

# Creating an EC2 Instance, SSH Access, and S3 Bucket on AWS

Priyanshu Kumar Sharma, 2022-B-17102004A, B.Tech CTIS, SEM-VI/B

March 11, 2025

## 1 Introduction

This report details the step-by-step procedure for launching an EC2 instance on AWS, connecting to it via SSH, and creating an S3 bucket using Terraform. Screenshots are provided for better understanding.

## 2 Creating an EC2 Instance on AWS Console

### 2.1 Step 1: Log in to AWS

Go to AWS Management Console and sign in.

### 2.2 Step 2: Navigate to EC2 Dashboard

From the AWS services, select **EC2**.

### 2.3 Step 3: Launch a New Instance

Click on **Launch Instance**.

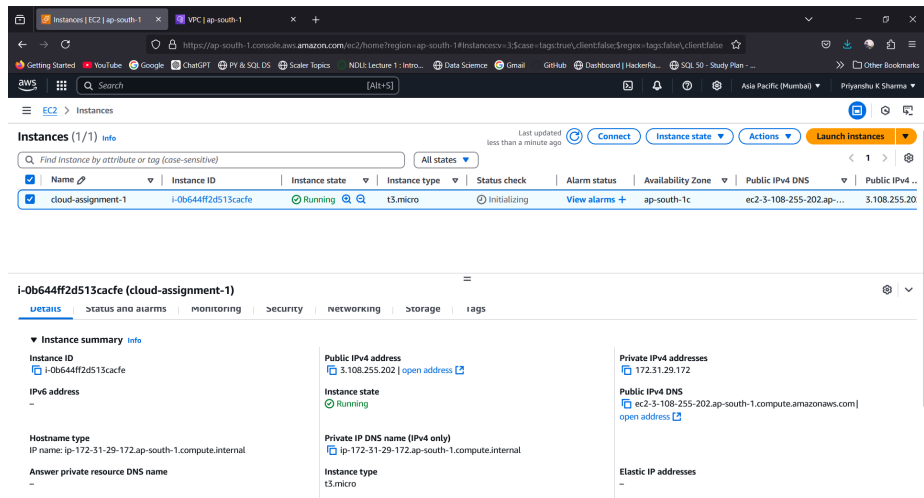


Figure 1: EC2 Dashboard

## 2.4 Step 4: Configure Instance Details

- Choose an AMI (e.g., Ubuntu 22.04 LTS).
- Select instance type (e.g., t2.micro).
- Configure security groups (allow SSH access from your IP).
- Assign a key pair for SSH access.

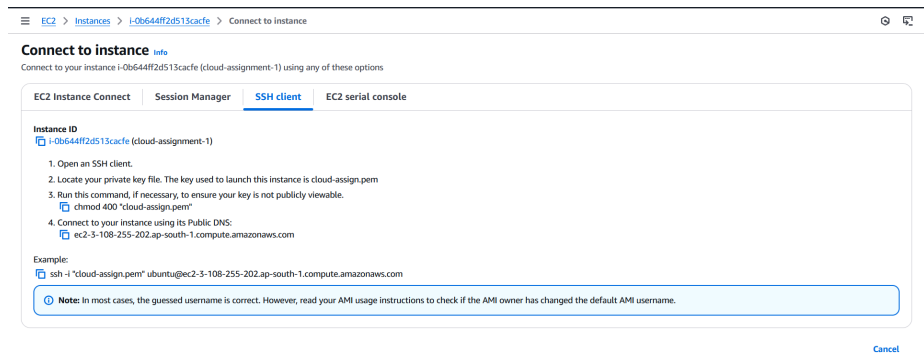


Figure 2: Connecting EC2 Instance with SSH

## 2.5 Step 5: Launch and Verify

Click **Launch** and wait for the instance to initialize.

