

Programmazione distribuita I

(01NVWOV)

AA 2010-2011, Esercitazione di laboratorio n. 1

Esercizio 1.1 (server di prova)

In ambiente linux, compilare il server di prova fornito, che si mette in ascolto sulla porta specificata, e somma i due numeri ricevuti.

Utilizzare il comando:

```
gcc -o server_test -DTRACE server_test.c errlib.c sockwrap.c
```

Per avviarlo:

```
./server_test formato porta
```

ove *formato* è una opzione che specifica il formato dei dati usato nel dialogo col client:

- **-a** per la rappresentazione dei dati in ASCII
- **-x** per la rappresentazione dei dati tramite XDR

e *porta* è il numero di porta sulla quale il server si mette in ascolto (per es. 1500).

Testare il funzionamento del server usando l'opzione **-a** e come client l'applicazione **telnet**, che prende come parametro l'indirizzo del server cui collegarsi e la porta.

Nota: Per interrompere il collegamento all'interno del programma telnet premere CTRL +] e poi q seguito da INVIO.

Nota 2: Per comodità, è possibile utilizzare le funzioni presenti in **sockwrap.c** (Socket, Connect, etc...) che sono l'esatta replica delle funzioni di libreria ma hanno già integrato il controllo dei valori di ritorno delle funzioni per segnalare e stampare eventuali errori.

Esercizio 1.2 (connessione)

Scrivere un client che si colleghi ad un server TCP all'indirizzo e porta specificati come primo e secondo parametro sulla riga di comando e quindi termini chiudendo correttamente il canale e segnalando se il collegamento è riuscito o fallito.

Esercizio 1.3 (dati ASCII)

Sviluppare un client che si colleghi ad un server TCP all'indirizzo e porta specificati come primo e secondo parametro sulla riga di comando e quindi invii due numeri interi senza segno (rappresentabili in binario puro su 16 bit) letti da standard input e riceva la risposta (somma, o errore) dal server. Come server può essere utilizzato quello compilato nell'esercizio 1.1

La somma è un numero intero senza segno a 16 bit, ma il server gestisce anche il caso di overflow, che è segnalato con uno specifico errore (si veda l'esempio).

Il client invia al server i due numeri in notazione decimale espressi mediante caratteri numerici ASCII. I numeri devono essere separati da un solo spazio e la trasmissione deve essere terminata con CR-LF. Il server restituisce il risultato (un solo numero) espresso mediante caratteri ASCII, privo di spazi e terminato con CR-LF oppure un messaggio di errore (caratteri ASCII) terminato anch'esso da CR-LF.

Esempi (ogni elemento rappresenta un singolo byte trasmesso in codice ASCII):

```
(client → server) 1 2 3 4 5      3  CR LF
```

(server → client) 1 2 3 4 8 CR LF

o in caso di errore:

(server → client) o v e r f l o w CR LF

(server → client) i n c o r r e c t o p e r a n d s CR LF

Esercizio 1.4 (client-server UDP base)

Scrivere un client che invii ad un server UDP (all'indirizzo e porta specificati come primo e secondo parametro sulla riga di comando) un datagramma contenente il nome (massimo 31 caratteri) passato come terzo parametro sulla riga di comando, attenda quindi un qualunque datagramma di risposta dal server. Il client termina mostrando il contenuto (testo ASCII) del datagramma ricevuto oppure segnalando che non ha ricevuto risposta dal server entro un certo tempo.

Sviluppare un server UDP (in ascolto sulla porta specificata come primo parametro sulla riga di comando) che risponda ad ogni datagramma ricevuto inviando un datagramma di risposta contenente lo stesso nome presente nel pacchetto ricevuto.

Provare ad effettuare degli scambi di datagrammi tra client e server con le seguenti configurazioni:

- client invia datagramma ad una porta su cui il server è in ascolto
- client invia datagramma ad una porta su cui il server non è in ascolto
- client invia datagramma ad un indirizzo non raggiungibile (es. 10.0.0.1)

Provare a collegare il proprio client col server sviluppato da un altro gruppo e viceversa.