Отчёт по лабораторной работе №3 по дисциплине Моделирование сетей передачи данных

Измерение и тестирование пропускной способности сети. Воспроизводимый эксперимент

Шаповалова Диана Дмитриевна, НПИбд-02-21, 1032211220

Содержание

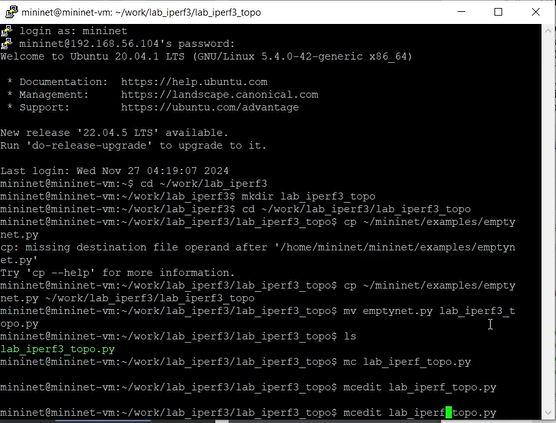
# 1 Цель работы

Основной целью работы является знакомство с инструментом для измерения пропускной способности сети в режиме реального времени — iPerf3, а также получение навыков проведения воспроизводимого эксперимента по измерению пропускной способности моделируемой сети в среде Mininet.

# 2 Выполнение лабораторной работы

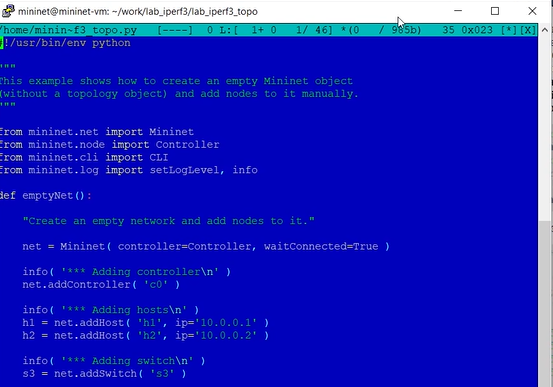
1. С помощью API Mininet создайте простейшую топологию сети, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8:

– В каталоге /work/lab\_iperf3 для работы над проектом создайте подкаталог lab\_iperf3\_topo и скопируйте в него файл с примером скрипта mininet/examples/emptynet.py, описывающего стандартную простую топологию сети mininet:



Создаем подкаталог

– Изучите содержание скрипта lab\_iperf3\_topo.py:



Содержание скрипта lab\_iperf3\_topo.py

Основные элементы:

– addSwitch(): добавляет коммутатор в топологию и возвращает имя коммутатора;

– ddHost(): добавляет хост в топологию и возвращает имя хоста;

– addLink(): добавляет двунаправленную ссылку в топологию (и возвращает ключ ссылки; ссылки в Mininet являются двунаправленными, если не указано иное);

– Mininet: основной класс для создания и управления сетью;

– start(): запускает сеть;

– pingAll(): проверяет подключение, пытаясь заставить все узлы пинговать друг друга;

– stop(): останавливает сеть;

– net.hosts: все хосты в сети;

– dumpNodeConnections(): сбрасывает подключения к/от набора узлов;

– setLogLevel( ‘info’ | ‘debug’ | ‘output’ ): устанавливает уровень вывода Mininet по умолчанию; рекомендуется info.

Запустите скрипт создания топологии lab\_iperf3\_topo.py:

sudo python lab\_iperf3\_topo.py

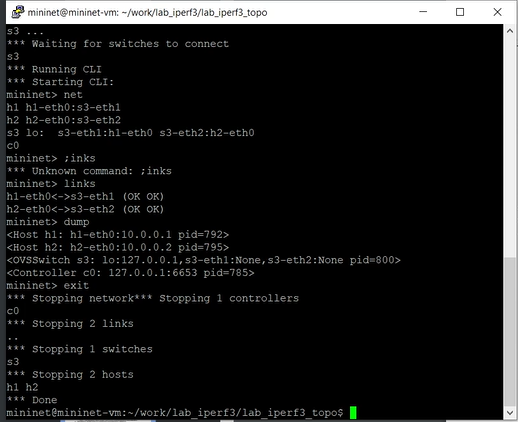
После отработки скрипта посмотрите элементы топологии и завершите работу mininet:

mininet> net

mininet> links

mininet> dump

mininet> exit



Запускаем скрипт, смотрим элементы топологии

1. Внесите в скрипт lab\_iperf3\_topo.py изменение, позволяющее вывести на экран информацию о хосте h1, а именно имя хоста, его IP-адрес, MACадрес. Для этого после строки, задающей старт работы сети, добавьте строку:

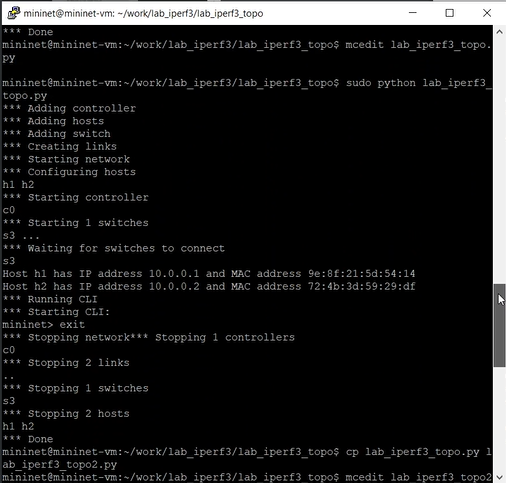
print( “Host”, h1.name, “has IP address”, h1.IP(), “and MAC address”, h1.MAC() )

Здесь:

– IP() возвращает IP-адрес хоста или определенного интерфейса;

– MAC() возвращает MAC-адрес хоста или определенного интерфейса.

1. Проверьте корректность отработки изменённого скрипта.
2. Измените скрипт lab\_iperf3\_topo.py так, чтобы на экран выводилась информация об имени, IP-адресе и MAC-адресе обоих хостов сети. Проверьте корректность отработки изменённого скрипта.



Проверяем корректность отработки изменённого скрипта

1. Mininet предоставляет функции ограничения производительности и изоляции с помощью классов CPULimitedHost и TCLink. Добавьте в скрипт настройки параметров производительности:

– Сделайте копию скрипта lab\_iperf3\_topo.py:

cp lab\_iperf3\_topo.py lab\_iperf3\_topo2.py

– В начале скрипта lab\_iperf3\_topo2.py добавьте записи об импорте классов CPULimitedHost и TCLink:

from mininet.node import CPULimitedHost

from mininet.link import TCLink

– В скрипте lab\_iperf3\_topo2.py измените строку описания сети, указав на использование ограничения производительности и изоляции:

net = Mininet( controller=Controller, waitConnected=True, host = CPULimitedHost, link = TCLink )

– В скрипте lab\_iperf3\_topo2.py измените функцию задания параметров виртуального хоста h1, указав, что ему будет выделено 50% от общих ресурсов процессора системы:

h1 = net.addHost( ‘h1’, ip=‘10.0.0.1’, cpu=50 )

– Аналогичным образом для хоста h2 задайте долю выделения ресурсов процессора в 45%.

– В скрипте lab\_iperf3\_topo2.py измените функцию параметров соединения между хостом h1 и коммутатором s3:

net.addLink( h1, s3, bw=10, delay=‘5ms’, max\_queue\_size=1000, loss=10, use\_htb=True )

Здесь добавляется двунаправленный канал с характеристиками пропускной способности, задержки и потерь:

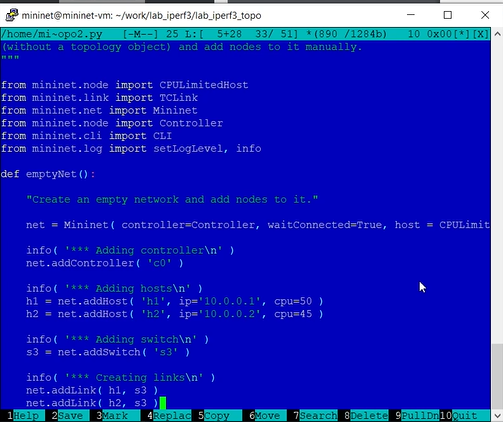
– параметр пропускной способности (bw) выражается числом в Мбит;

– задержка (delay) выражается в виде строки с заданными единицами измерения (например, 5ms, 100us, 1s);

– потери (loss) выражаются в процентах (от 0 до 100);

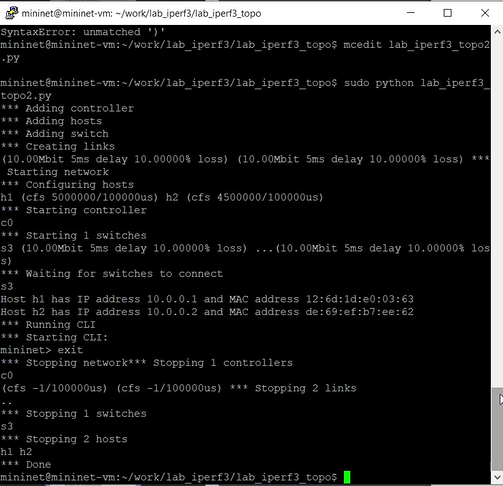
– параметр максимального значения очереди (max\_queue\_size) выражается в пакетах;

– параметр use\_htb указывает на использование ограничителя интенсивности входящего потока Hierarchical Token Bucket (HTB).



