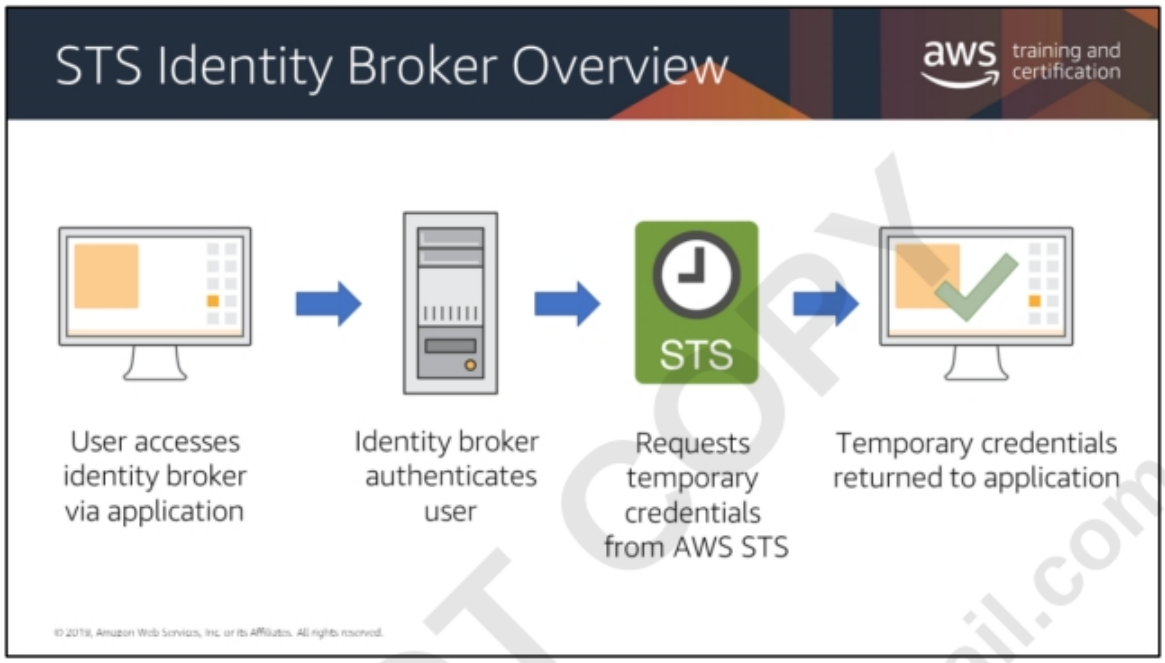
In the case of AssumeRole, it is possible to map calls back to the originating AWS service or to the account of the originating user.

The userIdentity section of the JSON data in the CloudTrail log entry contains the information needed to map the AssumeRole request with a specific federated user.

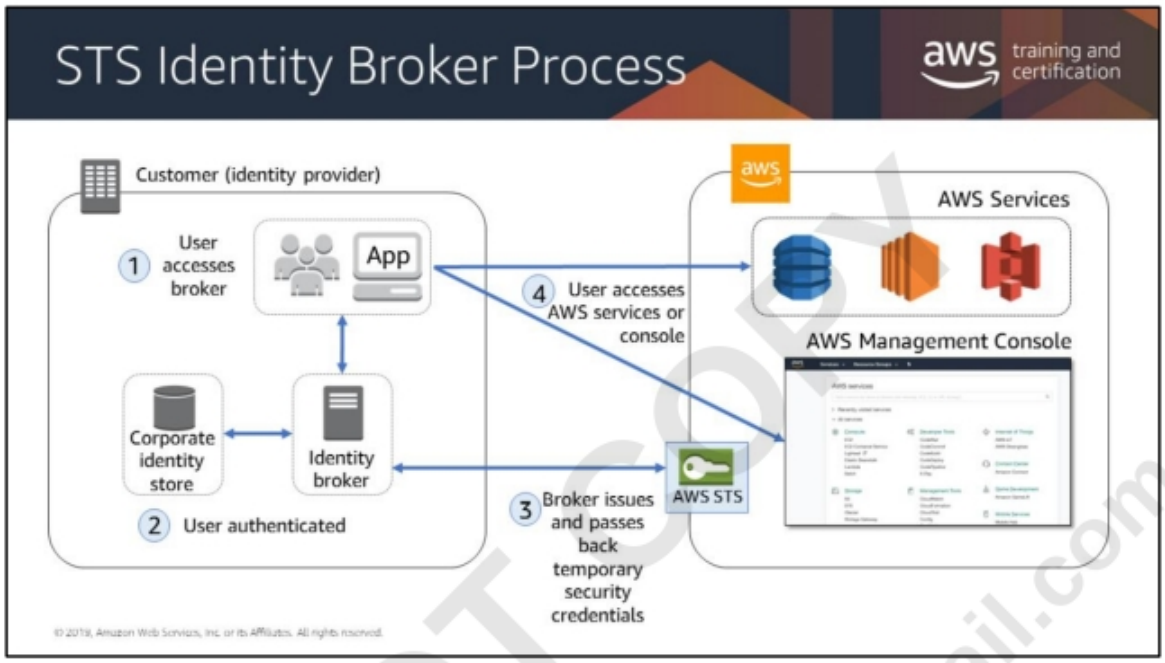
For more information, see:

- https://docs.aws.amazon.com/STS/latest/APIReference/Welcome.html
- https://docs.aws.amazon.com/IAM/latest/UserGuide/cloudtrail-integration.html

/



There are four primary steps in using AWS STS to create temporary credentials for an application that's using a third-party authentication service.



In this scenario:

- The identity broker application has permissions to access the AWS STS API to create temporary security credentials.
- The identity broker application can verify that employees are authenticated within the existing authentication system.
- Users get a temporary URL that gives them access to the console (which is referred to as single sign-on).

IAM user groups from another AWS account:
You can establish cross-account access by using IAM roles. In the trusting account, the resource must be in a service that supports roles.

IAM users within the current account:
For mission-critical permissions that IAM users might not frequently use, you can separate those permissions from their normal day-to-day permissions by using roles. Users have to actively assume a role, which can prevent them from accidentally performing disruptive actions.

For example, you might have Amazon EC2 instances that are critical to your organization. Instead of directly granting administrators permission to terminate the instances, you can create a role with those privileges and allow administrators to assume the role.

Administrators won't have permission to terminate those instances; they must first assume a role. By using a role, administrators must take an additional step to assume a role before they can stop an instance that's critical to your organization.

Third Parties:
When third parties require access to your organization's AWS resources, you can use roles to delegate API access to them. For example, a third party might provide a service for managing your AWS resources. With IAM roles, you can grant these third parties access to your AWS resources without sharing your AWS security credentials. Instead, they can assume a role that you created to access your AWS resources.

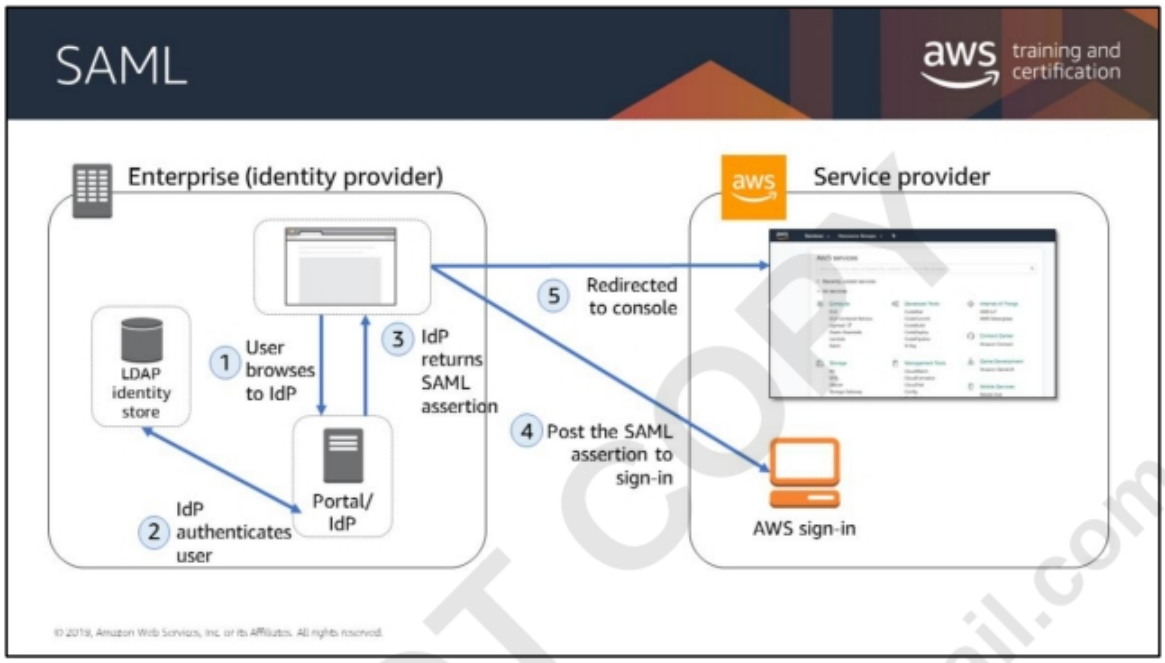
Third parties must provide you the following information for you to create a role that they can assume:

- The AWS account ID that the third party's IAM users use to assume your role. You specify their AWS account ID when you define the trusted entity for the role.
- An external ID that the third party can associate with your role. You specify the ID that was provided by the third party when you define the trusted entity for the role.
- The permissions that the third party requires in order to work with your AWS resources. You specify these permissions when defining the role's permission policy. This policy defines what actions they can take and what resources they can access.
- After you create the role, you must share the role's Amazon Resource Name (ARN) with the third party. They require your role's ARN in order to assume the role.

Identity Broker:

- Used to query AWS STS
- Determines user from a web request
- Uses AWS credentials (service account) to authenticate to AWS
- Issues temporary security credentials to access AWS APIs (through AWS STS)
- AWS permissions are configured by the administrator of the identity broker
- Configurable timeout: 1-36 hours
- For more information (including sample IIS authentication proxy C# code), see http://aws.amazon.com/code/1288653099190193.

/

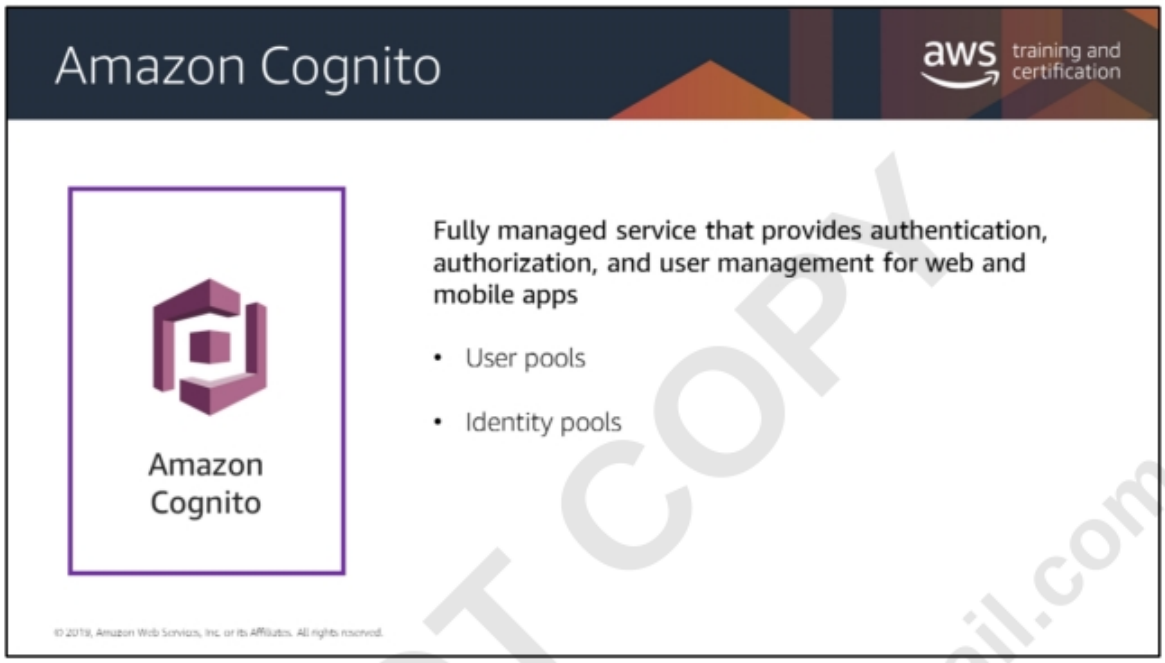


From the user's perspective, the process happens transparently. The user starts at your organization's internal portal and ends up at the AWS Management Console, without ever having to supply any AWS credentials.

1. **User browses to a URL.** A user in your organization browses to an internal portal in your network. The portal also functions as the IdP that handles the SAML trust between your organization and AWS.
2. **User is authenticated**. The identity provider (IdP) authenticates the user's identity against AD.
3. **User receives authentication response**. The client receives a SAML assertion (in the form of authentication response) from the IdP.
4. **Client posts to sign-in passing AuthN**. The client posts the SAML assertion to the new AWS sign-in endpoint. Behind the scenes, sign-in uses the AssumeRoleWithSAML API to request temporary security credentials and construct a sign-in URL.
5. **Client is redirected to the AWS Management Console**. The user's browser receives the sign-in URL and is redirected to the AWS Management Console.

For more information, see:

- https://aws.amazon.com/blogs/security/enabling-federation-to-aws-using-windows-active-directory-adfs-and-saml-2-0/
- https://aws.amazon.com/blogs/security/aws-federated-authentication-with-active-directory-federation-services-ad-fs/



Amazon Cognito is a fully-managed service that provides authentication, authorization, and user management for web and mobile apps. Users can sign in directly with a user name and password or through a third party such as Facebook, Amazon, or Google.

The two main components of Amazon Cognito are *user pools* and *identity pools*.

