**Case Study:**

**Part 1 – Automation**

* **Step-1 Cloudformation:**

**Code:**

Parameters:

  VpcCIDR:

    Description: Please enter the IP range (CIDR notation) for this VPC

    Type: String

    Default: "172.30.0.0/16"

  PublicSubnetCIDR:

    Description: Please enter the IP range (CIDR notation) for the public subnet in the first Availability Zone

    Type: String

    Default: "172.30.0.0/24"

  PrivateSubnetCIDR:

    Description: Please enter the IP range (CIDR notation) for the private subnet in the first Availability Zone

    Type: String

    Default: "172.30.1.0/24"

Resources:

  VPC:

    Type: 'AWS::EC2::VPC'

    Properties:

      CidrBlock:

        Ref: VpcCIDR

      EnableDnsSupport: true

      EnableDnsHostnames: true

      Tags:

        - Key: Name

          Value: Prj VPC

  InternetGateway:

    Type: 'AWS::EC2::InternetGateway'

    Properties:

      Tags:

        - Key: Name

          Value: IGW

  InternetGatewayAttachment:

    Type: AWS::EC2::VPCGatewayAttachment

    Properties:

      VpcId:

        Ref: VPC

      InternetGatewayId:

        Ref: InternetGateway

  PublicSubnet:

    Type: AWS::EC2::Subnet

    Properties:

      VpcId:

        Ref: VPC

      AvailabilityZone: "ap-south-1a"

      CidrBlock:

        Ref: PublicSubnetCIDR

      MapPublicIpOnLaunch: true

      Tags:

        - Key: Name

          Value: Public Subnet

  PrivateSubnet:

    Type: AWS::EC2::Subnet

    Properties:

      VpcId:

        Ref: VPC

      AvailabilityZone: "ap-south-1a"

      CidrBlock:

        Ref: PrivateSubnetCIDR

      MapPublicIpOnLaunch: false

      Tags:

        - Key: Name

          Value: Private Subnet

  PublicRouteTable:

    Type: AWS::EC2::RouteTable

    Properties:

      VpcId:

        Ref: VPC

      Tags:

        - Key: Name

          Value: Public Routes

  DefaultPublicRoute:

    Type: AWS::EC2::Route

    DependsOn: InternetGatewayAttachment

    Properties:

      RouteTableId:

        Ref: PublicRouteTable

      DestinationCidrBlock: 0.0.0.0/0

      GatewayId:

        Ref: InternetGateway

  PublicSubnetRouteTableAssociation:

    Type: AWS::EC2::SubnetRouteTableAssociation

    Properties:

      RouteTableId:

        Ref: PublicRouteTable

      SubnetId:

        Ref: PublicSubnet

  NatGateway1EIP:

    Type: AWS::EC2::EIP

    DependsOn: InternetGatewayAttachment

    Properties:

      Domain: vpc

  NatGateway1:

    Type: AWS::EC2::NatGateway

    Properties:

      AllocationId:

        Fn::GetAtt: NatGateway1EIP.AllocationId

      SubnetId:

        Ref: PublicSubnet

  PrivateRouteTable:

    Type: AWS::EC2::RouteTable

    Properties:

      VpcId:

        Ref: VPC

      Tags:

        - Key: Name

          Value: Private Routes

  DefaultPrivateRoute:

    Type: AWS::EC2::Route

    Properties:

      RouteTableId:

        Ref: PrivateRouteTable

      DestinationCidrBlock: 0.0.0.0/0

      NatGatewayId:

        Ref: NatGateway1

  PrivateSubnetRouteTableAssociation:

    Type: AWS::EC2::SubnetRouteTableAssociation

    Properties:

      RouteTableId:

        Ref: PrivateRouteTable

      SubnetId:

        Ref: PrivateSubnet

Outputs:

  VPC:

    Description: A reference to the created VPC

    Value:

      Ref: VPC

  PublicSubnet:

    Description: A reference to the public subnet in the 1st Availability Zone

    Value:

      Ref: PublicSubnet

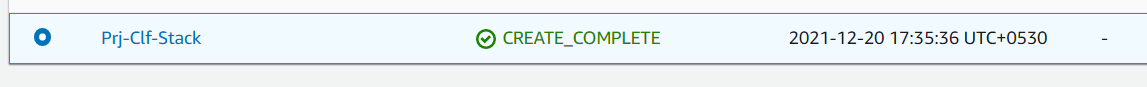
  PrivateSubnet:

    Description: A reference to the private subnet in the 1st Availability Zone

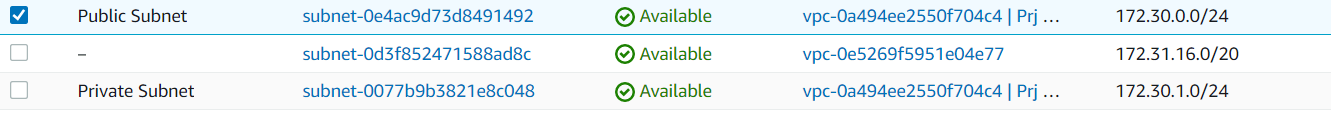
    Value:

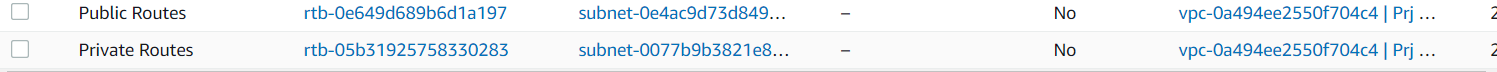
      Ref: PrivateSubnet

**Output:**

****

****

****

****

****

**Step 2: Terraform**

1. Created clf.tak:
2. provider "aws" {
3. region = "ap-south-1"
4. access\_key = "AKIAUJFZ6CS65V5IJI4W"
5. secret\_key = "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"
6. }
7. variable "region" {
8. type = string
9. default = "ap-south-1"
10. }
11. resource "aws\_security\_group" "prj-sg1" {
12. name   = "prj-sg"
13. vpc\_id = "vpc-0a494ee2550f704c4"
14. }
15. resource "aws\_security\_group\_rule" "inbound\_ssh" {
16. from\_port         = 22
17. protocol          = "tcp"
18. security\_group\_id = "${aws\_security\_group.prj-sg1.id}"
19. to\_port           = 22
20. type              = "ingress"
21. cidr\_blocks       = ["0.0.0.0/0"]
22. }
24. resource "aws\_security\_group\_rule" "inbound\_http" {
25. from\_port         = 80
26. protocol          = "tcp"
27. security\_group\_id = "${aws\_security\_group.prj-sg1.id}"
28. to\_port           = 80
29. type              = "ingress"
30. cidr\_blocks       = ["0.0.0.0/0"]
31. }
32. resource "aws\_security\_group\_rule" "outbound\_all" {
33. from\_port         = 0
34. protocol          = "-1"
35. security\_group\_id = "${aws\_security\_group.prj-sg1.id}"
36. to\_port           = 0
37. type              = "egress"
38. cidr\_blocks       = ["0.0.0.0/0"]
39. }

Created instance.tf:

# create ec2 instance in pulic subnet

resource "aws\_instance" "private-server" {

 ami           = "ami-08ee6644906ff4d6c"

  key\_name = "Mumbai123.kp"

  instance\_type = "t2.micro"

  security\_groups = [aws\_security\_group.prj-sg1.id]

  subnet\_id      = "subnet-0077b9b3821e8c048"

  tags= {

   Name = "EC2-with-VPC2"

  }

}

Create Elb.tf:

##Create Launch Configuration

resource "aws\_launch\_configuration" "prj-launch" {

  image\_id = "ami-08ee6644906ff4d6c"

  instance\_type = "t2.micro"

  iam\_instance\_profile = "S3\_full\_access"

  security\_groups = [aws\_security\_group.prj-sg1.id]

  key\_name = "Mumbai123.kp"

