Gmail **Mohit Rathore <mohit.rathore4@gmail.com>**

regarding terraform script..

IntellipaatSupportTeam <support@intellipaat.com> Fri, Apr 1, 2022 at 3:12 PM
Reply-To: support@intellipaat.com
To: Mohit Rathore <mohit.rathore4@gmail.com>
Cc: support1@intellipaat.com, aman dubey <amandubey@intellipaat.com>, sayed shoaib <shoaib@intellipaat.com>, anand padhy <anand.padhy@intellipaat.com>, ayush alawe <ayush.alawe@intellipaat.com>

Hi mohit.rathore4@gmail.com mohit.rathore4@gmail.com,

Thank you for raising your concern with Intellipaat!

Kindly find the script:

Terraform

Install terraform:

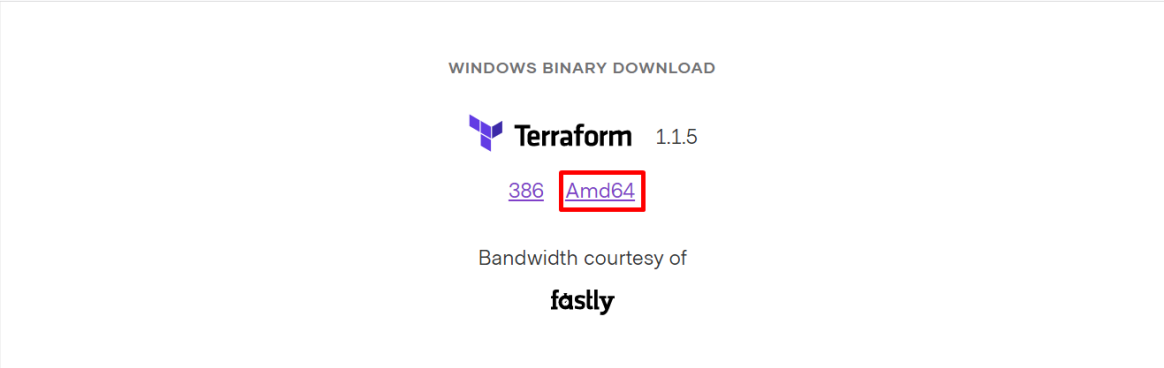
Goto https://www.terraform.io/downloads.html
Linux Installation commands :

curl -O https://releases.hashicorp.com/terraform/1.0.11/terraform_1.0.11_linux_amd64.zip

sudo apt install unzip -y
unzip terraform_1.0.11_linux_amd64.zip
sudo mv terraform /usr/local/bin

For windows Installation:

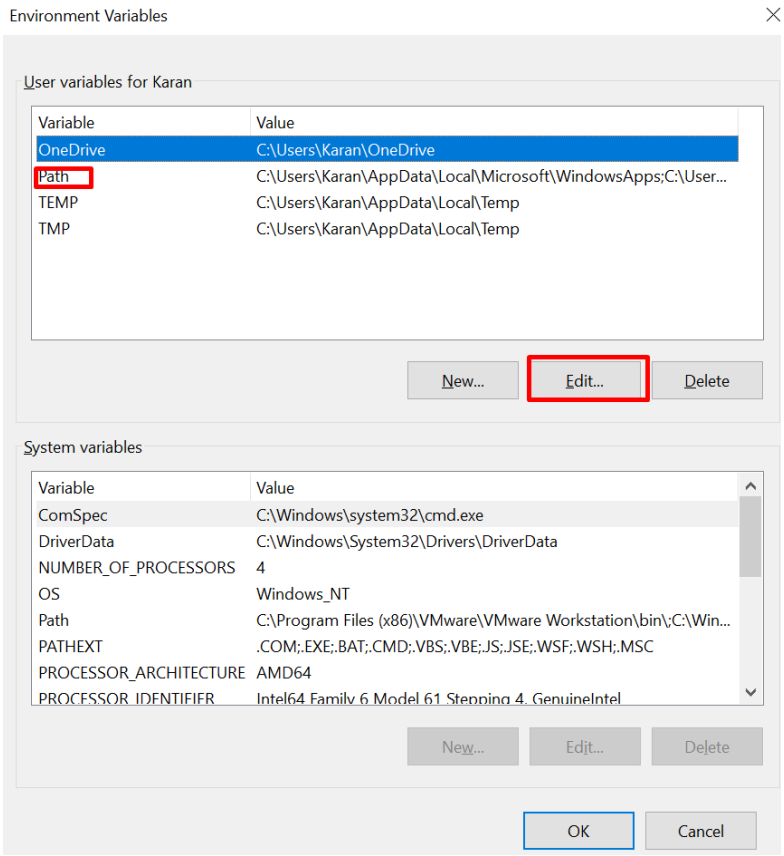
https://www.terraform.io/downloads



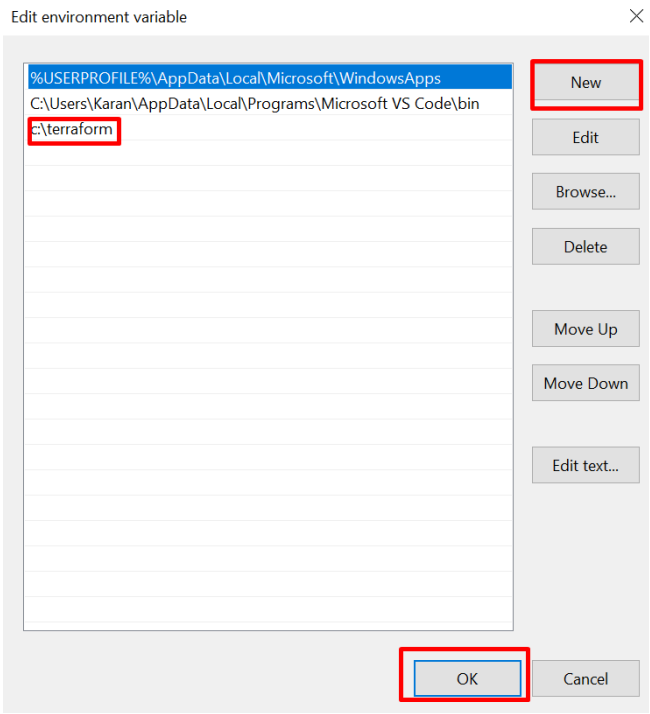
Extract to c:\terraform

Note: terraform is a directory that we have created.

Configure the environment variable —->



Add the following path.



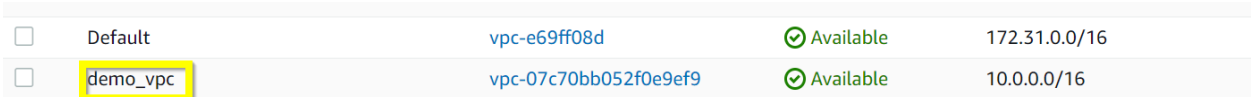
Common terraform commands:

terraform init
terraform plan
terraform apply
terraform validate
terraform destroy

CREATE VPC

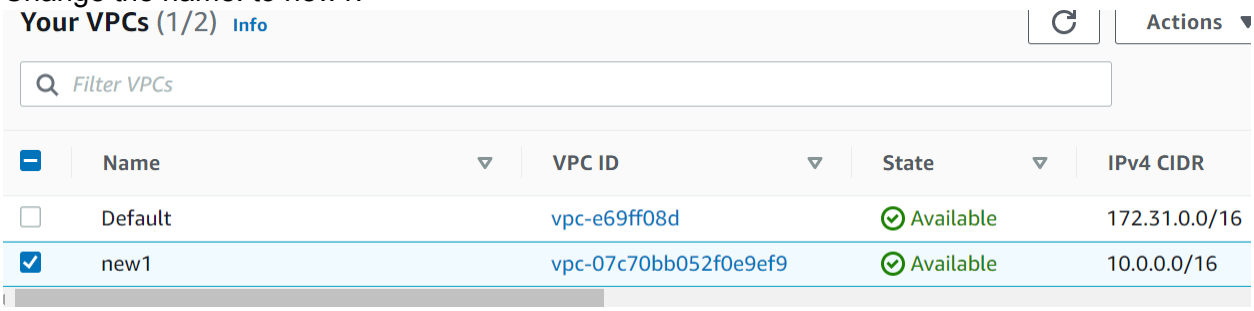
```
provider "aws" {
 region = "us-east-2"
 access_key = ""
 secret_key = ""
}
resource "aws_vpc" "main" {
  cidr_block = "10.0.0.0/16"
  tags={
    Name = "demo_vpc"
}
}
```

Note: terraform can handle configuration drift, eg: if I change the name of the vpc that I have created from the console then check from terraform plan in terminal it would show something like this:



☐	Default	vpc-e69ff08d	Available	172.31.0.0/16
☐	demo_vpc	vpc-07c70bb052f0e9ef9	Available	10.0.0.0/16

Change the name: to new1.