- **User pools** are user directories that provide sign-up and sign-in options for your app users.
- **Identity pools** enable you to grant your users access to other AWS services. Identity pools and user pools can be used separately or together.

A user pool is a user directory in Amazon Cognito. With a user pool, users can sign-in to a web or mobile app through Amazon Cognito, or federate through a third-party identity provider (IdP).

All members of the user pool have a directory profile that can be accessed through an SDK.

/

User pools provide:

- Sign-up and sign-in services
- A built-in, customizable web UI to sign in users
- Social sign-in with Facebook, Google, and Login with Amazon, and through SAML and OIDC identity providers from your user pool
- User directory management and user profiles
- Security features such as multi-factor authentication (MFA), checks for compromised credentials, account takeover protection, and phone and email verification
- Customized workflows and user migration through AWS Lambda triggers

For more information about user pools, see https://docs.aws.amazon.com/cognito/latest/developerguide/getting-started-with-cognito-user-pools.html

Amazon Cognito identity pools enable the creation of unique identities and permission assignment for users.

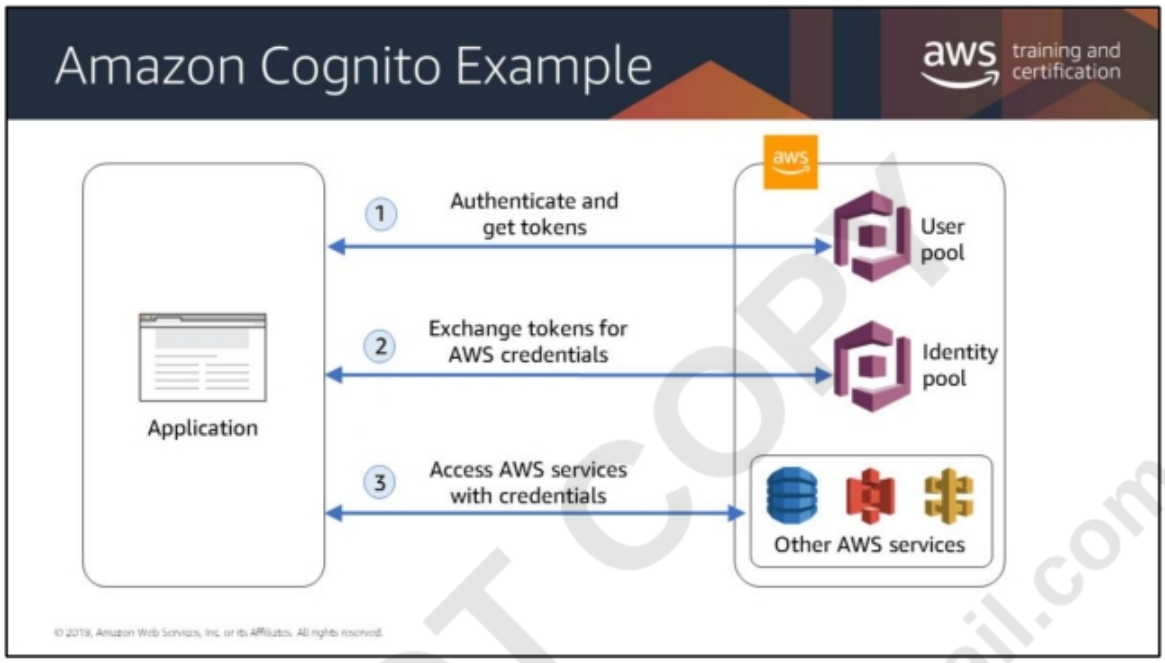
With an identity pool, users can obtain temporary AWS credentials to access AWS services or access resources through Amazon API Gateway.

Identity pools provide temporary AWS credentials for users who are guests (unauthenticated/anonymous) as well as the following identity providers:

- Amazon Cognito user pools
- Social sign-in with Facebook, Google, and Login with Amazon
- OpenID Connect (OIDC) providers
- SAML identity providers
- Developer authenticated identities

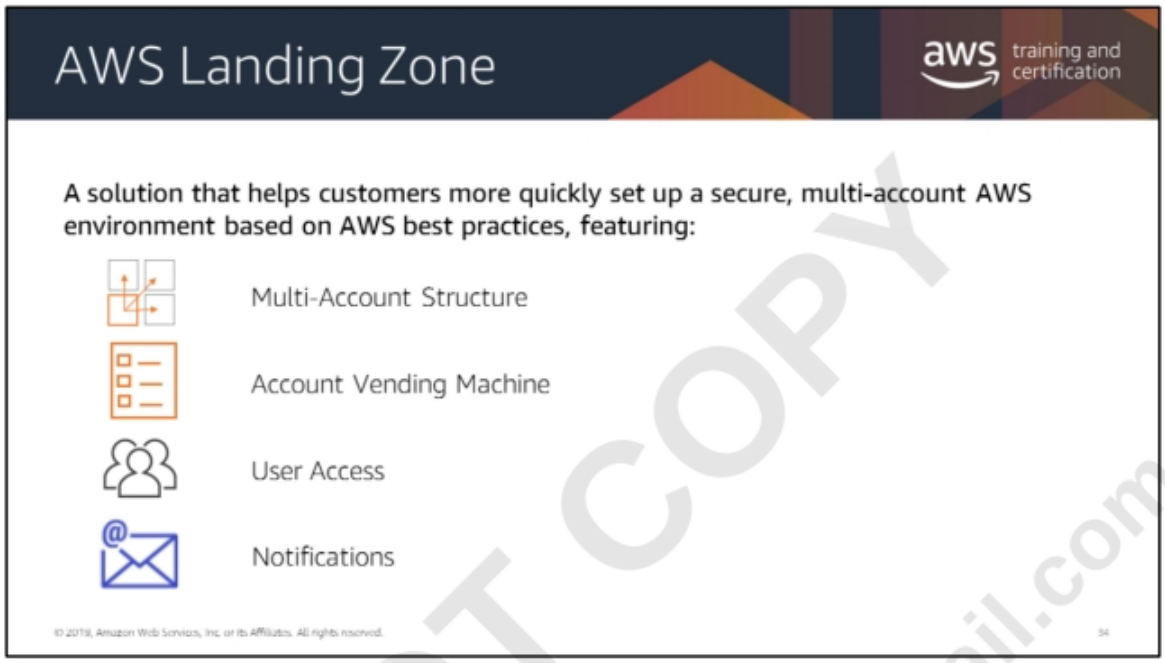
To save user profile information, an Amazon Cognito identity pool must be integrated with an Amazon Cognito user pool.

/



In this scenario, the goal is to authenticate a user and then grant that user access to another AWS service.

- In the first step, the app user signs in through a user pool and, after successfully authenticating, receives user pool tokens.
- Next, the app exchanges the user pool tokens for AWS credentials through an identity pool.
- Finally, the app user uses those AWS credentials to access other AWS services.



AWS Landing Zone is a solution that helps customers more quickly set up a secure, multi-account AWS environment based on AWS best practices. This solution can help save time by automating the set-up of an environment for running secure and scalable workloads while implementing an initial security baseline through the creation of core accounts and resources. It also provides a baseline environment to get started with a multi-account architecture, identity and access management, governance, data security, network design, and logging.

Multi-Account Structure: The AWS Landing Zone solution includes four accounts, and add-on products that can be deployed using the AWS Service Catalog such as the Centralized Logging solution and AWS Managed AD and Directory Connector for AWS SSO.

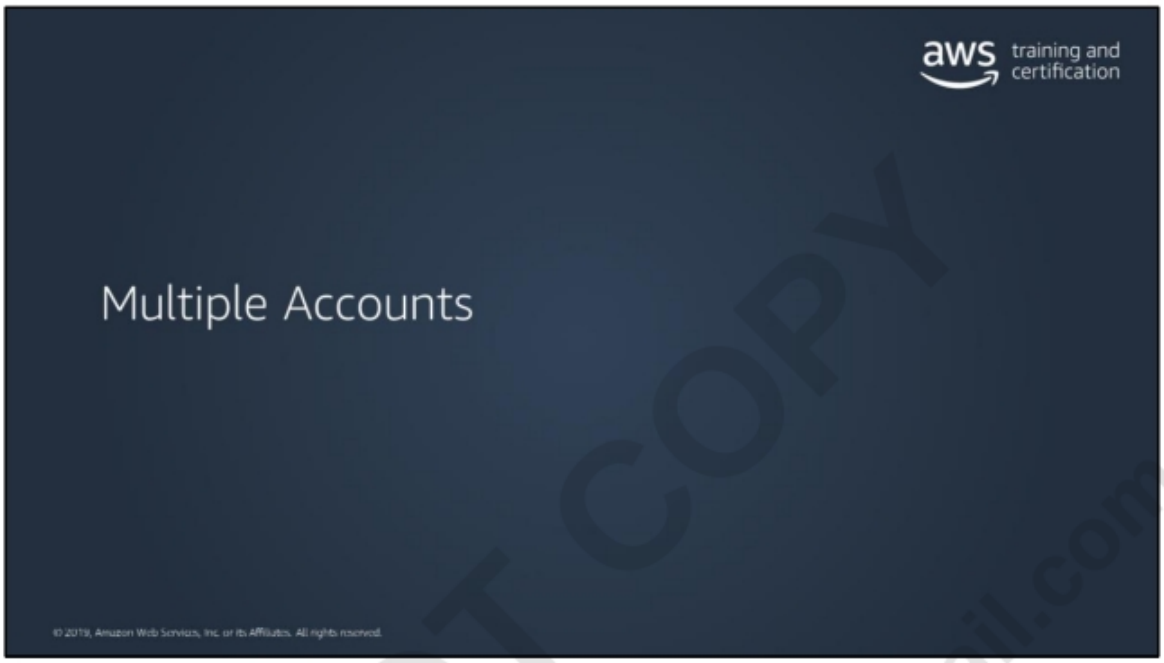
Account Vending Machine: The Account Vending Machine (AVM) is an AWS Landing Zone key component. The AVM is provided as an AWS Service Catalog product, which allows customers to create new AWS accounts in Organizational Units (OUs) preconfigured with an account security baseline, and a predefined network.

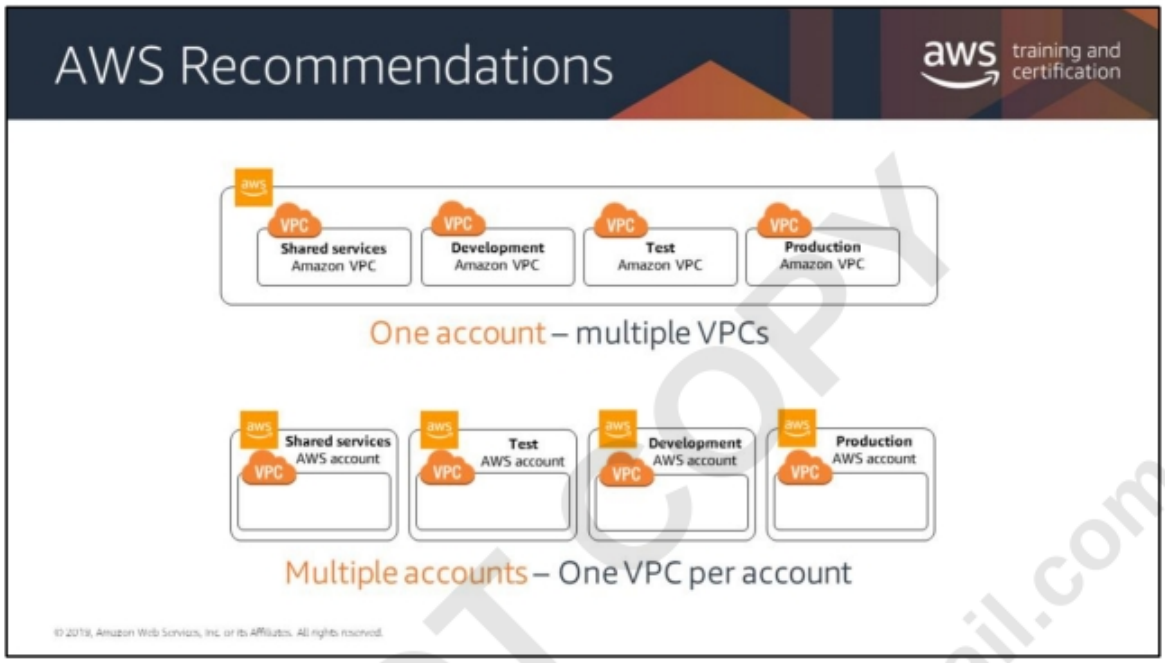
User Access: Providing least-privilege, individual user access to your AWS accounts is an essential, foundational component to AWS account management. The AWS Landing Zone solution provides customers two options to store their users and groups.

Notifications: The AWS Landing Zone solution configures Amazon CloudWatch alarms and events to send a notification on root account login, console sign-in failures, API authentication failures, and the following changes within an account: security groups, network ACLs, Amazon VPC gateways, peering connections, ClassicLink, Amazon Elastic Compute Cloud (Amazon EC2) instance state, large Amazon EC2 instance state, AWS CloudTrail, AWS Identity and Access Management (IAM) policies, and AWS Config rule compliance status.

