



General Immersion Day

Lab 4

IAM Hands on Lab

Identity & Access Management (IAM) Overview

[AWS Identity and Access Management \(IAM\)](#) is a web service that helps you securely control access to AWS resources. You use IAM to control who is **authenticated** (signed in) and **authorized** (has permissions) to use resources.

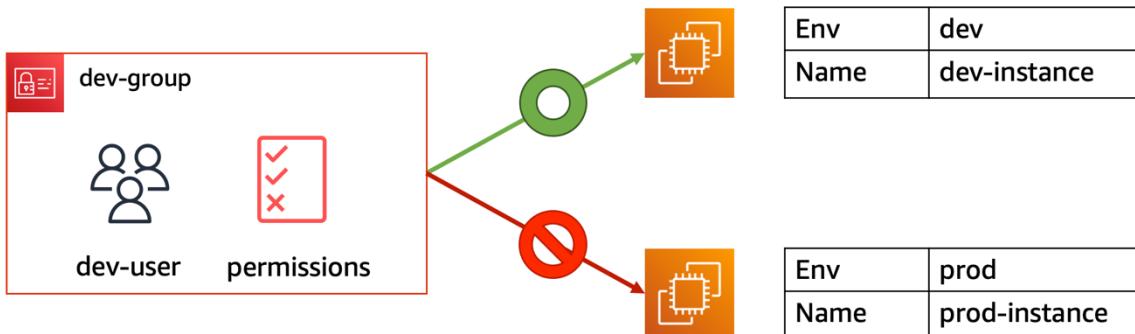
When you first create an AWS account, you begin with a single sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account **root user** and is accessed by signing in with the email address and password that you used to create the account.

Note: We strongly recommend that you **do not use the root user** for your everyday tasks, even the administrative ones. Instead, create and use **IAM user** by referring to the instructions [here](#).

AWS IAM key components:

- IAM Identities
 - [IAM Users](#)
 - [IAM User groups](#)
 - [IAM Roles](#)
- [IAM Policy](#)

And to help secure your AWS resources, check the document for [security best practices in IAM](#).



This hands on lab is broken into the following parts:

1. [Launch EC2 Instances with Tags](#)
2. [Create AWS IAM Identities](#)
3. [Test the access for resources](#)
4. [Assign IAM Role for EC2 Instance and Test the access](#)

4-1 Launch EC2 Instances with Tags

In this lab, we will launch two Amazon Linux 2 instances. Suppose one is an EC2 instance used in a development environment and the other one used in a production environment. We will use tags to distinguish these two instances.

1. Enter into [EC2 Console](#) in AWS Management Console.

Search results for 'ec2'

Services (6)

EC2

Virtual Servers in the Cloud

EC2 Image Builder

A managed service to automate build, customize and deploy OS images

Click on **EC2 Dashboard** near the top of the leftmost menu. And Click on **Launch instances**.

Resources

You are using the following Amazon EC2 resources in the Asia Pacific (Seoul) Region:

Instances (running)	0	Dedicated Hosts	0	Elastic IPs	0
Instances	0	Key pairs	1	Load balancers	0
Placement groups	0	Security groups	1	Snapshots	0
Volumes	0				

Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Launch Instance Migrate a server

Service health

Region: Asia Pacific (Seoul)

Status

2. In **Name**, enter the value **prod-instance**. And click on **Add additional tags**.

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name

prod-instance

Add additional tags

3. Click Add tag then in Key enter Env, and Value enter prod.

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

▼ Name and tags [Info](#)

Key Info Value Info Resource types Info

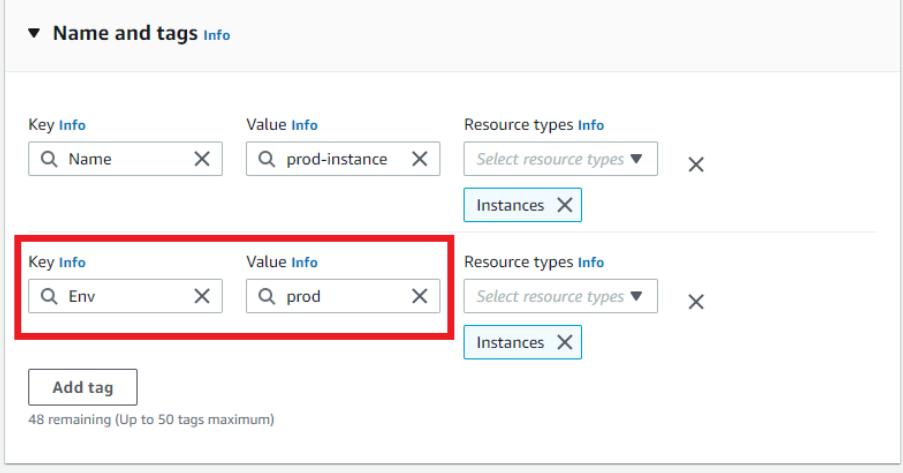
Name prod-instance Select resource types Instances

Key Info Value Info Resource types Info

Env prod Select resource types Instances

Add tag

48 remaining (Up to 50 tags maximum)



4. Check the default settings for Amazon Machine Image and select t2.micro in Instance Type.

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Quick Start

Amazon Linux Ubuntu Windows Red Hat SUSE Linux >  Browse more AMIs Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type Free tier eligible
ami-0022f774911c1d690 (64-bit (x86)) / ami-0e449176cecc3e577 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Amazon Linux 2 Kernel 5.10 AMI 2.0.20220426.0 x86_64 HVM gp2

