Implementing policies for least-privilege permissions for AWS CloudFormation

AWS Prescriptive Guidance

AWS Prescriptive Guidance: Implementing policies for least-privilege permissions for AWS CloudFormation

Table of Contents

Implementing policies for least-privilege permissions for AWS CloudFormation

Nima Fotouhi and Moumita Saha, Amazon Web Services (AWS)

May 2023 (document history)

AWS CloudFormation is an infrastructure as code (IaC) service that helps you scale your cloud infrastructure development by provisioning AWS resources. It also helps you manage those resources throughout their lifecycle, across AWS accounts and AWS Regions. In CloudFormation, you define templates, which act as a blueprint for a set of resources. You then provision those resources by creating and deploying a stack, which is a group of related resources that you manage as a single unit. You can also use CloudFormation to deploy stack sets, which are groups of stacks that you can create, update, and delete across multiple accounts and AWS Regions with a single operation. This guide provides an overview of how you can implement least-privilege permissions for AWS CloudFormation and resources provisioned through CloudFormation.

You can deploy CloudFormation stacks or stack sets by doing one of the following:

- Directly access the AWS environment through an AWS Identity and Access Management (IAM) principal and deploy CloudFormation stacks.
- Push the CloudFormation stacks in a deployment pipeline and initiate stack deployment through the pipeline. The pipeline accesses the AWS environment through an IAM principal and deploys the stacks. This approach is a recommended best practice.

For either of these approaches, permissions are required to deploy CloudFormation stacks. For example, consider a user planning to use CloudFormation to create an Amazon Elastic Compute Cloud (Amazon EC2) instance. That instance would require an IAM instance profile to access other AWS services. The IAM principal used to deploy the CloudFormation stack would require the following permissions:

- Permissions to access CloudFormation
- Permissions to create stacks in CloudFormation
- Permissions to create instances in Amazon EC2
- Permissions to create the required IAM instance profiles

What is least privilege?

Least privilege is the security best practice of granting the minimum permissions required to perform a task. The principle of least privilege is part of the Security pillar in the AWS Well-Architected Framework. When you implement this best practice, it can help protect your AWS environment from privilege escalation risks, reduce the attack surface, improve data security, and prevent user error (such as misconfiguring or deleting a resource by mistake).

To implement least privilege for your AWS resources, you configure policies, such as identity-based policies in AWS Identity and Access Management (IAM). These policies define permissions and specify access conditions. Organizations might start with AWS managed policies, but then they typically create custom policies that limit the scope of permissions to only the actions required for the workload or use case.

Least-privilege permissions for the CloudFormation service is an important security consideration. Because users and developers who interact with CloudFormation can have the ability to rapidly create, modify, or delete resources at scale, least privilege is especially critical. However, CloudFormation requires the permissions necessary to create, update, and modify resources in your AWS accounts. You must balance the need for permissions to operate CloudFormation with the principle of least privilege.

When applying the principle of least privilege to CloudFormation, you need to consider the following:

- **Permissions for the CloudFormation service** – Which users require access to CloudFormation, what level of access do they require, and what actions can they take to create, update, or delete stacks?
- **Permissions to provision resources** – Which resources can users provision through CloudFormation?
- **Permissions for provisioned resources** – How do you configure least-privilege permissions for the resources you provision through CloudFormation?

Targeted business outcomes

By following the best practices and recommendations in this guide, you can:

- Determine which users in your organization require access to CloudFormation, and then configure least-privilege permissions for those users.
- Use stack policies to help protect CloudFormation stacks from unintended updates.
- Configure least-privilege permissions for CloudFormation users and resources to help prevent privilege escalation and the confused deputy problem.
- Use AWS CloudFormation to provision AWS resources with least-privilege permissions. This helps your organization maintain a more robust security posture.
- Proactively reduce the amount of time, energy, and money required to investigate and mitigate security incidents.

Intended audience

This guide is intended for Cloud Infrastructure Architects, DevOps engineers, and site reliability engineers (SREs) who manage and provision resources by using CloudFormation.

Using access policies to grant permissions in AWS

You manage access in AWS by creating *identity-based policies* and attaching them to AWS Identity and Access Management (IAM) principals, such as roles or users, and by creating *resource-based policies* and attaching them to AWS resources. AWS evaluates these policies whenever a request is made. Permissions in the policies determine whether the request is allowed or denied.

To understand how to configure least-privilege access in policies, you need to understand the different types of policies, the elements and structure of a policy, and how policies are evaluated. This guide only focuses on identity-based policies and resource-based policies. However, AWS provides other types of policies, such as *service control policies (SCPs)*, *permissions boundaries*, and *session policies*. Each type of policy plays a role in implementing least-privilege permissions in your AWS accounts. For more information, see Policies and permissions and Apply least-privilege permissions in the IAM documentation.

Configuring least-privilege permissions to use CloudFormation

This chapter reviews the options for configuring permissions to access and use the AWS CloudFormation service.

When a user or a service provisions AWS resources through CloudFormation, the first step is to make a call to the CloudFormation service through an AWS Identity and Access Management (IAM) principal. This IAM principal must have permissions to create the CloudFormation stacks. Next, the IAM principal uses one of the following approaches to provision resources through CloudFormation:

- If the IAM principal doesn't pass the stack operations to a CloudFormation service role, CloudFormation uses the credentials of the IAM principal to perform the stack operations. This is the default. Therefore, in addition to permissions to perform the CloudFormation stack operations, the IAM principal also needs permissions to provision the resources defined in the CloudFormation templates they'll be using. For example, if the IAM principal doesn't have permissions to create Amazon Elastic Compute Cloud (Amazon EC2) instances, then they can't create a CloudFormation stack that would provision an Amazon EC2 instance.
- If the IAM principal passes the stack operations to a CloudFormation service role, then CloudFormation uses the service role to perform the stack operations and provision the resources in the CloudFormation template. This CloudFormation service role should be defined with permissions to provision the AWS services on behalf of the IAM principal. This approach avoids giving direct permissions to the IAM principal to provision the AWS resources defined in the CloudFormation templates. The IAM principal needs CloudFormation stack creation permissions, and CloudFormation uses the service role's policy to make calls instead of the IAM principal's policy.

