

Final Project

Part - 01

Setting up the environment:

- Creating an AWS org named **root**.

[AWS Organizations](#) > [AWS accounts](#)

AWS accounts

Add an AWS account

The accounts listed below are members of your organization. The organization's management account is responsible for paying the bills for all accounts in the organization. You can use the tools provided by AWS Organizations to centrally manage these accounts. [Learn more](#)

Organization

Organizational units (OUs) enable you to group several accounts together and administer them as a single unit instead of one at a time.

Hierarchy

List

Organizational structure

Account created/joined date

▼

Root

r-6nnv

Ashok

management account

533267354313 | Ashokkumarreddy.naguri@gmail.com

Joined 2024/12/06

- Now, we create a IAM group for instructors with non-CLI access which means we can give the ReadOnlyAccess to them.

InstructorsGroup

Info

Delete

Summary

Edit

User group name

InstructorsGroup

Creation time

December 06, 2024, 22:00 (UTC-05:00)

ARN

arn:aws:iam::533267354313:group/InstructorsGroup

Users

Permissions

Access Advisor

Users in this group (2)

Remove

Add users

An IAM user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS.

☐

User name

Instructor1

☐

Instructor2

Groups

1

Last activity

None

Creation time

7 days ago

Users (2)

Permissions

Access Advisor

Permissions policies (1) [Info](#)

Simulate

Remove

Add permissions

Filter by Type

Q Search

All types

<

1

>

<input type="checkbox"/>	Policy name	Type	Attached entities
<input type="checkbox"/>	Non-CLI	Customer inline	0

```
resource "aws_vpc" "main" {
  cidr_block      = "10.0.0.0/16"
  enable_dns_support = true
  enable_dns_hostnames = true
  tags = {
    Name = "MainVPC"
  }
}

# Data Source for Availability Zones
data "aws_availability_zones" "available" {
  state = "available"
}
```

- Creating 3 Subnets of equal sizes which can talk each other

```
# Subnets
resource "aws_subnet" "subnets" {
  count = 3 # Create 3 subnets

  vpc_id          = aws_vpc.main.id
  cidr_block      = cidrsubnet(aws_vpc.main.cidr_block, 8, count.index) # Dynamic CIDR blocks
  availability_zone = element(data.aws_availability_zones.available.names, count.index)
  map_public_ip_on_launch = true

  tags = {
    Name = "Subnet-${count.index + 1}"
  }
}
```

Subnets (3/8) [Info](#)

Find resources by attribute or tag

Last updated 1 minute ago [Refresh](#) [Actions](#) [Create subnet](#)

	Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR	IPv6
<input type="checkbox"/>	subnet-2	subnet-0be32e6aa80dc9593	Available	vpc-0ea00a5a1d0956661 auto...	Off	10.0.1.0/24	-
<input checked="" type="checkbox"/>	Subnet-2	subnet-0899b8bd684dae54b	Available	vpc-0d664b649134ddb9d Mai...	Off	10.0.1.0/24	-
<input checked="" type="checkbox"/>	Subnet-1	subnet-01666c6ce51cdabf2	Available	vpc-0d664b649134ddb9d Mai...	Off	10.0.0.0/24	-
<input checked="" type="checkbox"/>	Subnet-3	subnet-065270bbc8f87b9b2e	Available	vpc-0d664b649134ddb9d Mai...	Off	10.0.2.0/24	-

- Now, We create a single Gateway and attach it to our VPC

```
# Internet Gateway
resource "aws_internet_gateway" "igw" {
  vpc_id = aws_vpc.main.id

  tags = {
    Name = "MainInternetGateway"
  }
}
```

Internet gateways (1/3) [Info](#)

Search

[Refresh](#) [Actions](#) [Create internet gateway](#)

	Name	Internet gateway ID	State	VPC ID	Owner
<input checked="" type="checkbox"/>	MainInternetGateway	igw-07ffe2bc9e45c41b9	Attached	vpc-0d664b649134ddb9d MainVPC	533267354313
<input type="checkbox"/>	main-igw	igw-0ee2e2893440df30e	Attached	vpc-03a90f3dc4adea464 Part02autos...	533267354313