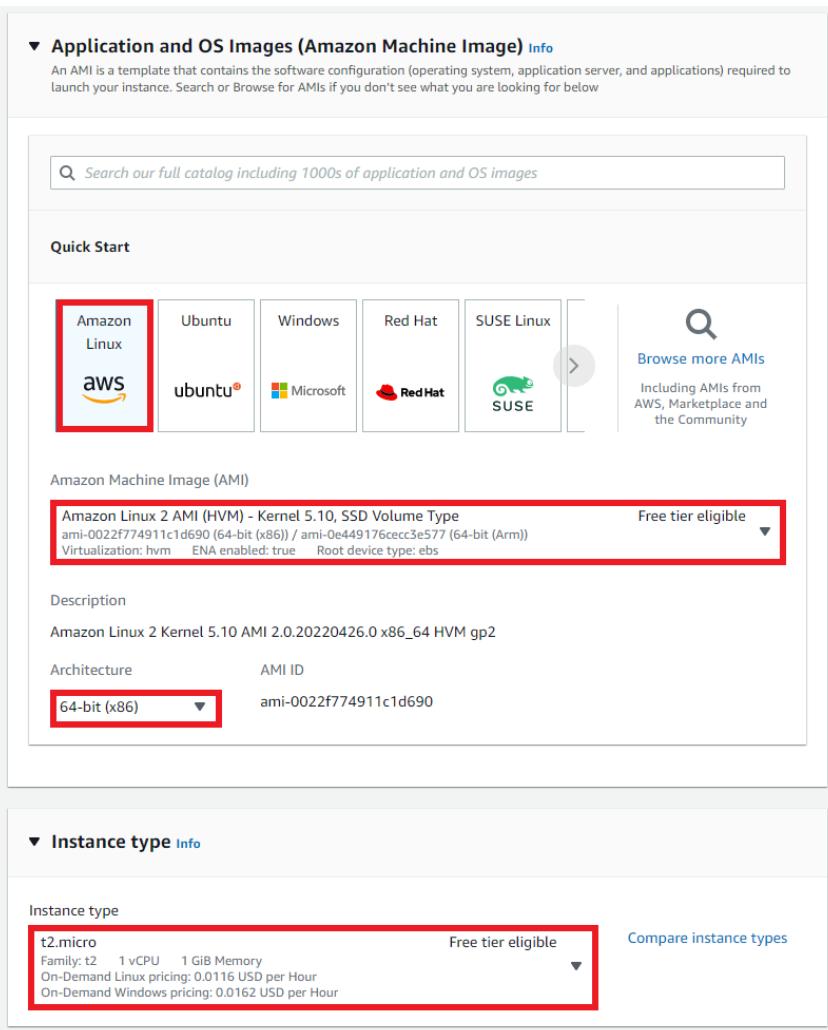
Architecture AMI ID

64-bit (x86) ami-0022f774911c1d690

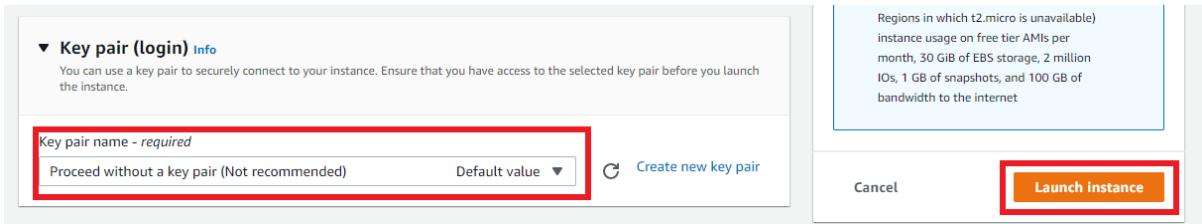
▼ Instance type [Info](#)

Instance type

t2.micro Free tier eligible Compare instance types
Family: t2 1 vCPU 1 GiB Memory
On-Demand Linux pricing: 0.0116 USD per Hour
On-Demand Windows pricing: 0.0162 USD per Hour

