Series Agenda

Part 1

- Introduction: Course Overview
- Module 1: Introduction to Amazon Web Services
- Module 2: Global Infrastructure and Reliability
- Module 3: Networking
- Module 4: Object Storage

Part 2

- Module 5: Security
- Module 6: Block and File Storage
- Module 7: Compute in the Cloud

Part 3

- Module 8: AWS Frameworks, Pricing, and Support
- Module 9: Applications in the Cloud
- Module 10: Databases
- Module 11: Monitoring and Analytics

Setup and books: One-time steps

Resources you will need during this course:

Demonstrations

- No setup is required. You will be able to watch as the instructor demonstrates.

Digital Books

- Will be emailed to you at the end of class.
- Based on attendance.

Course slides and notes can be accessed following the class.
Interactive Demos are performed using Skill Builder's AWS Labs.

This course offers many interactive activities

Instructor demonstrations

- You observe as the instructor demonstrates parts of the AWS console and features related to your discussion.

Knowledge Checks

- You answer questions to reinforce what you have learned over the course of the module.

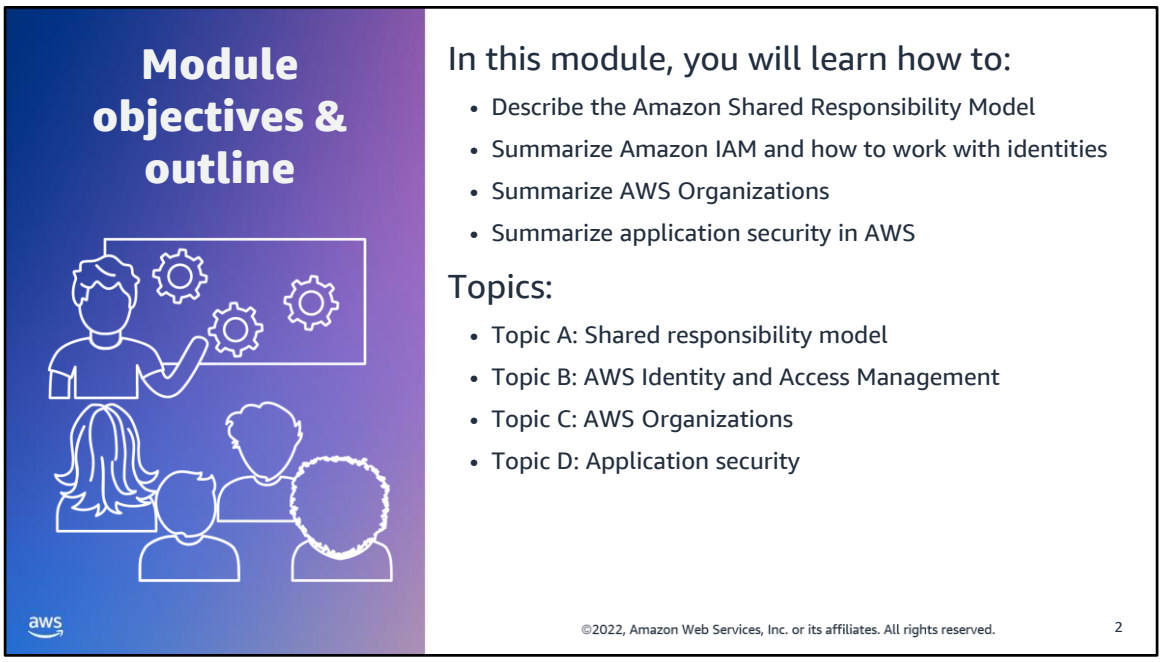
In this module, you will learn about how to secure you services and applications using AWS services and features.

Module Objectives:

- Describe the Amazon Shared Responsibility Model
- Summarize Amazon IAM and how to work with identities
- Summarize AWS Organizations
- Summarize application security in AWS

Topics:

- Topic A: Shared responsibility model
- Topic B: AWS Identity and Access Management
- Topic C: AWS Organizations
- Topic D: Application security



In this module, you will learn about how to secure you services and applications using AWS services and features.

Module Objectives:

- Describe the Amazon Shared Responsibility Model
- Summarize Amazon IAM and how to work with identities
- Summarize AWS Organizations
- Summarize application security in AWS

Topics:

- Topic A: Shared responsibility model
- Topic B: AWS Identity and Access Management
- Topic C: AWS Organizations
- Topic D: Application security

Acronyms covered in this module

Multi-factor Authentication

An authentication method that requires a user to provide two methods of identification.

- Text codes, email codes, USB devices

Web Application Firewall

A service that filters and monitors HTTP traffic between your application and the internet.

Terminology covered in this module (1 of 3)

Encryption

A process of replacing plain text with text created using a secret code that only you have the key to decipher.

Server-side encryption

Server-side encryption is the **encryption of data at its destination** by the application or service that receives it.

Client-side

Client-side encryption is the **encryption of data at its source** by the application or service that receives it.

aws

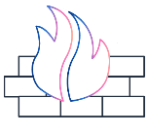
Terminology covered in this module (2 of 3)

Edge location

A site that CloudFront uses to cache copies of your content for faster delivery to users at any location.

Entity

An individual (person), organization, device or process.



Firewall

A Firewall is a network security device that monitors and filters incoming and outgoing network traffic based on a network.

- Packet filtering, network, and application.

aws

Terminology covered in this module (3 of 3)

Authentication

The process of determining that a connection was created by who or what they claim to be.

Who you are

Authorization

The process of granting permission to an authenticated entity.

What you can/can't do

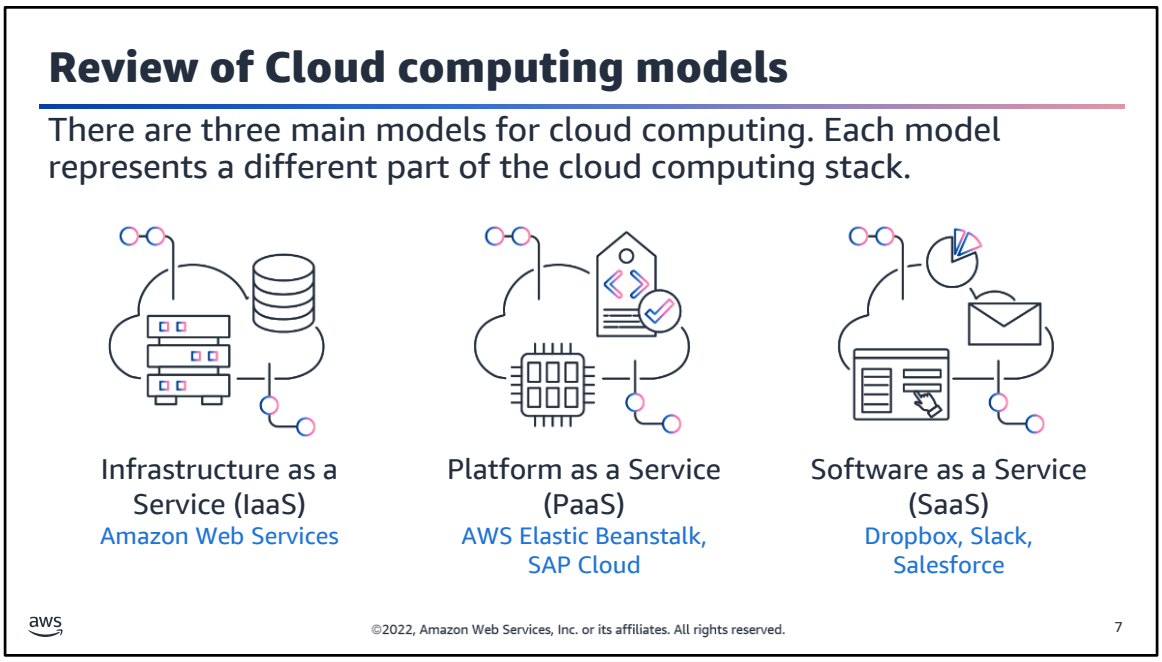
Credentials

Something that an entity has to prove their identity.

- Username and password, security key, or one-time use passcode.

What you have

aws

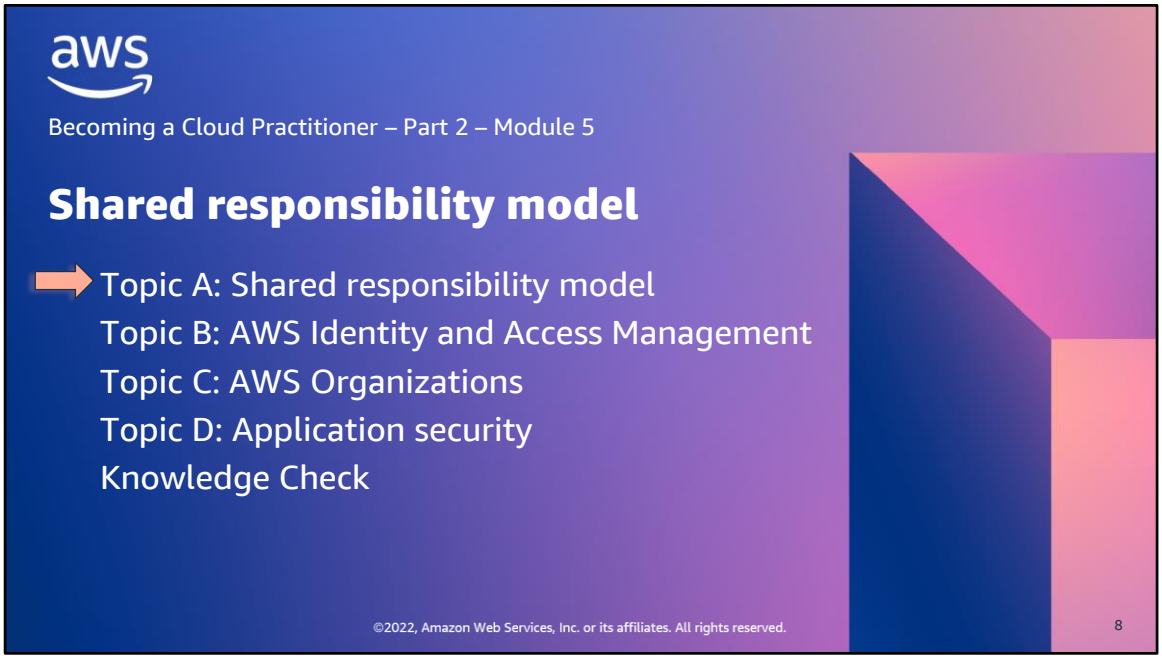


When selecting cloud services, there are three common phrases used; Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).

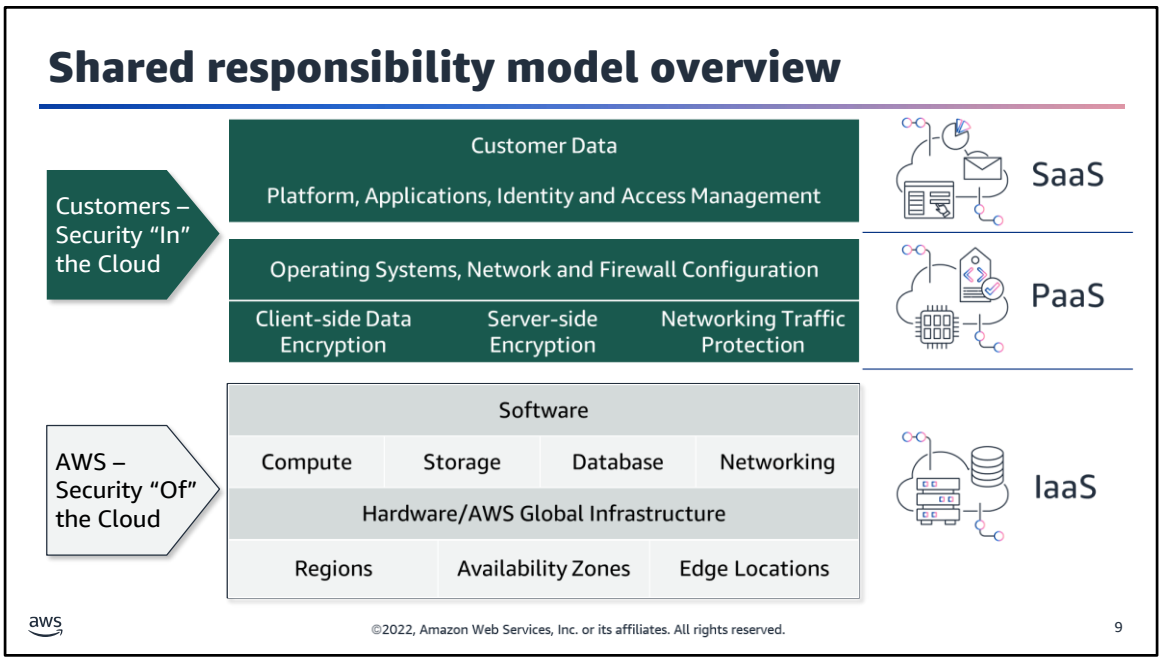
Infrastructure as a Service (IaaS) contains the basic building blocks for cloud IT and typically provide access to networking features, computers (virtual or on dedicated hardware), and data storage space. Infrastructure as a Service provides you with the highest level of flexibility and management control over your IT resources and is most similar to existing IT resources that many IT departments and developers are familiar with today. IaaS includes networking, computing, and data storage components that you manage.

Platform as a Service (PaaS) removes the need for organizations to manage the underlying infrastructure (usually hardware and operating systems) and allow you to focus on the deployment and management of your applications. This helps you be more efficient as you don't need to worry about resource procurement, capacity planning, software maintenance, patching, or any of the other undifferentiated heavy lifting involved in running your application. PaaS solutions are fully managed. You are able to focus on the applications and data without concern over complex networking and compute.

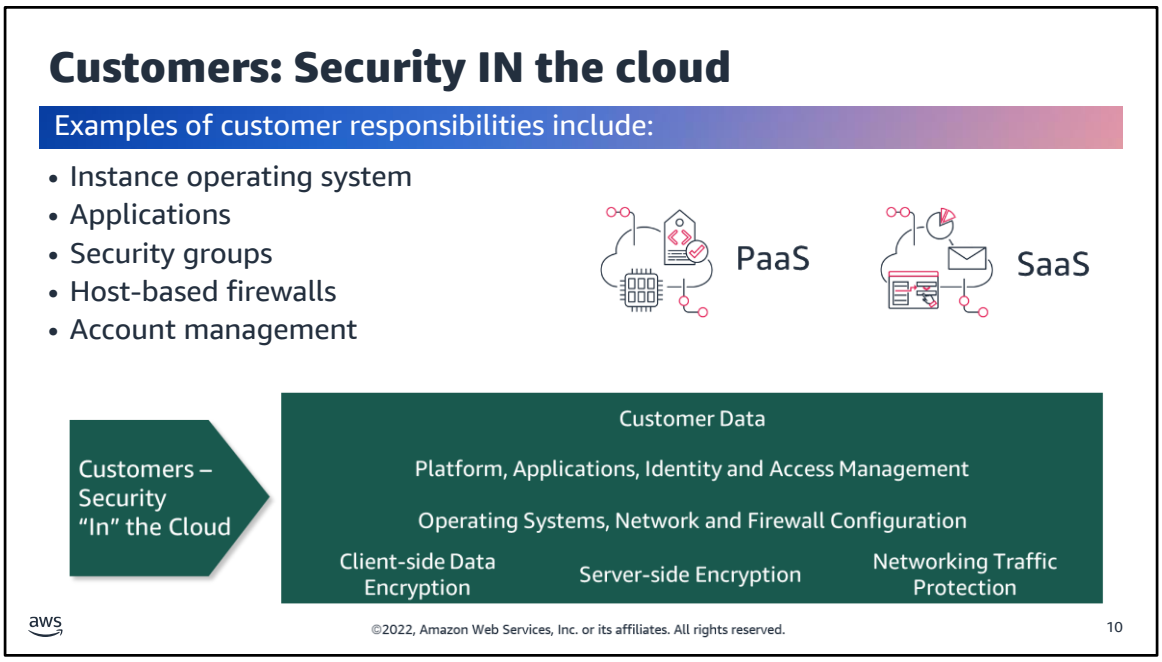
Software as a Service (SaaS) provides you with a completed product that is run and managed by the service provider. In most cases, people referring to Software as a Service are referring to end-user applications. With a SaaS offering you do not have to think about how the service is maintained or how the underlying infrastructure is managed; you only need to think about how you will use that particular piece of software. A common example of a SaaS application is web-based email where you can send and receive email without having to manage feature additions to the email product or maintaining the servers and operating systems that the email program is running on.



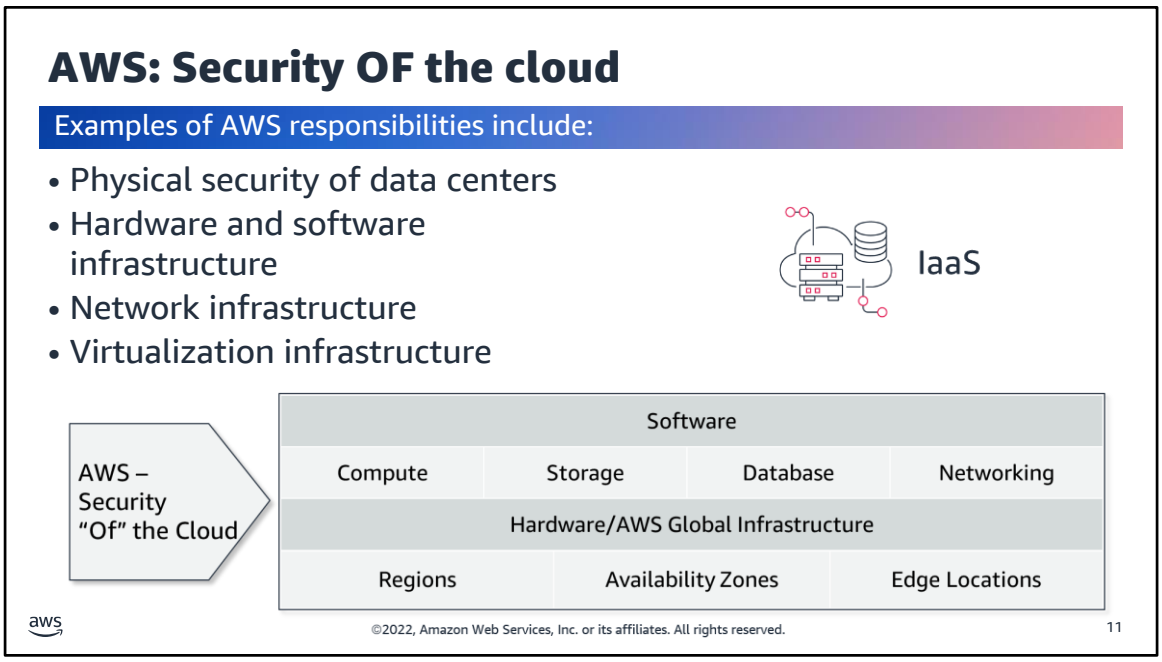
In this topic, you will learn about the AWS Shared responsibility model.



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. Relating this back to the Cloud computing model, this is defining an Infrastructure as a Service (IaaS) solution.
As the homeowner (the customer), it is your responsibility to secure everything in the house by ensuring that the doors are closed and locked. You move in your own furniture and belongings. Relating this back to the Cloud computing model, this is defining an Platform as a Service (PaaS) solution.
There are times when you will vacation, or you want a fully furnished home. In this case, the rental company handles all of the furnishings. You as the tenant are only responsible to bring in your personal belonging. Relating this back to the Cloud computing model, this is defining an Software as a Service (SaaS) solution.
This section examines the shared responsibility model in greater detail, beginning with customers' responsibilities.



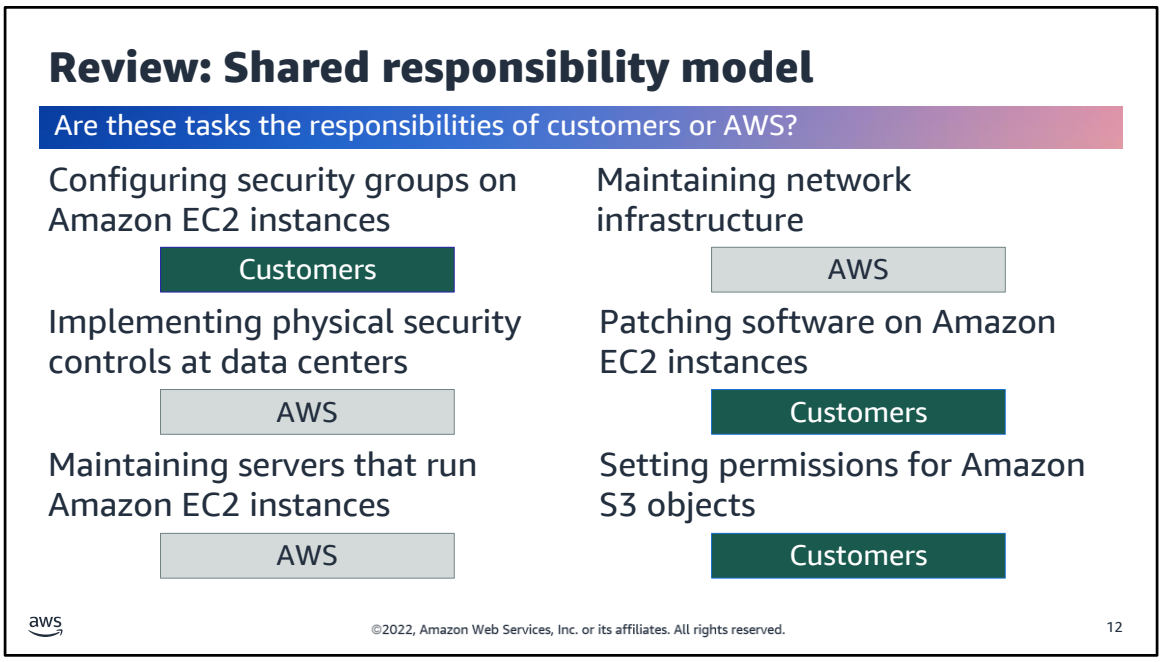
Customers are responsible for the security of everything that they create and put *in* the AWS Cloud.
When using AWS services, you, the customer, maintain complete control over your content. You are responsible for managing security requirements for your content, including which content you choose to 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.



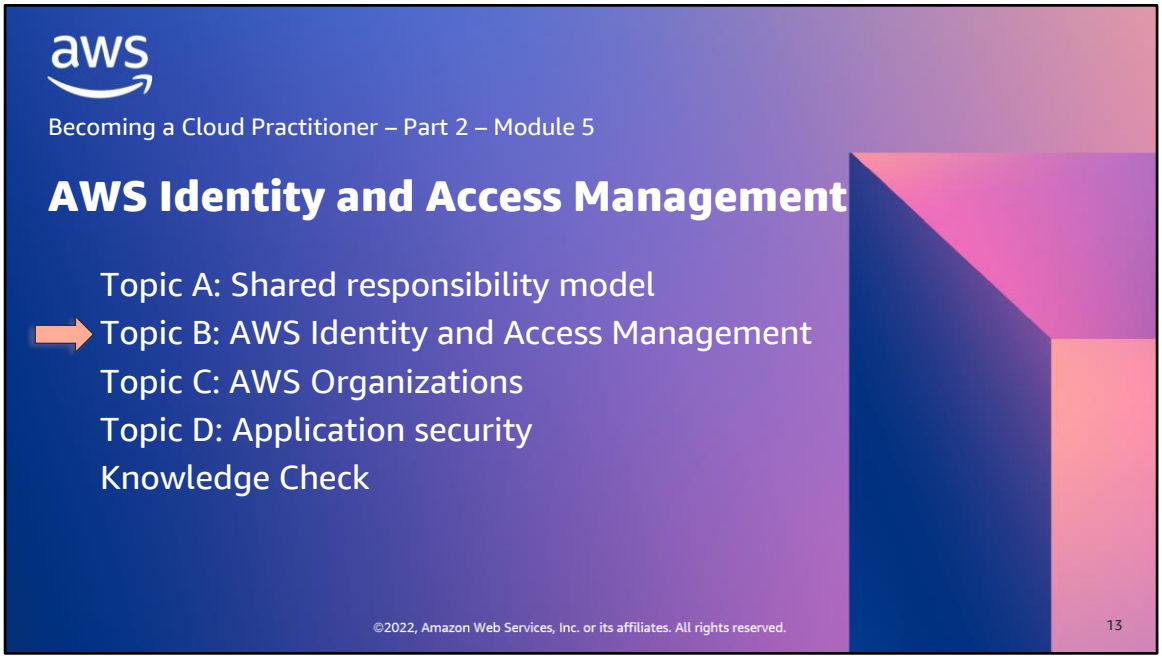
AWS is responsible for security *of* the cloud.
AWS operates, manages, and controls the components at all layers of infrastructure. This includes areas such as the host operating system, the virtualization layer, and even the physical security of the data centers from which services operate.
AWS is responsible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



Suppose that you are the owner of the coffee shop. Each item on this slide represents an aspect of running the coffee shop's website. For each item, determine if it is your (the customer's) responsibility or an AWS responsibility.
Items that are filled in with grey are the responsibility of AWS; those that are filled in with green are the responsibility of the customer.



In this topic, you will learn about Identity and Access Management (IAM) and how to create, manage, and maintain users, roles, and security policies.

Authentication and authorization at the airport

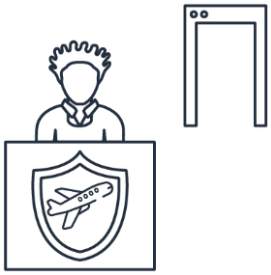
The first stop you make is at the check-in counter.

Do you have a ticket?

Authorization

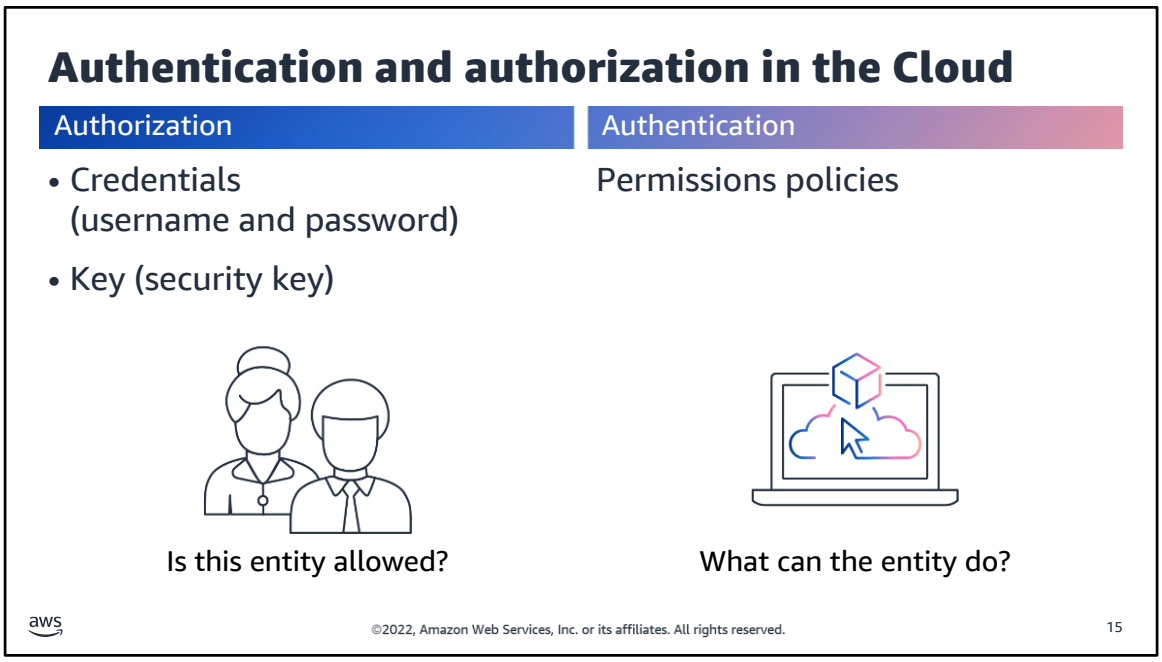
Next you must go through security.

Are you cleared to travel?

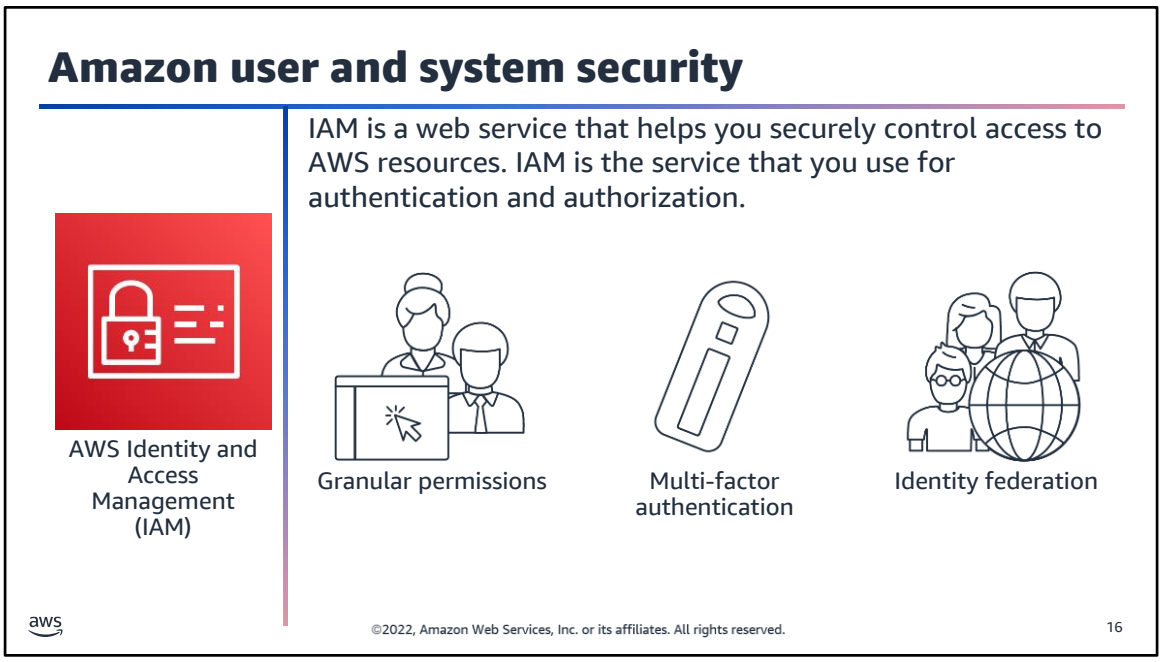


Authentication

Time for a new example. In this example, you are going to the airport. The first stop you make is at the check in counter. Here they check your id and make sure that you have a ticket for a flight. When they check for a ticket, they are authorizing you to fly.
Next you must go through security. Here you must show your ID or passport to go any farther in the airport. They are authenticating that you look like the picture on the ID or passport to ensure that you are who you say you are.



