# AWS IAM

## All you need to know

Documented by Alamgir Hosen

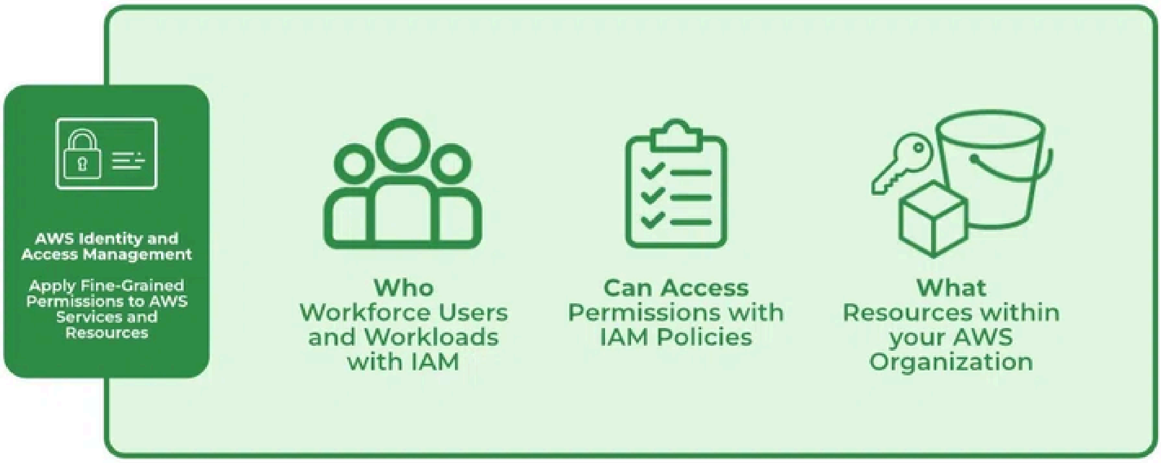
# Concept Overview:

# About IAM:

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. With IAM, you can manage permissions that control which AWS resources users can access. You use IAM to control who is authenticated (signed in) and authorized (has permissions) to use resources. IAM provides the infrastructure necessary to control authentication and authorization for your AWS accounts.

For better understanding follow the principle:

## Who->Can Access->What


AWS Identity and Access Management
Apply Fine-Grained Permissions to AWS Services and Resources
Who
Workforce Users and Workloads with IAM
Can Access
Permissions with IAM Policies
What
Resources within your AWS Organization

## About Access Control:

Access control in AWS IAM is the mechanism that determines who can perform which actions on what resources, under what conditions.

## IAM User & Group:

### IAM User

An IAM user is an entity that you create in your AWS account. The IAM user represents the human user or workload who uses the IAM user to interact with AWS resources. An IAM user consists of a name and credentials.