By using the service role approach and the principle of least privilege, you can standardize resource provisioning in your AWS environment and require that users provision resources as IaC through CloudFormation. Because the policies attached to IAM principals don't contain permissions to provision AWS resources directly, users must use CloudFormation to provision them.

This chapter reviews the following mechanisms for configuring and managing access to the CloudFormation service and to CloudFormation stacks:

- Identity-based policies for CloudFormation – Use this type of policy to configure which IAM principals can access CloudFormation and which actions they can perform in CloudFormation.
- Service roles for CloudFormation – Create a service role that allows CloudFormation to create, update, or delete stack resources on behalf of the IAM principal who deploys the stack. The service role is created in IAM and can be associated with one or more stacks.
- CloudFormation stack policies – Use this type of policy to determine when a stack can be updated. This type of policy can help prevent stack resources from being unintentionally updated or deleted. Stack policies are created and associated to stacks in CloudFormation.

Identity-based policies for CloudFormation

Consider the types of users who need access to AWS CloudFormation, and consider which actions those users need to perform in CloudFormation. You configure user permissions through identity-based policies, which you attach to an AWS Identity and Access Management (IAM) principal, such as a role or user.

When you configure an identity-based policy, the `Effect`, `Action`, and `Resource` elements are required. You can optionally define a `Condition` element too. For more information about these elements, see IAM JSON policy elements reference.

This section contains the following topics:

Best practices for configuring identity-based policies for least-privilege CloudFormation access

- For IAM principals who require permissions to access CloudFormation, you must balance the need for permissions to operate CloudFormation with the principle of least privilege. To help you adhere to the principle of least privilege, we recommend that you define the IAM principal's identity-based with specific actions that allow the principal to do the following:
  - Create, update, and delete a CloudFormation stack.
  - Pass one or more service roles that have the permissions required to deploy the resources defined in the CloudFormation templates. This allows CloudFormation to assume the service role and provision the resources in the stack on behalf of the IAM principal.

- *Privilege escalation* refers to the ability of a user with access, to elevate their permission levels and compromise security. Least-privilege is an important best practice that can help prevent privilege escalation. Because CloudFormation supports provisioning of IAM resource types, such as policies and roles, an IAM principal could escalate their privileges through CloudFormation by:
  - **Using a CloudFormation stack to provision an IAM principal with highly privileged permissions, policies, or credentials** – To help prevent this, we recommend using permission guardrails to constrain the level of access for IAM principals. *Permission guardrails* set the maximum permissions that an identity-based policy can grant to an IAM principal. This helps prevent intentional and unintentional privilege escalation. You can use the following types of policies as permissions guardrails:
    - Permissions boundaries define the maximum permissions that an identity-based policy can grant to an IAM principal. For more information, see Permissions boundaries for IAM entities.
    - In AWS Organizations, you can use service control policies (SCPs) to define the maximum available permissions at an organizational level. SCPs affect only IAM roles and users that are managed by accounts in the organization. You can attach SCPs to accounts, organizational units, or to the organizational root. For more information, see SCP effects on permissions.
  - **Creating a CloudFormation service role that offers extensive permissions** – To help prevent this, we recommend that you add the following fine-grained permissions to the identity-based policies for IAM principals who will be using CloudFormation:
    - Use the `cloudformation:RoleARN` condition key to control which CloudFormation service roles the IAM principal can use.
    - Allow the `iam:PassRole` action only for the specific CloudFormation service roles that the IAM principal needs to pass.

  For more information, see Granting an IAM principal permissions to use a CloudFormation service role in this guide.
- Restrict permissions by using permissions guardrails, such as permissions boundaries and SCPs, and grant permissions by using an identity-based or resource-based policy.

Sample identity-based policies for CloudFormation

This section contains sample identity-based policies that demonstrate how to grant and deny permissions for CloudFormation. You can use these sample policies to start designing your own policies that adhere to the principle of least privilege.

For a list of CloudFormation specific actions and conditions, see Actions, resources, and condition keys for AWS CloudFormation and AWS CloudFormation conditions. For a list of resource types to use with conditions, see AWS resource and property types reference.

This section contains the following example policies:

- Allow view access
- Allow stack creation based on template
- Deny update or deletion of a stack

Allow view access

View access is the least-privileged type of access to CloudFormation. This kind of policy might be appropriate for those IAM principals who want to view all of the CloudFormation stacks in the AWS account. The following sample policy grants permissions to view the details of any CloudFormation stack in the account.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:DescribeStacks",
        "cloudformation:DescribeStackEvents",
        "cloudformation:DescribeStackResource",
        "cloudformation:DescribeStackResources"
      ],
      "Resource": "*"
    }
  ]
}
```

Allow stack creation based on template

The following sample policy allows IAM principals to create stacks by using only the CloudFormation templates that are stored in a specific Amazon Simple Storage Service (Amazon S3) bucket. The bucket name is `my-CFN-templates`. You can upload approved templates to this bucket. The `cloudformation:TemplateUrl` condition key in the policy prevents the IAM principal from using any other templates to create stacks.

> **Important**
>
> Allow the IAM principal to have read-only access to this S3 bucket. This helps prevent the IAM principal from adding, removing, or modifying the approved templates.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:CreateStack"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "cloudformation:TemplateUrl": "https:// my-CFN-templates.s3.amazonaws.com/*"
        }
      }
    }
  ]
}
```

Deny update or deletion of a stack

To help protect specific CloudFormation stacks that provision business-critical AWS resources, you can restrict update and deletion actions for that specific stack. You can allow these actions for only a few specified IAM principals and deny them for any other IAM principal in the environment. The following policy statement denies permissions to update or delete a specific CloudFormation stack in a specific AWS Region and AWS account.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "cloudformation:DeleteStack",
        "cloudformation:UpdateStack"
```

