

Лабораторная работа №1

Введение в Mininet

Шияпова Дарина Илдаровна

Содержание

1	Цель работы	4
2	Теоретическое введение	5
3	Выполнение лабораторной работы	6
4	Выводы	17
	Список литературы	18

Список иллюстраций

3.1	Настройка сети	7
3.2	Настройка сети	7
3.3	Запуск mininet	8
3.4	Подключение к mininet через SSH	9
3.5	Просмотр IP-адресов машины	9
3.6	Файл /etc/netplan/01-netcfg.yaml	10
3.7	Обновление Mininet	10
3.8	Номер установленной версии mininet	11
3.9	Настройка соединения X11 для суперпользователя	11
3.10	Работа с Mininet с помощью командной строки	12
3.11	Работа с Mininet с помощью командной строки	13
3.12	Работа с Mininet с помощью командной строки	13
3.13	Проверка связности хостов	14
3.14	sudo ~/mininet/mininet/examples/miniedit.py	14
3.15	Назначение IP-адресов	15

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 Теоретическое введение

Mininet[**mininet?**] – 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.

Mininet создает реалистичную виртуальную сеть, выполняя реальный код ядра, коммутатора и приложения на одной машине (VM, облачной или собственной) за считанные секунды с помощью одной команды `sudo mn`.

3 Выполнение лабораторной работы

Перейдем в репозиторий Mininet, скачаем актуальный релиз ovf-образа виртуальной машины. Запустим систему виртуализации и импортируем файл .ovf и укажем параметры импорта.

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”. 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). В настройках сети первый адаптер должен иметь подключение типа NAT (рис. 3.1). Для второго адаптера укажите тип подключения host-only network adapter (виртуальный адаптер хоста), который в дальнейшем вы будете использовать для входа в образ виртуальной машины (рис. 3.2).

