



Assignment 1

Name: Keshav Mishra

Subject: System Provisioning and Configuration Management

Faculty: Dr. Hitesh Kumar Sharma

Batch: CSE-DevOps-B1

SAP ID: 500062098

Roll Number: R171217028

1. Make sure you have your access key and secret key. If not you can navigate to “**your security credentials**” and create one.
2. Write the script as needed to launch two ec2-instances and a VPN along with a bucket.
3. Mention your access key and secret key in the below script at the top. You have to create a **.tf** extension file and use the ami-id of the region in which you want.

```
resource "aws_instance" "Keshav_1" {
  ami = "ami-0a741b782c2c8632d"
  count=2
  key_name="KeshavMishra"
  instance_type = "t2.micro"
  tags = {
    Name = "Keshav_1"
  }
}
resource "aws_s3_bucket" "tf_course" {
  bucket = "keshav13660619999"
  acl = "private"
}
resource "aws_vpc" "vpc" {
  cidr_block = "10.0.0.0/20"
}
resource "aws_vpn_gateway" "vpn_gateway" {
  vpc_id = "vpc-04906a62"
}
resource "aws_customer_gateway" "customer_gateway" {
  bgp_asn=65000
  ip_address="172.0.0.2"
  type="ipsec.1"
}
resource "aws_vpn_connection" "main" {
  vpn_gateway_id=aws_vpn_gateway.vpn_gateway.id
  customer_gateway_id=aws_customer_gateway.customer_gateway.id
  type="ipsec.1"
  static_routes_only=true
}
```

4. After successfully writing the script, perform “**terraform init**” so that we can start working on terraform.

```
C:\Windows\System32\cmd.exe

E:\Terraform>terraform init

Initializing the backend...

Initializing provider plugins...
- Checking for available provider plugins...
- Downloading plugin for provider "aws" (hashicorp/aws) 3.16.0...

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, it is recommended to add version = "..." constraints to the
corresponding provider blocks in configuration, with the constraint strings
suggested below.

* provider.aws: version = "~> 3.16"

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.
```

5. Now as terraform init is executed successfully we can use “**terraform plan**” to see any changes.

```
E:\Terraform>terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.
```

```
-----

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
+ create
```

Terraform will perform the following actions:

```
# aws_customer_gateway.customer_gateway will be created
+ resource "aws_customer_gateway" "customer_gateway" {
  + arn          = (known after apply)
  + bgp_asn      = "65000"
  + id          = (known after apply)
  + ip_address   = "172.0.0.2"
  + type         = "ipsec.1"
}

# aws_instance.Keshav_1[0] will be created
+ resource "aws_instance" "Keshav_1" {
  + ami                  = "ami-0a741b782c2c8632d"
  + arn                  = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone     = (known after apply)
  + cpu_core_count       = (known after apply)
  + cpu_threads_per_core = (known after apply)
  + get_password_data     = false
  + host_id               = (known after apply)
  + id                   = (known after apply)
  + instance_state       = (known after apply)
  + instance_type        = "t2.micro"
  + ipv6_address_count    = (known after apply)
  + ipv6_addresses       = (known after apply)
  + key_name              = "KeshavMishra"
  + outpost_arn           = (known after apply)
  + password_data         = (known after apply)
  + placement_group       = (known after apply)
  + primary_network_interface_id = (known after apply)
  + private_dns           = (known after apply)
  + private_ip            = (known after apply)
  + public_dns            = (known after apply)
  + public_ip             = (known after apply)
  + secondary_private_ips = (known after apply)
  + security_groups       = (known after apply)
  + source_dest_check     = true
```

```

+ tags = {
  + "Name" = "Keshav_1"
}
+ tenancy = (known after apply)
+ volume_tags = (known after apply)
+ vpc_security_group_ids = (known after apply)

+ ebs_block_device {
  + delete_on_termination = (known after apply)
  + device_name = (known after apply)
  + encrypted = (known after apply)
  + iops = (known after apply)
  + kms_key_id = (known after apply)
  + snapshot_id = (known after apply)
  + volume_id = (known after apply)
  + volume_size = (known after apply)
  + volume_type = (known after apply)
}

+ ephemeral_block_device {
  + device_name = (known after apply)
  + no_device = (known after apply)
  + virtual_name = (known after apply)
}

+ metadata_options {
  + http_endpoint = (known after apply)
  + http_put_response_hop_limit = (known after apply)
  + http_tokens = (known after apply)
}

+ network_interface {
  + delete_on_termination = (known after apply)
  + device_index = (known after apply)
  + network_interface_id = (known after apply)
}

+ root_block_device {
  + delete_on_termination = (known after apply)
  + device_name = (known after apply)
  + encrypted = (known after apply)
  + iops = (known after apply)
  + kms_key_id = (known after apply)
  + volume_id = (known after apply)
  + volume_size = (known after apply)
  + volume_type = (known after apply)
}
}

