

Лабораторная работа № 5

Моделирование сетей передачи данных

Доберштейн Алина Сергеевна

Содержание

1 Цель работы	4
2 Задание	5
3 Теоретическое введение	6
4 Выполнение лабораторной работы	7
5 Интерактивные эксперименты	9
5.1 Добавление потери пакетов на интерфейс, подключённый к эмулируемой глобальной сети	9
5.2 Добавление значения корреляции для потери пакетов в эмулируемой глобальной сети	11
5.3 Добавление повреждения пакетов в эмулируемой глобальной сети	12
5.4 Добавление переупорядочивания пакетов в интерфейс подключения к эмулируемой глобальной сети	13
5.5 Добавление дублирования пакетов в интерфейс подключения к эмулируемой глобальной сети	15
6 Воспроизводимые эксперименты	16
6.1 Добавление потери пакетов	16
6.2 Добавление значения корреляции	18
6.3 Добавление повреждения пакетов	20
6.4 Добавление переупорядочивания пакетов	21
6.5 Добавление дублирования пакетов	23
7 Выводы	26
Список литературы	27

Список иллюстраций

4.1	Запуск простой топологии	7
4.2	Проверка соединения между хостами	8
5.1	Добавление потери пакетов	9
5.2	Добавление потери пакетов	10
5.3	Добавление потери пакетов	11
5.4	Конфигурация по умолчанию	11
5.5	Добавление значения корреляции для потери пакетов	12
5.6	Добавление значения корреляции для потери пакетов	12
5.7	Добавление повреждения пакетов	13
5.8	Добавление переупорядочивания пакетов	14
5.9	Добавление переупорядочивания пакетов	14
5.10	Добавление дублирования пакетов	15
6.1	Создание скрипта для эксперимента lab_netem_ii.py	16
6.2	Редактирование скрипта	17
6.3	Makefile	17
6.4	Проведение эксперимента	18
6.5	Создание скрипта для эксперимента lab_netem_ii.py	18
6.6	Редактирование скрипта	19
6.7	Проведение эксперимента	19
6.8	Создание скрипта для эксперимента lab_netem_ii.py	20
6.9	Редактирование скрипта	20
6.10	Проведение эксперимента	21
6.11	Создание скрипта для эксперимента lab_netem_ii.py	21
6.12	Редактирование скрипта	22
6.13	Проведение эксперимента	23
6.14	Создание скрипта для эксперимента lab_netem_ii.py	23
6.15	Редактирование скрипта	24
6.16	Проведение эксперимента	25

1 Цель работы

Основной целью работы является получение навыков проведения интерактивных экспериментов в среде Mininet по исследованию параметров сети, связанных с потерей, дублированием, изменением порядка и повреждением пакетов при передаче данных. Эти параметры влияют на производительность протоколов и сетей.

2 Задание

1. Задайте простейшую топологию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8.
2. Проведите интерактивные эксперименты по исследованию параметров сети, связанных с потерей, дублированием, изменением порядка и повреждением пакетов при передаче данных.
3. Реализуйте воспроизводимый эксперимент по добавлению правила отбрасывания пакетов в эмулируемой глобальной сети. На экран выведите сводную информацию о потерянных пакетах.
4. Самостоятельно реализуйте воспроизводимые эксперименты по исследованию параметров сети, связанных с потерей, изменением порядка и повреждением пакетов при передаче данных. На экран выведите сводную информацию о потерянных пакетах.

3 Теоретическое введение

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.

4 Выполнение лабораторной работы

Запустила виртуальную среду с mininet. Подключилась из ОС. Проверила права запуска X-соединения. Задала простейшую топологию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8. На хостах h1 и h2 ввела 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. 4.1).

```
mininet@mininet-vm:~/work$ xauth list $DISPLAY
xfrd: command not found
mininet@mininet-vm:~/work$ xauth list $DISPLAY
xauth: command not found
mininet@mininet-vm:~/work$xauth list $DISPLAY
mininet-vm:unix:12 MIT-MAGIC-COOKIE-1 9713b86bb098fbfdb897dd6199ab032e
mininet@mininet-vm:~/work$ sudo -i
root@mininet-vm:~# xauth list $DISPLAY
mininet-vm:unix:12 MIT-MAGIC-COOKIE-1 9713b86bb098fbfdb897dd6199ab032e
root@mininet-vm:~# logout
mininet@mininet-vm:~/work$ sudo mn --topo=single,2 -x
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
```

