Лабораторная работа №5

Эмуляция и измерение потерь пакетов в глобальных сетяx

Тазаева Анастасия Анатольевна

Содержание

# 1 Цель работы

Освоить моделирование следующих параметров сети: потеря пакетов, дублирование пакетов, изменение порядка и повреждение пакетов.

# 2 Задание

1. Задайте простейшую топологию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8.
2. Проведите интерактивные эксперименты по по исследованию параметров сети, связанных с потерей, дублированием, изменением порядка и повреждением пакетов при передаче данных.
3. Реализуйте воспроизводимый эксперимент по добавлению правила отбрасывания пакетов в эмулируемой глобальной сети. На экран выведите сводную информацию о потерянных пакетах.
4. Самостоятельно реализуйте воспроизводимые эксперименты по иссле- дованию параметров сети, связанных с потерей, изменением порядка и повреждением пакетов при передаче данных. На экран выведите сводную информацию о потерянных пакетах.

# 3 Выполнение лабораторной работы

## 3.1 Запуск лабораторной топологии

1. Запустила виртуальную среду с mininet. Из основной ОС подключилась к виртуальной машине (рис. 1). Исправила права запуска X-соединения (рис. 1)

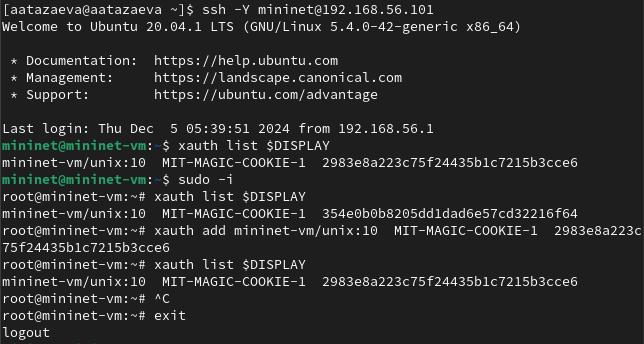


Рис. 1: Подключение к mininet. Исправление прав запуска X-соединения

1. Создала простейшую тополонию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8 (рис. 2). Терминалы коммутатора и контроллера закрыла.

sudo mn --topo=single,2 -x

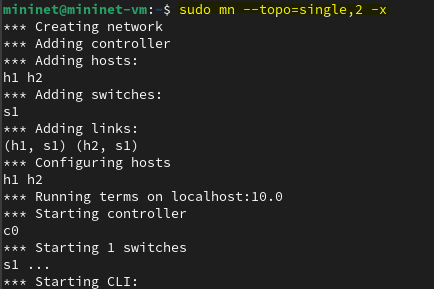


Рис. 2: Создание топологии

1. На хостах h1 и h2 ввела команду ifconfig (рис. 3 и 4), 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 .

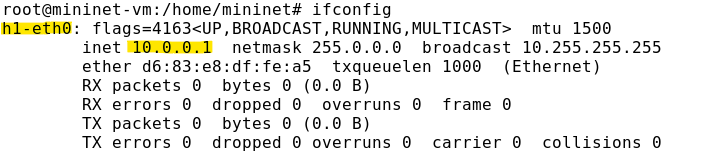


Рис. 3: ifconfig h1

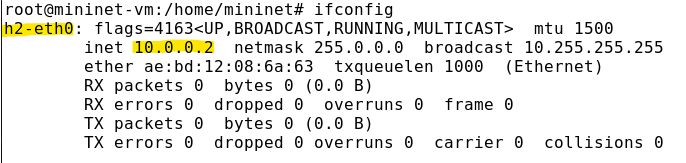


Рис. 4: ifconfig h2

1. Проверила подключение между хостами h1 и h2 с помощью команды ping с параметром -c 6 (рис. 5). На этом же рисунке выделила цветом значения всеъ возможных отклонений времени приёма-передачи.

