Week 10 - Friday

Network Programming (cont.)

School of Information Technology and Electrical Engineering
The University of Queensland

Outline

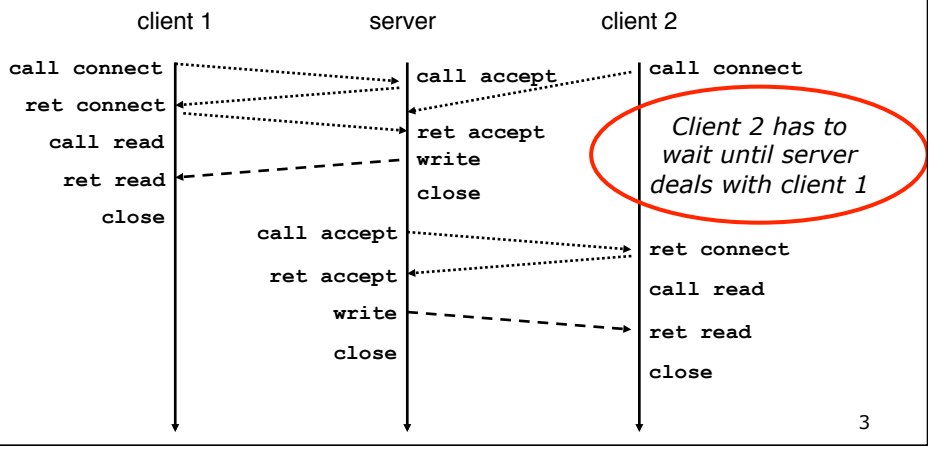
- C Network Programming
  - Concurrent servers – multi process
  - Concurrent servers – multi thread
- Credits:
  - Glass and Ables, "UNIX for Programmers and Users"
  - Bryant and O'Halloran, "Computer Systems: A Programmer's Perspective"
  - Rochkind, "Advanced UNIX Programming"
  - Tanenbaum, "Computer Networks"

2

CSSE2310 7231

Iterative Servers

- Iterative servers process one request at a time



3

CSSE2310 7231

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

CSSE2310 7231

Process-Based Concurrent Server

```
...
  /* main server loop */
  while (1) {
    connfd = accept(listenfd, (struct sockaddr *) &clientaddr,
                    &clientlen));
    if (fork() == 0) {
      close(listenfd); /* child closes its listening socket */
      ... /* Read data from connfd, process, write data etc. */
      close(connfd);   /* child is done with this client */
      exit(0);         /* child exits */
    }
    close(connfd); /* parent must close connected socket! */
  }
...
```

5

CSSE2310 7231

Process-Based Concurrent Server (cont.)

```
... /* parent must reap children! - set up handler */
struct sigaction sa;
sa.sa_handler = reapchildren;
sa.sa_flags = SA_RESTART | SA_NOCLDSTOP;
sigaction(SIGCHLD, &sa, NULL);...

/* signal handler - reaps children as they terminate */
void reapchildren(int sig) {
  pid_t pid;
  int stat;

  while ((pid = waitpid(-1, &stat, WNOHANG)) > 0) {
    ;
  }
  return;
}
```

- Alternatively, set handler to SIG_IGN (recent OS's)

6

CSSE2310 7231

Process-based Concurrent Server Example

- Tuesday's server example to be extended in class

7

CSSE2310 7231

Implementation Issues With Process-Based Designs

- Server should restart `accept` call if it is interrupted by a transfer of control to the SIGCHLD handler
  - Not necessary for systems with POSIX signal handling and restart flag is specified
  - Required for portability on some older Unix systems.
- Server must reap zombie children
  - to avoid running out of processes
- Server must `close` its copy of `connfd`
  - Kernel keeps reference count for each socket
  - After fork, `refcnt(connfd) = 2`
  - Connection will not be closed until `refcnt(connfd)=0`

8

CSSE2310 7231

Thread-Based Concurrent Server

```
...
    /* main server loop */
    while (1) {
        connfd = accept(listenfd, (struct sockaddr *) &clientaddr,
                &clientlen));
        pthread_create(&threadID, NULL, client_thread, (void*)fd);
        pthread_detach(threadID);
    }
...
void* client_thread(void* arg)
{
    int fd;
    fd = (int)arg;     /* Retrieve file descriptor from argument */

    ... /* Read data, process, write data etc. */

    close(fd);
    pthread_exit(NULL);
    return NULL;
}
```

CSSE2310 7231

Thread-based Concurrent Server Example

- Tuesday's server example to be extended in class

11

CSSE2310 7231

Pros and Cons of Process Based Designs

+ Handles multiple connections concurrently
+ Simple and straightforward
– Additional overhead for process control
– Nontrivial to share data between processes
  - Requires interprocess communication (IPC) mechanisms
    - FIFO's (named pipes), shared memory and semaphores

Thread based designs

+ Easier to share data between threads (though may need mutexes/semaphores)
– Do have thread overhead

12

CSSE2310 7231

Third option...

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

- Event-based design
- More on this next week (Friday)

13

CSSE2310 7231

Resources

- Beej's Guide to Network Programming
  - http://beej.us/guide/bgnet/
- Manual pages
  - On moss: `man <name>` or `man -s 2 <name>`
  - where `<name>` is socket, bind, connect, listen, accept, recv, send, …
- 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

14