

20

Методическое пособие по выполнению домашнего задания по курсу «Администратор Linux. Professional»

**Настройка ПК для создания стендов и выполнения домашних заданий**

**Цель домашнего задания**

Настроить среду для выполнения домашних работ

**Описание домашнего задания**

Подготовка ПК для выполнения домашних работ

**Введение**

В этом методическом указании рассмотрены основные инструкции по установке программ, которые понадобятся для выполнения домашних работ. Будет рассмотрена установка среды виртуализации VirtualBox, ПО для настройки виртуальной среды на базе VirtualBox и Hashicorp Vagrant, среда автоматического конфигурирования – Ansible и дополнительные программы, которые могут потребоваться для выполнения домашних работ.

**Функциональные и нефункциональные требования**

* ПК на Unix c 8ГБ ОЗУ или виртуальная машина с включенной Nested Virtualization.

**Инструкция по подготовке Вашего рабочего места**

Перед настройкой вашего ПК, нужно понять, какой операционной системой вы пользуетесь. Для выполнения домашних работ требуется ПК на Unix (MacOS или Linux-система).

Если у Вас есть только компьютер на Windows, тогда Вам потребуется настроить виртуальную машину и включить на ней функцию Nested Virtualization.

*При работе через виртуальную машину могут возникать проблемы*

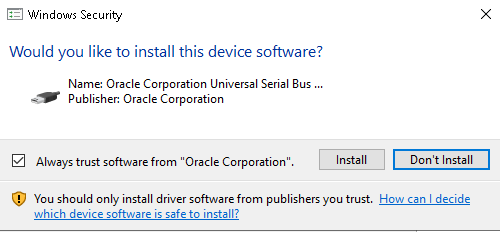
**Настройка виртуальной машины с Nested Virtualization на Windows**

1. Заходим на сайт Oracle ([Downloads – Oracle VM VirtualBox](https://www.virtualbox.org/wiki/Downloads))

и скачиваем последнюю версию VirtualBox

1. После скачивания программы, запускаем .exe-файл и следуем инструкциям установщика.

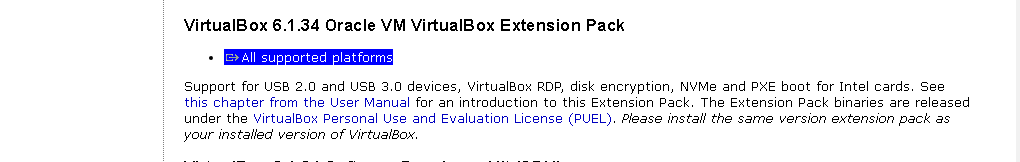
При запросе установки **Universal Serial Bus Adapter** нажмите **Install**.



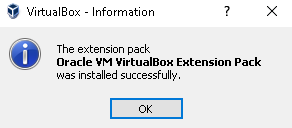
После успешной установки на рабочем столе появится значок **Oracle VM VirtualBox**, при его открытии должна запускаться (без ошибок) VirtualBox



3. Дополнительно, на сайте Oracle (<https://www.virtualbox.org/wiki/Downloads>) скачиваем VM VirtualBox Extension Pack

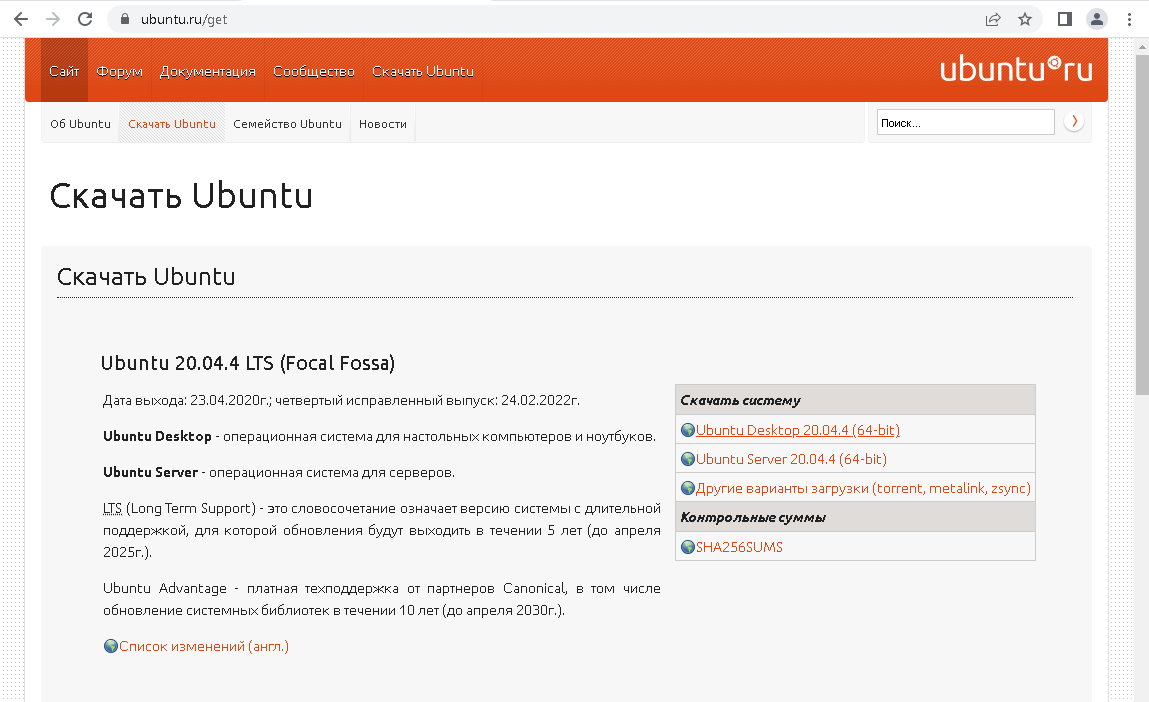


После скачивания также запускаем файл установки и следуюем инструкциям, после успешной установки должно появится следующее окно

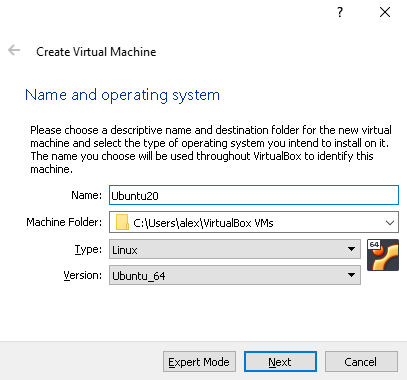
Теперь VirtualBox готова для установки виртуальной Linux-машины.

4. Скачиваем образ desktop-версии Ubuntu 20.04 с сайта(<https://ubuntu.ru/get>)

*Примечание: Вы можете использовать любой современный и понравившийся Вам Debian-based или RHEL-based дистрибутив. Ubuntu 20.04 рассматривается для примера.* 

5. Создаем виртуальную машину

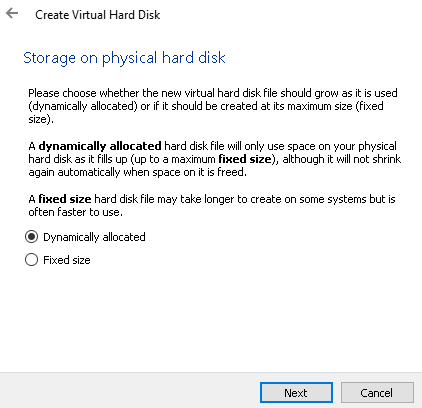
* Открываем **VirtualBox** — **new** — *откроется окно создания виртуальной машины*



