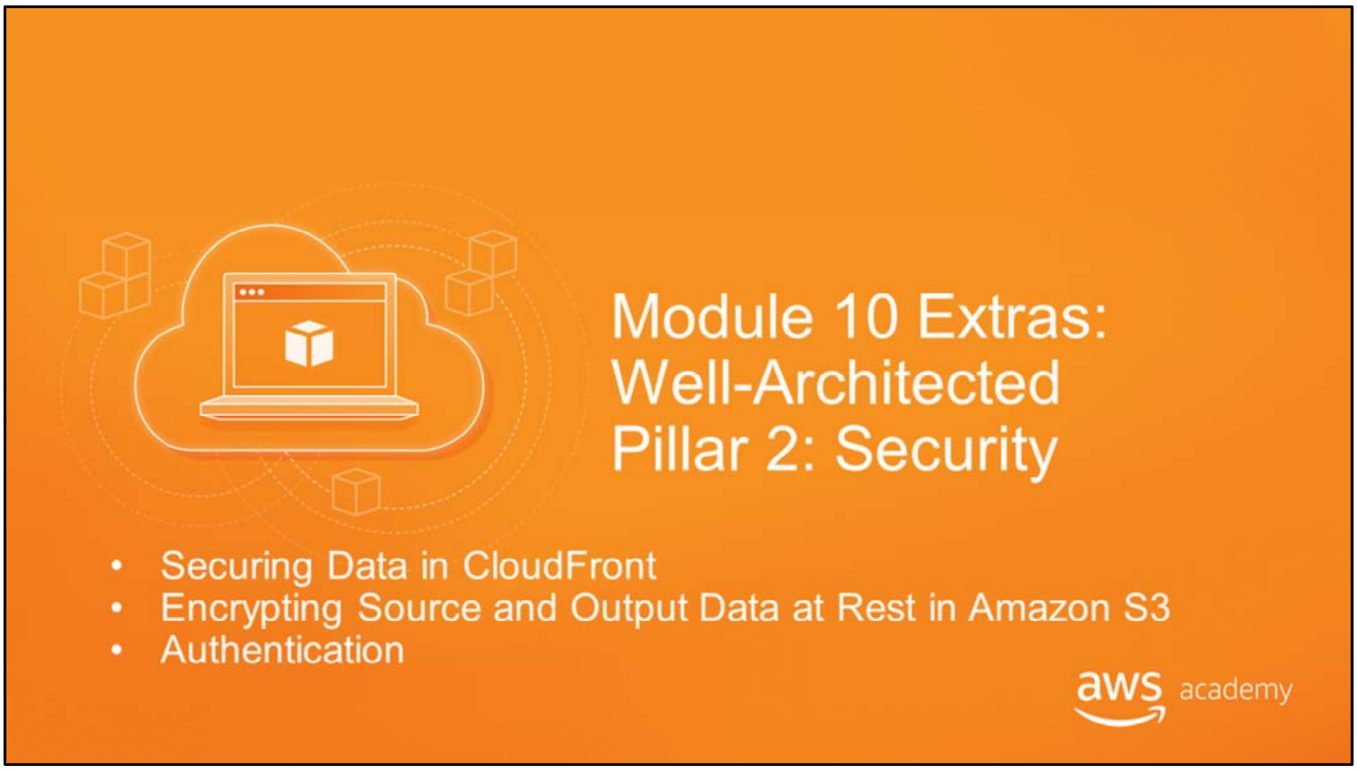


Module 10 Extras: Well-Architected Pillar 2: Security

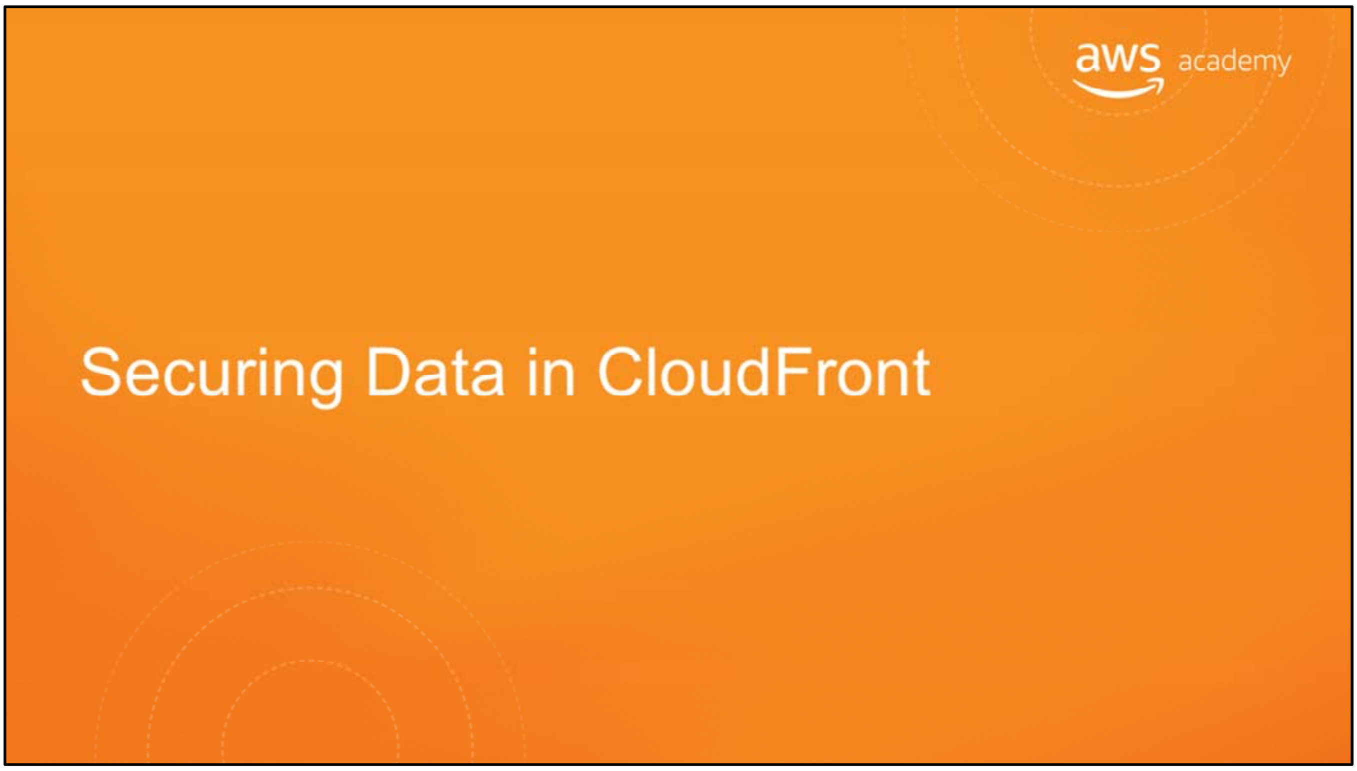
- Securing Data in CloudFront
- Encrypting Source and Output Data at Rest in Amazon S3
- Authentication

aws academy

Welcome to Module 10, where you will learn about the second pillar of the Well-Architected Framework: Security.

This module discusses many best practices, and you will learn how to secure data at every layer in the application. This module also mentions the following tools and services from Amazon Web Services—or AWS—that provide security-focused content:
- AWS Identity and Access Management, or IAM
- AWS Organizations
- Multi-factor authentication (MFA) token
- AWS CloudTrail
- AWS Config
- Amazon CloudWatch
- Amazon GuardDuty
- Amazon Virtual Private Cloud, or Amazon VPC
- Amazon Inspector
- AWS Shield
- AWS WAF
- Amazon Macie
- AWS Key Management Service, or AWS KMS
- Amazon Simple Storage Service, or Amazon S3
- Amazon Elastic Block Store, or Amazon EBS

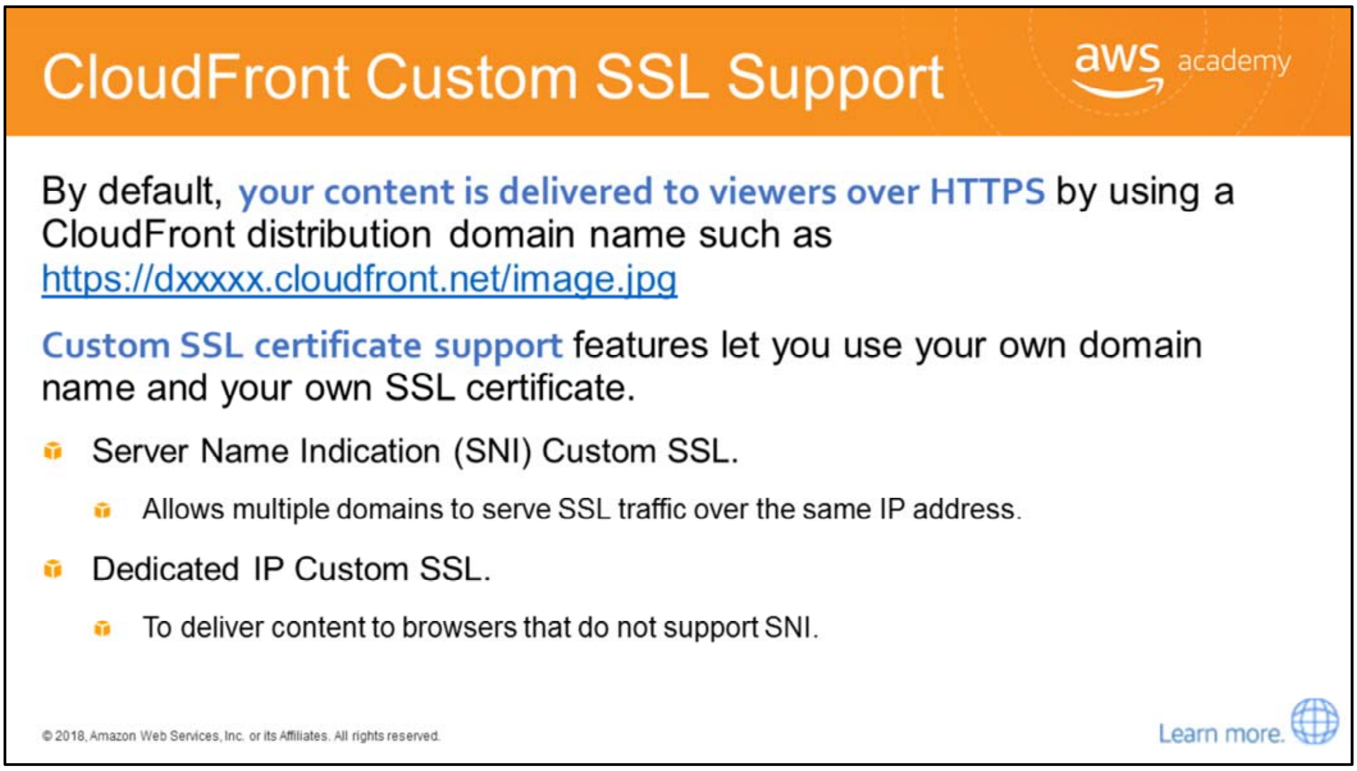
- And AWS CloudFormation



Securing Data in CloudFront

aws academy

Next, let’s discuss securing data in CloudFront.



CloudFront Custom SSL Support

By default, your content is delivered to viewers over HTTPS by using a CloudFront distribution domain name such as https://dxxxxx.cloudfront.net/image.jpg

Custom SSL certificate support features let you use your own domain name and your own SSL certificate.

- Server Name Indication (SNI) Custom SSL.
  - Allows multiple domains to serve SSL traffic over the same IP address.
- Dedicated IP Custom SSL.
  - To deliver content to browsers that do not support SNI.

Learn more.

By default, your content is delivered to viewers over HTTPS by using a CloudFront distribution domain name.

CloudFront has custom Secure Sockets Layer—or SSL—support. You can create your own certificate that will have the **CloudFront.net** domain, or you can bring your own certificate if you have a specific domain name that you want to use.

Server Name Indication—or SNI—Custom SSL relies on the SNI extension of the Transport Layer Security protocol, which allows multiple domains to serve SSL traffic over the same IP address. Amazon CloudFront delivers your content from each edge location, and offers the same security as the Dedicated IP Custom SSL feature.

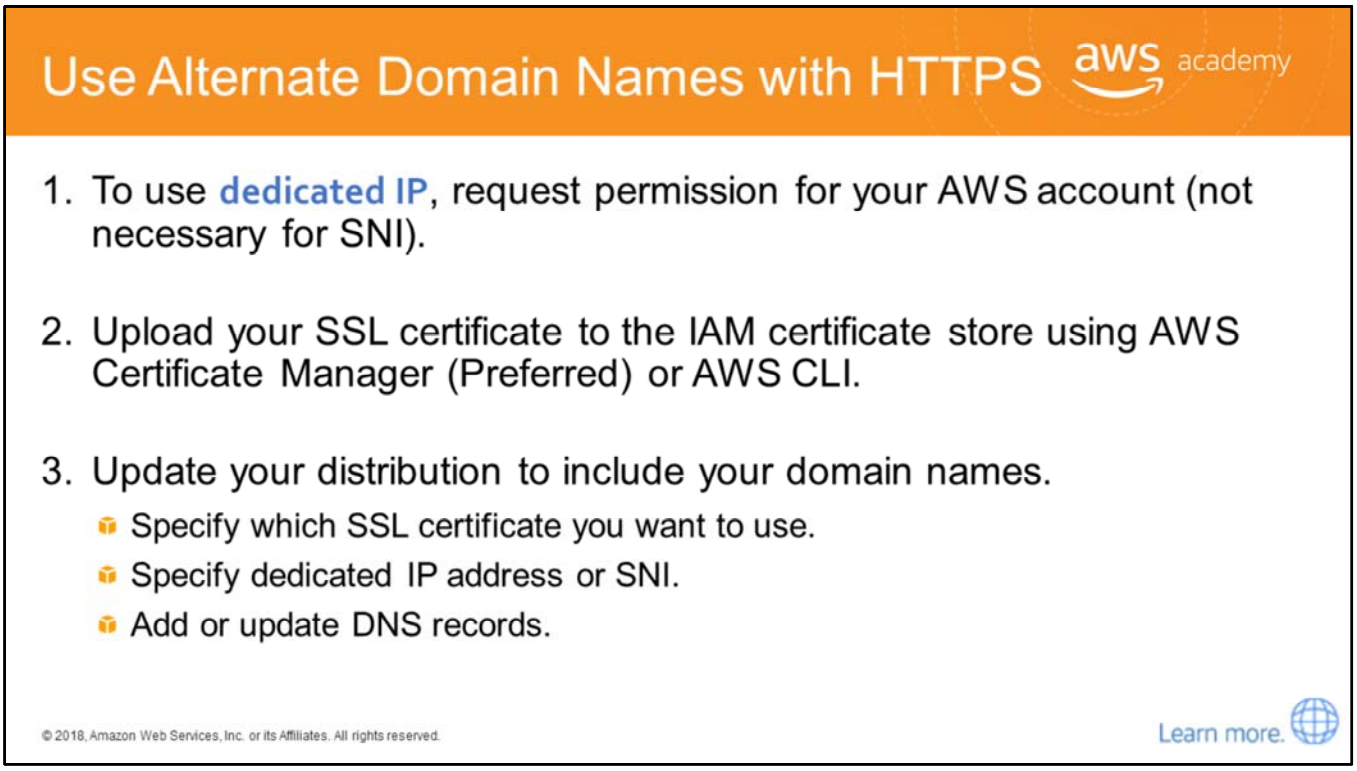
When you use SNI Custom SSL, some users might not be able to access your content if they use older browsers. Some older browsers do not support SNI, and these browsers will not be able to establish a connection with CloudFront to load the HTTPS version of your content. There is no separate pricing for this feature. You can use SNI Custom SSL with no upfront or monthly fees for certificate management. Instead, you pay normal CloudFront rates for data transfer and HTTPS requests.

In contrast, a Dedicated IP Custom SSL works for all clients.
If you need to deliver content to browsers that don't support SNI, you can use the Dedicated IP Custom SSL feature. For this feature, CloudFront allocates

dedicated IP addresses to serve your SSL content at each CloudFront edge location.

To learn more about Dedicated IP Custom SSL certificate support, select the link. http://aws.amazon.com/cloudfront/custom-ssl-domains/

When we approve your request , you can upload an SSL certificate and use the AWS Management Console to associate it with your CloudFront distributions. If you need to associate more than one custom SSL certificate with your CloudFront distribution, include details about your use case and the number of custom SSL certificates that you intend to use in the “Use Case and # of SSL Certs You Intend to Use” section of the form.



Use Alternate Domain Names with HTTPS

1. To use dedicated IP, request permission for your AWS account (not necessary for SNI).

2. Upload your SSL certificate to the IAM certificate store using AWS Certificate Manager (Preferred) or AWS CLI.

3. Update your distribution to include your domain names.
   - Specify which SSL certificate you want to use.
   - Specify dedicated IP address or SNI.
   - Add or update DNS records.

Learn more.

By default, when you request permission to use an alternate domain name with HTTPS, we update your account so that you can associate two custom SSL certificates with your CloudFront distributions. Typically, you only use the second certificate temporarily, such as when you have more than one distribution and you need to rotate certificates. If you need to permanently associate two or more certificates with your distributions, indicate how many certificates you need, and describe your circumstances in your request.

You can upload your SSL certificate to the IAM certificate store by using the AWS Certificate Manager, or ACM. You can also use the AWS Command Line Interface, or AWS CLI.
ACM is the preferred tool for provisioning, managing, and deploying your server certificates. With ACM, you can request a certificate. You can also use ACM to deploy an existing ACM or external certificate to AWS resources. Certificates that are provided by ACM are free, and they automatically renew. You can use ACM to manage server certificates from the console or programmatically.

You can use IAM as a certificate manager only when you must support HTTPS connections in a Region that is not supported by ACM. IAM securely encrypts your private keys, and it stores the encrypted version in IAM SSL certificate storage. IAM supports deploying server certificates in all Regions, but you must obtain your certificate from an external provider for use with AWS. You cannot upload an ACM certificate to IAM.

When you use the IAM CLI to upload your SSL certificate to the IAM certificate store, you must use the same AWS account that you used to create your CloudFront distribution.

When you upload your certificate to IAM, the value of the **-path** parameter, or certificate path, must start with **/cloudfront/**. Examples of the **-path** parameter include **/cloudfront/production/** or **/cloudfront/test/**. The path also must end with a **/**.

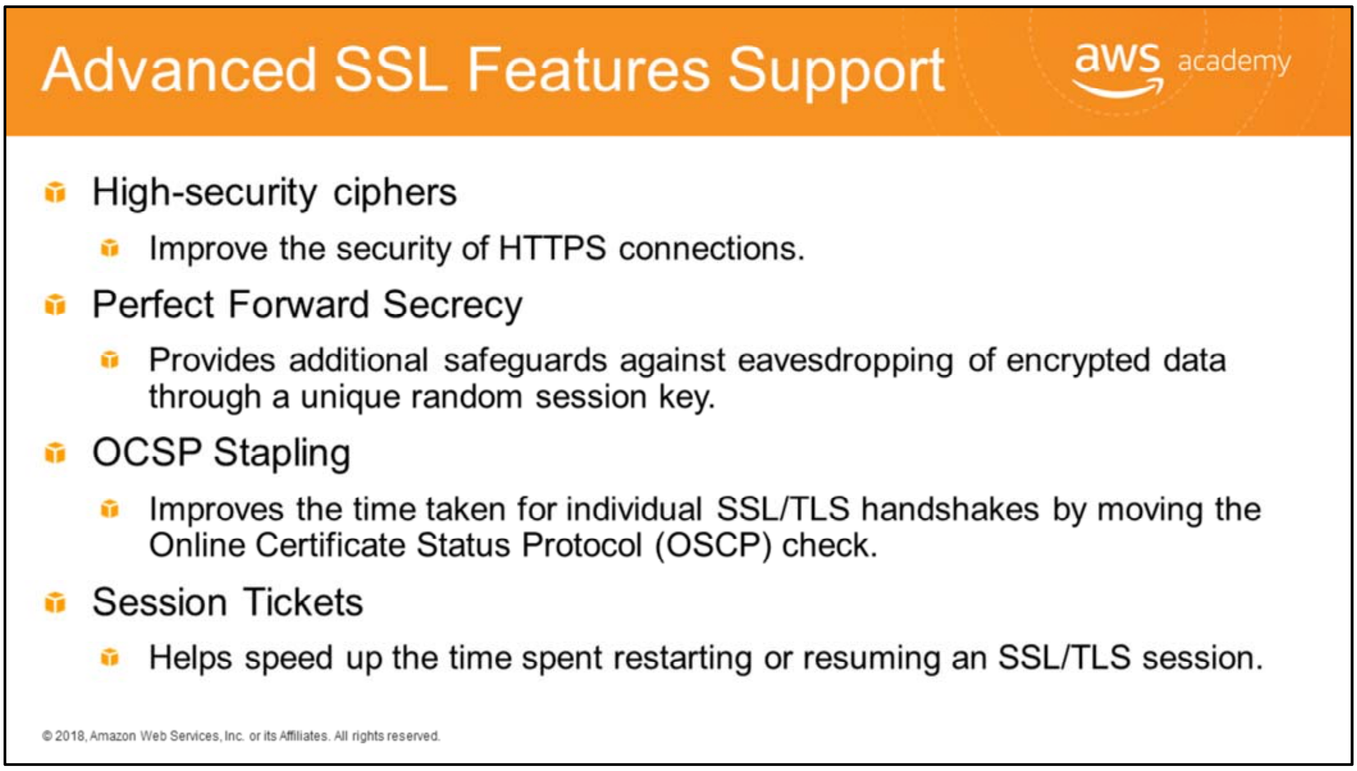
If you plan to use the CloudFront API to create or update your distribution, make sure that you note the alphanumeric string that the AWS CLI returns, such as **AS1A2M3P4L5E67SIIXR3J**. This string is the value that you will specify in the **IAMCertificateId** element. You do not need the IAM Amazon Resource Name—or ARN—which is also returned by the CLI.

