

Лабораторная работа №1

Дисциплина: Моделирование сетей передачи данных

Хрусталев Влад Николаевич

Содержание

1 Цель работы	4
2 Теоретическое введение	5
3 Выполнение работы	6
4 Выводы	29
Список литературы	30

Список иллюстраций

3.1 Установка и настройка виртуальной машины	7
3.2 Вход и просмотр адреса виртуальной машины	8
3.3 Подключение к виртуальной машине из терминала хостовой машины	9
3.4 Установка putty	10
3.5 Установка putty VcXsrv Windows X Server	11
3.6 Запуск и настройка Xserver Часть 1	12
3.7 Запуск и настройка Xserver Часть 2	13
3.8 Запуск и настройка Xserver Часть 3	14
3.9 Запуск и настройка Xserver Часть 4	15
3.10 Опция перенаправления X11	16
3.11 ОФайл /etc/netplan/01-netcfg.yaml	17
3.12 Обновление Mininet	17
3.13 Обновление Mininet/ВЕРСИЯ	17
3.14 Настройка шрифтов XTerm	18
3.15 Настройка соединения X11 для суперпользователя	19
3.16 Mininet с использованием топологии по умолчанию	19
3.17 Отображение результата help команды	20
3.18 Отображение доступных узлов	20
3.19 Просмотр доступных линков	21
3.20 Команда h1 ifconfig	21
3.21 Команда h1 ping 10.0.0.2	22
3.22 Очистка предыдущего экземпляра Mininet	23
3.23 Добавление двух хостов и одного коммутатора	24
3.24 Настройка IP-адреса на хосте h1 и h2	24
3.25 Проверка IP-адресов. Пинг	25
3.26 Смена базового IP-адреса	26
3.27 Просмотр IP-адреса на h1	26
3.28 Создание нового каталога	27
3.29 Сохранение топологии	27
3.30 Изменение прав доступа к файлам в каталоге проекта	28

1 Цель работы

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

2 Теоретическое введение

Mininet[1] – это эмулятор компьютерной сети. Под компьютерной сетью подразумеваются простые компьютеры – хосты, коммутаторы, а так же OpenFlow-контроллеры. С помощью простейшего синтаксиса в примитивном интерпретаторе команд можно разворачивать сети из произвольного количества хостов, коммутаторов в различных топологиях и все это в рамках одной виртуальной машины(ВМ). На всех хостах можно изменять сетевую конфигурацию, пользоваться стандартными утилитами(ifconfig, ping) и даже получать доступ к терминалу. На коммутаторы можно добавлять различные правила и маршрутизировать трафик.

3 Выполнение работы

1. Перейдем в репозиторий Mininet, скачаем актуальный релиз ovf-образа виртуальной машины. Запустим систему виртуализации и импортируем файл .ovf и укажем параметры импорт
2. Перейдем в настройки системы виртуализации и уточним параметры настройки виртуальной машины. В частности, для VirtualBox выберем импортированную виртуальную машину и перейдите в меню “Машина -> Настроить”.
3. Перейдем к опции «Система». Если внизу этого окна есть сообщение об обнаружении неправильных настроек, то, следуя рекомендациям, внесем исправления (изменим тип графического контроллера на рекомендуемый).
4. В настройках сети первый адаптер должен иметь подключение типа NAT, настроено по умолчанию. Для второго адаптера укажем тип подключения host-only network adapter (виртуальный адаптер хоста), который в дальнейшем мы будем использовать для входа в образ виртуальной машины.(рис. 3.1):

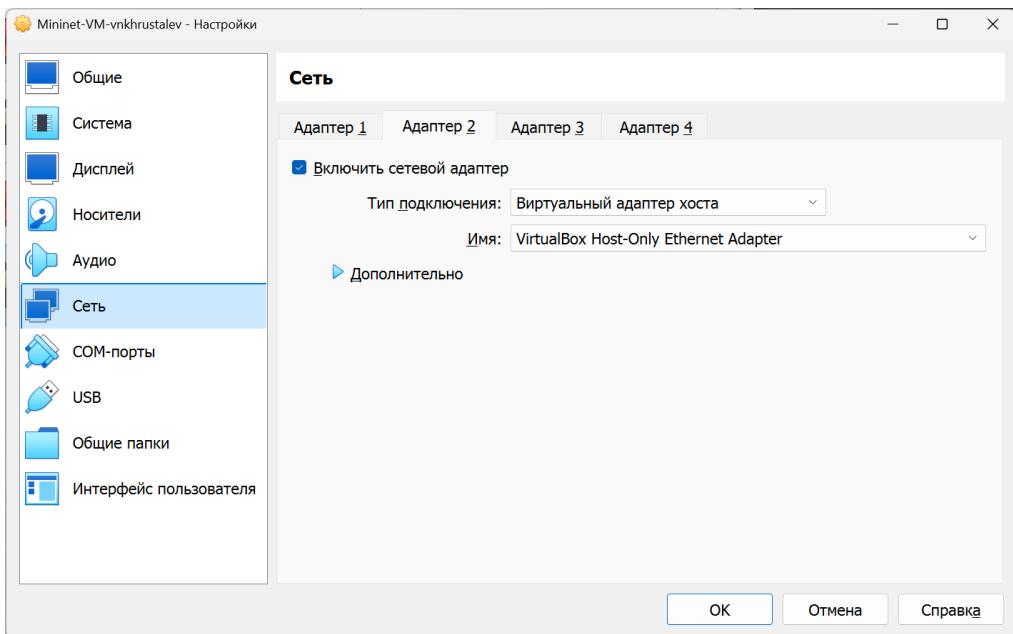


Рис. 3.1: Установка и настройка виртуальной машины

Запустим виртуальную машину с Mininet. Залогинимся в виртуальную машину:

- login: mininet - password: mininet

Посмотрим адрес машины с помощью `ifconfig` (рис. 3.2):

```
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  
  
New release '22.04.5 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Sat Nov 29 09:42:35 PST 2025 from 192.168.56.1 on pts/3  
mininet@mininet-vm:~$ ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
      inet 192.168.56.105 netmask 255.255.255.0 broadcast 192.168.56.255  
        ether 08:00:27:e8:b9 txqueuelen 1000 (Ethernet)  
          RX packets 5 bytes 1379 (1.3 KB)  
          RX errors 0 dropped 0 overruns 0 frame 0  
          TX packets 16 bytes 2084 (2.0 KB)  
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
      inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255  
        ether 08:00:27:d0:e5:b3 txqueuelen 1000 (Ethernet)  
          RX packets 207 bytes 22603 (22.6 KB)  
          RX errors 0 dropped 0 overruns 0 frame 0  
          TX packets 229 bytes 21087 (21.0 KB)  
          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 52 bytes 4518 (4.5 KB)  
          RX errors 0 dropped 0 overruns 0 frame 0  
          TX packets 52 bytes 4518 (4.5 KB)  
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
mininet@mininet-vm:~$
```

