

# **Отчёт по лабораторной работе №3**

**Измерение и тестирование пропускной способности сети.  
Воспроизводимый эксперимент.**

Козлов Всеволод Павлович НФИбд-02-22

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
<b>4</b>	<b>Выводы</b>	<b>16</b>
<b>5</b>	<b>Список литературы</b>	<b>17</b>

# Список иллюстраций

3.1	Подкаталог с примером скрипта . . . . .	7
3.2	Содержание скрипта . . . . .	8
3.3	Запуск скрипта . . . . .	8
3.4	Элементы топологии . . . . .	9
3.5	Вывд на экран информации о хосте h1 . . . . .	9
3.6	Запуск нового скрипта . . . . .	10
3.7	Информация об обоих хостах . . . . .	10
3.8	Запуск нового скрипта . . . . .	11
3.9	Копию скрипта lab_iperf3_topo.py . . . . .	11
3.10	Импорт классов, описание сети, размер CPU и тд . . . . .	12
3.11	Запуск нового скрипта . . . . .	12
3.12	Копия скрипта lab_iperf3_topo2.py . . . . .	13
3.13	Изменения в коде . . . . .	13
3.14	Запуск скрипта . . . . .	14
3.15	Графики из получившегося JSON-файла . . . . .	14
3.16	Makefile . . . . .	15
3.17	Проверка Makefile . . . . .	15

## Список таблиц

# 1 Цель работы

Основной целью работы является знакомство с инструментом для измерения пропускной способности сети в режиме реального времени — iPerf3, а также получение навыков проведения воспроизводимого эксперимента по измерению пропускной способности моделируемой сети в среде Mininet.

## 2 Задание

1. Воспроизвести посредством API Mininet эксперименты по измерению пропускной способности с помощью iPerf3.
2. Построить графики по проведённому эксперименту

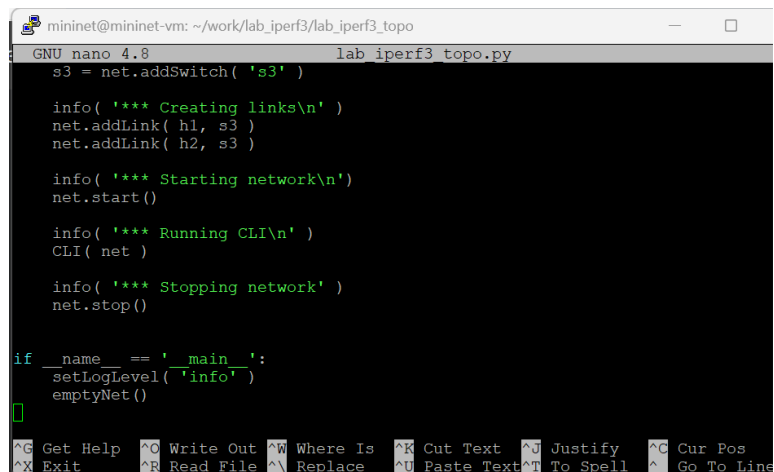
### 3 Выполнение лабораторной работы

Создал подкаталог с примером скрипта (рис. 3.1)

```
mininet@mininet-vm:~$ cd ~/work/lab_iperf3
mininet@mininet-vm:~/work/lab_iperf3$ mkdir lab_iperf3_topo
mininet@mininet-vm:~/work/lab_iperf3$ cd ~/work/lab_iperf3/lab_iperf3_topo
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ cp ~/mininet/examples/emptynet.py ~/work/lab_iperf3/lab_iperf3_topo
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ mv emptynet.py lab_iperf3_topo.py
```

