

❖ Multi-Cloud DR Project Using (AWS & AZURE Cloud)

Google Drive Link :

<https://drive.google.com/drive/folders/1UG6wh9vMlNgzJObVn-CCHhmcVBnrm0SW?usp=sharing>

GitHub Link :

<https://github.com/sagarpatilbox/Multi-Cloud-DR-UpGrad-Project>

❖ On your **local machine** Install

- Install:
 - Terraform
 - AWS CLI
 - Azure CLI

1. Install Terraform on Windows

Step 1 — Download Terraform

1. Go to: <https://developer.hashicorp.com/terraform/downloads>
2. Under **Windows**, download:
64-bit Terraform ZIP

Step 2 — Extract Terraform

1. Extract the ZIP file.
2. Move **terraform.exe** to a folder such as:

C:\terraform

Step 3 — Test installation

- Open PowerShell and run:

```
terraform -version
```

- You should see output like:

```
Terraform v1.x.x
```

```
Command Prompt  
Microsoft Windows [Version 10.0.19044.6575]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\admin>terraform --version  
Terraform v1.14.0  
on windows_amd64  
  
C:\Users\admin>
```

2. Install AWS CLI on Windows

Step 1 — Download AWS CLI

Download 64-bit Windows installer: <https://awscli.amazonaws.com/AWSCLIV2.msi>

Step 2 — Run Installer

1. Double-click **AWSCLIV2.msi**
2. Follow prompts → Finish

Step 3 — Verify installation

- Open PowerShell:

```
aws --version
```

- Expected output:

```
aws-cli/2.x.x Python/3.x
```

```
PS C:\Users\admin> aws --version  
aws-cli/2.32.3 Python/3.13.9 Windows/10 exe/AMD64  
PS C:\Users\admin>
```

3. Install Azure CLI on Windows

Step 1 — Download Azure CLI

Go to: <https://aka.ms/installazurecliwindows>

This downloads `AzureCLI.msi`.

Step 2 — Run installer

Follow wizard → Install

Step 3 — Verify installation

- Open PowerShell:

```
az --version
```

- You should see:

```
azure-cli 2.x.x
```

```
PS C:\Users\admin> az --version
azure-cli                  2.80.0
core                      2.80.0
telemetry                 1.1.0
Dependencies:
msal                      1.34.0b1
azure-mgmt-resource        23.3.0

Python location 'C:\Program Files (x86)\Microsoft SDKs\Azure\CLI2\python.exe'
Config directory 'C:\Users\admin\.azure'
Extensions directory 'C:\Users\admin\.azure\cliextensions'

Python (Windows) 3.13.9 (tags/v3.13.9:8183fa5, Oct 14 2025, 14:00:05) [MSC v.1944 32 bit (Intel)]
Legal docs and information: aka.ms/AzureCliLegal

Your CLI is up-to-date.
```



On your local machine Install

Using Git Bash (Recommended for Terraform & SSH)

Step 1 — Open Git Bash

- Install Git for Windows: <https://git-scm.com/download/win>

- Then open **Git Bash**.

Step 2 — Run the same SSH keygen command

```
ssh-keygen -t rsa -b 4096 -C "multi-cloud-dr" -f ~/.ssh/mcdr_key
```

Your key files will be created at:

```
C:\Users\<YourUsername>\.ssh\mcdr_key
C:\Users\<YourUsername>\.ssh\mcdr_key.pub
```

- `mcdr_key` → private key
- `mcdr_key.pub` → public key

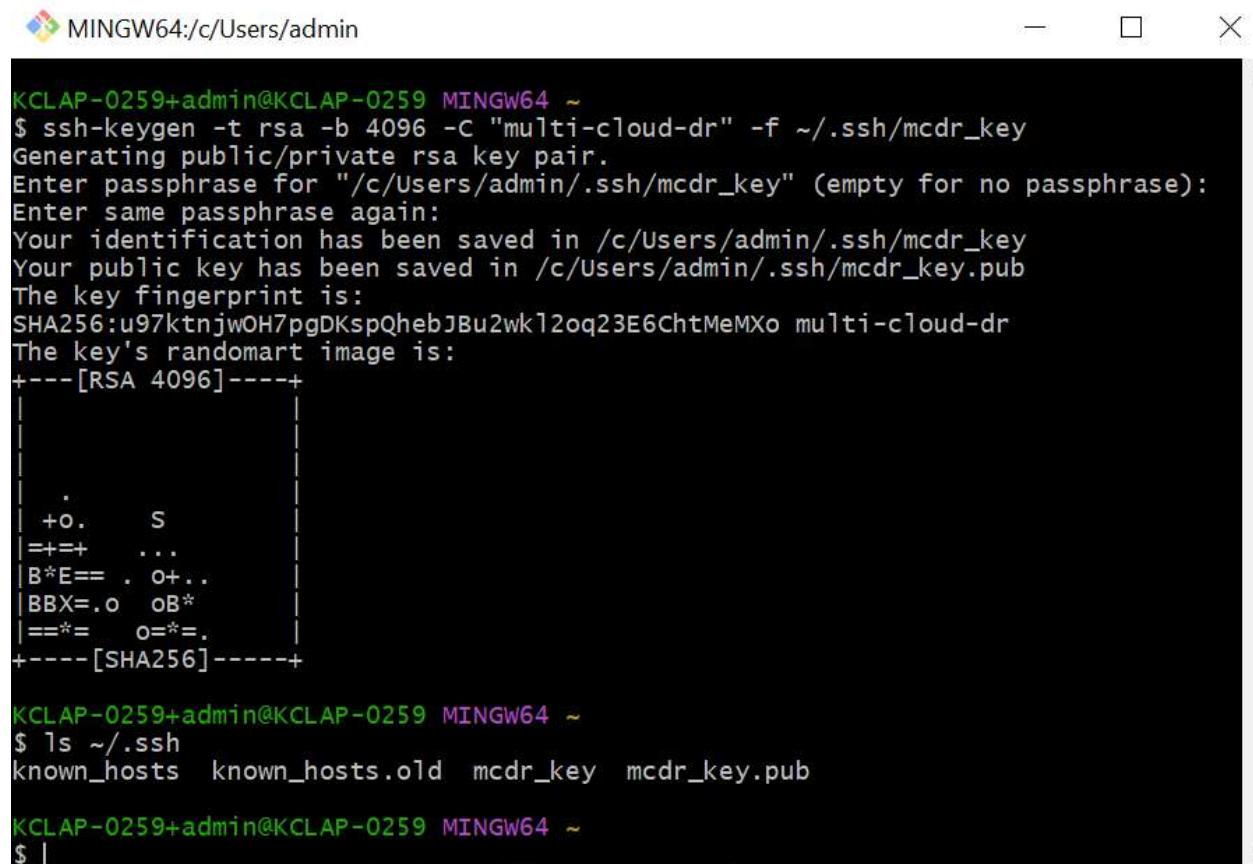
Step 3 — Verify keys

- In Git Bash:

```
ls ~/.ssh
```

- You should see:

```
mcdr_key  mcdr_key.pub
```



The screenshot shows a terminal window titled 'MINGW64:/c/Users/admin'. The terminal output is as follows:

```
KCLAP-0259+admin@KCLAP-0259 MINGW64 ~
$ ssh-keygen -t rsa -b 4096 -C "multi-cloud-dr" -f ~/.ssh/mcdr_key
Generating public/private rsa key pair.
Enter passphrase for "/c/Users/admin/.ssh/mcdr_key" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/admin/.ssh/mcdr_key
Your public key has been saved in /c/Users/admin/.ssh/mcdr_key.pub
The key fingerprint is:
SHA256:u97ktnjwOH7pgDKspQhebJBu2wk12oq23E6chtMeMXo multi-cloud-dr
The key's randomart image is:
+---[RSA 4096]---+
| . . . . . . . . . |
| . . . . . . . . . |
| . . . . . . . . . |
| . . . . . . . . . |
| . . . . . . . . . |
| . . . . . . . . . |
| . . . . . . . . . |
| . . . . . . . . . |
| . . . . . . . . . |
+---[SHA256]-----+
KCLAP-0259+admin@KCLAP-0259 MINGW64 ~
$ ls ~/.ssh
known_hosts  known_hosts.old  mcdr_key  mcdr_key.pub

KCLAP-0259+admin@KCLAP-0259 MINGW64 ~
$ |
```

Task 1 – Infrastructure Provisioning (AWS & Azure)

1.1 AWS – VPC, Subnets, EC2 (App + Tools)

1.1.1 Create folder & files

```
mkdir -p project/aws
cd project/aws
```

Create `variables.tf`:

```
variable "aws_region" {
  description = "AWS region"
  type        = string
  default     = "us-east-1"
}

variable "project_name" {
  description = "Project name prefix"
  type        = string
  default     = "multi-cloud-dr"
}

variable "public_key_path" {
  description = "Path to your SSH public key"
  type        = string
}
```

Create `main.tf`:

```
terraform {
  required_version = ">= 1.3.0"

  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}

provider "aws" {
  region = var.aws_region
}

# VPC
resource "aws_vpc" "main" {
  cidr_block          = "10.0.0.0/16"
  enable_dns_support = true
  enable_dns_hostnames = true
```

```

tags = {
    Name = "${var.project_name}-vpc"
}
}

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

    tags = {
        Name = "${var.project_name}-igw"
    }
}

# Public subnets
resource "aws_subnet" "public_1" {
    vpc_id           = aws_vpc.main.id
    cidr_block       = "10.0.1.0/24"
    availability_zone = "${var.aws_region}a"
    map_public_ip_on_launch = true
    tags = {
        Name = "${var.project_name}-public-1"
    }
}

resource "aws_subnet" "public_2" {
    vpc_id           = aws_vpc.main.id
    cidr_block       = "10.0.2.0/24"
    availability_zone = "${var.aws_region}b"
    map_public_ip_on_launch = true
    tags = {
        Name = "${var.project_name}-public-2"
    }
}

# Private subnets
resource "aws_subnet" "private_1" {
    vpc_id           = aws_vpc.main.id
    cidr_block       = "10.0.11.0/24"
    availability_zone = "${var.aws_region}a"
    tags = {
        Name = "${var.project_name}-private-1"
    }
}

resource "aws_subnet" "private_2" {
    vpc_id           = aws_vpc.main.id
    cidr_block       = "10.0.12.0/24"
    availability_zone = "${var.aws_region}b"
    tags = {
        Name = "${var.project_name}-private-2"
    }
}

# NAT
resource "aws_eip" "nat_eip" {
    domain = "vpc"
}

