**Develop an automation program that takes a YAML configuration file as input and deploys a Linux AWS EC2 instance with two volumes and two users.**

**Services used:** Lambda, S3, Key Pairs, IAM roles, EC2 Instances, Security Groups, CloudWatch Logs.

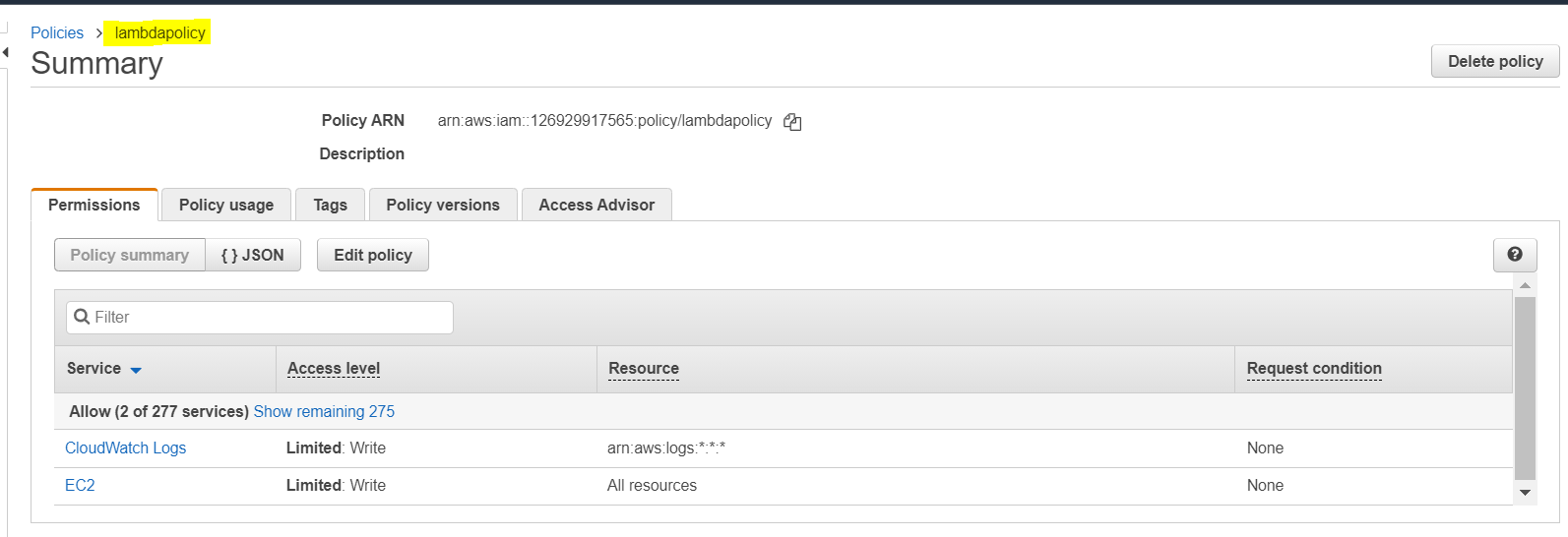
**Overview of the Program:**

I have used AWS Lambda for developing an automation program which further uses Python, Boto3 and takes a yaml configuration file as an input, that deploys a Linux AWS Ec2 instance with two volumes and two users. The trigger for Lambda function is the S3 PUT, so whenever the yaml file is uploaded in S3, it acts like a trigger to the lambda function and the program will be running where it fetches the data from S3 yaml file for further deploying the ec2 instance in the default VPC with the defined security group( Port 22- so that we can SSH into it) , and the users are created by defining a user data script at run time, along with two volumes of different sizes mentioned in the yaml file configuration.

