**Relazione sul progetto di Programmazione di Reti**

a.a. 2021/2022

Studente: Filippo Gurioli (0000971360)

Traccia n°: 2 (Server UDP protocol)

**Introduzione:**

In questa parte si vuole spiegare al meglio quali erano gli obiettivi da raggiungere e delineare la soluzione astenendosi però da dettagli implementativi.

La seconda traccia degli elaborati di Programmazione di Reti richiede la realizzazione di un software che permetta la creazione di una rete UDP che permetta lo scambio di qualsiasi tipo di file tra un server ed un client. Entrambi dovranno essere in grado di svolgere alcune funzioni, in particolare: get, put e list.

GET: funzione che permetta al client di scaricare un file presente nel server.

PUT: funzione che permetta al client di inviare un file presente in locale al server.

LIST: funzione che permetta al client di visualizzare tutti i file attualmente esistenti nel server.

Ragionando sulla consegna mi è venuto spontaneo pensare alla creazione di due entità principali, ovvero Client e Server. Il primo avrà il compito di interagire con l’utente diretto mentre l’altro avrà caratteristiche più semplici in quanto il Client avrà già fatto un perfezionamento dei dati in ingresso.

**Descrizione:**

In questa sezione si scenderà al dettaglio implementativo su come la consegna sia stata portata a termine.

Sia il client che il server saranno sempre attivi garantendo il servizio fin tanto che uno dei due non venga disconnesso forzatamente.

Client:

Si occupa di leggere l’input da tastiera dell’utente ed indipendentemente da ciò che scrive la prima cosa che fa è dividere la prima parola (che corrisponde al comando) dalla seconda (opzionale, che corrisponde al file). Procede interrogando il server in base al comando scritto, non prima però di aver scartato tutti i comandi che non rientrano tra quelli del vocabolario del server.

Una volta ricevuta la risposta la mostra all’utente e ricomincia il ciclo se si trattava di una richiesta di “list”, altrimenti invia al server il nome del file. Nel caso del “get” glielo invia così com’è, invece nel caso di “put”, valuta prima se il file esista veramente ed in caso negativo abortisce l’esecuzione della funzione tornando allo stadio iniziale. Nel caso in cui, invece, tutto sia corretto si procede con lo streaming di pacchetti alla fine del quale viene avvisato l’utente della corretta esecuzione dell’operazione.

Il criterio utilizzato per capire quando fosse necessario fermarsi è riconoscere nella stringa letta la stringa vuota (“”).

Alla fine dello streaming nel caso di “get” viene salvato il file passato nella directory corrente, mentre nel caso di “put” viene semplicemente chiuso lo streaming.

Server:

Questo componente, come già scritto, ha caratteristiche più semplici in quanto la parte di perfezionamento dei dati è interamente compiuta dal Client.

Il server è in perenne attesa di un comando da eseguire che può rientrare solo tra i già nominati get, put e list.

Nel caso di “list”, tramite un ciclo for, passa tutti i nomi dei file presenti nella sua directory al client in un’unica stringa in cui ogni nome è separato da un new line.

Nel caso di “get” si verifica innanzi tutto che il nome del file passato sia corretto e nel caso ciò non fosse vero, si avvisa il client e si ritorna in attesa del prossimo comando, se invece dovesse esserci effettivamente il file richiesto si inizia lo streaming dei dati fino a quando non si incontri la stringa vuota che fa da terminatore del file. In questo momento si stampa dalla parte server un testo che scandisca il termine dell’operazione e il ritorno allo stato di ascolto.

In fine, caso “put”, una volta ricevuto il nome del file, si apre la ricezione di dati che cesserà anch’essa con una stringa vuota e con l’avviso della conclusione dell’operazione.