**Global Understanding**

What is AWS?

Amazon Web Services offers a broad set of global cloud-based products including compute, storage, databases, analytics, networking, mobile, developer tools, management tools, IoT, security, and enterprise applications: on-demand, available in seconds, with pay-as-you-go pricing.

-Cloud Computing:

Cloud computing is an on-demand delivery of IT resources over the internet with primarily pay-as-you-go pricing. With cloud computing, companies do not have to manage and maintain their own hardware and data centers. Instead, companies like AWS own and maintain data centers and provide virtual data center technologies and services to companies and users over the internet.

AWS provides cloud computing services. The IT resources mentioned in the cloud computing definition are AWS services. For the course's corporate directory application, you will use AWS services to architect a scalable, highly available, and cost-effective infrastructure to host the corporate directory application. That way, you can get the app out into the world quickly, without managing heavy-duty physical hardware.

* Pay as you go: Instead of investing in data centers and hardware before you know how you are going to use them, you pay only when you use computing resources, and pay only for how much you use.
* Benefit from massive economies of scale: By using cloud computing, you can achieve a lower cost than you can get on your own. Because usage from hundreds of thousands of customers is aggregated in the cloud, AWS can achieve higher economies of scale, which translates into lower pay as-you-go prices.
* Stop guessing capacity:Eliminate guessing on your infrastructure capacity needs. When you make a capacity decision prior to deploying an application, you often end up either sitting on expensive idle resources or dealing with limited capacity. With cloud computing, these problems go away. You can access as much or as little capacity as you need, and scale up and down as required with only a few minutes notice.
* Increase speed and agility: IT resources are only a click away, which means that you reduce the time to make resources available to your developers from weeks to minutes. This results in a dramatic increase in agility for the organization, since the cost and time it takes to experiment and develop is significantly lower.
* Realize cost savings: Companies can focus on projects that differentiate their business instead of maintaining data centers. Cloud computing lets you focus on your customers, rather than on the heavy lifting of racking, stacking, and powering physical infrastructure. This is often referred to as undifferentiated heavy lifting.
* Go global in minutes: Applications can be deployed in multiple Regions around the world with a few clicks. This means that you can provide lower latency and a better experience for your customers at a minimal cost.

Types of cloud computing models:

* Infrastructure as a Service (IaaS): Contains the basic building blocks for cloud IT and typically provides access to networking features, computers (virtual or on dedicated hardware), and data storage space. IaaS provides you with the highest level of flexibility and management control over your IT resources and is most similar to existing IT resources that many IT departments and developers are familiar with today.
* Platform as a Service (PaaS): Removes the need for your organization to manage the underlying infrastructure (usually hardware and operating systems) and allows you to focus on the deployment and management of your applications. This helps you be more efficient as you don’t need to worry about resource procurement, capacity planning, software maintenance, patching, or any of the other undifferentiated heavy lifting involved in running your application.
* Software as a Service (SaaS): Provides you with a completed product that is run and managed by the service provider. In most cases, people referring to Software as a Service are referring to end-user applications. With a SaaS offering you do not have to think about how the service is maintained or how the underlying infrastructure is managed; you only need to think about how you will use that particular piece of software. A common example of a SaaS application is web-based email which you can use to send and receive email without having to manage feature additions to the email product or maintain the servers and operating systems that the email program is running on.

Cloud computing deployment models:

* Cloud: A cloud-based application is fully deployed in the cloud and all parts of the application run in the cloud. Applications in the cloud have either been created in the cloud or have been migrated from an existing infrastructure to take advantage of the benefits of cloud computing. Cloud-based applications can be built on low-level infrastructure pieces or can use higher level services that provide abstraction from the management, architecting, and scaling requirements of core infrastructure.
* Hybrid: A hybrid deployment is a way to connect infrastructure and applications between cloud-based resources and existing resources that are not located in the cloud. The most common method of hybrid deployment is between the cloud and existing on-premises infrastructure to extend, and grow, an organization's infrastructure into the cloud while connecting cloud resources to the internal system. For more information on how AWS can help you with your hybrid deployment.
* On-premises:The deployment of resources on-premises, using virtualization and resource management tools, is sometimes called the “private cloud.” On-premises deployment doesn’t provide many of the benefits of cloud computing but is sometimes sought for its ability to provide dedicated resources. In most cases this deployment model is the same as legacy IT infrastructure while using application management and virtualization technologies to try and increase resource utilization.

**AWS-GlobalInfraestructure**

Infrastructure, like data centers and networking connectivity, still exists as the foundation of every cloud application. In AWS, this physical infrastructure makes up the AWS Global Infrastructure, in the form of Regions and Availability Zones.

AWS has Regions in Asia Pacific, Canada, Europe, the Middle East, and South America, and we continue to expand to meet our customers' needs. Each AWS Region is associated with a geographical name and a Region code.

