

Отчёт по лабораторной работе №4

Эмуляция и измерение задержек в глобальных сетях

Студент: Кузнецова София Вадимовна

Содержание

Цель работы	5
Теоретическое введение	6
Выполнение лабораторной работы	7
Добавление/изменение задержки в эмулируемой глобальной сети	9
Изменение задержки в эмулируемой глобальной сети	10
Восстановление исходных значений (удаление правил) задержки в эмули- руемой глобальной сети	11
Добавление значения дрожания задержки в интерфейс подключения к эмулируемой глобальной сети	12
Добавление значения корреляции для джиттера и задержки в интерфейс подключения к эмулируемой глобальной сети	13
Распределение задержки в интерфейсе подключения к эмулируемой гло- бальной сети	13
Воспроизведение экспериментов. Добавление задержки для интерфейса, подключающегося к эмулируемой глобальной сети	14
Самостоятельная работа	20
Выводы	26

Список иллюстраций

0.1	Исправление прав запуска X-соединения	7
0.2	Простейшая топология	7
0.3	ifconfig на хостах h1	8
0.4	ifconfig на хостах h2	8
0.5	Проверка подключения между хостами	9
0.6	Проверка подключения между хостами	9
0.7	Добавление задержки в 100мс	10
0.8	Двунаправленная задержка соединения	10
0.9	Изменение задержки на 50мс	11
0.10	Изменение задержки на 50мс	11
0.11	Восстановление исходных значений задержки	11
0.12	Восстановление исходных значений задержки	12
0.13	Добавление значения дрожания задержки в интерфейс подключения	12
0.14	Добавление значения корреляции для джиттера и задержки в интерфейс подключения	13
0.15	Распределение задержки в интерфейсе подключения	14
0.16	Скрипт для визуализации ping_plot	17
0.17	Makefile для управления процессом проведения эксперимента	17
0.18	Результат выполнения скриптов	18
0.19	Результат выполнения скриптов	18
0.20	Скрипт rtt.py	19
0.21	Добавление правила запуска скрипта в Makefile	19
0.22	Результат работы скрипта rtt.py	19
0.23	Воспроизводимый эксперимент по изменению задержки	20
0.24	Воспроизводимый эксперимент по изменению задержки	20
0.25	Просмотр графика	21
0.26	Воспроизводимый эксперимент по изменению задержки	21
0.27	Воспроизводимый эксперимент по изменению задержки	22
0.28	Просмотр графика	22
0.29	Воспроизводимый эксперимент по изменению задержки	23
0.30	Воспроизводимый эксперимент по изменению задержки	23
0.31	Просмотр графика	24
0.32	Воспроизводимый эксперимент по изменению задержки	24
0.33	Воспроизводимый эксперимент по изменению задержки	25
0.34	Просмотр графика	25

Список таблиц

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.

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.

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.

```
mininet@mininet-vm:~$ xauth list $DISPLAY
mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 f2c4512fcecdabb350f4fe1968db2f00
mininet@mininet-vm:~$ sudo -i
root@mininet-vm:~# xauth list $DISPLAY
root@mininet-vm:~# xauth add mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 f2c4512fcecdabb35
0f4fe1968db2f00
root@mininet-vm:~# xauth list $DISPLAY
mininet-vm/unix:10 MIT-MAGIC-COOKIE-1 f2c4512fcecdabb350f4fe1968db2f00
root@mininet-vm:~# logout
```

Рис. 0.1: 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. 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. 0.2: 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.

```
X "host: h1"@mininet-vm
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 96:cb:90:d8:54:7c 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 815 bytes 255288 (255.2 KB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 815 bytes 255288 (255.2 KB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# █
```

Рис. 0.3: ifconfig на хостах h1

```
X "host: h2"@mininet-vm
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 5a:0d:94:a7:8e:0a 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 706 bytes 249964 (249.9 KB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 706 bytes 249964 (249.9 KB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-vm:/home/mininet# █
```

Рис. 0.4: ifconfig на хостах h2

Проверим подключение между хостами h1 и h2 с помощью команды ping с параметром -c 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=2.38 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.282 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.069 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.164 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.076 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.151 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5088ms
rtt min/avg/max/mdev = 0.069/0.520/2.382/0.835 ms
root@mininet-vm:/home/mininet# █

```

Рис. 0.5: Проверка подключения между хостами

```

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=1.84 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.068 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.066 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.068 ms
64 bytes from 10.0.0.1: icmp_seq=5 ttl=64 time=0.097 ms
64 bytes from 10.0.0.1: icmp_seq=6 ttl=64 time=0.076 ms

--- 10.0.0.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5073ms
rtt min/avg/max/mdev = 0.066/0.368/1.835/0.655 ms
root@mininet-vm:/home/mininet# █

```

Рис. 0.6: Проверка подключения между хостами

Добавление/изменение задержки в эмулируемой глобальной сети

На хосте h1 добавим задержку в 100 мс к выходному интерфейсу.