### IAM Group

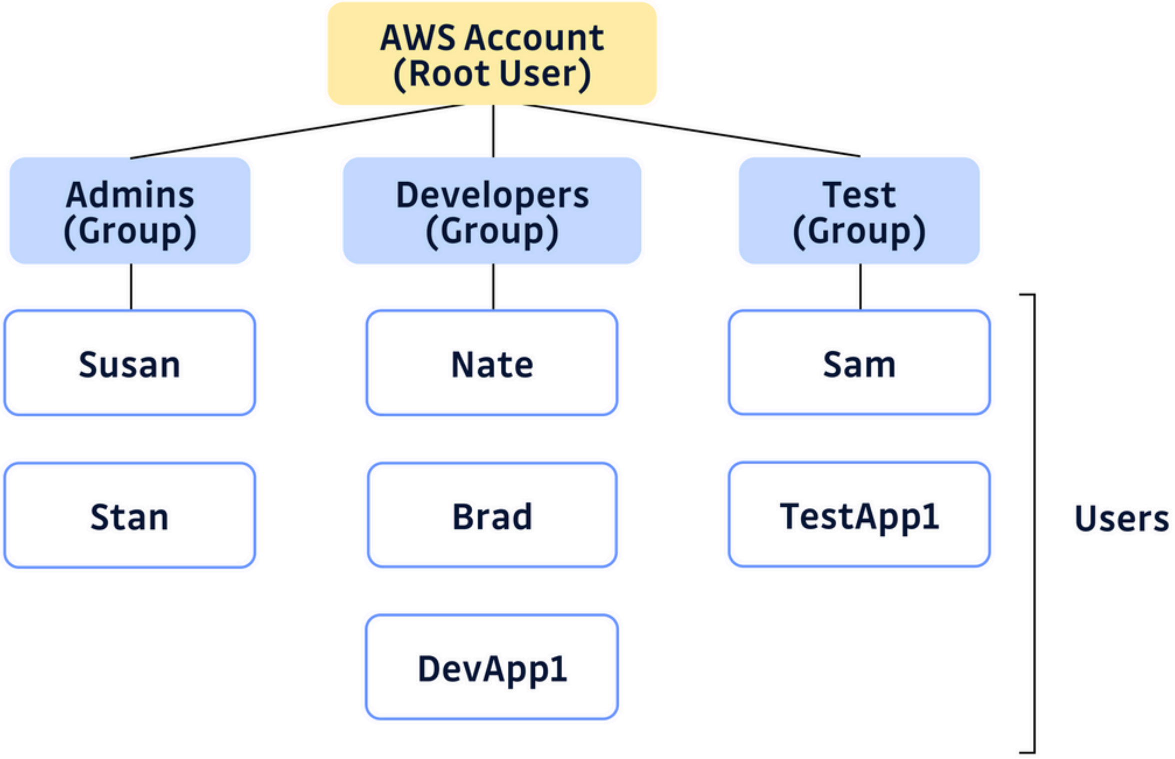
An IAM user group is a collection of IAM users that share the same permissions.

For example, an Admins group can have administrator rights, and any user added to it automatically gets those rights.

When someone changes roles, you can simply move them between groups instead of editing individual permissions.

# IAM User & Group:

**The following diagram shows the user and group example.**

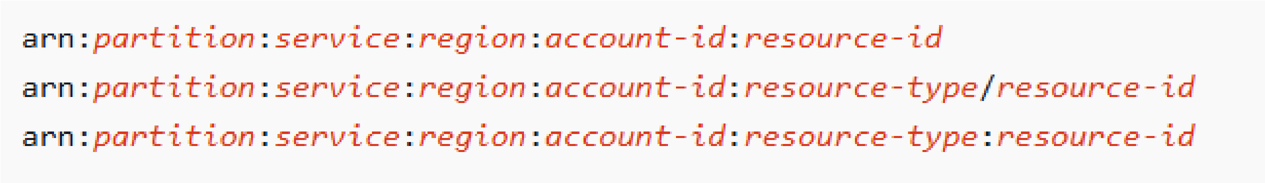


# Understanding ARN:

Understanding ARN before deep dive into IAM rules and policy.

### What is ARN & it’s format:

An AWS ARN (Amazon Resource Name) is a unique identifier for resources within the Amazon Web Services (AWS) cloud.



```
arn:partition:service:region:account-id:resource-id
arn:partition:service:region:account-id:resource-type/resource-id
arn:partition:service:region:account-id:resource-type:resource-id
```

- partition: Identifies the AWS partition
- service: Specifies the AWS service
- region: Indicates the AWS region
- account-id: Refers to the AWS account number.
- resource-id: Contains details specific to the resource, such as the bucket name for S3 or the instance ID for EC2.

# IAM Roles:

## IAM role:

An IAM role is an IAM identity that you can create in your account that has specific permissions. An IAM role is similar to an IAM user, in that it is an AWS identity with permission policies that determine what the identity can and cannot do in AWS.

## How it works:

Assume Role: A trusted entity (user, service, or app) requests to assume the role.

Temporary Credentials Issued: AWS grants short-lived credentials for the session.

Execute Actions: The entity performs tasks allowed by the role's permissions.

Expiration: Credentials expire when the session ends or after a set duration.

# IAM Policy & Permissions:

### IAM Policy:

An IAM policy is a JSON document that specifies permissions. Policies can be reused with different services in AWS. The same policy can be assigned to different people and teams.

### Policy Types:

- Identity-based policies – Attach managed and inline policies to IAM identities (users, groups to which users belong, or roles). Identity-based policies grant permissions to an identity.