## 3 Fixing SSH Private Key Permission Issues on Windows

When connecting to an EC2 instance using SSH on Windows, you may encounter an error due to improper file permissions on your private key (.pem file). The necessary fixes and configurations are stored in a GitHub repository:

EC2 Windows SSH Connection Fix

### 3.1 Error Message Example

#### Common Error

```

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@                WARNING: UNPROTECTED PRIVATE KEY FILE!                @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions for 'C:\path\to\your-key.pem' are too open.
It is required that your private key files are NOT
accessible by others.
Load key "C:\path\to\your-key.pem": bad permissions
Permission denied (publickey).
```

```

D:\SEM-6\CT\Keys>chmod 400 "cloud-assign.pem"

D:\SEM-6\CT\Keys>ssh -i "cloud-assign.pem" ubuntu@ec2-3-108-255-202.ap-south-1.compute.amazonaws.com
The authenticity of host 'ec2-3-108-255-202.ap-south-1.compute.amazonaws.com (3.108.255.202)' can't be established.
ED25519 key fingerprint is SHA256:fEc5jY24t9MMYVbxPhYM/Shw2ISPFZKrwBozQB1Agpw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-3-108-255-202.ap-south-1.compute.amazonaws.com' (ED25519) to the list of known hosts.
Bad permissions. Try removing permissions for user: NT AUTHORITY\Authenticated Users (S-1-5-11) on file D:/SEM-6/CT/Keys/cloud-assign.pem.
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@                WARNING: UNPROTECTED PRIVATE KEY FILE!                @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions for 'cloud-assign.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key 'cloud-assign.pem': bad permissions
ubuntu@ec2-3-108-255-202.ap-south-1.compute.amazonaws.com: Permission denied (publickey).
```

Figure 3: Configuring EC2 Instance

### 3.2 Step 1: Open PowerShell as Administrator

1. Press 'Win + X' and select \*\*PowerShell (Admin)\*\*. 2. Navigate to the directory where your .pem file is stored:

#### Navigate to Key Folder

```
cd "C://path//to//your-key-folder"
```

### 3.3 Step 2: Fix Key File Permissions

Run the following commands in PowerShell:

#### Remove Inherited Permissions

```
icacls "your-key.pem" /inheritance:r
```

#### Remove Unauthorized Users

```
icacls "your-key.pem" /remove "Authenticated Users" "Everyone"  
"BUILTIN  
Users"
```

#### Grant Read-Only Access to Current User

```
icacls "your-key.pem" /grant:r "
```

### 3.4 Step 3: Verify Correct Permissions

#### Check Key File Permissions

```
icacls "your-key.pem"
```

Output should look like: your-key.pem YOUR-PC:(R)

```
D:\SEM-6\CT\Keys>chmod 400 "cloud-assign.pem"  
D:\SEM-6\CT\Keys>ssh -i "cloud-assign.pem" ubuntu@ec2-3-108-255-202.ap-south-1.compute.amazonaws.com  
The authenticity of host 'ec2-3-108-255-202.ap-south-1.compute.amazonaws.com (3.108.255.202)' can't be established.  
ED25519 key fingerprint is SHA256:fec5jY24t9MMVbXPhYM/Shw2ISPFZKrv8ozQ81Agpw.  
This key is not known by any other names.  
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes  
Warning: Permanently added 'ec2-3-108-255-202.ap-south-1.compute.amazonaws.com' (ED25519) to the list of known hosts.  
Bad permissions. Try removing permissions for user: NT AUTHORITY\Authenticated Users (S-1-5-11) on file D:\SEM-6\CT\Keys/cloud-assign.pem.  
Warning: UNPROTECTED PRIVATE KEY FILE!  
Permissions for 'cloud-assign.pem' are too open.  
It is required that your private key files are NOT accessible by others.  
This private key will be ignored.  
Load key "cloud-assign.pem": bad permissions  
ubuntu@ec2-3-108-255-202.ap-south-1.compute.amazonaws.com: Permission denied (publickey).  
  
D:\SEM-6\CT\Keys>icacls "cloud-assign.pem" /inheritance:r  
processed file: cloud-assign.pem  
Successfully processed 1 files; Failed processing 0 files  
  
D:\SEM-6\CT\Keys>icacls "cloud-assign.pem" /remove "Authenticated Users" "Everyone" "BUILTIN\Users"  
processed file: cloud-assign.pem  
Successfully processed 1 files; Failed processing 0 files  
  
D:\SEM-6\CT\Keys>icacls "cloud-assign.pem" /grant:r "BUILTIN\Users"  
cloud-assign.pem: The filename, directory name, or volume label syntax is incorrect.  
Successfully processed 0 files; Failed processing 1 files  
  
D:\SEM-6\CT\Keys>icacls "cloud-assign.pem" /grant:r "BUILTIN\Users"  
cloud-assign.pem: The filename, directory name, or volume label syntax is incorrect.  
Successfully processed 0 files; Failed processing 1 files  
  
D:\SEM-6\CT\Keys>icacls "cloud-assign.pem" /grant:r "BUILTIN\Users"  
processed file: cloud-assign.pem  
Successfully processed 1 files; Failed processing 0 files  
  
D:\SEM-6\CT\Keys>icacls "cloud-assign.pem"  
cloud-assign.pem SHARMAJIKALAPTO:prtya:(R)  
BUILTIN\Administrators:(F)  
NT AUTHORITY\SYSTEM:(F)  
Successfully processed 1 files; Failed processing 0 files
```