```

```

tags = { Name = "${var.project_name}-nat-eip" }

resource "aws_nat_gateway" "nat" {
  allocation_id = aws_eip.nat_eip.id
  subnet_id     = aws_subnet.public_1.id
  depends_on    = [aws_internet_gateway.igw]

  tags = {
    Name = "${var.project_name}-nat"
  }
}

# Public route table
resource "aws_route_table" "public" {
  vpc_id = aws_vpc.main.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.igw.id
  }

  tags = {
    Name = "${var.project_name}-public-rt"
  }
}

resource "aws_route_table_association" "public_1" {
  subnet_id      = aws_subnet.public_1.id
  route_table_id = aws_route_table.public.id
}

resource "aws_route_table_association" "public_2" {
  subnet_id      = aws_subnet.public_2.id
  route_table_id = aws_route_table.public.id
}

# Private route table
resource "aws_route_table" "private" {
  vpc_id = aws_vpc.main.id

  route {
    cidr_block      = "0.0.0.0/0"
    nat_gateway_id = aws_nat_gateway.nat.id
  }

  tags = {
    Name = "${var.project_name}-private-rt"
  }
}

resource "aws_route_table_association" "private_1" {
  subnet_id      = aws_subnet.private_1.id
  route_table_id = aws_route_table.private.id
}

resource "aws_route_table_association" "private_2" {

```

```

    subnet_id      = aws_subnet.private_2.id
    route_table_id = aws_route_table.private.id
}

# Security group
resource "aws_security_group" "web_sg" {
    name          = "${var.project_name}-web-sg"
    description   = "Allow SSH and HTTP"
    vpc_id        = aws_vpc.main.id

    ingress {
        description = "SSH"
        from_port   = 22
        to_port     = 22
        protocol    = "tcp"
        cidr_blocks = ["0.0.0.0/0"]
    }

    ingress {
        description = "HTTP"
        from_port   = 80
        to_port     = 80
        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 = "${var.project_name}-web-sg"
    }
}

# Key pair using your public key
resource "aws_key_pair" "default" {
    key_name      = "${var.project_name}-key"
    public_key    = file(var.public_key_path)
}

# App instance
resource "aws_instance" "app" {
    ami           = "ami-0ecb62995f68bb549"
    instance_type = "t3.micro"
    subnet_id     = aws_subnet.public_1.id
    vpc_security_group_ids = [aws_security_group.web_sg.id]
    key_name      = aws_key_pair.default.key_name
    associate_public_ip_address = true

    tags = {
        Name = "${var.project_name}-app"
        Role = "app"
    }
}

```

```

        }
    }

# Tools instance
resource "aws_instance" "tools" {
    ami                      = "ami-0ecb62995f68bb549"
    instance_type             = "t3.micro"
    subnet_id                 = aws_subnet.public_2.id
    vpc_security_group_ids   = [aws_security_group.web_sg.id]
    key_name                  = aws_key_pair.default.key_name
    associate_public_ip_address = true

    tags = {
        Name = "${var.project_name}-tools"
        Role = "tools"
    }
}

```

Create **outputs.tf**:

```

output "app_public_ip" {
    value      = aws_instance.app.public_ip
    description = "Public IP of the App Machine"
}

output "tools_public_ip" {
    value      = aws_instance.tools.public_ip
    description = "Public IP of the Tools Machine"
}

```

1.1.2 Configure & Run Terraform

First Configure aws Login :

```
aws configure
```

Give Below Information :

```
AWS Access Key ID: <from CSV>
AWS Secret Access Key: <from CSV>
Default region name: us-east-1
Default output format: json
```

Run Terraform

```

cd project/aws
terraform init
terraform plan -var="public_key_path=$HOME/.ssh/mcdr_key.pub"
terraform apply -var="public_key_path=$HOME/.ssh/mcdr_key.pub"

```

```

PS C:\Users\admin\project\aws> terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.100.0

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\admin\project\aws>

```

❖ Screenshots to capture:

- Terminal with `terraform plan` summary (AWS).

```

+ cidr_block                  = "10.0.0.0/16"
+ default_network_acl_id     = (known after apply)
+ default_route_table_id      = (known after apply)
+ default_security_group_id   = (known after apply)
+ dhcp_options_id             = (known after apply)
+ enable_dns_hostnames        = true
+ enable_dns_support          = true
+ enable_network_address_usage_metrics = (known after apply)
+ id                          = (known after apply)
+ instance_tenancy            = "default"
+ ipv6_association_id         = (known after apply)
+ ipv6_cidr_block              = (known after apply)
+ ipv6_cidr_block_network_border_group = (known after apply)
+ main_route_table_id          = (known after apply)
+ owner_id                     = (known after apply)
+ tags                         = {
    + "Name" = "multi-cloud-dr-vpc"
}
+ tags_all                     = {
    + "Name" = "multi-cloud-dr-vpc"
}
}

Plan: 18 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ app_public_ip    = (known after apply)
+ tools_public_ip = (known after apply)

```

- Terminal with `terraform apply` success.

```

aws_route_table_association.public_2: Creation complete after 1s [id=rtbassoc-00d5857dbe291c9f7]
aws_route_table_association.public_1: Creation complete after 1s [id=rtbassoc-0e962ca424dbf3c3d]
aws_instance.tools: Still creating... [00m10s elapsed]
aws_nat_gateway.nat: Still creating... [00m10s elapsed]
aws_instance.app: Still creating... [00m10s elapsed]
aws_instance.tools: Creation complete after 16s [id=i-06057e17c07065325]
aws_instance.app: Creation complete after 16s [id=i-08552db2f4b74d714]
aws_nat_gateway.nat: Still creating... [00m20s elapsed]
aws_nat_gateway.nat: Still creating... [00m30s elapsed]
aws_nat_gateway.nat: Still creating... [00m40s elapsed]
aws_nat_gateway.nat: Still creating... [00m50s elapsed]
aws_nat_gateway.nat: Still creating... [01m00s elapsed]
aws_nat_gateway.nat: Still creating... [01m10s elapsed]
aws_nat_gateway.nat: Still creating... [01m20s elapsed]
aws_nat_gateway.nat: Still creating... [01m30s elapsed]
aws_nat_gateway.nat: Creation complete after 1m39s [id=nat-07e03dc3f9feb978f]
aws_route_table.private: Creating...
aws_route_table.private: Creation complete after 3s [id=rtb-08f15f4aa6a2d9d40]
aws_route_table_association.private_2: Creating...
aws_route_table_association.private_1: Creating...
aws_route_table_association.private_2: Creation complete after 1s [id=rtbassoc-0b9bd517e6e6d2fa6]
aws_route_table_association.private_1: Creation complete after 2s [id=rtbassoc-02bc13545f92ec502]

Apply complete! Resources: 18 added, 0 changed, 0 destroyed.

Outputs:

app_public_ip = "35.171.158.150"
tools_public_ip = "54.144.200.231"
PS C:\Users\admin\project\aws>

```

- AWS Console:
 - VPC details

The screenshot shows the AWS VPC console interface. The top navigation bar includes links for CCEP, Outskill, Google Security Op., and All Bookmarks. The main header shows the account ID: 9801-0457-6357 and the region: United States (N. Virginia). The user is Sagarpatl. The left sidebar has sections for VPC dashboard, Virtual private cloud (with subnets, route tables, Internet gateways, etc.), and Security (Network ACLs, Security groups). The main content area is titled 'Your VPCs' and shows a table with one row:

Name	VPC ID	State	Encryption c...	Encryption control ...
multi-cloud-dr-vpc	vpc-09549e9ef81f0e0e3	Available	-	-

Below the table, there's a note: 'Select a VPC above'.

- Subnets listing (2 public, 2 private)

Subnets (4) Info

Name	Subnet ID	State	VPC	Block Public
multi-cloud-dr-private-1	subnet-0d7a9ae98f90fd9c	Available	vpc-09349e9ef81f0e0e3 multi...	Off
multi-cloud-dr-public-1	subnet-07f2c0cc0ecce71f	Available	vpc-09349e9ef81f0e0e3 multi...	Off
multi-cloud-dr-public-2	subnet-0c54985c29894df0f	Available	vpc-09349e9ef81f0e0e3 multi...	Off
multi-cloud-dr-private-2	subnet-0e571639f92a2f60a	Available	vpc-09349e9ef81f0e0e3 multi...	Off

- Route tables (public + private)

Route tables (3) Info

Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
multi-cloud-dr-public-rt	rtb-09af94d912efedd2e	2 subnets	-	No	vpc
multi-cloud-dr-private-rt	rtb-08f15f4aa6a2d9d40	2 subnets	-	No	vpc
-	rtb-0108f67fe55e1cd11	-	-	Yes	vpc

- o NAT gateway

The screenshot shows the AWS VPC console with the URL us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#NatGateways. The left sidebar is expanded to show the 'Virtual private cloud' section, specifically the 'NAT gateways' option under 'Your VPCs'. The main content area displays a table titled 'NAT gateways (1)'. The table has columns for Name, NAT gateway ID, Connectivity..., State, State message, and Availability A single row is listed: 'multi-cloud-dr-nat' with ID 'nat-07e03dc3f9feb978f', State 'Public' and 'Available', and 'Zonal' availability.

Name	NAT gateway ID	Connectivity...	State	State message	Availability ...
multi-cloud-dr-nat	nat-07e03dc3f9feb978f	Public	Available	-	Zonal

- o Security group inbound rules (22, 80)

The screenshot shows the AWS VPC console with the URL us-east-1.console.aws.amazon.com/vpcconsole/home?region=us-east-1#SecurityGroup:groupId=sg-058200bbb491abe9a. The left sidebar is expanded to show the 'Virtual private cloud' section, specifically the 'Security groups' option under 'Your VPCs'. The main content area shows the details for the security group 'sg-058200bbb491abe9a - multi-cloud-dr-web-sg'. It includes fields for Security group name ('multi-cloud-dr-web-sg'), Security group ID ('sg-058200bbb491abe9a'), Description ('Allow SSH and HTTP'), and VPC ID ('vpc-09349e9ef81f0e0e3'). Below this, the 'Inbound rules' tab is selected, showing two entries:

Name	Security group rule ID	IP version	Type	Protocol	Port range
-	sgr-02cb7e6fd61efd30e	IPv4	HTTP	TCP	80
-	sgr-0d11a01552b3de15d	IPv4	SSH	TCP	22

- o EC2 instances list showing App + Tools machines

The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays a table titled "Instances (2) Info" with the following data:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public
multi-cloud-dr-tools	i-06057e17c07065325	Running	t3.micro	3/3 checks passed	View alarms	us-east-1b	ec2-5
multi-cloud-dr-app	i-08552db2f4b74d714	Running	t3.micro	3/3 checks passed	View alarms	us-east-1a	ec2-3

Below the table, a message says "Select an instance".

- o Internet Gateway

The screenshot shows the AWS VPC Internet Gateways page. The left sidebar is collapsed. The main area displays a table titled "Internet gateways (1) Info" with the following data:

Name	Internet gateway ID	State	VPC ID
multi-cloud-dr-igw	igw-0059439dc08cf78ce	Attached	vpc-09349e9ef81f0e0e3 multi-cloud-...

