

# **Лабораторная работа №6**

**Настройка пропускной способности глобальной сети с помощью  
Token Bucket Filter**

Ланцова Яна Игоревна

# **Содержание**

<b>1 Цель работы</b>	<b>5</b>
<b>2 Задачи</b>	<b>6</b>
<b>3 Выполнение лабораторной работы</b>	<b>7</b>
3.1 Запуск лабораторной топологии . . . . .	7
3.2 Интерактивные эксперименты . . . . .	10
3.3 Воспроизводимые эксперименты . . . . .	13
<b>4 Выводы</b>	<b>16</b>

# Список иллюстраций

3.1 Исправление MIT magic cookie . . . . .	7
3.2 Простейшая топология . . . . .	8
3.3 ifconfig на коммутаторах s1 и s2 . . . . .	8
3.4 ifconfig на хостах h1 и h2 . . . . .	9
3.5 Проверка соединения между хостами . . . . .	9
3.6 Запуск iperf3 на хостах . . . . .	10
3.7 Ограничение скорости на конечных хостах . . . . .	10
3.8 Проверка пропускной способности . . . . .	11
3.9 Ограничение скорости на коммутаторах . . . . .	12
3.10 Объединение NETEM и TBF . . . . .	12
3.11 Объединение NETEM и TBF . . . . .	13
3.12 Создание рабочего каталога . . . . .	13
3.13 Скрипт на Python для эксперимента . . . . .	14
3.14 Makefile для управления процессом проведения эксперимента . . . . .	14
3.15 Создание файла для изображения графика . . . . .	14
3.16 Запуск эксперимента . . . . .	15
3.17 График изменения скорости передачи . . . . .	15

# **Список таблиц**

# **1 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.

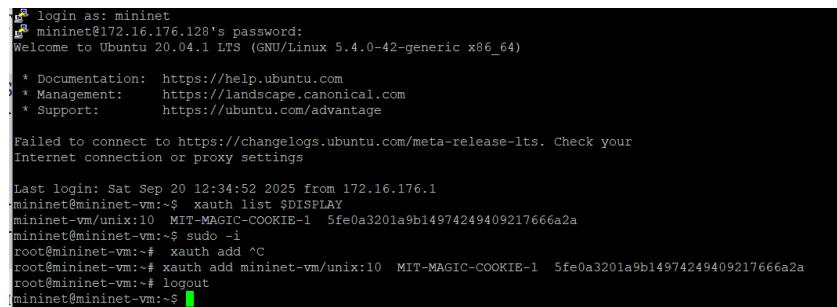
## **2 Задачи**

1. Задайте топологию, состоящую из двух хостов и двух коммутаторов с назначенной по умолчанию mininet сетью 10.0.0.0/8.
2. Проведите интерактивные эксперименты по ограничению пропускной способности сети с помощью TBF в эмулируемой глобальной сети.
3. Самостоятельно реализуйте воспроизводимые эксперимент по применению TBF для ограничения пропускной способности. Постройте соответствующие графики.

# 3 Выполнение лабораторной работы

## 3.1 Запуск лабораторной топологии

Из основной ОС подключимся к виртуальной машине и исправим права запуска X-соединения. Скопируем значение куки (MIT magic cookie) своего пользователя mininet в файл для пользователя root (рис. 3.1).



```
login as: mininet
mininet@172.16.176.128's password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-42-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your
Internet connection or proxy settings

Last login: Sat Sep 20 12:34:52 2025 from 172.16.176.1
mininet@mininet-vm:~$ xauth list $DISPLAY
mininet-vm@mininet-vm:~$ sudo -i
root@mininet-vm:~# xauth add ^C
root@mininet-vm:~# xauth add mininet-vm@172.16.176.128:0 MIT-MAGIC-COOKIE-1 5fe0a3201a9b14974249409217666a2a
root@mininet-vm:~# logout
mininet@mininet-vm:~$
```

Рис. 3.1: Исправление MIT magic cookie

Зададим простейшую топологию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8 (рис. 3.2). После введения этой команды запустятся терминалы двух хостов, коммутатора и контроллера. Терминалы коммутатора и контроллера можно закрыть.

```
mininet@mininet-vm:~$ sudo mn --topo=linear,2 -x
*** Creating network
*** Adding controller
*** Adding switch
h1 h2
*** Addi
s1 s2
*** Addi
h1_1 s1
*** Addi
h1_2 s2
h1 h2
*** Runn
*** Star
c0
c1
*** Star
s1 s2
*** S
mininet@mininet-vm:~$ "switch: s1" (root)@mininet-vm
[mininet-vm] "host: h1" @mininet-vm
root@mininet-vm:~$ "switch: s2" (root)@mininet-vm
[mininet-vm] "host: h2" @mininet-vm
[mininet-vm] "controller: c0" (root)@mininet-vm
[mininet-vm] root@mininet-vm:~$
```

