Operating Systems
Semester 1, 2015
Eike Ritter

The University of Birmingham
2 October, 2015

Assessed Exercise 1

Deadline: Friday 16 October, 4pm

The Task

Write a simple encryption server. We assume that all encrypted files have the ending `.gpg`. This encryption server should accept the following commands:

- Encrypt a file on the server with a passphrase provided by the client. If the file has already been encrypted, it will not be encrypted, and an error message will be raised.
- Decrypt a file on the server with a passphrase provided by the client. If the file is not encrypted, an error message will be raised.

The server should be multi-threaded and process several connections simultaneously. You should use the `pthread`-library to implement threading.

The server program should accept one arguments, namely the port the server listens on. TCP-sockets should be used for communication between client and server. You should ensure that only one client can encrypt or decrypt a particular file at a given time. If another client attempts to encrypt or decrypt the same file, it should get an error message.

The client program should accept four arguments. The first argument is either `--encrypt`, in which case encryption will be performed, or `--decrypt`, in which case decryption will be performed. The second, third and fourth argument are the hostname, the port number and the file to be encrypted/decrypted respectively.

You should also provide Makefiles for this system.

You should use gpg for encryption and decryption. The command for encrypting a file is

```
echo <passphrase> | gpg -c --passphrase-fd 0 <fileName>
```

and the command for decrypting a file is

```
echo <passphrase> | gpg -d --passphrase-fd 0 --output <fileName> <fileName>.gpg
```

where in both case `<filename>` is the name of the unencrypted file. You may use the library function `system` to execute this command.

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 multithreading correctly
- 7 marks for implementing the commands correctly
- 5 marks for correct interaction between client and server
- 3 marks for correct Makefiles