You can update your distribution to include your domain names so that you can specify which SSL certificate you want to use, specify a dedicated IP address or SNI, and add or update DNS records.

After you associate your SSL certificate with your CloudFront distribution, do not delete the certificate from the IAM certificate store until you remove the certificate from all distributions, and until the status of the distributions has changed to *Deployed*.

To request permission, select the link.

https://aws.amazon.com/cloudfront/custom-ssl-domains/



Advanced SSL Features Support

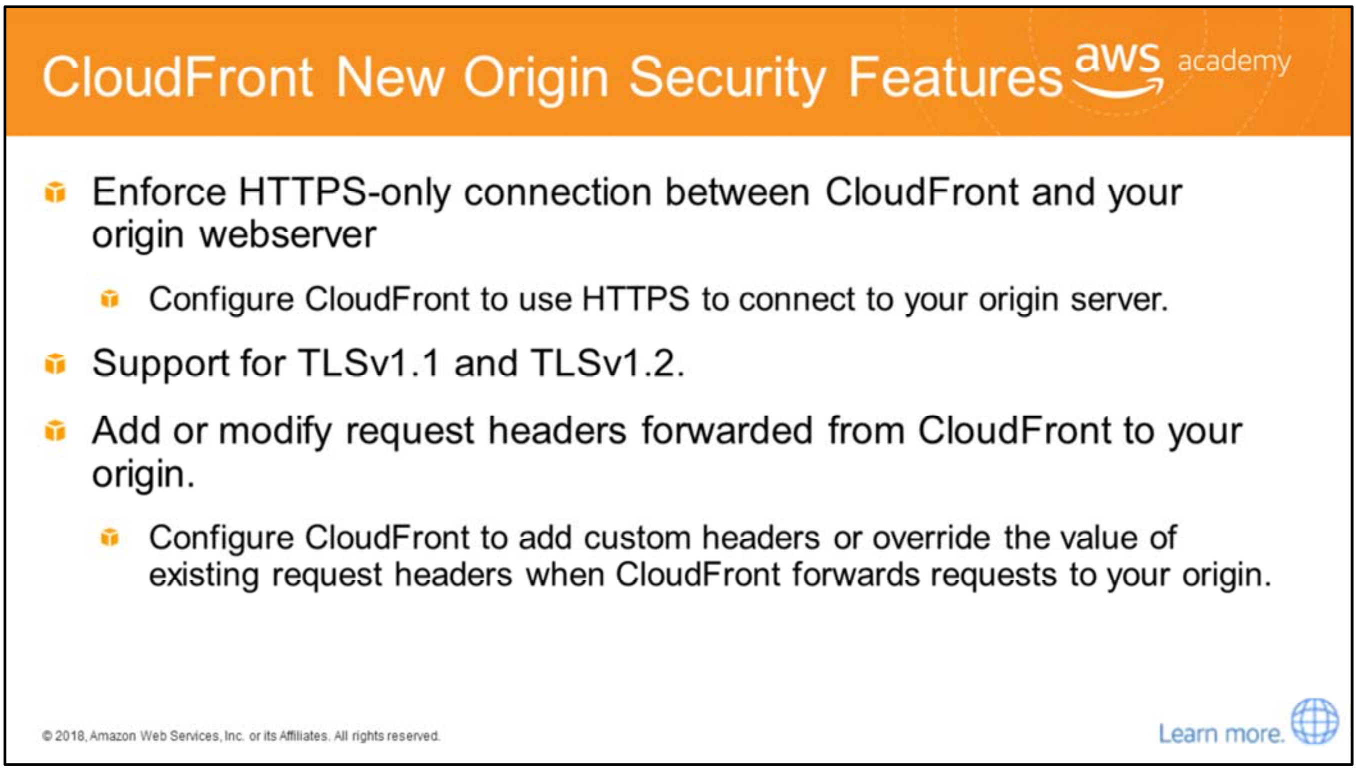
- High-security ciphers
  - Improve the security of HTTPS connections.
- Perfect Forward Secrecy
  - Provides additional safeguards against eavesdropping of encrypted data through a unique random session key.
- OCSP Stapling
  - Improves the time taken for individual SSL/TLS handshakes by moving the Online Certificate Status Protocol (OSCP) check.
- Session Tickets
  - Helps speed up the time spent restarting or resuming an SSL/TLS session.

Support for advanced SSL features include:
High security ciphers to improve the security of HTTPS connections. Amazon CloudFront edge servers and clients, such as browsers, automatically agree on a cipher as part of the SSL handshake process. The connections can now use ciphers with advanced features, such as Elliptic Curve signatures and key exchanges.

Perfect forward secrecy uses a unique random session key to provide additional safeguards against the eavesdropping of encrypted data. This feature prevents the decoding of captured data, even if the secret long-term key is compromised.

OCSP stapling improves the time taken for individual SSL handshakes. It moves the Online Certificate Status Protocol—or OSCP—check, which is a call that is used to obtain the revocation status of an SSL certificate, from the client to a periodic, secure check by the Amazon CloudFront servers. With OCSP stapling, the client no longer needs to handle certificate validation, which improves performance.

Session tickets help speed up the time that is spent restarting or resuming an SSL session. Amazon CloudFront encrypts SSL session information and stores it in a ticket. The client can use this ticket to resume a secure connection instead of repeating the SSL handshake process.


CloudFront New Origin Security Features

- Enforce HTTPS-only connection between CloudFront and your origin webserver
  - Configure CloudFront to use HTTPS to connect to your origin server.
- Support for TLSv1.1 and TLSv1.2.
- Add or modify request headers forwarded from CloudFront to your origin.
  - Configure CloudFront to add custom headers or override the value of existing request headers when CloudFront forwards requests to your origin.

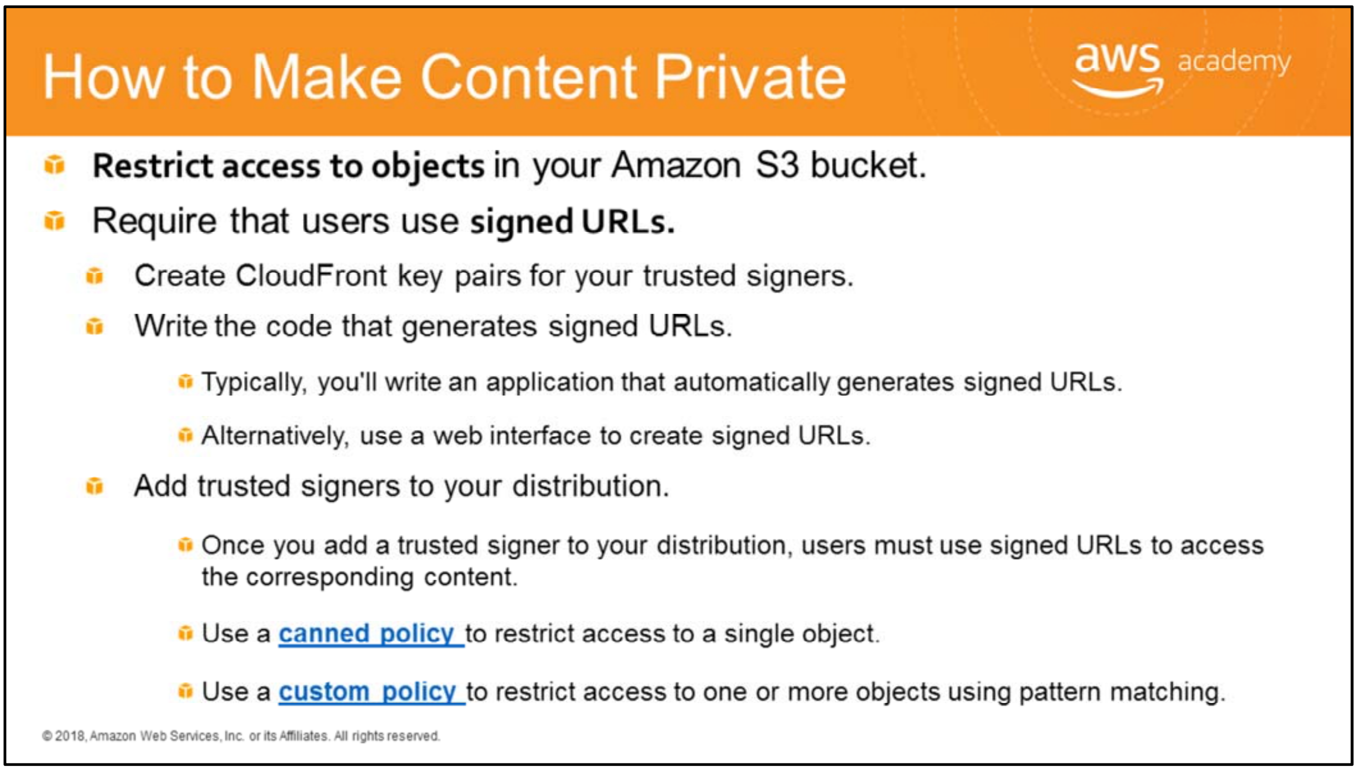
Learn more.

Amazon has announced three new security features that help further protect communication between Amazon CloudFront and your origin server:
You can configure CloudFront to connect to your origin server by using HTTPS, regardless of whether the viewer uses HTTP or HTTPS to make the request.

You can use Transport Layer Security—or TLS—version 1.1 and TLS version 1.2, which were both added to the list of supported protocol versions that you can use for the HTTPS connections between CloudFront and your custom origin web server. In addition, you can choose the protocols that CloudFront uses to communicate with your origin server. For example, you can prevent CloudFront from communicating with your origin server through SSL version 3, which is less secure than TLS.

You can add or modify request headers that are forwarded from CloudFront to your origin server. You can configure CloudFront to add custom headers, or override the value of existing request headers, when CloudFront forwards requests to your origin server. You can use these headers to help validate that the requests that were made to your origin server were sent from CloudFront, which is a shared secret. You can also configure your origin server to only allow requests that contain the custom header values that you specify. This feature also helps you set up cross-origin request sharing—or CORS—for your CloudFront distribution. You can configure CloudFront to always add custom headers to your origin to accommodate viewers that don't automatically include those headers in requests. You can also disable varying on the origin header, which improves your cache hit ratio, and still forward the appropriate headers so that your origin server can respond with the CORS header.

For more information, go to the CloudFront documentation about requiring HTTPS for communication:
http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/using-https-viewers-to-cloudfront.html


How to Make Content Private

- **Restrict access to objects** in your Amazon S3 bucket.
- Require that users use **signed URLs.**
  - Create CloudFront key pairs for your trusted signers.
  - Write the code that generates signed URLs.
    - Typically, you'll write an application that automatically generates signed URLs.
    - Alternatively, use a web interface to create signed URLs.
  - Add trusted signers to your distribution.
    - Once you add a trusted signer to your distribution, users must use signed URLs to access the corresponding content.
    - Use a canned policy to restrict access to a single object.
    - Use a custom policy to restrict access to one or more objects using pattern matching.

To make content private, you can restrict access to objects in your Amazon S3 bucket.

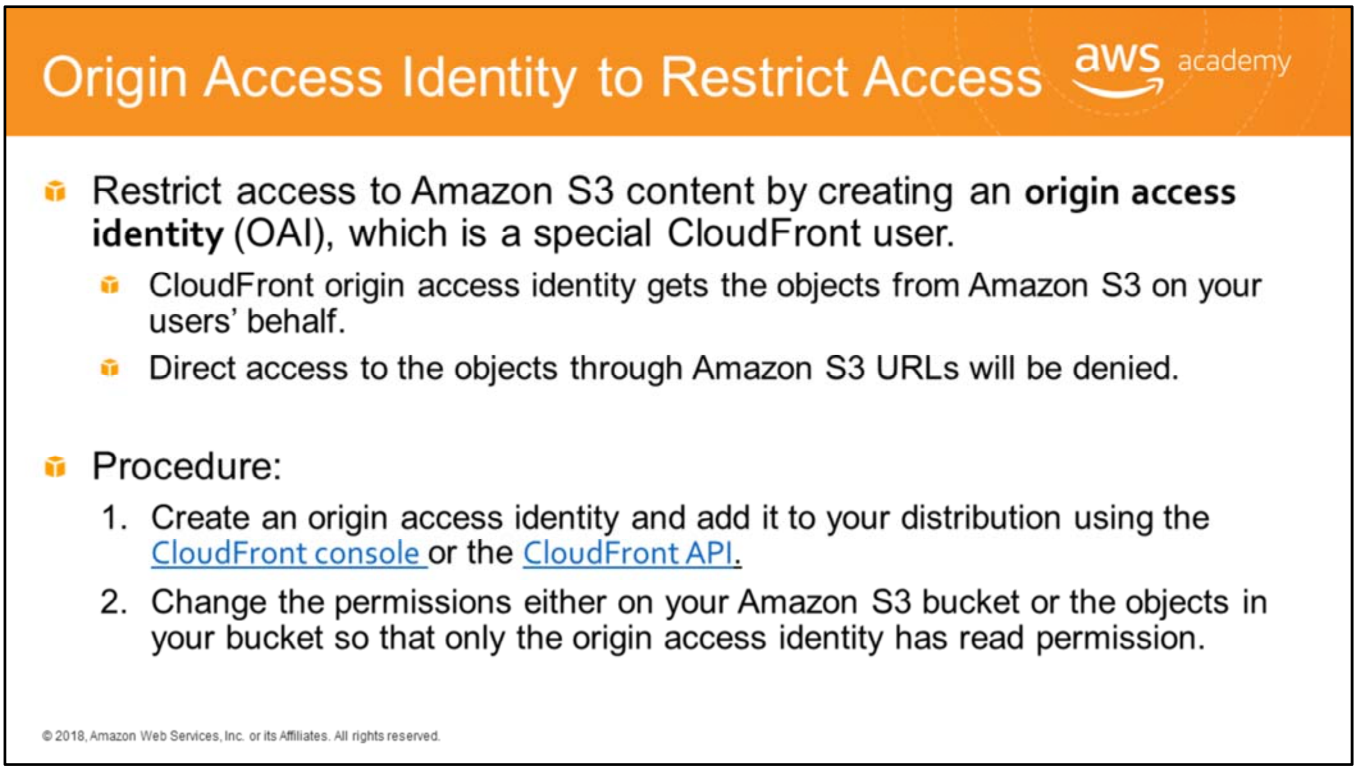
You can also require that users use signed URLs. You can create CloudFront key pairs for your trusted signers, and write the code that generates signed URLs. Typically, you will write an application that automatically generates signed URLs. Alternatively, you can use a web interface to create signed URLs.

You can add trusted signers to your distribution. After you add a trusted signer to your distribution, users must use signed URLs to access the corresponding content.

A signed URL includes additional information, such as an expiration date and time, which gives you more control over access to your content. This additional information appears in a policy statement that is based on either a **canned** policy or a **custom** policy:

- Use a **canned policy** to restrict access to a single object.
- Use a **custom policy** to restrict access to one or more objects by using pattern matching.

Select the links to learn more about each type of policy.
Canned policy:
http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/private-content-creating-signed-url-canned-policy.html
Custom policy:
http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/private-content-creating-signed-url-custom-policy.html



Origin Access Identity to Restrict Access

aws academy

- Restrict access to Amazon S3 content by creating an **origin access identity** (OAI), which is a special CloudFront user.
  - CloudFront origin access identity gets the objects from Amazon S3 on your users' behalf.
  - Direct access to the objects through Amazon S3 URLs will be denied.
- Procedure:
  1. Create an origin access identity and add it to your distribution using the CloudFront console or the CloudFront API.
  2. Change the permissions either on your Amazon S3 bucket or the objects in your bucket so that only the origin access identity has read permission.

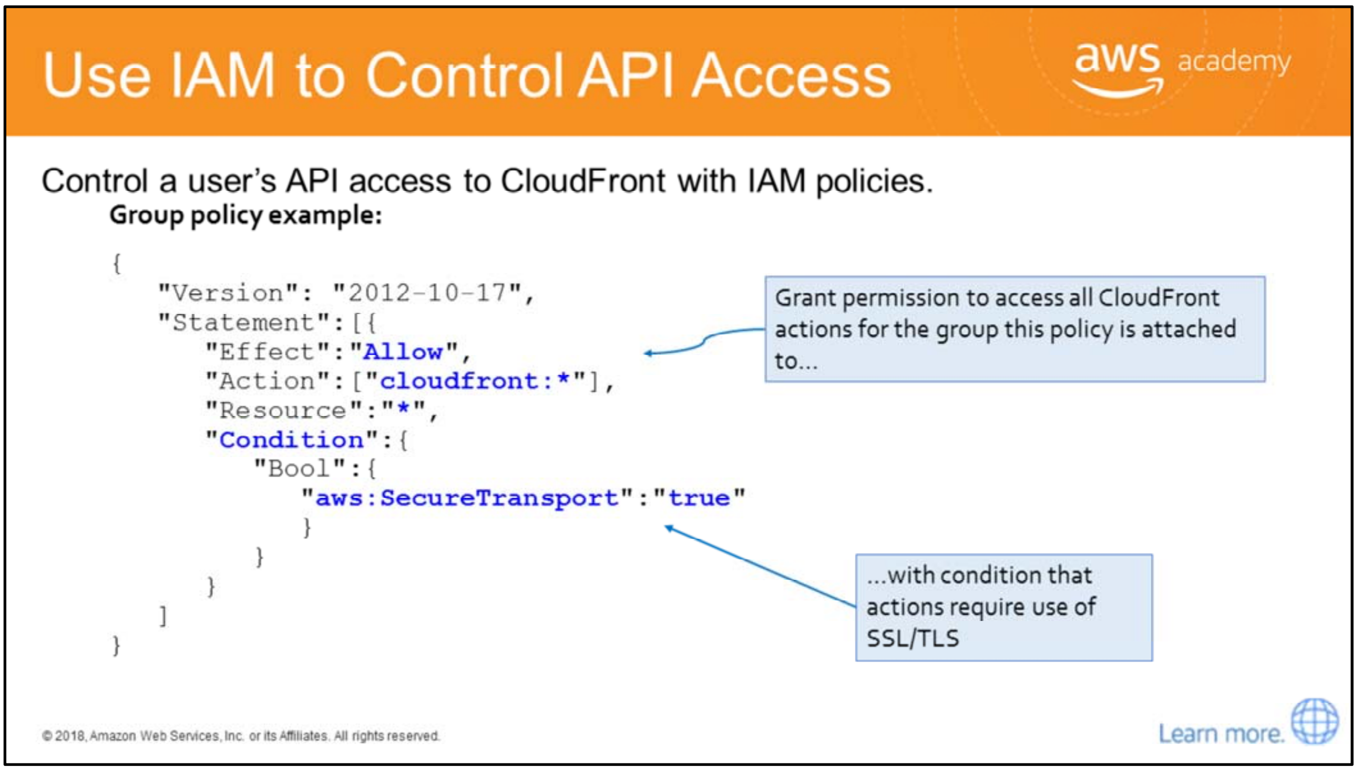
You can restrict access to Amazon S3 content by creating an **origin access identity,** or OAI, which is a special CloudFront user.
CloudFront origin access identity gets objects from Amazon S3 on behalf of your users. Direct access to the objects through Amazon S3 URLs will be denied.

To use this feature, you first create an origin access identity, and then add it to your distribution. You then change the permissions either on your Amazon S3 bucket or the objects in your bucket so that only the origin access identity has read permission.

You can create an origin access identity by using the CloudFront console or the CloudFront API.

To learn more about the CloudFront console and the CloudFront API, select the links.

- **CloudFront console**: http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/private-content-restricting-access-to-s3.html#private-content-creating-oai-console
- **CloudFront API**: http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/private-content-restricting-access-to-s3.html#private-content-creating-oai-api



Use IAM to Control API Access

Control a user's API access to CloudFront with IAM policies.

Group policy example:

```
{
   "Version": "2012-10-17",
   "Statement":[{
      "Effect":"Allow",
      "Action":["cloudfront:*"],
      "Resource":"*",
      "Condition":{
         "Bool":{
            "aws:SecureTransport":"true"
            }
         }
      }
   ]
}
```

Grant permission to access all CloudFront actions for the group this policy is attached to…

…with condition that actions require use of SSL/TLS

Learn more.

