unit_terraform

Homework 1. Terraform.

Подготовительные работы:

1. На Yandex cloud создаем сервисный аккаунт и iam ключ для доступа.
2. На main сервер устанавливаем terraform, для выполнения задания.
3. Т.к будет использоваться ansible, устанавливаем его на main сервер. Редактируем конфиг ansible.cfg на поиск файла hosts из каталога с проектом.
4. Для будущей авторизации на созданной ВМ и доступа по ssh генерируем связку ключей.
5. Прописываем зеркало для возможности использования Yandex облака.

Создание проекта:

1.Начальная структура:
- main.tf – главный файл, с описанием основного алгоритма развертывания и создаваемых ресурсов.
- outputs – файл с описанием переменных, для хранения выводимой информации.
- variables – файл с var переменными.
- gitignore
- плейбуки ansible

2. Описание main.tf

Выбор и подключение к провайдеру Yandex Cloud.

```
terraform {
  required_providers {
    yandex = {
        source = "yandex-cloud/yandex"
    }
  }
  required_version = ">= 0.13"
}

## Авторизация в облаке, параметры берутся из переменных, описанных
## в  variables
provider "yandex" {
  cloud_id = var.cloud_id_var
  folder_id = var.folder_id_var
  service_account_key_file = var.service_account_key_file_var
  zone = "ru-central1-a"


}


#=================================================================

##Создание ресурса внутренней сети

resource "yandex_vpc_subnet" "subnet_terraform" {
  zone           = "ru-central1-a"
  network_id     = yandex_vpc_network.network_terraform.id
  v4_cidr_blocks = ["192.168.15.0/24"]
}
```

```
#====================================================================

## Создание ресурса VM , с указание аппаратных характеристик и
## используемого образа.

      resource "yandex_compute_instance" "vm-test1" {
        name = "test1"

        resources {
          cores  = 2
          memory = 2
        }

        boot_disk {
          initialize_params {
          #      image_id = data.yandex_compute_image.ubuntu_image.id
          image_id = "fd8go38kje4f6v3g2k4q"

          }
        }

## Получение внешнего интерфейса
          network_interface {
          subnet_id = yandex_vpc_subnet.subnet_terraform.id
          security_group_ids =
      [yandex_vpc_security_group.my_webserver.id]
          nat       = true
        }
## Подключение конфига для создаваемого на VM пользователя.

       metadata = {
          user-data = "${file("./meta.yml")}"
        }


## Сам конфиг вынесен в отдельный файл. Будет создан
## пользователь akrokhalev с добавление паблик ключа для доступа.


          #cloud-config
 users:
  - name: akrokhalev
    lock_passwd: true
    groups: sudo
    shell: /bin/bash
    sudo: ['ALL=(ALL) NOPASSWD:ALL']
    ssh-authorized-keys:
      - ${var.my_pubic_key}


## Добавляем  блок по подключению к созданной VM и выполнении на ней
установки пакетов.

provisioner "remote-exec" {
    inline = [
      "sudo apt install -y curl | apt install mc"
    ]
    connection {
      host        = self.network_interface.0.nat_ip_address
      type        = "ssh"
```

```
      user        = "akrokhalev"
      private_key = file(var.private_key_file_path)
    }
  }


## По заданию необходимо установить nginx используя ansible.
## Для этого создадим playbook - create_nginx.yml

- hosts: all
  gather_facts: yes
  become: true
  tasks:
  - name: Install Nginx Web Server
    apt:
      name=nginx
      state=latest

## После, вызовем его через terraform. Т.к для удобства внешний адрес
## созданной VM поместим в хосты ansibl'a.

provisioner "local-exec" {
      when       = create
      on_failure = continue
      command    = "echo ${self.network_interface.0.nat_ip_address} >>
hosts |  ansible-playbook -u akrokhalev -i
'${self.network_interface.0.nat_ip_address},' --private-key
${var.private_key_file_path} create_nginx.yml"
  }
}


#=============================================================================
## Создание ресурса группы безопасности.
## т.к на сервере будет nginx, откроем 443 и 80 порты.
## Для доступа по ssh откроем  22 порт.

resource "yandex_vpc_security_group" "my_webserver" {
  name        = "WebServer security group"
  description = "my_security_group"
  network_id  = yandex_vpc_network.network_terraform.id

  ingress {
    protocol       = "TCP"
    v4_cidr_blocks = ["0.0.0.0/0"]
    port           = 80
  }

  ingress {
    protocol       = "TCP"
    v4_cidr_blocks = ["0.0.0.0/0"]
    port           = 443
  }

  ingress {
    protocol       = "TCP"
    v4_cidr_blocks = ["0.0.0.0/0"]
    port           = 22
  }
  egress {
    protocol       = "ANY"
```

```
    v4_cidr_blocks = ["0.0.0.0/0"]
    port           = -1
  }
}



#На этом создание основного файла закончено.

#=============================================================================
```