* В окне создания виртуальной машины указваем **имя виртуальной машины**, **папку в которой будут храниться файлы виртуальной машины, тип ОС, версию** и нажимаем **Next.**
* В следующем окне указываем **количество оперативной памяти** для ВМ. Укажите 8192MB или больше и нажмите **Next.**
* В следующем окне создаём новый виртуальный жесткий диск, выбираем пункт **Create a virtual hard disk now** и нажимаем **Create**
* Откроется окно создания виртуального диска:

Тип диска выбираем **VDI** и нажимаем **Next**

* Откроется окно выбора принципа храния данных на диске

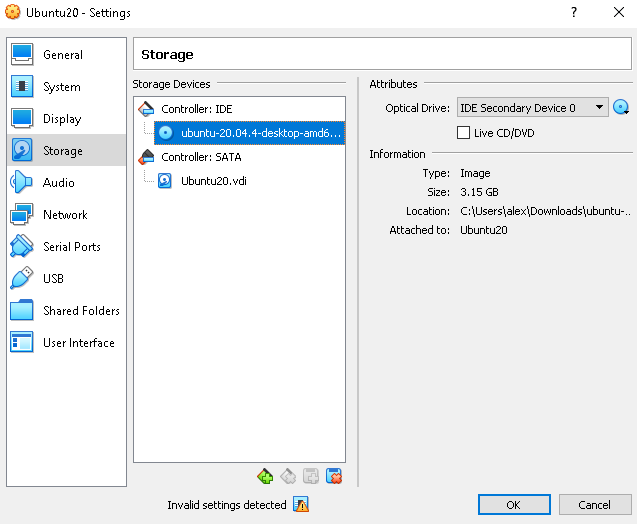
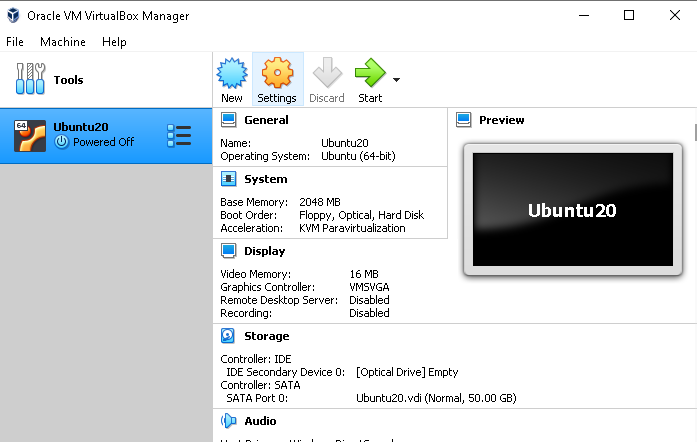
Тут выбираем **Dynamically allocated** и нажимаем **Next*.*** *Dynamically allocated означает, что место в вашей ОС будет занято по мере накопления информации на виртуальном жётском диске.* 

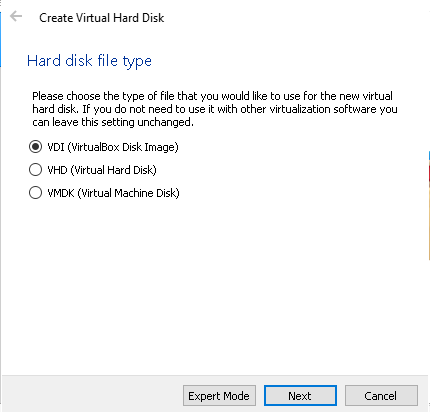
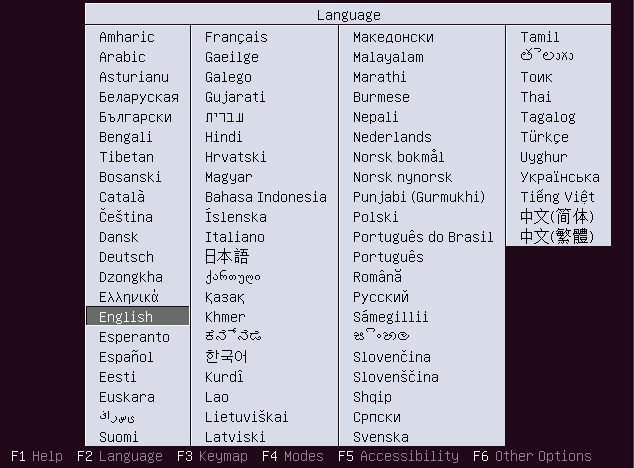
* *Указываем* ***размер виртуального жёсткого диска****. Рекомендуется указать* ***80 Гбайт или больше****. После указания размера нажимаем* ***Create****.*
* После создания виртуального жёсткого диска, виртуальная машина появится в списке виртуальных машин
* Нажмём на кнопку **Settings** и внесем дополнительные настройки:
  + **System** — **Processor** — *минимально надо указать 2 ядра, лучше указать 4.*
  + **System — Processor — Extended Features — Enable Nested VT-x/AMD-V**

*Если данный параметр будет неактивен, то в настройках BIOS нужно найти параметр, который включает технологии вложенной виртуализации. У всех производителей материнских плат эта технология может называться по-разному.*

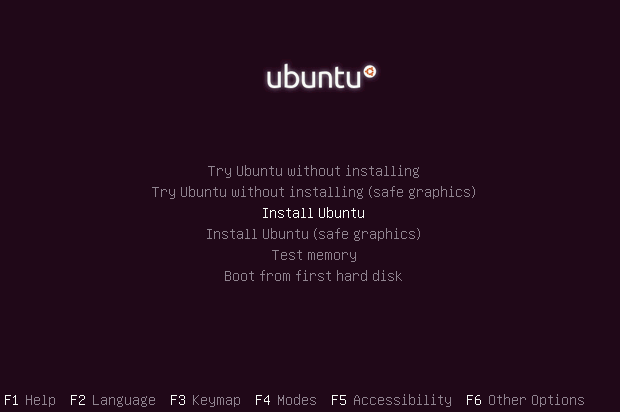
* + ***Storage — Controller:IDE*** *— щелкнем на значок диска и выберем параметр* ***choose a disk file*** *— укажем iso-файл с Ubuntu 20.04*

После внесения настроек, нажмём **OK** и запустим (**Start**) виртуальную машину. Начнётся процесс установки ОС:

* Выбираем язык и нажимаем **Enter**

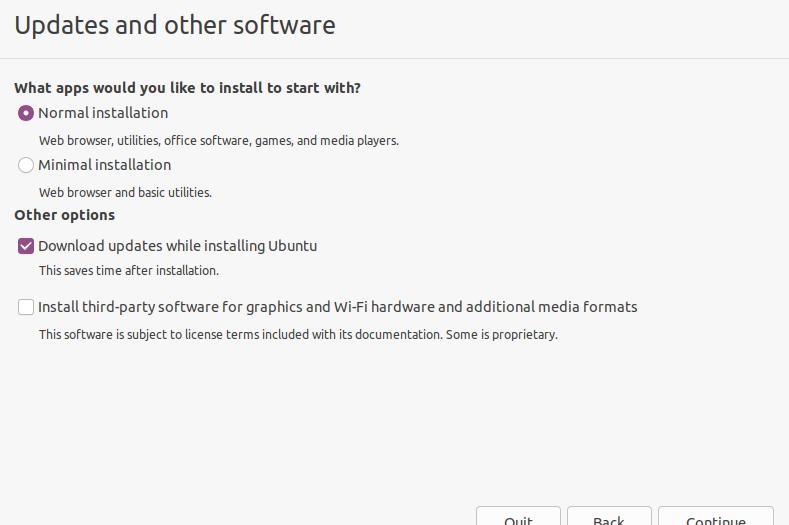


* Выбираем пункт **Install Ubuntu** и нажимаем **Enter**



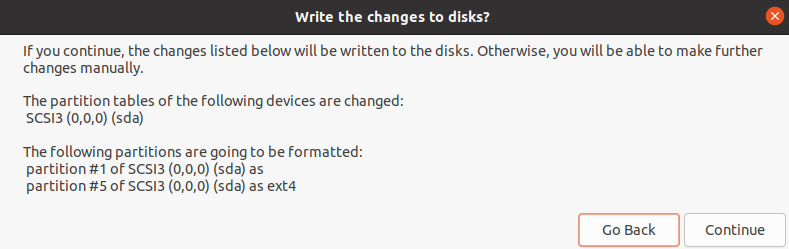
*После этого пункта может начаться процесс проверки диска, рекомендуется на пропускать этот пункт.*