```
      ],
      "Resource": "arn:aws:cloudformation:us-east-1:123456789012:stack/
MyProductionStack/<stack_ID>"
    }
  ]
}
```

This policy statement denies permissions to update or delete the `MyProductionStack` CloudFormation stack, which is in the `us-east-1` AWS Region and in the `123456789012` AWS account. You can view the stack ID in the CloudFormation console. The following are some examples of how you could modify the `Resource` element of this statement for your use case:

- You can add multiple CloudFormation stack IDs in the `Resource` element of this policy.
- You can use `arn:aws:cloudformation:us-east-1:123456789012:stack/*` to prevent IAM principals from updating or deleting any stack that is in the `us-east-1` AWS Region and in the `123456789012` account.

An important step is deciding which policy should contain this statement. You could add this statement to the following policies:

- **The identity-based policy attached to the IAM principal** – Putting the statement in this policy restricts the specific IAM principal from creating or deleting a specific CloudFormation stack.
- **A permissions boundary attached to the IAM principal** – Putting the statement in this policy creates a permission guardrail. It restricts more than one IAM principal from creating or deleting a specific CloudFormation stack, but it doesn't restrict all principals in your environment.
- **A SCP attached to an account, organizational unit, or organization** – Putting the statement in this policy creates a permission guardrail. It restricts all IAM principals in the target account, organizational unit, or organization from creating or deleting a specific CloudFormation stack.

  However, if you don't allow at least one IAM principal, a *privileged principal*, to update or delete the CloudFormation stack, then you will not be able to make any changes, when necessary, to the resources provisioned through this stack. A user or a development pipeline (recommended) can assume this privileged principal. If you want to deploy the restriction as an SCP, then we recommend the following policy statement instead.

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Deny",
      "Action": [
        "cloudformation:DeleteStack",
        "cloudformation:UpdateStack"
      ],
      "Resource": "arn:aws:cloudformation:us-east-1:123456789012:stack/
MyProductionStack/<stack_ID>",
      "Condition": {
        "ArnNotLike": {
          "aws:PrincipalARN": [
            "<ARN of the allowed privilege IAM principal>"
          ]
        }
      }
    }
  ]
}
```

In this statement, the `Condition` element defines the IAM principal that is excluded from the SCP. This statement denies any IAM principal permissions to update or delete CloudFormation stacks unless the ARN of the IAM principal matches the ARN in the `Condition` element. The `aws:PrincipalARN` condition key accepts a list, which means that you can exclude more than one IAM principal from the restrictions, as necessary for your environment. For a similar SCP that prevents modifications to CloudFormation resources, see SCP-CLOUDFORMATION-1 (GitHub).

Service roles for CloudFormation

A *service role* is an AWS Identity and Access Management (IAM) role that allows AWS CloudFormation to create, update, or delete stack resources. If you do not provide a service role, CloudFormation uses the credentials of the IAM principal to perform the stack operations. If you create a service role for CloudFormation and specify the service role during stack creation, then CloudFormation uses the credentials of the service role to perform the operations, instead of the credentials of the IAM principal.

When using a service role, the identity-based policy attached to the IAM principal doesn't require permissions to provision all of the AWS resources defined in the CloudFormation template. If you are not ready to provision AWS resources for critical business operations through a development

pipeline (an AWS recommended best practice), using a service role can add an extra layer of protection for resource management in AWS. The benefits of this approach are:

- The IAM principals in your organization follow a least-privilege model that prevents them from manually creating or changing AWS resources in your environment.
- To create, update, or delete AWS resources, IAM principals must use CloudFormation. This standardizes resource provisioning through infrastructure as code.

For example, to create a stack that contains an Amazon Elastic Compute Cloud (Amazon EC2) instance, the IAM principal would need to have permissions to create EC2 instances through their identity-based policy. Instead, CloudFormation can assume a service role that has permissions to create EC2 instances on the principal's behalf. With this approach, the IAM principal can create the stack, and you don't need to give the IAM principal overly broad permissions for a service that they shouldn't have regular access to.

To use a service role to create CloudFormation stacks, IAM principals must have permissions to pass the service role to CloudFormation, and the trust policy of the service role must allow CloudFormation to assume the role.

This section contains the following topics:

Implementing least privilege for CloudFormation service roles

In a service role, you define a permissions policy that explicitly specifies which actions the service can perform. These might not be the same actions that an IAM principal can perform. We recommend that you work backward from your CloudFormation templates to create a service role that adheres to the principle of least privilege.

Properly scoping the identity-based policy of an IAM principal to pass only specific service roles and scoping a service role's trust policy to allow only specific principals to assume the role helps prevent possible privilege escalation through service roles.

Configuring service roles

> **Note**
>
> Service roles are configured in IAM. To create a service role, you must have permissions to do so. An IAM principal with permissions to create a role and attach any policy can escalate their own permissions. AWS recommends creating one service role for each AWS service for each use case. After you create CloudFormation service roles for your use cases, you can allow users to pass only the approved service role to CloudFormation. For sample identity-based policies that allow users to create service roles, see Service role permissions in the IAM documentation.

For instructions about how to create service roles, see Creating a role to delegate permissions to an AWS service. Specify CloudFormation (`cloudformation.amazonaws.com`) as the service that can assume the role. This prevents an IAM principal from assuming the role themselves or passing it to other services. When you configure a service role, the `Effect`, `Action`, and `Resource` elements are required. You can optionally define a `Condition` element too.

For more information about these elements, see IAM JSON policy elements reference. For a complete list of actions, resources, and condition keys, see Actions, resources, and condition keys for Identity And Access Management.

Granting an IAM principal permissions to use a CloudFormation service role

To provision resources through CloudFormation by using the CloudFormation service role, the IAM principal must have permissions to pass the service role. You can limit the IAM principal's permissions to pass only certain roles by specifying the ARN of the role in the principal's permissions. For more information, see Granting a user permissions to pass a role to an AWS service in the IAM documentation.