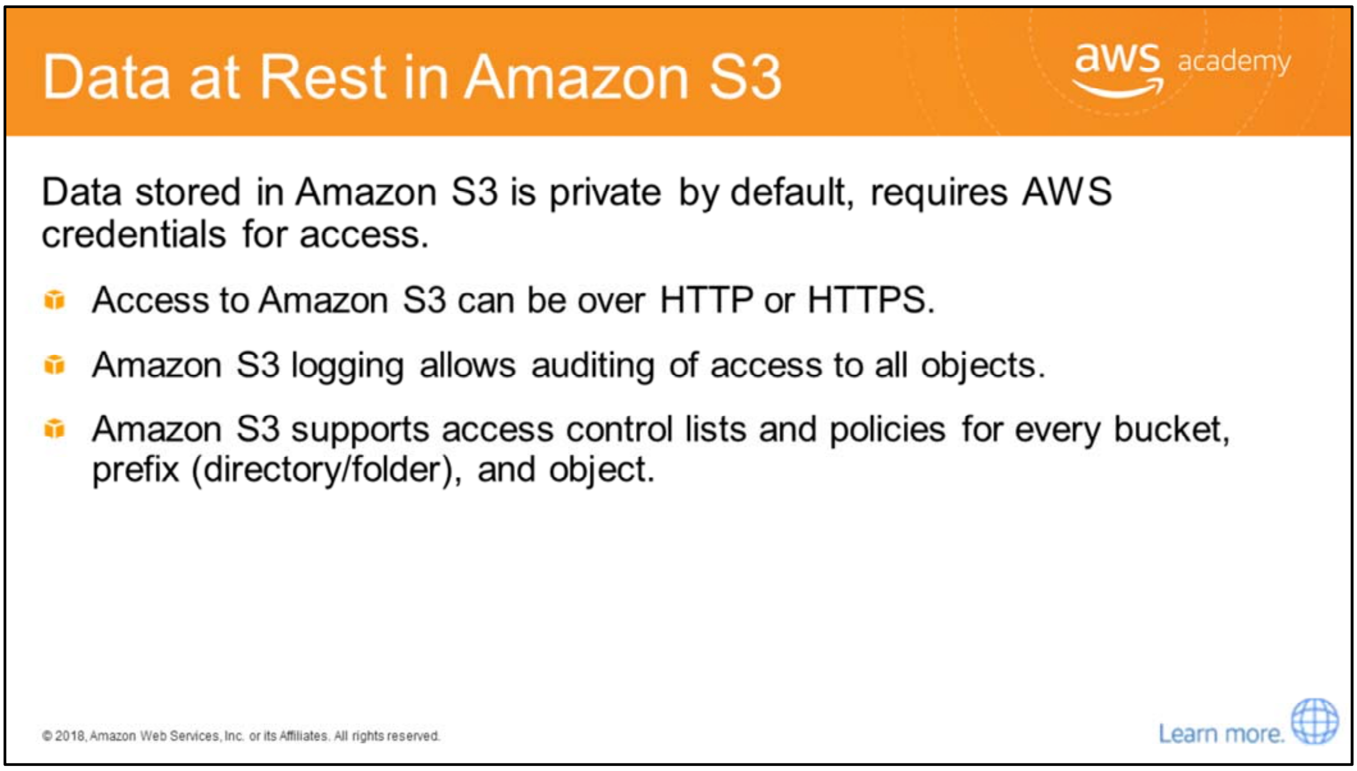
Displayed is an example of an IAM policy that enforces SSL. It grants application permission to access all of CloudFront. However, it also requires secure transport, which means you must use SSL or TLS.

In an IAM policy, you can specify any and all API actions that CloudFront offers. The action name must be prefixed with the lowercase string **cloudfront:**. An example of an action name is: **cloudfront:GetDistributionConfig**

To learn more about AWS Authentication and Access Control, select the link.
http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/UsingWithIAM.html

Encrypting Source and Output Data at Rest in Amazon S3

Now, let’s discuss encrypting source and output data at rest in Amazon S3.



Data at Rest in Amazon S3

aws academy

Data stored in Amazon S3 is private by default, requires AWS credentials for access.

- Access to Amazon S3 can be over HTTP or HTTPS.
- Amazon S3 logging allows auditing of access to all objects.
- Amazon S3 supports access control lists and policies for every bucket, prefix (directory/folder), and object.

Learn more.

Data that you store in Amazon S3 is private by default, and it requires AWS credentials for access.
You can access Amazon S3 over HTTP or HTTPS.
Amazon S3 logging allows you to audit access to all objects.

Server-side encryption is about data encryption at rest—that is, Amazon S3 encrypts your data as it writes it to disks in its data centers, and it decrypts your data for you when you access it. If you authenticate your request and have access permissions, there is no difference in the way you access encrypted or unencrypted objects. Amazon S3 manages encryption and decryption for you. For example, if you share your objects by using a pre-signed URL, the pre-signed URL works the same way for both encrypted and unencrypted objects.

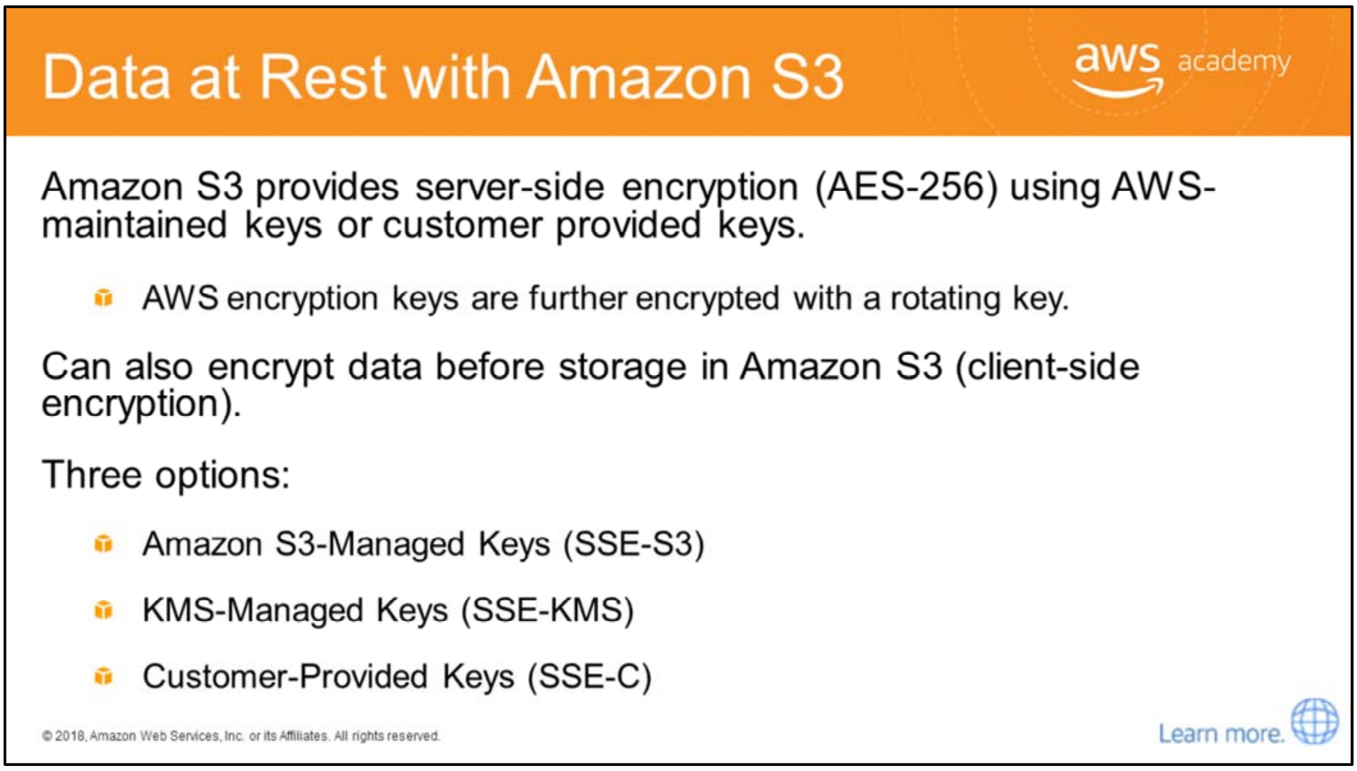
In client-side encryption, you manage the encryption and decryption of your data, the encryption keys, and any related tools. Server-side encryption is an alternative to client-side encryption. In this model, Amazon S3 manages the encryption of your data, which frees you from the tasks of managing encryption and encryption keys.

Server-side encryption with Amazon S3-managed keys—or SSE-S3—employs strong multi-factor encryption. With SSE-S3, Amazon S3 encrypts each object with a unique key. As an additional safeguard, it encrypts the key itself with a master key that it regularly rotates. SSE-S3 uses one of the strongest block ciphers available—256bit Advanced Encryption Standard, or AES-256—to encrypt your data.
You can encrypt data by using any encryption method you want, and then upload the encrypted data using the Amazon S3 APIs. The most common application languages include cryptographic libraries that allow you to perform encryption in your applications. Two commonly available open source tools are Bouncy Castle and OpenSSL.

It supports access control lists and policies for every bucket, prefix—such as a directory or folder—and object.
After you encrypt an object and safely store the key in your KMI, the encrypted object can be uploaded to Amazon S3 directly with a PUT request. To decrypt this data, you issue the GET request in the Amazon S3 API, and then pass the encrypted data to your local application for decryption.

AWS provides an alternative to these open source encryption tools with the Amazon S3 encryption client, which is an open source set of APIs that are embedded into the AWS SDKs. This client allows you to supply a key from your KMI that can be used to encrypt or decrypt your data as part of the API call to Amazon S3. The Amazon S3 encryption client is integrated into the AWS SDKs for Java, Ruby, and .NET. It provides a transparent drop-in replacement for any cryptographic code that you might have used previously with your application that interacts with Amazon S3. Although Amazon provides the encryption method, you control the security of your data because you control the keys for that engine to use. If you use the Amazon S3 encryption client on-premises, we never have access to your keys or unencrypted data. If you use the client in an application that runs in Amazon EC2, a best practice is to pass keys to the client by using secure transport—such as SSL or SSH—from your KMI to help ensure confidentiality.

To learn more, select the link to view the AWS whitepaper on securing data at rest with encryption.
https://aws.amazon.com/whitepapers/



Data at Rest with Amazon S3

Amazon S3 provides server-side encryption (AES-256) using AWS-maintained keys or customer provided keys.

- AWS encryption keys are further encrypted with a rotating key.

Can also encrypt data before storage in Amazon S3 (client-side encryption).

Three options:

- Amazon S3-Managed Keys (SSE-S3)
- KMS-Managed Keys (SSE-KMS)
- Customer-Provided Keys (SSE-C)

Learn more.

Amazon S3 provides server-side encryption by using AES-256. Server-side encryption can use AWS-maintained keys or customer-provided keys.
AWS encryption keys are further encrypted with a rotating key.
For client-side encryption, Amazon S3 also allows you to encrypt data before you store it in Amazon S3.

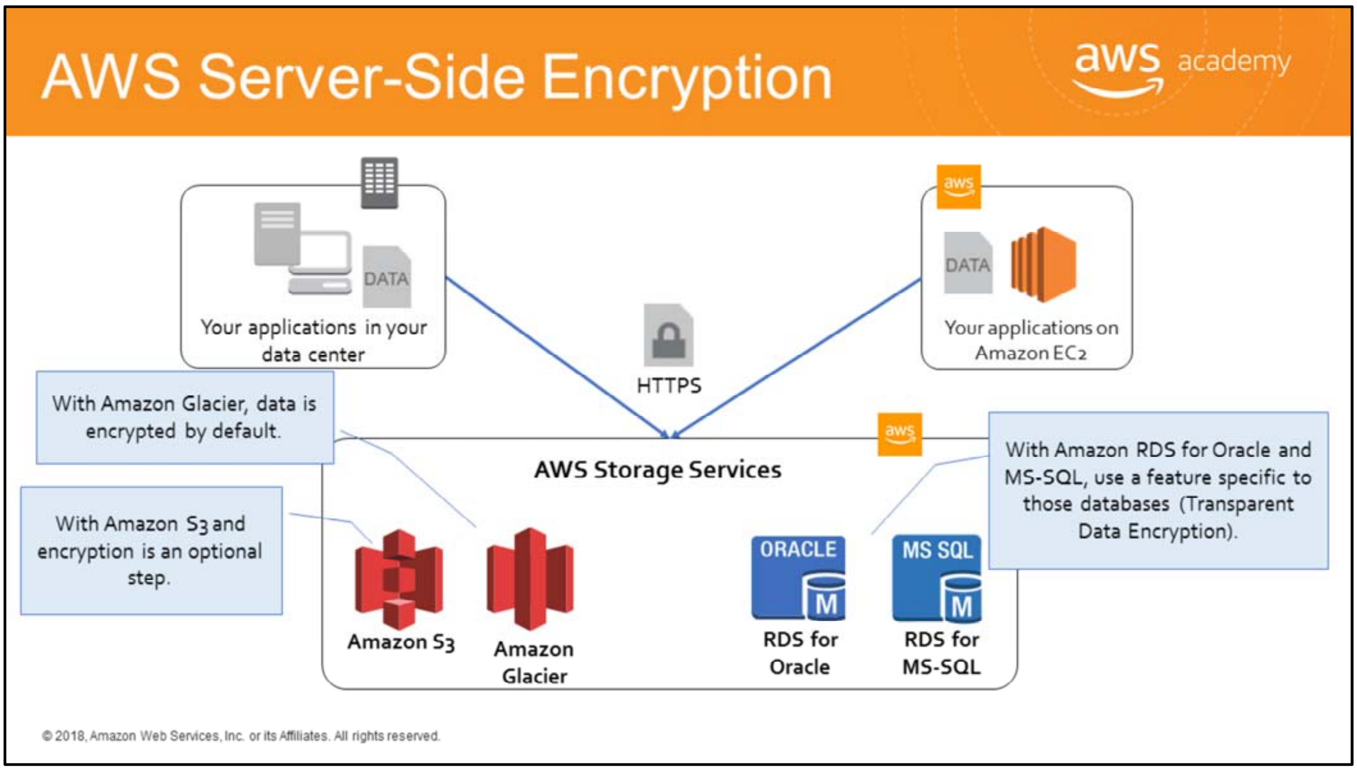
You have three mutually exclusive options, depending on how you choose to manage encryption keys:

• When you want to encrypt each object with a unique key that employs strong multi-factor encryption, use SSE-S3 with managed keys.

• Server-side encryption with AWS KMS-managed keys (SSE-KMS) provides a service that is similar to SSE-S3. It includes some additional benefits, and it also includes some additional charges for using this service.

• Use server-side encryption with customer-provided keys (SSE-C) when you want to manage the encryption keys. Amazon S3 manages the encryption as it writes to disks, and it also manages the decryption when you access your objects.

For more information about Amazon S3 server-side encryption, select the link.
https://docs.aws.amazon.com/AmazonS3/latest/dev/serv-side-encryption.html



AWS Server-Side Encryption

With Amazon Glacier, data is encrypted by default.

With Amazon S3 and encryption is an optional step.

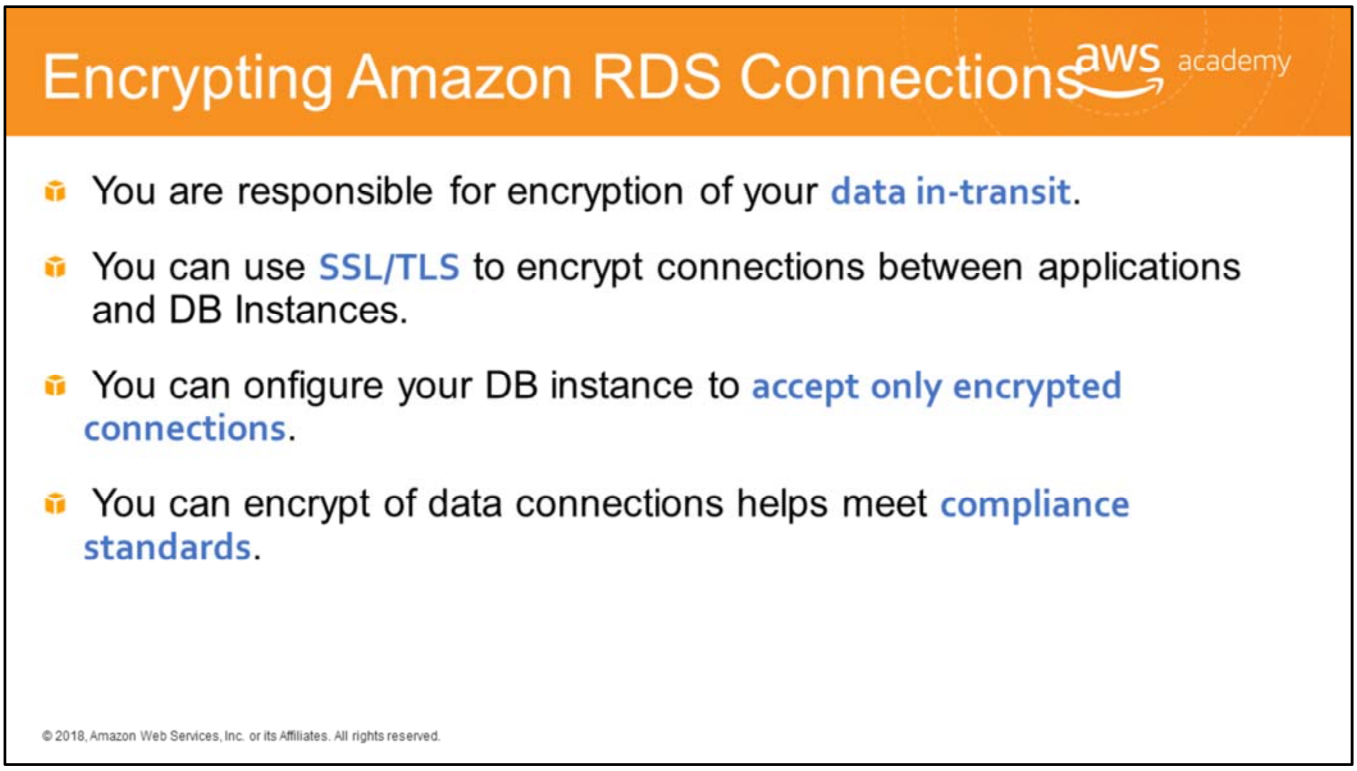
With Amazon RDS for Oracle and MS-SQL, use a feature specific to those databases (Transparent Data Encryption).

Displayed is an example of how server-side encryption works in AWS. In this diagram, you transmit data to Amazon S3 either from your on-premises data center or from an Amazon EC2 instance, if you are using HTTPS. You can choose to turn on encryption when you upload your data to the service.

Take note that in Amazon S3, encryption is not enabled by default.

With Amazon Glacier, all data is encrypted by default.

Amazon RDS for Oracle and Microsoft SQL uses a feature that is specific to those database packages, which is called Transparent Data Encryption, or TDE. TDE protects your data by using keys that are created in the database application with keys that are created by AWS.



Encrypting Amazon RDS Connections

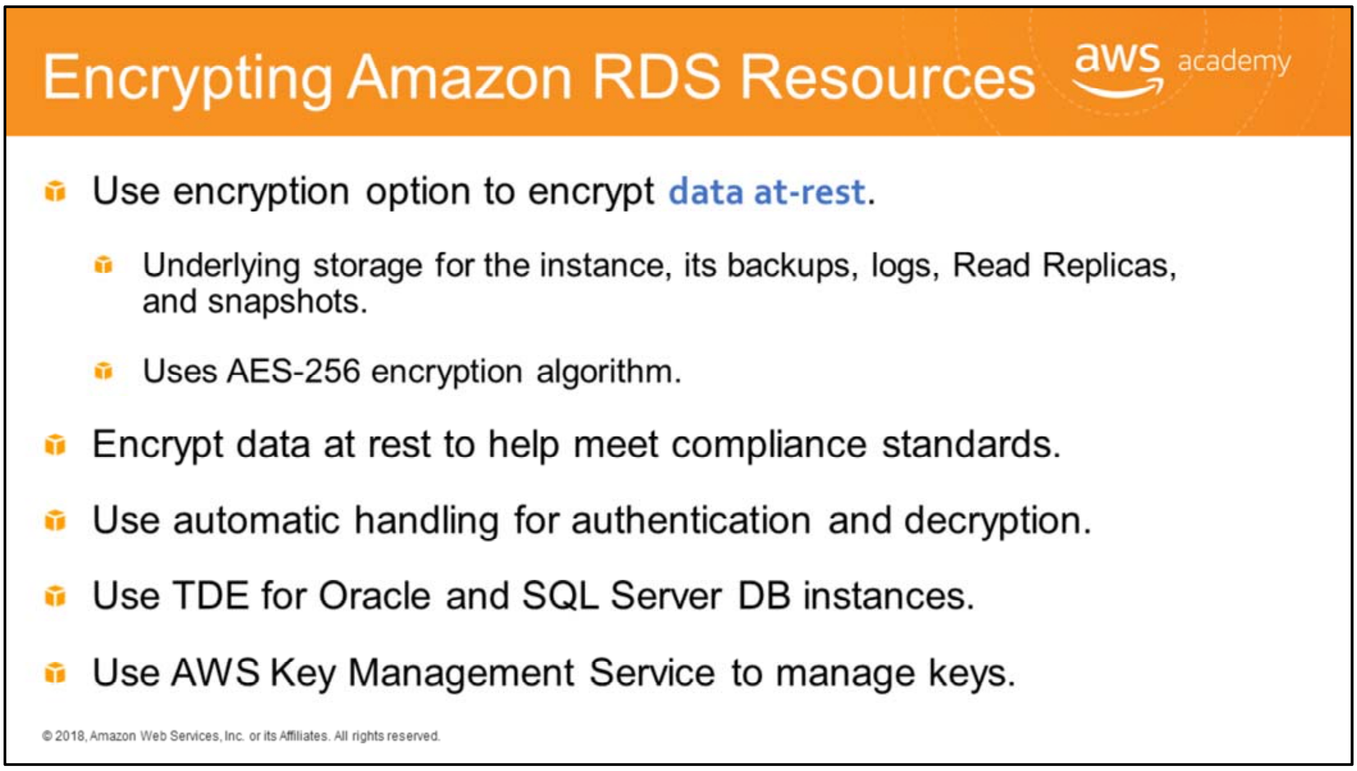
- You are responsible for encryption of your data in-transit.
- You can use SSL/TLS to encrypt connections between applications and DB Instances.
- You can onfigure your DB instance to accept only encrypted connections.
- You can encrypt of data connections helps meet compliance standards.

Keep in mind that when you connect to your relational database, you need to make sure the data is both encrypted on the server side in storage, and in transit.