Here are examples of region codes:

-us-east-1: The first Region created in the eastern US area. The geographical name for this Region is N. Virginia.

-ap-northeast-1: The first Region created in the northeast Asia Pacific area. The geographical name for this Region is Tokyo.

Choose the Right AWS Region

When you decide which AWS Region to host your applications and workloads, consider four main aspects – latency, price, service availability, and compliance.

* Latency: If your application is sensitive to latency (the delay between a request for data and the response), choose a Region that is close to your user base. This helps prevent long wait times for your customers. Synchronous applications such as gaming, telephony, WebSockets, and Internet of Things (IoT) are significantly affected by high latency. Asynchronous workloads, such as ecommerce applications, can also suffer from user connectivity delays.
* Pricing: Due to the local economy and the physical nature of operating data centers, prices vary from one Region to another. Internet connectivity imported equipment costs, customs, real estate, and other factors impact a Region's pricing. Instead of charging a flat rate worldwide, AWS charges based on the financial factors specific to each Region.
* Service Availability: Some services might not be available in some Regions. The AWS documentation provides a table that shows the services available in each Region.
* Data Compliance(\*): Enterprise companies often must comply with regulations that require customer data to be stored in a specific geographic territory. If applicable, choose a Region that meets your compliance requirements.

Inside every Region is a cluster of Availability Zones (AZs).

An AZ consists of one or more data centers with redundant power, networking, and connectivity. These data centers operate in discrete facilities in undisclosed locations. They are connected using redundant high-speed and low-latency links. AZs also have a code name.

Since they are located inside Regions, they can be addressed by appending a letter to the end of the Region code name.

For example:

-us-east-1a: An AZ in us-east-1 (N. Virginia Region)

-sa-east-1b: An AZ in sa-east-1 (São Paulo Region)

Scope AWS services

Depending on the AWS service you use, your resources are either deployed at the AZ, Region, or Global level. Each service is different, so you must understand how the scope of a service might affect your application architecture.

Maintain resiliency

To keep your application available, you must maintain high availability and resiliency. A well-known best practice for cloud architecture is to use Region-scoped, managed services.

AWS service endpoints

To connect programmatically to an AWS service, you use an endpoint. An endpoint is the URL of the entry point for an AWS web service. The AWS SDKs and the AWS Command Line Interface (AWS CLI) automatically use the default endpoint for each service in an AWS Region. But you can specify an alternate endpoint for your API requests.

**AWS-Interacting**

The AWS Management Console

One way to manage cloud resources is through the web-based console,

where you log in and choose the desired service. This can be the easiest way to create and manage resources

when you first begin working with the cloud. The AWS Management Console is a web application that comprises

and refers to a broad collection of service consoles for managing AWS resources.

The AWS Command Line Interface (AWS CLI)

Consider the scenario where you run tens of servers on AWS for your application’s front end.

You want to run a report to collect data from all the servers. You need to do this programmatically every day

because the server details might change. Instead of manually logging in to the AWS Management Console and then

copying and pasting information, you can schedule an AWS CLI script with an API call to pull this data for you.

AWS SDKs

API calls to AWS can also be performed by running code with programming languages.

You can do this by using AWS software development kits (SDKs). SDKs are open source and maintained by

AWS for the most popular programming languages, such as C++, Go, Java, JavaScript, .NET, Node.js, PHP, Python,

and Ruby.

**AWS-Security**

When you work with the AWS Cloud, managing security and compliance is a shared responsibility between AWS and you. To depict this shared responsibility, AWS created the shared responsibility model. The distinction of responsibility is commonly referred to as security OF the cloud versus security IN the cloud.