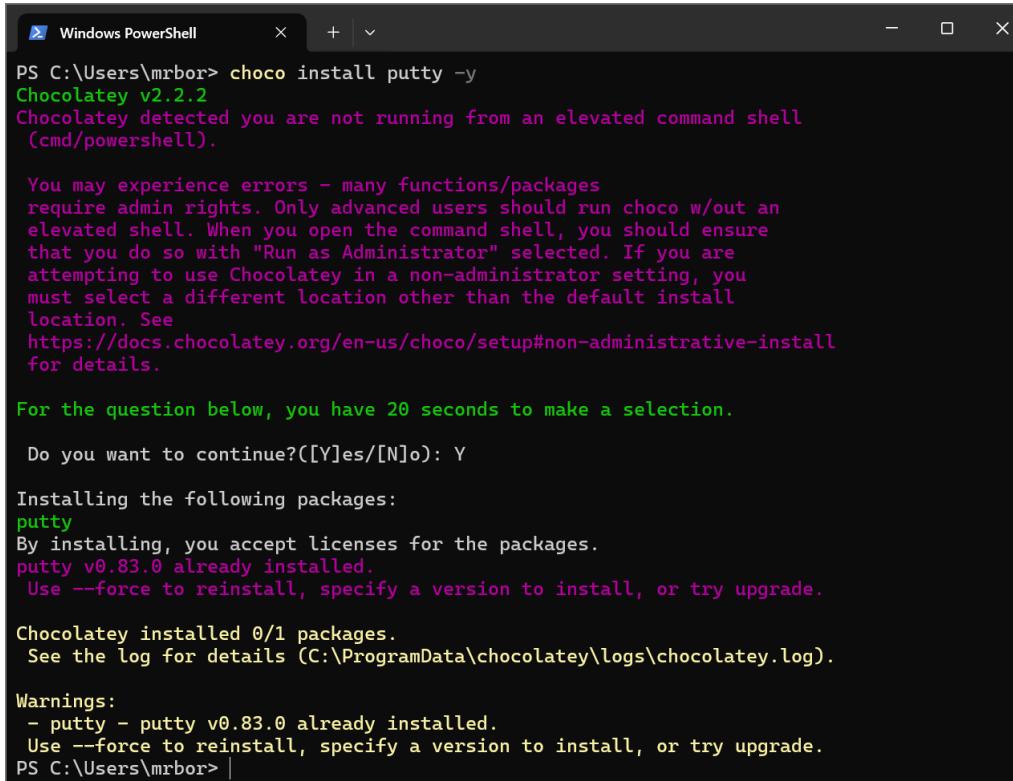
Рис. 3.2: Вход и просмотр адреса виртуальной машины

Подключимся к виртуальной машине (из терминала хостовой машины). Для отключения ssh-соединения с виртуальной машиной нажмём Ctrl + d (рис. 3.3):

```
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  
  
New release '22.04.5 LTS' available.  
Run 'do-release-upgrade' to upgrade to it.  
  
Last login: Sat Nov 29 09:42:35 PST 2025 from 192.168.56.1 on pts/3  
mininet@mininet-vm:~$ ifconfig  
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
      inet 192.168.56.105 netmask 255.255.255.0 broadcast 192.168.56.255  
        ether 08:00:27:e8:b9 txqueuelen 1000 (Ethernet)  
          RX packets 5 bytes 1379 (1.3 KB)  
          RX errors 0 dropped 0 overruns 0 frame 0  
          TX packets 16 bytes 2084 (2.0 KB)  
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
      inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255  
        ether 08:00:27:d0:e5:b3 txqueuelen 1000 (Ethernet)  
          RX packets 207 bytes 22603 (22.6 KB)  
          RX errors 0 dropped 0 overruns 0 frame 0  
          TX packets 229 bytes 21087 (21.0 KB)  
          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 52 bytes 4518 (4.5 KB)  
          RX errors 0 dropped 0 overruns 0 frame 0  
          TX packets 52 bytes 4518 (4.5 KB)  
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
mininet@mininet-vm:~$
```

Рис. 3.3: Подключение к виртуальной машине из терминала хостовой машины

Установим putty (рис. 3.4) и VcXsrv Windows X Server (рис. 3.5):



```
PS C:\Users\mrbor> choco install putty -y
Chocolatey v2.2.2
Chocolatey detected you are not running from an elevated command shell
(cmd/powershell).

You may experience errors - many functions/packages
require admin rights. Only advanced users should run choco w/out an
elevated shell. When you open the command shell, you should ensure
that you do so with "Run as Administrator" selected. If you are
attempting to use Chocolatey in a non-administrator setting, you
must select a different location other than the default install
location. See
https://docs.chocolatey.org/en-us/choco/setup#non-administrative-install
for details.

For the question below, you have 20 seconds to make a selection.

Do you want to continue?([Y]es/[N]o): Y

Installing the following packages:
putty
By installing, you accept licenses for the packages.
putty v0.83.0 already installed.
Use --force to reinstall, specify a version to install, or try upgrade.

Chocolatey installed 0/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).

Warnings:
- putty - putty v0.83.0 already installed.
Use --force to reinstall, specify a version to install, or try upgrade.
PS C:\Users\mrbor> |
```

Рис. 3.4: Установка putty

```
PS C:\Users\mrbor> choco install vcxsrv -y
Chocolatey v2.2.2
Chocolatey detected you are not running from an elevated command shell
(cmd/powershell).

You may experience errors - many functions/packages
require admin rights. Only advanced users should run choco w/out an
elevated shell. When you open the command shell, you should ensure
that you do so with "Run as Administrator" selected. If you are
attempting to use Chocolatey in a non-administrator setting, you
must select a different location other than the default install
location. See
https://docs.chocolatey.org/en-us/choco/setup#non-administrative-install
for details.

For the question below, you have 20 seconds to make a selection.

Do you want to continue?([Y]es/[N]o): Y

Installing the following packages:
vcxsrv
By installing, you accept licenses for the packages.
vcxsrv v21.1.10 already installed.
Use --force to reinstall, specify a version to install, or try upgrade.

Chocolatey installed 0/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).

Warnings:
- vcxsrv - vcxsrv v21.1.10 already installed.
Use --force to reinstall, specify a version to install, or try upgrade.
PS C:\Users\mrbor> |
```

Рис. 3.5: 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. 3.6; 3.7; 3.8; 3.9):

