Create an S3 Bucket event to get SNS Email Notification on Object upload using Terraform

Level: Intermediate

Amazon S3 Amazon SNS Amazon Web Services Terraform

0h 44m 47s left

End Lab

Open Console

Validation

Lab Credentials

User Name

Whiz_User_80425.72740096

Password

a07b9df6-7039-4a91-848f-ec7ec349d65d

Access Key

AKIAZOUNL5AS5XFEOZOX

Secret Key

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

Lab Resources

No Lab Resources Found

Support Documents

1. FAQs and Troubleshooting

Need help?

How to use Hands on Lab

Troubleshooting Lab

FAQs

Submit Feedback | Share

Lab Overview | Lab Steps | Lab Validation

Cloud Architect, Cloud Developer, Cloud DevOps Engineer, Cloud Administrator

Storage, Serverless, Infrastructure

Task 1: Sign in to AWS Management Console

1. Click on the **Open Console** button, and you will get redirected to AWS Console in a new browser tab.
2. On the AWS sign-in page,
    - Leave the Account ID as default. Never edit/remove the 12 digit Account ID present in the AWS Console. otherwise, you cannot proceed with the lab.
    - Now copy your **User Name** and **Password** in the Lab Console to the **IAM Username and Password** in AWS Console and click on the **Sign in** button.
3. Once Signed In to the AWS Management Console, Make the default AWS Region as **US East (N. Virginia) us-east-1.**

Note : If you face any issues, please go through **FAQs and Troubleshooting for Labs**.

Task 2: Setup Visual Studio Code

1. Open the visual studio code.

2. If you have already installed and using Visual studio code, open a new window.
3. A new window will open a new file and release notes page (only if you have installed or updated Visual Studio Code recently). Close the Release notes tab.
4. Open Terminal by selecting View from the Menu bar and choose Terminal.
5. It may take up to 2 minutes to open the terminal window.
6. Once the terminal is ready, let us navigate to the Desktop.

```
cd Desktop
```

7. Create a new folder by running the below command.

```
mkdir task_100012_sns_s3
```

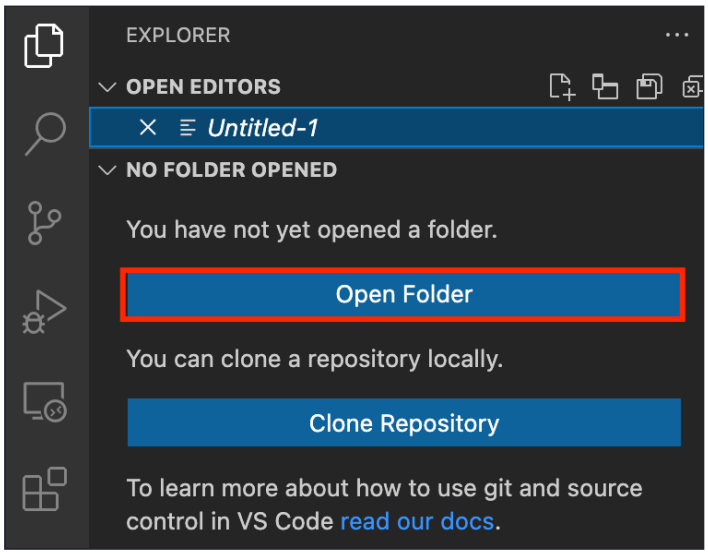
8. Change your present working directory to use the newly created folder by running the below command:

```
cd task_100012_sns_s3
```

9. Get the location of the present working directory by running the below command:

```
pwd
```

10. Note down the location, as you will open the same in the next steps.
11. Now click on the first icon Explorer present on the left sidebar.
12. Click on the button called Open folder and navigate to the location of folder **task_100012_sns_s3**.



13. (Optional) Click on Authorize button for allowing Visual Studio Code to use the **task_100012_sns_s3** folder. This will only be asked when you have been using Visual Studio code for a while as you are allowing a new folder to be accessed by VSC.

14. Visual Studio Code is now ready to use.

Task 3: Create a variables file

In this task, you will create variable files where you will declare all the global variables with a short description and a default value.

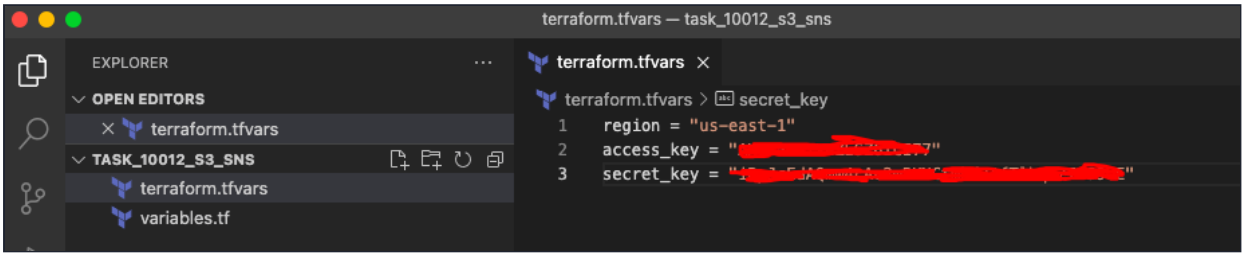
1. To create a variable file, expand the folder **task_100012_sns_s3** and click on the **New File** icon to add the file.
2. Name the file as **variables.tf** and press **Enter** to save it.
3. **Note:** Don't change the location of the new file, keep it default, i.e. inside the **task_100012_sns_s3** folder**.**
4. Paste the below contents in **variables.tf** file.

```
variable "access_key" {
    description = "Access key to AWS console"
}
variable "secret_key" {
    description = "Secret key to AWS console"
}
variable "region" {
    description = "AWS region"
}
variable "endpoint" {
    description = "SNS topic subscription endpoint"
}
```