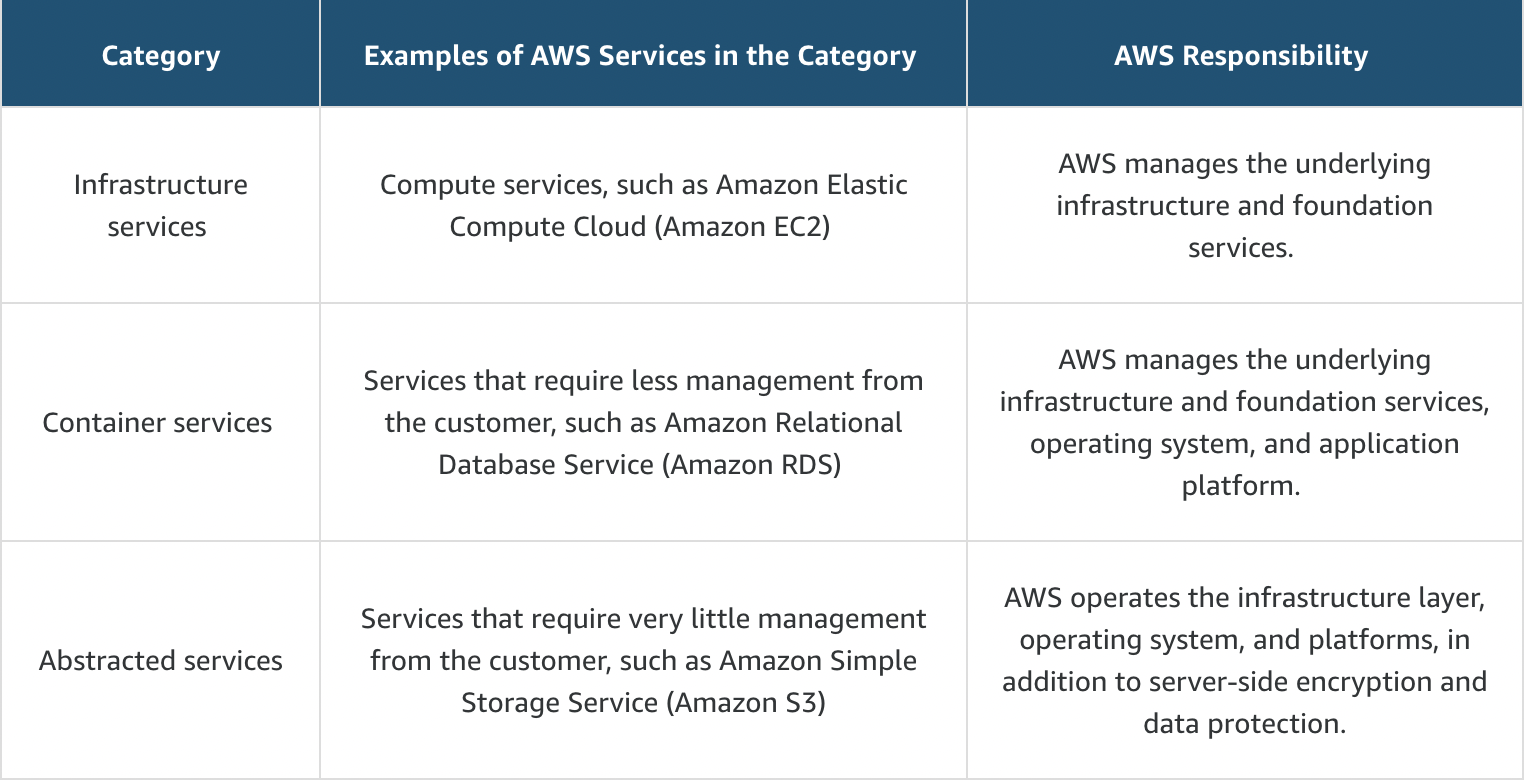
Timeline

Description automatically generated

AWS responsibility

AWS is responsible for security of the cloud. This means AWS protects and secures the infrastructure that runs the services offered in the AWS Cloud. AWS is responsible for:

* Protecting and securing AWS Regions, Availability Zones, and data centers, down to the physical security of the buildings.
* Managing the hardware, software, and networking components that run AWS services, such as the physical servers, host operating systems, virtualization layers, and AWS networking components.



Customer responsibility

Customers are responsible for security in the cloud. When using any AWS service, you’re responsible for properly configuring the service and your applications, in addition to ensuring that your data is secure.

Table

Description automatically generated

A key concept is that customers maintain complete control of their data and are responsible for managing the security related to their content. For example, you are responsible for the following:

* Choosing a Region for AWS resources in accordance with data sovereignty regulations.
* Implementing data-protection mechanisms, such as encryption and scheduled backups.
* Using access control to limit who can access to your data and AWS resources.

**ROOT USER**

Authentication

When you create your AWS account, you use the combination of an email address and a password to verify your identity. If a user types in the correct email and password, the system assumes the user is allowed to enter and grants them access. This is the process of authentication.

Authorization

Once you’re authenticated and in your AWS account, you might be curious about what actions you can take. This is where authorization comes in. Authorization is the process of giving users permission to access AWS resources and services.

Credentials

The AWS root user has two sets of credentials associated with it. One set of credentials is the email address and password used to create the account. This allows you to access the AWS Management Console. The second set of credentials is called access keys, which allow you to make programmatic requests from the AWS Command Line Interface (AWS CLI) or AWS API.

Supported MFA devices

AWS supports a variety of MFA mechanisms, such as virtual MFA devices, hardware devices, and Universal 2nd Factor (U2F) security keys. For instructions on how to set up each method, check out the Resources section.

