SOCKETS

COMP750

Distributed Systems
Sockets

The Socket library is a traditional Application Program Interface (API) to the transport layer. Sockets were originally implemented in Unix systems and have been ported to almost every system.

Sockets are a *low level* interface to the network.
Socket Functions

**gethostbyname**  - get the IP address for an IP name
**gethostname**  - get the name of the local machine
**socket**  - create a new socket
**connect**  - make connection to remote host
**send**  - transmit data through active connection
**recv**  - receive data through active connection
**close**  - terminate use of a socket
**bind**  - attach a network address to a socket
**listen**  - wait for incoming messages
**accept**  - begin using incoming connection
Sequence of Socket Procedure Calls
gethostbyname

\[
\text{hostent} = \text{gethostbyname(“IP name”)}
\]

- returns a pointer to a hostent structure.
- provides the IP address for an IP name.
gethostname

gethostname( hostname, buffersize )

• puts the IP name of the local computer in the hostname string.
socket

descriptor = socket(protofamily, type, protocol)

- Returns socket descriptor used in subsequent calls
- protofamily selects protocol family; (e.g. PF_INET - Internet protocols, PF_APPLETALK - AppleTalk protocols)
- type selects type of communication, SOCK_DGRAM - connectionless, SOCK_STREAM - connection-oriented
- protocol specifies protocol within protocol family: IPPROTO_TCP - selects TCP, IPPROTO_UDP - selects UDP
**connect**

`connect(socket, saddress, saddresslen)`

- Client uses `connect` to establish connection to server
- Blocks until connection completed (`accepted`)
- `socket` holds descriptor of socket to use
- `saddress` is a `struct sockaddr` that identifies server
- `saddresslen` gives length of `saddress`
- Usually used with connection-oriented transport protocol
- Can be used with connectionless protocol
- Marks local socket with server address
- Implicitly identifies server for subsequent messages
send

send(socket, data, length, flags)

- Used to send data through a connected socket
- socket identifies socket
- data points to data to be sent
- length gives length of data (in bytes)
- flags indicate special options
sendto

sendto(socket, data, length, flags, destaddr, addrlen)

- Used for *unconnected* sockets by explicitly specifying destination
- sendto adds additional parameters:
  - destaddr - struct sockaddr destination address
  - addrlen - length of destaddr
recv

recv(socket, buffer, length, flags)

- Used to receive incoming data through connected socket
- socket identifies the socket
- Data copied into buffer
- At most length bytes will be received
- flags give special options
- Returns number of bytes actually received
  - 0 implies connection closed
  - -1 implies error
recvfrom

recvfrom(socket, buffer, length, flags, sndraddr, addrlen)

• Like recvfrom (in reverse!)
• Address of source copied into sndraddr
• Length of address in addrlen
close

close(descriptors)

- Terminates use of socket descriptor
- descriptor contains descriptor of socket to be closed
bind

bind(socket, localaddr, address)

- Initially, socket has no addresses attached
- `bind` selects either local, remote or both addresses
- `server` binds local port number for incoming messages
- `client` binds remote address and port number to contact server
listen

listen(socket, queuesize)

- Server uses `listen` to wait for incoming connections
- `socket` identifies socket through which connections will arrive (address)
- New connection requests may arrive while server processes previous request
- Operating system can hold requests on queue
- `queuesize` sets upper limit on outstanding requests
accept

accept(socket, caddress, caddrlen)

- Server uses `accept` to accept the next connection request
- `accept` call blocks until connection request arrives
- Returns new socket with server's end of new connection
- *Old socket* remains unchanged and continues to field incoming requests
- `caddress` returns `struct sockaddr` client address; format depends on address family of socket
- `caddrlen` returns length of address
Socket Address Format

struct sockaddr_in {
    u_char sin_len;     /* total length of address */
    u_char sin_family;  /* family of the address */
    u_short sin_port;   /* protocol port number */
    struct in_addr sin_addr; /* IP address */
    char sin_zero[8]    /* unused */
}

Host entry structure returned by gethostbyname

struct hostent {
    char *h_name;        /* official name of host */
    char **h_aliases;    /* alias list */
    int h_addrtype;      /* host address type */
    int h_length;        /* length of address */
    char **h_addr_list;  /* list of addresses from name server */
};

#define h_addr h_addr_list[0] /* address, for backward compatibility */
Format Conversion

\[ \text{int} = \text{htons}( \text{short} ) \]

- **Host TO Network Short**
- Converts a short int to network standard format.
- Swaps bytes if necessary.

\[ \text{int} = \text{ntohs}( \text{short}) \]

- **Network TO Host Short**
- Converts a 16 bit integer in network standard format to a short in the local host format.
- Swaps bytes if necessary.
Microsoft Extensions

WSAStartup( version, &wsaData);

• Must be called before any other socket function.
• WSADATA is an structure that will receive information about the socket implementation on this system, such as the implementation version.

WSACleanup();

• Closes all sockets.
• WSAStartup must be called to use any socket function again.
Java Sockets

- There are two socket classes in Java
  - java.net.Socket for clients
  - java.net.ServerSocket for servers

- Sockets create streams that can be used exactly like file streams.
- Reading and writing to a socket is identical to reading and writing to a file.
Socket Constructor

public Socket(String host, int port)
    throws UnknownHostException,
    IOException

• Creates a stream socket and connects it to the specified port number on the named host.
• Other constructors are available.
Methods

public InputStream getInputStream()  
    throws IOException  
Returns an input stream for this socket.

public OutputStream getOutputStream()  
    throws IOException  
Returns an output stream for this socket.
java.net.ServerSocket

• Creates a TCP socket for use by a server program.
• A server socket waits for requests to come in over the network.
constructor

public ServerSocket(int port)
    throws IOException

• Creates a server socket on a specified port.
• A port of 0 creates a socket on any free port.
ServerSocket Methods

public Socket accept() throws IOException

• Listens for a connection to be made to this socket and accepts it.
• The method blocks until a connection is made.
• A new Socket is created.
Java Security

- Java applets running in a browser can only connect to the server that hosts the applet.
- Java applications generally have no restrictions.
- Experience shows that network programming is easiest on a PC with Java.