Establishing guardrails and monitoring for presigned URLs

AWS Prescriptive Guidance

AWS Prescriptive Guidance: Establishing guardrails and monitoring for presigned URLs

Table of Contents

Establishing guardrails and monitoring for presigned URLs

Ryan Baker, Amazon Web Services (AWS)

August 2025 (document history)

Security is a critical concern for all companies and a key pillar in the AWS Well-Architected Framework. As a security engineer, you will want to implement administrative guardrails that are aligned with organizational control requirements. In the AWS Well-Architected Framework, guardrails define the boundaries that limit activity.

This guide provides background information and best practices for using presigned URLs, which are used with Amazon Simple Storage Service (Amazon S3) objects. Presigned URLs allow users or applications that have access to valid credentials to generate requests that are signed in advance and are accepted until a defined expiration time. A common use case for presigned URLs is to extend access to objects or resources by sharing these requests. Shared presigned requests are generated by systems or users that have the rights to perform a specific request, and can then be sent to other systems or users to extend the ability to perform that same request.

In this guide, you will learn:

- The concepts of presigned URLs
- Use cases for presigned URLs
- Recommended and optional guardrails
- Monitoring options
- Examples of how AWS services use presigned URLs

Intended audience

This guide is intended for architects and security engineers who are responsible for implementing security controls in the AWS Cloud.

Objectives

As a security engineer, you want to be aware of how solution builders are implementing security and the type of access your end users have. This guide covers one type of access, presigned URLs, which are often used with Amazon S3. Presigned URLs provide builders with options for efficiently bridging authentication mechanisms.

In Amazon S3, presigned URLs represent a unique category of requests. Security engineers can monitor and manage these requests to ensure that they are used only where appropriate and necessary. The objective of this guide is to help security engineers provide this type of high-level oversight.

After reading this guide, you should understand what a presigned URL is, when it is typically used, and the motivations for its use.

Prerequisites

If your company has not defined a security policy, control objectives, or standards, as described in the guide Implementing security controls on AWS, we recommend that you complete those governance tasks before proceeding with this guide.

Before you begin, you should also be familiar with recommended and optional best practices for control and monitoring. For more information, see:

- Service control policies (AWS Organizations documentation)
- Resource control policies (AWS Organizations documentation)
- Bucket policies for Amazon S3 (Amazon S3 documentation)
- Logging requests with server access logging (Amazon S3 documentation)
- Logging Amazon S3 API calls using AWS CloudTrail (Amazon S3 documentation)

Overview of presigned URLs

A presigned URL is a type of HTTP request that's recognized by the AWS Identity and Access Management (IAM) service. What differentiates this type of request from all other AWS requests is the X-Amz-Expires query parameter. As with other authenticated requests, presigned URL requests include a signature. For presigned URL requests, this signature is transmitted in `X-Amz-Signature`. The signature uses Signature Version 4 cryptographic operations to encode all other request parameters.

Notes

- Signature Version 2 is currently in the process of being deprecated, but it is still supported in some AWS Regions. This guide applies to Signature Version 4 signing.
- The receiving service could process unsigned headers, but support for that option is limited and targeted, in line with best practices. Unless otherwise noted, assume that all headers must be signed for a request to be accepted.

The `X-Amz-Expires` parameter allows a signature to be processed as valid with a greater deviation from the encoded date time. Other aspects of signature validity are still evaluated. The signing credentials, if temporary, must not be expired at the time the signature is processed. The signing credentials must be attached to an IAM principal who has sufficient authorization at the time of processing.

Presigned URLs are a subset of presigned requests

A presigned URL is not the only method to sign a request for a future time. Amazon S3 also supports POST requests, which are commonly presigned as well. A presigned POST signature allows uploads that comply with a signed policy and has an expiration date that's embedded in that policy.

Signatures for requests can be future dated, although this is uncommon. As long as the underlying credentials are valid, the signature algorithm doesn't prohibit future dating. However, these requests can't be successfully processed until their valid timing window, which makes future dating impractical for most use cases.

What does a presigned request allow?

A presigned request can only allow actions that are allowed by the credentials that were used to sign the request. If the credentials implicitly or explicitly deny the action that's specified by the presigned request, the presigned request is denied when it's sent. This applies to the following:

- Session policies that are associated with the credentials
- Identity-based policies that are associated with the principal who is associated with the credentials
- Resource policies that affect the session or principal
- Service control policies that affect the session or principal
- Resource control policies that affect the session or principal

Motivations for using presigned requests

As a security engineer, you should be aware of what motivates solution builders to use presigned URLs. Understanding what is necessary and what is optional will help you communicate with solution builders. Motivations might include the following:

- To support a non-IAM authentication mechanism while benefiting from scalability in Amazon S3. A core motivation is to communicate directly with Amazon S3 to benefit from the built-in scalability provided by this service. Without this direct communication, a solution would need to support the load from retransmitting bytes that are sent in **PutObject** and **GetObject** calls. Depending on total load, this requirement adds scaling challenges a solution builder might want to avoid.

  Other means of communicating directly with Amazon S3, such as using temporary credentials in AWS Security Token Service (AWS STS) or Signature Version 4 signatures outside URLs, might not be appropriate for your use case. Amazon S3 identifies users through AWS credentials, whereas presigned requests presume identification through mechanisms other than AWS credentials. Bridging this difference while maintaining the direct communication for data is achievable through presigned requests.
- To benefit from a browser's native understanding of URLs. URLs are understood by browsers, whereas AWS STS credentials and Signature Version 4 signatures are not. This is beneficial when integrating with browser-based solutions. Alternative solutions require more code, will use more memory for large files, and might be treated differently by extensions such as malware and virus scanners.

