Module 6 – Security

Contents

Learning objectives

In this module, you will learn how to:

- Explain the benefits of the shared responsibility model.
- Describe multi-factor authentication (MFA).
- Differentiate between AWS Identity and Access Management (IAM) security levels.
- Explain the main benefits of AWS Organizations.
- Describe security policies at a basic level.
- Summarize the benefits of compliance with AWS.
- Explain additional AWS security services at a basic level.

The AWS shared responsibility model

Throughout this course, you have learned about a variety of resources that you can create in the AWS Cloud. These resources include Amazon EC2 instances, Amazon S3 buckets, and Amazon RDS databases. Who is responsible for keeping these resources secure: you (the customer) or AWS?

The answer is both. The reason is that you do not treat your AWS environment as a single object. Rather, you treat the environment as a collection of parts that build upon each other. AWS is responsible for some parts of your environment and you (the customer) are responsible for other parts. This concept is known as the **shared responsibility model**.

The shared responsibility model divides into customer responsibilities (commonly referred to as “security in the cloud”) and **AWS** responsibilities (commonly referred to as “security **of** the cloud”).

You can think of this model as being similar to the division of responsibilities between **a homeowner and a homebuilder**. The builder (AWS) is responsible for constructing your house and ensuring that it is solidly built. As the homeowner (the customer), it is your responsibility to secure everything in the house by ensuring that the doors are closed and locked.

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.

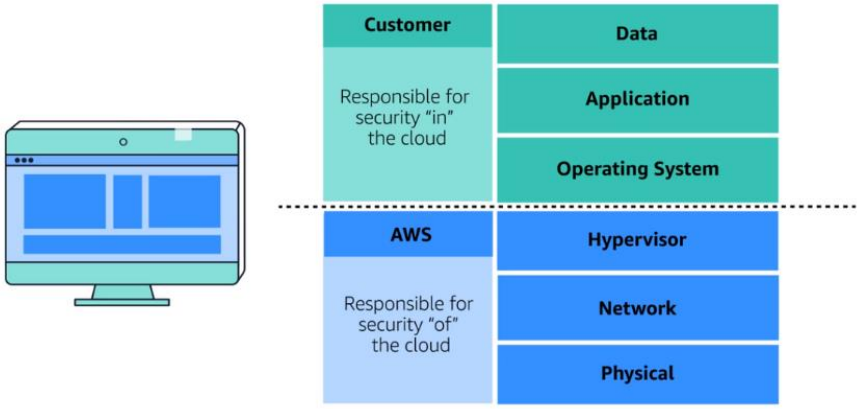
AWS operates, manages, and controls the components at all layers of infrastructure. This includes areas such as the host operating system, the virtualization layer, and even the physical security of the data centres from which services operate.

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

Although you cannot visit AWS data centres to see this protection first-hand, AWS provides several reports from third-party auditors. These auditors have verified its compliance with a variety of computer security standards and regulations.



AWS Identity and Access Management (IAM)

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

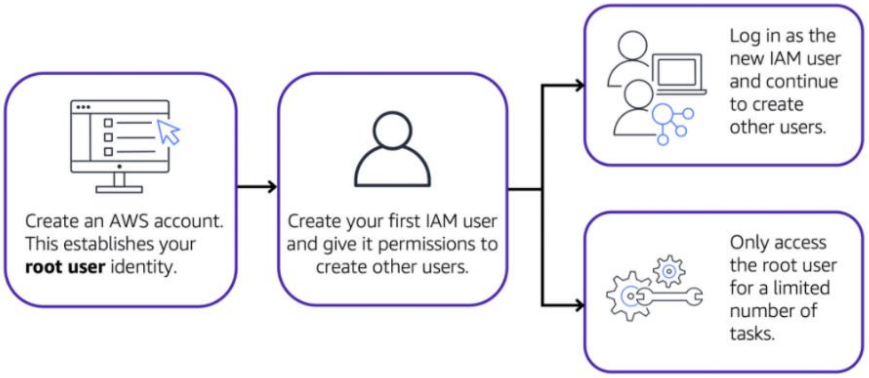
IAM gives you the flexibility to configure access based on your company's specific operational and security needs. You do this by using a combination of IAM features, which are explored in detail in this lesson:

- 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.



