**ПОДГОТОВИТЕЛЬНАЯ ЧАСТЬ**

**Проверяю обновления**

sudo apt update

sudo apt upgrade

sudo apt full-upgrade

**Устанавливаю ANSIBLE**

python --version

python3 --version

update-alternatives --install /usr/bin/python python /usr/bin/python3 2

sudo apt install git python3-pip

pip3 install yml

pip3 install ansible

ansible –version

Создаю файл ansible.cfg

mkdir ansible

cd ansible

ansible-config init --disabled > ansible.cfg *– генерация полностью закомментированного файла*

ansible-config init --disabled -t all > ansible.cfg *– генерация полностью закоммент. файла с существующими плагинами*

Устанавливаю TERRAFORM

mkdir terraform\_setup

cd terraform\_setup

wget https://hashicorp-releases.yandexcloud.net/terraform/1.6.2/terraform\_1.6.2\_linux\_amd64.zip

zcat terraform\_1.6.2\_linux\_amd64.zip > terraform

file terraform

chmod 766 terraform

./terraform -v

cp terraform /usr/local/bin/

terraform -v

cd

Настраиваю TERRAFORM

nano ~/.terraformrc

|  |
| --- |
| provider\_installation {  network\_mirror {  url = "https://terraform-mirror.yandexcloud.net/"  include = ["registry.terraform.io/\*/\*"]  }  direct {  exclude = ["registry.terraform.io/\*/\*"]  }  } |

mkdir terraform

cd terraform

nano main.tf

|  |
| --- |
| terraform {  required\_providers {  yandex = {  source = "yandex-cloud/yandex"  }  }  required\_version = ">= 0.13"  }  provider "yandex" {  token = "y0\_AgAAAAATsxVZAATuwQAAAADinqT3jXuJN8aPSkGeJzd-4gZxKkGjwZ4"  cloud\_id = "b1g09ilem72537mh3ies"  folder\_id = "b1gv1df2r3960gt3bb4j"  } |

terraform providers lock -net-mirror=https://terraform-mirror.yandexcloud.net -platform=linux\_amd64 -platform=darwin\_arm64 yandex-cloud/yandex

terraform init

cd

Устанавливаю nginx (https://nginx.org/ru/linux\_packages.html#Debian)

sudo apt install curl gnupg2 ca-certificates lsb-release debian-archive-keyring

curl https://nginx.org/keys/nginx\_signing.key | gpg --dearmor \

| sudo tee /usr/share/keyrings/nginx-archive-keyring.gpg >/dev/null

Проверка, верный ли ключ был загружен:

gpg --dry-run --quiet --no-keyring --import --import-options import-show /usr/share/keyrings/nginx-archive-keyring.gpg

|  |
| --- |
| Вывод команды должен содержать полный отпечаток ключа 573BFD6B3D8FBC641079A6ABABF5BD827BD9BF62:  pub rsa2048 2011-08-19 [SC] [expires: 2024-06-14]  573BFD6B3D8FBC641079A6ABABF5BD827BD9BF62  uid nginx signing key <signing-key@nginx.com> |

echo "deb [signed-by=/usr/share/keyrings/nginx-archive-keyring.gpg] \

http://nginx.org/packages/debian `lsb\_release -cs` nginx" \

| sudo tee /etc/apt/sources.list.d/nginx.list

Чтобы установить nginx, выполните следующие команды:

sudo apt update

sudo apt upgrade

sudo apt install nginx

Создание SSH ключа

ssh-keygen -t rsa

|  |
| --- |
| Created directory '/root/.ssh'.  Your identification has been saved in /root/.ssh/id\_rsa  Your public key has been saved in /root/.ssh/id\_rsa.pub |