Comparison with temporary AWS STS credentials

Temporary credentials are similar to presigned requests. They both expire, allow scoping of access, and are commonly used to bridge non-IAM credentials to usage that requires AWS credentials.

You can tightly scope a temporary AWS STS credential to a single S3 object and action, but this can result in scaling challenges because AWS STS APIs have limits. (For more information, see the article How can I resolve API throttling or "Rate exceeded" errors for IAM and AWS STS on the AWS re:Post website.) In addition, each generated credential requires an AWS STS API call, which adds latency and a new dependency that might affect resilience. A temporary AWS STS credential also has a minimum expiration time of 15 minutes, whereas a presigned request can support shorter durations. (60 seconds is practical given the right conditions.)

Comparison with signature-only solutions

The only inherently secret component of a presigned request is its Signature Version 4 signature. If a client knows the other details of a request and is provided with a valid signature that matches those details, it can send a valid request. Without a valid signature, it cannot.

Presigned URLs and signature-only solutions are cryptographically similar. However, signature-only solutions have practical advantages such as the ability to use an HTTP header instead of a query string parameter to transmit the signature (see the Logging interactions and mitigations section). Administrators should also consider that query strings are more commonly treated as metadata, whereas headers are less commonly treated as such.

On the other hand, AWS SDKs provide less support for generating and using signatures directly. Building a signature-only solution requires more custom code. From a practical perspective, using libraries instead of custom code for security is a general best practice, so the code for signature-only solutions requires extra scrutiny.

Signature-only solutions do not use the `X-Amz-Expires` query string and provide no explicit validity period. IAM manages the implicit validity periods of signatures that don't have an explicit expiration time. Those implicit periods aren't published. They don't typically change, but they are managed with security in mind, so you shouldn't take a dependency on the validity periods. There's a tradeoff between having explicit control over the expiration date and having IAM manage the expiration.

As an administrator, you might prefer a signature-only solution. However, in a practical sense, you'll need to support solutions as built.

Identifying presigned requests

Identifying requests that used a presigned URL

Amazon S3 provides two built-in mechanisms for monitoring usage at a request level: Amazon S3 server access logs and AWS CloudTrail data events. Both mechanisms can identify presigned URL usage.

To filter logs for presigned URL usage, you can use the authentication type. For server access logs, examine the Authentication Type field, which is typically named authtype when it's defined in an Amazon Athena table. For CloudTrail, examine AuthenticationMethod in the `additionalEventData` field. In both cases, the field value for requests that use presigned URLs is `QueryString`, whereas `AuthHeader` is the value for most other requests.

`QueryString` usage isn't always associated with presigned URLs. To restrict your search to only presigned URL usage, find requests that contain the query string parameter `X-Amz-Expires`. For server access logs, examine Request-URI and look for requests that have an `X-Amz-Expires` parameter in the query string. For CloudTrail, examine the `requestParameters` element for an `X-Amz-Expires` element.

```
{"Records": [{…, "requestParameters": {…, "X-Amz-Expires": "300"}}, …]}
```

The following Athena query applies this filter:

```
SELECT * FROM {athena-table} WHERE
  authtype = 'QueryString' AND
  request_uri LIKE '%X-Amz-Expires=%';
```

For AWS CloudTrail Lake, the following query applies this filter:

```
SELECT * FROM {data-store-event-id} WHERE
  additionalEventData['AuthenticationMethod'] = 'QueryString' AND
  requestParameters['X-Amz-Expires'] IS NOT NULL
```

Identifying other types of presigned requests

The POST request also has a unique authentication type, `HtmlForm`, in Amazon S3 server access logs and CloudTrail. This authentication type is less common, so you might not find these requests in your environment.

The following Athena query applies the filter for `HtmlForm`:

```
SELECT * FROM {athena-table} WHERE
  authtype = 'HtmlForm';
```

For CloudTrail Lake, the following query applies the filter:

```
SELECT * FROM {data-store-event-id} WHERE
  additionalEventData['AuthenticationMethod'] = 'HtmlForm'
```

Identifying request patterns

You can find presigned requests by using the techniques discussed in the previous section. However, to make that data useful, you'll want to find patterns. The simple `TOP 10` results for your query might provide an insight, but if that's not enough, use the grouping options in the following table.

Grouping option	Server access logs	CloudTrail Lake	Description
User agent	`GROUP BY useragent`	`GROUP BY userAgent`	This grouping option helps you find the source and purpose of requests. The user agent is user supplied and not reliable as an authentication or authorization mechanism. However, it can reveal a lot if you're looking for patterns, because

Grouping option	Server access logs	CloudTrail Lake	Description
			most clients use a unique string that is at least partially human readable.
Requester	`GROUP BY requester`	`GROUP BY userIdentity['arn']`	This grouping option helps find IAM principals who signed requests. If your goal is to block these requests or to create an exception for existing requests, these queries provide enough information for that purpose. When you use roles in accordance with IAM best practices, the role has a clearly identified owner, and you can use that information to find out more.

