

Моделирование сетей передачи данных

**Отчёт по лабораторной работе №4: Эмуляция и измерение задержек в
глобальных сетях**

Кармацкий Никита Сергеевич

Содержание

1	Цель работы	6
2	Выполнение лабораторной работы	7
3	Вывод	93
4	Список литературы. Библиография	94

List of Figures

2.1	Исправление прав запуска X-соединения в виртуальной машине mininet	8
2.2	Создание простейшей топологии	10
2.3	Отображение информации их сетевых интерфейсов и IP-адресов	12
2.4	Проверка подключения между хостами	14
2.5	Добавление задержки в 100 мс к выходному интерфейсу на хосте h1	16
2.6	Проверка	18
2.7	Добавление задержки в 100 мс к выходному интерфейсу на хосте h2	20
2.8	Проверка	22
2.9	Изменение задержек до 100 мс до 50мс на хостах	24
2.10	Проверка	26
2.11	Восстановление конфигураций по умолчанию	28
2.12	Добавление задержки 100 мс со случайным отклонением на хост h1	30
2.13	Проверка	32
2.14	Восстановление конфигурацию по умолчанию	34
2.15	Проверка	36
2.16	Восстановление конфигурацию по умолчанию	38
2.17	Настройка нормального распределения задержки на узле h1 в эмулируемой сети	40
2.18	Проверка	42
2.19	Восстановление конфигурацию по умолчанию	44
2.20	Завершение работу mininet в интерактивном режиме	46
2.21	Обновление репозитория ПО на VM	48
2.22	Установка пакета geee	50
2.23	Создание каталога	52
2.24	Создание каталога simple-delay	54
2.25	Создание скрипта для эксперимента lab_netem_i.py	56
2.26	Создание файла ping_plot	58
2.27	Создание скрипта ping_plot для визуализации результатов эксперимента	60
2.28	Настройка прав доступа к файлу скрипта	62
2.29	Создание файла Makefile	64
2.30	Добавления скрипта в Makefile для управления процессом проведения эксперимента	66
2.31	Выполнение эксперимента	68
2.32	Просмотр графика	70
2.33	Удаление первой строки из файла ping.dat	72

2.34	Повторное построение графика	73
2.35	Просмотр графика	75
2.36	Разработка скрипта для вычисления на основе данных файла ping.dat минимального, среднего, максимального и стандартного отклонения времени приёма-передачи	77
2.37	Добавление правила запуска скрипта в Makefil	78
2.38	Проверка	79
2.39	Воспроизводимый эксперимент по изменению задержки	81
2.40	Воспроизводимый эксперимент по изменению задержки	82
2.41	Просмотр графика	83
2.42	Воспроизводимый эксперимент по изменению джиттера	84
2.43	Воспроизводимый эксперимент по изменению джиттера	85
2.44	Просмотр графика	86
2.45	Воспроизводимый эксперимент по изменению значения корреля- ции для джиттера и задержки	87
2.46	Воспроизводимый эксперимент по изменению значения корреля- ции для джиттера и задержки	88
2.47	Просмотр графика	89
2.48	Воспроизводимый эксперимент по изменению распределения вре- мени задержки в эмулируемой глобальной сети	90
2.49	Воспроизводимый эксперимент по изменению распределения вре- мени задержки в эмулируемой глобальной сети	91
2.50	Просмотр графика	92

List of Tables

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 Выполнение лабораторной работы

1. В виртуальной машине mininet исправим права запуска X-соединения (рис. 2.1):

Age Group	Total	Male	Female	Male	Female
18-24	100	100	100	100	100
25-34	100	100	100	100	100
35-44	100	100	100	100	100
45-54	100	100	100	100	100
55-64	100	100	100	100	100
65+	100	100	100	100	100

Рис. 2.1: Исправление прав запуска X-соединения в виртуальной машине mininet

2. Зададим простейшую топологию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8 (рис. 2.2):

```
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> sudo tc qdisc add dev h1-eth0 root netem delay 100ms
*** Unknown command: sudo tc qdisc add dev h1-eth0 root netem delay 100ms
mininet> █
```

Рис. 2.2: Создание простейшей топологии

3. На хостах h1 и h2 введём 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. 2.3):

```
"host: h1"
root@mininet-virtual-machine: /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 1e:b9:46:ad:a8:6a 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 1791 bytes 725728 (725.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1791 bytes 725728 (725.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-virtual-machine: /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=6.74 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.347 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.098 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.067 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.142 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.160 ms

"host: h2"
root@mininet-virtual-machine: /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 f2:cc:60:57:22:ec 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 1878 bytes 683960 (683.9 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1878 bytes 683960 (683.9 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@mininet-virtual-machine: /home/mininet# sudo tc qdisc add dev h2-eth0 root netem delay 100ms
root@mininet-virtual-machine: /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=204 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
```

Рис. 2.3: Отображение информации их сетевых интерфейсов и IP-адресов

4. Проверим подключение между хостами h1 и h2 с помощью команды ping с параметром -c 6(рис. 2.4):

```
"host:h1"
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=6.74 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.347 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.098 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.067 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.142 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=0.160 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5102ms
rtt min/avg/max/mdev = 0.067/1.250/6.743/2.153 ms
```

Рис. 2.4: Проверка подключения между хостами

5. На хосте h1 добавим задержку в 100 мс к выходному интерфейсу (рис. 2.5):

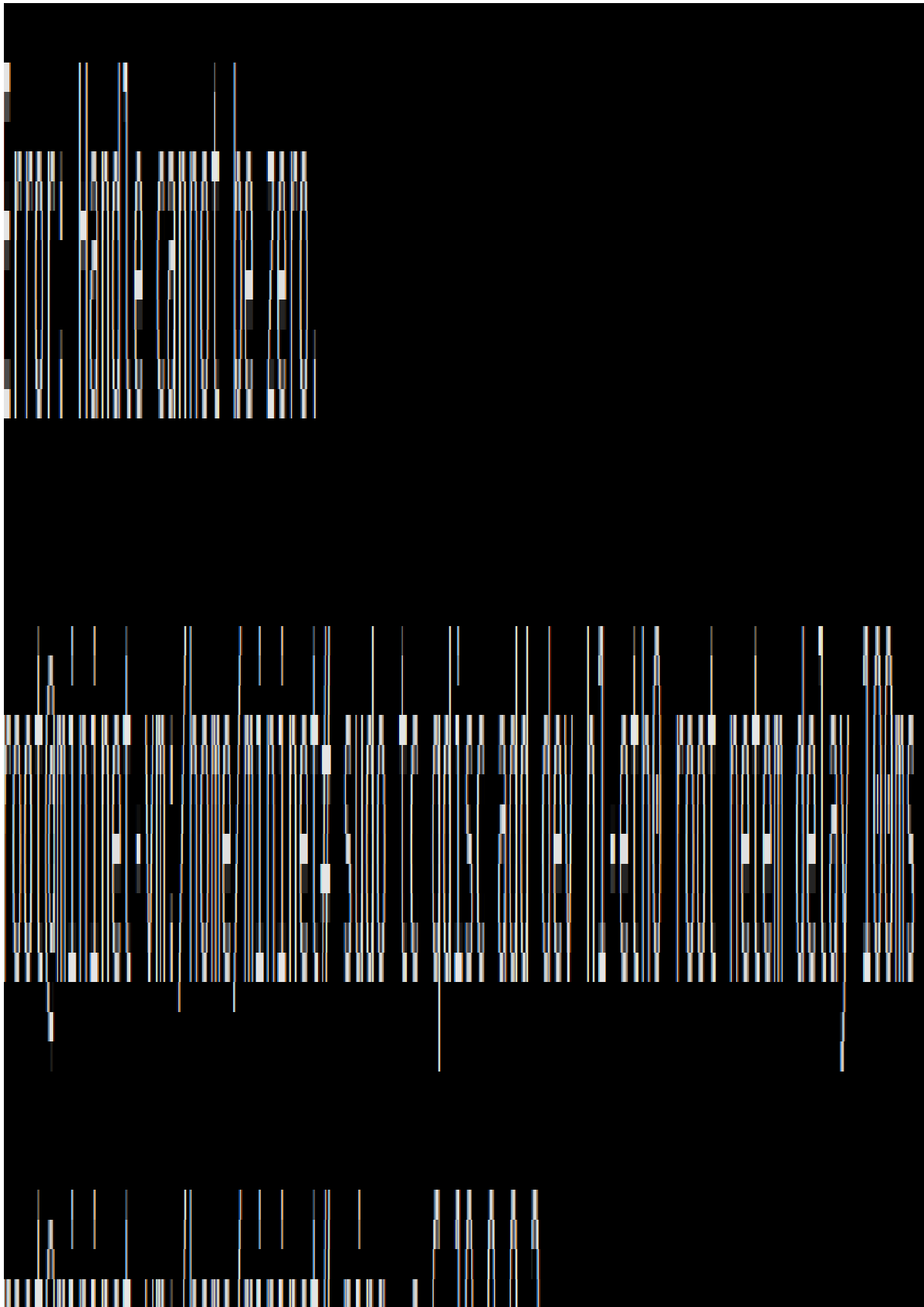


Рис. 2.5: Добавление задержки в 100 мс к выходному интерфейсу на хосте h1

6. Проверим, что соединение от хоста h1 к хосту h2 имеет задержку 100 мс, используя команду `ping` с параметром `-c 6` с хоста h1 (рис. 2.6):

"host: h1"

```
--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5102ms
rtt min/avg/max/mdev = 0.067/1.259/6.743/2.453 ms
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 netem delay 10
Error: Handle cannot be zero.
root@mininet-vm:/home/mininet# tc qdisc add dev h1-eth0 netem delay 100ms
Error: Handle cannot be zero.
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem del
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=105 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=101 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 100.578/101.785/105.390/1.702 ms
```

Рис. 2.6: Проверка

7. Для эмуляции глобальной сети с двунаправленной задержкой необходимо к соответствующему интерфейсу на хосте h2 также добавить задержку в 100 миллисекунд (рис. 2.7)

```
time=201 ms
"host:h2"

inet 127.0.0.1 netmask 255.0.0.0
loop txqueuelen 1000 (Local Loopback)
RX packets 1878 bytes 683960 (683.9 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 1878 bytes 683960 (683.9 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# ping -c 6 10.0.0.1
```

Рис. 2.7: Добавление задержки в 100 мс к выходному интерфейсу на хосте h2

8. 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 (рис. 2.8):

"host: h1"

```
rtt min/avg/max/mdev = 100.578/101.785/105.390/1.702 ms
```

```
root@mininet-vm:/home/mininet# ^C
```

```
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=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=204 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=204 ms
```

```
... 10.0.0.2 ping statistics ...
```

```
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
```

```
rtt min/avg/max/mdev = 200.512/201.868/204.071/1.389 ms
```

```
root@mininet-vm:/home/mininet# sudo tcpdump -i eth0 -s 1500 -w /tmp/eth0.pcap
```

Рис. 2.8: Проверка

9. Изменим задержку со 100 мс до 50 мс для отправителя h_1 и для получателя h_2 (рис. 2.9):

```
"host:h1"
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=204 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=204 ms

... 10.0.0.2 ping statistics ...
6 packets transmitted, 6 received, 0% packet loss, time 5000ms
rtt min/avg/max/mdev = 200.512/201.868/204.071/1.389 ms
root@mininet-vm:/home/mininet# sudo tc qdisc change dev h1-eth0 root netem delay 50ms

"host:h2"
rtt min/avg/max/mdev = 200.336/201.930/204.838/1.681 ms
root@mininet-vm:/home/mininet# sudo tc qdisc change dev h2-eth0 root netem delay 50ms
root@mininet-vm:/home/mininet# sudo tc qdisc del dev h2-eth0 root netem
```

