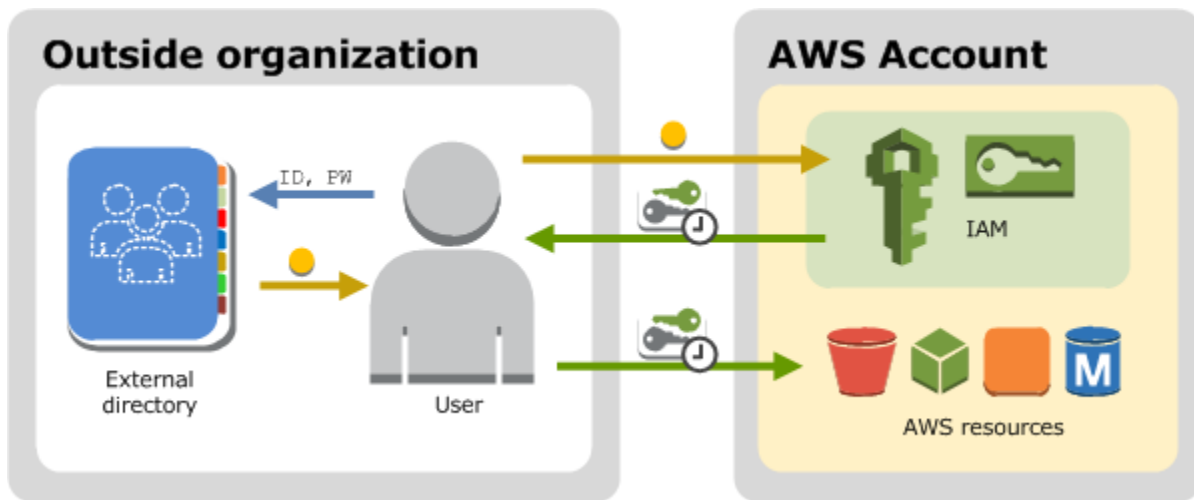


# AWS Lab Work - IAM Roles

**\*Prepared by Sriganesh Pera**

## Intro to IAM [Identity Access Management]

IAM is a web service that lets users' access AWS features in a secured way. Basic illustration is given in the diagram



IAM gives you the following features:

1. **Shared Access:** You can grant permissions to other users to use AWS without sharing password.
2. **Granular permissions:** You can grant different permissions to different people for using the AWS resources (In our group some can only use EC2 and some others can only use Amazon S3).
3. **Secure access to AWS resources for applications that run on Amazon EC2:** Secure access can be given to applications that run on EC2 and use other resources like Amazon S3 buckets or Dynamo DB.

4. **Identity federation:** You can allow users who already have passwords elsewhere and give them temporary access.
5. **Identity information for assurance** : If you use AWS CloudTrail, you receive log records that include information about those who made requests for resources in your account. That information is based on IAM identities.

**IAM can be accessed in 4 ways:**

1. **AWS Management Console:** Web interface in AWS homepage when you logged in.
2. **AWS Command Line Tools:** Jus CLI
3. **AWS SDKs:** AWS provides SDKs (software development kits) that consist of libraries and sample code for various programming languages and platforms (Java, Python, Ruby, .NET, iOS, Android, etc.). The SDKs provide a convenient way to create programmatic access to IAM and AWS.
4. **IAM HTTPS API:** You can access IAM and AWS programmatically by using the IAM HTTPS API, which lets you issue HTTPS requests directly to the service. When you use the HTTPS API, you must include code to digitally sign requests using your credentials

# Lab 1 Objective: Accessing S3 bucket on AWS using Python SDK set up on an EC2 machine with IAM roles

## Lab 1: Setting up an EC2 Instance with IAM Roles and the Python Boto3 SDK

### 1. Click on IAM on AWS console

The screenshot displays the AWS Management Console interface. The top navigation bar includes the AWS logo, a 'Services' dropdown menu, and the user's name 'SRIGANESH PERA' with a location of 'N. Virginia'. The main content area is titled 'Amazon Web Services' and lists various services categorized into Compute, Storage & Content Delivery, Database, Developer Tools, Management Tools, Security & Identity, Internet of Things, Game Development, Mobile Services, and Application Services. The 'Security & Identity' category is highlighted, showing 'Identity & Access Management' as the selected service. The right sidebar contains 'Resource Groups', 'Additional Resources' (including 'Getting Started', 'AWS Console Mobile App', 'AWS Marketplace', and 'AWS re:Invent Announcements'), and 'Service Health'.

Amazon Web Services

Compute

- EC2: Virtual Servers in the Cloud
- EC2 Container Service: Run and Manage Docker Containers
- Elastic Beanstalk: Run and Manage Web Apps
- Lambda: Run Code without Thinking about Servers

Storage & Content Delivery

- S3: Scalable Storage in the Cloud
- CloudFront: Global Content Delivery Network
- Elastic File System: Fully Managed File System for EC2
- Glacier: Archive Storage in the Cloud
- Snowball: Large Scale Data Transport
- Storage Gateway: Hybrid Storage Integration

Database

- RDS: Managed Relational Database Service
- DynamoDB: Managed NoSQL Database
- ElastiCache: In-Memory Cache

Developer Tools

- CodeCommit: Store Code in Private Git Repositories
- CodeDeploy: Automate Code Deployments
- CodePipeline: Release Software using Continuous Delivery

Management Tools

- CloudWatch: Monitor Resources and Applications
- CloudFormation: Create and Manage Resources with Templates
- CloudTrail: Track User Activity and API Usage
- Config: Track Resource Inventory and Changes
- OpsWorks: Automate Operations with Chef
- Service Catalog: Create and Use Standardized Products
- Trusted Advisor: Optimize Performance and Security

Security & Identity

- Identity & Access Management: Manage User Access and Encryption Keys
- Directory Service: Host and Manage Active Directory
- Inspector: Analyze Application Security

Internet of Things

- AWS IoT: Connect Devices to the Cloud

Game Development

- GameLift: Deploy and Scale Session-based Multiplayer Games

Mobile Services

- Mobile Hub: Build, Test, and Monitor Mobile Apps
- Cognito: User Identity and App Data Synchronization
- Device Farm: Test Android, iOS, and Web Apps on Real Devices in the Cloud
- Mobile Analytics: Collect, View and Export App Analytics
- SNS: Push Notification Service

Application Services

- API Gateway: Build, Deploy and Manage APIs
- AppStream: Low Latency Application Streaming
- CloudSearch: Managed Search Service
- Elastic Transcoder: Easy-to-Use Scalable Media Transcoding
- SES: Email Sending and Receiving Service

Resource Groups [Learn more](#)

A resource group is a collection of resources that share one or more tags. Create a group for each project, application, or environment in your account.

[Create a Group](#) [Tag Editor](#)

Additional Resources

- [Getting Started](#): Read our documentation or view our training to learn more about AWS.
- [AWS Console Mobile App](#): View your resources on the go with our AWS Console mobile app, available from Amazon Appstore, Google Play, or iTunes.
- [AWS Marketplace](#): Find and buy software, launch with 1-Click and pay by the hour.
- [AWS re:Invent Announcements](#): Explore the next generation of AWS cloud capabilities. See what's new.

Service Health

## 2. Then Click on Role -> Create Role (blue button) -> Name it as 'ec2-admin' and click next step

The screenshot shows the AWS IAM console interface. On the left, a sidebar lists the steps of the 'Create Role' process: Step 1: Set Role Name (active), Step 2: Select Role Type, Step 3: Establish Trust, Step 4: Attach Policy, and Step 5: Review. The main content area is titled 'Set Role Name' and includes the instruction: 'Enter a role name. You cannot edit the role name after the role is created.' Below this, there is a text input field labeled 'Role Name' containing the text 'ec2-admin'. A small note below the field states: 'Maximum 64 characters. Use alphanumeric and '+-.\_@-' characters'. At the bottom right of the main area, there are two buttons: 'Cancel' and 'Next Step'.

## 3. Select role type as Amazon EC2

The screenshot shows the AWS IAM console interface at the 'Select Role Type' step. The left sidebar shows the steps: Step 1: Set Role Name, Step 2: Select Role Type (active), Step 3: Establish Trust, Step 4: Attach Policy, and Step 5: Review. The main content area is titled 'Select Role Type' and features a list of role types under the heading 'AWS Service Roles'. The roles listed are: Amazon EC2 (Allows EC2 instances to call AWS services on your behalf), AWS Directory Service (Allows AWS Directory Service to manage access for existing directory users and groups to AWS services), AWS Lambda (Allows Lambda Function to call AWS services on your behalf), Amazon Redshift (Allows Amazon Redshift Clusters to call AWS services on your behalf), and Amazon API Gateway (Allows API Gateway to call AWS resources on your behalf). Each role has a 'Select' button to its right. Below these, there are two additional options: 'Role for Cross-Account Access' and 'Role for Identity Provider Access'. At the bottom right, there are three buttons: 'Cancel', 'Previous', and 'Next Step'.

## 4. Select Admin Access (so that EC2 instance that we'll set up will have access to all AWS services) and click next step

The screenshot shows the AWS IAM console interface. On the left, a sidebar lists the steps: Step 1: Set Role Name, Step 2: Select Role Type, Step 3: Establish Trust, Step 4: Attach Policy (highlighted), and Step 5: Review. The main content area is titled 'Attach Policy' and includes the instruction: 'Select one or more policies to attach. Each role can have up to 10 policies attached.' Below this is a table of available policies. The first policy, 'AdministratorAccess', is selected with a checked checkbox. The table has columns for Policy Name, Attached Entities, Creation Time, and Edited Time. At the bottom right, there are buttons for 'Cancel', 'Previous', and 'Next Step'.

Filter:	Policy Type	Filter	Showing 204 results	
	Policy Name	Attached Entities	Creation Time	Edited Time
<input checked="" type="checkbox"/>	AdministratorAccess	2	2015-02-07 00:09 UTC+0530	2015-02-07 00:09 UTC+0530
<input type="checkbox"/>	AmazonElasticMapReducefor...	1	2015-02-07 00:11 UTC+0530	2015-05-14 02:57 UTC+0530
<input type="checkbox"/>	AmazonElasticMapReduceRole	1	2015-02-07 00:11 UTC+0530	2016-02-11 04:11 UTC+0530
<input type="checkbox"/>	AmazonAPIGatewayAdminstr...	0	2015-07-09 23:04 UTC+0530	2015-07-09 23:04 UTC+0530
<input type="checkbox"/>	AmazonAPIGatewayInvokeFul...	0	2015-07-09 23:06 UTC+0530	2015-07-09 23:06 UTC+0530
<input type="checkbox"/>	AmazonAPIGatewayPushToCl...	0	2015-11-12 05:11 UTC+0530	2015-11-12 05:11 UTC+0530
<input type="checkbox"/>	AmazonAppStreamFullAccess	0	2015-02-07 00:10 UTC+0530	2015-02-07 00:10 UTC+0530
<input type="checkbox"/>	AmazonAppStreamReadOnly...	0	2015-02-07 00:10 UTC+0530	2015-02-07 00:10 UTC+0530
<input type="checkbox"/>	AmazonCognitoDeveloperAut...	0	2015-03-24 22:52 UTC+0530	2015-03-24 22:52 UTC+0530

## 5. You can review the role we just created and click on Create Role

The screenshot shows the 'Review' step in the AWS IAM console. The sidebar on the left shows Step 5: Review as the current step. The main content area is titled 'Review' and contains the instruction: 'Review the following role information. To edit the role, click an edit link, or click Create Role to finish.' Below this is a summary of the role's configuration: Role Name (ec2-admin), Role ARN (arn:aws:iam::876810897307:role/ec2-admin), Trusted Entities (The identity provider(s) ec2.amazonaws.com), and Policies (arn:aws:iam::aws:policy/AdministratorAccess). Each item has an associated edit link. At the bottom right, there are buttons for 'Cancel', 'Previous', and 'Create Role'.

Role Name	ec2-admin	<a href="#">Edit Role Name</a>
Role ARN	arn:aws:iam::876810897307:role/ec2-admin	
Trusted Entities	The identity provider(s) ec2.amazonaws.com	
Policies	arn:aws:iam::aws:policy/AdministratorAccess	<a href="#">Change Policies</a>

# Now launch EC2 instance and see how this 'ec2-admin' role gives us access in using all AWS resources

## 6.Go to EC2 on the AWS console and click EC2

The screenshot shows the AWS Management Console interface. The browser address bar displays `https://console.aws.amazon.com/console/home?region=us-east-1#`. The console header includes the AWS logo, navigation tabs for 'AWS', 'Services', and 'Edit', and user information for 'SRIGANESH PERA' in the 'N. Virginia' region. The main content area is titled 'Amazon Web Services' and lists various services categorized into groups:

- Compute**: EC2 (Virtual Servers in the Cloud), EC2 Container Service (Run and Manage Docker Containers), Elastic Beanstalk (Run and Manage Web Apps), Lambda (Run Code without Thinking about Servers).
- Storage & Content Delivery**: S3 (Scalable Storage in the Cloud), CloudFront (Global Content Delivery Network), Elastic File System (Fully Managed File System for EC2), Glacier (Archive Storage in the Cloud), Snowball (Large Scale Data Transport), Storage Gateway (Hybrid Storage Integration).
- Database**: RDS (Managed Relational Database Service), DynamoDB (Managed NoSQL Database), ElastiCache (In-Memory Cache).
- Developer Tools**: CodeCommit (Store Code in Private Git Repositories), CodeDeploy (Automate Code Deployments), CodePipeline (Release Software using Continuous Delivery).
- Management Tools**: CloudWatch (Monitor Resources and Applications), CloudFormation (Create and Manage Resources with Templates), CloudTrail (Track User Activity and API Usage), Config (Track Resource Inventory and Changes), OpsWorks (Automate Operations with Chef), Service Catalog (Create and Use Standardized Products), Trusted Advisor (Optimize Performance and Security).
- Security & Identity**: Identity & Access Management (Manage User Access and Encryption Keys), Directory Service (Host and Manage Active Directory), Inspector (Analyze Application Security).
- Internet of Things**: AWS IoT (Connect Devices to the Cloud).
- Game Development**: GameLift (Deploy and Scale Session-based Multiplayer Games).
- Mobile Services**: Mobile Hub (Build, Test, and Monitor Mobile Apps), Cognito (User Identity and App Data Synchronization), Device Farm (Test Android, iOS, and Web Apps on Real Devices in the Cloud), Mobile Analytics (Collect, View and Export App Analytics), SNS (Push Notification Service).
- Application Services**: API Gateway (Build, Deploy and Manage APIs), AppStream (Low Latency Application Streaming), CloudSearch (Managed Search Service), Elastic Transcoder (Easy-to-Use Scalable Media Transcoding), SES (Email Sending and Receiving Service).

