Лабораторная работа № 3

Измерение и тестирование пропускной способности сети. Воспроизводимый эксперимент

Хрусталев Влад Николаевич

Содержание

Список иллюстраций

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 Теоретическое введение

Mininet[1] – 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.

iPerf3[2] представляет собой кроссплатформенное клиент-серверное приложение с открытым исходным кодом, которое можно использовать для измерения пропускной способности между двумя конечными устройствами. iPerf3 может работать с транспортными протоколами TCP, UDP и SCTP:

- TCP и SCTP:
  - измеряет пропускную способность;
  - позволяет задать размер MSS/MTU;
  - отслеживает размер окна перегрузки TCP (CWnd).
- UDP:
  - измеряет пропускную способность;
  - измеряет потери пакетов;
  - измеряет колебания задержки (jitter);
  - поддерживает групповую рассылку пакетов (multicast).

3 Задание

1. Воспроизвести посредством API Mininet эксперименты по измерению пропускной способности с помощью iPerf3.
2. Построить графики по проведённому эксперименту.

4 Выполнение лабораторной работы

С помощью API Mininet создадим простейшую топологию сети, состоящую из двух хостов и коммутатора с назначенной по умолчанию mininet сетью 10.0.0.0/8. Для этого в каталоге /work/lab_iperf3 для работы над проектом создадим подкаталог lab_iperf3_topo и скопируем в него файл с примером скрипта mininet/examples/emptynet.py, описывающего стандартную простую топологию сети mininet (рис. 4.1):

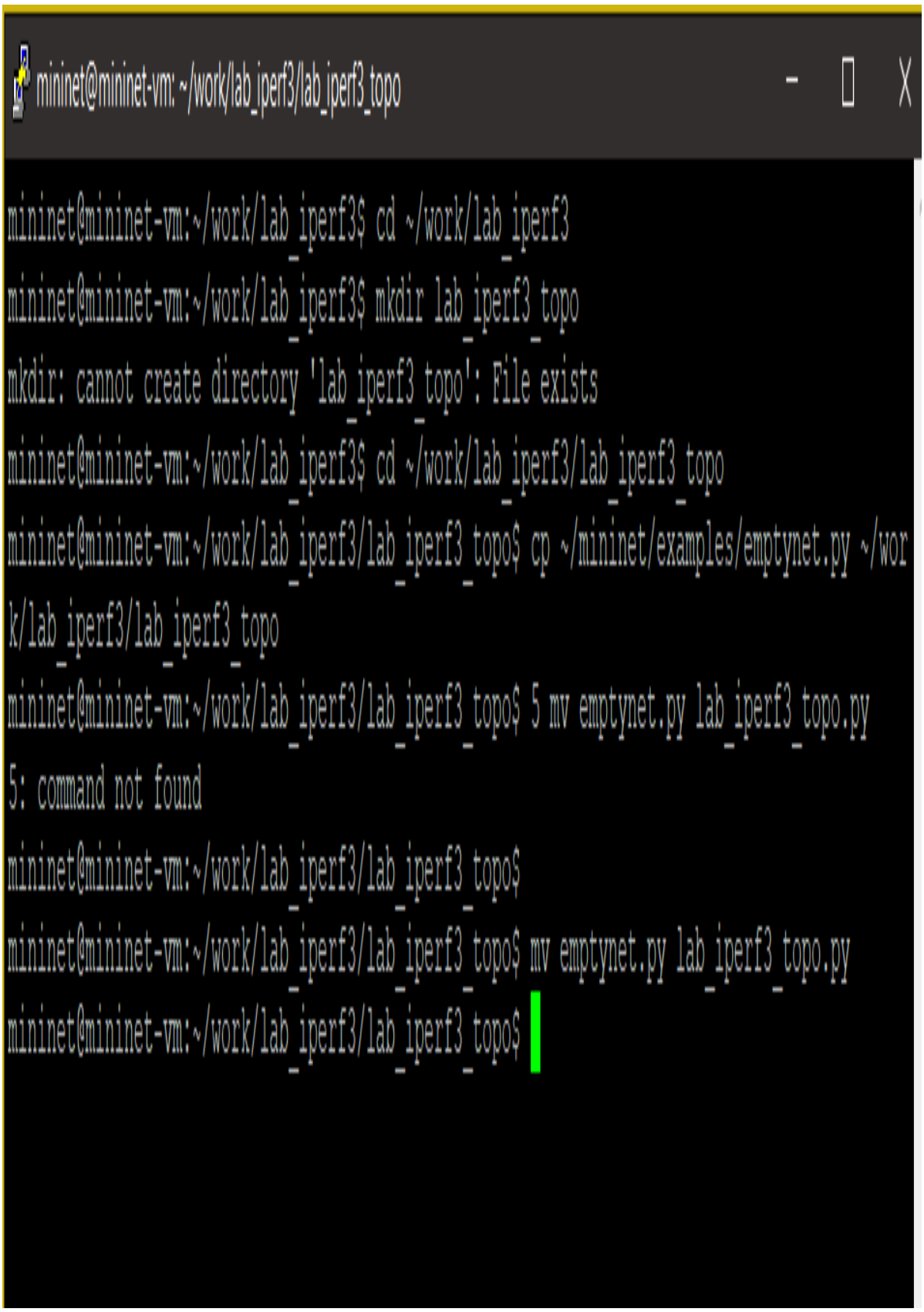

Рис. 4.1: Создание подкаталога, копирование файла с примером скрипта (описывающего стандартную простую топологию сети mininet)

Изучим содержание скрипта lab_iperf3_topo.py (рис. 4.2).

В нем написан скрипт по созданию простейшей топологии из двух хостов h1 и h2, а также коммутатора s3 и контроллера c0. В начале файла видим импорт необходимых библиотек.

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.

