ДЗ 11

Цель: Сможете защитить ваш сервер и настроить фильтрацию по протоколам **SSH и HTTP(S),** а также изучите технологию **NAT.**

Переосмыслив цель урока для себя понял её следующим образом:
Научиться практическим аспектам по защите сервера и настроить фильтрацию по протоколам ***SSH*** и ***HTTP(S)***, а также изучить технологию ***NAT***.

Условные обозначения (Тире – двойной дефис, поскольку при выполнении заданий LibreOffice его везде переправляет).

1. Почему нельзя так делать на ***удаленной машине***? И что делать если вдруг это произошло на виртуальной машине?

 # iptables -P INPUT DROP

Данная команда изменяет политики для входящих пакетов в цепочке правил ***INPUT*** *сервера, настроенным* ***netfilter*** *на отбрасывание (блокировку) → после применения данного правила на удалённой машине пропадёт интернет и, как следствие, произойдёт моментальная потеря связи с удалённым сервером. В случае работы с виртуальной машиной, всё поправимо либо перезагрузкой (если данные политики не сохранены в* ***iptables-save****, или в файл, при перезагрузке они будут потеряны) Либо можно поправить политики, залогинившись в любой* ***tty**** *сеанс и выполнить команду:*

 # iptables -P INPUT ACCEPT

2. Приведите пример настройки ***iptables***, который разрешит только ***порт 22*** и дальше проверять не будет. Те же, кто ***идет не*** на ***22 порт***, те ***будут отброшены***.

 # iptables -A INPUT -p tcp -m tcp –dport 22 -j ACCEPT
 # iptables -P INPUT DROP
 # iptables -P FORWARD DROP

```
usr@vbiptab:~$ sudo iptables -A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
usr@vbiptab:~$ sudo iptables -P INPUT DROP
usr@vbiptab:~$  sudo iptables -P FORWARD DROP
```

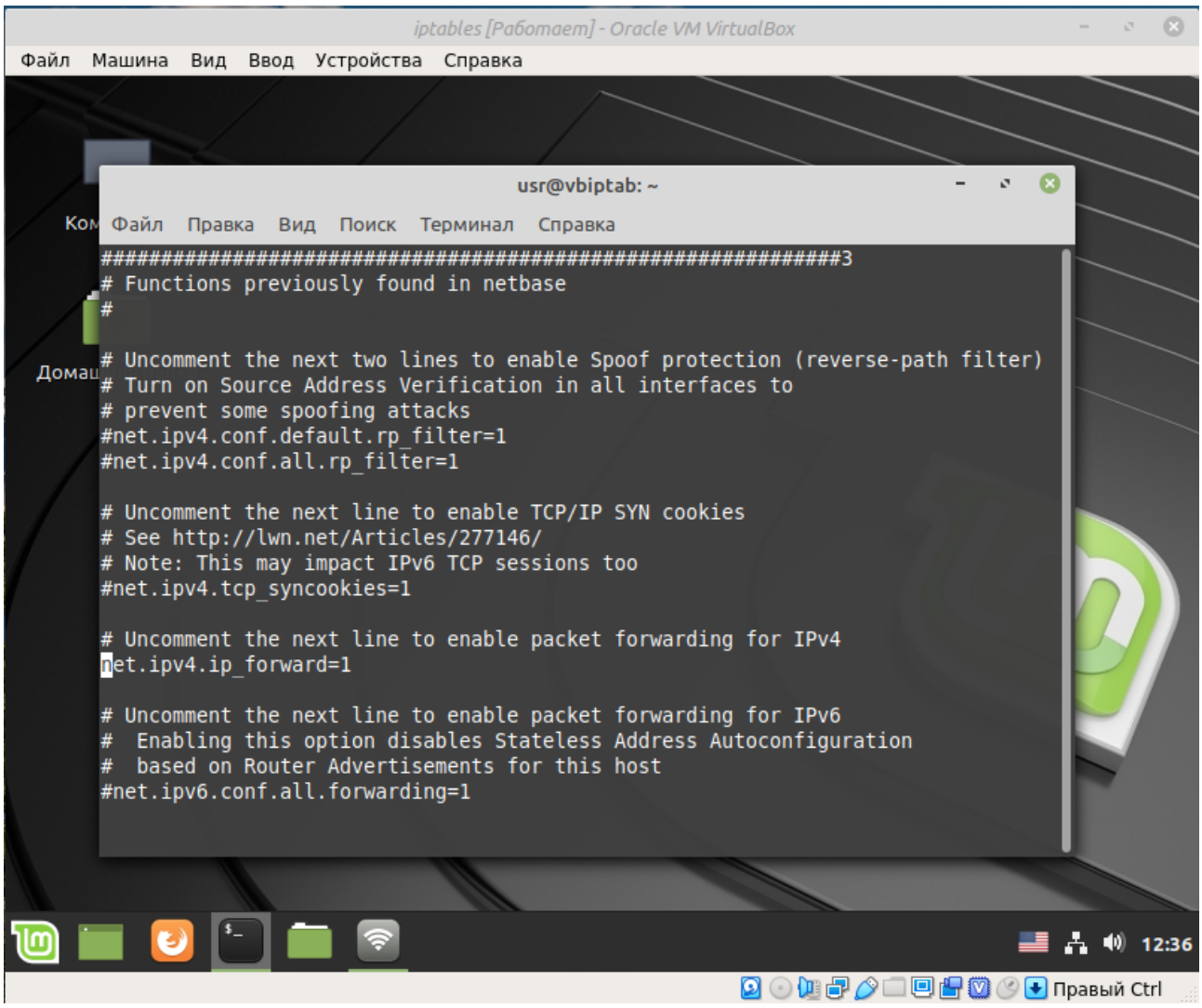
```
usr@vbiptab:~$ sudo iptables -L
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh

Chain FORWARD (policy DROP)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
usr@vbiptab:~$
```

