[Challenge] Using AWS CloudFormation to create an AWS VPC and Amazon EC2 instance

Lab Overview

This lab is an environment for creating an Amazon VPC and Amazon EC2 instance (and other supporting elements) using an AWS CloudFormation template. The goal of this lab is to create a CloudFormation template with the following components

- An Amazon Virtual Private Cloud
- An internet gateway attached to the VPC
- Security groups for accessing the VPC configured to allow SSH from anywhere
- A private subnet within the VPC
- An Amazon EC2 instance (t3.micro) within the private subnet (Note: It is not necessary to access the EC2 via SSH or Remote Desktop for a successful solution)

Build and test the lab iterating the solution until all components build. Let the instructor know when the template builds without error so they may review the completed solution.

Lab Restrictions

Access to services is limited to those necessary to successfully build the services listed above.

Step 1: Understanding Architecture in this lab

1) VPC - *Contains* -> PrivateSubnet
2) VPC - *Contains* -> AppSecurityGroup
3) VPC - *Connect* -> IGW
4) IGW - *Connected by* -> VPCtoIGWConnection
5) PrivateSubnet - *Associated with* -> PrivateRouteTable
6) PrivateSubnet - *Contains* -> Instance
7) Instance - *Assigned* -> AppSecurityGroup

Step 2: Configuration CloudFormation YAML template

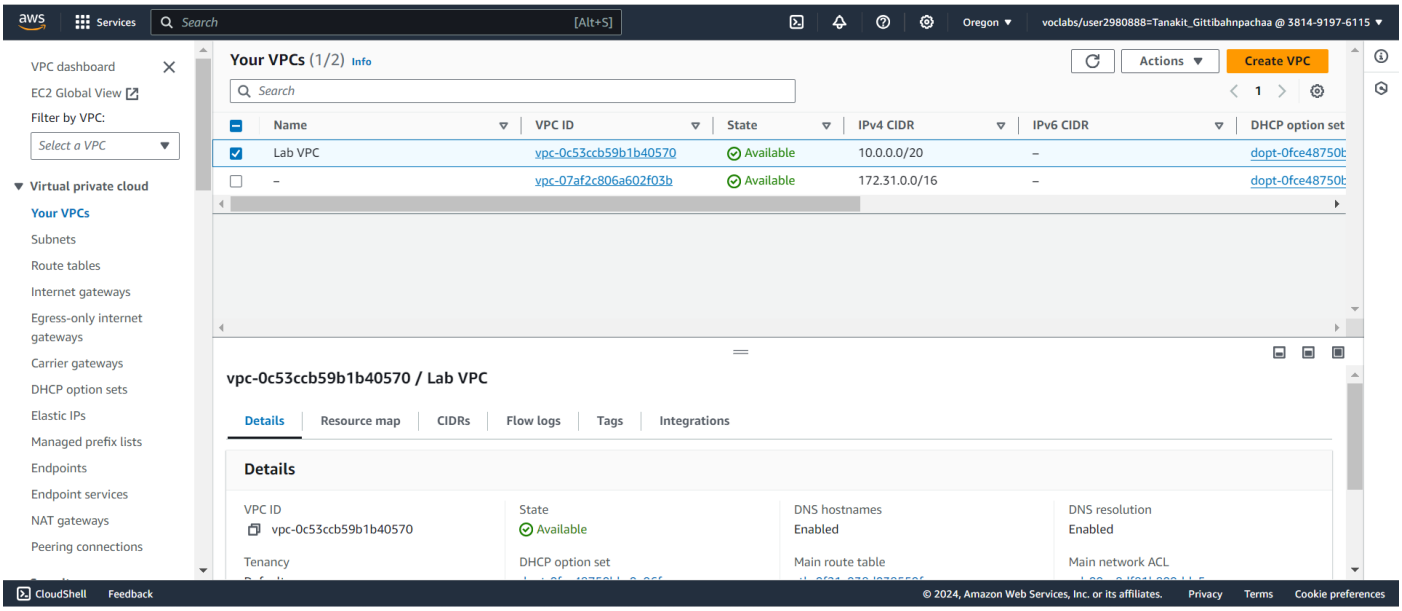
```yaml
# Lab VPC with Private subnet and Internet Gateway

Parameters:
  LabVpcCidr:
    Type: String
    Default: 10.0.0.0/20

  PrivateSubnetCidr:
    Type: String
    Default: 10.0.0.0/24

  AmazonLinuxAMIID:
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2

Resources:
  Instance:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: t3.micro
      ImageId: !Ref AmazonLinuxAMIID
      SubnetId: !Ref PrivateSubnet
      SecurityGroupIds:
        - !Ref AppSecurityGroup
      Tags:
        - Key: Name
          Value: App Server

# VPC with Internet Gateway

  LabVPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref LabVpcCidr
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: Lab VPC

  IGW:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: Lab IGW

  VPCtoIGWConnection:
    Type: AWS::EC2::VPCGatewayAttachment
    DependsOn:
      - IGW
```

```
      - LabVPC
    Properties:
      InternetGatewayId: !Ref IGW
      VpcId: !Ref LabVPC

# Private Route Table

  PrivateRouteTable:
    Type: AWS::EC2::RouteTable
    DependsOn: LabVPC
    Properties:
      VpcId: !Ref LabVPC
      Tags:
        - Key: Name
          Value: Private Route Table

# Private Subnet

  PrivateSubnet:
    Type: AWS::EC2::Subnet
    DependsOn: LabVPC
    Properties:
      VpcId: !Ref LabVPC
      CidrBlock: !Ref PrivateSubnetCidr
      AvailabilityZone: !Select
        - 0
        - !GetAZs
          Ref: AWS::Region
      Tags:
        - Key: Name
          Value: Private Subnet

  PrivateRouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    DependsOn:
      - PrivateRouteTable
      - PrivateSubnet
    Properties:
      RouteTableId: !Ref PrivateRouteTable
      SubnetId: !Ref PrivateSubnet

# App Security Group

  AppSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    DependsOn: LabVPC
    Properties:
      GroupName: App
      GroupDescription: Enable access to App
      VpcId: !Ref LabVPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
```