Grouping option	Server access logs	CloudTrail Lake	Description
Source IP address	`GROUP BY remoteip`	`GROUP BY sourceIPAddress`	This option groups by the last network translation hop before reaching Amazon S3. • If the traffic passes through a NAT gateway, this will be the NAT Gateway address. • If the traffic passes through an internet gateway, this will be the public IP address that sent the traffic to the internet gateway. • If the traffic originates from outside AWS, this will be the public internet address that's associated with the origin. • If it passes over a gateway virtual private cloud (VPC) endpoint, this will be the IP address

Grouping option	Server access logs	CloudTrail Lake	Description
			of the instance in the VPC. • If it passes through a public virtual interface (VIF), this will be the on-premises IP of the requester or any intermediary such as a proxy server or firewall that exposes only its IP address. • If it passes through an interface VPC endpoint, this could be the IP address from an instance in the VPC. It could also be an IP address from another VPC or an on-premis es network. As with public VIFs, this could be any intermediary's IP address. This data is useful if your goal is to impose network

Grouping option	Server access logs	CloudTrail Lake	Description
			controls. You might have to combine this option with data such as `endpoint` (for server access logs) or `vpcEndpointId` (for CloudTrail Lake) to clarify the source, because different networks might duplicate private IP addresses.
S3 bucket name	`GROUP BY bucket_name`	`GROUP BY requestParameters['bucketName']`	This grouping option helps find buckets that received requests. This helps you identify the need for exceptions.

Best practices for using presigned requests

This section discusses best practices for using presigned requests that a security engineer should consider. The guidelines include:

- Foundational best practices, which are practices that every organization should follow.
- Additional guardrails, which are practices that you should consider, but might decide to implement partially or with exceptions. These are intended to provide additional control and defense in depth, but should be balanced against overall complexity.
- Logging interactions, which might result from devices or services that are part of your or your customer's responsibility in the shared responsibility model. This section includes precautions to limit the information that is accessible through logs.

Foundational best practices

General best practices that are effective controls for other AWS API requests apply to presigned requests as well. This section reviews two of the most relevant practices: least privilege and data perimeters. These practices create a depth of controls that other practices extend.

Apply the principle of least privilege

The first step in limiting the use of presigned requests is limiting access to Amazon S3 in general. A presigned URL cannot provide access to resources that weren't granted to the principal that generated the signature for the presigned URL. Nor can it provide access to a resource in a way that wasn't granted to that principal. As such, applying best practices to grant those principals least privilege is an effective guardrail.

The process of creating a presigned URL is an algorithmic operation that's based on a published standard (Signature Version 4) for signature generation. Therefore, it's not possible to place limits on the generation of presigned URLs. However, to be relevant, a presigned URL must be valid and provide access to resources, so the validity of a presigned URL is also an effective guardrail.

For more information about least privilege, see Grant least privilege access in the AWS Well-Architected Framework, Security pillar.

Implement a data perimeter

An extension of least privilege is to maintain a data perimeter that's consistent your organization's needs. Presigned URLs are compatible with data perimeters. As with other requests, a presigned URL request's validity is evaluated at request time. If the properties of the network, resource, role-session, and principal change, they are evaluated at the time and by using the method by which a request is received.

For example, let's say that a service that's running in an Amazon Elastic Kubernetes Service (Amazon EKS) container signs a request. The request is later sent from a user's personal computer system that's connected to the internet. In this case, the aws:SourceIp condition evaluates the visible public IP address of the request from the the user's personal system, not the IP address of the service in the Amazon EKS container.

Similarly, if the tags of the principal or resource change before the request is sent, the updated, not original, values will apply to the request through the aws:PrincipalTag/tag-key and aws:ResourceTag/tag-key conditions.

Additional guardrails

When presigned requests are used appropriately by solution builders and users, they provide a secure mechanism for giving users access to data. In addition, the ability to generate presigned requests doesn’t provide principals with access that they didn’t already have.

In that context, are additional controls necessary? The justification for additional controls isn't based on a need to deny access but to provide the ability to monitor, to approve usage and set boundaries, and to reduce risk from user errors. In this way you can help ensure that usage is appropriate and necessary.

The following guardrails assist you in this goal. Before you enable these controls, you might want to determine existing usage by identifying presigned requests. This identification helps you prepare for the guardrail's impact to existing usage or to plan exceptions where needed.

Guardrail for s3:signatureAge

One defining characteristic of presigned requests is that they describe an expiration time. The signature for the request contains a date. This date is transmitted as an `X-Amz-Date` query string parameter for presigned URLs, and as a Date or x-amz-date header for a presigned POST.

Amazon S3 provides a condition key, s3:signatureAge, which you can use to limit the maximum time between the signed date and the effective expiration of the request. This condition can never increase the validity period, but it can reduce it.

In the following policy, the `s3:signatureAge` condition key limits presigned requests to 15 minutes of validity. The following examples all use 15 minutes to limit validity to a similar timeframe as standard signing supports.

The second statement from the policy denies any Signature Version 2 access. This version of the signing protocol is being deprecated, but it is still supported in some AWS Regions. We recommend that you explicitly block it before it is fully deprecated.

