

TERRAFORM

```
terraform {  
  required_providers {  
    aws = {  
      source = "hashicorp/aws"  
      version = "5.92.0"  
    }  
  }  
}  
  
provider "aws" {  
  # Configuration options  
  region = "us-east-1"  
}  
  
resource "aws_vpc" "myvpc" {  
  cidr_block = "10.0.0.0/16"  
  
  tags = {  
    Name = "demovpc"  
  }  
}  
  
resource "aws_subnet" "pubsub" {  
  vpc_id = aws_vpc.myvpc.id  
  cidr_block = "10.0.1.0/24"  
  availability_zone = "us-east-1a"  
  
  tags = {  
    Name = "sn1"  
  }  
}  
  
resource "aws_subnet" "pubsub" {  
  vpc_id = aws_vpc.myvpc.id
```

```
cidr_block = "10.0.2.0/24"
availability_zone = "us-east-1b"
tags = {
    Name = "sn2"
}
}

resource "aws_subnet" "prisub" {
    vpc_id    = aws_vpc.myvpc.id
    cidr_block = "10.0.3.0/24"
    availability_zone = "us-east-1a"

    tags = {
        Name = "sn3"
    }
}

resource "aws_subnet" "prisub" {
    vpc_id    = aws_vpc.myvpc.id
    cidr_block = "10.0.4.0/24"
    availability_zone = "us-east-1b"

    tags = {
        Name = "sn4"
    }
}

resource "aws_internet_gateway" "tfigw" {
    vpc_id = aws_vpc.myvpc.id

    tags = {
        Name = "tfigw"
    }
}
```

```
resource "aws_route_table" "tfpubrt" {  
  vpc_id = aws_vpc.myvpc.id  
  
  route {  
    cidr_block = "0.0.0.0/0"  
    gateway_id = aws_internet_gateway.tfigw.id  
  }  
  
  tags = {  
    Name = "tfpublicroute"  
  }  
}  
  
resource "aws_route_table_association" "pubsn1" {  
  subnet_id    = aws_subnet.pubsub.id  
  route_table_id = aws_route_table.tfpubrt.id  
}  
  
resource "aws_route_table_association" "pubsn2" {  
  subnet_id    = aws_subnet.pub_sub.id  
  route_table_id = aws_route_table.tfpubrt.id  
}  
  
resource "aws_eip" "tfeip" {  
  domain = "vpc"  
}  
  
resource "aws_nat_gateway" "tfnat" {  
  allocation_id = aws_eip.tfeip.id  
  subnet_id    = aws_subnet.pub_sub.id  
  
  tags = {  
    Name = "gw NAT"  
  }  
}
```

```

resource "aws_route_table" "tfprirt" {
  vpc_id = aws_vpc.myvpc.id

  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_nat_gateway.tfnat.id
  }

  tags = {
    Name = "tfprivateroute"
  }
}

resource "aws_route_table_association" "prisn3" {
  subnet_id    = aws_subnet.prisub.id
  route_table_id = aws_route_table.tfprirt.id
}

resource "aws_route_table_association" "prisn4" {
  subnet_id    = aws_subnet.pri_sub.id
  route_table_id = aws_route_table.tfprirt.id
}

resource "aws_security_group" "allow_tfsg" {
  name      = "allow_tfsg"
  description = "Allow TLS inbound traffic"
  vpc_id    = aws_vpc.myvpc.id

  ingress {
    description    = "HTTPS "
    from_port      = 443
    to_port        = 443
    protocol       = "tcp"
    cidr_blocks    = ["0.0.0.0/0"]
  }
}

```

```

}

ingress {
  description    = "HTTP "
  from_port     = 80
  to_port       = 80
  protocol      = "tcp"
  cidr_blocks   = ["0.0.0.0/0"]
}

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

egress {
  from_port     = 0
  to_port       = 0
  protocol      = "-1"
  cidr_blocks   = ["0.0.0.0/0"]
}

tags = {
  Name = "TfsecurityGroup"
}

}

resource "aws_instance" "pub_ins" {
  ami           = "ami-0fc5d935ebf8bc3bc"
  instance_type = "t2.micro"
  subnet_id     = aws_subnet.pub_sub.id

```

```
vpc_security_group_ids    = [aws_security_group.allow_tfsg.id]
key_name                  = "David"
associate_public_ip_address = "true"
}

resource "aws_instance" "pri_ins" {
  ami            = "ami-0fc5d935ebf8bc3bc"
  instance_type  = "t2.micro"
  subnet_id      = aws_subnet.prisub.id
  vpc_security_group_ids    = [aws_security_group.allow_tfsg.id]
  key_name            = "David"
}
```

```
# terraform init
```

```
# terraform validate
```

```
# terraform plan
```

```
# terraform apply
```

```
# terraform destroy
```



the essential Terraform Cheatsheet

by justin o'connor

general commands

get the terraform version
terraform version

download and update root modules
terraform get -update=true

open up a terraform interactive terminal
terraform console

create a dot diagram of terraform dependencies
terraform graph | dot -Tpng > graph.png

format terraform code to HCL standards
terraform fmt

validate terraform code syntax
terraform validate

enable tab auto-completion in the terminal
terraform -install-autocomplete

show information about provider requirements
terraform providers

login and logout of terraform cloud
terraform login and terraform logout

workspaces

list the available workspaces
terraform workspace list

create a new workspace
terraform workspace new development

select an existing workspace
terraform workspace select default

initialize terraform

initialize terraform in the current working directory
terraform init

skip plugin installation
terraform init -get-plugins=false

force plugin installation from a directory
terraform init -plugin-dir=PATH

upgrade modules and plugins at initialization
terraform init -upgrade

update backend configuration
terraform init -migrate-state -force-copy

skip backend configuration
terraform init -backend=false

use a local backend configuration
terraform init -backend-config=FILE

change state lock timeout (default is zero seconds)
terraform init -lock-timeout=120s

plan terraform

produce a plan with diff between code and state
terraform plan

output a plan file for reference during apply
terraform plan -out current.tfplan

output a plan to show effect of terraform destroy
terraform plan -destroy

target a specific resource for deployment
terraform plan -target=ADDRESS

note that the -target option is also available for the terraform apply and terraform destroy commands.

outputs

list available outputs
terraform output

output a specific value
terraform output NAME

apply terraform

apply the current state of terraform code
terraform apply

specify a previously generated plan to apply
terraform apply current.tfplan

enable auto-approval or automation
terraform apply -auto-approve

destroy terraform

destroy resources managed by terraform state
terraform destroy

enable auto-approval or automation
terraform destroy -auto-approve

manage terraform state

list all resources in terraform state
terraform state list

show details about a specific resource
terraform state show ADDRESS

track an existing resource in state under new name
terraform state mv SOURCE DESTINATION

import a manually created resource into state
terraform state import ADDRESS ID

pull state and save to a local file
terraform state pull > terraform.tfstate

push state to a remote location
terraform state push PATH

replace a resource provider
terraform state replace-provider A B

taint a resource to force redeployment on apply
terraform taint ADDRESS

untaint a previously tainted resource
terraform untaint ADDRESS

Version 1 <https://justinoconnor.codes>