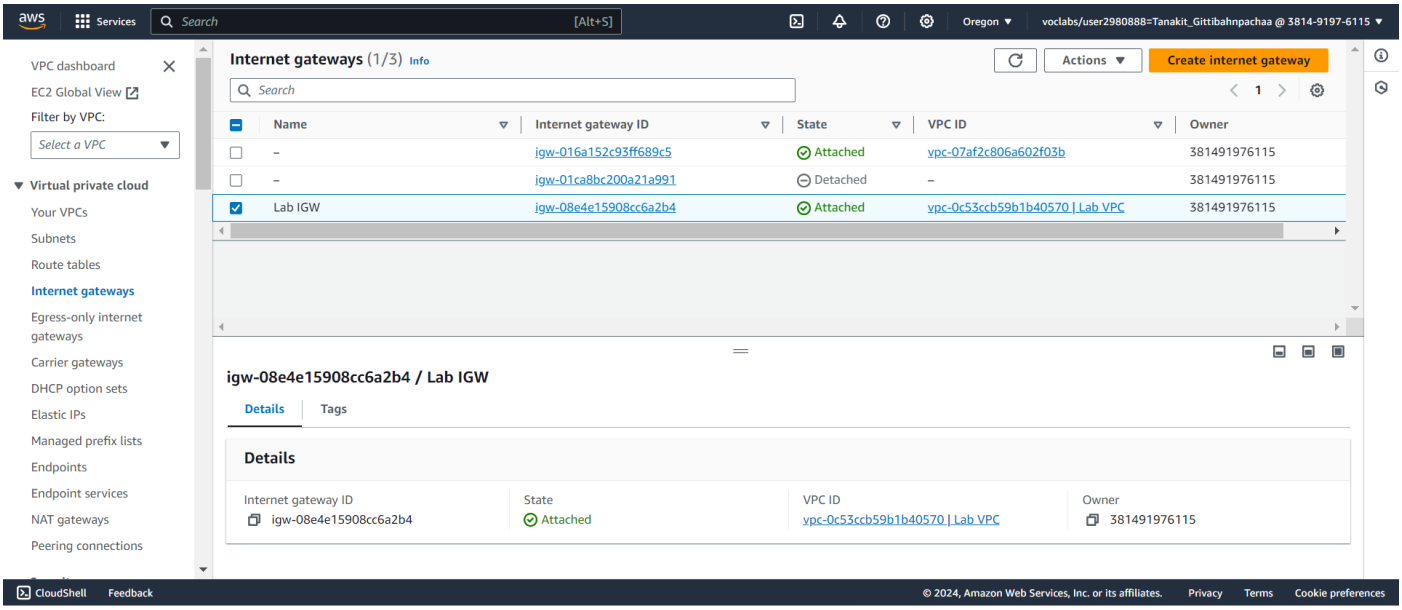
```
          CidrIp: 0.0.0.0/0
      Tags:
        - Key: Name
          Value: App

# Outputs

Outputs:
  LabVPCDefaultSecurityGroup:
    Value: !Sub ${LabVPC.DefaultSecurityGroup}
```

