Практика 3. Прикладной уровень (сдать до 12.03.2022)

1. Программирование сокетов. Веб-сервер

А. Однопоточный веб-сервер (3 балла)

Вам необходимо разработать простой веб-сервер, который будет возвращать содержимое локальных файлов по их имени. В этом задании сервер умеет обрабатывать только один запрос и работает в однопоточном режиме. Язык программирования вы можете выбрать любой.
Требования:

- веб-сервер создает сокет соединения при контакте с клиентом (браузером)
- получает HTTP-запрос из этого соединения
- анализирует запрос, чтобы определить конкретный запрашиваемый файл
- находит запрошенный файл в своей локальной файловой системе
- создает ответное HTTP-сообщение, состоящее из содержимого запрошенного файла и предшествующих ему строк заголовков
- отправляет ответ через TCP-соединение обратно клиенту
- если браузер запрашивает файл, которого нет на веб-сервере, то сервер должен вернуть сообщение об ошибке «404 Not Found»

Ваша задача – разработать и запустить свой локальный веб-сервер, а затем проверить его работу при помощи отправки запросов через браузер.
Скорее всего порт 80 у вас уже занят, поэтому вам необходимо использовать другой порт для работы вашей программы.

Б. Многопоточный веб-сервер (2 балла)

Реализуйте *многопоточный* сервер, который мог бы обслуживать несколько запросов одновременно. Сначала создайте основной поток (процесс), в котором ваш модифицированный сервер ожидает клиентов на определенном фиксированном порту. При получении запроса на TCP-соединение от клиента он будет устанавливать это соединение через другой порт и обслуживать запрос клиента в отдельном потоке. Таким образом, для каждой пары запрос-ответ будет создаваться отдельное TCP-соединение в отдельном потоке.

В. Клиент (2 балла)

Вместо использования браузера напишите собственный HTTP-клиент для тестирования вашего веб-сервера. Ваш клиент будет поддерживать работу с командной строкой, подключаться к серверу с помощью TCP-соединения, отправлять ему HTTP-запрос с помощью метода GET и отображать ответ сервера в качестве результата. Клиент должен будет в качестве входных параметров принимать аргументы командной строки, определяющие IP-адрес или имя сервера, порт сервера и имя файла на сервере. Формат команды для запуска клиента следующий:
client.exe хост_сервера порт_сервера имя_файла

Г. Ограничение потоков сервера (3 балла)

Пусть ресурсы вашего сервера ограничены и вы хотите контролировать максимальное количество потоков, с которыми может работать ваш многопоточный сервер одновременно. При запуске сервер считывает целочисленное значение concurrencyLevel (из командной строки, из конфигурационного файла или еще каким-либо образом). Если сервер получает запрос от клиента и при этом максимальное количество потоков уже запущено, то запрос от клиента блокируется (встает в очередь) и дожидается, пока не закончит работу один из запущенных потоков. После этого сервер может запустить новый поток для обработки запроса от клиента.

Д. Свой пул потоков (* - опциональное задание) (5 баллов)

Создание потока стоит довольно дорого. Поэтому может оказаться более эффективным переиспользовать потоки. Т.е. пока общее количество потоков, которые были порождены веб-сервером, не достигло значения concurrencyLevel, веб-сервер продолжает создавать новый поток каждый раз при новом запросе от клиента. Как только уже создано concurrencyLevel потоков, они переиспользуются для обработки новых запросов от клиентов, а не уничтожаются после того, как запрос был отработан (как это делается в предыдущем задании Г).

В этом задании предполагается, что вы не будете использовать готовый Thread Pool, а предложите свое решение. В качестве примера можно посмотреть сюда: https://stackoverflow.com/questions/5826981/how-to-reuse-threads-in-net-3-5

2. Задачки

Задание 1 (2 балла)

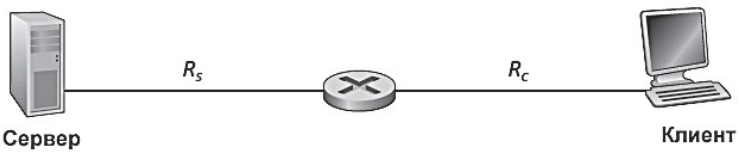
Голосовые сообщения отправляются от хоста А к хосту Б в сети с коммутацией пакетов в режиме реального времени. Хост А преобразует на лету аналоговый голосовой сигнал в цифровой поток битов, имеющий скорость 128 Кбит/с, и разбивает его на 56-байтные пакеты. Хосты А и Б соединены одной линией связи, в которой скорость передачи данных равна 1 Мбит/с, а задержка распространения составляет 5 мс. Как только хост А собирает пакет, он посылает его на хост Б, который, в свою очередь, при получении всего пакета преобразует биты в аналоговый сигнал. Сколько времени проходит с момента создания бита (из исходного аналогового сигнала на хосте А) до момента его декодирования (превращения в часть аналогового сигнала на хосте Б)?