On the right side, there is a 'Resource Groups' section with a 'Learn more' link. Below it, a 'Create a Group' button and a 'Tag Editor' button are visible. Further down, there are links for 'Getting Started', 'AWS Console Mobile App', 'AWS Marketplace', 'AWS re:Invent Announcements', and 'Service Health'.

## 7. Click on Launch instance which takes you to Step 1: Choose an Amazon Machine Image (AMI). Now click on Amazon Linux Image (the very first block)

The screenshot shows the AWS Management Console interface for the 'Launch Instance Wizard'. The browser tabs include 'EC2 Management Console', 'IAM Management Console', 'AWS Identity and Access', 'What Is IAM? - AWS Ident', and 'aws launch instance - Go'. The URL is 'https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#LaunchInstanceWizard:'. The console header shows 'AWS Services Edit' and user information 'SRIGANESH PERA N. Virginia Support'.

The wizard progress bar shows seven steps: 1. Choose AMI (selected), 2. Choose Instance Type, 3. Configure Instance, 4. Add Storage, 5. Tag Instance, 6. Configure Security Group, and 7. Review.

### Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

**Quick Start** | 1 to 25 of 25 AMIs

Category	AMI Name	AMI ID	Architecture	Buttons
My AMIs	Amazon Linux AMI 2016.03.3 (HVM), SSD Volume Type	ami-6869aa05	64-bit	Select
AWS Marketplace	Amazon Linux	The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.	64-bit	Free tier eligible
Community AMIs	Red Hat Enterprise Linux 7.2 (HVM), SSD Volume Type	ami-2051294a	64-bit	Select
	Red Hat	Red Hat Enterprise Linux version 7.2 (HVM), EBS General Purpose (SSD) Volume Type	64-bit	Free tier eligible
	SUSE Linux Enterprise Server 12 SP1 (HVM), SSD Volume Type	ami-b7b4fedd	64-bit	Select
	SUSE Linux	SUSE Linux Enterprise Server 12 Service Pack 1 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.	64-bit	Free tier eligible

Feedback English | © 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

## 8. It takes you to Choose instance type. Now click on Configure instance

The screenshot shows the AWS Management Console interface for the 'Launch Instance Wizard'. The browser address bar shows the URL: `https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#LaunchInstanceWizard:`. The console header includes the AWS logo, navigation menus for 'AWS', 'Services', and 'Edit', and user information for 'SRIGANESH PERA' in the 'N. Virginia' region. Below the header, a progress bar indicates the current step is '2. Choose Instance Type', with other steps being '1. Choose AMI', '3. Configure Instance', '4. Add Storage', '5. Tag Instance', '6. Configure Security Group', and '7. Review'.

### Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

Filter by: All instance types Current generation [Show/Hide Columns](#)

Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
<input type="checkbox"/>	General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate
<input checked="" type="checkbox"/>	General purpose	t2.micro <small>Free tier eligible</small>	1	1	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.large	2	8	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	m4.large	2	8	EBS only	Yes	Moderate

Buttons: [Cancel](#) [Previous](#) [Review and Launch](#) [Next: Configure Instance Details](#)

Footer: [Feedback](#) [English](#) © 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. [Privacy Policy](#) [Terms of Use](#)



## 9. Select IAM role as ec2-admin that we have created in the first step:

EC2 Management Console X IAM Management Console X AWS Identity and Access X What is IAM? - AWS Ident X aws launch instance - Go X

https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#LaunchInstanceWizard:

AWS Services Edit SRIGANESH PERA N. Virginia Support

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

### Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances *i* 1 Launch into Auto Scaling Group *i*

Purchasing option *i* ☐ Request Spot instances

Network *i* vpc-737a8117 (172.31.0.0/16) (default) *C* Create new VPC

Subnet *i* No preference (default subnet in any Availability Zone) *C* Create new subnet

Auto-assign Public IP *i* Use subnet setting (Enable)

IAM role *i*

ec2-admin *C* Create new IAM role

None

admin

ec2-admin

EMR\_EC2\_DefaultRole

Shutdown behavior *i*

Enable termination protection *i*

Monitoring *i* ☐ Enable CloudWatch detailed monitoring  
Additional charges apply.

Tenancy *i* Shared - Run a shared hardware instance

Cancel Previous **Review and Launch** Next: Add Storage

Feedback English

© 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

10. Click on add storage, tag instance, and configure security group. Make sure SSH is selected (by default you'll have it so don't change anything) and click in 'Review and Launch' and 'Launch'

The screenshot shows the AWS Management Console interface for the 'Launch Instance Wizard'. The browser address bar shows the URL: <https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#LaunchInstanceWizard>. The console header includes the AWS logo, 'Services', and 'Edit' buttons. The navigation bar shows the current step: '6. Configure Security Group'. The main content area is titled 'Step 6: Configure Security Group' and includes a description of security groups. Below this, there are two radio buttons for 'Assign a security group': 'Create a new security group' (selected) and 'Select an existing security group'. The 'Security group name' field is filled with 'launch-wizard-4' and the 'Description' field is filled with 'launch-wizard-4 created 2016-09-08T06:57:51.053+05:30'. A table with columns 'Type', 'Protocol', 'Port Range', and 'Source' contains one rule: 'SSH' (Type), 'TCP' (Protocol), '22' (Port Range), and 'Anywhere' (Source) with a '0.0.0.0/0' IP range. An 'Add Rule' button is below the table. A yellow warning box states: 'Warning: Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.' At the bottom right, there are 'Cancel', 'Previous', and 'Review and Launch' buttons. The footer includes 'Feedback', 'English', and copyright information.

EC2 Management Console x IAM Management Console x AWS Identity and Access x What Is IAM? - AWS Ident x aws launch instance - Go x

https://console.aws.amazon.com/ec2/v2/home?region=us-east-1#LaunchInstanceWizard

AWS Services Edit

SRIGANESH PERA N. Virginia Support

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

### Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group ☐ Select an existing security group

Security group name:

Description:

Type ⓘ	Protocol ⓘ	Port Range ⓘ	Source ⓘ
SSH	TCP	22	Anywhere 0.0.0.0/0

Add Rule

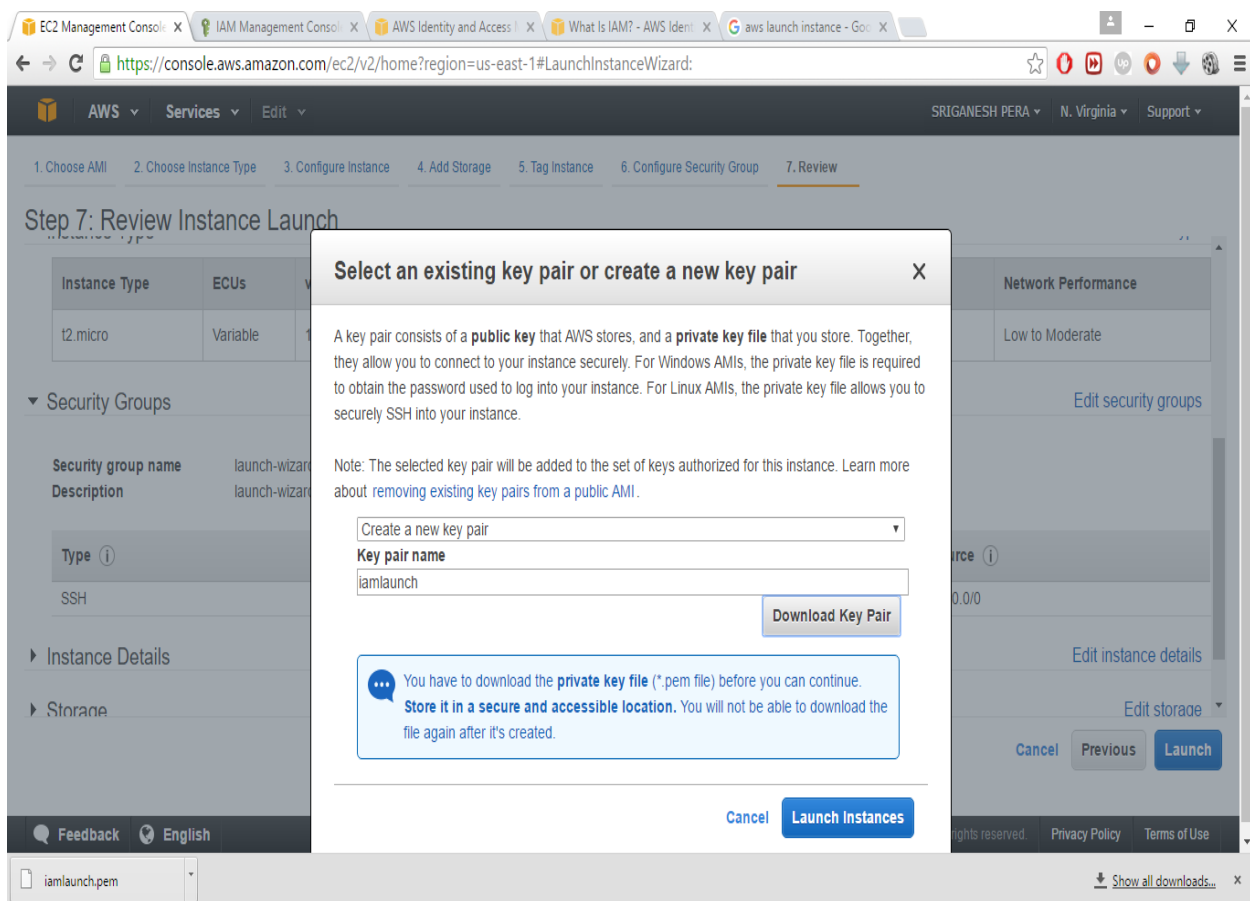
**Warning**

Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

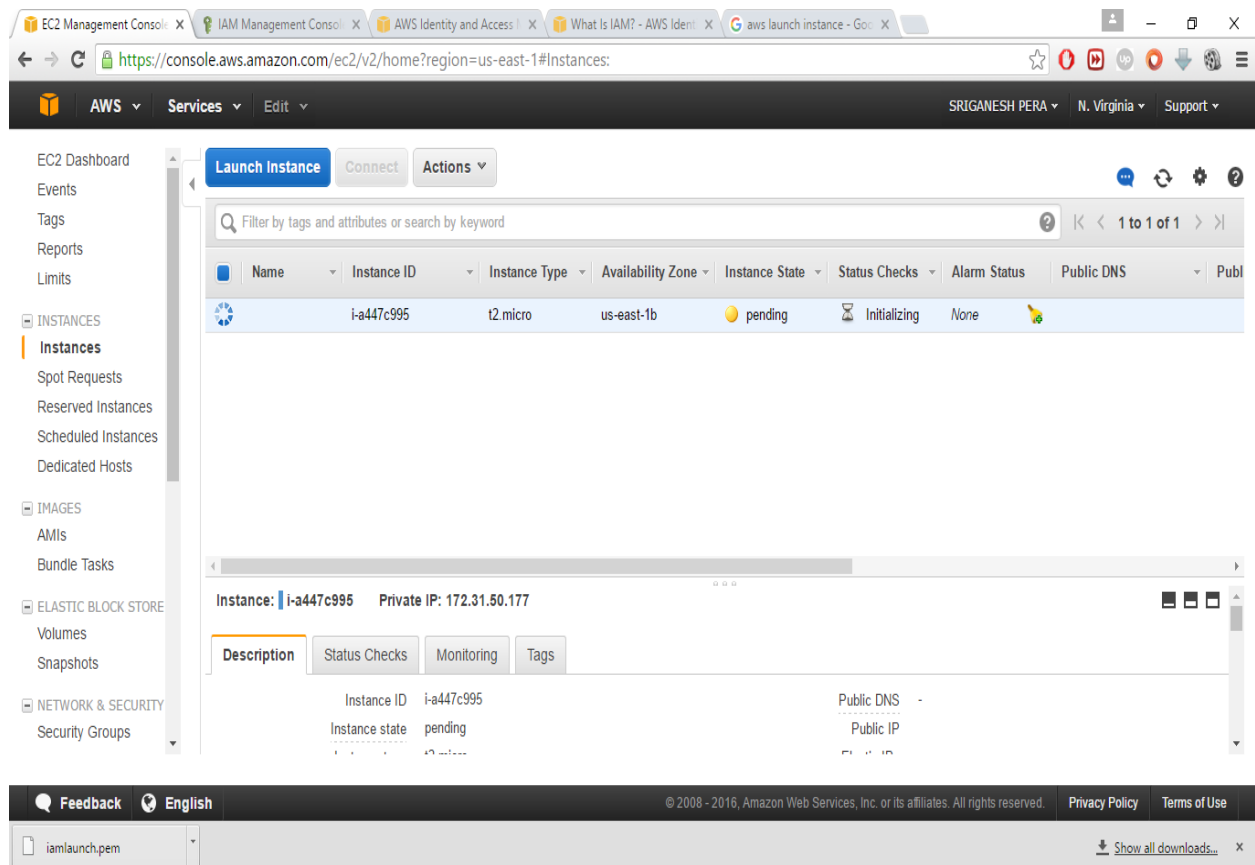
Cancel Previous **Review and Launch**

Feedback English © 2008 - 2016, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