You can apply the following policy as an AWS Organizations service control policy (SCP). Users can still use presigned requests and deploy solutions that depend on those requests, as long as the time between signature generation and usage is less than 15 minutes. Depending on the implementation, this limitation might have no impact, it might cause a solution to become unusable, or it might cause occasional failures that can be retried.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyPresignedOver15Minutes",
      "Effect": "Deny",
      "Action": "s3:*",
      "Resource": "*",
      "Condition": {
        "NumericGreaterThan": {
          "s3:signatureAge": "900000"
        }
      }
    },
    {
      "Sid": "DenySignatureVersion2",
      "Effect": "Deny",
      "Action": "s3:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
           "s3:signatureversion": "AWS"
        }
      }
    }
  ]
}
```

Exceptions

If a solution requires a longer time before expiration and is therefore affected by the preceding policy, we recommend that you provide a method to approve exceptions. To avoid enumerating exceptions in an SCP, use aws:PrincipalTag, as in the following policy, to manage exceptions in a scalable way. Other AWS examples, such as the AWS data perimeter policy examples, use this strategy.

If you implement an exception policy by using `aws:PrincipalTag`, you must control access to setting tags on principals. Tags of this type can come directly from principals and can be controlled by an SCP, as in this example of controlling which tag values can be set. A tag of this type can also come from session tags, which are set by an identity provider (IdP) or when using AWS STS. Controlling access to `aws:PrincipalTag` is a complex topic. However, an organization that has experience in using attribute-based access control (ABAC) will have the experience and controls to enable the appropriate use of `aws:PrincipalTag` for this use case.

In the following example, the `aws:PrincipalTag` condition creates an exception that allows any principal with the named tag (`long-presigned-allowed`) assigned and set to `true`. With this exception the restriction on signature age is not applied.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyPresignedOver15Minutes",
      "Effect": "Deny",
      "Action": "s3:*",
      "Resource": "*",
      "Condition": {
        "NumericGreaterThan": {
          "s3:signatureAge": "900000"
        },
        "StringNotEquals": {
          "aws:PrincipalTag/long-presigned-allowed": "true"
        }
      }
    }
  ]
}
```

Bucket policies

You can apply bucket policies to all or selected buckets by using a policy as in the following example. Unlike an SCP, a bucket policy also targets service principal usage. Appendix A doesn't document any expected service principal usage of presigned requests, but if you wanted to implement a control in order to prove that limit, the following policy would provide that control. Also, unlike an SCP, a bucket policy can apply to principals in your management account.

ABAC-based exceptions work in bucket policies in the same way as an SCP. A goal of a bucket policy might be to apply to principals outside the organization, so ABAC exceptions should be constrained to the principals for which ABAC controls apply.

In the following example, the `aws:PrincipalTag` condition in the first statement creates an exception for a principal with the named tag (`long-presigned-allowed`) assigned and set to `true`. With this exception the restriction on signature age is not applied. The second statement applies this restriction to all principals outside the AWS organization that owns the bucket. The scope of this second statement should match ABAC controls to set the named tag for principals.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyPresignedOver15MinWithExceptions",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::{bucket-name}/*",
      "Condition": {
        "NumericGreaterThan": {
          "s3:signatureAge": "900000"
        },
        "StringNotEquals": {
          "aws:PrincipalTag/long-presigned-allowed": "true"
        }
      }
    },
    {
      "Sid": "DenyPresignedOver15MinutesOutsideOrg",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::{bucket-name}/*",
      "Condition": {
        "NumericGreaterThan": {
          "s3:signatureAge": "900000"
        },
        "StringNotEquals": {
          "aws:PrincipalOrgID": "${aws:ResourceOrgID}"
        }
      }
    }
  ]
}
```

Resource control policies

You can apply a policy to buckets at scale by using resource control policies (RCPs). Like SCPs and unlike bucket policies, RCPs do not target service principal usage. RCPs affect non-service principals from any account, but they do not affect resources in the management account. For more information, see the AWS Organizations documentation.

As with bucket policies, if you use `aws:PrincipalTags` to create exceptions for principals, keep in mind the scope of ABAC controls on the tagging of principals.

The following RCP restricts presigned URL usage across all S3 buckets in an organization by limiting signature age to 15 minutes.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyPresignedOver15MinWithExceptions",
      "Effect": "Deny",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::*/*",
      "Condition": {
        "NumericGreaterThan": {
          "s3:signatureAge": "900000"
        },
        "StringNotEquals": {
          "aws:PrincipalTag/long-presigned-allowed": "true",
        }
      }
    },
    {
      "Sid": "DenyPresignedOver15MinutesOutsideOrg",
      "Effect": "Deny",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::*/*",
      "Condition": {
        "NumericGreaterThan": {
          "s3:signatureAge": "900000"
        },
        "StringNotEquals": {
          "aws:PrincipalOrgID": "${aws:ResourceOrgID}"
        }
      }
    }
  ]
}
```

Guardrail for s3:authType

Presigned URLs use query string authentication, and presigned POSTs always use POST authentication. Amazon S3 supports the denial of requests based on authentication type through the s3:authType condition key. `REST-QUERY-STRING` is the `s3:authType` value for query strings, and `POST` is the `s3:authType` value for POST.