Рис. 2.9: Изменение задержек до 100 мс до 50мс на хостах

10. Проверим, что соединение от хоста h1 к хосту h2 имеет задержку 100 мс, используя команду `ping` с параметром `-c 6` с терминала хоста h1 (рис. 2.10):

```
host: h1

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 200.512/201.868/204.071/1.389 ms
root@mininet-vm:/home/mininet# sudo tc qdisc change dev h1-eth0 root netem delay 50ms
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=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 5011ms
rtt min/avg/max/mdev = 100.614/101.062/101.832/0.506 ms
```

Рис. 2.10: Проверка

11. Восстановим конфигурацию по умолчанию, удалив все правила, применённые к сетевому планировщику соответствующего интерфейса(рис. 2.11):

```
"host: h1"

6 packets transmitted, 6 received, 0% packet loss, time 5011ms
rtt min/avg/max/mdev = 100.614/101.062/101.832/0.506 ms
root@mininet-virtual-machine: /home/mininet# sudo tc qdisc del dev h1-eth0 root netem

"host: h2"

t min/avg/max/mdev = 200.336/201.930/204.838/1.681 ms
ot@mininet-virtual-machine: /home/mininet# sudo tc qdisc change dev h2-eth0 root netem delay 50ms
ot@mininet-virtual-machine: /home/mininet# sudo tc qdisc del dev h2-eth0 root netem
ot@mininet-virtual-machine: /home/mininet#
```

Рис. 2.11: Восстановление конфигураций по умолчанию

12. Добавим на узле h1 задержку в 100 мс со случайным отклонением 10 мс (рис. 2.12):

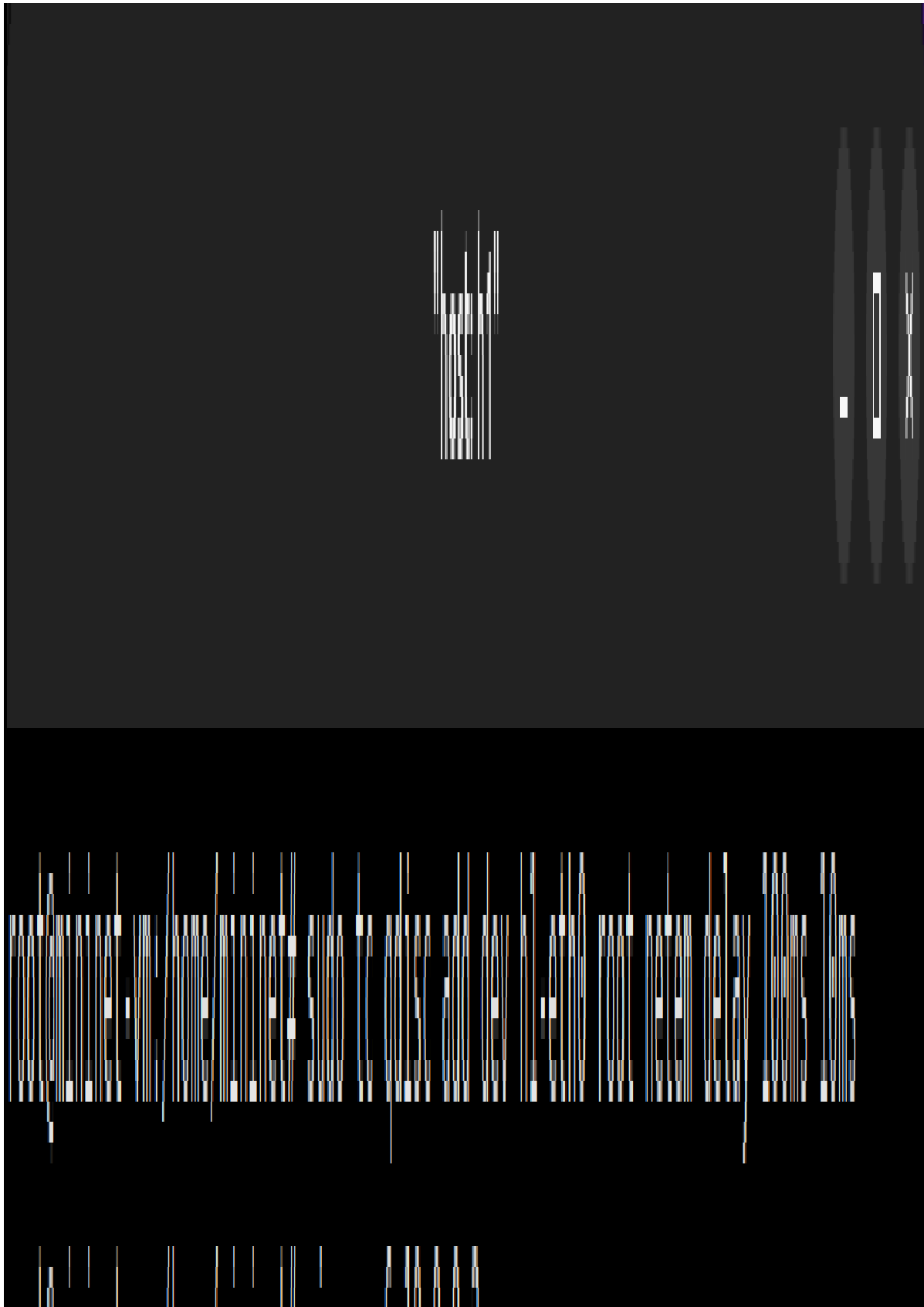


Рис. 2.12: Добавление задержки 100 мс со случайным отклонением на хост h1

13. 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` (рис. 2.13):

```

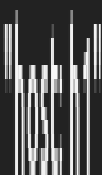
"host: h1"
root@mininet-vm:/home/mininet# sudo tc qdisc add dev h1-eth0 root netem delay 100ms 10ms
root@mininet-vm:/home/mininet# ping -c 6 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=98.2 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=104 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=106 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=96.2 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=105 ms

--- 10.0.0.2 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 96.182/102.169/105.643/3.645 ms

```

Рис. 2.13: Проверка

14. Восстановим конфигурацию интерфейса по умолчанию на узле h1(рис. 2.14):



34

15. 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` (рис. 2.15):

```

"host: h1"
rtt min/avg/max/mdev = 96.182/102.169/105.643/3.645 ms
root@mininet-virtual-machine:~# sudo tc qdisc del dev h1-eth0 root netem
root@mininet-virtual-machine:~# sudo tc qdisc add dev h1-eth0 root netem delay 100ms 10ms 25%
root@mininet-virtual-machine:~# 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=103 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=93.7 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=108 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=95.8 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=96.8 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=101 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=104 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=95.5 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=95.7 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=105 ms
64 bytes from 10.0.0.2: icmp_seq=11 ttl=64 time=100 ms
64 bytes from 10.0.0.2: icmp_seq=12 ttl=64 time=100 ms
64 bytes from 10.0.0.2: icmp_seq=13 ttl=64 time=90.7 ms
64 bytes from 10.0.0.2: icmp_seq=14 ttl=64 time=104 ms
64 bytes from 10.0.0.2: icmp_seq=15 ttl=64 time=107 ms
64 bytes from 10.0.0.2: icmp_seq=16 ttl=64 time=101 ms
64 bytes from 10.0.0.2: icmp_seq=17 ttl=64 time=95.7 ms
64 bytes from 10.0.0.2: icmp_seq=18 ttl=64 time=93.3 ms
64 bytes from 10.0.0.2: icmp_seq=19 ttl=64 time=105 ms
64 bytes from 10.0.0.2: icmp_seq=20 ttl=64 time=110 ms

--- 10.0.0.2 ping statistics ---
20 packets transmitted, 20 received, 0% packet loss, time 19041ms
rtt min/avg/max/mdev = 90.686/100.307/110.096/5.297 ms
root@mininet-virtual-machine:~# sudo tc qdisc del dev h1-eth0 root netem

```

Рис. 2.15: Проверка

16. Восстановим конфигурацию интерфейса по умолчанию на узле h1(рис. 2.16):

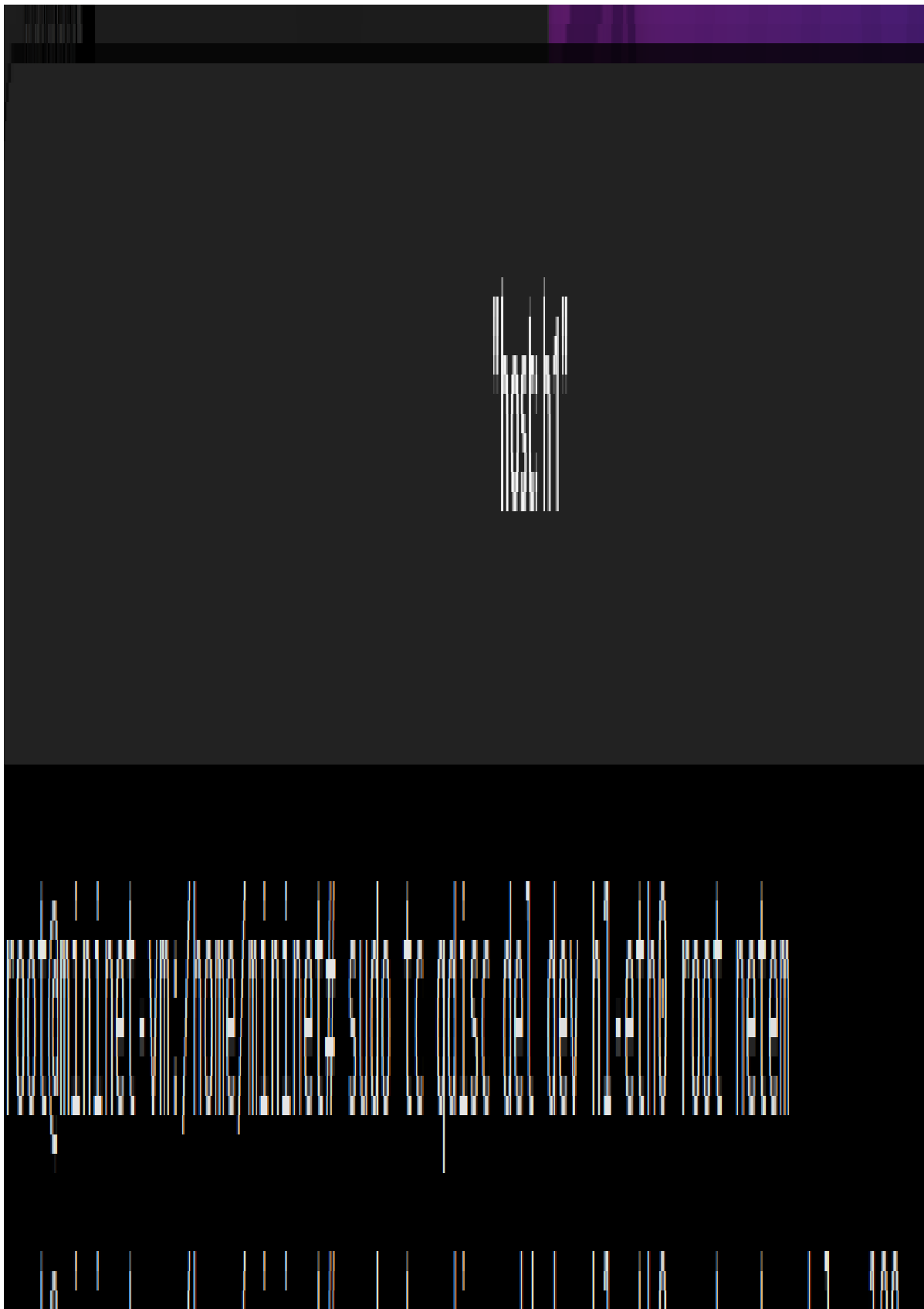


Рис. 2.16: Восстановление конфигурацию по умолчанию

17. Зададим нормальное распределение задержки на узле h1 в эмулируемой сети(рис. 2.17):