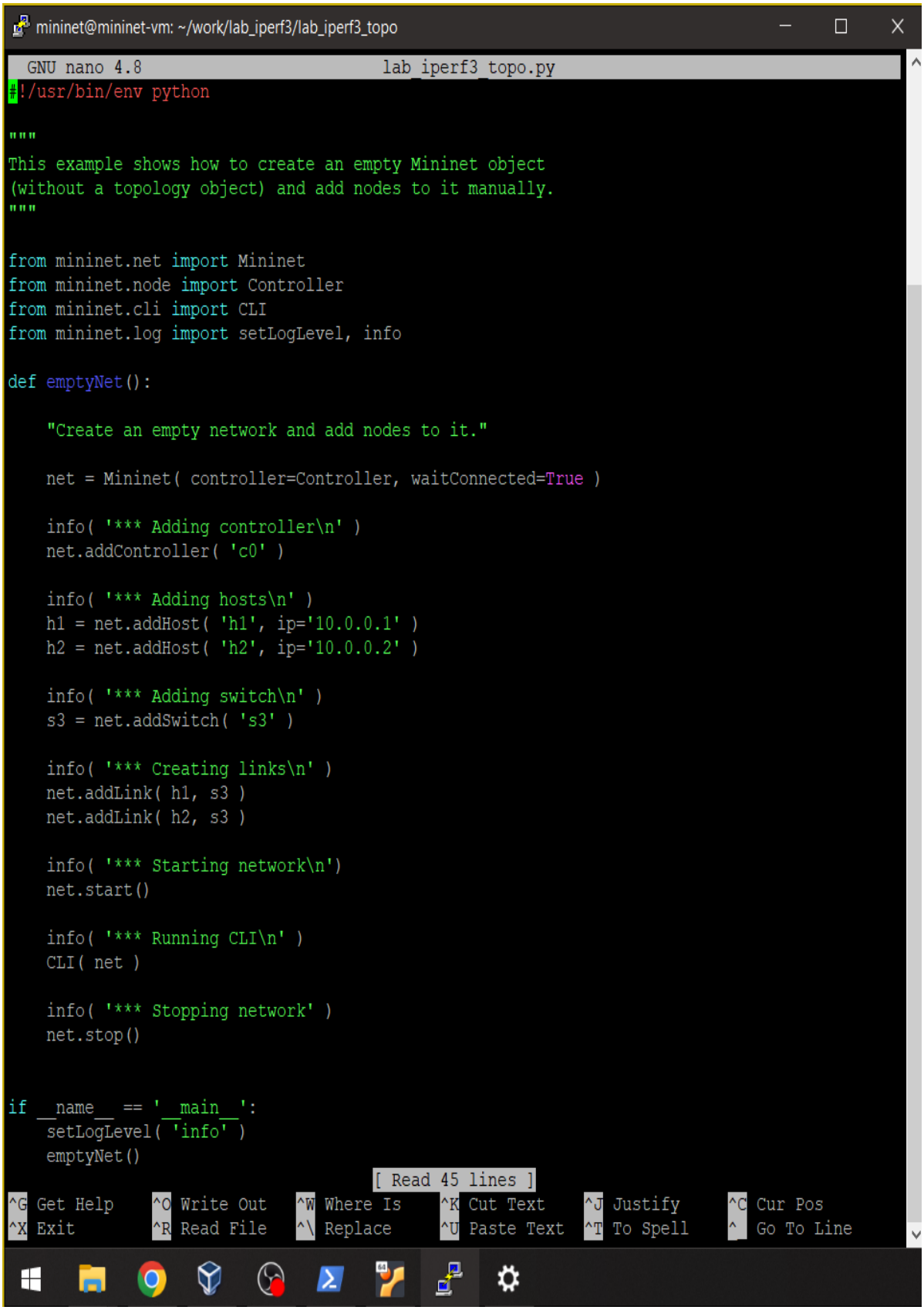

Рис. 4.2: Содержание файла lab_iperf3_topo.py

Запустим скрипт создания топологии lab_iperf3_topo.py. После отработки скрипта просмотрим элементы топологии и завершим работу mininet (рис. 4.3):