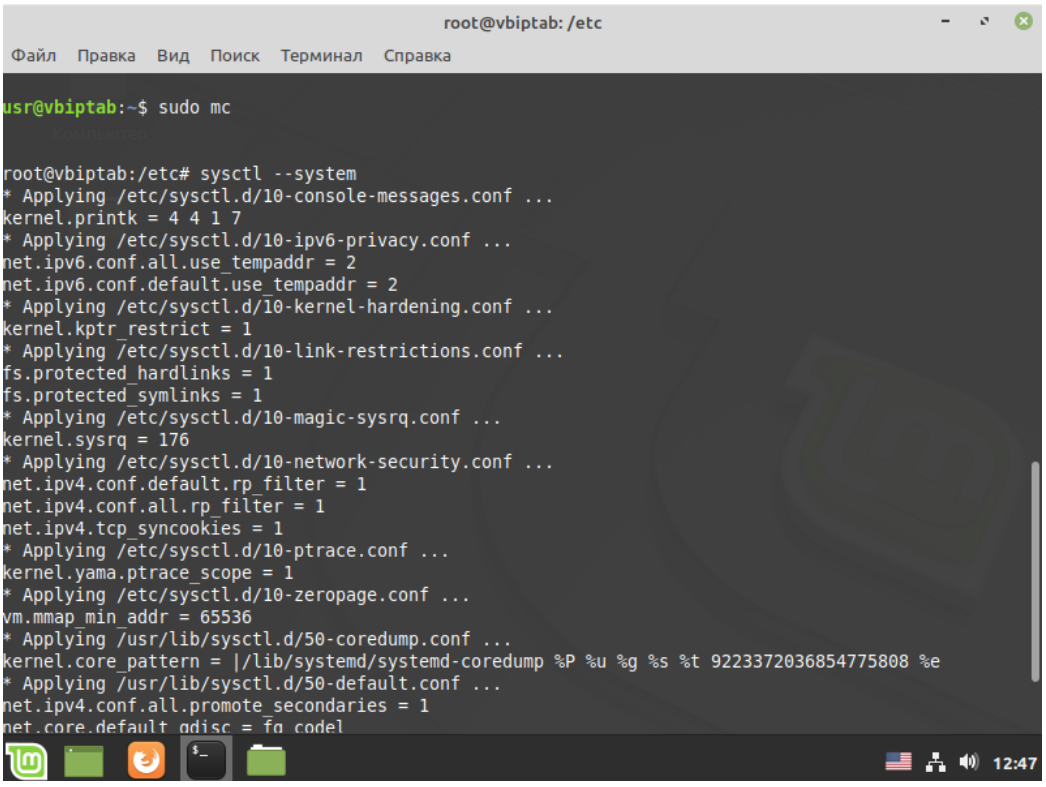
3. Настроить сетевой фильтр, чтобы был доступ только к сервисам ***HTTP(S) и SSH***.

Подготовка:

1) Для начала разрешим на своём сервере проброс пакетов из одной сети в другую, для этого раскомментируем строку ***net.ipv4.ip_forward=1*** в файле ***/etc/sysctl.conf*** и применим нашу настройку через команду:

sudo sysctl —system


iptables [Работаем] - Oracle VM VirtualBox
Файл Машина Вид Ввод Устройства Справка
usr@vbiptab: ~
Файл Правка Вид Поиск Терминал Справка
###3
Functions previously found in netbase
#
Uncomment the next two lines to enable Spoof protection (reverse-path filter)
Turn on Source Address Verification in all interfaces to
prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1
Uncomment the next line to enable TCP/IP SYN cookies
See http://lwn.net/Articles/277146/
Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1
# Uncomment the next line to enable packet forwarding for IPv6
# Enabling this option disables Stateless Address Autoconfiguration
# based on Router Advertisements for this host
#net.ipv6.conf.all.forwarding=1
12:36
Правый Ctrl


root@vbiptab: /etc
Файл Правка Вид Поиск Терминал Справка
usr@vbiptab:~$ sudo mc
root@vbiptab:/etc# sysctl --system
* Applying /etc/sysctl.d/10-console-messages.conf ...
kernel.printk = 4 4 1 7
* Applying /etc/sysctl.d/10-ipv6-privacy.conf ...
net.ipv6.conf.all.use_tempaddr = 2
net.ipv6.conf.default.use_tempaddr = 2
* Applying /etc/sysctl.d/10-kernel-hardening.conf ...
kernel.kptr_restrict = 1
* Applying /etc/sysctl.d/10-link-restrictions.conf ...
fs.protected_hardlinks = 1
fs.protected_symlinks = 1
* Applying /etc/sysctl.d/10-magic-sysrq.conf ...
kernel.sysrq = 176
* Applying /etc/sysctl.d/10-network-security.conf ...
net.ipv4.conf.default.rp_filter = 1
net.ipv4.conf.all.rp_filter = 1
net.ipv4.tcp_syncookies = 1
* Applying /etc/sysctl.d/10-ptrace.conf ...
kernel.yama.ptrace_scope = 1
* Applying /etc/sysctl.d/10-zeropage.conf ...
vm.mmap_min_addr = 65536
* Applying /usr/lib/sysctl.d/50-coredump.conf ...
kernel.core_pattern = |/lib/systemd/systemd-coredump %P %u %g %s %t 9223372036854775808 %e
* Applying /usr/lib/sysctl.d/50-default.conf ...
net.ipv4.conf.all.promote_secondaries = 1
12:47

2) Для всех установленных/проверенных соединений создадим правило, разрешающее пересылку всех пакетов state – **RELATED, ESTABLISHED**:

sudo iptables -A INPUT -m state –state RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A FORWARD -m state –state RELATED,ESTABLISHED -j ACCEPT

