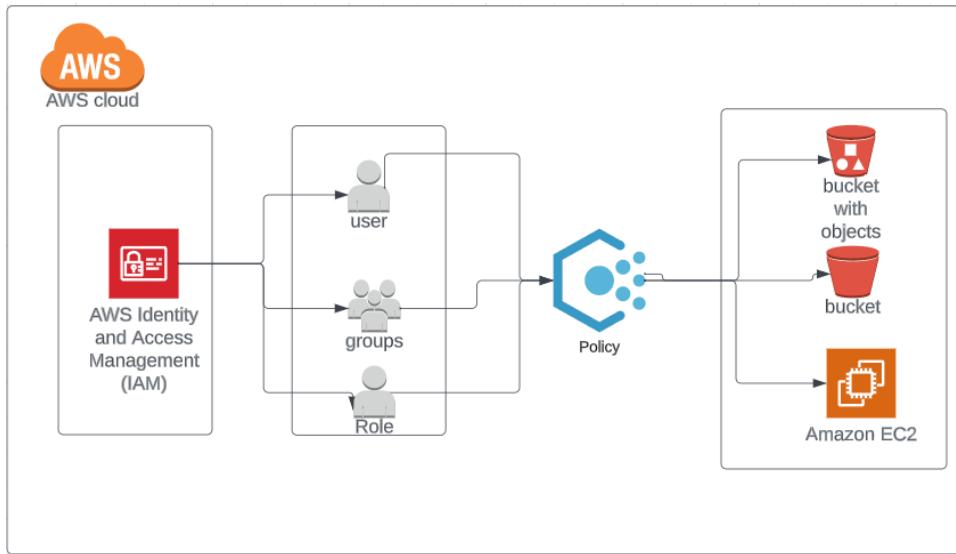


Description:

In this lab we are focusing on Amazon IAM services which is Identity Access Management. With this service we can create users, roles, groups, policies and limit the access to the users in AWS. The difference between user and role is a user can login to the AWS, but a role cannot login. We can attach a role to a user or group. We can attach policies to the users/roles in IAM.

Topology:

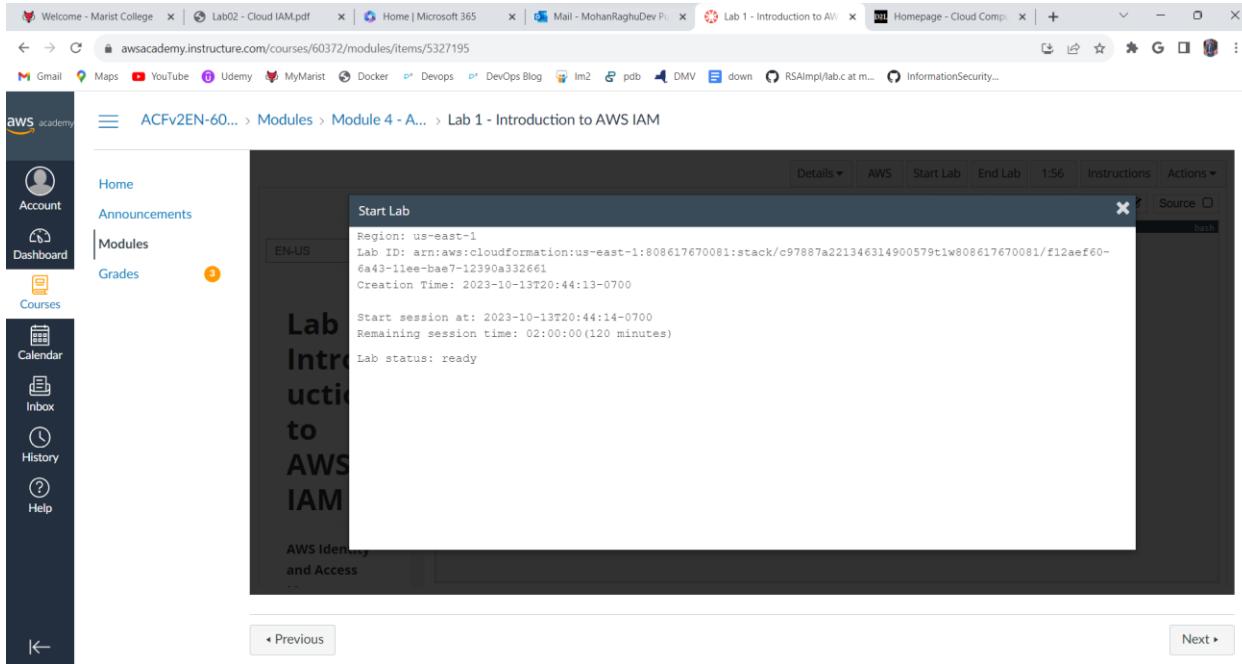


Verification:

