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

4.1. 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.

4.2. Предварительные сведения

NETEM — сетевой эмулятор Linux, используемый для тестирования производительности реальных клиент-серверных приложений в виртуальной сети. Виртуальная сеть в данном случае представляет собой лабораторную среду для воспроизведения поведения глобальной сети (Wide Area Network, WAN). NETEM позволяет пользователю задать ряд параметров сети, например, задержку, дрожание задержки (jitter), уровень потери пакетов, дублирование и изменение порядка пакетов.

NETEM реализован в Linux и состоит из двух частей: модуля ядра для организации очередей и утилиты командной строки для его настройки. Между протоколом IP и сетевым устройством создаётся очередь с дисциплиной обслуживания. Дисциплина обслуживания очереди реализуется как объект с двумя интерфейсами. Один интерфейс ставит пакеты в очередь для отправки, а другой интерфейс отправляет пакеты на сетевое устройство. На основе дисциплины обслуживания очередей принимается решение о том, какие пакеты отправлять, какие пакеты задерживать и какие пакеты отбрасывать.

Дисциплины обработки очередей можно разделить на бесклассовые и классовые. Бесклассовые дисциплины, в общем, получают данные, переупорядочивают, вносят задержку или уничтожают их. Такие дисциплины могут использоватся для указания ограничений интерфейса целиком, без какого-либо разделения по классам. Наиболее распространённой бесклассовой дисциплиной является FIFO (первым пришёл, первым обслужен). В NETEM и в Linux в целом эта дисциплина обслуживания очереди используется по умолчанию.

Классовые дисциплины широко используются в случаях, когда тот или иной вид трафика необходимо обрабатывать по разному. Примером классовой дисциплины может служить CBQ — Class Based Queueing (дисциплина обработки очередей на основе классов). Классы трафика организованы в дерево — у каждого класса есть не более одного родителя; класс может иметь множество потомков. Классы, которые не имеют родителей, называются *корневыми*. Классы, которые не имеют потомков, называются *классами-ветками*.

Модуль NETEM входит в ядро ОС Linux и управляется при помощи команды `tc` из пакета `iproute2`. Команда `tc` выполняет все необходимые действия при работе с системой управления трафиком.

Основной синтаксис `tc`, используемый с NETEM:

```
sudo tc qdisc [add|del|replace|change|show] dev dev_id root
  ↪  netem opts
```

Здесь:

- `sudo` — включить выполнение команды с более высокими привилегиями безопасности;
- `tc` — команда, используемая для взаимодействия с NETEM;
- qdisc — дисциплина очереди (qdisc), представляющая собой набор правил, определяющих порядок, в котором обслуживаются пакеты, поступающие с выхода протокола IP.
- `[add | del | replace | change | show]` ([добавить | удалить | заменить | изменить | показать]): указание на использование одной из операций над qdisc (например, для добавления задержки на определенном интерфейсе используется опция `add`, а для изменения или удаления задержки на определенном интерфейсе используется опция `change` или `del` соответственно);
- `dev_id` — параметр указывает интерфейс, подлежащий эмуляции (в данном случае указан корневой интерфейс — `root`);
- `opts` — параметр указывает величину задержки, потери пакетов, дублирования, повреждения и др.

4.3. Задание

1. Задайте простейшую топологию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8.
2. Проведите интерактивные эксперименты по добавлению/изменению задержки, джиттера, значения корреляции для джиттера и задержки, распределения времени задержки в эмулируемой глобальной сети.
3. Реализуйте воспроизводимый эксперимент по заданию значения задержки в эмулируемой глобальной сети. Постройте график.
4. Самостоятельно реализуйте воспроизводимые эксперименты по изменению задержки, джиттера, значения корреляции для джиттера и задержки, распределения времени задержки в эмулируемой глобальной сети. Постройте графики.

4.4. Последовательность выполнения работы

4.4.1. Запуск лабораторной топологии

1. Запустите виртуальную среду с mininet.