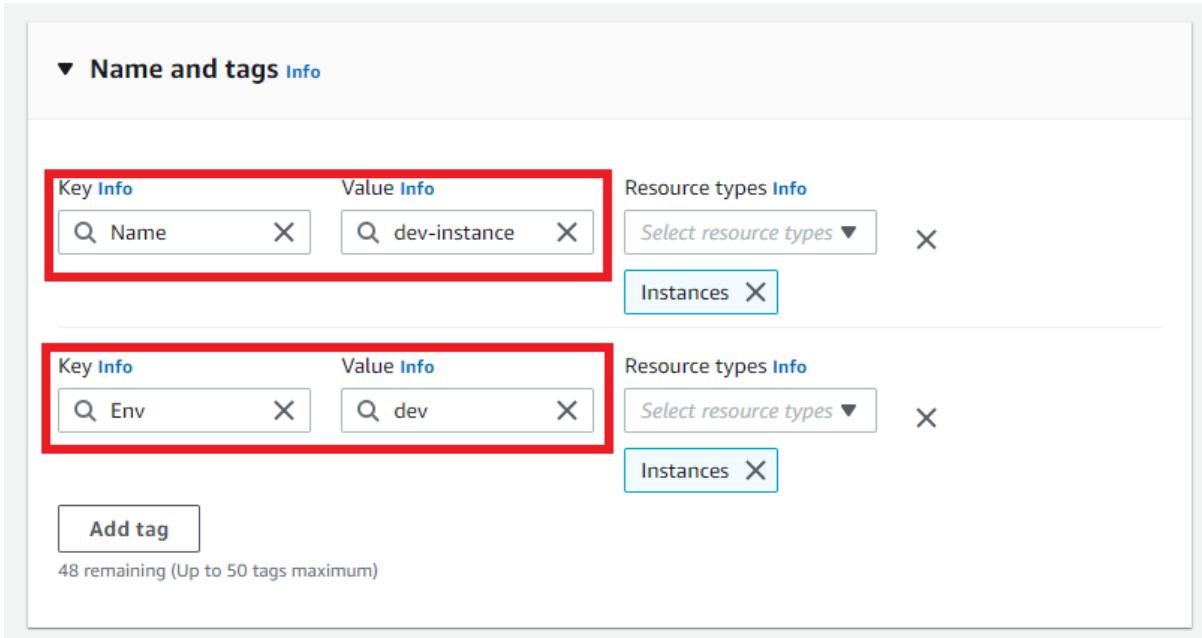


5. In Key pair (login) select **Proceed without a key pair**. Then click **Launch instance**.



6. You have launched an EC2 instance for production environment. Now you have to create **one more EC2 instance for development environment**. *Repeat the above instructions (start with no.1 to no.5), but with a different Name and Env tag in step no.2 and step no.3 as follows.*

KEY	VALUE
Name	dev-instance
Env	dev



7. After you launched two instances, you can find them in the instances menu in the sidebar. Click the prod-instance checkbox and click the **Tags** tab in the bottom of the page. You can see details about this EC2 instance's tag information.

Instances (1/2) Info

Instance ID	Instance state	Instance type	Status check
i-0e0218d4da46f0767	Running	t2.micro	Initializing
i-0fdcb8a9f7e28c41e	Running	t2.micro	-

Instance: i-0e0218d4da46f0767 (prod-instance)

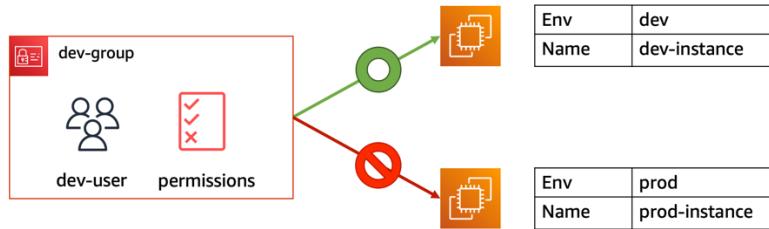
Tags

Key	Value
Env	prod
Name	prod-instance

Great Job! You have deployed two instances(servers) with different resource tags.

4-2 Create AWS IAM Identities

In this chapter, we will work on creating **AWS IAM Identities**. AWS IAM Identity includes IAM Users, IAM User groups, and IAM Roles. Also, we will work on creating **IAM Policy** an object in AWS that, when associated with an identity or resource, defines their permissions.



This chapter consists of the following steps.

- Create **IAM Policy** to attach to IAM User group
- Create **IAM User group** named dev-group
- Create **IAM User** which name is dev-user placed in dev-group

-
1. Log in to [IAM console](#). To generate a console login **Sign-in URL**, click the **customize** button as shown below.

IAM dashboard

Screenshot of the AWS IAM dashboard showing account statistics:

- Security recommendations**: 1 item (red dot).
- Add MFA for root user**: A warning message about enabling multi-factor authentication (MFA) for the root user.
- IAM resources**:
 - User groups: 1
 - Users: 2
 - Roles: 16
 - Policies: 4
 - Identity providers: 0
- AWS Account**:
 - Account ID: 233219696677
 - Account Alias: 233219696677 (highlighted with a red box)
 - Create button (green checkmark)
 - Sign-in URL for IAM users in this account: <https://233219696677.signin.aws.amazon.com/console>

2. Input account alias. For this lab, type aws-login-user_name and click **create alias** button.

Screenshot of the "Create alias for AWS account" dialog box:

Preferred alias:
aws-login-joozero (highlighted with a red box)
aws-login-[name] (highlighted with a green checkmark)

Must be not more than 63 characters. Valid characters are a-z, 0-9, and - (hyphen).

New sign-in URL:
<https://aws-login-joozero.signin.aws.amazon.com/console>

Info: IAM users will still be able to use the default URL containing the AWS account ID.

Buttons:
Click (highlighted with a green checkmark)
Cancel
Save changes

3. Click **Policies** on the left side of the IAM console, and then click the **Create Policy** button placed at the top of the right corner.

The screenshot shows the AWS IAM Policies page. On the left, there's a navigation sidebar with 'Identity and Access Management (IAM)' at the top, followed by 'Dashboard', 'Access management' (with 'User groups', 'Users', 'Roles', and 'Policies' listed), 'Identity providers', and 'Account settings'. The 'Policies' link is highlighted with a red box and a green checkmark. In the main content area, there's a header 'Policies (834) Info' with a search bar and pagination (1-42). Below is a table with columns 'Policy Name', 'Type', 'Used as', and 'Description'. Three policies are listed: '0d8fd6bfe6d74be58e0135dd7e51bb6e-policy' (Customer managed, Permissions policy ..., Team Policy), 'ops-default-policy' (Customer managed, Permissions policy ..., ops role default policy), and 'team-default-policy' (Customer managed, Permissions policy ..., team default policy).

