AWS Certified Cloud Practitioner (CLF-C01)

Credits

Some of these notes are adapted from Stephanie Marek's course CLF-C01 on Udemy. I also perused the internet for other open-source blogs and notes regarding the CLF-C01 exam. Some sections may be incomplete or outdated as Amazon frequently updates the exam verbiage and question pool.

General Information

AWS Certified Cloud Practitioner

Problems with traditional IT setups

- cost of renting data center
- costs for power supply, cooling, and maintenance
- adding/replacing hardware takes time/money
- scaling is limited and requires further installation/maintenance
- requires team to monitor infrastructure
- natural disasters or power shutdowns/intermittencies

Cloud Computing Fundamentals

Cloud computing (CC) is *on-demand delivery of compute power, database (db) storage, applications, and other IT resources.*

- Pay-as-you-go pricing
- Provision exactly the right type and size of computing resource you need
- Access resources almost instantly
- Simple way to access **servers**, **storage**, **databases** and a set of **application services**
- AWS owns and maintains network-connected hardware required for these services. You provision and use what you need via a web application (interface)

Five Characteristics of CC

1. On-demand self service
    - users provision resources and use them without human interaction
2. Broad network access
    - resources available over the network
    - can access via diverse client platforms
3. Multi-tenancy and resource pooling
    - multiple customers can share same infrastructure and applications with security

and privacy
    - multiple customers use services from the same physical resources
4. Rapid elasticity and scalability
    - automatically and quickly acquire and dispose resources when needed
    - quickly and easily scale based on demand
5. Measured service
    - usage is measured
    - users pay correctly for what they use/used

Six Advantages of CC

1. Trade capital expense (CAPEX) for operational expense (OPEX)
    - pay on-demand; no hardware owned
    - reduces Total Cost of Ownership (TCO) and OPEX
2. Benefit from massive economies of scale
    - prices reduced as AWS is efficient due to large scale
3. Stop guessing capacity
    - scaled based on actual measures usage
4. Increased speed and agility
5. No more spending on running/maintaining data centers
6. Global in minutes with AWS global infrastructure

CC solves the following problems

- flexibility
- cost-effectiveness
- scalability
- elasticity
- high-availability and fault-tolerance
- agility

Cloud Deployment Models

Private Cloud

- cloud services used by a *single* organization, not exposed to public
- complete control
- security for sensitive applications
- specific business needs
- e.g. *Rackspace*

Public Cloud

- cloud resources owned by a *third-party* service provider delivered over the internet
- e.g. Azure, Google Cloud, AWS

Hybrid Cloud

- some servers on-premises, some capabilities are on the Cloud
- you keep control over some private/sensitive assets in your private infrastructure

CC Types

1. Infrastructure as a Service (IaaS)
    - provide building blocks for cloud IT
    - provides networking, computers, data storage space
    - highest level of flexibility
    - easy parallel with traditional on-premises IT
    - e.g. Amazon EC2
2. Platform as a Service (PaaS)
    - removes need for your organization to manage underlying infrastructure
    - focus on deployment and management of your applications
    - e.g. Elastic Beanstalk
3. Software as a Service (SaaS)
    - completed product/application that is run and managed by service provider
    - e.g. Rekognition for ML

Cloud Pricing Overview

- *Compute*: pay for commute time
- *Storage*: pay for data stored in the Cloud
- *Data*: pay for data transfer *out* of the Cloud
    - **data transfer in is free**

Shared Responsibility Model

- Shared Responsibility Model - Amazon Web Services (AWS)
- **AWS responsibility "Security of the Cloud"** - AWS is responsible for protecting the infrastructure that runs all of the services offered in the AWS Cloud. This infrastructure is composed of the hardware, software, networking, and facilities that run AWS Cloud services. (Source: AWS)
- **Customer responsibility "Security in the Cloud"** – Customer responsibility will be determined by the AWS Cloud services that a customer selects. This determines the amount of configuration work the customer must perform as part of their security responsibilities. For example, a service such as Amazon Elastic Compute Cloud (Amazon EC2) is categorized as Infrastructure as a Service (IaaS) and, as such, requires the customer to perform all of the necessary security configuration and management tasks. Customers that deploy an Amazon EC2 instance are responsible for

management of the guest operating system (including updates and security patches), any application software or utilities installed by the customer on the instances, and the configuration of the AWS-provided firewall (called a security group) on each instance. For abstracted services, such as Amazon S3 and Amazon DynamoDB, AWS operates the infrastructure layer, the operating system, and platforms, and customers access the endpoints to store and retrieve data. Customers are responsible for managing their data (including encryption options), classifying their assets, and using IAM tools to apply the appropriate permissions. (Source: AWS)

IAM (Identity and Access Management)

- **Global** service
- **Root account is created by default**
- **Users** are people within your organization; can be grouped
  - Do not have to belong to a group, but can belong to many groups
- **Groups** only contain users
- **Least privilege principle** when applying permissions to users

Policy Inheritance

- Users inherit the policies/permissions of the group(s) they are part of
- Users not in any group have *inline* policies

Example policy:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowViewAccountInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetAccountPasswordPolicy",
                "iam:GetAccountSummary"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowManageOwnPasswords",
            "Effect": "Allow",
            "Action": [
                "iam:ChangePassword",
```