You can apply the following policy as an SCP. The policy uses `s3:authType` to allow only header-based authentication. It also configures a method to provide exceptions to individual users or roles.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyNonHeaderAuth",
      "Effect": "Deny",
      "Action": "s3:*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "s3:authType": "REST-HEADER",
          "aws:PrincipalTag/non-header-auth-allowed": "true"
        }
      }
    }
  ]
}
```

Denying requests based on authentication type affects any solution or feature that uses the denied authentication type. For example, denying `REST-QUERY-STRING` prevents users from performing uploads or downloads from the Amazon S3 console. If you want users to use the Amazon S3 console, don't use this guardrail, or make an exception for users. On the other hand, if you don't want users to use the Amazon S3 console, you can deny `REST-QUERY-STRING` for users.

Perhaps you already deny users direct access to Amazon S3 resources. In this case, a guardrail for authentication type is redundant. However, an `s3:authType` deny statement provides defense-in-depth utility because implementations to deny direct access usually span many control statements, some with exceptions.

Roles that are used for workloads won't typically need access to query string or `POST` authentication. Exceptions are roles that support services designed to use presigned requests. You can create specific exceptions for those roles.

You can also apply a bucket policy to all or selected buckets by using a policy such as the following:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyNonHeaderAuth",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::{bucket-name}/*",
      "Condition": {
        "StringNotEquals": {
          "s3:authType": "REST-HEADER",
          "aws:PrincipalTag/non-header-auth-allowed": "true"
        }
      }
    }
  ]
}
```

This bucket policy has the effect of denying the use of the **CopyObject** and **UploadPartCopy** APIs to make cross-Region copies. Amazon S3 Replication isn't affected because it doesn't rely on these APIs.

If you want to use a bucket policy such as the preceding policy and still support the cross-Region **CopyObject** or **UploadPartCopy** API, add a condition for `aws:ViaAWSService` similar to the following:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "DenyNonHeaderAuth",
      "Effect": "Deny",
      "Principal": "*",
      "Action": "s3:*",
      "Resource": "arn:aws:s3:::{bucket-name}/*",
      "Condition": {
        "StringNotEquals": {
          "s3:authType": "REST-HEADER",
          "aws:PrincipalTag/non-header-auth-allowed": "true"
        },
        "Bool": {
          "aws:ViaAWSService": "false"
        },
      }
    }
  ]
}
```

Combining presigned guardrails and exceptions to other guardrails

If you don't plan to apply a guardrail generally to your users and roles, you might want to apply it to the exceptions of other common guardrails, so those exceptions don't support presigned requests.

If you have network restrictions but allow exceptions for external partners or special use cases, you should block query string or `POST` authentication when those exceptions are applied, unless they're specifically identified as being required.

Limitations to s3:signatureAge

Administrators will find it useful to understand the implications of `s3:signatureAge` more fully. Every signed request includes `X-Amz-Date`, which should indicate the current time. This value is filled by the client and request signer. AWS rejects requests that it considers to have invalid times. However, a signer could generate signatures in advance with a future time. Amazon S3 rejects requests that specifies a future time if it's sent too far in advance. However, if the request isn't sent until the time that's signed into the signature, the signature could be generated earlier and sent later.

`s3:signatureAge` limits the maximum age of `X-Amz-Date` in a signature only for presigned requests. Requests that are older than the specified age are denied, even if the expiration in `X-Amz-Expires` or a `POST` policy would have declared it valid. `s3:signatureAge` doesn't change the valid period for requests that don't include an explicit expiration. It also doesn't control the value of `X-Amz-Date` that a client uses for a signature.

If the system clock is wrong or if a client intentionally future-dates requests, the signed time might not be the time the signature was generated. This limits how much `s3:signatureAge` can control solutions. A solution that uses the current time when it generates signatures is limited in the expected way: The signatures remain valid for the number of milliseconds specified in `s3:signatureAge`. A solution that doesn't use the current time will have different limits. One restriction is that the credentials that were used to sign the signature must still be valid. As an administrator, you can control the maximum validity of temporary credentials issued. You can allow the credentials to be valid for up to 36 hours or restrict validity to as low as 15 minutes. The expiration of temporary credentials isn't dependent on the value of `X-Amz-Date`.

Permanent credentials don't have this restriction. Using only temporary credentials is a best practice, and you can explicitly revoke any permanent credential, which would also invalidate any signatures based upon that credential.

Although `s3:signatureAge` is measured in milliseconds, it isn't practical to set it to less than 60 seconds, even if you have a well-synchronized clock and low-latency usage. Settings that are lower than 60 seconds carry the risk of rejecting valid requests. If you expect delays between signature generation and request submission, or issues with clock synchronization, you should account for these in your management of `s3:signatureAge`.

Targeting buckets at scale

SCPs and RCPs can use `aws:PrincipalTag` to make exceptions for users. You can't use tags on a bucket to control access through `aws:ResourceTag`—only object tags are used for access control. It isn't generally scalable to add a tag to every object you want to apply this control to.

A solution that fits many use cases is to apply the policy and exception at the account level, either by changing the accounts that the SCP or RCP applies to or by using aws:ResourceAccount, aws:ResourceOrgPaths, or aws:ResourceOrgID. For example, an SCP or RCP might be applied to a set of production accounts.

Another solution is to use a custom AWS Config rule to implement a detective control or responsive control. The goal would be for every bucket to contain a bucket policy with the appropriate guardrail. In addition to testing the content of a bucket policy, the custom AWS Config rule can retrieve the tags from the bucket and exclude the bucket from the rule if the bucket is tagged with a specific value. If that rule fails its compliance check, it could either mark the bucket as non-compliant or invoke a remediation to add the guardrail to the bucket's policy.

Note

You can't restrict the tag content of requests to PutBucketTagging. To maintain control over how a bucket is tagged, you must limit access to `PutBucketTagging` and DeleteBucketTagging.

Logging interactions and mitigations

A presigned URL contains a signature and can be used, during the period before expiration, to perform the specific API operation it was signed for. It should be treated as a temporary access credential. The signature should remain private to only parties that need to know it. In most environments, this is the client that sends the request and the server that receives it. Sending the signature as part of a direct HTTPS session maintains its private nature, because only a participant of the HTTPS session has visibility into the URI that transmits the signature.

For presigned URLs, the signature is transmitted as the `X-Amz-Signature` query string parameter. Query string parameters are a component of a URI. The risk is that clients could log the URI and the signature with it. Clients have access to the entire HTTP request and could log any part of the request, data, and headers (including authentication headers). However, this is by convention less common. URI logging is more common and is required in cases such as access logging. Clients should use redaction or masking to remove the signature before logging URIs.

In some environments, users allow intermediaries (proxies) to gain visibility into their HTTPS sessions. Enabling proxies requires a high level of privileged access to client systems, because they require configuration and trusted certificates. The installation of proxy configuration and trusted certificates, within the local context of the client intermediary environment, allows a very high level of privilege. For this reason, access to such intermediaries should be tightly controlled.

The purpose of an intermediary is usually to block unwanted egress and to track other egress. As such, it's common for such intermediaries to log requests. Although intermediaries could, like clients, log any content, headers, and data (all of which would be very sensitive), it's more common for them to log URIs, such as those that include the `X-Amz-Signature` query string parameter.

Mitigations

We recommend that URI logging redact the `X-Amz-Signature` query string parameter, redact the entire query string, or treat the information as highly confidential, as with direct access to the intermediary server. Although these protections are highly recommended, the fact that presigned URLs expire mitigates the risks of log exposure, as long as the exposure is delayed long enough for the signatures to expire.

Amazon S3 also sees the signatures and must handle them appropriately. Amazon S3 server access logs include the request URI but redact the `X-Amz-Signature`, as recommended. The same is true when CloudTrail data events are logged for Amazon S3. You can configure Amazon CloudWatch Logs to mask data by using custom data identifiers.

The following regular expression matches the `X-Amz-Signature` as it appears in a URI:

```
X-Amz-Signature=[a-f0-9]{64}
```

This following regular expression adds grouping patterns to identify the text to replace more specifically:

```
(?:X-Amz-Signature=)([a-f0-9]{64})
```

If you have an access log entry such as the following:

```
X-Amz-Signature=733255ef022bec3f2a8701cd61d4b371f3f28c9f193a1f02279211d48d5193d7
```

The first regular expression translates the access log entry to:

```
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

The second regular expression, on systems that support non-capturing groups, translates the access log entry to:

```
X-Amz-Signature=XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

FAQ

Can a presigned request be used multiple times? Is that a security risk?

Yes, a signature in a presigned request could be used more than once. Whether this is a security risk is a contextual question. Other methods of accessing AWS services allow repetition as well. A user or workload with AWS credentials can send many requests to AWS services, and any of those requests could be duplicates.

If your use case requires once and only once execution, you should implement other mechanisms to enforce single use. Single use is not a feature of presigned requests. As a security engineer, you should review the use cases and implementations, but in many cases, multiple use will fit acceptable use.

Can someone other than the intended user use a presigned request?

A signature in a presigned request can be sent by anyone in possession of it. It will be accepted only if it passes other forms of validation, such as data perimeter controls. If the signature has expired, the signing credentials have expired, or the signing credentials don't have access to the requested resources, the request will be denied.

The same is true for other methods of authenticating with AWS services. Credentials that are shared inappropriately allow inappropriate access. The core best practice is to share credentials and signatures only with the intended audience. If you can't trust your intended audience to keep private data secure and not share it with others, this will undermine any form of authentication.

Can an authorized user use a presigned request to exfiltrate data?

Securing data requires robust action. Enabling access for intended purposes while maintaining a data perimeter requires a comprehensive approach. Least-privilege access, data perimeter controls, and using only temporary access credentials are general best practices that apply to securing data.

The appropriate use of these controls also limits the ability of users to perform actions through the presigned requests they generate.

This is because the access provided by a presigned requests is a subset of the access granted to the credentials that are used to sign the request. In this context, best practices that apply to accessing data generally do apply to presigned requests, but presigned requests create no new access to data.

- The maximum expiration is limited to the expiration of the signing credentials. If the signing credentials are revoked, signatures based upon the credentials are no longer valid.
- If the permissions for the IAM principal that is associated with the signing credentials do not include the execution of the action associated with the presigned request, invoking a presigned request results in an "access denied" response. The response depends on the current state of permissions at the time of invocation, which has no relationship to the time that the presigned request's signature was generated.
- Properties of the principal are evaluated based upon the principal that is associated with the signing credentials.
- Properties of a role session are evaluated based on the role session that is associated with the signing credentials.
- Properties of the network are evaluated based upon how the request was received, as with normal requests.

In this context, the examination of risks associated with presigned requests is constrained to areas where they are signed with credentials that are different from the user's credentials and provide access that was not part of a user's principal. This examination should be applied to the design of the service, the workload, or the solution that generates signatures on a user's behalf, instead of the presigned request capability itself.

Can I deny access from a presigned URL if I suspect it's been shared in an unauthorized way?

Yes. This requires invalidating the credentials the URL was signed with. There are multiple ways to accomplish this:

- Remove permissions from the IAM principal the credentials belong to. If that IAM principal no longer has access to the resource and operation that the URL is signed for, the URL can't execute that operation. This affects all matching use from that IAM principal.
- If the credentials used to sign the URL are temporary AWS STS credentials, you can revoke the session permissions for temporary credentials that were issued before a specific time for the IAM principal. Depending on the use case, there might be other valid sessions that become invalidated before their normal expiration time, but new sessions won't be affected. Revoking session permissions also invalidates any URLs that were signed by using credentials associated with those sessions, but new URLs associated with new sessions won't be affected.
- If the credentials used to sign the URL are permanent credentials, deactivate the access key. This affects all usage tied to those credentials.

Resources

Amazon S3 documentation

- Authenticating Requests (AWS Signature Version 4)
- Authenticating Requests: Using Query Parameters (AWS Signature Version 4)
- Authenticating Requests: Browser-Based Uploads Using POST (AWS Signature Version 4)
- Amazon S3 Signature Version 4 Authentication Specific Policy Keys
- Working with presigned URLs