Authentication is the process of collecting identity information and comparing it to a database of approved identities. An identity is comprised of a user ID and credentials or security keys. Credentials are often made up of a username and password.
When your AWS account is authenticated, you will be able to access the AWS Management console.
Without being authorized, you will not be able to see any of the resources in any AWS service console. You must now be authorized.
Authorization is the process of assigning permissions to a given entity. Permissions define the access an IAM entity (user, group, or service) has to a service or feature.
For example, you can give a user access to Amazon S3. Until you do, they are not able to see any of the buckets within Amazon S3. This is called View permission. If you would like this user to be able to download and edit items in the Amazon S3 bucket, you must specifically give them Modify permissions on that specific bucket.



AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources.

Granular permissions

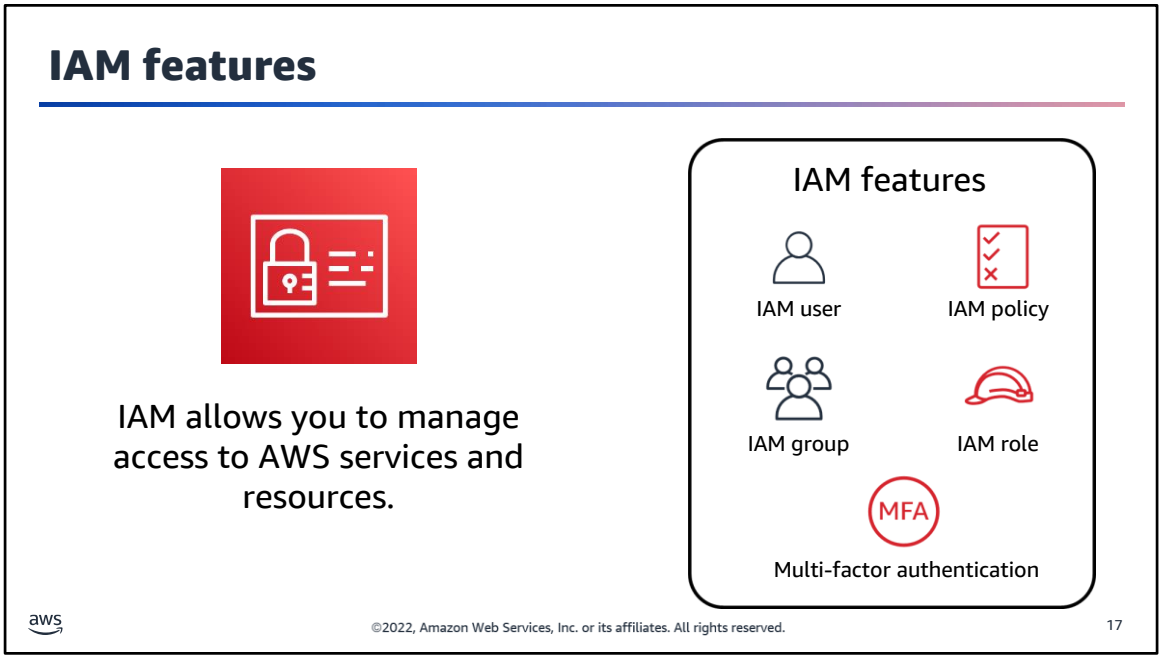
You can grant different permissions to different entities for different resources. For example, you might allow some users complete access to Amazon EC2 and Amazon S3. Other users may also need access to Amazon DynamoDB and other AWS services.

Multi-factor authentication (MFA)

You can add two-factor authentication to your account and to individual users for extra security. With MFA you or your users must provide not only a password or access key to work with your account, but also a code from a specially configured device.

Identity federation

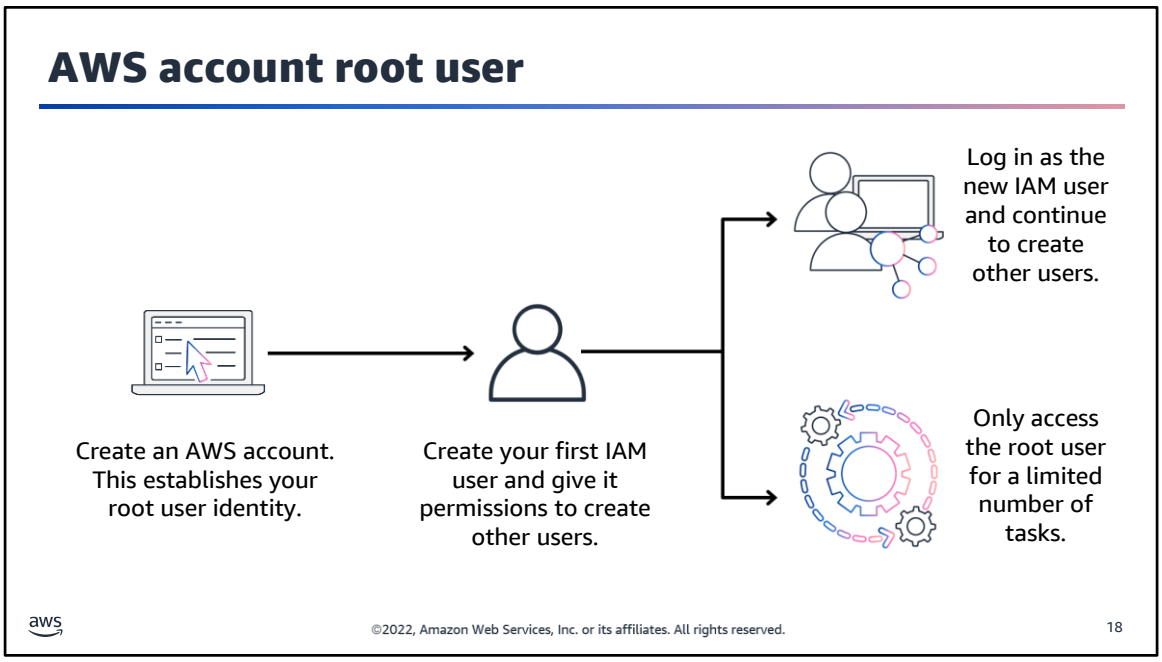
You can allow users who already have passwords elsewhere—for example, in your corporate network or with an internet identity provider—to get temporary access to your AWS account.



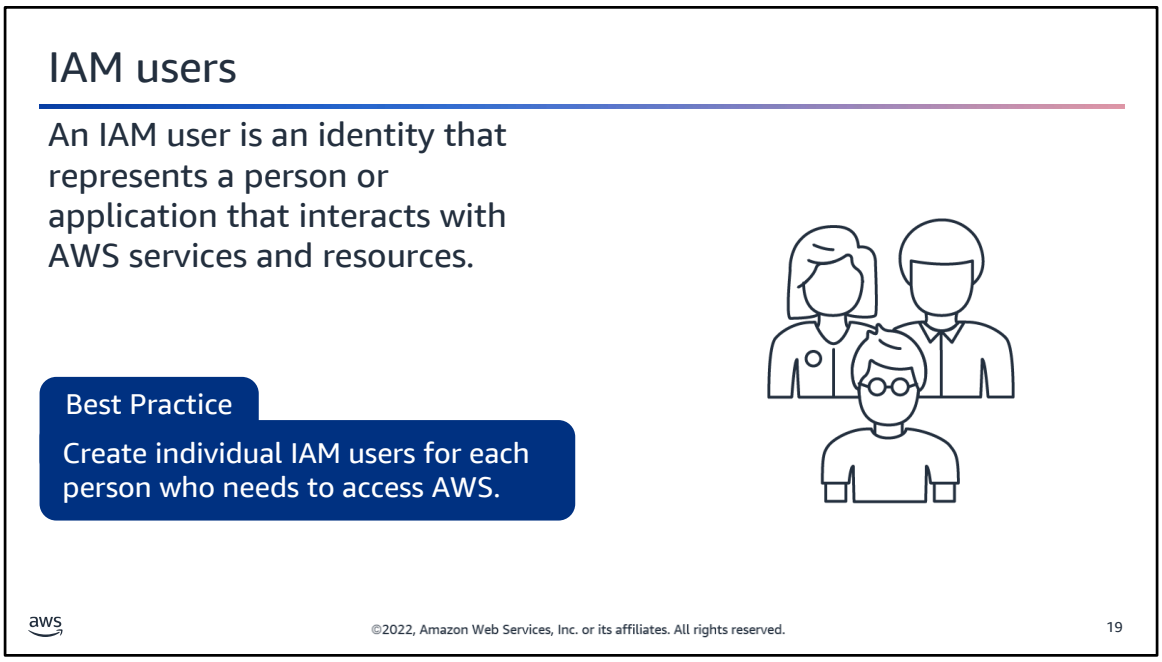
IAM gives you the flexibility to configure access based on your company's specific operational and security needs. You do this by using a combination of IAM features, which are explored in detail in this lesson:

- IAM users, groups, and roles
- IAM policies
- Multi-factor authentication

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



When you first create an AWS account, you begin with an identity that is 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 of the AWS services and resources within the account.
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.

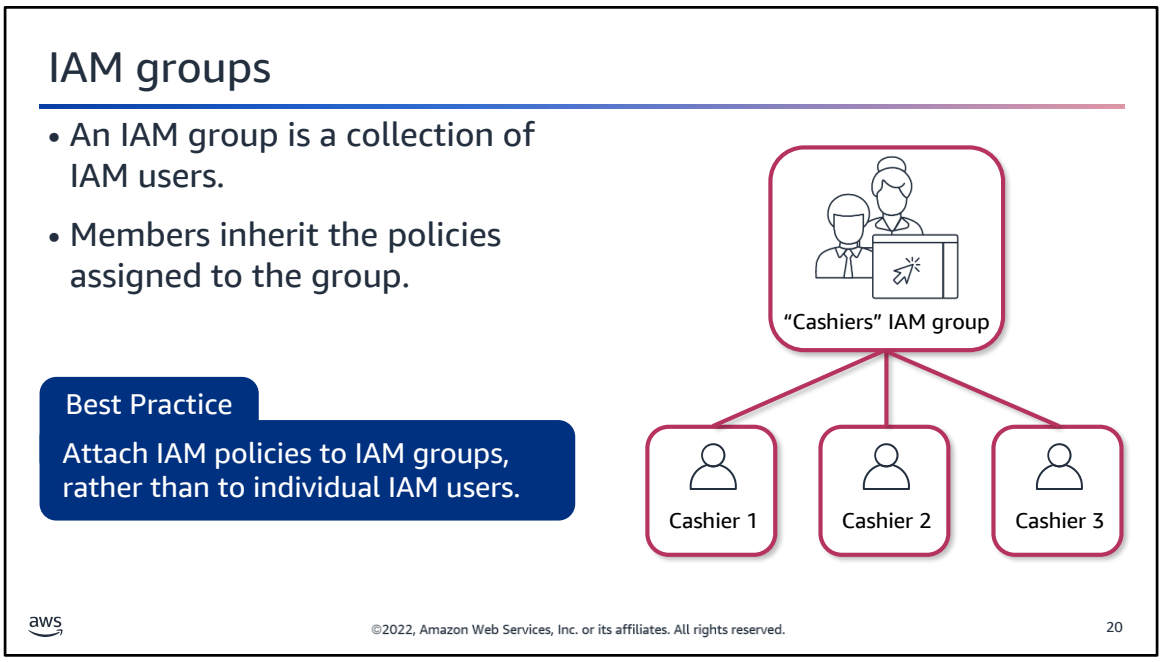


An **IAM user** is an identity that you create in AWS. It represents the person or application that interacts with AWS services and resources. It consists of a name and credentials.

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.

As a best practice, create individual IAM users for each person who must 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.



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.

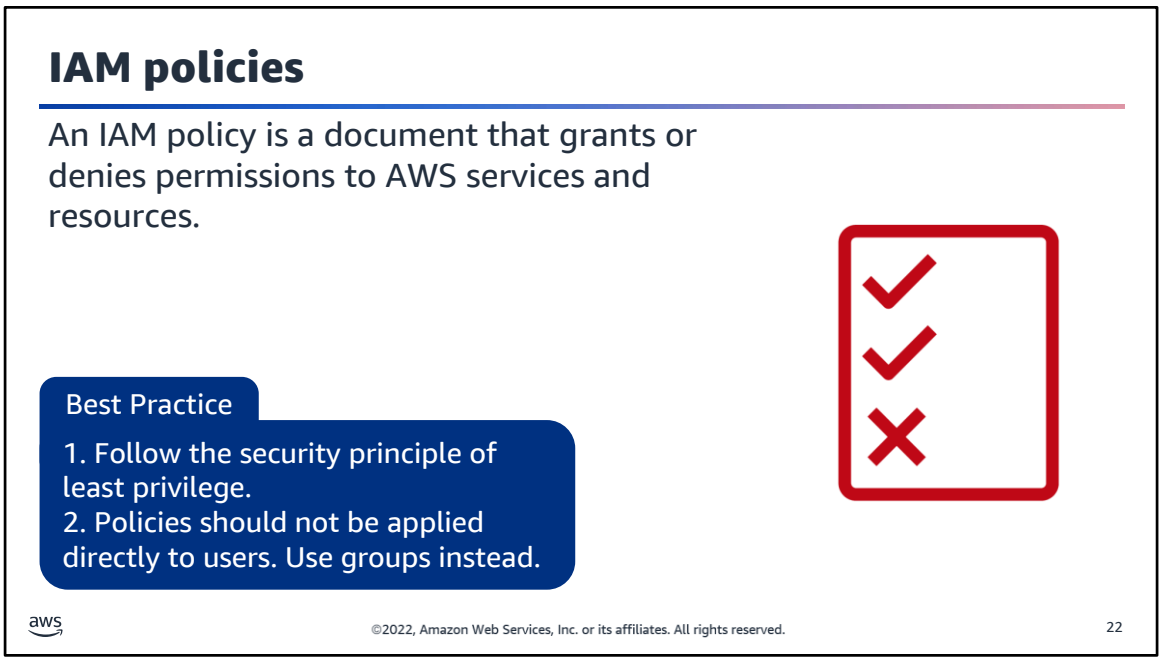
What if a coffee shop employee hasn't switched jobs permanently, but instead, rotates to different workstations throughout the day? This employee can get the access they need through **IAM roles**.

Interactive Demonstration 4

Working with IAM Users

In this demonstration you will:

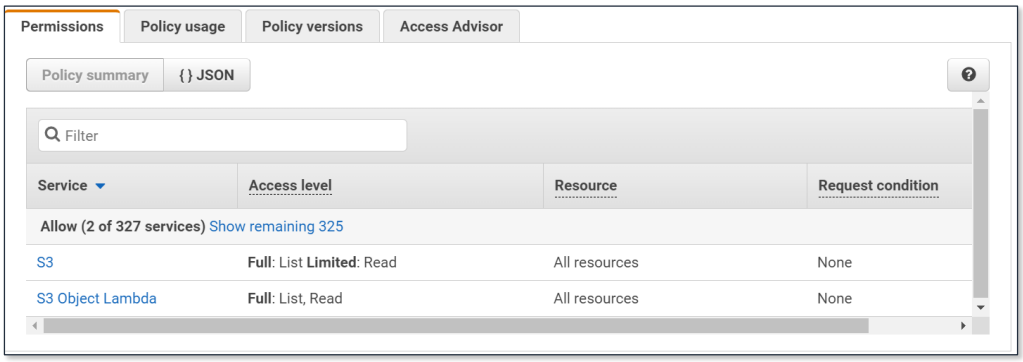
- Investigate the AWS IAM dashboard
- Add an IAM user to a user group



An **IAM policy** is a document that allows or denies permissions to AWS services and resources.
IAM policies allow 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.
As a 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 (1 of 2)

This sample IAM policy allows permission to view a list of objects in the Amazon S3 bucket with ID awsdoc-example-bucket, and also access them. This is the visual Policy summary.



Here's an example of how IAM policies work. Suppose that the coffee shop owner has created an IAM user for a newly hired cashier. The cashier needs access to the receipts that are kept in an Amazon S3 bucket with the ID: *awsdoc-example-bucket*.

The coffee shop owner uses the following IAM policy to grant the cashier access to the *awsdoc-example-bucket* bucket in Amazon S3.

In this example, you are seeing a summary of a policy that allows list and limited read permissions on S3 and list and read permissions on S3 Object Lambda. When the owner attaches this policy to the cashier's IAM user, it will allow the cashier to view a list of the objects in the *awsdoc-example-bucket* bucket and also access them.

Example: IAM policy (2 of 2)

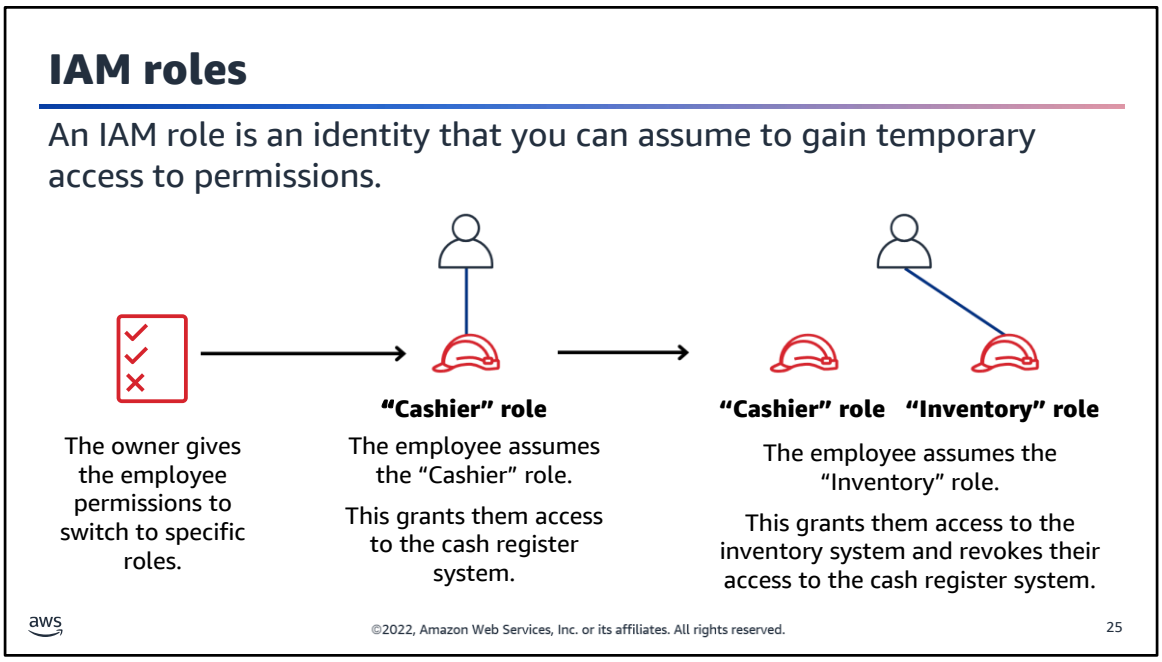
This is the same IAM policy however you are seeing the JSON template.

```
{  "Version": "2012-10-17",
   "Statement": [
     {
     "Effect": "Allow",
     "Action": [
         "s3:Get*",
         "s3:List*",
         "s3-object-lambda:Get*",
         "s3-object-lambda:List*“
     ],
     "Resource": "awsdoc-example-bucket“
     }
   ]
}
```

aws

24

You can see in the JSON that the IAM policy is allowing specific actions within Amazon S3: ListObject and GetObject. The policy also mentions a specific bucket ID: *awsdoc-example-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**.



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. (**Note:** When an IAM user is assuming a role, they remain logged into the AWS Management Console through their IAM user account and can switch back to their IAM user permissions at any time.)

IAM roles are the preferred method for interacting with AWS services. As a best practice, IAM roles are ideal for situations in which access to services or resources needs to be granted temporarily, rather than long term.

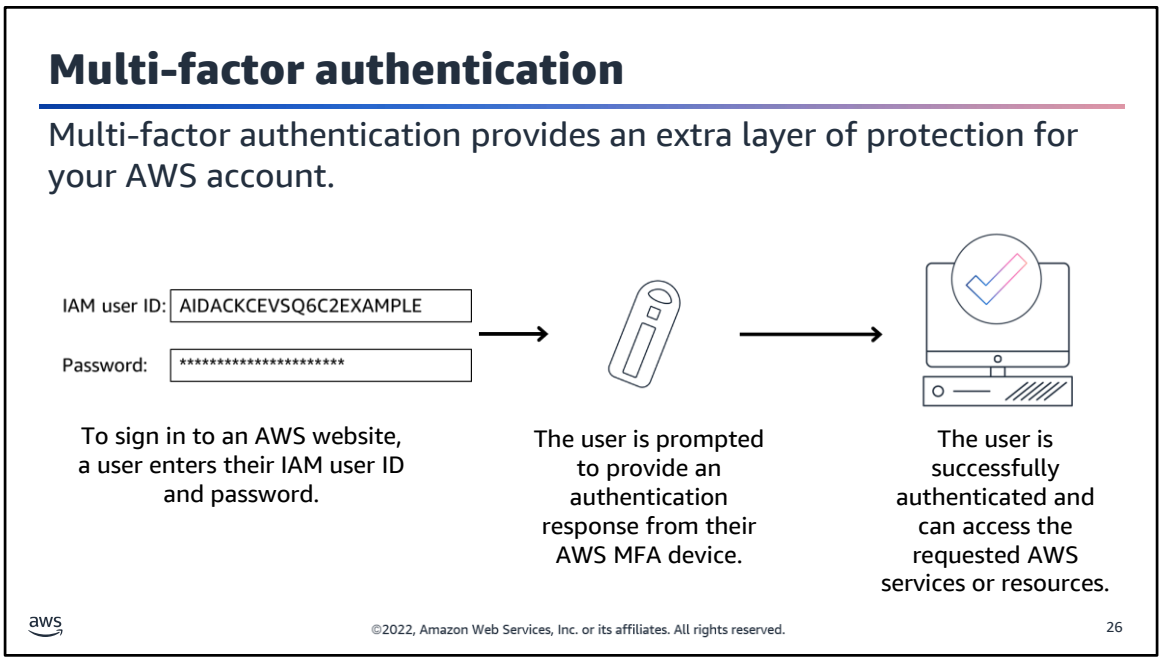
Consider this example of how IAM roles could be used in the coffee shop.

First, the owner gives the employee permissions to switch to a role for each workstation in the coffee shop.

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

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.

The final aspect of IAM that we will examine is **multi-factor authentication**.



Have you ever logged into 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.

First, when a user signs into an AWS website, they enter their IAM user ID and password.

Next, the user is prompted for an authentication response from their AWS MFA device. This device could be a hardware security key, a hardware device, or an MFA application on a device such as a smartphone.

When the user has been successfully authenticated, they are able to access the requested AWS services or resources.

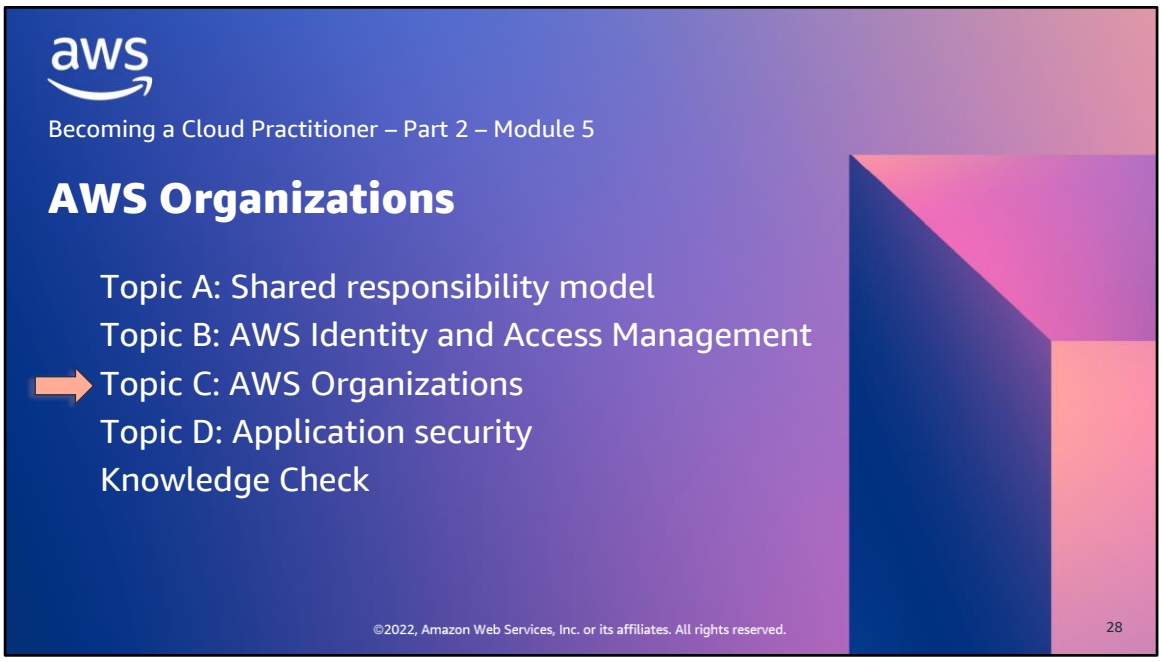
MFA can be enabled for the root user and IAM users. As a best practice, you should enable MFA for the root user and all IAM users in your account, since this will help to keep your AWS account safe from unauthorized access.

Interactive Demonstration 5

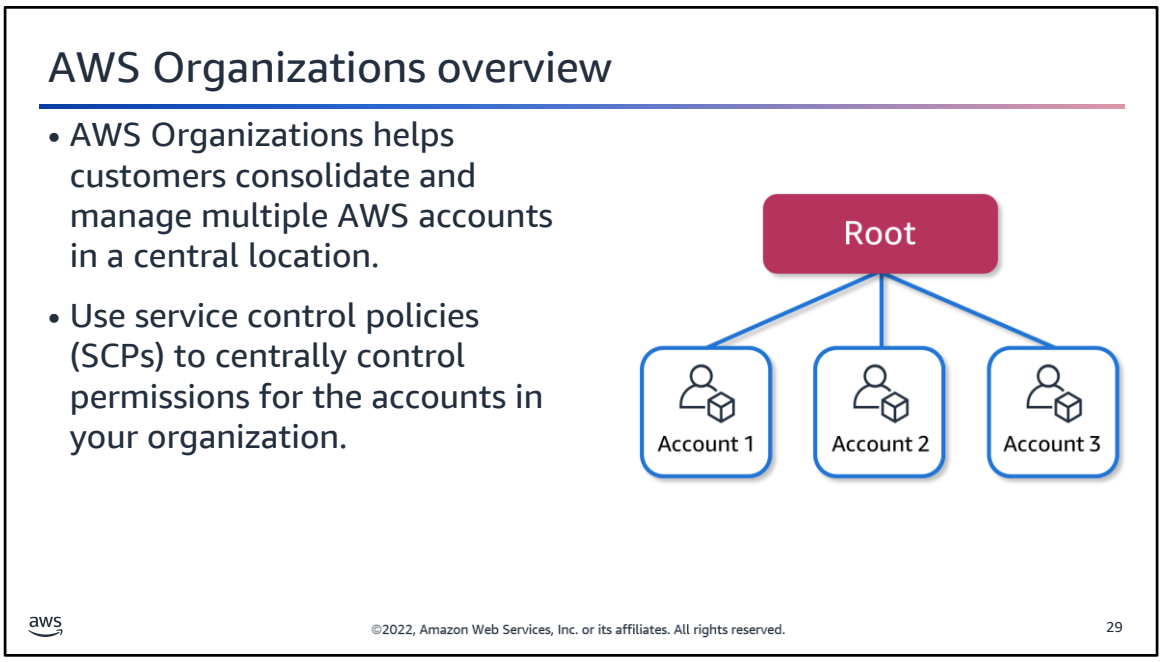
Working with IAM groups and policies

In this demo you will:

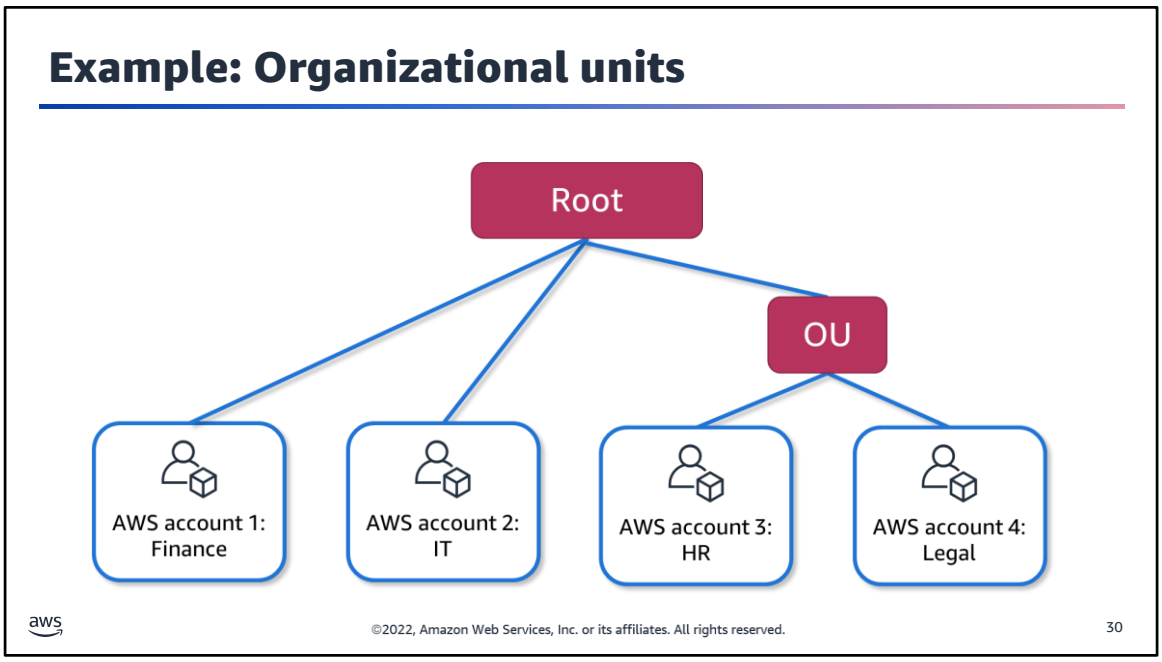
- Investigate the AWS IAM dashboard
- Create an IAM group
- Assign an IAM policy



In this topic, you will learn about AWS Organizations that helps companies to manage multiple AWS accounts in an efficient way.



