

Моделирование сетей передачи данных

**Лабораторная работа № 4. Эмуляция и измерение задержек в
глобальных сетях**

Демидова Екатерина Алексеевна

Содержание

1	Введение	4
2	Теоретическое введение	5
3	Выполнение лабораторной работы	6
3.1	Запуск лабораторной топологии	6
3.2	Добавление/изменение задержки в эмулируемой глобальной сети	8
3.3	Изменение задержки в эмулируемой глобальной сети	9
3.4	Восстановление исходных значений (удаление правил) задержки в эмулируемой глобальной сети	10
3.5	Добавление значения дрожания задержки в интерфейс подключения к эмулируемой глобальной сети	11
3.6	Добавление значения корреляции для джиттера и задержки в интерфейс подключения к эмулируемой глобальной сети	12
3.7	Распределение задержки в интерфейсе подключения к эмулируемой глобальной сети	13
3.8	Воспроизведение экспериментов	14
4	Выводы	20
	Список литературы	21

Список иллюстраций

3.1	Простейшая сеть	7
3.2	Проверка подключения	8
3.3	Изменение задержки на хосте h1	8
3.4	Задержка на обоих хостах	9
3.5	Изменение задержки	10
3.6	Восстановление исходных значений (удаление правил) задержки	11
3.7	Добавление значения дрожания задержки в интерфейс	12
3.8	Добавление значения корреляции для джиттера и задержки . . .	13
3.9	Распределение задержки в интерфейсе подключения к эмулируе- мой глобальной сети	14
3.10	Визуализация эксперимента	17
3.11	Визуализация эксперимента	18
3.12	Скрипт	18
3.13	Вывод скрипта	19

1 Введение

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.

Задачи

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.

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(`ipconfig`, `ping`) и даже получать доступ к терминалу. На коммутаторы можно добавлять различные правила и маршрутизировать трафик.

3 Выполнение лабораторной работы

3.1 Запуск лабораторной топологии

Зададим простейшую топологию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8. На хостах h1 и h2 введем команду `ifconfig`, чтобы отобразить информацию, относящуюся к их сетевым интерфейсам и назначенным им IP-адресам(рис. 3.1)

```
mininet@mininet-vm: ~  
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> После введения этой команды запуск  
ether 06:fd:b5:75:86:66 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 1111 bytes 572600 (572.6 KB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 1111 bytes 572600 (572.6 KB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
root@mininet-vm:/home/mininet#  
root@mininet-vm:/home/mininet# ifconfig  
h2-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
inet 10.0.0.2 netmask 255.0.0.0 broadcast 10.255.255.255  
ether ca:95:59:d9:2f:34 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 791 bytes 545596 (545.5 KB)  
RX errors 0 dropped 0 overruns 0 frame 0  
TX packets 791 bytes 545596 (545.5 KB)  
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
root@mininet-vm:/home/mininet#
```

Рис. 3.1: Простейшая сеть

Проверим подключение между хостами сети(рис. 3.2).

```
"host: h1" (на mininet-vm)
RX packets 1111 bytes 572600 (572.6 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 1111 bytes 572600 (572.6 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# ping 10.0.0.2 -c 6
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=2.35 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.361 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.076 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.078 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.080 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 5099ms
rtt min/avg/max/mdev = 0.076/0.504/2.350/0.831 ms
root@mininet-vm:/home/mininet#
```

Рис. 3.2: Проверка подключения

3.2 Добавление/изменение задержки в эмулируемой глобальной сети

В конце написано минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи (RTT): `rtt min/avg/max/mdev = 0.076/0.504/2.350/0.831 ms`.

Зададим задержку на хосте h1 и проверим ее наличие(рис. 3.3).

```
"host: h1" (на mininet-vm)
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.078 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.080 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 5099ms
rtt min/avg/max/mdev = 0.076/0.504/2.350/0.831 ms
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 100ms
root@mininet-vm:/home/mininet# ping 10.0.0.2 -c 6
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=100 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=100 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 5007ms
rtt min/avg/max/mdev = 100.104/100.363/100.977/0.306 ms
root@mininet-vm:/home/mininet#
```

Рис. 3.3: Изменение задержки на хосте h1

Действительно, теперь rtt около 100: rtt min/avg/max/mdev = 100.104/100.363/100.977/0 ms.

