

Socket Programming II

➤ Dealing with blocking and timeouts

▪ `select()` function

➤ Time server and client example

ee122 sp05
daron spektor

Time Client

1. Ask time server for current time.
2. Print response
3. Repeat

```
while (1) {
    sendto(sock, "send", 5, 0, (struct sockaddr *)&server,
            sizeof(struct sockaddr));
    recvfrom(sock, buf, 4, 0, (struct sockaddr *)&sender,
             &rlen);
    printf("the time is %d\n", word2int(buf));
}
```

But what if one of the packets gets dropped?

We want to resend the query.

Time Client

1. Ask time server for current time.
2. If no response after T seconds, ask again.
3. Print response
4. Repeat

Problem: `recvfrom()` blocks:

It will not return until there is a response.

```
while (1) {
    sendto(sock, "send", 5, 0, (struct sockaddr *)&server,
            sizeof(struct sockaddr));
    recvfrom(sock, buf, 4, 0, (struct sockaddr *)&sender,
             &rlen);
    printf("the time is %d\n", word2int(buf));
}
```

`select()` function

```
int select(int numfds, fd_set *readfds, fd_set *writefds,
           fd_set *exceptfds, struct timeval *timeout);
```

- monitors a set of file descriptors (in our case, sockets)
- returns when
 - one of them is ready to be read OR
 - the specified timeout has passed

```
fd_set readfds; // declare a read set
struct timeval timeout; // declare a timeval for our timer
FD_ZERO(&readfds); // zero out the read set
FD_SET(sock, &readfds) // add socket to the read set
timeout.tv_sec = 3; // timeout = 3 seconds
timeout.tv_usec = 500000; // timeout += 0.5 seconds
select(sock+1, &readfds, NULL, NULL, &timeout);
if (FD_ISSET(sock, &readfds) {
    /** do recvfrom stuff **/
} else {
    /** do timeout stuff **/
}
```

Time Client (see `timeclient.c`)

```
struct timeval timer;
fd_set readfds, master;
FD_ZERO(&readfds);
FD_ZERO(&master);
FD_SET(sock, &master);
sendto(sock, "send", 5, 0, (struct sockaddr *)&server,
        sizeof(struct sockaddr));
while (1) {
    timer.tv_sec=2;
    timer.tv_usec=0;
    readfds=master;
    select(sock+1, &readfds, NULL, NULL, &timer);
    if (FD_ISSET(sock, &readfds) {
        recvfrom(sock, buf, 4, 0, (struct sockaddr *)&sender, &rlen);
        printf("the time is %d\n", word2int(buf));
        FD_CLR(sock, &readfds);
    } else {
        printf("timeout!\n");
        sendto(sock, "send", 5, 0, (struct sockaddr *)&server,
                sizeof(struct sockaddr));
    }
}
```

Time Server (see `timeserver.c`)

```
struct timeval tv;
while (1) {
    recvfrom(sock, buf, 5, 0, (struct sockaddr *)&sender, &rlen);
    gettimeofday(&tv, 0);
    int2word(msg, tv.tv_sec);
    random=rand()%5;
    if (random==2) {
        sendto(sock, msg, 4, 0, (struct sockaddr *)&client,
                sizeof(struct sockaddr));
    }
}
```

Does this server need to worry about timeouts and use `select()`?