Ansible_node.sh: -

```
#!/bin/bash


#Ansible_node.sh

# This Script can use at boot time while launching the ec2 instance

#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

# Created Bootstrap script for user creation and install ansible and add inot adm group.

# Here i had used the Ubuntu 22 for Ansible Node as well as Ansible Server.

# Taken two EC2 instances one for Ansible-Server and one for Ansible-Node

# Created two user data scripts for Ansible-Server and one for Ansible-Node

# First execute the Ansible-Node.sh script to make the connection of remote server using Password-Based Authentication AWS by default disabled Password-Based Authentication

# Script is Prepared by Mr. Siva Kumar

# LinkedIn URL: "https://www.linkedin.com/in/sivakumar120406"

# Github repo url:"https://github.com/sivakumar1204/User-Data-Scripts-for-Ansible-Setup.git"


# Specify your variables

USERNAME="ansadmin"

DEFAULT_USERNAME="ubuntu"

DEFAULT_PASSWORD="test123"


# 1. Delete the current password for the $DEFAULT_USERNAME (optional)

echo -e "$DEFAULT_PASSWORD\n$DEFAULT_PASSWORD" | sudo passwd -d $DEFAULT_USERNAME


# 2. Set the new password for the $DEFAULT_USERNAME

echo -e "$DEFAULT_PASSWORD\n$DEFAULT_PASSWORD" | sudo passwd $DEFAULT_USERNAME


echo "User's password has been updated"


# 3. Add $DEFAULT_USERNAME to the adm group

sudo usermod -aG adm $DEFAULT_USERNAME
```

```
# 4. Enable Password-Based Authentication to Yes and Restart the ssh service to effect changes
sudo sed -i 's/PasswordAuthentication no/PasswordAuthentication yes/' /etc/ssh/sshd_config
echo "Password-based authentication has been enabled"

# 5. Restart SSHD service to apply the changes
sudo systemctl restart sshd
echo "sshd service has been restarted successfully"

# 6. Creation of user with default options
sudo adduser $USERNAME <<EOF
test123
test123
<Full Name>
<Room Number>
<Work Phone>
<Home Phone>
<Other>
y
EOF

# 7. Add the user $USERNAME into sudoers group
echo "$USERNAME ALL=(ALL) NOPASSWD:ALL" | sudo EDITOR='tee -a' visudo
echo "$DEFAULT_USERNAME ALL=(ALL) NOPASSWD:ALL" | sudo EDITOR='tee -a' visudo

# 8. Add user $USERNAME to the adm group
sudo usermod -aG adm $USERNAME
sudo usermod -aG sudo $USERNAME
```

Ansible-Server.sh:-

```
#!/bin/bash


#Ansible_server.sh

# This Script can use at boot time while launching the ec2 instance

#++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

# Created Bootstrap script is prepared to creation and install ansible and test the connection using ansible ping module in Ansible-Server.sh

# Here i had used the Ubuntu 22 for Ansible Node as well as Ansible Server.

# Taken two EC2 instances one for Ansible-Server and one for Ansible-Node

# Created two user data scripts for Ansible-Server and one for Ansible-Node

# First execute the Ansible-Node.sh script to make the connection of remote server using Password-Based Authentication AWS by default disabled Password-Based Authentication

# Script is Prepared by Mr. Siva Kumar

# LinkedIn URL: "https://www.linkedin.com/in/sivakumar120406"

# Github repo url:"https://github.com/sivakumar1204/User-Data-Scripts-for-Ansible-Setup.git"


# List the Environment variables


export DEFAULT_USERNAME="ubuntu"

export DEFAULT_PASSWORD="test123"

export USERNAME="ansadmin"

export PASSWORD="test123"

export REMOTE_USERNAME="ansadmin"

export REMOTE_SERVER_IP_ADDRESS="10.0.1.10"

export REMOTE_PORT="22"  # Typically 22


# Define your public key file

export PUBLIC_KEY_FILE="/home/$USERNAME/.ssh/id_rsa.pub"


# 1. Export the REMOTE_SERVER_IP_ADDRESS as an environment variable
# export REMOTE_SERVER_IP_ADDRESS

# 1. Delete the current password for the $DEFAULT_USERNAME (optional)

echo -e "$DEFAULT_PASSWORD\n$DEFAULT_PASSWORD" | sudo passwd -d $DEFAULT_USERNAME

