**Assignment 1: TCP Client-Server to Print and Manipulate Socket Options**

For this assignment, I decided to take some risk and covert the TCP client and server code from C to C++ while removing the dependency to unp.h header file. The intent was to use readily available, modern networking libraries.

**Why C++11?**

A small aside but worth mentioning. Why use c++11 compiler? C++11 now supports:

* lambda expressions,
* automatic type deduction of objects,
* uniform initialization syntax,
* delegating constructors,
* deleted and defaulted function declarations,
* nullptr,
* rvalue references

"The C++11 Standard Library was also revamped with new algorithms, new container classes, atomic operations, type traits, regular expressions, new smart pointers, async() facility, and of course a multithreading library"

When setting socket options to new values, we must be mindful. SO\_SNDBUF and SO\_RCVBUF have an upper limit, which if exceeded will ignore the value we are trying to use. Otherwise, if we set it to 50,000 for example, the value displayed will be double – 100,000. Likewise, if we set it to 40,000 the value displayed will be 80,000.

Please also note that some options may not be changed – either the protocol is not available or the server restricts access and modification fails with “Permission denied.” This occurs for SO\_DEBUG, SO\_TYPE, SO\_SNDLOWAT, and TCP\_MAXSEG.

TCP\_MAXSEG: “The maximum segment size for outgoing TCP packets. In Linux 2.2 and earlier, and in Linux 2.6.28 and later, if this option is set before connection establishment, it also changes the MSS value announced to the other end in the initial packet. Values greater than the (eventual) interface MTU have no effect. TCP will also impose its minimum and maximum bounds over the value provided.”

<https://linux.die.net/man/7/tcp>