cat ~/.ssh/id\_rsa.pub

Мои ключи

ssh-keygen -t rsa

/root/ansible/.ssh/mykey

|  |
| --- |
| root@mikhail:~# ssh-keygen  Generating public/private rsa key pair.  Enter file in which to save the key (/root/.ssh/id\_rsa): /root/ansible/.ssh/mykey  Enter passphrase (empty for no passphrase):  Enter same passphrase again:  Your identification has been saved in /root/ansible/.ssh/mykey  Your public key has been saved in /root/ansible/.ssh/mykey.pub  The key fingerprint is:  SHA256:yCEjBE9TV3y8YKwSJMSdIhDwnl2ohqul9uvSNJ9avLc root@mikhail |

cat /root/ansible/.ssh/mykey.pub

cd /root/ansible/.ssh

chmod 0400 mykey\*

cd

**ОСНОВНАЯ ЧАСТЬ**

**САЙТ**

Для удобства поиска (редактирования) ошибок в создаваемых конфигурационных файлах буду стараться делать их под каждое решение отдельно!

cd terraform

nano meta.txt

|  |
| --- |
| #cloud-config  users:  - name: mikhail  groups: sudo  shell: /bin/bash  sudo: ['ALL=(ALL) NOPASSWD:ALL']  ssh-authorized-keys:  - ssh-rsa  root@mikhail |

terraform validate

terraform plan

Устанавливаю две ВМ в разных зонах

nano website1.tf

|  |
| --- |
| resource "yandex\_compute\_instance" "website-a" {  name = "website1"  allow\_stopping\_for\_update = true  platform\_id = "standard-v3"  zone = "ru-central1-a"  resources {  cores = 2  memory = 1  core\_fraction = 20  }  boot\_disk {  initialize\_params {  image\_id = "fd87gocdmk3tosg6onpg"  size = 4  description = "boot disk for nginx\_server1"  }  }  network\_interface {  subnet\_id = "${yandex\_vpc\_subnet.subnet-a.id}"  nat = true  }  metadata = {  user-data = "${file("~/terraform/meta.txt")}"  }  scheduling\_policy {  preemptible = true  }  }  resource "yandex\_vpc\_network" "network-a" {  name = "network1"  }  resource "yandex\_vpc\_subnet" "subnet-a" {  name = "subnet1"  zone = "ru-central1-a"  v4\_cidr\_blocks = ["192.168.10.0/24"]  network\_id = "${yandex\_vpc\_network.network-a.id}"  } |

terraform validate

terraform plan

nano website2.tf

|  |
| --- |
| resource "yandex\_compute\_instance" "website-b" {  name = "website2"  allow\_stopping\_for\_update = true  platform\_id = "standard-v3"  zone = "ru-central1-b"  resources {  cores = 2  memory = 1  core\_fraction = 20  }  boot\_disk {  initialize\_params {  image\_id = "fd87gocdmk3tosg6onpg"  size = 4  description = "boot disk for nginx\_server1"  }  }  network\_interface {  subnet\_id = "${yandex\_vpc\_subnet.subnet-b.id}"  nat = true  }  metadata = {  user-data = "${file("~/terraform/meta.txt")}"  }  scheduling\_policy {  preemptible = true  }  }  resource "yandex\_vpc\_subnet" "subnet-b" {  name = "subnet2"  zone = "ru-central1-b"  v4\_cidr\_blocks = ["192.168.50.0/24"]  network\_id = "${yandex\_vpc\_network.network-a.id}"  } |

