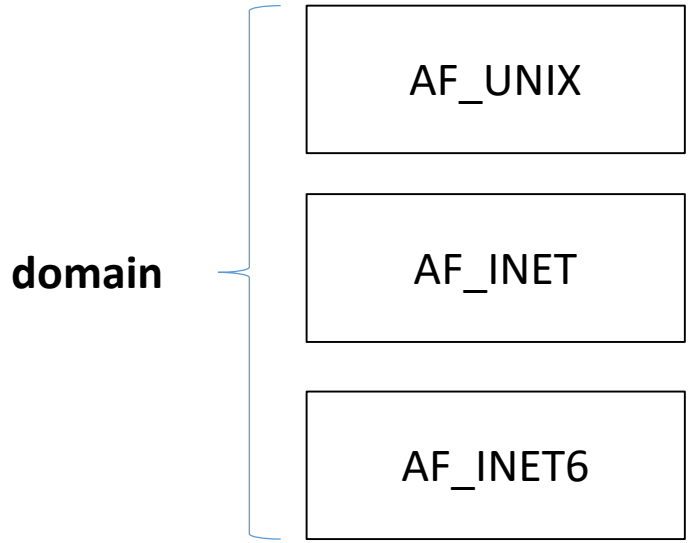
sockets

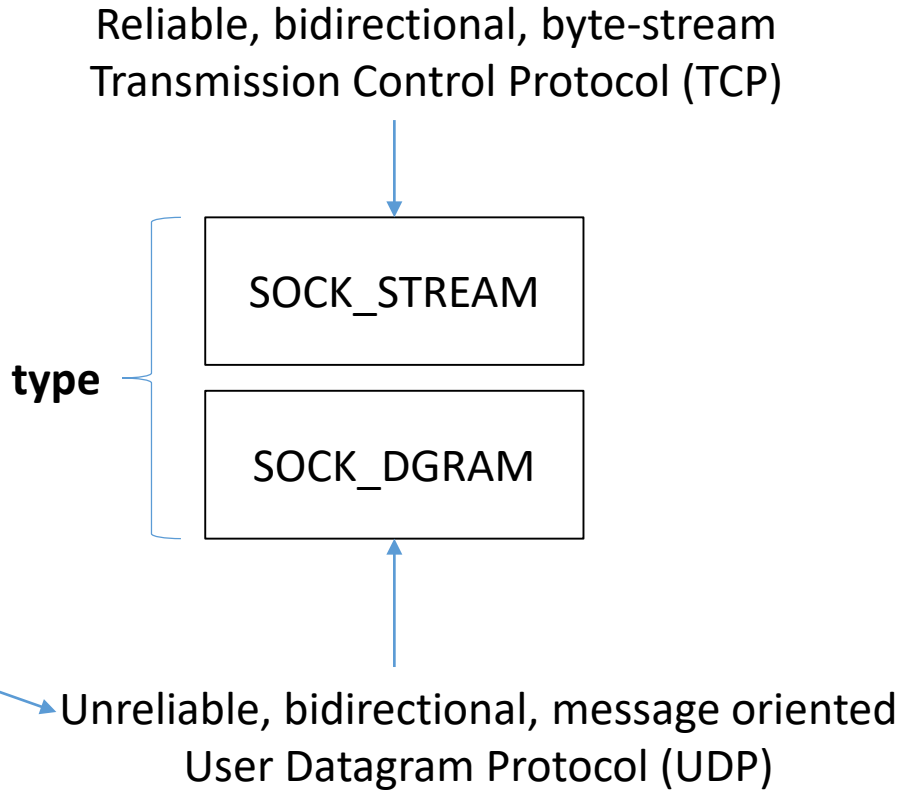
- Tipologia di Inter Process Communication
- Cosa: scambio dati bidirezionale tra applicazioni
- Dove: sullo stesso host o su host diversi connessi da una rete
- Scenario client-server:
- Ogni applicazione crea un «socket»
- Il socket è l'apparato che permette la comunicazione
- Sia il client che il server richiedono un socket ciascuno
- Il server collega (bind) il suo socket ad un indirizzo preciso (di rete), così i client possono trovarlo (o rintracciarlo)

Creare un socket:

fd = socket(domain, type, protocol);

«communication domain» (indirizzo, protocolli)





