

Operations Playbook

Name: Spurthy Mutturaj

Date: 29 Nov 2023

Class/Semester: IFT 562, 4th Semester

Contents

How to connect to the Mom & Pop Cafe Test EC2 instance	4
How to use the AWS CLI to connect to your AWS account	4
How to make a modification to the lab policy using the AWS CLI	4
How to add a parameter to the parameter store for allowing cookies on the website	4
How to connect to an EC2 instance to describe instances	5
How to launch an EC2 instance	5
How to fix a misconfigured web server with (_____) issue	6
How to change the AMI instance on the create-lamp-instance.sh script	6
How to tail a log in Linux	6
How to create an Auto Scaling Group in the AWS UI	7
How to create a Route 53 health check	9
How to create an Amazon RDS instance using the CLI	9
How to collect information about an instance	10
How to create two subnets in a subnet group via the AWS CLI	11
How to use the mysqldump tool to take a backup of a SQL database and restore it on another SQL instance	11
How to enable VPC Flow Logs via the command line interface	12
How to troubleshoot network connectivity on an instance	12
How to take a snapshot of an EBS volume	13
How to synchronize files using the command line (aws s3api and aws s3)	15
How to create a S3 bucket via the CLI	17
How to add an event notification to a S3 bucket	17
How to install the CloudWatch Agent	20
How to create a CloudWatch Events/CloudWatch EventBridge notification rule	20
How to use the prebuilt stopinator script to turn off instances with the tag value of your full name ...	21
How to resize an EC2 instance using the AWS CLI	23
How to detect drift in a CloudFormation template	25
How to create an Amazon Athena table	27
How to manually review access logs to find anomalous user activity	27
How to create a batch file to update the café website to change its colors	28
How to create a Lambda Layer and add it to a Lambda function	29
How to create a Lambda function from a prebuilt package	29

How to setup a VPC	30
How to add a bastion host (Linux) to the public subnet of a VPC to connect to instances in the private subnet	30
How to setup IAM so a user can assume an IAM role to access a resource	30
How to setup AWS Config to monitor resources.....	31
How to add inbound rules to both security groups and network ACLs	33
How to encrypt the root volume of an existing EC2 instance	35
How to create a SNS topic	36
How to subscribe to a SNS topic	37
How to create a CloudWatch alarm using a metrics-based filter	38

How to connect to the Mom & Pop Cafe Test EC2 instance

NOTE:I am using a MAC

1. Ensure you have a copy of pem file used to authenticate with your instance.
2. Open terminal and change permission on the pem file to read only.
3. Run the ssh command on the terminal to connect to the EC2 instance passing the following parameters and options
 - `ssh -i private_key.pem ec2-user@public_ip-address`

How to use the AWS CLI to connect to your AWS account

1. run the aws configure command
2. Each instance will have secret and access keys created at the set up of instance.
3. When prompted by the cli, enter the following details:
 - AWS Access Key ID
 - AWS Secret Access Key ID
 - Default region: us-east-1
 - Default O/P format: json

How to make a modification to the lab policy using the AWS CLI

1. Using the aws cli:
 - a. List the policies using the command `aws iam list-policies --scope local`
 - b. Using `get-policy` and the arn value from the previous step and the version-id pipe it to a json file:
`get-policy --policy-arn <arn value> --version-id <version id> > file_name.json`
get the lab_policy
2. Open the downloaded lab_policy in json format using a vi editor.
3. Make the necessary changes
4. Use the `aws iam create-policy-version` command to create a new version of the managed policy.
Replace <value> with your own values.
 - a. `aws iam create-policy-version \`
`--policy-arn <ar value> \`
`--policy-document <file:// the edited policy path file_name.json>`
`-- set-as-default`

In the last step the option - set as default, uses the new edited json file as the iam policy.

After running the command, you will receive a JSON response indicating the success of the operation.
Make sure to review it for any errors or issues.

How to add a parameter to the parameter store for allowing cookies on the website

1. Sign in to AWS Management Console:
 - a. Open your web browser and navigate to the AWS Management Console.
 - b. Sign in to your AWS account using your credentials.

2. Access Systems Manager Parameter Store:
 - a. In the AWS Management Console, navigate to the "Services" menu.
 - b. Under the "Management & Governance" section, select "Systems Manager."
3. Create a New Parameter:
 - a. In the Systems Manager dashboard, click on "Parameter Store" in the left navigation pane.
 - b. Click the "Create Parameter" button.
4. Configure Parameter:
 - a. Fill in the parameter details:
 - i. Name: /web.config/cookie_toggle.
 - ii. Description: This feature allows you to turn cookies on or off for the Cafe website.
 - iii. Value: "True"
5. We Enter Parameter Value:
 - a. In the "Parameter value" section, the value for allowing cookies, is True.
 - b. Optionally, you can leave the other options as default.
6. Create Parameter:
 - a. Click the "Create parameter" button to create the parameter.

How to connect to an EC2 instance to describe instances

1. Open your terminal or command prompt.
2. Use the ssh command to connect to your EC2 instance. Replace <your-instance-IP> with your EC2 instance's public IP address, and <your-key.pem> with the path to your private key file.
 - a. `ssh -i <your-key.pem> ec2-user@<your-instance-IP>`
3. Once connected to the EC2 instance, you can use the AWS CLI to describe instances. Run the following command to list all EC2 instances in your account:
 - a. `aws ec2 describe-instances`
4. You will receive a JSON response containing information about your EC2 instances.
5. To exit the SSH connection, simply type:
 - a. `exit`

These steps will allow you to connect to your EC2 instance and use the AWS CLI to describe instances

How to launch an EC2 instance

1. To launch an instance with specified details, first make a ssh connection to the ec2-instance using a key-pair .pem file on mac OS.
2. Use aws cli and specify the details as follows:

```
aws ec2 run-instances \  
  --image-id $AMI \ # Replace with the actual Amazon Linux 2 AMI ID  
  --instance-type t1.micro \  
  --key-name YourKeyName \      # Replace with your key pair name
```

`--security-group-ids sg-xyz\ # Replace with your security group ID`

`--subnet-id subnet-xyz # Replace with your subnet ID`

Note. : to get the Amazon Linux2 AMI ID, use the systems parameter store:

`AMI=$(aws ssm get-parameters --names /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2 --query 'Parameters[0].[Value]' --output text)`

The above steps will launch an EC2 Instance in the cloud

How to fix a misconfigured web server with (_____) issue

1. The issues is security-group's inbound rules.
2. The webserver security group doesn't allow ssh traffic , and hence when we try to connect using the terminal, it does not allow the connection.
3. To fix it , In the AWS UI-> s(ervices menu)EC2 ->instances, select the Misconfigured Web Server and in the security group tab , open the security groups and click on edit the inbound rules:
 - a. Add a rule
 - b. Set type-ssh
 - c. Source type-anywhere-IPv4
 - d. Click save rules
4. Make an ssh connection using the private_key.pem :
 - a. `Ssh -i private_key.pem ec2-user@<public ip address of Misconfigured Web Server >`

How to change the AMI instance on the create-lamp-instance.sh script

1. Establish SSH Connection: Begin by establishing an SSH connection with an EC2 instance. This initial step is crucial as it sets the stage for making changes to the AMI instance.
2. Configure AWS Environment: In the second stage, ensure your AWS environment is configured to meet the specific requirements of your task. This includes ensuring you have the necessary permissions and access to EC2 instances.
3. Backup and Open Script: Before making any changes, create a backup of the create-lamp-instance.sh script. Then, open this script in a text editor. Inside the script, take note of essential information such as VPC settings, Subnet ID, Security Group, and Instance ID.
4. Troubleshoot Errors: If you encounter any errors, especially if you see an error message like "InvalidAMIID.noound," carefully review the script. The error message provides a clue about the line where the issue originates. Identify and address any technical errors.
5. Update AMI ID: Within the script, you will find an existing AMI ID. Replace this AMI ID with the exact AMI ID of the instance you intend to use. This step is critical to ensure that you are specifying the correct AMI instance.
6. Rerun the Script: After making all the necessary adjustments and confirming that you have entered the correct AMI ID, rerun the create-lamp-instance.sh script. This action ensures that the AMI ID has been accurately updated and modified as intended.

How to tail a log in Linux

1. To tail a log, example-file.log located in a filepath:
Use command: `sudo tail -f filepath/ example-file.log`

The above will display the live output of the cloud-init process (example log below):
Cloud-init v. 21.2-3.el8 running 'modules:config' at Fri, 10 Sep 2023 15:24:11 +0000. Up 52.68 seconds.

Cloud-init v. 21.2-3.el8 running 'modules:final' at Fri, 10 Sep 2023 15:24:12 +0000. Up 53.44 seconds.

Cloud-init v. 21.2-3.el8 finished at Fri, 10 Sep 2023 15:24:12 +0000. Datasource DataSourceEc2. Up 53.58 seconds

2. Using the option `-n`, the command specifies the number of lines from the end of a file that should be displayed.
ex. `tail -n 10 example-file.log`
This will show you the last 10 lines of the log file.
3. `Cntl+c` is - sends an interrupt signal to the currently running process. This is typically used to stop or terminate the running process.
4. If you want to stop, type `":q"`

How to create an Auto Scaling Group in the AWS UI

1. Access the AWS Management Console:
 - a. Open a web browser and navigate to the AWS Management Console (<https://aws.amazon.com/>).
 - b. Sign in with your AWS account credentials.
2. Navigate to Auto Scaling Groups:
 - a. From the AWS Management Console, click on "Services" in the top-left corner.
 - b. Under the "Compute" section, select "Auto Scaling."
3. Create Auto Scaling Group:
 - a. In the Auto Scaling dashboard, scroll down and click on "Create Auto Scaling group."
4. Configure Launch Template and Network:
 - a. In the configuration step, provide the necessary details:
 - i. Auto Scaling group name: Enter a unique name.
 - ii. Choose the appropriate launch template or configuration.
 - iii. Click "Next" to proceed.
 - b. Set up network configurations:
 - i. Select the desired VPC.
 - ii. Choose the relevant subnets.
 - iii. Click "Next" to continue.
5. Load Balancing and Additional Settings:
 - a. Configure load balancing:
 - i. Select "Attach to an existing load balancer."
 - ii. Choose the target group for load balancing.
 - b. In the "Additional settings" section:
 - i. Enable group metrics collection within CloudWatch.

- ii. Click "Next."
6. Configure Group Size and Scaling Policies:
 - a. Set the group size:
 - i. Desired capacity: Enter 5 (Minimum: 5 nodes).
 - ii. Minimum capacity: Enter 5 (Minimum: 5 nodes).
 - iii. Maximum capacity: Enter 10 (Maximum: 10 nodes).
 - b. Configure scaling policies:
 - i. Select "Target tracking scaling policy."
 - ii. Scaling policy name: Enter a name (e.g., MyScalingPolicy).
 - iii. Metric type: Choose "Average CPU utilization."
 - iv. Target value: Enter 50 (Target: 7 nodes).
7. Add Tags:
 - a. On the "Add tags" page, click "Add tag" and configure:
 - i. Key: Enter a tag key (e.g., Name).
 - ii. Value: Enter a tag value (e.g., WebApp).
8. Review and Create:
 - a. Review the Auto Scaling group configuration details.
 - b. Once satisfied, click the "Create Auto Scaling group" button to create the group.

Your Auto Scaling group with step scaling, targeting a minimum of 5 nodes, a maximum of 10 nodes, and a target of 7 nodes, is now created and configured. It will automatically adjust the number of instances based on the defined CPU utilization target, ensuring efficient resource management.

To create a step scaling policy for scale out (increase capacity):

1. Open the Amazon EC2 console and navigate to "Auto Scaling Groups" from the navigation pane.
2. Select the checkbox next to your Auto Scaling group.
3. Verify and adjust the minimum and maximum size limits for your group if needed.
4. Go to the "Automatic scaling" tab and choose "Create dynamic scaling policy."
5. For the policy type, select "Step scaling."
6. Specify a name for the policy.
7. Choose an existing CloudWatch alarm or create a new one to monitor a metric like CPU utilization (set the alarm threshold to $\geq 80\%$).
8. Define the change in group size for this policy when executed (e.g., Add 30% of the group size).
9. Optionally, add more steps with specific scaling adjustments.
10. Choose "Create."

To create a step scaling policy for scale in (decrease capacity):

1. Continue from where you left off after creating the scale-out policy.
2. For the policy type, choose "Step scaling."
3. Specify a name for the policy.
4. Choose an existing CloudWatch alarm or create a new one to monitor the metric (e.g., CPU utilization $\leq 40\%$).
5. Define the change in group size for this policy when executed (e.g., Remove 2 capacity units).

6. Optionally, add more steps for scaling in.
7. Choose "Create."

How to create a Route 53 health check

1. Access Route 53:
 - a. Access the AWS Management Console.
 - b. From the Services menu, choose Route 53.
2. Create Health Check:
 - a. In the left navigation pane, click Health checks.
 - b. Click Create health check.
3. Configure Health Check:
 - a. Configure the following settings:
 - i. Name: Provide a name for the health check.
 - ii. What to monitor: Choose an appropriate option (e.g., Endpoint).
 - iii. Specify endpoint by: Select the method (e.g., IP address).
 - iv. IP address: Enter the target IP address.
 - v. Path: Specify the path to check (optional).
 - b. Expand Advanced configuration:
 - i. Configure settings like request interval and failure threshold according to your needs.
 - c. Click Next.
4. Create Alarm and Notification:
 - a. Configure the following:
 - i. Create alarm: Choose Yes.
 - ii. Send notification to: Select or create an SNS topic.
 - iii. Topic name: Provide a name for the SNS topic.
 - iv. Recipient email address: Enter an email address for notifications.
 - b. Click Create health check.

The Route 53 health check will now periodically monitor the specified endpoint and send notifications based on your configured settings. You can check the health check's status and monitoring information in the Route 53 console, and you'll receive email notifications if any issues are detected.