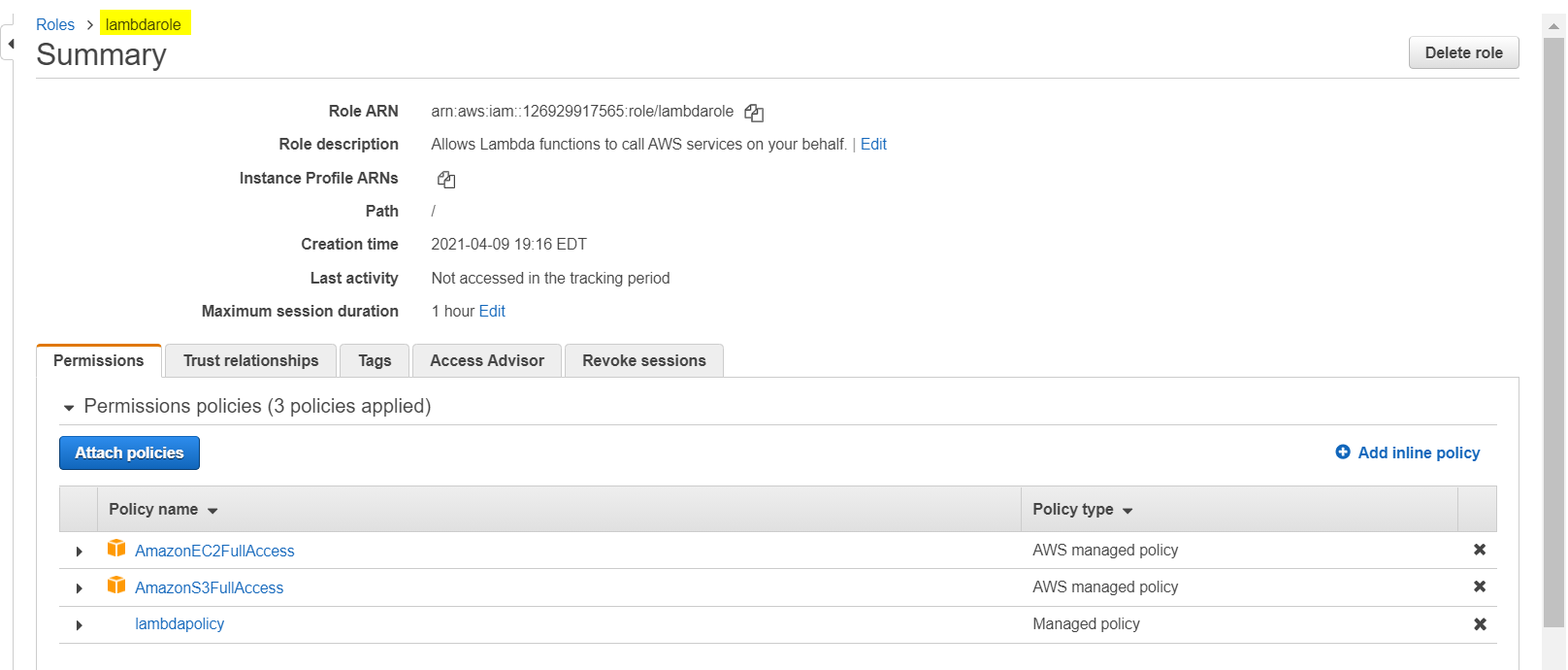
**Prerequisites before running the program:**

1. Create a policy “***lambdapolicy***” – Choose Lambda service and copy the JSON file content. (***lambdapolicy*.*json*** in present in the repository)

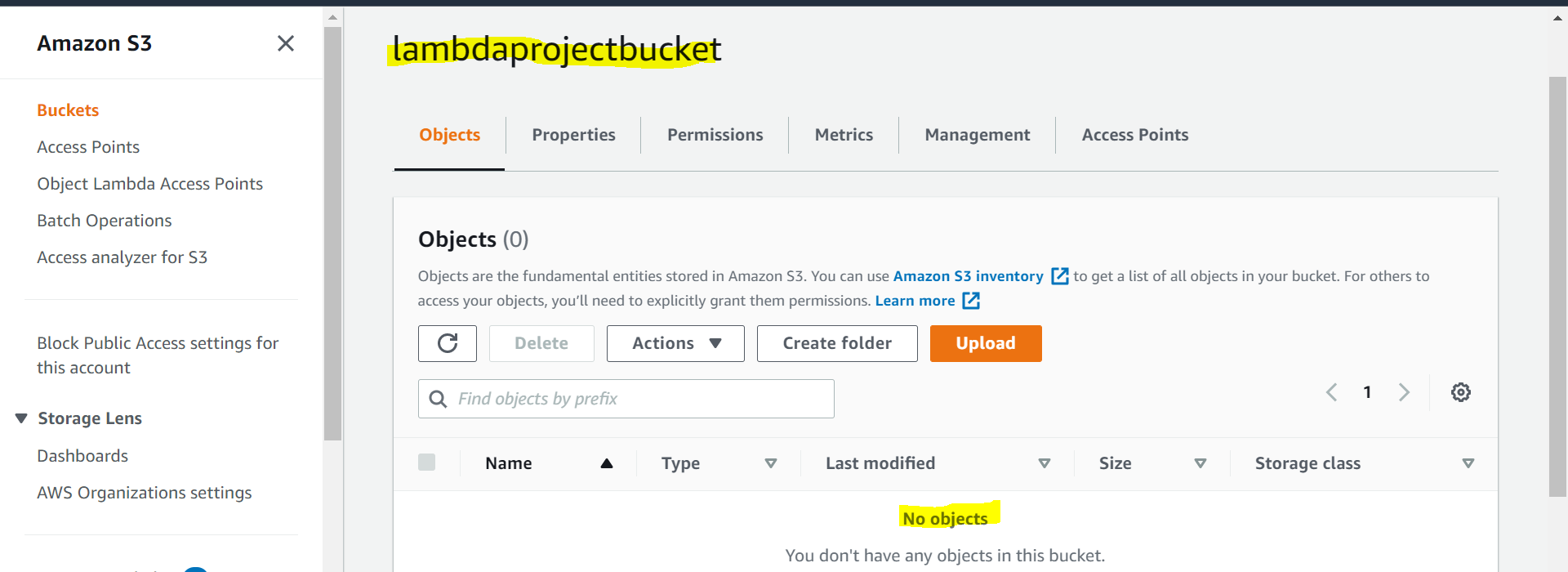
Reference-<https://docs.aws.amazon.com/server-migration-service/latest/userguide/cwe-sms.html>