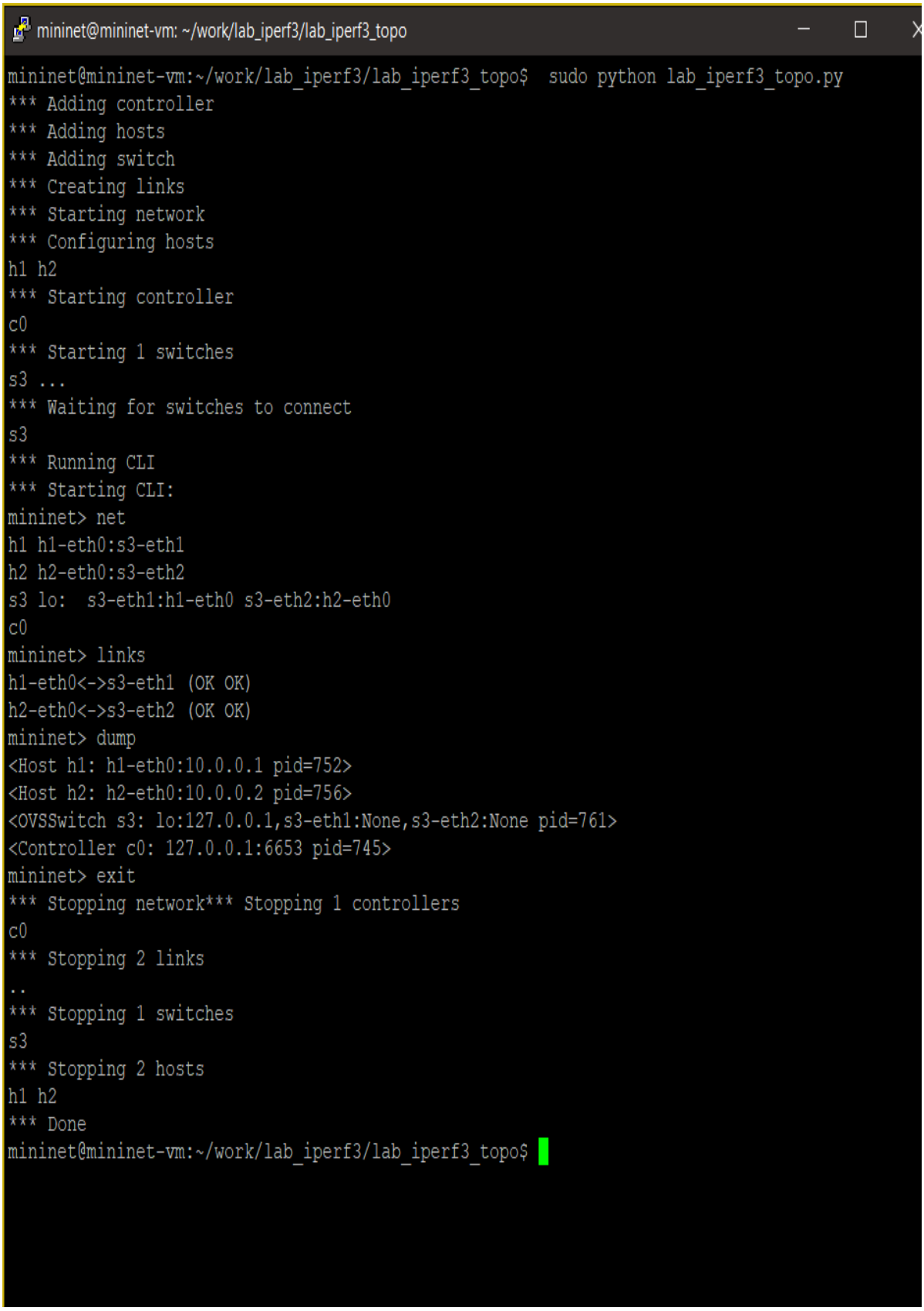

Рис. 4.3: Запуск скрипта создания топологии и дальнейший просмотр элементов

Объеденим два пункта лабораторной работы и внесём измекнения сразу для обоих хостов. А именно внесем в скрипт lab_iperf3_topo.py изменение, позволяющее вывести на экран информацию обоих хостов сети, а именно имя хоста, его IP-адрес, MAC-адрес(рис. 4.4):

Здесь:

- IP() возвращает IP-адрес хоста или определенного интерфейса;
- MAC() возвращает MAC-адрес хоста или определенного интерфейса.

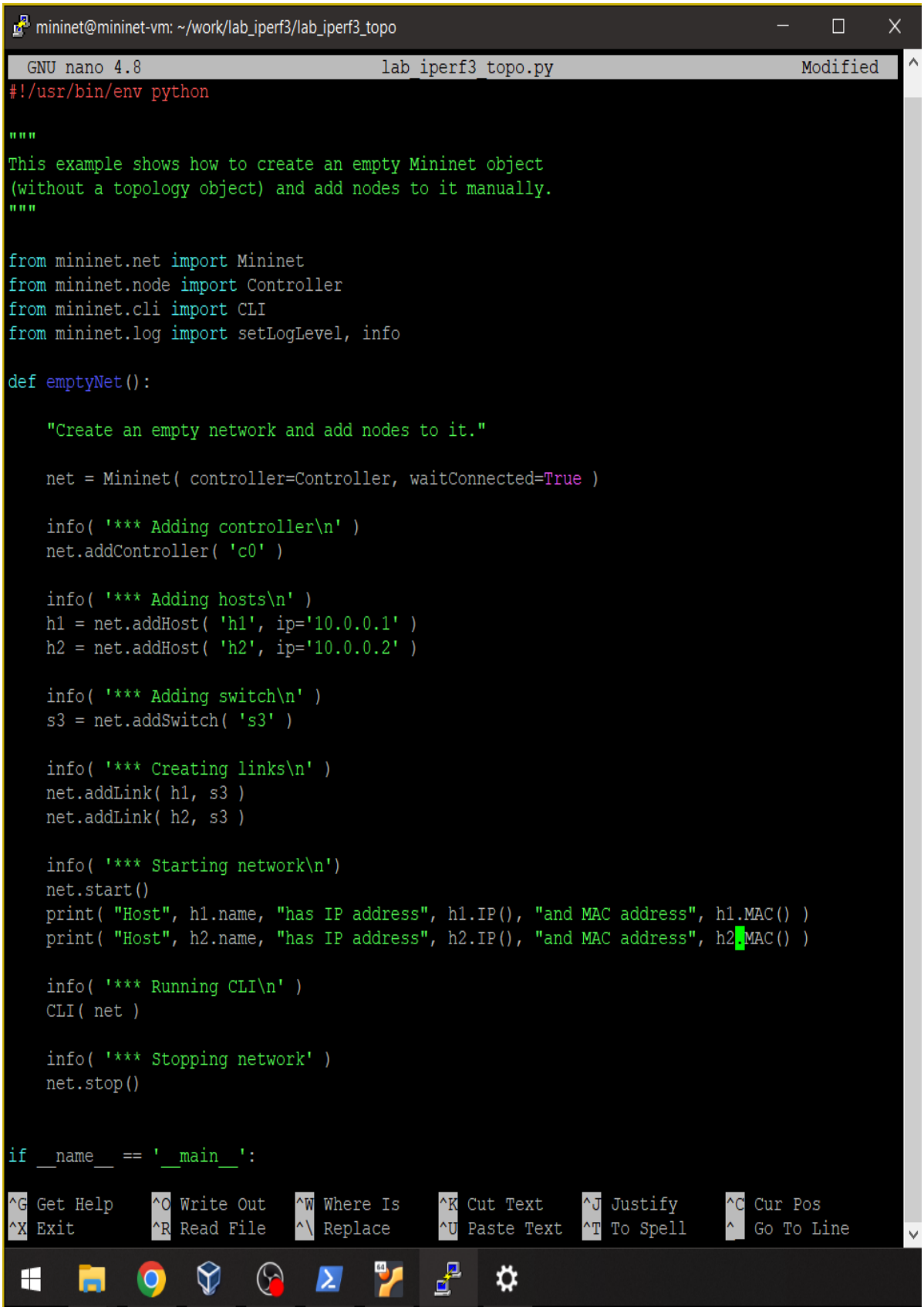

Рис. 4.4: Внесение изменения в скрипт, позволяющего вывести на экран информацию о хостах h1 и h2 (имя, IP-адрес, MAC-адрес)

Проверим корректность отработки изменённого скрипта (рис. 4.5):