- Create a Routing Table and giving a public route 0.0.0.0 to the internet and also associating this public route to each subnet

```
# Route Table
resource "aws_route_table" "public" {
  vpc_id = aws_vpc.main.id

  route {
    cidr_block = "0.0.0.0/0" # Route for all internet traffic
    gateway_id = aws_internet_gateway.igw.id
  }

  tags = {
    Name = "PublicRouteTable"
  }
}

# Associate Route Table with Subnets
resource "aws_route_table_association" "subnet1" {
  subnet_id      = aws_subnet.subnets[0].id
  route_table_id = aws_route_table.public.id
}

resource "aws_route_table_association" "subnet2" {
  subnet_id      = aws_subnet.subnets[1].id
  route_table_id = aws_route_table.public.id
}

resource "aws_route_table_association" "subnet3" {
  subnet_id      = aws_subnet.subnets[2].id
  route_table_id = aws_route_table.public.id
}
```

Route tables (1/8) Info

Find resources by attribute or tag

Last updated less than a minute ago

Actions

Create route table

< 1 >

Settings

	Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC	Owner ID
<input checked="" type="checkbox"/>	PublicRouteTable	rtb-02ee206f4a6300d76	3 subnets	-	No	vpc-0d664b649134ddb9d Mai...	53326735...
<input type="checkbox"/>	-	rtb-05db0fea235aabf41	-	-	Yes	vpc-02875be5e9591b657 Fina...	53326735...
<input type="checkbox"/>	-	rtb-031fbfd11bb826d9b	-	-	Yes	vpc-00f560255e55fa6c2	53326735...

rtb-02ee206f4a6300d76 / PublicRouteTable

Details

Routes

Subnet associations

Edge associations

Route propagation

Tags

Details

Route table ID
rtb-02ee206f4a6300d76

VPC
vpc-0d664b649134ddb9d | MainVPC

Main
No

Owner ID
533267354313

Explicit subnet associations
3 subnets

Edge associations
-

☒ MainVPC

vpc-0d664b649134ddb9d

Available

Off

10.0.0.0/16

-

dopt-045fd9

VPC Show details

Your AWS virtual network

MainVPC

Subnets (3)

Subnets within this VPC

us-east-1a
Subnet-1

us-east-1b
Subnet-2

us-east-1c
Subnet-3

Route tables (2)

Route network traffic to resources

PublicRouteTable
rtb-0afdc94db322928f

Network connections (1)

Connections to other networks

MainInternetGateway

- Creating a web-security group

```
# Security Group for Web Servers
resource "aws_security_group" "web_sg" {
  vpc_id = aws_vpc.main.id
  name   = "WebSecurityGroup"

  # Ingress Rules (Inbound Traffic)
  ingress {
    from_port = 22
    to_port   = 22
    protocol  = "tcp"
    cidr_blocks = ["0.0.0.0/0"] # Allow SSH from anywhere
  }

  ingress {
    from_port = 80
    to_port   = 80
    protocol  = "tcp"
    cidr_blocks = ["0.0.0.0/0"] # Allow HTTP from anywhere
  }

  ingress {
    from_port = 443
    to_port   = 443
    protocol  = "tcp"
    cidr_blocks = ["0.0.0.0/0"] # Allow HTTPS from anywhere
  }

  ingress {
    from_port = 8080
    to_port   = 8080
    protocol  = "tcp"
    cidr_blocks = ["0.0.0.0/0"] # Allow custom app traffic on port 8080
  }

  # Egress Rules (Outbound Traffic)
  egress {
    from_port = 0
    to_port   = 0
    protocol  = "-1"
    cidr_blocks = ["0.0.0.0/0"] # Allow all outbound traffic
  }

  tags = {
    Name = "WebSecurityGroup"
  }
}
```

Security Groups (1/10)

↻

Actions ▼

Export security groups to CSV ▼

Create security group

Q Find resources by attribute or tag

< 1 > ⚙

<input type="checkbox"/>	Name ▼	Security group ID ▼	Security group name ▼	VPC ID ▼	Description ▼
<input checked="" type="checkbox"/>	WebSecurityGroup	sg-0e7d3858afb80ac07	WebSecurityGroup	vpc-0fec369858fb82c3f	Managed