For more information, see: https://aws.amazon.com/solutions/aws-landing-zone/

/

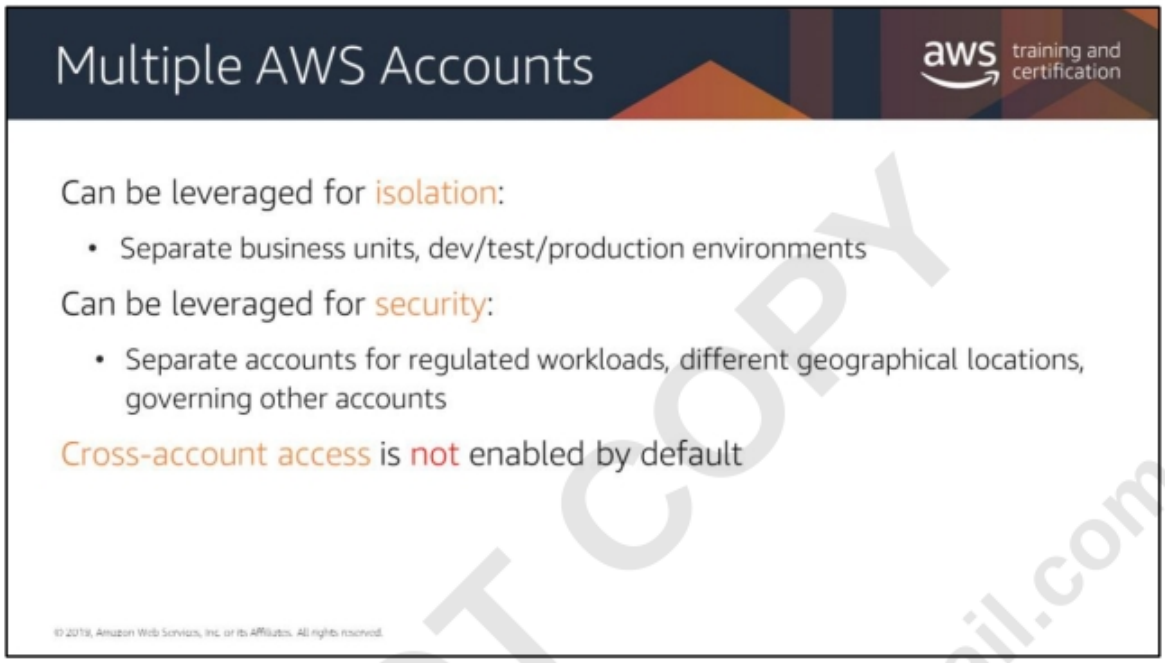




The two primary architectural patterns recommended by AWS are **multi-VPC** (in a single AWS account) and **multi-account**.

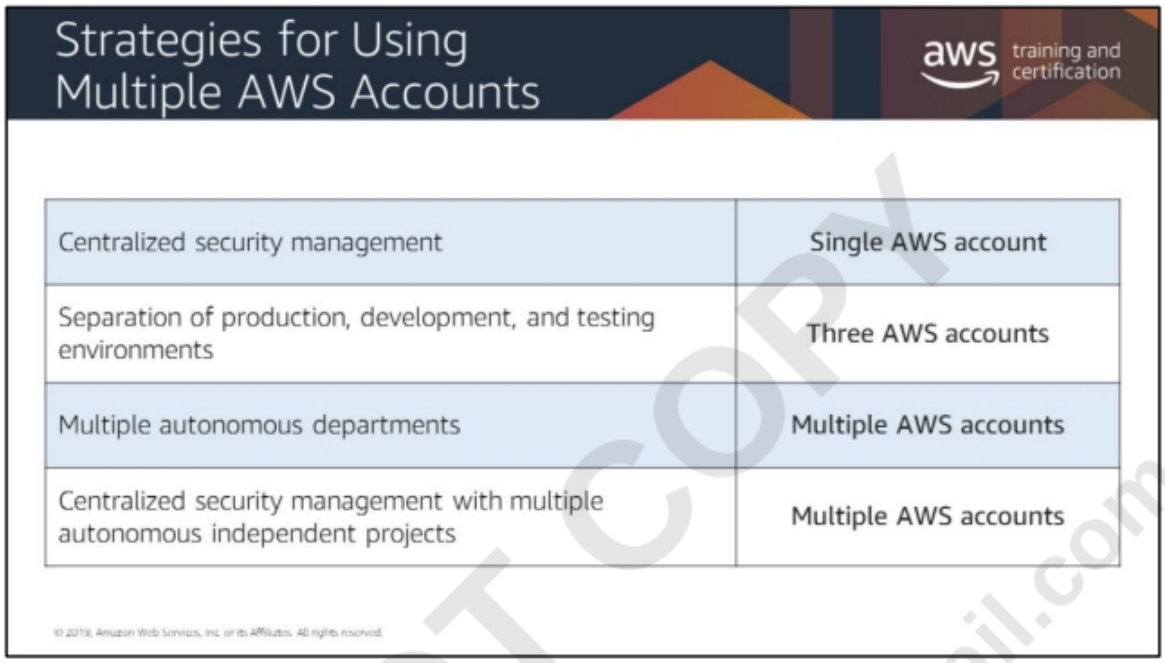
In a multi-account system, each account has a single VPC in it. In practice, organizations (large and small) create multiple accounts. They need to manage, maintain, and audit them.

/



Many AWS customers create multiple AWS accounts for their organizations, such as individual accounts for various business units or separate accounts for their development, test, and production resources.

Using separate AWS accounts (usually with consolidated billing) for development and production resources allows customers to cleanly separate different types of resources and can also provide some security benefits.

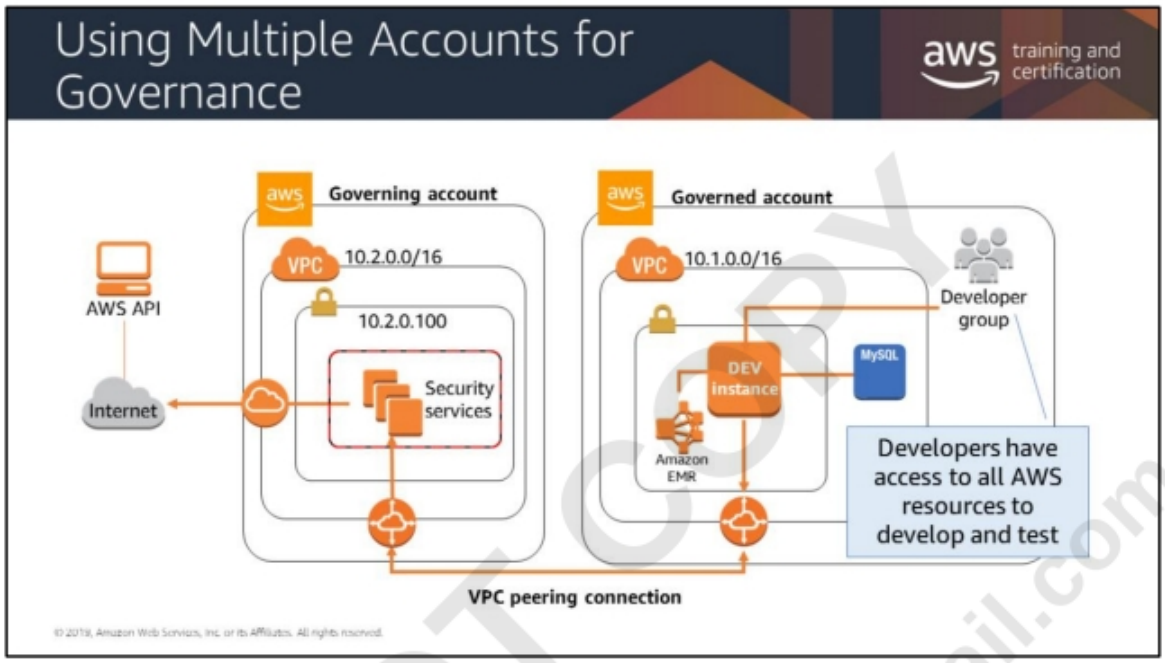


You can design your AWS account strategy to maximize security and follow your business and governance requirements.

If you prefer centralized information security management with minimum overhead, you could opt for a single AWS account. Alternatively, if your business maintains separate environments for production, development, and testing, you could configure three AWS accounts—one for each environment. Also, if you have multiple autonomous departments, you could also create separate AWS accounts for each autonomous part of the organization.

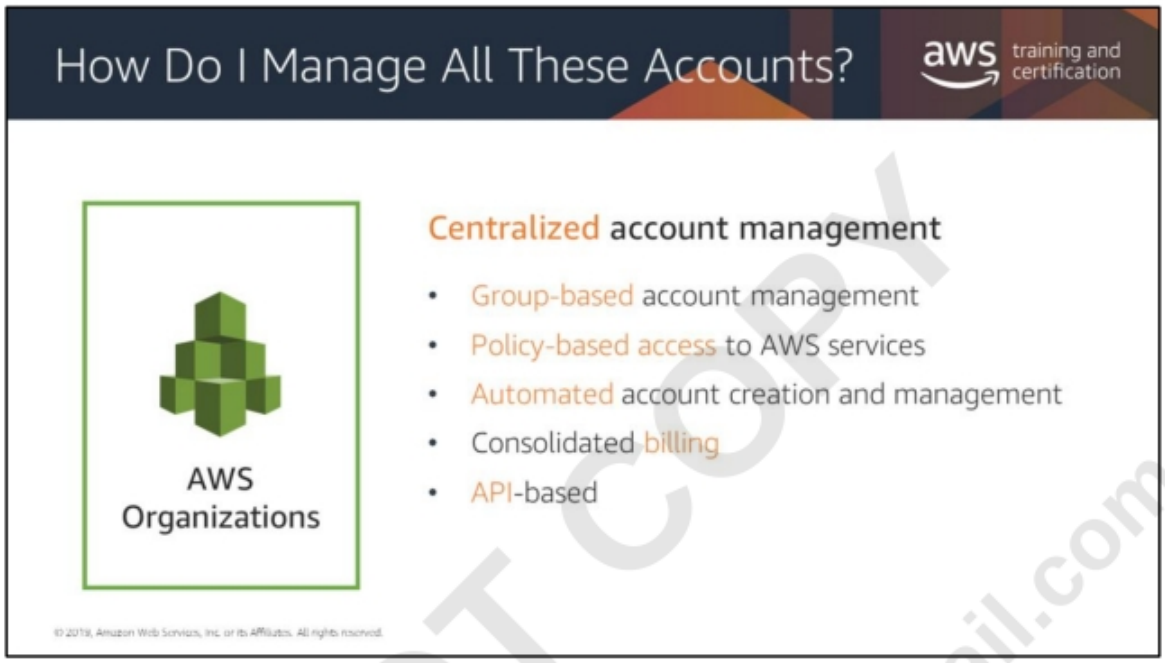
When you use multiple accounts, a more efficient strategy is to create a single AWS account for common project resources (such as DNS services, Active Directory, and CMS) and separate accounts for the autonomous projects/departments. This allows you to assign permissions and policies under each department/project account and grant access to resources across accounts.

/



Many large companies use multiple accounts for security and governance. In this approach, two or more AWS accounts are created, with one designated as a Governing account and the others designated as Governed accounts. The solution is architected to isolate all management resources to the Governing account's network. All ingress and egress traffic to the Governed account passes through the security-specific services in the Governing account. This allows for an additional layer of security configured in the Governing account for enhanced security and governance purposes.

The Governed account should still be architected following the security best practices. The Governing account is used to provide an extra layer of security that can be managed centrally.



AWS Organizations is a managed service for account management. An **organization** is an entity that you create to consolidate, centrally view, and manage all of your AWS accounts. In Organizations, an organization has the functionality that is determined by the feature set that you enable.

Centrally manage policies across multiple AWS accounts

Organizations help you manage policies for multiple AWS accounts. Use the service to create groups of accounts and then attach policies to a group to ensure the correct policies are applied across the accounts.

Organizations enables you to centrally manage policies across multiple accounts, without requiring custom scripts and manual processes.

Group-based Account Management

Using Organizations, you can create groups of AWS accounts. You can create separate groups of accounts to use with development and production resources, and then apply different policies to each group.

/

Policy-based Access to AWS Services

With Organizations, you can create Service Control Policies (SCPs) that centrally control AWS service use across multiple AWS accounts. SCPs put bounds around the permissions that IAM policies can grant to entities in an account, such as IAM users and roles. Entities can only use the services allowed by both the SCP and the IAM policy for the account. For example, If you want to restrict access to AWS Direct Connect, the SCP must allow access before IAM policies will work. You can apply policies to a group of accounts or all the accounts in your organization.

Automate AWS Account Creation and Management

Use the Organizations APIs to automate the creation and management of new AWS accounts. The Organizations APIs can create new accounts programmatically, and to add them to a group. The policies attached to the group are automatically applied to the new account. For example, you can automate the creation of sandbox accounts for developers and grant entities in those accounts access only to the necessary AWS services.

Consolidate billing across multiple AWS accounts

AS Organizations enables you to set up a single payment method for all the AWS accounts in your organization through consolidated billing. With consolidated billing, you can see a combined view of charges incurred by all your accounts. Consolidated billing also allows you to take advantage of pricing benefits from aggregated usage such as volume discounts for Amazon EC2 and Amazon S3.

API level control of AWS Services

With Organizations, you can use SCPs to manage the use of AWS services at an API level. For example, you can apply a policy to a group of accounts to only allow IAM users in those accounts to read data from Amazon S3 buckets.

