Università di Ferrara

Architettura di Reti

Lab 1

Carlo Giannelli
carlo.giannelli@unife.it
http://www.unife.it/scienze/informatica/insegnamenti/architettura-reti/
http://docente.unife.it/carlo.giannelli

Linguaggio C

- Compilazione:
  - gcc -g -Wall sorgente.c -o eseguibile
- Esecuzione:
  - ./eseguibile {parametri}
- Debugging
  - gdb ./eseguibile
  - run {parametri}
  - brack {nome funzione}, step, next, list, info locals…

Server Concorrente vs. Sequenziale

- Server *sempre in attesa* (ciclo infinito)
- *Delega* delle operazioni al processo figlio

```
for(;;){
    int ns, pid;
    ns = accept(sd, NULL, NULL);
    if ((pid = fork()) == 0)
    { /* FIGLIO che esegue le operazioni specifiche */
        /* Chiudo la socket passiva */
        close(sd);
        exit(1);
    }
    /* PADRE sempre in attesa*/
    /* Chiudo la socket attiva */
    close(ns);
}
```

Operazioni sul modello C/S in C

GetAddrInfo(.) Risoluzione dei nomi; restituisce la lista strct addrinfo in cui ogni nodo rappresenta un indirizzo fisico del servizio	
I/O ← Socket(.) → socket passiva Restituisce il socket-descriptor	
Connect(.) Richiede connessione tra socket locale e remota	Bind(.) Aggancio socket-descriptor alla porta
FreeAddrInfo(.) Libera memoria allocata da getaddrinfo terminato il suo utilizzo	
	Listen(.) Trasforma il socket-descriptor da attivo a passivo
	Accept(.) Estrae richiesta di connessione dalla listen

Esercizio 1

Si realizzi un'applicazione distribuita in C/Unix che, sfruttando la socket API, implementi un servizio di conteggio remoto del numero di caratteri contenuti in una stringa. Il Client deve presentare la seguente interfaccia:

rstrlen hostname porta stringa

dove "hostname" è il nome dell'host dove risiede il Server, "porta" è il numero di porta a cui esso è associato e "stringa" è la stringa di cui si vuole contare il numero di caratteri.

Per prima cosa, il Client deve inviare al Server la stringa a cui l'utente è interessato. Il Server deve quindi contare il numero di caratteri in essa e restituirlo al Client, che lo stamperà a video.

Gestione read/write di stringhe

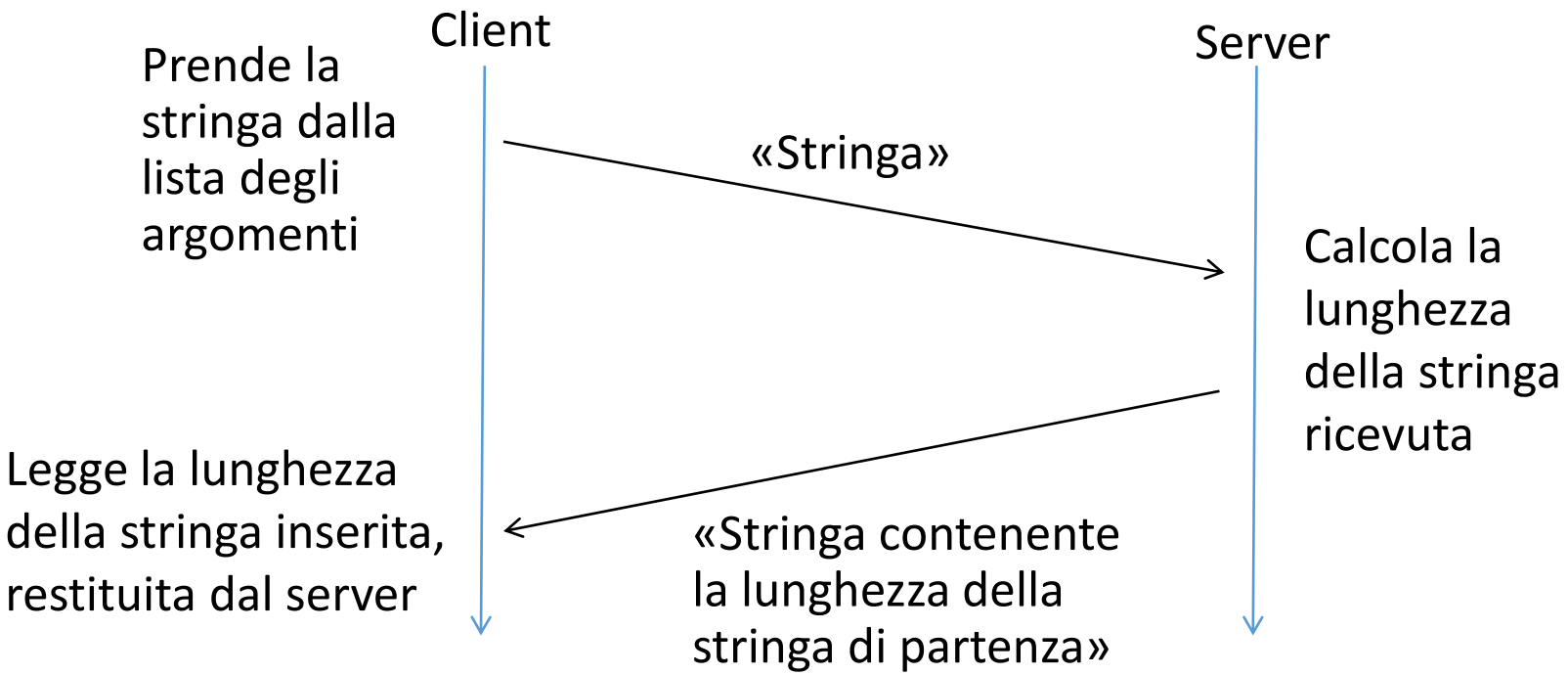
Che dimensione hanno le stringhe inviate?
Come possiamo essere certi che le read() leggano le stringhe **interamente**?