Other references

- Building a Data Perimeter on AWS (AWS whitepaper)
- SEC03-BP02 Grant least privilege access (AWS Well Architected Framework, Security pillar)
- SEC03-BP05 Define permission guardrails for your organization (AWS Well Architected Framework, Security Pillar)

Appendix A: How AWS services use presigned URLs

This appendix provides information about AWS services and features that use presigned URLs. This information serves two purposes:

- To provide security engineers who implement controls with information about the possible impacts of those controls.
- To create awareness of situations where this risk might be relevant for URL logging interactions.

Important

This appendix doesn't provide a complete list of AWS services or their usage of presigned URLs. It also doesn't cover custom or third-party solutions.

Amazon S3 console

Principal: Console user

Default expiration: 5 minutes

Disclaimer

This section documents the current behavior of the Amazon S3 console. AWS console behaviors are subject to change without notice.

The Amazon S3 console supports downloading and uploading objects. Downloads use a presigned URL that has an expiration time of 300 seconds (5 minutes). The URL is generated by a request to `https://<bucket-region>.console.aws.amazon.com/s3/batchOpsServlet-proxy`.

That request is initiated when the user clicks a download button, so the URL isn't generated in advance or sent to the client until the explicit request to download occurs.

Uploads are similar, except that the console sends two requests: `OPTIONS` as a pre-flight CORS check, and PUT. Both requests use the same signature.

The credentials used for signing are temporary credentials that are associated with the currently logged in user. Details about the method for obtaining those temporary credentials are out of scope for this guide.

Amazon S3 Object Lambda

Principal: Access point caller

Default expiration: 61 seconds

Amazon S3 Object Lambda uses AWS Lambda functions to automatically process and transform data as it is retrieved from Amazon S3. When S3 Object Lambda invokes a function, the function is provided a presigned URL (`inputS3Url`) that it can use to download the original object from the supporting access point.

These presigned URLs are signed for the supporting Amazon S3 access point, which is provided when you configure S3 Object Lambda. (This is not the same as the Object Lambda access point.) Instead of using a role that's bound to the Lambda function, the URL is signed by using the original caller's identity, and that user's permissions will apply when the URL is used. If there are signed headers in the URL, the Lambda function must include these headers in the call to Amazon S3.

The presigned URL that's returned has an expiration time of 61 seconds (one second more than the maximum duration for an S3 Object Lambda function). The generated URL can be used only with the supporting access point. The caller of the S3 Object Lambda access point needs to have access to this access point. You can limit that access to the context of S3 Object Lambda by using the condition `"aws:CalledVia": ["s3-object-lambda.amazonaws.com"]`. When that condition is attached to a supporting access point or bucket, a user can't access the supporting access point or bucket directly.

The value of this approach is that there's no need to grant the Lambda function access to your S3 bucket or access point. The role that's associated with the Lambda function will need permissions for **WriteGetObjectResponse**, but it doesn't need permissions for **GetObject**.

When S3 Object Lambda generates presigned URLs, it doesn't add network restrictions, so a URL can be used outside the Lambda function. However, any restrictions placed on the caller of S3 Object Lambda still apply. For example, if your Lambda function runs in a VPC and you restrict the caller to using a VPC endpoint, anyone in possession of the presigned URL would need the ability to send it through that VPC endpoint. This restriction also applies to **SourceIp** and **VpcSourceIp**.

Note

To use an S3 Object Lambda function in a VPC, the VPC must have a route to public S3 endpoints to call **WriteGetObjectResponse**. This does not indicate that requirements to use a VPC endpoint would not apply to the requests to retrieve data from the bucket.

AWS Lambda Cross-Region CopyObject

Principal: AWS internal

Default expiration: 3600 seconds

When you use the CopyObject or UploadPartCopy API to copy across AWS Regions, Amazon S3 uses presigned URLs internally. These APIs can be called directly from SDKs or from the AWS CLI commands `aws s3api copy-object` and `aws s3api upload-part`. These APIs aren't used for Amazon S3 Replication, but they are used by the AWS CLI `aws s3 cp` and `aws s3 sync` commands when the source and destination are S3 buckets. They are also supported by `TransferManager` implementations in various AWS SDKs.

AWS Lambda GetFunction

Principal: AWS internal

Default expiration: 10 minutes

AWS Lambda stores the user version in a S3 bucket that the Lambda team owns, before generating the assets deployed to Lambda containers. When you want to access the code for your function, you call the GetFunction API. This API responds with `Code.Location`, which contains a presigned URL that's valid for 10 minutes (this expiration time is current behavior and not a published contract). If you don't want the code, you can use a combination of GetFunctionConfiguration, GetFunctionConcurrency, and ListTags to retrieve the other data that's returned by `GetFunction`.

The returned URL isn't signed with the credentials of the currently logged in user, but on behalf of the user by Lambda. For this reason, condition keys (such as `aws:SourceIP`) that are applied to the currently logged in user or the user's temporary session credentials don't apply to the generated URL. This is true whether condition keys are applied to **GetFunction** only, or applied to all AWS API usage for the user or session.

