Отчёт по лабораторной работе №3

Измерение и тестирование пропускной способности сети. Воспроизводимый эксперимент

Ким Реачна

Содержание

Список иллюстраций

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 Выполнение лабораторной работы

1. Создание подкаталога и копирование скрипта lab_iperf3_topo.py для запуска моделирования, затем изменение скрипта для вывода отображаемой информации о хосте, IP-адресе и MAC-адресе:

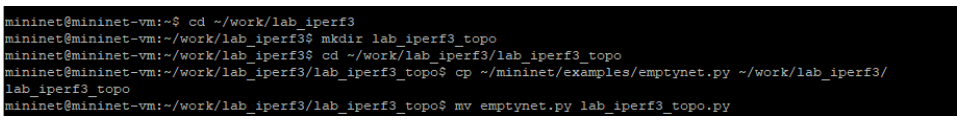

Рис. 2.1: Создание подкаталога и копирование скрипта lab_iperf3_topo.py

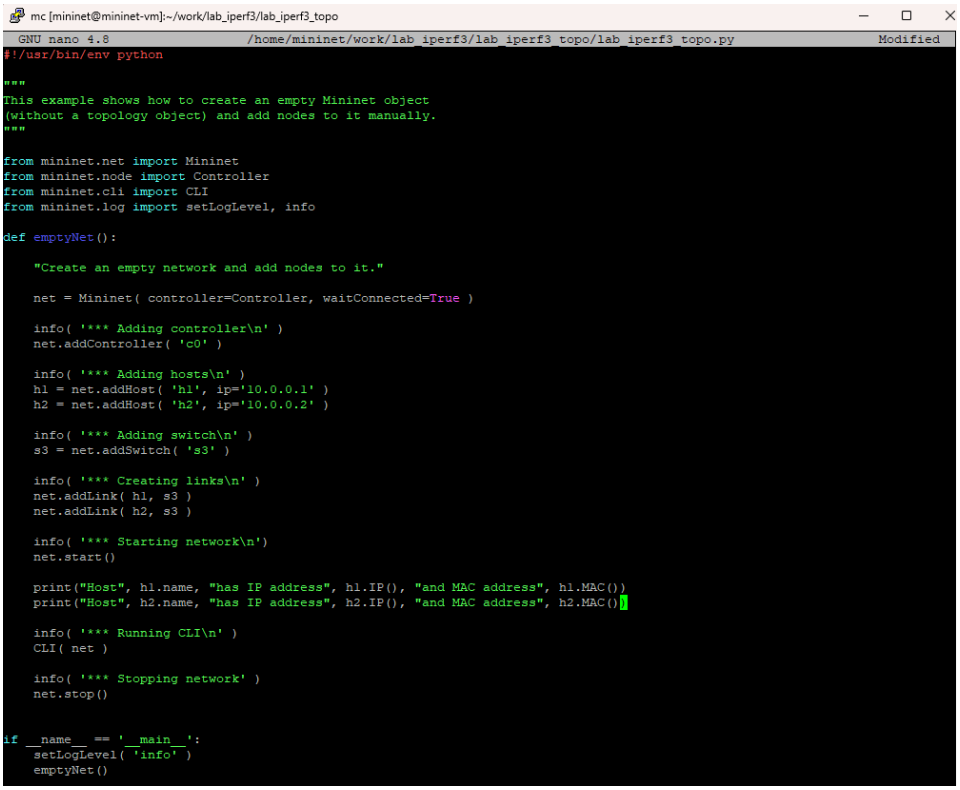

Рис. 2.2: Скрипт lab_iperf3_topo.py

