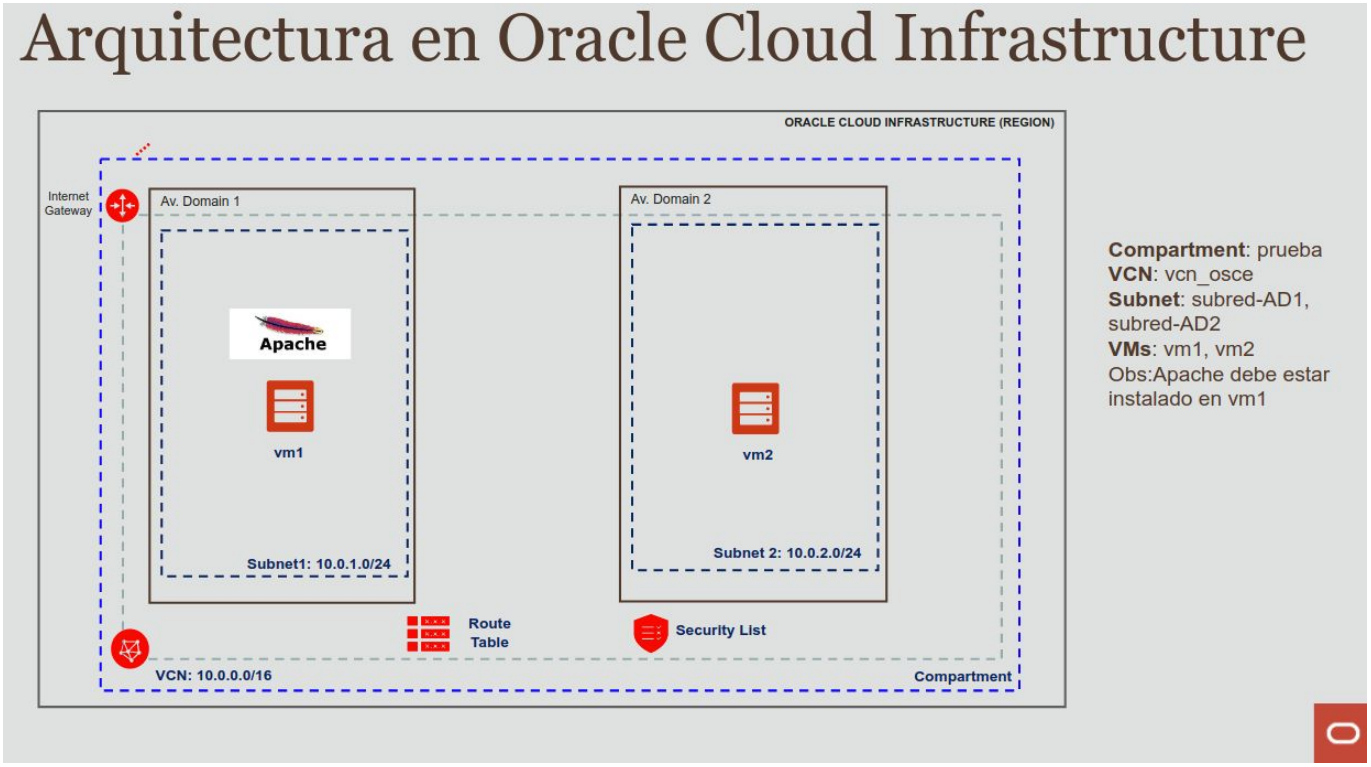
Despliegue de una arquitectura de desarrollo usando ‘Terraform’ - ‘Infrastructure as Code’ (IaC)

Introducción

‘Infrastructure as Code’ (IaC) se entiende como el concepto de despliegue de componentes de la nube (VCN, VM, storage, DB y demás servicios) a través de la ejecución de archivos (.tf), logrando así el ahorro de tiempo y disminuyendo notablemente los errores humanos al tratar de duplicar en otro arrendamiento una arquitectura previamente desplegada por consola. En la actualidad la herramienta de IaC más usada es Terraform, desarrollada por HashiCorp. Esta herramienta es compatible con las principales marcas de nube, como AWS, Azure, Google Cloud y Oracle.

Visión General

En el siguiente gráfico se puede observar la arquitectura a implementar usando la herramienta Terraform:



Todos los componentes deben ser creados en la infraestructura cloud proporcionada por Oracle.

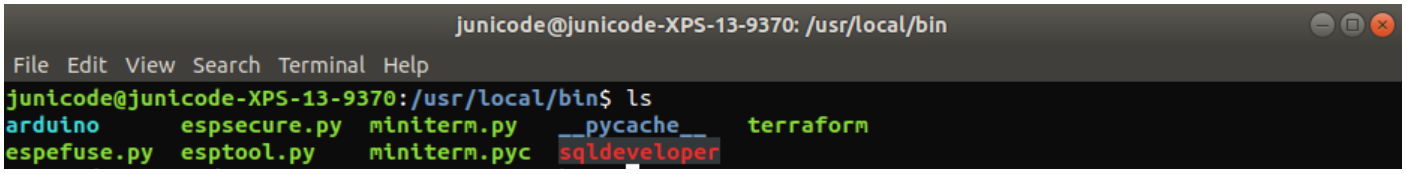
A continuación, se presentan las secciones en las que está dividido este artículo:

- **Primeros pasos con Terraform (creación de una VCN)**
- **Implementación de la Arquitectura**

Desarrollo

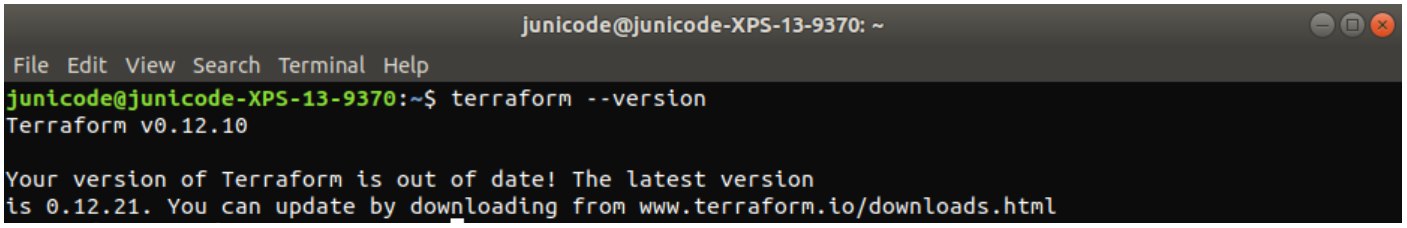
1. Primeros pasos con Terraform (creación de una VCN)

Para instalar correctamente Terraform en una máquina con sistema operativo Linux, debe ser descargado desde la página web https://www.terraform.io/downloads.html. Después, se descomprime el archivo .zip y extraer el archivo ‘terraform’. Por último, mover este archivo a la siguiente ruta: /usr/local/bin.



Se verifica que Terraform fue instalado correctamente se ejecuta el siguiente comando:

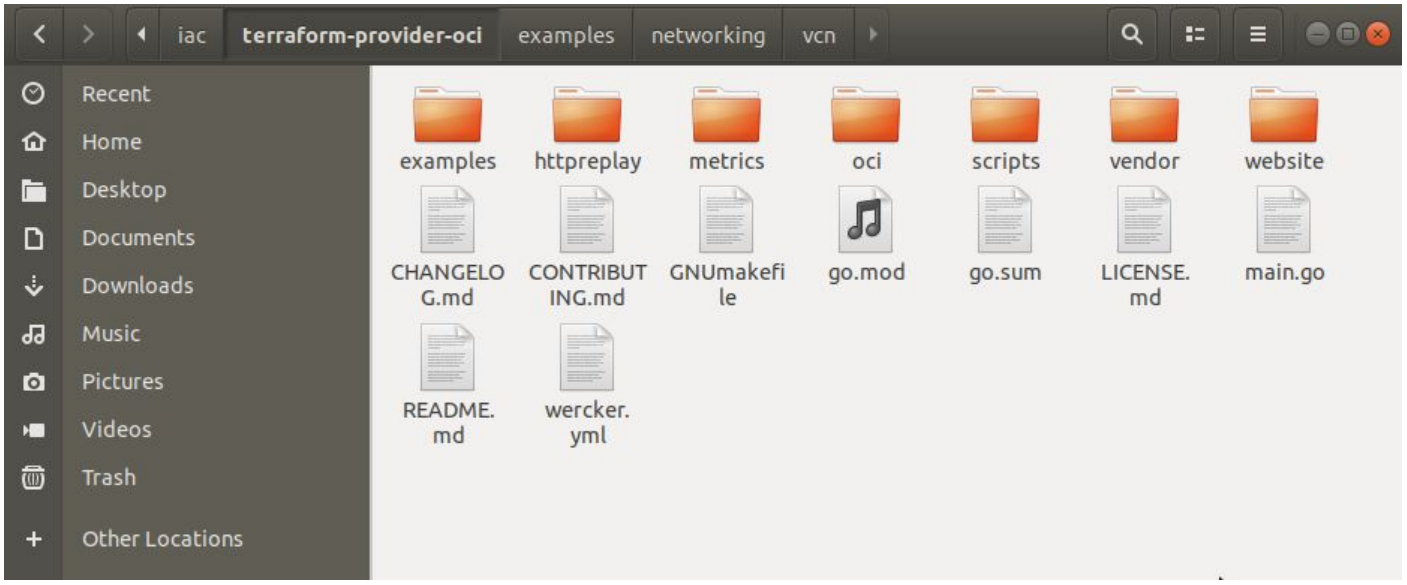
$ terraform --version



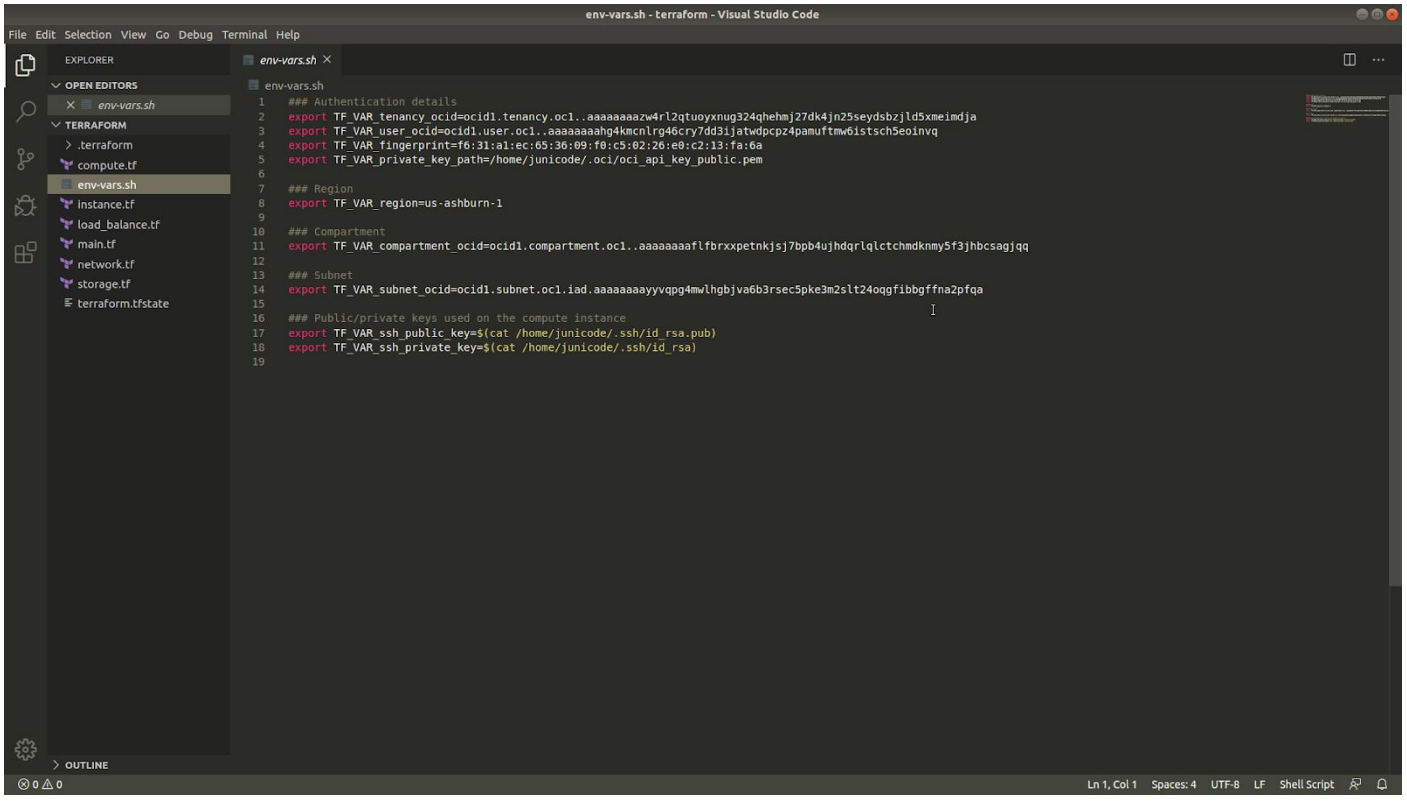
Listo. Ahora, con el fin de obtener los archivos de variables de entorno (env-vars.sh) y los que deben ser ejecutados (.tf), se descarga o clona el repositorio llamado 'terraform-providers/terraform-provider-oci':

$ git clone https://github.com/terraform-providers/terraform-provider-oci.git

Las carpetas y archivos que contiene el repositorio son mostrados en la siguiente imagen (se trabajará exclusivamente con la carpeta 'examples'):



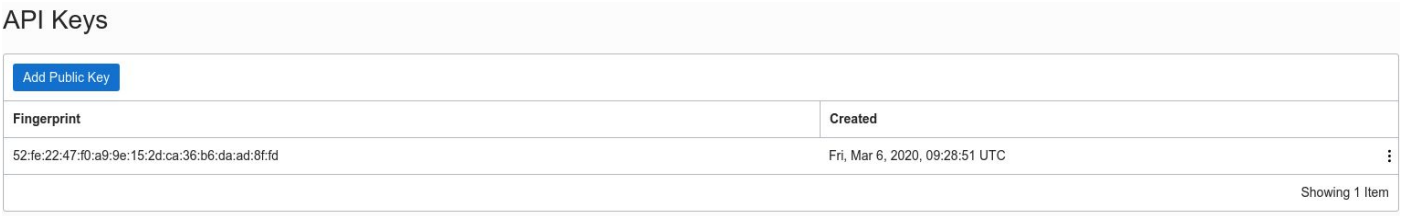
Se recomienda usar el editor de código 'Visual Studio Code', ya que soporta la sintaxis de Terraform.