Best practice:

Do **not** use the root user for everyday tasks. Instead, use the root user to **create your first IAM user and assign it permissions to create other users**. Because that user is so powerful, we recommend that as soon as you create an AWS account and log in with your root user, you turn on **multi-factor authentication**, or MFA, to ensure that you need not only the email and password, but also a randomized token to log in.

Then, continue to create other IAM users, and access those identities for performing regular tasks throughout AWS. **Only use the root user** when you need to perform a limited number of tasks that are only available to the root user**.** Examples of these tasks include **changing your root user email address** and **changing your AWS support plan**.

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.**

Best practice:

We recommend that you create individual IAM users for each person who needs to access AWS. Even if you have multiple employees who require the same level of access, you should create individual IAM users for each of them. This provides additional security by allowing each IAM user to have a unique set of security credentials.

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.

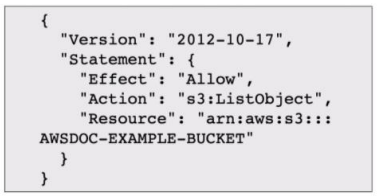
Best practice:

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.



```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "s3:ListObject",
    "Resource": "arn:aws:s3:::
AWSDOC-EXAMPLE-BUCKET"
  }
}
```

This example IAM policy allows permission to access the objects in the Amazon S3 bucket with ID: *AWSDOC-EXAMPLE-BUCKET*.

In this example, the IAM policy is allowing a specific action within Amazon S3: ListObject. The policy also mentions a specific bucket ID: AWSDOC-EXAMPLE-BUCKET. When the owner **attaches this policy to the cashier's IAM user**, it will allow the cashier to view all of the objects in the AWSDOC-EXAMPLE-BUCKET 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

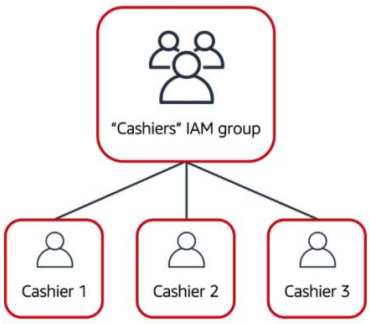
One way to make it easier to manage your users and their permissions is to organize them into IAM groups. An **IAM group is a collection of IAM users**. When you assign an IAM policy to a group, all users in the group are granted permissions specified by the policy.

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.

What if a coffee shop employee hasn’t switched jobs permanently, but instead, rotates to different workstations throughout the day? This employee can get the access they need through **IAM roles**.



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.

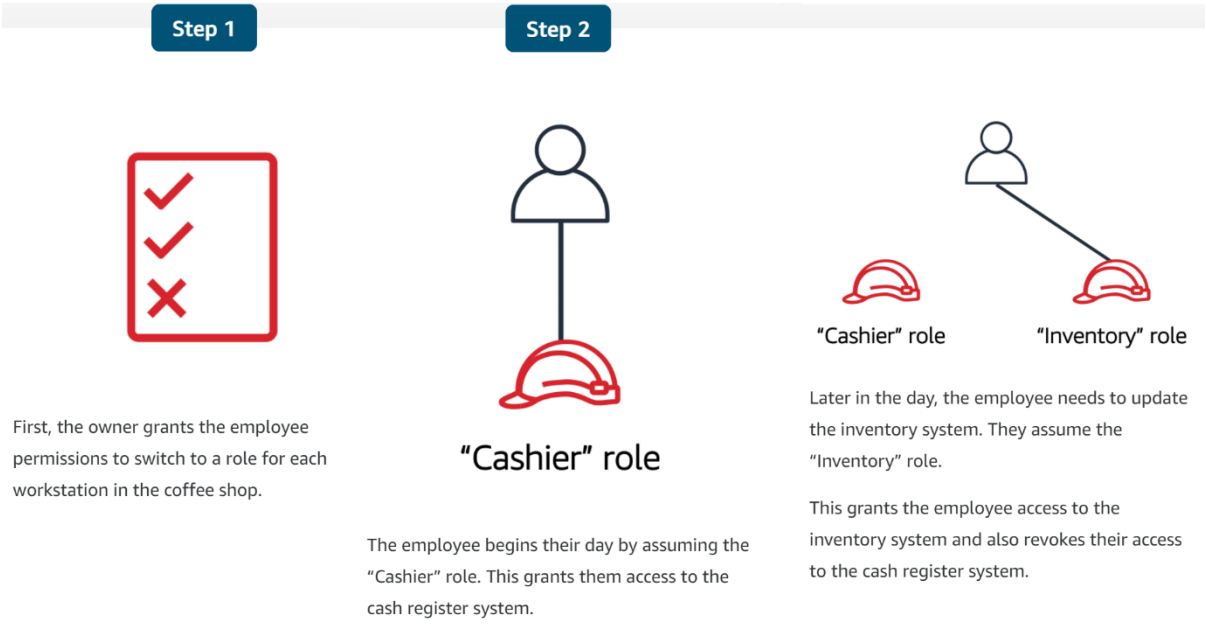
When the employee needs to switch to a different task, **they give up their access to one workstation and gain access to the next workstation**. The employee can easily switch between workstations, but at any given point in time, they can have access to only a single workstation. This same concept exists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