```
                "iam:GetUser"
            ],
            "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
    ]
}
```

IAM Best Practices

- Do not use *root* account except for AWS account setup
- One AWS user per physical user
- Assign users to *groups* and assign *permission* to groups
- Create a strong *password policy*
- Use and enforce *Multi factor Authentication* (MFA)
- Create and use *roles* for giving permission to AWS services
- Use *Access Keys* for programmatic access (CLI/SDK)
- Audit permissions with the IAM Credentials Report
- *Never* share IAM users and Access Keys

Shared Responsibility Model for IAM

AWS

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

You

- Users, groups, roles, management and monitoring
- Enable MFA on all accounts
- Rotate all keys often
- Use IAM tools to apply appropriate permissions
- Analyze access patterns and review permissions

Elastic Compute Cloud (EC2)

EC2 is IaaS consisting of:

- renting virtual machines (EC2)
- storing data on virtual drives (EBS)
- distributing load across machines (ELB)
- scaling services using an auto-scaling group (ASG)

EC2 Config Options

- OS: Linux, Windows, MacOS
- CPU power
- RAM amount
- Storage space
  - network-attached (EBS and EFS)
  - hardware (EC2 Instance Store)
- Network card: speed of card; public IP address
- firewall rules: **security group**
- Bootstrap script (configure at first launch): **EC2 User Data**
  - *bootstrapping* means launching commands when a machine starts
  - Bootstrap script runs once at instance *first start*
  - EC2 user data can automate tasks such as:
    - installing updates and/or software
    - downloading common files from internet
    - almost anything else we need
  - EC2 User Data Script runs with the root user

EC2 Instance naming convention

[x][y].[z] where:

- x: lower-case letter indicating *instance class*
- y: integer indicating *generation* (AWS improves and increments over time)
- z: *size* within the instance class
- example: m5.2xlarge

EC2 Instance Types

1. General Purpose
   - great for workload diversity such as web servers or code repos
   - balance between compute, memory, and networking
   - e.g. t2.micro
2. Compute Optimized
   - great for compute-intensive tasks that require high performance processors
     - batch processing workloads
     - media transcoding
     - high performance web servers
     - high-performance computing (HPC)
     - scientific modeling and ML
     - dedicated gaming servers
     - class name *C*
3. Memory Optimized
   - fast performance for workloads that process large data sets in memory
   - use cases:

    - high performance, relational/non-relational databases
    - distributed web scale cache stores
    - in-memory databases optimized for business intelligence (BI)
    - apps performing real-time processing of big unstructured data
    - class name *R*
4. Storage Optimized
    - great for storage-intensive tasks that require high, sequential read/write access to large data sets on local storage
    - use cases:
        - high-frequency online transaction processing (OLTP) systems
        - relational and NoSQL databases
        - cache for in-memory databases (e.g. Redis)
        - data warehousing applications
        - distributed file systems

EC2 Security Groups

- security groups are fundamental to AWS network security
- control inbound/outbound traffic to EC2 instances
- only contain *allow* rules
- can reference by IP or by security group
- act as a "firewall" on EC2 instances
- regulate:
    - access to ports
    - authorized IP ranges (IPv4 and IPv6)
    - control of inbound/outbound network
- can be attached to multiple instances
- locked down to a region/VPC combination
- *external* to the EC2; if traffic is blocked then the EC2 will *not* see it
- *good practice to maintain one separate security group for SSH access*
- application *time-out* generally implies a security group issue
- *connection refused* error implies and application error, failure to launch or non-launched application
- *all inbound traffic is blocked by default*
- *all outbound traffic is allowed by default*

Common Ports

- 21 | FTP (File Transfer Protocol) | upload files into a file share
- 22 | SSH (Secure Shell) | log into a Linux instance
- 22 | SFTP (Secure File Transfer Protocol) | upload files using SSH
- 80 | HTTP | access unsecured websites
- 443 | HTTPS | access secured websites

- 3389 | RDP (Remote Desktop Protocol) | log into a Windows instance

SSH into EC2