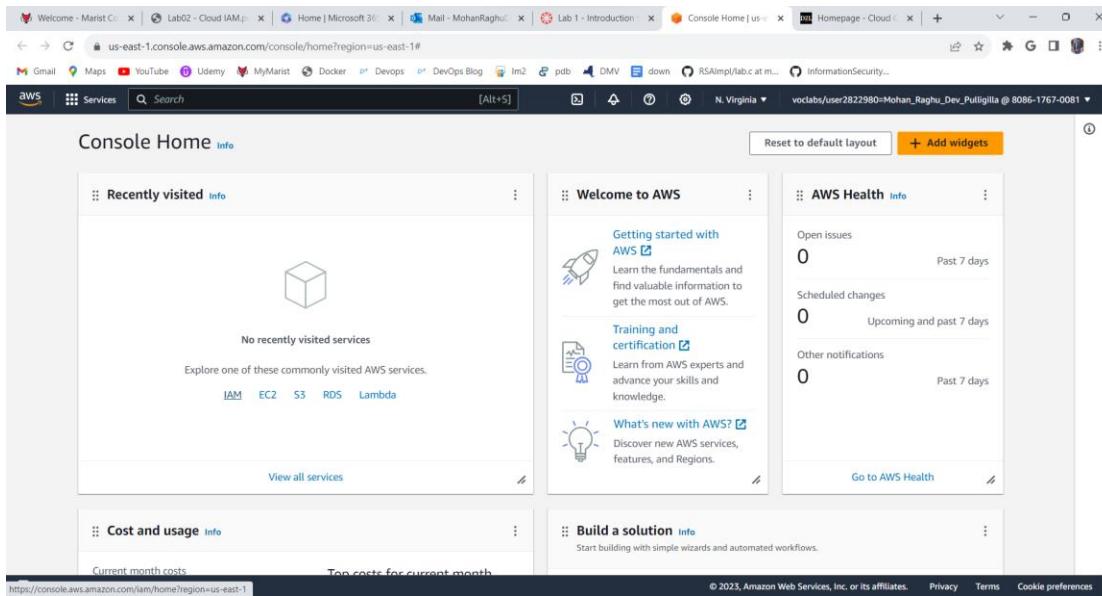
Below are the screenshots of task 1 and 2 exploring the pre created users using AWS CLI and AWS Management Console. Assigned the users with some groups and roles. They worked fine for EC2 and S3. S3 has only read access.

Task I:

- Once login to the AWS academy you need to click on start lab. After successfully starting the lab, it states that the lab is ready to use.



- After the lab is ready to use, we need to click on AWS so that it will redirect to AWS management console.



- We need to search for AWS IAM service for that we need to click on shortcut, or we can type in search bar for the required services.
- I have selected user groups which are pre created, only three groups EC2 Admin, Ec2 Support, S3 Support. The support group has read access to specific services. Admin group has full access to that service.

The screenshot shows the AWS IAM User Groups page. The left sidebar is collapsed, and the main content area displays a table titled "User groups (3) Info". The table has columns for Group name, Users, Permissions, and Creation time. Three groups are listed: EC2-Admin, EC2-Support, and S3-Support, all created 5 minutes ago. Each group has 0 users and defined permissions.

Group name	Users	Permissions	Creation time
EC2-Admin	0	Defined	5 minutes ago
EC2-Support	0	Defined	5 minutes ago
S3-Support	0	Defined	5 minutes ago

- I have selected the users to display the list of pre created users.
- There are Four users. AWS Student, User-1, User-2, User-3.

The screenshot shows the AWS IAM Users page. The left sidebar is collapsed, and the main content area displays a table titled "Users (4) Info". The table has columns for User name, Path, Group, Last activity, MFA, Password age, and Console last sign-in. Four users are listed: awsstudent, user-1, user-2, and user-3, all belonging to the / group and having access denied. Their last activity was 5 minutes ago, they do not have MFA, their password age is - days, and there is no console last sign-in information.

User name	Path	Group	Last activity	MFA	Password age	Console last sign-in
awsstudent	/	Access denied	Access denied	Access denied	Access denied	-
user-1	/spl66/	0	-	-	5 minutes	-
user-2	/spl66/	0	-	-	5 minutes	-
user-3	/spl66/	0	-	-	5 minutes	-

- I have selected policies to see the list of policies, there are 1131 policies which are created by AWS, we can Identify user created as well as AWS created policies by seeing the AWS logo Infront of the policies, whereas user created policies does not have any logo Infront of policy.

The screenshot shows the AWS IAM Policies page. The left sidebar includes sections for Dashboard, Access management (User groups, Users, Roles, Policies), Access reports, and Credential report. The main content area displays a table titled "Policies (1131) Info" with columns for Policy name, Type, Used as, and Description. The table lists various AWS managed policies such as AccessAnalyzerService, AdministratorAccess, AlexaForBusinessDevice, AlexaForBusinessFullAccess, AlexaForBusinessGateway, AlexaForBusinessLifesize, AlexaForBusinessNet, and AlexaForBusinessPoly. The "Create policy" button is located at the top right of the table.

- I have selected roles to see the pre created roles in AWS, we can attach these roles to the users as well as the machines we created. Example EC2_S3role is used to link EC2 and the S3 bucket to read and write to the bucket.

The screenshot shows the AWS IAM Roles page. The left sidebar includes sections for Dashboard, Access management (User groups, Users, Roles, Policies), Access reports, and Credential report. The main content area displays a table titled "Roles (13) Info" with columns for Role name, Trusted entities, and Last activity. The table lists various AWS service roles such as AWSServiceRoleForAWSCloud9, AWSServiceRoleForCloudWatchEvents, AWSServiceRoleForElastiCache, AWSServiceRoleForOrganizations, AWSServiceRoleForSupport, AWSServiceRoleForTrustedAdvisor, EMR_AutoScaling_DefaultRole, EMR_DefaultRole, and EMR_EC2_DefaultRole. The "Create role" button is located at the top right of the table.

- I have selected user-1 and checked permissions, but there are no permissions attached to it.

The screenshot shows the AWS IAM User Details page for 'user-1'. The 'Permissions' tab is active, displaying the message 'Permissions policies (0)'. Below this, there is a search bar and a table with columns for 'Policy name' and 'Type', which is currently empty. The 'Groups' tab is visible but has 0 entries. The 'Summary' section provides basic user information like ARN, creation date, and access keys.

- User-1 doesn't have any groups attached to it.

The screenshot shows the AWS IAM User Details page for 'user-1'. The 'Groups' tab is active, displaying the message 'User groups membership (0)'. Below this, there is a search bar and a table with columns for 'Group name' and 'Attached policies', which is currently empty. The 'Permissions' tab is visible but has 0 entries. The 'Summary' section provides basic user information like ARN, creation date, and access keys.

