МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Забайкальский государственный университет»

(ФГБОУ ВПО «ЗабГУ»)

Факультет: Энергетический

Кафедра: Информатики, вычислительной техники и прикладной математики

**КУРСОВОЙ ПРОЕКТ**

По дисциплине: Сети и телекоммуникации

На тему: FTP сервер

Выполнил студент группы ИВТ–18–1, Втулкин Владислав Викторович

Руководитель работы: ассистент кафедры ИВТиПМ

Забелин Вячеслав Олегович

Чита

2021

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Забайкальский государственный университет»

(ФГБОУ ВПО «ЗабГУ»)

Факультет: Энергетический

Кафедра: Информатики, вычислительной техники и прикладной математики

**ЗАДАНИЕ**

на курсовой проект

По дисциплине: Сети и телекоммуникации

Студенту: Втулкину Владиславу Викторовичу

Специальности (направления подготовки): Информатика и вычислительная техника

1 Тема курсовой работы: «Моделирования работы ЛВС»

2 Срок подачи студентом законченной работы: 14.06.2021

3 Исходные данные к работе: описание предметной области

Дата выдачи задания: 15.02.2021

Руководитель курсовой работы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Забелин В.О./

(подпись, расшифровка подписи)

Задание принял к исполнению

«15» февраля 2021 г.

Подпись студента\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Втулкин В.В. /

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«Забайкальский государственный университет»

(ФГБОУ ВПО «ЗабГУ»)

Факультет: Энергетический

Кафедра: Информатики, вычислительной техники и прикладной математики

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту

По дисциплине: Сети и телекоммуникации

На тему: «FTP сервер»

Выполнил студент группы ИВТ–18–1, Втулкин Владислав Викторович

Руководитель работы: Забелин Вячеслав Олегович

Чита

2021

КАЛЕНДАРНЫЙ ГРАФИК

выполнения курсового проекта

УТВЕРЖДАЮ

    Зав.кафедрой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Этапы выполнения курсовой работы | Месяцы и недели | | | | | | | | | | | | | | | |
| Февраль | | | Март | | | | Апрель | | | | Май | | | | |
| 1. Получение задания на курсовую работу | ++ | + |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 1. Анализ задачи |  | ++ | ++ |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 1. Анализ данных |  |  | + | ++ | ++ | + |  |  |  |  |  |  |  |  |  |  |
| 1. Программная реализация |  |  | + | + |  | ++ | ++ | ++ | ++ | ++ |  |  |  |  |  |  |
| 1. Тестирование |  |  |  | + |  |  |  |  |  |  | ++ | ++ |  |  |  |  |
| 1. Документирование |  |  |  |  |  |  |  |  |  |  |  |  | ++ | ++ | + |  |
| 1. Представление руководителю чернового варианта работы |  |  |  |  |  |  |  |  |  |  |  |  |  |  | ++ | + |
| 1. Корректировка работы в соответствии с замечаниями руководителя |  |  |  |  |  |  |  |  |  |  |  |  |  |  | ++ |  |
| 1. Защита работы |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | ++ |

План выполнен: руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись, расшифровка подписи)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г

# РЕФЕРАТ

В данной работе рассматривается и создаётся FTP сервер.

**СОДЕРЖАНИЕ**

[РЕФЕРАТ 5](#_Toc62826955)

[ВВЕДЕНИЕ 7](#_Toc62826956)

[1. Анализ задачи 8](#_Toc62826957)

[1.1 Описание предметной области...……………………………………8](#_Toc62826958)

[1.2 Постановка задачи……….. 8](#_Toc62826959)

[2. Теоретическая часть 9](#_Toc62826960)

3. [Программная реализация 14](#_Toc62826967)

[4. Документирование 16](#_Toc62826970)

[ЗАКЛЮЧЕНИЕ 17](#_Toc62826972)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 18](#_Toc62826973)

# ВВЕДЕНИЕ

Цель данной работы – продемонстрировать работу FTP сервера, создав клиент-серверное приложение. В приложении должно быть реализовано следующее:

* Возможность просмотра содержимого сервера
* Управление данными на FTP сервере
* Разделение пользователей по группам
* Права доступа для пользователей
* Логирование всех действий сервера
* Управление для администратора сервера куда входит: удаление/добавление пользователей в группы, создание групп, редактирование файлов, добавление/удаление файлов

# 1. Анализ задачи

## 1.1 Описание предметной области

Будет продемонстрирована работа протокола передачи файлов по сети FTP. Протокол построен на архитектуре «[клиент-сервер](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B8%D0%B5%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80)» и использует разные сетевые соединения для передачи команд и данных между клиентом и сервером. Пользователи FTP могут пройти аутентификацию, передавая логин и пароль [открытым текстом](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%B9_%D1%82%D0%B5%D0%BA%D1%81%D1%82), или же, если это разрешено на сервере, они могут подключиться анонимно.

