**Security and compliance:**

[ACCESS CONTROL](https://aws.amazon.com/iam/)

AWS Identity and Access Management (IAM) allows you to control users' access to AWS services. Create and manage users and groups, and grant or deny access.

[SINGLE SIGN ON](https://aws.amazon.com/single-sign-on/)

AWS Single Sign On (SSO) is a cloud SSO service that makes it easy to centrally manage SSO access to multiple AWS accounts and business applications.

[ACCOUNT MANAGEMENT](https://aws.amazon.com/organizations/)

AWS Organizations offers policy-based management for multiple AWS accounts. With AWS Organizations, you can create groups of accounts, automate account creation, and apply and manage policies for those groups.

[THREAT DETECTION](https://aws.amazon.com/guardduty/)

Amazon GuardDuty is a managed threat detection service that provides you with a more accurate and easy way to continuously monitor and protect your AWS accounts and workloads.

[SECURITY ASSESSMENT](https://aws.amazon.com/inspector/)

Amazon Inspector is an automated security assessment service that helps improve the security and compliance of applications deployed on AWS.

[VIRTUAL PRIVATE CLOUD](https://aws.amazon.com/vpc/)

Amazon Virtual Private Cloud (VPC) lets you provision a logically isolated section of the AWS Cloud where you can launch AWS resources in a virtual network that you define.

[WAF MANAGEMENT](https://aws.amazon.com/waf/)

AWS Firewall Manager is a security management service that makes it easier to centrally configure and manage AWS WAF rules across your accounts and applications.

[DDOS PROTECTION](https://aws.amazon.com/shield/)

AWS Shield is a managed Distributed Denial of Service (DDoS) protection service that safeguards applications running on AWS.

[ENCRYPTION KEYS](https://aws.amazon.com/kms/)

AWS Key Management Service (KMS) is a managed service that makes it easy for you to create and control the encryption keys used to encrypt your data.

[SSL/TLS CERTIFICATE MANAGEMENT](https://aws.amazon.com/certificate-manager/)

AWS Certificate Manager lets you easily provision, manage, and deploy public and private Secure Sockets Layer/Transport Layer Security (SSL/TLS) certificates for use with AWS services and your internal connected resources.

**Authentication in AWS:**

Key Points

* To authenticate from the console as a root user, you must sign in with your email address and password. As an IAM user, provide your account ID or alias, and then your user name and password.
* To authenticate from the API or AWS CLI, you must provide your access key and secret key.
* The AWS Security Token Service (STS) is a web service that enables you to request temporary, limited-privilege credentials for AWS Identity and Access Management (IAM) users or for users that you authenticate (federated users.

**IAM Authorization:**

The following policy types, listed in order of frequency, are available for use in AWS. For more details, see the sections below for each policy type.

* [**Identity-based policies**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html) – Attach managed and inline policies to IAM identities (users, groups to which users belong, or roles). Identity-based policies grant permissions to an identity.
* [**Resource-based policies**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html) – Attach inline policies to resources. The most common examples of resource-based policies are Amazon S3 bucket policies and IAM role trust policies. Resource-based policies grant permissions to a principal entity that is specified in the policy. Principals can be in the same account as the resource or in other accounts.
* [**Permissions boundaries**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html) – Use a managed policy as the permissions boundary for an IAM entity (user or role). That policy defines the maximum permissions that the identity-based policies can grant to an entity, but does not grant permissions. Permissions boundaries do not define the maximum permissions that a resource-based policy can grant to an entity.
* [**Organizations SCPs**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html) – Use an AWS Organizations service control policy (SCP) to define the maximum permissions for account members of an organization or organizational unit (OU). SCPs limit permissions that identity-based policies or resource-based policies grant to entities (users or roles) within the account, but do not grant permissions.
* [**Access control lists (ACLs)**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html) – Use ACLs to control which principals in other accounts can access the resource to which the ACL is attached. ACLs are similar to resource-based policies, although they are the only policy type that does not use the JSON policy document structure. ACLs are cross-account permissions policies that grant permissions to the specified principal entity. ACLs cannot grant permissions to entities within the same account.
* [**Session policies**](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html) – Pass advanced session policies when you use the AWS CLI or AWS API to assume a role or a federated user. Session policies limit the permissions that the role or user's identity-based policies grant to the session. Session policies limit permissions for a created session, but do not grant permissions. For more information, see [Session Policies](https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html).

**Infrastructure Security:**

Key Points

* Security groups are stateful: This means any changes applied to an incoming rule will be automatically applied to the outgoing rule. e.g. If you allow an incoming port 80, the outgoing port 80 will be automatically opened.
* Network ACLs are stateless: This means any changes applied to an incoming rule will not be applied to the outgoing rule. e.g. If you allow an incoming port 80, you would also need to apply the rule for outgoing traffic.

