Providing Credentials to the SDK

In order to authenticate requests, AWS services require you to provide your AWS access keys, also known as your AWS **access key ID** and **secret access key**.

There are many ways to provide credentials:

1. Using credentials from environment variables
2. Using IAM roles for Amazon EC2 instances
3. Using the AWS credentials file and credential profiles
4. Using a configuration file with the service builder
5. Passing credentials into a client factory method
6. Using temporary credentials from AWS STS

Which technique should you choose?

The technique that you use to provide credentials to the SDK for your application is entirely up to you. Please read each section on this page to determine what is the best fit for you. What you choose will depend on many different factors, including:

- The environment you are operating in (e.g., development, testing, production)
- The host of your application (e.g., localhost, Amazon EC2, third-party server)
- How many sets of credentials you are using
- The type of project you are developing (e.g., application, CLI, library)
- How often you rotate your credentials
- If you rely on temporary or federated credentials
- Your deployment process
- Your application framework

Regardless of the technique used, it is encouraged that you follow the IAM Best Practices when managing your credentials, including the recommendation to not use your AWS account's root credentials. Instead, create separate IAM users with their own access keys for each project, and tailor the permissions of the users specific to those projects.

In general, it is recommended that you use IAM roles when running your application on Amazon EC2 and use credential profiles or environment variables elsewhere.

Accessing IAM

You can work with AWS Identity and Access Management in any of the following ways.

AWS Management Console

The console is a browser-based interface to manage IAM and AWS resources. For more information about accessing IAM through the console, see The IAM Console and Sign-in Page. For a tutorial that guides you through using the console, see Creating Your First IAM Admin User and Group.

AWS Command Line Tools

You can use the AWS command line tools to issue commands at your system's command line to perform IAM and AWS tasks. Using the command line can be faster and more convenient than the console. The command line tools are also useful if you want to build scripts that perform AWS tasks.

AWS provides two sets of command line tools: the AWS Command Line Interface (AWS CLI) and the AWS Tools for Windows PowerShell. For information about installing and using the AWS CLI, see the AWS Command Line Interface User Guide. For information about installing and using the Tools for Windows PowerShell, see the AWS Tools for Windows PowerShell User Guide.

AWS SDKs

AWS provides SDKs (software development kits) that consist of libraries and sample code for various programming languages and platforms (Java, Python, Ruby, .NET, iOS, Android, etc.). The SDKs provide a convenient way to create programmatic access to IAM and AWS. For example, the SDKs take care of tasks such as cryptographically signing requests, managing errors, and retrying

requests automatically. For information about the AWS SDKs, including how to download and install them, see the Tools for Amazon Web Services page.

IAM HTTPS API

You can access IAM and AWS programmatically by using the IAM HTTPS API, which lets you issue HTTPS requests directly to the service. When you use the HTTPS API, you must include code to digitally sign requests using your credentials. For more information, see Calling the API by Making HTTP Query Requests and the IAM API Reference.

Using IAM roles for Amazon EC2 instances

Using IAM roles is the preferred technique for providing credentials to applications running on Amazon EC2. IAM roles remove the need to worry about credential management from your application. They allow an instance to "assume" a role by retrieving temporary credentials from the EC2 instance's metadata server. These temporary credentials, often referred to as **instance profile credentials**, allow access to the actions and resources that the role's policy allows.

When launching an EC2 instance, you can choose to associate it with an IAM role. Any application running on that EC2 instance is then allowed to assume the associated role. Amazon EC2 handles all the legwork of securely authenticating instances to the IAM service to assume the role and periodically refreshing the retrieved role credentials, keeping your application secure with almost no work on your part.

If you do not explicitly provide credentials to the client object and no environment variable credentials are available, the SDK attempts to retrieve instance profile credentials from an Amazon EC2 instance metadata server. These credentials are available only when running on Amazon EC2 instances that have been configured with an IAM role.

Note

Instance profile credentials and other temporary credentials generated by the AWS Security Token Service (AWS STS) are not supported by every service. Please check if the service you are using supports temporary credentials by reading AWS Services that Support AWS STS.

https://docs.aws.amazon.com/migrationhub/latest/ug/auth-and-access-explained.html#access-control-resources

- **Action** – You use action keywords to identify resource operations that you want to allow or deny. For example, you can use `mgh:AssociateDiscoveredResource` to allow the user permission to perform the Migration Hub `AssociateDiscoveredResource`operation.
- **Effect** – You specify the effect, either allow or deny, when the user requests the specific action. If you don't explicitly grant access to (allow) a resource, access is implicitly denied. You can also explicitly deny access to a resource, which you might do to make sure that a user cannot access it, even if a different policy grants access.
- **Principal** – In identity-based policies (IAM policies), the user that the policy is attached to is the implicit principal. For resource-based policies, you specify the user, account, service, or other entity that you want to receive permissions (applies to resource-based policies only). Migration Hub doesn't support resource-based policies.