```
>> ssh -i <path-to-pem> ec2-user@<public-IPv4>
```

EC2 Purchasing Options

1. On-Demand Instances
    - short workload, predictable pricing
    - recommended for *short-term* but *un-interrupted* workloads where you cannot predict how the app will behave
    - highest cost but no upfront payment
    - no long-term commitment
    - pay for what you use
        - Linux/Windows: billing per second after first minute
        - All other OS: billing per hour
2. Reserved
    - 1 year or 3 years (must choose 1 or 3 years)
    - *Reserved instances*: long workloads
    - *Convertible reserved instances*: long workloads with flexible instances
    - *Scheduled reserved instances*: scheduled reserved (e.g. every Thursday between 3pm and 6pm)
    - up to 75% discount compared to On-Demand
    - *reservation period* and *purchasing option* relates to discount percentage
        - 1 year = small discount
        - 3 years = larger discount
        - no upfront = small discount
        - partial upfront = larger discount
        - all upfront = largest discount
    - reserve a specific instance type
    - recommend for steady-state usage applications (e.g. database)
    - Convertible Reserved Instances
        - can change the EC2 instance type over time
        - up to 54% discount
    - Scheduled Reserved instances (deprecated)
        - launch within the reserved time window
        - good for when you require a fraction of day/week/month
        - still need to commit over 1 year to 3 years
3. Spot Instances
    - short workloads; cheap and lose instances (i.e. less reliable)
    - *up to 90%* discount compared to On-Demand
    - you can “lose” instance at any point in time if your max price is less than the

current spot price
    - *most cost-efficient* instances in AWS
    - *useful for workloads that are resilient to failure*
        - batch jobs
        - data analysis
        - image processing
        - any *distributed* workloads
        - workloads with a flexible start and end time
    - *not suitable for critical jobs or databases*
4. Dedicated Host
    - a physical server with EC2 instance capacity fully dedicated to your use
    - can help address *compliance requirements* and reduce costs by allowing you to use your *existing server-bound software licenses*
    - book an entire physical server and control the instance placement
    - allocated for your account for a 3-year reservation period
    - more expensive
    - useful for software with complicated licensing model (Bring Your Own License | BYOL)
    - useful for companies that have strong regulatory or compliance needs
5. EC2 Dedicated Instances
    - instances running on hardware that is dedicated to you
    - may share hardware with another instance in same account
    - no control over instance placement (can move hardware after Stop/Start)

EC2 Shared Responsibility Model

- AWS
    - infrastructure (global network security)
    - isolation on physical hosts
    - replacing faulty hardware
    - compliance verification
- Customer
    - Security Groups rules
    - operating-system patches and updates
    - software and utilities installed on EC2 instance
    - IAM Roles assigned to EC2 and IAM user access management
    - data security on your instances

EC2 Storage

High-Performance Block Storage – Amazon EBS – Amazon Web Services

Elastic Block Storage (EBS) Volume

- an EBS volume is a network drive (not a physical drive) you can attach to EC2 instances while they run
  - uses the network to communicate with the instance; there might be some latency because of this
  - can be detached from an EC2 instance and attached to another one quickly
- allows instances to persist data even after termination
- provisioned capacity
  - size is in GBs or I/O Operations per second (IOPS)
  - customer is billed for all provisioned capacity
  - customer can increase capacity of the drive over time
- can only be mounted to *one instance at a time* (at the CCP level)
- bound to a *specific Availability Zone (AZ)*
  - e.g. an EBS Volume in us-east-1a cannot be attached to us-east-1b
  - to move a volume across zones, we must first **snapshot** the volume
- think of them as a "network USB stick"
- AWS Free Tier: 30 GB of free EBS General Purpose (SSD) storage per month
- *Delete on Termination* attribute
  - controls EBS behavior when EC2 instance terminates
  - default: ON for root volume and OFF for non-root volumes
  - can be access by AWS CLI or console
  - use case for *disabling* delete on termination: preserve root volume when instance is terminated

EBS Snapshots

- snapshot is a *backup* of your EBS volume at a point in time
- linked to a Region
- used to restore an EBS Volume
- not necessary to detach volume to create snapshot, but recommended
- can copy snapshots across AZ or Region

Amazon Machine Image (AMI)

- AMI are a customization of an EC2 instance
  - customer adds their own software, configuration, OS, monitoring, etc.
- built for a *specific region* but can be copied across regions
- can launch EC2 instances from:
  - *Public AMI* provided by AWS
  - your *own AMI* that you create and painting yourself
  - *AWS Marketplace AMI*: AMI someone else made (and potentially sells)
- AMI Process

1. Start an EC2 instance and customize it
2. Stop the instance
3. Build an AMI (this also creates EBS snapshots)
4. Launch instance from other AMIs

EC2 Image Builder

EC2 Image Builder

- used to automate the create of Virtual Machines or other container images
- automate the creation, maintenance, validation, and testing of EC2 AMIs
- can be run on a schedule (e.g. weekly, upon package update, etc.)
- free service (only pay for underlying resources such as the EC2 instance itself)

EC2 Instance Store

- EBS volumes are network drives with *good but limited performance*
- *EC2 Instance Store* is for high-performance hardware
- better I/O performance
- **ephemeral**: lose their storage if stopped
- good for buffer, cache, scratch data, temporary content
- risk of data loss if hardware fails

Elastic File System (EFS)

- Managed network file system (NFS) that can be mounted on *hundreds* of EC2s simultaneously
- works with Linux EC2 instances and across multiple AZs
- highly available, scalable, expensive (~3x gp2 storage), pay-per-use, no capacity planning

> After you create an Amazon EFS file system, you can create mount targets. For Amazon EFS file systems that use Standard storage classes, you can create a **mount target** in each Availability Zone in an AWS Region. For EFS file systems that use One Zone storage classes, you can only create a single mount target in the same Availability Zone as the file system. Then you can mount the file system on compute instances, including Amazon EC2, Amazon ECS, and AWS Lambda in your virtual private cloud (VPC).

EFS Infrequent Access (EFS-IA)

- *storage class* that is cost-optimized for files not access daily
- up to 92% lower cost compared to EFS standard
- EFS will automatically move files to EFS-IA based on the last time they were accessed
- enable EFS-IA with a **Lifecycle Policy**
- transparent to the apps accessing EFS

EC2 Storage Shared Responsibility Model

- AWS

    - infrastructure
    - replication for data for EBS volumes and EFS drives
    - replacing faulty hardware
    - ensuring their employees cannot access customer data
- Customer
    - setting up backup and snapshot procedures
    - setting up data encryption
    - responsibility of any data on the drives
    - understanding risk of using EC2 Instance Store

Amazon Fix

- *launch third-party high-performance file systems on AWS*
- fully managed service
- FSx for Lustre
    - fully managed, high-performance, scalable *file storage for High Performance Computing (HPC)*
