

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра теории вероятностей и кибербезопасности

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №6

дисциплина: Моделирование сетей передачи данных

Студент: Быстров Глеб

Группа: НПИбд-01-20

МОСКВА

2023 г.

Цель работы

В данной лабораторной работе мне будет необходимо познакомиться с принципами работы дисциплины очереди Token Bucket Filter, которая формирует входящий/исходящий трафик для ограничения пропускной способности, а также получить навыки моделирования и исследования поведения трафика посредством проведения интерактивного и воспроизводимого экспериментов в Mininet.

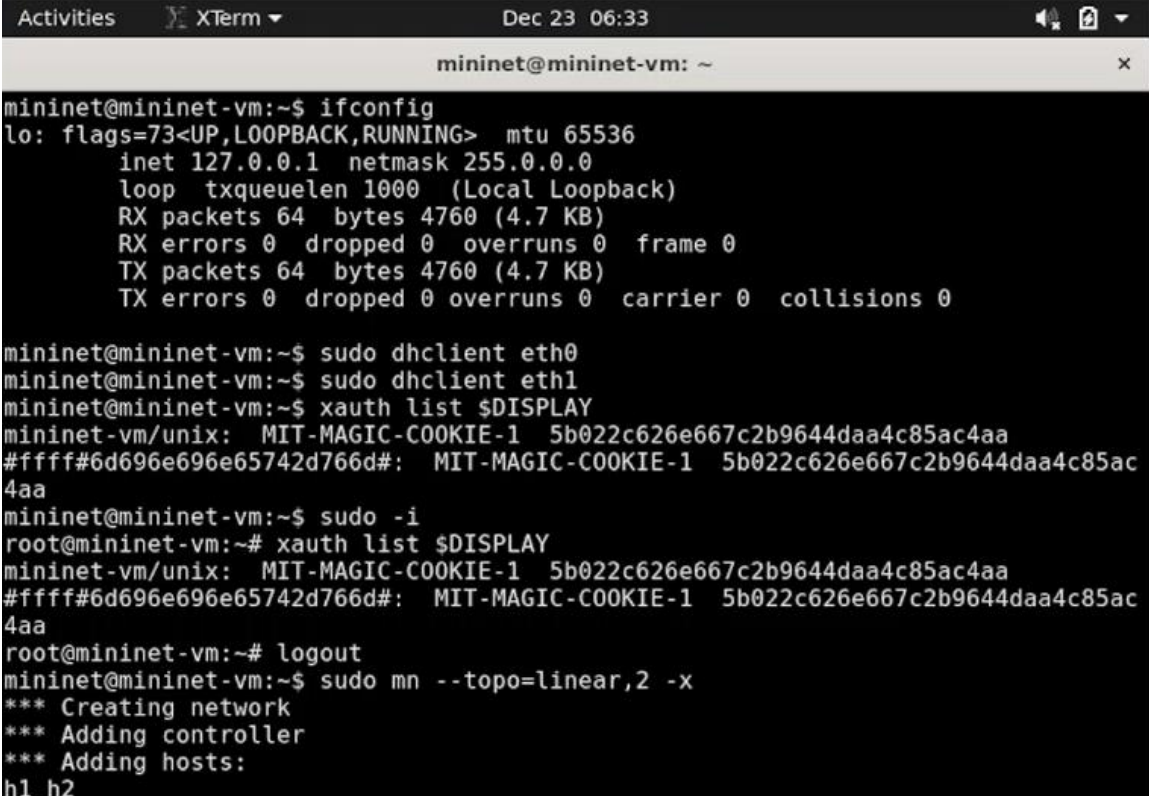
Описание процесса выполнения работы

Запуск лабораторной топологии

1. Запустите виртуальную среду с mininet. Задайте топологию сети, состоящую из двух хостов и двух коммутаторов с назначенной по умолчанию mininet сетью 10.0.0.0/8:

```
sudo mn --topo=linear,2 -x
```

После введения этой команды запустятся терминалы двух хостов, двух коммутаторов и контроллера (рис. 1).



```
Activities  XTerm  Dec 23 06:33
mininet@mininet-vm: ~

mininet@mininet-vm:~$ ifconfig
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 64  bytes 4760 (4.7 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 64  bytes 4760 (4.7 KB)
    TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

mininet@mininet-vm:~$ sudo dhclient eth0
mininet@mininet-vm:~$ sudo dhclient eth1
mininet@mininet-vm:~$ xauth list $DISPLAY
mininet-vm/unix: MIT-MAGIC-COOKIE-1  5b022c626e667c2b9644daa4c85ac4aa
#ffff#6d696e696e65742d766d#: MIT-MAGIC-COOKIE-1  5b022c626e667c2b9644daa4c85ac4aa
mininet@mininet-vm:~$ sudo -i
root@mininet-vm:~# xauth list $DISPLAY
mininet-vm/unix: MIT-MAGIC-COOKIE-1  5b022c626e667c2b9644daa4c85ac4aa
#ffff#6d696e696e65742d766d#: MIT-MAGIC-COOKIE-1  5b022c626e667c2b9644daa4c85ac4aa
root@mininet-vm:~# logout
mininet@mininet-vm:~$ sudo mn --topo=linear,2 -x
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
```