Задание 2 (2 балла)

Рассмотрим буфер маршрутизатора, где пакеты хранятся перед передачей их в исходящую линию связи. В этой задаче вы будете использовать широко известную из теории массового обслуживания (или теории очередей) формулу Литтла. Пусть **N** равно среднему числу пакетов в буфере плюс пакет, который передается в данный момент. Обозначим через **a** скорость поступления пакетов в буфер, а через **d** – среднюю общую задержку (т.е. сумму задержек ожидания и передачи), испытываемую пакетом. Согласно формуле Литтла N = a × d. Предположим, что в буфере содержится в среднем 10 пакетов, а средняя задержка ожидания для пакета равна 10 мс. Скорость передачи по линии связи составляет 100 пакетов в секунду. Используя формулу Литтла, определите среднюю скорость поступления пакета в очередь, предполагая, что потери пакетов отсутствуют.

Задание 3 (2 балла)

Рассмотрим рисунок



Предположим, нам известно, что на маршруте от сервера до клиента узким местом является первая линия связи, скорость передачи данных по которой равна R_S бит/с. Допустим, что мы отправляем два пакета друг за другом от сервера клиенту, и другой

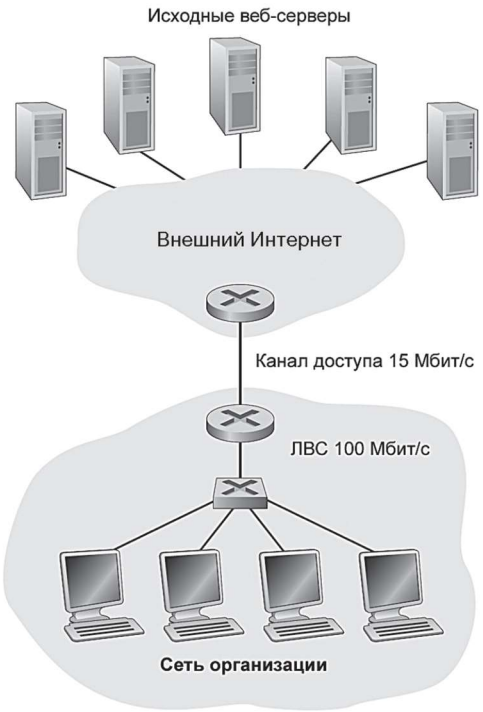
трафик на маршруте отсутствует. Размер каждого пакета составляет L бит, а скорость распространения сигнала по обеим линиям равна d_распростр.

а. Какова временная разница прибытия пакетов к месту назначения? То есть, сколько времени пройдет от момента получения клиентом последнего бита первого пакета до момента получения последнего бита второго пакета?

б. Теперь предположим, что узким местом является вторая линия связи (то есть R_C < R_S). Может ли второй пакет находиться во входном буфере, ожидая передачи во вторую линию? Почему? Если предположить, что сервер отправляет второй пакет, спустя T секунд после отправки первого, то каково должно быть минимальное значение T, чтобы очередь во вторую линию связи была нулевая? Обоснуйте ответ.

Задание 4 (4 балла)

На рисунке показана сеть организации, подключенная к Интернету:



Предположим, что средний размер объекта равен 850000 бит, а средняя скорость запросов от браузеров этой организации к веб-серверам составляет 16 запросов в секунду. Предположим также, что количество времени, прошедшее с момента, когда внешний маршрутизатор организации пересылает запрос HTTP, до момента, пока он не получит ответ, равно в среднем три секунды. Будем считать, что общее среднее время ответа равно сумме средней задержки доступа (то есть, задержки от маршрутизатора в Интернете до маршрутизатора организации) и средней задержки в Интернете. Для средней задержки доступа используем формулу $\Delta/(1-\Delta*B)$, где Δ – это среднее время, необходимое для отправки объекта по каналу связи, а B – частота поступления объектов в линию связи.

a. Найдите Δ (это среднее время, необходимое для отправки объекта по каналу связи).
b. Найдите общее среднее время ответа.
c. Предположим, что в локальной сети организации присутствует кэширующий сервер. Пусть коэффициент непопадания в кэш равен 0,4. Найдите общее время ответа.