

# Отчёт по лабораторной работе №6

Настройка пропускной способности глобальной сети с помощью Token  
Bucket Filter

Студент: Кузнецова София Вадимовна

# Содержание

|                                |    |
|--------------------------------|----|
| Цель работы                    | 5  |
| Теоретическое введение         | 6  |
| Выполнение лабораторной работы | 7  |
| Выводы                         | 14 |

# Список иллюстраций

|     |  |    |
|-----|--|----|
| 0.1 | Задание топологии . . . . .                        | 7  |
| 0.2 | ifconfig на хостах . . . . .                       | 8  |
| 0.3 | Запуск iperf3 на хостах . . . . .                  | 9  |
| 0.4 | Ограничение скорости на конечных хостах . . . . .  | 10 |
| 0.5 | Ограничение скорости на коммутаторах . . . . .     | 10 |
| 0.6 | Объединение NETEM и TBF . . . . .                  | 11 |
| 0.7 | Скрипт для воспроизводимого эксперимента . . . . . | 12 |
| 0.8 | Скрипт для отрисовки графика . . . . .             | 12 |
| 0.9 | График изменения скорости передачи . . . . .       | 13 |

## Список таблиц

# 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.

# Теоретическое введение

Mininet[@mininet] – это эмулятор компьютерной сети. Под компьютерной сетью подразумеваются простые компьютеры — хосты, коммутаторы, а так же OpenFlow-контроллеры. С помощью простейшего синтаксиса в примитивном интерпретаторе команд можно разворачивать сети из произвольного количества хостов, коммутаторов в различных топологиях и все это в рамках одной виртуальной машины(ВМ). На всех хостах можно изменять сетевую конфигурацию, пользоваться стандартными утилитами(ifconfig, ping) и даже получать доступ к терминалу. На коммутаторы можно добавлять различные правила и маршрутизировать трафик.

# Выполнение лабораторной работы

Зададим простейшую топологию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8. На хостах h1 и h2 введем команду ifconfig, чтобы отобразить информацию, относящуюся к их сетевым интерфейсам и назначенным им IP-адресам. Проверим подключение между хостами сети.

The screenshot shows four terminal windows from a Linux host running mininet. The windows are:

- "host: h1" (root@mininet-vm): Shows interface h1-eth0 with IP 10.0.0.1 and interface lo with IP 127.0.0.1.
- "host: h2" (root@mininet-vm): Shows interface h2-eth0 with IP 10.0.0.2 and interface lo with IP 127.0.0.1.
- "switch: s1" (root@mininet-vm): Shows interface s1-eth2 with IP 192.168.11.25 and interface lo with IP 127.0.0.1.
- "switch: s2" (root@mininet-vm): Shows interface s2-eth1 with IP 192.168.11.128 and interface lo with IP 127.0.0.1.

The ifconfig output for each interface includes statistics like RX/TX bytes, errors, dropped, overruns, frame, and carrier counts, along with MTU and queueing discipline information.

Рис. 0.1: Задание топологии

На хостах h1, h2 и на коммутаторах s1, s2 введем команду ifconfig, чтобы отобразить информацию, относящуюся к их сетевым интерфейсам и назначенным им IP-адресам. В дальнейшем при работе с NETEM и командой tc будут использоваться интерфейсы h1-eth0, h2-eth0, s1-eth2.

```
X "host: h1"@mininet-vm
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# ping -c 4
ping: usage error: Destination address required
root@mininet-vm:/home/mininet# ping -c 4 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=15.6 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.707 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.118 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.096 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3055ms
rtt min/avg/max/mdev = 0.096/4.139/15.637/6.642 ms
root@mininet-vm:/home/mininet# 

X "host: h2"@mininet-vm
RX packets 0 bytes 0 (0.0 B)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
loop txqueuelen 1000 (Local Loopback)
RX packets 995 bytes 114704 (114.7 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 995 bytes 114704 (114.7 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# ping -c 4 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=23.9 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.099 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.255 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.114 ms

--- 10.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3038ms
rtt min/avg/max/mdev = 0.099/6.099/23.928/10.293 ms
root@mininet-vm:/home/mininet#
```