Добаваляем в скрипт настройки параметров производительности

– Запустите на отработку скрипт lab\_iperf3\_topo2.py



Запускаем скрипт lab\_iperf3\_topo2.py

1. Постройте графики по проводимому эксперименту:

– Сделайте копию скрипта lab\_iperf3\_topo2.py и поместите его в подкаталог iperf:

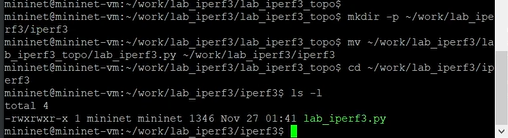
cp lab\_iperf3\_topo2.py lab\_iperf3.py

mkdir -p ~/work/lab\_iperf3/iperf3

mv ~/work/lab\_iperf3/lab\_iperf3\_topo/lab\_iperf3.py ~/work/lab\_iperf3/iperf3

cd ~/work/lab\_iperf3/iperf3

ls -l



Копируем скрипт lab\_iperf3\_topo2.py и помещаем его в подкаталог iperf

– В начале скрипта lab\_iperf3.py добавьте запись

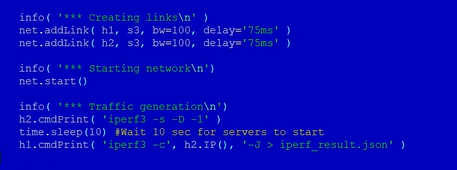
import time

– Измените код в скрипте lab\_iperf3.py так, чтобы:

– на хостах не было ограничения по использованию ресурсов процессора;

– каналы между хостами и коммутатором были по 100 Мбит/с с задержкой 75 мс, без потерь, без использования ограничителей пропускной способности и максимального размера очереди

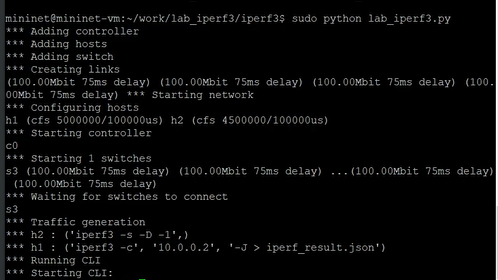
– После функции старта сети опишите запуск на хосте h2 сервера iPerf3, а на хосте h1 запуск с задержкой в 10 секунд клиента iPerf3 с экспортом результатов в JSON-файл, закомментируйте строки, отвечающие за запуск CLI-интерфейса



Меняем код в скрипте lab\_iperf3.py

– Запустите на отработку скрипт lab\_iperf3.py:

sudo python lab\_iperf3.py



Запускаем скрипт lab\_iperf3.py

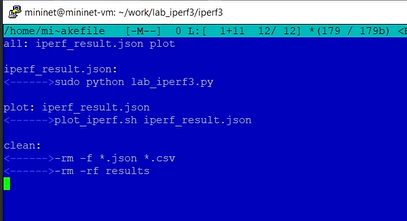
– Постройте графики из получившегося JSON-файла:

plot\_iperf.sh iperf\_result.json

– Создайте Makefile для проведения всего эксперимента:

touch Makefile

– В Makefile пропишите запуск скрипта эксперимента, построение графиков и очистку каталога от результатов:

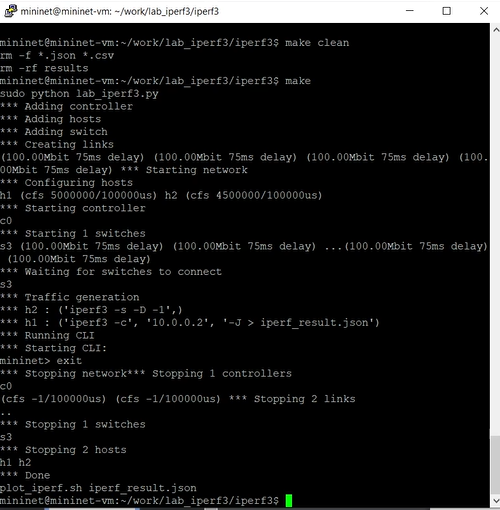


Прописываем запуск скрипта эксперимента, построение графиков и очистку каталога от результатов

– Проверьте корректность отработки Makefile:

make clean

make



Проверяем корректность отработки Makefile

# 3 Выводы

Мы познакомились с инструментом для измерения пропускной способности сети в режиме реального времени — iPerf3, а также получили навыки проведения воспроизводимого эксперимента по измерению пропускной способности моделируемой сети в среде Mininet.

# 4 Список литературы

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