Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

НА ТЕМУ:

Разработка чата для локальной сети с использованием алгоритма сквозного шифрования

Студент ИУ8-32 _____ Е. А. Чумичкина
(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсового проекта _____ А.Ю.Быков
(Подпись, дата) (И.О.Фамилия)

Консультант _____ _____
(Подпись, дата) (И.О.Фамилия)

2021 г.

Оглавление

Введение

Мессенджеры в современном мире обрели огромную популярность. Локальные мессенджеры используются на предприятиях, в школах, в офисах.

Данная программа позволяет общаться по локальной сети, не имея доступа к интернету.

Цель проекта заключается в создании проекта для отправки и получения сообщений по локальной сети, сообщения должны зашифровываться и расшифровываться алгоритмом асинхронного шифрования RSA.

Для достижения цели необходимо решить следующие задачи:

- изучить предметную область, посвященную разработке интерфейса пользователя;
- разработать техническое задание для проектирования программного обеспечения;
- выбрать язык программирования и среду разработки;
- выполнить разработку программы;
- выполнить тестирование и отладку программы;
- разработать описание программы.

Техническое задание

Введение

Настоящее техническое задание распространяется на разработку программной системы «Локальный чат с шифрованием», предназначенный для передачи и получения сообщений по локальной сети. Предполагается, что использовать данную систему будут школьники, студенты, офисные сотрудники для личных целей, имеющие начальные навыки работы с компьютером.

Разрабатываемая программная система позволит пользователям обмениваться короткими сообщениями без подключения к Интернету.

Основание для разработки

Система разрабатывается в качестве курсовой работы по курсу «Технологии и методы программирования» в соответствии с учебным планом кафедры «Информационная безопасность».

Назначение

Основное назначение программы является помощь непрофессиональным пользователям в обмене сообщениями с шифрованием без использования Интернета.

Требования к программной системе

Требования к функциональным характеристикам

Программа должна обеспечивать выполнение следующих функций:

- Ввод имени пользователя
- Создание сервера для начала общения
- Выбор порта обмена сообщениями
- Подключение к существующему серверу как клиент
- Отправить зашифрованное сообщение
- Получить расшифрованного сообщение
- Выход из чата
- Закрыть чат для всех
- Получить информацию о количестве человек в чате

Исходные данные

- Имя пользователя
- Номер порта

Требования к надежности

- Предусмотреть контроль вводимой информации
- Предусмотреть блокировку некорректных действий пользователя при работе с системой

Требования к информационной и программной совместимости

Система должна работать под управлением семейства операционных систем Windows.

Требования к программной документации

- Разрабатываемые программные модули должны быть самодокументированы, то есть тексты программ должны содержать все необходимые комментарии
- В состав сопровождающей документации должны входить:
  - Пояснительная записка, содержащая описание разработки
  - Руководство пользователя
  - Тексты программ

Техническое задание диктует следующие принципиальные решения начального этапа:

- Архитектура – многопользовательская
- Класс – программная система
- Интерфейс – со свободной навигацией и максимально использующий основные решения, предлагаемые Windows, чтобы не профессиональному пользователю было легче его освоить
- Подход к разработке – объектный, что упрощает создание стандартизированного интерфейса Windows.

Описание программы

Общие сведения

Программное обеспечение для по локальной сети с использованием шифрования.

Язык программирования – C++, среда разработки – Visual Studio 2019.

Функциональное назначение

Программное обеспечение для передачи и получения сообщений по локальной сети с использованием сквозного асинхронного алгоритма шифрования RAS.

Программное обеспечение решает следующие задачи:

- Ввод имени пользователя
- Создание сервера для начала общения
- Выбор порта обмена сообщениями
- Подключение к существующему серверу как клиент
- Отправить зашифрованное сообщение
- Получить расшифрованного сообщение
- Выход из чата
- Закрыть чат для всех
- Получить информацию о количестве человек в чате

Описание логической структуры

После запуска программы открывается окно чата. В заголовке программы выведено сообщение «Нет сети!», а в строке с количеством людей в чате «В чате 0 чел.», так как ни к какому чату пользователь ещё не подключён. Кнопки «Отправить» и «Выход из чата» не доступны.

Пользователю необходимо ввести имя и номер порта, по которому он будет подключаться к чату. Если поле оставить без изменений, то программа выведет окно с сообщением об ошибке.