```

```

# aws_instance.Keshav_1[1] will be created
+ resource "aws_instance" "Keshav_1" {
  + ami                        = "ami-0a741b782c2c8632d"
  + arn                       = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone          = (known after apply)
  + cpu_core_count             = (known after apply)
  + cpu_threads_per_core       = (known after apply)
  + get_password_data          = false
  + host_id                    = (known after apply)
  + id                         = (known after apply)
  + instance_state             = (known after apply)
  + instance_type              = "t2.micro"
  + ipv6_address_count          = (known after apply)
  + ipv6_addresses             = (known after apply)
  + key_name                    = "KeshavMishra"
  + outpost_arn                = (known after apply)
  + password_data              = (known after apply)
  + placement_group            = (known after apply)
  + primary_network_interface_id = (known after apply)
  + private_dns                 = (known after apply)
  + private_ip                 = (known after apply)
  + public_dns                 = (known after apply)
  + public_ip                  = (known after apply)
  + secondary_private_ips       = (known after apply)
  + security_groups            = (known after apply)
  + source_dest_check           = true
  + subnet_id                  = (known after apply)
  + tags                        = {
    + "Name" = "Keshav_1"
  }
  + tenancy                    = (known after apply)
  + volume_tags                = (known after apply)
  + vpc_security_group_ids     = (known after apply)

  + ebs_block_device {
    + delete_on_termination = (known after apply)
    + device_name            = (known after apply)
    + encrypted              = (known after apply)
    + iops                   = (known after apply)
    + kms_key_id             = (known after apply)
    + snapshot_id            = (known after apply)
    + volume_id              = (known after apply)
    + volume_size            = (known after apply)
    + volume_type            = (known after apply)
  }

  + ephemeral_block_device {
    + device_name = (known after apply)
  }
}

```



```

# aws_s3_bucket.tf_course will be created
+ resource "aws_s3_bucket" "tf_course" {
  + acceleration_status = (known after apply)
  + acl                 = "private"
  + arn                = (known after apply)
  + bucket              = "keshav13660619999"
  + bucket_domain_name = (known after apply)
  + bucket_regional_domain_name = (known after apply)
  + force_destroy      = false
  + hosted_zone_id     = (known after apply)
  + id                 = (known after apply)
  + region              = (known after apply)
  + request_payer      = (known after apply)
  + website_domain     = (known after apply)
  + website_endpoint   = (known after apply)

  + versioning {
    + enabled = (known after apply)
    + mfa_delete = (known after apply)
  }
}

# aws_vpc.vpc will be created
+ resource "aws_vpc" "vpc" {
  + arn = (known after apply)
  + assign_generated_ipv6_cidr_block = false
  + cidr_block = "10.0.0.0/20"
  + 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_classiclink = (known after apply)
  + enable_classiclink_dns_support = (known after apply)
  + enable_dns_hostnames = (known after apply)
  + enable_dns_support = true
  + id = (known after apply)
  + instance_tenancy = "default"
  + ipv6_association_id = (known after apply)
  + ipv6_cidr_block = (known after apply)
  + main_route_table_id = (known after apply)
  + owner_id = (known after apply)
}

```

```

+ tunnel1_vgw_inside_address = (known after apply)
+ tunnel2_address = (known after apply)
+ tunnel2_bgp_asn = (known after apply)
+ tunnel2_bgp_holdtime = (known after apply)
+ tunnel2_cgw_inside_address = (known after apply)
+ tunnel2_inside_cidr = (known after apply)
+ tunnel2_preshared_key = (sensitive value)
+ tunnel2_vgw_inside_address = (known after apply)
+ type = "ipsec.1"
+ vgw_telemetry = (known after apply)
+ vpn_gateway_id = (known after apply)
}

```

```

# aws_vpn_gateway.vpn_gateway will be created
+ resource "aws_vpn_gateway" "vpn_gateway" {
  + amazon_side_asn = (known after apply)
  + arn = (known after apply)
  + id = (known after apply)
  + vpc_id = "vpc-04906a62"
}

```

Plan: 7 to add, 0 to change, 0 to destroy.