**11. You'll be asked to select a private key air as you need to SSH into AMI instance that we created in 7<sup>th</sup> step. Create a private key and name it as you like and the key with the extension '.pem' will be downloaded. Save it in a folder. Then click on Launch instances and then click on View**



## 12. As we know it takes some time for the instance to be launched



The screenshot displays the AWS Management Console interface. The top navigation bar includes the AWS logo, 'Services', 'Edit', and user information for 'SRIGANESH PERA' in 'N. Virginia'. The left sidebar lists various services, with 'INSTANCES' expanded to show 'Instances', 'Spot Requests', 'Reserved Instances', 'Scheduled Instances', and 'Dedicated Hosts'. The main content area shows a table of EC2 instances. One instance, 'i-a447c995', is listed with a 't2.micro' instance type, 'us-east-1b' availability zone, and a 'pending' state. Below the table, the 'Description' tab for instance 'i-a447c995' is selected, showing details such as 'Instance ID: i-a447c995', 'Instance state: pending', and 'Private IP: 172.31.50.177'. The bottom of the console shows a footer with 'Feedback', 'English', copyright information, and links to 'Privacy Policy' and 'Terms of Use'. A file named 'iamlaunch.pem' is visible in the bottom left corner.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS
	i-a447c995	t2.micro	us-east-1b	pending	Initializing	None	

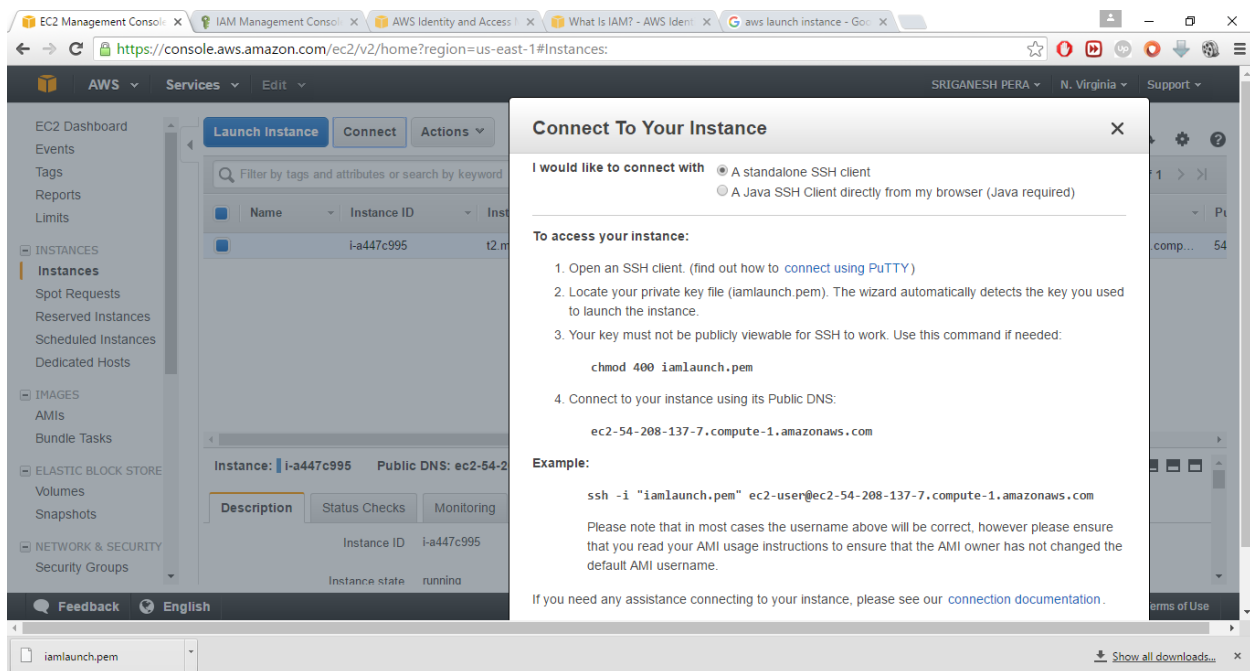
Instance: i-a447c995 Private IP: 172.31.50.177

**Description** Status Checks Monitoring Tags

Instance ID: i-a447c995  
Instance state: pending  
Private IP: 172.31.50.177  
Public DNS: -  
Public IP: -

**Now SSH the instance using Command Line**

**13. Now SSH that instance using your Command line.**  
**First click on connect button, you will notice a template pops out which gives you instruction on how to connect**



**14. Now open your Command Line terminal (either windows or OSX or Linux)**

**a. First cd to the folder where you've stored your .pem file. I've stored mine in 'Downloads' folder.**

**b. Then follow the third step in the popped out window which shows chmod 400 'your key filename'.**

This needs to be done as you're securely accessing AWS using the private key(as this is not for public access).

c.Then follow the 4<sup>th</sup> step example as it is given in your AWS popped out screen.

Now you're connect to your instance securely

```
C:\Users\spbg7\Downloads>chmod 400 iamlaunch.pem
```

```
C:\Users\spbg7\Downloads>ssh -i "iamlaunch.pem" ec2-user@ec2-54-208-137-7.compute-1.amazonaws.com
The authenticity of host 'ec2-54-208-137-7.compute-1.amazonaws.com (54.208.137.7)' can't be established.
ECDSA key fingerprint is SHA256:VlmKUs9x36n3AWFX0wBHcC+k8sY8lY8ufobDYjnidjM.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'ec2-54-208-137-7.compute-1.amazonaws.com,54.208.137.7' (ECDSA) to the list of known hosts.
```



```
https://aws.amazon.com/amazon-linux-ami/2016.03-release-notes/
10 package(s) needed for security, out of 22 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-50-177 ~]$
```

## 15. Now type in Command line:

1. Sudo yum update. After the process is complete

2.Type Sudo yum python-pip. After the process is complete.

3.Type sudo pip install boto3.

### *What is Boto3?*

**Boto** is the Amazon Web Services (AWS) SDK for Python, which allows Python developers to write software that makes use of Amazon services like S3 and EC2. **Boto** provides an easy to use, object-oriented API as well as low-level direct service access.

```
Command Prompt - ssh -i "iamlaunch.pem" ec2-user@ec2-54-208-137-7.compute-1.amazonaws.com
Installing : python26-setuptools-12.2-1.30.amzn1.noarch 5/6
Installing : python26-pip-6.1.1-1.21.amzn1.noarch 6/6
Verifying : python26-setuptools-12.2-1.30.amzn1.noarch 1/6
Verifying : python26-pip-6.1.1-1.21.amzn1.noarch 2/6
Verifying : python26-libs-2.6.9-2.88.amzn1.x86_64 3/6
Verifying : python26-2.6.9-2.88.amzn1.x86_64 4/6
Verifying : python26-backports-ssl_match_hostname-3.4.0.2-1.12.amzn1.noarch 5/6
Verifying : python26-backports-1.0-3.14.amzn1.x86_64 6/6

Installed:
python26-pip.noarch 0:6.1.1-1.21.amzn1

Dependency Installed:
python26.x86_64 0:2.6.9-2.88.amzn1 python26-backports.x86_64 0:1.0-3.14.amzn1 python26-backports-ssl_match_hostname.noarch 0:3.4.0.2-1.12.amzn1
python26-libs.x86_64 0:2.6.9-2.88.amzn1 python26-setuptools.noarch 0:12.2-1.30.amzn1

Complete!
[ec2-user@ip-172-31-50-177 ~]$ sudo pip install boto3
You are using pip version 6.1.1, however version 8.1.2 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
Collecting boto3
  Downloading boto3-1.4.0-py2.py3-none-any.whl (117kB)
    100% |#####| 118kB 2.6MB/s
Requirement already satisfied (use --upgrade to upgrade): jmespath<1.0.0,>=0.7.1 in /usr/lib/python2.7/dist-packages (from boto3)
Requirement already satisfied (use --upgrade to upgrade): botocore<1.5.0,>=1.4.1 in /usr/lib/python2.7/dist-packages (from boto3)
Collecting s3transfer<0.2.0,>=0.1.0 (from boto3)
  Downloading s3transfer-0.1.3-py2.py3-none-any.whl (50kB)
    100% |#####| 53kB 6.9MB/s
Requirement already satisfied (use --upgrade to upgrade): python-dateutil<3.0.0,>=2.1 in /usr/lib/python2.7/dist-packages (from botocore<1.5.0,>=1.4.1->boto3)
Requirement already satisfied (use --upgrade to upgrade): docutils>=0.10 in /usr/lib/python2.7/dist-packages (from botocore<1.5.0,>=1.4.1->boto3)
Requirement already satisfied (use --upgrade to upgrade): futures<4.0.0,>=2.2.0 in /usr/lib/python2.7/dist-packages (from s3transfer<0.2.0,>=0.1.0->boto3)
Requirement already satisfied (use --upgrade to upgrade): six in /usr/lib/python2.7/dist-packages (from python-dateutil<3.0.0,>=2.1->botocore<1.5.0,>=1.4.1->boto3)
Installing collected packages: s3transfer, boto3
Successfully installed boto3-1.4.0 s3transfer-0.1.3
[ec2-user@ip-172-31-50-177 ~]$
```

**16. Now let's write a simple python script about we can securely access S3 buckets .**

**a. Type vi dev.py (Name of your python script)**

**b. Type 'I' so that you can insert some code into it.**

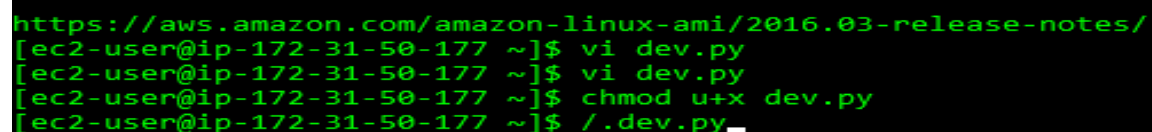
c. Write the following code:

**\*\*make sure that the bucket name you create must be uniquely named.**

```
#!/usr/bin/python
#imported boto3
import boto3
#create connection to S3 using
    boto S3=boto.resource('s3')
# Create a S3 bucket using boto3
    S3.create_bucket (Bucket='mydabba1')
```

d. Then click 'esc' button and type wq!, which save the code.

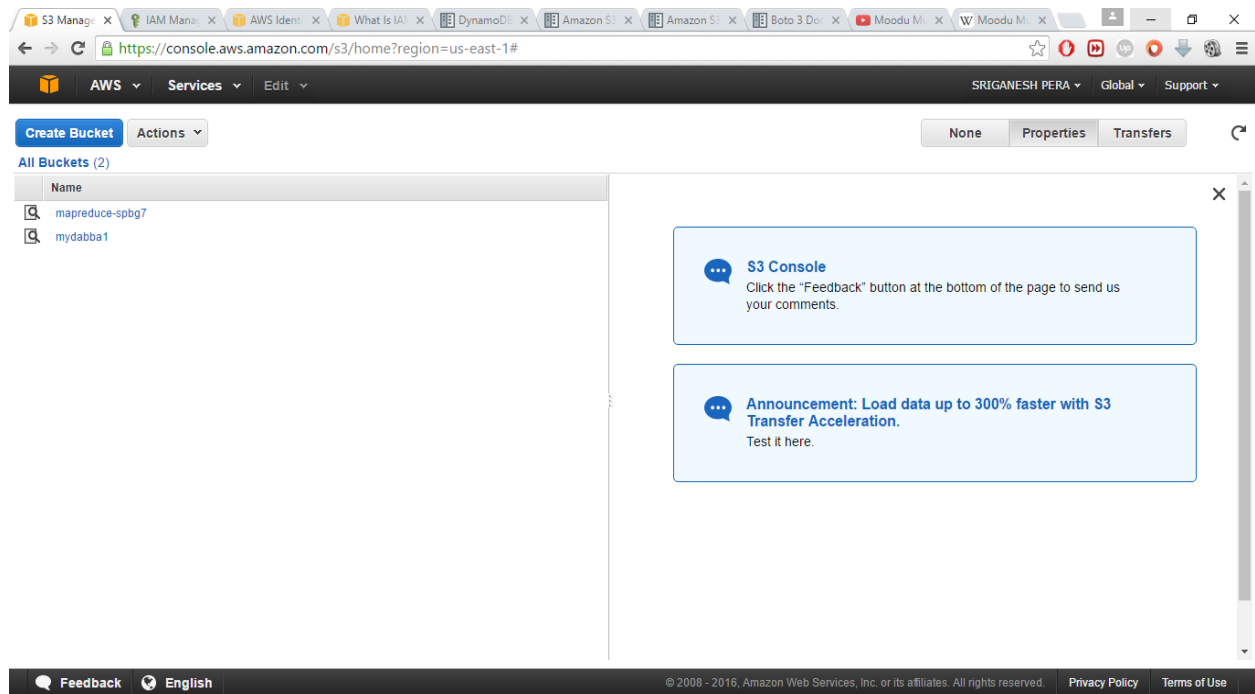
17. Now set the executable permission for our script dev.py by typing `chmod u+x dev.py` and run the script by typing `./dev.py`

