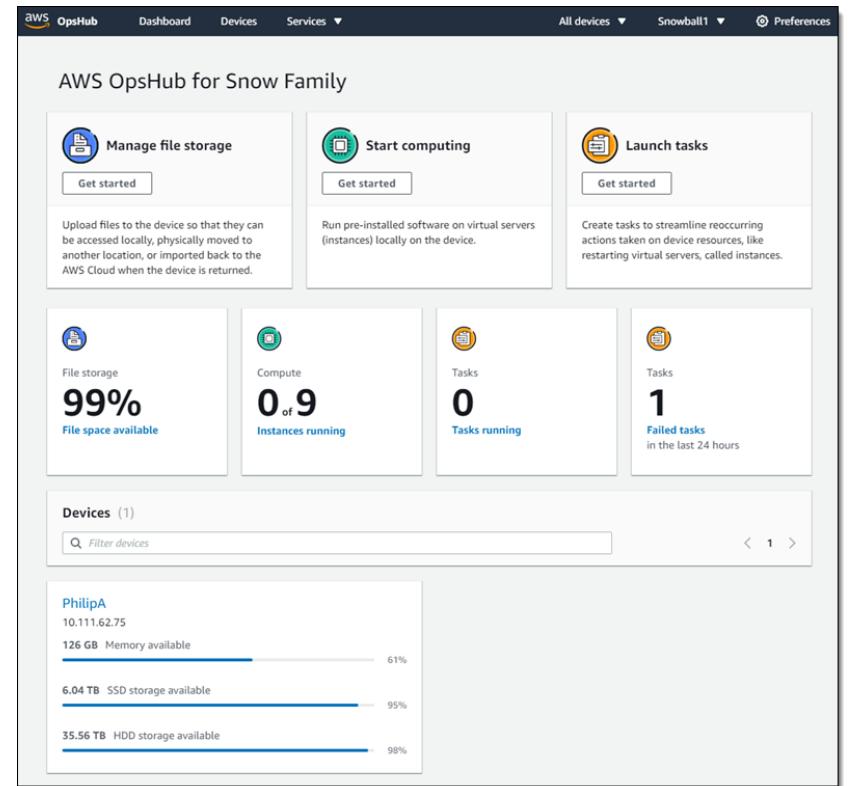
# Snow Family – Edge Computing

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



# AWS OpsHub

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



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

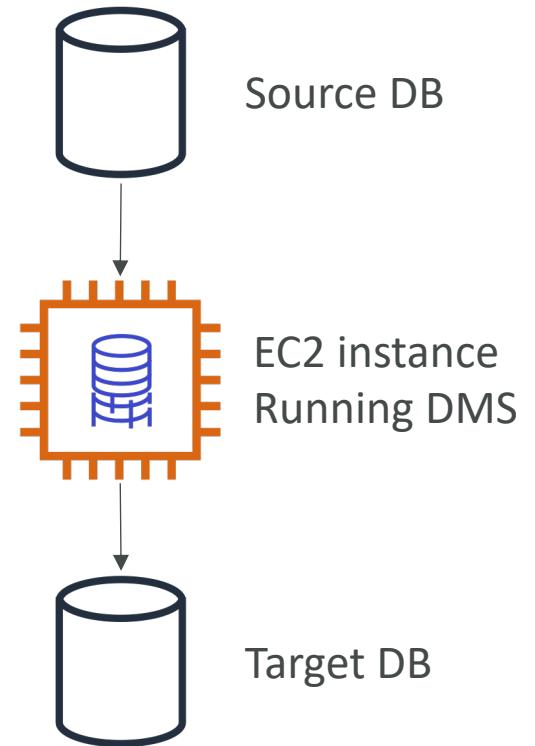
# Snow Family – Improving Transfer Performance

- Most impactful to least:
  - Perform multiple write operations at one time - from multiple terminals
  - Transfer small files in batches – zip up small files until at least 1 MB
  - Don't perform other operations on files during transfer
  - Reduce local network use
  - Eliminate unnecessary hops – directly connect to the computer
- The data transfer rate using the file interface is typically between 25 MB/s and 40 MB/s. If you need to transfer data faster than this, use the **Amazon S3 Adapter for Snowball**, which has a data transfer rate typically between 250 MB/s and 400 MB/s



# DMS – Database Migration Service

- Quickly and securely migrate databases to AWS, resilient, self healing
- The source database remains available during the migration
- Supports:
  - Homogeneous migrations: ex Oracle to Oracle
  - Heterogeneous migrations: ex Microsoft SQL Server to Aurora
- Continuous Data Replication using CDC
- You must create an EC2 instance to perform the replication tasks



# DMS Sources and Targets

## SOURCES:

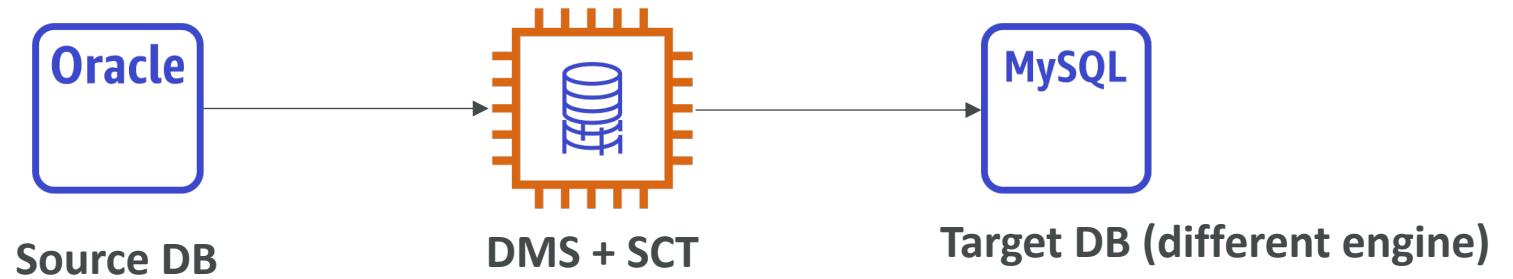
- On-premises and EC2 instances databases: Oracle, MS SQL Server, MySQL, MariaDB, PostgreSQL, MongoDB, SAP, DB2
- Azure: Azure SQL Database
- Amazon RDS: all including Aurora
- Amazon S3

## TARGETS:

- On-premises and EC2 instances databases: Oracle, MS SQL Server, MySQL, MariaDB, PostgreSQL, SAP
- Amazon RDS including Aurora
- Amazon Redshift
- Amazon DynamoDB
- Amazon S3
- ElasticSearch Service
- Kinesis Data Streams
- DocumentDB

# AWS Schema Conversion Tool (SCT)

- Convert your Database's Schema from one engine to another
- Example OLTP: (SQL Server or Oracle) to MySQL, PostgreSQL, Aurora
- Example OLAP: (Teradata or Oracle) to Amazon Redshift



- You do not need to use SCT if you are migrating the same DB engine
  - Ex: on-premises PostgreSQL => RDS PostgreSQL
  - The DB engine is still PostgreSQL (RDS is the platform)

# DMS – Good things to know

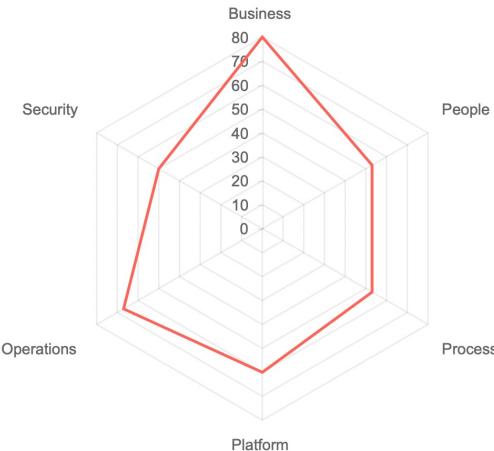
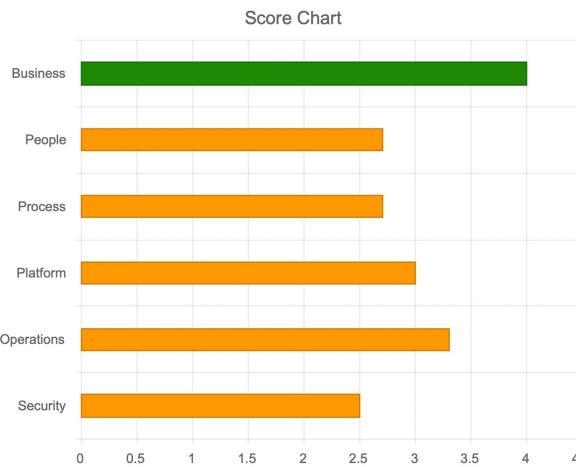
- Works over VPC Peering, VPN (site to site, software), Direct Connect
- Supports Full Load, Full Load + CDC, or CDC only
- **Oracle:**
  - Source: Supports TDE for the source using “BinaryReader”
  - Target: Supports BLOBs in tables that have a primary key, and TDE
- **ElasticSearch:**
  - Source: does not exist
  - Target: possible to migrate to DMS from a relational database
  - Therefore DMS cannot be used to replicate ElasticSearch data

# Snowball + Database Migration Service (DMS)

- Larger data migrations can include many terabytes of information.
- Can be limited due to network bandwidth or size of data
- AWS DMS can use Snowball Edge & Amazon S3 to speed up migration
- **Following stages:**
  1. You use the AWS Schema Conversion Tool (AWS SCT) to extract the data locally and move it to an Edge device.
  2. You ship the Edge device or devices back to AWS.
  3. After AWS receives your shipment, the Edge device automatically loads its data into an Amazon S3 bucket.
  4. AWS DMS takes the files and migrates the data to the target data store. If you are using change data capture (CDC), those updates are written to the Amazon S3 bucket and then applied to the target data store.

# AWS Cloud Adoption Readiness Tool (CART)

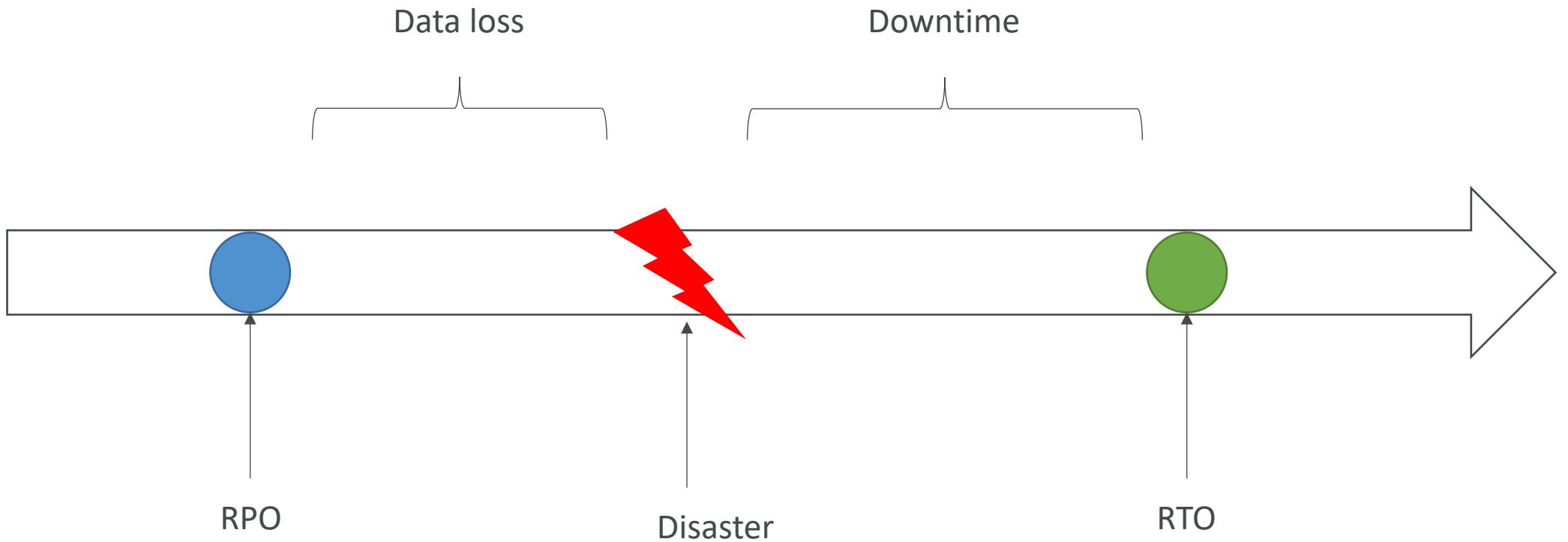
- Helps organizations develop efficient and effective plans for cloud adoption and migrations
- Transforms your idea of moving to the cloud into a detailed plan that follows AWS best practices
- Answer a set of questions across six perspectives (business, people, process, platform, operations, security)
- Generates a custom report on your level of migration readiness



# Disaster Recovery Overview

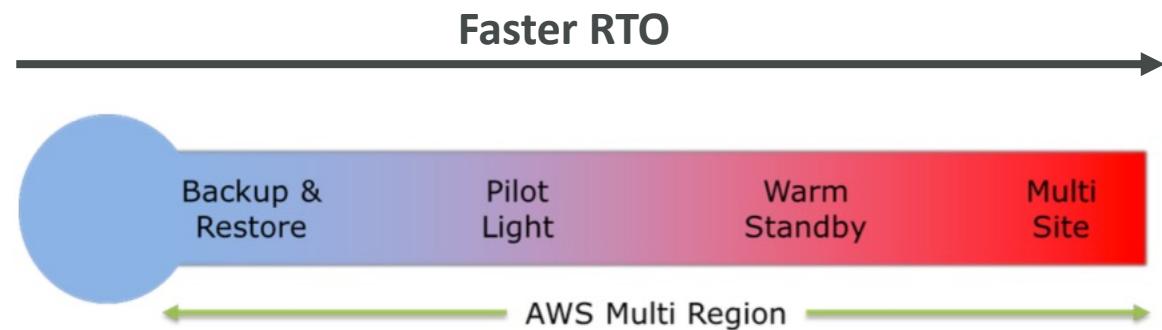
- Any event that has a negative impact on a company's business continuity or finances is a disaster
- Disaster recovery (DR) is about preparing for and recovering from a disaster
- What kind of disaster recovery?
  - on-premises => on-premises: traditional DR, and very expensive
  - on-premises => AWS Cloud: hybrid recovery
  - AWS Cloud Region A => AWS Cloud Region B
- Need to define two terms:
  - RPO: Recovery Point Objective
  - RTO: Recovery Time Objective

# RPO and RTO

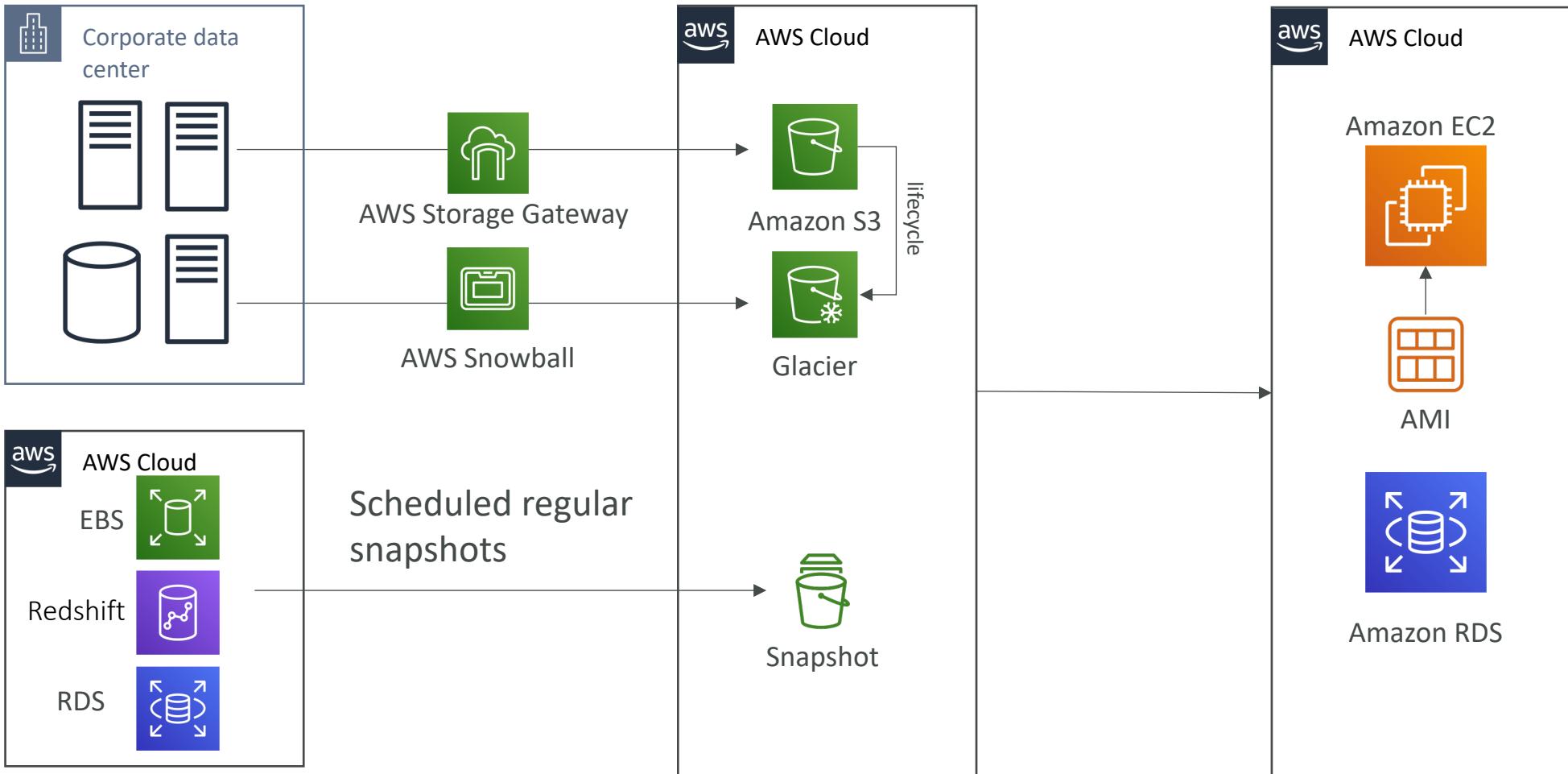


# Disaster Recovery Strategies

- Backup and Restore
- Pilot Light
- Warm Standby
- Hot Site / Multi Site Approach

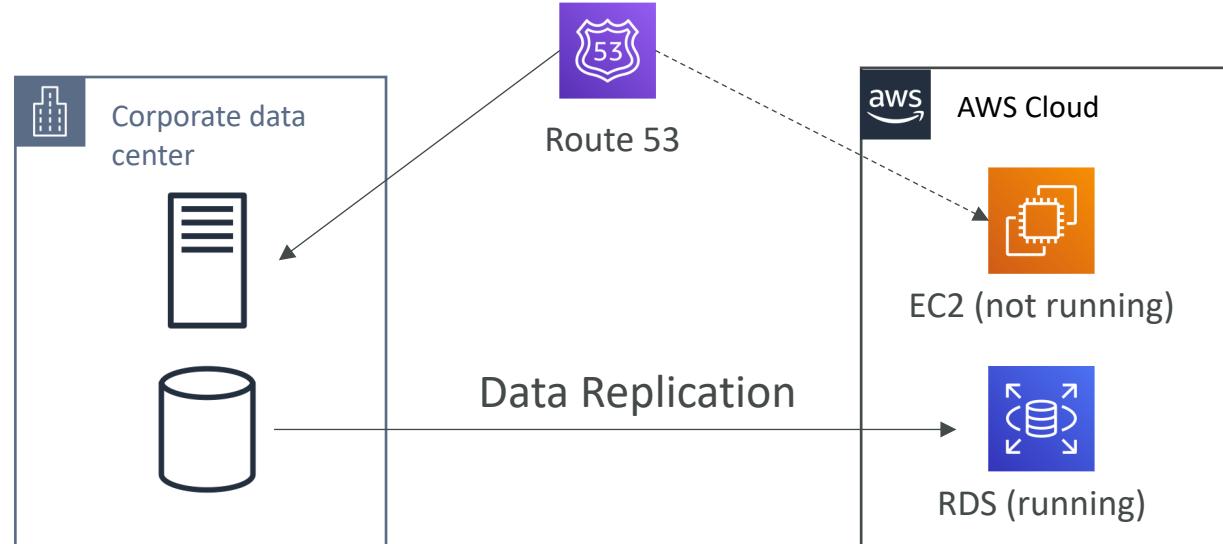


# Backup and Restore (High RPO)



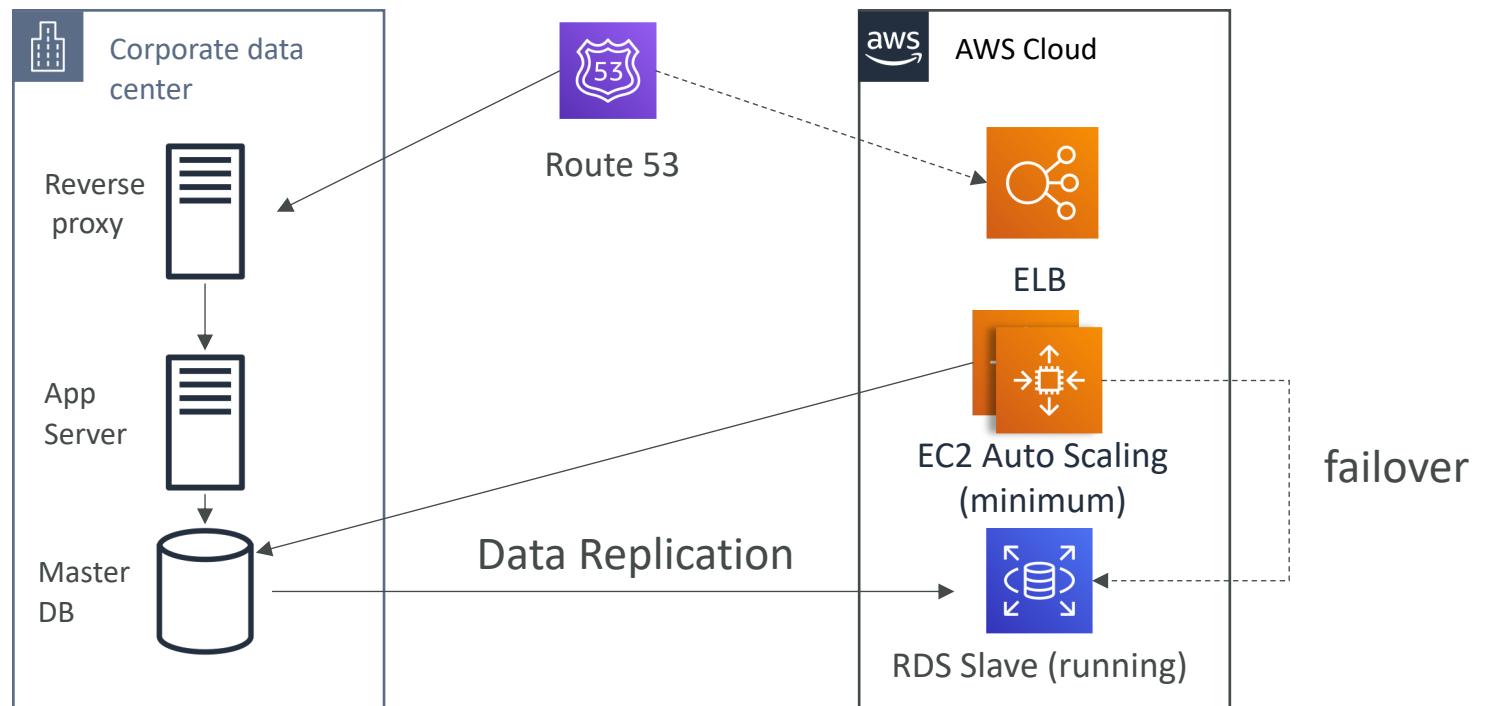
# Disaster Recovery – Pilot Light

- A small version of the app is always running in the cloud
- Useful for the critical core (pilot light)
- Very similar to Backup and Restore
- Faster than Backup and Restore as critical systems are already up



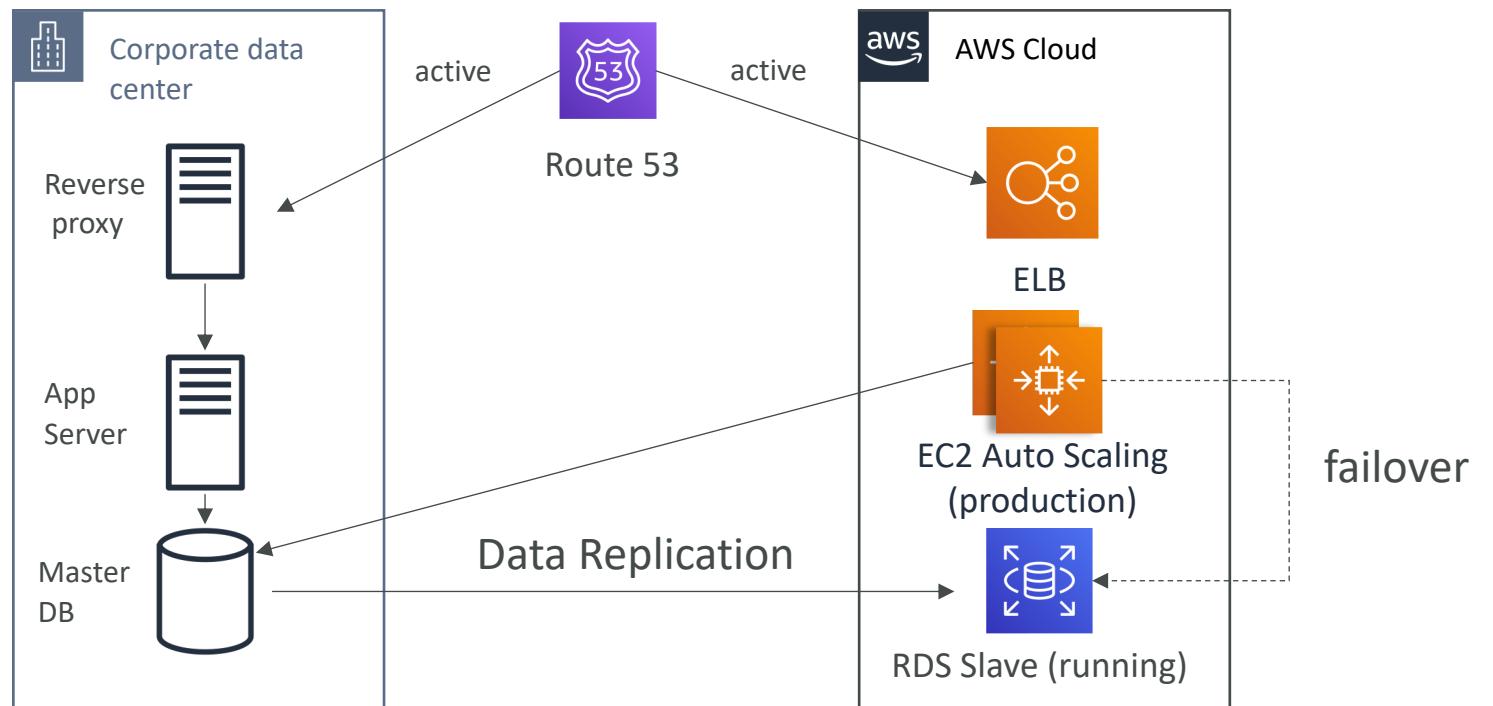
# Warm Standby

- Full system is up and running, but at minimum size
- Upon disaster, we can scale to production load

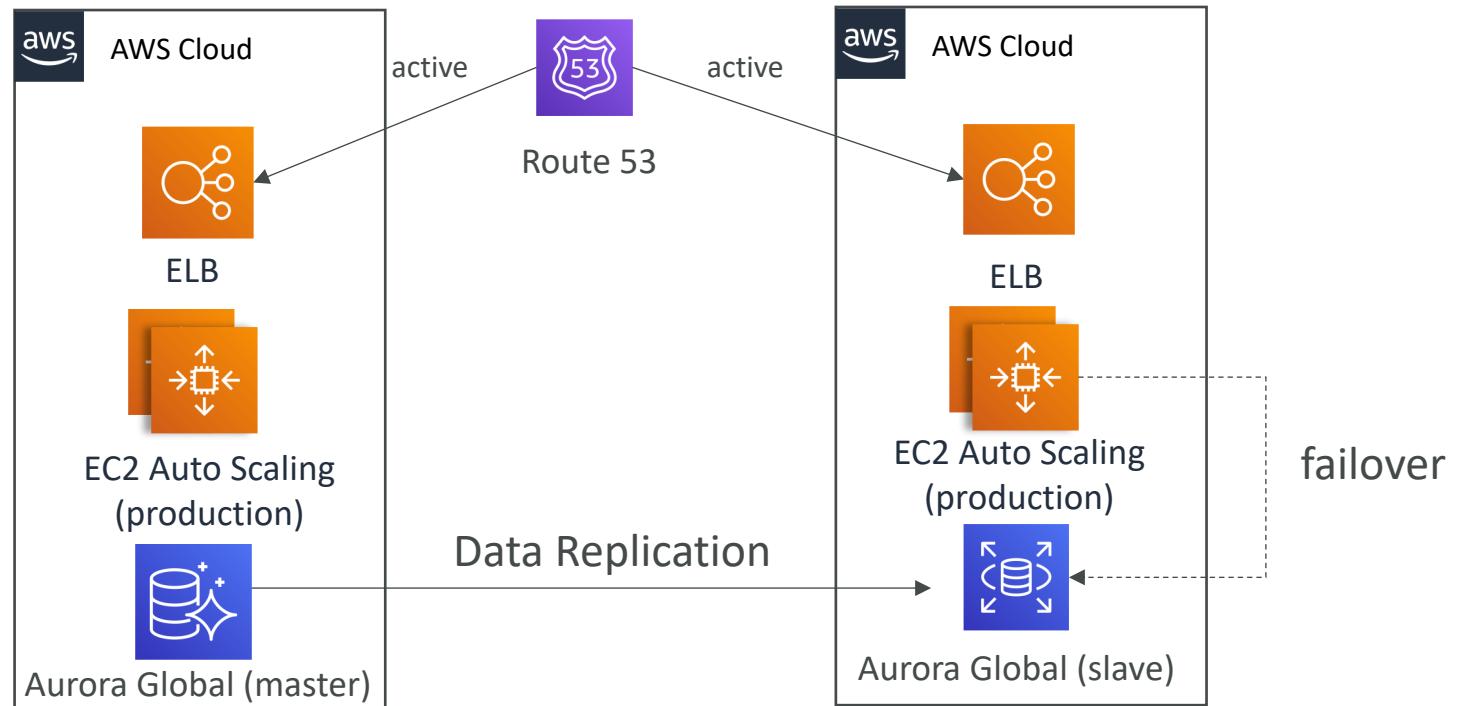


# Multi Site / Hot Site Approach

- Very low RTO (minutes or seconds) – very expensive
- Full Production Scale is running AWS and On Premise



# All AWS Multi Region



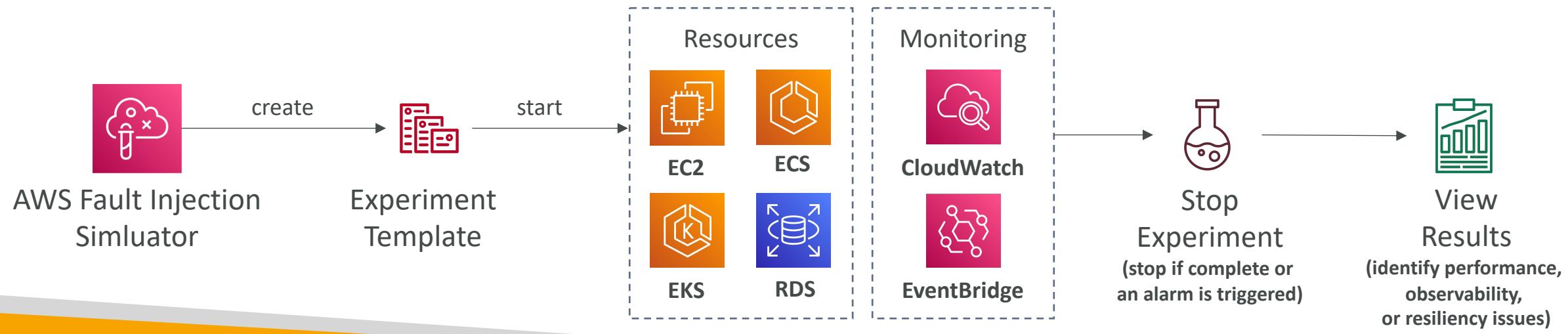
# Disaster Recovery Tips

- Backup
  - EBS Snapshots, RDS automated backups / Snapshots, etc...
  - Regular pushes to S3 / S3 IA / Glacier, Lifecycle Policy, Cross Region Replication
  - From on-premises: Snowball or Storage Gateway
- High Availability
  - Use Route53 to migrate DNS over from Region to Region
  - RDS Multi-AZ, ElastiCache Multi-AZ, EFS, S3
  - Site to Site VPN as a recovery from Direct Connect
- Replication
  - RDS Replication (Cross Region), AWS Aurora + Global Database
  - Database replication from on-premises to RDS
  - Storage Gateway
- Automation
  - CloudFormation / Elastic Beanstalk to re-create a whole new environment
  - Recover / Reboot EC2 instances with CloudWatch if alarms fail
  - AWS Lambda functions for customized automations
- Chaos
  - Netflix has a “simian-army” randomly terminating EC2

# AWS Fault Injection Simulator (FIS)



- A fully managed service for running fault injection experiments on AWS workloads
- Based on **Chaos Engineering** – stressing an application by creating disruptive events (e.g., sudden increase in CPU or memory), observing how the system responds, and implementing improvements
- Helps you uncover hidden bugs and performance bottlenecks
- Supports the following AWS services: EC2, ECS, EKS, RDS...
- Use pre-built templates that generate the desired disruptions



# AWS Application Discovery Service

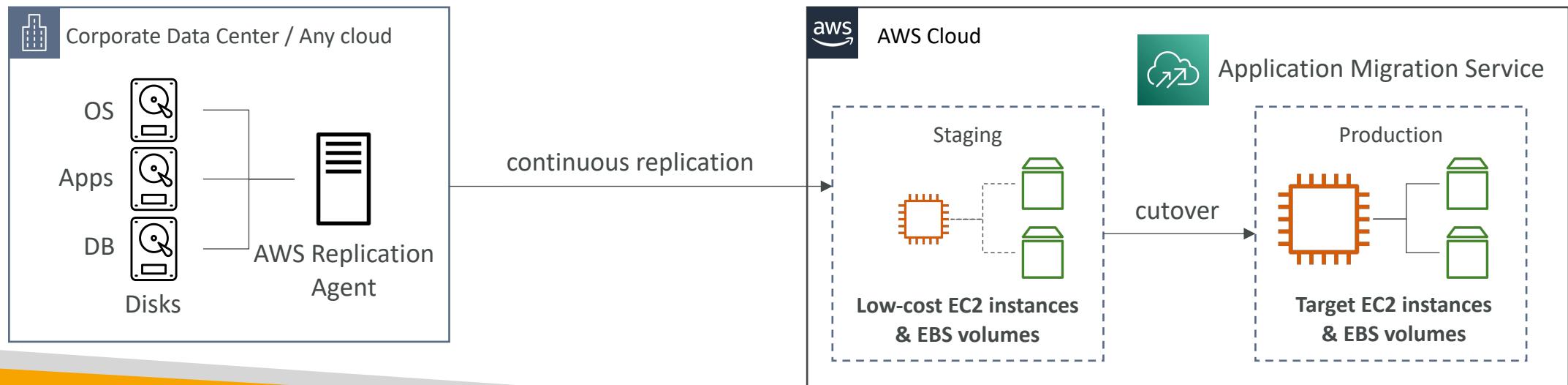


- Plan migration projects by gathering information about on-premises data centers
- Server utilization data and dependency mapping are important for migrations
- **Agentless Discovery (AWS Agentless Discovery Connector)**
  - Open Virtual Appliance (OVA) package that can be deployed to a VMware host
  - VM inventory, configuration, and performance history such as CPU, memory, and disk usage
  - OS agnostic
- **Agent-based Discovery (AWS Application Discovery Agent)**
  - System configuration, system performance, running processes, and details of the network connections between systems
  - Supports Microsoft Server, Amazon Linux, Ubuntu, RedHat, CentOS, SUSE...
- Resulting data can be exported as CSV or viewed within AWS Migration Hub
- Data can be explored using pre-defined queries in Amazon Athena

# AWS Application Migration Service (MGN)



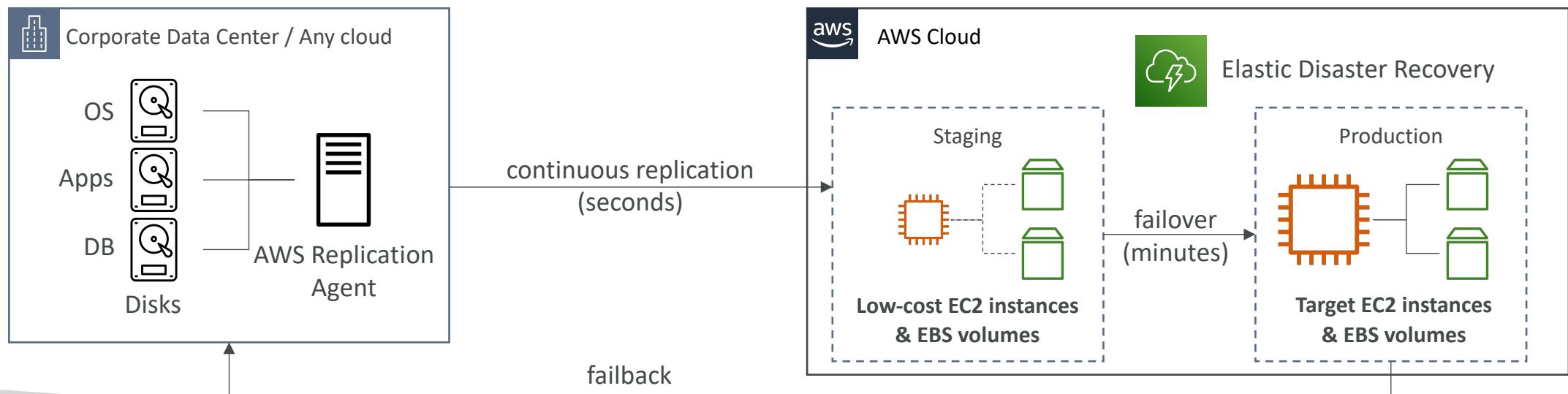
- The “AWS evolution” of CloudEndure Migration, replacing AWS Server Migration Service (SMS)
- Lift-and-shift (rehost) solution which simplify **migrating** applications to AWS
- Converts your physical, virtual, and cloud-based servers to run natively on AWS
- Supports wide range of platforms, Operating Systems, and databases
- Minimal downtime, reduced costs



# AWS Elastic Disaster Recovery (DRS)



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



# On-premises strategy with AWS

- Ability to download Amazon Linux 2 AMI as a VM (.iso format)
  - VMWare, KVM, VirtualBox (Oracle VM), Microsoft Hyper-V
- **AWS Application Discovery Service**
  - Gather information about your on-premises servers to plan a migration
  - Server utilization and dependency mappings
  - Track with AWS Migration Hub
- **AWS Application Migration Service (MGN)**
  - Replacing AWS Server Migration Services & CloudEndure Migration
  - Incremental replication of on-premises live servers to AWS
  - Migrates the entire VM into AWS
- **AWS Elastic Disaster Recovery (DRS)**
  - Replacing CloudEndure Disaster Recovery
  - Recover on-premises workloads onto AWS
- **AWS Database Migration Service (DMS)**
  - replicate on-premises => AWS , AWS => AWS, AWS => on-premises
  - Works with various database technologies (Oracle, MySQL, DynamoDB, etc..)



AWS Application  
Discovery Service



AWS Application  
Migration Service



AWS Elastic  
Disaster Recovery



AWS Database  
Migration Service

# VPC Section

# VPC Basics

- CIDR: Block of IP address
  - Example: 192.168.0.0/26: 192.168.0.0 – 192.168.0.63 (64 IP)
  - Used for security groups, route tables, VPC, subnets, etc...
- Private IP
  - 10.0.0.0 – 10.255.255.255 (10.0.0.0/8) <= in big networks
  - 172.16.0.0 – 172.31.255.255 (172.16.0.0/12) <= default AWS one
  - 192.168.0.0 – 192.168.255.255 (192.168.0.0/16) <= example: home networks
- Public IP
  - All the rest

# VPC Basics

- **VPC**
  - A VPC must have a defined list of CIDR blocks, that cannot be changed
  - Each CIDR within VPC: min size is /28, max size is /16 (65536 IP addresses)
  - VPC is private, so only Private IP CIDR ranges are allowed
- **Subnets**
  - Within a VPC, defined as a CIDR that is a subset of the VPC CIDR
  - All instances within subnets get a private IP
  - First 4 IP and last one in every subnet is reserved by AWS
- **Route Tables**
  - Used to control where the network traffic is directed to
  - Can be associated with specific subnets
  - The “most specific” routing rule is always followed (192.168.0.1/24 beats 0.0.0.0/0)

# VPC Basics

- **Internet Gateway (IGW)**
  - Helps our VPC connect to the internet, HA, scales horizontally
  - Acts as a NAT for instances that have a public IPv4 or public IPv6
- **Public Subnets**
  - Has a route table that sends 0.0.0.0/0 to an IGW
  - Instances must have a public IPv4 to talk to the internet
- **Private Subnets**
  - Access internet with a NAT Instance or NAT Gateway setup in a public subnet
  - Must edit routes so that 0.0.0.0/0 routes traffic to the NAT

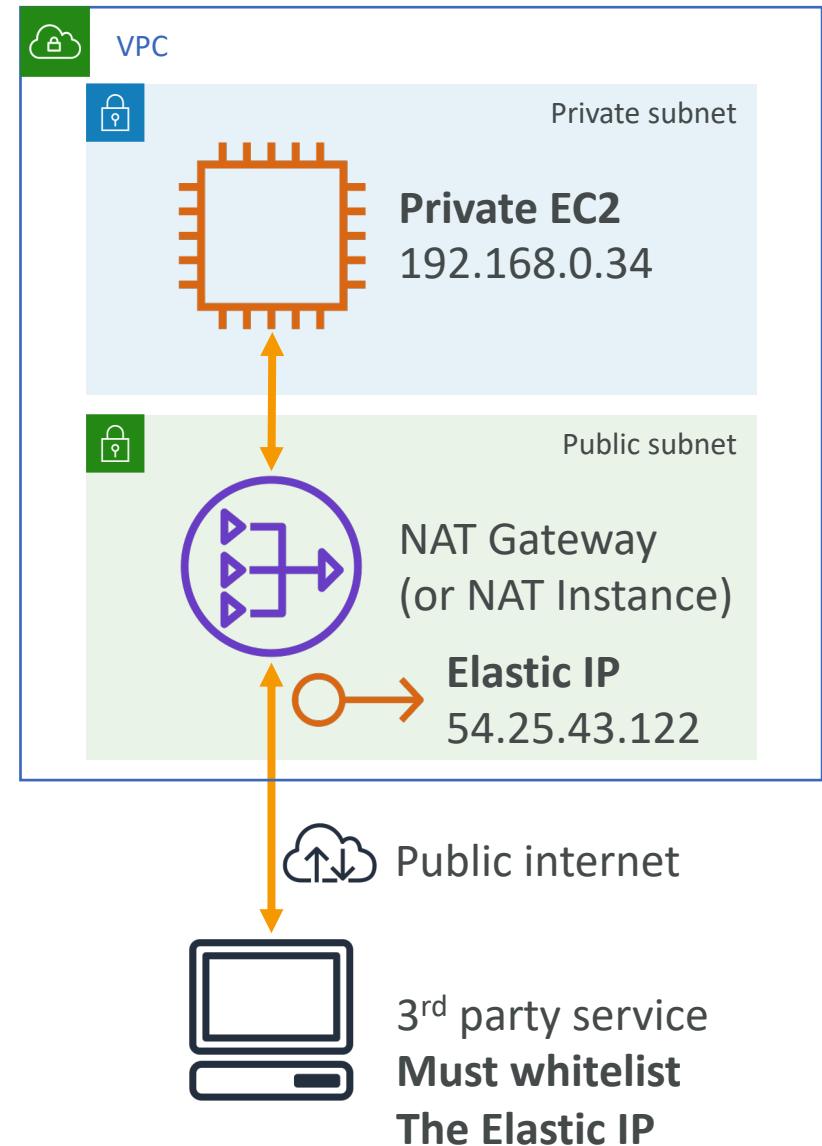
# VPC Basics

- **NAT Instance**

- EC2 instance you deploy in a public subnet
- Edit the route in your private subnet to route 0.0.0.0/0 to your NAT instance
- Not resilient to failure, limited bandwidth based on instance type, cheap
- Must manage failover yourself

- **NAT Gateway**

- Managed NAT solution, bandwidth scales automatically
- Resilient to failure within a single AZ
- Must deploy multiple NAT Gateways in multiple AZ for HA
- Has an Elastic IP, external services see the IP of the NAT Gateway as the source



# VPC Basics

- **Network ACL (NACL)**

- Stateless firewall defined at the subnet level, applies to all instances within
- Support for allow and deny rules
- Stateless = return traffic must be explicitly allowed by rules
- Helpful to quickly and cheaply block specific IP addresses

- **Security Groups**

- Applied at the instance level, only support for allow rules, no deny rules
- Stateful = return traffic is automatically allowed, regardless of rules
- Can reference other security groups in the same region (peered VPC, cross-account)

# VPC Basics

- **VPC Flows Logs**

- Log internet traffic going through your VPC
- Can be defined at the VPC level, Subnet level, or ENI-level
- Helpful to capture “denied internet traffic”
- Can be sent to CloudWatch Logs and Amazon S3

- **Bastion Hosts**

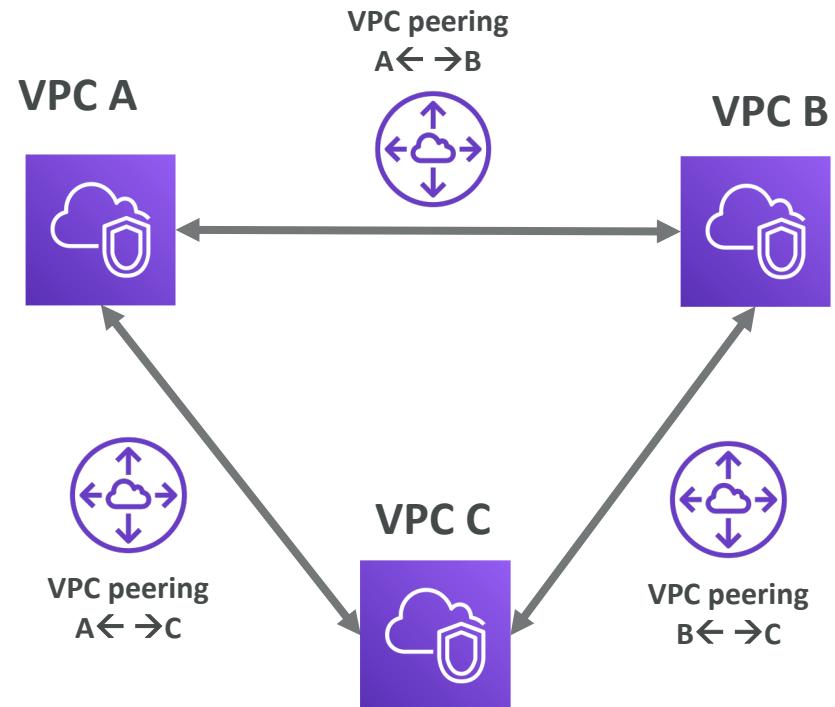
- SSH into private EC2 instances through a public EC2 instance (bastion host)
- You must manage these instances yourself (failover, recovery)
- SSM Session Manager is a more secure way to remote control without SSH

# VPC Basics

- IPv6 in short
  - All IPv6 addresses are public, total  $3.4 \times 10^{38}$  addresses (vs 4.3 billion IPv4)
  - Example CIDR: 2600:1f18:80c:a900::/56
  - Addresses are “random” and can’t be scanned online (because too many)
- VPC support for IPv6
  - Create an IPv6 CIDR for VPC & use an IGW (supports IPv6)
  - Public subnet:
    - Create an instance with IPv6 support
    - Create a route table entry to ::/0 (IPv6 “all”) to the IGW
  - Private subnet (instances cannot be reached by IPv6 but can reach IPv6):
    - Create an **Egress-Only Internet Gateway** in the public subnet
    - Add a route table entry for the private subnet from ::/0 to the Egress-Only IGW

# VPC Peering

- Connect two VPC, privately using AWS' network
- Make them behave as if they were in the same network
- Must not have overlapping CIDR
- VPC Peering connection is **not transitive** (must be established for each VPC that need to communicate with one another)
- You can do VPC peering with another AWS account
- You must update route tables in each VPC's subnets to ensure instances can communicate



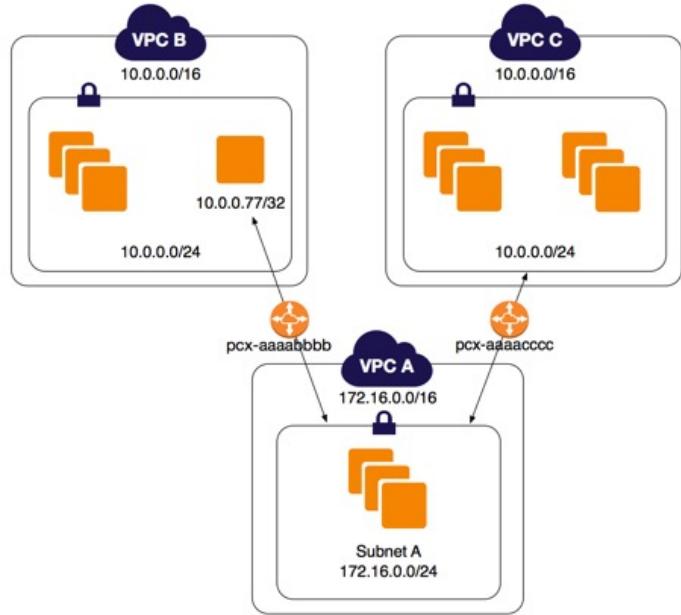
# VPC Peering – Good to know

- VPC peering can work **inter-region, cross-account**
- You can reference a security group of a peered VPC (works cross account)

