

Лабораторная работа №4

Эмуляция и измерение задержек в глобальных сетях

Тазаева Анастасия Анатольевна

Содержание

1	Цель работы	6
2	Задание	7
3	Выполнение лабораторной работы	8
3.1	Запуск лабораторной топологии	8
3.2	Интерактивные эксперименты	10
3.2.1	Добавление/изменение задержки в эмулируемой глобальной сети	10
3.2.2	Изменение задержки в эмулируемой глобальной сети	12
3.2.3	Добавление значения дрожания задержки в интерфейс подключения к эмулируемой глобальной сети	14
3.2.4	Добавление значения корреляции для джиттера и задержки в интерфейс подключения к эмулируемой глобальной сети	15
3.2.5	Распределение задержки в интерфейсе подключения к эмулируемой глобальной сети	16
3.3	Воспроизведение экспериментов	17
3.3.1	Предварительная подготовка	17
3.3.2	Добавление задержки для интерфейса, подключающегося к эмулируемой глобальной сети	18
4	Выводы	24

Список иллюстраций

3.1	Подключение к mininet	8
3.2	Исправление прав запуска X-соединения	8
3.3	Создание топологии	9
3.4	ifconfig h1	9
3.5	ifconfig h2	9
3.6	Проверка подключения ping h1 -> h2	10
3.7	Задержка для хоста h1	10
3.8	Задержка для хоста h1. Проверка	11
3.9	Задержка для хоста h2	11
3.10	Задержка для хоста h1(+задержка от хоста h2. Проверка	12
3.11	Изменение задержки для хоста h1	12
3.12	Изменение задержки для хоста h2	12
3.13	Задержка для хоста h1 после изменения задержки. Проверка	13
3.14	Удаление задержки для хоста h1	13
3.15	Удаление задержки для хоста h2	13
3.16	Задержка для хоста h1 после удаления задержки. Проверка	14
3.17	Установка задержки для хоста h1 с случайным отклонением	14
3.18	Задержка для хоста h1 с случайным отклонением. Проверка	14
3.19	Удаление задержки для хоста h1	15
3.20	Установка задержки для хоста h1 с значением корреляции	15
3.21	Задержка для хоста h1 с значением корреляции. Проверка	16
3.22	Установка задержки для хоста h1 и задание нормального распределения	16
3.23	Задержка для хоста h1 с нормальным распределением. Проверка .	17
3.24	Обновление репозитория	17
3.25	Установка пакета geeqie	18
3.26	Создание каталога для размещения файлов эксперимента	18
3.27	Создание каталога imple-delay	19
3.28	lab_netem_i.py	19
3.29	lab_netem_i.py. Продолжение	20
3.30	ping_plot	20
3.31	Изменение прав для файла скрипта	20
3.32	Makefile	21
3.33	Make	21
3.34	График с большой разницей первого значения с последующими .	22
3.35	График	22
3.36	samost.py	23

3.37 Makefile. измененный	23
3.38 samost.py выполнение	23

Список таблиц

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 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.

3 Выполнение лабораторной работы

3.1 Запуск лабораторной топологии

1. Запустила виртуальную среду с mininet. Из основной ОС подключилась к виртуальной машине (рис. 3.1).

```
[aatazaeva@aatazaeva ~]$ ssh -Y mininet@192.168.56.101
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-42-generic x86_64)

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

New release '22.04.5 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sat Nov 30 00:59:49 2024
```

Рис. 3.1: Подключение к mininet

2. Исправила права запуска X-соединения (рис. 3.2).

```
mininet@mininet-vm:~$ xauth list $DISPLAY
mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 354e0b0b8205dd1dad6e57cd32216f64
mininet@mininet-vm:~$ sudo -i
root@mininet-vm:~# xauth list $DISPLAY
mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 b12887d838599f0b89841942b78a6ed6
root@mininet-vm:~# xauth add mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 354e0b0b8205dd1dad6e57cd32216f64
root@mininet-vm:~# xauth list $DISPLAY
mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 354e0b0b8205dd1dad6e57cd32216f64
root@mininet-vm:~# exit
logout
```

Рис. 3.2: Исправление прав запуска X-соединения

3. Создала простейшую топологию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8 (рис. 3.3). Терминалы коммутатора и контроллера закрыла.


