Socket Programming

CS-370: Operating System

University of Nevada, Las Vegas

Sockets

- Used for bi-directional communication between programs.

Sockets

- Used for bi-directional communication between programs.
- They dont have to be on the same machine.

Sockets

- Used for bi-directional communication between programs.
- They dont have to be on the same machine.
- Most communication on the Internet is done by sockets.

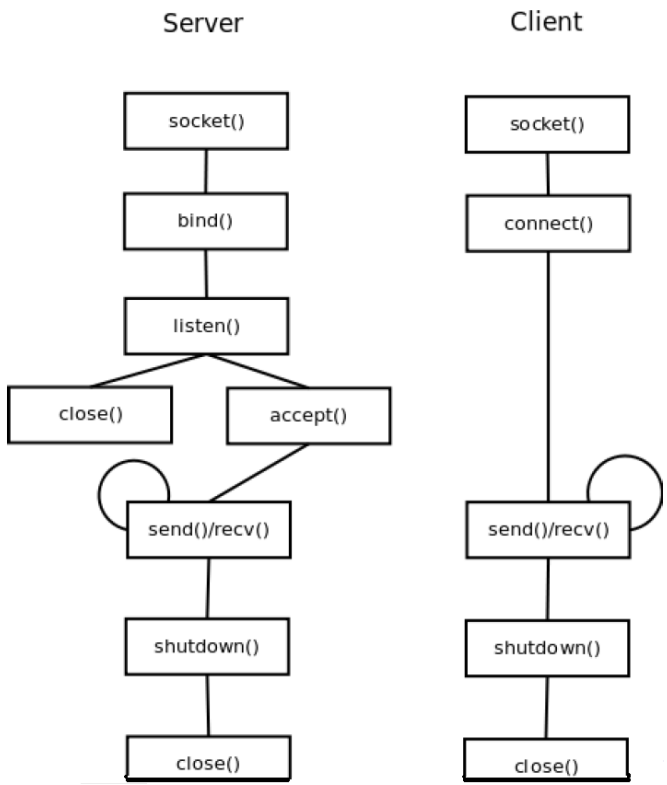
Sockets

- Used for bi-directional communication between programs.
- They dont have to be on the same machine.
- Most communication on the Internet is done by sockets.
- Two types of sockets
  - UDP - Packets
  - TCP Connection Oriented

Unix Sockets

- They allow communication between processes running on the same UNIX system.
- Addressing for the sockets are done with a UNIX path.

Client-Server Connection



socket()

- Purpose: To create endpoint for communication.

socket()

- Purpose: To create endpoint for communication.
- Header Files:
  - sys/socket.h
  - sys/types.h

socket()

- Purpose: To create endpoint for communication.
- Header Files:
  - sys/socket.h
  - sys/types.h
- Prototype: int socket(int domain, int type, int protocol);
- Parameters:
  - Domain: the domain of the socket, we will use AF_UNIX
  - Type: the type of the socket (UDP or TCP), we will be using TCP so use SOCK_STREAM
  - Protocol: use system default 0

socket()

- Purpose: To create endpoint for communication.
- Header Files:
  - sys/socket.h
  - sys/types.h
- Prototype: int socket(int domain, int type, int protocol);
- Parameters:
  - Domain: the domain of the socket, we will use AF_UNIX
  - Type: the type of the socket (UDP or TCP), we will be using TCP so use SOCK_STREAM
  - Protocol: use system default 0
- Return Value: socket descriptor on success and -1 on error

bind()

- Purpose: To assign a local socket address.

bind()

- Purpose: To assign a local socket address.
- Header Files:
  - sys/socket.h

bind()

- Purpose: To assign a local socket address.
- Header Files:
  - sys/socket.h
- Prototype: int bind(int sockfd, const struct sockaddr *my_addr, socklen_t addrlen);
- Parameters:
  - Sockfd: socket descriptor returned by socket()
  - My_addr: address to bind the socket to.
  - Addrlen: the length of the address

bind()

- Purpose: To assign a local socket address.
- Header Files:
  - sys/socket.h
- Prototype: int bind(int sockfd, const struct sockaddr *my_addr, socklen_t addrlen);
- Parameters:
  - Sockfd: socket descriptor returned by socket()
  - My_addr: address to bind the socket to.
  - Addrlen: the length of the address
- Return Value: -1 on error

unlink()

- Purpose: To remove a socket if it exists. Its called before bind()

unlink()