5. In the above content, you are declaring a variable called, access_key, secret_key, and region with a short description of all 3.
6. We are also declaring a variable called endpoint where you'll input your email address while creating terraform resources.
7. After pasting the above contents, save the file by pressing **ctrl + S**.
8. Now expand the folder **task_100012_sns_s3** and click on the **New File** icon to add the file.
9. Name the file as **terraform.tfvars** and press **Enter** to save it.
10. Paste the below content into the **terraform.tfvars** file.

```
region = "us-east-1"
access_key = "<YOUR AWS CONSOLE ACCESS ID>"
secret_key = "<YOUR AWS CONSOLE SECRET KEY>"
```

11. In the above code, you are defining the dynamic values of variables declared earlier.
12. Replace the values of access_key and secret_key by copying from the lab page.
13. After replacing the values of access_key and secret_key, save the file by pressing **Ctrl + S**.



Task 4: Create SNS, S3 and its components in main.tf file

In this task, you will create a **main.tf** file where you will add details of the provider and resources. You will also create SNS topic, subscription, and S3 bucket in the main.tf file.

1. To create a **main.tf** file, expand the folder **task_100012_sns_s3** and click on the **New File** icon to add the file.
2. Name the file as **main.tf** and press **Enter** to save it.SNS
3. Paste the below content into the **main.tf** file.

```
provider "aws" {
    region     = "${var.region}"
    access_key = "${var.access_key}"
    secret_key = "${var.secret_key}"
}
```

4. In the above code, you are defining the provider as aws.
5. Now, we will proceed with creating an S3 Bucket. Keep in mind that the bucket name must be unique globally across all existing bucket names in Amazon S3.
6. To ensure uniqueness, we will generate a random string that will be appended at the end of the bucket name. This approach will help avoid naming conflicts and ensure a unique bucket identifier.
7. Paste the below content into the **main.tf** file after the provider.

```
############ Creating a Random String ############
```

```
resource "random_string" "random" {
      length = 6
      special = false
      upper = false
  }

############ Creating an S3 Bucket ############
resource "aws_s3_bucket" "bucket" {
  bucket = "whizbucket-${random_string.random.result}"
  force_destroy = true
}
```

8. After creating the S3 Bucket, the next step is to create an SNS Topic and set its access policy. This will grant the SNS Topic the necessary permissions to send notifications based on events that occur in the S3 bucket.
9. With this setup, the SNS Topic will be able to handle event notifications triggered by the S3 bucket and enable seamless communication between the two services.
10. Paste the below content into the **main.tf** file after the previous code.

```
############ Creating an SNS Topic ############

resource "aws_sns_topic" "topic" {
  name = "whiz-s3-event-notification"
  policy = <<POLICY
{
    "Version":"2012-10-17",
    "Statement":[
        {
        "Effect": "Allow",
        "Principal": { "Service": "s3.amazonaws.com" },
        "Action": "SNS:Publish",
        "Resource": "arn:aws:sns:*:*:whiz-s3-event-notification",
        "Condition":{
            "ArnLike":{"aws:SourceArn":"${aws_s3_bucket.bucket.arn}"}
            }
        }
    ]
}
POLICY
}
```

11. Following the creation of the SNS Topic, the next step is to establish a subscription to the topic. For this subscription, we will use email as the communication protocol.
12. However, since the email address (endpoint) is not declared as a variable in the Terraform configuration and is currently empty, I will input the email address during the Terraform creation process.

13. This will allow us to set up the subscription and ensure that notifications are sent to the specified email address whenever events are triggered by the S3 bucket and processed by the SNS Topic.

14. Paste the below content into the **main.tf** file after the previous code.

```
############ Creating SNS Topic Subscription ############
resource "aws_sns_topic_subscription" "topic-subscription" {
  topic_arn = aws_sns_topic.topic.arn
  protocol  = "email"
  endpoint  = "${var.endpoint}"
}
```

15. Lastly, we will configure an event notification for the S3 bucket and link it to the previously created SNS Topic. This setup ensures that whenever a new object is uploaded to the S3 bucket, an event notification will be triggered and sent to the email address subscribed to the SNS Topic.

16. Paste the below content into the **main.tf** file after the previous code.

```
############ Creating bucket event notification ############
resource "aws_s3_bucket_notification" "bucket_notification" {
  bucket = aws_s3_bucket.bucket.id
  topic {
    topic_arn     = aws_sns_topic.topic.arn
    events        = ["s3:ObjectCreated:*"]
  }
}
```

---

17. Save the file by pressing **Ctrl + S**.

Task 5: Create an Output file

In this task, you will create an **output.tf** file where you add details of the output you want to display.

1. To create an **output.tf** file, expand the folder **task_100012_sns_s3** and click on the **New File** icon to add the file.

2. Name the file as **output.tf** and press **Enter** to save it.

3. Paste the below content into the **output.tf** file.

```
output "s3-bucket-name" {
  value = aws_s3_bucket.bucket.id
```

```
}

output "sns-topic-arn" {
  value = aws_sns_topic.topic.arn
}
```

4. In the above code, we will extract the S3 Bucket name and SNS topic ARN. We will display it once the resources are created.