Below the table, a message says "Select an internet gateway above".

1.1.3 SSH test to EC2s

From local:

```
ssh -i ~/.ssh/mcdr_key ubuntu@<AWS_APP_PUBLIC_IP>
ssh -i ~/.ssh/mcdr_key ubuntu@<AWS_TOOLS_PUBLIC_IP>
```

Screenshot:

- Terminal showing successful SSH into both EC2s.

```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

```
ubuntu@ip-10-0-2-34:~$
```

```
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

```
ubuntu@ip-10-0-1-197:~$ logout
Connection to 35.171.158.150 closed.
```

1.2 Azure – VNet, Subnet, NSG, VM

1.2.1 Create folder & files

```
cd project
mkdir -p azure
cd azure
```

variables.tf:

```
variable "azure_location" {
  description = "Azure location"
  type        = string
  default     = "eastus"
}

variable "project_name" {
  description = "Project name"
  type        = string
  default     = "multi-cloud-dr"
}

variable "admin_username" {
  description = "Admin username for VM"
```

```

type      = string
default   = "azureuser"
}

variable "public_key_path" {
  description = "Path to SSH public key"
  type        = string
}

main.tf:

terraform {
  required_version = ">= 1.3.0"

  required_providers {
    azurerm = {
      source  = "hashicorp/azurerm"
      version = "~> 4.0"
    }
  }
}

provider "azurerm" {
  features {}
}

resource "azurerm_resource_group" "rg" {
  name      = "${var.project_name}-rg"
  location  = var.azure_location
}

resource "azurerm_virtual_network" "vnet" {
  name          = "${var.project_name}-vnet"
  location      = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name
  address_space    = ["10.10.0.0/16"]
}

resource "azurerm_subnet" "subnet_app" {
  name          = "${var.project_name}-subnet-app"
  resource_group_name = azurerm_resource_group.rg.name
  virtual_network_name = azurerm_virtual_network.vnet.name
  address_prefixes  = ["10.10.1.0/24"]
}

resource "azurerm_network_security_group" "nsg" {
  name          = "${var.project_name}-nsg"
  location      = azurerm_resource_group.rg.location
  resource_group_name = azurerm_resource_group.rg.name

  security_rule {
    name      = "SSH"
    priority  = 100
    direction = "Inbound"
    access    = "Allow"
    protocol  = "Tcp"
  }
}

```

```

        source_port_range      = "*"
        destination_port_range = "22"
        source_address_prefix   = "*"
        destination_address_prefix = "*"
    }

    security_rule {
        name          = "HTTP"
        priority      = 110
        direction     = "Inbound"
        access         = "Allow"
        protocol       = "Tcp"
        source_port_range = "*"
        destination_port_range = "80"
        source_address_prefix   = "*"
        destination_address_prefix = "*"
    }
}

resource "azurerm_subnet_network_security_group_association"
"subnet_nsg_assoc" {
    subnet_id           = azurerm_subnet.subnet_app.id
    network_security_group_id = azurerm_network_security_group.nsg.id
}

resource "azurerm_public_ip" "app_pip" {
    name          = "${var.project_name}-app-pip"
    location      = azurerm_resource_group.rg.location
    resource_group_name = azurerm_resource_group.rg.name
    allocation_method = "Static"
    sku           = "Basic"
}

resource "azurerm_network_interface" "app_nic" {
    name          = "${var.project_name}-app-nic"
    location      = azurerm_resource_group.rg.location
    resource_group_name = azurerm_resource_group.rg.name

    ip_configuration {
        name          = "internal"
        subnet_id     = azurerm_subnet.subnet_app.id
        private_ip_address_allocation = "Dynamic"
        public_ip_address_id       = azurerm_public_ip.app_pip.id
    }
}

resource "azurerm_linux_virtual_machine" "app_vm" {
    name          = "${var.project_name}-app-vm"
    location      = azurerm_resource_group.rg.location
    resource_group_name = azurerm_resource_group.rg.name
    size          = "Standard_B1s"

    admin_username      = var.admin_username
    disable_password_authentication = true

    network_interface_ids = [azurerm_network_interface.app_nic.id]
}

```

```

admin_ssh_key {
  username    = var.admin_username
  public_key  = file(var.public_key_path)
}

os_disk {
  name          = "${var.project_name}-app-osdisk"
  caching       = "ReadWrite"
  storage_account_type = "Standard_LRS"
}

source_image_reference {
  publisher = "Canonical"
  offer     = "0001-com-ubuntu-server-jammy"
  sku       = "22_04-lts"
  version   = "latest"
}

computer_name = "appvm"

tags = {
  Role = "app"
}
}

```

outputs.tf:

```

output "app_vm_public_ip" {
  value      = azurerm_public_ip.app_pip.ip_address
  description = "Public IP of Azure App VM"
}

```

1.2.2 Run Terraform

az login # if not already logged in

```

Windows PowerShell
Directory: C:\Users\admin\project

Mode           LastWriteTime     Length Name
----           -----          ---- -
d----  11/22/2025  9:40 PM        0    azure

PS C:\Users\admin\project> cd azure
PS C:\Users\admin\project\azure> az login
Select the account you want to log in with. For more information on login with Azure CLI, see https://go.microsoft.com/fwlink/?linkid=2271136
Retrieving tenants and subscriptions for the selection...
[Tenant and subscription selection]
No   Subscription name      Subscription ID          Tenant
---  -----
[1] * npupgradl-1693990555121 ce01e918-0412-479f-9588-26fab9c46acb UpGrad

The default is marked with an *; the default tenant is 'UpGrad' and subscription is 'npupgradl-1693990555121' (ce01e918-0412-479f-9588-26fab9c46acb).

Select a subscription and tenant (Type a number or Enter for no changes):
Tenant: UpGrad
Subscription: npupgradl-1693990555121 (ce01e918-0412-479f-9588-26fab9c46acb)

[Announcements]
With the new Azure CLI login experience, you can select the subscription you want to use more easily. Learn more about it and its configuration at https://go.microsoft.com/fwlink/?linkid=2271236

If you encounter any problem, please open an issue at https://aka.ms/azclibug

[Warning] The login output has been updated. Please be aware that it no longer displays the full list of available subscriptions by default.

PS C:\Users\admin\project\azure>

```

```

cd project/azure
terraform init

PS C:\Users\admin\project\azure> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/azurerm versions matching "~> 4.0"...
- Installing hashicorp/azurerm v4.54.0...
- Installed hashicorp/azurerm v4.54.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\admin\project\azure>

```

```

terraform plan -var="public_key_path=$HOME/.ssh/mcdr_key.pub"
terraform apply -var="public_key_path=$HOME/.ssh/mcdr_key.pub"

```

Screenshots:

- Terminal with plan and apply (Azure).

```

+ address_prefixes          = [
+   "10.10.1.0/24",
]
+ default_outbound_access_enabled = true
+ id                         = (known after apply)
+ name                        = "multi-cloud-dr-subnet-app"
+ private_endpoint_network_policies = "Disabled"
+ private_link_service_network_policies_enabled = true
+ resource_group_name        = "multi-cloud-dr-rg"
+ virtual_network_name       = "multi-cloud-dr-vnet"
}

# azurerm_subnet_network_security_group_association.subnet_nsg_assoc will be created
+ resource "azurerm_subnet_network_security_group_association" "subnet_nsg_assoc" {
+   id           = (known after apply)
+   network_security_group_id = (known after apply)
+   subnet_id    = (known after apply)
}

# azurerm_virtual_network.vnet will be created
+ resource "azurerm_virtual_network" "vnet" {
+   address_space          = [
+     "10.10.0.0/16",
]
+   dns_servers            = (known after apply)
+   guid                   = (known after apply)
+   id                     = (known after apply)
+   location               = "eastus"
+   name                   = "multi-cloud-dr-vnet"
+   private_endpoint_vnet_policies = "Disabled"
+   resource_group_name    = "multi-cloud-dr-rg"
+   subnet                 = (known after apply)
}

Plan: 8 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ app_vm_public_ip = (known after apply)

```

```

azurerm_linux_virtual_machine.app_vm: Creating...
azurerm_linux_virtual_machine.app_vm: Still creating... [00m10s elapsed]
azurerm_linux_virtual_machine.app_vm: Still creating... [00m20s elapsed]
azurerm_linux_virtual_machine.app_vm: Still creating... [00m30s elapsed]
azurerm_linux_virtual_machine.app_vm: Still creating... [00m40s elapsed]
azurerm_linux_virtual_machine.app_vm: Still creating... [00m50s elapsed]
azurerm_linux_virtual_machine.app_vm: Creation complete after 54s [id=/subscriptions/ce01e918-0412-479f-9588-26fab9c46acb/resourceGroups/multi-cloud-dr-rg/providers/Microsoft.Compute/virtualMachines/multi-cloud-dr-app-vm]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:

app_vm_public_ip = "172.173.149.211"
PS C:\Users\admin\project\azure>

```

- Azure portal:
 - Resource group

Name	Subscription	Location
multi-cloud-dr-rg	npupgradl-1693990555121	East US
NetworkWatcherRG	npupgradl-1693990555121	East US

- VNet + subnet

Name	Resource Group	Location	Subscription
multi-cloud-dr-vnet	multi-cloud-dr-rg	East US	npupgradl-1693990555121

- o NSG inbound rules (22, 80)

The screenshot shows the Azure portal interface for managing Network Security Groups. The left sidebar navigation includes 'Virtual network', 'Network interfaces', and 'Network security groups'. The main content area is titled 'multi-cloud-dr-nsg' and shows the following details:

- Overview**: Shows the resource group (move) as 'multi-cloud-dr-rg', location as 'East US', and associated with 1 subnet, 0 network interfaces.
- Inbound Security Rules** table:

Priority	Name	Port	Protocol	Source
100	SSH	22	Tcp	Any
110	HTTP	80	Tcp	Any

- o VM overview (running, public IP).