- Resource-based policies – Attach inline policies to resources. The most common examples of resource-based policies are Amazon S3 bucket policies and IAM role trust policies. Resource-based policies grant permissions to the principal that is specified in the policy. Principals can be in the same account as the resource or in other accounts.

# IAM Policy & Permissions:

### Policy Types:

- Permissions boundaries – Use a managed policy as the permissions boundary for an IAM entity (user or role). That policy defines the maximum permissions that the identity-based policies can grant to an entity, but does not grant permissions. Permissions boundaries do not define the maximum permissions that a resource-based policy can grant to an entity.

- AWS Organizations SCPs – Use an AWS Organizations service control policy (SCP) to define the maximum permissions for IAM users and IAM roles within accounts in your organization or organizational unit (OU). SCPs limit permissions that identity-based policies or resource-based policies grant to IAM users or IAM roles within the account. SCPs do not grant permissions.

# IAM Policy & Permissions:

### Policy Types:

- AWS Organizations RCPs – Use an AWS Organizations resource control policy (RCP) to define the maximum permissions for resources within accounts in your organization or organizational unit (OU). RCPs limit permissions that identity-based and resource-based policies can grant to resources in accounts within your organization. RCPs do not grant permissions.
- Access control lists (ACLs) – Use ACLs to control which principals in other accounts can access the resource to which the ACL is attached. ACLs are similar to resource-based policies, although they are the only policy type that does not use the JSON policy document structure. ACLs are cross-account permissions policies that grant permissions to the specified principal. ACLs cannot grant permissions to entities within the same account.
- Session policies – Pass advanced session policies when you use the AWS CLI or AWS API to assume a role or a federated user. Session policies limit the permissions that the role or user's identity-based policies grant to the session. Session policies limit permissions for a created session, but do not grant permissions.

# IAM Policy & Permissions:

## Permissions in AWS IAM Policies:

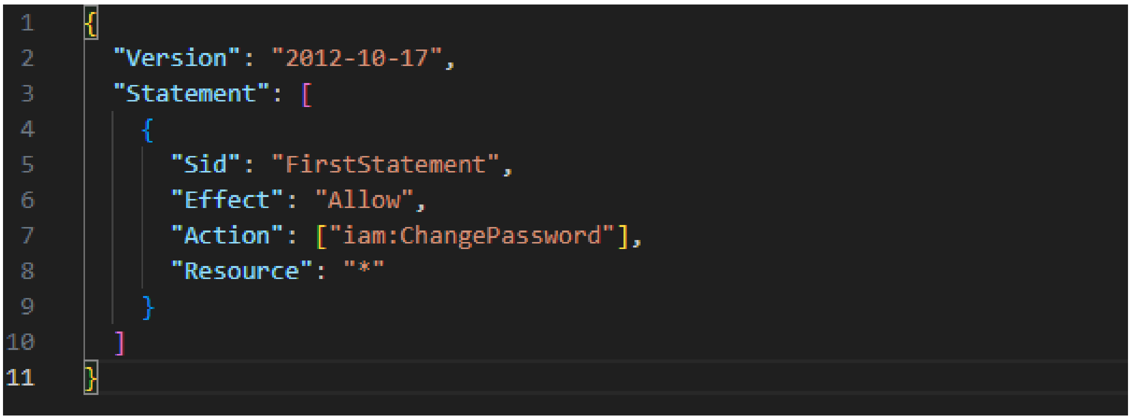
Permissions within an IAM policy define what actions are allowed or denied for specific AWS resources. The policy includes the following elements:

- Actions: Specifies the API operations (like s3:PutObject or ec2:StartInstances) that the policy allows or denies.
- Resources: Defines the specific AWS resources the policy applies to (like an S3 bucket or EC2 instance).
- Effect: Determines whether the action is allowed (Allow) or denied (Deny).
- Conditions: Optional, but they add an additional layer of control by specifying conditions under which the policy is in effect (for example, restricting actions based on the requester's IP address or the time of day).

# IAM Policy & Permissions:

## IAM Policy Document Structure:

To fully understand the structure of an IAM policy, let us see the default template provided by AWS and look at all the fields one by one.



