Operating Systems HW #1

Due 24.11.2016, 23:55

Part 1

In this part of the assignment, you will learn and experiment with Linux commands by creating a simple script.

- Read about the following commands (through man pages & google): *ls, rm, cd, mv, cp, cat, echo, mkdir, rmdir*.
- To write the output of a command into a file, use >
  e.g., *echo she sells seashells by the seashore > shells*
- Create a file named *script.sh*
  - Give it permissions to execute using the following command: *chmod 755 script.sh*
  - In this file, write shell commands to create a basic script
  - You can execute the script from the terminal by writing: ***./****script.sh*
- The script should perform the following:
  - Create a directory with your ID as its name (**9 digits**)
  - Set it as the current directory
  - Create another directory inside it called “temp”
  - In "temp", create 3 files – one named as your first name, one as your last name, and one as your TAU username. Each file's contents should be the same as its filename.
  - **Copy** the first 2 files (first & last name) to the original directory (with your ID) while reversing their names (first name contents is last name, and vice versa), then delete the original **2** files in "temp".
  - **Move** the 3rd file to the original directory, leave the temp directory, and delete the temp directory.
  - List the contents of the directory, as well as the contents of each of the 3 files. Precede each output with a descriptive text (of your choice). When you list, make sure to use long listing and list all files.

Guidelines

- You may assume that none of the directories and files exist beforehand.
- Use only the specified commands – no variables, functions, command-line arguments, etc.

Part 2

In this part of the assignment, you will build a command line utility for file encryption and decryption (XOR cipher).

Write a program that accepts three command-line arguments:

- Path to a directory with decrypted/encrypted files.
- Path to a file which contains the encryption key.
- Path to store resulting encrypted/decrypted file.

The input directory (the 1st argument) contains any number of files. The program will perform an encrypt/decrypt operation on each file. If it is a standard (decrypted) file, then the program will encrypt it, otherwise the program will decrypt it. Note that the operation is symmetric in both cases, thus the program doesn't care which file types it receives.

The output directory (3rd argument) is different from the input directory, and is where the output files should be placed. For each file in the input directory, a file with the same name should be created in the output directory, which contains the data resulting from the operation.

For each file, the operation will be performed by bitwise XOR-ing the contents of the input file and the key file. Please notice the following constraints:

- Input/output and key files can be of any size (1GB and much more). Your program should work in any case.
- The input folder is guaranteed to contain files only (no directories), no need to check.
- If an output file already exists, you should **overwrite** it.
- Input and key files do not have to be of the same size. In the case when the key file is larger than the input file, only part of the key file will be used. If the input file is longer than the key file, you should reuse bits of the key file by restarting reading from the beginning of the key file.
- There are no guarantees regarding the key file (it might not exist, be of any size, etc.). Handle all possible errors.
- If the input directory doesn't exist, exit with an error. If the output directory doesn't exist, create it.
- On any error (listed here or not), output a **proper** error message and exit.

Test your solution by encrypting/decrypting several different files (try different sizes) and make sure that you get the original file after performing the operation twice (i.e., encrypt then decrypt).

An encrypted **`cipher.bin`** file is attached to this homework. The encryption key is "`os17a`". The key file was generated by: *echo os17a > key.bin*
You may use it for testing your code. The result should be obvious if decrypted correctly (open it via the File Manager).

Guidelines

- Use only system calls (**learned in the recitations**) to access the files and folders.
- Don't forget to check the return values of **all** system calls.

Submit

- A single script file `script.sh`
- A single C file: `cipher.c`
- File names are **case-sensitive!**
- Read the submission guidelines.
- A sample zip file is attached. You should submit a zip file with **the same** file names and structure.