A terminal window with a black background and green text. It shows a URL at the top, followed by four command-line prompts and their corresponding commands: 'vi dev.py', 'vi dev.py', 'chmod u+x dev.py', and './dev.py'. The last command is followed by a green underscore character.

```
https://aws.amazon.com/amazon-linux-ami/2016.03-release-notes/
[ec2-user@ip-172-31-50-177 ~]$ vi dev.py
[ec2-user@ip-172-31-50-177 ~]$ vi dev.py
[ec2-user@ip-172-31-50-177 ~]$ chmod u+x dev.py
[ec2-user@ip-172-31-50-177 ~]$ ./dev.py _
```



**18. Now got to AWS console and click on S3 and you'll find the bucket created with the name 'mydabba1'**



**This is how you can access S3 buckets using Python code running on EC2 machine with IAM roles.**

# Lab 2: Configuring the Boto3 SDK with API Credentials

This time AWS access is given to users using API credentials

## 1. Go to AWS IAM

The screenshot displays the AWS Management Console interface. The top navigation bar includes the AWS logo, a dropdown menu for 'Services', and the user's name 'SRIGANESH PERA' along with the region 'N. Virginia'. The main content area is titled 'Amazon Web Services' and is divided into several columns of service categories. The 'Security & Identity' column is highlighted, and the 'Identity & Access Management' link is prominently displayed in orange. The right sidebar provides additional resources and service health information.

**Amazon Web Services**

**Compute**

- EC2: Virtual Servers in the Cloud
- EC2 Container Service: Run and Manage Docker Containers
- Elastic Beanstalk: Run and Manage Web Apps
- Lambda: Run Code without Thinking about Servers

**Storage & Content Delivery**

- S3: Scalable Storage in the Cloud
- CloudFront: Global Content Delivery Network
- Elastic File System: Fully Managed File System for EC2
- Glacier: Archive Storage in the Cloud
- Snowball: Large Scale Data Transport
- Storage Gateway: Hybrid Storage Integration

**Database**

- RDS: Managed Relational Database Service
- DynamoDB: Managed NoSQL Database
- ElastiCache: In-Memory Cache

**Developer Tools**

- CodeCommit: Store Code in Private Git Repositories
- CodeDeploy: Automate Code Deployments
- CodePipeline: Release Software using Continuous Delivery

**Management Tools**

- CloudWatch: Monitor Resources and Applications
- CloudFormation: Create and Manage Resources with Templates
- CloudTrail: Track User Activity and API Usage
- Config: Track Resource Inventory and Changes
- OpsWorks: Automate Operations with Chef
- Service Catalog: Create and Use Standardized Products
- Trusted Advisor: Optimize Performance and Security

**Security & Identity**

- Identity & Access Management**: Manage User Access and Encryption Keys
- Directory Service: Host and Manage Active Directory
- Inspector: Analyze Application Security

**Internet of Things**

- AWS IoT: Connect Devices to the Cloud

**Game Development**

- GameLift: Deploy and Scale Session-based Multiplayer Games

**Mobile Services**

- Mobile Hub: Build, Test, and Monitor Mobile Apps
- Cognito: User Identity and App Data Synchronization
- Device Farm: Test Android, iOS, and Web Apps on Real Devices in the Cloud
- Mobile Analytics: Collect, View and Export App Analytics
- SNS: Push Notification Service

**Application Services**

- API Gateway: Build, Deploy and Manage APIs
- AppStream: Low Latency Application Streaming
- CloudSearch: Managed Search Service
- Elastic Transcoder: Easy-to-Use Scalable Media Transcoding
- SES: Email Sending and Receiving Service

**Resource Groups** [Learn more](#)

A resource group is a collection of resources that share one or more tags. Create a group for each project, application, or environment in your account.

[Create a Group](#) [Tag Editor](#)

**Additional Resources**

[Getting Started](#)  
Read our documentation or view our training to learn more about AWS.

[AWS Console Mobile App](#)  
View your resources on the go with our AWS Console mobile app, available from Amazon Appstore, Google Play, or iTunes.

[AWS Marketplace](#)  
Find and buy software, launch with 1-Click and pay by the hour.

[AWS re:Invent Announcements](#)  
Explore the next generation of AWS cloud capabilities. See what's new

**Service Health**

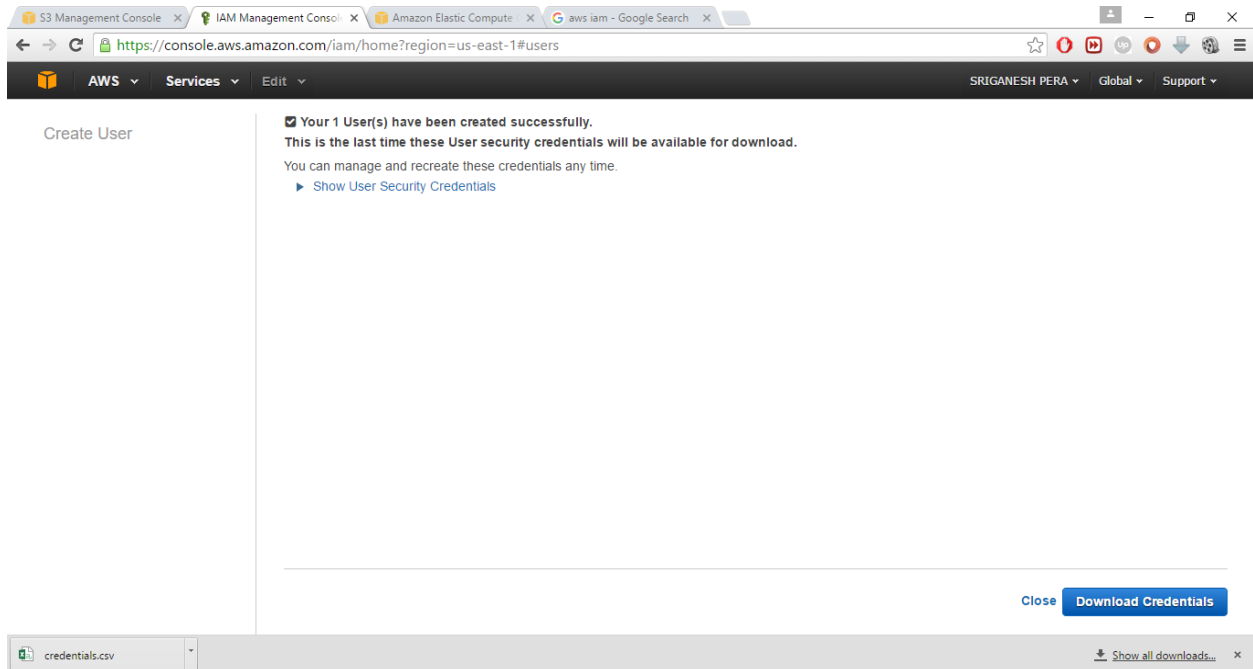
## 2. Click on Users

The screenshot shows the AWS IAM console dashboard. The left sidebar contains a 'Dashboard' section with a 'Search IAM' box and a list of links: Details, Groups, Users (highlighted in orange), Roles, Policies, Identity Providers, Account Settings, Credential Report, and Encryption Keys. The main content area is titled 'Welcome to Identity and Access Management' and displays IAM statistics: 1 User, 0 Groups, 4 Roles, and 0 Identity Providers. Below this is a 'Security Status' section with a progress bar at 1 out of 5 complete, listing tasks like deleting root access keys, activating MFA, and creating individual IAM users. The right sidebar features a 'Feature Spotlight' video and 'Additional Information' links to documentation, playgrounds, and simulators.

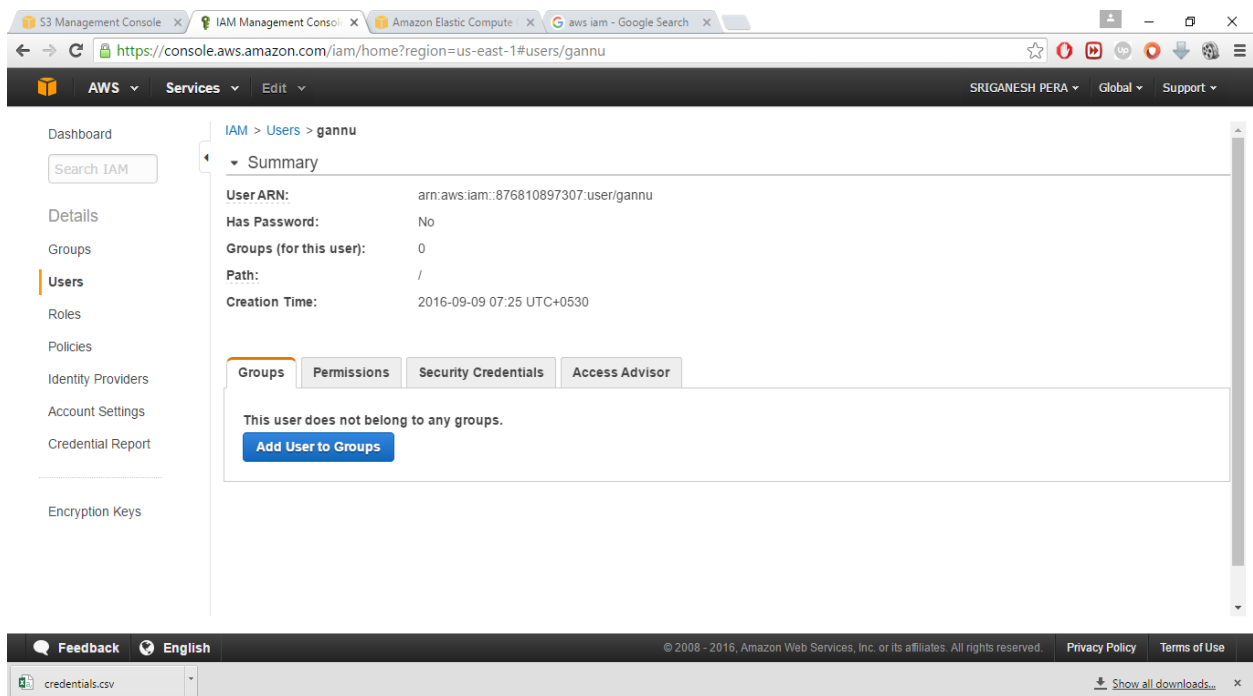
## 3. Click on create new user and type in the name of the user.

The screenshot shows the 'Create User' wizard in the AWS IAM console. The left sidebar has 'Create User' selected. The main content area is titled 'Enter User Names:' and contains five input fields. The first field contains the text 'gannu'. Below the fields is a note: 'Maximum 64 characters each'. A checkbox labeled 'Generate an access key for each user' is checked. Below this checkbox is a note: 'Users need access keys to make secure REST or Query protocol requests to AWS service APIs. For users who need access to the AWS Management Console, create a password in the Users panel after completing this wizard.' At the bottom right, there are 'Cancel' and 'Create' buttons.

## 4. Click on Create Users and then Download Credentials. You'll get a CSV file.



## 5. Now click on close and select the username you just created and click on the username.



## 5. Now click on attach policy and give access to whatever the services you want to give to that user In this case I'm giving admin access.

The screenshot shows the AWS IAM console interface. The browser tabs include 'S3 Management Console', 'IAM Management Console', 'Amazon Elastic Compute', and 'aws iam - Google Search'. The address bar shows the URL 'https://console.aws.amazon.com/iam/home?region=us-east-1#users/gannu'. The navigation bar includes 'AWS', 'Services', 'Edit', and a user profile 'SRIGANESH PERA'. The main content area is titled 'Attach Policy' and includes the instruction: 'Select one or more policies to attach. Each user can have up to 10 policies attached.' Below this is a table of policies with columns: Policy Name, Attached Entities, Creation Time, and Edited Time. The 'AdministratorAccess' policy is selected. At the bottom right, there are 'Cancel' and 'Attach Policy' buttons. A file download notification for 'credentials.csv' is visible at the bottom left.

Attach Policy

Select one or more policies to attach. Each user can have up to 10 policies attached.

