Terraform State Commands

Lokeshkumar

```
>_

$ vi terraform.tfstate

$ terraform state show aws_s3_bucket.finance

# terraform state <subcommand> [options]
[args]
```

Sub-command
list
mv
pull
rm
show

terraform.tfstate


```
{
      "mode": "managed",
      "type": "aws_instance",
      "name": "dev-ec2",
      "provider": "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-0a634ae95e11c6f91",
.
.
.
            "primary_network_interface_id": "eni-0ccd57b1597e633e0",
            "private_dns": "ip-172-31-7-21.us-west-2.compute.internal",
            "private_ip": "172.31.7.21",
            "public_dns": "ec2-54-71-34-19.us-west-2.compute.amazonaws.com",
            "public_ip": "54.71.34.19",
            "root_block_device": [
              {
                "delete_on_termination": true,
                "device_name": "/dev/sda1",
                "encrypted": false,
                "iops": 100,
                "kms_key_id": "",
                "volume_id": "vol-070720a3636979c22",
                "volume_size": 8,
                "volume_type": "gp2"
              }
            ],
```

```
>_

# terraform state list [options] [address]

$ terraform state list
aws_dynamodb_table.cars
aws_s3_bucket.finance-2020922

$ terraform state list aws_s3_bucket.finance-2020922

aws_s3_bucket.finance-2020922
```

```
>_

# terraform state show [options] [address]

$ terraform state show aws_s3_bucket.finance-

2020922 resource "aws_s3_bucket" "terraform-state"
{   acl                         = "private"
    arn                         = "arn:aws:s3::: finance-2020922 "
    bucket                      = "finance-2020922 "
    bucket_domain_name          = "finance-2020922.s3.amazonaws.com"
    bucket_regional_domain_name = " finance-2020922.s3.us-west-
    force_destroy               = false
    hosted_zone_id              = "Z2F5ABCDE1ACD"
    id                          = "finance-2020922 "
    region                      = "us-west-1"
    request_payer               = "BucketOwner"
    tags                        = {
        "Descritpion" = "Bucket to store Finance and Payroll
        Information"
    }

    versioning {
        enabled    = false
        mfa_delete = false
    }
}
```

main.tf

```
resource "aws_dynamodb_table" "state-locking"
    { name = "state-locking-db"
    billing_mode = "PAY_PER_REQUEST"
    hash_key = "LockID"
    attribute {
        name =  "LockID"
        type = "S"
    }
}
```

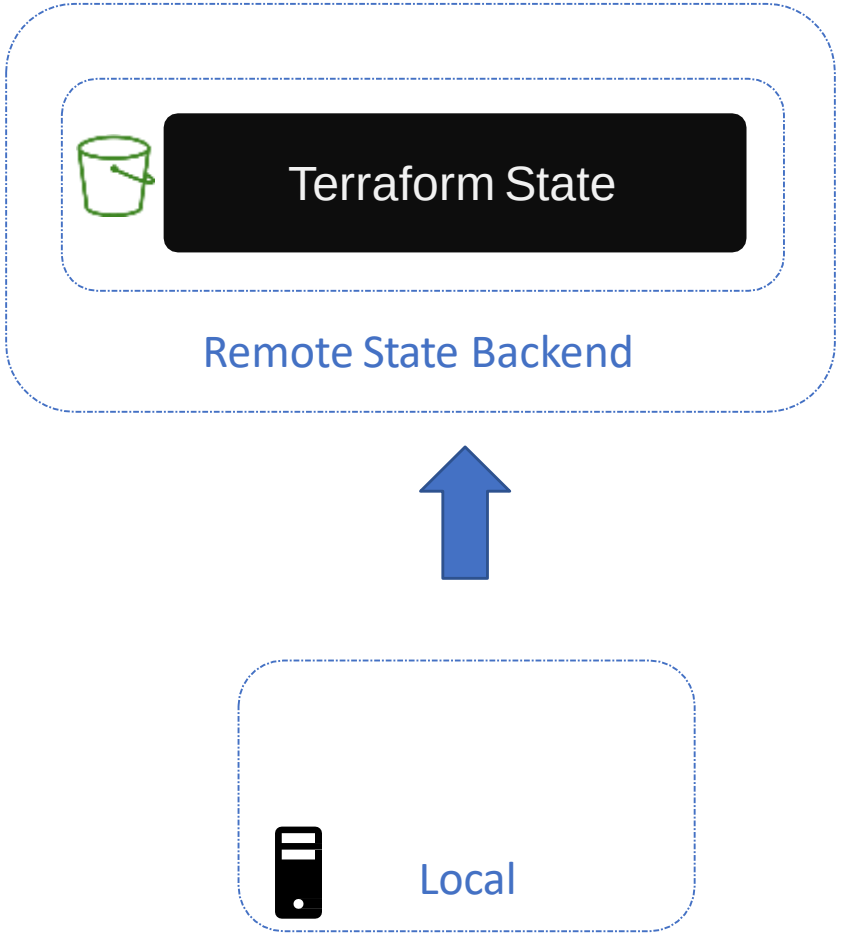
terraform.tfstate

```
"resources": [
    {
      "mode": "managed",
      "type": "aws_dynamodb_table",
      "name": "state-locking-db"
      "provider":
"provider[\"registry.terraform.io/hashicorp/aws\"
]",
.
.
```

>_

```
# terraform state mv [options] SOURCE  DESTINATION

$ terraform state mv aws_dynamodb_table.state-locking aws_dynamodb_table.state-locking-
db Move "aws_dynamodb_table.state-locking" to "aws_dynamodb_table.state-locking-db"
Successfully moved 1 object(s).

$ terraform apply

aws_dynamodb_table.state-locking-db: Refreshing state... [id=state-locking]

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

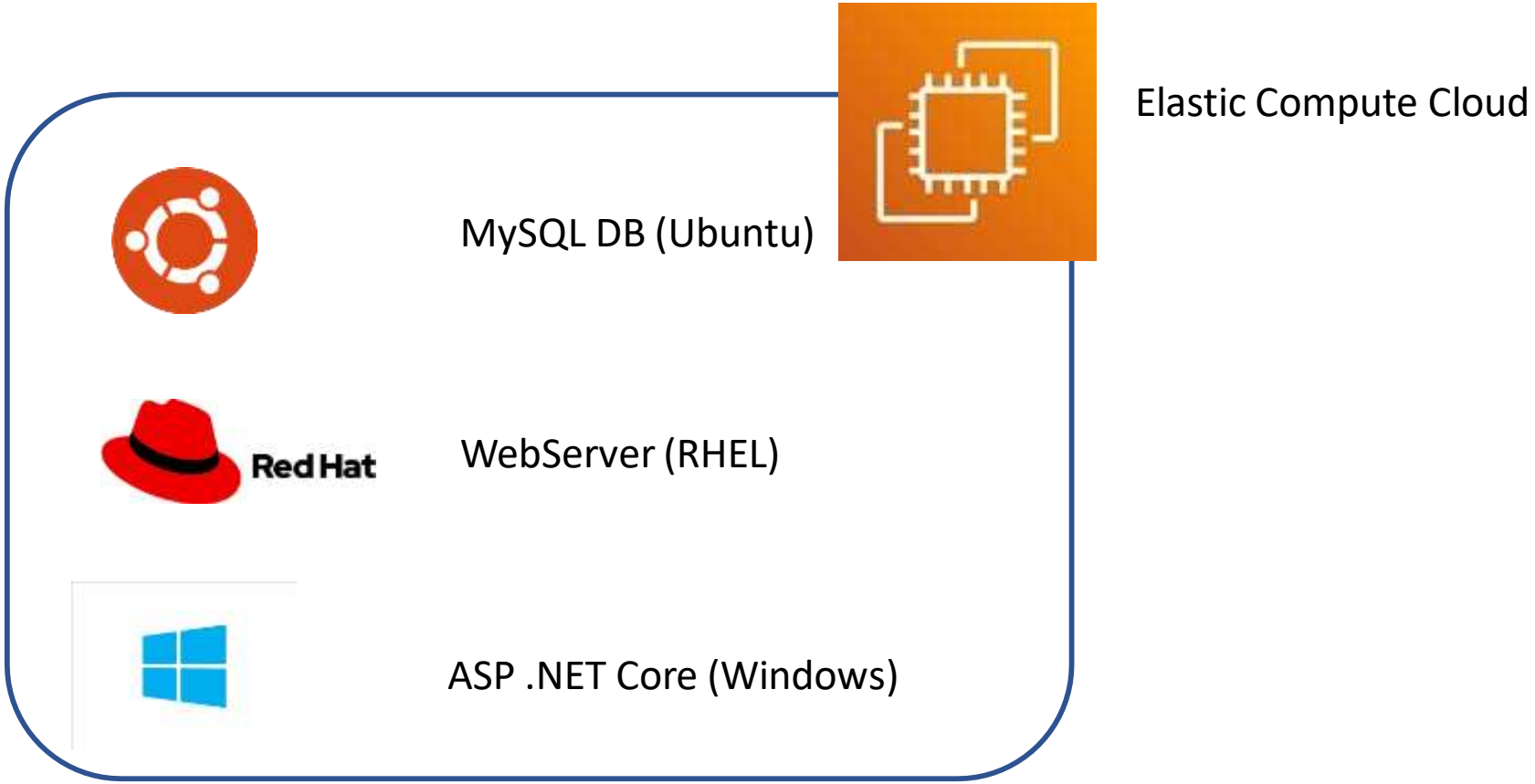
```
>_
$ ls
main.tf  provider.tf
#  terraform state pull [options] SOURCE  DESTINATION
$ terraform state pull
{
  "version": 4,
  "terraform_version": "0.13.0",
  "serial": 0,
  "lineage": "b6e2cf0e-ef8d-3c59-1e11-c6520dcd745c",
  "resources": [
    {
      "mode": "managed",
      "type": "aws_dynamodb_table",
      "name": "state-locking-db",
      "provider":
      "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
...
$ terraform state pull | jq '.resources[] | select(.name  == "state-locking-
db")|.instances[].attributes.hash_key'
"LockID"
```



```
>_

#  terraform state rm ADDRESS

$ terraform state rm aws_s3_bucket.finance-2020922

Acquiring state lock. This may take a few
moments... Removed aws_s3_bucket.finance-2020922
Successfully removed 1 resource instance(s).
Releasing state lock. This may take a few
moments...
```

Introduction to AWS EC2

Lokeshkumar



Amazon Machine Image (AMI‘s)

Amazon Linux 2 AMI ami-0c2f25c1f66a1ff4d

Red Hat Enterprise Linux 8 ami-04312317b9c8c4b51

Ubuntu Server 20.04 LTS ami-0edab43b6fa892279

Instance Types

General Purpose

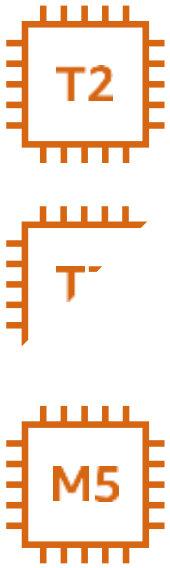
Compute Optimized

Memory Optimized

https://aws.amazon.com/ec2/instance-types/

T2 General Purpose

Instance Type	vCPU	Memory (GB)
t2.nano	1	0.5
t2.micro	1	1
t2.small	1	2
t2.medium	2	4
t2.large	2	8
t2.xlarge	4	16
t2.2xlarge	8	32



https://aws.amazon.com/ebs/volume-types/

EBS Volume

EBS Volume Types

Name	Type	Description
io1	SSD	For business-critical Apps
io2	SSD	For latency-sensitive transactional workloads
gp2	SSD	General Purpose
st1	HDD	Low Cost HDD frequently accessed, throughput-intensive workloads
sc1	HDD	Lowest cost HDD volume designed for less frequently accessed workloads

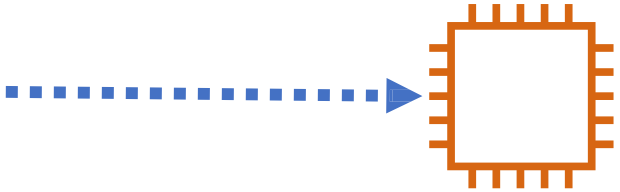
User Data

```
#!/bin/bash
sudo apt update
sudo apt install nginx
systemctl enable nginx
systemctl start nginx
```

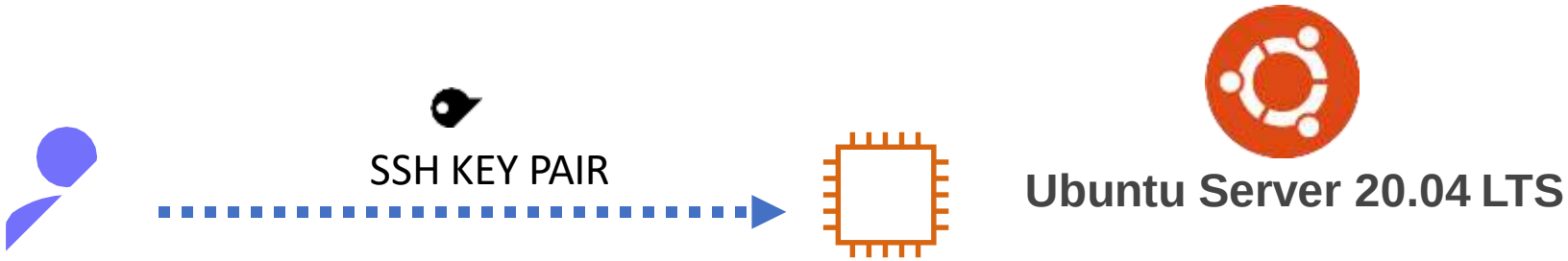
Ubuntu Web Server

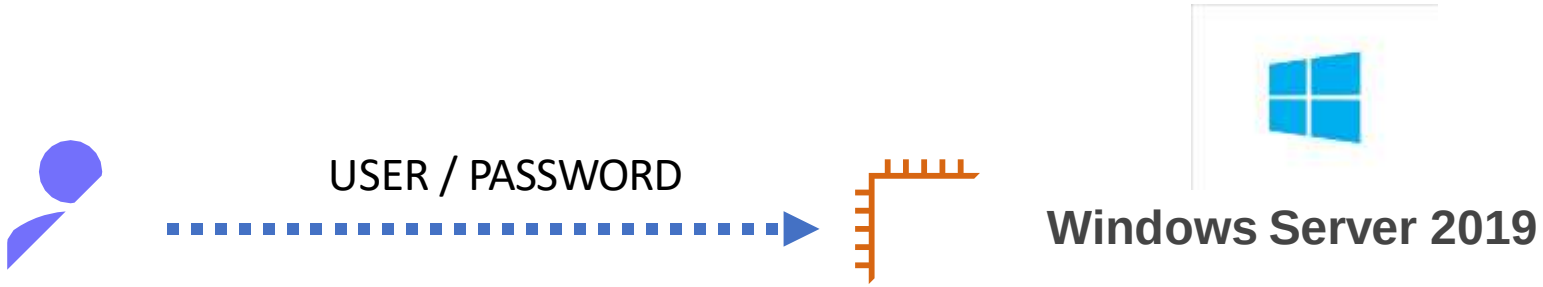
PowerShell PS1

Batch .bat



Windows Instance





AWS EC2 WITH Terraform

Lokeshkumar

main.tf

```
resource "aws_instance"  "webserver"  {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
}
```

Argument Reference

The following arguments are supported:

- `ami` - (Required) The AMI to use for the instance.
- `instance_type` - (Required) The type of instance to start. Updates to this field will trigger a stop/start of the EC2 instance.
- `tags` - (Optional) A map of tags to assign to the resource.

provider.tf

```
provider "aws" " {
  region = “us-west-1"
}
```

Lokeshkumar

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name        = "webserver"
    Description = "An Nginx WebServer on Ubuntu"
  }

}
```

provider.tf

```
provider "aws" " {
  region = “us-west-1"
}
```

Argument Reference

The following arguments are supported:

- `ami` - (Required) The AMI to use for the instance.
- `instance_type` - (Required) The type of instance to start. Updates to this field will trig a stop/start of the EC2 instance.
- `tags` - (Optional) A map of tags to assign to the resource.