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

```
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. 3.3: Создание топологии

4. На хостах h1 и h2 ввела команду `ifconfig` (рис. 3.4 и 3.5), 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`.

```
root@mininet-vm:/home/mininet# ifconfig
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.1 netmask 255.0.0.0 broadcast 10.255.255.255
    ether 7a:02:e8:8a:45:41 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. 3.4: `ifconfig h1`

```
root@mininet-vm:/home/mininet# ifconfig
h2-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.0.2 netmask 255.0.0.0 broadcast 10.255.255.255
    ether 62:a7:c8:a5:a0:36 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. 3.5: `ifconfig h2`

5. Проверила подключение между хостами h1 и h2 с помощью команды ping с параметром -c 6 (рис. 3.6).

```
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=0.529 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.035 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.051 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.042 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.051 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.047 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5103ms
rtt min/avg/max/mdev = 0.035/0.125/0.529/0.180 ms
```

Рис. 3.6: Проверка подключения ping h1 -> h2

6. Минимальное(=0.035 ms), среднее(=0.125 ms), максимальное(=0.529 ms) и стандартное(=0.180 ms) отклонение времени приёма-передачи (RTT).

3.2 Интерактивные эксперименты

3.2.1 Добавление/изменение задержки в эмулируемой глобальной сети

1. На хосте h1 добавила задержку в 100 мс к выходному интерфейсу (рис. 3.7):

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

```
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 100ms
```

Рис. 3.7: Задержка для хоста h1

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; - delay 100ms : задержка ввода 100 мс.

2. Проверила, что соединение от хоста h1 к хосту h2 имеет задержку 100 мс, используя команду ping с параметром -c 6 с хоста h1 . Минимальное(=100.221 ms), среднее(=100.647 ms), максимальное(=101.032 ms) и стандартное(=0.329 ms) отклонение времени приёма-передачи (RTT) (рис. 3.8).

```
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=101 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=100 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=100 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 5010ms
rtt min/avg/max/mdev = 100.221/100.647/101.032/0.329 ms
```

Рис. 3.8: Задержка для хоста h1. Проверка

3. Для эмуляции глобальной сети с двунаправленной задержкой необходимо к соответствующему интерфейсу на хосте h2 также добавить задержку в 100 миллисекунд (рис. 3.9):

```
sudo tc qdisc add dev h2-eth0 root netem delay 100ms
```

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

Рис. 3.9: Задержка для хоста h2

4. Проверила, 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 . Минимальное(=200.238 ms), среднее(=200.904 ms), максимальное(=201.980 ms) и стандартное(=0.632 ms) отклонение времени приёма-передачи (RTT) (рис. 3.10).

```

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=201 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=202 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=200 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=200 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=200 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5007ms
rtt min/avg/max/mdev = 200.238/200.904/201.980/0.632 ms

```

Рис. 3.10: Задержка для хоста h1(+задержка от хоста h2. Проверка

3.2.2 Изменение задержки в эмулируемой глобальной сети

1. Изменила задержку со 100 мс до 50 мс для отправителя h1 (рис. 3.11):

```
sudo tc qdisc change dev h1-eth0 root netem delay 50ms
```

и для получателя h2 (рис. 3.12):

```
sudo tc qdisc change dev h2-eth0 root netem delay 50ms
```

```

root@mininet-vm:/home/mininet# sudo tc qdisc change dev h1-eth0 root netem dela
y 50ms

```

Рис. 3.11: Изменение задержки для хоста h1

```

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

```

Рис. 3.12: Изменение задержки для хоста h2

2. Проверила, что соединение от хоста h1 к хосту h2 имеет задержку 100 мс, используя команду ping с параметром -c 6 с терминала хоста h1 . На (рис. 3.13) выделила минимальное, среднее, макси-мальное и стандартное отклонение времени приёма-передачи (RTT).

