Encryption best practices and features for AWS services

AWS Prescriptive Guidance

AWS Prescriptive Guidance: Encryption best practices and features for AWS services

Table of Contents

Introduction **1**
Intended audience 1
Cryptography approach **3**
AWS cryptographic foundations 3
Cryptographic algorithms 3
Recommended cryptographic algorithms in AWS 4
Asymmetric cryptography 4
Symmetric cryptography 5
Other cryptographic functions 6
Cryptography used in AWS services 6
General encryption best practices **8**
Data classification 8
Encryption of data in transit 8
Encryption of data at rest 9
Encryption best practices for AWS services **11**
AWS CloudTrail 11
Amazon DynamoDB 12
Amazon EC2 and Amazon EBS 14
Amazon ECR 15
Amazon ECS 16
Amazon EFS 17
Amazon EKS 18
AWS Encryption SDK 20
AWS KMS 21
AWS Lambda 24
Amazon RDS 24
AWS Secrets Manager 26
Amazon S3 27
Amazon VPC 28
Resources **30**
Document history **31**
Glossary **33**
..... 33
A 34

B 37
C 39
D 42
E 46
F 48
G 50
H 51
I 52
L 54
M 56
O 60
P 62
Q 65
R 65
S 68
T 72
U 73
V 74
W 74
Z 75

Encryption best practices and features for AWS services

Kurt Kumar, Amazon Web Services

February 2026 (document history)

Encryption is a fundamental cybersecurity tool for protecting sensitive data in the digital age. As organizations increasingly rely on data to drive their operations, including generative AI deployments, safeguarding this valuable information through robust encryption practices is an essential component of a comprehensive data protection strategy. This guide can help you understand encryption principles and the encryption capabilities that AWS offers.

Modern cybersecurity threats include the risk of a *data breach*, which is when unauthorized access to your information assets results in the loss of data. Data is a business asset that is unique to each organization. It can include customer information, business plans, design documents, or code. Protecting the business means protecting its data.

Data encryption can help protect your business data even after a breach occurs. It provides a layer of defense against unintended disclosure. To access encrypted data in the AWS Cloud, users need permissions to use the key to decrypt and need permissions to use the service where the data resides. Without both of these permissions, users are unable to decrypt and view the data.

Generally, there are three types of data that you can encrypt. *Data in transit* is data that is actively moving through your network, such as between network resources. *Data at rest* is data that is stationary and dormant, such as data that is in storage. Examples include block storage, object storage, databases, archives, and Internet of Things (IoT) devices. *Data in use* refers to data that applications or services are actively processing or using. By securing data at the point of use, organizations can help mitigate the risks of unintended disclosure.

This guide discusses considerations and best practices for encrypting data in transit and data at rest. It also reviews the encryption features and controls that are available in many AWS services. You can implement these encryption recommendations at the service level in your AWS Cloud environments.

Intended audience

This guide can be used by small, medium, and large organizations in both public and private sectors. Whether your organization is in the initial stages of assessing and implementing a data

protection strategy or aiming to enhance existing security controls, the recommendations outlined in this guide are best suited for the following audiences:

- Executive officers who formulate policies for their enterprise, such as chief executive officers (CEOs), chief technology officers (CTOs), chief information officers (CIOs), and chief information security officers (CISOs)
- Technology officers who are responsible for setting up technical standards, such as technical vice presidents and directors
- Business stakeholders and application owners who are responsible for:
  - Assessing risk posture, data classification, and protection requirements
  - Monitoring compliance with established organizational standards
- Compliance, internal audit, and governance officers who are in charge of monitoring adherence to compliance policies, including statutory and voluntary compliance regimes

AWS approach to cryptography

Cryptographic algorithms are mathematical constructions designed to provide security services like confidentiality (encryption), authenticity (message authentication codes and digital signatures) and non-repudiation (digital signatures). If you are new to cryptography, encryption, and related terminology, we recommend that you read About data encryption before proceeding with this guide.

AWS cryptographic foundations

Cryptography is an essential part of security for AWS. AWS services support encryption for data in transit, at rest, or in memory. You can learn more about the AWS commitment to innovation and investing in additional controls for sovereignty and encryption features in our blog post announcing the AWS digital sovereignty pledge.

AWS follows the shared responsibility model to protect your data. AWS services use trusted cryptographic algorithms that meet industry standards and foster interoperability. These algorithms are vetted by public standards bodies and academic research. The associated standards are widely accepted by governments, industry, and academia.

AWS defaults to high-assurance cryptographic implementations and prefers hardware-optimized solutions that are efficient. Our cryptographic core library, AWS-LC, is available as open source for transparency and industry-wide reuse. Many cryptographic algorithm implementations in AWS-LC are formally verified to increase assurance of the correctness and security of the implementation in several different platforms. The library is also validated under NIST's FIPS-140 program.

Cryptographic algorithms

We define three types of cryptographic algorithms:

- *Asymmetric cryptography* uses a pair of keys: a public key for encryption (or verifying) and a private key for decryption (or signing). You can share the public key because it isn't used for decryption, but access to the private key should be highly restricted. AWS services support or plan to support post-quantum algorithms, such as ML-KEM and ML-DSA. AWS services also support traditional cryptographic algorithms, such as RSA and elliptic-curve cryptography (ECC).

- *Symmetric cryptography* uses the same key to encrypt and decrypt, or authenticate and verify the data. AWS services generally integrate with AWS Key Management Service (AWS KMS) for encryption of data at rest, which uses a mode of AES-256.
- *Other cryptographic functions* are used in conjunction with asymmetric and symmetric cryptography to build secure, practical protocols for confidentiality, integrity, authentication, and non-repudiation applications. Examples include hash functions and key derivation functions.

Recommended cryptographic algorithms in AWS

The following tables summarize the cryptographic algorithms, modes, and key sizes that AWS considers suitable for deployment across its services to protect your data. This guidance will evolve over time as cryptographic standards evolve.

Algorithms available within services can vary and are explained in the documentation for each service. If you need a software library implementation for an approved algorithm, please check to see if it is included in the latest version of the AWS-LC library.

Algorithms are approved for use in AWS under one of two categories:

- *Preferred* algorithms meet the AWS security and performance standards.
- *Acceptable* algorithms can be used for compatibility in some applications but are not preferred.

Asymmetric cryptography

The following table lists asymmetric algorithms considered suitable for use within AWS for encryption, key agreement, and digital signatures.

Type	Algorithm	Status
Encryption	RSA-OAEP (≥2048-bit modulus)	Acceptable
Encryption	HPKE (P-256 or P-384, HKDF and AES-GCM)	Acceptable
Key agreement	ML-KEM-768 or ML-KEM-1024	Preferred (quantum-resistant)

Key agreement	ECDH(E) with P-256, P-384, P-521, or X25519	Acceptable
Key agreement	ECDH(E) with brainpool P256r1, brainpoolP384r1, or brainpoolP512r1	Acceptable
Signatures	ML-DSA-65 or ML-DSA-87	Preferred (quantum-resistant)
Signatures	SLH-DSA	Acceptable (quantum-resistant)
Signatures	ECDSA with P-256, P-384, P-521, or Ed25519	Acceptable
Signatures	RSA (≥2048-bit modulus)	Acceptable

Symmetric cryptography

The following table lists symmetric algorithms considered suitable for use within AWS for encryption, authenticated encryption, and key wrapping.

Type	Algorithm	Status
Authenticated encryption	AES-GCM-256	Preferred
Authenticated encryption	AES-GCM-128	Acceptable
Authenticated encryption	ChaCha20/Poly1305	Acceptable
Encryption modes	AES-XTS-256 (for block storage)	Preferred
Encryption modes	AES-CBC / CTR (unauthenticated modes)	Acceptable
Key wrapping	AES-GCM-256	Preferred

Key wrapping	AES-KW or AES-KWP with 256-bit keys	Acceptable

Other cryptographic functions

The following table lists algorithms considered suitable for use within AWS for hashing, key derivation, and message authentication.

Type	Algorithm	Status
Hashing	SHA-384	Preferred
Hashing	SHA-256	Acceptable
Hashing	SHA3	Acceptable
Key derivation	HKDF_Expand or HKDF with SHA-256	Preferred
Key derivation	Counter Mode KDF with HMAC-SHA-256	Acceptable
Message authentication code	HMAC-SHA-384	Preferred
Message authentication code	HMAC-SHA-256	Acceptable
Message authentication code	KMAC	Acceptable
Password hashing	scrypt with SHA384	Preferred
Password hashing	PBKDF2	Acceptable

Cryptography used in AWS services

AWS services rely on secure, open-source implementations of vetted algorithms to protect your data. The specific choices and configurations of algorithms will vary by service. Some AWS tools and services use a specific algorithm. In others, you can choose between supported algorithms and key lengths, or you can use the recommended defaults.

AWS cryptographic services comply with a wide range of cryptographic security standards, so you can comply with governmental or industry regulations. For a full list of the data security standards that AWS services comply with, see AWS compliance programs.

General encryption best practices

This section provides recommendations that apply when encrypting data in the AWS Cloud. These general encryption best practices are not specific to AWS services. This section includes the following topics:

- Data classification
- Encryption of data in transit
- Encryption of data at rest

Data classification

Data classification is a process for identifying and categorizing the data in your network based on its criticality and sensitivity. It is a critical component of any cybersecurity risk management strategy because it helps you determine the appropriate protection and retention controls for the data. Data classification is a component of the security pillar in the AWS Well-Architected Framework. Categories might include *highly confidential*, *confidential*, *non-confidential*, and *public*, but the classification tiers and their names can vary from organization to organization. For more information about the data classification process, considerations, and models, see Data classification (AWS Whitepaper).

After you have classified your data, you can create an encryption strategy for your organization based on the level of protection required for each category. For example, your organization might decide that highly confidential data should use asymmetric encryption and that public data doesn't require encryption. For more information about designing an encryption strategy, see Creating an enterprise encryption strategy for data at rest. Although the technical considerations and recommendations in that guide are specific to data at rest, you can use the phased approach to create an encryption strategy for data in transit as well.

Encryption of data in transit

All data transmitted between AWS Regions over the AWS global network is automatically encrypted by AWS at the physical layer before it leaves AWS secured facilities. AWS encrypts all traffic between Availability Zones.

For data flowing through your workloads, the following are general best practices when encrypting data in transit in the AWS Cloud:

- Define an organizational encryption policy for data in transit, based on your data classification, organizational requirements, and any applicable regulatory or compliance standards. We strongly recommend that you encrypt data in transit that is classified as highly confidential or confidential. Your policy might also specify encryption for other categories, such as non-confidential or public data, on an as-needed basis.
- When encrypting data in transit, we recommend using approved cryptography algorithms, block cipher modes, and key lengths, as defined in your encryption policy. We further recommend periodically reviewing the TLS policies associated with your Application Load Balancers, Amazon API Gateway resources, Amazon CloudFront resources, and Amazon Virtual Private Cloud (Amazon VPC) resources to make sure that they are aligned with your current encryption policy.
- Encrypt traffic between information assets and systems within the corporate network and AWS Cloud infrastructure by using one of the following:
  - AWS Site-to-Site VPN connections
  - A combination of AWS Site-to-Site VPN and AWS Direct Connect connections, which provides an IPsec-encrypted private connection
  - Direct Connect connections that support MAC Security (MACsec) to encrypt data from corporate networks to the Direct Connect location
- Identify access control policies for managed certificates and TLS policy configurations based on the principle of least privilege. *Least privilege* is the security best practice of granting users the minimum access they need to perform their job functions. For more information about applying least-privilege permissions, see Security best practices in IAM and Best practices for IAM policies.

Encryption of data at rest

All AWS data storage services, such as Amazon Simple Storage Service (Amazon S3) and Amazon Elastic File System (Amazon EFS), provide options to encrypt data at rest. Encryption is performed by using the 256-bit Advanced Encryption Standard (AES-256) block cipher and AWS cryptography services, such as AWS Key Management Service (AWS KMS) or AWS CloudHSM.

You can encrypt data using client-side encryption or server-side encryption, based on factors such as data classification, the need for end-to-end encryption, or technical limitations that prevent you from using end-to-end encryption:

- *Client-side encryption* is the act of encrypting data locally before the target application or service receives it. The AWS service receives encrypted data; it does not play a role in encrypting or decrypting it. For client-side encryption, you might use AWS KMS, the AWS Encryption SDK, or other third-party encryption tools or services.
- *Server-side encryption* is the act of encrypting data at its destination, by the application or service that receives it. For server-side encryption, you might use AWS KMS for encryption of the entire storage block. You can also use other third-party encryption tools or services, such as LUKS for encrypting a Linux file system at the operating system (OS) level.

