AWS cloud practitioner

---

0. Resources for learning

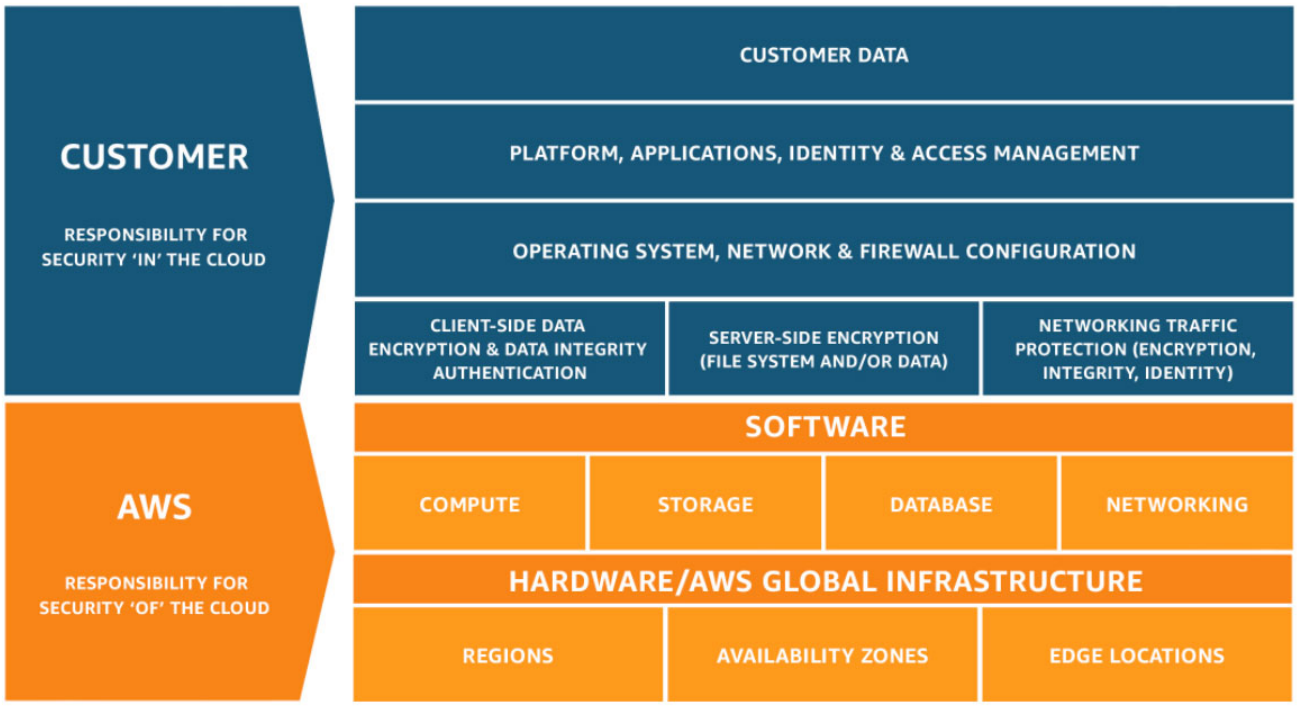
- A Cloud Guru: course and practice exams
- Udemy practice exams
- Braincert practice exams

1. Introduction to AWS Cloud

- Traditional, on-premises IT: all the hardware is owned, operated, maintained, and housed by the company
- Cloud: the resources are hosted by a cloud services platform, which owns and maintains the network-connected hardware, while customers provision and use what they need via a web application.

> **Cloud computing** is the *on-demand delivery* of compute power, database, storage, applications, and other IT resources via the Internet with *pay-as-you-go pricing*

Shared responsibility model: AWS is reponsible for security **of** the cloud, while customers and APN partners are responsible for security **in** the cloud. Responsibilities depend on the services used. IaaS are completely under customer control, and encryption is always a task of the customer.



Shared control: controls which apply to both the infrastructure layer and customer layers, but in completely separate contexts or perspectives: configuration management, patch management.

AWS decommissions storage devices that have reached the end of their useful life, and it uses industry-standard practices for it.

1.1. Cloud computing vs. traditional on-premises IT

- *Variable opex instead of fixed capex*: pay for what you use, no upfront payments. AWS offers lower upfront and lower variable costs
- *Benefit from large economies of scale*: because AWS stores many customers' data, the scale of it makes the pay-as-you-go prices lower.
- Flexibility: *easy scaling*: when properly configured, resources in the cloud can scale in our out depending upon demand. It is not necessary to guess infrastructure capacity needs.
- Agility: *easy to obtain new IT resources*: in the cloud, new hardware resources can be obtained instantly with a click.
- Global reach: *easy deployment*: customers can deploy their application in multiple regions around the world with a few clicks.