- Click on add user to group and add the groups to user, for user-1 I have added the default 3 pre created groups.

Permissions options

- Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
- Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.
- Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Group name	Users	Attached policies	Created
EC2-Admin	0	-	2023-10-13 (55 minutes ago)
EC2-Support	0	AmazonEC2ReadOnlyAccess	2023-10-13 (55 minutes ago)
S3-Support	0	AmazonS3ReadOnlyAccess	2023-10-13 (55 minutes ago)

Set permissions boundary - optional

- Once select the groups then click on next and add permissions.

Review

The following policies will be attached to this user. [Learn more](#)

Name	Type	Used as
EC2-Admin	Group	-
EC2-Support	Group	-
S3-Support	Group	-

- In the same way added EC2-Support group to user-2, which has only read access to EC2.

The screenshot shows the AWS IAM User-2 details page. The user was created on October 14, 2023, at 21:41 UTC-04:00. They have one access key (AKIA3YRK60XAYU5QRP4C) which is active and never used. The user is a member of the EC2-Support group, which is attached to the AmazonEC2ReadOnlyAccess policy.

Group	Attached Policies
EC2-Support	AmazonEC2ReadOnlyAccess

- Added S3-Support group to user-3, which has only read access to S3 bucket.

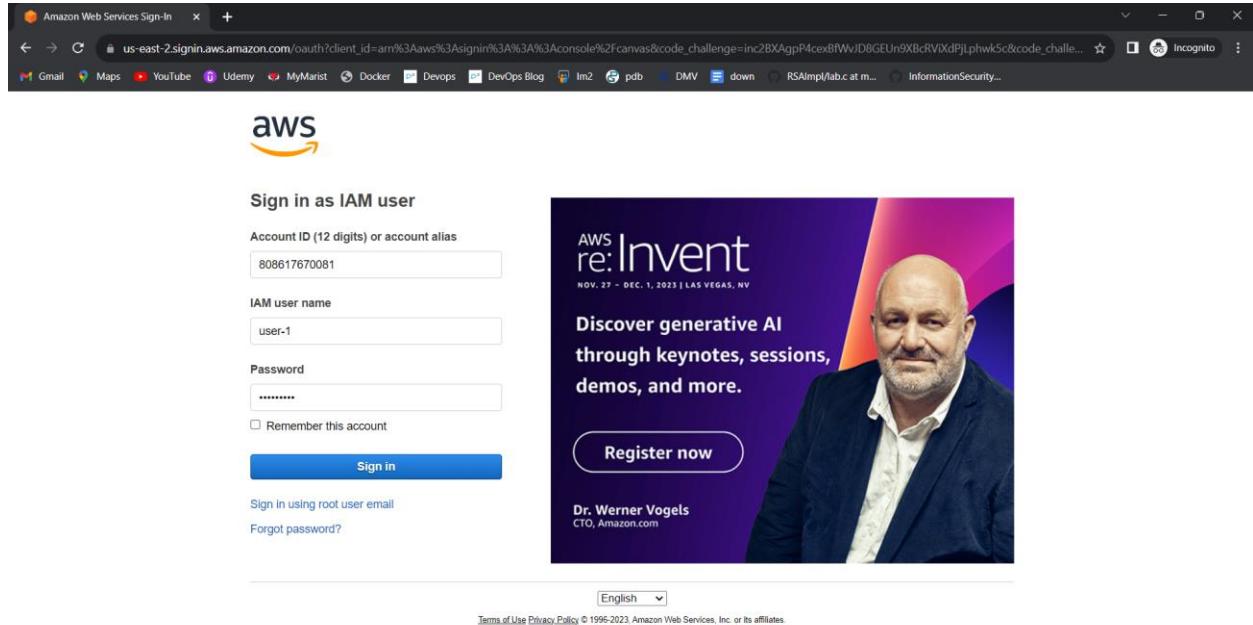
The screenshot shows the AWS IAM User-3 details page. The user was created on October 14, 2023, at 21:41 UTC-04:00. They have one access key (AKIA3YRK60XA6EDMJGTE) which is active and never used. The user is a member of the S3-Support group, which is attached to the AmazonS3ReadOnlyAccess policy.

Group	Attached Policies
S3-Support	AmazonS3ReadOnlyAccess

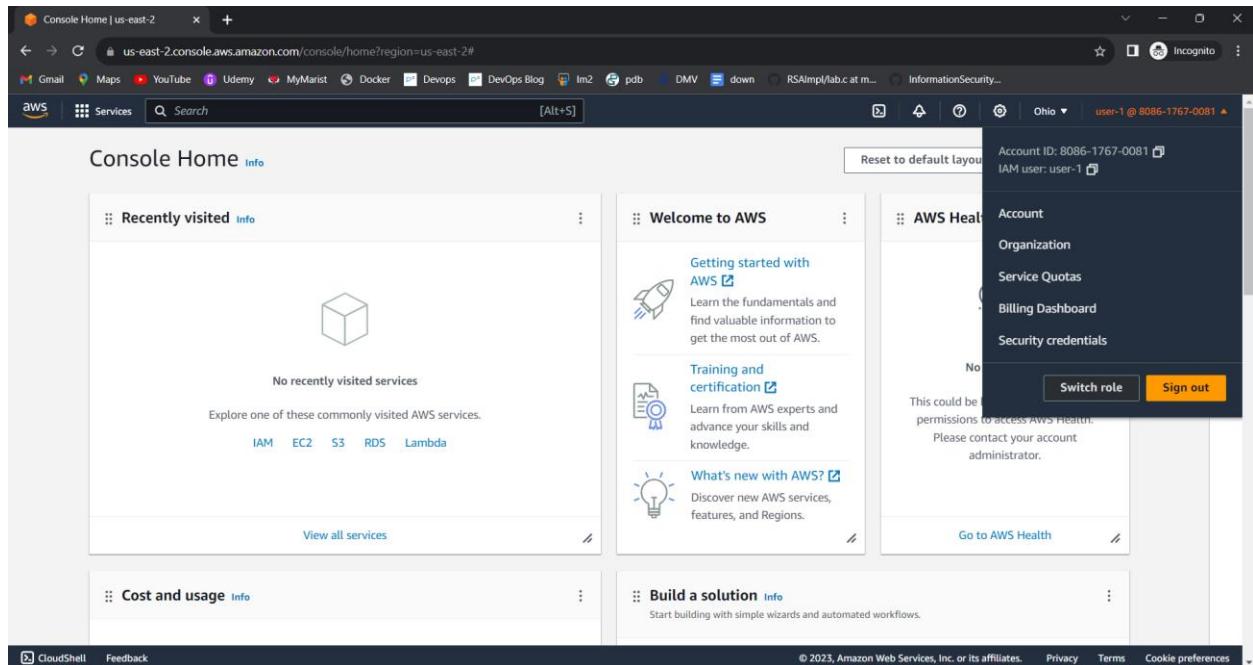
- Once the users are assigned with groups, worked on logging into AWS management console using user-1. For that select user-1 we will see console sign-in link, with that we can login to AWS management console.

