# 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"

}

1. В outputs выводим только адрес созданной VM.

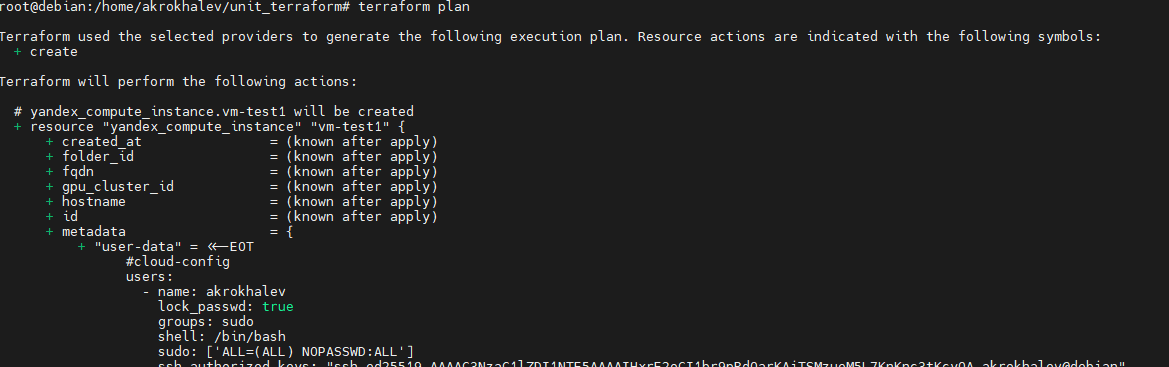
output "webserver\_ip" {

value = yandex\_compute\_instance.vm-test1.network\_interface.0.nat\_ip\_address

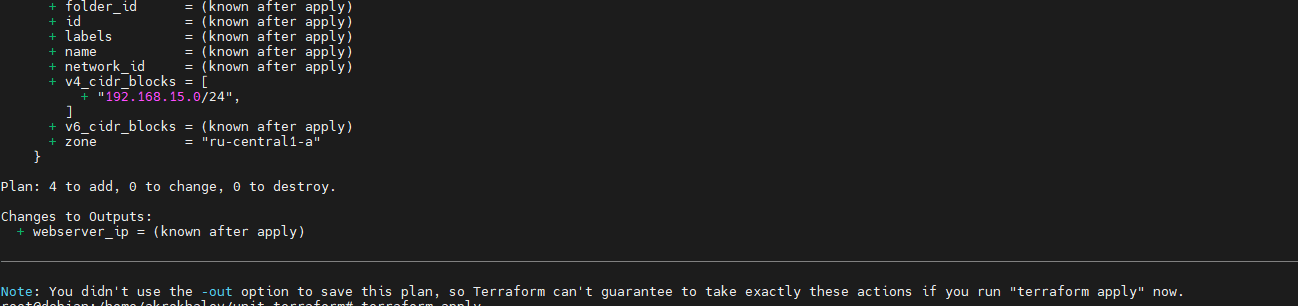
}

Тестирование и запуск.

