Computer Networks

Assignment - 2

Aim : To create an Application Level **File-Sharing-Protocol** with support for download and upload for files and indexed searching.

Features :

- The system should have **2 clients** (acting as servers simultaneously) listening to the communication channel for requests and waiting to share files (avoiding collisions) using an **application layer protocol** (like FTP/HTTP).
- Each client has the ability to do the following :
    - **Know the files present** on each others machines in the designated shared folders.
    - **Download files** from this shared folder.
    - **Upload files** to this shared folder.
    - Choose between **TCP and UDP** for transfer of files between the shared folders.
- The system should **periodically check** for any changes made to the shared folders.
- File transfer should incorporate **MD5-checksum** to handle file transfer errors.

Specifications : The system should incorporate the following commands :-

- ***IndexGet --flag (args)***
    - can request the display of the shared files on the connected system.
    - the history of requests made by either clients should be maintained at each of the clients respectively.
    - the *flag* variable can be **shortlist, longlist or regex.**
        - **shortlist :** flag would mean that the client only wants to know the names of files between a specific set of timestamps. The sample query is as below.
            - **$> IndexGet --shortlist <start-time-stamp> <end-time-stamp>**
            - **Output :** should include ‘name’, ‘size’, ‘timestamp’ and ‘type’ of the files between the start and end time stamps.
        - **longlist :** flag would mean that client wants to know the entire listing of the shared folder/directory including ‘name’, ‘size’, ‘timestamp’ and ‘type’ of the files.
            - **$> IndexGet --longlist**
            - **Output :** similar to above, but with complete file listing.

        - **regex :** flag would mean that client wants to know listing of the shared folder/directory which contain the specified regular expression pattern given as input argument in their filenames.
            - **$> IndexGet --regex <regex-argument>**
            - **Output :** similar to above, but with regular expression match.

- ***FileHash --flag (args) :***
    - this command indicates that the client wants to check if any of the files on the other end have been changed. The flag variable can take two values, **verify** and **checkall** :
        - **verify :** flag should check for the specific file name provided as command line argument and return its 'checksum' and 'last-modified' timestamp.
            - **$> FileHash --verify <filename>**
            - **Output :** checksum and last-modified timestamp of the input file.
        - **checkall :** flag should check perform what 'verify' does for all the files in the shared folder.
          (HINT : this command can be used for the periodic check of changes in the files of shared folders)
            - **$> FileHash --checkall**
            - **Output :** filename, checksum and last-modified timestamp of all the files in the shared directory.
- ***FileDownload --flag (args):***
    - as the name suggests, would be used to download files from the shared folder of connected user to our shared folder.
    - the flag variable can take the value **TCP** or **UDP** depending on the users request.
    - If a socket is not available, it should be created and both clients must use this socket for file transfer.
        - **$> FileDownload <filename>**
        - **Output :** should contain the filename, filesize, last-modified timestamp and the MD5-hash of the requested file.
    - HINT : the file-size parameter might be used for requesting the client side to allocate memory and use the allocated memory for downloading the file.
- **FileUpload --flag (args):**
    - as the name suggests, would be used to upload a file to the other clients shared folder.
    - The other client side may respond to this request with a **FileUploadDeny** or **FileUploadAllow** request which would cancel or proceed with the transfer, respectively.
    - The client should upload the file, its MD5-hash and last-modified timestamp if the response is FileUploadAllow and go back to listening to the communication channel if the response if FileUploadDeny.

- the flag variable can take the value **TCP** or **UDP** depending on the users request.
- If a socket is not available, it should be created and both clients must use this socket for file transfer.
  - **$> FileUpload <filename>**
  - **Output :** should contain the filename, filesize, last-modified timestamp and the MD5-hash of the uploaded file.

Instructions :

- The language permitted for this code is **C** (python/c++/java are not allowed).
- Put all your codes in a single folder named **'FileSharingProtocol'** and compress this to **.tar** format and upload it on the courses portal (Failing to follow the submission format would be penalised).
- The deadline for the submission of the assignment is **27th Feb'15 - 10:00 PM.** (Start your assignment early! It'll take a lot of time to complete).
- The code-folder should contain a makefile by the name '**run.sh'** which should compile the entire code. (Programs that do not compile will automatically be given 0 marks).
- All error scenarios must be gracefully handled (Programs crashing during testing will be penalised).
- Do attend the tutorial scheduled for **16th Feb'15**. No individual mails/pings to the TA's shall be entertained. Use the courses portal for general-doubts-discussion and clarification.
- You do not have to use threads for this project (You may use it for learning purposes, but no bonus marks shall be given for the same).
- Project can be done in groups of atmost 2 (however, the efficiency and usability of the application should be better in case of 2 member teams). Only one submission is required.
- **PLAGIARISM IN ANY FORM (YES! ANY FORM WHATSOEVER!) SHALL NOT BE TOLERATED AND A STRAIGHT 'F' GRADE FOR THE COURSE WILL BE GIVEN.**

"The richest people in the world look for and build networks,
everyone else looks for work."