- And then we need to add password for user-1 to login to the AWS management console and the click on apply.

- Once password is created, using that console link login to the AWS management console.



- After Successful login to AWS management console, selected few services to verify the access.



- I have selected the EC2 services and able to see the instances.

The screenshot shows the AWS EC2 Dashboard for the US East (Ohio) Region. On the left, a sidebar lists various EC2 services like Instances, Images, and Elastic Block Store. The main area displays resource counts: 0 Instances (running), 0 Auto Scaling Groups, 0 Dedicated Hosts, 0 Elastic IPs, 0 Instances, 0 Key pairs, 0 Load balancers, 0 Placement groups, 1 Security groups, 0 Snapshots, and 0 Volumes. To the right, the 'Account attributes' section shows the Default VPC (vpc-084c97fc14c682e0a) and other settings like Data protection and security, Zones, EC2 Serial Console, Default credit specification, and Console experiments. A 'Launch instance' button is also present.

- I have selected the S3 the see the list of buckets.

The screenshot shows the AWS S3 Bucket details page for 'samplebucket--f5858d50'. The left sidebar includes options for Buckets, Storage Lens, and AWS Marketplace for S3. The main content area shows the bucket name and a summary table with columns for Name, Type, Last modified, Size, and Storage class. A message indicates 'No objects' and 'You don't have any objects in this bucket.' A prominent 'Upload' button is at the bottom.

- As the User-1 has only EC2 and S3 access tried to access other services which user-1 doesn't have access to it. I have selected IAM the It showed Access denied.

The screenshot shows the AWS IAM Dashboard. On the left, a sidebar lists 'Access management' options like User groups, Roles, Policies, and Identity providers. The main area is titled 'IAM Dashboard' and contains sections for 'IAM resources' and 'AWS Account'. Both sections display 'Access denied' messages. The 'IAM resources' message is for 'iam:GetAccountSummary' and the 'AWS Account' message is for 'iam>ListAccountAliases'. Both messages provide troubleshooting links and copy buttons.

- I have tried to create the user when I have first logged into the AWS I have tried to create user with basic details . Then It showed user doesn't have access to create user.

The screenshot shows the 'Specify user details' step of the 'Create user' wizard. The left sidebar shows steps 1 through 4. Step 1 is selected. The main area has a 'User details' section with a 'User name' field containing 'raghu'. A note says the name can be up to 64 characters. A checked checkbox says 'Provide user access to the AWS Management Console - optional'. A callout box asks if you want to provide console access, explaining it's recommended for Identity Center. Below are fields for 'Console password' (radio buttons for 'Autogenerated' or 'Custom'), and a 'Custom password' input field with a placeholder 'Enter a custom password for the user.'

The screenshot shows the 'Create user | IAM' step 1 of 3. It asks if you want to provide console access to a person. A note says: 'We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications. To do so, sign into the console by using the credentials of the management account in AWS Organizations, and then enable Identity Center. If you aren't the management account owner, contact the owner to perform this task.' Below this, there are two password options: 'Autogenerated password' (radio button) and 'Custom password' (radio button, selected). A password field contains '*****'. Below it are password requirements: 'Must be at least 8 characters long' and 'Must include at least three of the following mix of character types: uppercase letters (A-Z), lowercase letters (a-z), numbers (0-9), and symbols ! @ # \$ % ^ & () _ + - (hyphen) = [] { } { } !'. There is also a 'Show password' checkbox. A note below says: 'Users must create a new password at next sign-in - Recommended' and 'Users automatically get the IAMUserChangePassword policy to allow them to change their own password.' A note at the bottom says: 'If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keypairs, you can generate them after you create this IAM user. [Learn more](#)'.

The screenshot shows the 'Create user | IAM' step 2 of 3. It's titled 'Set permissions'. On the left, a sidebar shows steps: Step 2 (selected), Step 3 (Review and create), and Step 4 (Retrieve password). The main area is titled 'Permissions options' and has three choices: 'Add user to group' (radio button), 'Copy permissions' (radio button, selected), and 'Attach policies directly'. Below this is a 'Users (4)' section with a search bar and a table:

User name	Groups	Attached policies
awsstudent	None	User: arn:aws:sts::808617670081:assumed-role/vo
user-1	None	None
user-2	None	None
user-3	None	None

At the bottom, there is a note: '► Set permissions boundary - optional'.

Permissions options

- Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
- Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.
- Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.