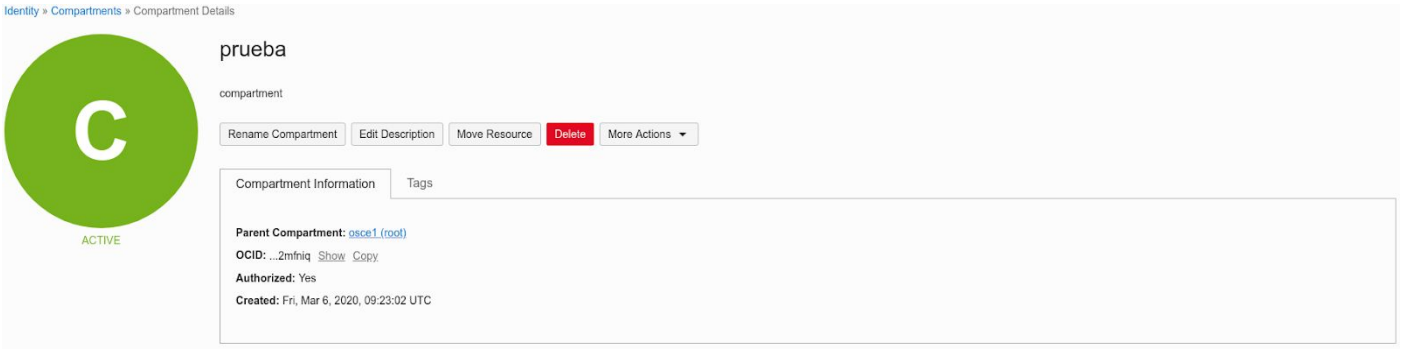
Ya teniendo Terraform instalado en nuestra computadora y la carpeta 'terraform' descargada, se procede a crear dos elementos básicos en la consola de OCI:

- Fingerprint del usuario (ncayllahua@ibitcore.com): 52:fe:22:47:f0:a9:9e:15:2d:ca:36:b6:da:ad:8f:fd

  Se crean las llaves .pem en la siguiente ruta: ~/.oci/
  $ openssl genrsa -des3 -out private.pem 2048
  $ openssl rsa -in private.pem -outform PEM -pubout -out public.pem ***(LLAVE PÚBLICA QUE DEBE SER AGREGADA)***



- Compartimiento: prueba



OCI soporta dos métodos de autenticación para Terraform: 'API Key based authentication' y 'Instance Principal based authentication'. El primero de estos dos será usado en este ejercicio.

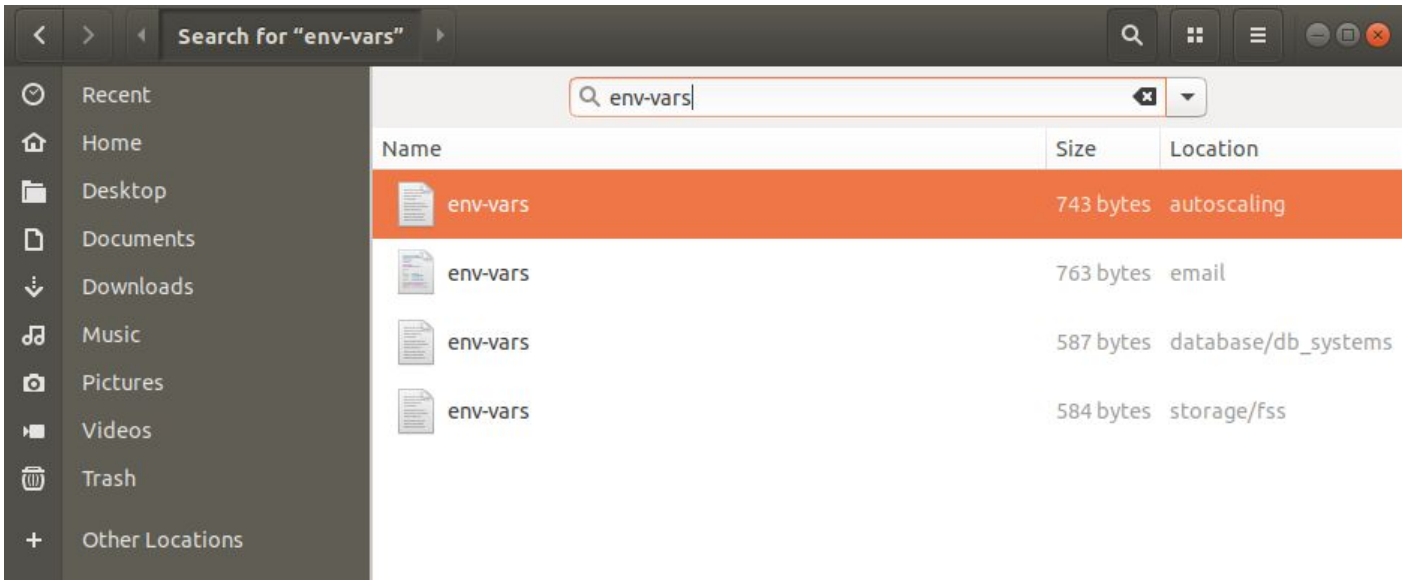
API Key based authentication

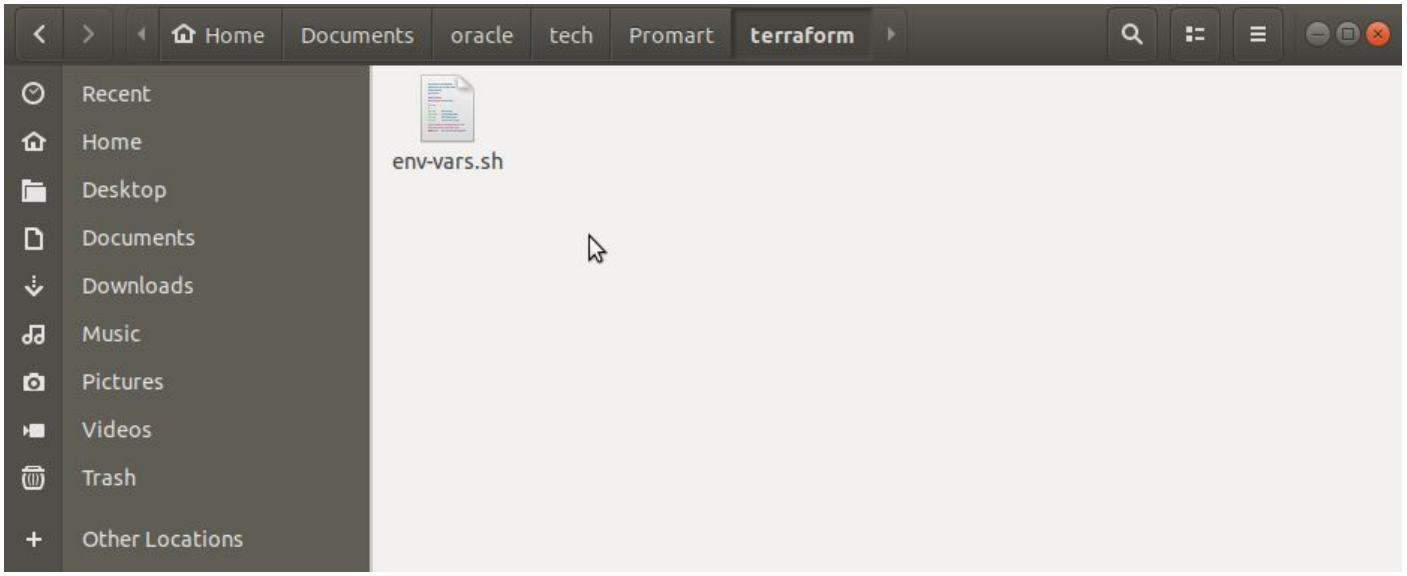
Calls to OCI using API Key authentication requires that you provide the following credentials:

- `tenancy_ocid` - OCID of your tenancy. To get the value, see Required Keys and OCIDs #Tenancy's OCID.
- `user_ocid` - OCID of the user calling the API. To get the value, see Required Keys and OCIDs #User's OCID.
- `private_key` - The contents of the private key file, required if `private_key_path` is not defined, takes precedence over `private_key_path` if both are defined. For details on how to create and configure keys see Required Keys and OCIDs #How to Upload the Public Key.
- `private_key_path` - The path (including filename) of the private key stored on your computer, required if `private_key` is not defined. For details on how to create and configure keys see Required Keys and OCIDs #How to Upload the Public Key.
- `private_key_password` - (Optional) Passphrase used for the key, if it is encrypted.
- `fingerprint` - Fingerprint for the key pair being used. To get the value, see Required Keys and OCIDs #How to Get the Key's Fingerprint.
- `region` - An Oracle Cloud Infrastructure region. See Regions and Availability Domains.

Se procede a crear una nueva carpeta, en donde se copiará y modificará ciertos archivos de la carpeta ‘examples’, proveniente del repositorio clonado en los anteriores pasos. Esta carpeta la llamaremos ‘terraform’.

Primero, se copia algún archivo de variables de entorno (env-vars.sh).





Seguidamente, se modifica este archivo con los datos respectivos de su cuenta OCI.

Authentication details

export TF_VAR_tenancy_ocid=ocid1.tenancy.oc1..aaaaaaaatbtjah3ht6up5waoi52wiyde5gdxbdvofjzok4ruo6k4tvlnl4ia
export TF_VAR_user_ocid=ocid1.user.oc1..aaaaaaaadzczsphbzlvsqymx6kbllrfybp6wpkx6hubfbxwyfjabg7ufm5zq
export TF_VAR_fingerprint=52:fe:22:47:f0:a9:9e:15:2d:ca:36:b6:da:ad:8f:fd
export TF_VAR_private_key_path=/home/junicode/.oci//private.pem
export TF_VAR_private_key_password=private

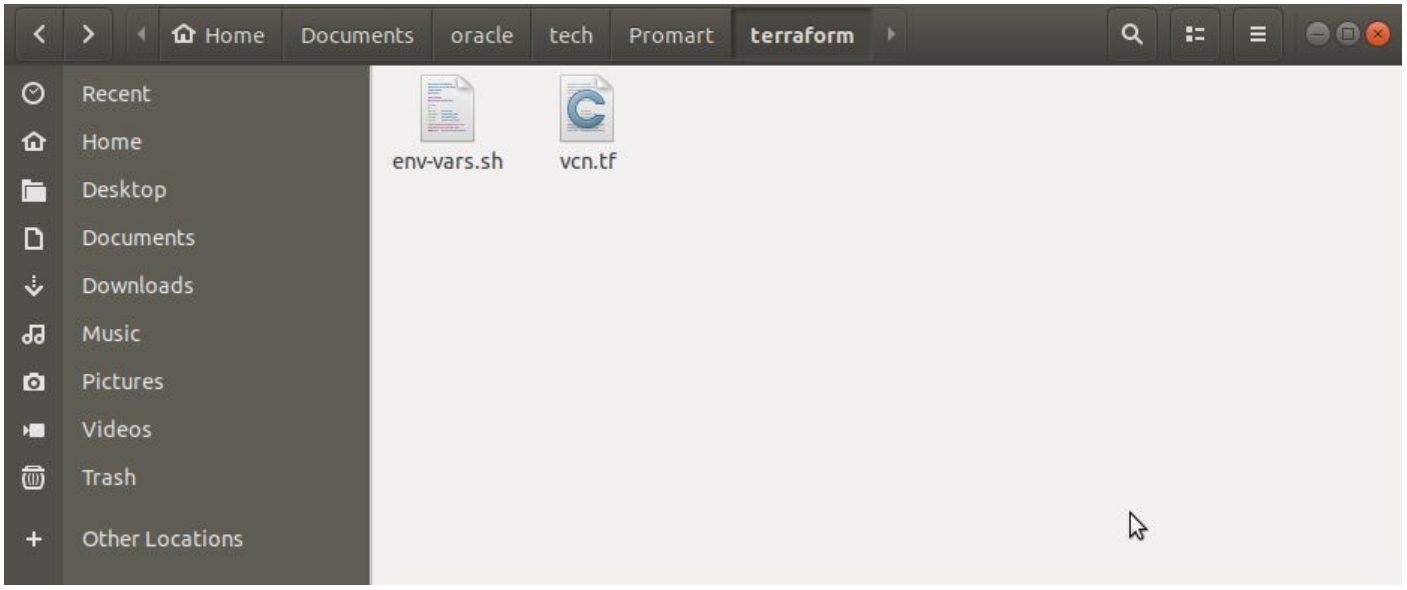
Region

export TF_VAR_region=us-ashburn-1

Compartment

export TF_VAR_compartment_ocid=ocid1.compartment.oc1..aaaaaaaalxo7zindcffskunoclqhfxnvmblgpjs4r7s7li7cvx2l3p2mfniq