Your VPCs (1/2) Info — Actions

Filter VPCs

	Name	VPC ID	State	IPv4 CIDR
☐	Default	vpc-e69ff08d	Available	172.31.0.0/16
☑	new1	vpc-07c70bb052f0e9ef9	Available	10.0.0.0/16

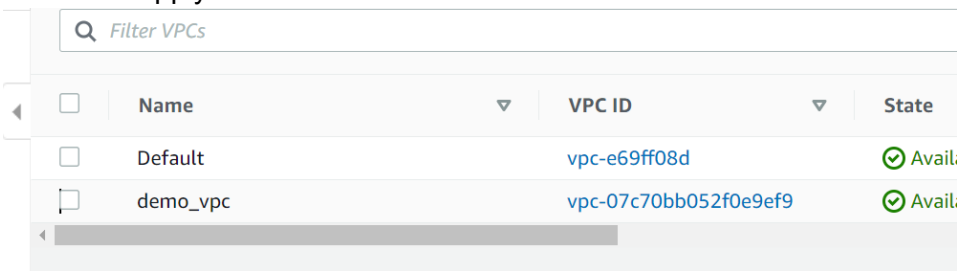
terraform plan

output:

```
 # aws_vpc.main will be updated in-place
  ~ resource "aws_vpc" "main" {
      id                               = "vpc-07c70bb052f0e9ef9"
    ~ tags                             = {
        ~ "Name" = "new1" -> "demo_vpc"
      }
    ~ tags_all                         = {
```

```
      ~ "Name" = "new1" -> "demo_vpc"
    }
    # (12 unchanged attributes hidden)
  }

Plan: 0 to add, 1 to change, 0 to destroy.
```

Now
terraform apply



CREATE SUBNET

```
resource "aws_subnet" "main" {
  vpc_id     = aws_vpc.main.id
  cidr_block = "10.0.1.0/24"

  tags = {
    Name = "Main"
  }
}
```

Note: terraform.tfstate is a file that contains the configuration of the desired state and actual state.
terraform show

The command terraform show will show the configuration of the resource.
Output:

```
ubuntu@ip-172-31-42-192:~/terraform$ terraform show
# aws_subnet.main:
resource "aws_subnet" "main" {
    arn                             = "arn:aws:ec2:us-east-2:124058707612:subnet/subnet-003a60619581612c1"
    assign_ipv6_address_on_creation = false
    availability_zone               = "us-east-2c"
    availability_zone_id            = "use2-az3"
    cidr_block                      = "10.0.1.0/24"
    id                              = "subnet-003a60619581612c1"
    map_customer_owned_ip_on_launch = false
    map_public_ip_on_launch         = false
```

```
    owner_id                      = "124058707612"
    tags                          = {
        "Name" = "Main"
    }
    tags_all                      = {
        "Name" = "Main"
    }
    vpc_id                        = "vpc-0efb36eb7c0d41ee7"
}

# aws_vpc.main:
resource "aws_vpc" "main" {
    arn                           = "arn:aws:ec2:us-east-2:124058707612:vpc/vpc-0efb36eb7c0d41ee7"
    assign_generated_ipv6_cidr_block = false
    cidr_block                    = "10.0.0.0/16"
    default_network_acl_id        = "acl-0b40e944ba0f609a8"
    default_route_table_id        = "rtb-06d86a7cc9b4352a7"
    default_security_group_id     = "sg-07ed0474deaed6518"
    dhcp_options_id               = "dopt-f030709b"
    enable_dns_hostnames          = false
    enable_dns_support            = true
    id                            = "vpc-0efb36eb7c0d41ee7"
    instance_tenancy              = "default"
    main_route_table_id           = "rtb-06d86a7cc9b4352a7"
    owner_id                      = "124058707612"
    tags                          = {
        "Name" = "demo_vpc"
    }
    tags_all                      = {
        "Name" = "demo_vpc"
    }
}
```

Creating 2 different instances in 2 different regions:

```
provider "aws" {
    access_key = ""
    secret_key = ""
    region     = "us-east-1"
    alias=  "virginia"
}
provider "aws" {
    access_key = ""
    secret_key = ""
```

```
    alias  = "ohio"
    region     = "us-east-2"
}

resource "aws_instance" "virginia" {
   provider = aws.virginia
   ami= "ami-0ed9277fb7eb570c9"
   instance_type = "t2.micro"
   key_name=”demo1009_nv”
  tags = {
    Name = "virginiaserver1"
  }
}
resource "aws_instance" "ohio" {
   provider = aws.ohio
   ami= "ami-0fb653ca2d3203ac1"
   instance_type = "t2.micro"
   key_name= “demo1009”
  tags = {
    Name = "ohio_instrance"
  }
}
```