Рис. 0.2: ifconfig на хостах

Запустим iPerf3 на хостах и посмотрим результат отработки на данном этапе.

```

X "host: h1"@mininet-vm
rtt min/avg/max/mdev = 0.096/4.139/15.637/6.642 ms
root@mininet-vm:/home/mininet# iperf3 -c 10.0.0.2
Connecting to host 10.0.0.2, port 5201
[ 7] local 10.0.0.1 port 32826 connected to 10.0.0.2 port 5201
[ ID] Interval Transfer Bitrate Retr Cwnd
[ 7] 0.00-1.00 sec 1.29 GBytes 11.0 Gbits/sec 0 379 KBytes
[ 7] 1.00-2.00 sec 1.11 GBytes 9.55 Gbits/sec 0 379 KBytes
[ 7] 2.00-3.01 sec 966 MBytes 8.05 Gbits/sec 0 379 KBytes
[ 7] 3.01-4.00 sec 1.12 GBytes 9.73 Gbits/sec 0 560 KBytes
[ 7] 4.00-5.00 sec 998 MBytes 8.38 Gbits/sec 0 560 KBytes
[ 7] 5.00-6.00 sec 784 MBytes 6.59 Gbits/sec 0 3.37 MBytes
[ 7] 6.00-7.00 sec 878 MBytes 7.36 Gbits/sec 0 6.66 MBytes
[ 7] 7.00-8.00 sec 1.24 GBytes 10.7 Gbits/sec 0 8.10 MBytes
[ 7] 8.00-9.00 sec 1.20 GBytes 10.3 Gbits/sec 0 8.10 MBytes
[ 7] 9.00-10.00 sec 1.11 GBytes 9.53 Gbits/sec 0 8.10 MBytes
-----
[ ID] Interval Transfer Bitrate Retr
[ 7] 0.00-10.00 sec 10.6 GBytes 9.12 Gbits/sec 0
[ 7] 0.00-10.00 sec 10.6 GBytes 9.12 Gbits/sec

iperf Done.
root@mininet-vm:/home/mininet# 

X "host: h2"@mininet-vm
Server listening on 5201
-----
Accepted connection from 10.0.0.1, port 32824
[ 7] local 10.0.0.2 port 5201 connected to 10.0.0.1 port 32826
[ ID] Interval Transfer Bitrate
[ 7] 0.00-1.00 sec 1.28 GBytes 11.0 Gbits/sec
[ 7] 1.00-2.00 sec 1.11 GBytes 9.55 Gbits/sec
[ 7] 2.00-3.00 sec 966 MBytes 8.11 Gbits/sec
[ 7] 3.00-4.00 sec 1.12 GBytes 9.66 Gbits/sec
[ 7] 4.00-5.00 sec 997 MBytes 8.37 Gbits/sec
[ 7] 5.00-6.00 sec 785 MBytes 6.58 Gbits/sec
[ 7] 6.00-7.00 sec 877 MBytes 7.36 Gbits/sec
[ 7] 7.00-8.00 sec 1.24 GBytes 10.6 Gbits/sec
[ 7] 8.00-9.00 sec 1.20 GBytes 10.3 Gbits/sec
[ 7] 9.00-10.00 sec 1.11 GBytes 9.53 Gbits/sec
[ 7] 10.00-10.00 sec 704 KBytes 5.35 Gbits/sec
-----
[ ID] Interval Transfer Bitrate
[ 7] 0.00-10.00 sec 10.6 GBytes 9.12 Gbits/sec

Server listening on 5201
-----
^Ciperf3: interrupt - the server has terminated

```

Рис. 0.3: Запуск iperf3 на хостах

Изменим пропускную способность хоста h1, установив пропускную способность на 10 Гбит/с на интерфейсе h1-eth0 и параметры TBF-фильтра.