terraform validate

terraform plan

nano target\_backend\_group.tf

|  |
| --- |
| resource "yandex\_alb\_target\_group" "target-group" {  name = "nginxweb-target-group"  description = "ALB:Target group"  target {  subnet\_id = yandex\_vpc\_subnet.subnet-a.id  ip\_address = yandex\_compute\_instance.website-a.network\_interface.0.ip\_address  }  target {  subnet\_id = yandex\_vpc\_subnet.subnet-b.id  ip\_address = yandex\_compute\_instance.website-b.network\_interface.0.ip\_address  }  }  resource "yandex\_alb\_http\_router" "http-router" {  name = "nginxweb-http-router"  description = "ALB:HTTP router"  }  resource "yandex\_alb\_load\_balancer" "target-backend-group" {  name = "nginxweb-target-backend-group"  description = "ALB:Target backend group"  network\_id = "${yandex\_vpc\_network.network-a.id}"  allocation\_policy {  location {  zone\_id = "ru-central1-a"  subnet\_id = "${yandex\_vpc\_subnet.subnet-a.id}"  }  }  listener {  name = "listener"  endpoint {  address {  external\_ipv4\_address {  }  }  ports = [ 80 ]  }  http {  handler {  http\_router\_id = "${yandex\_alb\_http\_router.http-router.id}"  }  }  }  } |

terraform validate

terraform plan

terraform apply

cd

**удалить созданные ресурсы**

**terraform destroy**

Настраиваю ANSIBLE

cd ansible

nano hosts.txt

|  |
| --- |
| [website]  website1 ansible\_host=158.160.127.59 ansible\_user=mikhail ansible\_ssh\_private\_key\_file=/root/ansible/.ssh/mykey  website2 ansible\_host=51.250.110.251 ansible\_user=mikhail ansible\_ssh\_private\_key\_file=/root/ansible/.ssh/mykey |

ansible -i hosts.txt website -m ping

nano ansible.cfg

|  |
| --- |
| [defaults]  host\_key\_checking = false  inventory = ./hosts.txt |

ansible all -m ping

ansible all --list-hosts

ansible-inventory --list

ansible-inventory --graph

ansible all -m setup

ansible -m ping all -u ansible -k

ansible all -m shell -a "uptime"

ansible all -m shell -a "ls /etc"

echo privet > privet.txt *– создаю файл для теста*

ansible all -m copy -a "src=privet.txt dest=/home" -b *– пример записи файла на управляемые машины*

ansible all -m shell -a "ls -la /home" *– проверяю как прошла записи файла на управляемых машинах*

ansible all -m file -a "path=/home/privet.txt state=absent" -b *– удаляю записанный файл на управляемых машинах*

ansible all -m get\_url -a "url=https://hashicorp-releases.yandexcloud.net/terraform/1.6.2/terraform\_1.6.2\_linux\_amd64.zip dest=/home" -b *– пример записи файла из интернета на управляемые машины*

Создаю playbook для установки nginx на ВМ

nano 001\_nginx\_playbook.yml

|  |
| --- |
| # МОЙ ПЕРВЫЙ РАБОТАЮЩИЙ PLAYBOOK  ---  - hosts: website  become: true  tasks:  - name: update  apt: update\_cache=yes  - name: Install Nginx  apt: name=nginx state=latest  notify:  - restart nginx  handlers:  - name: restart nginx  service: name=nginx state=reloaded |

ansible-playbook 001\_nginx\_playbook.yml

curl -v 158.160.127.59:80

curl -v 51.250.110.251:80

МОНИТОРИНГ

Создаю ВМ

nano zabbix1.tf

|  |
| --- |
| resource "yandex\_compute\_instance" "zabbix-1" {  name = "zabbix1"  allow\_stopping\_for\_update = true  platform\_id = "standard-v3"  zone = "ru-central1-a"  resources {  cores = 2  memory = 1  core\_fraction = 20  }  boot\_disk {  initialize\_params {  image\_id = "fd87gocdmk3tosg6onpg"  size = 4  description = "boot disk for zabbix-1"  }  }  network\_interface {  subnet\_id = "${yandex\_vpc\_subnet.subnet-a.id}"  nat = true  }  metadata = {  user-data = "${file("~/terraform/meta.txt")}"  }  scheduling\_policy {  preemptible = true  }  } |

terraform validate

terraform plan

terraform apply

Подключаюсь к вновь созданной ВМ и устанавливаю на ней Zabbix *(Как это делали на занятиях)*

На ВМ из первого задания устанавливаю Zabbix Agent

Настраиваю агенты на отправление метрик в Zabbix