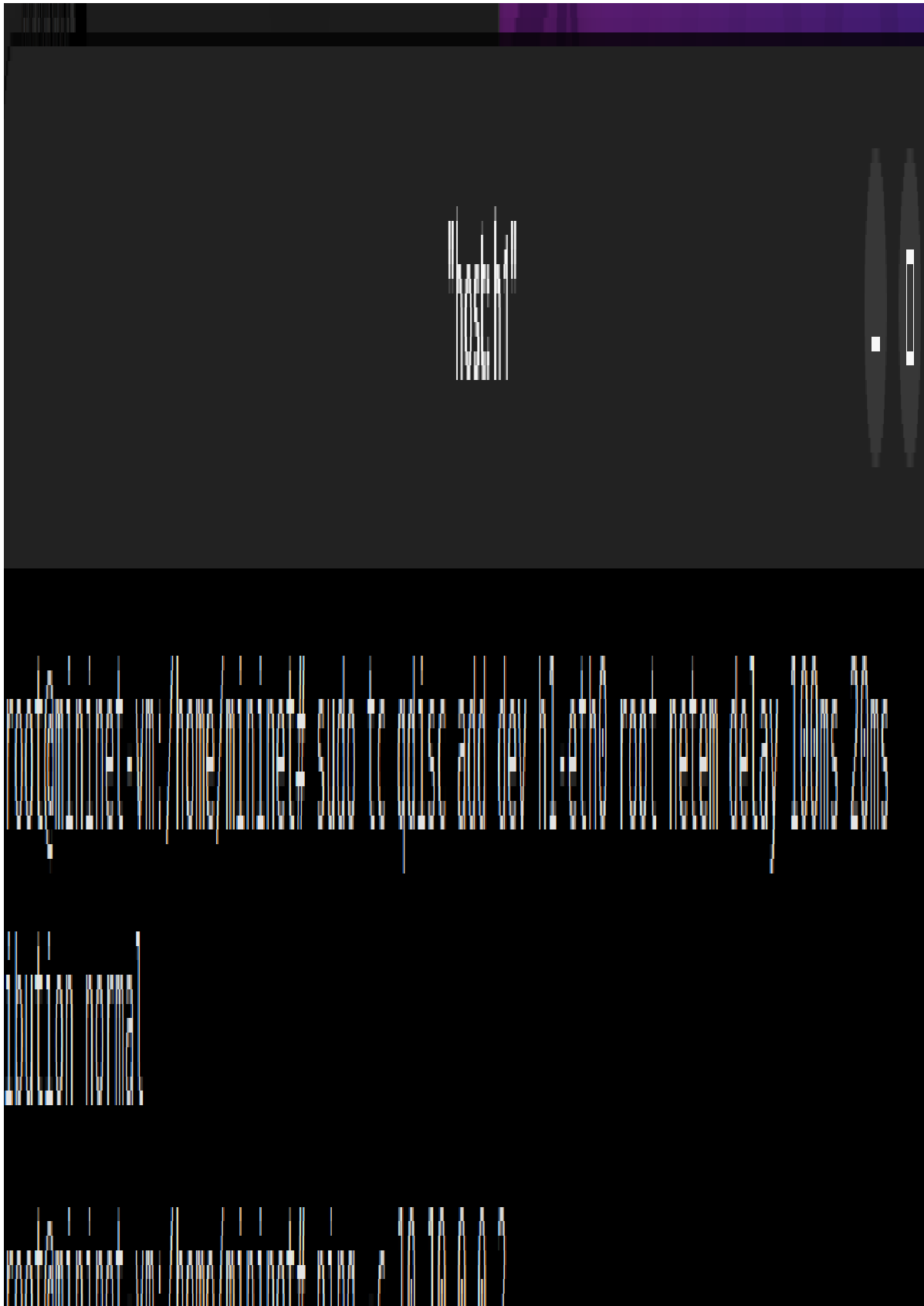
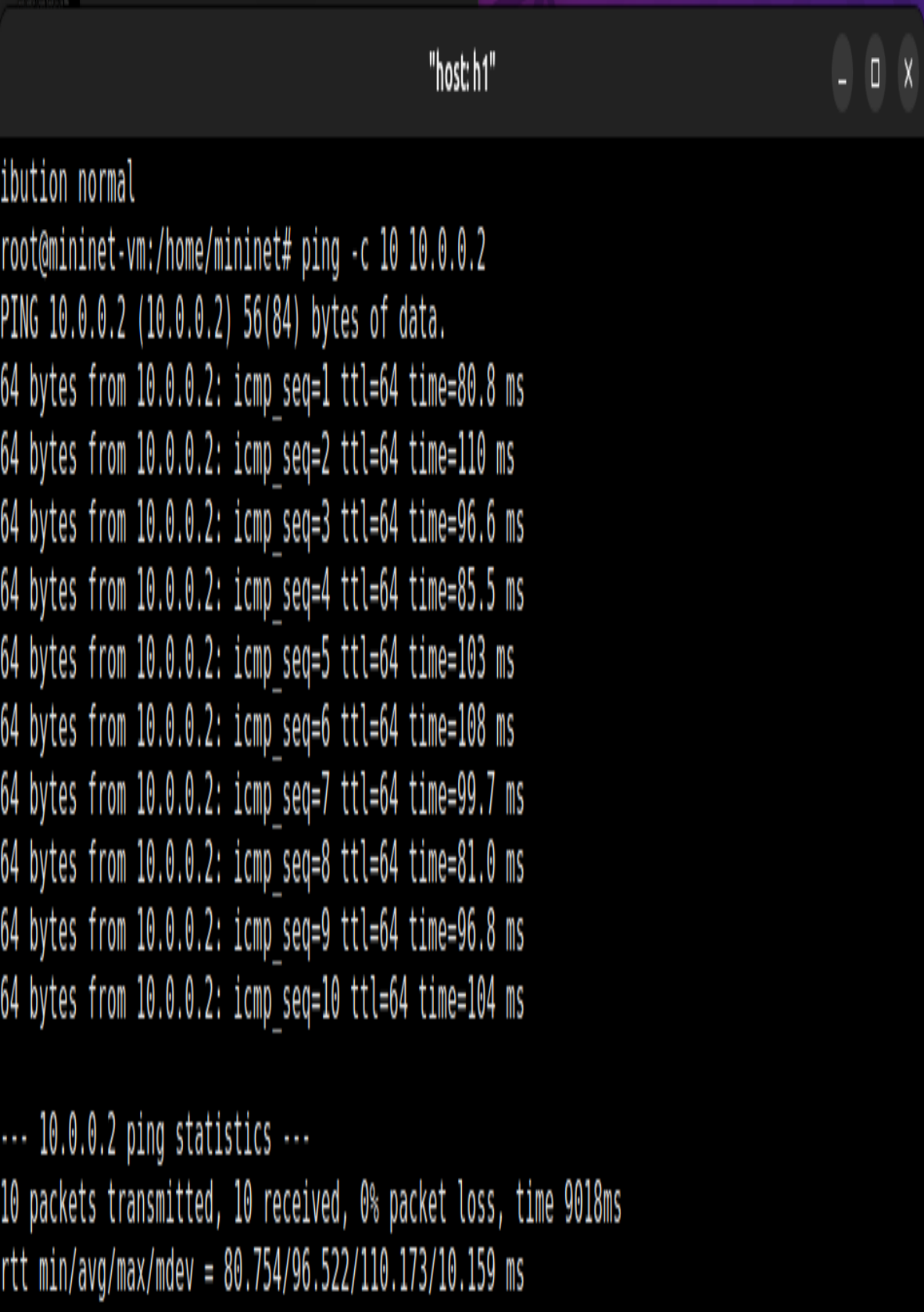


Рис. 2.17: Настройка нормального распределения задержки на узле h1 в эмулируемой сети

18. Убедимся, что все пакеты, покидающие хост h1 на интерфейсе h1-eth0, будут иметь время задержки, которое распределено в диапазоне $100 \text{ мс} \pm 20 \text{ мс}$. Используем для этого команду `ping` на терминале хоста h1 с параметром `-c 10` (рис. 2.18):



```

distribution normal
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=80.8 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=110 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=96.6 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=85.5 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=103 ms
64 bytes from 10.0.0.2: icmp_seq=6 ttl=64 time=108 ms
64 bytes from 10.0.0.2: icmp_seq=7 ttl=64 time=99.7 ms
64 bytes from 10.0.0.2: icmp_seq=8 ttl=64 time=81.0 ms
64 bytes from 10.0.0.2: icmp_seq=9 ttl=64 time=96.8 ms
64 bytes from 10.0.0.2: icmp_seq=10 ttl=64 time=104 ms

--- 10.0.0.2 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9018ms
rtt min/avg/max/mdev = 80.754/96.522/110.173/10.159 ms

```

Рис. 2.18: Проверка

19. Восстановим конфигурацию интерфейса по умолчанию на узле h1(рис. 2.19):