```
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

```
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 1
00ms
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=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 5008ms
rtt min/avg/max/mdev = 100.145/100.721/102.378/0.785 ms
root@mininet-vm:/home/mininet# █
```

Рис. 0.7: Добавление задержки в 100мс

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

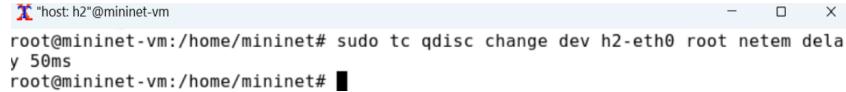
```
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h2-eth0 root netem delay 1
00ms
root@mininet-vm:/home/mininet# ping -c 6 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=201 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=201 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=201 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=201 ms
64 bytes from 10.0.0.1: icmp_seq=5 ttl=64 time=200 ms
64 bytes from 10.0.0.1: icmp_seq=6 ttl=64 time=201 ms

--- 10.0.0.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5007ms
rtt min/avg/max/mdev = 200.213/200.855/201.319/0.349 ms
root@mininet-vm:/home/mininet# █
```

Рис. 0.8: Двунаправленная задержка соединения

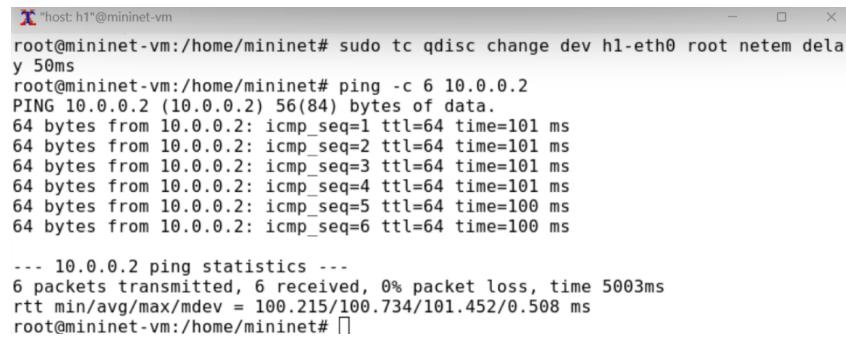
Изменение задержки в эмулируемой глобальной сети

Изменим задержку со 100 мс до 50 мс для отправителя h1 и для получателя h2. 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 h2-eth0 root netem delay 50ms
```

Рис. 0.9: Изменение задержки на 50мс



```
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=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=100 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=100 ms

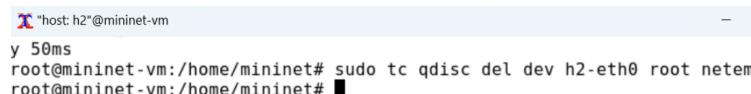
--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5003ms
rtt min/avg/max/mdev = 100.215/100.734/101.452/0.508 ms
root@mininet-vm:/home/mininet#
```

Рис. 0.10: 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.



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

Рис. 0.11: Восстановление исходных значений задержки

```

X "host: h1"@"mininet-vm
rtt min/avg/max/mdev = 100.215/100.734/101.452/0.508 ms
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=1.76 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.544 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.245 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.076 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.079 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.092 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5089ms
rtt min/avg/max/mdev = 0.076/0.465/1.756/0.600 ms
root@mininet-vm:/home/mininet#

```

Рис. 0.12: 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 -c 6. Восстановим конфигурацию интерфейса по умолчанию на узле h1.

```

X "host: h1"@"mininet-vm
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=99.5 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=97.4 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=110 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 5008ms
rtt min/avg/max/mdev = 97.390/104.275/109.619/4.377 ms
root@mininet-vm:/home/mininet# sudo tc qdisc del dev h1-eth0 root netem
root@mininet-vm:/home/mininet#

```

Рис. 0.13: 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.

