

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

### Table of Contents

Introduction .....	2
Why C++11? .....	2
Notes on Limitations and Design Constraints .....	2
Results and Program Execution .....	3
References .....	8
Figure 1: Build the project and start the server on Eros. ....	4
Figure 2: Run the client on Zeus. ....	5
Figure 3: Server on Eros processes client payload sent from Zeus. ....	6
Figure 4: New server socket opts after processing. Binary flags are inverted, numeric types are changed, timeval is set to sec = 1 and usec = 50000, Linger is kept intentionally the same. ....	7
Figure 5: Shutting down the server with Ctrl + C. Also Ctrl + Z may be used, or the Linux kill command. ....	7

## Introduction

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 the `unp.h` header file. The intent was to use readily available, modern networking libraries. Initially I thought to leverage classes and use an object-oriented design approach. I stuck to functional programming as there is no need to instantiate classes or deal with object reuse; the program is rather straightforward and short.

Source code for this program may be found on GitHub here:

<https://github.com/bss8/tcp-socket-options>

## 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."

## Notes on Limitations and Design Constraints

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>

## Results and Program Execution

The program is correct and performs the specified functionality. The client sends a string payload to the server, containing server options and values. It is in the form: opt\_str,opt\_name,opt\_val

Where opt\_str is the string representation of the socket option, opt\_name is an integer representation of the socket option, and opt\_val is the value. In the case of non-binary types, the value may consist of two items.

Binary values are inverted. If it is off on the client, it is set to on in the server and vice versa. For numeric types, a small value less than 10,000 is simply doubled. Larger values are set to 30,000 (which appear as twice this when getsockopt is invoked, thus we see 60,000).

Certain socket options are not available – getsockopt does not find them. In this case we skip it. Other times, the operating system has restrictions to prevent overutilization of resources. In the case of SO\_DEBUG, we receive a “Permission denied” error when we try to enable it. Lastly, some protocols are not available.

Overall, the client on Zeus TXST Linux server should send 20 socket options to the server on Eros. The server should, in turn, process 20 socket options.

Here is the client payload:

Msg received from client is:

```
SO_BROADCAST,6,off;SO_DEBUG,1,off;SO_DONTROUTE,5,off;SO_ERROR,4,0;SO_KEEPA  
LIVE,9,off;SO_LINGER,13,1_onoff = 0, 1_linger =  
0;SO_OOBINLINE,10,off;SO_RCVBUF,8,367360;SO_SNDBUF,7,87040;SO_RCVLOWAT,18,1;SO_S  
NLOWAT,19,1;SO_RCVTIMEO,20,0 sec, 0 usec;SO_SNDTIMEO,21,0 sec, 0  
usec;SO_REUSEADDR,2,off;SO_REUSEPORT,15,off;SO_TYPE,3,1;IP_TOS,1,0;IP_TTL,2,64;TC  
P_MAXSEG,2,1448;TCP_NODELAY,1,off;
```

Below are screenshots demonstrating a full run of the application on the TXST Linux hosts Zeus and Eros. Please refer to the README.md file for instructions on how to build and run this application and the source code for implementation details.

```
[bss64@eros tcp-socket-options]$ ls -l
total 104
drwx-----. 2 bss64 student 4096 Jun 17 22:12 include
-rw-----. 1 bss64 student 34523 Jun 17 22:12 LICENSE
-rw-----. 1 bss64 student 960 Jun 17 22:12 Makefile
-rw-----. 1 bss64 student 2630 Jun 17 22:12 README.md
drwx-----. 3 bss64 student 4096 Jun 17 22:12 resources
drwx-----. 2 bss64 student 4096 Jun 17 22:12 src
-rw-----. 1 bss64 student 1483 Jun 17 22:12 tcp_client.cpp
-rw-----. 1 bss64 student 9990 Jun 17 22:12 tcp_server.cpp
[bss64@eros tcp-socket-options]$ make
g++ -o client src/tcp_client.cpp -g -std=c++11
g++ -o server src/tcp_server.cpp -g -std=c++11
[bss64@eros tcp-socket-options]$ ./server
Hello! Starting TCP server on port 5001.....

Default server option values:
SO_BROADCAST: default = off
SO_DEBUG: default = off
SO_DONTROUTE: default = off
SO_ERROR: default = 0
SO_KEEPAIVE: default = off
SO_LINGER: default = 1_onoff = 0, 1_linger = 0
SO_OOINLINE: default = off
SO_RCVBUF: default = 87380
SO_SNDBUF: default = 16384
SO_RCVLOWAT: default = 1
SO_SNDLOWAT: default = 1
SO_RCVTIMEO: default = 0 sec, 0 usec
SO_SNDTIMEO: default = 0 sec, 0 usec
SO_REUSEADDR: default = off
SO_REUSEPORT: default = off
SO_TYPE: default = 1
SO_USELOOPBACK: getsockopt error
IP_TOS: default = 0
IP_TTL: default = 64
IPV6_DONTFRAG: (undefined)
IPV6_UNICAST_HOPS: Cannot create fd for level: 41
getsockopt error
IPV6_V6ONLY: Cannot create fd for level: 41
getsockopt error
TCP_MAXSEG: default = 536
TCP_NODELAY: default = off
SCTP_AUTOCLOSE: (undefined)
SCTP_MAXBURST: (undefined)
SCTP_MAXSEG: (undefined)
SCTP_NODELAY: (undefined)

Listening for connections....
```