How to create an Amazon RDS instance using the CLI

1. Configure you AWS CLI
2. An example command to create an RDS instance:
 - `aws rds create-db-instance \`
 - `--db-instance-identifier YourDBInstanceName \`
 - `--db-instance-class db.t2.micro \`
 - `--engine mysql \`
 - `--allocated-storage 20 \`
 - `--master-username YourDBUsername \`

- `--master-user-password YourDBPassword \`
- `--vpc-security-group-ids YourSecurityGroupID \`
- `--availability-zone YourAvailabilityZone \`
- `--db-subnet-group-name YourSubnetGroupName`

[Modify the parameters as needed for your specific instance]

3. After running the `create-db-instance` command, AWS will initiate the creation of your RDS instance. You can monitor the status using this command
 - `aws rds describe-db-instances --db-instance-identifier YourDBInstanceName`
4. Once the RDS instance is available, you can access it using the endpoint provided. You can retrieve the endpoint with the following command:
 - `aws rds describe-db-instances \`
 - `--db-instance-identifier YourDBInstanceName \`
 - `--query "DBInstances[0].Endpoint.Address" \`
 - `--output text`
5. Connect and Use Your RDS Instance: Now that you have created your RDS instance, you can connect to it using a MySQL client or any compatible database tool and start using it for your applications. Be sure to use the master username and password you specified during instance creation.

How to collect information about an instance

1. To collect information about an instance in AWS using the AWS CLI, you can use the following commands
 - a. This command will return the instance ID, instance type, public DNS name, public IP address, availability zone, VPC ID, and security group IDs for the specified instance.
Replace "YourInstanceName" with the name of your instance.
 - `aws ec2 describe-instances \`
 - `--filters "Name=tag:Name,Values=YourInstanceName" \`
 - `--query "Reservations[*].Instances[*].[InstanceId,InstanceType,PublicDnsName,PublicIpAddress,Placement.AvailabilityZone,VpcId,SecurityGroups[*].GroupId]"`
 - b. This command will return the IPv4 CIDR block of the VPC associated with the instance.
 - *# Replace "YourVPCID" with the VPC ID obtained in Step 1.*
 - `aws ec2 describe-vpcs --vpc-ids YourVPCID \`
 - `--query "Vpcs[*].CidrBlock"`
 - c. This command will return the subnet ID and IPv4 CIDR block of the subnet(s) associated with the instance's VPC.
 - a. *# Replace "YourVPCID" with the VPC ID obtained in Step 1.*
 - b. `aws ec2 describe-subnets \`
 - c. `--filters "Name=vpc-id,Values=YourVPCID" \`

d. `--query "Subnets[*].[SubnetId,CidrBlock]"`

d. This command will return a list of Availability Zones in the specified region.

- *# Replace "<region>" with your AWS region (e.g., us-east-1 or eu-west-2).*
- `aws ec2 describe-availability-zones \`
- `--filters "Name=region-name,Values=<region>" \`
- `--query "AvailabilityZones[*].ZoneName"`

Note: Make sure to replace the placeholders ("*<region>*", "*YourInstanceName*", "*YourVPCID*") with the actual values and run these commands in your AWS CLI to collect the instance information.

How to create two subnets in a subnet group via the AWS CLI

1. Create Private Subnet 1

- `aws ec2 create-subnet \`
- `--vpc-id <YourVPCID> \`
- `--cidr-block <CIDR block for Private Subnet 1> \`
- `--availability-zone <Availability Zone for Private Subnet 1>`

2. Create Private Subnet 2

- `aws ec2 create-subnet \`
- `--vpc-id <YourVPCID> \`
- `--cidr-block <CIDR block for Private Subnet 2> \`
- `--availability-zone <Availability Zone for Private Subnet 2>`

3. Create RDS Subnet Group

- `aws rds create-db-subnet-group \`
- `--db-subnet-group-name MyDBSubnetGroup \`
- `--db-subnet-group-description "Subnet group for my RDS instance" \`
- `--subnet-ids <SubnetId for Private Subnet 1> <SubnetId for Private Subnet 2>`

How to use the mysqldump tool to take a backup of a SQL database and restore it on another SQL instance

1. Backup the SQL Database:

- Open an SSH session to the source SQL instance.
- Use the ``mysqldump`` utility to create a backup of the source database.
- `mysqldump --user=<username> --password='<password>' --databases <database_name> > backup.sql`

2. Open an SSH session to the target SQL instance.

Use the ``mysql`` command to restore the backup to the target database.

```
mysql --user=<username> --password='<password>' --host=<target_host> < backup.sql
```

Replace ``<username>``, ``<password>``, and ``<database_name>`` with your MySQL credentials and the database you want to backup.

3. Verify that the database on the target instance was successfully created and populated by running SQL queries or examining the data as needed.

How to enable VPC Flow Logs via the command line interface

1. Ensure you are logged into the AWS CLI with the appropriate IAM user or credentials.
2. Create an S3 Bucket for Flow Logs
 - a. `aws s3api create-bucket --bucket flowlog#### --region <region> --create-bucket-configuration LocationConstraint=<region>`
Replace `<region>` and `<flowlog####>` with your desired AWS region and a unique bucket name.
3. Retrieve VPC ID
 - a. `vpc_id=$(aws ec2 describe-vpcs --query 'Vpcs[?Tags[?Key==`Name` && Value==`VPC1`]].VpcId' --output text)`
This command retrieves the VPC ID for a VPC named "VPC1." Modify the name as needed.
4. Enable Flow Logs
 - a. `aws ec2 create-flow-logs --resource-type VPC --resource-ids $vpc_id --traffic-type ALL --log-destination-type s3 --log-destination arn:aws:s3:::<flowlog####>`
Replace `<flowlog####>` with a unique name and ARN for your flow logs.
5. After executing the above command, the flow logs should be created. You can verify their status by running:
 - a. `aws ec2 describe-flow-logs`
 - b. You can analyze and review the flow logs data, which is stored in the S3 bucket you created earlier, using AWS S3 tools or log analysis tools as needed.

How to troubleshoot network connectivity on an instance

1. Checking Network Configuration and Security Groups:
 - a. Examine the network configuration of the VPC, including route tables and subnet associations.
 - b. To describe route tables in the VPC:
 - c. `aws ec2 describe-route-tables`
 - d. To describe security groups associated with instances in the VPC:
 - e. `aws ec2 describe-security-groups`
 - f. Analyze the route tables and security groups to identify any issues related to routing and network access restrictions.
2. Using Network ACLs (Access Control Lists):
 - a. Check Network ACLs to ensure they are not blocking required traffic.
 - b. To inspect Network ACL entries:
 - c. `aws ec2 describe-network-acls`
 - d. If necessary, remove problematic Network ACL entries using the `delete-network-acl-entry` command.
 - e. To delete a specific Network ACL entry:

- f. `aws ec2 delete-network-acl-entry --network-acl-id <network-acl-id> --rule-number <rule-number>`
- g. Be aware that Network ACLs are a critical layer of security in VPCs and can be a common source of problems.

3. VPC Flow Logs Analysis:

- a. Enable VPC Flow Logs to capture information about network traffic.
- b. Download and extract VPC Flow Logs from an S3 bucket.
- c. To create an S3 bucket for flow logs (if not already created):
 - i. `aws s3api create-bucket --bucket flowlog#### --region <region> --create-bucket-configuration LocationConstraint=<region>`
- d. To download and extract flow logs:
 - i. `mkdir flowlogs`
 - ii. `cd flowlogs`
 - iii. `aws s3 cp s3://<flowlog####>/ . --recursive`
 - iv. `gunzip *.gz`
- e. Use tools like `grep` to search for specific patterns in the logs, e.g., failed SSH connection attempts.
- f. To search for specific patterns in flow logs (e.g., failed SSH connections):
 - i. `grep -rn ' 22 ' . | grep REJECT`
- g. Analyze the flow logs to gain insights into traffic patterns and diagnose security and connectivity issues.

4. SSH and Connectivity Testing:

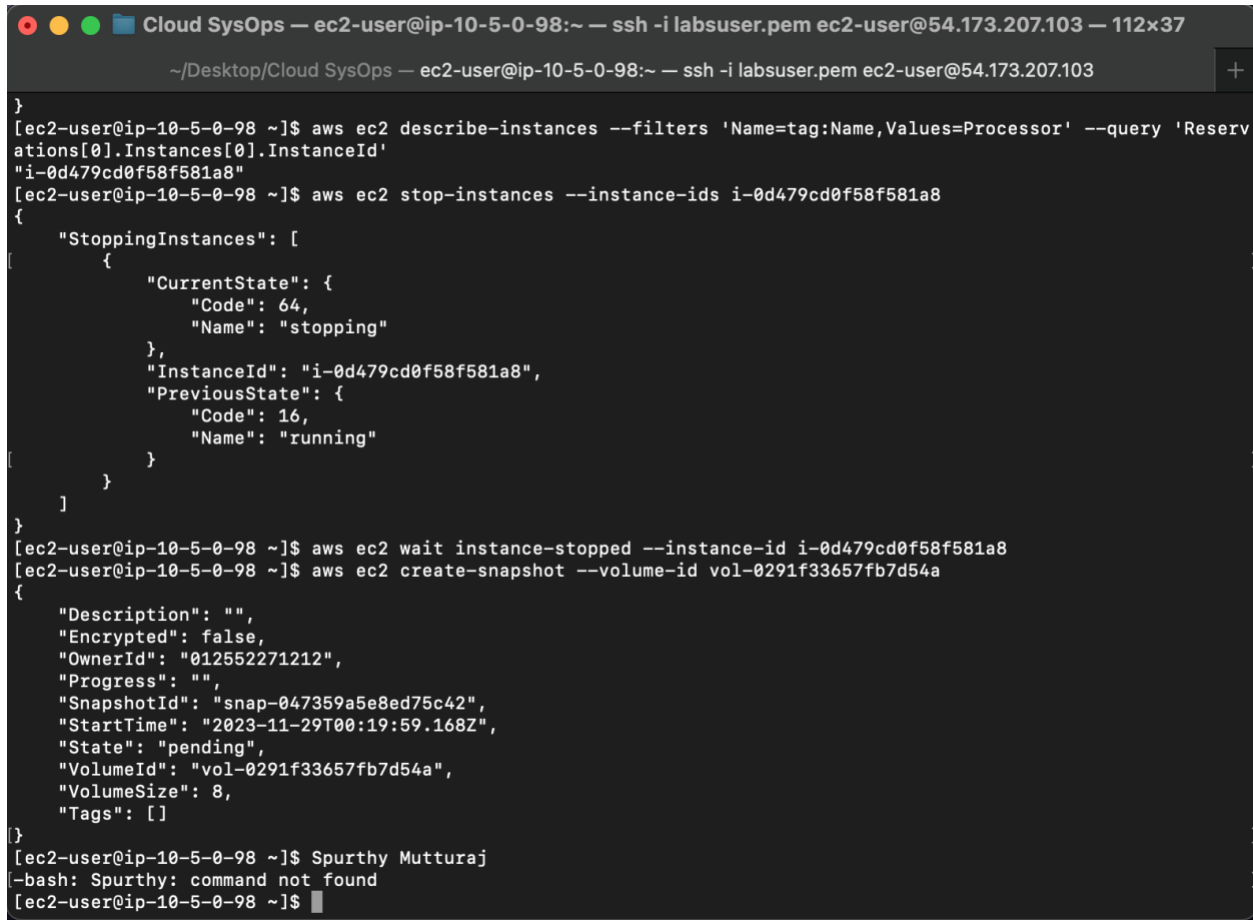
- a. Test SSH connectivity to EC2 instances within the VPC.
- b. For SSH connection testing, follow the specific steps for connecting to instances using SSH. `Ssh -i privatekey.pem ec2-user@<Ipaddress>`
- c. Use SSH to connect to instances and diagnose any connectivity issues.
- d. By testing SSH connectivity, determine if there are issues with access to specific instances, helping to narrow down the problem.

How to take a snapshot of an EBS volume

1. Find the volume ID of the EBS volume attached to your instance:
 - a. `aws ec2 describe-instances --filter 'Name=tag:Name,Values=Processor'`
Note the `VolumelId` value in the response as it will be referred to as `VOLUME-ID` in subsequent commands.
2. Shut down the instance to ensure a consistent snapshot:
 - a. `aws ec2 stop-instances --instance-ids INSTANCE-ID`
3. Create a snapshot of the EBS volume:
 - a. `aws ec2 create-snapshot --volume-id VOLUME-ID`

Note the `SnapshotId` value in the response as it will be referred to as `SNAPSHOT-ID` in subsequent commands.
4. Check the snapshot status to ensure it's completed:

- a. `aws ec2 wait snapshot-completed --snapshot-id SNAPSHOT-ID`
5. Restart the instance:
 - a. `aws ec2 start-instances --instance-ids INSTANCE-ID`
6. Verify that the instance is running again:
 - a. `aws ec2 wait instance-running --instance-id INSTANCE-ID`



```
Cloud SysOps — ec2-user@ip-10-5-0-98:~ — ssh -i labsuser.pem ec2-user@54.173.207.103 — 112x37
~/Desktop/Cloud SysOps — ec2-user@ip-10-5-0-98:~ — ssh -i labsuser.pem ec2-user@54.173.207.103 +
}
[ec2-user@ip-10-5-0-98 ~]$ aws ec2 describe-instances --filters 'Name=tag:Name,Values=Processor' --query 'Reservations[0].Instances[0].InstanceId'
"i-0d479cd0f58f581a8"
[ec2-user@ip-10-5-0-98 ~]$ aws ec2 stop-instances --instance-ids i-0d479cd0f58f581a8
{
  "StoppingInstances": [
    {
      "CurrentState": {
        "Code": 64,
        "Name": "stopping"
      },
      "InstanceId": "i-0d479cd0f58f581a8",
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
[ec2-user@ip-10-5-0-98 ~]$ aws ec2 wait instance-stopped --instance-id i-0d479cd0f58f581a8
[ec2-user@ip-10-5-0-98 ~]$ aws ec2 create-snapshot --volume-id vol-0291f33657fb7d54a
{
  "Description": "",
  "Encrypted": false,
  "OwnerId": "012552271212",
  "Progress": "",
  "SnapshotId": "snap-047359a5e8ed75c42",
  "StartTime": "2023-11-29T00:19:59.168Z",
  "State": "pending",
  "VolumeId": "vol-0291f33657fb7d54a",
  "VolumeSize": 8,
  "Tags": []
}
[ec2-user@ip-10-5-0-98 ~]$ Spurthy Mutturaj
[-bash: Spurthy: command not found
[ec2-user@ip-10-5-0-98 ~]$
```

