

# IAM

**Description** - Identity and access management (IAM) in enterprise IT is about defining and managing the roles and access privileges of individual network users and the circumstances in which users are granted (or denied) those privileges. Those users might be customers (customer identity management) or employees (employee identity management). The core objective of IAM systems is one digital identity per individual. Once that digital identity has been established, it must be maintained, modified and monitored throughout each user's "access lifecycle."

**Need of IAM** - Identity and access management is a critical part of any enterprise security plan, as it is inextricably linked to the security and productivity of organizations in today's digitally enabled economy.

Compromised user credentials often serve as an entry point into an organization's network and its information assets. Enterprises use identity management to safeguard their information assets against the rising threats of ransomware, criminal hacking, phishing and other malware attacks. Global ransomware damage costs alone are expected to exceed \$5 billion this year, up 15 percent from 2016, Cybersecurity Ventures predicted.

In many organizations, users sometimes have more access privileges than necessary. A robust IAM system can add an important layer of protection by ensuring a consistent application of user access rules and policies across an organization.

Identity and access management systems can enhance business productivity. The systems' central management capabilities can reduce the complexity and cost of safeguarding user credentials and access. At the same time, identity management systems enable workers to be more productive (while staying secure) in a variety of environments, whether they're working from home, the office, or on the road.

## Three Typical Systems Used for Identity and Access Management

There are many technologies to simplify password management and other aspects of IAM. A few common types of solutions that are used as part of an IAM program include:

- **Single Sign On (SSO):** An access and login system that allows users to authenticate themselves once and then grants them access to all the software, systems, and data they need without having to log into each of those areas individually.
- **Multi-Factor Authentication:** This system uses a combination of something the user knows (e.g. a password), something the user has (e.g. a security token), and something the user is (e.g. a fingerprint) to authenticate individuals and grant them access.
- **Privileged Access Management:** This system typically integrates with the employee database and pre-defined job roles to establish and provide the access employees need to perform their roles.

IAM technology can be provided on-premises, through a cloud-based model (i.e. identity-as-a-service, or IDaaS), or via a hybrid cloud setup. Practical applications of IAM, and how it is implemented, differ from organization to organization, and will also be shaped by applicable regulatory and compliance initiatives.

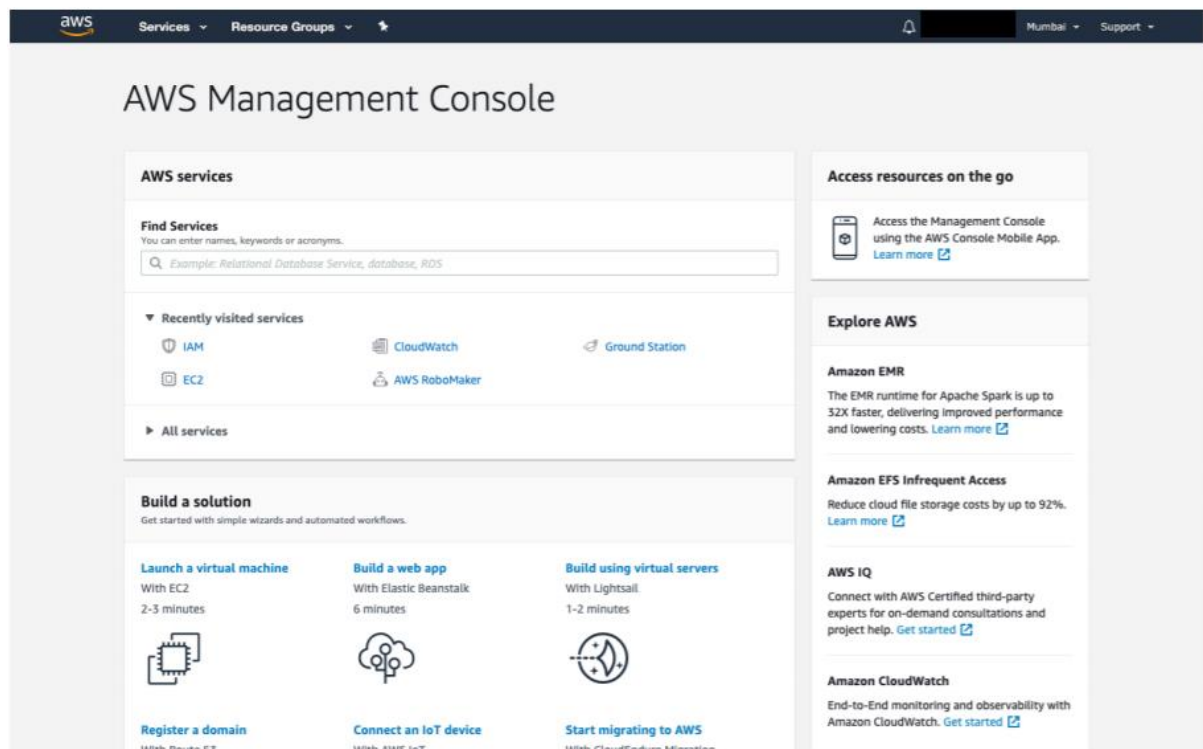
## Policy Types

The following policy types, listed in order of frequency, are available for use in AWS. For more details, see the sections below for each policy type.

