

Лабораторная работа №4

Моделирование сетей передачи данных

Хрусталев Влад Николаевич

Содержание

1 Цель работы	5
2 Теоретическое введение	6
3 Задание	7
4 Выполнение лабораторной работы	8
5 Выводы	30
Список литературы	31

Список иллюстраций

4.1 Исправление прав запуска X-соединения	8
4.2 Простейшая топология	9
4.3 ifconfig на хостах h1 и h2	10
4.4 Проверка подключения между хостами	11
4.5 Добавление задержки в 100мс	12
4.6 Двунаправленная задержка соединения	13
4.7 Изменение задержки на 50мс	14
4.8 Восстановление исходных значений задержки	15
4.9 Добавление значения дрожания задержки в интерфейс подключения	15
4.10 Добавление значения корреляции для джиттера и задержки в интерфейс подключения	16
4.11 Распределение задержки в интерфейсе подключения	17
4.12 Подготовка к производимому эксперименту	17
4.13 Листинг lab_netem_i.py	18
4.14 Листинг ping_plot	18
4.15 Выдача прав выполнения дл ping_plot	19
4.16 Листинг Makefile	19
4.17 Запуск эксперимента	19
4.18 Просмотр графика	20
4.19 Удаление первой строчки из файла ping.dat	20
4.20 Просмотр графика	21
4.21 Разработка скрипта для вычисления на основе данных файла ping.dat статистических данных	21
4.22 Тестирование разработанного скрипта	22
4.23 Добавление правила запуска скрипта в Makefile	22
4.24 Очистка результатов эксперимента	22
4.25 ЛИСТИНГ Воспроизводимый эксперимент по изменению задержки	23
4.26 ВЫВОД Воспроизводимый эксперимент по изменению задержки	23
4.27 ГРАФИК Воспроизводимый эксперимент по изменению задержки	24
4.28 ЛИСТИНГ Воспроизводимый эксперимент по изменению джиттера	25
4.29 ВЫВОД Воспроизводимый эксперимент по изменению джиттера	25
4.30 ГРАФИК Воспроизводимый эксперимент по изменению джиттера	26
4.31 ЛИСТИНГ Воспроизводимый эксперимент по изменению значения корреляции для джиттера и задержки	26
4.32 ВЫВОД Воспроизводимый эксперимент по изменению значения корреляции для джиттера и задержки	27

4.33 ГРАФИК Воспроизводимый эксперимент по изменению значения корреляции для джиттера и задержки	27
4.34 ЛИСТИНГ Воспроизводимый эксперимент по изменению распределения времени задержки в эмулируемой глобальной сети	28
4.35 ВЫВОД Воспроизводимый эксперимент по изменению распределения времени задержки в эмулируемой глобальной сети	28
4.36 ГРАФИК Воспроизводимый эксперимент по изменению распределения времени задержки в эмулируемой глобальной сети	29
4.37 Создание папок под эксперименты	29

1 Цель работы

Основной целью работы является знакомство с NETEM — инструментом для тестирования производительности приложений в виртуальной сети, а также получение навыков проведения интерактивного и воспроизводимого экспериментов по измерению задержки и её дрожания (jitter) в моделируемой сети в среде Mininet.

2 Теоретическое введение

Mininet[1] – 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.

3 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 добавлению/изменению задержки, джиттера, значения корреляции для джиттера и задержки, распределения времени задержки в эмулируемой глобальной сети.
3. Реализуйте воспроизводимый эксперимент по заданию значения задержки в эмулируемой глобальной сети. Постройте график.
4. Самостоятельно реализуйте воспроизводимые эксперименты по изменению задержки, джиттера, значения корреляции для джиттера и задержки, распределения времени задержки в эмулируемой глобальной сети. Постройте графики.

4 Выполнение лабораторной работы

Запустим виртуальную среду с mininet. Из основной ОС подключимся к виртуальной машине. В виртуальной машине mininet при необходимости исправим права запуска X-соединения. Скопируем значение куки (MIT magic cookie) своего пользователя mininet в файл для пользователя root (рис. 4.1).

```
Last login: Sat Oct 25 10:51:51 2025 from 192.168.56.1
mininet@mininet-vm:~$ xauth list $DISPLAY
mininet-vm/unix:10  MIT-MAGIC-COOKIE-1  0f656a238b0a277ac162ec35e2a18e23
mininet@mininet-vm:~$ sudo -i
root@mininet-vm:~# xauth list $DISPLAY
mininet-vm/unix:10  MIT-MAGIC-COOKIE-1  0f656a238b0a277ac162ec35e2a18e23
root@mininet-vm:~# logout
mininet@mininet-vm:~$ |
```

Рис. 4.1: Исправление прав запуска X-соединения

Задайте простейшую топологию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8 (рис. 4.2).

