Задание 1. Клиент-серверное приложение.

Написать программу, реализующую простое клиент-серверное приложение: клиент устанавливает связь с сервером, расположенным по заранее определённому адресу, на что сервер посылает ему сообщение, в котором сообщает адрес клиента. (Пример устройства подобного приложения приведён в главе 2.7 книги «Компьютерные сети. Нисходящий подход»).

Программа должна быть написана на языке Python. История коммитов Git должна содержать не менее 3 коммитов, сделанных на разных этапах разработки. Код программы должен быть проверен инструментом pylint (убедитесь, что версия инструмента поддерживает Python3) с исправлением указанных замечаний.

В режиме сервера (флаг ‘-s’) программа:

- слушает введённый адрес в бесконечном цикле;
- при получении запроса отправляет ответ, содержащий ip адрес и порт клиента.

В режиме клиента программа:

- посылает запрос серверу по указанному адресу;
- ждёт ответ сервера;
- выводит полученный ответ в формате <адрес>:<порт>;
- завершает свою работу.

В зависимости от флагов запуска связь устанавливается через сокеты по протоколу TCP или UDP. Протоколу TCP соответствует флаг ‘-t’, UDP - ‘-u’. Программа не может использовать одновременно два протокола, поэтому одновременное использование флагов ‘-t’ и ‘-u’ недопустимо, программа должна проверять корректность переданных параметров при разборе введенной команды. По умолчанию используется TCP (в случае если не указан ни один параметр).

Необходимо добавить в программу поддержку логирования (модуль logging) по следующим каналам:

- стандартный вывод (stdout) по флагу ‘-o’ или по умолчанию;
- файл <file> по флагу -f <file>.

Программа должна логировать следующие события:
- Открытие (привязка) сокета;
- Отправка сообщения;
- Получение сообщения;
- Закрытие сокета.

Запуск программы: python3 kmb.py <host> <port> [-s] [-t | -u] [-o | -f <file>]

Параметры:

- <host> - хост (ip адрес) сервера;

- <port> - номер порта сервера;
- -s - запуск программы в режиме сервера;
    - Сервер открывает сокет по указанному адресу (хост + порт);
    - Клиент устанавливает соединение с сервером по указанному адресу (хост + порт);
- -t - связь по протоколу TCP;
- -u - связь по протоколу UDP;
- -o - логирование в стандартный вывод;
- -f <file> - логирование в файл <file>.

Пример: python3 kmb.py 127.0.0.1 13000 -s -t -f log_output

- запуск программы в режиме сервера по протоколу TCP с записью логов в файл log_output;
- программа открывает сокет по адресу 127.0.0.1:13000.

Результаты работы.

Ссылка на github-репозиторий, содержащий код написанной программы в файле kmb.py;

Задание 2. Трассы Wireshark.

С помощью Wireshark сделать сетевые трассы взаимодействия клиента и сервера программы из задания 1. Отфильтровать примеры одного сеанса взаимодействия для каждого из транспортных протоколов (TCP, UDP) и сохранить в отдельные файлы. Метки времени должны быть корректны и отражать реальный порядок пакетов. TCP-соединения должны присутствовать целиком, включая SYN и FIN пакеты.

Материалы по Wireshark [1], [2].

Результаты работы.

Письмо на адрес insecon@ispras.ru с двумя прикреплёнными файлами:

- файл tcp.pcap, содержащий трассу взаимодействия по протоколу TCP;
- файл udp.pcap, содержащий трассу взаимодействия по протоколу UDP;

Задание 3. Статья.

Составить небольшой (около одной страницы) конспект предоставленной вам статьи. Он должен представлять из себя связный рассказ и включать в себя все основные моменты статьи или фрагмента статьи (если вам был выдан фрагмент).

Результаты работы.

Письмо на адрес insecon@ispras.ru с прикреплённым файлом, содержащим конспект статьи;