# 2. Set the new password for the $DEFAULT_USERNAME

echo -e "$DEFAULT_PASSWORD\n$DEFAULT_PASSWORD" | sudo passwd $DEFAULT_USERNAME

echo "User's password has been updated"

# 3. Add $DEFAULT_USERNAME to the adm group

#sudo usermod -aG adm $DEFAULT_USERNAME

# 4. Enable Password-Based Authentication to Yes and Restart the ssh services to effect changes

sudo sed -i 's/PasswordAuthentication no/PasswordAuthentication yes/' /etc/ssh/sshd_config

echo "Password-based authentication has been enabled"

# 5. Restart SSHD service to apply the changes

sudo systemctl restart sshd

echo "sshd service has been restarted successfully"

# 6. Creation of user with default options

sudo adduser $USERNAME <<EOF
```

```
test123

test123

Full Name

Room Number

Work Phone

Home Phone

Other

y

EOF


# 7. Add the user $USERNAME to sudoers group


echo "$USERNAME ALL=(ALL) NOPASSWD:ALL" | sudo EDITOR='tee -a' visudo

echo "$DEFAULT_USERNAME ALL=(ALL) NOPASSWD:ALL" | sudo EDITOR='tee -a' visudo


# 8. Add user $USERNAME to adm group


sudo usermod -aG adm $USERNAME

sudo usermod -aG sudo $USERNAME


# 9. Install Ansible


sudo apt-get update -y


echo "System update is completed successfully"


sudo apt install ansible -y


echo "Ansible installation completed successfully"


# 10. Check the version of Ansible
```

```
ansible --version


# 11. Switch to the user terminal


#sudo -u $USERNAME bash <<EOF


# 12. Create SSH keys for the user $USERNAME


sudo -u $USERNAME ssh-keygen -t rsa -b 2048 -f /home/$USERNAME/.ssh/id_rsa -N ""

sudo -u $USERNAME chmod 700 /home/$USERNAME/.ssh

sudo -u $USERNAME chmod 600 /home/$USERNAME/.ssh/id_rsa

sudo -u $USERNAME chmod 644 /home/$USERNAME/.ssh/id_rsa.pub

sudo -u $USERNAME chown $USERNAME:$USERNAME /home/$USERNAME/*

#sudo -u $USERNAME ssh-copy-id $REMOTE_USERNAME@$REMOTE_SERVER_IP_ADDRESS


# 13. Change to the home directory of '$USERNAME'

#cd ~


# 14. Create an Ansible inventory file

sudo -u $USERNAME echo "[web-server]" | tee -a /home/$USERNAME/hosts

sudo -u $USERNAME echo "$REMOTE_SERVER_IP_ADDRESS" | tee -a /home/$USERNAME/hosts


# 15. Copy the public key onto the remote server


# Copy the public key to the remote server using SSH

sudo -u $USERNAME ssh -p "$REMOTE_PORT"
"$REMOTE_USERNAME@$REMOTE_SERVER_IP_ADDRESS" "mkdir -p ~/.ssh && echo '$(cat
$PUBLIC_KEY_FILE)' >> ~/.ssh/authorized_keys"

#sudo -u $USERNAME ssh-copy-id $REMOTE_USERNAME@$REMOTE_SERVER_IP_ADDRESS


#echo "Copied public key to $REMOTE_SERVER_IP_ADDRESS successfully"
```

```
# 16. Test the connectivity using the Ansible ping module


sudo -u $USERNAME ansible all -i /home/$USERNAME/hosts -u $REMOTE_USERNAME -m ping
#ansible all -i ~/hosts -u $REMOTE_USERNAME -m ping


# Exit the '$USERNAME' user shell
```

VPC.tf

```
/*
provider "aws" {
  region = "ca-central-1"
}
*/
terraform {
 backend "s3" {
  bucket = "siva-project"
  key    = "DevOps-Terrraform/terraform.tfstate"
  region = "ca-central-1"
 }
}
# Create a VPC
resource "aws_vpc" "My-Project-VPC" {
 cidr_block = "10.0.0.0/16"
 enable_dns_support = true
 enable_dns_hostnames = true
 tags = {
  Name = "My-Project-VPC"
  Owner = "Siva"
  Dept = "DevOps"
}
}
# Create two subnets and associate public IPs

resource "aws_subnet" "My-Project-Public-Subnet-1A" {
 vpc_id                  = aws_vpc.My-Project-VPC.id
 cidr_block              = "10.0.1.0/24"
 availability_zone       = "ca-central-1a"
 map_public_ip_on_launch = true
```

```
  tags = {
    Name = "My-Project-Public-Subnet-1A"
    Owner = "Siva"
    Dept = "DevOps"
    }
}
# Create two subnets and associate public IPs
resource "aws_subnet" "My-Project-Public-Subnet-1B" {
  vpc_id                  = aws_vpc.My-Project-VPC.id
