**Relazione di Progetto**

* **Sviluppo di un sistema per la gestione remota -**

Corso di: Programmazione di Sistema

Prof. Marco Cianfriglia

Studente: Josef E. Zerpa Ruiz

**Introduzione:**

Per consolidare le conoscenze acquisite durante il corso di Programmazione di Sistema, si vuole sviluppare un sistema Client-Server, per la gestione remota di un calcolatore.

Per applicare gli studi sulla programmazione di sistema Unix e sistema Windows, i programmi sviluppati saranno eseguibili sia su un tipo sistema, che sull’altro, e anche in soluzioni miste; es. server Linux e client Windows. I programmi saranno quindi ‘Cross-Platform’.

In breve, il Client e il Server comunicheranno attraverso una connessione TCP, implementeranno un semplice protocollo di autenticazione, dopo un’autenticazione con successo il Server riceverà comandi dal Client, questi verranno svolti, restituendo al Client l’output dell’esecuzione, o in caso di errore, un codice identificativo. Inoltre il Server svolgerà del logging in locale, registrando ogni richiesta dopo un’autenticazione con successo. Il Server potrà essere configurato attraverso un file di configurazione, opportunamente impostato all’avvio, il quale potrà essere anche usato per aggiornare le configurazioni durante l’esecuzione.

**Scelte progettuali:**

Mano a mano che si sviluppava il progetto ci si è resi conto che le dimensioni del codice sarebbero state rilevanti. Per mantenere una buona lettura del codice, “readability”, e quindi facilitarne in mantenimento, ma anche lo sviluppo in sé, il progetto è stato modularizzato in vari moduli e file di supporto/ utilità.

Inoltre la funzione dei moduli e dei file di utilità è anche quella di mascherare le implementazioni specifiche ad un determinato sistema.

**Core:** *server.c client.c*

Al centro del sistema naturalmente ci sono il programma Server, *server.c*, ed il programma Client, *client.c*. Questi organizzano il flusso di esecuzione, assumendo l’input dall’utente, fornendo messaggi informativi a video, e eseguendo le funzioni dei moduli: autenticazione, lancio dei thread, esecuzione dei comandi, logging.

**Modules:** *authentication.h commands.h logging.h thread.h*

Ogni modulo svolge un compito a sé stante, e ricopre sia la parte eseguita dal Server, che la parte eseguita dal Client. Nell’autenticazione, ad esempio, c’è una sequenza di azioni svolte dal server, e una sequenza di risposte fornite dal client, e viceversa. Questa sequenza, o protocollo, è gestita interamente dentro ilo file, o modulo, *authentication.h*, un ‘header’ che ingloba il compito dell’autenticazione. Allo stesso modo l’esecuzione del comando ha come protocollo l’invio del comando richiesto, la valutazione (parsing) del comando, un’esecuzione diretta in caso di comandi da eseguire sul server o l’apertura di file e attesa di dati in una richiesta di trasferimento dati, l’invio di una risposta all’esecuzione del comando, o l’invio di un messaggio di errore. Anche in questo caso, questo flusso di azioni è stato impacchettato nel modulo *commands.h*.

**Utilities:** *constants.h networking.h security.h synchronization.h timing.h*

**Structures:** *connection.h queue.h*