1. Create an IAM role “***lambdarole***” -choose Lambda service (allows Lambda functions to call AWS services on our behalf.) and attach “[***AmazonEC2FullAccess***](https://console.aws.amazon.com/iam/home?region=us-east-2#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonEC2FullAccess)”, “[***AmazonS3FullAccess***](https://console.aws.amazon.com/iam/home?region=us-east-2#/policies/arn%3Aaws%3Aiam%3A%3Aaws%3Apolicy%2FAmazonS3FullAccess) and “***lambdapolicy***”.

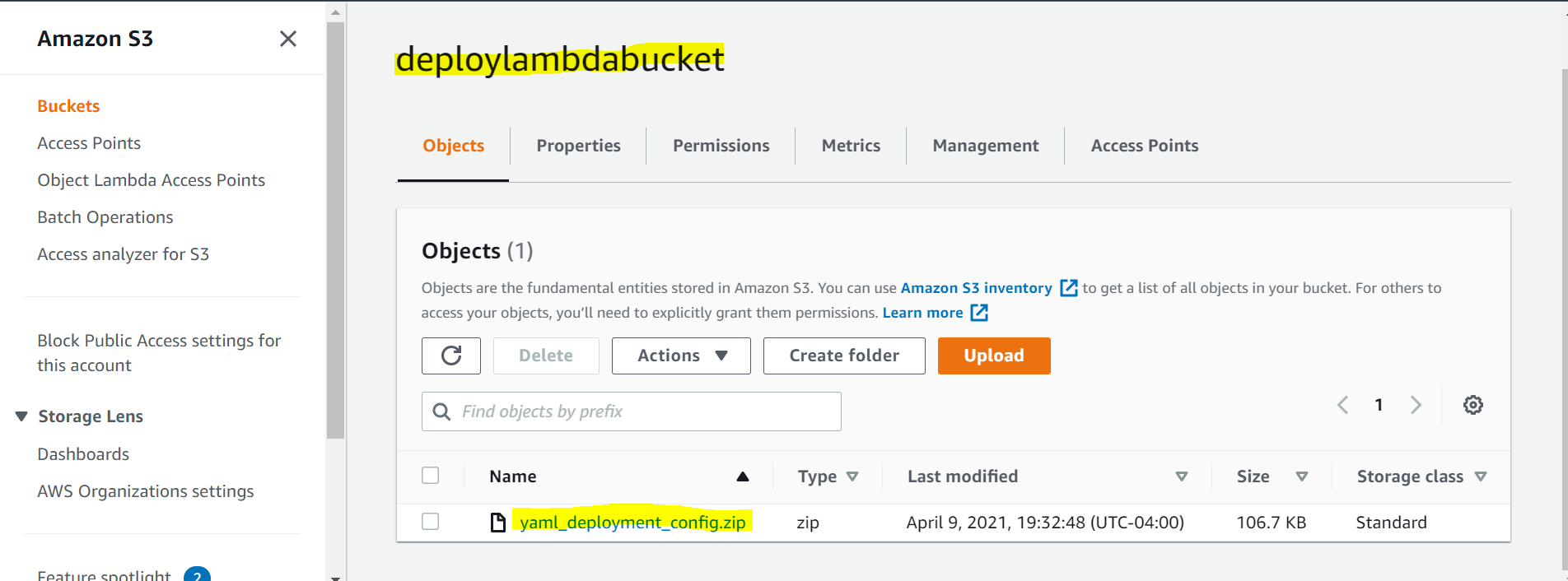


1. Create an S3 bucket “***lambdaprojectbucket***” in us-east-2 region with **turning off block all public access** and we will upload the “***deployec2.yaml”*** file (mentioned in the repository) later.

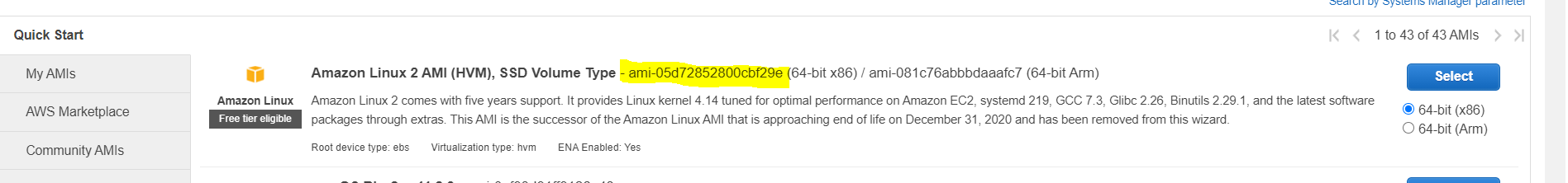


1. Create another S3 bucket “***deploylambdabucket***” like before bucket that stores the “***yaml\_deployment\_config. Zip”*** file -which contains yaml packages to handle the yaml data and lambda\_function.py file.