3. Описание используемых переменных:
Все переменные выносим в файл variables

```
## ID облака, выдается при создании сервисного аккаунта
variable "cloud_id_var" {
  type    = string
  default = "take_cloud_id"
}

## ID каталога, выдается при создании сервисного аккаунта

variable "folder_id_var" {
  type    = string
  default = "take_folder_id"
}

## IAM ключ, генерируется сервисом облака.

variable "service_account_key_file_var" {
  type    = string
  default = "/home/key/authorized_key.json"
}

## Приватный ключ. Генерируется на main хосте. Хранится на main хосте.
variable "private_key_file_path" {
  type    = string
  default = "~/.ssh/id_rsa"
}

## Паблик ключ, генерируется на main хосте, в связке с приватным.
## Отправляется на создаваемую VM.
variable "my_public_key" {
  type    = string
  default = "take_pub_key"
}
```

6. В outputs выводим только адрес созданной VM.

```
output "webserver_ip" {
  value = yandex_compute_instance.vm-
test1.network_interface.0.nat_ip_address
}
```

Тестирование и запуск.

1. Выполним terraform plan. Смотрим, какие ресурсы создадутся и их конфиги.

```
root@debian:/home/akrokhalev/unit_terraform# terraform plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # yandex_compute_instance.vm-test1 will be created
  + resource "yandex_compute_instance" "vm-test1" {
      + created_at                = (known after apply)
      + folder_id                 = (known after apply)
      + fqdn                      = (known after apply)
      + gpu_cluster_id            = (known after apply)
      + hostname                  = (known after apply)
      + id                        = (known after apply)
      + metadata                  = {
          + "user-data" = <<-EOT
                #cloud-config
                users:
                  - name: akrokhalev
                    lock_passwd: true
                    groups: sudo
                    shell: /bin/bash
                    sudo: ['ALL=(ALL) NOPASSWD:ALL']
```

2. Видим, что создастся 4 ресурса.

```
      + folder_id      = (known after apply)
      + id             = (known after apply)
      + labels         = (known after apply)
      + name           = (known after apply)
      + network_id     = (known after apply)
      + v4_cidr_blocks = [
          + "192.168.15.0/24",
        ]
      + v6_cidr_blocks = (known after apply)
      + zone           = "ru-central1-a"
    }

Plan: 4 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + webserver_ip = (known after apply)

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
```

