**Отчет по лабораторной работе № 1**

Тарасов А. Т.

Группа ИУ9-31Б

**Цель:** Реализовать совместную работу Go ftp-сервера и Go ftp-клиента

Заявленной цели соответствуют следующие **задачи:**

**Реализовать ftp-клиент на языке GO, поддерживающий следующие функции:**

* загрузка файла go ftp-клиентом на ftp-сервер;
* скачивание файла go ftp-клиентом с ftp-сервера;
* создание директории go ftp-клиентом на ftp-сервере;
* удаление go ftp-клиентом файла на ftp-сервере;
* получение содержимого директории на ftp-сервере с помощью go ftp-клиента.

**Реализовать ftp-сервер на языке GO обладающий следующими функциями:**

* авторизация клиента на ftp-сервере;
* передача клиенту список содержимого заданной директории ftp-сервера по запросу;
* возможность скачивать файлы из заданной директории ftp-сервера по запросу;
* возможность загружать файлы в заданную директорию ftp-сервера по запросу;
* возможность создавать директории на ftp-сервере по запросу;
* возможность удалять директории на ftp-сервере по запросу.

**Исходный код ftp-клиента:**

package main

import (

    "flag"

    "fmt"

    "io/ioutil"

    "os"

    "github.com/jlaffaye/ftp"

    "bytes"

    "log"

)

var connection \*ftp.ServerConn

var ftpURL = "students.yss.su:21"

var timeOut = 5

var login = "ftpiu8"

var password = "3Ru7yOTA"

var ftpPath = "/tat"

var localPath = "/downloads"

// Создает папку на удаленном сервере

func MakeDir(path string) {

    connection.MakeDir(path)

}

// Загрузка файла на сервер. Принимает локальный путь,

// и путь на ftp сервере

func UploadFile(srcPath string, destPath string) {

    data, err := ioutil.ReadFile(srcPath)

    reader := bytes.NewBuffer(data)

    err = connection.Stor(ftpPath+destPath, reader)

    if err != nil {

        panic(err)

    }

}

// Запись фала по переданному пути из \*[]byte

func WriteFile(path string, data \*[]byte) {

    file, \_ := os.Create(path)

    \_, err := file.Write(\*data)

    if err != nil {

        panic(err)

    }

}

// Загрузка файла с ftp сервера

func DownloadFile(srcPath, destPath string) {

    r, err := connection.Retr(srcPath)

    if err != nil {

        panic(err)

    }

    defer r.Close()

    buf, err := ioutil.ReadAll(r)

    if err != nil {

        panic(err)

    }

    WriteFile(destPath, &buf)

}

// Удаление файла с удаленного сервера

func RemoveFile(path string) {

    connection.Delete(path)

}

// Инициализация

func Init() {

    connection.MakeDir(ftpPath)

    connection.ChangeDir(ftpPath)

}

// Вывод содержимого директории

func RetriveDir(path string) {

    walker := connection.Walk(path)

    for walker.Next() {

        fmt.Println(walker.Path())

    }

}

// Точка входа в программу

func main() {

    var err error

    ftpURLF := flag.String("url", "students.yss.su:21", "Server address")

    loginF := flag.String("user", "ftpiu8", "Login")

    passwordF := flag.String("pass", "3Ru7yOTA", "Password for login")

    flag.Parse()

    ftpURL = \*ftpURLF

    login = \*loginF

    password = \*passwordF

    connection, err = ftp.Connect(ftpURL)

    if err != nil {

        log.Fatal(err)

    }

    err = connection.Login(login, password)

    if err != nil {

        log.Fatal(err)

    }

    Init()

    UploadFile("README.txt", "/README.txt")

    DownloadFile("/tat/README.txt", "downloads/README.txt")

    MakeDir("AnotherOne")

    UploadFile("README.txt", "/AnotherOne/README.txt")

    RetriveDir(ftpPath)

    if err := connection.Quit(); err != nil {

        log.Fatal(err)

    }

}

**Исходный код ftp-сервера:**

package main

import (

    "flag"

    "log"

    filedriver "github.com/goftp/file-driver"

    "github.com/goftp/server"

)

// Точка входа

func main() {

    // флаги для конфигурирования при запуске из терминала

    var (

        root = flag.String("root", "./files", "Root directory to serve")

        user = flag.String("user", "admin", "Username for login")

        pass = flag.String("pass", "123456", "Password for login")

        port = flag.Int("port", 3030, "Port")

        host = flag.String("host", "localhost", "Host")

    )

    flag.Parse()

    factory := &filedriver.FileDriverFactory{

        RootPath: \*root,

        Perm:     server.NewSimplePerm("user", "group"),

    }

    //Параметры ftp сервера

    opts := &server.ServerOpts{

        Factory:  factory,

        Port:     \*port,

        Hostname: \*host,

        Auth:     &server.SimpleAuth{Name: \*user, Password: \*pass},

    }

    log.Printf("Start ftp server on %v:%v", opts.Hostname, opts.Port)

    log.Printf("Server root dir %s", \*root)

    log.Printf("Username %v, Password %v", \*user, \*pass)

    server := server.NewServer(opts)

    err := server.ListenAndServe()

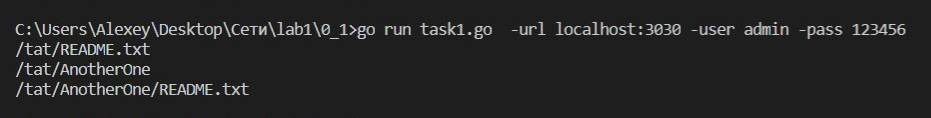
    if err != nil {

        log.Fatal("Error starting server:", err)

    }

}

**Пример запуска ftp-клиента:**



**Пример запуска ftp-сервера:**