The following are general best practices when encrypting data at rest in the AWS Cloud:

- Define an organizational encryption policy for data at rest, based on your data classification, organizational requirements, and any applicable regulatory or compliance standards. For more information, see Creating an enterprise encryption strategy for data at rest. We strongly recommend that you encrypt data at rest that is classified as highly confidential or confidential. Your policy might also specify encryption for other categories, such as non-confidential or public data, on an as-needed basis.
- When encrypting data at rest, we recommend using approved cryptography algorithms, block cipher modes, and key lengths.
- Identify access control policies for your encryption keys based on the principle of least privilege.

Encryption best practices for AWS services

This section includes best practices and recommendations for the following AWS services:

- AWS CloudTrail
- Amazon DynamoDB
- Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Elastic Block Store (Amazon EBS)
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon Elastic Container Service (Amazon ECS)
- Amazon Elastic File System (Amazon EFS)
- Amazon Elastic Kubernetes Service (Amazon EKS)
- AWS Encryption SDK
- AWS Key Management Service (AWS KMS)
- AWS Lambda
- Amazon Relational Database Service (Amazon RDS)
- AWS Secrets Manager
- Amazon Simple Storage Service (Amazon S3)
- Amazon Virtual Private Cloud (Amazon VPC)

Encryption best practices for AWS CloudTrail

AWS CloudTrail helps you audit the governance, compliance, and operational and risk of your AWS account.

Consider the following encryption best practices for this service:

- CloudTrail logs should be encrypted using a customer managed AWS KMS key. Choose a KMS key that is in the same region as the S3 bucket that receives your log files. For more information, see Updating a trail to use your KMS key.
- As an additional security layer, enable log file validation for trails. This helps you determine whether a log file was modified, deleted, or unchanged after CloudTrail delivered it. For instructions, see Enabling log file integrity validation for CloudTrail.

- Use interface VPC endpoints to enable CloudTrail to communicate with resources in other VPCs without traversing the public internet. For more information, see Using AWS CloudTrail with interface VPC endpoints.
- Add an `aws:SourceArn` condition key to the KMS key policy to ensure that CloudTrail uses the KMS key only for a specific trail or trails. For more information, see Configure AWS KMS key policies for CloudTrail.
- In AWS Config, implement the cloud-trail-encryption-enabled AWS managed rule to validate and enforce log file encryption.
- If CloudTrail is configured to send notifications through Amazon Simple Notification Service (Amazon SNS) topics, add an `aws:SourceArn` (or optionally `aws:SourceAccount`) condition key to the CloudTrail policy statement to prevent unauthorized account access to the SNS topic. For more information, see Amazon SNS topic policy for CloudTrail.
- If you are using AWS Organizations, create an organization trail that logs all events for the AWS accounts in that organization. This includes the management account and all member accounts in the organization. For more information, see Creating a trail for an organization.
- Create a trail that applies to all AWS Regions where you store corporate data, to record AWS account activity in those Regions. When AWS launches a new Region, CloudTrail automatically includes the new Region and logs events in that Region.

Encryption best practices for Amazon DynamoDB

Amazon DynamoDB is a fully managed NoSQL database service that provides fast, predictable, and scalable performance. DynamoDB encryption at rest secures data in an encrypted table—including its primary key, local and global secondary indexes, streams, global tables, backups, and DynamoDB Accelerator (DAX) clusters whenever the data is stored in durable media.

In accordance with data classification requirements, data confidentiality and integrity can be maintained by implementing server-side or client-side encryption:

For server-side encryption, when you create a new table, you can use AWS KMS keys to encrypt the table. You can use AWS owned keys, AWS managed keys, or customer managed keys. We recommend using customer managed keys because your organization has full control of the key, and because when you use this key type, the table-level encryption key, the DynamoDB table, local and global secondary indexes, and streams are all encrypted with the same key. For more information about these key types, see Customer keys and AWS keys.

> **Note**
>
> You can switch between an AWS owned key, AWS managed key, and customer managed key at any given time.

For client-side encryption and end-to-end protection of data, both at rest and in transit, you can use the Amazon DynamoDB Encryption Client. In addition to encryption, which protects the confidentiality of the item attribute value, DynamoDB Encryption Client signs the item. This provides integrity protection by enabling detection of unauthorized changes to the item, including adding or deleting attributes, or substituting one encrypted value for another.

Consider the following encryption best practices for this service:

- Limit permissions to disable or schedule deletion of the key to only those who need to perform these tasks. These states prevent all users and the DynamoDB service from being able to encrypt or decrypt data and to perform read and write operations on the table.
- While DynamoDB encrypts data in transit by using HTTPS by default, additional security controls are recommended. You can use any of the following options:
  - AWS Site-to-Site VPN connection using IPsec for encryption.
  - AWS Direct Connect connection to establish a private connection.
  - AWS Direct Connect connection with AWS Site-to-Site VPN connection for an IPsec-encrypted private connection.
  - If access to DynamoDB is required only from within a virtual private cloud (VPC), you can use a VPC gateway endpoint and allow only resources in the VPC to access it. This prevents the traffic from traversing the public internet.
- If you are using VPC endpoints, restrict the endpoint policies and IAM policies associated with the endpoint to only authorized users, resources, and services. For more information, see Control access to DynamoDB endpoints by using IAM policies and Control access to services using endpoint policies.
- You can implement column-level data encryption at the application level for data that requires encryption, according to your encryption policy.
- Configure DAX clusters to encrypt data at rest, such as data in cache, configuration data, and log files, at the time of setting up the cluster. You can't enable encryption at rest on an existing cluster. This server-side encryption helps protect data from unauthorized access through the underlying storage. DAX encryption at rest automatically integrates with AWS KMS for managing

the single-service default key that is used to encrypt the clusters. If a service default key doesn't exist when an encrypted DAX cluster is created, AWS KMS automatically creates a new AWS managed key. For more information, see DAX encryption at rest.

> **Note**
>
> Customer managed keys can't be used with DAX clusters.

- Configure DAX clusters to encrypt data in transit at the time of setting up the cluster. You can't enable encryption in transit on an existing cluster. DAX uses TLS to encrypt requests and responses between the application and the cluster, and it uses the cluster's x509 certificate to authenticate the identity of the cluster. For more information, see DAX encryption in transit.
- In AWS Config, implement the dax-encryption-enabled AWS managed rule to validate and maintain encryption of DAX clusters.

Encryption best practices for Amazon EC2 and Amazon EBS

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the AWS Cloud. You can launch as many virtual servers as you need and quickly scale them up or down. Amazon Elastic Block Store (Amazon EBS) provides block-level storage volumes for use with EC2 instances.

Consider the following encryption best practices for these services:

- Tag all EBS volumes with the appropriate data classification key and value. This helps you determine and implement the appropriate security and encryption requirements, according to your policy.
- According to your encryption policy and the technical feasibility, configure encryption for data in transit between EC2 instances or between EC2 instances and your on-premises network.
- Encrypt both the boot and data EBS volumes of an EC2 instance. An encrypted EBS volume protects the following data:
  - Data at rest inside the volume
  - All data moving between the volume and the instance
  - All snapshots created from the volume
  - All volumes created from those snapshots

For more information, see How EBS encryption works.

- Enable encryption by default for EBS volumes for your account in the current AWS Region. This enforces encryption of any new EBS volumes and snapshot copies. It has no effect on existing EBS volumes or snapshots. For more information, see Enable encryption by default.
- Encrypt the instance store root volume for an Amazon EC2 instance. This helps you protect configuration files and data stored with the operating system. For more information, see How to protect data at rest with Amazon EC2 instance store encryption (AWS blog post)
- In AWS Config, implement the encrypted-volumes rule to automated checks that validate and enforce appropriate encryption configurations.

Encryption best practices for Amazon ECR

Amazon Elastic Container Registry (Amazon ECR) is a managed container image registry service that's secure, scalable, and reliable.

Amazon ECR stores images in Amazon S3 buckets that Amazon ECR manages. Each Amazon ECR repository has an encryption configuration, which is set when the repository is created. By default, Amazon ECR uses server-side encryption with Amazon S3-managed (SSE-S3) encryption keys. For more information, see Encryption at rest (Amazon ECR documentation).

Consider the following encryption best practices for this service:

- Instead of using the default server-side encryption with Amazon S3-managed (SSE-S3) encryption keys, use customer managed KMS keys stored in AWS KMS. This key type provides the most granular control options.

> **Note**
>
> The KMS key must exist in the same AWS Region as the repository.

- Do not revoke the grants that Amazon ECR creates by default when you provision a repository. This can affect functionality, such as accessing data, encrypting new images pushed to the repository, or decrypting them when they are pulled.
- Use AWS CloudTrail to record the requests that Amazon ECR sends to AWS KMS. The log entries contain an encryption context key to make them more easily identifiable.

- Configure Amazon ECR policies to control access from specific Amazon VPC endpoints or specific VPCs. Effectively, this isolates network access to a specific Amazon ECR resource, allowing access from only the specific VPC. By establishing a virtual private network (VPN) connection with an Amazon VPC endpoint, you can encrypt data in transit.
- Amazon ECR supports resource-based policies. Using these policies, you can restrict access based on the source IP address or the specific AWS service.

Encryption best practices for Amazon ECS

Amazon Elastic Container Service (Amazon ECS) is a fast and scalable container management service that helps you run, stop, and manage containers on a cluster.

With Amazon ECS, you can encrypt data in transit by using any of the following approaches:

- Create a service mesh. Using AWS App Mesh, configure TLS connections between the deployed Envoy proxies and mesh endpoints, such as virtual nodes or virtual gateways. You can use TLS certificates from AWS Private Certificate Authority or customer-provided certificates. For more information and walkthroughs, see Enable traffic encryption between services in AWS App Mesh using AWS Certificate Manager (ACM) or customer-provided certificates (AWS blog post).
- If supported, use AWS Nitro Enclaves. AWS Nitro Enclaves is an Amazon EC2 feature that allows you to create isolated execution environments, called *enclaves*, from Amazon EC2 instances. They are designed to help protect your most sensitive data. Additionally, ACM for Nitro Enclaves allows you to use public and private SSL/TLS certificates with your web applications and web servers running on Amazon EC2 instances with AWS Nitro Enclaves. For more information, see AWS Nitro Enclaves – Isolated EC2 Environments to Process Confidential Data (AWS blog post).
- Use Server Name Indication (SNI) protocol with Application Load Balancers. You can deploy multiple applications behind a single HTTPS listener for an Application Load Balancer. Each listener has its own TLS certificate. You can certificates provided by ACM, or you can use self-signed certificates. Both Application Load Balancer and Network Load Balancer support SNI. For more information, see Application Load Balancers Now Support Multiple TLS Certificates with Smart Selection Using SNI (AWS blog post).
- For improved security and flexibility, use AWS Private Certificate Authority to deploy a TLS certificate with the Amazon ECS task. For more information, see Maintaining TLS all the way to your container part 2: Using AWS Private CA (AWS blog post).
- Implement mutual TLS (mTLS) in App Mesh by using Secret discovery service (Envoy) or certificates hosted in ACM (GitHub).

Consider the following encryption best practices for this service:

- Where technically feasible, for enhanced security, configure Amazon ECS interface VPC endpoints in AWS PrivateLink. Accessing these endpoints over a VPN connection encrypts data in transit.
- Store sensitive materials, such as API keys or database credentials, securely. You can store these as encrypted parameters in Parameter Store, a capability of AWS Systems Manager. However, we recommend you use AWS Secrets Manager because this service allows you to automatically rotate secrets, generate random secrets, and share secrets across AWS accounts.
- If users or applications in your data center or an external third party on the web are making direct HTTPS API requests to AWS services, sign those requests with temporary security credentials obtained from AWS Security Token Service (AWS STS).

Encryption best practices for Amazon EFS

Amazon Elastic File System (Amazon EFS) helps you create and configure shared file systems in the AWS Cloud.