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

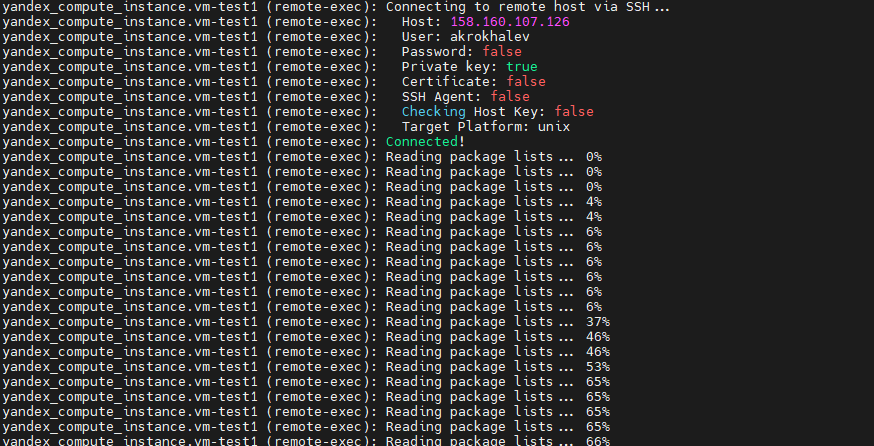
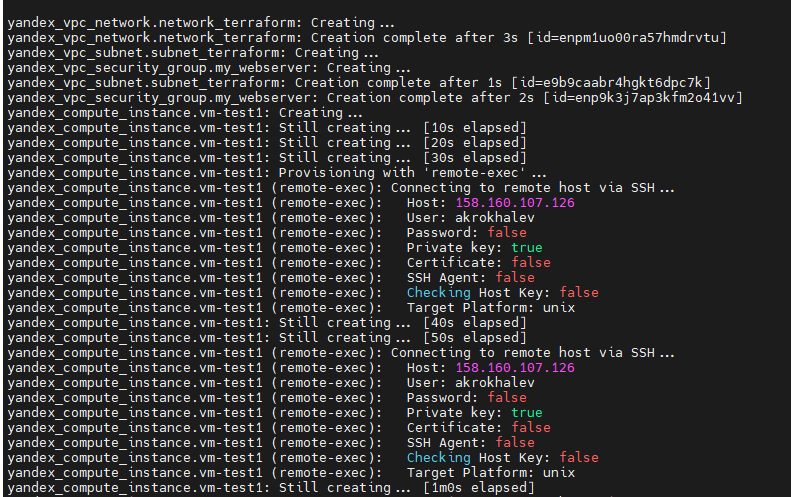


1. Видим, что создастся 4 ресурса.

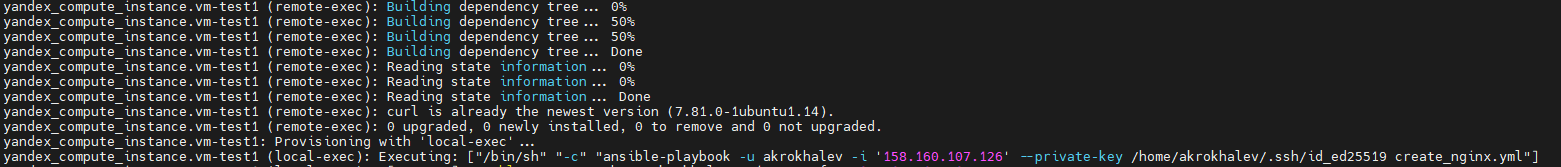


1. Вызываем terraform apply. Соглашаемся с внесением изменений.

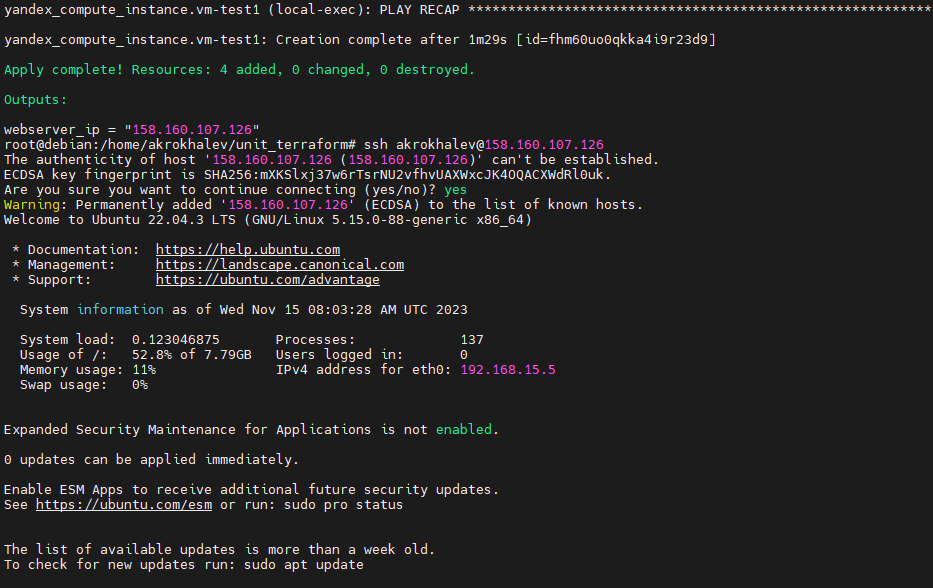
Начинается создание ресурсов и подключением к VM.