Lokeshkumar

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name        = "webserver"
    Description = "An Nginx WebServer on Ubuntu"
  }
  user_data = <<-EOF
              #!/bin/bash
              sudo apt update
              sudo apt install nginx -y
              systemctl enable nginx
              systemctl start nginx
              EOF
}
```

provider.tf

```
provider "aws" " {
  region = “us-west-1"
}
```

Argument Reference

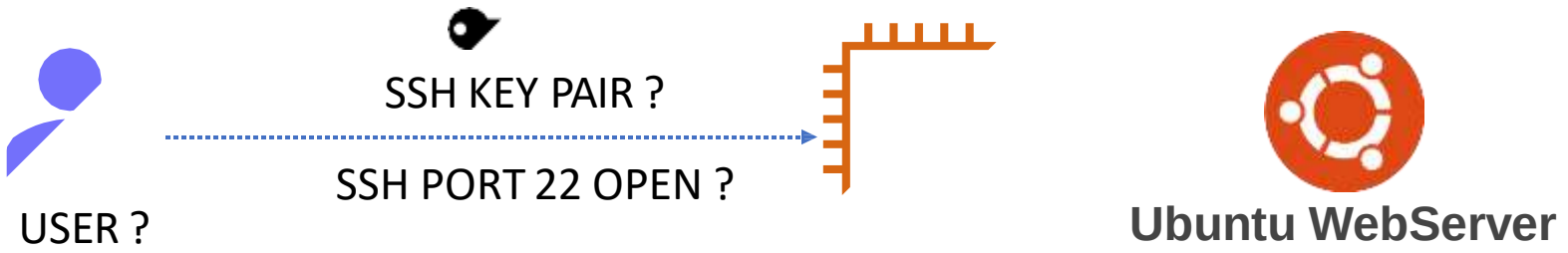
The following arguments are supported:

- `ami` - (Required) The AMI to use for the instance.
- `instance_type` - (Required) The type of instance to start. Updates to this field will trig a stop/start of the EC2 instance.
- `tags` - (Optional) A map of tags to assign to the resource.
- `user_data` - (Optional) The user data to provide when launching the instance. Do n pass gzip-compressed data via this argument; see `user_data_base64` instead.

Lokeshkumar

```
>_

$ terraform apply

  # aws_instance.webserver will be created
  + resource "aws_instance" "webserver" {
      + ami                          = "ami-0edab43b6fa892279"
.
.
      + instance_type                = "t2.micro"
      + ipv6_address_count           = (known after apply)
      + public_ip                    = (known after apply)
      + source_dest_check            = true
      + subnet_id                    = (known after apply)
      + tags                         = {
          + "Description" = "An NGINX WebServer on Ubuntu"
          + "Name"        = "webserver"
        }
      + tenancy                      = (known after apply)
      + user_data                    = "527516162d9d8675a26b6ca97664226e6e2bff82"
      + volume_tags                  = (known after apply)
      + vpc_security_group_ids       = (known after apply)
.
.
aws_instance.webserver: Creating...
aws_instance.webserver: Still creating... [20s elapsed]
aws_instance.webserver: Creation complete after 22s [id=i-0085e5d0f442f7c4f]

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

Name	Instance ID	Instance state	Instance type	Status check	Alarm Status	Availability zone
webserver	i-0085e5d0f442f7c4f	Running	t2.micro	2/2 checks ...	No alarms +	ca-central-1a



Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.

Create a new key pair

Key pair name

webserver

Download Key Pair

You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

Cancel | Launch Instances

Lokeshkumar

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name        = "webserver"
    Description = "An Nginx WebServer on Ubuntu"
  }
  user_data = <<-EOF
              #!/bin/bash
              sudo apt update
              sudo apt install nginx -y
              systemctl enable nginx
              systemctl start nginx
              EOF
}
```

main.tf

```
resource "aws_instance"  "webserver"  {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name        = "webserver"
    Description = "An Nginx WebServer on Ubuntu"
  }
  user_data = <<-EOF
              #!/bin/bash
              sudo apt update
              sudo apt install nginx -y
              systemctl enable nginx
              systemctl start nginx
              EOF
}

resource "aws_key_pair" "web" {
    public_key = file("/root/.ssh/web.pub")
}
```

main.tf

```
resource "aws_instance" "webserver" {
    ami = "ami-0edab43b6fa892279"
    instance_type = "t2.micro"
    tags = {
        Name = "webserver"
        Description = "An NGINX WebServer on Ubuntu"
    }
    user_data =<<-EOF
    #!/bin/bash
    sudo apt update
    sudo apt install nginx -y
    systemctl enable nginx
    systemctl start nginx
    EOF
}
resource "aws_key_pair" "web" {
    public_key = "ssh-
rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQDicpU+kT9isaZy7cHYa
+oCTUolS6Tg6vCEq+ufucIMrA7RLTngi+YfTfvgrY2UiHGxuuJ1lE
yT0x2UrGexVx4G2TzX/am2WFzNbcGSg2bCXTkVQY93KOhbW9y851a
+g1wI7TODC0oxEMFr/CVsrJ4bfbp8S896VKBxC1WpSU9GscPP28GV
uDgm2ATBuL78AF root@iac-server"
}
```

Lokeshkumar

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name        = "webserver"
    Description = "An Nginx WebServer on Ubuntu"
  }
  user_data = <<-EOF
              #!/bin/bash
              sudo apt update
              sudo apt install nginx -y
              systemctl enable nginx
              systemctl start nginx
              EOF
  key_name  = aws_key_pair.web.id
}

resource "aws_key_pair" "web" {
    public_key = file("/root/.ssh/web.pub")
}
```

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the low

Number of instances	1	Launch into Auto Scaling Group
Purchasing option	☐ Request Spot instances	
Network	vpc-7da8d215 (default)	Create new VPC
Subnet	No preference (default subnet in any Availability Zon	Create new subnet
Auto-assign Public IP	Use subnet setting (Enable)	

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.

Assign a security group: ◉ Create a **new** security group

○ Select an **existing** security group

Security group name: ssh-access

Description: SSH Access from the Internet

Type	Protocol	Port Range	Source
SSH	TCP	22	Custom 0.0.0.0/0

Add Rule

Lokeshkumar

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name        = "webserver"
    Description = "An Nginx WebServer on Ubuntu"
  }
  user_data = <<-EOF
              #!/bin/bash
              sudo apt update
              sudo apt install nginx -y
              systemctl enable nginx
              systemctl start nginx
              EOF

  key_name  = aws_key_pair.web.id

}
resource "aws_key_pair" "web" {
    public_key = file("/root/.ssh/web.pub")
}
```

```
  }
  user_data = <<-EOF
              #!/bin/bash
              sudo apt update
              sudo apt install nginx -y
              systemctl enable nginx
              systemctl start nginx
              EOF
  key_name  = aws_key_pair.web.id
}
resource "aws_key_pair" "web" {
    public_key = file("/root/.ssh/web.pub")
}
resource "aws_security_group" "ssh-access" {
  name        = "ssh-access"
  description = "Allow SSH access from the Internet"
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}
```

The `ingress` block supports:

- `cidr_blocks` - (Optional) List of CIDR blocks.
- `to_port` - (Required) The end range port (or ICMP code if protocol is "icmp"
- `from_port` - (Required) The start port (or ICMP type number if protocol is "ic "icmpv6")
- `protocol` - (Required) The protocol. If you select a protocol of "-1" (semanticall equivalent to `"all"`, which is not a valid value here), you must specify a "from_ and "to_port" equal to 0. If not icmp, icmpv6, tcp, udp, or "-1" use the protocol

```
  user_data = <<-EOF
              #!/bin/bash
              sudo apt update
              sudo apt install nginx -y
              systemctl enable nginx
              systemctl start nginx
              EOF

  key_name  = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id ]

}
resource "aws_key_pair" "web" {
    public_key = file("/root/.ssh/web.pub")
}
resource "aws_security_group" "ssh-access" {
  name        = "ssh-access"
  description = "AllowSSH access from the Internet"
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}
```

```
              systemctl start nginx
              EOF

  key_name  = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id ]

}
resource "aws_key_pair" "web" {
    public_key = file("/root/.ssh/web.pub")
}
resource "aws_security_group" "ssh-access" {
  name        = "ssh-access"
  description = "AllowSSH access from the Internet"
  ingress {
    from_port   = 22
    to_port     = 22
    protocol    = "tcp"
    cidr_blocks = ["0.0.0.0/0"]
  }
}

 output publicip {
   value       = aws_instance.webserver.public_ip
 }
```

```
$ terraform apply

Plan: 3 to add, 0 to change, 1 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_instance.webserver: Destroying... [id=i-015b579d0ea84fbb7]
aws_key_pair.web: Creating...
aws_security_group.ssh-access: Creating...
aws_key_pair.web: Creation complete after 1s [id=terraform-
20201014034144926200000001]
aws_security_group.ssh-access: Creation complete after 1s [id=sg-
0f02f3ea92b14bed8]
aws_instance.webserver: Still destroying... [id=i-015b579d0ea84fbb7, 10s elapsed]
aws_instance.webserver: Still destroying... [id=i-015b579d0ea84fbb7, 20s elapsed]
aws_instance.webserver: Destruction complete after 30s
aws_instance.webserver: Creating...
aws_instance.webserver: Still creating... [10s elapsed]
aws_instance.webserver: Still creating... [20s elapsed]
aws_instance.webserver: Still creating... [30s elapsed]
aws_instance.webserver: Creation complete after 32s [id=i-0fd2c1c5eb0762ff5]

Apply complete! Resources: 3 added, 0 changed, 1 destroyed.
Outputs:

publicip = 3.96.203.171
```

```
$ ssh -i /root/.ssh/web ubuntu@3.96.203.171
ubuntu@ip-172-31-19-161:~$

[ubuntu@ip-172-31-19-161]$ 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 Wed 2020-11-02 22:17:38 UTC;
2 min ago
  Process: 303 ExecStart=/usr/sbin/nginx -g daemon on;
master_process on; (code=exited, status=0
  Process: 264 ExecStartPre=/usr/sbin/nginx -t -q -g daemon
on; master_process on; (code=exited,
 Main PID: 304 (nginx)
```

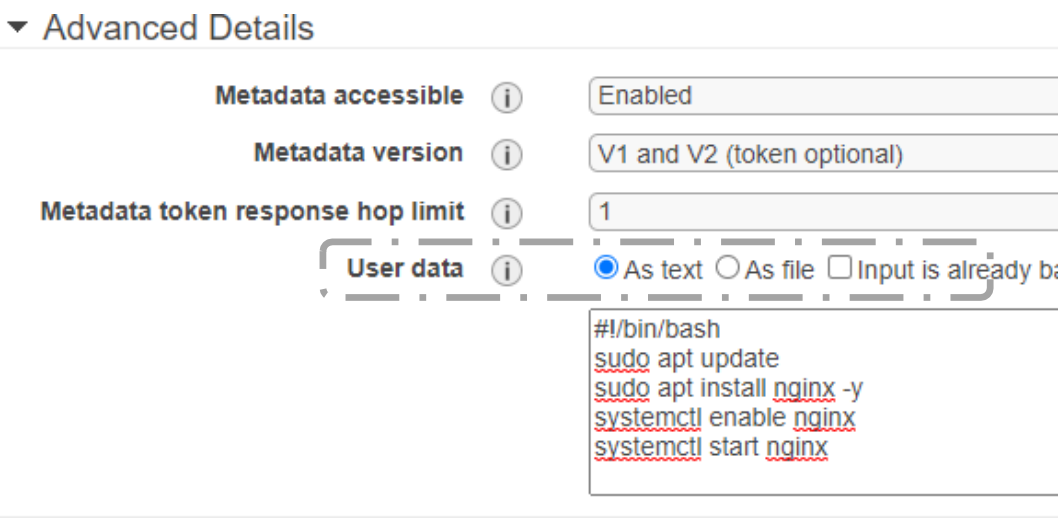
Terraform Provisioners

Lokeshkumar

Provisioners

main.tf

```
resource "aws_instance"  "webserver"  {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  user_data = <<-EOF
              #!/bin/bash
              sudo apt update
              sudo apt install nginx -y
              systemctl enable nginx
              systemctl start nginx
              EOF
  key_name  = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id
  ]

}

resource "aws_key_pair" "web" {
     << code hidden >>
}

resource "aws_security_group" "ssh-access" {
     << code hidden >>
}
```



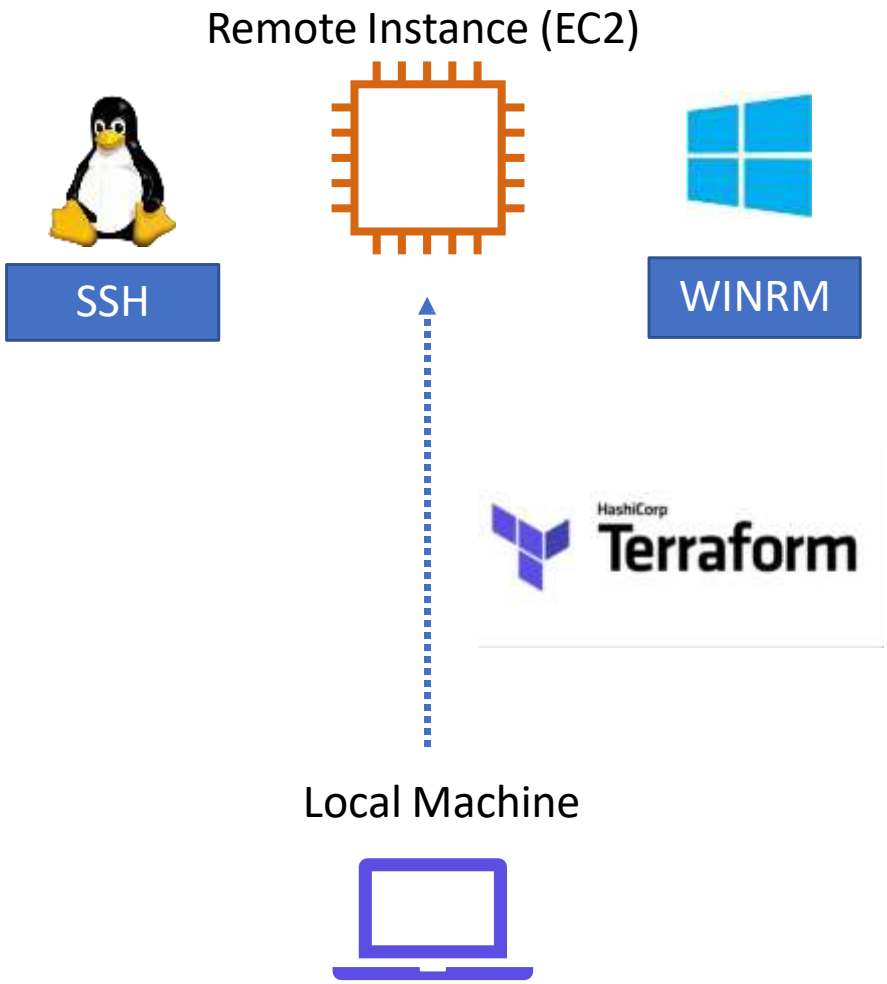
Lokeshkumar

Remote Exec

main.tf

```
resource "aws_instance" "webserver" {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  user_data = <<-EOF
              #!/bin/bash
              sudo apt update
              sudo apt install nginx -y
              systemctl enable nginx
              systemctl start nginx
              EOF
  key_name  = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id
  ]
}

resource "aws_key_pair" "web" {
     << code hidden >>
}

resource "aws_security_group" "ssh-access" {
     << code hidden >>
}
```

```
apt update
apt install nginx -y
systemctl enable nginx
systemctl start nginx
```



- ✓ Network Connectivity (Security Group)
- ✓ Authentication (SSH Key Pair)

Remote Exec

main.tf

```
resource "aws_instance"  "webserver"  {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  provisioner "remote-exec" {
      inline = [ "sudo apt update",
                 "sudo apt install nginx -y",
                 "sudo systemctl enable nginx",
                 "sudo systemctl start nginx",
               ]
    }
  key_name  = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id
  ]

}

resource "aws_key_pair" "web" {
     << code hidden >>
}

resource "aws_security_group" "ssh-access" {
     << code hidden >>
}
```

Lokeshkumar

Remote Exec

main.tf

