ft_irc

Интернет-релейный чат

Резюме:
Этот проект посвящен созданию собственного IRC-сервера.
Вы будете использовать настоящий IRC-клиент для подключения к вашему серверу и его тестирования.
Интернет управляется протоколами твердых стандартов, которые позволяют подключенным компьютерам взаимодействовать друг с другом.
Это всегда полезно знать.

Версия: 6

Содержание

Глава I

Введение

Internet Relay Chat или IRC - это текстовый протокол общения в Интернете. Он предлагает обмен сообщениями в реальном времени, который может быть как публичным, так и приватным. Пользователи могут обмениваться прямыми сообщениями и присоединяться к групповым каналам.

IRC-клиенты подключаются к IRC-серверам, чтобы присоединиться к каналам. IRC-серверы соединяются вместе, образуя сеть.

Глава II Общие правила

- Ваша программа не должна аварийно завершаться ни при каких обстоятельствах (даже когда у нее закончится память), и не должна неожиданно завершаться.
  Если это произойдет, ваш проект будет считаться нефункциональным, и ваша оценка будет
  0.
- Вы должны создать Makefile, который будет компилировать ваши исходные файлы. Он не должен перелинковываться.
- Ваш Makefile должен, по крайней мере, содержать эти правила:
  $(NAME), all, clean, fclean и re.
- Скомпилируйте ваш код с помощью c++ и флагов -Wall -Wextra -Werror
- Ваш код должен соответствовать **стандарту C++ 98**. Тогда он должен компилироваться, если вы добавите флаг -std=c++98.
- Старайтесь всегда разрабатывать с использованием наиболее доступных функций C++ (например, выбирайте
  <cstring> над <string.h>). Вы можете использовать функции языка C, но всегда предпочитайте их версии на C++, если это возможно.
- Любые внешние библиотеки и библиотеки Boost запрещены.

Глава III

Обязательная часть

Название программы	ircserv
Сдать файлы	Makefile, *.{h, hpp}, *.cpp, *.tpp, *.ipp, необязательный файл конфигурации
Makefile	NAME, all, clean, fclean, re
Аргументы	порт:Порт прослушивания пароль:Пароль подключения
Внешние функции.	Все на C++ 98. socket, setsockopt, getsockname, getprotobyname, gethostbyname, getaddrinfo, freeaddrinfo, bind, connect, listen, accept, htons, htonl, ntohs, ntohl, inet_addr, inet_ntoa, send, recv, signal, lseek, fstat, fcntl, poll (или эквивалент)
Либфт уполномочен	н/а
Описание	IRC-сервер на C++ 98

Вам необходимо разработать IRC-сервер на C++ 98.

Вы **не должны** развивать клиента.
Вы **не должны** управлять взаимодействием между серверами.

Ваш исполняемый файл будет запущен следующим образом:

```
./ircserv <порт> <пароль>
```

- порт: Номер порта, на котором ваш IRC-сервер будет прослушивать входящие IRC-соединения.
- password: Пароль подключения. Он потребуется любому IRC-клиенту, который попытается подключиться к вашему серверу.

Даже если poll() упоминается в теме и шкале оценки, вы можете использовать любой эквивалент, такой как select(), kqueue() или epoll().

III.1 Требования

- Сервер должен быть способен работать с несколькими клиентами одновременно и никогда не зависать.
- Форкинг не допускается. Все операции ввода-вывода должны быть **неблокирующими**.
- Для обработки всех этих операций (чтение, запись, а также прослушивание и так далее) можно использовать только **1** poll() (или эквивалент).

Поскольку вы должны использовать неблокирующие дескрипторы файлов, можно использовать функции read/recv или write/send без poll() (или эквивалента), и ваш сервер не будет блокироваться.
Но это потребует больше системных ресурсов.
Таким образом, если вы попытаетесь прочитать/отправить или записать/отправить в любом файловом дескрипторе без использования poll() (или эквивалента), ваша оценка будет равна 0.

- Существует несколько клиентов IRC. Вы должны выбрать одного из них в качестве **эталонного**. Ваш эталонный клиент будет использоваться в процессе оценки.
- Ваш эталонный клиент должен иметь возможность подключиться к вашему серверу без каких-либо ошибок.
- Связь между клиентом и сервером должна осуществляться через TCP/IP (v4 или v6).
- Использование вашего эталонного клиента с вашим сервером должно быть аналогично его использованию с любым официальным IRC-сервером. Однако вы должны реализовать только следующие функции:
  - Вы должны уметь проходить аутентификацию, устанавливать псевдоним, имя пользователя, присоединяться к каналу, отправлять и получать личные сообщения с помощью своего справочного клиента.
  - Все сообщения, отправленные от одного клиента в канал, должны быть пересланы каждому другому клиенту, присоединившемуся к каналу.
  - У вас должны быть операторы и постоянные пользователи.
  - Затем необходимо реализовать команды, характерные для операторов.
- Конечно, от вас ожидается, что вы будете писать чистый код.

III.2 Только для MacOS

Поскольку в MacOS функция write() реализована не так, как в других ОС Unix, вам разрешено использовать fcntl().
Вы должны использовать файловые дескрипторы в неблокирующем режиме, чтобы получить поведение, похожее на поведение других ОС Unix.

Однако вам разрешено использовать fcntl() только следующим образом: fcntl(fd, F_SETFL, O_NONBLOCK);
Любой другой флаг запрещен.

III.3 Пример испытания

Проверьте абсолютно все возможные ошибки и проблемы (получение частичных данных, низкая пропускная способность и так далее).

Чтобы убедиться, что ваш сервер правильно обрабатывает все, что вы ему посылаете, можно выполнить следующий простой тест с использованием nc:

```
\$> nc 127.0.0.1 6667
com^Dman^Dd
\$>
```

Используйте ctrl+D для отправки команды в нескольких частях: 'com', затем 'man', затем 'd\n'.

Чтобы обработать команду, необходимо сначала агрегировать полученные пакеты для ее восстановления.

Глава IV

Бонусная часть

Вот дополнительные возможности, которые вы можете добавить к своему IRC-серверу, чтобы он стал еще больше похож на настоящий IRC-сервер:

- Передача файлов.
- Бот.

Бонусная часть оценивается только в том случае, если обязательная часть выполнена безупречно. Совершенство означает, что обязательная часть выполнена полностью и работает без сбоев. Если вы не выполнили ВСЕ обязательные требования, ваша бонусная часть не будет оцениваться вообще.

Глава V

Представление и экспертная оценка

Сдайте задание в свой Git-репозиторий, как обычно. Во время защиты будет оцениваться только работа, находящаяся в вашем репозитории. Не стесняйтесь дважды проверять имена файлов, чтобы убедиться в их правильности.

Вам рекомендуется создавать тестовые программы для своего проекта, даже если они **не будут представлены и не будут оценены**. Эти тесты могут быть особенно полезны для проверки вашего сервера во время защиты, а также для вашего коллеги, если однажды вам придется оценивать другой ft_irc. Действительно, вы можете использовать любые тесты, которые вам нужны в процессе оценки.

Ваш референс-клиент будет использоваться в процессе оценки.