Figure 1: Build the project and start the server on Eros.

```
[bss64@zeus tcp-socket-options]$ ./client eros.cs.txstate.edu 5001
Sending socket options to server in format: opt_str,opt_name,opt_val;
  displayed below as opt_str(opt_name),opt_val , where opt_name is an integer value.
SO_BROADCAST(6),off
SO_DEBUG(1),off
SO_DONTROUTE(5),off
SO_ERROR(4),0
SO_KEEPAIVE(9),off
SO_LINGER(13),l_onoff = 0, l_linger = 0
SO_OOBINLINE(10),off
SO_RCVBUF(8),367360
SO_SNDBUF(7),87040
SO_RCVLOWAT(18),1
SO_SNDLOWAT(19),1
SO_RCVTIMEO(20),0 sec, 0 usec
SO_SNDTIMEO(21),0 sec, 0 usec
SO_REUSEADDR(2),off
SO_REUSEPORT(15),off
SO_TYPE(3),1
getsockopt did not find the option, skipping.....
IP_TOS(1),0
IP_TTL(2),64
IPV6_DONTFRAG: (undefined) - skipping.....
getsockopt did not find the option, skipping.....
getsockopt did not find the option, skipping.....
TCP_MAXSEG(2),1448
TCP_NODELAY(1),off
SCTP_AUTOCLOSE: (undefined) - skipping.....
SCTP_MAXBURST: (undefined) - skipping.....
SCTP_MAXSEG: (undefined) - skipping.....
SCTP_NODELAY: (undefined) - skipping.....

Sent 20 options to the server for processing.
Please verify server reply below AND output on the server itself!

Server reply is: Hello - I inverted/modified the socket options you sent me and printed on my end!
```

