



Week 11 - Friday

Network Programming (cont.)

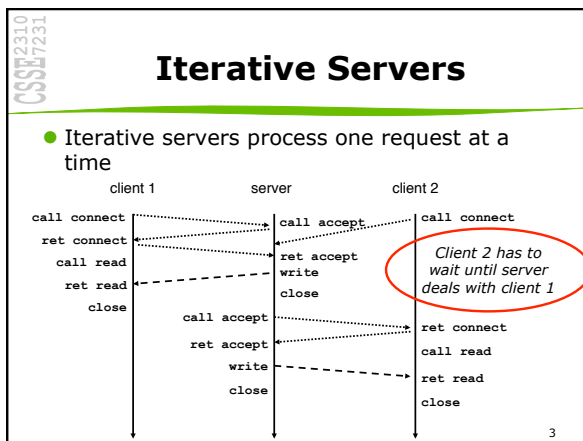
School of Information Technology and Electrical Engineering
The University of Queensland



Outline

- C Network Programming
 - Concurrent servers – single process (`select()` function)
 - UDP Example
- Credits:
 - Glass and Ables, "UNIX for Programmers and Users"
 - Bryant and O'Halloran, "Computer Systems: A Programmer's Perspective"
 - Rochkind, "Advanced UNIX Programming"
 - Tanenbaum, "Computer Networks"

2





3 Basic Mechanisms for Creating Concurrent Flows

1. Processes
 - Kernel automatically interleaves multiple logical flows
 - Each flow has its own private address space
2. Threads
 - Kernel automatically interleaves multiple logical flows
 - Each flow shares the same address space
3. I/O multiplexing with `select()`
 - User manually interleaves multiple logical flows
 - Each flow shares the same address space
 - Popular for high-performance server designs

4



Event-Based Concurrent Servers Using I/O Multiplexing

- Maintain a set of connected descriptors and service each as new data arrives
- Repeat the following forever:
 - Use the Unix `select()` function to block until:
 - (a) New connection request arrives on the listening descriptor, or
 - (b) New data arrives on an existing connected descriptor
 - If (a), add the new connection to the pool of connections
 - If (b), read any available data from the connection
 - Close connection on EOF and remove it from the set

5



The `select()` Function

- `select()` sleeps until one or more file descriptors in the set `readset` are ready for reading

```
#include <sys/select.h>

int select(int maxfdp1, fd_set *readset, NULL, NULL, NULL);
```

- `readset`
 - Opaque bit vector (max `FD_SETSIZE` bits) that indicates membership in a *descriptor set*
 - If bit `k` is 1, descriptor `k` is a member of the descriptor set
- `maxfdp1`
 - Maximum descriptor in descriptor set plus 1
 - Tests descriptors 0, 1, 2, ..., `maxfdp1 - 1` for set membership
- `select()` returns number of ready descriptors and sets each bit of `readset` to indicate the ready status of corresponding descriptor

6

CSSE2310
CSSE7231

Macros for Manipulating Set Descriptors

- `void FD_ZERO(fd_set *fdset);`
 - Turn off all bits in `fdset`
- `void FD_SET(int fd, fd_set *fdset);`
 - Turn on bit `fd` in `fdset`
- `void FD_CLR(int fd, fd_set *fdset);`
 - Turn off bit `fd` in `fdset`
- `int FD_ISSET(int fd, *fdset);`
 - Is bit `fd` in `fdset` turned on?

7

CSSE2310
CSSE7231

Sample `select()` Server Code

- *To be discussed in class*

8

CSSE2310
CSSE7231

Other options

- `poll()`
 - Does the same thing as `select`, but different interface
- `pselect()`, `ppoll()`
 - Can wait for a signal also

9

CSSE2310
CSSE7231

UDP Example Code

- *To be discussed in class*

11

CSSE2310
CSSE7231

Pro and Cons of Event-Based Designs

- + One logical control flow
- + Can single-step with a debugger
- + No process or thread control overhead
- More complex to code than process or thread-based designs
- Can be vulnerable to denial of service attack
 - How?

12

CSSE2310
CSSE7231

Resources

- Beej's Guide to Network Programming
 - <http://beej.us/guide/bgnet/> (section 7.2)
- Manual pages
 - On `moss`: `man <name>` or `man -s 2 <name>`
 - where `<name>` is `socket`, `bind`, `connect`, `listen`, `accept`, `recv`, `send`, ...
 - Try `man select_tut`
- Glass & Ables, "UNIX for Programmers and Users"
- Rochkind, "Advanced UNIX Programming"
- Bryant and O'Halloran, "Computer Systems: A Programmer's Perspective"
- Other UNIX Programming books...
 - See Reference text list in course profile

13