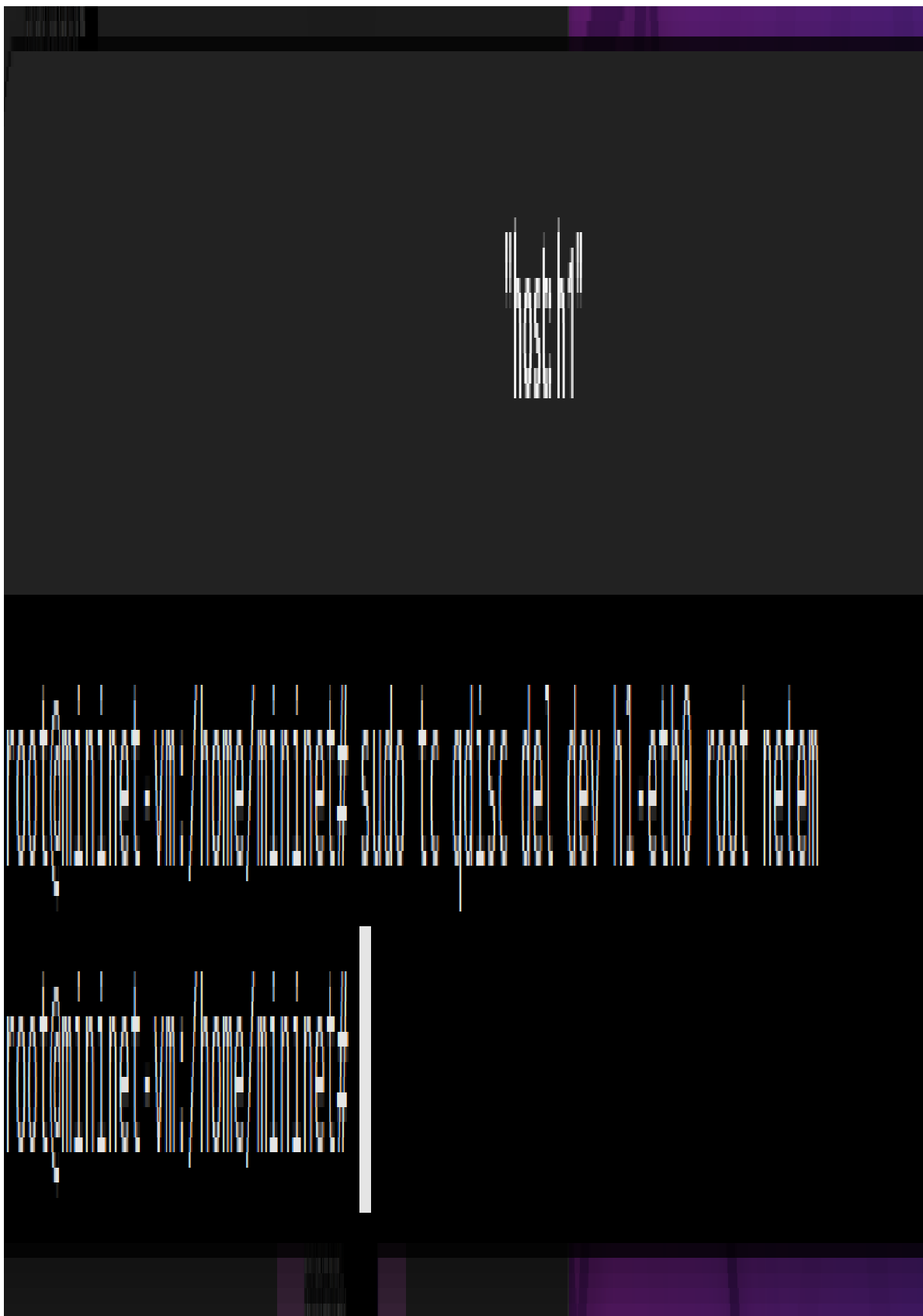


Рис. 2.19: Восстановление конфигурацию по умолчанию

20. Завершим работу mininet в интерактивном режиме(рис. 2.19):

```
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 8 terms
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
completed in 1247.365 seconds
mininet@mininet-vm:~$
```

Рис. 2.20: Завершение работу mininet в интерактивном режиме

21. Обновим репозитории программного обеспечения на виртуальной машине
(рис. 2.21):

```
mininet@mininet-vm:~$ sudo apt-get update
Get:1 http://security.ubuntu.com/ubuntu focal-security InRelease [128 kB]
Hit:2 http://us.archive.ubuntu.com/ubuntu focal InRelease
Get:3 http://us.archive.ubuntu.com/ubuntu focal-updates InRelease [128 kB]
Get:4 http://security.ubuntu.com/ubuntu focal-security/main i386 Packages [835 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu focal-backports InRelease [128 kB]
Get:6 http://security.ubuntu.com/ubuntu focal-security/main amd64 Packages [3,304 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [3,681 kB]
Get:8 http://security.ubuntu.com/ubuntu focal-security/main Translation-en [484 kB]
Get:9 http://security.ubuntu.com/ubuntu focal-security/universe amd64 Packages [1,016 kB]
Get:10 http://security.ubuntu.com/ubuntu focal-security/universe i386 Packages [683 kB]
Get:11 http://security.ubuntu.com/ubuntu focal-security/universe Translation-en [215 kB]
Get:12 http://us.archive.ubuntu.com/ubuntu focal-updates/main i386 Packages [1,056 kB]
Get:13 http://us.archive.ubuntu.com/ubuntu focal-updates/main Translation-en [564 kB]
Get:14 http://us.archive.ubuntu.com/ubuntu focal-updates/restricted amd64 Packages [3,397 kB]
Get:15 http://us.archive.ubuntu.com/ubuntu focal-updates/restricted i386 Packages [40.4 kB]
Get:16 http://us.archive.ubuntu.com/ubuntu focal-updates/restricted Translation-en [474 kB]
Get:17 http://us.archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [1,238 kB]
Get:18 http://us.archive.ubuntu.com/ubuntu focal-updates/universe i386 Packages [809 kB]
Get:19 http://us.archive.ubuntu.com/ubuntu focal-updates/universe Translation-en [297 kB]
Get:20 http://us.archive.ubuntu.com/ubuntu focal-updates/multiverse amd64 Packages [27.9 kB]
Get:21 http://us.archive.ubuntu.com/ubuntu focal-updates/multiverse Translation-en [7,968 B]
Fetched 18.5 MB in 6s (2,880 kB/s)
Reading package lists... Done
```

Рис. 2.21: Обновление репозитория ПО на ВМ

22. Установим пакет `ggee` для просмотра файлов `png` (рис. 2.22):

```
mininet@mininet-vm:~$ sudo apt install geeqie
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
acl apg apport apport-symptoms aptdaemon aptdaemon-data avahi-daemon avahi-utils bluez
bolt cheese-common colord colord-data cracklib-runtime cups-bsd cups-client cups-common
cups-pk-helper dbus dbus-x11 dconf-cli desktop-file-utils dns-root-data dnsmasq-base
docbook-xml evolution-data-server evolution-data-server-common exiftran exiv2 fprintd gcr
gdm3 geeqie-common geoclue-2.0 gir1.2-accountsservice-1.0 gir1.2-atk-1.0 gir1.2-atspi-2.0
gir1.2-freedesktop gir1.2-gck-1 gir1.2-gcr-3 gir1.2-gdesktopenums-3.0 gir1.2-gdkpixbuf-2.0
gir1.2-gdm-1.0 gir1.2-geoclue-2.0 gir1.2-gnomebluetooth-1.0 gir1.2-gnomedesktop-3.0
gir1.2-graphene-1.0 gir1.2-gtk-3.0 gir1.2-gweather-3.0 gir1.2-ibus-1.0 gir1.2-json-1.0
gir1.2-mutter-6 gir1.2-nm-1.0 gir1.2-nma-1.0 gir1.2-notify-0.7 gir1.2-packagekitglib-1.0
gir1.2-pango-1.0 gir1.2-polkit-1.0 gir1.2-rsvg-2.0 gir1.2-secret-1 gir1.2-soup-2.4
gir1.2-upowerglib-1.0 gir1.2-vte-2.91 gjs gkbd-caplet gnome-control-center
gnome-control-center-data gnome-control-center-faces gnome-keyring gnome-keyring-pkcs11
gnome-menus gnome-online-accounts gnome-session-bin gnome-session-common
gnome-settings-daemon gnome-settings-daemon-common gnome-shell gnome-shell-common
gnome-startup-applications gnome-user-docs gstreamer1.0-clutter-3.0 gstreamer1.0-gl
gstreamer1.0-plugins-good gstreamer1.0-pulseaudio gstreamer1.0-x i965-va-driver ibus
ibus-data ibus-gtk ibus-gtk3 iio-sensor-proxy im-config intel-media-va-driver ippusbxd
language-selector-common language-selector-gnome libaa1 libaacs0 libaom0
libappindicator3-1 libappstream4 libasound2-plugins libass9 libavahi-core7 libavahi-glib1
```

Рис. 2.22: Установка пакета geeqie

23. Для каждого воспроизводимого эксперимента ехрпате создадим свой каталог, в котором будут размещаться файлы эксперимента (рис. 2.23):

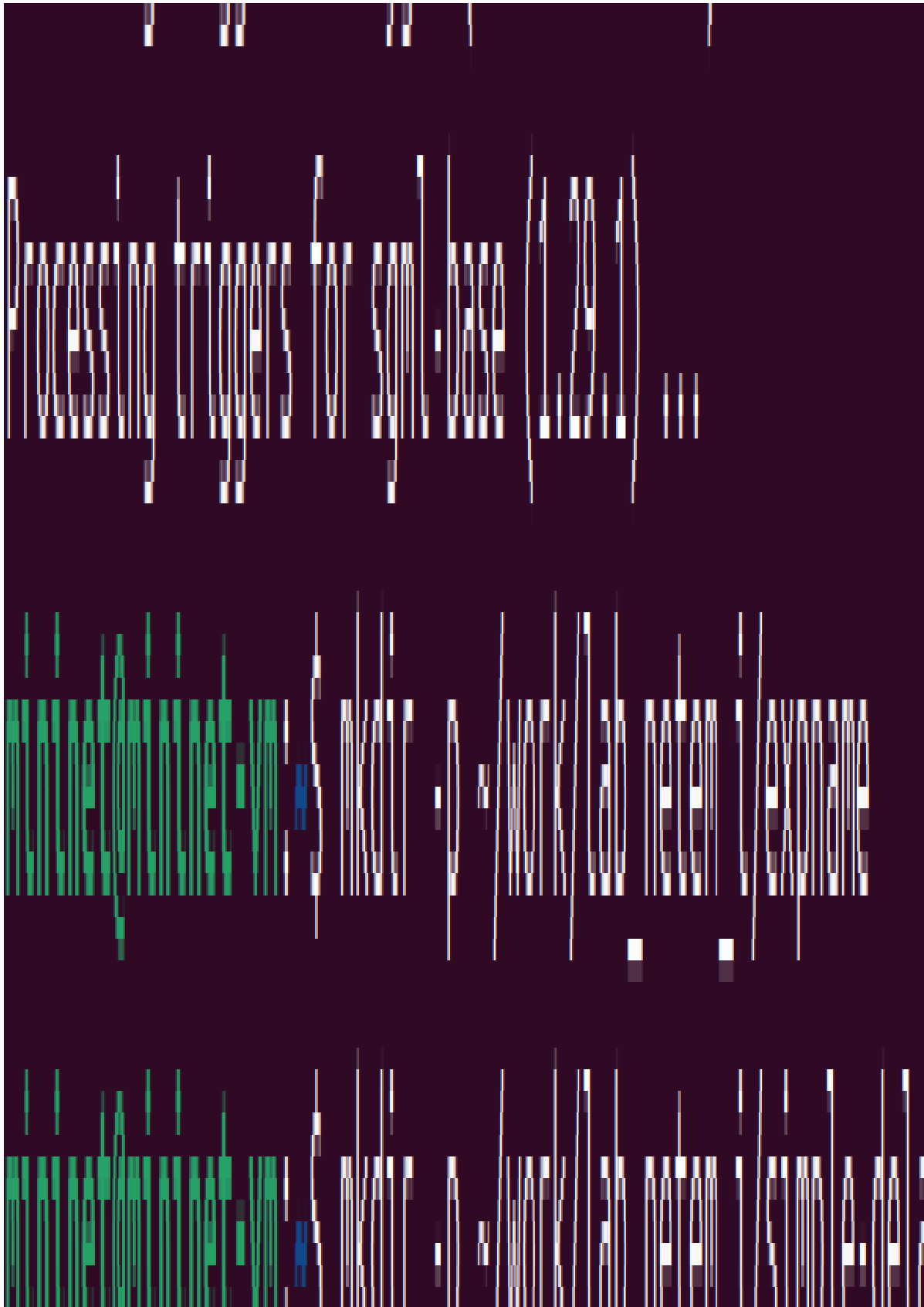


Рис. 2.23: Создание каталога

24. В виртуальной среде mininet в своём рабочем каталоге с проектами создадим каталог simple-delay и перейдём в него (рис. 2.24):