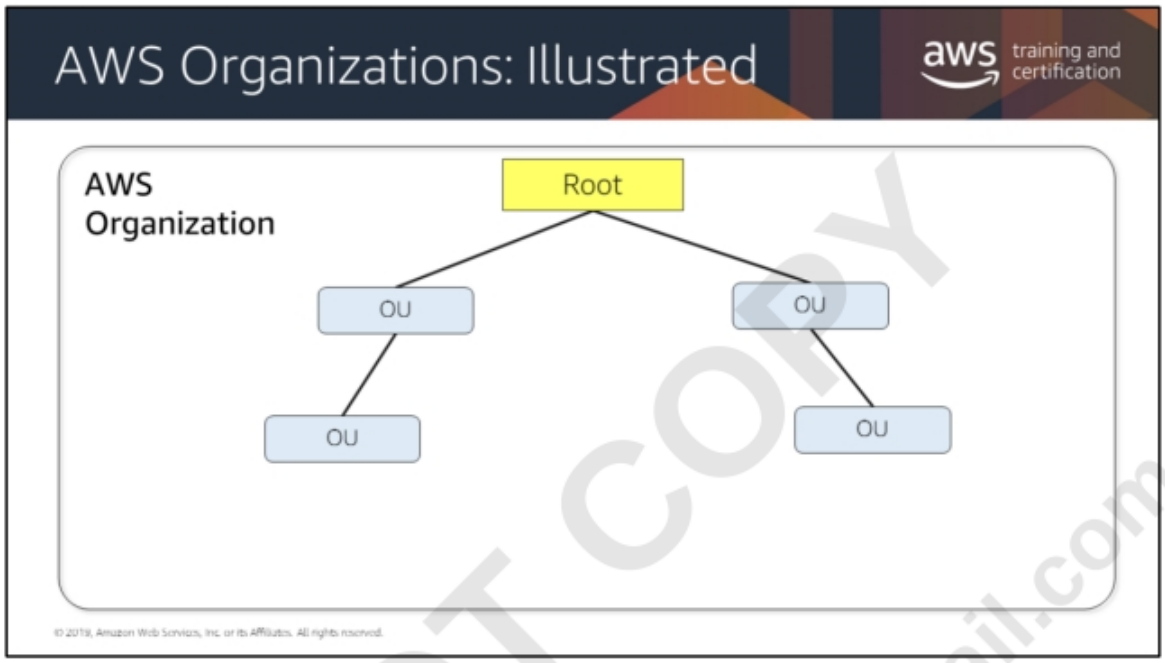
Using the Organizations APIs, you can create and add new accounts to a group. Policies attached to a group are automatically applied to accounts added to the group.

/

/

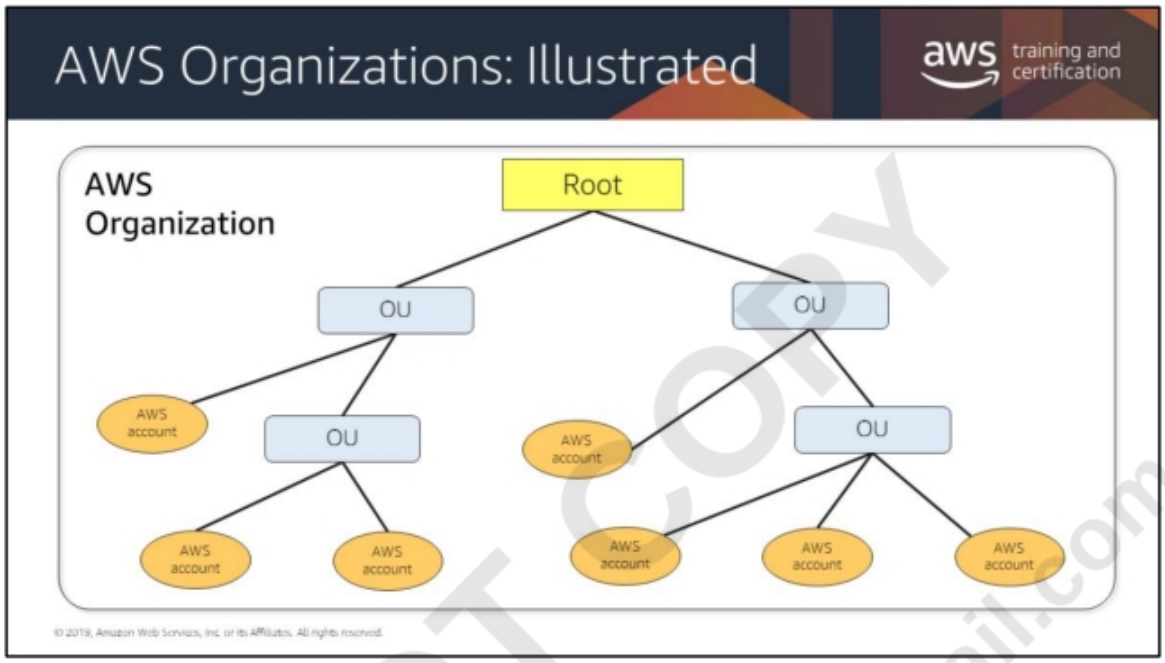
In this example, I have an organization has seven accounts that need to be organized into four Organizational Units (OUs) under the root.

/



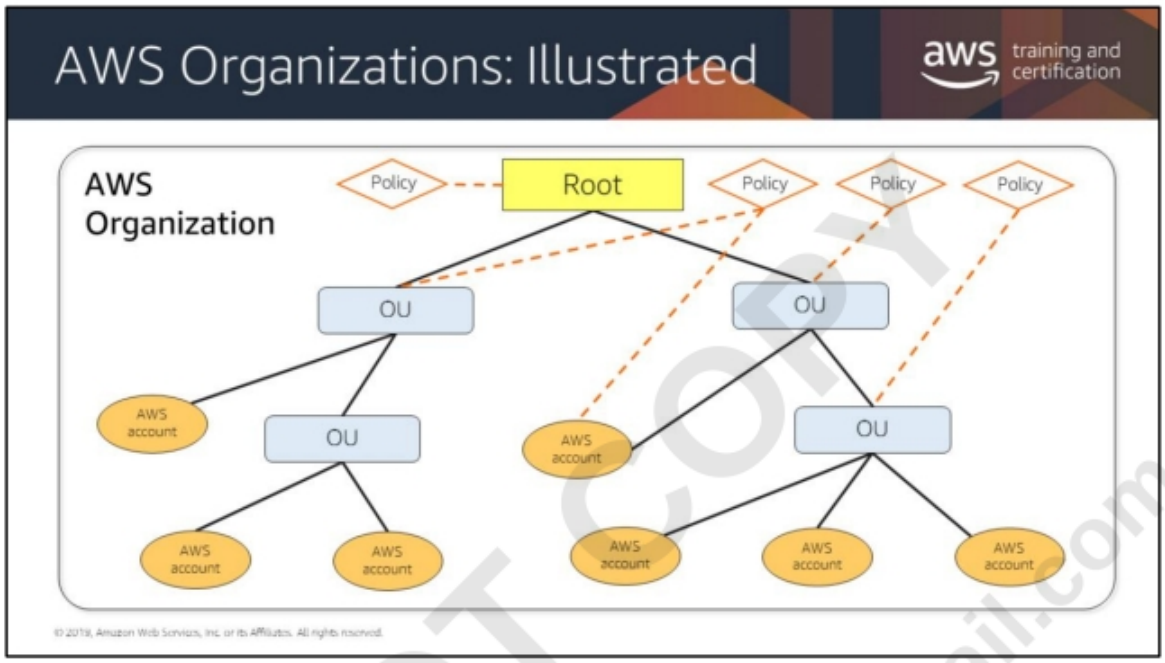
Here, I've added four Organizational Units (OUs) to my Organization. Two sit directly under the root. Then, I have a single OU in each of my primary OUs.

/



All seven of my AWS accounts are added to my organization and put into the appropriate OU.

/

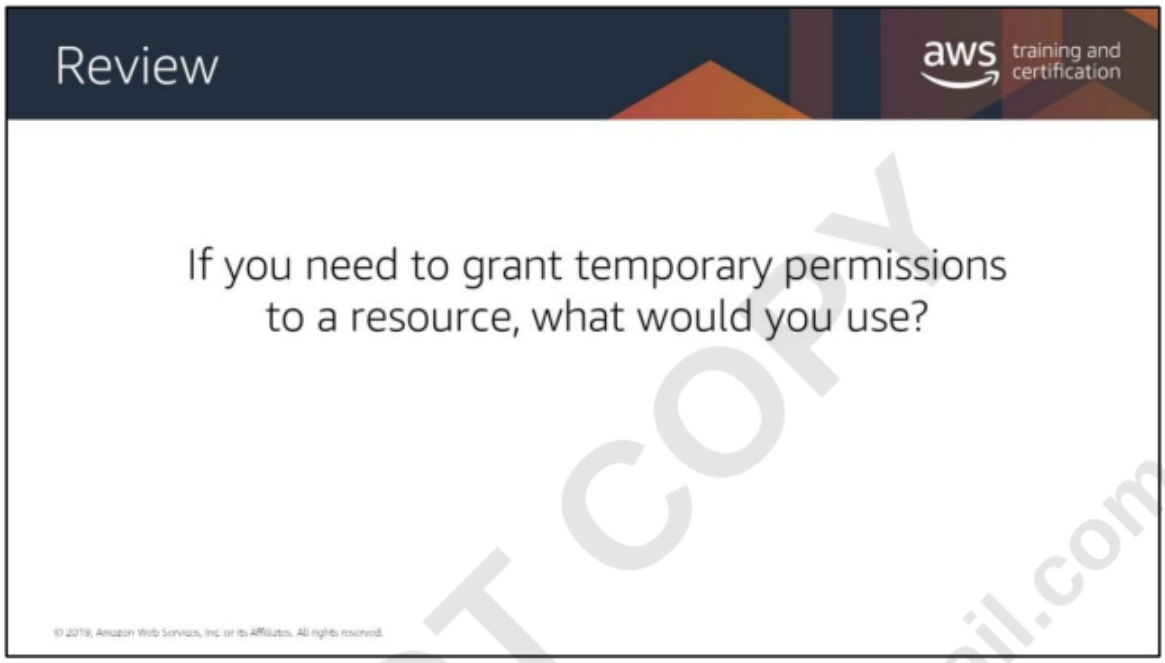


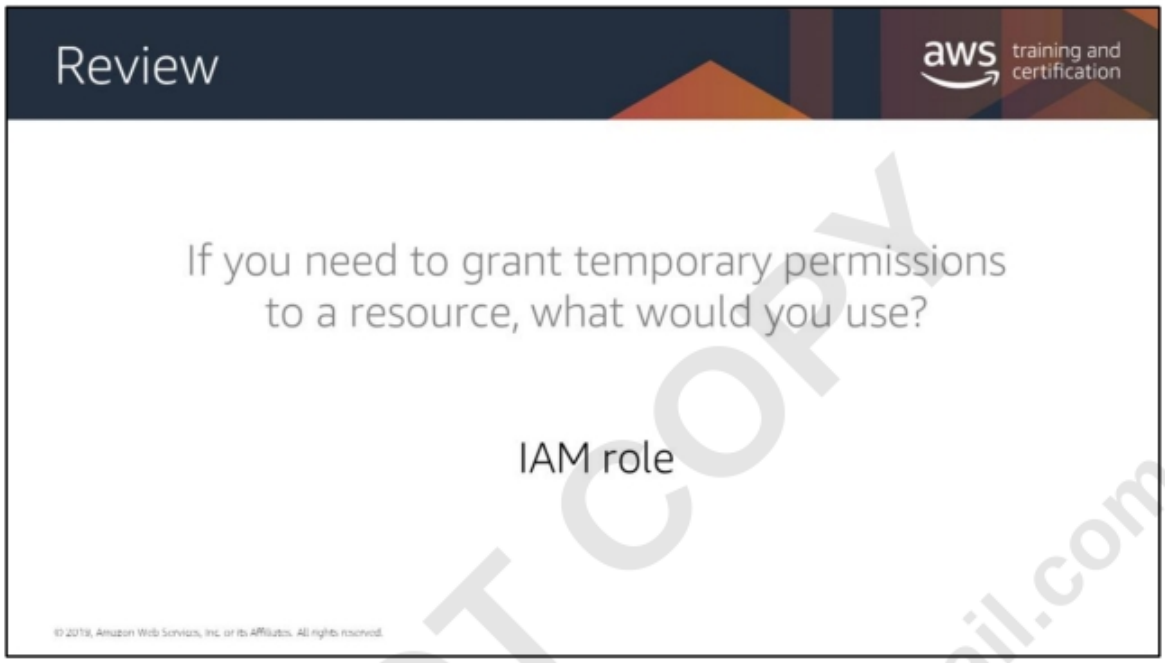
Once the accounts are added, I can apply SCPs to my organization.

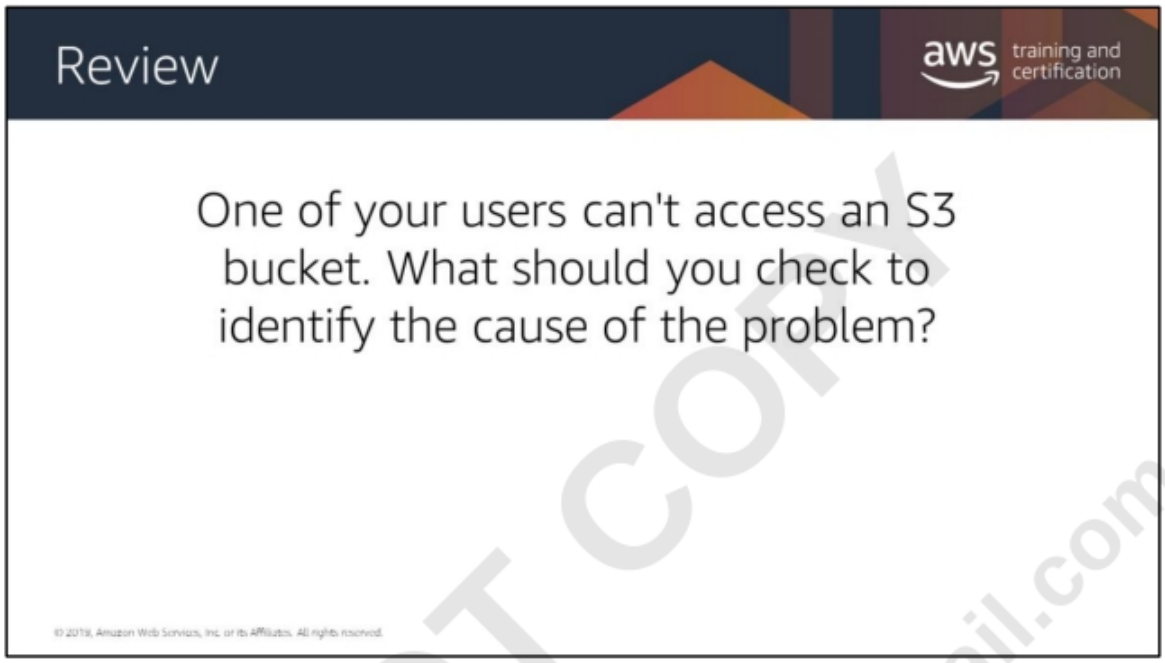
In this example, the root has an SCP attached to it. This policy will apply to all of the OUs and accounts in the organization. An SCP can be applied to one or more OUs or individual accounts.