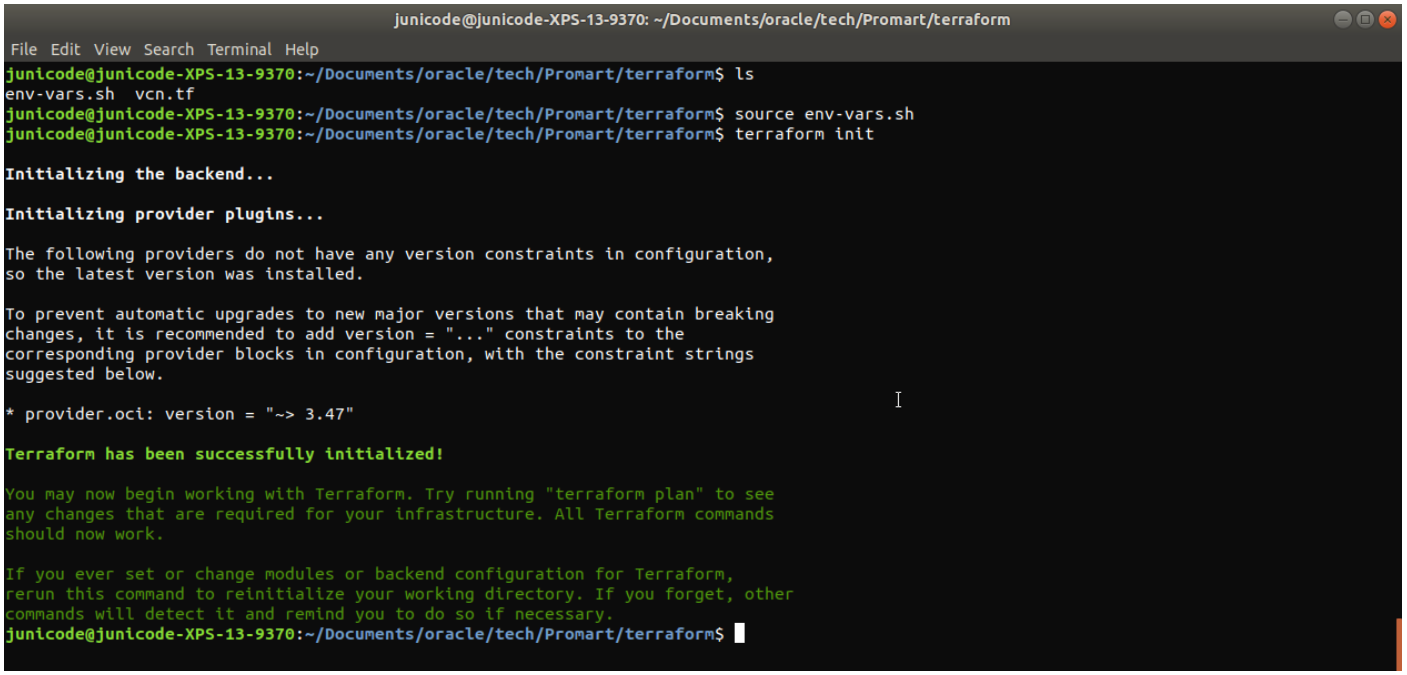
Se procede a crear la VCN, haciendo uso del archivo ‘vcn.t’. Dentro del archivo se cambia la palabra ‘vcn1’ por ‘vcn_osce’ (excepto en DNS label).




```
// Copyright (c) 2017, 2019, Oracle and/or its affiliates. All rights reserved.

variable "tenancy_ocid" {}
variable "user_ocid" {}
variable "fingerprint" {}
variable "private_key_path" {}
variable "compartment_ocid" {}
variable "region" {}

provider "oci" {
  tenancy_ocid     = "${var.tenancy_ocid}"
  user_ocid        = "${var.user_ocid}"
  fingerprint      = "${var.fingerprint}"
  private_key_path = "${var.private_key_path}"
  region           = "${var.region}"
}

resource "oci_core_vcn" "vcn_osce" {
  cidr_block     = "10.0.0.0/16"
  dns_label      = "vcn1"
  compartment_id = "${var.compartment_ocid}"
  display_name   = "vcn_osce"
}

output "vcn_id" {
  value = "${oci_core_vcn.vcn_osce.id}"
}
```

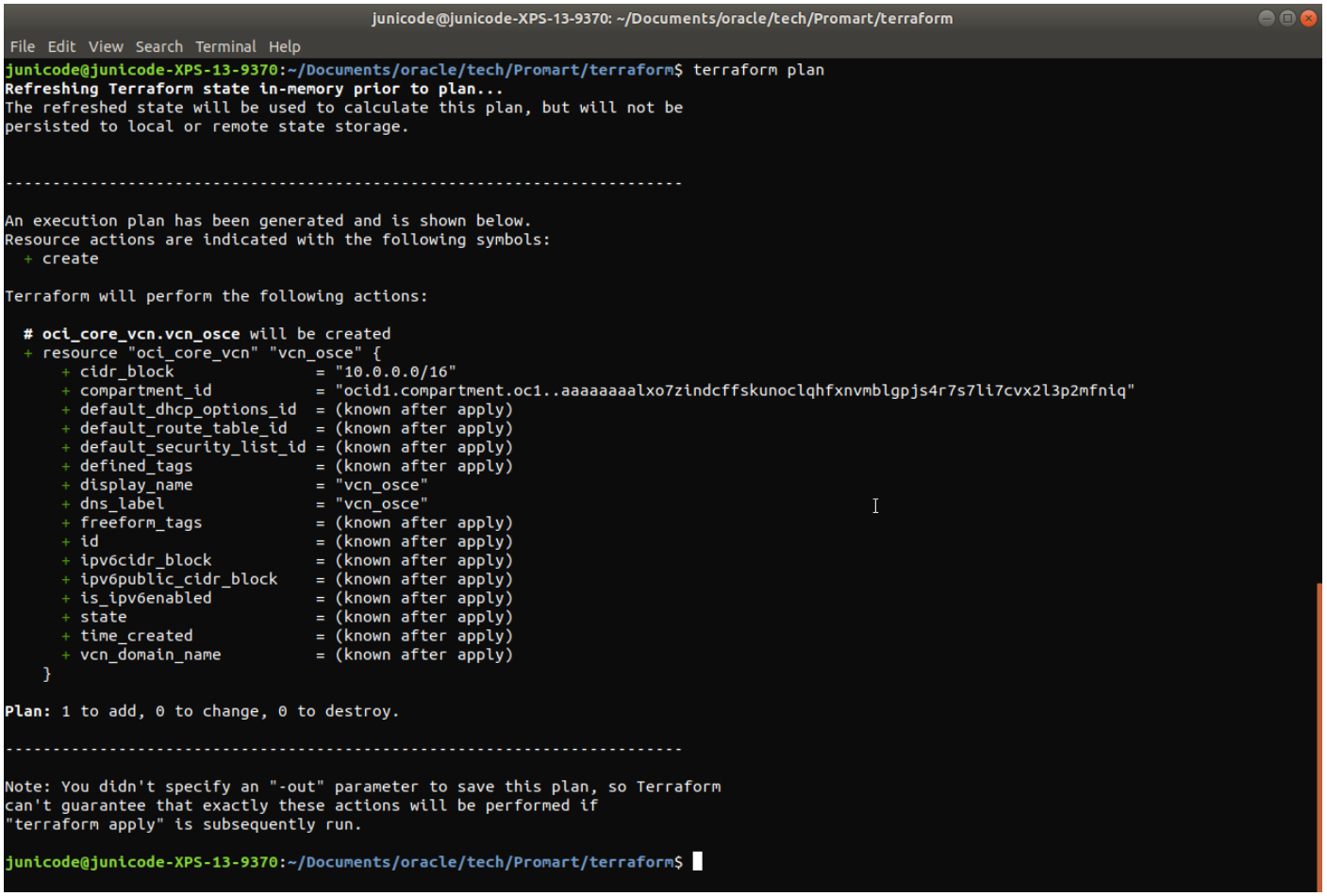
Y se ejecutan los siguientes comandos:

```
$ source env-vars.sh
$ terraform init
```



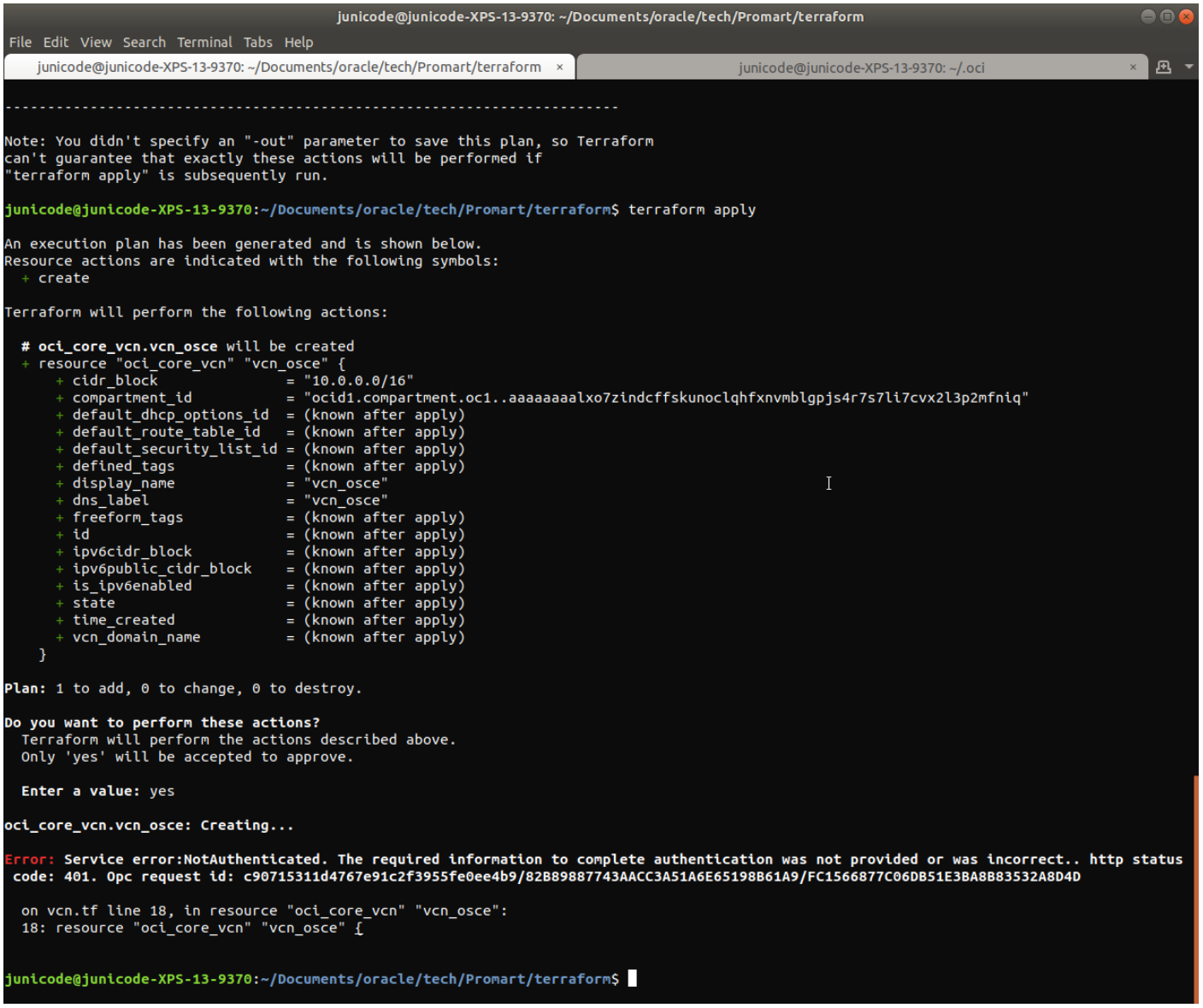
Del mensaje, se corrobora que Terraform se inicializó correctamente.

$ terraform plan



Al ejecutar 'terraform plan', en la línea de comando se puede observar lo que Terraform llevará a cabo. Para el presente caso, la creación de una VCN.

$ terraform apply



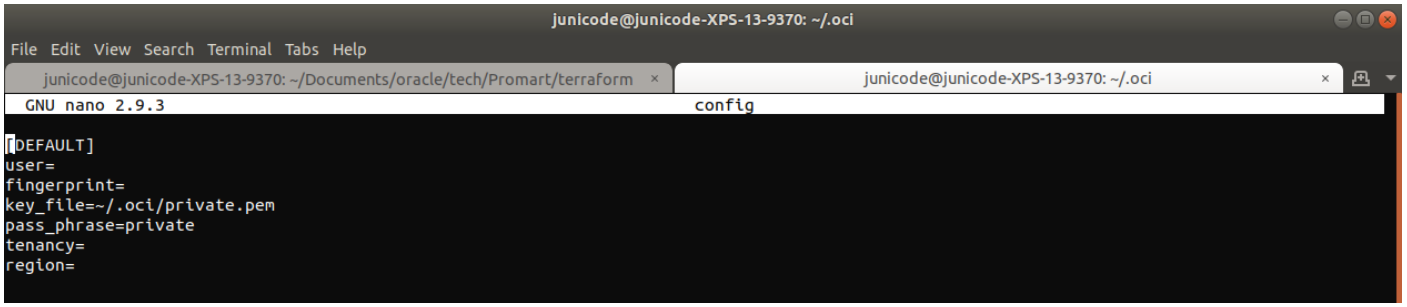
El error se debe al archivo de configuración del OCI CLI (~/.oci/config).