You can use SSL/TLS to encrypt connections between your application and your database—or DB—instances. For MySQL and SQL Server, Amazon RDS creates an SSL certificate, and it installs the certificate on the DB instance when the instance is provisioned. When you want to encrypt connections in MySQL, launch the MySQL client by using the --ssl_ca parameter to reference the public key. For SQL Server, download the public key, and import the certificate into your Windows operating system. Oracle RDS uses Oracle native network encryption with a DB instance. You add the native network encryption option to an option group, and associate that option group with the DB instance. After an encrypted connection is established, data that is transferred between the DB instance and your application will be encrypted during the transfer. You can also require your DB instance to only accept encrypted connections.

The encryption of data connections helps meet compliance standards.
SSL and TLS support within Amazon RDS is used to encrypt the connection between your application and your DB instance. You should not rely on SSL and TLS for authenticating the DB instance itself.

While SSL and TLS offers security benefits, be aware that the encryption is a compute-intensive operation, and it will increase the latency of your database connection.



Encrypting Amazon RDS Resources

- Use encryption option to encrypt **data at-rest**.
  - Underlying storage for the instance, its backups, logs, Read Replicas, and snapshots.
  - Uses AES-256 encryption algorithm.
- Encrypt data at rest to help meet compliance standards.
- Use automatic handling for authentication and decryption.
- Use TDE for Oracle and SQL Server DB instances.
- Use AWS Key Management Service to manage keys.

You can encrypt your Amazon RDS instances and snapshots at rest by enabling the encryption option for your Amazon RDS DB instance. Data that is encrypted at rest includes the underlying storage for a DB instance, its automated backups, Read Replicas, and snapshots.

Amazon RDS-encrypted instances use the industry standard AES-256 encryption algorithm to encrypt your data on the server that hosts your Amazon RDS instance.

When your data is encrypted, Amazon RDS handles authenticating access and decrypting your data transparently, with a minimal impact on performance. You don't need to modify your database client applications to use encryption.

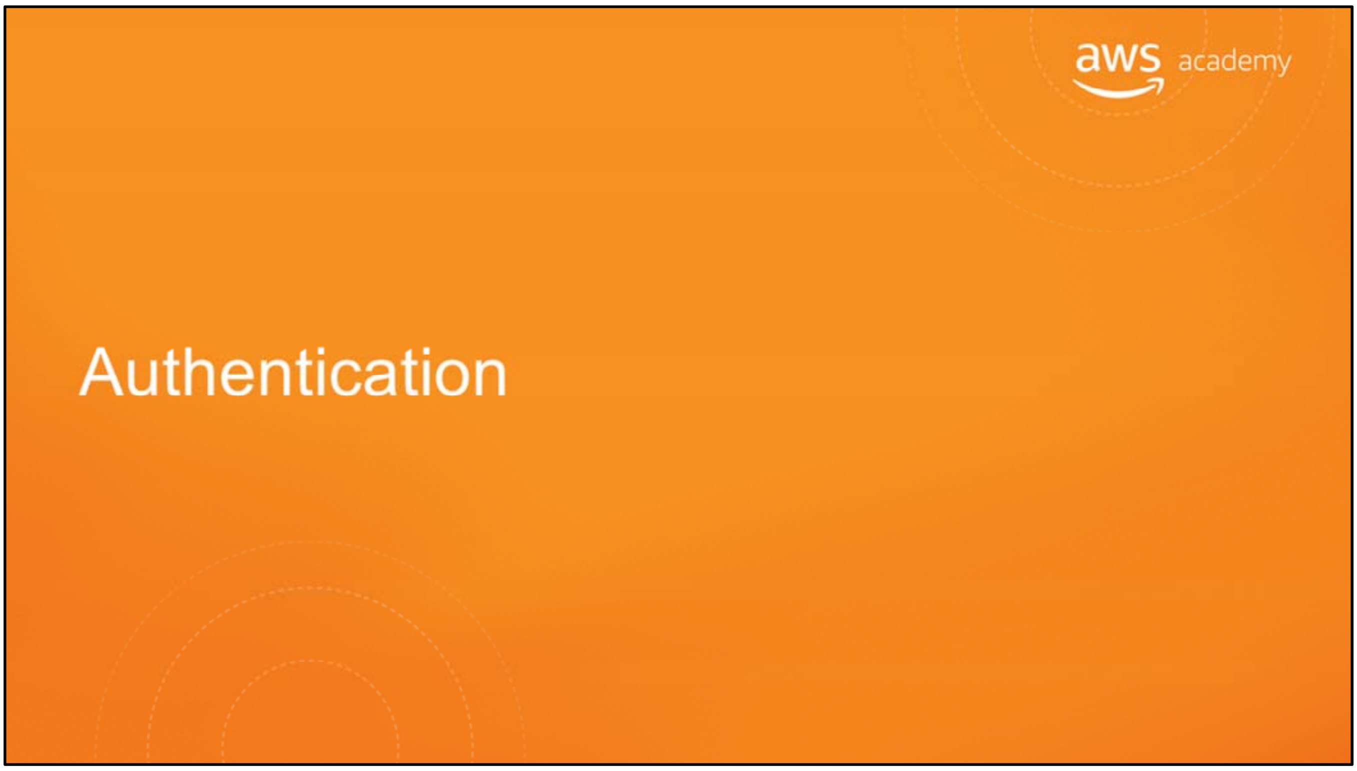
Amazon RDS-encrypted instances provide an additional layer of data protection by securing your data from unauthorized access to the underlying storage. You can use Amazon RDS encryption to increase data protection of your applications and to fulfill compliance requirements for data-at-rest encryption.

Amazon RDS-encrypted instances are currently available for MySQL, PostgreSQL, Oracle, and SQL Server DB instances.

Amazon RDS also supports encrypting an Oracle or SQL Server DB instance with Transparent Data Encryption, or TDE. TDE can be used in conjunction with encryption at rest, although using TDE and encryption at rest simultaneously might slightly affect the performance of your database. You must manage different keys for each encryption method. After you have created an encrypted DB instance, you cannot change the encryption key for that instance. Therefore, be sure to determine your encryption key requirements before you create your encrypted DB instance.

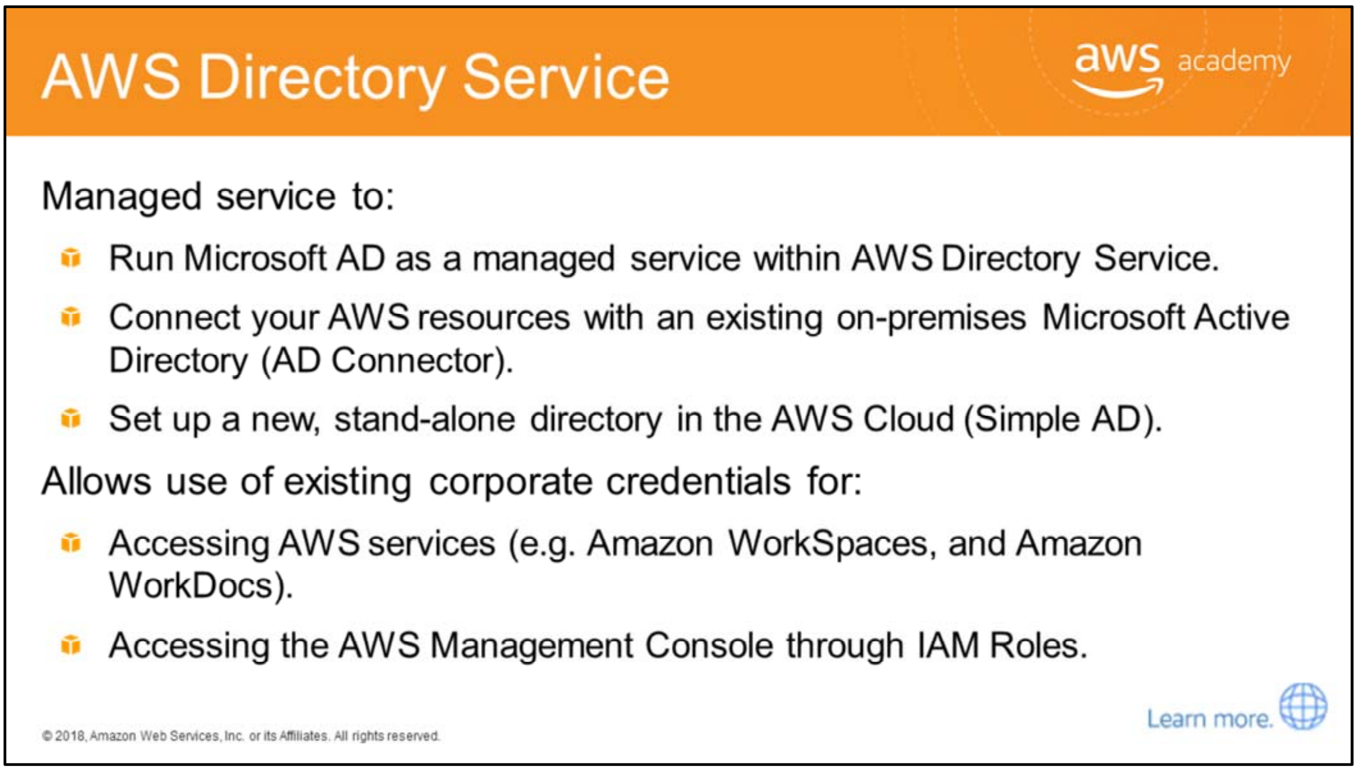
You can use AWS KMS to manage the keys that you use for encrypting and decrypting your Amazon RDS resources. AWS KMS combines secure, highly available hardware and software to provide a key management system that is scaled for the cloud. By using AWS KMS, you can create encryption keys and define the policies that control how these keys can be used. AWS KMS supports AWS CloudTrail, so you can audit key usage to verify that keys are being used appropriately.

All logs, backups, and snapshots for an Amazon RDS-encrypted instance are encrypted. A Read Replica of an Amazon RDS-encrypted instance is also encrypted by using the same key as the master instance.



Authentication

Now, let’s review best practices for authentication.



AWS Directory Service

Managed service to:

- Run Microsoft AD as a managed service within AWS Directory Service.
- Connect your AWS resources with an existing on-premises Microsoft Active Directory (AD Connector).
- Set up a new, stand-alone directory in the AWS Cloud (Simple AD).

Allows use of existing corporate credentials for:

- Accessing AWS services (e.g. Amazon WorkSpaces, and Amazon WorkDocs).
- Accessing the AWS Management Console through IAM Roles.

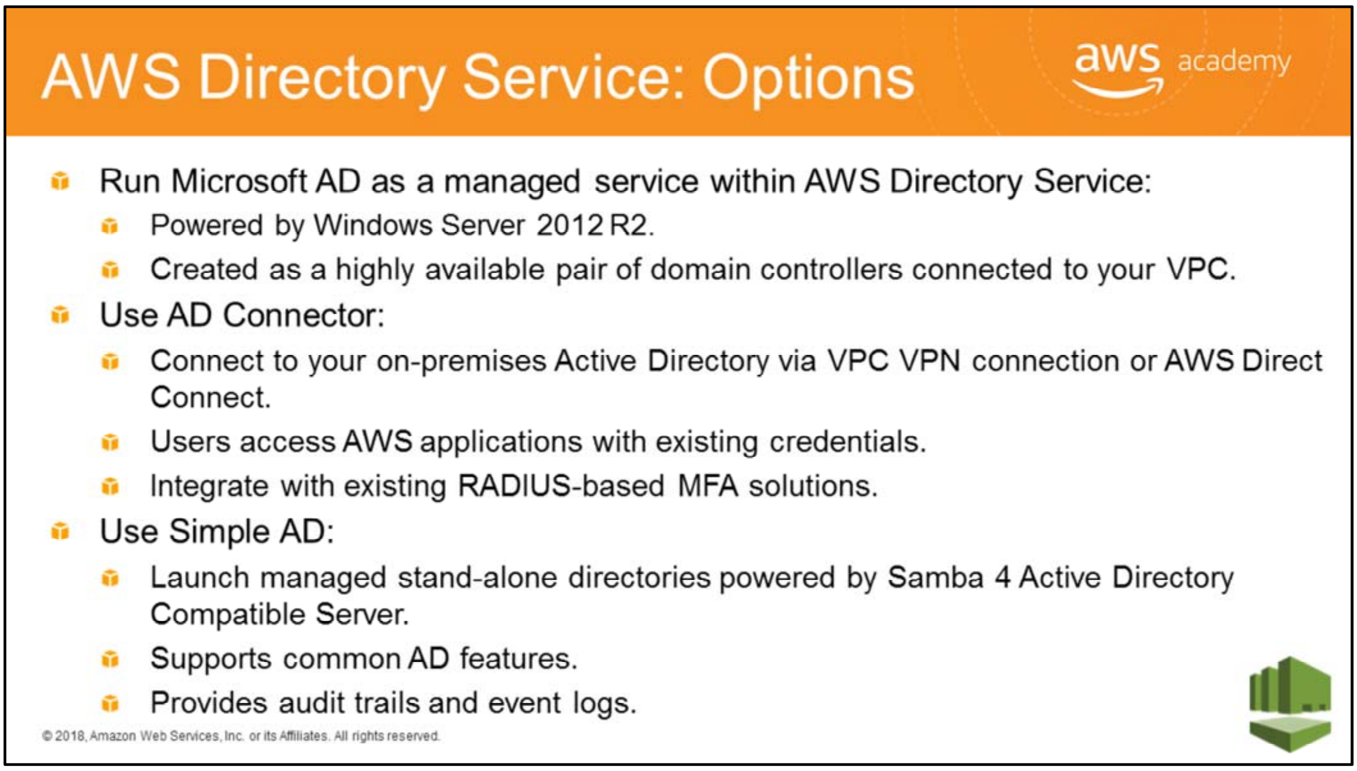
Learn more.

AWS Directory Service is a managed service that you can use to run Microsoft Active Directory within AWS Directory Service. It allows you to connect your AWS resources in an existing on-premises Microsoft Active Directory (also known as AD Connector).

You can also set up a new, stand-alone directory in the AWS Cloud. Simple AD offers this solution.

You can use IAM roles to allow the use of existing corporate credentials for accessing AWS services—such as Amazon WorkSpaces and Amazon WorkDocs—and the AWS Management Console.

Select the link to learn more about the AWS Directory Service.
https://aws.amazon.com/directoryservice/



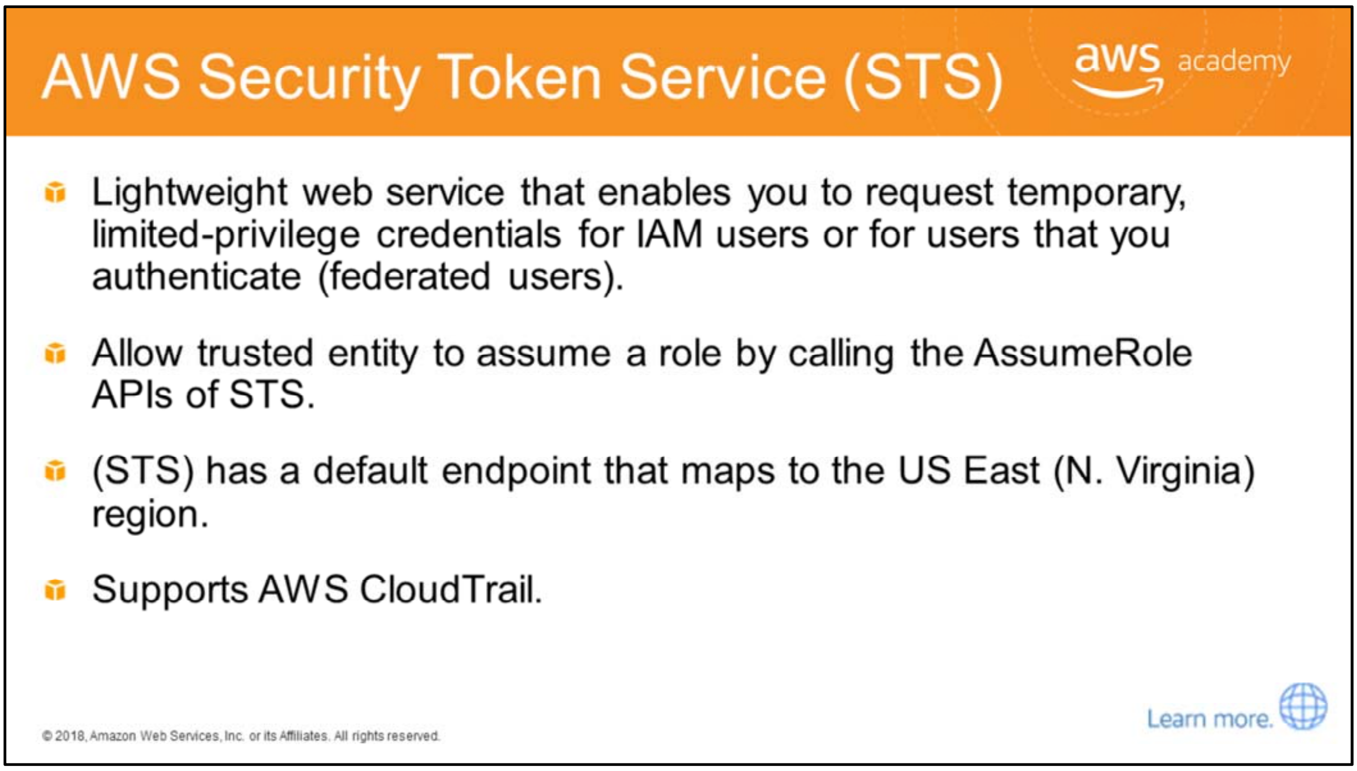
AWS Directory Service: Options

- Run Microsoft AD as a managed service within AWS Directory Service:
  - Powered by Windows Server 2012 R2.
  - Created as a highly available pair of domain controllers connected to your VPC.
- Use AD Connector:
  - Connect to your on-premises Active Directory via VPC VPN connection or AWS Direct Connect.
  - Users access AWS applications with existing credentials.
  - Integrate with existing RADIUS-based MFA solutions.
- Use Simple AD:
  - Launch managed stand-alone directories powered by Samba 4 Active Directory Compatible Server.
  - Supports common AD features.
  - Provides audit trails and event logs.

Let’s review a few options when you run Microsoft Active Directory as a managed service within AWS Directory Service.
It is powered by Windows Server 2012 Revision 2, and is created as a highly available pair of domain controllers that are connected to your VPC.

If you want to use AD Connector, connect to your on-premises Active Directory installation via a VPC VPN connection or through AWS Direct Connect.
Users can access AWS applications with existing credentials, and you can integrate with existing RADIUS-based multi-actor authentication solutions.

Simple AD provides launch-managed, stand-alone directories that are powered by Samba 4 Active Directory Compatible Server.
Simple AD supports common Active Directory features. It provides audit trails and event logs so that you can track how your directory is used, including which users logged in, when these users logged in, and any accounts that have been locked because of failed login attempts. You can use standard Microsoft Windows event log tools to review directory audit and event logs. By default, Simple AD also provides daily automated snapshots to enable point-in-time recovery.



AWS Security Token Service (STS)

- Lightweight web service that enables you to request temporary, limited-privilege credentials for IAM users or for users that you authenticate (federated users).
- Allow trusted entity to assume a role by calling the AssumeRole APIs of STS.
- (STS) has a default endpoint that maps to the US East (N. Virginia) region.
- Supports AWS CloudTrail.

Learn more.

AWS Security Token Service—or AWS STS—is a lightweight web service that enables you to request temporary, limited-privilege credentials for IAM users, or for users that you authenticate, such as federated users.

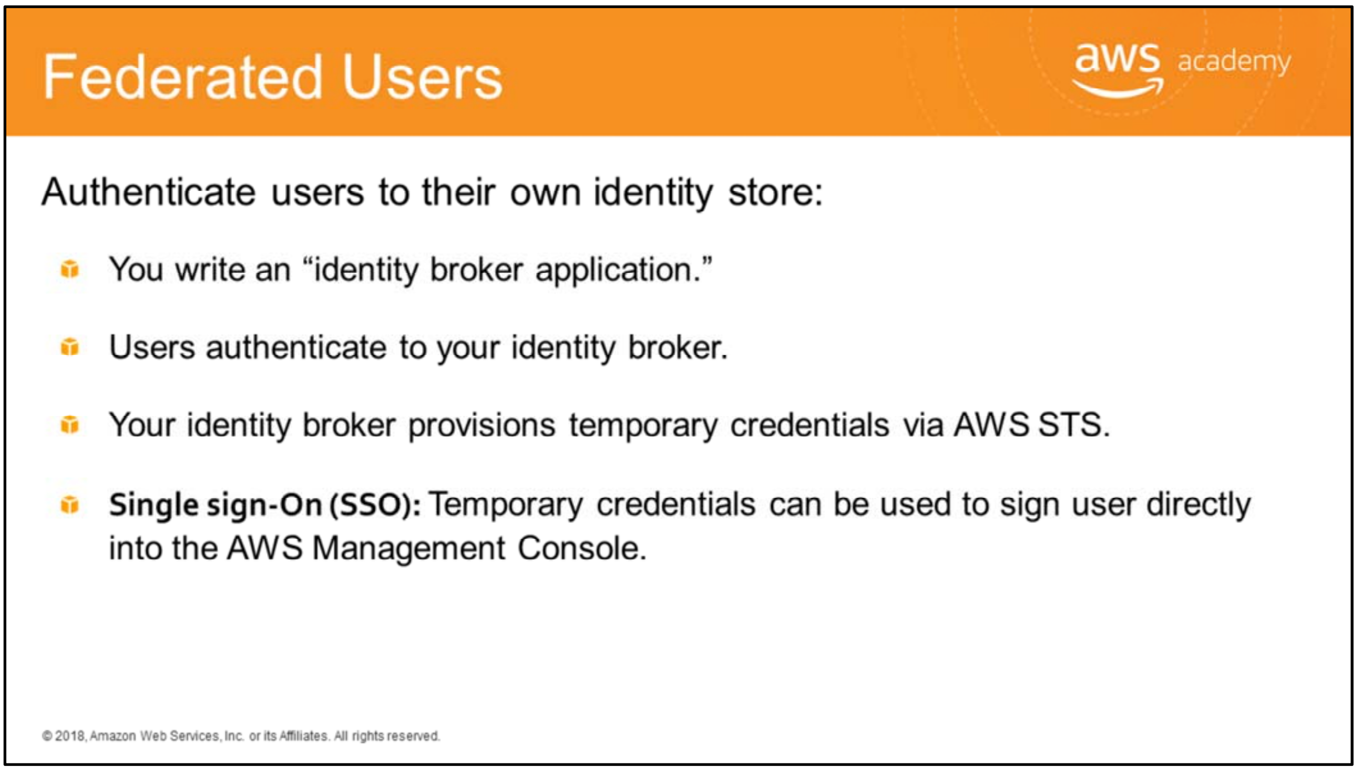
It allows a trusted entity to assume a role by calling the AssumeRole API operation of the security token service. The AssumeRole operation returns a set of temporary security credentials that consist of an access key ID, a secret access key, and a security token. You can use these temporary credentials to access AWS resources that you might not normally have access to. Typically, you use the AssumeRole operation for cross-account access or federation.