```
resource "aws_instance"  "webserver"  {
  ami            = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  provisioner "remote-exec" {
      inline = [ "sudo apt update",
                 "sudo apt install nginx -y",
                 "sudo systemctl enable nginx",
                 "sudo systemctl start nginx",
               ]
    }
  connection {
     type         = "ssh"
     host         = self.public_ip
     user         = "ubuntu"
     private_key = file("/root/.ssh/web")
   }
  key_name  = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id
} ]
resource "aws_key_pair" "web" {
     << code hidden >>
}
```

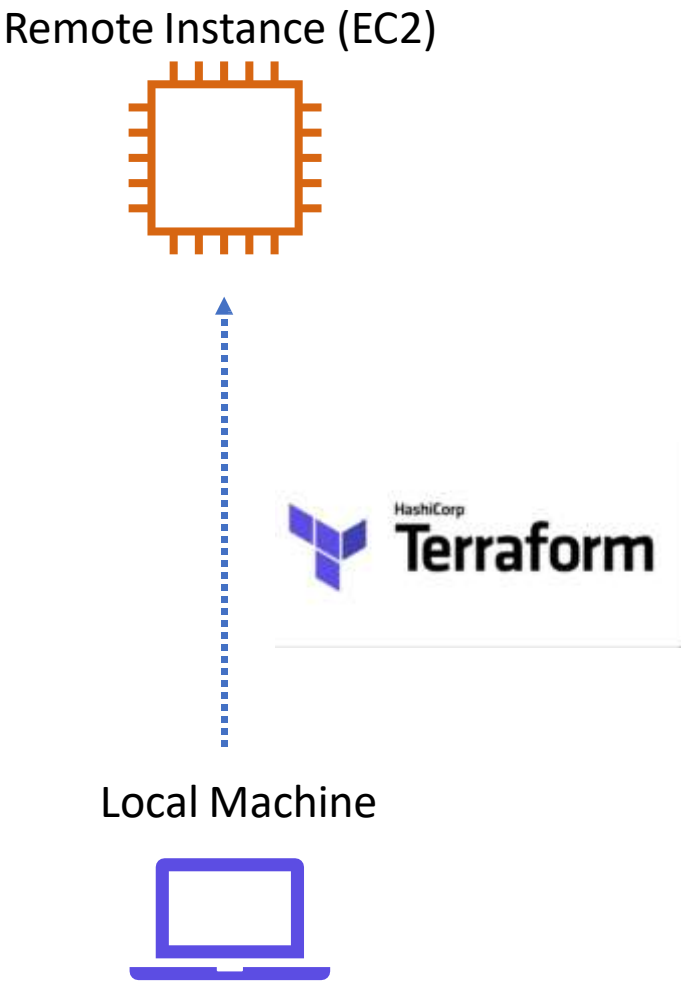
>_

```
$ terraform apply
aws_key_pair.web: Creating...
aws_security_group.ssh-access: Creating...
aws_key_pair.web: Creation complete after 0s [id=terraform-
20201015013048509100000001]
aws_security_group.ssh-access: Creation complete after 1s [id=sg-0
aws_instance.webserver:  Creating...
aws_instance.webserver: Still creating... [10s elapsed]
aws_instance.webserver: Still creating... [20s elapsed]
aws_instance.webserver: Still creating... [30s elapsed]
aws_instance.webserver: Provisioning with 'remote-exec'...
aws_instance.webserver (remote-exec): Connecting to remote host vi
aws_instance.webserver  (remote-exec):   Host: 3.96.136.157
aws_instance.webserver  (remote-exec):   User: ubuntu
aws_instance.webserver  (remote-exec):   Password: false
aws_instance.webserver  (remote-exec):   Private key: true
aws_instance.webserver  (remote-exec):   Certificate: false
aws_instance.webserver  (remote-exec):   SSH Agent: false
aws_instance.webserver (remote-exec):    Checking Host Key: false
aws_instance.webserver: Still creating... [40s elapsed]
aws_instance.webserver (remote-exec): Connecting to remote host vi
aws_instance.webserver  (remote-exec):   Host: 3.96.136.157
aws_instance.webserver  (remote-exec):   User: ubuntu
aws_instance.webserver  (remote-exec):   Password: false
aws_instance.webserver  (remote-exec):   Private key: true
aws_instance.webserver  (remote-exec):   Certificate: false
aws_instance.webserver  (remote-exec):   SSH Agent: false
aws_instance.webserver (remote-exec):    Checking Host Key: false
aws_instance.webserver (remote-exec): Connected!
aws_instance.webserver: Still creating... [50s elapsed]
aws_instance.webserver: Creation complete after 50s [id=i-068fad30
```

Local Exec

main.tf

```
resource "aws_instance"  "webserver"  {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  provisioner "remote-exec" {
      inline = [ "sudo apt update",
                 "sudo apt install nginx -y",
                 "sudo systemctl enable nginx",
                 "sudo systemctl start nginx",
               ]
    }
  connection {
     type        = "ssh"
     host        = self.public_ip
     user        = "ubuntu"
     private_key = file("/root/.ssh/web")
   }
  key_name  = aws_key_pair.web.id
  vpc_security_group_ids = [ aws_security_group.ssh-access.id
} ]
resource "aws_key_pair" "web" {
     << code hidden >>
}
```




```
echo ${aws_instance.webserver.public_ip} >> /tmp/ip.txt"
```

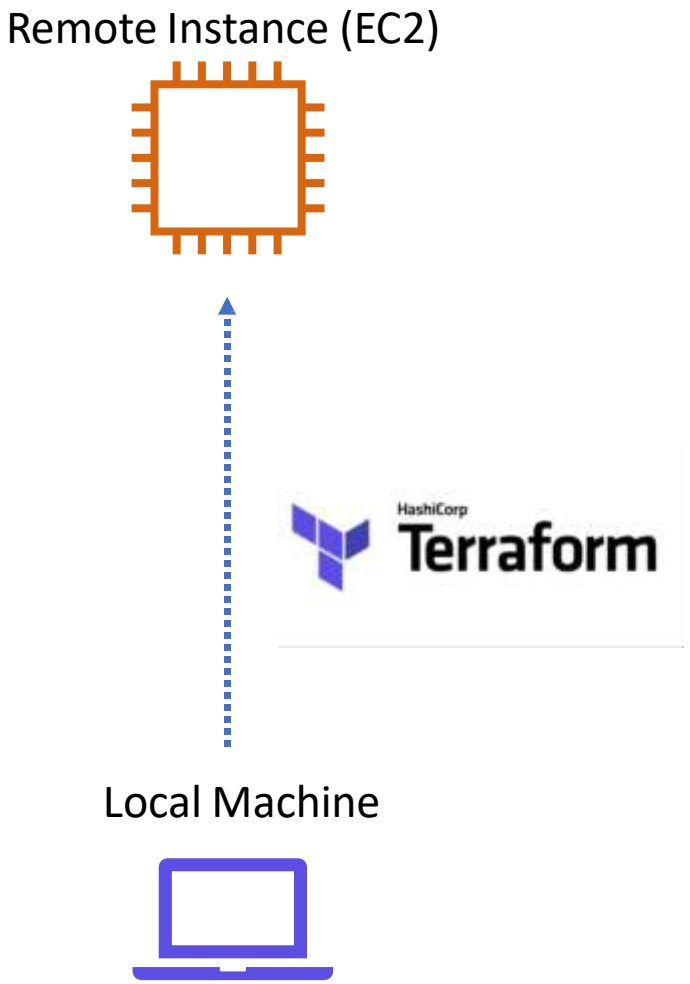
Local Exec

main.tf

```
resource "aws_instance"  "webserver"  {
  ami           = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"

  provisioner "local-exec" {
   command = "echo ${aws_instance.webserver2.public_ip} >> /tmp/ips.txt"
  }



}
```

```
>_

$ cat /tmp/ips.txt
54.214.68.27
```

- `command` - (Required) This is the command to execute. It can be provided as a relative path to the current working directory or as an absolute path. It is evaluated in a shell, and can use environment variables or Terraform variables.




```
echo ${aws_instance.webserver.public_ip} >> /tmp/ip.txt"
```

Provisioner Behavior

Lokeshkumar

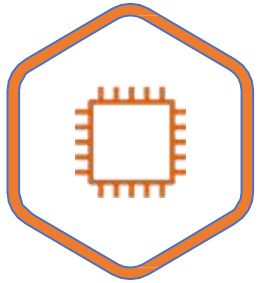
Creation Time Provisioner

main.tf

```
resource "aws_instance" "webserver" {
  ami                    = "ami-0edab43b6fa892279"
  instance_type          = "t2.micro"
  provisioner "local-exec" {
     command = "echo Instance ${aws_instance.webserver.public_ip}         > /tmp/instance_state.txt"
     Created!
} }
```

>_

```
$ cat /tmp/instance_state.txt
  Instance 3.96.136.157 Created!
```



Destroy Time Provisioner

main.tf

```
resource "aws_instance" "webserver" {
  ami                    = "ami-0edab43b6fa892279"
  instance_type          = "t2.micro"
  provisioner "local-exec" {
     command = "echo Instance ${aws_instance.webserver.public_ip} Created! > /tmp/instance_state.txt"
  }

  provisioner "local-exec" {
     when    = destroy
     command = "echo Instance ${aws_instance.webserver.public_ip} Destroyed! >
     /tmp/instance_state.txt"
   }

}
```

>_

```
$ cat /tmp/instance_state.txt
  Instance 3.96.136.157 Deleted!
```

Failure Behavior

main.tf

```
resource "aws_instance" "webserver" {
  ami                    = "ami-0edab43b6fa892279"
  instance_type          = "t2.micro"
  provisioner "local-exec" {
     on_failure = fail
     command = "echo Instance ${aws_instance.webserver.public_ip} Created!   /temp/instance_state.txt"
     >
  }
  provisioner "local-exec" {
     when    = destroy
     command = "echo Instance ${aws_instance.webserver.public_ip} Destroyed! >
     /tmp/instance_state.txt"
   }
}
```

>_

```
$ terraform apply

  Error: Error running command 'echo 35.183.14.192 > /temp/pub_ip.txt': exit status 1.
  Output: The system cannot find the path specified.
```

Failure Behavior

main.tf

```
resource "aws_instance" "webserver" {
  ami                    = "ami-0edab43b6fa892279"
  instance_type          = "t2.micro"
  provisioner "local-exec" {
     on_failure = continue
     command = "echo Instance ${aws_instance.webserver.public_ip} Created!   /temp/instance_state.txt"
     >
  }
  provisioner "local-exec" {
     when    = destroy
     command = "echo Instance ${aws_instance.webserver.public_ip} Destroyed! >
     /tmp/instance_state.txt"
   }
}
```

>_

```
$ terraform apply

  aws_instance.webserver (local-exec) The system cannot find the path specified.
  aws_instance.project: Creation complete after 22s [id=i-01585c2b9dbc445db]

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

Considerations with Provisioners

Lokeshkumar

Local-Exec | Remote-Exec

main.tf

```
resource "aws_instance" "webserver"
    { ami = "ami-0edab43b6fa892279"
    instance_type = "t2.micro"
    tags = {
        Name = "webserver"
        Description = "An NGINX WebServer on Ubuntu"
    }
    provisioner "remote-exec" {
    inline = ["echo $(hostname -i) >>
    /tmp/ips.txt"]
  }
}
```

No Provisioner Information in Plan

Network Connectivity and Authentication

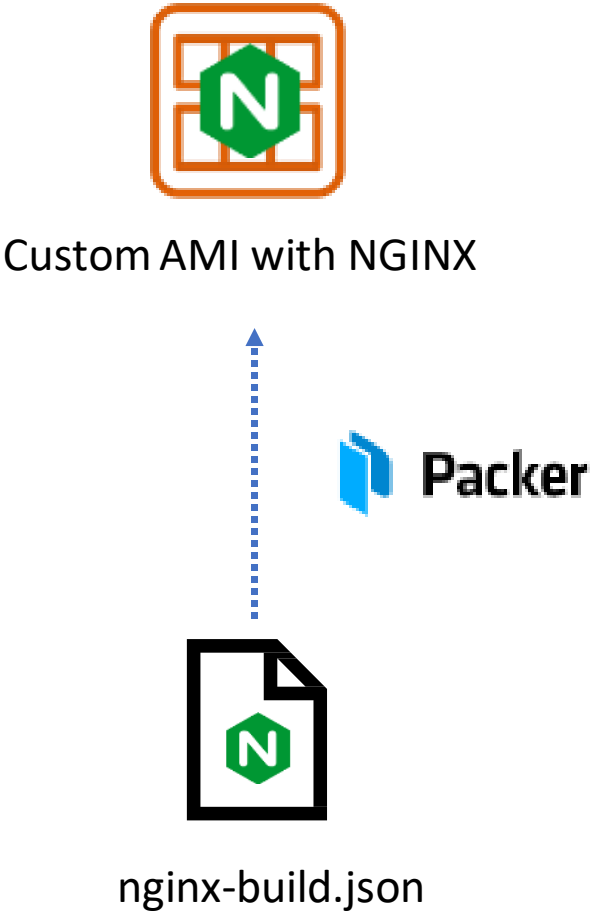
main.tf

```
resource "aws_instance" "webserver" {
    ami = "ami-0edab43b6fa892279"
    instance_type = "t2.micro"
    tags = {
        Name = "webserver"
        Description = "An NGINX WebServer on Ubuntu"
    }
    user_data = <<-EOF
                #!/bin/bash
                sudo apt update
                sudo apt install nginx -y
                systemctl enable nginx
                systemctl start nginx
                EOF
}
```

Provider	Resource	Option
AWS	aws_instance	user_data
Azure	azurrerm_virtual_machine	custom_data
GCP	google_compute_instance	meta_data
Vmware vSphere	vsphere_virtual_machine	user_data.txt

main.tf

```
resource "aws_instance" "webserver" {
    ami = "ami-XYZ"
    instance_type = "t2.micro"
    tags = {
        Name = "webserver"
        Description = "An NGINX WebServer on Ubuntu"
    }
}
```



Terraform Taint

Lokeshkumar

Taint

main.tf

```
resource "aws_instance" "webserver-3" {
  ami                    = "ami-0edab43b6fa892279"
  instance_type          = "t2.micro"
  key_name               = "ws"
  provisioner "local-exec" {
    command = "echo ${aws_instance.webserver-            > /temp/pub_ip.txt"
    3.public_ip}
  }
```

>_

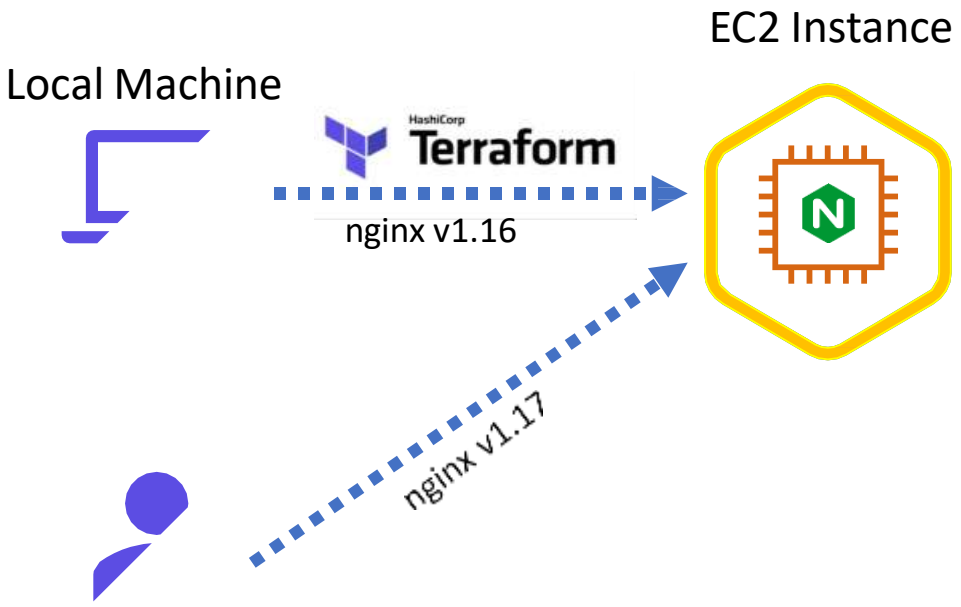
```
$ terraform apply

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


aws_instance.webserver: Creating...
aws_instance.webserver: Still creating... [10s elapsed]
aws_instance.webserver: Still creating... [20s elapsed]
aws_instance.webserver: Still creating... [30s elapsed]
aws_instance.webserver: Provisioning with 'local-exec'...
aws_instance.webserver (local-exec): Executing: ["cmd" "/C" "echo 35.183.14.192 > /temp/pub_ip.txt"]
aws_instance.webserver (local-exec): The system cannot find the path specified.


Error: Error running command 'echo 35.183.14.192 > /temp/pub_ip.txt': exit status 1. Output: The
system cannot find the path specified.
```