В случае успешного заполнения входных данных, пользователь выбирает способ подключения:

- как клиент, то есть по номеру порта к уже существующему чату;
- как сервер, то есть создать на этом порту сервер, куда могут подключиться другие пользователи.

В случае подключения как сервер:

- если по данному номеру порта уже создан чат, то программа уведомляет об этом;
- пользователь ожидает, когда к чату подключится другой пользователь;
- в поле «Отправка» пользователь пишет сообщение, нажимает кнопку «Отправить». Отправленное сообщение и имя отправителя отображается в поле «Окно чата» у всех подключенных пользователей.
- Чтобы остановить чат для всех пользователей, пользователь нажимает на кнопку «Остановить чат». Все пользователи получают сообщение: «<имя пользователя>: Чат остановлен!». Работа чата для всех пользователей завершена.

В случае подключения как клиент:

- если по данному порту нет сервера, программа сообщает об этом
- после подключения к существующему серверу, в поле «Отправка» пользователь пишет сообщение, после нажатия кнопки «Отправить» сообщение получают все пользователи чата.
- При нажатии на кнопку «Выход из чата» всем пользователям приходит сообщение «<имя пользователя> - покинула чат!». Остальные пользователи могут продолжать пользоваться чатом.

Шифрование сообщений осуществляется алгоритмом сквозного асимметричного шифрования RAS.

- Из массива 40 простых чисел случайно выбираются два: некоторые p и q;
- Вычисляется модуль всех дальнейших операций n = pq;
- Вычисляется функция Эйлера от числа n: f(n) = (p-1)(q-1);
- Выбирается число e, взаимно простое с f(n); в данном случае выбрана e=619 – достаточно большое простое число;
- Вычисляется число d, обратное e по модулю f(n)
- Формируем два ключа: {e, n} – открытый ключ и {d, n} – закрытый ключ
- Открытым ключом зашифровываем сообщение при отправке. Закрытым ключом расшифровываем при получении.

Для завершения программы необходимо закрыть окно программы.

Вызов и загрузка

Для запуска программы необходимо в папке Debug запустить файл ChatCpp.exe, после чего появится окно программы.

Входные данные

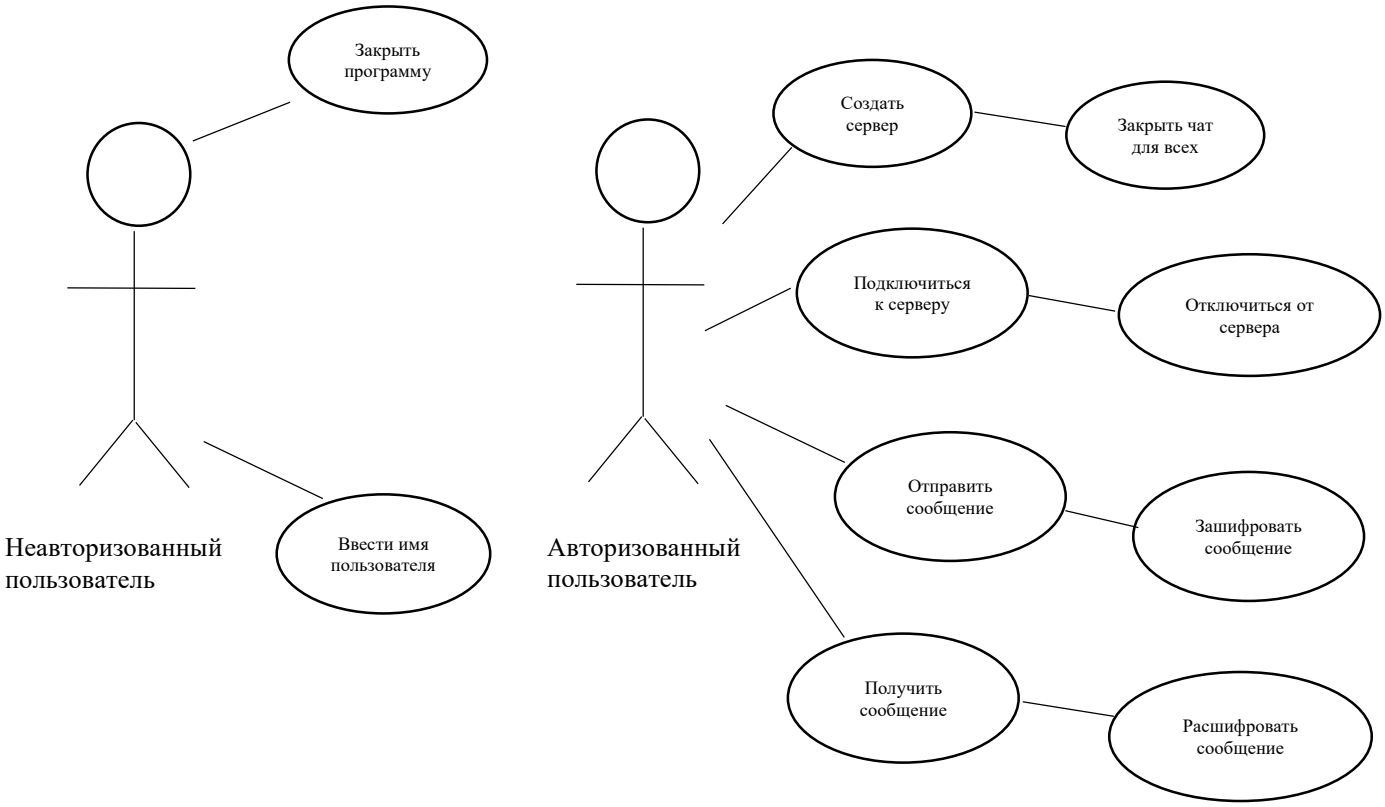
Входные данные читаются из поля программы.