Рис. 4.1: Запуск простой топологии

Проверила подключение между хостами h1 и h2 с помощью команды ping с параметром -c 6 (рис. 4.2).

```

"host:h2"
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 890 bytes 252928 (252.9 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 890 bytes 252928 (252.9 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet/work# ping -c 6 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=1.40 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.195 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.050 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.079 ms
64 bytes from 10.0.0.1: icmp_seq=5 ttl=64 time=0.058 ms
64 bytes from 10.0.0.1: icmp_seq=6 ttl=64 time=0.038 ms

--- 10.0.0.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5087ms
rtt min/avg/max/mdev = 0.038/0.303/1.399/0.492 ms
root@mininet-vm:/home/mininet/work# █

ping 2 hosts
e
"host:h1"

root@mininet-vm:/home/mininet/work# 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 46:d1:61:97:e1:91 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 898 bytes 253736 (253.7 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 898 bytes 253736 (253.7 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet/work# ping -c 6 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=0.650 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.085 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.097 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.059 ms

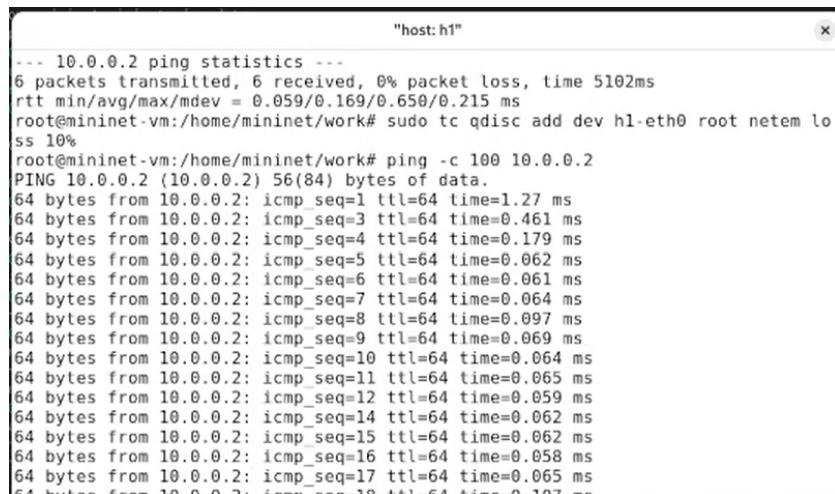
```

Рис. 4.2: Проверка соединения между хостами

5 Интерактивные эксперименты

5.1 Добавление потери пакетов на интерфейс, подключённый к эмулируемой глобальной сети

Пакеты могут быть потеряны в процессе передачи из-за таких факторов, как битовые ошибки и перегрузка сети. Скорость потери данных часто измеряется как процентная доля потерянных пакетов по отношению к количеству отправленных пакетов. На хосте h1 добавила 10% потерь пакетов к интерфейсу h1-eth0 (рис. 5.1-5.2).



```
"host: h1"
--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5102ms
rtt min/avg/max/mdev = 0.059/0.169/0.650/0.215 ms
root@mininet-vm:/home/mininet/work# sudo tc qdisc add dev h1-eth0 root netem lo
ss 10%
root@mininet-vm:/home/mininet/work# ping -c 100 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=1.27 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.461 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.179 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.061 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.064 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=0.097 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=0.069 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=0.064 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=0.065 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=0.059 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=0.058 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=0.065 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=0.107 ms
```

Рис. 5.1: Добавление потери пакетов

```

no
"host: h1"
tc
64 bytes from 10.0.0.2: icmp_seq=82 ttl=64 time=0.073 ms
64 bytes from 10.0.0.2: icmp_seq=83 ttl=64 time=0.064 ms
64 bytes from 10.0.0.2: icmp_seq=84 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=85 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=86 ttl=64 time=0.064 ms
64 bytes from 10.0.0.2: icmp_seq=87 ttl=64 time=0.061 ms
64 bytes from 10.0.0.2: icmp_seq=88 ttl=64 time=0.073 ms
64 bytes from 10.0.0.2: icmp_seq=89 ttl=64 time=0.071 ms
64 bytes from 10.0.0.2: icmp_seq=90 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=91 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=92 ttl=64 time=0.077 ms
64 bytes from 10.0.0.2: icmp_seq=93 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=94 ttl=64 time=0.060 ms
64 bytes from 10.0.0.2: icmp_seq=95 ttl=64 time=0.058 ms
64 bytes from 10.0.0.2: icmp_seq=96 ttl=64 time=0.060 ms
64 bytes from 10.0.0.2: icmp_seq=97 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=98 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=99 ttl=64 time=0.071 ms
64 bytes from 10.0.0.2: icmp_seq=100 ttl=64 time=0.062 ms

--- 10.0.0.2 ping statistics ---
100 packets transmitted, 91 received, 9% packet loss, time 10ms
rtt min/avg/max/mdev = 0.047/0.085/1.270/0.132 ms
root@mininet-vm:/home/mininet/work# 

```

Рис. 5.2: Добавление потери пакетов

Здесь:

- sudo: выполнить команду с более высокими привилегиями;
- tc: вызвать управление трафиком Linux;
- qdisc: изменить дисциплину очередей сетевого планировщика;
- add: создать новое правило;
- dev h1-eth0: указать интерфейс, на котором будет применяться правило;
- netem: использовать эмулятор сети;
- loss 10%: 10% потерь пакетов.