AWS Systems Manager Parameter Store Secrets and configuration data management

Centralized storage and management of your secrets and configuration data such as passwords, database strings, and license codes. You can encrypt values, or store as plain text, and secure access at every level.

How it works

1. 1

   Create a new parameter

2. 2

   Specify the parameter type and values

3. 3

   Reference parameters in your commands or code

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_job-functions.html

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_job-functions.html#jf_administrator

https://docs.aws.amazon.com/IAM/latest/UserGuide/tutorial_billing.html

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_actions-resources-contextkeys.html

Actions, Resources, and Condition Keys for Amazon EC2

Amazon EC2 (service prefix: `ec2`) provides the following service-specific resources, actions, and condition context keys for use in IAM permission policies.

https://docs.aws.amazon.com/IAM/latest/UserGuide/list_amazonec2.html#amazonec2-ec2_Region

Getting Credential Reports for Your AWS Account

https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_getting-report.html

You can generate and download a *credential report* that lists all users in your account and the status of their various credentials, including passwords, access keys, and MFA devices. You can get a credential report from the AWS Management Console, the AWS SDKs and Command Line Tools, or the IAM API.

IAM JSON Policy Elements Reference

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements.html

IAM JSON Policy Elements: Action

The `Action` element describes the specific action or actions that will be allowed or denied. Statements must include either an `Action` or`NotAction` element.

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements_action.html

IAM JSON Policy Elements: NotAction

`NotAction` is an advanced policy element that explicitly matches everything *except* the specified list of actions. Using `NotAction` can result in a shorter policy by listing only a few actions that should not match, rather than including a long list of actions that will match. When using`NotAction`, you should keep in mind that actions specified in this element are the *only* actions in that are limited.

NotAction with Allow

You can use the `NotAction` element in a statement with `"Effect": "Allow"` to provide access to all of the actions in an AWS service, except for the actions specified in `NotAction`. You can use it with the `Resource` element to provide scope for the policy, limiting the allowed actions to the actions that can be performed on the specified resource.

Example:

The following example allows users to access all of the Amazon S3 actions that can be performed on any S3 resource *except* for deleting a bucket.

```
"Effect": "Allow",
"NotAction": "s3:DeleteBucket",
"Resource": "arn:aws:s3:::*",
```

The following example allows users to access every action in every AWS service except for IAM.

```
"Effect": "Allow",
"NotAction": "iam:*",
"Resource": "*"
```

Be careful using the `NotAction` element and `"Effect": "Allow"` in the same statement or in a different statement within a policy. `NotAction` matches all services and actions that are not explicitly listed or applicable to the specified resource, and could result in granting users more permissions than you intended.

NotAction with Deny

You can use the `NotAction` element in a statement with `"Effect": "Deny"` to deny access to all of the listed resources except for the actions specified in the `NotAction` element. This combination does not allow the listed items, but instead explicitly denies the actions not listed. You must still allow actions that you want to allow.

The following conditional example denies access to non-IAM actions if the user is not signed in using MFA. If the user is signed in with MFA, then the `"Condition"` test fails and the final `"Deny"` statement has no effect. Note, however, that this would not grant the user access to any actions; it would only explicitly deny all other actions except IAM actions.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "DenyAllOutsideEU",
        "Effect": "Deny",
        "NotAction": "iam:*",
        "Resource": "*",
        "Condition": {"BoolIfExists": {"aws:MultiFactorAuthPresent": "false"}}
    }]
}
```

The following example policy denies access to any operations outside of the `eu-central-1` and `eu-west-1` regions, except for actions in the listed services. The services listed in the `NotActions` element are some of the AWS global services with a single endpoint physically located in the `us-east-1` region. Operations in these services would fail otherwise. This policy denies access and requires another policy to grant access.

```
  {
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "DenyAllOutsideEU",
        "Effect": "Deny",
        "NotAction": [
            "aws-portal:*",
            "iam:*",
            "organizations:*",
```

```
            "support:*",
            "sts:*"
        ],
        "Resource": "*",
        "Condition": {"StringNotEquals": {"aws:RequestedRegion": [
            "eu-central-1",
            "eu-west-1"
        ]}}
    }]
```

IAM JSON Policy Elements: Effect

The `Effect` element is required and specifies whether the statement results in an allow or an explicit deny. Valid values for `Effect` are `Allow`and `Deny`.