» Using the SDK and CLI Configuration File

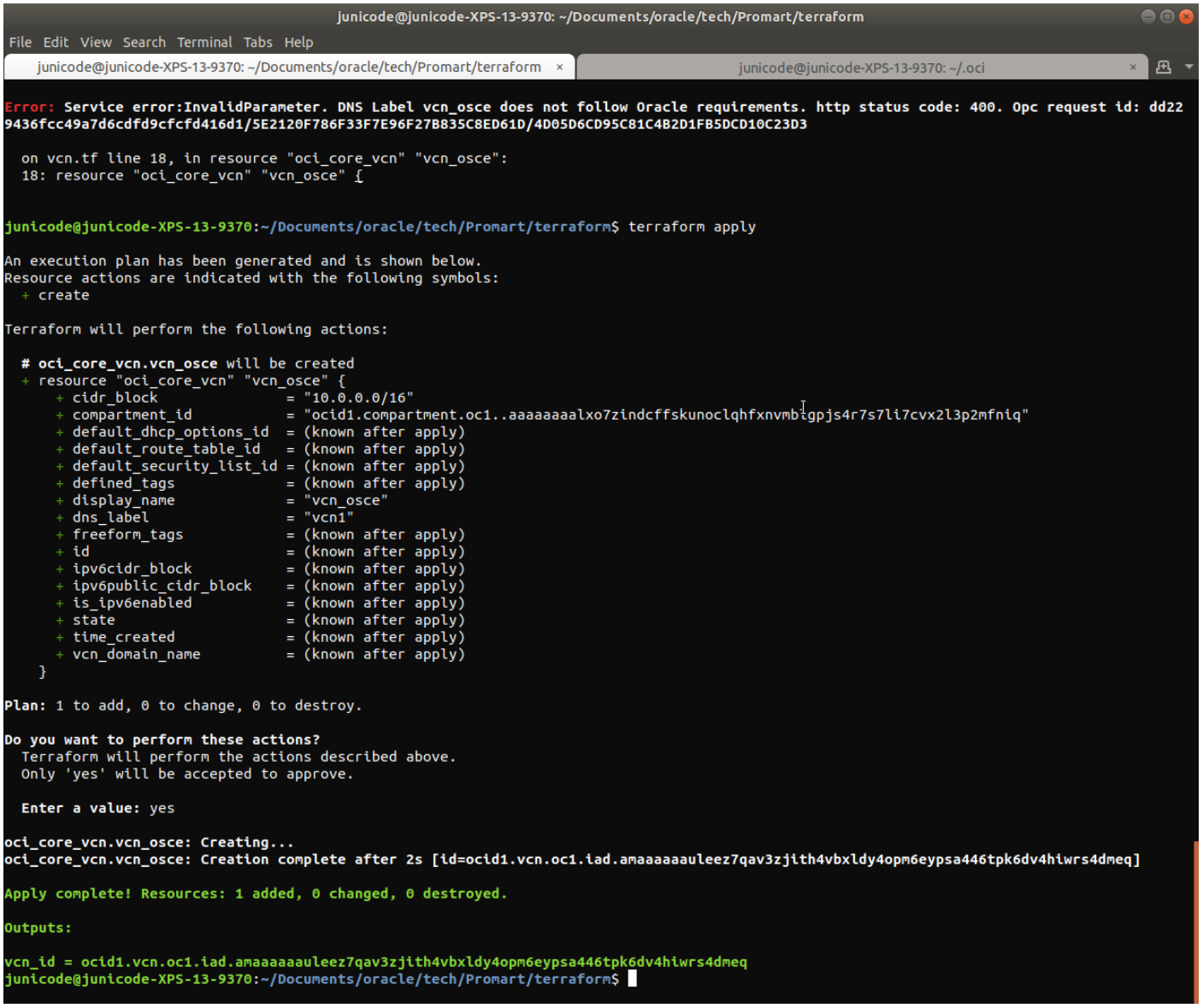
It is possible to define the required provider values in the same `~/.oci/config` file that the SDKs and CLI support. For details on setting up this configuration see SDK and CLI Configuration File.

Note: only the `[default]` profile is supported, and the parameter names are slightly different. Provider block from terraform config can be completely removed if all API Key based authentication required values are provided as environment variables, in a `*.tfvars file` or `~/.oci/config`. When using empty provider block, `private_key_password` if required should to be set in `~/.oci/config`.

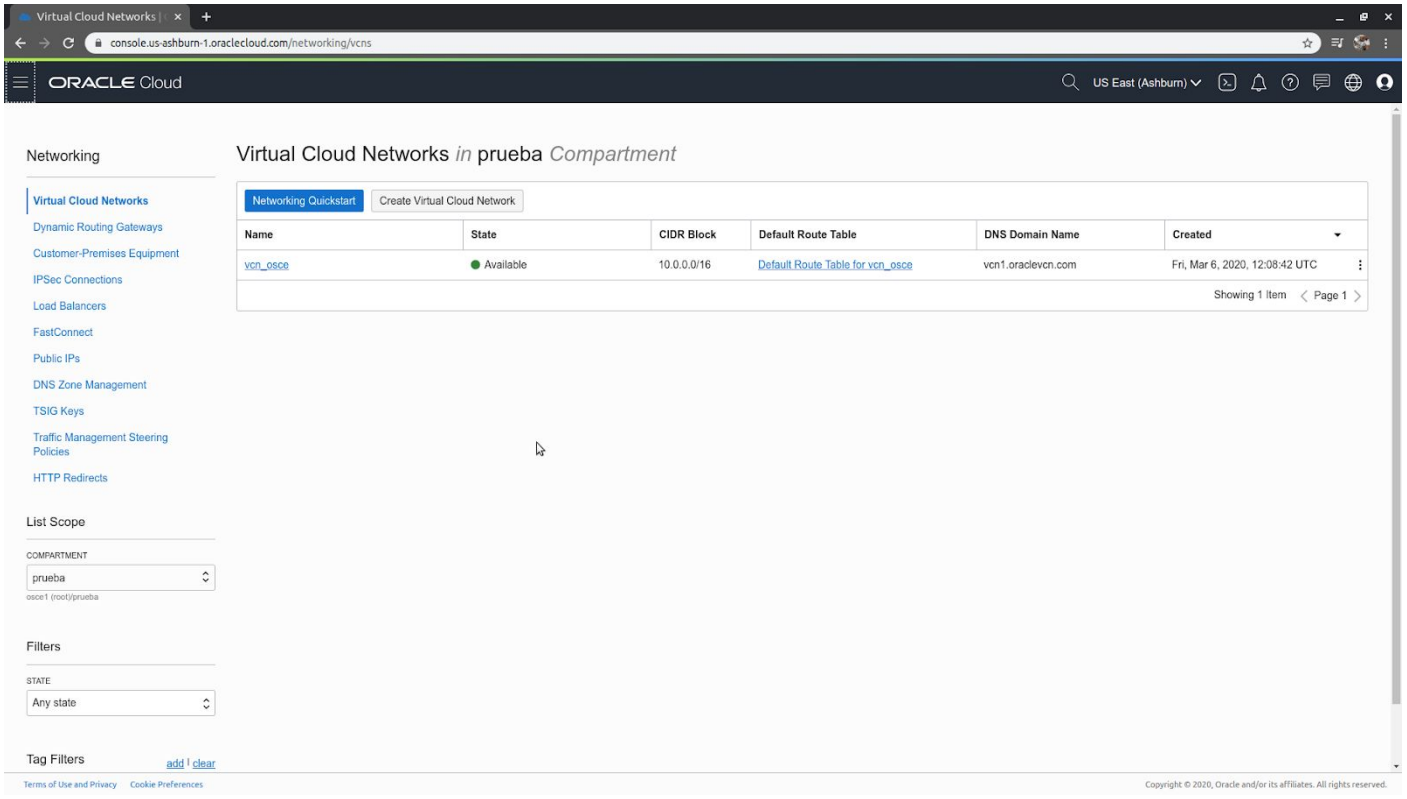
Con el objetivo de resolver el error, se debe modificar el archivo ~/oci/config, teniendo en cuenta la recomendación mencionada en la imagen de arriba.



$ terraform apply



Se verifica en la consola que se creó satisfactoriamente la VCN.


Virtual Cloud Networks
console.us-ashburn-1.oraclecloud.com/networking/vcns
ORACLE Cloud
US East (Ashburn)
Networking
Virtual Cloud Networks
Dynamic Routing Gateways
Customer-Premises Equipment
IPSec Connections
Load Balancers
FastConnect
Public IPs
DNS Zone Management
TSIG Keys
Traffic Management Steering Policies
HTTP Redirects
List Scope
COMPARTMENT
prueba
osce1 (root)/prueba
Filters
STATE
Any state
Tag Filters
add | clear
Virtual Cloud Networks in prueba Compartment
Networking Quickstart
Create Virtual Cloud Network
Name
State
CIDR Block
Default Route Table
DNS Domain Name
Created
vcn_osce
Available
10.0.0.0/16
Default Route Table for vcn_osce
vcn1.oraclevcn.com
Fri, Mar 6, 2020, 12:08:42 UTC
Showing 1 Item
Page 1
Terms of Use and Privacy
Cookie Preferences
Copyright © 2020, Oracle and/or its affiliates. All rights reserved.

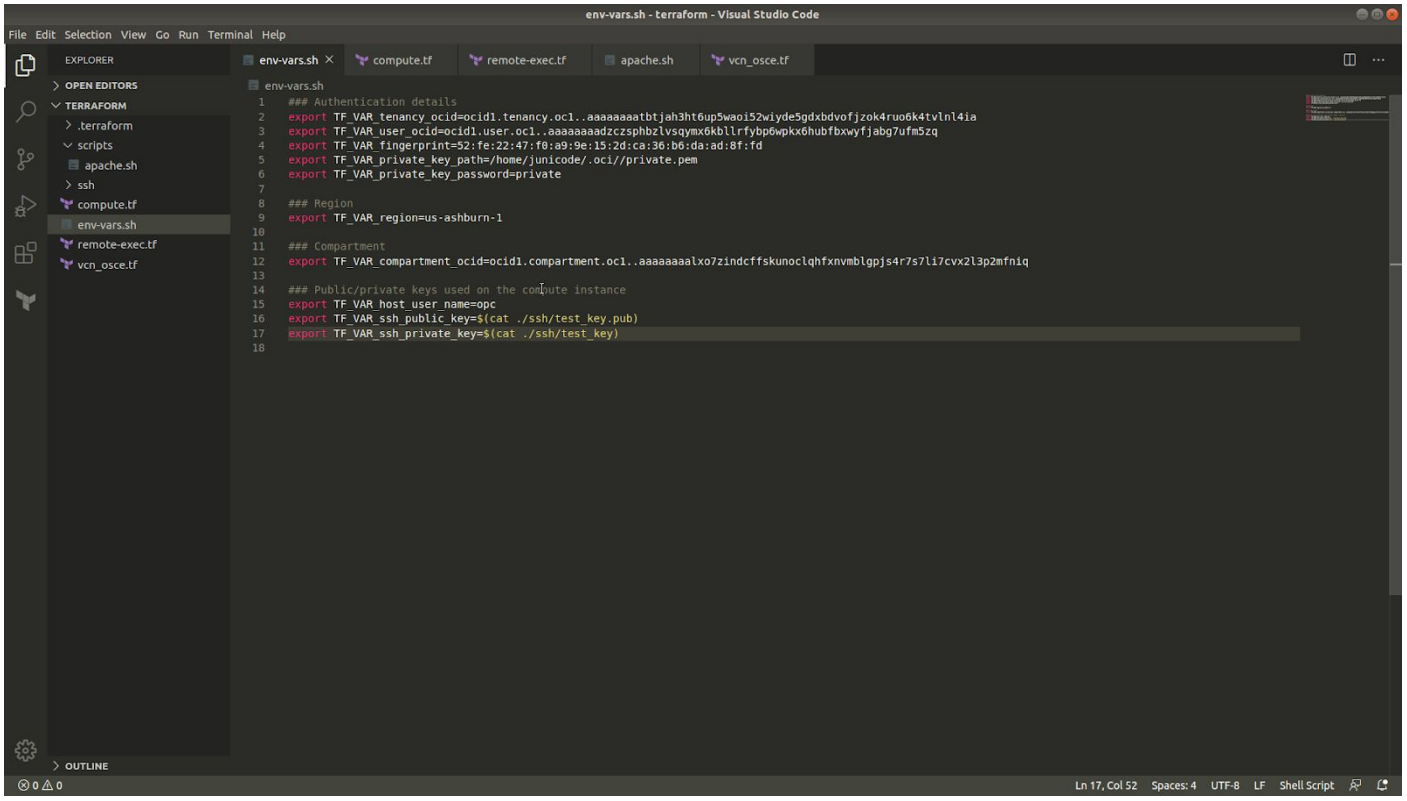
2. Implementación de la Arquitectura

Ahora, se procede a crear la arquitectura presentada al inicio.

$ git clone https://github.com/obi10/iac.git

env-var.sh

Aquí se declaran las variables globales que serán usadas por los demás archivos.



Authentication details

export TF_VAR_tenancy_ocid=ocid1.tenancy.oc1..aaaaaaaatbtjah3ht6up5waoi52wiyde5gdxbdvofjzok4ruo6k4tvlnl4ia
export TF_VAR_user_ocid=ocid1.user.oc1..aaaaaaaadzczsphbzlvsqymx6kbllrfybp6wpkx6hubfbxwyfjabg7ufm5zq
export TF_VAR_fingerprint=52:fe:22:47:f0:a9:9e:15:2d:ca:36:b6:da:ad:8f:fd
export TF_VAR_private_key_path=/home/junicode/.oci//private.pem
export TF_VAR_private_key_password=private