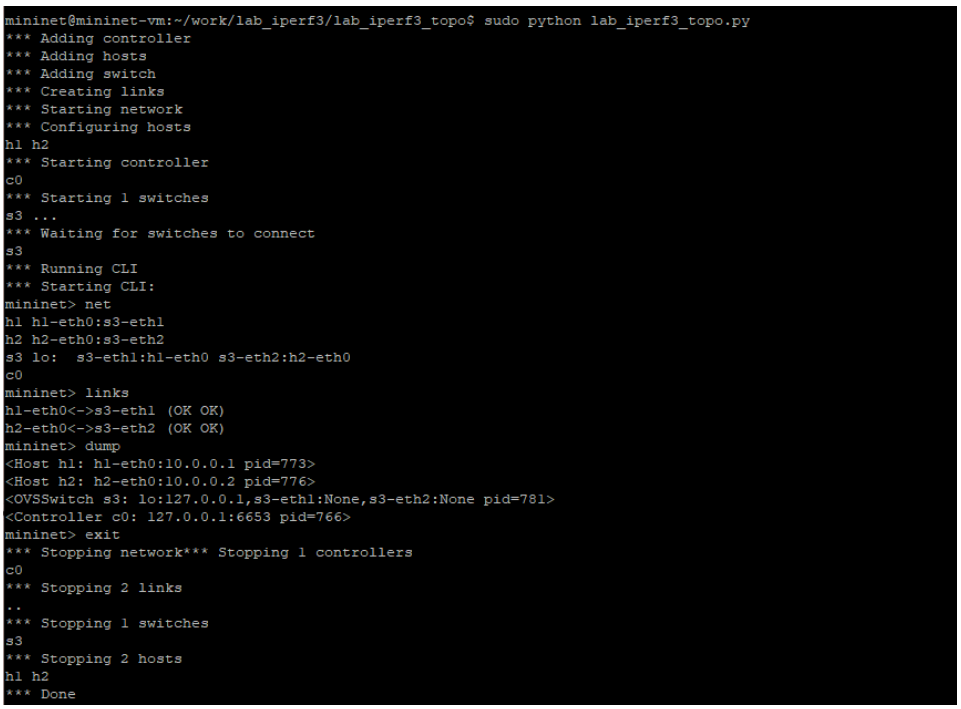

Рис. 2.3: Запуск скрипта lab_iperf3_topo.py до изменением

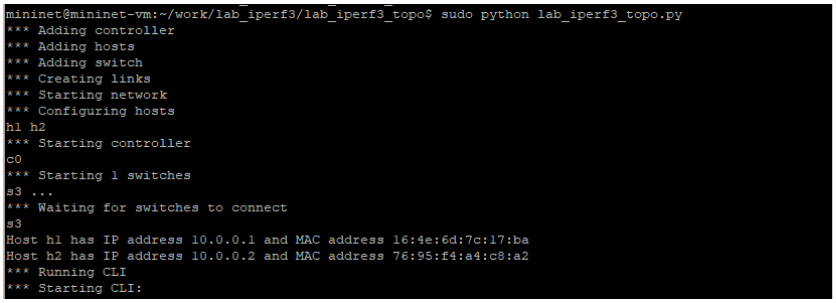

Рис. 2.4: Проверка корректность отработки после изменением

2. Изменение тополгию, скопировав в lab_iperf3_topo2.py, указание на использование ограничения производительности и изоляции, изменение функцию задания параметров виртуального хоста h1 и h2 и изменение функцию параметров соединения между хостом h1 и коммутатором s3:

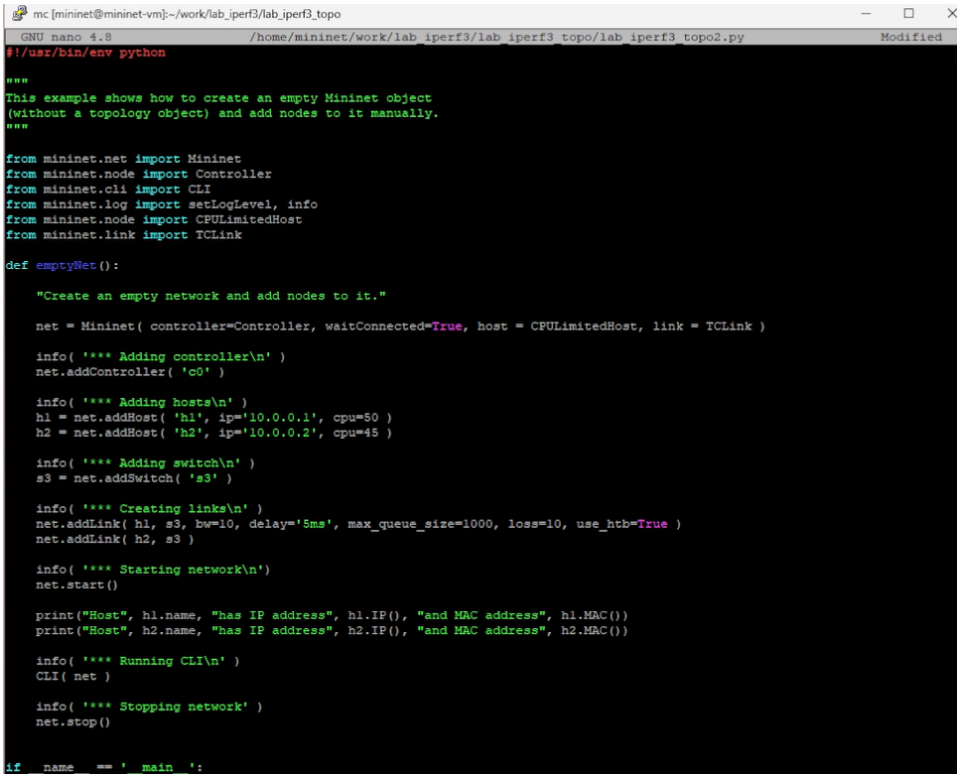

Рис. 2.5: Скрипт lab_iperf3_topo2.py

- Запуск на отработку сначала скрипт lab_iperf3_topo2.py, затем lab_iperf3_topo.py