Добавим задержку на хост h2 и увидим, что теперь rtt около 200 rtt min/avg/max/mdev = 200.203/200.564/201.179/0.358 ms(рис. 3.4).

Рис. 3.4: Задержка на обоих хостах

3.3 Изменение задержки в эмулируемой глобальной сети

Изменим задержку на хостах на 50 мс, при этом у нас rtt min/avg/max/mdev = 100.148/100.745/102.727/0.912 ms(рис. 3.5)

```
"host: h1" (на mininet-vm)
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=201 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5007ms
rtt min/avg/max/mdev = 200.203/200.564/201.179/0.358 ms
root@mininet-vm:/home/mininet# sudo tc qdisc change dev h1-eth0 root netem delay 50ms
root@mininet-vm:/home/mininet# ping 10.0.0.2 -c 6
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=101 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=100 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=100 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 5007ms
rtt min/avg/max/mdev = 100.148/100.745/102.727/0.912 ms
root@mininet-vm:/home/mininet#

      ether ca:95:59:d9:2f:34 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 791 bytes 545596 (545.5 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 791 bytes 545596 (545.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# sudo tc qdisc add dev h2-eth0 root netem delay 100ms
root@mininet-vm:/home/mininet# sudo tc qdisc change dev h2-eth0 root netem delay 50ms
root@mininet-vm:/home/mininet#
```

Рис. 3.5: Изменение задержки

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

Вернем изначальное значение задержки и проверим корректность изменений
rtt min/avg/max/mdev = 0.050/0.268/0.590/0.231 ms(рис. 3.6).

```
"host: h1" (ha mininet-vn)
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=100 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=100 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 5007ms
rtt min/avg/max/mdev = 100.148/100.745/102.727/0.912 ms
root@mininet-vn:/home/mininet# sudo tc qdisc del dev h1-eth0 root netem
root@mininet-vn:/home/mininet# ping 10.0.0.2 -c 6
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=0.590 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.589 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.158 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.050 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.176 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.050 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5122ms
rtt min/avg/max/mdev = 0.050/0.268/0.590/0.231 ms
root@mininet-vn:/home/mininet#

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 791 bytes 545596 (545.5 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 791 bytes 545596 (545.5 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vn:/home/mininet# sudo tc qdisc add dev h2-eth0 root netem delay 1
00ms
root@mininet-vn:/home/mininet# sudo tc qdisc change dev h2-eth0 root netem dela
y 50ms
root@mininet-vn:/home/mininet# sudo tc qdisc del dev h2-eth0 root netem
root@mininet-vn:/home/mininet#
```

Рис. 3.6: Восстановление исходных значений (удаление правил) задержки

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

Добавим на узле h1 задержку в 100 мс со случайным отклонением 10 мс и проверим изменение(рис. 3.7).

```
root@mininet-vm:/home/mininet# ping 10.0.0.2
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.176 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.050 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5122ms
rtt min/avg/max/mdev = 0.050/0.268/0.590/0.231 ms
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 100ms 10ms
root@mininet-vm:/home/mininet# ping 10.0.0.2 -c 6
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=95.4 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=91.1 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=104 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=95.1 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=96.4 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 5007ms
rtt min/avg/max/mdev = 91.076/97.167/103.692/4.190 ms
root@mininet-vm:/home/mininet# sudo tc qdisc del dev h1-eth0 root netem
root@mininet-vm:/home/mininet#
```

Рис. 3.7: Добавление значения дрожания задержки в интерфейс

Видно, что rtt равняется 'rtt min/avg/max/mdev = 91.076/97.167/103.692/4.190 m'.