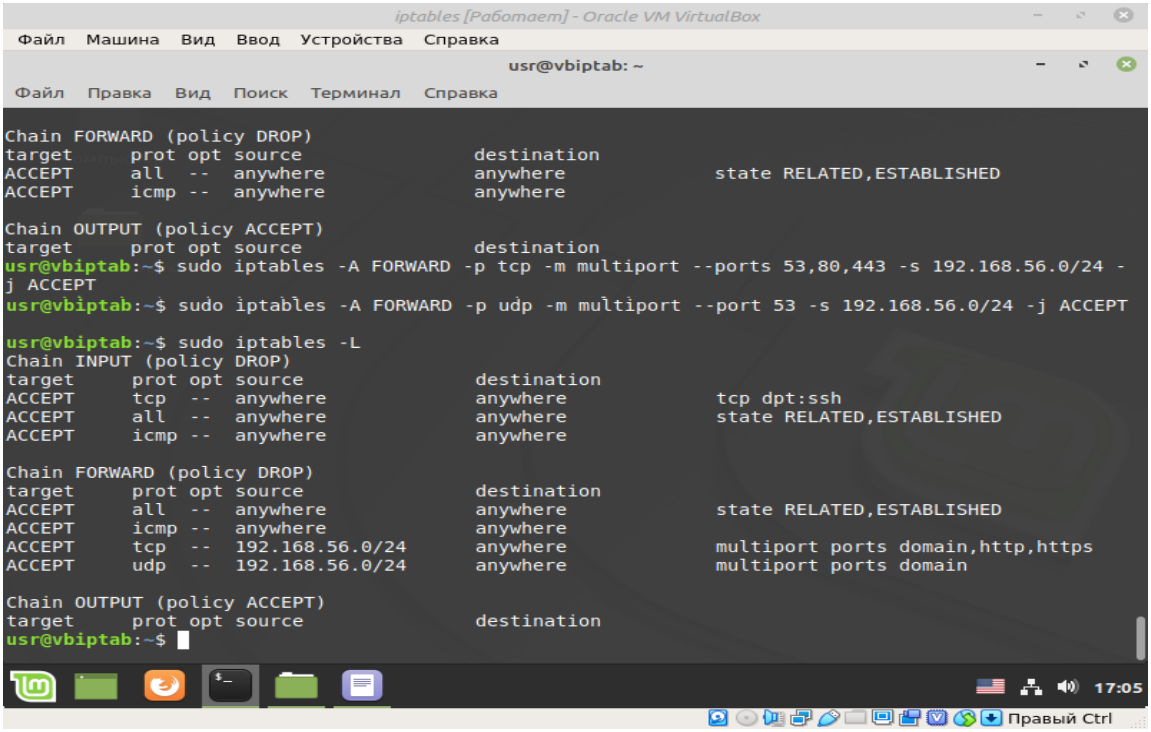
```
usr@vbiptab:~$ sudo iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
usr@vbiptab:~$
usr@vbiptab:~$ sudo iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
usr@vbiptab:~$ sudo iptables -A FORWARD -p icmp -j ACCEPT
```

3) В таблице ***nat*** создадим правило маскарадинга, позволяющее подменять наш итоговый ip-адрес ЛВС, ip-адресом ***нашего сервера*** при пересылке пакетов:

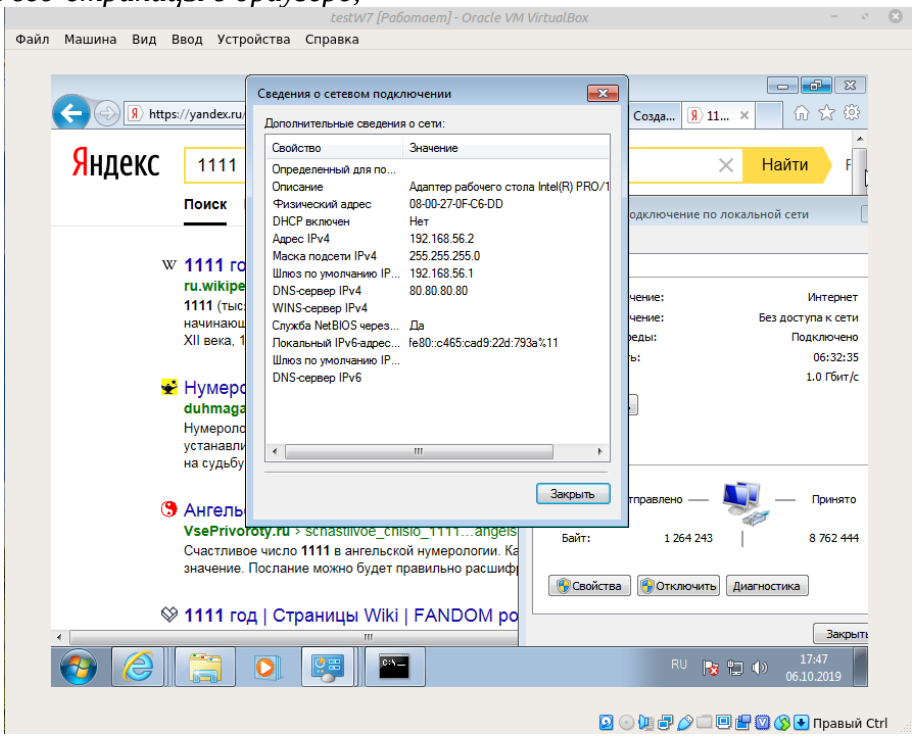
sudo iptables -t nat -A POSTROUTING -s 192.168.56.0/24 -j MASQUERADE

```
usr@vbiptab:~$ sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
MASQUERADE  all  --  192.168.56.0/24      anywhere
```