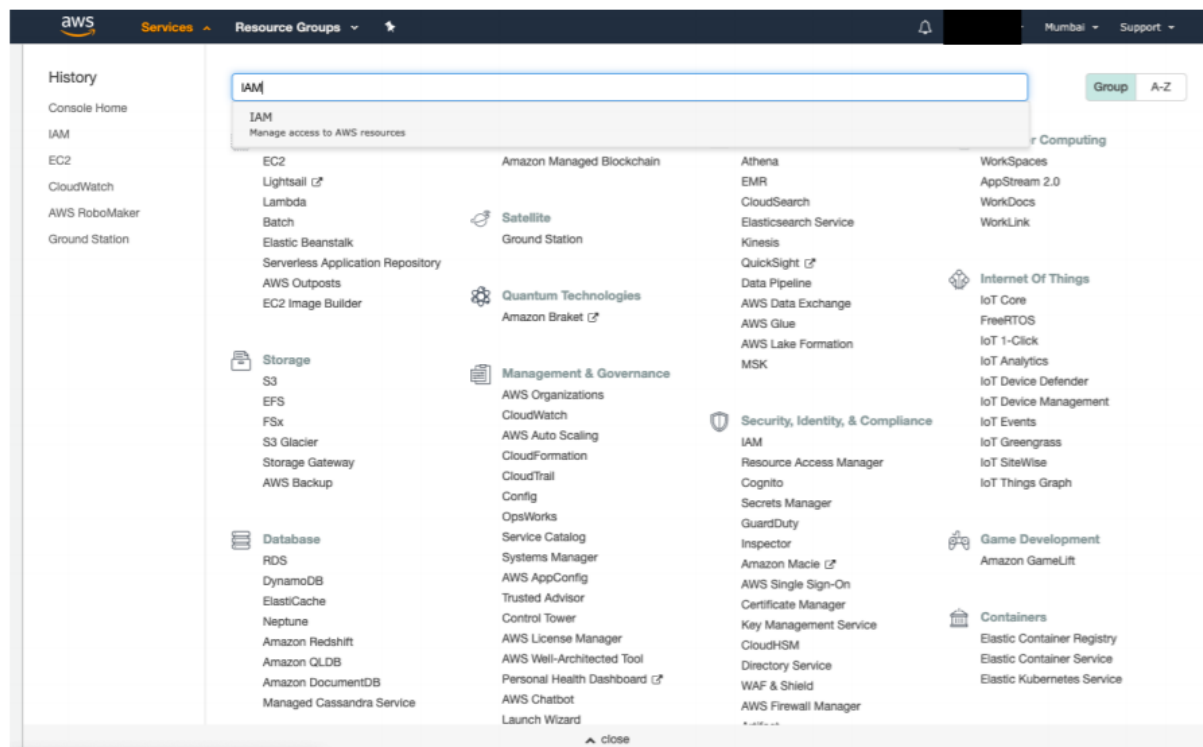
- **Identity-based policies** – Attach managed and inline policies to IAM identities (users, groups to which users belong, or roles). Identity-based policies grant permissions to an identity.
- **Resource-based policies** – Attach inline policies to resources. The most common examples of resource-based 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.
- **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 identitybased 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.
- **Organizations SCPs** – Use an AWS Organizations service control policy (SCP) to define the maximum permissions for account members of an organization or organizational unit (OU). SCPs limit permissions that identity-based policies or resource-based policies grant to entities (users or roles) within the account, but 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. For more information, see Session Policies.

# Create IAM Users in AWS

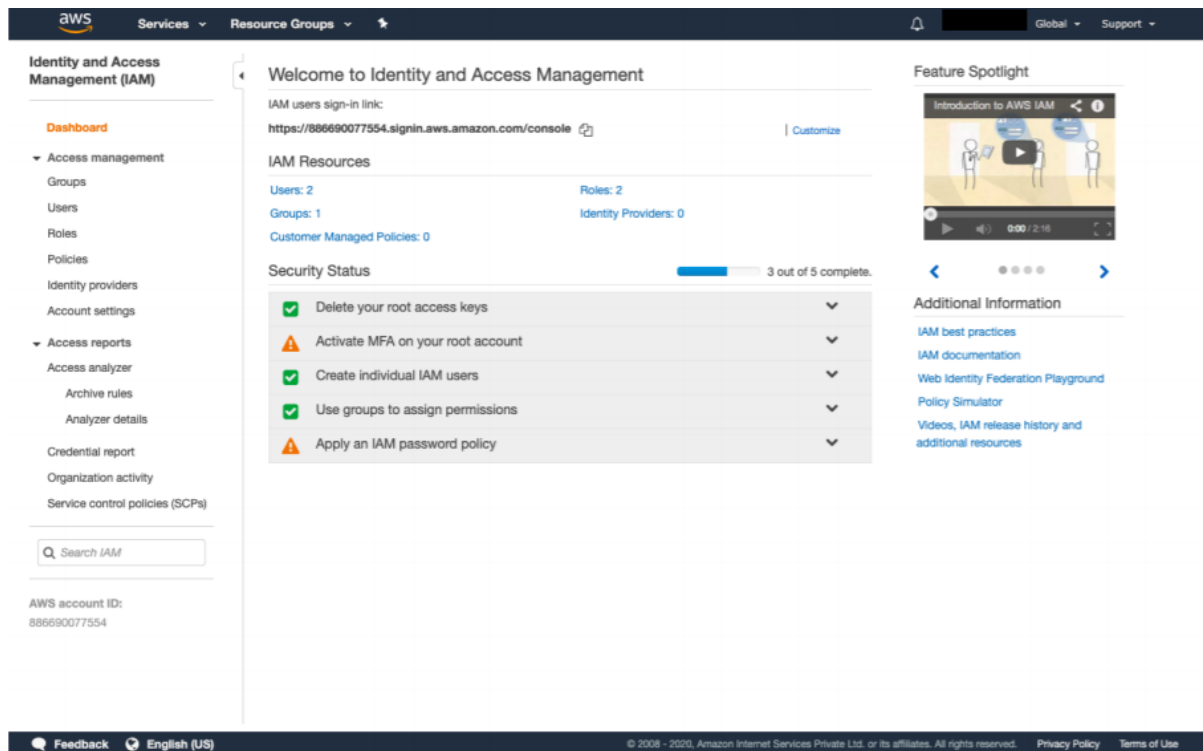
1. Log in to your AWS Account and go to Console