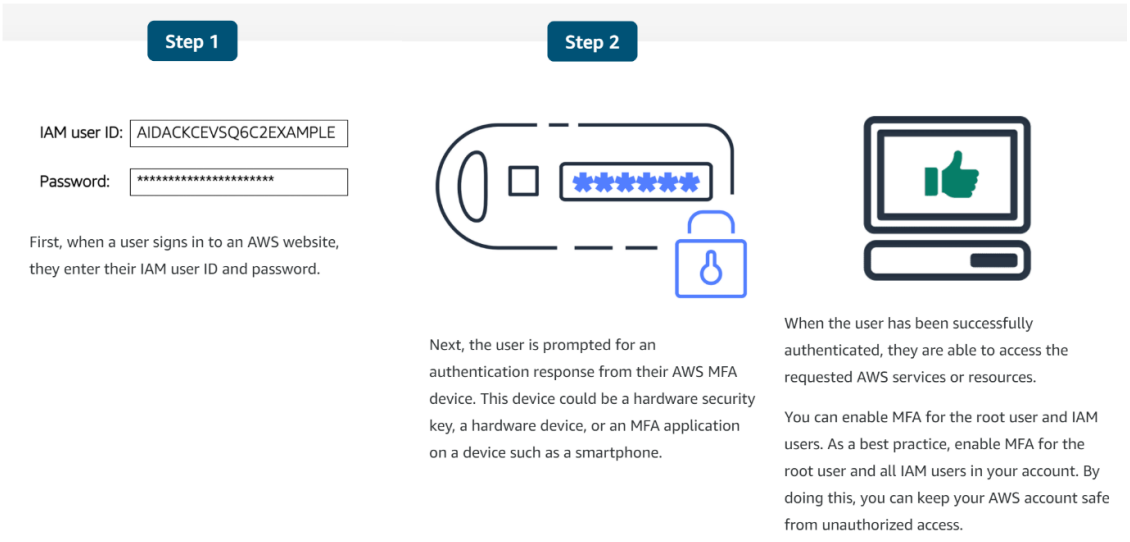
IAM roles are ideal for situations in which access to services or resources needs to be **granted temporarily**, instead of long-term.



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 you create an organization, AWS Organizations automatically creates a **root**, which is the parent container for all the accounts in your organization.

In AWS Organizations, you can **centrally control permissions for the accounts** in your organization by using **service control policies (SCPs)**. SCPs enable you to place restrictions on the AWS services, resources, and individual API actions that users and roles in each account can access. In other words, you can apply service control policies (SCPs) to the **organization root, an individual member account, or an organizational unit (OU)**.