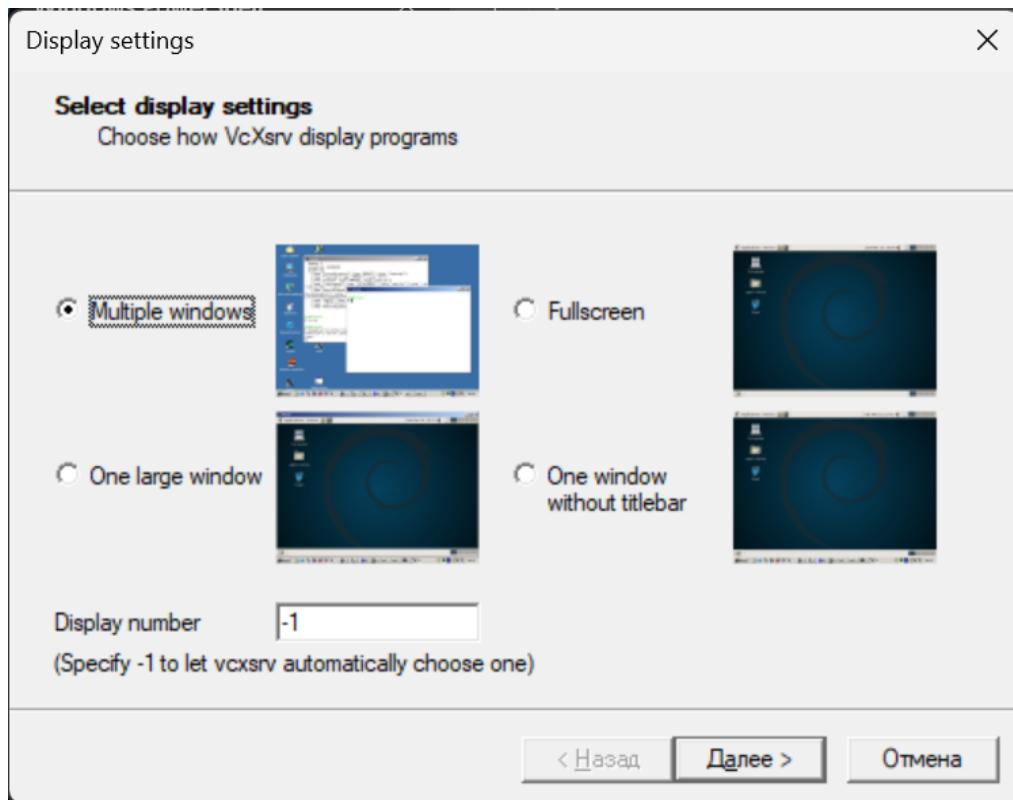


Рис. 3.6: Запуск и настройка Xserver | Часть 1

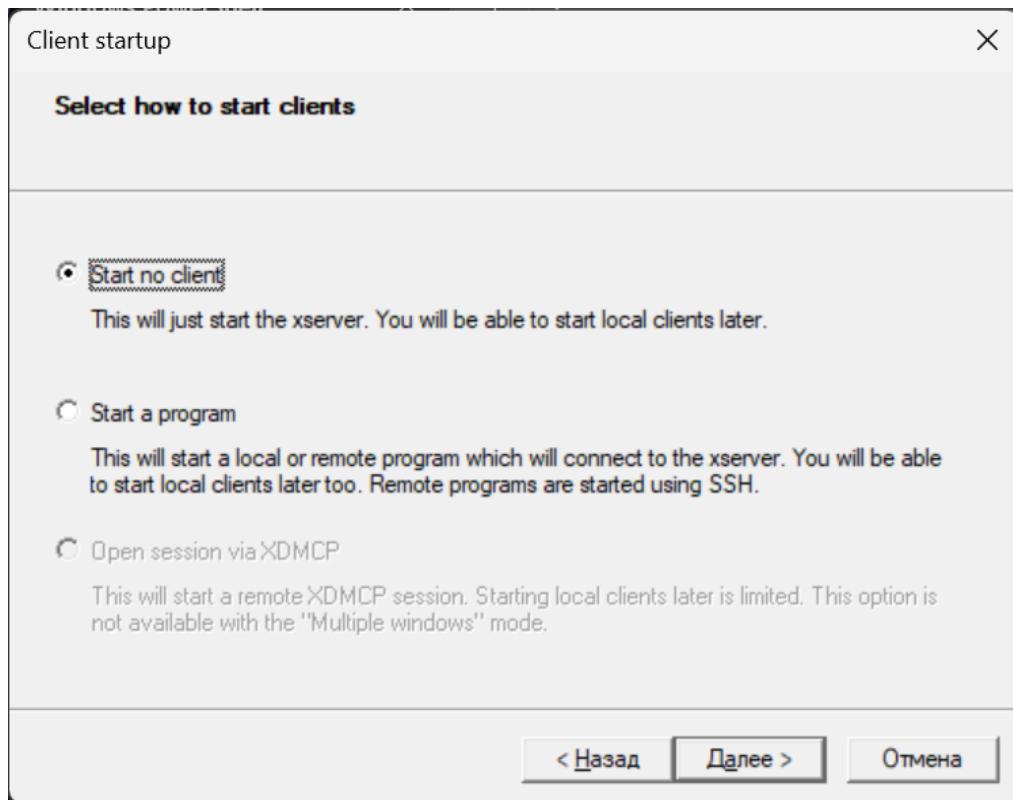


Рис. 3.7: Запуск и настройка Xserver | Часть 2

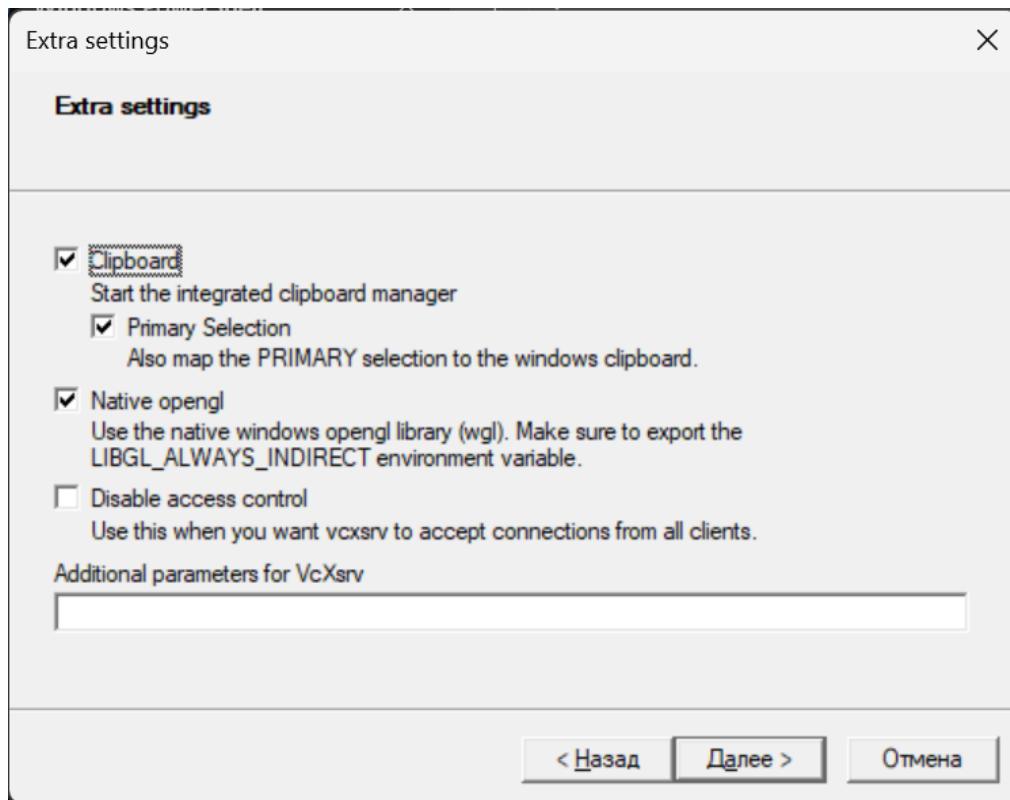


Рис. 3.8: Запуск и настройка Xserver | Часть 3

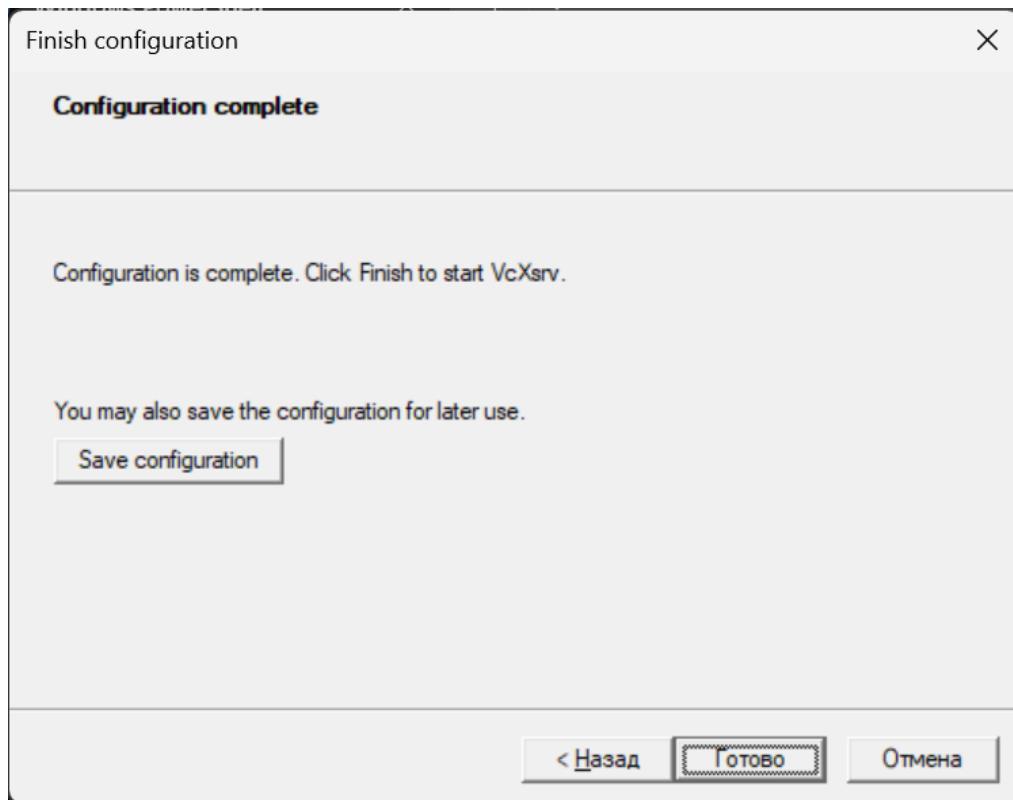


Рис. 3.9: Запуск и настройка Xserver | Часть 4

Запустим putty. При подключении добавим опцию перенаправления X11: – Connection SSH X11 : Enable X11 forwarding (рис. 3.10):

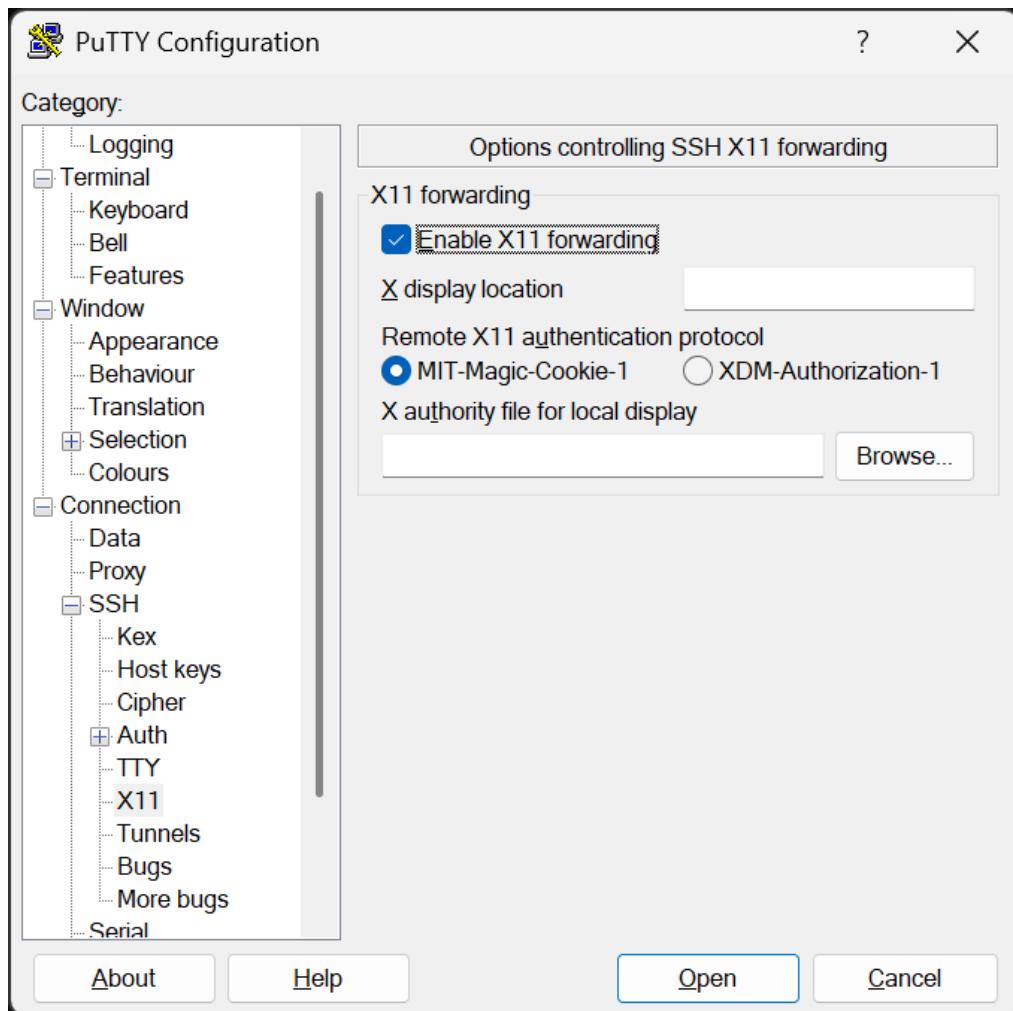


Рис. 3.10: Опция перенаправления X11

Для удобства дальнейшей работы добавим для mininet указание на использование двух адаптеров при запуске. Для этого требуется перейти в режим суперпользователя и внести изменения в файл /etc/netplan/01-netcfg.yaml виртуальной машины mininet. В результате файл /etc/netplan/01-netcfg.yaml должен иметь следующий вид(рис. 3.11):

```
mininet@mininet-vm:~$ cat /etc/netplan/01-netcfg.yaml
# This file describes the network interfaces available on your system
# For more information, see netplan(5).
network:
  version: 2
  renderer: networkd
  ethernets:
    eth0:
      dhcp4: yes
    eth1:
      dhcp4: yesmininet@mininet-vm:~$ _
```

Рис. 3.11: ОФайл /etc/netplan/01-netcfg.yaml

В виртуальной машине mininet переименуем предыдущую установку Mininet. Скачаем новую версию Mininet. Обновим исполняемые файлы(рис. 3.12). Проверим номер установленной версии mininet(рис. 3.13) :