Region

export TF_VAR_region=us-ashburn-1

Compartment

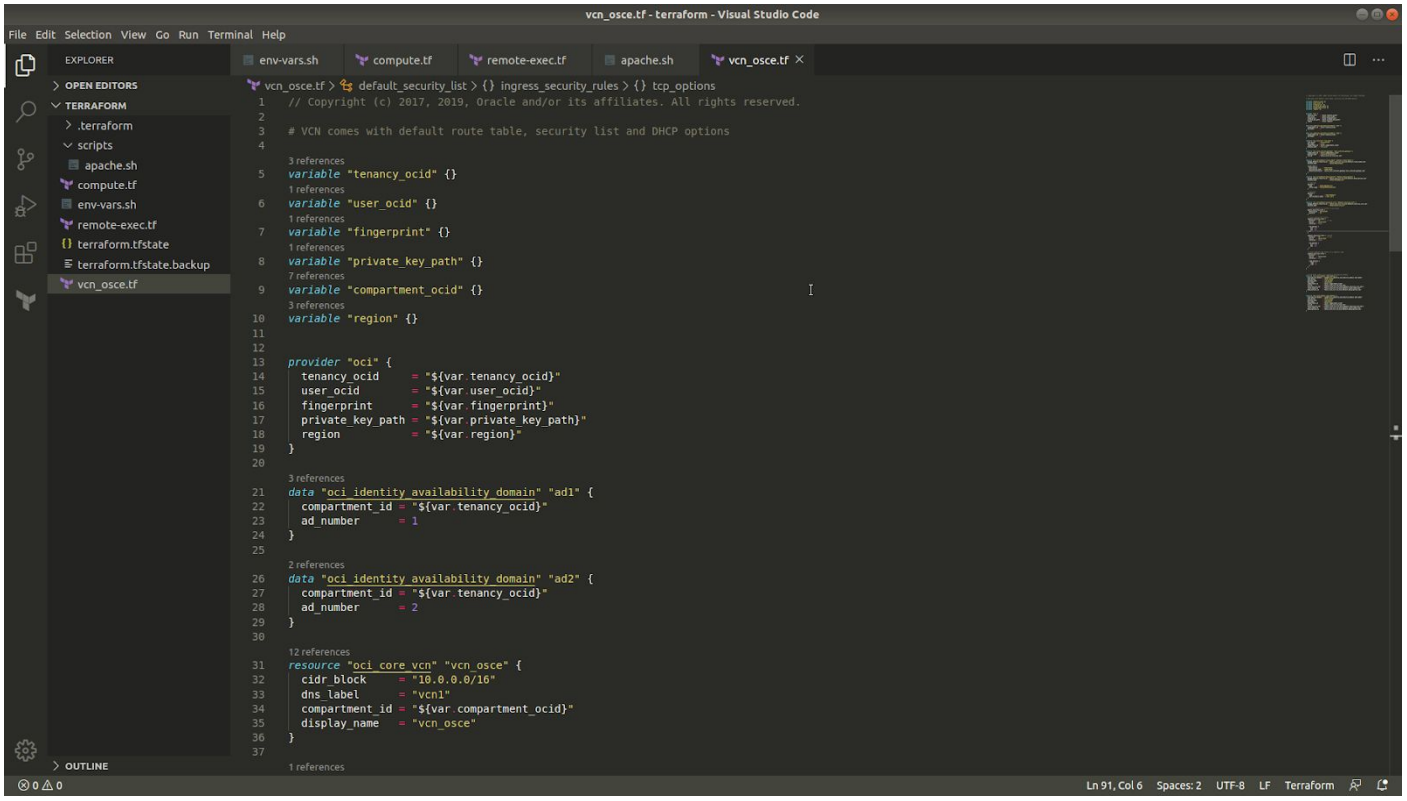
export TF_VAR_compartment_ocid=ocid1.compartment.oc1..aaaaaaaalxo7zindcffskunoclqhfxnvmblgpjs4r7s7li7cvx2l3p2mfniq

Public/private keys used on the compute instance

export TF_VAR_host_user_name=opc
export TF_VAR_ssh_public_key=$(cat ./ssh/test_key.pub)
export TF_VAR_ssh_private_key=$(cat ./ssh/test_key)

vcn_osce.tf

En este archivo se definen los dos dominios de disponibilidad a usar: AD1 y AD2. Asimismo, hace posible la creación de la vcn y sus componentes: internet gateway, route table, dhcp options, security list y subnets.



// Copyright (c) 2017, 2019, Oracle and/or its affiliates. All rights reserved.

 # VCN comes with default route table, security list and DHCP options

variable "tenancy_ocid" {}
variable "user_ocid" {}
variable "fingerprint" {}
variable "private_key_path" {}
variable "compartment_ocid" {}
variable "region" {}

provider "oci" {
* tenancy_ocid   = "${var.tenancy_ocid}"*
* user_ocid   = "${var.user_ocid}"*
* fingerprint   = "${var.fingerprint}"*
* private_key_path = "${var.private_key_path}"*
* region   = "${var.region}"*
}

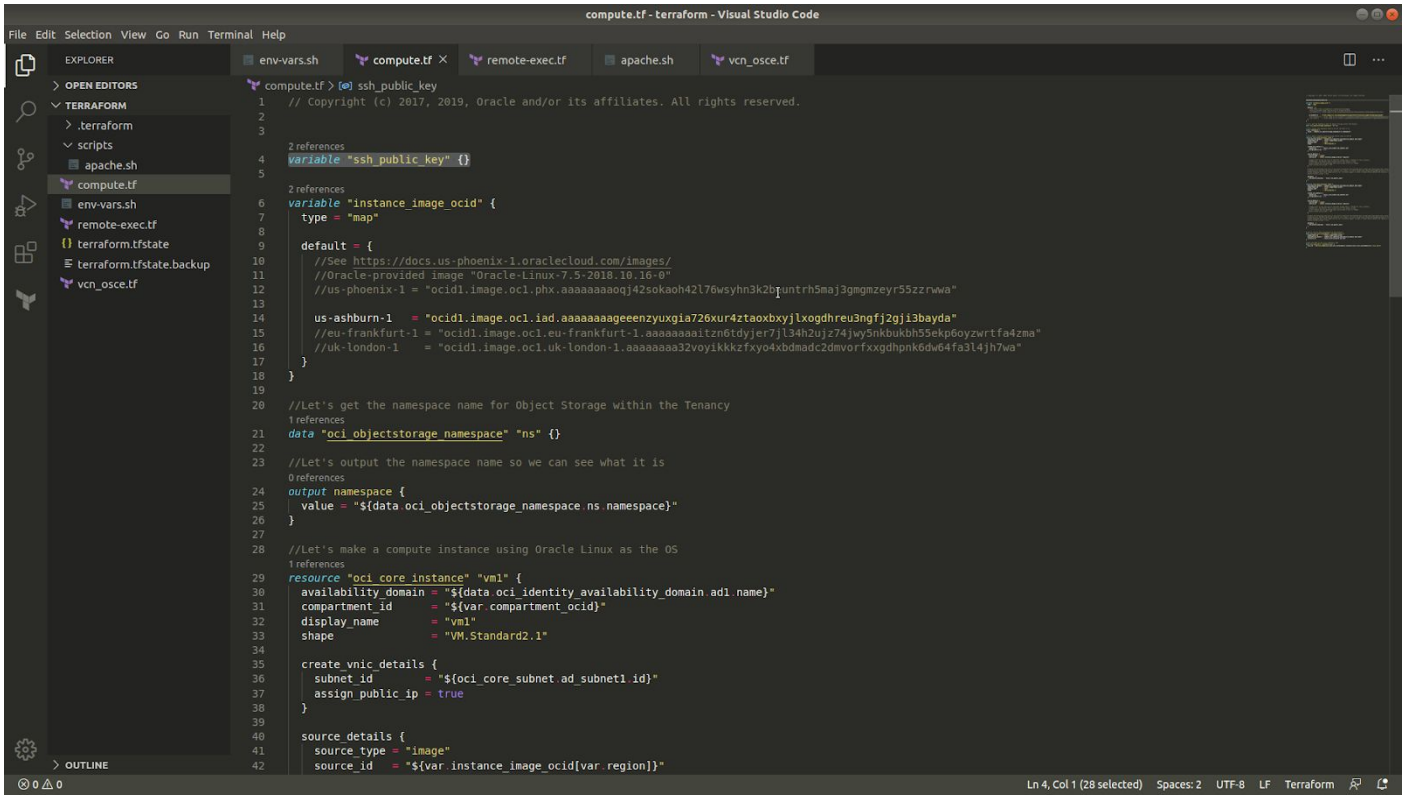
data "oci_identity_availability_domain" "ad1" {
* compartment_id = "${var.tenancy_ocid}"*

```
  ad_number      = 1
}

data "oci_identity_availability_domain" "ad2" {
  compartment_id = "${var.tenancy_ocid}"
  ad_number      = 2
}

resource "oci_core_vcn" "vcn_osce" {
  cidr_block     = "10.0.0.0/16"
  dns_label      = "vcn1"
  compartment_id = "${var.compartment_ocid}"
  display_name   = "vcn_osce"
}

resource "oci_core_internet_gateway" "test_internet_gateway" {
  compartment_id = "${var.compartment_ocid}"
  display_name   = "testInternetGateway"
  vcn_id         = "${oci_core_vcn.vcn_osce.id}"
}

resource "oci_core_default_route_table" "default_route_table" {
  manage_default_resource_id = "${oci_core_vcn.vcn_osce.default_route_table_id}"
  display_name               = "defaultRouteTable"

  route_rules {
    destination       = "0.0.0.0/0"
    destination_type  = "CIDR_BLOCK"
    network_entity_id = "${oci_core_internet_gateway.test_internet_gateway.id}"
  }
}

resource "oci_core_default_dhcp_options" "default_dhcp_options" {
  manage_default_resource_id = "${oci_core_vcn.vcn_osce.default_dhcp_options_id}"
  display_name               = "defaultDhcpOptions"

  // required
  options {
    type        = "DomainNameServer"
    server_type = "VcnLocalPlusInternet"
  }

  // optional
  options {
    type               = "SearchDomain"
    search_domain_names = ["abc.com"]
  }
}

resource "oci_core_default_security_list" "default_security_list" {
  manage_default_resource_id = "${oci_core_vcn.vcn_osce.default_security_list_id}"
  display_name               = "defaultSecurityList"

  // allow outbound tcp traffic on all ports
  egress_security_rules {
    destination = "0.0.0.0/0"
    protocol    = "6"
  }

  // allow inbound ssh traffic
  ingress_security_rules {
    protocol  = "6"         // tcp
    source    = "0.0.0.0/0"
    stateless = false
```

```
    tcp_options {
      min = 22
      max = 22
    }
  }

  ingress_security_rules { //apache
    protocol  = "6"         // tcp
    source    = "0.0.0.0/0"
    stateless = false

    tcp_options {
      min = 80
      max = 80
    }
  }

  // allow inbound icmp traffic of a specific type
  ingress_security_rules {
    protocol  = 1
    source    = "0.0.0.0/0"
    stateless = true

    icmp_options {
      type = 3
      code = 4
    }
  }
}



// An AD based subnet will supply an Availability Domain
resource "oci_core_subnet" "ad_subnet1" {
  availability_domain = "${data.oci_identity_availability_domain.ad1.name}"
  cidr_block          = "10.0.1.0/24"
  display_name        = "subred-AD1"
  dns_label           = "ad1subnet"
  compartment_id      = "${var.compartment_ocid}"
  vcn_id              = "${oci_core_vcn.vcn_osce.id}"
  security_list_ids   = ["${oci_core_vcn.vcn_osce.default_security_list_id}"]
  route_table_id      = "${oci_core_vcn.vcn_osce.default_route_table_id}"
  dhcp_options_id     = "${oci_core_vcn.vcn_osce.default_dhcp_options_id}"
}


resource "oci_core_subnet" "ad_subnet2" {
  availability_domain = "${data.oci_identity_availability_domain.ad2.name}"
  cidr_block          = "10.0.2.0/24"
  display_name        = "subred-AD2"
  dns_label           = "ad2subnet"
  compartment_id      = "${var.compartment_ocid}"
  vcn_id              = "${oci_core_vcn.vcn_osce.id}"
  security_list_ids   = ["${oci_core_vcn.vcn_osce.default_security_list_id}"]
  route_table_id      = "${oci_core_vcn.vcn_osce.default_route_table_id}"
  dhcp_options_id     = "${oci_core_vcn.vcn_osce.default_dhcp_options_id}"
}
```

compute.tf

Este archivo permite crear las instancias de cómputo vm1 y vm2 en los dominios de disponibilidad definidos en *vcn_osce.tf*.