Attaching EIP to Ohio instance.

```
provider "aws" {
    access_key = "AKIARZYTZU2ON6L2LDHQ"
    secret_key = "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"
    region     = "us-east-1"
    alias=  "virginia"
}
provider "aws" {
    access_key = "AKIARZYTZU2ON6L2LDHQ"
    secret_key = "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"
    alias  = "ohio"
    region     = "us-east-2"
}

resource "aws_instance" "virginia_instance" {
   provider = aws.virginia
   ami= "ami-0ed9277fb7eb570c9"
   instance_type = "t2.micro"
  tags = {
    Name = "virginiaserver1"
  }
```

```
}
resource "aws_instance" "ohio_instance" {
   provider = aws.ohio
   ami= "ami-0fb653ca2d3203ac1"
   instance_type = "t2.micro"
  tags = {
    Name = "ohio_instance1"
  }
}
resource "aws_eip" "lb" {
  provider = aws.ohio
  instance = aws_instance.ohio_instance.id
  vpc      = true
}
```

complete AWS architecture

```
ubuntu@ip-172-31-44-162:~/terrafrom_directory$ cat main.tf
provider "aws" {
 region = "us-east-2"
 access_key = ""
 secret_key = ""
}
resource "aws_vpc" "main" {
  cidr_block = "10.0.0.0/16"
  tags={
    Name = "demo_vpc_terraform"
  }
}
resource "aws_subnet" "public_subnet" {
  vpc_id     = aws_vpc.main.id
  cidr_block = "10.0.1.0/24"
  map_public_ip_on_launch = "1"
  tags = {
    Name = "public_tf_subnet"
  }
}

resource "aws_security_group" "securtiy_group" {
  vpc_id      = aws_vpc.main.id

  ingress {
```

```
    from_port       = 0
    to_port         = 0
    protocol        = "-1"
    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 = "tf-security_grp"
  }
}

resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main.id

  tags = {
    Name = "tf-IGW"
  }
}
#route table for public subnet with IGW
resource "aws_route_table" "table_public" {
  vpc_id = "${aws_vpc.main.id}"
 route {
    cidr_block = "0.0.0.0/0"
    gateway_id = "${aws_internet_gateway.gw.id}"
  }
  tags = {
    Name = "rt_public"
  }
}

# route table association public subnet

resource "aws_route_table_association" "association_rt_public" {
  subnet_id      = aws_subnet.public_subnet.id
  route_table_id = aws_route_table.table_public.id
}
# launch an instance
resource "aws_instance" "web" {
  ami           = "ami-0fb653ca2d3203ac1"
  instance_type = "t2.micro"
  vpc_security_group_ids= ["${aws_security_group.securtiy_group.id}"]
```

```
  subnet_id= aws_subnet.public_subnet.id
  key_name= "demo1009"
  tags = {
    Name = "TF_instance"
  }
}
```

#We are creating a terraform script for creating an instance and installing Jenkins init.

main.tf

```
provider "aws" {
    access_key = ""
    secret_key = ""
    region     = "us-east-1"
}
resource "aws_instance" "virginia" {
   ami= "ami-04505e74c0741db8d"
   instance_type = "t2.micro"
   key_name= "north-key-pair"
   user_data       = "${file("install_jenkins.sh")}"
  tags = {
    Name = "development_instance"
  }
}
```

Script file

```
sudo nano install_jenkins.sh
#!/bin/bash
sudo apt-get update
sudo apt install ca-certificates
sudo apt-get install openjdk-11-jdk -y
```

```
wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -
sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ >  /etc/apt/sources.list.d/jenkins.list'
sudo apt-get update
sudo apt-get install jenkins -y
```