Fig: Taking a snapshot of EBS Volume

```
Cloud SysOps — ec2-user@ip-10-5-0-98:~ — ssh -i labsuser.pem ec2-user@54.173.207.103 — 112x37
~/Desktop/Cloud SysOps — ec2-user@ip-10-5-0-98:~ — ssh -i labsuser.pem ec2-user@54.173.207.103 +
}
[ec2-user@ip-10-5-0-98 ~]$ aws ec2 wait instance-stopped --instance-id i-0d479cd0f58f581a8
[ec2-user@ip-10-5-0-98 ~]$ aws ec2 create-snapshot --volume-id vol-0291f33657fb7d54a
{
  "Description": "",
  "Encrypted": false,
  "OwnerId": "012552271212",
  "Progress": "",
  "SnapshotId": "snap-047359a5e8ed75c42",
  "StartTime": "2023-11-29T00:19:59.168Z",
  "State": "pending",
  "VolumeId": "vol-0291f33657fb7d54a",
  "VolumeSize": 8,
  "Tags": []
}
[ec2-user@ip-10-5-0-98 ~]$ Spurthy Mutturaj
-bash: Spurthy: command not found
[ec2-user@ip-10-5-0-98 ~]$ aws ec2 wait snapshot-completed --snapshot-id snap-047359a5e8ed75c42
[ec2-user@ip-10-5-0-98 ~]$ aws ec2 start-instances --instance-ids i-0d479cd0f58f581a8
{
  "StartingInstances": [
    {
      "CurrentState": {
        "Code": 0,
        "Name": "pending"
      },
      "InstanceId": "i-0d479cd0f58f581a8",
      "PreviousState": {
        "Code": 80,
        "Name": "stopped"
      }
    }
  ]
}
[ec2-user@ip-10-5-0-98 ~]$ Spurthy Mutturaj
-bash: Spurthy: command not found
[ec2-user@ip-10-5-0-98 ~]$
```

Fig: Starting the instance after taking a snapshot

How to synchronize files using the command line (aws s3api and aws s3)

1. Enable Versioning on Your Bucket:
2. Replace <YOUR-BUCKET-NAME> with your actual bucket name and run the following command to enable versioning on your S3 bucket:
aws s3api put-bucket-versioning --bucket <YOUR-BUCKET-NAME> --versioning-configuration Status=Enabled
3. Synchronize Local Files with S3 Bucket:
This command syncs the contents of your local folder with the S3 bucket:
aws s3 sync <LOCAL-FOLDER-PATH> s3://<YOUR-BUCKET-NAME>/<REMOTE-FOLDER-PATH>/
4. List Files in S3 Bucket:
To confirm the state of your files in the S3 bucket, use this command:
aws s3 ls s3://<YOUR-BUCKET-NAME>/<REMOTE-FOLDER-PATH>/
5. Delete a Local File:
rm <LOCAL-FILE>
6. Delete the Corresponding Remote File:
To delete the same file from the S3 bucket, use the --delete option with the aws s3 sync command:

- aws s3 sync <LOCAL-FOLDER-PATH> s3://<YOUR-BUCKET-NAME>/<REMOTE-FOLDER-PATH>/ --delete
7. Verify Remote File Deletion: Confirm that the file was deleted from the S3 bucket:
aws s3 ls s3://<YOUR-BUCKET-NAME>/<REMOTE-FOLDER-PATH>/
 8. Recover an Old Version of a File:
aws s3api list-object-versions --bucket <YOUR-BUCKET-NAME> --prefix <REMOTE-FILE-PATH>
 9. Restore an Older Version of a File:
aws s3api get-object --bucket <YOUR-BUCKET-NAME> --key <REMOTE-FILE-PATH> --version-id <VERSION-ID> <LOCAL-FILE>
 10. Verify Restored File Locally:
To confirm that the file has been restored locally, use the following command:
 - a. ls <LOCAL-FOLDER-PATH>
 11. Re-Sync Local Files with S3:
To sync the contents of the local folder back to Amazon S3 after restoring an older version, use this command:
aws s3 sync <LOCAL-FOLDER-PATH> s3://<YOUR-BUCKET-NAME>/<REMOTE-FOLDER-PATH>/
 12. Verify the New Version in S3:
Finally, confirm that the new version of the file has been pushed to Amazon S3:
aws s3 ls s3://<YOUR-BUCKET-NAME>/<REMOTE-FOLDER-PATH>/

```

inflating: files/file2.txt
inflating: files/file3.txt
[ec2-user@ip-10-5-0-98 ~]$ aws s3 sync files s3://buck914/files/
upload: files/file3.txt to s3://buck914/files/file3.txt
upload: files/file2.txt to s3://buck914/files/file2.txt
upload: files/file1.txt to s3://buck914/files/file1.txt
[ec2-user@ip-10-5-0-98 ~]$ aws s3 ls s3://buck914/files/
2023-11-29 00:28:02    38318 file1.txt
2023-11-29 00:28:02    43784 file2.txt
2023-11-29 00:28:02    96675 file3.txt
[ec2-user@ip-10-5-0-98 ~]$ rm files/file1.txt
[ec2-user@ip-10-5-0-98 ~]$ aws s3 sync files s3://buck914/files/ --delete
delete: s3://buck914/files/file1.txt
[ec2-user@ip-10-5-0-98 ~]$ aws s3 ls s3://buck914/files/
2023-11-29 00:28:02    43784 file2.txt
2023-11-29 00:28:02    96675 file3.txt
[ec2-user@ip-10-5-0-98 ~]$ aws s3api list-object-versions --bucket buck914 --prefix files/file1.txt
{
  "Versions": [
    {
      "ETag": "\"b76b2b775023e60e16bc332496f84091\"",
      "Size": 38318,
      "StorageClass": "STANDARD",
      "Key": "files/file1.txt",
      "VersionId": "SLIo_OABglqwSycPiCA_Ak9kY0PbaZ",
      "IsLatest": false,
      "LastModified": "2023-11-29T00:28:02.000Z",
      "Owner": {
        "DisplayName": "aws1absc0w6282171t1694542650",
        "ID": "fed1b4d2b1e84daf4bb3954f36f5c61f6859d0a8503f38b89ed9c79529c2d33f"
      }
    },
    {
      "Owner": {
        "DisplayName": "aws1absc0w6282171t1694542650",
        "ID": "fed1b4d2b1e84daf4bb3954f36f5c61f6859d0a8503f38b89ed9c79529c2d33f"
      },
      "DeleteMarkers": [
        {
          "Owner": {
            "DisplayName": "aws1absc0w6282171t1694542650",
            "ID": "fed1b4d2b1e84daf4bb3954f36f5c61f6859d0a8503f38b89ed9c79529c2d33f"
          },
          "Key": "files/file1.txt",
          "VersionId": "NRx6fS8UWKG_Le3SbIoU8eIMo12kf7ma",
          "IsLatest": true,
          "LastModified": "2023-11-29T00:28:48.000Z"
        }
      ],
      "RequestCharged": null
    }
  ]
}
[ec2-user@ip-10-5-0-98 ~]$ aws s3api get-object --bucket buck914 --key files/file1.txt --version-id SLIo_OABglqwSycPiCA_Ak9kY0PbaZ files/file1.txt
{
  "AcceptRanges": "bytes",
  "LastModified": "Wed, 29 Nov 2023 00:28:02 GMT",
  "ContentLength": 38318,
  "ETag": "\"b76b2b775023e60e16bc332496f84091\"",
  "VersionId": "SLIo_OABglqwSycPiCA_Ak9kY0PbaZ",
  "ContentType": "text/plain",
  "ServerSideEncryption": "AES256",
  "Metadata": {}
}
[ec2-user@ip-10-5-0-98 ~]$ Spurthy Mutturaj
-bash: Spurthy: command not found
[ec2-user@ip-10-5-0-98 ~]$

```

Fig: Syncing files in S3 bucket

How to create a S3 bucket via the CLI

1. Open a terminal or command prompt with the AWS CLI installed and configured.
2. Run the AWS CLI Command: Replace <mompopcafe-xxxxnnn> with your desired bucket name, <region> with your chosen AWS region, <xxx> with your initials, and <nnn> with a random number. Execute this command:
 - a. `aws s3 mb s3://mompopcafe-xxxxnnn --region <region>`
3. Verify the Bucket:
 - a. `aws s3 ls`

```
connection to 64.173.287.103 closed.
```

```
(base) spurthy@Spurthys-MBP Cloud SysOps % chmod 400 labuser.pem
```

```
(base) spurthy@Spurthys-MBP Cloud SysOps % ssh -i labuser.pem ec2-user@64.210.21.76
```

```
The authenticity of host "64.210.21.76 ([64.210.21.76])" can't be established.
```

```
ED25519 key fingerprint is SHA256:IjNBFYtZdMdxwxiBDRPzIWKtoGdpJWkuARfvmBgNg.
```

```
This key is not known by any other names
```

```
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

```
Warning: Permanently added "64.210.21.76" (ED25519) to the list of known hosts.
```

```
f#  
#####  
~\ #####  
~~ \ ####/  
~~~~ \ ##/  
~~~~ ~/\>  
~~~~ V'-' i-> A newer version of Amazon Linux is available!  
  
~~~~ ~-+ / |  
~~~~ ~-+ / | Amazon Linux 2023, GA and supported until 2028-03-15.  
~~~~ ~-+ / | https://aws.amazon.com/linux/amazon-linux-2023/  
~~~~ ~-+ / |  
~~~~ ~-+ / |
```

```
[ec2-user@ip-10-200-0-247 ~]$ curl http://169.254.169.254/latest/dynamic/instance-identity/document | grep region  
% Total    % Received % Xferd Average Speed   Time    Time     Current  
                                 Dload Upload   Total   Spent    Left     Speed  
  
100 476 100 476 0      0 69337 0 --:--:-- --:--:-- --:--:-- 79333  
  
"region": "us-east-1",  
[ec2-user@ip-10-200-0-247 ~]$ aws configure  
AWS Access Key ID [None]: AKIAISHXQAKA5P7HANNSS  
AWS Secret Access Key [None]: lhwPYQdaV8VBRetrfjf4ZqVRQNmqfojo7shHYFWnS  
Default region name [None]: us-east-1  
Default output format [None]: json  
[ec2-user@ip-10-200-0-247 ~]$ aws s3 mb s3://mompopcafe9714 --region us-east-1  
make_bucket: mompopcafe9714  
[ec2-user@ip-10-200-0-247 ~]$ Spurthy Mutturaj  
-bash: Spurthy: command not found  
[ec2-user@ip-10-200-0-247 ~]$ █
```

Fig: using mb command to create a bucket

How to add an event notification to a S3 bucket

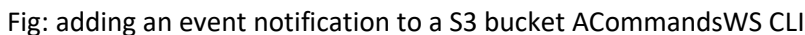
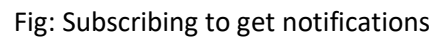
1. Create a JSON configuration file for the event notification, e.g., notification-config.json. Open the file for editing:
 - a. `vi notification-config.json`
 - b. Enter insert mode by pressing `i`.
2. Customize the JSON Configuration:
3. Replace `<ARN of S3NotificationTopic>` with your actual S3 notification topic's ARN and adjust event types and filter rules according to your requirements. Example configuration:

```
{
  "TopicConfigurations": [
    {
      "TopicArn": "arn:aws:sns:::S3NotificationTopic",
      "Events": ["s3:ObjectCreated:*", "s3:ObjectRemoved:*"],
      "Filter": {
        "Key": {
          "FilterRules": [
```