**Encryption in AWS:**

Resources

* <https://aws.amazon.com/blogs/security/how-to-share-encrypted-amis-across-accounts-to-launch-encrypted-ec2-instances/>
* <https://aws.amazon.com/blogs/security/are-kms-custom-key-stores-right-for-you/>
* <https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html>
* [https://docs.aws.**amazon**.com/AmazonS3/latest/dev/bucket-encryption.html](https://docs.aws.amazon.com/AmazonS3/latest/dev/bucket-encryption.html)

Key Points

* When you encrypt your data, your data is protected, but you have to protect your encryption key. One strategy is to encrypt it. Envelope encryption is the practice of encrypting plaintext data with a data key, and then encrypting the data key under another key.

**API Gateway Security:**

* **Resource policies** let you create resource-based policies to allow or deny access to your APIs and methods from *specified source IP addresses or VPC endpoints*. For more information, see Control Access to an API with Amazon API Gateway Resource Policies.
* **Standard AWS IAM roles** and policies offer flexible and robust access controls that can be applied to an entire API or individual methods. IAM roles and policies can be used for controlling who can create and manage your APIs as well as who can invoke them. For more information, see Control Access to an API with IAM Permissions.
* **IAM tags can be used together with IAM policies to control access**. For more information, see Using Tags to Control Access to API Gateway Resources.
* **Endpoint Policies** for Interface VPC Endpoints allow you to attach IAM resource policies to interface VPC endpoints to improve the security of your private APIs. For more information, see Use VPC Endpoint Policies for Private APIs in API Gateway.
* **Lambda authorizers** are Lambda functions that control access to REST API methods using bearer token authentication as well as information described by headers, paths, query strings, stage variables, or context variables request parameters. Lambda authorizers are used to control who can invoke REST API methods. For more information, see Use API Gateway Lambda Authorizers.
* **Amazon Cognito** user pools let you create customizable authentication and authorization solutions for your REST APIs. Amazon Cognito user pools are used to control who can invoke REST API methods. For more information, see Control Access to a REST API Using Amazon Cognito User Pools as Authorizer.

**Lambda Security:**

Resources

* <https://d1.awsstatic.com/whitepapers/Overview-AWS-Lambda-Security.pdf>

Key Points

* For AWS Lambda, AWS manages the underlying infrastructure and foundation services, the operating system, and the application platform. You are responsible for the security of your **code, the storage and accessibility of sensitive data, and identity and access management (IAM) to the Lambda service** and within your function.
* Each function runs in one or more **dedicated execution environments** that are used for the lifetime of the function and then destroyed. Each execution environment hosts one concurrent invocation, but is reused in place across multiple serial invocations of the same function.
* Each execution environment contains a dedicated copy of the following items:
* The function code
* Any Lambda layers selected for your function
* The function runtime, either built-in (Java 8, NodeJS 8, Python 3.7, etc.) or custom runtime
* A minimal Linux userland based on Amazon Linux
* Though Lambda execution environments are never reused across functions**, a single execution environment can be reused for invoking the same function**, potentially existing for hours before it is destroyed. Functions can take advantage of this behavior to improve efficiency by keeping local caches, reusing long-lived connections between invocations, and pre-computing common results.

**DynamoDB Security:**

Resources

* <https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/best-practices-security.html>

Key Points

* **DynamoDB encrypts at rest all user data** stored in tables, indexes, streams, and backups using encryption keys stored in AWS Key Management Service (AWS KMS). This provides an additional layer of data protection by securing your data from unauthorized access to the underlying storage .
* You can specify whether DynamoDB should use an AWS owned CMK (default encryption type) or an AWS managed CMK to encrypt the user data.
* for users, applications, and other AWS services to access DynamoDB, they must include valid AWS credentials in their AWS API requests. You should not store AWS credentials directly in the application or EC2 instance.
* If you only require access to DynamoDB from within a virtual private cloud (VPC), you should use a **VPC endpoint to limit access from only the required VPC**. Doing this prevents that traffic from traversing the open internet and being subject to that environment.
* if you store sensitive or confidential data in DynamoDB, you might want to **encrypt that data as close as possible to its origin** so that your data is protected throughout its lifecycle. Encrypting your sensitive data in transit and at rest helps ensure that your plaintext data isn’t available to any third party.
* The **Amazon DynamoDB Encryption Client** is a software library that helps you protect your table data before you send it to DynamoDB.