1.2. Global infrastructure

- *AWS regions*: each region is isolated from the others, achieving fault tolerance and stability. Resources in one region are not replicated in other regions. A particular region might be chosen because of:
    - Data sovereignty issues, where data should be stored,
    - Latency to end users
    - AWS services, some regions have more services than others (us-east-1 vs. Singapore)

    - Cost
- *AZ, availability zones*: each region has multiple locations isolated from each other. AZs are physically separated, but interconnected via high-bandwidth, low-latency, fully redundant fibre which enable synchronous communication between different AZs. **For high availability, enable multi-AZ.**
- *Edge locations* are regional edge cache servers, used by *CloudFront* to securely deliver resources to customers globally with low-latency, high transfer speeds.

Other concepts

- *Cloudfront* is a content delivery network (CDN), a system of distributed servers that deliver webpages and other web content to a user based on the geographical location of the user, the origin of the webpage, and a content delivery server
- *Global accelerator*: improves availability and performance of apps with local or global users. It optimizes the path from your users to your apps, monitors the health of your app and redirects traffic to healthy endpoints.
- *Transfer acceleration*: if an object is uploaded to a bucket, it's first transferred to an edge location and then from there, through Amazon's internal network (not out in the internet), it reaches the bucket.

1.3. Types of cloud computing

- Infrastructure as a service (using EC2)
- Infrastructure as code: **Cloudformation**
    - Automates the creation and setup of infrastructure, you create a template with the resources you want, and it provisions and configures them
- Platform as a service: no need to manage the underlying infrastructure (Elastic beanstalk for example)
- Software as a service (gmail)
- Desktop as a service: can provision win/linux desktops in a few minutes (AWS workspaces)

---

2. AWS core technologies

2.1. Compute services

Used to develop, deploy, run and scale applications and workloads in AWS.

2.1.1. Amazon EC2

It's an infrastructure as a service providing secure, resizable compute capacity. They are not automatically replicated across AZs. For Linux instances, you are billed by the second with a one minute minimum. For other instances, you are billed hourly with one hour minimum.

- General purpose: balance of compute, memory and networking resources
- Compute optimized: for high performance processors
- Memory optimized: fast performance to process large data sets in memory
- Accelerated computing: hardware accelerators
- Storage optimized: high, sequential read and write access to very large data sets on local storage

> **Amazon EC2 auto scaling**

It doesn't have any cost per se, but it's coupled to EC2.

- Triggers can automatically increase/decrease the number of instances in their deployment.
- It monitors the health of running instances. If an instance fails a health check, it gets automatically terminated and replaced with a new one.
- It supports dynamic (responds to changing demand) and predictive (based on predictive demand) scaling

> **ELB, Elastic load balancing**

It distributes incoming application *traffic* across multiple EC2 instances to achieve better fault tolerance and availability. Customers can configure health checks, so that the load balancer sends requests only to the healthy ones, and offload the work of encryption and decryption to the load balancer. Three types of load balancers:

- Application: designed for web apps with HTTP and HTTPS traffic
- Network: operates at the network layer, UDP/TCP requests
- Classic: the previous generation of load balancers, used for classic EC2 instance types

> Best practices

- Scale horizontally, not vertically. Add more compute resource to your app, instead of adding more power to your compute resources
- Decouple compute from session/state, to help with scaling and availability
- Right-size your infrastructure to your workload to get the best balance between cost and performance
- Higher availability by configuring the load balancer to serve traffic to many AZs. Load balancers cannot act multi-region

2.1.2. AWS Lambda

- Serverless compute service, focus on the code and not on the servers, OS, infrastructure, etc.
- Triggered by events
- Pricing is based on number of requests of the function, and their duration
- It scales automatically without the user's intervention
- Can be used for websites

2.1.3. Pricing models

- *On demand*: pay a fixed hourly/second rate, no commitment, ideal for short-term apps
- *Reserved instances (RIs)*: for workloads with no interruptions, they require a 1-3 year commitment
    - Convertible RIs: modify reservations across families, sizes, OS and tenancy, except region
- *Spot instances*:
    - For workloads that are stateless, fault-tolerant, loosely coupled, flexible and with an end-time
    - They use an unused AWS EC2 instance that's available for less than the on-demand price, they run whatever capacity is available and the maximum price per hour for your request exceeds the spot price
    - Containerized apps are a good target for spot, as well as big data analysis, CI/CD, web services, etc.