```
{
  "Name": "prefix",
  "Value": "your-prefix/"
}
]
}
}
}
]
}
```

Exit insert mode with ESC, then save and exit the editor with :wq.

4. Associate Configuration with S3 Bucket:
5. Use this AWS CLI command, replacing <YourBucketName> with the name of your S3 bucket:
 - a. `aws s3api put-bucket-notification-configuration --bucket <YourBucketName> --notification-configuration file://notification-config.json`
6. Wait briefly for the event notification setup to complete.
7. Check your email inbox for a message with the subject "Amazon S3 Notification." Open the email and examine the notification message, which contains details about the triggered event in your S3 bucket, including event type, timestamp, bucket name, request ID, and more.



How to install the CloudWatch Agent

1. Open AWS Management Console and navigate to Systems Manager.
2. Access "Run Command" within Systems Manager.
3. Install CloudWatch Agent on the target instance using AWS-ConfigureAWSPackage with the action set to "Install." Giving it a name and a version description.
4. Choose the instances you want to use as targets.
5. This configuration will install Cloudwatch Agent on your server
6. Store CloudWatch Agent configuration in Parameter Store.
7. Create a parameter for log and metric configurations.
8. Initiate the CloudWatch Agent on the Web Server using Run Command, configuring the action as "configure" and referencing the parameter.
9. Check CloudWatch Logs and Metrics for application logs and system performance.
10. Establish metric filters for specific log patterns (e.g., errors) and set up alarms.
11. Utilize CloudWatch Metrics for analyzing CPU, disk, and network usage.
12. Configure CloudWatch Events for real-time notifications and activate AWS Config Rules to ensure compliance, especially regarding tagging and EBS Volumes.

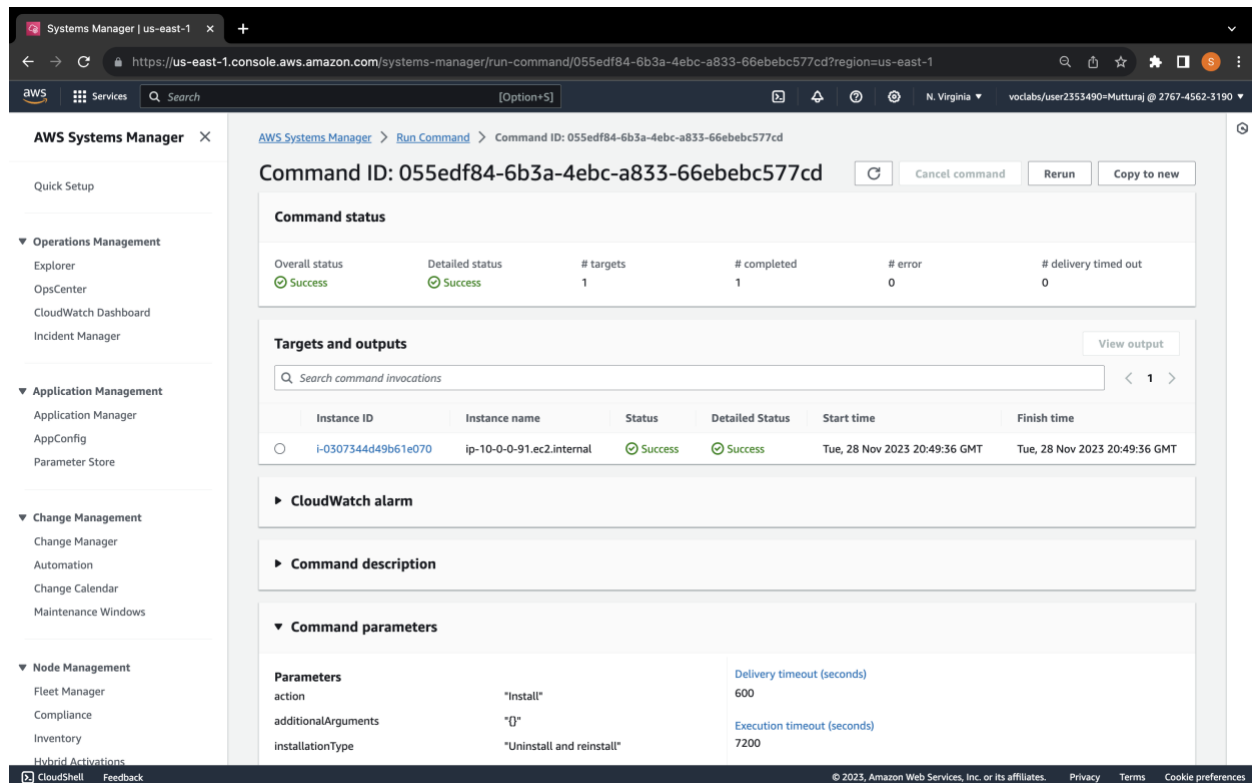


Fig: installing a CloudWatch Agent

How to create a CloudWatch Events/CloudWatch EventBridge notification rule

1. Navigate to AWS Management Console and access CloudWatch Events.
2. Create a new rule, providing a relevant name (e.g., Instance_Stopped_Terminated).
3. Configure the event pattern by selecting AWS services as the source, EC2 as the service, and EC2 Instance State-change Notification as the event type.

- Specify specific states, such as "stopped" and "terminated."
- In the Target section, choose SNS topic and select the desired topic for notifications.
- Optionally, add tags and review the configuration.
- Click "Create rule" to establish the CloudWatch Events rule.
- To enable SMS notifications, navigate to Simple Notification Service (SNS), select Topics, and verify the subscription associated with your email. Note that SMS functionality may require an AWS Support Center request.

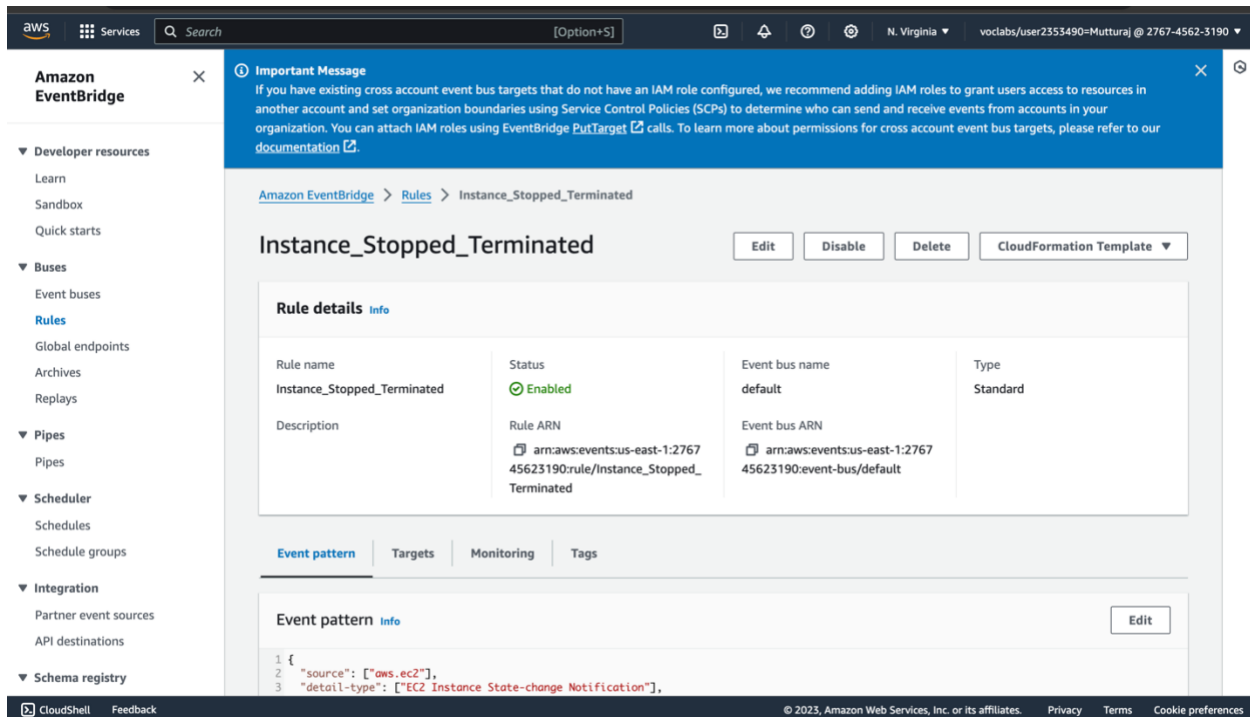


Fig: Creating a CloudWatch Events/CloudWatch EventBridge notification rule

How to use the prebuilt stopinator script to turn off instances with the tag value of your full name.

- Examine the Stopinator Script:
 - Access the Command Host Instance.
 - Navigate to the aws-tools directory: `cd aws-tools`.
 - Open the stopinator.php script: `nano stopinator.php`.
 - Examine the script contents, noting the format of tags and available arguments.
- Stop Instances with Your Full Name Tag:
 - Run the stopinator.php script to stop instances: `./stopinator.php -t"Name=YourFullName"`
- Verify the script output indicates the instances to be stopped.
 - In the EC2 Management Console, confirm that the instances are stopping or stopped.
- Restart the Instances:
- From the Command Host SSH session, restart instances:
 - `./stopinator.php -t"Name=YourFullName" -s`

- b. Verify in the EC2 Management Console that the previously stopped instances are now restarting.

Note: Adjust the tag value in the commands to match your full name. These steps showcase using the stopinator.php script to manage instances based on specified tags.

```
No instances to stop in Array.
Region is us-east-1
Identified instance i-0da93037f3b27004b
Identified instance i-0f60967c272c8489c
PHP Notice: Array to string conversion in /home/ec2-user/aws-tools/stopinator.php on line 115

Stopping identified instances in Array...
Region is us-east-2
PHP Notice: Array to string conversion in /home/ec2-user/aws-tools/stopinator.php on line 120

No instances to stop in Array.
Region is us-west-1
PHP Notice: Array to string conversion in /home/ec2-user/aws-tools/stopinator.php on line 120

No instances to stop in Array.
Region is us-west-2
PHP Notice: Array to string conversion in /home/ec2-user/aws-tools/stopinator.php on line 120

No instances to stop in Array.
[[ec2-user@ip-10-5-0-116 aws-tools]$ Spurthy Mutturaj]
-bash: Spurthy: command not found
[[ec2-user@ip-10-5-0-116 aws-tools]$
```

Fig: Using the prebuilt stopinator script

The screenshot shows the AWS Management Console interface. On the left, there's a navigation menu with options like EC2 Dashboard, EC2 Global View, Events, and various instance types. The main area displays 'Instances (2)' with a table listing two instances: 'web server' and 'app server'. Both are in a 'Stopped' state. Below this, the details for 'Instance: i-040136976ddb96c8 (Command Host)' are expanded, showing it is in a 'Running' state. The details include Instance ID, IP addresses, DNS names, and VPC ID. A warning message at the bottom indicates that the user is not authorized to perform certain actions due to an identity-based policy.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
web server	i-0da93037f3b27004b	Stopped	t2.micro	-	No alarms	us-east-1a	-
app server	i-0f60967c272c8489c	Stopped	t2.micro	-	No alarms	us-east-1a	-

Instance: i-040136976ddb96c8 (Command Host)

Details	Security	Networking	Storage	Status checks	Monitoring	Tags
Instance summary Instance ID: i-040136976ddb96c8 (Command Host) IPv6 address: - Hostname type: IP name: ip-10-5-0-116.ec2.internal Answer private resource DNS name: - Auto-assigned IP address: 54.234.176.35 [Public IP]	Public IPv4 address: 54.234.176.35 [open address] Instance state: Running Private IP DNS name (IPv4 only): ip-10-5-0-116.ec2.internal Instance type: t2.medium VPC ID: vpc-028826c3b04eb6e3 [Lab VPC]	Private IPv4 addresses: 10.5.0.116 Public IPv4 DNS: ec2-54-234-176-35.compute-1.amazonaws.com [open address] Elastic IP addresses: - AWS Compute Optimizer finding: User: arn:aws:sts::635513349516:assumed-role/voclabs/user2353490=Mutturaj is not authorized to perform: compute-optimizer:GetEnrollmentStatus on resource: * because no identity-based policy all				

Fig: Instances are turned off.

```

    No instances to start in Array
Region is ap-southeast-2
PHP Notice:  Array to string conversion in /home/ec2-user/aws-tools/stopinator.php on line 110

    No instances to start in Array
Region is eu-central-1
PHP Notice:  Array to string conversion in /home/ec2-user/aws-tools/stopinator.php on line 110

    No instances to start in Array
Region is us-east-1
    Identified instance i-0da93037f3b27004b
    Identified instance i-0f60967c272c8489c
PHP Notice:  Array to string conversion in /home/ec2-user/aws-tools/stopinator.php on line 105

    Starting identified instances in Array...
Region is us-east-2
PHP Notice:  Array to string conversion in /home/ec2-user/aws-tools/stopinator.php on line 110

    No instances to start in Array
Region is us-west-1
PHP Notice:  Array to string conversion in /home/ec2-user/aws-tools/stopinator.php on line 110

    No instances to start in Array
Region is us-west-2
PHP Notice:  Array to string conversion in /home/ec2-user/aws-tools/stopinator.php on line 110

    No instances to start in Array
[ec2-user@ip-10-5-0-116 aws-tools]$ Spurthy Mutturaj
-bash: Spurthy: command not found
[ec2-user@ip-10-5-0-116 aws-tools]$ █

```

Fig: Restarting the stopped instances

How to resize an EC2 instance using the AWS CLI

1. Identify Instance ID:
 - a. `aws ec2 describe-instances \`
`--filters "Name=tag:Name,Values=YourInstanceName" \`
`--query "Reservations[*].Instances[*].InstanceId"`
 Replace YourInstanceName with the name of your EC2 instance.
2. Stop the Instance:
 - a. `aws ec2 stop-instances --instance-ids <Your_Instance_ID>`
 Replace <Your_Instance_ID> with the actual instance ID.
3. Resize the Instance:
 - a. `aws ec2 modify-instance-attribute \`
`--instance-id <Your_Instance_ID> \`
`--instance-type "{\"Value\": \"t2.micro\"}"`
 Replace <Your_Instance_ID> with the actual instance ID.
4. Start the Resized Instance:
 - a. `aws ec2 start-instances --instance-ids <Your_Instance_ID>`
 Replace <Your_Instance_ID> with the actual instance ID.
5. Verify Resizing:
 - a. `aws ec2 describe-instances \`
`--instance-ids <Your_Instance_ID> \`
`--query "Reservations[*].Instances[*].[InstanceType,State.Name]"`
 Replace <Your_Instance_ID> with the actual instance ID.
6. Repeat the command until the status shows as "running."

Note: Ensure you replace YourInstanceName and <Your_Instance_ID> with your specific EC2 instance name and ID respectively.

```
Cloud SysOps — ec2-user@cli-host:~ — ssh -i labsuser.pem ec2-user@100.25.248.159 — 112x37
...cli-host:~ — ssh -i labsuser.pem ec2-user@100.25.248.159 ~ — -zsh ... +

    "i-01703d23160582c94"
  ]
]
[ec2-user@cli-host ~]$ aws ec2 stop-instances --instance-ids i-01703d23160582c94
Note: AWS CLI version 2, the latest major version of the AWS CLI, is now stable and recommended for general use.
For more information, see the AWS CLI version 2 installation instructions at: https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2.html

usage: aws [options] <command> [<subcommand> ...] [parameters]
To see help text, you can run:

    aws help
    aws <command> help
    aws <command> <subcommand> help
aws: error: argument --instance-ids is required
[ec2-user@cli-host ~]$ aws ec2 stop-instances --instance-ids i-01703d23160582c94
{
  "StoppingInstances": [
    {
      "InstanceId": "i-01703d23160582c94",
      "CurrentState": {
        "Code": 64,
        "Name": "stopping"
      },
      "PreviousState": {
        "Code": 16,
        "Name": "running"
      }
    }
  ]
}
[ec2-user@cli-host ~]$ aws ec2 modify-instance-attribute \
> --instance-id i-01703d23160582c94 \
> --instance-type "{\"Value\": \"t2.micro\"}"
[ec2-user@cli-host ~]$ Spurthy Mutturaj
-bash: Spurthy: command not found
[ec2-user@cli-host ~]$
```

Fig: Resize an EC2 instance using the AWS CLI


```
Cloud SysOps — ec2-user@cli-host:~ — ssh -i labsuser.pem ec2-user@100.25.248.159 — 112x37
...cli-host:~ — ssh -i labsuser.pem ec2-user@100.25.248.159 ~ — -zsh
    "CurrentState": {
      "Code": 64,
      "Name": "stopping"
    },
    "PreviousState": {
      "Code": 16,
      "Name": "running"
    }
  }
}
]
}
[ec2-user@cli-host ~]$ aws ec2 modify-instance-attribute \
> --instance-id i-01703d23160582c94 \
> --instance-type "{\"Value\": \"t2.micro\"}"
[ec2-user@cli-host ~]$ Spurthy Mutturaj
-bash: Spurthy: command not found
[ec2-user@cli-host ~]$ ws ec2 start-instances --instance-ids i-01703d23160582c94
-bash: ws: command not found
[ec2-user@cli-host ~]$ aaws ec2 start-instances --instance-ids i-01703d23160582c94
-bash: aaws: command not found
[ec2-user@cli-host ~]$ aws ec2 start-instances --instance-ids i-01703d23160582c94
{
  "StartingInstances": [
    {
      "InstanceId": "i-01703d23160582c94",
      "CurrentState": {
        "Code": 0,
        "Name": "pending"
      },
      "PreviousState": {
        "Code": 80,
        "Name": "stopped"
      }
    }
  ]
}
[ec2-user@cli-host ~]$
```

Fig: Restarting the instance after modifying it.

How to detect drift in a CloudFormation template

1. Open the AWS Management Console in a browser tab and Navigate to CloudFormation
2. From the Services menu, choose "CloudFormation."
3. Select Stack for Drift Detection:
4. Choose the specific CloudFormation stack you want to check for drift.
5. Initiate Drift Detection:
6. Run the command in the AWS CLI or SDK to initiate drift detection, obtaining a StackDriftDetectionId:
 - a. `aws cloudformation detect-stack-drift --stack-name yourStackName`
7. Monitor Drift Detection Status:
 - a. Check the drift detection status using the following command, replacing <driftId> with the actual value of StackDriftDetectionId:
`aws cloudformation describe-stack-drift-detection-status --stack-drift-detection-id <driftId>`
8. Run the command to get detailed information on drifted resources:
 - a. `aws cloudformation describe-stack-resource-drifts --stack-name yourStackName`
9. Refine Output for Readability:

10. Use a query parameter with the command to filter results and output only essential information:
 - a. `aws cloudformation describe-stack-resources \`
`--stack-name yourStackName \`
`--query`
`'StackResources[*].[ResourceType,ResourceStatus,DriftInformation.StackResourceDriftStatus]'` \
 - `--output table`
11. Use the following command with specific filters to retrieve detailed information on modified resources:
 - a. `aws cloudformation describe-stack-resource-drifts \`
`--stack-name yourStackName \`
`--stack-resource-drift-status-filters MODIFIED`
12. Address drift issues by manually updating the CloudFormation stack using the command. Resolve any errors indicated during the update process:
 - a. `aws cloudformation update-stack \`
`--stack-name yourStackName \`
`--template-body file://yourUpdatedTemplate.yaml \`
`--parameters ParameterKey=YourParameterKey,ParameterValue=YourParameterValue`

Note: Replace placeholders such as `yourStackName`, `yourUpdatedTemplate.yaml`, `YourParameterKey`, and `YourParameterValue` with the actual values relevant to your CloudFormation setup.