AWS Organizations helps you 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)**. SCPs help you place restrictions on the AWS services, resources, and individual API actions that the users and roles in each account can access.
You can also apply SCPs to the root. For example, you might apply an SCP that requires that multi-factor authentication (MFA) is enabled before an IAM user or role can perform specific actions, such as stopping or terminating an Amazon EC2 instance.
(**Note:** Consolidated billing is another feature of AWS Organizations. You will learn about consolidated billing in a later module.)



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.

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.

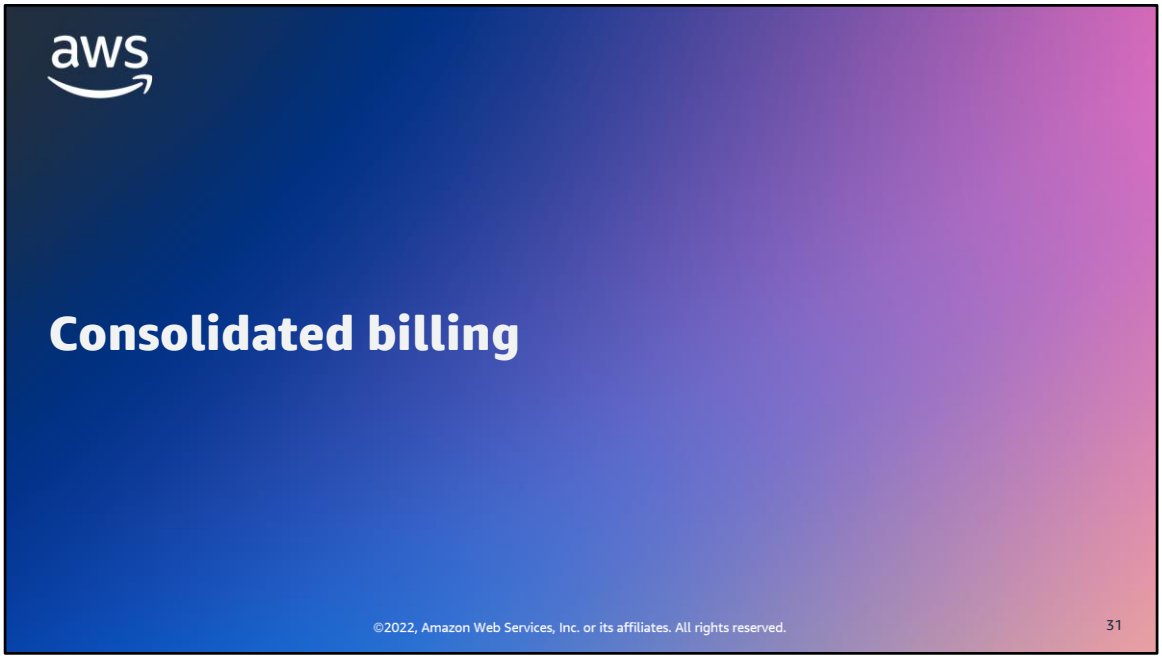
In designing your organization, you consider the business, security, and regulatory needs of each department. You use this information to decide which departments group together in OUs.

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.

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 allows 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.

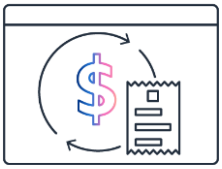
By grouping your accounts into OUs, you can more easily give them access to the services and resources that they need. You also prevent them from accessing any services or resources that they do not need.



In an earlier module, you learned about AWS Organizations, a service that you can use to manage multiple AWS accounts from a central location. AWS Organizations also provides the option for **consolidated billing**.

Consolidated billing overview

A feature of AWS Organizations allows you to receive a single bill for all AWS accounts in your organization



Receive a single bill for all the AWS accounts in your organization

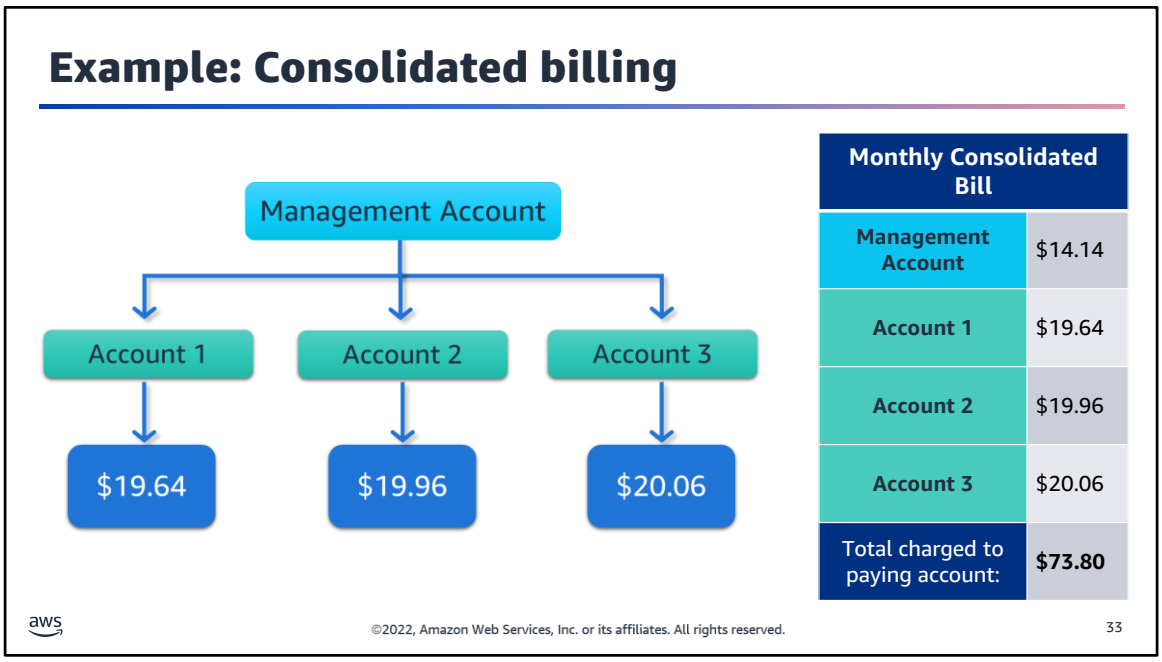
Review itemized charges that have been incurred by each account

Share savings across the accounts in your organization

The **consolidated billing** feature of AWS Organizations allows you to receive a single bill for all AWS accounts in your organization. By consolidating, you can track the combined costs of all the linked accounts in your organization. The default maximum number of accounts allowed for an organization is four, but you can contact AWS Support to increase your quota, if needed.

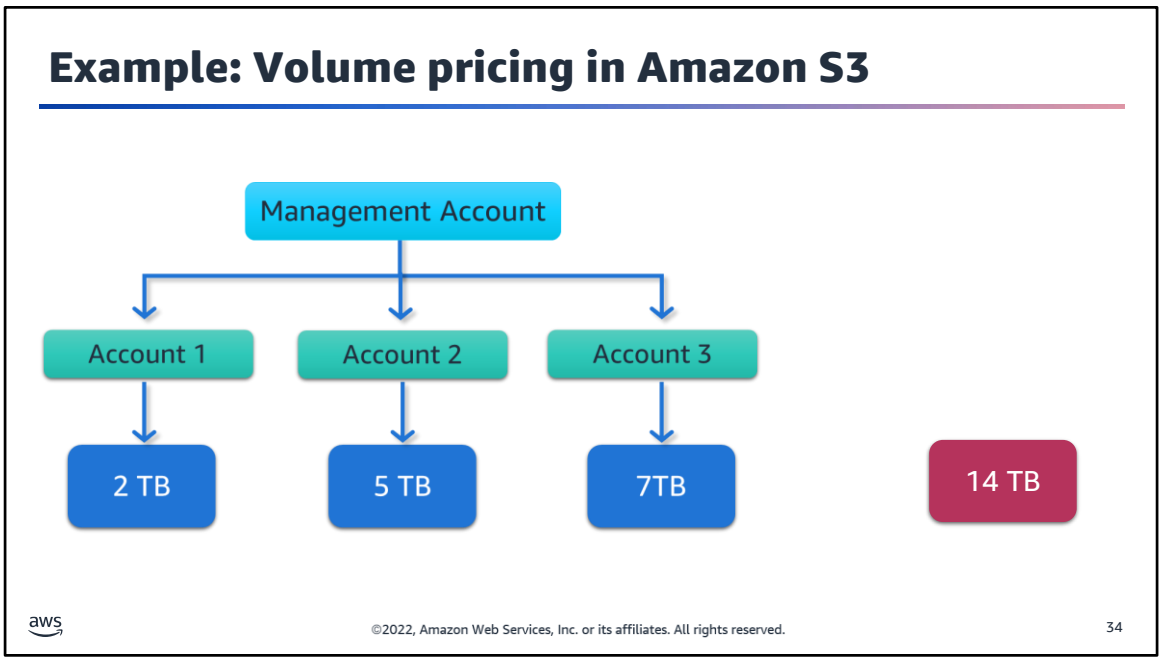
On your monthly bill, you can review itemized charges incurred by each account. This provides transparency into your organization's accounts while maintaining the convenience of receiving a single monthly bill.

Another benefit of consolidated billing is the ability to share bulk discount pricing, Savings Plans, and Reserved Instances across the accounts in your organization. For instance, one account might not have enough monthly usage to qualify for discount pricing. However, when multiple accounts are combined, their aggregated usage might result in a benefit that applies across all accounts in the organization.



Suppose that you are the business leader who oversees your company's AWS billing.
Your company has three AWS accounts used for separate departments. Instead of paying each location's monthly bill separately, you decide to create an organization and add the three accounts.
You manage the organization through the management account.
Each month, AWS charges your management payer account for all the linked accounts in a consolidated bill.
Through the management account, you can also get a detailed cost report for each linked account.
The monthly consolidated bill also includes the account usage costs incurred by the management account. This cost is not a premium charge for having a management account.
The consolidated bill shows the costs associated with any actions of the management account (such as storing files in Amazon S3 or running Amazon EC2 instances).
Fun trivia note: The dollar values on this slide correspond to significant years in Amazon's history:

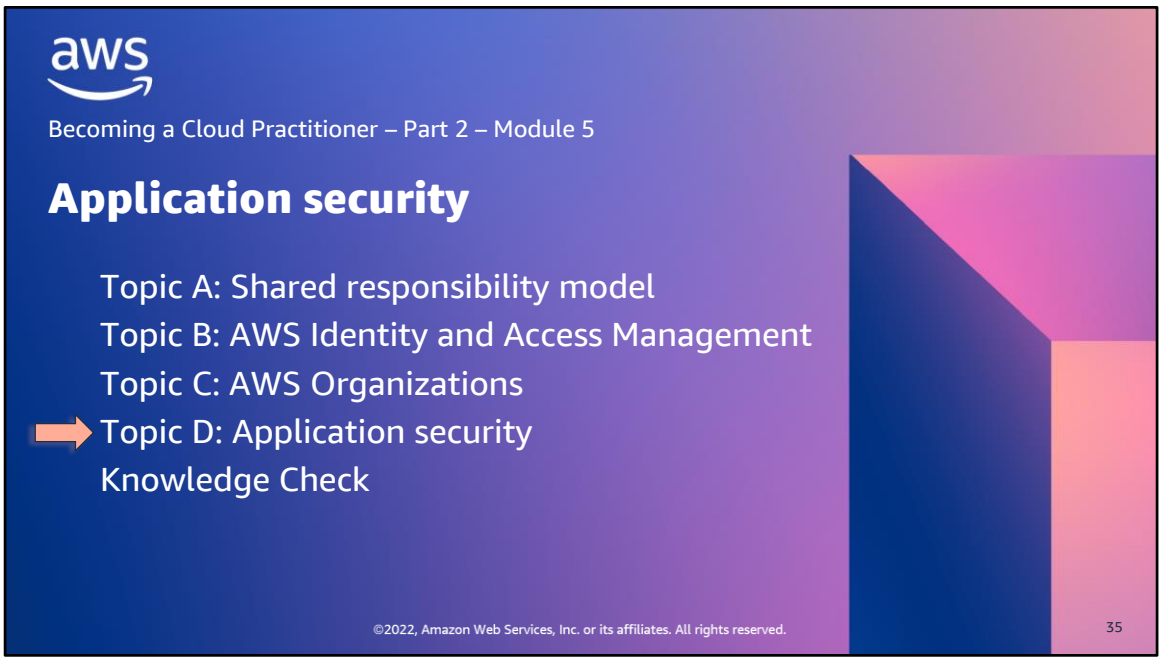
- *1964 is the year in which Jeff Bezos was born.*
- *1996 is the year in which Amazon was founded.*
- *2006 is the year in which AWS was founded.*



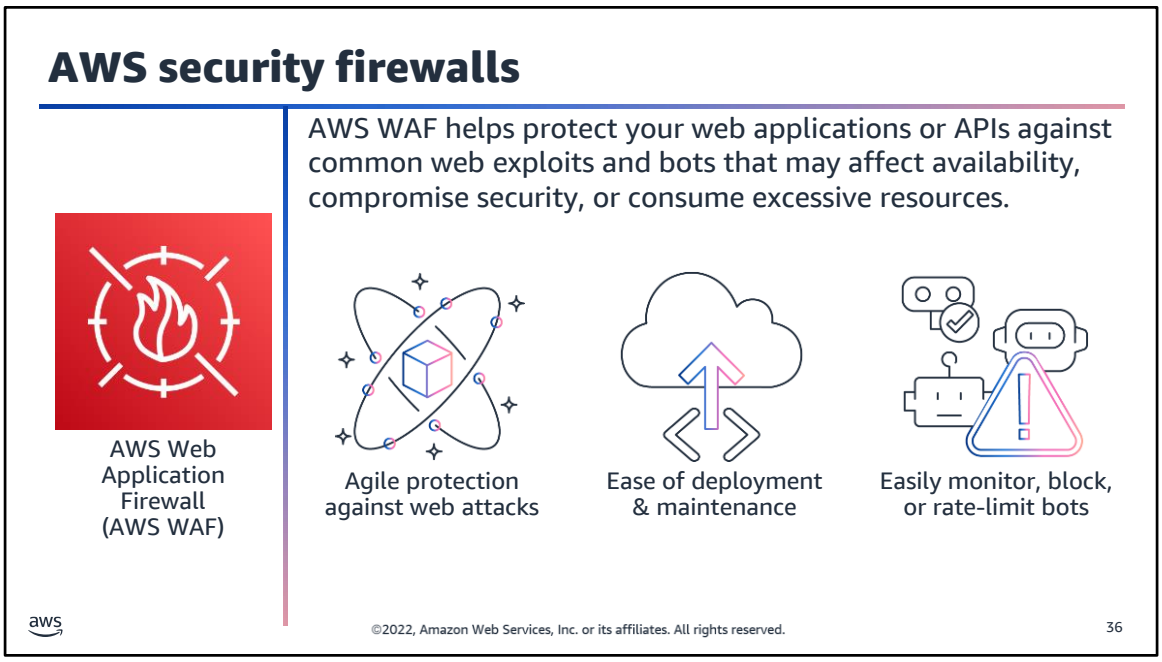
Consolidated billing also lets you share volume pricing discounts across accounts.
Some AWS services, such as Amazon S3, provide volume pricing discounts that give you lower prices the more that you use the service. In Amazon S3, after customers transfer 10 TB of data in a month, they pay a lower per-GB transfer price for the next 40 TB of data transferred.
In this example, three separate AWS accounts transferred different amounts of data in Amazon S3 during the current month:

- Account 1 transferred 2 TB of data.
- Account 2 transferred 5 TB of data.
- Account 3 transferred 7 TB of data.

Because no single account passed the 10 TB threshold, none of them is eligible for the lower per-GB transfer price for the next 40 TB of data transferred.
Now, suppose that the three accounts are linked in a single AWS organization and use consolidated billing. When the Amazon S3 usage for the three linked accounts is combined (2+5+7), it results in a combined data transfer amount of 14 TB. This exceeds the 10-TB threshold.
With consolidated billing, AWS combines the usage from all accounts to determine which volume pricing tiers to apply, giving customers a lower overall price whenever possible. AWS then allocates each linked account a portion of the overall volume discount based on the account's usage.
In this example, Account 3 would receive a greater portion of the overall volume discount because at 7 TB, it transferred more data than Account 1 (at 2 TB) and Account 2 (at 5 TB).



In this topic, you will learn about AWS services that will help you to secure applications you build in the cloud.



AWS Web Application Firewall (AWS WAF) helps protect your web applications or APIs against common web exploits and bots that may affect availability, compromise security, or consume excessive resources.

Agile protection against web attacks

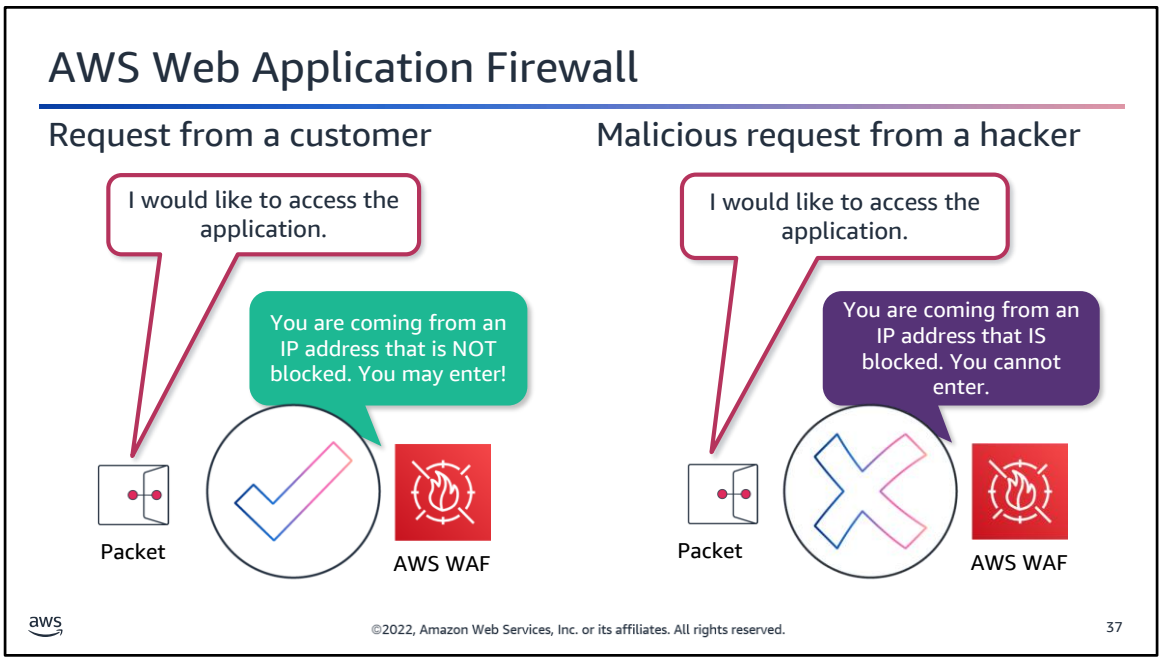
AWS WAF rule propagation and updates take under a minute, enabling you to quickly update security across your environment when issues arise. WAF supports hundreds of rules that can inspect any part of the web request with minimal latency impact to incoming traffic.

Ease of deployment & maintenance

AWS WAF is easy to deploy and protect applications deployed on either Amazon CloudFront as part of your CDN solution, the Application Load Balancer that fronts all your origin servers, Amazon API Gateway for your REST APIs, or AWS AppSync for your GraphQL APIs. There is no additional software to deploy, DNS configuration, SSL/TLS certificate to manage, or need for a reverse proxy setup. With AWS Firewall Manager integration, you can centrally define and manage your rules, and reuse them across all the web applications that you need to protect.

Easily monitor, block, or rate-limit bots

With AWS WAF Bot Control, you get visibility and control over common and pervasive bot traffic to your applications. Within the AWS WAF console, you can monitor common bots, such as status monitors and search engines, and get detailed, real-time visibility into the category, identity, and other details of bot traffic. You can also block, or rate-limit, traffic from pervasive bots, such as scrapers, scanners, and crawlers. Using AWS Firewall Manager, you can deploy the Bot Control managed rule group across multiple accounts in your AWS Organization



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

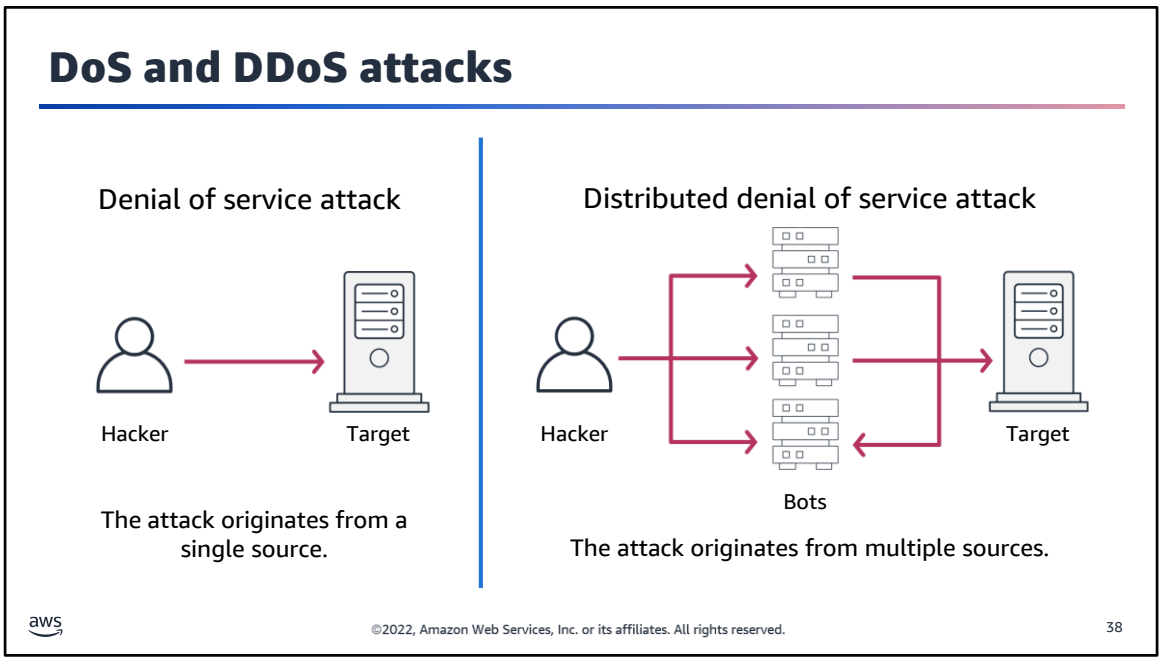
AWS WAF works together with Amazon CloudFront and an Application Load Balancer. Recall the network access control lists that you learned about in an earlier module. AWS WAF works in a similar way to block or allow traffic. However, it does this by using a **web access control list (ACL)** to protect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

Two types of application attacks that you might need to mitigate are denial of service and distributed 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**.

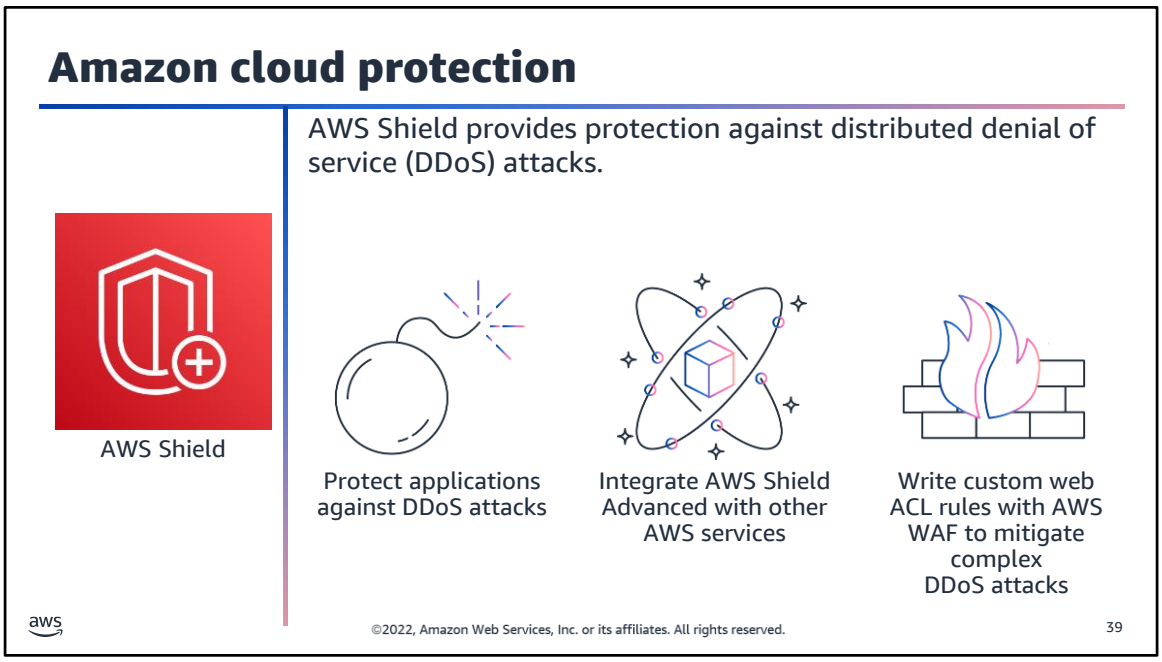
A **denial of service (DoS) attack** is a deliberate attempt to make a website or application unavailable to users. For example, an attacker might flood a website or application with excessive network traffic until the targeted website or application becomes overloaded and is no longer able to respond.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

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**.



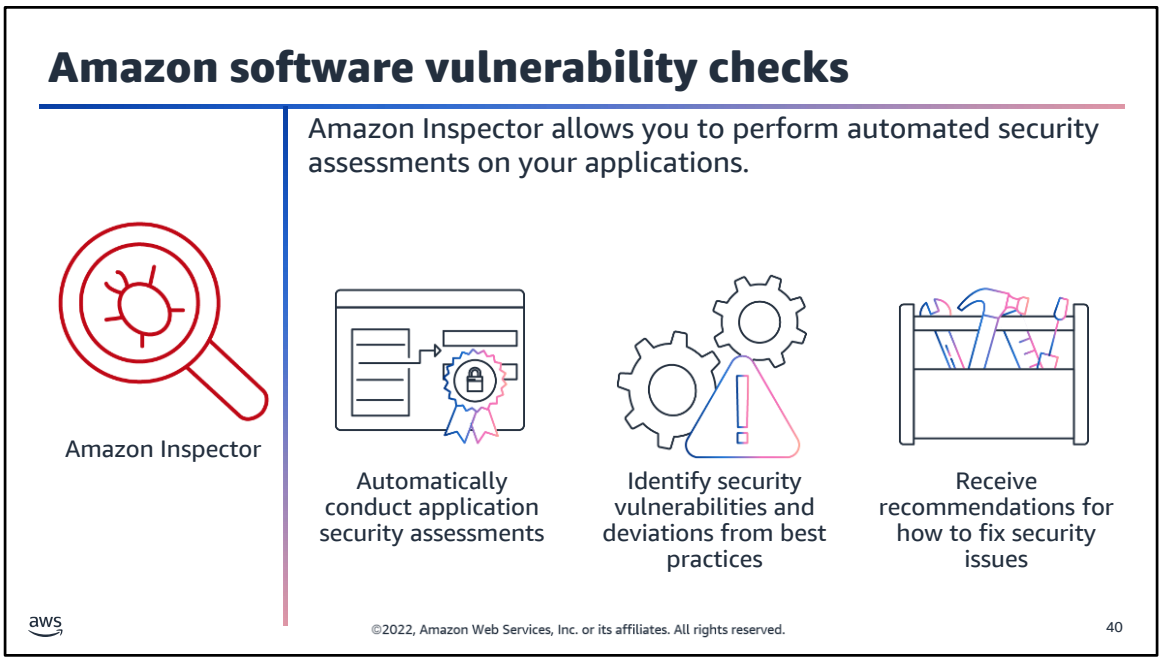
AWS Shield is a service that protects applications against DDoS attacks. AWS Shield provides two levels of protection: Standard and Advanced.

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 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.

Next, you will explore Amazon Inspector.



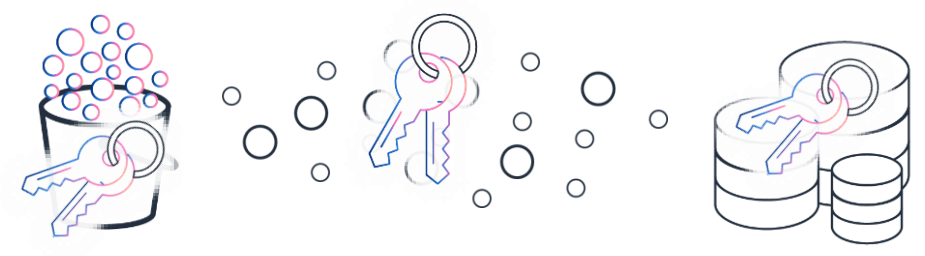
Suppose that the developers at the coffee shop are developing and testing a new ordering application. They want to make sure that they are designing the application in accordance with security best practices. However, they have several other applications to develop, so they cannot spend much time conducting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

Encryption

At-rest	In-transit
This is the encryption of data that is stored on a device or in a Cloud service.	This is the encryption of data as it travels from one place to another.

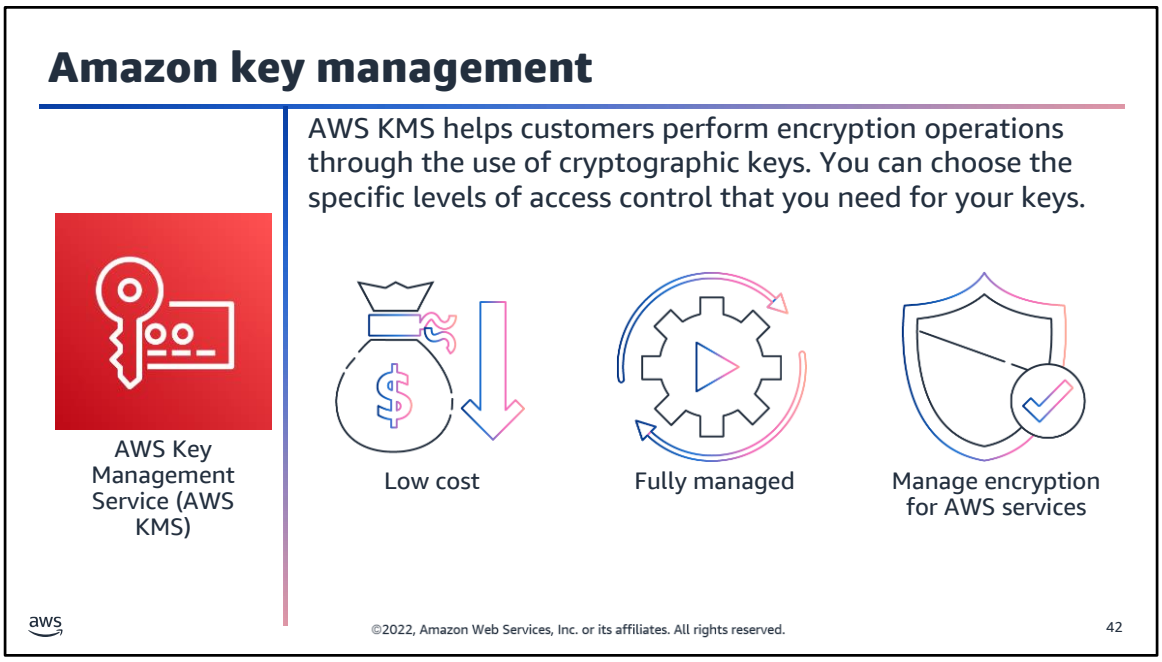


aws

41

In the airport there are hundreds of workers. You can think of them as data. The employees have information that they cannot share and keys that they cannot share when they are in the airport and when they travel from one concourse to another.