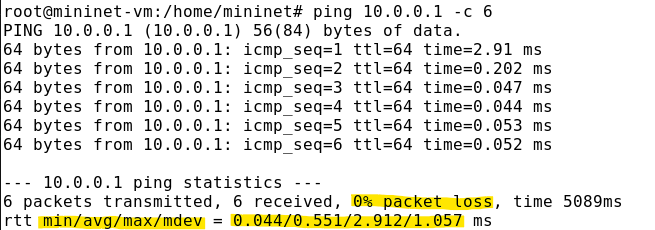


Рис. 5: Проверка подключения ping h1 -> h2

## 3.2 Интерактивные эксперименты

### 3.2.1 Добавление потери пакетов на интерфейс, подключённый к эмулируемой глобальной сети

1. На хосте h1 добавила 10% потерь пакетов к интерфейсу h1-eth0 (рис. 6):

sudo tc qdisc add dev h1-eth0 root netem delay loss 10%

Рис. 6: Потери пакетов для хоста h1

Рис. 6: Потери пакетов для хоста h1

Здесь: - sudo : выполнить команду с более высокими привилегиями; - tc : вызвать управление трафиком Linux; - qdisc : изменить дисциплину очередей сетевого планировщика; - add : создать новое правило; - dev h1-eth0 : указать интерфейс, на котором будет применяться правило; - netem : использовать эмулятор сети; - loss 10% : 10% потерь пакетов.

1. Проверила, что на соединении от хоста h1 к хосту h2 имеются потери пакетов, используя команду ping с параметром -c 100 с хоста h1 (рис. 7). Некоторые номера оследовательности отсутствуют из-за потери пакетов - выделены синим (рис. 7). В сводном отчёте ping сообщает о проценте потерянных пакетов после завершения передачи - 12% потерянных пакетов (рис. 8).

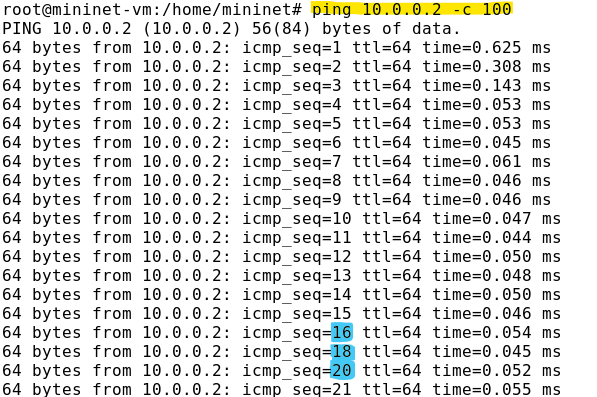


Рис. 7: Потери пакетов для хоста h1. Проверка

Рис. 8: Потери пакетов для хоста h1. Проверка. Результат

Рис. 8: Потери пакетов для хоста h1. Проверка. Результат

1. Для эмуляции глобальной сети с потерей пакетов в обоих направлениях необходимо к соответствующему интерфейсу на хосте h2 также добавить 10% потерь пакетов (рис. 9):

sudo tc qdisc add dev h2-eth0 root netem loss 10%

Рис. 9: Потери пакетов для хоста h2

Рис. 9: Потери пакетов для хоста h2

1. Проверила, что соединение между хостом h1 и хостом h2 имеет больший процент потерянных данных (10% от хоста h1 к хосту h2 и 10% от хоста h2 к хосту h1 ), повторив команду ping с параметром -c 100 на терминале хоста h1 (рис. 10). Отсутствующие из-за потери пакетов номера последовательности (4, 13, 18, 24, 27, 28, 34, 36, 45, 59,67, 72, 74, 77, 96, 100), процент потерянных пакетов после завершения передачи - 16%.

Рис. 10: Потеря пакетов для двух хостов. Проверка

Рис. 10: Потеря пакетов для двух хостов. Проверка

1. Восстановила конфигурацию по умолчанию, удалив все правила, применённые к сетевому планировщику соответствующего интерфейса. Для отправителя h1 (рис. 11):

sudo tc qdisc del dev h1-eth0 root netem

Для получателя h2 (рис. 12):

sudo tc qdisc del dev h2-eth0 root netem

Рис. 11: Удаление задержки для хоста h1. Проверка

Рис. 11: Удаление задержки для хоста h1. Проверка

Рис. 12: Удаление задержки для хоста h2

Рис. 12: Удаление задержки для хоста h2

1. Проверила, что соединение между хостом h1 и хостом h2 не имеет явноустановленной задержки, используя команду ping с параметром -c 6 с терминала хоста h1 (рис. 13 и 12).

