



# AWS Cloud

# Account Creation

- 12 months free – Limited
- Will cost if used excessively



# Understanding of Physical and Virtual Servers



# What is Server?

- Normal computer or device who hosts website can be considered server.
  - Including data centers, offices



# Problems in Traditional

- **Cost of physical assets**
- **Requirement of power supply, place, cooling, maintenance**
- **Manpower needed**
- **24\*7 monitoring**
- **Rent of office, data centers**
- **Issue in scaling**
- **Disaster issues**



# Rise of Cloud Computing

- On demand delivery
- Pay as u go
- Choose your preference of machine
- Instant
- Go global
- One click



# Some services you already use

- Gmail
- Hotstar
- Netflix
- Dropbox



# Types of Cloud



```
graph TD; A[Types of Cloud] --> B[Public]; A --> C[Private]; A --> D[Hybrid];
```

## Public

That is available for everyone.

AWS, AZURE, GCP

## Private

- Not exposed to everyone
- Complete control in ur hand
- Security specific

## Hybrid

Public + Private

- Sensitive info in private

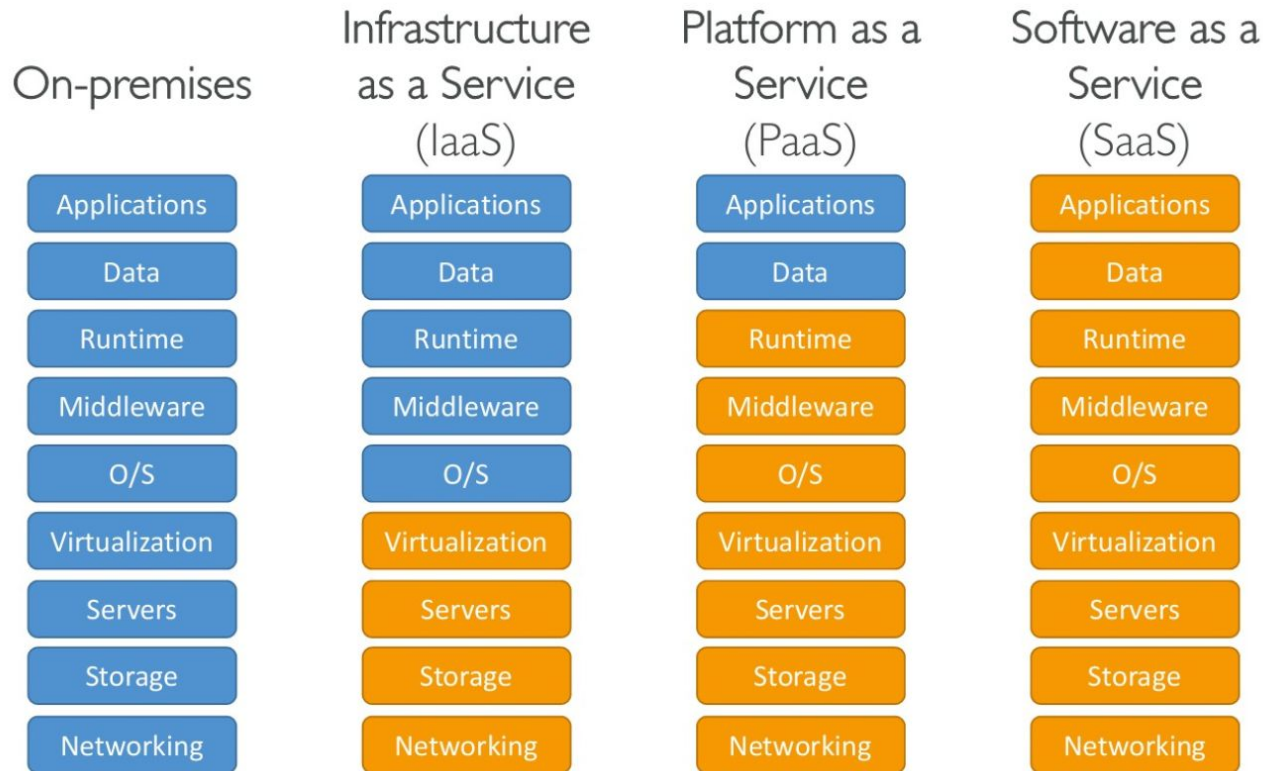


# Need/Benefits of Cloud

- **Flexibility**
- **Scalability**
- **Cost Effective**
- **High Availability**



# Cloud Computing Model



# Examples of Each

- **IaaS**
  - EC2
- **PaaS**
  - Elastic Beanstalk
- **SaaS**
  - DropBox, Gmail



# AWS Global Infra

- AWS Regions
- AWS AZ
- Edge Locations

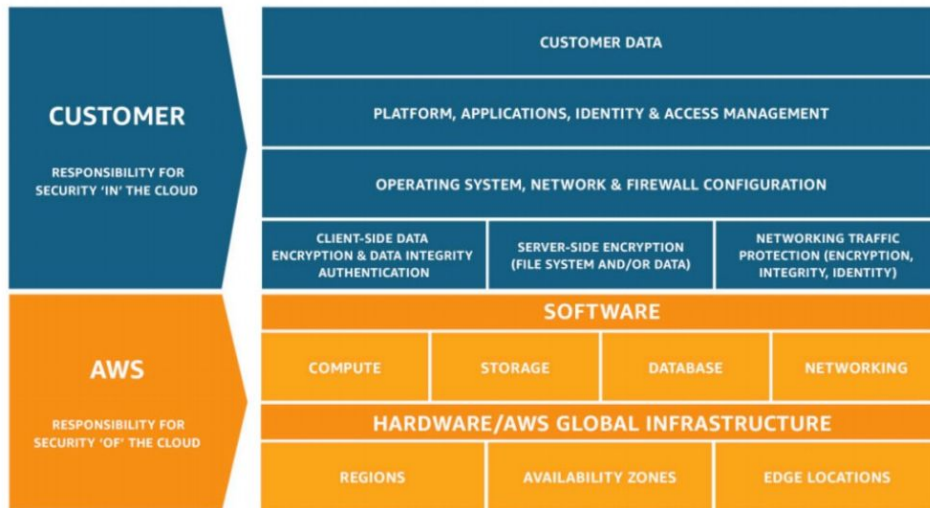


# Shared Responsibility Model

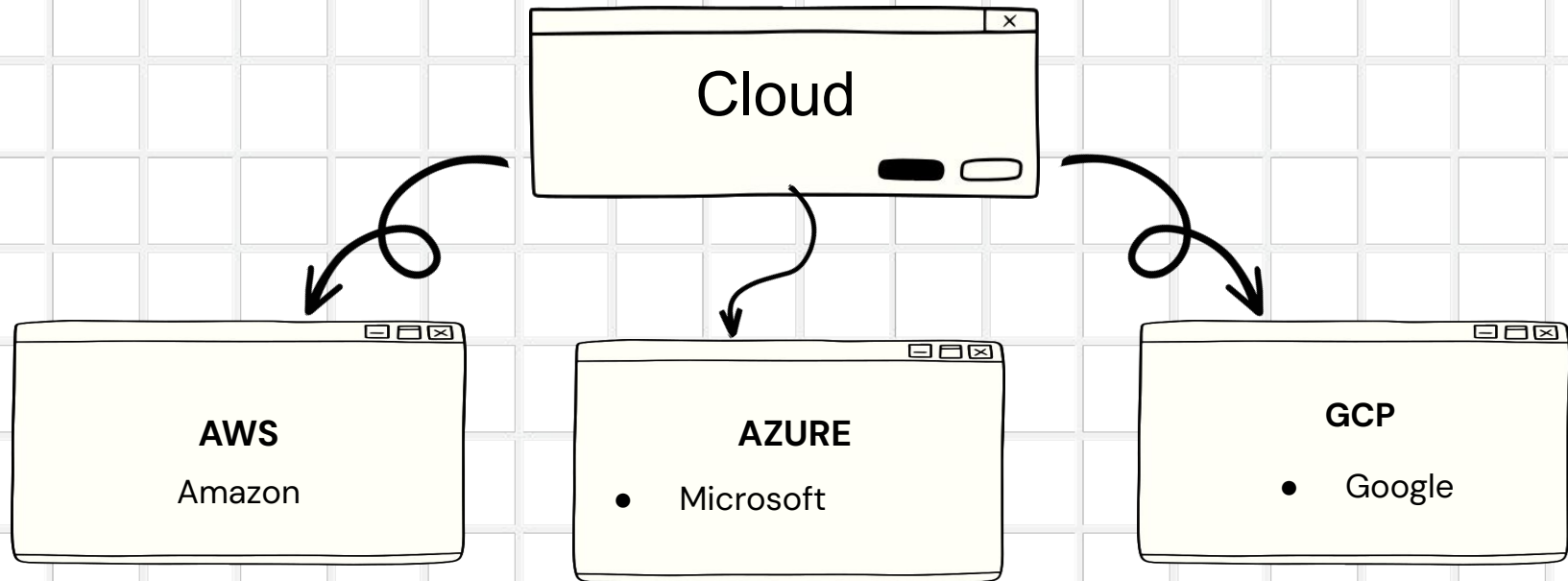
## Shared Responsibility Model diagram

CUSTOMER = RESPONSIBILITY FOR  
THE SECURITY IN THE CLOUD

AWS = RESPONSIBILITY FOR  
THE SECURITY OF THE CLOUD



# AWS vs Azure vs GCP



# AWS vs Azure vs GCP

- <https://cloud.google.com/docs/get-started/aws-azure-gcp-service-comparison>
- In 2019, AWS – 35\$ BILLION revenue
- 47% aws, 22%azure



# Pricing

- On aws account
- Cost calculator
- Billing
- budgets
- 





# Regional vs Global service

- **Global –**
  - **IAM**
  - **Organisations**
  - **Route 53**
  - **ACM**
  - **Cloudfront**



# IAM

- Identity and Access Management



# What is IAM

- Fine-grained control of who can do what
- Eg -user Bob can launch server

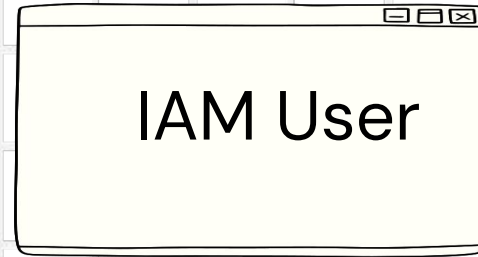
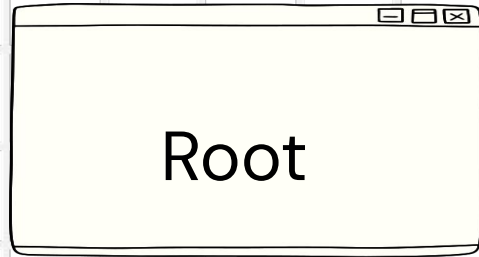


# IAM Characteristics

- free
- centralized AWS service
- default scope is AWS account
- deny by default



# IAM Users



# Root User

- Root User
  - the identity used to create AWS account
  - complete access
- Best practices
  - don't use this account for the everyday
  - setup physical MFA and lock it away
  - don't use your Amazon.com shopping account



# IAM User

- IAM Users
  - an identity with assigned permissions
  - can have username/password access to AWS console
  - can have (secret) key-based access to AWS APIs
- Best Practices
  - rotate credentials (keys, passwords)
  - MFA
  - password policy



# IAM Groups

- collection of IAM users
- operates like you'd think
- Best practices
  - manage permissions with groups
  - i.e., assign policies to groups instead of users





# IAM Policies

- set of permissions to be granted or denied
- JSON documents
- can be assigned directly to IAM users

```
{
  "Version": "2012-10-17",
  "Statement": [ {
    "Effect": "Allow",
    "Action": "s3:ListAllMyBuckets",
    "Resource": "arn:aws:s3::*"
  }, {
    "Effect": "Allow",
    "Action": [
      "s3:ListBucket",
      "s3:GetBucketLocation" ],
    "Resource":
      "arn:aws:s3:::EXAMPLE-BUCKET-NAME"
  }, {
    "Effect": "Allow",
    "Action": [
      "s3:PutObject",
      "s3:GetObject",
      "s3:DeleteObject" ],
    "Resource":
      "arn:aws:s3:::EXAMPLE-BUCKET-NAME/*"
  } ] }
```

# IAM Role

- a 2<sup>nd</sup> type of AWS identity
  - also has assigned permissions
  - similar to IAM users
- designed to be temporarily assumed
  - e.g. by an EC2 instance
- no associated credentials
- Instance Profiles
  - assigned to EC2 instance
  - container for one or more IAM roles



# Best Practice

- **Users** – Create individual users.
- **Permissions** – Grant least privilege.
- **Groups** – Manage permissions with groups.
- **Conditions** – Restrict privileged access further with conditions.
- **Password** – Configure a strong password policy.
- **Rotate** – Rotate security credentials regularly.
- **MFA** – Enable MFA for privileged users.
- **Roles** – Use IAM roles for Amazon EC2 instances.
- **Root** – Reduce or remove use of root.



# EC2

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable compute capacity in the cloud.

Access reliable, scalable infrastructure on demand. Scale capacity within minutes with SLA commitment of 99.99% availability.



# Benefits of EC2

- \* **Scalability:** Easily **scale up or down** resources based on demand.
- \* **Flexibility:** Choose from various **instance types** optimized for different workloads (compute, memory, GPU).
- \* **Cost-effectiveness:** **Pay-as-you-go** pricing model for only the resources you use.
- \* **Global Availability:** Available in multiple **AWS regions** worldwide.



# Different types of EC2

<https://aws.amazon.com/ec2/instance-types/>



# Size n Conf of EC2

- OS
- CPU
- RAM
- Space
- Network Card
- Firewall



# Use Cases of EC2

- \* Hosting websites and applications.
- \* Running batch jobs.
- \* Building and deploying cloud-native applications.
- \* Setting up development, testing, and staging environments.





# Key Pair

- AWS uses public-key cryptography to encrypt and decrypt login information.
- AWS only stores the public key, and the user stores the private key.



# Generate Key Pair

1. Open the Amazon EC2 console at <http://console.aws.amazon.com/ec2/>
2. On the navigation bar select region for the key pair
3. Click **Key Pairs** in the navigation pane to display the list of key pairs associated with the account
4. Click **Create Key Pair**
5. Enter a name for the key pair in the **Key Pair Name** field of the dialog box and click **Create**
6. The private key file, with .pem extension, will automatically be downloaded by the browser.



# Steps of creating EC2

Step 1: Sign up for Amazon EC2

Step 2: Create a key pair

Step 3: Launch an Amazon EC2 instance

Step 4: Connect to the instance

Step 5: Customize the instance

Step 6: Terminate instance and delete the volume created



# Connecting to EC2

- There are several ways to connect to an EC2 instance once it's launched.
- **Remote Desktop Connection** is the standard way to connect to Windows instances.
- An **SSH client** (standalone or web-based) is used to connect to Linux instances.



# Features of EC2

- Virtual Computing Environments, known as instances
- Preconfigured template for your instance is known as AMI
- Various configuration of CPU, memory, storage and network capacity is Instance type
- Secure login information for your instance using Key pairs



# More Features of EC2

- Storage volumes for temporary data that's deleted when hardware fails or terminate your instance, known as instance store volumes
- Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as Amazon EBS volumes
- A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using security groups
- Metadata, known as tags, that you can create and assign to your Amazon EC2 resources



# Intro to SG

A security group in the context of Amazon EC2 is essentially a virtual firewall that controls the traffic for one or more instances.

It acts as a virtual firewall for your EC2 instances to control incoming and outgoing traffic.



# Key Points – SG

1. Traffic Control
2. Stateful
3. Flexible rule
4. Layer of Defense
5. Only allowed rules





# EC2 Purchasing Option

1. On-Demand
2. Reserved
3. Spot



# EC2 Purchasing Option

## On-Demand

- Charged per second or hour of usage, offering maximum flexibility.
- **No upfront commitment:** Ideal for short-term workloads, testing, or unpredictable usage patterns.
- **Scalability:** Easily scale instances up or down based on real-time needs.
- **Availability:** Guaranteed instance availability within your chosen AWS region.



# EC2 Purchasing Option

## Reserved

- Offer significant discounts (up to 75%) compared to On-Demand pricing through a prepaid reservation for a specific instance type, region, and term (1 or 3 years).
- **Benefits:**
  - **Significant cost savings** for predictable, sustained workloads.
  - **Guaranteed capacity:** Ensures availability of the specified instance type during your reservation term.



# EC2 Purchasing Option

## Spot:

- Bid on spare EC2 capacity at significantly lower prices (up to 90% discount) compared to On-Demand Instances. The price fluctuates based on supply and demand.
- **Benefits:**
  - **Lowest cost option** for workloads that can tolerate interruptions.



# EBS volume

- EBS is drive that you attach when you run the machine/instance.
- Persist even after machine termination
- Can be mounted to one instance at a particular time
- Bound to AZ.



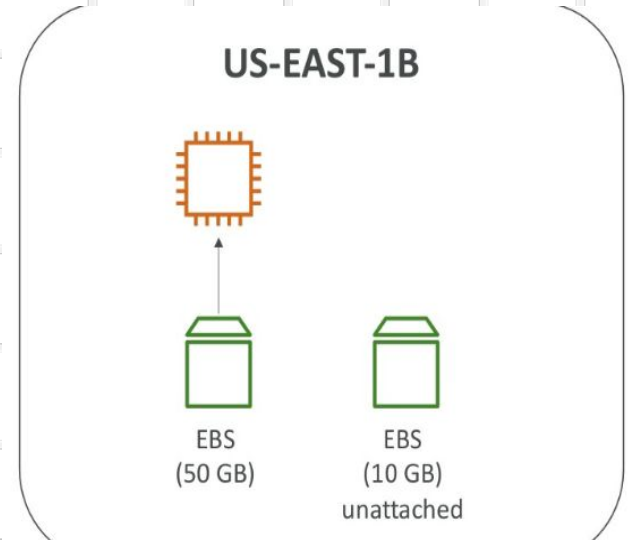
# EBS volume- Benefits

- Data Persistence
- Scalability
- Backup and Recovery
- High Availability
- Performance



# EBS volume- Key Info

- Restricted to 1 AZ
- It's a N/W drive
- Costly as compared to other Storage services



# EBS volume- Attach

- When you just wanna take the add volume which is available in nature.





# EBS volume- Detach

- When you just wanna take the backup of data with volume being available then you can detach the volume.



# EBS volume- Size Increase

- You can change size of EBS volume
- Only INCREASE possible
- Not decrease
- 1:1



# Types of EBS volumes

Amazon EBS provides the following volume types, which differ in performance characteristics and price.

- SSD-backed volumes optimized for transactional workloads involving frequent read/write operations with small I/O size, where the dominant performance attribute is IOPS
- HDD-backed volumes optimized for large streaming workloads where throughput (measured in MiB/s) is a better performance measure than IOPS

# Snapshot

- It acts like a frozen image of your data at a specific moment, allowing you to restore your data or create new EBS volumes from that saved state.
- Make a backup of EBS volume at that point of time
- Can be shared among AZ or Region.



# Snapshot – Benefits

- Data Backup and Recovery
- Disaster Recovery
- Data Archiving
- Data Migration



# AMI

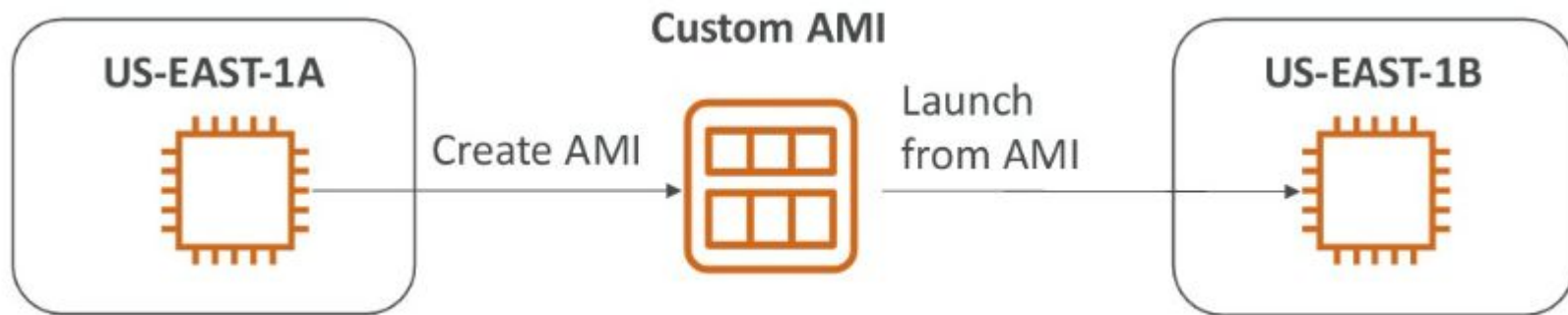
In Amazon Web Services (AWS), an Amazon Machine Image (AMI) acts as a template for creating virtual servers known as EC2 (Elastic Compute Cloud) instances. It essentially encapsulates the configuration of a server, including the operating system, applications, and settings.