Для эмуляции глобальной сети с потерей пакетов в обоих направлениях необходимо к соответствующему интерфейсу на хосте h2 также добавить 10% потерь пакетов. Проверила, что соединение между хостом h1 и хостом h2 имеет больший процент потерянных данных (10% от хоста h1 к хосту h2 и 10% от хоста h2 к хосту h1), повторив команду ping с параметром -c 100 на терминале хоста h1 (рис. 5.3).

```

root@mininet-vm:/home/mininet/work# sudo tc qdisc add dev h2-eth0 root netem loss 10%
root@mininet-vm:/home/mininet/work# ping 2 hosts
                                         "host: h1" x
ping 2 hosts

64 bytes from 10.0.0.2: icmp_seq=82 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=83 ttl=64 time=0.071 ms
64 bytes from 10.0.0.2: icmp_seq=84 ttl=64 time=0.066 ms
64 bytes from 10.0.0.2: icmp_seq=85 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=86 ttl=64 time=0.065 ms
64 bytes from 10.0.0.2: icmp_seq=87 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=88 ttl=64 time=0.098 ms
64 bytes from 10.0.0.2: icmp_seq=89 ttl=64 time=0.065 ms
64 bytes from 10.0.0.2: icmp_seq=90 ttl=64 time=0.065 ms
64 bytes from 10.0.0.2: icmp_seq=91 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=92 ttl=64 time=0.073 ms
64 bytes from 10.0.0.2: icmp_seq=94 ttl=64 time=0.076 ms
64 bytes from 10.0.0.2: icmp_seq=95 ttl=64 time=0.110 ms
64 bytes from 10.0.0.2: icmp_seq=96 ttl=64 time=0.084 ms
64 bytes from 10.0.0.2: icmp_seq=97 ttl=64 time=0.065 ms
64 bytes from 10.0.0.2: icmp_seq=98 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=99 ttl=64 time=0.070 ms
64 bytes from 10.0.0.2: icmp_seq=100 ttl=64 time=0.067 ms

--- 10.0.0.2 ping statistics ---
100 packets transmitted, 80 received, 20% packet loss, time 101322ms
rtt min/avg/max/mdev = 0.053/0.076/0.592/0.059 ms

```

Рис. 5.3: Добавление потери пакетов

Восстановила конфигурацию по умолчанию. (рис. 5.4).

```

root@mininet-vm:/home/mininet/work# sudo tc qdisc del dev h1-eth0 root netem
root@mininet-vm:/home/mininet/work# ping -c 6 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=0.571 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.065 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.064 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.056 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.078 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.077 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5115ms
rtt min/avg/max/mdev = 0.056/0.151/0.571/0.187 ms

```

Рис. 5.4: Конфигурация по умолчанию

5.2 Добавление значения корреляции для потери пакетов в эмулируемой глобальной сети

Добавила на интерфейсе узла h1 коэффициент потери пакетов 50% (такой высокий уровень потери пакетов маловероятен), и каждая последующая вероятность зависит на 50% от последней: Проверила, что на соединении от хоста h1 к хосту h2 имеются потери пакетов, используя команду ping с параметром -c 50 с хоста h1 (рис. 5.5-5.6).

```

root@mininet-vm:/home/mininet/work# sudo tc qdisc add dev h1-eth0 root netem loss 50% 50%
root@mininet-vm:/home/mininet/work# ping -c 50 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=0.623 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.073 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.080 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=0.065 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=0.066 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=0.087 ms
64 bytes from 10.0.0.2: icmp_seq=21 ttl=64 time=0.068 ms
64 bytes from 10.0.0.2: icmp_seq=22 ttl=64 time=0.070 ms
64 bytes from 10.0.0.2: icmp_seq=23 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=24 ttl=64 time=0.064 ms

```

Рис. 5.5: Добавление значения корреляции для потери пакетов

```

"host: h1"
64 bytes from 10.0.0.2: icmp_seq=23 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=24 ttl=64 time=0.064 ms
64 bytes from 10.0.0.2: icmp_seq=25 ttl=64 time=0.065 ms
64 bytes from 10.0.0.2: icmp_seq=26 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=29 ttl=64 time=0.060 ms
64 bytes from 10.0.0.2: icmp_seq=30 ttl=64 time=0.069 ms
64 bytes from 10.0.0.2: icmp_seq=32 ttl=64 time=0.064 ms
64 bytes from 10.0.0.2: icmp_seq=33 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=39 ttl=64 time=0.068 ms
64 bytes from 10.0.0.2: icmp_seq=40 ttl=64 time=0.070 ms
64 bytes from 10.0.0.2: icmp_seq=42 ttl=64 time=0.065 ms
64 bytes from 10.0.0.2: icmp_seq=43 ttl=64 time=0.066 ms
64 bytes from 10.0.0.2: icmp_seq=44 ttl=64 time=0.137 ms
64 bytes from 10.0.0.2: icmp_seq=45 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=46 ttl=64 time=0.067 ms
64 bytes from 10.0.0.2: icmp_seq=47 ttl=64 time=0.066 ms
64 bytes from 10.0.0.2: icmp_seq=48 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=49 ttl=64 time=0.078 ms
64 bytes from 10.0.0.2: icmp_seq=50 ttl=64 time=0.062 ms
--- 10.0.0.2 ping statistics ---
50 packets transmitted, 28 received, 44% packet loss, time 50166ms
rtt min/avg/max/mdev = 0.060/0.089/0.623/0.103 ms

