

Лабораторная работа 3

Петрушов Дмитрий, 1032212287

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Вывод	16
	Список литературы	17

Список иллюстраций

2.1	Создание подкаталога, копирование файла с примером скрипта (описывающего стандартную простую топологию сети mininet) .	6
2.2	Внесение изменения в скрипт, позволяющего вывести на экран информацию о хосте h1 (имя, IP-адрес, MAC-адрес)	7
2.3	Проверка корректности отработки скрипта	8
2.4	Внесение изменения в скрипт, позволяющего вывести на экран информацию о двух хостах (имя, IP-адрес, MAC-адрес)	9
2.5	Проверка корректности отработки скрипта	9
2.6	Описание запуска на хосте h2 сервера iPerf3, на хосте h1 запуска с задержкой в 10 секунд клиента iPerf3 с экспортом результатов в JSON-файл. Комментирование строк, отвечающих за запуск CLI-интерфейса	12
2.7	Запуск скрипта lab_iperf3.py на отработку	13
2.8	Добавление скрипта в Makefile	14
2.9	Проверка корректности отработки Makefile	15

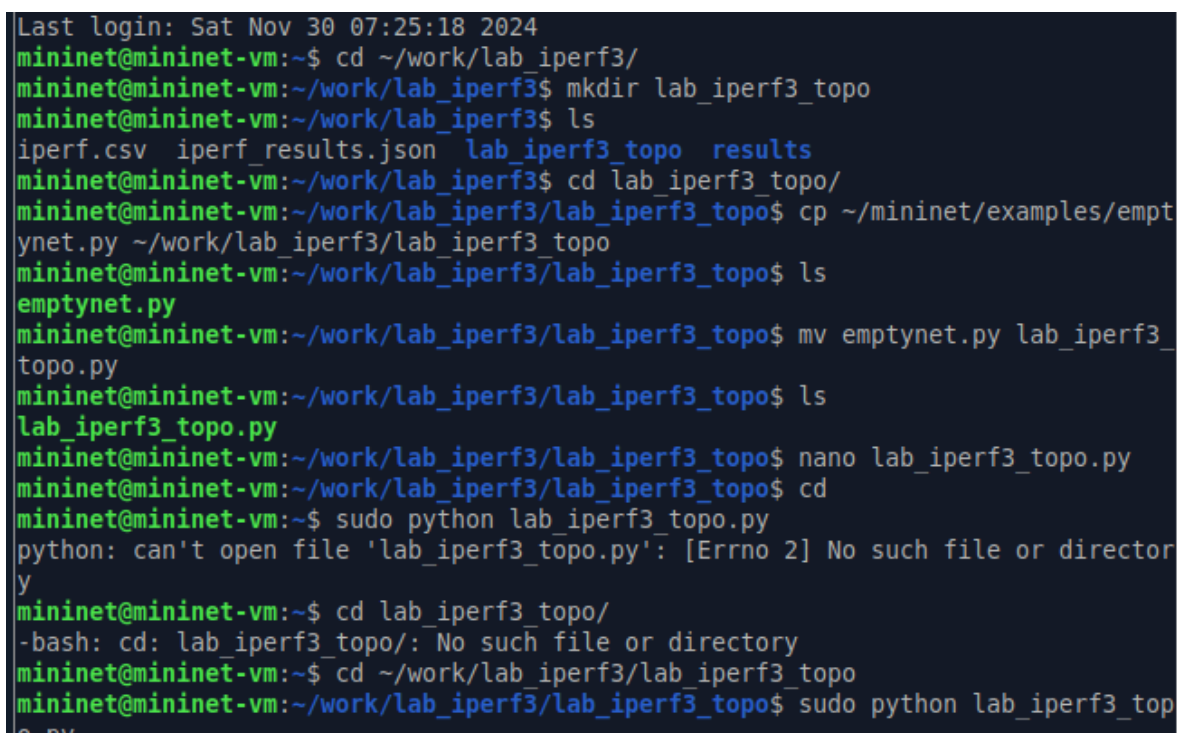
Список таблиц

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 Выполнение лабораторной работы

С помощью API Mininet создадим простейшую топологию сети, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8. Для этого в каталоге /work/lab_iperf3 для работы над проектом создадим подкаталог lab_iperf3_topo и скопируем в него файл с примером скрипта mininet/examples/emphynet.py, описывающего стандартную простую топологию сети mininet (рис. [2.1]):



```
Last login: Sat Nov 30 07:25:18 2024
mininet@mininet-vm:~$ cd ~/work/lab_iperf3/
mininet@mininet-vm:~/work/lab_iperf3$ mkdir lab_iperf3_topo
mininet@mininet-vm:~/work/lab_iperf3$ ls
iperf.csv  iperf_results.json  lab_iperf3_topo  results
mininet@mininet-vm:~/work/lab_iperf3$ cd 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$ ls
emphynet.py
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ mv emphynet.py lab_iperf3_topo.py
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ ls
lab_iperf3_topo.py
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ nano lab_iperf3_topo.py
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ cd
mininet@mininet-vm:~$ sudo python lab_iperf3_topo.py
python: can't open file 'lab_iperf3_topo.py': [Errno 2] No such file or directory
mininet@mininet-vm:~$ cd lab_iperf3_topo/
-bash: cd: lab_iperf3_topo/: No such file or directory
mininet@mininet-vm:~$ cd ~/work/lab_iperf3/lab_iperf3_topo
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ sudo python lab_iperf3_topo.py
```

Рис. 2.1: Создание подкаталога, копирование файла с примером скрипта (описывающего стандартную простую топологию сети mininet)