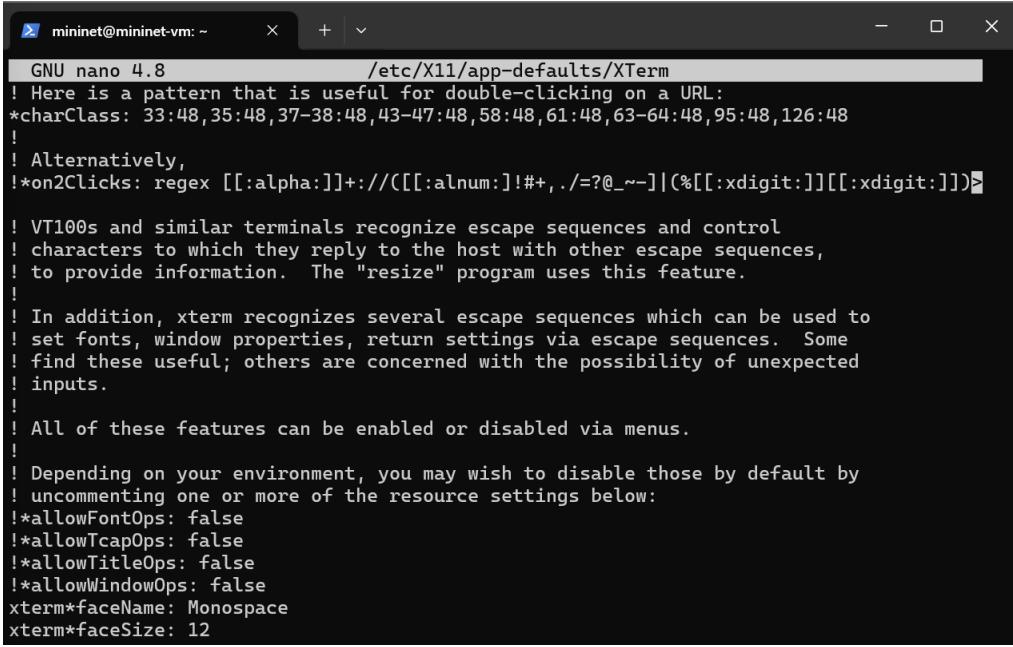
```
mininet@mininet-vm:~$ rm -rf mininet.orig/
mininet@mininet-vm:~$ mv -R ~/mininet ~/mininet.orig
mv: invalid option -- 'R'
Try 'mv --help' for more information.
mininet@mininet-vm:~$ mv ~/mininet ~/mininet.orig
mininet@mininet-vm:~$ cd ~
mininet@mininet-vm:~$ git clone https://github.com/mininet/mininet.git
Cloning into 'mininet'...
remote: Enumerating objects: 10388, done.
remote: Counting objects: 100% (131/131), done.
remote: Compressing objects: 100% (60/60), done.
remote: Total 10388 (delta 104), reused 71 (delta 71), pack-reused 10257 (from 3)
Receiving objects: 100% (10388/10388), 3.36 MiB | 3.18 MiB/s, done.
Resolving deltas: 100% (6906/6906), done.
mininet@mininet-vm:~$ cd ~/mininet
mininet@mininet-vm:~/mininet$ sudo make install
```

Рис. 3.12: Обновление Mininet

```
Successfully installed mininet-2.3.1b4
mininet@mininet-vm:~/mininet$ mn --version
2.3.1b4
mininet@mininet-vm:~/mininet$ _
```

Рис. 3.13: Обновление Mininet/ВЕРСИЯ

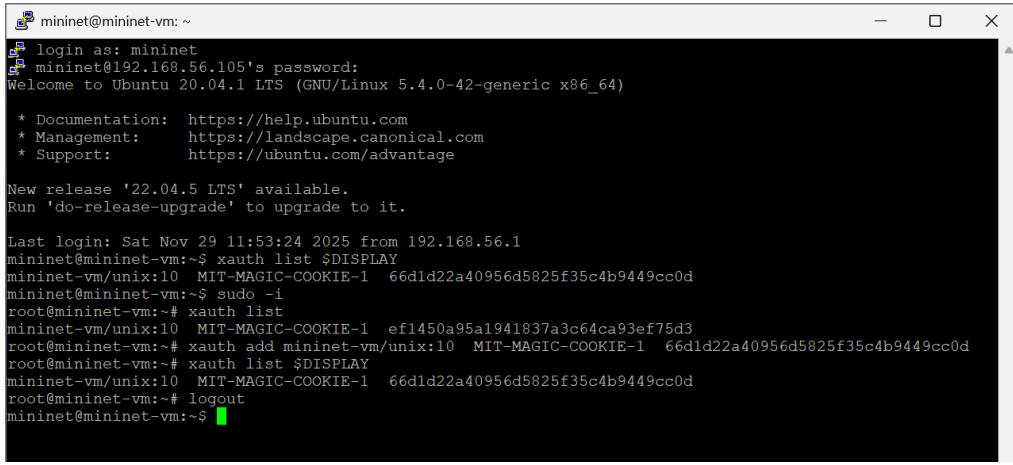
Настроим параметры XTerm для увеличения размера шрифта и применения векторных шрифтов вместо растровых. Внесем изменения в файл /etc/X11/app-defaults/XTerm (рис. 3.14)



```
mininet@mininet-vm: ~      + | -  X
GNU nano 4.8          /etc/X11/app-defaults/XTerm
! Here is a pattern that is useful for double-clicking on a URL:
*charClass: 33:48,35:48,37-38:48,43-47:48,58:48,61:48,63-64:48,95:48,126:48
!
! Alternatively,
!*on2Clicks: regex [[:alpha:]]+://([[:alnum:]!#+,.=/?@_~-]|(%[:xdigit:]][[:xdigit:]]))
!
! VT100s and similar terminals recognize escape sequences and control
! characters to which they reply to the host with other escape sequences,
! to provide information. The "resize" program uses this feature.
!
! In addition, xterm recognizes several escape sequences which can be used to
! set fonts, window properties, return settings via escape sequences. Some
! find these useful; others are concerned with the possibility of unexpected
! inputs.
!
! All of these features can be enabled or disabled via menus.
!
! Depending on your environment, you may wish to disable those by default by
! uncommenting one or more of the resource settings below:
!*allowFontOps: false
!*allowTcapOps: false
!*allowTitleOps: false
!*allowWindowOps: false
xterm*faceName: Monospace
xterm*faceSize: 12
```