Рис. 3.2: Простейшая топология

На комммутаторах s1 и 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 s1-eth2 (рис. 3.3).

```
root@mininet-vm:~/home/mininet# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 172.16.176.200 brd netmask 255.255.255.0 broadcast 172.16.176.255
              ether 00:0c:29:dc:6d:c5 txqueuelen 1000 (Ethernet)
              RX packets 0 bytes 0 (0.0 B)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 81908 bytes 44706646 (44.7 MB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 172.16.18.128 brd netmask 255.255.255.0 broadcast 172.16.18.255
              ether 00:0c:29:dc:6d:c5 txqueuelen 1000 (Ethernet)
              RX packets 16772 bytes 246881250 (246.8 MB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 82018 bytes 44740964 (44.7 MB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 5536
      inet 127.0.0.1 brd 127.0.0.1 netmask 255.255.255.0
          loop txqueuelen 1000 (Local Loopback)
          RX packets 71639 bytes 42639575 (42.6 MB)
          RX errors 0 dropped 0 overruns 0 frame 0
          TX packets 71639 bytes 42639575 (42.6 MB)
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

s1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        ether ca:73:7b:7d:1a:9e txqueuelen 1000 (Ethernet)
        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

s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        ether b6:07:al:b1:67:d3 txqueuelen 1000 (Ethernet)
        RX packets 0 bytes 0 (0.0 B)
        RX errors 0 dropped 0 overruns 0 frame 0

root@mininet-vm:~/home/mininet# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 172.16.176.128 brd netmask 255.255.255.0 broadcast 172.16.176.255
              ether 00:0c:29:dc:6c:eb txqueuelen 1000 (Ethernet)
              RX packets 0 bytes 0 (0.0 B)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 82018 bytes 44740964 (44.7 MB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 172.16.18.128 brd netmask 255.255.255.0 broadcast 172.16.18.255
              ether 00:0c:29:dc:6d:c5 txqueuelen 1000 (Ethernet)
              RX packets 167730 bytes 246881350 (246.8 MB)
              RX errors 0 dropped 0 overruns 0 frame 0
              TX packets 82040 bytes 4086933 (4.0 MB)
              TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 5536
      inet 127.0.0.1 brd 127.0.0.1 netmask 255.255.255.0
          loop txqueuelen 1000 (Local Loopback)
          RX packets 71952 bytes 42689971 (42.6 MB)
          RX errors 0 dropped 0 overruns 0 frame 0
          TX packets 71952 bytes 42689971 (42.6 MB)
          TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

s1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        ether ca:73:7b:7d:1a:9e txqueuelen 1000 (Ethernet)
        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

s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        ether b6:07:al:b1:67:d3 txqueuelen 1000 (Ethernet)
        RX packets 0 bytes 0 (0.0 B)
        RX errors 0 dropped 0 overruns 0 frame 0
```

Рис. 3.3: ifconfig на коммутаторах s1 и s2

На хостах h1 и h2 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 и h2-eth0 (рис. 3.4).

```

root@mininet-vm:/home/mininet# ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.1 netmask 255.0.0.0 broadcast 10.255.255.255
        ether 36:cc:4d:b5:ce:ad txqueuelen 1000 (Ethernet)
        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 0 bytes 0 (0.0 B) txqueuelen 0 (loopback)
        RX packets 1222 bytes 291544 (291.5 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 1222 bytes 291544 (291.5 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# 

root@mininet-vm:/home/mininet# ifconfig
h2-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.2 netmask 255.0.0.0 broadcast 10.255.255.255
        ether 2a:58:35:b2:28:62 txqueuelen 1000 (Ethernet)
        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 0 bytes 0 (0.0 B) txqueuelen 0 (loopback)
        RX packets 1164 bytes 290696 (290.6 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 1164 bytes 290696 (290.6 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet#

```

Рис. 3.4: ifconfig на хостах h1 и h2

Проверим подключение между хостами h1 и h2 с помощью команды ping с параметром -c 4 (рис. 3.5).

```

root@mininet-vm:/home/mininet# ping 10.0.0.2 -c 4
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=26.2 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=1.08 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.204 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.245 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3022ms
rtt min/avg/max/mdev = 0.204/6.927/26.183/11.122 ms
root@mininet-vm:/home/mininet#