Inbound rules (4)

↻

Manage tags

Edit inbound rules

Q Search

< 1 > ⚙

<input type="checkbox"/>	Name ▼	Security group rule... ▼	IP version ▼	Type ▼	Protocol ▼	Port range
<input type="checkbox"/>	-	sgr-095fbb7ca8ddf670b	IPv4	SSH	TCP	22
<input type="checkbox"/>	-	sgr-098ace73efb11196b	IPv4	HTTPS	TCP	443
<input type="checkbox"/>	-	sgr-06999e3191a214e...	IPv4	HTTP	TCP	80
<input type="checkbox"/>	-	sgr-09df644f0efa69999	IPv4	Custom TCP	TCP	8080

- Creating an EC2 instance with a standard AMI

```
# Amazon Linux 2 AMI data source
data "aws_ami" "amazon_linux" {
  most_recent = true
  owners      = ["amazon"]

  filter {
    name   = "name"
    values = ["amzn2-ami-hvm-*-x86_64-gp2"]
  }
}

# EC2 Instance with Apache
resource "aws_instance" "web_server" {
  ami            = data.aws_ami.amazon_linux.id
  instance_type  = "t2.micro"
  subnet_id      = aws_subnet.subnets[0].id

  vpc_security_group_ids = [aws_security_group.web_sg.id]

  # User data script to install and start Apache
  user_data = <<-EOF
    #!/bin/bash
    yum update -y
    yum install -y httpd
    systemctl start httpd
    systemctl enable httpd
    echo "<h1>Hello from Group3 of Final Project</h1>" > /var/www/html/index.html

    echo "RDS_USERNAME=${var.rds_username}" >> /etc/environment
    echo "RDS_PASSWORD=${var.rds_password}" >> /etc/environment
    echo "RDS_DBNAME=${var.rds_dbname}" >> /etc/environment
    echo "RDSSHOT_NAME=${var.rdshost_name}" >> /etc/environment

    # Reload environment variables
    source /etc/environment
  EOF

  # You might want to add your key pair name here
  # key_name = "your-key-pair-name"

  tags = {
    Name = "ApacheWebServer"
  }
}
```

```
# Output the public IP of the instance
output "web_server_public_ip" {
  value = aws_instance.web_server.public_ip
}

# Output the public DNS of the instance
output "web_server_public_dns" {
  value = aws_instance.web_server.public_dns
}
```

For the above variables we have created a terraform.tfvars file and store in our local system and not hardcoding the values into the script.

```
rds_username = "admin"
rds_password = "finalproject"
rds_dbname   = "mydb"
rdshost_name = "localhost"
```

Instances (1/2) [Info](#) Last updated less than a minute ago [Connect](#) [Instance state](#) [Actions](#) [Launch instances](#)

Find Instance by attribute or tag (case-sensitive) [Running](#) [< 1 >](#) [Settings](#)

<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
<input checked="" type="checkbox"/>	ApacheWebSe...	i-062530c10e7aeb17f	Running	t2.micro	2/2 checks passed	View alarms	us-east-1a	ec2-44-222
<input type="checkbox"/>	autoscaling-in...	i-0af54e563331cbe36	Running	t2.micro	2/2 checks passed	View alarms	us-east-1c	ec2-18-206

- Creating appropriate dbsecurity groups

```
# Security Group for Database - only allowing traffic from web servers
resource "aws_security_group" "db_sg" {
  vpc_id = aws_vpc.main.id
  name   = "DatabaseSecurityGroup"

  # Ingress rule - allow traffic from web servers only
  ingress {
    from_port = 3306
    to_port   = 3306
    protocol  = "tcp"
    security_groups = [aws_security_group.web_sg.id] # Only allow traffic from web security group
  }

  # Egress Rules
  egress {
    from_port = 0
    to_port   = 0
    protocol  = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }

  tags = {
    Name = "DatabaseSecurityGroup"
  }
}
```