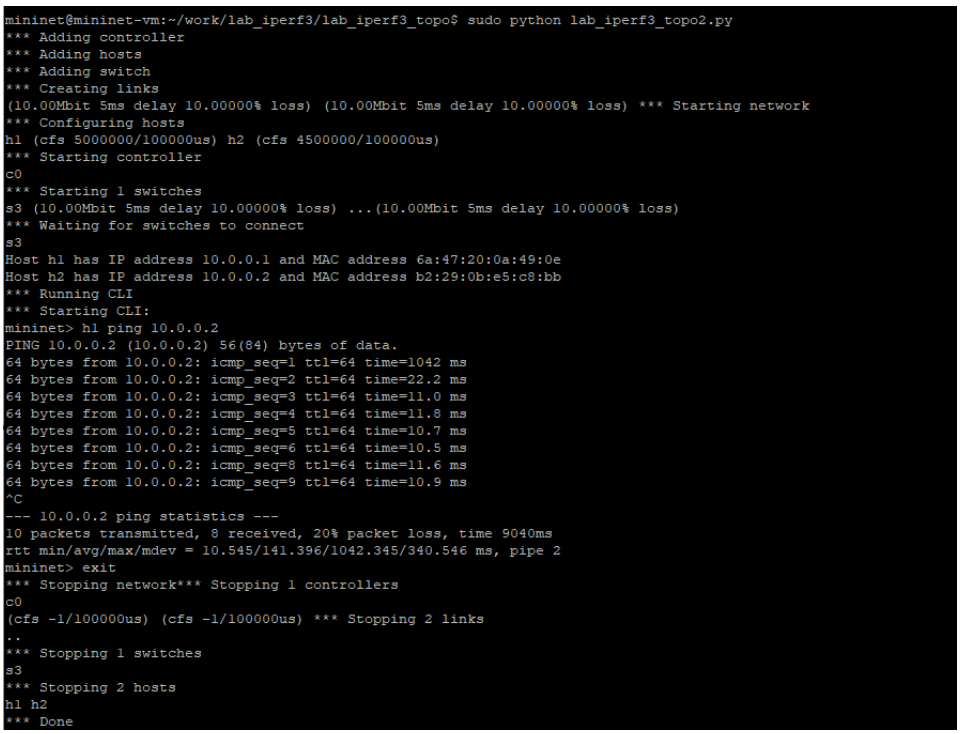

Рис. 2.6: Запуск скрипт создания топологии lab_iperf3_topo2.py

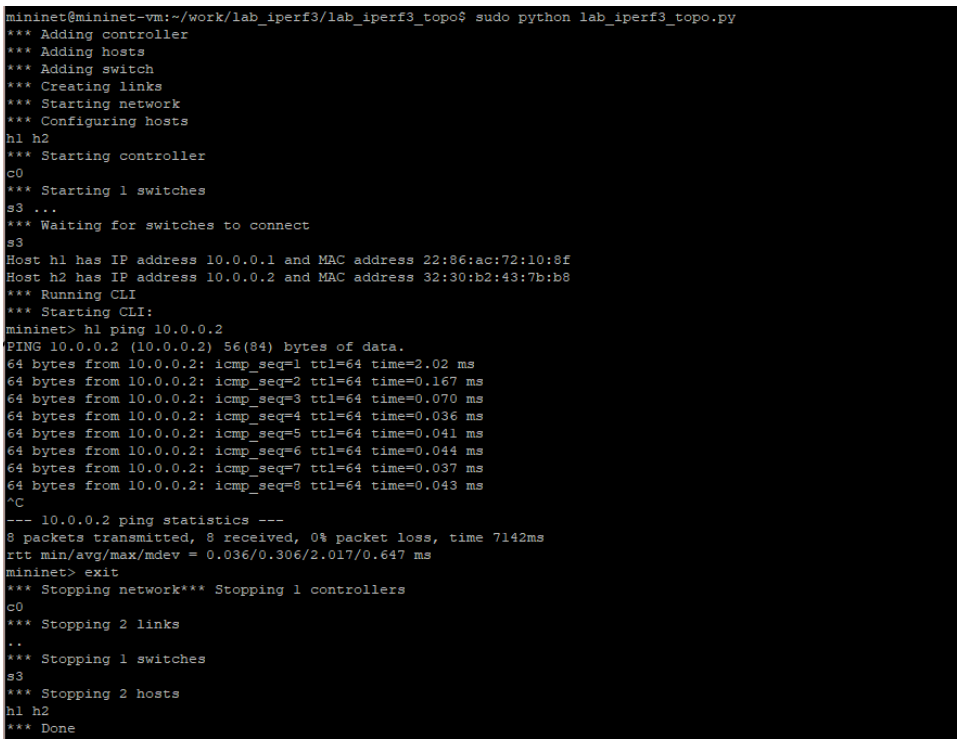

Рис. 2.7: Запуск скрипт создания топологии lab_iperf3_topo.py

3. Построение графика: изменение топологии, копирование в lab_iperf3.py , изменение параметров, создание Makefile для запуска всего эксперимента:

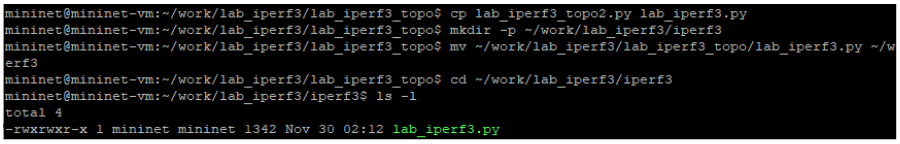

Рис. 2.8: Копирование в lab_iperf3.py

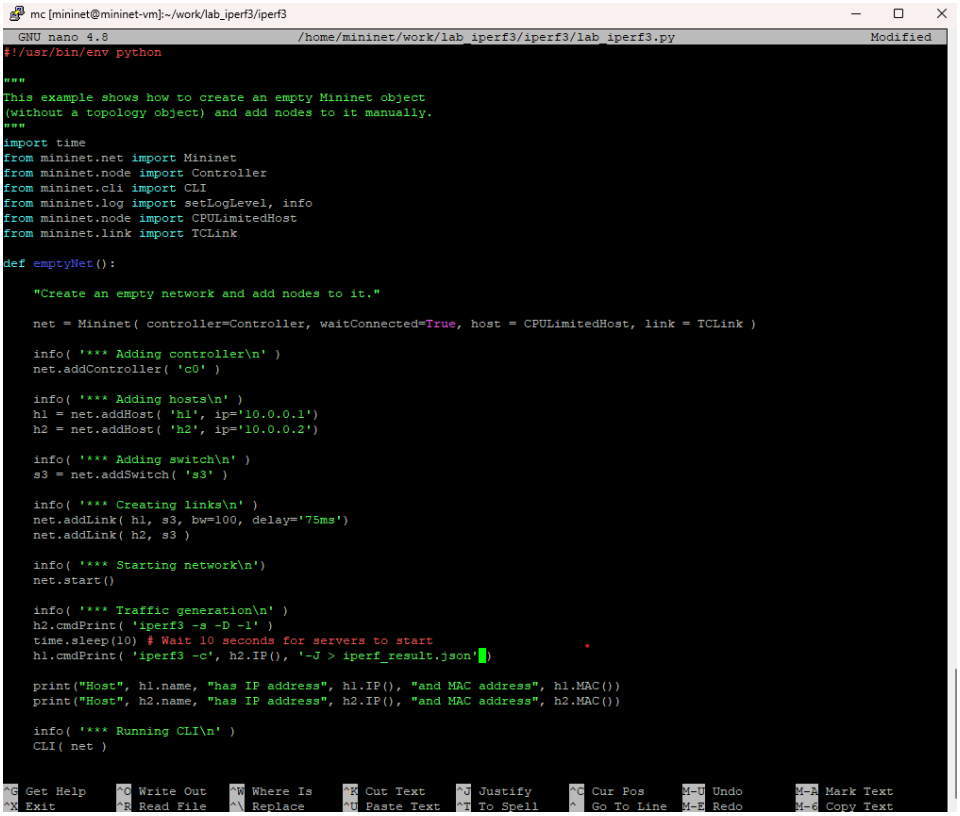

Рис. 2.9: Скрипт lab_iperf3.py

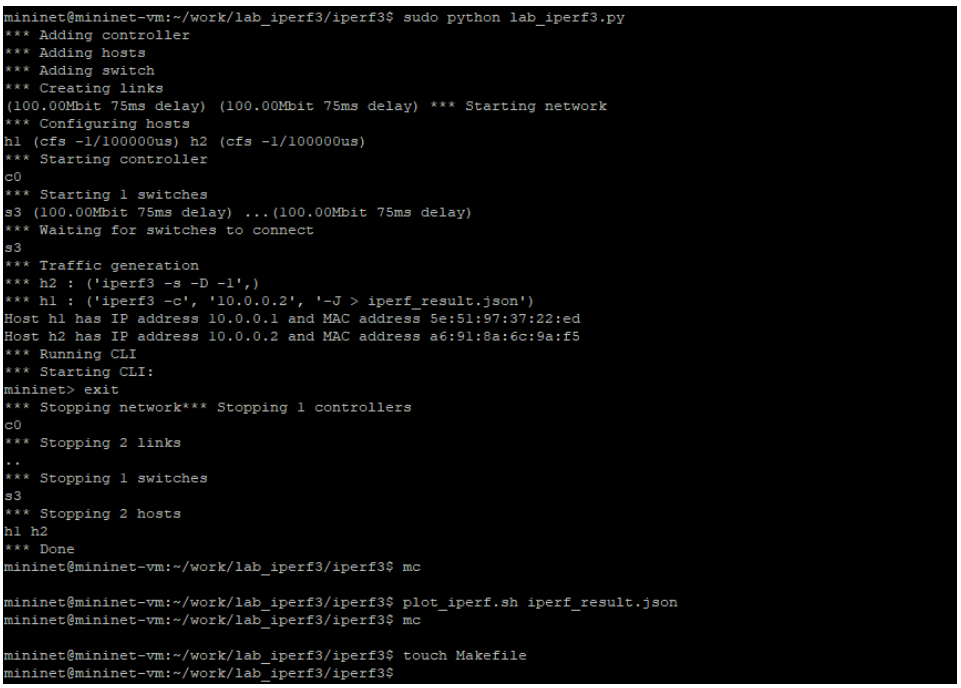

Рис. 2.10: Запуск скрипта lab_iperf3.py

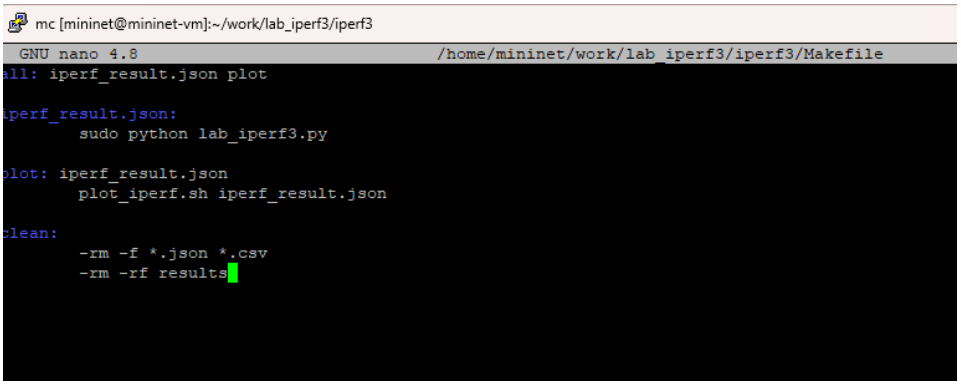

Рис. 2.11: Создание Makefile

- Проверка корректность отработки Makefile

```
mininet@mininet-vm:~/work/lab_iperf3/iperf3$ make clean
rm -f *.json *.csv
rm -rf results
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) (100.00Mbit 75ms delay) *** Starting network
*** Configuring hosts
h1 (cfs -1/100000us) h2 (cfs -1/100000us)
*** Starting controller
c0
*** Starting 1 switches
s3 (100.00Mbit 75ms delay) ...(100.00Mbit 75ms delay)
*** 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 ae:a7:7c:b2:40:2f
Host h2 has IP address 10.0.0.2 and MAC address d2:64:1b:78:0d:f4
*** Running CLI
*** Starting CLI:
mininet> exit
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s3
*** Stopping 2 hosts
h1 h2
*** Done
plot_iperf.sh iperf_result.json
```

