Module 6 Introduction

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 the 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.**

To start the module introduction video, choose the play button.

Play Video

To access a transcript of the video, choose the following transcript.

Video transcript

Looks like we're getting deeper into our AWS account. Things are running well. And I want to let you in on the security measures we have in place. By this, I mean, we have to describe the various security mechanisms we offer on the AWS Cloud, like the shared responsibility model.

With the shared responsibility model, AWS controls security of the cloud and customers control security in the cloud. We, as AWS, control the data centers, security of our services, and all the layers in this section. The next part are the workloads that AWS customers run in the cloud and those are the customer's responsibility to secure. It's something we share with customers to ensure security in the cloud.

Let's take a look at the various other security services, mechanisms, and features that AWS has to offer in this module. Stay tuned.

AWS Shared Responsibility Model

To start the video on the AWS shared responsibility model, choose the play button.

Play Video

To access a transcript of the video, choose the following transcript.

Video transcript

When it comes to securing your business on AWS, it's important to ask the question: Who is ultimately responsible for the security? Is it A: You, the customer? or B: AWS? And the correct answer is: yes. Both. Both are ultimately responsible for making sure that you are secure.

Now, if there's any security experts watching this right now you're probably shaking your head saying you can't have two different entities with the ultimate responsibility over a single object. That's not security, that's wishful thinking. At AWS, we agree completely. But for us, we don't look at your environment as a single object. Instead we see it as a collection of parts that build on each other. AWS is responsible for the security of some of the objects. Responsible 100% for those. The others, you are responsible 100% for their security. This is what's known as the shared responsibility model.

It's no different than securing a house. The builder constructed the house with four walls and a door. It's their responsibility to make sure the walls are strong and the doors are solid. It's your responsibility, the homeowner, to close and lock the doors.

It really is that simple on AWS as well. Take EC2, for instance. EC2 lives in a physical building, a data center that must be secured. It has a network and a hypervisor that supports your instances and their individual operating systems. On top of that operating system, you have your application, and that supports your data. So for EC2 and every service AWS offers, there's a similar stack of parts that build on top of each other. AWS is 100% responsible for some, you are responsible for the others.

So, starting with the physical layer. This is iron and concrete and fences and security guards. Someone has to own the concrete. Someone has to staff the physical perimeter, 24/7. This is AWS. On top of the physical layer we have our network and our hypervisor. Now, I'm not gonna go into details on how this is all secured, but basically we have reinvented those technologies to make them faster, stronger, tamper-proof.

But you don't have to just take our word for it. We have numerous third party auditors who have gone through the code and the way we build our infrastructure, and can provide the right documentation you need for your security compliance structures. Now, on top of all that, on EC2, you now get to pick what operating system you want to run.

This is the magic dividing line that separates our responsibility. AWS' responsibility and your responsibility. 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. I mean, no more than a construction company would keep copies of your front door key, AWS cannot enter your operating system. And here's a hint. If someone from AWS calls and asks you for your OS key, it is not AWS.

Now that means your operations team is 100% responsible for keeping the operating system patched. If AWS discovers there are some new vulnerabilities in your version of Windows, let's say, we can

certainly notify your account owner but we cannot deploy a patch. This is a really good thing for your security. This means no one can deploy anything that might break your system without your team being the ones that do it. Now, on top of that OS, you can run whatever applications you want. You own them. You maintain them.

Which takes us to the most important part of the stack, your data. Data. This is always your domain to control. And sometimes you might want to have your data open for everyone to see, like pictures on a retail website. Other times like banking or healthcare, yeah, not so much, not so much. AWS provides everyone with the tool set they need for their data to open it up to some authorized individuals, to everyone, to just a single person under specific conditions, or even lock it down so no one can access it. Plus, the ability to have ubiquitous encryption. That way even if you accidentally left your front door open, all anyone would see is unreadable encrypted content.

The AWS shared responsibility model is about making sure both sides understand exactly what tasks are ours. Basically, AWS is responsible for the security of the cloud and you are responsible for the security in the cloud. Together, you have an environment you can trust.

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**(opens in a new tab).

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.

To learn more about the shared responsibility model, expand each of the following two categories.

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.

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 centers 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 centers
- Hardware and software infrastructure
- Network infrastructure
- Virtualization infrastructure

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

Knowledge check

For guidance on navigating this question using the keyboard, expand the following keyboard instructions.

Keyboard instructions

Which tasks are the responsibilities of customers? (Select TWO.)