The screenshot shows the Azure portal interface for managing Virtual Machines. The left sidebar navigation includes 'Virtual machines'. The main content area is titled 'multi-cloud-dr-app-vm' and shows the following details:

- Virtual machines**: Shows the status as 'Running'.
- Essentials** table:

Resource group (move)	: multi-cloud-dr-rg
Status	: Running
Location	: East US
Subscription (move)	: npupgradl-1693990555121
Subscription ID	: ce01e918-0412-479f-9588-26fab9c46acb
Operating system	: Linux (Ubuntu 22.04)
Size	: Standard B2ms (2 vcpus, 8 GiB memory)
Primary NIC public IP	: 172.173.149.211 1 associated public IPs
Virtual network/subnet	: multi-cloud-dr-vnet/multi-cloud-dr-sub...
DNS name	: Not configured
Health state	: -
Time created	: 22/11/2025, 16:55 UTC
Tags (edit)	: Role : app

1.2.3 SSH test to Azure VM

```
ssh -i ~/.ssh/mcdr_key azureuser@<AZURE_APP_PUBLIC_IP>
```

Screenshot:

Terminal showing successful SSH login.

```
app_vm_public_ip = "172.173.149.211"
PS C:\Users\admin\project\azure> ssh -i ~/.ssh/mcdr_key azureuser@172.173.149.211
The authenticity of host '172.173.149.211 (172.173.149.211)' can't be established.
ED25519 key fingerprint is SHA256:7bjazYgeOz7P9NEcwqh3Yq5hJJjDn/IY8SeqDTpYK330.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.173.149.211' (ED25519) to the list of known hosts.
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-1041-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

System information as of Sat Nov 22 17:04:57 UTC 2025

System load:  0.08          Processes:      122
Usage of /:   5.5% of 28.89GB  Users logged in:    0
Memory usage: 3%
Swap usage:   0%
IPv4 address for eth0: 10.10.1.4

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

Task 2 – Configuration Management (Ansible)

All Ansible work happens on the **AWS Tools Machine**.

2.1 Install Ansible on Tools Machine

- SSH to Tools Machine:

```
ssh -i ~/.ssh/mcdr_key ubuntu@<AWS_TOOLS_PUBLIC_IP>
```

- Install Ansible:

```
sudo apt update
sudo apt install -y ansible git
ansible --version
```

Screenshot:

`ansible --version` output.

```
ubuntu@ip-10-0-2-34: $ ansible --version
ansible [core 2.16.3]
  config file = None
  configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/ubuntu/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible
  python version = 3.12.3 (main, Aug 14 2025, 17:47:21) [GCC 13.3.0] (/usr/bin/python3)
  jinja version = 3.1.2
  libyaml = True
ubuntu@ip-10-0-2-34: $
```

2.2 Copy SSH key to Tools Machine

On **local machine**:

```
scp -i C:\Users\admin\.ssh\mcdr_key C:\Users\admin\.ssh\mcdr_key
ubuntu@<AWS_TOOLS_PUBLIC_IP>:/home/ubuntu/.ssh/
```

```
PS C:\Users\admin> scp -i C:\Users\admin\.ssh\mcdr_key C:\Users\admin\.ssh\mcdr_key ubuntu@54.144.200.231:/home/ubuntu/.ssh/
mcdr_key                                         100% 3381    10.8KB/s   00:00
PS C:\Users\admin>
PS C:\Users\admin>
```

On **Tools Machine**:

```
chmod 600 ~/.ssh/mcdr_key
```

2.3 Create Ansible inventory

On Tools Machine:

```

mkdir -p ~/multi-cloud-ansible
cd ~/multi-cloud-ansible

vim inventory.ini

inventory.ini:

[aws_app]
aws_app ansible_host=<AWS_APP_PUBLIC_IP> ansible_user=ubuntu
ansible_ssh_private_key_file=/home/ubuntu/.ssh/mcdr_key

[azure_app]
azure_app ansible_host=<AZURE_APP_PUBLIC_IP> ansible_user=azureuser
ansible_ssh_private_key_file=/home/ubuntu/.ssh/mcdr_key

[all_app_servers:children]
aws_app
azure_app

```

Test connectivity:

```
ansible -i inventory.ini all_app_servers -m ping
```

Screenshot:

Ping output showing pong from both hosts.

```

ubuntu@ip-10-0-2-34:~/multi-cloud-ansible$ ansible -i inventory.ini all_app_servers -m ping
[WARNING]: Found both group and host with same name: aws_app
[WARNING]: Found both group and host with same name: azure_app
aws_app | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
aws_app | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
ubuntu@ip-10-0-2-34:~/multi-cloud-ansible$
```

2.4 Ansible playbook to install & start Nginx

Create `nginx_setup.yml`:

```

---
- name: Install and configure Nginx on AWS and Azure app servers
  hosts: all_app_servers
  become: yes
  vars:
    nginx_pkg_name: nginx
```

```
tasks:
  - name: Update APT cache
    ansible.builtin.apt:
      update_cache: yes
      cache_valid_time: 3600

  - name: Install Nginx
    ansible.builtin.apt:
      name: "{{ nginx_pkg_name }}"
      state: present

  - name: Ensure Nginx is enabled and started
    ansible.builtin.service:
      name: nginx
      state: started
      enabled: yes

  - name: Check Nginx default page
    ansible.builtin.command: curl -s http://localhost
    register: nginx_page
    changed_when: false

  - name: Show first line of Nginx page
    ansible.builtin.debug:
      msg: "{{ nginx_page.stdout_lines | default([]) | first | default('no output') }}"
```

Run it:

```
ansible-playbook -i inventory.ini nginx_setup.yml
```

Screenshots:

- Playbook run showing success.

```
ubuntu@ip-10-0-2-34:/multi-cloud/ansible$ ansible-playbook -i inventory.ini nginx_setup.yml
[WARNING]: Found both group and host with same name: aws_app
[WARNING]: Found both group and host with same name: azure_app

PLAY [Install and configure Nginx on AWS and Azure app servers] ****
TASK [Gathering Facts] ****
ok: [aws_app]
ok: [azure_app]

TASK [Update APT cache] ****
changed: [aws_app]
changed: [azure_app]

TASK [Install Nginx] ****
changed: [aws_app]
changed: [azure_app]

TASK [Ensure Nginx is enabled and started] ****
ok: [azure_app]
ok: [aws_app]

TASK [Check Nginx default page] ****
ok: [azure_app]
ok: [aws_app]

TASK [Show first line of Nginx page] ****
ok: [azure_app] => {
  "msg": "<!DOCTYPE html>"
}
ok: [aws_app] => {
  "msg": "<!DOCTYPE html>"
}

PLAY RECAP ****
aws_app : ok=6    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
azure_app : ok=6    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ubuntu@ip-10-0-2-34:/multi-cloud/ansible$
```

On each app server:

```
# On AWS app
systemctl status nginx
ls -l /var/www/html/
```

```
# On Azure app
systemctl status nginx
ls -l /var/www/html/
```

Screenshots:

- `systemctl status nginx` on AWS & Azure = active.

```
Last login: Sat Nov 22 17:42:59 2025 from 54.144.200.231
azureuser@appvm: $ systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
  Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
  Active: active (running) since Sat 2025-11-22 17:42:37 UTC; 7min ago
    Docs: man:nginx(8)
 Process: 2834 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
 Process: 2835 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
 Main PID: 2930 (nginx)
   Tasks: 3 (limit: 9524)
  Memory: 5.5M
     CPU: 42ms
    CGroup: /system.slice/nginx.service
            ├─2930 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
            ├─2932 "nginx: worker process"
            ├─2933 "nginx: worker process"
Nov 22 17:42:37 appvm systemd[1]: Starting A high performance web server and a reverse proxy server...
Nov 22 17:42:37 appvm systemd[1]: Started A high performance web server and a reverse proxy server.

[1]+  Stopped                  systemctl status nginx
azureuser@appvm: $ ls -l /var/www/html/
total 4
-rw-r--r-- 1 root root 612 Nov 22 17:42 index.nginx-debian.html
azureuser@appvm: $
```

- `/var/www/html` listing.

```
Last login: Sat Nov 22 17:42:59 2025 from 54.144.200.231
ubuntu@ip-10-0-1-197: $ systemctl status nginx
● nginx.service - A high performance web server and a reverse proxy server
  Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
  Active: active (running) since Sat 2025-11-22 17:42:31 UTC; 8min ago
    Docs: man:nginx(8)
 Process: 2572 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
 Process: 2574 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
 Main PID: 2604 (nginx)
   Tasks: 3 (limit: 1008)
  Memory: 2.4M (peak: 5.3M)
     CPU: 26ms
    CGroup: /system.slice/nginx.service
            ├─2604 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
            ├─2606 "nginx: worker process"
            ├─2607 "nginx: worker process"

Nov 22 17:42:31 ip-10-0-1-197 systemd[1]: Starting nginx.service - A high performance web server and a reverse proxy se>
Nov 22 17:42:31 ip-10-0-1-197 systemd[1]: Started nginx.service - A high performance web server and a reverse proxy ser>

[1]+  Stopped                  systemctl status nginx
ubuntu@ip-10-0-1-197: $ ls -l /var/www/html/
total 4
-rw-r--r-- 1 root root 615 Nov 22 17:42 index.nginx-debian.html
ubuntu@ip-10-0-1-197: $
```

Task 3 – Application Deployment (Custom Nginx Pages)

3.1 Create custom html files on Tools Machine

From ~/multi-cloud-ansible:

```
cat > index-aws.html << 'EOF'  
<!doctype html>  
<html>  
<head><title>Welcome to AWS</title></head>  
<body>  
<h1>Welcome to AWS</h1>  
<p>This page is served from the AWS App Machine.</p>  
</body>  
</html>  
EOF  
  
cat > index-azure.html << 'EOF'  
<!doctype html>  
<html>  
<head><title>Welcome to Azure</title></head>  
<body>  
<h1>Welcome to Azure</h1>  
<p>This page is served from the Azure VM.</p>  
</body>  
</html>  
EOF
```

Screenshot:

ls showing `index-aws.html` and `index-azure.html`.

```
ubuntu@ip-10-0-2-34:~/multi-cloud-ansible$ cat index-azure.html  
<!doctype html>  
<html>  
<head><title>Welcome to Azure</title></head>  
<body>  
<h1>Welcome to Azure</h1>  
<p>This page is served from the Azure VM.</p>  
</body>  
</html>  
ubuntu@ip-10-0-2-34:~/multi-cloud-ansible$ cat index-aws.html  
<!doctype html>  
<html>  
<head><title>Welcome to AWS</title></head>  
<body>  
<h1>Welcome to AWS</h1>  
<p>This page is served from the AWS App Machine.</p>  
</body>  
</html>  
ubuntu@ip-10-0-2-34:~/multi-cloud-ansible$ ls  
index-aws.html  index-azure.html  inventory.ini  nginx_setup.yml  
ubuntu@ip-10-0-2-34:~/multi-cloud-ansible$
```

3.2 Add cloud flag to inventory

Edit `inventory.ini`:

```
[aws_app]
aws_app ansible_host=<AWS_APP_PUBLIC_IP> ansible_user=ubuntu
ansible_ssh_private_key_file=/home/ubuntu/.ssh/mcdr_key cloud=aws

[azure_app]
azure_app ansible_host=<AZURE_APP_PUBLIC_IP> ansible_user=azureuser
ansible_ssh_private_key_file=/home/ubuntu/.ssh/mcdr_key cloud=azure

[all_app_servers:children]
aws_app
azure_app
```

3.3 Deployment playbook

Create `nginx_deploy.yml`:

```
---
- name: Deploy custom index pages to AWS and Azure
  hosts: all_app_servers
  become: yes

  tasks:
    - name: Set index source based on cloud
      ansible.builtin.set_fact:
        custom_index_src: "index-{{ cloud }}.html"

    - name: Copy custom index.html to web root
      ansible.builtin.copy:
        src: "{{ custom_index_src }}"
        dest: /var/www/html/index.html
        owner: www-data
        group: www-data
        mode: '0644'
      notify: Restart nginx

    - name: Ensure Nginx default site uses index.html
      ansible.builtin.lineinfile:
        path: /etc/nginx/sites-available/default
        regexp: '^s*index '
        line: "\tindex index.html;"
        backup: yes
      notify: Restart nginx

    - name: Check Nginx page from localhost
      ansible.builtin.command: curl -s http://localhost
      register: nginx_page
      changed_when: false

    - name: Assert correct welcome text
      ansible.builtin.assert:
```

```

that:
  - >
    (cloud == 'aws' and 'Welcome to AWS' in nginx_page.stdout)
    or
    (cloud == 'azure' and 'Welcome to Azure' in nginx_page.stdout)
    fail_msg: "Custom welcome text not correct on {{ inventory_hostname }}"
}
success_msg: "Custom welcome text correct on {{ inventory_hostname }}"
}

handlers:
  - name: Restart nginx
    ansible.builtin.service:
      name: nginx
      state: restarted

```

Run:

```
ansible-playbook -i inventory.ini nginx_deploy.yml
```

Screenshots:

- Playbook run showing copy + assert tasks success.