- Opzione 1: **ipotesi fortemente semplificativa**, le stringhe sono piccole e vengono sempre consegnate in un unico segmento TCP
  - la write() invia la stringa
  - un'unica read() legge l'intera stringa inviata

- Opzione 2: si inviano sempre e solo buffer di dimensioni note a priori
  - la write() invia un buffer di dimensione nota, parzialmente utilizzato per contenere la stringa: potenzialmente elevato **overhead di trasmissione!**
  - la read() legge l'intero buffer e poi ne estrae la stringa

- Opzione 3: si invia prima la dimensione della stringa, poi la stringa vera e propria
  - la write() invia (ad esempio) un byte con la dimensione e poi la stringa
  - la read() legge un byte e poi la stringa, **2 messaggi invece che 1**
  - se un byte per la dimensione → dimensione massima della stringa 2^8-1

rstrlen
protocollo con stringhe in singoli segmenti TCP



Librerie

```
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/stat.h>
#include <sys/wait.h>
#include <fcntl.h>
#include <netdb.h>
#include <netinet/in.h>
#include <signal.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
```

Esercizio 2

Si realizzi un'applicazione distribuita in C/Unix che, sfruttando la socket API, implementi un servizio di confronto remoto fra stringhe. Il Client deve offrire la seguente interfaccia:

rstrcmp hostname porta stringa1 stringa2

dove "hostname" è il nome dell'host dove risiede il Server, "porta" è il numero di porta a cui esso è associato e "stringa1" e "stringa2" sono le stringhe che si vogliono confrontare.

Per prima cosa, il Client deve inviare al Server le stringhe di cui l'utente desidera il confronto. Il Server deve quindi effettuare il confronto lessicografico tra le stringhe e restituire al client "SI" se le stringhe sono identiche, "NO" in caso contrario. Il Client deve stampare a video il risultato ricevuto dal Server e terminare.

Attenzione! Ipotesi di lavoro:
stringhe nel complesso piccole inviate all'interno di un unico segmento TCP.

Message boundaries

In TCP non esiste il concetto di message boundary.
Quando si inviano due stringhe consecutive, come capire dove finisce la prima e dove inizia la seconda?

Generalizzando, **in caso di messaggi consecutivi da stesso mittente a stesso destinatario, come capire quando termina un messaggio e inizia quello successivo?**

Alcuni esempi di possibile soluzione:

- messaggi/campi di dimensione nota (visto precedentemente)
- invio dimensione messaggio/campo e poi messaggio/campo vero e proprio (visto precedentemente)
- messaggi/campi con terminatore
- attesa di ack a livello applicativo a valle dell'invio di un messaggio/campo

Stringhe multiple

- Opzione 1: stringhe seguite da **terminatori**
  - terminatore di stringa per identificare la fine dei dati inviati dalla prima write()
  - è necessario leggere byte a byte alla ricerca del terminatore
  - **overhead a causa del frequente switch User/Kernel space**

- Opzione 2: stringhe intervallate da **ack a livello applicativo**
  - ack applicativo da non confondere con l'ack TCP a livello di trasporto!
  - ack esplicitamente inviato dal ricevente dopo la ricezione della prima stringa
  - Il mittente effettua la seconda write() solo a valle della ricezione dell'ack
  - **ritardi di trasmissione** causati dall'attesa dell'ack prima di poter inviare messaggi consecutivi (N messaggi → N * RTT)

rstrcmp
protocollo con ack applicativo