An **SCP affects all IAM users**, groups, and roles within an account, **including the AWS account root user**. You can apply IAM policies to IAM users, groups, or roles, but you **cannot apply an IAM policy to the AWS account root user.**

Consolidated billing is another feature of AWS Organizations. This means you can use the primary account of your organization to consolidate and pay for all member accounts. Another advantage of consolidated billing is **bulk discounts**. You will learn about consolidated billing in a later module.

Organizational units

In AWS Organizations, you can implement hierarchical groupings of your group accounts into organizational units (OUs) to make it easier to manage accounts with similar business or security requirements (kind of like business units). **When you apply a policy to an OU, all the accounts in the OU automatically inherit the permissions** specified in the policy.

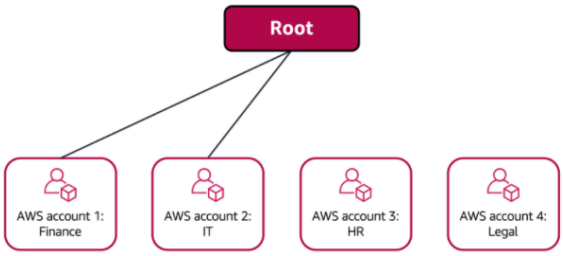
By organizing **separate accounts into OUs**, you can more easily isolate workloads or applications that have specific security requirements. For instance, if your company has accounts that can access only the AWS services that meet certain regulatory requirements, you can put these accounts into one OU. Then, you can **attach a policy to the OU** that blocks access to all other AWS services that do not meet the regulatory requirements.

Step 1

Imagine that your company has separate AWS accounts for the finance, information technology (IT), human resources (HR), and legal departments. You decide to consolidate these accounts into a single organization so that you can administer them from a central location. When you create the organization, this establishes the root.

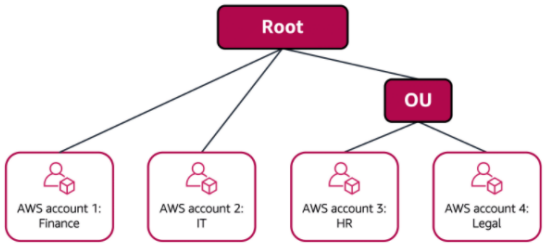
In designing your organization, you consider the business, security, and regulatory needs of each department. You use this information to decide which departments group together in OUs.

Step 2



The finance and IT departments have requirements that do not overlap with those of any other department. You bring these accounts into your organization to take advantage of benefits such as consolidated billing, but you do not place them into any OUs.

Step 3



The HR and legal departments need to access the same AWS services and resources, so you place them into an OU together**. Placing them into an OU enables you to attach policies that apply to both** the HR and legal departments' AWS accounts.

Even though you have placed these accounts into OUs, you can continue to provide access for users, groups, and roles through IAM.

By grouping your accounts into OUs, you can more easily give them access to the services and resources that they need. You also prevent them from accessing any services or resources that they do not need.

AWS Artifact

Depending on your company's industry, you may need to **uphold specific standards**. An audit or inspection will ensure that the company has met those standards. For example, you could be audited for taxes to see that you have run the back office correctly and have followed the law. You rely on documentation, records and inspections to pass audits and compliance checks as they come along. You'll need to devise a similar way to meet compliance and auditing in AWS.

AWS complies with a **long list of assurance programs** that you can find online. This means that segments of your compliance have already been completed, and you can focus on meeting compliance within your own architectures that you build on top of AWS.

The next thing to know in regards to compliance and AWS, is that the Region you choose to operate out of, might help you meet compliance regulations. If you can only legally store data in the country that the data is from, you can choose a Region that makes sense for you and AWS will not automatically replicate data across Regions.

AWS offers **multiple whitepapers and documents that you can download and use for compliance reports.** Since you aren't running the data centre yourself, you can essentially request that AWS provides you with documentation proving that they are following best practices for security and compliance. One place you can access these documents is through a service called AWS Artifact.

AWS Artifact is a service that provides **on-demand access to AWS security and compliance reports and select online agreements**. With AWS Artifact, you can gain access to compliance reports done by third parties who have validated a wide range of compliance standards. AWS Artifact consists of two main sections: AWS Artifact Agreements and AWS Artifact Reports.