    - name derived from "Linux" and "cluster"
    - good for machine learning, analytics, video processing, financial modeling
    - scales up to 100s GB/s, millions of IOPS, sub-millisecond latencies
- FSx for Windows File Server
    - fully managed, highly reliable, and scalable Windows native shared file system
    - built on *Windows File Server*
    - supports SMB protocol and Windows NTFS
    - integrate with Microsoft Active Directory
    - can be accessed from AWS or your on-premise infrastructure
- FSx for NetApp ONTAP (not important for exam)

Elastic Load Balancing and Auto-Scaling Groups

Terms:

- **Scalability**: ability to accommodate a larger load by making the hardware stronger (scale up) or by adding nodes (scale out)
- **Elasticity**: once a system is scalable, elasticity means there will be some "auto-scaling" so that the system can scale based on load. This is "cloud-friendly": pay-per-use, match demand, optimize costs
- **Agility** (not related to scalability - *distractor*): new IT resources are only a click away. Reduces time to make those resources available to developers
- Scalability means an application/system can handle greater loads by adapting
- Scalability is linked but different to High Availability

Vertical Scalability

- Vertical means increasing the size of the instance (scale up/down)
  - E.g. change EC2 instance from t2.micro to t2.large
- common for non-distributed systems such as a database
- hardware limit on how much we can vertically scale

Horizontal Scalability

- Horizontal means increasing the number of instances/systems for your application (scale in/out)
  - Auto-scaling group
  - load balancer
- Implies distributed systems
- Common for web apps or modern apps

High Availability

- usually goes hand-in-hand with horizontal scaling
- High availability means running your app/system in **at least 2 availability zones**
  - Auto-scaling group multi AZs
  - Load Balancer multi AZs
- goal is to survive a data center loss (disaster)

Load Balancing

- load balancers are servers that forward internet traffic to multiple servers (EC2 Instances) downstream.

Motivations:

- spread load across multiple downstream instances
- expose a single point of access (DNS) to your app
- seamlessly handle failures of downstream instances
- regularly check health of instances
- provide SSL termination (HTTPS) for your websites
- high availability across zones

Elastic Load Balancer (ELB)

- ELB is a managed load balancer
  - AWS guarantees that is will work
  - AWS takes care of upgrades, maintenance, high availability
  - AWS provides a few configuration knobs
- Cheaper to set up your own custom load balancer but it requires much more maintenance and effort to integrate/configure

Three major kinds of load balancers offered by AWS:

- Application Load Balancer (HTTP/HTTPS only) - Layer 7
- Network Load Balancer (ultra-high performance; allows for TCP) - Layer 4
- Classic Load Balancer (slowly retiring) - Layer 4 and 7
- Gateway Load Balancer (new released Nov. 2020): Deploy, scale, and run third-party

virtual appliances

Auto Scaling Group (ASG)

Goals:

- Scale out (add EC2 instances) to match increased load
- Scale in (remove EC2 instances) to match decreased load
- ensure we have a minimum and maximum number of machines running
- automatically register new instances to a load balancer
- replace unhealthy instances

Cost Savings: only run at an optimal capacity

Works well with a load balancer

Strategies:

- **Manual Scaling**: update the size of ASG manually
- **Dynamic Scaling**: respond to changing demand
  - *Simple/Step Scaling*
    - When a CloudWatch alarm is triggered (e.g. CPU > 70%): add 2 units
    - When a CloudWatch alarm is triggered (e.g. CPU < 30%): remove 1 unit
  - *Target Tracking Scaling*
    - E.g. I want the average ASG CPU to be ~40%
  - *Scheduled Scaling*
    - Anticipate scaling based on known usage patterns
    - E.g. increase min. Capacity to 10 at 5pm on Fridays
- **Predictive Scaling**
    - Uses machine learning (ML) to predict future traffic ahead of time
    - automatically provisions right number of EC2 instances in advance
    - useful when your load has predictable time-based patterns

Simple Storage Service (S3)

- One of the major building blocks of AWS; known as “infinitely scalable”

Use Cases:

  - backup and storage
  - disaster recovery
  - archive
  - hybrid cloud storage
  - application hosting
  - media hostage
  - data lakes and big data analytics
  - software delivery
  - static website
- S3 allows people to store objects (files) in “buckets” (directories)

- buckets must have a **globally unique name (across all regions & all accounts)**
- buckets are defined at the *region level*
- S3 looks like a global service but buckets are created in a region
- Naming convention
  - no uppercase
  - no underscore
  - 3-63 characters long
  - NOT an IP
  - must begin with lowercase letter or number

S3 Objects

- Objects (files) have a *key*
- Key is the **full** path
  - **Key is composed of prefix + object name**
  - `s3://my-bucket/my_file.txt`
    - my_file.txt is the key
  - `s3://my-bucket/my_folder1/another_folder/my_file.txt`
    - my*folder1/another*folder/ is the *prefix*
    - my_file.txt is the *object name*
  - No concept of "directories" within buckets (although the UI will trick you to think otherwise)
- Object values for the content of the body
  - Max object size is 5TB (5000 Gb)
  - if uploading > 5 Gb you must use a *multi-part upload*
- Objects contain **metadata**
  - list of key/value pairs
  - can be system or user metadata
- Objects contain **tags**
  - unicode key/value pairs
  - up to 10 per object
  - useful for security/lifecycle
- Objects have a **Version ID** (if versioning is enabled)

S3 Security

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

- an IAM principal can access an S3 object if:
  - the user IAM permission allow it *OR* the resource policy ALLOWS it
  - *AND* there is no explicit deny
- **Encryption**: encrypt objects in Amazon S3 using encryption keys
- *Cross-Account* Access ➜ use Bucket Policy
- S3 Bucket policies
  - *JSON-based* policies
  - *Resources*: buckets and objects
  - *Actions*: set of API to Allow or Deny
  - *Effect*: Allow/Deny
  - *Principal*: account or user to apply the policy to
- Use S3 bucket policies to:
  - grant public access to the bucket
  - force objects to be encrypted at upload
  - grant access to another account (Cross-Account)
  - NOTE: there are ~5 default settings to *block public access* and to *prevent company data leaks*
    - If you know your bucket should *never* be public, leave these ON

S3 Websites

- S3 can host static websites and have them accessible on the world-wide web (WWW)
- URL will be one of these:
  - <bucket-name>.s3-website-<AWS-region>.amazonaws.com
  - <bucket-name>.s3-website.<AWS-region>.amazonaws.com
- if 403 (forbidden) error - make sure **bucket policy allows public reads**

S3 Versioning

- version is enabled at **bucket level**
- same key overwrite will increment the **version**
- bets practice to version your buckets:
  - protect against unintended deletes (ability to restore a version)
  - easy to roll back to previous version
- Notes
  - any file that is NOT versioned prior to enabling versioning will have a “null” version
  - suspending versioning does *not* delete the previous versions

S3 Access Logs

- for audit purposes, you may want to **log all access to S3 buckets**
- can log any request made to S3 from any account, authorized or denied
- **data is logged into another S3 bucket**
- helpful to determine the root cause of an issue or to audit usage, view suspicious

patterns, etc…