Keep in mind that AWS STS has a default endpoint of https://sts.amazonaws.com that maps to the US East Region, which is in Northern Virginia. Additional Regions are available and activated by default.

If you are trying to authenticate users in Australia, this will introduce additional latency to the authentication process.

AWS STS supports AWS CloudTrail, which is a service that records AWS calls for your AWS account and delivers log files to an Amazon S3 bucket. By using information collected by AWS CloudTrail, you can determine which requests were successfully made to AWS STS, who made the request, when it was made, and so on.

Select the link to learn more about the AWS Token Security Service.
https://docs.aws.amazon.com/STS/latest/APIReference/Welcome.html



Federated Users

Authenticate users to their own identity store:

- You write an "identity broker application."
- Users authenticate to your identity broker.
- Your identity broker provisions temporary credentials via AWS STS.
- **Single sign-On (SSO):** Temporary credentials can be used to sign user directly into the AWS Management Console.

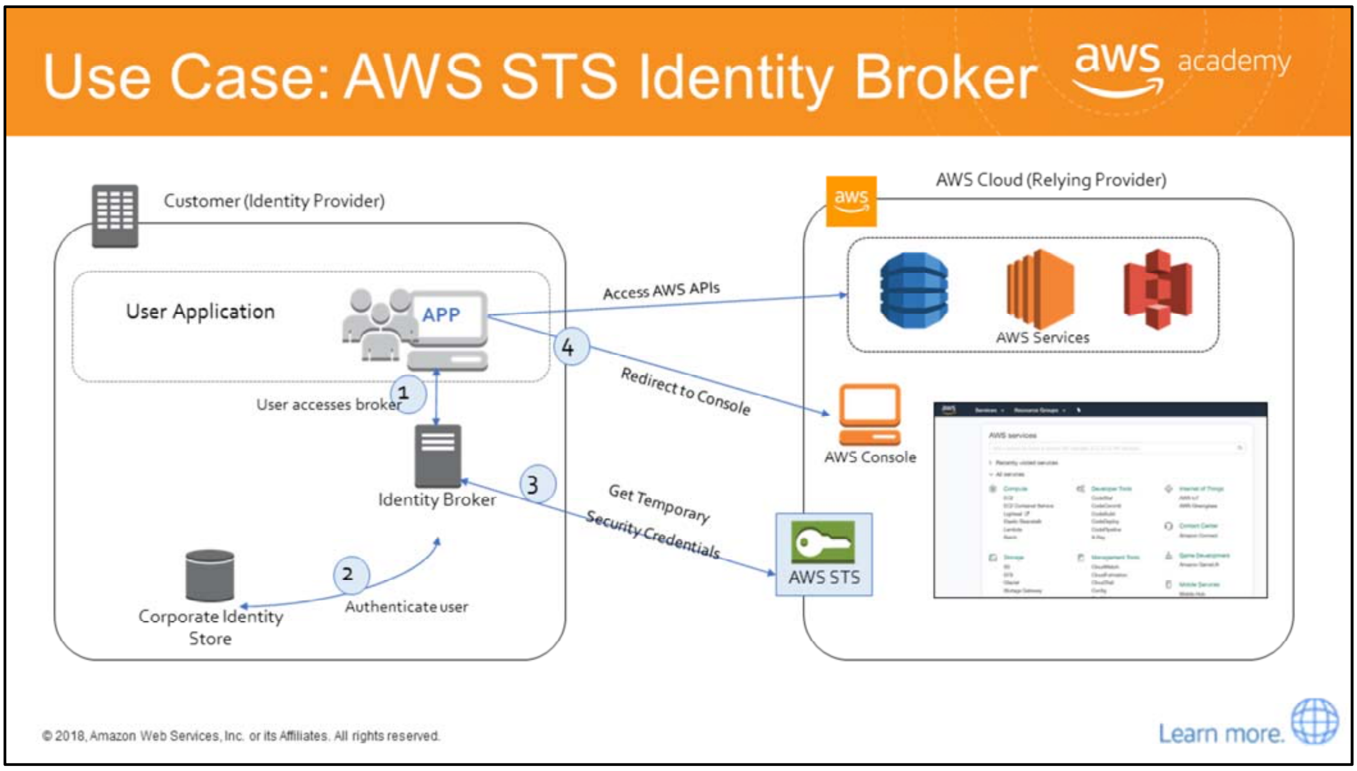
Federated users authenticate users to their own identity store.

First, you write an identity broker application.

Your users will then authenticate to your identity broker.

Next, your identity broker provisions temporary credentials via AWS STS.

This process allows you to use single sign-on, or SSO. Temporary credentials can be used to sign users directly into the AWS Management Console.

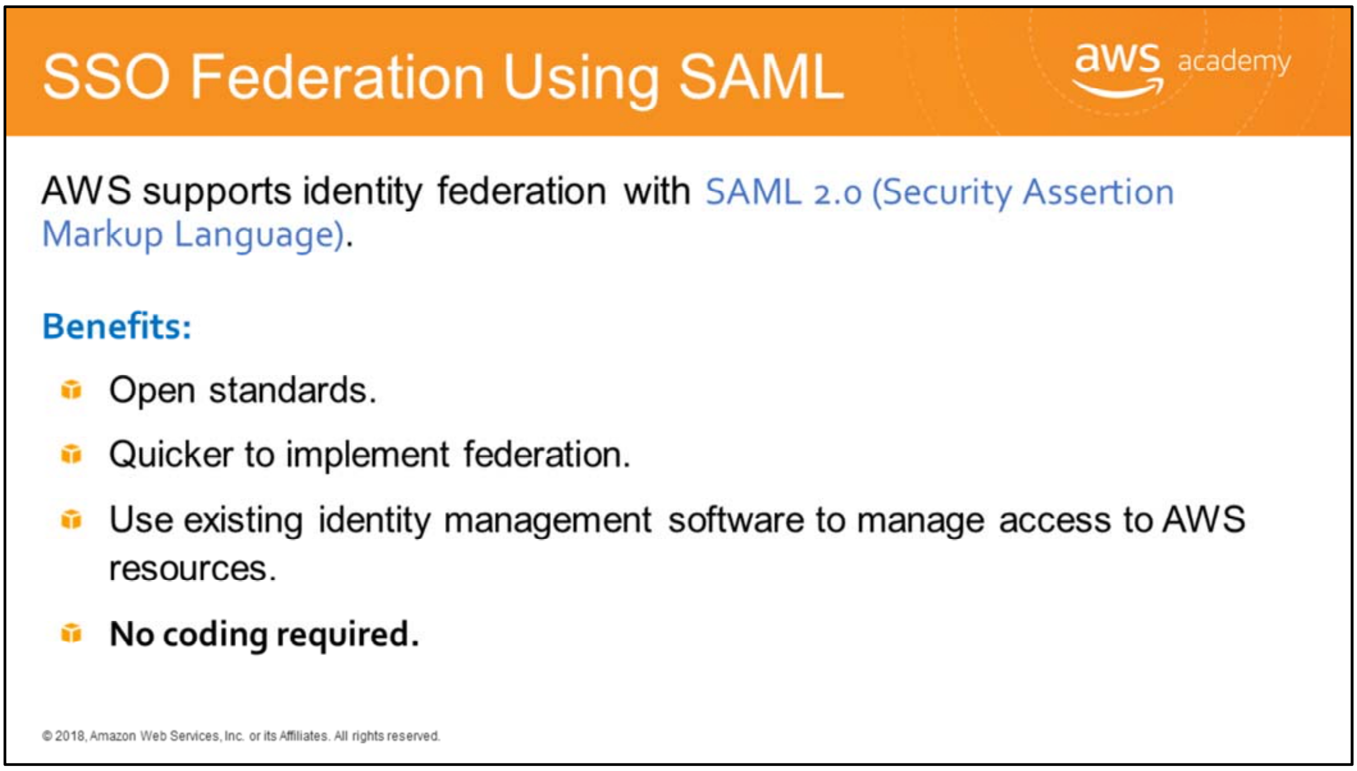


Use Case: AWS STS Identity Broker

Let's review a use case.

In step one, users try to access AWS services from their local machine. They first go to the identity broker. From the identity broker, they are authenticated within their corporate identity store. After authentication, the identity broker reaches back to AWS STS to gain a token. After the identity broker has a token, the end user is redirected to the AWS console or to one of the APIs.

Select the link to learn more about the identity federation sample application for Active Directory use case.
http://aws.amazon.com/code/1288653099190193



SSO Federation Using SAML

aws academy

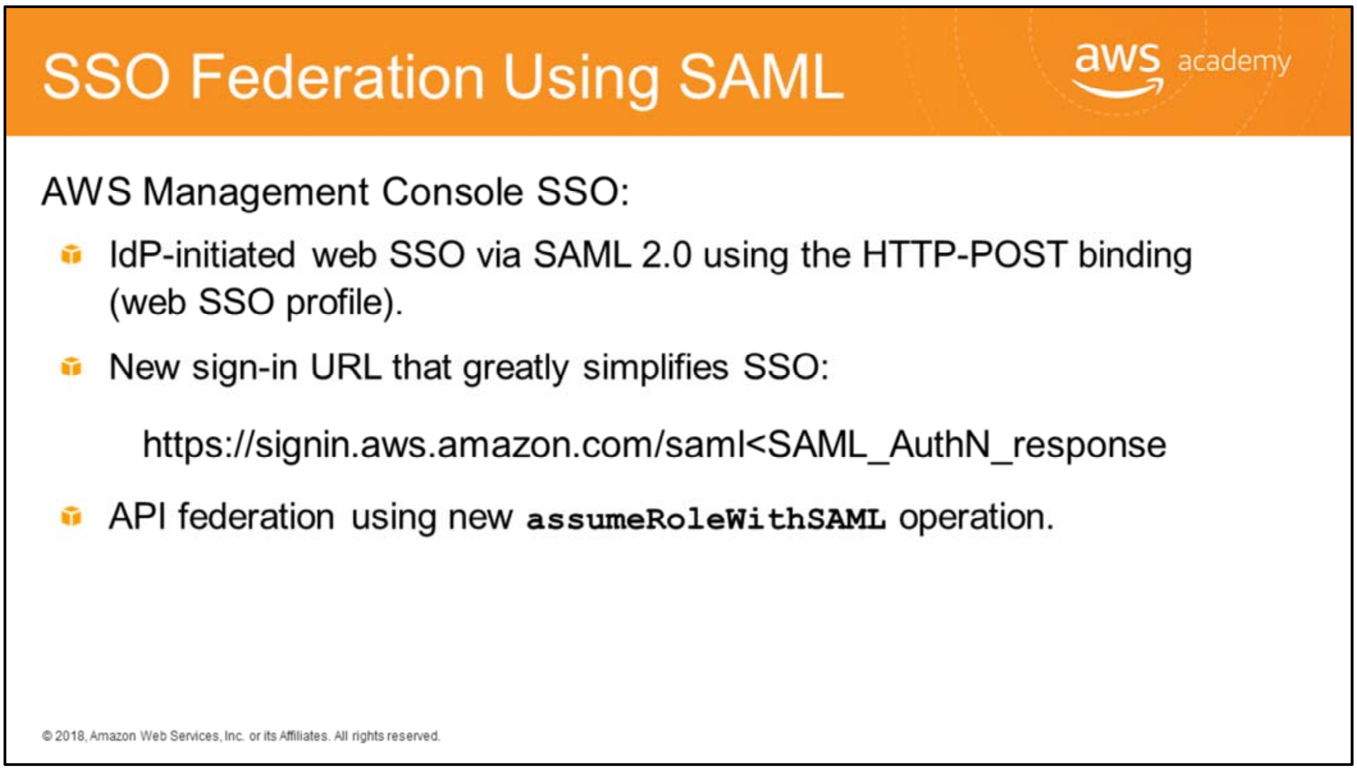
AWS supports identity federation with SAML 2.0 (Security Assertion Markup Language).

Benefits:

- Open standards.
- Quicker to implement federation.
- Use existing identity management software to manage access to AWS resources.
- **No coding required.**

AWS supports identity federation, which makes it easier to manage users by maintaining their identities in a single place. Identity federation includes support for the Security Assertion Markup Language—or SAML—2.0, which is an open standard that is used by many identity providers. You can use this feature to implement federation more quickly because it enables federated SSO. With federated SSO, users can sign in to the AWS Management Console or make programmatic calls to AWS APIs.

Identity federation uses existing identity management software to manage access to AWS resources. There are many use cases that illustrate how identity federation makes user administration easier. For instance, if a user leaves your company, you can simply delete the user's corporate identity, which then also revokes access to AWS. End users also benefit because they only need to remember one user name and password. You can use SAML to make configuring federation with AWS straightforward, because system administrators can set it up using identity provider software instead of writing code.



SSO Federation Using SAML

AWS Management Console SSO:

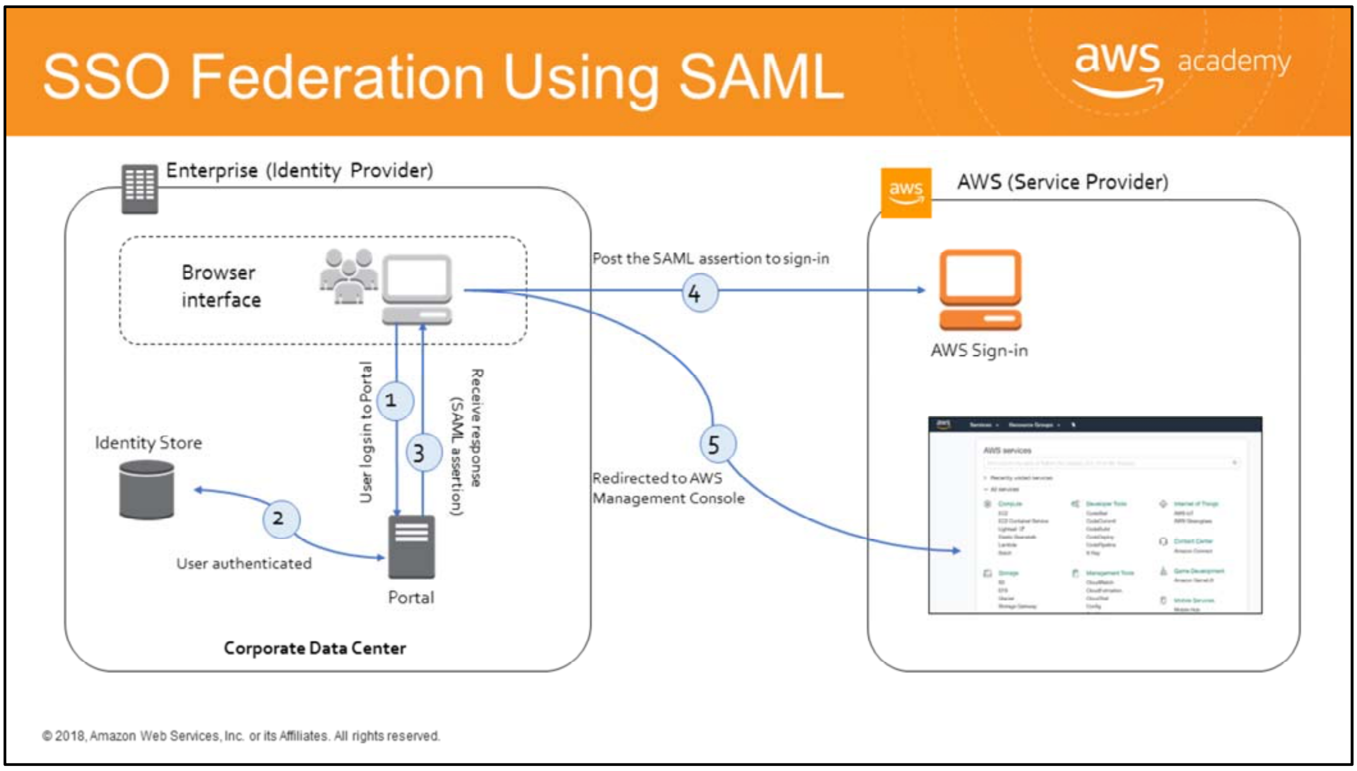
- IdP-initiated web SSO via SAML 2.0 using the HTTP-POST binding (web SSO profile).
- New sign-in URL that greatly simplifies SSO:

  https://signin.aws.amazon.com/saml<SAML_AuthN_response

- API federation using new `assumeRoleWithSAML` operation.

In order to use single sign-on for the AWS Management Console, your identity provider must initiate a web single sign-on session via SAML 2.0 by using the HTTP-POST binding.

A new sign-on URL greatly simplifies this process. In this case, the SAML authentication response and the API federation uses a new assumeRoleWithSAML operation within the code.

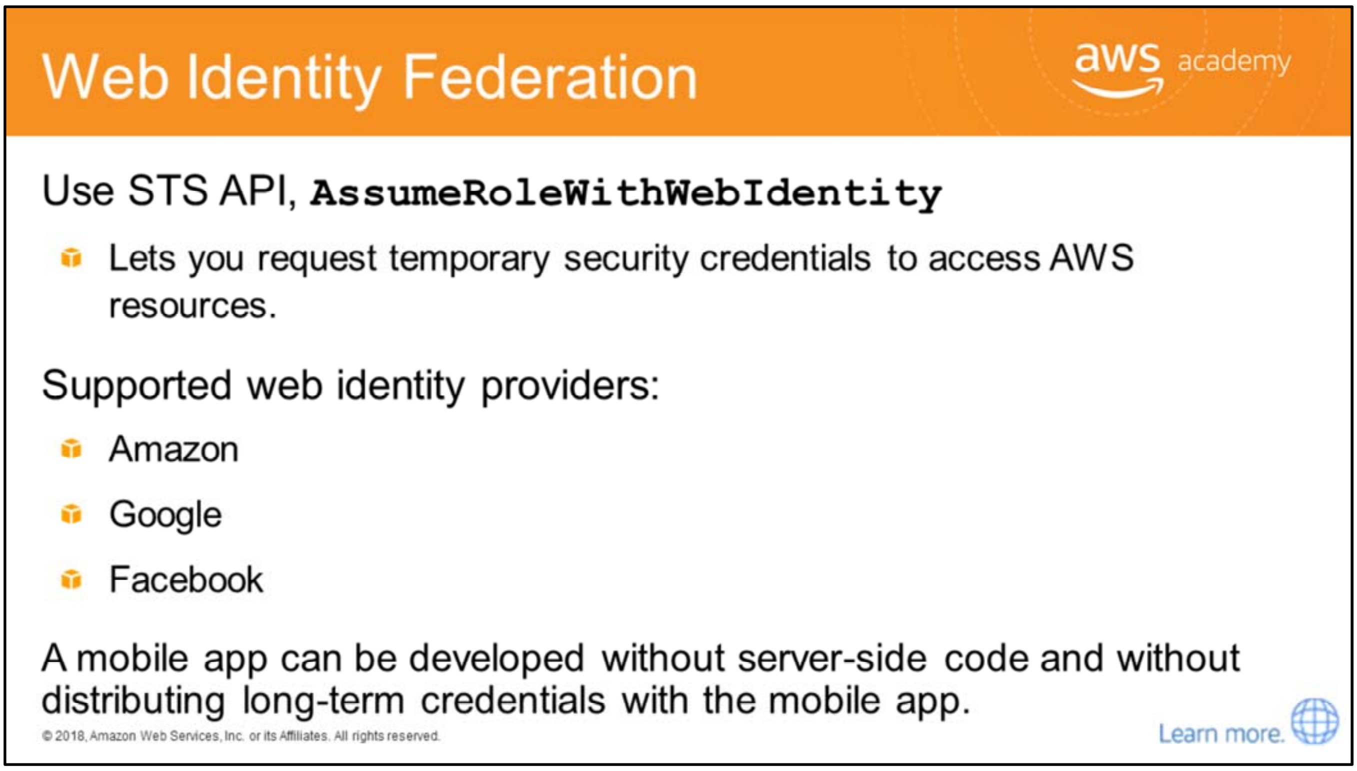


SSO Federation Using SAML

Displayed is a diagram of how SSO federation works within SAML.