Consider the following encryption best practices for this service:

- In AWS Config, implement the efs-encrypted-check AWS managed rule. This rule checks if Amazon EFS is configured to encrypt the file data using AWS KMS.
- Enforce encryption for Amazon EFS file systems by creating an Amazon CloudWatch alarm that monitors CloudTrail logs for `CreateFileSystem` events and triggers an alarm if an unencrypted file system is created. For more information, see Walkthrough: Enforcing Encryption on an Amazon EFS File System at Rest.
- Mount the file system by using the EFS mount helper. This sets up and maintains a TLS 1.2 tunnel between the client and the Amazon EFS service and routes all Network File System (NFS) traffic over this encrypted tunnel. The following command implements the use of TLS for in-transit encryption.

  ```
  sudo mount -t efs  -o tls file-system-id:/ /mnt/efs
  ```

  For more information, see Using EFS mount helper to mount EFS file systems.
- Using AWS PrivateLink, implement interface VPC endpoints to establish a private connection between VPCs and the Amazon EFS API. Data in transit over the VPN connection to and from the

endpoint is encrypted. For more information, see Access an AWS service using an interface VPC endpoint.

- Use the `elasticfilesystem:Encrypted` condition key in IAM identity-based policies to prevent users from creating EFS file systems that aren't encrypted. For more information, see Using IAM to enforce creating encrypted file systems.
- KMS keys used for EFS encryption should be configured for least-privilege access by using resource-based key policies.
- Use the `aws:SecureTransport` condition key in the EFS file system policy to enforce use of TLS for NFS clients when connecting to an EFS file system. For more information, see Encryption of data in transit in *Encrypting File Data with Amazon Elastic File System* (AWS Whitepaper).

Encryption best practices for Amazon EKS

Amazon Elastic Kubernetes Service (Amazon EKS) helps you run Kubernetes on AWS without needing to install or maintain your own Kubernetes control plane or nodes. In Kubernetes, *secrets* help you manage sensitive information, such as user certificates, passwords, or API keys. By default, these secrets are stored unencrypted in the API server's underlying data store, which is called etcd. On Amazon EKS, the Amazon Elastic Block Store (Amazon EBS) volumes for `etcd` nodes are encrypted with Amazon EBS encryption. Any user with API access or access to `etcd` can retrieve or modify a secret. Additionally, anyone who is authorized to create a pod in a namespace can use that access to read any secret in that namespace. You can encrypt these secrets at rest in Amazon EKS by using AWS KMS keys, either AWS managed keys or customer managed keys. An alternative approach to using `etcd` is using AWS Secrets and Config Provider (ASCP) (GitHub repository). ASCP integrates with IAM and resource-based policies to limit and restrict access to secrets only within specific Kubernetes pods inside a cluster.

You can use the following AWS storage services with Kubernetes:

- For Amazon EBS, you can use the in-tree storage driver or the Amazon EBS CSI driver. Both include parameters for encrypting volumes and supplying a customer managed key.
- For Amazon Elastic File System (Amazon EFS), you can use the Amazon EFS CSI driver with support for both dynamic and static provisioning.

Consider the following encryption best practices for this service:

- If you are using `etcd`, which stores secret objects unencrypted by default, do the following to help protect secrets:
  - Encrypt secret data at rest (Kubernetes documentation).
  - Use AWS KMS for envelope encryption of Kubernetes secrets. This allows you to encrypt your secrets with a unique data key. You can use an AWS KMS key encryption key to encrypt the data key. You can automatically rotate the key encryption key on a recurring schedule. With the AWS KMS plugin for Kubernetes, all Kubernetes secrets are stored in `etcd` in ciphertext. They can only be decrypted by the Kubernetes API server. For more information, see Using Amazon EKS encryption provider support for defense in depth and Encrypt Kubernetes secrets with AWS KMS on existing clusters.
  - Enable or configure authorization through role-based access control (RBAC) rules that restrict reading and writing the secret. Restrict permissions to create new secrets or replace existing ones. For more information, see Authorization overview (Kubernetes documentation).
  - If you are defining multiple containers in a pod and only one of those containers needs access to a secret, define the volume mount so that the other containers do not have access to that secret. Secrets that are mounted as volumes are instantiated as `tmpfs` volumes and are automatically removed from the node when the pod is deleted. You could also use environment variables, but we don't recommend this approach because the values of environment variables can appear in logs. For more information, see Secrets (Kubernetes documentation).
  - When possible, avoid granting access to `watch` and `list` requests for secrets within a namespace. In the Kubernetes API, these requests are powerful because they allow the client to inspect the values of every secret in that namespace.
  - Allow only cluster administrators to access `etcd`, including read-only access.
  - If there are multiple `etcd` instances, ensure `etcd` is using TLS for communication between `etcd` peers.
- If you are using ASCP, do the following to help protect secrets:
  - Use IAM roles for service accounts to limit secret access to only authorized pods.
  - Enable encryption of Kubernetes secrets by using the AWS Encryption Provider (GitHub repository) to implement envelope encryption with a customer managed KMS key.
- To help mitigate the risk of data leaks from environment variables, we recommend you use the AWS Secrets Manager and Config Provider for Secret Store CSI Driver (GitHub). This driver allows you to make secrets stored in Secrets Manager and parameters stored in Parameter Store appear as files mounted in Kubernetes pods.

> **Note**
>
> AWS Fargate is not supported.

- Create an Amazon CloudWatch metrics filter and alarm to send alerts for administrator-specified operations, such as secret deletion or use of a secret version in the waiting period to be deleted. For more information, see Creating an alarm based on anomaly detection.

Encryption best practices for AWS Encryption SDK

The AWS Encryption SDK is an open-source, client-side encryption library. It uses industry standards and best practices to support implementation and interoperability in several programming languages. AWS Encryption SDK encrypts data by using a secure, authenticated, symmetric key algorithm and offers default implementation that adheres to cryptography best practices. For more information, see Supported algorithm suites in the AWS Encryption SDK.

One of the key features of the AWS Encryption SDK is support for encrypting data in use. By adopting an encrypt-then-use approach, you can encrypt sensitive data before it is processed by your application logic. This can help protect the data from potential exposure or tampering, even if the application itself is affected by a security event.

Consider the following best practices for this service:

- Adhere to all of the recommendations in Best practices for the AWS Encryption SDK.
- Select one or more wrapping keys to help protect your data keys. For more information, see Select wrapping keys.
- Pass the `KeyId` parameter to the ReEncrypt operation to help prevent use of an untrusted KMS key. For more information, see Improved client-side encryption: Explicit KeyIds and key commitment (AWS blog post).
- When using the AWS Encryption SDK with AWS KMS, use local `KeyId` filtering. For more information, see Improved client-side encryption: Explicit KeyIds and key commitment (AWS blog post).
- For applications with large volumes of traffic requiring encryption or decryption, or if your account is exceeding AWS KMS request quotas, you can use the data key caching feature of the AWS Encryption SDK. Note the following best practices for data key caching:

- Configure cache security thresholds to limit how long each cached data key is used and how much data is protected under each data key. For recommendations when configuring these thresholds, see Setting cache security thresholds.
- Limit the local cache to the smallest number of data keys necessary to achieve the performance improvements for your specific application use case. For instructions and an example of configuring limits for the local cache, see Using data key caching: Step-by-step.

For more information, see AWS Encryption SDK: How to Decide if Data Key Caching Is Right for Your Application (AWS blog post).

Encryption best practices for AWS Key Management Service

AWS Key Management Service (AWS KMS) helps you create and control cryptographic keys to help protect your data. AWS KMS integrates with most other AWS services that can encrypt your data. For a complete list, see AWS services integrated with AWS KMS. AWS KMS also integrates with AWS CloudTrail to log use of your KMS keys for auditing, regulatory, and compliance needs.

KMS keys are the primary resource in AWS KMS, and they are logical representations of a cryptographic key. There are three primary types of KMS keys:

- Customer managed keys are KMS keys that you create.
- AWS managed keys are KMS keys that AWS services create in your account, on your behalf.
- AWS owned keys are KMS keys that an AWS service owns and manages, for use in multiple AWS accounts.

For more information about these key types, see Customer keys and AWS keys.

In the AWS Cloud, policies are used to control who can access resources and services. For example, in AWS Identity and Access Management (IAM), *identity-based policies* define permissions for users, user groups, or roles, and *resource-based policies* attach to a resource, such as an S3 bucket, and define which principals are allowed access, supported actions, and any other conditions that must be met. Similar to IAM policies, AWS KMS uses key policies to control access to a KMS key. Each KMS key must have a key policy, and each key can have only one key policy. Note the following when defining policies that allow or deny access to KMS keys:

- You can control the key policy for customer managed keys, but you can't directly control the key policy for AWS managed keys or for AWS owned keys.

- Key policies allow for granting granular access to AWS KMS API calls within an AWS account. Unless the key policy explicitly allows it, you cannot use IAM policies to allow access to a KMS key. Without permission from the key policy, IAM policies that allow permissions have no effect. For more information, see Allow IAM policies to allow access to the KMS key.
- You can use an IAM policy to deny access to a customer managed key without corresponding permission from the key policy.
- When designing key policies and IAM policies for multi-Region keys, consider the following:
  - Key policies are not shared properties of multi-Region keys and are not copied or synchronized among related multi-Region keys.
  - When a multi-Region key is created using the `CreateKey` and `ReplicateKey` actions, the default key policy is applied unless a key policy is specified in the request.
  - You can implement condition keys, such as aws:RequestedRegion, to limit permissions to a particular AWS Region.
  - You can use grants to allow permissions to a multi-Region primary key or replica key. However, a single grant cannot be used to allow permissions to multiple KMS keys, even if they are related multi-Region keys.

When using AWS KMS and creating key policies, consider the following encryption best practices and other security best practices:

- Adhere to the recommendations in the following resources for AWS KMS best practices:
  - Best practices for AWS KMS grants (AWS KMS documentation)
  - Best practices for IAM policies (AWS KMS documentation)
- In accordance with the separation of duties best practice, maintain separate identities for those who administer keys and those who use them:
  - Administrator roles that create and delete keys should not have the ability to use the key.
  - Some services may only need to encrypt data and should not be granted the ability to decrypt the data using the key.
- Key policies should always follow a model of least privilege. Do not use `kms:*` for actions in IAM or key policies because this gives the principal permissions to both administer and use the key.
- Limit the use of customer managed keys to specific AWS services by using the kms:ViaService condition key within the key policy.
- If you have a choice between key types, customer managed keys are preferred because they provide the most granular control options, including the following:

  - Managing authentication and access control
  - Enabling and disabling keys
  - Rotating AWS KMS keys
  - Tagging keys
  - Creating aliases
  - Deleting AWS KMS keys
- AWS KMS administrative and modification permissions must be explicitly denied to unapproved principals and AWS KMS modification permissions should not exist in an allow statement for any unauthorized principals. For more information, see Actions, resources, and condition keys for AWS Key Management Service.
- In order to detect unauthorized usage of KMS keys, in AWS Config, implement the iam-customer-policy-blocked-kms-actions and iam-inline-policy-blocked-kms-actions rules. This prevents principals from using the AWS KMS decryption actions on all resources.
- Implement service control policies (SCPs) in AWS Organizations to prevent unauthorized users or roles from deleting KMS keys, either directly as a command or through the console. For more information, see Using SCPs as preventative controls (AWS blog post).
- Log AWS KMS API calls in a CloudTrail log. This records the relevant event attributes, such as what requests were made, the source IP address from which the request was made, and who made the request. For more information, see Logging AWS KMS API calls with AWS CloudTrail.
- If you use encryption context, it shouldn't contain any sensitive information. CloudTrail stores the encryption context in plaintext JSON files, which can be viewed by anyone with access to the S3 bucket containing the information.
- When monitoring usage of customer managed keys, configure events to notify you if specific actions are detected, such as key creation, updates to customer managed key policies, or import of key material are detected. It's also recommended that you implement automated responses, such as an AWS Lambda function that disables the key or performs any other incident response actions as dictated by your organizational policies.
- Multi-Region keys are recommended for specific scenarios, such as compliance, disaster recovery, or backups. The security properties of multi-Region keys are significantly different than single-Region keys. The following recommendations apply when authorizing the creation, management, and use of multi-Region keys:
  - Allow principals to replicate a multi-Region key only into AWS Regions that require them.

- Give permission for multi-Region keys only to principals who need them and only for tasks that require them.

Encryption best practices for AWS Lambda

AWS Lambda is a compute service that helps you run code without needing to provision or manage servers. For securing your environment variables, you can use server-side encryption to protect your data at rest and client-side encryption to protect your data in transit.