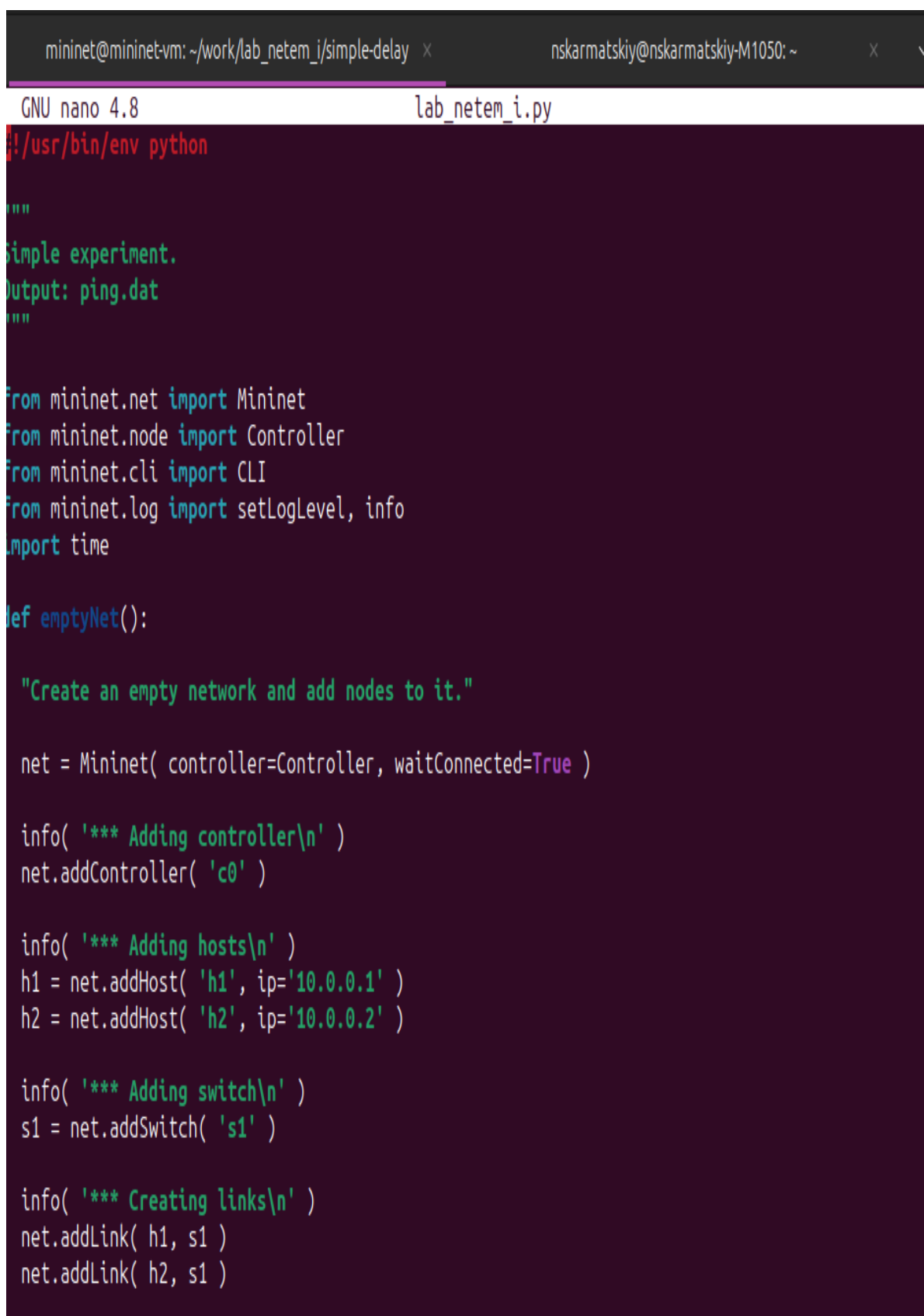
```
mininet@mininet-vm:~$ mkdir -p ~/work/lab_netem_i/simple-delay
mininet@mininet-vm:~$ cd ~/work/lab_netem_i/simple-delay
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ touch lab_netem_i.py
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano lab_netem_i.py
```

Use "fg" to return to nano.

```
[1]+  Stopped                  nano lab_netem_i.py
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano lab_netem_i.py
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano lab_netem_i.py
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano lab_netem_i.py
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ ls
lab_netem_i.py
```

Рис. 2.24: Создание каталога simple-delay

25. Создадим скрипт для эксперимента `lab_netem_i.py` (рис. 2.25):



The image shows a terminal window with two tabs. The active tab is titled 'mininet@mininet-vm: ~/work/lab_netem_i/simple-delay' and the other is 'nskarmatskiy@nskarmatskiy-M1050: ~'. The terminal is running GNU nano 4.8, editing a file named 'lab_netem_i.py'. The script content is as follows:

```
#!/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 )
```

Рис. 2.25: Создание скрипта для эксперимента lab_netem_i.py

26. Создадим файл `ping_plot` (рис. 2.26):

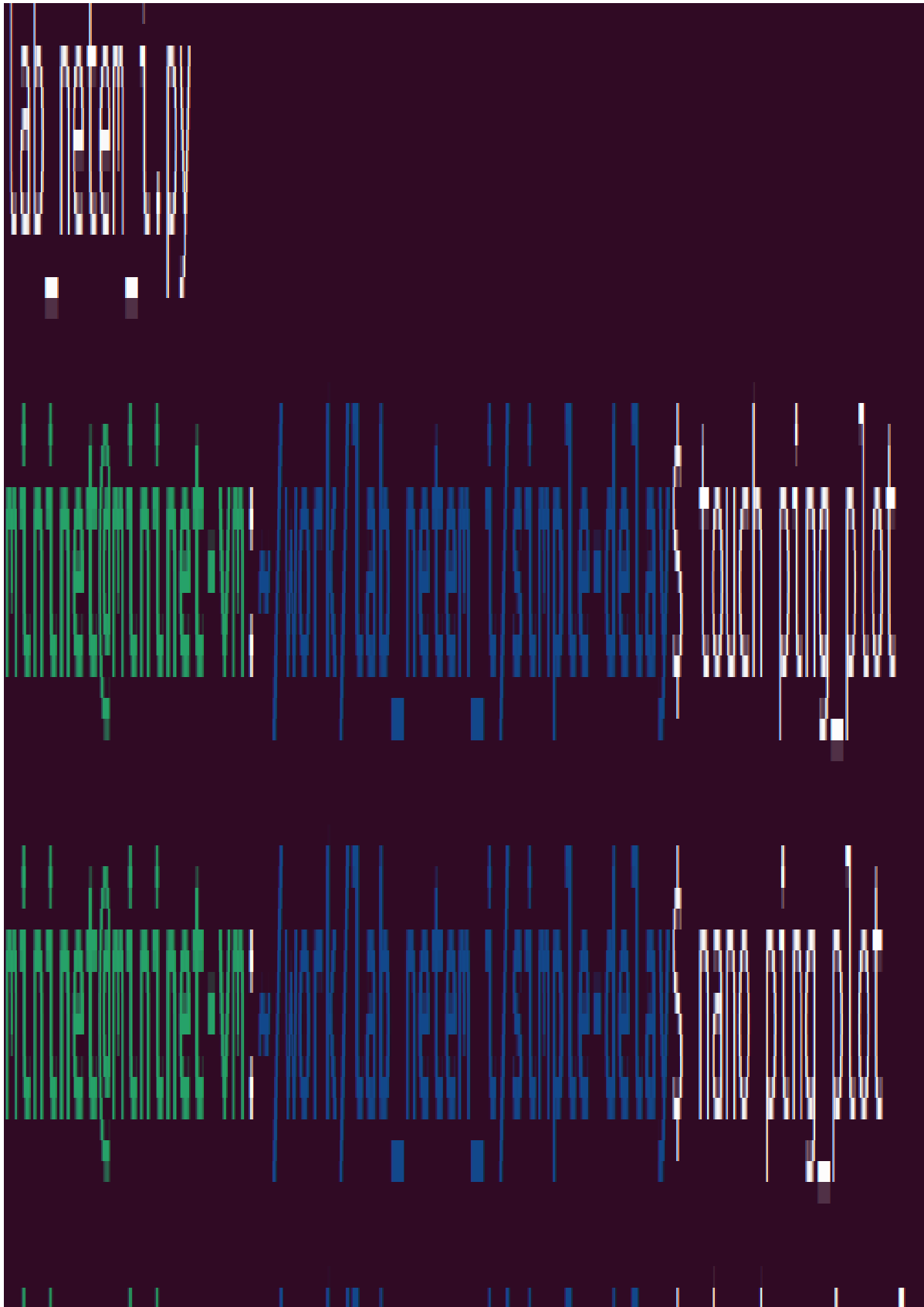


Рис. 2.26: Создание файла ping_plot

27. Затем создадим скрипт для визуализации `ring_plot` результатов эксперимента (рис. 2.27):

```
mininet@mininet-vm: ~/work/lab_netem_i/simple-delay X
```

```
GNU nano 4.8
```

```
ping
```

```
#!/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. 2.27: Создание скрипта ping_plot для визуализации результатов эксперимента

28. Зададим права доступа к файлу скрипта (рис. 2.28):

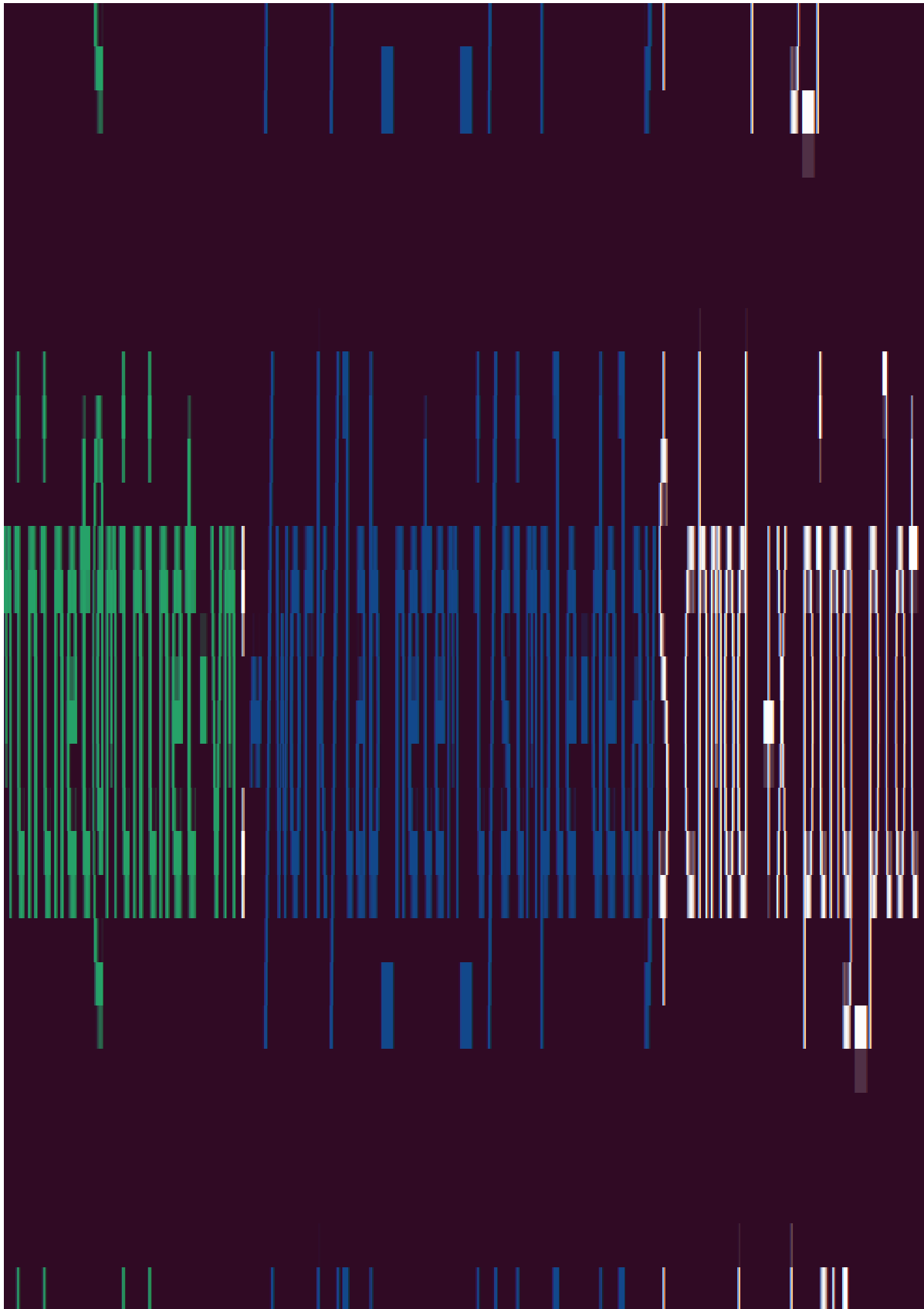


Рис. 2.28: Настройка прав доступа к файлу скрипта

29. Создадим файла Makefile (рис. 2.29):

30. Внутри файла Makefile поместим скрипт для управления процессом проведения эксперимента (рис. 2.30):

