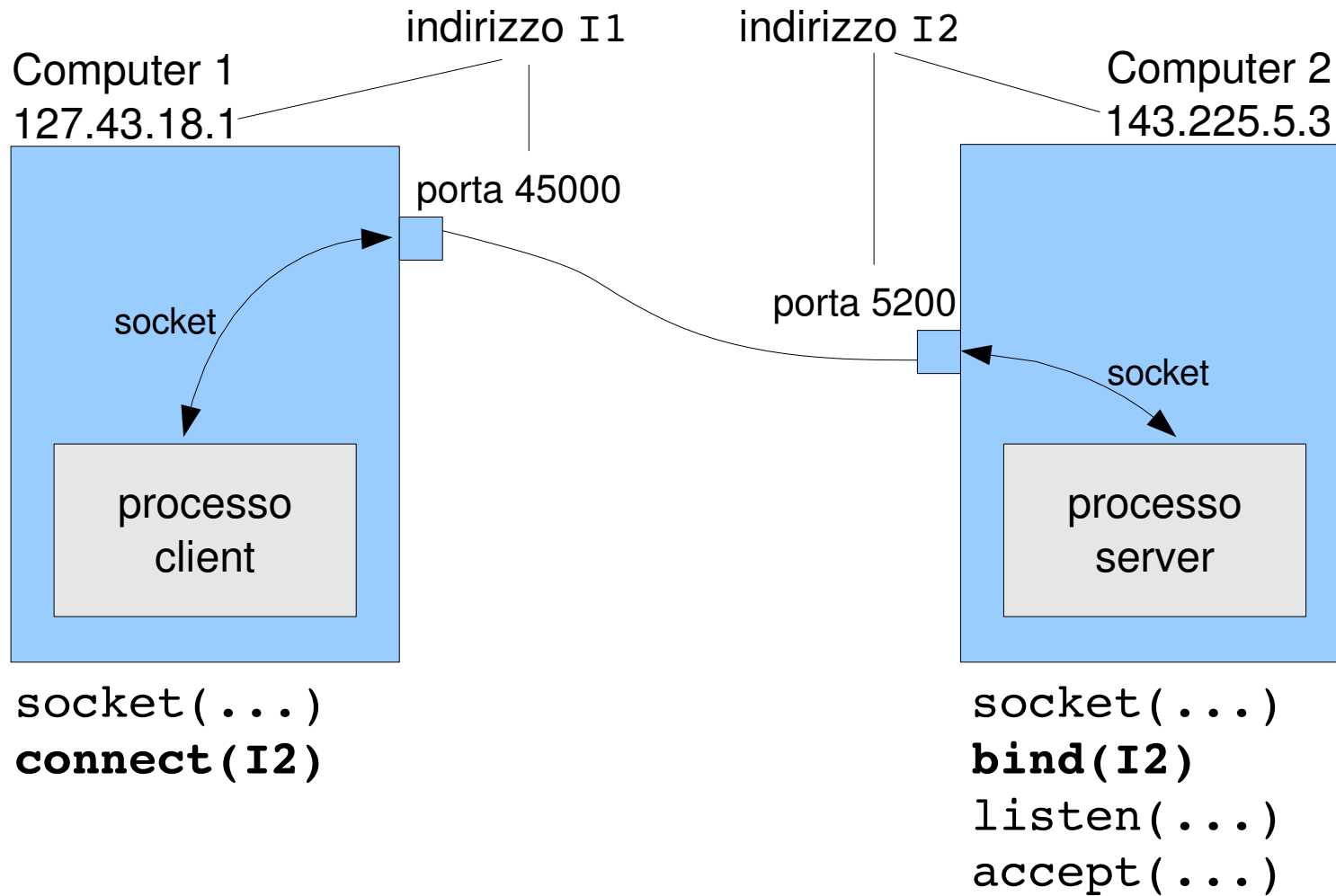


Socket TCP

seconda parte

Schema della connessione



Comandi utili: netstat

- `netstat [-t] [-all] [-p] [-n]`
 - elenca tutti i socket di rete del sistema (non riguarda i socket locali)
 - “-t” mostra solo i socket TCP, cioè quelli con famiglia=PF_INET e tipo=SOCK_STREAM
 - “-all” (oppure “-a”) mostra anche i socket in ascolto
 - “-p” specifica il pid del processo che ha creato ciascun socket
 - “-n” mostra gli indirizzi e le porte in formato numerico (invece di simbolico)
 - ad es., `netstat -t -a -p -n` mostra tutti i socket TCP aperti, indicando il PID del processo corrispondente e mostrando gli indirizzi in formato numerico

