Лабораторная работа №6

Настройка пропускной способности глобальной сети с помощью Token Bucket Filter

Хрусталев В.Н.

Российский университет дружбы народов, Москва, Россия

Информация

Докладчик

- Хрусталев Влад Николаевич
- студент
- Российский университет дружбы народов
- 1132222011@pfur.ru

Цель работы

Цель работы

Основной целью работы является знакомство с принципами работы дисциплины очереди Token Bucket Filter, которая формирует входящий/исходящий трафик для ограничения пропускной способности, а также получение навыков моделирования и исследования поведения трафика посредством проведения интерактивного и воспроизводимого экспериментов в Mininet.

Выполнение лабораторной работы

Настройка X-соединения

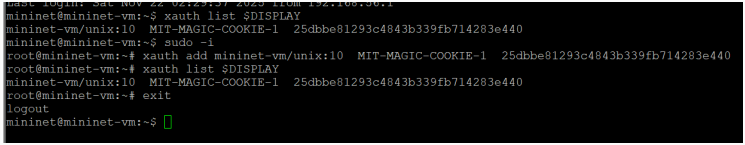


Рис. 1: Настройка X-соединения

Запуск простейшей топологии

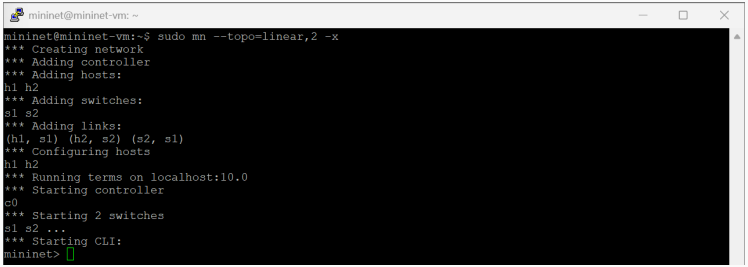


Рис. 2: Запуск простейшей топологии

Информация на хостах и коммутаторах об интерфейсах

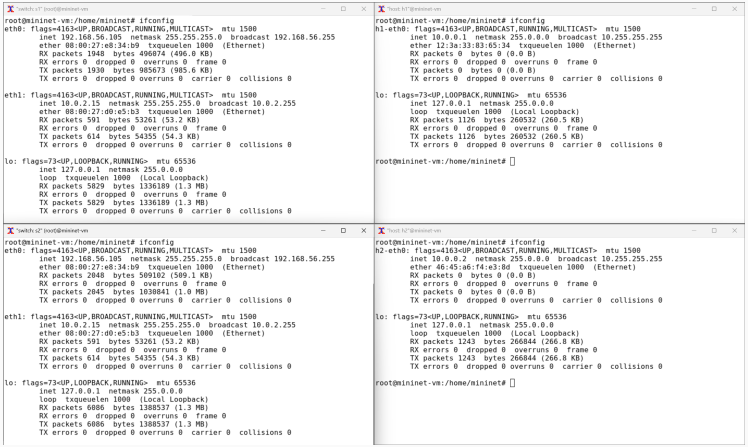
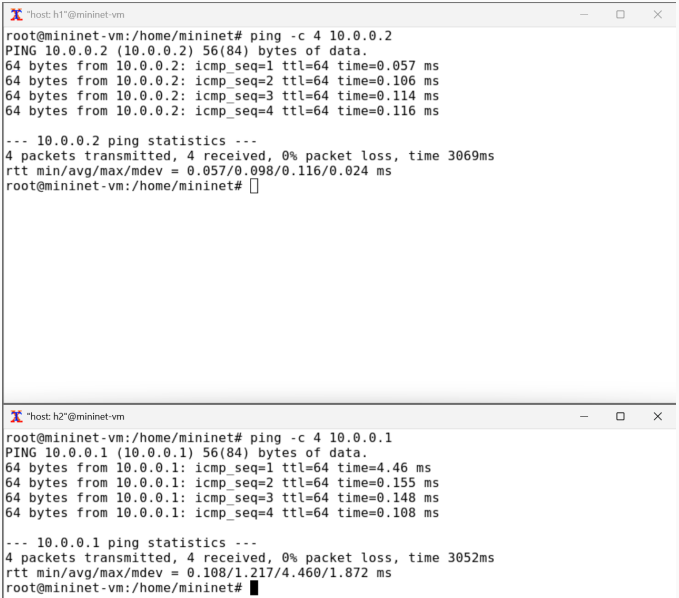
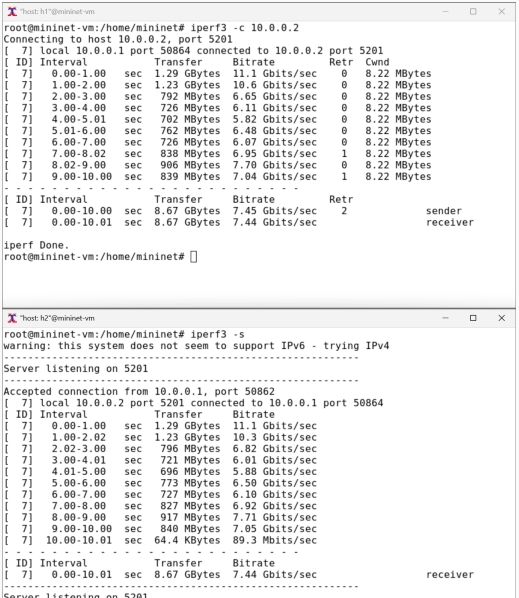