Domain	Communication performed	Communication between applications	Address format	Address structure
AF_UNIX	within kernel	on same host	pathname	*sockaddr_un*
AF_INET	via IPv4	on hosts connected via an IPv4 network	32-bit IPv4 address + 16-bit port number	*sockaddr_in*
AF_INET6	via IPv6	on hosts connected via an IPv6 network	128-bit IPv6 address + 16-bit port number	*sockaddr_in6*

Socket system calls

- **socket()** crea un nuovo socket
- **bind()** associa o lega un socket ad un indirizzo (così i client possono "trovare" il socket del server ed attaccarsi con connect())
- **listen()** dice al kernel che un socket di tipo SOCK_STREAM deve accettare connessioni entranti da altri socket (il socket diventa "passivo")
- **accept()** dice al kernel di utilizzare un socket (SOCK_STREAM su cui ho prima invocato listen()) per accettare una eventuale connessione entrante da un peer (blocca fino a che arriva una connessione)
- **connect**() stabilisce una connessione tra il socket (attivo) passato come parametron ed un altro socket (passivo)

Socket I/O

- read(), write()
- send(), recv(), sendto(), recvfrom()
- Default: bloccano se l'operazione di I/O non può essere completata immediatamente

- close(), shutdown()

int **socket**(int domain, int type, int protocol);

- #include <sys/socket.h>
- int **socket**(int domain, int type, int protocol);
- Returns file descriptor on success, or –1 on error

- Domain: AF_UNIX, AF_INET, AF_INET6
- Type: stream (SOCK_STREAM) o datagram (SOCK_DGRAM)

bind()

- int **bind**(int sockfd, const struct sockaddr *addr, socklen_t addrlen);
- Returns 0 on success, or –1 on error

- Specifico l’indirizzo a cui agganciare il socket (per renderlo rintracciabile)
- AF_UNIX -> pathname, AF_INET -> indirizzo_ip:numero_porta
- struct sockaddr è generico
- struct sockaddr_un per AF_UNIX
- struct sockaddr_in per AF_INET, struct sockaddr_in6 per AF_INET

listen()

- int **listen**(int sockfd, int backlog);
- Returns 0 on success, or –1 on error

- Solo su stream sockets, non già connessi ad un altro socket
- Marca il socket come “passive”
- Backlog: dimensione della coda (gestita dal kernel) delle connessioni entranti sui cui non ho ancora chiamato accept()

accept()

- int **accept**(int sockfd, struct sockaddr *addr, socklen_t *addrlen);
- Returns file descriptor on success, or –1 on error

- The accept() system call accepts an incoming connection on the listening stream socket referred to by the file descriptor sockfd.
- If there are no pending connections when accept() is called, the call blocks until a connection request arrives.

- accept() va invocato su un socket “passivo”