Taint

```
>_

$ 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.

aws_instance.webserver: Refreshing state... [id=i-0dba2d5dc22a9a904]
------------------------------------------------------------------------

-

An execution plan has been generated and is shown below.
Resource actions are indicated with the following
symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # aws_instance.webserver is tainted, so must be replaced
```

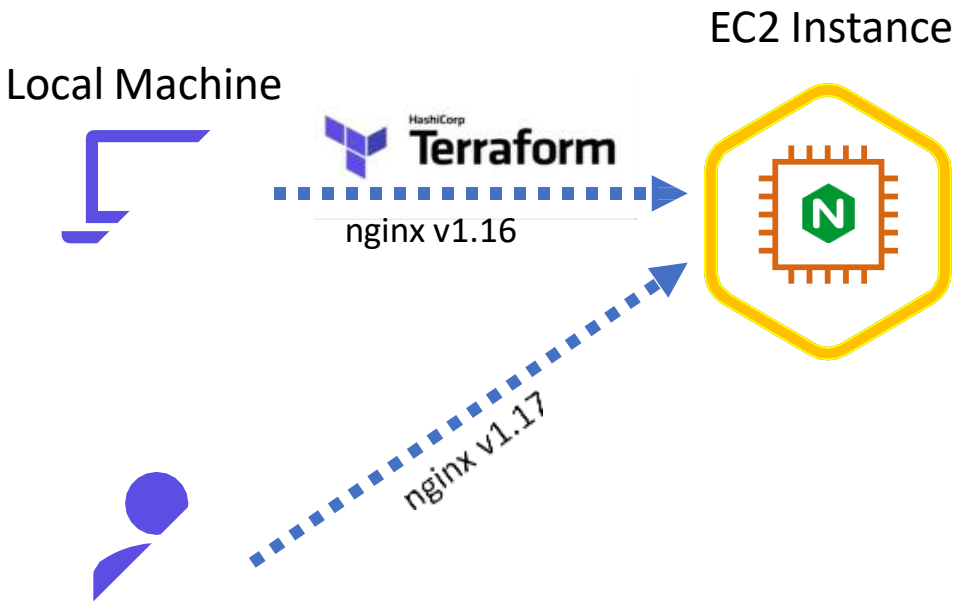
Taint

```
$ terraform taint aws_instance.webserver

Resource instance aws_instance.webserver has been marked as
tainted.
$ 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.

aws_instance.webserver: Refreshing state... [id=i-0fd3946f5b3ab8af8]

------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
-/+ destroy and then create replacement

Terraform will perform the following actions:

  # aws_instance.webserver is tainted, so must be replaced
-/+ resource "aws_instance" "webserver" {
```



Taint

```
$ terraform untaint aws_instance.webserver

Resource instance aws_instance.webserver has been
successfully untainted.


$ 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.

aws_instance.webserver: Refreshing state... [id=i-0fd3946f5b3ab8af8]

------------------------------------------------------------------------

No changes. Infrastructure is up-to-date.

This means that Terraform did not detect any differences between your
configuration and real physical resources that exist. As a result, no
actions need to be performed.
```

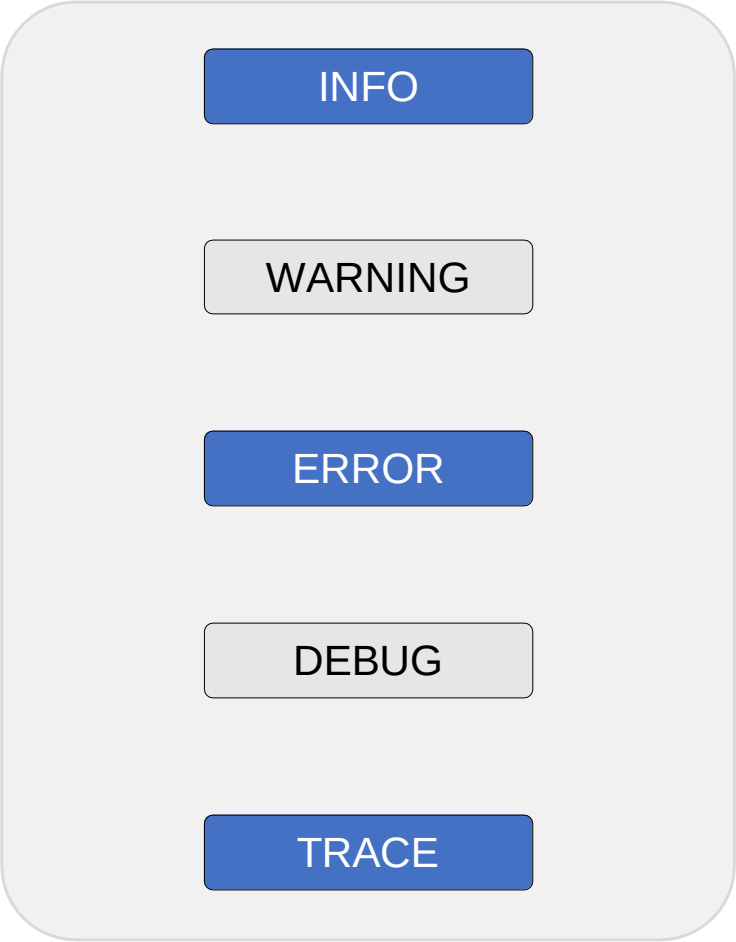


Debugging

Lokeshkumar

Log Levels

```
>_

# export TF_LOG=<log_level>

$ export TF_LOG=TRACE
```

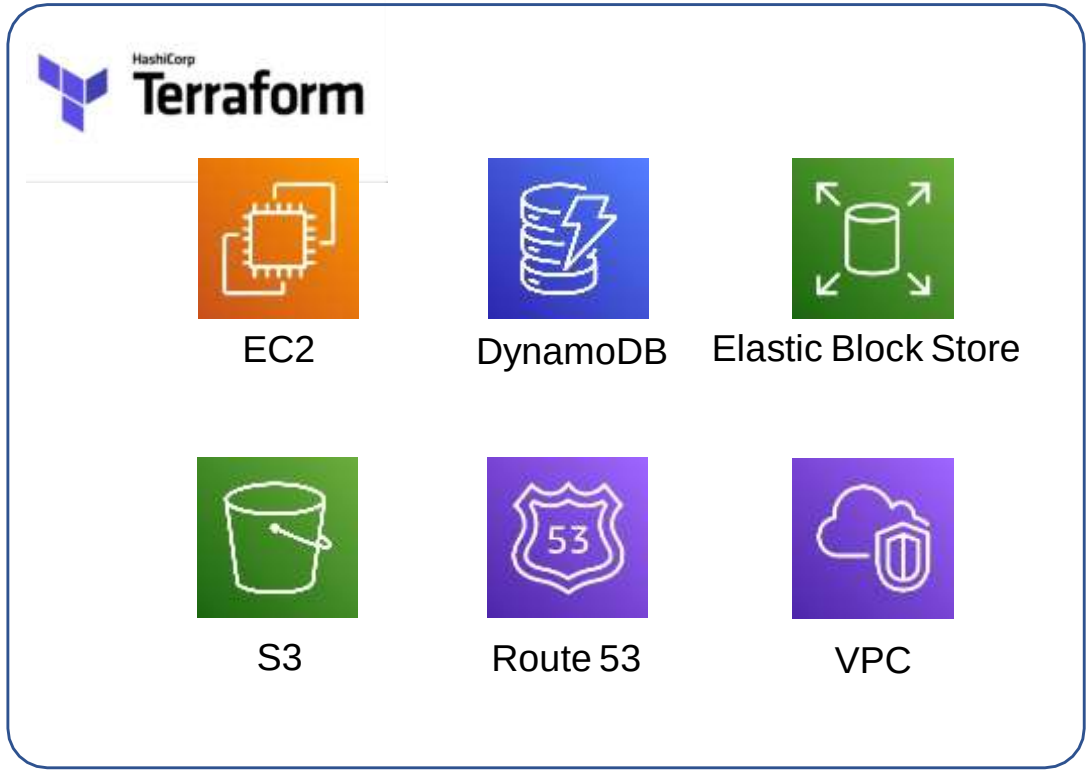


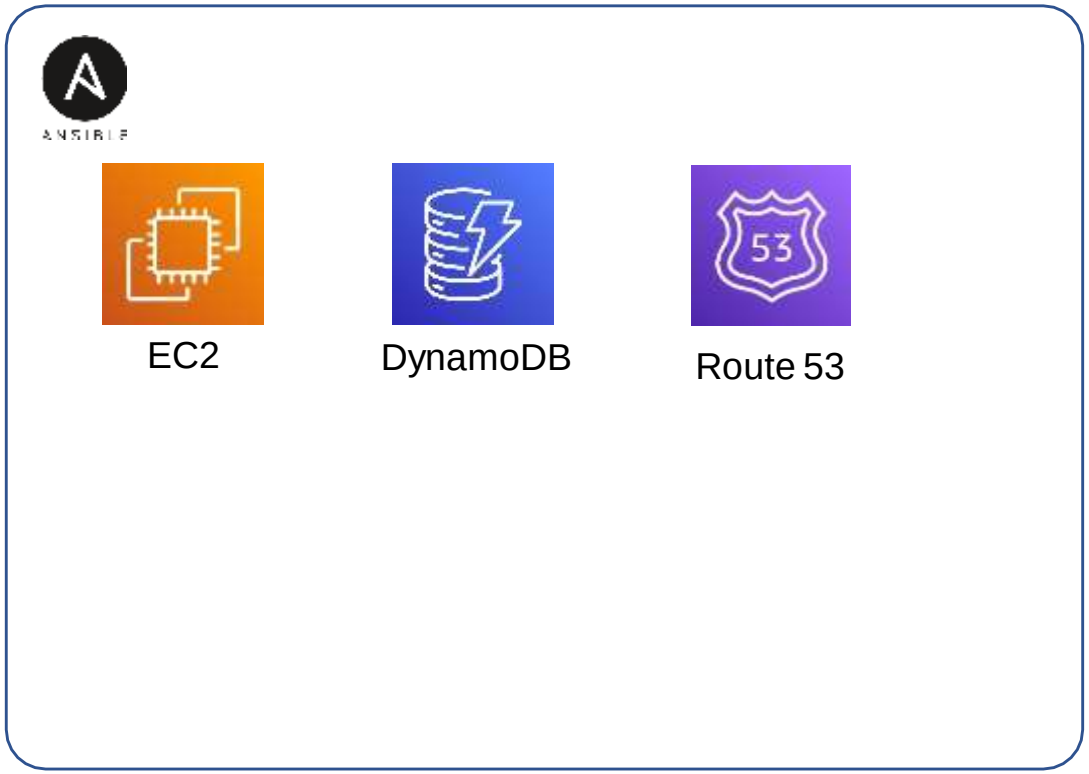
```
>_

  $ terraform plan
 ----
2020/10/18 22:08:30 [INFO] Terraform version: 0.13.0
2020/10/18 22:08:30 [INFO] Go runtime version: go1.14.2
2020/10/18 22:08:30 [INFO] CLI args: []string{"C:\\Windows\\system32\\terraform.exe", "plan"}
2020/10/18 22:08:30 [DEBUG] Attempting to open CLI config file:
C:\Users\vpala\AppData\Roaming\terraform.rc 2020/10/18 22:08:30 [DEBUG] File doesn't exist, but doesn't
need to. Ignoring.
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory terraform.d/plugins
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory
C:\Users\vpala\AppData\Roaming\terraform.d\plugins 2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory
C:\Users\vpala\AppData\Roaming\HashiCorp\Terraform\plugins
2020/10/18 22:08:30 [INFO] CLI command args: []string{"plan"}
2020/10/18 22:08:30 [WARN] Log levels other than TRACE are currently unreliable, and are supported only for backward
compatibility.
  Use -TF_LOG=TRACE to see Terraform's internal logs.

2020/10/18 22:08:30 [DEBUG] New state was assigned lineage "f413959c-538a-f9ce-524e-
1615073518d4" 2020/10/18 22:08:30 [DEBUG] checking for provisioner in "."
2020/10/18 22:08:30 [DEBUG] checking for provisioner in "C:\\Windows\\system32"
2020/10/18 22:08:30 [INFO] Failed to read plugin lock file .terraform\plugins\windows_amd64\lock.json: open
.terraform\plugins\windows_amd64\lock.json: The system cannot find the path specified.
2020/10/18 22:08:30 [INFO] backend/local: starting Plan operation
2020-10-18T22:08:30.625-0400 [INFO]  plugin: configuring client automatic mTLS
2020-10-18T22:08:30.646-0400 [DEBUG] plugin: starting plugin:
path=.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe
args=[.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe]
2020-10-18T22:08:30.935-0400 [DEBUG] plugin: plugin started:
path=.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe
pid=34016
2020-10-18T22:08:30.935-0400 [DEBUG] plugin: waiting for RPC address:
path=.terraform/plugins/registry.terraform.io/hashicorp/aws/3.11.0/windows_amd64/terraform-provider-aws_v3.11.0_x5.exe
```

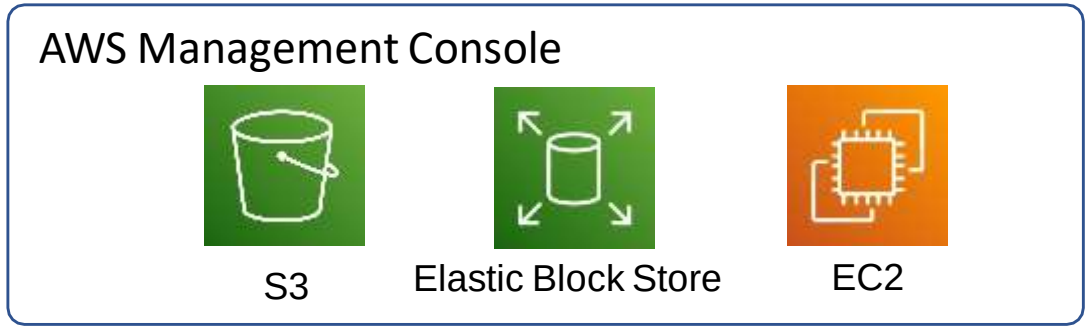
```
$ export TF_LOG_PATH=/tmp/terraform.log


$ head -10 /tmp/terraform.logs
----
2020/10/18 22:08:30 [INFO] Terraform version: 0.13.0
2020/10/18 22:08:30 [INFO] Go runtime version: go1.14.2
2020/10/18 22:08:30 [INFO] CLI args:
[]string{"C:\\Windows\\system32\\terraform.exe", "plan"}
2020/10/18 22:08:30 [DEBUG] Attempting to open CLI config file:
C:\Users\vpala\AppData\Roaming\terraform.rc
2020/10/18 22:08:30 [DEBUG] File doesn't exist, but doesn't need to.
Ignoring. 2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search
directory  terraform.d/plugins
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory
C:\Users\vpala\AppData\Roaming\terraform.d\plugins
2020/10/18 22:08:30 [DEBUG] ignoring non-existing provider search directory
C:\Users\vpala\AppData\Roaming\HashiCorp\Terraform\plugins
2020/10/18 22:08:30 [INFO] CLI command args: []string{"plan"}


$ unset TF_LOG_PATH
```

Terraform Import

Lokeshkumar







Lokeshkumar

HashiCorp Terraform

EC2

DynamoDB

Elastic Block Store

EC2

DynamoDB

Route 53

S3

Route 53

VPC

S3

Elastic Block Store

EC2

Lokeshkumar

Data Source

main.tf

```
.
.
data "aws_instance" "newserver" {
    instance_id = "i-026e13be10d5326f7"
}
output newserver {
  value       = data.aws_instance.newserver.public_ip
 }
```