```

Рис. 3.5: Проверка соединения между хостами

Запустим iPerf3 на хостах и посмотрим результат отработки на данном этапе (рис. 3.6).

```

root@mininet-vm:/home/mininet# iperf3 -s
warning: this system does not seem to support IPv6 - trying IPv4
-----
Server listening on 5201
-----
Accepted connection from 10.0.0.1, port 35052
[ 7] local 10.0.0.2 port 5201 connected to 10.0.0.1 port 35054
[ ID] Interval Transfer Bitrate
[ 7] 0.00-1.00 sec 275 MBytes 2.30 Gbytes/sec
[ 7] 1.00-2.00 sec 234 MBytes 1.97 Gbytes/sec
[ 7] 2.00-3.00 sec 250 MBytes 2.10 Gbytes/sec
[ 7] 3.00-4.00 sec 224 MBytes 1.88 Gbytes/sec
[ 7] 4.00-5.00 sec 177 MBytes 1.49 Gbytes/sec
-----
X "host h1" @mininet-vm
- □ ×
--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3022ms
rtt min/avg/max/mdev = 0.204/6.927/26.183/11.122 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 35054 connected to 10.0.0.2 port 5201
[ ID] Interval Transfer Bitrate Retr Cwnd
[ 7] 0.00-1.00 sec 278 MBytes 2.32 Gbytes/sec 0 4.74 MBytes
[ 7] 1.00-2.01 sec 234 MBytes 1.94 Gbytes/sec 0 4.98 MBytes
[ 7] 2.01-3.00 sec 251 MBytes 2.13 Gbytes/sec 0 4.98 MBytes
[ 7] 3.00-4.01 sec 224 MBytes 1.87 Gbytes/sec 0 4.98 MBytes
[ 7] 4.01-5.01 sec 176 MBytes 1.47 Gbytes/sec 0 4.98 MBytes
[ 7] 5.01-6.01 sec 198 MBytes 1.65 Gbytes/sec 0 4.98 MBytes
[ 7] 6.01-7.01 sec 138 MBytes 1.16 Gbytes/sec 0 4.98 MBytes
[ 7] 7.01-8.01 sec 231 MBytes 1.94 Gbytes/sec 0 4.98 MBytes
[ 7] 8.01-9.01 sec 275 MBytes 2.30 Gbytes/sec 0 4.98 MBytes
[ 7] 9.01-10.01 sec 265 MBytes 2.23 Gbytes/sec 0 4.98 MBytes
-----
[ ID] Interval Transfer Bitrate Retr
[ 7] 0.00-10.01 sec 2.22 GBytes 1.90 Gbytes/sec 0
[ 7] 0.00-10.02 sec 2.22 GBytes 1.90 Gbytes/sec 0
sender receiver
iperf Done.

```

Рис. 3.6: Запуск iperf3 на хостах

## 3.2 Интерактивные эксперименты

Изменим пропускную способность хоста  $h1$ , установив пропускную способность на 10 Гбит/с на интерфейсе  $h1\text{-}eth0$  и параметры TBF-фильтра (рис. 3.7).

```

root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root tbf rate 10gb
it burst 5000000 limit 15000000
Error: Exclusivity flag on, cannot modify.
root@mininet-vm:/home/mininet# egrep '^CONFIG_HZ_[0-9]+' /boot/config-`uname -r`
CONFIG_HZ_250=y
root@mininet-vm:/home/mininet#