AWS Artifact Agreements

Suppose that your company needs to **sign an agreement with AWS** regarding your use of certain types of information throughout AWS services. You can do this through **AWS Artifact Agreements**.

In AWS Artifact Agreements, you can review, accept, and manage agreements for an individual account and for all your accounts in AWS Organizations. Different types of agreements are offered to address the needs of customers who are subject to specific regulations, such as the Health Insurance Portability and Accountability Act (**HIPAA**). Another example is that if you run software that deals with consumer data in the EU, you

would need to make sure that you're in compliance with GDPR.

AWS Artifact Reports
Next, suppose that a member of your company's development team is building an application and needs more information about their responsibility for complying with certain regulatory standards. You can advise them to access this information in **AWS Artifact Reports**.

AWS Artifact Reports provide **compliance reports from third-party auditors**. These auditors have **tested and verified that AWS is compliant** with a variety of global, regional, and industry-specific security standards and regulations. AWS Artifact Reports remains up to date with the latest reports released. You can **provide the AWS audit artifacts to your auditors or regulators as evidence of AWS security controls**.

The following are some of the compliance reports and regulations that you can find within AWS Artifact. Each report includes a description of its contents and the reporting period for which the document is valid.

Customer Compliance Centre

The **Customer Compliance Centre** contains resources to help you learn more about AWS compliance.

In the Customer Compliance Centre, you can read customer compliance stories to discover how companies in regulated industries have solved various compliance, governance, and audit challenges. You can also access compliance whitepapers and documentation on topics such as:

- AWS answers to key compliance questions
- An overview of AWS risk and compliance
- An auditing security checklist

Additionally, the Customer Compliance Centre includes an **auditor learning path**. This learning path is designed for individuals in auditing, compliance, and legal roles who want to learn more about how their internal operations can demonstrate compliance using the AWS Cloud.

To know if you are compliant in AWS, please remember that we follow a shared responsibility. The underlying platform is secure and **AWS can provide documentation on what types of compliance requirements they meet**, through services like AWS Artifact and whitepapers. But, beyond that, what you build on AWS is up to you. You control the architecture of your applications and the solutions you build, and they need to be built with compliance, security, and the shared responsibility model in mind.

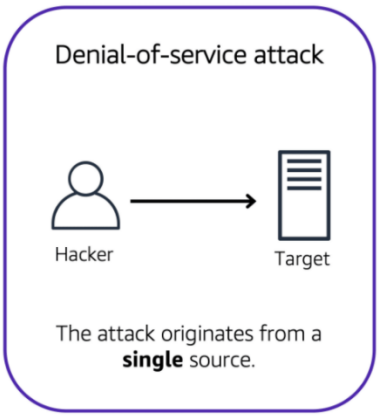
Denial-of-service attacks

Customers can call the coffee shop to place their orders. After answering each call, a cashier takes the order and gives it to the barista.

However, suppose that a prankster is calling in multiple times to place orders but is never picking up their drinks. This causes the cashier to be unavailable to take other customers' calls. The coffee shop can attempt to stop the false requests by blocking the phone number that the prankster is using.

In this scenario, the prankster's actions are similar to a **denial-of-service attack**.

A **denial-of-service (DoS) attack** is a deliberate attempt to make a website or application unavailable to users.

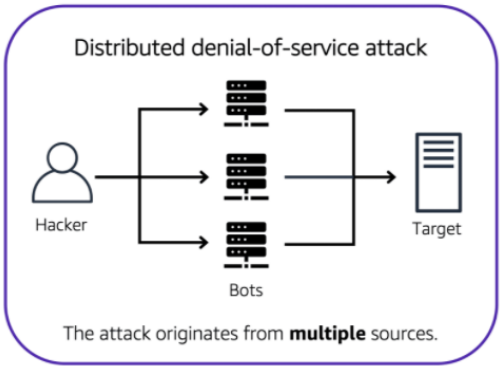


For example, an attacker **might flood a website or application with excessive network traffic until the targeted website or application becomes overloaded and is no longer able to respond**. If the website or application becomes unavailable, this denies service to users who are trying to make legitimate requests.