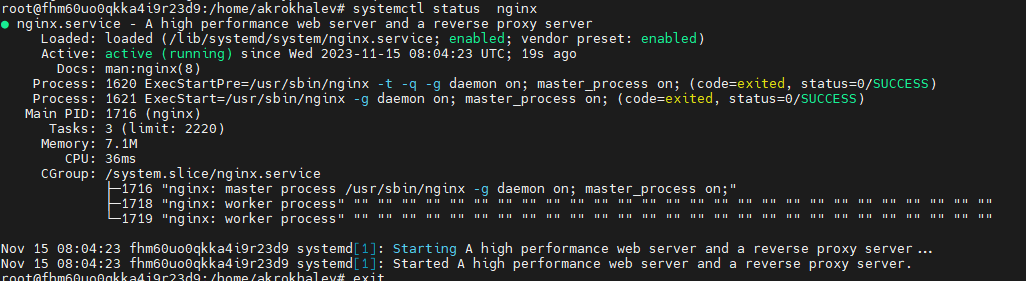
1. После идет вызов скрипта ansible с подставленным адресов VM хоста.



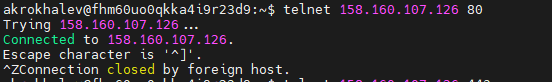
1. В конце видим успешное выполнение. Пробуем подключиться к созданной VM.



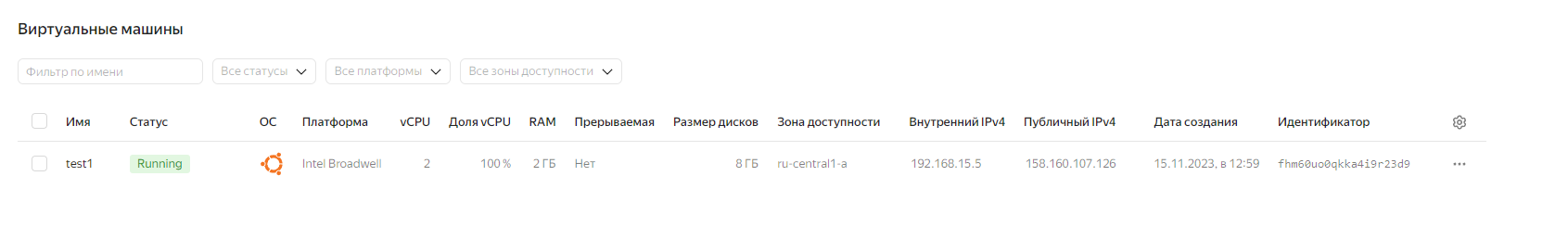
1. Проверяем статус nginx’a.