Рис. 1. Запуск топологии сети

2. На хостах h1, h2 и на коммутаторах s1, s2 введите 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 (рис. 2-5).

```

"host: h1"
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

root@mininet-virtual-machine:/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 7e:c8:48:eb:47:bf 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 txqueuelen 1000 (Local Loopback)
    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

```

Рис. 2. h1

```

"host: h2"
root@mininet-virtual-machine:/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 66:8a:7b:38:7f:19 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 txqueuelen 1000 (Local Loopback)
    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

```

Рис. 3. h2

```
"switch: s1" (root)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
ether a2:dd:4b:6a:3c:68 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
s2-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
ether de:cc:1b:a5:a0:0f 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
s2-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
ether 5e:17:37:9a:de:37 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
root@mininet-vm:/home/mininet#
```

Рис. 4. s1

```
"switch: s2" (root)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
ether a2:dd:4b:6a:3c:68 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
s2-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
ether de:cc:1b:a5:a0:0f 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
s2-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
ether 5e:17:37:9a:de:37 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
```

Рис. 5. s2

3. Проверьте подключение между хостами h1 и h2 с помощью команды ping с параметром -c 4 (рис. 6).

```

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=10.3 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=1.11 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.086 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.119 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3037ms
rtt min/avg/max/mdev = 0.086/2.899/10.280/4.281 ms

```

Рис. 6. Проверка подключения

4. В терминале хоста h2 запустите iPerf3 в режиме сервера (рис. 7).

```

root@mininet-vm:/home/mininet# iperf3 -s
warning: this system does not seem to support IPv6 - trying IPv4
-----
Server listening on 5201
-----

```

Рис. 7. Режим сервера

5. В терминале хоста h1 запустите iPerf3 в режиме клиента (рис. 8).

```

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 50670 connected to 10.0.0.2 port 5201
[ ID] Interval            Transfer          Bitrate          Retr  Cwnd
[ 7]  0.00-1.01 sec      1.20 GBytes      10.2 Gbits/sec    0     8.19 MBytes
[ 7]  1.01-2.00 sec      1.10 GBytes      9.56 Gbits/sec   0     8.19 MBytes
[ 7]  2.00-3.00 sec      1.23 GBytes      10.5 Gbits/sec   0     8.19 MBytes
[ 7]  3.00-4.02 sec      1.18 GBytes      9.98 Gbits/sec   0     8.19 MBytes
[ 7]  4.02-5.01 sec      1.27 GBytes      10.9 Gbits/sec   1     8.19 MBytes
[ 7]  5.01-6.01 sec      1.21 GBytes      10.5 Gbits/sec   2     8.19 MBytes
[ 7]  6.01-7.01 sec      1.28 GBytes      11.0 Gbits/sec   0     8.19 MBytes
[ 7]  7.01-8.00 sec      1.27 GBytes      11.0 Gbits/sec   2     8.19 MBytes
[ 7]  8.00-9.01 sec      1.18 GBytes      10.0 Gbits/sec   0     8.19 MBytes
[ 7]  9.01-10.02 sec     1.12 GBytes      9.62 Gbits/sec   0     8.19 MBytes
-----
[ ID] Interval            Transfer          Bitrate          Retr
[ 7]  0.00-10.02 sec     12.0 GBytes      10.3 Gbits/sec    5
[ 7]  0.00-10.03 sec     12.0 GBytes      10.3 Gbits/sec
iperf Done.

```

Рис. 8. Режим клиента

6. После завершения работы iPerf3 на хосте h1 остановите iPerf3 на хосте h2, нажав Ctrl + c . В отчёте зафиксируйте результат отработки iPerf3 на данном этапе проведения эксперимента, когда отсутствуют ограничения скорости передачи данных. (рис. 9).


```
"host: h2"
Server listening on 5201
-----
Accepted connection from 10.0.0.1, port 50668
[ 7] local 10.0.0.2 port 5201 connected to 10.0.0.1 port 50670
[ ID] Interval          Transfer      Bitrate
[ 7] 0.00-1.00 sec      1.18 GBytes  10.1 Gbits/sec
[ 7] 1.00-2.00 sec      1.10 GBytes  9.48 Gbits/sec
[ 7] 2.00-3.00 sec      1.22 GBytes  10.5 Gbits/sec
[ 7] 3.00-4.00 sec      1.18 GBytes  10.1 Gbits/sec
[ 7] 4.00-5.00 sec      1.27 GBytes  10.8 Gbits/sec
[ 7] 5.00-6.00 sec      1.22 GBytes  10.6 Gbits/sec
[ 7] 6.00-7.00 sec      1.26 GBytes  10.8 Gbits/sec
[ 7] 7.00-8.00 sec      1.28 GBytes  11.0 Gbits/sec
[ 7] 8.00-9.00 sec      1.17 GBytes  10.1 Gbits/sec
[ 7] 9.00-10.00 sec     1.13 GBytes  9.71 Gbits/sec
[ 7] 10.00-10.03 sec    21.7 MBytes  6.18 Gbits/sec
-----
[ ID] Interval          Transfer      Bitrate
[ 7] 0.00-10.03 sec    12.0 GBytes  10.3 Gbits/sec
-----
Server listening on 5201
-----
^Ciperf3: interrupt - the server has terminated
root@mininet-vm:/home/mininet#
```