Рис. 2.12: Проверка корректность отработки Makefile

4. Результат построения графика

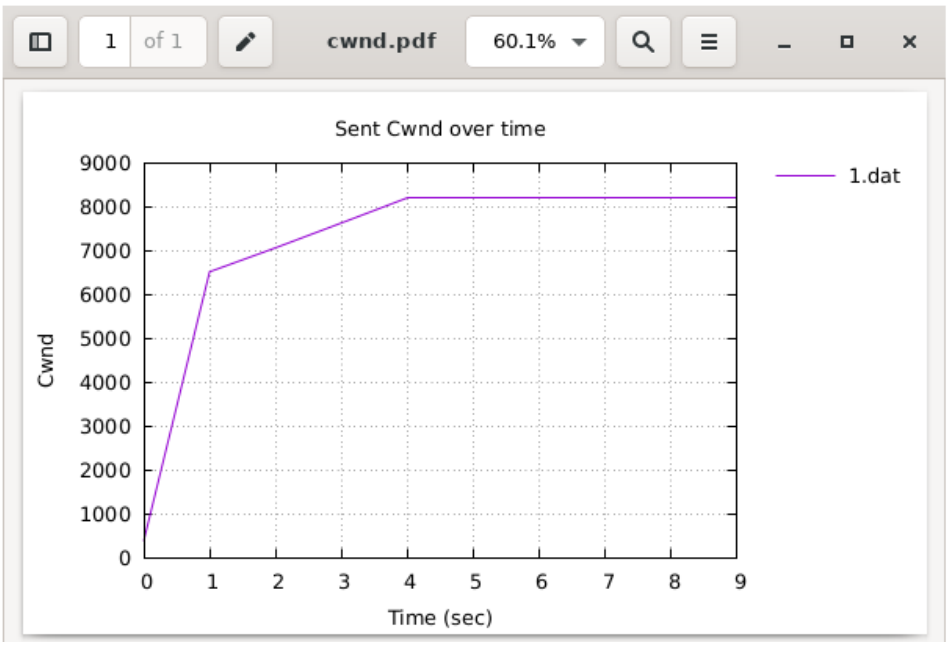

Рис. 2.13: Окно перегрузки

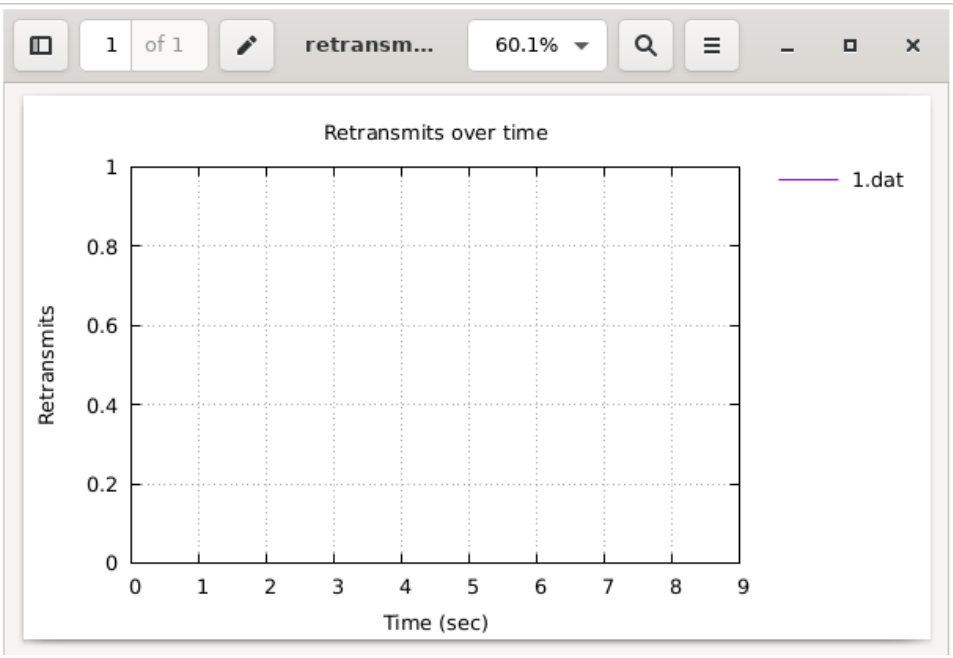

Рис. 2.14: Повторная передача

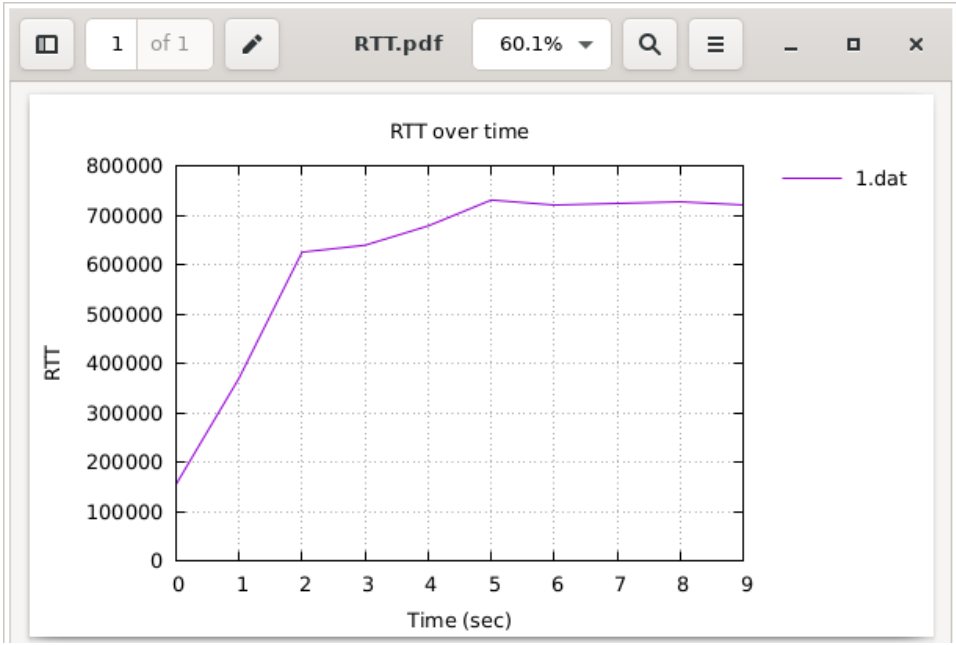

Рис. 2.15: Время приема-передачи

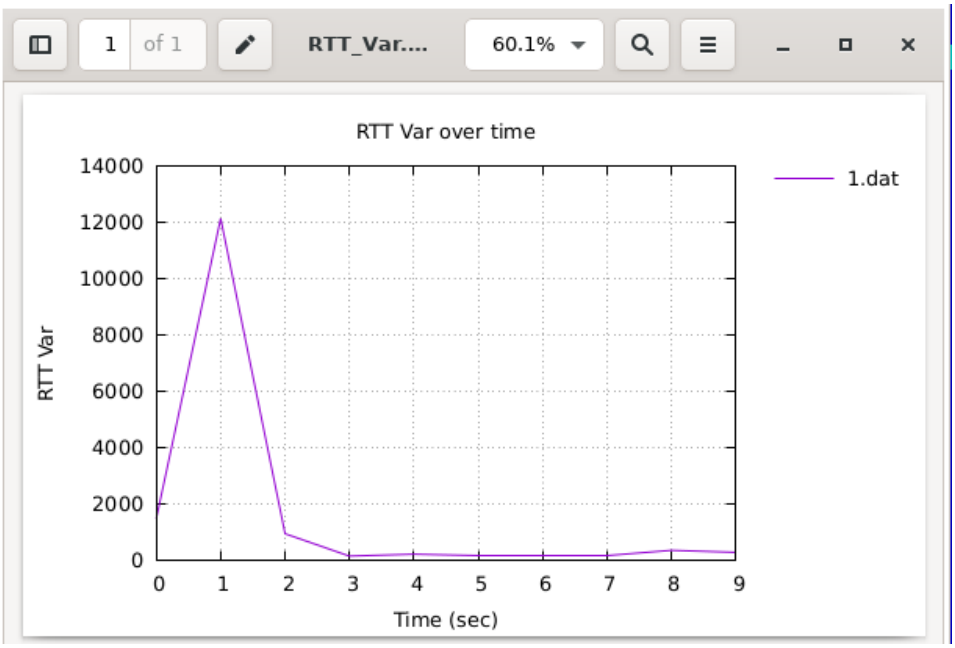

Рис. 2.16: Отклонение времени приема-передачи

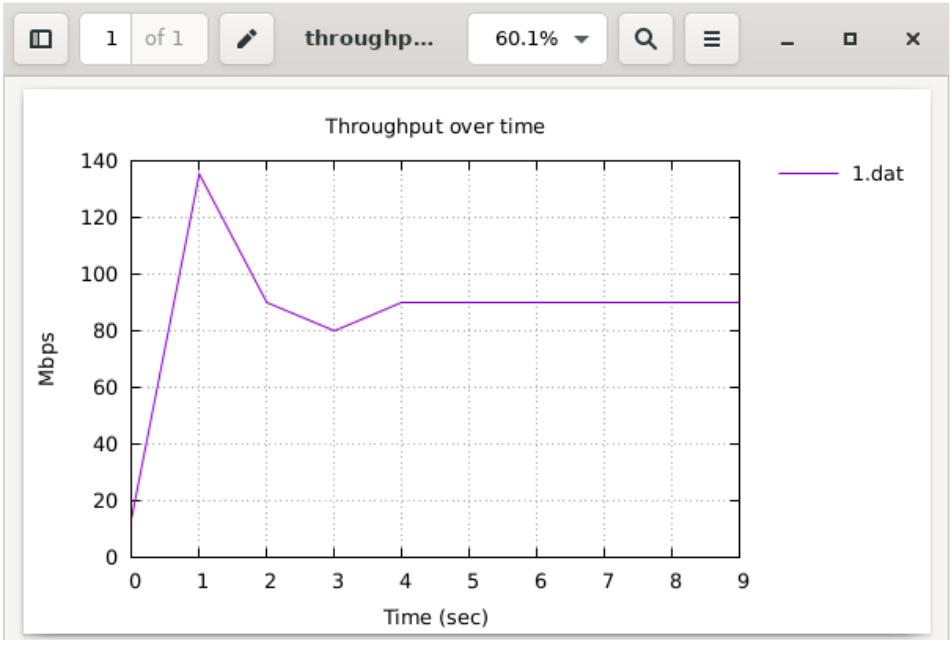

Рис. 2.17: Пропускная способность

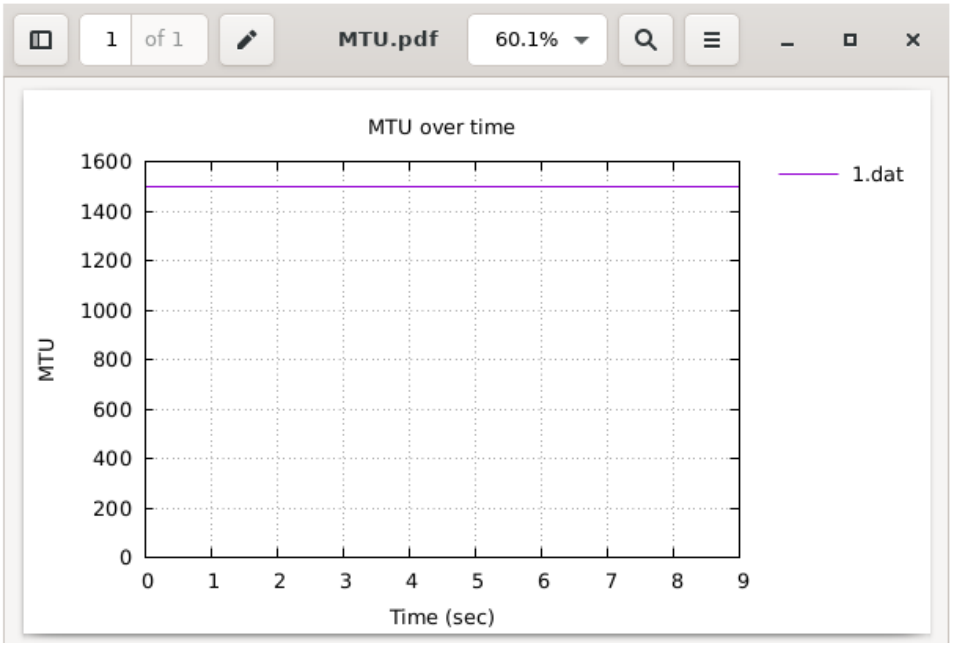

Рис. 2.18: Максимальная единица передачи

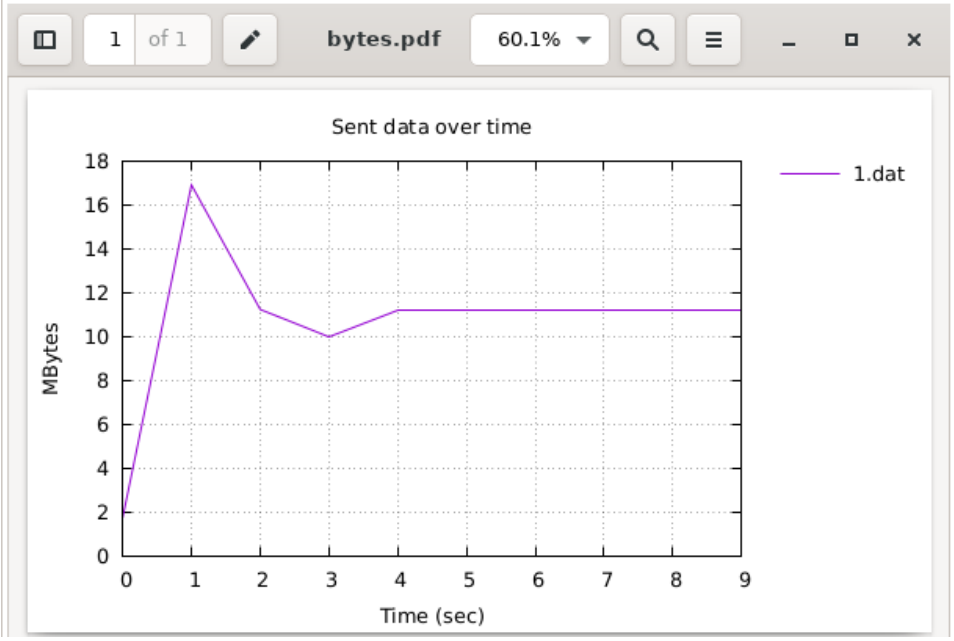

Рис. 2.19: Количество переданных байтов

3 Листинги программы

- Скрипт lab_iperf3_topo.py

```
#!/usr/bin/env python

"""
This example shows how to create an empty Mininet object
(without a topology object) and add nodes to it manually.
"""

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 )

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

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

    info( '*** Adding switch\n' )
    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())
    print("Host", h2.name, "has IP address", h2.IP(),
    "and MAC address", h2.MAC())

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

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

