

Отчёт по лабораторной работе №1

Моделирование сетей передачи данных

Введение в Mininet

Выполнил: Махорин Иван Сергеевич,
НПИБд-02-21, 1032211221

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
2.1	Настройка образа VirtualBox	5
2.2	Подключение к виртуальной машине	6
2.3	Работа с Mininet из-под Windows	8
2.4	Настройка параметров XTerm	11
2.5	Настройка соединения X11 для суперпользователя	12
2.6	Работа с Mininet с помощью командной строки	13
2.7	Построение и эмуляция сети в Mininet с использованием графического интерфейса	17
3	Вывод	25
4	Список литературы. Библиография	26

Список иллюстраций

2.1	Установка и настройка виртуальной машины	6
2.2	Вход и просмотр адреса виртуальной машины	7
2.3	Подключение к виртуальной машине из терминала хостовой машины	8
2.4	Установка putty	9
2.5	Установка putty VcXsrv Windows X Server	9
2.6	Запуск и настройка Xserver	10
2.7	Запуск putty и добавление опции перенаправления X11	11
2.8	Увеличение размера шрифта и применение векторного шрифта .	12
2.9	Заполнения файла полномочий /root/.Xauthority	13
2.10	Вызов Mininet с использованием топологии по умолчанию	14
2.11	Отображение списка команд и примеров их использования . . .	15
2.12	Отображение доступных узлов	15
2.13	Просмотр доступных линков	16
2.14	Выполнение команды для устройства h1	16
2.15	Проверка связи между узлами h1 и h2	17
2.16	Очистка предыдущего экземпляра Mininet	17
2.17	Добавление двух хостов и одного коммутатора	18
2.18	Настройка IP-адреса на хосте h1	19
2.19	Настройка IP-адреса на хосте h2	20
2.20	Проверка назначенных IP-адресов для h2 и проверка соединения между хостами	21
2.21	Проверка автоматического назначения адресов	22
2.22	Отображение IP-адреса, назначенного хосту h1	22
2.23	Создание нового каталога	23
2.24	Сохранение топологии	23
2.25	Изменение прав доступа к файлам в каталоге проекта	24

1 Цель работы

Основной целью работы является развёртывание в системе виртуализации (например, в VirtualBox) mininet, знакомство с основными командами для работы с Mininet через командную строку и через графический интерфейс.

2 Выполнение лабораторной работы

2.1 Настройка образа VirtualBox

Для начала перейдём в репозиторий Mininet и скачаем актуальный релиз ovf-образа виртуальной машины. После чего запустим систему виртуализации и импортируем файл .ovf. Перейдём в настройки системы виртуализации и уточним параметры настройки виртуальной машины. В частности, для VirtualBox выберем импортированную виртуальную машину и перейдите в меню “Машина”-“Настроить”. Перейдём к опции «Система». Внизу этого окна есть сообщение об обнаружении неправильных настроек, следуя рекомендациям, внесём исправления. В настройках сети первый адаптер должен иметь тип подключения host-only network adapter (виртуальный адаптер хоста), который в дальнейшем мы будем использовать для входа в образ виртуальной машины. В этом режиме адаптер хоста использует специальное устройство vboxnet0, создает подсеть и назначает IP-адрес сетевой карте гостевой операционной системы. Запустим виртуальную машину с Mininet (рис. 2.1):