Рис. 9. Результаты отработки

Ограничение скорости на конечных хостах

7. Команду `tc` можно применить к сетевому интерфейсу устройства для формирования исходящего трафика. Требуется ограничить скорость отправки данных с конечного хоста с помощью фильтра Token Bucket Filter (tbf).

Измените пропускную способность хоста `h1`, установив пропускную способность на 10 Гбит/с на интерфейсе `h1-eth0` и параметры TBF-фильтра:

```
sudo tc qdisc add dev h1-eth0 root tbf rate 10gbit burst 5000000 limit 15000000
```

Здесь:

- `sudo`: включить выполнение команды с более высокими привилегиями безопасности;
- `tc`: вызвать управление трафиком Linux;
- `qdisc`: изменить дисциплину очередей сетевого планировщика;
- `add` (добавить): создать новое правило;

- dev h1-eth0 root: интерфейс, на котором будет применяться правило;
- tbf: использовать алгоритм Token Bucket Filter;
- rate: указать скорость передачи (10 Гбит/с);
- burst: количество байтов, которое может поместиться в корзину (5 000 000);
- limit: размер очереди в байтах (15 000 000) (рис. 10).

```
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root tbf rate 10
qbit burst 5000000 limit 15000000
```

Рис. 10. Изменение пропускной способности

8. Фильтр tbf требует установки значения всплеска при ограничении скорости. Это значение должно быть достаточно высоким, чтобы обеспечить установленную скорость. Она должна быть не ниже указанной частоты, делённой на HZ, где HZ — тактовая частота, настроенная как параметр ядра, и может быть извлечена с помощью следующей команды:

`egrep '^CONFIG_HZ_[0-9]+' /boot/config-`uname -r`` (рис. 11).

Для расчёта значения всплеска (burst) необходимо скорость передачи (10 Гбит/с или 10 Gbps = 10,000,000,000 bps) разделить на полученное таким образом значение HZ (на хосте h1 HZ = 250): $Burst = 10,000,000,000 / 250 = 40,000,000 \text{ bits} = 40,000,000 / 8 \text{ bytes} = 5,000,000 \text{ bytes}$.

```
root@mininet-vm:/home/mininet# egrep '^CONFIG_HZ_[0-9]+' /boot/config-`uname
-r`
CONFIG_HZ_250=y
root@mininet-vm:/home/mininet#
```

Рис. 11. Задание фильтра

9. С помощью iPerf3 проверьте, что значение пропускной способности изменилось:

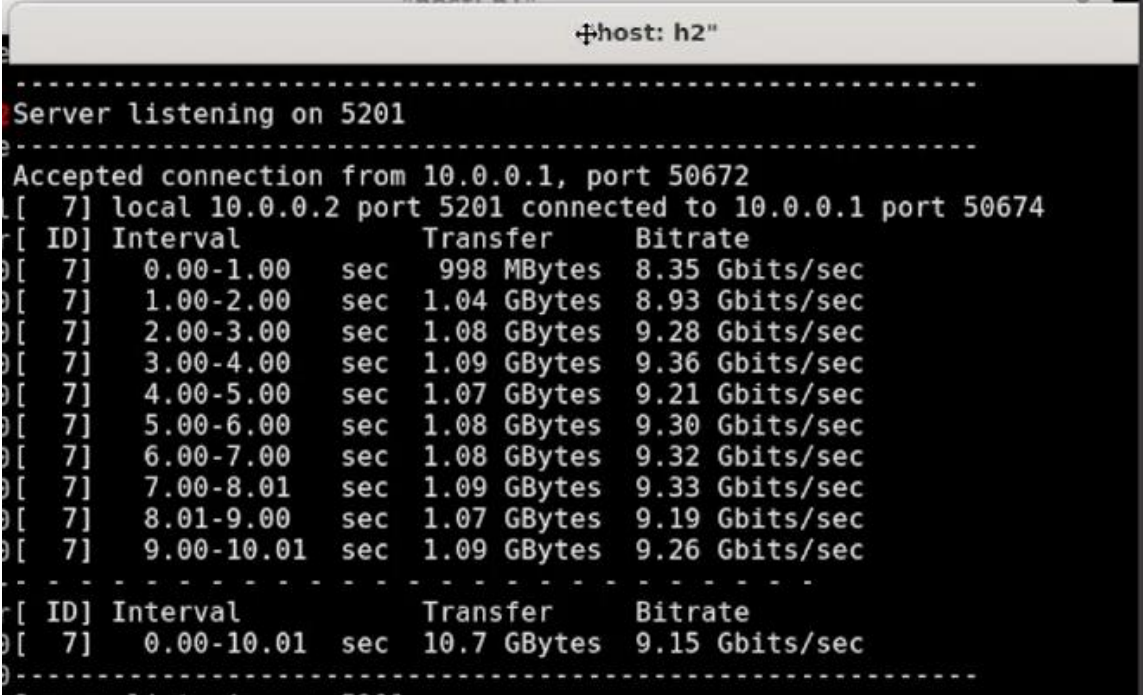
В терминале хоста h2 запустите iPerf3 в режиме сервера:

`iperf3 -s`

В терминале хоста h2 запустите iPerf3 в режиме клиента:

iperf3 -c 10.0.0.2

После завершения работы iPerf3 на хосте h1 остановите iPerf3 на хосте h2, нажав Ctrl + c . В отчёте зафиксируйте результат отработки iPerf3 на данном этапе проведения эксперимента (рис. 12).



```
host: h2
-----
Server listening on 5201
-----
Accepted connection from 10.0.0.1, port 50672
[ 7] local 10.0.0.2 port 5201 connected to 10.0.0.1 port 50674
[ ID] Interval           Transfer     Bitrate
[ 7] 0.00-1.00   sec    998 MBytes  8.35 Gbits/sec
[ 7] 1.00-2.00   sec   1.04 GBytes  8.93 Gbits/sec
[ 7] 2.00-3.00   sec   1.08 GBytes  9.28 Gbits/sec
[ 7] 3.00-4.00   sec   1.09 GBytes  9.36 Gbits/sec
[ 7] 4.00-5.00   sec   1.07 GBytes  9.21 Gbits/sec
[ 7] 5.00-6.00   sec   1.08 GBytes  9.30 Gbits/sec
[ 7] 6.00-7.00   sec   1.08 GBytes  9.32 Gbits/sec
[ 7] 7.00-8.01   sec   1.09 GBytes  9.33 Gbits/sec
[ 7] 8.01-9.00   sec   1.07 GBytes  9.19 Gbits/sec
[ 7] 9.00-10.01  sec   1.09 GBytes  9.26 Gbits/sec
-----
[ ID] Interval           Transfer     Bitrate
[ 7] 0.00-10.01  sec  10.7 GBytes  9.15 Gbits/sec
-----
```

Рис. 12. Результаты отработки

10. Удалите модифицированную конфигурацию на хосте h1 (рис. 13).

```
sudo tc qdisc del dev h1-eth0 root
```



```
root@mininet-vm:/home/mininet# sudo tc qdisc del dev h1-eth0 root
root@mininet-vm:/home/mininet#
```

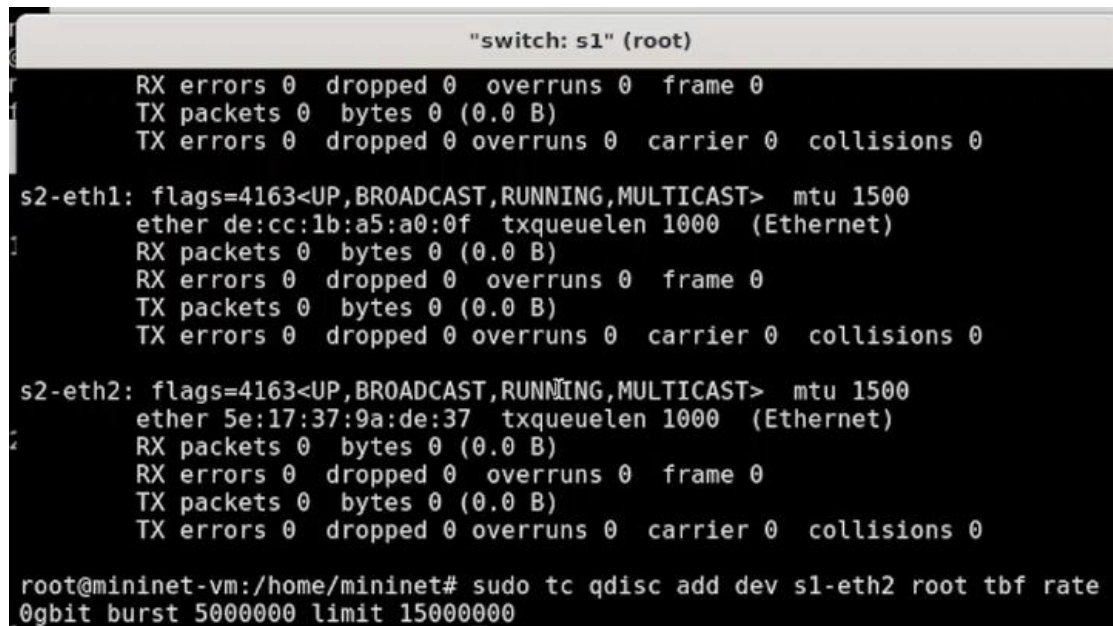
Рис. 13. Удаление конфигурации

Ограничение скорости на коммутаторах

11. При ограничении скорости на интерфейсе s1-eth2 коммутатора s1 все сеансы связи между коммутатором s1 и коммутатором s2 будут фильтроваться в соответствии с применяемыми правилами.