2. Из основной ОС подключитесь к виртуальной машине:

```
ssh -Y mininet@192.168.x.y
```

3. В виртуальной машине mininet при необходимости исправьте права запуска X-соединения. Скопируйте значение куки (MIT magic cookie)[1] своего пользователя `mininet` в файл для пользователя `root`:

```
mininet@mininet-vm:~$ xauth list $DISPLAY
mininet-vm/unix:10  MIT-MAGIC-COOKIE-1
 ↪  295acad8e35d17636924c5ab80e8462d

mininet@mininet-vm:~$ sudo -i
root@mininet-vm:~# xauth add mininet-vm/unix:10
 ↪  MIT-MAGIC-COOKIE-1  295acad8e35d17636924c5ab80e8462d
root@mininet-vm:~# logout
```

После выполнения этих действий графические приложения должны запускаться под пользователем mininet.

4. Задайте простейшую топологию, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8:

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

После введения этой команды запустятся терминалы двух хостов, коммутатора и контроллера. Терминалы коммутатора и контроллера можно закрыть.

5. На хостах `h1` и `h2` введите 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`.
6. Проверьте подключение между хостами `h1` и `h2` с помощью команды `ping` с параметром `-c 6`.
7. Укажите в отчёте минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи (RTT).

4.4.2. Интерактивные эксперименты

4.4.2.1. Добавление/изменение задержки в эмулируемой глобальной сети

Сетевые эмуляторы задают задержки на интерфейсе. Например, задержка, вносимая в интерфейс коммутатора A, который подключён к интерфейсу коммутатора B, может представлять собой задержку распространения WAN, соединяющей оба коммутатора.

[1] Значение для *MIT-MAGIC-COOKIE* приведено условно.

1. На хосте h1 добавьте задержку в 100 мс к выходному интерфейсу:

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

   Здесь:
   - sudo: выполнить команду с более высокими привилегиями;
   - tc: вызвать управление трафиком Linux;
   - qdisc: изменить дисциплину очередей сетевого планировщика;
   - add: создать новое правило;
   - dev h1-eth0: указать интерфейс, на котором будет применяться правило;
   - netem: использовать эмулятор сети;
   - delay 100ms: задержка ввода 100 мс.
2. Проверьте, что соединение от хоста h1 к хосту h2 имеет задержку 100 мс, используя команду ping с параметром -c 6 с хоста h1. Укажите в отчёте минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи (RTT).
3. Для эмуляции глобальной сети с двунаправленной задержкой необходимо к соответствующему интерфейсу на хосте h2 также добавить задержку в 100 миллисекунд:

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

4. Проверьте, 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. Укажите в отчёте минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи (RTT).

4.4.2.2. Изменение задержки в эмулируемой глобальной сети

1. Измените задержку со 100 мс до 50 мс для отправителя h1:

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

   и для получателя h2:

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

2. Проверьте, что соединение от хоста h1 к хосту h2 имеет задержку 100 мс, используя команду ping с параметром -c 6 с терминала хоста h1. Укажите в отчёте минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи (RTT).

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

1. Восстановите конфигурацию по умолчанию, удалив все правила, применённые к сетевому планировщику соответствующего интерфейса. Для отправителя h1:

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

   Для получателя h2:

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

2. Проверьте, что соединение между хостом h1 и хостом h2 не имеет явно установленной задержки, используя команду ping с параметром -c 6 с терминала хоста h1. Укажите в отчёте минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи (RTT).

4.4.2.4. Добавление значения дрожания задержки в интерфейс подключения к эмулируемой глобальной сети

В сетях нет постоянной задержки. Она может варьироваться в зависимости от других потоков трафика, конкурирующих за тот же путь. Джиттер (jitter) — это изменение времени задержки. Параметры задержки описываются в терминах теории вероятностей *средним значением* μ, *стандартным отклонением* σ и *корреляцией*. По умолчанию NETEM использует *равномерное распределение*, так что задержка находится в пределах $\mu \pm \sigma$. Параметр корреляции управляет отношением между последовательными псевдослучайными значениями.

1. При необходимости восстановите конфигурацию интерфейсов по умолчанию на узлах h1 и h2:

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

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

2. Добавьте на узле h1 задержку в 100 мс со случайным отклонением 10 мс:

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

3. Проверьте, 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. Укажите в отчёте минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи (RTT).
4. Восстановите конфигурацию интерфейса по умолчанию на узле h1.