In the same way, you must ensure that your applications' data is secure while in storage **(encryption at rest)** and while it is transmitted **(encryption in transit)**.



AWS KMS helps customers perform encryption operations through the use of cryptographic keys. You can choose the specific levels of access control that you need for your keys.

Fully managed

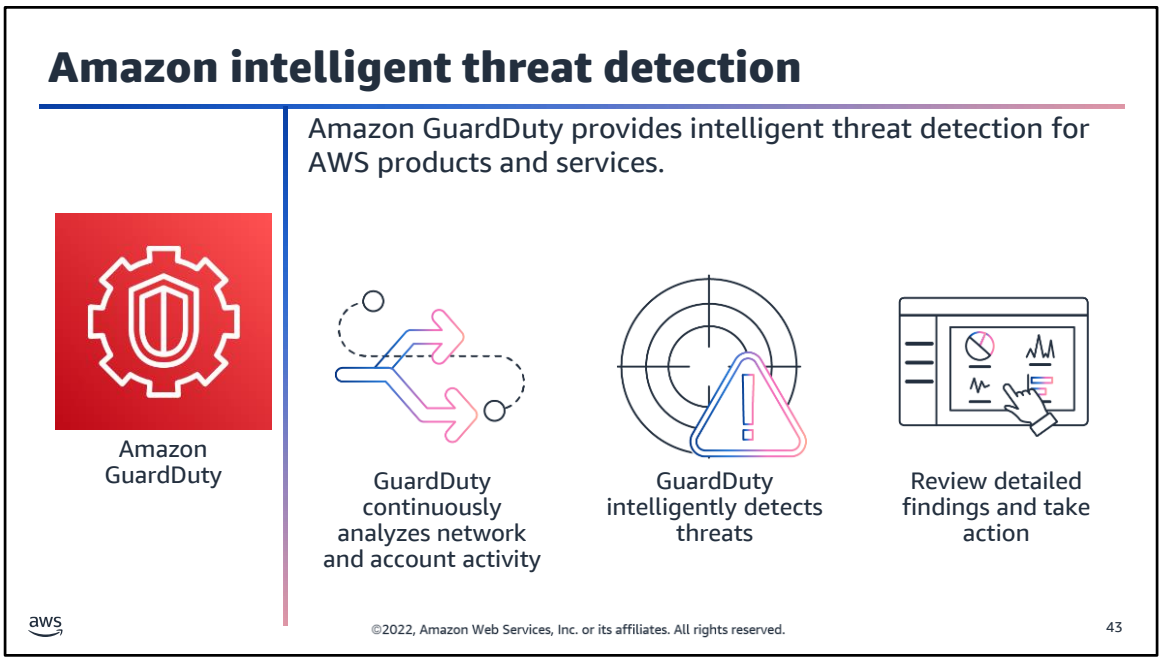
You control access to your encrypted data by defining permissions to use keys while AWS KMS enforces your permissions and handles the durability and physical security of your keys.

Manage encryption for AWS services

AWS KMS is integrated with AWS services to simplify using your keys to encrypt data across your AWS workloads. You choose the level of access control that you need, including the ability to share encrypted resources between accounts and services. KMS logs all use of keys to AWS CloudTrail to give you an independent view of who accessed your encrypted data, including AWS services using them on your behalf.

Low cost

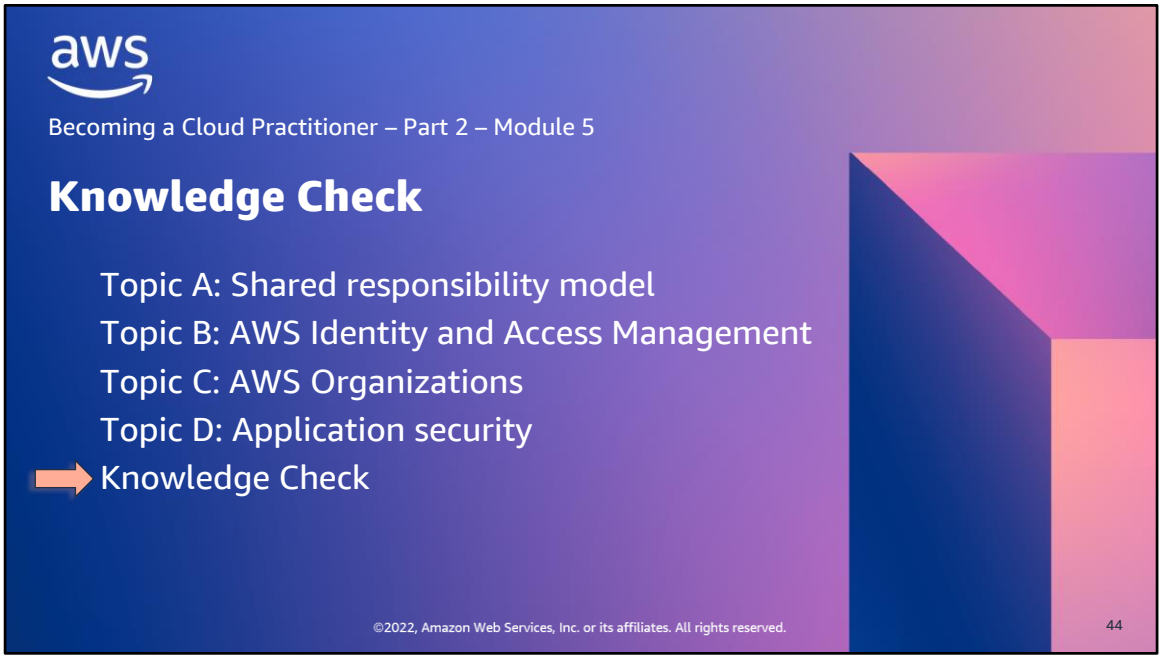
There is no commitment and no upfront charges to use AWS KMS. You only pay US $1/month to store any key that you create. AWS managed keys that are created on your behalf by AWS services are free to store. You are charged per-request when you use or manage your keys beyond the free tier.



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 behavior within your AWS environment.

After you enable 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.

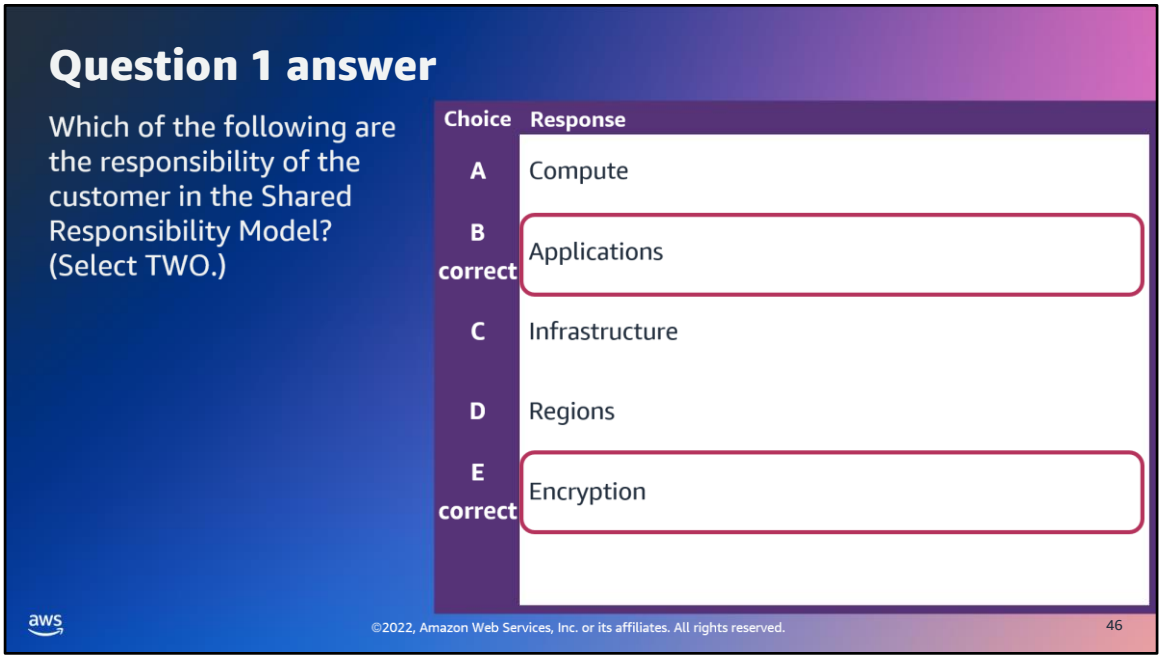
If GuardDuty detects any threats, you can review detailed findings about them from the AWS Management Console. Findings include recommended steps for remediation. You can also configure AWS Lambda functions to take remediation steps automatically in response to GuardDuty security findings.



Question 1

Which of the following are the responsibility of the customer in the Shared Responsibility Model? (Select TWO.)

Choice	Response
A	Compute
B	Applications
C	Infrastructure
D	Regions
E	Encryption

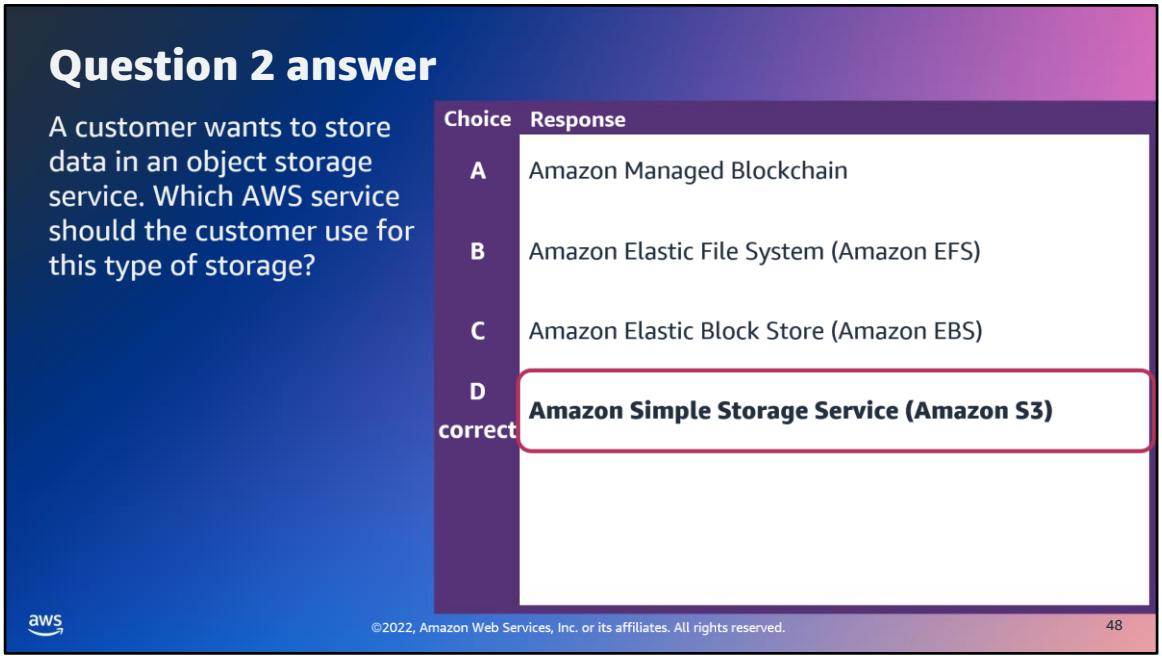


The correct answers are **Applications** and **Encryption**.
The other response options are incorrect because:
Compute, infrastructure, and regions are all the responsibility of AWS.

Question 2

Which of the following is NOT something you can create in AWS IAM?

Choice	Response
A	User
B	Policy
C	Group
D	Key

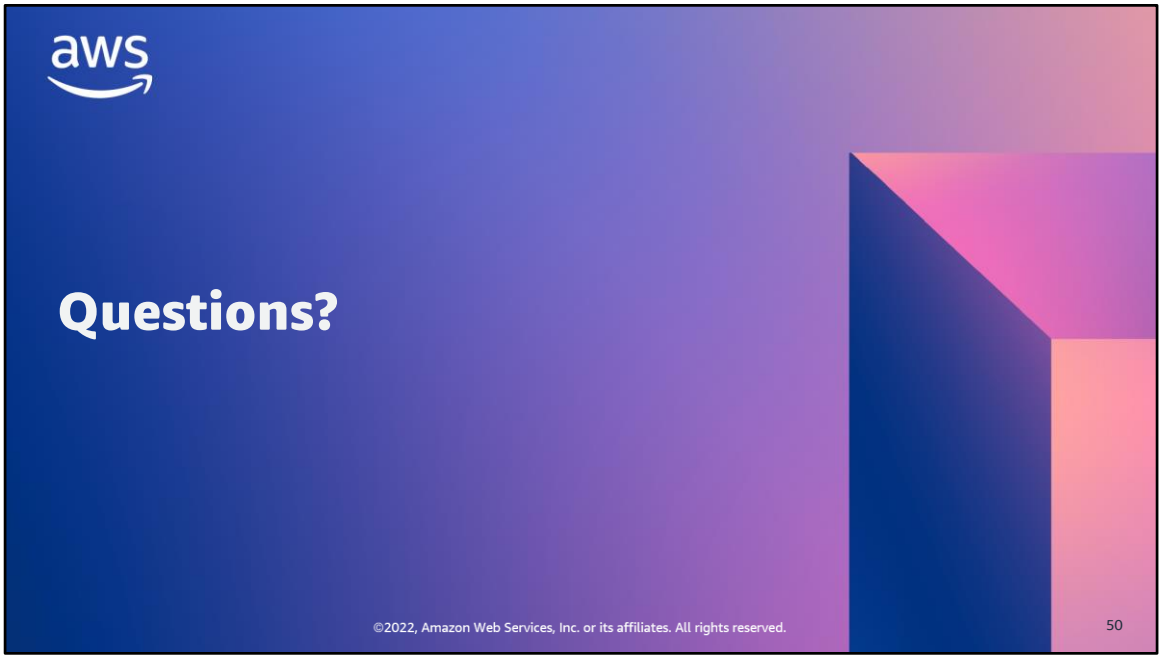


The correct answer is **Key**. A key is associated with encryption services such as AWS Key Management Service.
The other response options are incorrect because:
User, policy, and group are all features of AWS IAM.

Module 5 summary

In this module, you learned how to:

- Describe the Amazon Shared Responsibility Model
- Summarize Amazon IAM and how to work with identities
- Summarize AWS Organizations
- Summarize application security in AWS

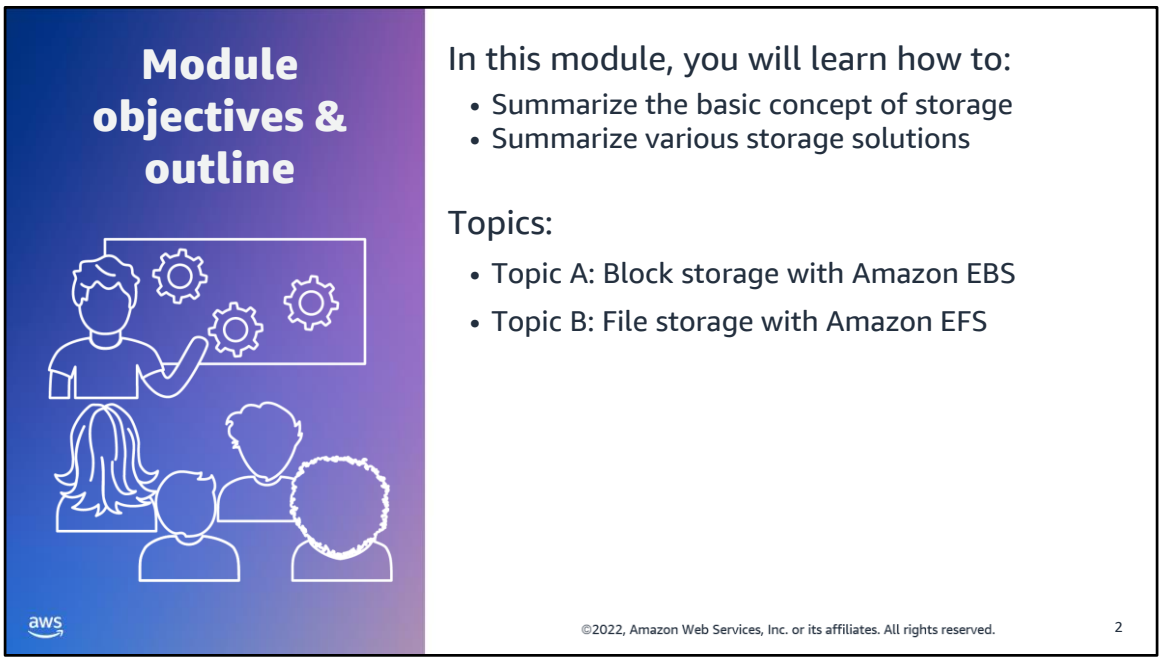


In this module, you will learn about block and file storage types, their specific use cases, and their benefits.
Module Objectives:

- Summarize the basic concept of storage
- Summarize various storage solutions

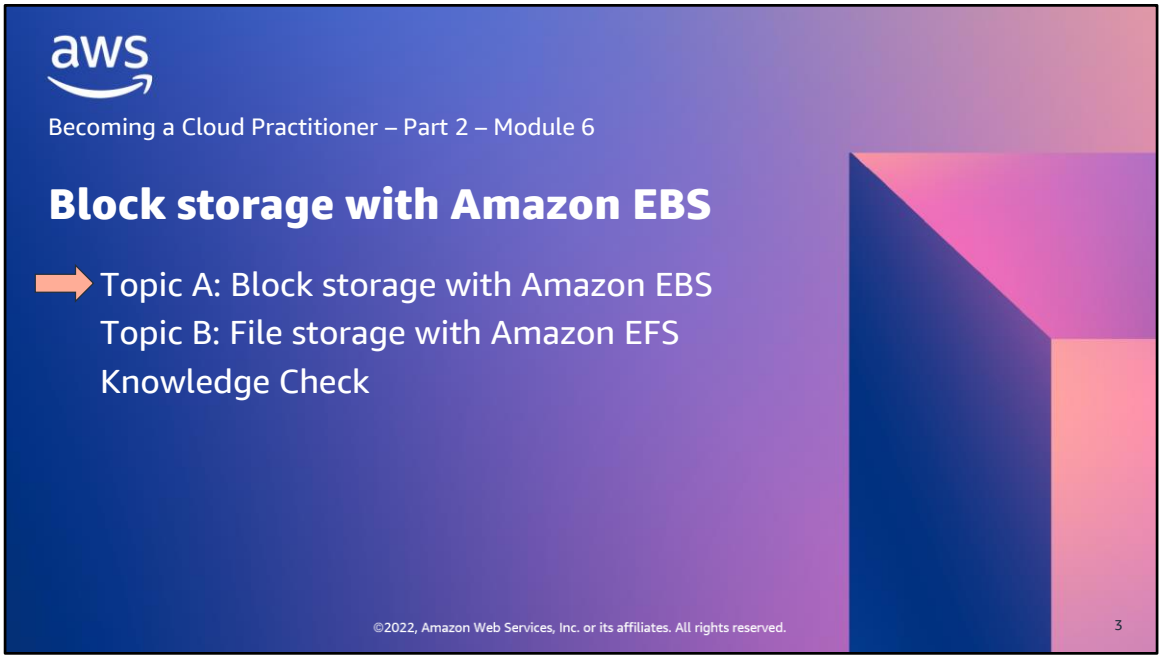
Topics:

- Topic A: Block storage with Amazon EBS
- Topic B: File storage with Amazon EFS



In this module, you will learn about storage and database types, their specific use cases, and their benefits.
Module Objectives:
- Summarize the basic concept of storage
- Summarize various storage solutions

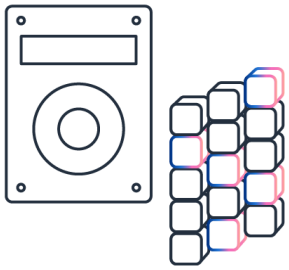
Topics:
- Topic A: Block storage with Amazon EBS
- Topic B: File storage with Amazon EFS



In this topic, you will learn block storage with Amazon Elastic Block Store (Amazon EBS).

Block storage

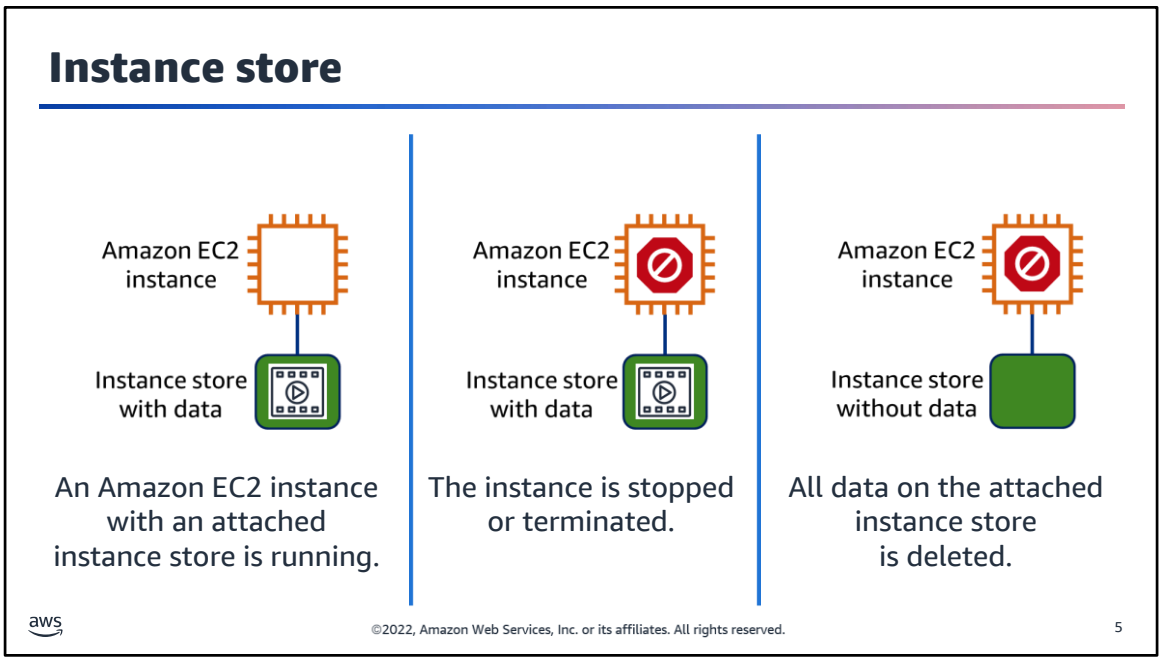
- In block storage, files are separated into equal-sized pieces (blocks) of data.
- Block storage is used for applications that run on Amazon EC2 instances.



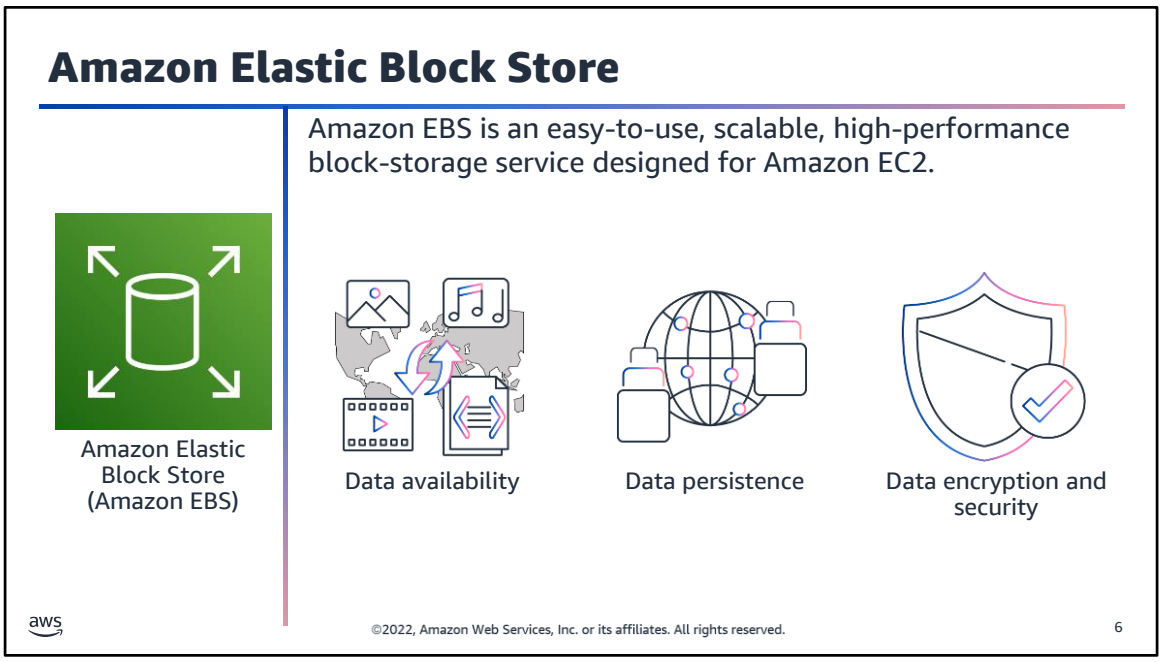
Block-level storage volumes behave like physical hard drives. In block storage, files are separated into equal-sized pieces (or blocks) of data. When a file in block storage is modified, only the pieces that are changed are updated.

Use block storage in a number of places such as databases, enterprise software applications, and computer hard drives. For example, suppose that you make a change to a single file on your computer's hard drive. Only the blocks for that file are updated. All of the blocks across the hard drive remain unchanged.

You can also use block storage for applications that run on Amazon EC2 instances. One type of block storage that you can use with Amazon EC2 instances is an **instance store**.



An **instance store** provides temporary block-level storage for an Amazon EC2 instance. If you stop or terminate an Amazon EC2 instance, all the data written to the attached instance store is deleted.
Suppose that you are working on a document on your computer. You can think of an instance store as a working copy of the document. While you are working on the document, all the information is there. You can review the document, copy and paste information from the document, and so on. However, if you turn off your computer without saving the document, the document will no longer be there when you turn on the computer again.
Recall that Amazon EC2 instances are virtual servers. If you start an instance from a stopped state, the instance may start on another host, where the previously used instance store volume does not exist. Therefore, AWS recommends instance stores for use cases that involve temporary data that you do not need in the long term.
If you are working with data that needs retention, you can store it in **Amazon Elastic Block Store (Amazon EBS)** volumes.



|Instructor notes

|

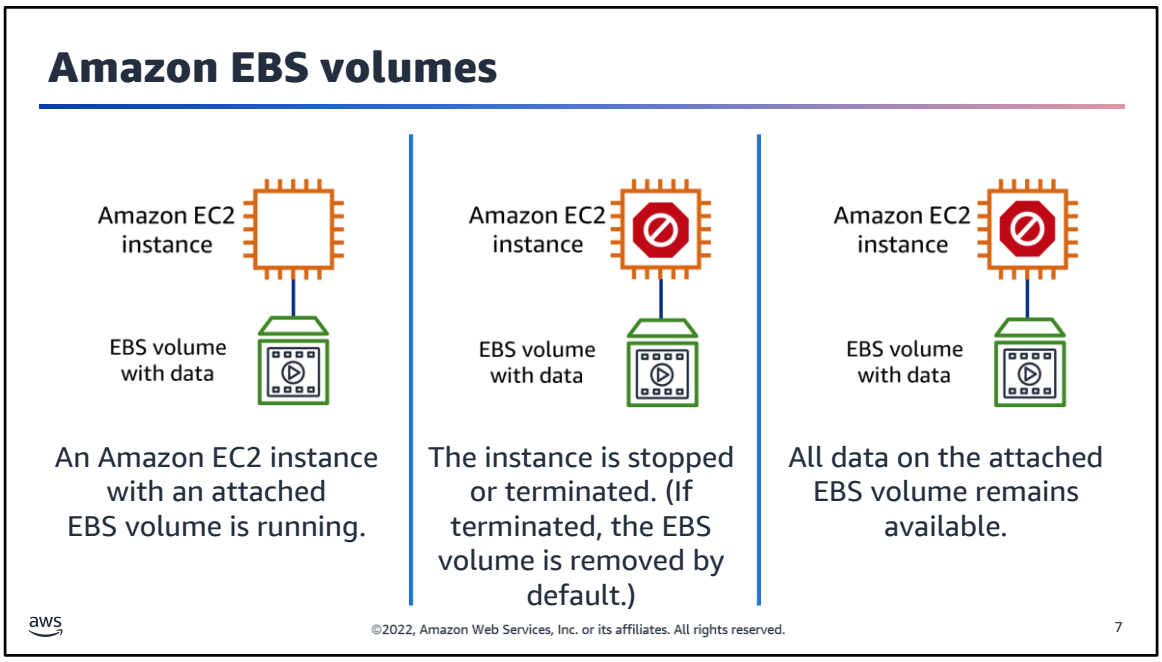
|Student notes

Amazon Elastic Block Store (Amazon EBS) is an easy-to-use, scalable, high-performance block-storage service designed for Amazon EC2.

An Amazon EBS volume is a durable, block-level storage device that you can attach to your instances. After you attach a volume to an instance, you can use it as you would use a physical hard drive. EBS volumes are flexible. For current-generation volumes attached to current-generation instance types, you can dynamically increase size, modify the provisioned IOPS capacity, and change volume type on live production volumes.

You can use EBS volumes as primary storage for data that requires frequent updates, such as the system drive for an instance or storage for a database application. You can also use them for throughput-intensive applications that perform continuous disk scans. EBS volumes persist independently from the running life of an EC2 instance.