Permissions policies (1/1133)

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

Policy name	Type	Attached entities
AccessAnalyzerServiceRolePolicy	AWS managed	0
AdministratorAccess	AWS managed - job function	1
AdministratorAccess-Amplify	AWS managed	0
AdministratorAccess-AWSElasticBea...	AWS managed	0
AlexaForBusinessDeviceSetup	AWS managed	0

User details

User name raghu	Console password type Custom password	Require password reset No
--------------------	--	------------------------------

Permissions summary

Name	Type	Used as
AdministratorAccess	AWS managed - job function	Permissions policy

Tags - optional

Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

The screenshot shows the AWS IAM 'Create user' wizard at Step 3: 'Review and create'. A modal dialog box titled 'Leave site?' is displayed, asking if changes made may not be saved. The main form shows a user named 'raghu' with a console password type of 'Custom password' and 'Require password' set to 'No'. In the 'Permissions summary' section, the 'AdministratorAccess' AWS managed - job function is listed. The 'Tags - optional' section is empty. At the bottom right, there are 'Cancel', 'Previous', and 'Create user' buttons.

The screenshot shows the AWS IAM 'user-1' details page. The 'Summary' section displays the ARN (arn:aws:iam:808617670081:user/spl66/user-1), console access status (Enabled without MFA), and two access keys. The 'Groups' tab is selected under 'Permissions', showing the user is a member of three groups: EC2-Admin, EC2-Support, and S3-Support. Attached policies for these groups include AmazonEC2ReadOnlyAccess and AmazonS3ReadOnlyAccess. The left sidebar shows navigation links for Identity and Access Management (IAM) like User groups, Roles, Policies, and Groups.

How would you rate your experience with this service console? ★ ★ ★ ★ ★

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management

- User groups
- Users**
- Roles
- Policies
- Identity providers
- Account settings

Access reports

- Access analyzer
- Archive rules
- Analyzers
- Settings

CloudShell Feedback

Permissions Groups (3) Tags (1) Security credentials Access Advisor

Console sign-in

Console sign-in link: https://808617670081.signin.aws.amazon.com/console

Console password: Updated 1 hour ago (2023-10-13 23:44 EDT)
Last console sign-in: Never

Manage console access

Multi-factor authentication (MFA) (0)

No MFA devices. Assign an MFA device to improve the security of your AWS environment.

Assign MFA device

Device type Identifier Certifications Created on

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

How would you rate your experience with this service console? ★ ★ ★ ★ ★

Step 2

Set up device

MFA device name

Device name: Enter a meaningful name to identify this device.
raghu
Maximum 128 characters. Use alphanumeric and '-' characters.

MFA device

Select an MFA device to use, in addition to your username and password, whenever you need to authenticate.

Authenticator app
Authenticate using a code generated by an app installed on your mobile device or computer.

Security Key
Authenticate using a code generated by touching a YubiKey or other supported FIDO security key.

CloudShell Feedback

© 2023, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

The screenshot shows two consecutive steps of the AWS IAM 'Create access key' wizard.

Step 1: Access key best practices & alternatives

This step provides guidance on using long-term credentials like access keys to improve security. It lists five use cases:

- Command Line Interface (CLI)**: You plan to use this access key to enable the AWS CLI to access your AWS account.
- Local code**: You plan to use this access key to enable application code in a local development environment to access your AWS account.
- Application running on an AWS compute service**: You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.
- Third-party service**: You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.
- Application running outside AWS**: You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

Step 2 - optional: Set description tag

Step 3: Retrieve access keys

You need permissions

You do not have the permission required to perform this operation. Ask your administrator to add permissions.

Retrieve access keys

This step shows that there are no resources available for retrieval.

Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the [best practices for managing AWS access keys](#).

Buttons at the bottom:

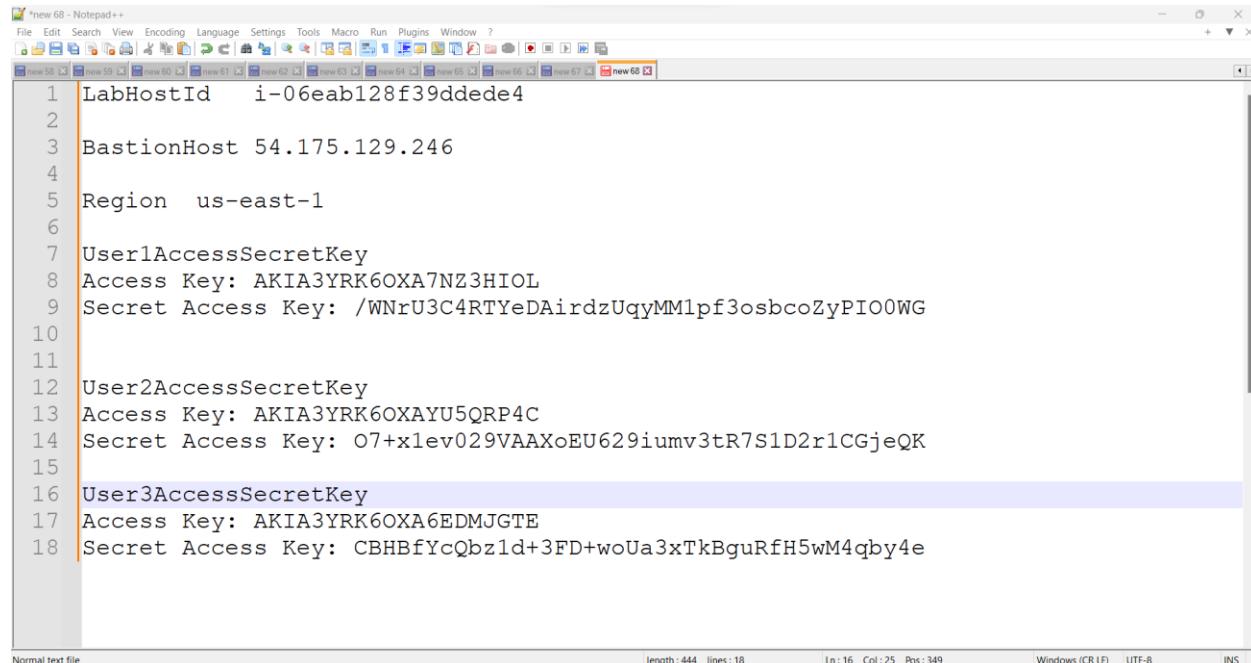
- Download .csv file
- Done

Task II:

- For Task II, check the user access key and secret key selecting details and then click on show. It will display all the details as shown in below screenshot.