```
X "host: h1"@mininet-vm
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 1
90ms 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.
54 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=99.8 ms
54 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=108 ms
54 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=108 ms
54 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=108 ms
54 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=99.8 ms
54 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=95.6 ms

--- 10.0.0.2 ping statistics ---
5 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 95.585/103.157/108.380/4.968 ms
root@mininet-vm:/home/mininet#
```

Рис. 0.14: Добавление значения корреляции для джиттера и задержки в интерфейс 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.

```

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 -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=51.7 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=113 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=86.4 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=120 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=72.8 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=77.2 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 51.725/86.861/120.263/23.506 ms
root@mininet-vm:/home/mininet# sudo tc qdisc del dev h1-eth0 root netem
root@mininet-vm:/home/mininet#

```

Рис. 0.15: Распределение задержки в интерфейсе подключения

Воспроизведение экспериментов. Добавление задержки для интерфейса, подключающегося к эмулируемой глобальной сети

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

```

1#!/usr/bin/env python
2
3"""
4 Simple experiment.
5 Output: ping.dat
6"""
7
8from mininet.net import Mininet
9from mininet.node import Controller

```

```
10 from mininet.cli import CLI
11 from mininet.log import setLogLevel, info
12 import time
13
14 def emptyNet():
15
16     "Create an empty network and add nodes to it."
17
18     net = Mininet( controller=Controller, waitConnected=True )
19
20     info( "*** Adding controller\n" )
21     net.addController( 'c0' )
22
23     info( "*** Adding hosts\n" )
24     h1 = net.addHost( 'h1', ip='10.0.0.1' )
25
26     h2 = net.addHost( 'h2', ip='10.0.0.2' )
27     info( "*** Adding switch\n" )
28     s1 = net.addSwitch( 's1' )
29
```

```

30 info( '*** Creating links\n' )

31 net.addLink( h1, s1 )
32 net.addLink( h2, s1 )
33

34 info( '*** Starting network\n' )
35 net.start()
36
37 info( '*** Set delay\n' )
38 h1.cmdPrint( 'tc qdisc add dev h1-eth0 root netem delay 100ms' )
39 h2.cmdPrint( 'tc qdisc add dev h2-eth0 root netem delay 100ms' )

40
41 time.sleep(10) # Wait 10 seconds
42

43 info( '*** Ping\n' )
44 h1.cmdPrint( 'ping -c 100', h2.IP(), '| grep "time=' | awk \'{print $5, $7}\'' | sed -e '\s/time=/'
45

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

48
49 if __name__ == '__main__':
50     setLogLevel( 'info' )

51 emptyNet()

```

Создаём скрипт для визуализации ping_plot результатов эксперимента.

```
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. 0.16: Скрипт для визуализации ping_plot