Выходные данные

Выходные данные хранятся в окне программы.

Реализация проекта

Диаграмма вариантов использования



Проектирование интерфейса приложения

Проектирование интерфейса приложений в среде Visual Studio 2019 удобно делать с помощью библиотеки MFC. Она автоматически создает методы для элементов интерфейса приложения.

Окно приложения состоит из двух блоков: управление и отправка/получения сообщений.

В блоке управления находятся поля для ввода имени, IP адреса и номера порта. Кнопка выбора входа в чат: как клиент или как сервер. Кнопка «Выход из чата» позволяет закрыть чат для всех, если это выполняется со стороны сервера, или выйти из чата, если пользователь клиент.

В блоке отправки/получения сообщений находится поле с количеством людей в чате, кнопка отправки сообщения, поле ввода сообщения для отправки и окно вывода чата. Вид окна чата представлен в приложении к документу.

Проектирование классов приложения

Для передачи сообщений используется класс CSock, класс сокетов производный от класса асинхронных сокетов CAsyncSocket.

Класс ChatCppApp используется для начальной инициализации программы.

Класс ChatCppDlg описывает диалоговое окно чата и основные функции передачи и получения сообщений.

Класс ChatCppAboutDlg описывает диалоговое окно справки о программе.
Данные передаются в виде структуры SENDBUFFER, хранящая текст сообщения, имя пользователя, тип сообщения (от пользователя, о прекращении работы чата или о выходе из чата пользователя), число людей в чате, и флаг остановки чата.

Описание класса CSock:

Все методы класса сокетов производный от класса асинхронных сокетов CAsyncSocket.

- Событие подключения на стороне клиентского приложения.

```
void CSock::OnConnect(int nErrorCode)
```

- Событие отключения от сети

```
void CSock::OnClose(int nErrorCode)
```

- Событие получения сообщений.

```
void CSock::OnReceive(int nErrorCode)
```

- Запрос на подключение, направляемый клиентом серверу. Происходит на стороне серверного приложения.

void CSock::OnAccept(int nErrorCode)

Описание класса ChatCppDlg

- Инициализация чата

  ```
  BOOL CChatCppDlg::OnInitDialog()
  ```

- Запускаем сервер. Проверка правильности ввода данных. Контроль несанкционированного запуска сервера: если кнопка не в состоянии нажатой, если сокет в работе (т.е. только с нулевым сокетом можно начинать работать)

  ```
  void CChatCppDlg::OnBnClickedRadioServer()
  ```

- Запускаем клиента. Проверка правильности ввода данных. Контроль несанкционированного запуска сервера: если кнопка не в состоянии нажатой, если сокет в работе (т.е. только с нулевым сокетом можно начинать работать)

  ```
  void CChatCppDlg::OnBnClickedRadioClient()
  ```