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "FirstStatement",
      "Effect": "Allow",
      "Action": ["iam:ChangePassword"],
      "Resource": "*"
    }
  ]
}
```

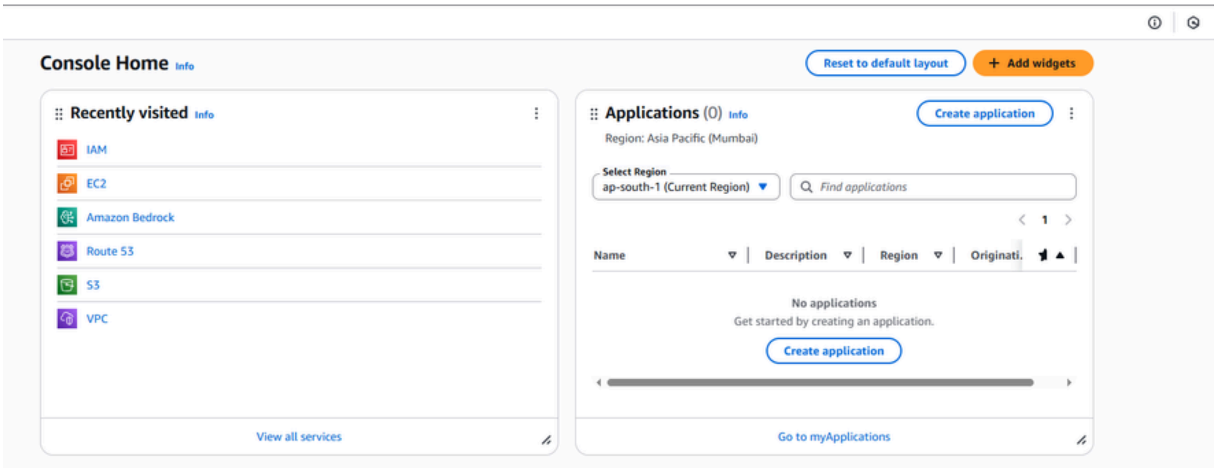
- Version-ID: This is a compulsory field in a policy that is uniquely used to identify a JSON policy.
- Statement: It defines the permission for a single resource.
- Sid: Is short for statement id. It is a unique identifier for a statement.
- Effect: It defines the Allow/Deny prospects to a resource.
- Action: This is used to state what service can perform what all actions.
- Resource: It consists of the list of resources that are impacted by the policy.

# IAM Policy & Permissions:

## Create policy step by step:

### Step: 1 - Log in to your AWS account and go to the IAM section

Dashboard



Console Home Info

Reset to default layout

+ Add widgets

Recently visited Info

- IAM
- EC2
- Amazon Bedrock
- Route 53
- S3
- VPC

View all services

Applications (0) Info

Create application

Region: Asia Pacific (Mumbai)

Select Region: ap-south-1 (Current Region)

Find applications

| Name | Description | Region | Originati... |
|---|---|---|---|

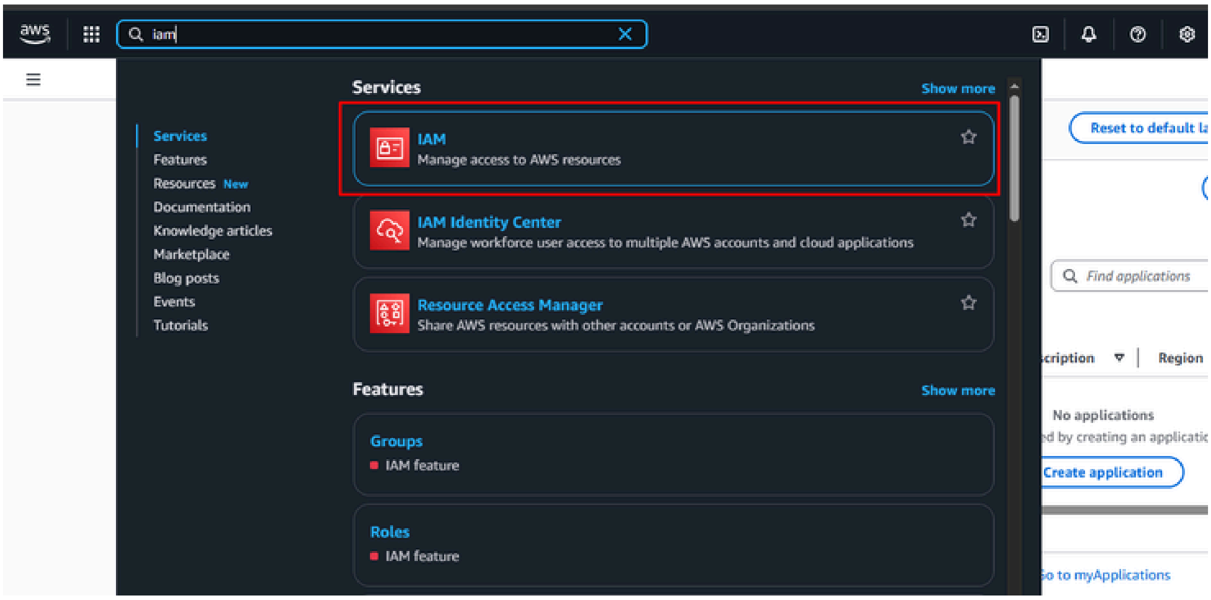
No applications

Get started by creating an application.

Create application

Go to myApplications