Step 3: Deploy a CloudFormation Stack

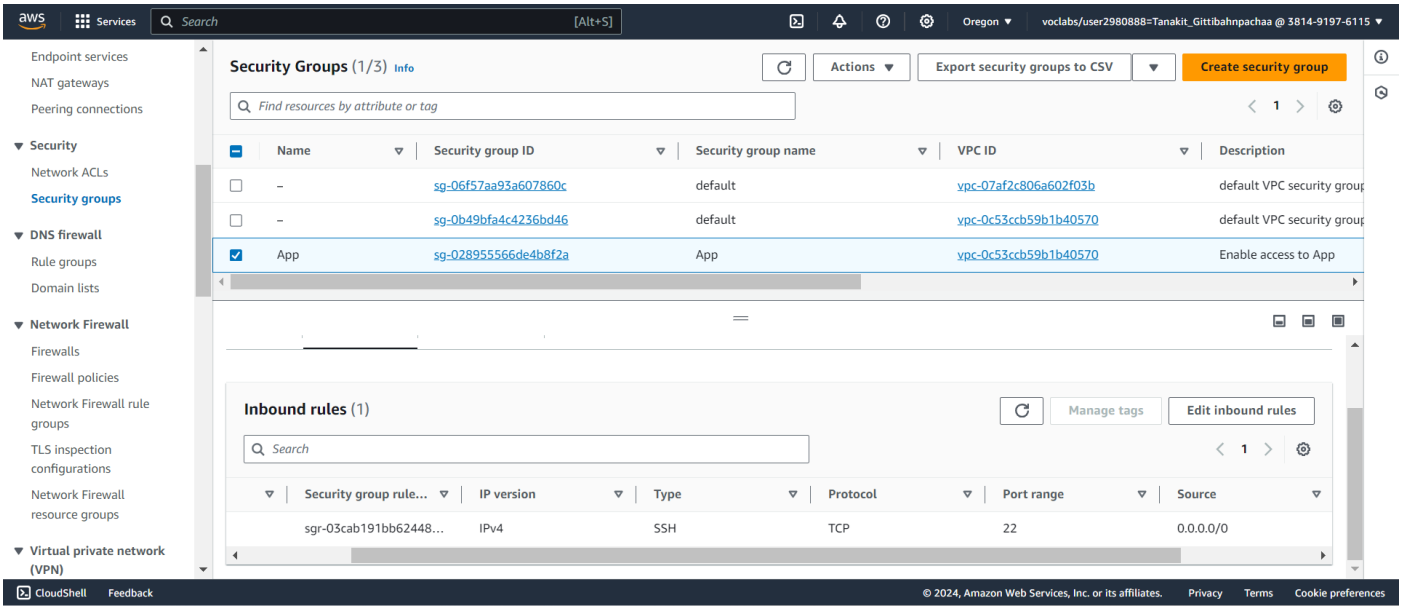
CloudFormation > Create Stack > Prepare template: Choose an existing template > Specify template: Upload a template file > Next > Next (ตั้งค่าหน้า Configure stack options เป็น default) > Review and create กด Submit > หากไม่มีข้อผิดพลาด จะเห็นว่า Stacks แสดงสถานะ CREATE_COMPLETE