2.2. Storage services

2.2.1. S3

- Object (key-value) based storage, unlimited storage but objects can be up to 5TB
- Instant file creation, but changing files takes longer
- It scales automatically without the user's intervention
- It offers volume discounts based on usage, with different tiers
- Good option for static websites

Types of S3 storage

- *Standard*: frequently accessed data, with low latency and high throughput. no retrieval fee per GB. Automatic replication in AZs
- *Standard-infrequent access (IA)*: lower per GB storage price and there's a per GB retrieval fee. Ideal for long-term storage, backups, and data store for disaster recovery files
- *One-zone - IA*: data is only stored in one AZ, it costs less than S3 standard-IA. Ideal for secondary backup copies of on-premises data
- *Glacier*: secure and durable storage for data archiving and backup. *S3 lifecycle policy*: migrate data between different types of S3 (normal -> glacier)
- *Intelligent-tiering*: data with unknown changing access patterns. Data is stored in 2 tiers: one for frequent access, another low-cost tier optimized for infrequent access. S3 monitors access to objects and moves the objects between tiers, according to their use. S3 charges a per-object monitoring fee. For smaller objects, the monitoring fee is smaller.

Other concepts

- *Kinesis*: securely stream from millions of devices to AWS for analytics, ML, etc
- *Macie*: uses NLP to discover, classify and protect sensitive data in S3
- *Athena*: serverless query service to analyse and query data in S3 using SQL to generate reports

2.2.2. EBS

Elastic block store: like a hard drive (filesystem) for EC2 instances. Customers can create partitions on it, format it, and boot their OS off it. They are attached to EC2 instances, only to only one instance at a time. When the instance is terminated, they detach. It is a regional service *automatically replicated within the AZ*, and a good option for high read/write databases. EBS data isn't wiped when it's un-mounted.

2.2.3. EFS

- Elastic file system: file storage, network attach storage
- similar to EBS but storage capacity is elastic, it adjusts size as you add or remove files from it
- Good for apps with a higher workload
- EFS can be mounted to different AWS services and accessed from all VMs
- offers a filesystem that can be mounted concurrently from *multiple* EC2 instances
- provides shared file system

Other concepts:

- *File/storage gateway* file system mount on S3, the OS is accessible via the computer's file system. On-premises access to virtually unlimited cloud storage
- AWS Snowball: data tansfer in petabytes up to 80TB, from on-premises to AWS. IT takes care of connectivity problems as well
- AWS Snowmobile: data transfer in exabytes

2.3. Database services

- *Relational database service (RDS)*: Amazon Aurora (5x times faster than SQL and scales automatically), PostgreSQL, MySQL, MariaDB, etc (these don't scale automatically). They use the EBS storage type. To offload database read activity, use read replicas. The customer is responsible for scheduling and performing backups
- *DynamoDB*: noSQL serverless database
- *Redshift*: data warehousing, to *analyze* data using SQL and traditional business intelligence tools
- *ElastiCache*: caching service to help deploy, operate and scale an in-memory caching system, instead of relying solely on slower disk-based databases. It can be a good option for intensive I/O RDS databases. Important to use between hosts and databases. Good option to improve performance of web apps.

2.4. Networking

Virtual private cloud (VPC): virtual network in the cloud

- *ACL*: access control list (subnet level), that allows/denies network traffic (IP addresses) to the subnet
- *Security groups* (instance level): virtual firewalls that restrict access to EC2 and can only set Allow rule

---

- *Virtual private gateway*: connect on-premises data center with AWS
- *Transit gateway*: network transit hub that simplifies the interconnection of VPCs across thoursands of AWS accounts and their on-premises networks
- *Internet gateway*: communication channel between a VPC and the internet

---

Route 53 configures DNS, connects user requests to infrastructure running in AWS, and outside of AWS. DNS health checks to route traffic to healthy endpoints, latency based routing (for multi-region apps, to direct users to region with lowest latency), geo DNS, geoproximity (depends un user's location), to make low-latency, fault-tolerant architectures.

2.5. Security

2.5.1. IAM

Identity and access management. IAM is global, not attached to a region.

IAM can only be used control who has access to launch, start, stop, terminate and other controls in AWS resources. Once the service is launched, the IAM role cannot control the access to the instance.