If you don’t see the IAM service in your dashboard then search for IAM. In my case it’s show becuase i already use it.



iam

- Services
- Features
- Resources New
- Documentation
- Knowledge articles
- Marketplace
- Blog posts
- Events
- Tutorials

Services Show more

IAM
Manage access to AWS resources

IAM Identity Center
Manage workforce user access to multiple AWS accounts and cloud applications

Resource Access Manager
Share AWS resources with other accounts or AWS Organizations

Features Show more

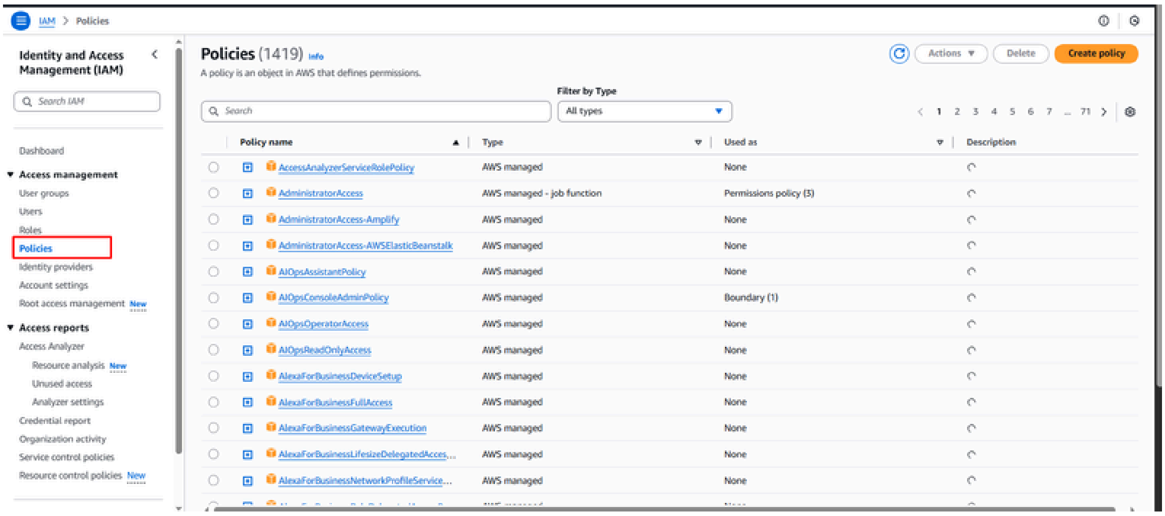
Groups
IAM feature

Roles
IAM feature

# IAM Policy & Permissions:

## Step: 2 - Navigate Policies section and choose the policy

### Policy section



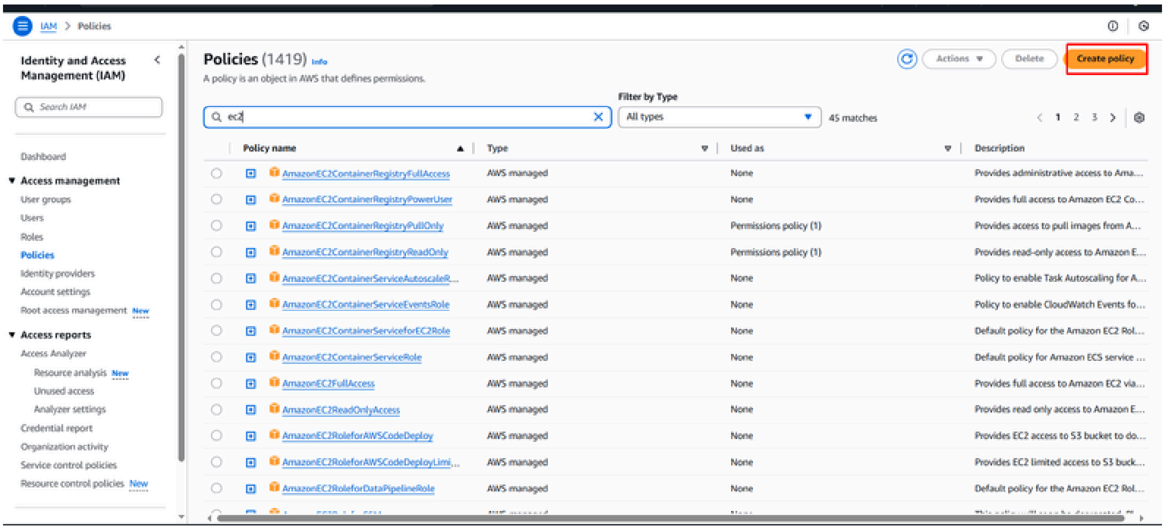
Policies (1419) Info

A policy is an object in AWS that defines permissions.

Filter by Type: All types

| Policy name | Type | Used as | Description |
|---|---|---|---|
| AccessAnalyzerServiceRolePolicy | AWS managed | None | |
| AdministratorAccess | AWS managed - job function | Permissions policy (3) | |
| AdministratorAccess-Amplify | AWS managed | None | |
| AdministratorAccess-AWSElasticBeanstalk | AWS managed | None | |
| AIOpsAssistantPolicy | AWS managed | None | |
| AIOpsConsoleAdminPolicy | AWS managed | Boundary (1) | |
| AIOpsOperatorAccess | AWS managed | None | |
| AIOpsReadOnlyAccess | AWS managed | None | |