UserAccessSecretKey	Access Key	Secret Access Key
UserAccessSecretKey	AKIA3YRK60XA7NZ3HIOL	Secret Access Key: /WNNU3C4RTYeDairdzUqyMMlpf3osbccZyPIOOWG
User2AccessSecretKey	AKIA3YRK60XAVI5QRP4C	Secret Access Key: o7+xlev029VAAKoEU629iumv3tR7S1D2r1CGjeQK
User3AccessSecretKey	AKIA3YRK60XA6EDMJGTE	Secret Access Key: CBHBFYcqzb1d+3FD+woUa3xTkBguRFH5wM4qby4e

- Copied all the details into a note pad and download the ppk file to connect to the bastion node using putty.

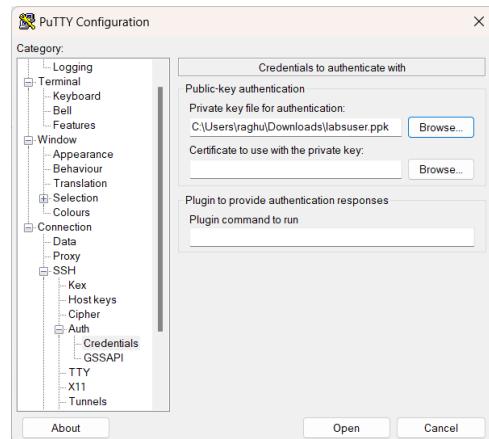


```

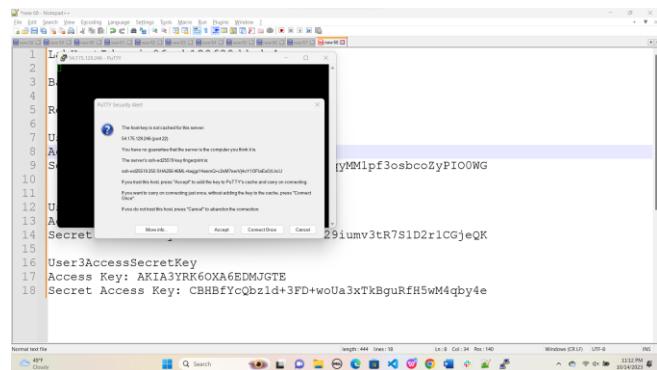
1 LabHostId i-06eab128f39ddede4
2
3 BastionHost 54.175.129.246
4
5 Region us-east-1
6
7 User1AccessSecretKey
8 Access Key: AKIA3YRK60XA7NZ3HIOL
9 Secret Access Key: /WNrU3C4RTYedAirdzUqyMMlpf3osbcoZyPIO0WG
10
11 User2AccessSecretKey
12 Access Key: AKIA3YRK60XAYU5QRP4C
13 Secret Access Key: O7+xlev029VAAXoEU629iumv3tR7S1D2r1CGjeQK
14
15 User3AccessSecretKey
16 Access Key: AKIA3YRK60XA6EDMJGTE
17 Secret Access Key: CBHBFYcQbz1d+3FD+woUa3xTkBguRfH5wM4qby4e

```

- After entering the bastion IP in connection and then select the ppk file and then click on open.



- Once we click on open then a pop will appear, then click on accept.



- Once login to the bastion configured AWS profile for user-1,user-2, user-3.

```

1 LabHostId i-06eab128f39ddede4
2
3 BastionHost 54.175.129.246
4
5 Region us-east-1
6
7 User1AccessSecretKe
8 Access Key: AKIA3YR
9 Secret Access Key:
10
11
12 User2AccessSecretKe
13 Access Key: AKIA3YR
14 Secret Access Key:
15
16 User3AccessSecretKe
17 Access Key: AKIA3YR
18 Secret Access Key: CBHBFYcQbz1d+3FD+woUa3xTkBguRfH5wM4qby4e

```

- Below screenshot shows the configuration of AWS profiles for user-1,user-2,user-3.