Рис. 3.14: Настройка шрифтов XTerm

При попытке запуска приложения из-под суперпользователя возникает ошибка: `X11 connection rejected because of wrong authentication`. 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. 3.15)



```
mininet@mininet-vm: ~
$ login as: mininet
$ mininet@192.168.56.105's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-42-generic x86_64)

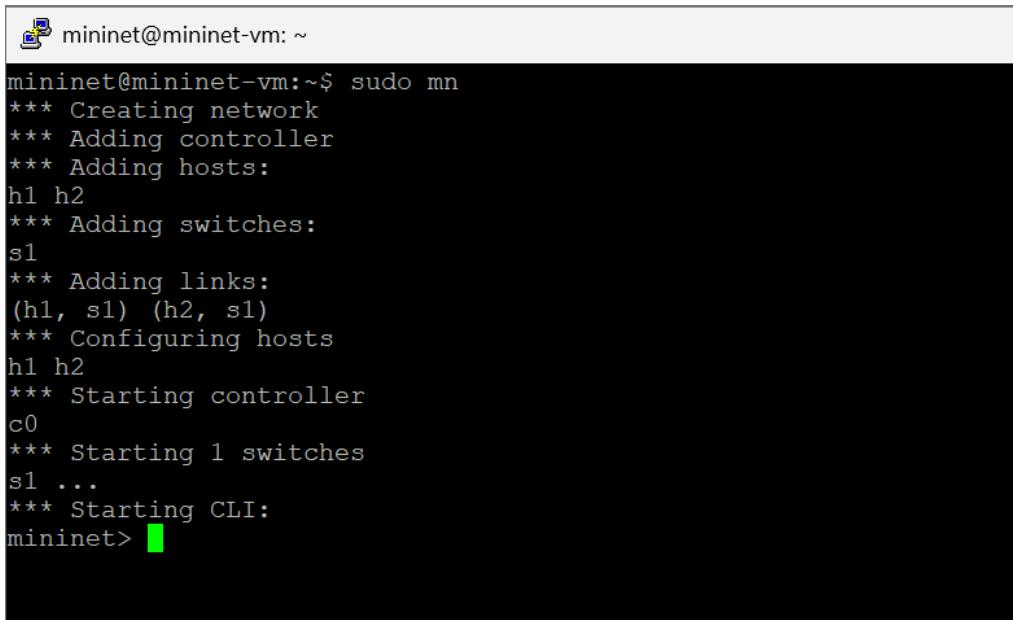
 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

New release '22.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat Nov 29 11:53:24 2025 from 192.168.56.1
mininet@mininet-vm:~$ xauth list $DISPLAY
mininet@mininet-vm:~$ sudo -i
root@mininet-vm:~# xauth list
mininet@mininet-vm:~# xauth add mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 66d1d22a40956d5825f35c4b9449cc0d
root@mininet-vm:~# xauth list $DISPLAY
mininet@mininet-vm:~# xauth add mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 66d1d22a40956d5825f35c4b9449cc0d
root@mininet-vm:~# logout
mininet@mininet-vm:~$
```

Рис. 3.15: Настройка соединения X11 для суперпользователя

Запустим минимальную топологию, состоящую из коммутатора, подключённого к двум хостам (рис. 3.16)



```
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. 3.16: Mininet с использованием топологии по умолчанию

Для отображения списка команд интерфейса командной строки Mininet и примеров их использования введём команду: `help` (рис. 3.17)

```
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. 3.17: Отображение результата help команды

Для отображения доступных узлов введём: `nodes`. Вывод этой команды показывает, что есть два хоста (хост `h1` и хост `h2`) и коммутатор (`s1`)(рис. 3.18)

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

Рис. 3.18: 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. 3.19)

```
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. 3.19: 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, например: h1 ifconfig.(рис. 3.20)

```
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 16:92:40:70:3e:7c 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. 3.20: Команда h1 ifconfig

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 со-

ответственно. Чтобы проверить связь между ними, используем команду ping. Команда ping работает, отправляя сообщения эхо-запроса протокола управляющих сообщений Интернета (ICMP) на удалённый компьютер и ожидая ответа. Например, команда h1 ping 10.0.0.2 проверяет соединение между хостами h1 и h2 (рис. 3.21)

```
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=3.65 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.394 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.052 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.092 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.086 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.088 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.089 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=0.087 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=0.135 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=0.081 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=0.069 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=0.080 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=0.105 ms
^C
--- 10.0.0.2 ping statistics ---
13 packets transmitted, 13 received, 0% packet loss, time 12274ms
rtt min/avg/max/mdev = 0.052/0.384/3.645/0.944 ms
mininet> █
```

Рис. 3.21: Команда h1 ping 10.0.0.2

Очистим предыдущий экземпляр Mininet(рис. 3.22)

```

mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 123.883 seconds
mininet@mininet-vm:~$ sudo mn -c
*** Removing excess controllers/ofprotocols/ofdatapaths/pings/noxes
killall controller ofprotoctl ofdatapath ping nox_core lt-nox_core ovs-openflowd ovs-controller ovs-test
controller udpbwtest mnexec ivs ryu-manager 2> /dev/null
killall -9 controller ofprotoctl ofdatapath ping nox_core lt-nox_core ovs-openflowd ovs-controller ovs-test
controller udpbwtest mnexec ivs ryu-manager 2> /dev/null
pkill -9 -f "sudo mnexec"
*** Removing junk from /tmp
rm -f /tmp/vconn* /tmp/vlogs* /tmp/*.out /tmp/*.log
*** Removing old X11 tunnels
*** Removing excess kernel datapaths
ps ax | egrep -o 'dp[0-9]+' | sed 's/dp/nl:/'
*** Removing OVS datapaths
ovs-vsctl --timeout=1 list-br
ovs-vsctl --timeout=1 list-br
*** Removing all links of the pattern foo-ethX
ip link show | egrep -o '([-_.[:alnum:]]+-eth[[:digit:]]+)'
ip link show
*** Killing stale mininet node processes
pkill -9 -f mininet:
*** Shutting down stale tunnels
pkill -9 -f Tunnel=Ethernet
pkill -9 -f .ssh/mn
rm -f ~/.ssh/mn/*
*** Cleanup complete.
mininet@mininet-vm:~$ █

```

Рис. 3.22: Очистка предыдущего экземпляра Mininet

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

Основные кнопки: - Select: позволяет выбирать/перемещать устройства. Нажатие Del на клавиатуре после выбора устройства удаляет его из топологии. - Host: позволяет добавить новый хост в топологию. После нажатия этой кнопки щелкните в любом месте пустого холста, чтобы вставить новый хост. - Switch: позволяет добавить в топологию новый коммутатор. После нажатия этой кнопки щелкните в любом месте пустого холста, чтобы вставить переключатель. - Link: соединяет устройства в топологии. После нажатия этой кнопки щелкните устройство и перетащите его на второе устройство, с которым необходимо установить связь. - Run: запускает эмуляцию. После проектирования и настройки топологии нажмите кнопку запуска. - Stop: останавливает эмуляцию.

Добавим два хоста и один коммутатор, соединим хосты с коммутатором.(рис. 3.23)