The following IAM identity-based policy statement allows the principal to pass roles, including service roles, that are in the `cfnroles` path. The principal cannot pass roles that are in a different path.

```
{
```

```
"Sid": "AllowPassingAppRoles",
"Effect": "Allow",
"Action": "iam:PassRole",
"Resource": "arn:aws:iam::<account ID>:role/cfnroles/*"
}
```

Another approach for limiting principals to certain roles is to use a prefix for CloudFormation service role names. The following policy statement allows IAM principals to pass only roles that have a `CFN-` prefix.

```
{
"Sid": "AllowPassingAppRoles",
"Effect": "Allow",
"Action": "iam:PassRole",
"Resource": "arn:aws:iam::<account ID>:role/CFN-*"
}
```

In addition to the previous policy statements, you can use the `cloudformation:RoleARN` condition key to provide further fine-grained controls in the identity-based policy, for least privilege access. The following policy statement allows the IAM principal to create, update, and delete stacks only if they pass a specific CloudFormation service role. As a variation, you can define the ARNs of more than one CloudFormation service role in the condition key.

```
{
  "Sid": "RestrictCloudFormationAccess",
  "Effect": "Allow",
  "Action": [
    "cloudformation:CreateStack",
    "cloudformation:DeleteStack",
    "cloudformation:UpdateStack"
  ],
  "Resource": "arn:aws:iam::<account ID>:role/CFN-*",
  "Condition": {
    "StringEquals": {
      "cloudformation:RoleArn": [
        "<ARN of the specific CloudFormation service role>"
      ]
    }
  }
}
```

Additionally you can also use the `cloudformation:RoleARN` condition key to restrict an IAM principal from passing a highly privileged CloudFormation service role for stack operations. The only change required is in the conditional operator, from `StringEquals` to `StringNotEquals`.

```
{
  "Sid": "RestrictCloudFormationAccess",
  "Effect": "Allow",
  "Action": [
    "cloudformation:CreateStack",
    "cloudformation:DeleteStack",
    "cloudformation:UpdateStack"
  ],
  "Resource": "arn:aws:iam::<account ID>:role/CFN-*",
  "Condition": {
    "StringNotEquals": {
      "cloudformation:RoleArn": [
        "<ARN of a privilege CloudFormation service role>"
      ]
    }
  }
}
```

Configuring a trust policy for the CloudFormation service role

A role *trust policy* is a required resource-based policy that is attached to an IAM role. A trust policy defines which IAM principals can assume the role. In a trust policy, you can specify users, roles, accounts, or services as principals. To prevent IAM principals from passing service roles for CloudFormation to other services, you can specify CloudFormation as the principal in role's trust policy.

The following trust policy allows only the CloudFormation service to assume the service role.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {
      "Service": "cloudformation.amazonaws.com"
    },
    "Action": "sts:AssumeRole"
  }
```

```
}
```

Associating a service role with a stack

After a service role is created, you can associate it with a stack when you create the stack. For more information, see Configure stack options. Before you specify a service role, make sure that IAM principals have permissions to pass it. For more information, see Granting an IAM principal permissions to use a CloudFormation service role.

CloudFormation stack policies

Stack policies can help prevent stack resources from being unintentionally updated or deleted during a stack update. A *stack policy* is a JSON document that defines the update actions that can be performed on designated resources. By default, any IAM principal with `cloudformation:UpdateStack` permissions can update all of the resources in an AWS CloudFormation stack. Updates can cause interruptions, or they can completely delete and replace resources. You can use a stack policy to help configure least-privilege permissions. The stack policies can provide an extra layer of protection.

By default, a stack policy helps protect all resources in the stack. However, the main benefit of stack policies that they provide granular control for each AWS resource deployed in a CloudFormation stack. You can use a stack policy to help protect only specific resources in a stack and allow updates or deletion of other resources in the same stack. To allow updates for specific resources, you include an explicit `Allow` statement for those resources in your stack policy.

Stack policies provide preventive controls for the CloudFormation stacks they are attached to. Each stack can have only one stack policy, but you can use that stack policy to help protect all resources within that stack. You can apply a stack policy to multiple stacks.

For example, imagine you have a pipeline that produces sensitive artifacts and stores them in an Amazon Simple Storage Service (Amazon S3) bucket temporarily for further processing. The S3 bucket is provisioned by CloudFormation, and all of the necessary security controls are in place. Without stack policies, a developer might intentionally or unintentionally change the destination of the pipeline artifacts to a less secure S3 bucket and expose sensitive data. If you have a stack policy applied to the stack, it prevents authorized users from performing unwanted update or delete actions.

This section contains the following topics:

Configuring stack policies

When you configure a stack policy, the `Effect`, `Action`, `Principal`, and `Resource` elements are required. You can optionally define a `Condition` element too.

When you create a stack policy, by default, it prevents updates for all resources in the stack. You customize the stack policy to define which actions are explicitly allowed. If you want to invert the policy, you can define an `Allow` statement that permits all actions and then specify explicit `Deny` statements that prevent actions on only specific resources. For reference, see this example stack policy in the CloudFormation documentation.

For more information about using these elements to create custom stack policies and more example policies, see Defining a stack policy and More example stack policies in the CloudFormation documentation.

Setting and overriding stack policies

After you create a stack policy, you associate it to a stack. If you are assigning the stack policy to an existing stack, you must use the AWS Command Line Interface (AWS CLI). However, if you are assigning the policy at the time of stack creation, you can use either the CloudFormation console or the AWS CLI. For instructions, see Setting a stack policy in the CloudFormation documentation.

When you do want to allow users to update or delete the resources in the stack, you need to temporarily override the stack policy. This override allows you to perform otherwise denied actions on the protected resources in that stack. For instructions, see Updating protected resources in the CloudFormation documentation.

Limiting and requiring stack policies

As a best practice for least-privilege permissions, consider requiring IAM principals to assign stack policies and limiting which stack policies IAM principals can assign. Many IAM principals should not have permissions to create and assign custom stack policies to their own stacks.

After you create your stack policies, we recommend that you upload them to an S3 bucket. You can then reference these stack policies by using the `cloudformation:StackPolicyUrl` condition key and providing the URL of the stack policy in the S3 bucket.

Granting permissions to attach stack policies

As a best practice for least-privilege permissions, consider limiting which stack policies IAM principals can attach to CloudFormation stacks. In the identity-based policy for the IAM principal, you can specify which stack policies the IAM principal has permissions to assign. This prevents the IAM principal from attaching any stack policy, which can reduce the risk of misconfiguration.

For example, an organization might have different teams with different requirements. Accordingly, each team builds stack policies for their team-specific CloudFormation stacks. In a shared environment, if all teams store their stack policies in the same S3 bucket, a team member might attach a stack policy that is available but not intended for their team's CloudFormation stacks. To avoid this scenario, you can define a policy statement that allows IAM principals to attach only specific stack policies.

The following sample policy allows the IAM principal to attach stack policies that are stored in a team-specific folder in an S3 bucket. You can store approved stack policies in this bucket.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "cloudformation:SetStackPolicy"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "cloudformation:StackPolicyUrl": "<Bucket URL>/<Team folder>/*"
        }
      }
    }
  ]
}
```