S3 Replication

- Crosds-Region Replication (CRR)
- Same-Region Replication (SRR)
- **must enable versioning in source and destination buckets**
- buckets can be in different accounts
- copying is asynchronous
- must give proper IAM permissions to S3

CRR Use Cases:

- compliance
- lower latency access
- replication across accounts

SRR Use Cases:

- log aggregation
- live replication between test and production accounts

S3 Storage Classes

Durability:

- high durability (99.99999999999%) of objects access multiple AZs
- if you store 10,000,000 objects with Amazon S3, you can, on average, expect to incur a loss of a single object once every 10,000 years
- same for all storage classes

Availability:

- Measures how readily available a service is
- S3 standard has 99.99% availability meaning it will not be available for 53 minutes per year
- varies depending on storage class

1. Amazon S3 Standard
    - 99.99% availability
    - used for frequently accessed data
    - low latency and high throughput
    - can sustain 2 concurrent facility failures
    - use cases: Big Data analytics, mobile/gaming applications, content distribution
2. Amazon S3 Standard - Infrequent Access (IA)
    - suitable for data that is less frequently access but requires rapid access when needed
    - 99.9% availability
    - lower cost compared to S3 Standard but incurs a *retrieval fee*
    - use cases: data store for disaster recovery, backups
3. Amazon S3 One Zone - Infrequent Access

    - same IA but data is stored in a *single AZ*
    - 99.5% availability
    - low latency and high throughput
    - lower cost compared to S3-IA (by 20%)
    - use cases: storing secondary backup copies of on-premise data; storing data you can recreate
4. Amazon S3 Intelligent Tiering
    - 99.9% availability
    - same low latency and high throughput performance of S3 Standard
    - **Cost-optimized** by automatically moving objects between two access tiers based on changing access patterns:
        - frequent access
        - infrequent access
    - resilient against events that impact an entire AZ

Glacier and Glacier Deep Dive

- low cost object storage (in Gb/month) meant for archiving/backup
- data is retrieved for longer term (years)
- various retrieval options of time + fees for retrieval

5. Amazon Glacier (cheap)
    - Expedited (1 to 5 minutes)
    - Standard (3 to 5 hours)
    - Bulk (5 to 12 hours)
6. Amazon Glacier Deep Archive (cheapest)
    - Standard (12 hours)
    - Bulk (48 hours)

- Amazon S3 Reduced Redundancy Storage (deprecated and not important)

Moving between storage classes

- We can transition objects between storage classes
- moving objects can be automated using a **lifecycle configuration**
- For infrequently accessed objects, move them to STANDARD_IA (infrequent access)
- for archive objects we do not need in real time, move to GLACIER or DEEP_ARCHIVE

S3 Object Lock & Glacier Vault Lock

S3 Object Lock

- adopt a WORM (write once read many) model
- block an object version from deletion for a specified amount of time

Glacier Vault Lock

- adopt a WORM (write once read many) model
- lock the policy for future edits (cannot be changed)
- helpful for compliance and data retention

S3 Encryption

1. No Encryption
2. Server-Side Encryption
    - server encrypts file after receiving it
3. Client-Side Encryption
    - user (client) encrypts file before uploading it

Snow Family

- highly secure, portable devices to **collect and process data at the edge** and to **migrate data into and out of AWS**
- AWS Snow Family are offline device to perform data migrations
- Data migration: use Snowcone, Snowball Edge, Snowmobile
- Edge computing: use Snowcone or Snowball Edge
- Rule of thumb: if it takes *more than a week* to transfer data over the network: use Snowball devices

Snowball Edge

- for data transfers in or out of AWS
- move TBs or PBs of data in or out of AWS
- alternative to moving data over the network (and paying network fees)
- pay per data transfer job
- provide block storage and Amazon S3-compatible object storage

Snowball Edge Storage Optimized

- *80 TB* of HDD capacity for block volume and S3 compatible object storage

Snowball Edge Compute Optimized

- *42 TB* of HDD capacity for block volume and S3 compatible object storage

Use cases: large data cloud migration, data center decommissions, disaster recovery

Snowcone

- small, portable computing anywhere, rugged and secure, withstands hard environments
- light (4.5 pounds)
- device used for edge computing, storage, and data transfer
- 8 TBs of *usable* storage | up to 24 TB of *migration* storage
- use Snowcone when Snowball cannot fit/survive (e.g. space or weight-constrained environment)
- must provide your own battery/cables
- can be sent back to AWS offline or can connect it to internet and use **AWS DataSync** to send data

Snowmobile

- transfer exabytes of data (1 EB = 1,000 PBs = 1,000,000 TBs)
- each snowmobile has 100 PB of capacity (use multiple in parallel)

- high security: temperature controlled, GPS, 24/7 video surveillance
- better than Snowball if transferring *more than 10 PBs*

Usage Process

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

Edge Computing

- process data while it is being created on an **edge location**
- edge locations are locations far away from the cloud
  - may have limited/no internet access
  - may have limited/no easy access to computing power
  - e.g. truck on the road, ship at sea, mining station underground

We step a **Snowball Edge/Snowcone** device to do edge computing

Use cases of edge computing:

- preprocess data
- machine learning at the edge
- transcoding media streams

Eventually (if necessary) we can ship device back to AWS (to transfer data)

Edge Computing with Snow Family

1. Snowcone (smaller)
   - 2 CPUs; 4 Gb memory; wired or wireless access
   - USB-C power using cable or optional battery
2. Snowball Edge - Compute Optimized
   - 52 vCPUs; 208 Gib RAM
   - Optional GPU for video processing or machine learning
   - 42 TB of HDD usable storage
3. Snowball Edge - Storage Optimized
   - Up to 40 vCPUs; 80 GiB RAM
   - 80 TB HDD usable storage
   - object storage cluttering available

All three can run EC2 instances and AWS Lambda functions (using AWS IoT Greengrass)

Long-term deployment options: 1 and 3 years discounted pricing

AWS OpsHub

Historically, to use Snow Family devices you needed a CLI. Now, we can use AWS OpsHub (a software we install in our computers) to manage Snow Family device(s)

- unlocking and configuring single or clustered devices

- transferring files
- launching and managing instances running on Snow Family devices
- Monitor device metrics (storage capacity, active instances on your device)
- launch compatible AWS services on your devices
    - e.g. Amazon EC2 instances, AWS DataSync, Network File System (NFS)

Hybrid Cloud for Storage

AWS is pushing for "hybrid cloud"

- part of infrastructure is on-premises
- part of infrastructure is on the cloud

Can be due to:

- long cloud migrations
- security requirements
- compliance requirements
- IT strategy

S3 is a proprietary storage technology (unlike EFS/NFS) so we need **AWS Storage Gateway** to expose S3 data on-premise.

AWS Storage Gateway

- bridge between on-premise data and cloud data in S3
- **hybrid storage service to allow on-premises to seamlessly use the AWS Cloud**
- use cases: disaster recovery, backup and restore, tiered storage

Types of Storage Gateway (do need to know details for exam)

- File Gateway
- Volume Gateway
- Tape Gateway

Databases and Analytics