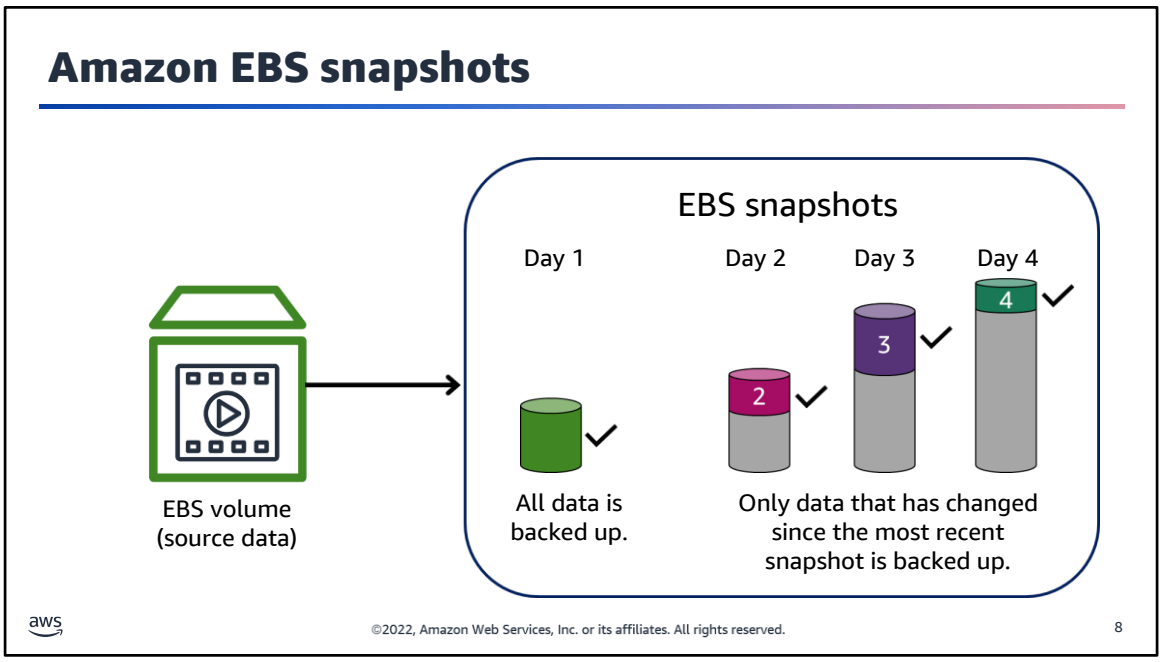
You can attach multiple EBS volumes to a single instance. The volume and instance must be in the same Availability Zone. Depending on the volume and instance types, you can use Multi-Attach to mount a volume to multiple instances at the same time.



If you stop or terminate an Amazon EC2 instance, all the data on the attached Amazon EBS volume remains available.

To create an Amazon EBS volume, you define the configuration (such as volume size and type) and provision it. After you create an Amazon EBS volume, it can attach to an Amazon EC2 instance.

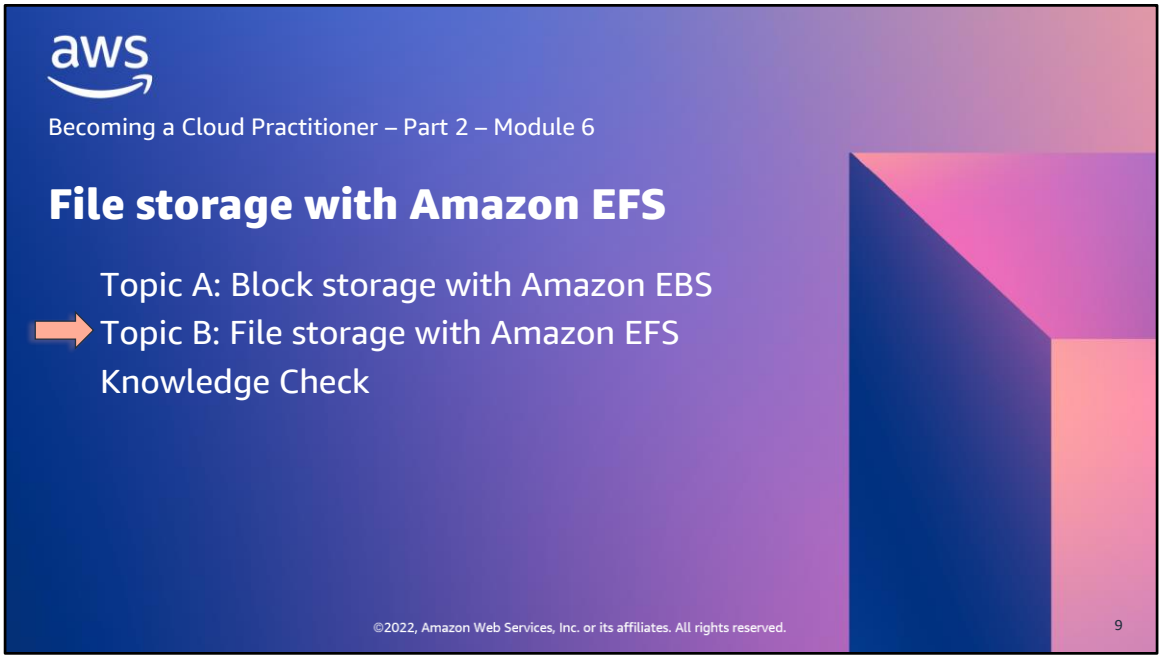
Because Amazon EBS volumes are intended for data that needs to persist, it's important to back up the data. You can take incremental backups of Amazon EBS volumes by creating Amazon EBS snapshots.



An **Amazon EBS snapshot** is an incremental backup. This means that the first backup taken of a volume copies all the data. For subsequent backups, only the blocks of data that have changed since the most recent snapshot are saved.

Incremental backups are different from full backups, in which *all* the data in a storage volume copies each time a backup occurs. The backup includes data that has not changed since the most recent backup.

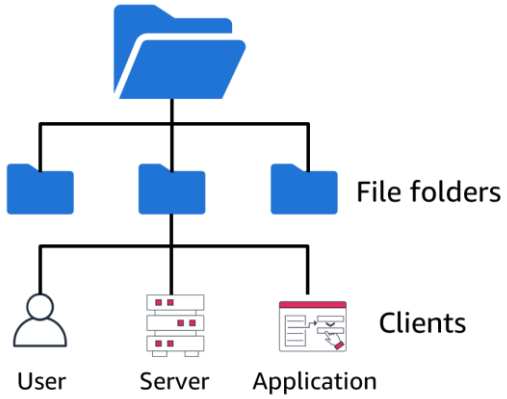
Later in this course, you will learn about Amazon CloudWatch. It is a service that you can use with Amazon EBS to automatically create Amazon EBS snapshots on a regular schedule.



In this topic, you will learn about file storage with Amazon EFS.

File storage

In file storage, multiple clients can access data that is stored in shared file folders.

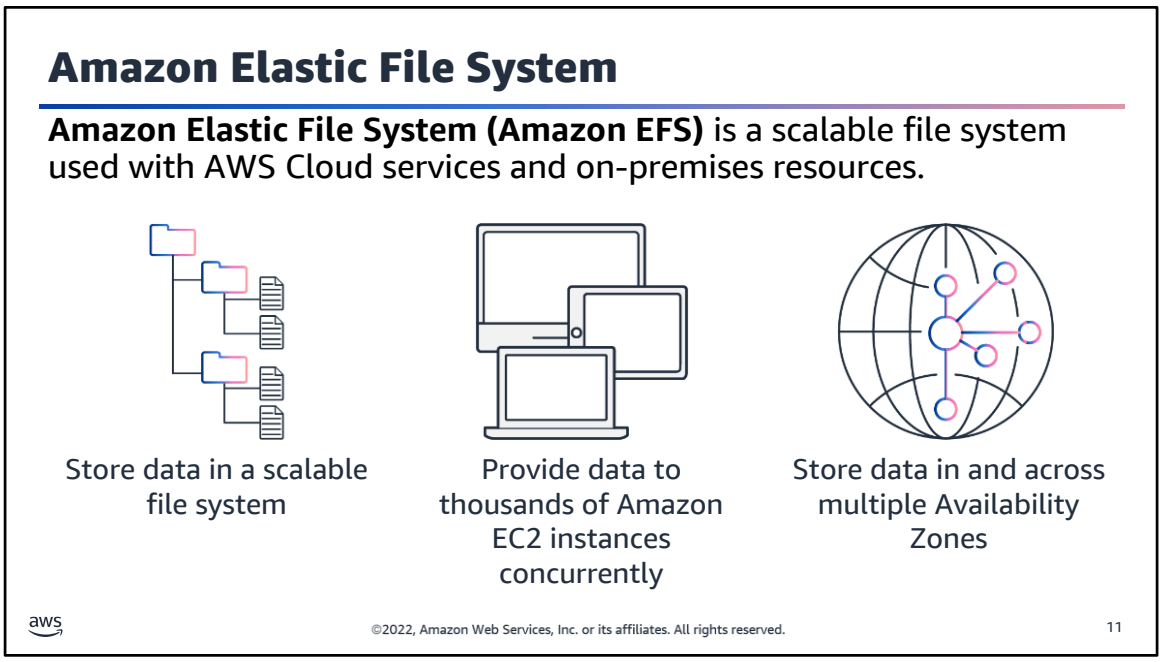


Suppose that your company has a large amount of data needing access by many users or applications simultaneously. For example, this data might connect to multiple servers that are constantly performing analytics on it.

In **file storage**, multiple clients (such as users, applications, servers, and so on) can access data that is stored in shared file folders. In this approach, a storage server uses block storage with a local file system to organize files. Clients access data through file paths.

Compared to block storage and object storage, file storage is ideal for use cases in which a large number of services and resources need to access the same data at the same time.

Next you will examine **Amazon Elastic File System (Amazon EFS)**, a service that provides file storage.

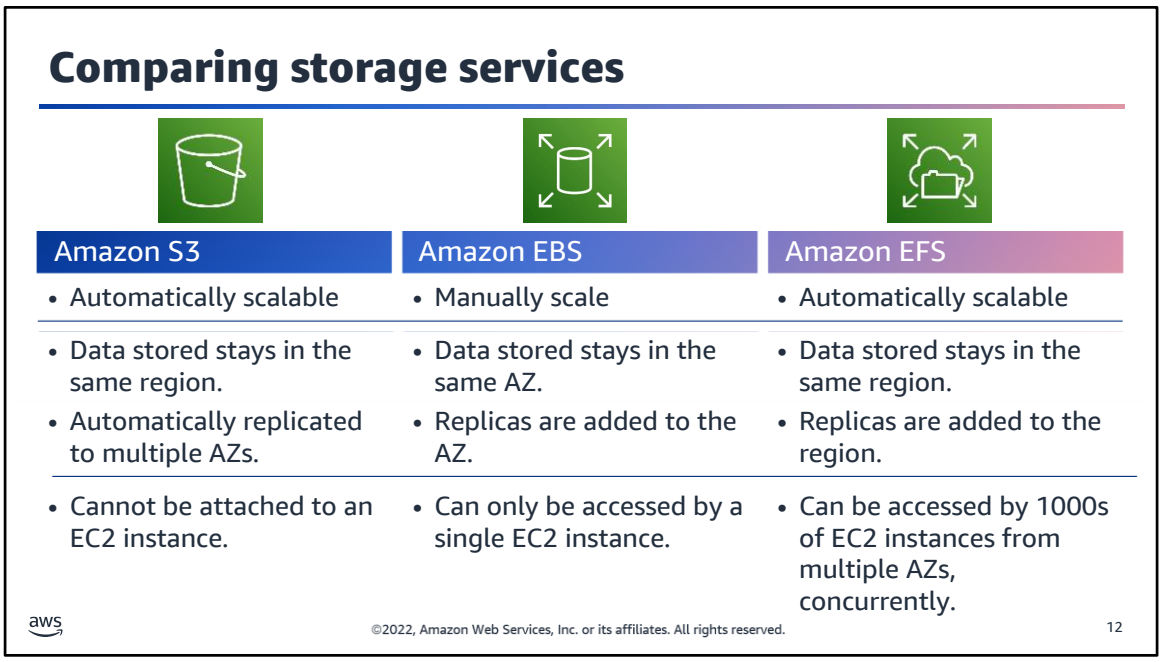


Amazon Elastic File System (Amazon EFS) is a scalable file system used with AWS Cloud services and on-premises resources. As you add and remove files, Amazon EFS grows and shrinks automatically. It can scale on demand to petabytes without disrupting applications.

Suppose that multiple applications need access to your data in Amazon EFS at the same time, but without affecting the applications' performance. Thousands of Amazon EC2 instances can access Amazon EFS concurrently.

One of the differences between Amazon EBS and Amazon EFS is how the two services work within and across Availability Zones. Amazon EBS volumes are an *Availability Zone-level* resource. An EBS volume stores data within a single Availability Zone. To attach an Amazon EC2 instance to an EBS volume, both the Amazon EC2 instance and the EBS volume must reside within the same Availability Zone.

By contrast, Amazon EFS is a *Regional* service. It stores data within and across multiple Availability Zones. This allows data to be accessed concurrently from all the Availability Zones in the Region where a file system is located. Additionally, on-premises servers are able to access Amazon EFS by using AWS Direct Connect.



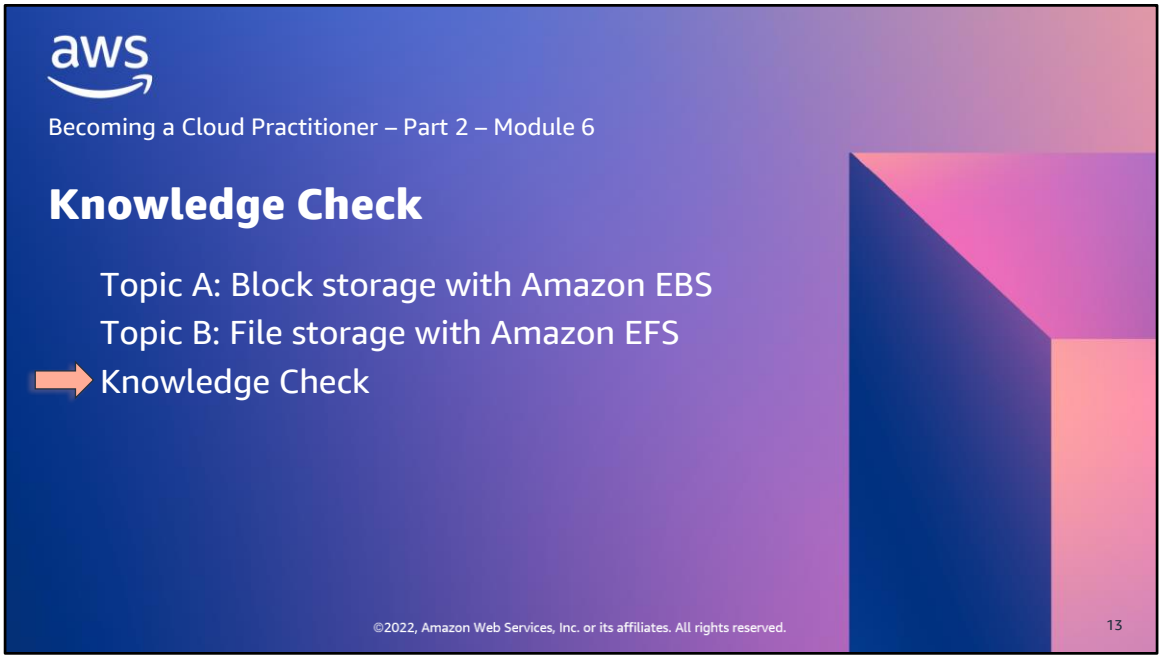
It can be easy to confuse the two storage types we have just discussed, Amazon EBS and Amazon EFS. This section should help you to understand additional key differences between these services.

Storage type: Amazon S3 is an object store, Amazon EBS is a block store, while Amazon EFS is a file store. This is a huge difference and cannot be overlooked.

Performance: Amazon S3 scales automatically without your involvement. Amazon EBS is scaled by adding additional Amazon EBS volumes to an Amazon EC2 instance. You cannot make an existing volume larger. Amazon EFS on the other hand automatically scales based on the volume that is required by the data being stored.

Data Storage: Amazon S3 is a regional service. Data stored within an S3 bucket stays in that same Region and is automatically duplicated across multiple AZs to provide durability and resilience. Amazon EBS is a zonal service. This means that all data stored on Amazon EBS volumes will always be recovered to or replicated within the same AZ it originated in. Amazon EFS however is a regional service. This means that all data stored on Amazon EFS will always be recovered to or replicated within the same Region but you can move the data to multiple AZs.

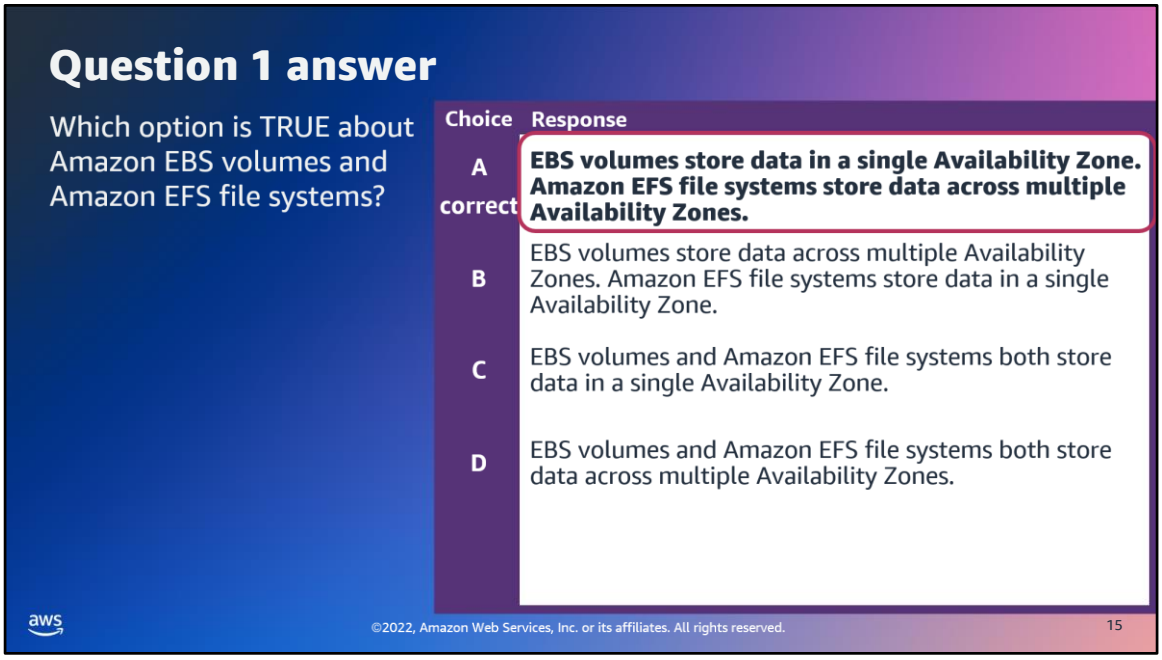
Data Access: Amazon S3 cannot be directly attached to an EC2 instance. You would need to write logic to instruct applications where the buckets are located. Amazon EBS volumes can only be attached to or accessed by a single Amazon EC2 instance at once. You can detach an Amazon EBS volume and attach it to a different instance if you need to. Amazon EFS files systems can be attached to thousands of Amazon EC2 instances at the same time. These instances do not have to be in the same AZ but they do have to be in the same Region.



Question 1

Which option is TRUE about Amazon EBS volumes and Amazon EFS file systems?

Choice	Response
A	EBS volumes store data in a single Availability Zone. Amazon EFS file systems store data across multiple Availability Zones.
B	EBS volumes store data across multiple Availability Zones. Amazon EFS file systems store data in a single Availability Zone.
C	EBS volumes and Amazon EFS file systems both store data in a single Availability Zone.
D	EBS volumes and Amazon EFS file systems both store data across multiple Availability Zones.

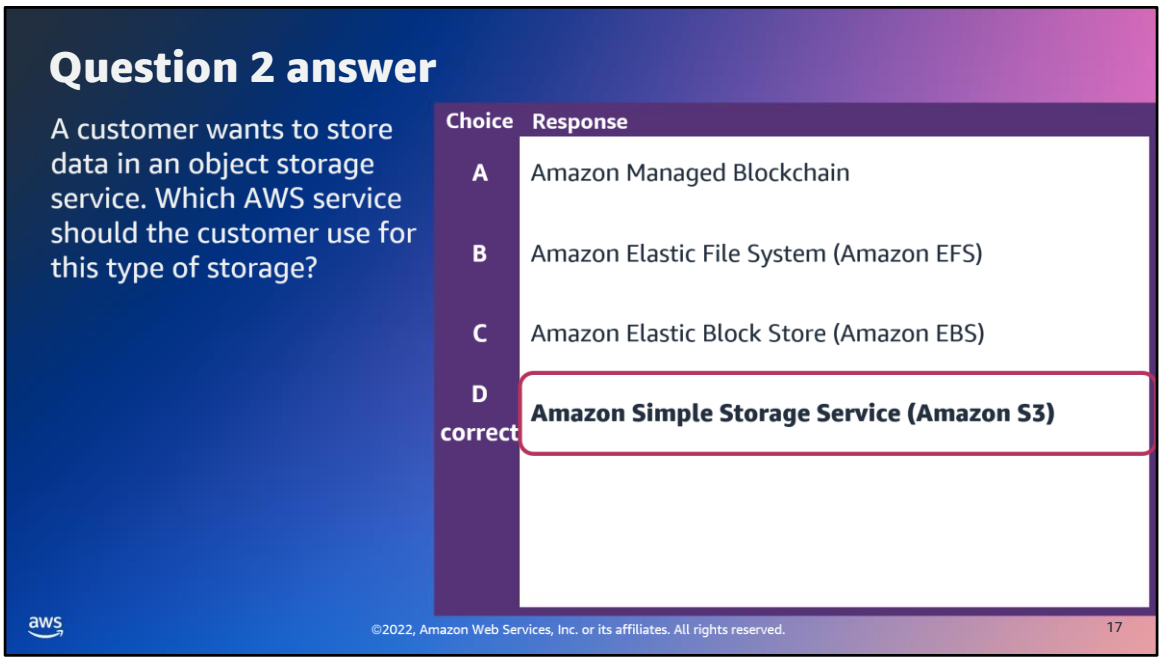


The correct response option is **EBS volumes store data within a single Availability Zone.**
The other responses are incorrect because:
Amazon EFS file systems store data across multiple Availability Zones.
An EBS volume must be located in the same Availability Zone as the Amazon EC2 instance to which it is attached. Data in an Amazon EFS file system can be accessed concurrently from all the Availability Zones in the Region where the file system is located.

Question 2

A customer wants to store data in an object storage service. Which AWS service should the customer use for this type of storage?

Choice	Response
A	Amazon Managed Blockchain
B	Amazon Elastic File System (Amazon EFS)
C	Amazon Elastic Block Store (Amazon EBS)
D	Amazon Simple Storage Service (Amazon S3)

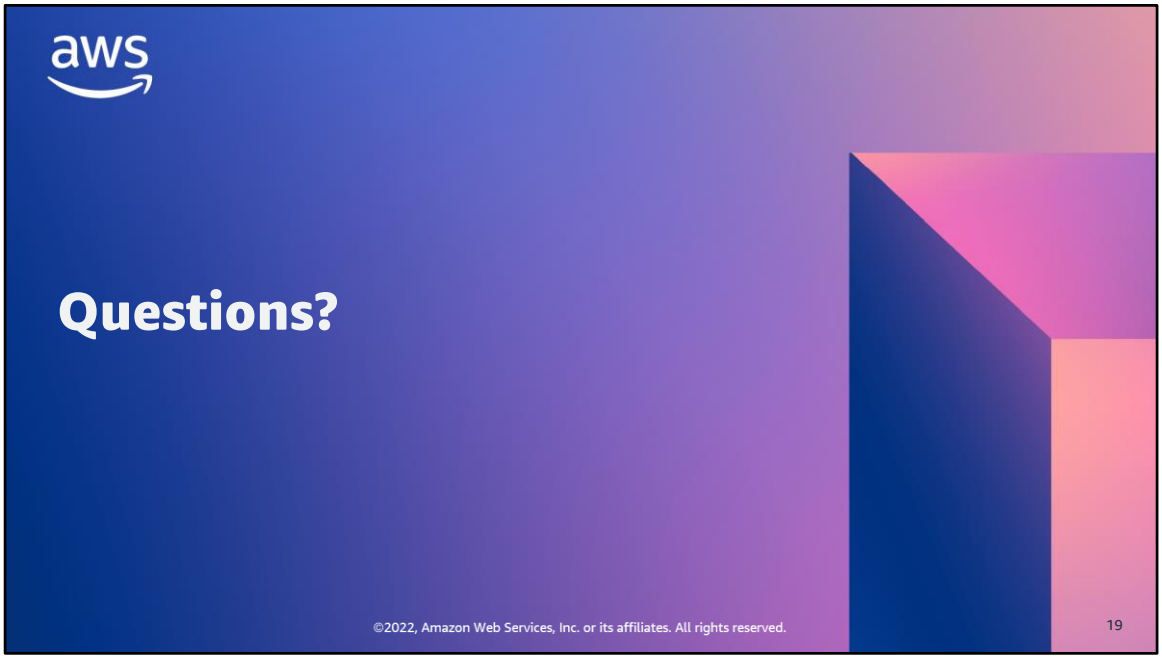


The correct response option is **Amazon Simple Storage Service (Amazon S3**).
The other response options are incorrect because:
Amazon Managed Blockchain is a service that you can use to create and manage blockchain networks with open-source frameworks. Blockchain is a distributed ledger system that lets multiple parties run transactions and share data without a central authority. Amazon Elastic File System (Amazon EFS) is a scalable file system used with AWS Cloud services and on-premises resources. It does not store data as object storage. Amazon Elastic Block Store (Amazon EBS) is a service that provides block-level storage volumes that you can use with Amazon EC2 instances.

Module 6 summary

In this module, you learned how to:

- Summarize the basic concept of storage and databases
- Summarize various storage solutions



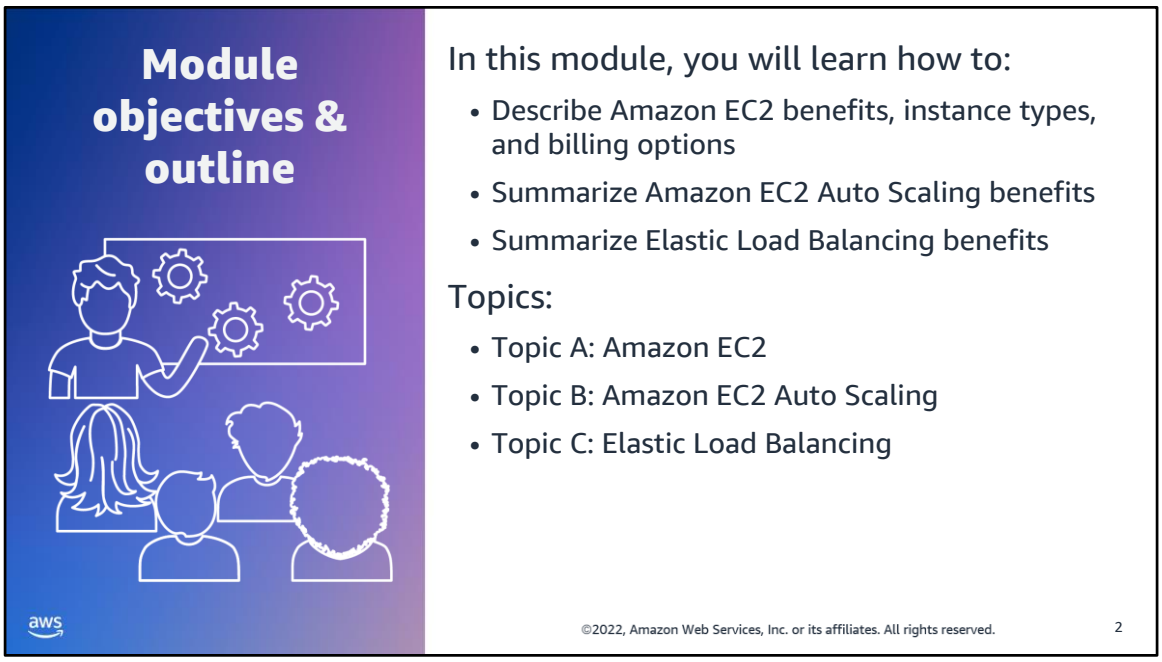
In this module, you will learn about compute in the AWS Cloud including Amazon EC2, auto scaling, and elastic load balancing services.

Module Objectives:

- Describe Amazon EC2 benefits, instance types, and billing options
- Summarize Amazon EC2 Auto Scaling benefits
- Summarize Elastic Load Balancing benefits

Topics:

- Topic A: Amazon EC2
- Topic B: Amazon EC2 Auto Scaling
- Topic C: Elastic Load Balancing



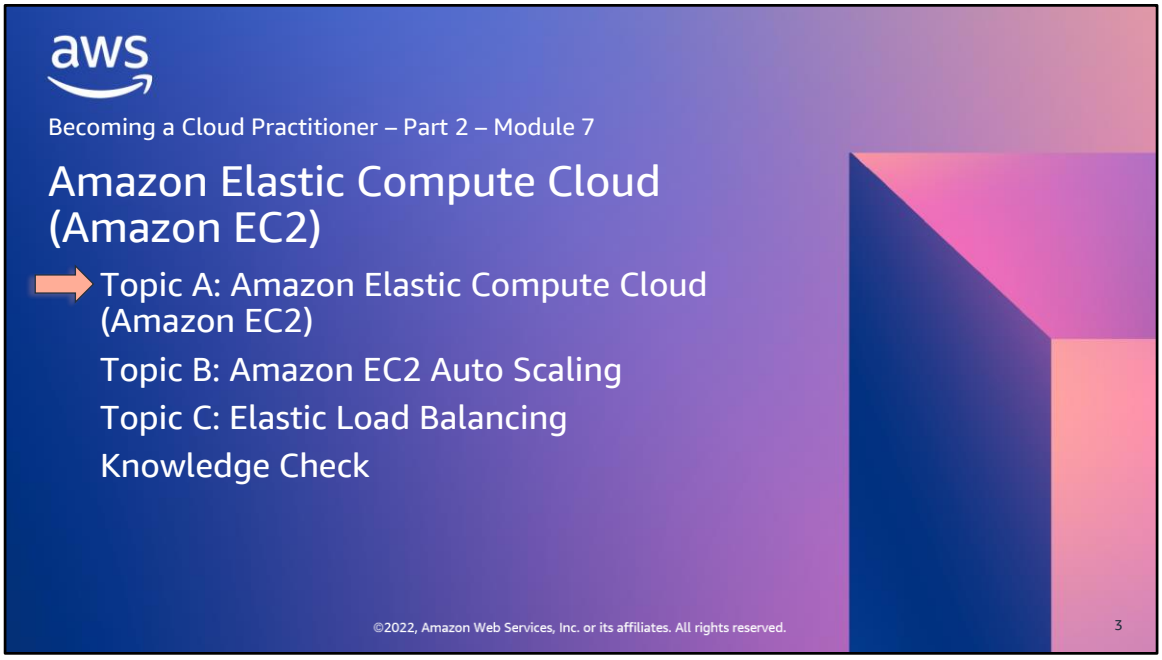
In this module, you will learn about compute in the AWS Cloud including Amazon EC2, auto scaling, and elastic load balancing services.

