T-409-TSAM-2015: Computer Networks Programming Assignment 1 – Trivial File Transfer Protocol

Lecturer: Marcel Kyas

August 24, 2016

Submission Deadline

You must submit the solutions to this assignment on myschool until 23:59:59 GMT on September 14, 2016.

Intended Learning Outcomes

You should be able to:

- Use the BSD socket API
- Understand the trivial file transfer protocol
- Write a server application

1 Instructions

Please implement a TFTP server, a program that serves files using the TFTP protocol (see below).

Your hand-in must conform to the following to be graded:

- All necessary files need to be stored in an archive, especially the content of the `src/` directory. If unpacked, we expect to see a `./README` file and a subdirectory `./src/` containing the source code and a `Makefile`; do not have those files in a subdirectory.
- You must *not* include the data files in `data/`.
- You must include a file called `./AUTHORS` that includes the name of each group member followed by the e-mail address *at ru.is* enclosed in angle brackets (see example) on separate lines.
- The `./README` file should contain notes about the implementation.
- The default rule in `Makefile` shall compile the tftpd program.

- Do not forget to comment your code and use proper indentation.

We will test your project on `skel.ru.is`. It is a good idea to test your project on this machine.

2 Problems

Your task for this assignment is to program an TFTP (Trivial File Transfer Protocol) server in the C programming language.

The TFTP protocol is defined in https://www.ietf.org/rfc/rfc1350.txt, which your server must conform to. TFTP is a protocol used to transfer files. It is considerably easier to implement than FTP. This makes it possible to implement it for example in the BIOS which allows to to boot diskless machines by downloading the boot loader from some TFTP server. You will need to read RFC 1350 to understand the protocol. The server should allow to download files, but must not allow uploading.

Implement the server in C and put it in a file named `src/tftpd.c`. Write a Makefile to compile the server. Make sure it compiles without warnings if using `gcc` using the flags `-O2 -Wall -Wextra -Wformat=2`. The server needs to accept two command line argument — the port to listen on and the directory containing the files to serve.

It needs to be possible to run the server using this command:

```
[student14@skel pa1]$ make -C ./src
[student14@skel pa1]$ ./src/tftpd 2000 data
```

Instead of port 2000, call `/labs/tsam15/my_port` to obtain a port that you can use, if you run it on skel.

The output of the server should list which file is requested from which IP and port:

```
file "example_data1" requested from 127.0.0.1:37242
```

For security reasons, the server should only send files that are inside the directory given as command line arguments. The names of all requested files should be treated as being under the named directory (even if the filename contains a path). Downloading files outside of this directory must not be allowed.

A TFTP client is installed on skel and can be used for testing, e.g.,

```
[student14@skel pa1]$ tftp 127.0.0.1 2000 -c get example_data1
[student14@skel pa1]$ ls
Makefile answers data example_data1
```

should download the file `example_data1` from the TFTP server into the current working directory. The downloaded copy must be identical to the original on the server. Compare the files using “diff”! Use “man tftp” to learn about different commands of the client. Try them and see if your server is handling the requests correctly (also try uploading).