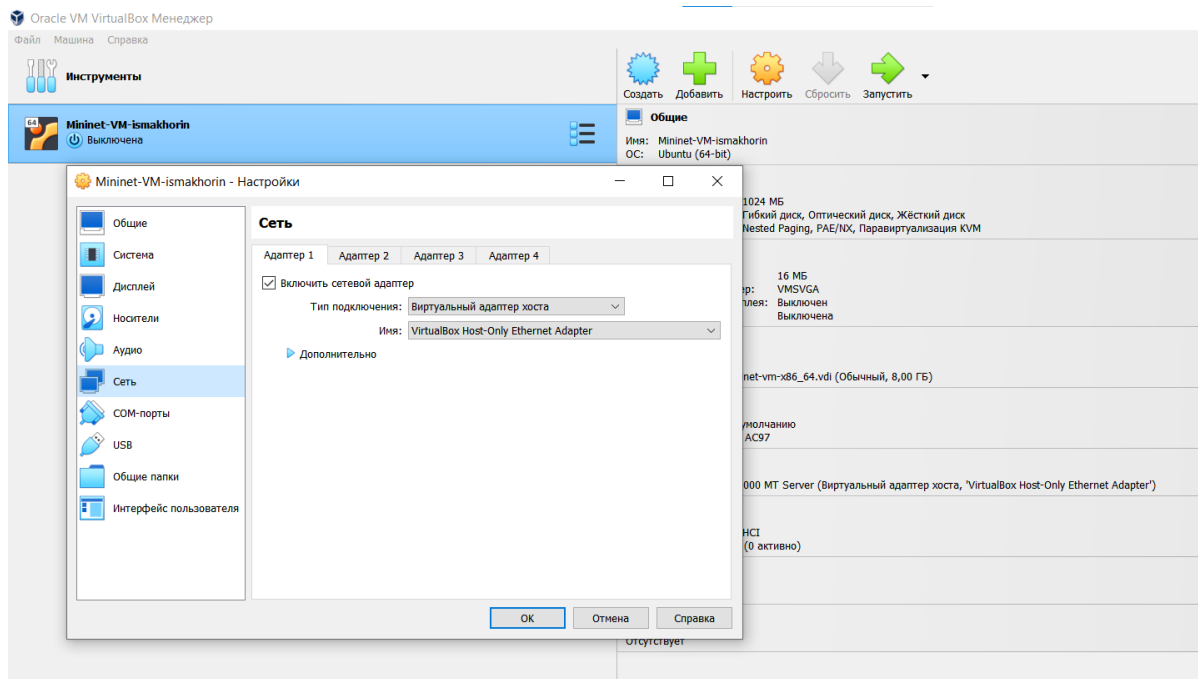


Рис. 2.1: Установка и настройка виртуальной машины

2.2 Подключение к виртуальной машине

Залогинемся в виртуальной машине и посмотрим её адрес (рис. 2.2):

```
Mininet-VM-ismakhorin [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка

Ubuntu 20.04.1 LTS mininet-vm tty1

mininet-vm login: mininet
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-42-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

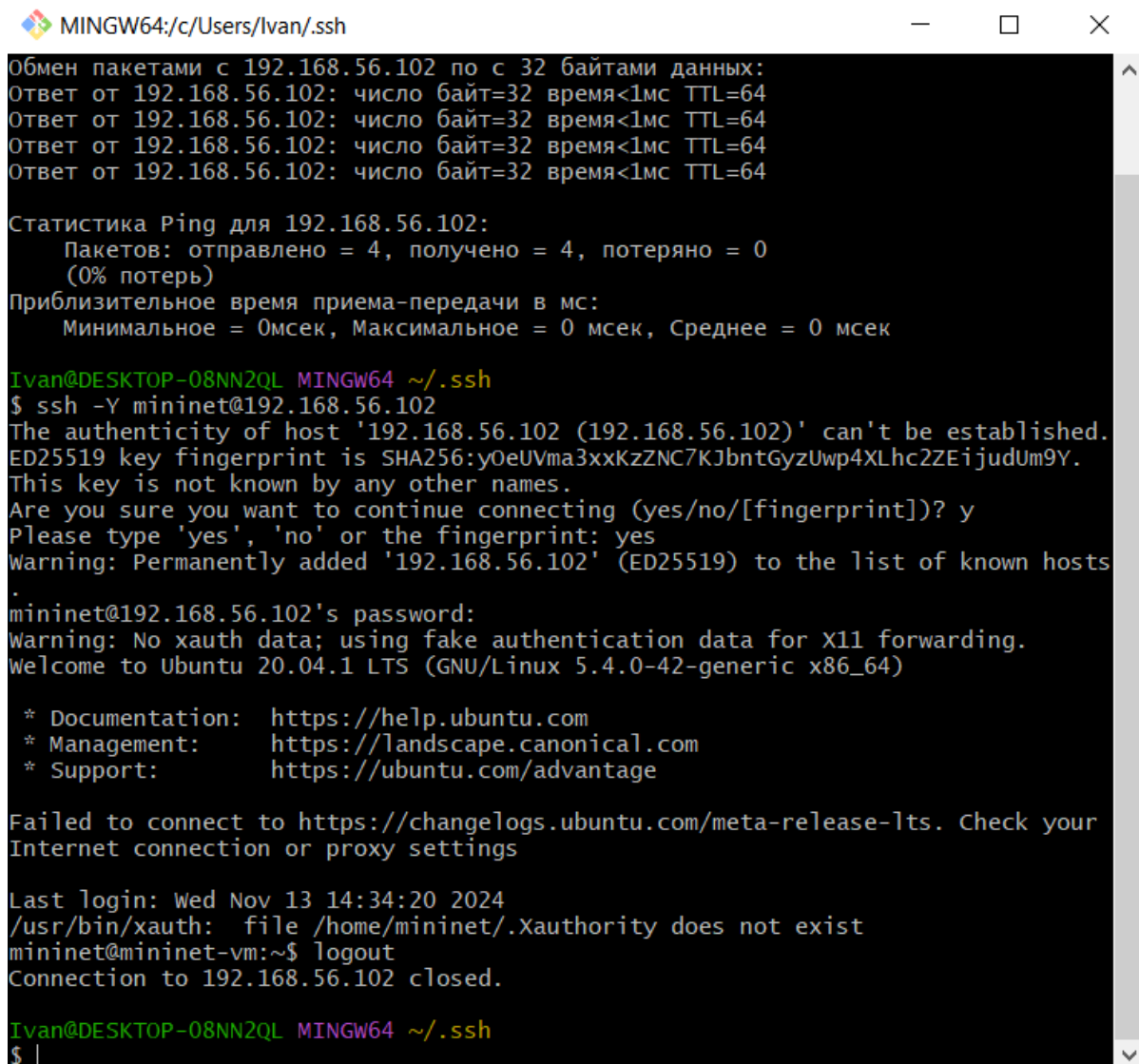
Last login: Wed Feb 10 21:03:31 PST 2021 on ttyS0
mininet@mininet-vm:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.56.102 netmask 255.255.255.0  broadcast 192.168.56.255
    ether 08:00:27:00:46:ad  txqueuelen 1000  (Ethernet)
    RX packets 2  bytes 1180 (1.1 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 2  bytes 684 (684.0 B)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop  txqueuelen 1000  (Local Loopback)
    RX packets 48  bytes 3688 (3.6 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 48  bytes 3688 (3.6 KB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

mininet@mininet-vm:~$ _
```

Рис. 2.2: Вход и просмотр адреса виртуальной машины

Внутренний адрес машины 192.168.56.102, подключимся к виртуальной машине (из терминала хостовой машины). Для отключения ssh-соединения с виртуальной машиной нажмём Ctrl + d (рис. 2.3):



```
MINGW64:/c/Users/Ivan/.ssh
Обмен пакетами с 192.168.56.102 по 32 байтами данных:
Ответ от 192.168.56.102: число байт=32 время<1мс TTL=64
Ответ от 192.168.56.102: число байт=32 время<1мс TTL=64
Ответ от 192.168.56.102: число байт=32 время<1мс TTL=64
Ответ от 192.168.56.102: число байт=32 время<1мс TTL=64

Статистика Ping для 192.168.56.102:
  Пакетов: отправлено = 4, получено = 4, потеряно = 0
  (0% потерь)
Приблизительное время приема-передачи в мс:
  Минимальное = 0мсек, Максимальное = 0 мсек, Среднее = 0 мсек

Ivan@DESKTOP-08NN2QL MINGW64 ~/.ssh
$ ssh -Y mininet@192.168.56.102
The authenticity of host '192.168.56.102 (192.168.56.102)' can't be established.
ED25519 key fingerprint is SHA256:yOeUVma3xxKzZNC7KJbntGyzUwp4XLhc2ZEijudUm9Y.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.56.102' (ED25519) to the list of known hosts
.
mininet@192.168.56.102's password:
Warning: No xauth data; using fake authentication data for X11 forwarding.
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-42-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your
Internet connection or proxy settings

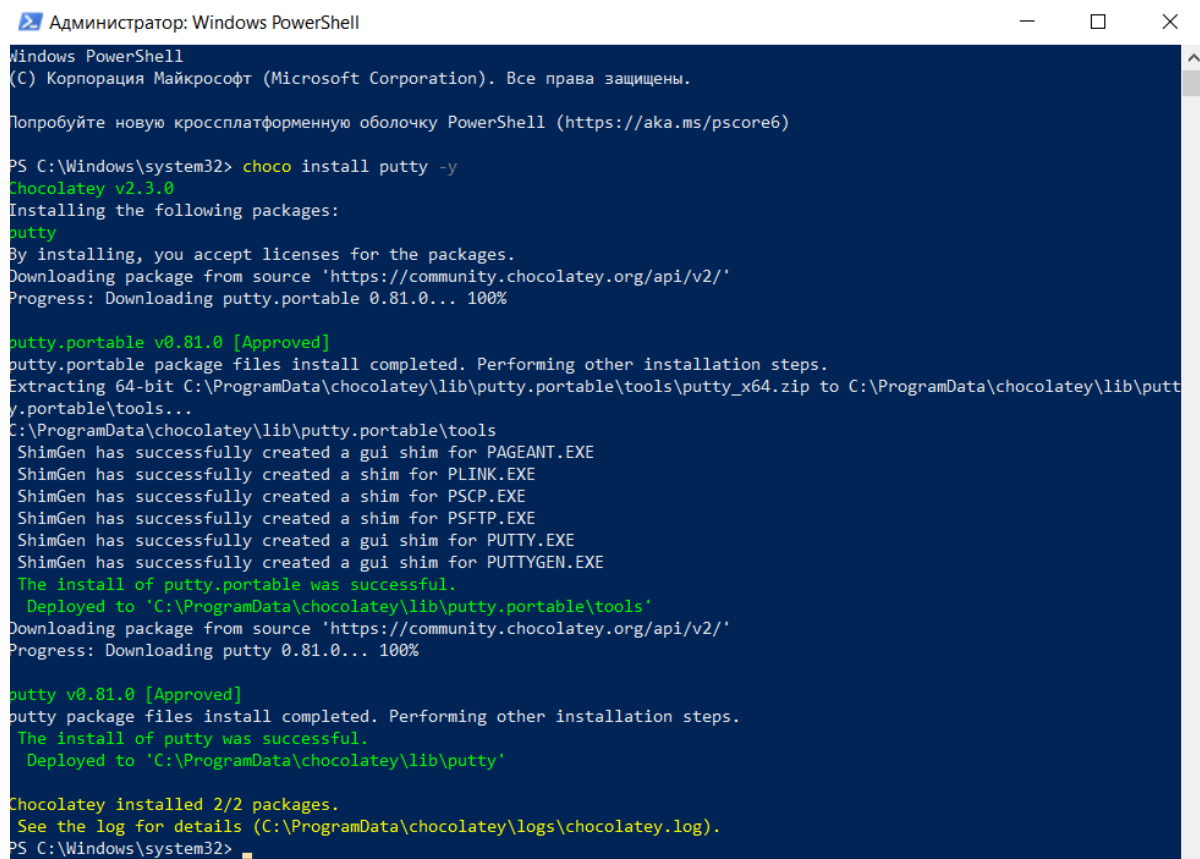
Last login: Wed Nov 13 14:34:20 2024
/usr/bin/xauth: file /home/mininet/.Xauthority does not exist
mininet@mininet-vm:~$ logout
Connection to 192.168.56.102 closed.

Ivan@DESKTOP-08NN2QL MINGW64 ~/.ssh
$
```