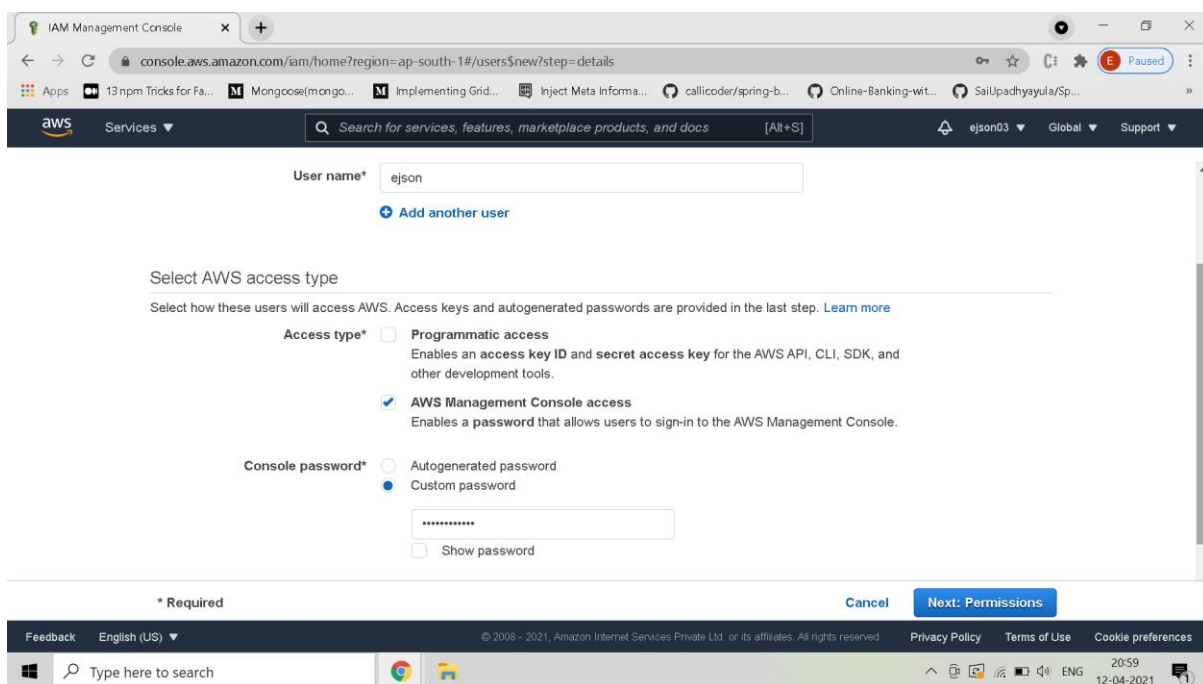
2. Search for IAM in services



### 3. In the IAM menu select users



### 4. Select add new user and enter details



5. Add administrative policy to user one to avoid logging in to root account to perform further operations

The screenshot shows the 'Add user' wizard in the AWS IAM Management Console, specifically the 'Set permissions' step (step 2 of 5). The 'Attach existing policies directly' option is selected. Below, a table lists policies with 'AdministratorAccess' selected.

	Policy name	Type	Used as
<input checked="" type="checkbox"/>	AdministratorAccess	Job function	None
<input type="checkbox"/>	AdministratorAccess-Amplify	AWS managed	None
<input type="checkbox"/>	AdministratorAccess-AWSElasticBeanstalk	AWS managed	None

6. Review and login using url provided and credentials mentioned above

The screenshot shows the 'Review' step of the 'Add user' wizard. It displays the user details for 'ejson' and a permissions summary.

**User details**

User name	ejson
AWS access type	AWS Management Console access - with a password
Console password type	Custom
Require password reset	No
Permissions boundary	Permissions boundary is not set

**Permissions summary**


The following policies will be attached to the user shown above.

Type	Name
Managed policy	AdministratorAccess

Amazon Web Services Sign-In

us-east-1.signin.aws.amazon.com/oauth?SignatureVersion=4&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAJMOATPLHVSJ563XQ&...

13 npm Tricks for Fa...Mongoose(mongo...Implementing Grid...Inject Meta Informa...calliopter/spring-b...Online-Banking-wit...SaiUpadhyayula/Sp...aleksey-lukyanets/...



Sign in as IAM user

Account ID (12 digits) or account alias

085542889984

IAM user name

ejson

Password

.....


Sign in


[Sign in using root user email](#)

[Forgot password?](#)



Amazon Elasticsearch Service

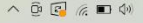
Fully managed real-time log analytics, without the operational overhead






