TECHNISCHE UNIVERSITÄT BERLIN
Fakultät IV – Elektrotechnik und Informatik
Fachgebiet Intelligente Netze und Management verteilter Systeme
Prof. Anja Feldmann, Ph. D.
Harald Schiöberg, Bernhard Ager, Vlad Manilici

2nd Assignment Protocol Design WS 08/09

Question 1: (60 points) Simple client and server

Develop a client and a server for file transfer. As a programming language you may choose any language **except Java**. When using Perl you are not allowed to make use of the IO::Socket module to maximize the training effect. Client and server should be in a single program, and a command line parameter should be used to switch the role.

Write an separate function to implement each client and server functionality!

(a) The client should implement the following features:

- You should be able to control it via the command line: `transfer <host> <port> <file>`.
- It should connect to the named host and fetch the file with `get <file>`. All lines should be ended with `\n`.
- The server will answer with `200 <count> Bytes`, a subsequent empty line and the file content in case the file exists, or otherwise with a `5<xx> <error-message>`.
- The client should write the contents of the file to the standard output if no error was encountered, or write an appropriate error message to the error output. Subsequently it should terminate with an appropriate exit code.

The client functionality should be put into an separate function that expects destination IP address and destination port as a parameter.

(b) The server should implement the following features:

- You should be able to control it with the following command line: `transfer -listen <port>`.
- It should wait for connections on the named port.
- If a client sends a request as described above, and a suitable file is found in the current directory, it should answer accordingly.
- If the file name contains a `/` it should answer with `551 Current directory only`.
- If it does not find a suitable file it should answer with `550 No such file or directory`.

Items to submit: The program as well-documented source code and, if applicable, a compiled version.

Question 2: (40 points) Simple proxy for the file transfer programm

Now the file transfer program should be *extended* so that it can act as a proxy to which a client can connect and over which the file transfer can be carried out.

We introduce the following modifications:

- The client can be started with `transfer <proxy> <port> <server> <port> <file>`, and behaves like the client from exercise 1, except it connects to the proxy instead of the server and the query string is changed to `get <hostname> <port> <extra string>`.
- If a proxy receives a command of that form it opens a client connection to `hostname` and requests `get <extra string>`. The server's answer is relayed to the client through the proxy.
- In case the proxy does not recognize a server with the given name it answers with `553 <hostname1>: <hostname> not known`, where `<hostname1>` is his own host name.

The proxy functionality should be implemented as an extension to the normal server functionality and make use of the normal client functionality. Therefore the server function must be able to handle more parameters (at least 2, for proxy IP address and port).

Items to submit: The program as well-documented source code and, if applicable, a compiled version.

Details on how to submit your assignments:
see FAQ: `http://www.net.t-labs.tu-berlin.de/teaching/ws0809/PD_labcourse/faq.shtml`
Due date: Wednesday, 5.11.2008 at 23:59 s.t.