Module Objectives:

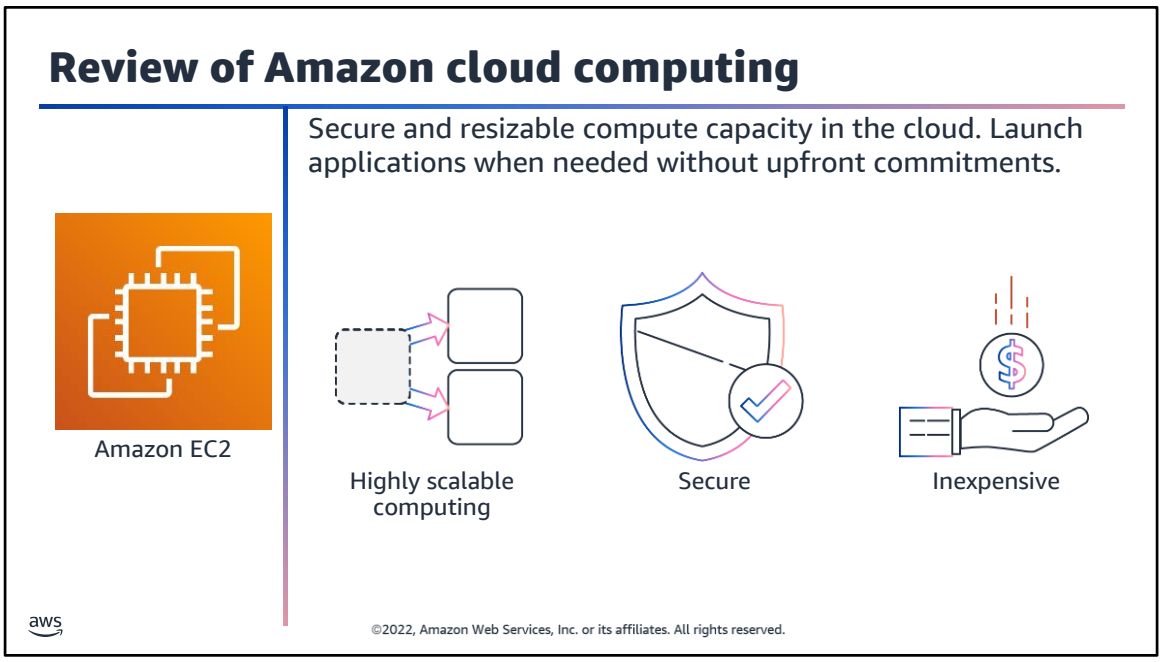
- Describe Amazon EC2 benefits, instance types, and billing options
- Summarize Amazon EC2 Auto Scaling benefits
- Summarize Elastic Load Balancing benefits

Topics:

- Topic A: Amazon EC2
- Topic B: Amazon EC2 Auto Scaling
- Topic C: Elastic Load Balancing



In this topic, you will learn about the Amazon EC2 service.



|Instructor Notes

| This slide is here to refresh the memories of learning as it was introduced in part 1.

|Student Notes

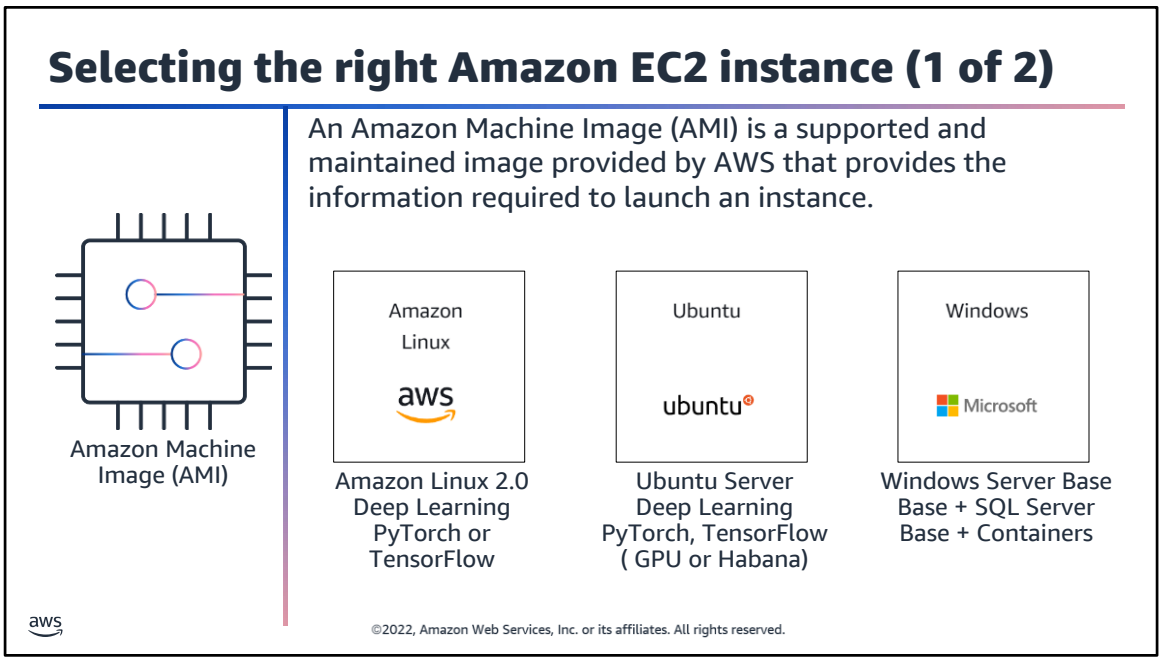
Amazon EC2 provides secure, resizable compute capacity in the cloud as Amazon EC2 instances.

Imagine that you are responsible for the architecture of your company's resources and must support new websites. With traditional on-premises resources, you would:

1. Spend money upfront to purchase hardware.
2. Wait for the servers to be delivered to you.
3. Install the servers in your physical data center.
4. Make all the necessary configurations.

By comparison, with an Amazon EC2 instance, you would use a virtual server to run applications in the AWS Cloud. You could:

- Provision and launch an Amazon EC2 instance within minutes
- Stop using it when you finish running a workload
- Pay only for the compute time you use when an instance is running, not when it is stopped or shut down
- Save costs by paying only for server capacity that you need or want



|Instructor Notes

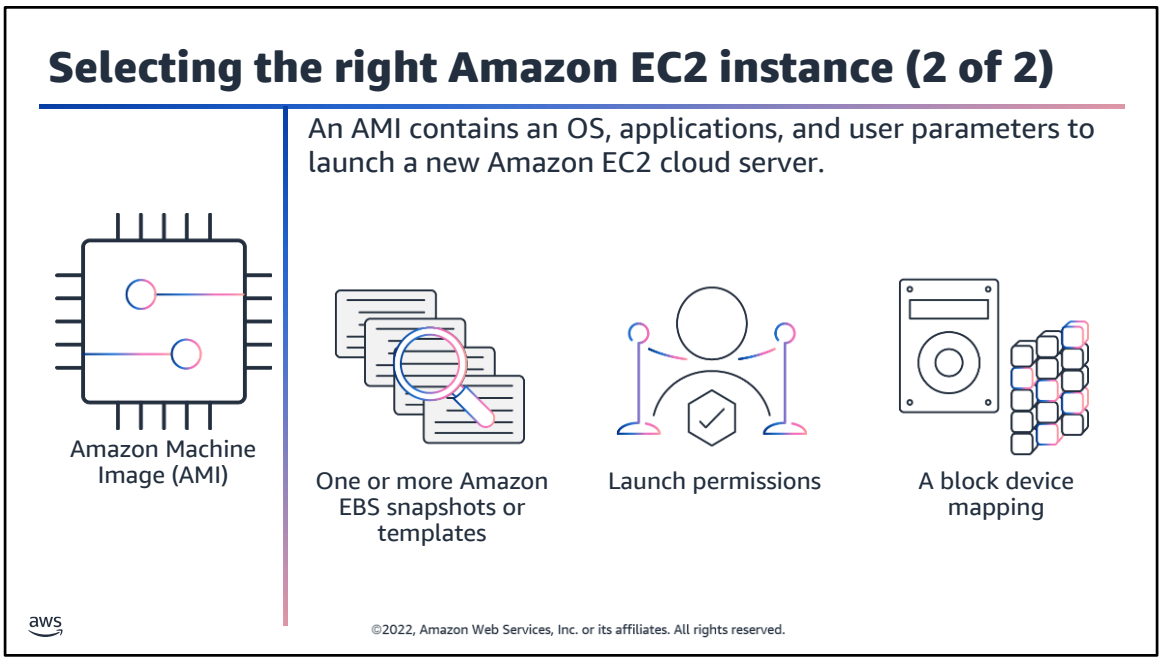
| This is a new slide to introduce the concept of AMIs

|Student Notes

An **Amazon** Machine Image (**AMI**) is a supported and maintained image provided by **AWS** that provides the information required to launch an instance.

An AMI includes the following:

- One or more Amazon Elastic Block Store (Amazon EBS) snapshots, or, for instance-store-backed AMIs, a template for the root volume of the instance (for example, an operating system, an application server, and applications).
- Launch permissions that control which AWS accounts can use the AMI to launch instances.
- A block device mapping that specifies the volumes to attach to the instance when it's launched.



|Instructor Notes

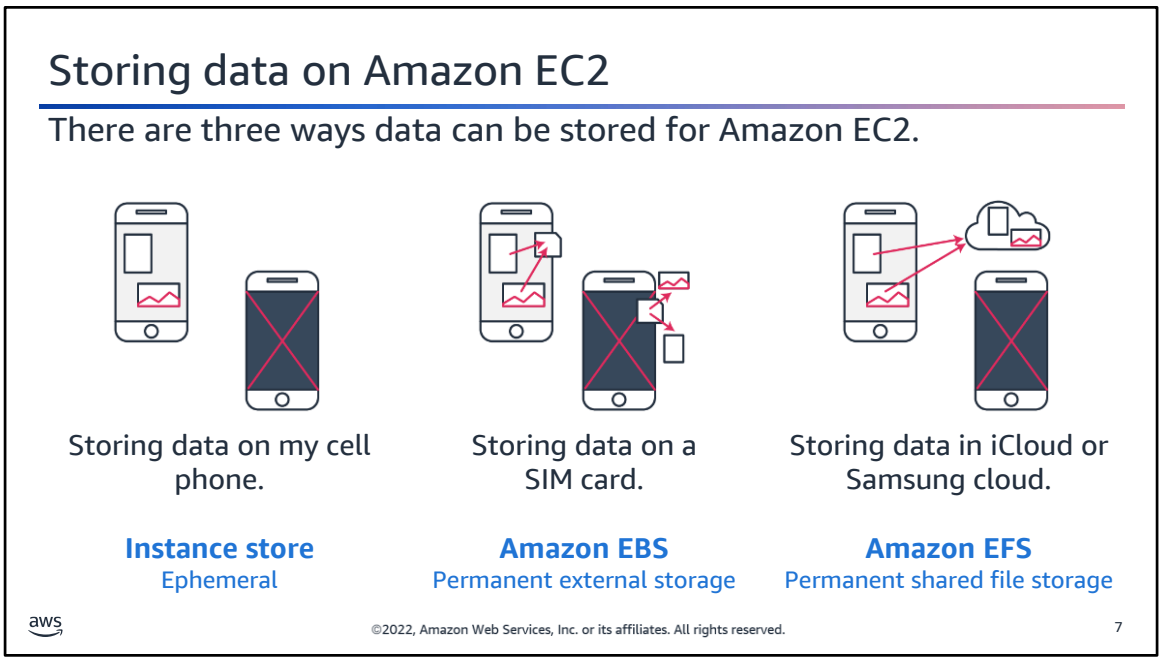
| This is a new slide to introduce what is contained in an AMI.

|Student Notes

An **Amazon** Machine Image (**AMI**) is a supported and maintained image provided by **AWS** that provides the information required to launch an instance.

An AMI includes the following:

- One or more Amazon Elastic Block Store (Amazon EBS) snapshots, or, for instance-store-backed AMIs, a template for the root volume of the instance (for example, an operating system, an application server, and applications).
- Launch permissions that control which AWS accounts can use the AMI to launch instances.
- A block device mapping that specifies the volumes to attach to the instance when it's launched.

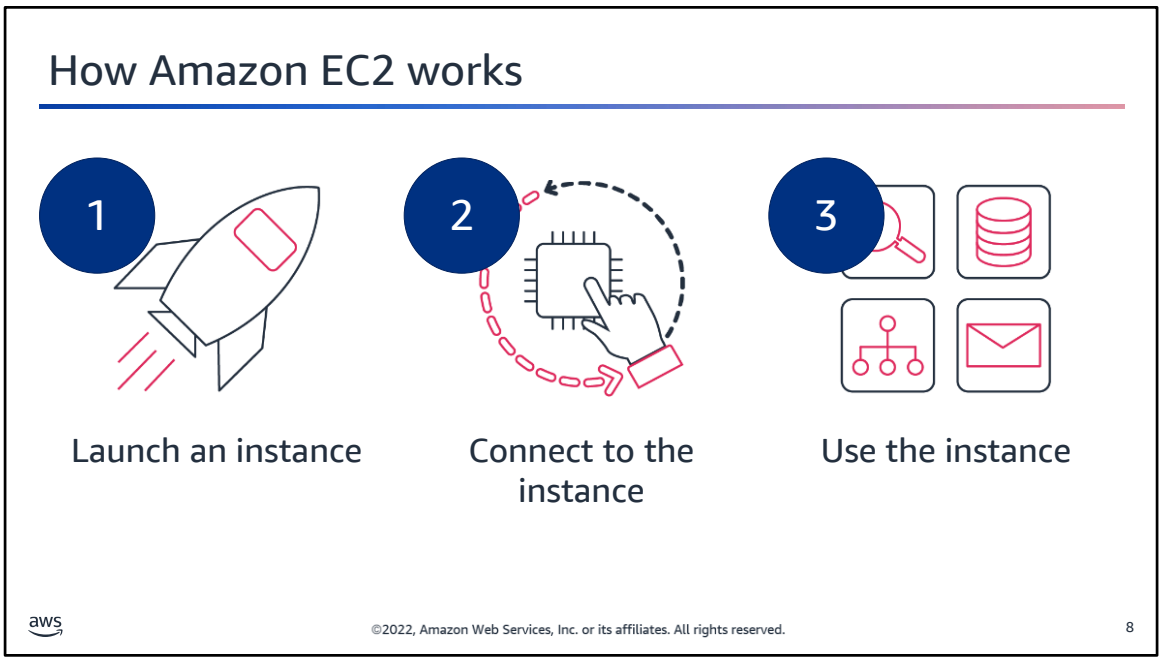


Imagine that you buy the latest and greatest cell phone. When you download files and take pictures they are stored right on your phone. What happens if you break your phone? All of your files and photos will be gone. Many cell phones offer the options of inserting a SIM card. When you save files and photos you can save them on the SIM card instead of right on the phone. Now, what happens if you break your phone? As long as the SIM card in not damaged, you can remove it and put it in a new phone. Your files and photos will be available. Lastly, you may have the options of saving files in a cloud service such as Apple iCloud or Samsung cloud. Anything stored in the cloud is available on other devices. If you break your phone, all of your data is still available. Relating this to our AWS services, The first example of storing data right on the phone is like using Amazon EC2's instance store. The second example of storing data on a SIM card is like using Amazon EBS volumes attached to your Amazon EC2 instance. And, the final example of storing data in the cloud is similar to using Amazon EFS volumes attached to your Amazon EC2 instance.

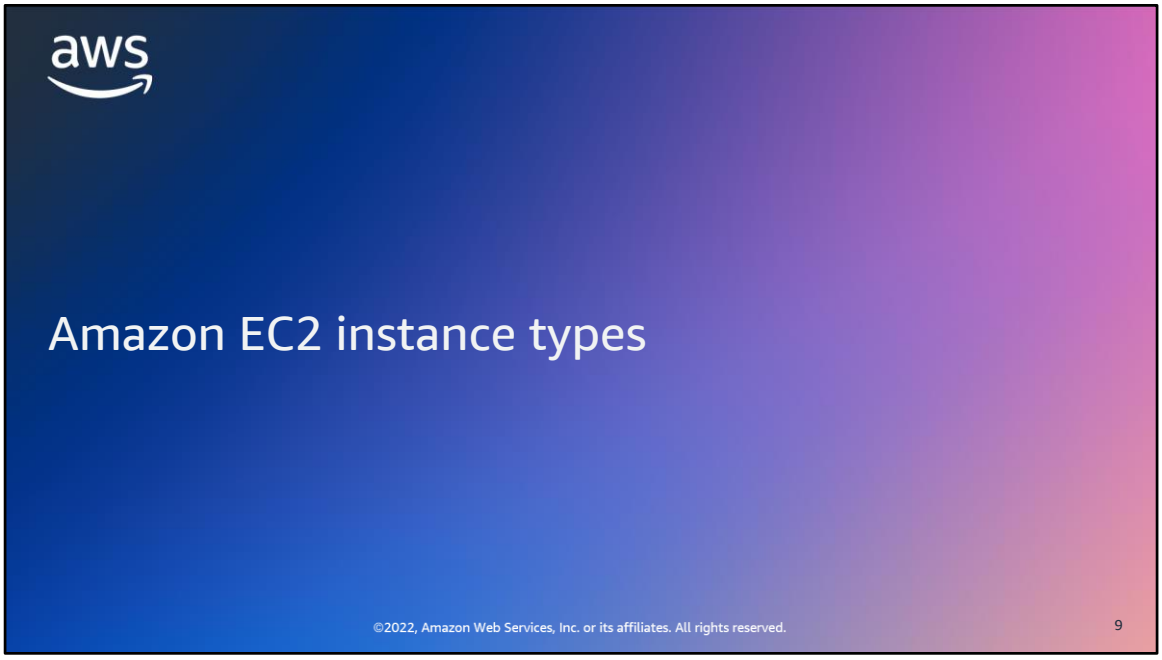
Instance store - The instance store is ideal for temporary storage, because the data stored in instance store volumes is not persistent. When the instance is shut down all data in this storage is removed.

Amazon Elastic Block Store - EBS volumes preserve their data through instance stops and terminations, can be easily backed up with EBS snapshots, can be removed from one instance and reattached to another, and support full-volume encryption.

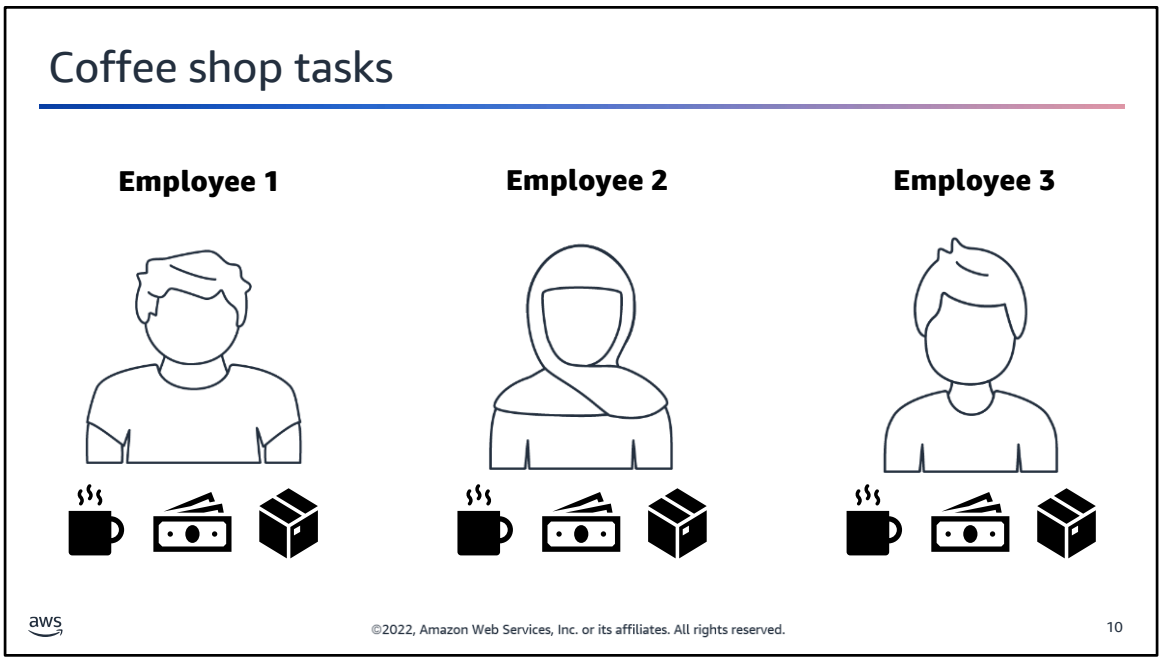
Amazon Elastic File Store - Amazon EFS provides scalable file storage for use with Amazon EC2. You can use an EFS file system as a common data source for workloads and applications running on multiple instances.



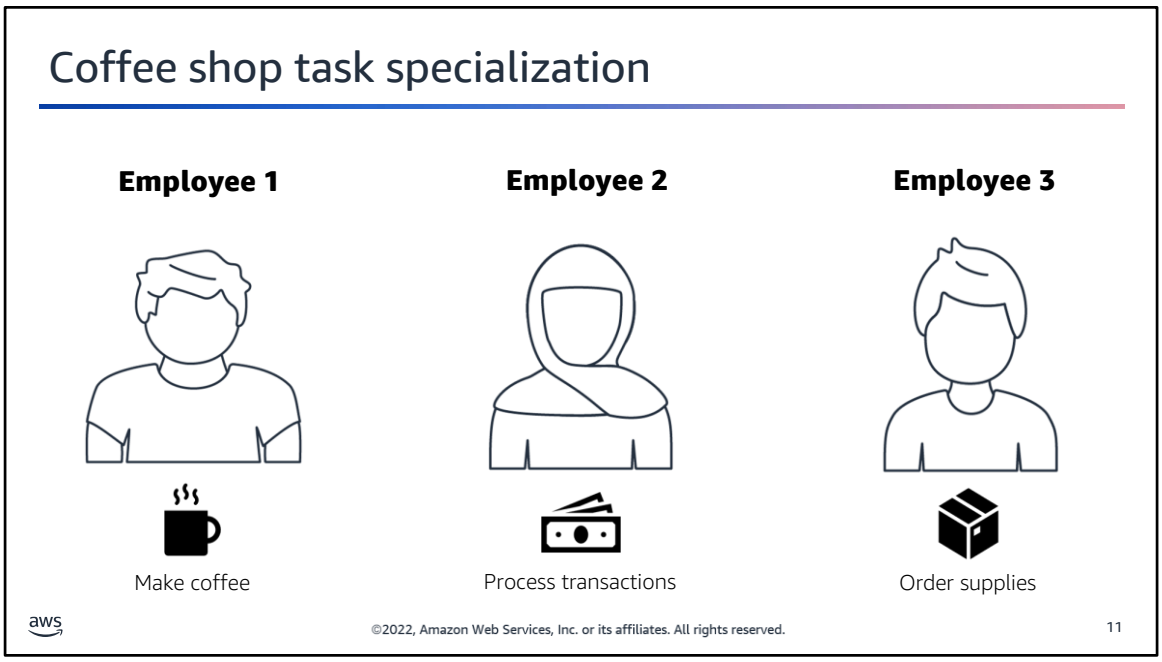
Here is a quick summary of how Amazon EC2 works.
First, you launch an instance. To do this, you choose a template with basic configurations for your instance. These configurations include the operating system, application server, or applications. You also choose the instance type, which is the specific hardware configuration of your instance.
As you prepare to launch an instance, you specify security settings to control the network traffic that can flow in and out of your instance. Later in this course, you will explore Amazon EC2 security features in greater detail.
Next, connect to the instance. You can connect to the instance in several ways. Your programs and applications have multiple methods to connect directly to the instance and exchange data. Users can also connect to the instance by logging in and accessing the computer desktop.
After you connect to the instance, you can use it. You can run commands to install software, add storage, copy and organize files, and more.



Amazon EC2 offers several instance types. This section focuses on what an instance type is and explores the various instance types that are available in Amazon EC2.



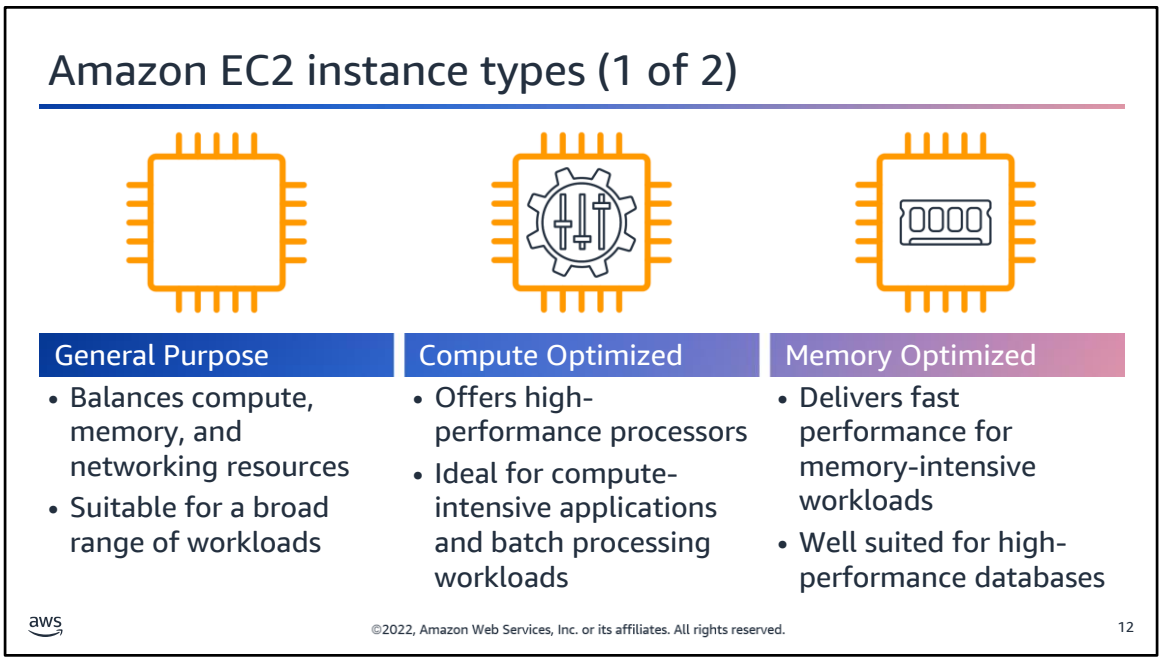
In a coffee shop, suppose that there is only one employee who does everything – makes coffee, processes transactions at the register, orders supplies, and so on. At each phase in the process, the customer ends up waiting. This would not be the most efficient use of resources or provide the best customer experience. Having several employees performing the same tasks would also not be efficient.



Different employees have different strengths, such as designing creative latte art, quickly completing payment transactions, or tracking inventory. To keep the coffee shop running efficiently, you could let your employees specialize and work in their areas of strength.

Now, think of the coffee shop employees as different types of Amazon EC2 instances. You can launch Amazon EC2 instances in your AWS environment to complete different tasks.

AWS provides a broad choice of instances. They can be general purpose or optimized for specific needs, such as high performance computing, big data, storage, and analytics.



When choosing an instance type, consider the specific needs of your workloads and applications. This might include requirements for compute, memory, or storage capabilities.

General purpose instances provide a balance of compute, memory, and networking resources. They can be used for a variety of workloads, such as application servers, gaming servers, backend servers for enterprise applications, and small and medium databases.

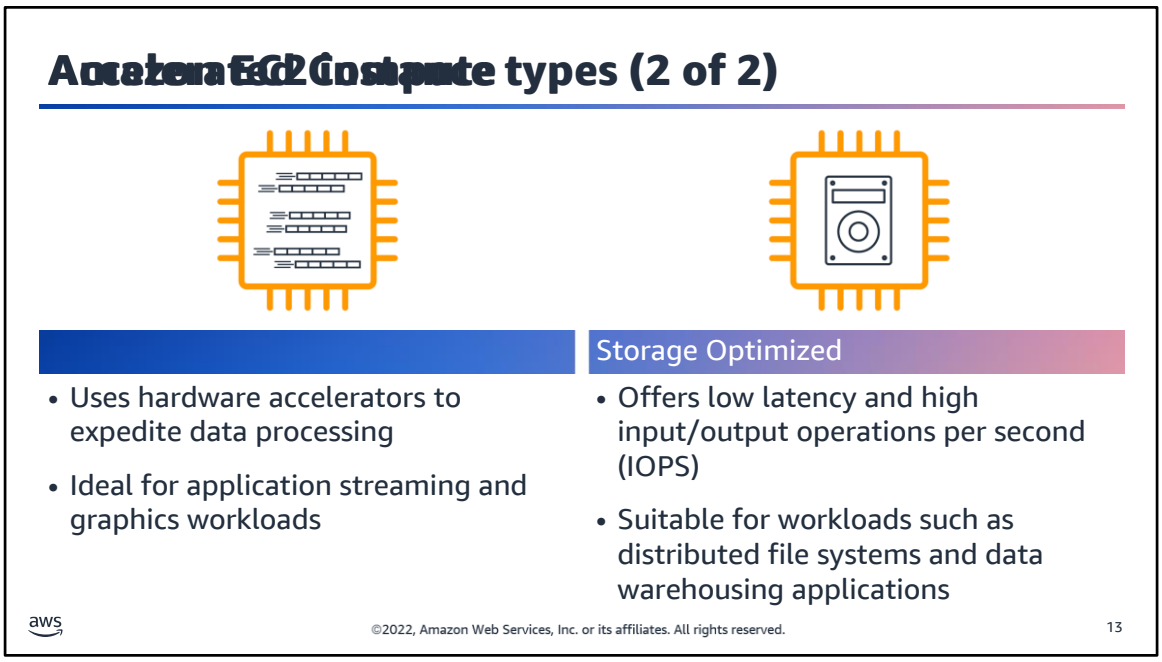
Suppose that you have an application in which the resource needs for compute, memory, and networking are roughly equivalent. You might consider running it on a general purpose instance because the application does not require optimization in any single resource area.