Type here to search



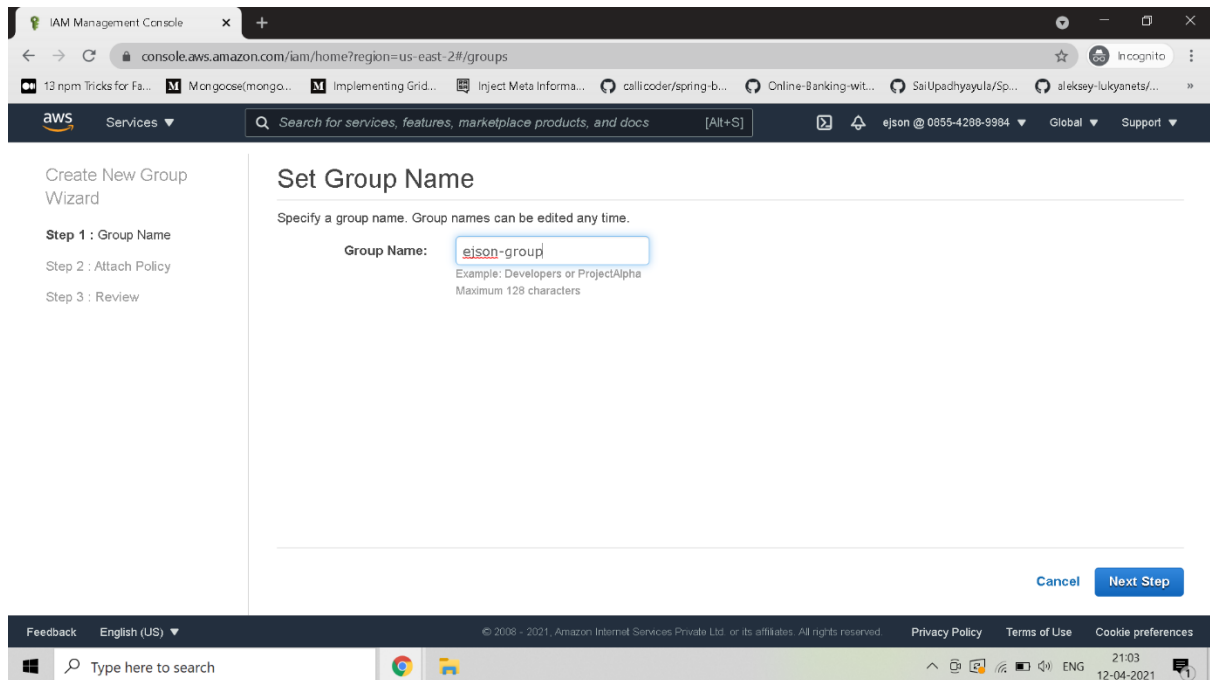


ENG21:0012-04-2021

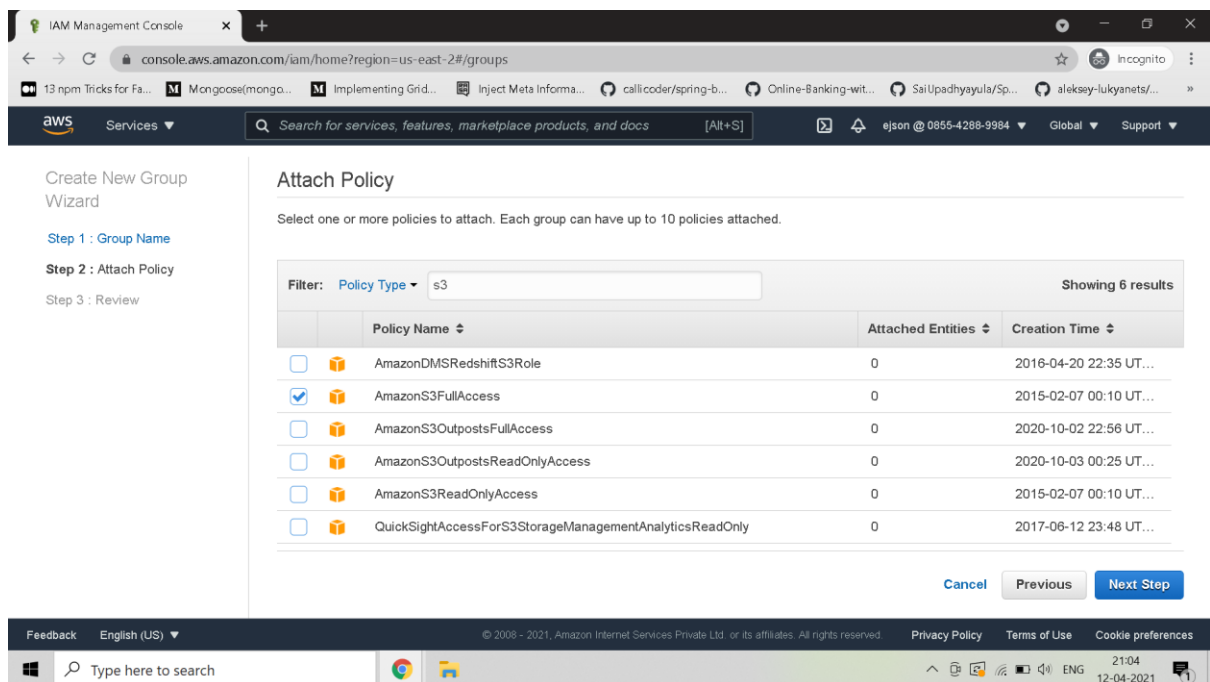


# Create IAM Groups in AWS

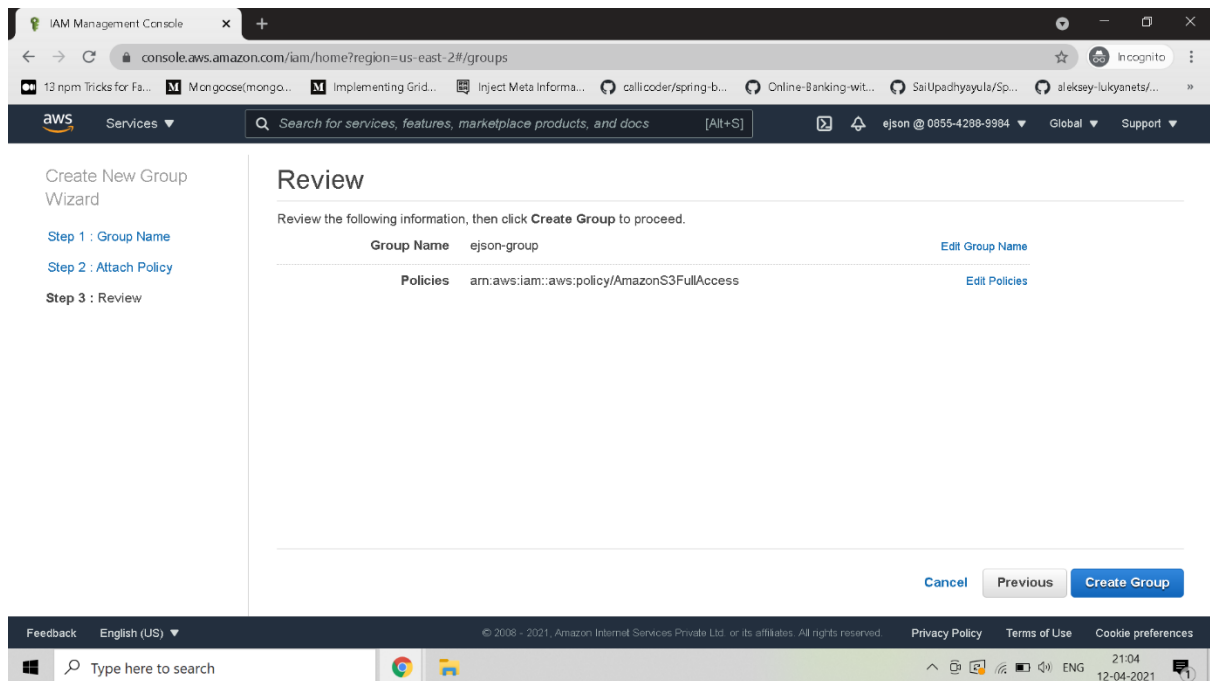