Рис. 2.3: Подключение к виртуальной машине из терминала хостовой машины

2.3 Работа с Mininet из-под Windows

Установим putty (рис. 2.4) и VcXsrv Windows X Server (рис. 2.5):



```
Администратор: Windows PowerShell
Windows PowerShell
(C) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

Попробуйте новую кроссплатформенную оболочку PowerShell (https://aka.ms/pscore6)

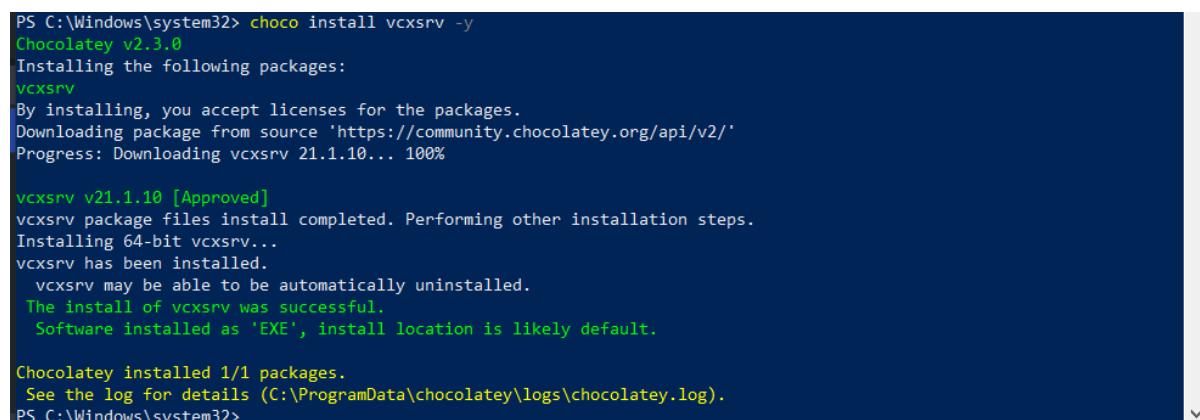
PS C:\Windows\system32> choco install putty -y
Chocolatey v2.3.0
Installing the following packages:
putty
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading putty.portable 0.81.0... 100%

putty.portable v0.81.0 [Approved]
putty.portable package files install completed. Performing other installation steps.
Extracting 64-bit C:\ProgramData\chocolatey\lib\putty.portable\tools\putty_x64.zip to C:\ProgramData\chocolatey\lib\putty.portable\tools...
C:\ProgramData\chocolatey\lib\putty.portable\tools
ShimGen has successfully created a gui shim for PAGEANT.EXE
ShimGen has successfully created a shim for PLINK.EXE
ShimGen has successfully created a shim for PSCP.EXE
ShimGen has successfully created a shim for PSFTP.EXE
ShimGen has successfully created a gui shim for PUTTY.EXE
ShimGen has successfully created a gui shim for PUTTYGEN.EXE
The install of putty.portable was successful.
Deployed to 'C:\ProgramData\chocolatey\lib\putty.portable\tools'
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading putty 0.81.0... 100%

putty v0.81.0 [Approved]
putty package files install completed. Performing other installation steps.
The install of putty was successful.
Deployed to 'C:\ProgramData\chocolatey\lib\putty'

Chocolatey installed 2/2 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\Windows\system32>
```

Рис. 2.4: Установка putty



```
PS C:\Windows\system32> choco install vcxsrv -y
Chocolatey v2.3.0
Installing the following packages:
vcxsrv
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading vcxsrv 21.1.10... 100%

vcxsrv v21.1.10 [Approved]
vcxsrv package files install completed. Performing other installation steps.
Installing 64-bit vcxsrv...
vcxsrv has been installed.
vcxsrv may be able to be automatically uninstalled.
The install of vcxsrv was successful.
Software installed as 'EXE', install location is likely default.

Chocolatey installed 1/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\Windows\system32>
```

Рис. 2.5: Установка putty VcXsrv Windows X Server

Запустим Xserver. Выберем опции: multiple windows, display number: -1, start no client. Сохраним параметры, тогда при следующем запуске не нужно будет отмечать эти опции (рис. 2.6):

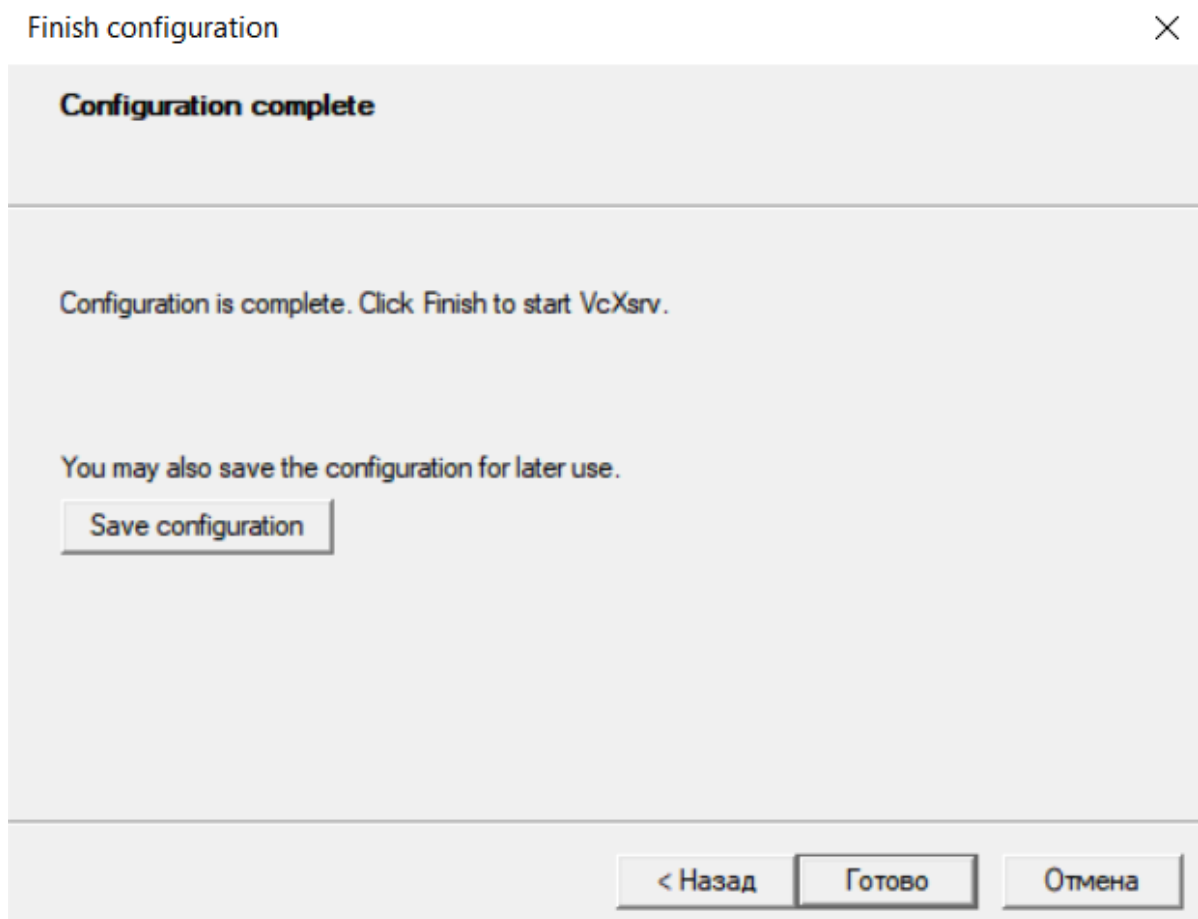


Рис. 2.6: Запуск и настройка Xserver

Запустим putty. При подключении добавим опцию перенаправления X11 (рис. 2.7):