Consider the following encryption best practices for this service:

- Lambda always provides server-side encryption at rest with an AWS KMS key. By default, Lambda uses an AWS managed key. We recommend you use a customer managed key because you have full control over the key, including management, rotation, and auditing.
- For data in transit that requires encryption, enable helpers, which ensures that environment variables are encrypted client-side for protection in transit by using the preferred KMS key. For more information, see *Security in transit* in Securing environment variables.
- Lambda function environment variables that hold sensitive or critical data should be encrypted in transit to help protect the data that is dynamically passed to the functions (usually access information) from unauthorized access.
- To prevent a user from viewing environment variables, add a statement to the user's permissions in the IAM policy or to the key policy that denies access to the default key, a customer managed key, or all keys. For more information, see Using AWS Lambda environment variables.

Encryption best practices for Amazon RDS

Amazon Relational Database Service (Amazon RDS) helps you set up, operate, and scale a relational database (DB) in the AWS Cloud. Data that is encrypted at rest includes the underlying storage for the DB instances, its automated backups, read replicas, and snapshots.

The following are the approaches you can use to encrypt data at rest in RDS DB instances:

- You can encrypt Amazon RDS DB instances with AWS KMS keys, either an AWS managed key or a customer managed key. For more information, see AWS Key Management Service in this guide.

- Amazon RDS for Oracle and Amazon RDS for SQL Server support encrypting DB instances with Transparent Data Encryption (TDE). For more information, see Oracle Transparent Data Encryption or Support for Transparent Data Encryption in SQL Server.

  You can use both TDE and KMS keys to encrypt DB instances. However, this can slightly affect the performance of your database, and you must manage these keys separately.

The following are the approaches you can use to encrypt data in transit to or from RDS DB instances:

- For an Amazon RDS DB instance running MariaDB, Microsoft SQL Server, MySQL, Oracle, or PostgreSQL, you can use SSL to encrypt the connection. For more information, see Using SSL/TLS to encrypt a connection to a DB instance.
- Amazon RDS for Oracle also supports Oracle native network encryption (NNE), which encrypts data as it moves to and from a DB instance. NNE and SSL encryption cannot be used simultaneously. For more information, see Oracle native network encryption.

Consider the following encryption best practices for this service:

- When connecting to Amazon RDS for SQL Server or Amazon RDS for PostgreSQL DB instances in order to process, store, or transmit data that requires encryption, use the RDS Transport Encryption feature to encrypt the connection. You can implement this by setting the `rds.force_ssl` parameter to `1` in the parameter group. For more information, see Working with parameter groups. Amazon RDS for Oracle uses Oracle database native network encryption.
- Customer managed keys for RDS DB instance encryption should be used solely for that purpose and not used with any other AWS services.
- Before encrypting an RDS DB instance, establish KMS key requirements. The key used by the instance cannot be changed later. For example, in your encryption policy, define use and management standards for AWS managed keys or customer managed keys, based on your business requirements.
- When authorizing access to a customer managed KMS key, follow the principle of least-privilege by using condition keys in IAM policies. For example, to allow a customer managed key to be used only for requests that originate in Amazon RDS, use the kms:ViaService condition key with the `rds.<region>.amazonaws.com` value. Additionally, you can use keys or values in the Amazon RDS encryption context as a condition for using the customer managed key.

- It is strongly recommended that you enable backups for encrypted RDS DB instances. Amazon RDS can lose access to the KMS key for a DB instance, such as when the KMS key isn't enabled or when RDS access to a KMS key is revoked. If this occurs, the encrypted DB instance goes into a recoverable state for seven days. If the DB instance does not regain access to the key after seven days, the database becomes terminally inaccessible and must be restored from a backup. For more information, see Encrypting a DB instance.
- If a read replica and its encrypted DB instance are in the same AWS Region, you must use the same KMS key to encrypt both.
- In AWS Config, implement the rds-storage-encrypted AWS managed rule to validate and enforce encryption for RDS DB instances and the rds-snapshots-encrypted rule to validate and enforce encryption for RDS database snapshots.
- Use AWS Security Hub CSPM to evaluate whether your Amazon RDS resources follow security best practices. For more information, see Security Hub CSPM controls for Amazon RDS.

Encryption best practices for AWS Secrets Manager

AWS Secrets Manager helps you replace hardcoded credentials in your code, including passwords, with an API call to Secrets Manager to retrieve the secret programmatically. Secrets Manager integrates with AWS KMS to encrypt every version of every secret value with a unique data key that is protected by an AWS KMS key. This integration protects stored secrets with encryption keys that never leave AWS KMS unencrypted. You can also define custom permissions on the KMS key to audit the operations that generate, encrypt, and decrypt the data keys that protect stored secrets. For more information, see Secret encryption and decryption in AWS Secrets Manager.

Consider the following encryption best practices for this service:

- For most cases, we recommend using the `aws/secretsmanager` AWS managed key to encrypt secrets. There is no cost for using it.
- To be able to access a secret from another account or to apply a key policy to the encryption key, use a customer managed key to encrypt the secret.
  - In the key policy, assign the value `secretsmanager.<region>.amazonaws.com` to the kms:ViaService condition key. This limits use of the key to only requests from Secrets Manager.
  - To further limit use of the key to only requests from Secrets Manager with the correct context, use keys or values in the Secrets Manager encryption context as a condition for using the KMS key by creating:

- A string condition operator in an IAM policy or key policy
- A grant constraint in a grant

Encryption best practices for Amazon S3

Amazon Simple Storage Service (Amazon S3) is a cloud-based object storage service that helps you store, protect, and retrieve any amount of data.

For server-side encryption in Amazon S3, there are three options:

- Server-side encryption with Amazon S3-managed encryption keys (SSE-S3)
- Server-side encryption with AWS Key Management Service (SSE-KMS)
- Server-side encryption with customer-provided encryption keys (SSE-C)

Amazon S3 applies server-side encryption with Amazon S3 managed keys (SSE-S3) as the base level of encryption for every bucket in Amazon S3. Starting January 5, 2023, all new object uploads to Amazon S3 are automatically encrypted at no additional cost and with no impact on performance. The automatic encryption status for S3 bucket default encryption configuration and for new object uploads is available in AWS CloudTrail logs, S3 Inventory, S3 Storage Lens, the Amazon S3 console, and as an additional Amazon S3 API response header in the AWS Command Line Interface (AWS CLI) and AWS SDKs. For more information, see Default encryption FAQ.

If server-side encryption is used to encrypt an object at the time of upload, add the `x-amz-server-side-encryption` header to the request to tell Amazon S3 to encrypt the object using SSE-S3, SSE-KMS, or SSE-C. The following are the possible values for the `x-amz-server-side-encryption` header:

- `AES256`, which tells Amazon S3 to use Amazon S3 managed keys.
- `aws:kms`, which tells Amazon S3 to use AWS KMS managed keys.
- Setting value as `True` or `False` for SSE-C

For more information, see *Defense-in-depth requirement 1: Data must be encrypted at rest and during transit* in How to Use Bucket Policies and Apply Defense-in-Depth to Help Secure Your Amazon S3 Data (AWS blog post).

For client-side encryption in Amazon S3, there are two options:

- A key stored in AWS KMS
- A key that is stored within the application

Consider the following encryption best practices for this service:

- In AWS Config, implement the s3-bucket-server-side-encryption-enabled AWS managed rule to validate and enforce S3 bucket encryption.
- Deploy an Amazon S3 bucket policy that validates that all objects being uploaded are encrypted using the `s3:x-amz-server-side-encryption` condition. For more information, see the example bucket policy in Protecting data using SSE-S3 and the instructions in Adding a bucket policy.
- Allow only encrypted connections over HTTPS (TLS) by using the `aws:SecureTransport` condition on S3 bucket policies. For more information, see What S3 bucket policy should I use to comply with the AWS Config rule s3-bucket-ssl-requests-only?
- In AWS Config, implement the s3-bucket-ssl-requests-only AWS managed rule to require requests to use SSL.
- Use a customer managed key when you need to grant cross-account access to Amazon S3 objects. Configure the key policy to allow access from another AWS account.

Encryption best practices for Amazon VPC

Amazon Virtual Private Cloud (Amazon VPC) helps you launch AWS resources into a virtual network that you've defined. This virtual network resembles a traditional network that you'd operate in your own data center, with the benefits of using the scalable infrastructure of AWS.

Consider the following encryption best practices for this service:

- Encrypt traffic between information assets and systems within the corporate network and VPCs by using one of the following:
  - AWS Site-to-Site VPN connections
  - A combination of AWS Site-to-Site VPN and AWS Direct Connect connections, which provides an IPsec-encrypted private connection
  - AWS Direct Connect connections that support MAC Security (MACsec) to encrypt data from corporate networks to the AWS Direct Connect location

- Use VPC endpoints in AWS PrivateLink to privately connect your VPCs to supported AWS services without using an internet gateway. You can use AWS Direct Connect or Site-to-Site VPN services to establish this connection. Traffic between your VPC and the other service does not leave the AWS network. For more information, see Access AWS services through AWS PrivateLink.
- Configure security group rules that allow traffic only from ports associated with secure protocols, such as HTTPS over TCP/443. Periodically audit security groups and their rules.

Resources

- Creating an enterprise encryption strategy for data at rest (AWS Prescriptive Guidance)
- Security best practices for AWS Key Management Service (AWS KMS documentation)
- How AWS services use AWS KMS (AWS KMS documentation)
- Security Pillar: Data protection (AWS Well-Architected Framework)

Document history

The following table describes significant changes to this guide. If you want to be notified about future updates, you can subscribe to an RSS feed.

Change	Description	Date
Cryptography updates	We updated the AWS approach to cryptography chapter.	February 19, 2026
Algorithm updates	We updated the Cryptographic algorithms section.	January 23, 2026
Algorithm and encryption in transit updates	We updated the About cryptographic algorithms section and the Encryption of data in transit section.	October 28, 2025
Algorithm updates	We add information about cryptography algorithms to the Cryptography algorithms and AWS services section.	June 18, 2025
Amazon EKS updates	We updated the encryption best practices for Amazon Elastic Kubernetes Service (Amazon EKS).	January 7, 2025
Secrets Manager updates	We updated the information and recommendations for AWS Secrets Manager.	September 9, 2024
AWS service updates	We updated the information and recommendations for Amazon EKS, AWS Encryption SDK, Amazon Relational Database Service (Amazon	September 4, 2024

	RDS), and Amazon Simple Storage Service (Amazon S3).	
Initial publication	—	December 2, 2022

AWS Prescriptive Guidance glossary

The following are commonly used terms in strategies, guides, and patterns provided by AWS Prescriptive Guidance. To suggest entries, please use the **Provide feedback** link at the end of the glossary.

Numbers

7 Rs

Seven common migration strategies for moving applications to the cloud. These strategies build upon the 5 Rs that Gartner identified in 2011 and consist of the following:

- Refactor/re-architect – Move an application and modify its architecture by taking full advantage of cloud-native features to improve agility, performance, and scalability. This typically involves porting the operating system and database. Example: Migrate your on-premises Oracle database to the Amazon Aurora PostgreSQL-Compatible Edition.
- Replatform (lift and reshape) – Move an application to the cloud, and introduce some level of optimization to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Amazon Relational Database Service (Amazon RDS) for Oracle in the AWS Cloud.
- Repurchase (drop and shop) – Switch to a different product, typically by moving from a traditional license to a SaaS model. Example: Migrate your customer relationship management (CRM) system to Salesforce.com.
- Rehost (lift and shift) – Move an application to the cloud without making any changes to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Oracle on an EC2 instance in the AWS Cloud.
- Relocate (hypervisor-level lift and shift) – Move infrastructure to the cloud without purchasing new hardware, rewriting applications, or modifying your existing operations. You migrate servers from an on-premises platform to a cloud service for the same platform. Example: Migrate a Microsoft Hyper-V application to AWS.
- Retain (revisit) – Keep applications in your source environment. These might include applications that require major refactoring, and you want to postpone that work until a later time, and legacy applications that you want to retain, because there's no business justification for migrating them.

- Retire – Decommission or remove applications that are no longer needed in your source environment.

A

ABAC

See attribute-based access control.

abstracted services

See managed services.