Рис. 3: Информация на хостах и коммутаторах об интерфейсах

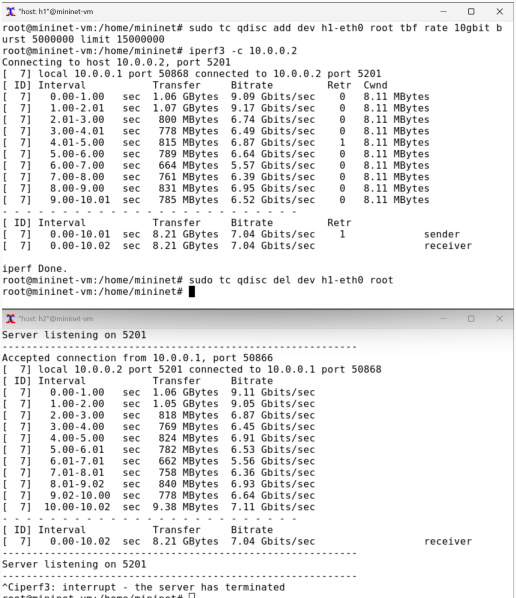
Тестирование соединения между h1 и h2



Проверка изначальной пропускной способности



Ограничение скорости на конечных хостах и тест



Ограничение скорости на коммутаторе и тест

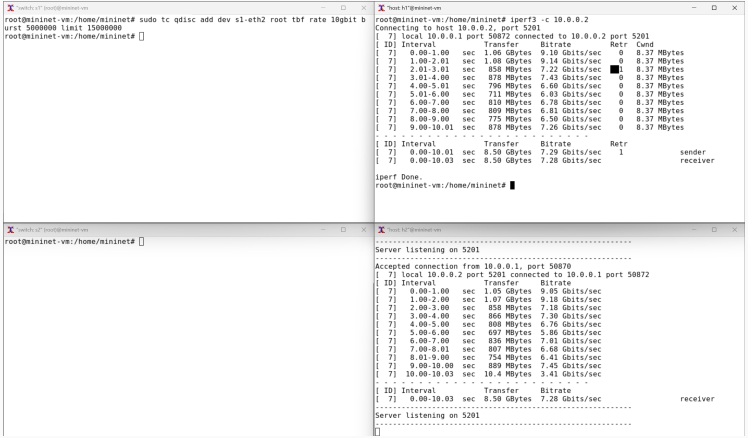


Рис. 7: Ограничение скорости на коммутаторе и тест

Удаление модифицированной конфигурации на коммутаторе s1

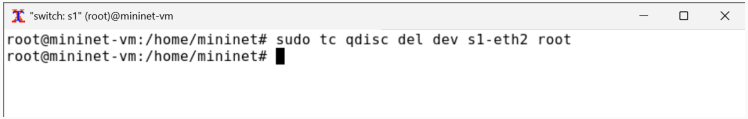


Рис. 8: Удаление модифицированной конфигурации на коммутаторе s1

Добавление NETEM правила задержки пакетов

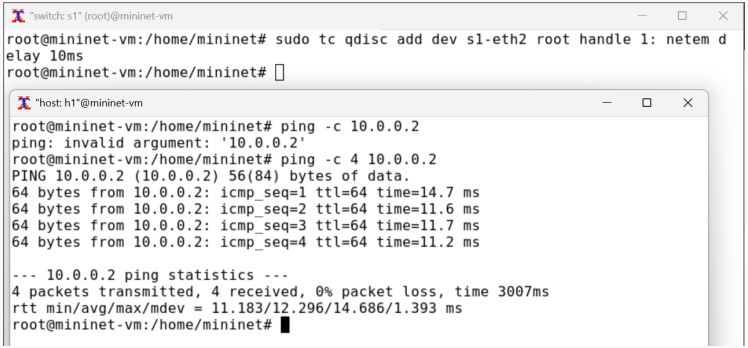


Рис. 9: Добавление NETEM правила задержки пакетов