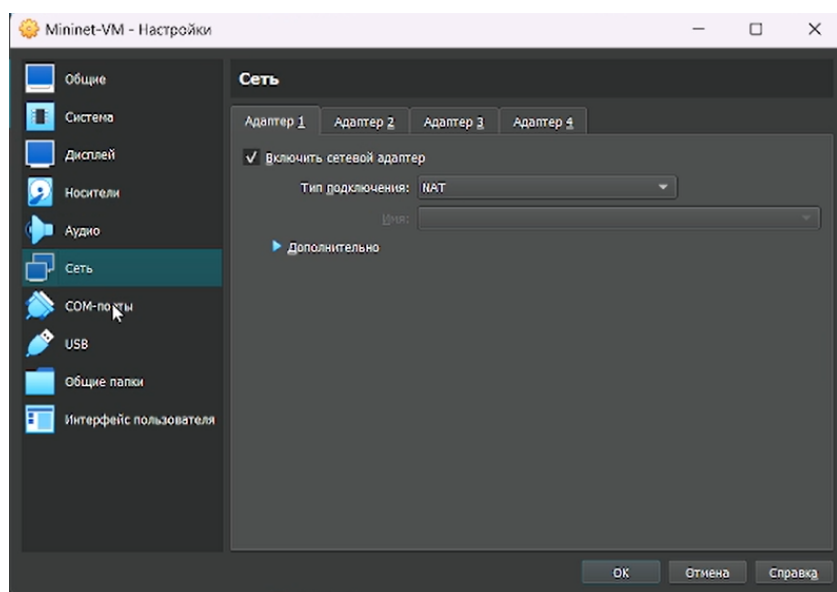


Рис. 3.1: Настройка сети

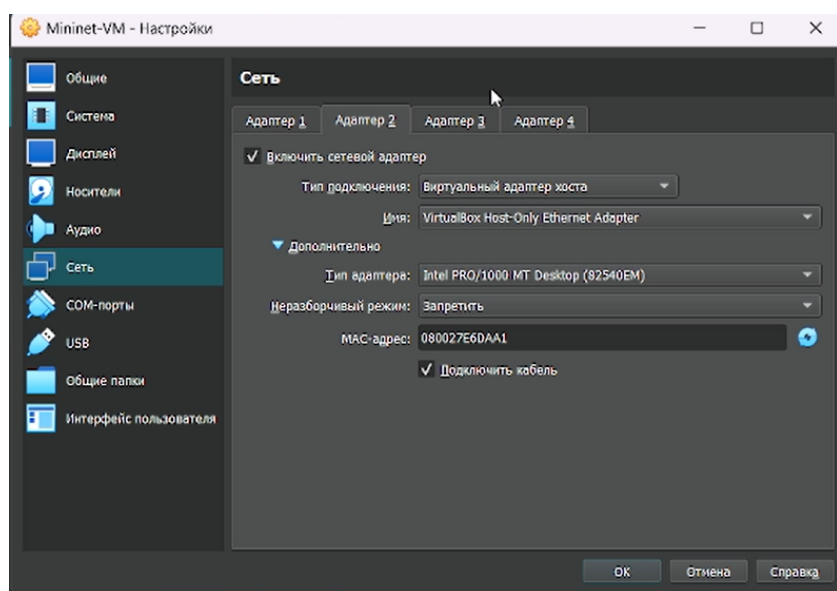
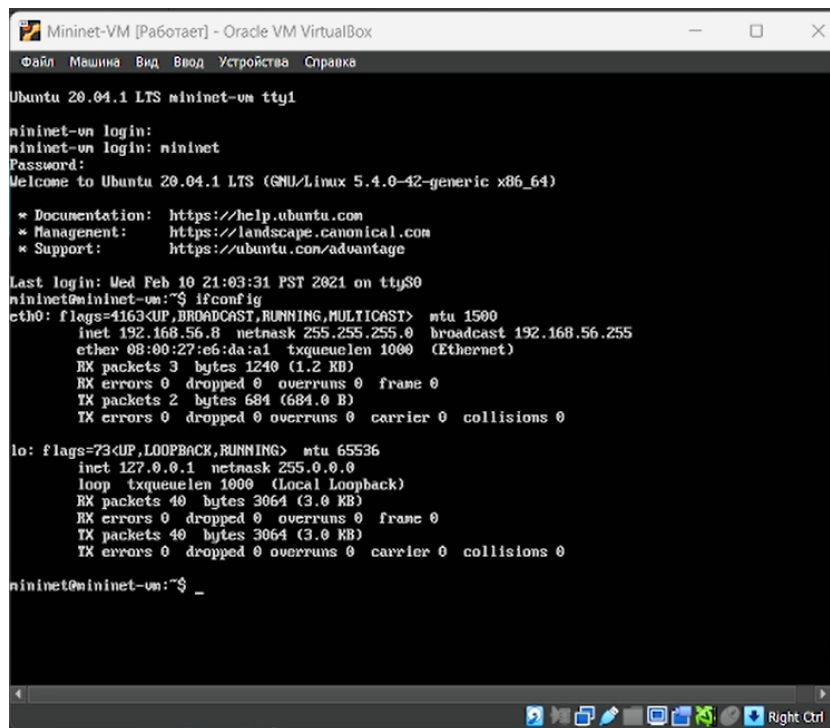


Рис. 3.2: Настройка сети

Запустим виртуальную машину с Mininet. Залогинимся в виртуальную машину:
 - login: mininet - password: mininet

Посмотрите адрес машины с помощью ifconfig (рис. 3.3).



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

Ubuntu 20.04.1 LTS mininet-vm tty1

mininet-vm login:
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 192.168.56.8  netmask 255.255.255.0  broadcast 192.168.56.255
    ether 08:00:27:e6:da:a1  txqueuelen 1000  (Ethernet)
    RX packets 3  bytes 1240 (1.2 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 40  bytes 3064 (3.0 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 40  bytes 3064 (3.0 KB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