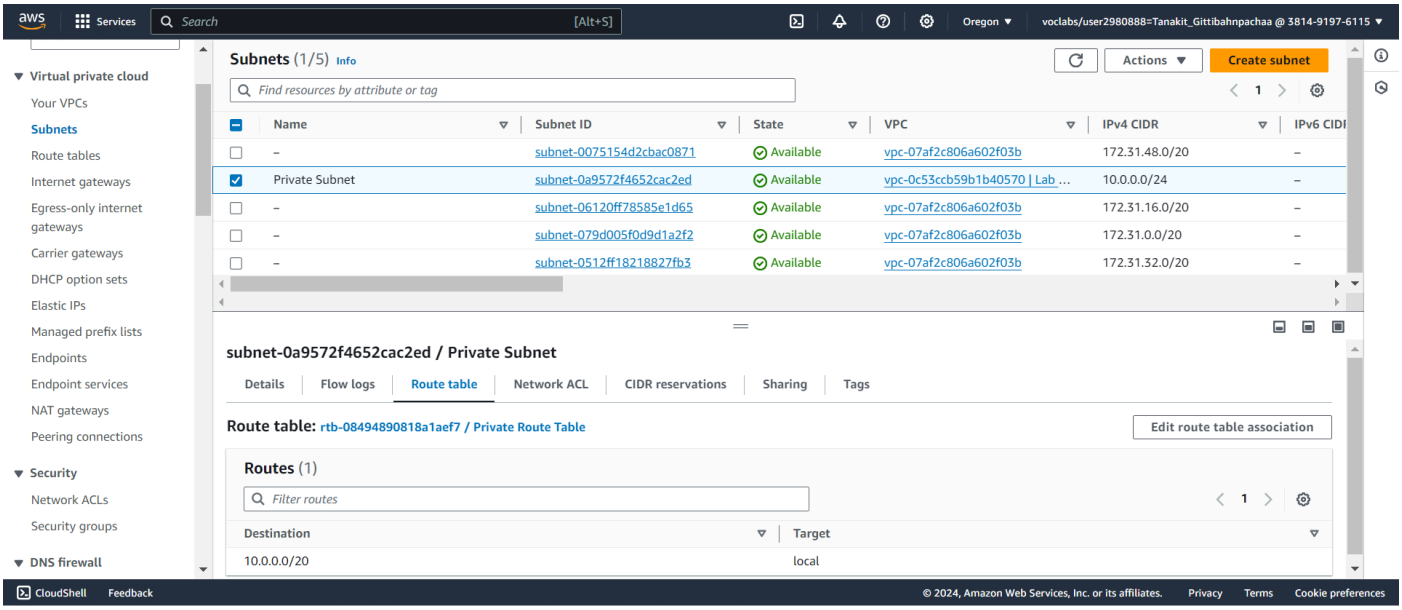
An Amazon Virtual Private Cloud



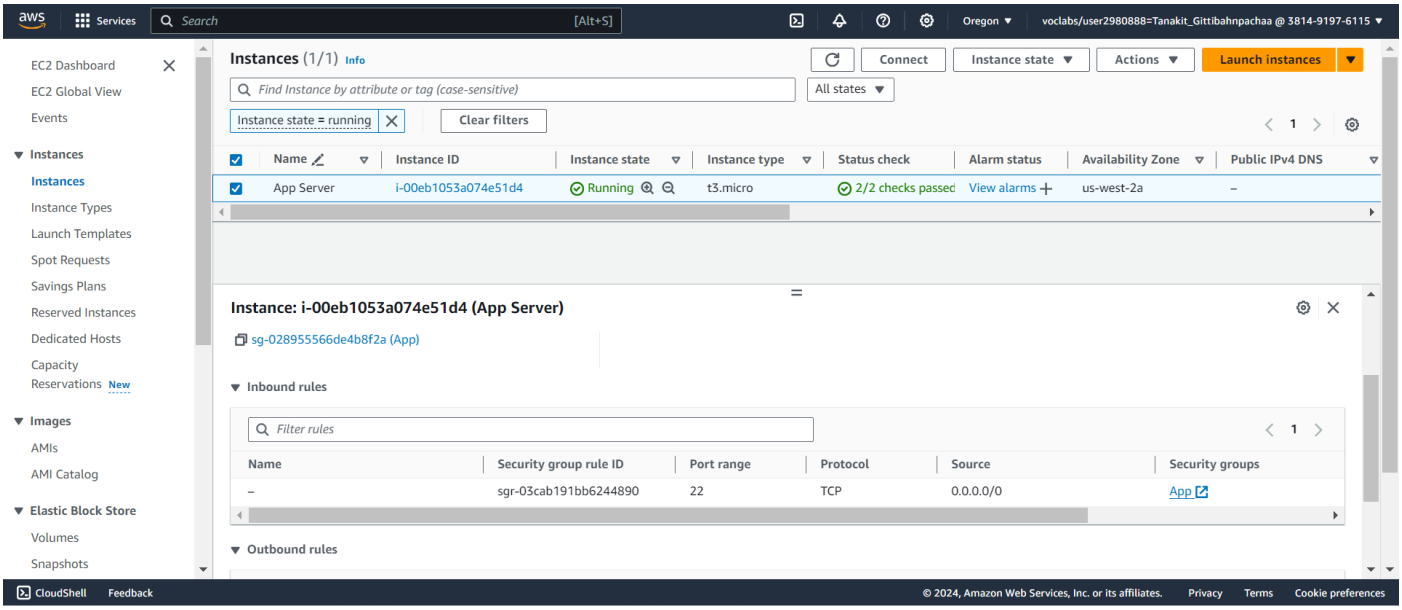
An internet gateway attached to the VPC



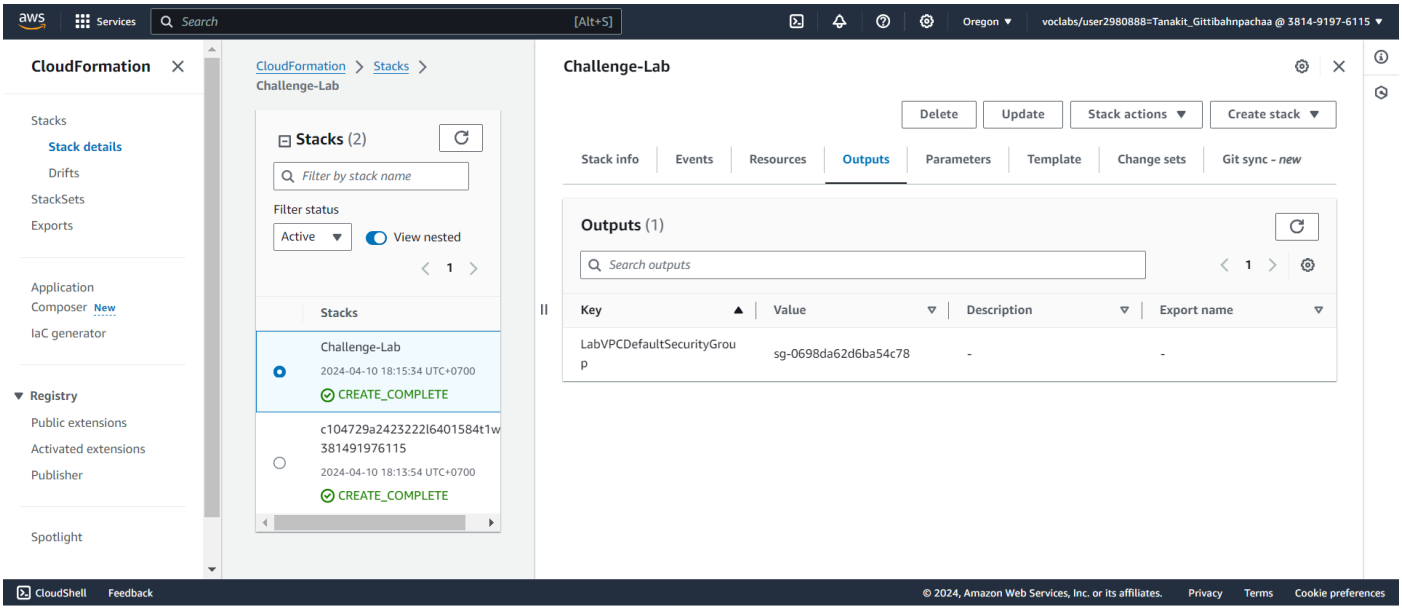
Security groups for accessing the VPC configured to allow SSH from anywhere



A private subnet within the VPC



An Amazon EC2 instance (t3.micro) within the private subnet (Note: It is not necessary to access the EC2 via SSH or Remote Desktop for a successful solution)



Outputs from LabVPC

หมายเหตุ:

ใน Outputs นี้ จะคืนค่า Default Security Group ของ LabVPC ที่ได้สร้างไว้ในส่วนของ Resources

ค่าที่จะได้รับจาก Output นี้ คือ !Sub ${LabVPC.DefaultSecurityGroup} ซึ่งเป็นการใช้ Intrinsic Function !Sub ในการรับค่าของ Default Security Group จาก Resource LabVPC