* Далее откроется мастер установки ОС:
  + Выбираем язык – **Continue**
  + Выбираем раскладку клавиатуры (рекомендуется выбрать английскую раскладку) – **Continue**
  + Выбираем **Normal Installation**, также рекомендуется поставить галочку рядом с пунктом **Download updates while installing Ubuntu**



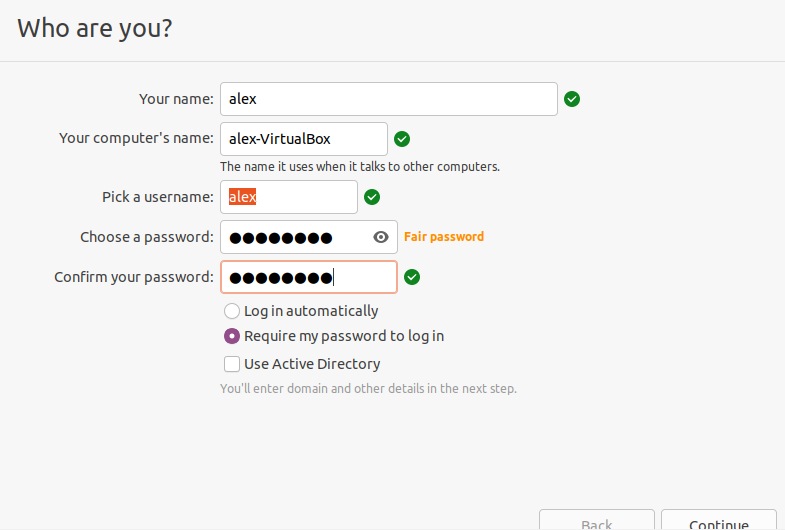
Далее нажимаем **Continue**

* + В следующем окне выбираем параметр **Erase disk and install Ubuntu - Install now**
  + Откроется окно уточнения в котором будут указаны разделы, которые появятся после очистки диска



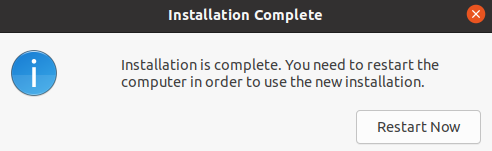
Нажимаем **Continue**

* + Указываем город, в котором Вы находитесь (Это нужно для часового пояса) – **Continue**
  + В следующем окне потребуется ввести **имя пользователя, имя хоста и пароль**



После заполнения всех полей нажимаем **Continue**

* Ожидаем пока процесс установки закончится, после успешной установки Вам будет предложено перезапустить ОС. Нажимаем кнопку **Restart Now**



После переустановки должна запуститься Ubuntu 20.04.

На этом процесс настройки виртуальной машины с включенной Nested Virtualization закончен. *При работающей Nested Virtualization у Вас должна появиться возможность создавать виртуальные машины внутри виртуальных машин.*

**Настройка VPN**

*К сожалению, на данный момент часть репозиториев недоступна для пользователей из РФ. Для работы с этими репозиториями потребуется VPN-сервис. Если у Вас уже есть любой настроенный VPN-клиент, то пропустите данный пункт.*

1. Переходим на сайт **нщ** и наживаем кнопку **Signup**
2. Перед нами появится список планов использования, находим план **Free** и наживаем **GET FREE**
3. Следуя инструкциям, создайте нового пользователя
4. В личном кабинете найдите пункт **Загрузки – Клиенты ProtonVPN** выберите вашу версию ОС, далее откроется инструкция по установке. Следуя инструкции установки установите VPN-клиент.

*Остальные пункты настройки VPN-клиента будут рассмотрены в следующих пунктах для каждой ОС отдельно.*

**Настройка рабочего места в Debian/Ubuntu**

*Настройка описана на примере Ubuntu 20.04*

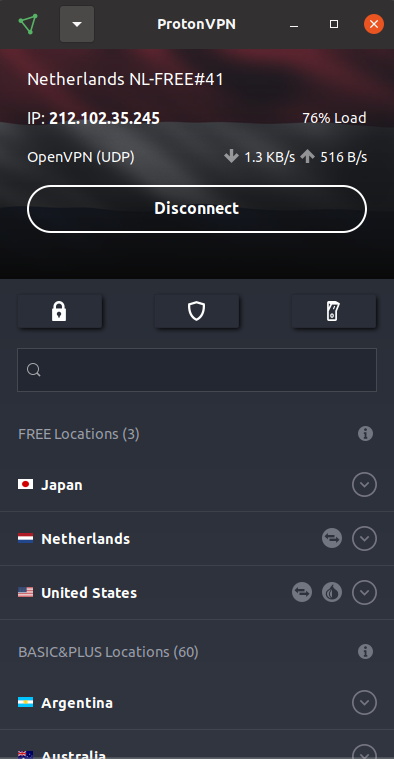
1. Настройка VPN-клиента *(Если вы уже пользуетесь другим VPN-клиентом, то пропустите данный пункт)*
   1. Переходим к инструкции по установке **-** <https://protonvpn.com/support/linux-ubuntu-vpn-setup/>
   2. Скачиваем .deb-пакет из пункта **Get the DEB setup package for the ProtonVPN repository**
   3. По умолчанию, файл должен быть скачан в папку загрузки, перейдём в папку Загрузки и установим файл:

*cd Downloads/*

*сprotonvpn-stable-release\_1.0.1-1\_all*

*(Версия protonVPN может отличаться)*

* 1. Обновим список пакетов: *sudo apt update*
  2. Установим protonVPN: *sudo apt install -y protonvpn*
  3. **Откроем ProtonVPN** и зайдём в него **с логином и паролем** от созданной учётной записи.
  4. Для включения VPN выберем одну из бесплатных локаций и нажмём **connec:Lt**. После этого произойдёт подключение к VPN:



1. Установка базовых программ, которые могут понадобиться при выполнении домашних заданий:
   1. Обновим список пакетов: *sudo apt update*
   2. Установим текстовый редактор Vim: *sudo apt install -y vim*
   3. Установим программы для анализа сетевого трафика:

*sudo apt install -y traceroute net-tools tcpdump*

* 1. Установим программы для передачи файлов: *sudo apt install -y curl wget*
  2. Установим программу контроля версий: *sudo apt install -y git*
  3. Для выполнения домашних заданий Вам потребуется редактор кода, можно пользоваться любым редактором кода, который Вам нравится.

Например:

* + - Visual Studio Code – <https://code.visualstudio.com/download>
    - Atom - <https://atom.io/>

Перейдите по ссылке – скачайте .deb-пакет – установите редактор *sudo dpkg -i <имя скаченного пакета>*

1. Установка Oracle VirtualBox
   1. Добавляем GPG-ключ репозитория: *wget -q https://www.virtualbox.org/download/oracle\_vbox\_2016.asc -O- | sudo apt-key add -*
   2. Добавляем репозиторий VirtualBox: *sudo add-apt-repository "deb [arch=amd64] http://download.virtualbox.org/virtualbox/debian $(lsb\_release -cs) contrib"*
   3. Обновляем список пакетов: *sudo apt update*
   4. Установим VirtualBox: *sudo apt install -y virtualbox-6.1*
   5. Установим VirtualBox extension pack: *sudo apt install -y virtualbox-ext-pack*

*(Во время установки потребуется принять лицензионное соглашение)*

На этом установка VirtualBox закончена.