Рис. 3.1: Подкаталог с примером скрипта

Изучил содержание скрипта (рис. 3.2)



```
mininet@mininet-vm: ~/work/lab_iperf3/lab_iperf3_topo
GNU nano 4.8 lab_iperf3_topo.py

s3 = net.addSwitch( 's3' )

info( '*** Creating links\n' )
net.addLink( h1, s3 )
net.addLink( h2, s3 )

info( '*** Starting network\n' )
net.start()

info( '*** Running CLI\n' )
CLI( net )

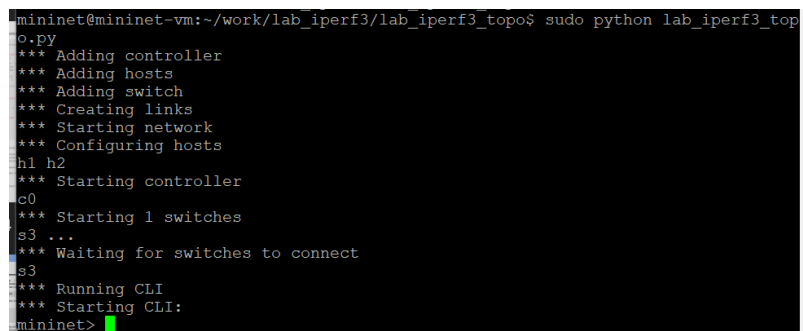
info( '*** Stopping network' )
net.stop()

if __name__ == '__main__':
    setLogLevel( 'info' )
    emptyNet()

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line
```

Рис. 3.2: Содержание скрипта

Запустил скрипт (рис. 3.3)



```
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ sudo python lab_iperf3_topo.py
*** Adding controller
*** Adding hosts
*** Adding switch
*** Creating links
*** Starting network
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s3 ...
*** Waiting for switches to connect
s3
*** Running CLI
*** Starting CLI:
mininet>
```

Рис. 3.3: Запуск скрипта

Посмотрел элементы топологии и завершил работу mininet (рис. 3.4)



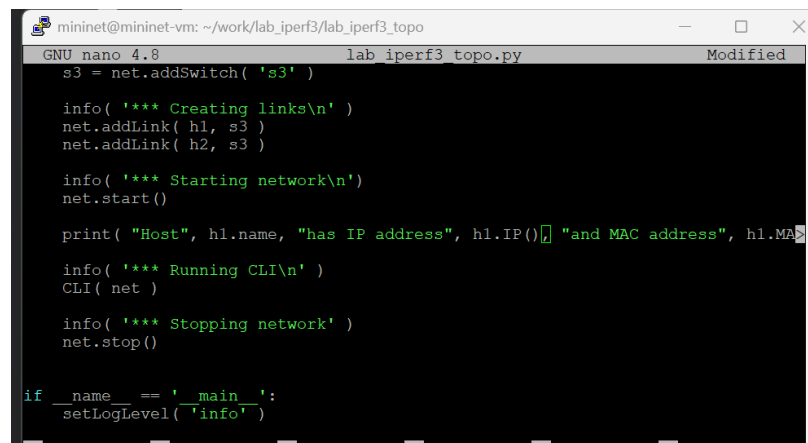
```

mininet> net
h1 h1-eth0:s3-eth1
h2 h2-eth0:s3-eth2
s3 lo: s3-eth1:h1-eth0 s3-eth2:h2-eth0
c0
mininet> links
h1-eth0<->s3-eth1 (OK OK)
h2-eth0<->s3-eth2 (OK OK)
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=1692>
<Host h2: h2-eth0:10.0.0.2 pid=1696>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None pid=1701>
<Controller c0: 127.0.0.1:6653 pid=1685>
mininet> exit
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s3
*** Stopping 2 hosts
h1 h2

```

Рис. 3.4: Элементы топологии

Изменение, позволяющее вывести на экран информацию о хосте h1 (рис. 3.5)



```

mininet@mininet-vm: ~/work/lab_iperf3/lab_iperf3_topo
GNU nano 4.8 lab_iperf3_topo.py Modified
s3 = net.addSwitch( 's3' )

info( '*** Creating links\n' )
net.addLink( h1, s3 )
net.addLink( h2, s3 )

info( '*** Starting network\n' )
net.start()

print( "Host", h1.name, "has IP address", h1.IP() "and MAC address", h1.MAC() )

info( '*** Running CLI\n' )
CLI( net )

info( '*** Stopping network' )
net.stop()

if __name__ == '__main__':
    setLogLevel( 'info' )