เมื่อต้องการดึงข้อมูลจาก Output นี้ เช่น ในการใช้ AWS CLI หรือ CloudFormation ในภายหลัง ก็จะได้ค่า Default Security Group ของ LabVPC ที่ได้สร้างไว้

ดังนั้น โดยสรุปแล้ว Output นี้จะเป็นการคืนค่า Default Security Group ของ LabVPC ออกมา เพื่อให้สามารถนำไปใช้งานได้ในภายหลัง เช่น ใช้กำหนด Security Group ให้กับ Resource อื่นๆ ที่เกี่ยวข้อง

ฟังก์ชัน Intrinsic Functions ที่สำคัญมีดังนี้:

- !Ref - ใช้อ้างอิงหา Resource ที่ได้กำหนดไว้ในแบบ
- !GetAtt - ใช้อ้างอิงหาแอตทริบิวต์ของ Resource
- !Sub - ใช้แทนที่ค่าต่างๆ ในสตริง
- !Join - ใช้ในการรวมสตริงหลายๆ อัน
- !Split - ใช้แยกสตริงออกเป็นหลายส่วน
- !FindInMap - ใช้ในการค้นหาข้อมูลจาก Mapping
- !Select - ใช้ในการเลือกข้อมูลจากลิสต์
- !Cidr - ใช้ในการคำนวณ CIDR block