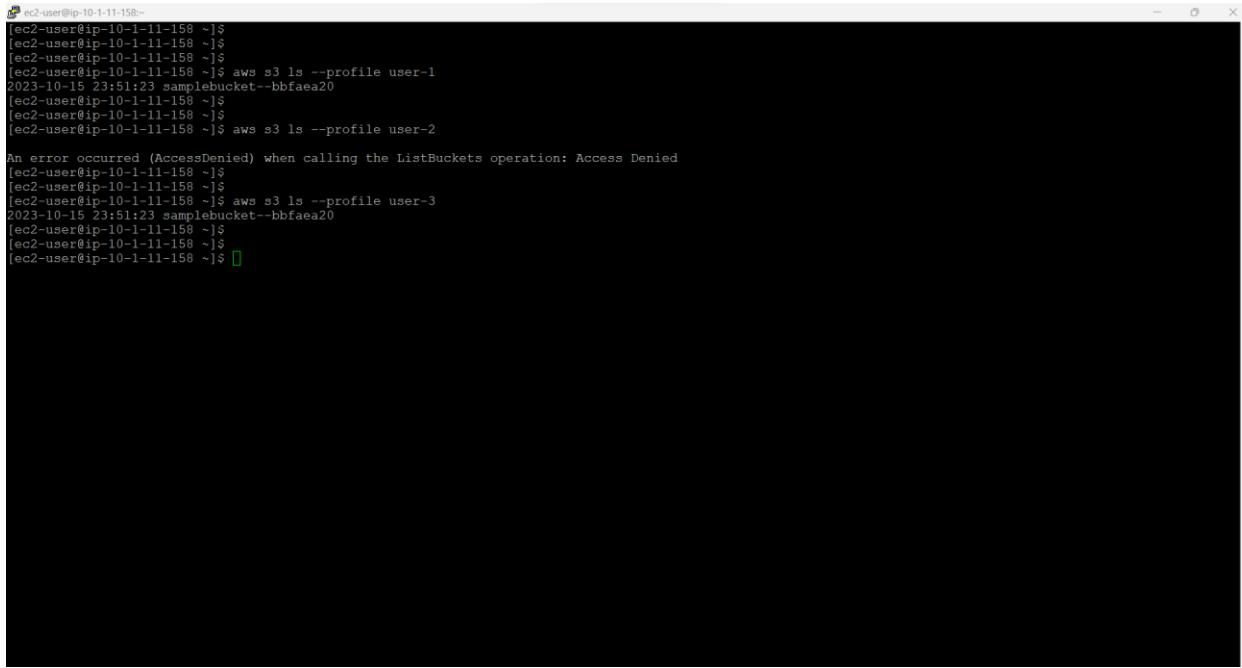
```

1 LabHostId i-06eab128f39ddede4
2
3 BastionHost 54.175.129.246
4
5 Region us-east-1
6
7 User1AccessSecretKe
8 Access Key: AKIA3YR
9 Secret Access Key: /Wn1334NTT86L1dz0igMlpC3obcdByF10WG
10
11 User2AccessSecretKe
12 Access Key: AKIA3YR
13 Secret Access Key: 07+kiev029VAM:EU029jomy9cR721D2z1C6jegK
14
15 User3AccessSecretKe
16 Access Key: AKIA3YR
17 Secret Access Key: CBHBFYcQbz1d+3FD+woUa3xTkBguRfH5wM4qby4e
18

```

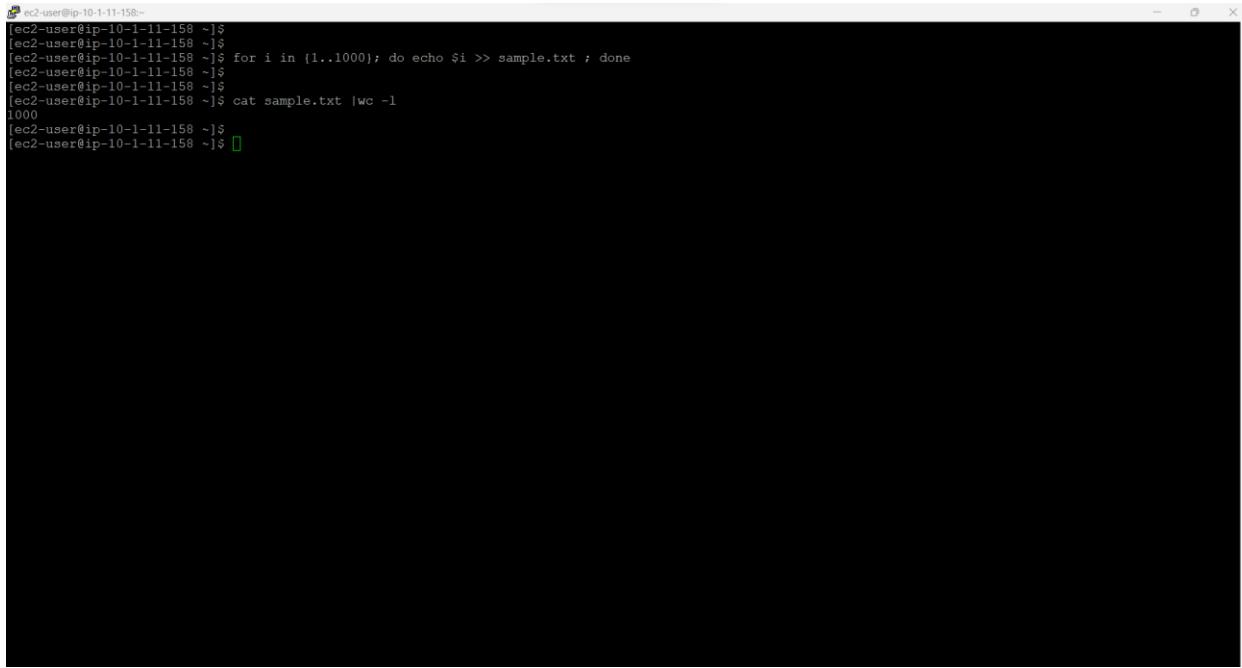
- After successful configuration of profile checking the users access on few AWS services.
- Below screenshot show that we have performed list S3 bucket with 3 users.
- User-1 and user-3 has read access to S3 buckets so that it showing list of buckets.