3. Вызываем terraform apply. Соглашаемся с внесением изменений. Начинается создание ресурсов и подключением к VM.

```
yandex_vpc_network.network_terraform: Creating...
yandex_vpc_network.network_terraform: Creation complete after 3s [id=enpm1uo00ra57hmdrvtu]
yandex_vpc_subnet.subnet_terraform: Creating...
yandex_vpc_security_group.my_webserver: Creating...
yandex_vpc_subnet.subnet_terraform: Creation complete after 1s [id=e9b9caabr4hgkt6dpc7k]
yandex_vpc_security_group.my_webserver: Creation complete after 2s [id=enp9k3j7ap3kfm2o41vv]
yandex_compute_instance.vm-test1: Creating...
yandex_compute_instance.vm-test1: Still creating... [10s elapsed]
yandex_compute_instance.vm-test1: Still creating... [20s elapsed]
yandex_compute_instance.vm-test1: Still creating... [30s elapsed]
yandex_compute_instance.vm-test1: Provisioning with 'remote-exec'...
yandex_compute_instance.vm-test1 (remote-exec): Connecting to remote host via SSH...
yandex_compute_instance.vm-test1 (remote-exec):   Host: 158.160.107.126
yandex_compute_instance.vm-test1 (remote-exec):   User: akrokhalev
yandex_compute_instance.vm-test1 (remote-exec):   Password: false
yandex_compute_instance.vm-test1 (remote-exec):   Private key: true
yandex_compute_instance.vm-test1 (remote-exec):   Certificate: false
yandex_compute_instance.vm-test1 (remote-exec):   SSH Agent: false
yandex_compute_instance.vm-test1 (remote-exec):   Checking Host Key: false
yandex_compute_instance.vm-test1 (remote-exec):   Target Platform: unix
yandex_compute_instance.vm-test1: Still creating... [40s elapsed]
yandex_compute_instance.vm-test1: Still creating... [50s elapsed]
yandex_compute_instance.vm-test1 (remote-exec): Connecting to remote host via SSH...
yandex_compute_instance.vm-test1 (remote-exec):   Host: 158.160.107.126
yandex_compute_instance.vm-test1 (remote-exec):   User: akrokhalev
yandex_compute_instance.vm-test1 (remote-exec):   Password: false
yandex_compute_instance.vm-test1 (remote-exec):   Private key: true
yandex_compute_instance.vm-test1 (remote-exec):   Certificate: false
yandex_compute_instance.vm-test1 (remote-exec):   SSH Agent: false
yandex_compute_instance.vm-test1 (remote-exec):   Checking Host Key: false
yandex_compute_instance.vm-test1 (remote-exec):   Target Platform: unix
yandex_compute_instance.vm-test1: Still creating... [1m0s elapsed]
yandex_compute_instance.vm-test1 (remote-exec): Connecting to remote host via SSH...
yandex_compute_instance.vm-test1 (remote-exec):   Host: 158.160.107.126
yandex_compute_instance.vm-test1 (remote-exec):   User: akrokhalev
yandex_compute_instance.vm-test1 (remote-exec):   Password: false
yandex_compute_instance.vm-test1 (remote-exec):   Private key: true
yandex_compute_instance.vm-test1 (remote-exec):   Certificate: false
yandex_compute_instance.vm-test1 (remote-exec):   SSH Agent: false
yandex_compute_instance.vm-test1 (remote-exec):   Checking Host Key: false
yandex_compute_instance.vm-test1 (remote-exec):   Target Platform: unix
yandex_compute_instance.vm-test1 (remote-exec): Connected!
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 0%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 0%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 0%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 4%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 4%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 6%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 6%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 6%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 6%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 6%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 6%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 37%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 46%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 46%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 53%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 65%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 65%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 65%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 65%
yandex_compute_instance.vm-test1 (remote-exec): Reading package lists... 66%
```

4. После идет вызов скрипта ansible с подставленным адресов VM хоста.

```
yandex_compute_instance.vm-test1 (remote-exec): Building dependency tree... 0%
yandex_compute_instance.vm-test1 (remote-exec): Building dependency tree... 50%
yandex_compute_instance.vm-test1 (remote-exec): Building dependency tree... 50%
yandex_compute_instance.vm-test1 (remote-exec): Building dependency tree... Done
yandex_compute_instance.vm-test1 (remote-exec): Reading state information... 0%
yandex_compute_instance.vm-test1 (remote-exec): Reading state information... 0%
yandex_compute_instance.vm-test1 (remote-exec): Reading state information... Done
yandex_compute_instance.vm-test1 (remote-exec): curl is already the newest version (7.81.0-1ubuntu1.14).
yandex_compute_instance.vm-test1 (remote-exec): 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
yandex_compute_instance.vm-test1: Provisioning with 'local-exec'...
yandex_compute_instance.vm-test1 (local-exec): Executing: ["/bin/sh" "-c" "ansible-playbook -u akrokhalev -i '158.160.107.126' --private-key /home/akrokhalev/.ssh/id_ed25519 create_nginx.yml"]
```