The **yaml\_deployment\_config.zip** file and **lambda\_function.py** file is present in the repository.



1. Create two SSH keypairs “***user1***” and “***user2***” and get the public keys. Add the public keys in the yaml file. (The private keys for both users “***user1.pem***”, “***user2.pem”*** are included in the repository)
2. Create a SSH keypair “***projectkey***” for launching the instance.
3. Make note of the “***Amazon Linux 2 AMI image id”*** in the “***us-east-2”*** region (Ohio region) since we ae going to deploy our Linux instance in that region using this image.



**Steps to follow:**

1. Create a Lambda function in Ohio region.

Author from Scratch

Function Name: ***lambdafunction***

Runtime: Python 3.8

Execution role: Use and existing role – Choose ***lambdarole.***

Create Function.

1. Add Trigger

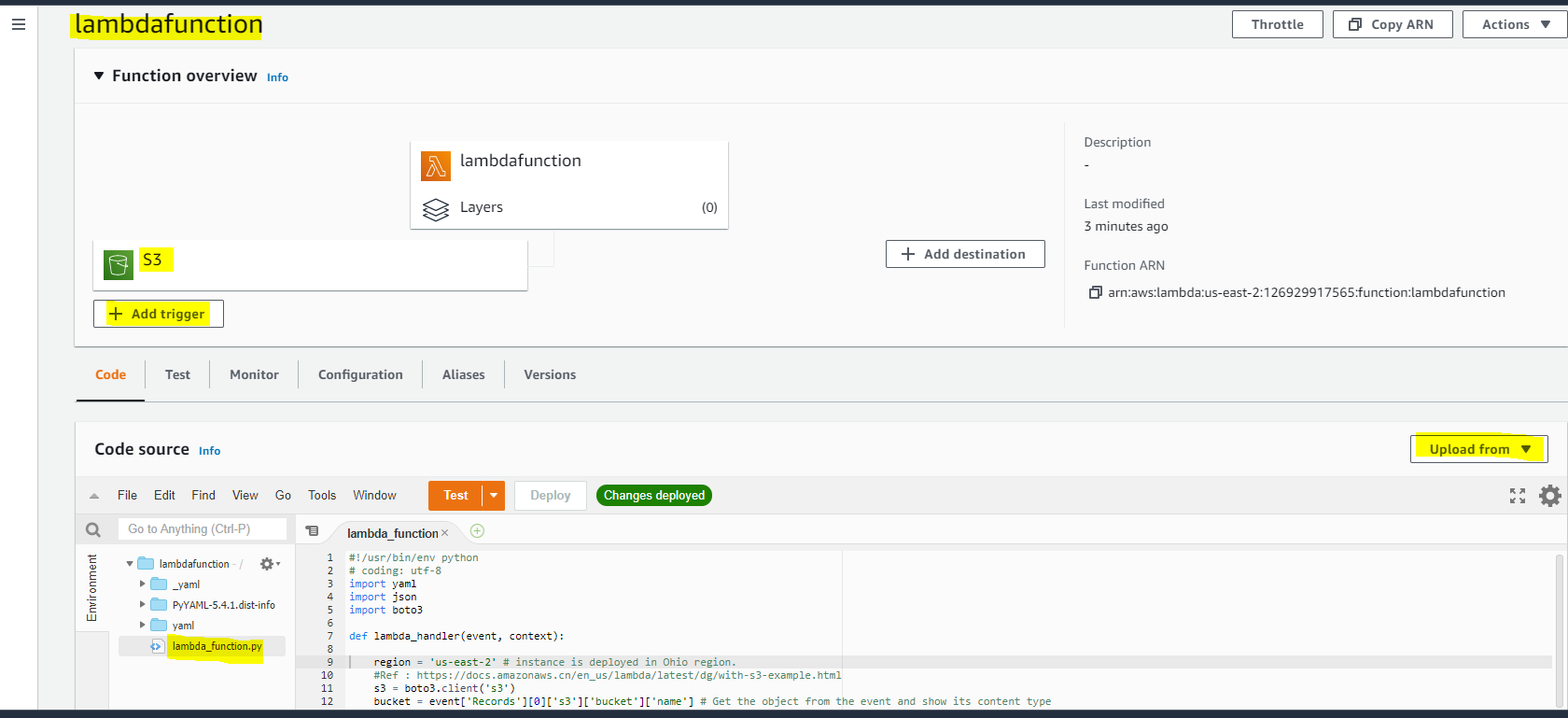
Trigger: S3

Bucket: ***lambdaprojectbucket***

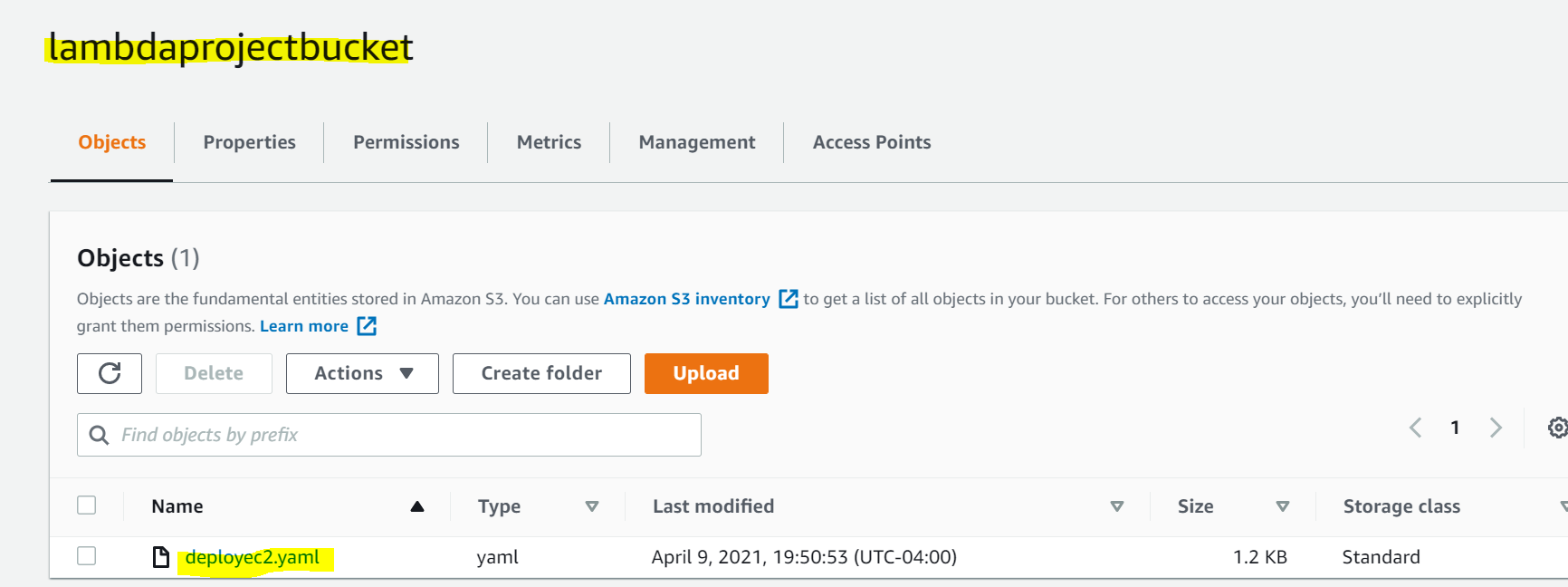
Event: PUT

1. Under Code section – Upload from Amazon S3 location. Paste the S3 bucket ***deploylambdabucket*** link URL (s3://deploylambdabucket/yaml\_deployment\_config.zip) and Save.

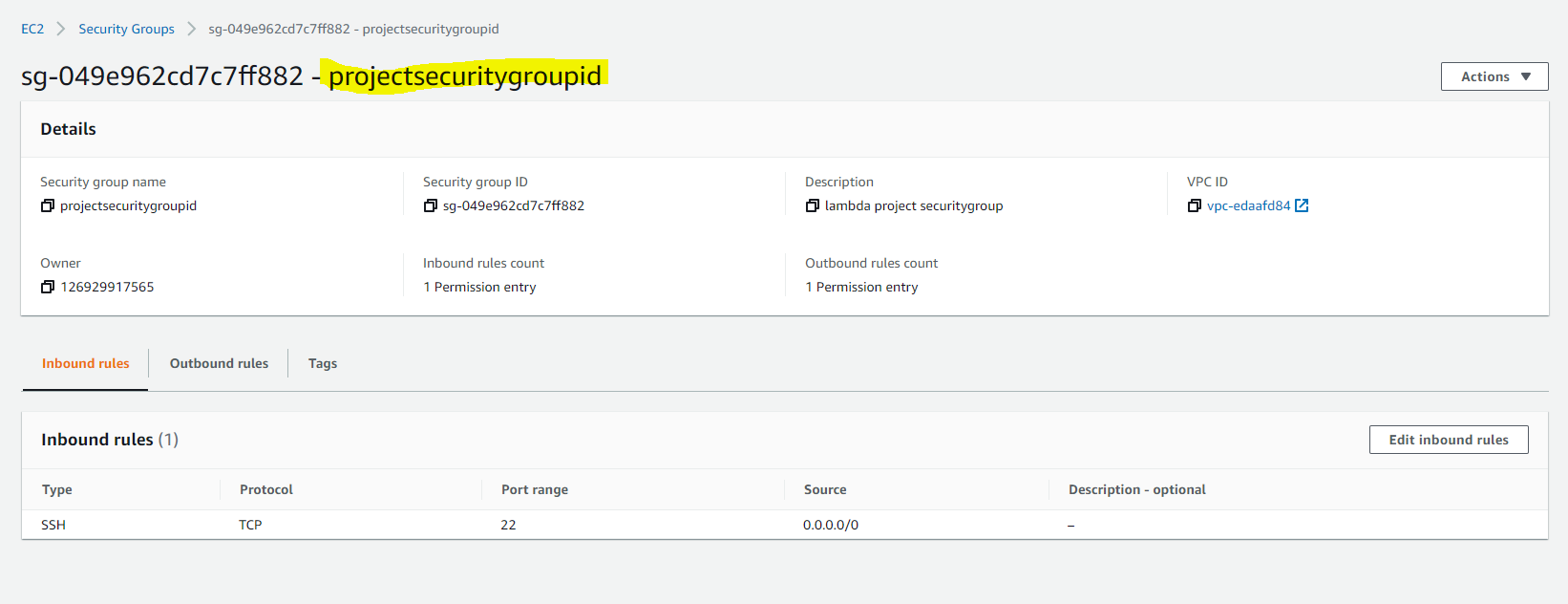
Can also increase the timeout to 5 minutes to avoid timeout errors.