```

Cloud SysOps — ec2-user@cli-host:~ — ssh -i labsuser.pem ec2-user@34.207.226.241 — 159x46
~/Desktop/Cloud SysOps — ec2-user@cli-host:~ — ssh -i labsuser.pem ec2-user@34.207.226.241

AWS::EC2::SubnetRouteTableAssociation | CREATE_COMPLETE | NOT_CHECKED
AWS::EC2::Subnet                      | CREATE_COMPLETE | IN_SYNC
AWS::EC2::VPCGatewayAttachment       | CREATE_COMPLETE | NOT_CHECKED
AWS::CloudFormation::WaitCondition   | CREATE_COMPLETE | NOT_CHECKED
AWS::CloudFormation::WaitConditionHandle | CREATE_COMPLETE | NOT_CHECKED
AWS::EC2::SecurityGroup              | CREATE_COMPLETE | MODIFIED
AWS::EC2::Instance                   | CREATE_COMPLETE | IN_SYNC

[ec2-user@cli-host ~]$ aws cloudformation describe-stack-resource-drifts \
> --stack-name myStack \
> --stack-resource-drift-status-filters MODIFIED
{
  "StackResourceDrifts": [
    {
      "StackId": "arn:aws:cloudformation:us-east-1:755316856011:stack/myStack/10d6a590-6547-11ee-bbd1-0ee9ed0db7f3",
      "ActualProperties": "{\n  \"GroupDescription\": \"Enable access to web server\",\n  \"GroupName\": \"WebServerSG\",\n  \"SecurityGroupIngress\": [{\n    \"CidrIp\": \"75.237.42.89/32\",\n    \"FromPort\": 22,\n    \"IpProtocol\": \"tcp\",\n    \"ToPort\": 22\n  }],\n  \"CidrIp\": \"0.0.0.0/0\",\n  \"FromPort\": 80,\n  \"IpProtocol\": \"tcp\",\n  \"ToPort\": 80\n  }],\n  \"Tags\": [{\n    \"Key\": \"Name\",\n    \"Value\": \"WebServerSG\"\n  }],\n  \"VpcId\": \"vpc-03079ca87b29d39b1\"\n}",
      "ResourceType": "AWS::EC2::SecurityGroup",
      "Timestamp": "2023-10-07T19:39:49.618Z",
      "PhysicalResourceId": "sg-01adc1df88a4b65f",
      "StackResourceDriftStatus": "MODIFIED",
      "ExpectedProperties": "{\n  \"GroupDescription\": \"Enable access to web server\",\n  \"GroupName\": \"WebServerSG\",\n  \"SecurityGroupIngress\": [{\n    \"CidrIp\": \"0.0.0.0/0\",\n    \"FromPort\": 22,\n    \"IpProtocol\": \"tcp\",\n    \"ToPort\": 22\n  }],\n  \"CidrIp\": \"0.0.0.0/0\",\n  \"FromPort\": 80,\n  \"IpProtocol\": \"tcp\",\n  \"ToPort\": 80\n  }],\n  \"Tags\": [{\n    \"Key\": \"Name\",\n    \"Value\": \"WebServerSG\"\n  }],\n  \"VpcId\": \"vpc-03079ca87b29d39b1\"\n}",
      "PropertyDifferences": [
        {
          "PropertyPath": "/SecurityGroupIngress/0/CidrIp",
          "ActualValue": "75.237.42.89/32",
          "ExpectedValue": "0.0.0.0/0",
          "DifferenceType": "NOT_EQUAL"
        }
      ]
    },
    {
      "LogicalResourceId": "WebSecurityGroup"
    }
  ]
}

[ec2-user@cli-host ~]$ aws cloudformation update-stack \
> --stack-name myStack \
> --template-body file://template1.yaml \
> --parameters ParameterKey=KeyName,ParameterValue=vockey

An error occurred (ValidationError) when calling the UpdateStack operation: No updates are to be performed.
[ec2-user@cli-host ~]$ Spurthy Mutturaj
-bash: Spurthy: command not found
[ec2-user@cli-host ~]$

```

Fig: Detecting drift in a CloudFormation template

How to create an Amazon Athena table

1. Access CloudTrail in AWS Console:
 - a. Use the AWS Management Console and search for "CloudTrail" in the services.
 - b. Navigate to the Event history page within CloudTrail.
2. Initiate Athena Table Creation:
 - a. From CloudTrail Event history, select "Create Athena table."
3. Specify Storage Location:
 - a. Choose the Amazon S3 bucket storing CloudTrail log files.
4. Review and Confirm Schema:
 - a. Analyze the generated Athena CREATE TABLE statement for proper schema.
 - b. Confirm the LOCATION statement for data storage on Amazon S3.
5. Complete Table Creation:
 - a. Select "Create table" to finalize the Athena table creation.

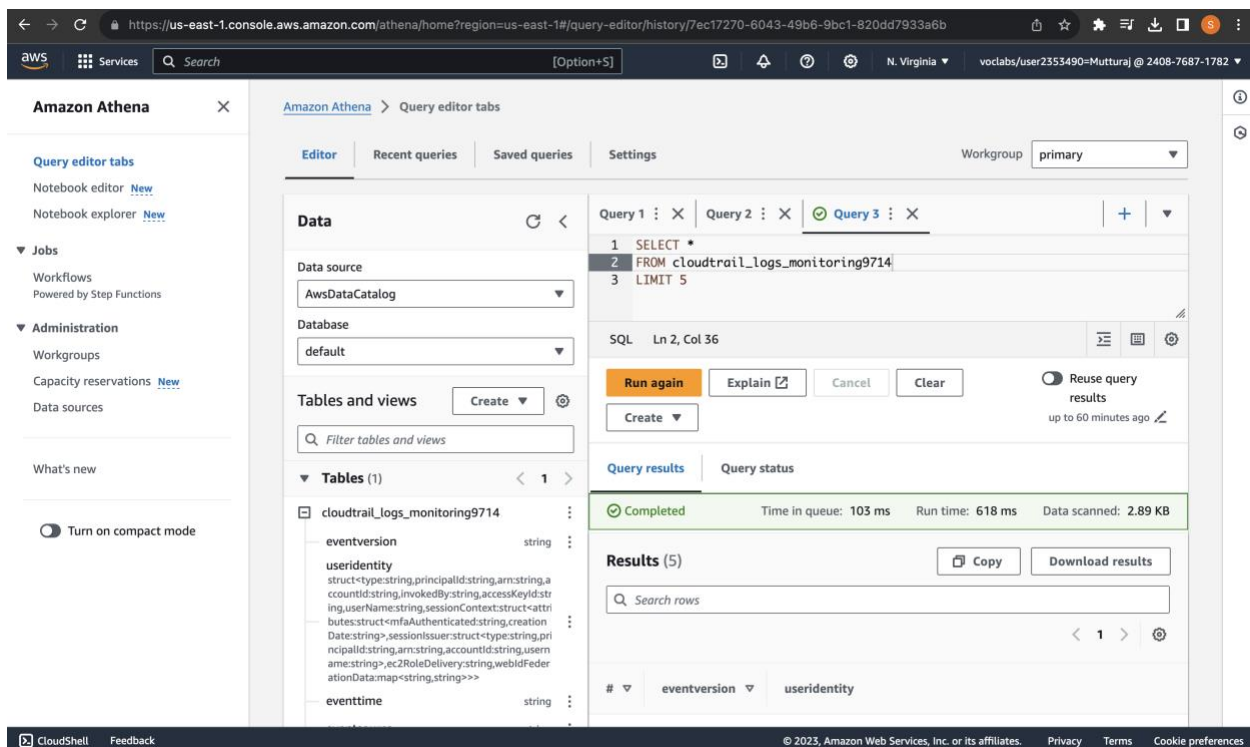


Fig: Creating an Amazon Athena table

How to manually review access logs to find anomalous user activity

1. Log in to the relevant system or service that provides access logs.
2. Navigate to Access Logs: Locate the section or interface that contains access logs for the system.
3. Experiment with Queries: Run various queries to explore log entries. Use the SQL-like querying capabilities, if available.
4. Examine Last Query Results: Analyze the data returned by the last executed query. Look for patterns or anomalies.
5. If applicable, filter the logs based on the service involved. For example, use WHERE clauses like WHERE eventsources = 'ec2.amazonaws.com' to focus on EC2-related events.

6. Refine Queries: Use WHERE clauses to filter for security-related events, such as WHERE eventname LIKE '%Security%'. Experiment with compound clauses for precise matches.
7. Analyze Suspicious Event Names: Once security-related actions are isolated, examine event names for suspicious activities. Adjust WHERE clauses to search for specific event names.
8. Identify User and Access Details:
9. Craft queries to extract information about the AWS user responsible, the exact time of the security breach, the IP address used, and the method employed (console or programmatic access).
10. Verify Results: Cross-reference and verify the identified information to ensure accuracy and completeness.

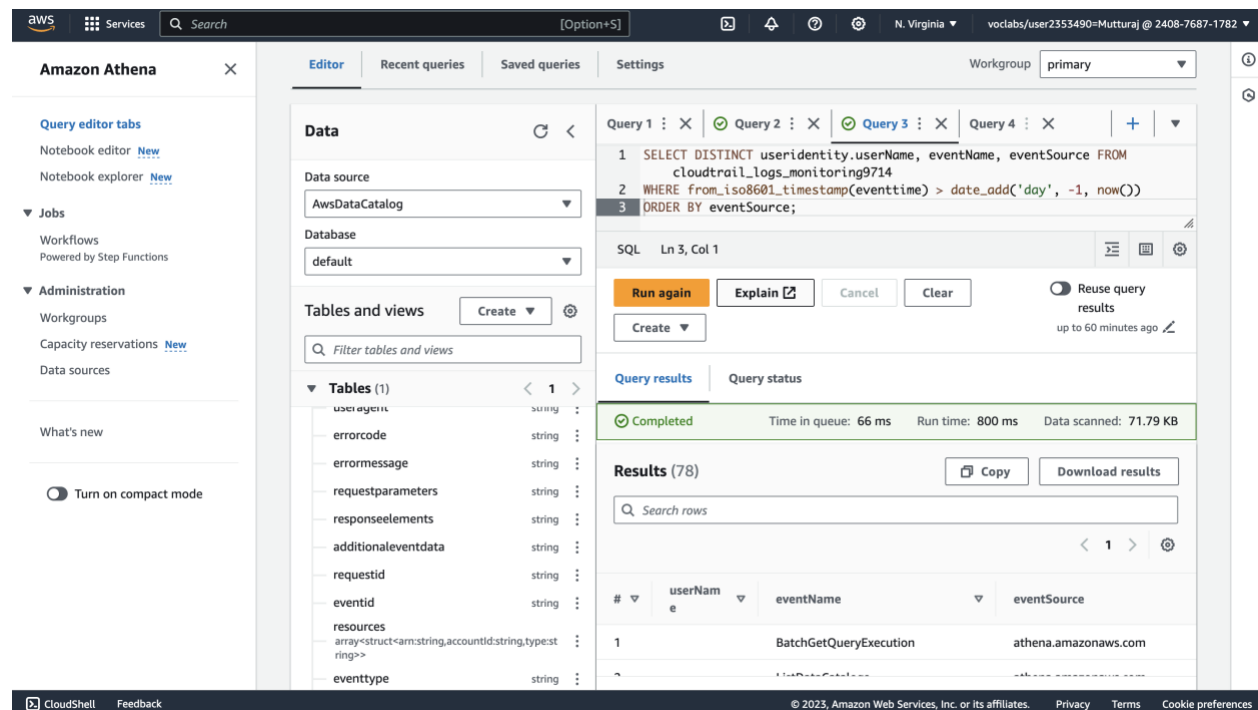


Fig: Manually querying the logs using Athena table

How to create a batch file to update the café website to change its colors

1. Create an empty file - filename.sh , using the touch command and open it in a vi editor.
2. Write a script with the following commands:
 - `#!/bin/bash - she-bang`
 - `aws s3 cp ~/sysops-activity-files/ s3://<my-bucket>/ --recursive --acl public-read`
 Note: replace <my-bucket> with the name of your bucket.
3. Make the file executable : `chmod +x filename.sh`
4. Open the index file of the website in a text editor
5. Locate and modify the color attributes , ex set bgcolor=Cyan , bgcolor=Coral, bgcolor=Brown.
 And save the file
6. Run the batch file using the command: `./filename.sh`

How to create a Lambda Layer and add it to a Lambda function

1. Create a Lambda Layer
 - a. Go to the AWS Management Console and select Services > Lambda.
 - b. Choose Layers.
 - c. Click Create layer.
 - d. Configure the layer settings:
 - e. Name: [LayerName]
 - f. Description: [LayerDescription]
 - g. Code entry type: Upload a .zip file
 - h. Choose Upload and select the [LayerZipFileName].zip file you downloaded.
 - i. Compatible runtimes: Python 3.7 (or your preferred runtime).
 - j. Click Create.
2. Create the Lambda Function

Configure the function as follows:

 - a. Author from scratch: Selected
 - b. Function name: [FunctionName]
 - c. Runtime: Python 3.7 (or your preferred runtime)
 - d. Expand Change default execution role and select "Use an existing role."
 - e. Choose the existing role: [ExistingRoleName]
 - f.
 - g. Click Create function.
3. Add the Lambda Layer to the Function
 - a. In the Function overview panel, under [FunctionName], choose Layers.
 - b. Click Add a layer.
 - c. In the Add layer page, configure as follows:
 - d. Choose a layer: Select the Custom layers card.
 - e. Custom layers: [LayerName]
 - f. Version: [LayerVersion]
 - g. Click Add.

How to create a Lambda function from a prebuilt package

1. Access AWS Lambda Console: Log in to your AWS Management Console and navigate to the AWS Lambda service.
2. Create a New Lambda Function:
 - a. Click on "Create function."
 - b. Choose "Author from scratch."
 - c. Fill in the function name, runtime (e.g., Python, Node.js), and existing role or create a new one.
3. Configure Function
 - a. In the "Function code" section, select "Upload a .zip file."
 - b. Upload a prebuilt deployment package that contains your function code.
 - c. Define the handler (entry point) for your function.
4. Create Function:

- a. Click the "Create function" button.

How to setup a VPC

1. Log into the AWS Management Console and in the services menu select vpc.
2. Click on create vpc:
 - a. Add a name tag of what you want name your vpc
 - b. Set a CIDR block range.
 - c. Click on create vpc

How to add a bastion host (Linux) to the public subnet of a VPC to connect to instances in the private subnet

1. In the AWS Console choose EC2 and click on launch Instance and configure the following:
 - a. Set a name to your bastion server.
 - b. Set your OS and application images:
 - i. Select Amazon Linux from Quick Start
 - c. Select your instance type.
 - d. Select if you want to use a key pair to login.
 - e. Select the VPC where you want to launch the instance.
 - f. Select the public subnet.
 - g. Select the appropriate security group.
 - h. Review the summary and launch the instance.

How to setup IAM so a user can assume an IAM role to access a resource

1. Create IAM Role:
 - a. `aws iam create-role --role-name MyRole --assume-role-policy-document file:///trust-policy.json`
2. Define Role Permissions:
 - a. `aws iam attach-role-policy --role-name MyRole --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess`
3. Grant AssumeRole Permission:
4. Trust Policy Example (trust-policy.json):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      }
    }
  ],
}
```

```

    "Action": "sts:AssumeRole"
  }
}
}