4. When defining policy for the AWS permissions, you can create and edit in visual editor or JSON. In this lab, we will use **JSON** method. Briefly describe the permission below, this policy allows all actions for **EC2 tagged as Env-dev**. Also, it **allows describe-related actions** for all EC2 instances. But it **denies create and delete tags action** to prevent users from modifying tags arbitrarily. Note that deny effect takes precedence over allow effect.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "ec2:ResourceTag/Env": "dev"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:Describe*",
      "Resource": "*"
    },
    {
      "Effect": "Deny",
      "Action": [
        "ec2:DeleteTags",
        "ec2:CreateTags"
      ],
      "Resource": "*"
    }
  ]
}
```

Click the JSON tab and paste the above policy and click **Next: Tags** button.

Create policy

1 2 3

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)

Visual editor JSON 1. Click Import managed policy

```
1 [ { "Version": "2012-10-17", "Statement": [ { "Effect": "Allow", "Action": "ec2:*", "Resource": "*", "Condition": { "StringEquals": { "ec2:ResourceTag/Env": "dev" } } }, { "Effect": "Allow", "Action": "ec2:Describe*", "Resource": "*" }, { "Effect": "Deny", "Action": [ "ec2>DeleteTags", "ec2>CreateTags" ], "Resource": "*" } ] }
```

2. Paste policy

If you want, you can read more on [JSON policy structure](#)

- Version: use the latest 2012-10-17 version
- Statement: can include more than one statement in a policy
- Sid(optional): an optional statement ID
- Effect: use **Allow** or **Deny** to indicate the policy allows or denies access. **Deny** has priority
- Principal: If you create a resource-based policy, you must indicate the account, user, role, or federated user to which you would like to allow or deny access. If you are creating an IAM permissions policy to attach to a user or role, you cannot include this element. The principal is implied as that user or role.
- Action: a list of actions that the policy allows or denies
- Resource: If you create an IAM permissions policy, you must specify a list of resources to which the actions apply. If you create a resource-based policy, this element is optional. If you do not include this element, then the resource to which the action applies is the resource to which the policy is attached.
- Condition Block(optional): Specify the circumstances under which the policy grants permission

-
5. Keep default settings on next step, click **Next: Review** button. Write down DevPolicy in the name section, and write down description for this policy. Next click the Create policy button.

Create policy

1 2 3

Review policy

Name* 1. DevPolicy
Use alphanumeric and '+=,@-' characters. Maximum 128 characters.

Description 2. Input description
Maximum 1000 characters. Use alphanumeric and '+=,@-' characters.

Summary

Service	Access level	Resource	Request condition
Explicit deny (1 of 285 services)			
EC2	Full: Tagging	All resources	None
Allow (1 of 285 services) Show remaining 284			
EC2	Full: List, Read, Write, Permissions management	All resources	ec2:ResourceTag/Env = dev

Tags

Key	Value
No tags associated with the resource.	

* Required

3. Click Create policy

KEY	VALUE
Name	DevPolicy
Description	IAM Policy for Dev Group

6. Click **User groups** on the left side of the IAM console, and then click the **Create group** button placed at the top of the right corner.

Identity and Access Management (IAM)

User groups 1. Click

IAM > User groups

User groups (1) Info

A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.

2. Click Create group

Group name	Users	Permissions
AdminGroup	1	Defined

7. Type dev-group for **User group name** and select DevPolicy in **Attach permissions policies - Optional** section.

Create user group

Name the group

User group name
Enter a meaningful name to identify this group.
dev-group  1. dev-group
Maximum 128 characters. Use alphanumeric and '+,-,@-' characters.

Add users to the group - Optional (1) Info

An IAM user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS. A user can belong to up to 10 groups.

<input type="checkbox"/>	User name 	Groups	Last activity 	Creation time 
<input type="checkbox"/>	EEOverlord	0	None	24 hours ago

Attach permissions policies - Optional (Selected 1/672) Info

You can attach up to 10 policies to this user group. All the users in this group will have permissions that are defined in the selected policies.

<input type="checkbox"/>	Policy Name 	Type	Description
<input type="checkbox"/>	0d8fd6bfe6d  2. Select lb6e-policy	Customer managed	Team Policy
<input checked="" type="checkbox"/>	DevPolicy 	Customer managed	IAM Policy for Dev Grou

KEY	VALUE
User group name	dev-group

8. Click **Users** on the left side of the page, and then click the **Add users** button.

Identity and Access Management (IAM)

- Dashboard
- Access management
 - Users**  1. Click
 - Roles
 - Policies
 - Identity providers
 - Account settings

IAM > Users

Users (1) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