Task 6: Confirm the installation of Terraform by checking the version

1. In the Visual Studio Code, open Terminal by selecting **View** from the Menu bar and choose **Terminal**.
2. If you are not in the newly created folder change your present working directory by running the below command.

```
cd task_100012_sns_s3
```

3. To confirm the installation of Terraform, run the below command to check the version:

```
terraform version
```

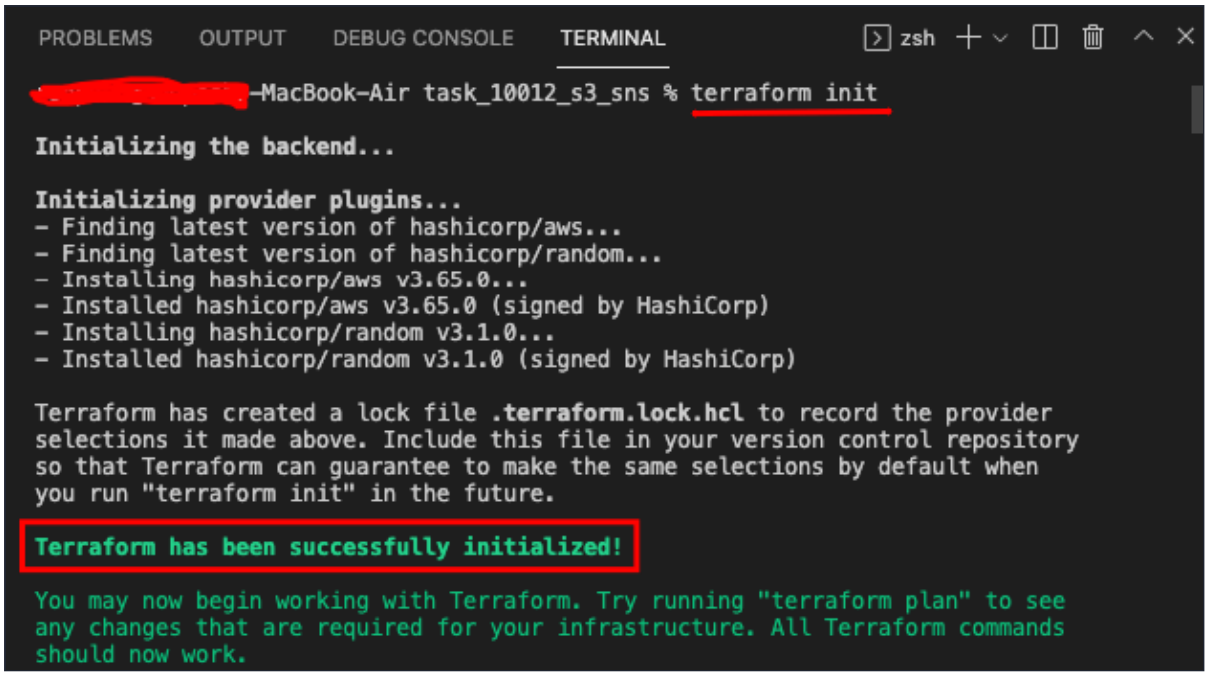
4. If you are getting output as command not found: terraform, this means that terraform is not installed on your system, To install terraform follow the official guide link provided in the Prerequisite section above.

Task 7: Apply terraform configurations

1. Initialize Terraform by running the below command,

```
terraform init
```

2. Note: terraform init will check for all the plugin dependencies and download them if required, this will be used for creating a deployment plan.