Compute optimized instances are ideal for compute-bound applications that benefit from high-performance processors. Like general purpose instances, compute optimized instances can be used for workloads such as web, application, and gaming servers.

However, the difference is that compute optimized applications are ideal for *high-performance* web servers, *compute-intensive* applications servers, and *dedicated* gaming servers. Compute optimized instances can also be used for batch processing workloads that require many transactions to be processed in a single group.

Memory optimized instances are designed to deliver fast performance for workloads that process large datasets in memory. In computing, memory is a temporary storage area. It holds all the data and instructions that a central processing unit (CPU) needs to be able to complete actions. Before a computer program or application can run, it is loaded from storage into memory. This preloading process gives the CPU direct access to the computer program.

Suppose that you have a workload that requires large amounts of data to be preloaded before an application is run. This might be a high-performance database or a workload that involves performing real-time processing of big unstructured data. In these types of use cases, consider using a memory optimized instance. Memory optimized instances allow you to run workloads with high memory needs and receive great performance.



Accelerated computing instances use hardware accelerators, or *coprocessors*, to perform some functions more efficiently than is possible in software running on CPUs. Examples of these functions include floating point number calculations, graphics processing, and data pattern matching.

In computing, a hardware accelerator is a component that can expedite data processing. Accelerated computing instances are ideal for workloads such as graphics applications, game streaming, and application streaming.

Storage optimized instances are designed for workloads that require high, sequential read and write access to large datasets on local storage. Examples of workloads suitable for storage optimized instances include distributed file systems, data warehousing applications, and high-frequency online transaction processing (OLTP) systems.

In computing, input/output operations per second (IOPS) is a metric that measures the performance of a storage device. It indicates how many different input or output operations a device can perform in one second. Storage optimized instances are designed to deliver tens of thousands of low-latency, random IOPS to applications.

You can think of input operations as data that is put into a system, such as records that are entered into a database. Output operations are data that is generated by a server. An example of output might be the analytics that are performed on the records in a database. If you have an application that has a high IOPS requirement, a storage optimized instance can potentially provide improved performance over other instance types that are not optimized for this kind of use case.

Reference

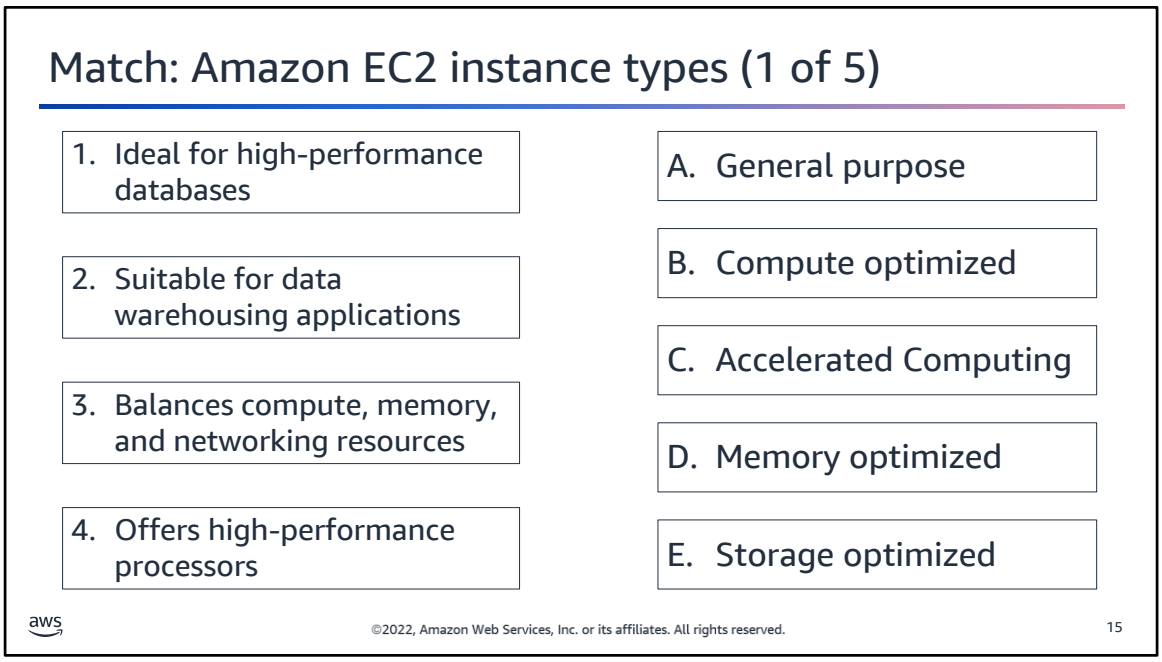
- For more information about Amazon EC2 instance types, review “Amazon EC2 Instance Types” at: https://aws.amazon.com/ec2/instance-types/

Interactive Demonstration 6

Creating an Amazon EC2 West Server

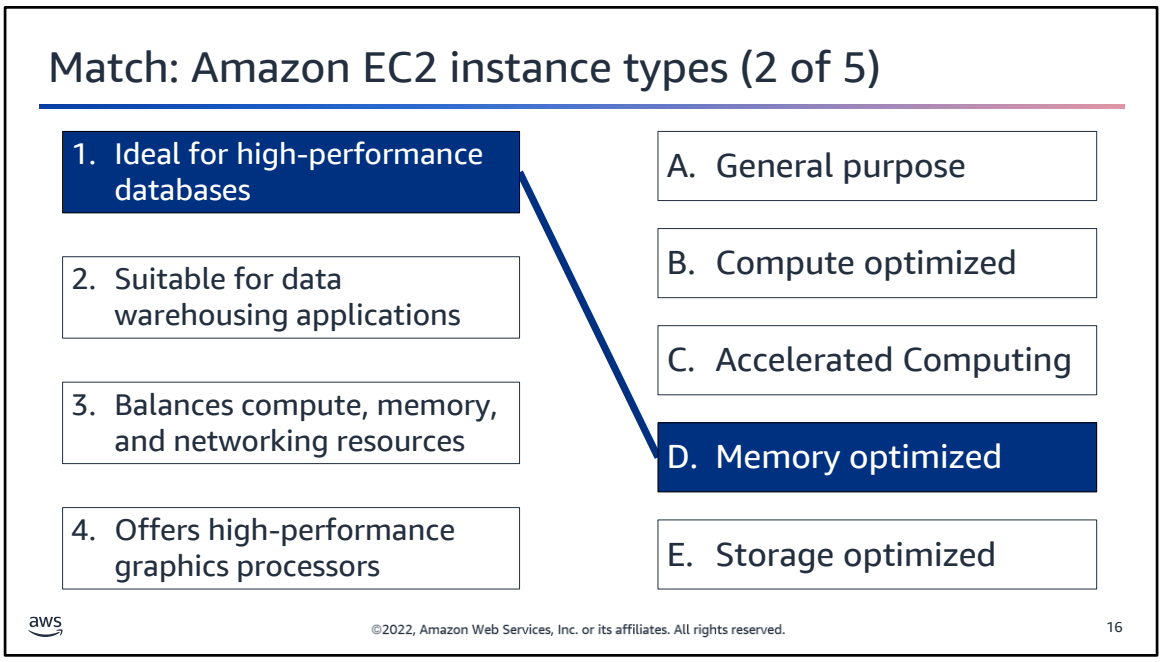
In this demonstration you will:

- Create and configure an Amazon EC2 instance.
- Configure the security group to allow inbound traffic.
- Visit the webpage installed using the User data script.

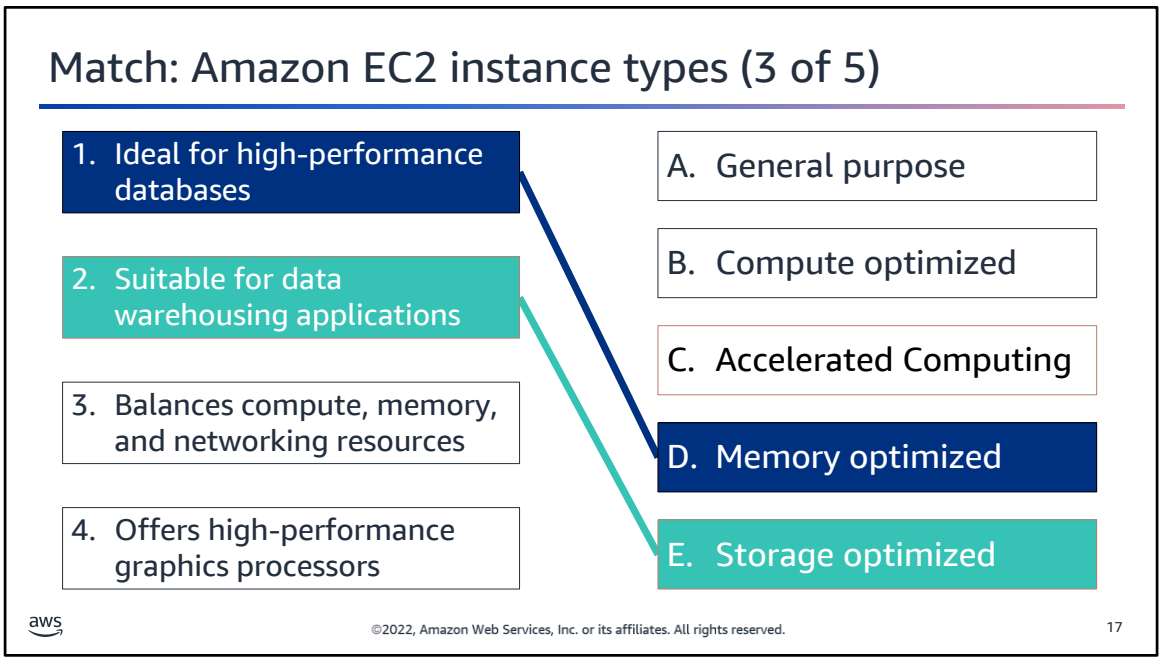


Before moving to the next topic, take a moment to review some descriptions and use cases for the Amazon EC2 instance types. This activity involves matching each option on the left to an Amazon EC2 instance type on the right.

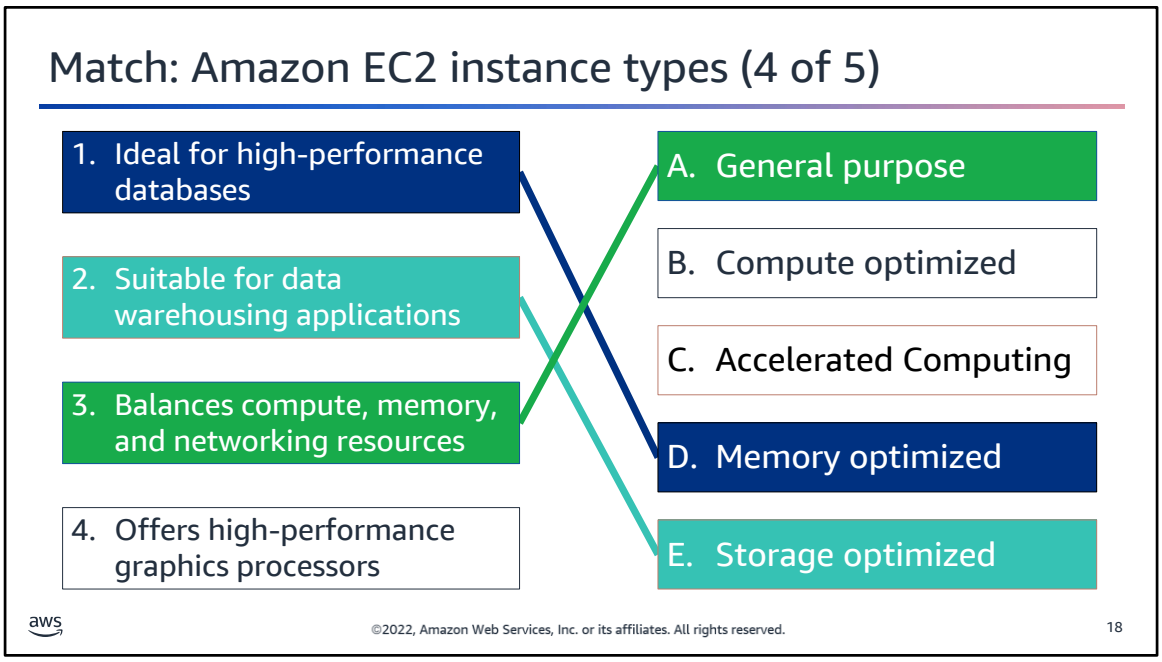
First, which Amazon EC2 instance type is ideal for high-performance databases?



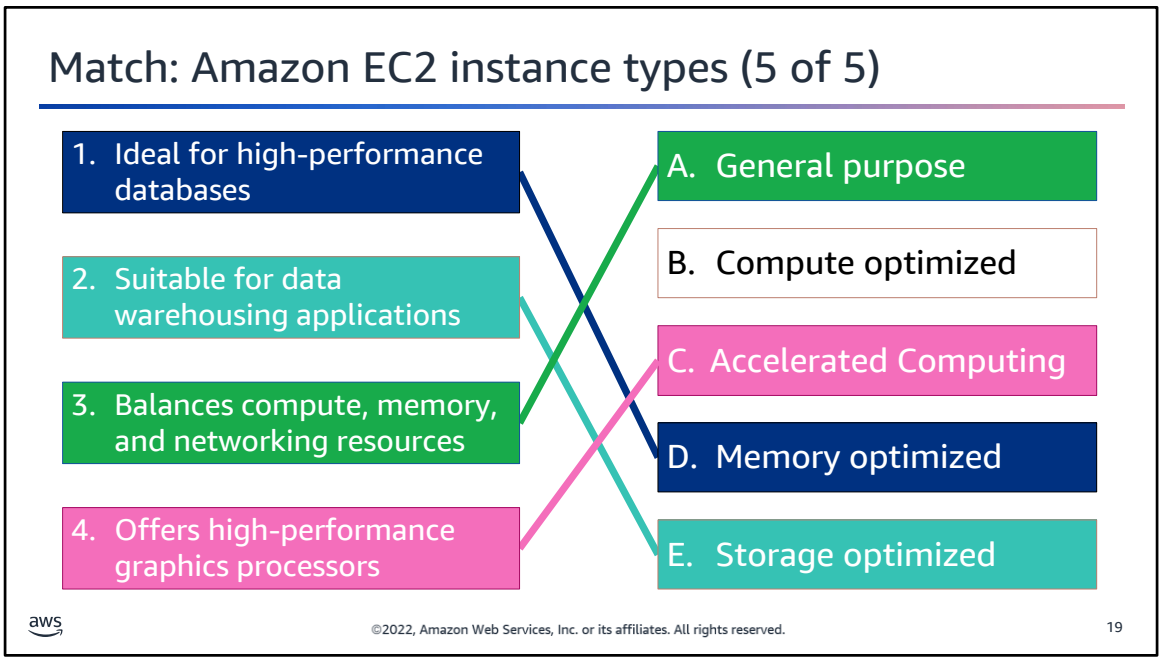
Memory optimized instances are ideal for high-performance databases.
Which Amazon EC2 instance type is suitable for data warehousing applications?



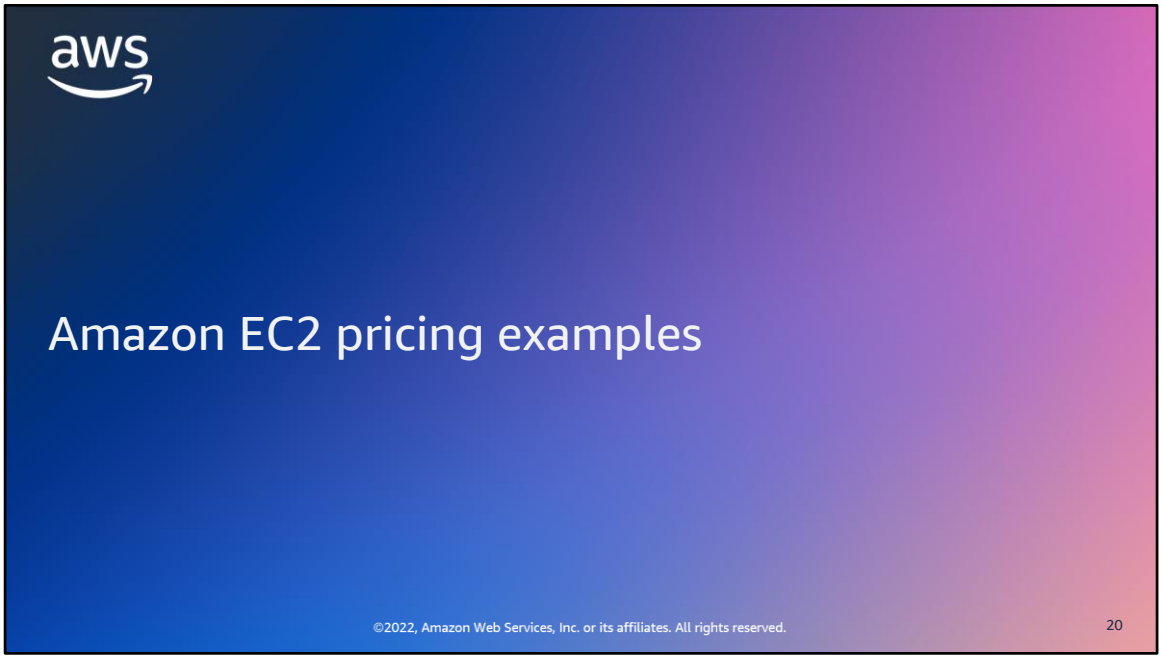
Storage optimized instances are suitable for data warehousing applications.
Which Amazon EC2 instance type balances compute, memory, and networking resources?



General purpose instances balance compute, memory, and networking resources.
Which Amazon EC2 instance type offers high-performance processors?

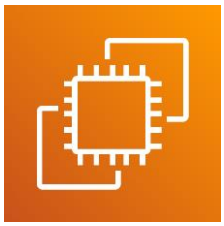


Accelerated compute instances offer high-performance graphics processors.
Next is a review of the Amazon EC2 pricing options.



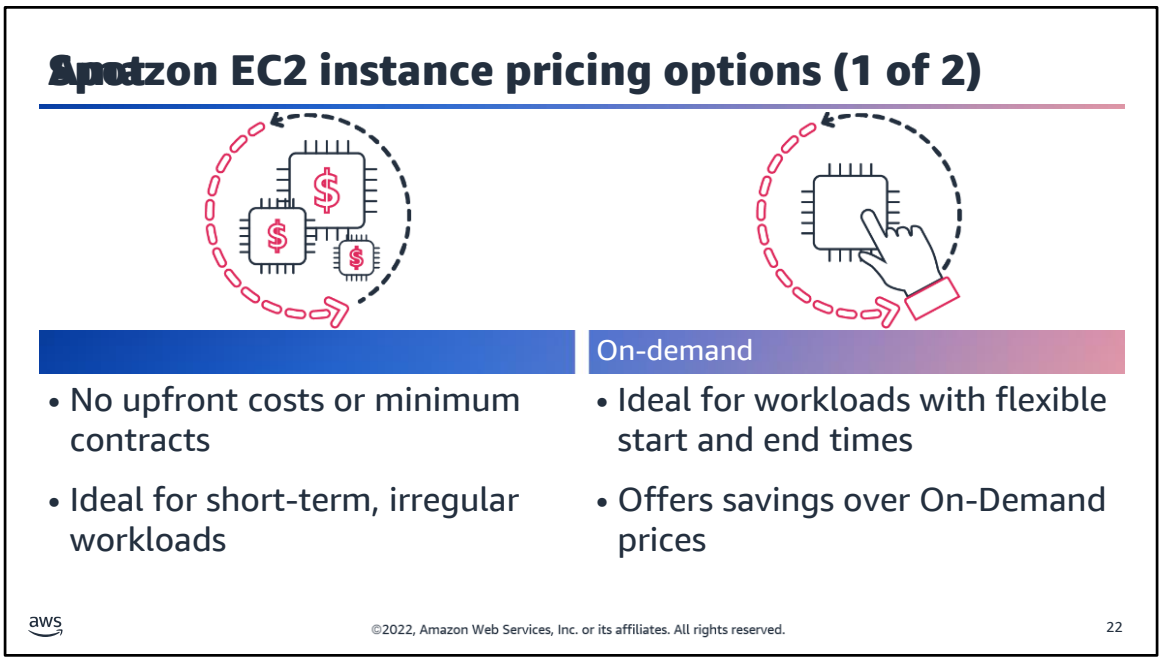
So far in this module, you have examined the Amazon EC2 instance types. This section describes the Amazon EC2 pricing.
Additional AWS pricing tools and services are explained later in this course.

Amazon EC2 pricing



- Pay only for the time that your On-Demand Instances run
- Reduce costs by using Spot Instances for recommended use cases
- Save by signing up for Compute Savings Plans
- Amazon EC2 pricing: https://aws.amazon.com/ec2/pricing

With Amazon EC2, you pay for only the compute time that you use while your On-Demand Instances are running.
For some workloads, you can significantly reduce Amazon EC2 costs by using Spot Instances. For example, suppose that you are running a batch processing job that can be interrupted. Using a Spot Instance would provide up to a 90% discount over the On-Demand Instance price.
You can find additional cost savings for Amazon EC2 by considering Savings Plans and Reserved Instances. Amazon EC2 pricing is based on the type of instances that you are running. For more information on Amazon EC2 pricing, review https://aws.amazon.com/ec2/pricing/.



With Amazon EC2, you pay only for the compute time that you use. Amazon EC2 offers a variety of pricing options for different use cases.

In the coffee shop example, suppose that the owners are experimenting with a new application that is still in the development and testing phases. The application does not yet need to run for long periods of time. However, when the application does run, it must do so without interruption so its performance can be accurately assessed.

On-Demand Instances are an excellent option to use for this type of short-term, irregular workload that cannot be interrupted. No upfront costs or minimum contracts apply. The instances run continuously until you stop them, and you pay for only the compute time you use.

Sample use cases for On-Demand Instances include developing and testing applications, and running applications that have unpredictable usage patterns. On-Demand Instances are not recommended for workloads that last a year or longer, because these workloads can experience greater cost savings through the use of Reserved Instances.

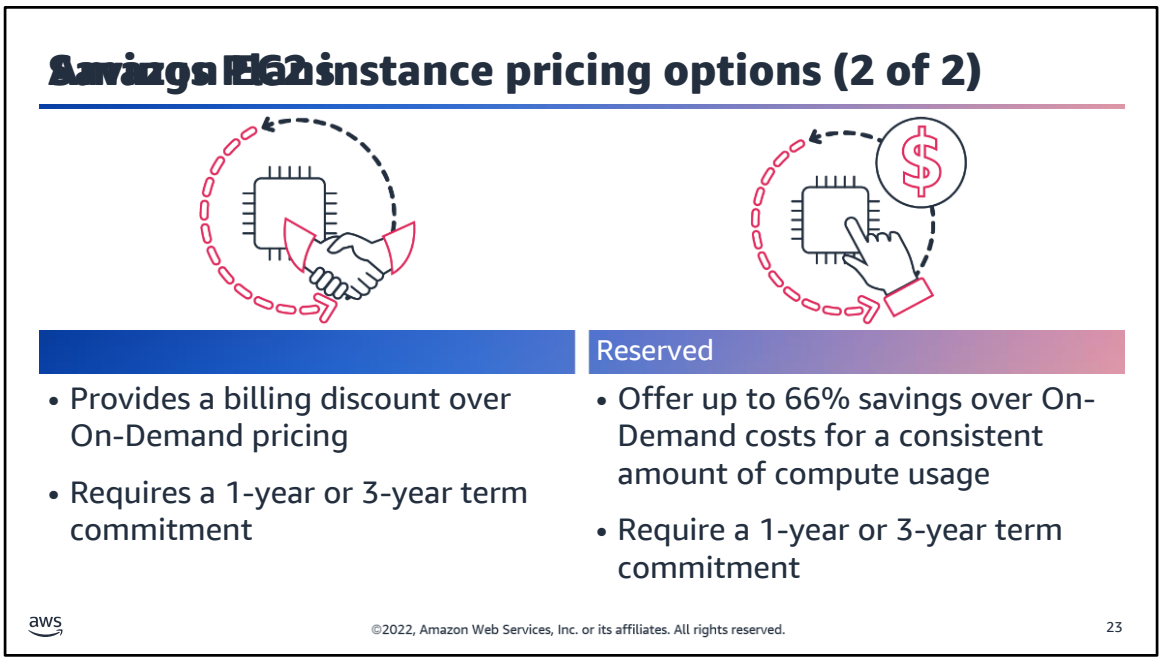
The owners of the coffee shop might also use an Amazon EC2 instance for their data processing, such as a batch workload that aggregates and analyzes customer survey data. Compared to other types of batch workloads in the coffee shop, such as daily financial processing, the survey data processing is not mission-critical. To save costs, the coffee shop owners decide to use a Spot Instance for their survey data processing.

Spot Instances are ideal for these types of workloads with flexible start and end times, or that can withstand interruptions. Spot Instances use unused EC2 computing capacity and offer you cost savings at up to 90 percent of On-Demand prices.

Suppose that you have a background processing job that can start and stop as needed (such as the customer survey data processing job). You want to start and stop the processing job without affecting the overall operations of your business. If you make a Spot request and Amazon EC2 capacity is available, your Spot

Instance launches. However, if you make a Spot request and Amazon EC2 capacity is unavailable, the request is not successful until capacity becomes available. The unavailable capacity might delay the launch of your background processing job.

After you have launched a Spot Instance, if capacity is no longer available or demand for Spot Instances increases, your instance might be interrupted. This might not pose any issues for your background processing job. However, in the earlier example of developing and testing applications, you would most likely want to avoid unexpected interruptions. Therefore, you should choose a different EC2 instance type that is more ideal for those tasks.



Suppose that the coffee shop owners have an application that will run continuously for at least a year. An example of this might be the main application that customers use for mobile ordering. The owners don't think that Spot Instances would be a good fit because of possible interruptions. They also considered On-Demand Instances, but the estimated price seems high for what they would pay for a year of compute time. This is an example of when to consider **Reserved Instances**.

Reserved Instances are a billing discount that is applied to the use of On-Demand Instances in your account. You can purchase Standard Reserved and Convertible Reserved Instances for a 1-year or 3-year term, and Scheduled Reserved Instances for a 1-year term. You realize greater cost savings with the 3-year option.

At the end of a Reserved Instance term, you can continue using the EC2 instance without interruption. However, you are charged On-Demand rates until you shut down the instance or purchase a new Reserved Instance that matches the instance attributes (instance type, Region, tenancy, and platform).

Next, suppose that the coffee shop owners want to save costs on their EC2 compute usage, but they want to have even more flexibility than what is possible with Reserved Instances. In this situation, they might consider **Compute Savings Plans**.

AWS offers Savings Plans for several compute services, including Amazon EC2. Amazon EC2 Savings Plans can help you reduce your compute costs by committing to a consistent amount of compute usage for a 1-year or 3-year term. This results in savings of up to 66 percent over On-Demand costs.

Any usage up to the commitment is charged at the discounted plan rate (for example, $10 an hour). Any usage beyond the commitment is charged at regular On-Demand rates.

Later in this course, you will review AWS Cost Explorer, a tool that can help you visualize, understand, and manage your AWS costs and usage over time. If you are considering your options for Savings Plans, AWS Cost Explorer can analyze your EC2 usage over the past 7, 30, and 60 days. AWS Cost Explorer also provides customized recommendations for Savings Plans. These recommendations estimate how much you could save on your monthly EC2 costs, based on previous EC2 usage and the hourly commitment amount in a 1-year or 3-year plan.

For more information on Savings Plans, please visit https://aws.amazon.com/savingsplans/pricing/.

Example: Amazon EC2 service charges

Elastic Compute Cloud		**$0.00**
US East (N. Virginia)		**$0.00**
Amazon Elastic Compute Cloud running Linux/UNIX		$0.00
$0.00 per Linux t2.micro instance-hour (or partial hour) under monthly free tier	106.512 Hrs	$0.00
EBS		$0.00
$0.00 per GB-month of General Purpose (SSD) provisioned storage under monthly free tier	11.294 GB-Mo	$0.00
Elastic Load Balancing - Application		$0.00
$0.00 per Application LoadBalancer-hour (or partial hour) under monthly free tier	268.000 Hrs	$0.00

The service charges in this example include details for the following items:

- Each Amazon EC2 instance type in use
- Amount of Amazon Elastic Block Store (Amazon EBS) storage space provisioned
- Length of time Elastic Load Balancing was used

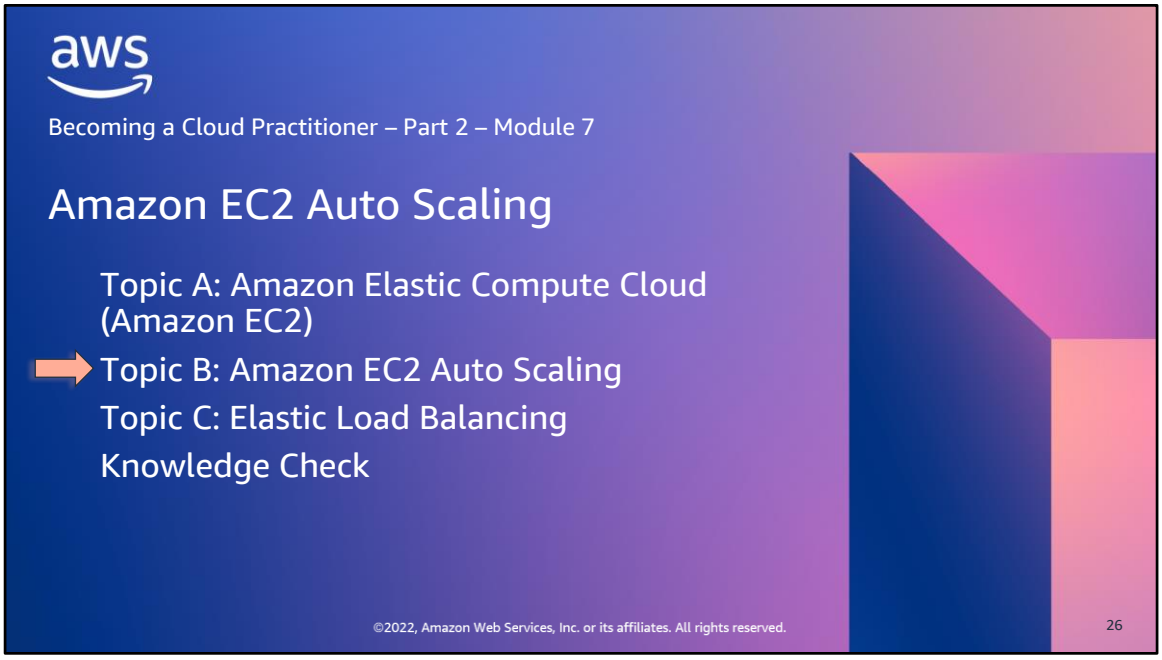
In this example, all the usage amounts are under the thresholds in the AWS Free Tier, so the account owner does not have to pay for any Amazon EC2 usage in this month.

