Санкт-Петербургский политехнический
университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Отчёт по лабораторной работе
Дисциплина: Сети и телекоммуникационные технологии
Тема: Организация сетевого взаимодействия. Протокол TCP

Выполнил студент группы 43501/1 _____ К.О. Поляков
(подпись)

Преподаватель _____ А.О.Алексюк
(подпись)

8 января 2018 г.

Санкт-Петербург
2017

Содержание

1 Цель работы

Изучение принципов программирования сокетов с использованием протокола TCP.

2 Индивидуальное задание

разработать приложение–клиент и приложение–сервер службы каталогов. Всякая запись в каталоге состоит из одного или не- скольких атрибутов и обладает уникальным именем. Уникальное имя со- стоит из одного или нескольких относительных уникальных имен, разде- лённых запятой (например, “ cn=Users, dc=myserver, dc=myprovider, dc=ru ”).

Основные возможности: Серверное приложение должно реализовывать следующие функции:

1. Прослушивание определенного порта
2. Обработка запросов на подключение по этому порту клиентов
3. Поддержка одновременной работы нескольких клиентов с использованием механизма нитей и средств синхронизации доступа к разделяемым между нитями ресурсам.
4. Прием запросов на поиск, добавление и удаление записей службы каталогов
5. Осуществление поиска, добавления, удаления записей службы ката- логов
6. Передача клиенту записей службы каталогов и подтверждений о вставке и удалении записей
7. Обработка запроса на отключение клиента

Клиентское приложение должно реализовывать следующие функции:

1. Возможность параллельной работы нескольких клиентов с одного или нескольких IP-адресов
2. Установление соединения с сервером (возможно, с регистрацией на сервере)
3. Разрыв соединения
4. Передача запросов о поиске, добавлении, удалении записей серверу
5. Передача команды на удаление валюты
6. Разрыв соединения
7. Обработка ситуации отключения клиента сервером

Настройка приложений:

Разработанное клиентское приложение должно предоставлять пользователю возможность задания IP-адреса или доменного имени сервера, а также номера порта сервера.

3 Разработанный прикладной протокол

Протокол TCP имеет следующий шаблон сообщения:

<команда> <аттрибут>

В начале сообщения, всегда присутствует тип команды, далее взависимости от команды могут идти(взависимости от типа команды) аттрибуты, которые отделены друг от друга пробелом.

Список команд, которыми оперирует клиент:

Add New Entity:	add <path>
Delete Entity:	del <path>
Read All Entities:	all
Find full path for Entity:	fnd <part of path>
Show help:	help
Exit :	exit

3.1 Описание структуры приложения

Сервер:
Функция main:

- Инициализация всех используемых переменных;
- Запуск событий, если таковые уже имеются;
- Создание сокета;
- Создание потока для прослушивания сокета;

Поток для прослушивания сокета:

- Прослушивание сокета;
- Добавление подключенного клиента в список;
- Создание сокета для работы с клиентом;
- Создание потока для работы с клиентом.

Цикл чтения команд:

- Чтение команды из стандартного ввода;
- Реакция на введенную команду (при корректном вводе команды) или вывод предупреждения.

Клиент:
Функция main:

- Чтение ip адреса сервера;
- Подключение к серверу;
- Создание потока для получения данных от сервера;
- Цикл чтения данных и отправка серверу.

Поток для получения данных от сервера:

- Побайтовое получение данных от сервера.

4 Реализация программы

4.1 Структура проекта

При разработке приложения для операционной системы семейства Windows использовалась среда разработки CLion.

Язык программирования — C.

4.2 Сетевая часть TCP

Клиентское приложение в TCP только отсылает команды на сервер, поэтому оно ничем не отличается от telnet клиента. Сервер обрабатывает команды, работает с коллекциями, сохраняет и загружает свое состояние, присылает уведомления и др. Делаем вывод, что клиентская программа потребляет ничтожно малый процессорный ресурс, в то время как сервер - наоборот.

На сервере, в первую очередь, происходит инициализация WinSock (на Windows), создание сокета (функция socket), привязка сокета к конкретному адресу (функция bind).

После этого ожидаем подключения клиентов в бесконечном цикле, с помощью функции accept. Если функция возвращает положительное значение, которое является клиентским сокетом, то создаем новый поток, в котором обрабатываем клиентские сообщения.

Клиентский поток вызывает функцию считывания символов в бесконечном цикле, как только при считывается знак перевода строки, функция возращает прочитанные символы. Если функция не вернула исключение, то посылаем команду на обработку, в противном случае это обозначает отключение клиента.

5 Вывод

В ходе работы были изучены принципы программирования сокетов с использованием протокола TCP.
В рамках модели клиент (linux) и сервер (windows). Были изучены и использованы разные библеотеки для построения работы сокетов. Серверное прилоежние имеет несколько потоков-обработчиков, для успешной работы с несколькими клиентами единовременно. Протокол TCP упростил логику приложения, переложив работу с неправильной передачей данных на себя. Так же в ходе разработки приложения была решениа проблема синхронизации данных в многопоточной среде.То есть были получены навыки организации многопоточного сервера, изучены принципы синхронизации доступа к глобальным переменным.
В разработанном в ходе работы сервере для каждого клиента создается отдельный поток. Такой подход оправдан, т.к. клиенты могут исполнять долгие операции и операции различной трудоемкости. В этом случае использование отдельного потока для каждого клиента обеспечивает минимизацию взаимного влияния клиентов друг на друга.