- Maintaining network infrastructure
- Patching software on Amazon EC2 instances
- Implementing physical security controls at data centers
- Setting permissions for Amazon S3 objects
- Maintaining servers that run Amazon EC2 instances

User Permissions and Access

To start the video on user permissions and access, choose the play button.

Play Video

To access a transcript of the video, choose the following transcript.

Video transcript

In the coffee shop, every employee has an identity. They come into work in the morning and they'd log into the system to clock in, use the registers and manage the systems, running the coffee shop, day to day. We have the cash registers, the computers helping run the whole operation. Each person has unique access to these systems based on who they are.

If I have Rudy at the register taking orders and Blaine in the back, checking on the inventory levels on the computer, they have two different logins and two different sets of permissions. Rudy can run the cash register, but if he went to log into the inventory system, he wouldn't be allowed to do that.

You will want to scope your users permissions in AWS in a similar way. When you create an AWS account, you are given what is called the root account user. This root user is the owner of the AWS account and has permission to do anything they want inside of that account. This is like being the owner of the coffee shop.

In this situation, let's say I am the owner of the coffee shop. I can come into the shop, use my credentials to work the register, work the inventory system or any other system in the coffee shop. I cannot be restricted. With the AWS root user, you can access and control any resource in the account. You can spin up databases, EC2 instances, blockchain services, or literally whatever you want. 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.

That is great. But even with MFA turned on, in reality you don't want to use the root user for everything. I, as the coffee shop owner, don't give my level of access to all employees. Rudy on the cash register cannot access the inventory system, remember? You control access in a granular way by using the AWS service, AWS Identity and Access Management, or IAM.

In IAM, you can create IAM users. When you create an IAM user, by default, it has no permissions. The user can't even log into the AWS account at first, it has absolutely zero permissions. It can not launch an EC2 instance. It can not create an S3 bucket. Nothing. You have to explicitly give the user permission to do anything in that account. Remember, by default, all actions are denied. You have to explicitly allow any action done by any user. You give people access only to what they need and nothing else. This idea is called the least privilege principle.

The way that you grant or deny permission is to associate what is called an IAM policy to an IAM user. An IAM policy is a JSON document that describes what API calls a user can or cannot make.

Let's look at this quick example. In this example, you can see we have a permission statement that has the effect as Allow, the action as s3:ListBucket. And the resource is a unique ID for an S3 bucket. So if I attach this policy to a user and that user could view the bucket "coffee_shop_reports" but perform no other action in this account. 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. Now, as a business person, you wouldn't need to write these policies, but they are used all over in your AWS accounts.

One way to make it easier to manage your users and their permissions is to organize them into IAM groups. Groups are, well, they are groupings of users. You can attach a policy to a group and all of the users in that group will have those permissions. If you have a bunch of cashiers in the coffee shop, instead of individually granting them all access to the register. Instead, you can grant all cashiers access then just add each individual cashier to the group. Same idea with groups in IAM.

All right, so far with IAM, you have the root user, they can do anything. You have users which can be organized into groups. And you also have policies which are documents that describe permissions that you can then attach to users or groups. There is one other major identity in IAM, and it's called a role.

To understand the idea of roles, let's think about the coffee shop. As we know, Blaine works in the shop and depending on the staffing of the shop day to day, he might work the register or the inventory system or he might be the one cleaning up at the end of the day with access to no systems. I, as the owner, have the authority to assign these different roles to Blaine. His responsibilities and access are variable and change from day to day. Just because he worked on tracking inventory in the system yesterday, doesn't mean that he should be at any time. His role at work changes and is temporary in nature. The same type of idea exists in AWS. You can create identities in AWS that are called roles.

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.

You can actually avoid creating IAM users for every person in your organization by federating users into your account. This means that they could use their regular corporate credentials to log into AWS by mapping their corporate identities to IAM roles. AWS IAM authentication and authorization as a service.

AWS Identity and Access Management (IAM)

AWS Identity and Access Management (IAM)(opens in a new tab) 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

You will also learn best practices for each of these features.

AWS account root user

When you first create an AWS account, you begin with an identity known as the **root user**(opens in a new tab).

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 the 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.

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. For more information, see "Tasks that require root user credentials" in the AWS Account Management Reference Guide(opens in a new tab).

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. 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 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.

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.

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**(opens in a new tab).

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**(opens in a new tab).

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.

Before an IAM user, application, or service can assume an IAM role, they must be granted permissions to switch to the role. When someone assumes an IAM role, they abandon all previous permissions that they had under a previous role and assume the permissions of the new 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.

To review an example of how IAM roles could be used in the coffee shop example, choose the arrow buttons to display each of the following two steps.