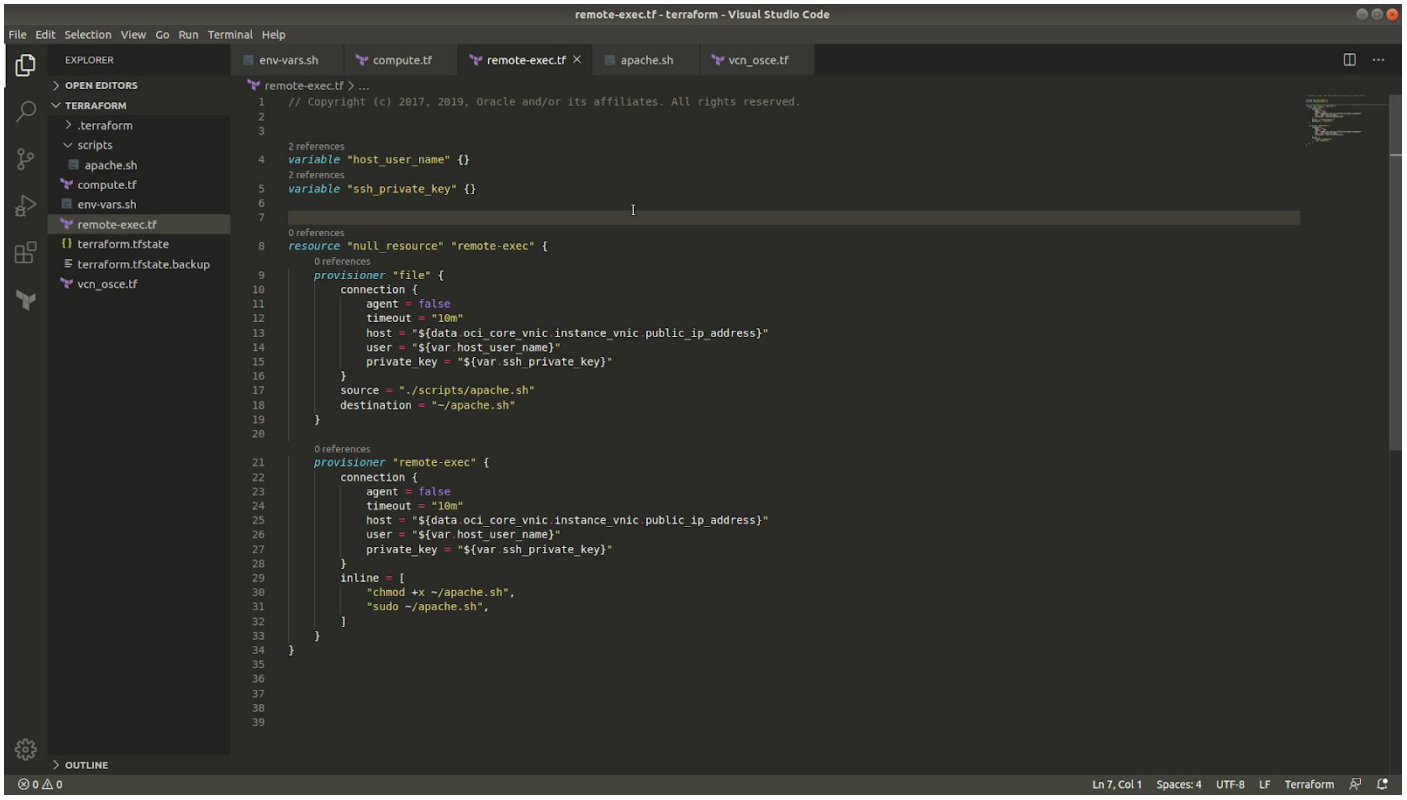

```
// Copyright (c) 2017, 2019, Oracle and/or its affiliates. All rights reserved.


variable "ssh_public_key" {}

variable "instance_image_ocid" {
  type = "map"

  default = {
    //See https://docs.us-phoenix-1.oraclecloud.com/images/
    //Oracle-provided image "Oracle-Linux-7.5-2018.10.16-0"
    //us-phoenix-1 = "ocid1.image.oc1.phx.aaaaaaaaoqj42sokaoh42l76wsyhn3k2beuntrh5maj3gmgmzeyr55zzrwwa"

    us-ashburn-1   = "ocid1.image.oc1.iad.aaaaaaaageeenzyuxgia726xur4ztaoxbxyjlxogdhreu3ngfj2gji3bayda"
    //eu-frankfurt-1 = "ocid1.image.oc1.eu-frankfurt-1.aaaaaaaaitzn6tdyjer7jl34h2ujz74jwy5nkbukbh55ekp6oyzwrtfa4zma"
    //uk-london-1    = "ocid1.image.oc1.uk-london-1.aaaaaaaa32voyikkkzfxyo4xbdmadc2dmvorfxxgdhpnk6dw64fa3l4jh7wa"
  }
}

//Let's get the namespace name for Object Storage within the Tenancy
data "oci_objectstorage_namespace" "ns" {}

//Let's output the namespace name so we can see what it is
output namespace {
  value = "${data.oci_objectstorage_namespace.ns.namespace}"
}

//Let's make a compute instance using Oracle Linux as the OS
resource "oci_core_instance" "vm1" {
  availability_domain = "${data.oci_identity_availability_domain.ad1.name}"
  compartment_id      = "${var.compartment_ocid}"
  display_name        = "vm1"
  shape               = "VM.Standard2.1"

  create_vnic_details {
    subnet_id        = "${oci_core_subnet.ad_subnet1.id}"
```

```
    assign_public_ip = true
  }

  source_details {
    source_type = "image"
    source_id   = "${var.instance_image_ocid[var.region]}"

    # Apply this to set the size of the boot volume that's created for this instance.
    # Otherwise, the default boot volume size of the image is used.
    # This should only be specified when source_type is set to "image".
    #boot_volume_size_in_gbs = "60"
  }

  # Apply the following flag only if you wish to preserve the attached boot volume upon destroying this instance
  # Setting this and destroying the instance will result in a boot volume that should be managed outside of this config.
  # When changing this value, make sure to run 'terraform apply' so that it takes effect before the resource is destroyed.
  #preserve_boot_volume = true

  metadata = {
    ssh_authorized_keys = "${var.ssh_public_key}"
  }
}

resource "oci_core_instance" "vm2" {
  availability_domain = "${data.oci_identity_availability_domain.ad2.name}"
  compartment_id      = "${var.compartment_ocid}"
  display_name        = "vm2"
  shape               = "VM.Standard2.1"

  create_vnic_details {
    subnet_id        = "${oci_core_subnet.ad_subnet2.id}"
    assign_public_ip = true
  }

  source_details {
    source_type = "image"
    source_id   = "${var.instance_image_ocid[var.region]}"

    # Apply this to set the size of the boot volume that's created for this instance.
    # Otherwise, the default boot volume size of the image is used.
    # This should only be specified when source_type is set to "image".
    #boot_volume_size_in_gbs = "60"
  }

  # Apply the following flag only if you wish to preserve the attached boot volume upon destroying this instance
  # Setting this and destroying the instance will result in a boot volume that should be managed outside of this config.
  # When changing this value, make sure to run 'terraform apply' so that it takes effect before the resource is destroyed.
  #preserve_boot_volume = true

  metadata = {
    ssh_authorized_keys = "${var.ssh_public_key}"
  }
}


# Gets a list of vNIC attachments on the instance
data "oci_core_vnic_attachments" "instance_vnics" {
  compartment_id      = "${var.compartment_ocid}"
  availability_domain = "${data.oci_identity_availability_domain.ad1.name}"
  instance_id         = "${oci_core_instance.vm1.id}"
}

# Gets the OCID of the first (default) vNIC
data "oci_core_vnic" "instance_vnic" {
  vnic_id = "${lookup(data.oci_core_vnic_attachments.instance_vnics.vnic_attachments[0],"vnic_id")}"
```

}

remote-exec.tf

Este archivo permite la conexión a la instancia vm1 para ejecutar el script *apache.sh*.



// Copyright (c) 2017, 2019, Oracle and/or its affiliates. All rights reserved.

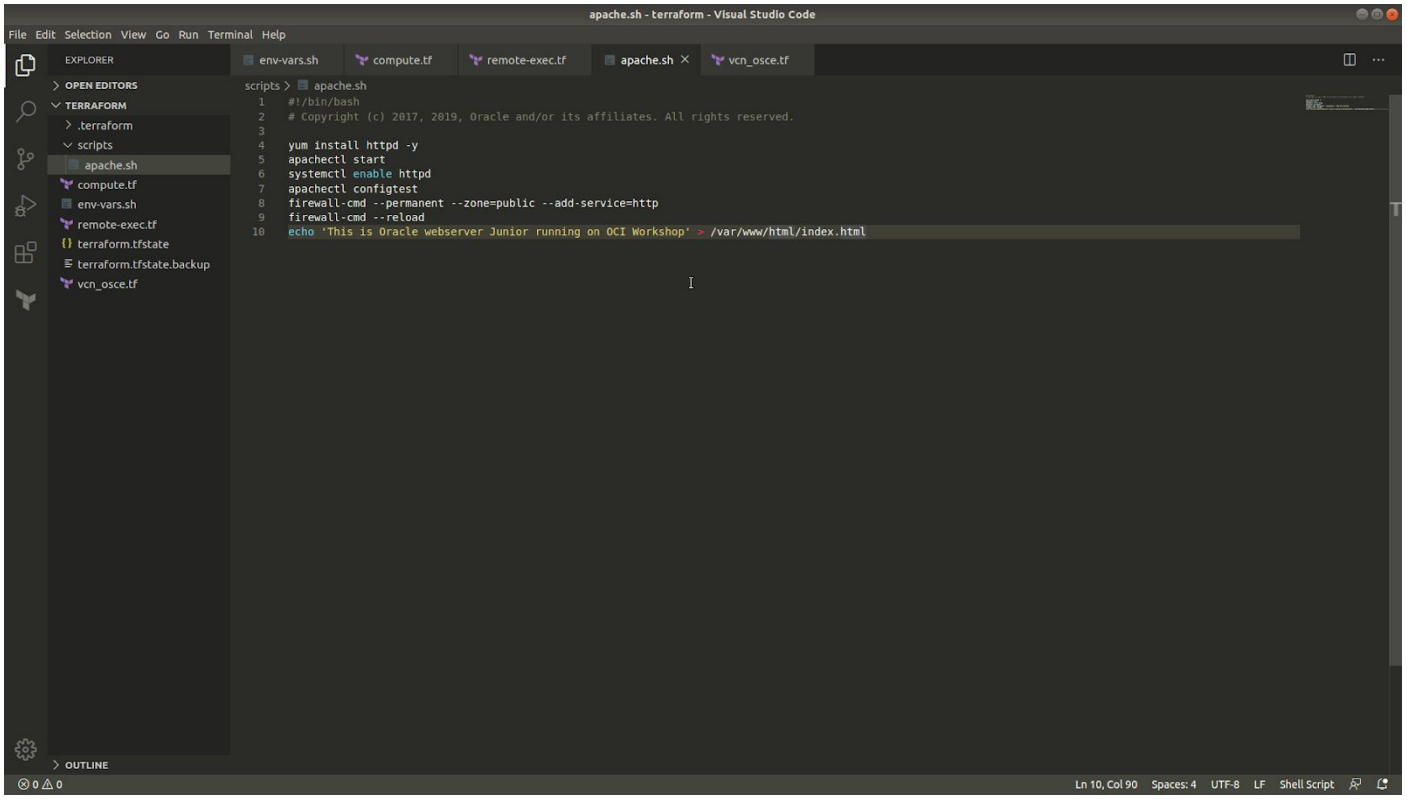
variable "host_user_name" {}
variable "ssh_private_key" {}

resource "null_resource" "remote-exec" {
provisioner "file" {
connection {
agent = false
timeout = "10m"
host = "${data.oci_core_vnic.instance_vnic.public_ip_address}"
user = "${var.host_user_name}"
private_key = "${var.ssh_private_key}"
}
source = "./scripts/apache.sh"
destination = "~/apache.sh"
}

provisioner "remote-exec" {
connection {
agent = false
timeout = "10m"
host = "${data.oci_core_vnic.instance_vnic.public_ip_address}"
user = "${var.host_user_name}"
private_key = "${var.ssh_private_key}"
}
inline = [
"chmod +x ~/apache.sh",
"sudo ~/apache.sh",
]

```
    }
}
```

apache.sh
Script que instala apache.