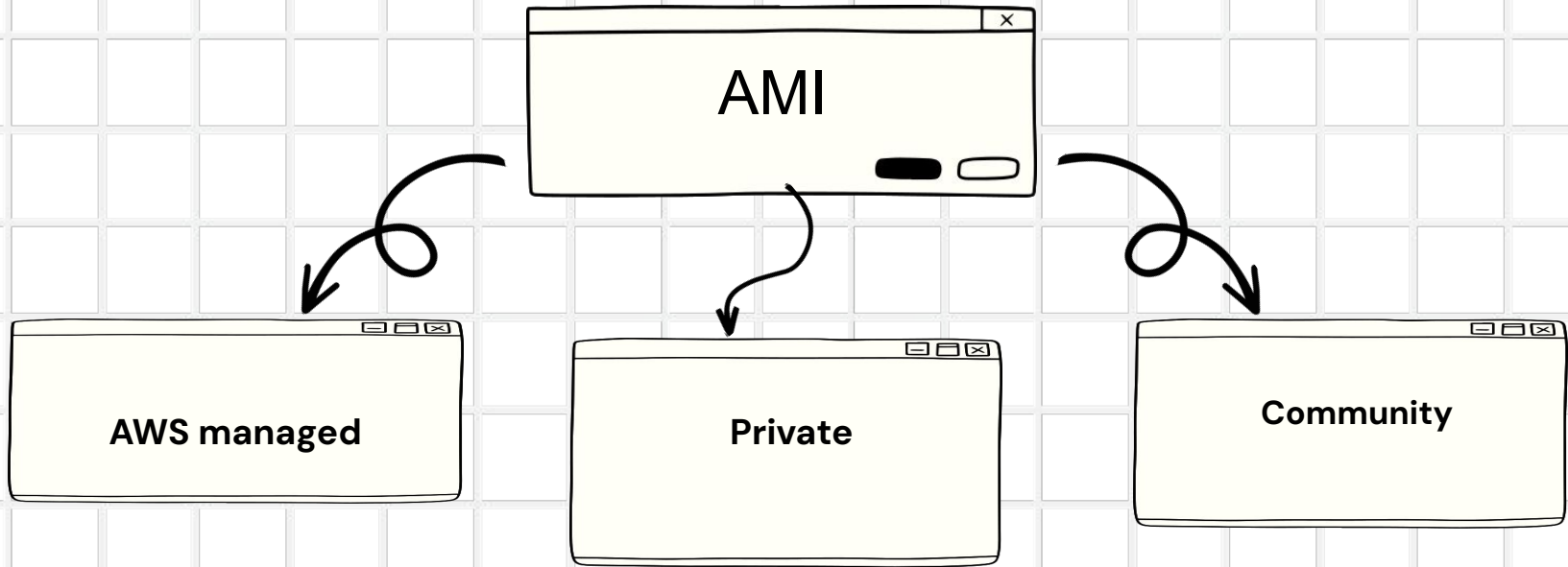


Amazon Machine  
Image (AMI)

# Benefits

- **Faster Deployments**
- **Consistency**
- **Repeatability**
- **Improved Manageability**







# Finding an Instance type

- Region
- The architecture: 32-bit (i386), 64-bit (x86\_64)
- Compute
- Memory
- Storage
- Network performance



# ○ Changing the Instance type

- As your needs change, you might find that your instance is over-utilized or underutilized.
- For example, if your t2.micro instance is too small for its workload, you can change it to another instance type that is appropriate for the workload.
- You might also want to migrate from a previous generation instance type to a current generation instance type to take advantage of some features; for example, support for IPv6.



# Placement Groups

You can launch or start instances in a placement group, which determines how instances are placed on underlying hardware. When you create a placement group, you can create one of the following strategies for the group:

- Cluster – clusters instances into a low-latency group in a single Availability Zone
- Partition – spreads instances across logical partitions, ensuring that instances in one partition do not share underlying hardware with instances in other partitions
- Spread – spreads instances across distinct underlying hardware



Availability Zone



Availability Zone 1

Partition 1



Partition 2

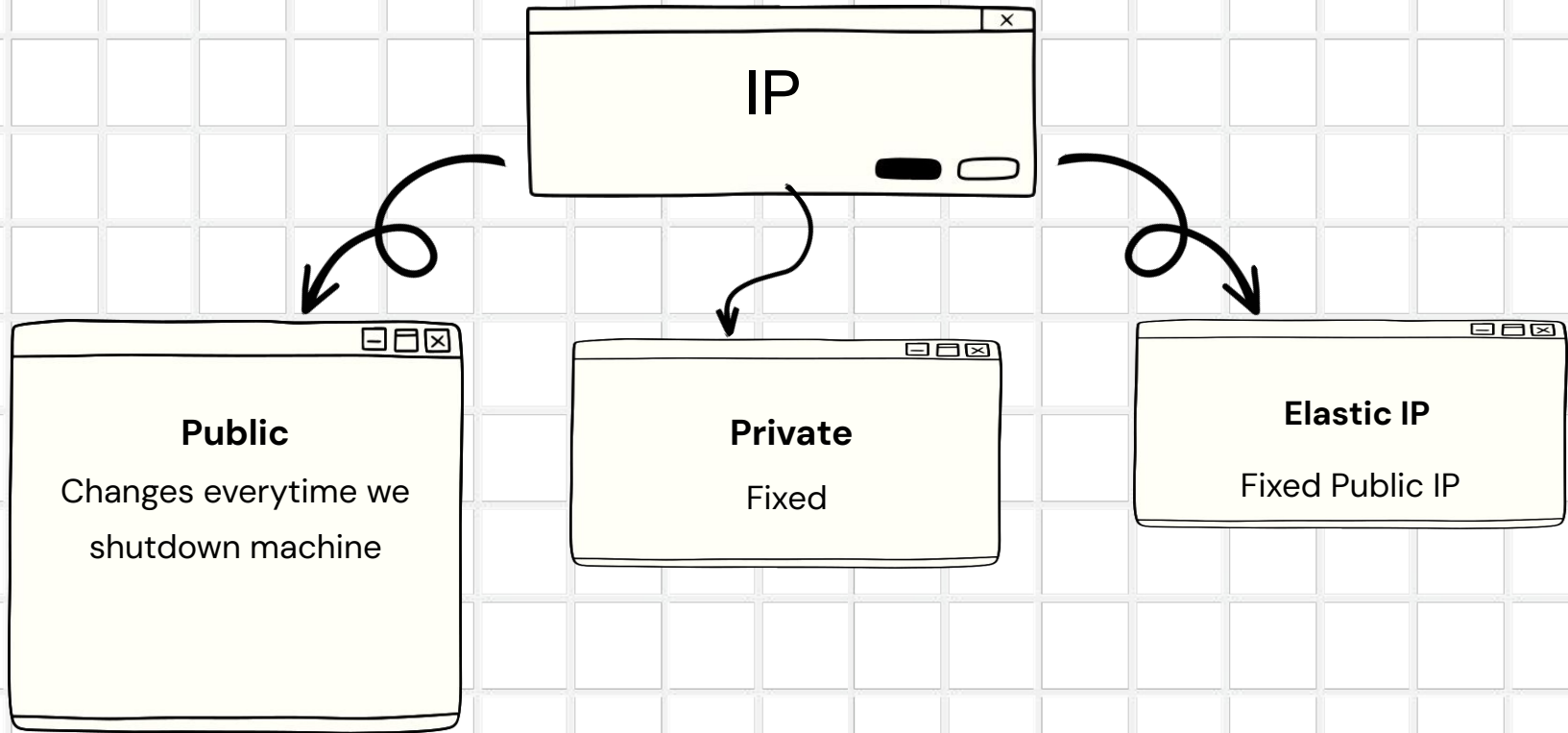


Partition 3



Availability Zone 1





# Public IP

- A public IP address is an IPv4 address that's reachable from the Internet. You can use public addresses for communication between your instances and the Internet.
- Each instance that receives a public IP address is also given an external DNS hostname; for example, `ec2-203-0-113-25.compute-1.amazonaws.com`.
- Changes on restart

# Private IP

When EC2 instances are launched, the primary IP is assigned a reserved private IP address

- The private IP address stays assigned to the network interface until it is deleted.
- It is not possible to remove or change the private IP address of the primary network interface

# Elastic IP

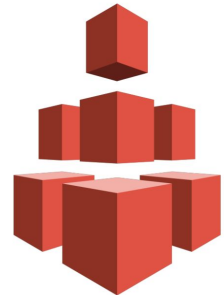
An Elastic IP address is a static IPv4 address designed for dynamic cloud computing.

- An Elastic IP address is associated with your AWS account.
- With an Elastic IP address, you can mask the failure of an instance or software by rapidly remapping the address to another instance in your account.
- An Elastic IP address is a public IPv4 address, which is reachable from the internet.



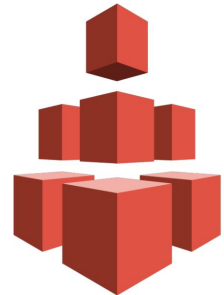
# EFS

- Amazon EFS provides a simple, scalable, elastic file system for Linux-based workloads for use with AWS Cloud services and on-premises resources.
- It is built to scale on demand to petabytes without disrupting applications, growing and shrinking automatically as you add and remove files
- It is designed to provide massively parallel shared access to thousands of Amazon EC2 instances
- It is a fully managed service



# EFS

- There is a Standard and an Infrequent Access storage class available with Amazon EFS using Lifecycle Management, files not accessed for 30 days will automatically be moved to a cost-optimized Infrequent Access storage class reducing cost up to 85%
- Amazon EFS is a regional service storing data within and across multiple Availability Zones (AZs) for high availability and durability.



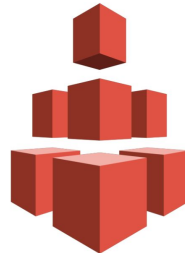
# EFS vs EBS

## Scalability:

- **EBS:** You can scale EBS volumes by attaching additional volumes to your EC2 instance or using larger volume sizes. However, scaling primarily involves individual instance storage capacity.
- **EFS:** EFS is inherently scalable. You can easily increase or decrease the storage capacity of your file system on demand to accommodate growing data needs.

## Availability:

- **EBS:** EBS volumes are designed to be highly available within an Availability Zone. However, they are directly attached to a specific EC2 instance.
- **EFS:** EFS provides higher availability through its distributed architecture across multiple Availability Zones within a region. This ensures that file system access remains available even if a single Availability Zone encounters an outage.



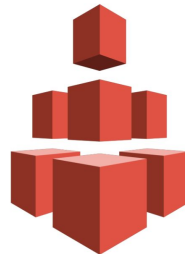
# EFS vs EBS

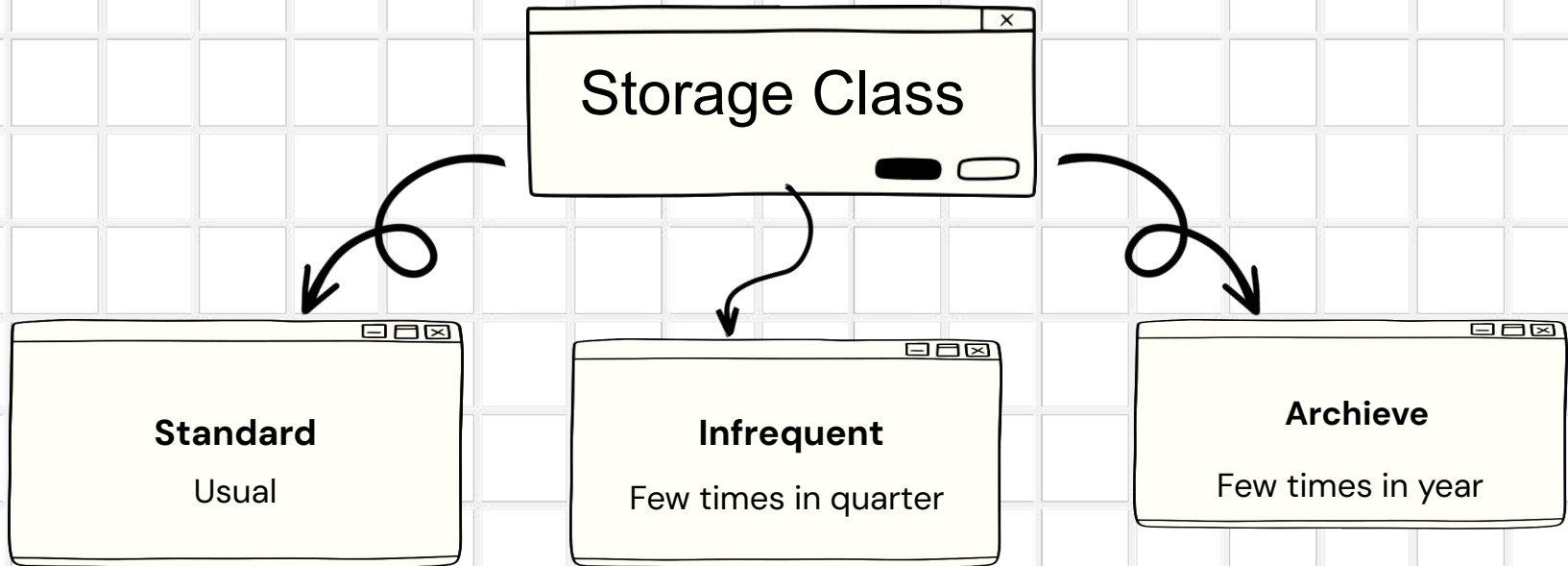
## Cost:

- **EBS:** EBS billing is based on the provisioned storage size (per GiB per month) and the type of volume (e.g., magnetic, SSD, NVMe SSD).
- **EFS:** EFS billing is based on the amount of data stored (per GiB per month) and the number of concurrent file operations (IOPS).

## Performance:

- **EBS:** EBS offers high performance, making it suitable for applications requiring fast storage access, such as databases or applications working with large datasets.
- **EFS:** While EFS provides good performance, it might not match the raw speed of EBS due to its distributed nature. However, EFS scales well for concurrent access from multiple instances.

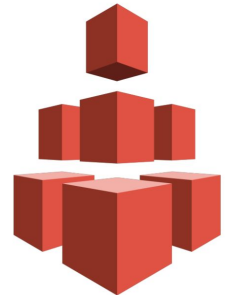




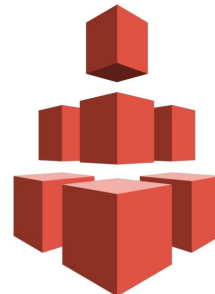
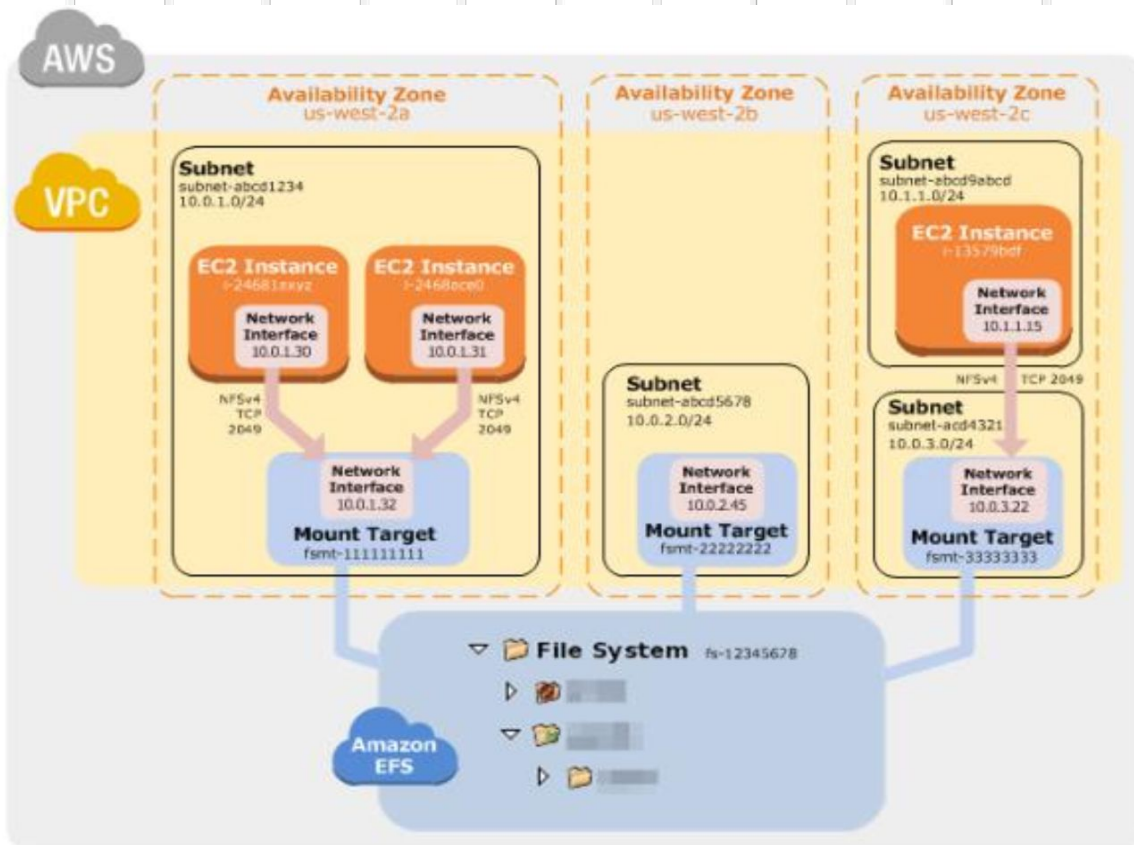
# EFS

- **Important:**

Amazon EFS is not supported on Windows instances.

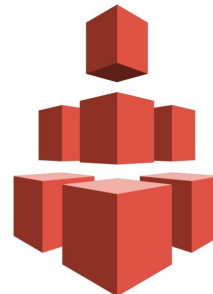


# EFS



# EFS-Benefits

- Dynamic elasticity
- Scalable performance
- Shared file storage
- Fully managed
- Cost effective – pay for what you use (no upfront capacity planning)
- Pricing

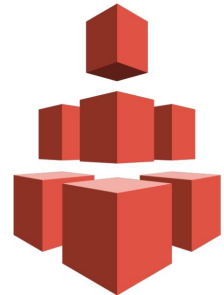




# EFS-Use Cases

Amazon EFS is designed to meet the performance needs of the following use cases.

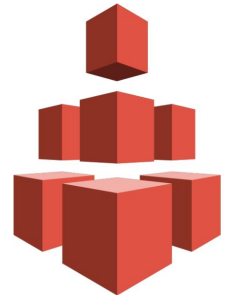
- Big Data and Analytics
- Media Processing Workflows
- Content Management and Web Serving
- Home Directories



# EFS-Use Cases

Amazon EFS is designed to meet the performance needs of the following use cases.

- Big Data and Analytics
- Media Processing Workflows
- Content Management and Web Serving
- Home Directories



# S3

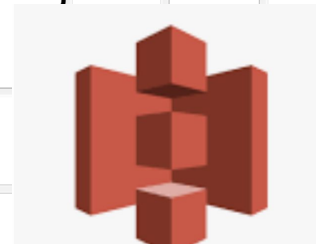
- Amazon Simple Storage Service is storage for the Internet
- Amazon S3 has a simple web services interface that you can use to store and retrieve any amount of data, at any time, from anywhere on the web.



# S3– Advantages

Amazon S3 is intentionally built with a minimal feature set that focuses on simplicity and robustness

- Creating buckets – Create and name a bucket that stores data. Buckets are the fundamental container in Amazon S3 for data storage.
- Storing data – Store an infinite amount of data in a bucket. Upload as many objects as you like into an Amazon S3 bucket. Each object can contain up to 5 TB of data.

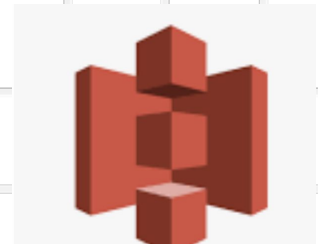


# S3 – Advantages Cont.

- **Downloading data** – Download your data or enable others to do so.

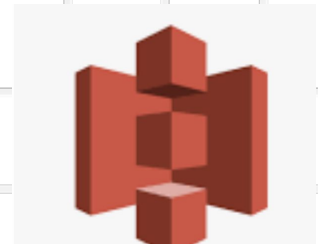
Download your data anytime you like, or allow others to do the same.