Step 1

First, the coffee shop owner grants the employee permissions to the "Cashier" and "Inventory" roles so they can switch between these two workstations.

1. 1
2. 2
3.

Step 2

The employee begins their day by assuming the “Cashier” role. This grants them access to the cash register system.

1. 1
2. 2
3.

Conclusion

Later in the day, the employee needs to update the inventory system. They assume the “Inventory” role.

This grants the employee access to the inventory system and also revokes their access to the cash register system.

START AGAIN

1. 1
2. 2
3.

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**(opens in a new tab).

In IAM, multi-factor authentication (MFA) provides an extra layer of security for your AWS account.

To learn more about how MFA works, choose the arrow buttons to display each of the following two steps.

AWS Organizations

To start the video on AWS Organizations, choose the play button.

Play Video

To access a transcript of the video, choose the following transcript.

Video transcript

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. So you start to add more cards for each person, whoever needs to onboard. And before you know it, you end up with a tangled bowl of AWS account spaghetti, not as tasty as you might imagine.

For example, you'll then have to keep track of Account A, F, and G, or maybe Account B has the wrong permissions and Account C has billing and compliance info. One way to install order and to enforce who is allowed to perform certain functions in what account is to make use of an AWS service called AWS Organizations.

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. Let's outline some of the main features of AWS Organizations, shall we?

The first is centralized management of all your AWS accounts. Think of all those AWS accounts, we had: A, B, C, F, G. Now you can combine them into an organization that enables us to manage the accounts centrally, and wow. Now we've found Accounts D and E in the process. Next up is consolidated billing for all member accounts. 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. Cash money, indeed.

The next feature is that you can implement hierarchical groupings of your accounts to meet security, compliance, or budgetary needs. This means you can group accounts into organizational units, or OUs, kind of like business units, or BUs. For example, if you have accounts that must access only the AWS services that meet certain regulatory requirements, you can put those accounts into one OU, or if you have accounts that fall under the developer OU, you can group them accordingly.

One of the last main features we'll touch upon is that you have control over the AWS services and API actions that each account can access as an administrator of the primary account of an organization. You can use something called service control policies, or SCPs, to specify the maximum permissions for member accounts in the organization. In essence, with SCPs you can restrict which AWS services, resources, and individual API actions, the users and roles in each member account can access.

AWS Organizations

Suppose that your company has multiple AWS accounts. You can use **AWS Organizations**(opens in a new tab) to consolidate and manage multiple AWS accounts within a central location.

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)**(opens in a new tab). SCPs enable you to place restrictions on the AWS services, resources, and individual API actions that users and roles in each account can access.

Consolidated billing is another feature of AWS Organizations. You will learn about consolidated billing in a later module.

Organizational units

In AWS Organizations, you can group accounts into organizational units (OUs) to make it easier to manage accounts with similar business or security requirements. 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.

To review an example of how a company might use AWS Organizations, choose the arrow buttons to display each step.

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.

1. 1
2. 2
3. 3
4.

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.

1. 1
2. 2
3. 3
4.

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 empowers you to attach policies that apply to both the HR and legal departments' AWS accounts.

1. 1
2. 2
3. 3
4.

Conclusion

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 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.

START AGAIN

1. 1
2. 2
3. 3
4.

Knowledge check

For guidance on navigating this question using the keyboard, expand the following keyboard instructions.

Keyboard instructions

You are configuring service control policies (SCPs) in AWS Organizations. Which identities and resources can SCPs be applied to? (Select TWO.)

- IAM users
- IAM groups
- An individual member account
- IAM roles
- An organizational unit (OU)

SUBMIT

Compliance

To start the video on compliance, choose the play button.

Play Video

To access a transcript of the video, choose the following transcript.

Video transcript

For every industry, there are specific standards that need to be upheld, and you will be audited or inspected to ensure that you have met those standards. For example, for a coffee shop, the health inspector will come by and check that everything is up to code and sanitary. Similarly, 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. Depending on what types of solutions you host on AWS, you will need to ensure that you are up to compliance for whatever standards and regulations your business is specifically held to. 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, or if you run healthcare applications in the US you will need to design your architectures to meet HIPAA compliance requirements.

Whatever your compliance need is, you'll need some tools to be able to collect documents, records and inspect your AWS environment to check if you meet the compliance regulations that you're under. The first thing to note is, AWS has already built out data center infrastructure and networking following industry best practices for security, and as an AWS customer, you inherit all the best practices of AWS policies, architecture, and operational processes.

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.