После введения этой команды запустятся терминалы двух хостов, коммутатора и контроллера. Терминалы коммутатора и контроллера можно закрыть.

```
mininet@mininet-vm:~$ 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
*** Running terms on localhost:10.0
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet> █
```

Рис. 4.2: Простейшая топология

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. 4.3).

```

X "host:h1" @mininet-vm
root@mininet-vm:/home/mininet# ifconfig
 1: 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 0e:c3:2d:36:4d:d3 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 1036 bytes 270956 (270.9 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 1036 bytes 270956 (270.9 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# 

X "host:h2" @mininet-vm
root@mininet-vm:/home/mininet# ifconfig
 2: 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 de:2a:5d:c5:ab:c5 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 1406 bytes 307892 (307.8 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 1406 bytes 307892 (307.8 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# 

```

Рис. 4.3: ifconfig на хостах h1 и h2

Проверим подключение между хостами h1 и h2 с помощью команды ping с параметром -c 6 (рис. 4.4).

```

X "host: h1"@mininet-vm
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 1036 bytes 270956 (270.9 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 1036 bytes 270956 (270.9 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# 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=4.50 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.466 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.092 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.092 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.101 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.109 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5096ms
rtt min/avg/max/mdev = 0.092/0.893/4.498/1.617 ms
root@mininet-vm:/home/mininet# 

X "host: h2"@mininet-vm
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 1406 bytes 307892 (307.8 KB)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets 1406 bytes 307892 (307.8 KB)
        TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# 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=3.27 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.085 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.122 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.085 ms
64 bytes from 10.0.0.1: icmp_seq=5 ttl=64 time=0.113 ms
64 bytes from 10.0.0.1: icmp_seq=6 ttl=64 time=0.114 ms

--- 10.0.0.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5078ms
rtt min/avg/max/mdev = 0.085/0.630/3.265/1.178 ms
root@mininet-vm:/home/mininet# 

```

Рис. 4.4: Проверка подключения между хостами

Добавление/изменение задержки в эмулируемой глобальной сети

На хосте h1 добавим задержку в 100 мс к выходному интерфейсу (рис. 4.5).

```
sudo tc qdisc add dev h1-eth0 root netem delay 100ms
```

- 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;
- delay 100ms: задержка ввода 100 мс.

Проверим, что соединение от хоста h1 к хосту h2 имеет задержку 100 мс, используя команду ping с параметром -c 6 с хоста h1

```
X "host h1"@mininet-vm
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 100ms
root@mininet-vm:/home/mininet# 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=102 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=102 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=101 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=101 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=101 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=100 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5009ms
rtt min/avg/max/mdev = 100.473/101.079/101.771/0.494 ms
root@mininet-vm:/home/mininet# ■
```

Рис. 4.5: Добавление задержки в 100мс

Для эмуляции глобальной сети с двунаправленной задержкой необходимо к соответствующему интерфейсу на хосте h2 также добавим задержку в 100 миллисекунд (рис. 4.6).

Проверим, что соединение между хостом h1 и хостом h2 имеет RTT в 200 мс (100 мс от хоста h1 к хосту h2 и 100 мс от хоста h2 к хосту h1), повторив команду ping с параметром -c 6 на терминале хоста h1.

The image shows two terminal windows side-by-side. The left window, titled "host h1" @ mininet-vm, displays the output of a ping command from host h1 to host h2. The right window, titled "host h2" @ mininet-vm, displays the configuration of a network discipline (qdisc) on host h2.

```
root@mininet-vm:/home/mininet# 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=202 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=201 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=201 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=201 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=201 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=202 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5009ms
rtt min/avg/max/mdev = 200.758/201.321/202.437/0.561 ms
root@mininet-vm:/home/mininet# █

root@mininet-vm:/home/mininet# sudo tc qdisc add dev h2-eth0 root netem delay 100ms
root@mininet-vm:/home/mininet# █
```

Рис. 4.6: Двунаправленная задержка соединения

Изменение задержки в эмулируемой глобальной сети

Изменим задержку со 100 мс до 50 мс для отправителя h1 и для получателя h2 (рис. 4.7).

Проверим, что соединение от хоста h1 к хосту h2 имеет задержку 100 мс, используя команду ping с параметром -c 6 с терминала хоста h1.

```

root@mininet-vm:/home/mininet# sudo tc qdisc change dev h1-eth0 root netem delay 50
ms
root@mininet-vm:/home/mininet# 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=110 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=105 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=102 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=101 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=101 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=101 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5010ms
rtt min/avg/max/mdev = 101.185/103.368/109.648/3.065 ms
root@mininet-vm:/home/mininet# █

root@mininet-vm:/home/mininet# sudo tc qdisc change dev h2-eth0 root netem delay 50
ms
root@mininet-vm:/home/mininet# █

```

Рис. 4.7: Изменение задержки на 50мс

Восстановление исходных значений (удаление правил) задержки в эмулируемой глобальной сети

Восстановим конфигурацию по умолчанию, удалив все правила, применённые к сетевому планировщику соответствующего интерфейса для отправителя h1 и для получателя h2. Проверим, что соединение между хостом h1 и хостом h2 не имеет явно установленной задержки, используя команду ping с параметром -c 6 с терминала хоста h1 (рис. 4.8).

```

X "host h1"@mininet-vm
root@mininet-vm:/home/mininet# sudo tc qdisc del dev h1-eth0 root netem
root@mininet-vm:/home/mininet# 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=2.40 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.717 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.288 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.090 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.170 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.083 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5095ms
rtt min/avg/max/mdev = 0.083/0.624/2.397/0.821 ms
root@mininet-vm:/home/mininet# █

X "host h2"@mininet-vm
root@mininet-vm:/home/mininet# sudo tc qdisc del dev h2-eth0 root netem
root@mininet-vm:/home/mininet# █

```

Рис. 4.8: Восстановление исходных значений задержки

Добавление значения дрожания задержки в интерфейс подключения к эмулируемой глобальной сети