- **Permissions** – Grant or deny access to others who want to upload or download data into your Amazon S3 bucket.



# S3 – Use Cases

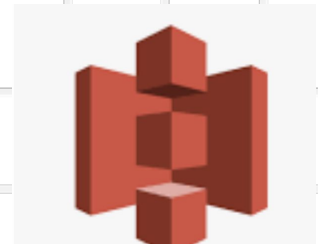
- Backup and Storage
- Disaster Recovery
- Archive
- Application Hosting
- Static Website



# S3- Concepts

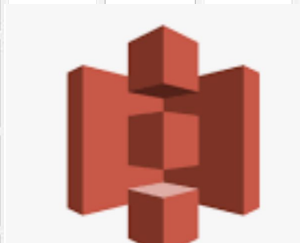
## Buckets

- To upload your data (photos, videos, documents etc.) to Amazon S3, you must first create an S3 bucket in one of the AWS Regions.
- A bucket is a region specific
- A bucket is a container for objects stored in Amazon S3.





# S3 – Concepts Cont.



- Every object is contained in a bucket.
- By default, you can create up to 100 buckets in each of your AWS accounts. If you need more buckets, you can increase your account bucket limit to a maximum of 1,000 buckets by submitting a service limit increase.
- For example, if the object named photos/puppy.jpg is stored in the john bucket in the US West (Oregon) Region, then it is addressable using the URL <https://john.s3.us-west-2.amazonaws.com/photos/puppy.jpg>



# S3– Buckets



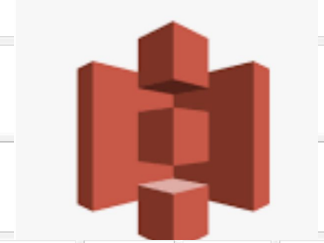
- **For Bucket name to be created, follow the naming guidelines**
  - Bucket name should be globally unique and the namespace is shared in all accounts. This means that after a bucket is created, the name of that bucket cannot be used by another AWS account in any AWS Region until the bucket is deleted.
  - Once created it cannot be changed

# S3– Buckets



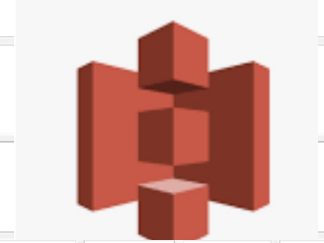
- Bucket names must be at least 3 and no more than 63 characters long.
- Bucket names must not contain uppercase characters or underscores.
- Bucket names must start with a lowercase letter or number.
- Bucket names must not be formatted as an IP address (for example, 192.168.5.4).
- After you create the bucket, you cannot change the name, so choose wisely.
- Choose a bucket name that reflects the objects in the bucket because the bucket name is visible in the URL that points to the objects that you're going to put in your bucket.

# S3– Regions



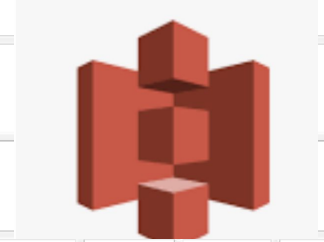
- You can choose the geographical AWS Region where Amazon S3 will store the buckets that you create.
- You might choose a Region to optimize latency, minimize costs, or address regulatory requirements.
- Objects stored in a Region never leave the Region unless you explicitly transfer them to another Region.
- For example, objects stored in the Europe (Ireland) Region never leave it.

# S3– Objects



- Amazon S3 is a simple key, value store designed to store as many objects as you want.
- You store these objects in one or more buckets.
- S3 supports object level storage i.e., it stores the file as a whole and does not divide them
- An object size can be in between 0 KB and 5 TB
- When you upload an object in a bucket, it replicates itself in multiple availability zones in the same region

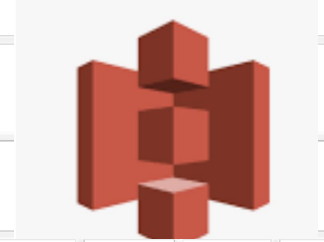
# S3 – Objects Cont.



An object consists of the following:

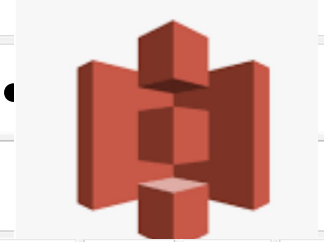
- Key – The name that you assign to an object.
- Version ID – Within a bucket, a key and version ID uniquely identify an object.
- Value – The content that you are storing.
- Metadata – A set of name-value pairs with which you can store information regarding the object.

# S3– Objects Versioning



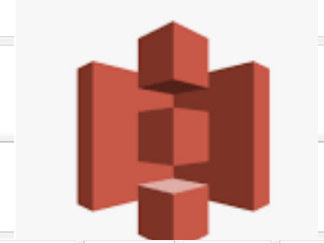
- When you re-upload the same object name in a bucket, it replaces the whole object
- You can use versioning to keep multiple versions of an object in one bucket.
- For example, you could store my-image.jpg (version 1) and my-image.jpg (version 2) in a single bucket.
- Versioning protects you from the consequences of unintended overwrites and deletions.
- You must explicitly enable versioning on your bucket. By default, versioning is disabled.

# Objects Versioning Cont.



- Regardless of whether you have enabled versioning, each object in your bucket has a version ID.
- If you have not enabled versioning, Amazon S3 sets the value of the version ID to null. If you have enabled versioning, Amazon S3 assigns a unique version ID value for the object.
- This functionality prevents you from accidentally overwriting or deleting objects and affords you the opportunity to retrieve a previous version of an object.

# S3 logs

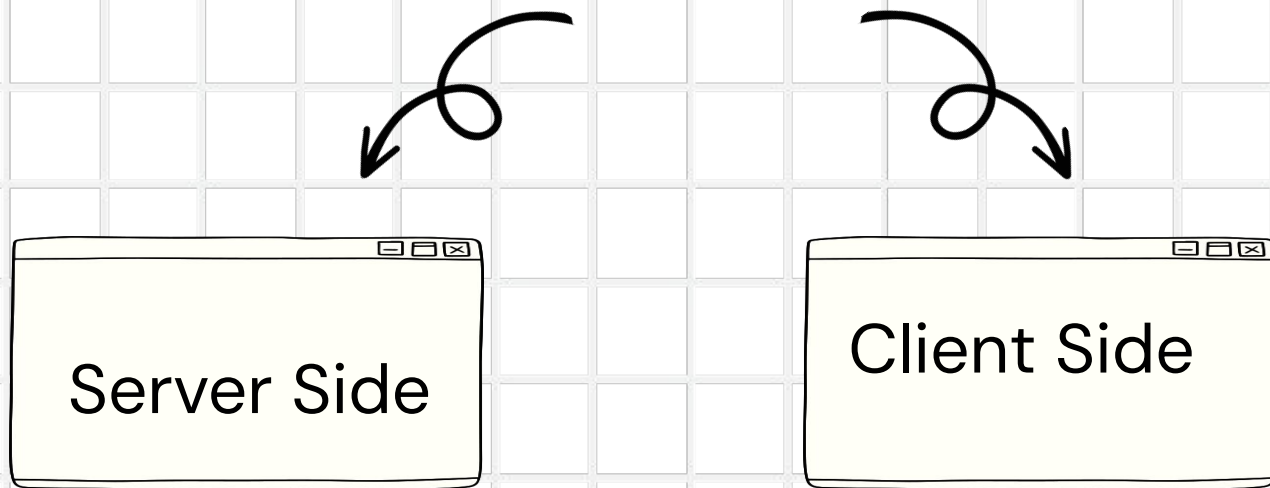


## Server Access Logging

- Server access logging provides detailed records for the requests that are made to a bucket. Server access logs are useful for many applications.
- For example, access log information can be useful in security and access audits.
- Each access log record provides details about a single access request, such as the requester, bucket name, request time, request action, response status, and an error code, if relevant.
- Both the source and target S3 buckets must be owned by the same AWS account, and the S3 buckets must both be in the same Region.



# S3 Encryption



# S3 Encryption



Data protection refers to protecting data while in-transit (as it travels to and from Amazon S3) and at rest (while it is stored on disks in Amazon S3 data centers). You have the following options for protecting data at rest in Amazon S3:

- Server-Side Encryption – Request Amazon S3 to encrypt your object before saving it on disks in its data centers and then decrypt it when you download the objects.
- Client-Side Encryption – Encrypt data client-side and upload the encrypted data to Amazon S3. In this case, you manage the encryption process, the encryption keys, and related tools.

# S3 Server Side Encryption



## Server Side Encryption

- Server-side encryption is the encryption of data at its destination by the application or service that receives it.
- Amazon S3 encrypts your data at the object level as it writes it to disks in its data centers and decrypts it for you when you access it.
- As long as you authenticate your request and you have access permissions, there is no difference in the way you access encrypted or unencrypted objects.

# SSE-S3



## Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)

- When you use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3), each object is encrypted with a unique key.
- As an additional safeguard, it encrypts the key itself with a master key that it regularly rotates.

# SSE-KMS



Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS)

- Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS) is similar to SSE-S3, but with some additional benefits and charges for using this service.

# SSE-KMS



Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS)

- Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS) is similar to SSE-S3, but with some additional benefits and charges for using this service.

# SSE-C



Server-Side Encryption with Customer-Provided Keys (SSE-C)

- With Server-Side Encryption with Customer-Provided Keys (SSE-C), you manage the encryption keys and Amazon S3 manages the encryption, as it writes to disks, and decryption, when you access your objects.

# Client Side Encryption



Client-side encryption is the act of encrypting data before sending it to Amazon S3. To enable client-side encryption, you have the following options:

- Use a customer master key (CMK) stored in AWS Key Management Service (AWS KMS).
- Use a master key you store within your application.



# Static Website Hosting



- You can host a static website on Amazon S3. On a static website, individual web pages include static content.
- To host a static website, you configure an Amazon S3 bucket for website hosting and then upload your website content to the bucket.
- This bucket must have public read access. It is intentional that everyone in the world will have read access to this bucket.

# Static Website Hosting



- Depending on your Region, Amazon S3 website endpoints follow one of these two formats:
- `http://bucket-name.s3-website.Region.amazonaws.com`
- `http://bucket-name.s3-website-Region.amazonaws.com`
- This above URL will return the default index document that you configured for the website.

# Static Website Hosting



1. Public access to bucket
2. ACL enabled
3. Public access in object

# Storage Class



- Each object in Amazon S3 has a storage class associated with it.
- Amazon S3 offers a range of storage classes for the objects that you store.
- You choose a class depending on your use case scenario and performance access requirements. All of these storage classes offer high durability.

# Storage Classes– Frequent

For performance-sensitive use cases (those that require millisecond access time) and frequently accessed data, Amazon S3 provides the following storage class

- **Standard**—The default storage class. If you don't specify the storage class when you upload an object, Amazon S3 assigns the Standard storage class.

# Storage Classes– Infrequent

- **The Standard\_IA and Onezone\_IA** storage classes are designed for long-lived and infrequently accessed data
- Standard\_IA and Onezone\_IA objects are available for millisecond access (similar to the Standard storage class)
- Amazon S3 charges a retrieval fee for these objects, so they are most suitable for infrequently accessed data.
- The Standard\_IA and Onezone\_IA storage classes are suitable for objects larger than 128KB that you plan to store for at least 30 days. If an object is less than 128 KB, Amazon S3 charges you for 128 KB.
- Onezone\_IA – Amazon S3 stores the object data in only one Availability Zone, which makes it less expensive than Standard\_IA

# Storage Classes– Archive

-The Glacier and Deep Archive storage classes are designed for low-cost data archiving

## **Glacier**

- Long-term data archiving with retrieval times ranging from minutes to hours
- It has minimum storage duration period of 90 days
- If you have deleted, overwritten, or transitioned to a different storage class an object before the 90-day minimum, you are charged for 90 days.

# Storage Classes– Archive

## **Glacier Deep Archive**

- Archiving rarely accessed data with a default retrieval time of 12 hours
- It has minimum storage duration period of 180 days
- If you have deleted, overwritten, or transitioned to a different storage class an object before the 180-day minimum, you are charged for 180 days.



# Storage Classes– Auto Optimizes

- The Intelligent\_Tiering storage class is designed to optimize storage costs by automatically moving data to the most cost-effective storage access tier, without performance impact or operational overhead.
- Intelligent\_Tiering delivers automatic cost savings by moving data on a granular object level between two access tiers, when access patterns change
- Frequent access tier
- Lower-cost infrequent access tier

# Object Lifecycle

- To manage your objects so that they are stored cost effectively throughout their lifecycle, configure their lifecycle.
- A lifecycle configuration is a set of rules that define actions that Amazon S3 applies to a group of objects.

There are two types of actions

Transition actions—Define when objects transition to another storage class.

Expiration actions—Define when objects expire. Amazon S3 deletes expired objects on your behalf.

# Replication

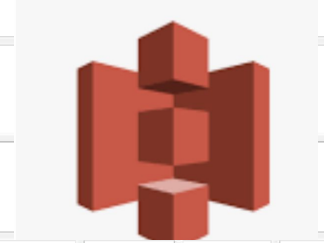
- Replication enables automatic, asynchronous copying of objects across Amazon S3 buckets.
- Buckets that are configured for object replication can be owned by the same AWS account or by different accounts.
- You can copy objects between different AWS Regions or within the same Region.

# Types of Object Replication

You can replicate objects between different AWS Regions or within the same AWS Region.

1. Cross-Region replication (CRR) is used to copy objects across Amazon S3 buckets in different AWS Regions.
2. Same-Region replication (SRR) is used to copy objects across Amazon S3 buckets in the same AWS Region.

# S3 Delete Bucket



- You can delete the objects individually. Or you can empty a bucket, which deletes all the objects in the bucket without deleting the bucket.
- You can also delete a bucket and all the objects contained in the bucket.
- If you want to use the same bucket, don't delete the bucket, can empty the bucket and keep it.
- After you delete the bucket, It is available for re-use, but the name might not be available for you to reuse for various reasons. For example, it might take some time before the name can be reused, and some other account could create a bucket with that name before you do.



# VPC



- Amazon Virtual Private Cloud (Amazon VPC) enables you to launch AWS resources into a virtual network that you've defined.
- A virtual private cloud (VPC) is a virtual network dedicated to your AWS account.
- It is logically isolated from other virtual networks in the AWS Cloud.
- You can launch your AWS resources, such as Amazon EC2 instances, into your VPC.

# VPC



- You can specify an IP address range for the VPC, add subnets, associate security groups, and configure route tables.
- A VPC spans all the Availability Zones in the region.
- After creating a VPC, you can add one or more subnets in each Availability Zone.
- Each subnet must reside entirely within one Availability Zone and cannot span zones.



# VPC



- You can specify an IP address range for the VPC, add subnets, associate security groups, and configure route tables.
- A VPC spans all the Availability Zones in the region.
- After creating a VPC, you can add one or more subnets in each Availability Zone.
- Each subnet must reside entirely within one Availability Zone and cannot span zones.

# Default vs Custom VPC

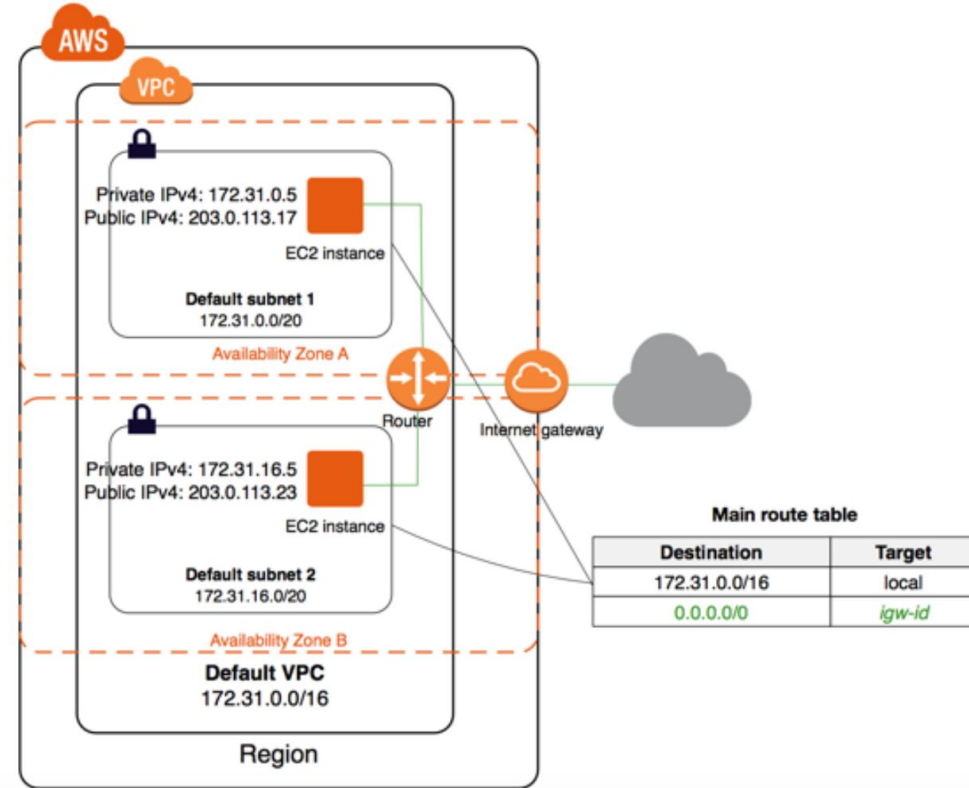


- If your account supports the EC2-VPC platform only, it comes with a default VPC that has a default subnet in each Availability Zone.
- A default VPC has the benefits of the advanced features provided by EC2-VPC, and is ready for you to use. If you have a default VPC and don't specify a subnet when you launch an instance, the instance is launched into your default VPC. You can launch instances into your default VPC without needing to know anything about Amazon VPC.
- Regardless of which platforms your account supports, you can create your own VPC, and configure it as you need. This is known as a non-default VPC. Subnets that you create in your non-default VPC and additional subnets that you create in your default VPC are called non-default subnets.

# Accessing the Internet



- Your default VPC includes an internet gateway, and each default subnet is a public subnet. Each instance that you launch into a default subnet has a private IPv4 address and a public IPv4 address. These instances can communicate with the internet through the internet gateway.



# VPC and Subnet



- When you create a VPC, you must specify an IPv4 CIDR block for the VPC
- The allowed block size is between a /16 netmask (65,536 IP addresses) and /28 netmask (16 IP addresses)
- The CIDR block of a subnet can be the same as the CIDR block for the VPC, or a subset of the CIDR block for the VPC (for multiple subnets)
- The allowed block size for a subnet is between a /28 netmask and /16 netmask. If you create more than one subnet in a VPC, the CIDR blocks of the subnets cannot overlap.

# Reserved IPs



- The first four IP addresses and the last IP address in each subnet CIDR block are not available for you to use, and cannot be assigned to an instance.
- For example, in a subnet with CIDR block 10.0.0.0/24, the following five IP addresses are reserved:
  - 10.0.0.0: Network address.
  - 10.0.0.1: Reserved by AWS for the VPC router.
  - 10.0.0.2: Reserved by AWS for the IP address of the DNS server
  - 10.0.0.3: Reserved by AWS for future use.
  - 10.0.0.255: Network broadcast address. We do not support broadcast in a VPC

# Public & Private Subnet



The instances in the public subnet can send outbound traffic directly to the Internet, whereas the instances in the private subnet can't. Instead, the instances in the private subnet can access the Internet by using a network address translation (NAT) gateway that resides in the public subnet.

# VPC Flow Logs



- VPC Flow Logs is a feature that enables you to capture information about the IP traffic going to and from network interfaces in your VPC.
- Flow log data can be published to Amazon CloudWatch Logs or Amazon S3
- You can create a flow log for a VPC, a subnet

# NACL



- A network access control list (ACL) is an optional layer of security for your VPC that acts as a firewall for controlling traffic in and out of one or more subnets
- Each subnet in your VPC must be associated with a network ACL
- You can associate a network ACL with multiple subnets
- A subnet can be associated with only one network ACL at a time
- A network ACL has separate inbound and outbound rules, and each rule can either allow or deny traffic
- Network ACLs are stateless



# NACL



Inbound					
Rule #	Type	Protocol	Port Range	Source	Allow/Deny
100	All IPv4 traffic	All	All	0.0.0.0/0	ALLOW
*	All IPv4 traffic	All	All	0.0.0.0/0	DENY
Outbound					
Rule #	Type	Protocol	Port Range	Destination	Allow/Deny
100	All IPv4 traffic	All	All	0.0.0.0/0	ALLOW
*	All IPv4 traffic	All	All	0.0.0.0/0	DENY

# Route Tables



- A route table contains a set of rules, called routes, that are used to determine where network traffic from your subnet or gateway is directed.
- Main route table—The route table that automatically comes with your VPC. It controls the routing for all subnets that are not explicitly associated with any other route table.
- Custom route table—A route table that you create for your VPC.
- Each subnet in your VPC must be associated with a route table
- A subnet can only be associated with one route table at a time, but you can associate multiple subnets with the same subnet route table

# Internet Gateway



- An internet gateway is a horizontally scaled, redundant, and highly available VPC component that allows communication between instances in your VPC and the internet.
- It therefore imposes no availability risks or bandwidth constraints on your network traffic.