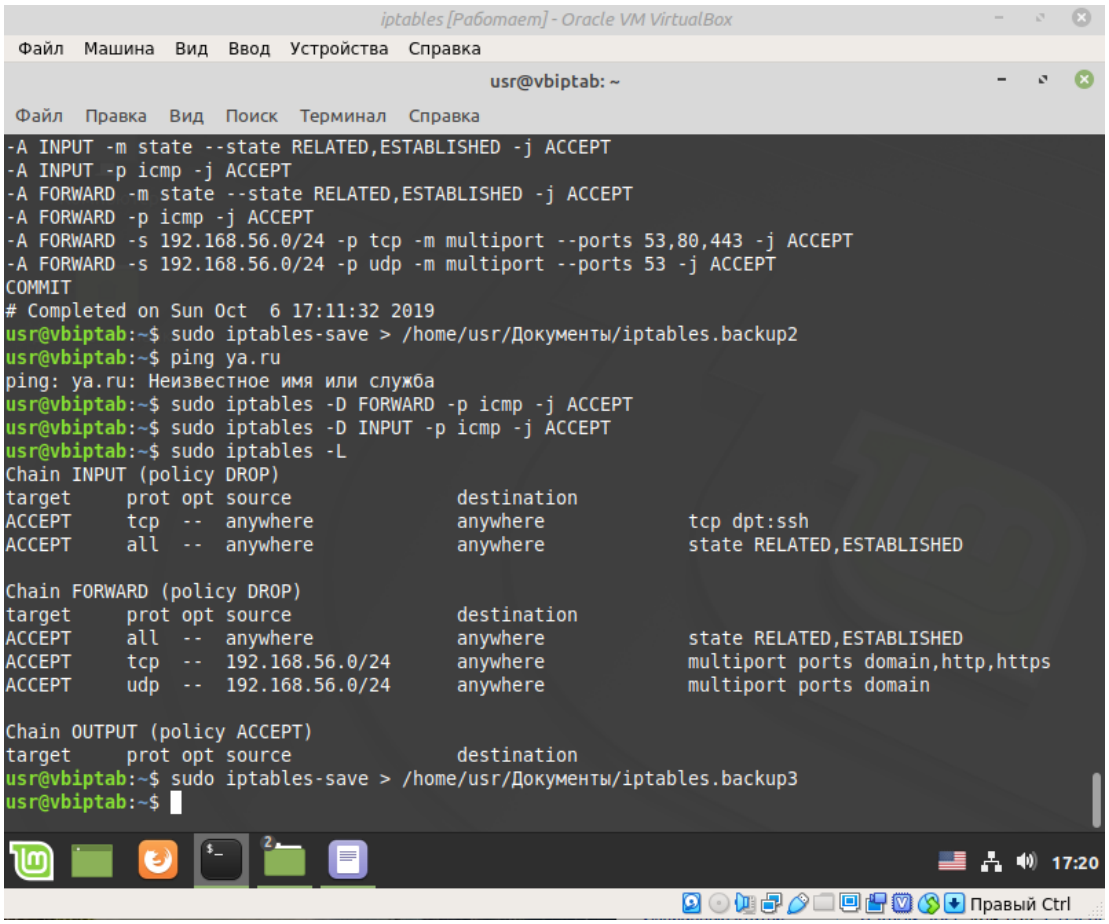
4) Разрешим пересылку DNS-запросов, а так же сервис ***HTTP(S) (SSH у нас был разрешён ещё в пункте 2 задания)***.
При этом на тестовой windows-машине я указал в качестве сервера DNS приватный DNS



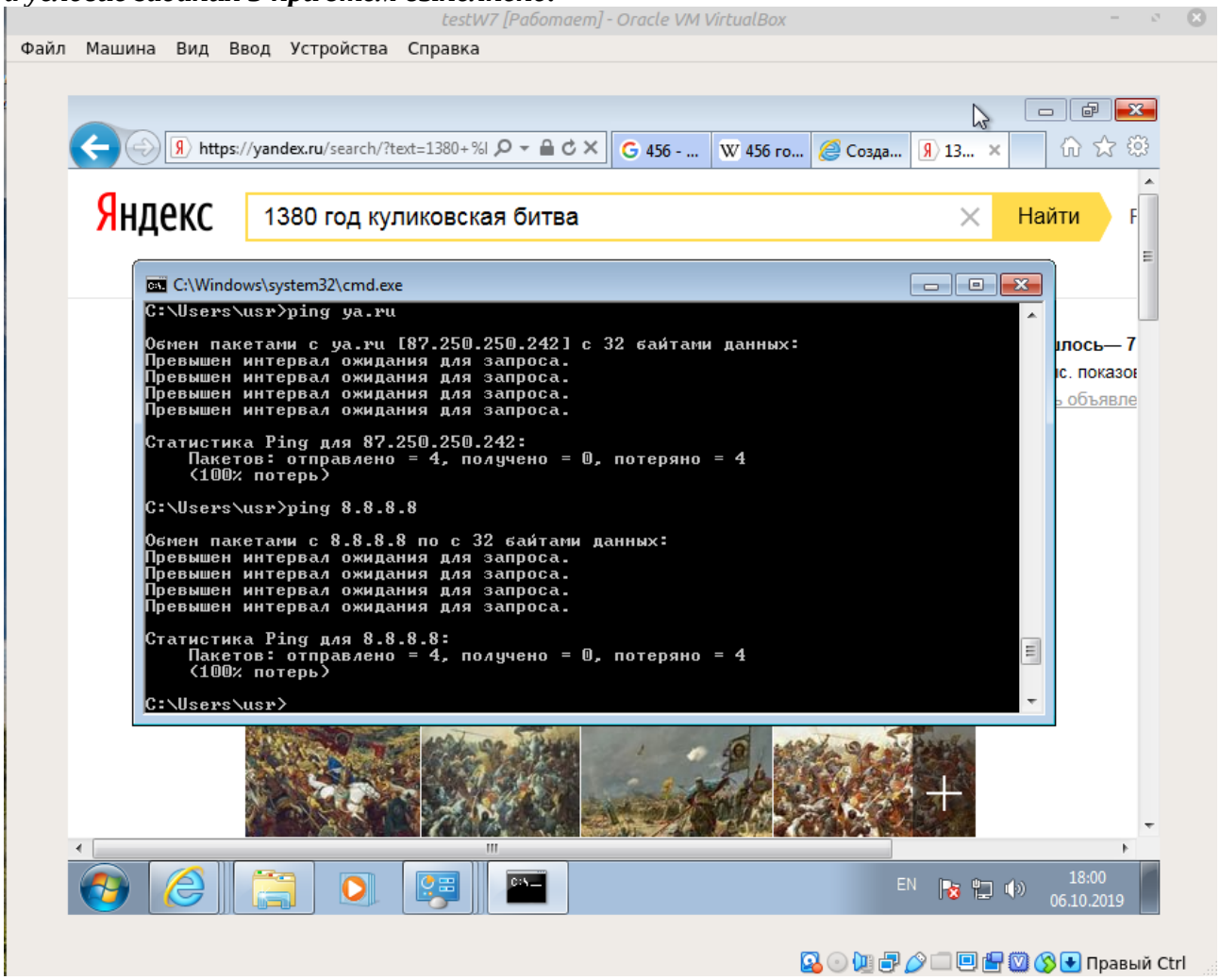
от GOOGLE; правила применились и Windows ПК начал пинговать оба интерфейса нашего интернет-сервера, веб-сайты в интернет как по ДНС-имени так и по ip-адресам, начали открываться веб-страницы в браузере,