Storing data on disk (EFS, EBS, EC2 Instance Store, S3) can have its limits. Sometimes we want to store data in a database. We may want to **structure** the data and build **indices** to efficiently **query/search** through the data. We can define **relationships** between our datasets (relation databases).

Databases are *optimized for a purpose* and come with different features, shapes, and constraints.

Relational Databases

SQL, PostgreSQL, etc…

Can perform queries or lookups using SQL logic/syntax.

NoSQL Databases

Non-relational databases (non-SQL). Built for specific data models and have flexible schemas for building modern apps.

Benefits:

- flexibility: easy to evolve data model
- scalability: designed to scale out by using distributed clusters
- high-performance: optimized for specific data model
- highly functional: types optimized for the data model

examples: key-value, document, graph, in-memory, search databases

JavaScript Object Notation (JSON) is common NoSQL data form

- data can be *nested*
- fields can *change* over time
- support for new types such as *arrays*

Shared Responsibility for Databases on AWS

- AWS offers use to manage different databases
- benefits:
  - quick provisioning; high availability; vertical and horizontal scaling
  - automated backup & restore; operations; upgrades
  - operating system patching handled by AWS
  - monitoring and alerts
  - NOTE: many database technologies could be run on EC2 but you must hand the resiliency, backup, patching, availability, fault tolerance, scaling, etc…

RDS

- Relational Database Services (RDS)
- Managed DB service that uses SQL query language
- allows us to create TBs in the cloud that are managed by AWS
  - MySQL
  - PostreSQL
  - MariaDB
  - Oracle
  - Microsoft SQL Server
  - Aurora (proprietary AWS database)
- Advantages
  - RDS is a *managed* service
    - automated provisioning; OS patching
    - continuous backups and restore to specific timestamp (Point in time Restore)
    - monitoring dashboards
    - read replicas for improved read performance
    - Multi AZ setup for disaster recovery
    - maintenance windows for upgrades
    - scaling capability (vert and horiz)
    - storage backed by EBS (gp2 or io 1)

- Cannot SSH into RDS instances
- **Aurora is a proprietary technology from AWS**
  - Aurora supports PostgreSQL and MySQL
  - Aurora is "AWS cloud optimized" and claims 5x performance improvement over MySQL on RDS; over 3x performance of PostgreSQL on RDS
  - Aurora storage automatically grows in increments of 10 GB up to 64 TB
  - Aurora costs more than RDS (20% more) but is more efficient
  - NOT in free tier

RDS Deployment

Read Replicas:

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

Multi-AZ:

- *failover* in case of AZ outage (high availability)
- *Data is only read and written to main DB*
- Can only have 1 AZ as failover

Multi-Region:

- multi-region read replicas (reads from different regions but writes go to main DB)
- disaster recovery in case of region issue
- **local performance** for global reads
- replication cost

ElastiCache

- ElastiCache is to get managed Redis or Memcached
- Caches are **in-memory databases** with high performance, low latency
- helps **reduce load off databases for read intensive workloads**
- AWS handles OS maintenance, patching, setup, optimizations, configuration, monitoring, failure, recovery/backups

DynamoDB

- fully-managed and highly available with replication across 3 AZs
- **NoSQL** database
- scales to massive workloads; distributed "server less" database
- millions of requests per second; trillions of rows; 100s of TBs of storage
- fast and consistent performance
- **single-digit millisecond latency**; low latency retrieval
- integrate with IAM for security, authorization and administration
- low cost and auto scaling capabilities

Data:

- key/value database

DynamoDb Accelerator (DAX)

- **fully managed in-memory cache for DynamoDB**
- 10x performance improvement; single-digit millisecond latency to microseconds latency when accessing DynamoDB tables
- secure, highly scalable, highly available
- difference with ElastiCache at the CCP exam level: **DAX is integrated with and only used for DynamoDB**, while ElastiCache can be used for other databases

DynamoDB Global Tables

- make a **DynamoDB table accessible with low latency in multiple regions**
- **Active-active replication** (read/write to an AWS Region)

Redshift

- based on PostgreSQL but it is NOT used for online transaction processing (OLTP)
- **Redshift is online analytical processing (OLAP) - analytics and data warehousing**
- load data once every hour rather than every second
- 10x better performance than other data warehouses; scales to PBs of data
- **Columnar** storage of data (instead of row-based)
- massively parallel query execution (MPP); highly available
- pay-as-you-go based on instances provisioned
- SQL interface for querying data
- Business intelligence (BI) tools such as AWS Quicksight for Tableau integrate with Redshift

Amazon Elastic MapReduce (EMR)

- **EMR helps create Hadoop clusters (Big Data)** to analyze and process vast amounts of data
- clusters can be made from *hundreds of EC2 instances*
- supports Apache Spark, HBase, Presto, Flink and others…
- EMR handles provisioning and configuration
- Auto-scaling and integrated with Spot instances
- Use cases: data processing, machine learning, web indexing, big data

Amazon Athena

- **Serverless query service to perform analytics again S3 objects**
- uses SQL language to query files
- Supports CSV, JSON, ORC, Avro, and Parquet (built on Presto)
- Pricing: $5.00 / TB of data scanned
- use compressed or columnar data for cost-savings (less scanning)
- use cases: business intelligence, analytics, reporting, analyze and query VPC flow logs, ELB logs, CloudTrails
- **exam tip: analyze data in S3 using serverless Sql → Athena**

Amazon QuickSight

- **serverless machine learning-powered business intelligence service to create interactive dashboards**
- fast, automatically scalable, embeddable, with per-session pricing
- use cases: business analytics, building visualizations, perform ad-hoc analysis, business insights using data
- **integrated with RDS, Aurora, Athen, Redshift, S3**

DocumentDB

- DocumentDB is "AWS implementation" of **MongoDB** (NoSQL database)
- **used to store, query, and index JSON data**
- similar deployment concepts as Aurora
- fully managed, highly available, with replication across 3 AZs
- Like Aurora, storage automatically grows in increments of 10 GB up to 64 TB
- Automatically scales to workloads with millions of requests per second

Amazon Neptune

- **fully managed graph database**
- popular **graph dataset** would be a **social network**
  - users have friends
  - posts have comments
  - comments have likes from users
  - users share and like posts
- highly available across 3 AZs; up to 15 read replicas
- build and run applications working with highly connected datasets; optimized for these complex and hard queries
- can store up to billions of relations and query the graph with millisecond latency
- great for knowledge graphs (Wikipedia), fraud detection, recommendation engines, social networking

Amazon QLDB

- Quantum Ledge Database (QLDB)
- **ledger is a book recording financial transactions**
- fully managed, serverless, highly available, replication across 3 AZs
- used to **review history of all changes made to your application data over time**
- **immutable system:** no entry can be removed or modified; cryptographically verifiable
- 2-3x better performance than common ledger blockchain frameworks; SQL to manipulate data
- **difference with Amazon Managed Blockchain: no decentralization concept**; in accordance with financial regulation rules
- centralized ledger

Amazon Managed Blockchain

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