Interactive Demonstration 7

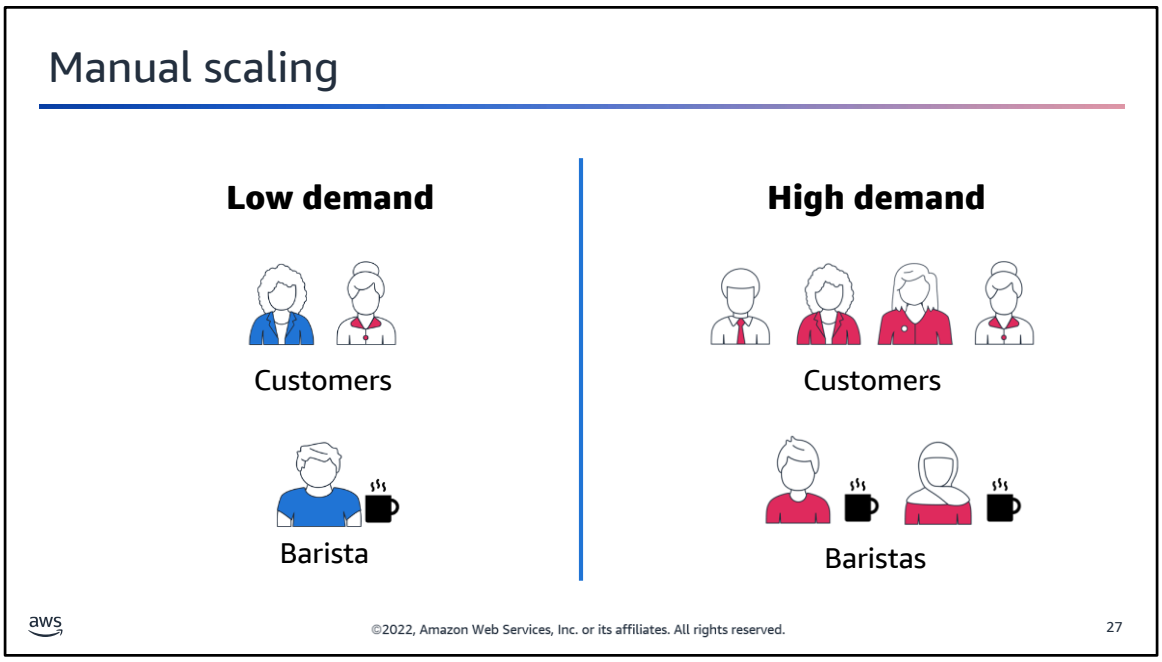
Viewing EC2 instance pricing

In this demonstration you will:

- View the pricing associated with the Coffee Shop web server.



This topic explores Amazon EC2 Auto Scaling.



Suppose that in the coffee shop, a barista is assigned to work at the register. When the coffee shop is in a period of low demand, the barista can readily manage their workload.

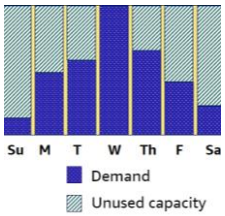
Now, suppose that the coffee shop is open during its busiest season of the year. Because of the increased demand, the barista feels overwhelmed by the increased workload.

The barista asks the manager for additional assistance, and the manager assigns another barista to help. When the workload decreases, the second barista can stop working at the register. This process is an example of *manual* scaling.

Scalability involves beginning with only the resources you need and designing your architecture to scale automatically in and out in response to changing demands. As a result, you pay for only the resources you use. You don’t have to worry about a lack of computing capacity to meet your customers’ needs.

What if you want scaling to happen automatically? The AWS service that provides this functionality for Amazon EC2 instances is **Amazon EC2 Auto Scaling**.

Amazon EC2 Auto Scaling (1 of 2)



- Scale capacity as computing requirements change
- Use dynamic scaling and predictive scaling

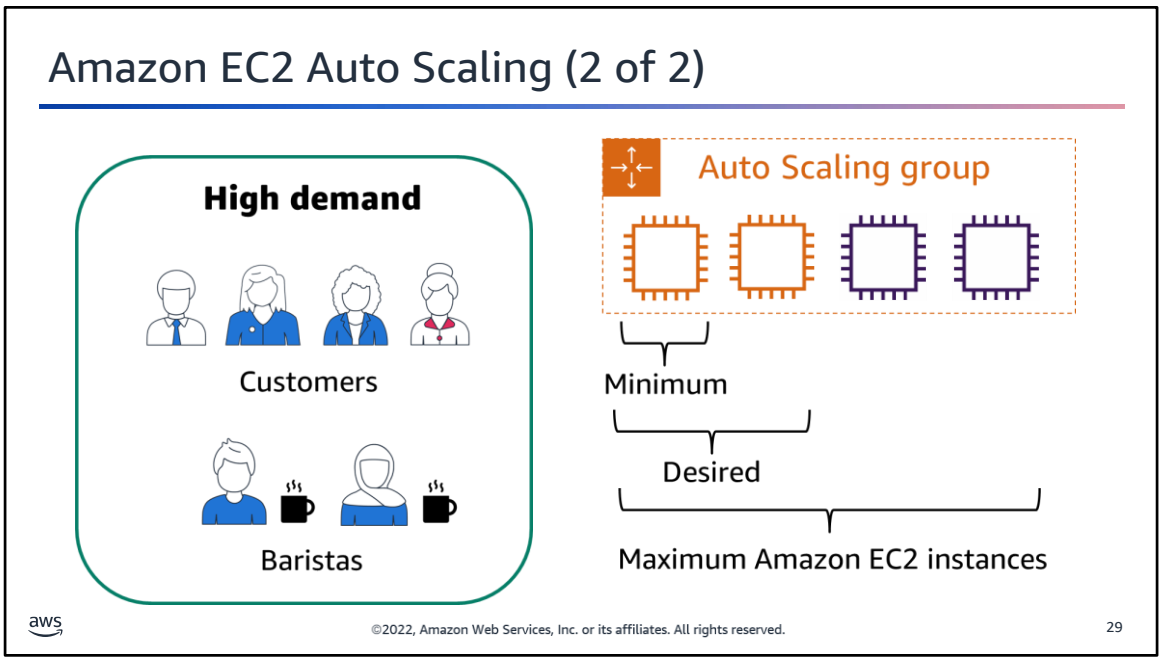
aws

28

Have you ever tried to access a website that wouldn't load and it kept timing out? The website might have been receiving more requests than it was able to handle. This is similar to the experience of waiting in a long line at a coffee shop, when there is only one barista present to take orders from customers.

Amazon EC2 Auto Scaling can help you automatically add or remove Amazon EC2 instances in response to changing application demand. By automatically scaling your instances in and out as needed, you can maintain a greater sense of application availability.

With Amazon EC2 Auto Scaling, you can use two approaches – dynamic scaling and predictive scaling. *Dynamic scaling* responds to changing demand. *Predictive scaling* automatically schedules the right number of Amazon EC2 instances based on predicted demand. To scale faster, you can use dynamic scaling and predictive scaling together.



In the cloud, computing power is a programmatic resource, so you can take a more flexible approach to the issue of scaling. By adding Amazon EC2 Auto Scaling to an application, you can add new instances to the application when necessary and remove them when no longer needed.

Suppose that you are preparing to run an application on Amazon EC2 instances. When configuring the size of your Auto Scaling group, you might set the minimum number of Amazon EC2 instances at one. This means that at all times, at least one Amazon EC2 instance must be running.

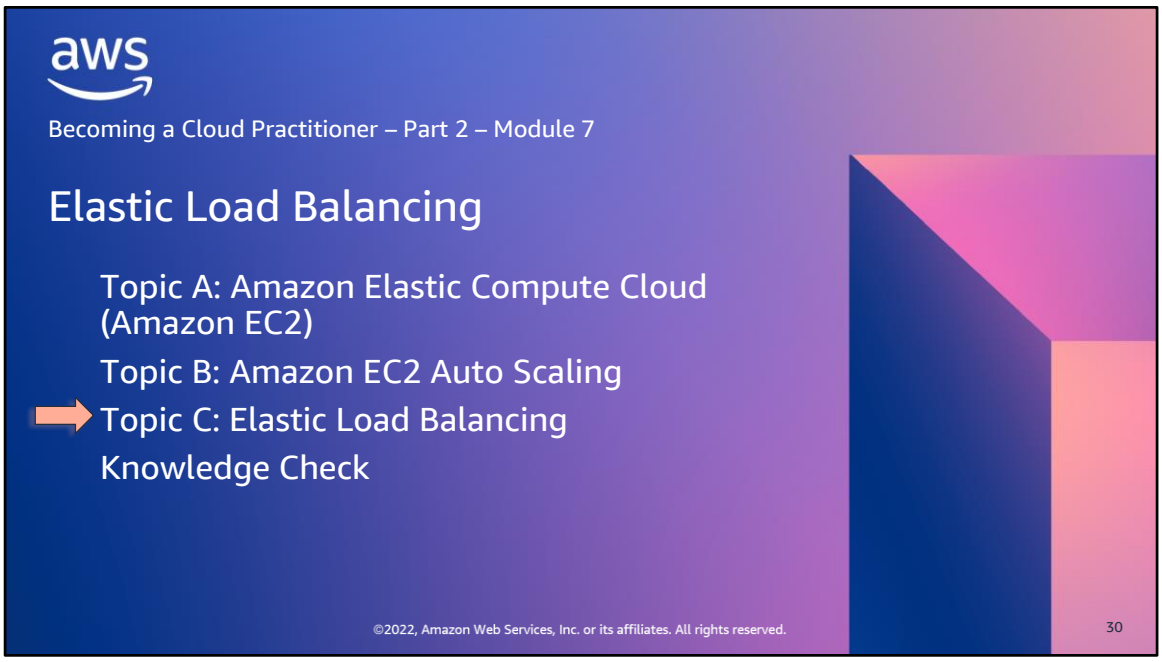
When you create an Auto Scaling group, you can set the minimum number of Amazon EC2 instances.

The **minimum capacity** is the number of Amazon EC2 instances that launch immediately after you create the Auto Scaling group. In this example, the Auto Scaling group has a minimum capacity of one Amazon EC2 instance.

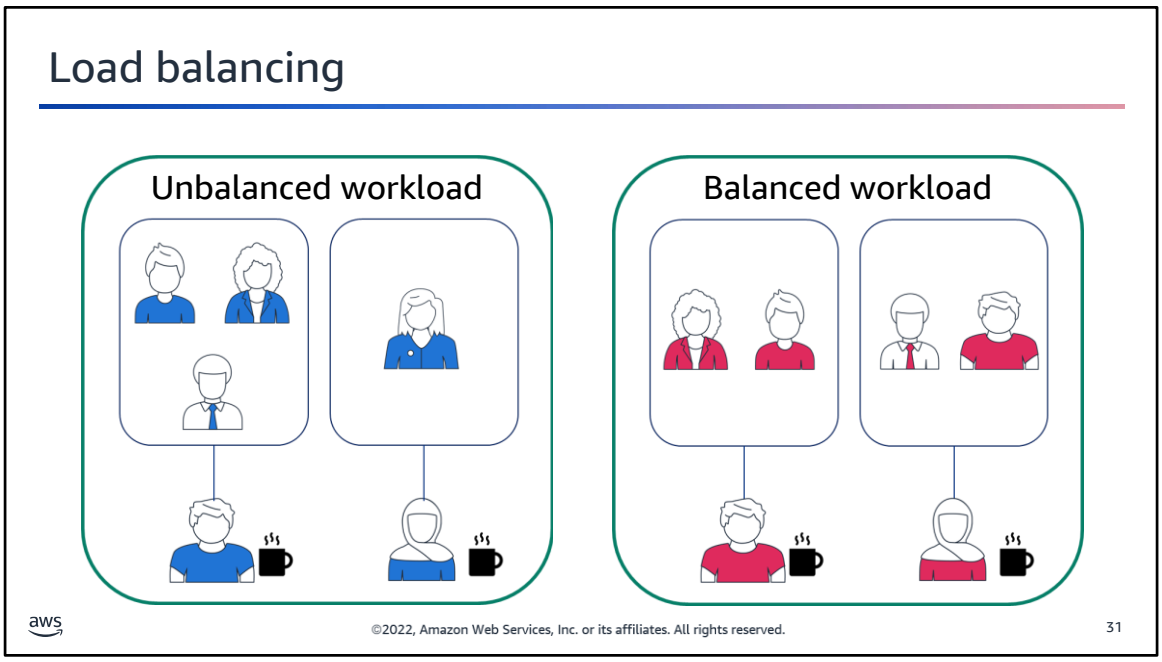
Next, you can set the **desired capacity** at two Amazon EC2 instances, even though your application needs a minimum of a single Amazon EC2 instance to run. If you do not specify the desired number of Amazon EC2 instances in an Auto Scaling group, the desired capacity defaults to your minimum capacity.

The third configuration that you can set in an Auto Scaling group is the **maximum capacity**. For example, you might configure the Auto Scaling group to scale out in response to increased demand, but only to a maximum of four Amazon EC2 instances.

Because Amazon EC2 Auto Scaling uses Amazon EC2 instances, you pay for only the instances you use, when you use them. You now have a cost-effective architecture that provides the best customer experience while reducing expenses.



In this topic, you will learn about Elastic Load Balancing and how it is used to provide high performance.

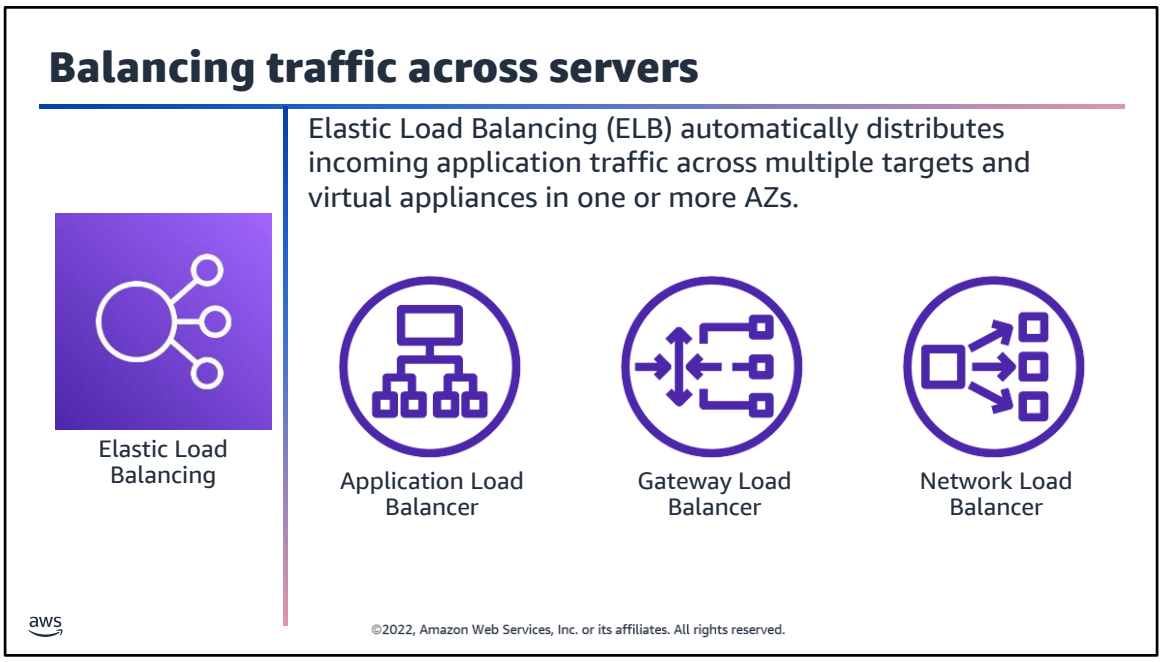


Using the previous example in the coffee shop, suppose that only one barista is doing the majority of the work and is overworked, while the other barista is underworked.

To prevent any barista from becoming overwhelmed, the workload can be redistributed so that both baristas will serve the same number of customers.

Spreading workloads improves the performance of your applications by preventing any single resource from having to handle the full workload on its own. In this example, the number of customers remains the same, but balancing the workload evenly distributes the customers across the two baristas.

With **Elastic Load Balancing** in AWS, the size of a workload remains the same, but the workload is balanced by evenly distributing it across Amazon EC2 instances.



|Instructor Notes

|

|Student Notes

Elastic Load Balancing is the AWS service that automatically distributes incoming application traffic across multiple resources, such as Amazon EC2 instances.

A load balancer acts as a single point of contact for all incoming web traffic to your Auto Scaling group. This means that as EC2 instances are added or removed in response to the amount of incoming traffic, these requests are routed to the load balancer first. Then, they are spread across multiple resources that will handle them. For example, if your application has been configured to have multiple EC2 instances, Elastic Load Balancing distributes the workload across the multiple instances so that no single instance has to carry the bulk of it.

Although Elastic Load Balancing and Amazon EC2 Auto Scaling are separate services, they work together to help ensure that applications running in Amazon EC2 can provide high performance and availability.

There are three types of load balancers that are used with AWS architectures.

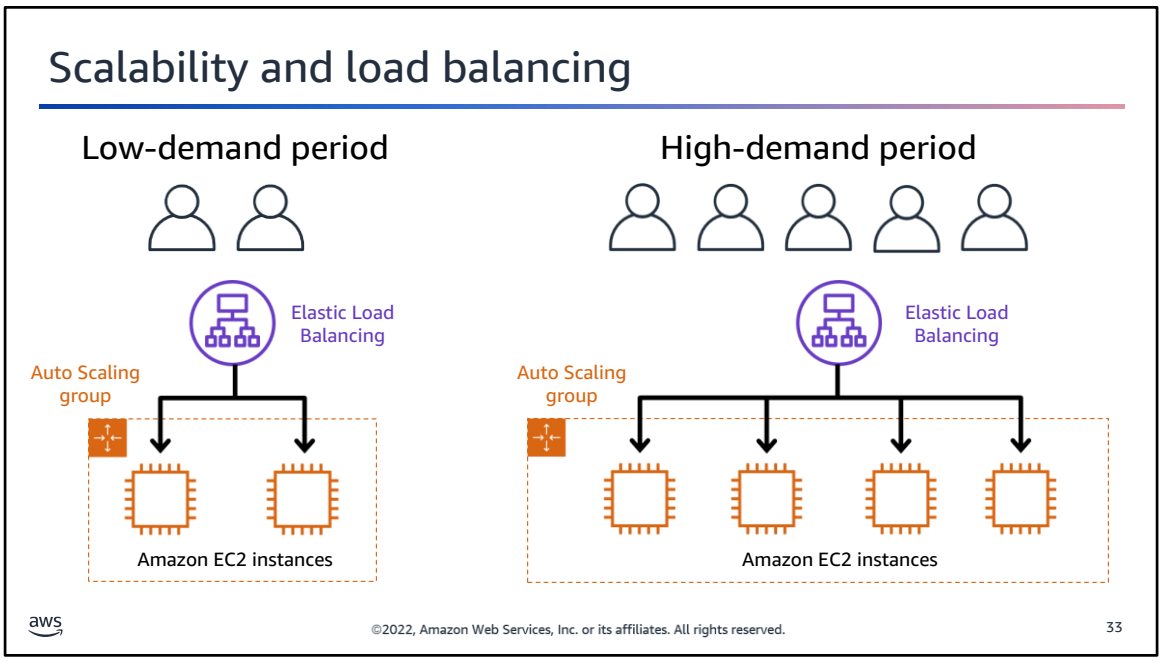
An **Application Load Balancer** functions at the application layer, the seventh layer of the Open Systems Interconnection (OSI) model. After the load balancer receives a request, it evaluates the listener rules in priority order to determine which rule to apply, and then selects a target from the target group for the rule action. You can configure listener rules to route requests to different target groups based on the content of the application traffic. Routing is performed independently for each target group, even when a target is registered with multiple target groups. You can configure the routing algorithm used at the target group level. The default routing algorithm is round robin; alternatively, you can specify the least outstanding requests routing algorithm.

Gateway Load Balancer helps you easily deploy, scale, and manage your third-party virtual appliances. It gives you one gateway for distributing traffic across multiple virtual appliances while scaling them up or down, based on demand.

A **Network Load Balancer** functions at the fourth layer of the Open Systems Interconnection (OSI) model. It can handle millions of requests per second. After the load balancer receives a connection request, it selects a target from the target group for the default rule. It attempts to open a TCP connection to the selected target on the port specified in the listener configuration.

When you enable an Availability Zone for the load balancer, Elastic Load Balancing creates a load balancer node

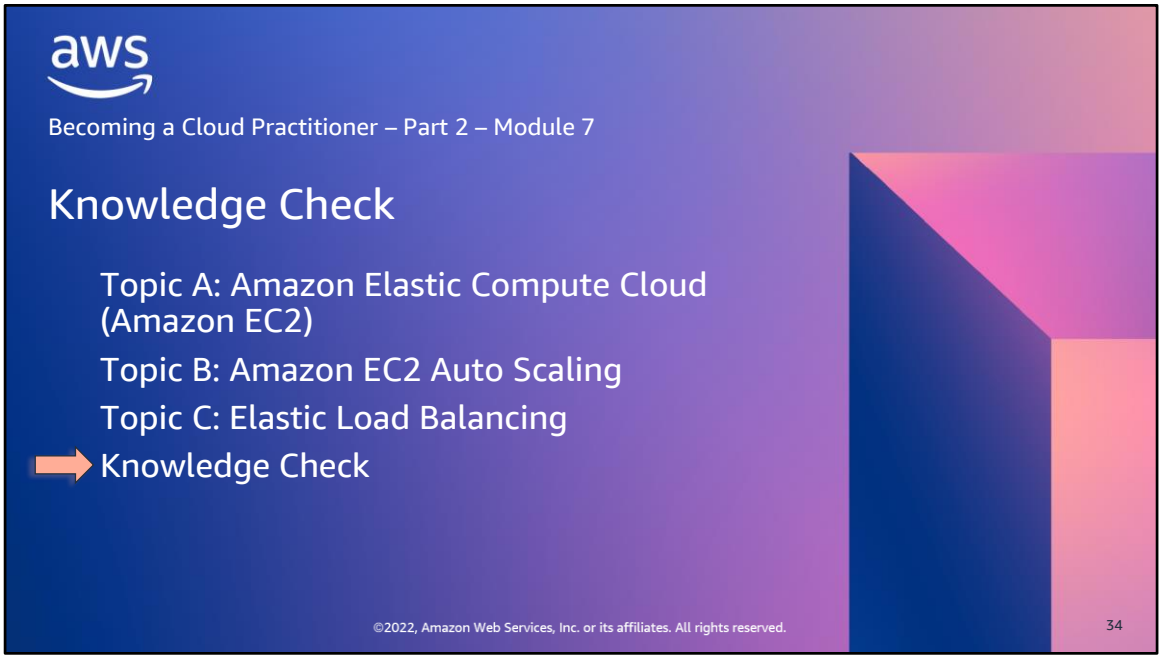
in the Availability Zone. By default, each load balancer node distributes traffic across the registered targets in its Availability Zone only. If you enable cross-zone load balancing, each load balancer node distributes traffic across the registered targets in all enabled Availability Zones.



Here's an example of how Elastic Load Balancing works. Suppose that a few customers have come to the coffee shop and are ready to place their orders. If only a few registers are open, this matches the demand of customers who need to be served. The coffee shop will be less likely to have open registers with no customers. In this example, you can think of the registers as Amazon EC2 instances.

Throughout the day, as the number of customers increases, the coffee shop opens more registers to accommodate them. In the diagram, this is represented by the Auto Scaling group.

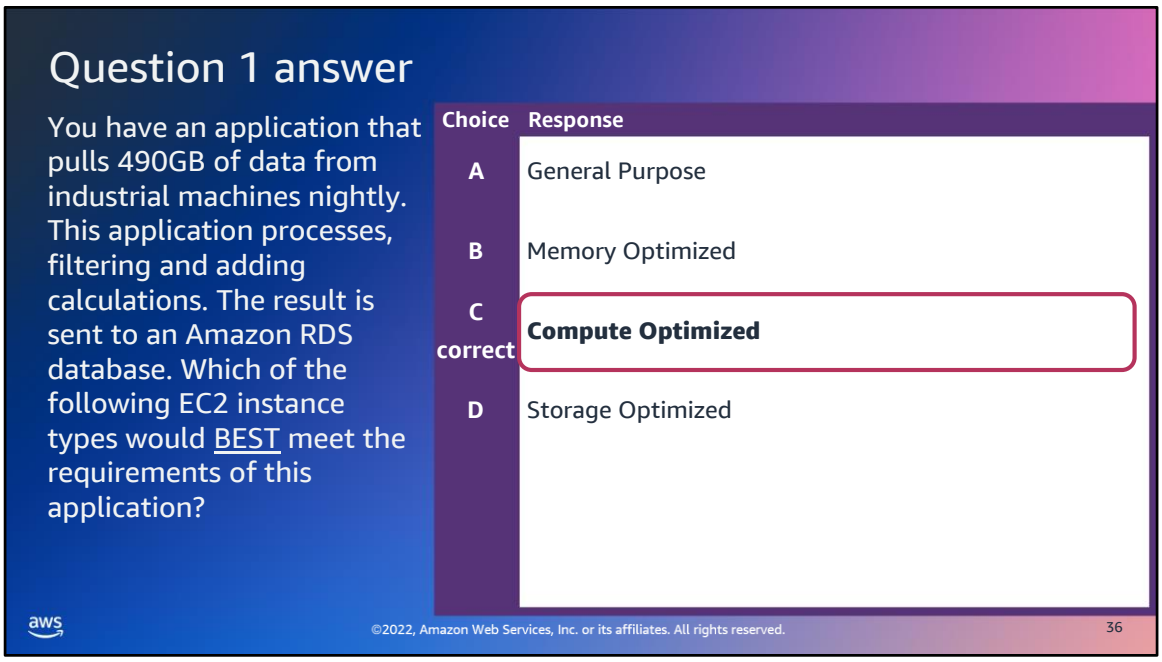
Additionally, a coffee shop employee directs customers to the most appropriate register so that the number of requests can be evenly distributed across the open registers. You can think of this coffee shop employee as a load balancer.



Question 1

You have an application that pulls 490GB of data from industrial machines nightly. This application processes, filtering and adding calculations. The result is sent to an Amazon RDS database. Which of the following EC2 instance types would BEST meet the requirements of this application?

Choice	Response
A	General Purpose
B	Memory Optimized
C	Compute Optimized
D	Storage Optimized

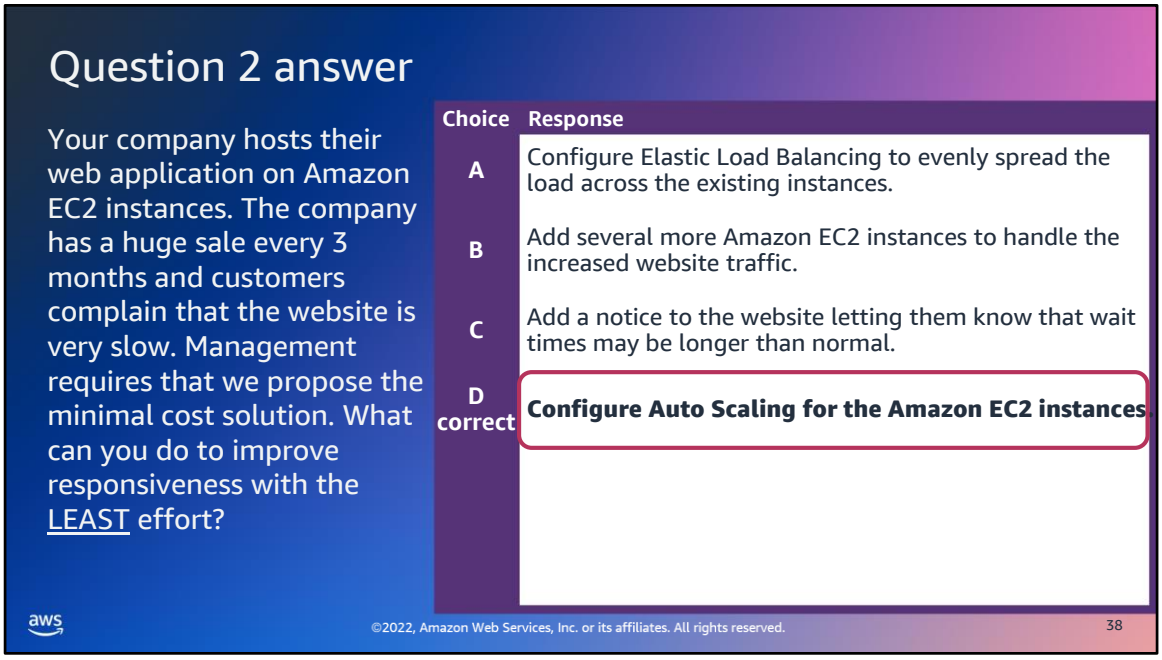


The correct answer is **Compute Optimized**.
This workload is a batch workload that runs nightly. It will benefit from an instance that is optimized for heavy batch compute workloads. The Accelerated Compute instance type would also work well for this scenario.
The other response options are incorrect because:
General purpose could perform the work but not as efficiently. Memory optimized would not perform much better than a General Purpose for this type of workload. Storage optimized, like Memory Optimized, would not perform much better than General Purpose.

Question 2

Your company hosts their web application on Amazon EC2 instances. The company has a huge sale every 3 months and customers complain that the website is very slow. Management requires that we propose the minimal cost solution. What can you do to improve responsiveness with the LEAST effort?

Choice	Response
A	Configure Elastic Load Balancing to evenly spread the load across the existing instances.
B	Add several more Amazon EC2 instances to handle the increased website traffic.
C	Add a notice to the website letting them know that wait times may be longer than normal.
D	Configure Auto Scaling for the Amazon EC2 instances.



The correct answer is **Configure Auto Scaling for the Amazon EC2 instances.** This allows your web application to dynamically grow and shrink as the demands change.
The other response options are incorrect because:
Elastic Load Balancing is not the right answer because it is responsible for distributing traffic against existing resources but does not increase resource availability.
Adding several more Amazon EC2 instances would work but then you would be paying for those additional instances even when they are not needed.
Adding a notice to the website does not address the problem.

Module 7 summary

In this module, you learned how to:

- Describe Amazon EC2 benefits, instance types, and billing options
- Summarize Amazon EC2 Auto Scaling benefits
- Summarize Elastic Load Balancing benefits