Advanced Programming in the UNIX Environment

Week 12, Segment 2:
Non-blocking I/O

Department of Computer Science
Stevens Institute of Technology

Jan Schaumann
jschauma@stevens.edu
https://stevens.netmeister.org/631/

Blocking functions

Some system calls can block for long periods of time (or forever). These include things like:

- `read(2)` from files that can block (pipes, networks, terminals)
- `write(2)` to the same sort of files
- `open(2)` of a device that waits until a condition occurs (for example, a modem)
- `pause(3)`, which purposefully puts a process to sleep until a signal occurs
- certain `ioctl(2)`s
- certain IPC functions
- file- or record-locking mechanisms

Non-blocking I/O

Non-blocking I/O lets us issue an I/O operation and not have it block forever. If the operation cannot be completed, return is made immediately with an error noting that the operating would have blocked (`EWOULDBLOCK` or `EAGAIN`).

Ways to specify nonblocking mode:

- pass `O_NONBLOCK` to `open(2)`:

```
open(path, O_RDRW | O_NONBLOCK);
```

- set `O_NONBLOCK` via `fcntl(2)`:

```
flags = fcntl(fd, F_GETFL, 0);
fcntl(fd, F_SETFL, flags | O_NONBLOCK);
```

```
        (void)argv;

        /* fill buffer with 'a' */
        memset(buf, 'a', BUFSIZE);

        if ((flags = fcntl(STDOUT_FILENO, F_GETFL, 0)) < 0) {
                perror("getting file flags");
                exit(EXIT_FAILURE);
                /* NOTREACHED */
        }

        if (argc > 1) {
                /* set non-blocking mode on stdout */
                if (fcntl(STDOUT_FILENO, F_SETFL, flags|O_NONBLOCK) < 0) {
                        perror("setting file flags");
                        exit(EXIT_FAILURE);
                        /* NOTREACHED */
                }
        }

        for (loops = 0; loops < 50; loops++) {
                ptr = buf;
                num = BUFSIZE;
                                                        23,9    607     37%
```

Non-blocking I/O

- Revisit Week 07, Segment 5 on interrupted syscalls
- Can you write another consumer that periodically blocks?
- Can you tune the network settings to adjust the relevant TCP buffers?
- Try to send data to a system on a different network and observe what, if any, delay you find.
- Observe the output of the sample program to a file on a network file system.