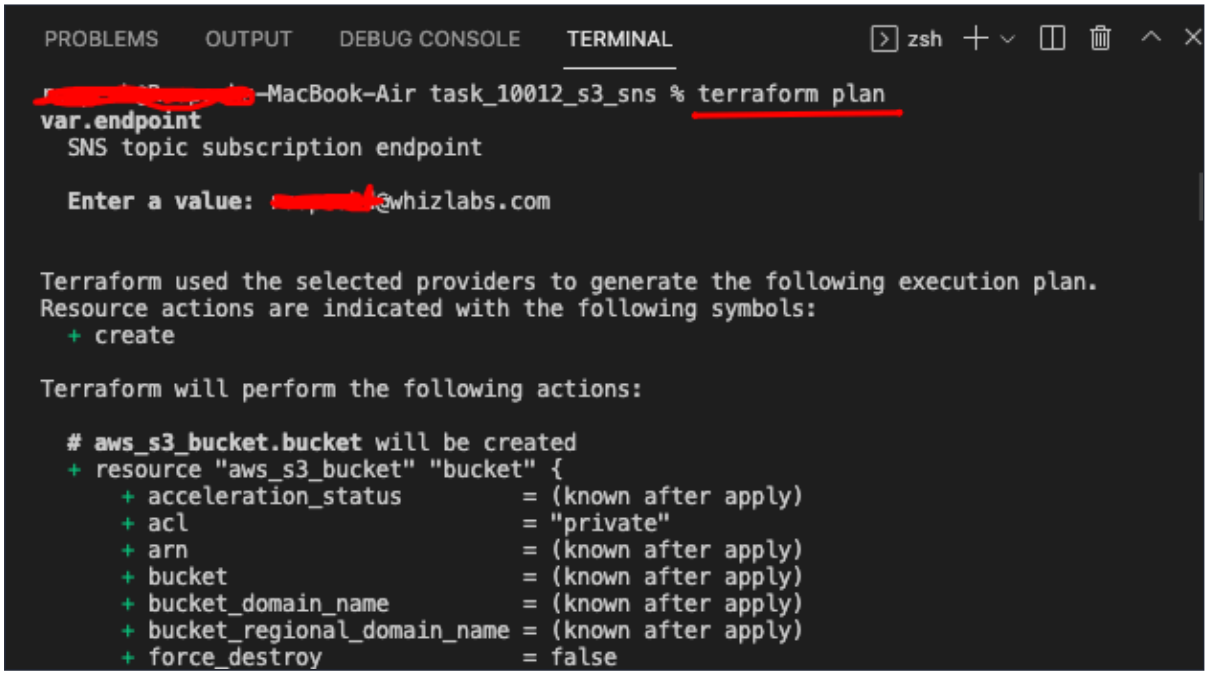


3. To generate the action plans run the below command,

```
terraform plan
```

4. Since we have not declared the endpoint variable, terraform will ask for the input. Enter your email address and press Enter.

5. Review the whole generated plan.

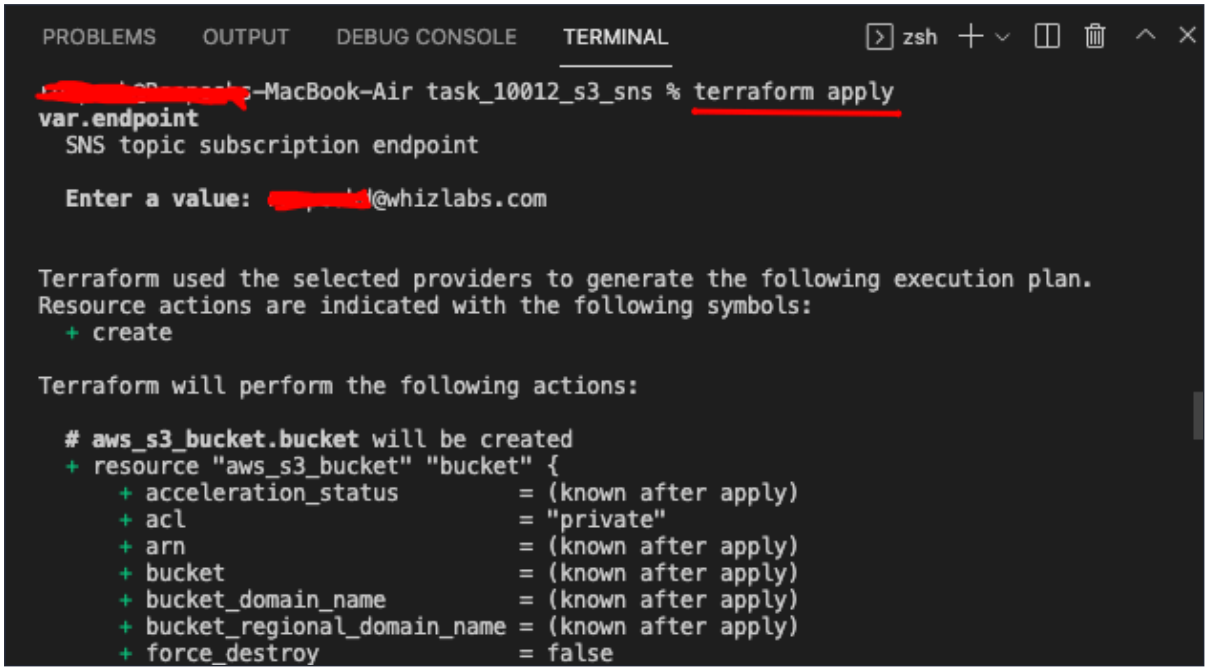


6. To create all the resources declared in main.tf configuration file, run the below command,

```
terraform apply
```

7. You will be asked to input the variable endpoint. Provide your email address and press Enter.

    - **Note: Make sure you give a valid email address as you will receive an SNS notification to this email address.**



8. You will be able to see the resources which will be created, approve the creation of all the resources by entering **yes**.
9. It may take up to 1-3 minutes for the terraform apply command to create the resources.
10. Id's of all the resources created by terraform will be visible there.
11. The output i.e S3 bucket name and SNS topic ARN is extracted and displayed.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL                                zsh

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

random_string.random: Creating...
random_string.random: Creation complete after 0s [id=x5zqi3]
aws_s3_bucket.bucket: Creating...
aws_s3_bucket.bucket: Still creating... [10s elapsed]
aws_s3_bucket.bucket: Creation complete after 16s [id=whizbucket-x5zqi3]
aws_sns_topic.topic: Creating...
aws_sns_topic.topic: Creation complete after 4s [id=arn:aws:sns:us-east-1:922922497623:whiz-s3-event-notification]
aws_sns_topic_subscription.topic-subscription: Creating...
aws_s3_bucket_notification.bucket_notification: Creating...
aws_s3_bucket_notification.bucket_notification: Creation complete after 2s [id=whizbucket-x5zqi3]
aws_sns_topic_subscription.topic-subscription: Creation complete after 3s [id=arn:aws:sns:us-east-1:922922497623:whiz-s3-event-notification:42358edf-7013-4702-9f03-f08331f4453a]