The Lambda console also uses **GetFunction** and the presigned URL it returns. The console uses the temporary credentials associated with the currently logged in user to call **GetFunction**. Details about obtaining those temporary credentials are out of scope for this document.

Amazon ECR

Principal: AWS internal

Default expiration: 1 hour

Amazon Elastic Container Registry (Amazon ECR) provides the GetDownloadUrlForLayer API, which returns a presigned URL that's valid for one hour and supports the download of a single layer from an Amazon ECR image. However, this operation is used by the Amazon ECR proxy and isn't generally used by users for pulling and pushing images.

Amazon Redshift Spectrum

Principal: Role passed to CREATE EXTERNAL SCHEMA through `IAM_ROLE`

Default expiration: 1 hour

Amazon Redshift Spectrum uses presigned URLs internally and prohibits restrictions on the combination of the bucket and Amazon Redshift role that would limit presigned URLs. You can use a `s3:signatureAge` value of 16 minutes, but very low values are unreliable. The minimum value you can use depends on the timing and size of your query. Although a value that's lower than 16 minutes works for many scenarios, it requires testing. The role can and should be restricted to be used only by Redshift Spectrum, which does not disclose the URLs it generates, thus mitigating the typical justification for lower expiration values.

Amazon SageMaker AI Studio

Amazon SageMaker AI Studio supports two API actions: CreatePresignedDomainUrl and CreatePresignedNotebookInstanceUrl. However, these APIs aren't related to the Signature Version 4 presigned URL feature. These APIs create a URL that uses an `authToken` parameter, but they don't support any of the standard Signature Version 4 query parameters.

`authToken` is a different mechanism but has similarities to presigned URLs. It's sent as a query string parameter and supports an expiration time of 5 minutes.

SageMaker AI supports network restrictions. If you place a restriction on the `sagemaker:CreatePresignedDomainUrl` action, that action applies both to calling CreatePresignedDomainUrl and to the use of the generated URL. If a URL is generated from a valid network and then sent by a non-valid network, the API call to generate the URL succeeds, but the request that sends the URL fails. The same is true of CreatePresignedNotebookInstanceUrl and the `sagemaker:CreatePresignedNotebookInstanceUrl` action.

For more information, see the SageMaker AI documentation.

Appendix B: How controls for presigned URLs affect AWS services

This appendix describes interactions between the AWS services that use presigned URLs, as described in Appendix A, and the controls described earlier in this guide.

Guardrail for s3:signatureAge

The Amazon S3 console isn't disrupted by the maximum expiration of 5 minutes that's set by the `s3:signatureAge` condition key. The Amazon S3 console generates presigned URLs when you choose the **Download** button and applies its own 5-minute expiration time. A maximum duration that's shorter than 2 minutes might create random failures based on clock synchronization and latencies.

Amazon S3 Object Lambda uses an expiration time of 61 seconds, so setting conditions on an `s3:signatureAge` value of 61 seconds or more won't cause any disruption. Shorter durations might be less reliable and might cause intermittent failures.

Amazon S3 cross-Region **CopyObject** isn't disrupted by a maximum expiration of 5 minutes. However, shorter durations might create random failures based on clock synchronization and latencies.

In AWS Lambda, **GetFunction** provides a URL to objects outside the customer account, so customer policies don't affect the generated URLs.

Amazon Redshift Spectrum has been tested with an `s3:signatureAge` condition of 16 minutes. However, shorter durations might cause disruption.

Guardrail for s3:authType when not using network restrictions

The Amazon S3 console is usually affected by the `s3:authType` guardrail. The console routes to Amazon S3 based on the local network configuration. If the local network routes to Amazon S3 in a way that the network restriction allows, the Amazon S3 console would still work. However, if it's routed through a proxy or the public internet in a way that isn't allowed, usage would be blocked. However, blocking usage is probably the intent of this policy.

Amazon S3 Object Lambda is affected if the Lambda function isn't connected to an appropriate VPC. In this configuration, the VPC must have a NAT gateway, not to access the S3 bucket, but to call **WriteGetObjectResponse**.

Amazon S3 cross-Region **CopyObject** is disrupted if this guardrail is applied to a bucket policy without the recommended exception for when `aws:viaAWSService` is **true**.

Amazon Redshift Spectrum is affected by the `s3:authType` guardrail unless enhanced VPC routing is used. Currently, Redshift Spectrum supports enhanced VPC routing only with serverless clusters, not with provisioned clusters.

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
Updates and clarifications	In the Additional guardrails section, added information about RCPs and clarified exceptions.	August 8, 2025
Initial publication	—	July 23, 2024

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, 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. For more information, see Enabling data persistence in microservices.

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. For more information, see the Quantifying uncertainty in deep learning systems guide.

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.