Добавление 2-го правила ограничения скорости и тест

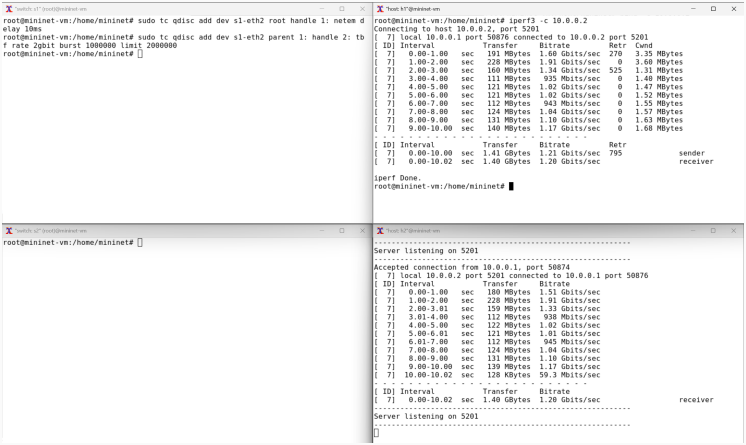


Рис. 10: Добавление 2-го правила ограничения скорости и тест

Удаление модифицированной конфигурации на коммутаторе s1

Рис. 11: Удаление модифицированной конфигурации на коммутаторе s1

Создание необходимых файлов

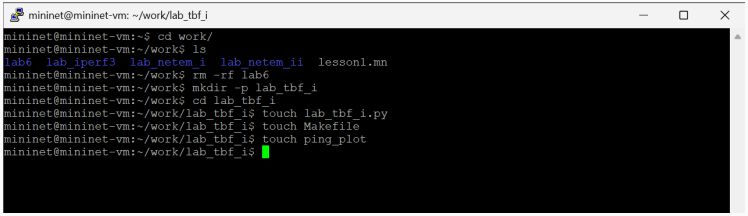


Рис. 12: Создание необходимых файлов

Листинг exp1/lab_tbf_i.py

mininet@mininet-vm: ~/work/lab_tbf_i/exp1

GNU nano 4.8 lab_tbf_i.py

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

"""
Simple experiment.
Output: ping.dat
"""

import time
from mininet.log import setLogLevel, info
from mininet.net import Mininet
from mininet.node import Controller