Comandi utili: netstat

```
netstat -t -a -p
```

| Proto | Recv-Q | Send-Q | Local Address | Foreign Address | State | PID/Program name |
|-------|--------|--------|-------------------|---------------------|-------------|------------------|
| tcp | 0 | 0 | *:sunrpc | *:* | LISTEN | 6519/portmap |
| tcp | 0 | 0 | *:ipp | *:* | LISTEN | 6677/cupsd |
| tcp | 0 | 0 | localhost:smtp | *:* | LISTEN | 6612/master |
| tcp | 0 | 0 | *:ssh | *:* | LISTEN | 6873/sshd |
| tcp | 0 | 0 | pcfaella.na:44619 | alfa.na.infn.it:ssh | ESTABLISHED | 17378/ssh |

Comandi utili: netstat

```
netstat -t -a -p
```

indirizzo
(INADDR_ANY)

porta (111)

| Proto | Recv-Q | Send-Q | Local Address | Foreign Address | State | PID/Program name |
|-------|--------|--------|-------------------|---------------------|-------------|------------------|
| tcp | 0 | 0 | *:sunrpc | *:* | LISTEN | 6519/portmap |
| tcp | 0 | 0 | *:ipp | *:* | LISTEN | 6677/cupsd |
| tcp | 0 | 0 | localhost:smtp | *:* | LISTEN | 6612/master |
| tcp | 0 | 0 | *:ssh | *:* | LISTEN | 6873/sshd |
| tcp | 0 | 0 | pcfaella.na:44619 | alfa.na.infn.it:ssh | ESTABLISHED | 17378/ssh |

porta effimera

client
ssh

server
ssh

Comandi utili: telnet

- `telnet <indirizzo> <porta>`
 - crea un socket e lo collega all'indirizzo remoto dato
 - esempio: “telnet www.unina.it 80” si mette in comunicazione con il server http dell'università
 - quello che si digita da terminale viene mandato sul socket
 - quello che proviene dal socket viene stampato su stdout
 - telnet può quindi fungere da “client generico”

Altri comandi utili

- `/sbin/ifconfig`
 - mostra l'indirizzo IP della macchina corrente
- `nslookup <nome di dominio>`
 - fornisce l'indirizzo IP di un host del quale conosciamo il nome
 - esempio: “`nslookup www.unina.it`”

Server concorrenti

- Per gestire più client contemporaneamente, un server può generare un nuovo figlio per ogni nuova connessione
- Il server esegue una `fork` dopo ogni `accept`
 - il padre torna subito ad eseguire una nuova `accept`
 - il figlio gestisce la connessione e poi termina
- In alternativa, si possono usare i `thread`

Schema di un server concorrente

```
socket(...);
bind(...);
listen(...);

while (1) {
    fd2 = accept(fd1, (struct sockaddr *)NULL, NULL);

    if ( (pid = fork()) < 0 ) {
        perror("fork");
        exit(1);
    } else if (pid == 0) {          // processo figlio
        close(fd1); // al figlio non serve fd1
        // gestisce la connessione usando fd2
        ...
        exit(0);    // poi il figlio termina
    }
    // il processo padre chiude fd2 e ripete il ciclo
    close(fd2);
}
```

Nomi di dominio

- Ad un host può venir assegnato un nome di dominio (*domain name*), come www.unina.it
- Il servizio di rete chiamato DNS (*Domain Name Service*) converte i nomi di dominio in indirizzi IP
 - www.unina.it → 143.225.5.3
 - questa conversione prende il nome di “risoluzione del nome”, dall'inglese “domain name resolution”
- Dalla shell, il comando `nslookup` esegue la conversione
 - `nslookup <nome>`
 - esempio: `nslookup www.unina.it`

