

ЦЕЛЬ РАБОТЫ

Следует реализовать приложение на C99/C++11, которое должно уметь:

1. Получать данные из socket.
2. Передавать данные в socket.
3. Идентифицировать отправителя/получателя.
4. Вести историю передаваемых данных.

ОСНОВНАЯ ЧАСТЬ

1.1 Теоретическая часть

unix-socket (англ. Unix domain socket, UDS) — конечная точка обмена данными, подобная Интернет-сокету, но не использующая сетевой протокол для взаимодействия (обмена данными). Используется в операционных системах, поддерживающих стандарт POSIX, для межпроцессного взаимодействия.

Доменные соединения Unix являются по сути байтовыми потоками, сильно напоминая сетевые соединения, но при этом все данные остаются внутри одного компьютера (то есть обмен данными происходит локально). UDS используют файловую систему как адресное пространство имен, то есть они представляются процессами как иномы в файловой системе. Это позволяет двум различным процессам открывать один и тот же сокет для взаимодействия между собой. Однако, конкретное взаимодействие, обмен данными, не использует файловую систему, а только буферы памяти ядра.

1.2 Практическая часть

Было создано приложение на C99/C++11, которое действует по сценарию:

1. Сценарий сервиса по подсчету символов в слове, представленный в приложении А

1. Создается unix-сокет, связывается с `./socket` и начинает ждать подключений.
2. При подключении пользователя сервис просит его авторизоваться.
3. После авторизации пользователя сервис выдает правила своей работы и ждет ввода.

2. Сценарий пользователя, представленный в приложении Б

1. Обращается к созданному сервисом unix-сокету, связанному с `./socket`.
2. Вводит свой login по просьбе сервиса.
3. Производит ввод слов.

Есть три вида ввода пользователя:

- Слово - подсчет символов в данном слове. При вводе несколько слов (последовательность символов через пробел), сервис подсчитает символы в каждом слове отдельно.
- `logout` - отключение текущего пользователя и ожидание нового подключения.
- `end` - завершение работы сервиса.

Ввод пользователя и ответ сервиса выводятся в консоль в виде:

`Service -> user : сообщение`

`user -> Service : сообщение`

```

hruleva@DebianHruleva:~$ cd lab5
hruleva@DebianHruleva:~/lab5$ ./client
Service -> unknown: Здравствуйте, как можно к вам обращаться?
alena
alena -> Service : alena
Service -> alena :
1. Сервис выдает количество букв в слове. При вводе нескольких слов, подсчитает для каждого.
2. Закончить работу пользователю "logout"
3. Закончить работу с сервисом - "end"
Вы были авторизованы как alena.
Введите строку
milk
alena -> Service : milk
Service -> alena :
В слове milk 4 символов.
Введите новое слово
red
alena -> Service : red
Service -> alena :
В слове red 3 символов.
Введите новое слово
blue
alena -> Service : blue
Service -> alena :
В слове blue 4 символов.
Введите новое слово
possible
alena -> Service : possible
Service -> alena :
В слове possible 8 символов.
Введите новое слово
logout
alena -> Service : logout
hruleva@DebianHruleva:~/lab5$ ./client
Service -> unknown: Здравствуйте, как можно к вам обращаться?
effy
effy -> Service : effy

```

Рисунок 1 – Пример работы сервиса с пользователем

```

logout
alena -> Service : logout
hruleva@DebianHruleva:~/lab5$ ./client
Service -> unknown: Здравствуйте, как можно к вам обращаться?
effy
effy -> Service : effy
Service -> effy :
1. Сервис выдает количество букв в слове. При вводе нескольких слов, подсчитает для каждого.
2. Закончить работу пользователю "logout"
3. Закончить работу с сервисом - "end"
Вы были авторизованы как effy.
Введите строку
limon
effy -> Service : limon
Service -> effy :
В слове limon 5 символов.
Введите новое слово
black
effy -> Service : black
Service -> effy :
В слове black 5 символов.
Введите новое слово
moscow
effy -> Service : moscow
Service -> effy :
В слове moscow 6 символов.
Введите новое слово
end
effy -> Service : end
hruleva@DebianHruleva:~/lab5$ █

```

Рисунок 2 – Пример работы сервиса с пользователем

```

hruleva@DebianHruleva:~/lab5$ ./server
Подключен к ./socket
Есть новое подключение
Service -> unknown: Здравствуйте, как можно к вам обращаться?
alena -> Service : alena
Service -> alena :
1. Сервис выдает количество букв в слове. При вводе нескольких слов, подсчитает для каждого.
2. Закончить работу пользователю "logout"
3. Закончить работу с сервисом - "end"
Вы были авторизованы как alena.
Введите строку
alena -> Service : milk
Service -> alena :
В слове milk 4 символов.
Введите новое слово
alena -> Service : red
Service -> alena :
В слове red 3 символов.
Введите новое слово
alena -> Service : blue
Service -> alena :
В слове blue 4 символов.
Введите новое слово
alena -> Service : possible
Service -> alena :
В слове possible 8 символов.
Введите новое слово
alena -> Service : logout
Service -> alena : Подключение пользователя alena остановлено
Есть новое подключение
Service -> unknown: Здравствуйте, как можно к вам обращаться?
effy -> Service : effy
Service -> effy :
1. Сервис выдает количество букв в слове. При вводе нескольких слов, подсчитает для каждого.
2. Закончить работу пользователю "logout"
3. Закончить работу с сервисом - "end"
Вы были авторизованы как effy.
Введите строку
effy -> Service : limon
Service -> effy :
В слове limon 5 символов.
Введите новое слово
effy -> Service : black
Service -> effy :
В слове black 5 символов.
Введите новое слово
effy -> Service : moscow
Service -> effy :
В слове moscow 6 символов.
Введите новое слово
effy -> Service : end
Service -> effy : Закрытие сервиса
hruleva@DebianHruleva:~/lab5$

```

Рисунок 3 – Пример работы сервиса с пользователем

ВЫВОДЫ

Таким образом, в ходе лабораторной работы были изучены unix-sockets и реализовано приложение на C99/C++11, осуществляющее работу с unix-socket.