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')
    s1 = net.addSwitch('s1')
    s2 = net.addSwitch('s2')
    info('*** Creating links\n')
    net.addLink(h1, s1)
    net.addLink(h2, s2)
    net.addLink(s2, s1)
    info('*** Starting network\n')
    net.start()

    info('*** Set rate\n')
    h1.cmdPrint('tc qdisc add dev h1-eth0 root tbf rate 10gbit burst 5000000 limit 15000000')
    info('*** Start iperf server on h2\n')
    h2.cmdPrint('iperf3 -s &')
    time.sleep(10)

    info('*** Run iperf3 client from h1 to h2')
    h1.cmdPrint('iperf3 -c', h2.IP(), '-J > iperf_result.json')
    h1.cmdPrint('iperf3 -c',  h2.IP(), '| grep "MBytes" | awk \'{print $7}\'  > iperf3.dat')

    info('*** Run ping from h1 to h2\n')
    h1.cmdPrint('ping -c 15',  h2.IP(), '| grep "time=" | awk \'{print $5, $7}\' | sed -e \'s/time=//g\'

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


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

Листинг Makefile и ping_plot

```
mininet@mininet-vm:~/work/lab_tbf_i/exp1$ cat Makefile
all: ping.dat ping.png

ping.dat:
	sudo python lab_tbf_i.py
	sudo chown mininet:mininet ping.dat

ping.png:
	./ping_plot

clean:
	-rm -f *.dat *.png
mininet@mininet-vm:~/work/lab_tbf_i/exp1$ cat ping_plot
#!/usr/bin/gnuplot --persist

set terminal png crop
set output 'ping.png'
set xlabel "Sequence number"
set ylabel "Delay (ms)"
set grid
plot "ping.dat" with lines

set terminal png crop
set output 'iperf3.png'
set xlabel "Packet number"
set ylabel "Rate (Gbytes/sec)"
set grid
plot "iperf3.dat" with lines
```