>_

```
$ terraform apply

$ data.aws_instance.newserver: Refreshing state... [id=i-026e13be10d5326f7]
aws_key_pair.web: Refreshing state... [id=terraform-
20201015013048509100000001] aws_security_group.ssh-access: Refreshing state...
[id=sg-0a543f25009e14628]  aws_instance.webserver: Refreshing state... [id=i-
068fad300d9df27ac]

Apply complete! Resources: 0 added, 0 changed, 0

destroyed. Outputs:

newserver = 15.223.1.176
```

Terraform Import

```
>_

# terraform import  <resource_type>.<resource_name> <attribute>
$ terraform import aws_instance.webserver-2 i-026e13be10d5326f7

Error: resource address "aws_instance.webserver-2" does not
exist in the configuration.

Before importing this resource, please create its configuration
in the root module. For example:

resource "aws_instance" "webserver-2"
  {  # (resource arguments)
}
```

Terraform Import

main.tf

```
resource "aws_instance" "webserver-2" {
  # (resource arguments)
}
```

>_

```
$ terraform import aws_instance.webserver-2 i-026e13be10d5326f7

aws_instance.webserver-2: Importing from ID "i-
026e13be10d5326f7"... aws_instance.webserver-2: Import prepared!
  Prepared aws_instance for import
aws_instance.webserver-2: Refreshing state... [id=i-026e13be10d5326f7]

Import successful!

The resources that were imported are shown above. These resources are now
in your Terraform state and will henceforth be managed by Terraform.
```

Instance summary for i-0d7c0088069819ff8 (old-ec2) Info

Updated less than a minute ago

Connect | Actions

Instance ID i-0d7c0088069819ff8 (old-ec2)	Public IPv4 address 15.223.5.69 | open address	Private IPv4 addresses 172.31.23.147
Instance state Running	Public IPv4 DNS ec2-15-223-5-69.ca-central-1.compute.amazonaws.com | open address	Private IPv4 DNS ip-172-31-23-147.ca-central-1.compute.internal
Instance type t2.micro	Elastic IP addresses –	VPC ID vpc-7da8d215
IAM Role –	Subnet ID subnet-c6c0a8ae	

AWS Compute Optimizer

Opt-in to AWS Compute Optimizer for recommendations. Learn more

Details | Security | Networking | Storage | Monitoring | Tags

Instance details Info

Platform Ubuntu (Inferred)	AMI ID ami-0edab43b6fa892279	Monitoring disabled

Lokeshkumar

terraform.tfstate

```
{
      "mode": "managed",
      "type": "aws_instance",
      "name": "webserver-2",
      "provider":
      "provider[\"registry.terraform.io/hashicorp/aws\"]",
      "instances": [
        {
          "schema_version": 1,
          "attributes": {
            "ami": "ami-0edab43b6fa892279",
            "instance_state": "running",
            "instance_type": "t2.micro",
            "key_name": "ws",
             .
            "tags": {
              "Name": "old-ec2"
            },
             .
             .
            "vpc_security_group_ids": [
              "sg-8064fdee"
            ]
          },
             .
             .
        }
      ]
```

main.tf

```
resource "aws_instance" "webserver-2" {
   ami                    = "ami-0edab43b6fa892279"
   instance_type          = "t2.micro"
   key_name               = "ws"
   vpc_security_group_ids = ["sg-8064fdee"]

}
```

>_

```
$ 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.

aws_instance.webserver-2: Refreshing state... [id=i-0d7c0088069819ff8]

------------------------------------------------------------------------

No changes. Infrastructure is up-to-date.

This means that Terraform did not detect any differences between your
configuration and real physical resources that exist. As a result, no
actions need to be performed.
```

Terraform Modules

Lokeshkumar

main.tf

```
resource  "aws_instance" "weberver" {
  # configuration here
}
resource "aws_key_pair" "key" {
  # configuration here
}
resource "aws_security_group" "ssh-access" {
  # configuration here
}
resource  "aws_s3_bucket" "data-bucket" {
  # configuration here
}
resource  "aws_dynamodb_table" "user-data" {
  # configuration here
}
resource "aws_instance" "web-server-2" {
  # configuration here
}
```

aws_instance

aws_key_pair

aws_iam_policy

aws_s3_bucket

aws_dynamodb_table

aws_instance

main.tf

```
resource  "aws_instance" "webserver" {
  # configuration here
}
```

key_pair.tf

```
resource "aws_key_pair" "web" {
  # configuration here
}
```

dynamodb_table.tf

```
resource  "aws_dynamodb_table" "state-locking" {
  # configuration here
}
```

security_group.tf

```
resource "aws_security_group" "ssh-access"
  # configuration here
}
```

ec2_instance.tf

```
resource "aws_instance" "webserver-2" {
  # configuration here
}
```

s3_bucket.tf

```
resource  "aws_s3_bucket" "terraform-state
  # configuration here
}
```

```
>_

$ ls

provider.tf
id_rsa
id_rsa.pub
main.tf
pub_ip.txt
terraform.tfstate.backup
terraform.tfstate
iam_roles.tf
iam_users.tf
security_groups.tf
variables.tf
outputs.tf
s3_buckets.tf
dynamo_db.tf
local.tf
```

Complex Configuration Files

Duplicate Code

Increased Risk

Limits Reusability

Root Module

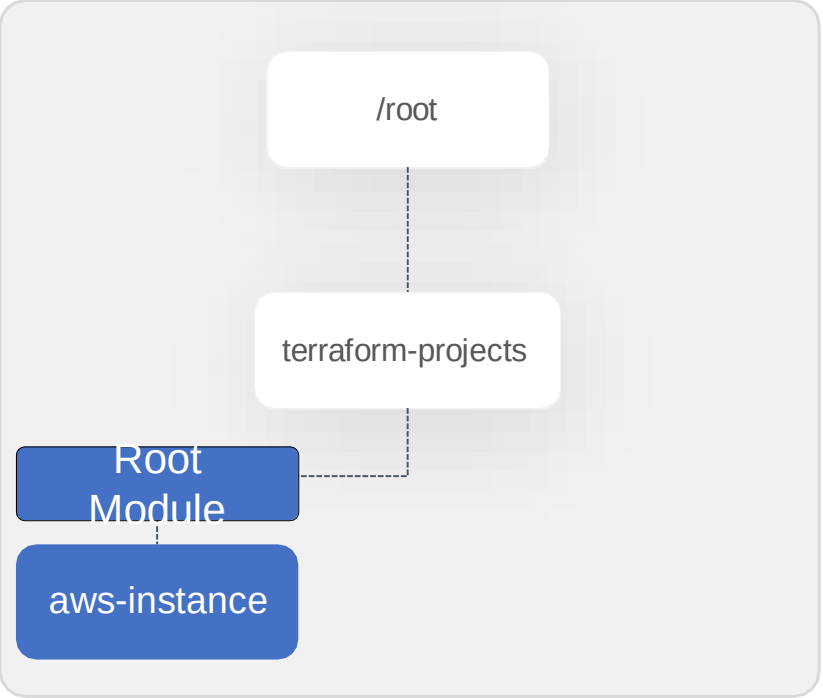
```
>_
$ ls /root/terraform-projects/aws-instance
main.tf      variables.tf
```

main.tf

```
resource "aws_instance" "webserver"
    { ami = var.ami
    instance_type = var.instance_type
    key_name = var.key
}
```

variables.tf

```
variable ami {
  type        = string
  default     = "ami-0edab43b6fa892279"
  description = "Ubuntu AMI ID in the ca-
central-1 region"
}
```

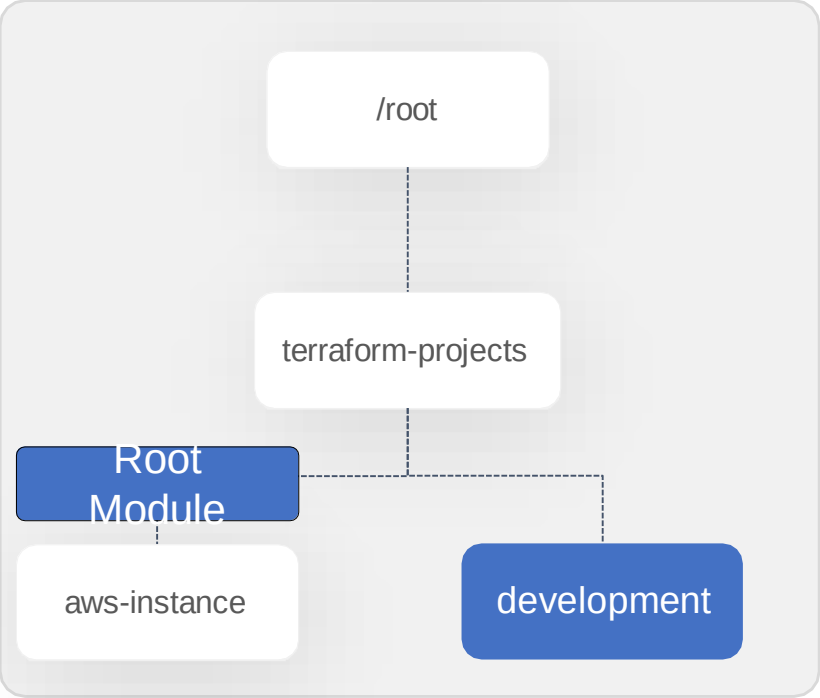


Root Module

```
>_
$ mkdir /root/terraform-
projects/development main.tf
```

main.tf

```
module "dev-webserver" {
  source = "../aws-instance"
}
```

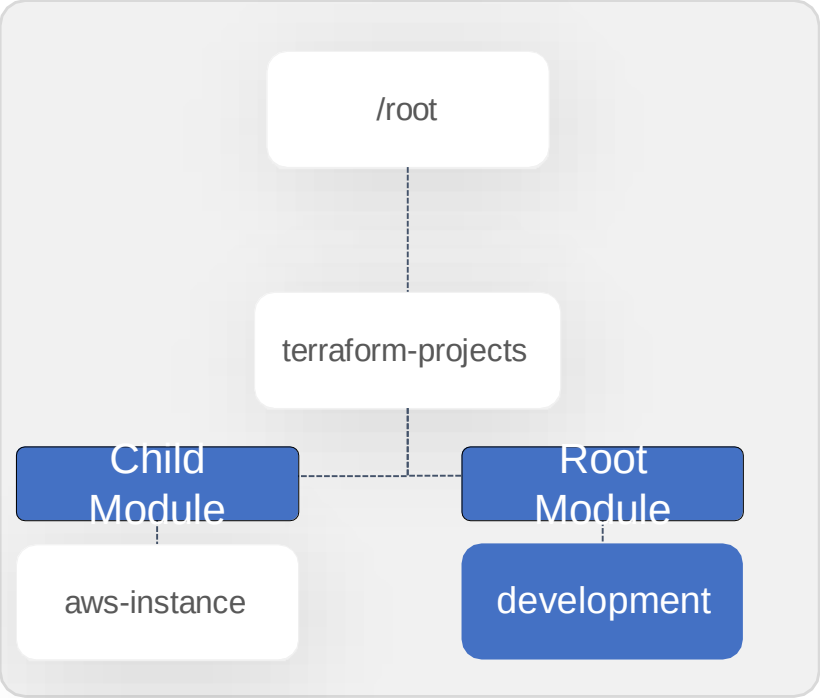


Root Module

```
>_
$ mkdir /root/terraform-
projects/development main.tf
```

main.tf

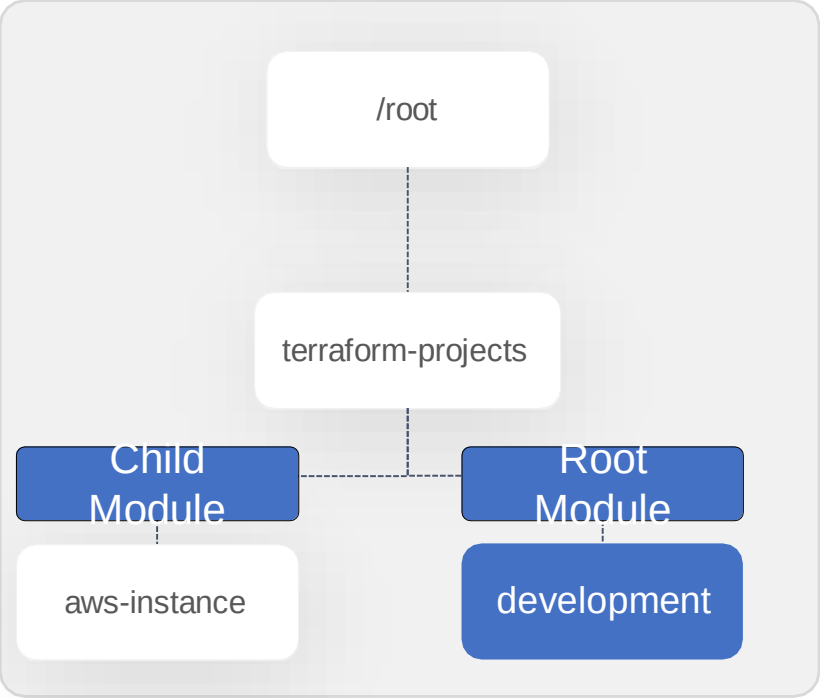
```
module "dev-webserver" {
  source = "../aws-instance"
}
```



```
>_
$ mkdir /root/terraform-
projects/development main.tf
```

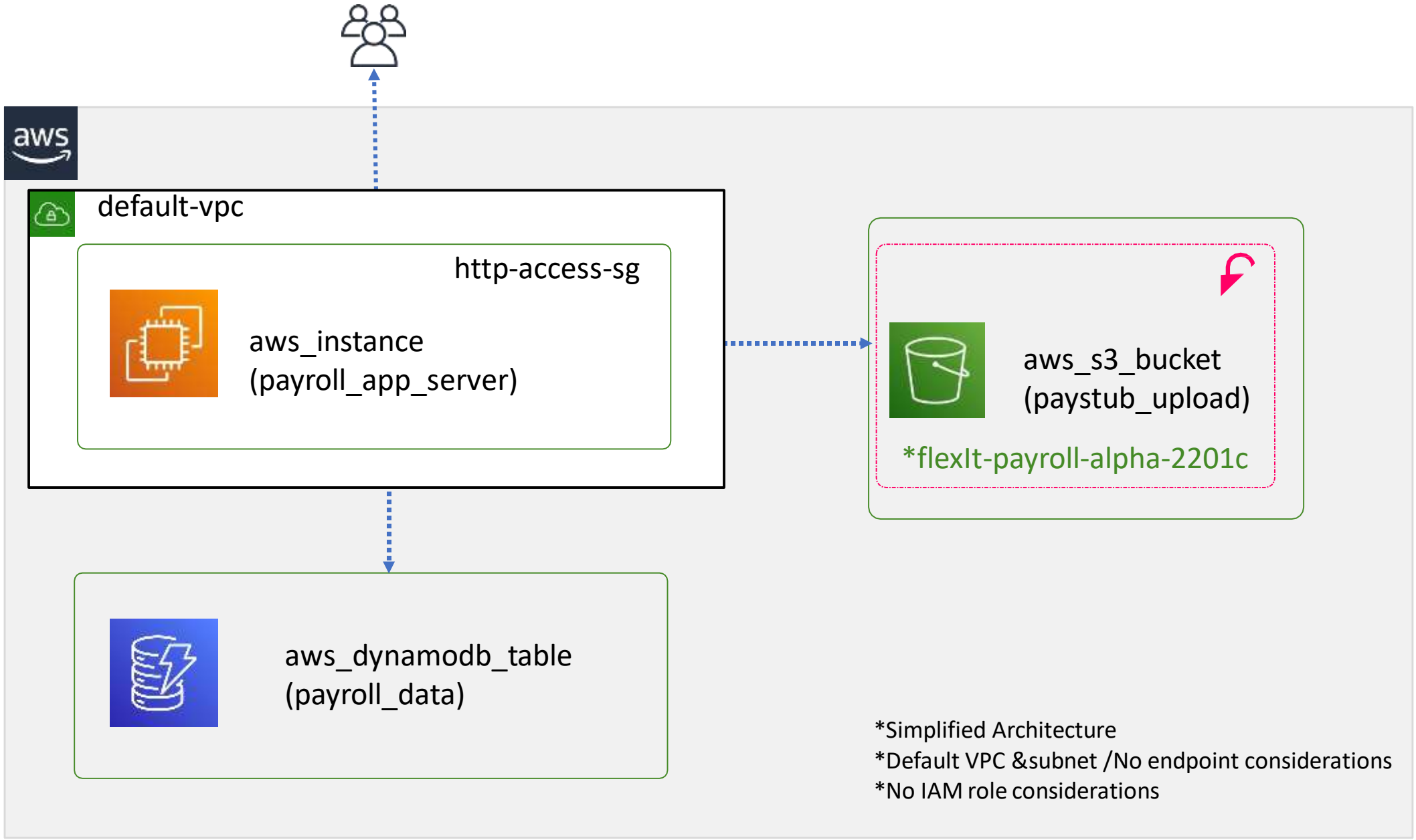
main.tf

```
module "dev-webserver" {
  source = "../aws-instance"
}
```