но заметим, что в условии задания не значилось, что ПК из нашей ЛВС должны что-то пинговать как с наружи, так и наш Интернет-сервер с преднастроенным iptables, поэтому удалим эти правила, добавленные мной с целью ознакомления:



при этом пинг пропал, но страницы продолжили загружаться с прежней стабильностью, и ***условие задания 3 при этом выполнено.***



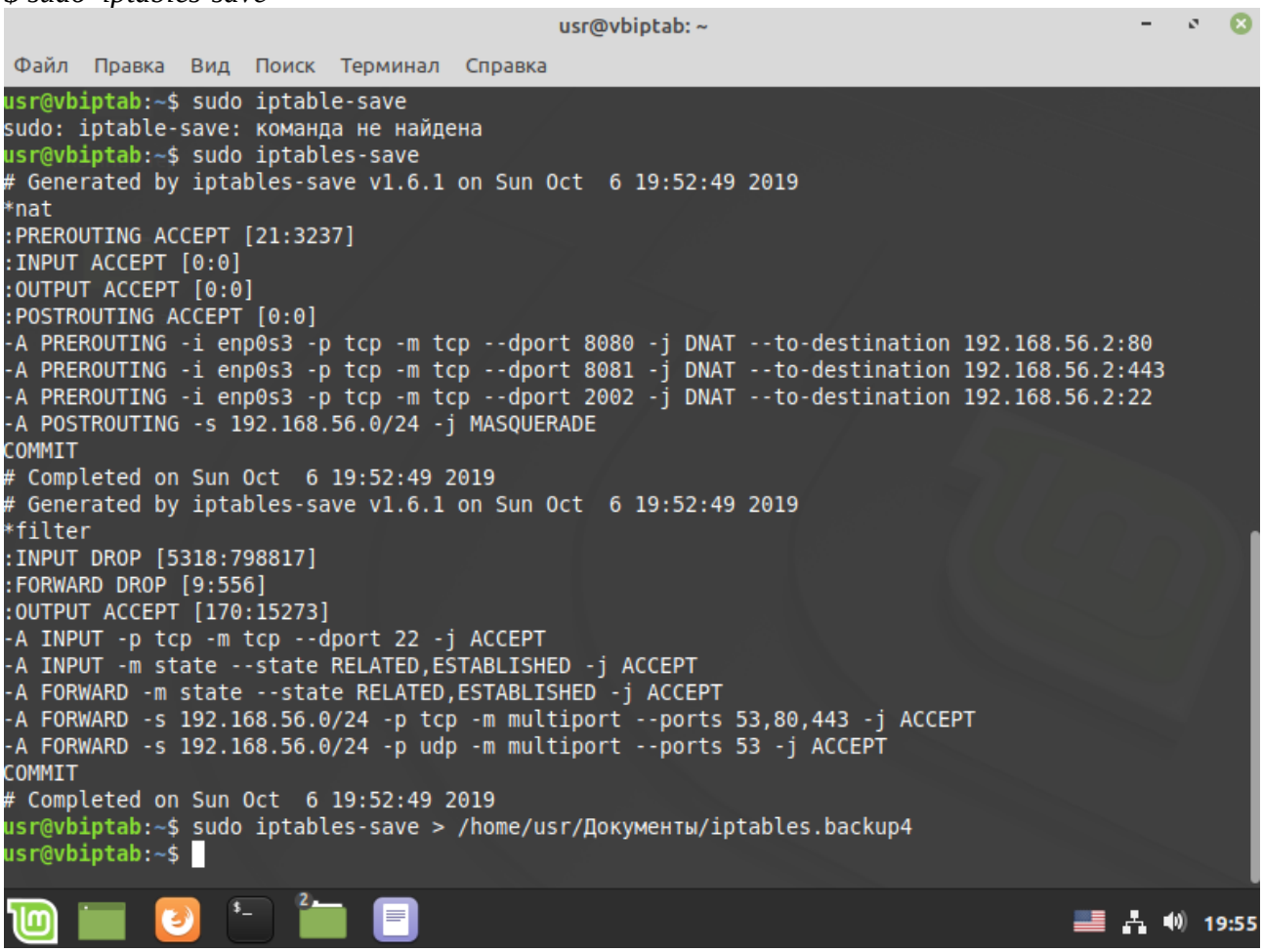
4. Настроить правила ***iptables***, чтобы из внешней сети можно было обратиться только к портам ***2002, 8080*** и ***8081***. Запросы, идущие ***на внешний порт 8080***, перенаправлять на ***внутренний порт 80***, ***запросы на порт 8081*** - перенаправлять ***на 443 порт*** и запросы на ***2002*** перенаправлять на внутренний ***22-й порт***.