Рис. 14: Листинг Makefile и ping_plot

Запуск exp1

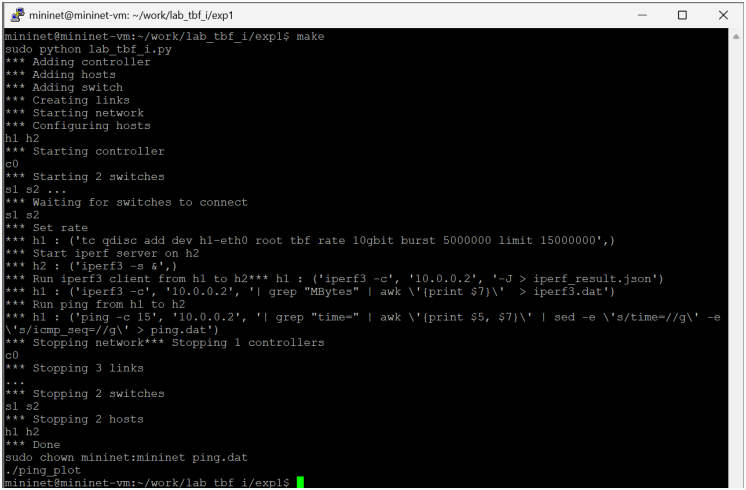


Рис. 15: Запуск exp1

График iperf3.png из exp1

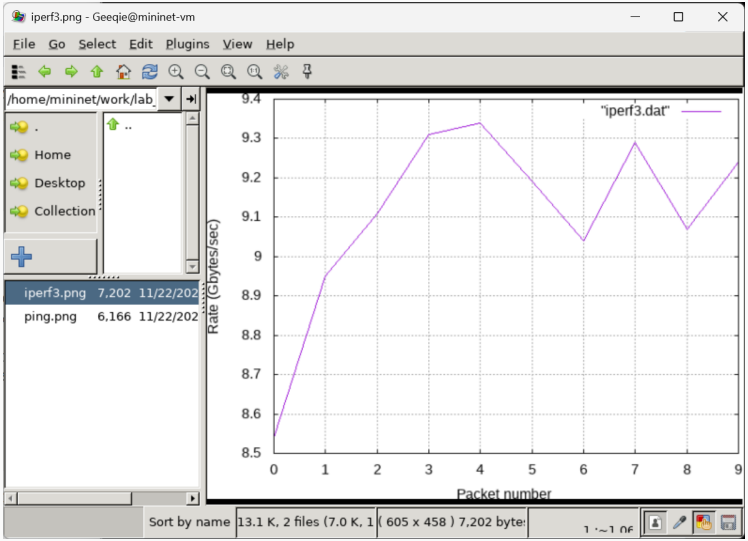


Рис. 16: График iperf3.png из exp1

График ping.png из exp1

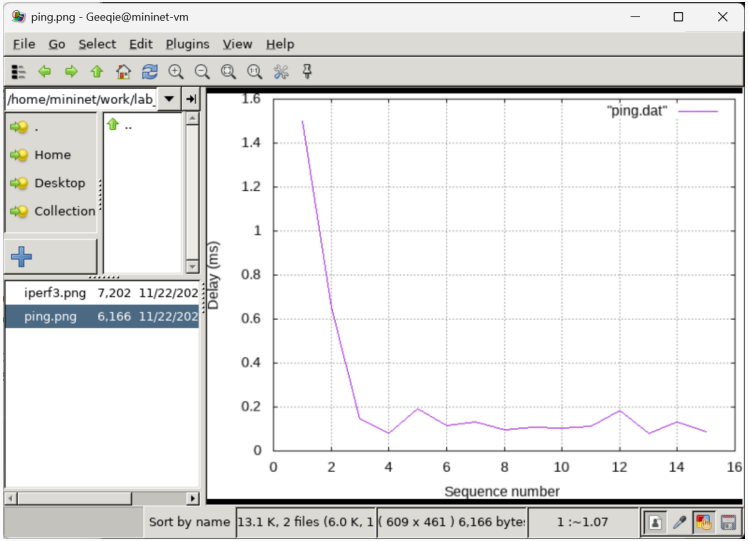
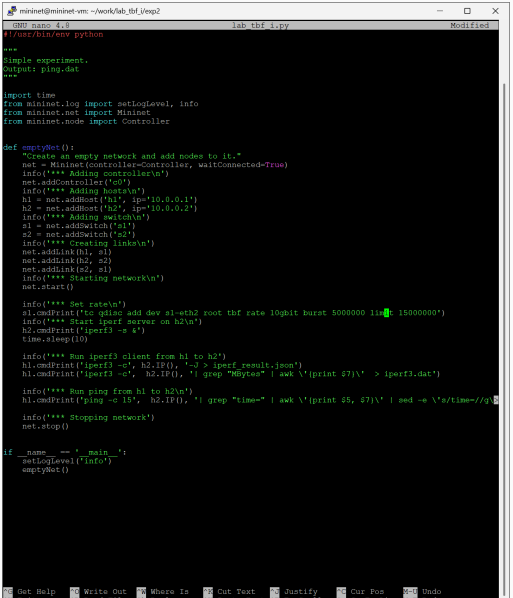