Distributed denial-of-service attacks

A single machine attacking your application has no hope of providing enough of an attack by itself, so the distributed part is that the attack leverages other machines around the internet to unknowingly attack your infrastructure. Now, suppose that the prankster has enlisted the help of friends.

The prankster and their friends repeatedly call the coffee shop with requests to place orders, even though they do not intend to pick them up. These requests are coming in from different phone numbers, and it's impossible for the coffee shop to block them all. Additionally, the influx of calls has made it increasingly difficult for customers to be able to get their calls through. This is similar to a **distributed denial-of-service attack**.



In a distributed denial-of-service (DDoS) attack, multiple sources are used to start an attack that aims to make a website or application unavailable. This can come from a group of attackers, or even a single attacker. The single attacker can use multiple infected computers (also known as "bots") to send excessive traffic to a website or application.

Examples of DDoS Attacks

UDP Flood

It is based on the helpful parts of the internet, like the National Weather Service. Now anyone can send a small request to the Weather Service, and ask, "Give me weather," and in return, the Weather Service's fleet of machines will send back a massive amount of weather telemetry, forecasts, updates, lots of stuff. So the attack here is simple. The bad actor sends a simple request, give me weather. But it gives a **fake return address** on the request, **your return address**. So now the Weather Service very happily floods your server with megabytes of rain forecasts, and your system could be brought to a standstill, just sorting through the information it never wanted in the first place.

HTTP level attacks

Some attacks are much more sophisticated, like the HTTP level attacks, which look like normal customers asking for normal things like complicated product searches over and over

and over, all coming from an army of zombified bot machines. They ask for so much attention that regular customers can't get in.

Slowloris attack

Imagine standing in line at the coffee shop, when someone in front of you takes seven minutes to order whatever it is they're ordering, and you don't get to order until they finish and get out of your way. Well, Slowloris attack is the exact same thing. Instead of a normal connection, I would like to place an order, the attacker pretends to have a terribly slow connection. You get the picture. Meanwhile, your production servers are standing there waiting for the customer to finish their request so they can dash off and return the result. But until they get the entire packet, they can't move on to the next thread, the next customer. A few Slowloris attackers can exhaust the capacity of your entire front end with almost no effort at all.

To help minimize the effect of DoS and DDoS attacks on your applications, you can use **AWS Shield**.

AWS Shield

AWS Shield is a service that protects applications against DDoS attacks. AWS Shield provides two levels of protection: Standard and Advanced.

AWS Shield Standard automatically protects all AWS customers **at no cost**. It protects your AWS resources from the most common, frequently occurring types of DDoS attacks.

As network traffic comes into your applications, AWS Shield Standard uses a variety of analysis techniques (e.g. security groups) to detect malicious traffic in real time and automatically mitigates it.

AWS Shield Advanced is a **paid** service that provides detailed attack diagnostics and the ability to detect and mitigate sophisticated DDoS attacks.

It also integrates with other services such as Amazon CloudFront, Amazon Route 53, and Elastic Load Balancing. Additionally, you can **integrate AWS Shield with AWS WAF** by writing custom rules to mitigate complex DDoS attacks.

For the sharpest, most sophisticated attacks, AWS offers specialized defence tools called AWS Shield with **AWS WAF**. AWS WAF uses a **web application firewall** to filter incoming traffic for the signatures of bad actors. It has extensive **machine learning capabilities**, and can recognize new threats as they evolve and proactively help defend your system against an ever-growing list of destructive vectors.

AWS Key Management Service (AWS KMS)

The coffee shop has many items, such as coffee machines, pastries, money in the cash registers, and so on. You can think of these items as data. The coffee shop owners want to ensure that all of these items are secure, whether they're sitting in the storage room or being transported between shop locations.