```

ubuntu@ip-10-0-2-34:/multi-cloud-ansible$ ansible-playbook -i inventory.ini nginx_deploy.yml
[WARNING]: Found both group and host with same name: azure_app
[WARNING]: Found both group and host with same name: aws_app

PLAY [Deploy custom index pages to AWS and Azure] ****
TASK [Gathering Facts] **** ok: [aws_app]
ok: [azure_app]

TASK [Set index source based on cloud] **** ok: [aws_app]
ok: [azure_app]

TASK [Copy custom index.html to web root] **** changed: [aws_app]
changed: [azure_app]

TASK [Ensure Nginx default site uses index.html] **** changed: [aws_app]
changed: [azure_app]

TASK [Check Nginx page from localhost] **** ok: [aws_app]
ok: [aws_app]

TASK [Assert correct welcome text] **** ok: [aws_app] =>
{
  "changed": false,
  "msg": "Custom welcome text correct on aws_app"
}
ok: [aws_app] => {
  "changed": false,
  "msg": "Custom welcome text correct on azure_app"
}

RUNNING HANDLER [Restart nginx] **** changed: [aws_app]
changed: [aws_app]

PLAY RECAP **** aws_app : ok=7    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
azure_app   : ok=7    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

ubuntu@ip-10-0-2-34:/multi-cloud-ansible$ A
C

```

On AWS app:

```
cat /var/www/html/index.html
```

```
ubuntu@ip-10-0-1-197: $ cat /var/www/html/index.html
<!doctype html>
<html>
<head><title>Welcome to AWS</title></head>
<body>
<h1>Welcome to AWS</h1>
<p>This page is served from the AWS App Machine.</p>
</body>
</html>
ubuntu@ip-10-0-1-197: $
```

On Azure app: `cat /var/www/html/index.html`

```
azureuser@appvm:~$ cat /var/www/html/index.html
<!doctype html>
<html>
<head><title>Welcome to Azure</title></head>
<body>
<h1>Welcome to Azure</h1>
<p>This page is served from the Azure VM.</p>
</body>
</html>
azureuser@appvm:~$
```

Nginx config:

```
sudo cat /etc/nginx/sites-available/default | grep -i "index"
```

Screenshot:

Line showing `index index.html;`.

```
ubuntu@ip-10-0-1-197:~$ sudo cat /etc/nginx/sites-available/default | grep -i "index"
    # Add index.php to the list if you are using PHP
    index index.html;
#
#       index index.html;
ubuntu@ip-10-0-1-197:~$
```

```
azureuser@appvm:~$ sudo cat /etc/nginx/sites-available/default | grep -i "index"
    # Add index.php to the list if you are using PHP
    index index.html;
#
#       index index.html;
azureuser@appvm:~$
```

3.4 Browser verification

From your local machine:

- Open `http://<AWS_APP_PUBLIC_IP>` → **Welcome to AWS**



Welcome to AWS

This page is served from the AWS App Machine.

- Open http://<AZURE_APP_PUBLIC_IP> → **Welcome to Azure**



Welcome to Azure

This page is served from the Azure VM.

Screenshots: Browser for AWS page.



Welcome to AWS

This page is served from the AWS App Machine.

- Browser for Azure page.



Welcome to Azure

This page is served from the Azure VM.

Task 4 – Jenkins Setup for Continuous Deployment

4.1 Install Jenkins on Tools Machine

On Tools Machine:

```
sudo apt update
sudo apt install -y openjdk-17-jdk
java -version
```

Add Jenkins repo & install:

```
curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
/usr/share/keyrings/jenkins-keyring.asc > /dev/null

echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
/etc/apt/sources.list.d/jenkins.list > /dev/null

sudo apt update
sudo apt install -y jenkins
sudo systemctl enable jenkins
sudo systemctl start jenkins
sudo systemctl status jenkins
```

Screenshot:

systemctl status jenkins = active.

```
ubuntu@ip-10-0-2-34: $ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
  Loaded: loaded (/usr/lib/systemd/system/jenkins.service; enabled; preset: enabled)
  Active: active (running) since Sat 2025-11-22 18:21:07 UTC; 37s ago
    Main PID: 6994 (java)
       Tasks: 44 (limit: 1008)
      Memory: 341.0M (peak: 351.7M)
        CPU: 22.277s
       CGroup: /system.slice/jenkins.service
               └─6994 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Nov 22 18:21:02 ip-10-0-2-34 jenkins[6994]: [LF]> This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Nov 22 18:21:02 ip-10-0-2-34 jenkins[6994]: [LF]>
Nov 22 18:21:02 ip-10-0-2-34 jenkins[6994]: [LF]> ****
Nov 22 18:21:07 ip-10-0-2-34 jenkins[6994]: 2025-11-22 18:21:07.959+0000 [id=32]      INFO      jenkins.InitReactorRunner$1#onAttained: Complete
Nov 22 18:21:07 ip-10-0-2-34 jenkins[6994]: 2025-11-22 18:21:07.988+0000 [id=23]      INFO      hudson.lifecycle.Lifecycle#onReady: Jenkins is f
Nov 22 18:21:07 ip-10-0-2-34 systemd[1]: Started jenkins.service - Jenkins Continuous Integration Server.
Nov 22 18:21:08 ip-10-0-2-34 jenkins[6994]: 2025-11-22 18:21:08.294+0000 [id=49]      INFO      h.m.DownloadService$Downloadable#load: Obtained
Nov 22 18:21:08 ip-10-0-2-34 jenkins[6994]: 2025-11-22 18:21:08.295+0000 [id=49]      INFO      hudson.util.Retriger#start: Performed the action
lines 1-20/20 (END)
```

Update security group for Tools Machine to allow **TCP 8080** from your IP.

Open from browser:

http://<AWS_TOOLS_PUBLIC_IP>:8080

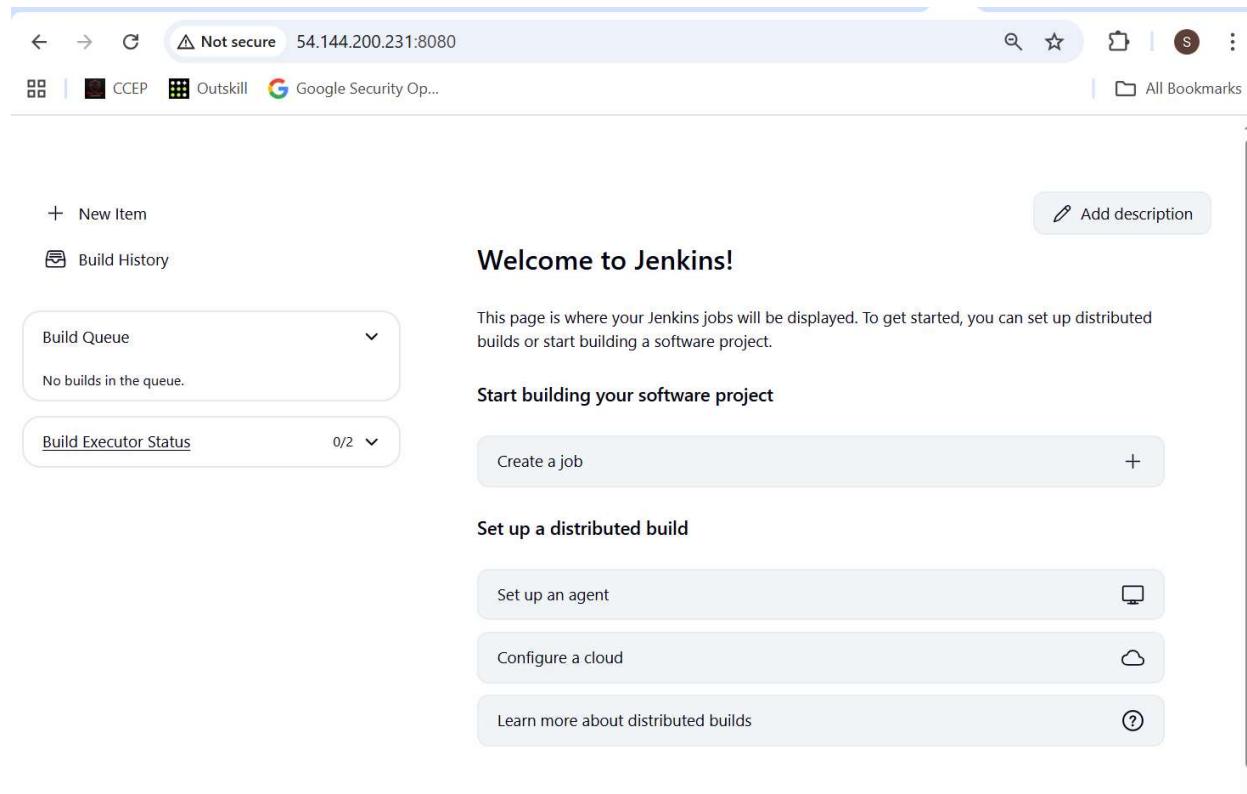
Get initial password:

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

Finish Jenkins setup and create admin user.

Screenshots:

- Jenkins first dashboard after setup.



4.2 Prepare GitHub repo

Create a GitHub repo, e.g. `multi-cloud-dr-cd` containing:

- `inventory.ini` (same content but key path `/var/lib/jenkins/.ssh/mcdr_key`)
- `nginx_deploy.yml`
- `index-aws.html`
- `index-azure.html`
- `Jenkinsfile`

On Tools Machine, copy key for Jenkins:

```
sudo mkdir -p ~jenkins/.ssh
sudo cp /home/ubuntu/.ssh/mcdr_key ~jenkins/.ssh/
```

```
sudo chown -R jenkins:jenkins ~jenkins/.ssh
sudo chmod 700 ~jenkins/.ssh
sudo chmod 600 ~jenkins/.ssh/mcdr_key
```

In repo's `inventory.ini`:

```
[aws_app]
aws_app ansible_host=<AWS_APP_PUBLIC_IP> ansible_user=ubuntu
ansible_ssh_private_key_file=/var/lib/jenkins/.ssh/mcdr_key cloud=aws

[azure_app]
azure_app ansible_host=<AZURE_APP_PUBLIC_IP> ansible_user=azureuser
ansible_ssh_private_key_file=/var/lib/jenkins/.ssh/mcdr_key cloud=azure

[all_app_servers:children]
aws_app
azure_app
```

4.3 Jenkinsfile

Create `Jenkinsfile` in repo:

```
pipeline {
    agent any

    environment {
        ANSIBLE_FORCE_COLOR = 'true'
    }

    stages {
        stage('Checkout') {
            steps {
                checkout scm
            }
        }

        stage('Deploy via Ansible') {
            steps {
                sh '''
                    ansible-playbook -i inventory.ini nginx_deploy.yml
                '''
            }
        }
    }

    post {
        success {
            echo 'Deployment successful.'
        }
        failure {
            echo 'Deployment failed. Check logs.'
        }
    }
}
```

Push everything to GitHub.

4.4 Configure Jenkins job

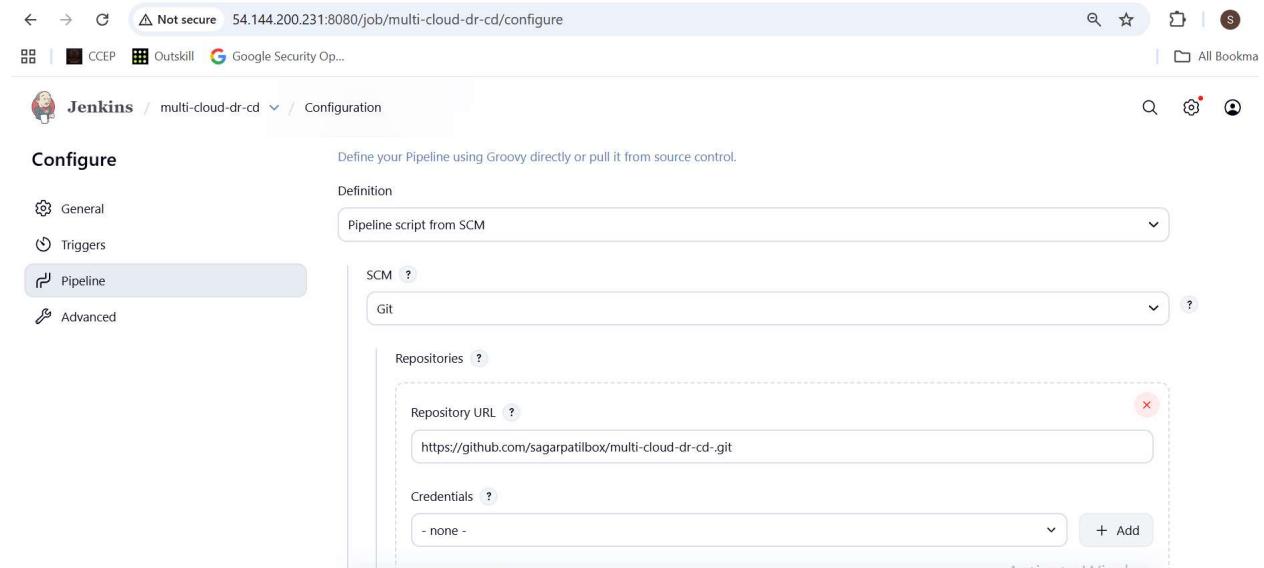
In Jenkins UI:

1. **New Item** → choose **Pipeline**.
2. Name: `multi-cloud-dr-cd`.
3. Under **Pipeline**:
 - o Definition: **Pipeline script from SCM**
 - o SCM: **Git**
 - o Repo URL: <https://github.com/sagarpatilbox/multi-cloud-dr-cd.git>
 - o Credentials: add if private
 - o Script Path: **Jenkinsfile**

Save.

Screenshot:

Jenkins job config page.



4.5 Run pipeline

Click **Build Now**.

Screenshots:

- Console output showing:



A screenshot of a web browser window showing Jenkins console output. The URL is 34.229.139.74:8080/job/multi-cloud-dr-cd/1/console. The page title is Jenkins / multi-cloud-dr-cd #1 / Console Output. The console output shows a Jenkins pipeline script with the following log entries:

```
[Pipeline]
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] echo
Deployment successful.
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Now edit HTML in GitHub (e.g. add v2) → run/build again → check pages updated.

Screenshots:

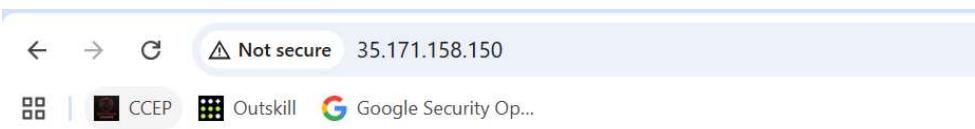
- Before/after browser pages.





Welcome to Azure Sagar

This page is served from the Azure VM.



Welcome to AWS

This page is served from the AWS App Machine.



Welcome to AWS Sagar

This page is served from the AWS App Machine.

Task 5 – Traffic Management Using AWS Route 53

I am Using My domain `app.sagardevsecops.online`

5.1 Create hosted zone

In AWS Console → Route 53 → Hosted zones → Create hosted zone:

- Domain name: `sagardevsecops.online`
- Type: Public hosted zone

Screenshot:

Hosted zone overview.

The screenshot shows the AWS Route 53 Hosted Zone Overview page. On the left, there's a navigation sidebar with options like Dashboard, Hosted zones (which is selected), Health checks, Profiles, IP-based routing, Traffic flow, Domains, Resolver, VPCs, Inbound endpoints, Outbound endpoints, and Rules. The main content area has a breadcrumb path: Route 53 > Hosted zones > sagardevsecops.online. A prominent blue success message box at the top says "Record for sagardevsecops.online was successfully created. Route 53 propagates your changes to all of the Route 53 authoritative DNS servers within 60 seconds. Use "View status" button to check propagation status." Below this, there are tabs for Records (6), DNSSEC signing, and Hosted zone tags (0). The Records tab is active, showing a table with columns: Record name, Type, Routing policy, Differentiator, and Alias. The records listed are:

Record name	Type	Routing policy	Differentiator	Alias
sagardevsecops.online	NS	Simple	-	No
sagardevsecops.online	SOA	Simple	-	No
app.sagardevsecops.online	A	Failover	Primary	No
app.sagardevsecops.online	A	Failover	Secondary	No

5.2 Create health check for AWS app

Route 53 → Health checks → Create health check:

- Name: `aws-app-health`
- Endpoint → IP address = `<AWS_APP_PUBLIC_IP>`
- Protocol: `HTTP`
- Port: `80`
- Path: `/`

Screenshot:

Health check details (status OK when app running).

The screenshot shows the AWS Route 53 Health checks interface. On the left, there's a navigation sidebar with options like Dashboard, Hosted zones, Health checks (selected), Profiles, IP-based routing, Traffic flow, Domains, Resolver, and VPCs. The main content area is titled "Health checks (1) Info" and contains a table with one row. The table columns are ID, Name, Details, Status in last 24 hours, Current s..., Alarm, and Actions. The single entry is "Ofbfdcf3-1438-42cf..." with the name "aws-app-health" and the URL "http://35.171.158....". The status is "Healthy" with a green icon. Under "Actions", there's a link "None, Create alarm". At the bottom of the main area, it says "Select a health check". A watermark at the bottom right says "Activate Windows Go to Settings to activate Windows."

5.3 Create failover records

In hosted zone → **Create record:**

Primary record (AWS)

- Record name: `app.sagardevsecops.online`
- Type: A
- Routing policy: **Failover**
- Failover record type: **Primary**
- Value: `<AWS_APP_PUBLIC_IP>`
- Health check: `aws-app-health`
- Record ID: `aws-primary`

Secondary record (Azure)

- Record name: `app.sagardevsecops.online`
- Type: A
- Routing policy: **Failover**
- Failover record type: **Secondary**
- Value: `<AZURE_APP_PUBLIC_IP>`
- No health check
- Record ID: `azure-secondary`

Screenshots:

app.sagardevsecops.online



Welcome to AWS

This page is served from the AWS App Machine.

5.4 Test failover

1. With AWS **app running**:
2. curl <http://app.sagardevsecops.online>

Screenshot: Browser output.



Welcome to AWS

This page is served from the AWS App Machine.

3. Stop AWS App instance in EC2 console.
 - o Wait for health check to show **Unhealthy**.

Now you should see **Welcome to Azure**.

Screenshots:

- o EC2 instance state = stopped.

The screenshot shows the AWS EC2 Instances page. The left sidebar navigation includes EC2, Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager, Images (AMIs, AMI Catalog), and Elastic Block Store. The main content area displays a table titled 'Instances (2) Info' with columns: Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, and Public IPv4 DNS. The first instance, 'multi-cloud-dr-tools', is listed as 'Running' with 't3.medium' type, '3/3 checks passed' status, 'View alarms' button, 'us-east-1b' zone, and 'ec2-34-229-139-74.co...' DNS. The second instance, 'multi-cloud-dr-app', is listed as 'Stopped' with 't3.micro' type, no status check information, 'View alarms' button, 'us-east-1a' zone, and no DNS listed. A search bar at the top is set to 'Find Instance by attribute or tag (case-sensitive)'.

- o Health check status = Unhealthy.

The screenshot shows the AWS Route 53 Health checks page. The left sidebar navigation includes Route 53 (selected), Dashboard, Hosted zones, Health checks (selected), Profiles (New), IP-based routing (CIDR collections), Traffic flow (Traffic policies, Policy records), Domains (Registered domains, Requests), and Resolver (VPCs, Inbound endpoints, Outbound endpoints). The main content area displays a table titled 'Health checks (1) Info' with columns: ID, Name, Details, Status in last 24 hours, Current s..., Alarm, and Actions. One health check is listed: 'aws-primary-health' with ID 'a33f7cd6-5a6d-40...', endpoint 'http://ec2-44-202...', status 'Unhealthy', and an 'None, Create alarm' action. A 'Create health check' button is located at the top right of the table area. A search bar at the top is set to 'Find health check'.

- Browser showing Azure page via `app.sagardevsecops.online`.



Welcome to Azure

This page is served from the Azure VM.