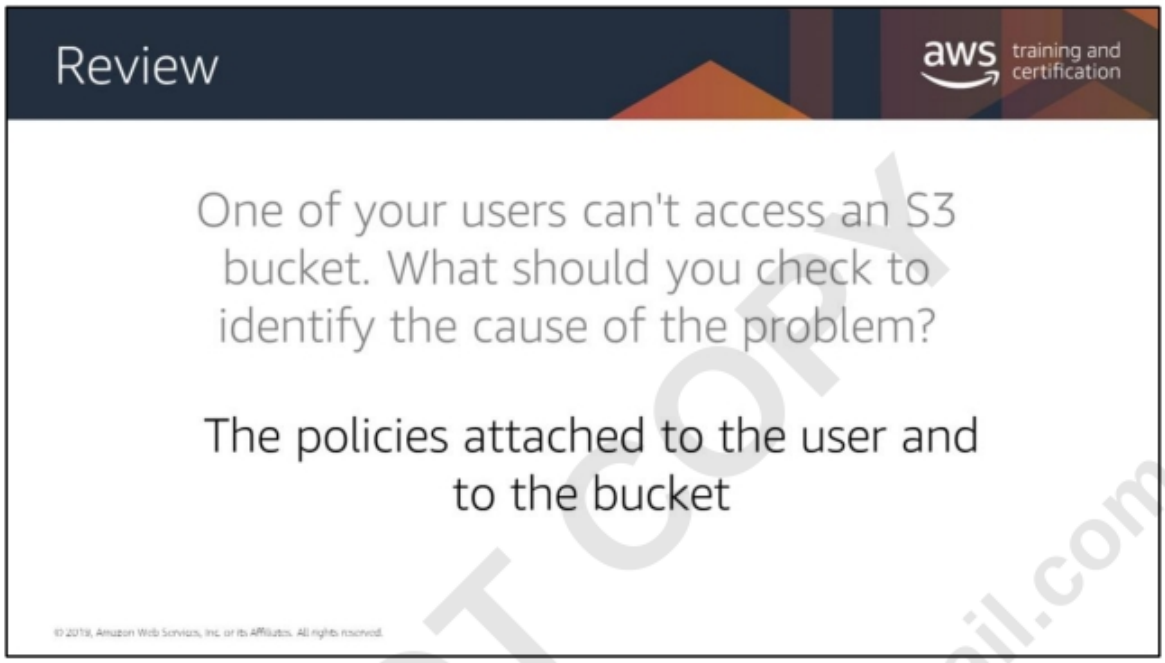
Service control policies in AWS Organizations enable fine-grained permission controls. For more information see: https://aws.amazon.com/about-aws/whats-new/2019/03/service-control-policies-enable-fine-grained-permission-controls/.