Security Groups (1/11) [Refresh](#) [Actions](#) [Export security groups to CSV](#) [Create security group](#)

Find resources by attribute or tag [< 1 >](#) [Settings](#)

<input type="checkbox"/>	Name	Security group ID	Security group name	VPC ID	Description
<input checked="" type="checkbox"/>	DatabaseSecurityGr...	sg-0590d57894b05fdc1	DatabaseSecurityGroup	vpc-0d664b649134ddb9d	Managed by

sg-0590d57894b05fdc1 - DatabaseSecurityGroup

[Details](#) [Inbound rules](#) [Outbound rules](#) [Sharing - new](#) [VPC associations - new](#) [Tags](#)

Inbound rules (1) [Refresh](#) [Manage tags](#) [Edit inbound rules](#)

Search [< 1 >](#) [Settings](#)

<input type="checkbox"/>	Name	Security group rule...	IP version	Type	Protocol	Port range
<input type="checkbox"/>	-	sgr-049d58b1d6b11acf2	-	MYSQL/Aurora	TCP	3306

Part-02

- Baking a new AMI ID using Ansible

```
---
- name: Create Custom AMI with Apache, Ansible, and Flask
  hosts: localhost
  gather_facts: false
  vars:
    instance_type: t2.micro
    region: us-east-1
    vpc_id: "{{ lookup('env', 'TF_VAR_vpc_id') }}"
    subnet_id: "{{ lookup('env', 'TF_VAR_subnet_id') }}"
    security_group_id: "{{ lookup('env', 'TF_VAR_security_group_id') }}"
    ami_name: "final-Project-{{ lookup('pipe', 'date +%Y%m%d%H%M%S') }}"
    base_ami: "ami-0453ec754f44f9a4a"

  tasks:
    - name: Launch EC2 instance
      amazon.aws.ec2_instance:
        name: "temp-instance-for-ami"
        instance_type: "{{ instance_type }}"
        image_id: "{{ base_ami }}"
        region: "{{ region }}"
        vpc_subnet_id: "{{ subnet_id }}"
        security_group: "{{ security_group_id }}"
        network:
          assign_public_ip: true
        wait: yes
        state: present
        user_data: |
          #!/bin/bash
          sudo yum update -y
          sudo yum install -y python3 python3-pip
          sudo yum groupinstall -y "Development Tools"
          sudo yum install -y python3-devel
          sudo pip3 install ansible flask flask-sqlalchemy flask-migrate mysql-connector-python pymysql psutil
          sudo yum install -y httpd
          sudo systemctl start httpd
          sudo systemctl enable httpd
          echo "<h1>Hello from Group4 of Final Project</h1>" | sudo tee /var/www/html/index.html
```

