AWS Mini Lab with Proxy, Patch Management, and DB Backup

Objective:

To design and deploy a **secure, automated, and cost-effective hybrid infrastructure on AWS** that includes:

- A **public proxy EC2 instance** for internet access,
- A **private EC2 instance** for internal services,
- **Automated patch management** using scripts or tools (e.g., Ansible),
- And **scheduled backup automation** of critical data to Amazon S3.

This setup simulates a real-world enterprise environment, focusing on **security, automation, and best practices** in AWS infrastructure management.

Tools & Technologies Used:

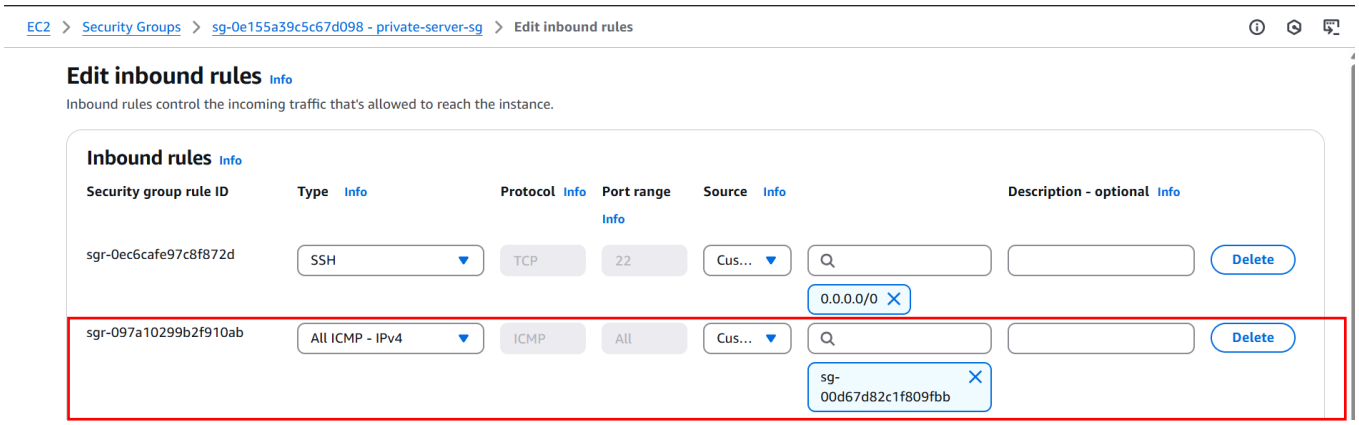
Cloud Platform	AWS (EC2, VPC, S3, IAM)
OS & Scripting	Ubuntu, Shell scripting, Cron
Proxy Service	Squid Proxy
Patch Management	Ansible
Database	MySQL
Programming Language	Python
Monitoring/Logging	System logs, cron log, /var/log/

Process:

- Create vpc for selected region
- Create subnets 1. Public subnet 2. Private subnet
- Create a igw and attached public route table
- Launch ubuntu server 1. Public server 2. Private server
- Change the security groups
    - First public server sg

Give here custom private sg id

- Change the security groups:
  - Private sg



Give here public sg id.

- Connect to public server
- Executive this cmds:
  - sudo apt update
  - sudo apt install squid -y
  - sudo nano /etc/squid/squid.conf (alt + /) paste this one