ACID

See atomicity, consistency, isolation, durability.

active-active migration

A database migration method in which the source and target databases are kept in sync (by using a bidirectional replication tool or dual write operations), and both databases handle transactions from connecting applications during migration. This method supports migration in small, controlled batches instead of requiring a one-time cutover. It's more flexible but requires more work than active-passive migration.

active-passive migration

A database migration method in which the source and target databases are kept in sync, but only the source database handles transactions from connecting applications while data is replicated to the target database. The target database doesn't accept any transactions during migration.

aggregate function

A SQL function that operates on a group of rows and calculates a single return value for the group. Examples of aggregate functions include `SUM` and `MAX`.

AI

See artificial intelligence.

AIOps

See artificial intelligence operations.

anonymization

The process of permanently deleting personal information in a dataset. Anonymization can help protect personal privacy. Anonymized data is no longer considered to be personal data.

anti-pattern

A frequently used solution for a recurring issue where the solution is counter-productive, ineffective, or less effective than an alternative.

application control

A security approach that allows the use of only approved applications in order to help protect a system from malware.

application portfolio

A collection of detailed information about each application used by an organization, including the cost to build and maintain the application, and its business value. This information is key to the portfolio discovery and analysis process and helps identify and prioritize the applications to be migrated, modernized, and optimized.

artificial intelligence (AI)

The field of computer science that is dedicated to using computing technologies to perform cognitive functions that are typically associated with humans, such as learning, solving problems, and recognizing patterns. For more information, see What is Artificial Intelligence?

artificial intelligence operations (AIOps)

The process of using machine learning techniques to solve operational problems, reduce operational incidents and human intervention, and increase service quality. For more information about how AIOps is used in the AWS migration strategy, see the operations integration guide.

asymmetric encryption

An encryption algorithm that uses a pair of keys, a public key for encryption and a private key for decryption. You can share the public key because it isn't used for decryption, but access to the private key should be highly restricted.

atomicity, consistency, isolation, durability (ACID)

A set of software properties that guarantee the data validity and operational reliability of a database, even in the case of errors, power failures, or other problems.

attribute-based access control (ABAC)

The practice of creating fine-grained permissions based on user attributes, such as department, job role, and team name. For more information, see ABAC for AWS in the AWS Identity and Access Management (IAM) documentation.

authoritative data source

A location where you store the primary version of data, which is considered to be the most reliable source of information. You can copy data from the authoritative data source to other locations for the purposes of processing or modifying the data, such as anonymizing, redacting, or pseudonymizing it.

Availability Zone

A distinct location within an AWS Region that is insulated from failures in other Availability Zones and provides inexpensive, low-latency network connectivity to other Availability Zones in the same Region.

AWS Cloud Adoption Framework (AWS CAF)

A framework of guidelines and best practices from AWS to help organizations develop an efficient and effective plan to move successfully to the cloud. AWS CAF organizes guidance into six focus areas called perspectives: business, people, governance, platform, security, and operations. The business, people, and governance perspectives focus on business skills and processes; the platform, security, and operations perspectives focus on technical skills and processes. For example, the people perspective targets stakeholders who handle human resources (HR), staffing functions, and people management. For this perspective, AWS CAF provides guidance for people development, training, and communications to help ready the organization for successful cloud adoption. For more information, see the AWS CAF website and the AWS CAF whitepaper.

AWS Workload Qualification Framework (AWS WQF)

A tool that evaluates database migration workloads, recommends migration strategies, and provides work estimates. AWS WQF is included with AWS Schema Conversion Tool (AWS SCT). It analyzes database schemas and code objects, application code, dependencies, and performance characteristics, and provides assessment reports.

B

bad bot

A bot that is intended to disrupt or cause harm to individuals or organizations.

BCP

See business continuity planning.

behavior graph

A unified, interactive view of resource behavior and interactions over time. You can use a behavior graph with Amazon Detective to examine failed logon attempts, suspicious API calls, and similar actions. For more information, see Data in a behavior graph in the Detective documentation.

big-endian system

A system that stores the most significant byte first. See also endianness.

binary classification

A process that predicts a binary outcome (one of two possible classes). For example, your ML model might need to predict problems such as “Is this email spam or not spam?" or "Is this product a book or a car?"

bloom filter

A probabilistic, memory-efficient data structure that is used to test whether an element is a member of a set.

blue/green deployment

A deployment strategy where you create two separate but identical environments. You run the current application version in one environment (blue) and the new application version in the other environment (green). This strategy helps you quickly roll back with minimal impact.

bot

A software application that runs automated tasks over the internet and simulates human activity or interaction. Some bots are useful or beneficial, such as web crawlers that index information on the internet. Some other bots, known as *bad bots*, are intended to disrupt or cause harm to individuals or organizations.

botnet

Networks of bots that are infected by malware and are under the control of a single party, known as a *bot herder* or *bot operator*. Botnets are the best-known mechanism to scale bots and their impact.

branch

A contained area of a code repository. The first branch created in a repository is the *main branch*. You can create a new branch from an existing branch, and you can then develop features or fix bugs in the new branch. A branch you create to build a feature is commonly referred to as a *feature branch*. When the feature is ready for release, you merge the feature branch back into the main branch. For more information, see About branches (GitHub documentation).

break-glass access

In exceptional circumstances and through an approved process, a quick means for a user to gain access to an AWS account that they don't typically have permissions to access. For more information, see the Implement break-glass procedures indicator in the AWS Well-Architected guidance.

brownfield strategy

The existing infrastructure in your environment. When adopting a brownfield strategy for a system architecture, you design the architecture around the constraints of the current systems and infrastructure. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

buffer cache

The memory area where the most frequently accessed data is stored.

business capability

What a business does to generate value (for example, sales, customer service, or marketing). Microservices architectures and development decisions can be driven by business capabilities. For more information, see the Organized around business capabilities section of the Running containerized microservices on AWS whitepaper.

business continuity planning (BCP)

A plan that addresses the potential impact of a disruptive event, such as a large-scale migration, on operations and enables a business to resume operations quickly.

C

CAF

See AWS Cloud Adoption Framework.

canary deployment

The slow and incremental release of a version to end users. When you are confident, you deploy the new version and replace the current version in its entirety.

CCoE

See Cloud Center of Excellence.

CDC

See change data capture.

change data capture (CDC)

The process of tracking changes to a data source, such as a database table, and recording metadata about the change. You can use CDC for various purposes, such as auditing or replicating changes in a target system to maintain synchronization.

chaos engineering

Intentionally introducing failures or disruptive events to test a system's resilience. You can use AWS Fault Injection Service (AWS FIS) to perform experiments that stress your AWS workloads and evaluate their response.

CI/CD

See continuous integration and continuous delivery.

classification

A categorization process that helps generate predictions. ML models for classification problems predict a discrete value. Discrete values are always distinct from one another. For example, a model might need to evaluate whether or not there is a car in an image.

client-side encryption

Encryption of data locally, before the target AWS service receives it.

Cloud Center of Excellence (CCoE)

A multi-disciplinary team that drives cloud adoption efforts across an organization, including developing cloud best practices, mobilizing resources, establishing migration timelines, and leading the organization through large-scale transformations. For more information, see the CCoE posts on the AWS Cloud Enterprise Strategy Blog.

cloud computing

The cloud technology that is typically used for remote data storage and IoT device management. Cloud computing is commonly connected to edge computing technology.

cloud operating model

In an IT organization, the operating model that is used to build, mature, and optimize one or more cloud environments. For more information, see Building your Cloud Operating Model.

cloud stages of adoption

The four phases that organizations typically go through when they migrate to the AWS Cloud:

- Project – Running a few cloud-related projects for proof of concept and learning purposes
- Foundation – Making foundational investments to scale your cloud adoption (e.g., creating a landing zone, defining a CCoE, establishing an operations model)
- Migration – Migrating individual applications
- Re-invention – Optimizing products and services, and innovating in the cloud

These stages were defined by Stephen Orban in the blog post The Journey Toward Cloud-First & the Stages of Adoption on the AWS Cloud Enterprise Strategy blog. For information about how they relate to the AWS migration strategy, see the migration readiness guide.

CMDB

See configuration management database.

code repository

A location where source code and other assets, such as documentation, samples, and scripts, are stored and updated through version control processes. Common cloud repositories include GitHub or Bitbucket Cloud. Each version of the code is called a *branch*. In a microservice structure, each repository is devoted to a single piece of functionality. A single CI/CD pipeline can use multiple repositories.

cold cache

A buffer cache that is empty, not well populated, or contains stale or irrelevant data. This affects performance because the database instance must read from the main memory or disk, which is slower than reading from the buffer cache.

cold data

Data that is rarely accessed and is typically historical. When querying this kind of data, slow queries are typically acceptable. Moving this data to lower-performing and less expensive storage tiers or classes can reduce costs.

computer vision (CV)

A field of AI that uses machine learning to analyze and extract information from visual formats such as digital images and videos. For example, Amazon SageMaker AI provides image processing algorithms for CV.

configuration drift

For a workload, a configuration change from the expected state. It might cause the workload to become noncompliant, and it's typically gradual and unintentional.

configuration management database (CMDB)

A repository that stores and manages information about a database and its IT environment, including both hardware and software components and their configurations. You typically use data from a CMDB in the portfolio discovery and analysis stage of migration.

conformance pack

A collection of AWS Config rules and remediation actions that you can assemble to customize your compliance and security checks. You can deploy a conformance pack as a single entity in an AWS account and Region, or across an organization, by using a YAML template. For more information, see Conformance packs in the AWS Config documentation.

continuous integration and continuous delivery (CI/CD)

The process of automating the source, build, test, staging, and production stages of the software release process. CI/CD is commonly described as a pipeline. CI/CD can help you automate processes, improve productivity, improve code quality, and deliver faster. For more information, see Benefits of continuous delivery. CD can also stand for *continuous deployment*. For more information, see Continuous Delivery vs. Continuous Deployment.

CV

See computer vision.

D

data at rest

Data that is stationary in your network, such as data that is in storage.

data classification

A process for identifying and categorizing the data in your network based on its criticality and sensitivity. It is a critical component of any cybersecurity risk management strategy because it helps you determine the appropriate protection and retention controls for the data. Data classification is a component of the security pillar in the AWS Well-Architected Framework. For more information, see Data classification.

data drift

A meaningful variation between the production data and the data that was used to train an ML model, or a meaningful change in the input data over time. Data drift can reduce the overall quality, accuracy, and fairness in ML model predictions.

data in transit

Data that is actively moving through your network, such as between network resources.

data mesh

An architectural framework that provides distributed, decentralized data ownership with centralized management and governance.

data minimization

The principle of collecting and processing only the data that is strictly necessary. Practicing data minimization in the AWS Cloud can reduce privacy risks, costs, and your analytics carbon footprint.

data perimeter

A set of preventive guardrails in your AWS environment that help make sure that only trusted identities are accessing trusted resources from expected networks. For more information, see Building a data perimeter on AWS.

data preprocessing

To transform raw data into a format that is easily parsed by your ML model. Preprocessing data can mean removing certain columns or rows and addressing missing, inconsistent, or duplicate values.

data provenance

The process of tracking the origin and history of data throughout its lifecycle, such as how the data was generated, transmitted, and stored.

data subject

An individual whose data is being collected and processed.

data warehouse

A data management system that supports business intelligence, such as analytics. Data warehouses commonly contain large amounts of historical data, and they are typically used for queries and analysis.

database definition language (DDL)

Statements or commands for creating or modifying the structure of tables and objects in a database.

database manipulation language (DML)

Statements or commands for modifying (inserting, updating, and deleting) information in a database.