Рис. 13: Задержка для хоста h1 после удаления задержки. Результат

Рис. 13: Задержка для хоста h1 после удаления задержки. Результат

### 3.2.2 Добавление значения корреляции для потери пакетов в эмулируемой глобальной сети

1. Добавила на интерфейсе узла h1 коэффициент потери пакетов 50% (такой высокий уровень потери пакетов маловероятен), и каждая последующая вероятность зависит на 50% от последней (рис. 14):

sudo tc qdisc add dev h1-eth0 root netem loss 50% 50%

Рис. 14: Значение корреляции для потери пакетов для хоста h1

Рис. 14: Значение корреляции для потери пакетов для хоста h1

1. Проверила, что на соединении от хоста h1 к хосту h2 имеются потери пакетов, используя команду ping с параметром -c 50 с хоста h1 . Отсутствующие из-за потери пакетов номера последовательности (17, 27, 29, 30, 31, 32, 33, 36, 42, 43, 44, 50), процент потерянных пакетов после завершения передачи - 24% (рис. 15).

Рис. 15: Значение корреляции для потери пакетов для хоста h1. Проверка

Рис. 15: Значение корреляции для потери пакетов для хоста h1. Проверка

1. Восстановила для узла h1 конфигурацию по умолчанию, удалив все правила, применённые к сетевому планировщику соответствующего интерфейса:

sudo tc qdisc del dev h1-eth0 root netem

### 3.2.3 Добавление повреждения пакетов в эмулируемой глобальной сети

1. Добавила на интерфейсе узла h1 0,01% повреждения пакетов (рис. 16):

sudo tc qdisc add dev h1-eth0 root netem corrupt 0.01%

Рис. 16: Повреждение пакетов для хоста h1

Рис. 16: Повреждение пакетов для хоста h1

1. Проверила конфигурацию с помощью инструмента iPerf3 для проверки повторных передач. Для этого:

* запустила iPerf3 в режиме сервера в терминале хоста h2 (рис. 17):

iperf3 -s

* запустила iPerf3 в клиентском режиме в терминале хоста h1 (рис. 18):

iperf3 -c 10.0.0.2

Рис. 17: Повреждение пакетов для хоста h1. Проверка. Запуск сервера

Рис. 17: Повреждение пакетов для хоста h1. Проверка. Запуск сервера

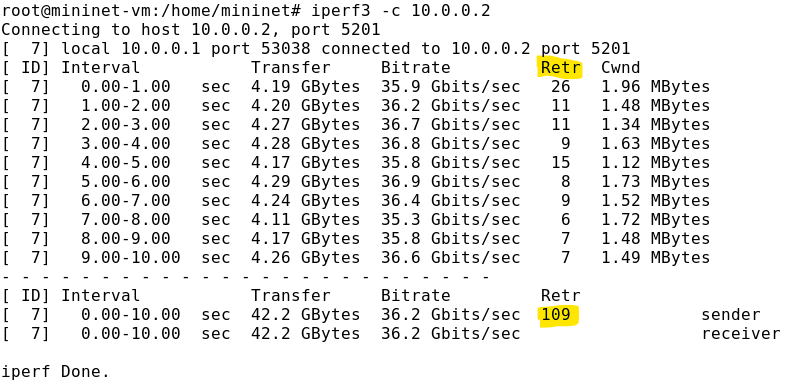


Рис. 18: Повреждение пакетов для хоста h1. Проверка. Запуск клиента

* общее количество повторно переданных пакетов указано в поле Retr (рис. 18).

1. Восстановила для узла h1 конфигурацию по умолчанию, удалив все правила, применённые к сетевому планировщику соответствующего интерфейса.

### 3.2.4 Добавление переупорядочивания пакетов в интерфейс подключения к эмулируемой глобальной сети

1. Добавила на интерфейсе узла h1 следующее правило (рис. 19):

sudo tc qdisc add dev h1-eth0 root netem delay 10ms reorder 25% 50%

Здесь 25% пакетов (со значением корреляции 50%) будут отправлены немедленно, а остальные 75% будут задержаны на 10 мс.

Рис. 19: Переупорядочивание пакетов для хоста h1

Рис. 19: Переупорядочивание пакетов для хоста h1

1. Проверила, что на соединении от хоста h1 к хосту h2 имеются потери пакетов, используя команду ping с параметром -c 20 с хоста h1 . Часть пакетов не имеют задержки, а последующие несколько пакетов будут иметь задержку около 10 миллисекунд, выделила цветом (рис. 20).