Рис. 17: График ping.png из exp1

Листинг exp2/lab_tbf_i.py



Запуск exp2

```
mininet@mininet-vm:~/work/lab_tbf_i/exp2$ make
sudo python lab_tbf_i.py
*** Adding controller
*** Adding hosts
*** Adding switch
*** Creating links
*** Starting network
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 2 switches
s1 s2 ...
*** Waiting for switches to connect
s1 s2
*** Set rate
*** s1 : ('tc qdisc add dev s1-eth2 root tbf rate 10gbit burst 5000000 limit 15000000',)
*** Start iperf server on h2
*** h2 : ('iperf3 -s &',)
*** Run iperf3 client from h1 to h2*** h1 : ('iperf3 -c', '10.0.0.2', '-J > iperf_result.json')
*** h1 : ('iperf3 -c', '10.0.0.2', '| grep "MBytes" | awk \'{print $7}\'  > iperf3.dat')
*** Run ping from h1 to h2
*** h1 : ('ping -c 15', '10.0.0.2', '| grep "time=" | awk \'{print $5, $7}\' | sed -e \'s/time=//g\' -e
\'s/icmp_seq=//g\' > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 3 links
...
*** Stopping 2 switches
s1 s2
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot
mininet@mininet-vm:~/work/lab_tbf_i/exp2$ ls
iperf3.dat  iperf3.png  iperf_result.json  lab_tbf_i.py  Makefile  ping.dat  ping_plot  ping.png
mininet@mininet-vm:~/work/lab_tbf_i/exp2$
```

Рис. 19: Запуск exp2

График iperf3.png из exp2

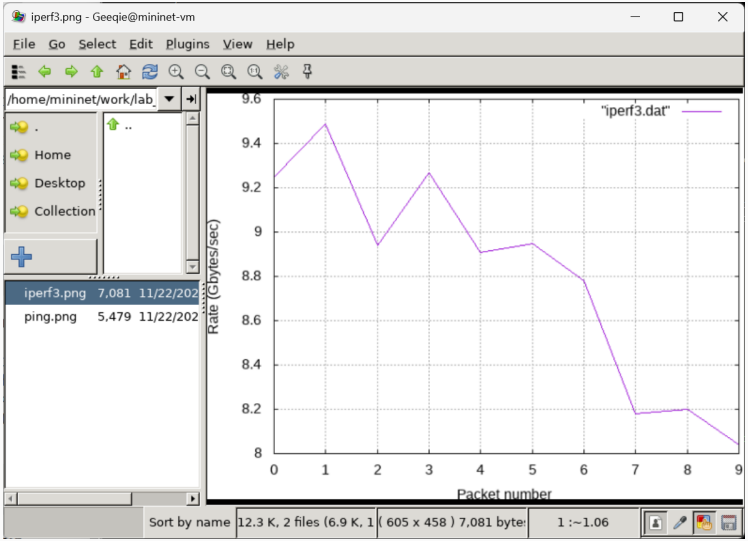


Рис. 20: График iperf3.png из exp2

График ping.png из exp2

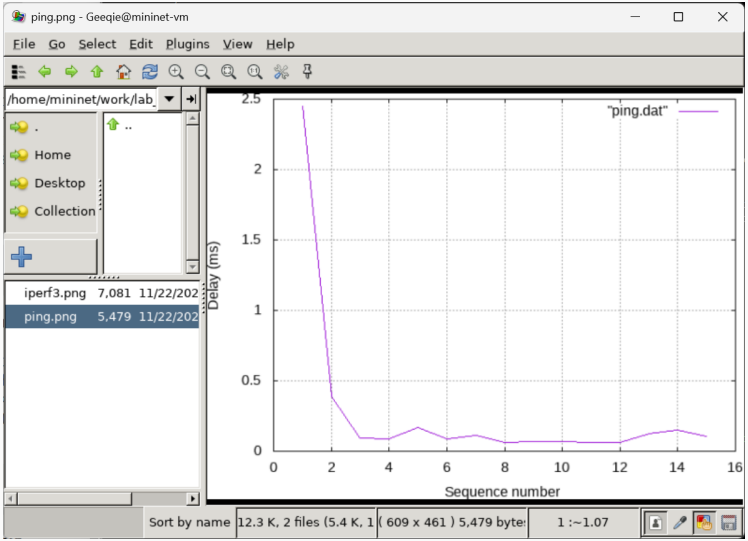


Рис. 21: График ping.png из exp2

Листинг exp3/lab_tbf_i.py

mininet@mininet-vm: ~/work/lab_tbf_i/exp3

GNU nano 4.8 lab_tbf_1.py Modified

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

"""
Simple experiment.
Output: ping.dat
"""

import time
from mininet.log import setLogLevel, info
from mininet.net import Mininet
from mininet.node import Controller