Apply complete! Resources: 5 added, 0 changed, 0 destroyed.
```

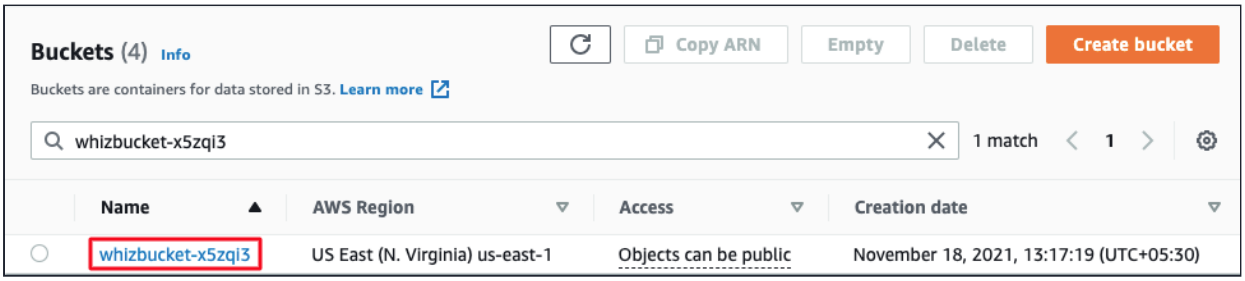
```
Outputs:

s3-bucket-name = "whizbucket-x5zqi3"
sns-topic-arn = "arn:aws:sns:us-east-1:922922497623:whiz-s3-event-notification"
```

12. Optionally, you can note down the IDs of all the resources.

Task 9: Check the resources in AWS Console

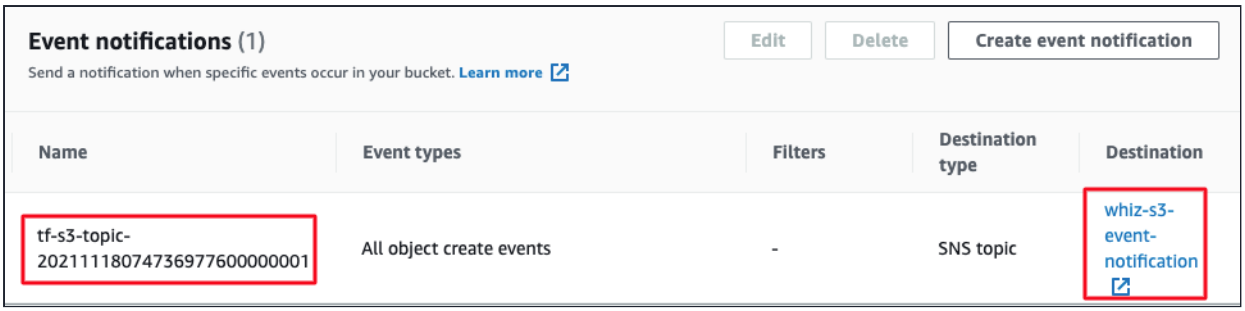
1. Navigate to **S3** by clicking on **Services** on the top, then click on **S3** in the **Storage** section.

2. Copy the S3-bucket-name from the output of the terraform and paste it in the S3 console to filter the bucket. You will be able to see the S3 bucket created by the terraform.



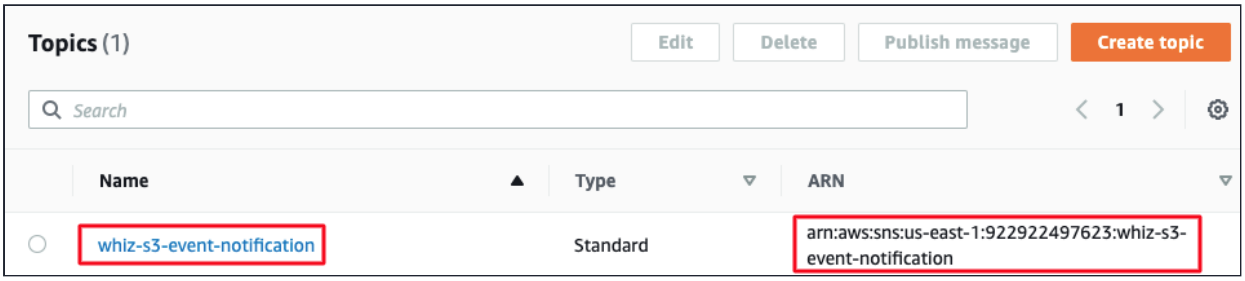
3. Click on the bucket and navigate to the **Properties** tab.

4. Scroll down to the **Event notifications**. You will be able to see an event notification created.



5. Navigate to **SNS** by clicking on **Services** on the top, then click on **SNS** in the **Application Integration** section.

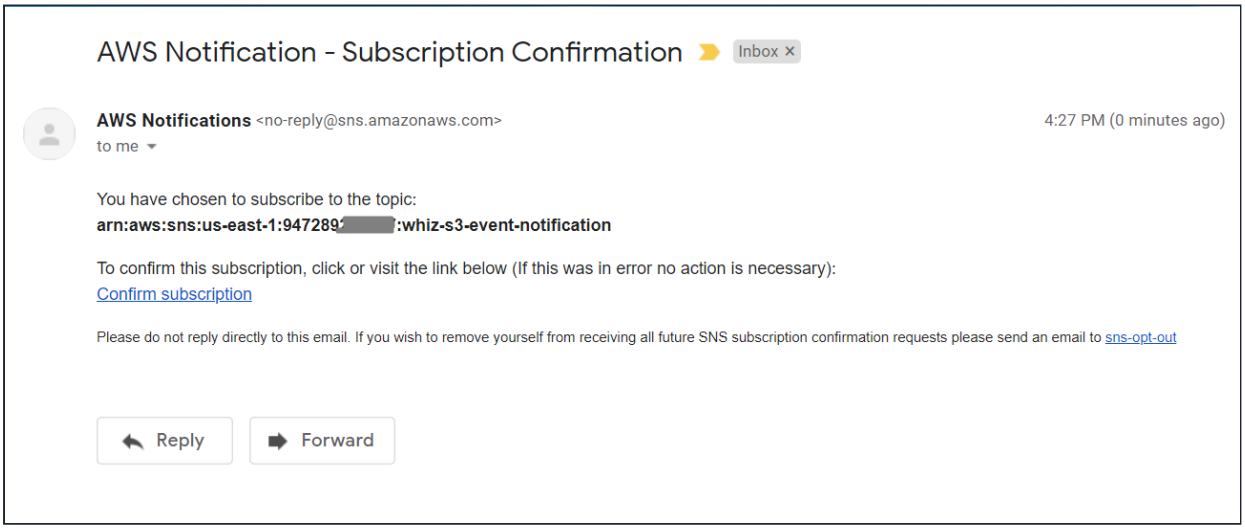
6. Click the topics on the left panel. You will be able to see the created SNS topic which we have created in the terraform.



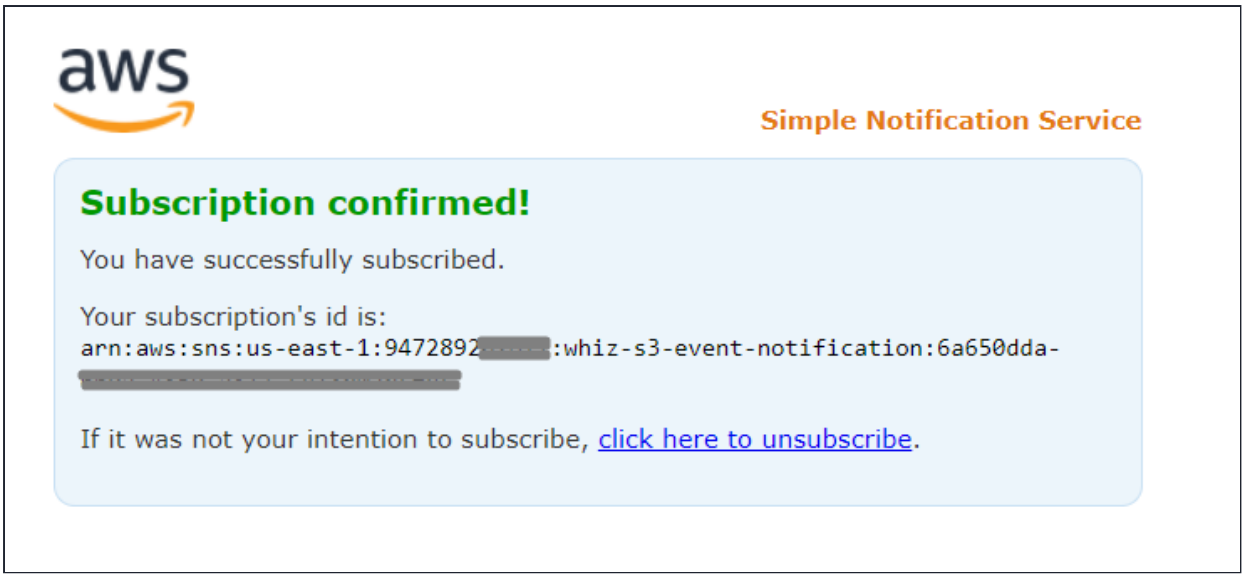
7. Click on the topic and in the **Subscriptions** tab, you will see the created subscription with the input of your email address.

8. The status of the subscription is in **Pending confirmation**.