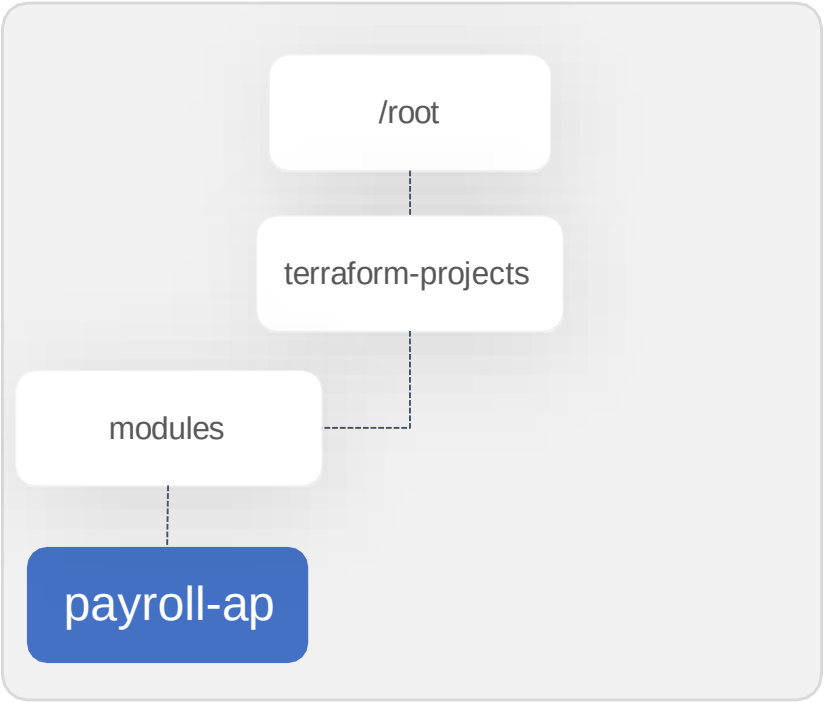
Creating and Using a Module

Lokeshkumar



Lokeshkumar

```
>_

$ mkdir /root/terraform-projects/modules/payroll-app
app_server.tf dynamodb_table.tf s3_bucket.tf
variables.tf
```



app_server.tf

```
resource "aws_instance" "app_server" {
  ami           = var.ami
  instance_type =  "t2.medium"
  tags = {
    Name = "${var.app_region}-app-server"
  }
  depends_on = [
  aws_dynamodb_table.payroll_db,
                aws_s3_bucket.payroll_data
}                ]
```

dynamodb_table.tf

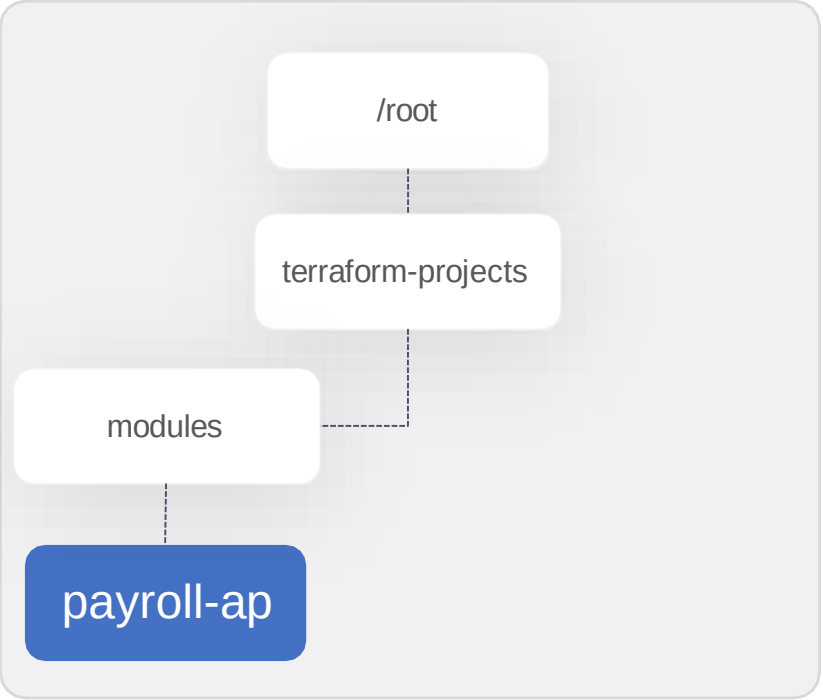
```
resource "aws_dynamodb_table" "payroll_db" {
  name           = "user_data"
  billing_mode   = "PAY_PER_REQUEST"
  hash_key       = "EmployeeID"

  attribute {
    name = "EmployeeID"
    type = "N"
  }
}
```

s3_bucket.tf

```
resource  "aws_s3_bucket"  "payroll_data" {
  bucket = "${var.app_region}-${var.bucket}"
}
```

variables.tf

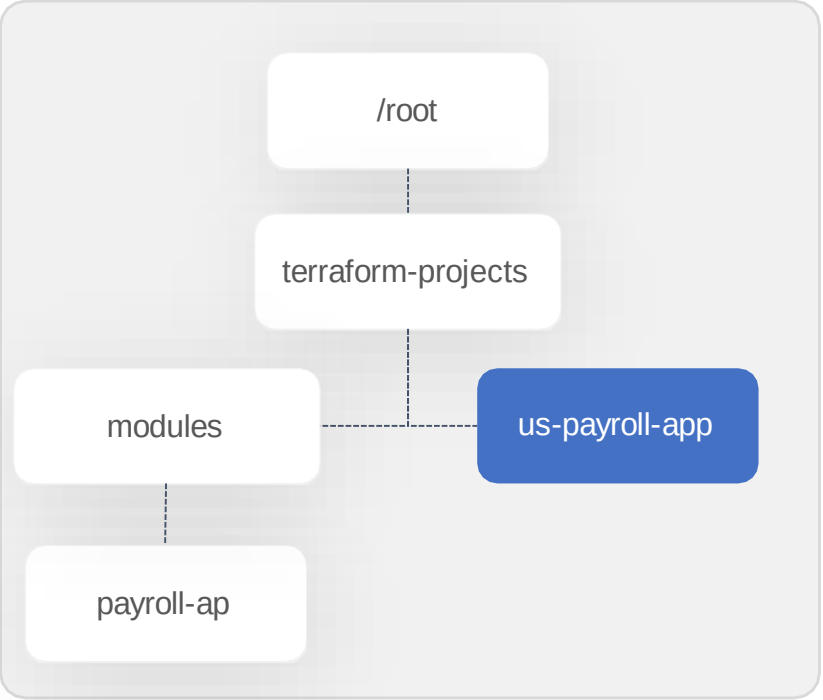
```
variable  "app_region" {
    type = string
}
variable "bucket" {
    default = "flexit-payroll-alpha-22001c"
}
variable "ami" {
    type = string
}
```



```
>_

$ mkdir /root/terraform-projects/us-payroll-
app
main.tf provider.tf
```

main.tf

```
module  "us_payroll" {
  source = "../modules/payroll-app"
  app_region = "us-east-1"
  ami        = "ami-24e140119877avm"
}
```



```
>_

$ terraform init

Initializing modules...
- us_payroll in .terraform/modules/us_payroll

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v3.11.0...
- Installed hashicorp/aws v3.11.0 (signed by HashiCorp)

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, we recommend adding version constraints in a
required_providers block
in your configuration, with the constraint strings suggested below.

* hashicorp/aws: version = "~> 3.11.0"

Terraform has been successfully initialized!
```

```
>_
$ terraform apply

.
.

Terraform will perform the following actions:

  # module.us_payroll.aws_dynamodb_table.payroll_db  will be created
  + resource "aws_dynamodb_table" "payroll_db" {
      + arn              = (known after apply)
      + billing_mode     = "PAY_PER_REQUEST"
      + hash_key         = "EmployeeID"
      + name             = "user_data"
.
.
# module.us_payroll.aws_instance.app_server will be created
  + resource "aws_instance" "app_server" {
      + ami                          = "ami-24e140119877avm"
      + instance_type                = "t2.medium‘
.
.
+ resource "aws_s3_bucket" "payroll_data" {
      + acceleration_status         = (known after apply)
      + acl                         = "private"
      + arn                         = (known after apply)
      + bucket                      = "us-east-1-flexit-payroll-alpha-22001c"
Enter a value: yes

module.us_payroll.aws_dynamodb_table.payroll_db: Creating...
module.us_payroll.aws_s3_bucket.payroll_data: Creating...
```

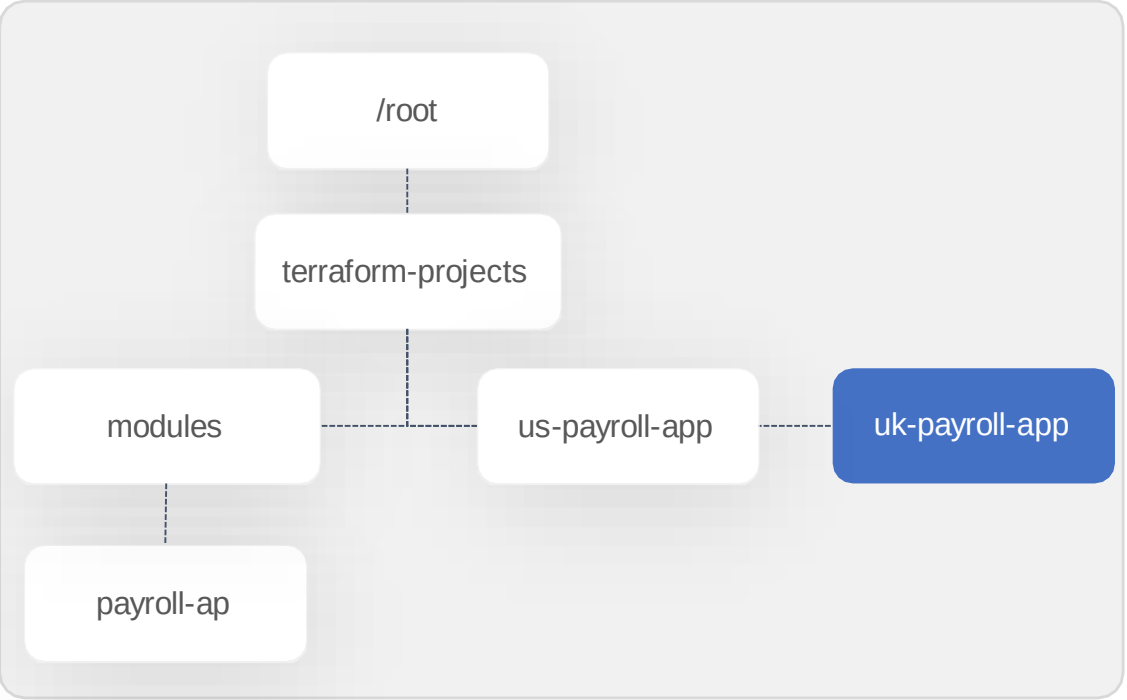
```
>_
$ mkdir /root/terraform-projects/uk-payroll-app
main.tf provider.tf
```

main.tf

```
module  "uk_payroll" {
  source = "../modules/payroll-app"
  app_region = "eu-west-2"
  ami        = "ami-35e140119877avm"
}
```

provider.tf

```
provider "aws" {
  region    = "eu-west-2"
}
```



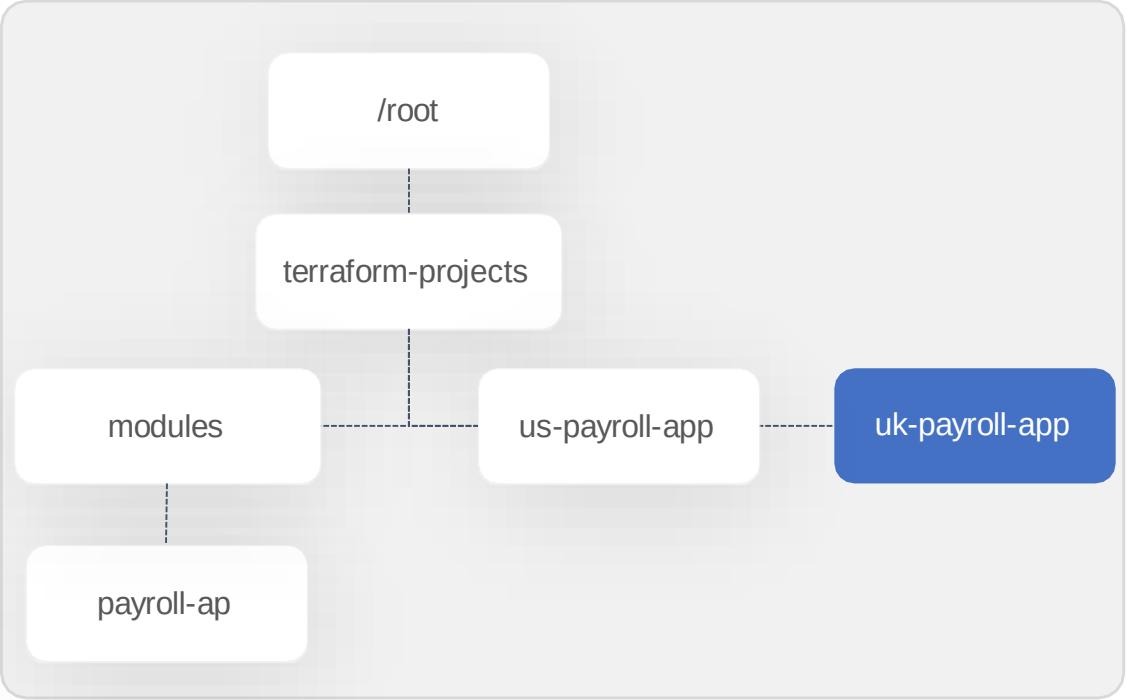
Lokeshkumar

```
>_
$ terraform apply
.
.

Terraform will perform the following actions:

  # module.us_payroll.aws_dynamodb_table.payroll_db will be created
  + resource "aws_dynamodb_table" "payroll_db" {
      + arn              = (known after apply)
      + billing_mode     = "PAY_PER_REQUEST"
      + hash_key         = "EmployeeID"
      + name             = "user_data"
.
.
# module.us_payroll.aws_instance.app_server will be created
  + resource "aws_instance" "app_server" {
      + ami                          = "ami-35e140119877avm"
      + instance_type                = "t2.medium"
.
.
+ resource "aws_s3_bucket" "payroll_data" {
      + acceleration_status         = (known after apply)
      + acl                         = "private"
      + arn                         = (known after apply)
      + bucket                      = "eu-west-2-flexit-payroll-alpha-22001c"
Enter a value: yes

module.us_payroll.aws_dynamodb_table.payroll_db: Creating...
module.us_payroll.aws_s3_bucket.payroll_data: Creating...
module.us_payroll.aws_dynamodb_table.payroll_db: Creation complete after 1s [id=user_data]
```

```
.
.

Terraform will perform the following actions:

  # module.us_payroll.aws_dynamodb_table.payroll_db will be c
  + resource "aws_dynamodb_table" "payroll_db" {
      + arn              = (known after apply)
      + billing_mode     = "PAY_PER_REQUEST"
      + hash_key         = "EmployeeID"
      + name             = "user_data"
.
.
# module.us_payroll.aws_instance.app_server will be created
  + resource "aws_instance" "app_server" {
      + ami                          = "ami-35e140119877avm"
```

main.tf

```
module  "us_payroll" {
  source = "../modules/payroll-app"
  app_region = "eu-west-2“
  ami        = "ami-35e140119877avm"
}
```