Type	Protocol	Port Range	Source
HTTP	TCP	80	sg-00d2b0f5fd6de757e
HTTP	TCP	80	sg-013347765f7a63aae/12356788

# VPC Peering – Longest Prefix Match

- VPC uses the longest prefix match to select the most specific route



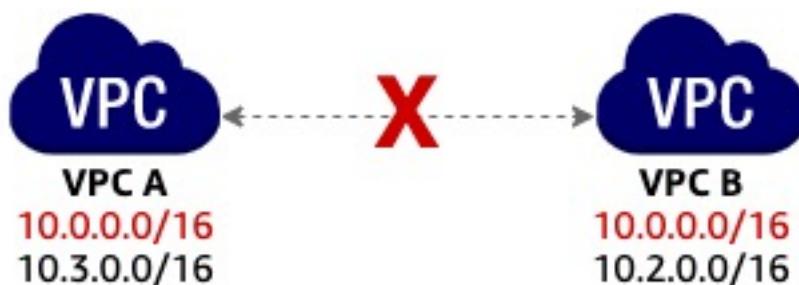
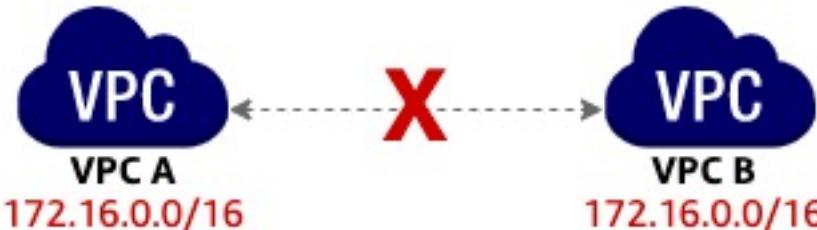
Route Table	Destination	Target
VPC A	172.16.0.0/16	Local
	10.0.0.77/32	pcx-aaaabbbb
	10.0.0.0/16	pcx-aaaacccc
VPC B	10.0.0.0/16	Local
	172.16.0.0/16	pcx-aaaabbbb
VPC C	10.0.0.0/16	Local
	172.16.0.0/16	pcx-aaaacccc

- Here the longest prefix for 10.0.0.77 is 10.0.0.77/32 (route table VPC A)
- (other way of saying it is “most specific route”)

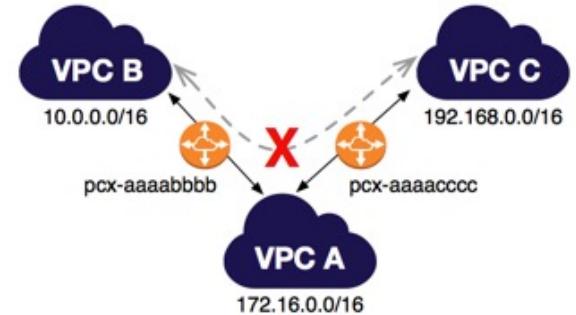
<https://docs.aws.amazon.com/vpc/latest/peering/peering-configurations-partial-access.html#one-to-two-vpcs-lpm>

# VPC Peering – Invalid Configurations

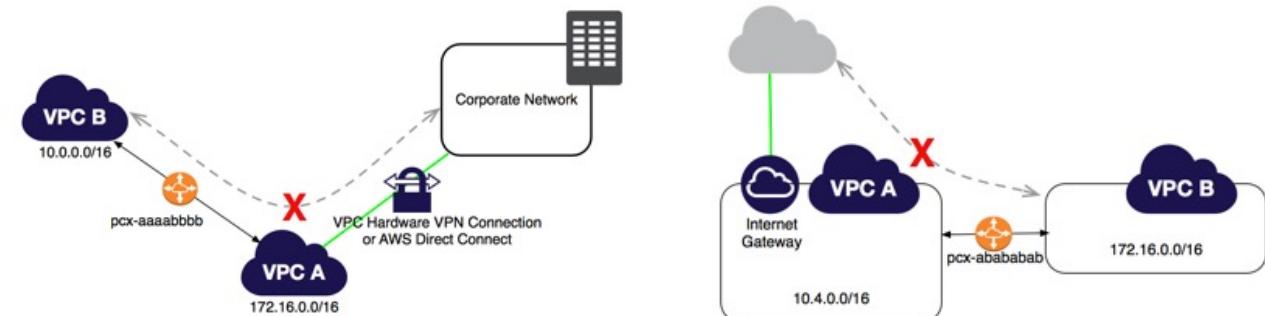
## Overlapping CIDR for IPv4



## No Transitive VPC Peering



## No Edge to Edge Routing



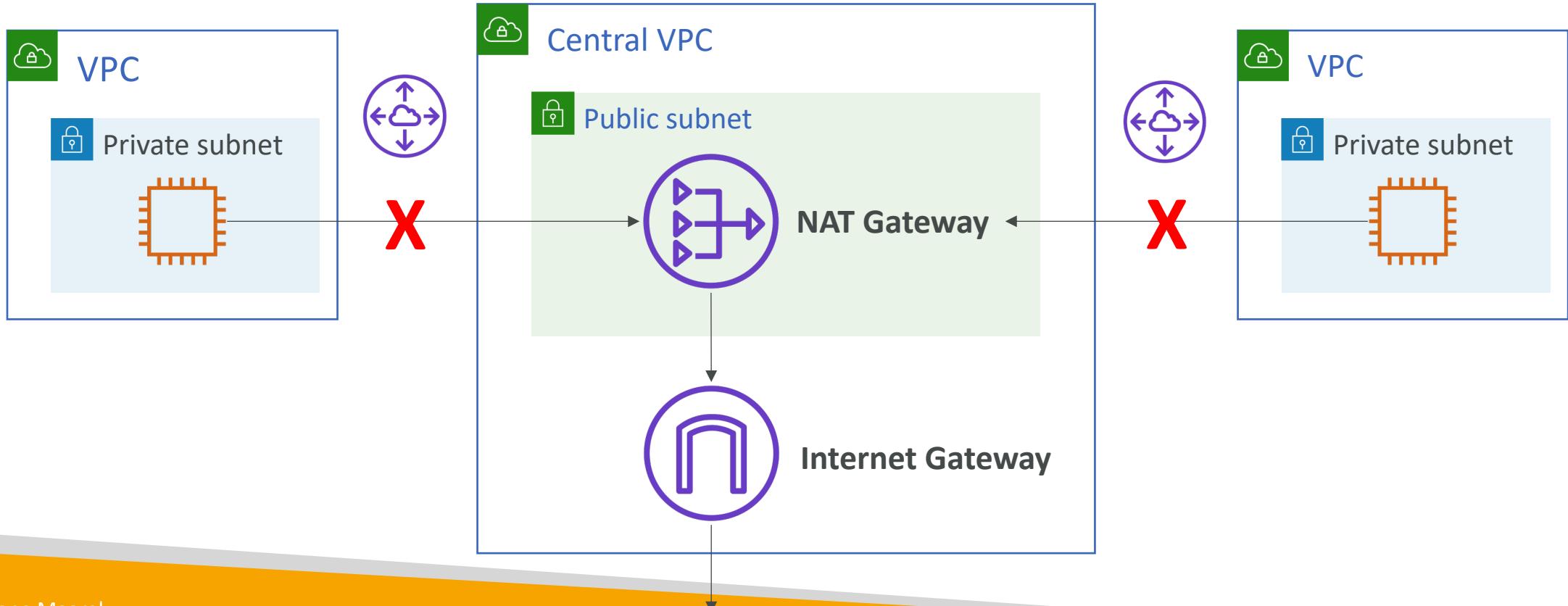
VPN, Direct Connect, IGW, NAT, Gateway VPC Endpoint (S3 & DynamoDB)

<https://docs.aws.amazon.com/vpc/latest/peering/invalid-peering-configurations.html>

# VPC Peering – Invalid Configuration

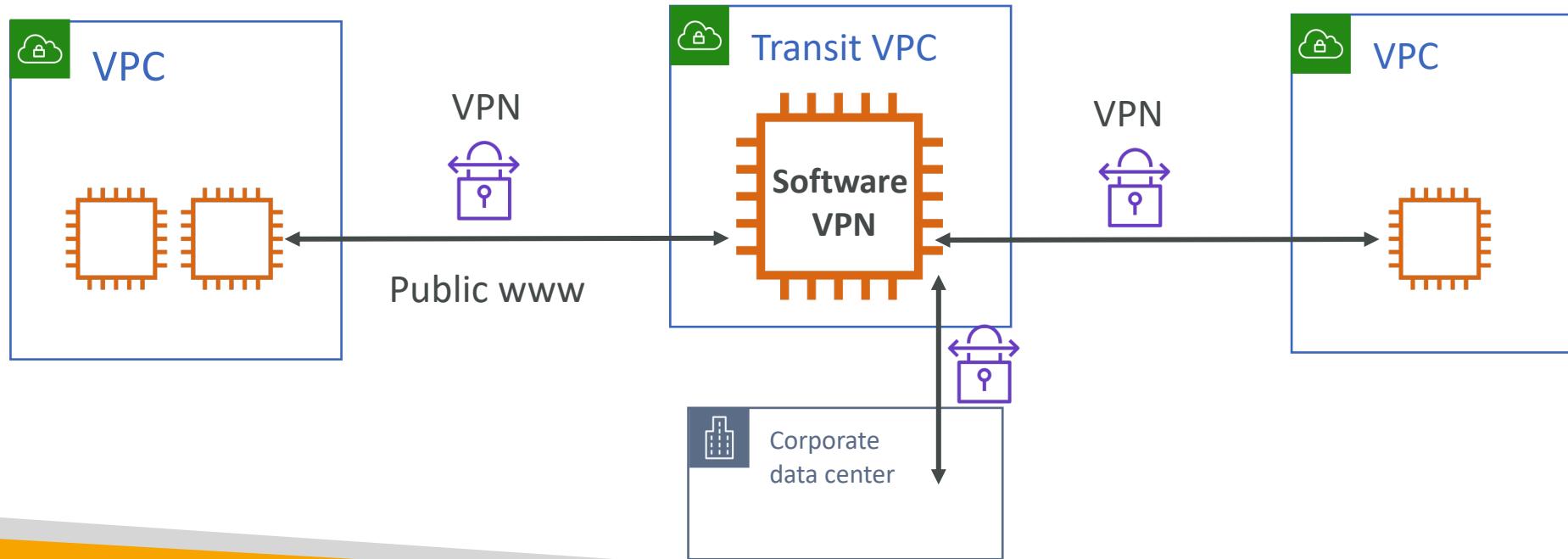
## No edge to edge routing

- This is an invalid configuration
- VPC Peering does not support edge to edge routing for NAT devices

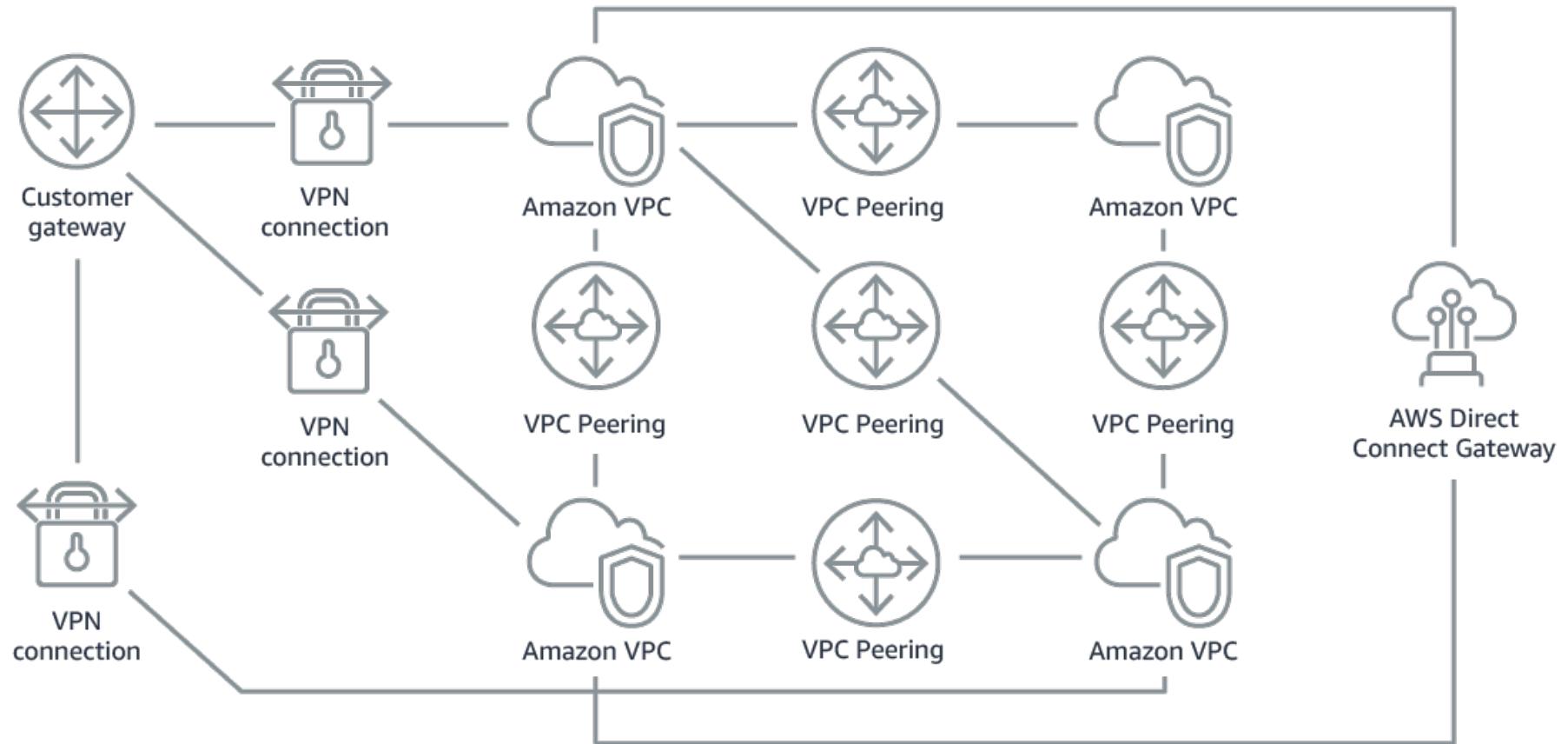


# Transit VPC (=Software VPN)

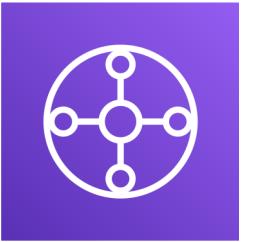
- Not an AWS offering, newer managed solution is Transit Gateway
- Uses the public internet with a software VPN solution
- Allows for transitive connectivity between VPC & locations
- More complex routing rules, overlapping CIDR ranges, network-level packet filtering



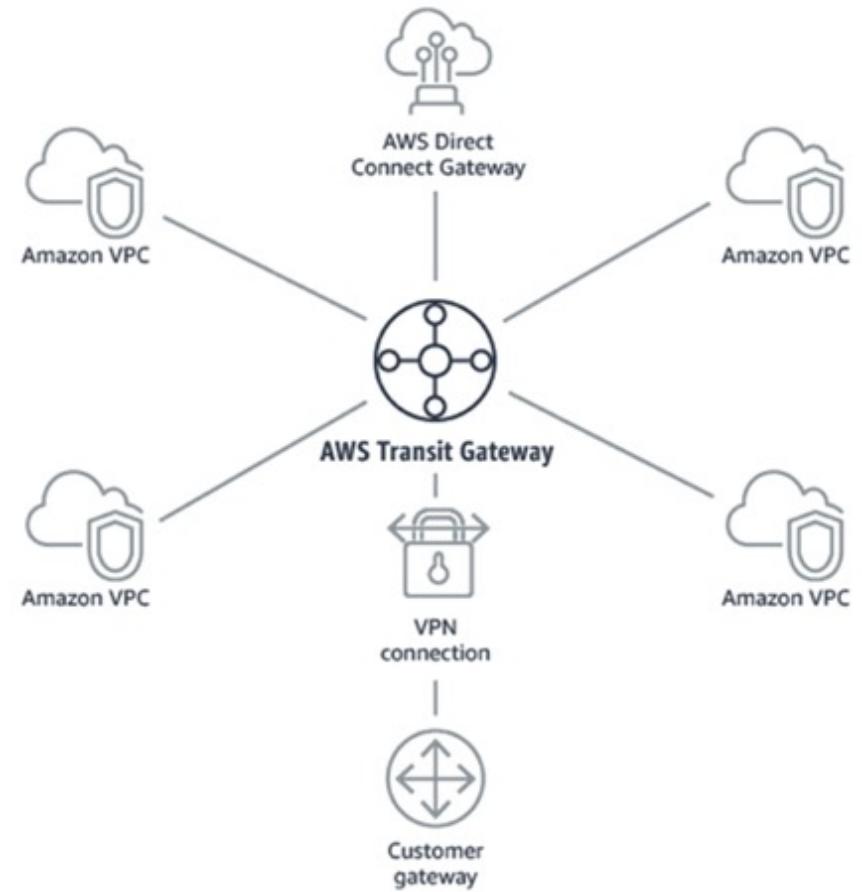
# Network topologies can become complicated



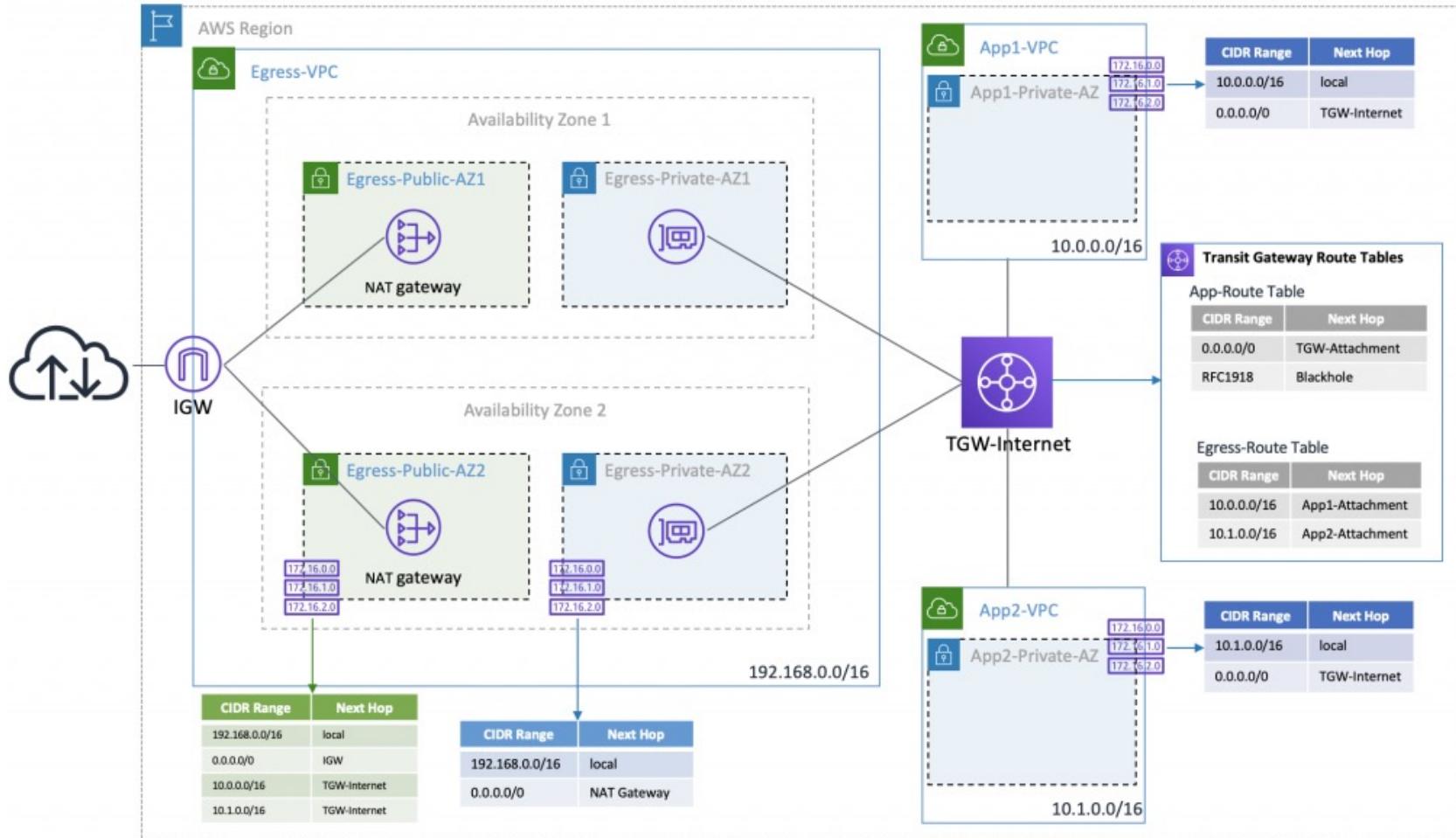
# Transit Gateway



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



# Transit Gateway – Central NAT Gateway

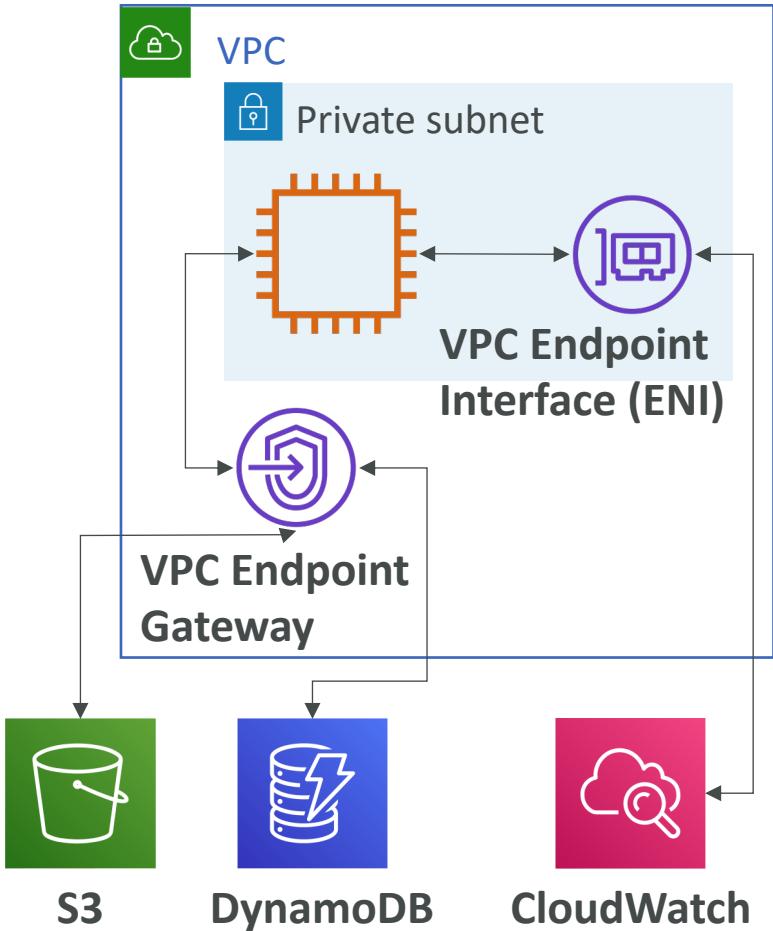


- The NAT Gateway is shared in the Egress-VPC
- The private App VPC can access internet through the TGW
- In this example: the App VPCs cannot communicate with each other based on the TGW route table

<https://aws.amazon.com/blogs/networking-and-content-delivery/creating-a-single-internet-exit-point-from-multiple-vpcs-using-aws-transit-gateway/>

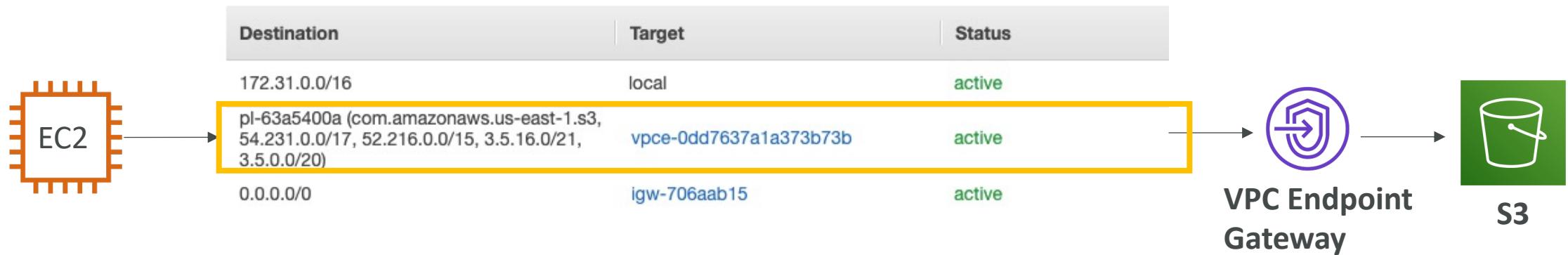
# VPC Endpoints

- Endpoints allow you to connect to AWS Services using a private network instead of the public www network
- They scale horizontally and are redundant
- No more IGW, NAT, etc... to access AWS Services
- VPC Endpoint Gateway (S3 & DynamoDB)
- VPC Endpoint Interface (all incl. S3 & DDB)
- In case of issues:
  - Check DNS Setting Resolution in your VPC
  - Check Route Tables



# VPC Endpoint Gateway

- Only works for S3 and DynamoDB, must create one gateway per VPC
- Must update route tables entries
- Gateway is defined at the VPC level



- DNS resolution must be enabled in the VPC
- The same public hostname for S3 can be used
- Gateway endpoint cannot be extended out of a VPC (VPN, DX, TGW, peering)

# VPC Endpoints Interface

- Provision an ENI that will have a private endpoint interface hostname
- Leverage Security Groups for security
- Private DNS (setting when you create the endpoint)
  - The public hostname of a service will resolve to the private Endpoint Interface hostname
  - VPC Setting: “Enable DNS hostnames” and “Enable DNS Support” must be ‘true’
  - Example for Athena:
    - vpce-0b7d2995e9dfe5418-mwrths3x.athena.us-east-1.vpce.amazonaws.com
    - vpce-0b7d2995e9dfe5418-mwrths3x-us-east-1a.athena.us-east-1.vpce.amazonaws.com
    - vpce-0b7d2995e9dfe5418-mwrths3x-us-east-1b.athena.us-east-1.vpce.amazonaws.com
    - athena.us-east-1.amazonaws.com (private DNS name)
- Interface can be accessed from Direct Connect and Site-to-Site VPN

# VPC Endpoint Policies

- Endpoint Policies are JSON documents to control access to services
- Does not override or replace IAM user policies or service-specific policies (such as S3 bucket policies)

```
{  
  "Statement": [  
    {"Action": ["sns:Publish"],  
     "Effect": "Allow",  
     "Resource": "arn:aws:sns:us-east-2:123456789012:MyTopic",  
     "Principal": "  
       "AWS": "arn:aws:iam:123456789012:user/MyUser"  
     }  
   ]  
}
```

- **Note:** the IAM user can still use other SQS API from outside the VPC Endpoint
- You could add an SQS queue policy to deny any action not done through the VPC endpoint

# VPC Endpoint Policy & S3 bucket policy

- VPC Endpoint Policy to restrict access to bucket “my\_secure\_bucket”

```
{  
    "Statement": [  
        {  
            "Sid": "Access-to-specific-bucket-only",  
            "Principal": "*",  
            "Action": [  
                "s3:GetObject",  
                "s3:PutObject"  
            ],  
            "Effect": "Allow",  
            "Resource": ["arn:aws:s3:::my_secure_bucket",  
                        "arn:aws:s3:::my_secure_bucket/*"]  
        }  
    ]  
}
```

# VPC Endpoint Policy & S3 bucket policy

- VPC Endpoint Policy to allow access to Amazon Linux 2 repositories

```
{  
    "Statement": [  
        {  
            "Sid": "AmazonLinux2AMIRRepositoryAccess",  
            "Principal": "*",  
            "Action": [  
                "s3:GetObject"  
            ],  
            "Effect": "Allow",  
            "Resource": [  
                "arn:aws:s3:::amazonlinux.*.amazonaws.com/*"  
            ]  
        }  
    ]  
}
```

# VPC Endpoint Policy & S3 bucket policy

- S3 bucket policy may have
  - Condition: "aws:sourceVpce": "vpce-1a2b3c4d" to Deny any traffic that doesn't come from a specific VPC endpoint (more secure)
  - Condition: "aws:sourceVpc": "vpc-111bbb22" for a specific VPC
- The **aws:sourceVpc** condition only works for VPC Endpoints, in case you have multiple endpoints and want to manage access to your S3 buckets for all your endpoints
- The S3 bucket policies can restrict access only from a specific public IP address or an elastic IP address. You can't restrict based on private IP
- Therefore aws:SourceIp condition doesn't apply for VPC endpoints

# Example S3 bucket policies

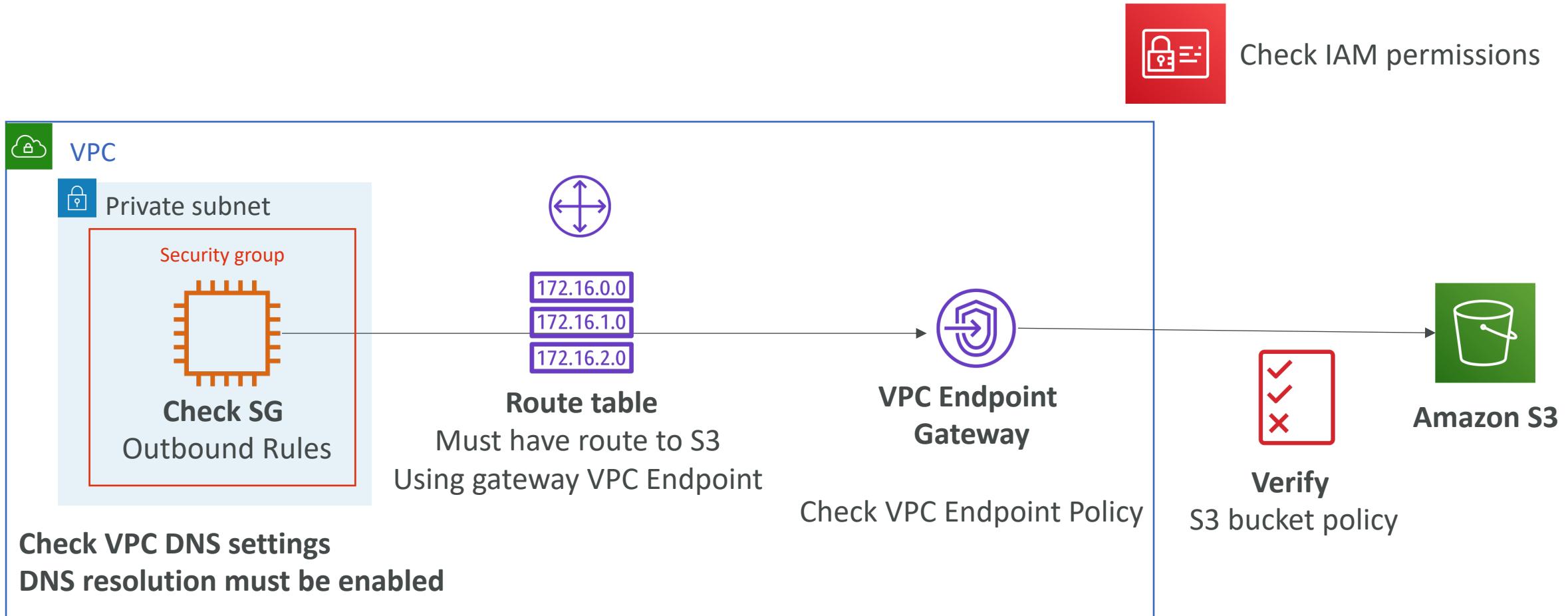
- S3 bucket policy to restrict to one specific VPC Endpoint

```
{  
  "Version": "2012-10-17",  
  "Id": "Policy1415115909152",  
  "Statement": [  
    {  
      "Sid": "Access-to-specific-VPCE-only",  
      "Principal": "*",  
      "Action": "s3:*",  
      "Effect": "Deny",  
      "Resource": ["arn:aws:s3::::my_secure_bucket",  
                  "arn:aws:s3::::my_secure_bucket/*"],  
      "Condition": {  
        "StringNotEquals": {  
          "aws:sourceVpce": "vpce-1a2b3c4d"  
        }  
      }  
    }  
  ]  
}
```

- S3 bucket policy to restrict to an entire VPC (multiple VPC Endpoints)

```
{  
  "Version": "2012-10-17",  
  "Id": "Policy1415115909152",  
  "Statement": [  
    {  
      "Sid": "Access-to-specific-VPCE-only",  
      "Principal": "*",  
      "Action": "s3:*",  
      "Effect": "Deny",  
      "Resource": ["arn:aws:s3::::my_secure_bucket",  
                  "arn:aws:s3::::my_secure_bucket/*"],  
      "Condition": {  
        "StringNotEquals": {  
          "aws:sourceVpc": "vpc-111bbb22"  
        }  
      }  
    }  
  ]  
}
```

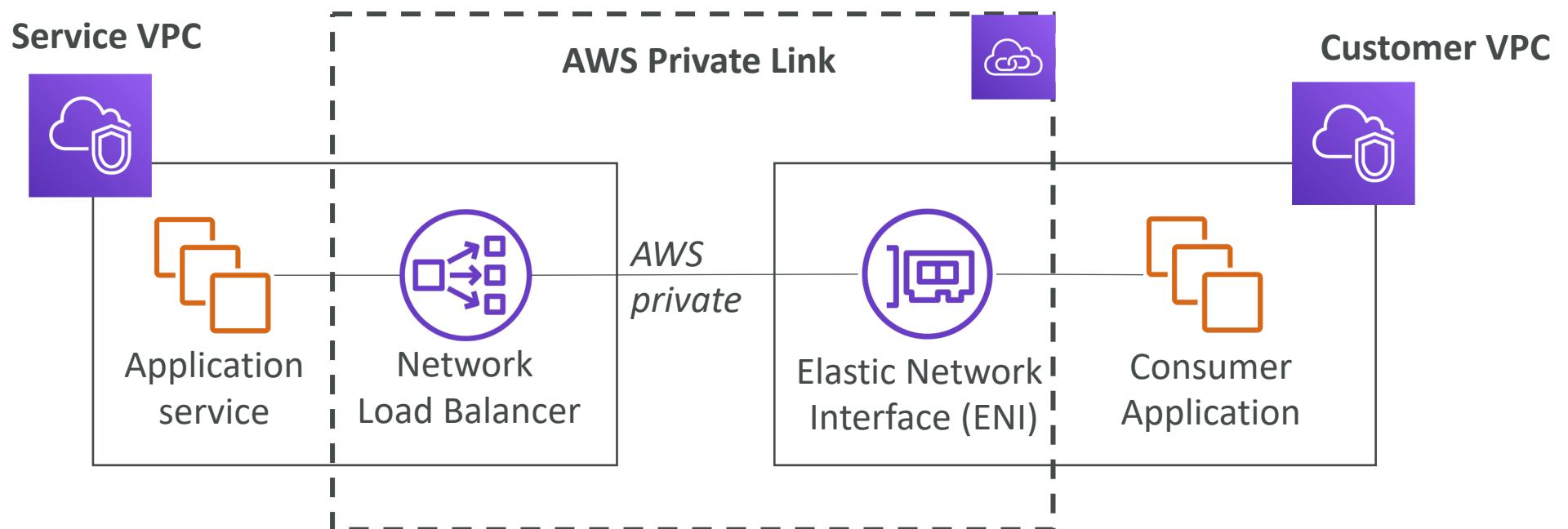
# VPC Endpoint Policies for S3 Troubleshooting



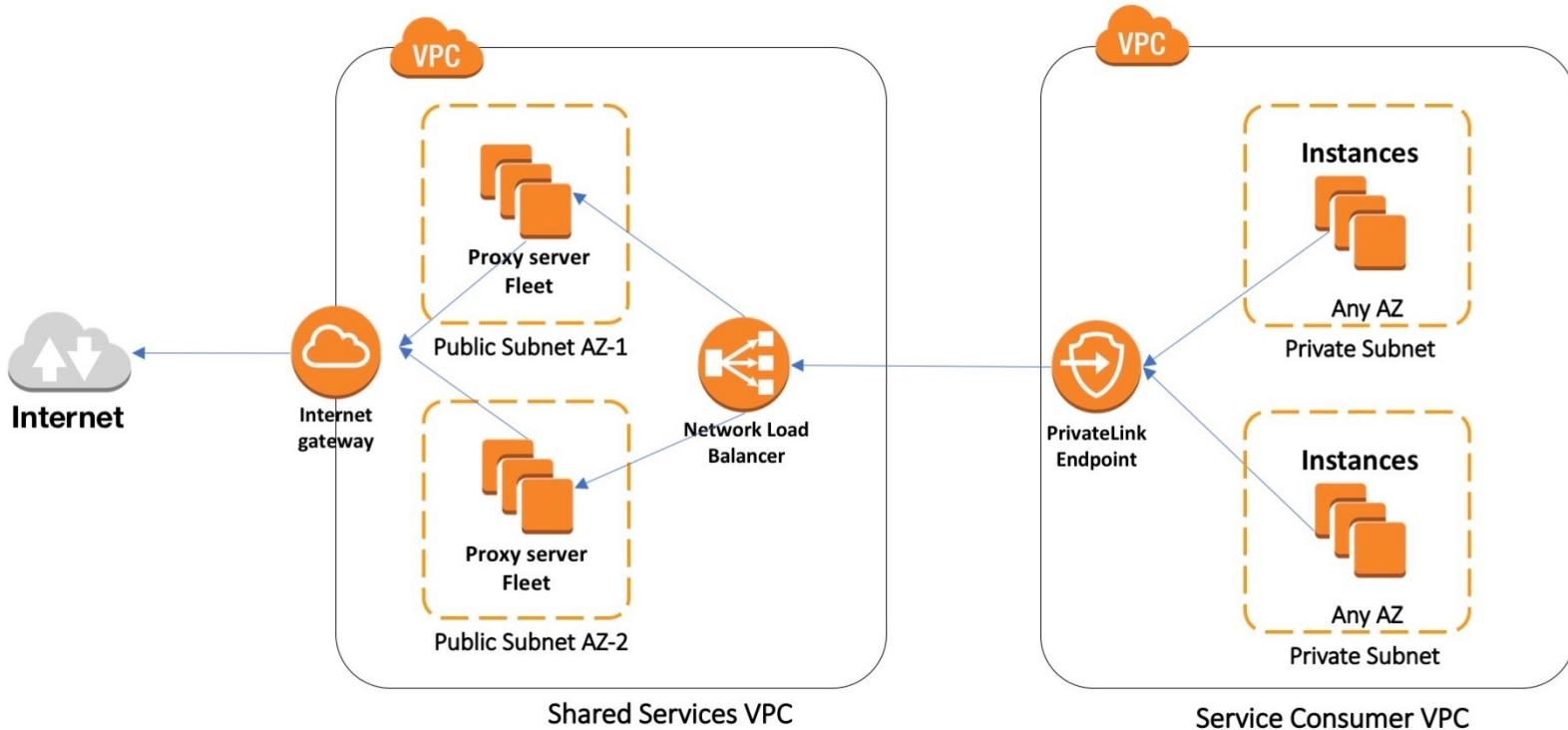
# AWS PrivateLink (VPC Endpoint Services)



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



# Secure and Scale Web Filtering using Explicit Proxy

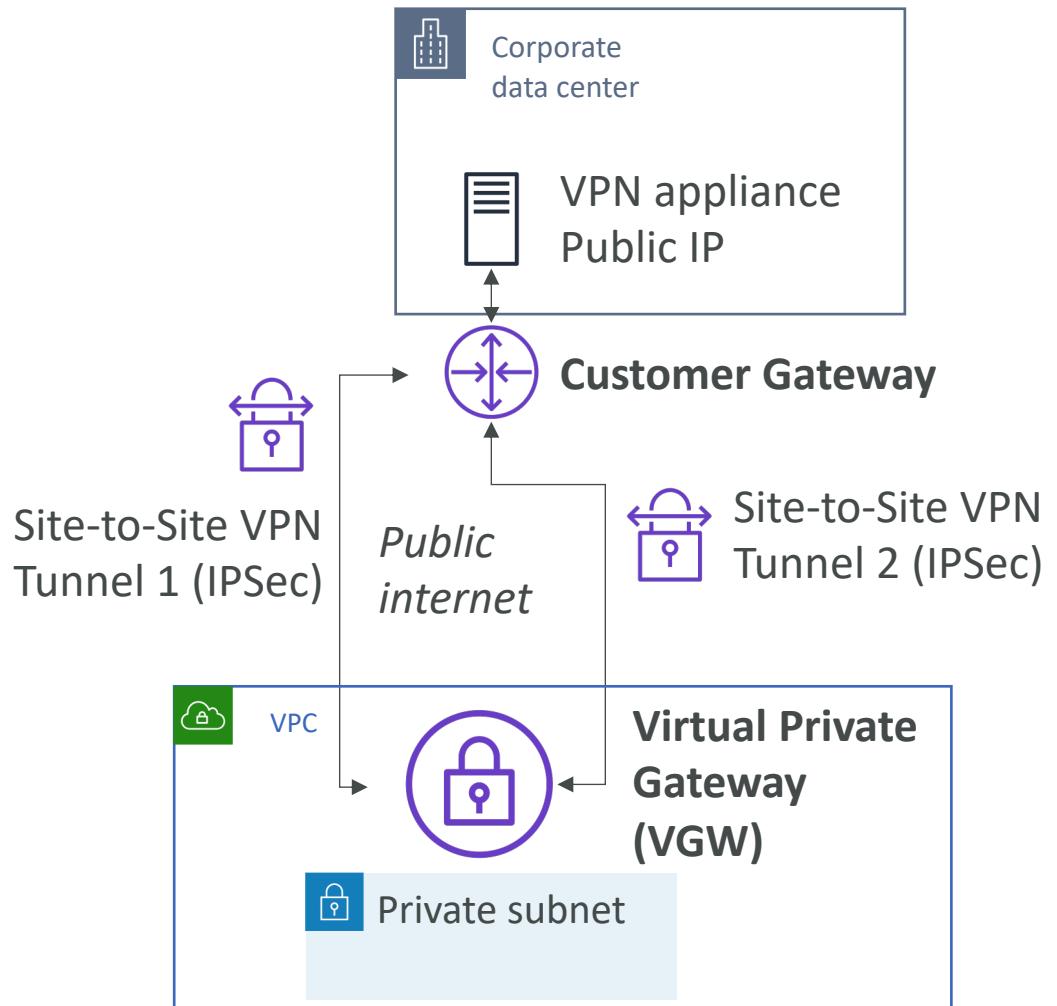


<https://aws.amazon.com/blogs/networking-and-content-delivery/how-to-use-aws-privatelink-to-secure-and-scale-web-filtering-using-explicit-proxy/>

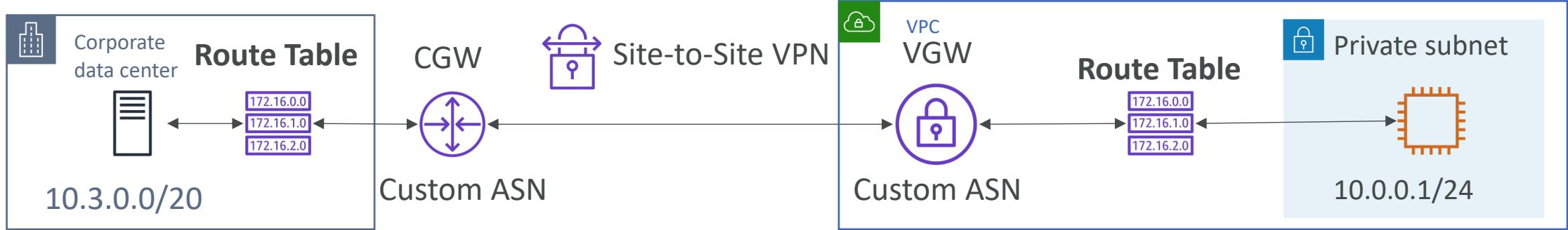
# Site to Site VPN (AWS Managed VPN)



- **on-premises:**
  - Setup a software or hardware VPN appliance to your on-premises network.
  - The on-premises VPN should be accessible using a public IP
- **AWS-side:**
  - Setup a Virtual Private Gateway (VGW) and attach to your VPC
  - Setup a Customer Gateway to point the on-premises VPN appliance
- Two VPN connections (tunnels) are created for redundancy, encrypted using IPSec
- Can optionally accelerate it using Global Accelerator (for worldwide networks)



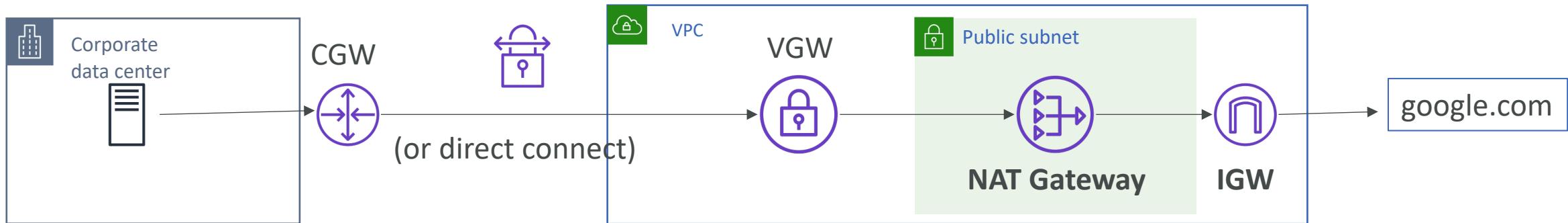
# Route Propagation in Site-to-Site VPN



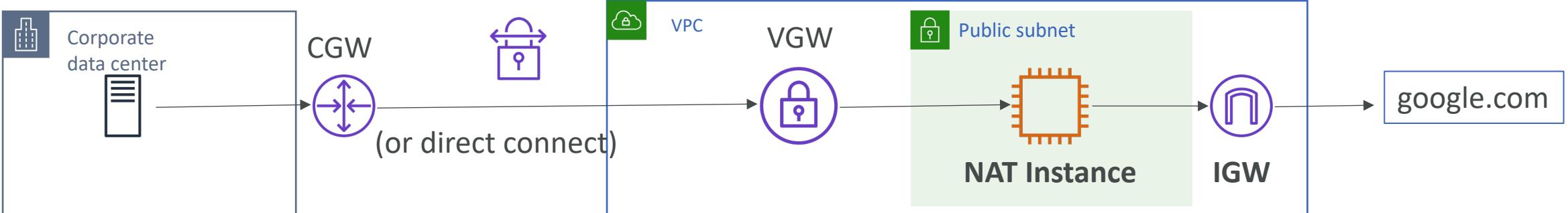
- **Static Routing:**
  - Create static route in corporate data center for 10.0.0.1/24 through the CGW
  - Create static route in AWS for 10.3.0.0/20 through the VGW
- **Dynamic Routing (BGP):**
  - Uses BGP (Border Gateway Protocol) to share routes automatically (eBGP for internet)
  - We don't need to update the routing tables, it will be done for us dynamically
  - Just need to specify the ASN (Autonomous System Number) of the CGW and VGW

# Site to Site VPN and Internet Access

- NOT OKAY (blocked by NAT Gateway restrictions)

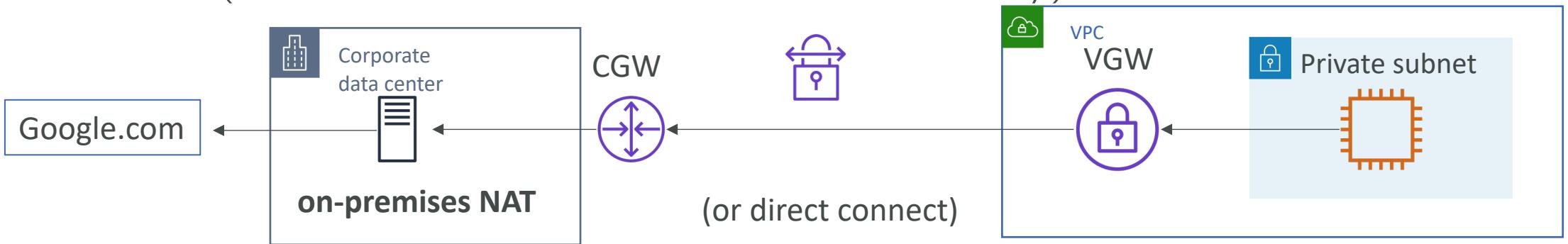


- OKAY (self managed NAT Instance – more control)



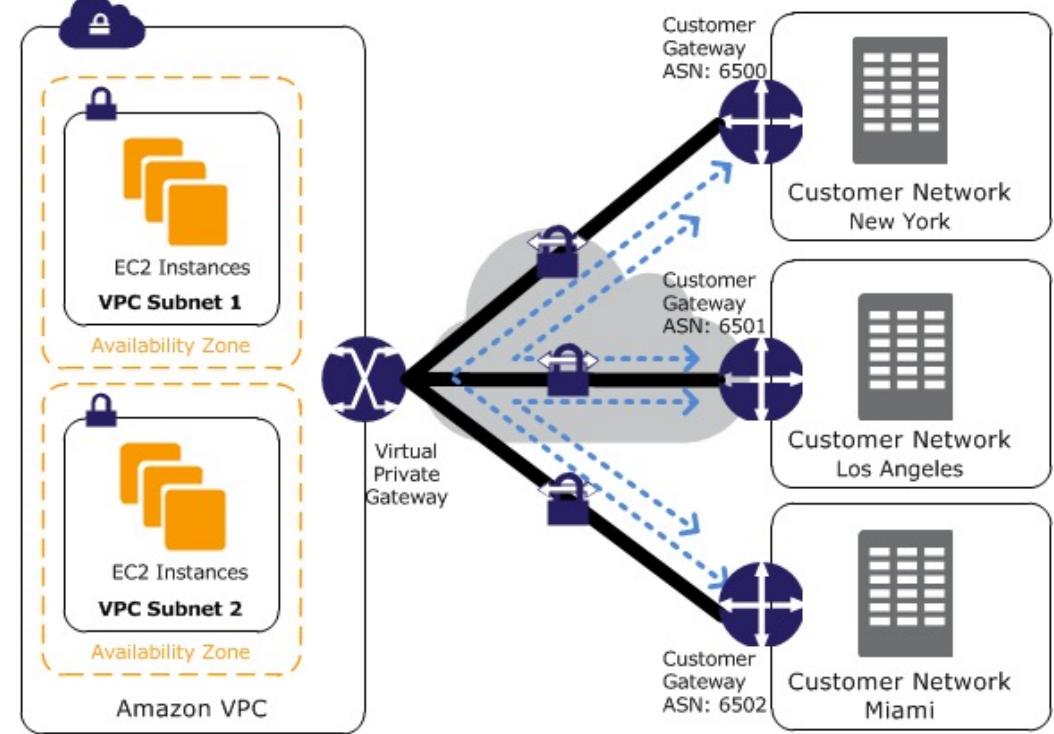
# Site to Site VPN and Internet Access

- **OKAY** (alternative to NAT Instances / Gateway)