Примените правило ограничения скорости `tbft` с параметрами `rate = 10gbit`, `burst = 5,000,000`, `limit= 15,000,000` к интерфейсу s1-eth2 коммутатора s1, который соединяет его с коммутатором s2:

`sudo tc qdisc add dev s1-eth2 root tbf rate 10gbit burst 5000000 limit 15000000` (рис. 14).



```
"switch: s1" (root)
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

s2-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
ether de:cc:1b:a5:a0:0f  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

s2-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
ether 5e:17:37:9a:de:37  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

root@mininet-vm:/home/mininet# sudo tc qdisc add dev s1-eth2 root tbf rate
0gbit burst 5000000 limit 15000000
```

Рис. 14. Ограничение скорости

12. Проверьте конфигурацию с помощью инструмента `iperf3` для измерения пропускной способности:

В терминале хоста `h2` запустите `iPerf3` в режиме сервера:

`iperf3 -s`

В терминале хоста `h2` запустите `iPerf3` в режиме клиента:

`iperf3 -c 10.0.0.2`

После завершения работы `iPerf3` на хосте `h1` остановите `iPerf3` на хосте `h2`, нажав `Ctrl + c`. В отчёте зафиксируйте результат отработки `iPerf3` на данном этапе проведения эксперимента (рис. 15).

```
"host: h2"
Server listening on 5201
Accepted connection from 10.0.0.1, port 50676
[ 7] local 10.0.0.2 port 5201 connected to 10.0.0.1 port 50678
[ ID] Interval          Transfer      Bitrate
[ 7] 0.00-1.00 sec      1.03 GBytes   8.82 Gbits/sec
[ 7] 1.00-2.00 sec      1.07 GBytes   9.19 Gbits/sec
[ 7] 2.00-3.01 sec      1.07 GBytes   9.15 Gbits/sec
[ 7] 3.01-4.01 sec      1.07 GBytes   9.15 Gbits/sec
[ 7] 4.01-5.00 sec      1.04 GBytes   9.02 Gbits/sec
[ 7] 5.00-6.00 sec      1.05 GBytes   8.98 Gbits/sec
[ 7] 6.00-7.00 sec      1.05 GBytes   9.00 Gbits/sec
[ 7] 7.00-8.00 sec      1.06 GBytes   9.12 Gbits/sec
[ 7] 8.00-9.00 sec      1.03 GBytes   8.87 Gbits/sec
[ 7] 9.00-10.01 sec     1.06 GBytes   8.95 Gbits/sec
[ ID] Interval          Transfer      Bitrate
[ 7] 0.00-10.01 sec    10.5 GBytes   9.03 Gbits/sec
Server listening on 5201
```

Рис. 15. Результаты отработки

13. Удалите модифицированную конфигурацию на коммутаторе s1 (рис. 16):

```
root@mininet-vm:/home/mininet# sudo tc qdisc del dev s1-eth2 root
root@mininet-vm:/home/mininet#
```

Рис. 16. Удаление конфигурации

Объединение NETEM и TBF

14. NETEM используется для изменения задержки, джиттера, повреждения пакетов и т.д. TBF может использоваться для ограничения скорости. Утилита tc позволяет комбинировать несколько модулей. При этом первая дисциплина очереди (qdisc1) присоединяется к корневой метке, последующие дисциплины очереди можно прикрепить к своим родителям, указав правильную метку (рис. 17).

Объедините NETEM и TBF, введя на интерфейсе s1-eth2 коммутатора s1 задержку, джиттер, повреждение пакетов и указав скорость:

```
sudo tc qdisc add dev s1-eth2 root handle 1: netem delay 10ms
```

```

"switch: s1" (root)
ether de:cc:1b:a5:a0:0f 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

s2-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
ether 5e:17:37:9a:de:37 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

root@mininet-vm:/home/mininet# sudo tc qdisc add dev s1-eth2 root tbf rate
0gbit 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

```

Рис. 17. Объединение NETEM и TBF

15. Убедитесь, что соединение от хоста h1 к хосту h2 имеет заданную задержку. Для этого запустите команду `ping` с параметром `-c 4` с терминала хоста h1 (рис. 18).

```

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=14.2 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=12.3 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=10.9 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=12.0 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 10.932/12.372/14.235/1.192 ms

```

Рис. 18. Проверка соединения

16. Добавьте второе правило на коммутаторе s1, которое задаёт ограничение скорости с помощью `tbf` с параметрами `rate=2gbit`, `burst=1,000,000`, `limit=2,000,000` (рис. 19):

```

sudo tc qdisc add dev s1-eth2 parent 1: handle 2: tbf rate 2gbit burst
1000000 limit 2000000