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

1. При необходимости восстановите конфигурацию интерфейсов по умолчанию на узлах `h1` и `h2`.
2. Добавьте на интерфейсе хоста `h1` задержку в 100 мс с вариацией ± 10 мс и значением корреляции в 25%:

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

3. Убедитесь, что все пакеты, покидающие устройство `h1` на интерфейсе `h1-eth0`, будут иметь время задержки 100 мс со случайным отклонением ± 10 мс, при этом время передачи следующего пакета зависит от предыдущего значения на 25%. Используйте для этого в терминале хоста `h1` команду `ping` с параметром `-c 20`. Укажите в отчёте минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи (RTT).
4. Восстановите конфигурацию интерфейса по умолчанию на узле `h1`.

4.4.2.6. Распределение задержки в интерфейсе подключения к эмулируемой глобальной сети

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

1. При необходимости восстановите конфигурацию интерфейсов по умолчанию на узлах `h1` и `h2`.
2. Задайте нормальное распределение задержки на узле `h1` в эмулируемой сети:

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

3. Убедитесь, что все пакеты, покидающие хост `h1` на интерфейсе `h1-eth0`, будут иметь время задержки, которое распределено в диапазоне 100 мс ± 20 мс. Используйте для этого команду `ping` на терминале хоста `h1` с параметром `-c 10`.
4. Восстановите конфигурацию интерфейса по умолчанию на узле `h1`.
5. Завершите работу mininet в интерактивном режиме, введя в интерфейсе mininet:

```
mininet> exit
```

4.4.3. Воспроизведение экспериментов

4.4.3.1. Предварительная подготовка

1. Обновите репозитории программного обеспечения на виртуальной машине:

```
sudo apt-get update
```

2. Установите пакет geeqie — понадобится для просмотра файлов png:

```
sudo apt install geeqie
```

3. Для каждого воспроизводимого эксперимента `expname` создайте свой каталог, в котором будут размещаться файлы эксперимента:

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

   Здесь `expname` может принимать значения `simple-delay`, `change-delay`, `jitter-delay`, `correlation-delay` и т.п.
4. Для каждого случая создайте скрипт для проведения эксперимента `lab_netem_i.py` и скрипт для визуализации результатов `ping_plot`.

4.4.3.2. Добавление задержки для интерфейса, подключающегося к эмулируемой глобальной сети

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

1. В виртуальной среде mininet в своём рабочем каталоге с проектами создайте каталог `simple-delay` и перейдите в него:

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

2. Создаёте 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}\' | sed -e \'s/time=//g\' -e
     ↪  \'s/icmp_seq=//g\' > ping.dat' )

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

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

3. В отчёте поясните содержание скрипта `lab_netem_i.py`. В каких строках скрипта задается значение задержки для интерфейса хоста? Каким образом формируется файл с результатами эксперимента для последующего построения графиков, какие значения в нём размещены?
4. Создаёте 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
```

5. Задайте права доступа к файлу скрипта:

```
chmod +x ping_plot
```

6. Создайте `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
```

7. Выполните эксперимент:

```
make
```

8. Продемонстрируйте построенный в результате выполнения скриптов график.
9. Из файла `ping.dat` удалите первую строку и заново постройте график:

```
make ping.png
```

10. Продемонстрируйте построенный в результате график.
11. Разработайте скрипт для вычисления на основе данных файла ping.dat минимального, среднего, максимального и стандартного отклонения времени

приёма-передачи. Добавьте правило запуска скрипта в Makefile. Продемонстрируйте работу скрипта с выводом значений на экран или в отдельный файл.

12. Очистите каталог от результатов проведения экспериментов:

```
make clean
```

4.4.3.3. Задание для самостоятельной работы

Самостоятельно реализуйте воспроизводимые эксперименты по изменению задержки, джиттера, значения корреляции для джиттера и задержки, распределения времени задержки в эмулируемой глобальной сети. Постройте графики. Вычислите минимальное, среднее, максимальное и стандартное отклонение времени приёма-передачи для каждого случая.

4.5. Содержание отчёта

1. Титульный лист с указанием номера лабораторной работы и ФИО студента.
2. Формулировка задания работы.
3. Описание результатов выполнения задания:
   - скриншоты (снимки экрана), фиксирующие выполнение работы;
   - подробное описание настроек служб в соответствии с заданием;
   - полные тексты конфигурационных файлов настраиваемых в работе служб;
   - результаты проверки корректности настроек служб в соответствии с заданием (подтверждённые скриншотами).
4. Выводы, согласованные с заданием работы.