- Нажали кнопку "Выйти из чата".

  ```
  void CChatCppDlg::OnBnClickedButtonStopchat()
  ```

- Запрещает доступ к управлениям при работе приложения в режиме сервера или клиента. Цель запрета - избежать исключения от случайного нажатия "неправильных" кнопок.

  ```
  void CChatCppDlg::DisabledControl(bool server)
  ```

- Разрешить доступ к управлениям после закрытия сокетов. Цель запрета - избежать исключения от случайного нажатия "неправильных" кнопок.

  ```
  void CChatCppDlg::EnabledControl(void)
  ```

- Принимаем запросы на подключения

  ```
  void CChatCppDlg::OnAccept(void)
  ```

- Выход из чата, если это сработало на стороне сервера, то это полная остановка чата.

  ```
  void CChatCppDlg::StopChat(void)
  ```

- Событие нажатие кнопки отправить

  ```
  void CChatCppDlg::OnBnClickedButtonSend()
  ```

- Извлечение сообщений из сети чата.

  ```
  void CChatCppDlg::OnReceive(void)
  ```

- При закрытии приложения отправим в чат информацию об отключении пользователя

  ```
  void CChatCppDlg::OnClose()
  ```

- Послать строку-сообщение в чат.

  ```
  void CChatCppDlg::SendToChat(CString strMessage)
  ```

- Послать буфер подготовленного сообщения в сеть.

  ```
  void CChatCppDlg::SendBuffer(SENDBUFFER sb, bool toserver)
  ```

- Формируем и отправляем сообщение об отключении от сети.

  ```
  void CChatCppDlg::SendDisconnect(void)
  ```

- Событие подключения, вызывается на стороне клиента.

  ```
  void CChatCppDlg::OnConnect(BOOL Error)
  ```

- Сервер отправляет клиентам количество людей в чате.

  ```
  void CChatCppDlg::SendCountPeople(void)
  ```

- Запрос имени чатующего перед созданием сокета.

  ```
  bool CChatCppDlg::QueryName(void)
  ```

Реализация алгоритма RSA

- Пользовательский тип данных, хранящий случайные простые числа

  ```
  typedef struct {
  uint64_t p;
  uint64_t q;
  }key;
  ```

- Массив простых чисел

```
static uint64_t prost[40] = {
	5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181};
```

- Карта для хранения ключей для каждого чата

```
static std::map<CString, key>keys;
```

- Генерация случайных чисел p и q

```
void GenerateKeys(const CString& s);
```

- Расширенный алгоритм Евклида

```
uint64_t gcdex(uint64_t a, uint64_t b, uint64_t& x, uint64_t& y);
```

- Возведение в квадрат типа uint64_t

```
uint64_t sqr(uint64_t x);
```

- Быстрое возведение в степень

```
uint64_t binpow(uint64_t a, uint64_t e, uint64_t mod);
```

- Зашифровка сообщения

```
CString Encrypt(const CString& s, const CString& port);
```

- Нахождение обратного по модулю числа

```
uint64_t invmod(uint64_t a, uint64_t m);
	CString Decrypt(const CString& s, const CString& port);
```

Программа и методика испытаний

Введение

Данная программа и методика испытаний разработана как руководство для проведения предварительных (комплексных испытаний) доработанного программного обеспечения.

Объект испытаний

Объектом испытаний является программа локальный чат с шифрованием «ChatCpp».

Цель испытаний

Цель испытаний состоит в проверке степени соответствия реальных характеристик разработанного программного обеспечения в соответствии техническому заданию.

Требования к программе

Требования к программе изложено в пункте 3 технического задания.

Средства и порядок испытаний

Испытания проводятся на реальных рабочих местах, на которых должен устанавливаться программа, в обычных условиях эксплуатации. Порядок испытаний: Реализация проверки функционирования программного комплекса и соответствия функциональных характеристик комплекса требованиям ТЗ.

Методы испытаний

Реализация проверки функционирования программного комплекса и соответствия функциональных характеристик комплекса требованиям ТЗ.

- Проверка ввода имени пользователя

Если поле имени пользователя не изменено, то программа должна выдать предупреждение об этом. Длина имени пользователя подразумевается не длиннее 14 символов. Любые другие комбинации символов допустимы.