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

- Скрипт lab_iperf3_topo2.py

```
#!/usr/bin/env python

"""
This example shows how to create an empty Mininet object
(without a topology object) and add nodes to it manually.
"""

from mininet.net import Mininet
from mininet.node import Controller
from mininet.cli import CLI
from mininet.log import setLogLevel, info
from mininet.node import CPULimitedHost
from mininet.link import TCLink

def emptyNet():

    "Create an empty network and add nodes to it."

    net = Mininet( controller=Controller, waitConnected=True,
    host = CPULimitedHost, link = TCLink )

    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 )

    info( '*** Adding switch\n' )
```

```
    s3 = net.addSwitch( 's3' )

    info( '*** Creating links\n' )
    net.addLink( h1, s3, bw=10, delay='5ms', max_queue_size=1000,
    loss=10, use_htb=True )
    net.addLink( h2, s3 )

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

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

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

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

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

- Скрипт lab_iperf3.py

```
#!/usr/bin/env python

"""
This example shows how to create an empty Mininet object
```

```
(without a topology object) and add nodes to it manually.
"""
import time
from mininet.net import Mininet
from mininet.node import Controller
from mininet.cli import CLI
from mininet.log import setLogLevel, info
from mininet.node import CPULimitedHost
from mininet.link import TCLink

def emptyNet():

    "Create an empty network and add nodes to it."

    net = Mininet( controller=Controller, waitConnected=True,
    host = CPULimitedHost, link = TCLink )

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

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

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

    info( '*** Creating links\n' )
    net.addLink( h1, s3, bw=100, delay='75ms')
```

```
    net.addLink( h2, s3 )

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

    info( '*** Traffic generation\n' )
    h2.cmdPrint( 'iperf3 -s -D -1' )
    time.sleep(10) # Wait 10 seconds for servers to start
    h1.cmdPrint( 'iperf3 -c', h2.IP(), '-J > iperf_result.json' )

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

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

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

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

4 Вывод

Я познакомилась 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.