# NAT



- You can use a NAT device to enable instances in a private subnet to connect to the internet (for example, for software updates) or other AWS services, but prevent the internet from initiating connections with the instances. A NAT device forwards traffic from the instances in the private subnet to the internet or other AWS services, and then sends the response back to the instances.
- AWS offers two kinds of NAT devices
  - NAT Gateway : You are charged for creating and using a NAT gateway in your account. NAT gateway hourly usage and data processing rates apply.
  - NAT Instance - A NAT instance is launched from a NAT AMI

# VPC Peering

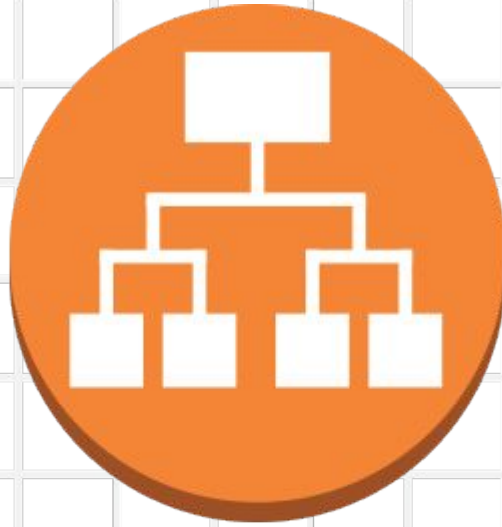


- A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them privately.
- Instances in either VPC can communicate with each other as if they are within the same network.
- You can create a VPC peering connection between your own VPCs, with a VPC in another AWS account, or with a VPC in a different AWS Region.
- AWS uses the existing infrastructure of a VPC to create a VPC peering connection; it is neither a gateway nor an VPN connection, and does not rely on a separate piece of physical hardware.
- There is no single point of failure for communication or a bandwidth bottleneck.

# Direct Connect



- AWS Direct Connect is a cloud service solution that makes it easy to establish a dedicated network connection from your premises to AWS.
- Using AWS Direct Connect, you can establish private connectivity between AWS and your datacenter, which in many cases can reduce your network costs, increase bandwidth throughput, and provide a more consistent network experience than Internet-based connections.



Elastic  
Load Balancer



# ELB



Elastic  
Load Balancer

- Elastic Load Balancing distributes incoming application or network traffic across multiple targets, such as Amazon EC2 instances, containers, and IP addresses, in multiple Availability Zones
- Elastic Load Balancing scales your load balancer as traffic to your application changes over time. It can automatically scale to the vast majority of workloads

# ELB – Benefits



Elastic  
Load Balancer

- A load balancer distributes workloads across multiple compute resources, such as virtual servers. Using a load balancer increases the availability and fault tolerance of your applications
- You can add and remove compute resources from your load balancer as your needs change, without disrupting the overall flow of requests to your applications
- You can configure health checks, which monitor the health of the compute resources, so that the load balancer sends requests only to the healthy ones

# ELB – Need



Elastic  
Load Balancer

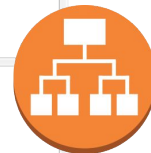
- To distribute incoming application traffic
- To achieve fault tolerant
- To detect unhealthy instances
- Enable ELB within single and multiple availability zone

# ELB – Nodes features



Elastic  
Load Balancer

- When you enable an Availability Zone for your load balancer, Elastic Load Balancing creates a load balancer node in the Availability Zone
- Your load balancer is most effective when you ensure that each enabled Availability Zone has at least one registered target
- AWS recommend that you enable multiple Availability Zones
- This configuration helps ensure that the load balancer can continue to route traffic If one Availability Zone becomes unavailable or has no healthy targets, the load balancer can route traffic to the healthy targets in another Availability Zone



Elastic  
Load Balancer

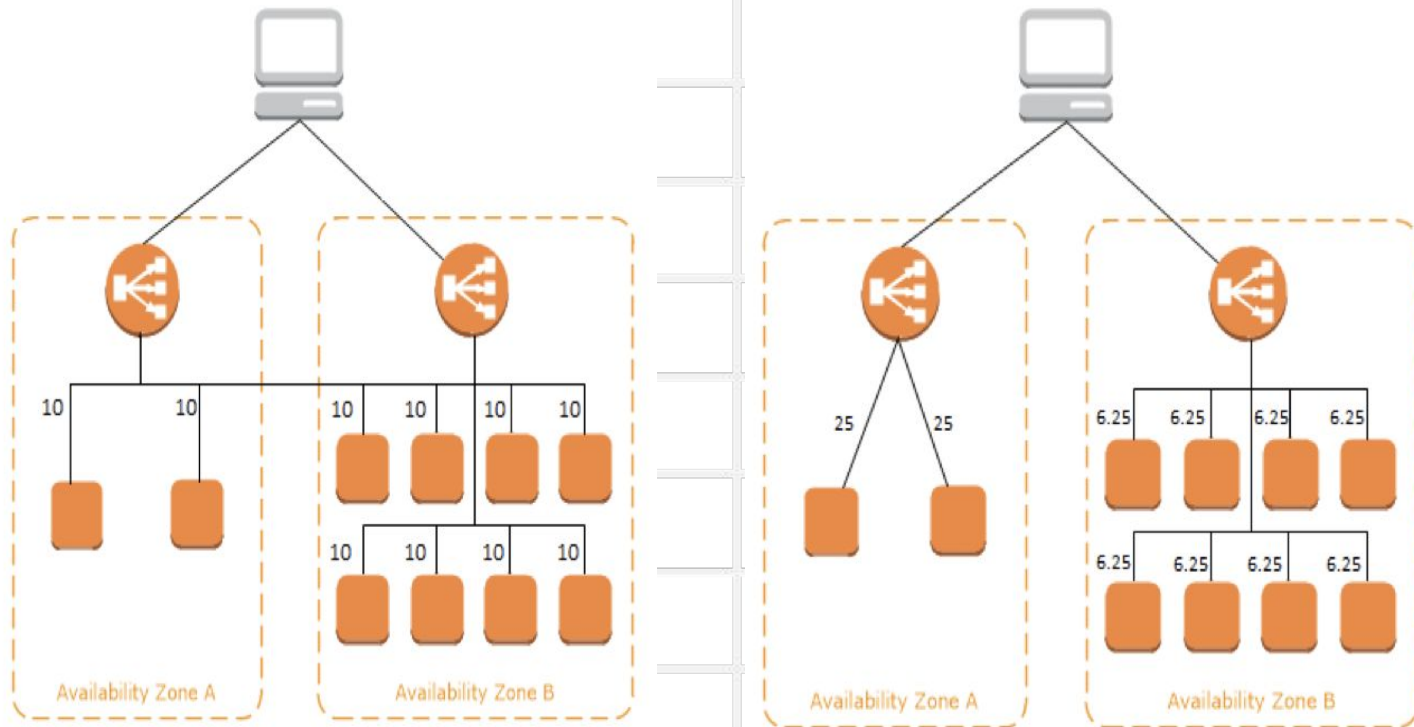
# Cross-Zone Load Balancing

- The nodes for your load balancer distribute requests from clients to registered targets
- When cross-zone load balancing is enabled, each load balancer node distributes traffic across the registered targets in all enabled Availability Zones
- When cross-zone load balancing is disabled, each load balancer node distributes traffic only across the registered targets in its Availability Zone

# Cross-Zone Load Balancing



Elastic  
Load Balancer



# Internet and Internal Load Balancer



Elastic  
Load Balancer

When you create a load balancer in a VPC, you must choose whether to make it an internal load balancer or an Internet-facing load balancer.

- The nodes of an Internet-facing load balancer have public IP addresses
- The DNS name of an Internet-facing load balancer is publicly resolvable to the public IP addresses of the nodes. Therefore, Internet-facing load balancers can route requests from clients over the Internet
- The nodes of an internal load balancer have only private IP addresses

# Internet and Internal Load Balancer



Elastic  
Load Balancer

When you create a load balancer in a VPC, you must choose whether to make it an internal load balancer or an Internet-facing load balancer.

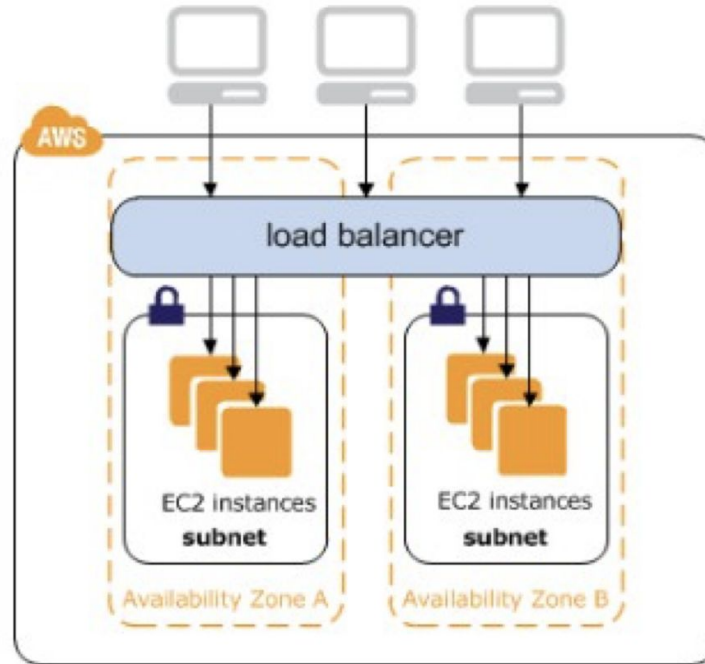
- The nodes of an Internet-facing load balancer have public IP addresses
- The DNS name of an Internet-facing load balancer is publicly resolvable to the public IP addresses of the nodes. Therefore, Internet-facing load balancers can route requests from clients over the Internet
- The nodes of an internal load balancer have only private IP addresses



# Internet Load Balancer



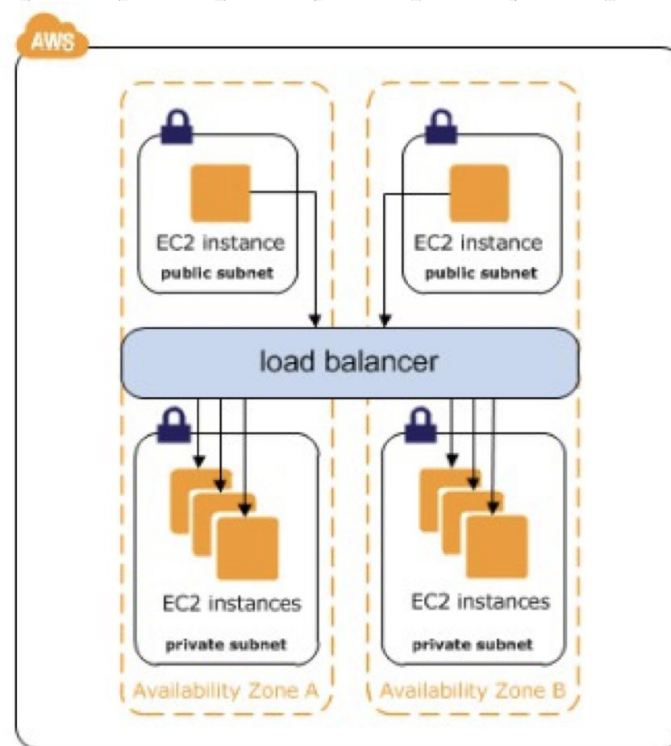
Elastic  
Load Balancer



# Internal Load Balancer



Elastic  
Load Balancer



# ELB – Types



Elastic  
Load Balancer

Elastic Load Balancing supports three types of load balancers

- Application Load Balancers
- Network Load Balancers
- Classic Load Balancers
- Gateway Load Balancer

# ELB – Classic



Elastic  
Load Balancer

- Classic Load Balancer provides basic load balancing across multiple Amazon EC2 instances
- Classic Load Balancer is intended for applications that were built within the EC2-Classic network
- CLB send the requests directly to EC2 instances
- AWS recommend Application Load Balancer for Layer 7 and Network Load Balancer for Layer 4 when using Virtual Private Cloud (VPC)



# ELB – Application



Elastic  
Load Balancer

- A load balancer serves as the single point of contact for clients
- A listener checks for connection requests from clients, using the protocol and port that you configure, and forwards requests to one or more target groups, based on the rules that you define
- Each rule specifies a target group, condition, and priority. When the condition is met, the traffic is forwarded to the target group.
- An Application Load Balancer functions at the application layer, the seventh layer of the Open Systems Interconnection (OSI) model

# ELB – Application



Elastic  
Load Balancer

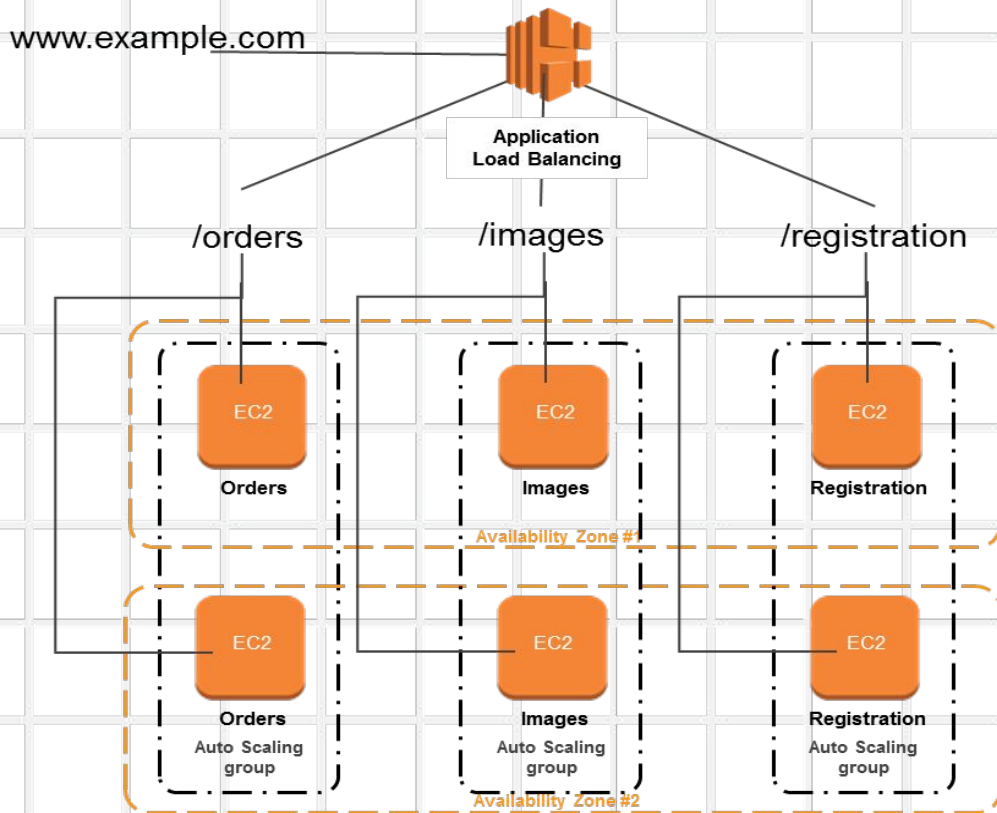
## Features

- Content based routing
- Support for container based applications
- Better metrics
- Support for additional Protocols and Workloads

# ELB - Application



Elastic  
Load Balancer





# ELB – Network



Elastic  
Load Balancer

- It is designed to handle tens of millions of requests per second while maintaining high throughput at ultra low latency, with no effort on your part
- A Network Load Balancer functions at the fourth layer of the Open Systems Interconnection (OSI) model
- When you enable an Availability Zone for the load balancer, Elastic Load Balancing creates a load balancer node in the Availability Zone

# ELB – Network



Elastic  
Load Balancer

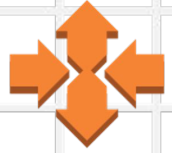
- Long-running Connections – NLB handles connections with built-in fault tolerance, and can handle connections that are open for months or years, making them a great fit for IoT, gaming, and messaging applications

## Benefits

- -Ability to handle volatile workloads and scale to millions of requests per second
- -Support for containerized applications
- -Support for monitoring the health of each service independently

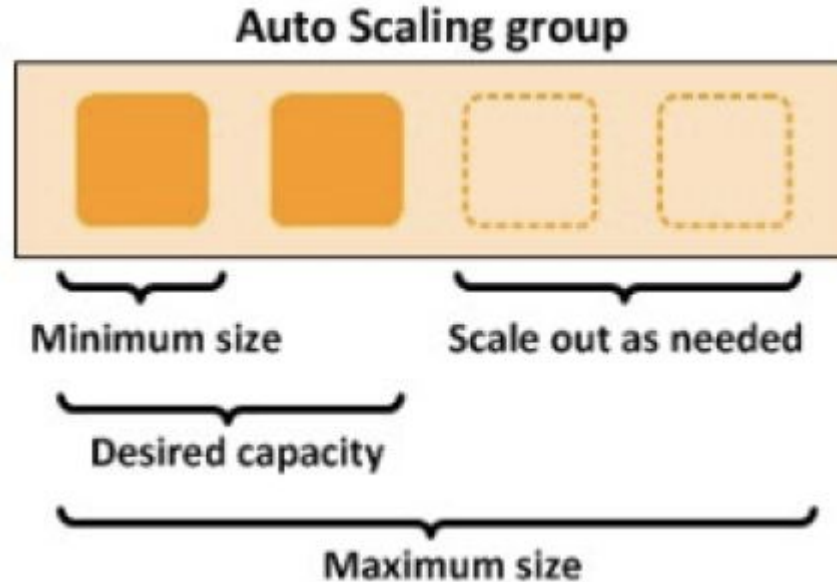
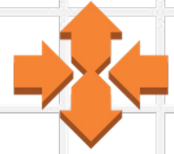


# Auto Scaling

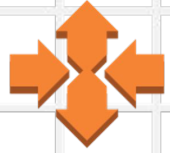


- Amazon EC2 Auto Scaling helps you ensure that you have the correct number of Amazon EC2 instances available to handle the load for your application
- You create collections of EC2 instances, called *Auto Scaling groups*
- You can specify the minimum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes below this size
- You can specify the maximum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes above this size
- If you specify scaling policies, then Amazon EC2 Auto Scaling can launch or terminate instances as demand on your application increases or decreases

# Auto Scaling



# Auto Scaling



- - Amazon EC2 Auto Scaling helps you ensure that you have the correct number of Amazon EC2 instances available to handle the load for your application
- - You create collections of EC2 instances, called *Auto Scaling groups*
- - You can specify the minimum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes below this size
- - You can specify the maximum number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling ensures that your group never goes above this size
- - If you specify scaling policies, then Amazon EC2 Auto Scaling can launch or terminate instances as demand on your application increases or decreases

# Auto Scaling – Benefits



- Setup Scaling quickly
- Make smart Scaling decisions
- Automatically maintain performance
- Pay only for what you need

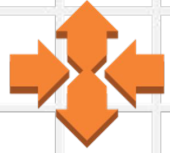
# Auto Scaling – Policies



- Simple Scaling Policy (Desired counts on particular threshold)
- Step Scaling Policy (More granular control)
- Target Scaling Policy (Dynamic in nature)

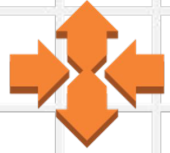


# Launch Config/Template



- A launch configuration is an instance configuration template that an Auto Scaling group uses to launch EC2 instances
- When you create a launch configuration/template, you specify information for the instances which include the ID of the Amazon Machine Image (AMI), the instance type, a key pair, one or more security groups, and a block device mapping
- If you've launched an EC2 instance before, you specified the same information in order to launch the instance

# Launch Config/Template



- You can specify your launch configuration with multiple Auto Scaling groups.  
However, you can only specify one launch configuration for an Auto Scaling group at a time, and you can't modify a launch configuration after you've created it
- In launch template you can do make changes.



# **The 5 Pillars of the Framework in AWS**

# The Five Pillars of the Framework

- Creating a software system is a lot like constructing a building. If the foundation is not solid, structural problems can undermine the integrity and function of the building.
- When architecting technology solutions, if you neglect the five pillars of **operational excellence, security, reliability, performance efficiency, and cost optimization** it can become challenging to build a system that delivers on your expectations and requirements.
- Incorporating these pillars into your architecture will help you produce stable and efficient systems.