Simpler Configuration Files

Lower Risk

Re-Usability

Standardized Configuration

Lokeshkumar

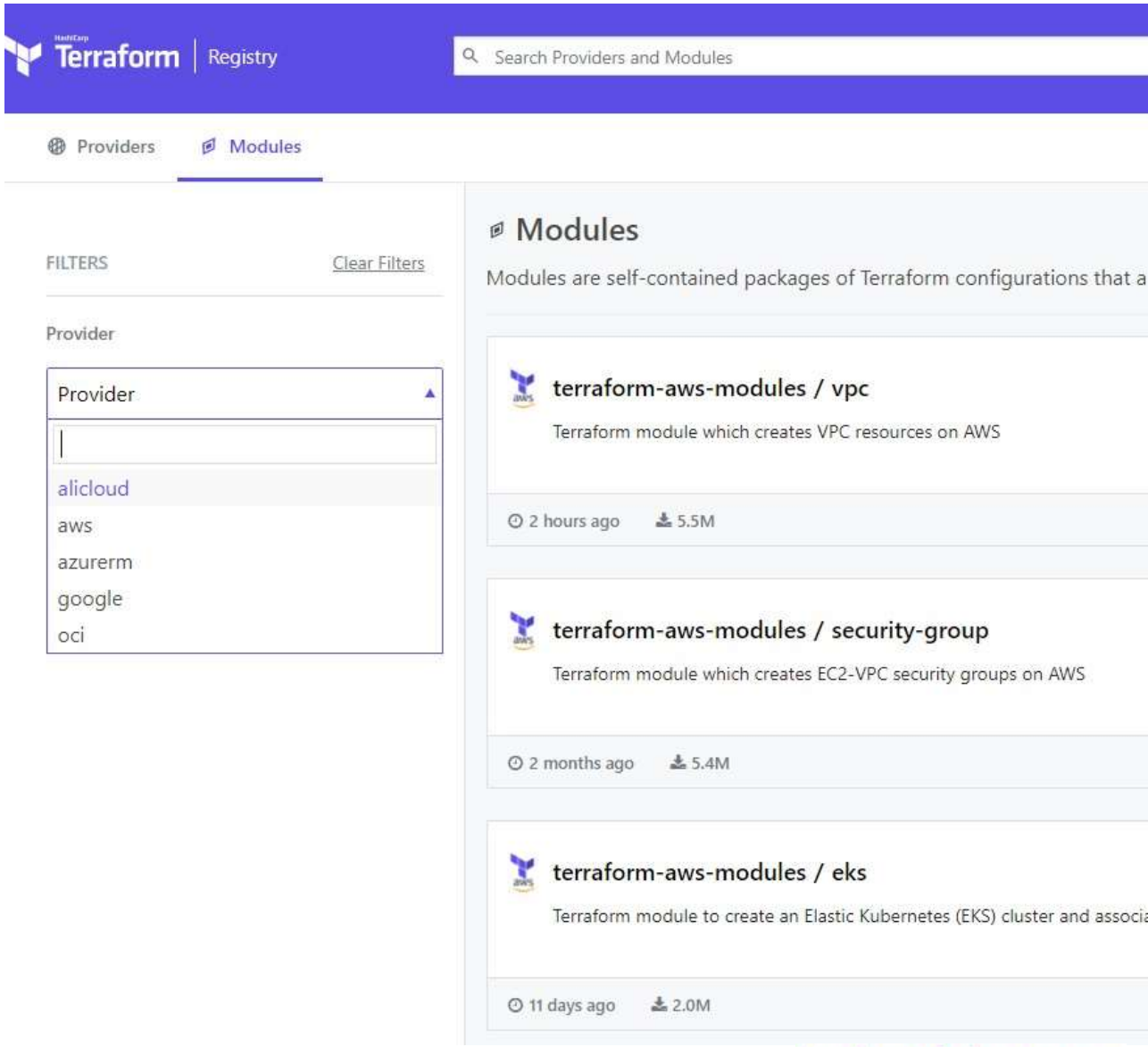
Using modules from Registery

Lokeshkumar

Local Module

main.tf

```
module "dev-webserver" {
  source = "../aws-instance/"
  key    =  "webserver"
}
```



https://registry.terraform.io/browse/modules

Terraform Registry

security-group

terraform-aws-modules/security-group
Terraform module which creates EC2-VPC security groups on AWS

dcos-terraform/security-groups
Create DC/OS related security groups

Azure/network-security-group
Terraform module to create a network security group and assign it to the specified subnet

devops-workflow/security-group
Terraform module which creates EC2-VPC security groups on AWS

claranet/nsg
Terraform module for Azure Network Security Group

security-group
AWS

Terraform module which creates EC2-VPC security groups on AWS

Published August 20, 2020 by terraform-aws-modules
Module managed by antonbabenko
Total provisions: 5.4M
Source Code: github.com/terraform-aws-modules/terraform-aws-security-group (report an iss

Submodules Examples

Terraform Module

security-group

AWS

Terraform module which creates EC2-VPC security groups on AWS

Version 3.16.0 (latest)

Published August 20, 2020 by terraform-aws-modules

Module managed by antonbabenko

Total provisions: 5.4M

Source Code: github.com/terraform-aws-modules/terraform-aws-security-group (report an issue)

Submodules

Examples

Provision Instructions

Copy and paste into your Terraform configuration, insert the variables, and run `terraform init` :

```
module "security-group" {
  source  = "terraform-aws-modules/security-group/
  version = "3.16.0"
  # insert the 2 required variables here
}
```

Terraform Module

security-group

AWS

Version 3.16.0 (latest)

Terraform module which creates EC2-VPC security groups on AWS

Published August 20, 2020 by terraform-aws-modules

Module managed by antonbabenko

Total provisions: 5.4M

Source Code: github.com/terraform-aws-modules/terraform-aws-security-group (report an issue)

Submodules

Examples

- activemq
- alertmanager
- carbon-relay-ng
- cassandra
- consul
- docker-swarm
- elasticsearch
- grafana
- graphite-statsd
- http-80
- http-8080
- https-443

Provision Instructions

Copy and paste into your Terraform configuration, insert the variables, and run `terraform init` :

```
module "security-group" {
  source  = "terraform-aws-modules/security-group/
  version = "3.16.0"
  # insert the 2 required variables here
}
```

Lokeshkumar

main.tf

```
module "security-group_ssh" {
  source  = "terraform-aws-modules/security-group/aws/modules/ssh"
  version = "3.16.0"
  # insert the 2 required variables here
  vpc_id = "vpc-7d8d215"
  ingress_cidr_blocks = [
  "10.10.0.0/16"] name = "ssh-access"

}
```

Provision Instructions

Copy and paste into your Terraform configuration, insert the variables, and run `terraform init` :

```
module "security-group" {
  source  = "terraform-aws-modules/security-group/
  version = "3.16.0"
  # insert the 2 required variables here
}
```

>_

```
$ terraform get

Downloading terraform-aws-modules/security-group/aws 3.16.0 for security-group_ssh...
- security-group_ssh in .terraform\modules\security-group_ssh\modules\ssh
```

https://registry.terraform.io/modules/terraform-aws-modules/security-group/aws/latest/submodules/ssh

Terraform Functions

Functions

main.tf

```
resource "aws_iam_policy" "adminUser"
  { name = "AdminUsers"
  policy = file("admin-policy.json")
}

resource "local_file" "pet" {
   filename = var.filename
   count = length(var.filename)
}
```

main.tf

```
resource "local_file" "pet" {
   filename = var.filename

   for_each = toset(var.region)

}

variable region {
  type        = list
  default     = ["us-east-1",
                 "us-east-1",
                 "ca-central-1"]
  description = "A list of AWS Regions"
}
```

>_

```
$ terraform console
>file("/root/terraform-projects/main.tf)
 resource "aws_instance" "development" {
   ami           = "ami-0edab43b6fa892279"
   instance_type = "t2.micro"
 }
>length(var.region)
3
> toset(var.region)
[
  "ca-central-1",
  "us-east-1",

]
>
```

Lokeshkumar

Functions

Numeric Functions

String Functions

Collection Functions

Type Conversion Functions

Numeric Functions

variables.tf

```
variable "num" {
    type = set(number)
    default = [ 250, 10, 11, 5]
    description = "A set of numbers"
}
```

>_

console

String Functions

variables.tf

```
variable "ami" {
  type = string
  default = "ami-xyz,AMI-ABC,ami-efg"
  description = "A string containing ami ids"
}
```

>_

```
$ terraform console
> split(",", "ami-xyz,AMI-ABC,ami-efg")
[ "ami-xyz","AMI-ABC","ami-efg" ]

> split(",", var.ami)
[ "ami-xyz","AMI-ABC","ami-efg“ ]

> lower(var.ami)
ami-xyz,ami-abc,ami-efg

> upper(var.ami)
AMI-XYZ,AMI-ABC,AMI-EFG

> title(var.ami)
Ami-Xyz,AMI-ABC,Ami-Efg

> substr(var.ami, 0, 7)
ami-xyz

>substr(var.ami, 8, 7)
AMI-ABC

> substr(var.ami, 16, 7)
ami-efg
```

String Functions

variables.tf

```
variable "ami" {
  type = list
  default = ["ami-xyz", "AMI-ABC", "ami-efg"]
    description = "A list of numbers"
}
```

>_

```
$ terraform console
>join(",", ["ami-xyz", "AMI-ABC", "ami-efg"])
ami-xyz,AMI-ABC,ami-efg

> join(",", var.ami)
ami-xyz,AMI-ABC,ami-efg
```

Collection Functions

variables.tf

```
variable "ami" {
  type = list
  default = ["ami-xyz", "AMI-ABC", "ami-efg"]
    description = "A list of numbers"
}
```

>_

```
$ terraform console
>length(var.ami)
3

> index(var.ami, "AMI-ABC")
1

>element(var.ami,2)
ami-efg

>contains(var.ami, "AMI-ABC")
true

>contains(var.ami, "AMI-XYZ")
false
```

Map Functions

variables.tf

```
variable "ami" {
  type = map
  default = { "us-east-1" = "ami-xyz",
              "ca-central-1" = "ami-efg",
              "ap-south-1" = "ami-ABC"
  }
  description = "A map of AMI ID's for specific regions"
}
```

>_

```
$ terraform console
>keys(var.ami)
[
  "ap-south-1",
  "ca-central-1",
  "us-east-1",
]

>values(var.ami)
[
  "ami-ABC",
  "ami-efg",
  "ami-xyz",
]

>lookup(var.ami, "ca-central-1")
ami-efg
```

Map Functions

variables.tf

```
variable "ami" {
  type = map
  default = { "us-east-1" = "ami-xyz",
              "ca-central-1" = "ami-efg",
              "ap-south-1" = "ami-ABC"
  }
  description = "A map of AMI ID's for specific regions"
}
```

>_

```
$ terraform console
>lookup(var.ami, "us-west-2")
Error: Error in function call

  on <console-input> line 1:
  (source code not available)
    |----------------
    | var.ami is map of string with 3 elements

Call to function "lookup" failed: lookup
failed to find 'us-west-2'.

> lookup (var.ami, "us-west-2", "ami-pqr")
ami-pqr
```

Operators & Conditional Expressions

Lokeshkumar

Numeric Operators

```
>_

$ terraform console
> 1 + 2
3

> 5 - 3
2

> 2 * 2
4

> 8 / 2
4
```

Equality Operators

```
>_

$ terraform console

> 8 == 8
true

8 == 7
false

> 8 != "8"
true
```

Comparison Operators

```
>_

 $ terraform console

> 5 > 7
false

> 5 > 4
true

> 5 > 5
False

> 5 >= 5
true

> 4 < 5
true

> 3 <= 4
true
```

Logical Operators

```
>_

$ terraform console

> 8 > 7 && 8 < 10
true

> 8 > 10 && 8 < 10
false

> 8 > 9 || 8 < 10
True

> var.special
true

>! var.special
false


> ! (var.b > 30)
true
```

variables.tf

```
variable special {
  type        = bool
  default     = true
  description = "Set to true to
      use special characters"
}

variable b {
    type = number
    default = 25
}
```

Lokeshkumar

Logical Operators

```
>_

$ terraform console

> var.a > var.b
true

>var.a  < var.b
false

> var.a + var.b
75
```

variables.tf

```
variable a {
    type = number
    default = 50
}
variable b {
    type = number
    default = 25
}
```

main.tf

```
resource "random_password" "password-generator"
    { length       = var.length
}

output password {
  value  = random_password.password-generator.result
}
```

variables.tf

```
variable length {
  type        = number
  description = "The length of the password"
}
```

>_

```
$ terraform apply -var=length=5 -auto-approve
random_password.password-generator:
Creating... random_password.password-generator:
Creation  complete after 0s [id=none]

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

Outputs:

password = sjsrW]
```

```
$ if [ $length -lt 8 ]
      then
         length=8;
         echo $length;
      else
         echo $length;
      fi
# Generate Password
```

Condition

If True

If False

Lokeshkumar

condition | If True | If False

main.tf

```
resource "random_password" "password-generator"
{   length  = var.length <  8 ? 8 : var.length
}

output password {
  value  = random_password.password-generator.result
}
```

```
condition ? true_val : false_val
```

variables.tf

```
variable length {
  type        = number
  description = "The length of the password"
}
```

```
$ if [ $length -lt 8 ]
    then
        length=8;
        echo $length;
    else
        echo $length;
    fi
# Generate Password
```