```

```
"switch: s1" (root)
ether de:cc:1b:a5:a0:0f 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

s2-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
ether 5e:17:37:9a:de:37 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

root@mininet-vm:/home/mininet# sudo tc qdisc add dev s1-eth2 root tbf rate
0gbit 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: hand
e 2: tbf rate 2gbit burst 1000000 limit 2000000
root@mininet-vm:/home/mininet# █
```

Рис. 19. Второе правило

17. Проверьте конфигурацию с помощью инструмента `iperf3` для измерения пропускной способности:

В терминале хоста `h2` запустите `iPerf3` в режиме сервера:

`iperf3 -s`

В терминале хоста `h2` запустите `iPerf3` в режиме клиента:

`iperf3 -c 10.0.0.2`

После завершения работы `iPerf3` на хосте `h1` остановите `iPerf3` на хосте `h2`, нажав `Ctrl + c`. В отчёте зафиксируйте результат отработки `iPerf3` на данном этапе проведения эксперимента (рис. 20).


```
"host: h2"
-----
Accepted connection from 10.0.0.1, port 50682
[ 7] local 10.0.0.2 port 5201 connected to 10.0.0.1 port 50684
[ ID] Interval          Transfer      Bitrate
[ 7]  0.00-1.00      sec    163 MBytes    1.37 Gbits/sec
[ 7]  1.00-2.00      sec    127 MBytes    1.06 Gbits/sec
[ 7]  2.00-3.00      sec    121 MBytes    1.01 Gbits/sec
[ 7]  3.00-4.00      sec    149 MBytes    1.25 Gbits/sec
[ 7]  4.00-5.00      sec    153 MBytes    1.29 Gbits/sec
[ 7]  5.00-6.00      sec    154 MBytes    1.29 Gbits/sec
[ 7]  6.00-7.00      sec    158 MBytes    1.33 Gbits/sec
[ 7]  7.00-8.00      sec    114 MBytes    955 Mbits/sec
[ 7]  8.00-9.00      sec    114 MBytes    955 Mbits/sec
[ 7]  9.00-10.00     sec    126 MBytes    1.06 Gbits/sec
[ 7] 10.00-10.02     sec     318 KBytes    142 Mbits/sec
-----
[ ID] Interval          Transfer      Bitrate
[ 7]  0.00-10.02     sec    1.35 GBytes    1.15 Gbits/sec
-----
Server listening on 5201
-----
```

Рис. 20. Результаты отработки

18. Удалите модифицированную конфигурацию на коммутаторе s1 (рис. 21):

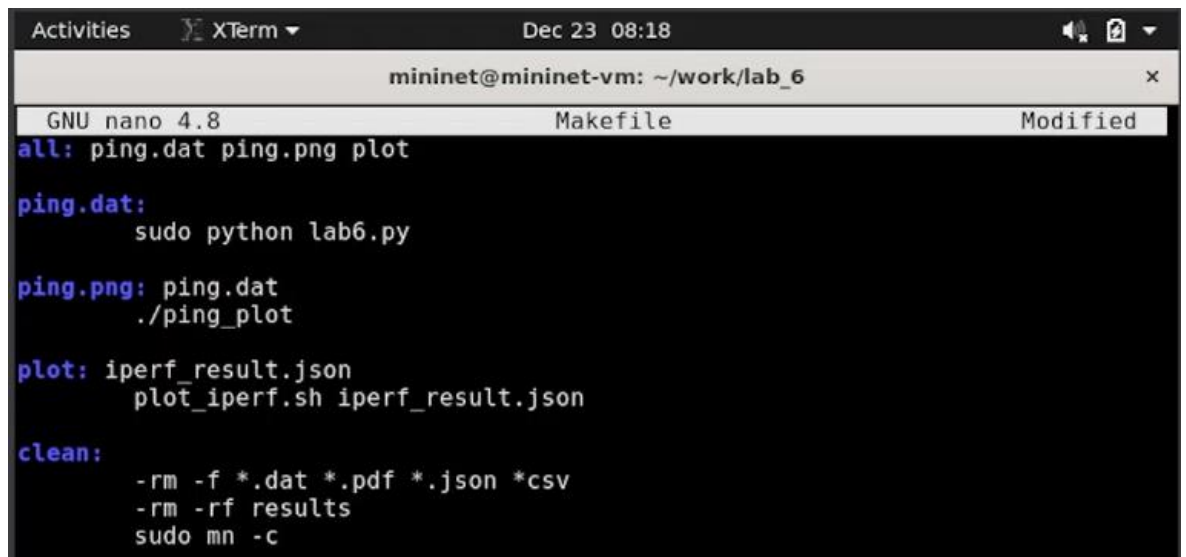
```
sudo tc qdisc del dev s1-eth2 root
```

```
root@mininet-vm:/home/mininet# sudo tc qdisc del dev s1-eth2 root
s1:root@mininet-vm:/home/mininet#
```