# 1. Operational Excellence

The Operational Excellence pillar includes the ability to run and monitor systems to deliver business value and to continually improve supporting processes and procedures.

The design principles for operational excellence in the cloud

- Perform operations as code
- Annotate documentation
- Make frequent, small, reversible changes
- Refine operations procedures frequently
- Anticipate failure
- Learn from all operational failures

# 2.Security

The Security pillar includes the ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies.

## **The design principles for security in the cloud:**

- Enable traceability
- Apply security at all layers
- Automate security best practices
- Protect data in transit and at rest
- Prepare for security events

# 3. Reliability

The Reliability pillar includes the ability of a system to recover from infrastructure or service disruptions, dynamically acquire computing resources to meet demand, and mitigate disruptions such as misconfigurations or transient network issues.

The design principles for reliability in the cloud:

- Automatically recover from failure
- Scale horizontally to increase aggregate system availability
- Manage change in automation

# 4. Performance Efficiency

The Performance Efficiency pillar includes the ability to use computing resources efficiently to meet system requirements, and to maintain that efficiency as demand changes and technologies evolve.

The design principles for performance efficiency in the cloud:

- Democratize advanced technologies
- Go global in minutes
- Use serverless architectures
- Experiment more often
- Mechanical sympathy



# 5. Cost Optimization

The Cost Optimization pillar includes the ability to run systems to deliver business value at the lowest price point.

The design principles for cost optimization in the cloud:

- Adopt a consumption model:
- Measure overall efficiency
- Stop spending money on data center operations
- Analyze and attribute expenditure:
- Use managed and application level services to reduce cost of ownership



Trusted Advisor

# Trusted Advisor

- AWS Trusted Advisor acts like your customized cloud expert, and it helps you provision your resources by following best practices.
- Trusted Advisor inspects your AWS environment and finds opportunities to save money, improve system performance and reliability, or help close security gaps.



## Cost Optimization checks

- ✓ Low Utilization Amazon EC2 Instances : Checks the Amazon Elastic Compute Cloud (Amazon EC2) instances that were running at any time during the last 14 days and alerts you if the daily CPU utilization was 10% or less and network I/O was 5 MB or less on 4 or more days
- ✓ Underutilized Amazon EBS Volumes: Checks Amazon Elastic Block Store (Amazon EBS) volume configurations and warns when volumes appear to be underused
- ✓ Unassociated Elastic IP Addresses: Checks for Elastic IP addresses (EIPs) that are not associated with a running Amazon Elastic Compute Cloud (Amazon EC2) instance
- ✓ Amazon RDS Idle DB Instances: Checks the configuration of your Amazon Relational Database Service (Amazon RDS) for any DB instances that appear to be idle. If a DB instance has not had a connection for a prolonged period of time, you can delete the instance to reduce costs

## Performance Checks

- ✓ **High Utilization Amazon EC2 Instances:** Checks the Amazon Elastic Compute Cloud (Amazon EC2) instances that were running at any time during the last 14 days and alerts you if the daily CPU utilization was more than 90% on 4 or more days
- ✓ **Large Number of Rules in an EC2 Security Group:** Checks each Amazon Elastic Compute Cloud (EC2) security group for an excessive number of rules. If a security group has a large number of rules, performance can be degraded
- ✓ **Over utilized Amazon EBS Magnetic Volumes:** Checks for Amazon Elastic Block Store (EBS) Magnetic volumes that are potentially over utilized and might benefit from a more efficient configuration

## Security Checks

- ✓ Security Groups - Specific Ports Unrestricted: Checks security groups for rules that allow unrestricted access (0.0.0.0/0) to specific ports
- ✓ Amazon EBS Public Snapshots: Checks the permission settings for your Amazon Elastic Block Store (Amazon EBS) volume snapshots and alerts you if any snapshots are marked as public
- ✓ AWS CloudTrail Logging: Checks for your use of AWS CloudTrail. CloudTrail provides increased visibility into activity in your AWS account by recording information about AWS API calls made on the account

## Fault Tolerance Checks

- ✓ Amazon EBS Snapshots: Checks the age of the snapshots for your Amazon Elastic Block Store (Amazon EBS) volumes (available or in-use)
- ✓ Amazon EC2 Availability Zone Balance: Checks the distribution of Amazon Elastic Compute Cloud (Amazon EC2) instances across Availability Zones in a region
- ✓ Amazon RDS Backups: Checks for automated backups of Amazon RDS DB instances
- ✓ Amazon RDS Multi-AZ: Checks for DB instances that are deployed in a single Availability Zone
- ✓ Amazon S3 Bucket Logging: Checks the logging configuration of Amazon Simple Storage Service (Amazon S3) buckets

## Service Limit Checks

- ✓ EBS Active Snapshots: Checks for usage that is more than 80% of the EBS Active Snapshots Limit
- ✓ EC2 Elastic IP Addresses: Checks for usage that is more than 80% of the EC2 Elastic IP Addresses Limit
- ✓ RDS DB Instances: Checks for usage that is more than 80% of the RDS DB Instances Limit
- ✓ VPC: Checks for usage that is more than 80% of the VPC Limit
- ✓ VPC Internet Gateways: Checks for usage that is more than 80% of the VPC Internet Gateways Limit





Amazon  
**CloudTrail**

# Introduction to Cloud Trail

- AWS CloudTrail is an AWS service that helps you enable governance, compliance, and operational and risk auditing of your AWS account
- Actions taken by a user, role, or an AWS service are recorded as events in CloudTrail
- CloudTrail is enabled on your AWS account when you create it
- **You can identify**
  - who or what took which action
  - what resources were acted upon
  - when the event occurred
  - other details to help you analyze and respond to activity in your AWS account
- Can create 5 Trails per region (cannot be increased)

# CloudTrail Concepts

## CloudTrail Events

- An event in CloudTrail is the record of an activity in an AWS account
- This activity can be an action taken by a user, role, or service that is monitorable by

## CloudTrail

- There are two types of events that can be logged in CloudTrail
  - Management Events
  - Data Events
- By default, trails log management events, but not data events
- CloudTrail does not log all AWS services. Some AWS services do not enable logging of all APIs and events

## Management Events

Management events provide information about management operations that are performed on resources in your AWS account. These are also known as control plane operations

- Example
  - Registering devices
  - Configuring rules for routing data

## Data Events

Data events provide information about the resource operations performed on or in a resource. These are also known as data plane operations. Data events are often high-volume activities

- Example

- Amazon S3 object-level API activity Data events are disabled by default when you create a trail. To record CloudTrail data events, you must explicitly add to a trail the supported resources or resource types for which you want to collect activity

## Insights Events

CloudTrail Insights events capture unusual activity in your AWS account. If you have Insights events enabled, and CloudTrail detects unusual activity, Insights events are logged to a different folder or prefix in the destination S3 bucket for your trail

## CloudTrail Event History

CloudTrail event history provides a viewable, searchable, and downloadable record of the past 90 days of CloudTrail events. You can use this history to gain visibility into actions taken in your AWS account in the AWS Management Console, AWS SDKs, command line tools, and other AWS services

## Encryption

- By default, AWS CloudTrail encrypts all log files delivered to your specified Amazon S3 bucket using Amazon S3 server-side encryption (SSE)
- Optionally, can add a layer of security to your CloudTrail log files by encrypting the log files with your AWS Key Management Service (AWS KMS) key



# Cloudwatch



# Introduction to Cloud Watch

- Amazon Cloudwatch monitors your Amazon Web Services (AWS) resources and the applications you run on AWS in real time
- The Cloudwatch home page automatically displays metrics about every AWS service you use
- You can additionally create custom dashboards to display metrics about your custom applications by installing the CloudWatch agent on the instance/server

# Introduction to Cloud Watch

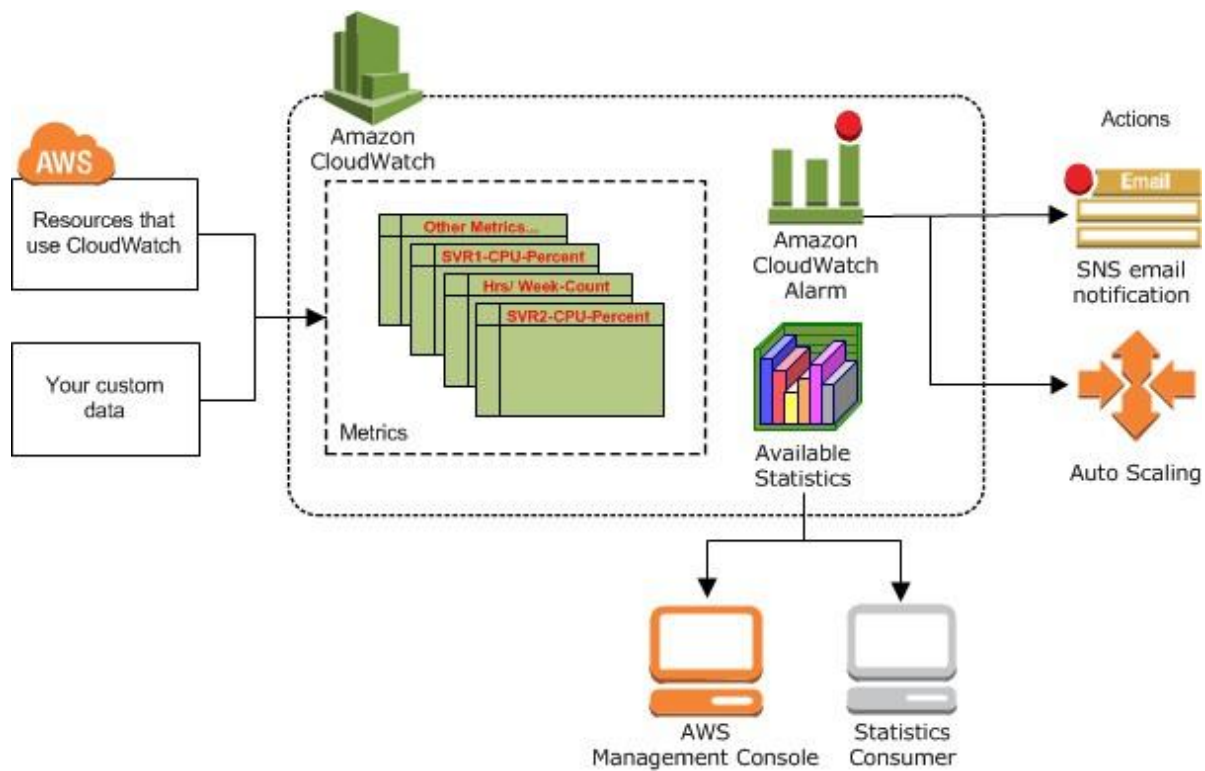
- You can create alarms which watch metrics and send notifications or automatically make changes to the resources you are monitoring when a threshold is breached
  - For example, you can monitor the CPU usage and disk reads and writes of your Amazon EC2 instances and then use this data to determine whether you should launch additional instances to handle increased load. You can also use this data to stop under-used instances to save money
- With Cloudwatch, you gain system-wide visibility into resource utilization, application performance, and operational health

# AWS Services which are integrated with Cloud watch

- Amazon Simple Notification Service (Amazon SNS)
- Amazon EC2 Auto Scaling
- AWS CloudTrail
- AWS Identity and Access Management (IAM)

# How Amazon Cloudwatch Works

- Amazon Cloudwatch is basically a metrics repository
- An AWS service—such as Amazon EC2—puts metrics into the repository, and you retrieve statistics based on those metrics
- If you put your own custom metrics into the repository, you can retrieve statistics on these metrics as well
- You can use metrics to calculate statistics and then present the data graphically in the Cloudwatch console
- You can configure alarm actions to stop, start, or terminate an Amazon EC2 instance when certain criteria are met
  - In addition, you can create alarms that initiate Amazon EC2 Auto Scaling and Amazon Simple Notification Service (Amazon SNS) actions on your behalf



# Cloudwatch Concepts

## Metric

- *Metrics* are the fundamental concept in Cloudwatch
- A metric represents a time-ordered set of data points that are published to Cloudwatch
- Think of a metric as a variable to monitor, and the data points as representing the values of that variable over time
- For example, the CPU usage of a particular EC2 instance is one metric provided by Amazon EC2. The data points themselves can come from any application or business activity from which you collect data
- By default, several services provide free metrics for resources (such as Amazon EC2 instances, Amazon EBS volumes, and Amazon RDS DB instances). You can also enable detailed monitoring for some resources

# Alarm

- You can create a CloudWatch alarm that watches a single CloudWatch metric
- The alarm performs one or more actions based on the value of the metric or expression relative to a threshold over a number of time periods.
- The action can be an Amazon EC2 action, an Amazon EC2 Auto Scaling action, or a notification sent to an Amazon SNS topic.
- You can also add alarms to CloudWatch dashboards and monitor them visually.
- When an alarm is on a dashboard, it turns red when it is in the ALARM state, making it easier for you to monitor its status proactively.
- After an alarm invokes an action due to a change in state, its subsequent behavior depends on the type of action that you have associated with the alarm.

# An alarm has the following possible states:

*OK*—The metric or expression is within the defined threshold.

*ALARM*—The metric or expression is outside of the defined threshold.

*INSUFFICIENT\_DATA*—The alarm has just started, the metric is not available, or not enough data is available for the metric to determine the alarm state.





# Database services of AWS

## 1. Relational

Used for: Traditional applications, ERP, CRM, and e-commerce.

### AWS Offerings

Amazon RDS

Aurora

Amazon Redshift



**Relational**

Students

Student ID	Dept ID	Name	Email
1	M01	Joe Miller	joe@abc.com
2	B01	Sarah T	sarah@abc.com

Departments

Dept ID	SPOC	Email	Phone
M01	Kelly Jones	kelly@abc.com	+1234567890
B01	Satish Kumar	satish@abc.com	+1234567891

Subjects

Student ID	Subject
1	Physics
1	Chemistry
1	Math
2	History
2	Geography
2	Economics



Amazon  
**RDS**

# **Relational Database Services**





# Introduction to Amazon RDS

- Easy to set up, operate, and scale a relational database in the cloud.
- Provides cost-efficient and resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching and backups.
- Makes to focus on your applications so you can give them the fast performance, high availability, security and compatibility they need.
- Primary use case is a transactional database (rather than analytical)

# Why Managed RDS vs On Servers

- Automated Scaling
- Easy to administer
- Highly scalable
- Available and durable
- Fast
- Secure
- Inexpensive
- Automated and Manual Backups

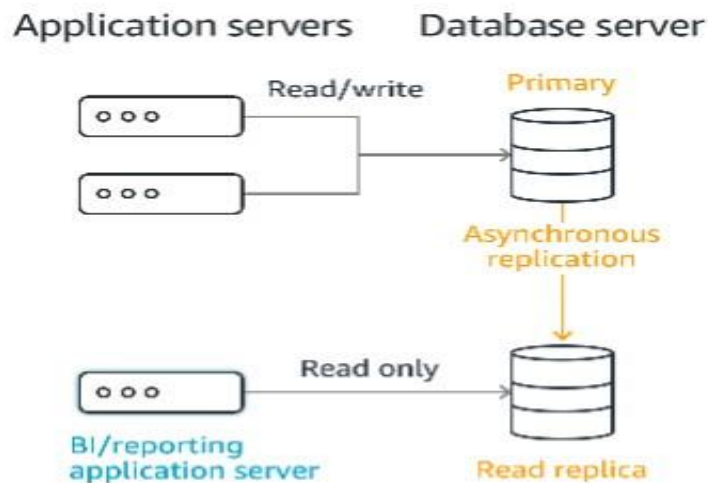


# Different types of database engines supported in AWS RDS

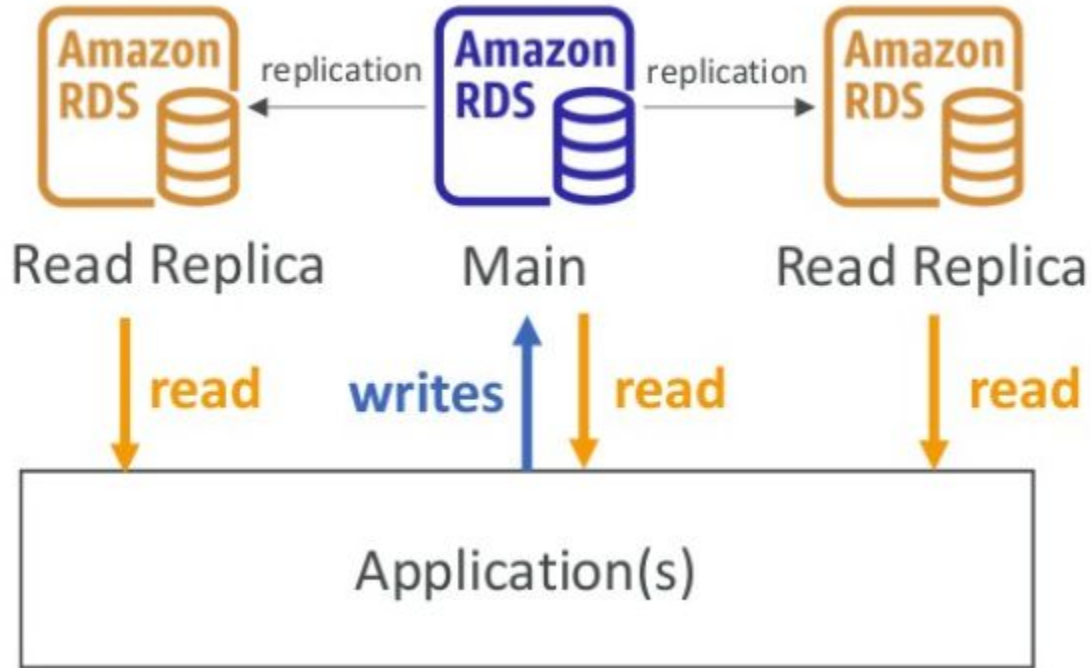
- MySQL
- Maria DB
- PostgreSQL
- Oracle
- Microsoft SQL Server DB engines
- Amazon Aurora

# Database Read replicas

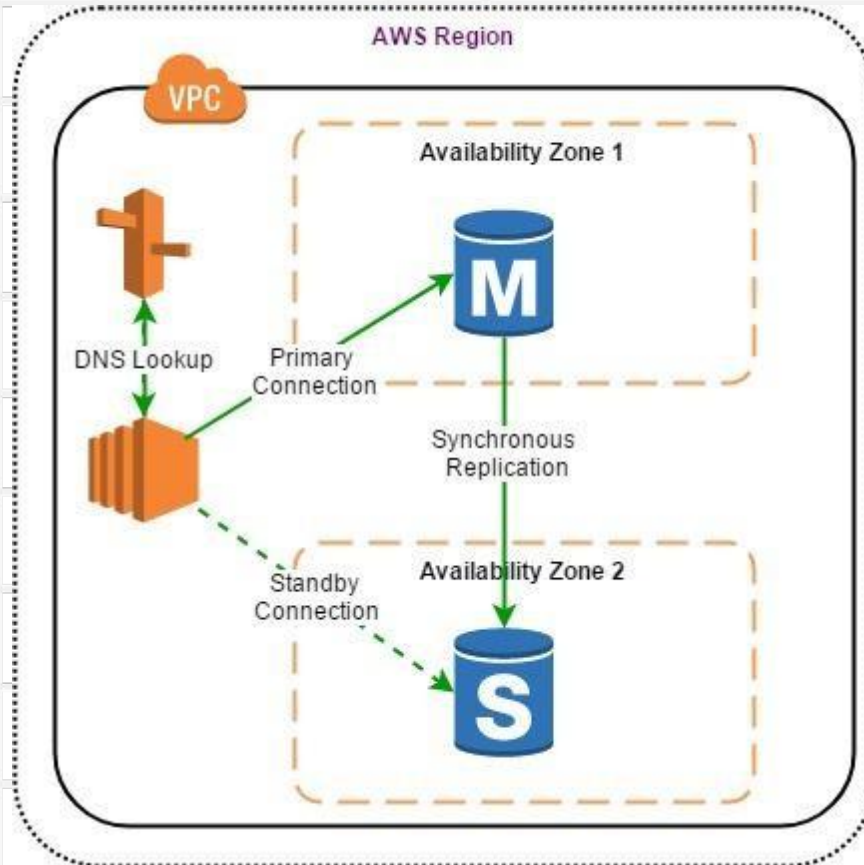
- When the required read I/O capacity is reached but still more I/O capacity is required for heavy/intensive read applications, RDS read replicas can be helpful
- A read replica is a replica of the primary database that can be used only for read actions



# Read Replicas



# Multi-AZ Deployment



# Multi-AZ Deployment

- Multi-AZ for RDS provides high availability, data durability, and fault tolerance for DB instances
- You can select the Multi-AZ option during RDS DB instance launch or modify an existing stand alone RDS instance
- AWS creates a secondary database in different availability zone in the same region for high availability
- Not possible to Insert/Update/Select the data to the Secondary (stand-by) RDS database
- OS patching, System upgrades and DB scaling are done on standby DB first and then on primary

# Encrypting Amazon RDS Resources

- You can encrypt your Amazon RDS DB instances and snapshots at rest by enabling the encryption option for your Amazon RDS DB instances.
- You can't disable encryption on an encrypted DB
- You can not enable encryption for an existing, un-encrypted database instance, but there is an alternate way
  - Create a snapshot of the DB
  - Copy the snapshot and choose to encrypt it during the copy process
  - Restore the encrypted copy into a New DB



# Alternative to Amazon RDS

If your use case isn't supported on RDS, you can run databases on Amazon EC2.