Condition

If True

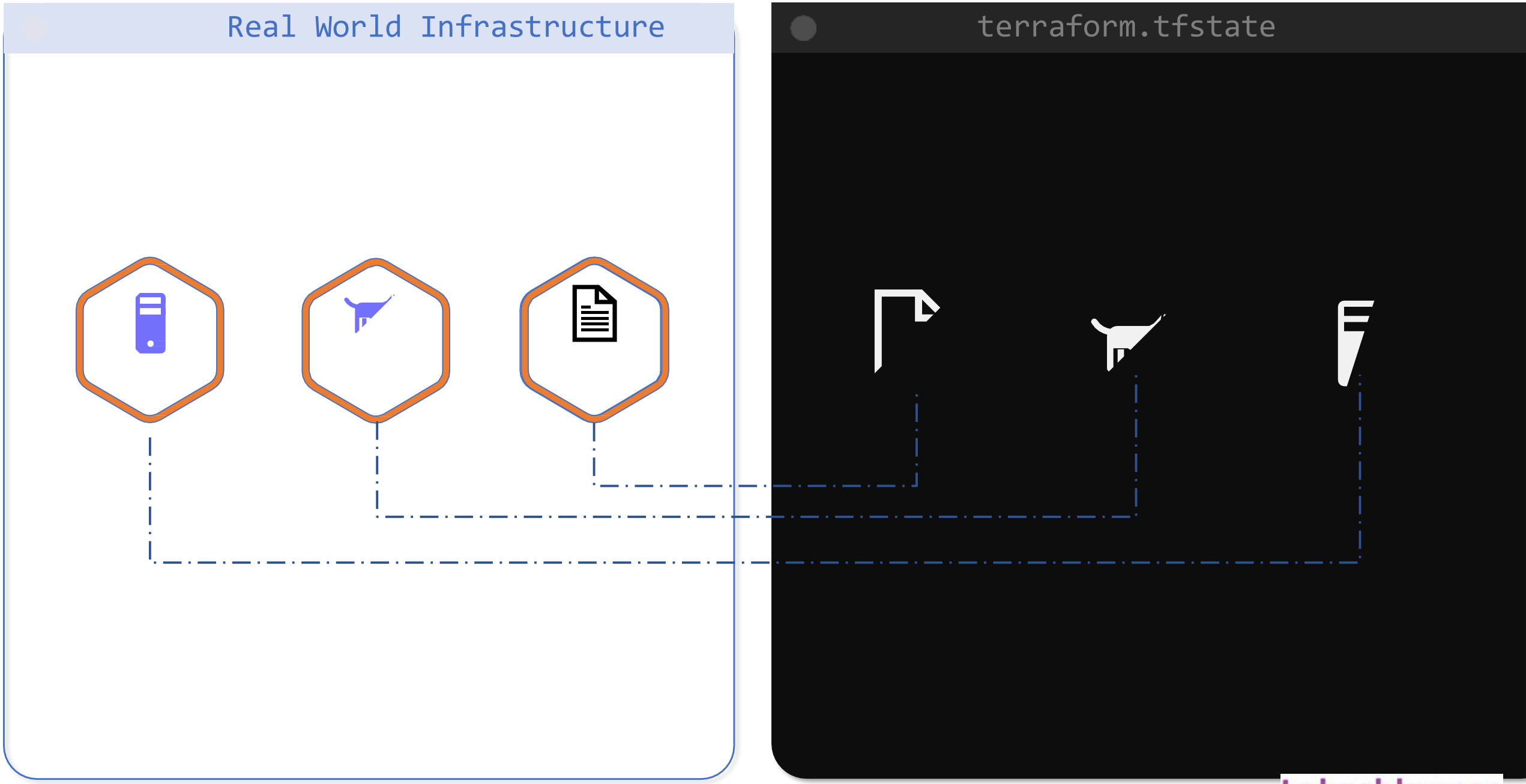
If False

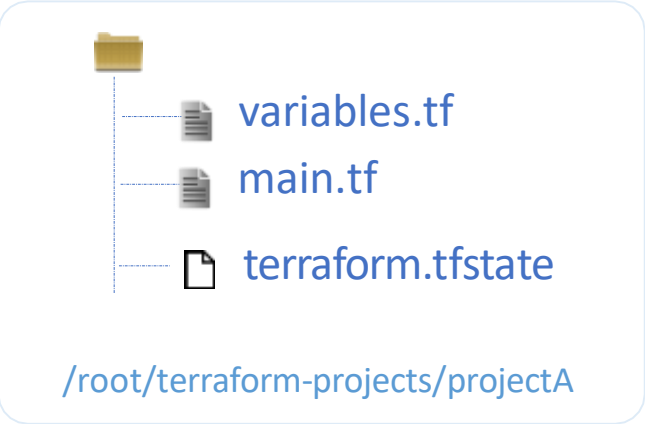
Lokeshkumar

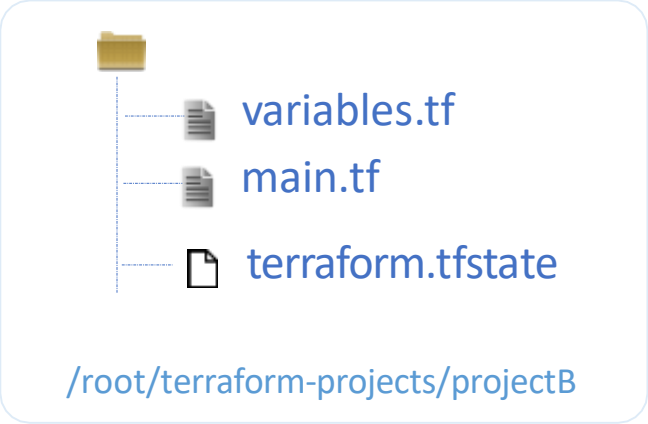
```
>_
$ terraform apply -var=length=5
Terraform will perform the following actions:

  # random_password.password-generator will be
  created
  + resource "random_password" "password-generator" {
      + id        = (known after apply)
      + length    = 8
.
Apply complete! Resources: 1 added, 0 changed, 0
destroyed.

Outputs:
password = &(1Beiaq

$ terraform apply -var=length=12
Terraform will perform the following actions:

 # random_password.password-generator must be replaced
-/+ resource "random_password" "password-generator" {
      ~ id        = "none" -> (known after apply)
      ~ length    = 8 -> 12 # forces replacement.
.
Apply complete! Resources: 1 added, 0 changed, 0
destroyed.

Outputs:

password = 8B@o}{cUzrZ7
```

Terraform Workspaces

Lokeshkumar







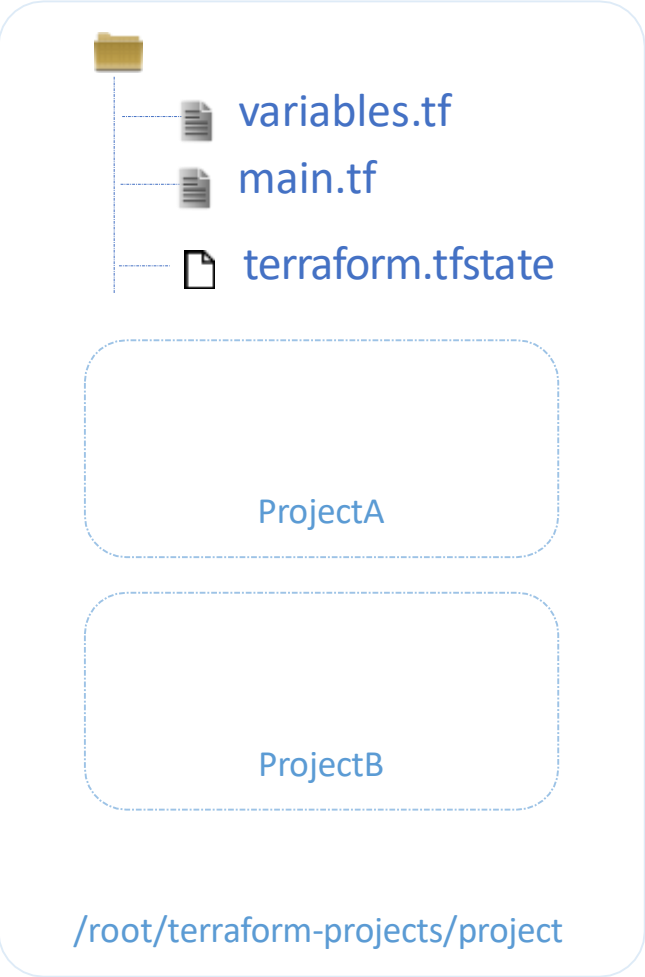
main.tf

```
resource "aws_instance" "projectA" {
  ami = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name = "ProjectA"
  }
}
```

main.tf

```
resource "aws_instance" "projectB" {
  ami = "ami-0c2f25c1f66a1ff4d"
  instance_type = "t2.micro"
  tags = {
    Name = "ProjectB"
  }
}
```

Workspace

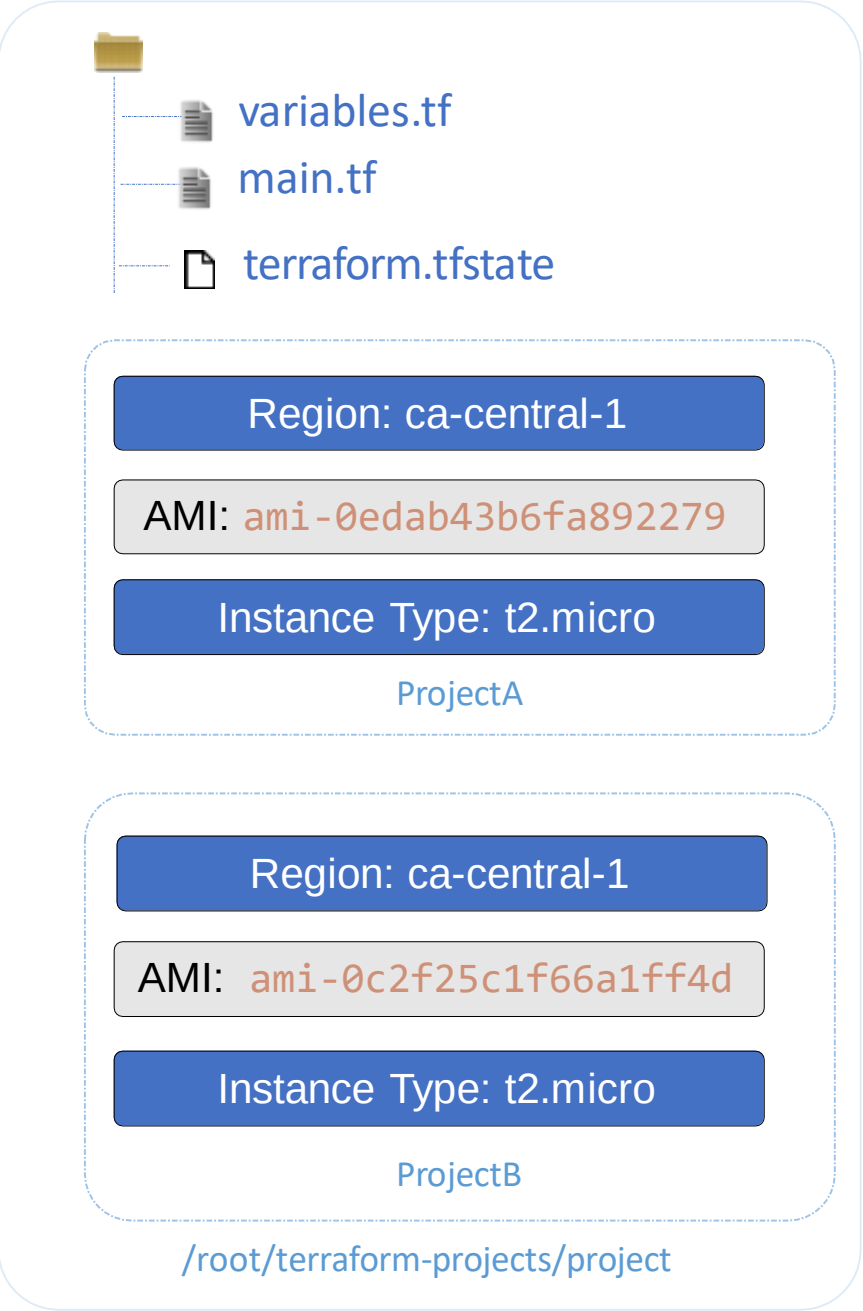


main.tf

```
resource "aws_instance" "projectA" {
  ami = "ami-0edab43b6fa892279"
  instance_type = "t2.micro"
  tags = {
    Name = "ProjectA"
  }
}
```

Workspace

```
>_

$ terraform workspace new ProjectA
Created and switched to workspace "ProjectA"!

You're now on a new, empty workspace. Workspaces isolate their state,
so if you run "terraform plan" Terraform will not see any existing
state for this configuration.


$ terraform workspace list
 default
* ProjectA
```

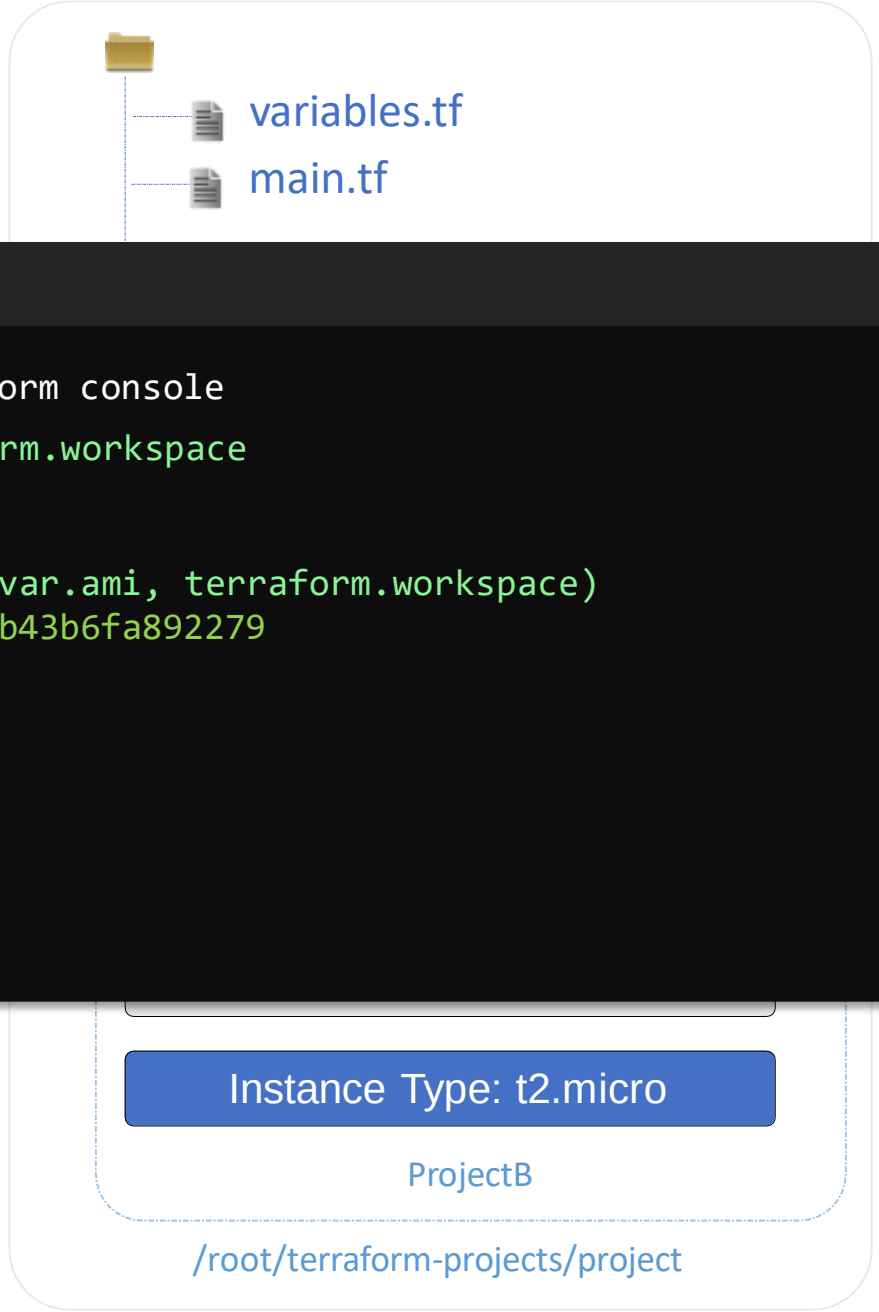


variables.tf

```
variable region {
  default = "ca-central-1"
}
variable instance_type {
  default = "t2.micro"
}
variable ami {
  type        = map
  default     = {
    "ProjectA" = "ami-0edab43b6fa892279",
    "ProjectB" = "ami-0c2f25c1f66a1ff4d"
  }
}
```

main.tf

```
resource "aws_instance" "projectA" {
  ami = "                    "
  instance_type = "        "
  tags = {
    Name = "          "
  }
}
```




```
>_

$ terraform console
>terraform.workspace
ProjectA

>lookup(var.ami, terraform.workspace)
ami-0edab43b6fa892279
```

variables.tf

```
variable region {
  default = "ca-central-1"
}
variable instance_type {
  default = "t2.micro"
}
variable ami {
  type        = map
  default     = {
    "ProjectA" = "ami-0edab43b6fa892279",
    "ProjectB" = "ami-0c2f25c1f66a1ff4d"
  }
}
```

main.tf

```
resource "aws_instance" "projectA" {
  ami = lookup(var.ami, terraform.workspace)
  instance_type =  var.instance_type
  tags = {
    Name = terraform.workspace
  }
}
```

```
>_

$ terraform plan

Terraform will perform the following
actions:

  # aws_instance.project will be created
  + resource "aws_instance" "project"={"ami-0edab43b6fa892279"
      + ami                          = "t2.micro"
      + instance_type                = "t2.micro"
      + tags                         = {
          + "Name" = "ProjectA"
        }
  .
  .

$ terraform workspace new ProjectB

Created and switched to workspace "ProjectB"!

You're now on a new, empty workspace. Workspaces isolate their state,
so if you run "terraform plan" Terraform will not see any existing
state for this configuration.
```

```
>_

$ terraform plan
Terraform will perform the following
actions:

  # aws_instance.project will be created
  + resource "aws_instance" "project"={"ami-0c2f25c1f66a1ff4d"
      + ami                          = "t2.micro"
      + instance_type                = {
      + tags                         = {
          + "Name" = "ProjectB"
        }
  .
  .
  .



$ terraform workspace select
ProjectA Switched to workspace
"ProjectA".
```

Lokeshkumar

```
>_

$ ls
main.tf  provider.tf  terraform.tfstate.d  variables.tf



$ tree terraform.tfstate.d/
terraform.tfstate.d/
|-- ProjectA
|   `-- terraform.tfstate
`-- ProjectB
    `-- terraform.tfstate

2 directories, 2 files
```