4. Start AWS instance again, wait until health check is healthy, and re-test.

The screenshot shows the AWS Route 53 Health Checks console. The left sidebar lists navigation options: Dashboard, Hosted zones, Health checks (selected), Profiles (New), IP-based routing, Traffic flow, Domains, and Resolver. The main content area is titled "Health checks (1) Info" and displays a table of health checks. One entry is shown: "aws-primary-health" with ID "a53f7cd6-5a6d-40...". The status is "Healthy" and the alarm status is "None, Create alarm".

ID	Name	Details	Status in last 24 hours	Current s...	Alarm	Actions
a53f7cd6-5a6d-40...	aws-primary-health	http://ec2-13-221...	Healthy		None, Create alarm	⋮

- Browser showing AWS page again.



Welcome to AWS

This page is served from the AWS App Machine.

★ Bonus Task – Kubernetes (EKS + AKS + Nginx + Failover)

This is optional but nice.

B.1 EKS via Terraform (high-level)

In project/k8s/aws-eks/main.tf you can use:

```
terraform {
  required_version = ">= 1.3.0"

  required_providers {
    aws = {
      source  = "hashicorp/aws"
      version = "~> 5.0"
    }
  }
}

provider "aws" {
  region = "us-east-1"
}

# 1) VPC module
module "vpc" {
  source  = "terraform-aws-modules/vpc/aws"
  version = "~> 5.0"

  name = "eks-dr-vpc"
  cidr = "10.20.0.0/16"

  azs          = ["us-east-1a", "us-east-1b"]
  private_subnets = ["10.20.1.0/24", "10.20.2.0/24"]
  public_subnets = ["10.20.11.0/24", "10.20.12.0/24"]

  enable_nat_gateway = true
}

# 2) EKS module
module "eks" {
  source  = "terraform-aws-modules/eks/aws"
  version = "~> 20.0"

  cluster_name     = "eks-dr-cluster"
  cluster_version = "1.30"

  vpc_id       = module.vpc.vpc_id
  subnet_ids = module.vpc.private_subnets

  # EKS API endpoint - public everywhere (lab only!)
  cluster_endpoint_private_access      = true
  cluster_endpoint_public_access       = true
```

```

cluster_endpoint_public_access_cidrs = ["0.0.0.0/0"]

# Let the Terraform caller (your admin user) be cluster admin
authentication_mode = "API_AND_CONFIG_MAP"
enable_cluster_creator_admin_permissions = true

eks_managed_node_groups = {
  default = {
    min_size      = 1
    max_size      = 2
    desired_size  = 1
    instance_types = ["t3.small"]
  }
}
}

```

Run:

```

terraform init
terraform apply
aws eks update-kubeconfig --name eks-dr-cluster --region us-east-1
kubectl get nodes

```

Screenshots:

kubectl get nodes , EKS console.

```

PS C:\Users\admin\project\k8s\aws-eks> kubectl get pods
No resources found in default namespace.
PS C:\Users\admin\project\k8s\aws-eks>

```

The screenshot shows the AWS EKS Cluster Management interface. On the left, there's a sidebar with navigation links: Dashboard, Clusters (selected), Settings, Amazon EKS Anywhere, and Related services (Amazon ECR, AWS Batch). The main content area is titled 'Clusters (1) Info'. It displays a table with one row for the cluster 'eks-dr-cluster'. The columns in the table are: Cluster name (eks-dr-cluster), Status (Active), Kubernetes version (1.30), Support period (Extended support until July 23, 2026), Upgrade policy (Extended support), Created (17 minutes ago), and Provider (EKS). There are buttons for 'Delete' and 'Create cluster' at the top right of the table. The browser address bar shows 'us-east-1.console.aws.amazon.com/eks/clusters?region=us-east-1'.

Cluster name	Status	Kubernetes version	Support period	Upgrade policy	Created	Provider
eks-dr-cluster	Active	1.30	Extended support until July 23, 2026	Extended support	17 minutes ago	EKS

```

}

# module.eks.module.eks_managed_node_group["default"].module.user_data.null_resource.validate_cluster_service_cidr will be created
+ resource "null_resource" "validate_cluster_service_cidr" {
  + id = (known after apply)
}

Plan: 56 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

module.eks.module.eks_managed_node_group["default"].aws_iam_role.this[0]: Creating

[redacted]

module.eks.aws_eks_cluster.this[0]: Still creating... [09m10s elapsed]
module.eks.aws_eks_cluster.this[0]: Still creating... [09m20s elapsed]
module.eks.aws_eks_cluster.this[0]: Still creating... [09m30s elapsed]
module.eks.aws_eks_cluster.this[0]: Creation complete after 9m32s [id=eks-dr-cluster]
module.eks.data.tls_certificate.this[0]: Reading...
module.eks.time_sleep.this[0]: Creating...
module.eks.data.tls_certificate.this[0]: Read complete after 2s [id=0c37bb608751c2a14704338d4b14c14e6dbe88ac]
module.eks.aws_iam_openid_connect_provider.oidc_provider[0]: Creating...
module.eks.aws_iam_openid_connect_provider.oidc_provider[0]: Creation complete after 2s [id=arn:aws:iam::980104576357:oidc-provider/oidc.eks.us-east-1.amazonaws.com/id/417D59DA8BC2B0E6656A8348C81517B2]
module.eks.time_sleep.this[0]: Still creating... [00m10s elapsed]
module.eks.time_sleep.this[0]: Still creating... [00m20s elapsed]
module.eks.time_sleep.this[0]: Still creating... [00m30s elapsed]
module.eks.time_sleep.this[0]: Creation complete after 30s [id=2025-11-23T13:10:45Z]
module.eks.module.eks_managed_node_group["default"].module.user_data.null_resource.validate_cluster_service_cidr: Creating...
module.eks.module.eks_managed_node_group["default"].module.user_data.null_resource.validate_cluster_service_cidr: Creation complete after 0s [id=5786262614895386475]
module.eks.module.eks_managed_node_group["default"].aws_launch_template.this[0]: Creating...
module.eks.module.eks_managed_node_group["default"].aws_launch_template.this[0]: Creation complete after 9s [id=lt-0e21f1df7dd79b976]
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Creating...
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Still creating... [00m10s elapsed]
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Still creating... [00m20s elapsed]
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Still creating... [00m30s elapsed]
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Still creating... [00m40s elapsed]
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Still creating... [00m50s elapsed]
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Still creating... [01m00s elapsed]
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Still creating... [01m10s elapsed]
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Still creating... [01m20s elapsed]
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Still creating... [01m30s elapsed]
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Still creating... [01m40s elapsed]
module.eks.module.eks_managed_node_group["default"].aws_eks_node_group.this[0]: Creation complete after 1m42s [id=eks-dr-cluster:default-20251123131054085200000015]

Apply complete! Resources: 56 added, 0 changed, 0 destroyed.
PS C:\Users\admin\project\k8s\aws-eks> ■

```

B.2 AKS via Terraform

project/k8s/azure-aks/main.tf similar to earlier AKS config:

```

provider "azurerm" {
  features {}
  subscription_id = "4c419c18-22ee-4dfa-8014-71f05d545136"
}

resource "azurerm_resource_group" "rg" {
  name      = "aks-dr-rg"
  location  = "West US3"
}

```

```

resource "azurerm_kubernetes_cluster" "aks" {
  name                  = "aks-dr-cluster"
  location              = azurerm_resource_group.rg.location
  resource_group_name   = azurerm_resource_group.rg.name
  dns_prefix            = "aksdr"

  default_node_pool {
    name      = "default"
    node_count = 1
    vm_size    = "Standard_B2s"
  }

  identity {
    type = "SystemAssigned"
  }
}

```

Then: `az login` # for login

```

terraform init
terraform apply
terraform output -raw kube_config > aks.kubeconfig
KUBECONFIG=aks.kubeconfig kubectl get nodes

```

Screenshots:

AKS cluster, `kubectl get nodes`.

```

PS C:\Users\admin\project\k8s\azure-aks> kubectl get nodes
NAME           STATUS   ROLES   AGE   VERSION
aks-default-24710189-vmss000000   Ready    <none>  15m   v1.32.9
PS C:\Users\admin\project\k8s\azure-aks>

```

The screenshot shows the Microsoft Azure portal interface. On the left, the sidebar navigation includes 'Resource Manager' under 'Default Directory'. The main content area displays the 'aks-dr-rg' resource group. The 'Overview' tab is selected, showing the 'Essentials' section with a table of resources. One resource is listed: 'aks-dr-cluster' (Type: Kubernetes service, Location: West US 3). The URL in the browser bar is `portal.azure.com/?quickstart=True#@sagardpatil55outlook.onmicrosoft.com/resource/subscriptions/4c419c18-22ee-4dfa-8014-71f05...`.

```

+ default_node_pool {
    + kubelet_disk_type      = (known after apply)
    + max_pods                = (known after apply)
    + name                     = "default"
    + node_count               = 1
    + node_labels              = (known after apply)
    + orchestrator_version     = (known after apply)
    + os_disk_size_gb          = (known after apply)
    + os_disk_type              = "Managed"
    + os_sku                    = (known after apply)
    + scale_down_mode           = "Delete"
    + type                      = "VirtualMachineScaleSets"
    + ultra_ssd_enabled         = false
    + vm_size                   = "Standard_B2s"
    + workload_runtime           = (known after apply)
}

+ identity {
    + principal_id = (known after apply)
    + tenant_id    = (known after apply)
    + type          = "SystemAssigned"
}

+ kubelet_identity (known after apply)

+ network_profile (known after apply)

+ windows_profile (known after apply)
}

# azurerm_resource_group.rg will be created
+ resource "azurerm_resource_group" "rg" {
    + id      = (known after apply)
    + location = "eastus"
    + name    = "rg-aks-dr"
}

```

Plan: 2 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?

Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value:

B.3 Nginx manifests (same for both clusters)

`configmap-aws.yaml` (apply to EKS):

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: nginx-index
data:
  index.html: |
    <html><body><h1>Welcome to AWS - Kubernetes</h1></body></html>
```

`configmap-azure.yaml` (apply to AKS):

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: nginx-index
data:
  index.html: |
    <html><body><h1>Welcome to Azure - Kubernetes</h1></body></html>
```

`nginx-deployment.yaml`:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-k8s
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx-k8s
  template:
    metadata:
      labels:
        app: nginx-k8s
    spec:
      containers:
        - name: nginx
          image: nginx:stable
          ports:
            - containerPort: 80
      volumeMounts:
        - name: index-html
          mountPath: /usr/share/nginx/html/index.html
          subPath: index.html
  volumes:
    - name: index-html
      configMap:
        name: nginx-index
        items:
          - key: index.html
            path: index.html
```

```
nginx-service.yaml:

apiVersion: v1
kind: Service
metadata:
  name: nginx-lb
spec:
  type: LoadBalancer
  selector:
    app: nginx-k8s
  ports:
    - port: 80
      targetPort: 80
```

Apply to **EKS**:

```
kubectl apply -f configmap-aws.yaml
kubectl apply -f nginx-deployment.yaml
kubectl apply -f nginx-service.yaml
kubectl get svc nginx-lb
```

Apply to **AKS**:

For powershell :

```
$Env:KUBECONFIG = "aks.kubeconfig"

az aks get-credentials `

--resource-group aks-dr-rg `

--name aks-dr-cluster

KUBECONFIG=aks.kubeconfig kubectl apply -f configmap-azure.yaml
KUBECONFIG=aks.kubeconfig kubectl apply -f nginx-deployment.yaml
KUBECONFIG=aks.kubeconfig kubectl apply -f nginx-service.yaml
KUBECONFIG=aks.kubeconfig kubectl get svc nginx-lb
```

Use the **EXTERNAL-IP** values as endpoints.

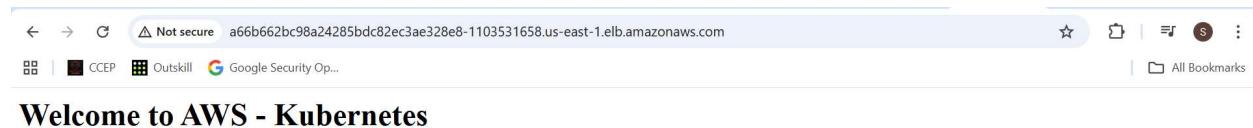
Screenshots:

- `kubectl get svc nginx-lb` for both clusters.

```
PS C:\Users\admin\project\k8s\aws-eks> kubectl get pods
No resources found in default namespace.
PS C:\Users\admin\project\k8s\aws-eks> kubectl apply -f configmap-aws.yaml
configmap/nginx-index created
PS C:\Users\admin\project\k8s\aws-eks> kubectl apply -f nginx-deployment.yaml
deployment.apps/nginx-k8s created
PS C:\Users\admin\project\k8s\aws-eks> kubectl apply -f nginx-service.yaml
service/nginx-lb created
PS C:\Users\admin\project\k8s\aws-eks> kubectl get svc nginx-lb
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
nginx-lb   LoadBalancer   172.20.161.185   a66b662bc98a24285bcd82ec3ae328e8-1103531658.us-east-1.elb.amazonaws.com   80:32056/TCP   28s
PS C:\Users\admin\project\k8s\aws-eks>
```

```
PS C:\Users\admin\project\k8s\azure-aks> kubectl apply -f configmap-azure.yaml
configmap/nginx-index unchanged
PS C:\Users\admin\project\k8s\azure-aks> kubectl apply -f nginx-deployment.yaml
deployment.apps/nginx-k8s created
PS C:\Users\admin\project\k8s\azure-aks> kubectl apply -f nginx-service.yaml
service/nginx-lb created
PS C:\Users\admin\project\k8s\azure-aks> kubectl get svc nginx-lb
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
nginx-lb   LoadBalancer   10.0.176.11   <pending>      80:32395/TCP   13s
PS C:\Users\admin\project\k8s\azure-aks> kubectl get svc nginx-lb
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
nginx-lb   LoadBalancer   10.0.176.11   20.171.172.54   80:32395/TCP   63s
PS C:\Users\admin\project\k8s\azure-aks>
```

- Browser showing AWS K8s page and Azure K8s page.





Welcome to Azure - Kubernetes



Welcome to AWS - Kubernetes

B.4 Route 53 failover with LB IPs

Update `k8s.sagardevsecops.online` failover records:

- Primary A → EKS LB IP
- Secondary A → AKS LB IP

(**Health check on EKS LB.**)

Screenshots:

- Route 53 records with LB IPs.

The screenshot shows the AWS Route 53 console interface. A success message at the top states: "Record for sagardevsecops.online was successfully created. Route 53 propagates your changes to all of the Route 53 authoritative DNS servers within 60 seconds. Use "View status" button to check propagation status." Below this, a table lists four DNS records:

Record name	Type	Routing	Difference	Alias	Value/Route traffic to	TTL (s...)
sagardevsec... (checkbox)	NS	Simple	-	No	ns-1171.awsdns-18.org. ns-163.awsdns-20.com. ns-645.awsdns-16.net. ns-1934.awsdns-49.co.uk.	172800
sagardevsec... (checkbox)	SOA	Simple	-	No	ns-1171.awsdns-18.org. aw... (checkbox)	900
k8s.sagardev... (checkbox)	A	Failover	Primary	No	184.73.206.21	300
k8s.sagardev... (checkbox)	A	Failover	Secondary	No	20.171.172.54	300

Below the table, a message says "0 records selected" and "Select a record to see its details".

At the bottom of the screenshot, there is another browser window showing a "Welcome to AWS - Kubernetes" page.

B.5 Simulate EKS failure

On EKS:

```
kubectl scale deploy nginx-k8s --replicas=0  
kubectl get pods
```

When health check fails, `k8s.sagardevsecops.online` should serve Azure K8s page.

Screenshots:

- `kubectl get pods` showing 0 pods.

```
PS C:\Users\admin\project\k8s\aws-eks> kubectl get pods
No resources found in default namespace.
PS C:\Users\admin\project\k8s\aws-eks>
```

```
PS C:\Users\admin\project\k8s\azure-aks> kubectl get pods
NAME                   READY   STATUS    RESTARTS   AGE
nginx-k8s-5688d7b88c-ccsk4  1/1     Running   0          37m
nginx-k8s-5688d7b88c-h5dx7  1/1     Running   0          37m
PS C:\Users\admin\project\k8s\azure-aks> kubectl scale deploy nginx-k8s --replicas=0
deployment.apps/nginx-k8s scaled
PS C:\Users\admin\project\k8s\azure-aks> kubectl get pods
No resources found in default namespace.
PS C:\Users\admin\project\k8s\azure-aks>
```

- Health check unhealthy.

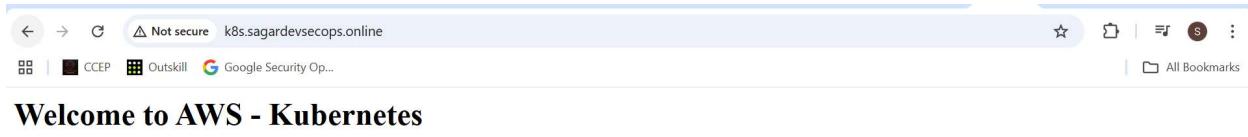
The screenshot shows the AWS Route 53 Health Checks interface. On the left, there's a navigation sidebar with options like Dashboard, Hosted zones, Health checks (which is selected), Profiles, IP-based routing, Traffic flow, and Domains. The main content area is titled "Health checks (1)" and contains a table with one row. The table columns are ID, Name, Details, Status in last 24 hours, Current s..., Alarm, and Actions. The single entry is "a31b6929-20d5-4a...", "aws-primary-health...", "http://184.73.206...", "Unhealthy", "None, Create alarm", and a three-dot menu icon. A search bar at the top says "Find health check".

- Browser showing Azure K8s page via app.sagardevsecops.online.

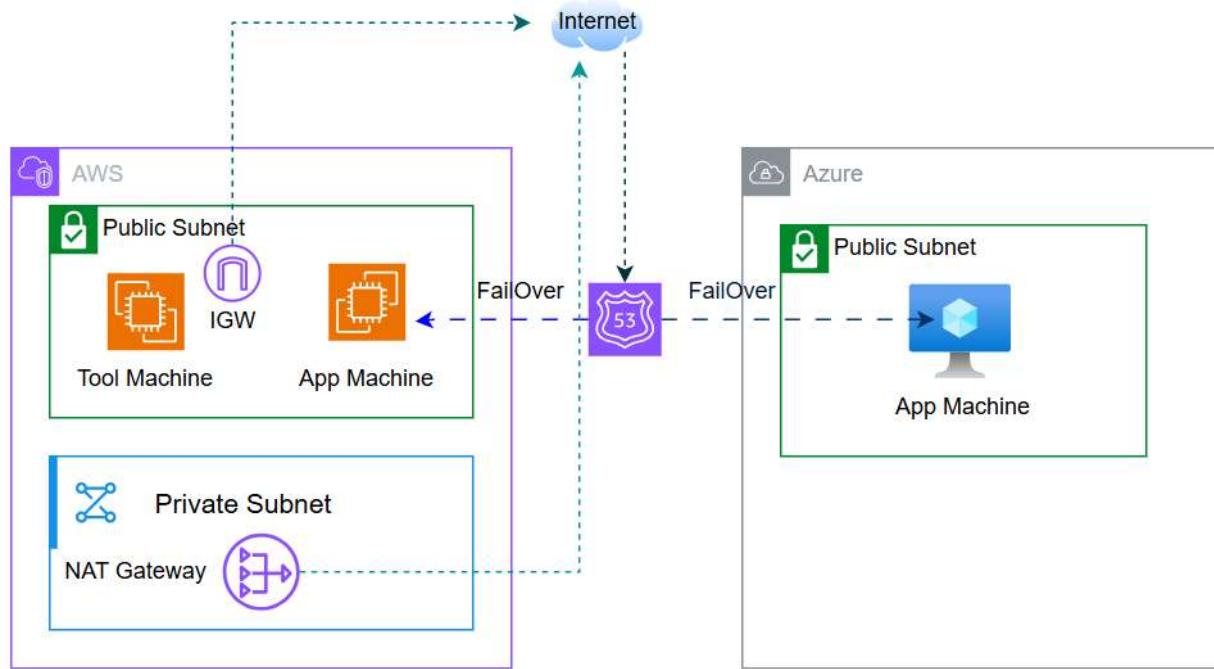
The screenshot shows a browser window with the URL "k8s.sagardevsecops.online". The page content is "Welcome to Azure - Kubernetes". The browser's address bar shows "Not secure" and the URL "k8s.sagardevsecops.online". The status bar at the bottom includes "CCEP", "Outskill", "Google Security Op...", and "All Bookmarks".

Scale back:

```
kubectl scale deploy nginx-k8s --replicas=2
```



Multi-Cloud DR Project Architecture Diagram :



❖ Multi-Cloud DR Project – Total Cost to Company (TCO) Report :

Cloud Provider	Resource	Qty	Estimate (US\$/mo)	Estimate (INR/mo)
AWS	App EC2 t3.micro	1	8.5	680
AWS	Tools EC2 t3.micro	1	8.5	680
AWS	NAT Gateway	1	32	2,560
AWS	Route 53 hosted zone + health check	1	1	80
AWS Total			≈ 50	≈ 3,998
Azure	VM Standard_B2ms	1	70	5,600
Azure	Public IP (Standard SKU)	1	3	240
Azure	OS disk + storage	1	5	400
Azure Total			≈ 78	≈ 6,240
Grand Total	Combined AWS + Azure		≈ US\$ 128	≈ ₹10,250