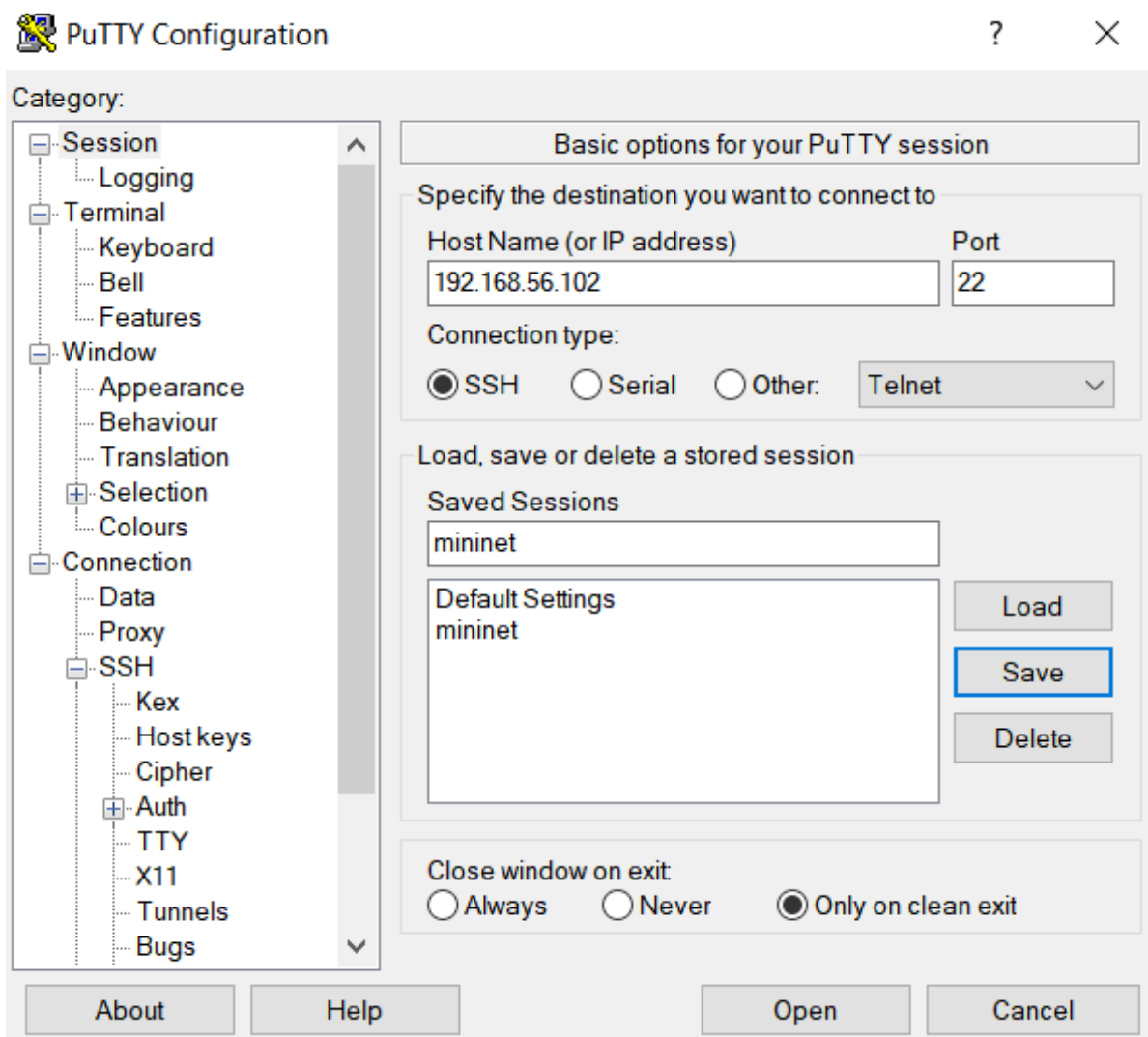


Рис. 2.7: Запуск putty и добавление опции перенаправления X11

2.4 Настройка параметров XTerm

По умолчанию XTerm использует растровые шрифты малого кегля. Для увеличения размера шрифта и применения векторных шрифтов вместо растровых необходимо внести изменения в файл `/etc/X11/app-defaults/XTerm` и в конце файла добавить нужные строки. Перед этим установим текстовый редактор `mcedit` (рис. 2.8):

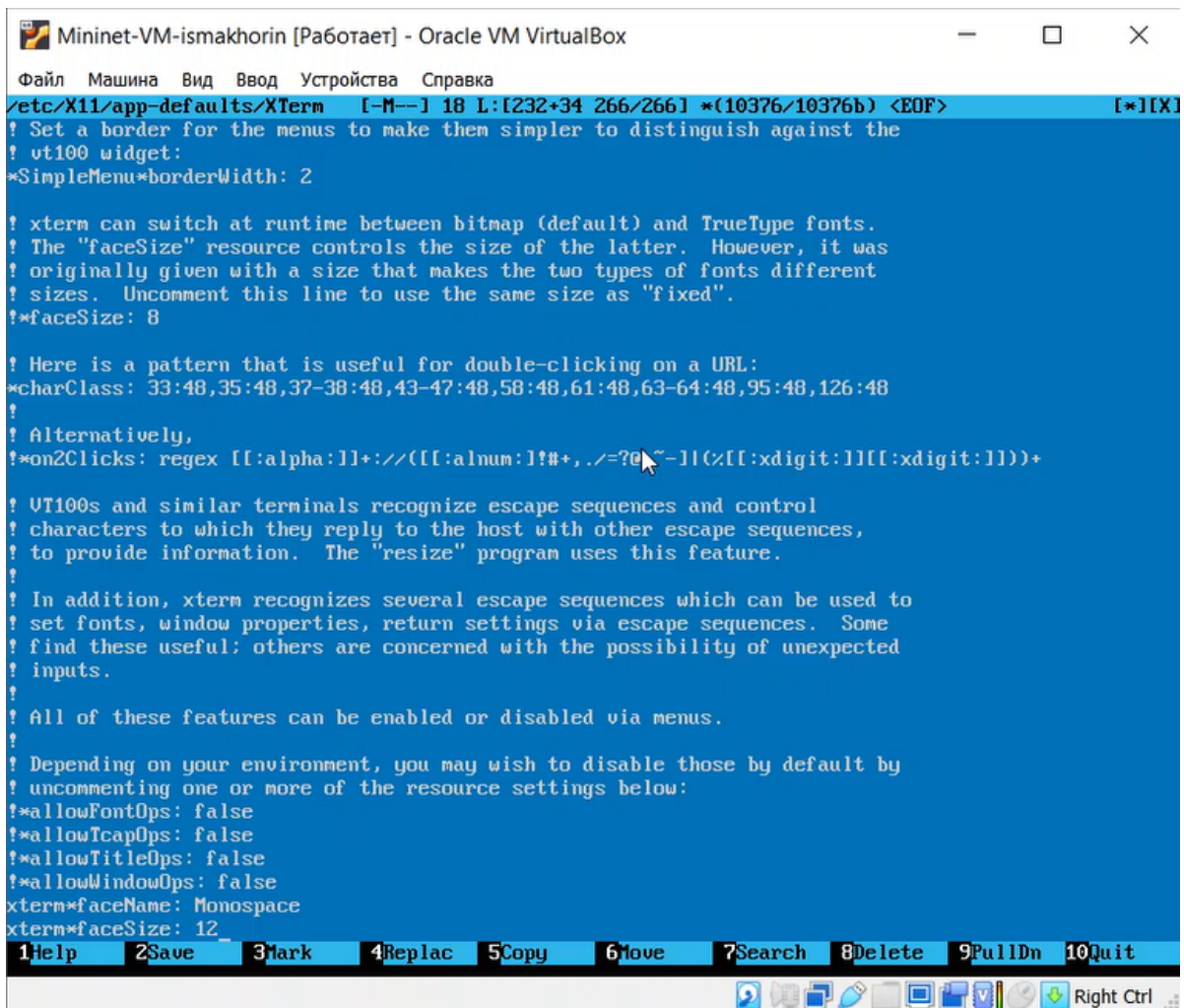
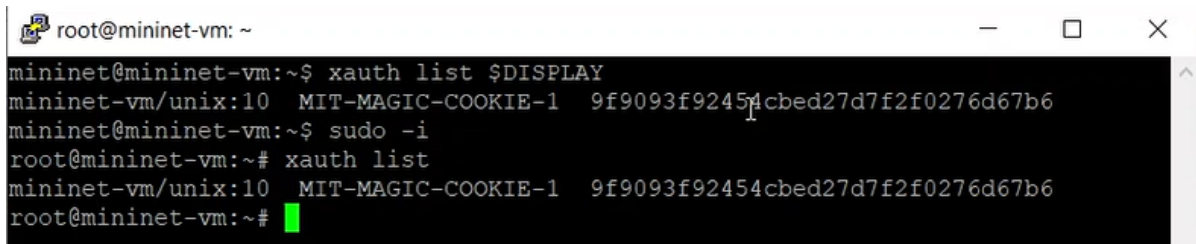