Consider the following points when considering a DB on EC2:

- You can run any database you like with full control and ultimate flexibility.
- You must manage everything like backups, redundancy, patching and scaling.
- Good option if you require a database not yet supported by RDS, such as SAP HANA.
- Good option if it is not feasible to migrate to AWS-managed database.



## 2. In-memory (Cache)

In-memory databases are used for applications that require real time access to data. By storing data directly in memory, these databases provide microsecond latency where millisecond latency is not enough.

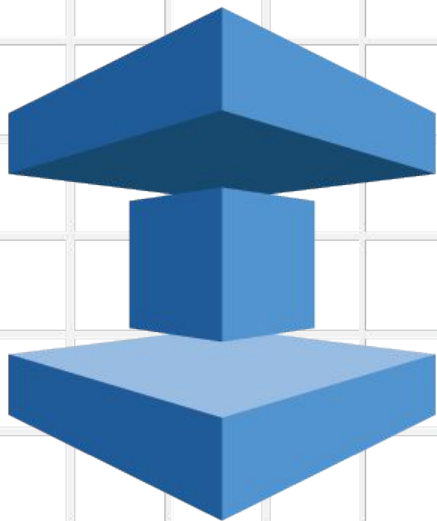
Used for: Caching, gaming leaderboards, and real-time analytics.

*AWS Offerings:*

- Amazon ElastiCache for Redis
- Amazon ElastiCache for Memcached



**In-memory**



ElastiCache

# Introduction to ElastiCache

- Amazon ElastiCache allows you to seamlessly set up, run, and scale popular open-Source compatible in-memory data stores in the cloud
- Build data-intensive apps or boost the performance of your existing databases by retrieving data from high throughput and low latency in-memory data stores
- ElastiCache can be used if data stores have areas of data that are frequently accessed but seldom updated
  - Additionally, querying a database will always be slower and more expensive than locating a key in a key-value pair cache.

# Uses cases

- ✓ Session Stores
- ✓ Gaming
- ✓ Real-Time Analytics
- ✓ Queuing

# Features

- Extreme performance by allowing for the retrieval of information from a fast, managed, in- memory system (instead of reading from the DB itself)
- Improves response times for user transactions and queries
- It offloads the read workload from the main DB instances (less I/O load on the DB)
  - It does this by storing the results of frequently accessed pieces of data (or computationally intensive calculations) in-memory
- Fully managed
- Scalable

# Supports two caching engines

- Memcached (is not a Data store [DB], only a cache)
- Redis can be used as a DB (data store)

# Amazon ElastiCache for Memcached

- Is not persistent
- Can not be used as a data store
- If the node fails, the cached data (in the node) is lost
- Ideal front-end for data stores (RDS, DynamoDB...etc)
- Does not support Multi-AZ failover, replication, nor does it support Snapshots for backup/restore
  - Node failure means data loss
- You can, however, place your Memcached nodes in different AZs to minimize the impact of an AZ failure and to contain the data loss in such an incident

## Use cases

- Cache contents of a DB
- Cache data from dynamically generated webpages

# Amazon ElastiCache for Redis

- Is persistent, using the snapshot feature
- At any time, you can restore your data by creating a new Redis cluster and populating it with data from a backup
- Supports Redis master/slave replication
- Supports snapshots (automatic and manual) to S3 (managed by AWS)
  - The back up can be used to restore a cluster or to seed a new cluster
  - The back up includes cluster metadata and all data in the cluster





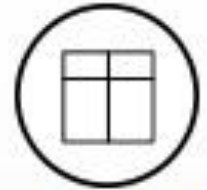
# 3. Nosql Database

Key-value databases are optimized to store and retrieve key-value pairs in large volumes and in milliseconds, without the performance overhead and scale limitations of relational databases.

Used for: Internet-scale applications, real-time bidding, shopping carts, and customer preferences.

## *AWS Offerings*

- Amazon DynamoDB



**Key-value**

# Nosql Database

- Document databases are designed to store semi-structured data as documents and are intuitive for developers to use because the data is typically represented as a readable document.
- Used for: Content management, personalization, and mobile applications.
- *AWS Offerings*
- Amazon DocumentDB (with MongoDB compatibility)



Document

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



Amazon DynamoDB

# Introduction to Dynamodb

- It is a key-value and document database that delivers single-digit millisecond performance at any scale
- It's a fully managed, multi-region, multi-master database with built-in security, backup and restore, and in-memory caching for internet-scale applications
- Can handle more than 10 trillion requests per day and support peaks of more than 20 million requests per second
- Many of the world's fastest growing businesses such as Lyft, Airbnb, and Redfin as well as enterprises such as Samsung, Toyota, and Capital One depend on the scale and performance of DynamoDB to support their mission-critical workloads

# Introduction to Dynamodb

## Benefits

- Performance at scale: DynamoDB supports some of the world's largest scale applications by providing consistent, single-digit millisecond response times at any scale
- Serverless: there are no servers to provision, patch, or manage and no software to install, maintain, or operate

# Use cases

- Serverless Web Applications
- Microservices Data Store
- Mobile Back ends
- Gaming
- IOT

# Customers

- Nike
- Samsung
- Netflix

# Tables

- Dynamo DB tables are schema less
  - Which means that neither the attributes nor their data types need to be defined before hand
  - Each item can have its own distinct attributes
- Dynamo DB does not support
  - Complex relations DB querying or joins
  - Does not support complex transactions



# Durability and performance

- Dynamo DB automatically keep data across three facilities(Datacenters) in a region for High availability and data durability
  - It also partitions your DB over sufficient number of servers according to reads/write capacity
  - Performs automatic failover in case of any failure
- Dynamo DB runs exclusively on SSD volumes which provides
  - Low latency
  - Predictable performance
  - High I/O's

# Dynamo DB basic Components

## Tables

- Like all other DBs, Dynamo DB stores data in tables
- A table is a collection of data items
- Each table can have an infinite number of data items

## Items

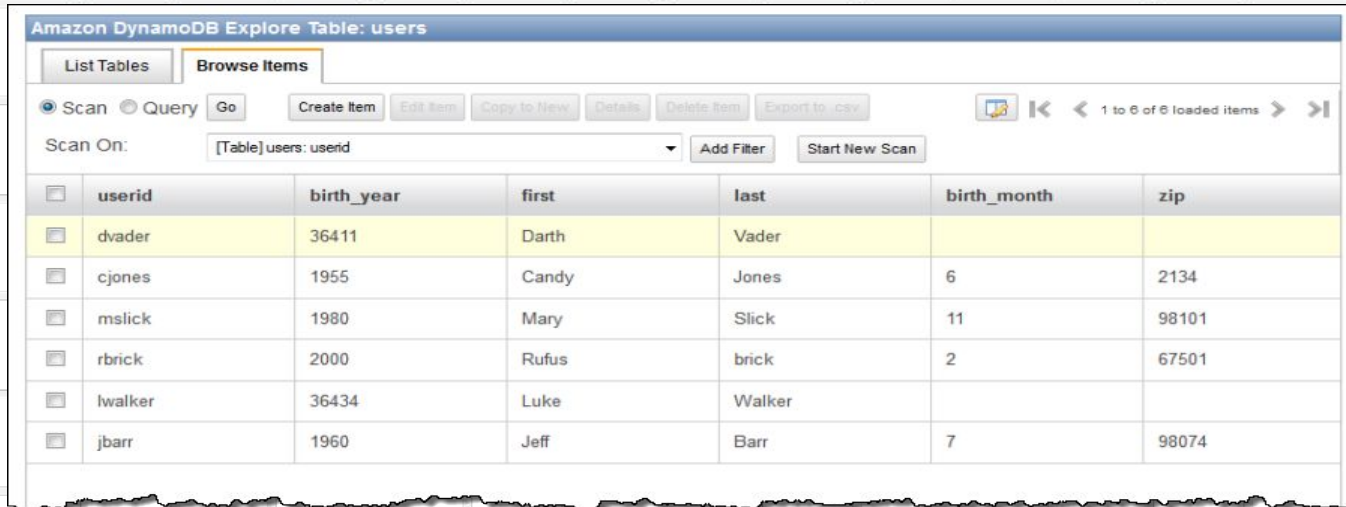
- Each table contains multiple data items
- An data item consists of a primary or composite key and a flexible number of attributes
- There is no limit to the number of items you can store in a table

```
{  
  "PersonID": 101,  
  "Last name": "Cam",  
  "First name": "Raj",  
  "Phone": "5214896"  
}
```

```
{  
  "PersonID": 102,  
  "Last name": "Jam",  
  "First name": "Chai",  
  "Address": {  
    "Street": "521/13",  
    "City": "Tonw",  
    "State": "Gojk"  
    "Zipcode": 125842  
  }  
}
```

# Attributes

- Each item is composed of one or more attributes
- An attribute consists of the attribute name and a value or a set of values
- An attribute is a fundamental data element
- Attributes in Dynamo DB are similar into fields or columns in other database systems



Amazon DynamoDB Explore Table: users

List Tables Browse Items

Scan Query Go Create Item Edit Item Copy to New Details Delete Item Export to csv

Scan On: [Table] users: usend Add Filter Start New Scan

1 to 6 of 6 loaded items

	userid	birth_year	first	last	birth_month	zip
<input checked="" type="checkbox"/>	dvader	36411	Darth	Vader		
<input type="checkbox"/>	cjones	1955	Candy	Jones	6	2134
<input type="checkbox"/>	mslick	1980	Mary	Slick	11	98101
<input type="checkbox"/>	rbrick	2000	Rufus	brick	2	67501
<input type="checkbox"/>	lwalker	36434	Luke	Walker		
<input type="checkbox"/>	jbarr	1960	Jeff	Barr	7	98074

## Read Capacity Units

- One read capacity unit represents one strongly consistent read per second, or two eventually consistent reads per second for an item up to 4KB in size
- If you need to read an item that is larger than 4 KB, Dynamo DB will need to consume additional read capacity units
- The total number of read capacity units required depends on the item size, and whether you want an eventually consistent or strongly consistent read.

## Write Capacity Units

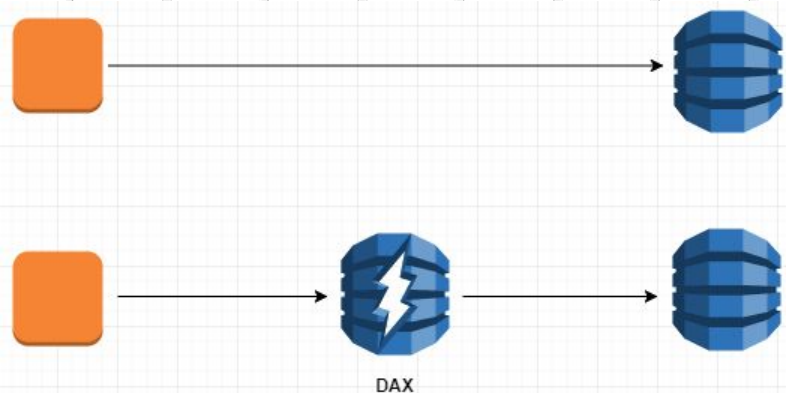
- One write capacity unit represents one write per second for an item up to 1 KB in size
- If you need to write an item that is larger than 1 KB, Dynamo DB will need to consume additional write capacity units
- The total number of write capacity units required depends on the item size.

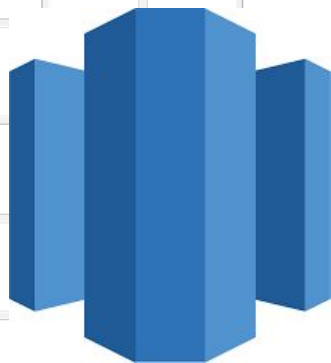
# Scalability

- It provides for a push button scaling on AWS where you can increase the read/write throughput and AWS will go a head and scale it for you (up or down) without downtime or performance degradations
- You can scale the provisioned capacity of your Dynamo DB table any time you want
- There is no limit to the number of items(data) you can store in a Dynamo DB table
- There is no limit on how much data you can store per Dynamo DB table

# Dynamo DB Accelerator

- Amazon Dynamo DB Accelerator (DAX) is a fully managed, highly available, in-memory cache for Dynamo DB that delivers up to a 10x performance improvement – from milliseconds to microseconds – even at millions of requests per second
- Now you can focus on building great applications for your customers without worrying about performance at scale
- You can enable DAX with just a few clicks





**Redshift**

# Introduction to Redshift

- Redshift, is an AWS fully managed, petabyte scale data warehouse service in the cloud
  - A data warehouse is a relational database that is designed for query and analysis rather than for transaction processing
  - It usually contains historical data derived from transaction data, but it can include data from other sources
  - To perform analytics you need a data warehouse not a regular database
  - OLAP (On-line Analytical Processing) is characterized by relatively low volume of transactions
    - Queries are often very complex and involve aggregations (group the data)
  - RDS (MySQL.. Etc.) is an OLTP database, where there is detailed and current data, and a schema used to store transactional data



# Data Security (Exam View)

## At Rest

- Supports encryption of data “at rest” using hardware accelerated AES-256bits (Advanced Encryption Standard)
- By default, AWS Redshift takes care of encryption key management
- You can choose to manage your own keys through HSM (Hardware Security Modules), or AWS KMS (Key Management Service)

## In-Transit

- Supports SSL Encryption, in-transit, between client applications and Redshift data warehouse cluster
- You can't have direct access to your AWS Redshift cluster nodes, however, you can through the applications themselves

# Redshift Cluster

- No upfront commitment, you can start small and grow as required
  - You can start with a single, 160GB, Redshift data warehouse node
- For a multi-node deployment (Cluster), you need a leader node and compute node(s)
  - The leader node manages client connections and receives queries
  - The compute nodes store data and perform queries and computations
  - You can have up to 128 compute nodes in a cluster

# Back-Up Retention

- Amazon Redshift automatically patches and backs up (Snapshots) your data warehouse, storing the backups for a user-defined retention period in AWS S3
  - It keeps the backup by default for one day (24hours) but you can configure it from 0 to 35days
  - Automatic backups are stopped if you choose retention period of 0
  - You have access to these automated snapshots during the retention period

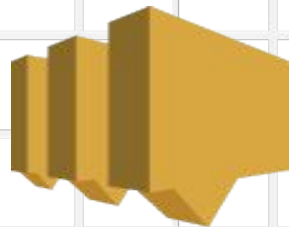
- If you delete the cluster
  - You can choose to have a final snapshot to use later
  - Manual backups are not deleted automatically, if you do not manually delete them, you will be charged standard S3 storage rates
- AWS Redshift currently supports only one AZ (no Multi-AZ option)
- You can restore from your backup to a new Redshift cluster in the same or a different AZ
  - This is helpful in case the AZ hosting your cluster fails

# Availability and Durability

- Redshift automatically replicates all your data within your data warehouse cluster
- Redshift always keeps three copies of your data
  - The original one
  - A replica on compute nodes (within the cluster)
  - A backup copy on S3

## - Cross Region Replication

- Redshift can asynchronously replicate your snapshots to S3 in another region for DR
- Amazon Redshift automatically detect and replace a failure node in your data warehouse cluster
  - The data warehouse cluster will be unavailable for the queries and updates until a replacement node is provided and added to the DB
  - Amazon Redshift makes your replacement node available immediately and loads the most frequently accessed data from S3 first to allow you to resume querying your data as quickly as possible



Amazon  
**SNS**

# Introduction to Simple Notification Services

- Amazon Simple Notification Service (SNS) is a highly available, durable, secure, fully managed pub/sub messaging service that enables you to decouple micro services, distributed systems, and serverless applications
- Amazon SNS provides topics for high-throughput, push-based, many-to-many messaging.
- Using Amazon SNS topics, your publisher systems can fan out messages to a large number of subscriber endpoints for parallel processing, including Amazon SQS queues, AWS Lambda functions, and HTTP/S web hooks
- Additionally, SNS can be used to fan out notifications to end users using mobile push, SMS, and email.



# Benefits

- Reliably deliver messages with durability
- Automatically scale your workload
- Simplify your architecture with Message Filtering
- Keep messages private and secure



### Publisher

Publish messages from distributed systems, microservices, and other AWS services



### Amazon SNS

Fully-managed pub/sub messaging and event-driven computing service



### SNS Topic

Decouple message publishers from subscribers with topics



### Message Filtering & Fanout

Filter messages according to subscription filter policies, and deliver them to subscribers



AWS Lambda



Amazon SQS

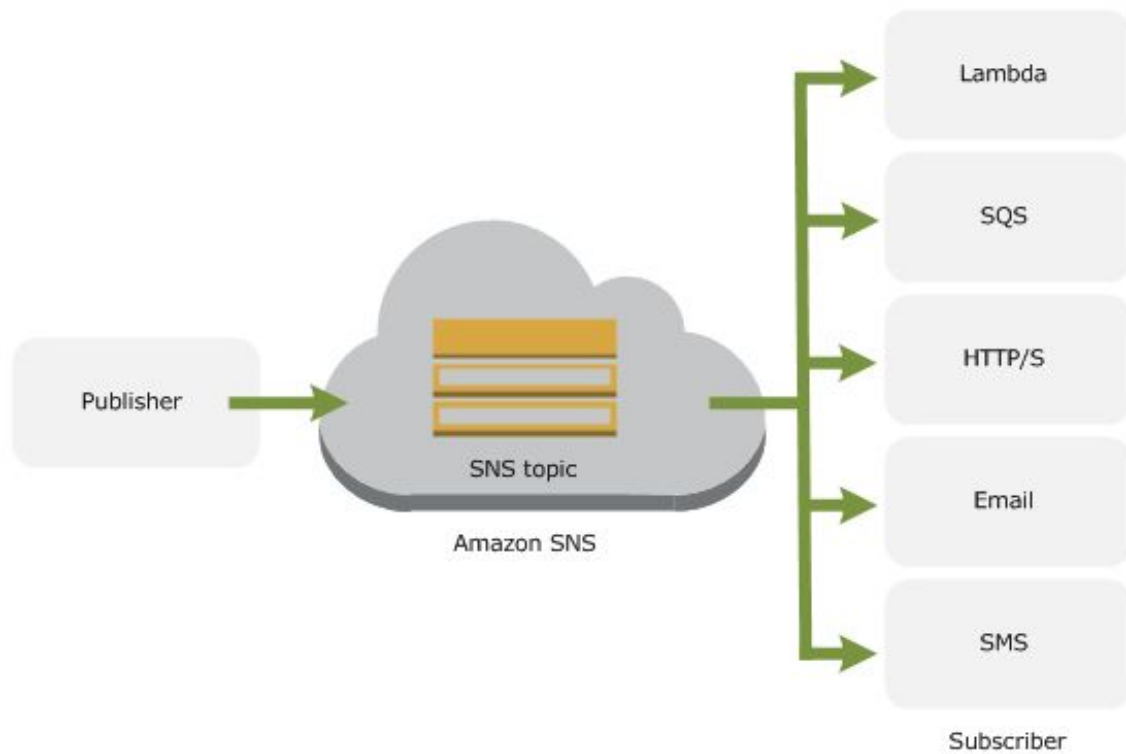


HTTP/S

### Subscribers

Receive messages in subscribing serverless functions, queues, microservices, and more

- In Amazon SNS, there are two types of clients—publishers and subscribers—also referred to as producers and consumers
- Publishers communicate asynchronously with subscribers by producing and sending a message to a topic, which is a logical access point and communication channel
- Subscribers (i.e., web servers, email addresses, Amazon SQS queues, AWS Lambda functions) consume or receive the message or notification over one of the supported protocols (i.e., Amazon SQS, HTTP/S, email, SMS, Lambda) when they are subscribed to the topic





# Simple Queue Service

# Introduction to SQS

- Amazon Simple Queue Service (Amazon SQS) offers a secure, durable, and available hosted queue that lets you integrate and decouple distributed software systems and components
- Fully managed Queuing service
- Eliminates the complexity and overhead associated with managing and operating message oriented middleware
- Can send, store, and receive messages between software components at any volume, without losing messages or requiring other services to be available

# Lifecycle of an Amazon SQS message in Distributed Queues

- There are three main parts in a distributed messaging system: the components of your distributed system, your queue (distributed on Amazon SQS servers), and the messages in the queue
- Your system has several *producers* (components that send messages to the queue) and *consumers* (components that receive messages from the queue). The queue redundantly stores the messages across multiple Amazon SQS servers



## Step 1

A producer (component 1) sends message A to a queue, and the message is distributed across the Amazon SQS servers redundantly

## Step 2

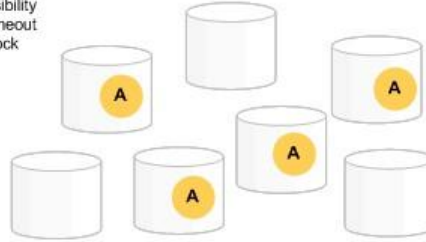
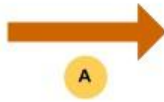
When a consumer (component 2) is ready to process messages, it consumes messages from the queue, and message A is returned

## Step 3

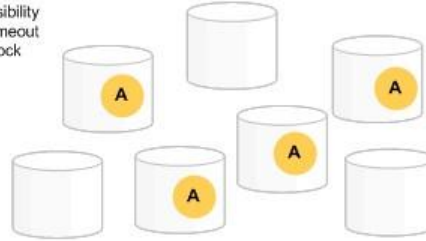
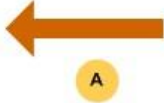
The consumer (component 2) deletes message A from the queue to prevent the message from being received and processed again when the visibility timeout expires



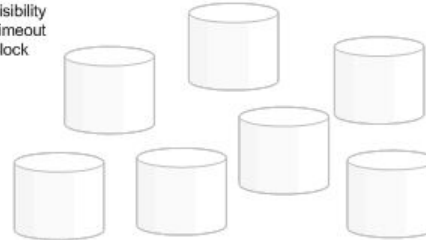
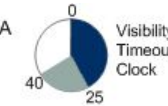
Component 1 sends  
Message A to the queue



Component 2 retrieves Message A  
from the queue and the visibility  
timeout period starts



Component 2 processes Message A  
and then deletes it from the queue  
during the visibility timeout period



# Use Case

SQS : Decoupling 2 applications and allowing parallel asynchronous processing

SNS : Fan-out - Processing the same message in multiple ways

# Persistence

SQS : Messages are persisted for some (configurable) duration if no consumer is available

SNS : No persistence. Whichever consumer is present at the time of message arrival gets the message and the message is deleted. If no consumers are available then the message is lost.