```

Рис. 3.7: Ограничение скорости на конечных хостах

Убедимся, что пропускная способность (Bitrate) увеличится (рис. 3.8)

```

X "host: h2"@"mininet-vm
Server listening on 5201
-----
Accepted connection from 10.0.0.1, port 35060
[ 7] local 10.0.0.2 port 5201 connected to 10.0.0.1 port 35062
[ ID] Interval Transfer Bitrate
[ 7] 0.00-1.00 sec 459 MBytes 3.85 Gbits/sec
[ 7] 1.00-2.00 sec 474 MBytes 3.97 Gbits/sec
[ 7] 2.00-3.00 sec 482 MBytes 4.04 Gbits/sec
[ 7] 3.00-4.00 sec 476 MBytes 3.99 Gbits/sec
[ 7] 4.00-5.00 sec 466 MBytes 3.91 Gbits/sec
[ 7] 5.00-6.00 sec 466 MBytes 3.91 Gbits/sec
[ 7] 6.00-7.00 sec 478 MBytes 4.01 Gbits/sec
[ 7] 7.00-8.00 sec 477 MBytes 4.00 Gbits/sec
[ 7] 8.00-9.00 sec 481 MBytes 4.04 Gbits/sec
[ 7] 9.00-10.00 sec 481 MBytes 4.03 Gbits/sec
[ 7] 10.00-10.00 sec 1.00 MBytes 2.11 Gbits/sec
----- [ ID] Interval Transfer Bitrate
[ 7] 0.00-10.00 sec 4.63 GBytes 3.97 Gbits/sec
                                                receiver

X "host: h1"@"mininet-vm
iperf Done.
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 35062 connected to 10.0.0.2 port 5201
[ ID] Interval Transfer Bitrate Retr Cwnd
[ 7] 0.00-1.00 sec 467 MBytes 3.92 Gbits/sec 0 513 KBytes
[ 7] 1.00-2.00 sec 474 MBytes 3.97 Gbytes/sec 0 513 KBytes
[ 7] 2.00-3.00 sec 482 MBytes 4.04 Gbits/sec 0 513 KBytes
[ 7] 3.00-4.00 sec 475 MBytes 3.99 Gbits/sec 0 513 KBytes
[ 7] 4.00-5.00 sec 468 MBytes 3.92 Gbits/sec 0 513 KBytes
[ 7] 5.00-6.00 sec 465 MBytes 3.91 Gbits/sec 0 513 KBytes
[ 7] 6.00-7.00 sec 477 MBytes 4.00 Gbits/sec 0 513 KBytes
[ 7] 7.00-8.00 sec 477 MBytes 4.00 Gbytes/sec 0 513 KBytes
[ 7] 8.00-9.00 sec 481 MBytes 4.04 Gbits/sec 0 513 KBytes
[ 7] 9.00-10.00 sec 481 MBytes 4.04 Gbits/sec 0 513 KBytes
----- [ ID] Interval Transfer Bitrate Retr
[ 7] 0.00-10.00 sec 4.64 GBytes 3.98 Gbits/sec 0
                                                sender
[ 7] 0.00-10.00 sec 4.63 GBytes 3.97 Gbits/sec
                                                receiver

```

Рис. 3.8: Проверка пропускной способности

Удалим модифицированную конфигурацию на хосте h1.

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. 3.9). Проверим примененные ограничения

```

Server listening on 5201
-----
Accepted connection from 10.0.0.1, port 35080
[ 7] local 10.0.0.2 port 5201 connected to 10.0.0.1 port 35082
[ ID] Interval Transfer Bitrate
[ 7] 0.00-1.00 sec 444 MBytes 3.73 Gbits/sec
[ 7] 1.00-2.00 sec 507 MBytes 4.25 Gbits/sec
[ 7] 2.00-3.00 sec 499 MBytes 4.19 Gbits/sec
[ 7] 3.00-4.00 sec 484 MBytes 4.05 Gbits/sec
[ 7] 4.00-5.01 sec 499 MBytes 4.18 Gbits/sec
[ 7] 5.01-6.00 sec 495 MBytes 4.16 Gbits/sec
[ 7] 6.00-7.00 sec 504 MBytes 4.23 Gbits/sec
[ 7] 7.00-8.00 sec 503 MBytes 4.22 Gbits/sec
[ 7] 8.00-9.00 sec 515 MBytes 4.31 Gbits/sec
[ 7] 9.00-10.00 sec 472 MBytes 3.97 Gbits/sec
-----
iperf Done.
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 35082 connected to 10.0.0.2 port 5201
[ ID] Interval Transfer Bitrate Retr Cwnd
[ 7] 0.00-1.00 sec 470 MBytes 3.94 Gbits/sec 0 2.79 MBytes
[ 7] 1.00-2.00 sec 508 MBytes 4.25 Gbits/sec 0 2.79 MBytes
[ 7] 2.00-3.00 sec 499 MBytes 4.18 Gbits/sec 0 2.79 MBytes
[ 7] 3.00-4.00 sec 484 MBytes 4.06 Gbits/sec 0 2.79 MBytes
[ 7] 4.00-5.00 sec 499 MBytes 4.19 Gbits/sec 0 2.79 MBytes
[ 7] 5.00-6.00 sec 495 MBytes 4.15 Gbits/sec 0 2.79 MBytes
[ 7] 6.00-7.00 sec 504 MBytes 4.22 Gbits/sec 0 2.93 MBytes
[ 7] 7.00-8.00 sec 502 MBytes 4.22 Gbits/sec 0 2.93 MBytes
[ 7] 8.00-9.00 sec 515 MBytes 4.32 Gbits/sec 0 2.93 MBytes
[ 7] 9.00-10.00 sec 471 MBytes 3.95 Gbits/sec 0 2.93 MBytes
-----
[ ID] Interval Transfer Bitrate Retr
[ 7] 0.00-10.00 sec 4.83 GBytes 4.15 Gbits/sec 0
[ 7] 0.00-10.01 sec 4.81 GBytes 4.13 Gbits/sec