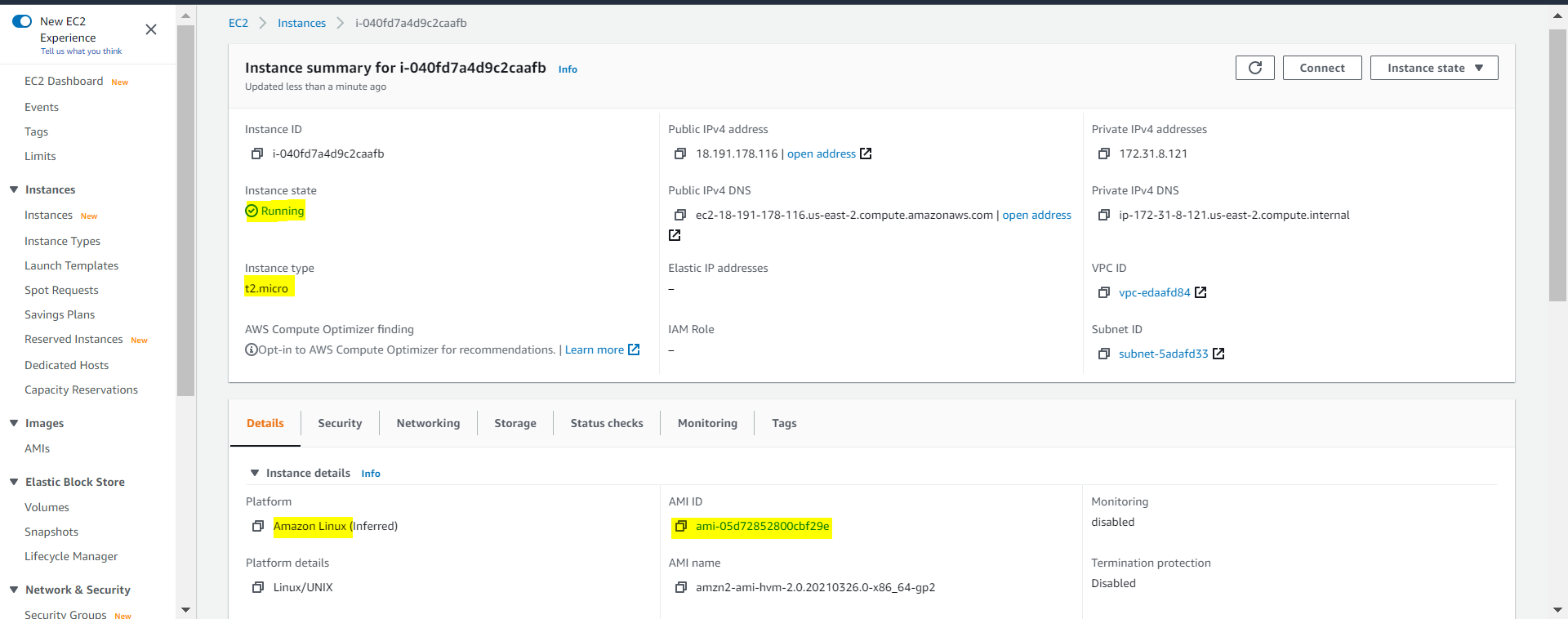


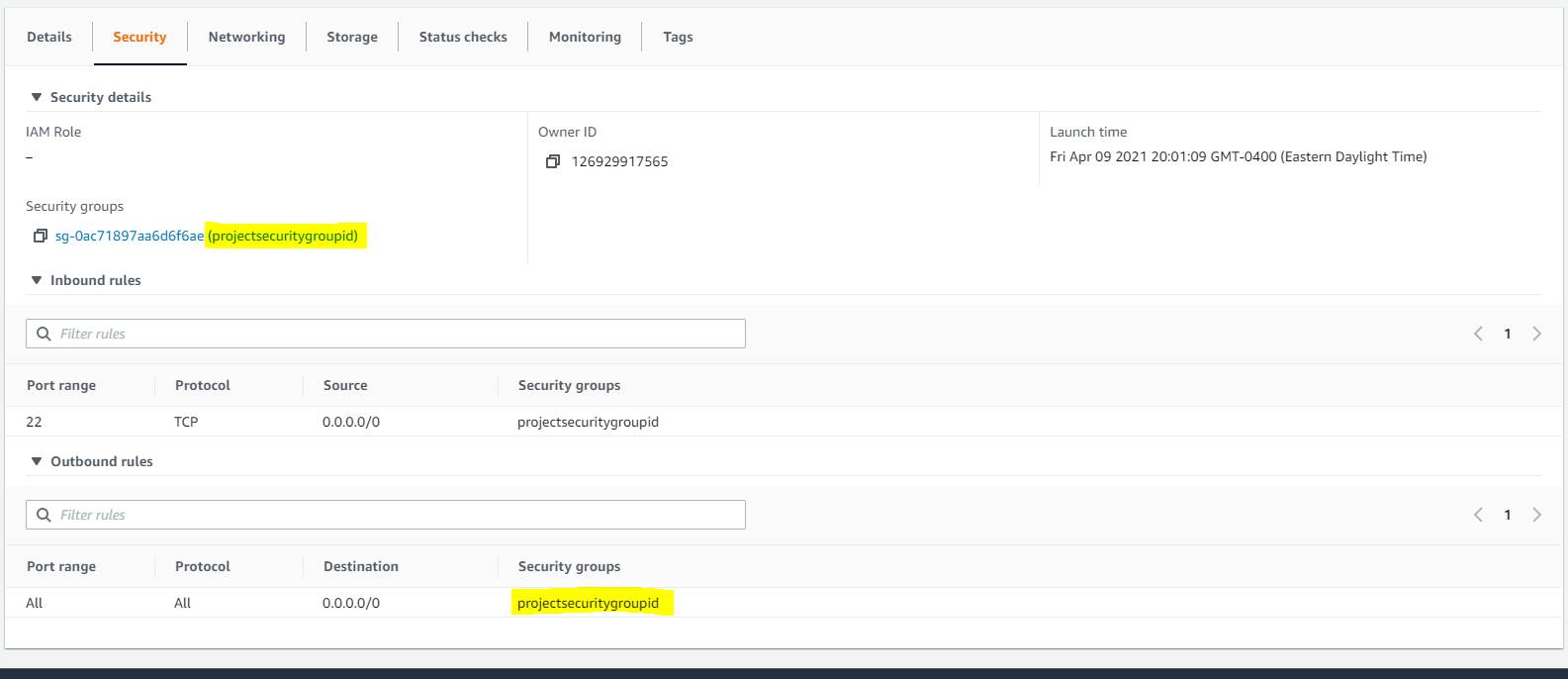
1. Deploy the code if any changes made.
2. Now upload the ***deployec2.yml file*** tothe ***lambdaprojectbucket.***

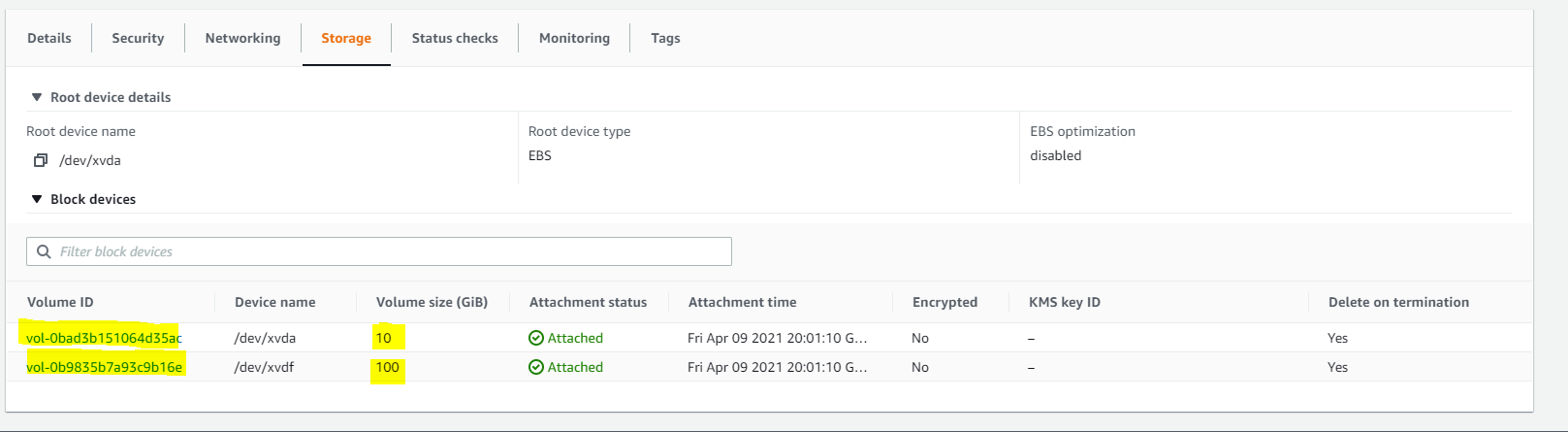


1. As soon as the file is uploaded the, it acts like a trigger to the lambda function and the program runs. Wait for some time, we can see that EC2 instance(with two users and two volumes) and the associated security group (**Name**: *projectsecuritygroupid*, **Description**: *lambda project security group*) has been created.