# AWS VPN CloudHub

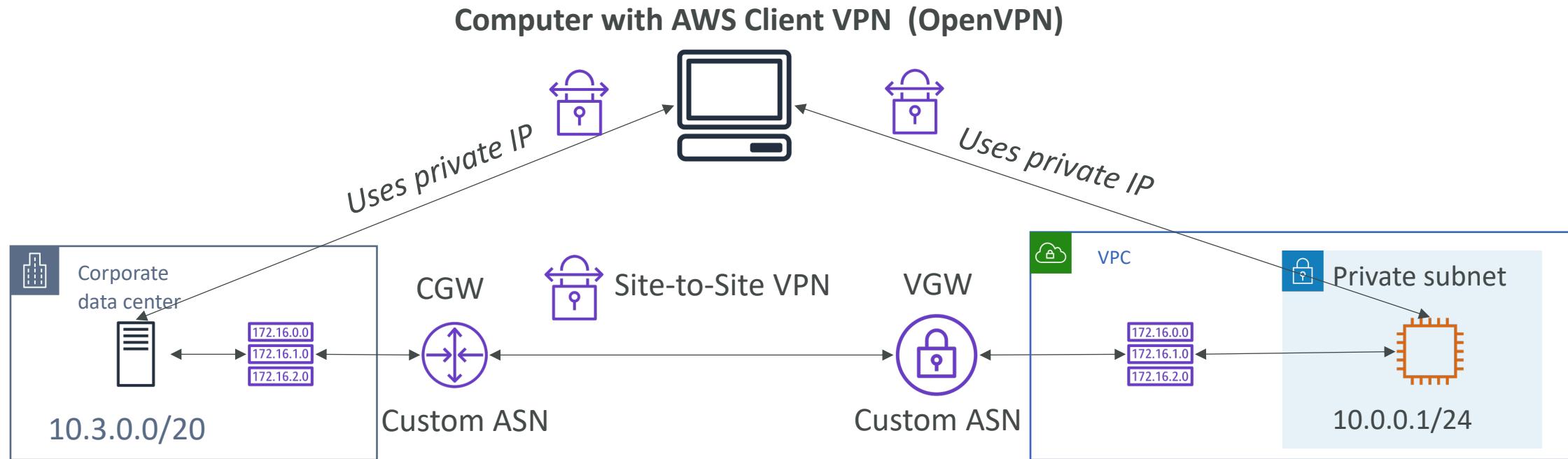
- Can connect up to 10 Customer Gateway for each Virtual Private Gateway (VGW)
- Low cost hub-and-spoke model for primary or secondary network connectivity between locations
- Provide secure communication between sites, if you have multiple VPN connections
- It's a VPN connection so it goes over the public internet
- Can be a **failover connection** between your on-premises locations



# AWS Client VPN

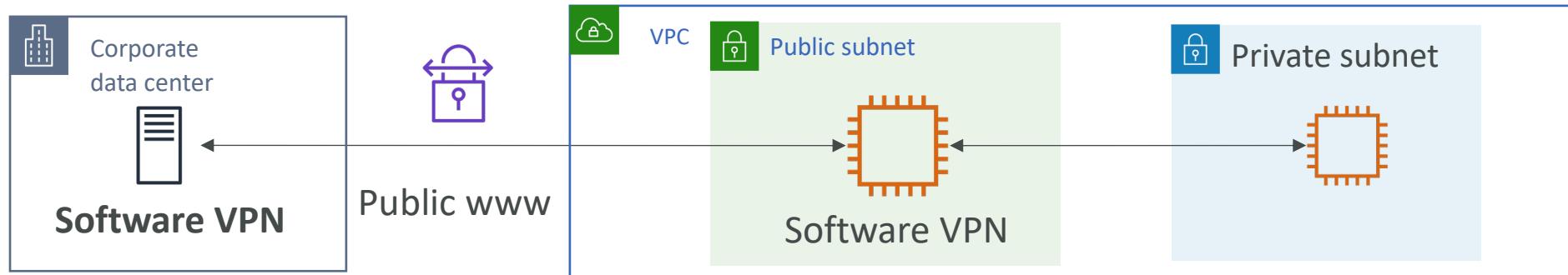


- Connect from your computer using OpenVPN to your private network in AWS and on-premises



# Software VPN (not AWS managed)

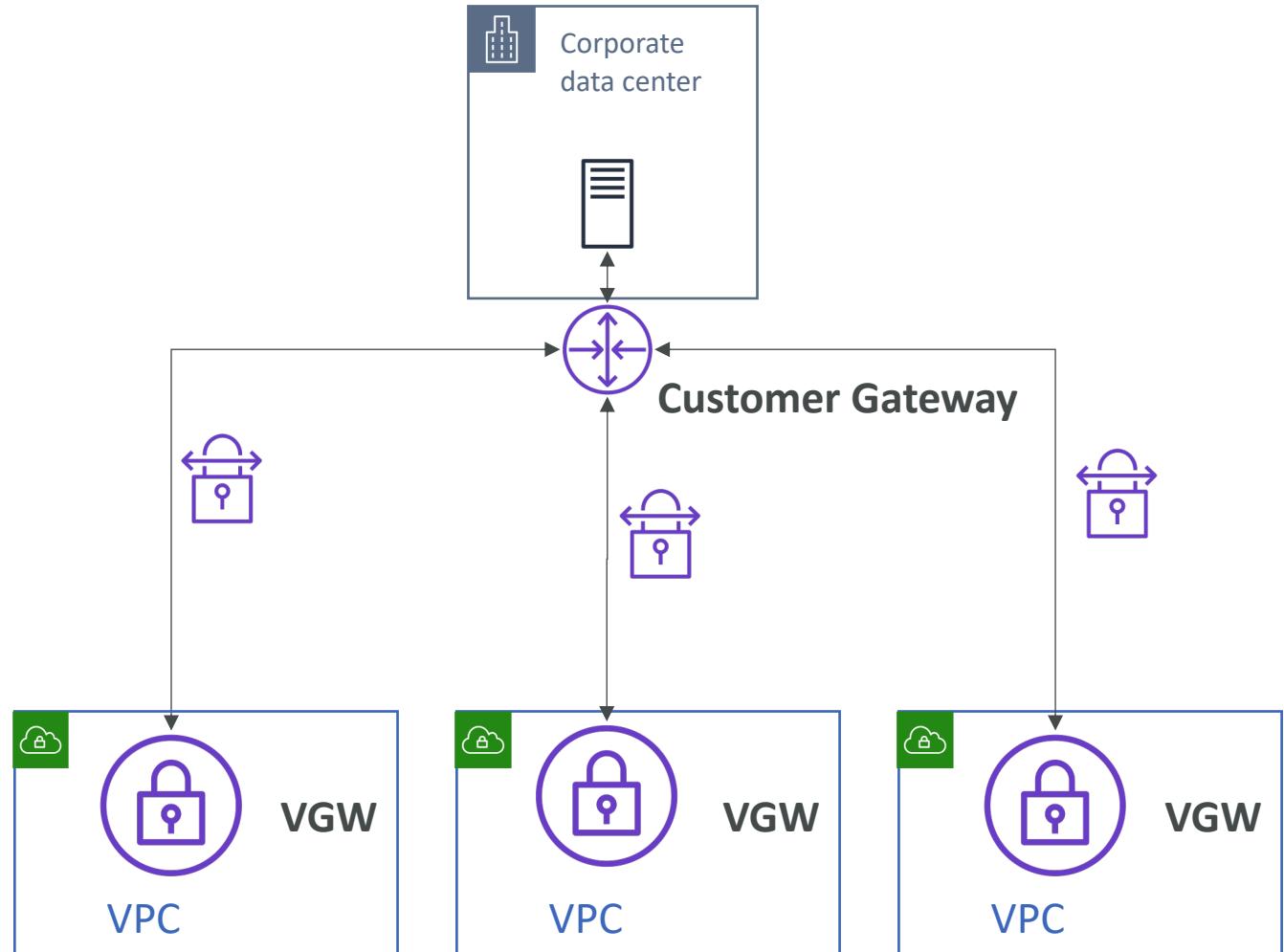
- You can setup your own software VPN, but you have to manage everything including bandwidth, redundancy, etc.



- You would have more control over the setup and routing options

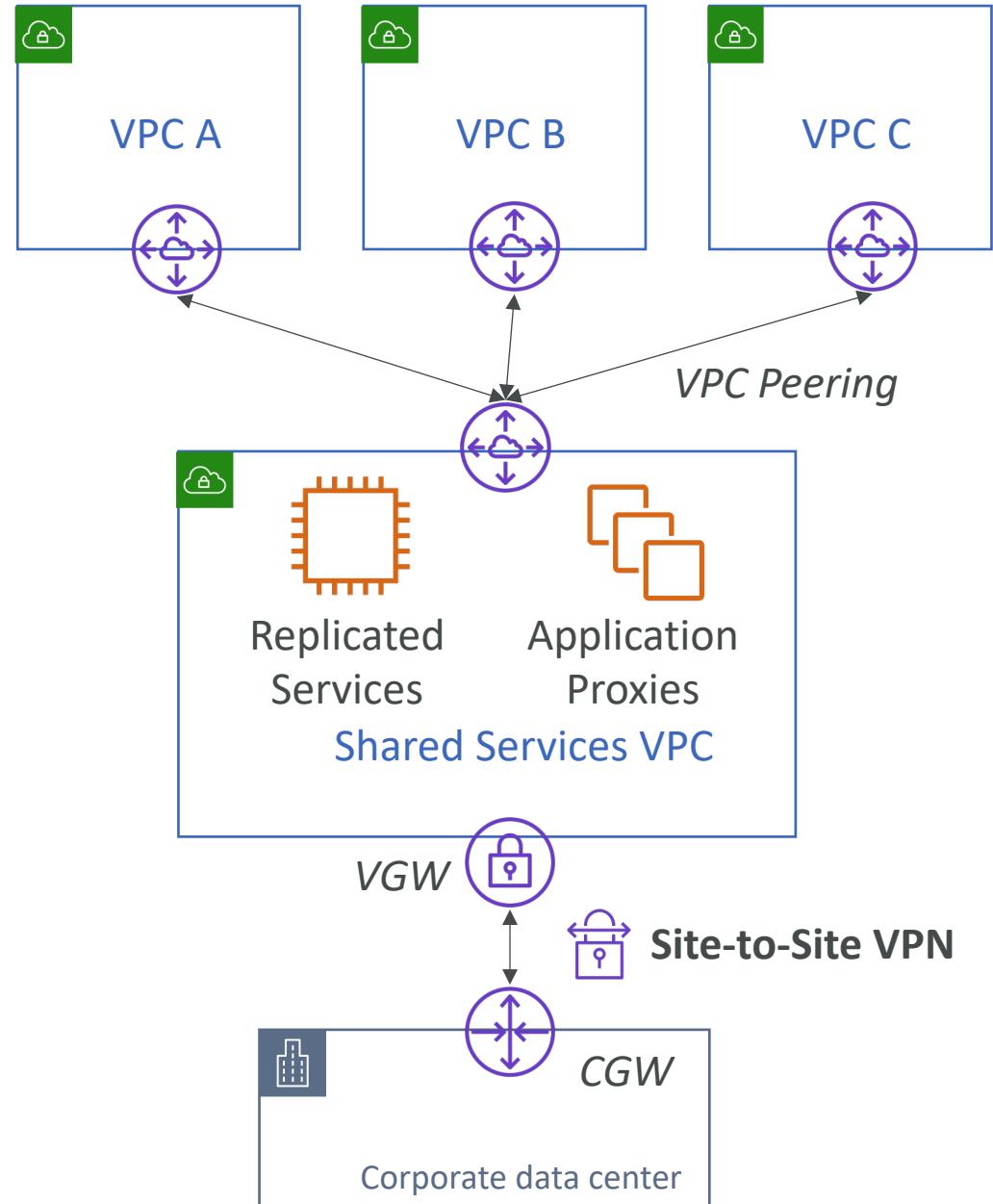
# VPN to multiple VPC

- For VPN-based customers, AWS recommends creating a separate VPN connection for each customer VPC.
- Direct Connect is recommended because it has a Direct Connect Gateway

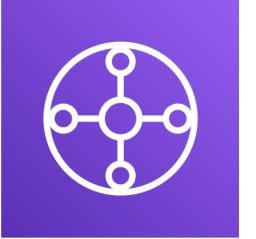


# Shared Services VPC

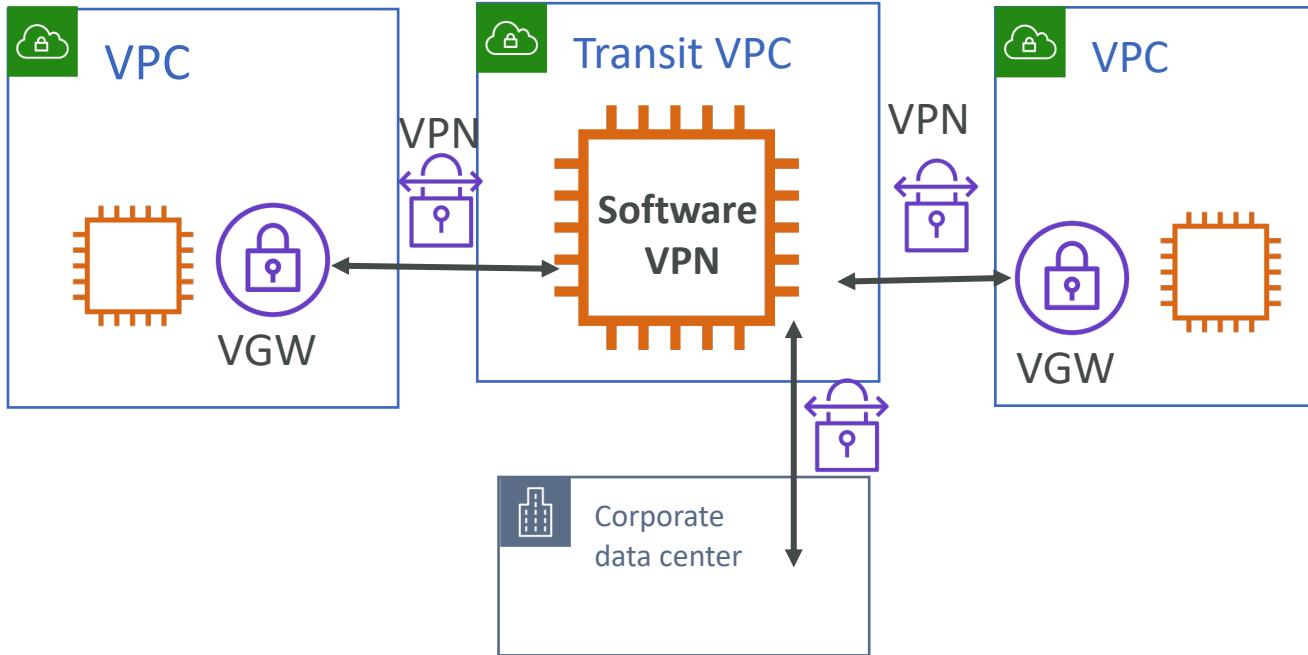
- Create a VPN connection between on-premises and shared service VPC
- Replicate services, applications, databases between on-premises and the Shared Services VPC or deploy proxies in the shared service VPC
- Do VPC peering between the VPC and the shared service VPC
- VPCs can directly access the Shared Service VPC services and do not need VPN connections to on-premises



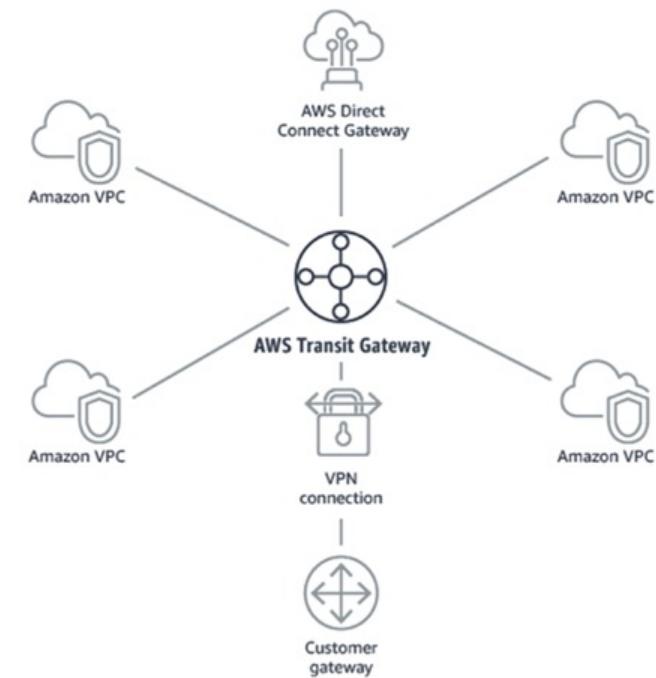
# Other Solutions



- **Transit VPC (complicated)**
  - Good for resources that are hard to replicate on the cloud
  - Must use VPN as VPC peering does not support transitive routing



- **Transit Gateway (simple)**



# Direct Connect

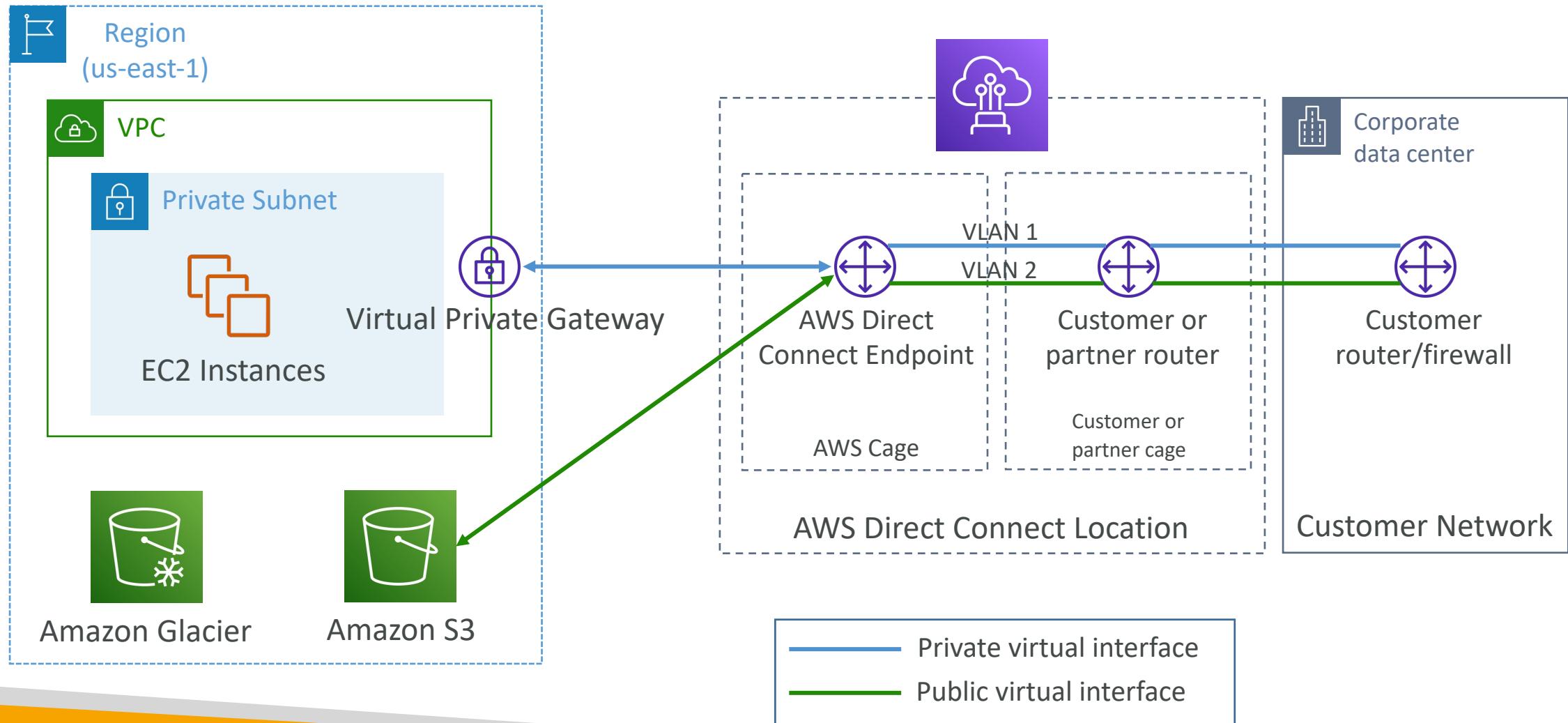


- Provides a dedicated private connection from a remote network to your VPC
- Dedicated connection must be setup between your DC and AWS Direct Connect locations
- More expensive than running a VPN solution
- Private access to AWS services through VIF
- Bypass ISP, reduce network cost, increase bandwidth and stability
- Not redundant by default (must setup a failover DX or VPN)

# Direct Connect – Virtual Interfaces (VIF)

- Public VIF – connect to Public AWS Endpoints (S3 buckets, EC2 service, anything AWS ...)
- Private VIF – connect to resources in your VPC (EC2 instances, ALB, ...)
- Transit Virtual Interface – connect to resources in a VPC using a Transit Gateway
- VPC Endpoints can't be accessed through Private VIF (you don't need them)

# Direct Connect Diagram

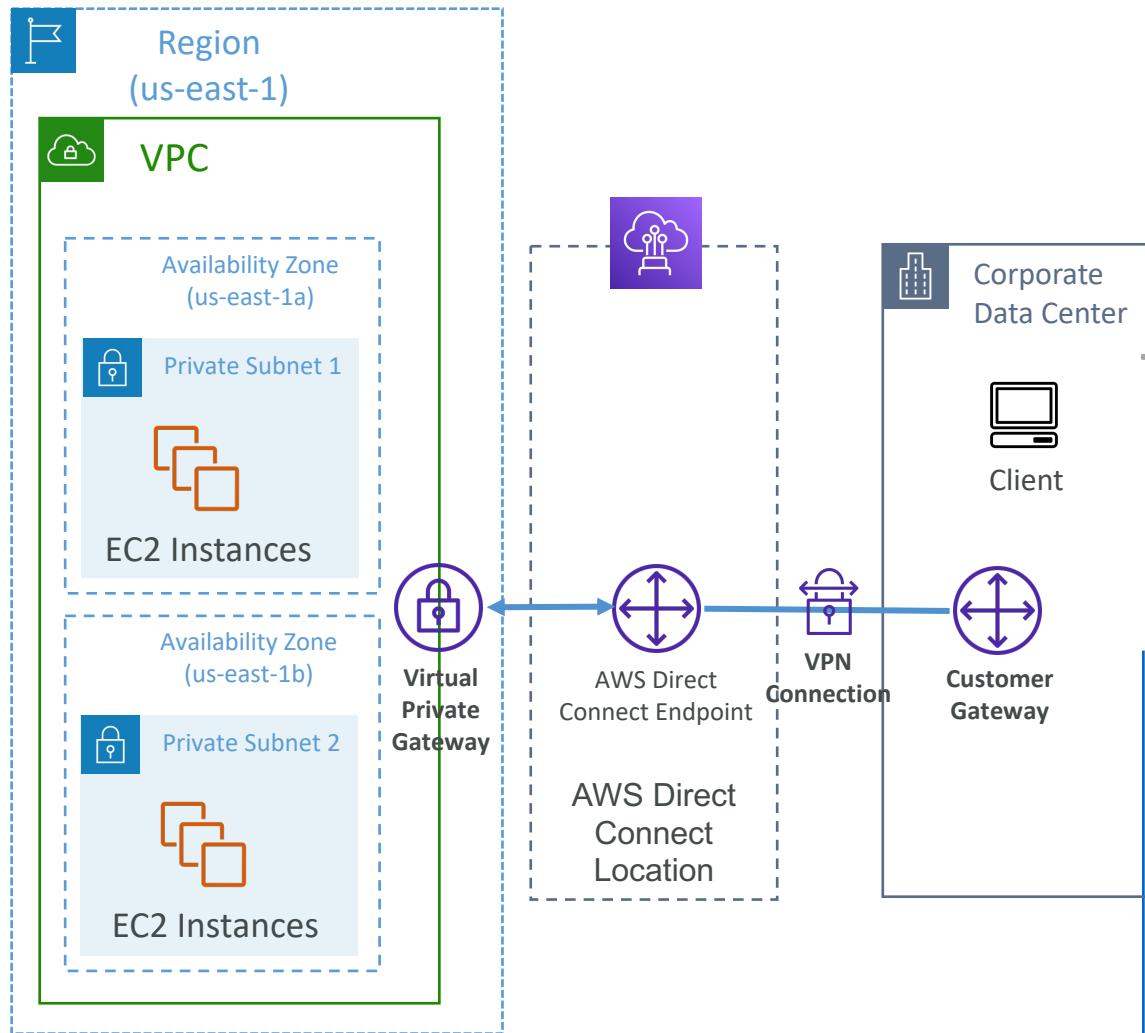


# Direct Connect – Connection Types

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

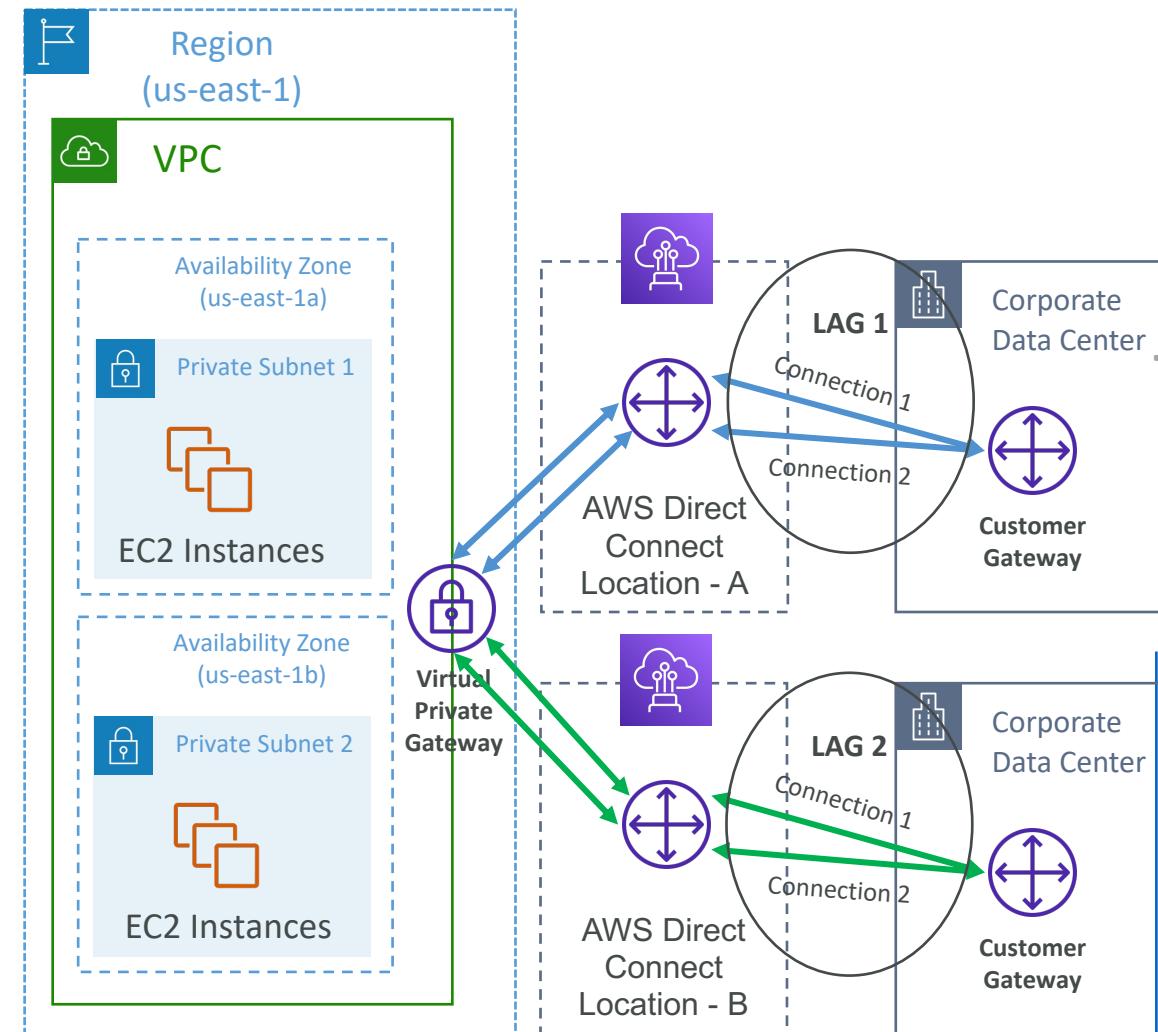
# Direct Connect – Encryption

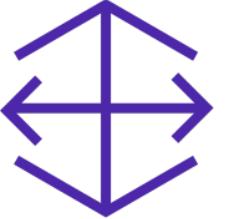
- Data in transit is not encrypted but is private
- AWS Direct Connect + VPN provides an IPsec-encrypted private connection
- VPN over Direct Connect connection Uses Public VIF
- Good for an extra level of security, but slightly more complex to put in place



# Direct Connect – Link Aggregation Groups (LAG)

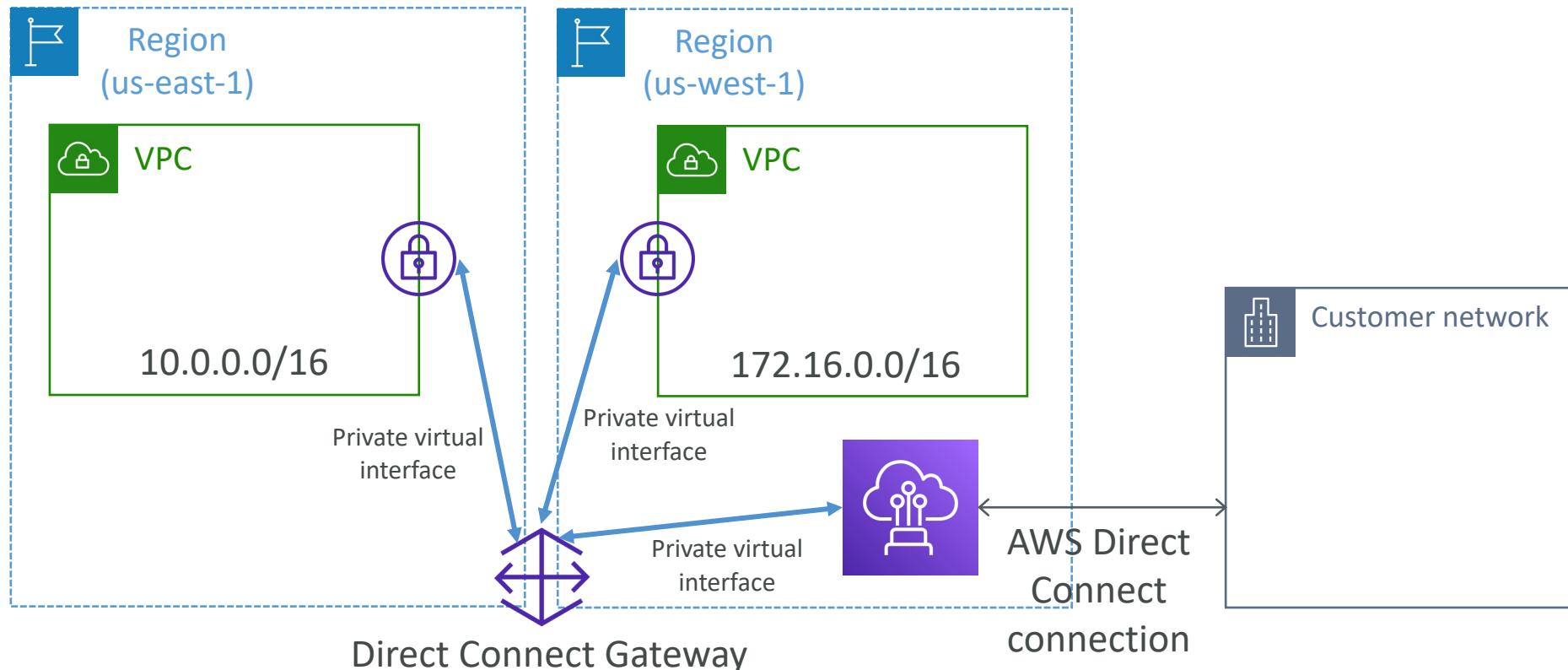
- Get increased speed and failover by summing up existing DX connections into a **logical one**
- Can aggregate up to 4 connections (active-active mode)
- Can add connections over time to the LAG
- All connections in the LAG:
  - Must be dedicated connections
  - Must have the same bandwidth
  - Must terminate at the same AWS Direct Connect Endpoint
- Can set a minimum number of connections for the LAG to function



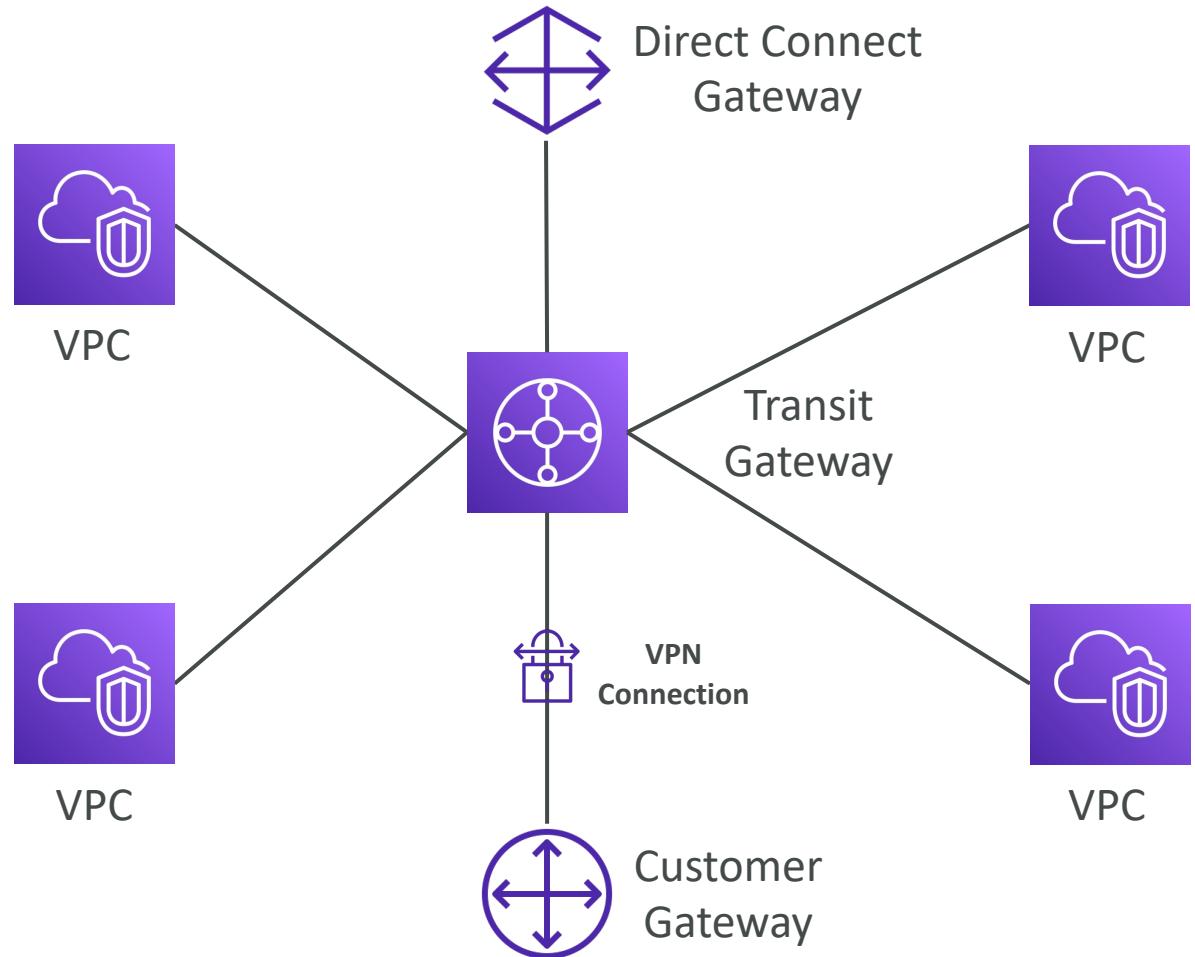


# Direct Connect Gateway

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

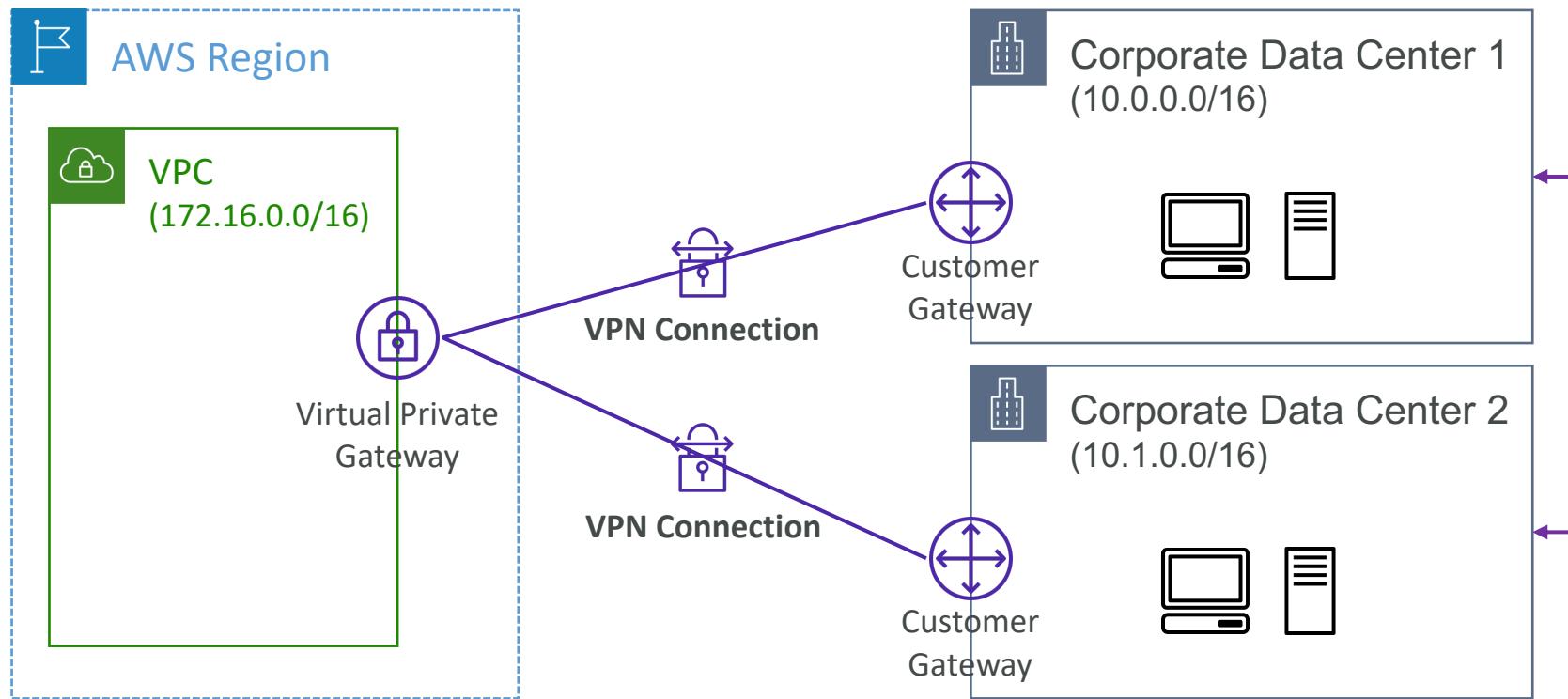


# Direct Connect Gateway + Transit Gateway



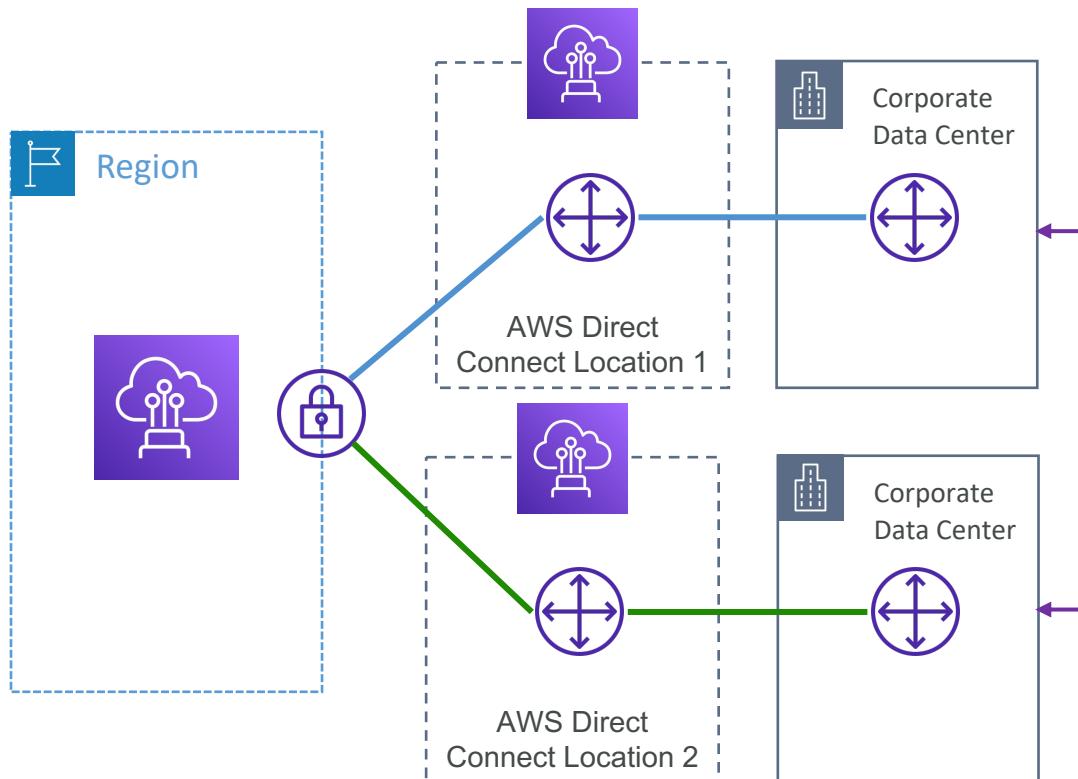
# Site-to-Site Active-Active Connection

## Active-Active VPN Connection



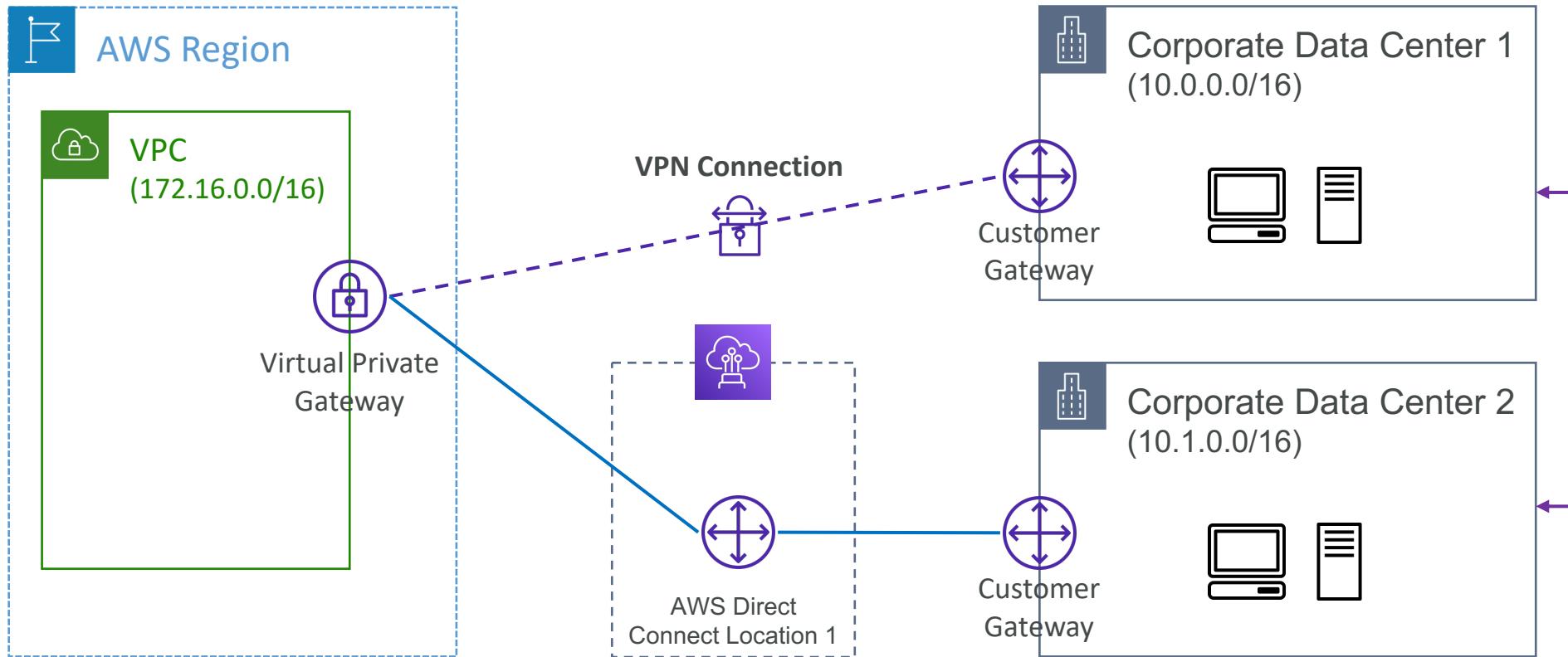
# Direct Connect – High Availability

Multiple connections at multiple AWS Direct Connect locations



# Direct Connect – High Availability

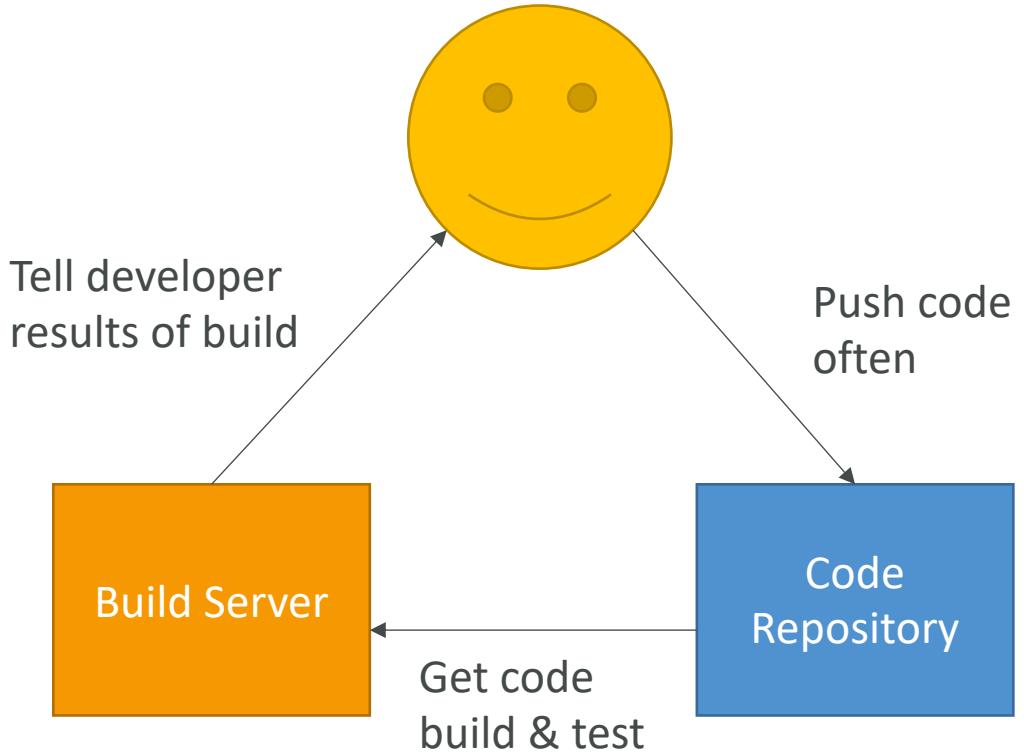
## Backup VPN Connection



# Other Services Section

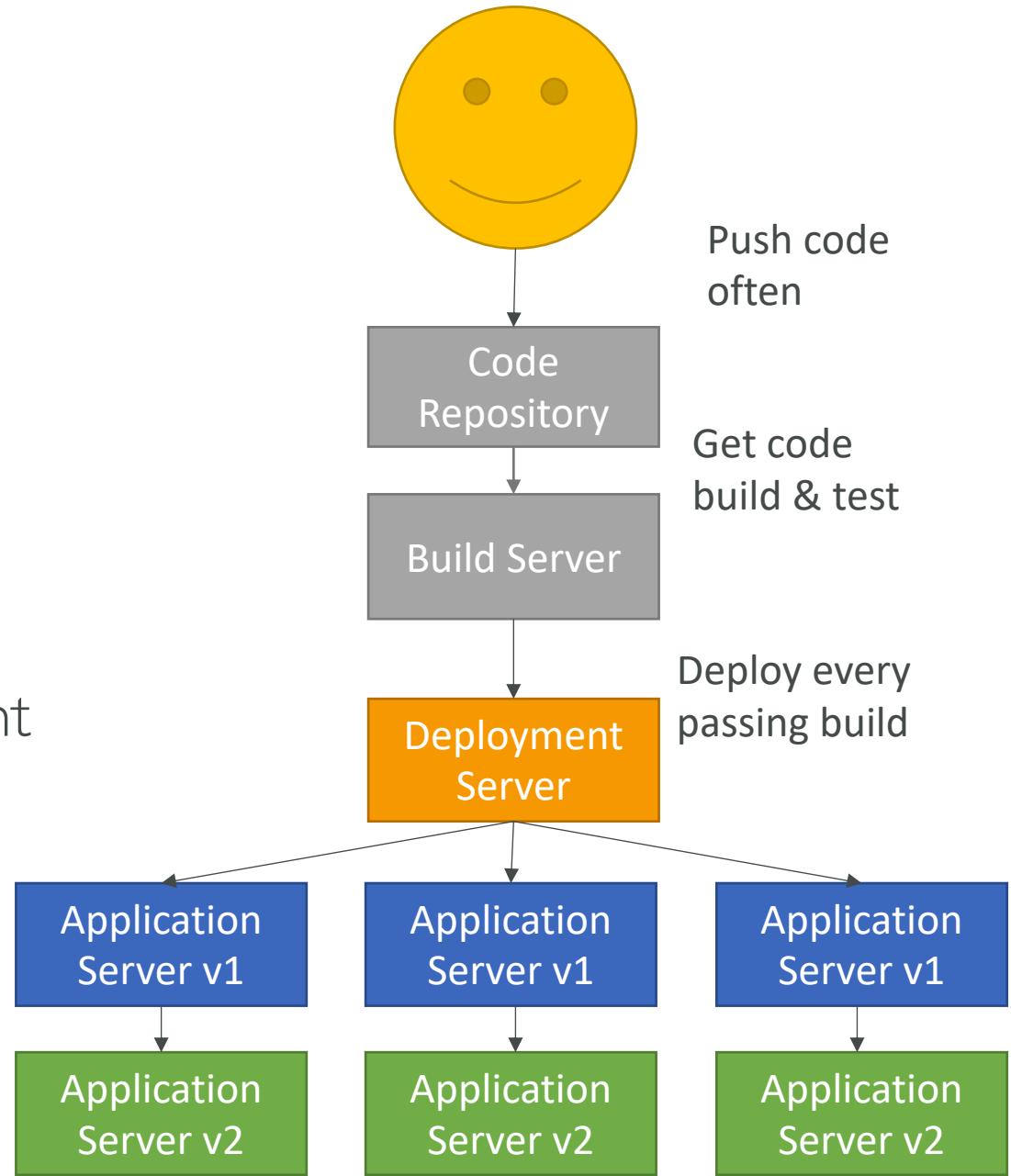
# Continuous Integration

- Developers push the code to a code repository often (GitHub / CodeCommit / Bitbucket / etc...)
- A testing / build server checks the code as soon as it's pushed (CodeBuild / Jenkins CI / etc...)
- The developer gets feedback about the tests and checks that have passed / failed
- Find bugs early, fix bugs
- Deliver faster as the code is tested
- Deploy often
- Happier developers, as they're unblocked