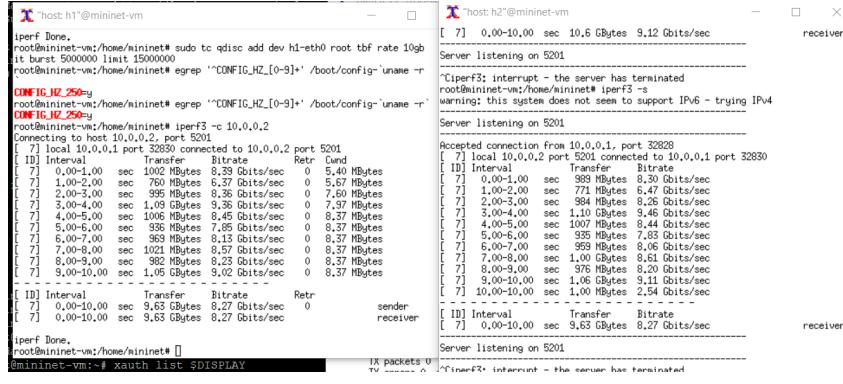


Рис. 0.4: Ограничение скорости на конечных хостах

Применим правило ограничения скорости tbf с параметрами rate = 10gbit, burst = 5,000,000, limit = 15,000,000 к интерфейсу s1-eth2 коммутатора s1, который соединяет его с коммутатором s2.

