

## Re: Socket Server... Why two?

**Source:** <http://unix.derkeiler.com/Newsgroups/comp.unix.programmer/2004-12/0425.html>

---

**From:** dragoncoder (*pktiwary\_at\_gmail.com*)

**Date:** 12/14/04

Date: 13 Dec 2004 15:10:50 -0800

>> *accept() -> select() -> recv() --- in a blocking way ---*

I hope you meant non-blocking way

>>*Both wait for data to be received before to unblock.*

Not exactly. The `accept` just accepts a connection and returns as soon as the connection is established. There is no data transfer till now. It is just an information to the application that now one more socket can send data so, be prepared to read from this socket also.

>>*But what's the advantage and disadvantage in the first and in second solution?*

When the connections have been established, any socket can data for the application. If you have just one socket descriptor and you have nothing to do without that data, you can do a blocking `recv()`. But if you have more than one descriptors through which data can come and you can proceed when any of them is available with the data, the non-blocking `recv()` is done.

The advantage of non-blocking `recv()` is that you can proceed with the job as soon as there are any data available on any of the descriptors. Consider an example of a chat server. There can be thousands of users connected at the same time and data can come from any user for any other user. So, you can not do a blocking `recv()` because this will cause other users to wait until the `recv()` is complete. I don't know of any disadvantages of this approach.

Remember, `select` does not read any data. It just notifies the application that xyz socket is available for reading. After that you can call `read()` or `recv()` accordingly.