SQS : Queue

SNS : Topic (Pub/Sub system)

## **Message consumption**

**SQS : Pull Mechanism - Consumers poll and pull messages from SQS**

- SQS is distributed queuing system
  - Messages are NOT pushed to receivers
  - Receivers have to poll or pull messages from SQS

**SNS : Push Mechanism - SNS Pushes messages to consumers**

- SNS is a distributed publish-subscribe system

## SQS:

- Messages can't be received by multiple receivers at the same time
- Polling inherently introduces some latency in message delivery in SQS unlike SNS where messages are immediately pushed to subscribers.

## SNS :

- Messages are pushed to subscribers as and when they are sent by publishers to SNS
- SNS supports several end points such as email, sms, http end point and SQS



**Route53**

# How DNS Works

- DNS is a global system for translating IP addresses to human-readable domain names.
- When a user tries to access a web address like “example.com”, their web browser or application performs a DNS Query against a DNS server, supplying the hostname.
- The DNS server takes the hostname and resolves it into a numeric IP address, which the web browser can connect to.
- A component called a DNS Resolver is responsible for checking if the hostname is available in local cache, and if not, contacts a series of DNS Name Servers, until eventually it receives the IP of the service the user is trying to reach, and returns it to the browser or application. This usually takes less than a second.

# Introduction to Route53

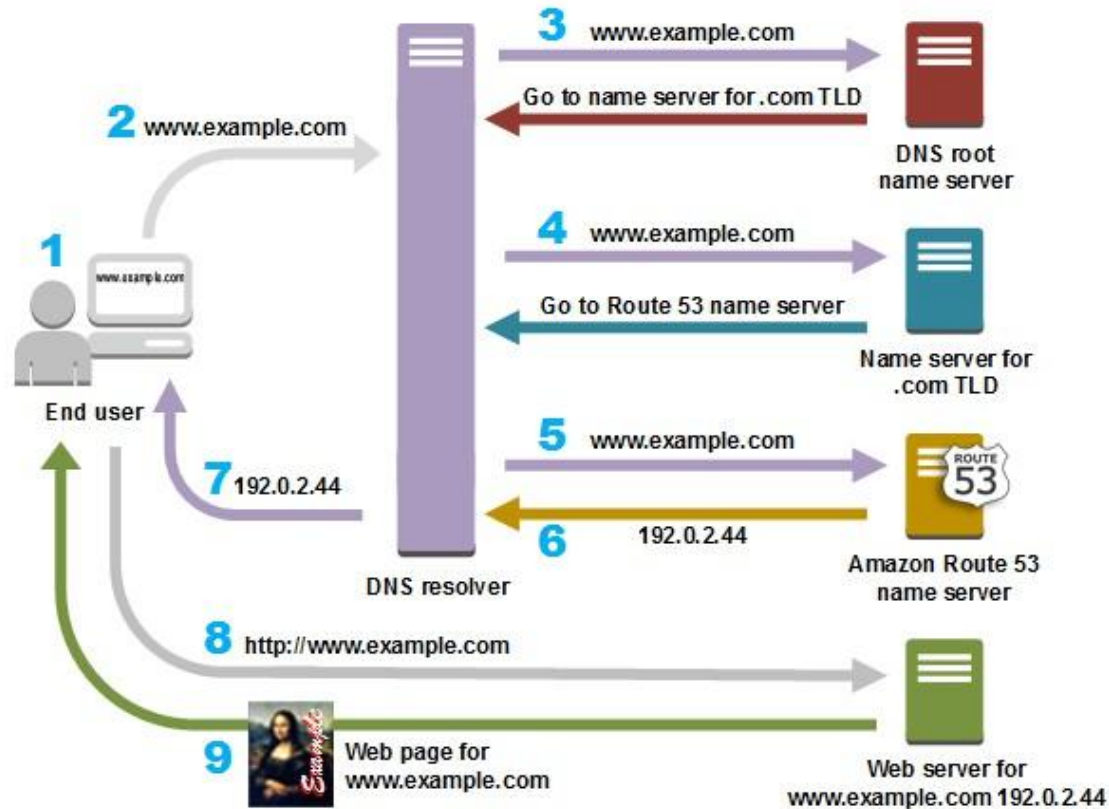
- Amazon Route 53 is a highly available and scalable Domain Name System (DNS) web service
- It is designed to give developers and businesses an extremely reliable and cost effective way to route end users to Internet applications by translating names like [www.example.com](http://www.example.com) into the numeric IP addresses like 192.0.2.1 that computers use to connect to each other
- Amazon Route 53 is fully compliant with IPv6 as well
- Amazon Route 53 effectively connects user requests to infrastructure running in AWS as well as infrastructure outside of AWS

# Benefits

- Highly available and reliable
- Flexible
- Designed for use with other Amazon Web Services
- Simple, Fast, Cost-effective
- Secure & Scalable
- Simplify the hybrid cloud

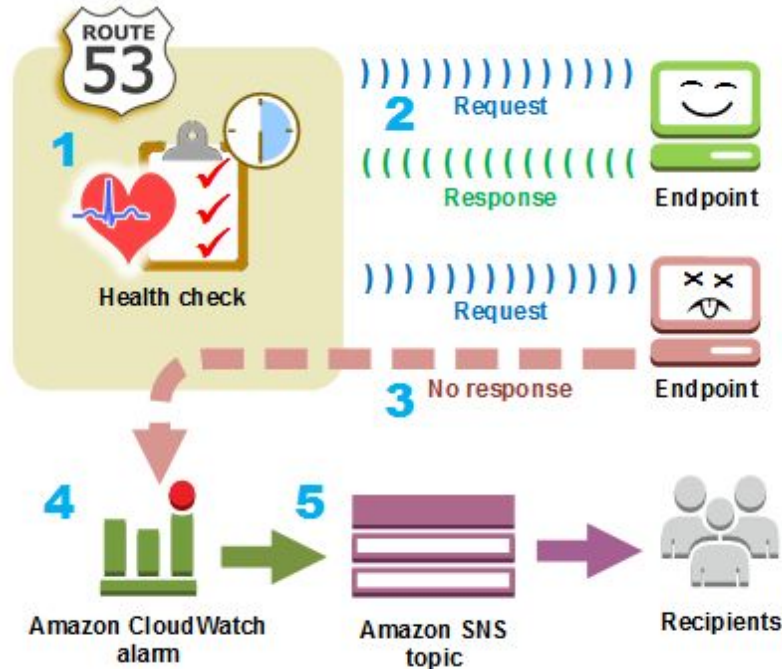


# How Internet Traffic Is Routed to Your Website or Web Application



# How Route 53 Checks the Health of Your Resources

- Amazon Route 53 health checks monitor the health of your resources such as web servers and email servers. You can optionally configure Amazon CloudWatch alarms for your health checks, so that you receive notification when a resource becomes unavailable



# Hosted Zones

A hosted zone is a container for records, and records contain information about how you want to route traffic for a specific domain, such as example.com, and its subdomains (acme.example.com, zenith.example.com).

There are two types of hosted zones:

- *Public hosted zones* contain records that specify how you want to route traffic on the internet
- You get a public hosted zone in one of two ways:
  - When you register a domain with Route 53, AWS create a hosted zone for you automatically.
  - When you transfer DNS service for an existing domain to Route 53, you start by creating a hosted zone for the domain.

- *Private hosted zones* contain records that specify how you want to route traffic in an Amazon VPC
  - You create a private hosted zone, such as example.com, and specify the VPCs that you want to associate with the hosted zone.
  - You create records in the hosted zone that determine how Route 53 responds to DNS queries for your domain and subdomains within and among your VPCs. For example, suppose you have a database server that runs on an EC2 instance in one of the VPCs that you associated with your private hosted zone

# Routing Policies

- **Simple routing policy** – Use for a single resource that performs a given function for your domain. If you choose the simple routing policy in the Route 53 console, you can specify multiple values in the same record, such as multiple IP addresses. If you specify multiple values in a record, Route 53 returns all values to the recursive resolver in random order, and the resolver returns the values to the client (such as a web browser) that submitted the DNS query. The client then chooses a value and resubmits the query.
- **Failover routing policy** – Use when you want to configure active-passive failover. Failover routing lets you route traffic to a resource when the resource is healthy or to a different resource when the first resource is unhealthy.



## Routing Policies

- **Geo-location routing policy** – Geo-location routing lets you choose the resources that serve your traffic based on the geographic location of your users, meaning the location that DNS queries originate from. For example, you might want all queries from Europe to be routed to an ELB load balancer in the Frankfurt region.
- **Latency routing policy** – Use when you have resources in multiple AWS Regions and you want to route traffic to the region that provides the best latency. If your application is hosted in multiple AWS Regions, you can improve performance for your users by serving their requests from the AWS Region that provides the lowest latency.

- **Multi-value answer routing policy** – Use when you want Route 53 to respond to DNS queries with up to eight healthy records selected at random. Multi-value answer routing lets you configure Amazon Route 53 to return multiple values, such as IP addresses for your web servers, in response to DNS queries. Multi-value answer routing also lets you check the health of each resource, so Route 53 returns only values for healthy resources.
- **Weighted routing policy** – Weighted routing lets you associate multiple resources with a single domain name (example.com) or subdomain name (acme.example.com) and choose how much traffic is routed to each resource



# Domain registration, Domain Hosting and Domain transfer

## **Domain registration:**

When you want to get a new domain name, such as the example.com part of the URL <http://example.com>, you can register it with Amazon Route 53.

You can also transfer the registration for existing domains from other registrars to Route 53 or transfer the registration for domains that you register with Route 53 to another registrar.

## **Domain Hosting:**

Domain hosting refers to businesses that specialize in hosting domain names for individuals and companies. Domain names are used in URLs to identify particular Web pages.

## **Domain transfer:**

A domain transfer refers to the process of changing the designated registrar of a domain name. ... Domain names may be transferred only if they have been registered with the previous registrar for 60 days or more .



# Global Accelerator

# Global Accelerator

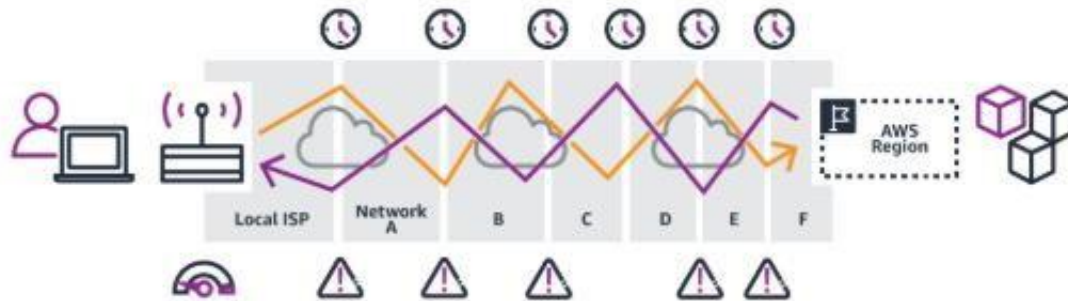
- AWS Global Accelerator is a service that improves the availability and performance of your applications with local or global users
- AWS Global Accelerator uses the AWS global network to optimize the path from your users to your applications, improving the performance of your traffic by as much as 60%
- AWS Global Accelerator continually monitors the health of your application endpoints and redirects traffic to healthy endpoints in less than 30 seconds

# Benefits

- Improve global application availability
- Accelerate your global applications
- Easily manage endpoints

# Without AWS Global Accelerator

It can take many networks to reach the application. Paths to and from the application may differ. Each hop impacts performance and can introduce risks



# With AWS Global Accelerator

Adding AWS Global Accelerator removes these inefficiencies. It leverages the Global AWS Network, resulting in improved performance.





**AWS Lambda**

# Introduction to Lambda

- AWS Lambda is a compute service that lets you run code without provisioning or managing servers
- AWS Lambda executes your code only when needed and scales automatically, from a few requests per day to thousands per second
- You pay only for the compute time you consume - there is no charge when your code is not running
- With AWS Lambda, you can run code for virtually any type of application or backend service - all with zero administration



- AWS Lambda runs your code on a high-availability compute infrastructure and performs all of the administration of the compute resources, including server and operating system maintenance, capacity provisioning and automatic scaling, code monitoring and logging
- All you need to do is supply your code in one of the languages that AWS Lambda supports
- You can use AWS Lambda to run your code in response to events, such as changes to data in an Amazon S3 bucket or an Amazon DynamoDB table

# When Should I Use AWS Lambda?

- AWS Lambda is an ideal compute platform for many application scenarios, provided that you can write your application code in languages supported by AWS Lambda, and run within the AWS Lambda standard runtime environment and resources provided by Lambda
- When using AWS Lambda, you are responsible only for your code
- AWS Lambda manages the compute fleet that offers a balance of memory, CPU, network, and other resources

- This is in exchange for flexibility, which means you cannot log in to compute instances, or customize the operating system on provided runtimes
- These constraints enable AWS Lambda to perform operational and administrative activities on your behalf, including provisioning capacity, monitoring fleet health, applying security patches, deploying your code, and monitoring and logging your Lambda functions
- Lambda is a highly available service
- Languages supported by AWS Lambda are .NET, Go, Java, Node.js, Python, Ruby

- If you need to manage your own compute resources, Amazon Web Services also offers other compute services to meet your needs
  - Amazon Elastic Compute Cloud (Amazon EC2) service offers flexibility and a wide range of EC2 instance types to choose from. It gives you the option to customize operating systems, network and security settings, and the entire software stack, but you are responsible for provisioning capacity, monitoring fleet health and performance, and using Availability Zones for fault tolerance

## AWS Lambda - Building Block

- Lambda Function : The foundation, it is comprised of your custom code and any dependent libraries
- Event Source : An AWS service, such as Amazon SNS, that triggers your function and executes its logic
- Downstream resources : An AWS service, such as Dynamo DB tables or Amazon S3 buckets, that your Lambda function calls once it is triggered
- Log Streams : While Lambda automatically monitors your function invocations and reports metrics to CloudWatch

# Event Source Mapping

- In AWS Lambda, Lambda functions and event sources are the core components
- An event source is the entity that publishes events, and a Lambda function is the custom code that processes the events
- Supported event sources refer to those AWS services that can be pre-configured to work with AWS Lambda
- The configuration is referred to as event source mapping which maps an event source to a Lambda function

- AWS Lambda supports many AWS services as event sources
- When you configure these event sources to trigger a Lambda function, the Lambda function is invoked automatically when events occur
- Some of the supported AWS Event sources for Lambda functions are
  - Amazon S3
  - Amazon DynamoDB
  - Amazon Simple Notification Service
  - Amazon SQS
  - Amazon Cloudwatch

# Lambda Function Configuration

Compute resources that you need

- You only specify the amount of memory you want to allocate for your Lambda function
- AWS Lambda allocates CPU power proportional to the memory
- You can update the configuration and request additional memory in 64 MB increments from 128 MB to 3008 MB
  - If the maximum memory use is exceeded, function invocation will be terminated
- Functions larger than 1536 MB are allocated multiple CPU threads, and multi-threaded or multi-process code is needed to take advantage of the additional performance



## ***Maximum execution time (timeout)***

- You pay for the AWS resources that are used to run your Lambda function
- To prevent your Lambda function from running indefinitely, you specify a timeout
- When the specified time out is reached, AWS Lambda terminates your Lambda function
- Default is 3 seconds, maximum is 900 seconds (15minutes)



AWS  
CloudFormation

# Overview of Cloud Formation

- AWS Cloud Formation provides a common language for you to describe and provision all the infrastructure resources in your cloud environment
- Cloud Formation allows you to use a simple text file to model and provision, in an automated and secure manner, all the resources needed for your applications across all regions and accounts
- This file serves as the single source of truth for your cloud environment.
- AWS Cloud Formation is available at no additional charge, and you pay only for the AWS resources needed to run your applications

# Benefits

- MODEL IT ALL

AWS Cloud Formation allows you to model your entire infrastructure in a text file. This template becomes the single source of truth for your infrastructure. This helps you to standardize infrastructure components used across your organization, enabling configuration compliance and faster troubleshooting.

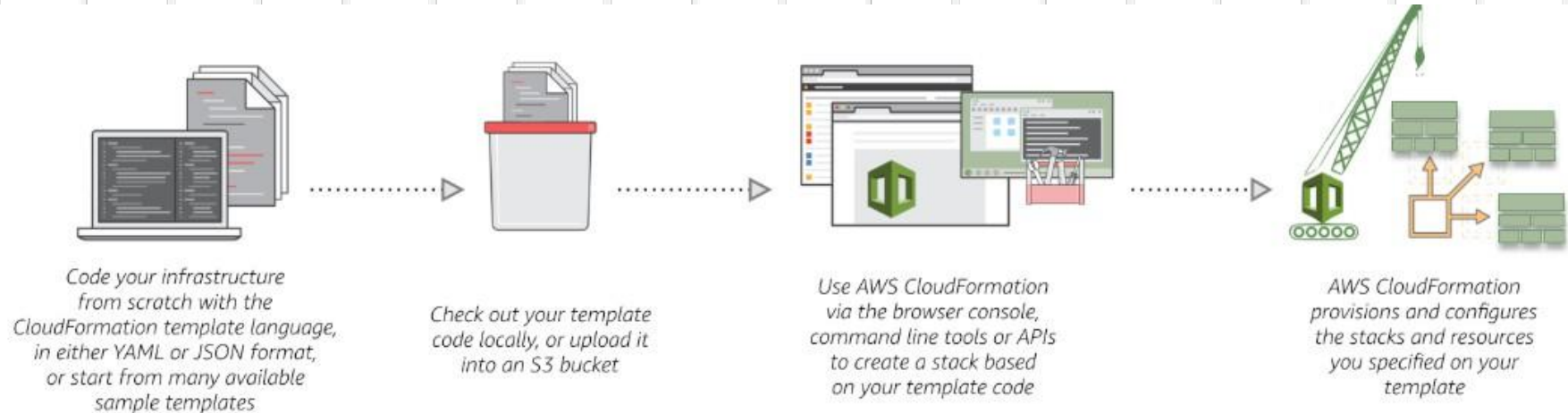
- AUTOMATE AND DEPLOY

AWS Cloud Formation provisions your resources in a safe, repeatable manner, allowing you to build and rebuild your infrastructure and applications, without having to perform manual actions or write custom scripts.