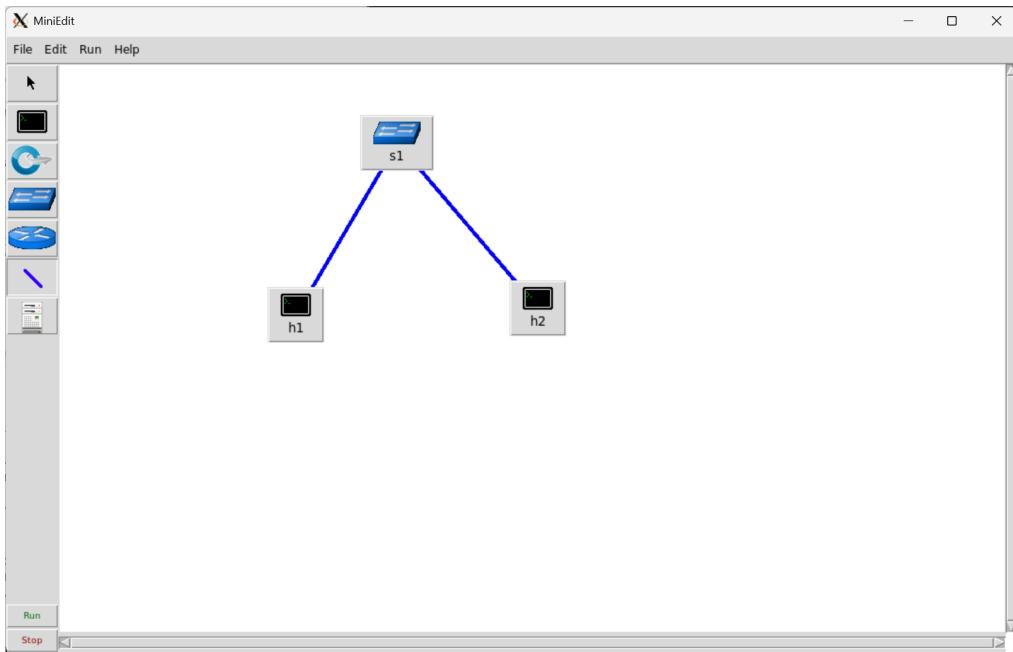


Рис. 3.23: Добавление двух хостов и одного коммутатора

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, а для хоста h2 – 10.0.0.2/8.(рис. 3.24)

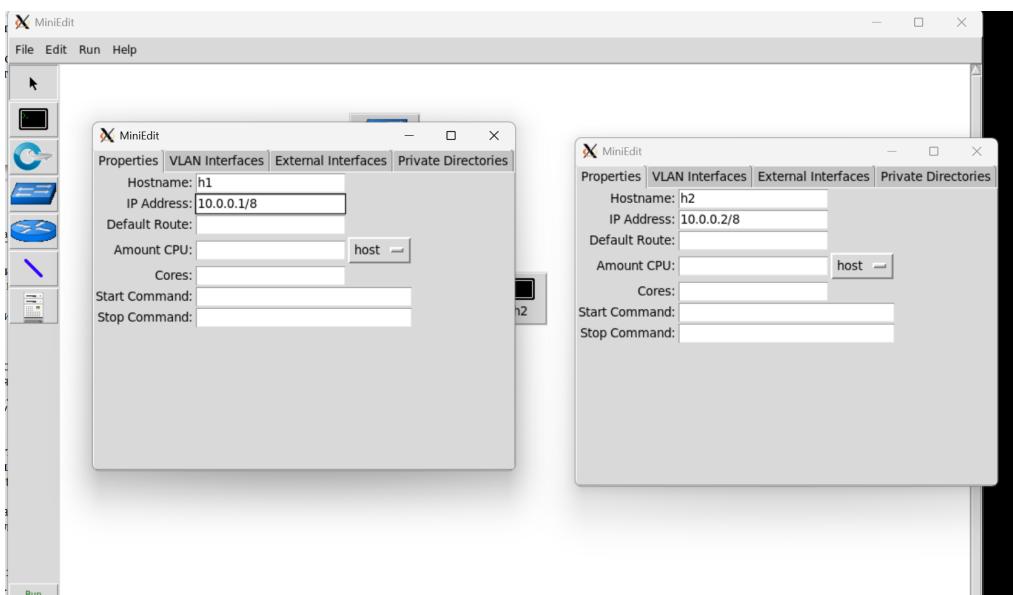


Рис. 3.24: Настройка IP-адреса на хосте h1 и h2

Перед проверкой соединения между хостом h1 и хостом h2 необходимо за-

пустить эмуляцию. Для запуска эмуляции нажмем кнопку Run. После начала эмуляции кнопки панели MiniEdit станут серыми, указывая на то, что в настоящее время они отключены.

Откроем терминал на хосте h1, удерживая правую кнопку мыши на хосте h1 и выбрав Terminal. Это действие позволит выполнять команды на хосте h1. Откроем терминал на хосте h2. На терминале хоста h1 введем 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 подсети 255.0.0.0. Проверим соединение между хостами, введя в терминале хоста h1 команду ping 10.0.0.2. Для остановки теста нажмем Ctrl + c. Остановим эмуляцию, нажав кнопку Stop.(рис. 3.25)

The screenshot shows two terminal windows side-by-side. The left window is titled 'Host:h1@mininet-vm' and the right is 'Host:h2@mininet-vm'. Both show the output of the 'ifconfig' command. The h1-eth0 interface on h1 has an IP of 10.0.0.1 and the h2-eth0 interface on h2 has an IP of 10.0.0.2. The right window then shows the output of a 'ping' command from h1 to h2, which is successful.

```
root@mininet-vm:/home/mininet# ifconfig
h1-eth0: flags=4163 mtu 1500
    inet 10.0.0.1 netmask 255.0.0.0 broadcast 10.255.255.255
        ether f6:99:d2:05:30:c0 txqueuelen 1000 (Ethernet)
        RX packets 283 bytes 27118 (27.1 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 283 bytes 27118 (27.1 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73 mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        loop txqueuelen 1000 (Local Loopback)
        RX packets 5732 bytes 1206480 (1.2 MB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 5732 bytes 1206480 (1.2 MB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# 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=0.046 ms

root@mininet-vm:/home/mininet# ifconfig
h2-eth0: flags=4163 mtu 1500
    inet 10.0.0.2 netmask 255.0.0.0 broadcast 10.255.255.255
        ether 2a:e0:1a:c8:6e:49 txqueuelen 1000 (Ethernet)
        RX packets 283 bytes 27118 (27.1 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 283 bytes 27118 (27.1 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73 mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
        loop txqueuelen 1000 (Local Loopback)
        RX packets 2897 bytes 621460 (621.4 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 2897 bytes 621460 (621.4 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=0.363 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.062 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.089 ms
```

Рис. 3.25: Проверка IP-адресов. Пинг

Ранее IP-адреса узлам h1 и h2 были назначены вручную. В качестве альтернативы можно полагаться на Mininet для автоматического назначения IP-адресов.

Удалим назначенный вручную IP-адрес с хостов h1 и h2. В MiniEdit нажмем 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 (рис. 3.26)