```

Рис. 3.5: Вывд на экран информации о хосте h1

Запуск нового скрипта (рис. 3.6)

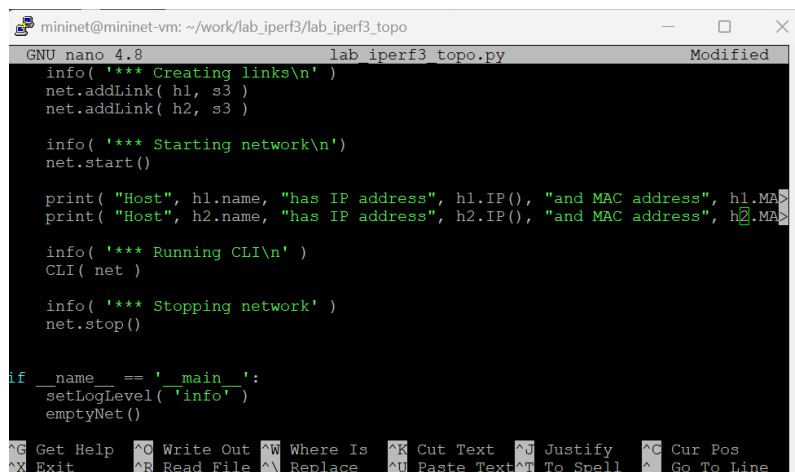
```

mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ sudo python lab_iperf3_topo.py
*** Adding controller
*** Adding hosts
*** Adding switch
*** Creating links
*** Starting network
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s3 ...
*** Waiting for switches to connect
s3
Host h1 has IP address 10.0.0.1 and MAC address fe:0b:ea:c0:b6:33
*** Running CLI
*** Starting CLI:
mininet>

```

Рис. 3.6: Запуск нового скрипта

Изменил, чтобы выводилась информация об обоих хостах (рис. 3.7)



```

mininet@mininet-vm: ~/work/lab_iperf3/lab_iperf3_topo
GNU nano 4.8 lab_iperf3_topo.py Modified
info( '*** Creating links\n' )
net.addLink( h1, s3 )
net.addLink( h2, s3 )

info( '*** Starting network\n' )
net.start()

print( "Host", h1.name, "has IP address", h1.IP(), "and MAC address", h1.MA
print( "Host", h2.name, "has IP address", h2.IP(), "and MAC address", h2.MA

info( '*** Running CLI\n' )
CLI( net )

info( '*** Stopping network' )
net.stop()

if __name__ == '__main__':
    setLogLevel( 'info' )
    emptyNet()

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Paste Text ^T To Spell ^_ Go To Line

```

Рис. 3.7: Информация об обоих хостах

Запуск нового скрипта (рис. 3.8)

```

mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ sudo python lab_iperf3_topo.py
*** Adding controller
*** Adding hosts
*** Adding switch
*** Creating links
*** Starting network
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s3 ...
*** Waiting for switches to connect
s3
Host h1 has IP address 10.0.0.1 and MAC address 12:39:1f:b5:da:e3
Host h2 has IP address 10.0.0.2 and MAC address 6a:5f:45:7c:57:3a
*** Running CLI
*** Starting CLI:
mininet>

```

Рис. 3.8: Запуск нового скрипта

Сделал копию скрипта lab\_iperf3\_topo.py (рис. 3.9)

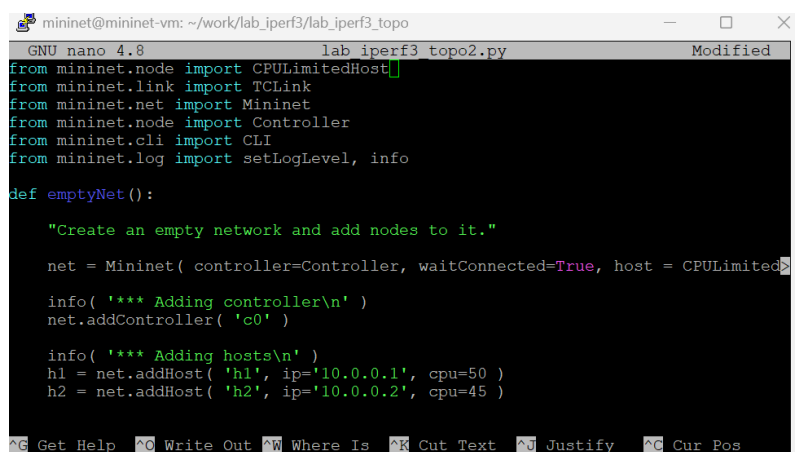
```

mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ cp lab_iperf3_topo.py lab_iperf3_topo2.py
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$