DNAT: проброс портов. Согласно условию задания, понимаем, что в таблице ***FORWARD*** *должен быть открыт проброс портов во внутреннюю ЛВС на порты 22, 80, 443. А в таблице предварительной маршрутизации* ***PREROUTING,*** *создадим правила проброса портов во внутреннюю ЛВС для заданного(ых) ip-адреса(ов). В таблице* ***INPUT*** *для внешнего сетевого интерфейса (глядящего в интернет) должны быть открыты входящие tcp-порты 2002, 8080, 8081*

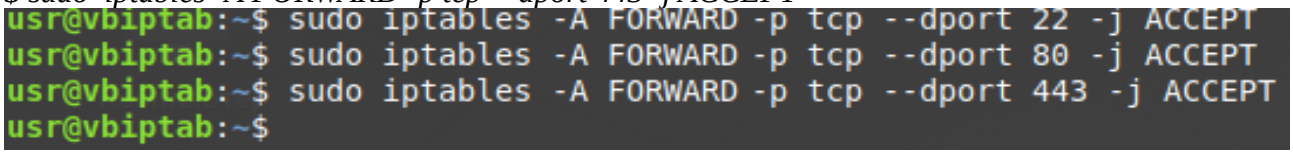
#В таблице ***nat*** *создаём(добавляем) правила предварительной маршрутизации при обращении к внешнему интерфейсу интернет-сервера* ***enp0s3*** *по протоколу* ***tcp*** *с маркером* ***tcp*** *выполнить действие по перенаправлению через* ***DNAT*** *пакетов* ***внутрь ЛВС*** *по заданным портам:*
$ sudo iptables -t nat -A PREROUTING -i enp0s3 -p tcp -m tcp —dport 8080 -j DNAT —to-destination 192.168.56.2:80
$ sudo iptables -t nat -A PREROUTING -i enp0s3 -p tcp -m tcp —dport 8081 -j DNAT —to-destination 192.168.56.2:443

$ sudo iptables -t nat -A PREROUTING -i enp0s3 -p tcp -m tcp —dport 2002 -j DNAT —to-destination 192.168.56.2:22

$ sudo iptables-save

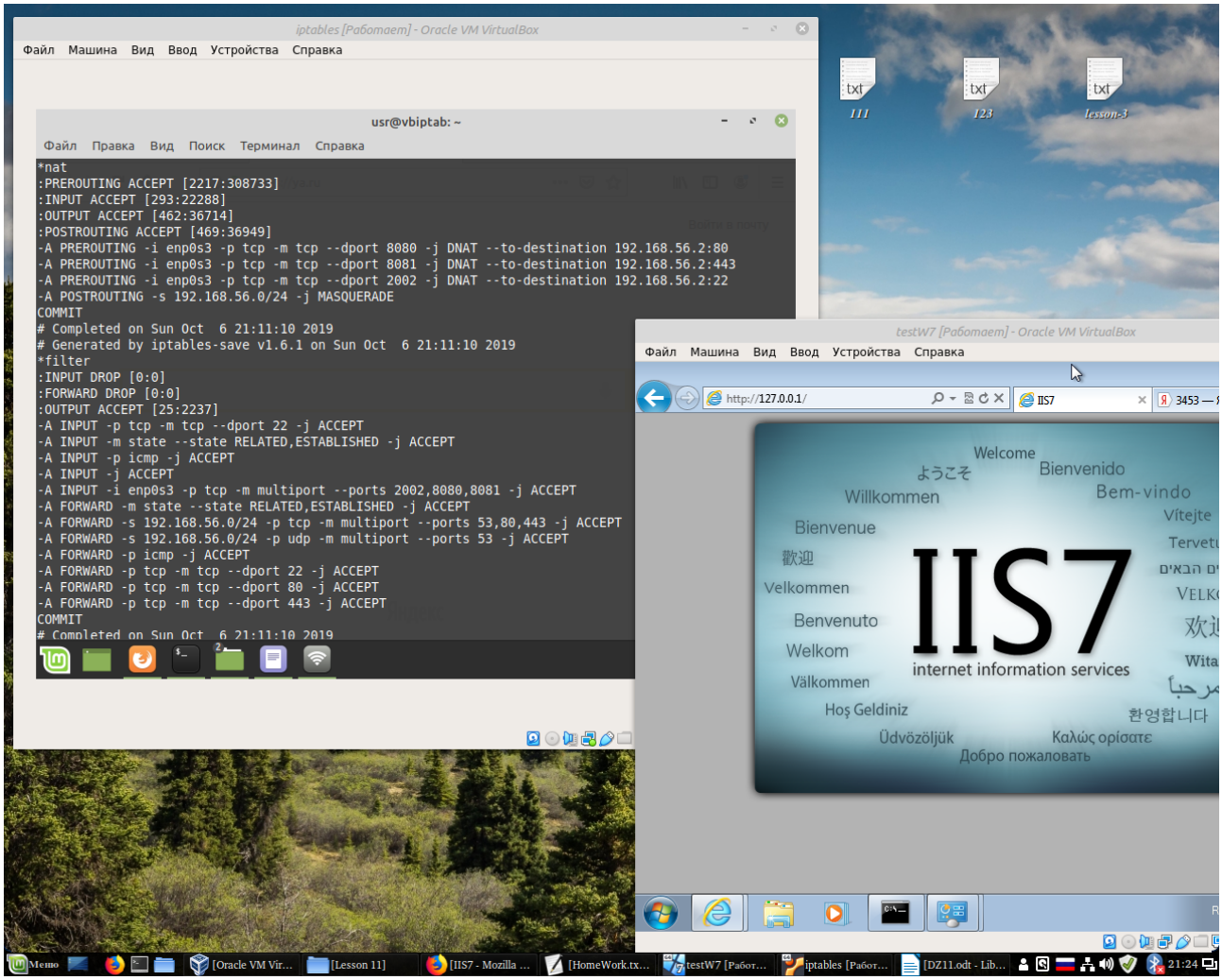


 # Разрешаем проброс портов, при обращении интернет-сервера внутрь ЛВС на внутренние порты.
$ sudo iptables -A FORWARD -p tcp —dport 22 -j ACCEPT
$ sudo iptables -A FORWARD -p tcp —dport 80 -j ACCEPT
$ sudo iptables -A FORWARD -p tcp —dport 443 -j ACCEPT