Рис. 21. Удаление конфигурации

Воспроизводимые эксперименты

19. Самостоятельно реализуйте воспроизводимые эксперименты по использованию TBF для ограничения пропускной способности. Постройте соответствующие графики. Для начала разработал скрипт Makefile (рис. 22):



The screenshot shows a terminal window titled 'mininet@mininet-vm: ~/work/lab_6'. The terminal is running GNU nano 4.8, editing a file named 'Makefile'. The content of the Makefile is as follows:

```
all: ping.dat ping.png plot

ping.dat:
    sudo python lab6.py

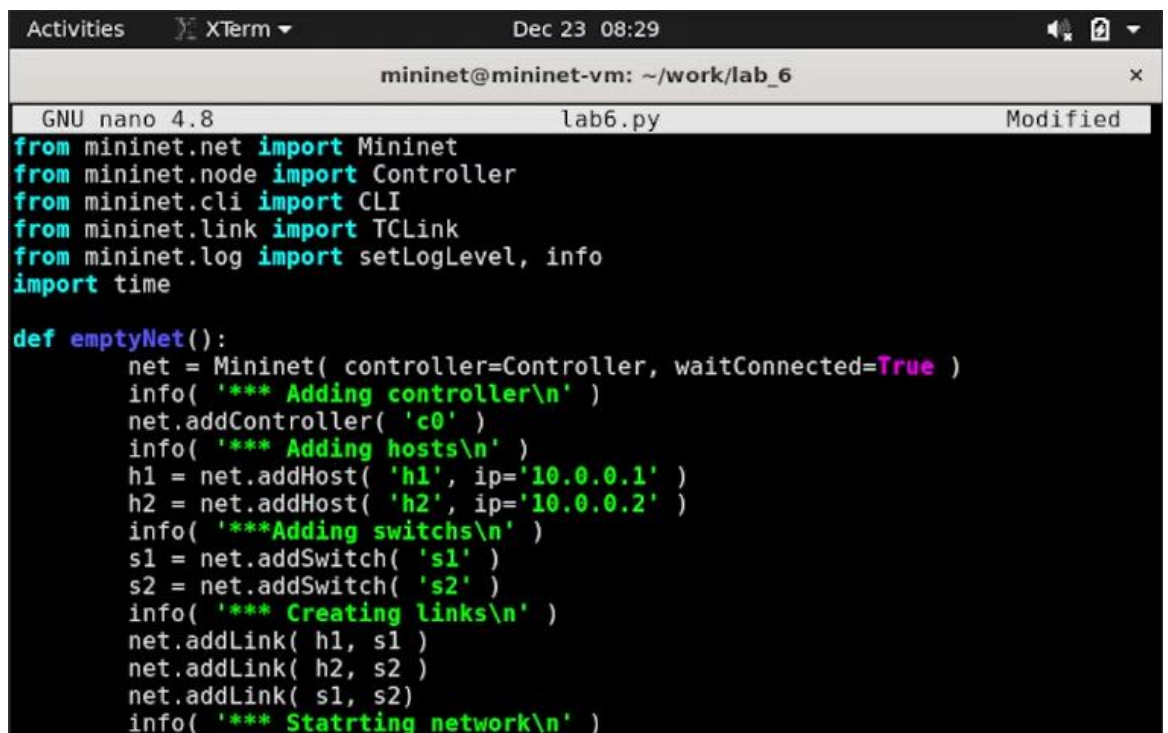
ping.png: ping.dat
    ./ping_plot

plot: iperf_result.json
    plot_iperf.sh iperf_result.json

clean:
    -rm -f *.dat *.pdf *.json *.csv
    -rm -rf results
    sudo mn -c
```

Рис. 22. Makefile

20. Написал код на языке Python для воспроизведения эксперимента (рис. 23-24).



The screenshot shows a terminal window titled 'mininet@mininet-vm: ~/work/lab_6'. The terminal is running GNU nano 4.8, editing a file named 'lab6.py'. The content of the Python code is as follows:

```
from mininet.net import Mininet
from mininet.node import Controller
from mininet.cli import CLI
from mininet.link import TCLink
from mininet.log import setLogLevel, info
import time

def emptyNet():
    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 switches\n' )
    s1 = net.addSwitch( 's1' )
    s2 = net.addSwitch( 's2' )
    info( '*** Creating links\n' )
    net.addLink( h1, s1 )
    net.addLink( h2, s2 )
    net.addLink( s1, s2 )
    info( '*** Statrting network\n' )
```