Table

Description automatically generated

**AWS Identity and Access Management**

**IAM**

AWS Identity and Access Management (IAM) is an AWS service that helps you manage access to your AWS account and resources. It also provides a centralized view of who and what are allowed inside your AWS account (authentication), and who and what have permissions to use and work with your AWS resources (authorization). With IAM, you can share access to an AWS account and resources without sharing your set of access keys or password. You can also provide granular access to those working in your account, so people and services only have permissions to the resources they need.

**IAM features**

To help control access and manage identities in your AWS account, IAM offers many features to ensure security.

* IAM is global and not specific to any one Region. You can see and use your IAM configurations from any Region in the AWS Management Console.
* IAM is integrated with many AWS services by default.
* You can establish password policies in IAM to specify complexity requirements and mandatory rotation periods for users.
* IAM supports MFA.
* IAM supports identity federation, which allows users who already have passwords elsewhere – for example, in your corporate network or with an internet identity provider – to get temporary access to your AWS account.
* Any AWS customer can use IAM; the service is offered at no additional charge.

**IAM user**

An IAM user represents a person or service that interacts with AWS. You define the user in your AWS account. Any activity done by that user is billed to your account. Once you create a user, that user can sign in to gain access to the AWS resources inside your account.

**IAM user credentials**

An IAM user consists of a name and a set of credentials. When you create a user, you can provide them with the following types of access:

Access to the AWS Management Console

Programmatic access to the AWS Command Line Interface (AWS CLI) and AWS application programming interface (AWS API).

**IAM groups**

An IAM group is a collection of users. All users in the group inherit the permissions assigned to the group. This makes it possible to give permissions to multiple users at once. It’s a more convenient and scalable way of managing permissions for users in your AWS account.

**IAM policies**

To manage access and provide permissions to AWS services and resources, you create IAM policies and attach them to IAM users, groups, and roles. Whenever a user or role makes a request, AWS evaluates the policies associated with them.

**IAM policy**

Most policies are stored in AWS as JSON documents with several policy elements. The following example provides admin access through an IAM identity-based policy.

Table

Description automatically generated

Table

Description automatically generated{

"Version": "2012-10-17",

"Statement": [{

"Effect": "Allow",

"Action": "\*",

"Resource": "\*"

}]

}

This policy has four major JSON elements: Version, Effect, Action, and Resource.

* The Version element defines the version of the policy language. It specifies the language syntax rules that are needed by AWS to process a policy. To use all the available policy features, include "Version": "2012-10-17" before the "Statement" element in your policies.
* The Effect element specifies whether the statement will allow or deny access. In this policy, the Effect is "Allow", which means you’re providing access to a particular resource.
* The Action element describes the type of action that should be allowed or denied. In the example policy, the action is "\*". This is called a wildcard, and it is used to symbolize every action inside your AWS account.
* The Resource element specifies the object or objects that the policy statement covers. In the policy example, the resource is the wildcard "\*". This represents all resources inside your AWS console.

Good Practices:

• Lock down the AWS root user

• Follow the principle of least privilege

• Use IAM appropriately

• Use IAM roles when possible

• Consider using an identity provider

• Consider AWS Single Sign-On

**Compute as a Service**

**Servers**

The first building block you need to host an application is a server. Servers usually can handle Hypertext Transfer Protocol (HTTP) requests and send responses to clients following the client-server model, although any API-based communication also falls under this model. A client is a person or computer that sends a request. A server handling the requests is a computer, or collection of computers, connected to the internet serving websites to internet users.

**Choose the right compute option**

If you’re responsible for setting up servers on AWS to run your infrastructure, you have many compute options. You need to know which service to use for which use case. At a fundamental level, three types of compute options are available – virtual machines (VMs), container services, and serverless.

If you have prior infrastructure knowledge, a virtual machine is often be the easiest compute option to understand. This is because a virtual machine emulates a physical server and allows you to install an HTTP server to run your applications. To run virtual machines, you install a hypervisor on a host machine. The hypervisor provisions the resources to create and run your VMs.

In AWS, virtual machines are called Amazon Elastic Compute Cloud, or Amazon EC2. Behind the scenes, AWS operates and manages the host machines and the hypervisor layer. AWS also installs the virtual machine operating system, called the guest operating system.

**Amazon Elastic Compute Cloud**

Amazon EC2 is a web service that provides secure, resizable compute capacity in the cloud. It allows you to provision virtual servers called EC2 instances. Although AWS uses the phrase “web service” to describe it, you are not limited to running just web servers on your EC2 instances.

You can create and manage EC2 instances through the AWS Management Console, the AWS Command Line Interface (CLI), AWS software development kits (SDKs), automation tools, and infrastructure orchestration services.

A picture containing graphical user interface

Description automatically generated

Diagram

Description automatically generated with low confidence

A picture containing graphical user interface

Description automatically generated

A picture containing diagram

Description automatically generated