- Purpose: To remove a socket if it exists. Its called before bind()
- Header Files:
  - unistd.h

unlink()

- Purpose: To remove a socket if it exists. Its called before bind()
- Header Files:
  - unistd.h
- Prototype: int unlink(const char *pathname);
- Parameters:
  - Pathname: name/path of the socket to clear.

unlink()

- Purpose: To remove a socket if it exists. Its called before bind()
- Header Files:
  - unistd.h
- Prototype: int unlink(const char *pathname);
- Parameters:
  - Pathname: name/path of the socket to clear.
- Return Value: -1 on error

listen()

- Purpose: To instruct a socket to listen for incoming connections. It specifies the number of pending connections that can be queued for a server socket.

listen()

- Purpose: To instruct a socket to listen for incoming connections. It specifies the number of pending connections that can be queued for a server socket.
- Header Files:
  - sys/socket.h

listen()

- Purpose: To instruct a socket to listen for incoming connections. It specifies the number of pending connections that can be queued for a server socket.
- Header Files:
  - sys/socket.h
- Prototype: int listen(int sockfd, int backlog);
- Parameters:
  - Sockfd: socket descriptor returned by socket()
  - Backlog: the maximum number of connections at one time.

listen()

- Purpose: To instruct a socket to listen for incoming connections. It specifies the number of pending connections that can be queued for a server socket.
- Header Files:
  - sys/socket.h
- Prototype: int listen(int sockfd, int backlog);
- Parameters:
  - Sockfd: socket descriptor returned by socket()
  - Backlog: the maximum number of connections at one time.
- Return Value: -1 on error

accept()

- Purpose: To accept the connection request from a client.

accept()

- Purpose: To accept the connection request from a client.
- Header Files:
  - sys/types.h
  - sys/socket.h

accept()

- Purpose: To accept the connection request from a client.
- Header Files:
  - sys/types.h
  - sys/socket.h
- Prototype: int accept(int sockfd, struct sockaddr *addr, socklen_t *addrlen);
- Parameters:
  - Sockfd: socket descriptor returned by socket()
  - Addr: address of the socket
  - Addrlen: size of the address

accept()

- Purpose: To accept the connection request from a client.
- Header Files:
  - sys/types.h
  - sys/socket.h
- Prototype: int accept(int sockfd, struct sockaddr *addr, socklen_t *addrlen);
- Parameters:
  - Sockfd: socket descriptor returned by socket()
  - Addr: address of the socket
  - Addrlen: size of the address
- Return Value: connected socket descriptor on success and -1 on failure

send()

- Purpose: To send a message via socket

send()

- Purpose: To send a message via socket
- Header Files:
  - sys/socket.h

send()

- Purpose: To send a message via socket
- Header Files:
  - sys/socket.h
- Prototype: ssize_t send(int s, const void *buf, size_t len, int flags);
- Parameters:
  - S: socket descriptor
  - Buf: pointer to data to send
  - Len: size of data to send
  - flags: options for the communication, we will use 0.

send()

- Purpose: To send a message via socket
- Header Files:
  - sys/socket.h
- Prototype: ssize_t send(int s, const void *buf, size_t len, int flags);
- Parameters:
  - S: socket descriptor
  - Buf: pointer to data to send
  - Len: size of data to send
  - flags: options for the communication, we will use 0.
- Return Value: Number of bytes sent on success and -1 on failure

recv()

- Purpose: To receive message via socket.

recv()

- Purpose: To receive message via socket.
- Header Files:
  - sys/socket.h

recv()

- Purpose: To receive message via socket.
- Header Files:
  - sys/socket.h
- Prototype: ssize_t recv(int s, void *buf, size_t len, int flags);
- Parameters:
  - S: socket descriptor
  - Buf: pointer where to store the data
  - Len: size of data to receive
  - flags: options for the communication, we will use 0.

recv()

- Purpose: To receive message via socket.
- Header Files:
  - sys/socket.h
- Prototype: ssize_t recv(int s, void *buf, size_t len, int flags);
- Parameters:
  - S: socket descriptor
  - Buf: pointer where to store the data
  - Len: size of data to receive
  - flags: options for the communication, we will use 0.
- Return Value: Number of bytes read on success and -1 on failure

Creating an Address

```
#include <sys/un.h>
struct sockaddr_un address;
address.sun_family = AF_UNIX;
strcpy(address.sun_path, "mySocket");
unsigned int addressLength = sizeof(address);
```