

# Лабораторная работа №3

Измерение и тестирование пропускной способности сети. Воспроизводимый эксперимент

Барабанова Кристина

Российский университет дружбы народов, Москва, Россия

Информация

## Докладчик

Барабанова Кристина, студент группы НФИбд-02-22

## 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.

## Задание

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

Выполнение лабораторной работы

С помощью API Mininet создайте простейшую топологию сети, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8: – В каталоге /work/lab\_iperf3 для работы над проектом создайте подкаталог lab\_iperf3\_topo и скопируйте в него файл с примером скрипта mininet/examples/emphynet.py, описывающего стандартную простую топологию сети mininet:

```
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/emphynet.py ~/work/lab_iperf3/lab_iperf3_topo
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ mv emphynet.py lab_iperf3_topo.py
```

Рис. 1: рис. 1

## - Изучила содержание скрипта lab\_iperf3\_topo.py:

```
GNU nano 4.8 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()

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

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



- Запустила скрипт создания топологии lab\_iperf3\_topo.py:

```
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ sudo py  
thon 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: рис. 3

- После отработки скрипта посмотрела элементы топологии и завершите работу mininet:

```
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=870>
<Host h2: h2-eth0:10.0.0.2 pid=872>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None pid=877>
<Controller c0: 127.0.0.1:6653 pid=863>
mininet> exit
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s3
*** Stopping 2 hosts
h1 h2
*** Done
```

Рис. 4: рис. 4

Внесите в скрипт lab\_iperf3\_topo.py изменение, позволяющее вывести на экран информацию о хосте h1, а именно имя хоста, его IP-адрес, MACадрес. Для этого после строки, задающей старт работы сети, добавьте строку:

```
info( '*** Starting network\n')  
net.start()  
  
print( "Host", h1.name, "has IP address", h1.IP(), "and MAC address">
```

Рис. 5: рис. 5

Здесь: – IP() возвращает IP-адрес хоста или определенного интерфейса; – MAC() возвращает MAC-адрес хоста или определенного интерфейса.

Проверьте корректность отработки изменённого скрипта.

```
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 f6:d4:84:ca:e3:b6
*** Running CLI
*** Starting CLI:
```

Рис. 6: рис. 6

Измените скрипт lab\_iperf3\_topo.py так, чтобы на экран выводилась информация об имени, IP-адресе и MAC-адресе обоих хостов сети. Проверьте корректность отработки изменённого скрипта.

```
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() )
```

Рис. 7: рис. 7

```
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 ae:e2:2b:23:01:b9
Host h2 has IP address 10.0.0.2 and MAC address de:2e:74:07:7e:46
*** Running CLI
*** Starting CLI:
```

Рис. 8: рис. 8

Mininet предоставляет функции ограничения производительности и изоляции с помощью классов CPULimitedHost и TCLink. Добавьте в скрипт настройки параметров производительности: – Сделайте копию скрипта lab\_iperf3\_topo.py:

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

Рис. 9: рис. 9

- В начале скрипта lab\_iperf3\_topo2.py добавьте записи об импорте классов CPULimitedHost и TCLink

```
GNU nano 4.8 lab_iperf3_topo2.py Modified
#!/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.node import CPULimitedHost
from mininet.link import TCLink
```

Рис. 10: рис. 10

В скрипте lab\_iperf3\_topo2.py измените строку описания сети, указав на использование ограничения производительности и изоляции

В скрипте lab\_iperf3\_topo2.py измените функцию задания параметров виртуального хоста h1, указав, что ему будет выделено 50% от общих ресурсов процессора системы:

Аналогичным образом для хоста h2 задайте долю выделения ресурсов процессора в 45%.

В скрипте lab\_iperf3\_topo2.py измените функцию параметров соединения между хостом h1 и коммутатором s3

```
GNU nano 4.8 lab_iperf3_topo2.py Modified
"""
This example shows how to create an empty Mininet object
(without a topology object) and add nodes to it manually.
"""

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, 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 )
```



– Запустите на отработку сначала скрипт lab\_iperf3\_topo2.py, затем lab\_iperf3\_topo.py и сравните результат.

```
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 10.00000% loss) (10.00Mbit 5ms delay 10.00000% loss) *** Starting network
*** Configuring hosts
h1 (cfs 5000000/1000000us) h2 (cfs 4500000/1000000us)
*** Starting controller
c0
*** Starting 1 switches
s3 (10.00Mbit 5ms delay 10.00000% loss) ... (10.00Mbit 5ms delay 10.00000% loss)
*** Waiting for switches to connect
s3
Host h1 has IP address 10.0.0.1 and MAC address b6:9c:5d:12:5a:ad
Host h2 has IP address 10.0.0.2 and MAC address 06:3b:c8:74:b9:5e
*** Running CLI
*** Starting CLI:
mininet> net
h1 h1-eth0:s3-eth1
h2 h2-eth0:s3-eth2
s3 lo: s3-eth1:h1-eth0 s3-eth2:h2-eth0
c0
mininet>
*** Stopping network*** Stopping 1 controllers
c0
(cfs -1/1000000us) (cfs -1/1000000us) *** Stopping 2 links
..
*** Stopping 1 switches
s3
*** Stopping 2 hosts
h1 h2
*** Done
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
```