```
"Effect":"Allow"
```

By default, access to resources is denied. To allow access to a resource, you must set the `Effect` element to `Allow`. To override an allow (for example, to override an allow that is otherwise in force), you set the `Effect` element

to `Deny`. For more information, see Policy Evaluation Logic■

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements_notaction.html

IAM JSON Policy Elements: Effect

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_elements_effect.html

Specifying Permissions in a Policy

Amazon S3 defines a set of permissions that you can specify in a policy. These are keywords, each of which maps to specific Amazon S3 operations (see Operations on Buckets, and Operations on Objects in the *Amazon Simple Storage Service API Reference*).

EC2:

https://docs.aws.amazon.com/AWSEC2/latest/APIReference/API_Operations.html

The following actions are supported:

- AcceptReservedInstancesExchangeQuote
- AcceptVpcEndpointConnections
- AcceptVpcPeeringConnection
- AllocateAddress
- AllocateHosts
- AssignIpv6Addresses
- AssignPrivateIpAddresses
- AssociateAddress
- AssociateDhcpOptions
- AssociateIamInstanceProfile
- AssociateRouteTable
- AssociateSubnetCidrBlock
- AssociateVpcCidrBlock
- AttachClassicLinkVpc
- AttachInternetGateway
- AttachNetworkInterface
- AttachVolume
- AttachVpnGateway
- AuthorizeSecurityGroupEgress
- AuthorizeSecurityGroupIngress
- BundleInstance
- CancelBundleTask
- CancelConversionTask
- CancelExportTask
- CancelImportTask
- CancelReservedInstancesListing
- CancelSpotFleetRequests
- CancelSpotInstanceRequests
- ConfirmProductInstance
- CopyFpgaImage

- CopyImage
- CopySnapshot
- CreateCustomerGateway
- CreateDefaultSubnet
- CreateDefaultVpc
- CreateDhcpOptions
- CreateEgressOnlyInternetGateway
- CreateFleet
- CreateFlowLogs
- CreateFpgaImage
- CreateImage
- CreateInstanceExportTask
- CreateInternetGateway
- CreateKeyPair
- CreateLaunchTemplate
- CreateLaunchTemplateVersion
- CreateNatGateway
- CreateNetworkAcl
- CreateNetworkAclEntry
- CreateNetworkInterface
- CreateNetworkInterfacePermission
- CreatePlacementGroup
- CreateReservedInstancesListing
- CreateRoute
- CreateRouteTable
- CreateSecurityGroup
- CreateSnapshot
- CreateSpotDatafeedSubscription
- CreateSubnet
- CreateTags
- CreateVolume
- CreateVpc
- CreateVpcEndpoint
- CreateVpcEndpointConnectionNotification
- CreateVpcEndpointServiceConfiguration
- CreateVpcPeeringConnection

- CreateVpnConnection
- CreateVpnConnectionRoute
- CreateVpnGateway
- DeleteCustomerGateway
- DeleteDhcpOptions
- DeleteEgressOnlyInternetGateway
- DeleteFleets
- DeleteFlowLogs
- DeleteFpgaImage
- DeleteInternetGateway
- DeleteKeyPair
- DeleteLaunchTemplate
- DeleteLaunchTemplateVersions
- DeleteNatGateway
- DeleteNetworkAcl
- DeleteNetworkAclEntry
- DeleteNetworkInterface
- DeleteNetworkInterfacePermission
- DeletePlacementGroup
- DeleteRoute
- DeleteRouteTable
- DeleteSecurityGroup
- DeleteSnapshot
- DeleteSpotDatafeedSubscription
- DeleteSubnet
- DeleteTags
- DeleteVolume
- DeleteVpc
- DeleteVpcEndpointConnectionNotifications
- DeleteVpcEndpoints
- DeleteVpcEndpointServiceConfigurations
- DeleteVpcPeeringConnection
- DeleteVpnConnection
- DeleteVpnConnectionRoute
- DeleteVpnGateway
- DeregisterImage

- DescribeAccountAttributes
- DescribeAddresses
- DescribeAggregateIdFormat
- DescribeAvailabilityZones
- DescribeBundleTasks
- DescribeClassicLinkInstances
- DescribeConversionTasks
- DescribeCustomerGateways
- DescribeDhcpOptions
- DescribeEgressOnlyInternetGateways
- DescribeElasticGpus
- DescribeExportTasks
- DescribeFleetHistory
- DescribeFleetInstances
- DescribeFleets
- DescribeFlowLogs
- DescribeFpgaImageAttribute
- DescribeFpgaImages
- DescribeHostReservationOfferings
- DescribeHostReservations
- DescribeHosts
- DescribeIamInstanceProfileAssociations
- DescribeIdentityIdFormat
- DescribeIdFormat
- DescribeImageAttribute
- DescribeImages
- DescribeImportImageTasks
- DescribeImportSnapshotTasks
- DescribeInstanceAttribute
- DescribeInstanceCreditSpecifications
- DescribeInstances
- DescribeInstanceStatus
- DescribeInternetGateways
- DescribeKeyPairs
- DescribeLaunchTemplates
- DescribeLaunchTemplateVersions

- DescribeMovingAddresses
- DescribeNatGateways
- DescribeNetworkAcls
- DescribeNetworkInterfaceAttribute
- DescribeNetworkInterfacePermissions
- DescribeNetworkInterfaces
- DescribePlacementGroups
- DescribePrefixLists
- DescribePrincipalIdFormat
- DescribeRegions
- DescribeReservedInstances
- DescribeReservedInstancesListings
- DescribeReservedInstancesModifications
- DescribeReservedInstancesOfferings
- DescribeRouteTables
- DescribeScheduledInstanceAvailability
- DescribeScheduledInstances