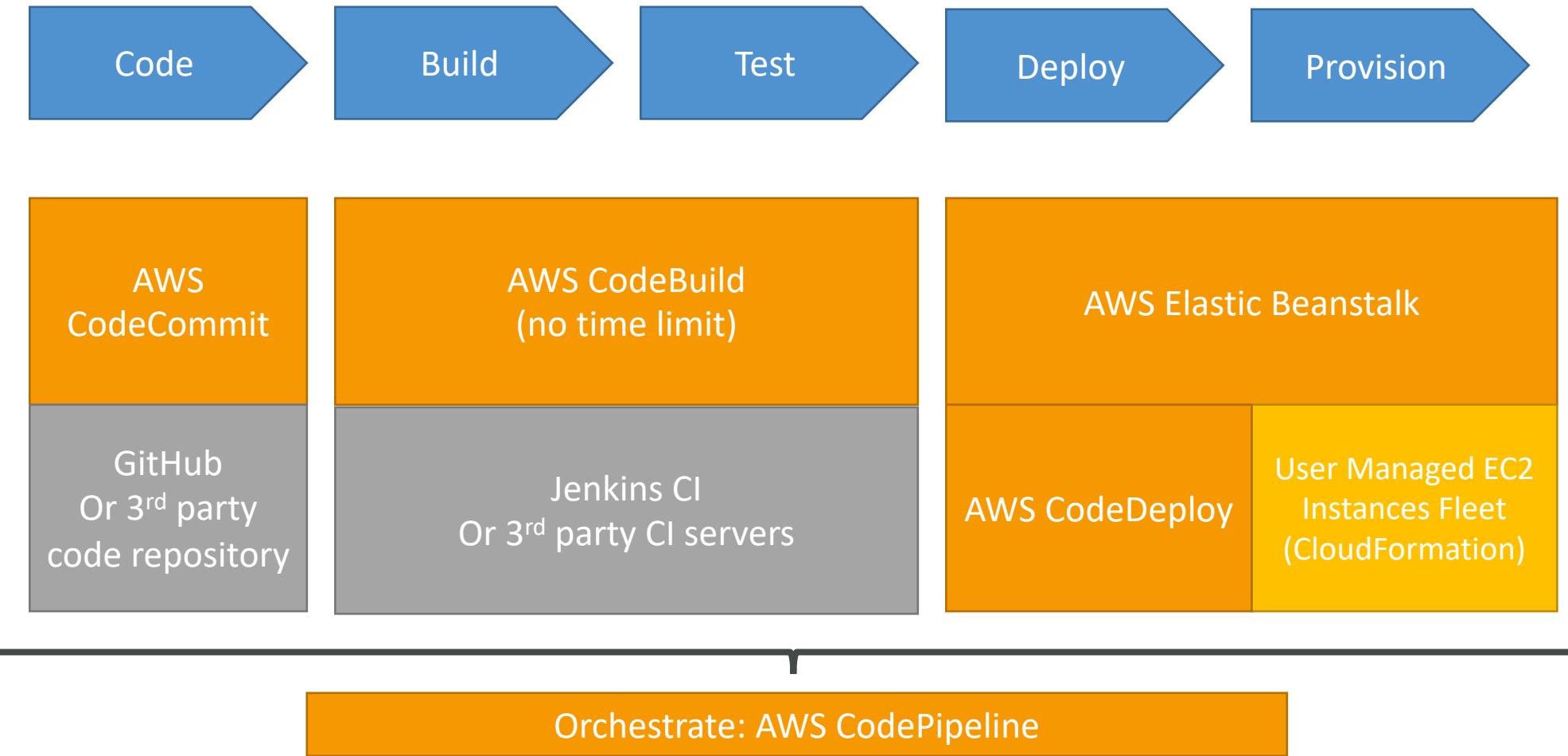


# Continuous Delivery

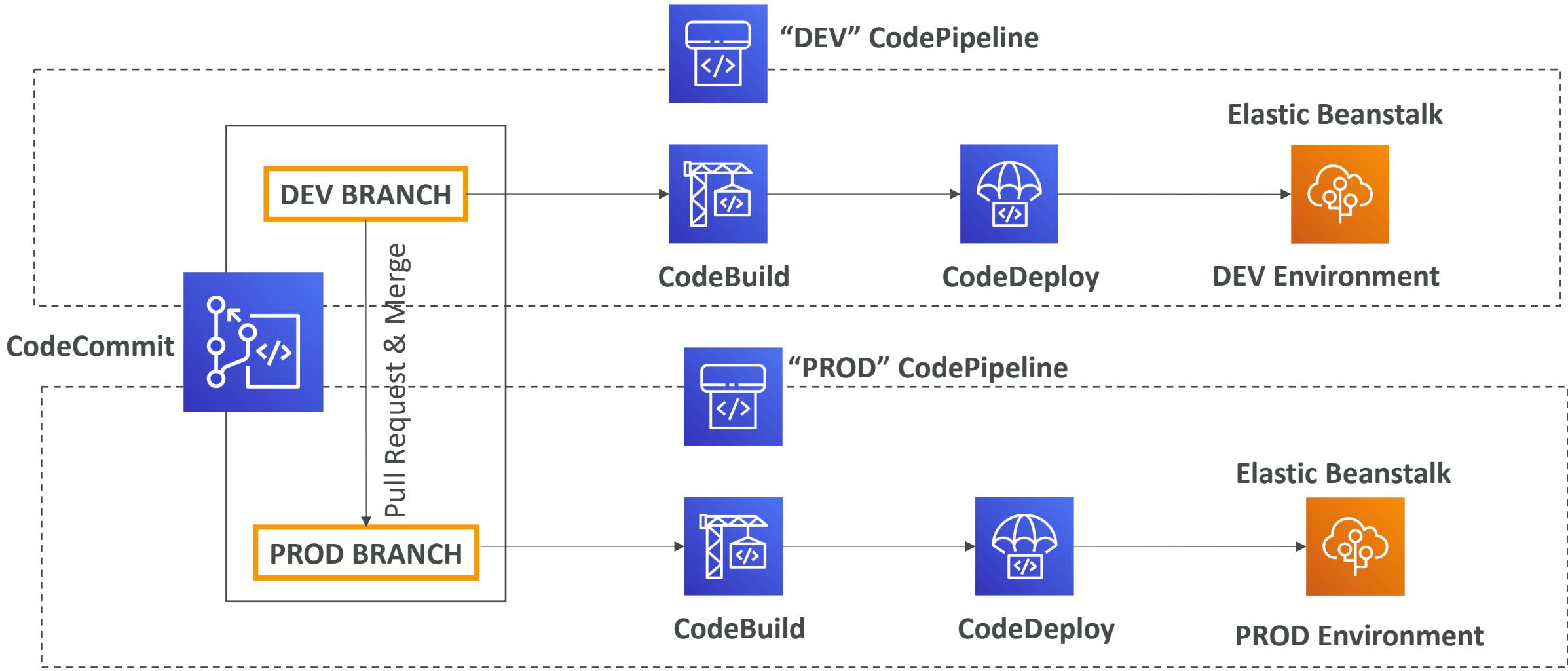
- Ensure that the software can be released reliably whenever needed.
- Ensures deployments happen often and are quick
- Shift away from “one release every 3 months” to “5 releases a day”
- That usually means automated deployment
  - CodeDeploy
  - Jenkins CD
  - Spinnaker
  - Etc...



# Technology Stack for CICD

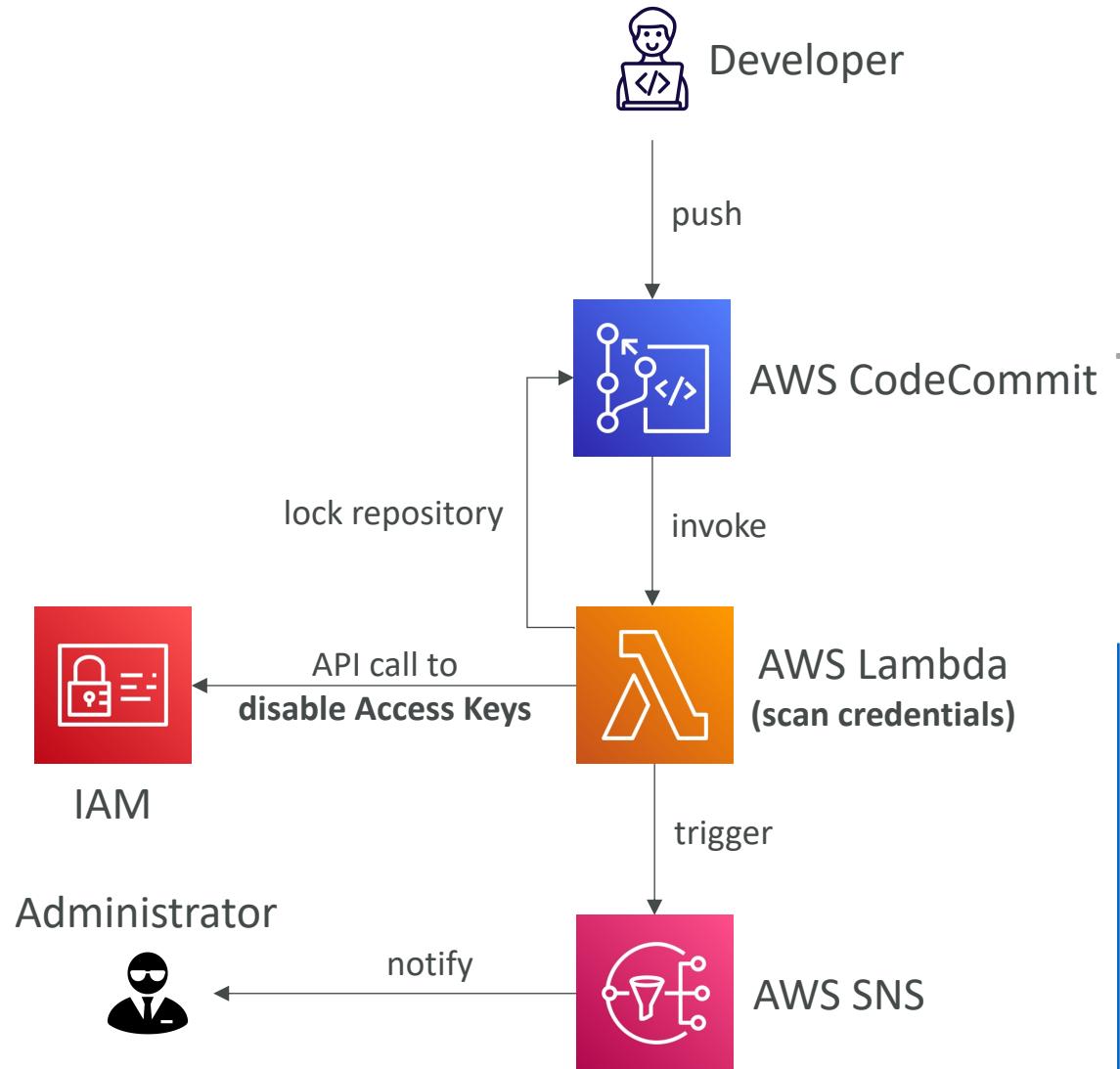


# CICD Architecture



# CodeCommit Trigger for AWS Lambda

- Every push to CodeCommit can trigger a Lambda function
- The Lambda function can scan for leaked AWS credentials on every code push, and disable them automatically to remedy the issue



# Good to know – CICD

- You can use a **manual approval stage** in CodePipeline
- Running unit tests CodeCommit + CodeBuild + Code Pipeline



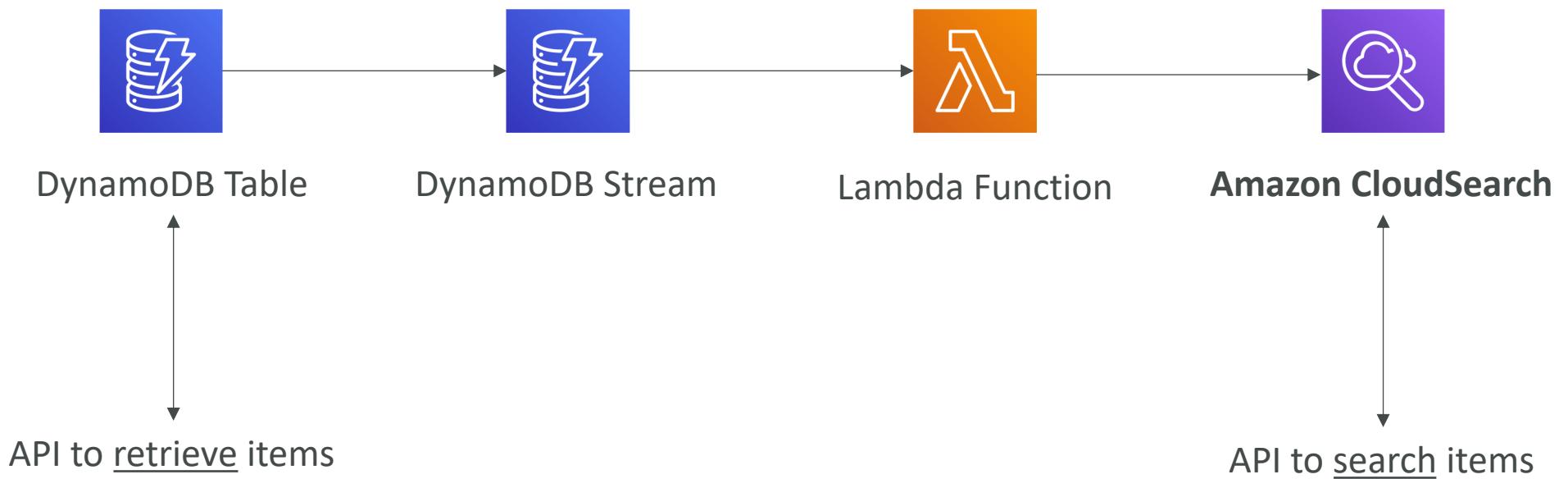
- Build and Store Docker Images: CodeBuild + ECR



- Automated CloudFormation deployment: CodePipeline

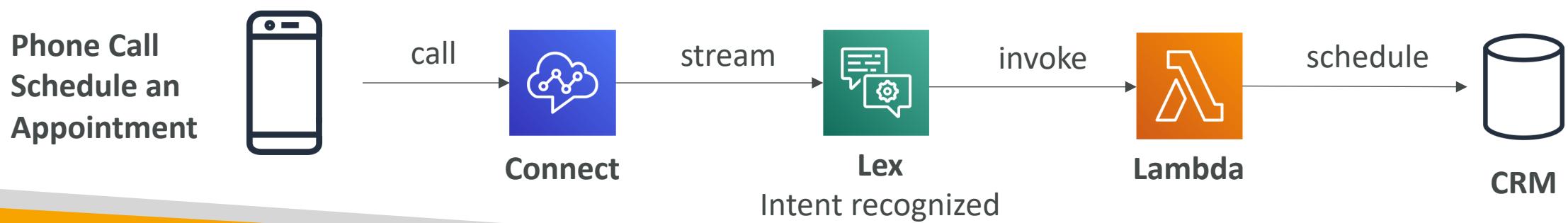
# CloudSearch

- Managed service to setup, manage and scale a search solution
- Managed alternative to ElasticSearch
- Free text, Boolean, autocomplete suggestions, geospatial search...



# Alexa for Business, Lex & Connect

- Alexa for Business:
  - Use Alexa to help employees be more productive in meeting rooms and their desk
  - Measure and increase the utilization of meeting rooms in their workplace
- Amazon Lex: (same technology that powers Alexa)
  - Automatic Speech Recognition (ASR) to convert speech to text
  - Natural Language Understanding to recognize the intent of text, callers
  - Helps build chatbots, call center bots
- Amazon Connect:
  - Receive calls, create contact flows, cloud-based virtual contact center
  - Can integrate with other CRM systems or AWS

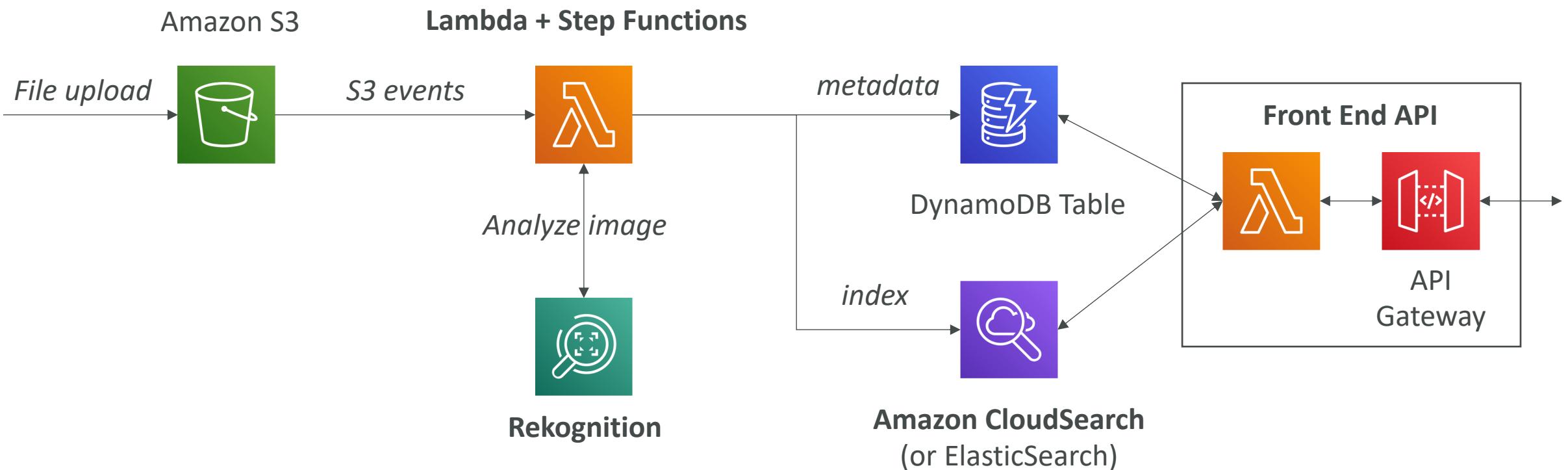


# AWS Rekognition



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

# Rekognition: static images

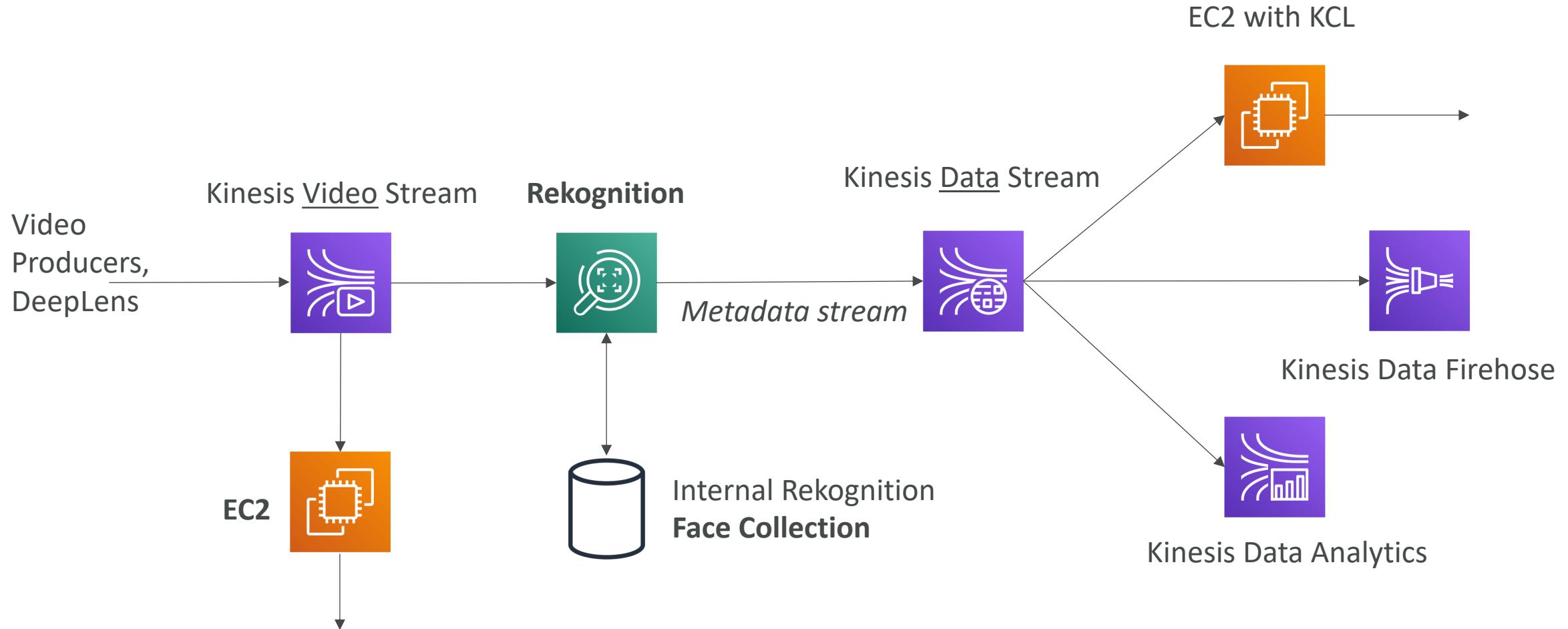




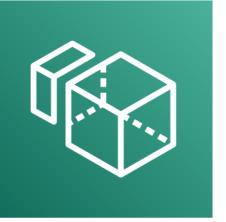
# Kinesis Video Streams

- One video stream per streaming device (producers)
  - Security cameras, body worn camera, smartphone
  - Can use a Kinesis Video Streams Producer library
- Underlying data is stored in S3 (but we don't have access to it)
- Cannot output the stream data to S3 (must build custom solution)
- Consumers:
  - Consumed by EC2 instances for real time analysis, or in batch
  - Can leverage the Kinesis Video Stream Parser Library
  - Integration with AWS Rekognition for facial detection

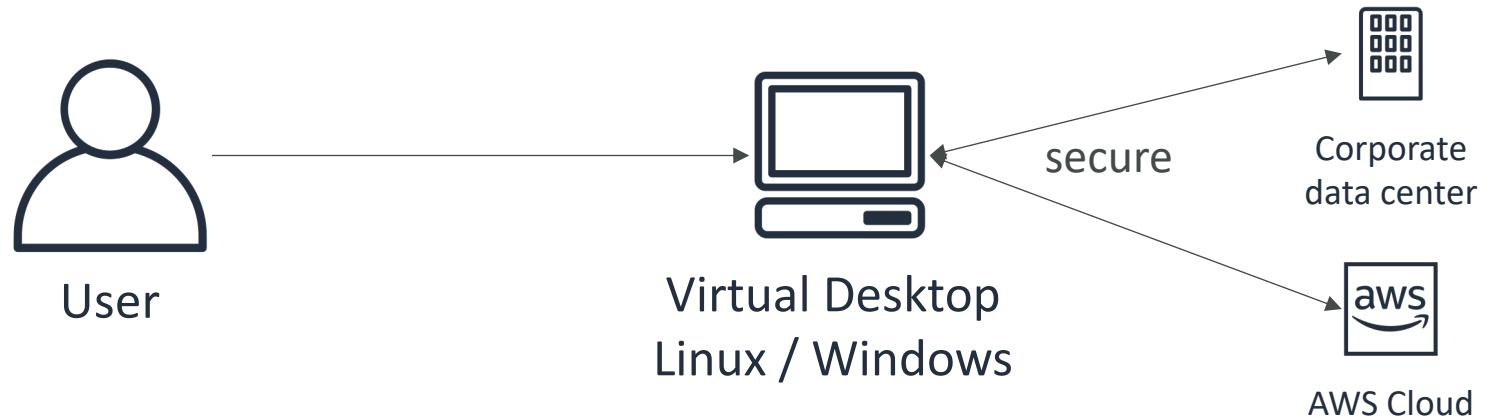
# Video Streaming & Rekognition



# AWS WorkSpaces



- Managed, Secure Cloud Desktop
- Great to eliminate management of on-premises VDI (Virtual Desktop Infrastructure)
- On Demand, pay per usage
- Secure, Encrypted, Network Isolation
- Integrated with Microsoft Active Directory



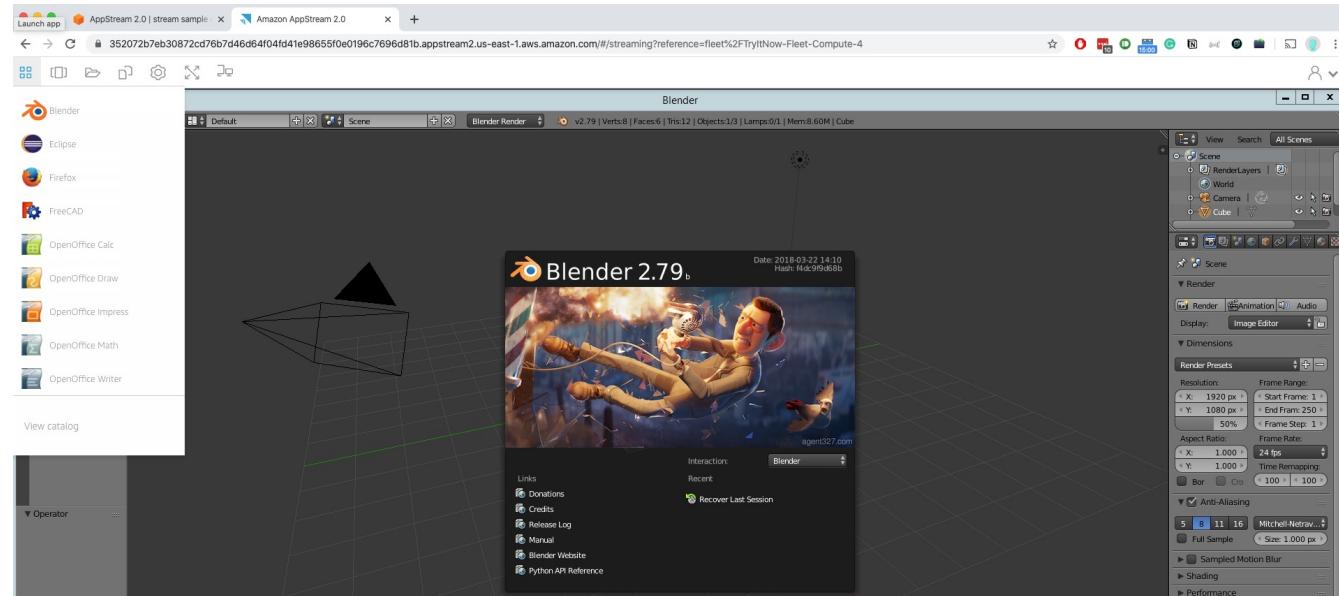
# AWS WorkSpaces

- **WorkSpaces Application Manager (WAM)**
  - Deploy and Manage applications as virtualized application containers
  - Provision at scale, and keep the applications updated using WAM
- **Windows Updates**
  - By default, Amazon Workspaces are configured to install software updates
  - Amazon WorkSpaces with Windows will have Windows Update turned on
  - You have full control over the Windows Update frequency
- **Maintenance Windows**
  - Updates are installed during maintenance windows (you define them)
  - Always On WorkSpaces: default is from 00h00 to 04h00 on Sunday morning
  - AutoStop WorkSpaces: automatically starts once a month to install updates
  - Manual maintenance: you define your windows and perform maintenance



# Amazon AppStream 2.0

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



# Amazon AppStream 2.0 vs WorkSpaces

- **Workspaces**

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

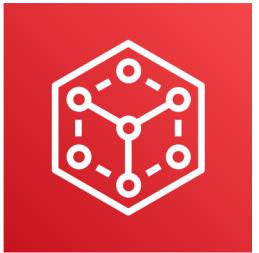
- **AppStream 2.0**

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

# Amazon Mechanical Turk

- Crowdsourcing marketplace to perform simple human tasks
- Distributed virtual workforce.
- Integrates with SWF natively, does not integrate with Step Functions
- Example:
  - You have a list of 10,000 restaurant names in your area and you want to get the telephone number, opening hours, address, etc...
  - Assume the restaurant name is not perfect, therefore Google API cannot help
  - You distribute the task on Mechanical Turk and **humans** will fill your database
- Other use cases: image classification, data collection, business processing

# AWS Device Farm



- Application testing service for your mobile and web applications
- Test across **real browsers** and **real mobiles devices**
- Fully automated using framework
- Improve the quality of web and mobile apps
- Generates videos and logs to document the issues encountered
- Can **remotely log-in** to devices for debugging

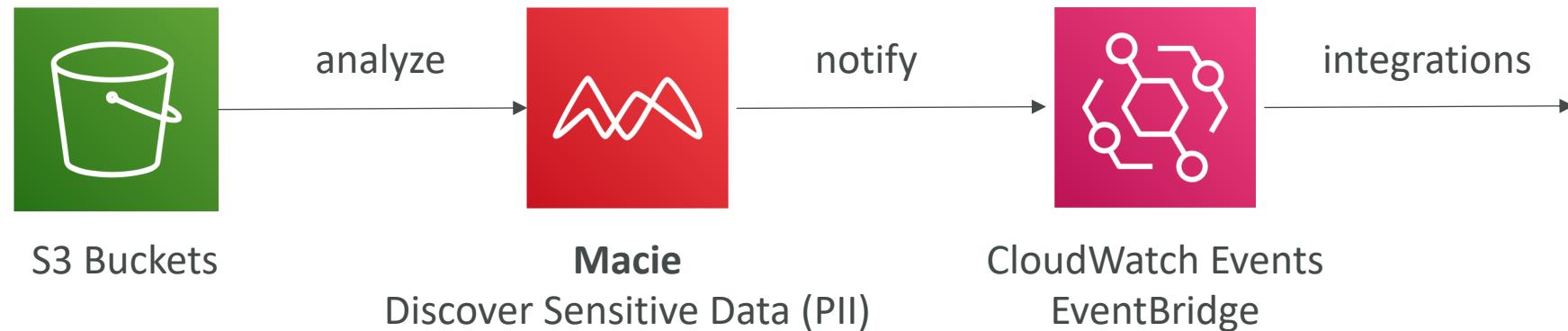
The screenshot shows the AWS Device Farm test results interface. At the top, a summary bar indicates 20 failures and 28 successes. Below it, the 'Unique problems' section highlights '1 UNIQUE FAILURE FOUND' related to a movie search timeout. The 'Devices' section lists three devices: HTC One M7 (AT&T), HTC One M8 (Sprint), and LG G Flex2 (Sprint), each with its own test results table.

Device	Test Results	Created
HTC One M7 (AT&T)	4.4.2 (green, yellow, red)	2015-07-07T15:06:0700
HTC One M8 (Sprint)	4.4.4 (green, yellow, red)	2015-07-07T15:06:0700
LG G Flex2 (Sprint)	5.0.1 (green, yellow, red)	2015-07-07T15:06:0700

# AWS Macie



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



# Amazon Transcribe



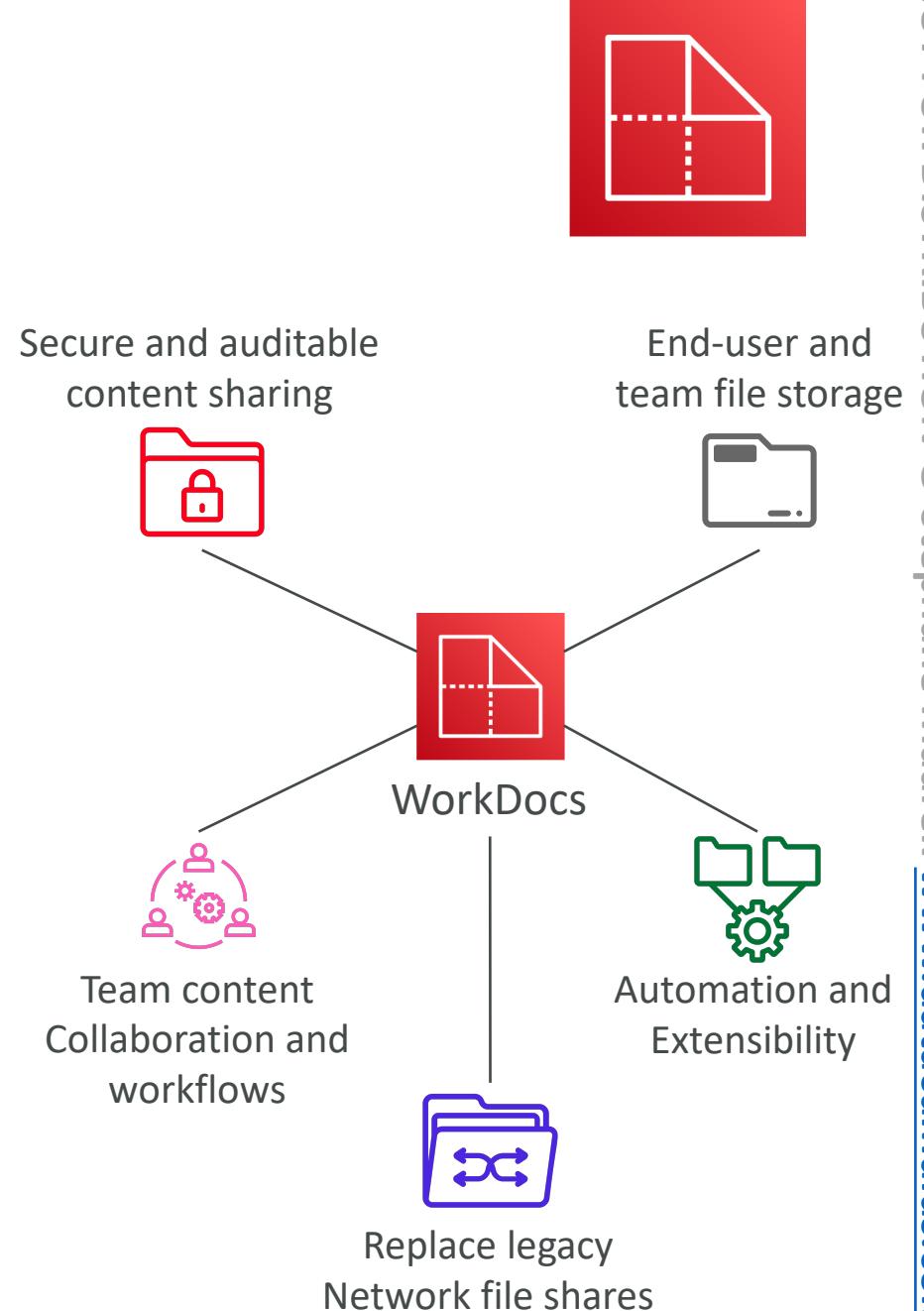
- Automatically convert speech to text
- Uses a **deep learning process** called **automatic speech recognition (ASR)** to convert speech to text quickly and accurately
- Use cases:
  - transcribe customer service calls
  - automate closed captioning and subtitling
  - generate metadata for media assets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



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

# Amazon WorkDocs

- Alternative to Google Drive / OneDrive / etc
- A secure content management system that can be accessed through API calls from external customer apps
- Historical versions of files is kept for easy rollback
- Create, store, sync, and share files from any location
- Content encrypted at rest and in-transit
- Approval workflow, reminders, notifications
- WorkDocs Drive – native desktop application



# Final Tips & Sample Questions

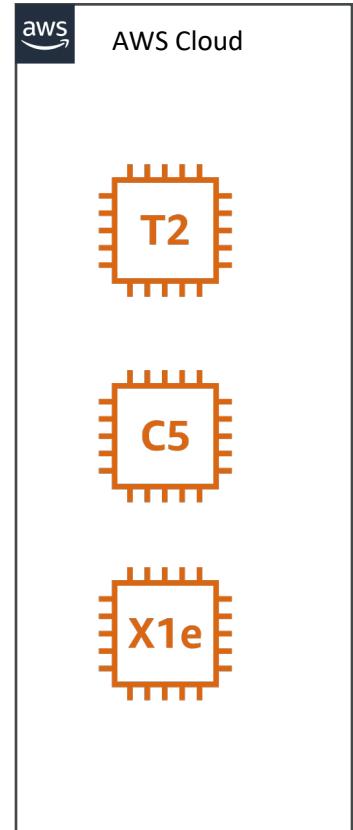
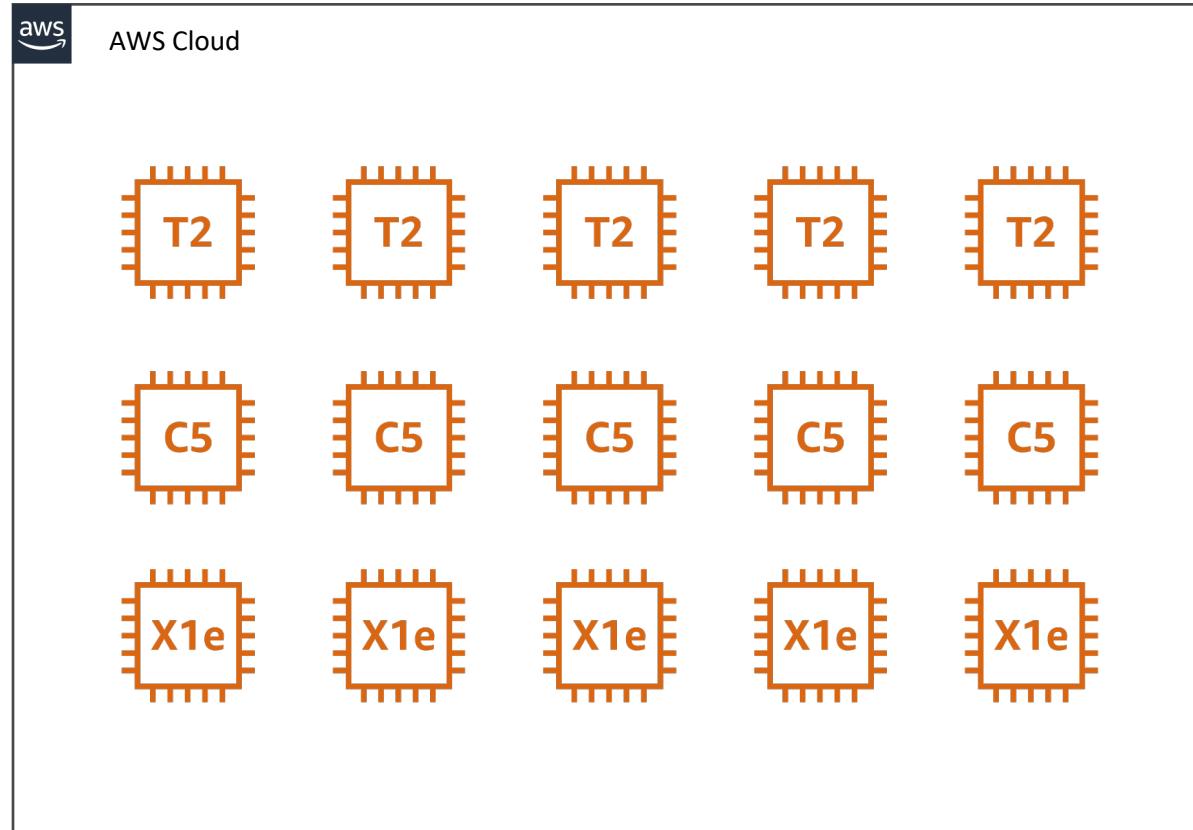
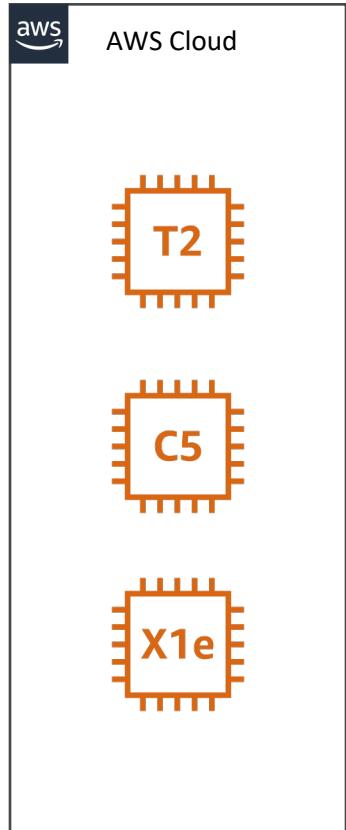
# Analysis from the Practice Sample questions

- Exam Page: <https://aws.amazon.com/certification/certified-solutions-architect-professional/>

# Question 1

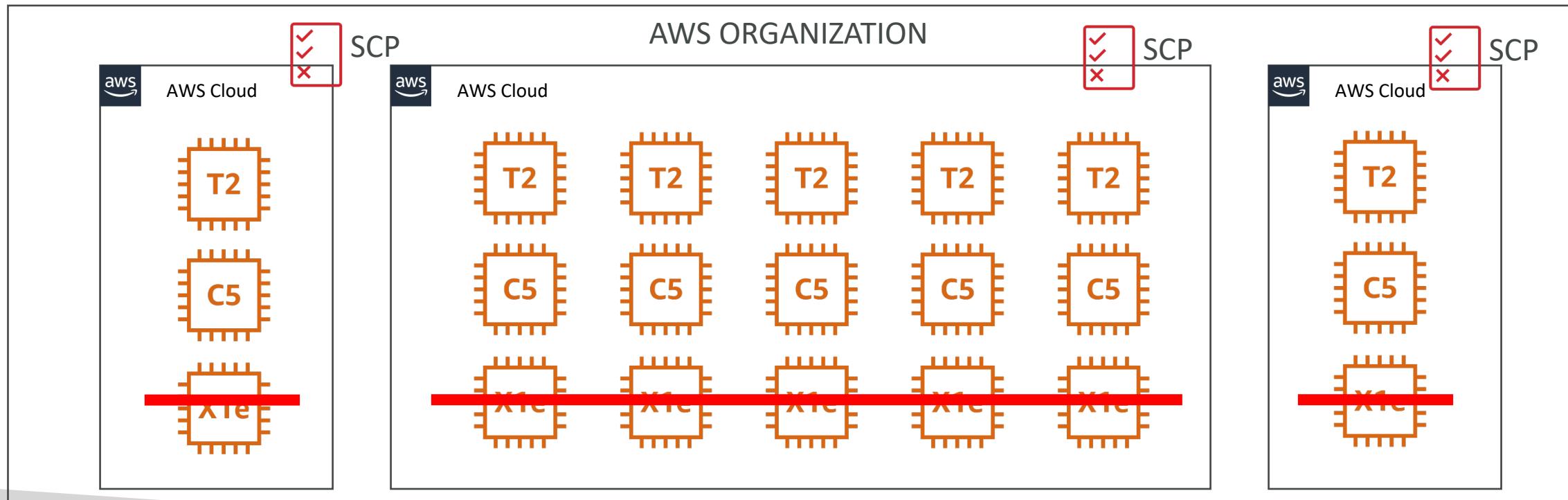
- An enterprise has a large number of AWS accounts owned by separate business groups. One of the accounts was recently compromised. The attacker launched a large number of instances, resulting in a high bill for that account.
- The security breach was addressed, but management has asked a solutions architect to develop a solution to **prevent excessive spending** in all accounts. **Each business group wants to retain full control over its AWS account.**
- Which solution should the solutions architect recommend to meet these requirements?

# Question I – Architecture Diagram



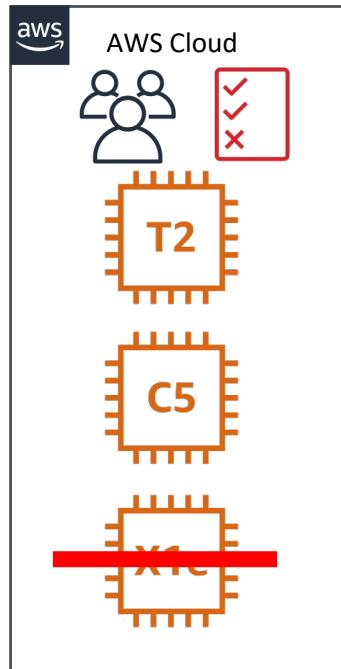
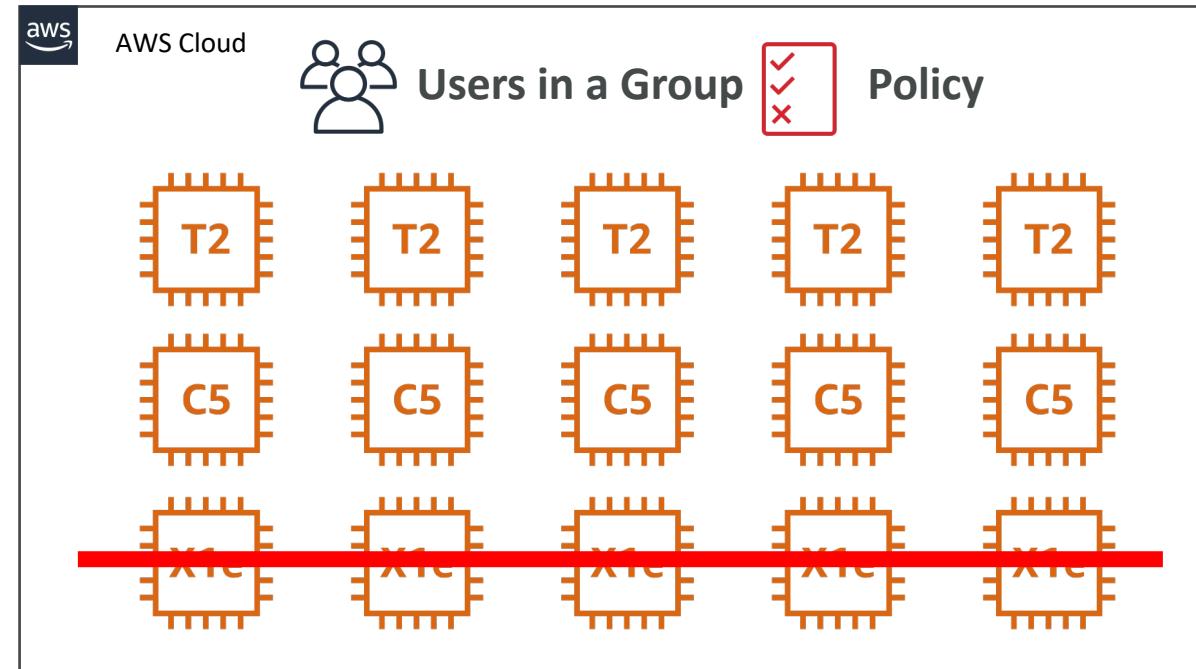
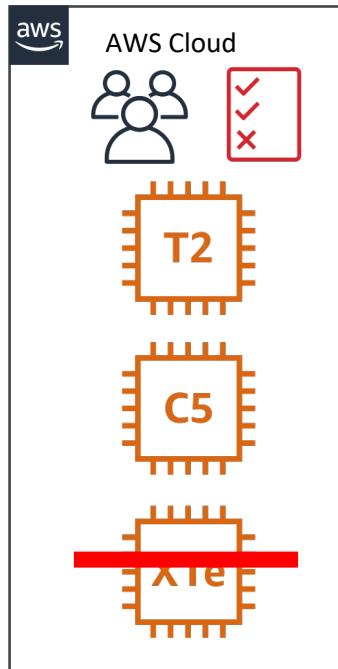
# Option A

- Use AWS Organizations to add each AWS account to the master account. Create a service control policy (SCP) that uses the `ec2:instanceType` condition key to prevent the launch of high-cost instance types in each account.