```

Рис. 5.6: Добавление значения корреляции для потери пакетов

5.3 Добавление повреждения пакетов в эмулируемой глобальной сети

Добавила на интерфейсе узла h1 0,01% повреждения пакетов. Проверила конфигурацию с помощью инструмента iPerf3 для проверки повторных передач (рис. 5.7).

The screenshot shows two terminal windows. The top window is titled "host: h2" and displays an iperf3 test between host h2 (port 5201) and host h1 (port 34958). The bottom window is titled "host: h1" and displays an iperf3 test between host h1 (port 5201) and host h2 (port 34958). Both tests show a transfer rate of 47.6 GBytes over 10 seconds, with a bitrate of 40.9 Gbits/sec. The "retr" column in the h2 log indicates retransmissions for each 1-second interval.

```
"host: h2"
-----
Server listening on 5201
-----
Accepted connection from 10.0.0.1, port 34956
[ 7] local 10.0.0.2 port 5201 connected to 10.0.0.1 port 34958
[ ID] Interval Transfer Bitrate
[ 7] 0.00-1.00 sec 4.91 GBytes 42.2 Gbits/sec
[ 7] 1.00-2.00 sec 4.98 GBytes 42.8 Gbits/sec
[ 7] 2.00-3.00 sec 4.76 GBytes 40.9 Gbits/sec
[ 7] 3.00-4.00 sec 4.69 GBytes 40.3 Gbits/sec
[ 7] 4.00-5.00 sec 5.16 GBytes 44.3 Gbits/sec
[ 7] 5.00-6.00 sec 5.03 GBytes 43.2 Gbits/sec
[ 7] 6.00-7.00 sec 4.17 GBytes 35.8 Gbits/sec
[ 7] 7.00-8.00 sec 4.53 GBytes 38.9 Gbits/sec
[ 7] 8.00-9.00 sec 4.31 GBytes 37.0 Gbits/sec
[ 7] 9.00-10.00 sec 5.06 GBytes 43.4 Gbits/sec
[ 7] 10.00-10.00 sec 1.19 MBytes 7.31 Gbits/sec
[ ID] Interval Transfer Bitrate
[ 7] 0.00-10.00 sec 47.6 GBytes 40.9 Gbits/sec
----- receiver
-----
Server listening on 5201
-----
[ 7] 0.00-10.00 sec 47.6 GBytes 40.9 Gbits/sec
----- receiver

"host: h1"
-----
root@mininet-vm:/home/mininet/work# sudo tc qdisc del dev h1-eth0 root netem
root@mininet-vm:/home/mininet/work# sudo tc qdisc add dev h1-eth0 root netem co
rrupt 0.01%
root@mininet-vm:/home/mininet/work# iperf3 -c 10.0.0.2
Connecting to host 10.0.0.2, port 5201
[ 7] local 10.0.0.1 port 34958 connected to 10.0.0.2 port 5201
[ ID] Interval Transfer Bitrate Retr Cwnd
[ 7] 0.00-1.00 sec 4.92 GBytes 42.2 Gbits/sec 13 1.85 MBytes
[ 7] 1.00-2.00 sec 4.97 GBytes 42.8 Gbits/sec 14 3.20 MBytes
[ 7] 2.00-3.00 sec 4.76 GBytes 40.9 Gbits/sec 12 1.46 MBytes
[ 7] 3.00-4.00 sec 4.70 GBytes 40.3 Gbits/sec 3 1.95 MBytes
[ 7] 4.00-5.00 sec 5.16 GBytes 44.3 Gbits/sec 13 2.29 MBytes
[ 7] 5.00-6.00 sec 5.03 GBytes 43.2 Gbits/sec 9 1017 KBytes
[ 7] 6.00-7.00 sec 4.17 GBytes 35.7 Gbits/sec 10 1.64 MBytes
[ 7] 7.00-8.00 sec 4.53 GBytes 38.9 Gbits/sec 13 1.61 MBytes
[ 7] 8.00-9.00 sec 4.30 GBytes 37.0 Gbits/sec 7 1012 KBytes
[ 7] 9.00-10.00 sec 5.06 GBytes 43.5 Gbits/sec 12 1.40 MBytes
[ ID] Interval Transfer Bitrate Retr
[ 7] 0.00-10.00 sec 47.6 GBytes 40.9 Gbits/sec 106
----- sender
[ 7] 0.00-10.00 sec 47.6 GBytes 40.9 Gbits/sec
----- receiver
----- iperf Done
```

Рис. 5.7: Добавление повреждения пакетов

Значения повторной передачи на каждом временном интервале можно посмотреть в столбце Retr. Например, на 0.00-1.00 интервале retr=13, на следующем 14 и т.д. Всего 106 повторно отправленных пакетов.