- For example, a user in your organization browses to an internal portal in your network. The portal also functions as the identity provider (IdP) that handles the SAML trust between your organization and AWS.
- The identity provider authenticates the user’s identity against the ID Store.
- After the client is authenticated, the client receives a SAML assertion in the form of authentication response from the IdP.
- It can then post the SAML assertion to the new AWS sign-in portal. Behind the scenes, sign-in uses the AssumeRoleWithSAML API operation to request temporary security credentials and construct a sign-in URL.
- Finally, the user’s browser receives the sign-in URL and is automatically redirected to the AWS Management Console.

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.



Web Identity Federation

aws academy

Use STS API, **`AssumeRoleWithWebIdentity`**

- Lets you request temporary security credentials to access AWS resources.

Supported web identity providers:

- Amazon
- Google
- Facebook

A mobile app can be developed without server-side code and without distributing long-term credentials with the mobile app.

Learn more.

You can also use web identity federation when you use secure token service API with the AssumeRoleWithWebIdentity operation. This API operation lets you request temporary security credentials so that you can access AWS resources.

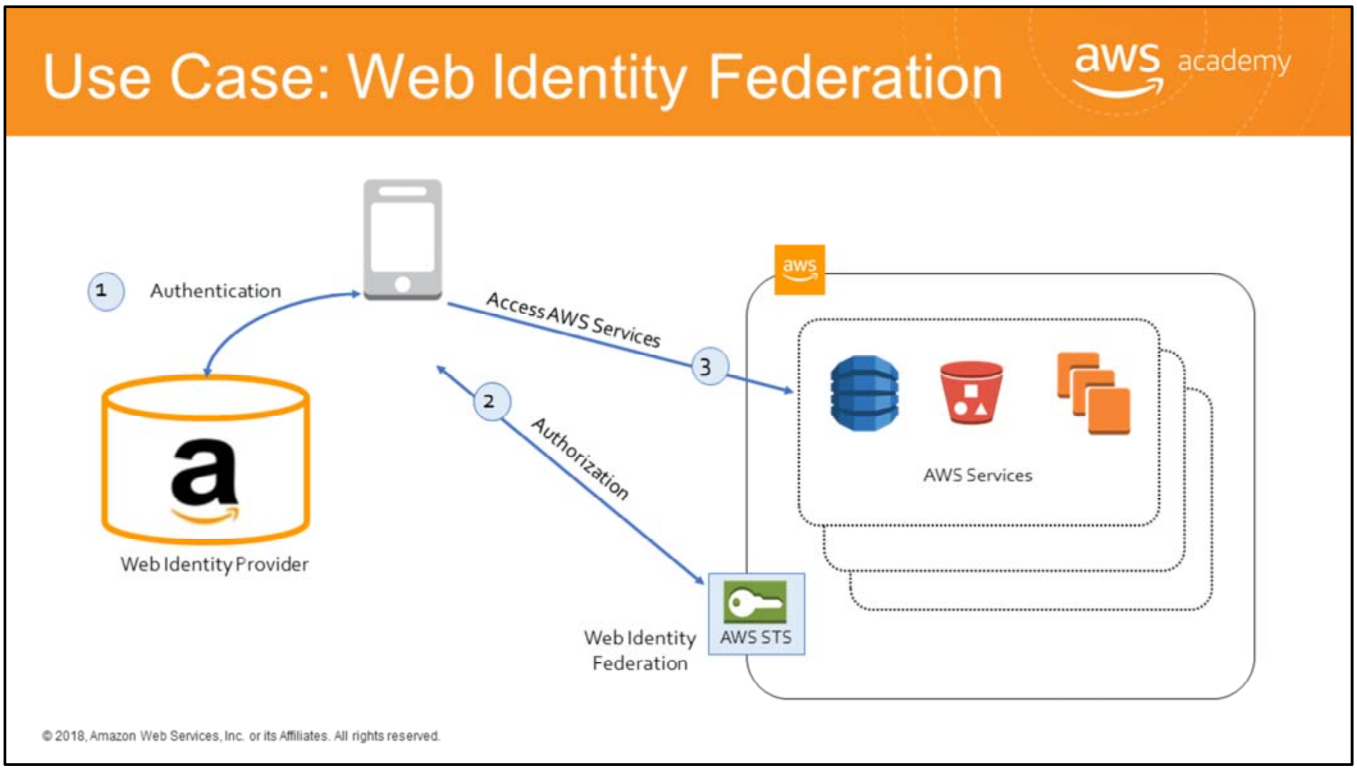
Supported web identity providers include Amazon, Google and Facebook.
Web identity federation is great for mobile apps where you don't want to use server-side code, and you don't want to distribute long term credentials.

To configure your application with federated credentials using trusted identity providers, you'd:

- Register an application with the identity provider.
- Create an IAM role for the identity provider.
- Set up permissions for the IAM role.
- Use the identity provider's SDK to get an access to a token after logging in.
- And use the AWS SDK for JavaScript to get temporary credentials to your application.

For more information on Web Identity Federations, select the link.
https://aws.amazon.com/identity/federation/

Use Case: Web Identity Federation

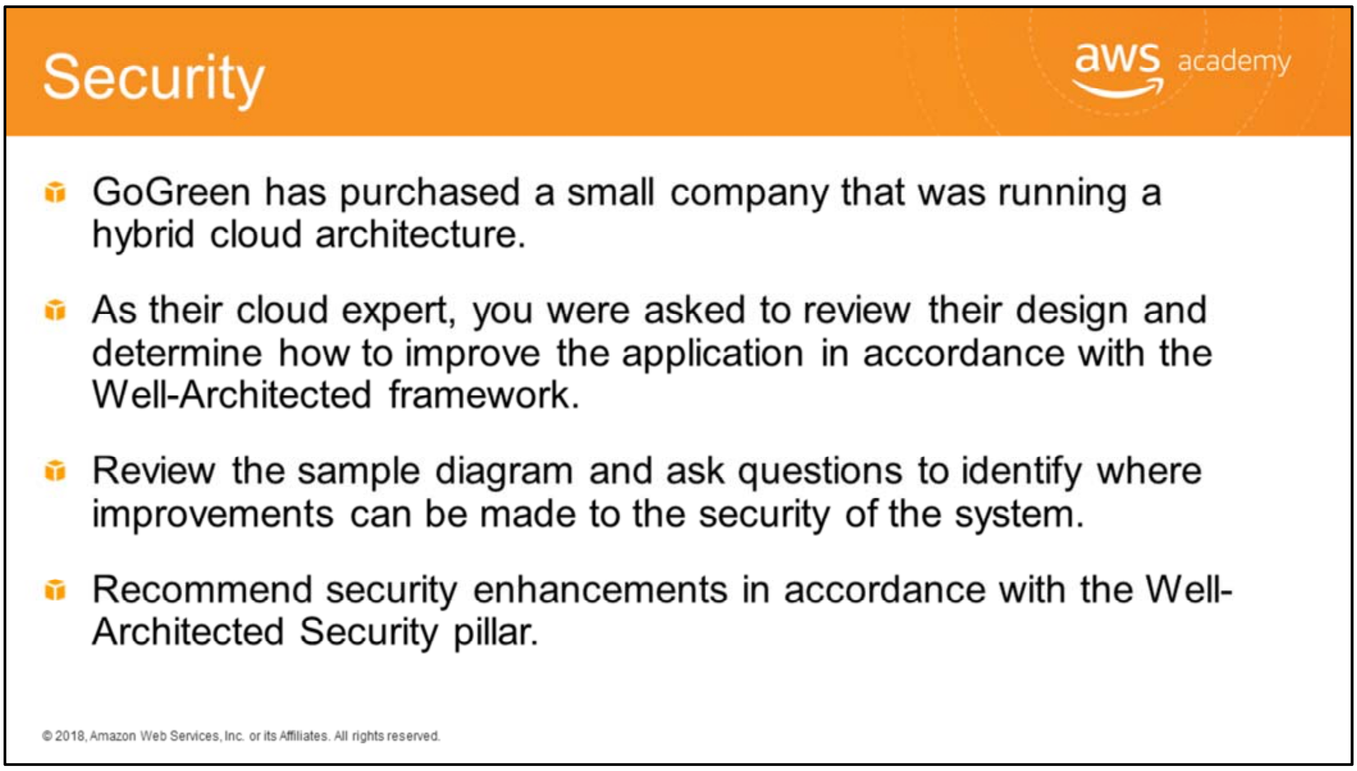


Displayed is a use case for web identity federation.
First, the user logs in from a mobile phone. In this case, the login request then goes to Amazon to authenticate the user ID. After the user is authenticated, it receives the authorization from the Web Identity Federation to return a token. With that token, the user can now to log in to the AWS services they can access. By default, the credentials can be used for one hour.

If the role’s access policy uses variables that reference the application ID and the user ID, the temporary security credentials are scoped to that end user, and will prevent him or her from accessing objects owned by other users.

Exercise: Improve This Architecture

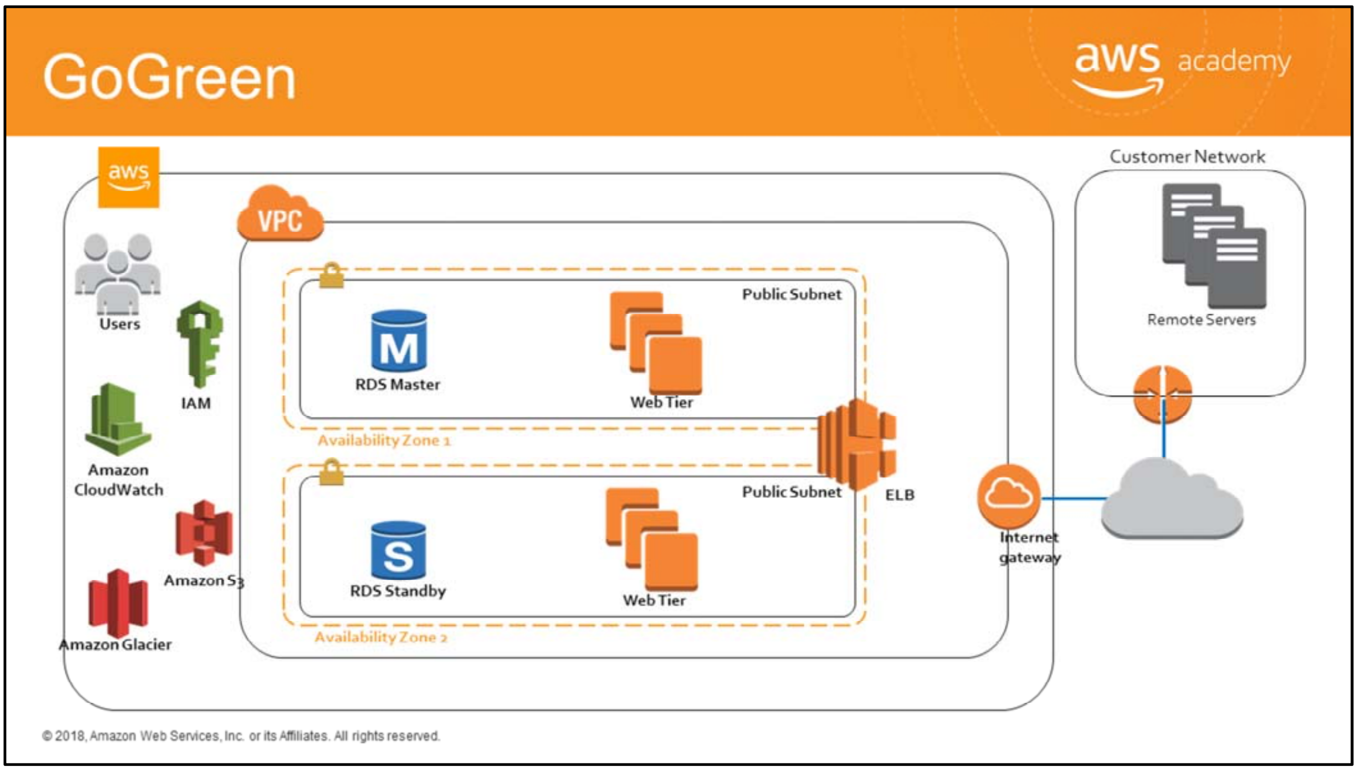
Now, you will complete an exercise on how to improve an architecture.



Security

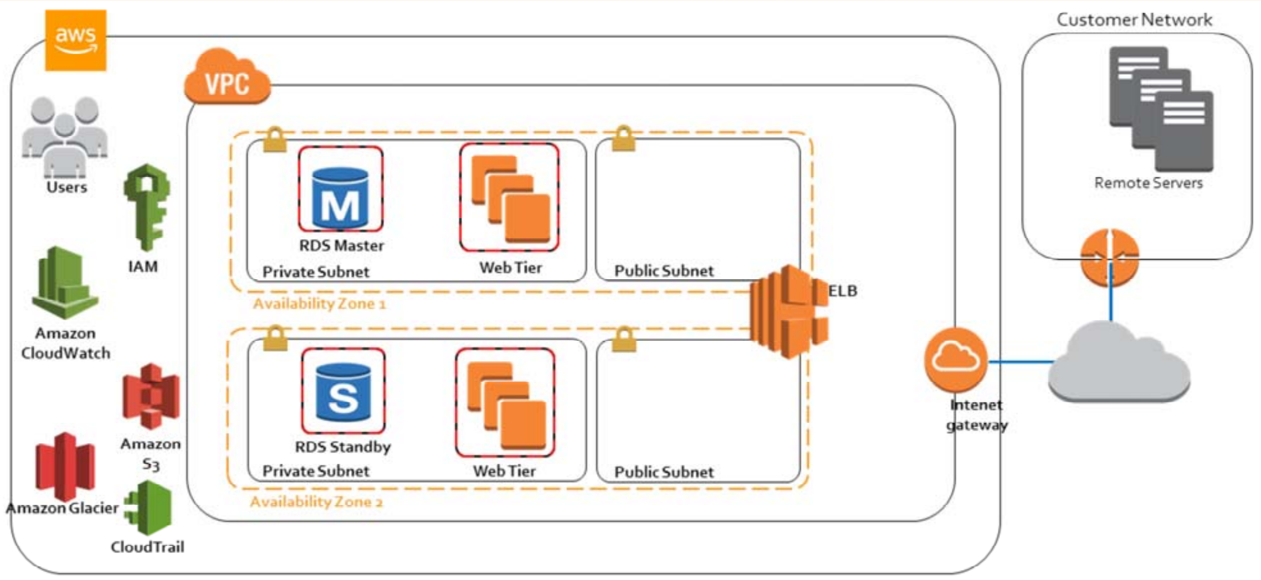
- GoGreen has purchased a small company that was running a hybrid cloud architecture.
- As their cloud expert, you were asked to review their design and determine how to improve the application in accordance with the Well-Architected framework.
- Review the sample diagram and ask questions to identify where improvements can be made to the security of the system.
- Recommend security enhancements in accordance with the Well-Architected Security pillar.

In this scenario, GoGreen purchased a small company that was running a hybrid cloud architecture.
As their cloud expert, you were asked to review their design and determine how to improve the application in accordance with the Well-Architected Framework.
Next, you will review a sample diagram and ask questions to identify where improvements can be made to the security of the system.
You should consider the recommended security enhancements that you would make in accordance with the Security pillar of the Well-Architected Framework.

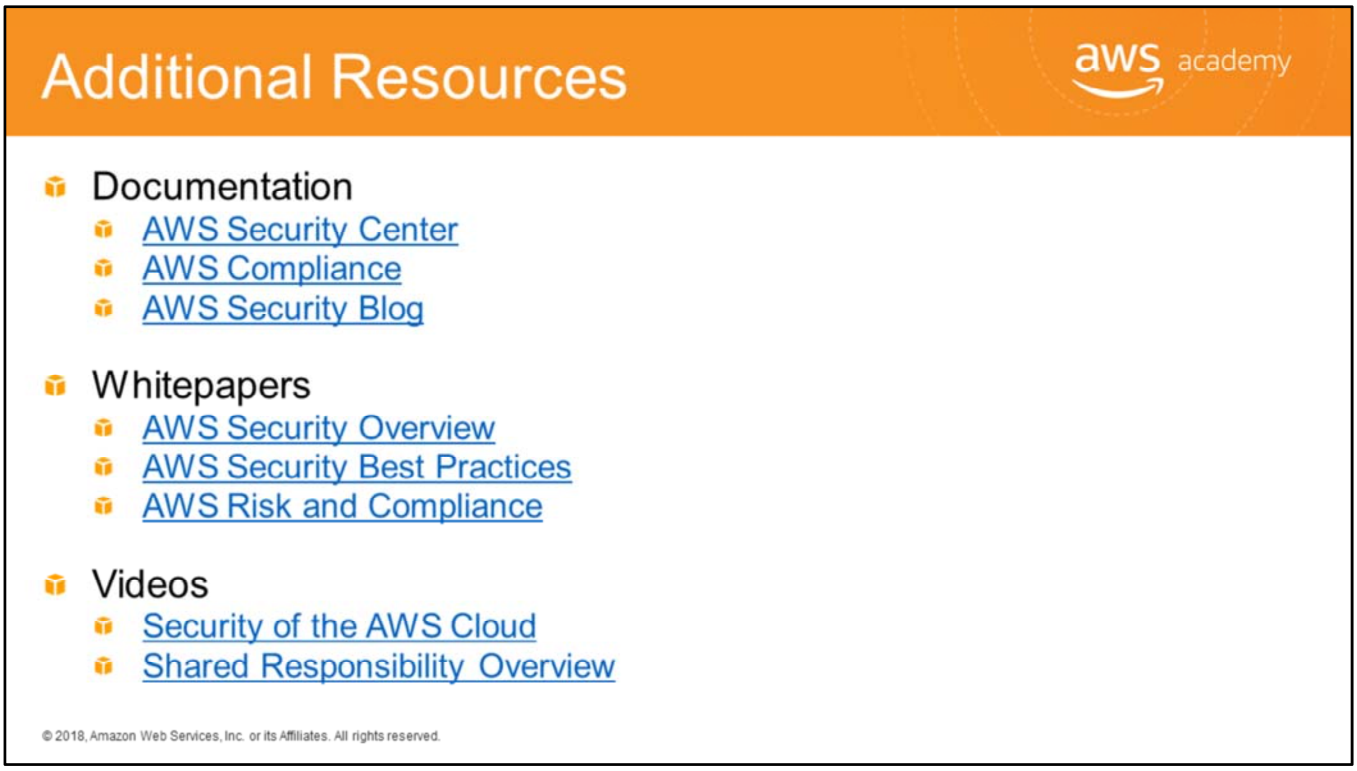


Take some time to review this sample diagram and ask questions that will help you identify where improvements can be made to the security of the system.
You should consider the recommended security enhancements that you would make in accordance with the Security pillar of the Well-Architected Framework.

GoGreen



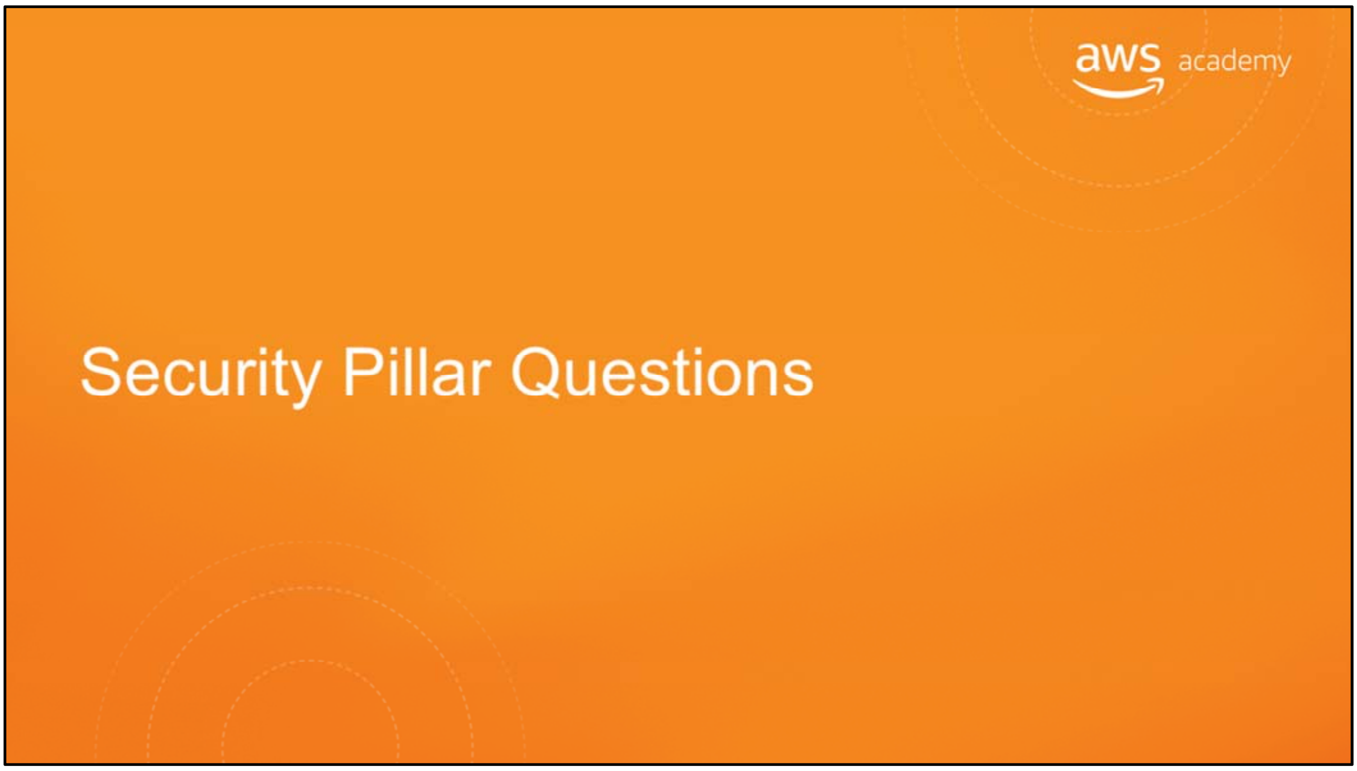
A few items to consider regarding security include:
- Moving to a private subnet.
- Implementing AWS Direct Connect instead of using internet routing.
- Verify the principles of least privilege on users.
- Ensure separation of duties through IAM.
- Ensure that Amazon CloudWatch has monitors for utilization, abnormal traffic, and AWS CloudTrail Logs.
- Enable AWS CloudTrail.
- Verify access control list policies and bucket policies for Amazon S3.
- Verify Amazon S3-SSE usage.
- Ensure users do not have delete privileges for Amazon Glacier.
- Consider using security groups for the web and Amazon RDS.
- And ensure that SSH requests come from only the administrator IP address.