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

Рис. 3.3: Запуск mininet

Подключимся к виртуальной машине (из терминала хостовой машины). Настроим ssh-подсоединение по ключу к виртуальной машине. Вновь подключимся к виртуальной машине и убедимся, что подсоединение происходит успешно и без ввода пароля (рис. 3.4).


```

/usr/bin/xauth: file /home/mininet/.Xauthority does not exist
mininet@mininet-vm:~$
mininet@mininet-vm:~$ logout
Connection to 192.168.56.8 closed.
darina@LAPTOP-ONSDH9GT:~$ ssh -Y mininet@192.168.56.8
mininet@192.168.56.8'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)

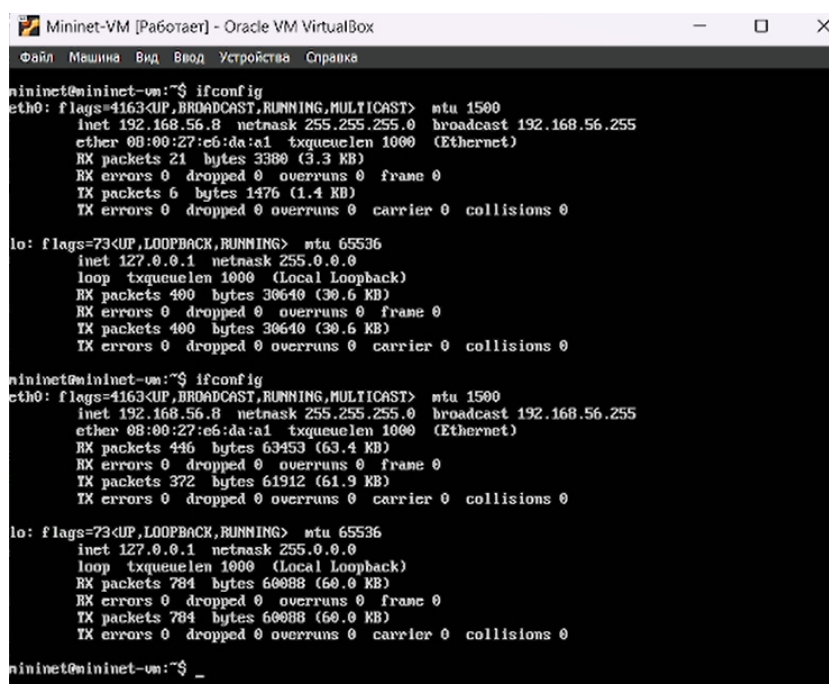
 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Last login: Wed Sep 10 11:01:32 2025 from 192.168.56.1
mininet@mininet-vm:~$ ssh-copy-id mininet@192.168.56.8
/usr/bin/ssh-copy-id: ERROR: No identities found
mininet@mininet-vm:~$ logout
Connection to 192.168.56.8 closed.
darina@LAPTOP-ONSDH9GT:~$ ssh-copy-id mininet@192.168.56.8
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/darina/.ssh/id_ed25519.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
mininet@192.168.56.8's password:

```

Рис. 3.4: Подключение к mininet через SSH

После подключения к виртуальной машине mininet посмотрим IP-адреса машины. Активен только внутренний адрес машины вида 192.168.x.y, поэтому активируем второй интерфейс (рис. 3.5).



```

Mininet-VM [Работае] - Oracle VM VirtualBox
Файл  Машинка  Вид  Ввод  Устройства  Справка
mininet@mininet-vm:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.8 netmask 255.255.255.0 broadcast 192.168.56.255
    ether 08:00:27:c6:da:a1 txqueuelen 1000 (Ethernet)
    RX packets 21 bytes 3380 (3.3 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 6 bytes 1476 (1.4 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 400 bytes 30640 (30.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 400 bytes 30640 (30.6 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

mininet@mininet-vm:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.56.8 netmask 255.255.255.0 broadcast 192.168.56.255
    ether 08:00:27:c6:da:a1 txqueuelen 1000 (Ethernet)
    RX packets 446 bytes 63453 (63.4 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 372 bytes 61912 (61.9 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 784 bytes 60088 (60.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 784 bytes 60088 (60.0 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

mininet@mininet-vm:~$ _

```

Рис. 3.5: Просмотр IP-адресов машины

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. 3.6).

```

/etc/netplan/01-netcfg.yaml 1-M--1 18 L:1 1+ 9 10/ 101 *(214 / 214b) <EOF> 1-M1X1
# 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: yes