6. As we can see the terraform plan has been executed and it shows that there will be 7 changes done. Now we can go for **“terraform apply”**.

```
E:\Terraform>terraform apply
```

```
An execution plan has been generated and is shown below.  
Resource actions are indicated with the following symbols:  
+ create
```

```
Terraform will perform the following actions:
```

```
# aws_customer_gateway.customer_gateway will be created  
+ resource "aws_customer_gateway" "customer_gateway" {  
  + arn          = (known after apply)  
  + bgp_asn      = "65000"  
  + id          = (known after apply)  
  + ip_address   = "172.0.0.2"  
  + type        = "ipsec.1"  
}  
  
# aws_instance.Keshav_1[0] will be created  
+ resource "aws_instance" "Keshav_1" {  
  + ami                  = "ami-0a741b782c2c8632d"  
  + arn                  = (known after apply)  
  + associate_public_ip_address = (known after apply)  
  + availability_zone     = (known after apply)  
  + cpu_core_count       = (known after apply)  
  + cpu_threads_per_core = (known after apply)  
  + get_password_data     = false  
  + host_id              = (known after apply)  
  + id                   = (known after apply)  
  + instance_state       = (known after apply)  
  + instance_type        = "t2.micro"  
  + ipv6_address_count    = (known after apply)  
  + ipv6_addresses       = (known after apply)  
  + key_name              = "KeshavMishra"  
  + outpost_arn          = (known after apply)  
  + password_data        = (known after apply)  
  + placement_group      = (known after apply)  
  + primary_network_interface_id = (known after apply)  
  + private_dns          = (known after apply)  
  + private_ip           = (known after apply)  
  + public_dns           = (known after apply)  
  + public_ip            = (known after apply)  
  + secondary_private_ips = (known after apply)  
  + security_groups       = (known after apply)  
  + source_dest_check     = true  
  + subnet_id            = (known after apply)  
  + tags                 = {  
    + "Name" = "Keshav_1"  
  }  
  + tenancy              = (known after apply)  
  + volume_tags          = (known after apply)  
  + vpc_security_group_ids = (known after apply)
```



```

# aws_instance.Keshav_1[1] will be created
+ resource "aws_instance" "Keshav_1" {
  + ami                      = "ami-0a741b782c2c8632d"
  + arn                     = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone        = (known after apply)
  + cpu_core_count           = (known after apply)
  + cpu_threads_per_core     = (known after apply)
  + get_password_data        = false
  + host_id                  = (known after apply)
  + id                       = (known after apply)
  + instance_state           = (known after apply)
  + instance_type            = "t2.micro"
  + ipv6_address_count       = (known after apply)
  + ipv6_addresses           = (known after apply)
  + key_name                  = "KeshavMishra"
  + outpost_arn              = (known after apply)
  + password_data            = (known after apply)
  + placement_group          = (known after apply)
  + primary_network_interface_id = (known after apply)
  + private_dns              = (known after apply)
  + private_ip               = (known after apply)
  + public_dns               = (known after apply)
  + public_ip                = (known after apply)
  + secondary_private_ips    = (known after apply)
  + security_groups           = (known after apply)
  + source_dest_check        = true
  + subnet_id                = (known after apply)
  + tags                      = {
    + "Name" = "Keshav_1"
  }
  + tenancy                  = (known after apply)
  + volume_tags              = (known after apply)
  + vpc_security_group_ids   = (known after apply)

  + ebs_block_device {
    + delete_on_termination = (known after apply)
    + device_name           = (known after apply)
    + encrypted             = (known after apply)
    + iops                  = (known after apply)
    + kms_key_id            = (known after apply)
    + snapshot_id           = (known after apply)
    + volume_id             = (known after apply)
    + volume_size           = (known after apply)
    + volume_type           = (known after apply)
  }
}

```

7. If we would have used **-auto-approve** then it would not have asked for the confirmation of these tasks but now we have to pass “**yes**” for further execution.

```

Plan: 7 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

```

```

aws_vpc.vpc: Creating...

```

```

aws_vpn_gateway.vpn_gateway: Creating...

```

```

aws_customer_gateway.customer_gateway: Creating...

```

```

aws_instance.Keshav_1[1]: Creating...

```

```

aws_s3_bucket.tf_course: Creating...