```

Рис. 3.9: Ограничение скорости на коммутаторах

Удалим модифицированную конфигурацию на коммутаторе s1.

Объединим NETEM и TBF, введя на интерфейсе s1-eth2 коммутатора s1 задержку, джиттер, повреждение пакетов и указав скорость. Убедимся, что соединение от хоста h1 к хосту h2 имеет заданную задержку. Для этого запустим команду ping с параметром -c 4 с терминала хоста h1. (рис. 3.10).

```

root@mininet-vm:/home/mininet# sudo tc qdisc add dev s1-eth2 root handle 1: netem delay 10ms
X "host: h1"@mininet-vm
iperf Done.
root@mininet-vm:/home/mininet# ping 10.0.0.2 -c 4
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=23.4 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=15.1 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=12.3 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=10.9 ms
--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3008ms
rtt min/avg/max/mdev = 10.908/15.438/23.424/4.847 ms

```

Рис. 3.10: Объединение NETEM и TBF

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 внесенные изменения с помощью iPerf3 (рис. 3.11):

```

Accepted connection from 10.0.0.1, port 35100
[ 7] local 10.0.0.2 port 5201 connected to 10.0.0.1 port 35102
[ ID] Interval Transfer Bitrate
[ 7] 0.00-1.01 sec 175 MBytes 1.45 Gbits/sec
[ 7] 1.01-2.01 sec 226 MBytes 1.89 Gbits/sec
[ 7] 2.01-3.01 sec 227 MBytes 1.90 Gbits/sec
[ 7] 3.01-4.00 sec 218 MBytes 1.84 Gbits/sec
[ 7] 4.00-5.01 sec 217 MBytes 1.80 Gbits/sec
[ 7] 5.01-6.00 sec 152 MBytes 1.29 Gbits/sec
[ 7] 6.00-7.00 sec 119 MBytes 998 Mbytes/sec
[ 7] 7.00-8.00 sec 101 MBytes 847 Mbytes/sec
[ 7] 8.00-9.00 sec 111 MBytes 929 Mbytes/sec

x host:h1@mininet-vm
iperf Done.
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 35102 connected to 10.0.0.2 port 5201
[ ID] Interval Transfer Bitrate Retr Cwnd
[ 7] 0.00-1.00 sec 187 MBytes 1.57 Gbits/sec 135 3.21 MBytes
[ 7] 1.00-2.00 sec 226 MBytes 1.90 Gbits/sec 0 3.46 MBytes
[ 7] 2.00-3.00 sec 226 MBytes 1.90 Gbits/sec 0 3.67 MBytes
[ 7] 3.00-4.00 sec 218 MBytes 1.83 Gbits/sec 180 2.71 MBytes
[ 7] 4.00-5.00 sec 216 MBytes 1.81 Gbits/sec 0 2.83 MBytes
[ 7] 5.00-6.00 sec 151 MBytes 1.27 Gbits/sec 270 2.10 MBytes
[ 7] 6.00-7.00 sec 120 MBytes 1.01 Gbits/sec 23 1.56 MBytes
[ 7] 7.00-8.00 sec 101 MBytes 849 Mbytes/sec 0 1.65 MBytes
[ 7] 8.00-9.00 sec 110 MBytes 923 Mbytes/sec 0 1.72 MBytes
[ 7] 9.00-10.00 sec 120 MBytes 1.01 Gbits/sec 0 1.76 MBytes

[ ID] Interval Transfer Bitrate Retr
[ 7] 0.00-10.00 sec 1.64 GBytes 1.41 Gbits/sec 608 sender
[ 7] 0.00-10.01 sec 1.63 GBytes 1.40 Gbits/sec receiver

iperf Done.

```

Рис. 3.11: Объединение NETEM и TBF

Удалим модифицированную конфигурацию на коммутаторе s1.

