Introduction to Networking

Instructor: Prof. Aleksandar Kuzmanovic

Advanced Sockets

The problem

You are a server and you want to listen for incoming connections as well as keep reading from the connections you already have.

- Blocking
  - "block" is techie jargon for "sleep".
- Lots of functions block.
  - accept() blocks.
  - All the recv() functions block

The select call

Enables you to deal with many clients at the same time

HOW ?

Monitors several sockets at the same time.
tell you which ones are ready for reading,
which are ready for writing,

Synopsis of select()

- #include <sys/time.h>
- #include <sys/types.h>
- #include <unistd.h>
- int select(int numfds, fd_set *readfds, fd_set *writefds, fd_set *exceptfds, struct timeval *timeout);
- It'll tell you which ones are ready for reading, which are ready for writing, and which sockets have raised exceptions, if you really want to know that.

Manipulate sets

- Each set is of the type fd_set. The following macros operate on this type:
- FD_ZERO(fd_set *set) -- clears a file descriptor set
- FD_SET(int fd, fd_set *set) -- adds *fd* to the set
- FD_CLR(int fd, fd_set *set) -- removes *fd* from the set
- FD_ISSET(int fd, fd_set *set) -- tests to see if *fd* is in the set
- Example - when select() returns, *readfds* will be modified to reflect which of the file descriptors you selected is ready for reading. Test them with the macro FD_ISSET(),

THE CODE

A multi-person chat server

DEMO

A multi-person chat server

Important Points

- Accepting new connections via select

- Client closes connection
  - Select returns “socket ready to read”
  - recv() will return 0.