5.4 Добавление переупорядочивания пакетов в интерфейс подключения к эмулируемой глобальной сети

Добавим на интерфейсе узла h1 следующее правило: 25% пакетов (со значением корреляции 50%) будут отправлены немедленно, а остальные 75% будут задержаны на 10 мс. Проверим, что на

соединении от хоста h1 к хосту h2 имеются потери пакетов, используя команду ping с параметром -c 20 с хоста h1. Убедимся, что часть пакетов не будут иметь задержки (один из четырех, или 25%), а последующие несколько пакетов будут иметь задержку около 10 миллисекунд (три из четырех, или 75%) (рис. 5.8).

```
root@mininet-vm:/home/mininet/work# sudo tc qdisc del dev h1-eth0 root netem
root@mininet-vm:/home/mininet/work# sudo tc qdisc add dev h1-eth0 root netem de
lay 10ms reorder 25% 50%
root@mininet-vm:/home/mininet/work# ping -c 20 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=11.2 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=10.8 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=10.6 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=10.5 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=10.5 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=10.4 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=11.0 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=11.7 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=10.4 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=10.2 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=10.1 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=10.5 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=10.7 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=10.4 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=10.4 ms
```

Рис. 5.8: Добавление переупорядочивания пакетов

Изменила значение корреляции для более вероятного результата. (рис. 5.9).

```
root@mininet-vm:/home/mininet/work# sudo tc qdisc change dev
delay 10ms reorder 25% 25%
root@mininet-vm:/home/mininet/work# ping -c 20 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=0.297 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=10.9 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=10.5 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=10.4 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=10.4 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=10.4 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=10.3 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=10.8 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=10.7 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=11.1 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=11.1 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=10.8 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=10.3 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=10.5 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=10.6 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=0.063 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=10.8 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=10.5 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=0.058 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=0.149 ms

--- 10.0.0.2 ping statistics ---
20 packets transmitted, 20 received, 0% packet loss, time 196
```

Рис. 5.9: Добавление переупорядочивания пакетов

5.5 Добавление дублирования пакетов в интерфейс подключения к эмулируемой глобальной сети

Для интерфейса узла h1 зададим правило с дублированием 50% пакетов (т.е. 50% пакетов должны быть получены дважды): Проверим, что на соединении от хоста h1 к хосту h2 имеются дублированные пакеты, используя команду ping с параметром -c 20 с хоста h1. Дубликаты пакетов помечаются как DUP! (рис. 5.10).

```
"host: h1"
root@mininet-vm:/home/mininet/work# sudo tc qdisc add dev h1-eth0 root netem du
plicate 50%
root@mininet-vm:/home/mininet/work# ping -c 20 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=1.05 ms
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=1.06 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.457 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.235 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.067 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.067 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.068 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.068 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.074 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.067 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=0.067 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=0.067 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=0.067 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=0.075 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=0.075 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=0.080 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=0.080 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=0.062 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=0.094 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=0.071 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=0.071 ms (DUP!)
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=0.059 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=0.059 ms (DUP!)
```

Рис. 5.10: Добавление дублирования пакетов

6 Воспроизводимые эксперименты

6.1 Добавление потери пакетов

В виртуальной среде mininet в своём рабочем каталоге с проектами создадим каталог simple-drop и перейдем в него. Создадим скрипт для эксперимента lab_netem_ii.py (рис. 6.1).

```
mininet@mininet-vm:~/work$ mkdir -p ~/work/lab_netem_ii/simple-drop
mininet@mininet-vm:~/work$ cd ~/work/lab_netem_ii/simple-drop
mininet@mininet-vm:~/work/lab_netem_ii/simple-drop$ touch lab_netem_ii.py
mininet@mininet-vm:~/work/lab_netem_ii/simple-drop$ chmod +x lab_netem_ii.py
mininet@mininet-vm:~/work/lab_netem_ii/simple-drop$ cp ~/work/lab_netem_ii/correlation-delay/lab_netem_i.py ~/work/lab_netem_ii/simple-drop/lab_netem_ii.py
mininet@mininet-vm:~/work/lab_netem_ii/simple-drop$ ls
lab_netem_ii.py
mininet@mininet-vm:~/work/lab_netem_ii/simple-drop$ more lab_netem_ii.py
```

Рис. 6.1: Создание скрипта для эксперимента lab_netem_ii.py

Скорректируем скрипт так, чтобы на экран или в отдельный файл выводилась информация о потерях пакетов (рис. 6.2).

```

/home/mininet/work/lab_neterm_ii/simple-drop/lab_neterm_ii.py [-M--] 0 L:[ 1+44 45/ 51 +(1084/1197b) 10 0x0A
/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 switches')
    s1 = net.addSwitch( 's1' )

    info('*** Creating links\n')
    net.addLink( h1, s1 )
    net.addLink( h2, s1 )

    info('*** Starting network\n')
    net.start()

    info('*** Set loglevel')
    h1.cmdPrint('tc qdisc add dev h1-eth0 root netem loss 10%')
    h2.cmdPrint('tc qdisc add dev h2-eth0 root netem loss 10%')

    time.sleep(10)

    info('*** Ping\n')
    h1.cmdPrint('ping -c 100', h2.IP(), '| grep "times" | awk \'{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. 6.2: Редактирование скрипта

Создадим Makefile для управления процессом проведения эксперимента (рис. 6.3).