Рис. 2.8: Увеличение размера шрифта и применение векторного шрифта

2.5 Настройка соединения X11 для суперпользователя

При попытке запуска приложения из-под суперпользователя возникает ошибка. Ошибка возникает из-за того, что X-соединение выполняется от имени пользователя mininet, а приложение запускается от имени пользователя root с использованием sudo. Для исправления этой ситуации необходимо заполнить файл полномочий /root/.Xauthority, используя утилиту xauth. Скопируем значение куки (MIT magic cookie)1 пользователя mininet в файл для пользователя root (рис. 2.9):

A terminal window titled 'root@mininet-vm: ~' with standard window controls. The terminal shows a sequence of commands and their outputs. The first command is 'xauth list \$DISPLAY', which outputs 'mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 9f9093f92454cbed27d7f2f0276d67b6'. The second command is 'sudo -i', which outputs 'root@mininet-vm:~#'. The third command is 'xauth list', which outputs 'mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 9f9093f92454cbed27d7f2f0276d67b6'. The terminal ends with a red prompt character.

```
root@mininet-vm: ~
mininet@mininet-vm:~$ xauth list $DISPLAY
mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 9f9093f92454cbed27d7f2f0276d67b6
mininet@mininet-vm:~$ sudo -i
root@mininet-vm:~# xauth list
mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 9f9093f92454cbed27d7f2f0276d67b6
root@mininet-vm:~#
```

Рис. 2.9: Заполнения файла полномочий /root/.Xauthority

2.6 Работа с Mininet с помощью командной строки

Запустим минимальную топологию, состоящую из коммутатора, подключённого к двум хостам (рис. 2.10):



mininet@mininet-vm: ~

```
mininet@mininet-vm:~$ sudo mn
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> 
```

Рис. 2.10: Вызов Mininet с использованием топологии по умолчанию

Для отображения списка команд интерфейса командной строки Mininet и примеров их использования введём команду: help (рис. 2.11):

```

mininet> help

Documented commands (type help <topic>):
=====
EOF      gterm  iperfudp  nodes      pingpair    py      switch  xterm
dpctl    help   link      noecho     pingpairfull  quit    time
dump     intfs  links     pingall    ports       sh      wait
exit     iperf  net       pingallfull px          source  x

You may also send a command to a node using:
  <node> command {args}
For example:
  mininet> h1 ifconfig

The interpreter automatically substitutes IP addresses
for node names when a node is the first arg, so commands
like
  mininet> h2 ping h3
should work.

Some character-oriented interactive commands require
noecho:
  mininet> noecho h2 vi foo.py
However, starting up an xterm/gterm is generally better:
  mininet> xterm h2

mininet>

```

Рис. 2.11: Отображение списка команд и примеров их использования

Для отображения доступных узлов введём: `nodes`. Вывод этой команды показывает, что есть два хоста (хост `h1` и хост `h2`) и коммутатор (`s1`) (рис. 2.12):

```

mininet> nodes
available nodes are:
c0 h1 h2 s1
mininet>

```

Рис. 2.12: Отображение доступных узлов

Иногда бывает полезно отобразить связи между устройствами в Mininet, чтобы понять топологию. Введём команду `net` в интерфейсе командной строки Mininet, чтобы просмотреть доступные линки (рис. 2.13):

```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo: s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
mininet> █
```

Рис. 2.13: Просмотр доступных линков

Вывод этой команды показывает: - Хост h1 подключён через свой сетевой интерфейс h1-eth0 к коммутатору на интерфейсе s1-eth1. - Хост h2 подключён через свой сетевой интерфейс h2-eth0 к коммутатору на интерфейсе s1-eth2. - Коммутатор s1: - имеет петлевой интерфейс lo. - подключается к h1-eth0 через интерфейс s1-eth1. - подключается к h2-eth0 через интерфейс s1-eth2.

Mininet позволяет выполнять команды на конкретном устройстве. Чтобы выполнить команду для определенного узла, необходимо сначала указать устройство, а затем команду (рис. 2.14):

```
mininet> h1 ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.1 netmask 255.0.0.0 broadcast 10.255.255.255
    ether 62:3a:54:ac:ff:7d txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop txqueuelen 1000 (Local Loopback)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
mininet> █
```

Рис. 2.14: Выполнение команды для устройства h1

Эта запись выполняет команду ifconfig на хосте h1 и показывает интерфейсы хоста h1 — хост h1 имеет интерфейс h1-eth0, настроенный с IP-адресом 10.0.0.1, и другой интерфейс lo, настроенный с IP-адресом 127.0.0.1.

По умолчанию узлам h1 и h2 назначаются IP-адреса 10.0.0.1/8 и 10.0.0.2/8

соответственно. Чтобы проверить связь между ними, мы можем использовать команду ping (рис. 2.15):

```
mininet> h1 ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=1.87 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.173 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.071 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.048 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.422 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.042 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.039 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=0.045 ms
^C
--- 10.0.0.2 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7126ms
rtt min/avg/max/mdev = 0.039/0.338/1.866/0.590 ms
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 220.242 seconds
mininet@mininet-vm:~$
```

Рис. 2.15: Проверка связи между узлами h1 и h2

Очистим предыдущий экземпляр Mininet (рис. 2.16):

```
mininet@mininet-vm:~$ sudo mn -c
*** Removing excess controllers/ofprotocols/ofdatapaths/pings/noxes
killall controller ofprotocol ofdatapath ping nox_corelt-nox_core ovs-openflowd
ovs-controllerovs-testcontroller udpbwtest mnexec ivs ryu-manager 2> /dev/null
```

Рис. 2.16: Очистка предыдущего экземпляра Mininet

2.7 Построение и эмуляция сети в Mininet с использованием графического интерфейса

В терминале виртуальной машины mininet запустим MiniEdit: `sudo ~/mininet/mininet/examples/miniedit.py`