- DescribeSecurityGroupReferences
- DescribeSecurityGroups
- DescribeSnapshotAttribute
- DescribeSnapshots
- DescribeSpotDatafeedSubscription
- DescribeSpotFleetInstances
- DescribeSpotFleetRequestHistory
- DescribeSpotFleetRequests
- DescribeSpotInstanceRequests
- DescribeSpotPriceHistory
- DescribeStaleSecurityGroups
- DescribeSubnets
- DescribeTags
- DescribeVolumeAttribute
- DescribeVolumes
- DescribeVolumesModifications
- DescribeVolumeStatus
- DescribeVpcAttribute
- DescribeVpcClassicLink

- DescribeVpcClassicLinkDnsSupport
- DescribeVpcEndpointConnectionNotifications
- DescribeVpcEndpointConnections
- DescribeVpcEndpoints
- DescribeVpcEndpointServiceConfigurations
- DescribeVpcEndpointServicePermissions
- DescribeVpcEndpointServices
- DescribeVpcPeeringConnections
- DescribeVpcs
- DescribeVpnConnections
- DescribeVpnGateways
- DetachClassicLinkVpc
- DetachInternetGateway
- DetachNetworkInterface
- DetachVolume
- DetachVpnGateway
- DisableVgwRoutePropagation
- DisableVpcClassicLink
- DisableVpcClassicLinkDnsSupport
- DisassociateAddress
- DisassociateIamInstanceProfile
- DisassociateRouteTable
- DisassociateSubnetCidrBlock
- DisassociateVpcCidrBlock
- EnableVgwRoutePropagation
- EnableVolumeIO
- EnableVpcClassicLink
- EnableVpcClassicLinkDnsSupport
- GetConsoleOutput
- GetConsoleScreenshot
- GetHostReservationPurchasePreview
- GetLaunchTemplateData
- GetPasswordData
- GetReservedInstancesExchangeQuote
- ImportImage
- ImportInstance

- ImportKeyPair
- ImportSnapshot
- ImportVolume
- ModifyFleet
- ModifyFpgaImageAttribute
- ModifyHosts
- ModifyIdentityIdFormat
- ModifyIdFormat
- ModifyImageAttribute
- ModifyInstanceAttribute
- ModifyInstanceCreditSpecification
- ModifyInstancePlacement
- ModifyLaunchTemplate
- ModifyNetworkInterfaceAttribute
- ModifyReservedInstances
- ModifySnapshotAttribute
- ModifySpotFleetRequest
- ModifySubnetAttribute
- ModifyVolume
- ModifyVolumeAttribute
- ModifyVpcAttribute
- ModifyVpcEndpoint
- ModifyVpcEndpointConnectionNotification
- ModifyVpcEndpointServiceConfiguration
- ModifyVpcEndpointServicePermissions
- ModifyVpcPeeringConnectionOptions
- ModifyVpcTenancy
- MonitorInstances
- MoveAddressToVpc
- PurchaseHostReservation
- PurchaseReservedInstancesOffering
- PurchaseScheduledInstances
- RebootInstances
- RegisterImage
- RejectVpcEndpointConnections
- RejectVpcPeeringConnection

- ReleaseAddress
- ReleaseHosts
- ReplaceIamInstanceProfileAssociation
- ReplaceNetworkAclAssociation
- ReplaceNetworkAclEntry
- ReplaceRoute
- ReplaceRouteTableAssociation
- ReportInstanceStatus
- RequestSpotFleet
- RequestSpotInstances
- ResetFpgaImageAttribute
- ResetImageAttribute
- ResetInstanceAttribute
- ResetNetworkInterfaceAttribute
- ResetSnapshotAttribute
- RestoreAddressToClassic
- RevokeSecurityGroupEgress
- RevokeSecurityGroupIngress
- RunInstances
- RunScheduledInstances
- StartInstances
- StopInstances
- TerminateInstances
- UnassignIpv6Addresses
- UnassignPrivateIpAddresses
- UnmonitorInstances
- UpdateSecurityGroupRuleDescriptionsEgress
- UpdateSecurityGroupRuleDescriptionsIngress

https://docs.aws.amazon.com/AmazonS3/latest/dev/using-with-s3-actions.html

You can use credential reports to assist in your auditing and compliance efforts. You can use the report to audit the effects of credential lifecycle requirements, such as password and access key rotation. You can provide the report to an external auditor, or grant permissions to an auditor so that he or she can download the report directly.

You can generate a credential report as often as once every four hours. When you request a report, IAM first checks whether a report for the AWS account has been generated within the past four hours. If so, the most recent report is downloaded. If the most recent report for the account is older than four hours, or if there are no previous reports for the account, IAM generates and downloads a new report.

Topics

Getting Credential Reports (Console)

You can use the AWS Management Console to download a credential report as a comma-separated values (CSV) file.

To download a credential report (console)

1. Sign in to the AWS Management Console and open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation pane, click **Credential report**.
3. Click **Download Report**.

IAM JSON Policy Reference

This section presents detailed syntax, descriptions, and examples of the elements, variables, and evaluation logic of JSON policies in IAM. It includes the following sections.

- IAM JSON Policy Elements Reference — Learn more about the elements that you can use when you create a policy. View additional policy examples and learn about conditions, supported data types, and how they are used in various services.