5. В конце видим успешное выполнение. Пробуем подключиться к созданной VM.

```
yandex_compute_instance.vm-test1 (local-exec): PLAY RECAP ***************************************************************

yandex_compute_instance.vm-test1: Creation complete after 1m29s [id=fhm60uo0qkka4i9r23d9]

Apply complete! Resources: 4 added, 0 changed, 0 destroyed.

Outputs:

webserver_ip = "158.160.107.126"
root@debian:/home/akrokhalev/unit_terraform# ssh akrokhalev@158.160.107.126
The authenticity of host '158.160.107.126 (158.160.107.126)' can't be established.
ECDSA key fingerprint is SHA256:mXKSlxj37w6rTsrNU2vfhvUAXWxcJK4OQACXWdRl0uk.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '158.160.107.126' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 22.04.3 LTS (GNU/Linux 5.15.0-88-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Wed Nov 15 08:03:28 AM UTC 2023

  System load:  0.123046875       Processes:             137
  Usage of /:   52.8% of 7.79GB   Users logged in:       0
  Memory usage: 11%               IPv4 address for eth0: 192.168.15.5
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
```

6. Проверяем статус nginx'a.

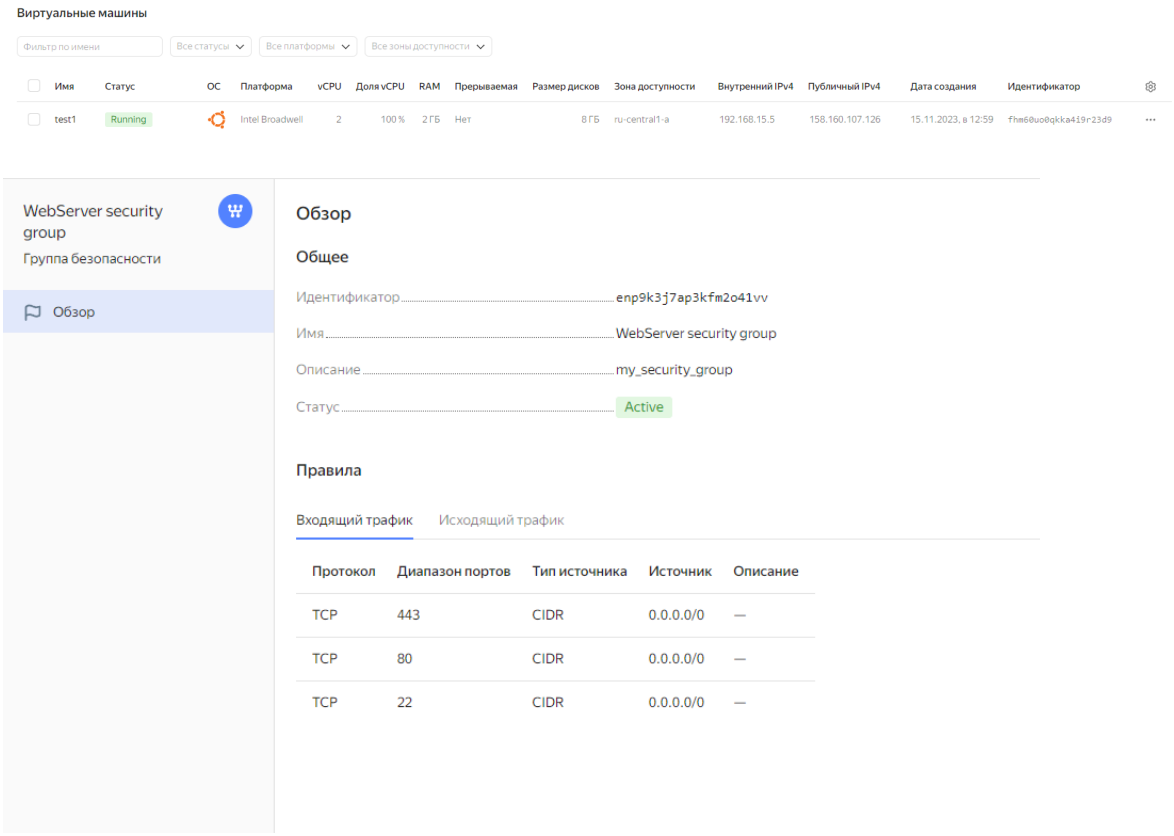
```
root@fhm60uo0qkka4i9r23d9:/home/akrokhalev# systemctl status  nginx
● nginx.service - A high performance web server and a reverse proxy server
     Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2023-11-15 08:04:23 UTC; 19s ago
       Docs: man:nginx(8)
    Process: 1620 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
    Process: 1621 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
   Main PID: 1716 (nginx)
      Tasks: 3 (limit: 2220)
     Memory: 7.1M
        CPU: 36ms
     CGroup: /system.slice/nginx.service
             ├─1716 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             ├─1718 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""
             └─1719 "nginx: worker process" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" "" ""

Nov 15 08:04:23 fhm60uo0qkka4i9r23d9 systemd[1]: Starting A high performance web server and a reverse proxy server...
Nov 15 08:04:23 fhm60uo0qkka4i9r23d9 systemd[1]: Started A high performance web server and a reverse proxy server.
root@fhm60uo0qkka4i9r23d9:/home/akrokhalev# exit
```