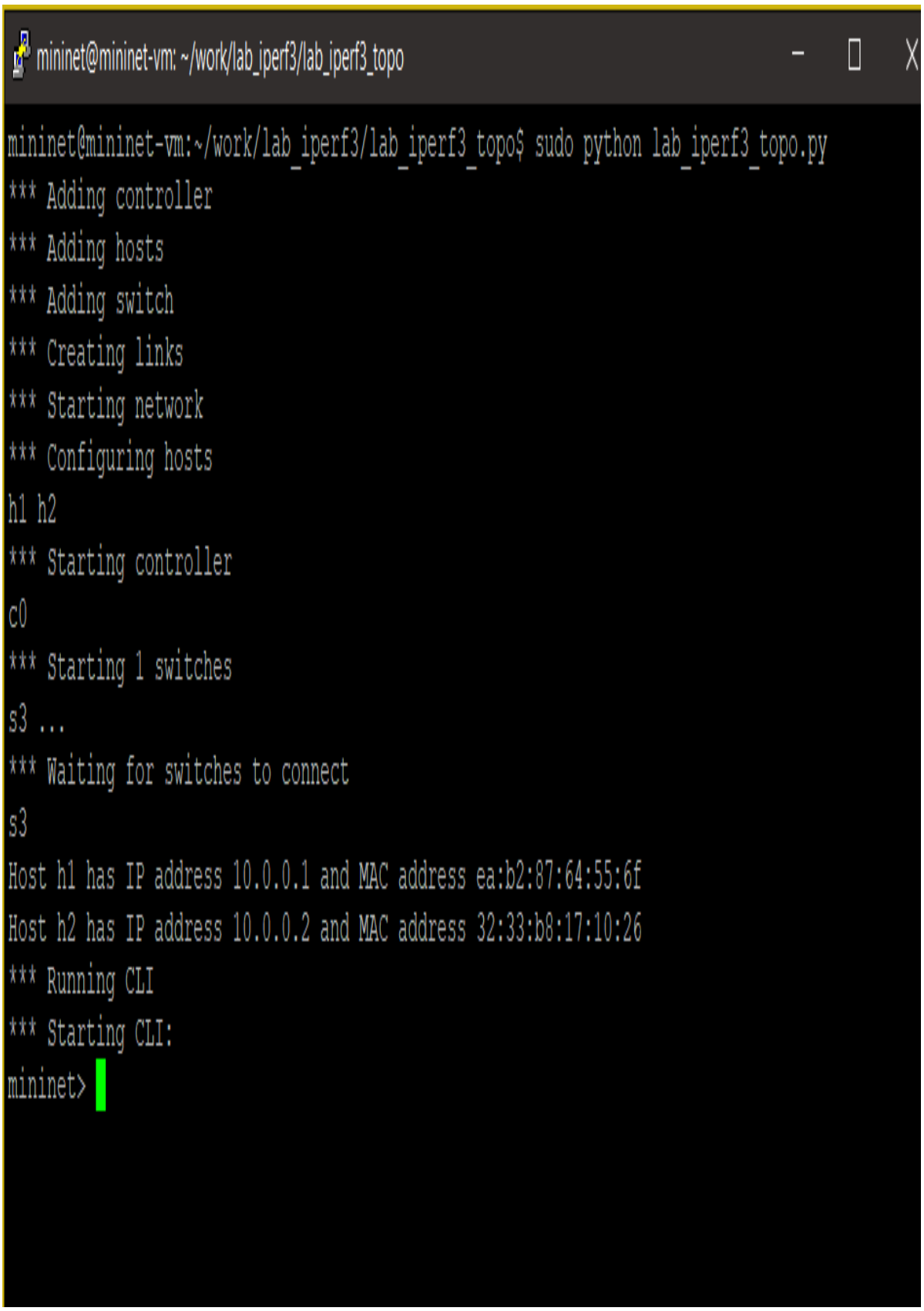

Рис. 4.5: Проверка корректности отработки скрипта

Нам вывелась информация об IP и mac адресах хостов.

Mininet предоставляет функции ограничения производительности и изоляции с помощью классов CPULimitedHost и TCLink. Добавим в скрипт настройки параметров производительности. Для начала сделаем копию скрипта lab_iperf3_topo.py. В начале скрипта lab_iperf3_topo2.py добавим записи об импорте классов CPULimitedHost и TCLink. Далее изменим строку описания сети, указав на использование ограничения производительности и изоляции. Следующим шагом изменим функцию задания параметров виртуального хоста h1, указав, что ему будет выделено 50% от общих ресурсов процессора системы. Аналогичным образом для хоста h2 зададим долю выделения ресурсов процессора в 45%. В конце изменим функцию параметров соединения между хостом h1 и коммутатором s3 (рис. 4.6).