```

Рис. 3.6: Файл /etc/netplan/01-netcfg.yaml

В виртуальной машине mininet переименуем предыдущую установку Mininet. Скачаем новую версию Mininet. Обновим исполняемые файлы (рис. 3.7).

```

mininet@mininet-vm:~$
mininet@mininet-vm:~$
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: 10308, done.
remote: Counting objects: 100% (128/128), done.
remote: Compressing objects: 100% (60/60), done.
remote: Total 10308 (delta 102), reused 60 (delta 60), pack-reused 10260 (from 3)
Receiving objects: 100% (10308/10308), 3.36 MiB | 2.52 MiB/s, done.
Resolving deltas: 100% (6905/6905), done.
mininet@mininet-vm:~$ cd ~/mininet
mininet@mininet-vm:~/mininet$ sudo make install
cc -Wall -Wextra \
-DVERSION="\"PYTHONPATH=. python -B bin/mn --version 2>&1\" mnexec.c -o mnexec
install -D mnexec /usr/bin/mnexec
PYTHONPATH=. help2man -N -n "create a Mininet network." \
--no-discard-stderr "python -B bin/mn" -o mn.1
help2man -N -n "execution utility for Mininet." \
-h "-h" -v "-v" --no-discard-stderr ./mnexec -o mnexec.1
install -D -t /usr/share/man/man1 mn.1 mnexec.1
python -m pip uninstall -y mininet || true
Found existing installation: mininet 2.3.0
Uninstalling mininet-2.3.0:
  Successfully uninstalled mininet-2.3.0
python -m pip install .
Processing /home/mininet/mininet
Requirement already satisfied: setuptools in /usr/lib/python3/dist-packages (from mininet==2.3.1b4)
(45.2.0)
Building wheels for collected packages: mininet

```

Рис. 3.7: Обновление Mininet

Проверим номер установленной версии mininet (рис. 3.8).

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

Рис. 3.8: Номер установленной версии mininet

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. 3.9).

```
mininet@mininet-vm:~/mininet$ xauth list $DISPLAY
mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 667b953b15572667cd602272875a2bb5
mininet-vm/unix:11 MIT-MAGIC-COOKIE-1 b94ad20bf2c8f7655250d0e6df0c6ba5
mininet@mininet-vm:~/mininet$ MIT-MAGIC-COOKIE-1 667b953b15572667cd602272875a2bb5
MIT-MAGIC-COOKIE-1: command not found
mininet@mininet-vm:~/mininet$ /unix:10 MIT-MAGIC-COOKIE-1 667b953b15572667cd602272875a2bb5
-bash: /unix:10: No such file or directory
mininet@mininet-vm:~/mininet$ /unix:10 MIT-MAGIC-COOKIE-1 667b953b15572667cd602272875a2bb5
-bash: /unix:10: No such file or directory
mininet@mininet-vm:~/mininet$ unix:10 MIT-MAGIC-COOKIE-1 667b953b15572667cd602272875a2bb5
unix:10: command not found
mininet@mininet-vm:~/mininet$ MIT-MAGIC-COOKIE-1 667b953b15572667cd602272875a2bb5
MIT-MAGIC-COOKIE-1: command not found
mininet@mininet-vm:~/mininet$ xauth list $DISPLAY
mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 667b953b15572667cd602272875a2bb5
mininet-vm/unix:11 MIT-MAGIC-COOKIE-1 b94ad20bf2c8f7655250d0e6df0c6ba5
mininet@mininet-vm:~/mininet$ sudo -i
root@mininet-vm:~# xauth list
```

Рис. 3.9: Настройка соединения X11 для суперпользователя

Для запуска минимальной топологии введем в командной строке (рис. 3.10): `sudo mn`. Эта команда запускает Mininet с минимальной топологией, состоящей из коммутатора, подключённого к двум хостам. Для отображения списка команд интерфейса командной строки Mininet и примеров их использования введем команду в интерфейсе командной строки Mininet: `help` Для отображения доступных узлов введем: `nodes` Вывод этой команды показывает, что есть два хоста (хост h1 и хост h2) и коммутатор (s1). Иногда бывает полезно отобразить связи между устройствами в Mininet, чтобы понять топологию. Введем команду `net` в интерфейсе командной строки Mininet, чтобы просмотреть доступные линки: `net` 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.

```

*** 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> help

Documented commands (type help <topic>):
=====
EOF      gterm  iperfudp  nodes      pingpair    py      switch  xterm
dpctl    help   link      noecho     pingpairfull  quit    time
dump     intfz  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 b
mininet> xterm h2

```

Рис. 3.10: Работа с Mininet с помощью командной строки

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

```

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 36:3c:cb:6f:6c:97 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

```

Рис. 3.11: Работа с Mininet с помощью командной строки

Посмотрим конфигурацию всех узлов.

```

mininet> h2 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 46:a0:db:37:7c:c6 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> 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=13.4 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.200 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.049 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.052 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.050 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.0
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.0
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=0.0
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=0.0
^C