Постройте графики по проводимому эксперименту: – Сделайте копию скрипта lab\_iperf3\_topo2.py и поместите его в подкаталог iperf:

```
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_iperf3_topo/lab_iperf3.py ~/work/lab_iperf3/iperf3
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ cd ~/work/lab_iperf3/iperf3
mininet@mininet-vm:~/work/lab_iperf3/iperf3$ ls -l
total 4
-rwxrwxr-x 1 mininet mininet 1351 Jan 30 00:57 lab_iperf3.py
mininet@mininet-vm:~/work/lab_iperf3/iperf3$ |
```

Рис. 13: рис. 13

– Измените код в скрипте lab\_iperf3.py так, чтобы:

- на хостах не было ограничения по использованию ресурсов процессора;
- каналы между хостами и коммутатором были по 100 Мбит/с с задержкой 75 мс, без потерь, без использования ограничителей пропускной способности и максимального размера очереди.

– После функции старта сети опишите запуск на хосте h2 сервера iPerf3, а на хосте h1 запуск с задержкой в 10 секунд клиента iPerf3 с экспортом результатов в JSON-файл, прокомментируйте строки, отвечающие за запуск CLI-интерфейса:

```
def emptyNet():  
  
    "Create an empty network and add nodes to it."  
  
    net = Mininet( controller=Controller, waitConnected=True, host = CPULimitedHost, l2  
  
    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, bw=100, delay='75ms' )  
  
    info( '*** Starting network\n' )  
    net.start()  
  
    info( '*** Starting network\n' )  
  
    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' )
```

Запустите на отработку скрипт lab\_iperf3.py:

```
mininet@mininet-vm:~/work/lab_iperf3/iperf3$ sudo python lab_iperf3.py
*** Adding controller
*** Adding hosts
*** Adding switch
*** Creating links
(100.00Mbit 75ms delay) (100.00Mbit 75ms delay) (100.00Mbit 75ms delay) (100.00Mbit 75ms
delay) *** Starting network
*** Configuring hosts
h1 (cfs -1/1000000us) h2 (cfs -1/1000000us)
*** Starting controller
c0
*** Starting 1 switches
s3 (100.00Mbit 75ms delay) (100.00Mbit 75ms delay) ...(100.00Mbit 75ms delay) (100.00Mbi
t 75ms delay)
*** Waiting for switches to connect
s3
*** Starting network
*** 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 26:0c:9b:74:71:13
Host h2 has IP address 10.0.0.2 and MAC address 86:a2:f9:a3:5e:eb
*** Running CLI
*** Starting CLI:
mininet>
```

Рис. 15: рис. 15

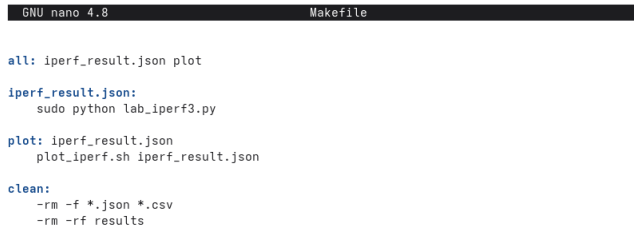
Постройте графики из получившегося JSON-файла:

```
-----  
mininet@mininet-vm:~/work/lab_iperf3/iperf3$ plot_iperf.sh iperf_result.json
```

Рис. 16: рис. 16

Создайте Makefile для проведения всего эксперимента:

В Makefile пропишите запуск скрипта эксперимента, построение графиков и очистку каталога от результатов:



```
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. 17: рис. 17

## Проверьте 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) (100.00Mbit 75ms delay) (100.00Mbit 75ms
delay) *** Starting network
*** Configuring hosts
h1 (cfs -1/1000000us) h2 (cfs -1/1000000us)
*** Starting controller
c0
*** Starting 1 switches
s3 (100.00Mbit 75ms delay) (100.00Mbit 75ms delay) ...(100.00Mbit 75ms delay) (100.00Mbi
t 75ms delay)
*** Waiting for switches to connect
s3
*** Starting network
*** 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 b6:0d:4e:3b:fc:a8
Host h2 has IP address 10.0.0.2 and MAC address aa:71:9e:7e:c2:42
*** Running CLI
*** Starting CLI:
mininet>
```

Рис. 18: рис. 18

## Вывод

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