You also should be very aware of the fact that you own your data in AWS. As shown in the AWS shared responsibility model, you have complete control over the data that you store in AWS. You can employ multiple different encryption mechanisms to keep your data safe, and that varies from service to service. So, if you need specific standards for data storage, you can devise a way to either reach those requirements by building it yourself on top of AWS or using the features that already exist in many services. For a lot of services, enabling data protection is a configuration setting on the resource.

AWS also offers multiple whitepapers and documents that you can download and use for compliance reports. Since you aren't running the data center 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. With AWS Artifact, you can gain access to compliance reports done by third parties who have validated a wide range of compliance standards. Check out the AWS Compliance Center in order to find compliance information all in one place. It will show you compliance enabling services as well as documentation like the AWS Risk and Security Whitepaper, which you should read to ensure that you understand security and compliance with AWS.

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.

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.

AWS Artifact(opens in a new tab) is a service that provides on-demand access to AWS security and compliance reports and select online agreements. AWS Artifact consists of two main sections: AWS Artifact Agreements and AWS Artifact Reports.

To learn more, expand each of the following two categories.

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).

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.

AWS Artifact provides access to AWS security and compliance documents, such as AWS ISO certifications, Payment Card Industry (PCI) reports, and Service Organization Control (SOC) reports. To learn more about the available compliance reports, visit AWS Compliance Programs(opens in a new tab).

Customer Compliance Center

The **Customer Compliance Center**(opens in a new tab) contains resources to help you learn more about AWS compliance.

In the Customer Compliance Center, 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 Center 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.

Knowledge check

For guidance on navigating this question using the keyboard, expand the following keyboard instructions.

Keyboard instructions

Select the correct answers and choose SUBMIT. For keyboard only accessibility, press TAB to navigate to the correct answer, press SPACEBAR to select. Repeat step until all correct responses are checked, then press ENTER to submit.

Which tasks can you complete in AWS Artifact? (Select TWO.)

- Access AWS compliance reports on-demand.
- Consolidate and manage multiple AWS accounts within a central location.
- Create users to enable people and applications to interact with AWS services and resources.
- Set permissions for accounts by configuring service control policies (SCPs).
- Review, accept, and manage agreements with AWS.

SUBMIT

1.
2.
3.
    1.
    2.
    3.
4.
    1.
    2.
    3.
    4.
    5.
    6.
    7.
    8.
    9.
5.
    1.
    2.

3.
4.
5.
6.

6.

1.
2.
3.
4.
5.
6.

7.

1.
2.
3.
4.
5.
6.
7.
8.
9.
10.
11.

8.

1.
2.
3.
4.
5.
6.
7.
8.

9.
9.
    1.
    2.
    3.
    4.
    5.
    6.
10.
    1.
    2.
    3.
    4.
    5.
    6.
    7.
    8.
    9.
    10.
    11.
11.
    1.
    2.
    3.
    4.
    5.
    6.
    7.
12.
    1.
    2.
    3.

4.
5.
13.
    1.
    2.
    3.
14.
    1.
15.
    1.

Lesson 42 - Compliance

Lesson content

Denial-of-Service Attacks

To start the video on denial-of-service attacks, choose the play button.

Play Video

To access a transcript of the video, choose the following transcript.

Video transcript

D-D-o-S, DDoS, the distributed denial-of-service. It's an attack on your enterprise's infrastructure, and you've heard of it. Your security team might have written a plan for it, and you know that many businesses have been devastated by it. But what exactly is it, and more importantly, how can you defend against it?

Now to be clear, this is a 14-hour discussion to really understand it all, but you need to at least know the fundamentals of how the attacks are carried out, and how AWS can automatically defend your infrastructure from these crippling assaults. Now we don't have a lot of time to cover all this, and the clock starts now. The objective of a DDoS attack is to shut down your application's ability to function by overwhelming the system to the point it can no longer operate.

In normal operations, your application takes requests from customers and returns results. In a DDoS attack, the bad actor tries to overwhelm the capacity of your application, basically to deny anyone your services. But 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. The bad actor creates an army of zombie bots, brainlessly assaulting your enterprise. The key to a good attack, and I call it that when I should call it powerful. I mean, it's definitely chaotic evil, but the key is to have the assault commander do the smallest amount

of work needed, and have the targeted victim receive an unbearable load of resulting work they must process through.

So let me cherry-pick a few specific attack examples that work really well. The 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. Now that is one example of half a dozen low-level, brute force attacks, all designed to exhaust your network.

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.