1. Установка Hashicorp Vagrant
   1. Добавляем GPG-ключ репозитория: *curl -fsSL https://apt.releases.hashicorp.com/gpg | sudo apt-key add -*
   2. Добавляем репозиторий Hashicorp: sudo apt-add-repository "deb [arch=amd64] https://apt.releases.hashicorp.com $(lsb\_release -cs) main"
   3. Обновляем список пакетов: *sudo apt update*
   4. Установим Vagrant: *sudo apt install -y vagrant*
   5. Проверим версию Vagrant: *vagrant version \*

*Версия Vagrant должна быть 2.2.17 или выше*

На этом установка Hashicorp Vagrant завершена

1. Установка Packer

*Packer также можно установить из репозитория Hashicorp*

* 1. Установим packer: *sudo apt install -y packer*
  2. Проверим версию packer: *packer --version*

1. Установка Ansible
   1. Устанавливаем Ansible: *sudo apt install -y ansible*
   2. Проверяем версию Ansible: *ansible --version*

*Версия Ansible должна быть 2.8 или выше*

**Настройка рабочего места в RHEL-based системах**

*Настройка описана на примере Fedora 35*

1. Настройка VPN-клиента *(Если вы уже пользуетесь другим VPN-клиентом, то пропустите данный пункт)*
   1. Переходим к инструкции по установке **-** <https://protonvpn.com/support/official-linux-vpn-fedora/>
   2. Скачиваем .rpm-пакет из пункта **Get the ProtonVPN repo setup RPM package**
   3. По умолчанию, файл должен быть скачан в папку загрузки, перейдём в папку Загрузки и установим файл:

*cd Downloads/*

*sudo rpm -i protonvpn-stable-release-1.0.1-1.noarch.rpm*

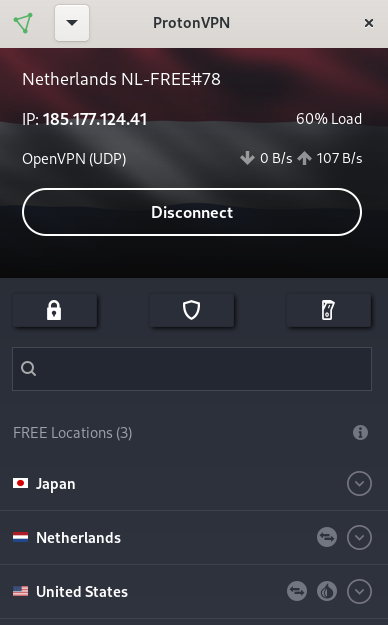
*(Версия protonVPN может отличаться)*

* 1. Обновим список пакетов: *sudo dnf update -y*
  2. Установим protonVPN: *sudo dnf install -y protonvpn*
  3. Установим пакеты зависимостей для protonVPN:

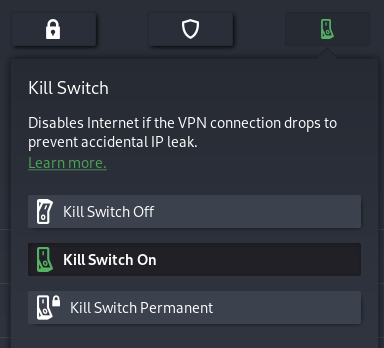
*sudo dnf install -y python3-pip*

*pip3 install --user dnspython>=1.16.0*

* 1. **Откроем ProtonVPN** и зайдём в него **с логином и паролем** от созданной учётной записи.
  2. Для включения VPN выберем одну из бесплатных локаций и нажмём **connect**. После этого произойдёт подключение к VPN:



В дистрибутивах Fedora, после установки ProtonVPN могут возникнуть проблемы с работой DNS: все DNS-запросы не будут получать ответ, если ProtonVPN не будет включен. Чтобы не допустить такую ситуацию, включите функцию **Kill switch**



1. Установка базовых программ, которые могут понадобиться при выполнении домашних заданий:
   1. Обновим список пакетов: *sudo dnf update -y*
   2. Установим текстовый редактор Vim: *sudo dnf install -y vim*
   3. Установим программы для анализа сетевого трафика:

*sudo dnf install -y traceroute net-tools tcpdump*

* 1. Установим утилиты для передачи файлов: *sudo dnf install -y curl wget*
  2. Установим утилиту контроля версий: *to*
  3. Для выполнения домашних заданий Вам потребуется редактор кода, можно пользоваться любым редактором кода, который Вам нравится.

Например:

* + - Visual Studio Code – <https://code.visualstudio.com/download>
    - Atom - <https://atom.io/>

Перейдите по ссылке – скачайте .rpm-пакет – установите редактор  *sudo dnf install -y <имя скаченного пакета>*

1. Установка Oracle VirtualBox
   1. Добавляем репозиторий VirtualBox:

*cd /etc/yum.repos.d/*

*sudo wget http://download.virtualbox.org/virtualbox/rpm/fedora/virtualbox.repo*

* 1. Обновляем список пакетов: *sudo dnf -y update*
  2. Установим пакеты, которые потребуются для корректной работы VirtualBox: *sudo dnf install -y binutils gcc make patch libgomp glibc-headers glibc-devel kernel-headers kernel-devel dkms qt5-qtx11extras libxkbcommon*
  3. Установим VirtualBox: *sudo dnf install -y virtualbox-6.1*
  4. Добавим права для пользователя:

*sudo usermod -a -G vboxusers <имя пользователя>*

* 1. Скачиваем с сайта Orcale (<https://www.virtualbox.org/wiki/Downloads>) **Oracle VM VirtualBox Extension Pack**. После скачивания запускаем файл двойным щелчком.

*(Во время установки потребуется принять лицензионное соглашение)*

На этом установка VirtualBox закончена.

1. Установка Hashicorp Vagrant
   1. Установим yum-utils: *sudo dnf install -y dnf-plugins-core*
   2. *Добавляем репозиторий Hashicorp:*

*sudo dnf config-manager --add-repo https://rpm.releases.hashicorp.com/fedora/hashicorp.repo*

* 1. Обновим пакеты: *sudo dnf -y update*
  2. *Установим Vagrant: sudo dnf -y install vagrant*
  3. Проверим версию Vagrant: *vagrant version*

*Версия Vagrant должна быть 2.2.17 или выше*

На этом установка Hashicorp Vagrant завершена

1. Установка Packer

*Packer также можно установить из репозитория Hashicorp*

* 1. Установим packer: *sudo dnf install -y packer*
  2. Проверим версию packer: *packer –version*

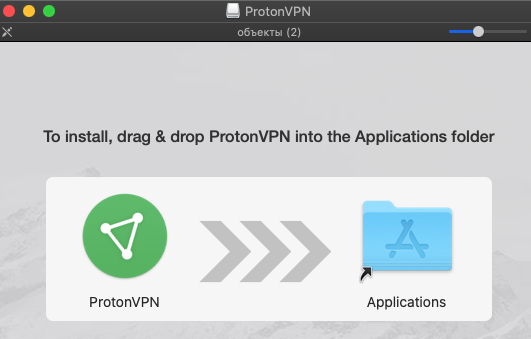
1. Установка Ansible
   1. Устанавливаем Ansible: *sudo dnf install -y ansible*
   2. Проверяем версию Ansible: *ansible --version*

*Версия Ansible должна быть 2.8 или выше*

**Настройка рабочего места в MacOS**

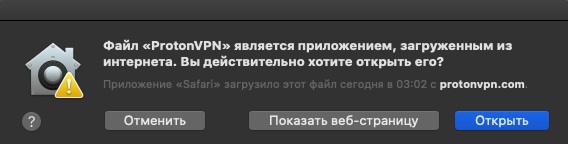
*Настройка описана на примере MacOS Monterey (На процессоре Intel)*

1. Настройка VPN-клиента *(Если вы уже пользуетесь другим VPN-клиентом, то пропустите данный пункт)*
   1. Переходим к инструкции по установке **-** <https://protonvpn.com/support/protonvpn-mac-vpn-application/>
   2. Скачиваем .dmg-пакет из пункта **Download the ProtonVPN app for macOS.**
   3. Щёлкаем 2 раза по скачанному файлу, откроется окно установки ProtonVPN



* 1. Перетащим значок ProtonVPN в Applications и дождёмся, пока завершится копирование
  2. **Откроем ProtonVPN** и зайдём в него **с логином и паролем** от созданной учётной записи.