# Option B

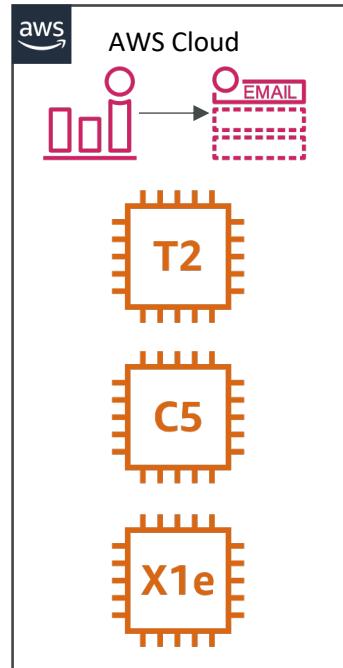
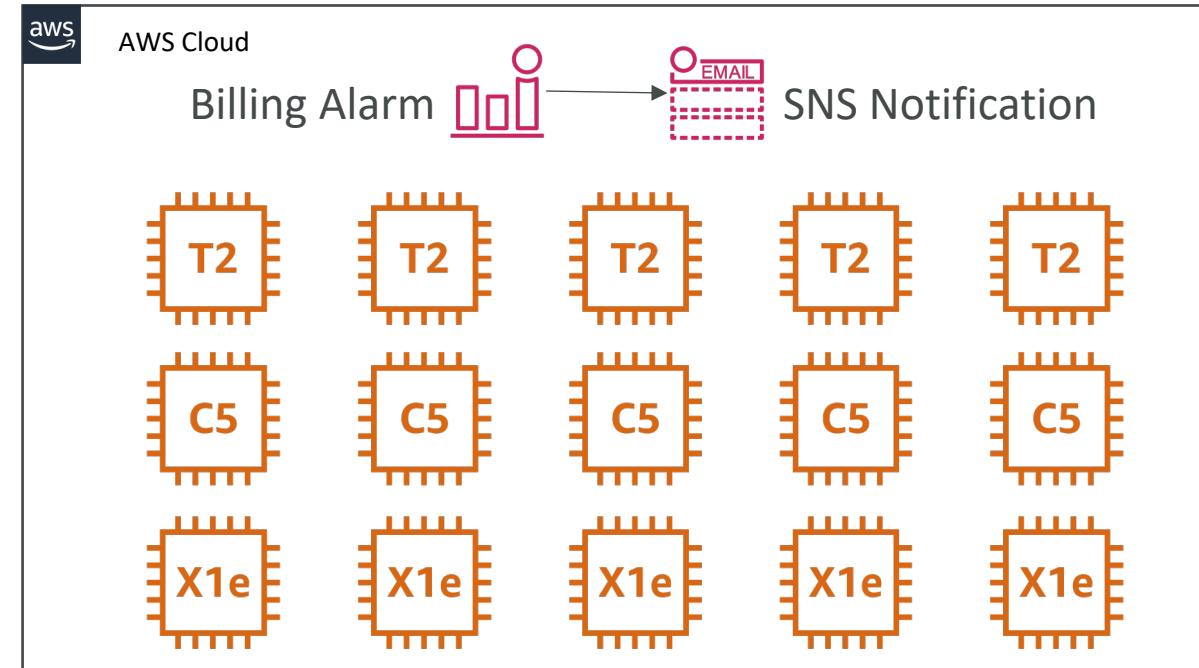
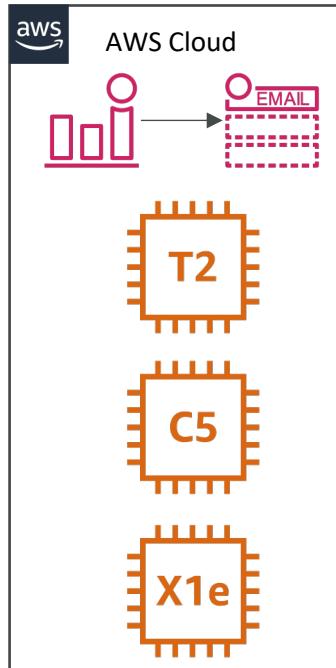
- Attach a new customer-managed IAM policy to an IAM group in each account that uses the `ec2:instanceType` condition key to prevent the launch of high-cost instance types. Place all of the existing IAM users in each group.



**CORRECT ANSWER**

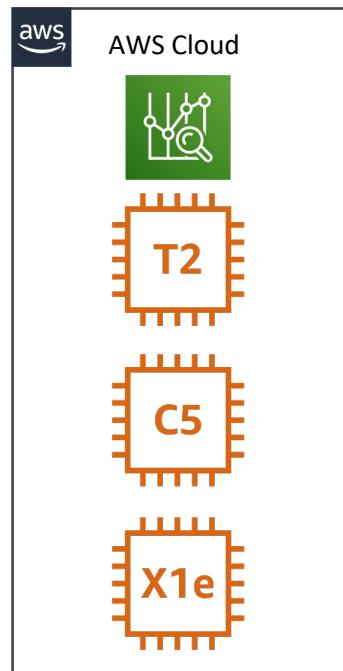
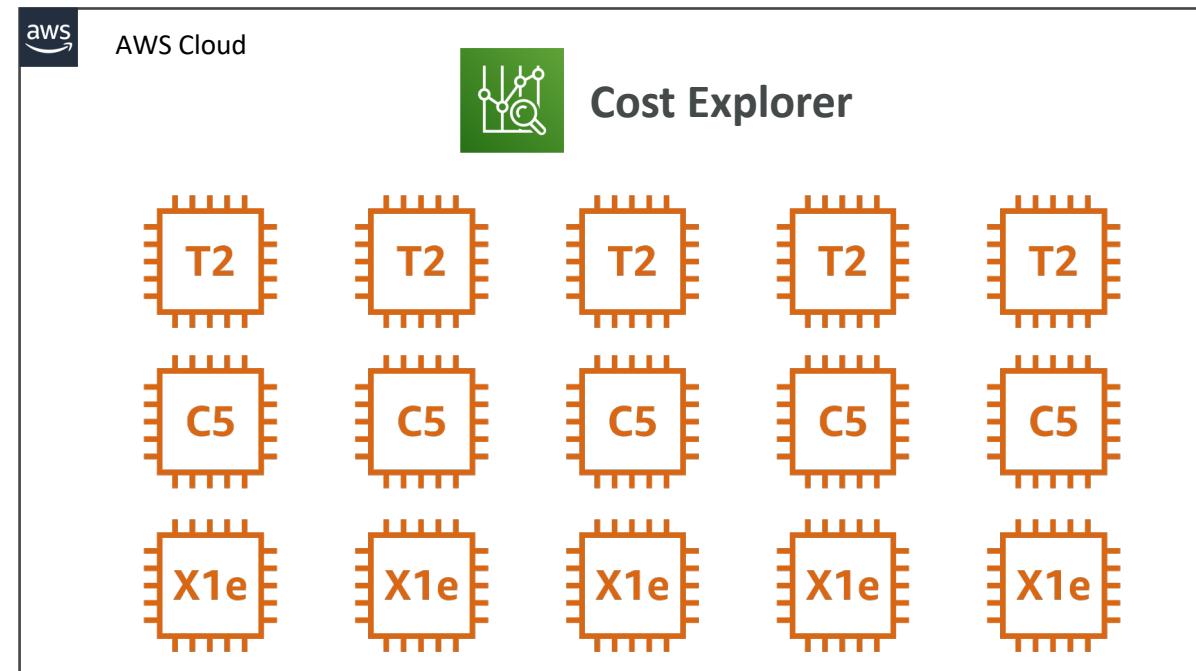
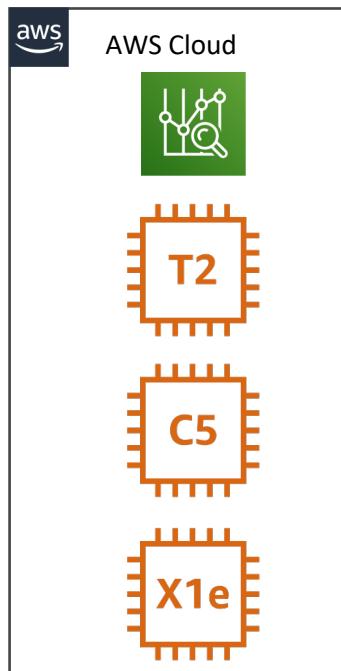
# Option C

- Enable billing alerts on each AWS account. Create Amazon CloudWatch alarms that send an Amazon SNS notification to the account administrator whenever their account exceeds the spending budget.



# Option D

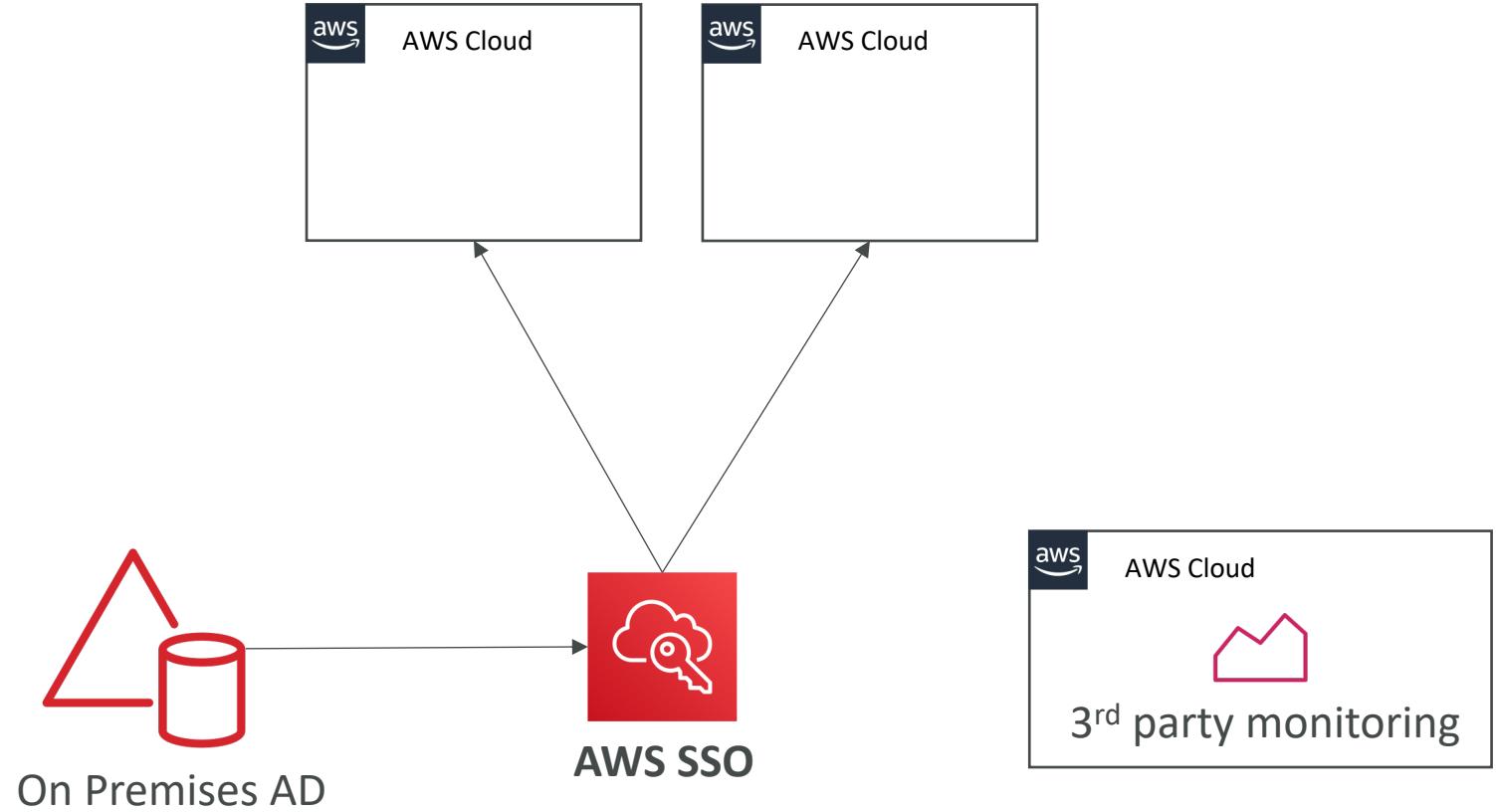
- Enable AWS Cost Explorer in each account. Regularly review the Cost Explorer reports for each account to ensure spending does not exceed the planned budget



# Question 2

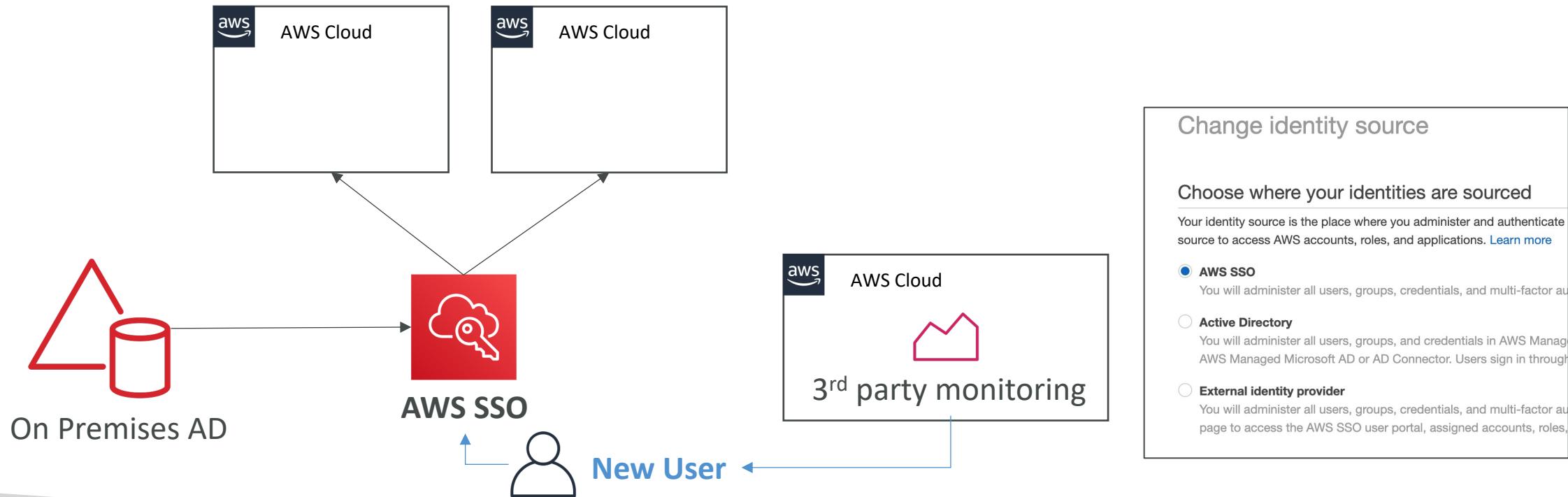
- A company has multiple AWS accounts. The company has integrated its on-premises Active Directory (AD) with AWS SSO to grant AD users least privilege abilities to manage infrastructure across all the accounts.
- A solutions architect must integrate a third-party monitoring solution that requires **read-only** access across all AWS accounts. The monitoring solutions will run **in its own AWS account**.
- How can the monitoring solution be given the required permissions?

# Question 2 - Architecture



# Option A

- Create a user in an AWS SSO directory and assign a read-only permissions set. Assign all AWS accounts to be monitored to the new user. Provide the third-party monitoring solution with the user name and password.

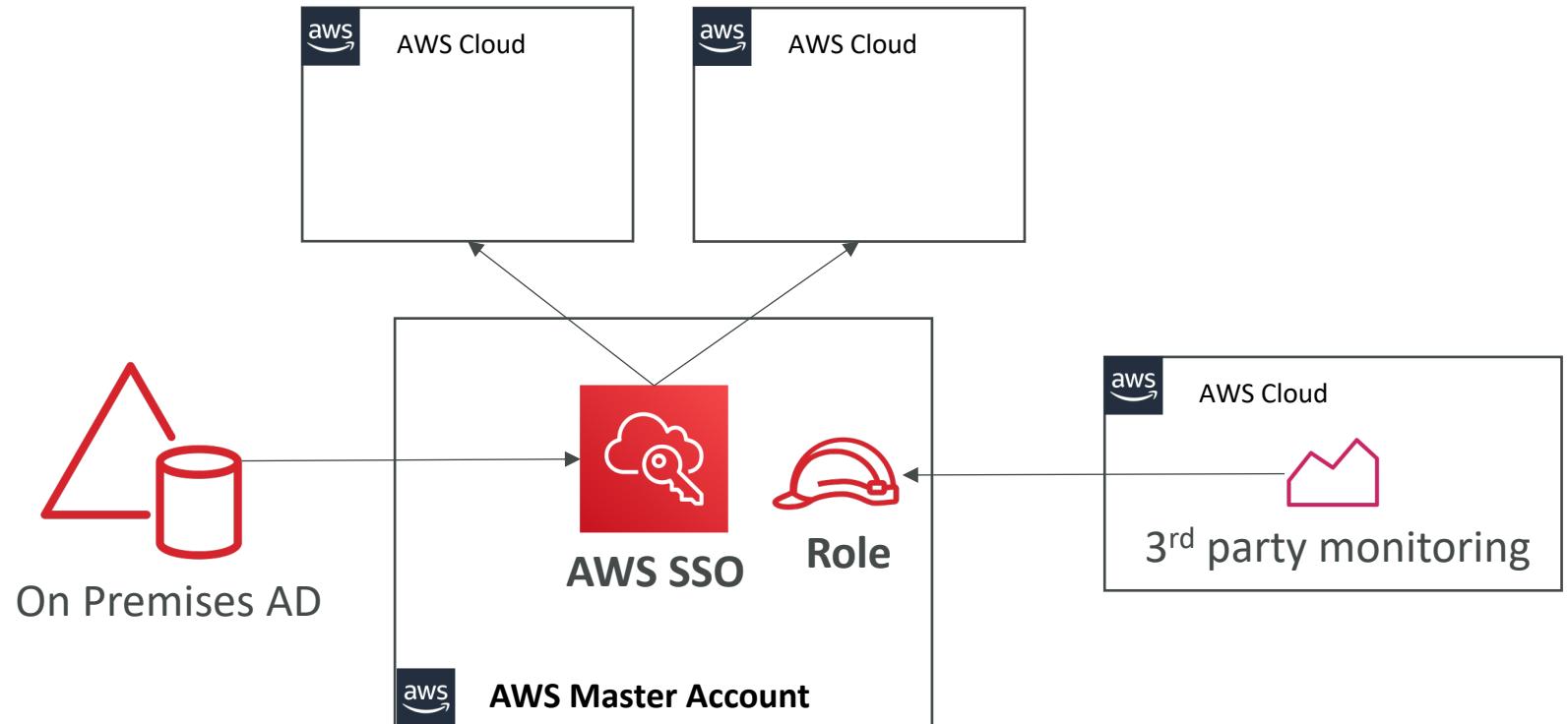


# Note on option A

- Currently, the sample question PDF says:
- “A is incorrect because credentials supplied by AWS SSO are temporary, so the application would lose permissions and have to re-login”
- That is wrong.
- Users created in AWS SSO have a password that doesn't change and must respect the password policy defined.
- Here Option A is wrong because you can't have both users defined in AWS SSO and in Active Directory. SSO only allows for one Identity source (SSO, AD or IdP).

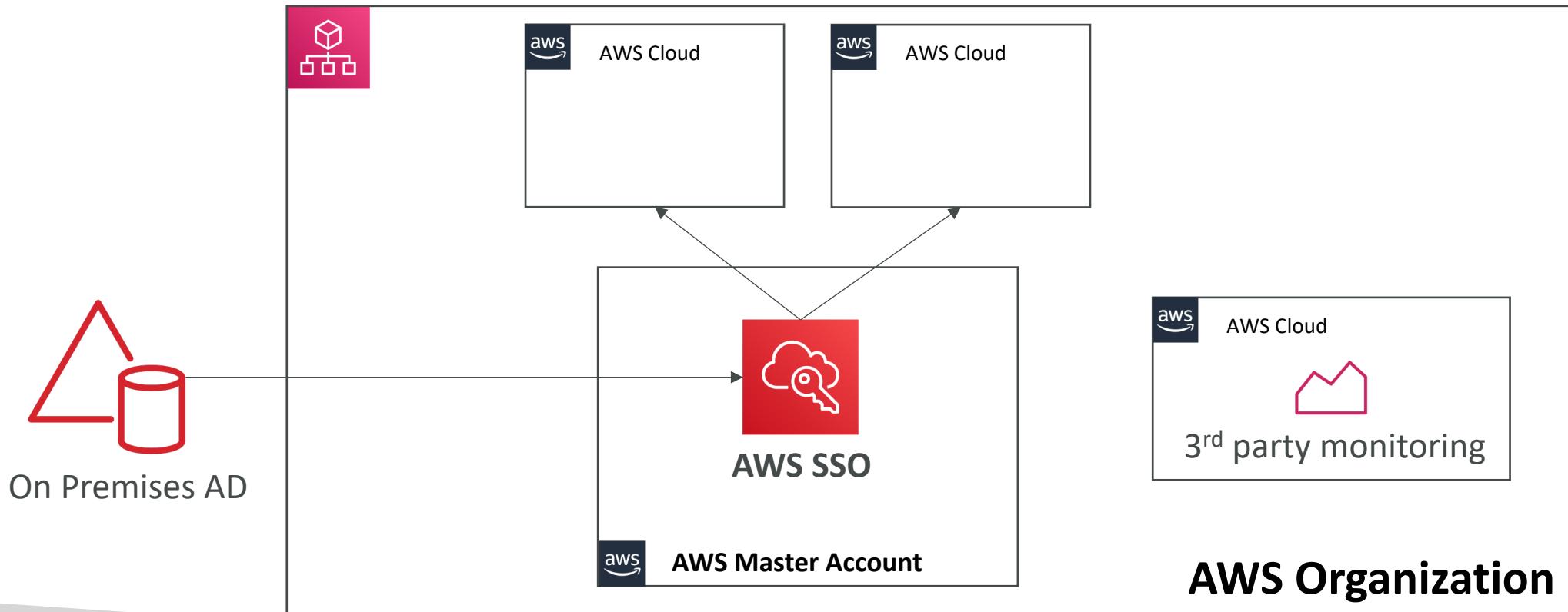
# Option B

- Create an AWS IAM role in the organization's master account. Allow the AWS account of the third-party monitoring solution to assume the role.



# Option C

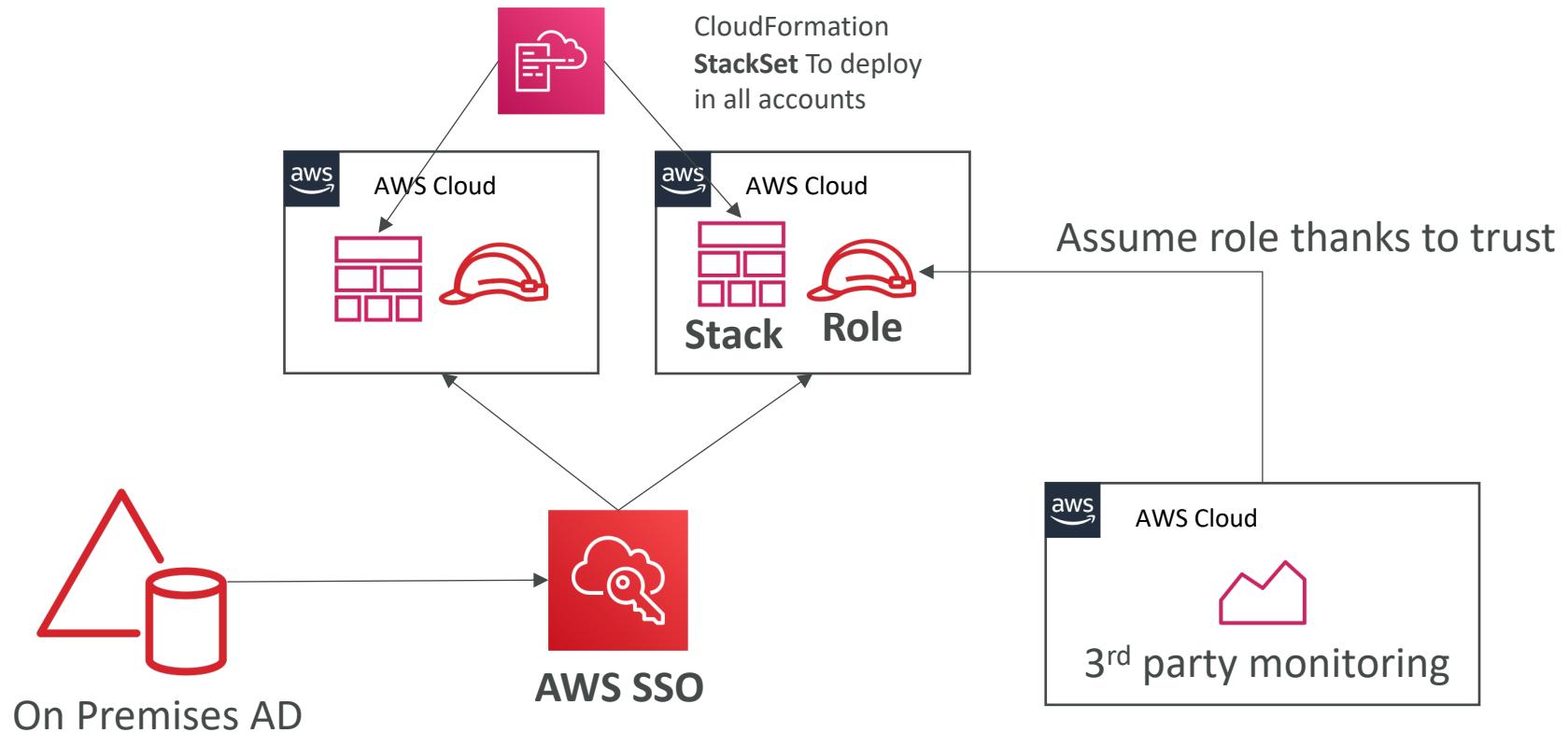
- Invite the AWS account of the third-party monitoring solution to join the organization. Enable all features



**CORRECT ANSWER**

# Option D

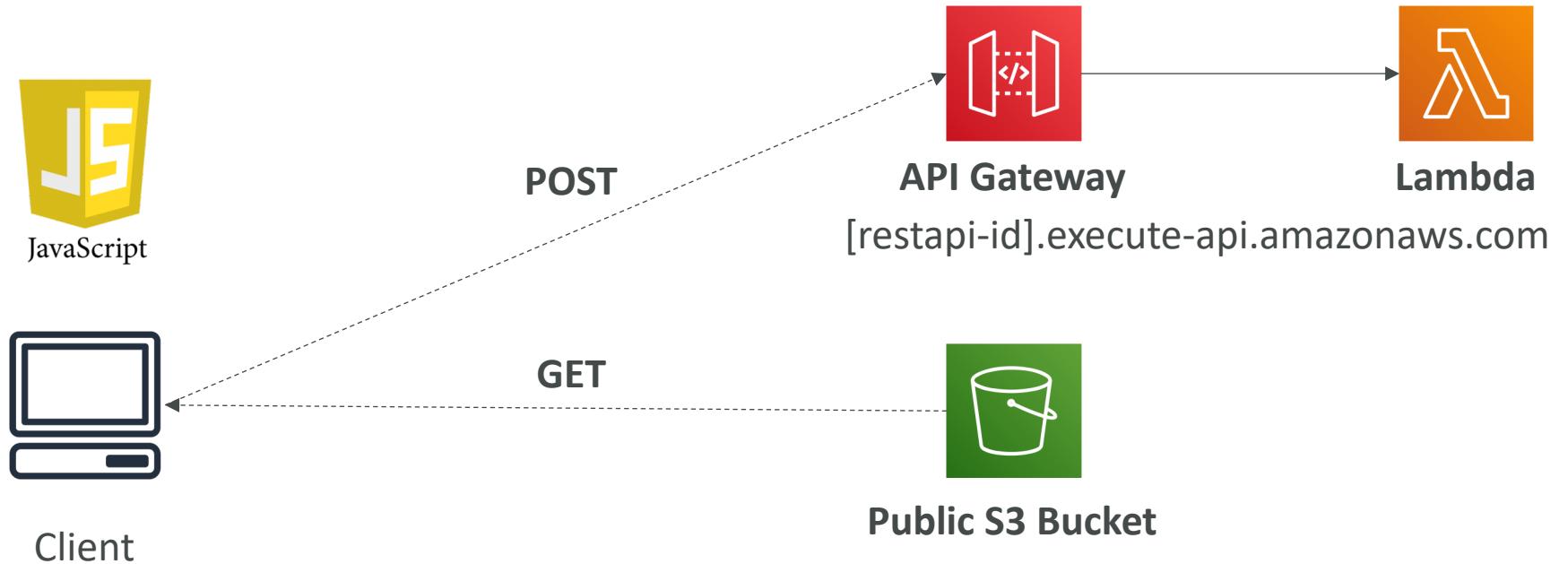
- Create an AWS CloudFormation template that defines a new AWS IAM role for the third-party monitoring solution with the account of the third party listed in the trust policy. Create the IAM role across all linked AWS accounts by using a stack set.



# Question 3

- A team is building an **HTML** form hosted in a public Amazon S3 bucket. The form uses **JavaScript** to post data to an Amazon API Gateway endpoint. The endpoint is integrated with AWS Lambda functions. The team has tested each method in the API Gateway console and received valid responses.
- Which combination of steps must be completed for the form **to successfully post to the API Gateway** and receive a valid response? (Select **TWO**.)

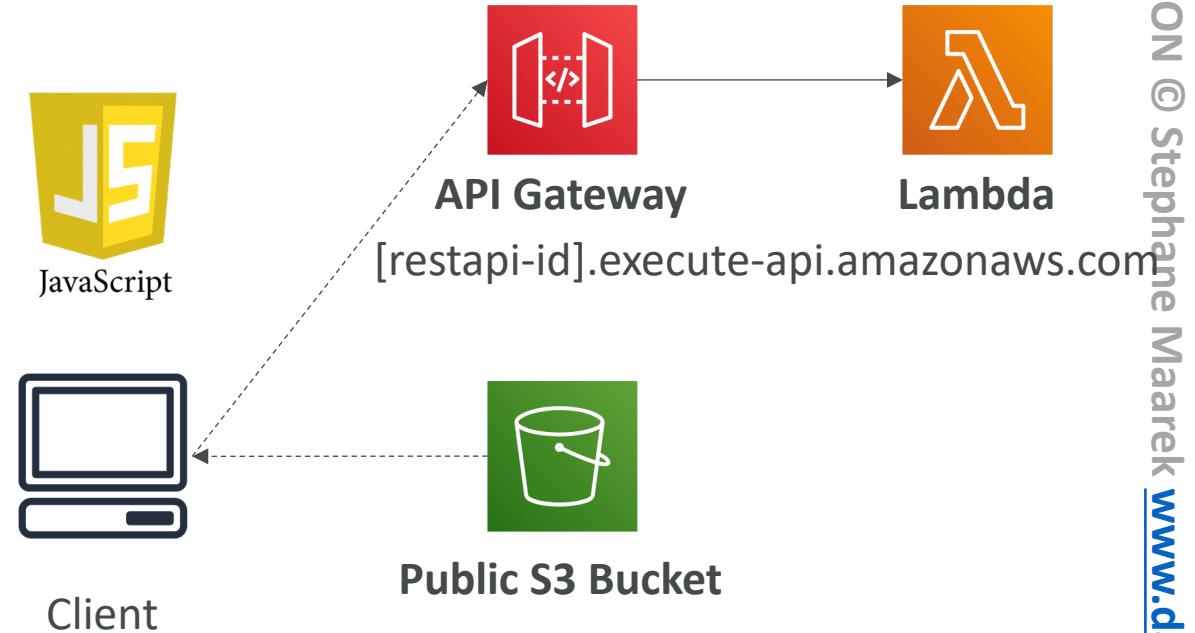
# Question 3 – Architecture



**CORRECT ANSWERS D,E**

# Options

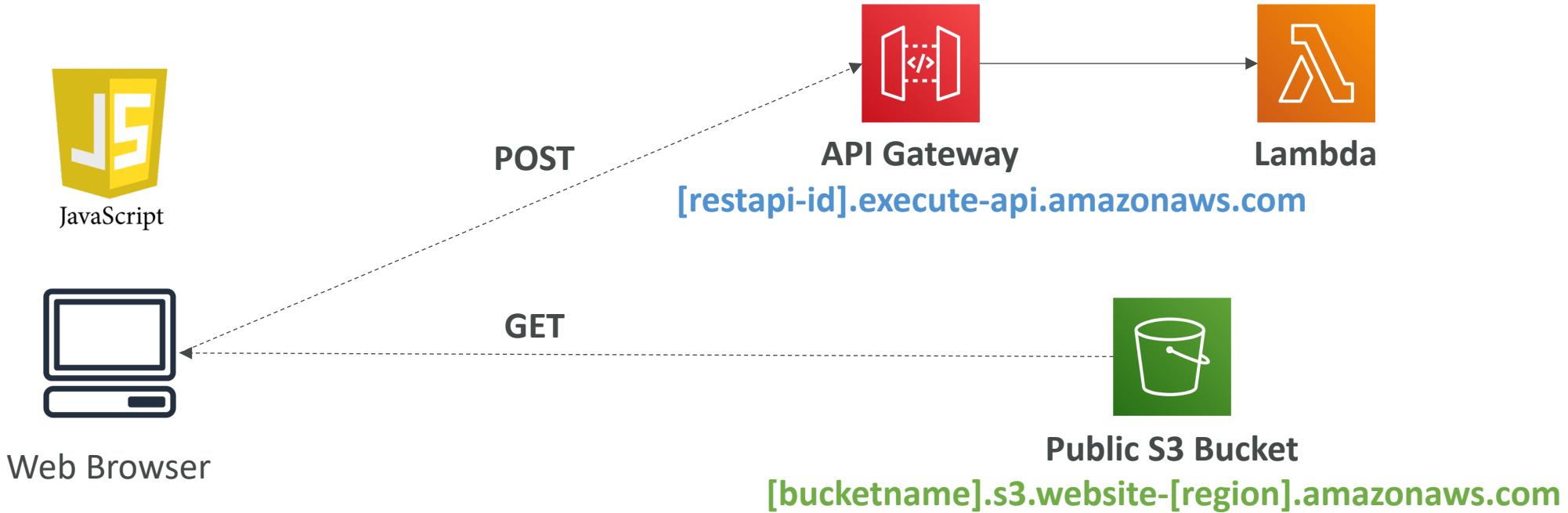
- A) Configure the S3 bucket to allow cross-origin resource sharing (CORS).
- B) Host the form on Amazon EC2 rather than Amazon S3.
- C) Request a limit increase for API Gateway.
- D) Enable cross-origin resource sharing (CORS) in API Gateway.
- E) Configure the S3 bucket for web hosting.



# Question 3 – Final Architecture

## CORS is a Browser based security

CORS to allow calls with Origin **[bucketname].s3.website-[region].amazonaws.com**  
Using the header **Access-Control-Allow-Origin**

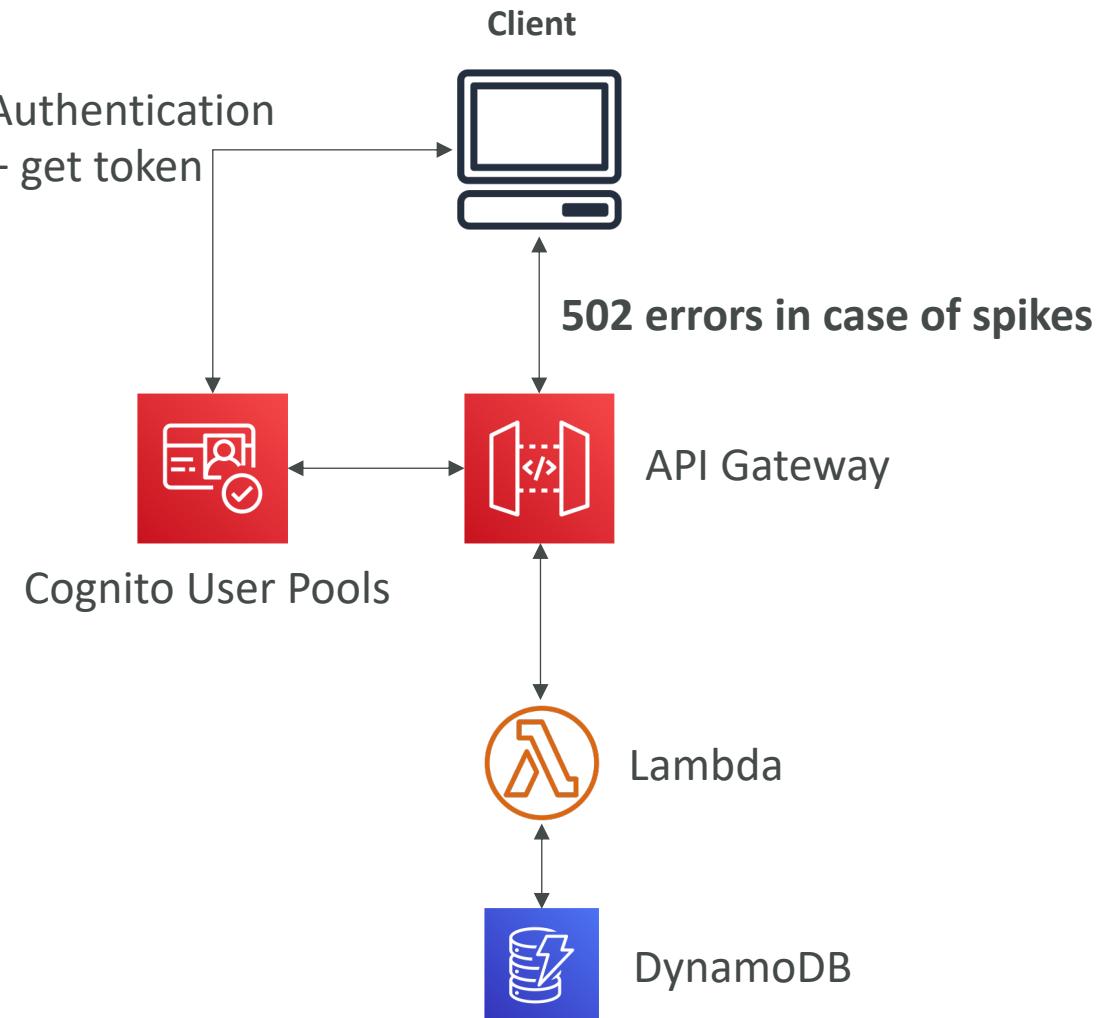


Visits **[bucketname].s3.website-[region].amazonaws.com**  
Makes API calls to **[restapi-id].execute-api.amazonaws.com**  
With **Origin: [bucketname].s3.website-[region].amazonaws.com**

# Question 4

- A retail company runs a serverless mobile app built on Amazon API Gateway, AWS Lambda, Amazon Cognito, and Amazon DynamoDB. During **heavy holiday traffic spikes**, the company receives complaints of **intermittent system failures**. Developers find that the API Gateway endpoint is returning **502 Bad Gateway errors** to seemingly valid requests.
- Which method should address this issue?

# Question 4 – Architecture



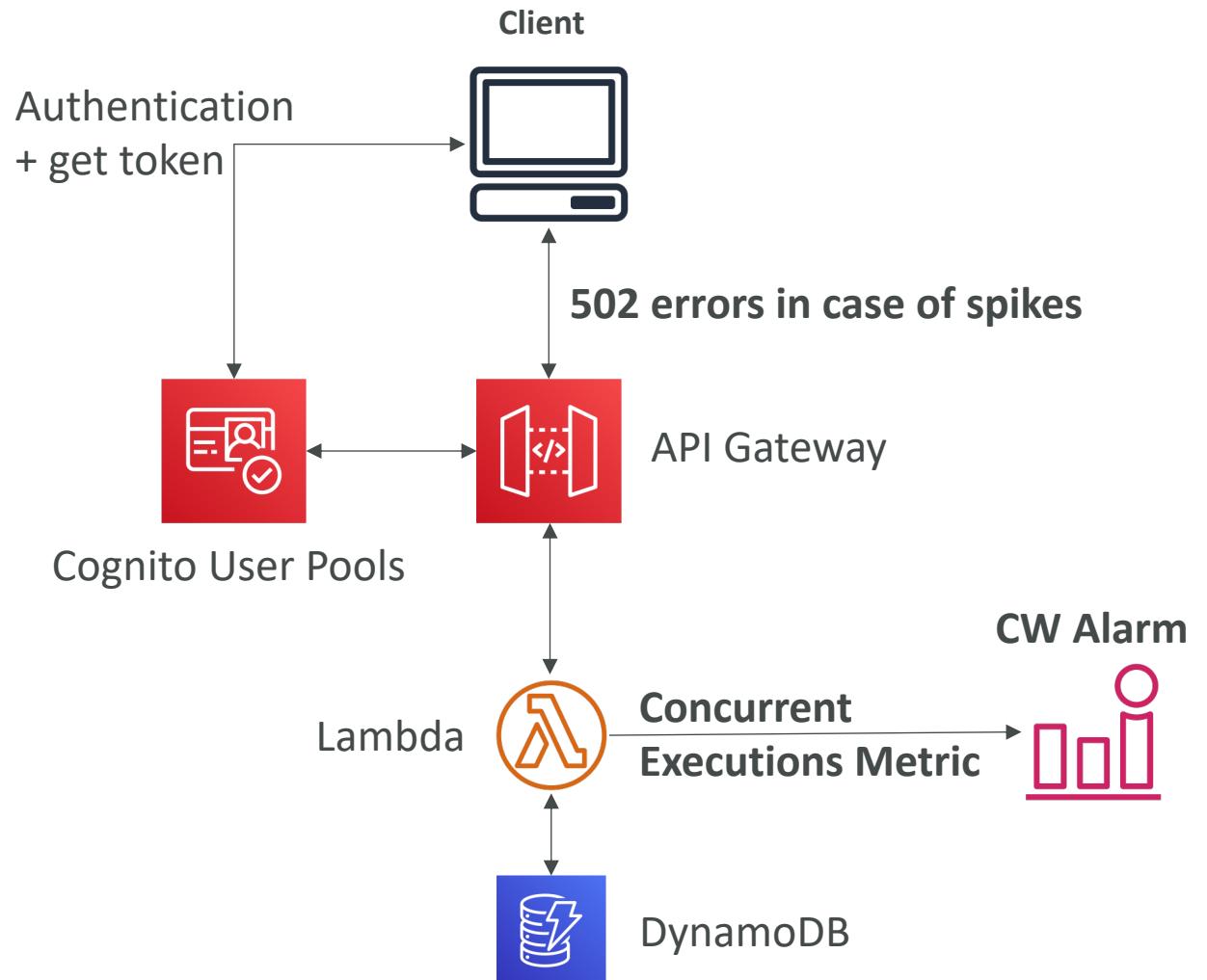
# API Gateway - Errors

- 4xx means Client errors
  - 400: Bad Request
  - 403: Access Denied, WAF filtered
  - 429: Quota exceeded, Throttle
- 5xx means Server errors
  - 502: Bad Gateway Exception, usually for an incompatible output returned from a Lambda proxy integration backend and occasionally for out-of-order invocations due to heavy loads.
  - 503: Service Unavailable Exception
  - 504: Integration Failure – ex Endpoint Request Timed-out Exception  
API Gateway requests time out after 29 second maximum

**CORRECT ANSWER**

# Option A

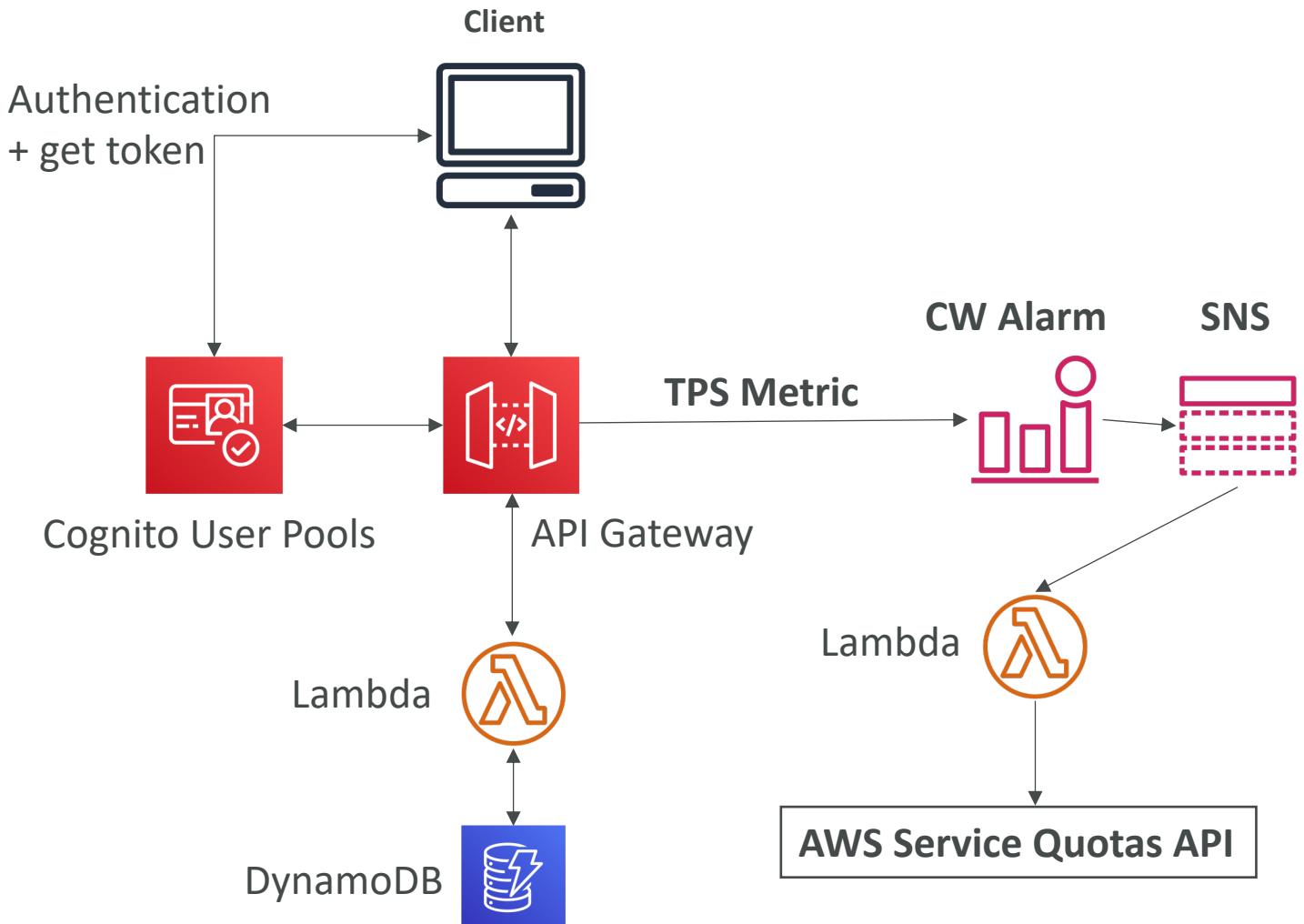
- Increase the concurrency limit for Lambda functions and configure notification alerts to be sent by Amazon CloudWatch when the **ConcurrentExecutions** metric approaches the limit.



# Option B

- Configure notification alerts for the limit of transactions per second on the API Gateway endpoint and create a Lambda function that will increase this limit, as needed.