Рис. 23. Код для эксперимента


```

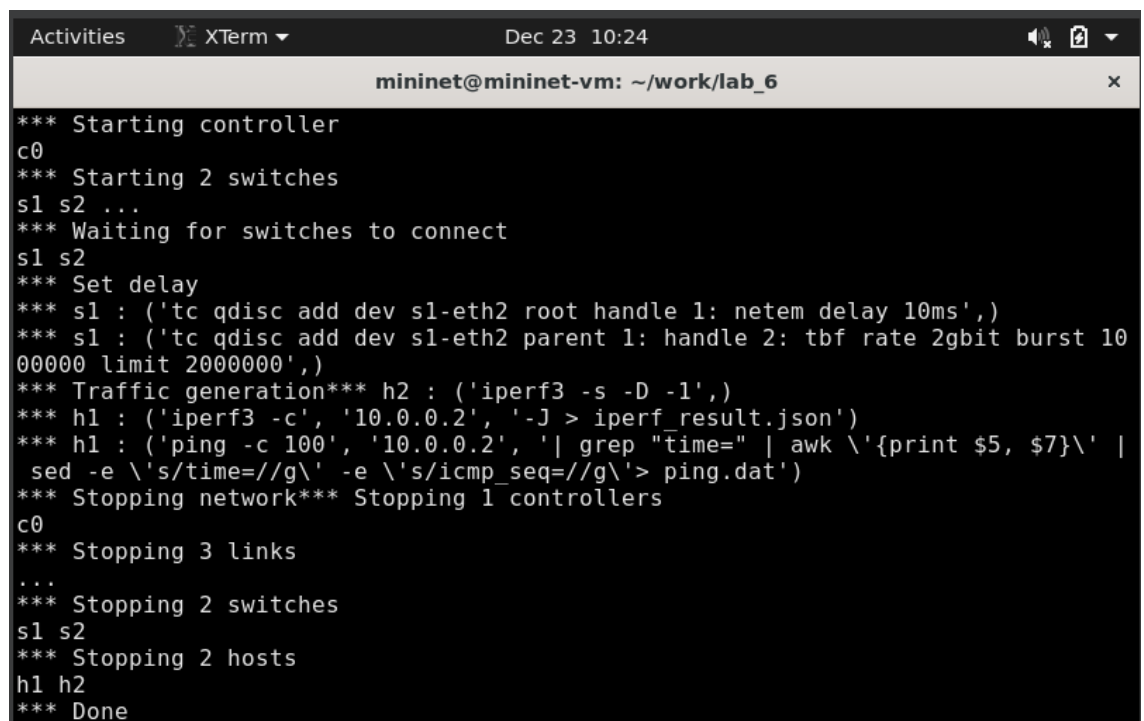
net.start()
info( '*** Set delay\n' )
s1.cmdPrint( 'tc qdisc add dev s1-eth2 root handle 1: netem delay 10ms' )
s1.cmdPrint( 'tc qdisc add dev s1-eth2 parent 1: handle 2: tbf rate 2gbit burst 100000 limit 2000000', )
info( '*** Traffic generation' )
h2.cmdPrint( 'iperf3 -s -D -1' )
time.sleep(10)
h1.cmdPrint( 'iperf3 -c', h2.IP(), '-J > iperf_result.json' )
print '$5, $7}' | sed -e 's/time=//g' -e 's/icmp_seq=//g' > ping.dat' )
info( '*** Stopping network' )
net.stop()

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

```

Рис. 24. Код для эксперимента

21. Запустил Makefile, чтобы выполнить эксперимент (рис. 25):



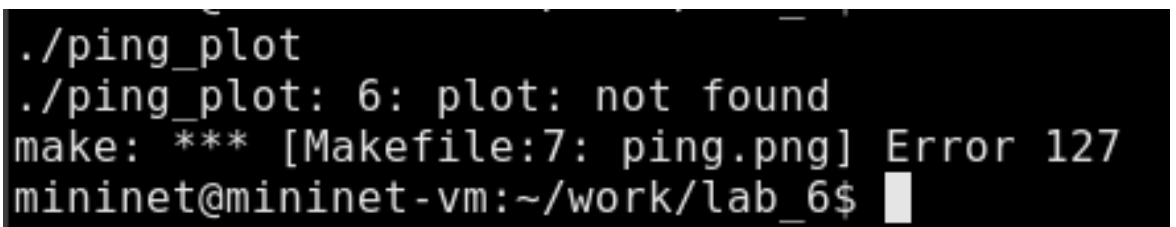
```

mininet@mininet-vm: ~/work/lab_6
*** Starting controller
c0
*** Starting 2 switches
s1 s2 ...
*** Waiting for switches to connect
s1 s2
*** Set delay
*** 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 100000 limit 2000000',)
*** Traffic generation*** h2 : ('iperf3 -s -D -1',)
*** h1 : ('iperf3 -c', '10.0.0.2', '-J > iperf_result.json')
*** h1 : ('ping -c 100', '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

```

Рис. 25. Отработка эксперимента

22. Консоль выдавала ошибку. Около часа пытался решить проблему, но не получилось.



```

./ping_plot
./ping_plot: 6: plot: not found
make: *** [Makefile:7: ping.png] Error 127
mininet@mininet-vm:~/work/lab_6$

```

Рис. 26. Ошибка

Вывод

В данной лабораторной работе мне успешно удалось познакомиться с принципами работы дисциплины очереди Token Bucket Filter, которая формирует входящий/исходящий трафик для ограничения пропускной способности, а также получить навыки моделирования и исследования поведения трафика посредством проведения интерактивного и воспроизводимого экспериментов в Mininet.