```
mininet@mininet-vm: ~/work/lab_netem_i/simple-delay x nskar
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. 2.30: Добавления скрипта в Makefile для управления процессом проведения эксперимента

31. Выполним эксперимент (рис. 2.31):

```

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, $7}\'' | sed -e \'/t
e=//g\' -e \'/icmp_seq=//g\' > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ ls

```

Рис. 2.31: Выполнение эксперимента

32. Просмотрим построенный в результате выполнения скриптов график (рис. 2.32):

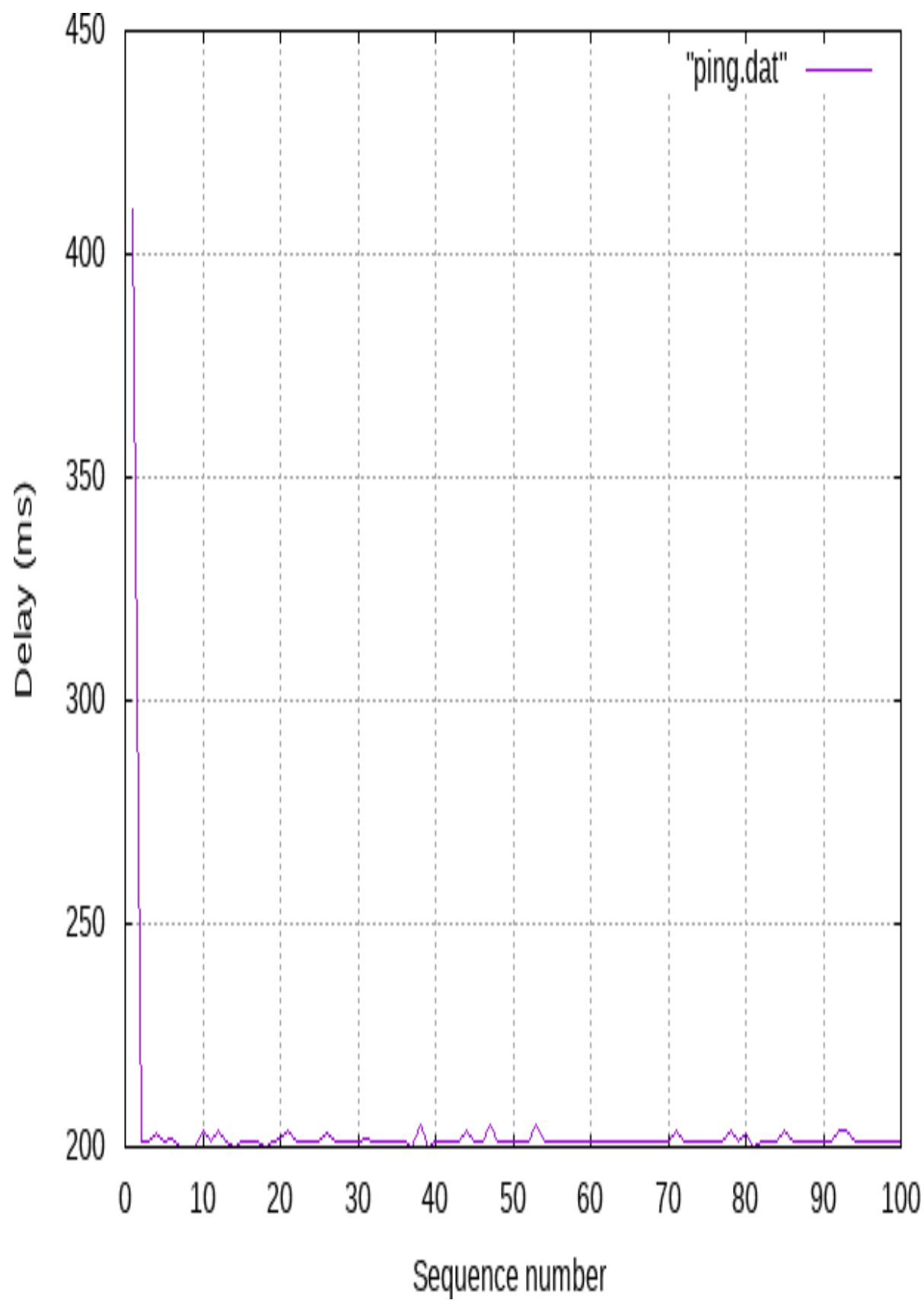


Рис. 2.32: Просмотр графика

33. Из файла `ping.dat` удалим первую строку и заново построим график (рис. 2.33 - рис. 2.34):

```
mininet@mininet-vm: ~/work/lab_netem_i/simple-delay x nskarmatskiy@nskarmatskiy-M1050: ~ x v
GNU nano 4.8 ping.dat Modified
2 202
3 201
4 202
5 201
6 201
7 202
8 201
9 201
10 201
11 201
12 200
13 201
14 201
15 201
16 201
17 201
18 201
19 201
20 201
21 201
22 201
23 201
24 201
25 201
26 201
27 201
28 201
29 201
30 201
```

Рис. 2.33: Удаление первой строки из файла ping.dat

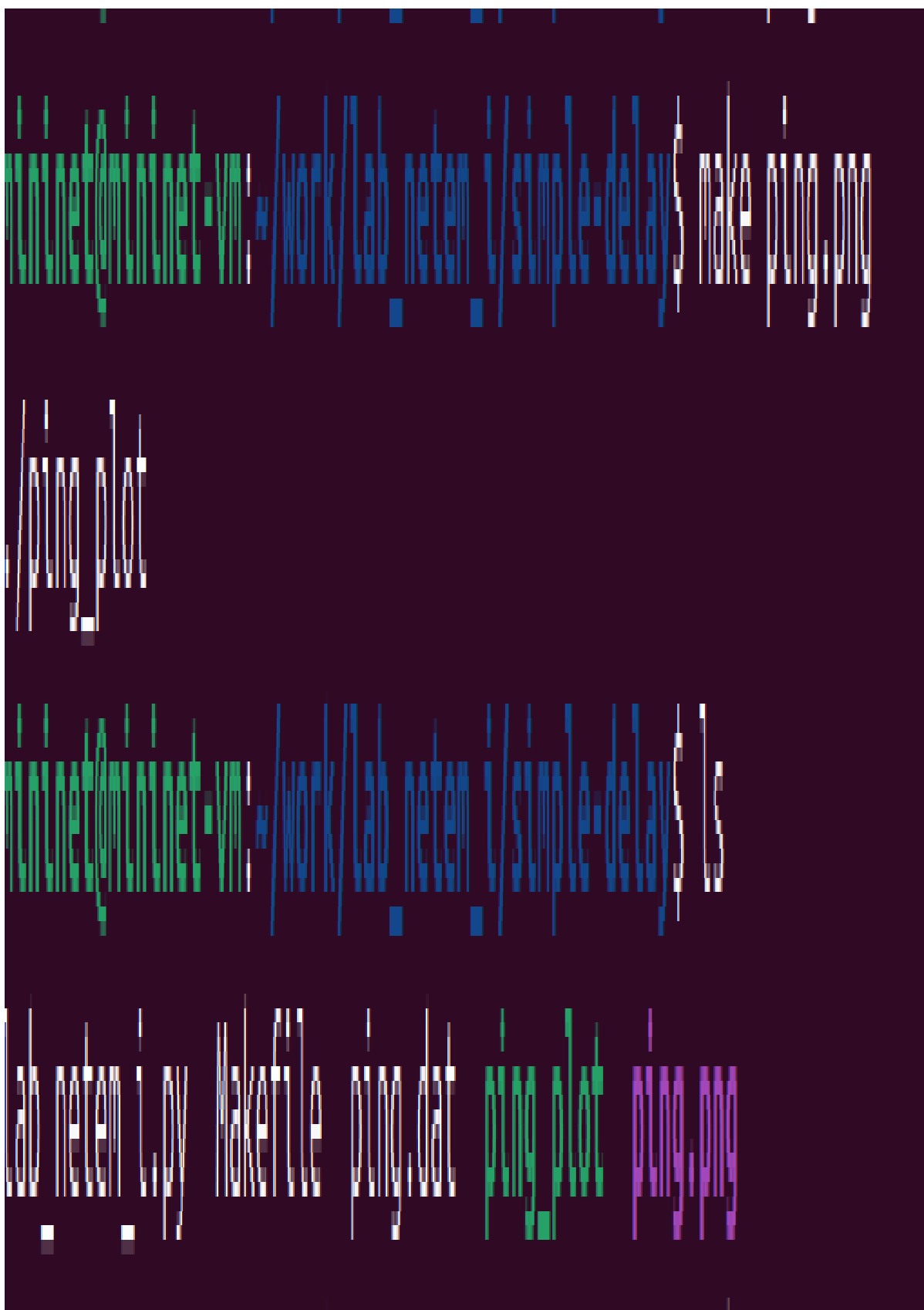


Рис. 2.34: Повторное построение графика

34. Просмотрим заново построенный график (рис. 2.35):