Добавим два хоста и один коммутатор, соединим хосты с коммутатором (рис. 2.17). Настроим IP-адреса на хостах h1 и h2. Для этого удерживая правую кнопку мыши на устройстве выберем свойства. Для хоста h1 укажем IP-адрес 10.0.0.1/8 (рис. 2.18), а для хоста h2 — 10.0.0.2/8 (рис. 2.19):

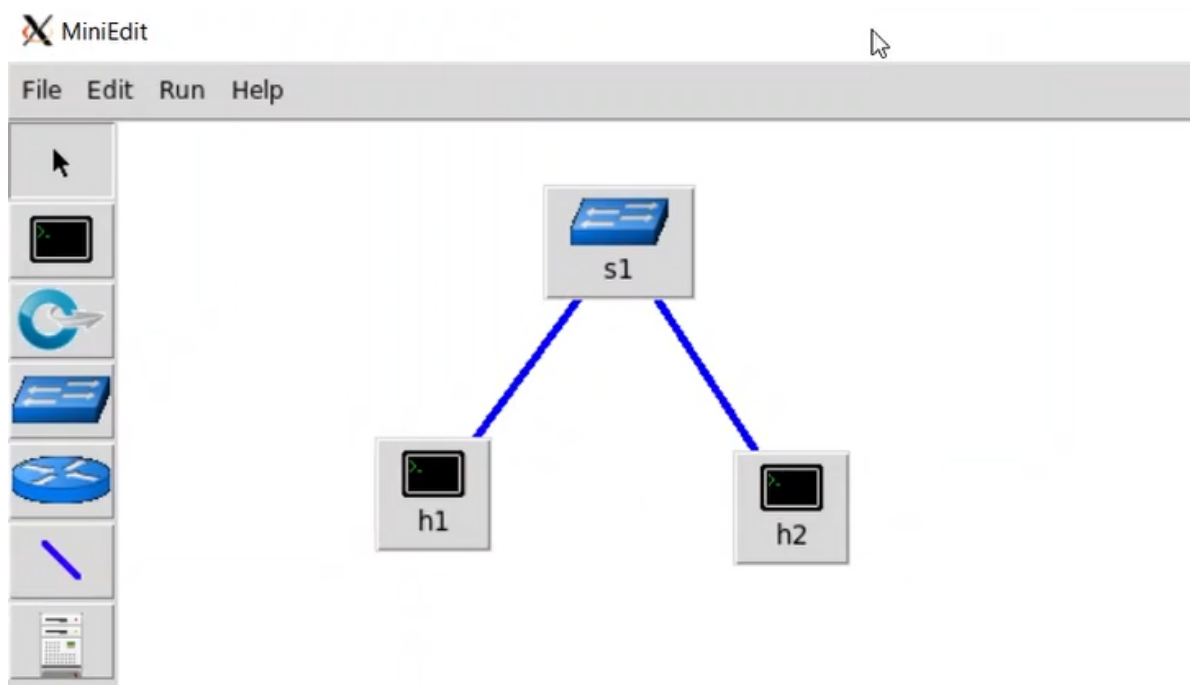


Рис. 2.17: Добавление двух хостов и одного коммутатора

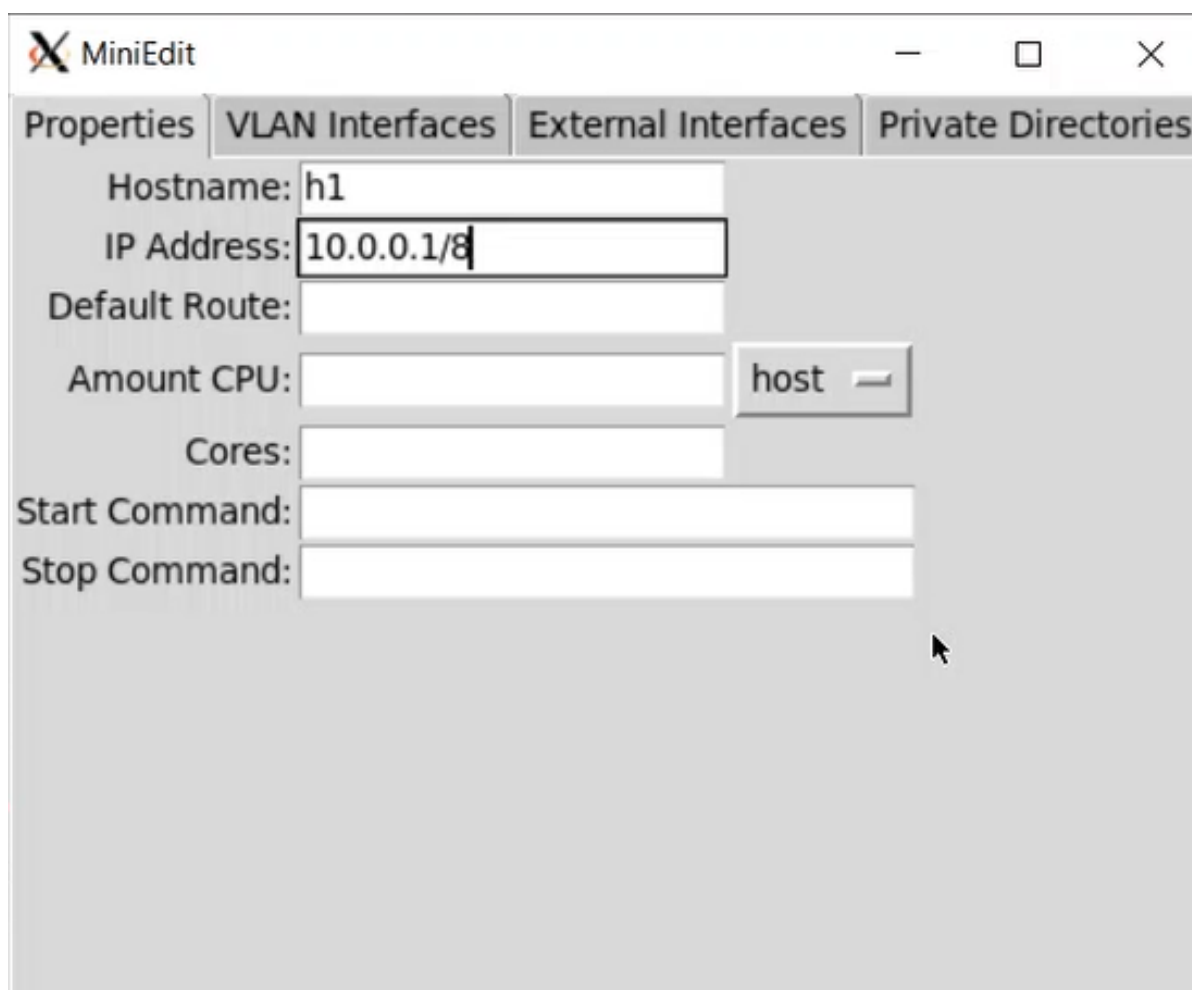


Рис. 2.18: Настройка IP-адреса на хосте h1

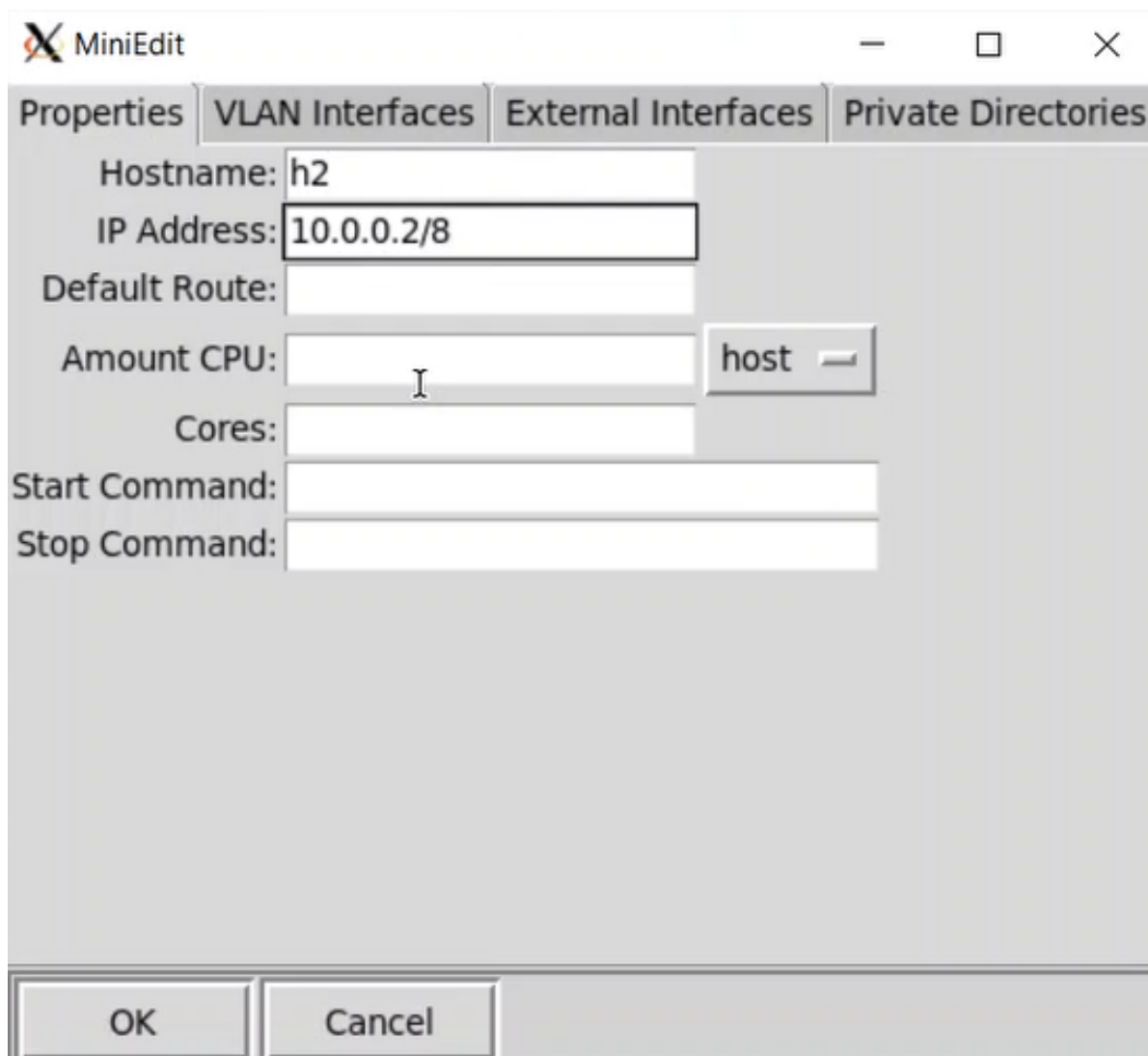
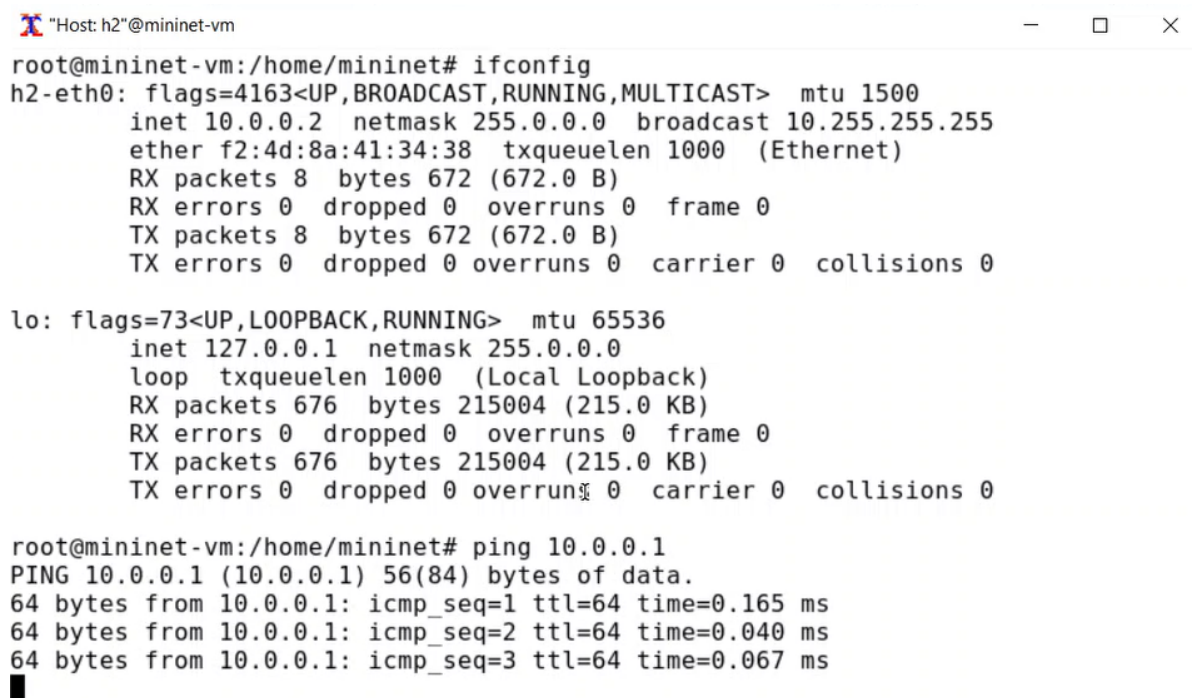


Рис. 2.19: Настройка IP-адреса на хосте h2

Перед проверкой соединения между хостом h1 и хостом h2 необходимо запустить эмуляцию. Для запуска эмуляции нажмём кнопку Run. После начала эмуляции кнопки панели MiniEdit стали серыми, указывая на то, что в настоящее время они отключены.

Откроем терминал на хосте h2. На терминале хоста h1 введём команду `ifconfig`, чтобы отобразить назначенные ему IP-адреса. Интерфейс h1-eth0 на хосте h1 настроен с IP-адресом 10.0.0.1 и маской подсети 255.0.0.0. Повторим эти действия на хосте h2. Его интерфейс h2-eth0 настроен с IP-адресом 10.0.0.2 и маской под-

сети 255.0.0.0. Проверим соединение между хостами, введя в терминале хоста h2 команду `ping 10.0.0.1`. Для остановки теста нажмём `Ctrl + c`. Остановим эмуляцию, нажав кнопку `Stop` (рис. 2.20):



```
root@mininet-vm:/home/mininet# ifconfig
h2-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.2 netmask 255.0.0.0 broadcast 10.255.255.255
    ether f2:4d:8a:41:34:38 txqueuelen 1000 (Ethernet)
    RX packets 8 bytes 672 (672.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 8 bytes 672 (672.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop txqueuelen 1000 (Local Loopback)
    RX packets 676 bytes 215004 (215.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 676 bytes 215004 (215.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.165 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.040 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.067 ms
■
```

Рис. 2.20: Проверка назначенных IP-адресов для h2 и проверка соединения между хостами

Ранее IP-адреса узлам h1 и h2 были назначены вручную. В качестве альтернативы можно полагаться на Mininet для автоматического назначения IP-адресов. Для этого удалим назначенный вручную IP-адрес с хостов h1 и h2. В MiniEdit нажмём `Edit Preferences`. По умолчанию в поле базовые значения IP-адресов (IP Base) установлено 10.0.0.0/8. Изменим это значение на 15.0.0.0/8. Затем запустим эмуляцию, нажав кнопку `Run` (рис. 2.21):