```

/home/mininet/work/lab_neterm_ii/simple-drop/Makefile [-M--] 20 L:[ 1
all: ping.dat

ping.dat:
<-----> sudo python lab_neterm_ii.py
<-----> sudo chown mininet:mininet ping.dat

clean:
<-----> rm -f *.dat

```

Рис. 6.3: Makefile

Выполним эксперимент (рис. 6.4).

```

mininet@mininet-vm:~/work/lab_netem_ii/simple-drop$ make
sudo python lab_netem_ii.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 loss
*** h1 : ('tc qdisc add dev h1-eth0 root netem loss 10%',)
*** h2 : ('tc qdisc add dev h2-eth0 root netem loss 10%',)
*** Ping
*** h1 : ('ping -c 100', '10.0.0.2', '| grep "packet loss" | awk \'{print $6, $7, $8}\' > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
...
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
mininet@mininet-vm:~/work/lab_netem_ii/simple-drop$ cat oing.dat
cat: oing.dat: No such file or directory
mininet@mininet-vm:~/work/lab_netem_ii/simple-drop$ cat ping.dat
16% packet loss,

```

Рис. 6.4: Проведение эксперимента

6.2 Добавление значения корреляции

В виртуальной среде mininet в своём рабочем каталоге с проектами создадим каталог correaltion-drop и перейдем в него. Создадим скрипт для эксперимента lab_netem_ii.py (рис. 6.5).

```

mininet@mininet-vm:~/work/lab_netem_ii/simple-drop$ cd ..
mininet@mininet-vm:~/work/lab_netem_ii$ mkdir -p correlation-drop
mininet@mininet-vm:~/work/lab_netem_ii$ cd correlation-drop
mininet@mininet-vm:~/work/lab_netem_ii$ cp ~/work/lab_netem_ii/simple-drop/Makefile ~/work/lab_netem_ii/correlation-drop/Makefile
cp: cannot stat '/home/mininet/work/lab_netem_ii/simple-drop/Makefile': No such file or directory
mininet@mininet-vm:~/work/lab_netem_ii$ cp ~/work/lab_netem_ii/simple-drop/Makefile ~/work/lab_netem_ii/correlation-drop/Makefile
mininet@mininet-vm:~/work/lab_netem_ii$ cp ~/work/lab_netem_ii/simple-drop/lab_netem_ii.py ~/work/lab_netem_ii/correlation-drop/lab_netem_ii.py
lab_netem_ii.py Makefile
mininet@mininet-vm:~/work/lab_netem_ii/correlation-drop$ mcedit lab_netem_ii.py

```

Рис. 6.5: Создание скрипта для эксперимента lab_netem_ii.py

Скорректируем скрипт так, чтобы на экран или в отдельный файл выводилась информация о потерях пакетов (рис. 6.6).

```

/home/mininet/work/lab_netem_if/correlation-drop/lab_netem_if.py [-H--] 65 L:[ 1+38 39/ 51 ]*(905 /1169b) 39 6x027
/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 loss\n')
    h1.cmdPrint('tc qdisc add dev h1-eth0 root netem loss 50% 50%')
    h2.cmdPrint('tc qdisc add dev h2-eth0 root netem loss 50% 50%')

    time.sleep(10)

    info('*** Ping\n')
    h1.cmdPrint('ping -c 100', h2.IP(), '| grep "packet loss" | awk \'[print $6, $7, $8]\' > ping.dat')

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

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

```

Рис. 6.6: Редактирование скрипта

Выполним эксперимент (рис. 6.7).

```

mininet@mininet-vm:/work/lab_netem_if/correlation-drop$ make
sudo python lab_netem_if.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 loss
*** h1 : ('tc qdisc add dev h1-eth0 root netem loss 50% 50%',)
*** Ping
*** h1 : ('ping -c 100', '10.0.0.2', '| grep "packet loss" | awk \'[print $6, $7, $8]\' > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
...
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
mininet@mininet-vm:/work/lab_netem_if/correlation-drop$ cat ping.dat
58% packet loss,

```

Рис. 6.7: Проведение эксперимента

6.3 Добавление повреждения пакетов

В виртуальной среде mininet в своём рабочем каталоге с проектами создадим каталог simple-corrupt и перейдем в него. Создадим скрипт для эксперимента lab_netem_ii.py (рис. 6.8).