| AlexaForBusinessDeviceSetup | AWS managed | None | |
| AlexaForBusinessFullAccess | AWS managed | None | |
| AlexaForBusinessGatewayExecution | AWS managed | None | |
| AlexaForBusinessLifesizeDelegatedAcces... | AWS managed | None | |
| AlexaForBusinessNetworkProfileService... | AWS managed | None | |

### Choose the policy and create



Policies (1419) Info

A policy is an object in AWS that defines permissions.

Search: ec2

Filter by Type: All types

45 matches

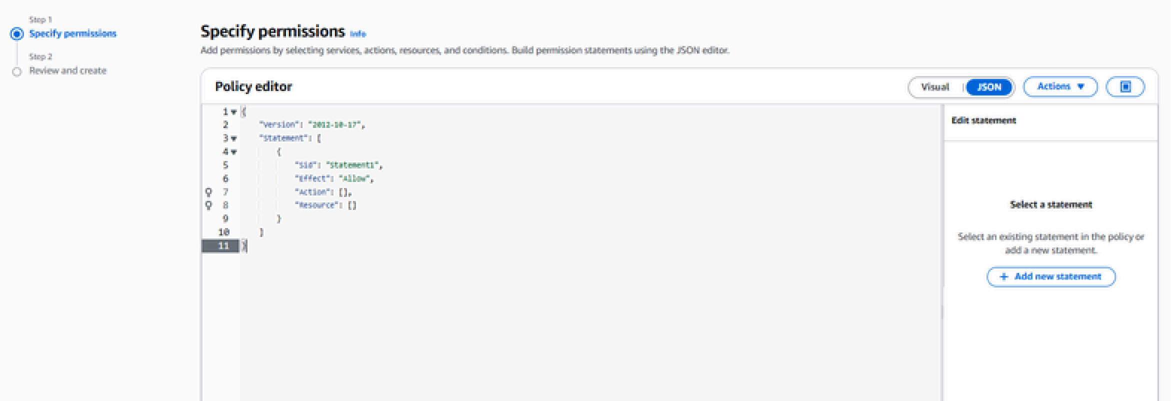
| Policy name | Type | Used as | Description |
|---|---|---|---|
| AmazonEC2ContainerRegistryFullAccess | AWS managed | None | Provides administrative access to Ama... |
| AmazonEC2ContainerRegistryPowerUser | AWS managed | None | Provides full access to Amazon EC2 Co... |
| AmazonEC2ContainerRegistryPullOnly | AWS managed | Permissions policy (1) | Provides access to pull images from A... |
| AmazonEC2ContainerRegistryReadOnly | AWS managed | Permissions policy (1) | Provides read-only access to Amazon E... |
| AmazonEC2ContainerServiceAutoscaleR... | AWS managed | None | Policy to enable Task Autoscaling for A... |
| AmazonEC2ContainerServiceEventsRole | AWS managed | None | Policy to enable CloudWatch Events fo... |
| AmazonEC2ContainerServiceforEC2Role | AWS managed | None | Default policy for the Amazon EC2 Rol... |
