**Задание 1.1**

* Установить VirtualBox (или VMware workstation player), разобраться как создать виртуальную машину и создать новую vm на базе дистрибутива Centos ( Centos это параллельная open source ветка rhel );
* Подключиться к созданной vm по ssh через любой клиент.;
* Установить Python на созданной vm;
* В домашней директорий пользователя создать папку task; реализовать собственное key-values хранилище на Python. Данные будут сохраняться в файле storage.data(в формате JSON, можно использовать библиотеку tempfile, для хранения данных во временных файлах). Добавление новых данных в хранилище и получение текущих значений осуществляется с помощью утилиты командной строки storage.py. Пример работы утилиты:

Сохранение данных

$ storage.py --key key\_name --val value

Получение данных

$ storage.py --key key\_name

Обратите внимание, что значения по одному ключу не перезаписываются, а добавляются к уже сохраненным. Другими словами - по одному ключу могут храниться несколько значений. При выводе на печать, значения выводятся в порядке их добавления в хранилище (Пример ввода "test\_value,test\_value2,test\_value3" ). Формат вывода на печать для нескольких значений через запятую. Если значений по ключу не было найдено, выведите пустую строку или None. Сделать обработку исключений, если они будут возникать при тестировании. Скрипт должен работать в разных ОС.

Прислать оба файла на проверку, работу скрипта продемонстрировать в отчёте.

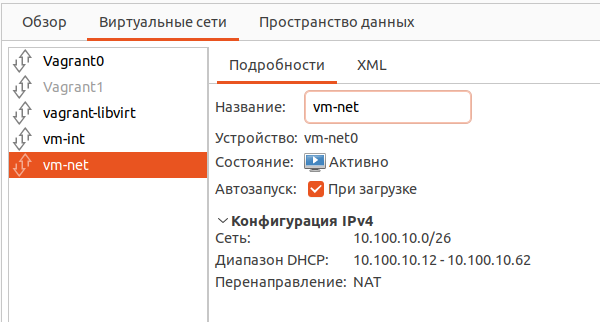
Как вариант установил Centos с помощью Vagrant сгенерировал ключи на хост машине, публичный закинул на виртуальную машину, добавил инфу о ключе в авторизованные.

Подключение через ssh доступно.

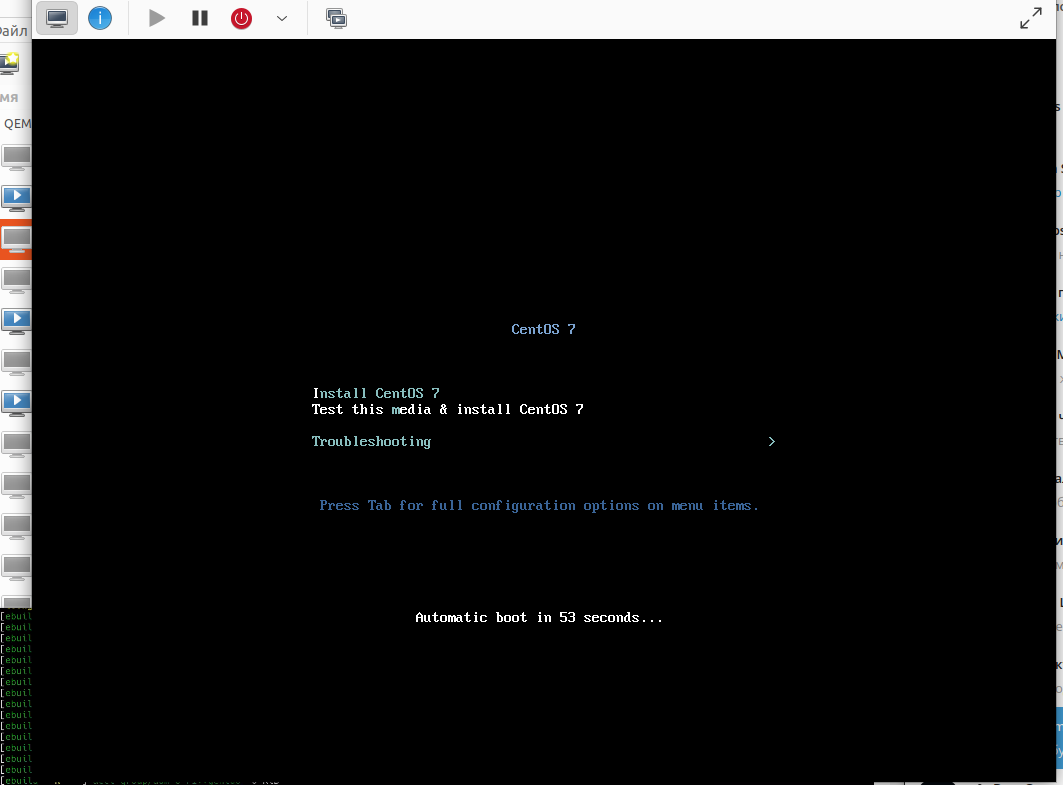
Vagrantfile:

*# -\*- 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.*  
  
ENV['VAGRANT\_DEFAULT\_PROVIDER'] = 'libvirt'  
Vagrant.configure("2") do |config|  
  
 config.vm.provider *:libvirt* do |vb|  
 vb.memory = *2048*  
end  
  
 config.vm.define "centos\_python" do |centos\_python|  
 centos\_python.vm.box = "centos/7"  
 centos\_python.vm.box\_check\_update = *false*  
 centos\_python.vm.hostname = "centos"  
 centos\_python.vm.network "private\_network", *ip:* "10.100.10.15"  
 config.vm.provision "file", *source:* "files/vagrant\_key.pub", *destination:* "/home/vagrant/.ssh/"  
 centos\_python.vm.provision "shell", *inline:* <<-SHELL  
 chmod 600 /home/vagrant/.ssh/vagrant\_key.pub  
 cat /home/vagrant/.ssh/vagrant\_key.pub >> /home/vagrant/.ssh/authorized\_keys  
 SHELL  
 end  
end

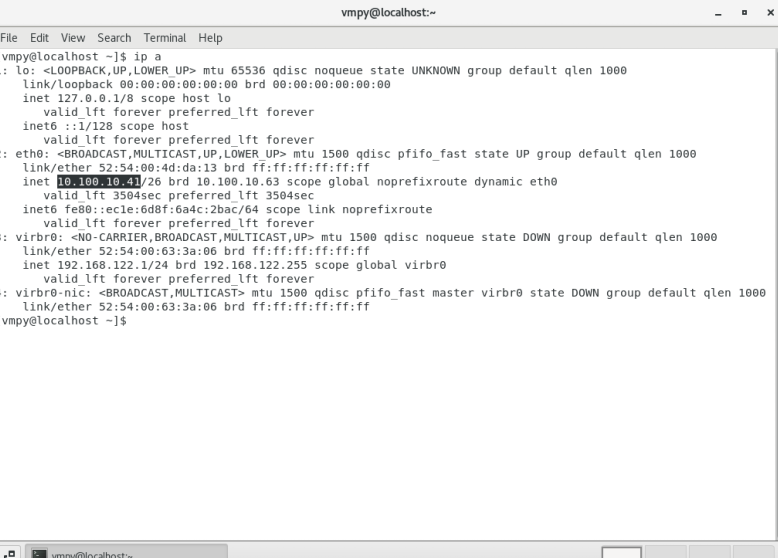
Создал виртуальную сеть, в настройках мост с хост системой, DHCP и NAT.