- Users: end users, usually one person per user
- Groups: a collection of users, each user in the group inherits the permissions of the group. To set the permissions in a group, apply a policy to that group
- Role: to control access of AWS resources

- Policy: json/yaml document providing permissions for users, groups and roles
- Principal: person/app using the root user, IAM user or an IAM role
- Entity: IAM resource objects that AWS uses for authentication
- Identity: IAM users and roles
- Resource: user, group, role, policy stored in IAM

> Best practices

- Encrypt data at rest and in transit
- Enforce *principle of least privilege* in IAM, so that no individual or service has more privileges than they absolutely require

2.5.2. Auditing and compliance

- *Artifact*: self-service *audit artifact retrieval* portal that provides customers with on-demand access to AWS' compliance documentation and AWS agreement
- *Inspector*: security assessment service to check *vulnerabilities*, help improve the security and compliance of applications deployed on AWS
- *Trusted advisor*: online tool that provides you real time *guidance* to help follow AWS best practices: cost optimization, performance, security and fault tolerance
- *Config*: discover existing and deleted resources, determine overall compliance against rules. It enables compliance auditing, security analysis, resource change tracking, troubleshooting.
- *Personal health dashboard*: personalized view of AWS service health, detailed troubleshooting guidance to address AWS events impacting your resources

2.5.3. Other security

- *WAF*: web app firewall (application layer) that protects web apps from common web exploits.
- *Shield*: DDoS protection service
- *Guard duty*: monitors for malicious or unauthorized behavior
- *KMS, key management service*: storing of cryptographic keys
- *Secrets manager*: encryption of secrets with KMS generated keys
- *Penetration testing*: customers can do it without AWS approval for EC2, RDS, CloudFront, Aurora, API Gateways, Lambda, Lightsail and Elastic Beanstalk.

2.6. Monitoring

- **CloudWatch**: monitors AWS resources + billing and creates logs, can create alerts for when some threshold are met
- **CloudTrail** auditing tool that monitors activity in AWS, what *actions* were taken, by whom, when and where
- *Cost explorer*: see the distribution of past costs
- *Simple monthly calculator*: estimate future costs
- *Tagging* strategy: to identify resources belonging to a specific project, and to track AWS spending across multiple resources
- *SQS*: simple queue service to send, store and receive messages between software components, at any volume, without losing messages. Allows decoupling services and apps.

- *SNS*: simple notification service, used for mass delivery of messages predominantly to mobile users. It can be used for private cloud as well. It can send data to Lambda, HTTP/S, Email, SQS and SMS.

2.7. Management interfaces

AWS can be accessed in these 3 ways

- AWS management console: graphical interface, access and permission through IAM
- AWS command line interface (CLI), needs *access keys* to get access to AWS
- Software development kits (SDK) are packages that enable access to AWS in many programming languages

2.8. Support

All plans have access to 24x7 customer service, but developer only email

- Basic
- Developer
- Business: from this tier, AWS Support API, 24x7 phone call support
- Enterprise
    - appropriate for mission critical workloads
    - TAM, a technical account manager from AWS constantly monitoring cloudwatch
    - Well-architected review delivered by AWS solutions architect
    - architectural and scaling guidance: infrastructure event management

---

3. Migrating strategies

- **Rehost**
    - Recreation of the on-premises network in AWS automatically with migration tools like VM Import/Export, or manually.
    - Lifting and shifting an application to the cloud. Servers are migrated to EC2 instances. Network configurations are recreated using VPCs, subnets, security groups, internet and VPN gateways, etc.
- **Replatform**
    - The core architecture of the application remains, only targeted AWS cloud optimizations. Migrating databases to RDS.
- **Refractor**
    - Re-imagine how the application is architected using cloud-native features. Changing a database structure from a standard RDS database to Amazon Aurora.
    - Typically done by customers that need to add features, scale or change performance that would be difficult to achieve in the application's existing environment.
- **Retire**
    - Shutting off non-useful applications
- **Retain/revisit**
    - Keep certain applications on-premises
- **Repurchases**
    - Move workflows to software as a service (SaaS)

The *total cost ownership calculator* calculates the cost of running your app in AWS vs. on-premises.

---

4. Advanced concepts

- AWS consulting partner: help customers design, architect, build, migrate, and manage their workloads and applications on AWS
- AWS technology partner: provide software solutions that are either hosted on, or integrated with, the AWS platform. APN Technology Partners include Independent Software Vendors (ISVs), SaaS, PaaS, Developer Tools, Management and Security Vendors

Other AWS services

- Lex: builds conversational chatbox
- Transcribe: converts speech into text
- Polly: converts text to life-like voice
- Rekognition: converts image into text
- Sumerian: 3D design and engineering