```

5. IAM User Configuration:
 - a. `aws iam attach-user-policy --user-name MyUser --policy-arn arn:aws:iam::aws:policy/SecurityAudit`
6. Assume Role that you created and want in AWS console.

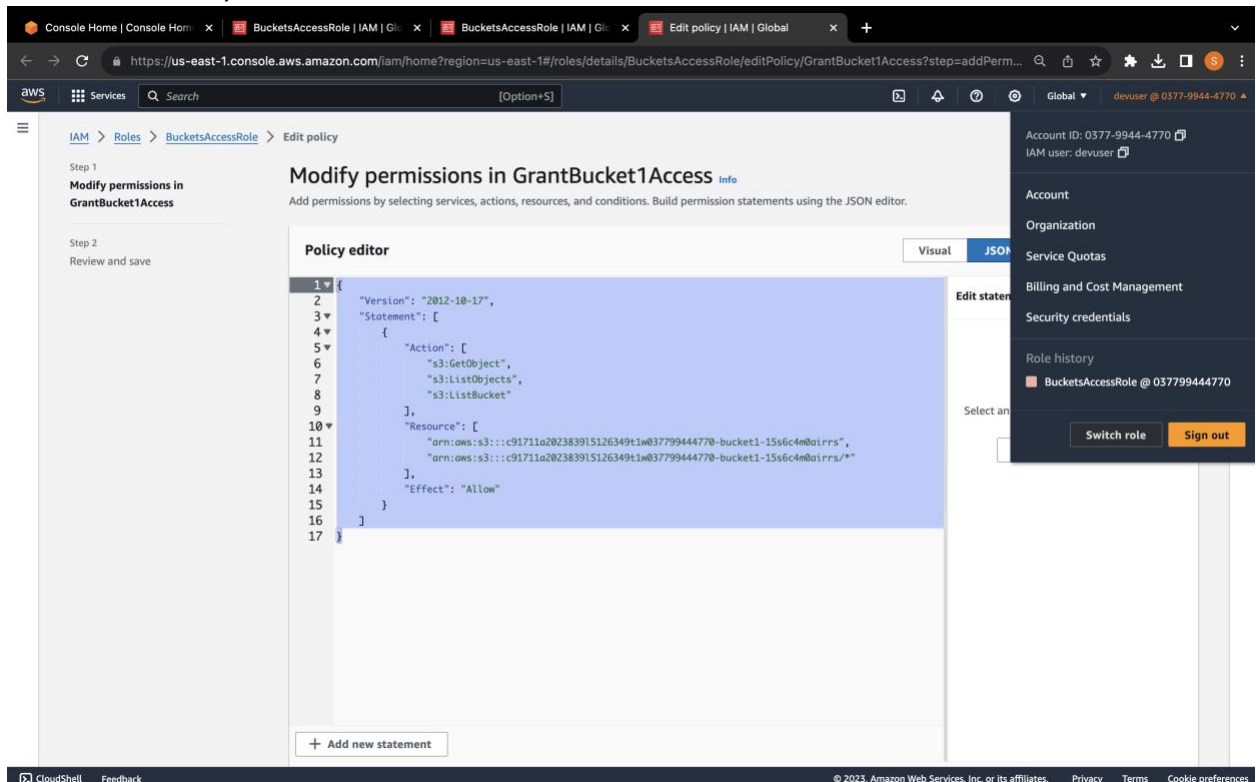


Fig: Different IAM roles.

How to setup AWS Config to monitor resources

1. Navigate to AWS Config: In the AWS Management Console, go to AWS Config and choose "Get started."
2. Configure Resource Monitoring:
 - a. Select "Record specific resource types."
 - b. Choose the resource category (e.g., AWS resources) and type (e.g., AWS EC2 SecurityGroup).
3. Choose AWS Config Role: Select an existing AWS Config role (e.g., AwsConfigRole).
4. Configure Delivery Method:
 - a. Keep the default settings for storing findings in an S3 bucket.
 - b. Choose "Next" to proceed.

5. Review Managed Rules:
6. Skip managed rules configuration by choosing "Next."
7. Review setup details and choose "Confirm" to initiate the configuration.
8. Observe Resource Inventory: Access the AWS Config Dashboard and navigate to "Resources."

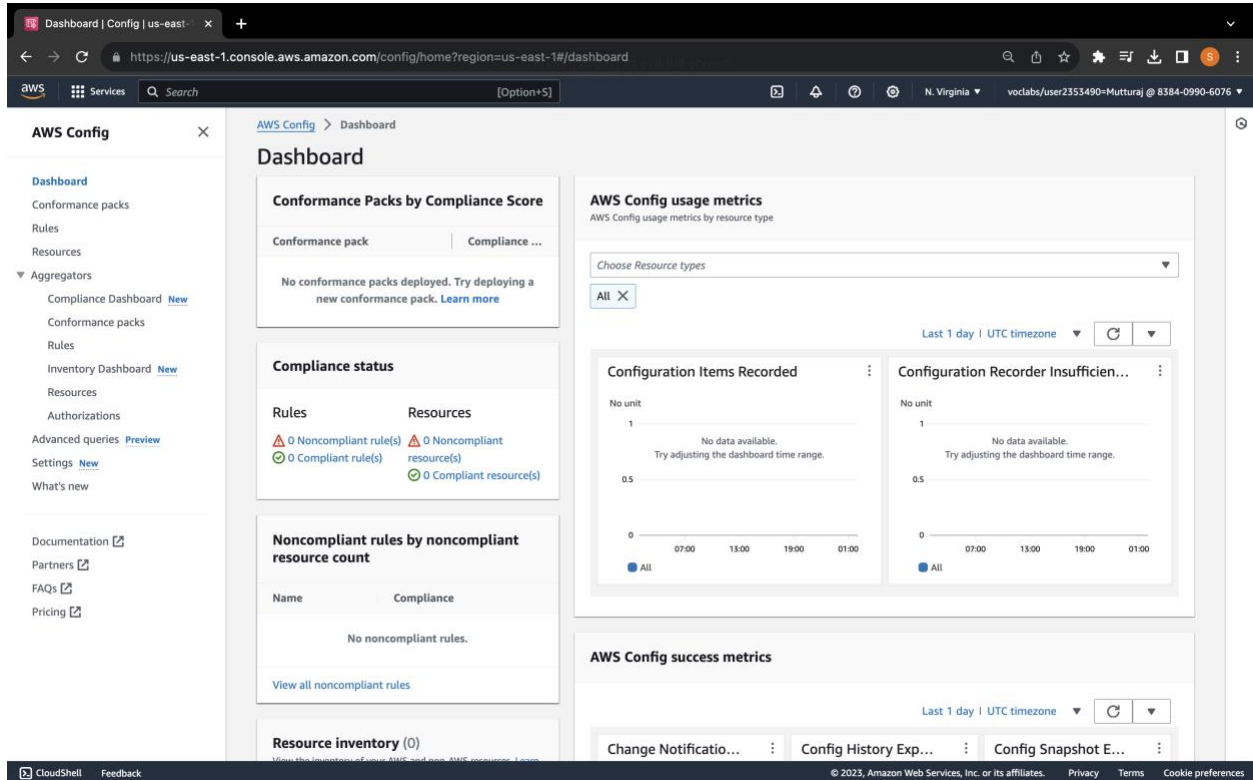


Fig: Setting up AWS Config to monitor resources

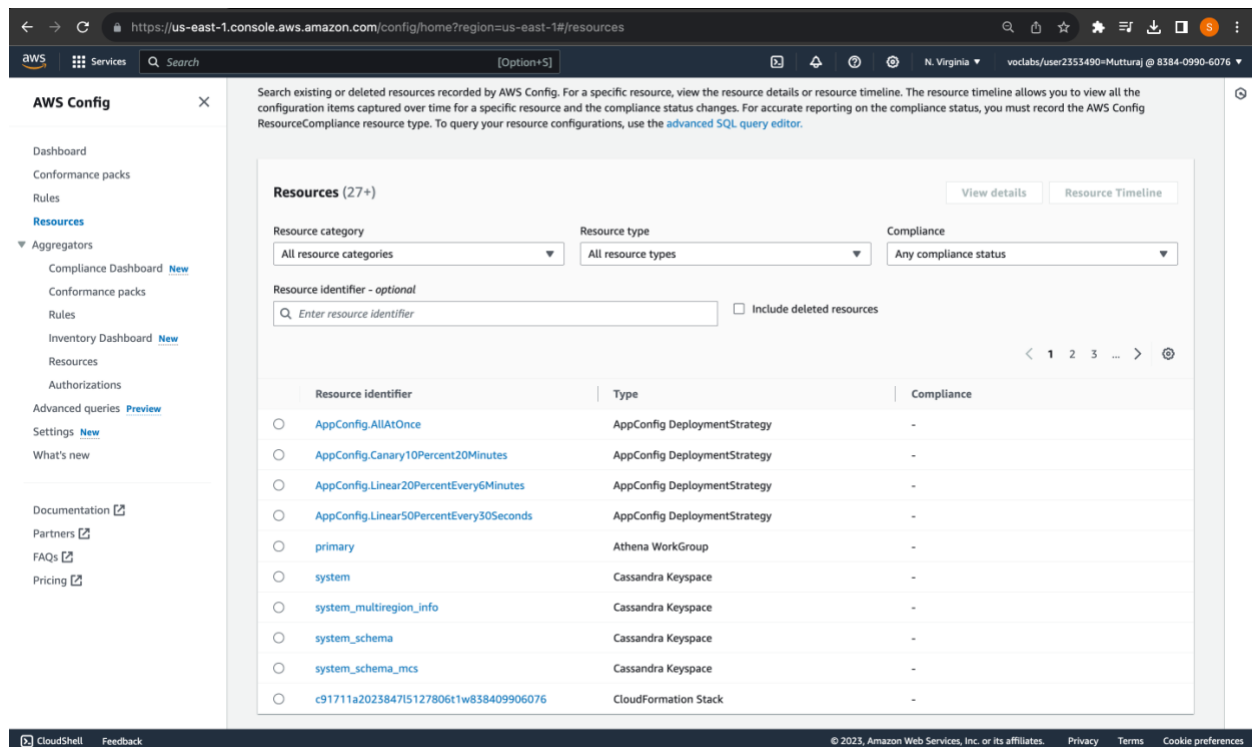


Fig: AWS Config Resources

How to add inbound rules to both security groups and network ACLs

1. To add inbound rules to both security groups and network ACLs:
2. **Update Security Group (TargetSecurityGroup):**
3. In the EC2 console, navigate to Security Groups.
4. Select the target security group (e.g., TargetSecurityGroup).
5. Edit inbound rules, delete existing rule, and add a new rule.
6. Type: HTTP
7. Source: Custom
8. Enter "sg" and select SourceSecurityGroup.
9. Update Network ACL (TargetNetworkACL):
10. In the VPC console, go to Network ACLs.
11. Select the relevant network ACL associated with the target (e.g., TargetNetworkACL).
12. Edit inbound rules, add a new rule.
13. Rule Number: 99
14. Type: HTTP
15. Allow/Deny: Deny

