Министерство науки и высшего образования РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Тульский государственный университет»

Институт прикладной математики и компьютерных наук

Кафедра информационной безопасности

**БЕЗОПАСНОСТЬ СЕТЕЙ ЭЛЕКТРОННЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН**

Отчет по выполнению лабораторной работы №2

Выполнил

ст. гр. 230781 А. Р. Ивлев

Проверил

Асс. Греков Михаил Михайлович

Тула 2022

**Лабораторная работа №2**

**Взаимодействие прикладных программ с помощью протоколов электронной почты и протоколов прикладного уровня**

**Цель работы**

Изучение принципов работы протоколов электронной почты, разработка прикладных программ, осуществляющих взаимодействие и передачу данных друг другу на основе прикладных протоколов.

**Задание на работу**

Вариант №1

Разработать программу, позволяющую принимать выбранный файл с удаленной ЭВМ по протоколу FTP. Адрес ЭВМ, имя и пароль пользователя, имя и местоположения файла должны настраиваться при использовании программы. Порт обмена данными можно считать фиксированным.

**Ход работы**

На виртуальной машине был создан FTP-сервер.

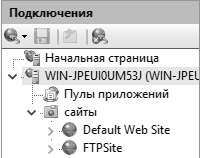


Рисунок 1. Созданный FTP-сайт

На языке Python был написан класс, который позволяет работать с удаленной ЭВМ по протоколу FTP.

*Листинг 1. Класс FTP*

import socket

class FTP:

\_\_ALL = 10240

\_\_socket = None

\_\_IP = None

\_\_user = None

\_\_password = None

\_\_port = None

\_\_data\_socket = None

def \_\_init\_\_(self):

self.\_\_socket = socket.socket()

# Загружает данные из потока

def \_\_load\_data(self, size: int) -> bytes:

self.\_\_data\_socket = socket.socket()

self.\_\_data\_socket.connect((self.\_\_IP, self.\_\_port))

data = self.\_\_data\_socket.recv(size)

self.\_\_data\_socket.close()

return data

# Отправляет данные в поток

def \_\_send\_data(self, data: bytes) -> bytes:

self.\_\_data\_socket = socket.socket()

self.\_\_data\_socket.connect((self.\_\_IP, self.\_\_port))

self.\_\_data\_socket.send(data)

self.\_\_data\_socket.close()

return data

# Открывает поток для чтения или записи

def \_\_open\_data\_connection(self):

self.\_\_socket.send(b"PASV\r\n")

ret = ['']

while ret == ['']:

ret = self.\_\_socket.recv(self.\_\_ALL)[27:-4].decode("UTF-8").split(',')

self.\_\_port = (int(ret[4]) << 8) + int(ret[5])

# Подключает к серверу

def connect(self, ip: str, user: str, password: str) -> bool:

try:

self.\_\_IP = ip

self.\_\_user = user

self.\_\_password = password

self.\_\_socket.connect((self.\_\_IP, 21))

self.\_\_socket.recv(self.\_\_ALL)

self.\_\_socket.send(bytes("USER " + self.\_\_user + "\r\n", encoding="UTF-8"))

self.\_\_socket.recv(self.\_\_ALL)

self.\_\_socket.send(bytes("PASS " + self.\_\_password + "\r\n", encoding="UTF-8"))

ret = self.\_\_socket.recv(self.\_\_ALL).split(b" ")[0]

if ret == b"230":

return True

return False

except:

return False

# Выводит список всех файлов и папок в директории

def list\_dir(self):

self.\_\_open\_data\_connection()

self.\_\_socket.send(bytes("LIST\r\n", encoding="UTF-8"))

data = self.\_\_load\_data(self.\_\_ALL).decode('utf-8')

print(data)

self.\_\_socket.recv(self.\_\_ALL)

# Делает указанную папку текущей директорией

def change\_dir(self, dir: str):

if dir == '..':

self.\_\_socket.send(bytes("CDUP\r\n", encoding="UTF-8"))

else:

self.\_\_socket.send(bytes("CWD " + dir + "\r\n", encoding="UTF-8"))

self.\_\_socket.recv(self.\_\_ALL)