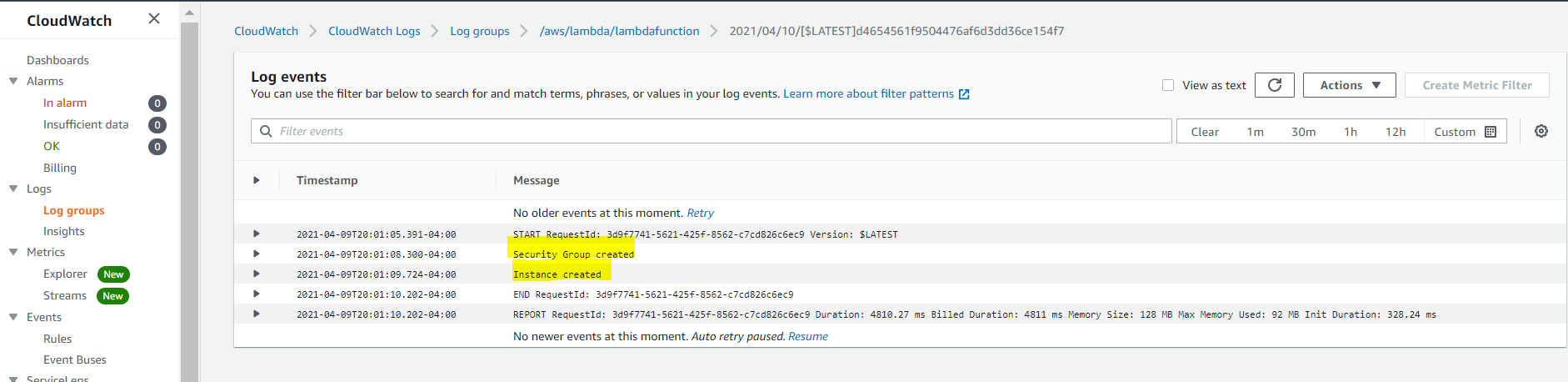






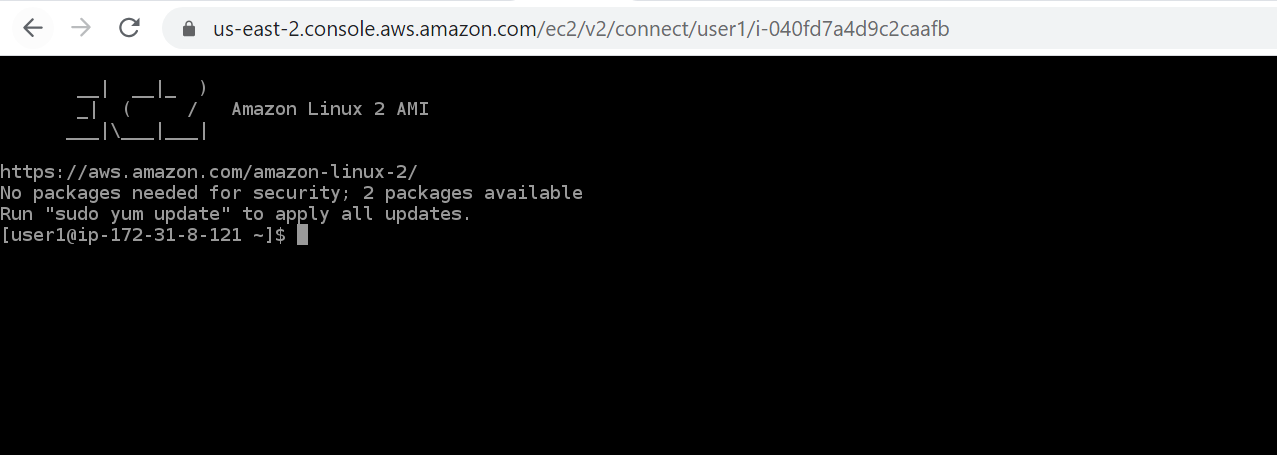


1. We can as well view the logs in the CloudWatch Logs. If there is any error, we can identify here.



1. After that, we can SSH into the instance as user1 and user2 by using private keys user1.pem and user2.pem and therefore read from and write to each of two volumes.

Ec2 Connect using user1



Ec2 connect using user2