- Policy Evaluation Logic — This section describes AWS requests, how they are authenticated, and how AWS uses policies to determine access to resources.
- Grammar of the IAM JSON Policy Language — This section presents a formal grammar for the language that is used to create policies in IAM.
- AWS Managed Policies for Job Functions — This section lists all the AWS managed policies that directly map to common job functions in the IT industry. Use these policies to grant the permissions that are needed to carry out the tasks expected of someone in a specific job function. These policies consolidate permissions for many services into a single policy.
- AWS Global Condition Context Keys — This section includes a list of all the AWS global condition keys that you can use to limit permissions in an IAM policy.
- IAM Condition Context Keys — This section includes a list of all the IAM and AWS STS condition keys that you can use to limit permissions in an IAM policy.
- Actions, Resources, and Condition Keys for AWS Services — This section presents a list of all the AWS API operations that you can use as permissions in an IAM policy. It also includes the service-specific condition keys that can be used to further refine the request.

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies.html

Policy Evaluation Logic

When a principal tries to use the AWS Management Console, the AWS API, or the AWS CLI, that principal sends a *request* to AWS. When an AWS service receives the request, AWS completes several steps to determine whether to allow or deny the request.

1. **Authentication** – AWS first authenticates the principal that makes the request, if necessary. This step is not necessary for a few services, such as Amazon S3, that allow some requests from anonymous users.
2. **Processing the Request Context** – AWS processes the information gathered in the request to determine which policies apply to the request.
3. **Evaluating Policies** – AWS evaluates all of the policy types, which affect the order in which the policies are evaluated.
4. **Determining Whether a Request Is Allowed or Denied** – AWS then processes the policies against the request context to determine whether the request is allowed or denied.

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_evaluation-logic.html

****

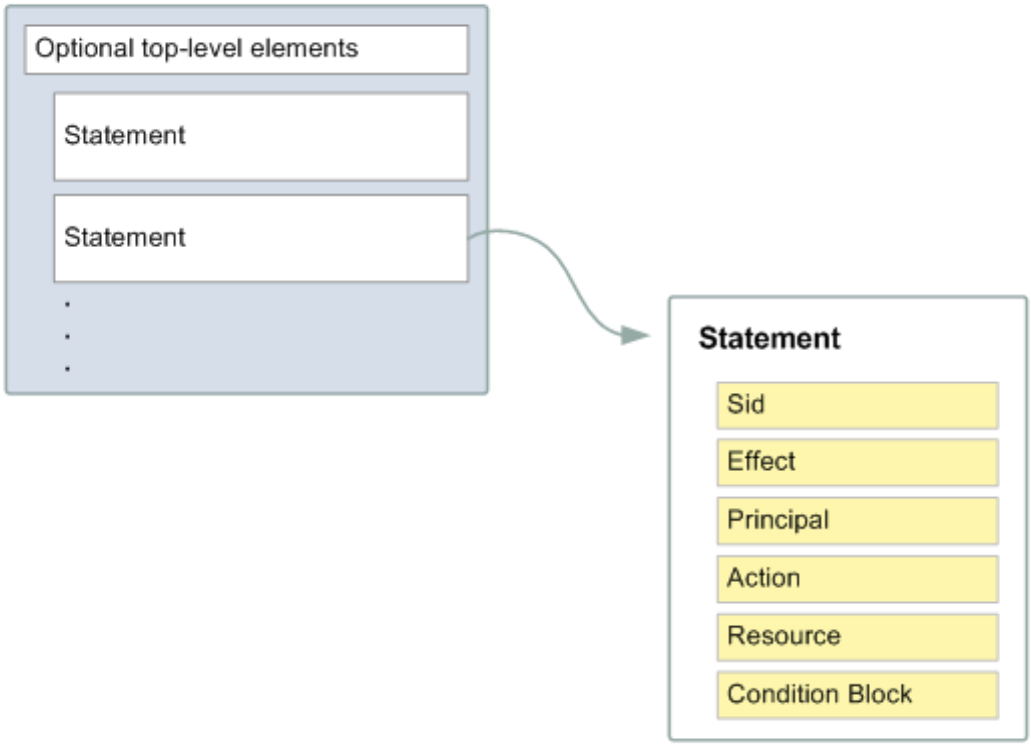
https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html#access_policy-types

JSON Policy Document Structure

As illustrated in the following figure, a JSON policy document includes these elements:

- Optional policywide information at the top of the document
- One or more individual statements

Each statement includes information about a single permission. If a policy includes multiple statements, AWS applies a logical OR across the statements when evaluating them. If multiple policies apply to a request, AWS applies a logical OR across all of those policies when evaluating them.



The information in a statement is contained within a series of elements.

- **Version** – Specify the version of the policy language that you want to use. As a best practice, use the latest `2012-10-17` version.
- **Statement** – Use this main policy element as a container for the following elements. You can include more than one statement in a policy.
- **Sid** – Include an optional statement ID to differentiate between your statements.
- **Effect** – Use `Allow` or `Deny` to indicate whether the policy allows or denies access.
- **Principal** – Indicate the account, user, role, or federated user to which you would like to allow or deny access. If you are creating a policy to attach to a user or role, you cannot include this element. The principal is implied as that user or role.
- **Action** – Include a list of actions that the policy allows or denies.
- **Resource** – Specify a list of resources to which the actions apply.
- **Condition** (Optional) – Specify the circumstances under which the policy grants permission.

Multiple Statements and Multiple Policies

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FirstStatement",
      "Effect": "Allow",
      "Action": ["iam:ChangePassword"],
      "Resource": "*"
    },
    {
      "Sid": "SecondStatement",
      "Effect": "Allow",
      "Action": "s3:ListAllMyBuckets",
      "Resource": "*"
    },
    {
      "Sid": "ThirdStatement",
      "Effect": "Allow",
      "Action": [
        "s3:List*",
        "s3:Get*"
      ],
      "Resource": [
        "arn:aws:s3:::confidential-data",
        "arn:aws:s3:::confidential-data/*"
      ],
      "Condition": {"Bool": {"aws:MultiFactorAuthPresent": "true"}}
    }
  ]
}
```