```
mininet@mininet-vm:~/work/lab_netem_ii/simple-corrupt$ cd ..
mininet@mininet-vm:~/work/lab_netem_ii$ mkdir -p simple-corrupt
mininet@mininet-vm:~/work/lab_netem_ii$ cd simple-corrupt
mininet@mininet-vm:~/work/lab_netem_ii/simple-corrupt$ cp ~/work/lab_netem_ii/simple-drop/lab_netem_ii.py ~/work/lab_netem_ii/simple-corrupt/lab_netem_ii.py
mininet@mininet-vm:~/work/lab_netem_ii/simple-corrupt$ cp ~/work/lab_netem_ii/simple-corrupt ~/work/lab_netem_ii/simple-corrupt/Makefile
mininet@mininet-vm:~/work/lab_netem_ii/simple-corrupt$ ls
lab_netem_ii.py  Makefile
mininet@mininet-vm:~/work/lab_netem_ii/simple-corrupt$ mcedit lab_netem_ii.py
```

Рис. 6.8: Создание скрипта для эксперимента lab_netem_ii.py

Скорректируем скрипт так, чтобы на экран или в отдельный файл выводилась информация о потерях пакетов (рис. 6.9).

```
/home/mininet/work/lab_netem_ii/simple-corrupt/lab_netem_ii.py  [-M--] 50 L:[ 1+44 45/ 53 ] +(1006/1123b)  39 0x627
#!/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 corrupt\n')
    h1.cmdPrint('tc qdisc add dev h1-eth0 root netem corrupt 0.01%')
    time.sleep(10)

    info('*** Traffic generation\n')
    h2.cmdPrint('iperf3 -s -D -l 1')
    time.sleep(10)
    h1.cmdPrint('iperf3 -c', h2.IP(), '> ping.dat')

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

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

Рис. 6.9: Редактирование скрипта

Выполним эксперимент (рис. 6.10).

```

mininet@mininet-vm:~/work/lab_netem_ii/simple-corrupt$ make
sudo python lab_netem_ii.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 corrupt
*** h1 : ('tc qdisc add dev h1-eth0 root netem corrupt 0.01%',)
*** h2 : ('iperf3 -s -D -1',)
*** h1 : ('iperf3 -c', '10.0.0.2', ' > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
...
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
mininet@mininet-vm:~/work/lab_netem_ii/simple-corrupt$ cat ping.dat
Connecting to host 10.0.0.2, port 5201
[ 5] local 10.0.0.1 port 35332 connected to 10.0.0.2 port 5201
[ ID] Interval           Transfer     Bitrate    Retr  Cwnd
[ 5]  0.00-1.00   sec  3.18 GBytes   27.3 Gbits/sec   6  1.31 MBytes
[ 5]  1.00-2.00   sec  2.77 GBytes   23.9 Gbits/sec  12  2.21 MBytes
[ 5]  2.00-3.00   sec  2.98 GBytes   25.6 Gbits/sec   4  2.18 MBytes
[ 5]  3.00-4.00   sec  2.56 GBytes   22.0 Gbits/sec   4  1.18 MBytes
[ 5]  4.00-5.00   sec  2.97 GBytes   25.6 Gbits/sec   6  1.40 MBytes
[ 5]  5.00-6.00   sec  2.96 GBytes   25.4 Gbits/sec   6  2.28 MBytes
[ 5]  6.00-7.00   sec  3.34 GBytes   28.7 Gbits/sec   4  1.49 MBytes
[ 5]  7.00-8.00   sec  3.13 GBytes   26.9 Gbits/sec   8  2.92 MBytes
[ 5]  8.00-9.00   sec  3.07 GBytes   26.3 Gbits/sec   7  2.06 MBytes
[ 5]  9.00-10.00  sec  3.33 GBytes   28.6 Gbits/sec   5  1.64 MBytes
- - - - - 
[ ID] Interval           Transfer     Bitrate    Retr
[ 5]  0.00-10.00  sec  30.3 GBytes   26.0 Gbits/sec  62      sender
[ 5]  0.00-10.01  sec  30.3 GBytes   26.0 Gbits/sec          receiver
iperf Done.

```

Рис. 6.10: Проведение эксперимента

6.4 Добавление переупорядочивания пакетов

В виртуальной среде mininet в своём рабочем каталоге с проектами создадим каталог simple-reorder и перейдем в него. Создадим скрипт для эксперимента lab_netem_ii.py (рис. 6.11).

```

mininet@mininet-vm:~/work/lab_netem_ii/simple-corrupt$ cd ..
mininet@mininet-vm:~/work/lab_netem_ii$ mkdir -p simple-reorder
mininet@mininet-vm:~/work/lab_netem_ii$ cp ./simple-corrupt/simple-reorder/* ~/simple-reorder
mininet@mininet-vm:~/work/lab_netem_ii/simple-reorder$ cp ./work/lab_netem_ii/simple-drop/Makefile ./work/lab_netem_ii/simple-reorder/Makefile
mininet@mininet-vm:~/work/lab_netem_ii/simple-reorder$ cp ./work/lab_netem_ii/simple-drop/lab_netem_ii.py ./work/lab_netem_ii/simple-reorder/lab_netem_ii.py
lab_netem_ii.py Makefile
mininet@mininet-vm:~/work/lab_netem_ii/simple-reorder$ mcedit lab_netem_ii.py