This policy statement doesn't require an IAM principal to assign a stack policy to every stack. Even if the IAM principal has permissions to create stacks with a specific stack policy, they could choose to create a stack that doesn't have a stack policy.

Requiring stack policies

To ensure all IAM principals assign stack policies to their stacks, you can define a service control policy (SCP) or permissions boundary as a preventive guardrail.

The following sample policy shows how you can configure an SCP that requires IAM principals to assign a stack policy when creating a stack. If the IAM principal doesn't attach a stack policy, they can't create the stack. Additionally, this policy prevents IAM principals with stack update permissions from removing the stack policy during an update. the policy restricts the `cloudformation:UpdateStack` action by using the `cloudformation:StackPolicyUrl` condition key.

```
        {
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "cloudformation:CreateStack",
        "cloudformation:UpdateStack"
      ],
      "Resource": "*",
      "Condition": {
        "Null": {
          "cloudformation:StackPolicyUrl": "true"
        }
      }
    }
  ]
}
```

By including this policy statement in an SCP instead of a permissions boundary, you can apply your guardrail to all accounts in the organization. This can do the following:

1. Reduce the effort to attach the policy individually to multiple IAM principals in an AWS account. Permissions boundaries can be directly attached only to an IAM principal.

2. Reduce the effort to create and manage multiple copies of the permissions boundary for different AWS accounts. This reduces the risk of configuration error in multiple identical permissions boundaries.

> **Note**
>
> SCPs and permissions boundaries are permissions guardrails that define the maximum available permissions for IAM principals in an account or organization. These policies do not grant permissions to the IAM principals. If you want to standardize the requirement that all IAM principals in your account or organization assign stack policies, you need to use both permission guardrails and the identity-based policies.

Configuring least-privilege permissions for resources provisioned through CloudFormation

AWS CloudFormation allows you to provision many different types of AWS resources. Provisioned resources require their own set of permissions to function as intended and to configure who has access to those resources. The previous chapter reviewed options for configuring permissions to access and use the CloudFormation service. This chapter reviews how you can apply the principle of least privilege to resources provisioned through CloudFormation.

In this guide, it would be practically impossible to review the security recommendations and best practices for every type of AWS resource that can be provisioned through CloudFormation. If you have questions related to a specific service, we recommend that you review the documentation for that service. Most AWS service documents contain a security section and information about the permissions required to use that service. For a complete list of AWS service documentation, see AWS Documentation.

The following are high-level, service-agnostic steps you can take to create CloudFormation templates that adhere to the principle of least privilege:

1. Prepare a list of resources that you are planning to provision by using CloudFormation.
2. See the AWS Documentation for the corresponding services and review the sections about security and access management. This helps you understand the service-specific requirements and recommendations.
3. Use the information you gathered in the previous steps to design CloudFormation templates and associated policies that allow only the required permissions and deny all others.

Next, this guide reviews an example of how you can apply the principle of least privilege in CloudFormation templates, using a real-world use case.

Example: Amazon S3 bucket for storing pipeline artifacts

This example creates an Amazon Simple Storage Service (Amazon S3) bucket that is used to store AWS CodeBuild project artifacts. AWS CodePipeline uses these stored artifacts. You can allow CodeBuild and CodePipeline to access this S3 bucket through service roles, and you control that

access by using an Amazon S3 bucket policy. The following are the resource names used in this example:

- `Deployfiles_build` is the name of the CodeBuild project.
- `Deployment-Pipeline` is the name of the pipeline in CodePipeline.

Define the Amazon S3 bucket

First, you define the S3 bucket in the CloudFormation template, which is a YAML-formatted text file.

```
amzn-s3-demo-bucket:
  Type: AWS::S3::Bucket
  Properties:
    PublicAccessBlockConfiguration:
        BlockPublicAcls: true
        BlockPublicPolicy: true
        IgnorePublicAcls: true
        RestrictPublicBuckets: true
```

Define the Amazon S3 bucket policy

Next, in the CloudFormation template, you create a bucket policy that allows only the `Deployfiles_build` project and the `Deployment-Pipeline` pipeline to access the bucket.

```
MyBucketPolicy:
  Type: AWS::S3::BucketPolicy
  Properties:
    Bucket: !Ref amzn-s3-demo-bucket
    PolicyDocument:
      Version: "2012-10-17"
      Statement:
      - Sid: "S3ArtifactRepoAccess"
        Effect: Allow
        Action:
          - ‘s3:GetObject'
          - ‘s3:GetObjectVersion'
          - ‘s3:PutObject'
          - ‘s3:GetBucketVersioning'
        Resource:
          - !Sub ‘arn:aws:s3:::${amzn-s3-demo-bucket}'
```

```
          - !Sub 'arn:aws:s3:::${amzn-s3-demo-bucket}/*'
        Principal:
          Service:
            - codebuild.amazonaws.com
            - codepipeline.amazonaws.com
        Condition:
          StringLike:
            'aws:SourceArn':
              - !Sub 'arn:aws:codebuild:${AWS::Region}:${AWS::AccountId}:project/
Deployfiles_build'
              - !Sub 'arn:aws:codepipeline:${AWS::Region}:${AWS::AccountId}:Deployment-
Pipeline'
              - !Sub 'arn:aws:codepipeline:${AWS::Region}:${AWS::AccountId}:Deployment-
Pipeline/*'
```