```

Рис. 3.9: Копию скрипта lab\_iperf3\_topo.py

Импорт классов, изменил описание сети, размер CPU и тд (рис. 3.10)

A screenshot of a terminal window showing the nano 4.8 text editor. The file being edited is lab\_iperf3\_topo2.py. The script imports various classes from mininet (CPULimitedHost, TCLink, Mininet, Controller, CLI, setLogLevel) and defines a function emptyNet() that creates a Mininet instance with a controller and two hosts (h1 and h2) with specific IP addresses and CPU limits.

```
mininet@mininet-vm: ~/work/lab_iperf3/lab_iperf3_topo
lab_iperf3_topo2.py Modified
GNU nano 4.8
from mininet.node import CPULimitedHost
from mininet.link import TCLink
from mininet.net import Mininet
from mininet.node import Controller
from mininet.cli import CLI
from mininet.log import setLogLevel, info

def emptyNet():
    "Create an empty network and add nodes to it."

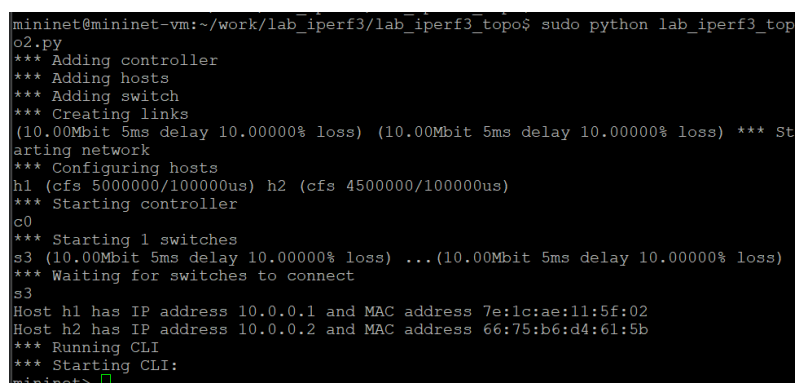
    net = Mininet( controller=Controller, waitConnected=True, host = CPULimitedHost

    info( '*** Adding controller\n' )
    net.addController( 'c0' )

    info( '*** Adding hosts\n' )
    h1 = net.addHost( 'h1', ip='10.0.0.1', cpu=50 )
    h2 = net.addHost( 'h2', ip='10.0.0.2', cpu=45 )
```

Рис. 3.10: Импорт классов, описание сети, размер CPU и тд

Запуск нового скрипта (рис. 3.11)

A screenshot of a terminal window showing the execution of the lab\_iperf3\_topo2.py script. The output shows the creation of the network, adding of controller and hosts, and the configuration of links and hosts. It also shows the starting of the controller and switches, and the configuration of the hosts.

```
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ sudo python lab_iperf3_topo2.py
*** Adding controller
*** Adding hosts
*** Adding switch
*** Creating links
(10.00Mbit 5ms delay 10.000000% loss) (10.00Mbit 5ms delay 10.000000% loss) *** Starting network
*** Configuring hosts
h1 (cfs 5000000/1000000us) h2 (cfs 4500000/1000000us)
*** Starting controller
c0
*** Starting 1 switches
s3 (10.00Mbit 5ms delay 10.000000% loss) ... (10.00Mbit 5ms delay 10.000000% loss)
*** Waiting for switches to connect
s3
Host h1 has IP address 10.0.0.1 and MAC address 7e:1c:ae:11:5f:02
Host h2 has IP address 10.0.0.2 and MAC address 66:75:b6:d4:61:5b
*** Running CLI
*** Starting CLI:
mininet>
```