Добавим на узле h1 задержку в 100 мс со случайным отклонением 10 мс. Проверим, что соединение от хоста h1 к хосту h2 имеет задержку 100 мс со случайным отклонением ± 10 мс, используя в терминале хоста h1 команду ping с параметром -с 6. Восстановим конфигурацию интерфейса по умолчанию на узле h1 (рис. 4.9).

```

X "host h1"@mininet-vm
root@mininet-vm:/home/mininet# sudo tc qdisc del dev h1-eth0 root netem
Error: Invalid qdisc name.
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 100ms
10ms
root@mininet-vm:/home/mininet# 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=111 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=101 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=97.7 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=94.7 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=107 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=101 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5012ms
rtt min/avg/max/mdev = 94.729/102.178/111.134/5.498 ms
root@mininet-vm:/home/mininet# █

```

Рис. 4.9: Добавление значения дрожания задержки в интерфейс подключения

Добавление значения корреляции для джиттера и задержки в интерфейс подключения к эмулируемой глобальной сети

Добавим на интерфейсе хоста h1 задержку в 100 мс с вариацией ± 10 мс и значением корреляции в 25%. Убедимся, что все пакеты, покидающие устройство h1 на интерфейсе h1-eth0, будут иметь время задержки 100 мс со случайным отклонением ± 10 мс, при этом время передачи следующего пакета зависит от предыдущего значения на 25%. Используем для этого в терминале хоста h1 команду ping с параметром -c 20. Восстановим конфигурацию интерфейса по умолчанию на узле h1 (рис. 4.10).

```
root@mininet-vm:/home/mininet# sudo tc qdisc del dev h1-eth0 root netem
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 100ms
10ms 25%
root@mininet-vm:/home/mininet# 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=103 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=103 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=105 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=105 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=106 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=108 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5009ms
rtt min/avg/max/mdev = 102.915/105.032/108.211/1.788 ms
root@mininet-vm:/home/mininet# █
```

Рис. 4.10: Добавление значения корреляции для джиттера и задержки в интерфейсе подключения

Распределение задержки в интерфейсе подключения к эмулируемой глобальной сети

Зададим нормальное распределение задержки на узле h1 в эмулируемой сети. Убедимся, что все пакеты, покидающие хост h1 на интерфейсе h1-eth0, будут иметь время задержки, которое распределено в диапазоне 100 мс ± 20 мс. Используем для этого команду ping на терминале хоста h1 с параметром -c 10. Восстановим конфигурацию интерфейса по умолчанию на узле h1. Завершим работу mininet в интерактивном режиме (рис. 4.11).

```

root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 100ms
 20ms distribution normal
root@mininet-vm:/home/mininet# 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=100 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=77.0 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=81.4 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=77.2 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=102 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=100 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5011ms
rtt min/avg/max/mdev = 77.024/89.730/102.369/11.292 ms
root@mininet-vm:/home/mininet# ping -c 10 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=92.6 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=101 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=92.9 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=87.6 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=90.7 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=94.1 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=123 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=73.4 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=81.4 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=85.8 ms

--- 10.0.0.2 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9012ms
rtt min/avg/max/mdev = 73.363/92.244/123.105/12.544 ms
root@mininet-vm:/home/mininet#

```

Рис. 4.11: Распределение задержки в интерфейсе подключения

Обновим репозитории программного обеспечения на виртуальной машине.

Установим пакет geeqie для просмотра файлов png. Для каждого воспроизводимого эксперимента expname создадим свой каталог, в котором будут размещаться файлы эксперимента (рис. 4.12).

```

mininet@mininet-vm:~$ sudo apt-get update
Hit:1 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:2 http://us.archive.ubuntu.com/ubuntu focal InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease
Reading package lists... Done
mininet@mininet-vm:~$ sudo apt install geeqie
Reading package lists... Done
Building dependency tree
Reading state information... Done
geeqie is already the newest version (1:1.5.1-8build1).
0 upgraded, 0 newly installed, 0 to remove and 362 not upgraded.
mininet@mininet-vm:~$ mkdir -p ~/work/lab_netem_i/expname
mininet@mininet-vm:~$ cd work/lab_netem_i/
mininet@mininet-vm:~/work/lab_netem_i$ 

```

Рис. 4.12: Подготовка к производимому эксперименту

Создадим скрипт для эксперимента lab_netem_i.py (рис. 4.13).

```

mininet@mininet-vm: ~/work, + - X
GNU nano 4.8          lab_netem_i.py          Modified
#!/usr/bin/env python

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

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

def emptyNet():
    "Create an empty network and add nodes to it."
    net = Mininet(controller=Controller, waitConnected=True)
    info('*** Adding controller\n')
    net.addController('c0')
    info('*** Adding hosts\n')
    h1 = net.addHost('h1', ip='10.0.0.1')
    h2 = net.addHost('h2', ip='10.0.0.2')
    info('*** Adding switch\n')
    s1 = net.addSwitch('s1')
    info('*** Creating links\n')
    net.addLink(h1, s1)
    net.addLink(h2, s1)
    info('*** Starting network\n')
    net.start()
    info('*** Set delay\n')
    h1.cmdPrint('tc qdisc add dev h1-eth0 root netem delay 100ms')
    h2.cmdPrint('tc qdisc add dev h2-eth0 root netem delay 100ms')
    time.sleep(10) # Wait 10 seconds
    info('*** Ping\n')
    h1.cmdPrint('ping -c 100', h2.IP(),
               '| grep "time=" | awk \'print $5, $7\' | sed -e \'s/time=/g\' -e \'> ping.dat')
    info('*** Stopping network')
    net.stop()

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