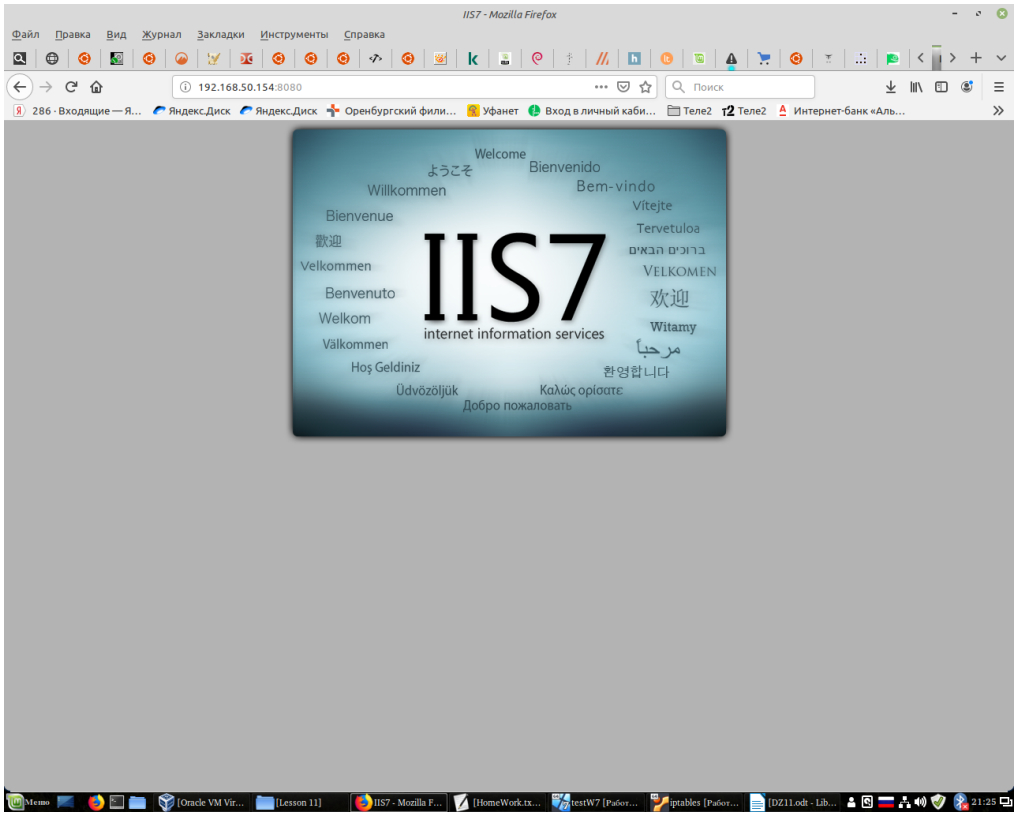


$ sudo iptables -A INPUT -p tcp -m multiport —ports 2002,8080,8081 -i enp0s3 -j ACCEPT
$ sudo iptables-save

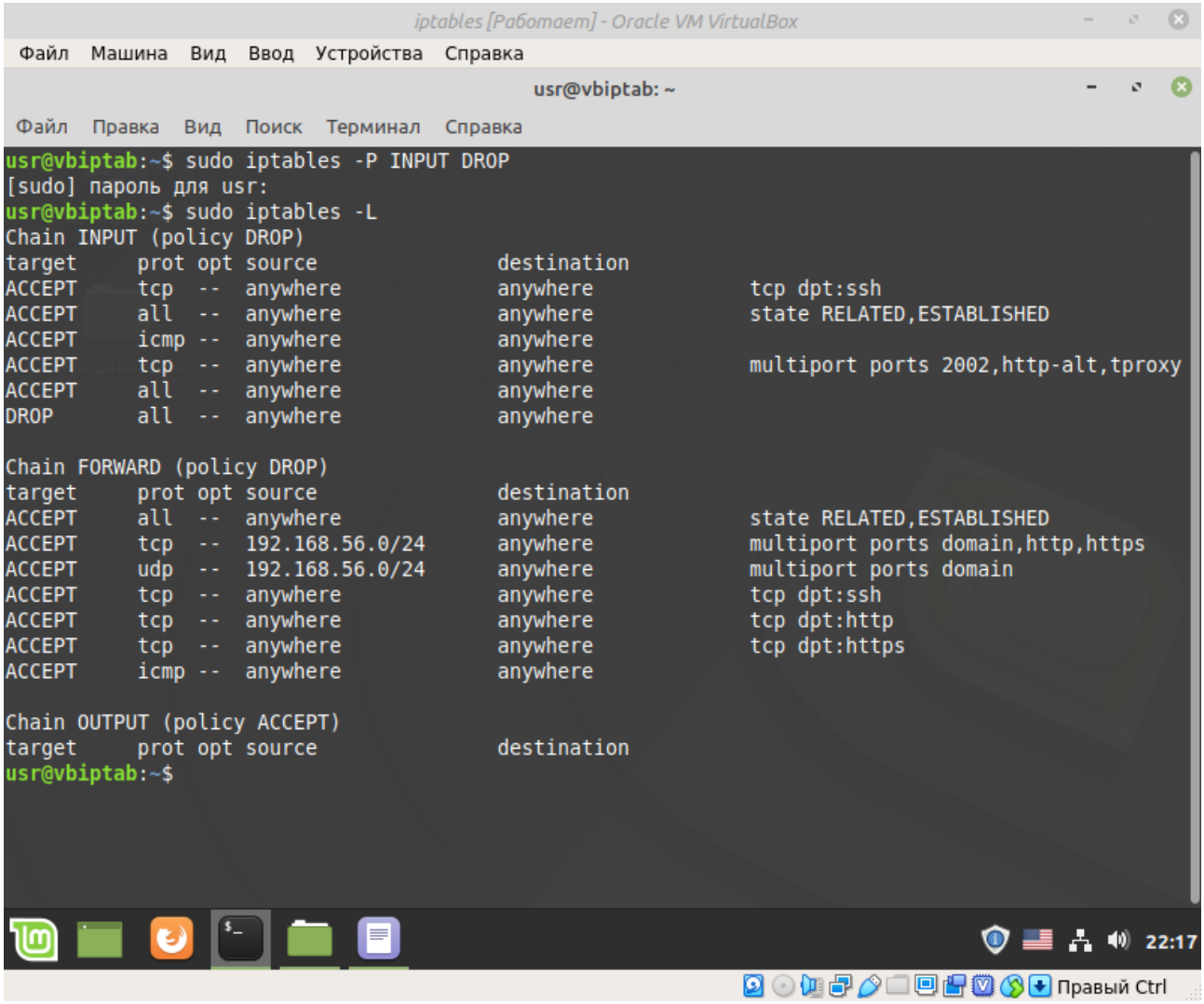
Проверяю свои действия (для проверки выбрал порт 8080, если выполнились правила для него, с остальными выполнятся аналогично):



