**Задача 1**

Проверка Virtualbox:

virtualbox --help

Oracle VM VirtualBox VM Selector v7.0.18

Проверка Vagrant:

vagrant -v

Vagrant 2.4.1

Проверка Packer:

packer -v

Packer v1.11.0

Установка, проверка и инициализация yandex cloud cli:

curl -sSL https://storage.yandexcloud.net/yandexcloud-yc/install.sh | bash

yc -v

Yandex Cloud CLI 0.127.0 linux/amd64

yc init

Your current cloud has been set to 'cloud-neto'

Your current folder has been set to 'default'

Your profile default Compute zone has been set to 'ru-central1-a'

**Задача 2**

Скачиваем box-образ ubuntu 20.04:

wget -O ~/ubuntu2004-vagrant.box https://app.vagrantup.com/bento/boxes/ubuntu-20.04/versions/202404.23.0/providers/virtualbox/amd64/vagrant.box

Сохранение в: «~/ubuntu2004-vagrant.box»

~/ubuntu2004-vagrant.box 7%[==> ] 51,12M 2,98MB/s ост 3m 39s

Добавляем скачанный box в vagrant:

vagrant box add bento/ubuntu-20.04 ~/ubuntu2004-vagrant.box

==> box: Box file was not detected as metadata. Adding it directly...

==> box: Adding box 'bento/ubuntu-20.04' (v0) for provider:

box: Unpacking necessary files from: file:///home/udjin/ubuntu2004-vagrant.box

==> box: Successfully added box 'bento/ubuntu-20.04' (v0) for ''!

Переходим в каталог с Vagrantfile и выполняем команду:

vagrant up

Bringing machine 'server1.netology' up with 'virtualbox' provider...

Stderr: VBoxManage: error: VT-x is not available (VERR\_VMX\_NO\_VMX)

Забыли включить вложенную виртуализацию в vmware workstation, исправляемся и перезапускаем.

Bringing machine 'server1.netology' up with 'virtualbox' provider...

==> server1.netology: Clearing any previously set forwarded ports...

==> server1.netology: Clearing any previously set network interfaces...

==> server1.netology: Preparing network interfaces based on configuration...

server1.netology: Adapter 1: nat

server1.netology: Adapter 2: hostonly

==> server1.netology: Forwarding ports...

server1.netology: 22 (guest) => 20011 (host) (adapter 1)

server1.netology: 22 (guest) => 2222 (host) (adapter 1)

==> server1.netology: Running 'pre-boot' VM customizations...

==> server1.netology: Booting VM...

==> server1.netology: Waiting for machine to boot. This may take a few minutes...

server1.netology: SSH address: 127.0.0.1:2222

server1.netology: SSH username: vagrant

server1.netology: SSH auth method: private key

server1.netology: Warning: Connection reset. Retrying...

server1.netology:

server1.netology: Vagrant insecure key detected. Vagrant will automatically replace

server1.netology: this with a newly generated keypair for better security.

server1.netology:

server1.netology: Inserting generated public key within guest...

server1.netology: Removing insecure key from the guest if it's present...

server1.netology: Key inserted! Disconnecting and reconnecting using new SSH key...

==> server1.netology: Machine booted and ready!

==> server1.netology: Checking for guest additions in VM...

==> server1.netology: Setting hostname...

==> server1.netology: Configuring and enabling network interfaces...

==> server1.netology: Mounting shared folders...

server1.netology: /vagrant => /home/udjin/Документы/REPOS/virtd-homeworks/05-virt-02-iaac/src

==> server1.netology: Running provisioner: shell...

server1.netology: Running: inline script

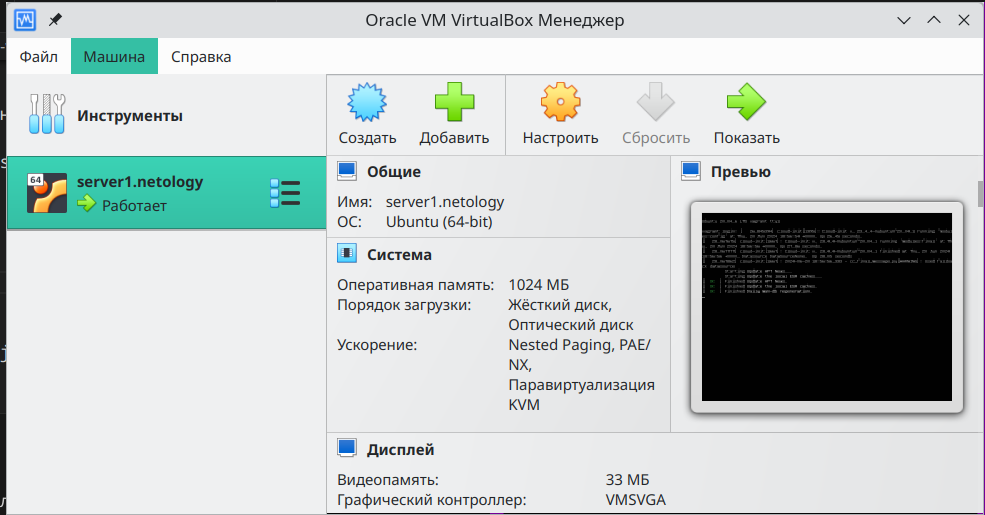
Убеждаемся что виртуальная машина создана:

vagrant status

Current machine states:

server1.netology running (virtualbox)