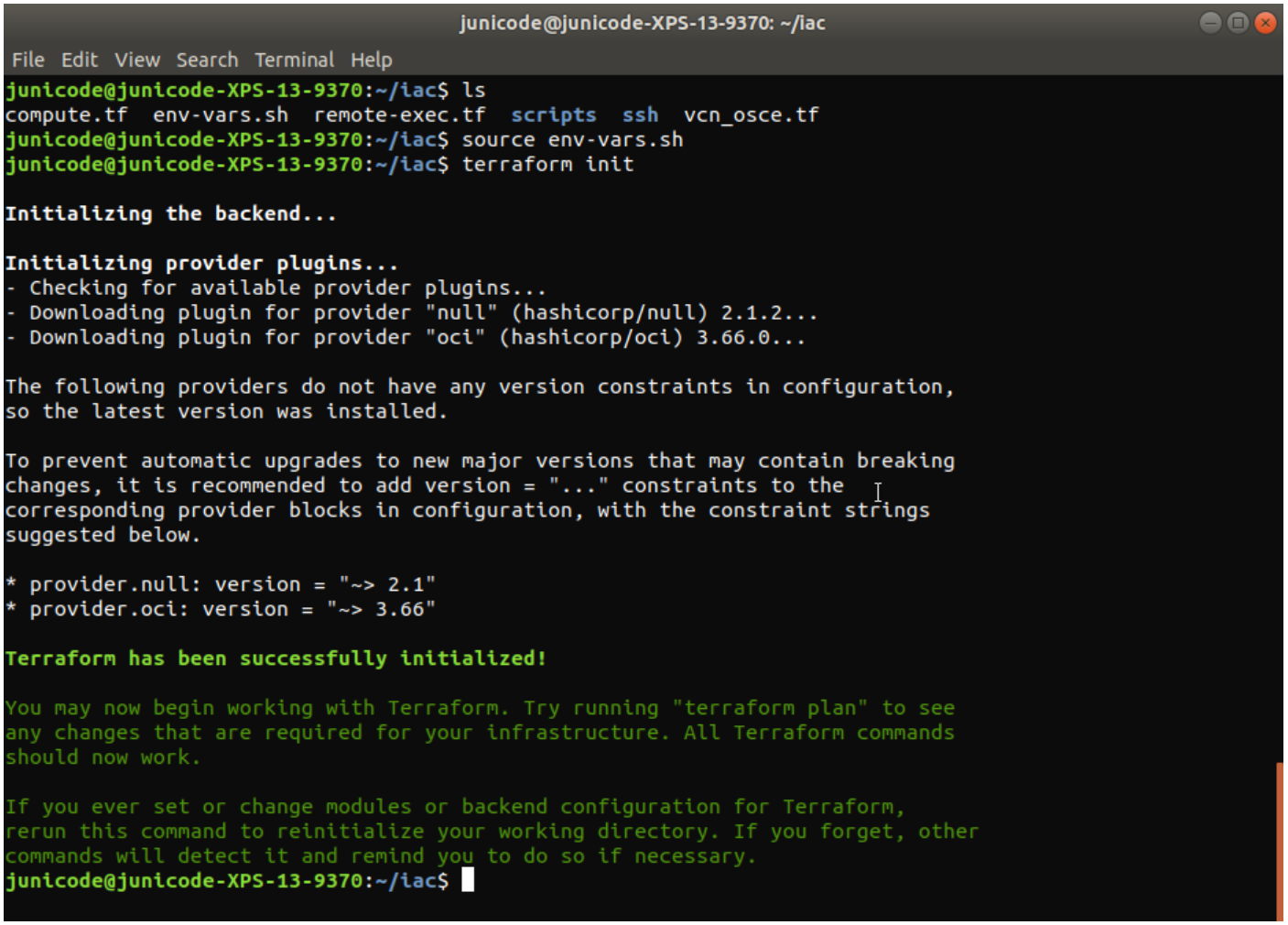


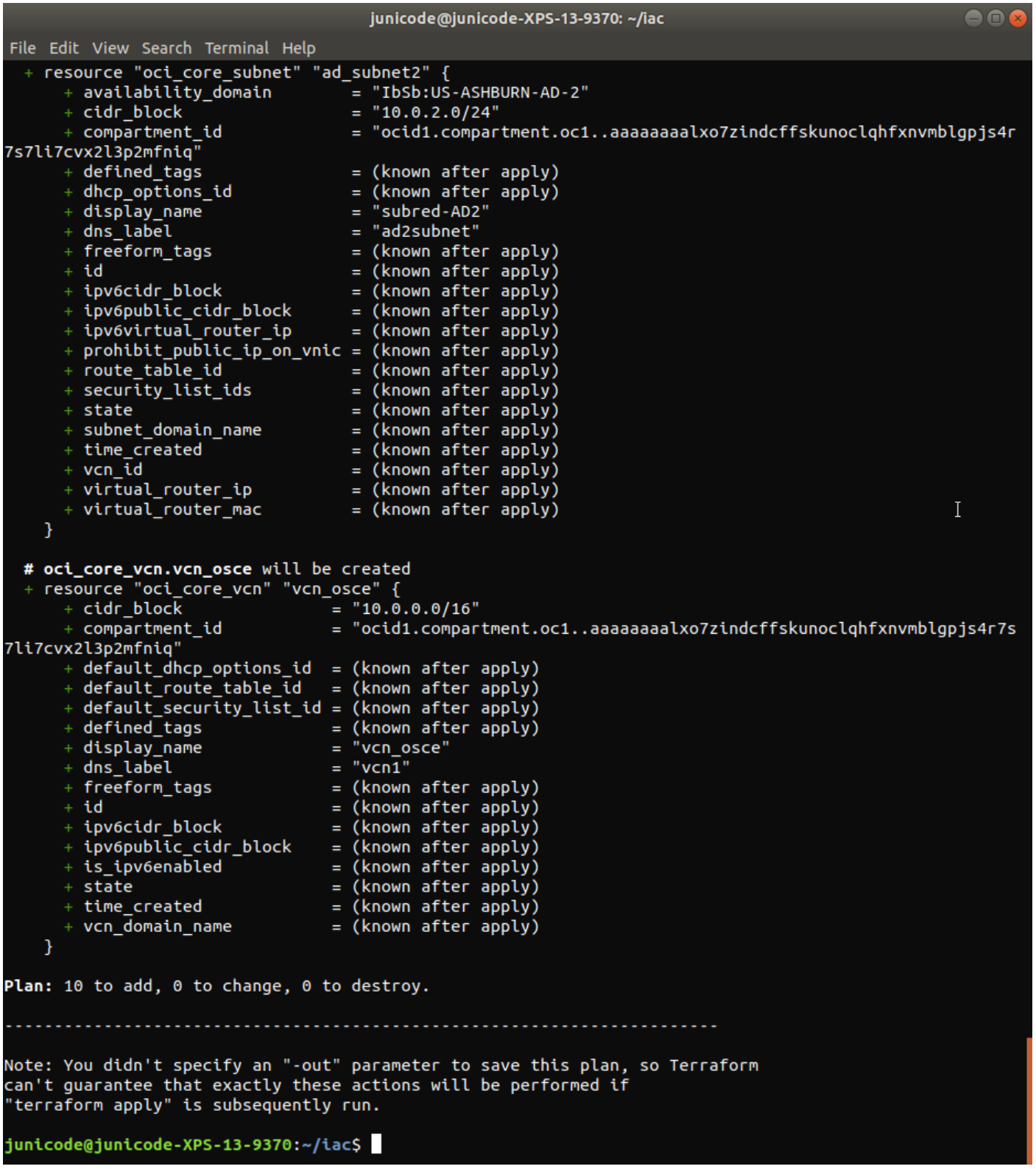
#!/bin/bash
 # Copyright (c) 2017, 2019, Oracle and/or its affiliates. All rights reserved.

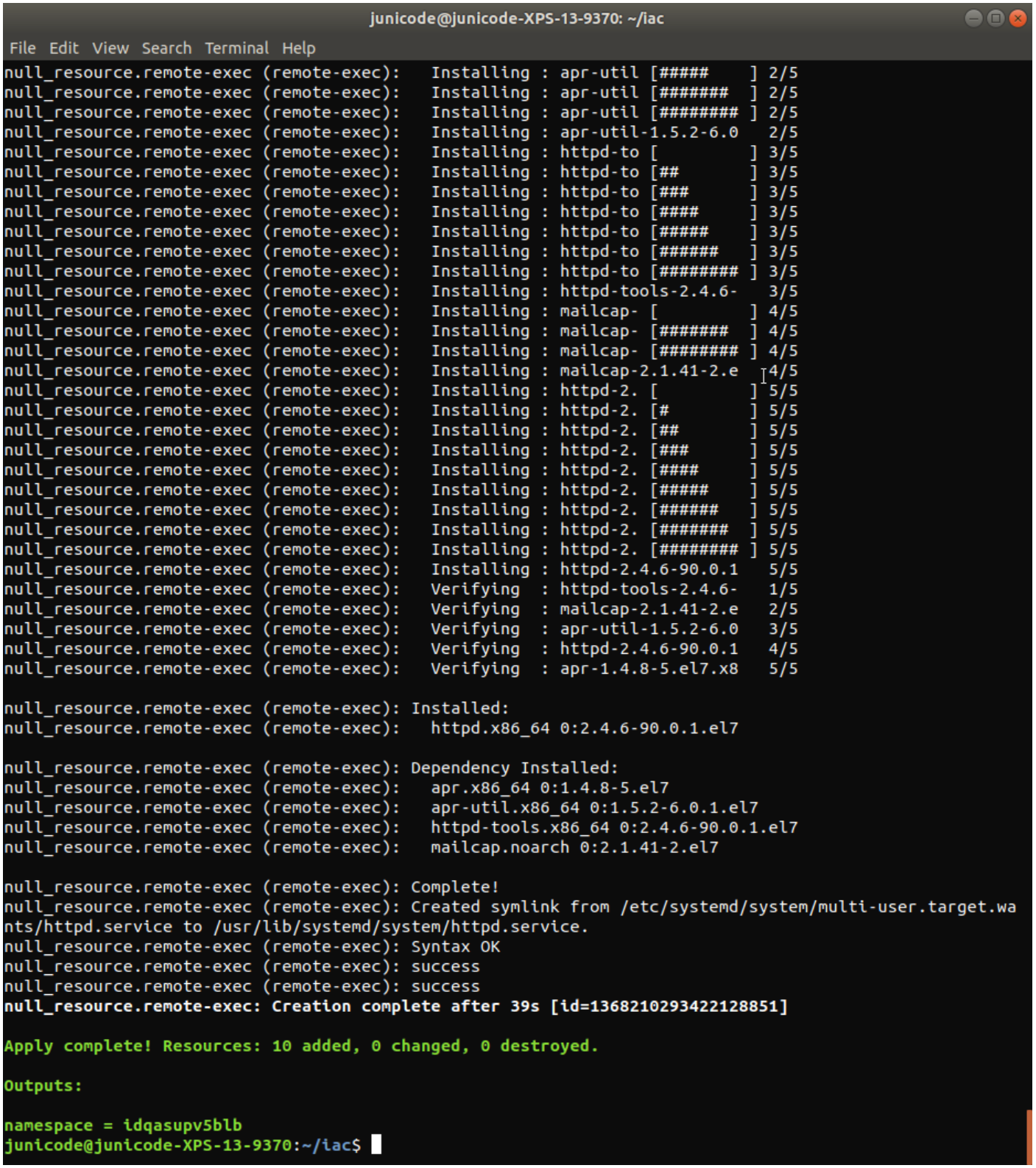
yum install httpd -y
apachectl start
systemctl enable httpd
apachectl configtest
firewall-cmd --permanent --zone=public --add-service=http
firewall-cmd --reload
echo 'This is Oracle webserver Junior running on OCI Workshop' > /var/www/html/index.html

Se deben ejecutar los siguientes comandos:

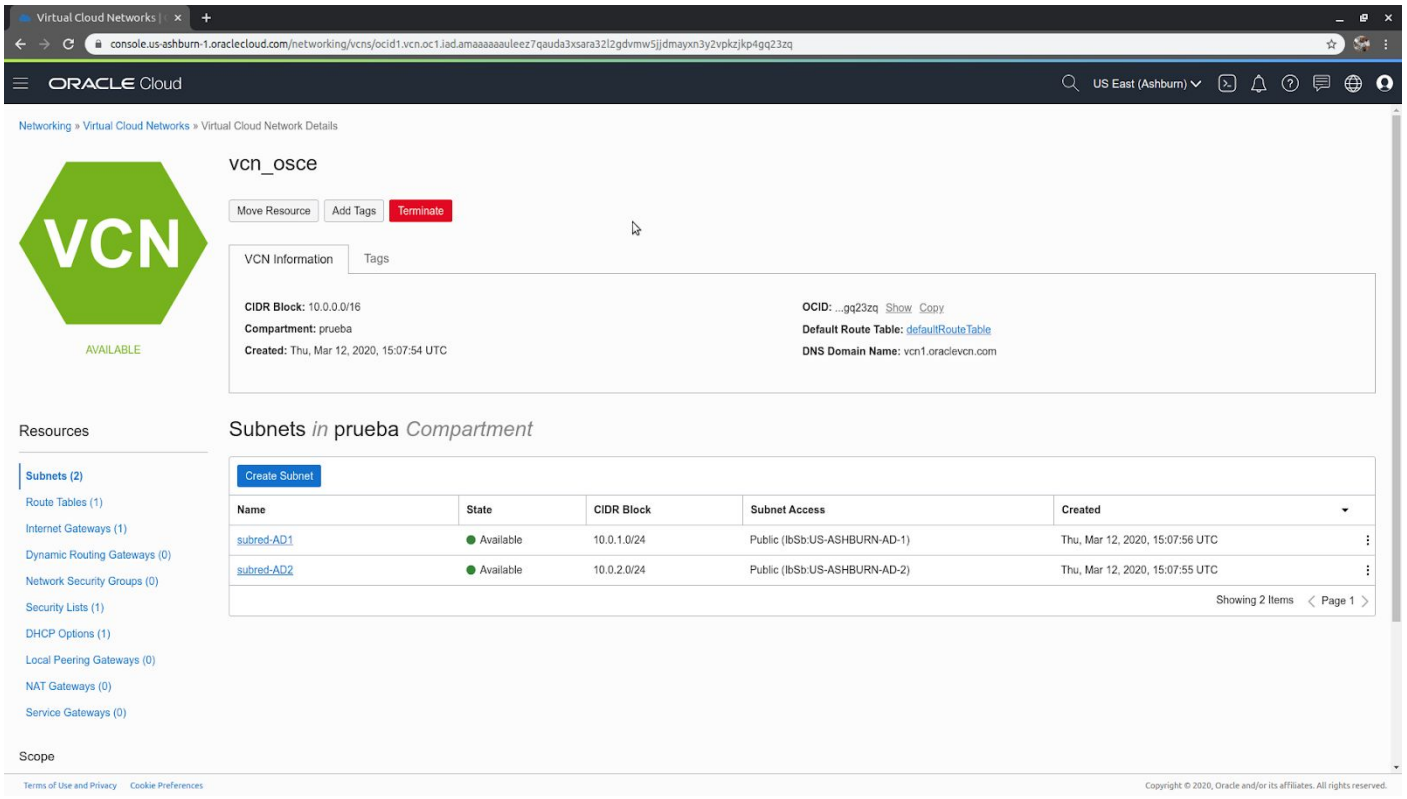
$ source env-vars.sh
$ terraform init
$ terraform plan
$ terraform apply