They even try horrible tricks like the Slowloris attack. Mm-hmm. Imagine standing in line at the coffee shop, when someone in front of you takes seven minutes to order their 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. I could go on monologuing for hours just talking about the elegantly evil architecture of these attacks, but we are on the clock here, and it is time to stop these attacks cold. And here's the cool solution: You already know the solution.

Everything we've been talking about over this entire course is not only good architecture, but it also helps solve almost all DDoS attack vectors with zero additional effort or cost. First attack, the low level network attacks like the UDP floods. Solution, security groups. The security groups only allow in proper request traffic. Things like weather reports use an entirely different protocol than the ones your customers use. Not on the list, you don't get to talk to the server. And what's more, security groups operate at the AWS network level, not at the EC2 instance level, like an operating system firewall might.

So massive attacks like UDP floods or reflection attacks just get shrugged off by the scale of the entire AWS Regions capacity, not your individual EC2's capacity. This is a case where our size is a huge advantage in your protection. I won't say it's impossible to overwhelm AWS, but the scale it would take, it would be too expensive for these bad actors. Slowloris attacks? Look at our elastic load balancer. Because the ELB handles the http traffic request first, so it waits until the entire message, no matter how fast or slow, is complete before sending it over to the front end web server. I mean, sure,

you can try to overwhelm it, but remember how the ELB is scalable and how it runs at the region level?

To overwhelm ELB, you would once again have to overwhelm the entire AWS region. It's not theoretically impossible, but too massively expensive for anyone to pull off. For the sharpest, most sophisticated attacks, AWS also offers specialized defense 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.

All right, that's it, the clock is almost up. The takeaway is a well-architected system is already defended against most attacks. And by using AWS Shield Advanced, you can turn AWS into your partner against DDoS attacks. Oh, oh.

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**.

Denial-of-service attacks

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

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.

To help minimize the effect of DoS and DDoS attacks on your applications, you can use **AWS Shield**(opens in a new tab).

AWS Shield

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

To learn more about AWS Shield, expand each of the following two categories.

AWS Shield Standard

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 to detect malicious traffic in real time and automatically mitigates it.

AWS Shield Advanced

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.

Additional Security Services

To start the video on additional security features, choose the play button.

Play Video

To access a transcript of the video, choose the following transcript.

Video transcript

With all the comings and goings on in your coffee shop, you'll want to increase security to your coffee beans, equipment, and even money in the till. For your beans, this could be when they're sitting in your store room, or even when you're transporting them between shops. After all, we don't want unwanted visitors with access to our coffee beans, or even running off with precious equipment.

To start off, let's chat about how you can secure your coffee beans, or your data whether it's at rest, or in transit. For our beans, the simple way to do it would be to lock the door when we'd leave at night. That's the notion of encryption, which is securing a message or data in a way that can only be accessed by authorized parties. Non-authorized parties are therefore less likely to be able to access the message. Or not able to access it at all. Think of it as that key and door example. If you have the key, you can unlock the door. But if you don't, then you cannot unlock that door.

At AWS, this comes in two variations. Encryption at rest and encryption in transit. By at rest, we mean when your data is idle. It's just being stored and not moving. 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. So make sure to keep it safe.

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.

But speaking of other services, the next service we want to highlight is called Amazon Inspector. Inspector helps to improve security, and compliance of your AWS deployed applications by running an automated security assessment against your infrastructure. Specifically, it helps to check on deviations of security best practices, exposure of EC2 instances, vulnerabilities, and so forth. The service consists of three parts a network configuration reachability piece, an Amazon agent, which can be installed an EC2 instances, and a security assessment service that brings them all together. To use it, you configure Inspector options, run the service, out pops a list of potential security issues. The resulting findings are displayed in the Amazon Inspector console, and they are presented with a detailed description of the security issue, and a recommendation on how to fix it. Additionally, you can retrieve findings through an API. So as to go towards the best practice of performing remediation to fix issues.

Another service dimension is our threat detection offering known as Amazon GuardDuty. It analyzes continuous streams of metadata generated from your account, and network activity found on AWS CloudTrail events, Amazon 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.

They are so many other security services like Advanced Shield and Security Hub. So please check out the Resources section to learn more. Thanks for following along.

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**.

AWS Key Management Service (AWS KMS)(opens in a new tab) 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.

AWS WAF

AWS WAF(opens in a new tab) 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)**(opens in a new tab) 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 does not come from one of the blocked IP addresses, it allows access to the application.

However, if a request comes from one of the blocked IP addresses that you have specified in the web ACL, AWS WAF denies 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**(opens in a new tab).

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(opens in a new tab) 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 behavior 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.

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.