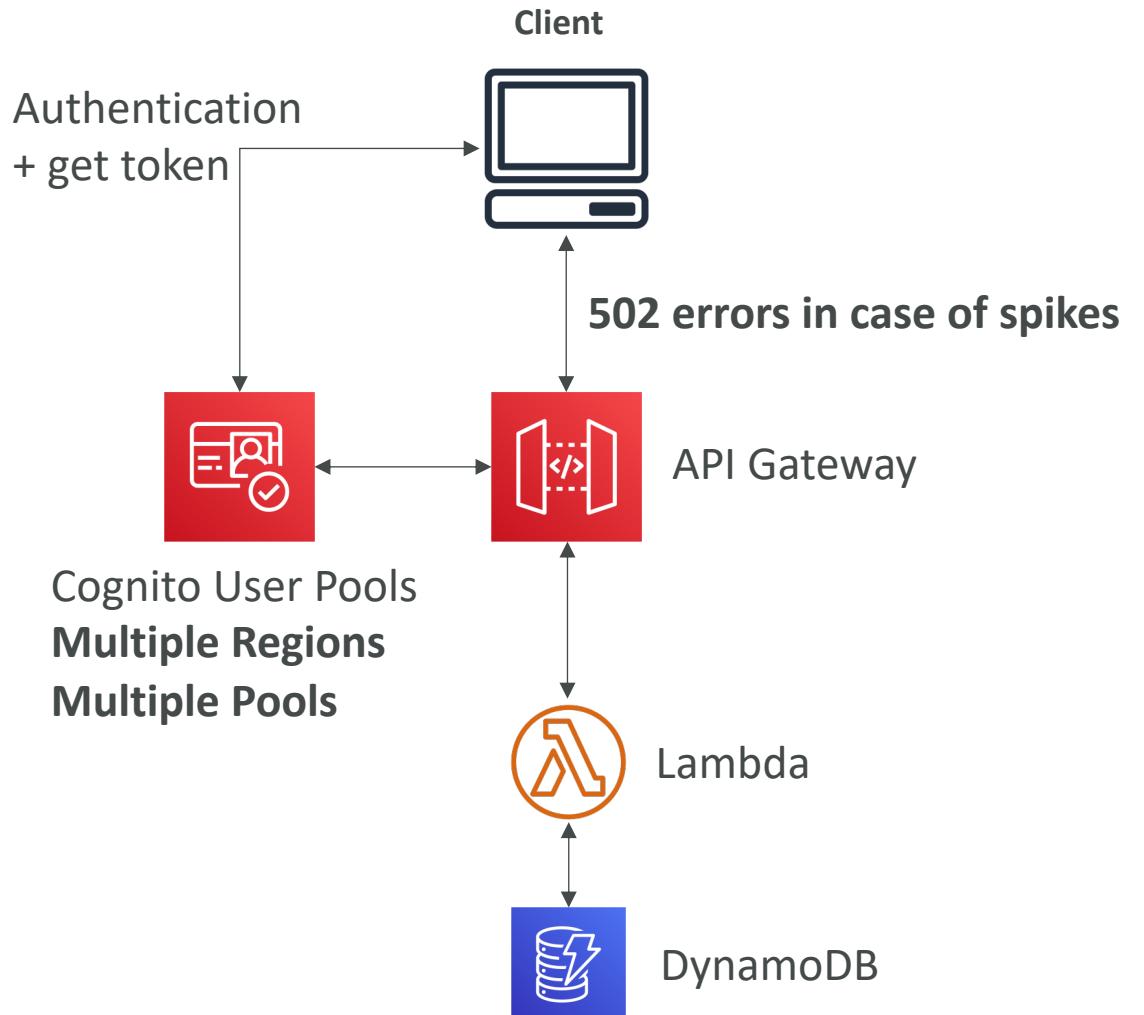
**Option B would work if we were receiving 429 errors  
429: Quota exceeded, Throttle**



# Option C

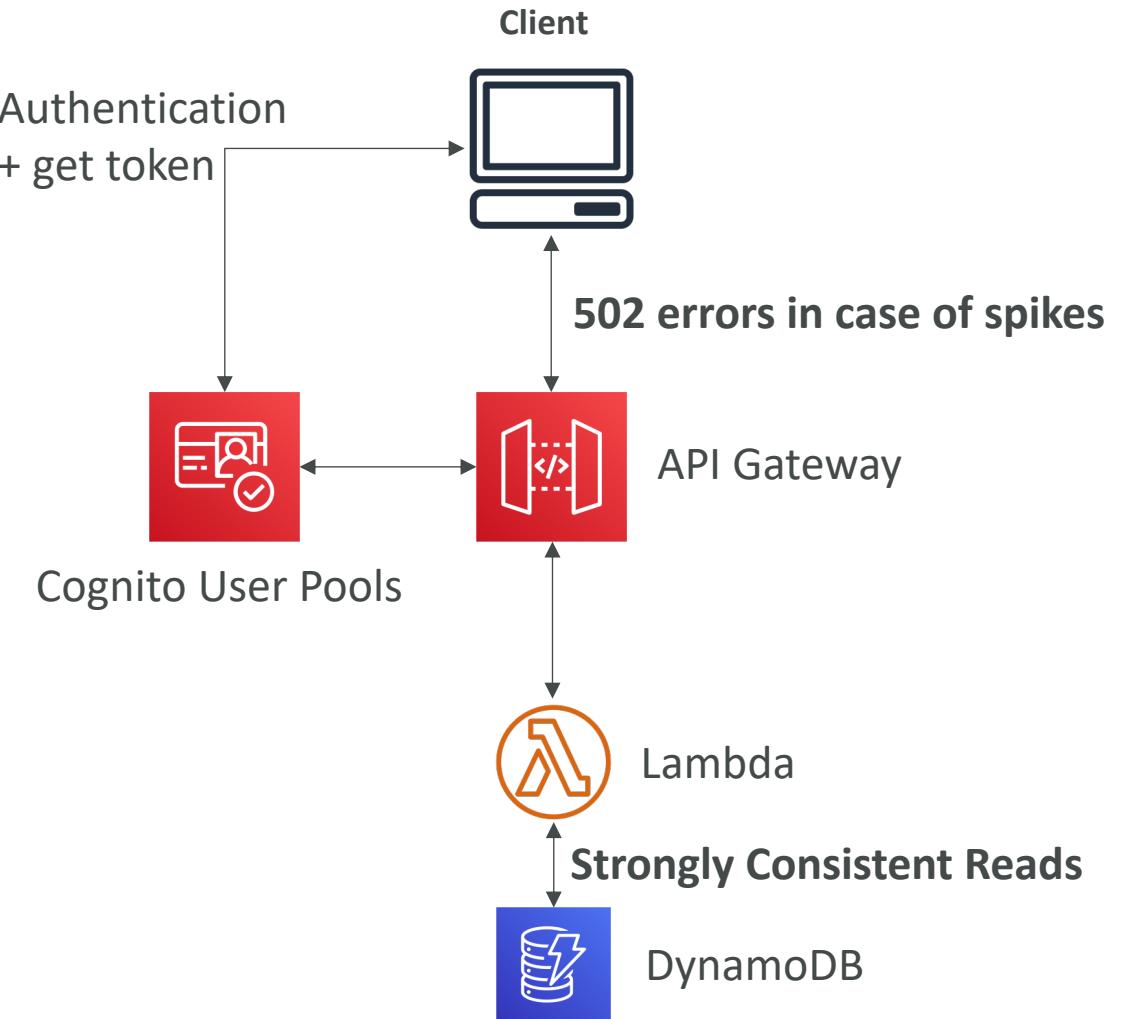
- Shard users to Amazon Cognito user pools in multiple regions to reduce user authentication latency.

Option C would be valid if we had performance issues At Cognito User Pools, but that's not the question



# Option D

- Use DynamoDB strongly consistent reads to ensure the latest data is always returned to the client application.

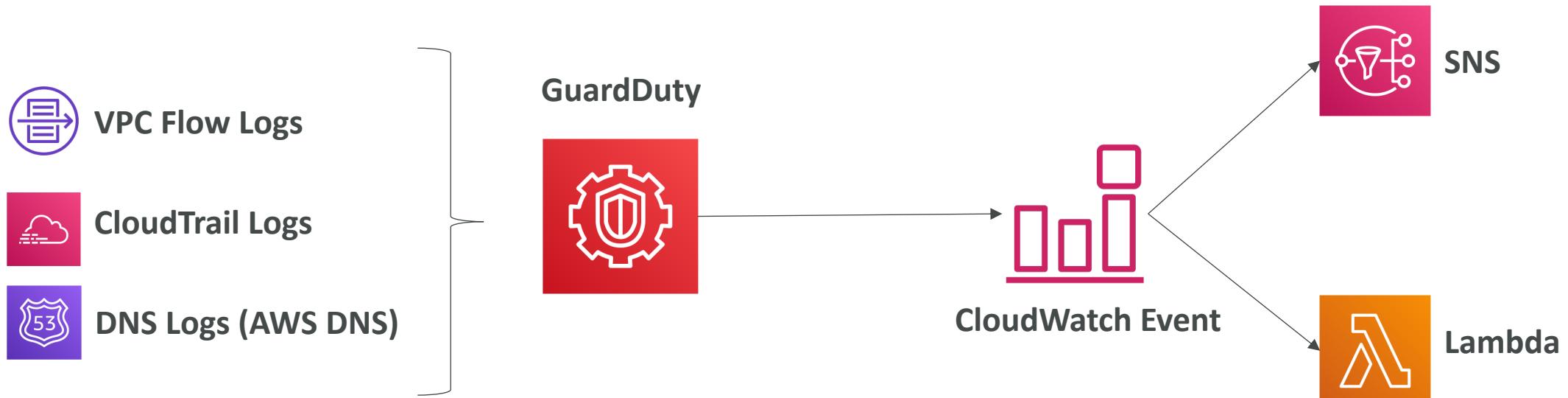


# Question 5 – Architecture

- A web hosting company has enabled Amazon GuardDuty in every AWS Region for all of its accounts. A system administrator must create an automated response to high-severity events.
- How should this be accomplished?

# Options

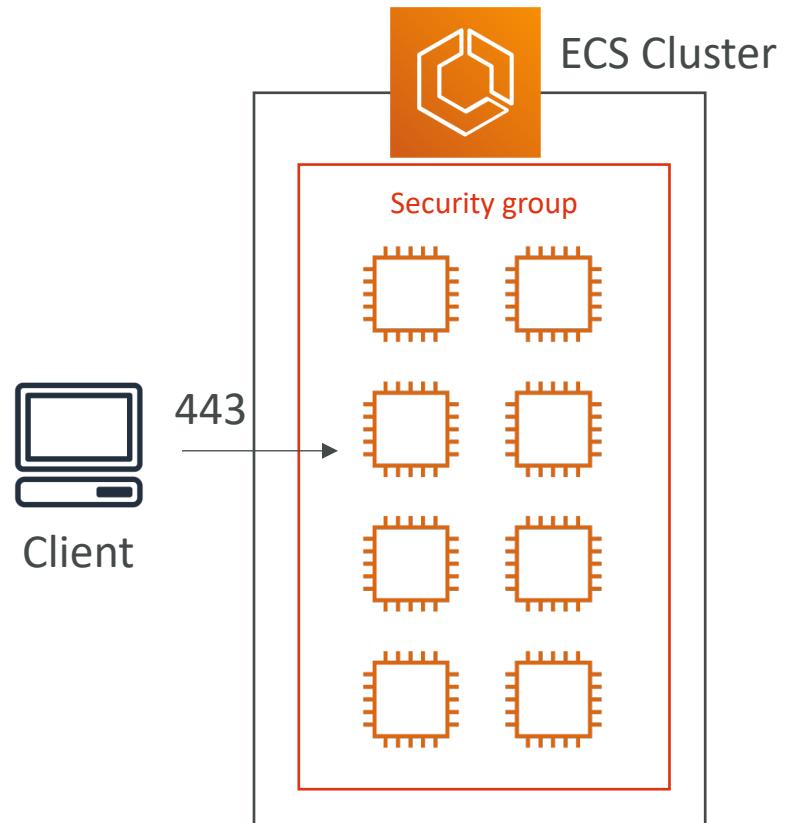
- A) Create rules through VPC Flow Logs that trigger an AWS Lambda function that programmatically addresses the issue.
- B) Create an AWS CloudWatch Events rule that triggers an AWS Lambda function that programmatically addresses the issue.
- C) Configure AWS Trusted Advisor to trigger an AWS Lambda function that programmatically addresses the issue.
- D) Configure AWS CloudTrail to trigger an AWS Lambda function that programmatically addresses the issue.



# Question 6

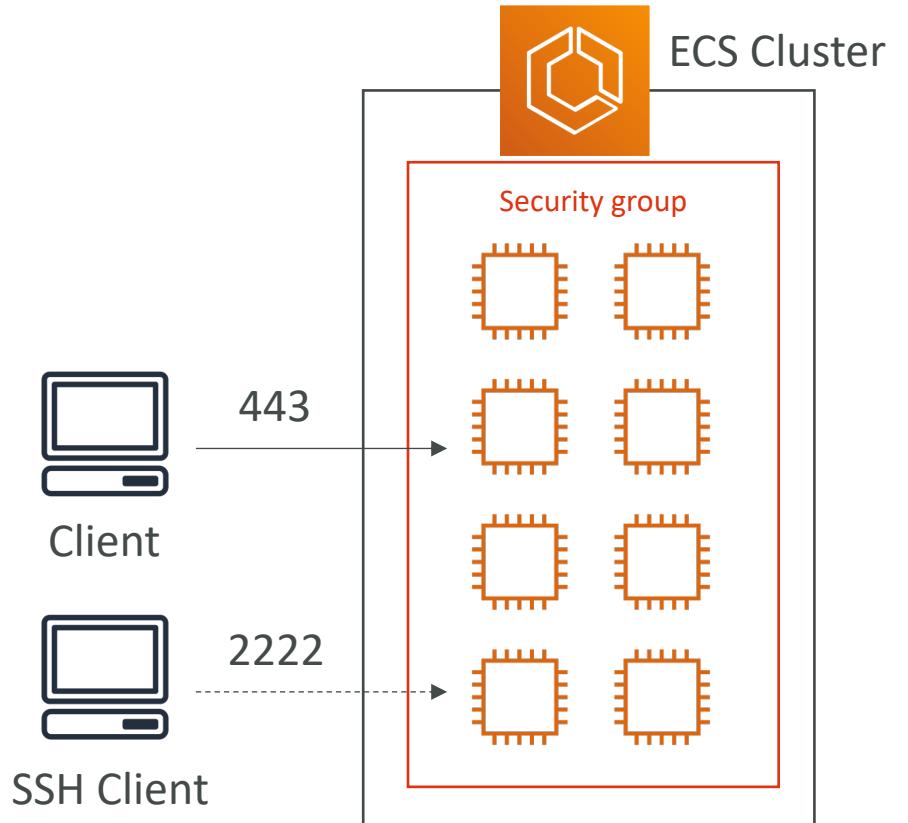
- A company is launching a new web service on an Amazon ECS cluster. Company policy requires that the security group on the cluster instances block all inbound traffic but HTTPS (port 443). The cluster consists of Amazon 100 EC2 instances. Security engineers are responsible for managing and updating the cluster instances. The security engineering team is small, so any management efforts must be minimized.
- How can the service be designed to meet these operational requirements?

# Question 6 – Architecture



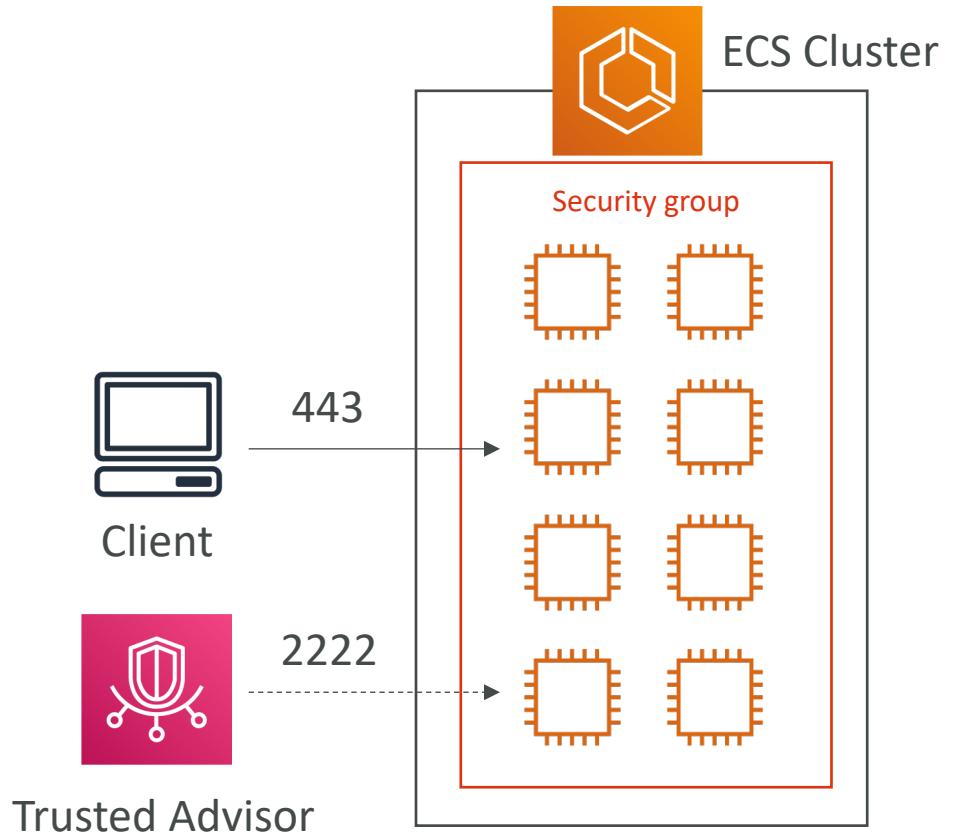
# Option A

- Change the SSH port to 2222 on the cluster instances with a user data script. Log in to each instance using SSH over port 2222



# Option B

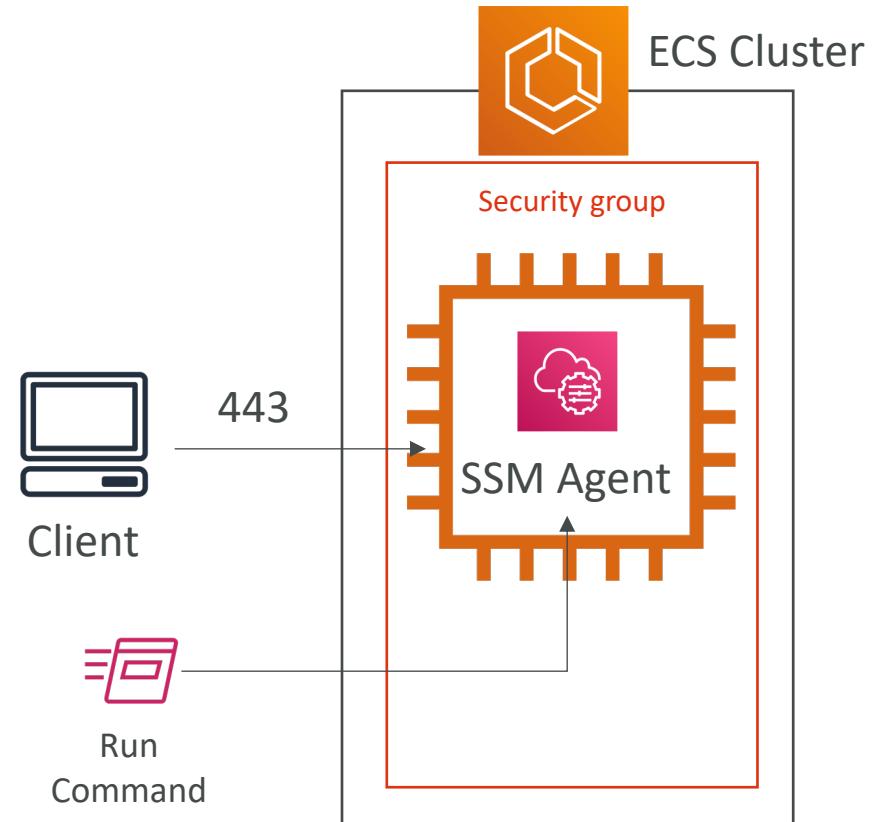
- Change the SSH port to 2222 on the cluster instances with a user data script. Use AWS Trusted Advisor to remotely manage the cluster instances over port 2222



**CORRECT ANSWER**

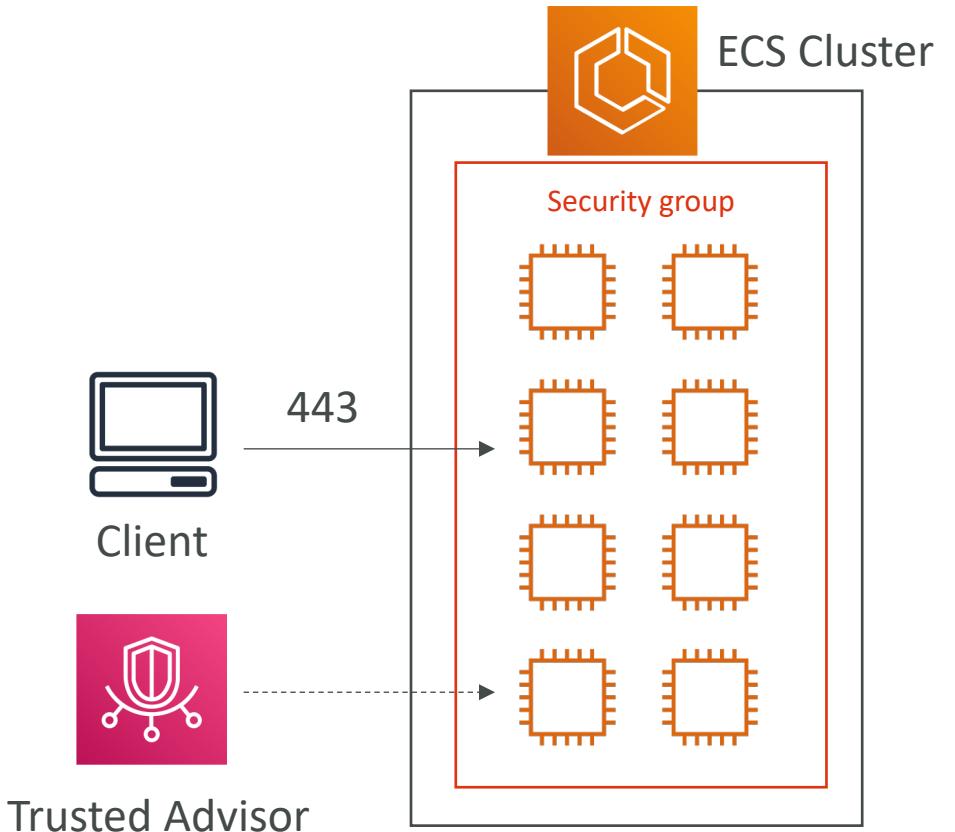
# Option C

- Launch the cluster instances with no SSH key pairs. Use the Amazon EC2 Systems Manager Run Command to remotely manage the cluster instances



# Option D

- Launch the cluster instances with no SSH key pairs. Use AWS Trusted Advisor to remotely manage the cluster instances.



# Question 7

- A company has two AWS accounts: one for production workloads and one for development workloads. Creating and managing these workloads are a development team and an operations team. The company needs a security strategy that meets the following requirements:
  - Developers need to create and delete development application infrastructure.
  - Operators need to create and delete both development and production application infrastructure.
  - Developers should have no access to production infrastructure.
  - All users should have a single set of AWS credentials.
- What strategy meets these requirements?

# Option A

- In the development account:
  - Create a development IAM group with the ability to create and delete application infrastructure.
  - Create an IAM user for each operator and developer and assign them to the development group.
- In the production account:
  - Create an operations IAM group with the ability to create and delete application infrastructure.
  - Create an IAM user for each operator and assign them to the operations group.

# Option B

- In the development account:
  - Create a development IAM group with the ability to create and delete application infrastructure.
  - Create an IAM user for each developer and assign them to the development group.
  - Create an IAM user for each operator and assign them to the development group and the operations group in the production account.
- In the production account:
  - Create an operations IAM group with the ability to create and delete application infrastructure.

# Option C

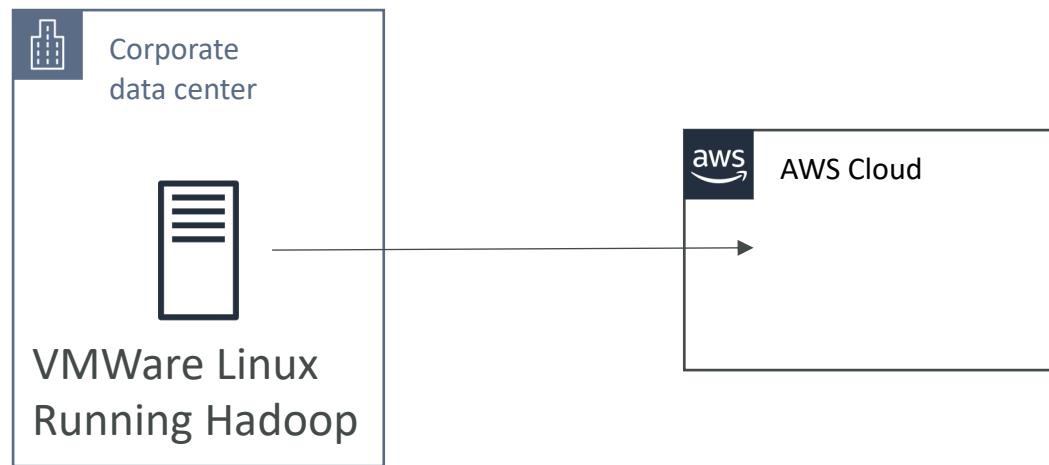
- In the development account:
  - Create a shared IAM role with the ability to create and delete application infrastructure in the production account.
  - Create a development IAM group with the ability to create and delete application infrastructure.
  - Create an operations IAM group with the ability to assume the shared role.
  - Create an IAM user for each developer and assign them to the development group.
  - Create an IAM user for each operator and assign them to the development group and the operations group.

# Option D

- In the development account:
  - Create a development IAM group with the ability to create and delete application infrastructure.
  - Create an operations IAM group with the ability to assume the shared role in the production account.
  - Create an IAM user for each developer and assign them to the development group.
  - Create an IAM user for each operator and assign them to the development group and the operations group.
- In the production account:
  - Create a shared IAM role with the ability to create and delete application infrastructure.
  - Add the development account to the trust policy for the shared role.

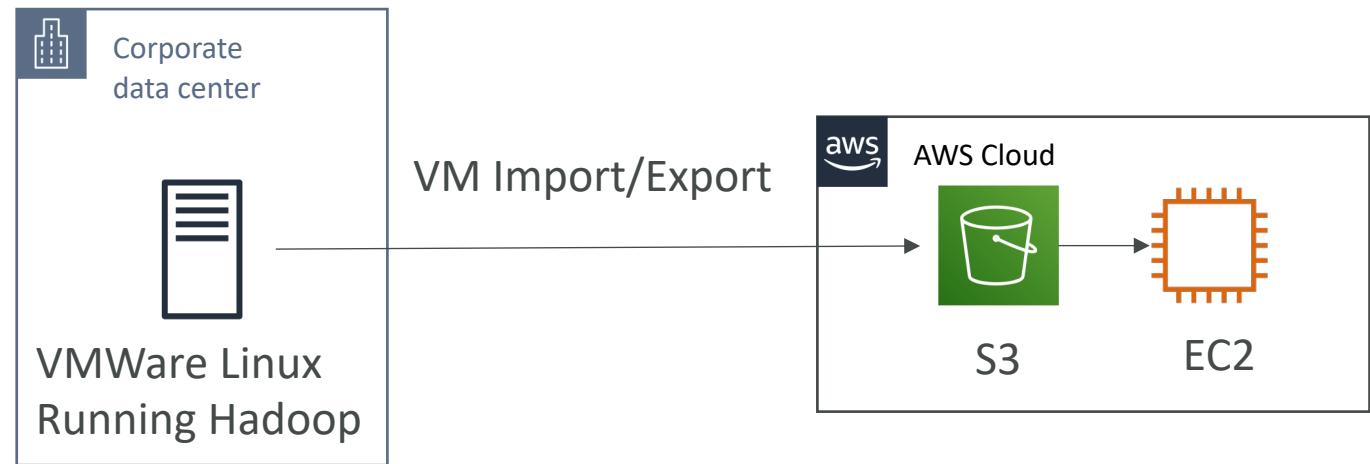
# Question 8 – Architecture

- A company is migrating an Apache Hadoop cluster from its data center to AWS. The cluster consists of 60 VMware Linux virtual machines (VMs). During the migration cluster, downtime should be minimized.
- Which process will minimize downtime?



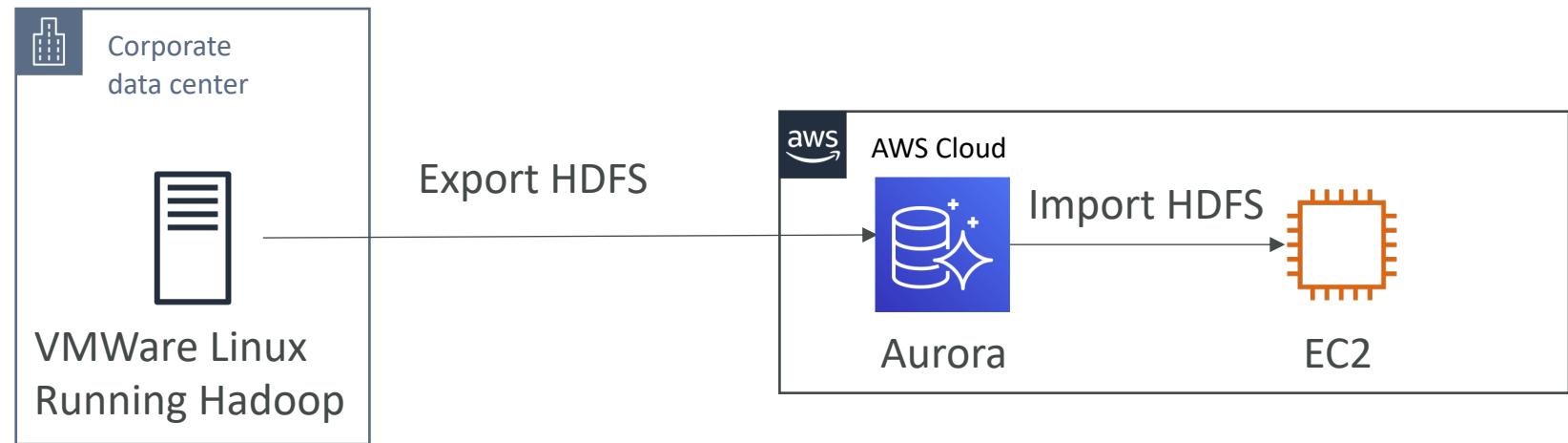
# Option C

- Create OVA files of the VMs. Upload the OVA files to Amazon S3. Use VM Import/Export to create AMIs from the OVA files. Launch the cluster on AWS as Amazon EC2 instances from the AMIs



# Option D

- Export the HDFS data from the VMs to a new Amazon Aurora database. Launch a new Hadoop cluster on Amazon EC2 instances. Import the data from the Aurora database to HDFS on the new cluster.



# Option A

- Use the AWS Management Portal for vCenter to migrate the VMs to AWS as Amazon EC2 instances

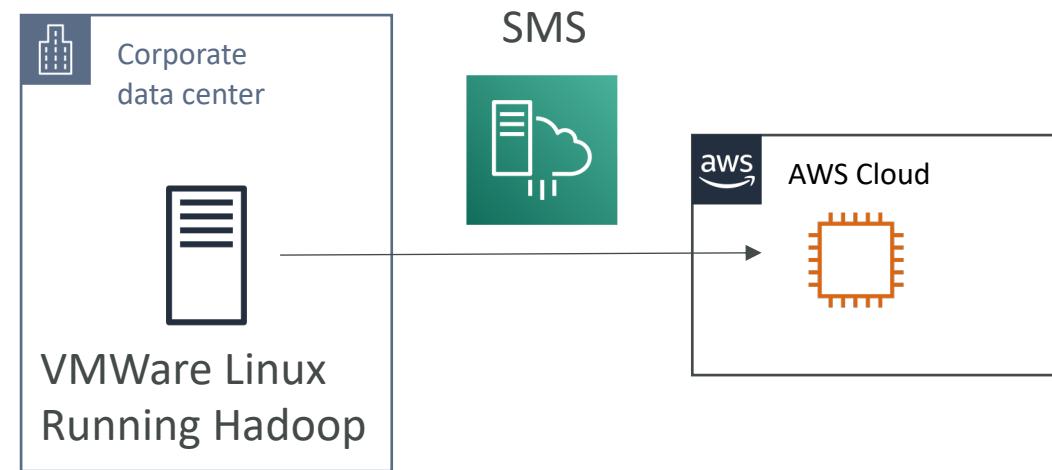
We recommend using AWS Server Migration Service (SMS) to migrate VMs from a vCenter environment to AWS. SMS automates the migration process by replicating on-premises VMs incrementally and converting them to Amazon machine images (AMIs). You can continue using your on-premises VMs while migration is in progress. For more information about AWS SMS, see [AWS Server Migration Service](#).

If any of the following are true, you should consider using AWS SMS:

- You are using vCenter 6.5 Server.
- You want to specify BYOL licenses during migration.
- You are interested in migrating VMs to Amazon EC2.
- You want to use incremental migration.

# Option B

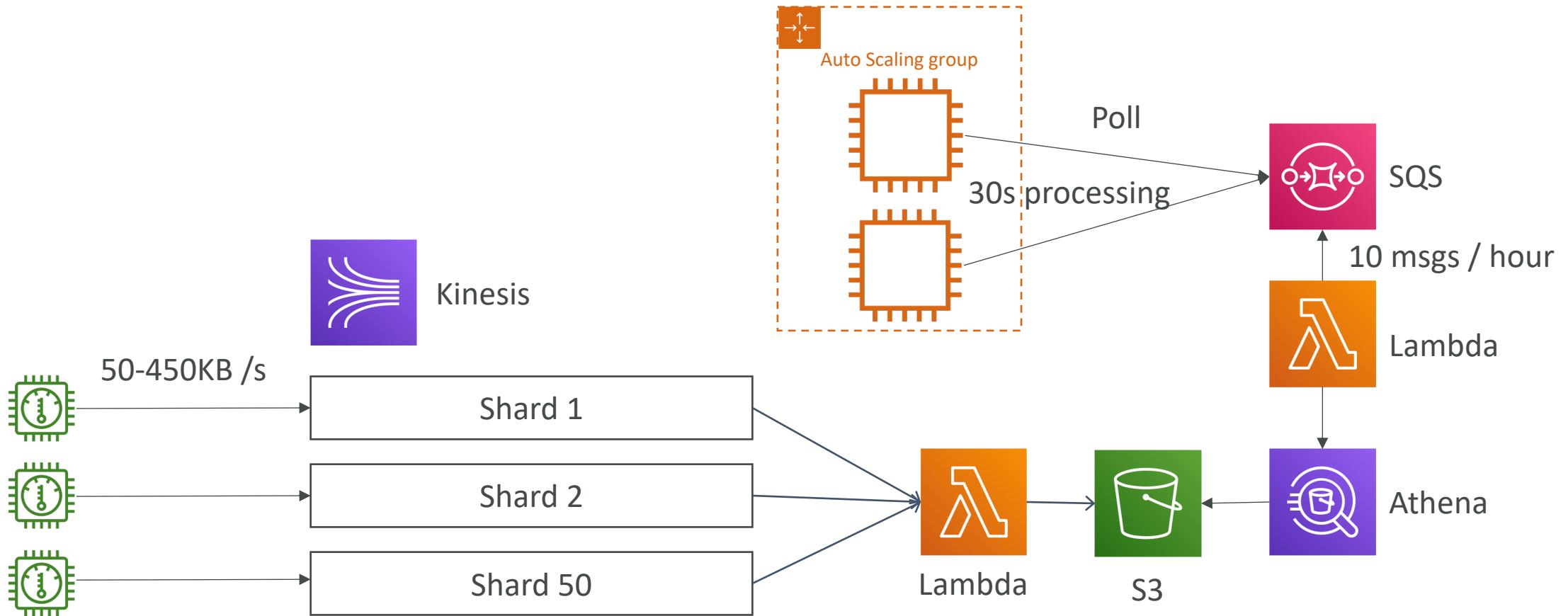
- Use AWS SMS to migrate the VMs to AWS as AMIs. Launch the cluster on AWS as Amazon EC2 instances from the migrated AMIs



# Question 9

- A solutions architect needs to reduce costs for a big data application. The application environment consists of hundreds of devices that send events to Amazon Kinesis Data Streams. The device ID is used as the partition key, so each device gets a separate shard. Each device sends between 50 KB and 450 KB of data per second. The shards are polled by an AWS Lambda function that processes the data and stores the result on Amazon S3.
- Every hour, an AWS Lambda function runs an Amazon Athena query against the result data that identifies any outliers and places them in an Amazon SQS queue. An Amazon EC2 Auto Scaling group of two EC2 instances monitors the queue and runs a short (approximately 30-second) process to address the outliers. The devices submit an average of 10 outlying values every hour.
- Which combination of changes to the application would MOST reduce costs? (Select TWO.)

# Question 9 – Architecture



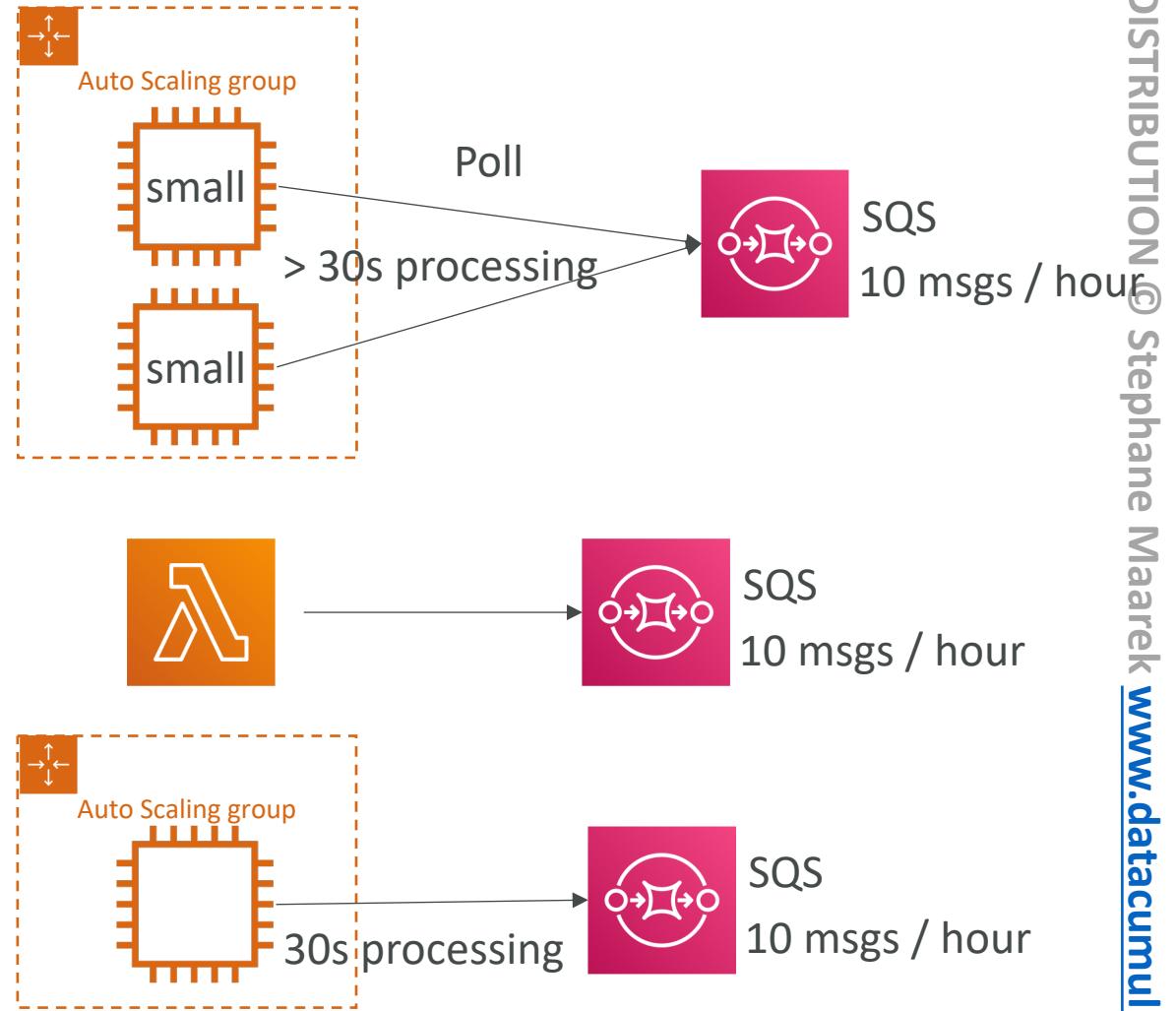
# Options (choose 2)

- A) Change the Auto Scaling group launch configuration to use smaller instance types in the same instance family.
- B) Replace the Auto Scaling group with an AWS Lambda function triggered by messages arriving in the Amazon SQS queue.
- C) Reconfigure the devices and data stream to set a ratio of 10 devices to 1 data stream shard.
- D) Reconfigure the devices and data stream to set a ratio of 2 devices to 1 data stream shard.
- E) Change the desired capacity of the Auto Scaling group to a single EC2 instance.

**CORRECT ANSWER B**

# Option Group I

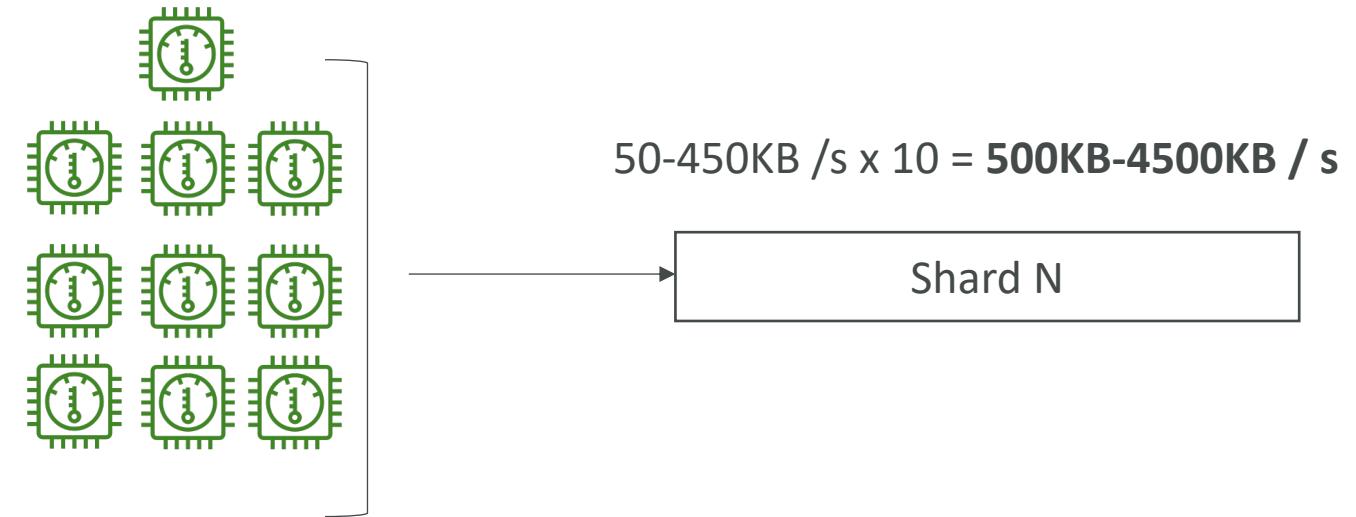
- A) Change the Auto Scaling group launch configuration to use smaller instance types in the same instance family.
- B) Replace the Auto Scaling group with an AWS Lambda function triggered by messages arriving in the Amazon SQS queue.
- E) Change the desired capacity of the Auto Scaling group to a single EC2 instance.



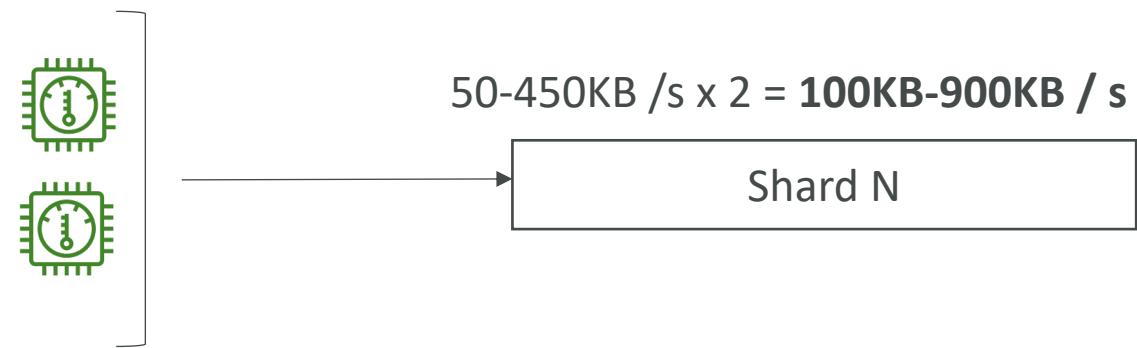
**CORRECT ANSWER D**

# Option Group 2

C) Reconfigure the devices and data stream to set a ratio of 10 devices to 1 data stream shard.



D) Reconfigure the devices and data stream to set a ratio of 2 devices to 1 data stream shard.

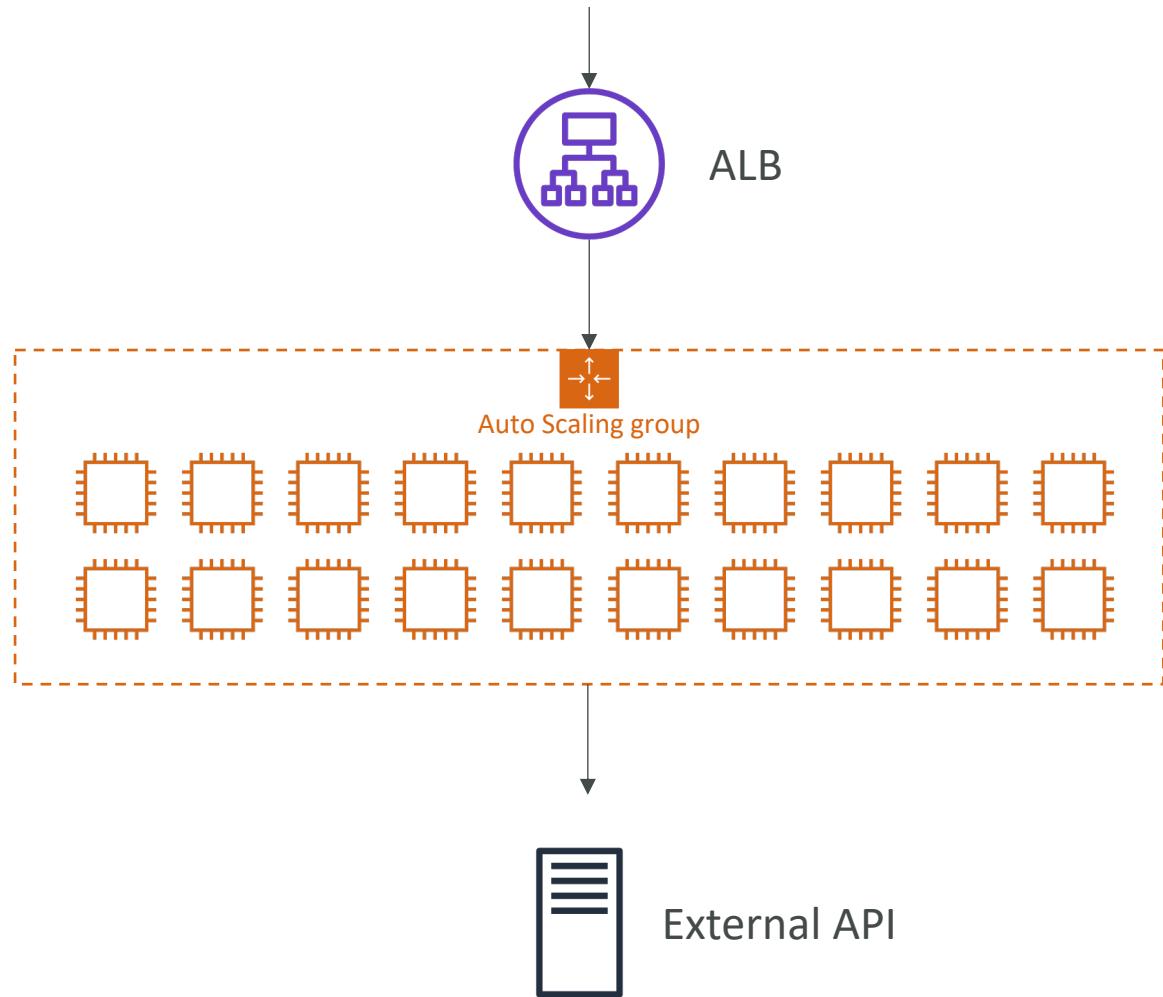


Each shard has a limit of 1MB/s = 1000KB/s

# Question 10

- A company operates an ecommerce application on Amazon EC2 instances behind an ELB Application Load Balancer. The instances run in an Amazon EC2 Auto Scaling group across multiple Availability Zones. After an order is successfully processed, the application immediately posts order data to an external third-party affiliate tracking system that pays sales commissions for order referrals. During a highly successful marketing promotion, the number of EC2 instances increased from 2 to 20. The application continued to work correctly, but the increased request rate overwhelmed the third-party affiliate and resulted in failed requests.
- Which combination of architectural changes could ensure that the entire process functions correctly under load? (Select TWO.)

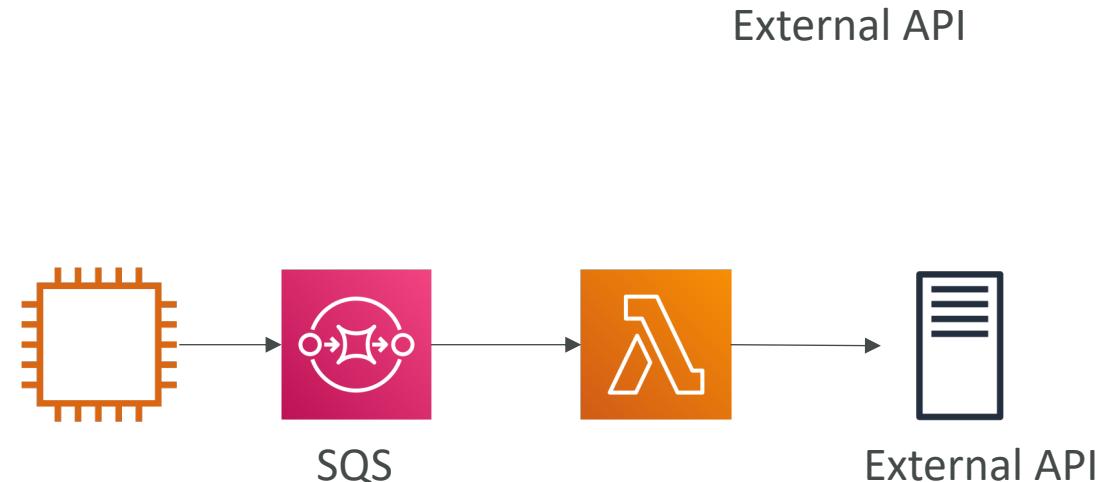
# Question 10 – Architecture



**CORRECT ANSWER B**

# Option Group I

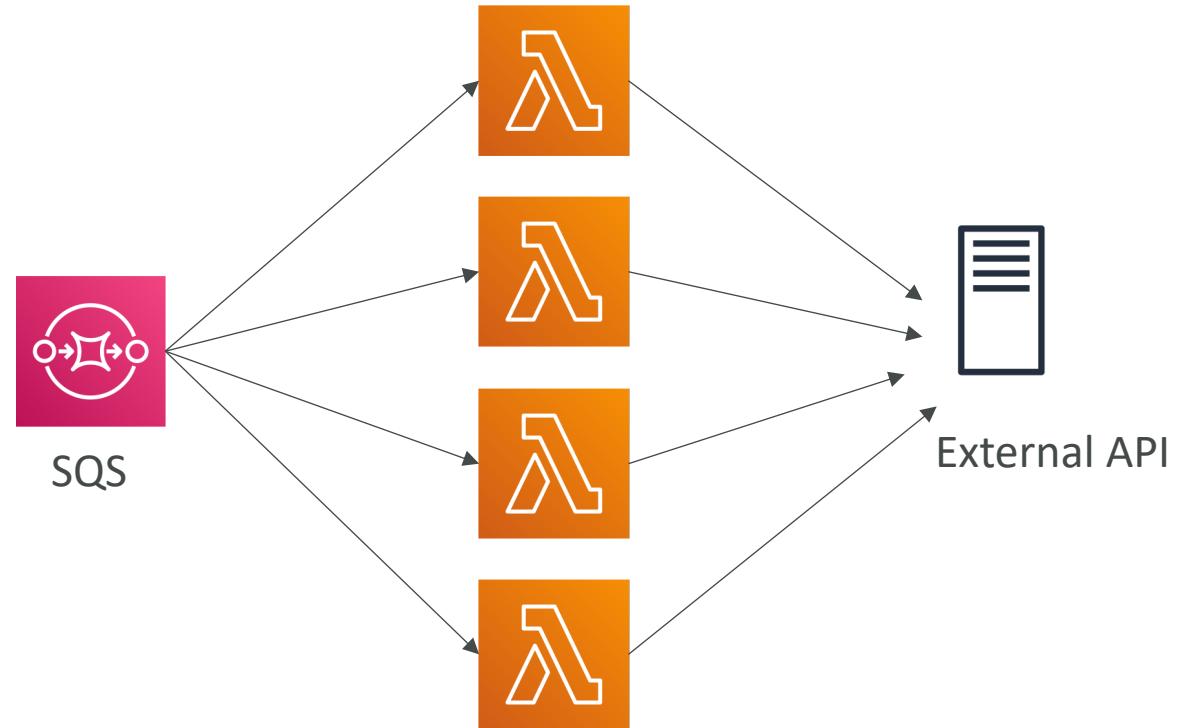
- A) Move the code that calls the affiliate to a new AWS Lambda function. Modify the application to invoke the Lambda function asynchronously.
- B) Move the code that calls the affiliate to a new AWS Lambda function. Modify the application to place the order data in an Amazon SQS queue. Trigger the Lambda function from the queue.