Зададим права доступа к файлу скрипта: chmod +x ping_plot. Создадим Makefile для управления процессом проведения эксперимента.

```
GNU nano 4.8                               Makefile                                Modified
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. 0.17: Makefile для управления процессом проведения эксперимента

Выполним эксперимент. Продемонстрируем построенный в результате выполнения скриптов график.

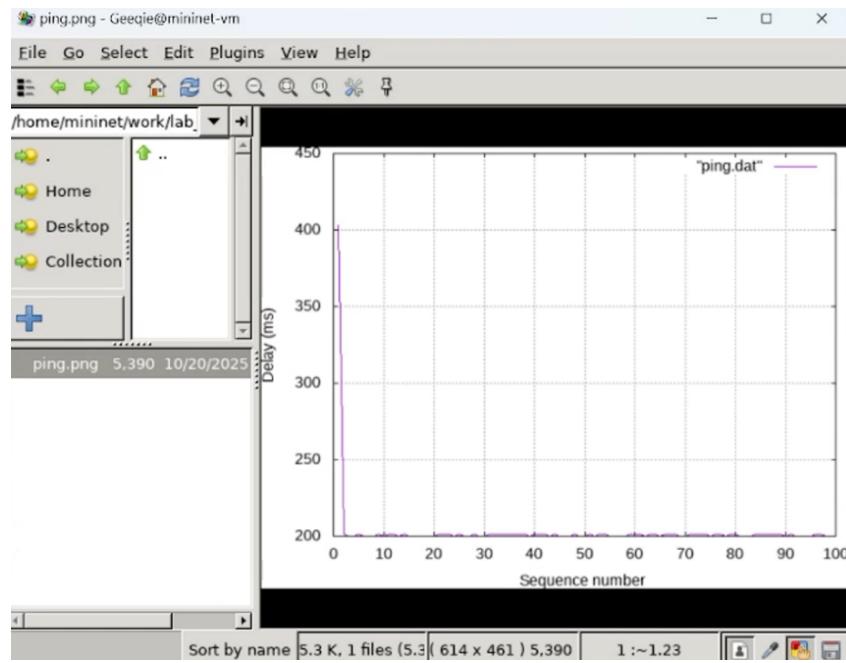


Рис. 0.18: Результат выполнения скриптов

Из файла ping.dat удалим первую строку и заново постройте график. Продемонстрируем построенный в результате график.

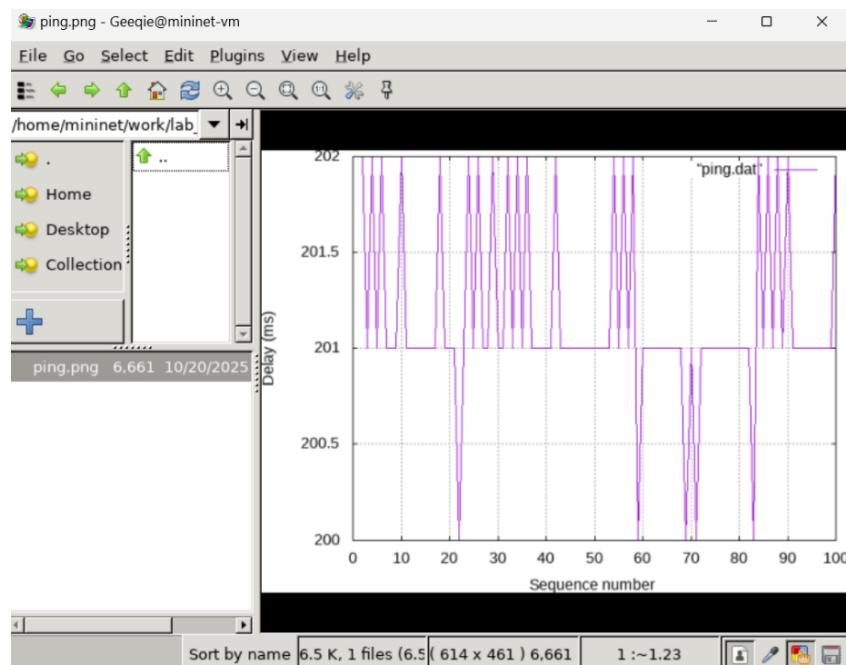


Рис. 0.19: Результат выполнения скриптов

Разработайте скрипт для вычисления на основе данных файла ping.dat минимального, среднего, максимального и стандартного отклонения времени приёма-передачи.

```
GNU nano 4.8                                         rtt.py                                         Modified
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)} \n max: {max(s)} \n avg: {sum(s) / len(s)} \n std: {std}")
```

Рис. 0.20: Скрипт rtt.py

Добавим правило запуска скрипта в Makefile.

```
GNU nano 4.8                                         Makefile                                         Modified
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
stats: ping.dat
    python rtt.py
clean:
    -rm -f *.dat *.png
```

Рис. 0.21: Добавление правила запуска скрипта в Makefile

Продемонстрируем работу скрипта с выводом значений на экран или в отдельный файл.

```
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ sudo python rtt.py
min: 200
max: 201
avg: 200.43434343434345
std: 0.49819298288539626
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$
```

Рис. 0.22: Результат работы скрипта rtt.py

Очистим каталог от результатов проведения экспериментов.

Самостоятельная работа

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.

```
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 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
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot
```

Рис. 0.23: Воспроизводимый эксперимент по изменению задержки