Следующим шагом внесём в скрипт `lab_iperf3_toro.py` изменение, позволяющее вывести на экран информацию о хосте `h1`, а именно имя хоста, его IP-адрес, MAC-адрес. Для этого после строки, задающей старт работы сети, добавим нужную строку (рис. [2.2]):

```
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() )

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

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

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

Рис. 2.2: Внесение изменения в скрипт, позволяющего вывести на экран информацию о хосте `h1` (имя, IP-адрес, MAC-адрес)

Проверим корректность отработки изменённого скрипта (рис. [2.3]):

```
h1 h2
*** Done
mininet@mininet-vm:~/work/lab_iperf3/lab_iperf3_topo$ nano lab_iperf3_topo.py
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 be:da:bc:2c:3a:f3
*** Running CLI
*** Starting CLI:
```

Рис. 2.3: Проверка корректности отработки скрипта

Затем изменим скрипт `lab_iperf3_topo.py` так, чтобы на экран выводилась информация об имени, IP-адресе и MAC-адресе обоих хостов сети и проверим корректность отработки изменённого скрипта (рис. [2.4] - рис. [2.5]):


```

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__':

```

Рис. 2.4: Внесение изменения в скрипт, позволяющего вывести на экран информацию о двух хостах (имя, IP-адрес, MAC-адрес)

```

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 82:57:1a:08:37:79
Host h2 has IP address 10.0.0.2 and MAC address fa:1a:5b:99:f6:0c
*** Running CLI
*** Starting CLI:
mininet>

```

Рис. 2.5: Проверка корректности отработки скрипта

Mininet предоставляет функции ограничения производительности и изоляции с помощью классов `CPUimitedHost` и `TCLink`. Добавим в скрипт на-

стройки параметров производительности. Для начала сделаем копию скрипта lab_iperf3_topo.py:

В начале скрипта lab_iperf3_topo2.py добавим записи об импорте классов CPULimitedHost и TCLink. Далее изменим строку описания сети, указав на использование ограничения производительности и изоляции. Следующим шагом изменим функцию задания параметров виртуального хоста h1, указав, что ему будет выделено 50% от общих ресурсов процессора системы. Аналогичным образом для хоста h2 зададим долю выделения ресурсов процессора в 50%. В конце изменим функцию параметров соединения между хостом h1 и коммутатором s3 (рис. [??]):

Изменение скрипта lab_iperf3_topo2.py: добавление ипорта классов, изменение строки описания сети, изменение функции задания параметров виртуального хоста h1 и h2, изменение функции параметров соединения между хостом h1 и коммутатором s3

Запустим на отработку скрипт lab_iperf3_topo2.py (рис. [??] - рис. [??]):

```
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 5000000/1000000us)
*** 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 46:45:43:72:bd:c4
Host h2 has IP address 10.0.0.2 and MAC address 3a:6f:a2:1d:47:da
*** Running CLI
*** Starting CLI:
```

```
*** 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 96:c3:97:0f:29:cb
Host h2 has IP address 10.0.0.2 and MAC address 12:b6:36:13:d0:d5
*** Running CLI
*** Starting CLI:
```

Перед завершением лабораторной работы, построим графики по проводимому эксперименту. Для этого сделаем копию скрипта `lab_iperf3_topo2.py` и поместим его в подкаталог `iperf`. В начале скрипта `lab_iperf3.py` добавим запись об импорте `time` и изменим код в скрипте 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 опишем запуск на хосте `h2` сервера `iPerf3`, а на хосте `h1` запуск с задержкой в 10 секунд клиента `iPerf3` с экспортом результатов в JSON-файл, закомментируем строки, отвечающие за запуск CLI-интерфейса (рис. [2.6]):

```

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( '*** 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' )

    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 )

```

Рис. 2.6: Описание запуска на хосте h2 сервера iPerf3, на хосте h1 запуска с поддержкой в 10 секунд клиента iPerf3 с экспортом результатов в JSON-файл. Комментирование строк, отвечающих за запуск CLI-интерфейса

Запустим на отработку скрипт lab_iperf3.py (рис. [2.7]):

```

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) *** Starting network
*** Configuring hosts
h1 (cfs -l/1000000us) h2 (cfs -l/1000000us)
*** Starting controller
c0
*** Starting 1 switches
s3 (100.00Mbit 75ms delay) ...(100.00Mbit 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 be:fb:32:25:c1:7d
Host h2 has IP address 10.0.0.2 and MAC address 8e:5c:ef:d5:5f:ce
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s3
*** Stopping 2 hosts
h1 h2
*** Done

```

Рис. 2.7: Запуск скрипта lab_iperf3.py на отработку

Построим графики из получившегося JSON-файла и создадим Makefile для проведения всего эксперимента:

В Makefile пропишем запуск скрипта эксперимента, построение графиков и очистку каталога от результатов (рис. [2.8]):

```
GNU nano 4.8                                     maketitle
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. 2.8: Добавление скрипта в Makefile

Проверим корректность отработки Makefile (рис. [2.9]):

```

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 -1/1000000us) h2 (cfs -1/1000000us)
*** Starting controller
c0
*** Starting 1 switches
s3 (100.00Mbit 75ms delay) ...(100.00Mbit 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 46:e0:53:83:52:3e
Host h2 has IP address 10.0.0.2 and MAC address 32:38:dc:8f:96:d4
*** 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
mininet@mininet-vm:~/work/lab_iperf3/iperf3$

```

Рис. 2.9: Проверка корректности отработки Makefile

3 Вывод

В ходе выполнения лабораторной работы познакомились с инструментом для измерения пропускной способности сети в режиме реального времени — iPerf3, а также получили навыки проведения воспроизводимого эксперимента по измерению пропускной способности моделируемой сети в среде Mininet

Список литературы