<input type="checkbox"/>	User name	Groups	Last activity	MFA	Password age
<input type="checkbox"/>	EEOverlord	None	Never	None	None

2. Click  Add users 

9. Type dev-user on **User name** and allow two access type both Programmatic access and AWS Management console access. And then, select **Custom password** and type password that you want. Finally, uncheck Require password reset function for quick action. Click **Next: Permissions** button.

Add user

1 2 3 4 5

Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name* 1. dev-user

[+ Add another user](#)

Select AWS access type

Select how these users will access AWS. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

Access type* 2. Click
Programmatic access
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.

3. Click
AWS Management Console access
Enables a **password** that allows users to sign in to the AWS Management Console.

Console password* 4. Custom password
 Autogenerated password

5. Input password

Show password

Require password reset 6. Uncheck
User must create a new password at next sign-in
Users automatically get the [IAMUserChangePassword](#) permission to allow them to change their own password.

10. Select dev-group that we created right before and click **Next: Tags** button. Skip Add user page and move on to next step.

Add user

1 2 3 4 5

Set permissions

Add user to group

Copy permissions from existing user

Attach existing policies directly

Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. [Learn more](#)

Add user to group

[Create group](#)

[Refresh](#)

Search

Showing 1 result

Group ▾

Attached policies

dev-group

Select

DevPolicy

Set permissions boundary

Click **Create user** button for adding dev-user.

Add user

1 2 3 4 5

Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details

User name	dev-user
AWS access type	Programmatic access and AWS Management Console access
Console password type	Custom
Require password reset	No
Permissions boundary	Permissions boundary is not set

Permissions summary

The user shown above will be added to the following groups.

Type	Name
Group	dev-group

Tags

No tags were added.

Download csv file to attain Access key ID and Secret access key. Also, you can send instructions for login.

11. Click **Sign-in URL** to log in as dev-user.

Add user

1 2 3 4 5

Success

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at <https://aws-login-joozero.signin.aws.amazon.com/console>  Click

 Download .csv

	User	Access key ID	Secret access key	Email login instructions
▼	dev-user	[REDACTED]	***** Show	Send email

- ✓ Created user dev-user
- ✓ Added user dev-user to group dev-group
- ✓ Created access key for user dev-user
- ✓ Created login profile for user dev-user

12. Type **IAM user name** and **Password** for log in and sign in for entering into AWS management console.

aws

Sign in as IAM user

Account ID (12 digits) or account alias

IAM user name
 dev-user dev-user

Password
 Type password

Sign in

[Sign in using root user email](#)
[Forgot password?](#)



Great Job! You have created AWS Identities. Next chapter, we will check whether the permission we set works properly.

4-3 Test the access for resources

1. Log in AWS console and check **IAM User** and account alias. Also, check **AWS region** that launched EC2 instances are located.

The screenshot shows the AWS Management Console homepage. At the top, there's a navigation bar with 'Services ▾', a search bar ('Search for services, features, marketplace products, and docs'), and a user dropdown ('dev-user @ aws-login-joozero'). Below the search bar, the title 'AWS Management Console' is displayed. On the left, under 'AWS services', there's a 'Recently visited services' section with links to EC2, IAM, S3, Amazon SageMaker, IoT SiteWise, IoT Core, AWS Amplify, and Elastic Container Registry. To the right, there's a box titled 'Stay connected to your AWS resources on-the-go' which says 'AWS Console Mobile App now supports four additional regions. Download the AWS Console Mobile App to your iOS or Android mobile device.' A 'Learn more' link is provided. At the bottom right, there's a 'Explore AWS' button.

2. Move on to the [EC2 console](#) and click **Instances** menu. Select the instance named prod-instance and click **Instance state** button and **Stop instance** button. And click **Stop** button in the pop-up window.

The screenshot shows the EC2 Instances page. On the left, there's a sidebar with 'New EC2 Experience' (disabled), 'EC2 Dashboard', 'Events', 'Tags', 'Limits', and 'Instances'. Under 'Instances', there are 'Instances New' and 'Instance Types' buttons. A green arrow with a checkmark points to the 'Instances New' button, labeled '1. Click'. Another green arrow with a checkmark points to the 'prod-instance' row, labeled '2. Click prod'. On the main table, a red box highlights the 'Actions' column. A green arrow with a checkmark points to the 'Stop instance' button, labeled '3. Click'. A red box highlights the 'Stop instance' button. A green arrow with a checkmark points to the 'Stop instance' button in the context menu, labeled '4. Click'.