```
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ geeqie ping.png
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano ping.dat
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make ping.png
./ping_plot
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ geeqie ping.png
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make stats
python rtt.py
min: 100
max: 102
avg: 100.8989898989899
std: 0.33487583733010773
```

Рис. 0.24: Воспроизводимый эксперимент по изменению задержки

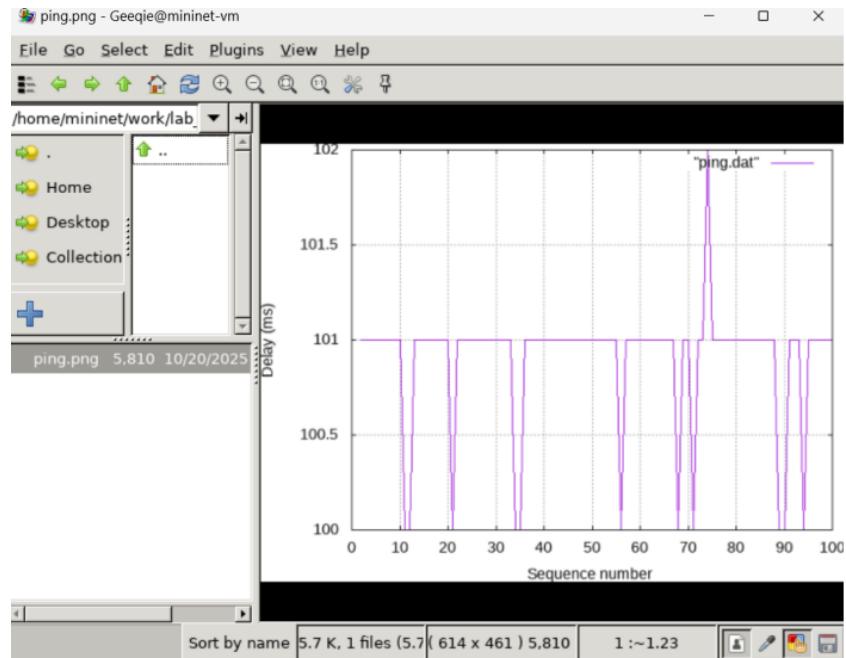


Рис. 0.25: Просмотр графика

```
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 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
```

Рис. 0.26: Воспроизводимый эксперимент по изменению задержки

```
mininet@mininet-vm:~/work/lab_neterm_i/simple-delay$ nano ping.dat
mininet@mininet-vm:~/work/lab_neterm_i/simple-delay$ make ping.png
./ping_plot
mininet@mininet-vm:~/work/lab_neterm_i/simple-delay$ geeqie ping.png
mininet@mininet-vm:~/work/lab_neterm_i/simple-delay$ make stats
python rtt.py
min: 190
max: 211
avg: 201.27272727272728
std: 6.291909579734498
```

Рис. 0.27: Воспроизводимый эксперимент по изменению задержки

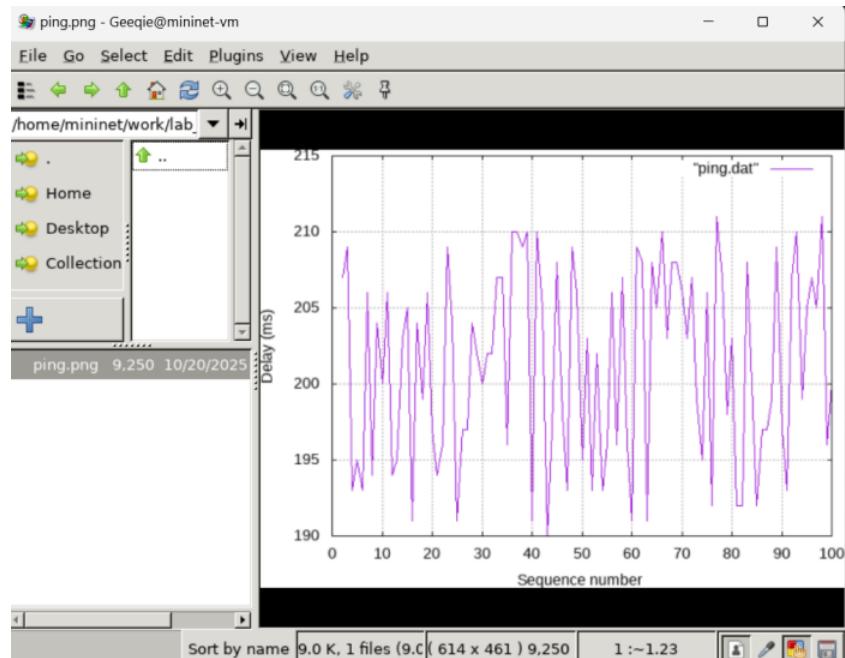


Рис. 0.28: Просмотр графика

```

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 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

```

Рис. 0.29: Воспроизводимый эксперимент по изменению задержки