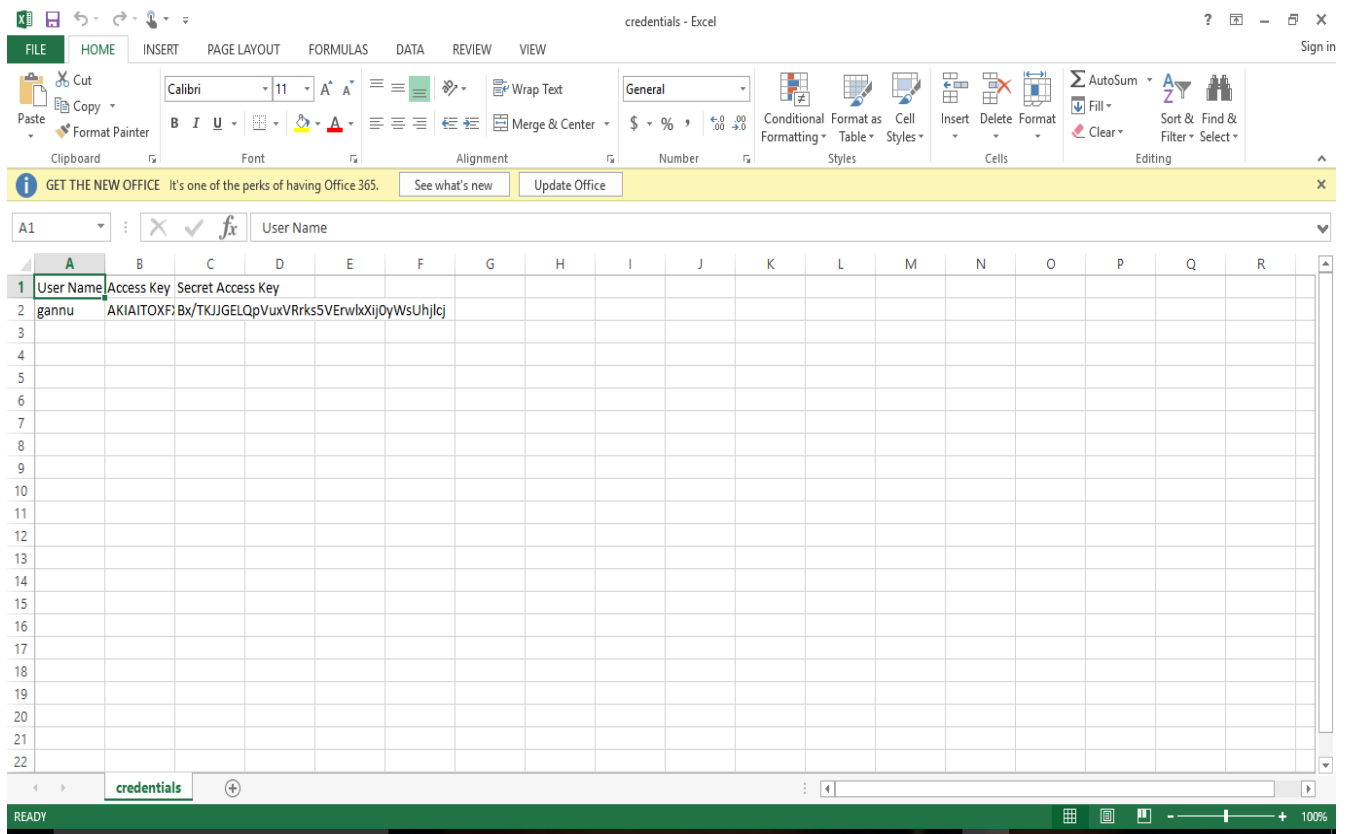
Filter: Policy Type  Showing 204 results

	Policy Name	Attached Entities	Creation Time	Edited Time
<input checked="" type="checkbox"/>	AdministratorAccess	3	2015-02-07 00:09 UTC+0530	2015-02-07 00:09 UTC+0530
<input type="checkbox"/>	AmazonElasticMapReducefor...	1	2015-02-07 00:11 UTC+0530	2015-05-14 02:57 UTC+0530
<input type="checkbox"/>	AmazonElasticMapReduceRole	1	2015-02-07 00:11 UTC+0530	2016-02-11 04:11 UTC+0530
<input type="checkbox"/>	AmazonAPIGatewayAdministr...	0	2015-07-09 23:04 UTC+0530	2015-07-09 23:04 UTC+0530
<input type="checkbox"/>	AmazonAPIGatewayInvokeFul...	0	2015-07-09 23:06 UTC+0530	2015-07-09 23:06 UTC+0530
<input type="checkbox"/>	AmazonAPIGatewayPushToCl...	0	2015-11-12 05:11 UTC+0530	2015-11-12 05:11 UTC+0530
<input type="checkbox"/>	AmazonAppStreamFullAccess	0	2015-02-07 00:10 UTC+0530	2015-02-07 00:10 UTC+0530
<input type="checkbox"/>	AmazonAppStreamReadOnly...	0	2015-02-07 00:10 UTC+0530	2015-02-07 00:10 UTC+0530

Cancel Attach Policy

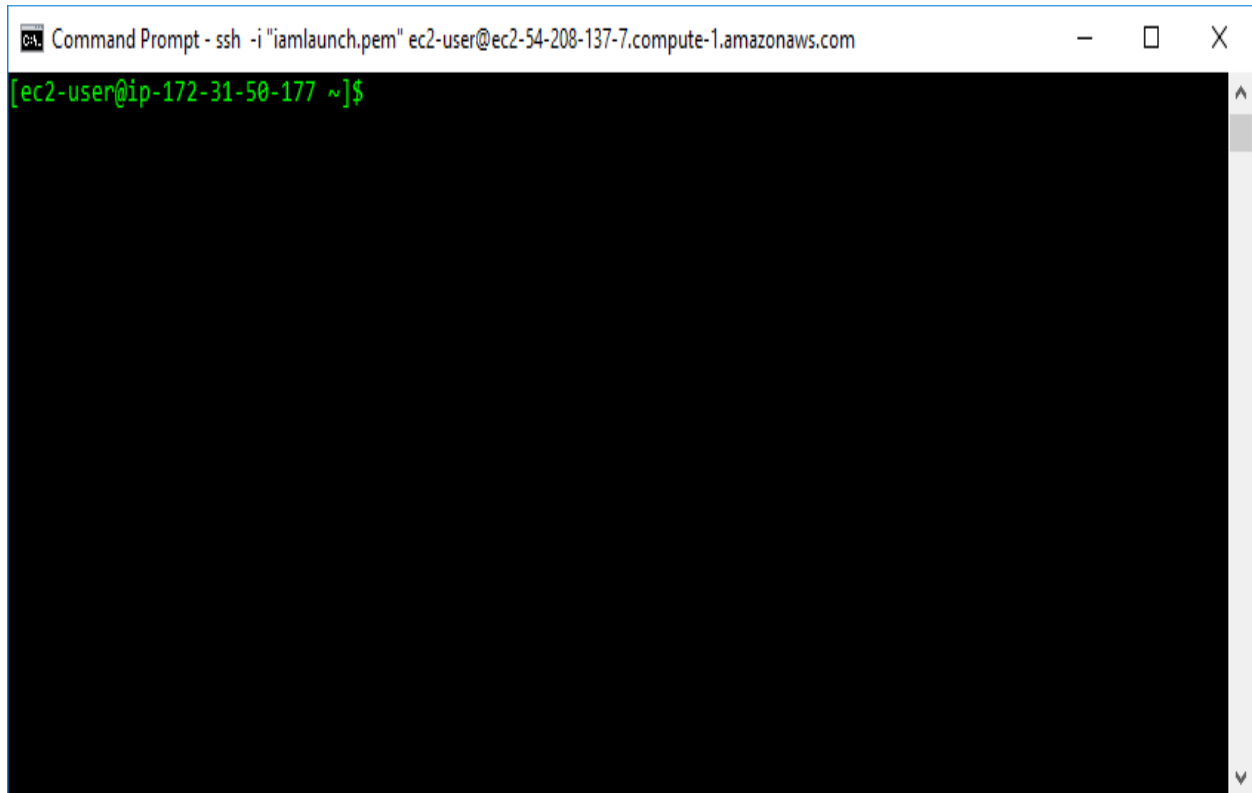
credentials.csv Show all downloads...

6. Now you've attached policy. Then click on the credentials we downloaded earlier to see what it has to offer



As you can see you can find username 'gannu' (which I have created along with the access key and secret access key)

7. Now let's get back to our terminal window and connect it back to the AMI instance we have launched. If you're already connected then don't worry.



The image shows a Windows Command Prompt window with the title bar text: "Command Prompt - ssh -i "iamlaunch.pem" ec2-user@ec2-54-208-137-7.compute-1.amazonaws.com". The window has standard Windows window controls (minimize, maximize, close) on the right. The terminal area has a black background. The prompt is green text: "[ec2-user@ip-172-31-50-177 ~]\$". There is a vertical scrollbar on the right side of the terminal area.

8. Now write the following Python Boto3 code in your favorite editor (for me it's Vim) and run the code the same way we have done it in previous lab.

In you cmd prompt type: vi dev 1.py

Then type i to insert the following code:

```
#!/usr/bin/python
```

```
from boto3.session import Session
```

**\*\* rather than importing boto3 we import a session from boto3\*\***

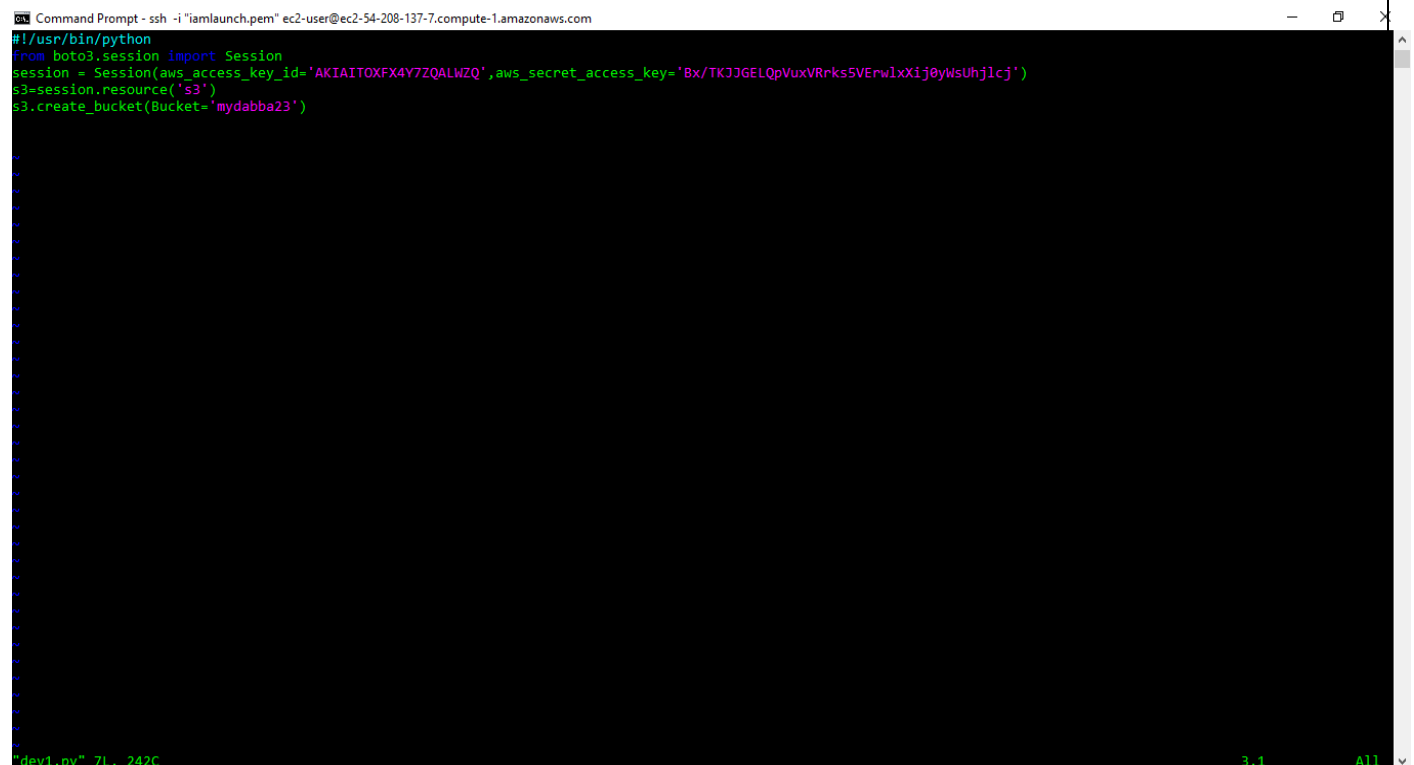
```
session = Session
```

```
(aws_access_key_id='AKIAITOXFX4Y7ZQALWZQ',aws_secret_access_key=' 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')
```

```
s3 = session.resource('s3') #creating a session to S3
```

```
s3.create_bucket(Bucket='mydabba23') #creating a bucket
```

**In the parameters for Session i.e aws\_access\_key\_id and aws\_secret\_Access\_key.You need to add the secret keys you downloaded in the step**



The screenshot shows a terminal window titled "Command Prompt - ssh -i 'iamlaunch.pem' ec2-user@ec2-54-208-137-7.compute-1.amazonaws.com". The terminal displays the following Python code:

```
#!/usr/bin/python
from boto3.session import Session
session = Session(aws_access_key_id='AKIAITOXFX4Y7ZQALWZQ',aws_secret_access_key=' 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')
s3=session.resource('s3')
s3.create_bucket(Bucket='mydabba23')
```

The terminal output shows a series of tilde characters (~) indicating the script is running. At the bottom of the terminal, it shows the file path "dev1.py" 71, 242C and the status "3,1 All".