- Installing all necessary packages and also included the If the CPU usage increases over 90% then the script should stop.

```
---
- name: Create Custom AMI with Apache, Ansible, and Flask
  hosts: localhost
  gather_facts: false
  vars:
    instance_type: t2.micro
    region: us-east-1
    vpc_id: "{{ lookup('env', 'TF_VAR_vpc_id') }}"
    subnet_id: "{{ lookup('env', 'TF_VAR_subnet_id') }}"
    security_group_id: "{{ lookup('env', 'TF_VAR_security_group_id') }}"
    ami_name: "final-Project-{{ lookup('pipe', 'date +%Y%m%d%H%M%S') }}"
    base_ami: "ami-0453ec754f44f9a4a"

  tasks:
    - name: Launch EC2 instance
      amazon.aws.ec2_instance:
        name: "temp-instance-for-ami"
        instance_type: "{{ instance_type }}"
        image_id: "{{ base_ami }}"
        region: "{{ region }}"
        vpc_subnet_id: "{{ subnet_id }}"
        security_group: "{{ security_group_id }}"
        network:
          assign_public_ip: true
        wait: yes
        state: present
        user_data: |
          #!/bin/bash
          sudo yum update -y
          sudo yum install -y python3 python3-pip
          sudo yum groupinstall -y "Development Tools"
          sudo yum install -y python3-devel
          sudo pip3 install ansible flask flask-sqlalchemy flask-migrate mysql-connector-python pymysql psutil
          sudo yum install -y httpd
          sudo systemctl start httpd
          sudo systemctl enable httpd
          echo "<h1>Hello from Group4 of Final Project</h1>" | sudo tee /var/www/html/index.html
```



```

# Start Python's built-in HTTP server
nohup python3 -m http.server 80 --directory /var/www/html &
# CPU Load Simulation Script
nohup bash -c '
while true; do
    # Start stress to load CPU for 30 seconds
    stress --cpu 2 --timeout 30 &
    PID=$!
    sleep 30
    kill $PID

    # Get current CPU utilization
    CURRENT_CPU=$(top -bn1 | grep "Cpu(s)" | \
    sed "s/./, *([0-9.]*%)* id.*/\1/" | \
    awk '{print 100 - $1}')

    # Check if CPU utilization exceeds 90%
    CPU_EXCEEDS=$(echo "$CURRENT_CPU > 90" | bc -l)

    if [ "$CPU_EXCEEDS" -eq 1 ]; then
        # Stop the loop if CPU usage is over 90%
        break
    fi

    # Maintain load for 30 seconds
    stress --cpu 2 --timeout 30 &
    sleep 30
    kill $!
done' &
# Set environment variables for RDS
sudo echo "RDS_USERNAME=${var.rds_username}" >> /etc/environment
sudo echo "RDS_PASSWORD=${var.rds_password}" >> /etc/environment
sudo echo "RDS_DBNAME=${var.rds_dbname}" >> /etc/environment
sudo echo "RDSHOST_NAME=${var.rdshost_name}" >> /etc/environment

```

register: ec2

- name: Wait for user data script to complete

pause:

minutes: 2

- name: Create AMI

amazon.aws.ec2_ami:

```

instance_id: "{{ ec2.instances[0].instance_id }}"
name: "{{ ami_name }}"

```

```

name: "{{ ami_name }}"
region: "{{ region }}"
wait: yes
tags:
    Name: "{{ ami_name }}"
    Description: "AMI with Ansible, Apache, and Flask pre-installed"
register: ami

```

- name: Terminate temporary instance

amazon.aws.ec2_instance:

```

instance_ids: "{{ ec2.instances[0].instance_id }}"
region: "{{ region }}"
state: absent
wait: yes

```

- name: Save AMI ID to a file

copy:

```

content: "{{ ami.image_id }}"
dest: "/mnt/d/ami_id.txt"

```

- name: Output AMI ID

debug:

```

msg: "Created AMI with ID: {{ ami.image_id }}"

```

```

root@LAPTOP-2CSMSCDE:/mnt/d/Test folder for Infrastructure Automation and Tools/Project/part02# ansible-playbook playbook-ami.yaml
[WARNING]: No inventory was parsed, only implicit localhost is available
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'

PLAY [Create Custom AMI with Apache, Ansible, and Flask] *****

TASK [Launch EC2 instance] *****
[DEPRECATION WARNING]: The network parameter has been deprecated, please use network_interfaces and/or network_interfaces_ids instead. This
Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
changed: [localhost]

TASK [Wait for user data script to complete] *****
Pausing for 60 seconds
(ctrl+C then 'C' = continue early, ctrl+C then 'A' = abort)
ok: [localhost]

TASK [Create AMI] *****
changed: [localhost]

TASK [Terminate temporary instance] *****
changed: [localhost]

TASK [Save AMI ID to a file] *****
changed: [localhost]

TASK [Output AMI ID] *****
ok: [localhost] => {
  "msg": "Created AMI with ID: ami-09f85dc11d55053aa"
}

PLAY RECAP *****
localhost                : ok=6    changed=4    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