### 3.3 Воспроизводимые эксперименты

Самостоятельно реализуем воспроизводимые эксперименты по использованию TBF для ограничения пропускной способности. Для этого создадим рабочий каталог и файл со скриптом lab\_netem\_iii.py (рис. 3.12).

```

mininet@mininet-vm:~$ mkdir -p ~/work/lab_netem_iii/simple-tbf
mininet@mininet-vm:~$ cd ~/work/lab_netem_iii/simple-tbf
mininet@mininet-vm:~/work/lab_netem_iii/simple-tbf$ touch lab_netem_iii.py
mininet@mininet-vm:~/work/lab_netem_iii/simple-tbf$ nano lab_netem_iii.py

```

Рис. 3.12: Создание рабочего каталога

Создадим скрипт для эксперимента lab\_netem\_iii.py. В этом скрипте создается простейшая топология сети, затем с помощью команд, использованных нами ранее изменяется пропускная способность для первого хоста, после чего пингуется второй хост (100 сообщений отправляется), при этом из сообщений при пинге вытаскиваются номер сообщения и значение времени, которые записываются в файл с данными. (рис. 3.13).

```

GNU nano 4.8                               lab netem iii.py
#!/usr/bin/env python
"""
Simple experiment.
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 8gbit burst 5000000 limit 10000000')
    time.sleep(10) # Wait 10 seconds

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

```

Рис. 3.13: Скрипт на Python для эксперимента

Создадим Makefile для управления процессом проведения эксперимента и записи данных в файл(рис. 3.14).

```

GNU nano 4.8                               Makefile
all: ping.dat

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

ping.png: ping.dat
    ./ping_plot

clean:
    -rm -f *.dat *.png

```

Рис. 3.14: Makefile для управления процессом проведения эксперимента

Создадим файл ping\_plot для отображения данных (рис. 3.15).

```

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. 3.15: Создание файла для изображения графика

Выполним эксперимент, написав команду make (рис. 3.16).

```
mininet@mininet-vm:~/work/lab_netem_iii/simple-tbf$ make
sudo python lab_netem_iii.py
*** Adding controller
*** Adding hosts
*** Adding switch
*** Creating links
*** Starting network
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Waiting for switches to connect
s1
*** Set rate
*** h1 : ('tc qdisc add dev h1-eth0 root tbf rate 8gbit burst 5000000 limit 10000000',)
*** Starting iperf server on h2
*** h2 : ('iperf3 -s &')
*** Running iperf client from h1 to h2
*** h1 : ('iperf3 -c 10.0.0.2 | grep "MBytes" | awk \'(print $7)\' > ping.dat',)
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
...
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chmod mininet:ping.dat
```

Рис. 3.16: Запуск эксперимента

Изучим созданный график (рис. 3.17).

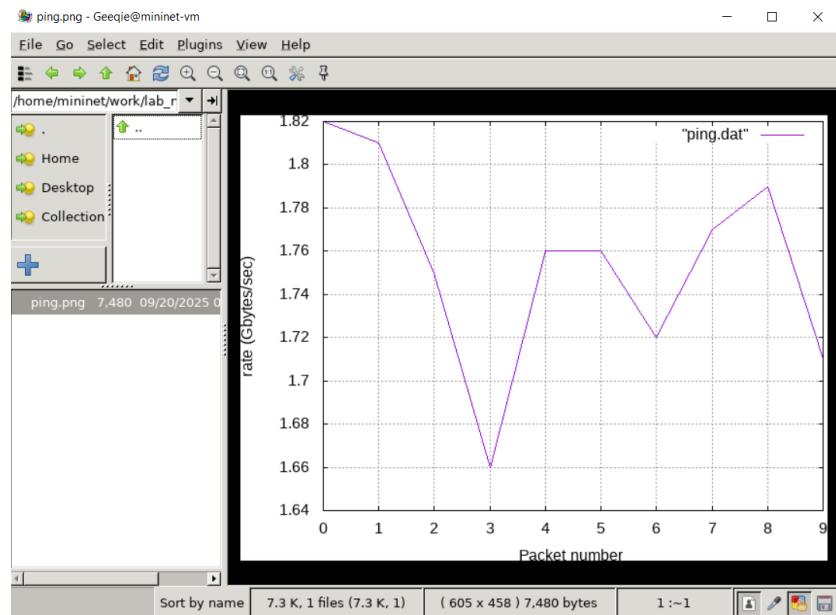


Рис. 3.17: График изменения скорости передачи

## **4 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.