Socket - Indirizzi

Esistono anche delle funzioni per convertire gli indirizzi dal formato rete binario al *dotted decimal*, o *decimale puntato*, caratteristico degli indirizzi IPv4; tali funzioni sono definite in **<arpa/inet.h>** e i seguenti sono i loro prototipi:

int inet_aton(const char *src, struct in_addr *dest)
converte la stringa *src* decimale puntata in un indirizzo IP;
char *inet_ntoa(struct in_addr ind)
converte l'indirizzo IP *ind* in una stringa decimale puntata.

Socket - Indirizzi

Esempio indirizzi: Vediamo alcuni esempi di uso delle funzioni precedentemente introdotte.

Esercizio indirizzi: Viste le due funzioni precedenti scrivere un programma che dato come argomento da linea di comando 1 indirizzo in formato dotted decimal, lo converta in in_addr.

Socket - IOCTL

int ioctl(int d, int request, ...);
ioctl e' una funzione che permette di gestire i device quando non sono piu' sufficienti le normali funzioni di interazione con i file (open, read, write, close).

Il suo utilizzo e' chiaramente specifico per il tipo di dispositivo. E' possibile ad esempio usare la IOCTL per determinare MAC, IP ed altro della scheda di rete.

E' ad esempio possibile, mediante l'uso della IOCTL, configurare un'interfaccia di rete in promiscuous mode.

Socket - connessione

socket pair: con tale termine si intende la combinazione di quattro valori che identificano i due estremi della comunicazione: *IP_locale:porta_locale, IP_remoto:porta_remota*.

La sequenza di operazioni da svolgere per gestire un socket TCP è:

1. creazione del socket
2. assegnazione dell'indirizzo
3. connessione o attesa di connessione
4. invio o ricezione dei dati
5. chiusura del socket

Socket - connessione

int bind(int sockfd, const struct sockaddr *my_addr, socklen_t addrlen);
bind e' la funzione che serve ad assegnare l'indirizzo e la porta sulla quale il servente si deve mettere in ascolto. La funzione ritorna zero o -1 in caso di errore.

Vediamo dunque gli argomenti: **sockfd** e' l'identificativo (o *file descriptor*) del socket ottenuto dalla creazione con la funzione **socket();** **my_addr** è l'indirizzo; ed in fine **addrlen** e' la lunghezza dell'indirizzo.

Socket - connessione

Costante	Significato
INADDR_ANY	Indirizzo generico (0.0.0.0)
INADDR_BROADCAST	Indirizzo di broadcast.
INADDR_LOOPBACK	Indirizzo di loopback (127.0.0.1).
INADDR_NONE	Indirizzo errato.

Socket - connessione

int listen(int sockfd, int backlog);

la funzione **listen** serve a specificare che un processo e' in ascolto. Anche questa funzione ritorna zero oppure -1 in caso di errore. Al solito **sockfd** e' l'identificativo del socket, mentre **backlog** specifica il numero massimo di connessioni che il processo puo' accettare, dal kernel 2.2 e' la lunghezza delle sole connessioni completate per evitare attacchi SYN flood (cenni di SYN cookies) (serve solo per SOCK_STREAM).

Esempio listen: vediamo un semplice esempio di listen (utilizzo del programma netstat).

Socket - connessione

int accept(int sockfd, struct sockaddr *addr, socklen_t *addrlen);
La funzione **accept** serve appunto a stabilire una connessione da parte del server. Essa ritorna un valore maggiore di zero, oppure -1 in caso di errore. Il valore ritornato e' il cosi' detto socket connesso, tale socket ha le stesse caratteristiche del socket originale **sockfd** che invece resta in ascolto. **addr** e' l'indirizzo del cliente che ha inviato la richiesta di connessione, ed al solito **addrlen** e' la lunghezza dell'indirizzo (perche' si passano due puntatori alla funzione ?)

Socket - connessione

Esempio accept: esempio di accept. Prima provate voi a modificare l'esempio listen aggiungendo 1 accept , il “server” dovra' stampare anche indirizzo e porta del client. (scrivere anche il Makefile)