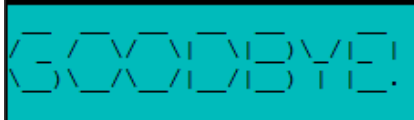
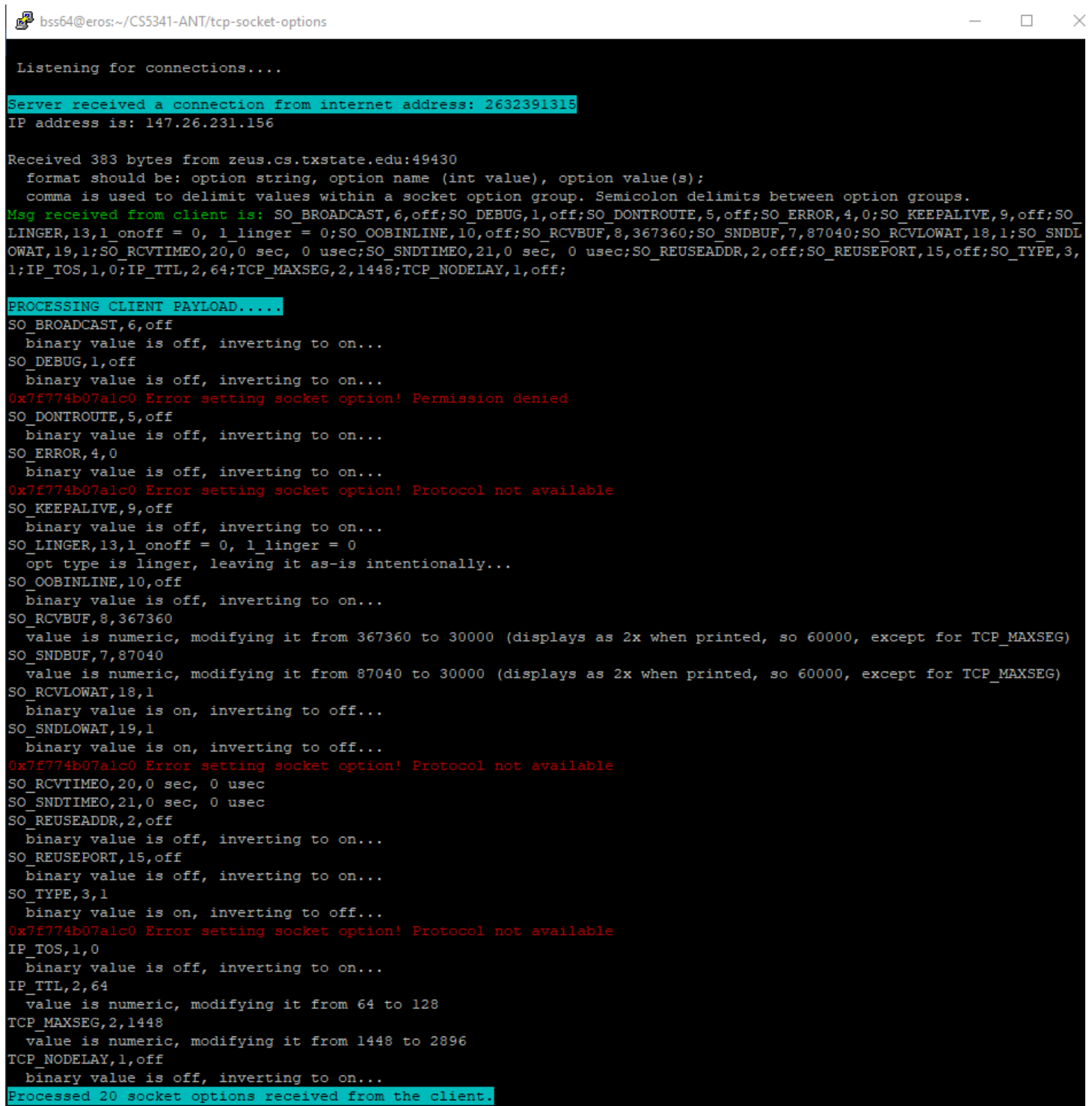