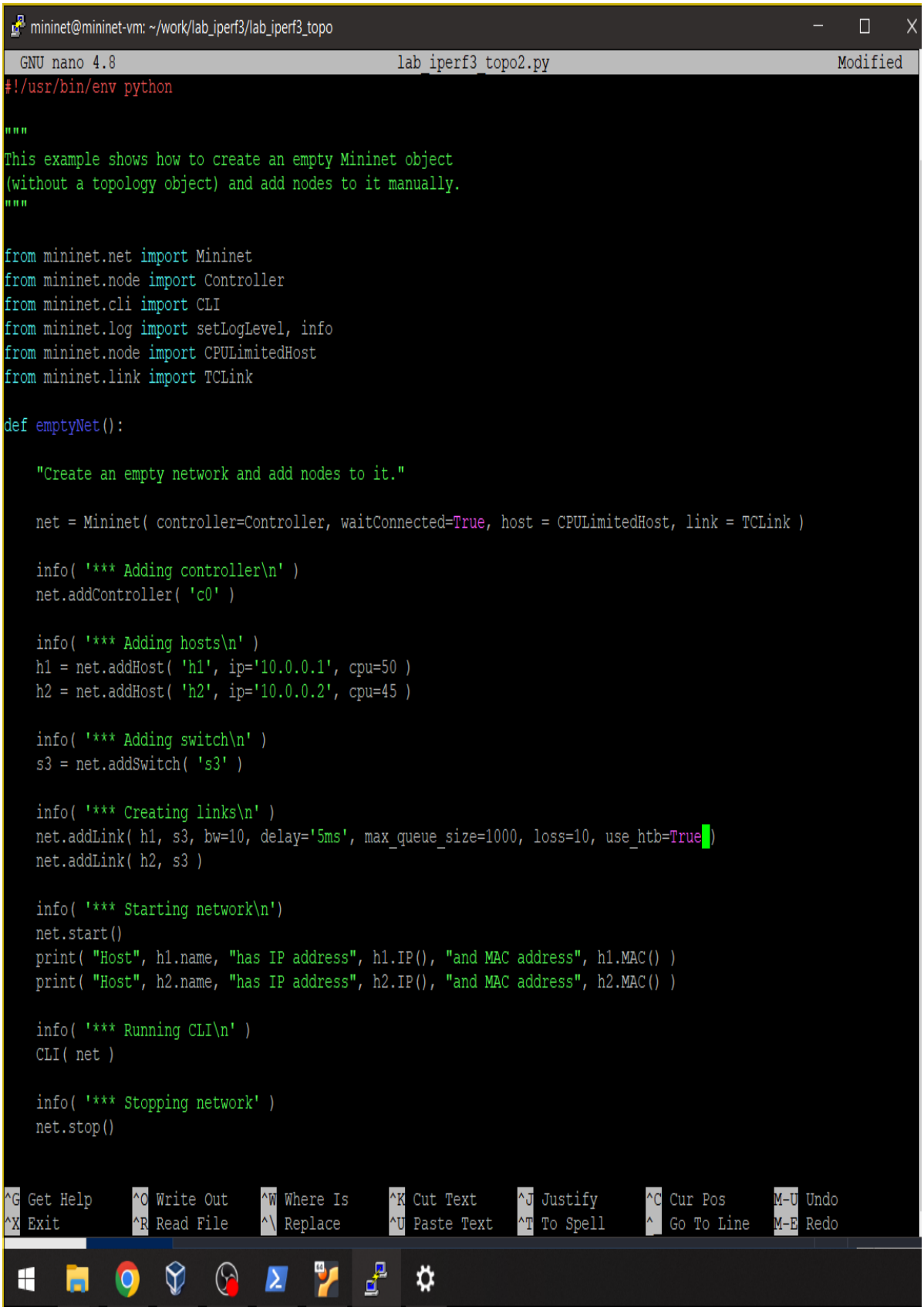

Рис. 4.6: Настройка параметров производительности

Запустим на отработку сначала скрипт lab_iperf3_topo2.py, затем lab_iperf3_topo.py и сравним результат (рис. 4.7). Видим, что в первом случае у нас создалась сеть с настроенными параметрами, а во втором случае дефолтная сеть без этих параметров.

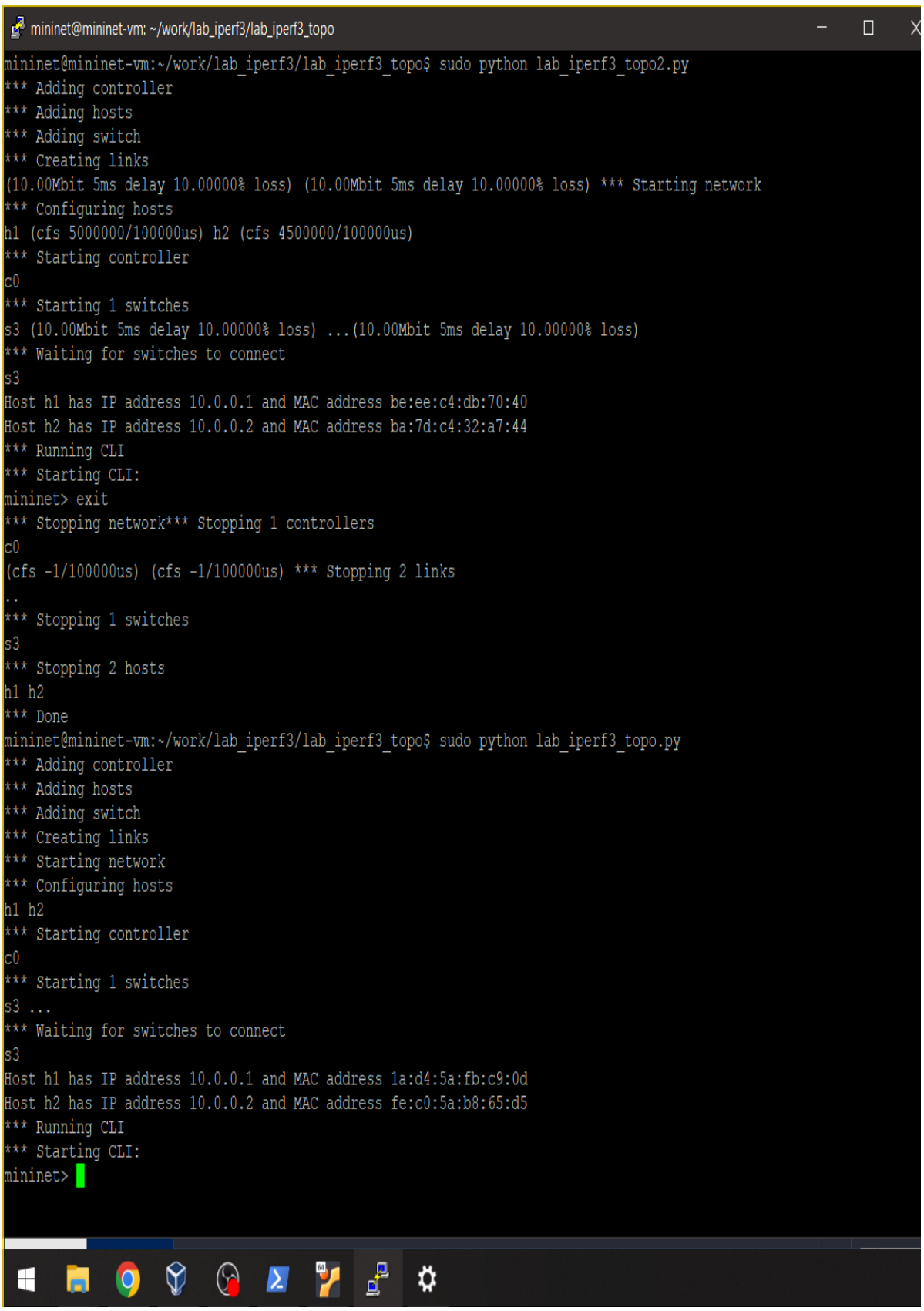

Рис. 4.7: Запуск скрипта с настройкой параметров производительности и без нее

Сделаем копию скрипта lab_iperf3_topo2.py и поместим его в подкаталог iperf рис. 4.8).