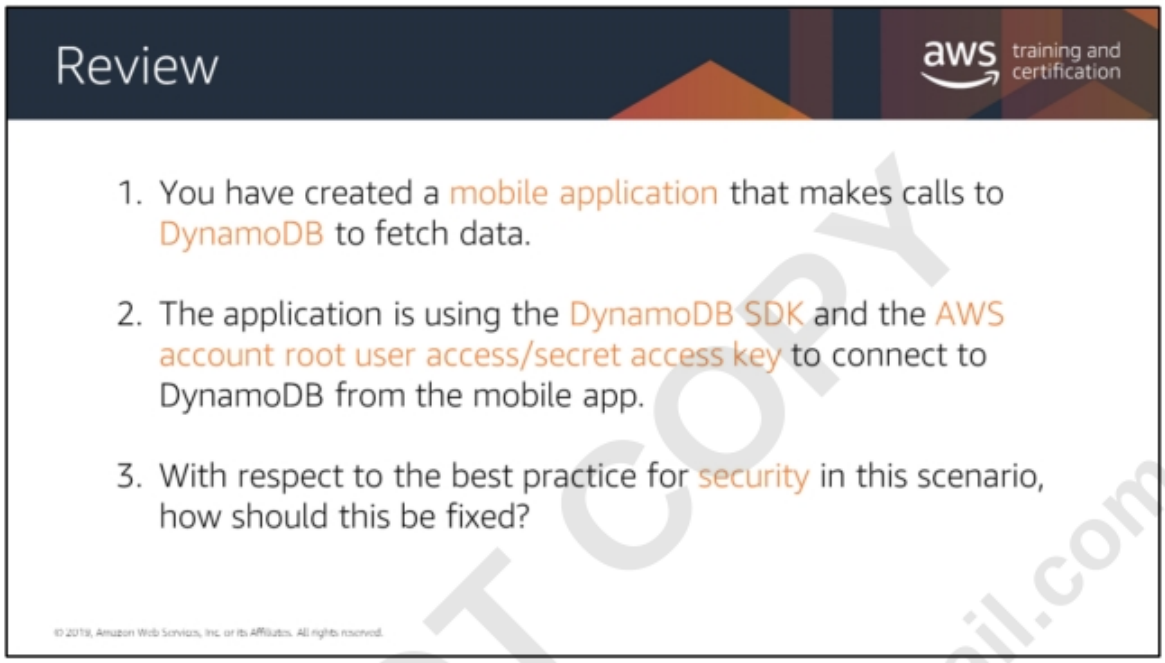
/

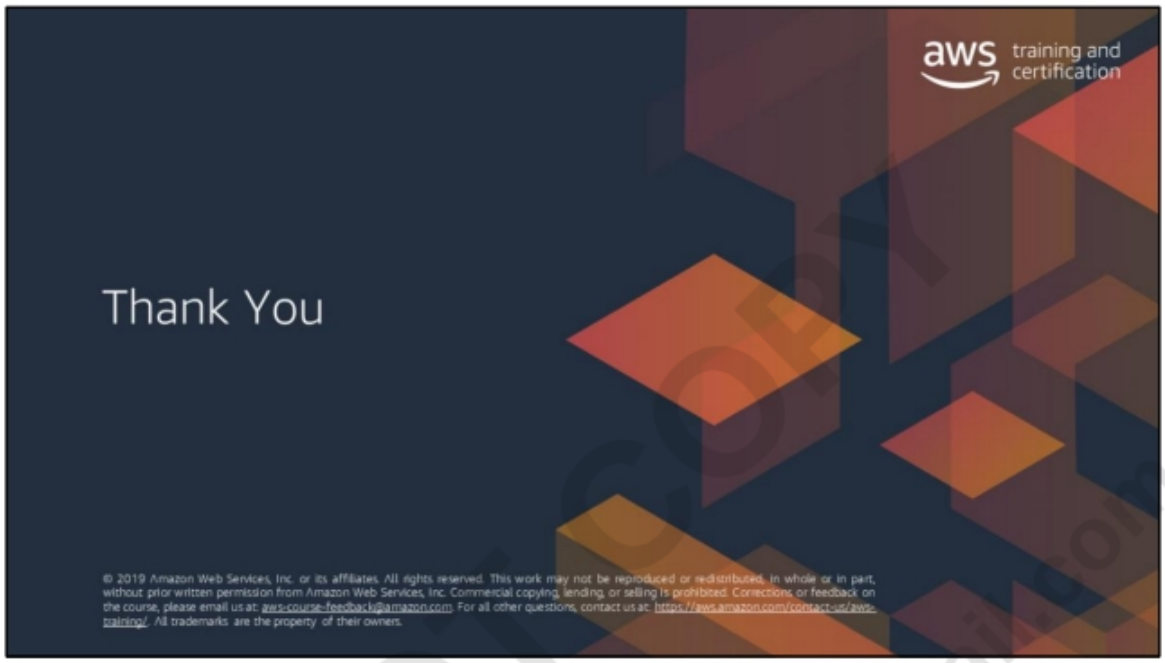


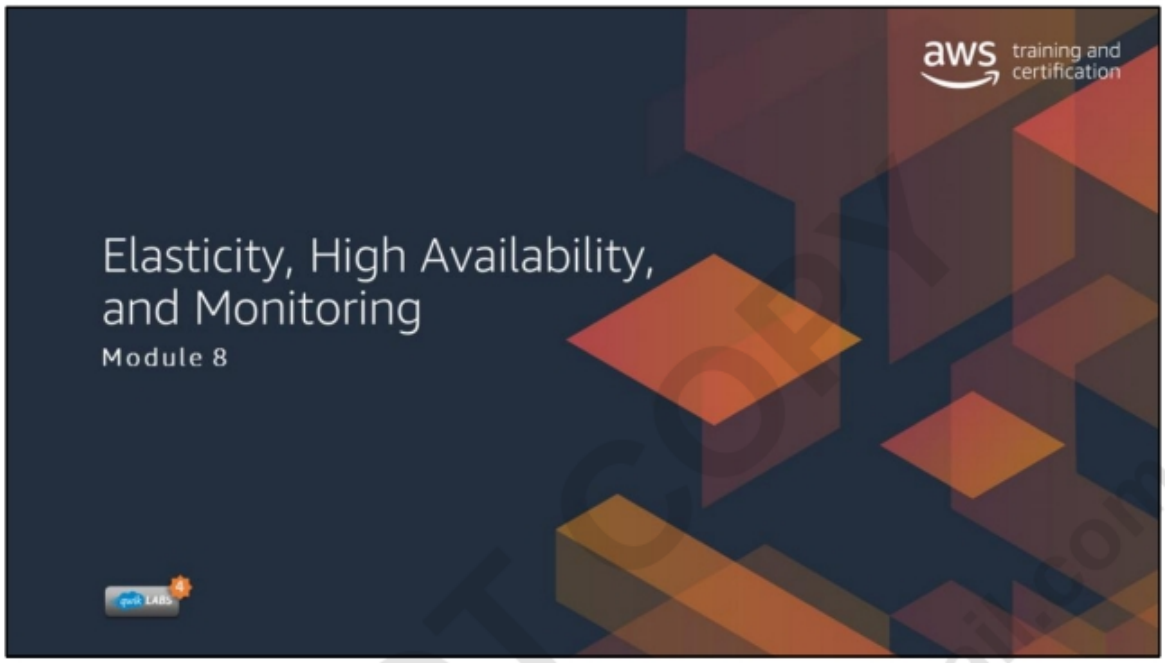


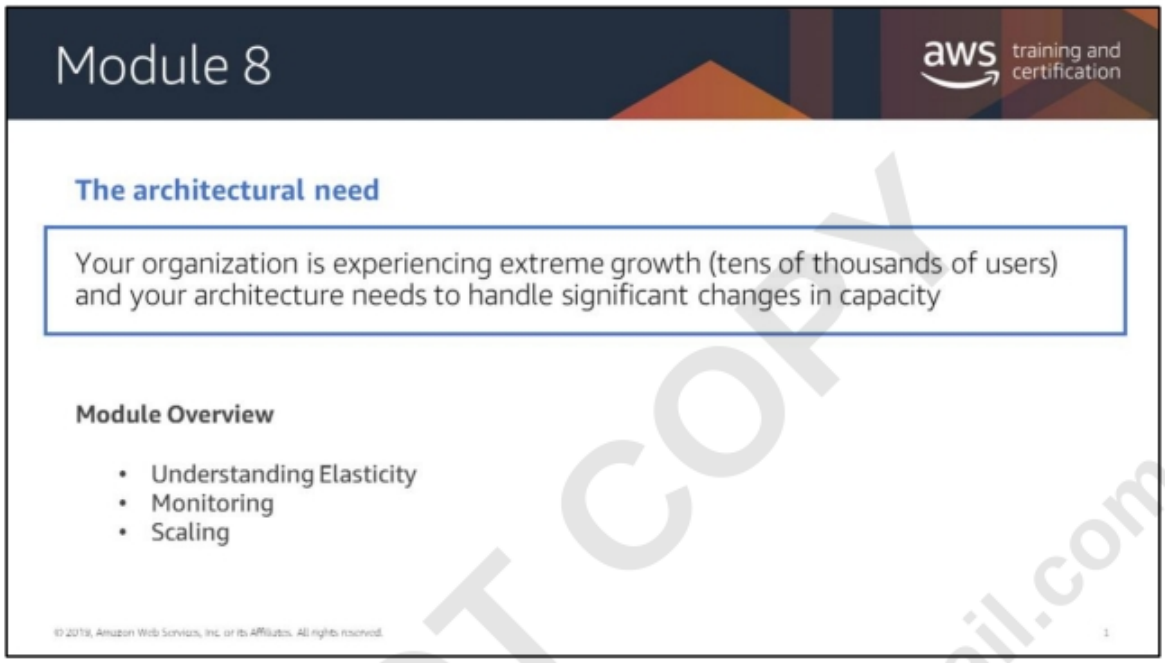


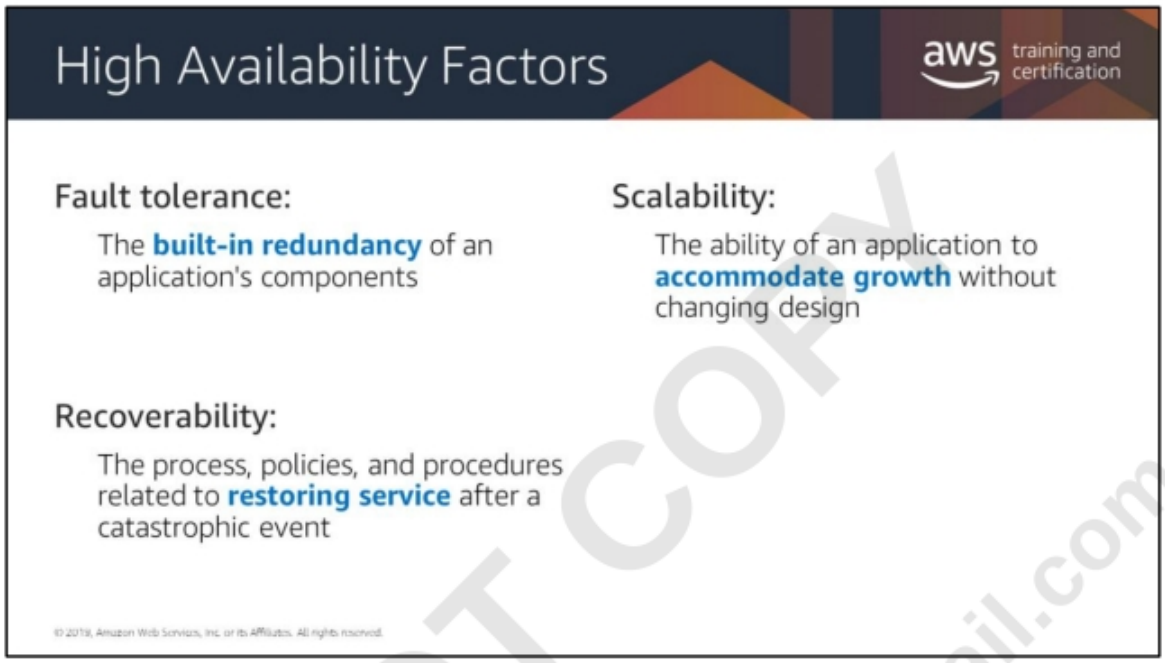












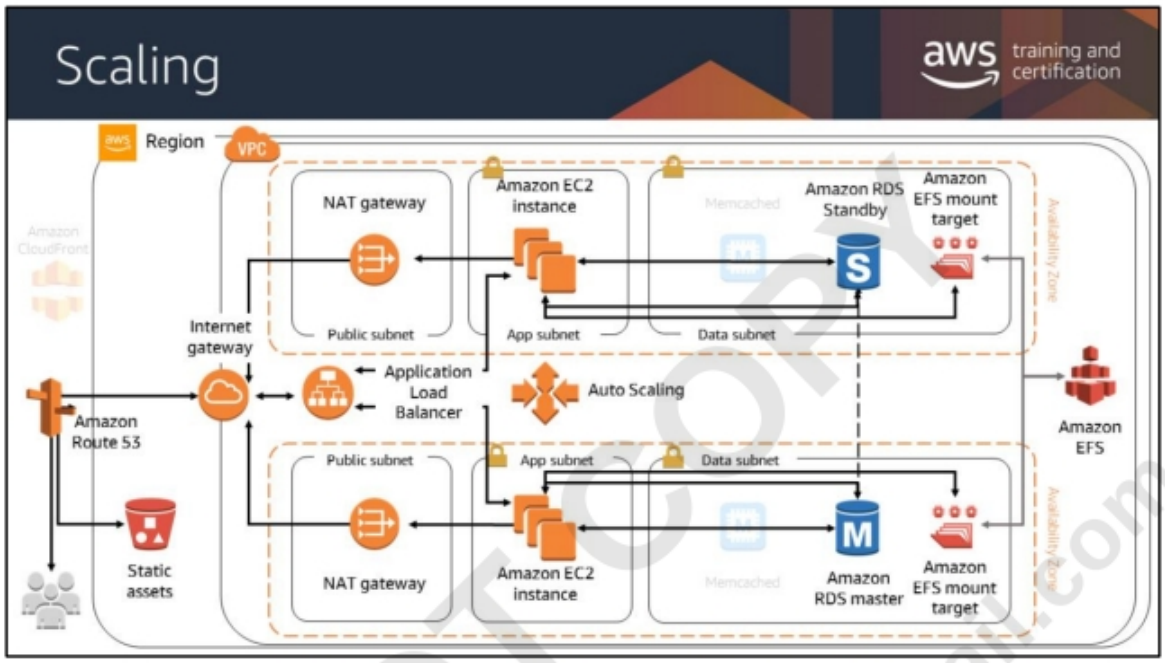
Three factors that determine the overall availability of your application are fault tolerance, recoverability, and scalability.

Fault tolerance is often confused with high availability, but fault tolerance refers to the built-in redundancy of an application's components. Does it avoid single points of failure? This module covers fault tolerance later.

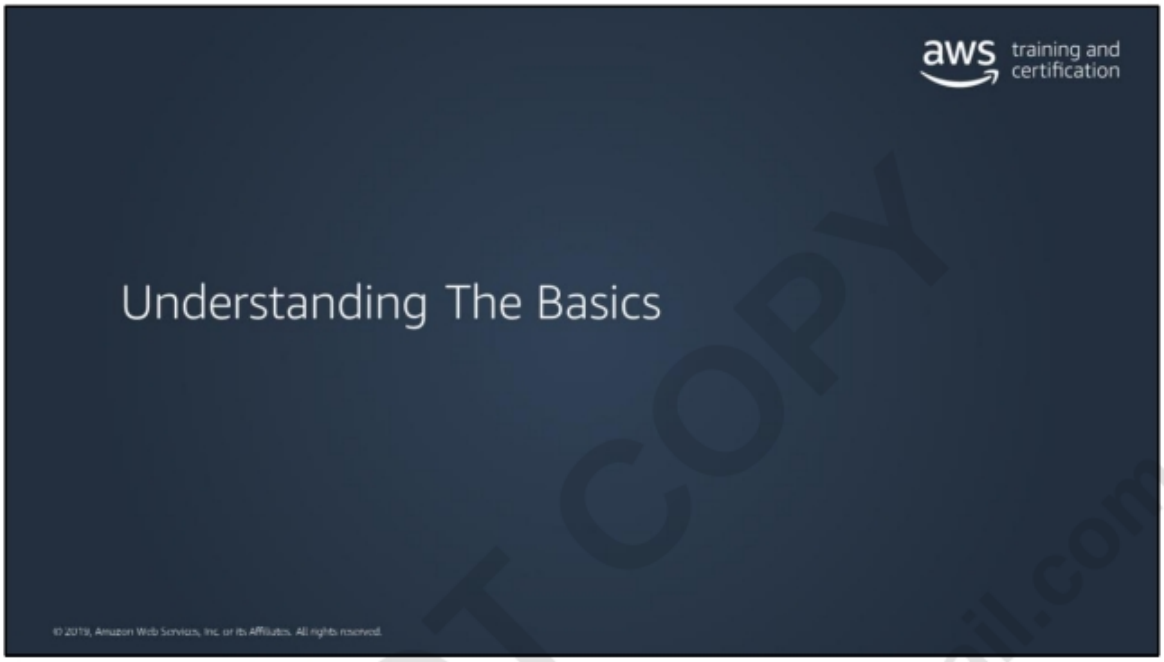
Recoverability is often overlooked as a component of availability. If a natural disaster makes one or more of your components unavailable or destroys your primary data source, can you restore service quickly and without lost data? Specific disaster recovery strategies will be covered in a later module.

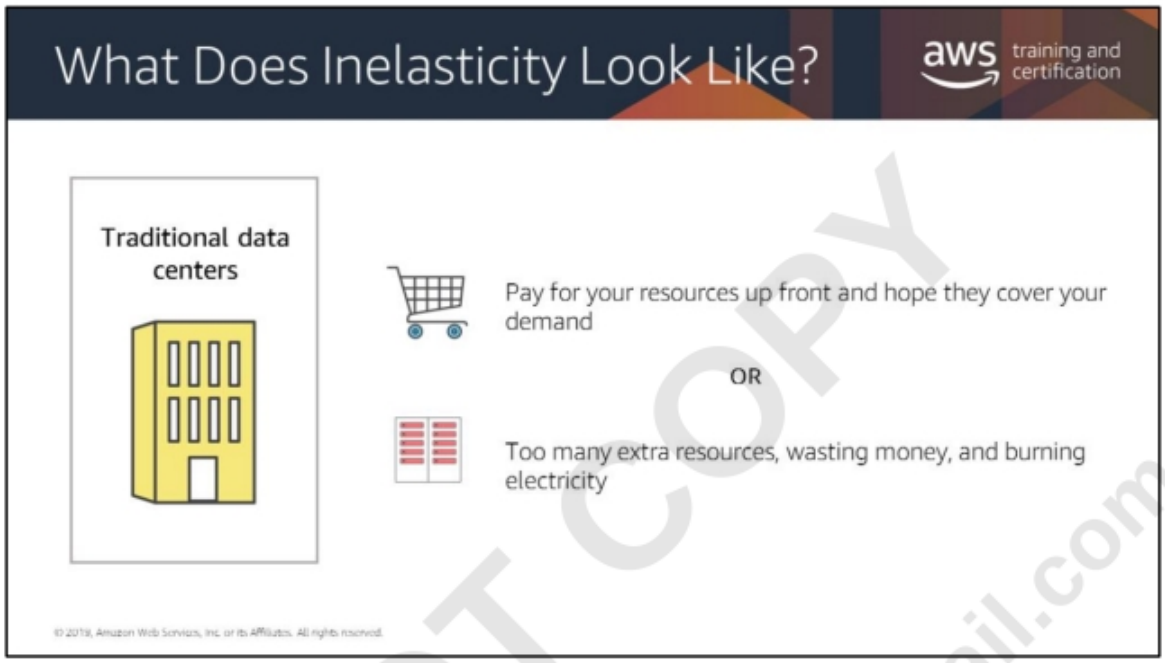
Scalability is the measure of how quickly your application's infrastructure can respond to increased capacity needs so that your application is available and performing within your required standards. It does not guarantee availability, but is one part of your application's availability.

/



By the end of class, you will be able to understand all of the components of this architectural diagram. You will also be able to construct your own architectural solutions that are just as large and robust.

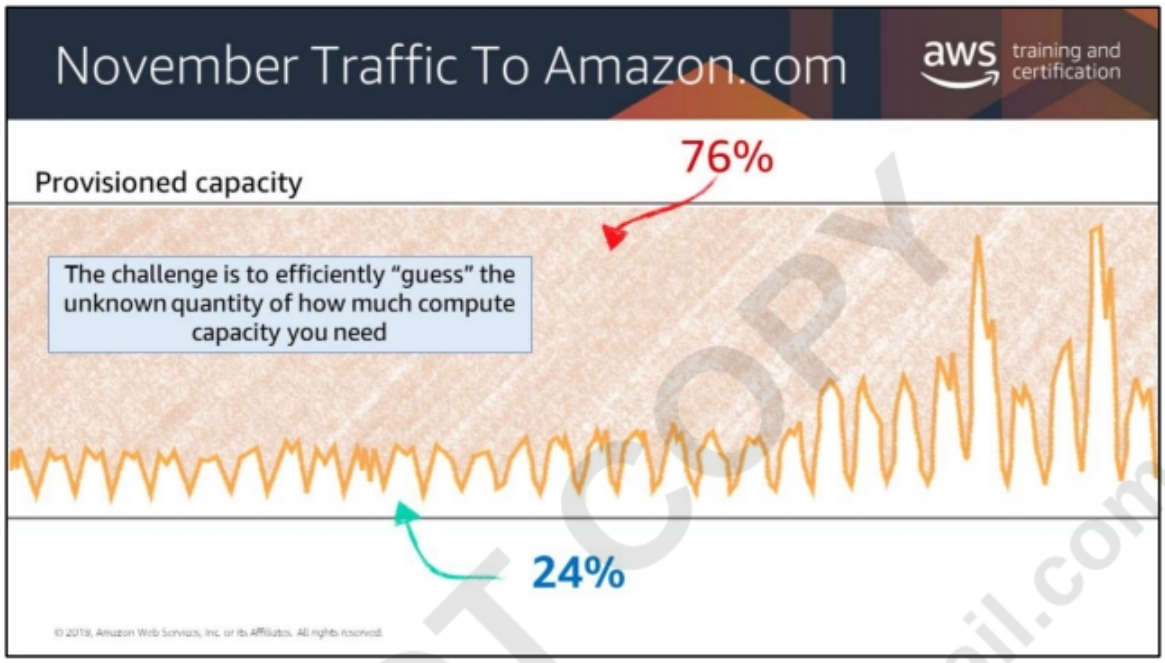




Traditional data centers: Once deployed, resources typically runs whether they are needed or not. You will end up paying for capacity that might not ever be used. Worse yet, you could need more capacity immediately, but be unable to obtain it.

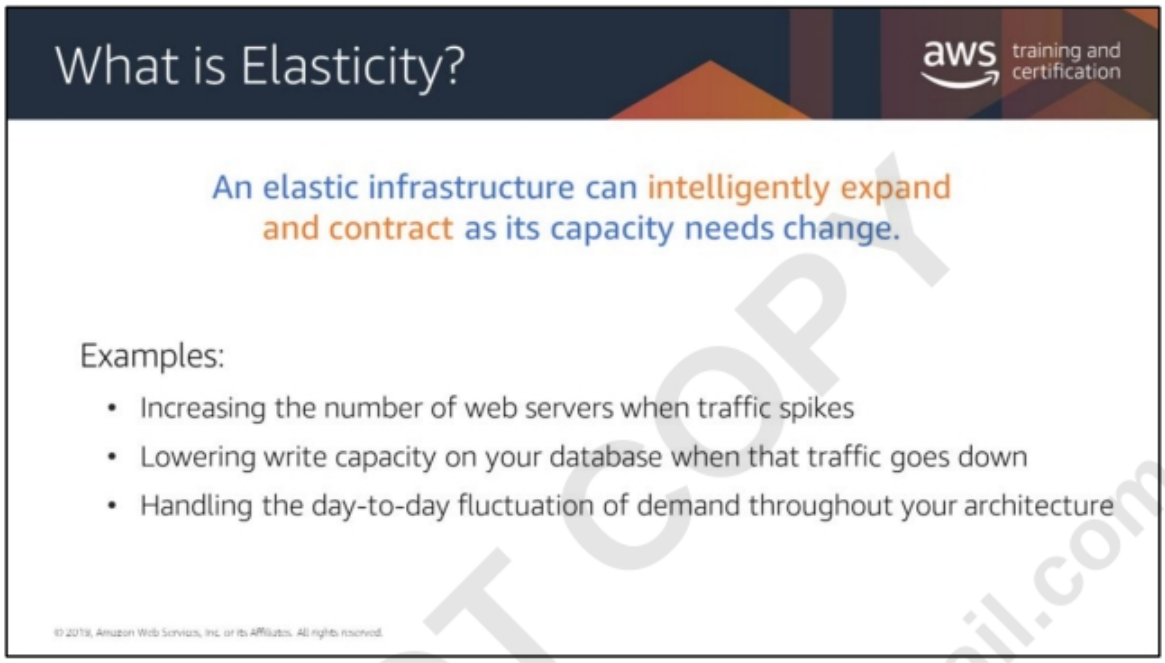
can grow or shrink to match the required demand.