    acl allowed_ip src <private server private ip>

    http_access allow allowed_ip

Check the line number 1625----- http_access allow all

Check the line number 2175---- http_port 3128

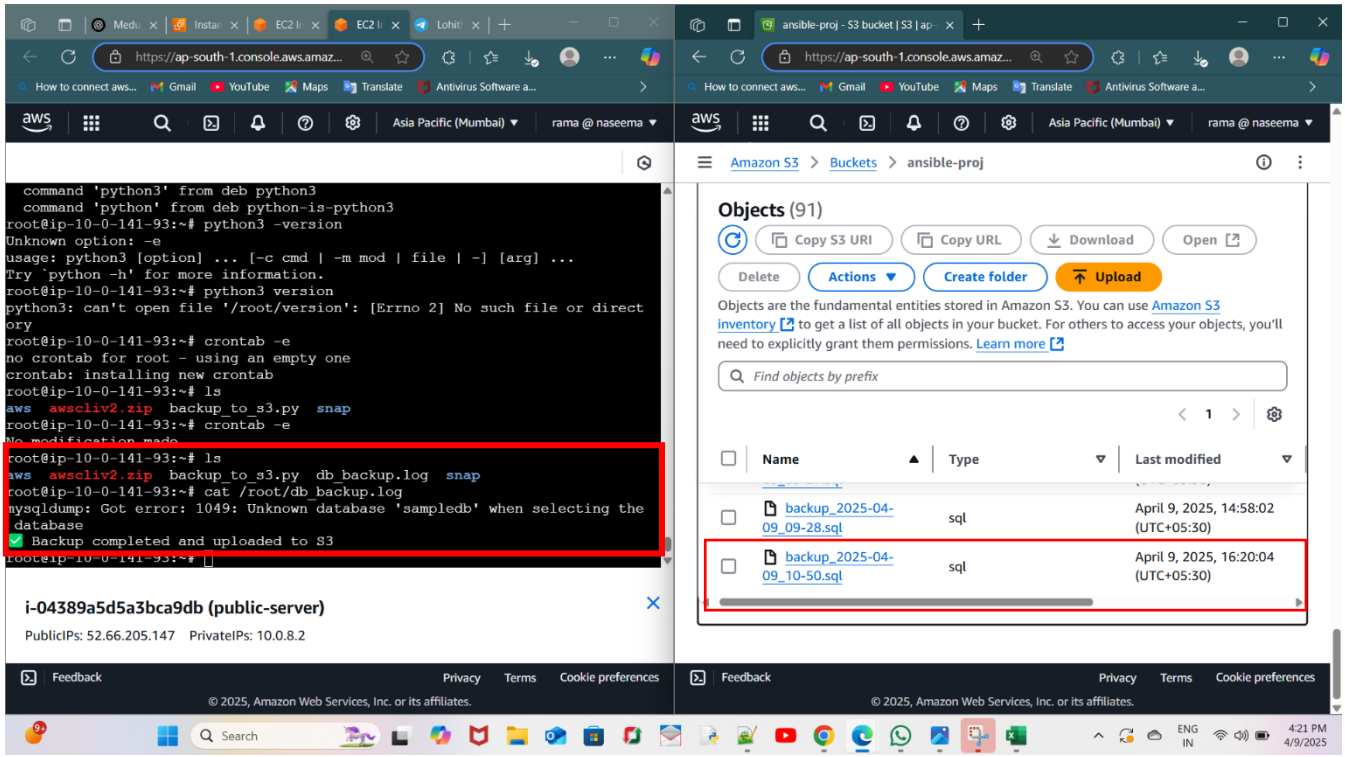
- ctrl+x
- sudo systemctl restart squid
- sudo systemctl enable squid

- Connect to private server
- Attach a role ec2-s3
- Execute this cmds:
  - Vi /etc/environment
    Paste --- export http_proxy=http://<Public-server-pri-ip>:3128
    export https_proxy=http://<Public-server-pri-ip >:3128
  - Source /etc/environment
  - Curl -h google.com #just test only
  - Ping publicserver-pri-ip
  - sudo apt install python3 python3-pip -y
  - python3 --version
  - pip3 –version
  - sudo apt install mysql-server -y
  - sudo mysql_secure_installation
  - sudo systemctl status mysql
  - sudo systemctl start mysql
  - sudo systemctl enable mysql
  - sudo mysql #insert the data in database
  - create bucket with disable block public access
  - aws s3 ls s3://ansible-proj
  - curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
  - apt install unzip
  - unzip awscliv2.zip
  - sudo ./aws/install
  - aws –version
  - aws configure
  - aws s3 ls s3://ansible-proj

- write a code in python
  - vi backup_to_s3.py

```
import boto3
import os
import datetime

#Define bucket name here
bucket_name = 'ansible-proj' #here replace bucket name

timestamp = datetime.datetime.now().strftime('%Y-%m-%d_%H-%M')
backup_file = f"/tmp/backup_{timestamp}.sql"

#Perform MySQL dump
os.system(f"mysqldump -u root sampledb > {backup_file}")

#Upload to S3
s3 = boto3.client('s3')
s3.upload_file(backup_file, bucket_name, f"backup_{timestamp}.sql")
print("Backup completed and uploaded to S3")
```

- crontab -e
  - select 1
  - paste------> *** * * * * /root/bin/python3 /root/backup_to_s3.py >> /root/db_backup.log 2>&1**
  - cat /root/db_backup.log #check the logs
  - aws s3 ls s3://ansible-proj /



Benefits:

- Secure Private Networking
  - Keeps sensitive services (like the database) isolated from the internet.
- Centralized Internet Control
  - All traffic from the private subnet goes through the public proxy—enabling filtering, logging, and monitoring.
- Automated Patch Management
  - Ensures systems are always up-to-date with the latest security updates.
- Scheduled DB Backups
  - Regular and automatic backups prevent data loss.
- Data Durability with S3
  - Amazon S3 provides a secure, reliable backup storage solution.
- Limited External Exposure
  - Only the proxy server has internet access, reducing attack surfaces.

Advantages:

- Improved Security Posture
  - Better control over traffic and access through subnet isolation and IAM policies.
- Efficient Resource Management

    - Public EC2 acts as a hub for managing updates and internet traffic.
- Reliability and Continuity
    - Even if an instance is lost, backups on S3 can be used to restore data.
- Scalability
    - Easily extendable architecture—more private instances can route traffic through the same proxy.
- Customizability
    - Tools like Ansible allow easy customization for updates and monitoring.
- Cost Efficiency