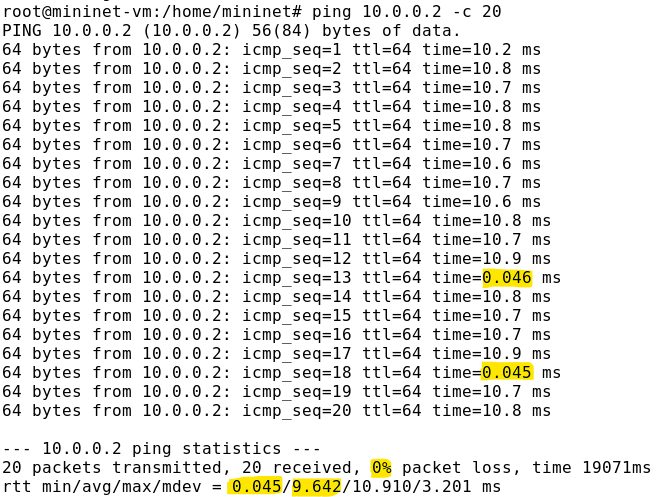


Рис. 20: Переупорядочивание пакетов для хоста h1. Проверка

1. Восстановила для узла h1 конфигурацию по умолчанию, удалив все правила, применённые к сетевому планировщику соответствующего интерфейса.

### 3.2.5 Добавление дублирования пакетов в интерфейс подключения к эмулируемой глобальной сети

1. Для интерфейса узла h1 задала правило c дублированием 50% пакетов (т.е. 50% пакетов должны быть получены дважды) (рис. 21):

sudo tc qdisc add dev h1-eth0 root netem duplicate 50%

Рис. 21: Дублирование пакетов для хоста h1

Рис. 21: Дублирование пакетов для хоста h1

1. Проверила, что на соединении от хоста h1 к хосту h2 имеются дублированные пакеты, используя команду ping с параметром -c 20 с хоста h1 (рис. 22). Дубликаты пакетов помечаются как DUP! . Измеренная скорость дублирования пакетов будет приближаться к настроенной скорости по мере выполнения большего количества попыток.

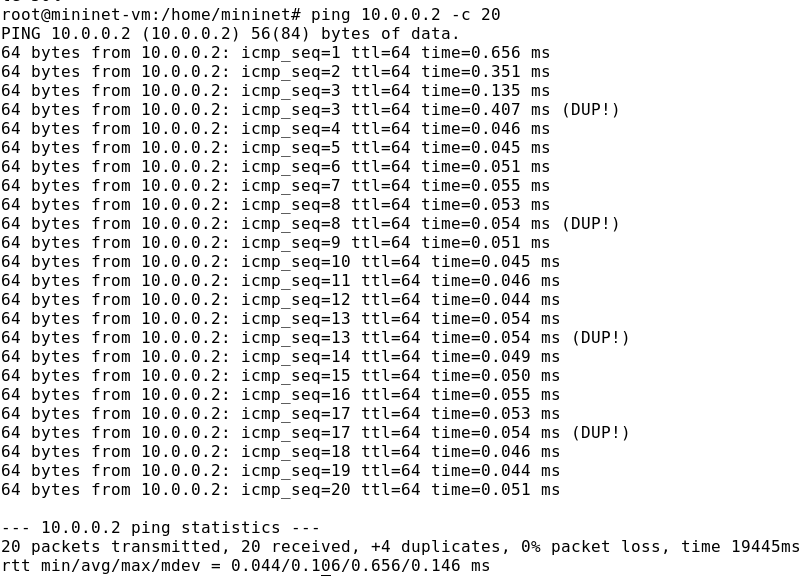


Рис. 22: Дублирование пакетов для хоста h1. Проверка

1. Восстановила для узла h1 конфигурацию по умолчанию, удалив все правила, применённые к сетевому планировщику соответствующего интерфейса.

## 3.3 Воспроизведение экспериментов

### 3.3.1 Предварительная подготовка

1. Для каждого воспроизводимого эксперимента expname создала свой каталог, в котором будут размещаться файлы эксперимента (рис. 23):

mkdir -p ~/work/lab\_netem\_ii/expname

Рис. 23: Каталог для размещения файлов эксперимента

Рис. 23: Каталог для размещения файлов эксперимента

### 3.3.2 Добавление потери пакетов на интерфейс, подключённый к эмулируемой глобальной сети

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

1. В виртуальной среде mininet в своём рабочем каталоге с проектами создала каталог simple-drop и перешла в него. Создала скрипт для эксперимента lab\_netem\_ii.py (рис. 24). Скорректировала скрипт так, чтобы в отдельный файл выводилась информация о потерях пакетов.