| AmazonEC2ContainerServiceRole | AWS managed | None | Default policy for Amazon ECS service ... |
| AmazonEC2FullAccess | AWS managed | None | Provides full access to Amazon EC2 via... |
| AmazonEC2ReadOnlyAccess | AWS managed | None | Provides read only access to Amazon E... |
| AmazonEC2RoleforAWSCodeDeploy | AWS managed | None | Provides EC2 access to S3 bucket to do... |
| AmazonEC2RoleforAWSCodeDeployLimi... | AWS managed | None | Provides EC2 limited access to S3 buck... |
| AmazonEC2RoleforDataPipelineRole | AWS managed | None | Default policy for the Amazon EC2 Rol... |

# IAM Policy & Permissions:

### Step: 3 - Validate/Review policy then create

AWS offers IAM Access Anylizer with enhanced policy checks and recommendations to optimize your policies. When creating or editing a policy in the JSON tab, a policy validation pane appears below, highlighting various findings. These findings are categorized into Security, Errors, Warnings, and Suggestions, helping you identify potential issues. You can use this information to adjust your policy and address the findings to ensure improved security and compliance

## JSON Format



Specify permissions Info

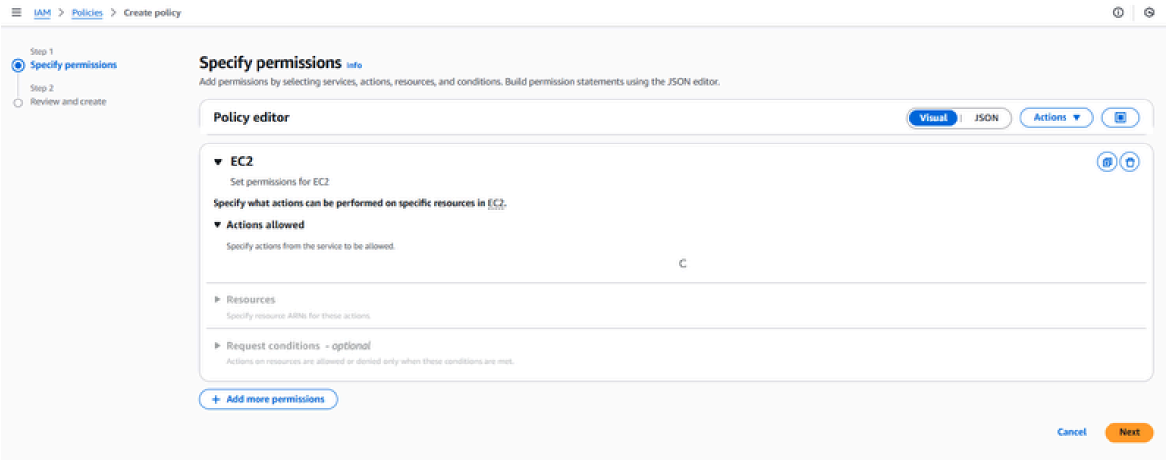
Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Action": [],
            "Resource": []
        }
    ]
}
```