Nomi di dominio

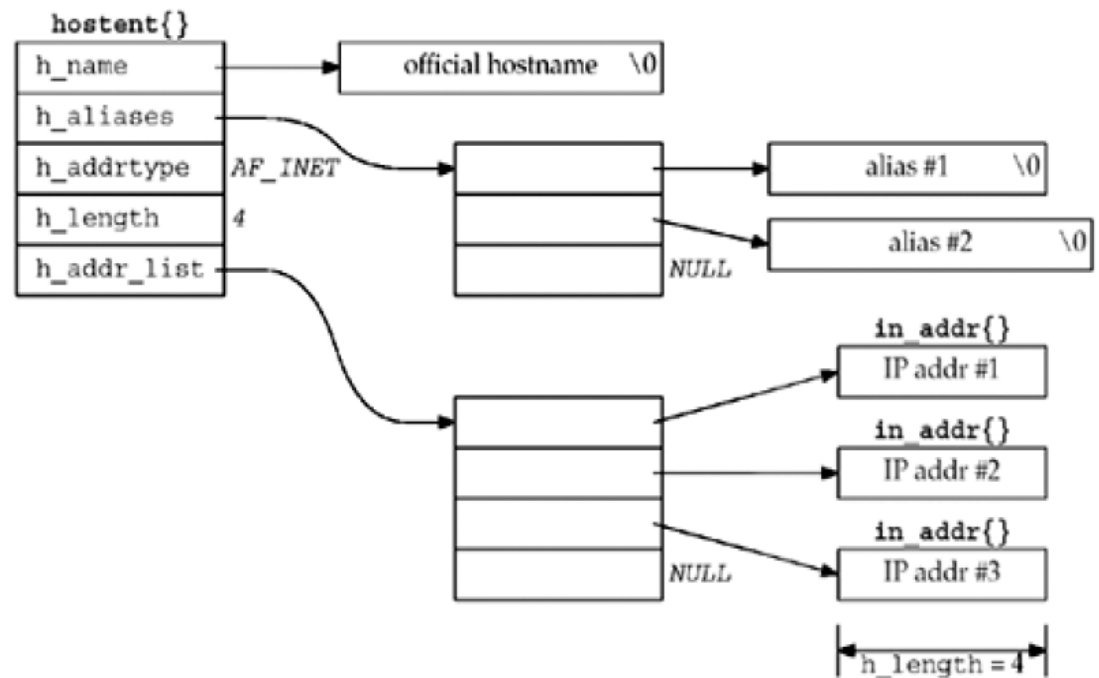
```
struct hostent *gethostbyname(const char *nome);
```

```
struct hostent {  
    char    *h_name;           nome canonico dell'host  
    char    **h_aliases;      lista di alias  
    int     h_addrtype;       famiglia dell'indirizzo (AF_INET)  
    int     h_length;         lunghezza dell'indirizzo (4)  
    char    **h_addr_list;   lista di indirizzi  
}
```

- Restituisce l'indirizzo IP (oltre ad altre informazioni) corrispondente al nome di dominio dato
- L'indirizzo si trova in `h_addr_list[0]`, già in network order
- Un nome può corrispondere a più indirizzi `h_addr_list[0]`, `h_addr_list[1]`, ...

Nomi di dominio

- Struttura hostent
[Stevens]



```
struct sockaddr_in indirizzo;  
  
struct hostent *p = gethostbyname("www.unina.it");  
if (!p) herror("gethostbyname"), exit(1);  
indirizzo.sin_addr.s_addr = *(uint32_t *) (p->h_addr_list[0]);
```

Nomi di dominio

- Restituisce NULL in caso di errore
- La struttura con il risultato viene riutilizzata ad ogni nuova chiamata!
 - Quindi, `gethostbyname` non è thread-safe
 - C'è una versione thread-safe chiamata `gethostbyname_r`
- Accetta anche un indirizzo in formato dotted
 - Quindi, rimpiazza anche `inet_aton`

Stampare un indirizzo IP

- Partiamo da un indirizzo IP in network order, come quello ottenuto da `accept`

```
char *inet_ntoa(struct in_addr in);
```

- Converte l'indirizzo in una stringa in formato dotted
- La stringa viene sovrascritta da ogni nuova chiamata
 - `inet_ntoa` non è thread-safe

Informazioni utili

- Indirizzo server Centri Comuni (Infserv64)
 - 143.225.178.5
- Indirizzo server Biologia (Infserv64bio)
 - 143.225.117.252
- Per trasferire file da un server a un altro
 - comando `scp` (secure copy)

Informazioni utili

Dimensione dei tipi base

| | Pentium | AMD64 |
|--------|---------|-------|
| void * | 4 | 8 |
| int | 4 | 4 |
| long | 4 | 8 |
| short | 2 | 2 |

Esercizio 1

- Modificare il server dell'esercizio 1 dei socket TCP, in modo che sia concorrente
- Modificare il client in modo che accetti come indirizzo IP del server anche un nome di dominio
 - esempi:
 - `./client www.unina.it 5510`
 - `./client localhost 5510`
 - `./client 143.225.5.3 5510`