Additional Resources

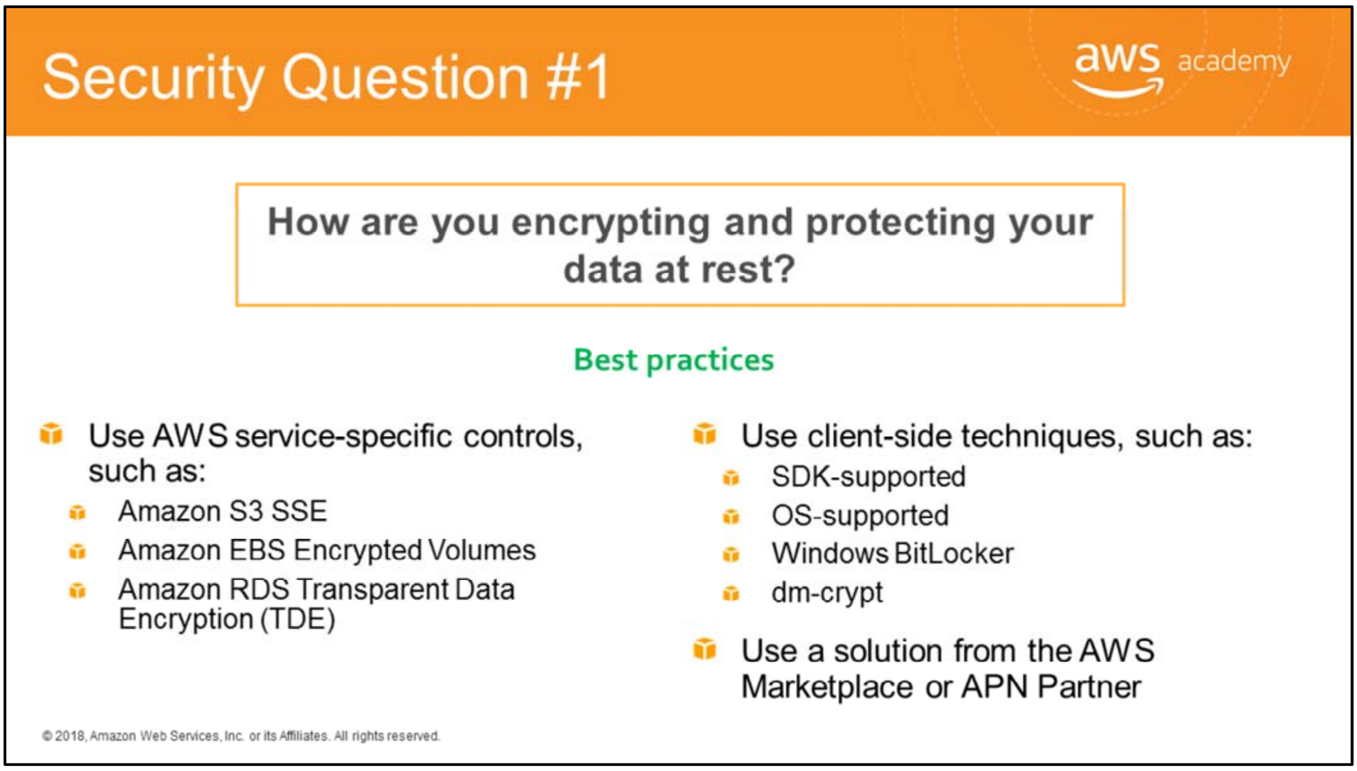
- Documentation
  - AWS Security Center
  - AWS Compliance
  - AWS Security Blog
- Whitepapers
  - AWS Security Overview
  - AWS Security Best Practices
  - AWS Risk and Compliance
- Videos
  - Security of the AWS Cloud
  - Shared Responsibility Overview

Displayed is a list of additional resources you can refer to. Select each link to learn more.



Security Pillar Questions

Before we finalize this module, let’s review some questions and best practices regarding the Security pillar.



Security Question #1

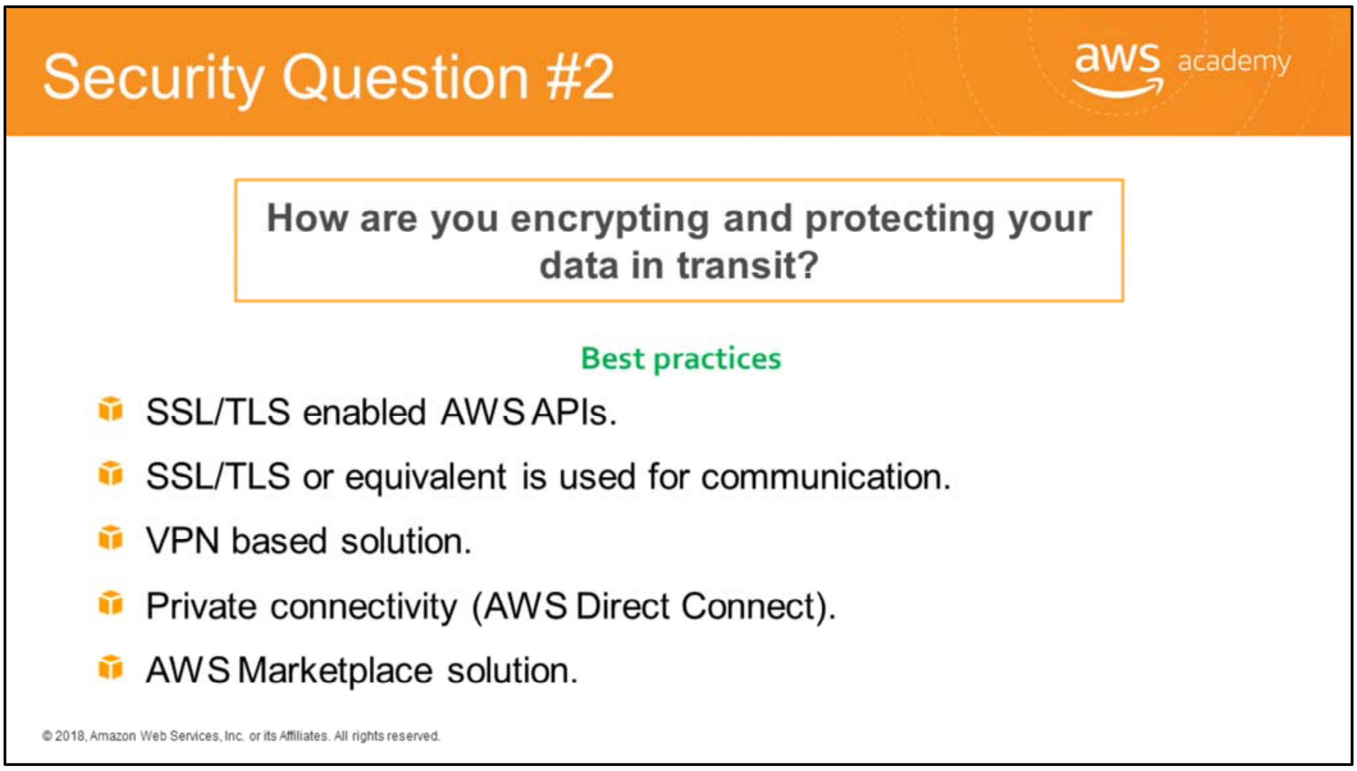
How are you encrypting and protecting your data at rest?

Best practices

- Use AWS service-specific controls, such as:
  - Amazon S3 SSE
  - Amazon EBS Encrypted Volumes
  - Amazon RDS Transparent Data Encryption (TDE)
- Use client-side techniques, such as:
  - SDK-supported
  - OS-supported
  - Windows BitLocker
  - dm-crypt
- Use a solution from the AWS Marketplace or APN Partner

How are you encrypting and protecting your data at rest?

Take a moment to review some best practices.



Security Question #2

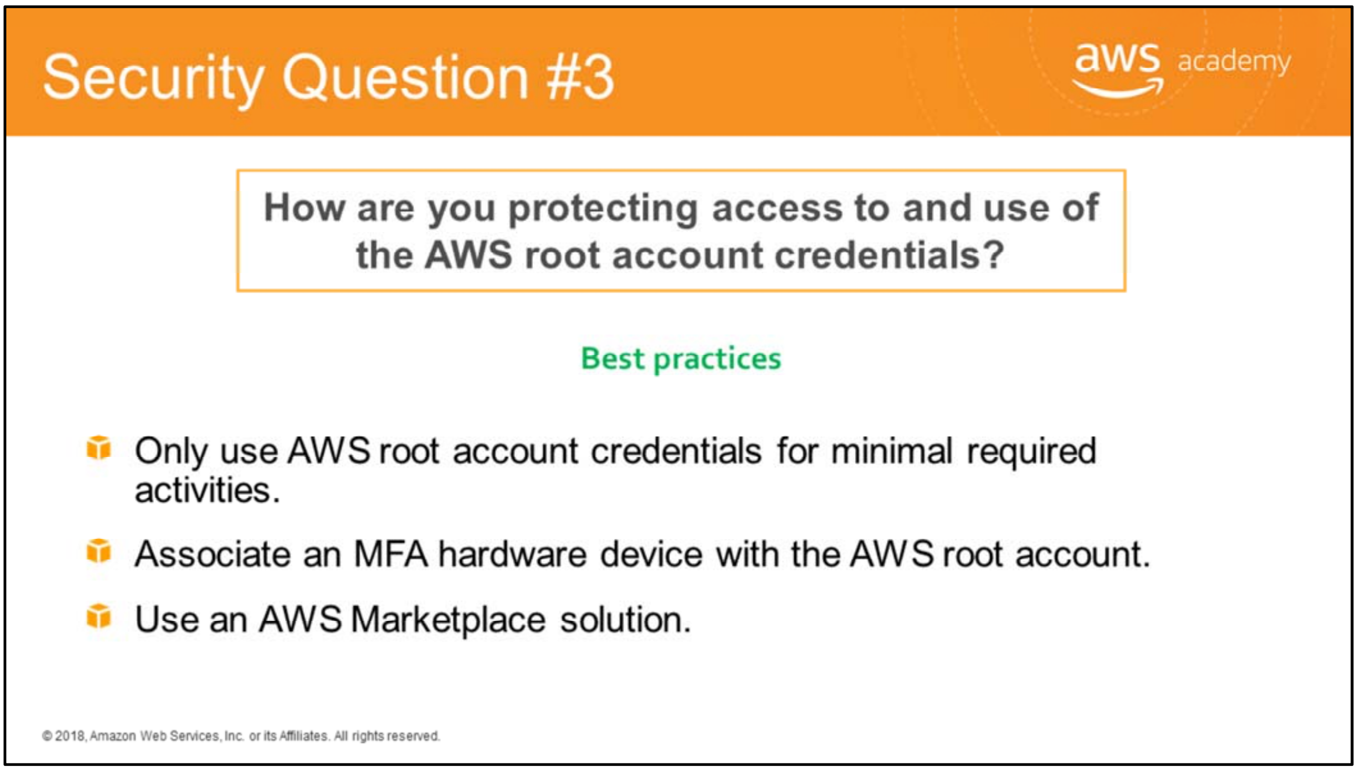
How are you encrypting and protecting your data in transit?

Best practices

- SSL/TLS enabled AWS APIs.
- SSL/TLS or equivalent is used for communication.
- VPN based solution.
- Private connectivity (AWS Direct Connect).
- AWS Marketplace solution.

How are you encrypting and protecting your data in transit?

Take a moment to review some best practices.



Security Question #3

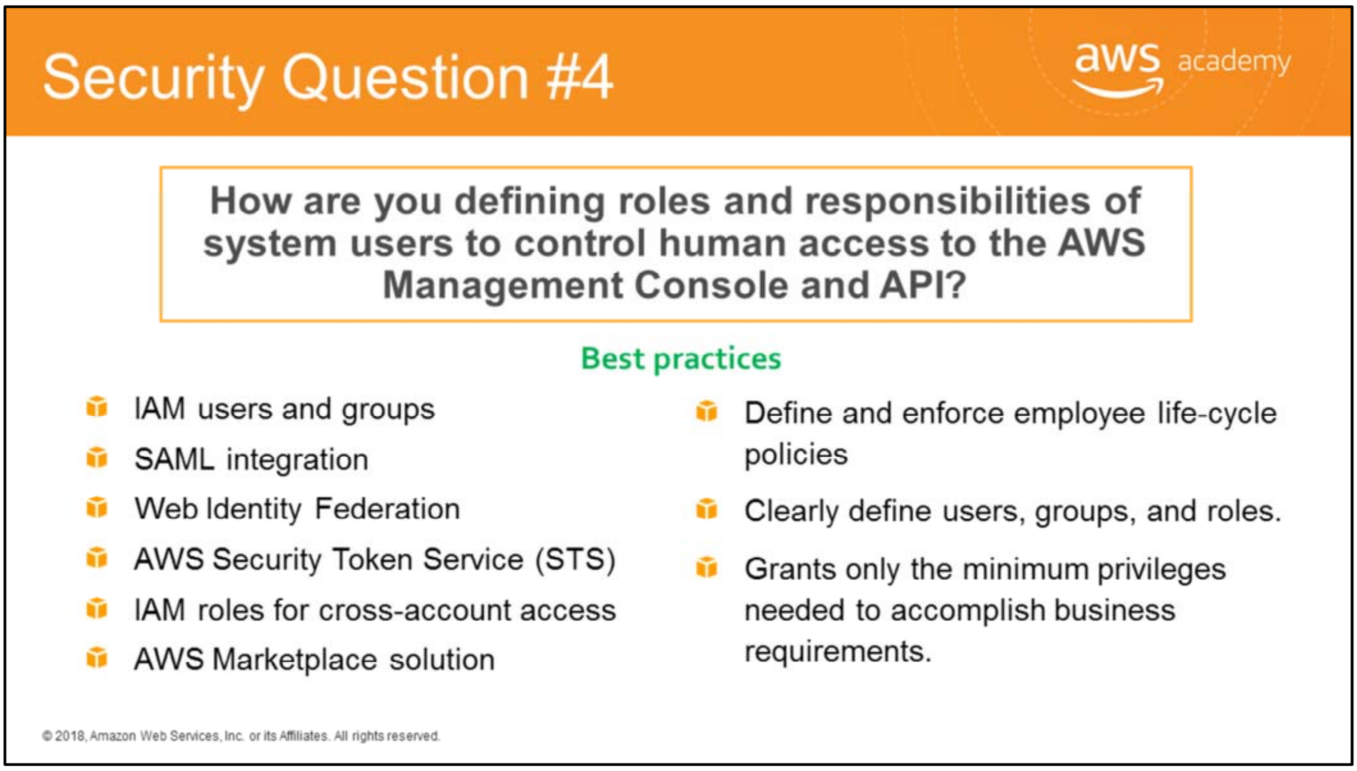
How are you protecting access to and use of the AWS root account credentials?

Best practices

- Only use AWS root account credentials for minimal required activities.
- Associate an MFA hardware device with the AWS root account.
- Use an AWS Marketplace solution.

How do you protect access to the AWS root account credentials, and how they are used?

Take a moment to review some best practices.



Security Question #4

How are you defining roles and responsibilities of system users to control human access to the AWS Management Console and API?

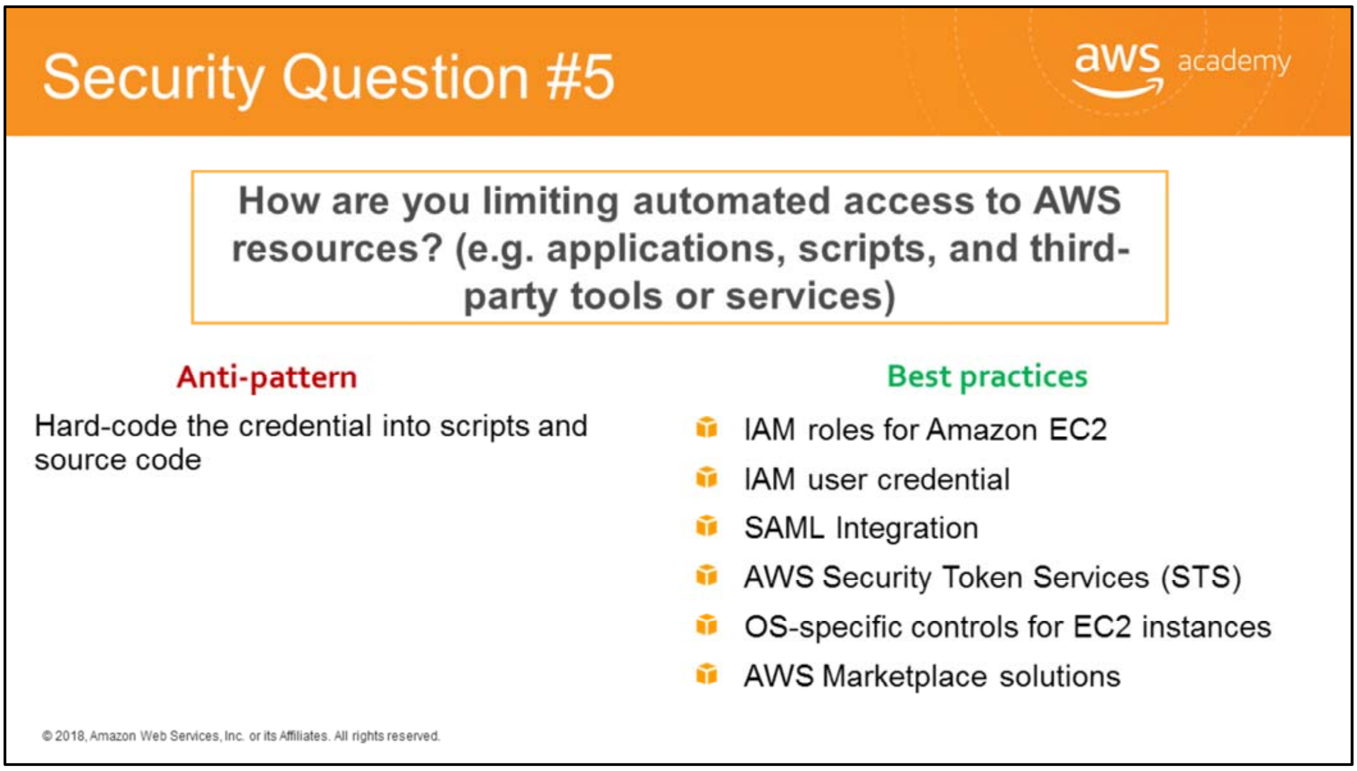
Best practices

- IAM users and groups
- SAML integration
- Web Identity Federation
- AWS Security Token Service (STS)
- IAM roles for cross-account access
- AWS Marketplace solution

- Define and enforce employee life-cycle policies
- Clearly define users, groups, and roles.
- Grants only the minimum privileges needed to accomplish business requirements.

How do you define the roles and responsibilities of system users so that you can control human access to the AWS Management Console and the API?

Take a moment to review some best practices.