Now run the code :

**chmod u+x dev1.py**

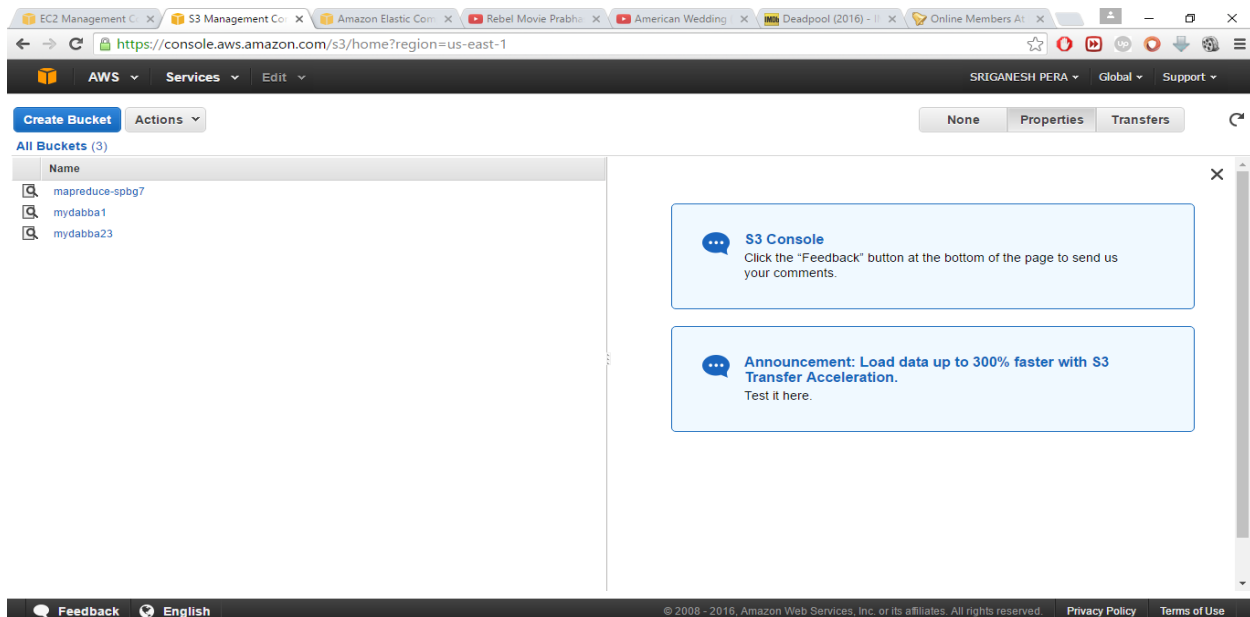
**./dev1.py**

```
Command Prompt - ssh -i "iamlaunch.pem" ec2-user@ec2-54-208-137-7.compute-1.amazonaws.com
Last login: Fri Sep  9 02:43:30 2016 from 103.49.206.19

  _ _ _ _ _
 _ _ _ _ _ ) Amazon Linux AMI
 _ _ _ _ _

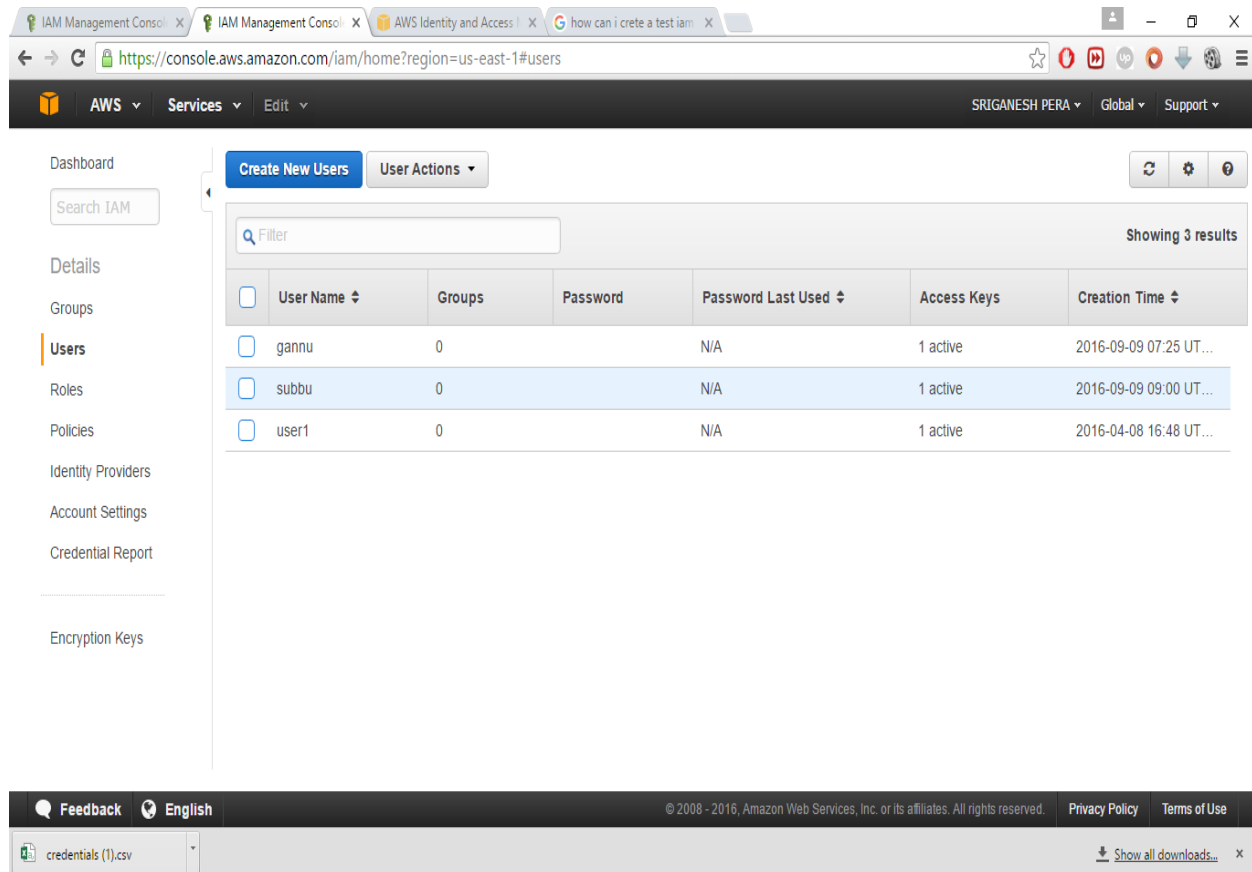
https://aws.amazon.com/amazon-linux-ami/2016.03-release-notes/
No packages needed for security; 1 packages available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-50-177 ~]$ chmod u+x dev1.py
[ec2-user@ip-172-31-50-177 ~]$ ./dev1.py
[ec2-user@ip-172-31-50-177 ~]$
```

Now check your S3 in AWS console.You'll find the new bucket 'mydabba23' .



# LAB 3 : Create policy for users for restricted access to AWS.

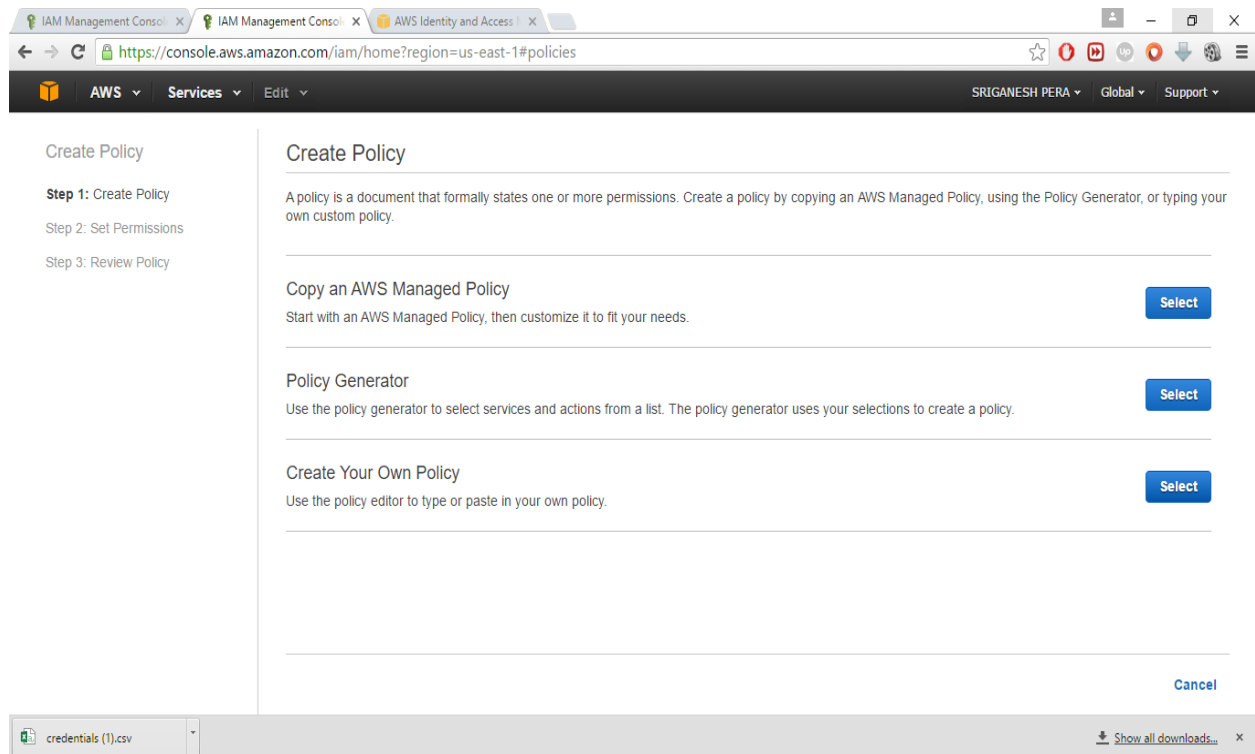
## 1.Create a new user in IAM console:



The screenshot displays the AWS IAM console interface. The top navigation bar includes the AWS logo, a dropdown menu for 'Services', and an 'Edit' button. The user's name 'SRIGANESH PERA' and the region 'Global' are shown. The left sidebar contains a 'Dashboard' section with a 'Search IAM' box, and a 'Details' section with links to 'Groups', 'Users', 'Roles', 'Policies', 'Identity Providers', 'Account Settings', 'Credential Report', and 'Encryption Keys'. The 'Users' link is highlighted. The main content area features a 'Create New Users' button and a 'User Actions' dropdown. Below these is a search bar labeled 'Filter' and a table showing 3 results. The table has columns for 'User Name', 'Groups', 'Password', 'Password Last Used', 'Access Keys', and 'Creation Time'. The users listed are 'gannu', 'subbu', and 'user1'. The 'subbu' user is highlighted. At the bottom, there is a footer with 'Feedback', 'English', copyright information, 'Privacy Policy', and 'Terms of Use'. A download bar at the very bottom shows a file named 'credentials (1).csv' and a 'Show all downloads...' link.

<input type="checkbox"/>	User Name ↕	Groups	Password	Password Last Used ↕	Access Keys	Creation Time ↕
<input type="checkbox"/>	gannu	0		N/A	1 active	2016-09-09 07:25 UT...
<input type="checkbox"/>	subbu	0		N/A	1 active	2016-09-09 09:00 UT...
<input type="checkbox"/>	user1	0		N/A	1 active	2016-04-08 16:48 UT...