#!/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' )  
 net.start()  
  
 info('\*\*\*set loss\n' )  
 h1.cmdPrint( 'tc qdisc add dev h1-eth0 root netem loss 10%' )  
 h2.cmdPrint( 'tc qdisc add dev h2-eth0 root netem loss 10%' )  
  
 time.sleep(10)  
  
 info('\*\*\*ping\n' )  
 h1.cmdPrint( 'ping -c 100', h2.IP(), '| grep "packet loss" | awk \'{print $6, $7, $8}\' > ping.dat' )  
  
 info('\*\*\*stopping network' )  
 net.stop()  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 setLogLevel( 'info' )  
 emptyNet()

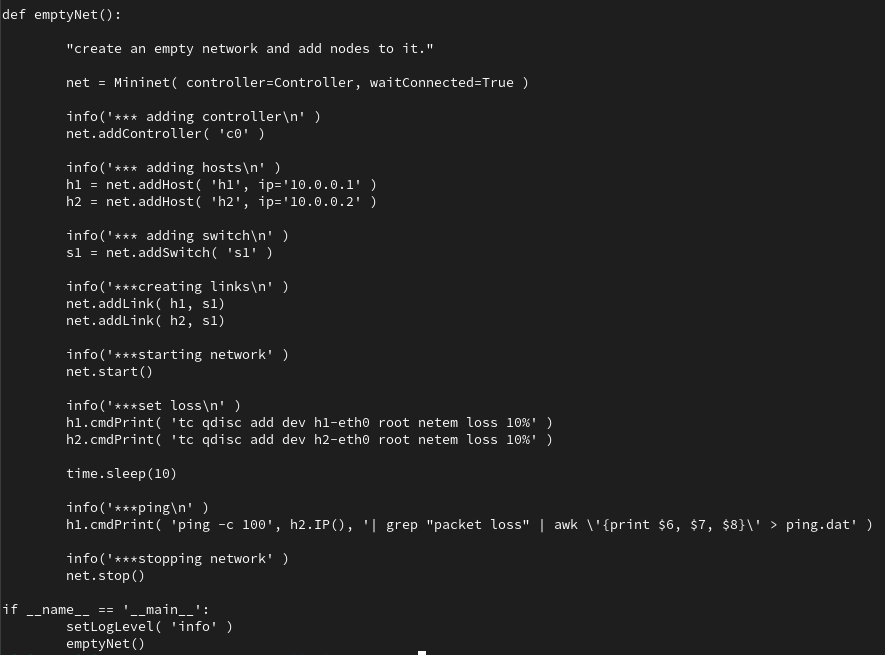


Рис. 24: lab\_netem\_ii.py

-В каких строках скрипта задается значение потери пакетов для интерфейса хоста?

h1.cmdPrint( 'tc qdisc add dev h1-eth0 root netem loss 10%' )  
 h2.cmdPrint( 'tc qdisc add dev h2-eth0 root netem loss 10%' )

1. Создала Makefile для управления процессом проведения эксперимента (рис. 25).

all: ping.dat  
  
ping.dat:  
 sudo python lab\_netem\_ii.py  
 sudo chown mininet:mininet ping.dat  
  
clean:  
 -rm -f \*.dat

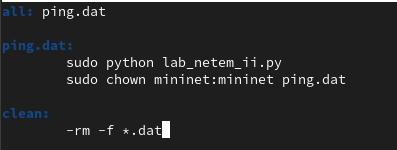


Рис. 25: Makefile

1. Выполнила эксперимент (рис. 26).

make

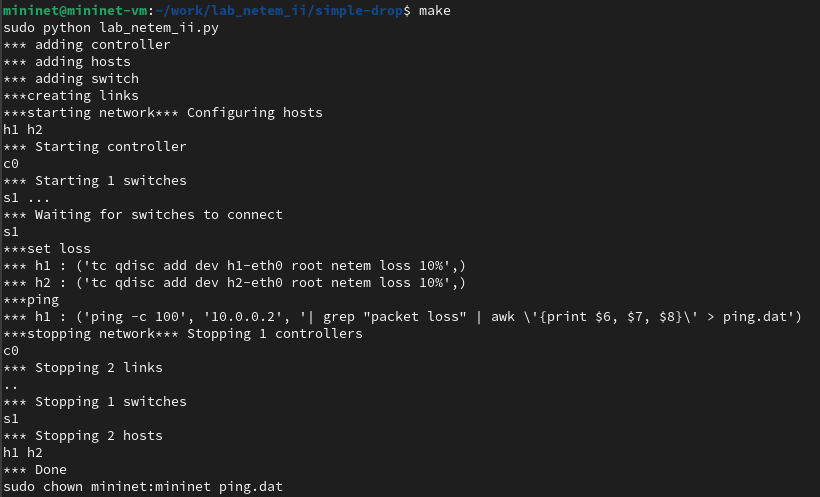


Рис. 26: Make

1. В файл ping.dat вывелась информация о потерянных пакетах в % (рис. 27).