```

Рис. 4.13: Листинг lab_netem_i.py

Затем создадим скрипт для визуализации ping_plot результатов эксперимента(рис. 4.14).

```

mininet@mininet-vm: ~/work, + - X
GNU nano 4.8          ping_plot          Modified
#!/usr/bin/gnuplot --persist

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

```

Рис. 4.14: Листинг ping_plot

Зададим права доступа к файлу скрипта(рис. 4.15).

```

mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano ping_plot
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ chmod +x ping_plot
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ 

```

Рис. 4.15: Выдача прав выполнения для ping_plot

Создадим файла Makefile. Внутри файла Makefile поместим скрипт для управления процессом проведения эксперимента.(рис. 4.16).

```

GNU nano 4.8                               Makefile
all: ping.dat ping.png

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

ping.png: ping.dat
    ./ping_plot

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

```

Рис. 4.16: Листинг Makefile

Выполним эксперимент(рис. 4.17).

```

mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make
sudo python lab_netem_i.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 delay
*** h1 : ('tc qdisc add dev h1-eth0 root netem delay 100ms',)
*** h2 : ('tc qdisc add dev h2-eth0 root netem delay 100ms',)
*** Ping
*** 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 2 links
...
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot

```

Рис. 4.17: Запуск эксперимента

Просмотрим построенный в результате выполнения скриптов график(рис. 4.18).

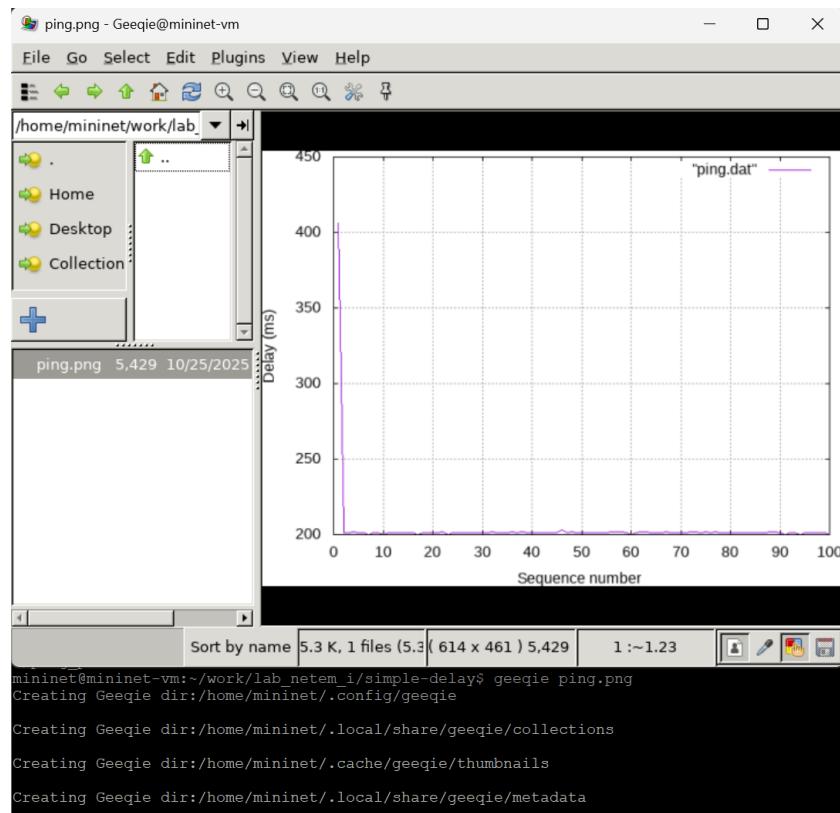


Рис. 4.18: Просмотр графика

Из файла ping.dat удалим первую строку и заново построим график(рис. 4.19).

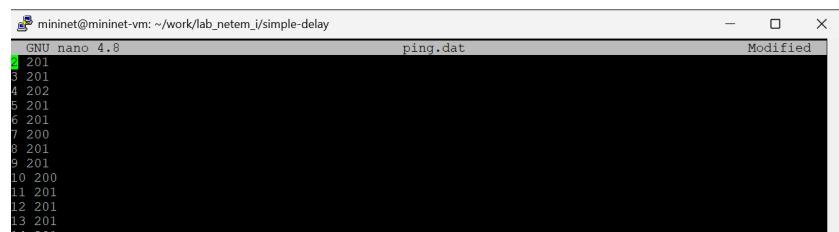


Рис. 4.19: Удаление первой строчки из файла ping.dat

Просмотрим заново построенный график (рис. 4.20).