## - IT'S JUST CODE

Codifying your infrastructure allows you to treat your infrastructure as just code. You can author it with any code editor, check it into a version control system, and review the files with team members before deploying into production.

### How it Works



FSX 

# Introduction to FSx

- Amazon FSx makes it easy and cost effective to launch and run popular file systems
- With Amazon FSx, you can leverage the rich feature sets and fast performance of widely-used open source and commercially-licensed file systems, while avoiding time-consuming administrative tasks like hardware provisioning, software configuration, patching, and backups
- It provides cost-efficient capacity and high levels of reliability, and it integrates with other AWS services so that you can manage and use the file systems in cloud-native ways

Amazon FSx provides you with two file systems to choose from

- Amazon FSx for Windows File Server for business applications
- Amazon FSx for Lustre for high-performance workloads

# Amazon FSx for Windows File Server

- Amazon FSx for Windows File Server provides fully managed, highly reliable file storage that is accessible over the industry-standard Service Message Block (SMB) protocol
- It is built on Windows Server, delivering a wide range of administrative features such as user quotas, end-user file restore, and Microsoft Active Directory (AD) integration
- It offers single-AZ and multi-AZ deployment options, fully managed backups, and encryption of data at rest and in transit
- Amazon FSx file storage is accessible from Windows, Linux, and MacOS compute instances and devices running on AWS or on premises
- You can optimize cost and performance for your workload needs with SSD and HDD storage options
- Amazon FSx helps you optimize TCO with no data duplication, reducing costs by up to 50-60% on your general-purpose file shares



# Amazon FSx for Lustre for high-performance file system

- Amazon FSx for Lustre makes it easy and cost effective to launch and run the world's most popular high-performance file system, Lustre
- Use it for workloads where speed matters, such as machine learning, high performance computing (HPC), video processing, and financial modeling
- The open source Lustre file system was built for and is used for the world's most demanding workloads, and is the most widely used file system for the 500 fastest computers in the world
- Amazon FSx brings the Lustre file system to the masses by making it easy and cost effective for you to use Lustre for any workload where you want to process data as quickly as possible
- You can also link your FSx for Lustre file systems to Amazon S3, making it simple to process data stored on S3

# Benefits

- Simple and fully managed
- Fast and inexpensive
- Available and durable
- Integration with other AWS services

# **AWS Migration Services**

# Overview of Migration services

Moving your data and applications to the cloud isn't easy, but Amazon has a number of services that can take some of the load off. Some of them are :

## AWS Migration Hub

- AWS Migration Hub provides a single place to monitor migrations in any AWS region where your migration tools are available
- There is no additional cost for using Migration Hub. You only pay for the cost of the individual migration tools you use, and any resources being consumed on AWS

# Features

- Allows you to import information about their on-premises servers and applications into the Migration Hub so you can track the status of application migrations.
- Shows the latest status and metrics for your entire migration portfolio. This allows you to quickly understand the progress of your migrations, as well as identify and troubleshoot any issues that arise.
- Provides all application details in a central location. This allows you to track the status of all the moving parts across all migrations, making it easier to view overall migration progress

# AWS Database Migration Service

- AWS Database Migration Service helps you migrate databases to AWS quickly and securely
- Can migrate your data to and from most widely used commercial and open-source databases
- Supports homogeneous migrations such as Oracle to Oracle, as well as heterogeneous migrations between different database platforms, such as Oracle or Microsoft SQL Server to Amazon Aurora

# Benefits

- Simple to use
- Minimal downtime
- Supports widely used databases
- Low cost
- Fast and easy to setup
- Reliable

# Introduction to the Snow Family

- The Snow Family of services offers a number of physical devices and capacity points, including some with built-in compute capabilities
- These services help physically transport up to Exabyte's of data into and out of AWS
- The Snow Family of services are owned and managed by AWS and integrate with AWS security, monitoring, storage management and computing capabilities



# AWS Snowball Services

- AWS Snowball is a petabyte-scale data transport solution that uses secure appliances to transfer large amounts of data into and out of AWS
- The AWS Snowball service uses physical storage devices to transfer large amounts of data between Amazon Simple Storage Service (Amazon S3) and your onsite data storage location at faster-than-internet speeds
- Snowball uses multiple layers of security designed to protect your data including tamper-resistant enclosures, 256-bit encryption, and an industry-standard Trusted Platform Module (TPM) designed to ensure both security and full chain of custody of your data.
- Once the data transfer job has been processed and verified, AWS performs a software erasure of the Snowball appliance

# How it works



## Snowball Features

- 80 TB and 50 TB models are available in US Regions
- 50 -100 TB model available in all AWS Regions
- Enforced encryption protects your data at rest and in physical transit
- There's no need to buy or maintain your own hardware devices

# Benefits

- High speed & Extremely scalable
- Tamper resistant and secure
- Simple and compatible
- Easy data retrieval

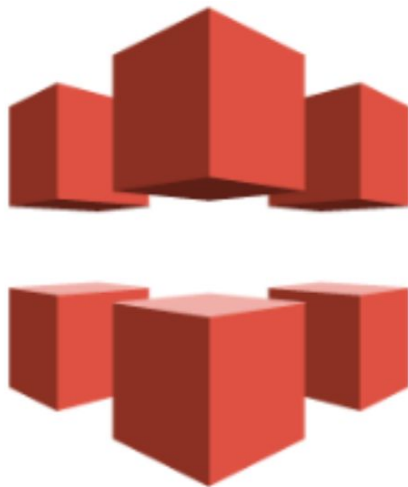
# Snowmobile

- The AWS Snowmobile moves up to 100PB of data (equivalent to 1,250 AWS Snowball devices) in a 45-foot long ruggedized shipping container and is ideal for multi-petabyte or Exabyte-scale digital media migrations and datacenter shutdowns
- A Snowmobile arrives at the customer site and appears as a network-attached data store for more secure, high-speed data transfer
- After data is transferred to Snowmobile, it is driven back to an AWS Region where the data is loaded into Amazon S3



# Key Features

- Faster data transfer
- Strong encryption
- Rugged, durable and Secure
- Customized for your needs
- Massively Scalable



**Cloudfront**

# Introduction to Cloud Front distributions and Cloud Front Edge locations

- Is a web service that speeds up distribution of your static and dynamic web content, such as .html, .css, .js, and image files, to your users
- - CloudFront delivers your content through a worldwide network of data centers called edge locations
- - When a user requests content that you're serving with CloudFront, the user is routed to the edge location that provides the lowest latency (time delay), so that content is delivered with the best possible performance

- If the content is already in the edge location with the lowest latency, CloudFront delivers it immediately
- If the content is not in that edge location, CloudFront retrieves it from an origin that you've defined—such as an Amazon S3 bucket, a Media Package channel, or an HTTP server (for example, a web server)
- You also get increased reliability and availability because copies of your files (also known as objects) are now held (or cached) in multiple edge locations around the world





# Storage Gateway

# AWS Storage Gateway

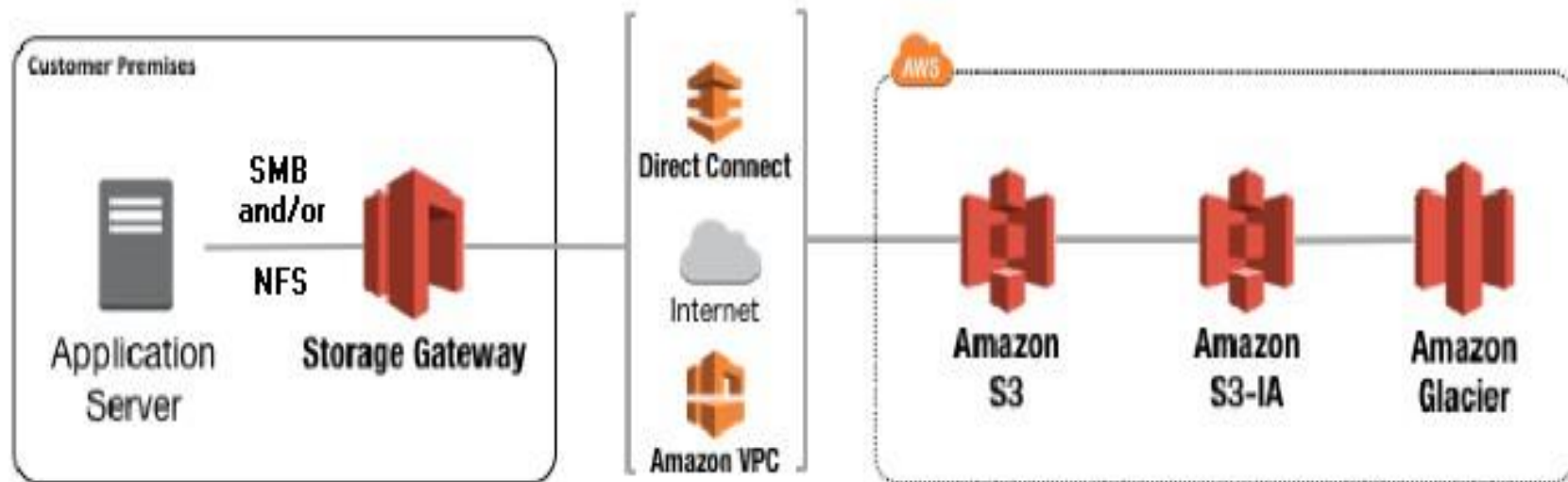
- AWS Storage Gateway connects an on-premises software appliance with cloud-based storage
- You can use the service to store data in the AWS Cloud for scalable and cost-effective storage that helps maintain data security

AWS Storage Gateway offers file-based, volume-based, and tape-based storage solutions.

# File Gateway

- A file gateway supports a file interface into Amazon Simple Storage Service (Amazon S3) and combines a service and a virtual software appliance
- By using this combination, you can store and retrieve objects in Amazon S3 using industry-standard file protocols such as Network File System (NFS) and Server Message Block (SMB)
- The gateway, is deployed into your on-premises environment as a virtual machine (VM) running on VMware ESXi or Microsoft Hyper-V hypervisor
- The gateway provides access to objects in S3 as files or file share mount points

## Simple Architecture



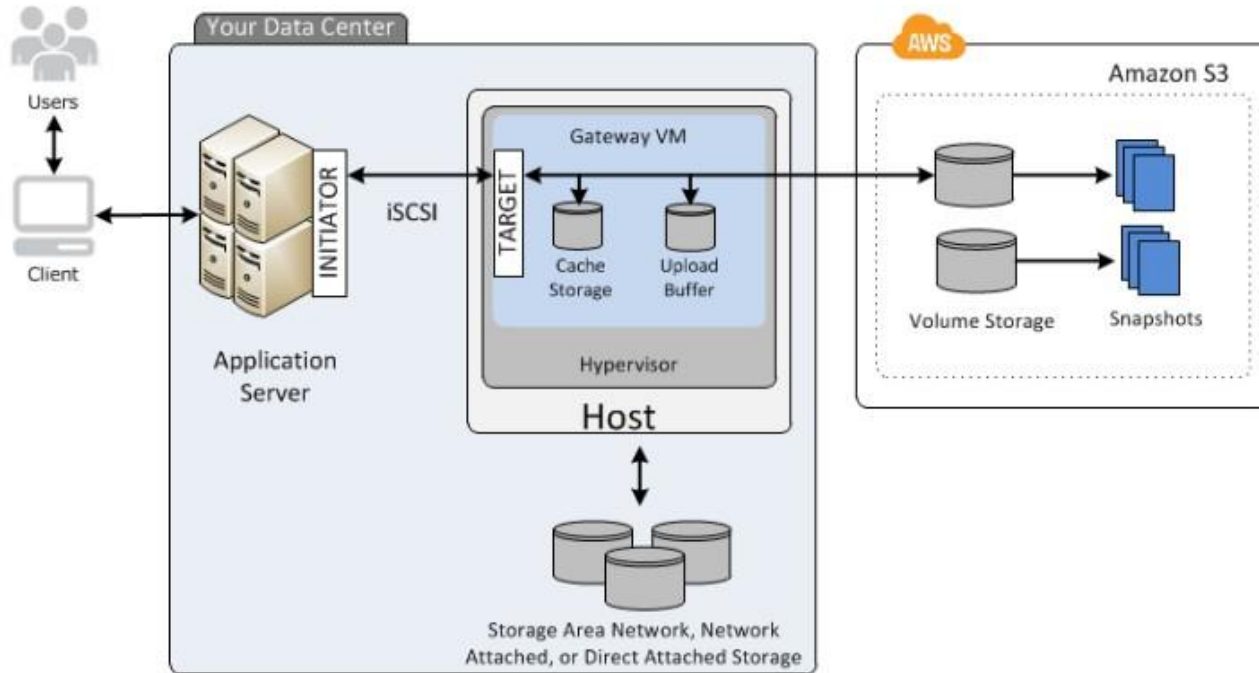
# Volume Gateway

A volume gateway provides cloud-backed storage volumes that you can mount as Internet Small Computer System Interface (iSCSI) devices from your on-premises application servers.

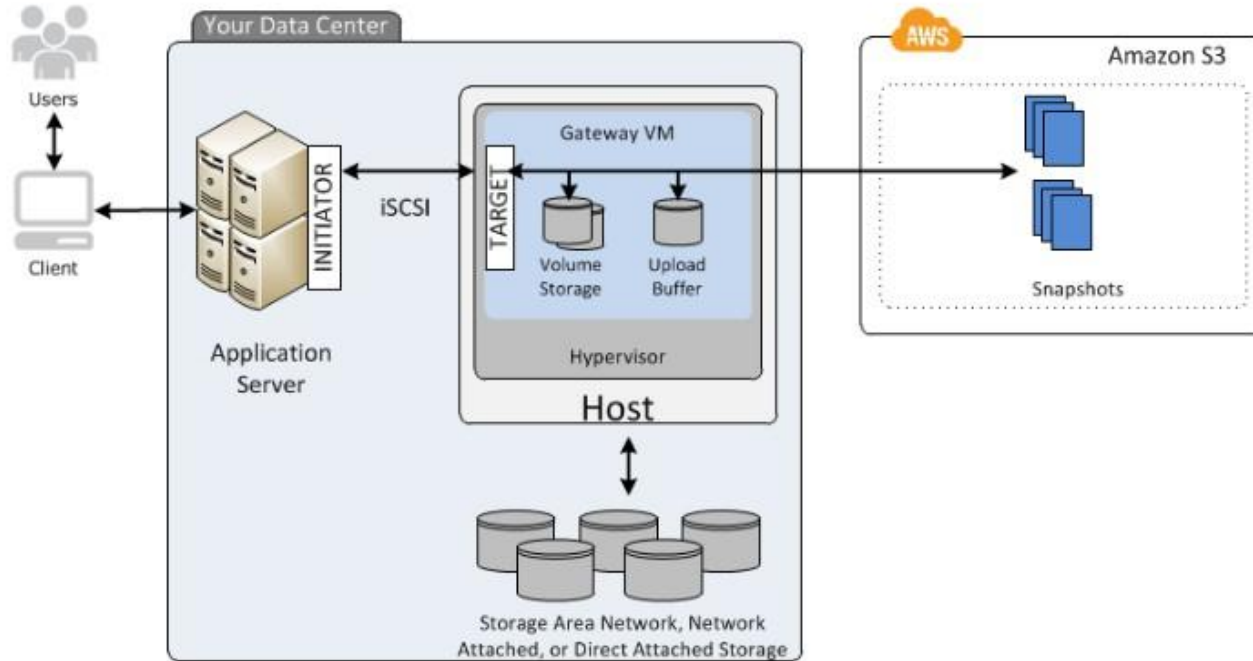
The gateway supports the following volume configurations

- Cached volumes
- Stored volumes

# Cache Volume Architecture



# Stored Volume Architecture

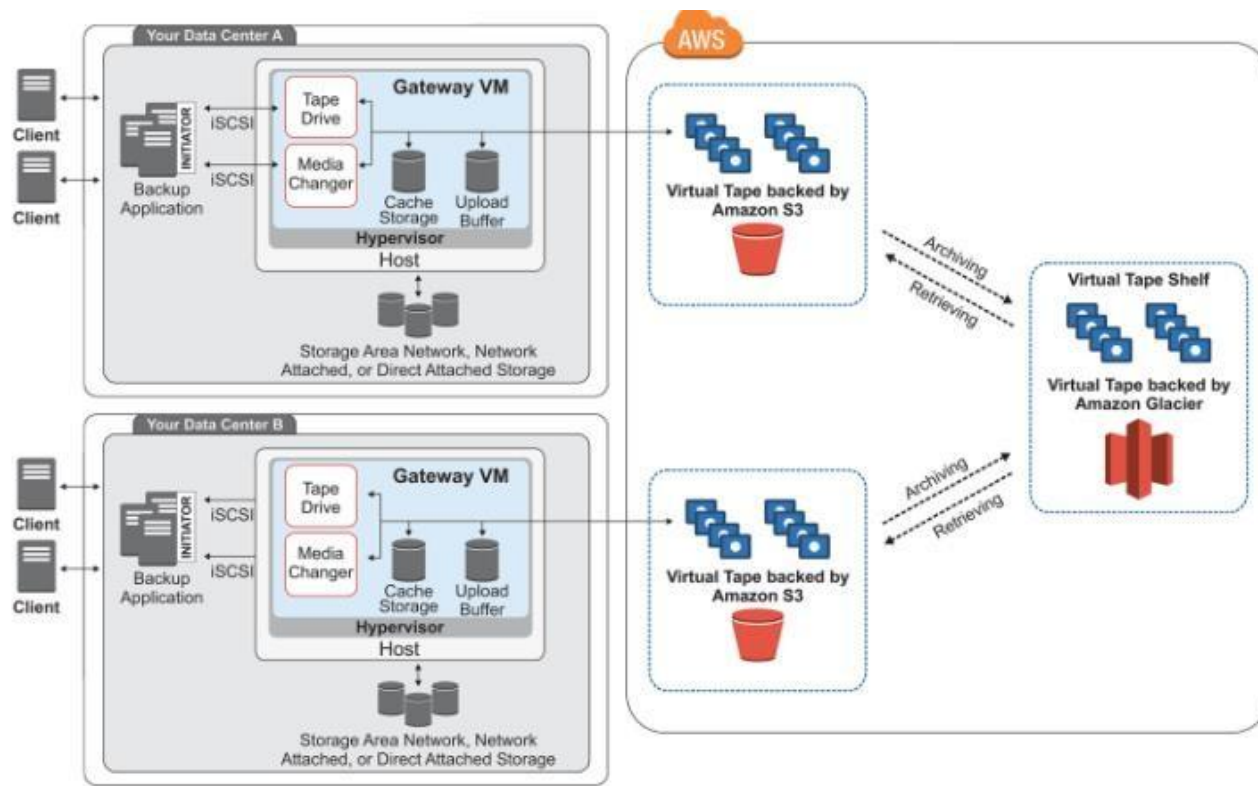


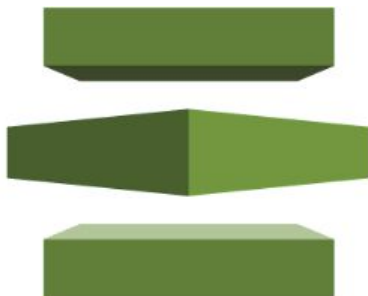
# Tape Gateway

- Can cost-effectively and durably archive backup data in Amazon S3 Glacier or S3 Glacier Deep Archive
- A tape gateway provides a virtual tape infrastructure that scales seamlessly with your business needs and eliminates the operational burden of provisioning, scaling, and maintaining a physical tape infrastructure
- You can run AWS Storage Gateway either on-premises as a VM appliance, as a hardware appliance, or in AWS as an Amazon Elastic Compute Cloud (Amazon EC2) instance



# Tape Gateway Architecture





**AWS Certificate Manager**

# AWS Certificate Manager

- AWS Certificate Manager is a service that lets you easily provision, manage, and deploy public and private Secure Sockets Layer/Transport Layer Security (SSL/TLS) certificates for use with AWS services and your internal connected resources
- SSL/TLS certificates are used to secure network communications and establish the identity of websites over the Internet as well as resources on private networks
- AWS Certificate Manager removes the time-consuming manual process of purchasing, uploading, and renewing SSL/TLS certificates

- With AWS Certificate Manager, you can quickly request a certificate, deploy it on ACM-integrated AWS resources, such as Elastic Load Balancers, Amazon CloudFront distributions, and APIs on API Gateway, and let AWS Certificate Manager handle certificate renewals
- It also enables you to create private certificates for your internal resources and manage the certificate lifecycle centrally

# Use Cases

Protect and secure your website: SSL, and its successor TLS, are industry standard protocols for encrypting network communications and establishing the identity of websites over the Internet

Protect and Secure your internal resources: Private certificates are used for identifying and securing communication between connected resources on private networks, such as servers, mobile and IoT devices, and applications