DDL

See database definition language.

deep ensemble

To combine multiple deep learning models for prediction. You can use deep ensembles to obtain a more accurate prediction or for estimating uncertainty in predictions.

deep learning

An ML subfield that uses multiple layers of artificial neural networks to identify mapping between input data and target variables of interest.

defense-in-depth

An information security approach in which a series of security mechanisms and controls are thoughtfully layered throughout a computer network to protect the confidentiality, integrity, and availability of the network and the data within. When you adopt this strategy on AWS, you add multiple controls at different layers of the AWS Organizations structure to help secure resources. For example, a defense-in-depth approach might combine multi-factor authentication, network segmentation, and encryption.

delegated administrator

In AWS Organizations, a compatible service can register an AWS member account to administer the organization's accounts and manage permissions for that service. This account is called the *delegated administrator* for that service. For more information and a list of compatible services, see Services that work with AWS Organizations in the AWS Organizations documentation.

deployment

The process of making an application, new features, or code fixes available in the target environment. Deployment involves implementing changes in a code base and then building and running that code base in the application's environments.

development environment

See environment.

detective control

A security control that is designed to detect, log, and alert after an event has occurred. These controls are a second line of defense, alerting you to security events that bypassed the preventative controls in place. For more information, see Detective controls in *Implementing security controls on AWS*.

development value stream mapping (DVSM)

A process used to identify and prioritize constraints that adversely affect speed and quality in a software development lifecycle. DVSM extends the value stream mapping process originally designed for lean manufacturing practices. It focuses on the steps and teams required to create and move value through the software development process.

digital twin

A virtual representation of a real-world system, such as a building, factory, industrial equipment, or production line. Digital twins support predictive maintenance, remote monitoring, and production optimization.

dimension table

In a star schema, a smaller table that contains data attributes about quantitative data in a fact table. Dimension table attributes are typically text fields or discrete numbers that behave like text. These attributes are commonly used for query constraining, filtering, and result set labeling.

disaster

An event that prevents a workload or system from fulfilling its business objectives in its primary deployed location. These events can be natural disasters, technical failures, or the result of human actions, such as unintentional misconfiguration or a malware attack.

disaster recovery (DR)

The strategy and process you use to minimize downtime and data loss caused by a disaster. For more information, see Disaster Recovery of Workloads on AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML

See database manipulation language.

domain-driven design

An approach to developing a complex software system by connecting its components to evolving domains, or core business goals, that each component serves. This concept was introduced by Eric Evans in his book, *Domain-Driven Design: Tackling Complexity in the Heart of Software* (Boston: Addison-Wesley Professional, 2003). For information about how you can use domain-driven design with the strangler fig pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

DR

See disaster recovery.

drift detection

Tracking deviations from a baselined configuration. For example, you can use AWS CloudFormation to detect drift in system resources, or you can use AWS Control Tower to detect changes in your landing zone that might affect compliance with governance requirements.

DVSM

See development value stream mapping.

E

EDA

See exploratory data analysis.

EDI

See electronic data interchange.

edge computing

The technology that increases the computing power for smart devices at the edges of an IoT network. When compared with cloud computing, edge computing can reduce communication latency and improve response time.

electronic data interchange (EDI)

The automated exchange of business documents between organizations. For more information, see What is Electronic Data Interchange.

encryption

A computing process that transforms plaintext data, which is human-readable, into ciphertext.

encryption key

A cryptographic string of randomized bits that is generated by an encryption algorithm. Keys can vary in length, and each key is designed to be unpredictable and unique.

endianness

The order in which bytes are stored in computer memory. Big-endian systems store the most significant byte first. Little-endian systems store the least significant byte first.

endpoint

See service endpoint.

endpoint service

A service that you can host in a virtual private cloud (VPC) to share with other users. You can create an endpoint service with AWS PrivateLink and grant permissions to other AWS accounts or to AWS Identity and Access Management (IAM) principals. These accounts or principals can connect to your endpoint service privately by creating interface VPC endpoints. For more

information, see Create an endpoint service in the Amazon Virtual Private Cloud (Amazon VPC) documentation.

enterprise resource planning (ERP)

A system that automates and manages key business processes (such as accounting, MES, and project management) for an enterprise.

envelope encryption

The process of encrypting an encryption key with another encryption key. For more information, see Envelope encryption in the AWS Key Management Service (AWS KMS) documentation.

environment

An instance of a running application. The following are common types of environments in cloud computing:

- development environment – An instance of a running application that is available only to the core team responsible for maintaining the application. Development environments are used to test changes before promoting them to upper environments. This type of environment is sometimes referred to as a *test environment*.
- lower environments – All development environments for an application, such as those used for initial builds and tests.
- production environment – An instance of a running application that end users can access. In a CI/CD pipeline, the production environment is the last deployment environment.
- upper environments – All environments that can be accessed by users other than the core development team. This can include a production environment, preproduction environments, and environments for user acceptance testing.

epic

In agile methodologies, functional categories that help organize and prioritize your work. Epics provide a high-level description of requirements and implementation tasks. For example, AWS CAF security epics include identity and access management, detective controls, infrastructure security, data protection, and incident response. For more information about epics in the AWS migration strategy, see the program implementation guide.

ERP

See enterprise resource planning.

exploratory data analysis (EDA)

The process of analyzing a dataset to understand its main characteristics. You collect or aggregate data and then perform initial investigations to find patterns, detect anomalies, and check assumptions. EDA is performed by calculating summary statistics and creating data visualizations.

F

fact table

The central table in a star schema. It stores quantitative data about business operations. Typically, a fact table contains two types of columns: those that contain measures and those that contain a foreign key to a dimension table.

fail fast

A philosophy that uses frequent and incremental testing to reduce the development lifecycle. It is a critical part of an agile approach.

fault isolation boundary

In the AWS Cloud, a boundary such as an Availability Zone, AWS Region, control plane, or data plane that limits the effect of a failure and helps improve the resilience of workloads. For more information, see AWS Fault Isolation Boundaries.

feature branch

See branch.

features

The input data that you use to make a prediction. For example, in a manufacturing context, features could be images that are periodically captured from the manufacturing line.

feature importance

How significant a feature is for a model's predictions. This is usually expressed as a numerical score that can be calculated through various techniques, such as Shapley Additive Explanations (SHAP) and integrated gradients. For more information, see Machine learning model interpretability with AWS.

feature transformation

To optimize data for the ML process, including enriching data with additional sources, scaling values, or extracting multiple sets of information from a single data field. This enables the ML model to benefit from the data. For example, if you break down the “2021-05-27 00:15:37” date into “2021”, “May”, “Thu”, and “15”, you can help the learning algorithm learn nuanced patterns associated with different data components.

few-shot prompting

Providing an LLM with a small number of examples that demonstrate the task and desired output before asking it to perform a similar task. This technique is an application of in-context learning, where models learn from examples (*shots*) that are embedded in prompts. Few-shot prompting can be effective for tasks that require specific formatting, reasoning, or domain knowledge. See also zero-shot prompting.

FGAC

See fine-grained access control.

fine-grained access control (FGAC)

The use of multiple conditions to allow or deny an access request.

flash-cut migration

A database migration method that uses continuous data replication through change data capture to migrate data in the shortest time possible, instead of using a phased approach. The objective is to keep downtime to a minimum.

FM

See foundation model.

foundation model (FM)

A large deep-learning neural network that has been training on massive datasets of generalized and unlabeled data. FMs are capable of performing a wide variety of general tasks, such as understanding language, generating text and images, and conversing in natural language. For more information, see What are Foundation Models.

G

generative AI

A subset of AI models that have been trained on large amounts of data and that can use a simple text prompt to create new content and artifacts, such as images, videos, text, and audio. For more information, see What is Generative AI.

geo blocking

See geographic restrictions.

geographic restrictions (geo blocking)

In Amazon CloudFront, an option to prevent users in specific countries from accessing content distributions. You can use an allow list or block list to specify approved and banned countries. For more information, see Restricting the geographic distribution of your content in the CloudFront documentation.

Gitflow workflow

An approach in which lower and upper environments use different branches in a source code repository. The Gitflow workflow is considered legacy, and the trunk-based workflow is the modern, preferred approach.

golden image

A snapshot of a system or software that is used as a template to deploy new instances of that system or software. For example, in manufacturing, a golden image can be used to provision software on multiple devices and helps improve speed, scalability, and productivity in device manufacturing operations.

greenfield strategy

The absence of existing infrastructure in a new environment. When adopting a greenfield strategy for a system architecture, you can select all new technologies without the restriction of compatibility with existing infrastructure, also known as brownfield. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

guardrail

A high-level rule that helps govern resources, policies, and compliance across organizational units (OUs). *Preventive guardrails* enforce policies to ensure alignment to compliance standards. They are implemented by using service control policies and IAM permissions boundaries.

Detective guardrails detect policy violations and compliance issues, and generate alerts for remediation. They are implemented by using AWS Config, AWS Security Hub CSPM, Amazon GuardDuty, AWS Trusted Advisor, Amazon Inspector, and custom AWS Lambda checks.

H

HA

See high availability.

heterogeneous database migration

Migrating your source database to a target database that uses a different database engine (for example, Oracle to Amazon Aurora). Heterogeneous migration is typically part of a re-architecting effort, and converting the schema can be a complex task. AWS provides AWS SCT that helps with schema conversions.

high availability (HA)

The ability of a workload to operate continuously, without intervention, in the event of challenges or disasters. HA systems are designed to automatically fail over, consistently deliver high-quality performance, and handle different loads and failures with minimal performance impact.

historian modernization

An approach used to modernize and upgrade operational technology (OT) systems to better serve the needs of the manufacturing industry. A *historian* is a type of database that is used to collect and store data from various sources in a factory.

holdout data

A portion of historical, labeled data that is withheld from a dataset that is used to train a machine learning model. You can use holdout data to evaluate the model performance by comparing the model predictions against the holdout data.

homogeneous database migration

Migrating your source database to a target database that shares the same database engine (for example, Microsoft SQL Server to Amazon RDS for SQL Server). Homogeneous migration is typically part of a rehosting or replatforming effort. You can use native database utilities to migrate the schema.

hot data

Data that is frequently accessed, such as real-time data or recent translational data. This data typically requires a high-performance storage tier or class to provide fast query responses.

hotfix

An urgent fix for a critical issue in a production environment. Due to its urgency, a hotfix is usually made outside of the typical DevOps release workflow.

hypercare period

Immediately following cutover, the period of time when a migration team manages and monitors the migrated applications in the cloud in order to address any issues. Typically, this period is 1–4 days in length. At the end of the hypercare period, the migration team typically transfers responsibility for the applications to the cloud operations team.

I

IaC

See infrastructure as code.

identity-based policy

A policy attached to one or more IAM principals that defines their permissions within the AWS Cloud environment.

idle application

An application that has an average CPU and memory usage between 5 and 20 percent over a period of 90 days. In a migration project, it is common to retire these applications or retain them on premises.

IIoT

See industrial Internet of Things.

immutable infrastructure

A model that deploys new infrastructure for production workloads instead of updating, patching, or modifying the existing infrastructure. Immutable infrastructures are inherently more consistent, reliable, and predictable than mutable infrastructure. For more information, see the Deploy using immutable infrastructure best practice in the AWS Well-Architected Framework.

inbound (ingress) VPC

In an AWS multi-account architecture, a VPC that accepts, inspects, and routes network connections from outside an application. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

incremental migration

A cutover strategy in which you migrate your application in small parts instead of performing a single, full cutover. For example, you might move only a few microservices or users to the new system initially. After you verify that everything is working properly, you can incrementally move additional microservices or users until you can decommission your legacy system. This strategy reduces the risks associated with large migrations.

Industry 4.0

A term that was introduced by Klaus Schwab in 2016 to refer to the modernization of manufacturing processes through advances in connectivity, real-time data, automation, analytics, and AI/ML.

infrastructure

All of the resources and assets contained within an application's environment.

infrastructure as code (IaC)

The process of provisioning and managing an application's infrastructure through a set of configuration files. IaC is designed to help you centralize infrastructure management, standardize resources, and scale quickly so that new environments are repeatable, reliable, and consistent.

industrial Internet of Things (IIoT)

The use of internet-connected sensors and devices in the industrial sectors, such as manufacturing, energy, automotive, healthcare, life sciences, and agriculture. For more information, see Building an industrial Internet of Things (IIoT) digital transformation strategy.

inspection VPC

In an AWS multi-account architecture, a centralized VPC that manages inspections of network traffic between VPCs (in the same or different AWS Regions), the internet, and on-premises networks. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

Internet of Things (IoT)

The network of connected physical objects with embedded sensors or processors that communicate with other devices and systems through the internet or over a local communication network. For more information, see What is IoT?

interpretability

A characteristic of a machine learning model that describes the degree to which a human can understand how the model's predictions depend on its inputs. For more information, see Machine learning model interpretability with AWS.

IoT

See Internet of Things.

IT information library (ITIL)

A set of best practices for delivering IT services and aligning these services with business requirements. ITIL provides the foundation for ITSM.

IT service management (ITSM)

Activities associated with designing, implementing, managing, and supporting IT services for an organization. For information about integrating cloud operations with ITSM tools, see the operations integration guide.

ITIL

See IT information library.

ITSM

See IT service management.