3. A warning sign appears that dev-user doesn't have permissions to perform the stop EC2 instance operation. This is because the previous hands on lab only allowed **dev-user** to perform the stop action on the ec2 instances with the resource tag **Env(key)-dev(value)**.

The screenshot shows the EC2 Instances page with a prominent red error message box at the top. The message reads: 'Failed to stop the instance i-0e0218d4da46f0767' and 'You are not authorized to perform this operation. Encoded authorization failure message: mwBAWnhG-QKtnuVVRJya2lQwvIKSPZW_08AGhCkpO6Ca7gBjz-sQF83muXGeInEhtcz7MMK0NGGNdBkj5Ii4LQ958_FNir-Y1_vfNT1HrJGw9ImSe5WVktwJa5y1wuD6xhGg7KYnjUMZ6A7ZetTsrf-5lwWMgw-c__MMAc9hBmWQlVyXW7Irov4dmPAn_wOfp251xPxspPC_YW_3RvL1jh9Qdb8L6XKGDrnLdTtj-I20ouwOZnT6XCGP_yv3R1WjzaBs3T-W1lhebhNZHO8HMzw6ggj5pOhIOQswL8Wo-67Hii5hVYJIV0cG0o_aMX9ffFoxql5Rvv-xH4QxmEsplqXlODCLsChE5hdn1APeU3lmAHWKQOBVTxaZ68Elquik-wbPiWj0lmlrOHxdMO5mBvNrSAnts-nKw0qShzbEpK-LhX4f318vXZl3oQjFE0lEV9-3hvpoqchXYY6pu!SgwDl-NqB6SSQ-8jH3PJvXOT-WOTTNgS2MeM65DSzrPRUOB35CRbzrprP9laQmyDuQlmYlxD_B_rw7-mah1huMie2LkmVgzNqTPExgjTmaxApfyb87abdMjwn1PuKvX55b28TN0kwyTPdOoG_gvaLy1zfz1EEw_sl_OCb6GyaY9zOZ6jz98tegvH5RzZT-eAJGGF2qnbcDm9ETRGe19WiXlKec31kV4GEQ'. Below the error message, the table shows two instances: 'dev-instance' (Running) and 'prod-instance' (Running). The 'Actions' column for 'prod-instance' has a 'Launch Instances' button.

4. Select the instance named dev-instance and click **Instance state** button and **Stop instance** button. And click **Stop** button in the pop-up window.

The screenshot shows the AWS EC2 Instances page with two instances listed: 'dev-instance' and 'prod-instance'. The 'dev-instance' row has a checked checkbox. A context menu is open over the 'dev-instance' row, with the 'Stop instance' option highlighted by a red box. Other options in the menu include 'Start instance', 'Reboot instance', 'Hibernate instance', and 'Terminate instance'.

5. After a few seconds, you can see the dev instance has been stopped.

The screenshot shows the AWS EC2 Instances page after stopping the 'dev-instance'. The 'dev-instance' row now has a status of 'Stopped' indicated by a red box. The 'prod-instance' row remains 'Running'. The status check column shows '2/2 checks passed' for both instances.

Great Job! You tested the permission for each EC2 instance. Next chapter, we will learn about AWS IAM Role for AWS resources!

The screenshot shows the IAM Policy Simulator interface. On the left, under 'Policies', there is a 'Selected group: dev-group' and a 'Filter' section containing 'DevPolicy'. On the right, the 'Policy Simulator' section shows a 'Select 'EC2'' dropdown, a 'Select Actions' button, and a 'Run Simulation' button. Below these are 'Global Settings' and 'Action Settings and Results' tables. The 'Action Settings and Results' table shows two actions: 'DeleteTags' and 'StopInstances', both of which are denied. The 'DeleteTags' action is denied because of '1 matching statements' and the 'StopInstances' action is denied because it is 'Implicitly denied (no matching statements)'.

4. You can see both actions are denied but with different reasons. **DeleteTags** was denied because of **1 matching statements** and **StopInstances** was **Implicitly denied (no matching statements)**.

Service	Action	Resource Type	Simulation Resource	Permission
Amazon EC2	DeleteTags	not required	*	denied 1 matching statements.
Amazon EC2	StopInstances	instance	*	denied Implicitly denied (no matching statem...)

The screenshot shows the IAM Policy Simulator interface. On the left, the 'Policies' section displays a JSON-based policy document named 'DevPolicy'. The policy contains two statements: one allowing EC2 actions on resources tagged with 'dev' and another allowing EC2.Describe on all resources. On the right, the 'Policy Simulator' section shows the results of a simulation for the 'Amazon EC2' service. It lists two actions: 'DeleteTags' and 'StopInstances'. The 'DeleteTags' action is allowed (not required resource), while the 'StopInstances' action is denied (no matching statement). A red arrow points from the 'DeleteTags' row in the simulator results back to the corresponding statement in the policy document.

6. Expand StopInstances. The action was denied *because the simulation resource is "/*"*. Please note that **dev-group** can only stop EC2 instances having **dev** tags.

This screenshot focuses on the 'StopInstances' action settings in the simulator. It shows the 'Resource' configuration where the simulation resource is set to '*' (all resources). Below this, there is a field for specifying an instance, which contains the value 'ec2:resourcetag/env' with a note 'Leave blank to ignore key.'

7. Now we will test the policy after adding **dev** tag to validate the policy correctly.

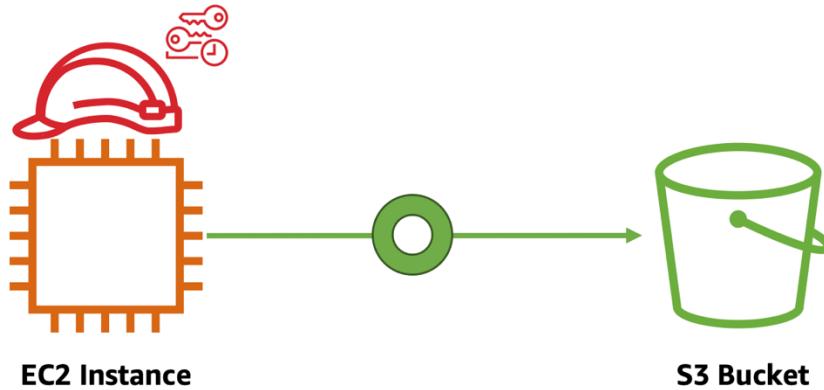