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')
    s1 = net.addSwitch('s1')
    s2 = net.addSwitch('s2')
    info('*** Creating links\n')
    net.addLink(h1, s1)
    net.addLink(h2, s2)
    net.addLink(s2, s1)
    info('*** Starting network\n')
    net.start()

    info('*** Set rate\n')
    s1.cmdPrint('tc qdisc add dev s1-eth2 root handle 1: netem delay 10ms')
    s1.cmdPrint('tc qdisc add dev s1-eth2 root parent 1: handle 2: tbf rate 2gbit burst 1000000 limit 2000000')
    info('*** Start iperf server on h2\n')
    h2.cmdPrint('iperf3 -s &')
    time.sleep(10)

    info('*** Run iperf3 client from h1 to h2')
    h1.cmdPrint('iperf3 -c', h2.IP(), '-J > iperf_result.json')
    h1.cmdPrint('iperf3 -c',  h2.IP(), '| grep "MBytes" | awk \'{print $7}\'  > iperf3.dat')

    info('*** Run ping from h1 to h2\n')
    h1.cmdPrint('ping -c 15',  h2.IP(), '| grep "time=" | awk \'{print $5, $7}\' | sed -e \'s/time=//g\' -e \'s/icmp_s

    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 M-U Undo

Запуск exp3

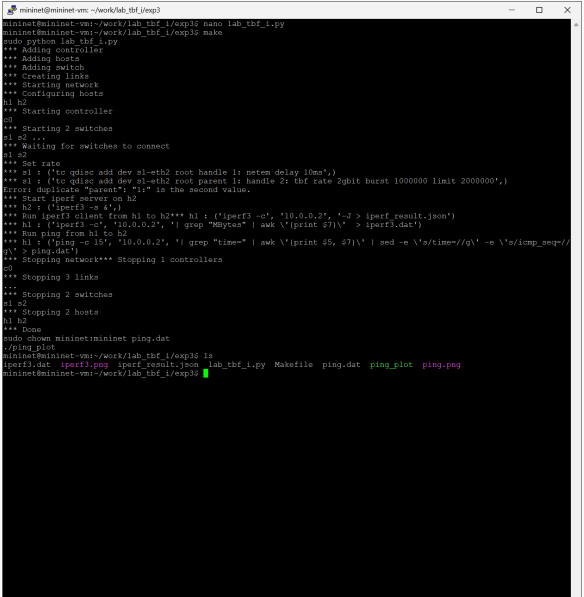

mininet@mininet-vm: ~/work/lab_tbf_i/exp3
mininet@mininet-vm:~/work/lab_tbf_i/exp3$ nano lab_tbf_i.py
mininet@mininet-vm:~/work/lab_tbf_i/exp3$ make
sudo python lab_tbf_i.py
*** Adding controller
*** Adding hosts
*** Adding switch
*** Creating links
*** Starting network
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 2 switches
s1 s2 ...
*** Waiting for switches to connect
s1 s2
*** Set rate
*** s1 : ('tc qdisc add dev s1-eth2 root handle 1: netem delay 10ms',)
*** s1 : ('tc qdisc add dev s1-eth2 parent 1: handle 2: tbf rate 2gbit burst 1000000 limit 2000000',)
Error: duplicate "parent": "1:" is the second value.
*** Start iperf server on h2
*** h2 : ('iperf3 -s &',)
*** Run iperf3 client from h1 to h2*** h1 : ('iperf3 -c', '10.0.0.2', '-J > iperf_result.json')
*** h1 : ('iperf3 -c', '10.0.0.2', '| grep "MBytes" | awk \'{print $7}\' > iperf3.dat')
*** Run ping from h1 to h2
*** h1 : ('ping -c 15', '10.0.0.2', '| grep "time=" | awk \'{print $5, $7}\' | sed -e \'s/time=//g\' -e \'s/icmp_seq=//g\' > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 3 links
...
*** Stopping 2 switches
s1 s2
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot
mininet@mininet-vm:~/work/lab_tbf_i/exp3$ ls
iperf3.dat iperf3.png iperf_result.json lab_tbf_i.py Makefile ping.dat ping_plot ping.png
mininet@mininet-vm:~/work/lab_tbf_i/exp3$