# Загружает файл с сервера

def load\_file(self, filename: str):

self.\_\_open\_data\_connection()

self.\_\_socket.send(bytes("SIZE " + filename + "\r\n", encoding="UTF-8"))

size = (int(self.\_\_socket.recv(self.\_\_ALL).decode("UTF-8").split(' ')[1]))

self.\_\_socket.send(bytes("RETR " + filename + "\r\n", encoding="UTF-8"))

data = self.\_\_load\_data(size)

self.\_\_socket.recv(self.\_\_ALL)

self.\_\_socket.recv(self.\_\_ALL)

with open(filename, 'wb') as file:

file.write(data)

# Отправляет файл на сервер

def send\_file(self, filename: str):

self.\_\_open\_data\_connection()

self.\_\_socket.send(bytes("STOR " + filename + "\r\n", encoding="UTF-8"))

data = b""

with open(filename, "rb") as file:

data = file.read()

self.\_\_send\_data(data)

self.\_\_socket.recv(self.\_\_ALL)

self.\_\_socket.recv(self.\_\_ALL)

# Возвращает полное имя текущей директории

def get\_dir(self) -> str:

self.\_\_open\_data\_connection()

self.\_\_socket.send(bytes("PWD\r\n", encoding="UTF-8"))

data = self.\_\_socket.recv(self.\_\_ALL)[5:-25]

return data.decode(encoding="UTF-8")

Была написана программа, позволяющая пользователю использовать методы класса.

*Листинг 2. Основная программа*

from FTP import FTP

ftp\_server = FTP()

while True:

IP = input('IP:\t\t\t')

user = input('Username:\t')

password = input('Password:\t')

if ftp\_server.connect(IP, user, password):

print("Connected\n")

break

print("IP, username, or password is not valid\n")

help = """

ls Lists the current folder's content

cd {folder} Makes the specified folder the current one

load {file} Loads the specified file from the server

send {file} Sends the specified file to the server

quit Closes the program

"""

print("Type 'help' for detailed information on available options\n")

while True:

command = input('~' + ftp\_server.get\_dir() + ': ').split()

try:

if command[0] == "ls":

ftp\_server.list\_dir()

elif command[0] == "cd":

ftp\_server.change\_dir(command[1])

elif command[0] == "load":

ftp\_server.load\_file(command[1])

print("Done\n")

elif command[0] == "send":

ftp\_server.send\_file(command[1])

print("Done\n")

elif command[0] == "quit":

break

elif command[0] == "help":

print(help)

else:

print("Unknown command:\t" + command[0])

except:

print("Not enough parameters\n")

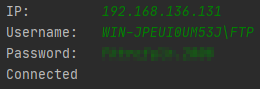


Рисунок 2. Подключение к серверу

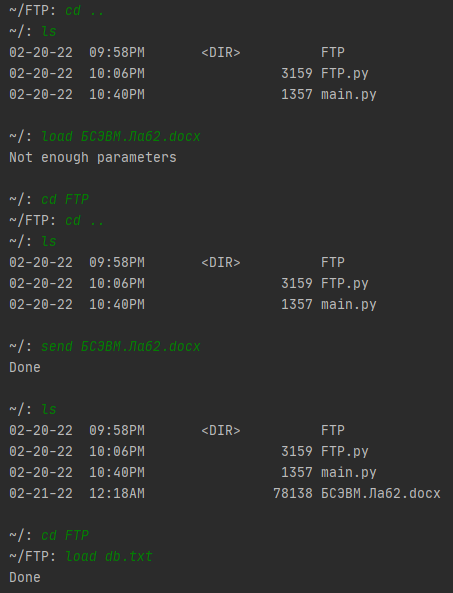


Рисунок 3. Загрузка и отправка файлов

**Вывод**

FTP-сервер — это сервер, работающий по протоколу File Transfer Protocol и предназначенный для обмена файлами через Интернет или локальную компьютерную сеть. Встроенные инструменты FTP-сервера позволяют гибко настроить разрешения доступа к серверу и шифрование передаваемых данных, что обеспечивает хороший уровень безопасности.

Была разработана программа, позволяющая принимать выбранный файл с удаленной ЭВМ или отправлять файл на удаленную ЭВМ по протоколу FTP.