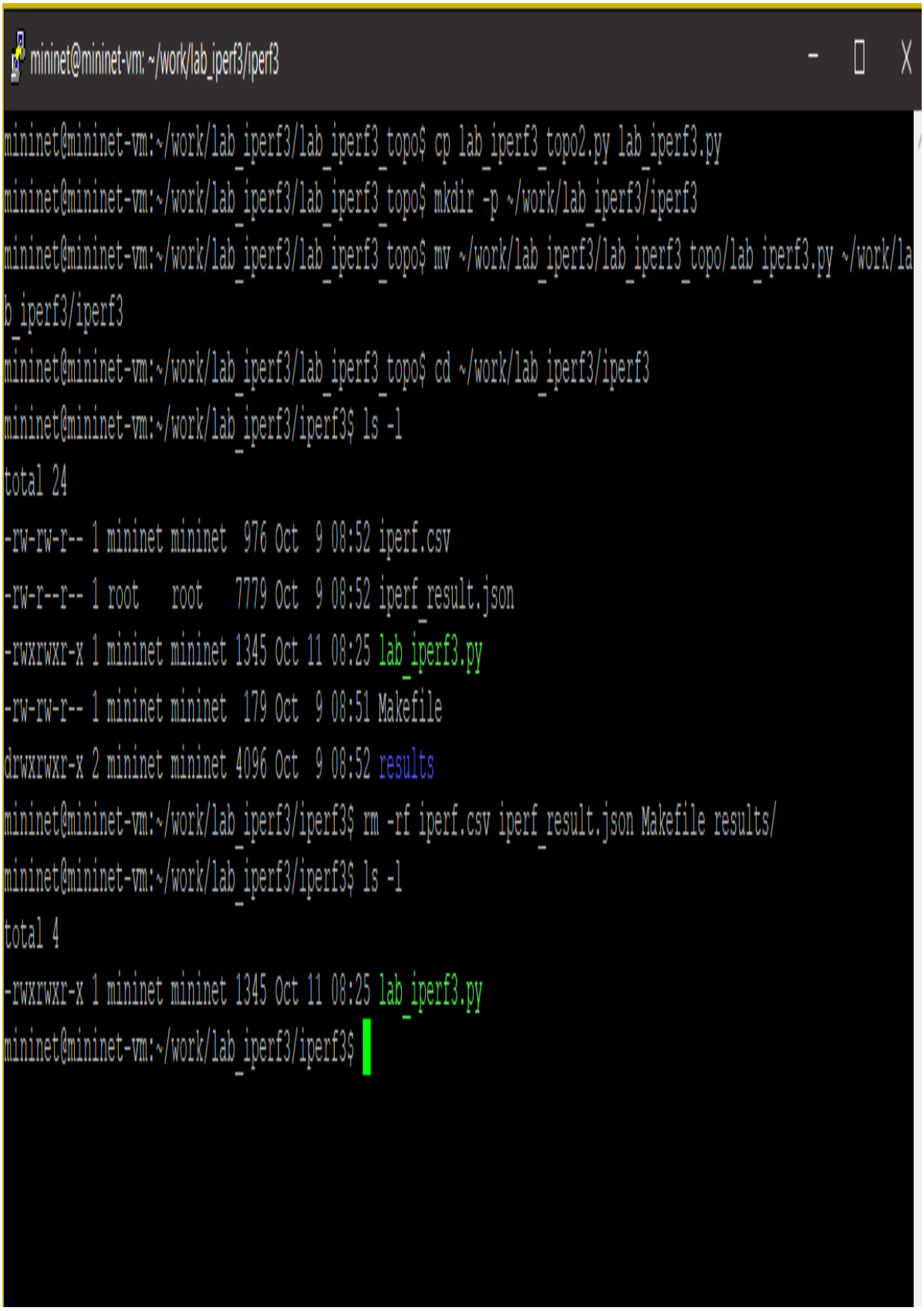

Рис. 4.8: Создание копии скрипта lab_iperf3_topo2.py

В начале скрипта lab_iperf3.py добавим запись об импорте time и изменим код в скрипте так, чтобы (рис. 4.9): - 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

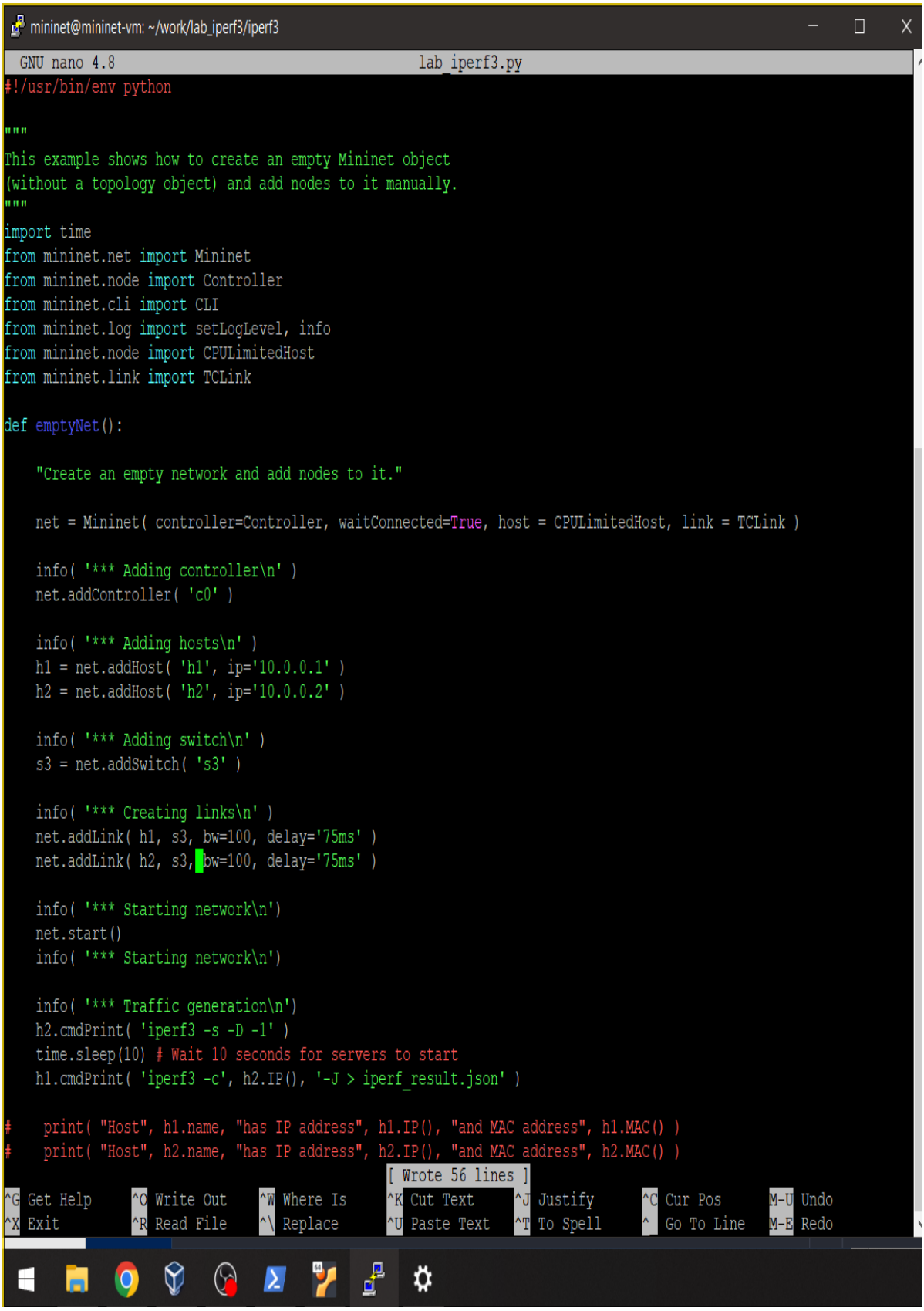

Рис. 4.9: Изменен ия кода в скрипте lab_iperf3.py

Запустим на отработку скрипт lab_iperf3.py (рис. 4.10).