Use Roles for Applications That Run on Amazon EC2 Instances

Applications that run on an Amazon EC2 instance need credentials in order to access other AWS services. To provide credentials to the application in a secure way, use IAM *roles*. A role is an entity that has its own set of permissions, but that isn't a user or group. Roles also don't have their own permanent set of credentials the way IAM users do. In the case of Amazon EC2, IAM dynamically provides temporary credentials to the EC2 instance, and these credentials are automatically rotated for you.

When you launch an EC2 instance, you can specify a role for the instance as a launch parameter. Applications that run on the EC2 instance can use the role's credentials

when they access AWS resources. The role's permissions determine what the application is allowed to do.

For more information, see Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances.

https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#use-roles-with-ec2

Topics

- How Do Roles for EC2 Instances Work?
- Permissions Required for Using Roles with Amazon EC2
- How Do I Get Started?
- Related Information
- Using Instance Profiles

https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_switch-role-ec2.html

****

Controlling Access to Amazon EC2 Resources

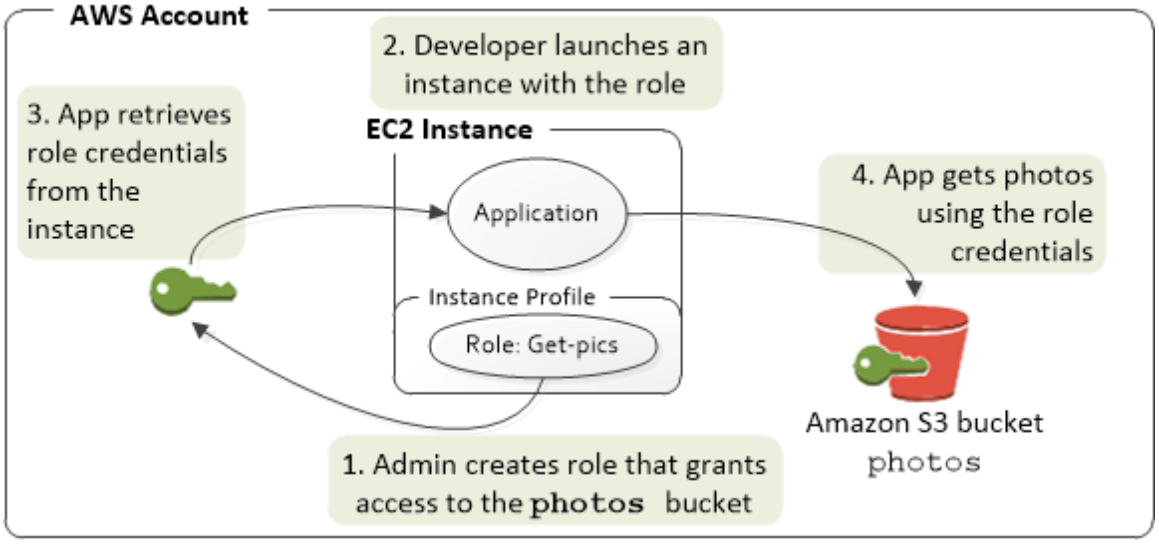
https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/UsingIAM.html#UsingIAMrolesWithAmazonEC2Instances