Note the following about this bucket policy:

- The `Resource` element lists two different types of resources that use the following Amazon Resource Name (ARN) formats:
  - The ARN format of an S3 object is `arn:$`*`<Partition>`*`:s3:::$`*`<BucketName>`*`/$`*`<ObjectName>`*.
  - The ARN format of an S3 bucket is `arn:$`*`<Partition>`*`:s3:::$`*`<BucketName>`*.

  `s3:GetObject`, `s3:GetObjectVersion`, and `s3:PutObject` require an S3 object resource type, and `s3:GetBucketVersioning` requires an S3 bucket resource type. For more information about the required resource types for each action, see Actions, resources, and condition keys for Amazon S3.
- The `Principal` element lists the entities that are allowed to perform the Amazon S3 actions defined in the statement. In this case, only CodeBuild and CodePipeline are allowed to perform these actions.
- The `Condition` element further restricts access to the S3 bucket so that only the `Deployfiles_build` CodeBuild project, the `Deployment-Pipeline` CodePipeline pipeline, and the pipeline actions can access the bucket.

Create the service roles

Although the bucket policy controls access to the bucket, it doesn't grant permissions to CodeBuild and CodePipeline to access it. To grant access, you need to create a service role for each service and

add the following statement to each. The services roles for CodeBuild and CodePipeline allow the services to access the S3 bucket and its objects.

```
Sid: "ViewAccessToS3ArtifactRepo"
Effect: Allow
Action:
  - ‘s3:GetObject'
  - ‘s3:GetObjectVersion'
  - ‘s3:PutObject'
  - ‘s3:GetBucketVersioning'
Resource:
  - !Sub ‘arn:aws:s3:::${BuildArtifactsBucket}'
  - !Sub ‘arn:aws:s3:::${BuildArtifactsBucket}/*'
```

Best practices for least-privilege permissions for AWS CloudFormation

This guide reviews different approaches and some types of policies that you can use to configure least-privilege access to AWS CloudFormation and resources provisioned through CloudFormation. This guide focuses on configuring access to CloudFormation through IAM principals, service roles, and stack policies. The included recommendations and best practices are designed to help protect your accounts and stack resources from unintended actions by authorized users and from bad actors who might exploit excessive permissions.

The following is a summary of the best practices explained in this guide. These best practices can help you adhere to the principle of least privilege when configuring permissions to use CloudFormation and resources provisioned through CloudFormation:

- Determine what level of access users and teams need to use the CloudFormation service, and grant only the minimum access required. For example, grant view access to interns and auditors, and do not allow these types of users to create, update, or delete stacks.
- For IAM principals who need to provision multiple types of AWS resources through CloudFormation stacks, consider using service roles to allow CloudFormation to provision resources on the principal's behalf, instead of configuring access to those AWS services in the principal's identity-based policies.
- In identity-based policies for IAM principals, use the `cloudformation:RoleARN` condition key to control which CloudFormation service roles can be passed.
- To help prevent privilege escalation, do the following:
  - Strictly monitor all the IAM principals that have access to the CloudFormation service and the levels of access they have.
  - Strictly monitor which users can access these IAM principals.
  - Monitor the activity of IAM principals that can pass a privileged service role to CloudFormation. Although they might not have permissions to create IAM resources through their identity-based policy, the service role they can pass could create IAM resources.
- Specify a stack policy whenever you create a stack that has critical resources. This can help protect critical stack resources from unintentional updates that could cause those resources to be interrupted or replaced.

- For resources provisioned through CloudFormation, refer to the access management recommendations and security best practices for that service.
- To complement the recommendations in this guide for identity-based policies and resource-based policies, consider implementing additional security controls for least-privilege permissions, such as service control policies (SCPs) and permissions boundaries. For more information, see Next steps.

The CloudFormation documentation contains additional Best practices and Security best practices that can help you use CloudFormation more effectively and securely. In addition, see Best practices for configuring identity-based policies for least-privilege CloudFormation access in this guide.

Next steps

You can use the information and examples in this guide to start applying the principle of least privilege in your organization. We recommend that you review the additional resources in the Resources section, which contains documentation references and tools that can help you refine your policies.

This guide is intended to help you start implementing least-privilege access for AWS CloudFormation. However, there are additional types of policies that can help you strengthen the principle of least-privilege in your organization. Based on your environment and business requirements, you might want to implement additional controls that are not discussed in this guide. As a next step and for more information, we recommend that you review the following topics related to least privilege and configuring access and permissions:

- Permissions boundaries for IAM entities
- Service control policies (SCP)
- Roles for cross-account access
- Identity federation
- Viewing last accessed information for IAM

The following tools can help you monitor least-privilege access and permissions for CloudFormation:

- AWS Identity and Access Management Access Analyzer
- You can use the Access Advisor tab in the AWS Identity and Access Management (IAM) console to identify excessive permissions for IAM identities. For an example, see Tighten S3 permissions for your IAM users and roles using access history of S3 actions (AWS blog post).
- You can use a linting tool, such as cfn-policy-validator (GitHub), to help identify excessive permissions.

When you are comfortable with creating and managing CloudFormation permissions, it is recommended that you use continuous integration and continuous delivery (CI/CD) pipelines to deploy your CloudFormation templates. This reduces the risk of human errors and speeds up your deployment process.