L

label-based access control (LBAC)

An implementation of mandatory access control (MAC) where the users and the data itself are each explicitly assigned a security label value. The intersection between the user security label and data security label determines which rows and columns can be seen by the user.

landing zone

A landing zone is a well-architected, multi-account AWS environment that is scalable and secure. This is a starting point from which your organizations can quickly launch and deploy workloads and applications with confidence in their security and infrastructure environment. For more information about landing zones, see Setting up a secure and scalable multi-account AWS environment.

large language model (LLM)

A deep learning AI model that is pretrained on a vast amount of data. An LLM can perform multiple tasks, such as answering questions, summarizing documents, translating text into other languages, and completing sentences. For more information, see What are LLMs.

large migration

A migration of 300 or more servers.

LBAC

See label-based access control.

least privilege

The security best practice of granting the minimum permissions required to perform a task. For more information, see Apply least-privilege permissions in the IAM documentation.

lift and shift

See 7 Rs.

little-endian system

A system that stores the least significant byte first. See also endianness.

LLM

See large language model.

lower environments

See environment.

M

machine learning (ML)

A type of artificial intelligence that uses algorithms and techniques for pattern recognition and learning. ML analyzes and learns from recorded data, such as Internet of Things (IoT) data, to generate a statistical model based on patterns. For more information, see Machine Learning.

main branch

See branch.

malware

Software that is designed to compromise computer security or privacy. Malware might disrupt computer systems, leak sensitive information, or gain unauthorized access. Examples of malware include viruses, worms, ransomware, Trojan horses, spyware, and keyloggers.

managed services

AWS services for which AWS operates the infrastructure layer, the operating system, and platforms, and you access the endpoints to store and retrieve data. Amazon Simple Storage Service (Amazon S3) and Amazon DynamoDB are examples of managed services. These are also known as *abstracted services*.

manufacturing execution system (MES)

A software system for tracking, monitoring, documenting, and controlling production processes that convert raw materials to finished products on the shop floor.

MAP

See Migration Acceleration Program.

mechanism

A complete process in which you create a tool, drive adoption of the tool, and then inspect the results in order to make adjustments. A mechanism is a cycle that reinforces and improves itself as it operates. For more information, see Building mechanisms in the AWS Well-Architected Framework.

member account

All AWS accounts other than the management account that are part of an organization in AWS Organizations. An account can be a member of only one organization at a time.

MES

See manufacturing execution system.

Message Queuing Telemetry Transport (MQTT)

A lightweight, machine-to-machine (M2M) communication protocol, based on the publish/subscribe pattern, for resource-constrained IoT devices.

microservice

A small, independent service that communicates over well-defined APIs and is typically owned by small, self-contained teams. For example, an insurance system might include microservices that map to business capabilities, such as sales or marketing, or subdomains, such as purchasing, claims, or analytics. The benefits of microservices include agility, flexible scaling, easy deployment, reusable code, and resilience. For more information, see Integrating microservices by using AWS serverless services.

microservices architecture

An approach to building an application with independent components that run each application process as a microservice. These microservices communicate through a well-defined interface by using lightweight APIs. Each microservice in this architecture can be updated, deployed, and scaled to meet demand for specific functions of an application. For more information, see Implementing microservices on AWS.

Migration Acceleration Program (MAP)

An AWS program that provides consulting support, training, and services to help organizations build a strong operational foundation for moving to the cloud, and to help offset the initial cost of migrations. MAP includes a migration methodology for executing legacy migrations in a methodical way and a set of tools to automate and accelerate common migration scenarios.

migration at scale

The process of moving the majority of the application portfolio to the cloud in waves, with more applications moved at a faster rate in each wave. This phase uses the best practices and lessons learned from the earlier phases to implement a *migration factory* of teams, tools, and processes to streamline the migration of workloads through automation and agile delivery. This is the third phase of the AWS migration strategy.

migration factory

Cross-functional teams that streamline the migration of workloads through automated, agile approaches. Migration factory teams typically include operations, business analysts and owners,

migration engineers, developers, and DevOps professionals working in sprints. Between 20 and 50 percent of an enterprise application portfolio consists of repeated patterns that can be optimized by a factory approach. For more information, see the discussion of migration factories and the Cloud Migration Factory guide in this content set.

migration metadata

The information about the application and server that is needed to complete the migration. Each migration pattern requires a different set of migration metadata. Examples of migration metadata include the target subnet, security group, and AWS account.

migration pattern

A repeatable migration task that details the migration strategy, the migration destination, and the migration application or service used. Example: Rehost migration to Amazon EC2 with AWS Application Migration Service.

Migration Portfolio Assessment (MPA)

An online tool that provides information for validating the business case for migrating to the AWS Cloud. MPA provides detailed portfolio assessment (server right-sizing, pricing, TCO comparisons, migration cost analysis) as well as migration planning (application data analysis and data collection, application grouping, migration prioritization, and wave planning). The MPA tool (requires login) is available free of charge to all AWS consultants and APN Partner consultants.

Migration Readiness Assessment (MRA)

The process of gaining insights about an organization's cloud readiness status, identifying strengths and weaknesses, and building an action plan to close identified gaps, using the AWS CAF. For more information, see the migration readiness guide. MRA is the first phase of the AWS migration strategy.

migration strategy

The approach used to migrate a workload to the AWS Cloud. For more information, see the 7 Rs entry in this glossary and see Mobilize your organization to accelerate large-scale migrations.

ML

See machine learning.

modernization

Transforming an outdated (legacy or monolithic) application and its infrastructure into an agile, elastic, and highly available system in the cloud to reduce costs, gain efficiencies, and take advantage of innovations. For more information, see Strategy for modernizing applications in the AWS Cloud.

modernization readiness assessment

An evaluation that helps determine the modernization readiness of an organization's applications; identifies benefits, risks, and dependencies; and determines how well the organization can support the future state of those applications. The outcome of the assessment is a blueprint of the target architecture, a roadmap that details development phases and milestones for the modernization process, and an action plan for addressing identified gaps. For more information, see Evaluating modernization readiness for applications in the AWS Cloud.

monolithic applications (monoliths)

Applications that run as a single service with tightly coupled processes. Monolithic applications have several drawbacks. If one application feature experiences a spike in demand, the entire architecture must be scaled. Adding or improving a monolithic application's features also becomes more complex when the code base grows. To address these issues, you can use a microservices architecture. For more information, see Decomposing monoliths into microservices.

MPA

See Migration Portfolio Assessment.

MQTT

See Message Queuing Telemetry Transport.

multiclass classification

A process that helps generate predictions for multiple classes (predicting one of more than two outcomes). For example, an ML model might ask "Is this product a book, car, or phone?" or "Which product category is most interesting to this customer?"

mutable infrastructure

A model that updates and modifies the existing infrastructure for production workloads. For improved consistency, reliability, and predictability, the AWS Well-Architected Framework recommends the use of immutable infrastructure as a best practice.

O

OAC

See origin access control.

OAI

See origin access identity.

OCM

See organizational change management.

offline migration

A migration method in which the source workload is taken down during the migration process. This method involves extended downtime and is typically used for small, non-critical workloads.

OI

See operations integration.

OLA

See operational-level agreement.

online migration

A migration method in which the source workload is copied to the target system without being taken offline. Applications that are connected to the workload can continue to function during the migration. This method involves zero to minimal downtime and is typically used for critical production workloads.

OPC-UA

See Open Process Communications - Unified Architecture.

Open Process Communications - Unified Architecture (OPC-UA)

A machine-to-machine (M2M) communication protocol for industrial automation. OPC-UA provides an interoperability standard with data encryption, authentication, and authorization schemes.

operational-level agreement (OLA)

An agreement that clarifies what functional IT groups promise to deliver to each other, to support a service-level agreement (SLA).

operational readiness review (ORR)

A checklist of questions and associated best practices that help you understand, evaluate, prevent, or reduce the scope of incidents and possible failures. For more information, see Operational Readiness Reviews (ORR) in the AWS Well-Architected Framework.

operational technology (OT)

Hardware and software systems that work with the physical environment to control industrial operations, equipment, and infrastructure. In manufacturing, the integration of OT and information technology (IT) systems is a key focus for Industry 4.0 transformations.

operations integration (OI)

The process of modernizing operations in the cloud, which involves readiness planning, automation, and integration. For more information, see the operations integration guide.

organization trail

A trail that's created by AWS CloudTrail that logs all events for all AWS accounts in an organization in AWS Organizations. This trail is created in each AWS account that's part of the organization and tracks the activity in each account. For more information, see Creating a trail for an organization in the CloudTrail documentation.

organizational change management (OCM)

A framework for managing major, disruptive business transformations from a people, culture, and leadership perspective. OCM helps organizations prepare for, and transition to, new systems and strategies by accelerating change adoption, addressing transitional issues, and driving cultural and organizational changes. In the AWS migration strategy, this framework is called *people acceleration*, because of the speed of change required in cloud adoption projects. For more information, see the OCM guide.

origin access control (OAC)

In CloudFront, an enhanced option for restricting access to secure your Amazon Simple Storage Service (Amazon S3) content. OAC supports all S3 buckets in all AWS Regions, server-side encryption with AWS KMS (SSE-KMS), and dynamic PUT and DELETE requests to the S3 bucket.

origin access identity (OAI)

In CloudFront, an option for restricting access to secure your Amazon S3 content. When you use OAI, CloudFront creates a principal that Amazon S3 can authenticate with. Authenticated principals can access content in an S3 bucket only through a specific CloudFront distribution. See also OAC, which provides more granular and enhanced access control.

ORR

See operational readiness review.

OT

See operational technology.

outbound (egress) VPC

In an AWS multi-account architecture, a VPC that handles network connections that are initiated from within an application. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

P

permissions boundary

An IAM management policy that is attached to IAM principals to set the maximum permissions that the user or role can have. For more information, see Permissions boundaries in the IAM documentation.

personally identifiable information (PII)

Information that, when viewed directly or paired with other related data, can be used to reasonably infer the identity of an individual. Examples of PII include names, addresses, and contact information.

PII

See personally identifiable information.

playbook

A set of predefined steps that capture the work associated with migrations, such as delivering core operations functions in the cloud. A playbook can take the form of scripts, automated runbooks, or a summary of processes or steps required to operate your modernized environment.

PLC

See programmable logic controller.

PLM

See product lifecycle management.

policy

An object that can define permissions (see identity-based policy), specify access conditions (see resource-based policy), or define the maximum permissions for all accounts in an organization in AWS Organizations (see service control policy).

polyglot persistence

Independently choosing a microservice's data storage technology based on data access patterns and other requirements. If your microservices have the same data storage technology, they can encounter implementation challenges or experience poor performance. Microservices are more easily implemented and achieve better performance and scalability if they use the data store best adapted to their requirements.

portfolio assessment

A process of discovering, analyzing, and prioritizing the application portfolio in order to plan the migration. For more information, see Evaluating migration readiness.

predicate

A query condition that returns `true` or `false`, commonly located in a `WHERE` clause.

predicate pushdown

A database query optimization technique that filters the data in the query before transfer. This reduces the amount of data that must be retrieved and processed from the relational database, and it improves query performance.

preventative control

A security control that is designed to prevent an event from occurring. These controls are a first line of defense to help prevent unauthorized access or unwanted changes to your network. For more information, see Preventative controls in *Implementing security controls on AWS*.

principal

An entity in AWS that can perform actions and access resources. This entity is typically a root user for an AWS account, an IAM role, or a user. For more information, see *Principal* in Roles terms and concepts in the IAM documentation.

privacy by design

A system engineering approach that takes privacy into account through the whole development process.

private hosted zones

A container that holds information about how you want Amazon Route 53 to respond to DNS queries for a domain and its subdomains within one or more VPCs. For more information, see Working with private hosted zones in the Route 53 documentation.

proactive control

A security control designed to prevent the deployment of noncompliant resources. These controls scan resources before they are provisioned. If the resource is not compliant with the control, then it isn't provisioned. For more information, see the Controls reference guide in the AWS Control Tower documentation and see Proactive controls in *Implementing security controls on AWS*.

product lifecycle management (PLM)

The management of data and processes for a product throughout its entire lifecycle, from design, development, and launch, through growth and maturity, to decline and removal.

production environment

See environment.

programmable logic controller (PLC)

In manufacturing, a highly reliable, adaptable computer that monitors machines and automates manufacturing processes.

prompt chaining

Using the output of one LLM prompt as the input for the next prompt to generate better responses. This technique is used to break down a complex task into subtasks, or to iteratively refine or expand a preliminary response. It helps improve the accuracy and relevance of a model's responses and allows for more granular, personalized results.

pseudonymization

The process of replacing personal identifiers in a dataset with placeholder values. Pseudonymization can help protect personal privacy. Pseudonymized data is still considered to be personal data.

publish/subscribe (pub/sub)

A pattern that enables asynchronous communications among microservices to improve scalability and responsiveness. For example, in a microservices-based MES, a microservice can publish event messages to a channel that other microservices can subscribe to. The system can add new microservices without changing the publishing service.

