**Customers: Security in the cloud**

Customers are responsible for the security of everything that they create and put *in* the AWS Cloud.

When using AWS services, you, the customer, maintain complete control over your content. You are responsible for managing security requirements for your content, including which **content** you choose to store on AWS, which AWS services you use, and who has **access** to that content. You also control how access rights are granted, managed, and revoked.

The security steps that you take will depend on factors such as the services that you use, the complexity of your systems, and your company’s specific operational and security needs. Steps include **selecting, configuring**, **and patching the operating systems** that will run on Amazon EC2 instances, **configuring security groups**, and **managing user accounts**.

This is your operating system. You're 100% in charge of this. AWS does not have any backdoor into your system here. You and you alone have the only encryption key to log onto the root of this OS or to create any user accounts there.

**AWS: Security of the cloud**

AWS is responsible for security *of* the cloud.

AWS operates, manages, and controls the components at all layers of infrastructure. This includes areas such as the host operating system, the virtualization layer, and even the physical security of the data centres from which services operate.

AWS is responsible for protecting the global infrastructure that runs all of the services offered in the AWS Cloud. This infrastructure includes AWS Regions, Availability Zones, and edge locations.

AWS manages the security of the cloud, specifically the physical infrastructure that hosts your resources, which include:

 Physical security of data centres

 Hardware and software infrastructure

 Network infrastructure

 Virtualization infrastructure

.

AWS Identity and Access Management (IAM)

**AWS Identity and Access Management (IAM)** enables you to manage access to AWS services and resources securely.

IAM gives you the flexibility to configure access based on your company’s specific operational and security needs. You do this by using a combination of IAM features, which are explored in detail in this lesson:

 IAM users, groups, and roles

 IAM policies

 Multi-factor authentication

AWS account root user

When you first create an AWS account, you begin with an identity known as the **root user**.

The root user is accessed by signing in with the email address and password that you used to create your AWS account. You can think of the root user as being similar to the **owner** of the coffee shop. It has **complete access** to **all AWS services and resources** in the account.

IAM users

An **IAM user** is an **identity** that you create in AWS. It represents the person or application that interacts with AWS services and resources. It consists of a name and credentials.

By default, when you create a new IAM user in AWS, it has **no permissions** associated with it. This is based on the **principle of least privilege**, where a user is granted access only to what they need. To allow the IAM user to perform specific actions in AWS, such as launching an Amazon EC2 instance or creating an Amazon S3 bucket, you must **grant the IAM user the necessary permissions.**

IAM policies

An **IAM policy** is a **JSON** document that allows or denies permissions to AWS services and resources.

IAM policies enable you to customize users’ levels of access to resources. For example, you can allow users to access all of the Amazon S3 buckets within your AWS account, or only a specific bucket.

**Best practice:**

Follow the security **principle of least privilege** when granting permissions. By following this principle, you help to prevent users or roles from having more permissions than needed to perform their tasks.

For example, if an employee needs access to only a specific bucket, **specify the bucket in the IAM policy**. Do this instead of granting the employee access to all of the buckets in your AWS account.

Example: IAM policy

Here’s an example of how IAM policies work. Suppose that the coffee shop owner has to create an IAM user for a newly hired cashier. The cashier needs access to the receipts kept in an Amazon S3 bucket with the ID: AWSDOC-EXAMPLE-BUCKET.

There were only two potential options for the **effect** on any policy. **Either allow or deny**. **For action**, you can list any AWS API call and **for resource**, you would list what AWS resource that specific API call is for.

If the owner wants the cashier to be able to access other services and perform other actions in AWS, the owner must **attach additional policies to specify these services** and actions.

Now, suppose that the coffee shop has hired a few more cashiers. Instead of assigning permissions to each individual IAM user, the owner places the users into an **IAM group**.

IAM groups

One way to make it easier to manage your users and their permissions is to organize them into IAM groups. An **IAM group is a collection of IAM users**. When you assign an IAM policy to a group, all users in the group are granted permissions specified by the policy.

Here’s an example of how this might work in the coffee shop. Instead of assigning permissions to cashiers one at a time, the owner can create a “Cashiers” IAM group. The owner can then add IAM users to the group and then **attach permissions at the group level**.

Assigning IAM policies at the group level also makes it easier to adjust permissions when an employee transfers to a different job. For example, if a cashier becomes an inventory specialist, the coffee shop owner removes them from the “Cashiers” IAM group and adds them into the “Inventory Specialists” IAM group. This ensures that employees have only the permissions that are required for their current role.

IAM roles

In the coffee shop, an employee rotates to different workstations throughout the day. Depending on the staffing of the coffee shop, this employee might perform several duties: work at the cash register, update the inventory system, process online orders, and so on.

When the employee needs to switch to a different task, **they give up their access to one workstation and gain access to the next workstation**. The employee can easily switch between workstations, but at any given point in time, they can have access to only a single workstation. This same concept exists in AWS with IAM roles.

An IAM role is an identity that you can assume to gain **temporary access** to permissions.

Roles have associated permissions that allow or deny specific actions. And these roles can be assumed for temporary amounts of time. It is similar to a user, but has **no username and password.** Instead, it is an **identity that you can assume to gain access to temporary permissions.** You use roles to temporarily grant access to AWS resources, to users, external identities, applications, and even other AWS services. When an identity assumes a role, it **abandons all of the previous permissions** that it has and it assumes the permissions of that role. **Best practice:** IAM roles are ideal for situations in which access to services or resources needs to be **granted temporarily**, instead of long-term.

Multi-factor authentication

Have you ever signed in to a website that required you to provide multiple pieces of information to verify your identity? You might have needed to provide your password and then a second form of authentication, such as a random code sent to your phone. This is an example of **multi-factor authentication**. In IAM, multi-factor authentication (MFA) provides an extra layer of security for your AWS account.

**How multi-factor authentication works**

AWS Organizations

With your first foray into the AWS Cloud, you most likely will start with one AWS account and have everything reside in there. Most people start this way, but as your company grows or even begins their cloud journey, it's important to have a separation of duties.

For example, you want your developers to have access to development resources, have your accounting staff able to access billing information, or even have business units separate so that they can experiment with AWS services without effecting each other.

Suppose that your company has multiple AWS accounts. You can use **AWS Organizations** to consolidate and **manage multiple AWS accounts within a central location**. The easiest way to think of Organizations is as a **central location to manage multiple AWS accounts**. You can manage billing control, access, compliance, security, and share resources across your AWS accounts.

When