Рис. 27: Результат скрипта

Рис. 27: Результат скрипта

1. Очистила каталог от результатов проведения экспериментов:

make clean

## 3.4 Самостоятельная работа

### 3.4.1 Добавление дублирования пакетов в интерфейс подключения к эмулируемой глобальной сети

1. Создала скрипт /dup/lab\_netem\_ii.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' )  
 net.start()  
  
 info('\*\*\*set loss\n' )  
 h1.cmdPrint( 'tc qdisc add dev h1-eth0 root netem duplicate 50%' )  
  
 time.sleep(10)  
  
 info('\*\*\*ping\n' )  
 h1.cmdPrint( 'ping -c 100', h2.IP(), '| grep "duplicates" | awk \'{print "for ", $1, $2, " - ", $6, "duplacates"}\' > ping.dat' )  
  
 info('\*\*\*stopping network' )  
 net.stop()  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 setLogLevel( 'info' )  
 emptyNet()

1. Результат скрипта выводится в файл /dup/ping.dat (рис. 28).

Рис. 28: Результат скрипта /dup/lab_netem_ii.py

Рис. 28: Результат скрипта /dup/lab\_netem\_ii.py

### 3.4.2 Добавление повреждения пакетов в эмулируемой глобальной сети

1. Создала скрипт /corrupt/lab\_netem\_ii.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' )  
 net.start()  
  
 info('\*\*\*set loss\n' )  
 h1.cmdPrint( 'tc qdisc add dev h1-eth0 root netem corrupt 0.01%' )  
 time.sleep(10)  
  
 info('\*\*\*ping\n' )  
 h2.cmdPrint( 'iperf3 -s -D -1' )  
 time.sleep(10)  
 h1.cmdPrint( 'iperf3 -c ', h2.IP(), ' -J > res.json' )  
  
 info('\*\*\*stopping network' )  
 net.stop()  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 setLogLevel( 'info' )  
 emptyNet()

1. Результат скрипта выводится в файл /corrupt/res.json. Также получены графики, расположенные в каталоге /dup/results (рис. 29).

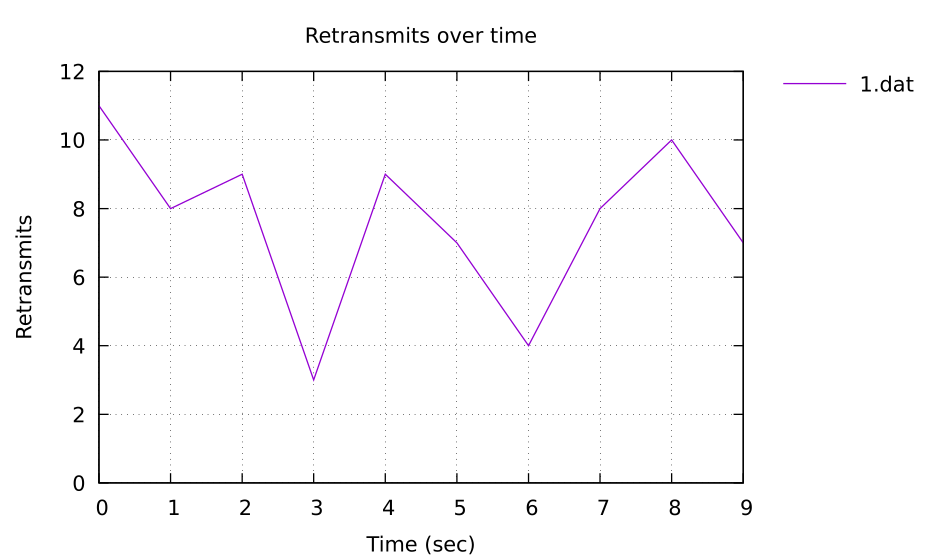


Рис. 29: Результат скрипта /dup/lab\_netem\_ii.py

# 4 Выводы

В ходе лабораторной работы мною было освоено моделирование следующих параметров сети: потеря пакетов, дублирование пакетов, изменение порядка и повреждение пакетов.