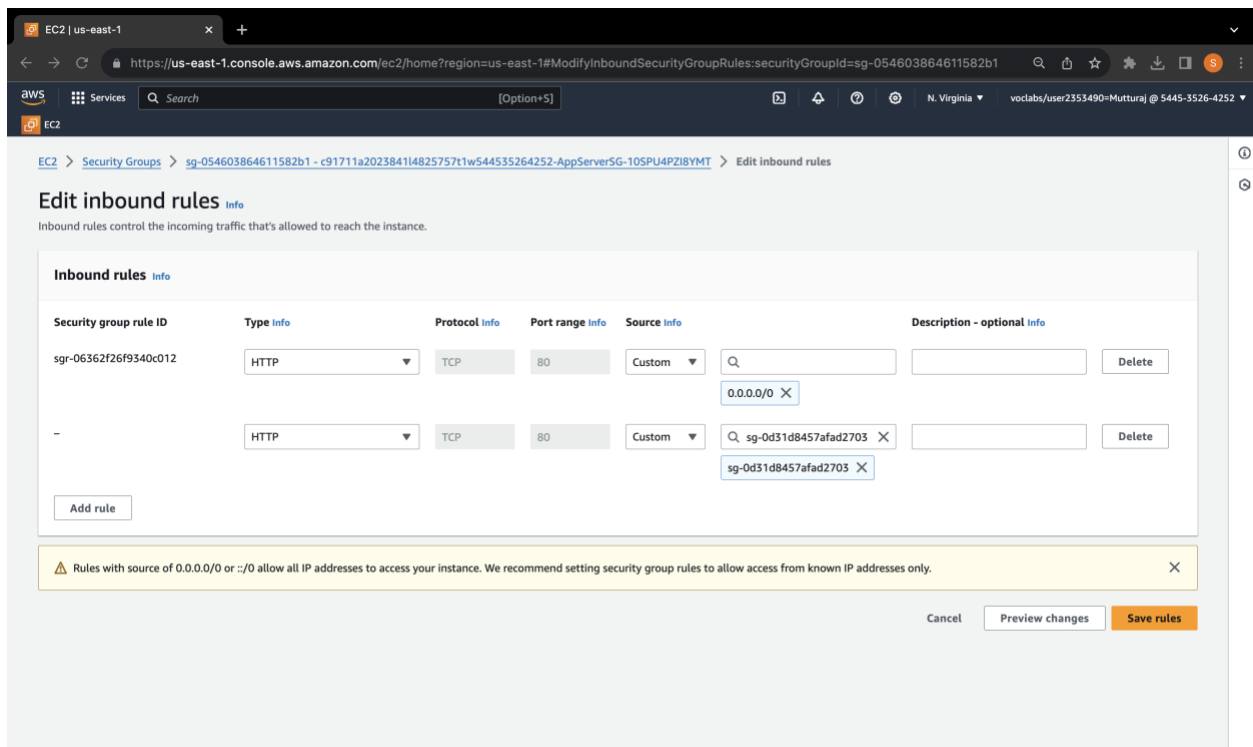


Fig: Adding inbound rules to security groups

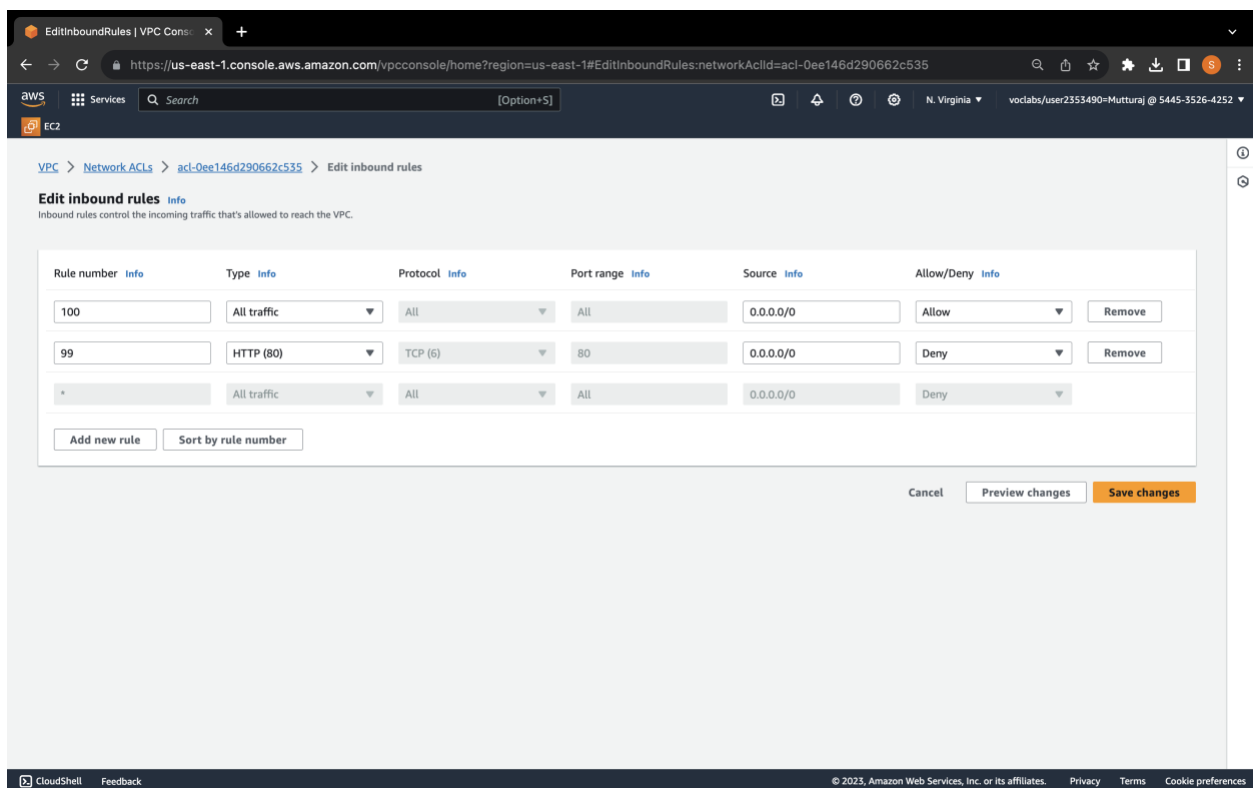


Fig: Adding inbound rules to network ACLs

How to encrypt the root volume of an existing EC2 instance

1. Stop the EC2 Instance:
 - a. In the AWS EC2 Dashboard, select the target instance and stop it.
2. Create an Encrypted Snapshot:
 - a. Select the unencrypted root volume.
 - b. Create a snapshot and tag it as "Unencrypted Root Volume."
3. Create an Encrypted Volume:
 - a. In the "Snapshots" section, select the snapshot you created.
 - b. Choose "Actions" > "Create volume from snapshot."
 - c. Choose the same Availability Zone and enable encryption using a KMS key.
 - d. Create the volume.
4. Swap the Root Volume:
 - a. In the "Volumes" section, detach the old unencrypted root volume.
 - b. Attach the new encrypted volume to the EC2 instance with the device name `"/dev/xvda."`
5. Verify Encrypted Volume:
 - a. Go back to the EC2 instance details and ensure that the attached volume is now encrypted with an associated KMS key.

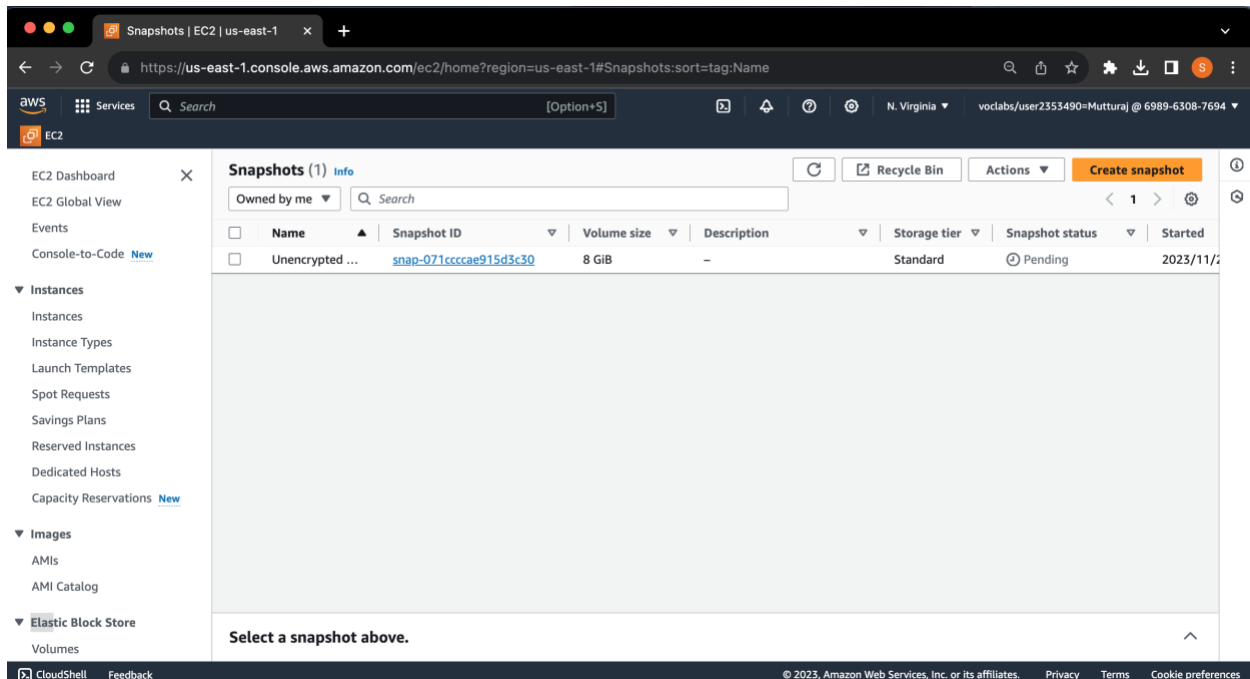


Fig: Unencrypted root volume of an existing EC2 instance

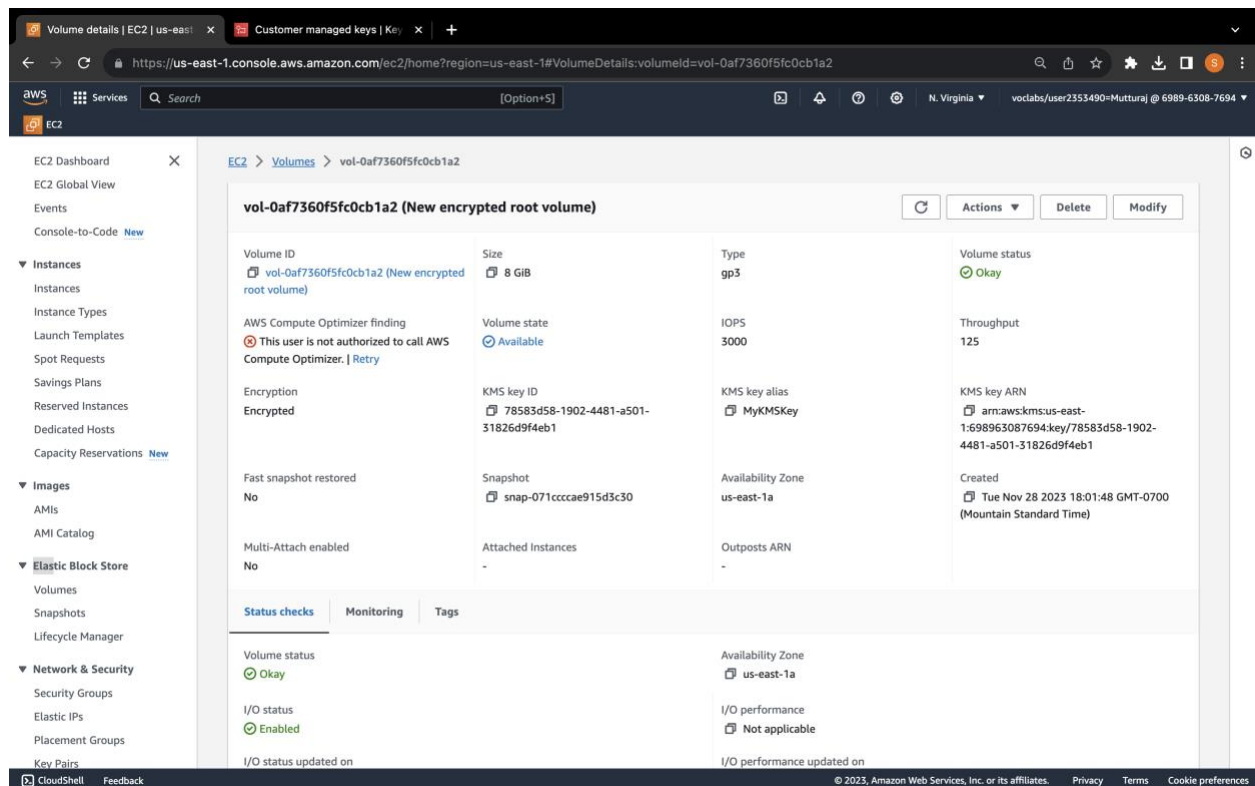


Fig: Encrypted root volume of an existing EC2 instance

How to create a SNS topic

1. In the AWS Management Console, use the search box to find and select "Simple Notification Service (SNS)."
2. f
3. Navigate to Topics, In the navigation pane, choose "Topics."
4. Create a Topic: Click on "Create topic." And Choose "Standard" as the type for the new topic.
5. Enter a name for the topic,
6. Configure Access Policy:
7. Expand the "Access policy - optional" section.
8. Define who can publish messages to the topic by selecting the appropriate option.
9. Define who can subscribe to this topic by choosing the desired option.
10. At the bottom of the page, click "Create topic" to complete the creation of the new SNS topic.