3.6 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%(рис. 3.8).

```
"host: h1" (na mininet-vm)
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 5007ms
rtt min/avg/max/mdev = 91.076/97.167/103.692/4.190 ms
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 1
00ms 10ms 25%
root@mininet-vm:/home/mininet# ping 10.0.0.2 -c 6
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=106 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=95.6 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=106 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=91.4 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=90.9 ms
--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5007ms
rtt min/avg/max/mdev = 90.922/98.734/106.415/6.396 ms
root@mininet-vm:/home/mininet#
```

Рис. 3.8: Добавление значения корреляции для джиттера и задержки

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%: $rtt\ min/avg/max/mdev = 90.922/98.734/106.415/6.396\ ms$.

3.7 Распределение задержки в интерфейсе подключения к эмулируемой глобальной сети

Задайте нормальное распределение задержки на узле h1 в эмулируемой сети(рис. 3.9).

```
"host: h1" (на mininet-vm)
00ms 20ms distribution normal
Error: Exclusivity flag on, cannot modify.
root@mininet-vm:/home/mininet#
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 1
00ms 20ms distribution normal
Error: Exclusivity flag on, cannot modify.
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 1
00ms 20ms distribution normal
root@mininet-vm:/home/mininet# ping 10.0.0.2 -c 6
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=73.1 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=93.5 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=71.3 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=63.0 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=114 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=136 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5007ms
rtt min/avg/max/mdev = 63.043/91.901/136.353/26.096 ms
root@mininet-vm:/home/mininet#
```

Рис. 3.9: Распределение задержки в интерфейсе подключения к эмулируемой глобальной сети

Убедимся, что все пакеты, покидающие хост h1 на интерфейсе h1-eth0, будут иметь время задержки, которое распределено в диапазоне 100 мс \pm 20 мс.

3.8 Воспроизведение экспериментов

В виртуальной среде mininet в своём рабочем каталоге с проектами создадим каталог simple-delay и перейдем в него. Создадим скрипт для эксперимента lab_netem_i.py:

```
#!/usr/bin/env python
"""
Simple experiment.
Output: ping.dat
"""

from mininet.net import Mininet
from mininet.node import Controller
from mininet.cli import CLI
```

```

from mininet.log import setLogLevel, info
import time

def emptyNet():

    "Create an empty network and add nodes to it."
    net = Mininet( controller=Controller, waitConnected=True )

    info( '*** Adding controller\n' )
    net.addController( 'c0' )

    info( '*** Adding hosts\n' )
    h1 = net.addHost( 'h1', ip='10.0.0.1' )
    h2 = net.addHost( 'h2', ip='10.0.0.2' )

    info( '*** Adding switch\n' )
    s1 = net.addSwitch( 's1' )

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

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

    info( '*** Set 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}\'' |
e '\s/time=//g\' -e '\s/icmp_seq=//g\' > ping.dat' )

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

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

```

В этом скрипте создается простейшая топология сети, затем с помощью команд, использованных нами ранее задается задержка в 100 мс для обоих хостов, после чего пингуется второй хост(100 сообщений отправляется), при этом из сообщений при пинге вытаскиваются номер сообщения и значение времени, которые записываются в файл с данными.

Создадим также скрипт для визуализации ping_plot результатов эксперимента:

```

#!/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

```

Создадим Makefile для управления процессом проведения эксперимента:


```
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
```

В результате получим график(рис. 3.10).