Security Question #5

How are you limiting automated access to AWS resources? (e.g. applications, scripts, and third-party tools or services)

Anti-pattern

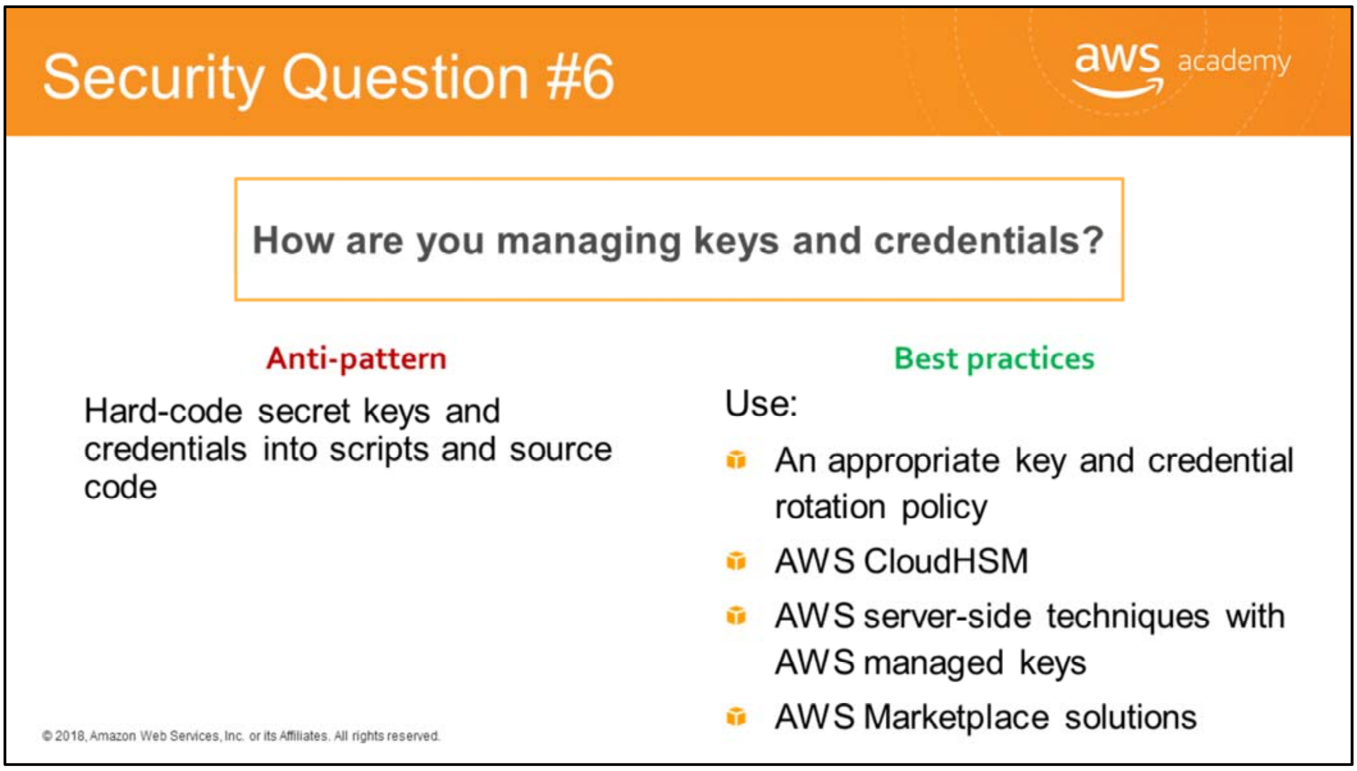
Hard-code the credential into scripts and source code

Best practices

- IAM roles for Amazon EC2
- IAM user credential
- SAML Integration
- AWS Security Token Services (STS)
- OS-specific controls for EC2 instances
- AWS Marketplace solutions

How are you limiting automated access to AWS resources, such as applications, scripts, and third-party tools or services?

Take a moment to review the anti-pattern and some best practices.



Security Question #6

How are you managing keys and credentials?

Anti-pattern

Hard-code secret keys and credentials into scripts and source code

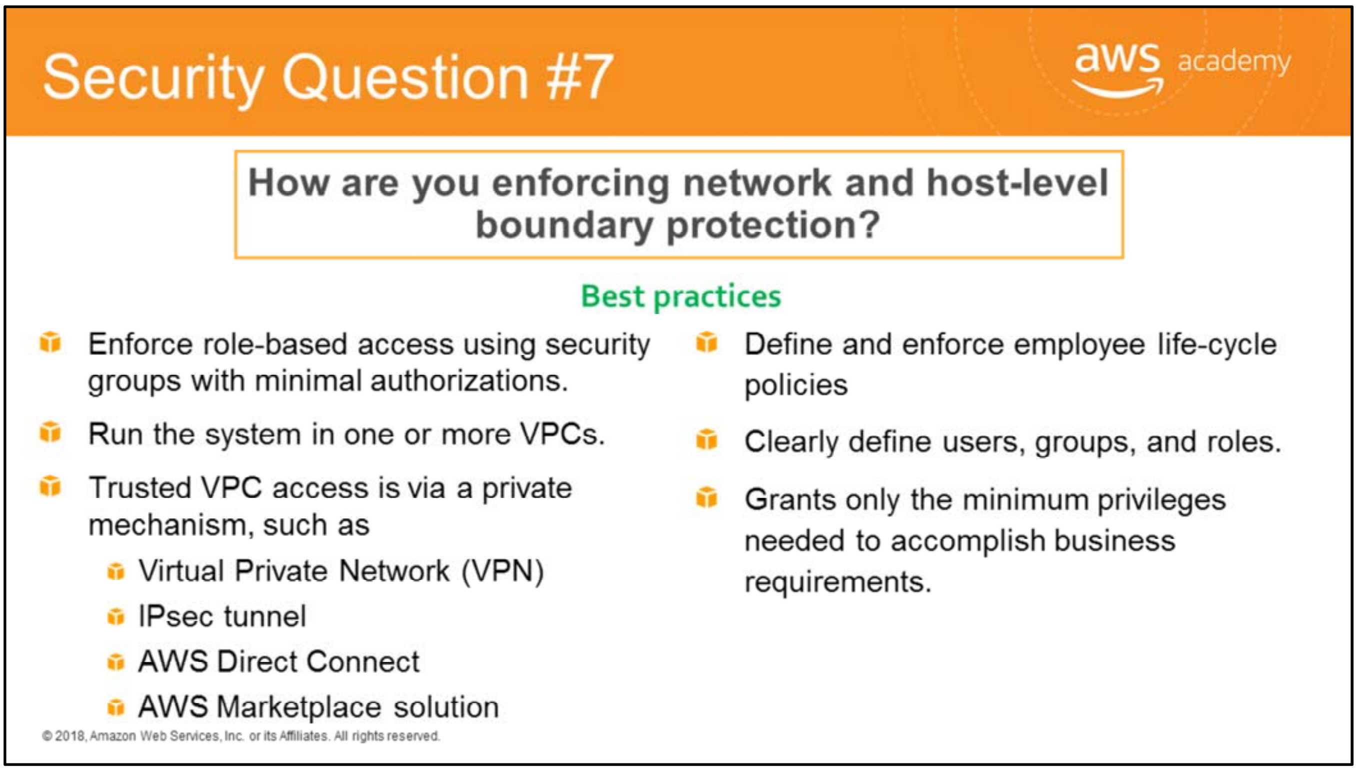
Best practices

Use:

- An appropriate key and credential rotation policy
- AWS CloudHSM
- AWS server-side techniques with AWS managed keys
- AWS Marketplace solutions

How are you managing keys and credentials?

Take a moment to review the anti-pattern and some best practices.



Security Question #7

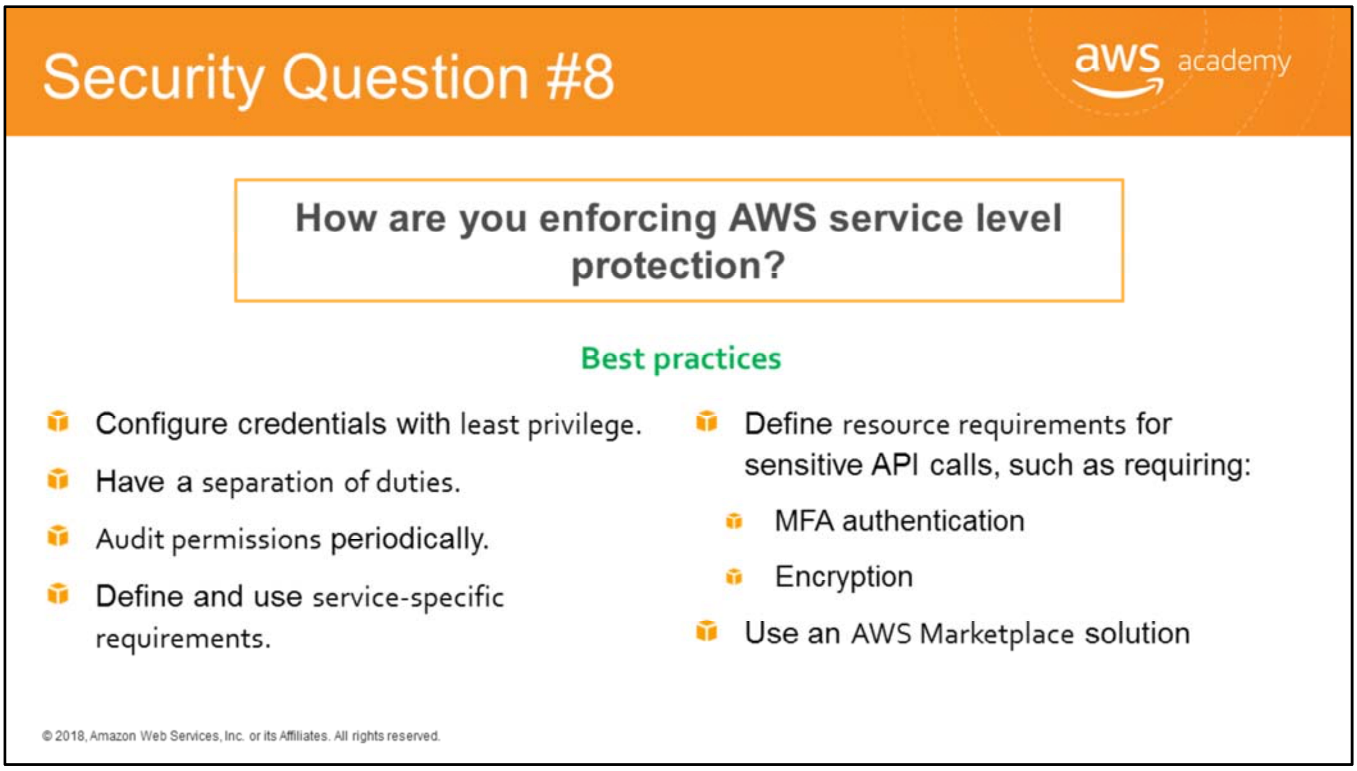
How are you enforcing network and host-level boundary protection?

Best practices

- Enforce role-based access using security groups with minimal authorizations.
- Run the system in one or more VPCs.
- Trusted VPC access is via a private mechanism, such as
  - Virtual Private Network (VPN)
  - IPsec tunnel
  - AWS Direct Connect
  - AWS Marketplace solution
- Define and enforce employee life-cycle policies
- Clearly define users, groups, and roles.
- Grants only the minimum privileges needed to accomplish business requirements.

How are you enforcing network and host-level boundary protection?

Take a moment to review some best practices.



Security Question #8

aws academy

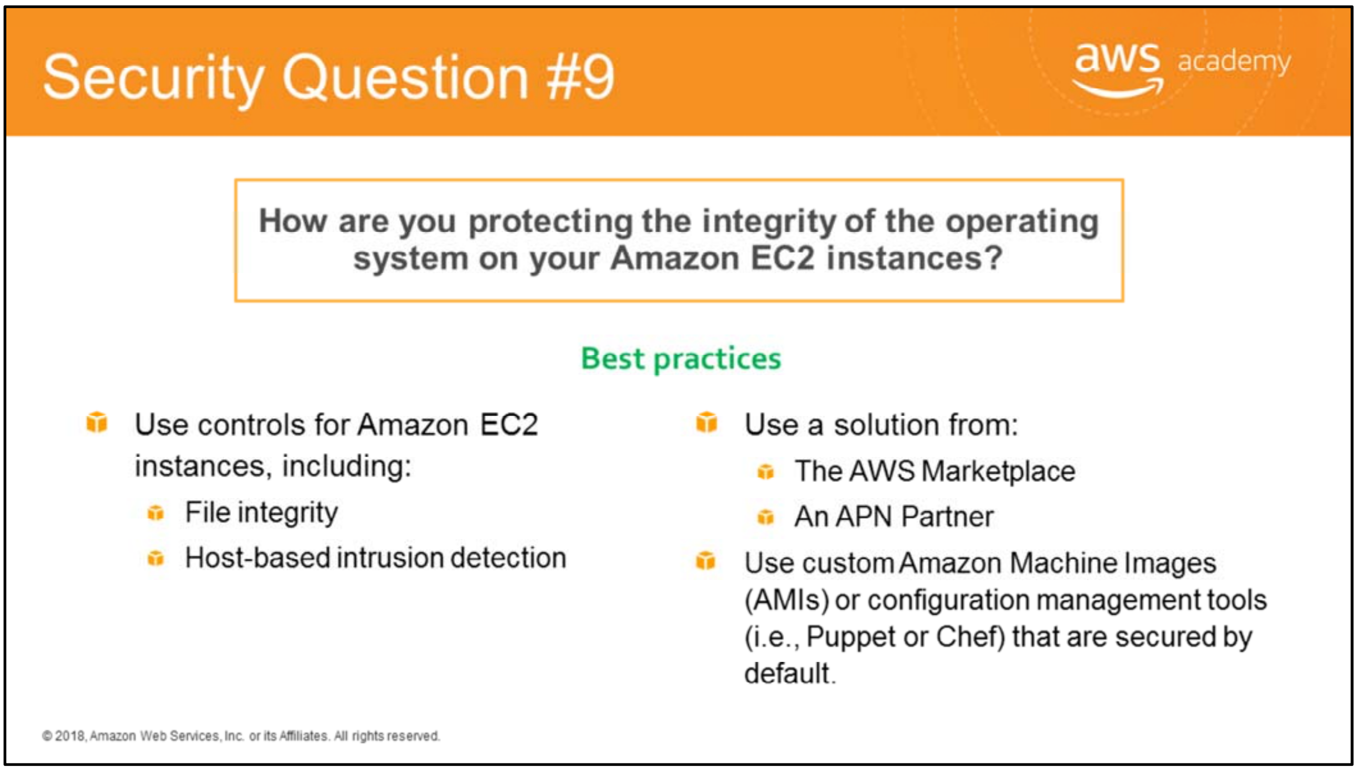
How are you enforcing AWS service level protection?

Best practices

- Configure credentials with least privilege.
- Have a separation of duties.
- Audit permissions periodically.
- Define and use service-specific requirements.
- Define resource requirements for sensitive API calls, such as requiring:
  - MFA authentication
  - Encryption
- Use an AWS Marketplace solution

How are you enforcing AWS service-level protection?

Take a moment to review some best practices.



Security Question #9

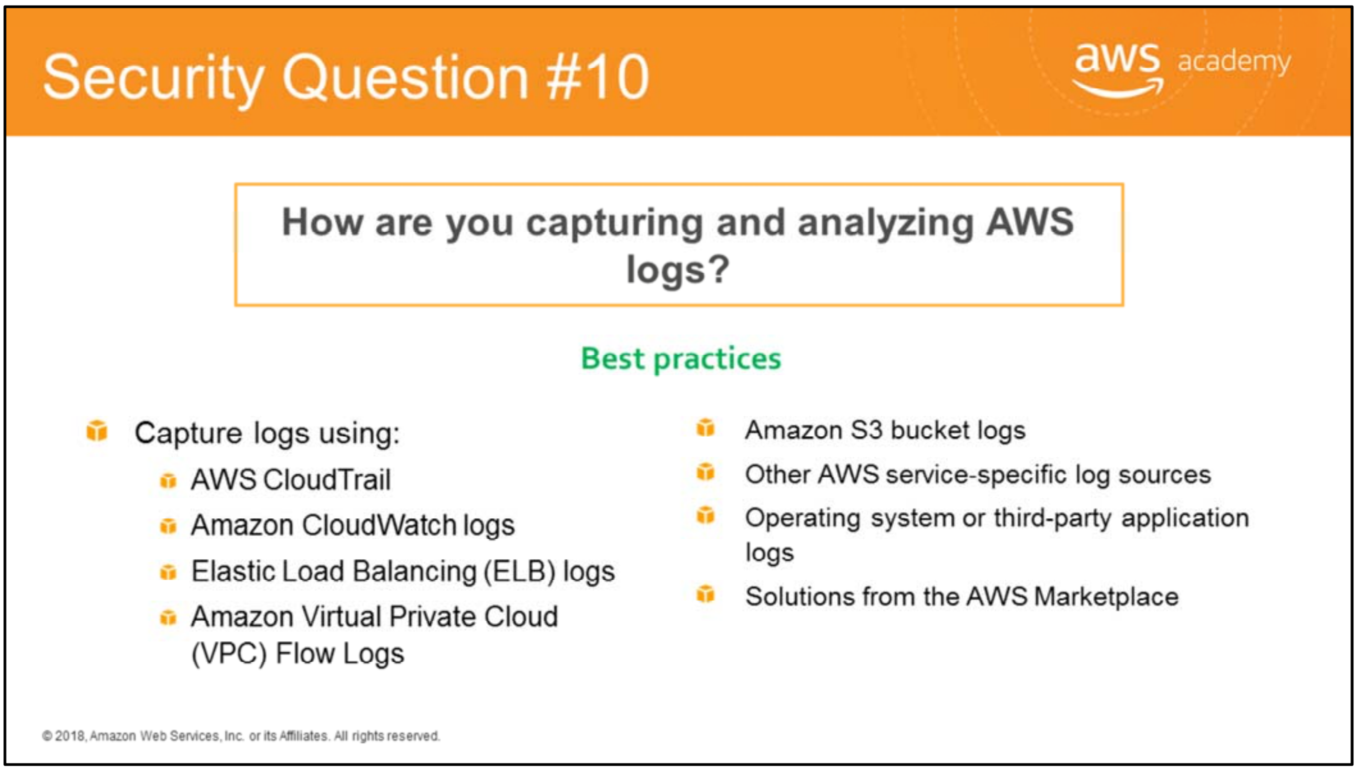
How are you protecting the integrity of the operating system on your Amazon EC2 instances?

Best practices

- Use controls for Amazon EC2 instances, including:
  - File integrity
  - Host-based intrusion detection
- Use a solution from:
  - The AWS Marketplace
  - An APN Partner
- Use custom Amazon Machine Images (AMIs) or configuration management tools (i.e., Puppet or Chef) that are secured by default.

How are you protecting the integrity of the operating system on your Amazon EC2 instances?

Take a moment to review some best practices.



Security Question #10

aws academy

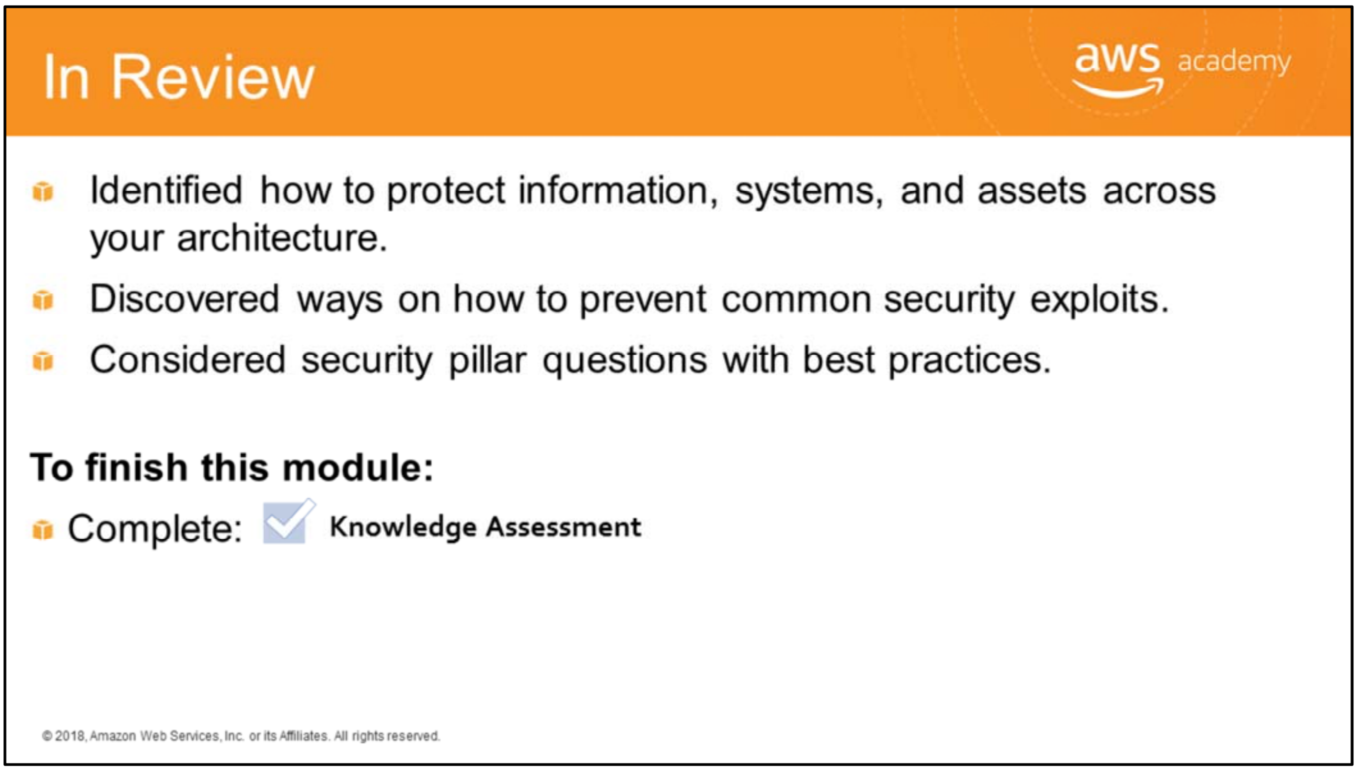
How are you capturing and analyzing AWS logs?

Best practices

- Capture logs using:
  - AWS CloudTrail
  - Amazon CloudWatch logs
  - Elastic Load Balancing (ELB) logs
  - Amazon Virtual Private Cloud (VPC) Flow Logs
- Amazon S3 bucket logs
- Other AWS service-specific log sources
- Operating system or third-party application logs
- Solutions from the AWS Marketplace

How are you capturing and analyzing AWS logs?

Take a moment to review some best practices.


In Review

- Identified how to protect information, systems, and assets across your architecture.
- Discovered ways on how to prevent common security exploits.
- Considered security pillar questions with best practices.

To finish this module:

- Complete: ✓ **Knowledge Assessment**

In review, we:

- Identified how to protect information, systems, and assets across your architecture.
- Discovered ways on how to prevent common security exploits.
- And considered security pillar questions with best practices.

To finish this module, please complete the corresponding knowledge assessment.