9. You will receive an email to confirm your SNS subscription. Check your Spam if you don't see the email in your Inbox.

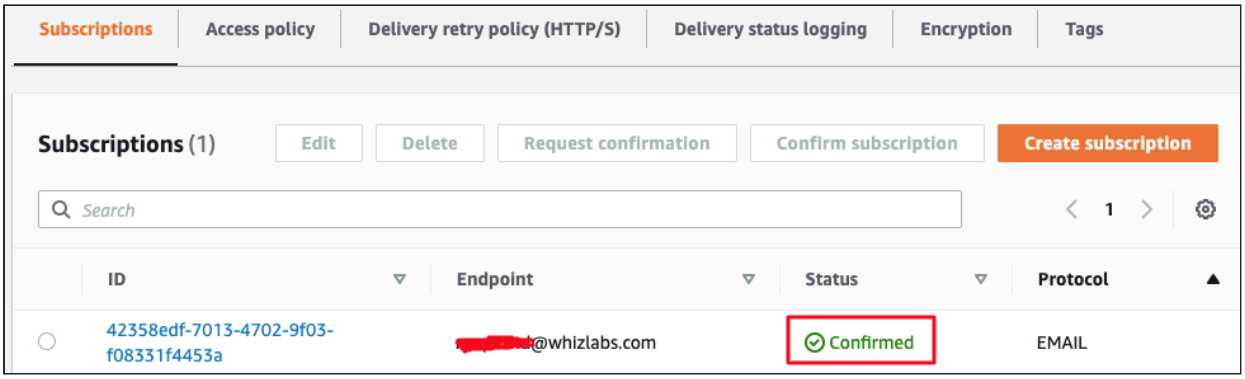


10. Click on **Confirm subscription**.



11. Your email address is now subscribed to the SNS Topic **whiz-s3-event-notification**.

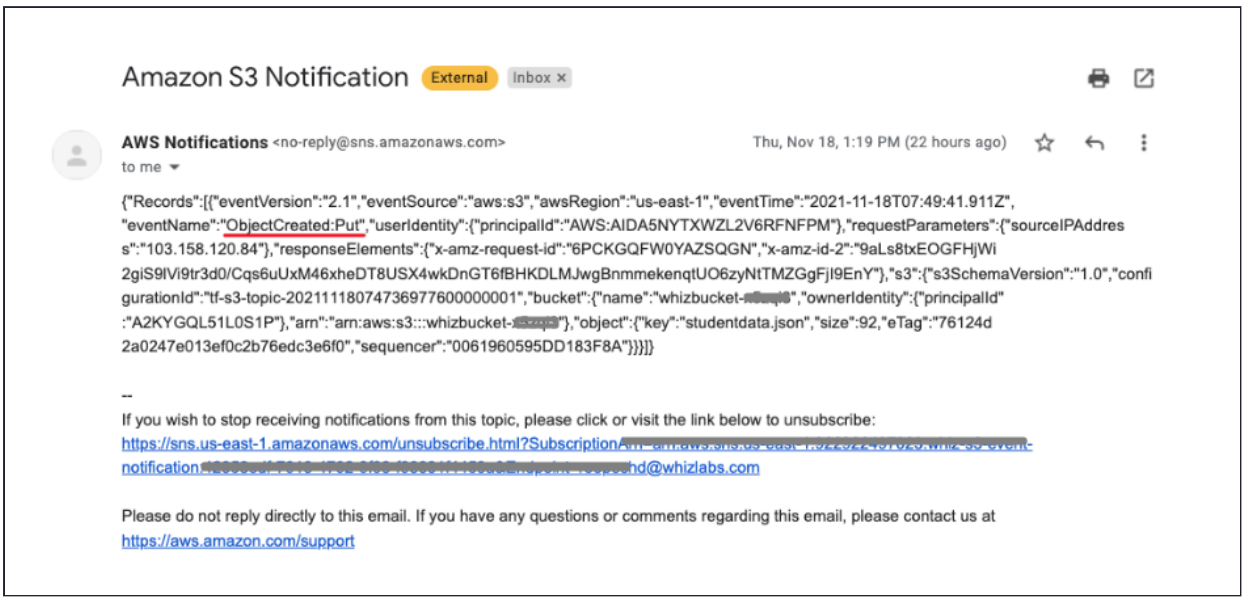
12. You can confirm it by navigating to the SNS topic's subscription tab. Refresh the page to see the status of your subscription as **Confirmed**.



Task 10: Upload an object into S3 bucket and test the SNS notification

1. Navigate to **S3** by clicking on **Services** on the top, then click on **S3** in the **Storage** section.
2. Click and open your S3 bucket **whizbucket-x5zqi3**.
3. In the bucket, under **Objects** click on **Upload**
4. Now click on **Add Files** and upload any file from your local system.
5. Once the file is successfully uploaded to the S3 bucket, click on **Close**. Now you can see the uploaded image under **Objects**.

6. Navigate to your Email and check the SNS notification. Keep refreshing as it may take some time to receive the notification.
7. You have successfully received a SNS notification based on the PUT object event in S3 bucket.

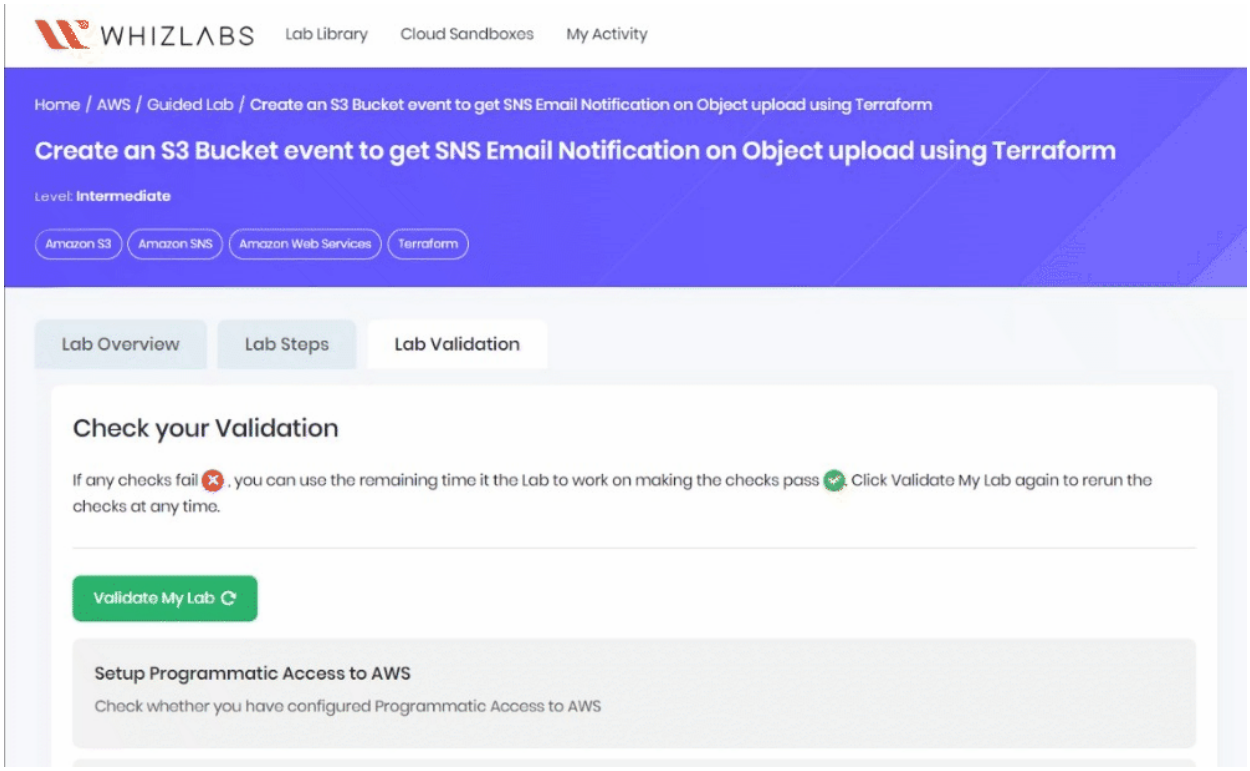


Do you know ?