*При первом открытии может появиться окно безопасности, тут нажимаем* ***Открыть***



* 1. Для включения VPN выберем одну из бесплатных локаций и нажмём **connect**. После этого произойдёт подключение к VPN:

*При первом подключении MacOS может запросить разрешение на создание VPN-туннелей, с помощью программы ProtonVPN. При появлении данного запроса нажмите* ***разрешить****.*

1. Установка базовых программ, которые могут понадобиться при выполнении домашних заданий:
   1. Заходим в терминал: **Переход – Утилиты – Терминал**
   2. Установим менеджер пакетов Homebrew: */bin/bash -c "$(curl -fsSL* [*https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh*](https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh)*)"*
   3. Во время установки несколько раз надо будет ввести пароль пользователя с правами администратора
   4. Установим текстовый редактор Vim: *brew install vim*
   5. Установим программы для анализа сетевого трафика:

*brew install traceroute net-tools tcpdump*

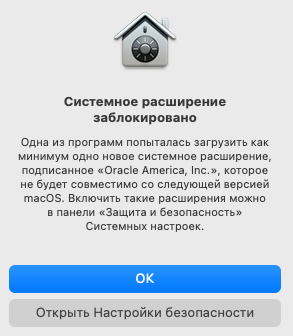
* 1. Установим утилиты для передачи файлов: *brew install curl wget*
  2. Установим утилиту контроля версий: *brew install git*
  3. Для выполнения домашних заданий Вам потребуется редактор кода, **можно пользоваться любым редактором кода**, который Вам нравится.

Например:

* + - Visual Studio Code – <https://code.visualstudio.com/download>
    - Atom - <https://atom.io/>

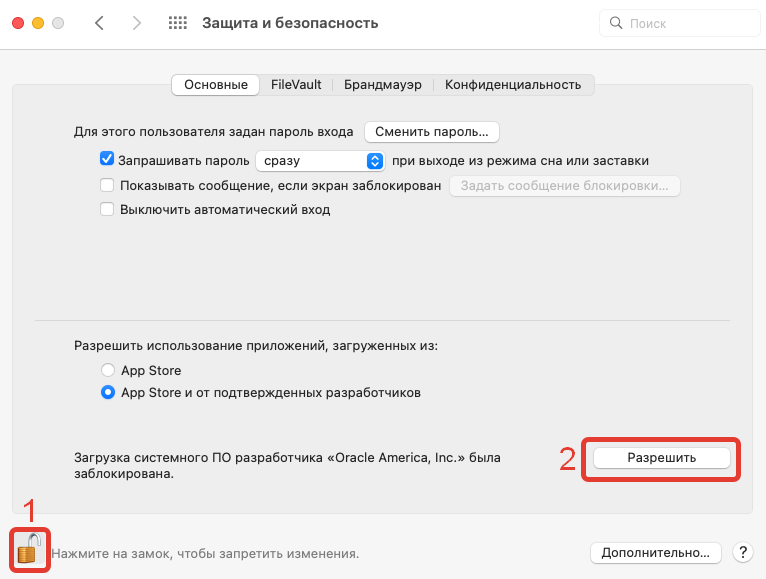
Перейдите по ссылке – скачайте .dmg-пакет – установите редактор дважды щёлкнув по скачанному файлу.

1. Установка Oracle VirtualBox
   1. Заходим на сайт Oracle VirtualBox (<https://www.virtualbox.org/wiki/Downloads>)
   2. Выбираем версию ОС **OS X hosts**
   3. Начнётся скачивание файла, после скачивания запускаем .dmg-файл двойным щелчком
   4. В появившемся окне запускам файл **VirtualBox.pkg** и следуем мастеру установки
   5. После установки может появиться ошибка **Системное расширение заблокировано**

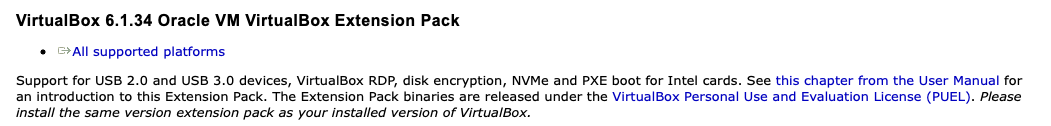


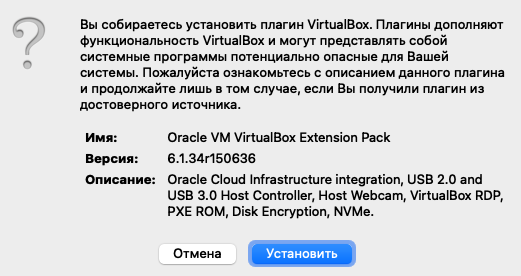
Нажимаем **Открыть Настройки безопасности**

* 1. Откроется окно «Защита и безопасность»



В данном окне сначала нужно нажать на **замок** – **ввести пароль** и потом нажать **разрешить**

* 1. Перезагружаем компьютер
  2. Скачиваем **Oracle VM VirtualBox Extension Pack**
  3. После скачивания запускаем файл двойным щелчком, он откроется в VirtualBox



Нажимаем **Установить,** далее принимаем условия лицензионного соглашения, **вводим пароль** и нажимаем **ОК**

* 1. После успешной установки появится окно с сообщением что плагин успешно установлен

На этом установка VirtualBox закончена.

1. Установка Hashicorp Vagrant
   1. Открываем терминал **(Переход – Утилиты - Терминал)**
   2. Установим Vagrant: *brew install vagrant*
   3. Проверим версию vagrant: *vagrant version*

*Версия Vagrant должна быть 2.2.17 или выше*

На этом установка Hashicorp Vagrant завершена

1. Установка Packer
   1. Установим packer:

*brew tap hashicorp/tap*

*brew install hashicorp/tap/packer*

* 1. Проверим версию packer: *packer –version*

1. Установка Ansible
   1. Устанавливаем Ansible: *brew install ansible*
   2. Проверяем версию Ansible: *ansible –version*

*Версия Ansible должна быть 2.8 или выше*

**Vagrant. Базовая информация**

*Vagrant — ПО для создания и настройки виртуальных сред. Vagrant является надстройкой над программами виртуализации, например VirtualBox.*

*Vagrant состоит из нескольких элементов:*

* *Boxes — сжатые образы виртуальных машин. Один образ можно использовать несколько раз (аналог iso- файла)*
* *cli — консольные команды для управления средой*
* *VagrantFile — файл с описанием конфигурации*
* *Plugins — отдельные компоненты, которые упростят работу. Например можно поставить надстройку над vmWare и управлять машинами через Vagrant*