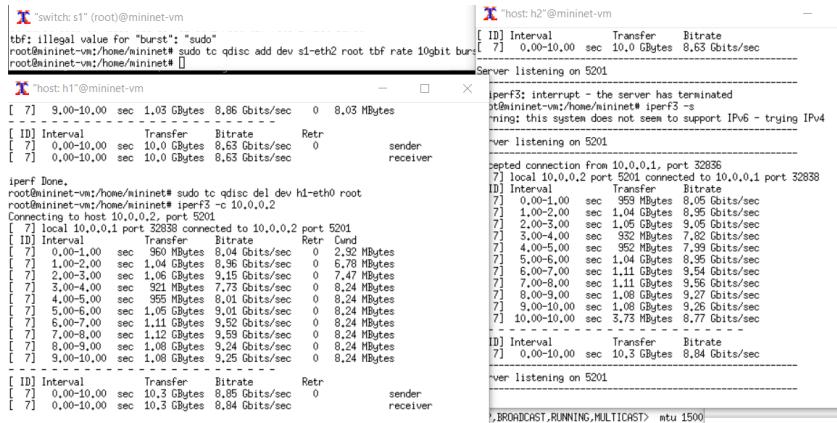


Рис. 0.5: Ограничение скорости на коммутаторах

Объединим NETEM и TBF, введя на интерфейсе s1-eth2 коммутатора s1 задержку, джиттер, повреждение пакетов и указав скорость. Добавим второе правило на коммутаторе s1, которое задаёт ограничение скорости с помощью tbf с параметрами rate=2gbit, burst=1,000,000, limit=2,000,000: и проверим.

```

"switch: s1" (root)@mininet-vm
root@mininet-vm:~/home/mininet# sudo tc qdisc add dev s1-eth2 root tbf rate 10gbit burst 5000000 limit 15000000
root@mininet-vm:~/home/mininet# sudo tc qdisc del dev s1-eth2 root
root@mininet-vm:~/home/mininet# sudo tc qdisc add dev s1-eth2 root handle 1: netem delay 10ms
root@mininet-vm:~/home/mininet# sudo tc qdisc add dev s1-eth2 parent 1: handle 2: tbf rate 2gbit burst 1000000 limit 2000000
root@mininet-vm:~/home/mininet# 

"host: h1" @mininet-vm
4 bytes From 10.0.0.2: icmp_seq=1 ttl=64 time=14.5 ms
4 bytes From 10.0.0.2: icmp_seq=2 ttl=64 time=12.0 ms
4 bytes From 10.0.0.2: icmp_seq=3 ttl=64 time=11.1 ms
4 bytes From 10.0.0.2: icmp_seq=4 ttl=64 time=10.5 ms
-- 10.0.0.2 ping statistics --
packets transmitted: 4 received, 0% packet loss, time 3005ms
tt min/avg/max = 10.508/12.007/14.463/1.521 ms
root@mininet-vm:~/home/mininet# iperf3 -c 10.0.0.2
connecting to host 10.0.0.2, port 5201
[ 7] local 10.0.0.1 port 32842 connected to 10.0.0.2 port 5201
[ ID] Interval Transfer Bitrate Retr. lost/seq
[ 7] 0.00-1.00 sec 223 MBytes 1.77 Gbits/sec 223 MBytes 0 3.26 MBytes
[ 7] 1.00-2.00 sec 223 MBytes 1.92 Gbits/sec 0 3.50 MBytes
[ 7] 2.00-3.00 sec 223 MBytes 1.91 Gbits/sec 0 3.70 MBytes
[ 7] 3.00-4.00 sec 223 MBytes 1.91 Gbits/sec 90 2.73 MBytes
[ 7] 4.00-5.00 sec 223 MBytes 1.92 Gbits/sec 0 2.07 MBytes
[ 7] 5.00-6.00 sec 223 MBytes 1.92 Gbits/sec 0 2.08 MBytes
[ 7] 6.00-7.00 sec 223 MBytes 1.92 Gbits/sec 0 3.06 MBytes
[ 7] 7.00-8.00 sec 206 MBytes 1.74 Gbits/sec 405 2.25 MBytes
[ 7] 8.00-9.00 sec 215 MBytes 1.80 Gbits/sec 0 2.36 MBytes
[ 7] 9.00-10.00 sec 223 MBytes 1.89 Gbits/sec 0 2.45 MBytes
[ ID] Interval Transfer Bitrate Retr. lost/seq
[ 7] 0.00-10.00 sec 2.18 GBytes 1.87 Gbits/sec 720 sender
[ 7] 0.00-10.02 sec 2.17 GBytes 1.86 Gbits/sec receiver
perf Done.
root@mininet-vm:~/home/mininet# 

"host: h2" @mininet-vm
0.00-10.00 sec 10.3 GBytes 8.84 Gbits/sec
Server listening on 5201
-----[ ID] Interval Transfer Bitrate Retr. lost/seq
[ 7] 0.00-1.00 sec 201 MBytes 1.68 Gbits/sec
[ 7] 1.00-2.00 sec 223 MBytes 1.91 Gbits/sec
[ 7] 2.00-3.00 sec 223 MBytes 1.91 Gbits/sec
[ 7] 3.00-4.00 sec 223 MBytes 1.91 Gbits/sec
[ 7] 4.00-5.00 sec 223 MBytes 1.91 Gbits/sec
[ 7] 5.00-6.00 sec 223 MBytes 1.91 Gbits/sec
[ 7] 6.00-7.00 sec 223 MBytes 1.91 Gbits/sec
[ 7] 7.00-8.00 sec 206 MBytes 1.75 Gbits/sec
[ 7] 8.00-9.00 sec 215 MBytes 1.80 Gbits/sec
[ 7] 9.00-10.00 sec 223 MBytes 1.89 Gbits/sec
[ 7] 10.00-10.02 sec 2.00 MBytes 1.06 Gbits/sec
[ ID] Interval Transfer Bitrate Retr. lost/seq
[ 7] 0.00-10.02 sec 2.17 GBytes 1.86 Gbits/sec
[ 7] 0.00-10.02 sec 2.17 GBytes 1.86 Gbits/sec
Server listening on 5201
-----[ ID] Interval Transfer Bitrate Retr. lost/seq
[ 7] 0.00-10.02 sec 2.17 GBytes 1.86 Gbits/sec
[ 7] 0.00-10.02 sec 2.17 GBytes 1.86 Gbits/sec
errroot@mininet-vm:~/home/mininet# 