```

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=101 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=102 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=102 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 5001ms
rtt min/avg/max/mdev = 100.880/101.279/101.860/0.323 ms

```

Рис. 3.13: Задержка для хоста h1 после изменения задержки. Проверка

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

1. Восстановила конфигурацию по умолчанию, удалив все правила, применённые к сетевому планировщику соответствующего интерфейса. Для отправителя h1 (рис. 3.14):

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

Для получателя h2 (рис. 3.15):

```
sudo tc qdisc del dev h2-eth0 root netem
```

```

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

```

Рис. 3.14: Удаление задержки для хоста h1

```

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

```

Рис. 3.15: Удаление задержки для хоста h2

2. Проверила, что соединение между хостом h1 и хостом h2 не имеет явноустановленной задержки, используя команду ping с параметром -c 6 с терминала хоста h1. На (рис. 3.16) выделила минимальное, среднее, макси-мальное и стандартное отклонение времени приёма-передачи (RTT).

```

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=101 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=102 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=102 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 5001ms
rtt min/avg/max/mdev = 100.880/101.279/101.860/0.323 ms

```

Рис. 3.16: Задержка для хоста h1 после удаления задержки. Проверка

3.2.3 Добавление значения дрожания задержки в интерфейс подключения к эмулируемой глобальной сети

1. Добавила на узле h1 задержку в 100 мс со случайным отклонением 10 мс (рис. 3.17):

```

sudo tc qdisc add dev h1-eth0 root netem delay 100ms 10ms

```

```

root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 100ms 10ms

```

Рис. 3.17: Установка задержки для хоста h1 с случайным отклонением

2. Проверила, 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 -c 6. На (рис. 3.18) видно минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи (RTT).

```

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=94.8 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=96.5 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=103 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=102 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=94.3 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5004ms
rtt min/avg/max/mdev = 94.316/97.667/103.460/3.769 ms

```

Рис. 3.18: Задержка для хоста h1 с случайным отклонением. Проверка

3. Восстановила конфигурацию интерфейса по умолчанию на узле h1 (рис. 3.19).

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

Рис. 3.19: Удаление задержки для хоста h1

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

1. Добавила на интерфейсе хоста h1 задержку в 100 мс с вариацией ± 10 мс и значением корреляции в 25% (рис. 3.20):

```
sudo tc qdisc add dev h1-eth0 root netem delay 100ms 10ms 25%
```

```
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 100ms 10ms 25%
```

Рис. 3.20: Установка задержки для хоста h1 с значением корреляции

2. Убедилась, 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%. Используйте для этого в терминале хоста h1 команду `ping` с параметром `-c 20`. На (рис. 3.21) видно минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи (RTT).

```

root@mininet-vm:/home/mininet# 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=99.9 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=91.0 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=106 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=103 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=110 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=100 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=101 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=93.0 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=96.7 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=91.1 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=95.2 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=99.5 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=108 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=109 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=105 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=107 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=99.9 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=91.5 ms

--- 10.0.0.2 ping statistics ---
20 packets transmitted, 20 received, 0% packet loss, time 19031ms
rtt min/avg/max/mdev = 91.013/100.438/110.377/5.811 ms

```

Рис. 3.21: Задержка для хоста h1 с значением корреляции. Проверка

3. Восстановила конфигурацию интерфейса по умолчанию на узле h1.

3.2.5 Распределение задержки в интерфейсе подключения к эмулируемой глобальной сети

NETEM позволяет пользователю указать распределение, которое описывает, как задержки изменяются в сети. В реальных сетях передачи данных задержки неравномерны, поэтому при моделировании может быть удобно использовать некоторое случайное распределение, например, нормальное, парето или парето-нормальное.

1. Задала нормальное распределение задержки на узле h1 в эмулируемой сети (рис. 3.22):