- Создание сервера для начала общения

Если сервер пытаются создать на занятом порту, программа предупреждает об этом. Если порт указан верно, данный пользователь не подключён к другому чату, то сервер удачно создан.

- Выбор порта обмена сообщениями

Если пользователь подключается как клиент, а на указанном порту сервер ещё не создан, то выводится предупреждение. Если пользователь подключается как сервер, а на указанный порт занят, то выводится предупреждение. Если всё в порядке, то программа выполняется корректно.

- Подключение к существующему серверу как клиент

Если подключение выполняется по существующему порту, то программа выполняется корректно. Если указанный порт не существует, то выводится предупреждение.

- Отправить зашифрованное сообщение

Если в чате находится не менее 2 людей, то ошибок не возникает. Если менее, то кнопка отправки сообщения не доступна.

- Получить расшифрованного сообщение

Если сообщение было отправлено с того же порта, где находится получатель, то сообщение будет получено.

- Выход из чата

Если с клиента пользователь нажимает на кнопку «Выйти из чата», то чат прекращается только для этого пользователя (если в чате было 2 больше пользователей) или для всех (если в чате было только 2 пользователя) и у других пользователей выводится сообщение «<имя пользователя> - покинул(а) чат!». Программа возвращается в начальное состояние.

- Закрыть чат для всех

Если с сервера пользователь нажимает на кнопку «Остановить чат», то чат прекращается для всех пользователей на этом порту и все пользователи получают сообщение «<имя пользователя>: Чат остановлен!». Программа возвращается в начальное состояние.

- Получить информацию о количестве человек в чате

Количество пользователей отображается в поле «В чате <количество людей> чел.»

Заключение

В результате проделанной работы был разработан чат для локальной сети с использованием шифрования алгоритмом RSA – асимметричного алгоритма шифрования. Все поставленные задачи в ходе выполнения проекта были полностью решены. Разработанное ПО может быть встроено в любой персональный компьютер. В дальнейшем программа может быть улучшена добавлением книги контактов, базы данных сообщений, статусов сообщений, поддержку передачи файлов.

Приложение

Ссылка на исходный код:

Пример выполнения программы:

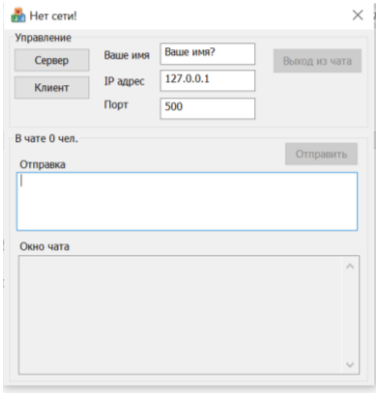


Рисунок 1. Начальное окно

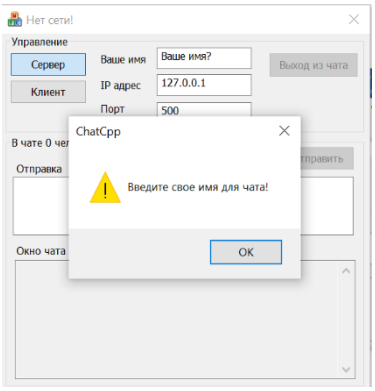


Рисунок 2. Ошибка ввода имени

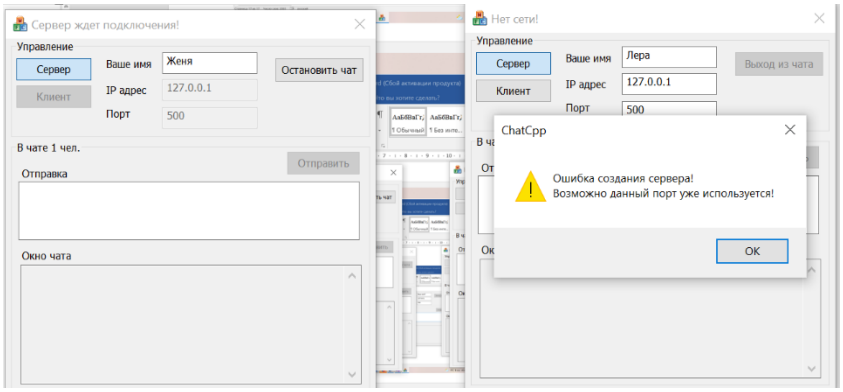


Рисунок 3. Ошибка создания сервера 1

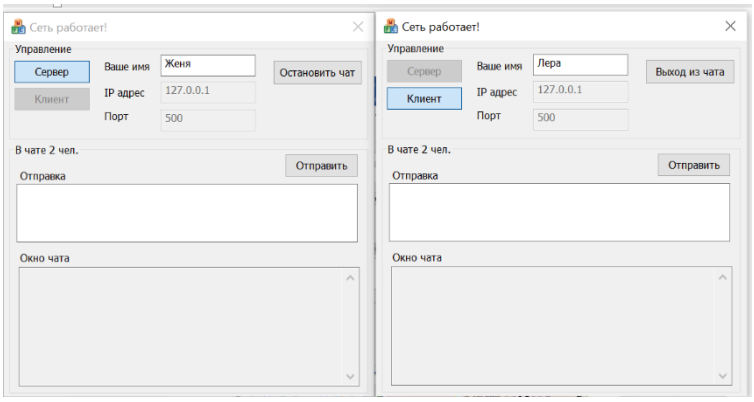


Рисунок 4. Подключение

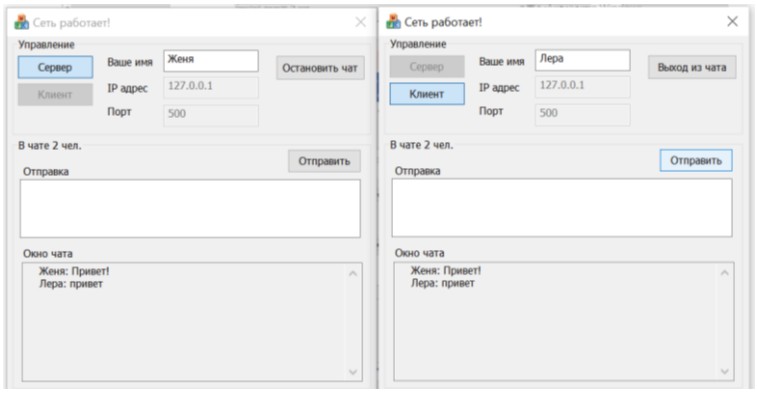


Рисунок 5. Пример сообщений

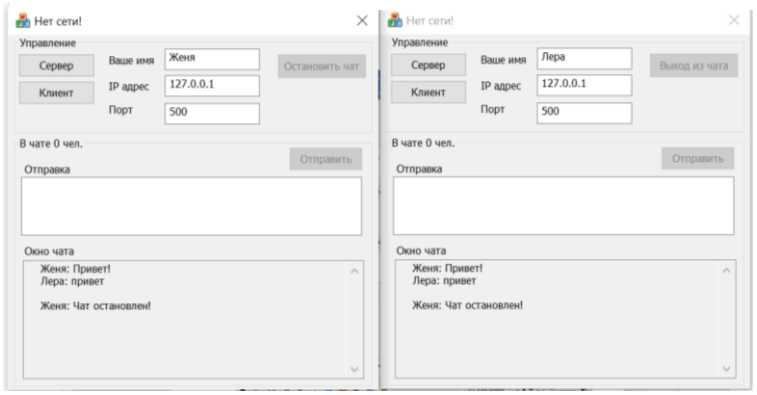


Рисунок 6. Остановка чата сервером

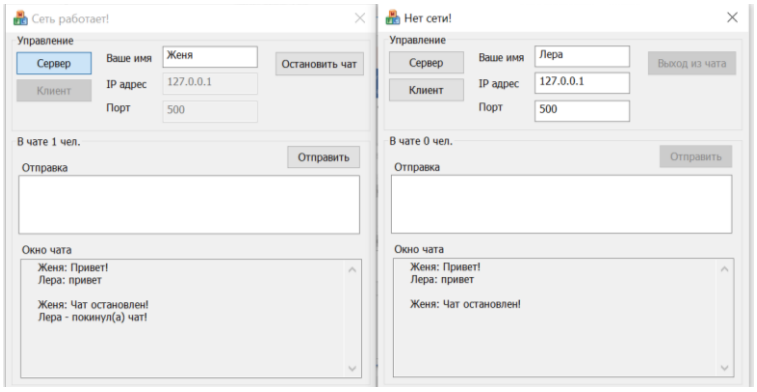


Рисунок 7 Клиент вышел из чата