**CORRECT ANSWER D**

# Option Group 2

- C) Increase the timeout of the new AWS Lambda function.
- D) Adjust the concurrency limit of the new AWS Lambda function.
- E) Increase the memory of the new AWS Lambda function.

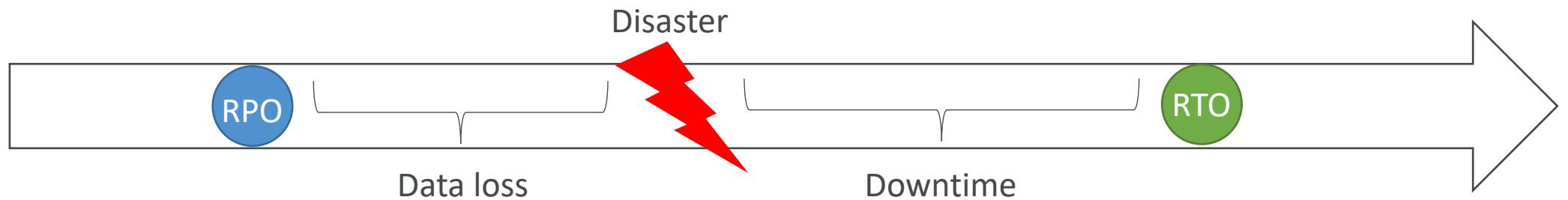


# Analysis from the Practice Sample questions

- Note: the questions are from 2015 – the certification has definitely evolved since then, but the questions are still interesting!
- Download them from the PDF resource attached to this lecture

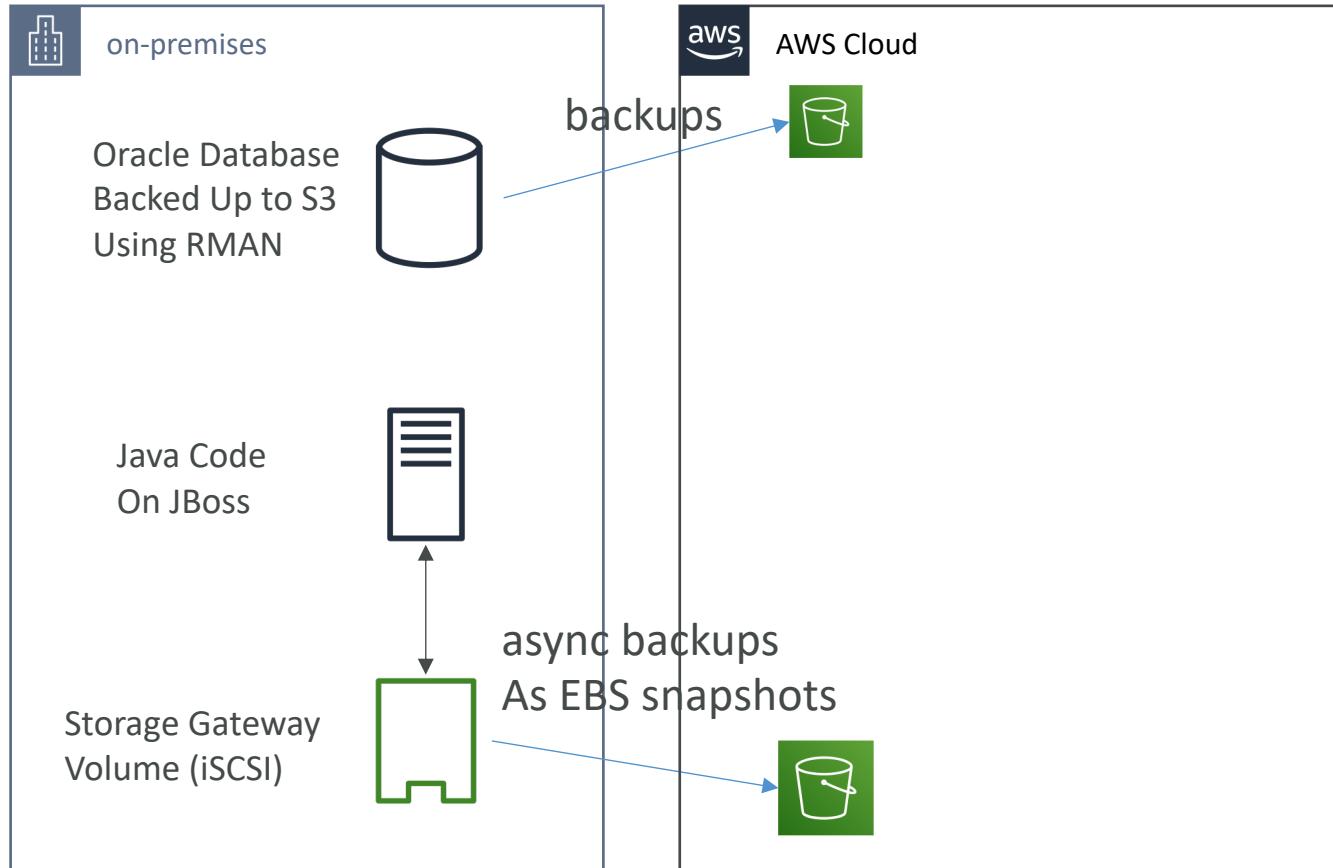
# Question 1

- Your company's on-premises content management system has the following architecture:
  - Application Tier – Java code on a JBoss application server
  - Database Tier – Oracle database regularly backed up to Amazon Simple Storage Service (S3) using the Oracle RMAN backup utility
  - Static Content – stored on a 512GB gateway stored Storage Gateway volume attached to the application server via the iSCSI interface
- Which AWS based disaster recovery strategy will give you the best RTO?



# Question 1 – Architecture Diagram

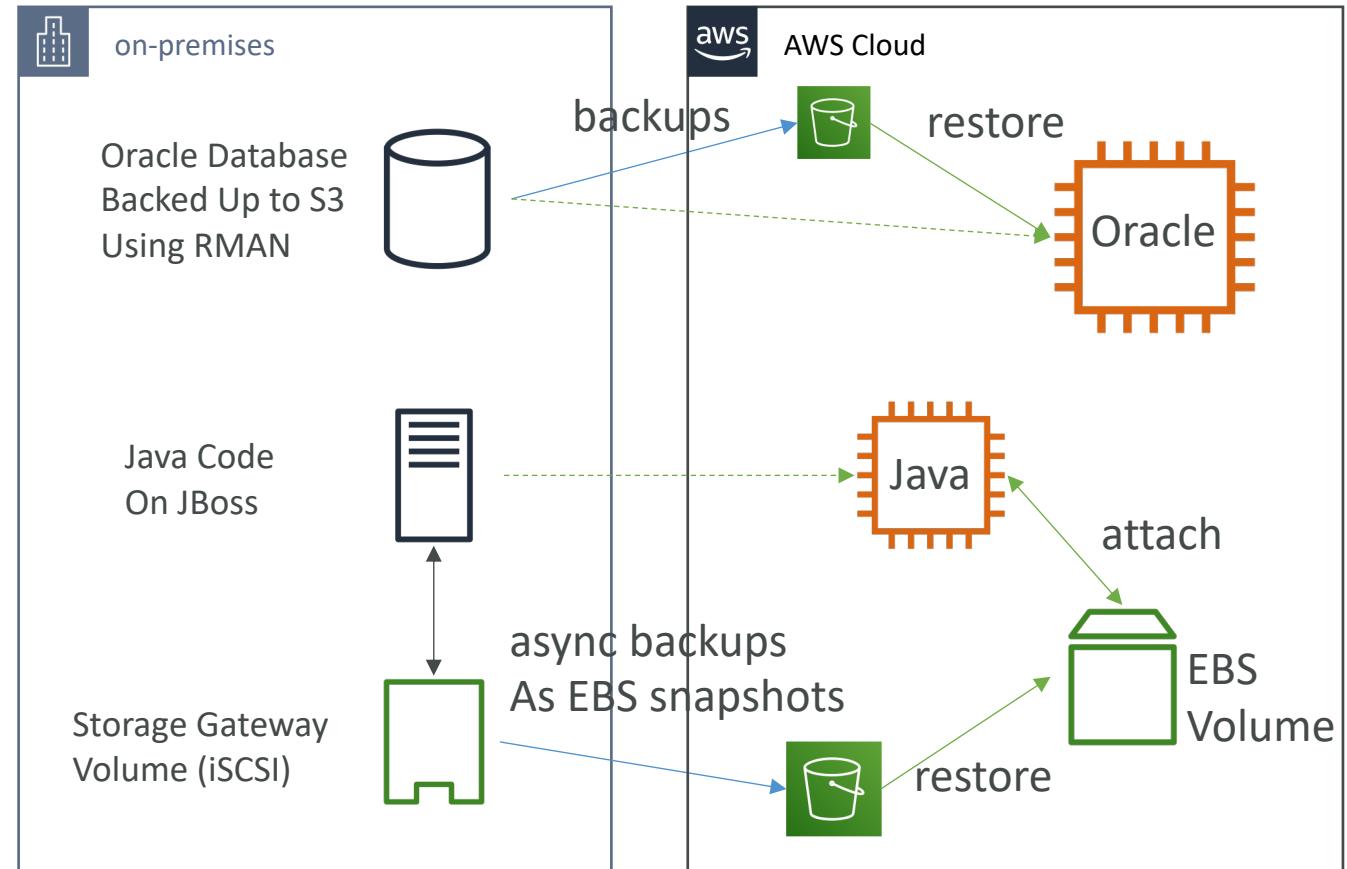
- Your company's on-premises content management system has the following architecture:
  - Application Tier – Java code on a JBoss application server
  - Database Tier – Oracle database regularly backed up to Amazon Simple Storage Service (S3) using the Oracle RMAN backup utility
  - Static Content – stored on a 512GB gateway stored Storage Gateway volume attached to the application server via the iSCSI interface
- Which AWS based disaster recovery strategy will give you the best RTO?



**CORRECT ANSWER**

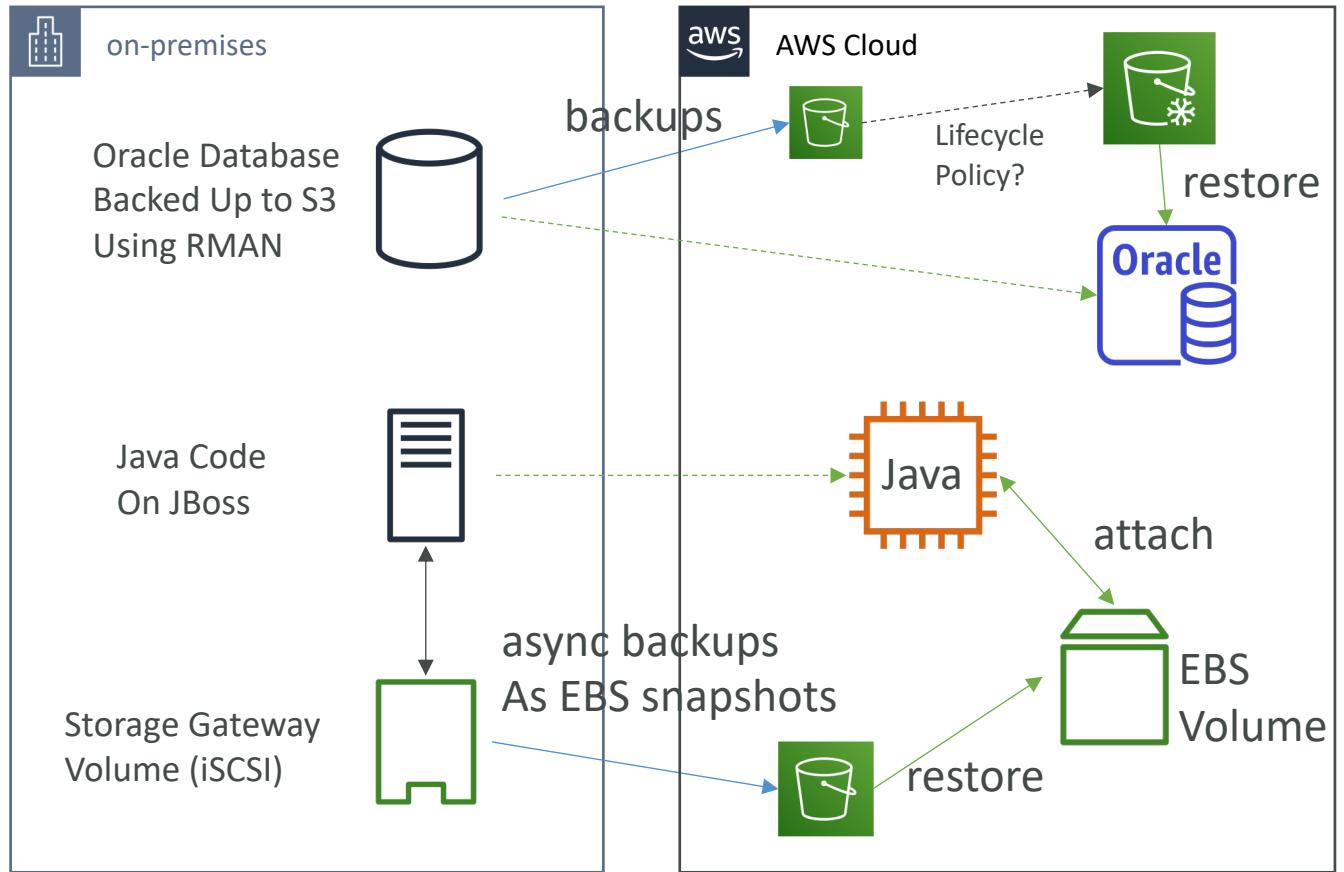
# Option A

- Deploy the Oracle database and the JBoss app server on EC2. Restore the RMAN Oracle backups from Amazon S3. Generate an EBS volume of static content from the Storage Gateway and attach it to the JBoss EC2 server.



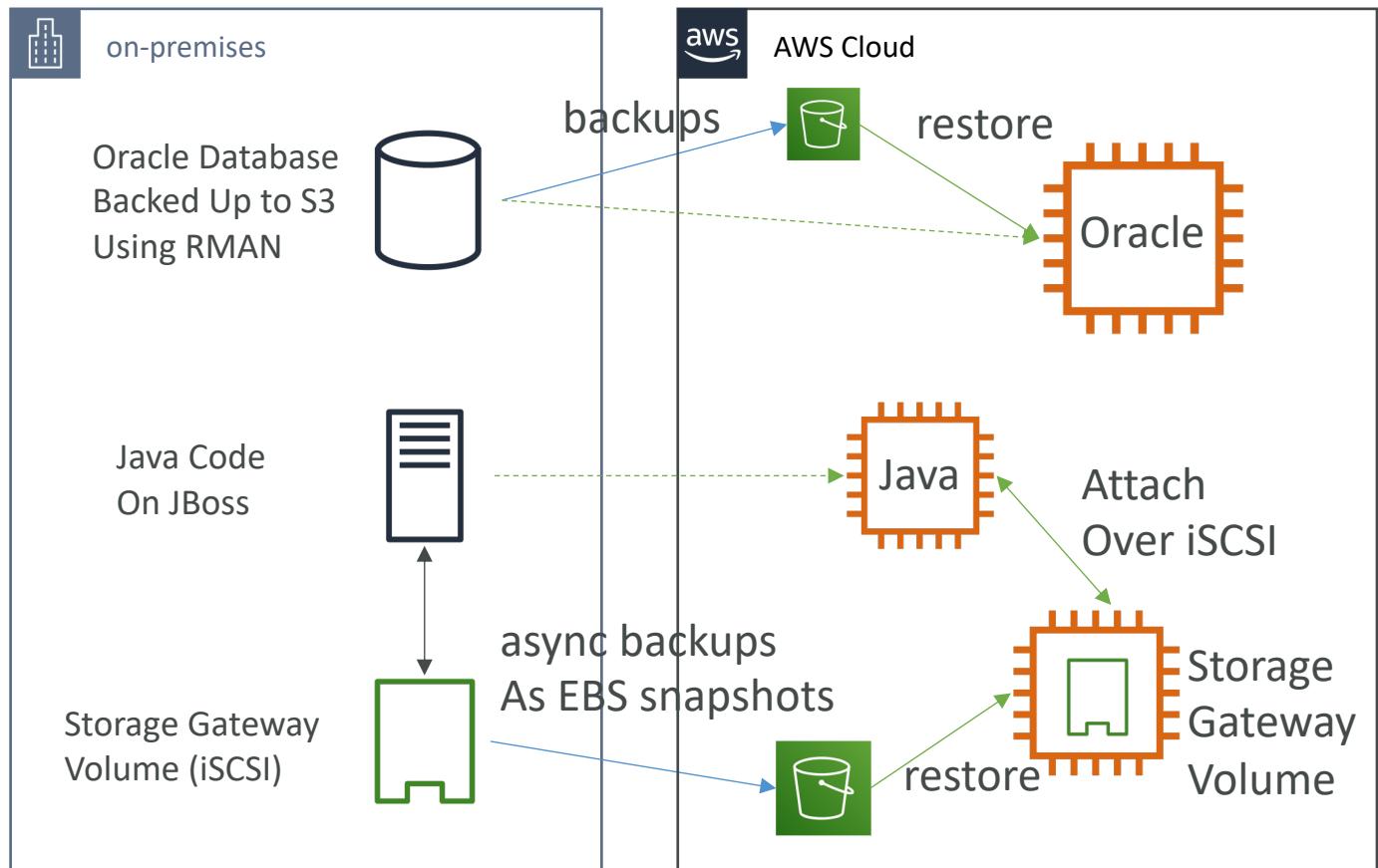
# Option B

- Deploy the Oracle database on RDS.
- Deploy the JBoss app server on EC2.
- Restore the RMAN Oracle backups from Amazon Glacier.
- Generate an EBS volume of static content from the Storage Gateway and attach it to the JBoss EC2 server.



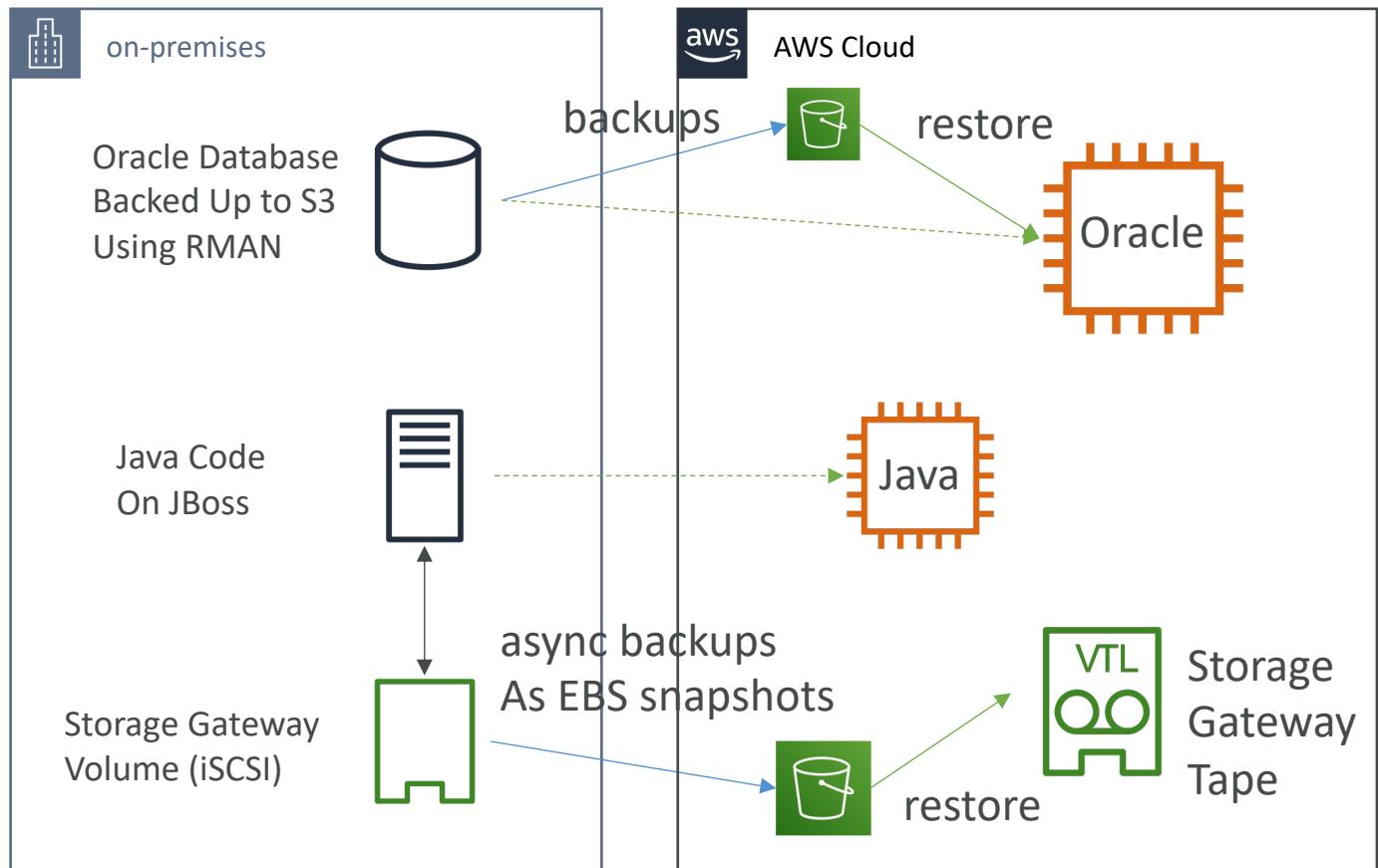
# Option C

- Deploy the Oracle database and the JBoss app server on EC2. Restore the RMAN Oracle backups from Amazon S3. Restore the static content by attaching an AWS Storage Gateway running on Amazon EC2 as an iSCSI volume to the JBoss EC2 server.



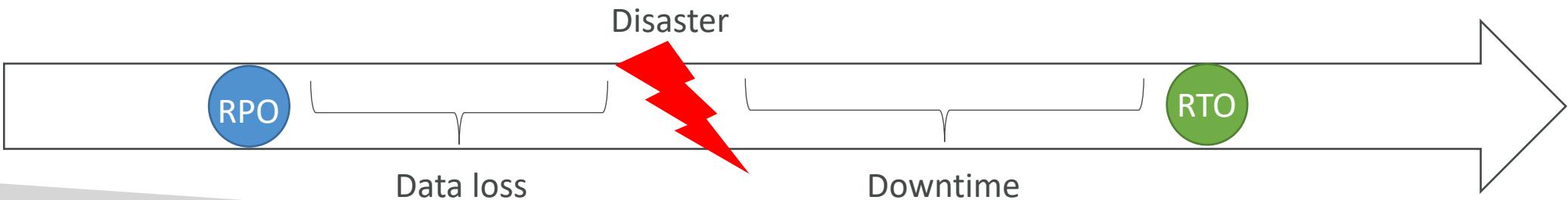
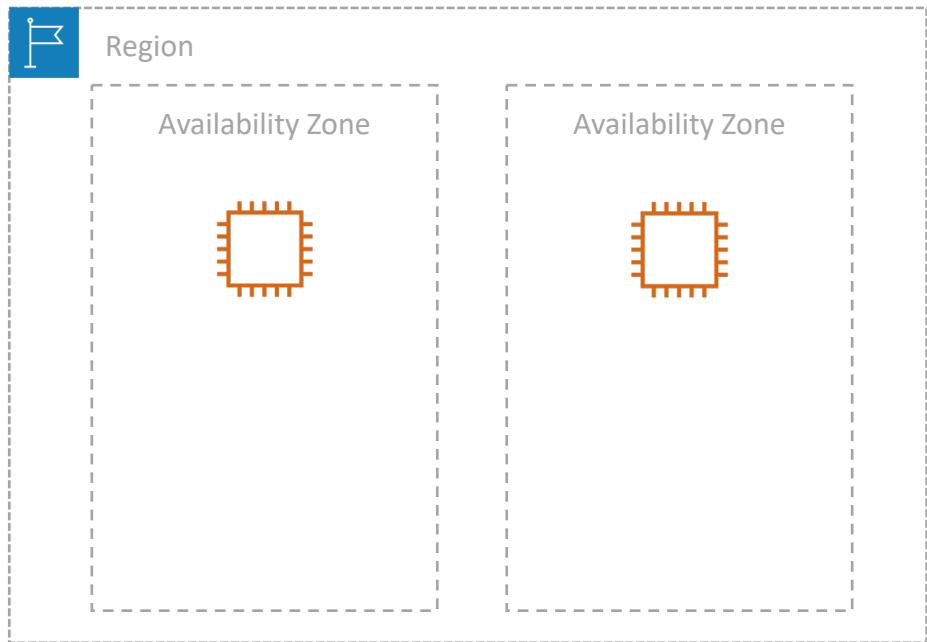
# Option D

- Deploy the Oracle database and the JBoss app server on EC2. Restore the RMAN Oracle backups from Amazon S3. Restore the static content from an AWS Storage Gateway-VTL running on Amazon EC2



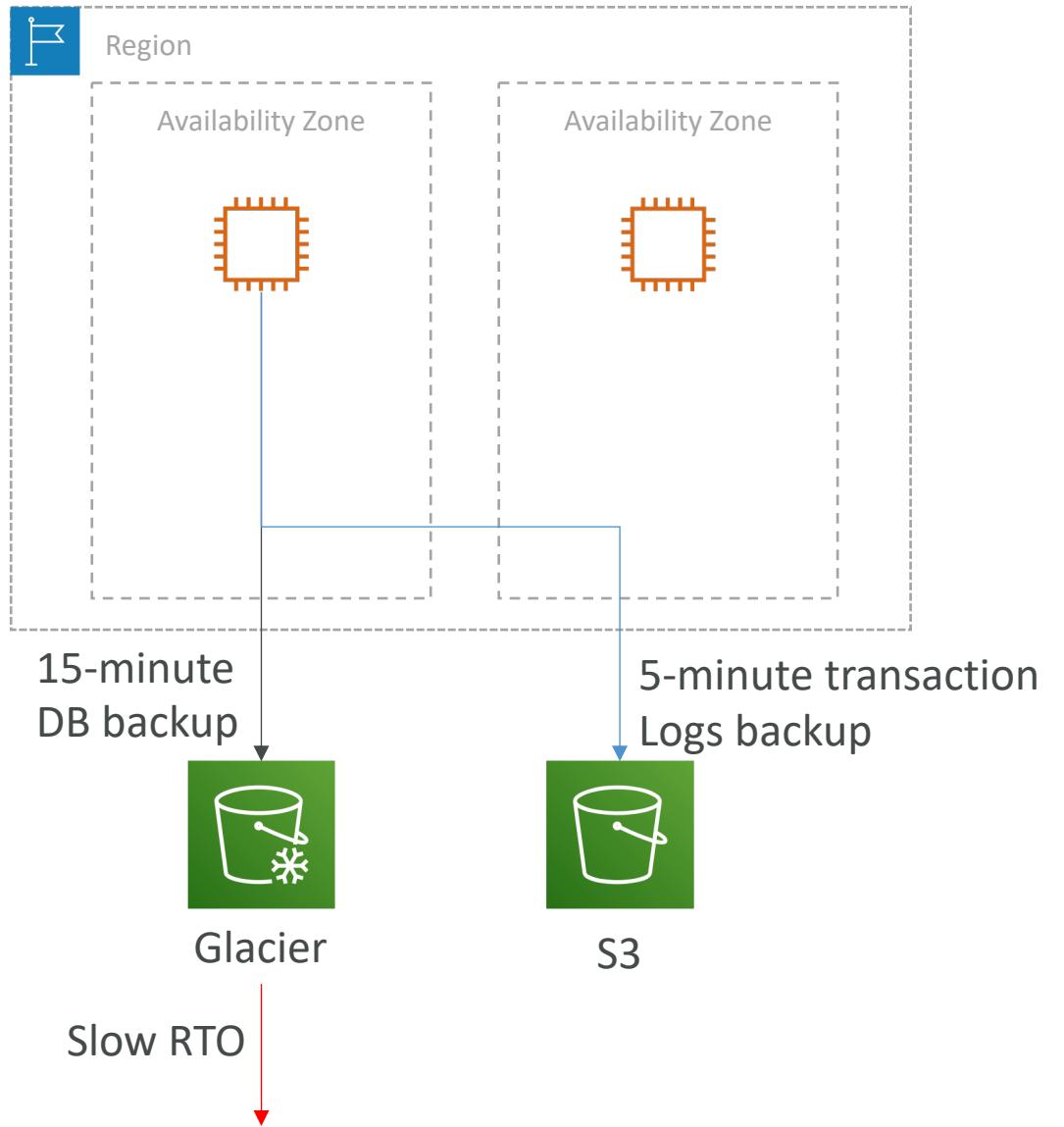
# Question 2

- An ERP application is deployed in multiple Availability Zones in a single region. In the event of failure, the RTO must be less than 3 hours, and the RPO is 15 minutes. The customer realizes that data corruption occurred roughly 1.5 hours ago. Which DR strategy can be used to achieve this RTO and RPO in the event of this kind of failure?



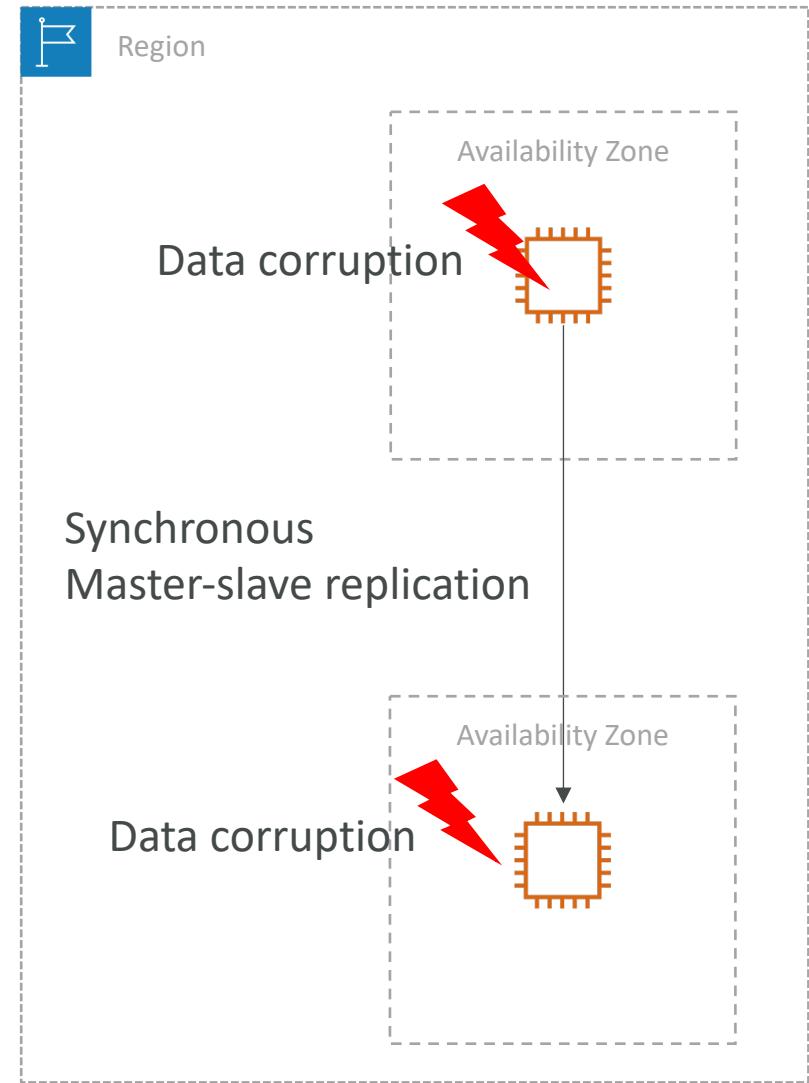
# Option A

- Take 15-minute DB backups stored in Amazon Glacier, with transaction logs stored in Amazon S3 every 5 minutes.



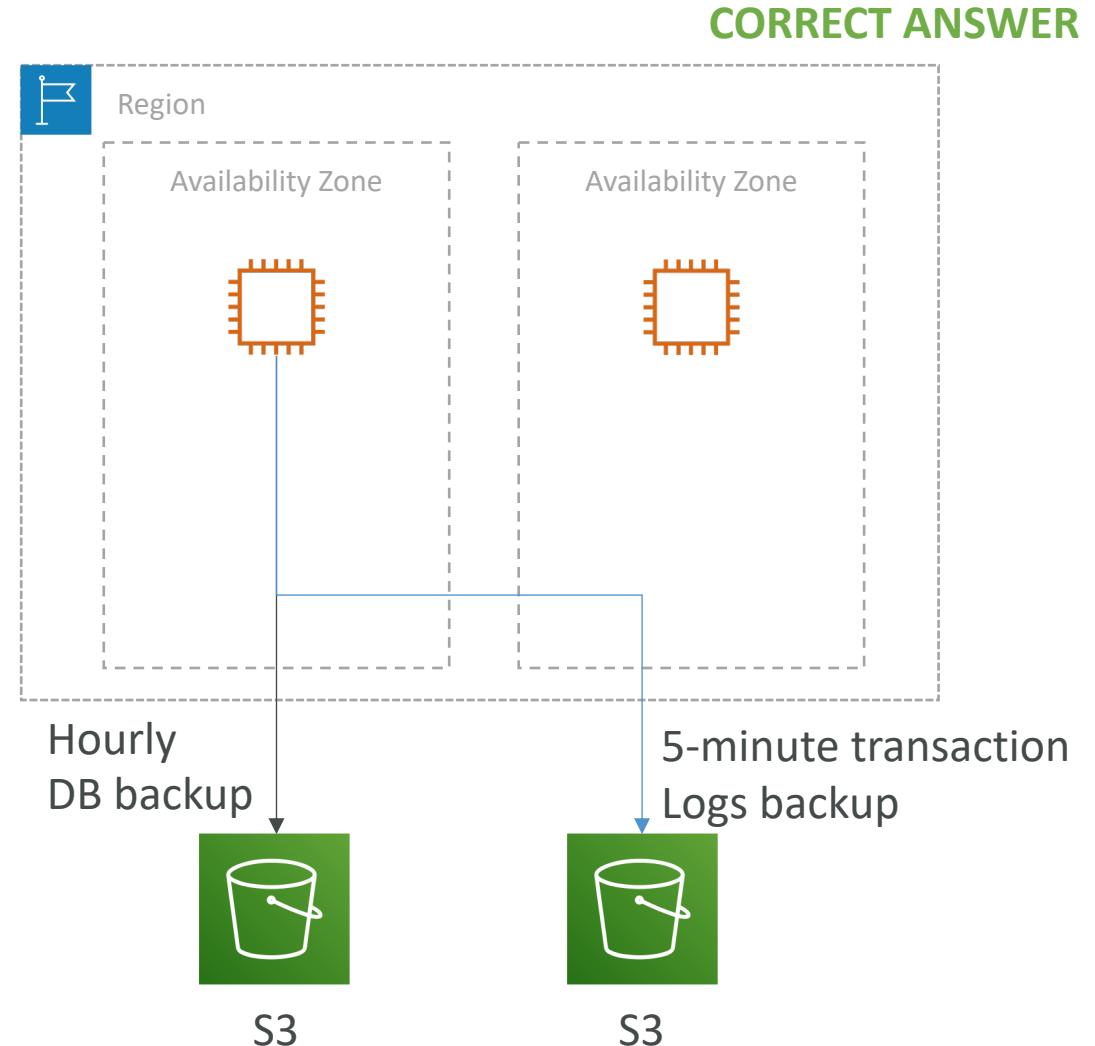
# Option B

- Use synchronous database master-slave replication between two Availability Zones.



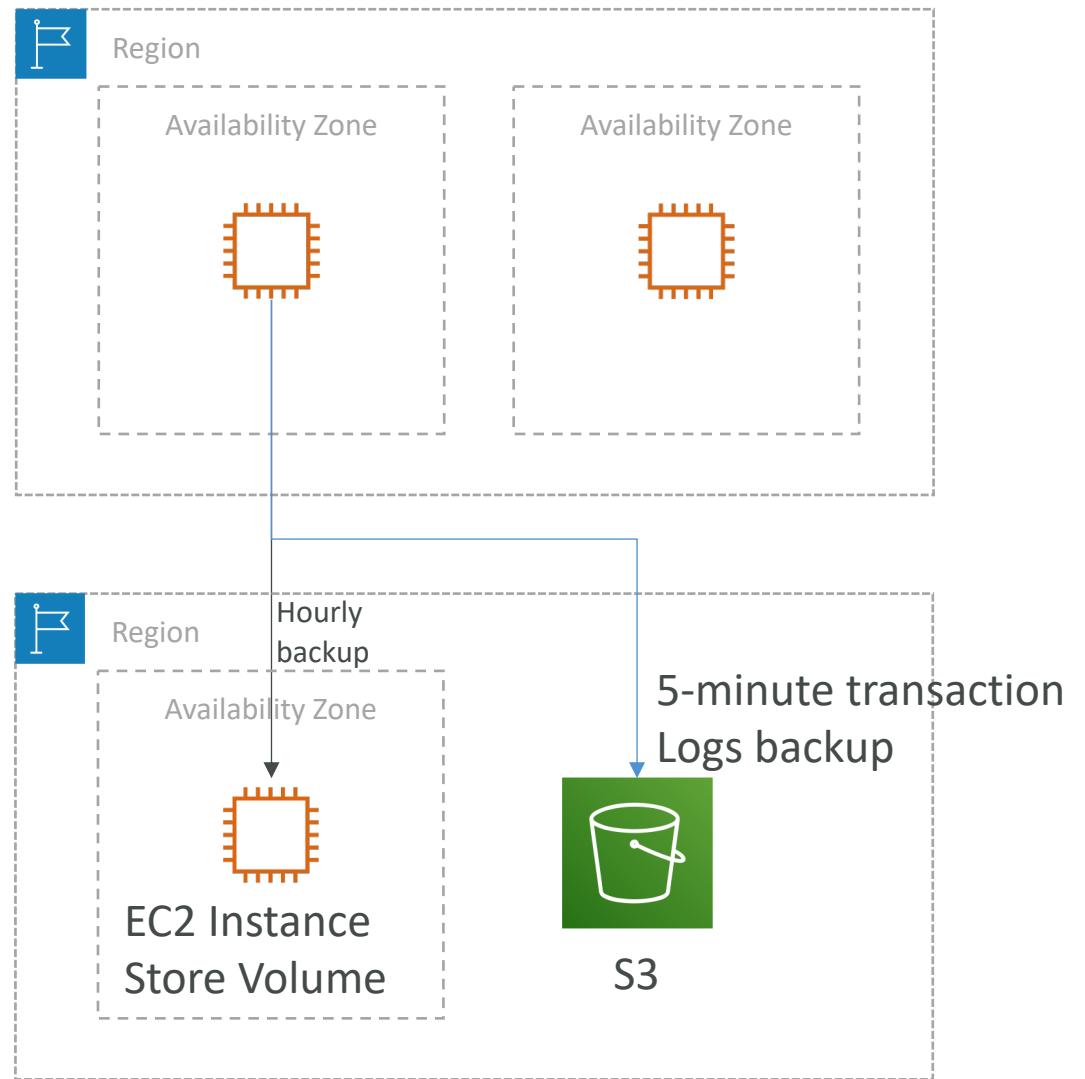
# Option C

- Take hourly DB backups to Amazon S3, with transaction logs stored in S3 every 5 minutes



# Option D

- Take hourly DB backups to an Amazon EC2 instance store volume, with transaction logs stored in Amazon S3 every 5 minutes.



# Question 3

- The Marketing Director in your company asked you to create a mobile app that lets users post sightings of good deeds known as random acts of kindness in 80-character summaries. You decided to write the application in JavaScript so that it would run on the broadest range of phones, browsers, and tablets. Your application should provide access to Amazon DynamoDB to store the good deed summaries. Initial testing of a prototype shows that there aren't large spikes in usage. Which option provides the most **cost-effective and scalable** architecture for this application?

# Question 3 – Architecture Diagram



JavaScript

Run on Tablets, Web Browsers,  
Mobile Applications



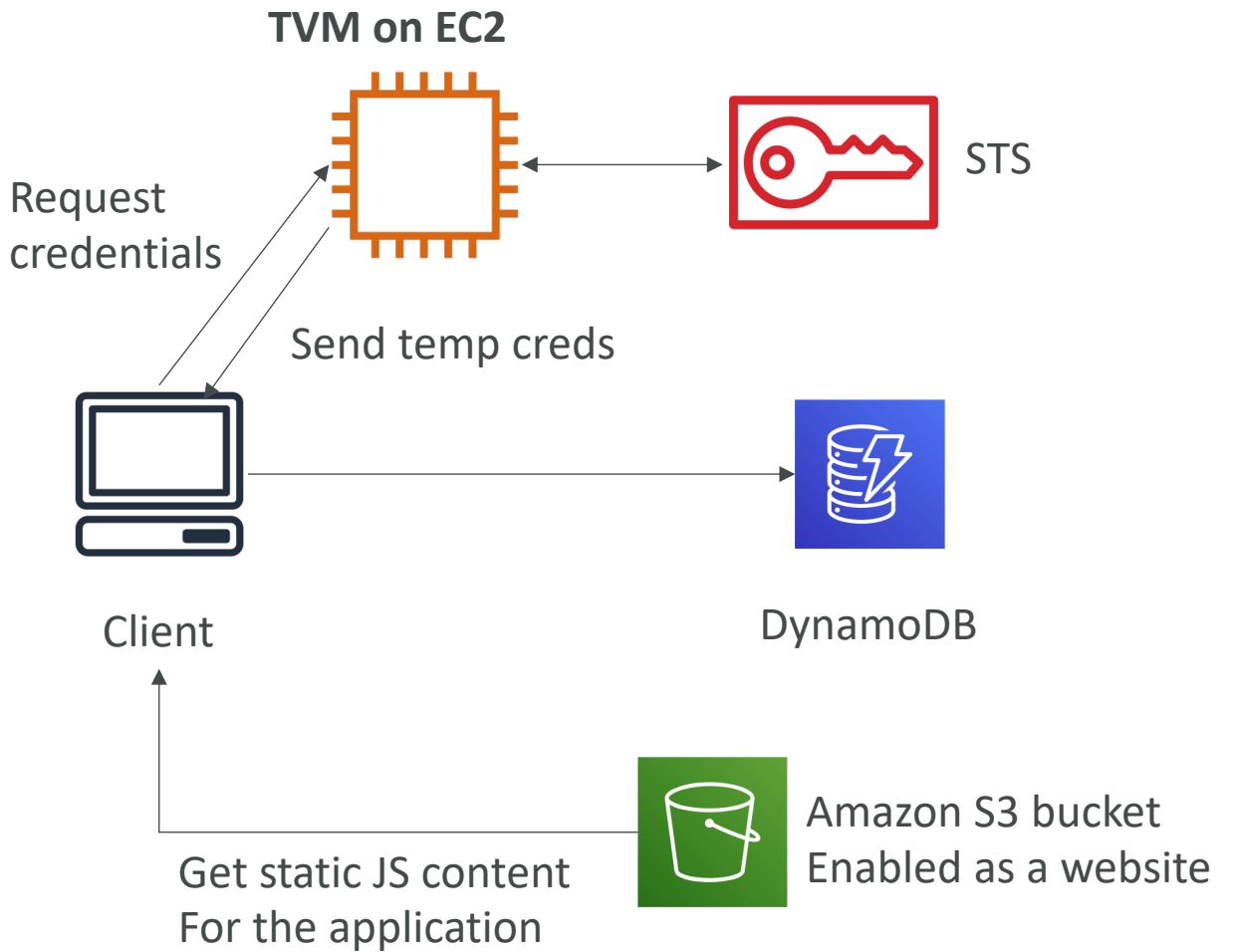
Client



DynamoDB

# Option A

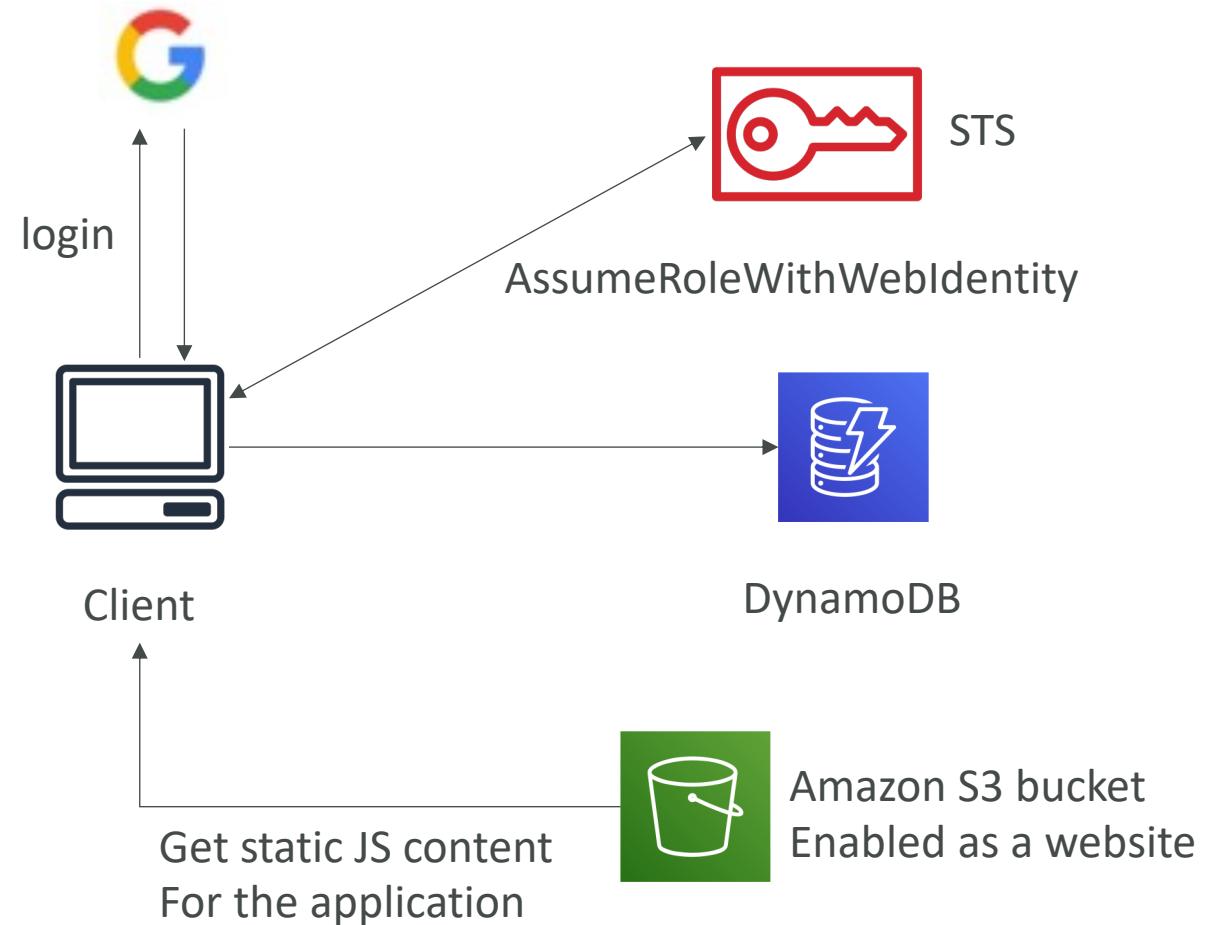
- Provide the JavaScript client with temporary credentials from the Security Token Service using a Token Vending Machine (TVM) on an EC2 instance to provide signed credentials mapped to an Amazon Identity and Access Management (IAM) user allowing DynamoDB puts and S3 gets. You serve your mobile application out of an S3 bucket enabled as a web site. Your client updates DynamoDB.



**CORRECT ANSWER**

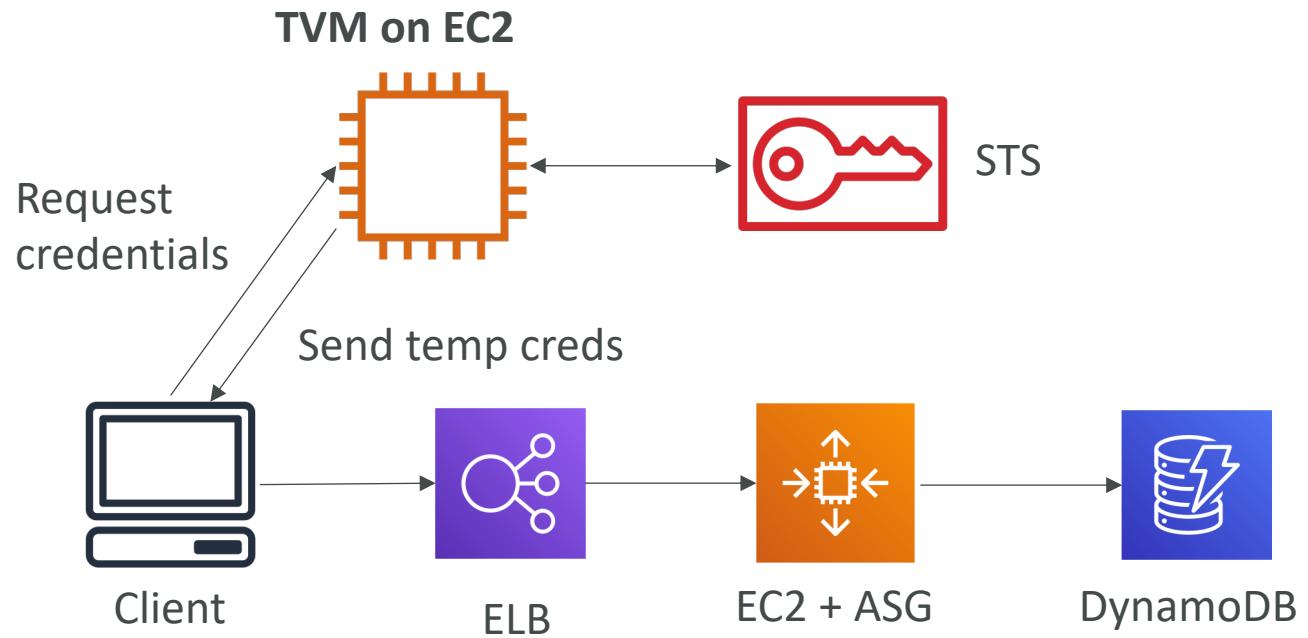
# Option B

- Register the application with a Web Identity Provider like Amazon, Google, or Facebook, create an IAM role for that provider, and set up permissions for the IAM role to allow S3 gets and DynamoDB puts. You serve your mobile application out of an S3 bucket enabled as a web site. Your client updates DynamoDB.