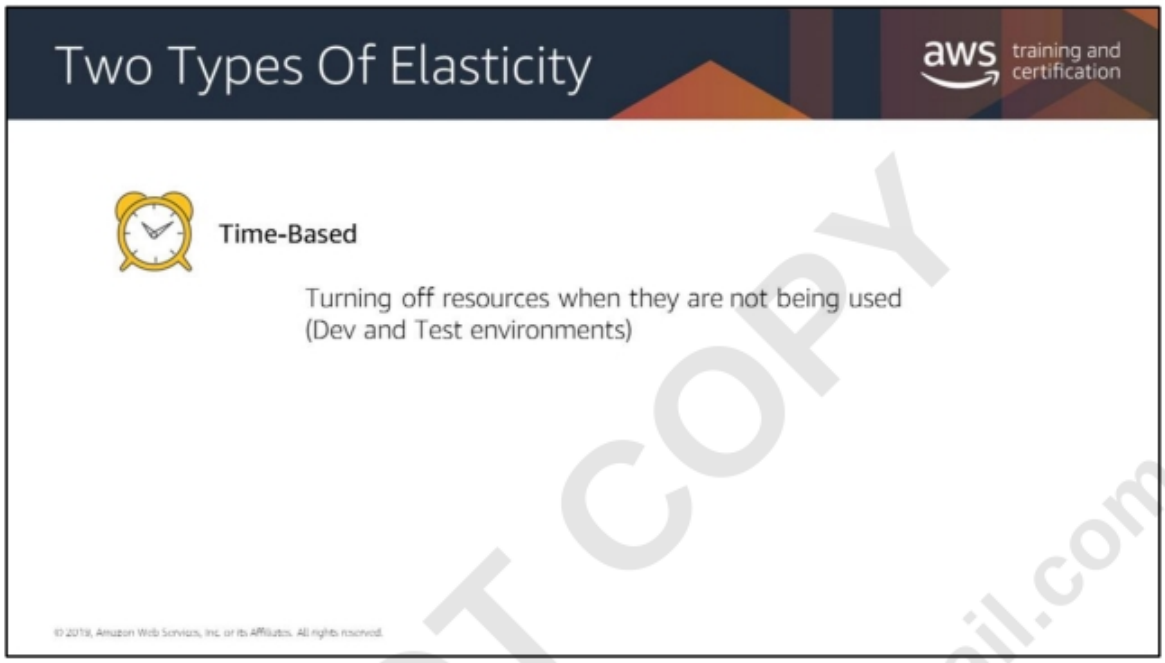
Needless to say, the retail company Amazon.com is one of the largest AWS customers. Typically, the incoming traffic is very predictable. Before Amazon.com moved their infrastructure onto AWS, they had a traditional data center, as many companies had. In order to support the peak load, your data center must provide enough hardware and software to support the capacity.

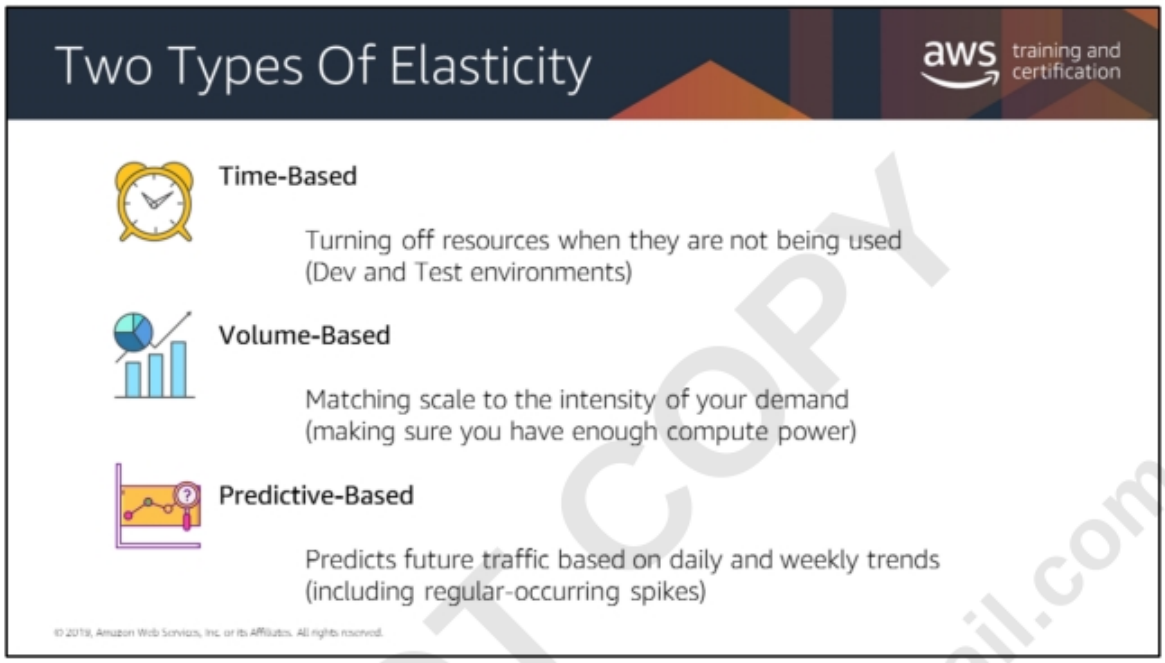


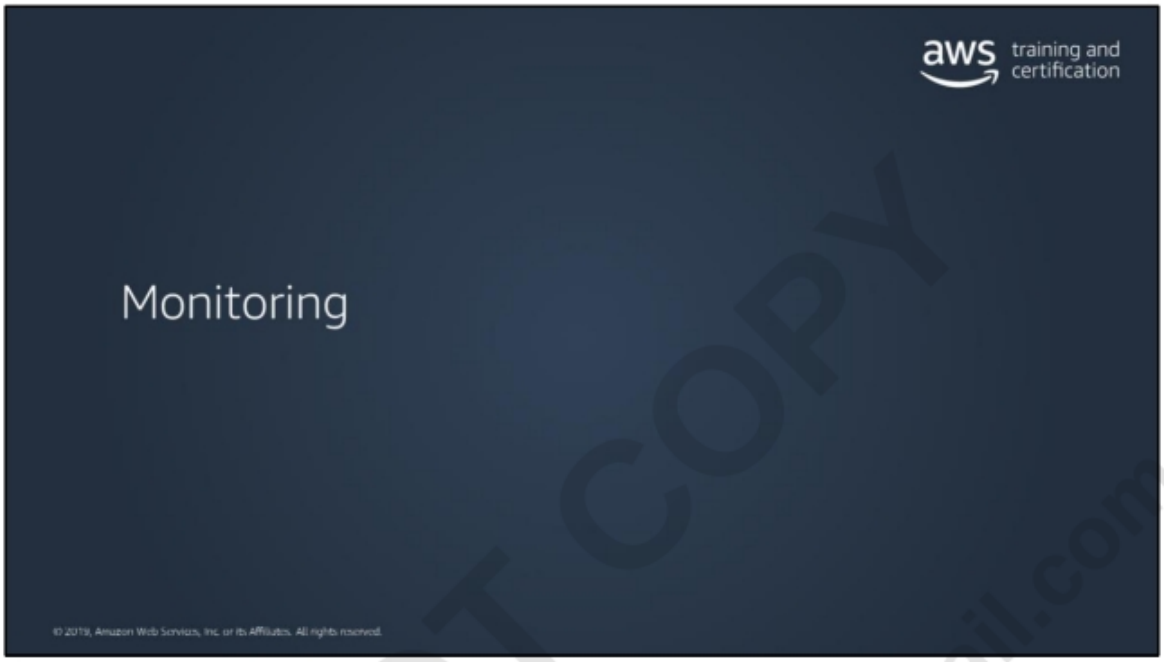
Amazon.com experiences a seasonal peak in November (Black Friday, a key consumer shopping day in the United States). The company had to invest in enough resources to support this once-yearly seasonal peak. As the business grew, Amazon.com had to keep investing in additional hardware and software. At some point, they ran out of space, so they had to add a new data center.

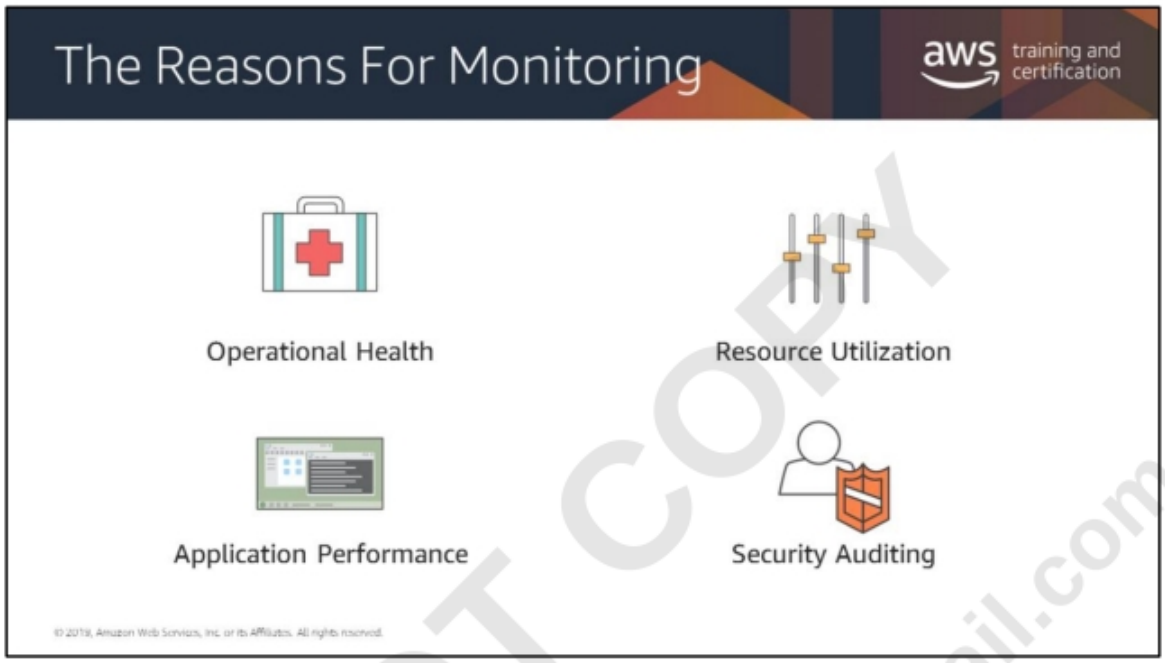
By using an on-premises solution, about 76 percent of the resources were left idle for most of the year, which wastes resources. But without investing in that additional hardware, the company might not have had enough compute capability to support the seasonal peak. If the servers had crashed, the business might have lost customer confidence.







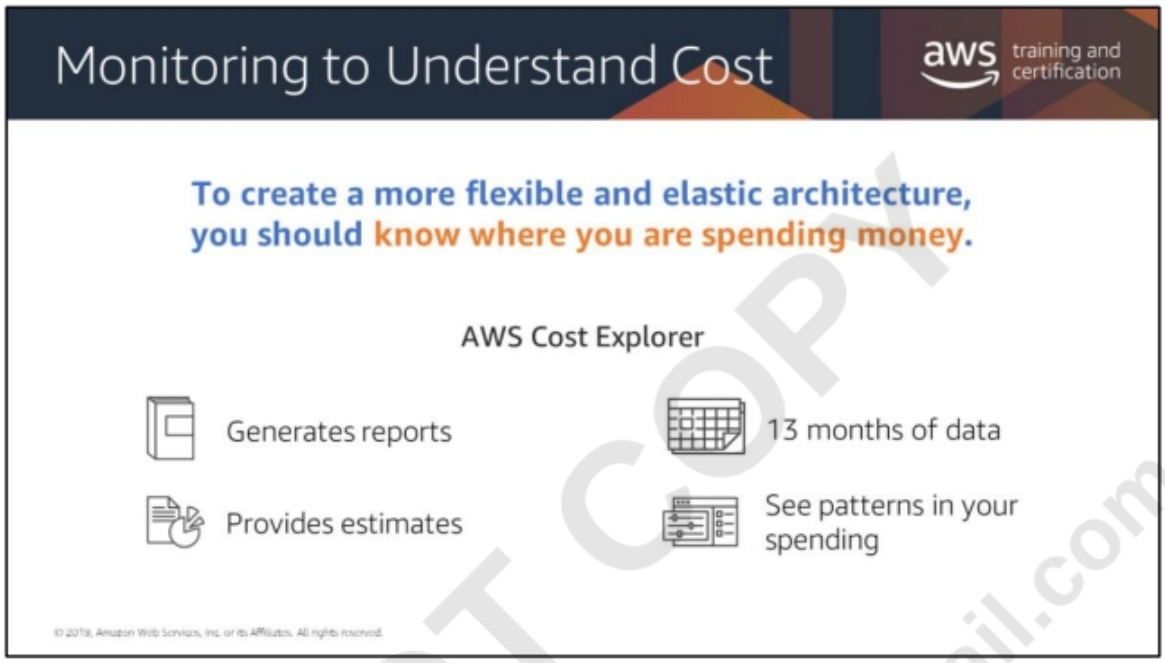




Monitoring your environment is one of the most important things to think about when creating architecture. You will always need a way to keep track of how your resources are operating and performing. Monitoring give you the first hints for asking the question does something need to change. Here are a few points to remember:

- Monitoring is really the first step to building a reactive architecture that can scale as demand climbs and pull back when focus shifts away. This kind of scaling will greatly save you money and provide a better user experience for you and your customers.
- Resource utilization and application performance will be a large component for making sure our infrastructure is satisfying our demand. You can pull this information through monitoring.
- Monitoring also is very important on a security standpoint. With effective parameters in place you can understand when your users are accessing pieces of your AWS environment that they shouldn't.