Рис. 3.11: Запуск нового скрипта

В lab\_iperf3\_topo2.py: - параметры линков: Отображаются характеристики (10.00Mbit 5ms delay 10.000000% loss) - конфигурация хостов: Подробная информация о CFS (Completely Fair Scheduler) Вывод: lab\_iperf3\_topo2.py показывает более детальную информацию о параметрах сети и планировщике, в то время как lab\_iperf3\_topo.py вывод упрощен

Сделал копию скрипта lab\_iperf3\_topo2.py и поместил его в подкаталог iperf (рис. 3.12)

```

mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ cp lab_iperf3_topo2.py lab
_iperf3.py
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ mkdir -p ~/work/lab_iperf3
/iperf3
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ mv ~/work/lab_iperf3/lab_i
per3_topo/lab_iperf3.py ~/work/lab_iperf3/iperf3
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ cd ~/work/lab_iperf3/iperf
3
mininet@mininet-vm:~/work/lab_iperf3/iperf3$ ls -l
total 4
-rwxrwxr-x 1 mininet mininet 1346 Oct 11 01:56 lab_iperf3.py
mininet@mininet-vm:~/work/lab_iperf3/iperf3$

```

Рис. 3.12: Копия скрипта lab\_iperf3\_topo2.py

Далее будут внесены след. изменения: Хосты: Без ограничений CPU Каналы: 100 Мбит/с, задержка 75 мс, без потерь, без ограничителей iPerf3 тест: h2 запускает сервер iPerf3 h1 через 10 сек запускает клиент iPerf3 с сохранением в JSON

Внес изменения в код (рис. 3.13)

```

info( '*** Adding hosts\n' )
h1 = net.addHost( 'h1', ip='10.0.0.1', cpu=1.0 )
h2 = net.addHost( 'h2', ip='10.0.0.2', cpu=1.0 )

info( '*** Adding switch\n' )
s3 = net.addSwitch( 's3' )

info( '*** Creating links\n' )
net.addLink( h1, s3, bw=100, delay='75ms', loss=0, use_htb=False )
net.addLink( h2, s3, bw=100, delay='75ms', loss=0, use_htb=False )

info( '*** Starting network\n' )
net.start()

info( '*** Traffic generation\n' )
h2.cmdPrint( 'iperf3 -s -D -l' )
time.sleep(10)
h1.cmdPrint( 'iperf3 -c', h2.IP(), '-J > iperf_result.json' )

```

Рис. 3.13: Изменения в коде

Запуск скрипта (рис. 3.14)

```

mininet@mininet-vm:~/work/lab_iperf3/iperf3$ sudo python lab_iperf3.py
*** Adding controller
*** Adding hosts
*** Adding switch
*** Creating links
(100.00Mbit 75ms delay 0.00000% loss) (100.00Mbit 75ms delay 0.00000% loss) (100
.00Mbit 75ms delay 0.00000% loss) (100.00Mbit 75ms delay 0.00000% loss) *** Star
ting network
*** Configuring hosts
h1 (cfs 100000/100000us) h2 (cfs 100000/100000us)
*** Starting controller
c0
*** Starting 1 switches
s3 (100.00Mbit 75ms delay 0.00000% loss) (100.00Mbit 75ms delay 0.00000% loss) .
..(100.00Mbit 75ms delay 0.00000% loss) (100.00Mbit 75ms delay 0.00000% loss)
*** Waiting for switches to connect
s3
*** Traffic generation
*** h2 : ('iperf3 -s -D -1',)
*** h1 : ('iperf3 -c', '10.0.0.2', '-J > iperf_result.json')
Host h1 has IP address 10.0.0.1 and MAC address 32:43:31:01:96:8f
Host h2 has IP address 10.0.0.2 and MAC address fa:b2:c2:94:1a:b3
*** Running CLI
*** Starting CLI:

```

Рис. 3.14: Запуск скрипта

Характеристики каналов: - пропускная способность: 100 Мбит/с - задержка: 75 мс - потери: 0% (без потерь) Конфигурация хостов: - h1 и h2 - оба настроены с параметрами CFS (без ограничений CPU) Автоматический тест iperf3: - h2 запущен как сервер: iperf3 -s -D -1 - h1 запущен как клиент к 10.0.0.2 с сохранением в JSON

Построил графики из получившегося JSON-файла (рис. 3.15)

```

mininet@mininet-vm:~/work/lab_iperf3/iperf3$ plot_iperf.sh iperf_result.json
mininet@mininet-vm:~/work/lab_iperf3/iperf3$ cd results
mininet@mininet-vm:~/work/lab_iperf3/iperf3/results$ ls
l.dat  cwnd.pdf  retransmits.pdf  RTT_Var.pdf
bytes.pdf  MTU.pdf  RTT.pdf  throughput.pdf
mininet@mininet-vm:~/work/lab_iperf3/iperf3/results$

```

Рис. 3.15: Графики из получившегося JSON-файла

Создал Makefile (рис. 3.16)

```
mininet@mininet-vm: ~/work/lab_iperf3/iperf3
GNU nano 4.8 Makefile
all: iperf_result.json plot

iperf_result.json:
    sudo python lab_iperf3.py

plot: iperf_result.json
    plot_iperf.sh iperf_result.json

clean:
    rm -f *.json *.csv
    rm -rf results
```

Рис. 3.16: Makefile

Проверка Makefile, все работает успешно (рис. 3.17)

```
mininet@mininet-vm:~/work/lab_iperf3/iperf3$ make
make: command not found
mininet@mininet-vm:~/work/lab_iperf3/iperf3$ make
sudo python lab_iperf3.py
*** Adding controller
*** Adding hosts
*** Adding switch
*** Creating links
(100.00Mbit 75ms delay 0.00000% loss) (100.00Mbit 75ms delay 0.00000% loss) (100.00Mbit 75ms delay 0.00000% loss) (100.00Mbit 75ms delay 0.00000% loss) (100.00Mbit 75ms delay 0.00000% loss) *** Starting network
*** Configuring hosts
h1 (cfs 100000/100000us) h2 (cfs 100000/100000us)
*** Starting controller
c0
*** Starting 1 switches
s3 (100.00Mbit 75ms delay 0.00000% loss) (100.00Mbit 75ms delay 0.00000% loss) .. (100.00Mbit 75ms delay 0.00000% loss) (100.00Mbit 75ms delay 0.00000% loss)
*** Waiting for switches to connect
s3
*** Traffic generation
*** h2 : ('iperf3 -s -D -1',)
*** h1 : ('iperf3 -c', '10.0.0.2', '-J > iperf_result.json')
Host h1 has IP address 10.0.0.1 and MAC address 72:eb:6e:1f:ac:5f
Host h2 has IP address 10.0.0.2 and MAC address da:0f:0e:6e:5f:56
*** Running CLI
*** Starting CLI:
mininet>
```

Рис. 3.17: Проверка Makefile

## 4 Выводы

Ознакомился с инструментом для измерения пропускной способности сети в режиме реального времени — iPerf3, а также получил навыки проведения воспроизводимого эксперимента по измерению пропускной способности моделируемой сети в среде Mininet.



## 5 Список литературы

### 1. Официальный сайт и репозиторий Mininet

- Mininet Official Website. – URL: <http://mininet.org/>
- Mininet GitHub Repository. – URL: <https://github.com/mininet/mininet>

### 2. Документация по системам виртуализации и графическому интерфейсу

- VirtualBox Official Manual. – URL: <https://www.virtualbox.org/manual/UserManual.html>
- X Window System Protocol Documentation. – URL: <https://www.x.org/releases/current/doc/xproto/x11protocol.html>

### 3. Вспомогательные ресурсы

- VcXsrv Windows X Server. – URL: <https://sourceforge.net/projects/vcxsrv/>
- Xming X Server. – URL: <http://www.straightrunning.com/XmingNotes/>