Select a statement

Select an existing statement in the policy or add a new statement.

## Visual



Specify permissions Info

Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.

▼ EC2

Set permissions for EC2

Specify what actions can be performed on specific resources in EC2.

▼ Actions allowed

Specify actions from the service to be allowed.

▶ Resources

Specify resource ARNs for these actions.

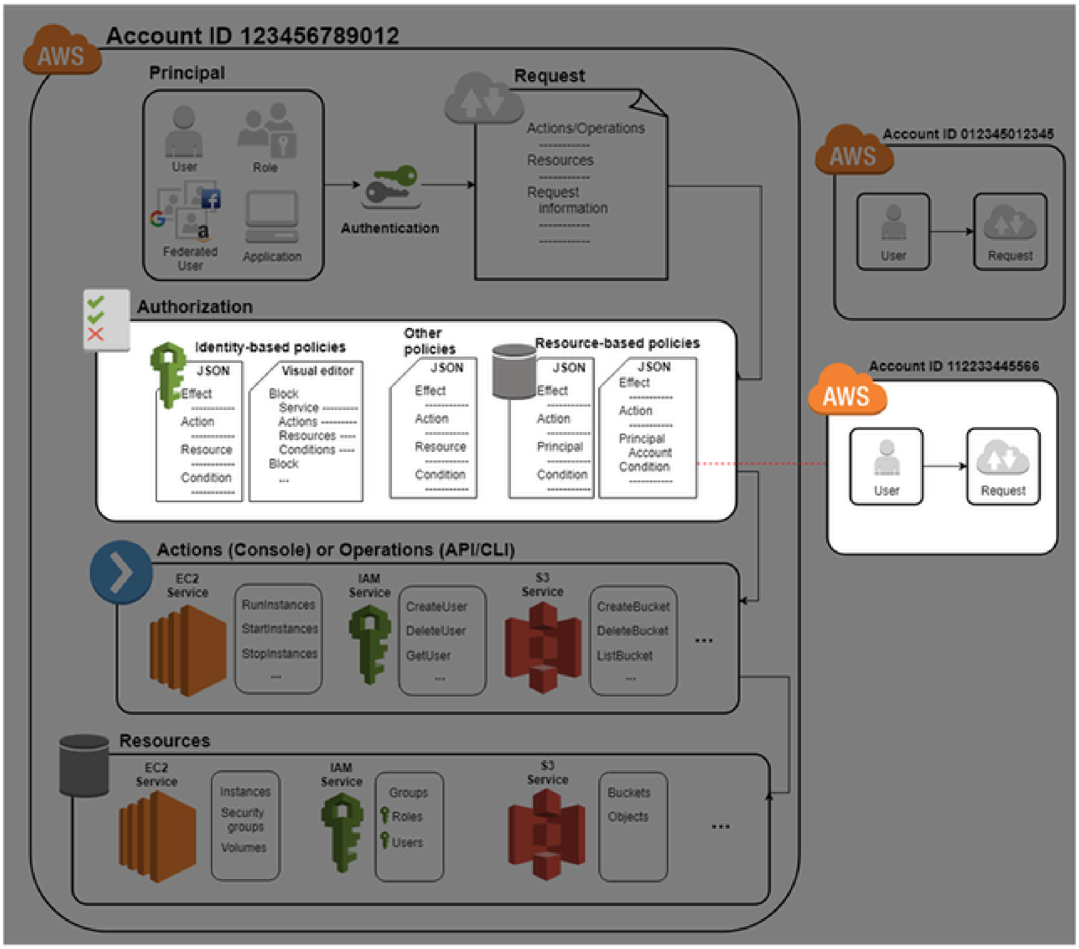
▶ Request conditions - optional

Actions on resources are allowed or denied only when these conditions are met.

# Access Management:

AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources. When a principle makes a request in AWS, the AWS enforcement code checks whether the principal is authenticated (signed in) and authorized (has permissions). You manage access in AWS by creating policies and attaching them to IAM identities or AWS resources. Policies are JSON documents in AWS that, when attached to an identity or resource, define their permissions.

Here the complete visual view:



# Thank You

## Stay Connect:

/in/alamgirweb11

/alamgirweb11