**Как работает Vagrant**

*Вся виртуальная среда описывается в Vagrant-файле. После создания Vagrantfile среду нужно запустить с помощью команды Vagrant up. После ввода команды запуска Vagrant ищет Vagrantfile в каталоге, откуда была запущена данная команда, если в текущем каталоге Vagrantfile не найден, то Vagrant поднимется на каталог выше по иерархии в файловой системе.*

*В разных каталогах можно хранить разные Vagrant-файлы*

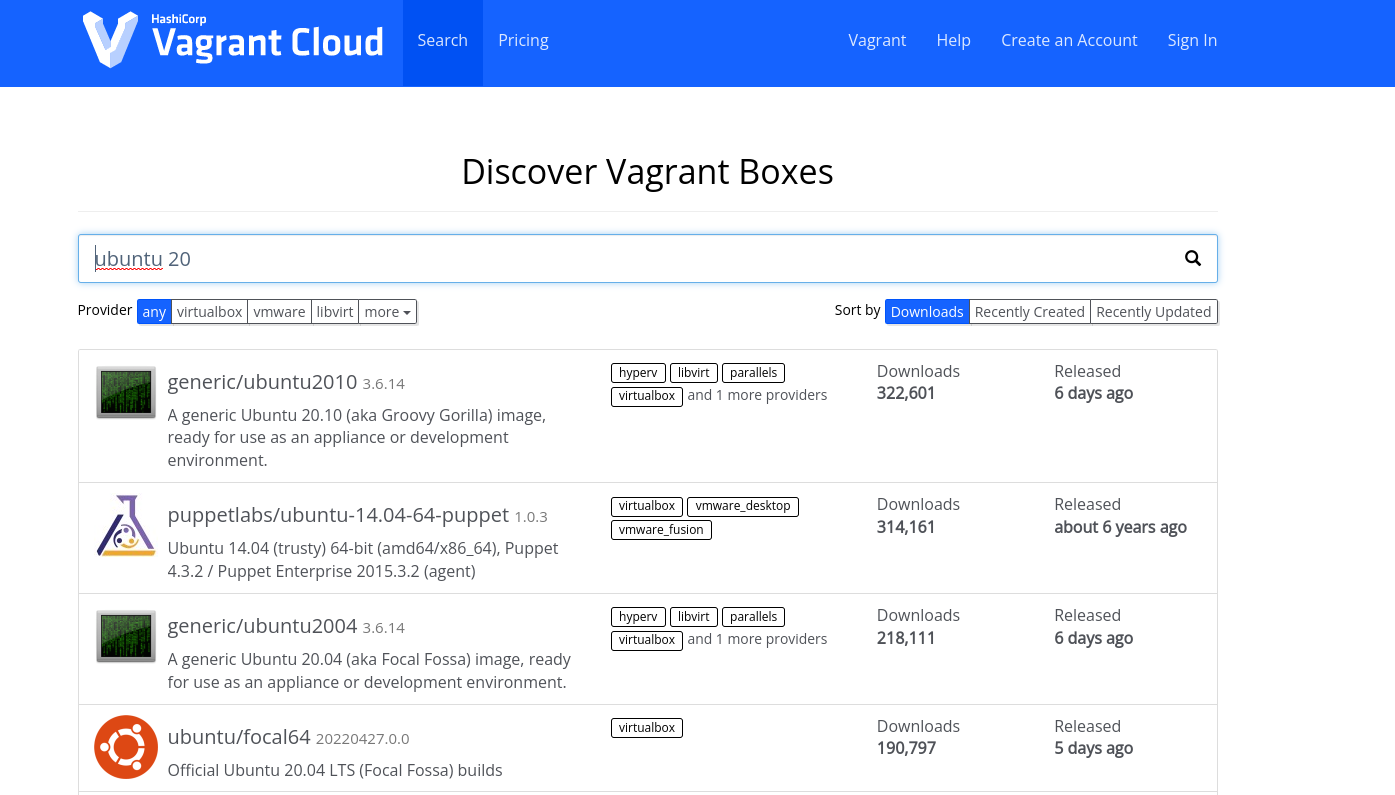
*Давайте подробнее рассмотрим создание простой виртуальной машины с помощью Vagrant:*

* *Создадим каталог* ***test\_vm*** *и зайдём в него: mkdir test\_vm && cd test\_vm*
* Далее мы можем начать писать Vagrantfile с нуля или запросить файл из Vagrant cloud (потребуется включенный VPN, если Вы находитесь в России)

Запрос из Vagrant cloud: *vagrant init <имя Vagrant box>*

Vagrant boxes можно посмотреть на сайте <https://app.vagrantup.com/boxes/search>

Допустим мы хотим запустить виртуальную машину на Ubuntu 20.04 ищем в Vagrant Cloud образы Ubuntu 20.04



Тут мы видим название нашего Vagrant box — ***ubuntu/focal64***

Тогда команда запроса vagrant-файла будет следующей: *vagrant init ubuntu/focal64*

После выполнения данной команды, мы видим скачанный Vagrantfile. Рассмотрим его более подробно:

# -\*- mode: ruby -\*-

# vi: set ft=ruby :

# All Vagrant configuration is done below. The "2" in Vagrant.configure

# configures the configuration version (we support older styles for

# backwards compatibility). Please don't change it unless you know what

# you're doing.

Vagrant**.**configure**(**"2"**)** **do** **|**config**|**

# The most common configuration options are documented and commented below.

# For a complete reference, please see the online documentation at

# https://docs.vagrantup.com.

# Every Vagrant development environment requires a box. You can search for

# boxes at https://vagrantcloud.com/search.

config**.**vm**.**box **=** "ubuntu/focal64"