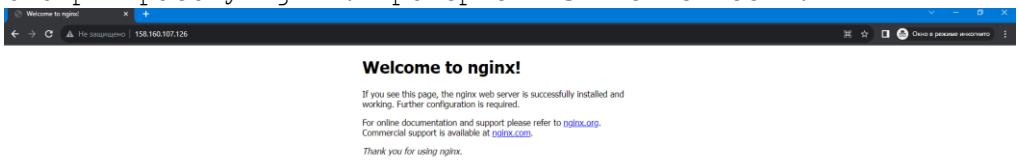
7. Проверяем доступность портов.

```
akrokhalev@fhm60uo0qkka4i9r23d9:~$ telnet 158.160.107.126 80
Trying 158.160.107.126...
Connected to 158.160.107.126.
Escape character is '^]'.
^ZConnection closed by foreign host.
```

8. Проверяем созданную VM через вэб интерфейс Yandex’a.



9. Смотрим работу Nginx. Проверяем из внешней сети.



10. Удаляем всю инфраструктуру через terraform destroy.

```
  Enter a value: yes

yandex_compute_instance.vm-test1: Destroying... [id=fhm60uo0qkka4i9r23d9]
yandex_compute_instance.vm-test1: Still destroying... [id=fhm60uo0qkka4i9r23d9, 10s elapsed]
yandex_compute_instance.vm-test1: Still destroying... [id=fhm60uo0qkka4i9r23d9, 20s elapsed]
yandex_compute_instance.vm-test1: Still destroying... [id=fhm60uo0qkka4i9r23d9, 30s elapsed]
yandex_compute_instance.vm-test1: Still destroying... [id=fhm60uo0qkka4i9r23d9, 40s elapsed]
yandex_compute_instance.vm-test1: Destruction complete after 47s
yandex_vpc_subnet.subnet_terraform: Destroying... [id=e9b9caabr4hgkt6dpc7k]
yandex_vpc_security_group.my_webserver: Destroying... [id=enp9k3j7ap3kfm2o41vv]
yandex_vpc_subnet.subnet_terraform: Destruction complete after 3s
yandex_vpc_security_group.my_webserver: Destruction complete after 3s
yandex_vpc_network.network_terraform: Destroying... [id=enpm1uo00ra57hmdrvtu]
yandex_vpc_network.network_terraform: Destruction complete after 3s

Destroy complete! Resources: 4 destroyed.
```