Figure 4: Error Fixing SSH

### 3.5 Step 4: Attempt SSH Connection Again

#### SSH Command

```
ssh -i "C://path//to//your-key.pem" ubuntu@your-ec2-  
instance.amazonaws.com
```

```
D:\SEM-6\CT\Keys>ssh -i cloud-assign.pem ubuntu@ec2-3-108-255-202.ap-south-1.compute.amazonaws.com  
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1021-aws x86_64)  
  
* Documentation:  https://help.ubuntu.com  
* Management:    https://landscape.canonical.com  
* Support:        https://ubuntu.com/pro  
  
System information as of Mon Mar 10 21:31:41 UTC 2025  
  
System load:  0.07      Temperature:   -273.1 C  
Usage of /:   25.1% of 6.71GB  Processes:    109  
Memory usage: 24%      Users logged in: 0  
Swap usage:   0%         IPv4 address for ens5: 172.31.29.172  
  
Expanded Security Maintenance for Applications is not enabled.  
0 updates can be applied immediately.  
  
Enable ESM Apps to receive additional future security updates.  
See https://ubuntu.com/esm or run: sudo pro status  
  
The list of available updates is more than a week old.  
To check for new updates run: sudo apt update  
  
Last login: Mon Mar 10 21:30:30 2025 from 152.59.63.10  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
  
ubuntu@ip-172-31-29-172:~$ |
```

Figure 5: SSH Connect Windows

## 4 Creating an S3 Bucket Using Terraform

### 4.1 Step 1: Install Terraform (If Not Installed)

Run the following commands:

#### Install Terraform

```
sudo apt update sudo apt install -y terraform terraform --version
```

Ensure AWS credentials are configured:

#### Configure AWS CLI

```
aws configure
```

```
D:\SEM-6\CT\Assignment\cloud_assignment_sem-6>terraform -version
Terraform v1.10.5
on windows_amd64
```

Figure 6: Installing Terraform and Configuring AWS CLI

## 4.2 Step 2: Create Terraform Configuration Files

### 4.2.1 Create a Terraform Directory

#### Create Directory

```
mkdir terraform-s3 cd terraform-s3
```

### 4.2.2 Create the main.tf File

Create a new file named `main.tf` and add the following Terraform configuration:

#### Terraform Configuration

```
provider "aws" {
  region = "us-east-1"
}

resource "aws_s3_bucket" "cloud_assignment_bucket" {
  bucket = "cloud-assignment-bucket-1710"
}
```

#### Uploading file to S3

```
resource "aws_s3_object" "example_file" {
  bucket = aws_s3_bucket.cloud_assignment_bucket.id
  key    = "uploaded-file.txt"
  source = "../uploaded-file.txt"
  acl    = "private"
}
```