Amazon Database Migration Service (DMS)

- Database Migration Service **quickly and securely migrates databases to AWS**
- resilient and self-healing
- **source database remains available during migration**
- *Homogenous* migrations (e.g. Oracle to Oracle)
- *Heterogeneous* migrations (e.g. SQL Server to Aurora)

AWS Glue

- managed **extract, transform, and load (ETL) service**
- ETL is useful to prepare and transform data for analytics
- Glue is **serverless**
- Use script(s) to extract, transform, and load data
- **Glue Data Catalog**: catalog of datasets (reference of column names, field types, etc.)

Infrastructure and Deployment at Scale

- CloudFormation
  - infrastructure as code
- Cloud Development Kit (CDK)
- Elastic Beanstalk
  - Platform as a Service (PaaS)
- CodeDeploy
- CodeCommit (essentially an AWS version of GitHub)
- CodeBuild
- CodePipeline
- CodeArtifact
- CodeStar
- Cloud9
- Systems Manager (SSM)
- OpsWorks

Global Infrastructure

- Route 53
  - Managed domain name system (DNS)

- CloudFront
  - Global edge network
  - better for static site that needs to globally available
- S3 Cross Region
- Global Accelerator
  - Good fit for non-HTTP use cases, such as gaming (UDP), IoT (MQTT), or Voice over IP, as well as for HTTP use cases that specifically require static IP addresses or deterministic, fast regional failover
  - Provides static IP addresses that provide a fixed entry point to your applications and eliminate the complexity of managing specific IP addresses for different AWS Regions and Availability Zones
- AWS Outposts
- Wavelength
- Local Zones

Cloud Integration

- SNS
  - publish/subscribe system
- SQS
  - decoupling applications
- Kinesis (real-time big data streaming)
- Managed Apache ActiveMQ (MQ)

Cloud Monitoring

- CloudWatch
  - Metrics
    - metric is a **variable to monitor** (e.g. CPU utilization)
  - Alarms
  - Logs
    - real-time log monitoring
  - Events
  - EventBridge
    - evolution of CloudWatch Events (I.e. similar to events but with new capabilities)
- CloudTrail
  - *Enabled by default*
  - *Managed* Events (logged by default)
  - *Data* Events (not logged by default)

    - *Insight* Events
        - can detect unusual activity in account such as:
            - bursts of IAM actions
            - hitting service limits
    - Events Retention
        - events stores for 90 days by default
        - log to S3 if you want to retain for a longer period
- X-Ray
- CodeGuru
- Public Health Dashboard
- Personal Health Dashboard

Virtual Private Cloud (VPC)

- Subnet
- Internet Gateway
- NAT Gateway
- Network Access Control List (ACL)
    - stateless (does *not* remember previous traffic)
- Security Group
    - at the instance level
    - stateful (return traffic allowed)
- VPC Flow Log
- VPC Peering
- VPC Endpoints
    - **Gateway** for S3 and DynamoDB
    - **Interface** for other AWS services
- Direct Connect (DX)
    - physical connection between on-premises and AWS
    - private network
- Site-to-Site VPN
    - public network, but encrypted traffic
    - on-premises: must use a Customer Gateway (CGW)
    - AWS: use a Virtual Private Gateway (VPG)
- Transit Gateway
    - **transitive** peering between thousands of VPC and on-premises via hub-and-spoke connection

Security and Compliance

- Shield (ford DDoS)
  - Layer 3/4 (TCP) attacks
- Shield Advanced (premium paid service)
  - NO support plan offers this by default
- Web Application Firewall (WAF)
  - Layer 7 (HTTP/HTTPS) attacks
  - Can be deployed to:
    - CloudFront
    - App Load Balancer
    - API Gateway
    - AppSync
- Penetration Testing
- Key Management Service (KMS)
- CloudHSM
  - hardware security model
- Certificate Manager (ACM)
- Secrets Manager
  - secretes are encrypted using KMS
  - integrated with Amazon RDS
  - can rotate secrets every x days
- Artifacts
- GuardDuty
- Inspector
  - automated security assessments for *EC2 instances*
- AWS Config
  - per-region service for auditing and recording compliance of your AWS services
- Amazon Macie
- Security Hub
- Amazon Detective
  - identifies root causes of security issues
  - Data sources:
    - AWS CloudTrail logs
    - VPC Flow Logs
    - GuardDuty findings
- users can report abuse to AWS
  - spam, port-scanning, copyrighted material, etc.
- **Root user** only:
  - change account settings (email, account name
  - close AWS account

    - change/cancel AWS Support plan
    - register as seller in Reserved Instance Marketplace
    - sign up for GovCloud
    - other actions…

Machine Learning

- Rekognition
    - region-based service
- Transcribe
    - speech-to-text
- Polly
    - text-to-speech
- Translate
- Lex + Connect
    - same tech used to power Alexa
    - Connect : virtual call center
    - Lex: conversational bots
- Comprehend
    - Natural Language Processing (NLP)
- SageMaker
    - helps to build/train ML models
- Forecast
- Kendra
    - fully managed document search service
    - incremental learning
- Personalize
    - personalized recommendations

Account Management, Billing, and Support

Organizations

- Global service
- Can manage **multiple AWS accounts**
- Main account is **master account**
- Able to automate AWS account creation
- Consolidated billing
- Polling of Reserved EC2 instances
- Service Control Policy (SCP)
    - restrict account privileges

- Consolidated Billing
  - combined usage
  - One bill
- AWS Control Tower
  - set up and govern a **secure and compliant multi-account AWS environment**
  - runs on top of AWS organizations
- Pricing Plans
  - Compute Pricing
- Savings Plan
  - EC2 Savings Plan
    - up to 72% discount compared to On-Demand
    - commit to usage of individual instance families in a region (e.g. C5 or M5)
  - Compute Savings Plan
    - regardless of family, region, size, OS, tenancy, compute options
    - up to 66% discount compared to On-Demand
- Compute Optimizer
- Billing & Costing Tools
  - Estimate cost in cloud:
    - Total Cost of Ownership (TCO) Calculator
      - Compare on-premise vs AWS cost for you application
    - Simple Monthly Calculator/pricing calculator
      - **replaced by AWS Pricing Calculator**
  - Tracking cloud costs
    - Billing Dashboard
    - Cost Allocation tags
      - Track AWS costs on a deeper level
      - AWS-generated tags (prefixed with **aws:**)
      - User-defined tags (prefixed with **user:**)
      - Can create a *Resource Group*
    - Cost and Usage reports
      - most comprehensive set of AWS cast and usage data available
    - Cost Explorer
      - visualize spending/costs
  - Monitoring against cost plans
    - Billing data metric is stored in CloudWatch us-east-I
    - Billing data are for overall *worldwide* AWS costs
    - Billing Alarms
    - Budgets
      - Usage Budget

        - Cost Budget
        - Reservation Budget
    - For Reserved Instances:
        - Track utilization
        - Supports EC2, ElastiCache, RDS, Redshift
    - Up to *five SNS notifications* per budget
- Trusted Advisor
    - High level AWS account assessment
    - Analyzes AWS accounts and provides recommendations based on (exam may ask to list 2 of 5)
        - Cost optimization
        - Performance
        - Security
        - Fault tolerance
        - Service limits
    - 7 Core Checks for Basic and Developer Support plans
        - S3 bucket permissions
        - security groups
        - IAM Use
        - MFS on Root account
        - EBS Public Snapshots
        - RDS Public Snapshots
        - Service Limits
    - Full Checks for Business and Enterprise Support Plans
        - Programmatic access using **AWS Support API**
- Support Plans
    1. **Basic** Support is free
        - 7 Core checks
    2. **Developer**
        - All from basic plan +
