Homework 2

Introduzione

L'homework 2 consiste nella realizzazione di un'applicazione client-server che permetta di:

- trasferire file da un client and un server (scrittura di un file)
- trasferire file da un server ad un client (lettura di un file)

L'applicazione è composta da due programmi, un client che effettua richieste di lettura e scrittura di un file, un server che riceve le richieste dei client e le gestisce.

Operazioni

Il client può effettuare tre operazioni:

- lettura di un file remoto
- scrittura di un file
- ottenere il contenuto di una directory remota

Lettura

Se il client deve leggere un file dal server, i seguenti step vengono effettuati:

1. il client informa il server che vuole effettuare un'operazione di read
2. il server invia al client la dimensione del file in bytes
3. il server invia al client il file

Un esempio di lettura:

1. il server apre la connessione: **`francescopio@francescopio:~/homework2$ ./myFTserver -a 127.0.0.1 -p 9999 -d /home/francescopio/homework2/server/test10`**
2. il client effettua una richiesta di lettura: **`francescopio@francescopio:~/homework2$ ./myFTclient -a`**

```
127.0.0.1 -p 9999 -f /home/francescopio/provasent/testsent.txt
-o ~/provasent/testsent.txt2 -r
```

Il risultato ottenuto è che il file remoto **/home/francescopio/homework2/server/test10/home/francescopio/provasent/testsent.txt** viene scritto nel file locale **~/provasent/testsent.txt2**.

Scrittura

Se il client deve scrivere un file sul server, i seguenti step vengono effettuati:

1. il client informa il server che vuole effettuare un'operazione di write
2. il client invia al server la dimensione del file in bytes
3. il client invia al server il file

Un esempio di scrittura:

1. il server apre la connessione: **francescopio@francescopio:~/homework2$ ./myFTserver -a 127.0.0.1 -p 9999 -d /home/francescopio/homework2/server/test10**
2. il client effettua una richiesta di scrittura: **francescopio@francescopio:~/homework2$ ./myFTclient -a 127.0.0.1 -p 9999 -f /home/francescopio/provasent/testsent.txt -o ~/provasent/testsent.txt2 -w**

Il risultato ottenuto è che il file locale **/home/francescopio/provasent/testsent.txt** viene scritto nel file remoto **/home/francescopio/homework2/server/test10/provasent/testsent.txt2**.

Scrittura concorrente

Potrebbe capitare di avere più client che cercano di scrivere sul medesimo file remoto.

In tal caso, la gestione della race condition mediante mutua esclusione avviene tramite l'utilizzo della strategia di **locking**.

La strategia di **locking** prevede i seguenti step:

1. il server pone un **lock esclusivo** sul file: in questo modo nessun processo potrà accedere al file. Questo avviene mediante l'utilizzo della system call **`flock`** con il flag **`LOCK_EX`** per indicare che il file descriptor ha un lock esclusivo
2. effettua le operazioni di scrittura
3. il server rimuove il lock posto in precedenza. Questo avviene mediante l'utilizzo della system call **`flock`** con il flag **`LOCK_UN`** per rimuovere il lock creato

Contenuto directory

Se il client deve ottenere il contenuto di una directory sul server, i seguenti step vengono effettuati:

1. il client informa il server che vuole effettuare un'operazione di list
2. il server invia al client la lista dei file nella directory

Un esempio di esplorazione contenuto di una directory:

1. il server apre la connessione: **`francescopio@francescopio:~/homework2$ ./myFTserver -a 127.0.0.1 -p 9999 -d /home/francescopio/homework2/server/test10`**
2. il client effettua una richiesta list: **`francescopio@francescopio:~/homework2$ ./myFTclient -a 127.0.0.1 -p 9999 -f /home/francescopio/provasent -l`**

Il seguente output viene stampato a video:

```
[...]
The content of the directory /home/francescopio/provasent is:
[INFO] Message "testsent.txt2" has been received ...
      testsent.txt2
[INFO] Message "testsent.txt" has been received ...
      testsent.txt
```

Parser

I parametri forniti in ingresso al programma **server** sono gestiti tramite apposito parser che verifica la validità dei parametri e popola le struttura dati.

Alcuni casi di non validità:

- opzione senza nome (ad esempio passando solo `-`)
- opzione senza valore
- opzione inesistente (non è nessuna tra `-a`, `-p`, `-d`)
- argomento passato senza alcuna opzione (ad esempio passando solo `8080`)

Ci sono casi di opzioni obbligatorie:

- definire l'indirizzo `-a`
- definire la porta `-p`
- definire la directory principale `-d`

I parametri forniti in ingresso al programma **client** sono gestiti tramite apposito parser che verifica la validità dei parametri e popola le struttura dati.

Alcuni casi di non validità:

- opzione senza nome (ad esempio passando solo `-`)
- opzione senza valore che non sia il comando `-r`, `-w` o `-l` (ad esempio passando `-k`)
- opzione inesistente (non è nessuna tra `-r`, `-w`, `-l`, `-a`, `-p`, `-f`, `-o`)
- argomento passato senza alcuna opzione (ad esempio passando solo `8080`)

Ci sono casi di opzioni esclusive:

- non è possibile definire un mix di operazioni (ad esempio `-r` e `-w` oppure `-l` e `-r` e così via)

Ci sono casi di opzioni obbligatorie:
- definire almeno una tra `-r`, `-w` e `-l`
- definire l'indirizzo `-a`
- definire la porta `-p`
- definire il path sorgente `-f`

Progetto

Il progetto è strutturato nel seguente modo:
- programma server
- programma client
- parte di utility per funzionalità condivise:
    - utility per gestire la connessione tramite socket
    - utility per ricevere messaggi, comandi, file
    - utility per ricevere messaggi, comandi, file
    - utility per funzioni generali

Startup

Build

La fase di compilazione e linking è automatizzata mediante l'uso di Make.

Quindi vi è la presenza del file Makefile che definisce:
- compilazione delle utility (utils, connection, sender e receiver)
- compilazione di server e creazione dello script **`myFTserver`**
- compilazione di client e creazione dello script **`myFTclient`**

Per eseguire la build del programma posizionarsi nella cartella dei sorgenti ed eseguire il comando `make` come mostrato di seguito:

```
francescopio@francescopio:~/homework2$ make
cc -c utils/utils.c
cc -c connection/connection.c
cc -c sender/sender.c
cc -c receiver/receiver.c
cc -c server/parser.c -o server-parser.o
cc -c server/server.c
cc -o myFTserver utils.o connection.o sender.o receiver.o
server-parser.o server.o
cc -c client/parser.c -o client-parser.o
cc -c client/client.c
```

```
cc -o myFTclient utils.o connection.o sender.o receiver.o
client-parser.o client.o
```

Run

Per eseguire il server è necessario posizionarsi nella cartella dei sorgenti ed eseguire lo script denominato **`myFTserver`** nel seguente modo:
`francescopio@francescopio:~/homework2$ ./myFTserver ...`

Per eseguire il client è necessario posizionarsi nella cartella dei sorgenti ed eseguire lo script denominato **`myFTclient`** nel seguente modo:
`francescopio@francescopio:~/homework2$ ./myFTclient ...`