```
sudo tc qdisc add dev h1-eth0 root netem delay 100ms 20ms distribution normal
```

```

root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 1
00ms 20ms distribution normal

```

Рис. 3.22: Установка задержки для хоста h1 и задание нормального распределения

2. Убедилась, что все пакеты, покидающие хост h1 на интерфейсе h1-eth0 , будут иметь время задержки, которое распределено в диапазоне 100 мс \pm 20 мс. Используйте для этого команду ping на терминале хоста h1 с параметром -c 10 (рис. 3.23).

```
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=96.4 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=98.7 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=97.1 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=88.5 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=111 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=105 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=111 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=82.2 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=109 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=84.6 ms

--- 10.0.0.2 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 901ms
rtt min/avg/max/mdev = 82.214/98.286/111.169/10.106 ms
```

Рис. 3.23: Задержка для хоста h1 с нормальным распределением. Проверка

3. Восстановила конфигурацию интерфейса по умолчанию на узле h1 и завершила работу mininet в интерактивном режиме.

3.3 Воспроизведение экспериментов

3.3.1 Предварительная подготовка

1. Обновила репозитории программного обеспечения на виртуальной машине (рис. 3.24):

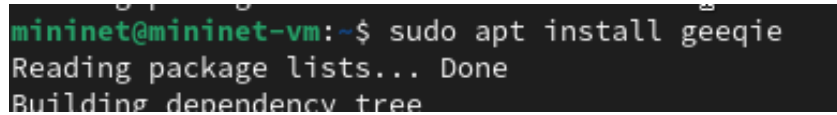
```
sudo apt-get update
```

```
mininet@mininet-vm:~$ sudo apt-get update
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [128 kB]
Get:2 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [3,304 kB]
Hit:3 http://us.archive.ubuntu.com/ubuntu focal InRelease
Get:4 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease [128 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [3,304 kB]
```

Рис. 3.24: Обновление репозитория

2. Установила пакет `geeqie` — понадобится для просмотра файлов `png` (рис. 3.25):

```
sudo apt install geeqie
```

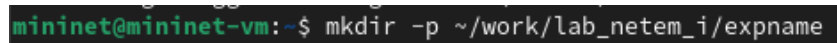


```
mininet@mininet-vm:~$ sudo apt install geeqie
Reading package lists... Done
Building dependency tree
```

Рис. 3.25: Установка пакета `geeqie`

3. Для каждого воспроизводимого эксперимента `exrname` создала свой каталог, в котором будут размещаться файлы эксперимента (рис. 3.26):

```
mkdir -p ~/work/lab_netem_i/exrname
```



```
mininet@mininet-vm:~$ mkdir -p ~/work/lab_netem_i/exrname
```

Рис. 3.26: Создание каталога для размещения файлов эксперимента

3.3.2 Добавление задержки для интерфейса, подключающегося к эмулируемой глобальной сети

С помощью API Mininet воспроизвела эксперимент по добавлению задержки для интерфейса хоста, подключающегося к эмулируемой глобальной сети.

1. В виртуальной среде `mininet` в своём рабочем каталоге с проектами создала каталог `simple-delay` и перешла в него (рис. 3.27):

```
mkdir -p ~/work/lab_netem_i/simple-delay
cd ~/work/lab_netem_i/simple-delay
```