## 1. Select groups and add new group



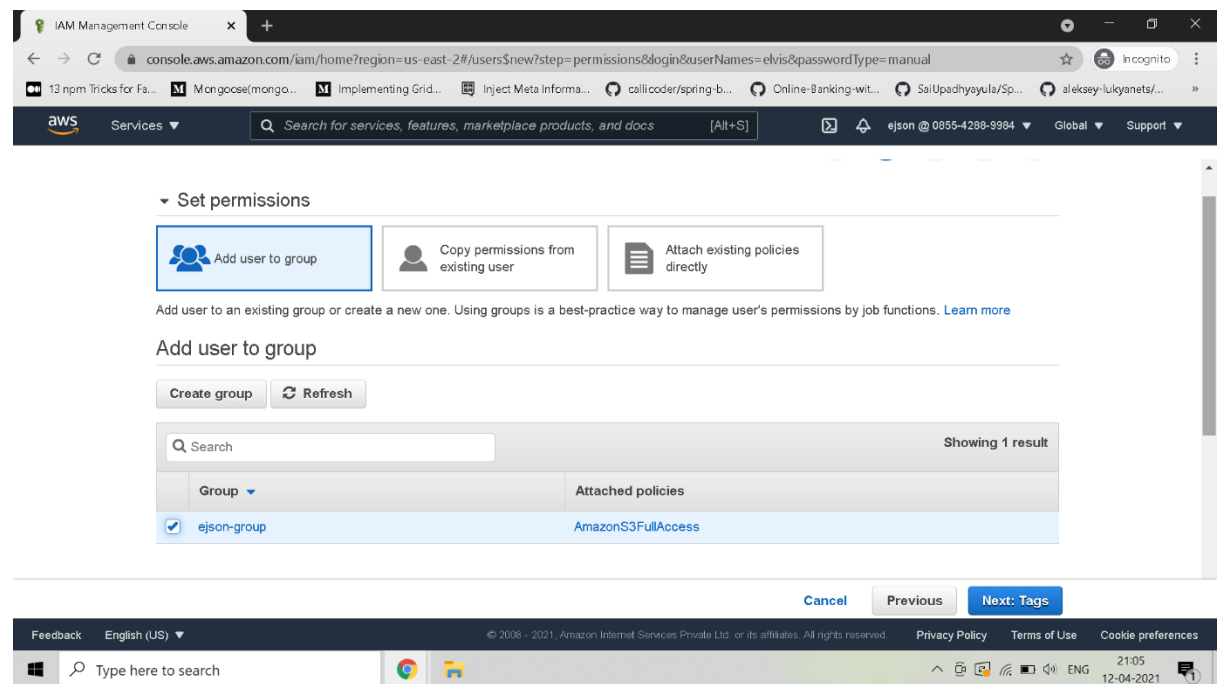
## 2. Attach from the list of policies (in my case S3 access)