График iperf3.png из exp3

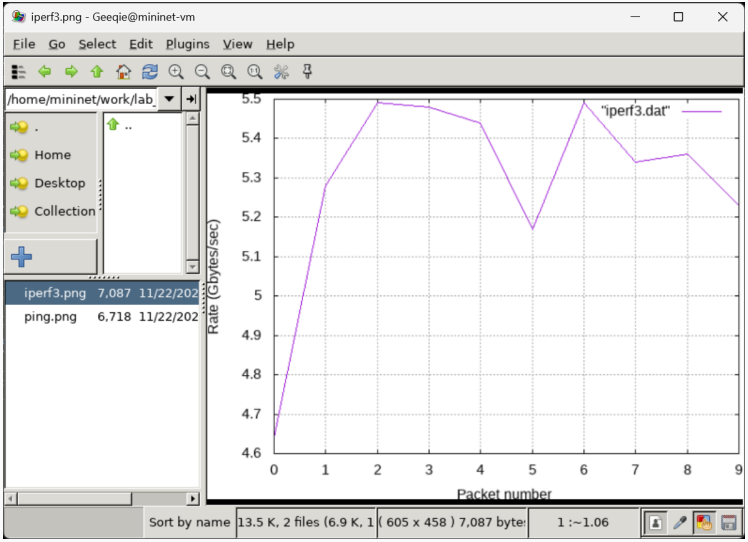


Рис. 24: График iperf3.png из exp3

График ping.png из exp3

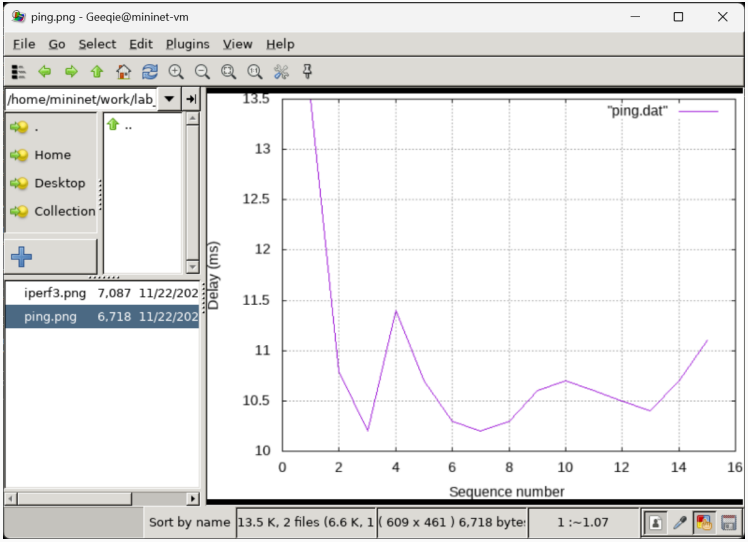


Рис. 25: График ping.png из exp3

Выводы

Выводы

В результате выполнения работы я познакомился с принципами работы Token Bucket Filter, а также получили навыки моделирования и исследования поведения трафика посредством проведения интерактивного и воспроизводимого экспериментов в Mininet.

Список литературы

Список литературы

1. Mininet [Электронный ресурс]. Mininet Project Contributors. URL: http://mini net.org/ (дата обращения: 06.10.2025).