```

mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ geeqie ping.png
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano ping.dat
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make ping.png
./ping_plot
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ geeqie ping.png
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make stats
python rtt.py
min: 191
max: 211
avg: 200.646464646464
std: 5.738009914761444

```

Рис. 0.30: Воспроизводимый эксперимент по изменению задержки

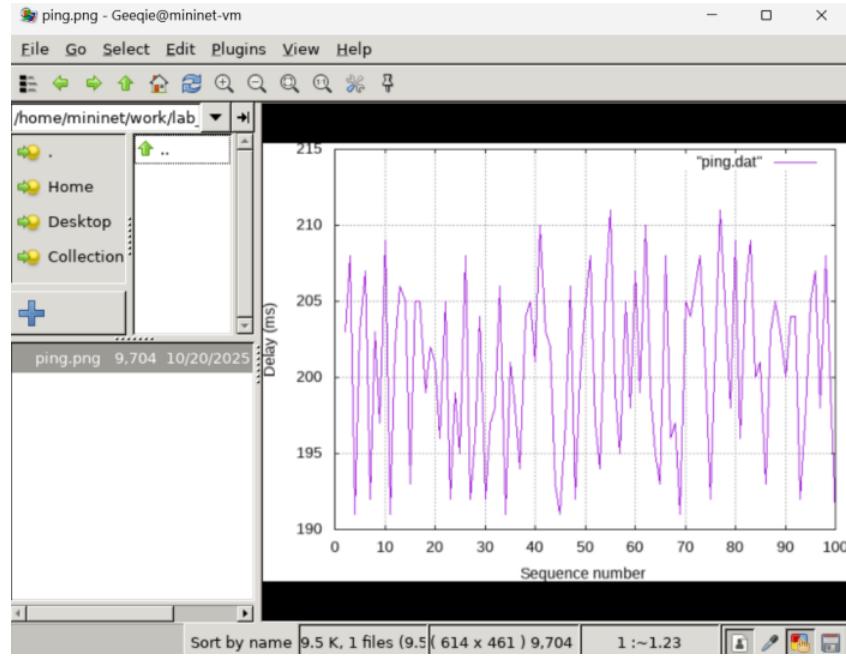


Рис. 0.31: Просмотр графика

```
mininet@mininet-vm:~/work/lab_netem_1/simple-delay$ make
sudo python lab_netem_1.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
```

Рис. 0.32: Воспроизведимый эксперимент по изменению задержки

```
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano ping.dat
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make ping.png
./ping_plot
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ geeqie ping.png
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make stats
python rtt.py
min: 174
max: 225
avg: 199.82828282828282
std: 10.830833163761353
```

Рис. 0.33: Воспроизводимый эксперимент по изменению задержки

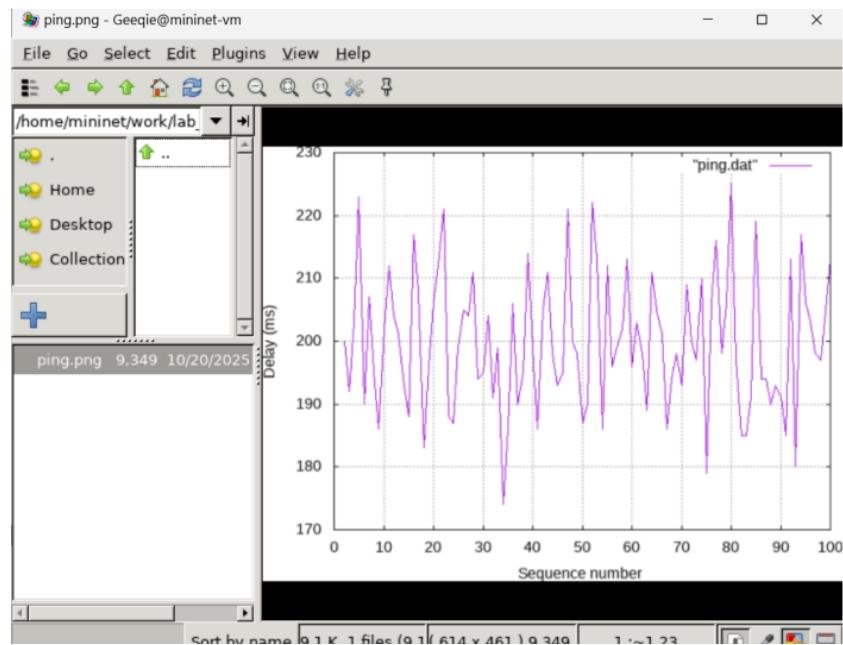


Рис. 0.34: Просмотр графика

Выводы

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