SOCK_STREAM

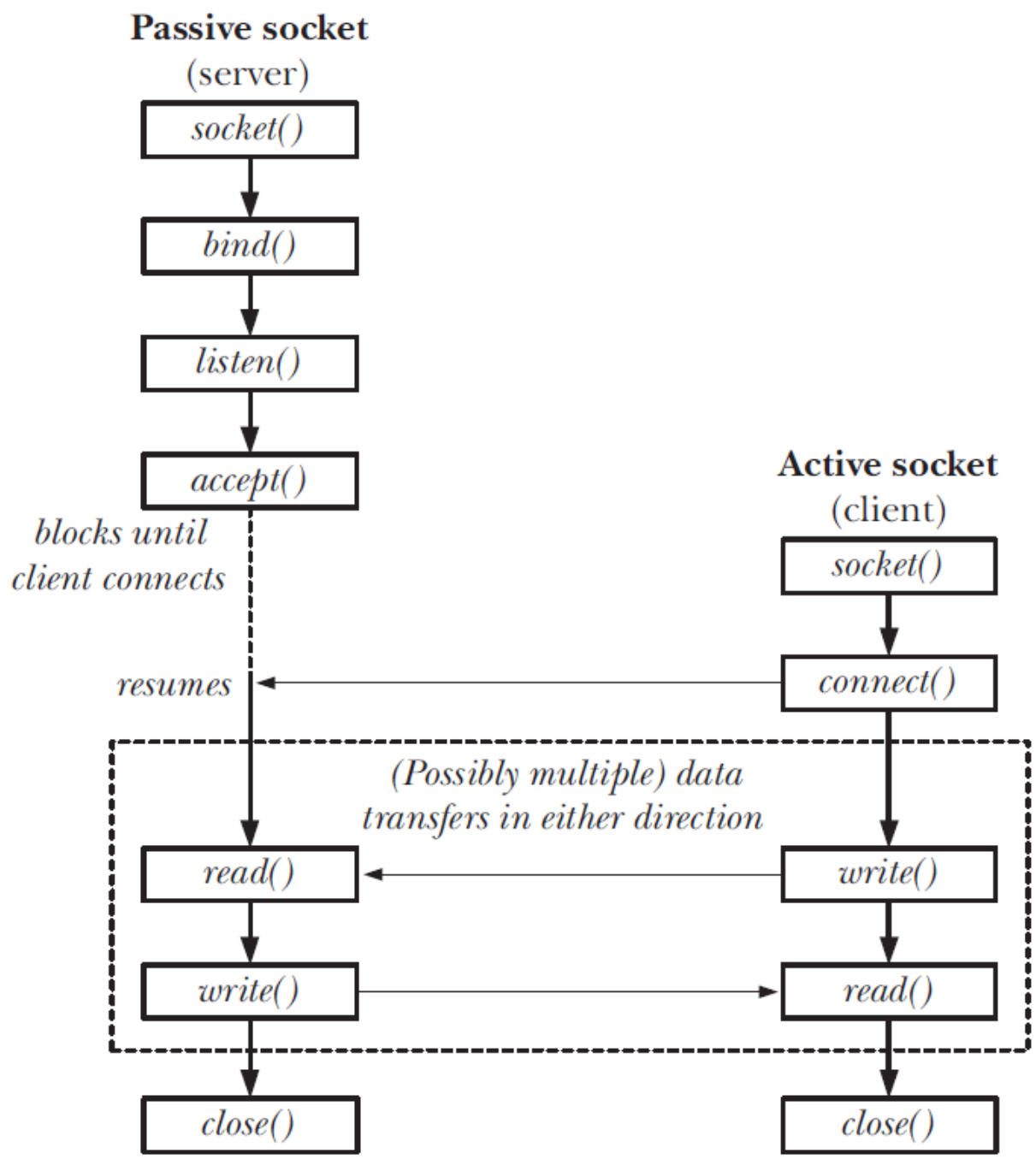


Figure 56-1: Overview of system calls used with stream sockets

courtesy Kerrisk, pag. 1156

connect()

- int **connect**(int sockfd, const struct sockaddr *addr, socklen_t addrlen);
- Returns 0 on success, or –1 on error

- connette il socket attivo (sockfd) al socket passivo in ascolto, identificato dall’indirizzo addr

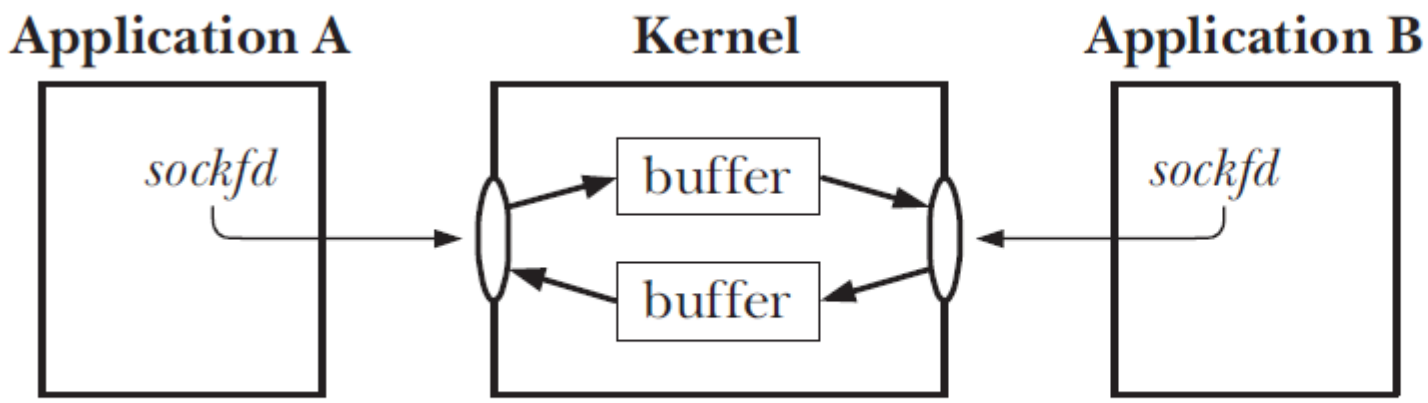


Figure 56-3: UNIX domain stream sockets provide a bidirectional communication channel

courtesy Kerrisk, pag. 1159