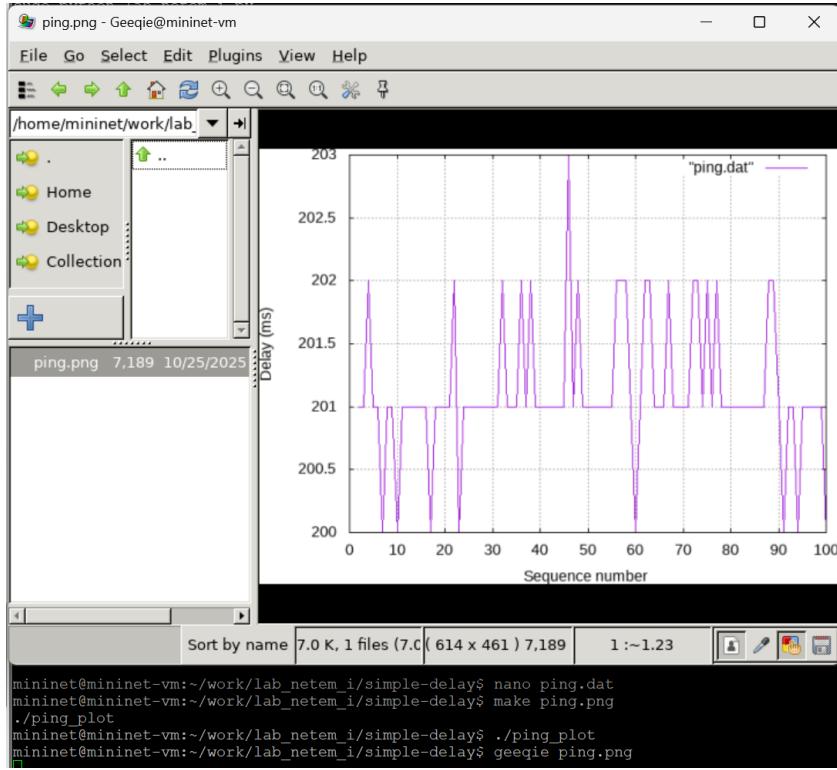


Рис. 4.20: Просмотр графика

Разработаем скрипт для вычисления на основе данных файла ping.dat минимального, среднего, максимального и стандартного отклонения времени приёма-передачи. Протестируем его. Также добавим правило запуска скрипта в Makefile (рис. 4.21; 4.22; 4.23;).

```

mininet@mininet-vm: ~/work/lab_neterm_i/simple-delay
GNU nano 4.8                                         analitic.py
with open('ping.dat', 'r') as file:
    s = []
    for line in file.readlines():
        if '\n' in line:
            line.replace('\n', '')
        if len(line.strip()):
            s.append(int(line.split(" ")[1]))
    std = ((sum([(i - (sum(s) / len(s))) ** 2 for i in s]) / (len(s) - 1)) ** 0.5)
    avg_z = sum(s) / len(s)
    max_z = max(s)
    min_z = min(s)
    print(f"min: {min_z} \n max: {max_z} \n avg: {avg_z} \n std: {std}")

```

Рис. 4.21: Разработка скрипта для вычисления на основе данных файла ping.dat статистических данных

```
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ sudo python analitic.py
min: 200
max: 203
avg: 201.12121212121212
std: 0.539703759874809
```

Рис. 4.22: Тестирование разработанного скрипта

```
GNU nano 4.8
Makefile
all: ping.dat ping.png analitic
ping.dat:
    sudo python lab_netem_i.py
    sudo chown mininet:mininet ping.dat
ping.png: ping.dat
    ./ping_plot
analitic:
    sudo python analitic.py
clean:
    -rm -f *.dat *.png
```

Рис. 4.23: Добавление правила запуска скрипта в Makefile

Очистим каталог от результатов проведения экспериментов.(рис. 4.24).

```
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make clean
rm -f *.dat *.png
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$
```

Рис. 4.24: Очистка результатов эксперимента

Самостоятельно реализуем воспроизводимые эксперименты по изменению задержки, джиттера, значения корреляции для джиттера и задержки, распределения времени задержки в эмулируемой глобальной сети. Построим графики. Вычислим минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи для каждого случая (рис. 4.25 - рис. 4.36):

```

GNU nano 4.0                               lab netem i.py
GNU nano 4.0                               lab netem i.py
!#/usr/bin/env python
"""

Simple experiment.
Output: ping.dat

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

def emptyNet():
    "Create an empty network and add nodes to it."
    net = Mininet(controller=Controller, waitConnected=True)
    info("**** Adding controller\n")
    net.addController('c0')
    info("**** Adding hosts\n")
    h1 = net.addHost('h1', ip='10.0.0.1')
    h2 = net.addHost('h2', ip='10.0.0.2')
    info("**** Adding switch\n")
    s1 = net.addSwitch('s1')
    info("**** Creating links\n")
    net.addLink(h1, s1)
    net.addLink(h2, s1)
    info("**** Starting network\n")
    net.start()
    info("**** Set delay\n")
    h1.cmdPrint('tc qdisc add dev h1-eth0 root netem delay 50ms')
    h2.cmdPrint('tc qdisc add dev h2-eth0 root netem delay 50ms')
    time.sleep(10) # Wait 10 seconds
    info("**** Ping\n")
    h1.cmdPrint('ping -c 100', h2.IP(),
                '| grep "time=" | awk \'(print $5, $7)\' | sed -e \'s/time=/g\' -e \'s/icmp_seq=/g\'')
    info("**** Stopping network")
    net.stop()

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