## Постановка задачи

Задачей является написание клиент-серверного приложения для передачи файлов по сети. Передача файлов должна осуществляться с помощью FTP протокола.

Сервер должен принимать и обрабатывать несколько пользовательских соединений и отвечать на соответствующие запросы. Также должна быть возможность у администратора сервера с помощью программного интерфейса взаимодействовать с сервером и фалами на нем.

Клиентское приложение должно отображать информацию о файлах на сервере, обрабатывать и отправлять соответствующие FTP запросы.

# 2. Теоретическая часть

FTP — [протокол передачи](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB%D1%8B_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) [файлов](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB) по [сети](https://ru.wikipedia.org/wiki/%D0%92%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C), появившийся в 1971 году задолго до [HTTP](https://ru.wikipedia.org/wiki/HTTP) и даже до [TCP/IP](https://ru.wikipedia.org/wiki/TCP/IP), благодаря чему является одним из старейших прикладных протоколов. Изначально FTP работал поверх протокола [NCP](https://ru.wikipedia.org/wiki/Network_Control_Protocol)[[1]](https://ru.wikipedia.org/wiki/FTP#cite_note-1), на сегодняшний день широко используется для распространения [ПО](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) и доступа к удалённым [хостам](https://ru.wikipedia.org/wiki/%D0%A5%D0%BE%D1%81%D1%82).

Протокол построен на архитектуре «[клиент-сервер](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B8%D0%B5%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80)» и использует разные сетевые соединения для передачи команд и данных между клиентом и сервером. Пользователи FTP могут пройти аутентификацию, передавая логин и пароль [открытым текстом](https://ru.wikipedia.org/wiki/%D0%9E%D1%82%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%B9_%D1%82%D0%B5%D0%BA%D1%81%D1%82), или же, если это разрешено на сервере, они могут подключиться анонимно.

## Соединение и передача данных

Протокол определён в [RFC 959](https://tools.ietf.org/html/rfc959). Сервер отвечает по потоку управления [трёхзначными ASCII-кодами состояния](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_%D0%BA%D0%BE%D0%B4%D0%BE%D0%B2_%D0%BE%D1%82%D0%B2%D0%B5%D1%82%D0%BE%D0%B2_FTP) с необязательным текстовым сообщением. Например, «200» (или «200 ОК») означает, что последняя команда была успешно выполнена. Цифры представляют код ответа, а текст — разъяснение или запрос. Текущая передача по потоку данных может быть прервана с помощью прерывающего сообщения, посылаемого по потоку управления.

FTP может работать в активном или пассивном режиме, от выбора которого зависит способ установки соединения. В активном режиме клиент создаёт управляющее TCP-соединение с сервером и отправляет серверу свой IP-адрес и произвольный номер клиентского порта, после чего ждёт, пока сервер запустит TCP-соединение с этим адресом и номером порта. В случае, если клиент находится за [брандмауэром](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D1%81%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D1%8D%D0%BA%D1%80%D0%B0%D0%BD) и не может принять входящее TCP-соединение, может быть использован пассивный режим. В этом режиме клиент использует поток управления, чтобы послать серверу команду PASV, и затем получает от сервера его IP-адрес и номер порта, которые затем используются клиентом для открытия потока данных с произвольного клиентского порта к полученному адресу и порту. Оба режима были обновлены в сентябре 1998 г. для поддержки IPv6. В это время были проведены дальнейшие изменения пассивного режима, обновившие его до расширенного пассивного режима.

