Operating Systems with C/C++
Semester 1, 2012
Eike Ritter

The University of Birmingham
19 October, 2012

Assessed Exercise 2

Deadline: Monday 29 October, 12noon

The Task

Write a system for logging entries to a file. This system should be a client-server application. The server should listen for incoming connections and accept an arbitrary number of log-entries from any client. It should check whether these entries are well-formed, and if they are, write them to a file.

An entry is well-formed if it can be generated by the following grammar:
$\langle$`entry`$\rangle$ $\rightarrow$ $\langle$`alphanum`$\rangle^+$ ':' $\langle$`textchar`$\rangle^+$
where $\langle$`alphanum`$\rangle$ is any letter or digit and $\langle$`textchar`$\rangle$ is any character with a character code between 32 and 126 inclusive. An example of a valid line would be

```
cupsd: kippenberger out of paper
```

An example of an invalid line would be

```
program; program : Invalid input
```

The server should be multi-threaded and process several connections simultaneously. You should use the `pthread`-library to implement threading. Moreover, you should ensure that each line is written either completely to the file or not at all, without any interference with other lines.

The server program should accept two arguments, namely the filename and the port the server listens on. TCP-sockets should be used for communication between client and server. If the file is not writable the server should immediately terminate with an appropriate error message. The server should append entries to an existing file.

The client program should accept two arguments, namely the hostname and the port number. It should read lines from the standard input and send each line separately to the server. The client program terminates if an EOF is read. The combination `Control-d` generates an EOF-character. The connection between client and server should be properly closed in this case.

For three bonus marks you should add a Makefile for this system. These marks are on top of a potential full mark.

Marking Scheme

Please use the School submission system for submitting your code. Please submit only the source files you have written yourself. We will compile and run your code on the Linux machines and mark it accordingly. Please in particular note that we will use the compiler option introduced in the lecture and will deduct 6 marks immediately if there is any compiler error or warning.

We will award marks as follows:

- 5 marks for handling the mulithreading correctly
- 5 marks for handling the synchronisation correctly
- 5 marks for checking format of lines correctly
- 5 marks for correct interaction between client and server
- 3 marks bonus point for adding a correct Makefile