```

Рис. 4.25: ЛИСТИНГ|Воспроизводимый эксперимент по изменению задержки

```

mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano lab_netem_i.py
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make
udo python lab_netem_i.py
** Adding controller
** Adding hosts
** Adding switch
** Creating links
** Starting network
** Starting network
** Configuring hosts
1 h2
** Starting controller
0
** Starting 1 switches
1 ...
** Waiting for switches to connect
1
** Set delay
** h1 : ('tc qdisc add dev h1-eth0 root netem delay 50ms',)
** h2 : ('tc qdisc add dev h2-eth0 root netem delay 50ms')
** Ping
** h1 : ('ping -c 100', '10.0.0.2', '| grep "time=" | awk \'(print $5, $7)\' | sed -e \'s/time=/g\' -e \'s/icmp_seq=/g\' > ping.dat')
** Stopping network** Stopping 1 controllers
0
** Stopping 2 links
.
** Stopping 1 switches
1
** Stopping 2 hosts
1 h2
** Done
udo chown mininet:mininet ping.dat
/ping plot
udo python analitic.py
in: 100
max: 209
avg: 102.52
std: 10.780153856588376
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ 

```

Рис. 4.26: ВЫВОД|Воспроизводимый эксперимент по изменению задержки

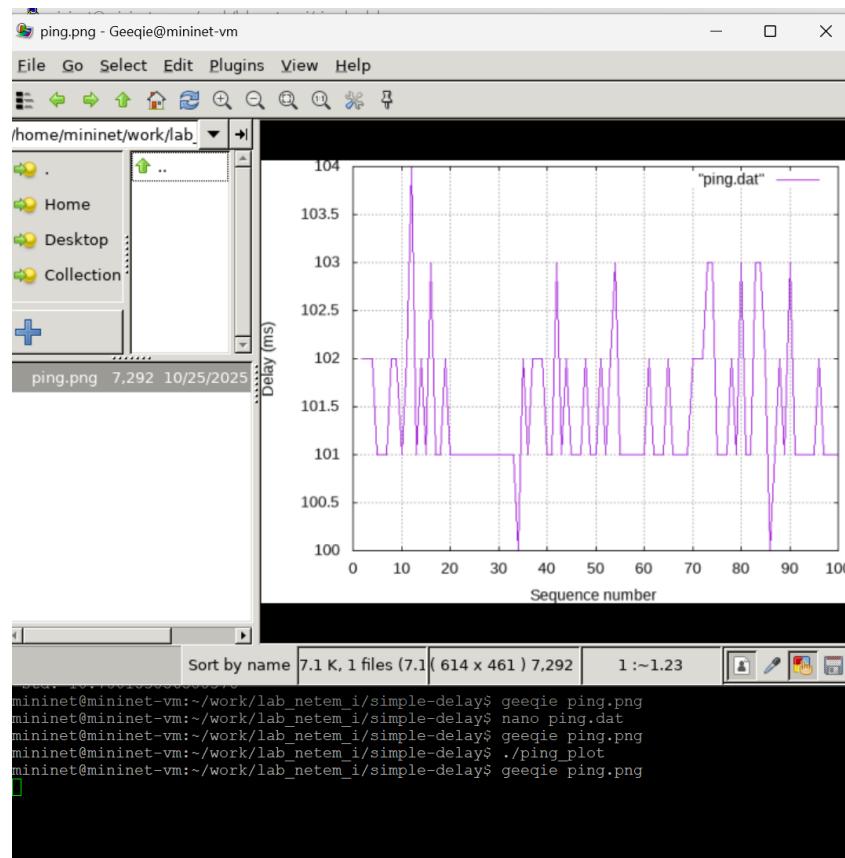


Рис. 4.27: ГРАФИК|Воспроизводимый эксперимент по изменению задержки

```

mininet@mininet-vm: ~/work/lab_netem_i/simple-delay$ nano lab_netem_i.py
GNU nano 4.8                               lab_netem_i.py                                Modified
#!/usr/bin/env python

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

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

def emptyNet():
    """
    Create an empty network and add nodes to it.
    net = Mininet(controller=Controller, waitConnected=True)
    info('*** Adding controller\n')
    net.addController('c0')
    info('*** Adding hosts\n')
    h1 = net.addHost('h1', ip='10.0.0.1')
    h2 = net.addHost('h2', ip='10.0.0.2')
    info('*** Adding switch\n')
    s1 = net.addSwitch('s1')
    info('*** Creating links\n')
    net.addLink(h1, s1)
    net.addLink(h2, s1)
    info('*** Starting network\n')
    net.start()
    info('*** Set delay\n')
    h1.cmdPrint('tc qdisc add dev h1-eth0 root netem delay 100ms 10ms')
    h2.cmdPrint('tc qdisc add dev h2-eth0 root netem delay 10ms')
    time.sleep(10) # Wait 10 seconds
    info('*** Ping\n')
    h1.cmdPrint('ping -c 100', h2.IP(),
               '| grep "time=' | awk \'(print $5, $7)\' | sed -e \'s/time=/g\' -e \'s/icmp_seq=/g\'')
    info('*** Stopping network')
    net.stop()

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

```

Рис. 4.28: ЛИСТИНГ|Воспроизводимый эксперимент по изменению джиттера

```

mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make clean
rm -f *.dat *.png
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano lab_netem_i.py
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make
sudo python lab_netem_i.py
*** Adding controller
*** Adding hosts
*** Adding switch
*** Creating switch
*** Starting network
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Waiting for switches to connect
s1 ...
*** Set delay
*** h1 : ('tc qdisc add dev h1-eth0 root netem delay 100ms 10ms',)
*** h2 : ('tc qdisc add dev h2-eth0 root netem delay 100ms',)
*** Ping
*** h1 : ('ping -c 100', '10.0.0.2', '| grep "time=' | awk \'(print $5, $7)\' | sed -e \'s/time=/g\' -e \'s/icmp_seq=/g\' > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot
sudo python analitic.py
min: 191
max: 406
avg: 203.76
std: 21.194205421160365
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ 