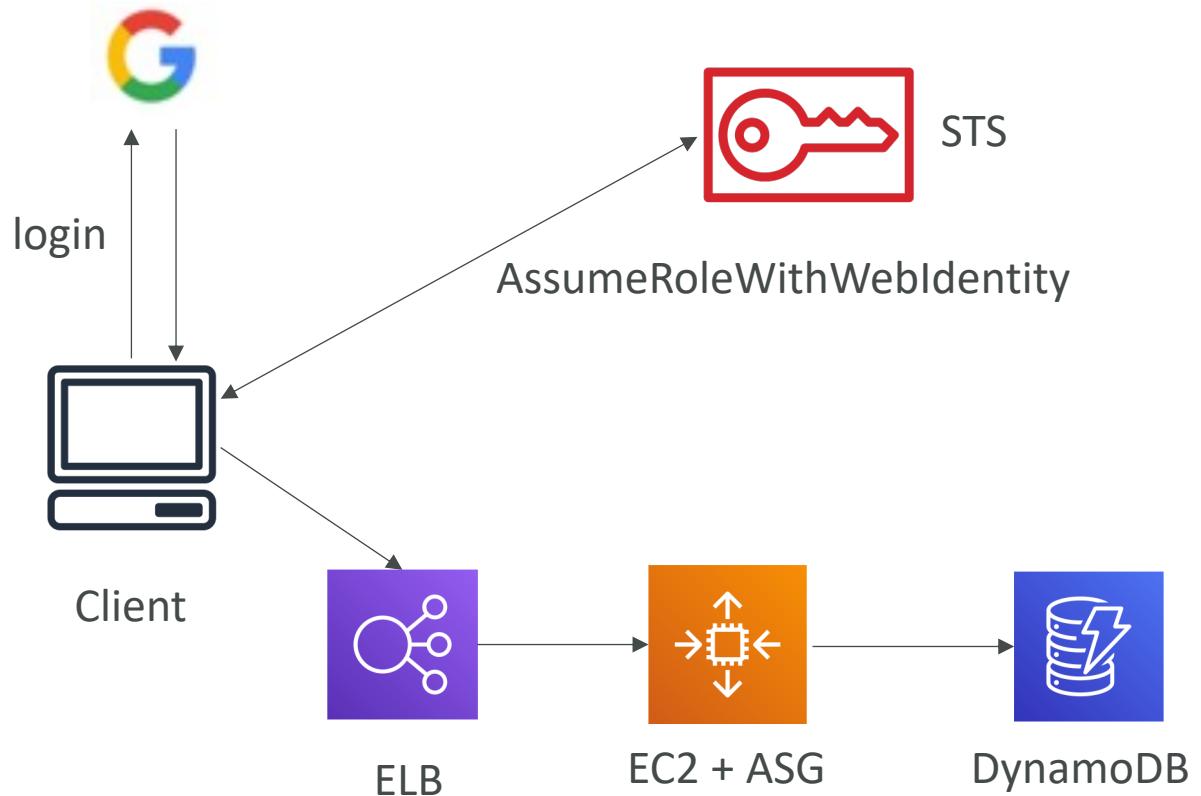
# Option C

- Provide the JavaScript client with temporary credentials from the Security Token Service using a Token Vending Machine (TVM) to provide signed credentials mapped to an IAM user allowing DynamoDB puts. You serve your mobile application out of Apache EC2 instances that are load-balanced and autoscaled. Your EC2 instances are configured with an IAM role that allows DynamoDB puts. Your server updates DynamoDB.



# Option D

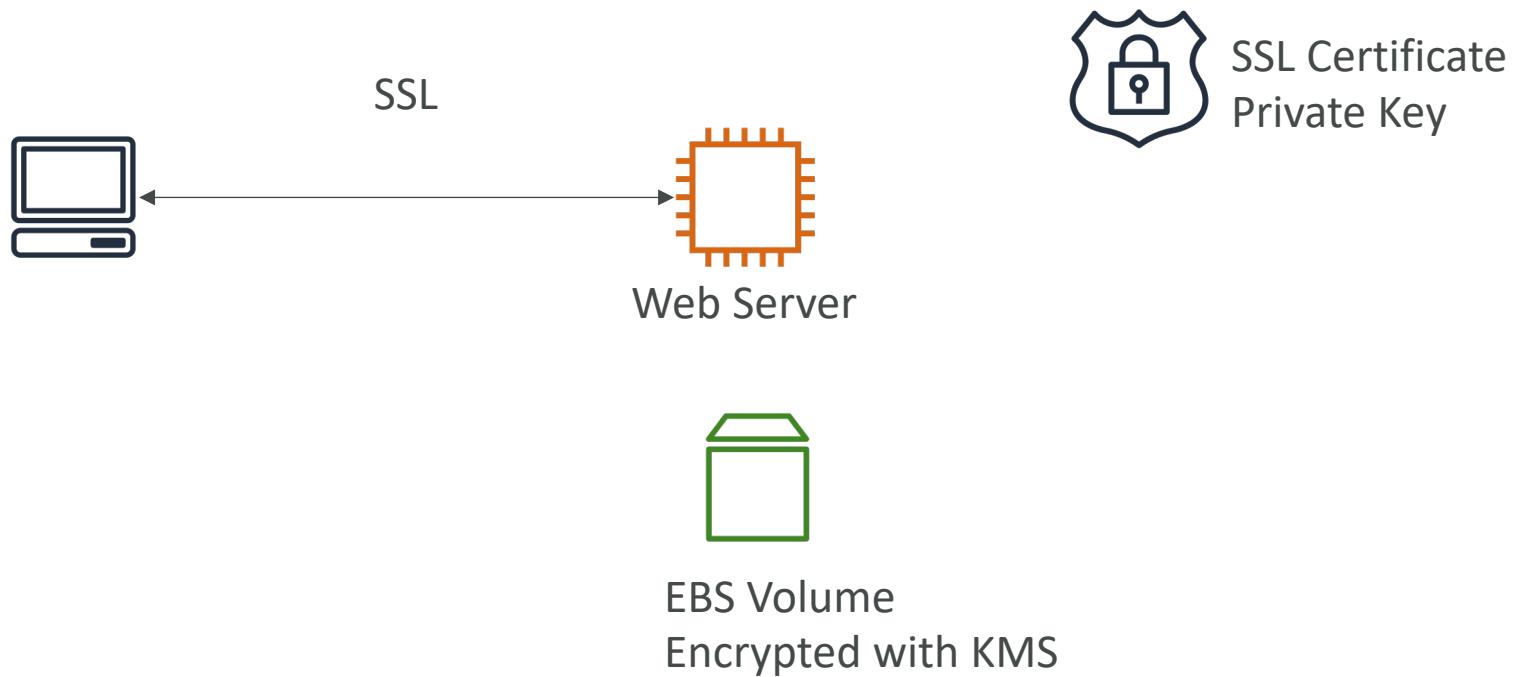
- Register the JavaScript application with a Web Identity Provider like Amazon, Google, or Facebook, create an IAM role for that provider, and set up permissions for the IAM role to allow DynamoDB puts. You serve your mobile application out of Apache EC2 instances that are load-balanced and autoscaled. Your EC2 instances are configured with an IAM role that allows DynamoDB puts. Your server updates DynamoDB.



# Question 4

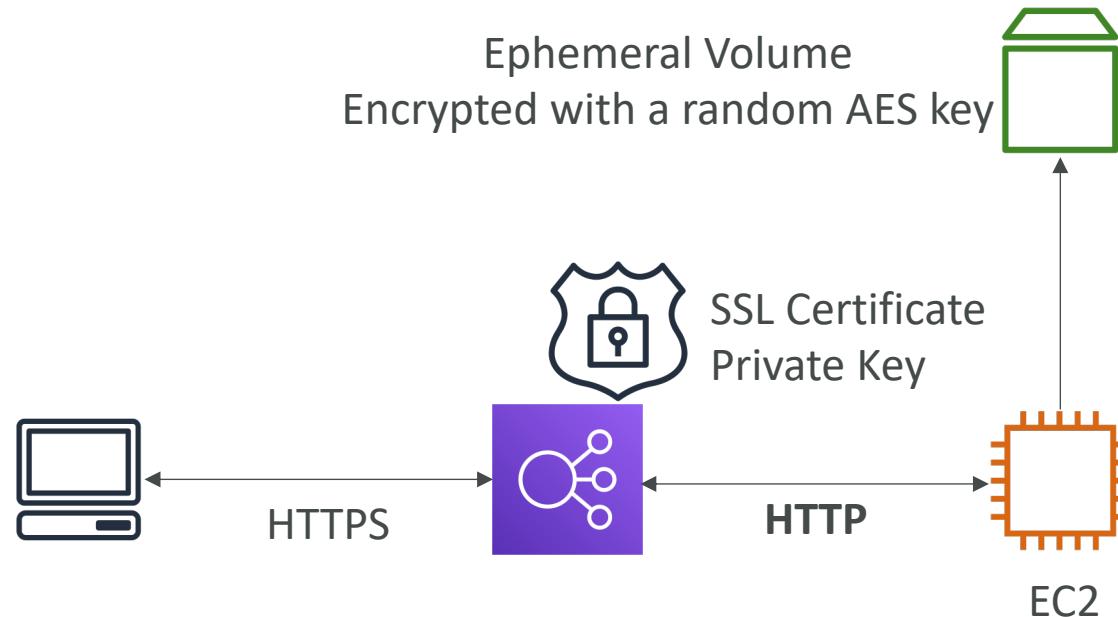
- You are building a website that will retrieve and display highly sensitive information to users. The amount of traffic the site will receive is known and not expected to fluctuate. The site will leverage SSL to protect the communication **between the clients and the web servers**. Due to the nature of the site you are very concerned about the security of your SSL private key and want to ensure that the key cannot be accidentally or intentionally moved outside your environment. Additionally, while the data the site will display is stored on an encrypted EBS volume, you are also concerned that the web servers' logs might contain some sensitive information; therefore, the logs must be stored so that they can only be decrypted by employees of your company. Which of these architectures meets all of the requirements?

# Question 4 – Architecture Diagram



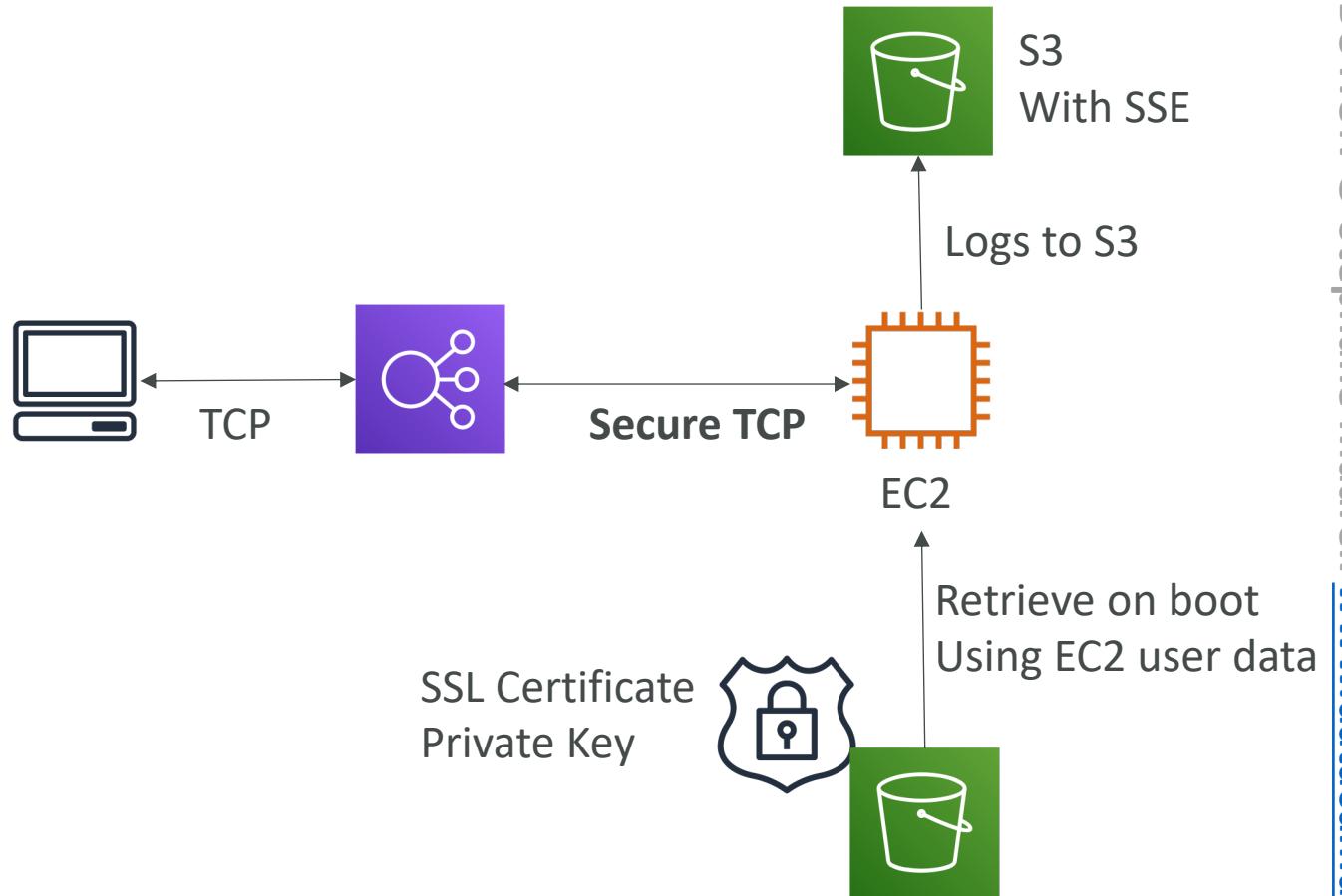
# Option A

- Use Elastic Load Balancing to distribute traffic to a set of web servers. To protect the SSL private key, upload the key to the load balancer and configure the load balancer to offload the SSL traffic. Write your web server logs to an ephemeral volume that has been encrypted using a randomly generated AES key.



# Option B

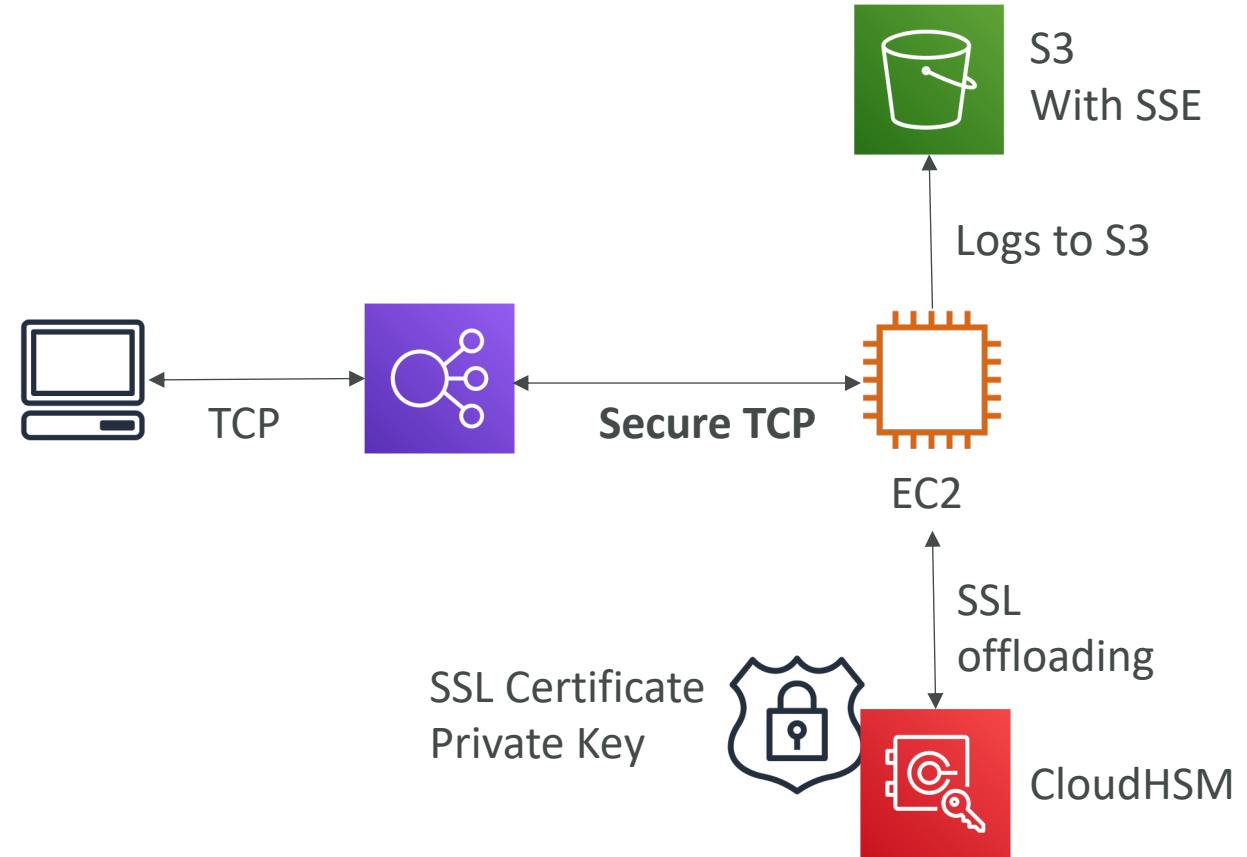
- Use Elastic Load Balancing to distribute traffic to a set of web servers. Use TCP load balancing on the load balancer and configure your web servers to retrieve the private key from a private Amazon S3 bucket on boot. Write your web server logs to a private Amazon S3 bucket using Amazon S3 server-side encryption.



**CORRECT ANSWER**

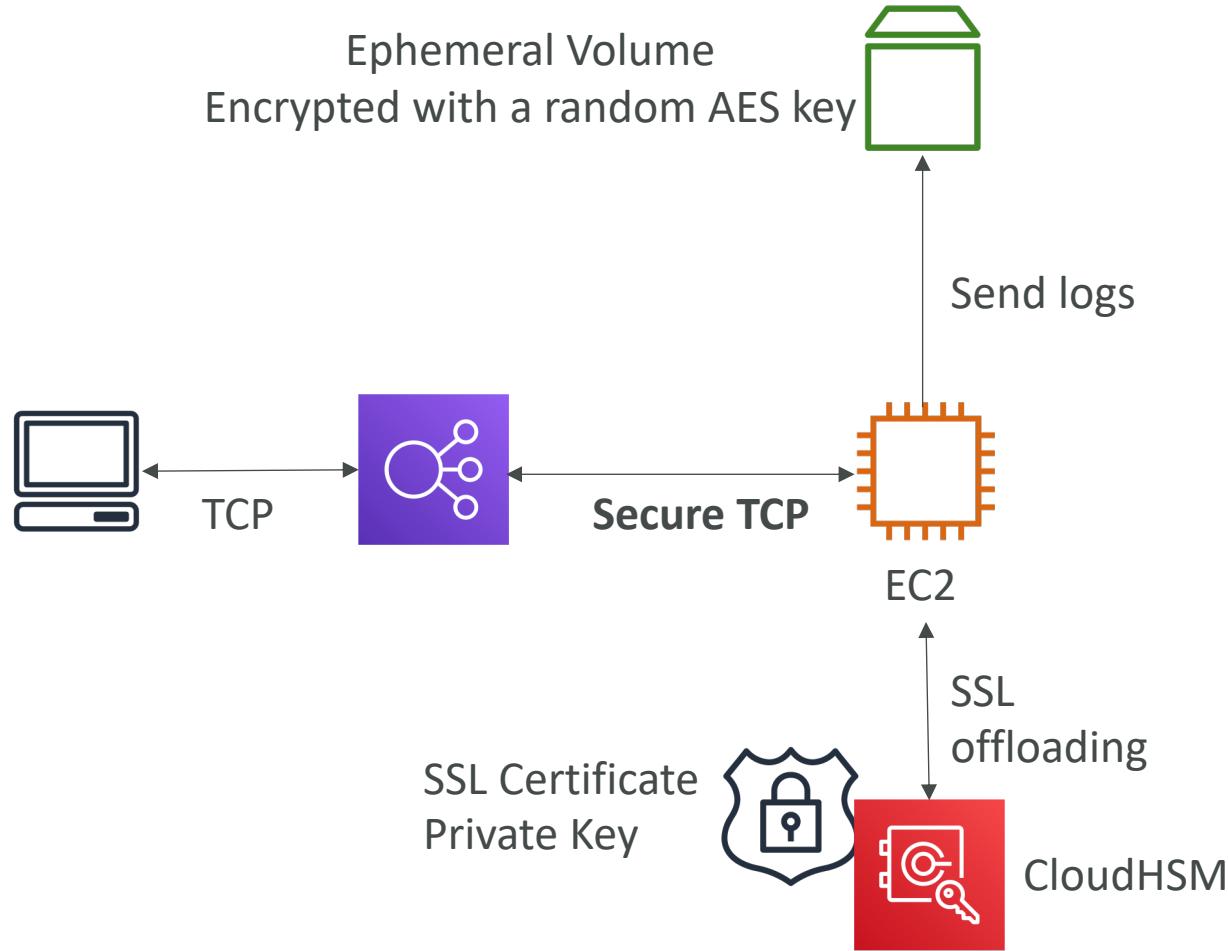
# Option C

- Use Elastic Load Balancing to distribute traffic to a set of web servers, configure the load balancer to perform TCP load balancing, use an AWS CloudHSM to perform the SSL transactions, and write your web server logs to a private Amazon S3 bucket using Amazon S3 server-side encryption.



# Option D

- Use Elastic Load Balancing to distribute traffic to a set of web servers. Configure the load balancer to perform TCP load balancing, use an AWS CloudHSM to perform the SSL transactions, and write your web server logs to an ephemeral volume that has been encrypted using a randomly generated AES key.

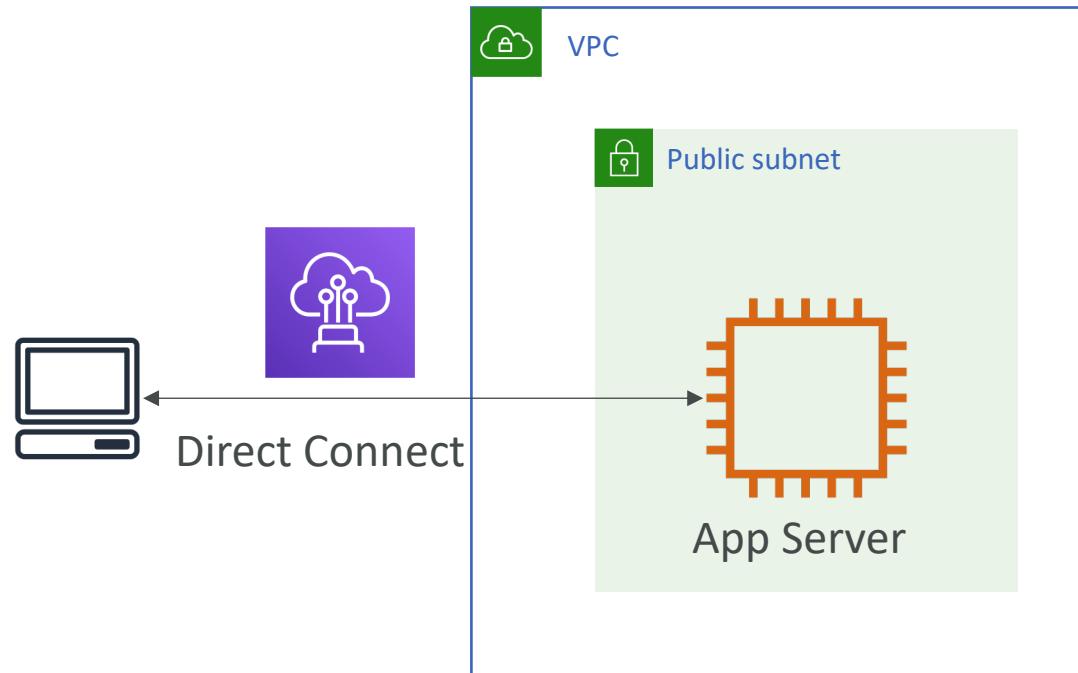


# Question 5

- You are designing network connectivity for your fat client application. The application is designed for **business travelers** who must be able to connect to it from their hotel rooms, cafes, public Wi-Fi hotspots, and **elsewhere on the Internet**. You do not want to publish the application on the Internet.
- Which network design meets the above requirements while minimizing deployment and operational costs?

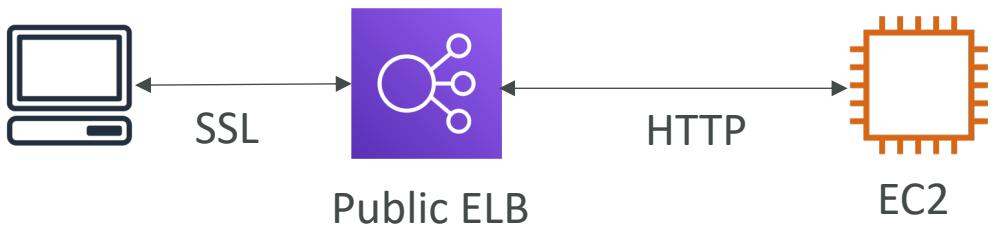
# Option A

- Implement AWS Direct Connect, and create a private interface to your VPC. Create a **public subnet** and place your application servers in it.



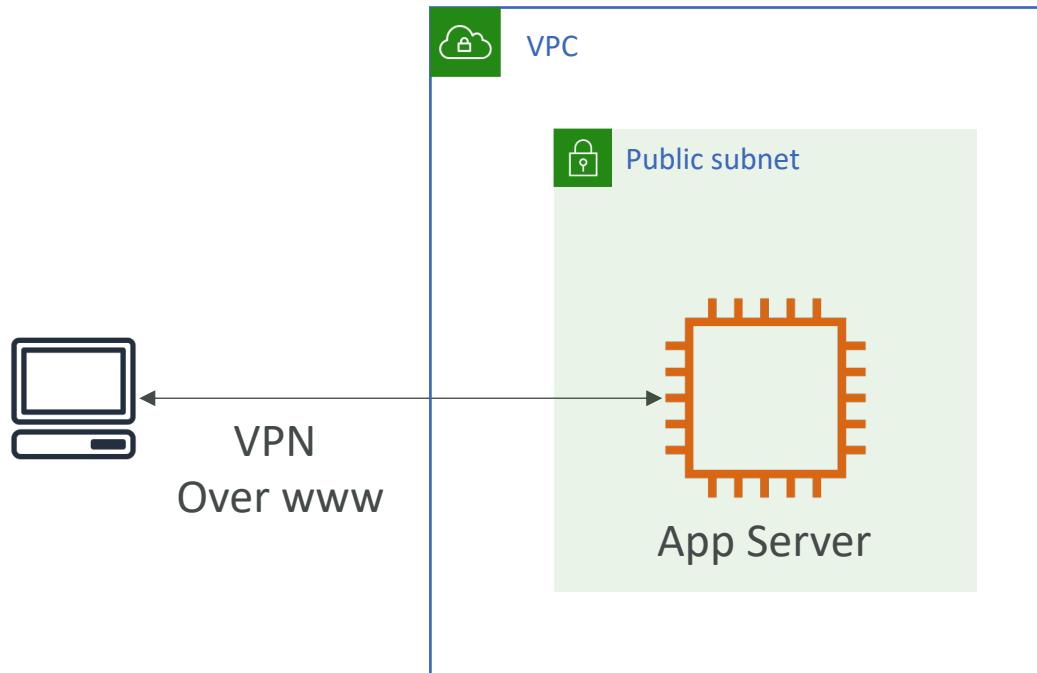
# Option B

- Implement Elastic Load Balancing with an SSL listener that terminates the back-end connection to the application.



# Option C

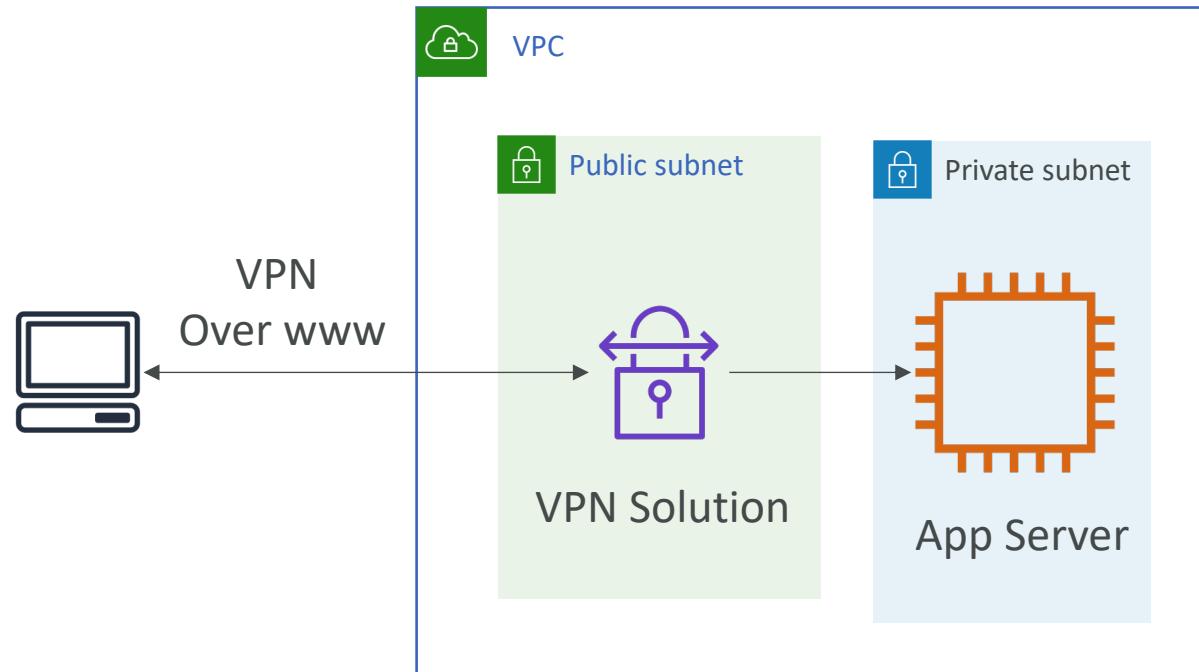
- Configure an IPsec VPN connection, and provide the users with the configuration details. Create a **public subnet** in your VPC, and place your application servers in it.



**CORRECT ANSWER**

# Option D

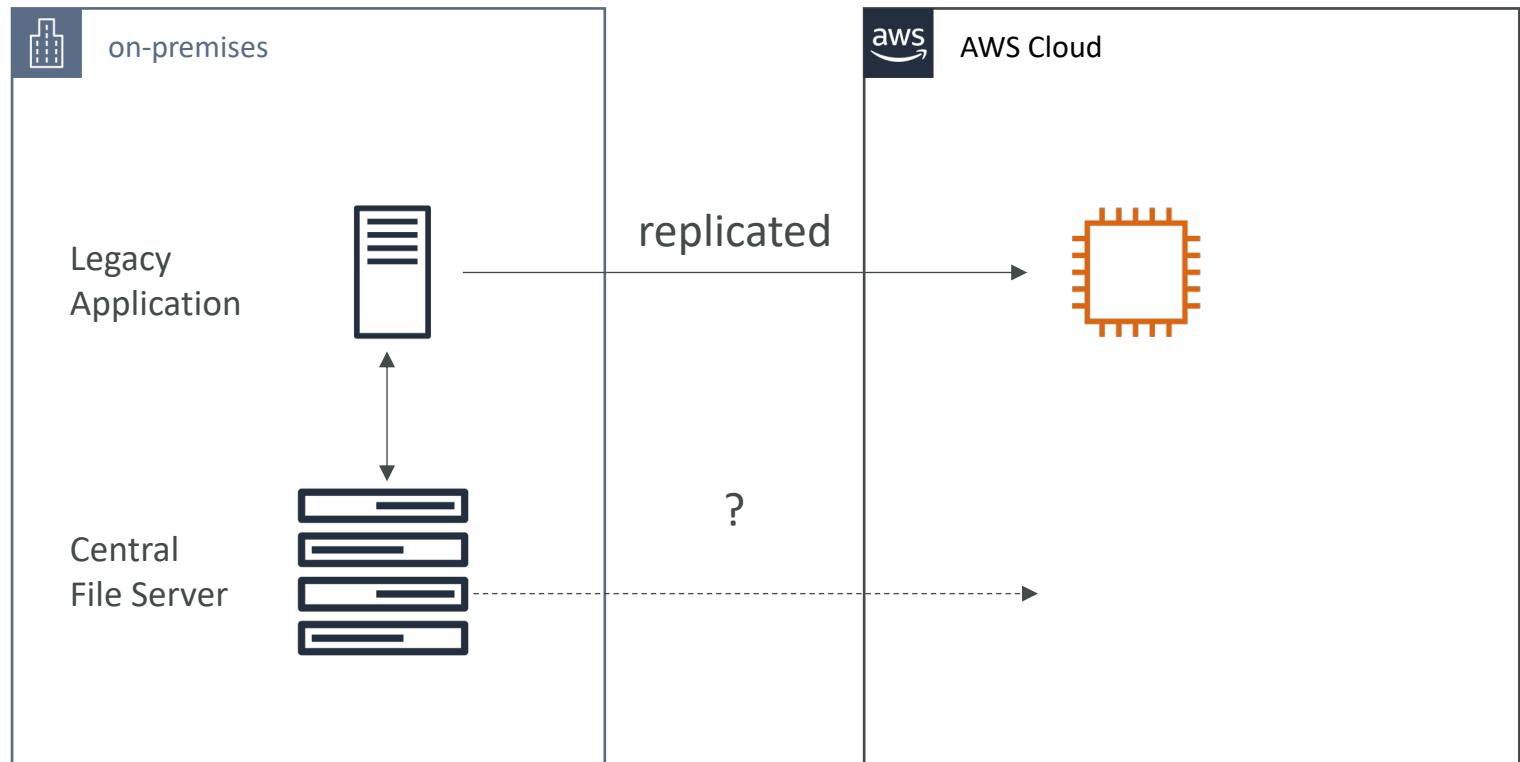
- Configure an SSL VPN solution in a public subnet of your VPC, then install and configure SSL VPN client software on all user computers. Create a **private subnet** in your VPC and place your application servers in it.



# Question 6

- Your company hosts an on-premises legacy engineering application with 900GB of data shared via a central file server. The engineering data consists of thousands of individual files ranging in size from megabytes to multiple gigabytes. Engineers typically modify **5-10 percent of the files a day**. Your CTO would like to migrate this application to AWS, but only if the application can be migrated over the weekend to minimize user downtime. You calculate that it will take a minimum of 48 hours to transfer 900GB of data using your company's existing **45-Mbps Internet connection**.
- After replicating the application's environment in AWS, which option will allow you to move the application's data to AWS **without losing any data** and **within the given timeframe**?

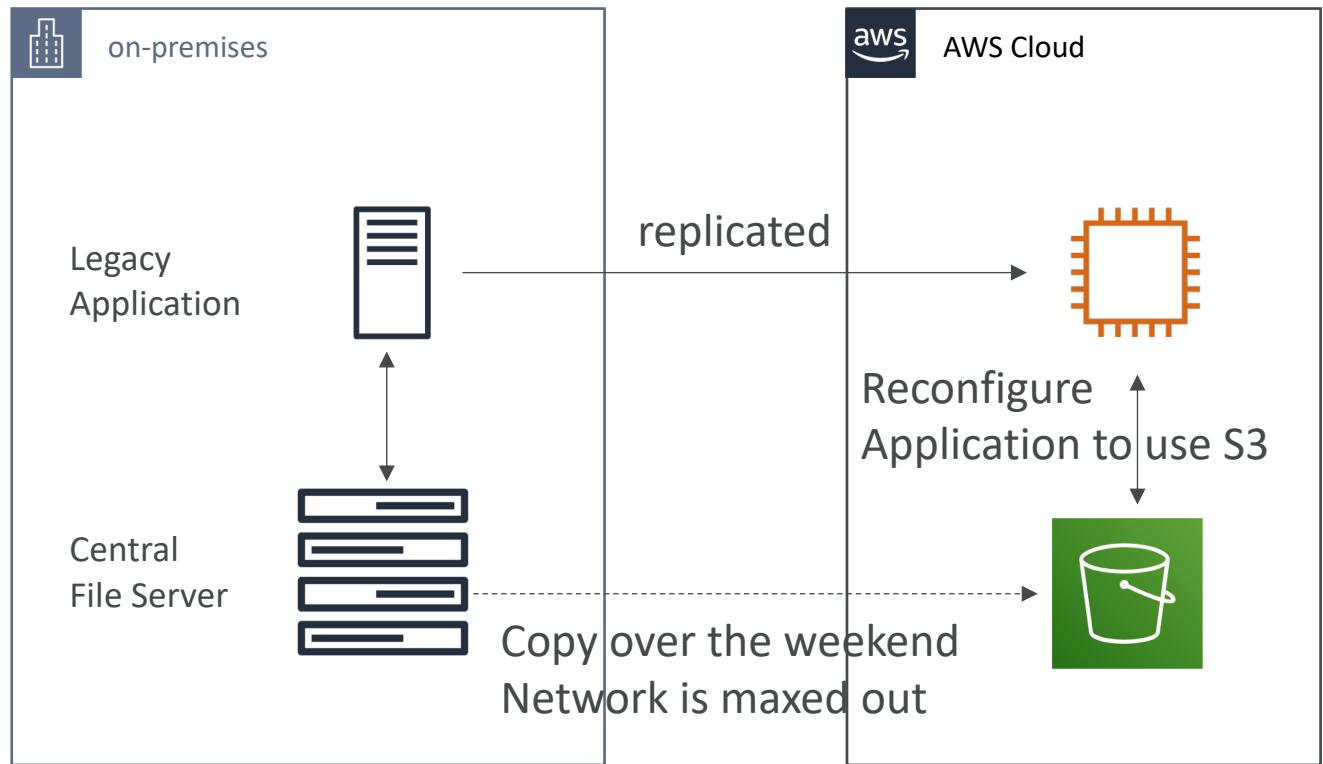
# Question 6 – Architecture Diagram



45 Mbps  
> 48 hours for 900 GB  
5-10% changing every day

# Option A

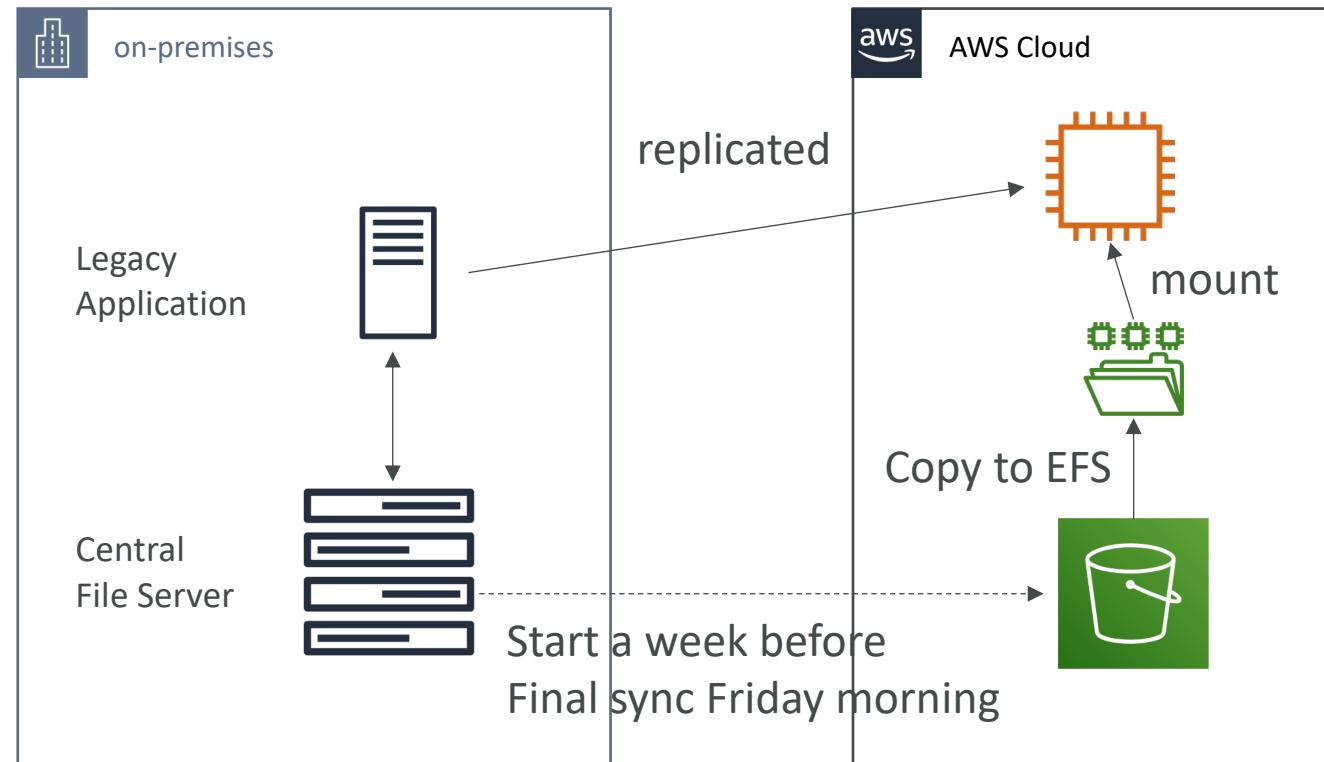
- Copy the data to Amazon S3 using multiple threads and multi-part upload for large files over the weekend, and work in parallel with your developers to reconfigure the replicated application environment to leverage Amazon S3 to serve the engineering files.



**CORRECT ANSWER**

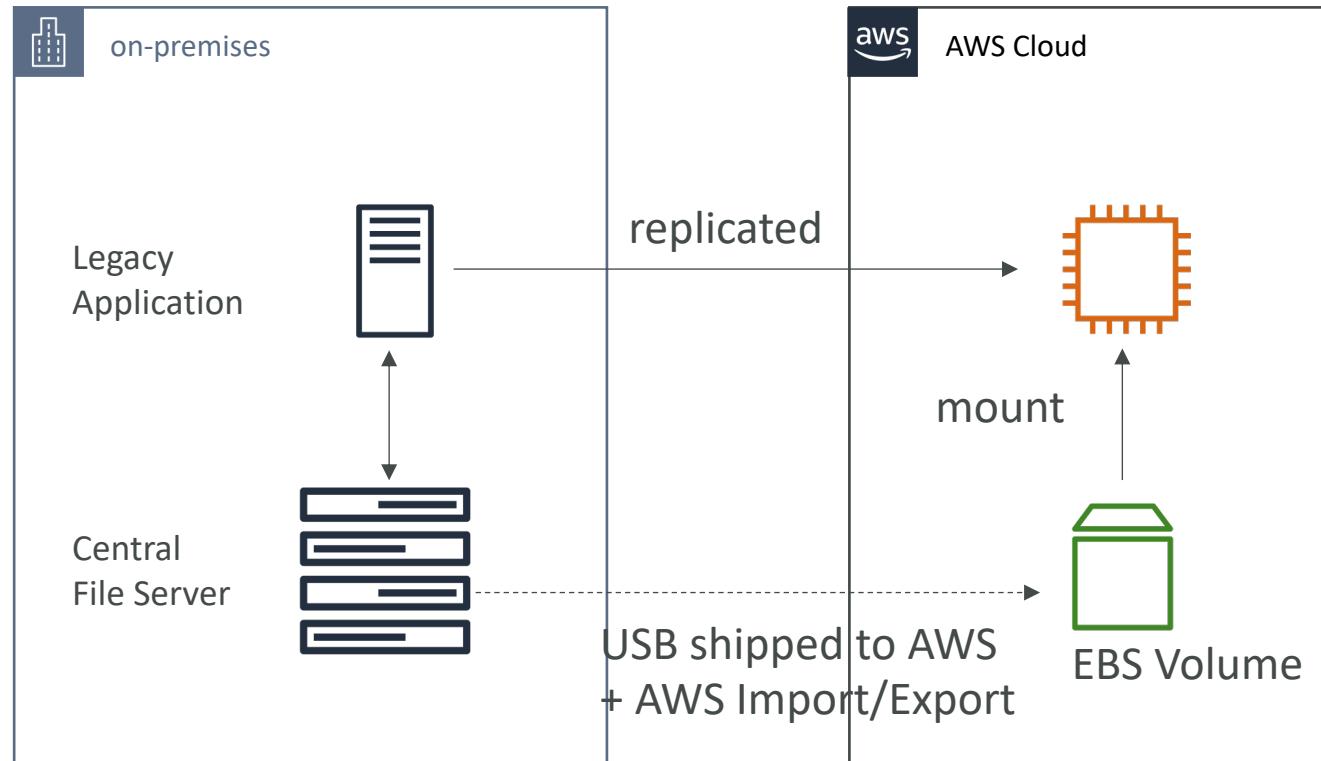
# Option B

- Sync the application data to Amazon S3 starting a week before the migration, on Friday morning perform a final sync, and copy the entire data set to your AWS file server after the sync completes.



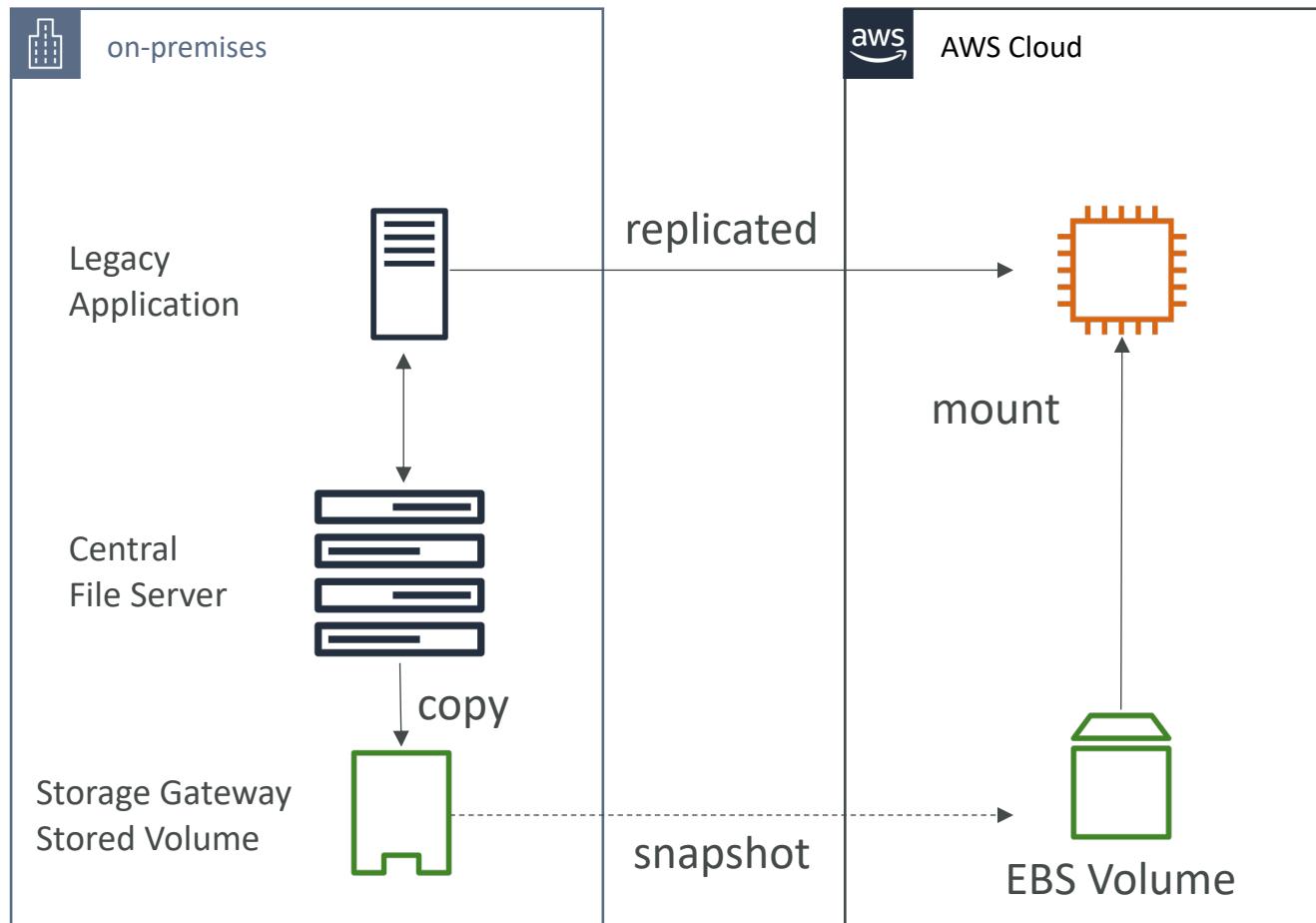
# Option C

- Copy the application data to a 1-TB USB drive on Friday and immediately send overnight, with Saturday delivery, the USB drive to AWS Import/Export to be imported as an EBS volume, mount the resulting EBS volume to your AWS file server on Sunday.



# Option D

- Leverage the AWS Storage Gateway to create a Gateway-Stored volume. On Friday copy the application data to the Storage Gateway volume. After the data has been copied, perform a snapshot of the volume and restore the volume as an EBS volume to be attached to your AWS file server on Sunday.



# Next steps

- Congratulations, you have covered all the domains!
- Make sure you revisit the lectures as much as possible
- A good extra resource to do is the AWS Exam Readiness course at:
  - <https://www.aws.training/Details/eLearning?id=34737>
- The AWS Certified SA Pro exam is hard, and tests experience...
- Make sure you master every single concept outlined in this course