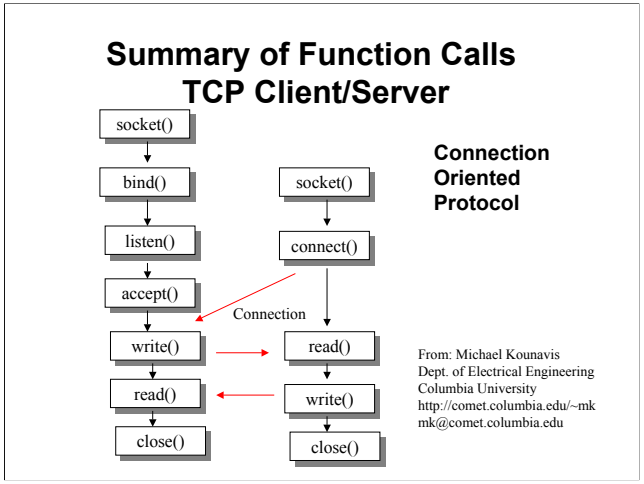
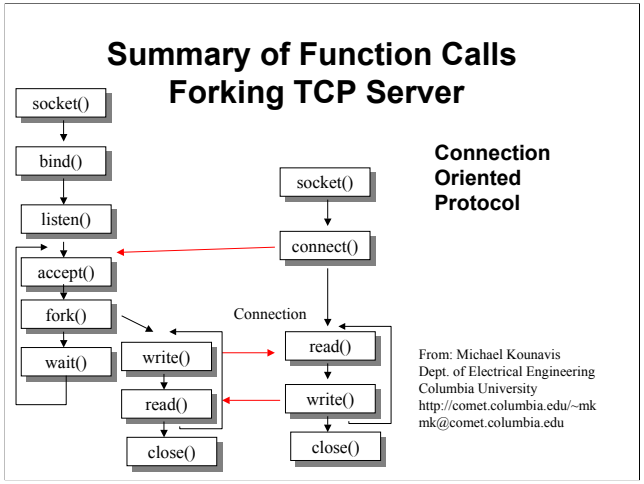
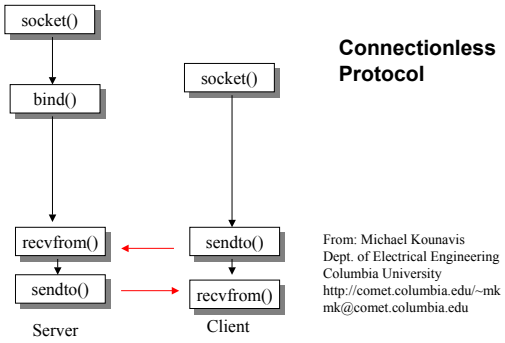


Note use of write() and read() instead of send() and recv(): either is fine; send and recv have additional flag parameter providing additional networking-related control. Another related pair of system calls is sendmsg and recvmsg which allow "scatter/gather" distribution/collection of message data.



Note use of wait(). Wait is not a networking-related system call; rather it is used for process management. Unix requires that the parent of any fork'ed process "reap" the child's status before the child can completely disappear from the system. This is done with the wait() system call. Thus the parent server process must periodically wait for its children: appropriate choice of wait() system call and parameters can let it check for a completed child process without actually blocking.

Summary of Function Calls UDP Client/Server



Note use of bind in client.

Also note use of recvfrom and sendto in place of recv and send or read and write. These calls have extra parameters that either indicate to the kernel the destination of the messsage (sendto) or allow the application to determine the source of the message (recvfrom).

Additional useful functions

- `unsigned long int inet_addr(char *s)`
- `struct hostent *gethostbyname(const char *name)`
- `struct hostent *gethostbyaddr(const char *addr, int len, int type)`
- `int getsockname(int s, struct sockaddr *name, int *namelen)`

Note the last parameter to getsockname is int *namelen, not int namelen. Why?

struct hostent

The hostent structure is defined in <netdb.h> as follows:

```
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 */
}
#define h_addr  h_addr_list[0]  /* for backward compatibility */
```

What's going on with **h_addr_list and h_addr?

Review

- Concepts
  - host address, port number
  - socket
  - TCP client and server patterns
  - UDP client and server patterns
  - Boundary-less TCP data streams
- Contrast
  - host address vs host name
  - socket vs host address and port
  - TCP vs UDP usage patterns