```

mininet@mininet-vm:~$ mkdir -p ~/work/lab_netem_i/expname
mininet@mininet-vm:~$ ls
mininet  mininet.orig  oflops  ofttest  openflow  pox  work
mininet@mininet-vm:~$ cd work/
mininet@mininet-vm:~/work$ ls
lab_iperf3  lab_netem_i
mininet@mininet-vm:~/work$ cd lab_netem_i/
mininet@mininet-vm:~/work/lab_netem_i$ ls
expname
mininet@mininet-vm:~/work/lab_netem_i$ S
S: command not found
mininet@mininet-vm:~/work/lab_netem_i$ mkdir simple-delay
mininet@mininet-vm:~/work/lab_netem_i$ ls
expname  simple-delay
mininet@mininet-vm:~/work/lab_netem_i$ cd simple-delay/
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ touch lab_netem_i.py
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ ls
lab_netem_i.py

```

Рис. 3.27: Создание каталога imple-delay

2. Создала скрипт для эксперимента lab_netem_i.py (рис. 3.28 и 3.29).

```

GNU nano 4.8 lab_netem_i.py
"""
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 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' )

```

Рис. 3.28: lab_netem_i.py

```

h2.cmdPrint( 'tc qdisc add dev h2-eth0 root netem delay 100ms' )

time.sleep(10) # Wait 10 sec

info( '*** Ping \n')
h1.cmdPrint( 'ping -c 100', h2.IP(), '| grep "time=" | awk \'{print $5, $7}\'' | sed -e \'/time=//g\' -e \'/icmp_seq=//g\' > ping.dat' )

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

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

```

Рис. 3.29: lab_netem_i.py. Продолжение

- В каких строках скрипта задается значение задержки для интерфейса хоста?

```
h1.cmdPrint( 'tc qdisc add dev h1-eth0 root netem delay 100ms' )
```

```
h2.cmdPrint( 'tc qdisc add dev h2-eth0 root netem delay 100ms' )
```

- Каким образом формируется файл с результатами эксперимента для последующего построения графиков?

```
h1.cmdPrint( 'ping -c 100', h2.IP(), '| grep "time=" | awk \'{print $5, $7}\'' | sed -e \'/time=//g\' -e \'/icmp_seq=//g\' > ping.dat' )
```

4. Создала скрипт для визуализации ping_plot результатов эксперимента (рис. 3.30).

```

GNU nano 4.8                                     ping_plot
#!/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. 3.30: ping_plot

5. Задала права доступа к файлу скрипта (рис. 3.31).

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

Рис. 3.31: Изменение прав для файла скрипта

6. Создала Makefile для управления процессом проведения эксперимента (рис. 3.32).

```
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. 3.32: Makefile

7. Выполнила эксперимент (рис. 3.33).

make

```
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. 3.33: Make

8. Получился следующий график (рис. 3.34).