```

- Using Terraform to create a VPC with 3 subnets

```

# AWS Provider Configuration
provider "aws" {
  region = "us-east-1" # Update with your desired region
}

data "aws_availability_zones" "available" {
  state = "available"
}

variable "instance_type" {
  description = "Type of EC2 instance"
}

variable "key_name" {
  description = "Name of the SSH key pair"
}

data "aws_ami" "latest_ami" {
  most_recent = true
  filter {
    name = "name"
    values = ["amzn2-ami-hvm-*-x86_64-gp2"]
  }
  owners = ["amazon"]
}

# VPC Configuration
resource "aws_vpc" "main" {
  cidr_block = "10.0.0.0/16"
  enable_dns_support = true
  enable_dns_hostnames = true
  tags = { Name = "Part02autoscaling-vpc" }
}

# Subnet Configuration
resource "aws_subnet" "subnets" {
  count = 3 # Create three subnets in different Availability Zones
  vpc_id = aws_vpc.main.id
  cidr_block = cidrsubnet(aws_vpc.main.cidr_block, 8, count.index)
  availability_zone = data.aws_availability_zones.available.names[count.index]
  map_public_ip_on_launch = true

  tags = {

```

```

    Name = "subnet-${count.index + 1}"
  }
}

# Internet Gateway
resource "aws_internet_gateway" "main" {
  vpc_id = aws_vpc.main.id
  tags   = { Name = "main-igw" }
}

# Route Table
resource "aws_route_table" "public_rt" {
  vpc_id = aws_vpc.main.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.main.id
  }
}

# Associate Route Table with Subnet
resource "aws_route_table_association" "subnet_association" {
  count          = length(aws_subnet.subnets)
  subnet_id      = aws_subnet.subnets[count.index].id
  route_table_id = aws_route_table.public_rt.id
}

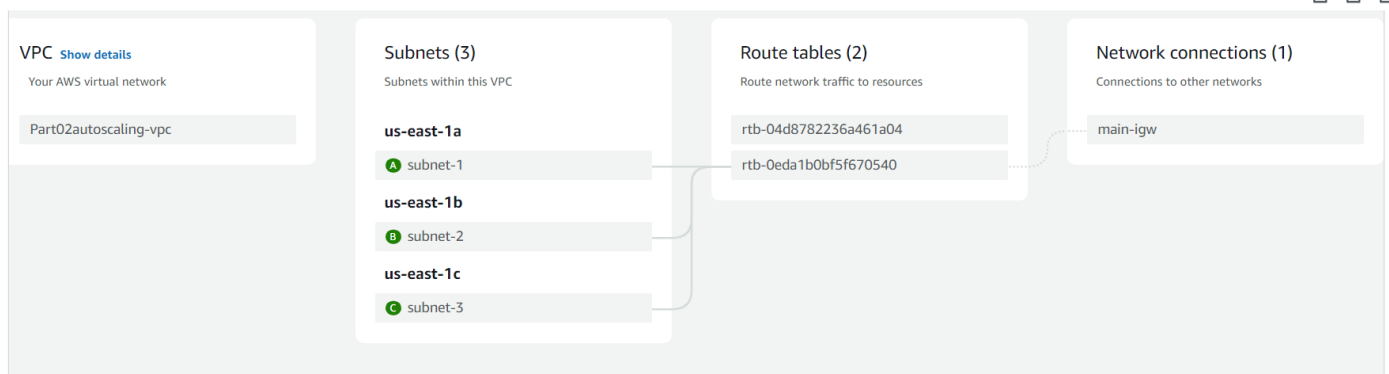
```

Your VPCs (1/5) [Info](#)

Search

Last updated 1 minute ago [Refresh](#) [Actions](#) [Create VPC](#)

	Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR	DHCP opti...
<input checked="" type="checkbox"/>	Part02autoscaling-vpc	vpc-03a90f3dc4adea464	Available	Off	10.0.0.0/16	-	dopt-045fd
<input type="checkbox"/>	MainVPC	vpc-0d664b649134ddb9d	Available	Off	10.0.0.0/16	-	dopt-045fd



Subnets (3/8) [Info](#)

Find resources by attribute or tag

Last updated 1 minute ago [Refresh](#) [Actions](#) [Create subnet](#)

	Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
<input type="checkbox"/>	FinalProject	subnet-0d5909dcac29fb329	Available	vpc-02875be5e9591b657 FinalProject	Off	10.0.0.0/16
<input checked="" type="checkbox"/>	subnet-2	subnet-04a00e5a5010550fa	Available	vpc-03a90f3dc4adea464 Part02autoscaling-vpc	Off	10.0.1.0/24
<input checked="" type="checkbox"/>	subnet-1	subnet-0577557b0c1de1654	Available	vpc-03a90f3dc4adea464 Part02autoscaling-vpc	Off	10.0.0.0/24
<input checked="" type="checkbox"/>	subnet-3	subnet-0b6c079b9414e2b9c	Available	vpc-03a90f3dc4adea464 Part02autoscaling-vpc	Off	10.0.2.0/24

Subnets: [subnet-0577557b0c1de1654](#), [subnet-0b6c079b9414e2b9c](#), [subnet-04a00e5a5010550fa](#)

```
# Security Group
resource "aws_security_group" "web_sg" {
  vpc_id = aws_vpc.main.id
  ingress {
    from_port = 80
    to_port   = 80
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  ingress {
    from_port = 22
    to_port   = 22
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  # Allow HTTPS (port 443) traffic
  ingress {
    from_port = 443
    to_port   = 443
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }

  # Allow custom HTTP (port 8080) traffic
  ingress {
    from_port = 8080
    to_port   = 8080
    protocol = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
  egress {
    from_port = 0
    to_port   = 0
    protocol = "-1"
    cidr_blocks = ["0.0.0.0/0"]
  }
  tags = { Name = "web-sg" }
}
```

```
resource "aws_launch_template" "app_lt" {
  name_prefix = "app-launch-template-"
  image_id    = data.aws_ami.latest_ami.id
  instance_type = var.instance_type
  key_name    = var.key_name

  network_interfaces {
    security_groups = [aws_security_group.web_sg.id]
    associate_public_ip_address = true
  }

  lifecycle {
    create_before_destroy = true
  }

  tags = {
    Name = "app-launch-template"
  }
}
```

Instances (1/2) Info

Last updated 6 minutes ago

Connect

Instance state

Actions

Launch instances

Find Instance by attribute or tag (case-sensitive)

Running

< 1 >

<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
<input type="checkbox"/>	ApacheWebSe...	i-062530c10e7aeb17f	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1a	ec2-44-222
<input checked="" type="checkbox"/>	autoscaling-in...	i-0af54e563331cbe36	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1c	ec2-18-206

- Creating Auto Scaling Group

```
# Auto Scaling Group
resource "aws_autoscaling_group" "app_asg" {
  launch_template {
    id      = aws_launch_template.app_lt.id
    version = "$Latest"
  }
  min_size      = 1
  max_size      = 5
  desired_capacity = 1
  vpc_zone_identifier = aws_subnet.subnets[*].id
  health_check_type = "EC2"
  health_check_grace_period = 300

  tag {
    key      = "Name"
    value    = "autoscaling-instance"
    propagate_at_launch = true
  }
}

# Scale Up Policy
resource "aws_autoscaling_policy" "scale_up" {
  name                = "scale-up-policy"
  scaling_adjustment  = 1
  adjustment_type     = "ChangeInCapacity"
  autoscaling_group_name = aws_autoscaling_group.app_asg.name
}

# Scale Down Policy
resource "aws_autoscaling_policy" "scale_down" {
  name                = "scale-down-policy"
  scaling_adjustment  = -1
  adjustment_type     = "ChangeInCapacity"
  autoscaling_group_name = aws_autoscaling_group.app_asg.name
}
```

Auto Scaling groups (1) [Info](#)


[Launch configurations](#)
[Launch templates](#)
[Actions](#)
[Create Auto Scaling group](#)

< 1 >

<input type="checkbox"/>	Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Ma
<input type="checkbox"/>	terraform-20241214112905013300000005	app-launch-template-202412141128567	1	-	1	1	5