### 3. Review

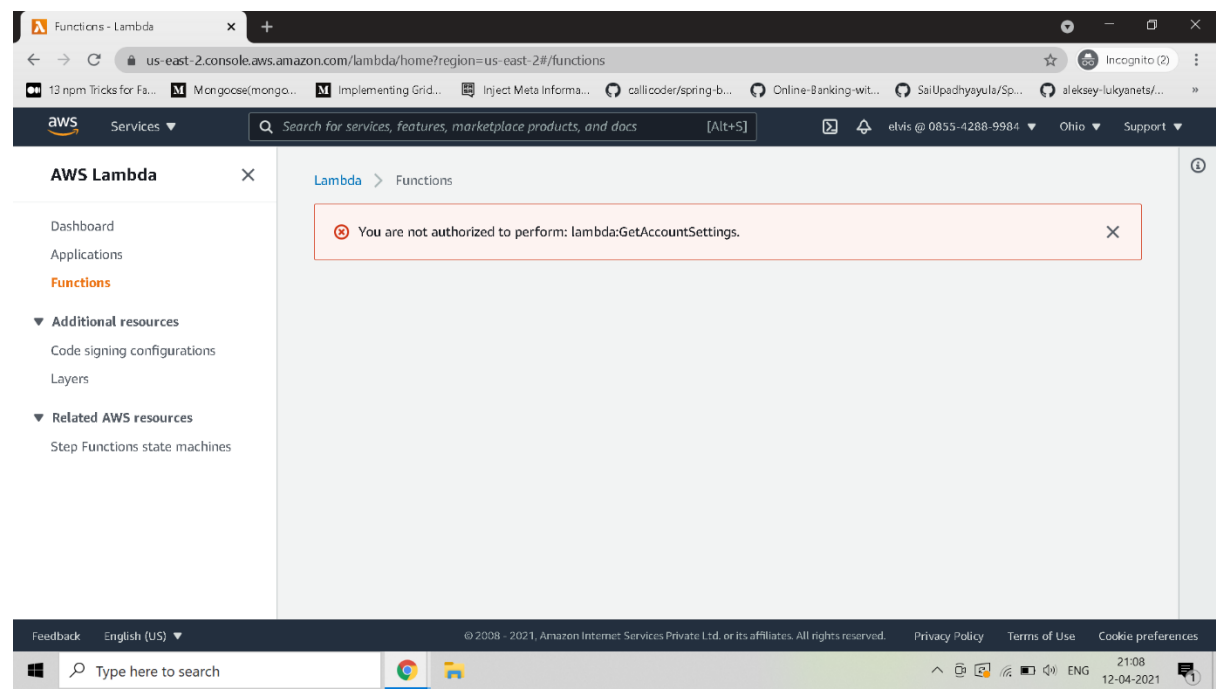


### 4. To test the groups create a new user and add the user to this group





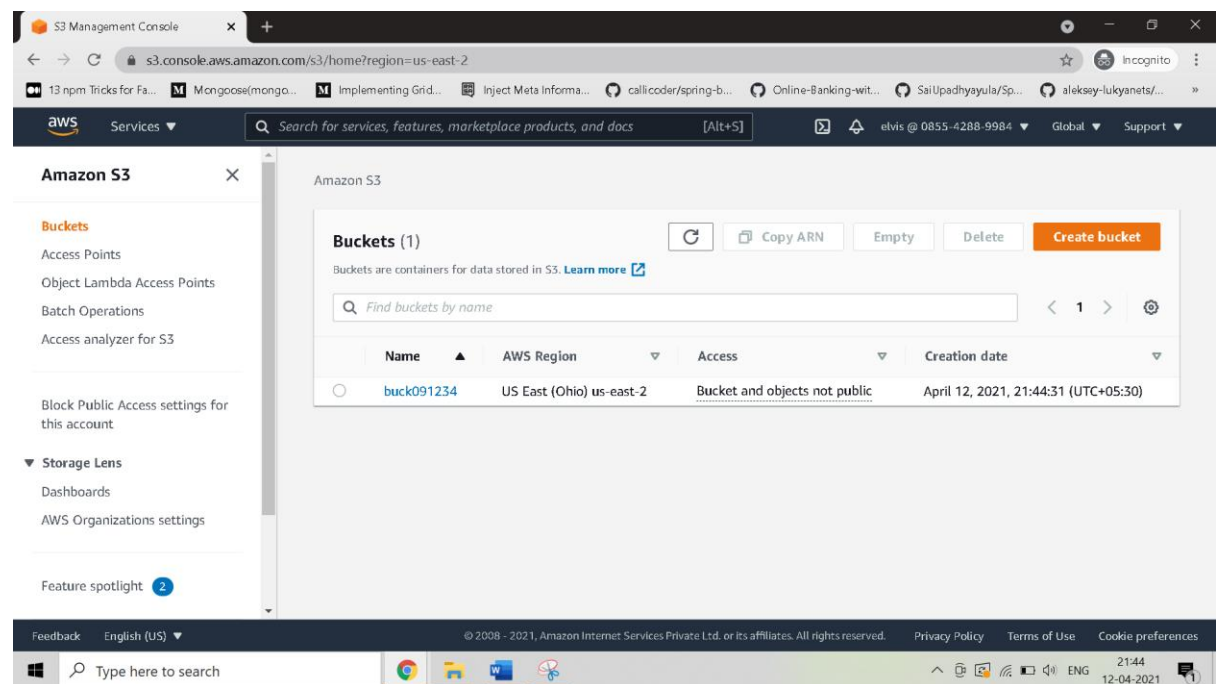
## 5. User cannot access any other service except S3



The screenshot shows the AWS Lambda console in the 'us-east-2' region. A red error box at the top of the main content area states: "You are not authorized to perform: lambda:GetAccountSettings." The left sidebar shows the 'Functions' section is selected. The bottom of the console shows the standard AWS footer with copyright information and links to Privacy Policy, Terms of Use, and Cookie preferences.

Feedback English (US) © 2008 - 2021, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use Cookie preferences

Type here to search



The screenshot shows the Amazon S3 console in the 'us-east-2' region. The 'Buckets (1)' section is active, displaying a table with one bucket. The table has columns for Name, AWS Region, Access, and Creation date. The bucket 'buck091234' is listed with the region 'US East (Ohio) us-east-2' and access level 'Bucket and objects not public'. The creation date is 'April 12, 2021, 21:44:31 (UTC+05:30)'. The left sidebar shows the 'Buckets' section is selected. The bottom of the console shows the standard AWS footer with copyright information and links to Privacy Policy, Terms of Use, and Cookie preferences.

Feedback English (US) © 2008 - 2021, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use Cookie preferences

Type here to search

# Create IAM Groups in AWS

## 1. Select roles and create new role (in our case for EC2 service)

The screenshot shows the AWS IAM Management Console 'Create role' page. The browser address bar shows the URL: `console.aws.amazon.com/iam/home?region=ap-south-1#/roles$new?step=type&commonUseCase=EC2%2BEC2&selectedUseCase=EC2`. The page has a navigation bar with the AWS logo, 'Services' dropdown, a search bar, and user information 'ejson03'. The main content area is titled 'Create role' with a progress indicator showing 4 steps. Step 1, 'Select type of trusted entity', is active. It displays four options: 'AWS service' (selected), 'Another AWS account', 'Web identity', and 'SAML 2.0 federation'. Below these, it says 'Allows AWS services to perform actions on your behalf. Learn more'. Under 'Choose a use case', 'Common use cases' are listed: 'EC2' (selected) and 'Lambda'. The 'EC2' option is highlighted with a blue bar and text: 'Allows EC2 instances to call AWS services on your behalf.' The 'Lambda' option is also listed: 'Allows Lambda functions to call AWS services on your behalf.' At the bottom, there is a 'Cancel' button and a 'Next: Permissions' button. The footer includes 'Feedback', 'English (US)', copyright information, and links to 'Privacy Policy', 'Terms of Use', and 'Cookie preferences'.

## 2. Attach policies to role

The screenshot shows the AWS IAM Management Console 'Create role' page, step 2: 'Attach permissions policies'. The browser address bar shows the URL: `console.aws.amazon.com/iam/home?region=ap-south-1#/roles$new?step=permissions&commonUseCase=EC2%2BEC2&selectedUseCase=EC2`. The page has the same navigation bar as the previous screenshot. The main content area is titled 'Create role' with a progress indicator showing 4 steps. Step 2 is active. It displays the heading 'Attach permissions policies' and the instruction 'Choose one or more policies to attach to your new role.' Below this is a 'Create policy' button. A search bar with 's3' entered shows 'Showing 8 results'. A table lists the policies:

	Policy name	Used as
<input type="checkbox"/>	AmazonDMSRedshiftS3Role	None
<input checked="" type="checkbox"/>	AmazonS3FullAccess	Permissions policy (1)
<input type="checkbox"/>	AmazonS3OutpostsFullAccess	None
<input type="checkbox"/>	AmazonS3OutpostsReadOnlyAccess	None
<input type="checkbox"/>	AmazonS3ReadOnlyAccess	None

At the bottom, there is a 'Cancel' button, a 'Previous' button, and a 'Next: Tags' button. The footer is identical to the previous screenshot.

### 3. Confirm and create role

**Create role**

**Review**

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

**Role name\*** elvis\_role  
Use alphanumeric and '+', '@', '\_' characters. Maximum 64 characters.

**Role description** Allows EC2 instances to call AWS services on your behalf.  
Maximum 1000 characters. Use alphanumeric and '+', '@', '\_' characters.