Можно открыть интерфейс virtualbox:



Проверим наличие docker:

vagrant ssh

Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-176-generic x86\_64)

docker version

Client: Docker Engine - Community

Version: 26.1.4

API version: 1.45

Go version: go1.21.11

Git commit: 5650f9b

Built: Wed Jun 5 11:29:19 2024

OS/Arch: linux/amd64

Context: default

гасим ВМ:

vagrant destroy

server1.netology: Are you sure you want to destroy the 'server1.netology' VM? [y/N]

==> server1.netology: Forcing shutdown of VM...

==> server1.netology: Destroying VM and associated drives...

**Задача 3**

Вносим в файл mydebian.json.pkr.hcl необходимые авторизационные данные к yandex cloud. Устанавливаем плагин Яндекс:

packer init config.pkr.hcl

Installed plugin github.com/hashicorp/yandex v1.1.3 in "/home/udjin/.config/packer/plugins/github.com/hashicorp/yandex/packer-plugin-yandex\_v1.1.3\_x5.0\_linux\_amd64"

Копируем скрипт установки docker из Vagrantfile в файл mydebian.json.pkr.hcl в блок provisioner с помощью heredoc (EOF):

provisioner "shell" {

inline = [ <<EOF

export DEBIAN\_FRONTEND=noninteractive

# Add Docker's official GPG key:

sudo apt-get update

sudo apt-get install ca-certificates curl gnupg

sudo install -m 0755 -d /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg

sudo chmod a+r /etc/apt/keyrings/docker.gpg

# Add the repository to Apt sources:

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \

$(. /etc/os-release && echo "$VERSION\_CODENAME") stable" | \

sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt-get update

sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

EOF

]

}

Запускаем сборку:

packer build mydebian.json.pkr.hcl

yandex.debian\_docker: output will be in this color.

==> yandex.debian\_docker: Creating temporary RSA SSH key for instance...

==> yandex.debian\_docker: Using as source image: fd8e1hrq5vjet0qtcikv (name: "debian-11-v20240617", family: "debian-11")

==> yandex.debian\_docker: Use provided subnet id e9bv1hf6tojiasrg3k2b

==> yandex.debian\_docker: Creating disk...

==> yandex.debian\_docker: Creating instance...

==> yandex.debian\_docker: Waiting for instance with id fhm8n43j1d953atgol50 to become active...

yandex.debian\_docker: Detected instance IP: 84.201.134.110

==> yandex.debian\_docker: Using SSH communicator to connect: 84.201.134.110

==> yandex.debian\_docker: Waiting for SSH to become available...

==> yandex.debian\_docker: Connected to SSH!

==> yandex.debian\_docker: Provisioning with shell script: /tmp/packer-shell3943696921

yandex.debian\_docker: hello from packer

==> yandex.debian\_docker: Stopping instance...

==> yandex.debian\_docker: Deleting instance...

yandex.debian\_docker: Instance has been deleted!

==> yandex.debian\_docker: Creating image: debian-11-docker

==> yandex.debian\_docker: Waiting for image to complete...

==> yandex.debian\_docker: Success image create...

==> yandex.debian\_docker: Destroying boot disk...

yandex.debian\_docker: Disk has been deleted!

Build 'yandex.debian\_docker' finished after 2 minutes 8 seconds.

==> Wait completed after 2 minutes 8 seconds

==> Builds finished. The artifacts of successful builds are:

--> yandex.debian\_docker: A disk image was created: debian-11-docker

Проверяем наличие нового образа:

yc compute image list

+----------------------+------------------+--------+----------------------+--------+

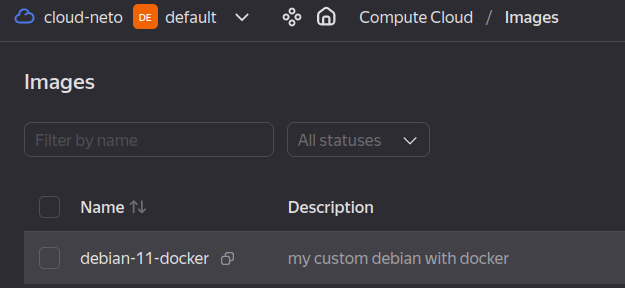
| ID | NAME | FAMILY | PRODUCT IDS | STATUS |

+----------------------+------------------+--------+----------------------+--------+

| fd8cruege75cljh4c2kk | debian-11-docker | | f2epoc18trn9lumt5fbd | READY |

+----------------------+------------------+--------+----------------------+--------+

Можно убедиться в web-консоли:



Удаляем более ненужный образ:

yc compute image delete fd8cruege75cljh4c2kk

done (20s)

yc compute image list

+----+------+--------+-------------+--------+

| ID | NAME | FAMILY | PRODUCT IDS | STATUS |

+----+------+--------+-------------+--------+

+----+------+--------+-------------+--------+

И удаляем token из файла mydebian.json.pkr.hcl