Q

query plan

A series of steps, like instructions, that are used to access the data in a SQL relational database system.

query plan regression

When a database service optimizer chooses a less optimal plan than it did before a given change to the database environment. This can be caused by changes to statistics, constraints, environment settings, query parameter bindings, and updates to the database engine.

R

RACI matrix

See responsible, accountable, consulted, informed (RACI).

RAG

See Retrieval Augmented Generation.

ransomware

A malicious software that is designed to block access to a computer system or data until a payment is made.

RASCI matrix

See responsible, accountable, consulted, informed (RACI).

RCAC

See row and column access control.

read replica

A copy of a database that's used for read-only purposes. You can route queries to the read replica to reduce the load on your primary database.

re-architect

See 7 Rs.

recovery point objective (RPO)

The maximum acceptable amount of time since the last data recovery point. This determines what is considered an acceptable loss of data between the last recovery point and the interruption of service.

recovery time objective (RTO)

The maximum acceptable delay between the interruption of service and restoration of service.

refactor

See 7 Rs.

Region

A collection of AWS resources in a geographic area. Each AWS Region is isolated and independent of the others to provide fault tolerance, stability, and resilience. For more information, see Specify which AWS Regions your account can use.

regression

An ML technique that predicts a numeric value. For example, to solve the problem of "What price will this house sell for?" an ML model could use a linear regression model to predict a house's sale price based on known facts about the house (for example, the square footage).

rehost

See 7 Rs.

release

In a deployment process, the act of promoting changes to a production environment.

relocate

See 7 Rs.

replatform

See 7 Rs.

repurchase

See 7 Rs.

resiliency

An application's ability to resist or recover from disruptions. High availability and disaster recovery are common considerations when planning for resiliency in the AWS Cloud. For more information, see AWS Cloud Resilience.

resource-based policy

A policy attached to a resource, such as an Amazon S3 bucket, an endpoint, or an encryption key. This type of policy specifies which principals are allowed access, supported actions, and any other conditions that must be met.

responsible, accountable, consulted, informed (RACI) matrix

A matrix that defines the roles and responsibilities for all parties involved in migration activities and cloud operations. The matrix name is derived from the responsibility types defined in the matrix: responsible (R), accountable (A), consulted (C), and informed (I). The support (S) type is optional. If you include support, the matrix is called a *RASCI matrix*, and if you exclude it, it's called a *RACI matrix*.

responsive control

A security control that is designed to drive remediation of adverse events or deviations from your security baseline. For more information, see Responsive controls in *Implementing security controls on AWS*.

retain

See 7 Rs.

retire

See 7 Rs.

Retrieval Augmented Generation (RAG)

A generative AI technology in which an LLM references an authoritative data source that is outside of its training data sources before generating a response. For example, a RAG model might perform a semantic search of an organization's knowledge base or custom data. For more information, see What is RAG.

rotation

The process of periodically updating a secret to make it more difficult for an attacker to access the credentials.

row and column access control (RCAC)

The use of basic, flexible SQL expressions that have defined access rules. RCAC consists of row permissions and column masks.

RPO

See recovery point objective.

RTO

See recovery time objective.

runbook

A set of manual or automated procedures required to perform a specific task. These are typically built to streamline repetitive operations or procedures with high error rates.

S

SAML 2.0

An open standard that many identity providers (IdPs) use. This feature enables federated single sign-on (SSO), so users can log into the AWS Management Console or call the AWS API operations without you having to create user in IAM for everyone in your organization. For more information about SAML 2.0-based federation, see About SAML 2.0-based federation in the IAM documentation.

SCADA

See supervisory control and data acquisition.

SCP

See service control policy.

secret

In AWS Secrets Manager, confidential or restricted information, such as a password or user credentials, that you store in encrypted form. It consists of the secret value and its metadata.

The secret value can be binary, a single string, or multiple strings. For more information, see What's in a Secrets Manager secret? in the Secrets Manager documentation.

security by design

A system engineering approach that takes security into account through the whole development process.

security control

A technical or administrative guardrail that prevents, detects, or reduces the ability of a threat actor to exploit a security vulnerability. There are four primary types of security controls: preventative, detective, responsive, and proactive.

security hardening

The process of reducing the attack surface to make it more resistant to attacks. This can include actions such as removing resources that are no longer needed, implementing the security best practice of granting least privilege, or deactivating unnecessary features in configuration files.

security information and event management (SIEM) system

Tools and services that combine security information management (SIM) and security event management (SEM) systems. A SIEM system collects, monitors, and analyzes data from servers, networks, devices, and other sources to detect threats and security breaches, and to generate alerts.

security response automation

A predefined and programmed action that is designed to automatically respond to or remediate a security event. These automations serve as detective or responsive security controls that help you implement AWS security best practices. Examples of automated response actions include modifying a VPC security group, patching an Amazon EC2 instance, or rotating credentials.

server-side encryption

Encryption of data at its destination, by the AWS service that receives it.

service control policy (SCP)

A policy that provides centralized control over permissions for all accounts in an organization in AWS Organizations. SCPs define guardrails or set limits on actions that an administrator can delegate to users or roles. You can use SCPs as allow lists or deny lists, to specify which services or actions are permitted or prohibited. For more information, see Service control policies in the AWS Organizations documentation.

service endpoint

The URL of the entry point for an AWS service. You can use the endpoint to connect programmatically to the target service. For more information, see AWS service endpoints in *AWS General Reference*.

service-level agreement (SLA)

An agreement that clarifies what an IT team promises to deliver to their customers, such as service uptime and performance.

service-level indicator (SLI)

A measurement of a performance aspect of a service, such as its error rate, availability, or throughput.

service-level objective (SLO)

A target metric that represents the health of a service, as measured by a service-level indicator.

shared responsibility model

A model describing the responsibility you share with AWS for cloud security and compliance. AWS is responsible for security *of* the cloud, whereas you are responsible for security *in* the cloud. For more information, see Shared responsibility model.

SIEM

See security information and event management system.

single point of failure (SPOF)

A failure in a single, critical component of an application that can disrupt the system.

SLA

See service-level agreement.

SLI

See service-level indicator.

SLO

See service-level objective.

split-and-seed model

A pattern for scaling and accelerating modernization projects. As new features and product releases are defined, the core team splits up to create new product teams. This helps scale your

organization's capabilities and services, improves developer productivity, and supports rapid innovation. For more information, see Phased approach to modernizing applications in the AWS Cloud.

SPOF

See single point of failure.

star schema

A database organizational structure that uses one large fact table to store transactional or measured data and uses one or more smaller dimensional tables to store data attributes. This structure is designed for use in a data warehouse or for business intelligence purposes.

strangler fig pattern

An approach to modernizing monolithic systems by incrementally rewriting and replacing system functionality until the legacy system can be decommissioned. This pattern uses the analogy of a fig vine that grows into an established tree and eventually overcomes and replaces its host. The pattern was introduced by Martin Fowler as a way to manage risk when rewriting monolithic systems. For an example of how to apply this pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

subnet

A range of IP addresses in your VPC. A subnet must reside in a single Availability Zone.

supervisory control and data acquisition (SCADA)

In manufacturing, a system that uses hardware and software to monitor physical assets and production operations.

symmetric encryption

An encryption algorithm that uses the same key to encrypt and decrypt the data.

synthetic testing

Testing a system in a way that simulates user interactions to detect potential issues or to monitor performance. You can use Amazon CloudWatch Synthetics to create these tests.

system prompt

A technique for providing context, instructions, or guidelines to an LLM to direct its behavior. System prompts help set context and establish rules for interactions with users.

T

tags

Key-value pairs that act as metadata for organizing your AWS resources. Tags can help you manage, identify, organize, search for, and filter resources. For more information, see Tagging your AWS resources.

target variable

The value that you are trying to predict in supervised ML. This is also referred to as an *outcome variable*. For example, in a manufacturing setting the target variable could be a product defect.

task list

A tool that is used to track progress through a runbook. A task list contains an overview of the runbook and a list of general tasks to be completed. For each general task, it includes the estimated amount of time required, the owner, and the progress.

test environment

See environment.

training

To provide data for your ML model to learn from. The training data must contain the correct answer. The learning algorithm finds patterns in the training data that map the input data attributes to the target (the answer that you want to predict). It outputs an ML model that captures these patterns. You can then use the ML model to make predictions on new data for which you don't know the target.

transit gateway

A network transit hub that you can use to interconnect your VPCs and on-premises networks. For more information, see What is a transit gateway in the AWS Transit Gateway documentation.

trunk-based workflow

An approach in which developers build and test features locally in a feature branch and then merge those changes into the main branch. The main branch is then built to the development, preproduction, and production environments, sequentially.

trusted access

Granting permissions to a service that you specify to perform tasks in your organization in AWS Organizations and in its accounts on your behalf. The trusted service creates a service-linked role in each account, when that role is needed, to perform management tasks for you. For more information, see Using AWS Organizations with other AWS services in the AWS Organizations documentation.

tuning

To change aspects of your training process to improve the ML model's accuracy. For example, you can train the ML model by generating a labeling set, adding labels, and then repeating these steps several times under different settings to optimize the model.

two-pizza team

A small DevOps team that you can feed with two pizzas. A two-pizza team size ensures the best possible opportunity for collaboration in software development.

U

uncertainty

A concept that refers to imprecise, incomplete, or unknown information that can undermine the reliability of predictive ML models. There are two types of uncertainty: *Epistemic uncertainty* is caused by limited, incomplete data, whereas *aleatoric uncertainty* is caused by the noise and randomness inherent in the data.

undifferentiated tasks

Also known as *heavy lifting*, work that is necessary to create and operate an application but that doesn't provide direct value to the end user or provide competitive advantage. Examples of undifferentiated tasks include procurement, maintenance, and capacity planning.

upper environments

See environment.

V

vacuuming

A database maintenance operation that involves cleaning up after incremental updates to reclaim storage and improve performance.

version control

Processes and tools that track changes, such as changes to source code in a repository.

VPC peering

A connection between two VPCs that allows you to route traffic by using private IP addresses. For more information, see What is VPC peering in the Amazon VPC documentation.

vulnerability

A software or hardware flaw that compromises the security of the system.

W

warm cache

A buffer cache that contains current, relevant data that is frequently accessed. The database instance can read from the buffer cache, which is faster than reading from the main memory or disk.

warm data

Data that is infrequently accessed. When querying this kind of data, moderately slow queries are typically acceptable.

window function

A SQL function that performs a calculation on a group of rows that relate in some way to the current record. Window functions are useful for processing tasks, such as calculating a moving average or accessing the value of rows based on the relative position of the current row.

workload

A collection of resources and code that delivers business value, such as a customer-facing application or backend process.

workstream

Functional groups in a migration project that are responsible for a specific set of tasks. Each workstream is independent but supports the other workstreams in the project. For example, the portfolio workstream is responsible for prioritizing applications, wave planning, and collecting migration metadata. The portfolio workstream delivers these assets to the migration workstream, which then migrates the servers and applications.

WORM

See write once, read many.

WQF

See AWS Workload Qualification Framework.

write once, read many (WORM)

A storage model that writes data a single time and prevents the data from being deleted or modified. Authorized users can read the data as many times as needed, but they cannot change it. This data storage infrastructure is considered immutable.

Z

zero-day exploit

An attack, typically malware, that takes advantage of a zero-day vulnerability.

zero-day vulnerability

An unmitigated flaw or vulnerability in a production system. Threat actors can use this type of vulnerability to attack the system. Developers frequently become aware of the vulnerability as a result of the attack.

zero-shot prompting

Providing an LLM with instructions for performing a task but no examples (*shots*) that can help guide it. The LLM must use its pre-trained knowledge to handle the task. The effectiveness of zero-shot prompting depends on the complexity of the task and the quality of the prompt. See also few-shot prompting.

zombie application

An application that has an average CPU and memory usage below 5 percent. In a migration project, it is common to retire these applications.