При передаче данных по сети могут быть использованы четыре представления данных:

* [ASCII](https://ru.wikipedia.org/wiki/ASCII) — используется для текста. Данные, если необходимо, до передачи конвертируются из символьного представления на хосте-отправителе в «восьмибитный ASCII», и (опять же, если необходимо) в символьное представление принимающего хоста. В частности, изменяются символы перевода строки (CR /chr(13)/, LF /chr(10)/ в Windows на LF /chr(10)/ в Unix/Linux. Как следствие, этот режим не подходит для файлов, содержащих не только обычный текст.
* Режим изображения (обычно именуемый бинарным) — устройство-отправитель посылает каждый файл [байт](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B9%D1%82) за байтом, а получатель сохраняет поток байтов при получении. Поддержка данного режима была рекомендована для всех реализаций FTP.
* [EBCDIC](https://ru.wikipedia.org/wiki/EBCDIC) — используется для передачи обычного текста между хостами в кодировке EBCDIC. В остальном этот режим аналогичен ASCII-режиму.
* Локальный режим — позволяет двум компьютерам с идентичными установками посылать данные в собственном формате без конвертации в ASCII.

Для текстовых файлов предоставлены различные форматы управления и настройки структуры записи. Эти особенности были разработаны для работы с файлами, содержащими [Telnet](https://ru.wikipedia.org/wiki/Telnet" \o "Telnet) или ASA-форматирование.

Передача данных может осуществляться в любом из трёх режимов:

## Аутентификация

FTP-аутентификация использует схему имя пользователя/пароль для предоставления доступа. Имя пользователя посылается серверу командой USER, а пароль — командой PASS. Если предоставленная клиентом информация принята сервером, то сервер отправит клиенту приглашение и начинается сессия. Пользователи могут, если сервер поддерживает эту особенность, войти в систему [без предоставления учётных данных](https://ru.wikipedia.org/wiki/FTP#%D0%90%D0%BD%D0%BE%D0%BD%D0%B8%D0%BC%D0%BD%D1%8B%D0%B9_FTP), но сервер может предоставить только ограниченный доступ для таких сессий.

Хост, обеспечивающий FTP-сервис, может предоставить [анонимный](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%BE%D0%BD%D0%B8%D0%BC%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%B2_%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82%D0%B5) доступ к FTP. Пользователи обычно входят в систему как «anonymous» в качестве имени пользователя. Хотя обычно пользователей просят прислать адрес их [электронной почты](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BE%D1%87%D1%82%D0%B0) вместо пароля, никакой проверки фактически не производится. Многие FTP-хосты, предоставляющие обновления программного обеспечения, поддерживают анонимный доступ.

* Поточный режим — данные посылаются в виде непрерывного потока, освобождая FTP от выполнения какой бы то ни было обработки. Вместо этого вся обработка выполняется [TCP](https://ru.wikipedia.org/wiki/TCP). Индикатор конца файла не нужен, за исключением разделения данных на [записи](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)).
* Блочный режим — FTP разбивает данные на несколько блоков (блок заголовка, количество байт, поле данных) и затем передаёт их TCP.
* Режим сжатия — данные сжимаются единым алгоритмом (обычно кодированием длин серий).

## Основные команды

* ABOR — Прервать передачу файла
* CDUP — Сменить каталог на вышестоящий.
* CWD — Сменить каталог.
* DELE — Удалить файл (DELE filename).
* EPSV — Войти в расширенный пассивный режим. Применяется вместо PASV.
* HELP — Выводит список команд, принимаемых сервером.
* LIST — Возвращает список файлов каталога. Список передаётся через соединение данных.
* MDTM — Возвращает время модификации файла.
* MKD — Создать каталог.
* NLST — Возвращает список файлов каталога в более кратком формате, чем LIST. Список передаётся через соединение данных.
* NOOP — Пустая операция.
* PASS — Пароль.
* PASV — Войти в пассивный режим. Сервер вернёт адрес и порт, к которому нужно подключиться, чтобы забрать данные. Передача начнётся при введении следующих команд: RETR, LIST и т. д.
* PORT — Войти в активный режим. Например PORT 12,34,45,56,78,89. В отличие от пассивного режима для передачи данных сервер сам подключается к клиенту.
* PWD — Возвращает текущий каталог.
* QUIT — Отключиться.
* REIN — Реинициализировать подключение.
* RETR — Скачать файл. Перед RETR должна быть команда PASV или PORT.
* RMD — Удалить каталог.
* RNFR и RNTO — Переименовать файл. RNFR — что переименовывать, RNTO — во что.
* SIZE — Возвращает размер файла.
* STOR — Закачать файл. Перед STOR должна быть команда PASV или PORT.
* SYST — Возвращает тип системы (UNIX, WIN, …).
* TYPE — Установить тип передачи файла (бинарный, текстовый).
* USER — Имя пользователя для входа на сервер.