This screenshot shows the same IAM Policy Simulator interface after adding a 'dev' tag to the EC2 instance. The 'Run Simulation' button is highlighted with a green checkmark. The results show that the 'DeleteTags' action remains denied, but the 'StopInstances' action is now allowed (resource 'ec2:resourcetag/dev'). A green checkmark indicates the 'Allowed' status for the 'StopInstances' action. A callout bubble labeled '3. Allowed' points to this result.

Great Job! You just tested the your custom IAM policy without affecting your environment.

With the IAM policy simulator, you can test and troubleshoot identity-based policies, IAM permissions boundaries, Organizations service control policies (SCPs), and resource-based policies. You can learn [more details here](#).

4-4 Assign IAM Role for EC2 Instance and Test the access

Re-login to the **AWS account with administrator role** at the beginning of the hands on lab, not dev-user before proceeding with this chapter.



1. To create S3 Bucket, enter into the [S3 console](#). And then, click **Create bucket** button. For more detail information on Amazon S3, please refer to the Storage on AWS chapter.

The screenshot shows the Amazon S3 console interface. On the left, there's a sidebar with options like Buckets, Storage Lens, and Feature spotlight. The main area has a heading "Account snapshot" and a "Buckets (0)" section. Below the buckets section is a table with columns: Name, AWS Region, Access, and Creation date. A large orange "Create bucket" button is prominently displayed at the top of the buckets section. A green arrow points to this button with the label "Click".

2. Enter a unique bucket name in the **Bucket name** field. In this lab, type iam-test-user_name. All bucket names in Amazon S3 have to be unique and cannot be duplicated. Click Create bucket button, leaving the remaining settings intact.

Create bucket Info

Buckets are containers for data stored in S3. [Learn more](#)

General configuration

Bucket name

iam-test-joozero

Bucket name must be unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region

Asia Pacific (Seoul) ap-northeast-2 ▾

Copy settings from existing bucket - optional

Only the bucket settings in the following configuration are copied.

[Choose bucket](#)

Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

Block public access to buckets and objects granted through new access control lists (ACLs)

S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

Block public access to buckets and objects granted through any access control lists (ACLs)

S3 will ignore all ACLs that grant public access to buckets and objects.

Block public access to buckets and objects granted through new public bucket or access point policies

S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

Block public and cross-account access to buckets and objects through any public bucket or access point policies

S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

3. Upload any file (sample: [aws-logo-picture](#)) in the S3 bucket.

Upload Info

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)

Drag and drop files and folders you want to upload here, or choose **Add files**, or **Add folders**.

Files and folders (1 Total, 33.9 KB)

All files and folders in this table will be uploaded.

[Remove](#)

[Add files](#)

[Add folder](#)

<input type="checkbox"/>	Name	Folder	Type	Size
<input type="checkbox"/>	aws-logo-sample.jpg	-	image/jpeg	33.9 KB

Destination

Destination

s3://iam-test-joozero

▶ Destination details

Bucket settings that impact new objects stored in the specified destination.

▶ Permissions

Grant public access and access to other AWS accounts.

▶ Properties

Specify storage class, encryption settings, tags, and more.

[Cancel](#)

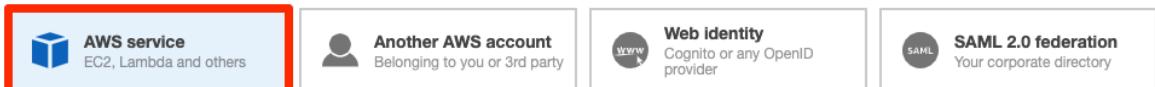
[Upload](#)

4. Create one more bucket named iam-test-other-user_name. Upload any file in the S3 bucket.
5. Move on to the [IAM console](#) to create IAM role for EC2 instance. Click **Roles** on the left side of the IAM console and click **Create role** button. On first step, choose **EC2** for trusted entity and click **Next: Permissions**. IAM Role can be applied to many AWS services including EC2 and Lambda, as well as other AWS accounts, Web identity, and SAML 2.0 federation.

Create role

1 2 3 4

Select type of trusted entity



Allows AWS services to perform actions on your behalf. [Learn more](#)

Choose a use case

Common use cases

EC2

Allows EC2 instances to call AWS services on your behalf.



Lambda

Allows Lambda functions to call AWS services on your behalf.

6. In step 2, click **Create policy** to create a policy to attach to the EC2 instance's role. On the **JSON** tab, paste the policy below and click **Next: Tags**.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Action": ["s3>ListAllMyBuckets", "s3:GetBucketLocation"],  
      "Effect": "Allow",  
      "Resource": ["arn:aws:s3:::*"]  
    },  
    {  
      "Effect": "Allow",  
      "Action": [  
        "s3:Get*",  
        "s3>List*"  
      ],  
      "Resource": [  
        "arn:aws:s3:::iam-test-user_name/*",  
        "arn:aws:s3:::iam-test-user_name"  
      ]  
    }  
  ]  
}
```