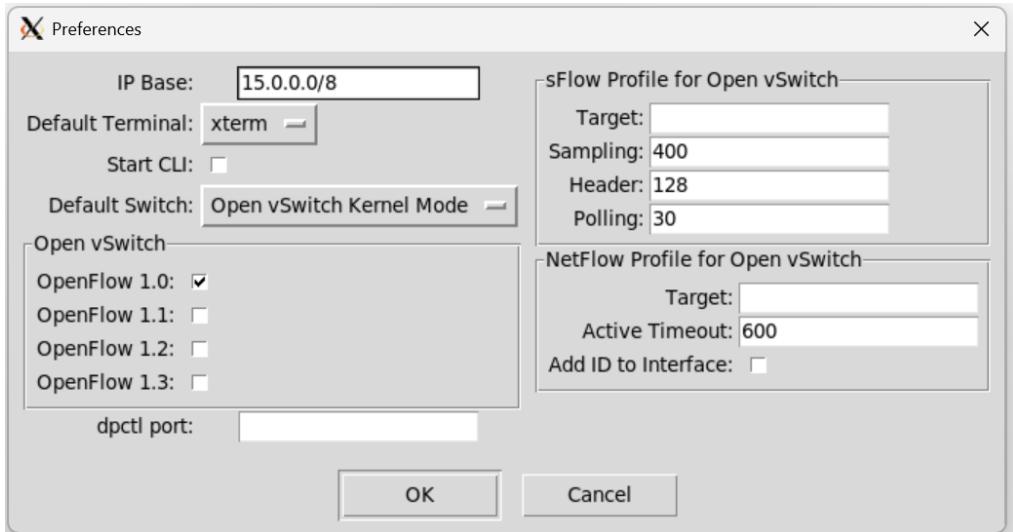


Рис. 3.26: Смена базового IP-адреса

Запустим эмуляцию, нажав кнопку Run. Откроем терминал на хосте h1, удерживая правую кнопку мыши на хосте h1 и выбрав Terminal. Чтобы отобразить IP-адреса, назначенные хосту h1, введем команду ifconfig. Интерфейс h1-eth0 на узле h1 теперь имеет IP-адрес 15.0.0.1 и маску подсети 255.0.0.0. Проверим IP-адрес, назначенный хосту h2. Соответствующий интерфейс h2-eth0 на хосте h2 имеет IP-адрес 15.0.0.2 и маску подсети 255.0.0.0(рис. 3.27)

```
h1 "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 f2:c6:09:b8:18:24 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

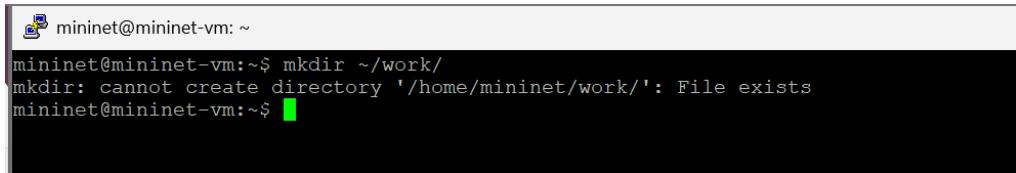
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
              loop txqueuelen 1000 (Local Loopback)
              RX packets 864 bytes 224688 (224.6 KB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 864 bytes 224688 (224.6 KB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet#
```

Рис. 3.27: Просмотр IP-адреса на h1

Остановим эмуляцию, нажав кнопку Stop.

В домашнем каталоге виртуальной машины mininet создадим каталог для работы с проектами mininet: `mkdir ~ /work.`(рис. 3.28)



```
mininet@mininet-vm:~$ mkdir ~ /work/
mkdir: cannot create directory '/home/mininet/work/': File exists
mininet@mininet-vm:~$
```

Рис. 3.28: Создание нового каталога

Для сохранения топологии сети в файл нажмем в MiniEdit File Save. Укажем имя для топологии и сохраним на своём компьютере.(рис. 3.29)

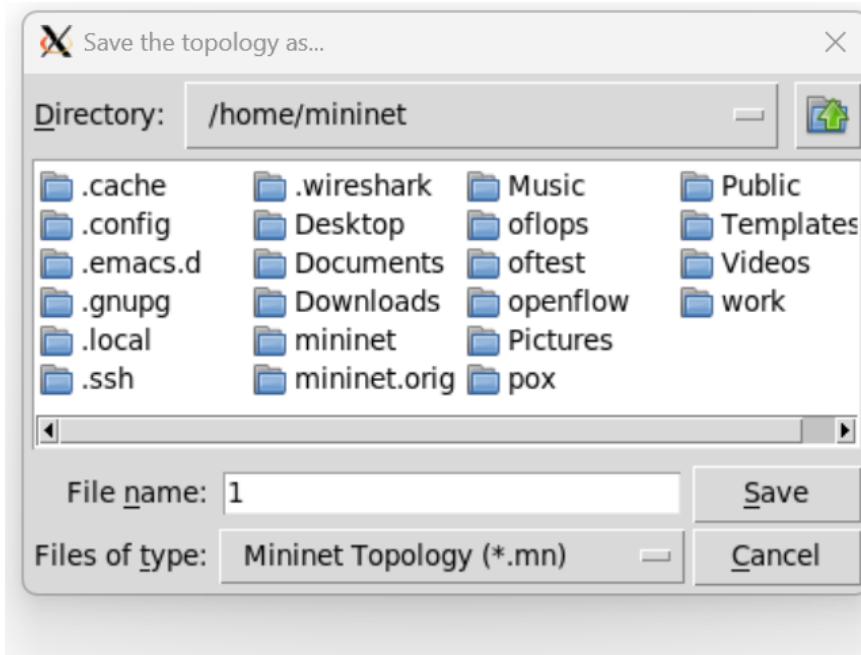
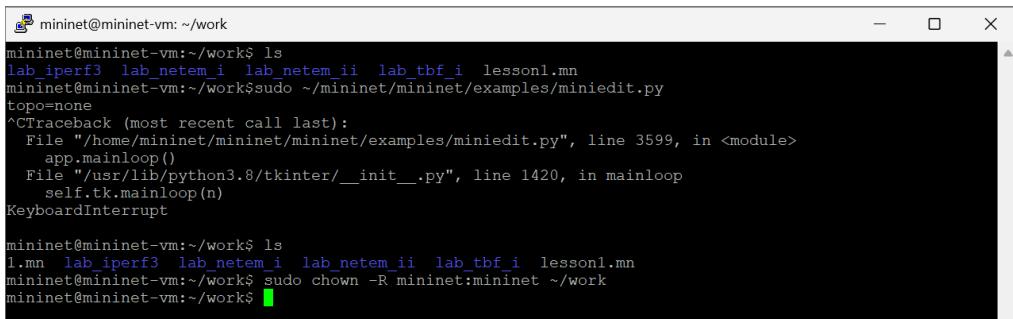


Рис. 3.29: Сохранение топологии

После сохранения проекта поменяем права доступа к файлам в каталоге проекта: `sudo chown -R mininet:mininet ~ /work.` Для загрузки топологии в MiniEdit нажмем File Open(рис. 3.30)



```
mininet@mininet-vm:~/work$ ls
lab_iperf3 lab_netem_i lab_netem_ii lab_tbf_i lesson1.mn
mininet@mininet-vm:~/work$ sudo ~/mininet/mininet/examples/miniedit.py
topo=none
^CTraceback (most recent call last):
  File "/home/mininet/mininet/mininet/examples/miniedit.py", line 3599, in <module>
    app.mainloop()
  File "/usr/lib/python3.8/tkinter/__init__.py", line 1420, in mainloop
    self.tk.mainloop(n)
KeyboardInterrupt

mininet@mininet-vm:~/work$ ls
1.mn lab_iperf3 lab_netem_i lab_netem_ii lab_tbf_i lesson1.mn
mininet@mininet-vm:~/work$ sudo chown -R mininet:mininet ~/work
mininet@mininet-vm:~/work$
```

Рис. 3.30: Изменение прав доступа к файлам в каталоге проекта

4 Выводы

В результате выполнения работы я развернул mininet в системе виртуализации VirtualBox и ознакомилась с основными командами для работы с Mininet через командную строку и через графический интерфейс.

Список литературы

1. Mininet [Электронный ресурс]. Mininet Project Contributors. URL: <http://mininet.org/> (дата обращения: 07.10.2025).