In the same way, you must ensure that your applications' data is secure while in storage **(encryption at rest)** and while it is transmitted, known as **encryption in transit**. **Encryption is the securing of a message or data in a way that only authorized parties can access it.**

AWS Key Management Service (AWS KMS) enables you to perform encryption operations through the use of **cryptographic keys**. A cryptographic key is a random string of digits used for locking (encrypting) and unlocking (decrypting) data. You can **use AWS KMS to create, manage, and use cryptographic keys.** You can also control the use of keys across a wide range of services and in your applications.

With AWS KMS, you can choose the **specific levels of access control** that you need for your keys. For example, you can specify **which IAM users and roles** are able to manage keys. Alternatively, you can temporarily disable keys so that they are no longer in use by anyone. Your keys never leave AWS KMS, and you are always in control of them.

For example, server-side encryption at rest is enabled on all DynamoDB table data. And that helps prevent unauthorized access. **DynamoDB's encryption at rest** also integrates with AWS KMS, or Key Management Service, for managing the encryption key that is used to encrypt your tables. That's the key for your door, remember? And without it, you won't be able to access your data.

Similarly, in-transit means that the data is traveling between, say A and B. Where A is the AWS service, and B could be a client accessing the service. Or even another AWS service itself. For example, let's say we have a Redshift instance running. And we want to connect it with a SQL client. We use **secure sockets layer**, or **SSL connections** to encrypt data, and we can use service certificates to validate, and authorize a client. This means that **data is protected when passing between Redshift, and our client.** And this functionality exists in numerous other AWS services such as SQS, S3, RDS, and many more.

AWS WAF

AWS WAF is a web application firewall that lets you **monitor network requests** that come into your web applications.

AWS WAF works together with Amazon CloudFront and an Application Load Balancer. Recall the network access control lists that you learned about in an earlier module. AWS WAF works in a similar way to block or allow traffic. However, it does this by using a **web access control list (ACL)** to protect your AWS resources.

Here’s an example of how you can use AWS WAF to allow and block specific requests.

Suppose that your application has been receiving malicious network requests from several IP addresses. You want to prevent these requests from continuing to access your application, but you also want to ensure that legitimate users can still access it. You configure the web ACL to allow all requests except those from the IP addresses that you have specified.

When a request comes into AWS WAF, it checks against the list of rules that you have configured in the web ACL. If a request did not come from one of the blocked IP addresses, it allows access to the application.

However, if a request came from one of the blocked IP addresses that you have specified in the web ACL, it is denied access.

Amazon Inspector

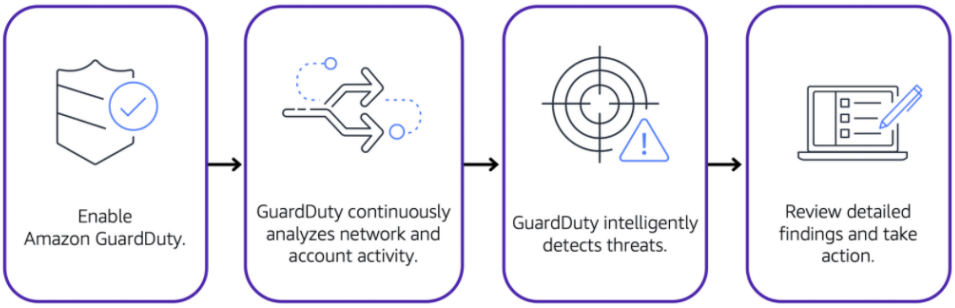
Suppose that the developers at the coffee shop are developing and testing a new ordering application. They want to make sure that they are designing the application in accordance with security best practices. However, they have several other applications to develop, so they cannot spend much time conducting manual assessments. To perform **automated security assessments**, they decide to use **Amazon Inspector**.

Amazon Inspector helps to improve the security and compliance of applications by running automated security assessments. It **checks applications for security vulnerabilities and deviations from security best practices**, such as open access to Amazon EC2 instances and installations of vulnerable software versions.

After Amazon Inspector has performed an assessment, it provides you with a list of security findings. The list prioritizes by severity level, including a detailed description of each security issue and a recommendation for how to fix it. However, AWS does not guarantee that following the provided recommendations resolves every potential security issue. Under the shared responsibility model, customers are responsible for the security of their applications, processes, and tools that run on AWS services.