  cidr_block              = "10.0.2.0/24"
  availability_zone       = "ca-central-1b"
  map_public_ip_on_launch = true
  tags = {
    Name = "My-Project-Public-Subnet-1B"
    Owner = "Siva"
    Dept = "DevOps"
    }
}
# Create an internet gateway and attach it to the VPC
resource "aws_internet_gateway" "My-Project-IGW" {
  vpc_id = aws_vpc.My-Project-VPC.id
  tags = {
    Name = "My-Project-IGW"
    Owner = "Siva"
    Dept = "DevOps"
    }
}

# Create a route table and associate it with the VPC
resource "aws_route_table" "My-Project-RT" {
  vpc_id = aws_vpc.My-Project-VPC.id
  tags = {
    Name = "My-Project-RT"
    Owner = "Siva"
    Dept = "DevOps"
    }
}

# Create a route for the internet gateway
resource "aws_route" "My-Project-Route_To_My-Project-IGW" {
  route_table_id         = aws_route_table.My-Project-RT.id
  destination_cidr_block = "0.0.0.0/0"
  gateway_id             = aws_internet_gateway.My-Project-IGW.id

}

# Attach the subnets to the route table
resource "aws_route_table_association" "My-Project-Public-Subnet-1A_Association" {
  subnet_id      = aws_subnet.My-Project-Public-Subnet-1A.id
  route_table_id = aws_route_table.My-Project-RT.id
}

resource "aws_route_table_association" "My-Project-Public-Subnet-1B_Association" {
  subnet_id      = aws_subnet.My-Project-Public-Subnet-1B.id
  route_table_id = aws_route_table.My-Project-RT.id
}
```

Variables.tf

```
locals {
  Owner="Siva"
  Dept = "DevOps"
  Env ="Sandbox"
}

variable "region" {
  default = "ca-central-1"
}

variable "vpc-id" {
    default = "vpc-087cbdfe0debcf675"

}

variable "subnet-1A-id" {
    default = "subnet-0afbb341f551e0e5b"

}

variable "subnet-1B-id" {
    default = "subnet-0a26fb8165aa35599"

}

variable "vpc_security_group_id" {
    default = "sg-085b5bb1f7225ee35"

}
```

```
variable "ansible-node-private-ip" {
  default = "10.0.1.10"


}
variable "ansible-server-private-ip" {
  default = "10.0.2.15"
}
variable "key-name" {
  default = "My-Project"


}
variable "ubuntu-ami-id" {
  default = "ami-0ea18256de20ecdfc" # Ubuntu Server 22.04
}

variable "instance-type" {
  default = "t2.micro"
}
```

```
# Create a security group allowing SSH and all traffic from anywhere
resource "aws_security_group" "My-Project-SG" {
 name        = "My-Project-SG"
 description = "My-Project-SG"
 vpc_id      = aws_vpc.My-Project-VPC.id
 tags = {
  Name = "My-Project-SG"
  Owner = "Siva"
  Dept = "DevOps"
  }
 ingress {
  from_port   = 22
  to_port     = 22
  protocol    = "tcp"
  cidr_blocks = ["0.0.0.0/0"]
 }

 ingress {
  from_port   = 0
  to_port     = 65535
  protocol    = "tcp"
  cidr_blocks = ["0.0.0.0/0"]
 }
  # Outbound rule to allow all traffic
 egress {
  from_port   = 0
  to_port     = 65535
```

```
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}
```

Provider.tf

```
/*
provider "aws" {
    region = "ca-central-1"
}
*/
```

Ansible-Node.tf

+++++++++++

```
/*

data "aws_subnet" "My-Project-Public-Subnet-1A" {

  id = aws_subnet.My-Project-Public-Subnet-1A.id

}

*/


resource "aws_instance" "Ubuntu-Ansible-Node" {

  ami            = var.ubuntu-ami-id          # Replace with the desired Ubuntu AMI ID

  instance_type   = var.instance-type           # Change to your preferred instance type

  #key_name       = aws_key_pair.My-Project-key.key_name       # Replace with your created pem key
file

  key_name       = var.key-name

  subnet_id      = var.subnet-1A-id

  #subnet_id      = data.aws_subnet.My-Project-Public-Subnet-1A.id # Refer from the data block

  #security_group_id = [aws_security_group.My-Project-SG.id]  # Replace with your Security Group id

  vpc_security_group_ids = [var.vpc_security_group_id]

  #vpc_security_group_ids = [aws_security_group.My-Project-SG.id] # Replace with your Security
Group id