        - Business hours email access to Cloud Support
        - Unlimited cases and 1 primary contact
    3. **Business**
        - Intended for production workloads
        - Full checks from Trusted Advisor
        - *24/7 access to phone, email, and chat access to Cloud Support*
    4. **Enterprise**
        - Intended for mission-critical workloads
        - Access to dedicated **Technical Account Manager (TAM)**

        - Concierge Support Team
        - Infrastructure Event Management, Well-Architected & Operations Reviews
        - **Business-critical system down response in under 15 minutes**
- Account Best Practices
    - Operate multiple accounts using **Organizations**
    - Use **SCPs** to restrict account power
    - easily setup multiple accounts with AWS Control Tower
    - Use **Tags** and Cost Allocation Tags for way management and billing
    - **IAM** Guidelines: MFA, least-privilege, strong password policy, password rotation
    - **Config** to record all resource configurations and compliance over time
    - **CloudFormation** to deploy stacks across accounts and regions
    - **Trusted Advisor** to gain insights, Support Plans adapted to your needs
    - Send Service Logs and Access Logs to **S3** and **CloudWatch** logs
    - CloudTrail to log API calls made within account
    - If account is *compromised*: change root password, delete/rotate all passwords/keys, contact AWS support

Advanced Security

- Security Token Service (STS)
    - Enables you to create **temporary, limited privilege credentials** to access AWS resources
    - Use Cases:
        - Identity federation
        - IAM roles for cross/same account access
        - IAM roles for Amazon EC2 (create temp credential for instances to access AWS resources)
- Amazon Cognito
    - Identity for web and mobile application users
    - This is because we do *NOT* want to create IAM users for your application users
- Microsoft Active Directory (AD)
    - database of **objects** (user accounts, computers, printers, etc.)
    - found on any Windows Server with AD Domain Services
- AWS Directory Services
    - Three flavors:
        1. AWS Managed Microsoft AD
        2. AD Connector
        3. Simple AD
            - *cannot* be joined with on-premise AD

- AWS Single Sign-On (SSO)
  - Centrally manage single sign-on to access multiple accounts and third-party business applications
  - Integrated with AWS Organizations
  - Supports SAML 2.0 markup
  - Integration with on-premise Active Directory

Other AWS Services

- Amazon WorkSpaces
  - Managed desktop as a service (DaaS) solution to easily provision **Windows of Linux desktops**
  - Regionally based
  - best practice: deploy in region closest to users for best performance (lowest latency)
- AppStream 2.0
  - Desktop app streaming service
  - **Stream application to a computer via web browser**
- Amazon Sumerian
  - create VR/AR/3D applications
- Internet of Things (IoT) Core
  - easily connect IoT devices to the AWS cloud
  - serverless
  - integrations with Lambda, S3, SageMaker, etc.
- Elastic Transcoder
  - used to convert media files stored in S3 into media files in the formats required by consumer playback devices
    - devices such as phone, tablets, etc.
- Device Farm
  - test web and mobile applications against browsers, physical mobile devices and tablets
- AWS Backup
  - centrally manage and automate backups across AWS services
  - Supports Point-in-time Recovery (PITR)
- Disaster Recovery Strategies
  1. Backup and Restore (simple and cheapest; stored in S3)
  2. Pilot Light (core app functions; ready to scale but minimal setup; via EC2)
  3. Warm StandBy (full version of app via EC2 but at minimum size)
  4. Multi-Site/Hot-Site (most expensive; full app ready to use)

- CloudEndure Disaster Recovery (purchased by AWS)
  - recover physical, virtual, and cloud-based servers into AWS
  - Continuous block-level replication for your servers
- AWS DataSync
  - Move large amounts of data from on-premises to AWS
  - Replication tasks are **incremental** after the first full load
- AWS Fault Injection Simulator (FIS)
  - Chaos Engineering
  - Can run fault injection experiments on AWS workloads
  - Helps uncover hidden bugs and reduce bottlenecks
  - Can use pre-built templates

AWS Architecting & Ecosystem

Well Architected Framework Guidelines/Principles

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

AWS Cloud Best Practices - Design Principles

- Scalability: vertical and horizontal
- Disposable Resources: servers should be disposable and easily configured
- Automation: Serverless, IaaS, Auto Scaling, etc.
- Loose coupling: if one piece fails the other piece(s) will not be directly affected
- Services, Not Servers: don't just use EC2

Five Pillars of Well Architected Framework

1. Operational Excellence
   - Perform operations as code (IaaS)
   - Annotate documentation
   - Make frequent, small, reversible changes
   - Refine operation procedures frequently
   - Anticipate failure
   - Learn from failures

    - **CloudFormation** is key for this pillar
2. Security
    - Implement strong identity foundation
    - Enable traceability
    - Apply security at all layers
    - Automate security best practices
    - Protect data in-transit and at rest
    - Keep people away from data
    - Prepare for security events
    - **IAM** is key for this pillar
3. Reliability
    - Test recovery procedures
    - Automatically recover from failure
    - Scale horizontally to increase aggregate system availability
    - Stop guessing capacity (use auto-scaling)
    - Manage change in automation
    - Auto Scaling and IAM are important
    - **Service Quotas and Trusted Advisor**
4. Performance Efficiency
    - Democratize advanced technologies
    - Go global in minutes
    - Use server-less architectures
    - Experiment more often
    - Mechanical sympathy (be aware of all AWS services)
5. Cost Optimization
    - Adopt a consumption mode (pay for what you use)
    - Measure overall efficiency
    - Stop spending money on data center operations (migrate to cloud)
    - Analyze and attribute expenditure
    - Use managed and application level services to reduce cost of ownership
- **Five Pillars are not something to balance – they are synergetic.**
- AWS Well-Architected Tool
    - Review your architectures against five pillars and obtain advice
- Right Sizing
    - cloud is elastic
    - match instance types and sizes to meet power/capacity requirements at lowest cost
    - Scaling up is easy so always start small
    - Right Size before cloud migration

        - continuously re-size after onboarding process (because requirements change over time)
- AWS Ecosystem
    - Blogs
    - Forums (community)
    - White papers and Guides
    - Quick Starts (e.g. Wordpress on AWS)
    - AWS Solutions
    - AWS Marketplace (e.g. buy and sell AMIs)
    - AWS Training
        - Digital and In-Class
        - Private training for your organization
        - AWS Academy
    - AWS Professional Service and Partner Network (APN)
        - APN Technology Partners (providing hardware, connectivity, software)
        - APN Consulting Partners (professional services firm to help build on AWS)
        - APN Training Partners (help people learn AWS)
        - AWS Competency Program
        - AWS Navigate Program (help partners become better partners)
- *AWS Knowledge Center* (FAQs and common requests)

Distractors

Some AWS services are included as answers and are **distractors**
If you come across a seemingly random or new concept/service - it may be a distractor.

Trust your training. The exam questions should be simple and straightforward. If an answer seems complex or sophisticated, it is likely incorrect.