```

Рис. 0.6: Объединение NETEM и TBF

В виртуальной среде mininet в своём рабочем каталоге с проектами создадим каталог simple-tbf и перейдем в него. Создадим скрипт для эксперимента lab\_netem\_iii.py.

```

GNU nano 4.8                               lab netem ii.py
Output: ping.dat
"""

from mininet.net import Mininet
from mininet.node import Controller
from mininet.cli import CLI

from mininet.log import setLogLevel, info
import time

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' )

    info( '*** Creating links\n' )
    net.addLink( h1, s1 )
    net.addLink( h2, 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')
    time.sleep(10) # Wait 10 seconds

    info('*** Starting iperf server on h2\n')
    h2.cmdPrint('iperf3 -s &') # Launch server in foreground mode
    info('*** Running iperf client from h1 to h2\n')
    h1.cmdPrint('iperf3 -c ' + h2.IP() + ' | grep "MBytes" | awk \'(print $7)\' > ping.dat')

    info( '*** Stopping network' )

```

Рис. 0.7: Скрипт для воспроизводимого эксперимента

Создадим также скрипт для визуализации ping\_plot результатов эксперимента.

```

GNU nano 4.8                               ping_plot
#!/usr/bin/gnuplot --persist
set terminal png crop
set output 'ping.png'
set xlabel "Packet number"
set ylabel "rate (Gbytes/sec)"
set grid
plot "ping.dat" with lines

```

Рис. 0.8: Скрипт для отрисовки графика

Получим следующий график.

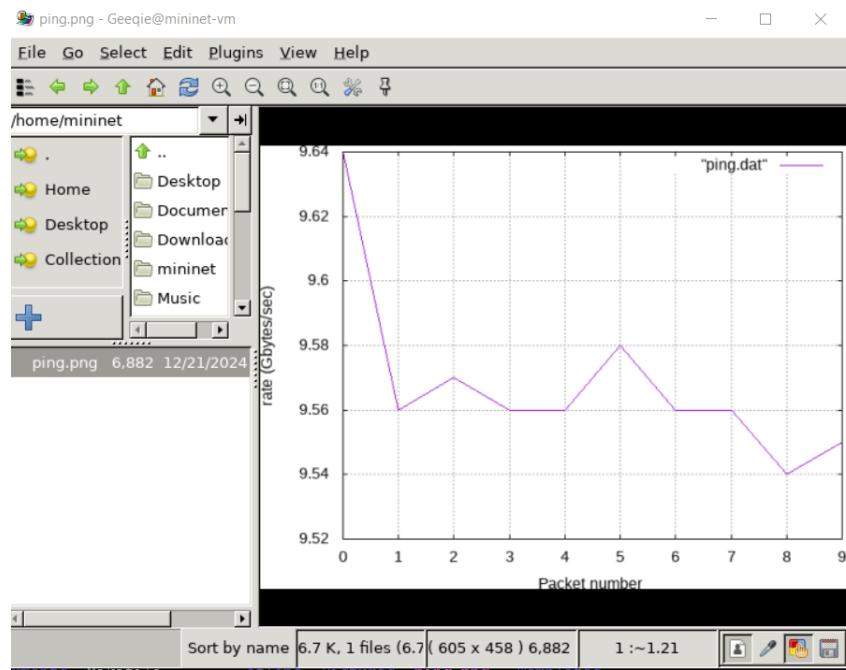


Рис. 0.9: График изменения скорости передачи

## Выводы

В результате выполнения данной лабораторной работы я познакомилась с принципами работы дисциплины очереди Token Bucket Filter, которая формирует входящий/исходящий трафик для ограничения пропускной способности, а также получила навыки моделирования и исследования поведения трафика посредством проведения интерактивного и воспроизводимого экспериментов в Mininet.