а теперь открываю браузер хоста, и в корневой машине тоже работает:

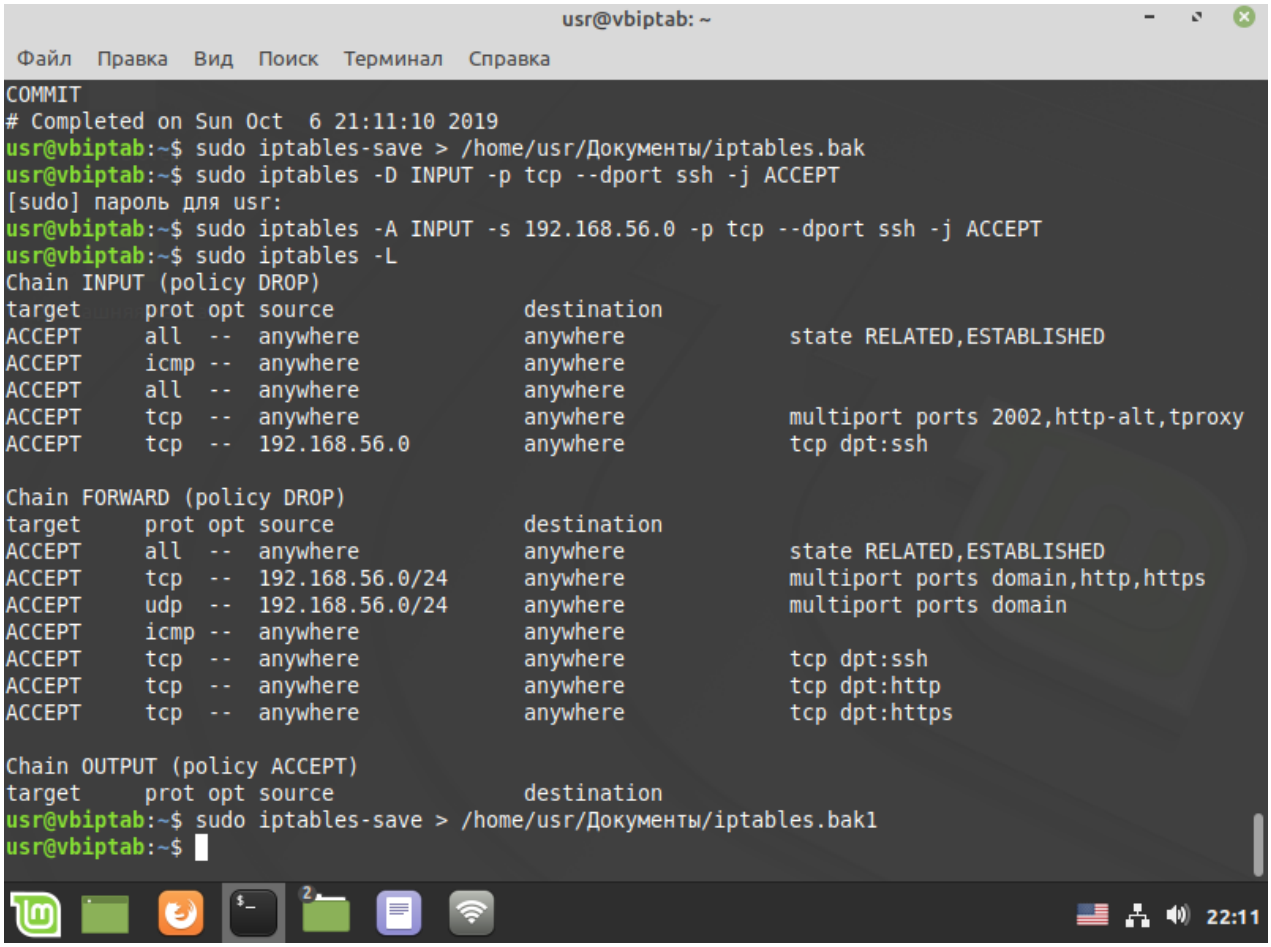


Теперь меняю политику для цепочки входящи пакетов таблицы netfilter на запрещающую:
sudo iptables -P INPUT DROP



и тоже всё работает при проверке.

5. Настроить доступ по **SSH** только **из указанного** адреса или **сетевого диапазона**.



Для реализации задания нужно удалить старое правило из iptables INPUT, для этого будем использовать ключ -D, а после пересоздадим правило согласно условию задания.