# Disable automatic box update checking. If you disable this, then

# boxes will only be checked for updates when the user runs

# `vagrant box outdated`. This is not recommended.

# config.vm.box\_check\_update = false

# Create a forwarded port mapping which allows access to a specific port

# within the machine from a port on the host machine. In the example below,

# accessing "localhost:8080" will access port 80 on the guest machine.

# NOTE: This will enable public access to the opened port

# config.vm.network "forwarded\_port", guest: 80, host: 8080

# Create a forwarded port mapping which allows access to a specific port

# within the machine from a port on the host machine and only allow access

# via 127.0.0.1 to disable public access

# config.vm.network "forwarded\_port", guest: 80, host: 8080, host\_ip: "127.0.0.1"

# Create a private network, which allows host-only access to the machine

# using a specific IP.

# config.vm.network "private\_network", ip: "192.168.33.10"

# Create a public network, which generally matched to bridged network.

# Bridged networks make the machine appear as another physical device on

# your network.

# config.vm.network "public\_network"

# Share an additional folder to the guest VM. The first argument is

# the path on the host to the actual folder. The second argument is

# the path on the guest to mount the folder. And the optional third

# argument is a set of non-required options.

# config.vm.synced\_folder "../data", "/vagrant\_data"

# Provider-specific configuration so you can fine-tune various

# backing providers for Vagrant. These expose provider-specific options.

# Example for VirtualBox:

#

# config.vm.provider "virtualbox" do |vb|

# # Display the VirtualBox GUI when booting the machine

# vb.gui = true

#

# # Customize the amount of memory on the VM:

# vb.memory = "1024"

# end

#

# View the documentation for the provider you are using for more

# information on available options.

# Enable provisioning with a shell script. Additional provisioners such as

# Ansible, Chef, Docker, Puppet and Salt are also available. Please see the

# documentation for more information about their specific syntax and use.

# config.vm.provision "shell", inline: <<-SHELL

# apt-get update

# apt-get install -y apache2

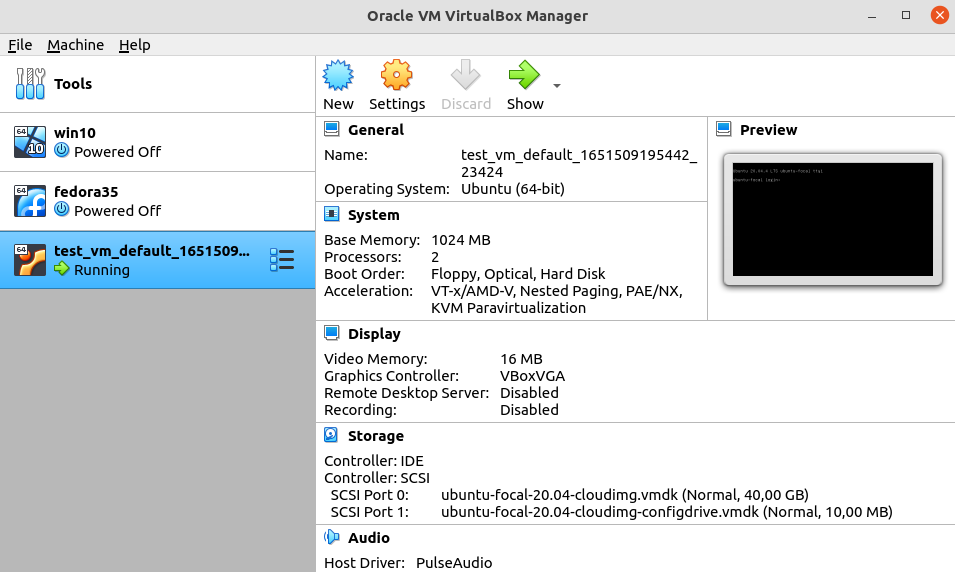
# SHELL

**end**

Мы видим, что большая часть текста закомментирована. Код, который незакомментирован, просто создаст нам одну виртуальную машину со специафиациями по умолчанию, работающую на ОС Ubuntu 20.04

Для запуска данной виртуальной машины потребуется сохранить Vagrantfile и ввести команду *vagrant up*

После завешения работы команды мы увидим в VirtualBox новую запущенную виртуальную машину



Для того, чтобы подключиться по ssh к созданной ВМ, введём команду *vagrant ssh* (Если vagrantfile содержит несколько виртуальных машин, то для подключения к конкретной ВМ нужно ввести команду *vagrant ssh <имя ВМ> Если имя ВМ не указано в Vagrantfile, то имя по умолчанию будет* ***default****).*

Для удаления виртуальной машины введём команду *vagrant destroy*

Давайте внесём в наш Vagrantfile более подробные настройки:

# -\*- mode: ruby -\*-

# vi: set ft=ruby :

**#Параметры указываются в цикле**

Vagrant**.**configure**(**"2"**)** **do** **|**config**|**

**#Указываем, какую ОС мы будем использовать**

config**.**vm**.**box **=** "ubuntu/focal64"

**#Можно указать конкретную версию сборки**

**#Номера сборок можно посмотреть в Vagrant Cloud**

config**.**vm**.**box\_version **=** "20220427.0.0"

**#Проброс порта с гостевой машины в хост**

**#Порт 80 в созданной ВМ будет доступен нам на порту 8080 хоста**

config**.**vm**.**network "forwarded\_port"**,** guest: 80**,** host: 8080

**#Указываем настройки спецификации ВМ**

**#Указывается в отдельном цикле**

config**.**vm**.**provider "virtualbox" **do** **|**vb**|**

**# Указываем количество ОЗУ и ядер процессора**

vb**.**memory **=** "1024"

vb**.**cpus **=** "1"

**end**

**#Первоначальная настройка созданной ВМ**

**#Установка и запуск Веб-сервера Apache2**

config**.**vm**.**provision "shell"**,** inline: **<<-**SHELL

sudo apt**-**get update

sudo apt**-**get install **-**y apache2

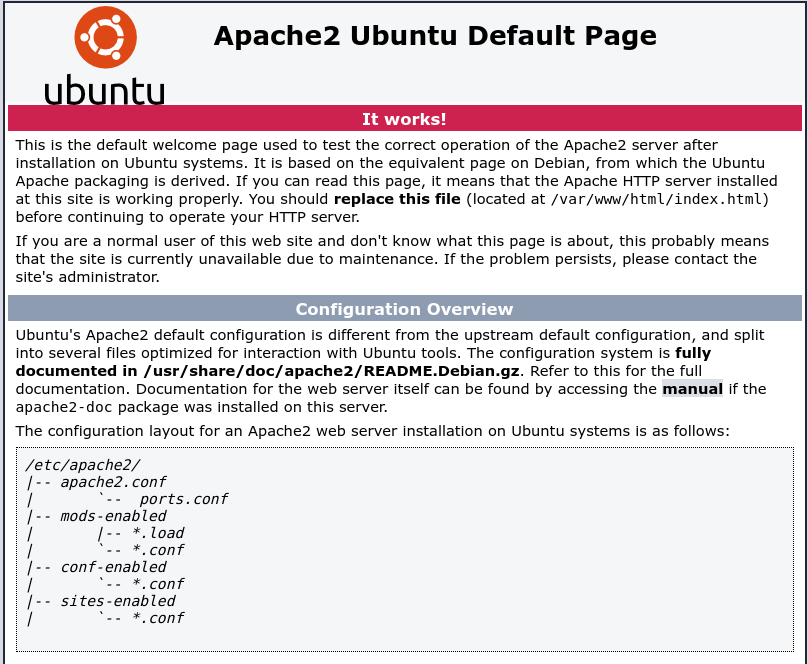
SHELL

**end**

Сохраним и снова введём команду *vagrant up*

Тогда Vagrant развернёт нам одну ВМ на Ubuntu 20.04, с 1 Гбайтом ОЗУ и 1 ядром процессора. Также в ОС после развертывания установится Веб-сервер Apache и будет проброшен порт для проверки работы веб-сервера.

После успешного развертывания ВМ, зайдём в наш браузер и перейдём по адресу [http://127.0.0.1:8080](http://127.0.0.1:8080/) если всё настроено правильно, то мы увидим страницу приветствия Apache.



Если мы захотим поменять какой-нибудь параметр ВМ, например сделать 2 ядра процессора, то нам потребуется **внести изменения** в Vagrantfile, **сохранить** его и ввести команду *vagrant reload. ВМ перезапустится и добавит ещё одно ядро к нашей ВМ.*

*Обратите внимание, скрипт первоначальной установки снова запускаться не будет, он стартует один раз при создании ВМ. Если по какой-либо причине его нужно запустить снова, то тогда введите команду vagrant provision*

Vagrantfile имеет синтаксическую структуру языка Ruby. Это значит, что в Vagrantfile можно добавлять конструции циклов и условий, используя синтаксис языка Ruby.

Более подробно о настройке всех параметров можно почитать в официальной документации (ссылка в источниках).

**Загрузка домашнего задания в GitHub**

Все ваши домашние работы будут загружаться в GitHub.

Для работы с GitHub в нём нужно зарегистироваться:

* + Переходим на сайт [https://github.com](https://github.com/) и нажимаем на кнопку **sign up**
  + Указываем данные для регистрации аккаунта
  + После ввода данных Вам придёт код подтверждения на почту
  + При выборе типа аккаунта выберите **Free аккаунт**
  + После завершения регистрации Вы попадёте в ваш аккаунт
  + Для того, чтобы загружать ДЗ вам потребуется токен, для создания токена **щёлкните по вашему аватару** (в правом верхнем углу) — **settings** — **Developer settings — Personal access tokens — Generate new token**
  + Указываем имя токена, дату истечения действия токена, ниже ставим все галочки
  + Далее наживаем **Generate token.** После этого на экране появится токен, скопируйте и сохраните его.