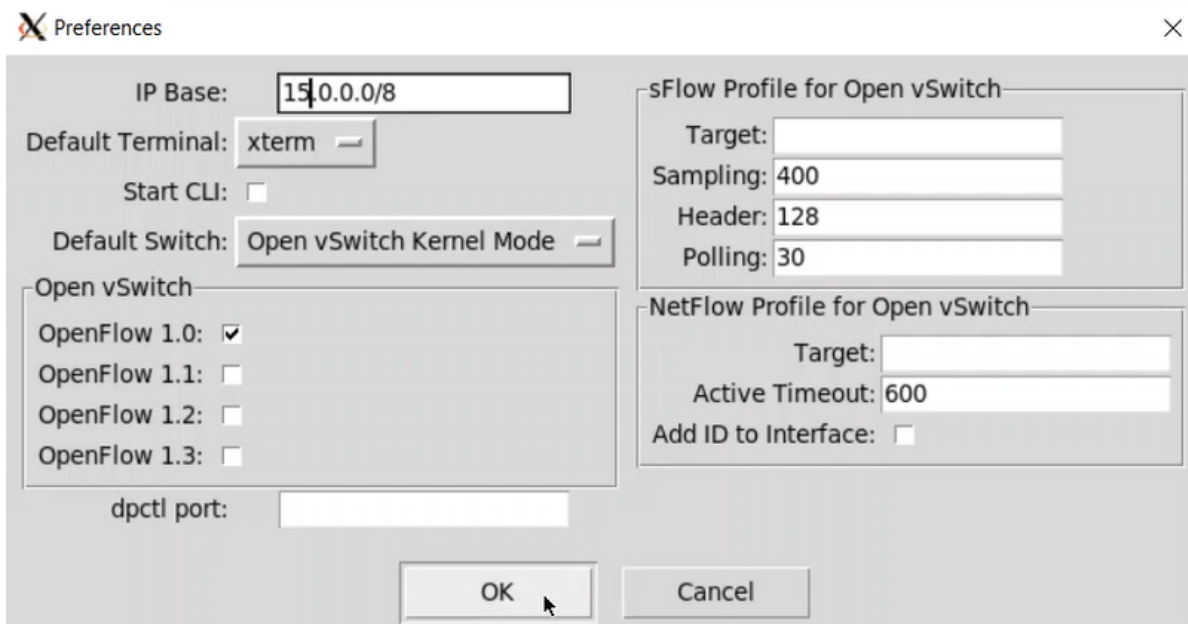


Рис. 2.21: Проверка автоматического назначения адресов

Откроем терминал на хосте h1, удерживая правую кнопку мыши на хосте h1 и выбрав Terminal. Отобразим IP-адреса, назначенные хосту h1. Интерфейс h1-eth0 на узле h1 теперь имеет IP-адрес 15.0.0.1 и маску подсети 255.0.0.0 (рис. 2.22):

```

"Host: h1"@mininet-vm
root@mininet-vm:/home/mininet# ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 15.0.0.1 netmask 255.0.0.0 broadcast 15.255.255.255
    ether 0a:2b:f9:c5:53:91 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    loop txqueuelen 1000 (Local Loopback)
    RX packets 692 bytes 215836 (215.8 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 692 bytes 215836 (215.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet#

```

Рис. 2.22: Отображение IP-адреса, назначенного хосту h1

В домашнем каталоге виртуальной машины mininet создадим каталог для

работы с проектами mininet (рис. 2.23):

```
mininet@mininet-vm:~$ mkdir ~/work  
mininet@mininet-vm:~$
```

Рис. 2.23: Создание нового каталога

Для сохранения топологии сети в файл нажмём в MiniEdit “File”-“Save”. Укажем имя для топологии и сохраним на своём компьютере (рис. 2.24):

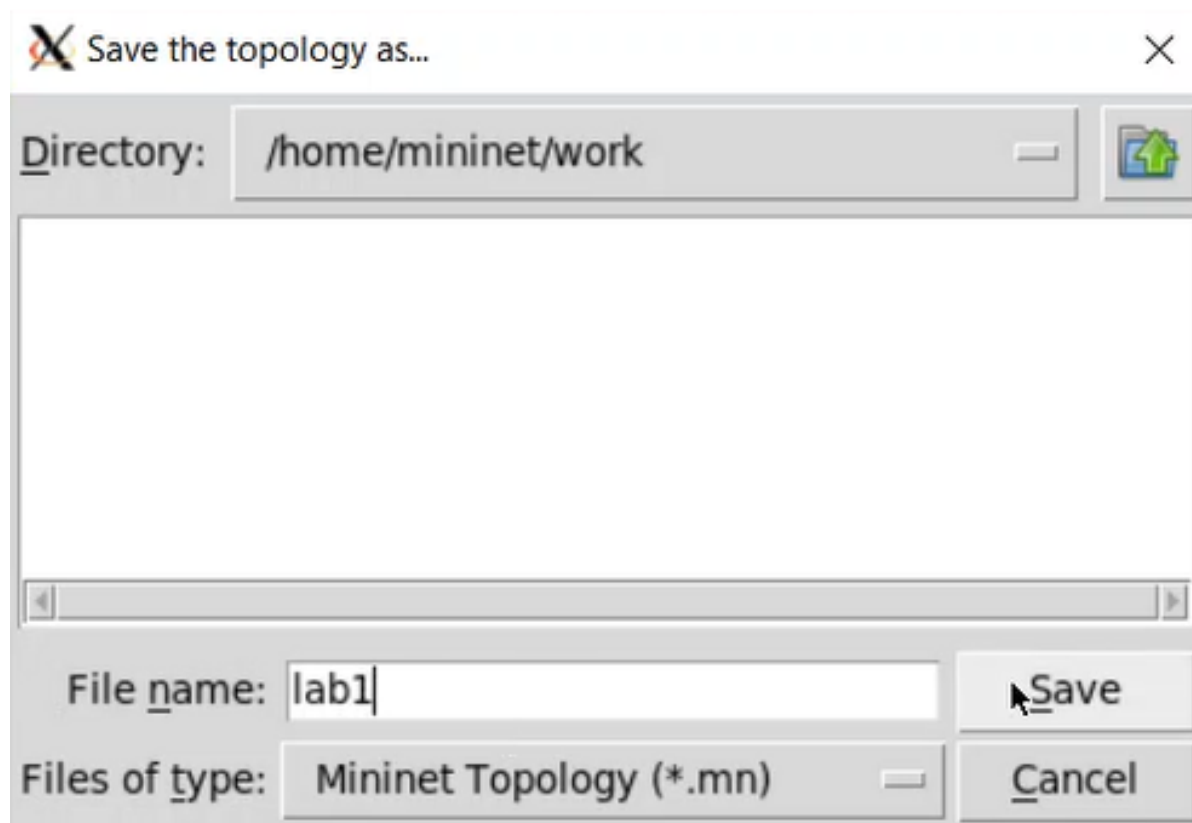


Рис. 2.24: Сохранение топологии

После сохранения проекта поменяем права доступа к файлам в каталоге проекта (рис. 2.25):

```
mininet@mininet-vm:~/work$ ls
lab1.mn
mininet@mininet-vm:~/work$ ls -Al
total 0
-rw-r--r-- 1 root root 0 Nov 13 15:27 lab1.mn
mininet@mininet-vm:~/work$ sudo chown -R mininet ~/work/
mininet@mininet-vm:~/work$
```

Рис. 2.25: Изменение прав доступа к файлам в каталоге проекта

3 Вывод

В ходе выполнения лабораторной работы были получены навыки по развёртыванию в системе виртуализации (например, в VirtualBox) mininet, а также познакомились с основными командами для работы с Mininet через командную строку и через графический интерфейс.

4 Список литературы. Библиография

[1] Mininet: <https://mininet.org/>