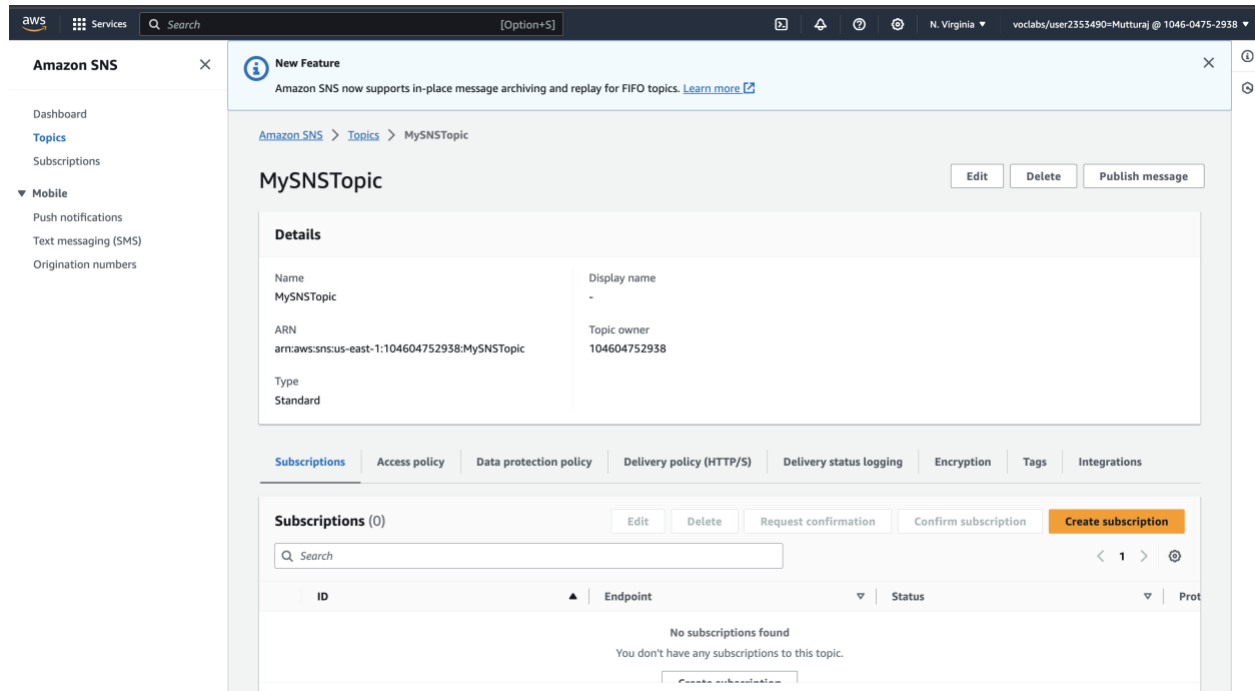


Fig: Creating a SNS topic

How to subscribe to a SNS topic

1. Access Amazon SNS Console in the AWS Management Console.
2. Navigate to "Topics" in the navigation pane.
3. Choose the desired SNS topic and select "Create subscription."
4. Configure the subscription by confirming the pre-filled Topic ARN.
5. Choose "Email" as the protocol and enter a valid email address as the endpoint.
6. Finalize the subscription by scrolling to the bottom and selecting "Create subscription."
7. Check your email for a message from AWS Notifications, click the "Confirm subscription" link, and verify the successful confirmation on the opened webpage.

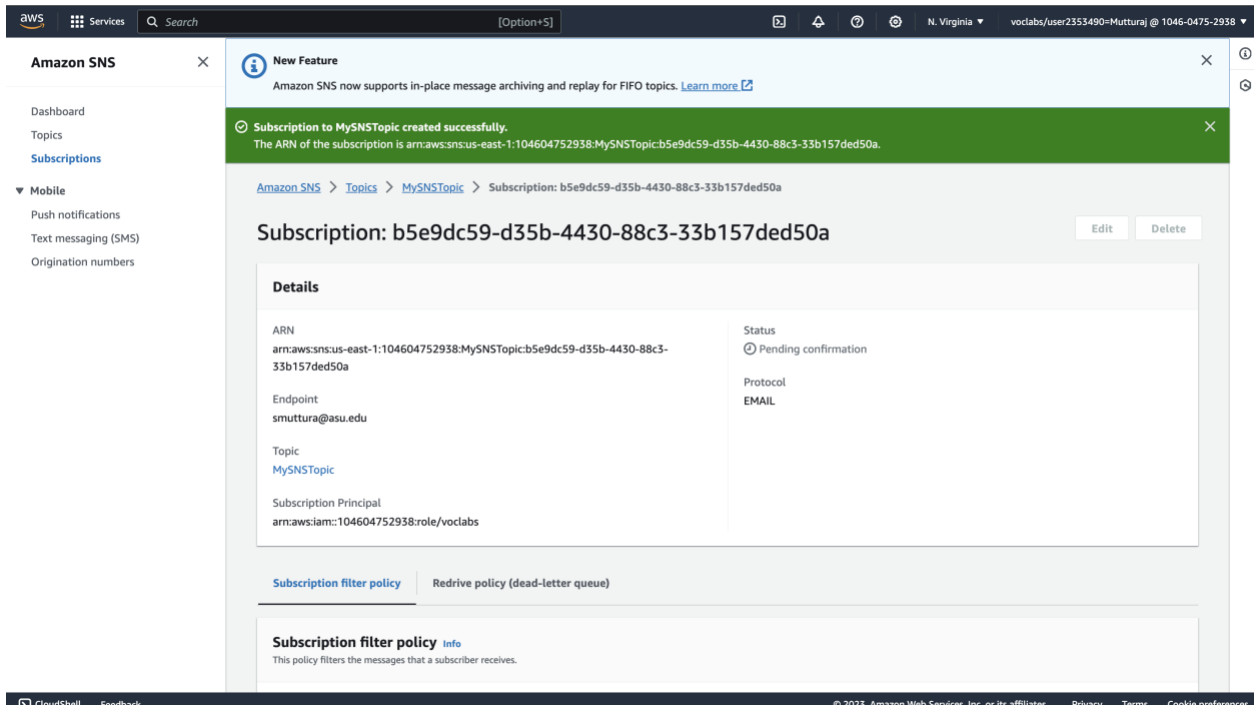


Fig: Subscribing to a SNS topic

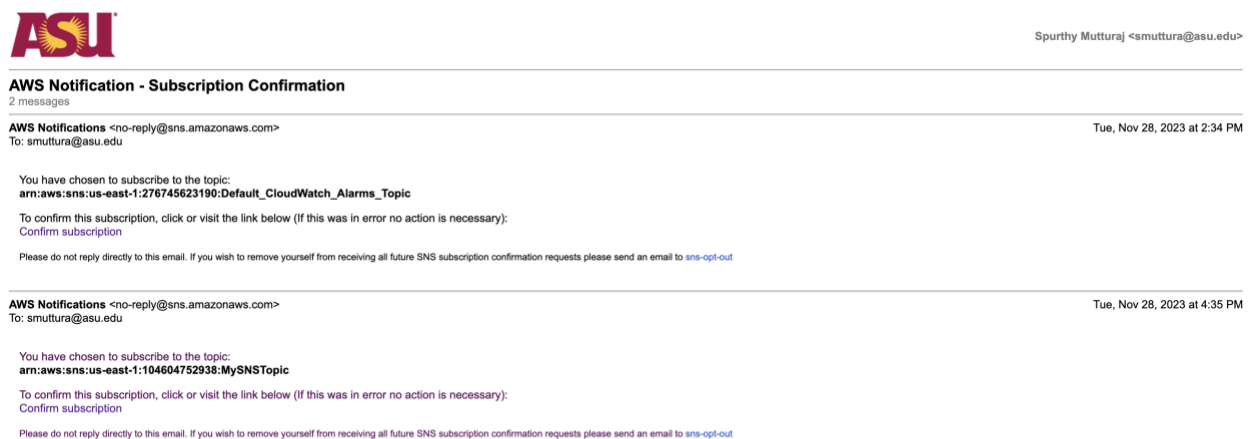


Fig: Email Confirmation of subscribing to a SNS topic

How to create a CloudWatch alarm using a metrics-based filter

1. Access CloudWatch Console in AWS Management Console.
2. Expand Logs in the navigation pane and choose Log groups.
3. Select the desired log group and choose "Create metric filter."

- a. Define the filter pattern using CloudWatch syntax.
 - b. Set filter name, metric namespace, metric name, and metric value.
 - c. Complete the filter creation process.
4. Create CloudWatch Alarm based on Metric Filter.
 - a. Select the created metric filter from the Metric filters tab.
 - b. Choose "Create alarm."
 - c. Configure alarm conditions (e.g., greater/equal to a specific value within a time period).
 - d. Choose an SNS topic for notifications.
 - e. Complete the alarm creation process.
5. Test the Alarm.
 - a. Attempt actions that trigger the metric filter (e.g., incorrect login attempts).
 - b. Observe the alarm state in the CloudWatch console.
6. Graph the Metric.
 - a. Navigate to CloudWatch console and choose All metrics.
 - b. Under Custom namespaces, choose the relevant metric and graph it.
7. Check Alarm Status and Details.
 - a. In the CloudWatch console, expand Alarms and choose All alarms.
 - b. Verify the alarm state, and check the history tab for recent invocations.
8. Check Email Notification.
 - a. Check the inbox of the subscribed email address for SNS notifications related to the alarm.

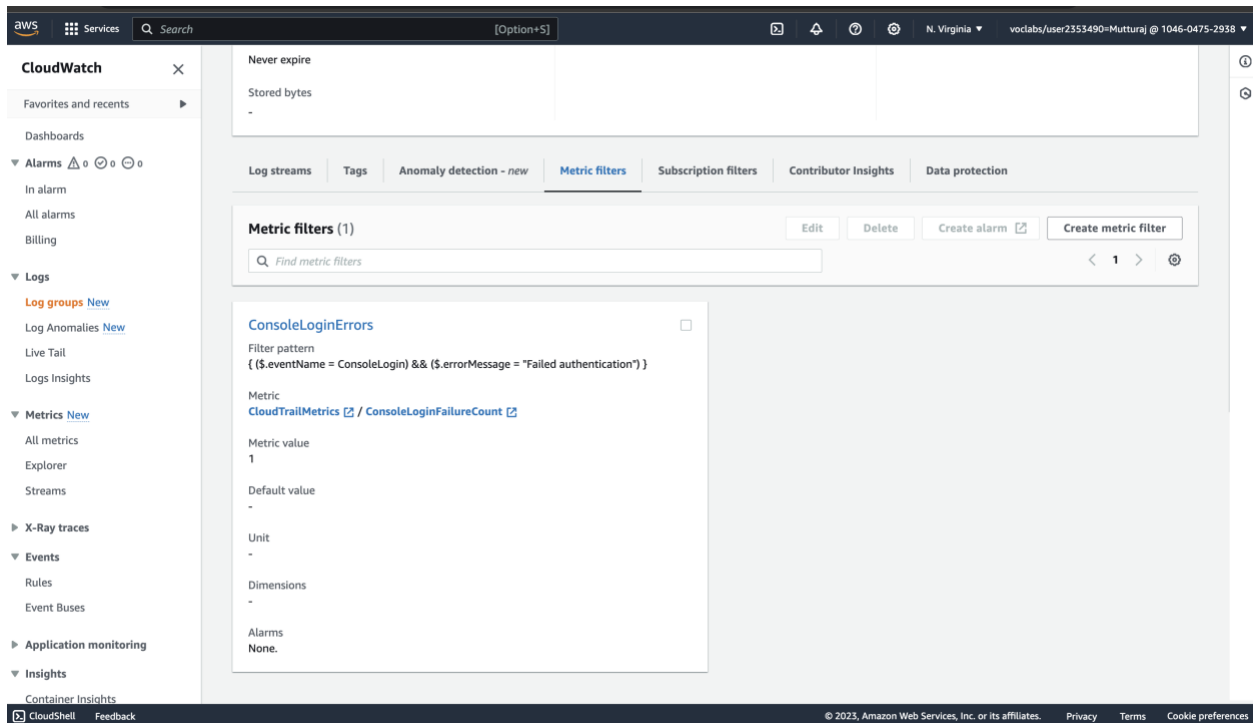


Fig: Creating a metric-based filter

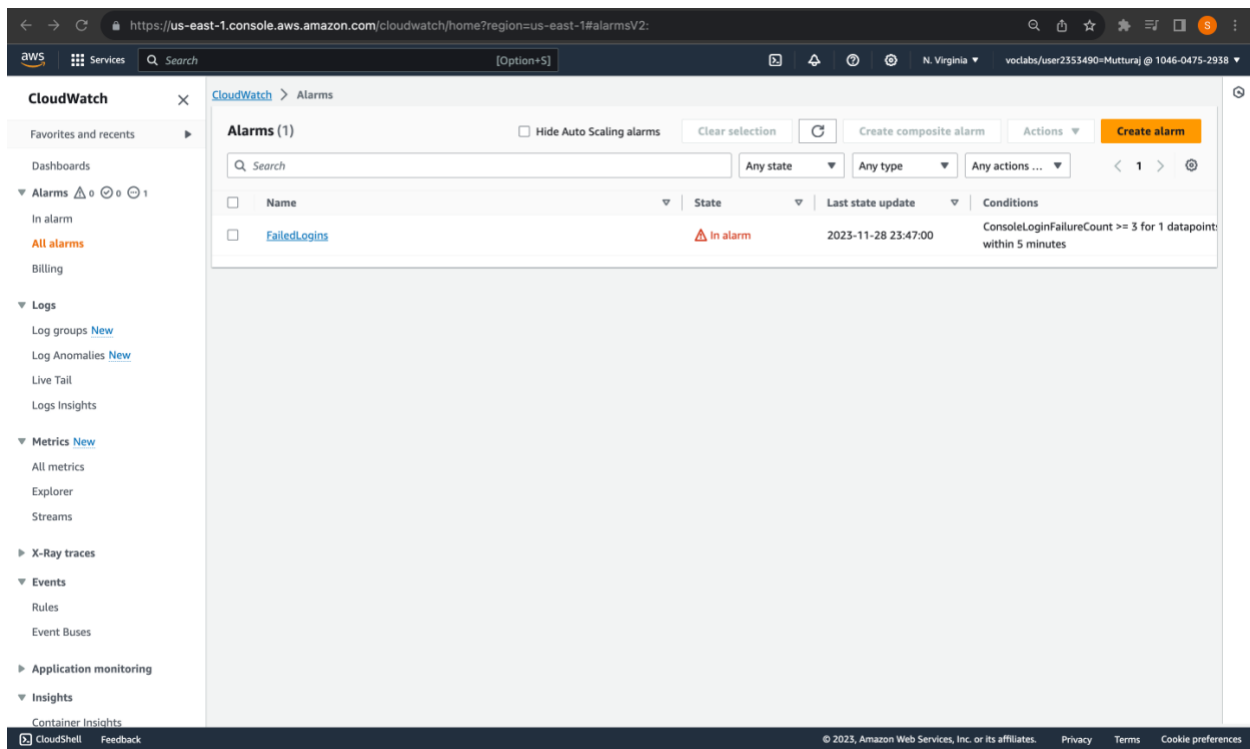


Fig: Creating a CloudWatch alarm using the metrics-based filter