- Similarly, created a Cloudwatch alarm for high and low cpu usage

```
# CloudWatch Alarm for Scale Up
resource "aws_cloudwatch_metric_alarm" "scale_up_alarm" {
  alarm_name         = "scale-up-cpu-utilization"
  comparison_operator = "GreaterThanThreshold"
  evaluation_periods  = 2
  metric_name         = "CPUUtilization"
  namespace           = "AWS/EC2"
  period              = 60
  statistic            = "Average"
  threshold           = 70
  alarm_actions       = [aws_autoscaling_policy.scale_up.arn]
  dimensions = {
    AutoScalingGroupName = aws_autoscaling_group.app_asg.name
  }
}

# CloudWatch Alarm for Scale Down
resource "aws_cloudwatch_metric_alarm" "scale_down_alarm" {
  alarm_name         = "scale-down-cpu-utilization"
  comparison_operator = "LessThanThreshold"
  evaluation_periods  = 2
  metric_name         = "CPUUtilization"
  namespace           = "AWS/EC2"
  period              = 60
  statistic            = "Average"
  threshold           = 40
  alarm_actions       = [aws_autoscaling_policy.scale_down.arn]
  dimensions = {
    AutoScalingGroupName = aws_autoscaling_group.app_asg.name
  }
}
```

```
run aws iam update --cli-input-json file.json
[ec2-user@ip-10-0-0-177 ~]$ ls
cpu_load.sh
[ec2-user@ip-10-0-0-177 ~]$ sudo ./cpu_load.sh
Starting CPU Load Test
stress: info: [4321] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
Current CPU Usage: 0%
./cpu_load.sh: line 17: 4321 Terminated                  stress --cpu 2 --timeout 30
stress: info: [4333] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
```

```
Running transaction check
Running transaction test
top - 11:40:37 up 11 min,  1 user,  load average: 1.13, 0.42, 0.15
Tasks:  95 total,   3 running,  55 sleeping,   0 stopped,   0 zombie
top - 11:41:43 up 12 min,  2 users,  load average: 0.75, 0.43, 0.17
Tasks: 101 total,   3 running,  61 sleeping,   0 stopped,   0 zombie
%Cpu(s): 99.0 us,  0.0 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.0 si,  1.0 st
KiB Mem :  993492 total,  438160 free,   87716 used,  467616 buff/cache
KiB Swap:   0 total,    0 free,    0 used.  762156 avail Mem

 3896 root      20   0   7580    96    0 R 49.7  0.0   0:13.95 stress
 4323 root      20   0   7580   100    0 R 49.8  0.0   0:06.40 stress
 4324 root      20   0   7580   100    0 R 49.8  0.0   0:06.40 stress
   1 root       20   0 123600  5544  3920 S  0.0  0.6   0:00.74 systemd
   2 root       20   0    0      0    0 S  0.0  0.0   0:00.00 kthreadd
   3 root       20   0    0      0    0 I  0.0  0.0   0:00.04 kworker/0:0
   4 root       0 -20   0      0    0 I  0.0  0.0   0:00.00 kworker/0:0H
   6 root       0 -20   0      0    0 I  0.0  0.0   0:00.00 mm_percpu_wq
   7 root       20   0    0      0    0 S  0.0  0.0   0:00.15 ksoftirqd/0
   8 root       20   0    0      0    0 I  0.0  0.0   0:00.04 rcu_sched
   9 root       20   0    0      0    0 I  0.0  0.0   0:00.00 rcu_bh
  10 root      rt  0    0      0    0 S  0.0  0.0   0:00.00 migration/0
  11 root      rt  0    0      0    0 S  0.0  0.0   0:00.00 watchdog/0
  12 root       20   0    0      0    0 S  0.0  0.0   0:00.00 cpuhp/0
  14 root       20   0    0      0    0 S  0.0  0.0   0:00.00 kdevtmpfs
```

Alarms (2) ☐ Hide Auto Scaling alarms

Clear selection

Create composite alarm

Actions

Create alarm

Alarm state: Any

Alarm type: Any

Actions status: Any

< 1 >

<input type="checkbox"/>	Name	State	Last state update (UTC)	Conditions	Actions
<input type="checkbox"/>	scale-down-cpu-utilization	In alarm	2024-12-14 11:53:15	CPUUtilization < 40 for 2 datapoints within 2 minutes	Actions enabled
<input type="checkbox"/>	scale-up-cpu-utilization	OK	2024-12-14 11:50:14	CPUUtilization > 70 for 2 datapoints within 2 minutes	Actions enabled