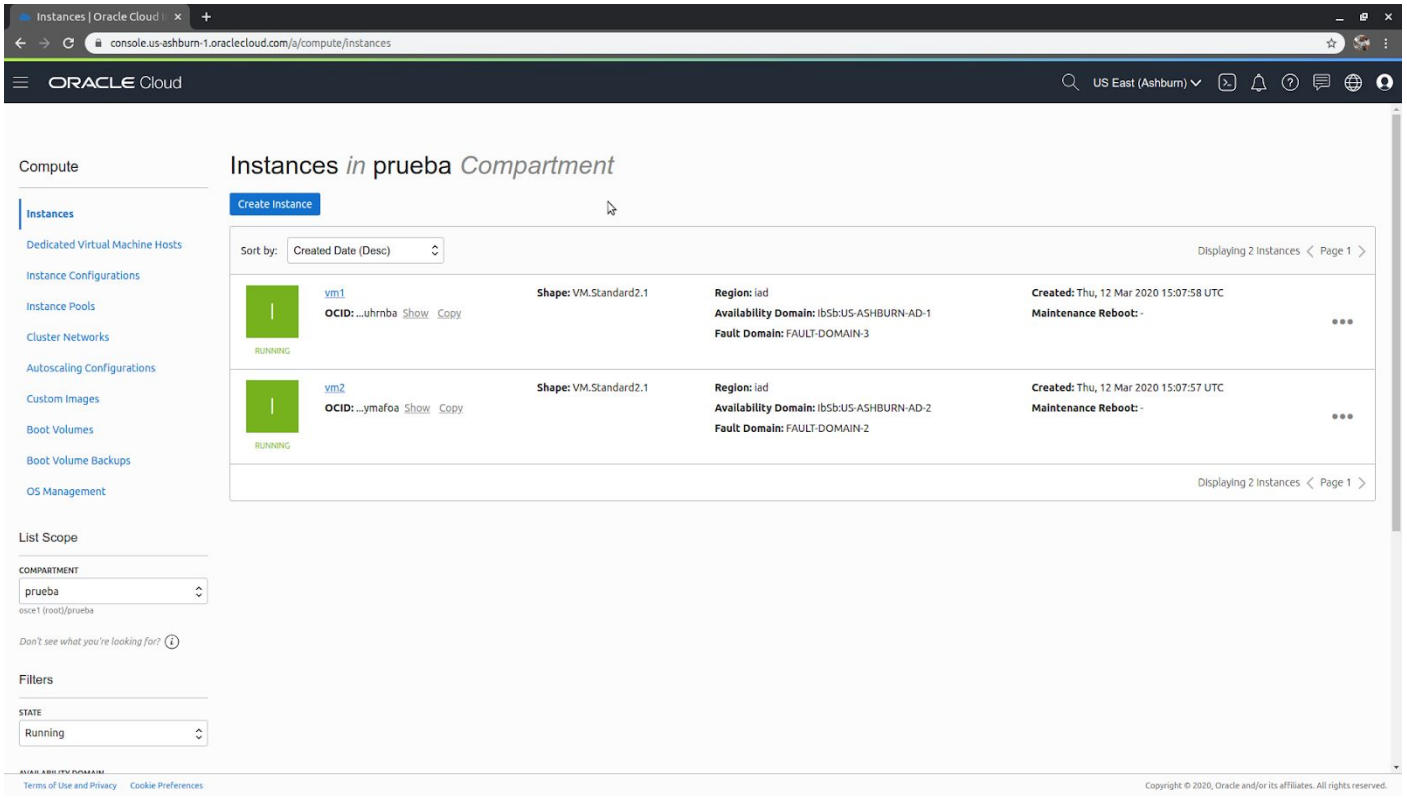


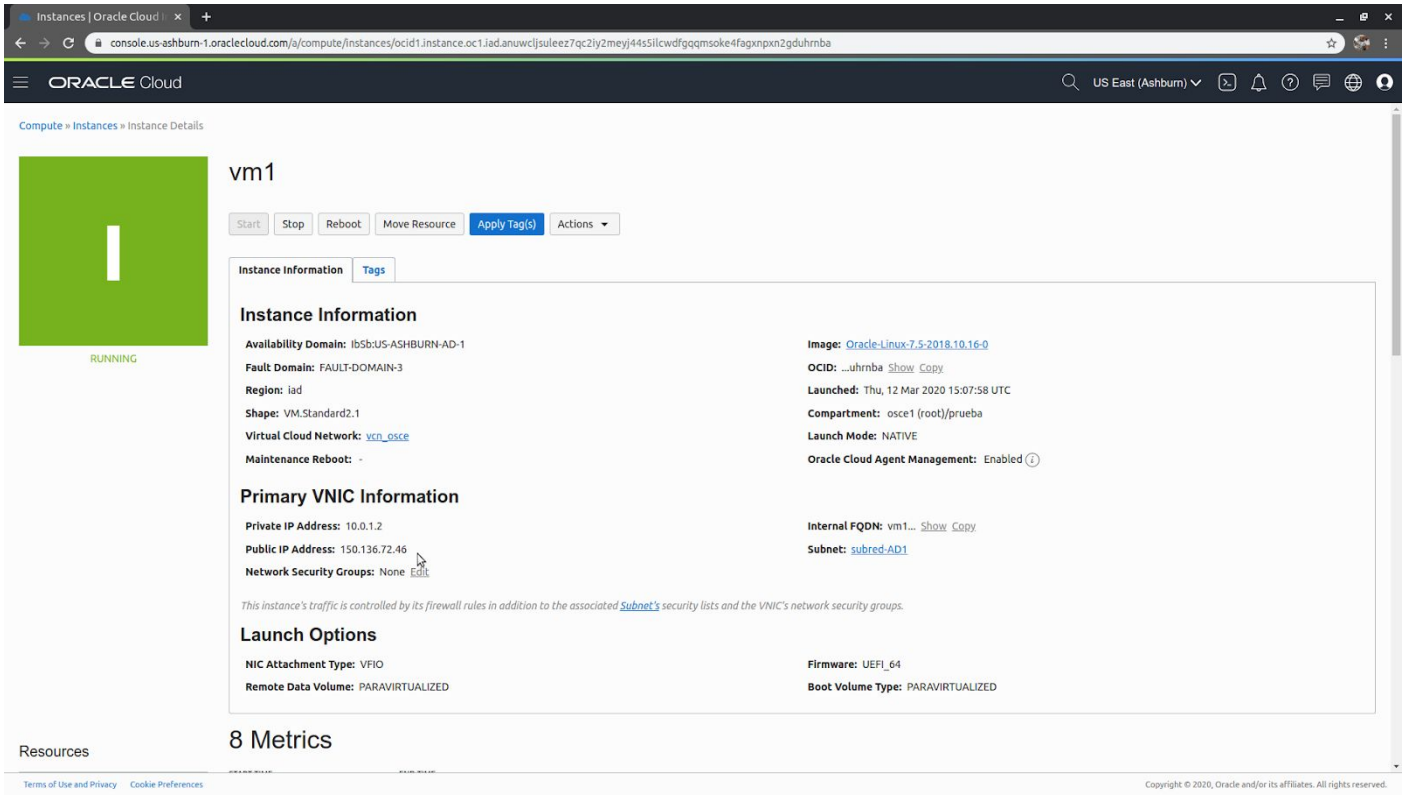

junicode@junicode-XPS-13-9370: ~/iac
File Edit View Search Terminal Help
+ resource "oci_core_subnet" "ad_subnet2" {
+ availability_domain = "IbSb:US-ASHBURN-AD-2"
+ cidr_block = "10.0.2.0/24"
+ compartment_id = "ocid1.compartment.oc1..aaaaaaaalxo7zindcffskunoclqhfxnvmblgpjs4r
7s7li7cvx2l3p2mfniq"
+ defined_tags = (known after apply)
+ dhcp_options_id = (known after apply)
+ display_name = "subred-AD2"
+ dns_label = "ad2subnet"
+ freeform_tags = (known after apply)
+ id = (known after apply)
+ ipv6cidr_block = (known after apply)
+ ipv6public_cidr_block = (known after apply)
+ ipv6virtual_router_ip = (known after apply)
+ prohibit_public_ip_on_vnic = (known after apply)
+ route_table_id = (known after apply)
+ security_list_ids = (known after apply)
+ state = (known after apply)
+ subnet_domain_name = (known after apply)
+ time_created = (known after apply)
+ vcn_id = (known after apply)
+ virtual_router_ip = (known after apply)
+ virtual_router_mac = (known after apply)
}
# oci_core_vcn.vcn_osce will be created
+ resource "oci_core_vcn" "vcn_osce" {
+ cidr_block = "10.0.0.0/16"
+ compartment_id = "ocid1.compartment.oc1..aaaaaaaalxo7zindcffskunoclqhfxnvmblgpjs4r7s
7li7cvx2l3p2mfniq"
+ default_dhcp_options_id = (known after apply)
+ default_route_table_id = (known after apply)
+ default_security_list_id = (known after apply)
+ defined_tags = (known after apply)
+ display_name = "vcn_osce"
+ dns_label = "vcn1"
+ freeform_tags = (known after apply)
+ id = (known after apply)
+ ipv6cidr_block = (known after apply)
+ ipv6public_cidr_block = (known after apply)
+ is_ipv6enabled = (known after apply)
+ state = (known after apply)
+ time_created = (known after apply)
+ vcn_domain_name = (known after apply)
}
Plan: 10 to add, 0 to change, 0 to destroy.
Note: You didn't specify an "-out" parameter to save this plan, so Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.
junicode@junicode-XPS-13-9370:~/iac$


junicode@junicode-XPS-13-9370: ~/iac
File Edit View Search Terminal Help
null_resource.remote-exec (remote-exec): Installing : apr-util [#####] 2/5
null_resource.remote-exec (remote-exec): Installing : apr-util [#######] 2/5
null_resource.remote-exec (remote-exec): Installing : apr-util [########] 2/5
null_resource.remote-exec (remote-exec): Installing : apr-util-1.5.2-6.0 2/5
null_resource.remote-exec (remote-exec): Installing : httpd-to [ ] 3/5
null_resource.remote-exec (remote-exec): Installing : httpd-to [##] 3/5
null_resource.remote-exec (remote-exec): Installing : httpd-to [###] 3/5
null_resource.remote-exec (remote-exec): Installing : httpd-to [####] 3/5
null_resource.remote-exec (remote-exec): Installing : httpd-to [#####] 3/5
null_resource.remote-exec (remote-exec): Installing : httpd-to [######] 3/5
null_resource.remote-exec (remote-exec): Installing : httpd-to [########] 3/5
null_resource.remote-exec (remote-exec): Installing : httpd-tools-2.4.6- 3/5
null_resource.remote-exec (remote-exec): Installing : mailcap- [ ] 4/5
null_resource.remote-exec (remote-exec): Installing : mailcap- [#######] 4/5
null_resource.remote-exec (remote-exec): Installing : mailcap- [########] 4/5
null_resource.remote-exec (remote-exec): Installing : mailcap-2.1.41-2.e 4/5
null_resource.remote-exec (remote-exec): Installing : httpd-2. [ ] 5/5
null_resource.remote-exec (remote-exec): Installing : httpd-2. [#] 5/5
null_resource.remote-exec (remote-exec): Installing : httpd-2. [##] 5/5
null_resource.remote-exec (remote-exec): Installing : httpd-2. [###] 5/5
null_resource.remote-exec (remote-exec): Installing : httpd-2. [####] 5/5
null_resource.remote-exec (remote-exec): Installing : httpd-2. [#####] 5/5
null_resource.remote-exec (remote-exec): Installing : httpd-2. [######] 5/5
null_resource.remote-exec (remote-exec): Installing : httpd-2. [#######] 5/5
null_resource.remote-exec (remote-exec): Installing : httpd-2. [########] 5/5
null_resource.remote-exec (remote-exec): Installing : httpd-2.4.6-90.0.1 5/5
null_resource.remote-exec (remote-exec): Verifying : httpd-tools-2.4.6- 1/5
null_resource.remote-exec (remote-exec): Verifying : mailcap-2.1.41-2.e 2/5
null_resource.remote-exec (remote-exec): Verifying : apr-util-1.5.2-6.0 3/5
null_resource.remote-exec (remote-exec): Verifying : httpd-2.4.6-90.0.1 4/5
null_resource.remote-exec (remote-exec): Verifying : apr-1.4.8-5.el7.x8 5/5
null_resource.remote-exec (remote-exec): Installed:
null_resource.remote-exec (remote-exec): httpd.x86_64 0:2.4.6-90.0.1.el7
null_resource.remote-exec (remote-exec): Dependency Installed:
null_resource.remote-exec (remote-exec): apr.x86_64 0:1.4.8-5.el7
null_resource.remote-exec (remote-exec): apr-util.x86_64 0:1.5.2-6.0.1.el7
null_resource.remote-exec (remote-exec): httpd-tools.x86_64 0:2.4.6-90.0.1.el7
null_resource.remote-exec (remote-exec): mailcap.noarch 0:2.1.41-2.el7
null_resource.remote-exec (remote-exec): Complete!
null_resource.remote-exec (remote-exec): Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service.
null_resource.remote-exec (remote-exec): Syntax OK
null_resource.remote-exec (remote-exec): success
null_resource.remote-exec (remote-exec): success
null_resource.remote-exec: Creation complete after 39s [id=1368210293422128851]
Apply complete! Resources: 10 added, 0 changed, 0 destroyed.
Outputs:
namespace = idqasupv5blb
junicode@junicode-XPS-13-9370:~/iac$

Resultados:


Virtual Cloud Networks
console.us-ashburn-1.oraclecloud.com/networking/vcns/ocid1.vcn.oc1.iad.amaaaaaauleez7qauda3xsara32l2gdvmw5jjdmayxn3y2vpkzjkp4gq23zq
ORACLE Cloud
US East (Ashburn)
Networking » Virtual Cloud Networks » Virtual Cloud Network Details
VCN
AVAILABLE
vcn_osce
Move Resource
Add Tags
Terminate
VCN Information
Tags
CIDR Block: 10.0.0.0/16
Compartment: prueba
Created: Thu, Mar 12, 2020, 15:07:54 UTC
OCID: ...gq23zq Show Copy
Default Route Table: defaultRouteTable
DNS Domain Name: vcn1.oraclevcn.com
Resources
Subnets (2)
Route Tables (1)
Internet Gateways (1)
Dynamic Routing Gateways (0)
Network Security Groups (0)
Security Lists (1)
DHCP Options (1)
Local Peering Gateways (0)
NAT Gateways (0)
Service Gateways (0)
Scope
Subnets in prueba Compartment
Create Subnet
Name
State
CIDR Block
Subnet Access
Created
subred-AD1
Available
10.0.1.0/24
Public (lbSb:US-ASHBURN-AD-1)
Thu, Mar 12, 2020, 15:07:56 UTC
subred-AD2
Available
10.0.2.0/24
Public (lbSb:US-ASHBURN-AD-2)
Thu, Mar 12, 2020, 15:07:55 UTC
Showing 2 Items
Page 1
Terms of Use and Privacy
Cookie Preferences
Copyright © 2020, Oracle and/or its affiliates. All rights reserved.


Instances | Oracle Cloud
console.us-ashburn-1.oraclecloud.com/a/compute/instances
ORACLE Cloud
US East (Ashburn)
Compute
Instances
Dedicated Virtual Machine Hosts
Instance Configurations
Instance Pools
Cluster Networks
Autoscaling Configurations
Custom Images
Boot Volumes
Boot Volume Backups
OS Management
List Scope
COMPARTMENT
prueba
osce1 (root)/prueba
Don't see what you're looking for?
Filters
STATE
Running
Instances in prueba Compartment
Create Instance
Sort by:
Created Date (Desc)
Displaying 2 Instances
Page 1
vm1
OCID: ...uhrnba Show Copy
Shape: VM.Standard2.1
Region: iad
Availability Domain: lbSb:US-ASHBURN-AD-1
Fault Domain: FAULT-DOMAIN-3
Created: Thu, 12 Mar 2020 15:07:58 UTC
Maintenance Reboot: -
RUNNING
vm2
OCID: ...ymafoa Show Copy
Shape: VM.Standard2.1
Region: iad
Availability Domain: lbSb:US-ASHBURN-AD-2
Fault Domain: FAULT-DOMAIN-2
Created: Thu, 12 Mar 2020 15:07:57 UTC
Maintenance Reboot: -
RUNNING
Displaying 2 Instances
Page 1
Terms of Use and Privacy
Cookie Preferences
Copyright © 2020, Oracle and/or its affiliates. All rights reserved.



Referencias:

Terraform official web site - https://www.terraform.io/docs/providers/oci/index.html
How to use Terraform on Oracle Cloud Infrastructure (OCI) - https://www.youtube.com/watch?v=GIRjciGRDl4

Autor: Ygnacio Junior Palomino Reyes, Bachiller en Ing. Mecatrónica - Pontificia Universidad Católica del Perú (PUCP), actualmente me desempeño como GenO en Oracle Perú. **Contacto: y.palomino@pucp.pe**