- User-2 has only read access to EC2. That why access denied for user-2.



```
[ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ aws s3 ls --profile user-1 2023-10-15 23:51:23 samplebucket--bbfaea20 [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ aws s3 ls --profile user-2 An error occurred (AccessDenied) when calling the ListBuckets operation: Access Denied [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ aws s3 ls --profile user-3 2023-10-15 23:51:23 samplebucket--bbfaea20 [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$
```

- I have created a sample.txt with 100- lines in it using for loop to insert data in it.



```
[ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ for i in {1..1000}; do echo $i >> sample.txt ; done [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ cat sample.txt |wc -l 1000 [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$
```

- Once the sample.txt is created tried to upload it to S3 bucket with user-1 and user-3 , it denied because both users have read only access.
- So uploaded the file to S3 bucket with default user and it was successful.
- The tries to list the objects in the bucket with user-1 and user-3 it was successful.

```

[ec2-user@ip-10-1-11-158 ~]$ aws s3 cp sample.txt s3://samplebucket--bbfaea20/
upload: ./sample.txt to s3://samplebucket--bbfaea20/sample.txt
[ec2-user@ip-10-1-11-158 ~]$ aws s3 ls s3://samplebucket--bbfaea20/ --profile user-1
2023-10-16 00:23:02      3893 sample.txt
[ec2-user@ip-10-1-11-158 ~]$ aws s3 ls s3://samplebucket--bbfaea20/ --profile user-2
An error occurred (AccessDenied) when calling the ListObjectsV2 operation: Access Denied
[ec2-user@ip-10-1-11-158 ~]$ aws s3 ls s3://samplebucket--bbfaea20/ --profile user-3
2023-10-16 00:23:02      3893 sample.txt
[ec2-user@ip-10-1-11-158 ~]$ aws s3 ls s3://samplebucket--bbfaea20/ --profile user-2
[ec2-user@ip-10-1-11-158 ~]$ 

```

- User-1 has full access to EC2 so tried to describe an instance. Below is screenshot showing successful result of it.

```

[ec2-user@ip-10-1-11-158 ~]$ aws ec2 describe-instance-status --instance-ids i-0279fd7f3eld5f59c --profile user-1
{
    "InstanceStatuses": [
        {
            "AvailabilityZone": "us-east-1a",
            "InstanceId": "i-0279fd7f3eld5f59c",
            "InstanceState": {
                "Code": 16,
                "Name": "running"
            },
            "InstanceStatus": {
                "Details": [
                    {
                        "Name": "reachability",
                        "Status": "passed"
                    }
                ],
                "Status": "ok"
            },
            "SystemStatus": {
                "Details": [
                    {
                        "Name": "reachability",
                        "Status": "passed"
                    }
                ],
                "Status": "ok"
            }
        }
    ]
}
[ec2-user@ip-10-1-11-158 ~]$ 

```

- Tried to describe instance with user-2 as it has EC2 read access, below screenshot shows successful result of it.

```
[ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ aws ec2 describe-instance-status --instance-ids i-0279fd7f3e1d5f59c --profile user-2
{
    "InstanceStatuses": [
        {
            "AvailabilityZone": "us-east-1a",
            "InstanceId": "i-0279fd7f3e1d5f59c",
            "InstanceState": {
                "Code": 16,
                "Name": "running"
            },
            "InstanceStatus": {
                "Details": [
                    {
                        "Name": "reachability",
                        "Status": "passed"
                    }
                ],
                "Status": "ok"
            },
            "SystemStatus": {
                "Details": [
                    {
                        "Name": "reachability",
                        "Status": "passed"
                    }
                ],
                "Status": "ok"
            }
        }
    ]
}
[ec2-user@ip-10-1-11-158 ~]$
```

- Tried to list IAM roles with user-1, user-2, user-3 as it doesn't have any access to IAM its access denies, below screenshot shows the result of it.

```
[ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ aws iam list-roles --profile user-1
An error occurred (AccessDenied) when calling the ListRoles operation: User: arn:aws:iam::808617670081:user/spl66/user-1 is not authorized to perform: iam:ListRoles on resource: arn:aws:iam::808617670081:role/ because no identity-based policy allows the iam:ListRoles action
[ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ aws iam list-roles --profile user-2
An error occurred (AccessDenied) when calling the ListRoles operation: User: arn:aws:iam::808617670081:user/spl66/user-2 is not authorized to perform: iam:ListRoles on resource: arn:aws:iam::808617670081:role/ because no identity-based policy allows the iam:ListRoles action
[ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ aws iam list-roles --profile user-3
An error occurred (AccessDenied) when calling the ListRoles operation: User: arn:aws:iam::808617670081:user/spl66/user-3 is not authorized to perform: iam:ListRoles on resource: arn:aws:iam::808617670081:role/ because no identity-based policy allows the iam:ListRoles action
[ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$
```

- Tried the same list roles with default user then it was successful, below screenshot shows the result of it



```
[ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ [ec2-user@ip-10-1-11-158 ~]$ aws iam list-users
{
    "Users": [
        {
            "Path": "/",
            "UserName": "awsstudent",
            "UserId": "AIDA3YRK60XA4US6BS6PH",
            "Arn": "arn:aws:iam::808617670081:user/awsstudent",
            "CreateDate": "2023-10-15T23:51:58+00:00"
        },
        {
            "Path": "/spl66/",
            "UserName": "user-1",
            "UserId": "AIDA3YRK60XA56TSQ6A3P",
            "Arn": "arn:aws:iam::808617670081:user/spl66/user-1",
            "CreateDate": "2023-10-15T23:51:22+00:00"
        },
        {
            "Path": "/spl66/",
            "UserName": "user-2",
            "UserId": "AIDA3YRK60XASNRUPSYDC",
            "Arn": "arn:aws:iam::808617670081:user/spl66/user-2",
            "CreateDate": "2023-10-15T23:51:21+00:00"
        },
        {
            "Path": "/spl66/",
            "UserName": "user-3",
            "UserId": "AIDA3YRK60XASJ73WNN75",
            "Arn": "arn:aws:iam::808617670081:user/spl66/user-3",
            "CreateDate": "2023-10-15T23:51:21+00:00"
        }
    ]
}
[ec2-user@ip-10-1-11-158 ~]$
```

Conclusion:

In this Lab I have worked on AWS IAM service which is Identity Access Management. I have worked on assigning the groups to the users and verified them using the Aws management console and AWS CLI. With the pre created user tried the access to specific services it worked for the services for which user has access to it, and access was denied for the services where users doesn't have access to it. So that we can conclude that a user created in IAM has not access to every service, but they have access to the services which we have provided to them. We can limit the access to the users by adding policies, groups, or roles.