Resources

AWS CloudFormation documentation

- Controlling access with AWS Identity and Access Management
- AWS resource and property types reference
- Setting AWS CloudFormation stack options
- AWS CloudFormation service role

AWS Identity and Access Management (IAM) documentation

- Policies and permissions in IAM
- IAM JSON policy elements reference
- Policy evaluation logic
- AWS services that work with IAM
- Creating a role to delegate permissions to an AWS service
- The confused deputy problem
- Security best practices in IAM

Other AWS references

- Actions, resources, and condition keys for AWS services (Service Authorization Reference)
- Grant least privilege access (AWS Well-Architected Framework)
- Techniques for writing least privilege IAM policies (AWS blog post)

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
Significant updates	We significantly revised and refined the guidance and sample policy statements to address common organizat ional use cases.	May 5, 2023
Initial publication	—	March 9, 2023

AWS Prescriptive Guidance glossary

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

Numbers

7 Rs

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

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

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

A

ABAC

See attribute-based access control.

abstracted services

See managed services.

ACID

See atomicity, consistency, isolation, durability.

active-active migration

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

active-passive migration

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

aggregate function

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

AI

See artificial intelligence.

AIOps

See artificial intelligence operations.

anonymization

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

anti-pattern

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

application control

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

application portfolio

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

artificial intelligence (AI)

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

artificial intelligence operations (AIOps)

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

asymmetric encryption

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

atomicity, consistency, isolation, durability (ACID)

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

attribute-based access control (ABAC)

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

authoritative data source

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

Availability Zone

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

AWS Cloud Adoption Framework (AWS CAF)

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

AWS Workload Qualification Framework (AWS WQF)

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

B

bad bot

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

BCP

See business continuity planning.

behavior graph

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

big-endian system

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

binary classification

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

bloom filter

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

blue/green deployment

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

bot

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

botnet

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

branch

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

break-glass access

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

brownfield strategy

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

buffer cache

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

business capability

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

business continuity planning (BCP)

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

C

CAF

See AWS Cloud Adoption Framework.

canary deployment

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

CCoE

See Cloud Center of Excellence.

CDC

See change data capture.

change data capture (CDC)

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

chaos engineering

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

CI/CD

See continuous integration and continuous delivery.

classification

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

client-side encryption

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

Cloud Center of Excellence (CCoE)

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

cloud computing

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

cloud operating model

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

cloud stages of adoption

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

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

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

CMDB

See configuration management database.

code repository

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

cold cache

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

cold data

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

computer vision (CV)

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

configuration drift

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

configuration management database (CMDB)

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

conformance pack

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

continuous integration and continuous delivery (CI/CD)

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

CV

See computer vision.

D

data at rest

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

data classification

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

data drift

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

data in transit

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

data mesh

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

data minimization

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

data perimeter

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

data preprocessing

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

data provenance

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

data subject

An individual whose data is being collected and processed.

data warehouse

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

database definition language (DDL)

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

database manipulation language (DML)

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

DDL

See database definition language.

deep ensemble

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

deep learning

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

defense-in-depth

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

delegated administrator

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

deployment

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

development environment

See environment.

detective control

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

development value stream mapping (DVSM)

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

digital twin

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

dimension table

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

disaster

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

disaster recovery (DR)

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

DML

See database manipulation language.

domain-driven design

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

DR

See disaster recovery.

drift detection

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

DVSM

See development value stream mapping.

E

EDA

See exploratory data analysis.

EDI

See electronic data interchange.

edge computing

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

electronic data interchange (EDI)

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

encryption

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

encryption key

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

endianness

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

endpoint

See service endpoint.

endpoint service

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

enterprise resource planning (ERP)

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

envelope encryption

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

environment

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

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

epic

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

ERP

See enterprise resource planning.

exploratory data analysis (EDA)

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

F

fact table

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

fail fast

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

fault isolation boundary

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

feature branch

See branch.

features

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

feature importance

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

feature transformation

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

few-shot prompting

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

FGAC

See fine-grained access control.

fine-grained access control (FGAC)

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

flash-cut migration

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

FM

See foundation model.

foundation model (FM)

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

G

generative AI

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

geo blocking

See geographic restrictions.

geographic restrictions (geo blocking)

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

Gitflow workflow

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

golden image

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

greenfield strategy

The absence of existing infrastructure in a new environment. When adopting a greenfield strategy for a system architecture, you can select all new technologies without the restriction

of compatibility with existing infrastructure, also known as brownfield. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

guardrail

A high-level rule that helps govern resources, policies, and compliance across organizational units (OUs). *Preventive guardrails* enforce policies to ensure alignment to compliance standards. They are implemented by using service control policies and IAM permissions boundaries. *Detective guardrails* detect policy violations and compliance issues, and generate alerts for remediation. They are implemented by using AWS Config, AWS Security Hub CSPM, Amazon GuardDuty, AWS Trusted Advisor, Amazon Inspector, and custom AWS Lambda checks.

H

HA

See high availability.

heterogeneous database migration

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

high availability (HA)

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

historian modernization

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

holdout data

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

homogeneous database migration

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

hot data

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

hotfix

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

hypercare period

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

I

IaC

See infrastructure as code.

identity-based policy

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

idle application

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

IIoT

See industrial Internet of Things.

immutable infrastructure

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

inbound (ingress) VPC

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

incremental migration

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

Industry 4.0

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

infrastructure

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

infrastructure as code (IaC)

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

industrial Internet of Things (IIoT)

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

inspection VPC

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

Internet of Things (IoT)

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

interpretability

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

IoT

See Internet of Things.

IT information library (ITIL)

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

IT service management (ITSM)

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

ITIL

See IT information library.

ITSM

See IT service management.

L

label-based access control (LBAC)

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

landing zone

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

large language model (LLM)

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

large migration

A migration of 300 or more servers.

LBAC

See label-based access control.

least privilege

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

lift and shift

See 7 Rs.

little-endian system

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

LLM

See large language model.

lower environments

See environment.