Your security credentials identify you to services in AWS and grant you unlimited use of your AWS resources, such as your Amazon EC2 resources. You can use features of Amazon EC2 and AWS Identity and Access Management (IAM) to allow other users, services, and applications to use your Amazon EC2 resources without sharing your security credentials. You can use IAM to control how other users use resources in your AWS account, and you can use security groups to control access to your Amazon EC2 instances. You can choose to allow full use or limited use of your Amazon EC2 resources.

Contents

Using an IAM Role to Grant Permissions to Applications Running on Amazon EC2 Instances

https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_switch-role-ec2.html#roles-usingrole-ec2instance-permissions



IAM Policies for Amazon EC2

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/iam-policies-for-amazon-ec2.html

Getting Started

An IAM policy must grant or deny permissions to use one or more Amazon EC2 actions. It must also specify the resources that can be used with the action, which can be all resources, or in some cases, specific resources. The policy can also include conditions that you apply to the resource.

Amazon EC2 partially supports resource-level permissions. This means that for some EC2 API actions, you cannot specify which resource a user is allowed to work with for that action; instead, you have to allow users to work with all resources for that action.

Task	Topic
Understand the basic structure of a policy	Policy Syntax
Define actions in your policy	Actions for Amazon EC2
Define specific resources in your policy	Amazon Resource Names for Amazon EC2
Apply conditions to the use of the resources	Condition Keys for Amazon EC2
Work with the available resource-level permissions for Amazon EC2	Supported Resource-Level Permissions for Amazon EC2 API Actions
Test your policy	Checking That Users Have the Required Permissions
Example policies for a CLI or SDK	Example Policies for Working with the AWS CLI or an AWS SDK
Example policies for the Amazon EC2 console	Example Policies for Working in the Amazon EC2 Console

Example Policies: Amazon EC2

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_examples.html (IMP)

- Allows an Amazon EC2 instance to attach or detach volumes (View this policy)
- Allows attaching or detaching Amazon EBS volumes to Amazon EC2 instances based on tags (View this policy)
- Allows launching Amazon EC2 instances in a specific subnet, programmatically and in the console (View this policy)
- Allows managing Amazon EC2 security groups associated with a specific VPC, programmatically and in the console (View this policy)
- Allows starting or stopping Amazon EC2 instances a user has tagged, programmatically and in the console (View this policy)
- Allows full Amazon EC2 access within a specific region, programmatically and in the console (View this policy)
- Allows starting or stopping a specific Amazon EC2 instance and modifying a specific security group, programmatically and in the console (View this policy)
- Limits terminating Amazon EC2 instances to a specific IP address range (View this policy)

Example Policies: AWS Identity and Access Management (IAM)

- Allows access to the policy simulator API (View this policy)
- Allows access to the policy simulator console (View this policy)
- Allows using the policy simulator API for users with a specific path (View this policy)
- Allows using the policy simulator console for users with a specific path (View this policy)
- Allows IAM users to self-manage an MFA device (View this policy)
- Allows IAM users to rotate their own credentials, programmatically and in the console (View this policy)
- Limits managed policies that can be applied to a new IAM user, group, or role (View this policy)

Example Policies: Amazon RDS

- Allows full Amazon RDS database access within a specific region (View this policy)
- Allows restoring Amazon RDS databases, programmatically and in the console (View this policy)

- Allows tag owners full access to Amazon RDS resources that they have tagged (View this policy)

Example Policies: Amazon S3

- Allows an Amazon Cognito user to access objects in their own Amazon S3 bucket (View this policy)
- Allows IAM users to access their own home directory in Amazon S3, programmatically and in the console (View this policy)
- Allows a user to manage a single Amazon S3 bucket and denies every other AWS action and resource (View this policy)
- Allows `Read` and `Write` access to a specific Amazon S3 bucket (View this policy)
- Allows `Read` and `Write` access to a specific Amazon S3 bucket, programmatically and in the console (View this policy)