## 2. Click on Policies and then click on Create your own policy

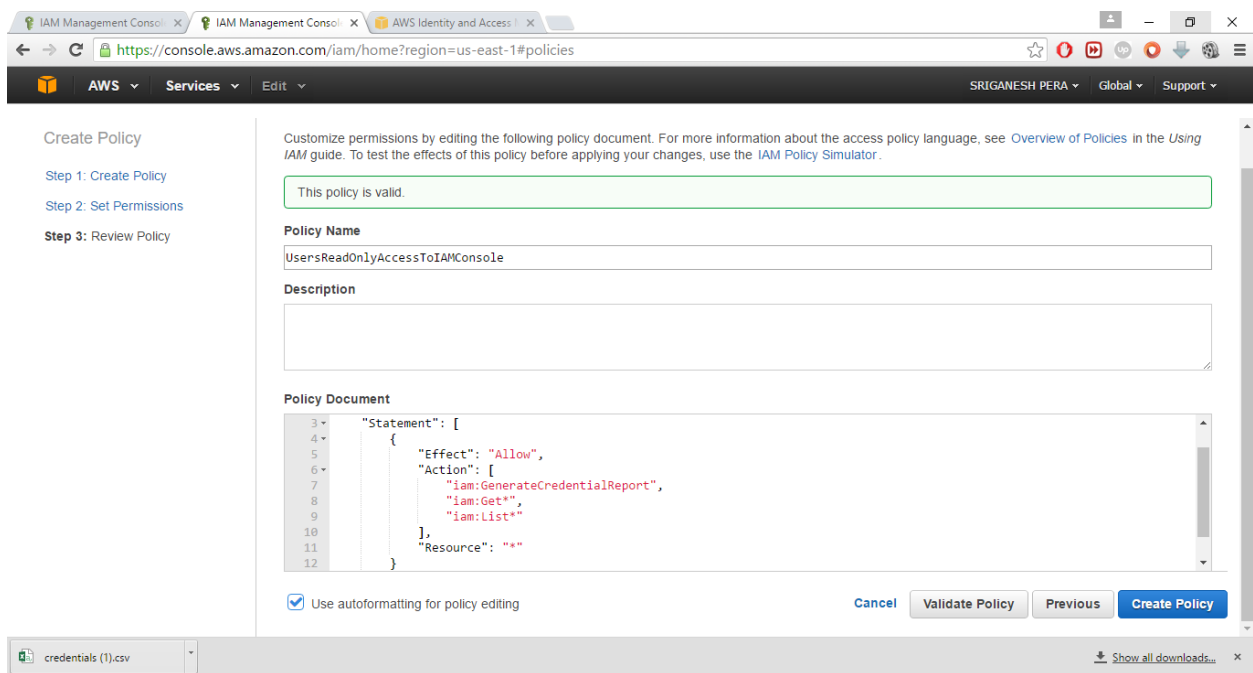


3. In the Policyname type :  
UsersReadOnlyAccessToIAMConsole and in the Policy Document type the following code:

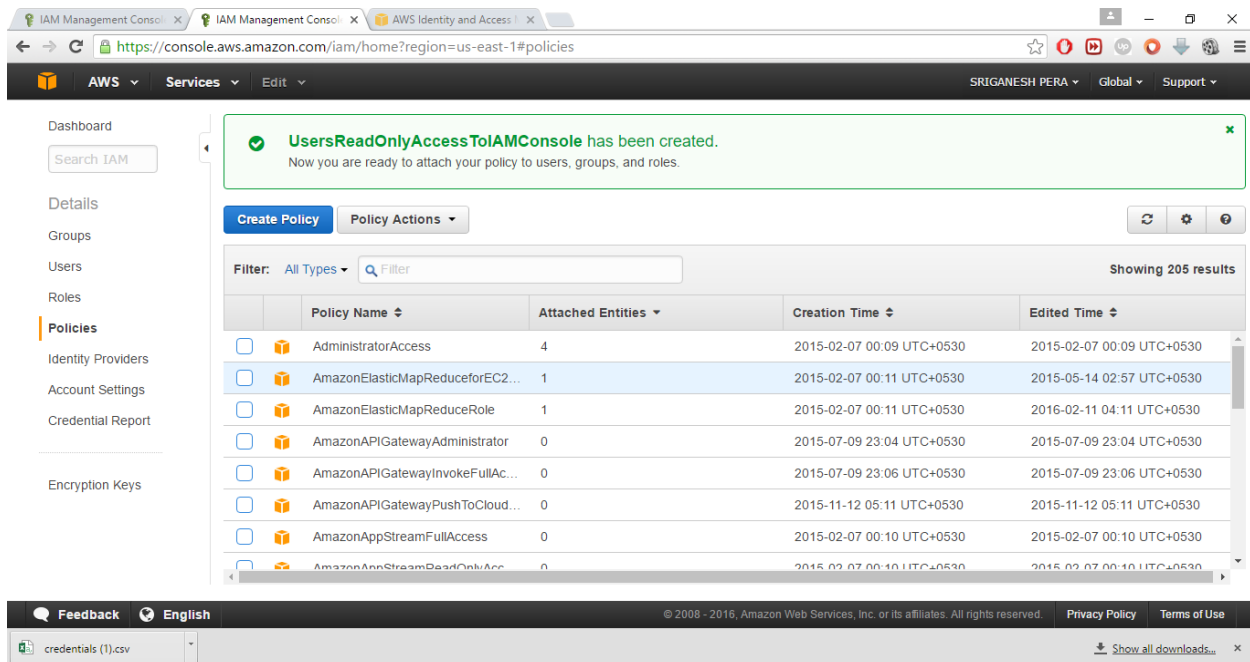
```
{  
  "Version": "2012-10-17",  
  "Statement": [ {  
    "Effect": "Allow",  
    "Action": [  
      "iam:GenerateCredentialReport",
```

```
"iam:Get*",  
"iam:List*"  
,  
"Resource": "*" ]  
}
```

Click on **Validate** now to make sure the code is error free and if you get 'policy is valid' in green color at the top

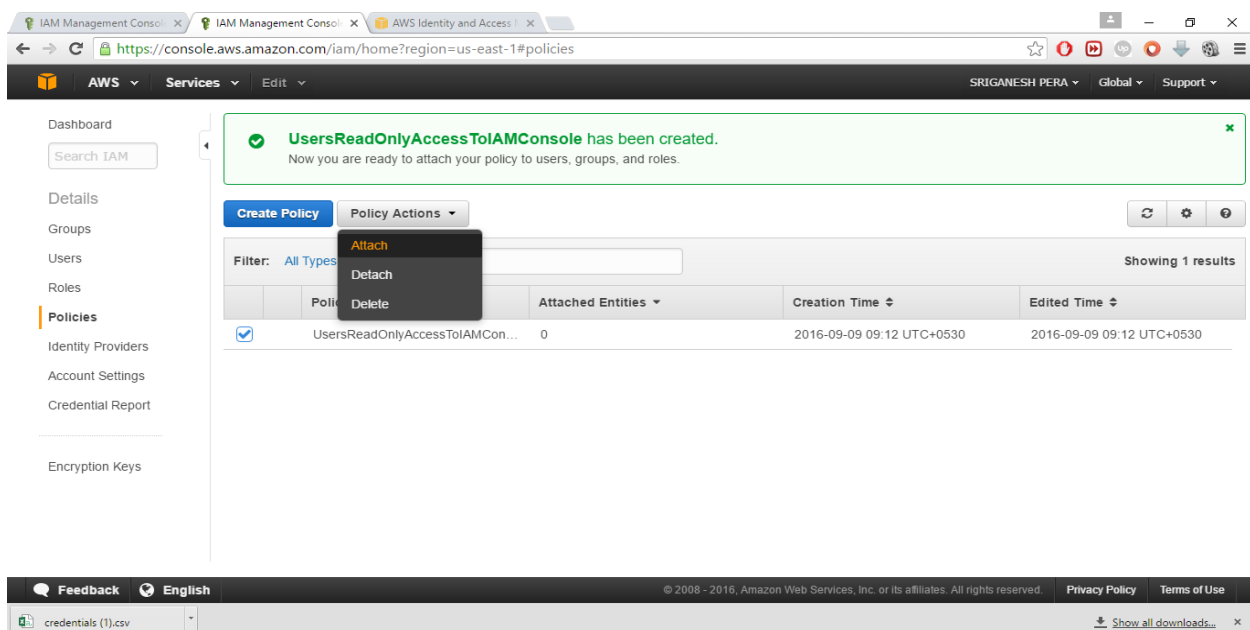


Click on create policy.

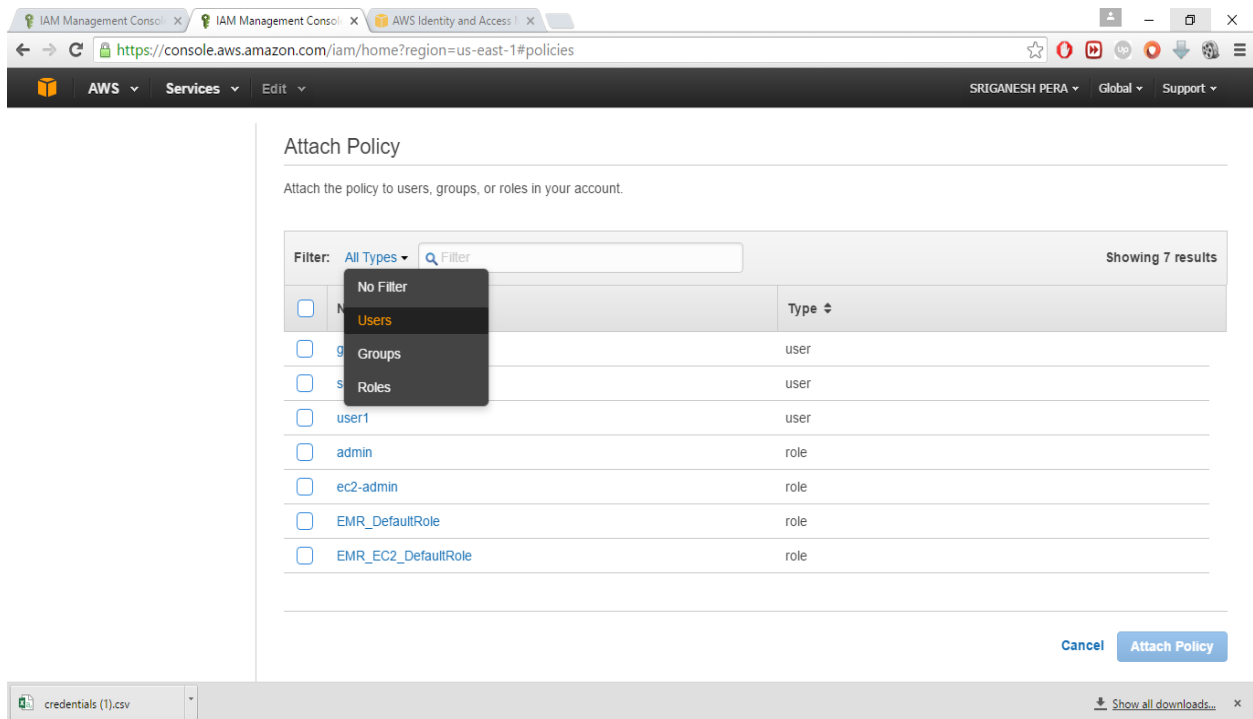


If you can search the policy you just created in search bar. You'll find it.

4. Select your policy and click on policy actions and choose Attach policy form the drop-down list



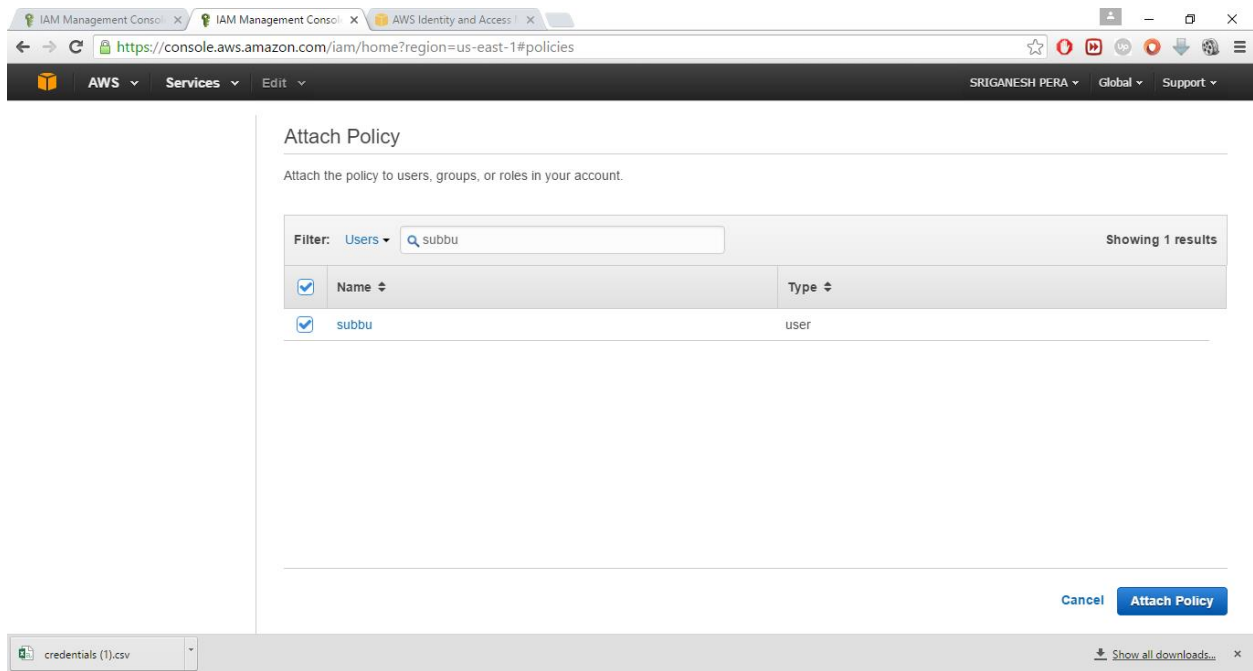
## 5. Now Choose All Types to display the filter menu, and then choose Users.