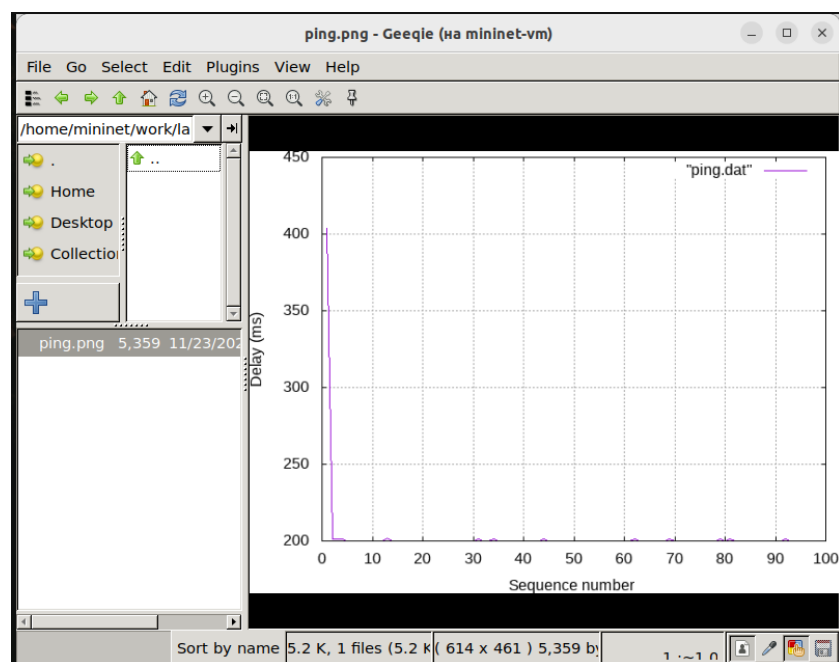


Рис. 3.10: Визуализация эксперимента

Из файла ping.dat удалим первую строку и заново построим график(рис. 3.11).

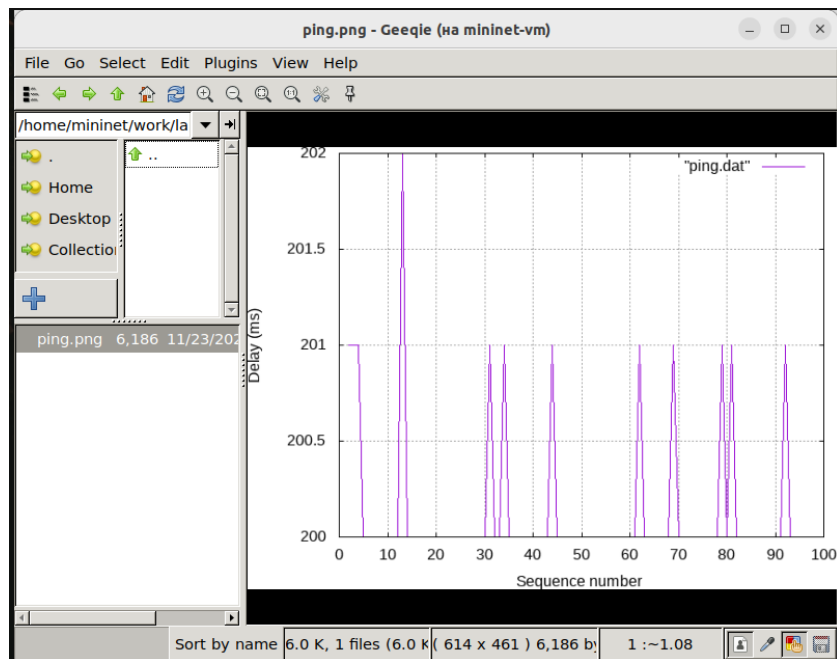


Рис. 3.11: Визуализация эксперимента

Разработаем скрипт для вычисления на основе данных файла `ping.dat` минимального, среднего, максимального и стандартного отклонения времени приёма-передачи. Добавьте правило запуска скрипта в `Makefile` (рис. 3.12).

```

mininet@mininet-vm: ~/work/lab_netem_i/simple-delay
GNU nano 4.8 script rtt.py
with open('ping.dat', 'r') as f:
    s = []
    for line in f.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([(i-(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. 3.12: Скрипт

Запустим скрипт с помощью `Makefile` (рис. 3.13).

```
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano Makefile
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make rtt
sudo python script_rtt.py
min: 200
max: 202
avg: 200.13131313131314
std: 0.36829626069529314
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$
```

Рис. 3.13: Вывод скрипта

4 Выводы

В результате выполнения работы познакомились с NETEM – инструментом для тестирования производительности приложений в виртуальной сети, а также получили навык 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: <http://mininet.org/> (дата обращения: 17.11.2024).