```

Рис. 4.29: ВЫВОД|Воспроизводимый эксперимент по изменению джиттера

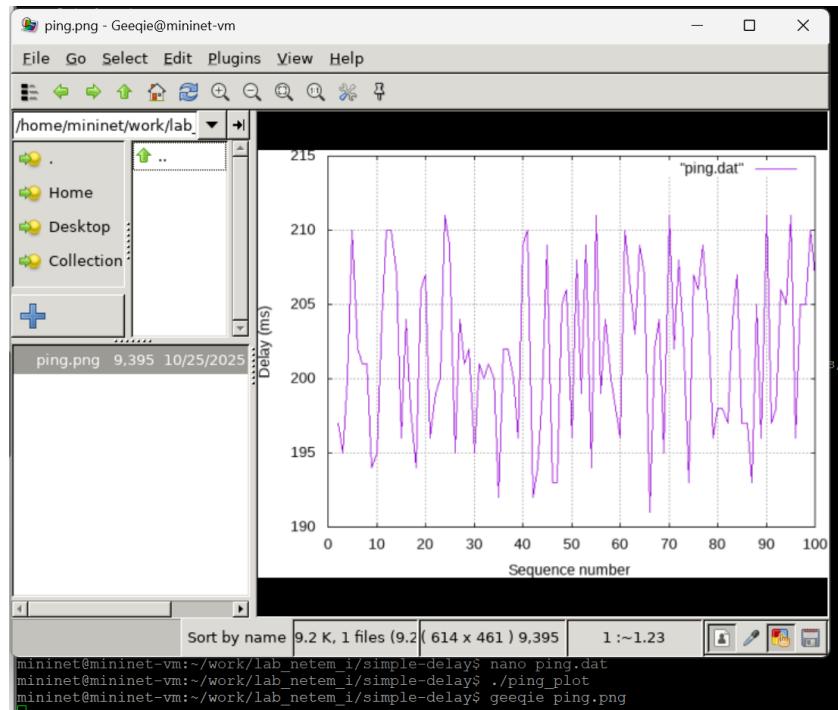


Рис. 4.30: ГРАФИК|Воспроизводимый эксперимент по изменению джиттера

```
GNU nano 4.8
#!/usr/bin/env python

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

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

def emptyNet():
    "Create an empty network and add nodes to it."
    net = Mininet(controller=Controller, waitConnected=True)
    info('*** Adding controller\n')
    net.addController('c0')
    info('*** Adding hosts\n')
    h1 = net.addHost('h1', ip='10.0.0.1')
    h2 = net.addHost('h2', ip='10.0.0.2')
    info('*** Adding switch\n')
    s1 = net.addSwitch('s1')
    info('*** Creating links\n')
    net.addLink(h1, s1)
    net.addLink(h2, s1)
    info('*** Starting network\n')
    net.start()
    info('*** Set delay\n')
    h1.cmdPrint('tc qdisc add dev h1-eth0 root netem delay 100ms 10ms 25%')
    h2.cmdPrint('tc qdisc add dev h2-eth0 root netem delay 100ms')
    time.sleep(10) # Wait 10 seconds
    info('*** Ping\n')
    h1.cmdPrint('ping -c 100', h2.IPs(),
               '| grep "times" | awk \'(print $5, $7)\' | sed -e \'s/time=/g\' -e \'s/icmp_seq=/g\'')
    info('*** Stopping network')
    net.stop()

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

Рис. 4.31: ЛИСТИНГ|Воспроизводимый эксперимент по изменению значения корреляции для джиттера и задержки

```

mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano lab_netem_i.py
sudo python lab_netem_i.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 delay
*** h1 : ('tc qdisc add dev h1-eth0 root netem delay 100ms 10ms 25%',)
*** h2 : ('tc qdisc add dev h2-eth0 root netem delay 100ms',)
*** Ping
*** 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 2 links
...
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot
sudo python analitic.py
min: 193
max: 403
avg: 202.98
std: 20.96075216477529

```

Рис. 4.32: ВЫВОД|Воспроизводимый эксперимент по изменению значения корреляции для джиттера и задержки

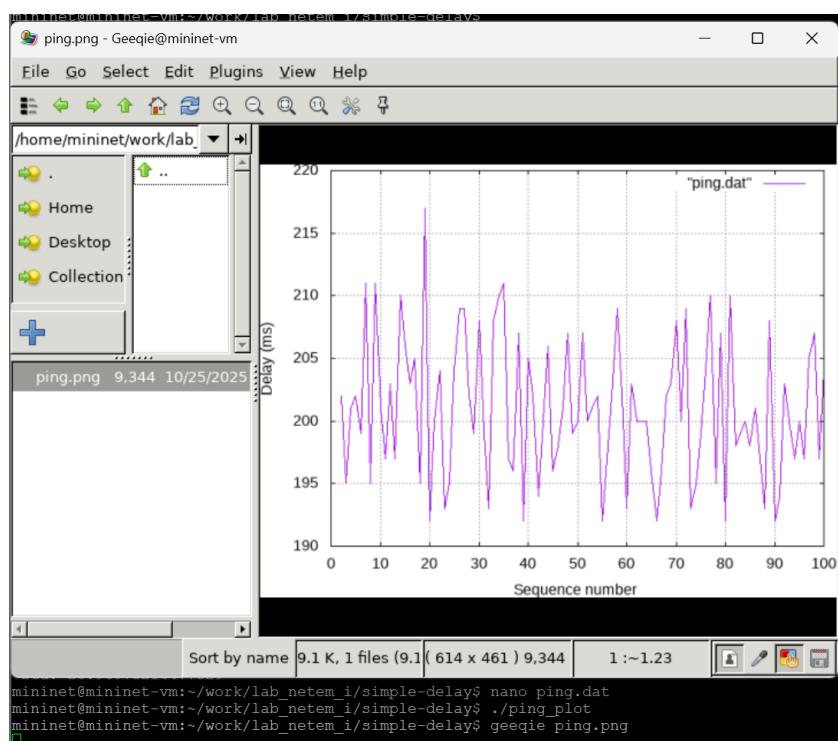


Рис. 4.33: ГРАФИК|Воспроизводимый эксперимент по изменению значения корреляции для джиттера и задержки