```

Рис. 6.11: Создание скрипта для эксперимента lab_netem_ii.py

Скорректируем скрипт так, чтобы на экран или в отдельный файл выводилась информация о потерях пакетов (рис. 6.12).

```

/home/mininet/work/lab_neterm_ii/simple-reorder/lab_neterm_ii.py [-K--] 78 L:[ 1+37 38/ 50] +(853 /1068b) 37 0x625
#!/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 reorder\n')
    h1.cmdPrint('tc qdisc add dev h1-eth0 root netem delay 10ms reorder 25% 25%')
    time.sleep(10)

    info('*** Ping\n')
    h1.cmdPrint('ping -c 20', h2.IP(), '> ping.dat')

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

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

```

Рис. 6.12: Редактирование скрипта

Выполним эксперимент (рис. 6.13).

```

mininet@mininet-vm:~/work/lab_neterm_ii/simple-reorder$ make
sudo python lab_neterm_ii.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 reorder
*** h1 : ('tc qdisc add dev h1-eth0 root netem delay 10ms reorder 25% 25%',)
*** Ping
*** h1 : ('ping -c 20', '10.0.0.2', '> ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
...
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
mininet@mininet-vm:~/work/lab_neterm_ii/simple-reorder$ cat ping.dat
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=22.5 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=10.6 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=11.6 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=10.4 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=11.3 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=10.8 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=12.2 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=10.3 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=10.3 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=11.4 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=11.0 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=10.4 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=10.2 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=10.3 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=11.1 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=0.263 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=10.4 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=12.3 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=0.076 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=10.1 ms

--- 10.0.0.2 ping statistics ---
20 packets transmitted, 20 received, 0% packet loss, time 19078ms
rtt min/avg/max/mdev = 0.076/10.382/22.491/4.283 ms

```

Рис. 6.13: Проведение эксперимента

6.5 Добавление дублирования пакетов

В виртуальной среде mininet в своём рабочем каталоге с проектами создадим каталог simple-duplicate и перейдем в него. Создадим скрипт для эксперимента lab_neterm_ii.py (рис. 6.14).

```

mininet@mininet-vm:~/work/lab_neterm_ii$ cd simple-duplicate
mininet@mininet-vm:~/work/lab_neterm_ii/simple-duplicate$ cp ~/work/lab_neterm_ii/simple-drop/lab_neterm_ii.py ~/work/lab_neterm_ii/simple-duplicate/lab_neterm_ii.py
mininet@mininet-vm:~/work/lab_neterm_ii/simple-duplicate$ cp ~/work/lab_neterm_ii/simple-drop/Makefile ~/work/lab_neterm_ii/simple-duplicate/Makefile
mininet@mininet-vm:~/work/lab_neterm_ii/simple-duplicate$ mcedit lab_neterm_ii.py

```

Рис. 6.14: Создание скрипта для эксперимента lab_neterm_ii.py

Скорректируем скрипт так, чтобы на экран или в отдельный файл выводилась информация о

потерях пакетов (рис. 6.15).

```
/home/mininet/work/lab_neterm_ii/simple-duplicate/lab_neterm_ii.py [~R--] 79 L:[ 1+42 43/ 50 ] *(969 /1097b) 92.6x056
#!/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 duplicate\n')
    h1.cmdPrint('tc qdisc add dev h1-eth0 root netem duplicate 50%')

    time.sleep(10)

    info('*** Ping\n')
    h1.cmdPrint('ping -c 20', h2.IP(), '| grep "duplicates" | awk \'{print $6}\' > ping.dat')

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

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

Рис. 6.15: Редактирование скрипта

Выполним эксперимент (рис. 6.16).

```
mininet@mininet-vm:~/work/lab_netem_ii/simple-duplicate$ make
sudo python lab_netem_ii.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 duplicate
*** h1 : ('tc qdisc add dev h1-eth0 root netem duplicate 50%',)
*** Ping
*** h1 : ('ping -c 20', '10.0.0.2', '| grep "duplicates" | awk "{print $6}"\\ > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
mininet@mininet-vm:~/work/lab_netem_ii/simple-duplicate$ cat ping.dat
20 packets transmitted, 20 received, +8 duplicates, 0% packet loss, time 19432ms
```

Рис. 6.16: Проведение эксперимента

7 Выводы

В процессе выполнения лабораторной работы я получила навыки проведения интерактивных экспериментов в среде Mininet по исследованию параметров сети, связанных с потерей, дублированием, изменением порядка и повреждением пакетов при передаче данных. Эти параметры влияют на производительность протоколов и сетей.

Список литературы