3. Программная реализация

## Структура приложения

### Структура сервера

Серверная часть состоит из следующих файлов:

1. ftpServer,py. Содержит класс FTP протокола, который вызывается в отдельном потоке для каждого нового пользователя для обработки сессии. А также этот модуль – основная точка входа для запуска сервера.
2. adminPanel.py. Основная точка входа для панели администратора.
3. database.py. Содержит класс DatabaseFTP для обработки групп пользователей и разрешения для файлов.
4. init\_db.py. Родительский класс для DatabaseFTP, который инициализирует базу данных.
5. log\_func.py. Содержит функции для записи информации в лог-файлик.

### Структура клиента

Клиентская часть состоит из следующих файлов:

* 1. Design.py. Содержит класс отвечающий за GUI для приложения.
  2. loginDesign.py. Содержит класс модального окна для подключения к серверам.
  3. ftpClient.py. Содержит класс клиентского приложения и является основной точкой входа в программу для пользователя.

4. Документирование

Проект имеет три точки входа для разных целей пользователей:

1. ftpServer.py для запуска сервера.
2. adminPanel.py для запуска терминала администратора.
3. ftpClient.py как точка входа для пользователя.

При запуске сервера в терминале будет показан ip адрес и порт, на котором запустился сервер. В дальнейшем записи о подключении и действиях пользователей будут записываться не только в файл логов но и отображаться в в этом терминале. Для того, чтобы отключить сервер, нужно ввести в терминал «q», после чего работа сервера прекратиться. Каких-либо других действий с сервером производить из данного модуля не получится: для управления файлами и группами пользователей есть модуль adminPanel.py.

Рассмотри модуль adminPanel.py. При запуске данного модуля в терминале отобразятся логи запуска соединения с базой данных, в которой хранятся пользователи и группы. Далее терминал будет ожидать ввода команд, а для получения информации о доступных командах предложит ввести «help».

Терминал администраторской панели поддерживает следующие команды:

* list : [dirpath] отображает информацию о файле или каталоге и запоминает (если каталог) его как текущий
* help : отображает команды терминала
* mode : [file, group, mode] изменяет права доступа для указанной группы (group) и указанного файла (file), права нужно указывать в виде «rwx», «rw-», «r-x» и т.д.
* groups : показывает список всех групп пользователей
* users : показывает список всех пользователей, которые зашли на сервер зарегестрированными
* files : показывает список всех файлов
* usergroup : [username , group] задает группу пользователю, если такой группы нет – она будет создана
* quit : закрывает терминал
* q : закрывает терминал

Клиентская часть имеет пользовательский графический интерфейс. При запуске ftpClient.py отобразится пустое окошко для просмотра файлов на сервере. Для подключения к серверу предусмотрена единственная кнопка справа сверху основного окна. При нажатии на нее откроется модальное окно для ввода адреса сервера, к которому нужно подключится, а после нажатия на кнопку «ОК», еще одно модальное окно для ввода данных пользователя: логин, пароль. Если адрес верен – модальное окно закроется и на основном окне для просмотра файлов отобразятся файлы на сервере. Для перехода по каталогам – двойной клик левой кнопкой мыши по иконке каталога, для взаимодействия с файлом двойной клик по иконке файла.

# ЗАКЛЮЧЕНИЕ

Было разработано клиент-серверное приложение для передачи файлов по сети интернет с помощью протокола FTP. Для разработки данного программного обеспечения были применены следующий стек технологий: python, PYQT5, QT DESIGNER, socket, ftplib, sqlite3.

Был разработан пользовательский интерфейс для клиентской стороны.

Для хранения информации о пользователях и группах была разработана база данных на основе sqlite3.

На стороне сервера используются потоки для обработки нескольких пользователей сразу.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Работа с FTP: python-scripts [Электронный ресурс] – URL  [https://python-scripts.com/ftplib/](https://habr.com/ru/post/334778/)
2. FTP. [Электронный ресурс] – URL <https://ru.wikipedia.org/wiki/FTP#%D0%A1%D0%BE%D0%B5%D0%B4%D0%B8%D0%BD%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B8_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85>
3. Internet standart FTP: [Электронный ресурс] – https://datatracker.ietf.org/doc/html/rfc959