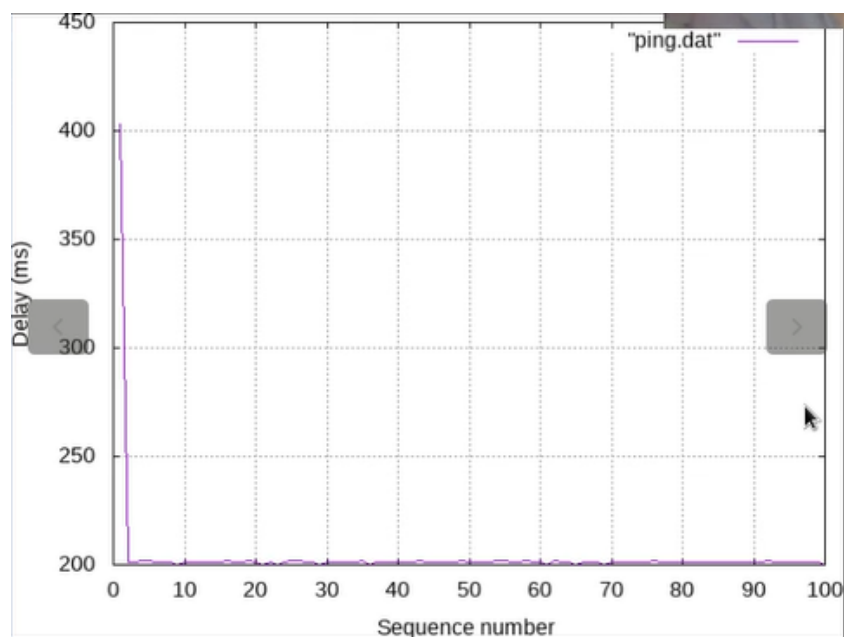


Рис. 3.34: График с большой разницей первого значения с последующими

9. Из файла ping.dat удалила первую строку и заново построила график (рис. 3.35):

make ping.png

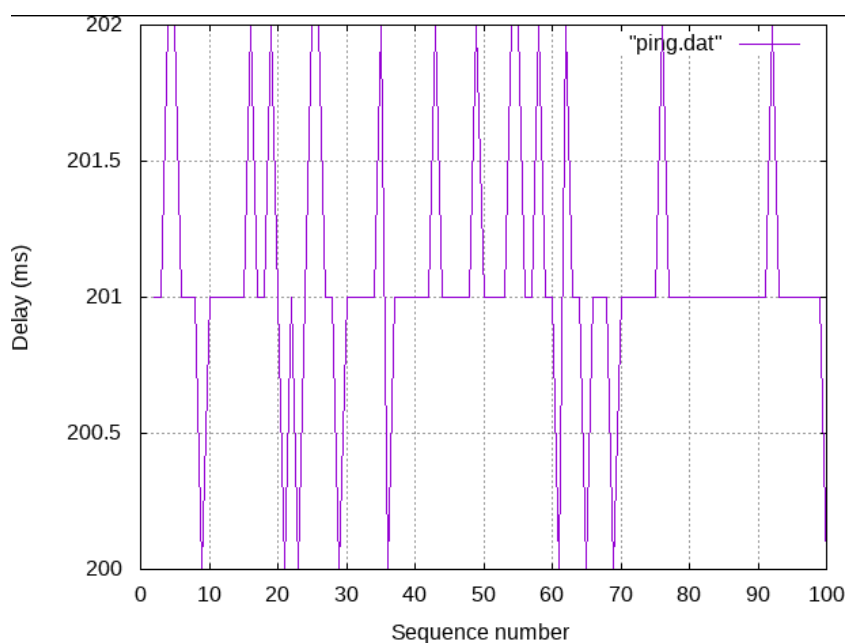


Рис. 3.35: График

10. Разработала скрипт для вычисления на основе данных файла ping.dat минимального, среднего, максимального и стандартного отклонения времени (рис. 3.36 и 3.38) и изменила файл Makefile (рис. 3.37).

```
/home/aatazaeva/work/study/2024-2025/МСПД/study_2024-2025_simulation-networks/labs/lab4/lab_netem_i/simple-delay/samost.py
with open('ping.dat', 'r') as file:
    s = []
    for line in file.readlines():
        if '\n' in line:
            line.replace('\n', "")
            s.append([int(j) for j in (line.split(" "))])
    s = [j[1] for j in s]

std = (sum([(1-(sum(s)/len(s))**2 for i in s))/(len(s)-1))**0.5
print(f"min: {min(s)} \nmax: {max(s)} \navg: {sum(s)/len(s)} \nstd: {std}")
```

Рис. 3.36: samost.py

```
/home/aatazaeva/work/study/2024-2025/МСПД/study_2024-2025_simulation-networks/labs/lab4/lab_netem_i/simple-delay/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

samost:
    sudo python samost.py
```

Рис. 3.37: Makefile. измененный

```
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make samost
sudo python samost.py
min: 200
max: 202
avg: 201.06060606060606
std: 0.49110832210880684
```

Рис. 3.38: samost.py выполнение

12. Очистила каталог от результатов проведения экспериментов:

make clean

4 Выводы

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