# **1. socket**

📌 **Descripción:**  
 Crea un nuevo **socket**, que es un punto final para la comunicación en red.

📌 **Uso:**  
 Se usa para establecer una conexión de red, ya sea para **TCP** (orientado a conexión) o **UDP** (sin conexión).

📌 **Ejemplo real:**  
 Un servidor web crea un socket para escuchar conexiones entrantes.

*c*

*int sockfd = socket(AF\_INET, SOCK\_STREAM, 0);*  
*if (sockfd == -1) {*  
 *perror("Error al crear el socket");*  
 *exit(1);*  
*}*

# **🔹 2. close**

📌 **Descripción:**  
 Cierra un socket, liberando sus recursos.

📌 **Uso:**  
 Se debe llamar cuando ya no se necesita el socket para evitar fugas de recursos.

📌 **Ejemplo real:**  
 Un servidor HTTP cierra la conexión después de enviar la respuesta.

*c*

*close(sockfd);*

# **🔹 3. setsockopt**

📌 **Descripción:**  
 Modifica opciones del socket, como el tiempo de espera o la reutilización de direcciones.

📌 **Uso:**  
 Se usa para configurar el comportamiento del socket, como habilitar la opción SO\_REUSEADDR.

📌 **Ejemplo real:**  
 Evitar el error "Address already in use" al reiniciar un servidor.

*c*

*int opt = 1;*  
*setsockopt(sockfd, SOL\_SOCKET, SO\_REUSEADDR, &opt, sizeof(opt));*

# **🔹 4. getsockname**

📌 **Descripción:**  
 Obtiene la dirección y puerto en los que un socket está ligado.

📌 **Uso:**  
 Útil para conocer en qué puerto aleatorio está escuchando un socket.

📌 **Ejemplo real:**  
 Un servidor que se ejecuta en un puerto dinámico y quiere informar a los clientes.

c

struct sockaddr\_in addr;  
socklen\_t len = sizeof(addr);  
getsockname(sockfd, (struct sockaddr \*)&addr, &len);  
printf("Escuchando en el puerto %d\n", ntohs(addr.sin\_port));

# **🔹 5. getprotobyname**

📌 **Descripción:**  
 Obtiene el número de protocolo de un nombre, como "tcp" o "udp".

📌 **Uso:**  
 Se usa para configurar un socket correctamente con el protocolo adecuado.

📌 **Ejemplo real:**  
 Un programa que quiere obtener el número de protocolo de **TCP**.

c

struct protoent \*proto = getprotobyname("tcp");  
printf("Protocolo TCP número: %d\n", proto->p\_proto);

# **🔹 6. gethostbyname**

📌 **Descripción:**  
 Convierte un nombre de dominio en una dirección IP.

📌 **Uso:**  
 Se usa para resolver nombres de dominio a direcciones IP antes de conectarse.

📌 **Ejemplo real:**  
 Un navegador obtiene la IP de "www.google.com".

c

struct hostent \*host = gethostbyname("www.google.com");  
printf("IP: %s\n", inet\_ntoa(\*(struct in\_addr \*)host->h\_addr));

# **🔹 7. getaddrinfo y freeaddrinfo**

📌 **Descripción:**  
 getaddrinfo obtiene información de direcciones de red.  
 freeaddrinfo libera la memoria de getaddrinfo.

📌 **Uso:**  
 Se usa en aplicaciones modernas en lugar de gethostbyname.

📌 **Ejemplo real:**  
 Un cliente busca la IP de un servidor antes de conectarse.

c

struct addrinfo hints, \*res;  
memset(&hints, 0, sizeof(hints));  
hints.ai\_family = AF\_INET;  
hints.ai\_socktype = SOCK\_STREAM;  
getaddrinfo("www.google.com", "80", &hints, &res);  
freeaddrinfo(res);

# **🔹 8. bind**

📌 **Descripción:**  
 Asigna un socket a un puerto y dirección IP.

📌 **Uso:**  
 Un servidor usa bind para especificar en qué puerto escuchará.

📌 **Ejemplo real:**  
 Un servidor HTTP se asocia al puerto 80.

c

struct sockaddr\_in server;  
server.sin\_family = AF\_INET;  
server.sin\_addr.s\_addr = INADDR\_ANY;  
server.sin\_port = htons(80);  
bind(sockfd, (struct sockaddr \*)&server, sizeof(server));

# **🔹 9. connect**

📌 **Descripción:**  
 Conecta un socket a un servidor.

📌 **Uso:**  
 Un cliente usa connect para iniciar una conexión.

📌 **Ejemplo real:**  
 Un navegador se conecta a un servidor web.

c

connect(sockfd, (struct sockaddr \*)&server, sizeof(server));

# **🔹 10. listen y accept**

📌 **Descripción:**  
 listen pone el socket en modo de escucha.  
 accept espera conexiones de clientes.

📌 **Uso:**  
 Usados por servidores para aceptar clientes.

📌 **Ejemplo real:**  
 Un servidor FTP espera conexiones.

c

listen(sockfd, 5);  
int new\_socket = accept(sockfd, NULL, NULL);

# **🔹 11. htons, htonl, ntohs, ntohl**

📌 **Descripción:**  
 Convierte enteros entre **orden de bytes de host** y **orden de red**.

📌 **Uso:**  
 Garantiza compatibilidad en redes.

📌 **Ejemplo real:**  
 Convertir un puerto antes de enviarlo por red.

c

uint16\_t port = htons(8080);

# **🔹 12. inet\_addr, inet\_ntoa**

📌 **Descripción:**  
 inet\_addr convierte una IP en número.  
 inet\_ntoa convierte una IP en cadena.

📌 **Uso:**  
 Conversión de direcciones IP.

📌 **Ejemplo real:**  
 Un cliente convierte "192.168.1.1" a formato numérico.

c

struct in\_addr addr;  
addr.s\_addr = inet\_addr("192.168.1.1");  
printf("IP: %s\n", inet\_ntoa(addr));

# **🔹 13. send, recv**

📌 **Descripción:**  
 send envía datos, recv los recibe.

📌 **Uso:**  
 Comunicación entre cliente y servidor.

📌 **Ejemplo real:**  
 Enviar un mensaje HTTP.

c

send(sockfd, "GET / HTTP/1.1\r\n\r\n", 18, 0);  
recv(sockfd, buffer, 1024, 0);

# **🔹 14. signal, sigaction**

📌 **Descripción:**  
 signal y sigaction manejan señales de Unix, como SIGINT.

📌 **Uso:**  
 Manejar interrupciones (Ctrl+C).

📌 **Ejemplo real:**  
 Interceptar Ctrl+C en un servidor.

void handler(int sig) { printf("Señal recibida\n"); }  
signal(SIGINT, handler);

# **🔹 15. lseek, fstat, fcntl**

📌 **Descripción:**  
 lseek: Mueve el puntero de un archivo.  
 fstat: Obtiene información de un archivo.  
 fcntl: Modifica propiedades de archivos.

📌 **Ejemplo real:**  
 Un servidor HTTP usa fstat para conocer el tamaño de un archivo.

c

struct stat st;  
fstat(fd, &st);  
printf("Tamaño del archivo: %ld bytes\n", st.st\_size);

# **🔹 16. poll**

📌 **Descripción:**  
 Monitorea múltiples sockets o archivos.

📌 **Uso:**  
 Permite manejar múltiples clientes.

📌 **Ejemplo real:**  
 Un servidor de chat maneja múltiples conexiones.

c

struct pollfd fds[1];  
fds[0].fd = sockfd;  
fds[0].events = POLLIN;  
poll(fds, 1, 1000);