You can customize the S3 Bucket event notifications to suit your needs. For instance, you can filter notifications based on the object's key or specific metadata, allowing you to target specific events and avoid unnecessary notifications.

Task 11: Validation of the lab

1. Once the lab steps are completed, please click on the **Validation** button on the left side panel.
2. This will validate the resources in the AWS account and displays whether you have completed this lab successfully or not.
3. Sample output :

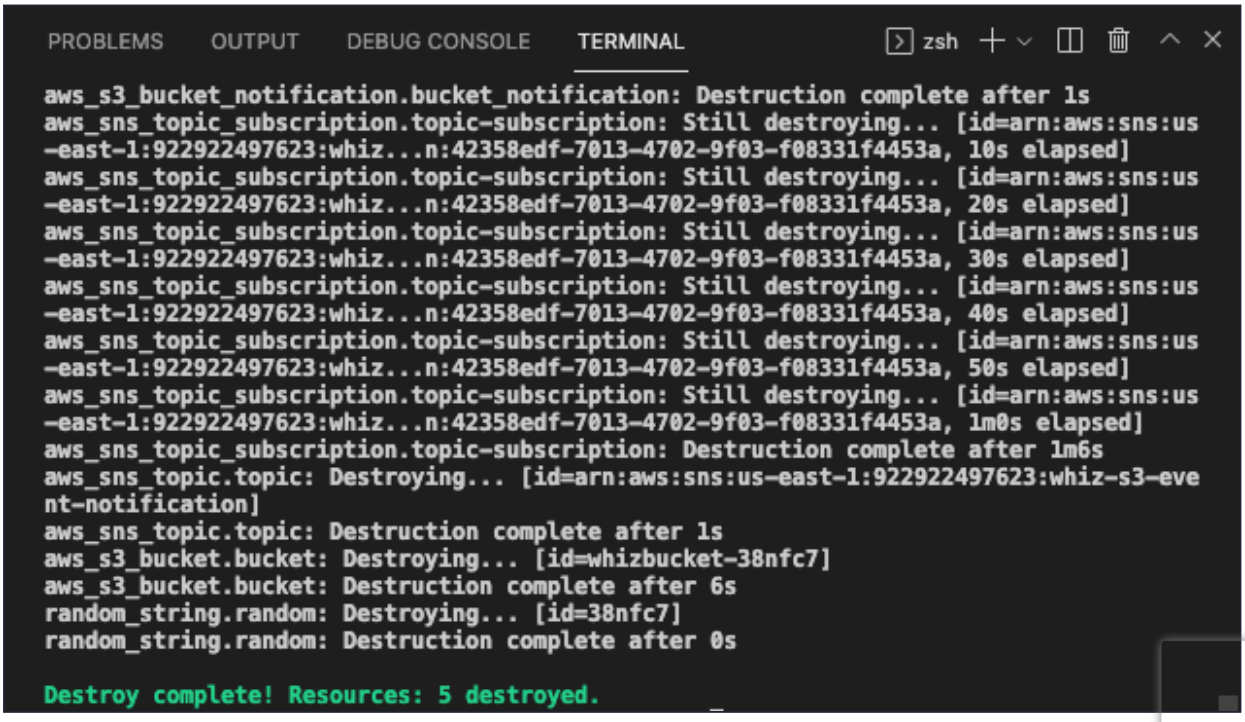


Task 12: Clean up AWS Resources

1. To delete the resources, open Terminal again.
2. Run the below command to delete all the resources.

```
terraform destroy
```

3. Again provide the endpoint variable details and press Enter.
4. Enter **yes** to confirm the deletion.



Completion and Conclusion

- You have set up the Visual Studio Code editor.
- You have created variables.tf and terraform.tfvars files.
- You have created a main.tf file.
- You have executed the terraform configuration commands to create the resources.
- You have checked all the resources created by opening the Console.
- You have uploaded an object in S3 bucket and tested the SNS notification.
- You have deleted all the resources.

End Lab

1. Sign out of AWS Account.
2. You have successfully completed the lab.
3. Once you have completed the steps, click on **End Lab** from your whizlabs dashboard.