На этом регистрация аккуанта на GitHub закончена.

**Для примера отправим наш Vagrantfile в GitHub:**

1) Заходим в Ваш аккаунт GitHub и создаем там репозиторий (**Create repository**)

2) В окне создания репозитория **указываем имя репозитория** (оно не должно дублироваться с именами уже созданых Вами репозиториев)и выбираем **тип репозитория** **Public** (видно всем)

3) Далее нажимаем **Create repository**

*После создания репозитория откроется окно с инструкциями как отправить файлы в указанный репозиторий*

4) В каталоге с нашим Vagrantfile создадим файл README.md в котором напишем информацию о нашем файле, например: *echo «Create Ubuntu 20.04 VM» >> README.md*

5) Добавим в папку систему контроля версий git: *git init*

6) Добавим наши файлы в систему контроля версий: *git add README.md Vagrantfile*

7) Зафиксируем изменения: *git commit -m «test commit»*

8) Добавим информацию о созданном репозитории в GitHub: *git remote add origin https://github.com/alex8443/test\_vm.git*

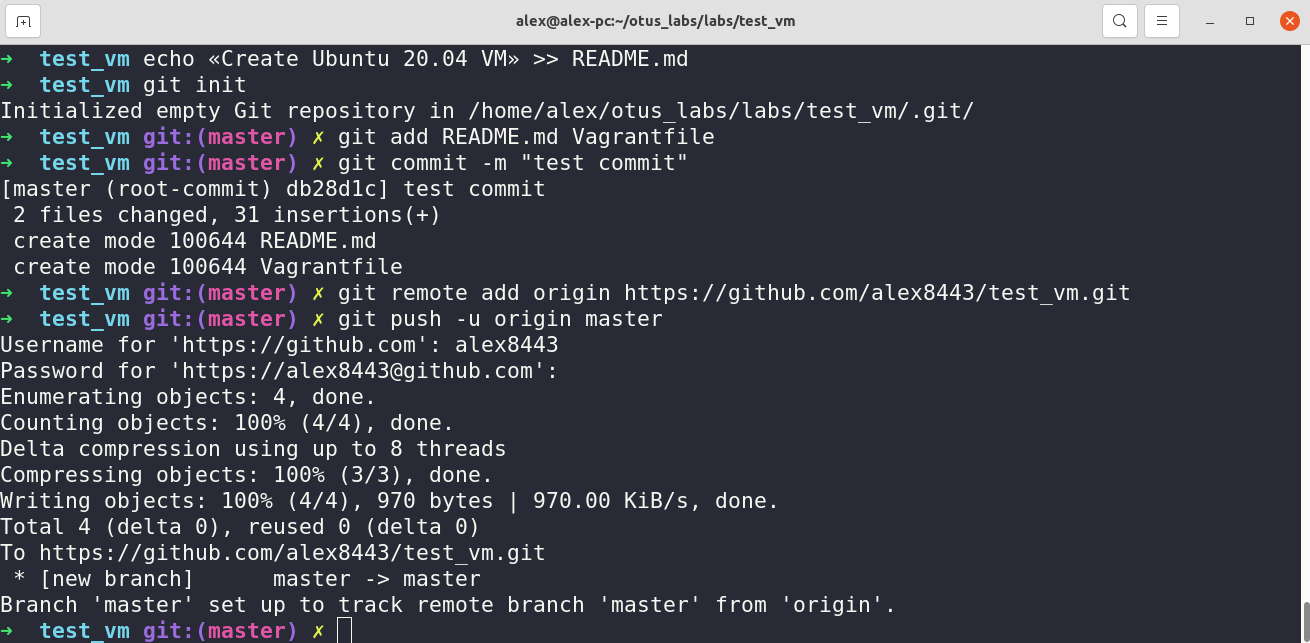
*(Адрес будет показан после создания репозитория)*

9) Отправляем файлы в удаленный репозиторий GitHub: *git push -u origin master*

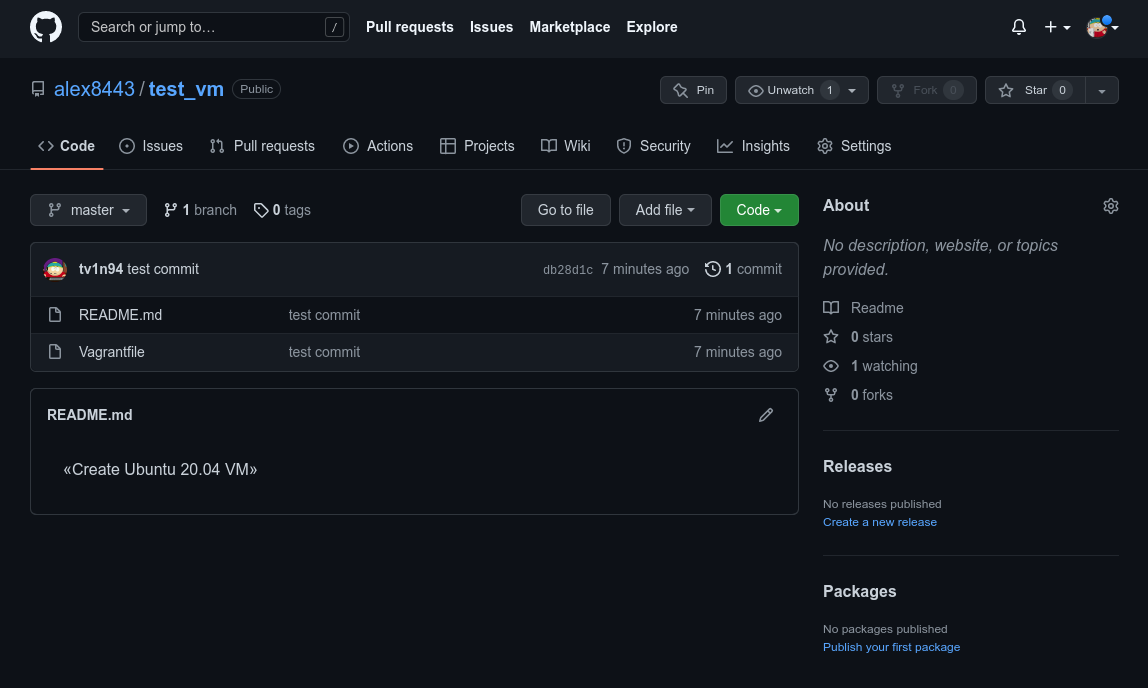
После ввода данной команды потребуется ввести:

* + Username *(можно указать имя пользователя или почтовый ящик)*
  + Password *(указываем token, который создали в прошлом пункте)*

После успешной аутентификации начнётся отправка файлов в GitHub.



Проверим, что файлы успешно загрузились в репозиторий:

Содержимое файла README.md будет отображаться на главной странице репозитория. 

**Критерии оценивания**

После настройки рабочего места у вас не должно возникать проблем с функциональными требованиями для выполнения домашних работ.

**Рекомендуемые источники**

* Инструкция по установке Hashicorp Vagrant - <https://www.vagrantup.com/downloads>
* Инструкции по настройке Packer - <https://www.packer.io/downloads>
* Инструкции по установке Ansible - <https://docs.ansible.com/ansible/2.9/installation_guide/intro_installation.html>
* Vagrant documentation - <https://www.vagrantup.com/docs>
* Информация о настройке ProtonVPN - <https://protonvpn.com/support/>
* Вебинар OTUS. Виртуальная лаборатория на Vagrant - <https://www.youtube.com/watch?v=koJ1UpEOcVw&ab_channel=OTUS%D0%9E%D0%BD%D0%BB%D0%B0%D0%B9%D0%BD-%D0%BE%D0%B1%D1%80%D0%B0%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5>