Amazon GuardDuty

Amazon GuardDuty is a service that provides **intelligent threat detection** for your AWS infrastructure and resources. It **identifies threats by continuously monitoring the network activity and account behaviour within your AWS environment**.



After you have enabled GuardDuty for your AWS account, GuardDuty begins monitoring your network and account activity. You do not have to deploy or manage any additional security software. GuardDuty then continuously analyzes data from multiple AWS sources, including VPC Flow Logs and DNS logs.

It uses integrated threat intelligence such as known **malicious IP addresses, anomaly detection, and machine learning to identify threats more accurately**. The best part is that it runs **independently from your other AWS services**. So it won't affect performance or availability of your existing infrastructure, and workloads.

If GuardDuty detects any threats, you can review detailed findings about them from the AWS Management Console. Findings include recommended steps for remediation. You can also configure AWS Lambda functions to take remediation steps automatically in response to GuardDuty's security findings.

Summary

Alright, it's time to break down what we just covered. First things first, AWS follows a **shared responsibility model**. AWS is responsible for security of the cloud, and you are responsible for security in the cloud.

With **IAM**, you have users, groups, roles, and policies. Users log in with a username and password and by default they have no permissions. Groups are groupings of users and roles

are identities that you can assume to gain access to temporary credentials and permissions for a configurable amount of time. In order to give permissions to an identity, you need to create **policies** that either explicitly allow or deny a specific action in AWS. With IAM also comes identity federation.

If you have an existing corporate identity store, you can federate those users to AWS, using **role based access**, which allows your users to use one login for both your corporate systems as well as AWS. One final point to remember about IAM is that you should make sure that you turn on **multi-factor authentication** for users, but especially for your root user which has all the permissions by default and cannot be restricted.

Next up, we discussed **AWS Organizations**. With AWS, it's likely you'll have multiple accounts. Accounts are commonly used to isolate workloads, environments, teams, or applications. AWS Organizations helps you manage multiple accounts in a hierarchical fashion.

We then discussed **compliance**. AWS uses third-party auditors to prove its adherence to a wide variety of compliance programs. You can use the **AWS Compliance Center** to find more information on compliance and **AWS Artifact** to **gain access to compliance documents**. The compliance requirements you have will vary from application to application and between areas of operation.

Then we talked about distributed denial-of-service attacks, or **DDoS** attacks, and how to combat them with AWS using **tools like ELB, security groups, AWS Shield, and AWS WAF**. We also talked about encryption. In AWS, you are the owner of your data, and you are responsible for security. That means you need to pay attention to encryption, in transit and at rest.

There are lots of considerations when dealing with security in AWS. Security is AWS's top priority, and will continue to be so. Please make sure you read the documentation on securing your AWS resources, as it does vary from service to service.

Use the **least privilege principle when scoping permissions for users** and roles in IAM, **encrypt your data at every layer, both in transit and at rest.** And make sure you use AWS services to protect your environment.

Quiz

Which statement best describes an IAM policy?

- A document that grants or denies permissions to AWS services and resources. IAM policies provide you with the flexibility to customize users’ levels of access to resources.

An employee requires temporary access to create several Amazon S3 buckets. Which option would be the best choice for this task?

- IAM role

Which statement best describes the principle of least privilege?

- Granting only the permissions that are needed to perform specific tasks. When you grant permissions by following the principle of least privilege, you prevent users or roles from having more permissions than needed to perform specific job tasks.

Which service helps protect your applications against distributed denial-of-service (DDoS) attacks?

- AWS Shield. As network traffic comes into your applications, AWS Shield uses a variety of analysis techniques to detect potential DDoS attacks in real time and automatically mitigates them.

Which task can AWS Key Management Service (AWS KMS) perform?

- Create cryptographic keys

Additional resources

To learn more about the concepts that were explored in Module 6, review these resources.

- Security, Identity, and Compliance on AWS
- Whitepaper: Introduction to AWS Security
- Whitepaper: Amazon Web Services - Overview of Security Processes
- AWS Security Blog
- AWS Compliance
- AWS Customer Stories: Security, Identity, and Compliance