}

## Creating ASG in Public Subnet:

resource "aws\_autoscaling\_group" "prj-asg" {

max\_size = 5

min\_size = 2

health\_check\_grace\_period = 300

health\_check\_type = "ELB"

desired\_capacity = 2

force\_delete = true

launch\_configuration = "${aws\_launch\_configuration.prj-launch.name}"

vpc\_zone\_identifier = ["subnet-0e4ac9d73d8491492"]

tag {

  key = "Name"

  value = "Prj\_ASG"

  propagate\_at\_launch = true

}

lifecycle {create\_before\_destroy = true}

}

### Creating ELB in public subnet

resource "aws\_elb" "Prj-ELB" {

  name= "Prj-ELB"

  security\_groups = ["${aws\_security\_group.prj-sg1.id}"]

  subnets = ["subnet-0e4ac9d73d8491492"]

  cross\_zone\_load\_balancing = true

  health\_check {

    healthy\_threshold = 5

    unhealthy\_threshold = 2

    timeout = 3

    interval = 30

    target = "HTTP:80/"

  }

  listener {

        lb\_port= "80"

        lb\_protocol = "http"

        instance\_port = "80"

        instance\_protocol = "http"

  }

}

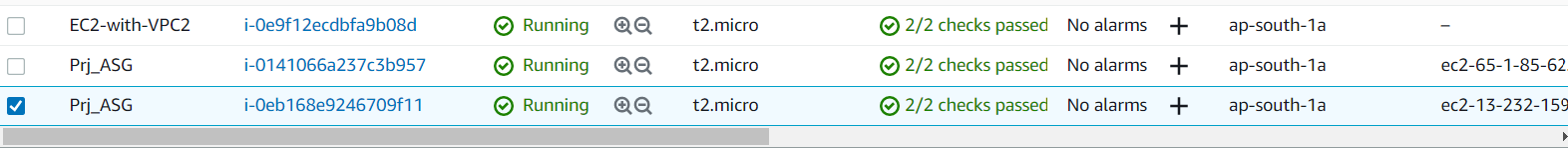
resource "aws\_autoscaling\_attachment" "asg\_attachment" {

  autoscaling\_group\_name = aws\_autoscaling\_group.prj-asg.id

  elb = aws\_elb.Prj-ELB.id

}

Outputs:







Step-3 Ansible:

Login as ubuntu

mkdir .ssh

vi .ssh/authorized\_keys

paste the content of id\_rsa.pub here

finally from ansible server execute the following command and check:

ssh ubuntu@ansible-node-privateip

ansible-server:

-----------------

sudo apt-get install software-properties-common

sudo apt-add-repository ppa:ansible/ansible

sudo apt-get update

sudo apt-get install ansible

sudo apt-get install unzip tree

curl "https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.zip" -o "awscliv2.zip"

unzip awscliv2.zip

sudo ./aws/install

sudo apt-get install -y python-pip

pip –version

pip install boto3

aws configure

mkdir roles

cd roles

ansible-galaxy init apacherole

ls

tree apacherole/

tasks/install.yml:

# installing apache2

- name: installing apache2 server

apt:

name: apache2

state: present

tasks/configure.yml:

---------------------------

---

- name: create the webpage index.html

copy: src=index.html dest=/var/www/html/index.html

pbdemo.yaml:

-------------------------

---

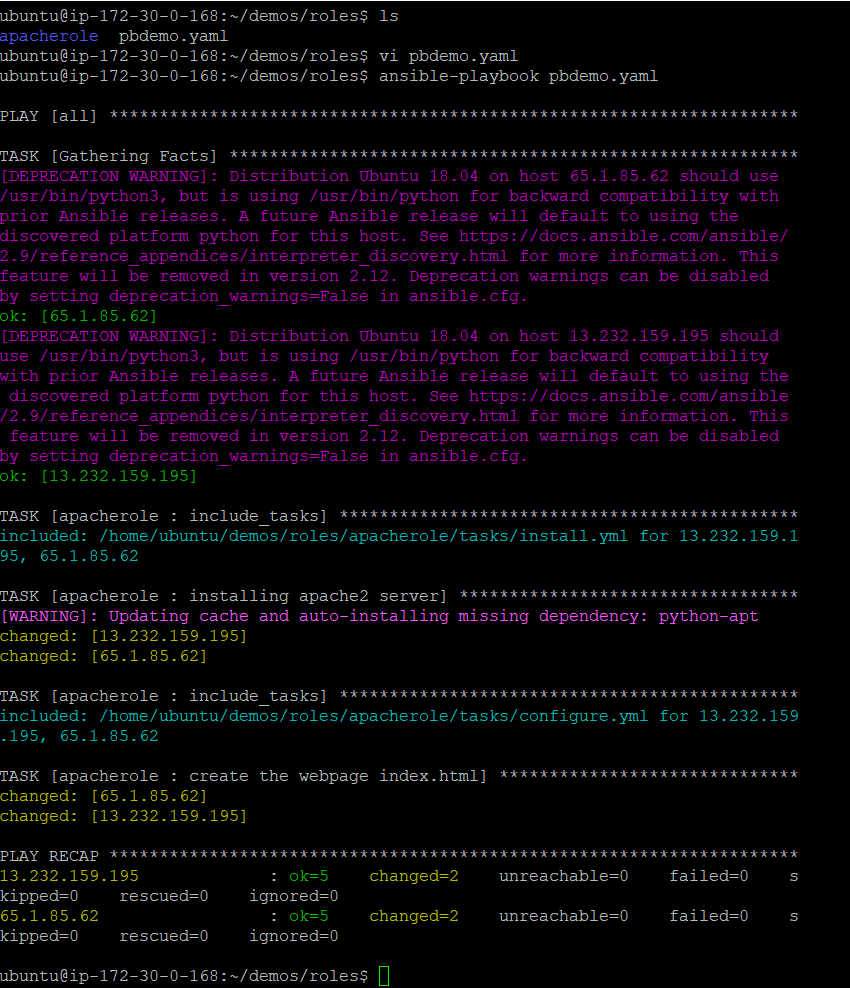
- hosts: all

become: true

roles:

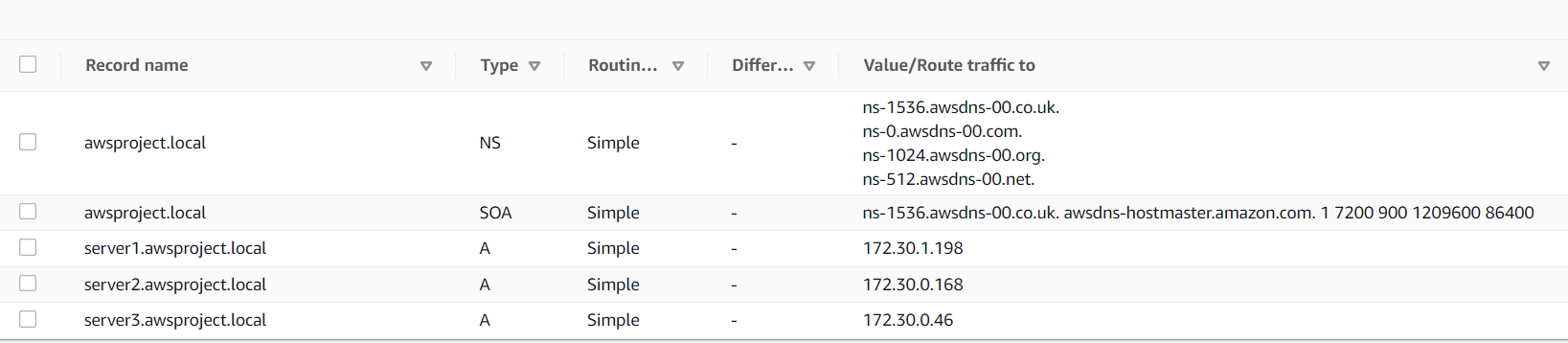
- apacherole

Output:



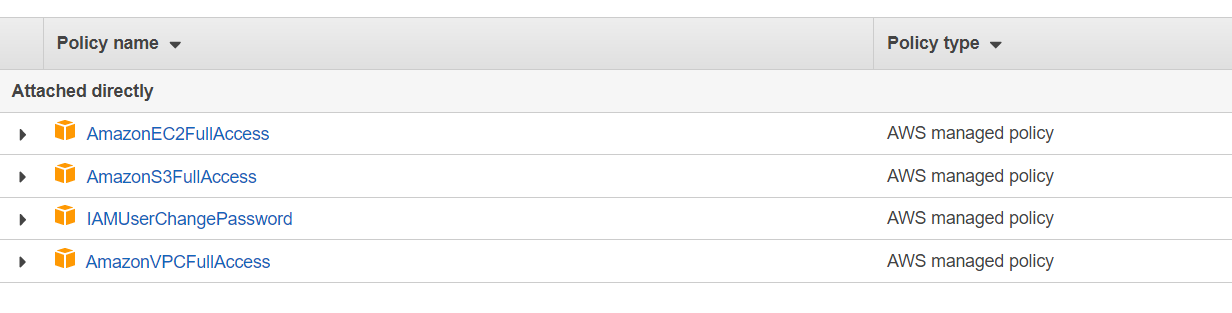
**Part-2 (Manual Configuration):**

1. **Route53 Configuration for VPC internal communication:**

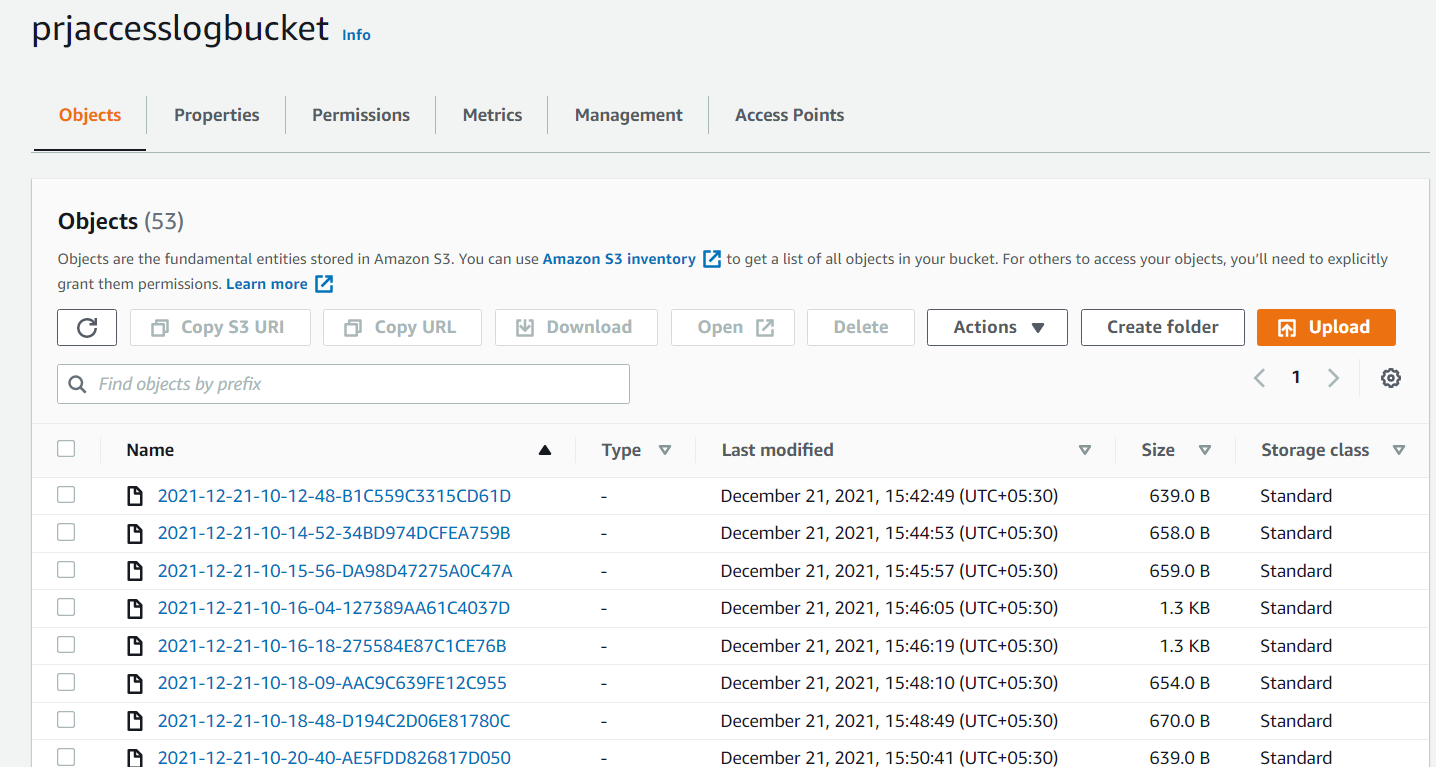
****

1. **A user to manage application infrastructure (VPC, Route53, EC2)**