```

Рис. 3.12: Работа с Mininet с помощью командной строки

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, 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.


```

--- 10.0.0.2 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8160ms
rtt min/avg/max/mdev = 0.043/1.554/13.427/4.197 ms
mininet> h2 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=4.09 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.049 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.067 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.054 ms
64 bytes from 10.0.0.1: icmp_seq=5 ttl=64 time=0.051 ms
64 bytes from 10.0.0.1: icmp_seq=6 ttl=64 time=0.051 ms
64 bytes from 10.0.0.1: icmp_seq=7 ttl=64 time=0.170 ms
64 bytes from 10.0.0.1: icmp_seq=8 ttl=64 time=0.068 ms
64 bytes from 10.0.0.1: icmp_seq=9 ttl=64 time=0.060 ms
^C

```

Рис. 3.13: Проверка связности хостов

Построение и эмуляция сети в Mininet с использованием графического интерфейса

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

```

mininet@mininet-vm:~$ sudo ~/mininet/mininet/examples/miniedit.py
Traceback (most recent call last):
  File "/home/mininet/mininet/mininet/examples/miniedit.py", line 3595, in <module>
    app = MiniEdit()
  File "/home/mininet/mininet/mininet/examples/miniedit.py", line 1123, in __init__
    Frame.__init__(self, parent)
  File "/usr/lib/python3.8/tkinter/__init__.py", line 3119, in __init__
    Widget.__init__(self, master, 'frame', cnf, O, extra)
  File "/usr/lib/python3.8/tkinter/__init__.py", line 2561, in __init__
    BaseWidget._setup(self, master, cnf)
  File "/usr/lib/python3.8/tkinter/__init__.py", line 2527, in _setup
    _default_root = Tk()
  File "/usr/lib/python3.8/tkinter/__init__.py", line 2261, in __init__
    self.tk = _tkinter.create(screenName, baseName, className, interactive, wantobjects, useTk, sync, use)
_tkinter.TclError: no display name and no $DISPLAY environment variable
mininet@mininet-vm:~$

```

Рис. 3.14: `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. – Настроим IP-адреса на хостах h1 и h2 . Для этого удерживая правую кнопку мыши на устройстве выберем свойства. Для хоста h1 укажите IP-адрес 10.0.0.1/8 , а для хоста h2 – 10.0.0.2/8 .

Проверка связности. – Перед проверкой соединения между хостом h1 и хостом h2 необходимо запустить 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 должен быть 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 должен быть 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` . – Остановим эмуляцию, нажав кнопку Stop .

Автоматическое назначение IP-адресов.

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

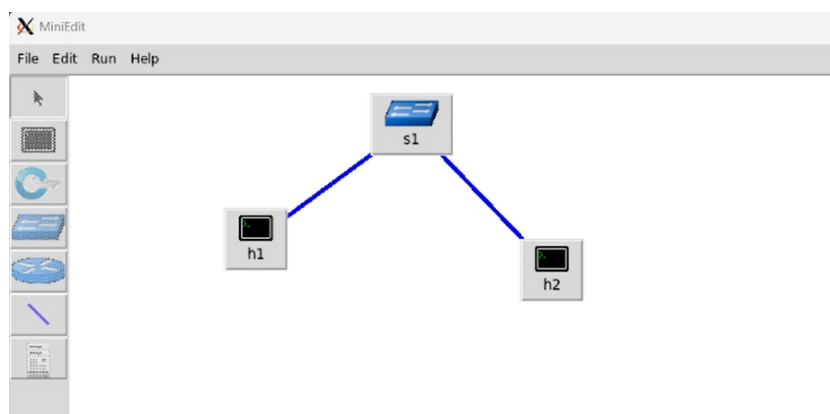


Рис. 3.15: Назначение IP-адресов

– 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 . – 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 .12 – Проверим IP-адрес, назначенный хосту h2 . Соответствующий интерфейс h2-eth0 на хосте h2 должен иметь IP-адрес 15.0.0.2 и маску подсети 255.0.0.0 . – Остановим эмуляцию, нажав кнопку Stop .

4 Выводы

В результате выполнения данной лабораторной работы я развёрнула mininet в системе виртуализации VirtualBox, а также ознакомилась с основными командами для работы с Mininet через командную строку и через графический интерфейс.

Список литературы