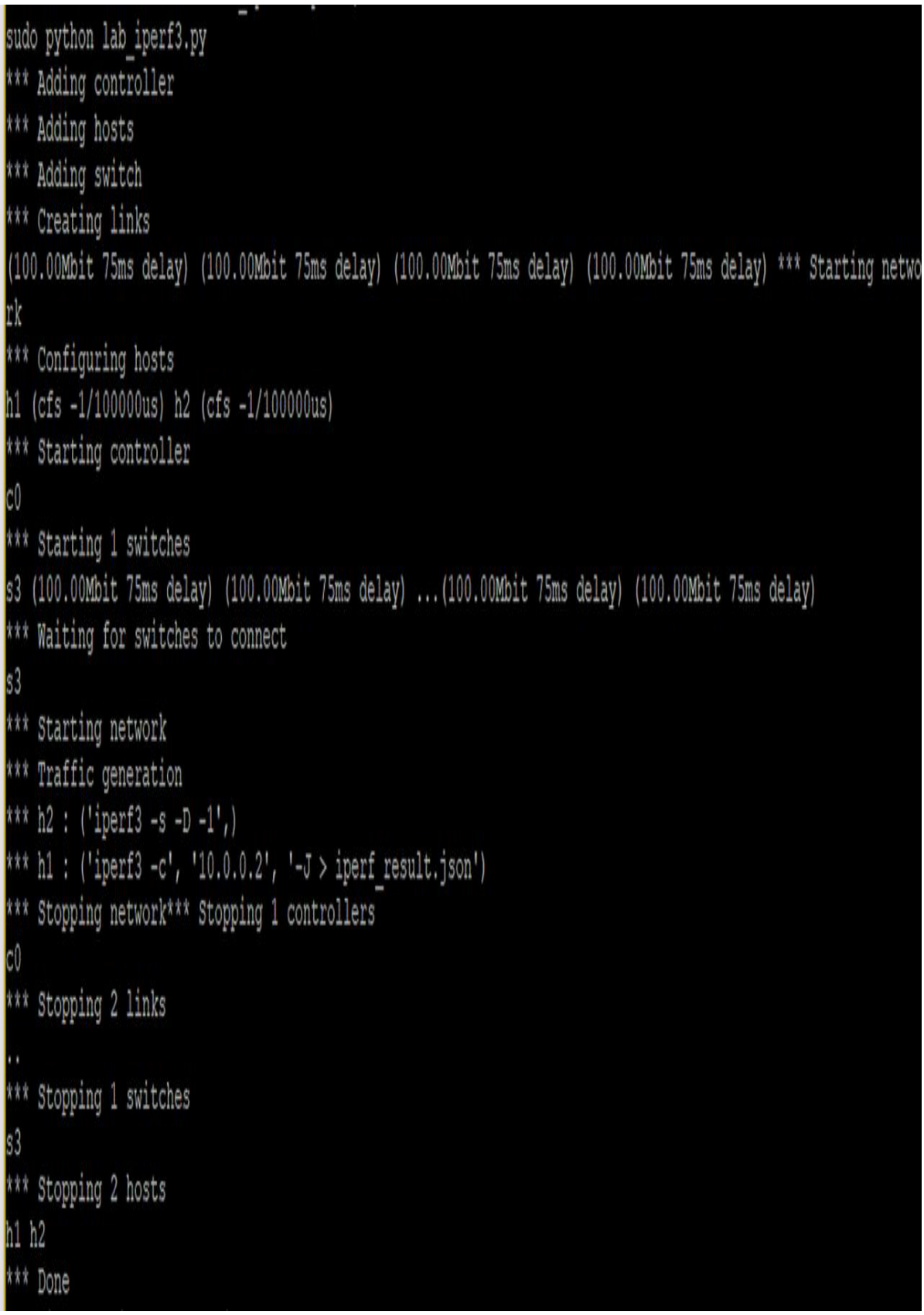

Рис. 4.10: Запуск скрипта lab_iperf3.py

Построим графики из получившегося JSON-файла (рис. 4.12). Создадим Makefile для проведения всего эксперимента. В Makefile пропишем запуск скрипта эксперимента, построение графиков и очистку каталога от результатов (рис. 4.11).