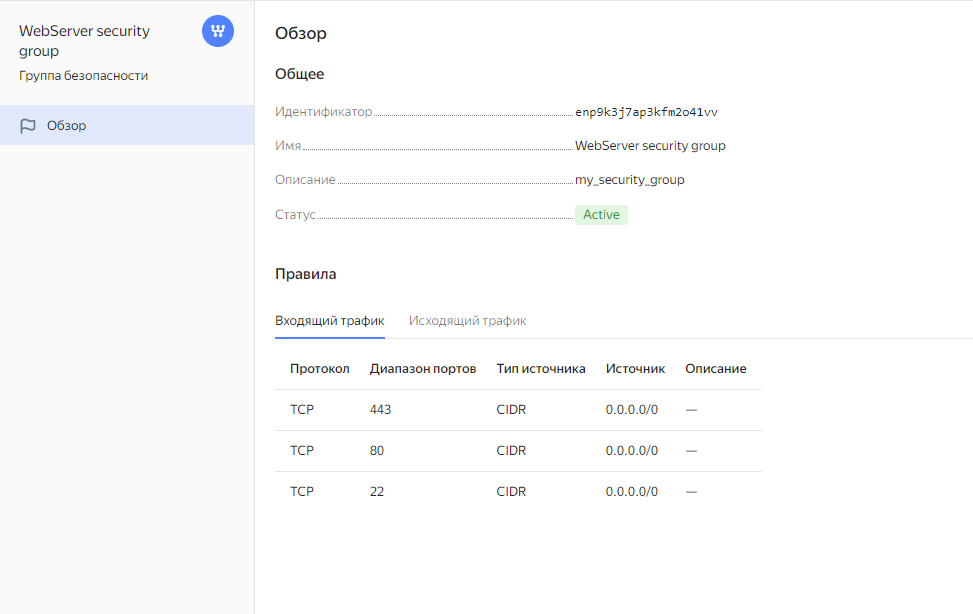


1. Проверяем доступность портов.

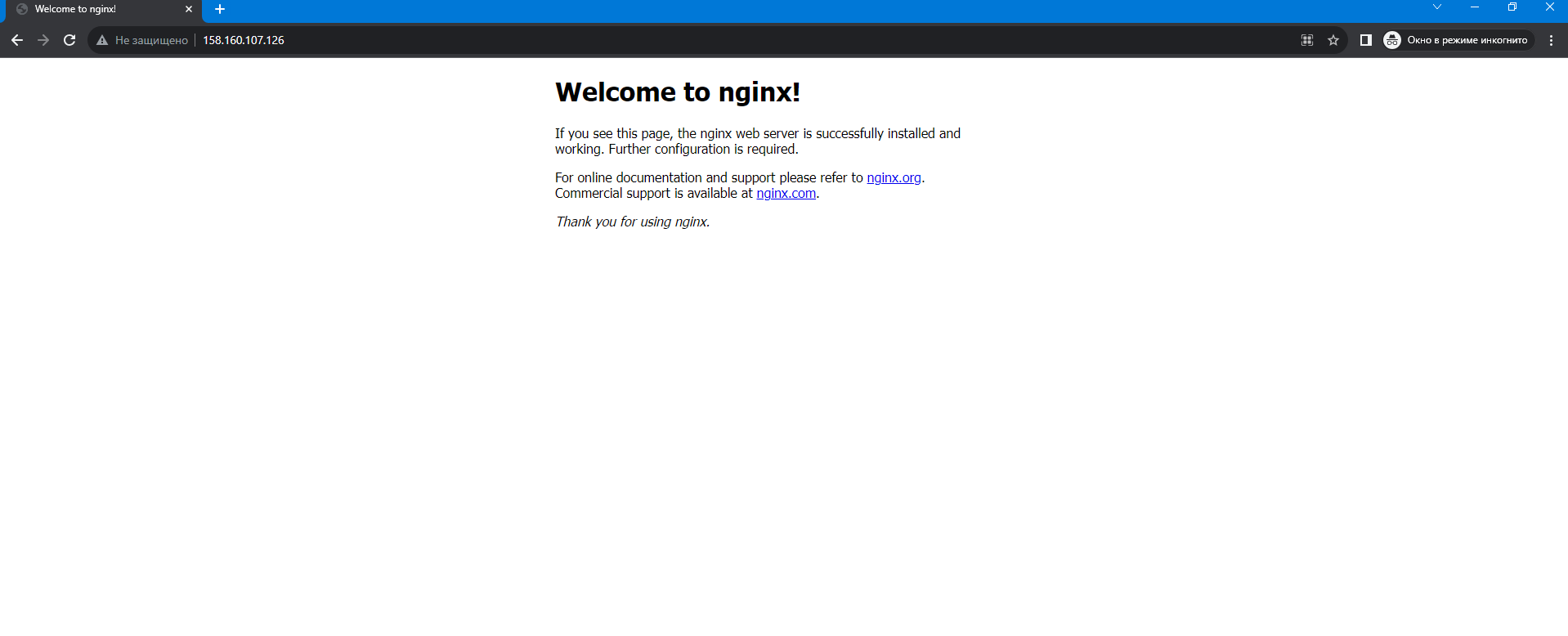


1. Проверяем созданную VM через вэб интерфейс Yandex’a.





1. Смотрим работу Nginx. Проверяем из внешней сети.



1. Удаляем всю инфраструктуру через terraform destroy.