```

mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ sudo python lab_netem_i.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 delay
*** h1 : ('tc qdisc add dev h1-eth0 root netem delay 100ms 10ms 25% distribution normal',)
*** h2 : ('tc qdisc add dev h2-eth0 root netem delay 100ms',)
*** Ping
*** 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 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot
sudo python analitic.py
min: 180
max: 417
avg: 204.21
std: 23.08845310250704

```

Рис. 4.34: ЛИСТИНГ|Воспроизводимый эксперимент по изменению распределения времени задержки в эмулируемой глобальной сети

Рис. 4.35: ВЫВОД|Воспроизводимый эксперимент по изменению распределения времени задержки в эмулируемой глобальной сети

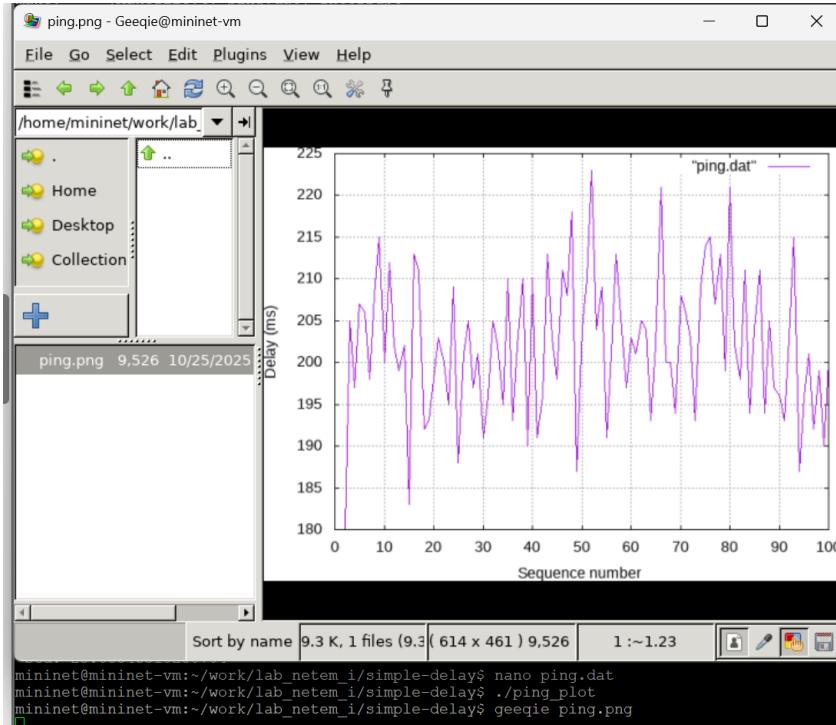


Рис. 4.36: ГРАФИК|Воспроизводимый эксперимент по изменению распределения времени задержки в эмулируемой глобальной сети

Так как все эти эксперименты желательно по заданию распределить по папкам, то сделаем копию где мы делали эксперимент и перенесём известные нам уже программы.(рис. 4.37).

```
mininet@mininet-vm:~/work/lab_neterm_i/simple-delay$ geeqie ping.png
mininet@mininet-vm:~/work/lab_neterm_i/simple-delay$ cd ..
mininet@mininet-vm:~/work/lab_neterm$ cp -R simple-delay correlation-delay
mininet@mininet-vm:~/work/lab_neterm$ cp -R simple-delay jitter-delay
mininet@mininet-vm:~/work/lab_neterm$ cp -R simple-delay change-delay
mininet@mininet-vm:~/work/lab_neterm$ cp -R simple-delay change-jitter
mininet@mininet-vm:~/work/lab_neterm$ cp -R simple-delay change-jitter-delay
mininet@mininet-vm:~/work/lab_neterm$
```

Рис. 4.37: Создание папок под эксперименты

5 Выводы

В результате выполнения данной лабораторной работы я познакомился с NETEM – инструментом для тестирования производительности приложений в виртуальной сети, а также получил навыки проведения интерактивного и воспроизводимого экспериментов по измерению задержки и её дрожания (jitter) в моделируемой сети в среде Mininet.

Список литературы

1. Mininet [Электронный ресурс]. Mininet Project Contributors. URL: <https://mininet.org/> (дата обращения: 07.10.2025).