**Trusted entities** AWS service: ec2.amazonaws.com

**Policies** AmazonS3FullAccess

\* Required

Cancel Previous **Create role**

### 4. Select instance settings and assign IAM role to EC2 instance

**Instances (1/1) Info**

Filter instances

Instance state: running X Clear filters

	Name	Instance ID	Instance state	Instance type
<input checked="" type="checkbox"/>	-	i-0e531c349483d2959	Running	t2.micro

Actions:

- Connect
- View details
- Manage instance state
- Instance settings
- Networking
- Security
- Image and templates
- Monitor and troubleshoot

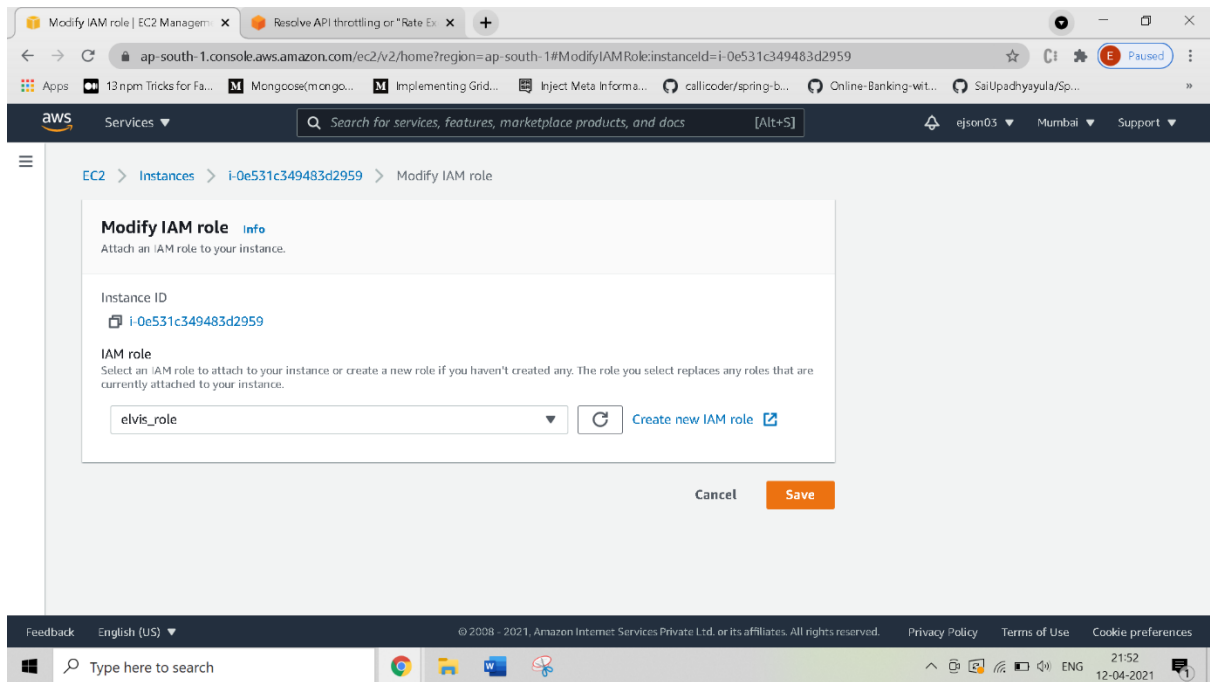
Change security groups

Get Windows password

Modify IAM role

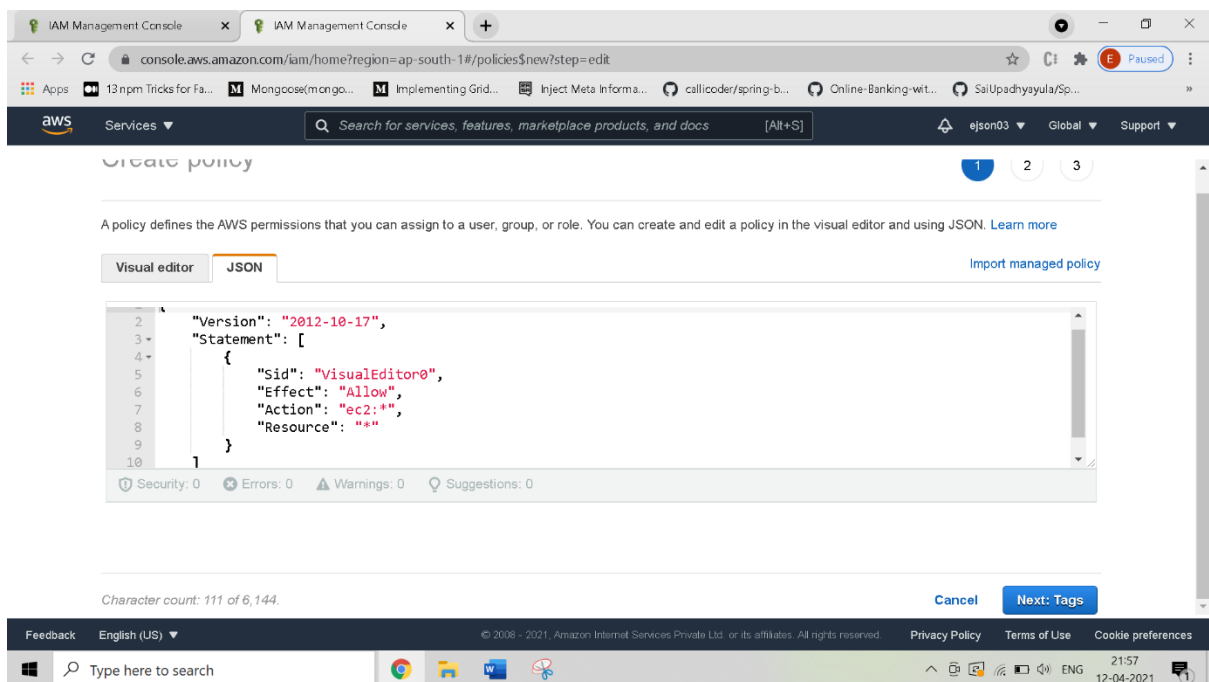
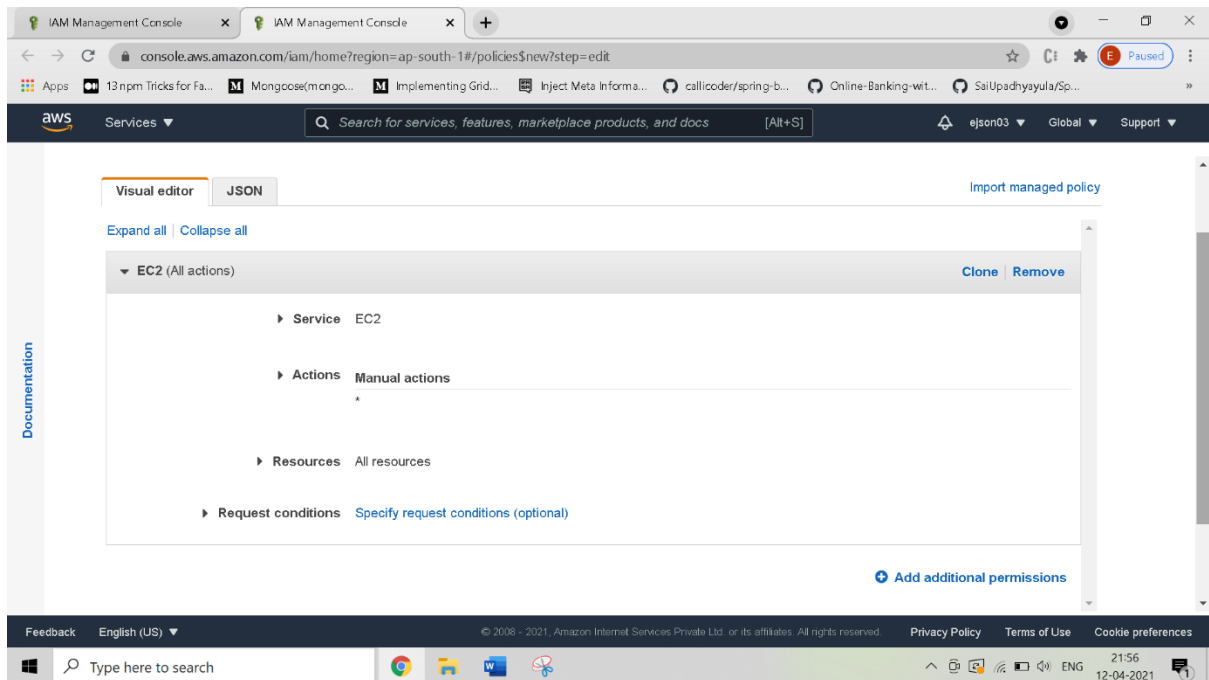
**Instance: i-0e531c349483d2959**

