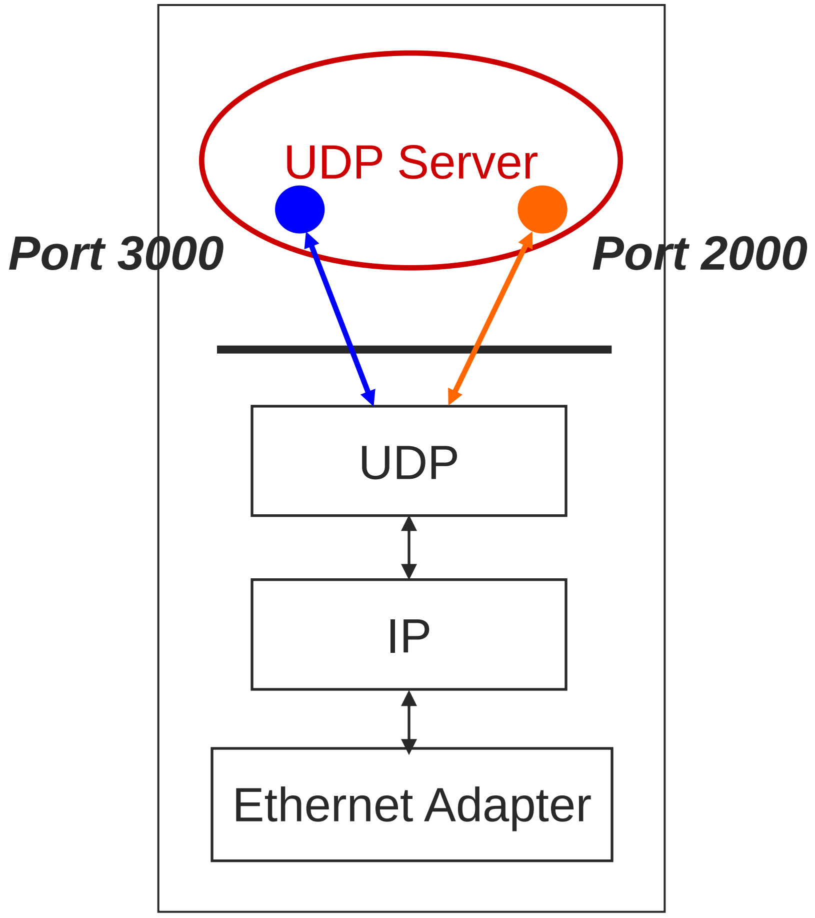
More Network Programming

More Network Programming

- Advanced uses of sockets
  - How to handle multiple sockets
  - How to create timers
  - How to survive abrupt channel closure
  - What if `bind()` says “Address already in use” ?

A UDP Server



- How can a UDP server service multiple ports simultaneously?

UDP Server: Servicing Two Ports

```
int s1;                          /* socket descriptor 1 */
int s2;                          /* socket descriptor 2 */

/* 1) create socket s1 */
/* 2) create socket s2 */
/* 3) bind s1 to port 2000 */
/* 4) bind s2 to port 3000 */

while(1) {
    recvfrom(s1, buf, sizeof(buf), ...);
    /* process buf */
    recvfrom(s2, buf, sizeof(buf), ...);
    /* process buf */
}
```

What problems does this code have?

Building Timeouts with Select and Poll

- Time structure

Number of seconds since
midnight, January 1, 1970 GMT

```
struct timeval {
   long tv_sec;           /* seconds */
   long tv_usec;  /* microseconds */
};
```

unix will have its own "Y2K" problem one second after 10:14:07pm, Monday January 18, 2038 (will appear to be 3:45:52pm, Friday December 13, 1901)

Select

Which file descriptors are set and what should the timeout value be?

- High-resolution sleep function
  - All descriptor sets **`NULL`**
  - Positive **`timeout`**
- Wait until descriptor(s) become ready
  - At least one descriptor in set
  - **`timeout`** **`NULL`**
- Wait until descriptor(s) become ready or timeout occurs
  - At least one descriptor in set
  - Positive **`timeout`**
- Check descriptors immediately (poll)
  - At least one descriptor in set
  - 0 **`timeout`**

Select: Example

```
fd_set my_read;
FD_ZERO(&my_read);
FD_SET(0, &my_read);

if (select(1, &my_read, NULL, NULL) == 1) {
   ASSERT(FD_ISSET(0, &my_read);
   /* data ready on stdin */
```

What went wrong: after select indicates data available on a connection, read returns no data?

Select: Timeout Example

```
int main(void) {
    struct timeval tv;
    fd_set readfds;

    tv.tv_sec = 2;
    tv.tv_usec = 500000;

    FD_ZERO(&readfds);
    FD_SET(STDIN, &readfds);

    // don't care about writefds and exceptfds:
    select(1, &readfds, NULL, NULL, &tv);

    if (FD_ISSET(STDIN, &readfds))
        printf("A key was pressed!\n");
    else
        printf("Timed out.\n");

    return 0;
}
```

Wait 2.5 seconds for something to appear on standard input

Poll

- High-resolution sleep function
  - 0 `nfds`
  - Positive `timeout`
- Wait until descriptor(s) become ready
  - `nfds` > 0
  - `timeout` `INFTIM` or -1
- Wait until descriptor(s) become ready or timeout occurs
  - `nfds` > 0
  - Positive `timeout`
- Check descriptors immediately (poll)
  - `nfds` > 0
  - 0 `timeout`

`select()` vs. `poll()`

Which to use?

- BSD-family (e.g., FreeBSD, MacOS)
  - **`poll()`** just calls **`select()`** internally
- System V family (e.g., AT&T Unix)
  - **`select()`** just calls **`poll()`** internally

Advanced Sockets: `signal`

- Problem: Socket at other end is closed
  - Write to your end generates **`SIGPIPE`**
  - This signal kills the program by default!

```
signal (SIGPIPE, SIG_IGN);
```

  - Call at start of main in server
  - Allows you to ignore broken pipe signals
  - Can ignore or install a proper signal handler
  - Default handler exits (terminates process)

Advanced Sockets

- Problem: How come I get "address already in use" from **`bind()`**?
  - You have stopped your server, and then re-started it right away
  - The sockets that were used by the first incarnation of the server are still active

Advanced Sockets: `setsockopt`

```
int yes = 1;
setsockopt (fd, SOL_SOCKET,
   SO_REUSEADDR, (char *) &yes, sizeof
   (yes));
```

- Call just before `bind()`
- Allows bind to succeed despite the existence of existing connections in the requested TCP port
- Connections in limbo (e.g. lost final ACK) will cause bind to fail