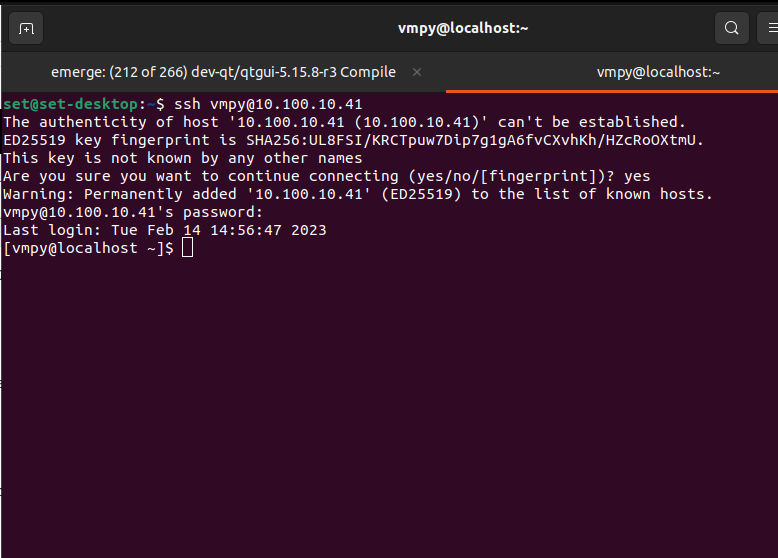


Установил Centos из образа ISO

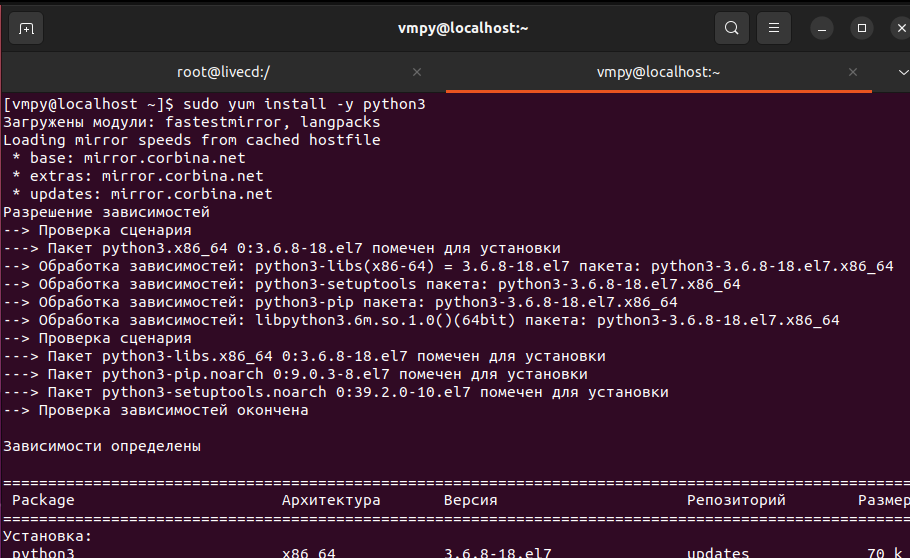


Узнал полученный IP

Подключился к ВМ через SSH



Установил Python



Мои знания по питону базовые, но разобраться в коде могу, по данной причине нашёл код в сети разобрался в нём немного скорректировал и проверил работоспособность.

Вот код подписал комменты как я его понял:

import argparse  
import json  
import os  
import tempfile

# данный блок считывает аргументы подсталенные при запуске кода, это --key и --val  
# записываем в args

parser = argparse.ArgumentParser()  
parser.add\_argument("--key", help=' random text')  
parser.add\_argument("--val", help='random\_text')  
args = parser.parse\_args()

#тут задаётся дирректория темп для любой ос и имя файла где будут храниться данные  
storage\_path = os.path.join(tempfile.gettempdir(), 'storage.data')  
  
if os.path.isfile(storage\_path): # проверяется есть ли файл storage.data  
 if args.val: #проверяется есть ли значение в ключе или был введен только ключ  
 with open(str(storage\_path), "r") as f: # если значение есть открываем файл на чтение  
 m = json.load(f) # переносим данные в «m» в формате json  
 if args.key in m: # проверяем существует ли ключ в файле  
 m[args.key] = m[args.key] + [args.val] # если есть то добавляем к нему значение  
 else:  
 m.update({args.key: [args.val]}) # если нет то добавляем новую запись  
 with open(str(storage\_path), "w") as f: # открываем файл на запись  
 json.dump(m, f) # переонсим данные в файл в формате json  
 else: # если значение ключа не было введено  
 try: # то считываем его из файла и выводим  
 with open(str(storage\_path), "r") as f: # открываем файл на чтение  
 m = json.load(f) # считываем данные  
 if m[args.key] is None: # если в файле нет введёного ключа  
 print(None) # выводим None  
 if len(m[args.key]) > 1: # если количество значений в ключе больше 1  
 print(', '.join(m.get(args.key))) # печатаем их через запятую  
 else: # иначе  
 print(\*m.get(args.key)) # печатаем значение ключа  
 except:  
 print(None) # при исключениях вывод none  
else: # если файла нет  
 if args.val: # проверяем было ли значение ключа  
 d = {args.key: [args.val]} # если да то заполняем массив значением  
 with open(str(storage\_path), "w") as f: # создаём и открываем файл на запись  
 json.dump(d, f) # записываем значения в файл  
 else: # если значения нет   
 print("no file to read") # выводим что файла нет

Работа кода

