🚀 Uploading a File from an EC2 Instance to S3 Using Python – Step-by-Step

Cloud skills aren't just about theory — they're about **getting hands-on and making services talk to each other**.
Recently, I walked through connecting **Amazon EC2** with **Amazon S3** using **Python**, and I wanted to share my process so others can follow along.
This combines **AWS IAM**, **EC2**, **S3**, and **Boto3** — a great mini-project for cloud learners and a practical skill for real-world workflows.

Step 1 – Launch an EC2 Instance

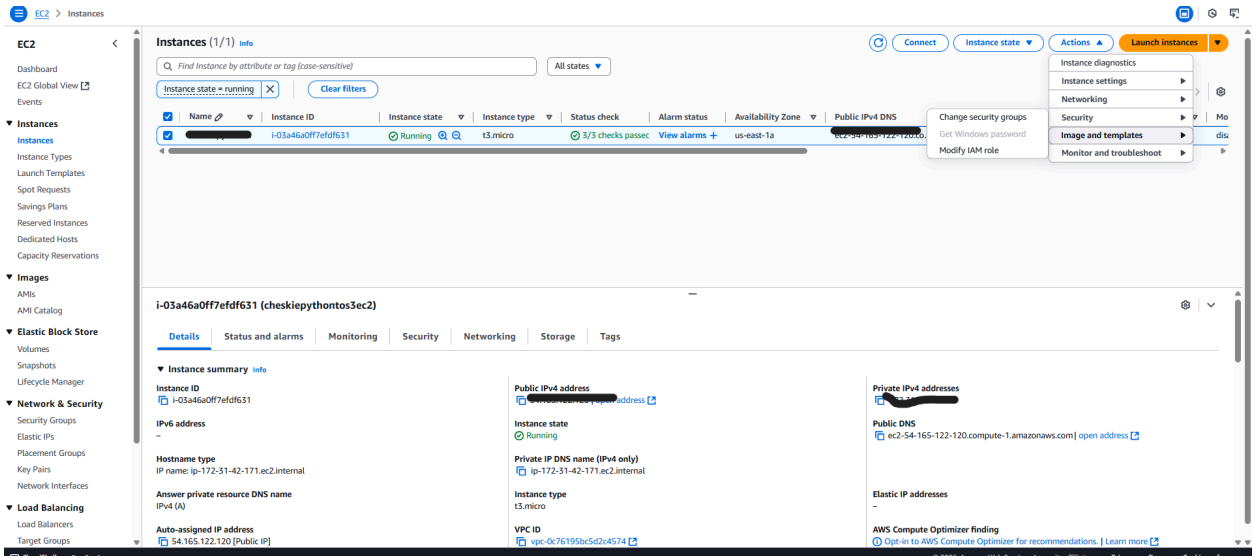
Create an **EC2 instance** with **Amazon Linux** as the operating system.
Choose your key pair and click **Launch Instance**.

Step 2 – Create an IAM Role with S3 Full Access

1. In the AWS search bar, type **Roles** and select it.
2. Under **Use Case**, choose **EC2** → **Create Role**.
3. In the search box, type and select **AmazonS3FullAccess** (or equivalent).
4. Give your role a specific name and click **Create Role**.

Step 3 – Attach the Role to Your EC2 Instance

Go back to your EC2 dashboard → Select your instance → Click **Actions** (top right) → **Security** → **Modify IAM Role**.



Step 4 – Create an S3 Bucket

1. In the AWS search bar, type **S3** and open **S3 Buckets**.
2. Click **Create Bucket**.
3. Give your bucket a unique name → Click **Create Bucket** at the bottom.

Step 5 – Prepare Your Python Script

Write the following Python code to upload files to S3.

```
#!/bin/bash

import boto3
import sys

def upload_to_s3(local_file, bucket_name, s3_key):
    s3 = boto3.client('s3')
    try:
        s3.upload_file(local_file, bucket_name, s3_key)
        print(f"Upload successful: {local_file} --> s3://{bucket_name}/{s3_key}")
    except Exception as e:
        print(f"Upload failed: {e}")

if __name__ == "__main__":
    if len(sys.argv) != 4:
        print("Usage: python s3_upload.py <local_file> <bucket_name> <s3_key>")
        sys.exit(1)

    local_file = sys.argv[1]
    bucket_name = sys.argv[2]
    s3_key = sys.argv[3]

    upload_to_s3(local_file, bucket_name, s3_key)
```

Then, from the EC2 dashboard:

- Click your instance → **Connect** → Ensure **EC2 Instance Connect** is selected → Click **Connect** again.

Step 6 – Create the Python File on EC2 Instance Connect

vi uploadtos3.py

Paste your Python code from earlier.

Press **Shift + :**, type `wq`, and hit **Enter** to save and exit.

Step 7 – Make the Script Executable

chmod +x uploadtos3.py

Step 8 – Create an Empty File to Upload

touch uploadtos3.py

Step 9 – Add Content to the File

echo "Hello World" > uploadtos3.py

Step 10 – Repeat Steps 5&6

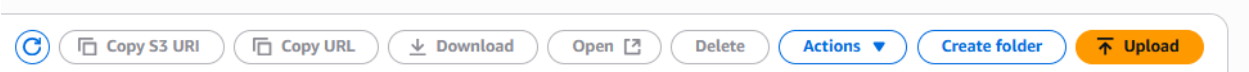
Step 11 – Install Required Python & S3 Intergration Tools

sudo yum install pip
sudo pip install boto3

Step 12 – Create a Folder Inside Your S3 Bucket

1. Go back to **S3 Buckets** → Open your bucket.

2. Click **Create Folder**.



3. Name the folder `txt` → Click **Create Folder**.

Step 13 – Create a Test File

echo "Hello World" > test.txt

Step 14 – Run the Upload Script

python3 uploadtos3.py test.txt YourS3BucketName YourS3BucketName/txt

Replace `YourS3BucketName` with your actual bucket name.
You should see **"Upload successful"**.

Step 15 – Verify in S3

Go back to your S3 bucket in AWS → Open the `txt` folder → Confirm that `test.txt` is there.

✅ **Done!** You've successfully uploaded a file from EC2 to S3 using Python.

💡 **Why this is useful:**

- Automates file uploads to S3 from compute instances.
- Demonstrates IAM role best practices (no hardcoding credentials).
- Combines multiple AWS services in one workflow.