## 4.3 Step 3: Initialize and Apply Terraform Configuration

#### Initialize Terraform

```
terraform init
```

```

D:\SEM-6\CT\Assignment\cloud_assignment_sem-6\Assignment-1\terraform-s3>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.90.0...
- Installed hashicorp/aws v5.90.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

```

Figure 7: Creating Terraform Configuration Files

### Validate Terraform Configuration

```
terraform validate
```

### Apply Terraform Configuration

```
terraform apply -auto-approve
```

```

D:\SEM-6\CT\Assignment\cloud_assignment_sem-6\Assignment-1\terraform-s3>terraform apply
Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
+ create

Terraform will perform the following actions:

# aws_s3_bucket.cloud_assignment_bucket will be created
+ resource "aws_s3_bucket" "cloud_assignment_bucket" {
+   acceleration_status = (known after apply)
+   acl                 = (known after apply)
+   arn                 = (known after apply)
+   bucket              = "cloud-assignment-bucket-unique-name"
+   bucket_domain_name = (known after apply)
+   bucket_prefix       = (known after apply)
+   bucket_regional_domain_name = (known after apply)
+   force_destroy       = false
+   hosted_zone_id      = (known after apply)
+   id                  = (known after apply)
+   object_lock_enabled = (known after apply)
+   policy              = (known after apply)
+   region              = (known after apply)
+   request_payer       = (known after apply)
+   tags_all            = (known after apply)
+   website_domain       = (known after apply)
+   website_endpoint    = (known after apply)
+   cors_rule (known after apply)
+   grant (known after apply)
+   lifecycle_rule (known after apply)
+   logging (known after apply)
+   object_lock_configuration (known after apply)
+   replication_configuration (known after apply)
+   server_side_encryption_configuration (known after apply)
+   versioning (known after apply)
+   website (known after apply)
}

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_s3_bucket.cloud_assignment_bucket: Creating...
aws_s3_bucket.cloud_assignment_bucket: Still creating... [10s elapsed]
aws_s3_bucket.cloud_assignment_bucket: Creation complete after 13s [id=cloud-assignment-bucket-unique-name]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

```

Figure 8: Initializing and Applying Terraform Configuration

## 4.4 Step 4: Verify S3 Bucket Creation

After successful execution, verify the bucket creation in AWS Console.

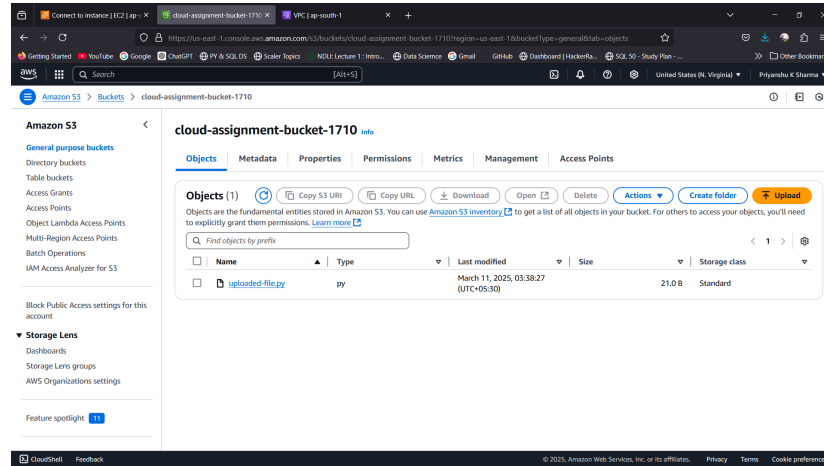


Figure 9: S3 Bucket Created Successfully

## 5 Conclusion

This report covered the process of launching an EC2 instance, fixing SSH private key permissions on Windows, and setting up an S3 bucket using Terraform. These are fundamental cloud operations useful in AWS environments.