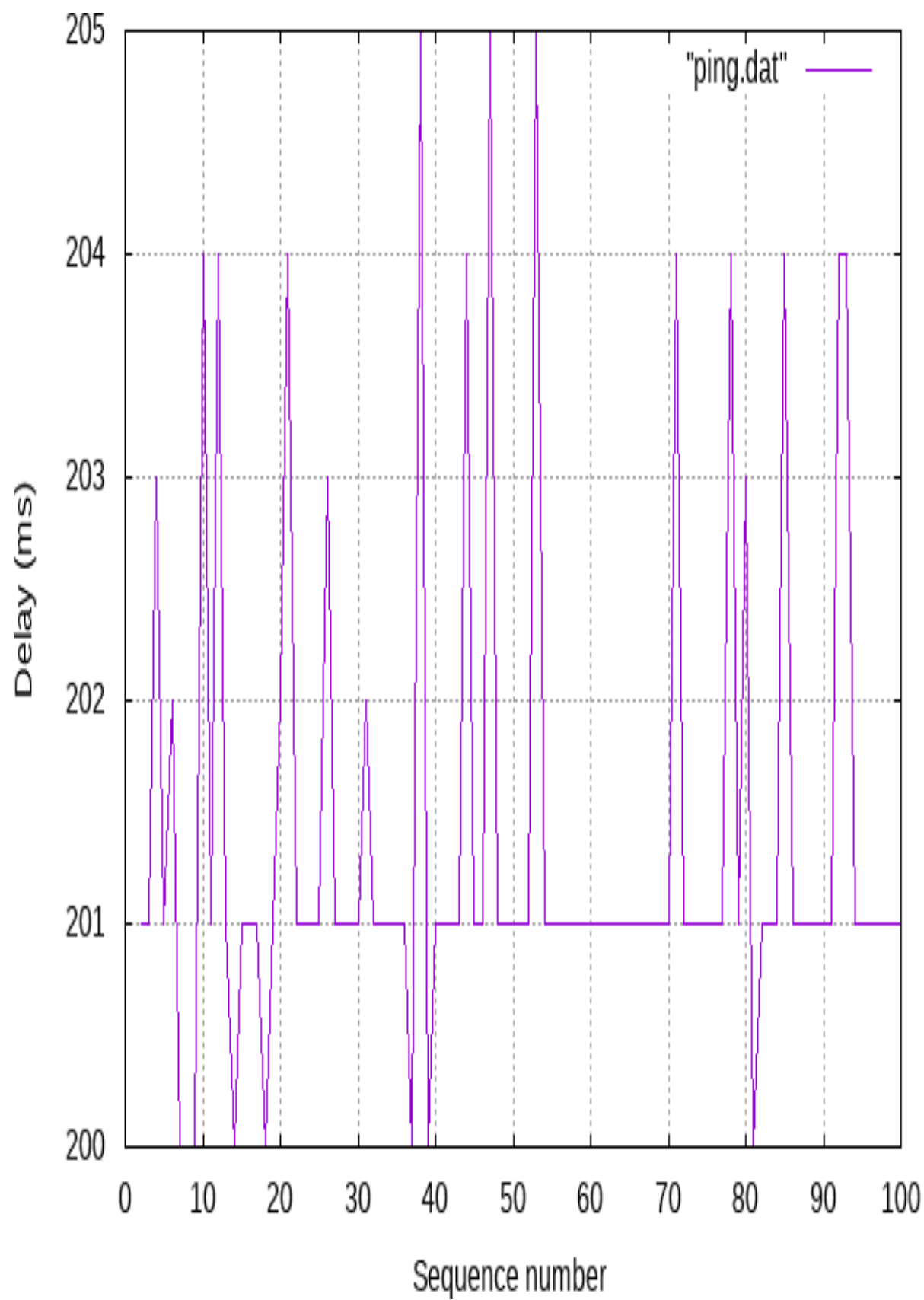


Рис. 2.35: Просмотр графика

35. Разработаем скрипт для вычисления на основе данных файла `ping.dat` минимального, среднего, максимального и стандартного отклонения времени приёма-передачи. Также добавим правило запуска скрипта в `Makefile` (рис. 2.36 - рис. 2.38):

```
mininet@mininet-vm: ~/work/lab_netem_i/simple-delay × nskarmatskiy@nskarm
GNU nano 4.8 rtr.py
import numpy as np
def calc_stat (data):

    times = np.array([float(line.split()[1]) for line in data])
    min_time = np.min(times)
    avg_time = np.mean(times)
    max_time = np.max(times)
    std_dev = np.std(times)

    return min_time, avg_time, max_time, std_dev

def read_file():

    with open('ping.dat', 'r') as file:
        data = file.readlines()
        min_time, avg_time, max_time, std_dev = calc_stat(data)

    print (f"Min time: {min_time} ms")
    print (f"Avg time: {avg_time} ms")
    print (f"Max time: {max_time} ms")
    print (f"Std dev: {std_dev} ms")

if __name__ == "__main__":
    read_file()
```

Рис. 2.36: Разработка скрипта для вычисления на основе данных файла ping.dat минимального, среднего, максимального и стандартного отклонения времени приёма-передачи 77

```
mininet@mininet-vm: ~/work/lab_netem_i/simple-delay x
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

stats: ping.dat
    python rtr.py

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

Рис. 2.37: Добавление правила запуска скрипта в Makefile

```
Processing triggers for libc-bin (2.14-0ubuntu2) ...  
miniinet@miniinet-vm:~/work/lab_netem_i/simple-delay$ make stats  
python rtr.py  
Min time: 200.0 ms  
Avg time: 201.40404040404042 ms  
Max time: 205.0 ms  
Std dev: 1.1713688092383212 ms
```

Рис. 2.38: Проверка

36. Очистим каталог от результатов проведения экспериментов.
37. 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. 2.39 - рис. 2.50):


```
"""
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 50ms' )
    h2.cmdPrint( 'tc qdisc add dev h2-eth0 root netem delay 50ms' )
```

Рис. 2.39: Воспроизводимый эксперимент по изменению задержки

```

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, $7}\'' | sed -e \'s/time=//g\' -e \'s/icmp_seq=//g\' > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make stats
python rtr.py
Min time: 100.0 ms
Avg time: 102.17 ms
Max time: 202.0 ms
Std dev: 10.098569205585509 ms
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ ls
lab_netem_i.py Makefile ping.dat ping_plot ping.png rtr.py

```

Рис. 2.40: Воспроизводимый эксперимент по изменению задержки

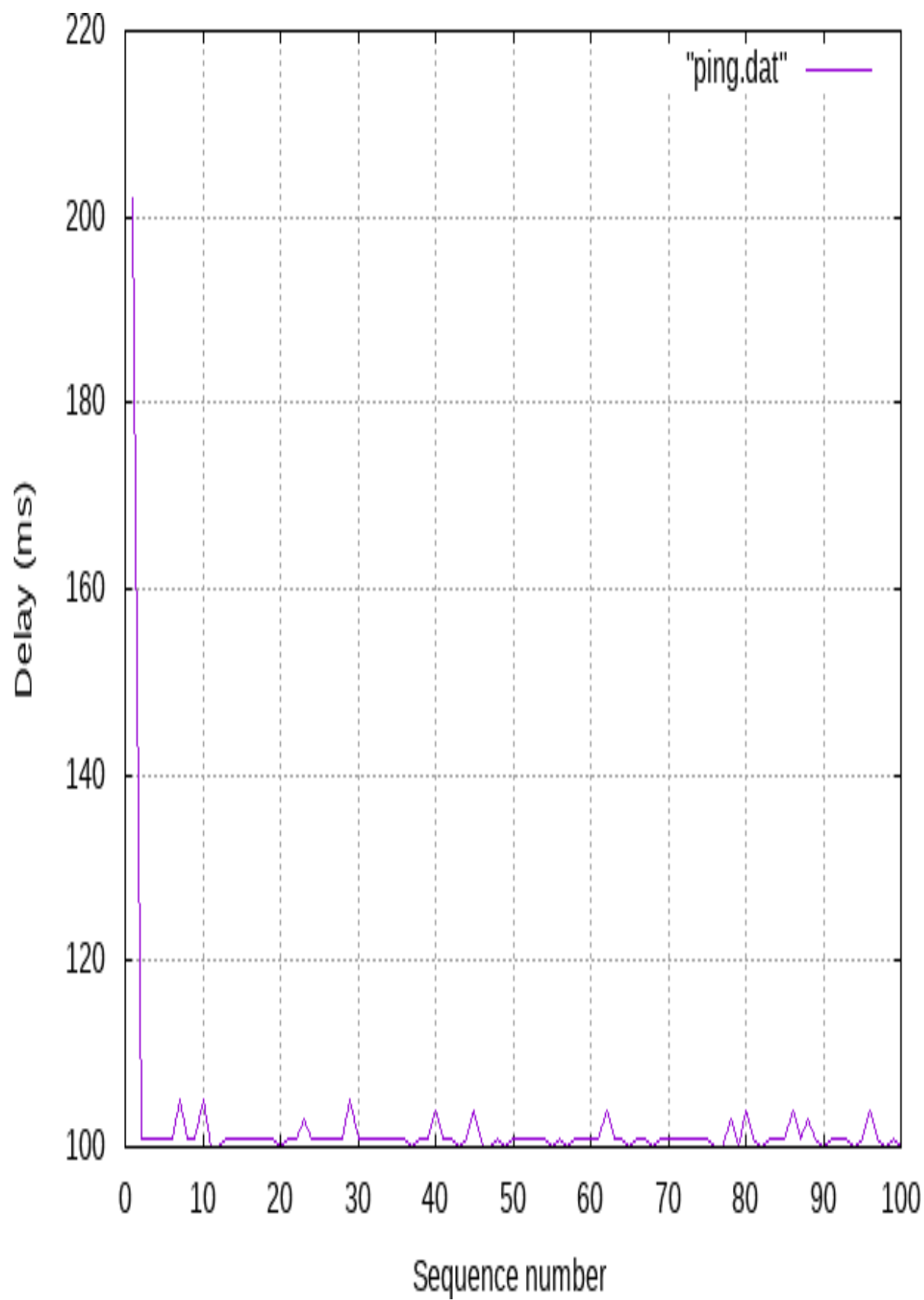


Рис. 2.41: Просмотр графика

```
GNU nano 4.8                                lab_netem_i.py                                Modified
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 10ms' )
    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/>

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

^G Get Help    ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
```

Рис. 2.42: 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 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, $7}\'' | sed -e \'s/time=//g\' -e \'s/icmp_seq=//g\' > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make stats
python rtr.py
Min time: 191.0 ms
Avg time: 202.9 ms
Max time: 409.0 ms
Std dev: 21.5 ms