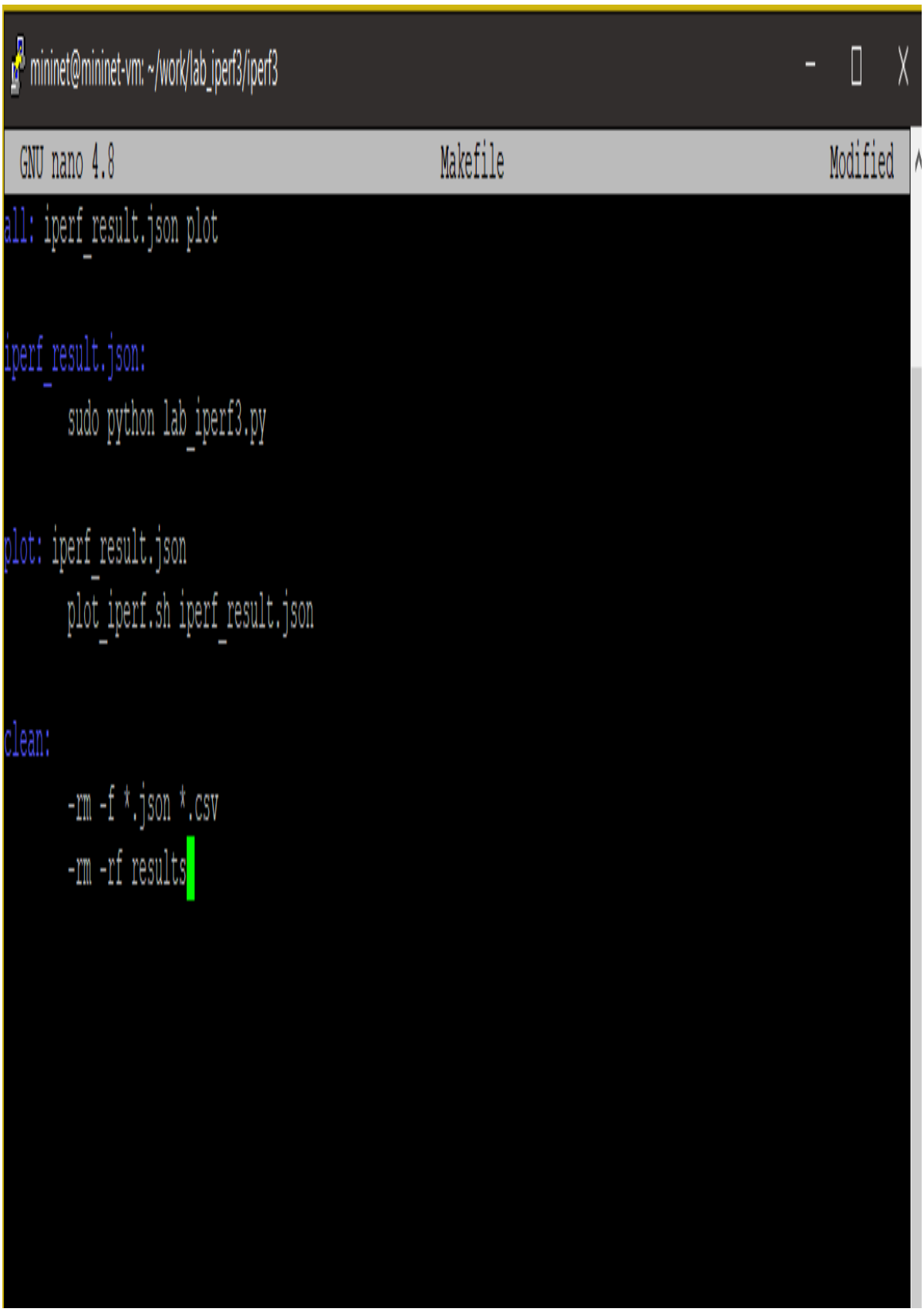

Рис. 4.11: Создание Makefile

Проверьте корректность отработки Makefile (рис. 4.12).

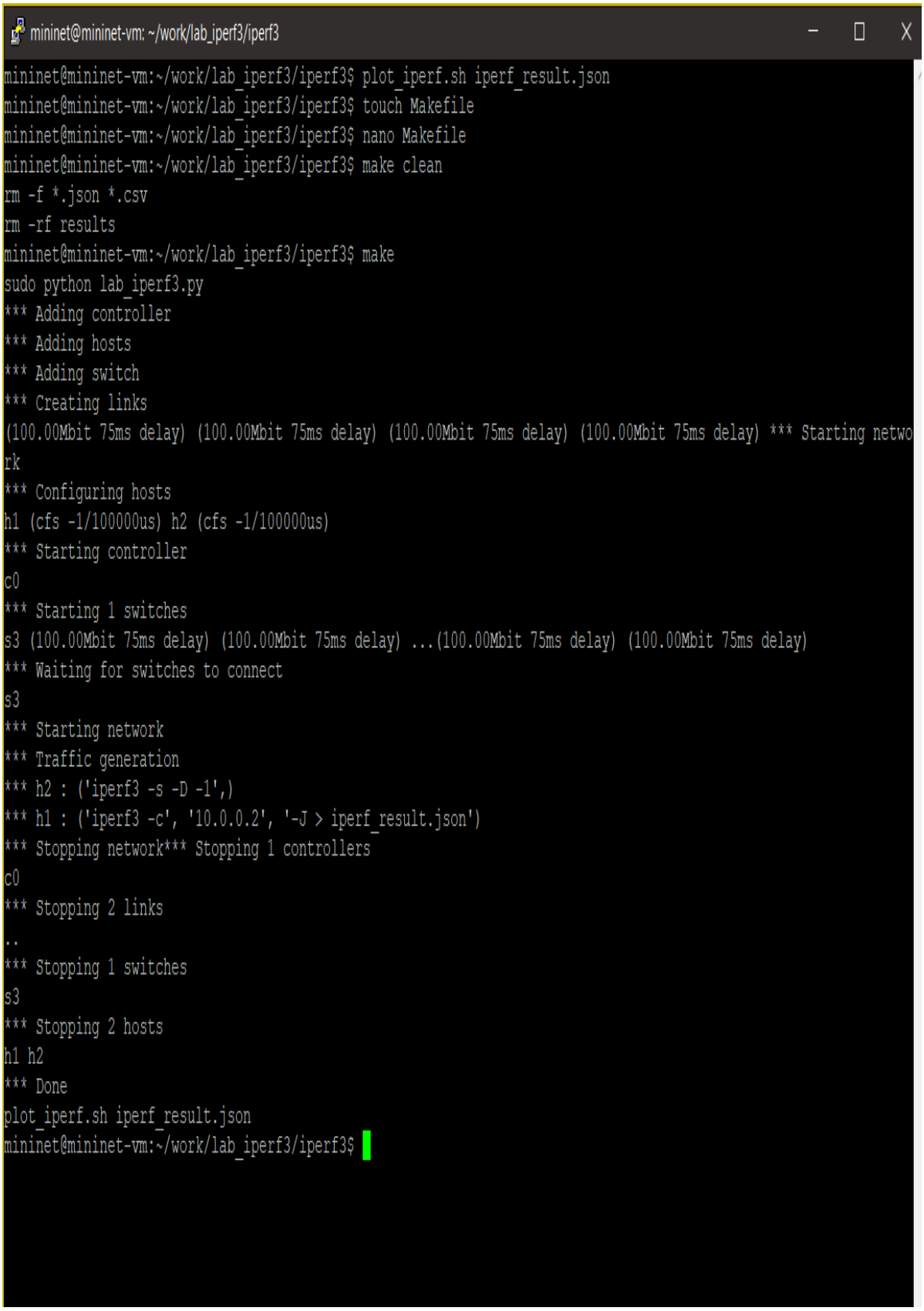

Рис. 4.12: Проверка работы Makefile

5 Выводы

В ходе выполнения лабораторной работы я познакомился с инструментом для измерения пропускной способности сети в режиме реального времени — iPerf3, а также получение навыков проведения интерактивного эксперимента по измерению пропускной способности моделируемой сети в среде Mininet.

Список литературы

1. Mininet [Электронный ресурс]. Mininet Project Contributors. URL: https://mininet.org/ (дата обращения: 07.10.2025).
2. Iperf [Электронный ресурс]. iPerf - The ultimate speed test tool for TCP, UDP; SCTP. URL: https://iperf.fr/ (дата обращения: 07.10.2025).