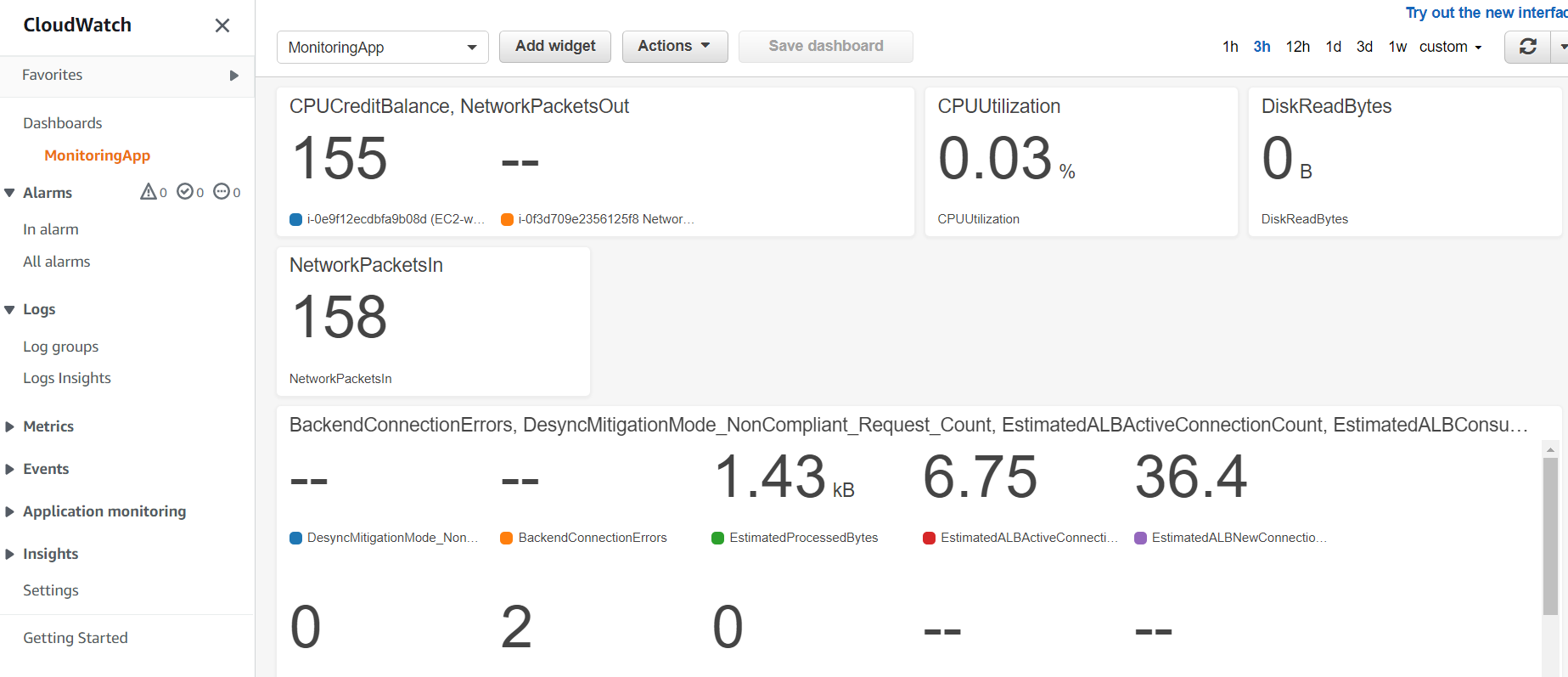
**IAM role created with the policies**

****

1. **S3 Bucket to store log files:**



**Part 3 : Required to create a CloudWatch Dashboard to monitor :**

****