The screenshot shows the AWS IAM console 'Attach Policy' page. The 'Filter' dropdown is set to 'All Types' and a filter menu is open showing 'Users' selected. The table lists 7 results:

Name	Type
user	user
user	user
user	user
admin	role
ec2-admin	role
EMR_DefaultRole	role
EMR_EC2_DefaultRole	role

## 6. Now select your user and click on attach your policy



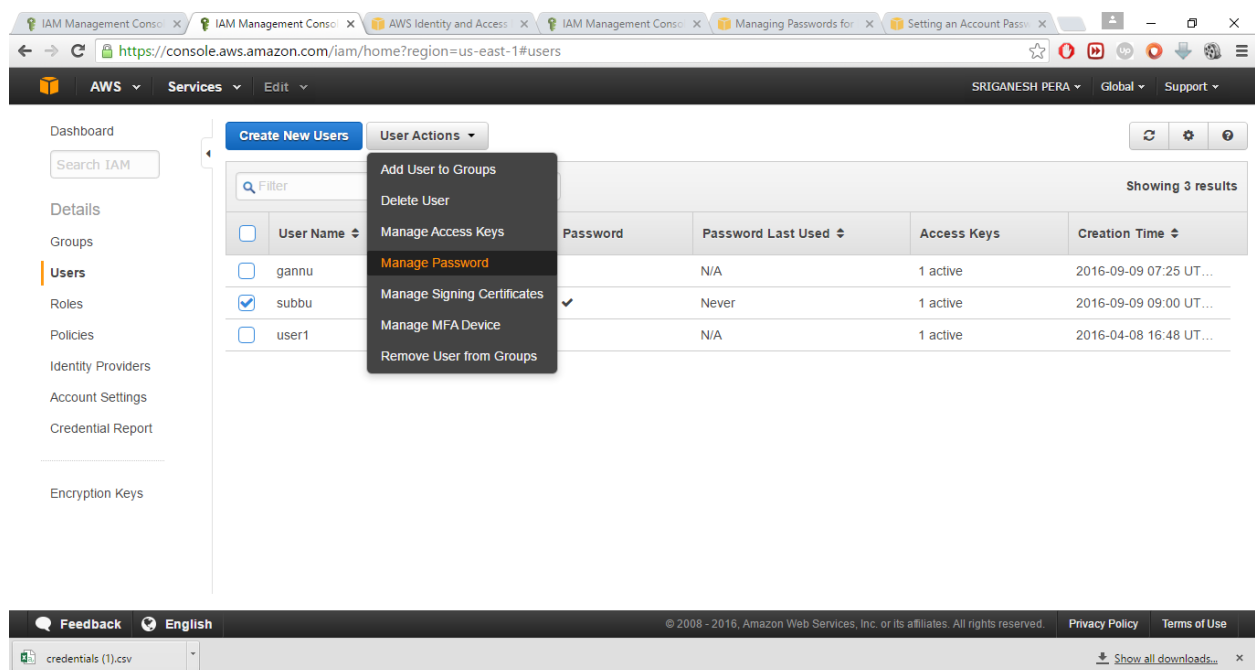
The screenshot shows the AWS IAM console 'Attach Policy' page. The 'Filter' dropdown is set to 'Users' and the search bar contains 'subbu'. The table shows 1 result:

Name	Type
subbu	user

Now You have attached the policy to your IAM test user('subbu' in my case), which means that user now has read-only access to the IAM console.

Now we need to test the user whether he can access as read only user. Before we test it we need to create a password for the user

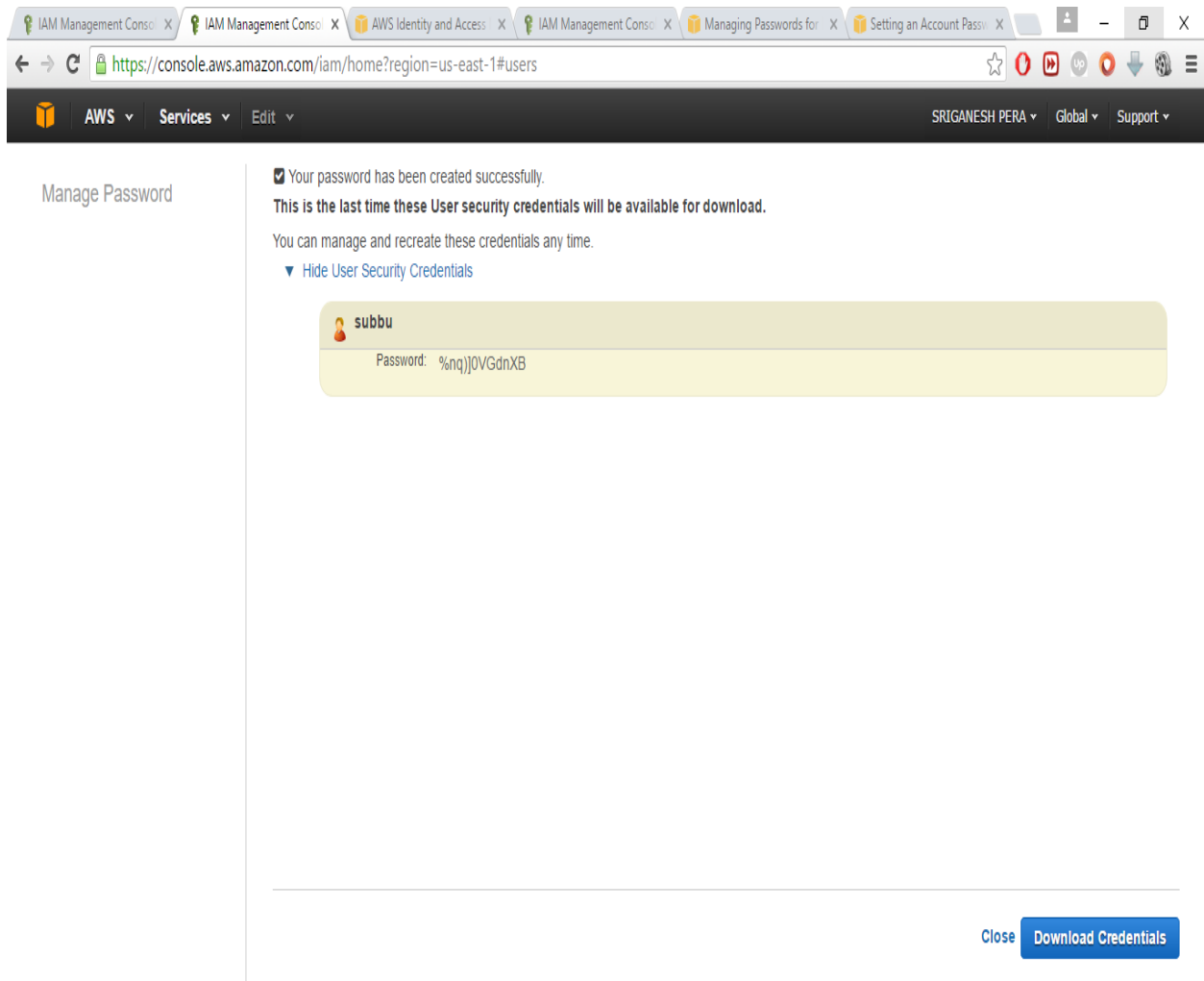
7.In the IAM console Select **users** and check the respective user you would like to create a password and the click on **user actions** and then click on **Manage passwords**



The screenshot displays the AWS IAM console interface. On the left, the navigation menu includes Dashboard, Search IAM, Details, Groups, **Users**, Roles, Policies, Identity Providers, Account Settings, Credential Report, and Encryption Keys. The main content area shows the 'Users' page with a 'Create New Users' button and a 'User Actions' dropdown menu. The dropdown menu is open, showing options: Add User to Groups, Delete User, Manage Access Keys, **Manage Password** (highlighted), Manage Signing Certificates, Manage MFA Device, and Remove User from Groups. Below the dropdown, a table lists three users: gannu, subbu, and user1. The 'subbu' user is selected with a blue checkmark. The table columns are: User Name, Password, Password Last Used, Access Keys, and Creation Time. The 'subbu' user has a checkmark in the 'Password' column, indicating a password is set. The footer of the console shows 'Feedback', 'English', copyright information, and links to 'Privacy Policy' and 'Terms of Use'. A download bar at the bottom shows a file named 'credentials (1).csv'.

User Name	Password	Password Last Used	Access Keys	Creation Time
<input type="checkbox"/> gannu		N/A	1 active	2016-09-09 07:25 UT...
<input checked="" type="checkbox"/> subbu	✓	Never	1 active	2016-09-09 09:00 UT...
<input type="checkbox"/> user1		N/A	1 active	2016-04-08 16:48 UT...

8. Then click on auto generate password and click on apply. Now click on show user security credentials. Copy the password and save it in a file



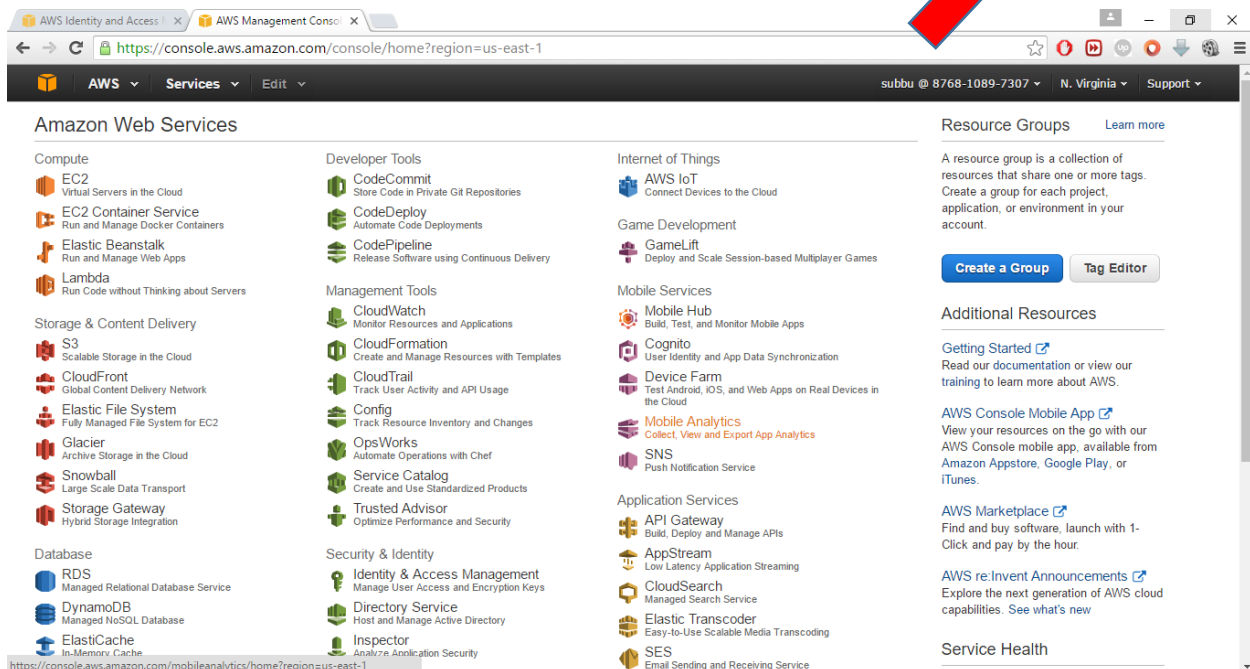
qyjEgZc}ptYB



9. Now it's quite simple. Go back to IAM console Dashboard. On the top of the screen you'll find IAM users sign-in link: Copy that link and go to that link. Now you'll get the login page asking for username and password.

In the username box type the name of the user you created in the IAM console and in the password field type the password you just saved.

If you notice I've logged in as the IAM user 'Subbu' I've created.

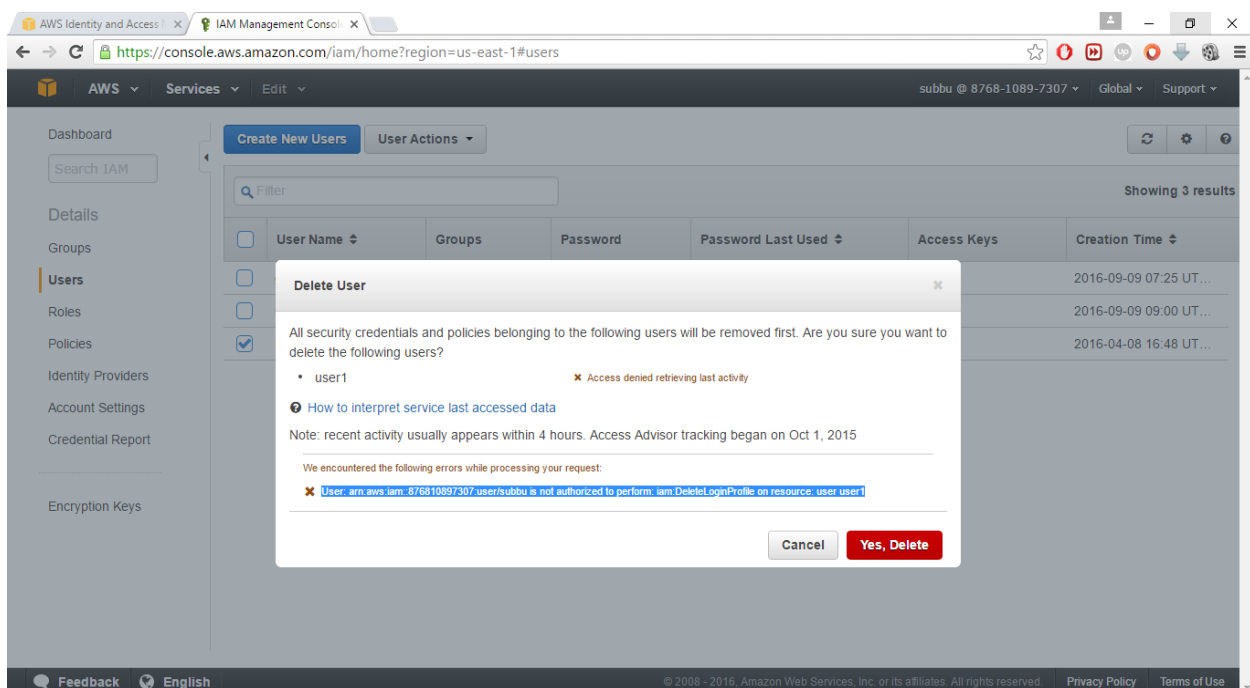


Now lets try to make changes with IAM and see whtehr the user Subbu has the read-only access or not.

I've gone to IAM Console and tried deleting an user there.

# Bingo!

I received an error message that I'm not authorized to do that!



-----End of IAM tutorial-----