Retrieving Security Credentials from Instance Metadata

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/iam-roles-for-amazon-ec2.html#instance-metadata-security-credentials

https://aws.amazon.com/documentation/

Guides and API References

Compute
AWS BatchAmazon EC2Amazon ECRAmazon ECSAmazon EKSAWS Elastic BeanstalkAWS LambdaAmazon LightsailAWS Serverless Application RepositoryAmazon VPC

Storage
Amazon EBSAmazon EFSAmazon GlacierAmazon S3AWS SnowballAWS Storage Gateway

Database
Amazon DynamoDBAmazon ElastiCacheAmazon NeptuneAmazon RDSAmazon Redshift

Migration
AWS Application Discovery ServiceAWS Database Migration ServiceAWS Migration HubAWS Schema Conversion ToolAWS Server Migration ServiceAWS Snowball

Networking & Content Delivery
Amazon API GatewayAmazon CloudFrontAWS Direct ConnectElastic Load BalancingAmazon Route 53Amazon VPC

Application Integration
Amazon MQAmazon SNSAmazon SQSAWS Step FunctionsAmazon SWF

Developer Tools
AWS Cloud9AWS CodeBuildAWS CodeCommitAWS CodeDeployAWS CodePipelineAWS CodeStarAWS Tools & SDKsAWS X-Ray

Management Tools
AWS Auto ScalingAWS CloudFormationAWS CloudTrailAmazon CloudWatchAWS Command Line InterfaceAWS ConfigAmazon Data Lifecyle ManagerAWS HealthAWS Management ConsoleAWS OpsWorksAWS Service CatalogAWS Systems ManagerAWS Tools for Windows PowershellTrusted Advisor

Media Services
Amazon Elastic TranscoderAWS Elemental MediaConvertAWS Elemental MediaLiveAWS Elemental MediaPackageAWS Elemental MediaStoreAWS Elemental MediaTailor

Machine Learning
Apache MXNet on AWSAmazon ComprehendAWS Deep Learning AMIsAWS DeepLensAmazon LexAmazon Machine LearningAmazon PollyAmazon RekognitionAmazon SageMakerAmazon TranscribeAmazon Translate

Internet of Things
Amazon FreeRTOSAWS GreengrassAWS IoT 1-ClickAWS IoT AnalyticsAWS IoT CoreAWS IoT Device DefenderAWS IoT Device Management

Analytics
Amazon AthenaAmazon CloudSearchAWS Data PipelineAmazon Elasticsearch ServiceAmazon EMRAWS GlueAmazon KinesisAmazon QuickSightAmazon Redshift

Security, Identity, & Compliance
AWS ArtifactAWS Certificate ManagerAWS CloudHSMAmazon CognitoAWS Crypto ToolsAWS Directory ServiceAWS Firewall ManagerAmazon Cloud DirectoryAmazon GuardDutyIdentity & Access ManagementAmazon InspectorAWS Key Management ServiceAmazon MacieAWS OrganizationsAWS Secrets ManagerAWS ShieldAWS Single Sign-OnAWS WAF

Mobile Services
AWS AppSyncAWS Device FarmAmazon Mobile AnalyticsAWS Mobile HubAWS Mobile SDK for AndroidAWS Mobile SDK for iOSAWS Mobile SDK for UnityAWS Mobile SDK for XamarinAmazon PinpointAmazon SNS

Desktop & App Streaming
Amazon AppStream 2.0Amazon WAMAmazon WorkSpacesNICE Desktop Cloud Visualization

Business Productivity
Alexa for BusinessAmazon ChimeAmazon WorkDocsAmazon WorkMail

AR & VR
Amazon Sumerian

Customer Engagement
Amazon ConnectAmazon PinpointAmazon Simple Email Service (SES)

Game Development
Amazon GameLiftAmazon Lumberyard (Beta)

SDKs & Toolkits

AWS Crypto ToolsAWS Guide for .NET DevelopersAWS SDK for C++AWS SDK for GoAWS SDK for JavaAWS SDK for JavaScriptAWS SDK for .NETAWS SDK for PHPAWS SDK for Python (Boto 3)AWS SDK for RubyAWS Toolkit for EclipseAWS Toolkit for Visual StudioAWS Tools for Visual Studio Team Services

General Reference

ARNs & Service NamespacesAWS GlossaryRegions and EndpointsSecurity CredentialsService Limits

Additional References

Alexa Top SitesAlexa Web Information ServiceAWS Billing and Cost ManagementAWS Blockchain TemplatesAWS General ReferenceAWS GovCloud (US)AWS MarketplaceAWS Quick StartsAmazon Silk

AWS Management Console

Resource GroupsResource Groups Tagging APITag Editor