# Progetto di reti

Traccia 2

Daniel Capannini, matricola: 0000971194, mail: daniel.capannini@studio.unibo.it

la traccia 2 chiedeva di costruire un client-server UDP in cui dal client fosse possibile fare l'upload e download di file di qualsiasi dimensione sul server e di richiedere la lista dei file presenti in esso.

Struttura server:

il server è formato da un ciclo while nel quale rimane in attesa se richieste dal client, una volta ricevuta una richiesta controlla se è accettabile come comando e se lo è lo esegue. Tutte le operazioni all'interno del ciclo sono in un try il quale in caso di errore stampa il tipo di errore e poi permette che si torni al ciclo nella fase di attesa di una richiesta.

Struttura client:

anche il client presenta un ciclo while con un try esattamente come il server, all'interno del ciclo si parte tenendo traccia dei file che si hanno all'interno della directory del client tramite il comando file_name = os.listdir('./file_client/'), e si aspetta che l'utente fornisca un comando da terminale, una volta ricevuto si controlla se è valido e poi si gestisce, una volta che tutte le azioni rischieste dal comando si sono svolte il client torna in attesa di un comando da parte dell'utente.

I comandi che si possono usufruire sono:

LIST: questo comando richiesto al server, fornisce la lista di file che il server contiene e quindi possono essere scaricati. I file vengono salvati in file_name che viene aggiornato ad ogni passaggio dentro il while con il comando file_name = os.listdir('./file_server/'), la './file_server/' è la directory del server.

GET: questo comando seguito dal nome di un file permette di ricevere una copia del file se questo file è presente nella directory del server e non è presente nella directory del client, anche se il file è diverso o aggiornato ed è presente della directory del client non è possibile scaricarlo, i file devono avere obbligatoriamente nomi diversi. I file di dimensioni maggiori di BUFFER_SIZE=4096 vengono inviati suddivisi in più pacchetti e ricomposti dal client, grazie a questo metodo è possibile il download di file di qualsiasi dimensione. Quando si è trasmesso tutto il file viene inviato un messaggio di avviso di fine 'fine_fine' per avvisare il client della fine della trasmizione del file.

PUT: questo comando seguito dal nome di un file se questo è presente sul client e non sul server permette di farne l'upload, come per GET i file devono avere nomi diversi e non semplicemente contenuto diverso. Il funzionamento e l'implementazione è uguale a quella della GET con parti invertite. Il server prima che la transizione abbia inizio invia un messaggio di conferma 'GET'.

EXIT: comando aggiunto per permettere la chiusura del socket e dell'eseguibile del client, non tocca il server il quale rimane in attesa di comandi.