Create policy

1 2 3

A policy defines the AWS permissions that you can assign to a user, group, or role. You can create and edit a policy in the visual editor and using JSON. [Learn more](#)

Visual editor

JSON

Import managed policy

```
1 {  
2     "Version": "2012-10-17",  
3     "Statement": [  
4         {  
5             "Action": ["s3>ListAllMyBuckets", "s3:GetBucketLocation"],  
6             "Effect": "Allow",  
7             "Resource": ["arn:aws:s3:::*"]  
8         },  
9         {  
10            "Effect": "Allow",  
11            "Action": [  
12                "s3:Get*",  
13                "s3>List*"  
14            ],  
15            "Resource": [  
16                "arn:aws:s3:::iam-test-joozero/*",  
17                "arn:aws:s3:::iam-test-joozero"  
18            ]  
19        }  
20    ]  
21}  
22 }
```

In the json file above, make sure to change the value in the **Resource** to the name of the S3 bucket you created.

6. To skip adding tags operation, click **Next:Review** and type IAMBucketTestPolicy in the **Name** input box. Click **Create policy** to attach to IAM role.

Create policy

Review policy

Name*

IAMBucketTestPolicy

Use alphanumeric and '+,-,@-_` characters. Maximum 128 characters.

Description

Maximum 1000 characters. Use alphanumeric and '+,-,@-_` characters.

7. Back to the page where you were creating IAM Role, press the refresh button on the right corner and type IAMBucketTestPolicy. If you see the policy you just created in the list below, select it and go to the next step.

Create role

1 2

▼ Attach permissions policies

Choose one or more policies to attach to your new role.

Create policy

Filter policies ▾ Type policy name

Policy name ▾	Used as
<input checked="" type="checkbox"/> IAMBucketTestPolicy	None

Click

8. In step 4, type IAMBucketTestRole and create role.

Create role

1 2 3 4

Review

Provide the required information below and review this role before you create it.

Role name*

Use alphanumeric and '+,-,@-' characters. Maximum 64 characters.

Role description

Maximum 1000 characters. Use alphanumeric and '+,-,@-' characters.

Trusted entities AWS service: ec2.amazonaws.com

Policies [IAMBucketTestPolicy](#)

Permissions boundary Permissions boundary is not set

No tags were added.

9. Move on to the [EC2 console](#) to attach IAM role on the EC2 instance. At this time, if you don't have EC2 instances created in the previous lab, check the top right to see whether AWS Region is properly set up. Select prod-instance, then click the **Connect** button.

Instances (1/2) [Info](#)

<input type="checkbox"/>	Name	Instance ID
<input type="checkbox"/>	dev-instance	i-0fdcb8a9f7e28c41e
<input checked="" type="checkbox"/>	prod-instance	i-0e0218d4da46f0767

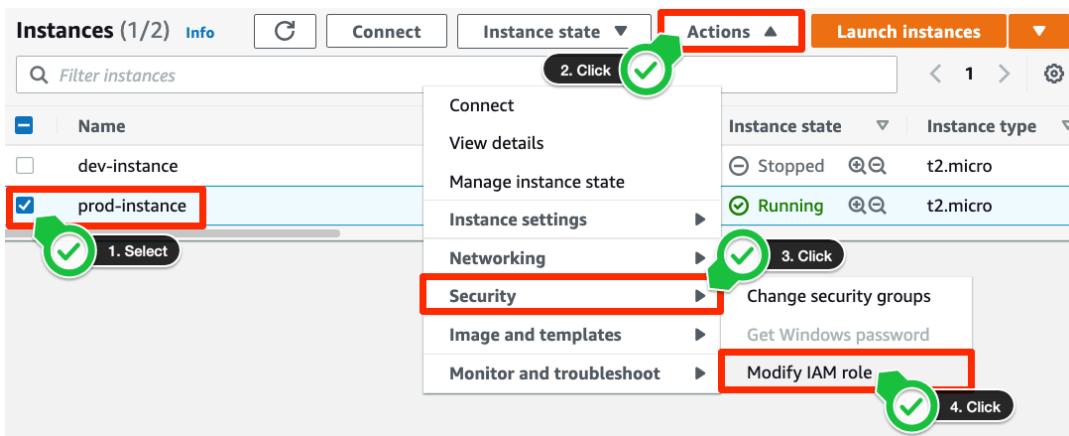
10. Connect to instance by using EC2 Instance Connect option, and click **Connect** button. Then the terminal will come out as below. Type aws s3 ls and you cannot get S3 bucket lists.

```

[ec2-user@ip-172-31-16-4 ~]$ ls
.
..
[ec2-user@ip-172-31-16-4 ~]$ aws s3 ls
An error occurred (AccessDenied) when calling the ListBuckets operation: Access Denied
[ec2-user@ip-172-31-16-4 ~]$ 

```

11. Back to the **Instances** page and select prod-instance. Click **Actions** button and select **Security** and click **Modify IAM role**.



12. In IAM role, select IAMBucketTestRole and click **Save** button to attach IAM role to EC2 instance.

The screenshot shows the 'Modify IAM role' dialog box. The 'Instance ID' field contains 'i-0e0218d4da46f0767 (prod-instance)'. The 'IAM role' section shows a dropdown menu with 'No IAM Role' selected. A red box highlights the 'IAMBucketTestRole' option in the dropdown menu. A green checkmark icon with the number '1. Select' is placed over the 'No IAM Role' option. A green checkmark icon with the number '2. Click' is placed over the 'IAMBucketTestRole' option. A green checkmark icon with the number '3. Click' is placed over the 'Create new IAM role' link. A green checkmark icon with the number '4. Click' is placed over the 'Save' button. The 'Save' button is highlighted with a red box.

13. Again, connect to the EC2 instance and type aws s3 ls again. Now you can see S3 bucket lists. Also, you can see object list placed in iam-test-user_name but not in iam-test-other-user_name because you don't have IAM policy for this bucket.

```
[ec2-user@ip-172-31-4-64 ~]$ aws s3 ls
2021-07-31 15:30:58 iam-test-joozero
2021-07-31 16:20:04 iam-test-other-joozero
[ec2-user@ip-172-31-4-64 ~]$ aws s3 ls iam-test-joozero
2021-07-31 17:19:30      34664 aws-logo-sample.jpg
[ec2-user@ip-172-31-4-64 ~]$ aws s3 ls iam-test-other-joozero
An error occurred (AccessDenied) when calling the ListObjectsV2 operation: Access Denied
[ec2-user@ip-172-31-4-64 ~]$ █
```

```
aws s3 ls

aws s3 ls iam-test-user_name

aws s3 ls iam-test-other-user_name
```

Great Job! You finished all hands on lab about IAM