```

Рис. 2.43: Воспроизводимый эксперимент по изменению джиттера

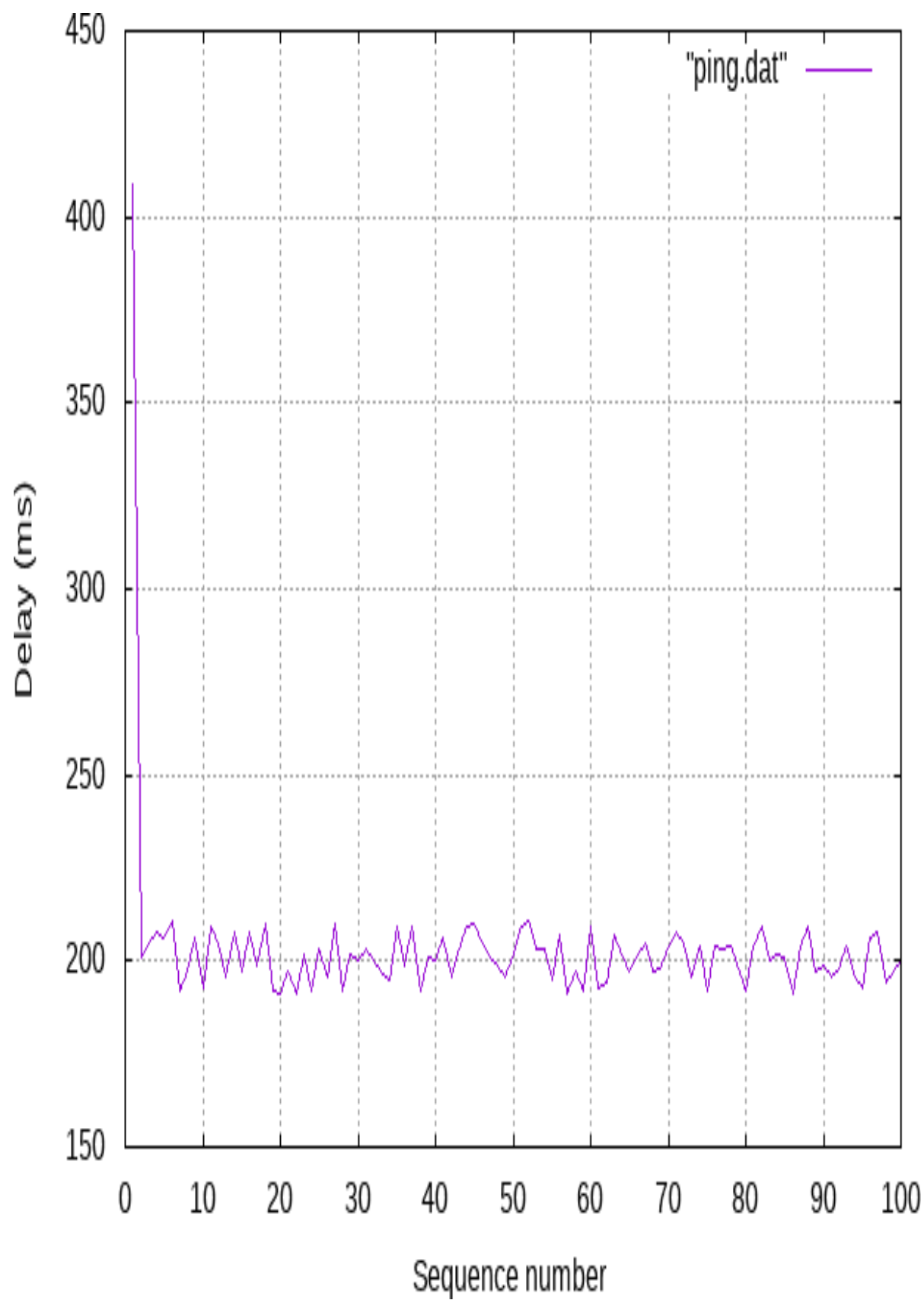


Рис. 2.44: Просмотр графика

```
GNU nano 4.8                                lab_netem_i.py                                Modified
"""

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 10ms 25%' )
    h2.cmdPrint( 'tc qdisc add dev h2-eth0 root netem delay 100ms' )

    time.sleep(10) # Wait 10 seconds

    info( '*** Ping\n' )
```

Рис. 2.45: Воспроизводимый эксперимент по изменению значения корреляции для джиттера и задержки

```

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, $7}\'' | sed -e \'/time=//g\' -e \'/icmp_seq=//g\' > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make stats
python rtr.py
Min time: 191.0 ms
Avg time: 203.46 ms
Max time: 406.0 ms
Std dev: 21.166681364824292 ms
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$

```

Рис. 2.46: Воспроизводимый эксперимент по изменению значения корреляции для джиттера и задержки

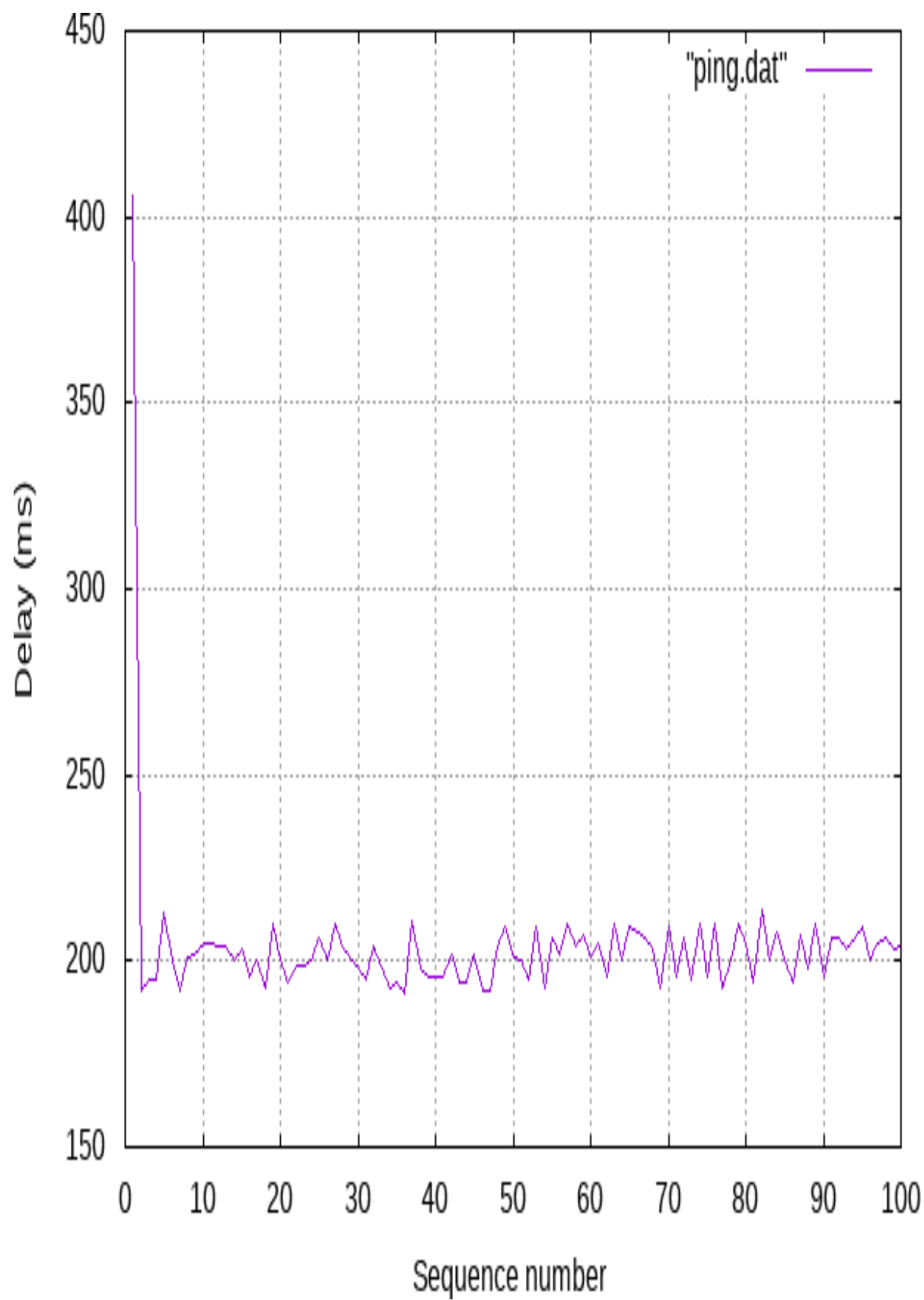


Рис. 2.47: Просмотр графика

```
GNU nano 4.8                                lab_netem_i.py
"""

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 10ms 25% distribution normal' )
    h2.cmdPrint( 'tc qdisc add dev h2-eth0 root netem delay 100ms' )

    time.sleep(10) # Wait 10 seconds

    info( '*** Ping\n' )

[ Wrote 51 lines ]
^G Get Help    ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
```

Рис. 2.48: Воспроизводимый эксперимент по изменению распределения времени задержки в эмулируемой глобальной сети

```

mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ nano lab_netem_i.py
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 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, $7}\'' | sed -e \'/time
g\' -e \'/icmp_seq=//g\' > ping.dat')
*** Stopping network*** Stopping 1 controllers
c0
*** Stopping 2 links
..
*** Stopping 1 switches
s1
*** Stopping 2 hosts
h1 h2
*** Done
sudo chown mininet:mininet ping.dat
./ping_plot
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$ make stats
python rtr.py
Min time: 179.0 ms
Avg time: 203.25 ms
Max time: 386.0 ms
Std dev: 21.171856319179952 ms
mininet@mininet-vm:~/work/lab_netem_i/simple-delay$

```

Рис. 2.49: Воспроизводимый эксперимент по изменению распределения времени задержки в эмулируемой глобальной сети

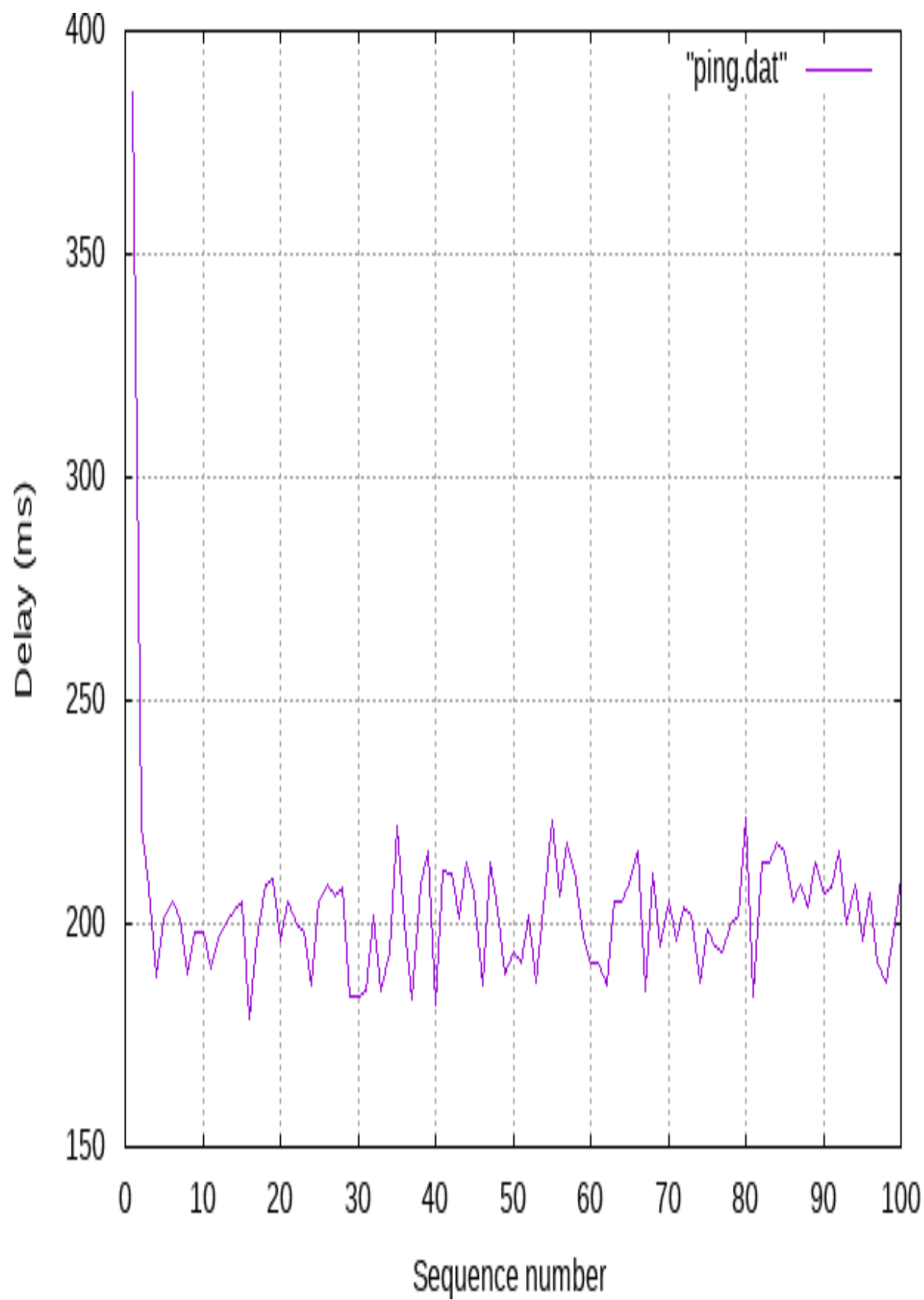


Рис. 2.50: Просмотр графика

3 Вывод

В ходе выполнения лабораторной работы познакомились с NETEM — инструментом для тестирования производительности приложений в виртуальной сети, а также получили навыки проведения интерактивного и воспроизводимого экспериментов по измерению задержки и её дрожания (jitter) в моделируемой сети в среде Mininet.

4 Список литературы. Библиография

[1] Mininet: <https://mininet.org/>