  associate_public_ip_address = true

  private_ip = var.ansible-node-private-ip

  #public_ip = "10.0.1.10"

  user_data = file("Ansible-Node.sh")

  /*

  #user_data = <<-EOF

            #!/bin/bash

            export CURRENT_USER="ubuntu"

            export NEW_USER="ansadmin"

            export NEW_PASSWORD="test123"

            useradd -m -s /bin/bash $NEW_USER

            echo "$NEW_USER:$NEW_PASSWORD" | chpasswd
            usermod -aG adm $CURRENT_USER

            usermod -aG adm $NEW_USER

            echo "$CURRENT_USER ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers

            echo "$NEW_USER ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers

            sed -i 's/PasswordAuthentication no/PasswordAuthentication yes/' /etc/ssh/sshd_config

            service ssh restart


            EOF

  */

  tags = {

    Name = "Ubuntu-Ansible-Node"

    Owner = local.Owner

    Dept = local.Dept

    Env = local.Env

    }

    #Store the public ip and instance ip into  the file respective files

    provisioner "local-exec" {

    command = "echo ${aws_instance.Ubuntu-Ansible-Node.public_ip} > Ubuntu-Ansible-Node-
Public-IP.txt"

    }

    provisioner "local-exec" {

    command = "echo ${aws_instance.Ubuntu-Ansible-Node.id} > Ubuntu-Ansible-Node-
Instance.id.txt"

    }



    /*

    provisioner "remote-exec" {

      inline = [

        "sudo apt-get update -y",

        "sudo apt install ansible -y"

       ]

      connection {
        type        = "ssh"

        user        = "ubuntu"

        private_key = file("My-Project.pem")  //Copied pem key file  manually and stored in the terrsfom
working directory

        #private_key = file("Provide the full path where is your pem file")

        host    = self.public_ip

        timeout = "30m"

      }

    }

    */

}


#Display the Public IP

output "Ubuntu-Ansible-Node-Public-IP" {

  value = aws_instance.Ubuntu-Ansible-Node.public_ip


}


#Display the INstance ID

output "Ubuntu-Ansible-Node-id" {

  value = aws_instance.Ubuntu-Ansible-Node.id


}


/*

# If you want to create cusotm key and add the key into your ec2 instance then use the below
resource section do the necessary changes

resource "aws_key_pair" "My-Project-key" {

  key_name   = "My-Project-key"

  # teh best approch is to provide the path where the public key is stored in your local system

  public_key = file("C:/Users/sivar/Desktop/My-Project/My-Project/Terraform/id_rsa.pub")
//providing the pem file path
```

```
  #if you havve pem file with you tehn place that pem key file in to the current terraform working
directory

  #public_key = file("id_rsa.pub")

  # public_key = file("id_rsa.pub") //placing the public file in terraform lab directory

  # the other way is to provide the public key copy the public key and paste direct in the public key
attribute same as below

  #public_key = "ssh-rsa
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxeJZ7Isv87uAMoxTUBK91MY2J
x4o55Ejqan0i7gn0ijJ8apgA0uawZT6hA8QIC2zmpaqwNRdibgdh8N5yyS79o7gpxbzTtwLwhjtVwVGS9R
3SYuK/3R0RAeeG+JKmHdjPPGqGZHQl59+QBxFFVeVHurOqIgnfVA1KIDC0lolLfyKSKq5yvYIAV6t0/TYI7V
LblJL+OgU8oiREPqHrUJ3iI0TNctsO/bhn7zKFkrRyB8EyZU2MYdOKhtWEw6V8UOs0FNLXqCe1mY/4z0US
ju0AWO+q+1i6w1wV8= XXXX@LAPTOP-XXXXXX""

  tags = {

    Name = "My-Project-key"

    Owner = "Siva"

    Dept = "DevOps"

    }

}


*/
```

Ansible-Server.tf

+++++++++++++

```
/*
data "aws_subnet" "My-Project-Public-Subnet-1B" {
  id = aws_subnet.My-Project-Public-Subnet-1B.id
}
*/

resource "aws_instance" "Ubuntu-Ansible-Server" {
  ami           = var.ubuntu-ami-id         # Replace with the desired Ubuntu AMI ID
  instance_type   = var.instance-type          # Change to your preferred instance type