VPC-2 subnet-2 instance-igw-nat-sg-with apace with hello world page

vpc.tf

```
# defining provider
provider "aws" {
 access_key = "AKIARZYTZU2OL6RKJZWB"
    secret_key = "  "
    region     = "us-west-2"
}

# creating vpc
resource "aws_vpc" "main" {
  cidr_block = "10.0.0.0/16"
  tags={
    Name = "demo_vpc_terraform"
  }
}

# creating a public subnet in vpc
resource "aws_subnet" "public_subnet" {
  vpc_id     = aws_vpc.main.id
  cidr_block = "10.0.1.0/24"
  availability_zone = "us-west-2a"
  map_public_ip_on_launch = "1"
  tags = {
    Name = "public_tf_subnet"
  }
}
```

```
# creating private subnet in vpc
resource "aws_subnet" "private_subnet" {
  vpc_id     = aws_vpc.main.id
  cidr_block = "10.0.2.0/24"
   availability_zone = "us-west-2b"
  tags = {
    Name = "private_tf_subnet"
  }
}

#creating Elastic IP
resource "aws_eip" "eip" {
  vpc      = true
  tags = {
    Name = "tf_elastic_ip"
  }
}

resource "aws_nat_gateway" "nat" {
  allocation_id = aws_eip.eip.id
  subnet_id     = aws_subnet.public_subnet.id

  tags = {
    Name = "tf_NAT"
  }

  # To ensure proper ordering, it is recommended to add an explicit dependency
  # on the Internet Gateway for the VPC.
  depends_on = [aws_internet_gateway.gw]
}



# creating a sg and associating it with the vpc
resource "aws_security_group" "securtiy_group" {
  vpc_id      = aws_vpc.main.id

  ingress {
    from_port        = 0
    to_port          = 0
    protocol         = "-1"
    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 = "tf-security_grp"
  }
}

# creating a internet gateway and associating with vpc
resource "aws_internet_gateway" "gw" {
  vpc_id = aws_vpc.main.id

  tags = {
    Name = "tf-IGW"
  }
}
#route table for public subnet with IGW
resource "aws_route_table" "table_public" {
  vpc_id = "${aws_vpc.main.id}"
 route {
    cidr_block = "0.0.0.0/0"
    gateway_id = "${aws_internet_gateway.gw.id}"
  }
  tags = {
    Name = "rt_public"
  }
}
# route table association public subnet

resource "aws_route_table_association" "association_rt_public" {
  subnet_id      = aws_subnet.public_subnet.id
  route_table_id = aws_route_table.table_public.id
}

#route table for private subnet with Nat
resource "aws_route_table" "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 = "rt_private"
  }
}
# route table association public subnet

resource "aws_route_table_association" "association_rt_private" {
  subnet_id      = aws_subnet.private_subnet.id
```

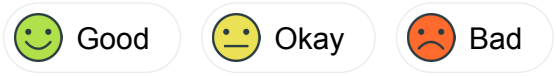
```
  route_table_id = aws_route_table.table_private.id
}

# launch an instance
resource "aws_instance" "web" {
  ami           = "ami-0892d3c7ee96c0bf7"
  instance_type = "t2.micro"
  vpc_security_group_ids= ["${aws_security_group.securtiy_group.id}"]
  subnet_id= aws_subnet.public_subnet.id
  key_name= "iam-oregon"
  user_data = "${file("./apache.sh")}"
  tags = {
    Name = "TF_public_instance"
  }
}

# launch an instance
resource "aws_instance" "db" {
  ami           = "ami-0892d3c7ee96c0bf7"
  instance_type = "t2.micro"
  vpc_security_group_ids= ["${aws_security_group.securtiy_group.id}"]
  subnet_id= aws_subnet.private_subnet.id
  key_name= "iam-oregon"
  tags = {
    Name = "TF_private_instance"
  }
}

output "IP" {
  value = aws_instance.web.public_ip
}
```

Apache.sh

```
#!/bin/bash
sudo apt-get update
sudo apt-get install -y apache2
sudo su
rm /var/www/html/index.html
echo "<html><body bgcolor="olive"><h1 align="center"> This is my Application Page<br>MAIN
PAGE<br></h1></body></html>"> /var/www/html/index.html
```

Please share your feedback to make our relationship stronger.

How would you rate our Intellipaat Support?



Please feel free to reach us @ +91-080-471-88468, +91-7022374614 (press 2), +91-080-471-86988 or revert to this mail for any further queries/concerns.

Warm regards,
Shiv

If you are unsatisfied with my response, please drop an email to escalations@intellipaat.com

Join us on Intellipaat Community for group learning with https://intellipaat.com/community/

For More Intellipaat videos please click here: https://www.youtube.com/user/intellipaaat

https://intellipaat.com/wp-content/themes/intellipaat/images/logo.png

---- On Fri, 1 Apr 2022 14:53:28 +0530 **"Mohit Rathore"<mohit.rathore4@gmail.com>** wrote ----

> hi team any update..i am waiting for your response..
>
> On Fri, Apr 1, 2022 at 7:12 AM Mohit Rathore <mohit.rathore4@gmail.com> wrote:
> > Hi team, can you please send me a script on how to create virtual machines in two different regions?
> > how to create vpc and inside vpc is one ec2 virtual machine.
> > how to create elastic ip in a machine.
> > how to install apache2 inside a newly created virtual machine...