Details Security Networking Storage Status checks Monitoring Tags



# Create IAM Policies in AWS

## 1. Select custom policies and create new policy



## 2. Review

The screenshot shows the AWS IAM Management Console interface. The browser address bar displays the URL: `console.aws.amazon.com/iam/home?region=ap-south-1#/policies$new?step=review`. The page title is "Create policy". In the top right corner, there are three numbered steps: 1, 2, and 3, with step 3 being the active step.

The main section is titled "Review policy". It contains the following fields:

- Name\***: A text input field containing the value `elvis_policy`. Below the field, a note states: "Use alphanumeric and '+', '@', '\_' characters. Maximum 128 characters."
- Description**: A large text area for a description. Below the field, a note states: "Maximum 1000 characters. Use alphanumeric and '+', '@', '\_' characters."
- Summary**: A section containing a search bar labeled "Filter" and a table.

The table has the following structure:

Service	Access level	Resource	Request condition
Allow (1 of 277 services) <a href="#">Show remaining 276</a>			

The bottom of the page features a dark navigation bar with links for "Feedback", "English (US)", "© 2008 - 2021. Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.", "Privacy Policy", "Terms of Use", and "Cookie preferences". The Windows taskbar at the very bottom shows the search bar, task view button, and several open applications (Chrome, File Explorer, Word, and a help icon). The system clock indicates the time is 21:57 on 12-04-2021.

## IAM from AWS CLI

To create an IAM group and add a new IAM user to it

1. Use the `create-group` command to create the group.

```
$ aws iam create-group --group-name MyIamGroup
{
  "Group": {
    "GroupName": "MyIamGroup",
    "CreateDate": "2018-12-14T03:03:52.834Z",
    "GroupId": "AGPAJNUJ2W4IJVEXAMPLE",
    "Arn": "arn:aws:iam::123456789012:group/MyIamGroup",
    "Path": "/"
  }
}
```

2. Use the `create-user` command to create the user.

```
$ aws iam create-user --user-name MyUser
{
  "User": {
    "UserName": "MyUser",
    "Path": "/",
    "CreateDate": "2018-12-14T03:13:02.581Z",
    "UserId": "AIDAJY2PE5XUZ4EXAMPLE",
    "Arn": "arn:aws:iam::123456789012:user/MyUser"
  }
}
```

3. Use the `add-user-to-group` command to add the user to the group.

```
$ aws iam add-user-to-group --user-name MyUser --group-name MyIamGroup
```

4. To verify that the MyIamGroup group contains the MyUser, use the `get-group` command.

```
$ aws iam get-group --group-name MyIamGroup
{
  "Group": {
    "GroupName": "MyIamGroup",
    "CreateDate": "2018-12-14T03:03:52Z",
    "GroupId": "AGPAJNUJ2W4IJVEXAMPLE",
    "Arn": "arn:aws:iam::123456789012:group/MyIamGroup",
    "Path": "/"
  },
}
```

```

    "Users": [
      {
        "UserName": "MyUser",
        "Path": "/",
        "CreateDate": "2018-12-14T03:13:02Z",
        "UserId": "AIDAJY2PE5XUZ4EXAMPLE",
        "Arn": "arn:aws:iam::123456789012:user/MyUser"
      }
    ],
    "IsTruncated": "false"
  }

```

### To attach an IAM managed policy to an IAM user

1. Determine the Amazon Resource Name (ARN) of the policy to attach. The following command uses `list-policies` to find the ARN of the policy with the name `PowerUserAccess`. It then stores that ARN in an environment variable.

```

$ export POLICYARN=$(aws iam list-policies --query
'Policies[?PolicyName==`PowerUserAccess`].{ARN:Arn}' --output text)
~
$ echo $POLICYARN
arn:aws:iam::aws:policy/PowerUserAccess

```

2. To attach the policy, use the `attach-user-policy` command, and reference the environment variable that holds the policy ARN.

```

$ aws iam attach-user-policy --user-name MyUser --policy-arn $POLICYARN

```

3. Verify that the policy is attached to the user by running the `list-attached-user-policies` command.

```

$ aws iam list-attached-user-policies --user-name MyUser
{
  "AttachedPolicies": [
    {
      "PolicyName": "PowerUserAccess",
      "PolicyArn": "arn:aws:iam::aws:policy/PowerUserAccess"
    }
  ]
}

```