  #key_name       = aws_key_pair.My-Project-key.key_name       # Replace with your pem key file
  key_name       = var.key-name
  subnet_id      = var.subnet-1B-id
  #subnet_id      = data.aws_subnet.My-Project-Public-Subnet-1B.id # Replace with your subnet id
  #security_group_id = [aws_security_group.My-Project-SG.id]  # Replace with your Security Group id
  vpc_security_group_ids = [var.vpc_security_group_id]
  #vpc_security_group_ids = [aws_security_group.My-Project-SG.id] # Replace with your Security
Group id
  associate_public_ip_address = true
  private_ip = var.ansible-server-private-ip
  #public_ip = "10.0.2.15"
  user_data = file("Ansible-Server.sh")
  tags = {
    Name = "Ubuntu-Ansible-Server"
    Owner = local.Owner
    Dept = local.Dept
    Env = local.Env
    }
  provisioner "local-exec" {
    command = "echo ${aws_instance.Ubuntu-Ansible-Server.public_ip} > Ubuntu-Ansible-Server-
Public-IP.txt"
```

```
   }
  provisioner "local-exec" {
    command = "echo ${aws_instance.Ubuntu-Ansible-Server.id} > Ubuntu-Ansible-Server-
Instance.id.txt"
   }
  /*
  provisioner "remote-exec" {
     inline = [
     "#!/bin/bash",
     "export USERNAME=\"ansadmin\"",
     "export PASSWORD=\"test123\"",
     "export REMOTE_USERNAME=\"ansadmin\"",
     "export REMOTE_SERVER_IP_ADDRESS=\"10.0.1.10\"",
     "export REMOTE_PORT=\"22\"",
     "export PUBLIC_KEY_FILE=\"/home/$USERNAME/.ssh/id_rsa.pub\"",
     "sudo -u $USERNAME ssh -p \"$REMOTE_PORT\"
\"$REMOTE_USERNAME@$REMOTE_SERVER_IP_ADDRESS\" \"mkdir -p ~/.ssh && echo '$(cat
$PUBLIC_KEY_FILE)' >> ~/.ssh/authorized_keys\"",
     "sudo -u $USERNAME ansible all -i /home/$USERNAME/hosts -u $REMOTE_USERNAME -m ping",
   ]


     connection {
      type       = "ssh"
      port       = "22"
      user       = "ubuntu"
      private_key = file("My-Project.pem")  //created this pem key file  manually and stored in the
terrsfom working directory
      host    = self.public_ip
      timeout = "30m"
     }
   }
  */
```

```
}


#Display the Public IP

output "Ubuntu-Ansible-Server-Public-IP" {

  value = aws_instance.Ubuntu-Ansible-Server.public_ip


}


#Display the INstance ID

output "Ubuntu-Ansible-Server-id" {

  value = aws_instance.Ubuntu-Ansible-Server.id


}


/*

# Copy the public key on to the remote server using below command


sudo -u ansadmin ssh-copy-id ansadmin@10.0.1.10


# Check teh connectivity


sudo -u ansadmin ansible all -i /home/ansadmin/hosts -u ansadmin -m ping


/*

# If you want to create cusotm key and add the key into your ec2 instance then use the below
resource section do the necessary changes

resource "aws_key_pair" "My-Project-key" {

  key_name   = "My-Project-key"

  # teh best approch is to provide the path where the public key is stored in your local system

  public_key = file("C:/Users/sivar/Desktop/My-Project/My-Project/Terraform/id_rsa.pub")
//providing the pem file path
```

```
  #if you havve pem file with you tehn place that pem key file in to the current terraform working
directory
  #public_key = file("id_rsa.pub")
  # public_key = file("id_rsa.pub") //placing the public file in terraform lab directory
  # the other way is to provide the public key copy the public key and paste direct in the public key
attribute same as below
  #public_key = "ssh-rsa
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxeJZ7Isv87uAMoxTUBK91MY2J
x4o55Ejqan0i7gn0ijJ8apgA0uawZT6hA8QIC2zmpaqwNRdibgdh8N5yyS79o7gpxbzTtwLwhjtVwVGS9R
3SYuK/3R0RAeeG+JKmHdjPPGqGZHQl59+QBxFFVeVHurOqIgnfVA1KIDC0lolLfyKSKq5yvYIAV6t0/TYI7V
LblJL+OgU8oiREPqHrUJ3iI0TNctsO/bhn7zKFkrRyB8EyZU2MYdOKhtWEw6V8UOs0FNLXqCe1mY/4z0US
ju0AWO+q+1i6w1wV8= XXXX@LAPTOP-XXXXXX""
  tags = {
    Name = "My-Project-key"
    Owner = "Siva"
    Dept = "DevOps"
    }
}

*/
```