```

```

aws_customer_gateway.customer_gateway: Still creating... [10s elapsed]
aws_s3_bucket.tf_course: Still creating... [10s elapsed]
aws_customer_gateway.customer_gateway: Creation complete after 15s [id=cgw-049368113bdcbe0f4]
aws_vpn_gateway.vpn_gateway: Creation complete after 18s [id=vgw-083146699a1f95ca1]
aws_vpn_connection.main: Creating...
aws_instance.Keshav_1[1]: Still creating... [20s elapsed]
aws_vpc.vpc: Still creating... [20s elapsed]
aws_instance.Keshav_1[0]: Still creating... [20s elapsed]
aws_s3_bucket.tf_course: Still creating... [20s elapsed]
aws_vpc.vpc: Creation complete after 24s [id=vpc-0ffaf5fad677b313a]
aws_vpn_connection.main: Still creating... [10s elapsed]
aws_instance.Keshav_1[0]: Still creating... [30s elapsed]
aws_s3_bucket.tf_course: Still creating... [30s elapsed]
aws_instance.Keshav_1[1]: Still creating... [30s elapsed]
aws_vpn_connection.main: Still creating... [20s elapsed]
aws_s3_bucket.tf_course: Still creating... [40s elapsed]
aws_instance.Keshav_1[1]: Still creating... [40s elapsed]
aws_instance.Keshav_1[0]: Still creating... [40s elapsed]
aws_s3_bucket.tf_course: Creation complete after 43s [id=keshav13660619999]
aws_vpn_connection.main: Still creating... [30s elapsed]
aws_instance.Keshav_1[0]: Still creating... [50s elapsed]
aws_instance.Keshav_1[1]: Still creating... [50s elapsed]
aws_instance.Keshav_1[0]: Creation complete after 51s [id=i-0d948cdcb9cb2b969]
aws_vpn_connection.main: Still creating... [40s elapsed]
aws_instance.Keshav_1[1]: Still creating... [1m0s elapsed]
aws_vpn_connection.main: Still creating... [50s elapsed]
aws_instance.Keshav_1[1]: Still creating... [1m10s elapsed]
aws_vpn_connection.main: Still creating... [1m0s elapsed]
aws_instance.Keshav_1[1]: Creation complete after 1m20s [id=i-0b66596b6ee64c3a7]
aws_vpn_connection.main: Still creating... [1m10s elapsed]
aws_vpn_connection.main: Still creating... [1m20s elapsed]
aws_vpn_connection.main: Still creating... [1m30s elapsed]
aws_vpn_connection.main: Still creating... [1m40s elapsed]
aws_vpn_connection.main: Still creating... [1m50s elapsed]
aws_vpn_connection.main: Still creating... [2m0s elapsed]
aws_vpn_connection.main: Still creating... [2m10s elapsed]
aws_vpn_connection.main: Still creating... [2m20s elapsed]
aws_vpn_connection.main: Still creating... [2m30s elapsed]
aws_vpn_connection.main: Still creating... [2m40s elapsed]
aws_vpn_connection.main: Still creating... [2m50s elapsed]
aws_vpn_connection.main: Still creating... [3m0s elapsed]
aws_vpn_connection.main: Still creating... [3m10s elapsed]
aws_vpn_connection.main: Still creating... [3m20s elapsed]
aws_vpn_connection.main: Still creating... [3m30s elapsed]
aws_vpn_connection.main: Still creating... [3m40s elapsed]
aws_vpn_connection.main: Still creating... [3m50s elapsed]
aws_vpn_connection.main: Still creating... [4m0s elapsed]
aws_vpn_connection.main: Creation complete after 4m9s [id=vpn-0fbdef7c4c3a3fc5f]

Apply complete! Resources: 7 added, 0 changed, 0 destroyed.

```

As we can see the resources have been added from the above screenshot.

8. Now let's verify whether these resources have been created or not.

a) First let's verify ec2-instances.

Instances (2) Info

🔄

Connect

Instance state ▾

Actions ▾

Launch instances ▾

🔍 Filter instances

< 1 > ⚙️

Instance state: running ✕

Clear filters

<input type="checkbox"/>	Name ▾	Instance ID	Instance state ▾	Instance type ▾	Status check	Alarm status	Availability Zone ▾	Public IPv4 DNS
<input type="checkbox"/>	Keshav_1	i-0b66596b6ee64c3a7	🟢 Running 🔍	t2.micro	🟢 2/2 checks ... 🔍	No alarms +	us-west-1c	ec2-54-193-32-
<input type="checkbox"/>	Keshav_1	i-0d948cdcb9cb2b969	🟢 Running 🔍	t2.micro	🟢 2/2 checks ... 🔍	No alarms +	us-west-1c	ec2-50-18-4-68

<

>

b) Now let's verify the VPN.

Create VPN Connection Download Configuration Actions						
Filter by tags and attributes or search by keyword						
	Name	VPN ID	State	Virtual Private Gateway	Transit Gateway	Customer Gateway
		vpn-0fb93d66ef28338bd	deleted	vgw-0404b5581ab2b9998	-	cgw-0c11a2849d13c85d8
		vpn-0fbdef7c4c3a3fc5f	available	vgw-083146699a1f95ca1	-	cgw-049368113bdcbe0f4

Here we can see a VPN has been created successfully.

c) Now let's verify the bucket.

Amazon S3

Buckets

Access points

Batch Operations

Access analyzer for S3

Account settings for Block Public Access

Storage Lens

Buckets (1)

Find buckets by name

Name	Region	Access	Creation date
keshav13660619999	US West (N. California) us-west-1	Objects can be public	November 20, 2020, 19:34 (UTC+05:30)

We successfully completed the whole task assigned.