M

machine learning (ML)

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

main branch

See branch.

malware

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

managed services

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

manufacturing execution system (MES)

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

MAP

See Migration Acceleration Program.

mechanism

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

member account

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

MES

See manufacturing execution system.

Message Queuing Telemetry Transport (MQTT)

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

microservice

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

microservices architecture

An approach to building an application with independent components that run each application process as a microservice. These microservices communicate through a well-defined interface by using lightweight APIs. Each microservice in this architecture can be updated, deployed,

and scaled to meet demand for specific functions of an application. For more information, see Implementing microservices on AWS.

Migration Acceleration Program (MAP)

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

migration at scale

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

migration factory

Cross-functional teams that streamline the migration of workloads through automated, agile approaches. Migration factory teams typically include operations, business analysts and owners, migration engineers, developers, and DevOps professionals working in sprints. Between 20 and 50 percent of an enterprise application portfolio consists of repeated patterns that can be optimized by a factory approach. For more information, see the discussion of migration factories and the Cloud Migration Factory guide in this content set.

migration metadata

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

migration pattern

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

Migration Portfolio Assessment (MPA)

An online tool that provides information for validating the business case for migrating to the AWS Cloud. MPA provides detailed portfolio assessment (server right-sizing, pricing, TCO

comparisons, migration cost analysis) as well as migration planning (application data analysis and data collection, application grouping, migration prioritization, and wave planning). The MPA tool (requires login) is available free of charge to all AWS consultants and APN Partner consultants.

Migration Readiness Assessment (MRA)

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

migration strategy

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

ML

See machine learning.

modernization

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

modernization readiness assessment

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

monolithic applications (monoliths)

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

use a microservices architecture. For more information, see Decomposing monoliths into microservices.

MPA

See Migration Portfolio Assessment.

MQTT

See Message Queuing Telemetry Transport.

multiclass classification

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

mutable infrastructure

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

O

OAC

See origin access control.

OAI

See origin access identity.

OCM

See organizational change management.

offline migration

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

OI

See operations integration.

OLA

See operational-level agreement.

online migration

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

OPC-UA

See Open Process Communications - Unified Architecture.

Open Process Communications - Unified Architecture (OPC-UA)

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

operational-level agreement (OLA)

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

operational readiness review (ORR)

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

operational technology (OT)

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

operations integration (OI)

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

organization trail

A trail that's created by AWS CloudTrail that logs all events for all AWS accounts in an organization in AWS Organizations. This trail is created in each AWS account that's part of the

organization and tracks the activity in each account. For more information, see Creating a trail for an organization in the CloudTrail documentation.

organizational change management (OCM)

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

origin access control (OAC)

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

origin access identity (OAI)

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

ORR

See operational readiness review.

OT

See operational technology.

outbound (egress) VPC

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

P

permissions boundary

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

personally identifiable information (PII)

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

PII

See personally identifiable information.

playbook

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

PLC

See programmable logic controller.

PLM

See product lifecycle management.

policy

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

polyglot persistence

Independently choosing a microservice's data storage technology based on data access patterns and other requirements. If your microservices have the same data storage technology, they can encounter implementation challenges or experience poor performance. Microservices are more

easily implemented and achieve better performance and scalability if they use the data store best adapted to their requirements.

portfolio assessment

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

predicate

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

predicate pushdown

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

preventative control

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

principal

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

privacy by design

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

private hosted zones

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

proactive control

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

AWS Control Tower documentation and see Proactive controls in *Implementing security controls on AWS*.

product lifecycle management (PLM)

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

production environment

See environment.

programmable logic controller (PLC)

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

prompt chaining

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

pseudonymization

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

publish/subscribe (pub/sub)

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

Q

query plan

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

query plan regression

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

R

RACI matrix

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

RAG

See Retrieval Augmented Generation.

ransomware

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

RASCI matrix

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

RCAC

See row and column access control.

read replica

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

re-architect

See 7 Rs.

recovery point objective (RPO)

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

recovery time objective (RTO)

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

refactor

See 7 Rs.

Region

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

regression

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

rehost

See 7 Rs.

release

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

relocate

See 7 Rs.

replatform

See 7 Rs.

repurchase

See 7 Rs.

resiliency

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

resource-based policy

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

responsible, accountable, consulted, informed (RACI) matrix

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

responsive control

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

retain

See 7 Rs.

retire

See 7 Rs.

Retrieval Augmented Generation (RAG)

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

rotation

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

row and column access control (RCAC)

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

RPO

See recovery point objective.

RTO

See recovery time objective.

runbook

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

S

SAML 2.0

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

SCADA

See supervisory control and data acquisition.

SCP

See service control policy.

secret

In AWS Secrets Manager, confidential or restricted information, such as a password or user credentials, that you store in encrypted form. It consists of the secret value and its metadata. The secret value can be binary, a single string, or multiple strings. For more information, see What's in a Secrets Manager secret? in the Secrets Manager documentation.

security by design

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

security control

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

security hardening

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

security information and event management (SIEM) system

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

security response automation

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

server-side encryption

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

service control policy (SCP)

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

service endpoint

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

service-level agreement (SLA)

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

service-level indicator (SLI)

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

service-level objective (SLO)

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

shared responsibility model

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

SIEM

See security information and event management system.

single point of failure (SPOF)

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

SLA

See service-level agreement.

SLI

See service-level indicator.

SLO

See service-level objective.

split-and-seed model

A pattern for scaling and accelerating modernization projects. As new features and product releases are defined, the core team splits up to create new product teams. This helps scale your organization's capabilities and services, improves developer productivity, and supports rapid

innovation. For more information, see Phased approach to modernizing applications in the AWS Cloud.

SPOF

See single point of failure.

star schema

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

strangler fig pattern

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

subnet

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

supervisory control and data acquisition (SCADA)

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

symmetric encryption

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

synthetic testing

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

system prompt

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

T

tags

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

target variable

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

task list

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

test environment

See environment.

training

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

transit gateway

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

trunk-based workflow

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

trusted access

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

tuning

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

two-pizza team

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

U

uncertainty

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