



Socket Programming

Tutorial

Michael Kounavis

Dept. of Electrical Engineering

Columbia University

<http://comet.columbia.edu/~mk>

mk@comet.columbia.edu

15 September, 1998

1



What are Sockets?

- Communication abstractions
- Support the TCP/IP protocol stack
- Provide access to both reliable (TCP) and unreliable (UDP) transport services
- Programming tools to implement client-server applications

2



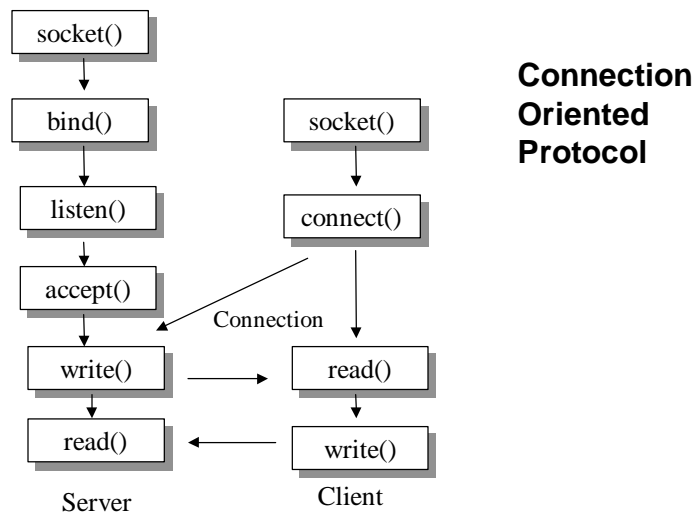
IP addresses and Ports

- IP address
 - >> A four byte number. It identifies and *Internet host*
eg. 128.59.69.3
- Port number
 - >> A two byte short. It identifies a *client or a server process*

3

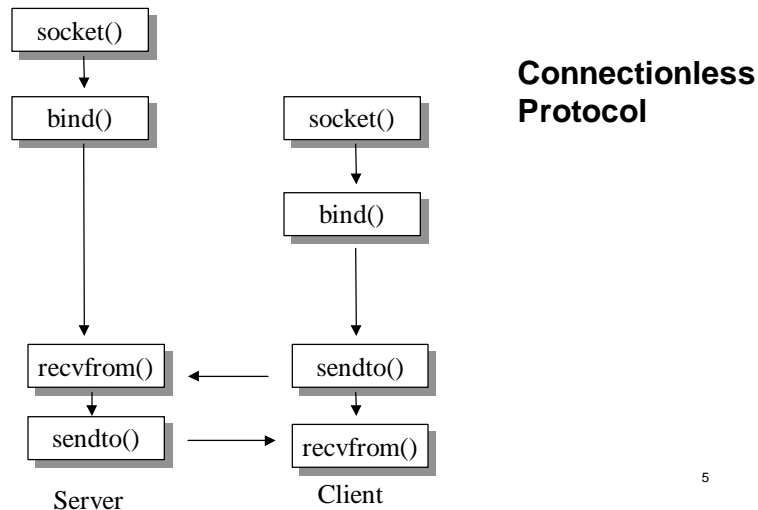


Summary of Function Calls (I)





Summary of Function Calls (II)



The structures *sockaddr* and *sockaddr_in*

- >> Contain address information associated with a socket
- >> They have the same length in bytes
- >> *sockaddr_in* applies to Internet protocols (contains IP address and port)

```
struct sockaddr {
    u_short sa_family;
    u_short sa_data[14];
}
```

```
struct sockaddr_in {
    short sin_family;
    u_short sin_port;
    struct in_addr sin_addr;
    char sin_zero[8]; // unused
}
```



Filling the sockaddr_in structure (I)

Example I:

```
sock_addr.sin_addr.s_addr = htonl(INADDR_ANY);  
sock_addr.sin_port = htons(0);  
sock_addr.sin_family = AF_INET;
```

Example II:

```
sock_addr.sin_addr.s_addr = inet_addr("128.59.69.7");  
sock_addr.sin_port = htons(5115);  
sock_addr.sin_family = AF_INET
```

7



Filling the sockaddr_in structure (II)

- INADDR_ANY is used for accepting connections on any local interface if the system is multi-homed
- We assign a zero port number when filling the sockaddr_in data structure, if we want a free port to be allocated to a socket during the bind() procedure
- htonl(), htons() handle byte-ordering differences between computer architectures and network protocols
 - >> *redundant in Sun's sparc*
 - >> *needed in x86 architectures*

8



The socket() function

```
int socket(int family, int type, int protocol)
```

Example:

```
if ((sock = socket(AF_INET, SOCK_STREAM, 0)) == -1)
{ printf(": error opening socket");
  exit(-1);
}
```

- Creates a socket: Not associated with a client or server process yet
- SOCK_STREAM vs. SOCK_DGRAM



The bind() function

```
int bind(int sockfd, struct sockaddr *myaddr, int addrlen)
```

Example:

```
if ( bind(sock, (struct sockaddr *) &sock_addr, sizeof(sock_addr)) < 0)
{ printf(": error binding socket to local address");
  exit(-1);
}
```

- Binds a socket to an IP address and port.



The connect() function

int connect (int *sockfd*, struct sockaddr **myaddr*, int *addrlen*)

Example:

```
if (connect(sock, ( struct sockaddr *) &sock_addr, sizeof sock_addr) == -1)
    { printf(": socket connection error");
      exit(-1);
    }
```

- used in connection oriented communications
- Connects a client to a server

11



The read() and write() functions

int read(int *sockfd*, char **buf*, unsigned int *nbytes*)
int write(int *sockfd*, char **buf*, unsigned int *nbytes*)

Examples:

```
if (write(sock, message, size) < 0 )
    { printf(": error writing to remote host");
      exit(-1);
    }
```

```
bytes_received = read(sock, reply, sizeof(reply));
```

- used in connection oriented communications
- used for requesting and transmitting data

12



The sendto() and recvfrom() functions

```
int recvfrom(int sockfd, char *buf, int nbytes, int flags,  
            struct sockaddr *from, int addrlen)
```

```
int sendto(int sockfd, char *buf, int nbytes, int flags,  
          struct sockaddr *to, int addrlen)
```

- used in connectionless communications
- peer end-system address information is passed or returned from these function calls (why?)

13



What do I need to include?

- In Unix:

```
#include <sys/types.h>  
#include <sys/socket.h>  
#include <netdb.h>  
#include <netinet/in.h>
```

- In Windows NT/95

```
#include <windows.h>  
#include <sys/types.h>  
#include <winsock.h>
```

14



References

Required Reading

- W.R.Stevens "Unix Network Programming", 1st edition, Prentice Hall, *Chapter 6 "Berkeley Sockets" p. 259-298*, or any equivalent chapter from other textbooks

Recommended Reading:

- W.R.Stevens and G.R.Wright "TCP/IP Illustrated", Vol II "The Implementation", Addison Wesley