/



Cost Optimization Monitor – Can generate reports that provide insight into service usage and cost. Provides estimates costs that can break down by period, account, resource or tags.

AWS Cost Explorer – Can view data up to the last 13 months, allowing you to see patterns in how you spend on AWS resources over time.

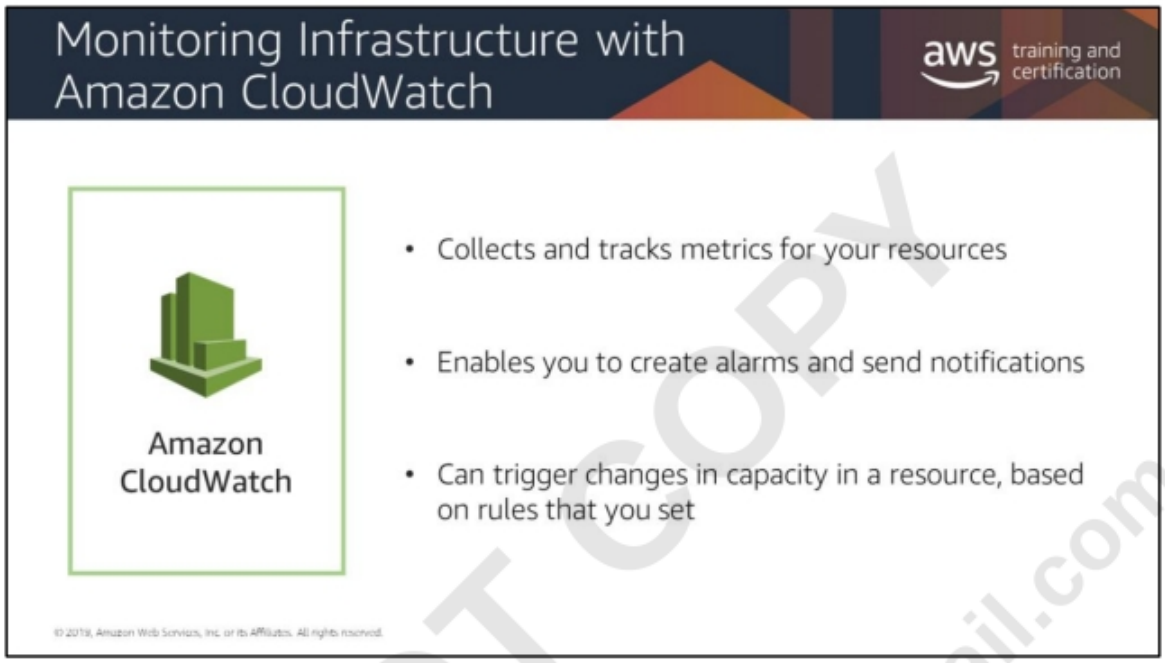
Forecasting with AWS Cost Explorer - A forecast is a prediction of how much you will use AWS services over the forecast time period you selected, based on your past usage. You create a forecast by selecting a future time range for your report. Forecasting provides an estimate of what your AWS bill will be and enables you to use alarms and budgets for amounts that you're predicted to use. Because forecasts are predictions, the forecasted billing amounts are estimated and might differ from your actual charges for each statement period.

/

Different ranges of accuracy have different confidence intervals. The higher the confidence interval, the more likely the forecast is to be correct. AWS Cost Explorer forecasts have a confidence interval of 80%. If AWS doesn't have enough data to forecast within an 80% confidence interval, AWS Cost Explorer doesn't show a forecast.

https://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/ce-modify.html#ce-timerange

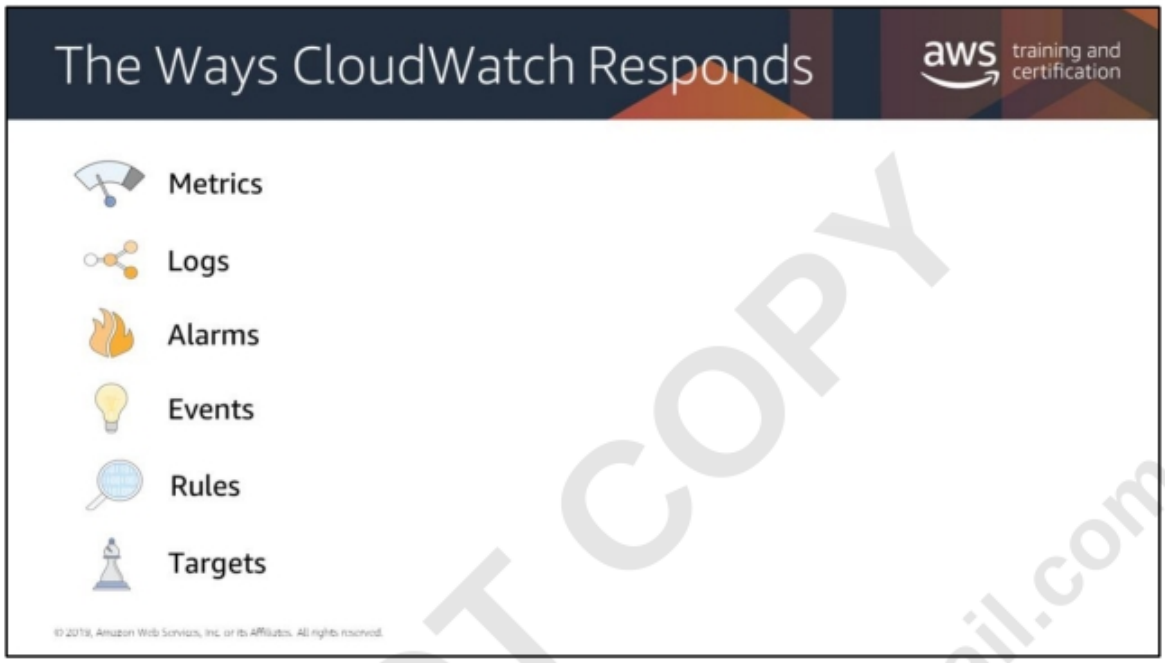
/

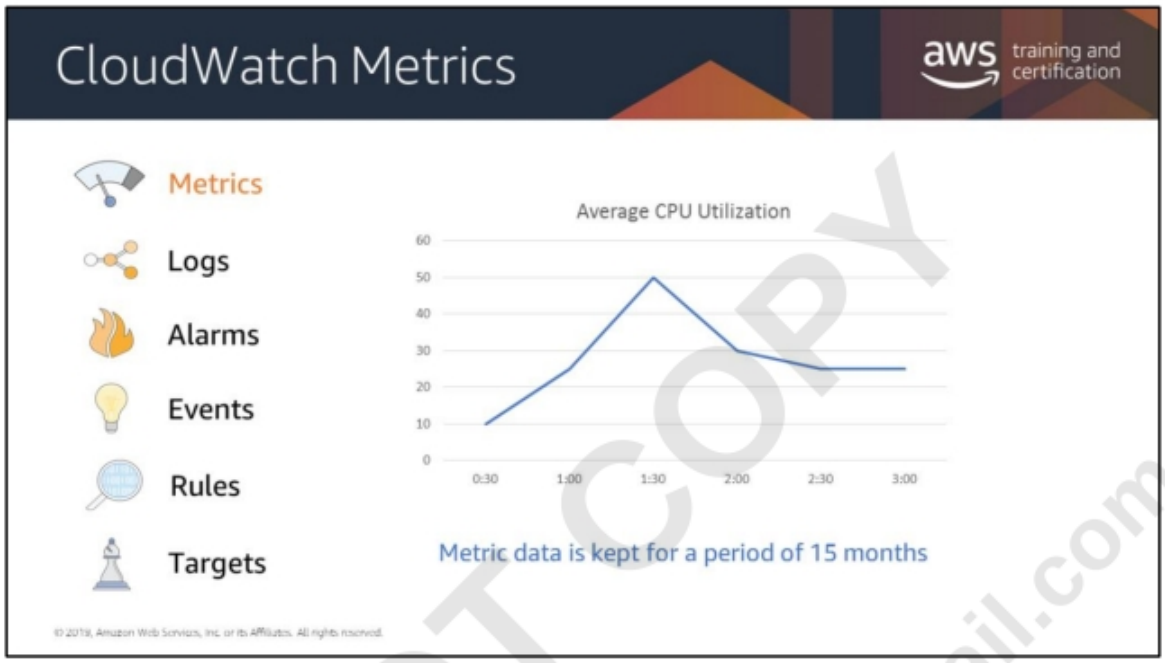


The first step on our journey to create elastic architectures is an overview of Amazon CloudWatch. CloudWatch helps to provide a greater level of visibility into your AWS resource and applications.

You can use CloudWatch to collect and track metrics, which are variables you can measure for your resources and applications. CloudWatch alarms send notifications or automatically make changes to the resources you are monitoring based on rules that you define. For example, you can monitor the CPU usage and disk reads and writes of your Amazon EC2 instances and then use this data to determine whether you should launch additional instances to handle increased load. You can also use this data to stop under-used instances to save money. In addition to monitoring the built-in metrics that come with AWS, you can monitor your own custom metrics. With CloudWatch, you gain system-wide visibility into resource utilization, application performance, and operational health.

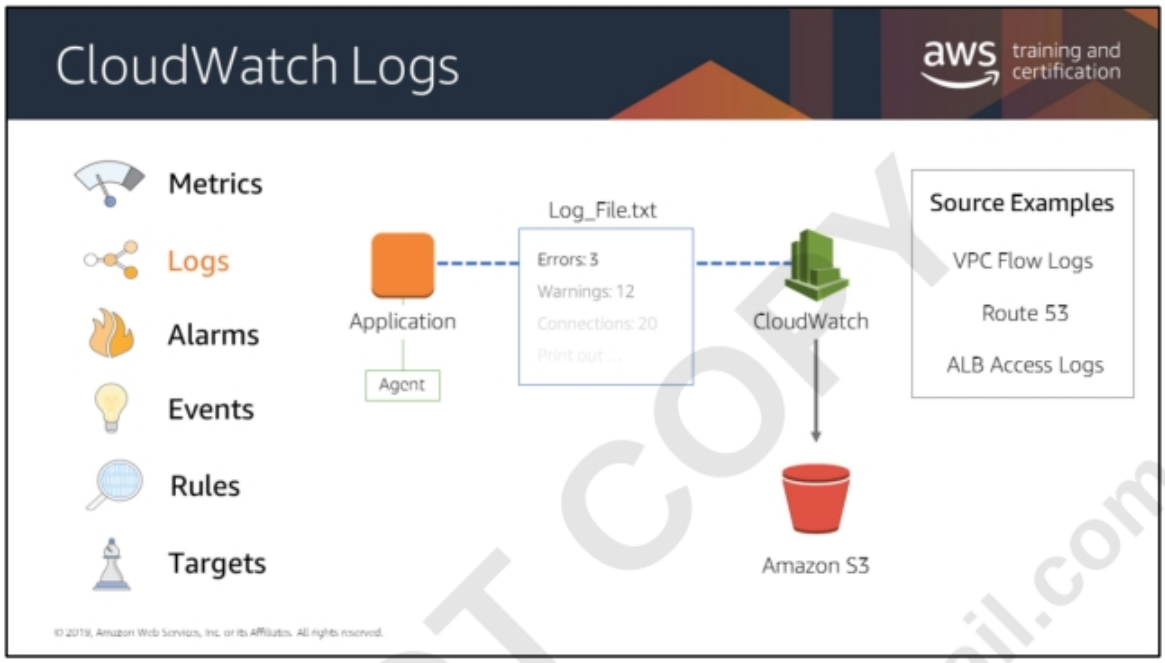
For more information, see https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/WhatIsCloudWatch.html.





Metrics are data about the performance of your systems. Many AWS services provide metrics for resources by default (such as Amazon EC2 instances, Amazon EBS volumes, and Amazon RDS DB instances). You can also enable detailed monitoring some resources, such as your Amazon EC2 instances, or publish your own application metrics. Amazon CloudWatch can load all the metrics in your account (both AWS resource metrics and application metrics that you provide) for search, graphing, and alarms.

/



CloudWatch Logs allows you to monitor, store, and access your log files from sources such as EC2 instances, Amazon Route 53, AWS CloudTrail, and other AWS services.

For example, you could monitor logs from Amazon EC2 instances in real time. You could track the number of errors that have occurred in your application logs and send a notification if that rate exceeds a previously defined amount.

CloudWatch Logs specifically monitors your log data itself, so no code changes are required.

Additionally you can use CloudWatch Logs Insights to analyze your logs in seconds to give you fast, interactive queries and visualizations. You can visualize query results using line or stacked area charts, and add those queries to a CloudWatch Dashboard.

For more information, see https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/WhatIsCloudWatchLogs.html

https://aws.amazon.com/blogs/aws/new-amazon-cloudwatch-logs-insights-fast-interactive-log-analytics/