Figure 2: Run the client on Zeus.



```

bss64@eros:~/CS5341-ANT/tcp-socket-options
Listening for connections....


Server received a connection from internet address: 2632391315
IP address is: 147.26.231.156

Received 383 bytes from zeus.cs.txstate.edu:49430
  format should be: option string, option name (int value), option value(s):
  comma is used to delimit values within a socket option group. Semicolon delimits between option groups.
Msg received from client is: SO_BROADCAST,6,off;SO_DEBUG,1,off;SO_DONTROUTE,5,off;SO_ERROR,4,0;SO_KEEPA_LIVE,9,off;SO_LINGER,13,1,onoff = 0, 1_linger = 0;SO_OOBINLINE,10,off;SO_RCVBUF,8,367360;SO_SNDBUF,7,87040;SO_RCVLOWAT,18,1;SO_SNDLOWAT,19,1;SO_RCVTIMEO,20,0 sec, 0 usec;SO_SNDTIMEO,21,0 sec, 0 usec;SO_REUSEADDR,2,off;SO_REUSEPORT,15,off;SO_TYPE,3,1;IP_TOS,1,0;IP_TTL,2,64;TCP_MAXSEG,2,1448;TCP_NODELAY,1,off;

PROCESSING CLIENT PAYLOAD....
SO_BROADCAST,6,off
  binary value is off, inverting to on...
SO_DEBUG,1,off
  binary value is off, inverting to on...
0x7f774b07alc0 Error setting socket option! Permission denied
SO_DONTROUTE,5,off
  binary value is off, inverting to on...
SO_ERROR,4,0
  binary value is off, inverting to on...
0x7f774b07alc0 Error setting socket option! Protocol not available
SO_KEEPA_LIVE,9,off
  binary value is off, inverting to on...
SO_LINGER,13,1,onoff = 0, 1_linger = 0
  opt type is linger, leaving it as-is intentionally...
SO_OOBINLINE,10,off
  binary value is off, inverting to on...
SO_RCVBUF,8,367360
  value is numeric, modifying it from 367360 to 30000 (displays as 2x when printed, so 60000, except for TCP_MAXSEG)
SO_SNDBUF,7,87040
  value is numeric, modifying it from 87040 to 30000 (displays as 2x when printed, so 60000, except for TCP_MAXSEG)
SO_RCVLOWAT,18,1
  binary value is on, inverting to off...
SO_SNDLOWAT,19,1
  binary value is on, inverting to off...
0x7f774b07alc0 Error setting socket option! Protocol not available
SO_RCVTIMEO,20,0 sec, 0 usec
SO_SNDTIMEO,21,0 sec, 0 usec
SO_REUSEADDR,2,off
  binary value is off, inverting to on...
SO_REUSEPORT,15,off
  binary value is off, inverting to on...
SO_TYPE,3,1
  binary value is on, inverting to off...
0x7f774b07alc0 Error setting socket option! Protocol not available
IP_TOS,1,0
  binary value is off, inverting to on...
IP_TTL,2,64
  value is numeric, modifying it from 64 to 128
TCP_MAXSEG,2,1448
  value is numeric, modifying it from 1448 to 2896
TCP_NODELAY,1,off
  binary value is off, inverting to on...
Processed 20 socket options received from the client.

```

Figure 3: Server on Eros processes client payload sent from Zeus.

 bss64@eros:~/CS5341-ANT/tcp-socket-options

```

*****
NEW server socket option values (after modifying client values):
SO_BROADCAST: default = on
SO_DEBUG: default = off
SO_DONTROUTE: default = on
SO_ERROR: default = 0
SO_KEEPAIVE: default = on
SO_LINGER: default = 1_onoff = 0, 1_linger = 0
SO_OOBINLINE: default = on
SO_RCVBUF: default = 60000
SO_SNDBUF: default = 60000
SO_RCVLOWAT: default = 1
SO_SNDLOWAT: default = 1
SO_RCVTIMEO: default = 1 sec, 500000 usec
SO_SNDTIMEO: default = 1 sec, 500000 usec
SO_REUSEADDR: default = on
SO_REUSEPORT: default = on
SO_TYPE: default = 1
SO_USELOOPBACK:  getsockopt error
IP_TOS: default = 0
IP_TTL: default = 128
IPV6_DONTFRAG:  (undefined)
IPV6_UNICAST_HOPS:  getsockopt error
IPV6_V6ONLY:  getsockopt error
TCP_MAXSEG: default = 1448
TCP_NODELAY: default = on
SCTP_AUTOCLOSE:  (undefined)
SCTP_MAXBURST:  (undefined)
SCTP_MAXSEG:  (undefined)
SCTP_NODELAY:  (undefined)

```

Figure 4: New server socket opts after processing. Binary flags are inverted, numeric types are changed, timeval is set to sec = 1 and usec = 50000, Linger is kept intentionally the same.

```

^C
Termination command invoked. Shutting down server.

```

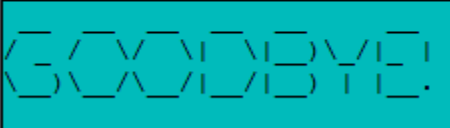


Figure 5: Shutting down the server with Ctrl + C. Also Ctrl + Z may be used, or the Linux kill command.

## References

- [1] <https://stackoverflow.com/questions/9402254/how-do-you-run-a-function-on-exit-in-c>
- [2] [https://notes.shichao.io/unp/ch7/#so\\_broadcast-socket-option](https://notes.shichao.io/unp/ch7/#so_broadcast-socket-option)
- [3] [https://www.bogotobogo.com/cplusplus/sockets\\_server\\_client.php](https://www.bogotobogo.com/cplusplus/sockets_server_client.php)
- [4] <https://www.beej.us/guide/bgnet/html/>
- [5] <https://smartbear.com/blog/develop/the-biggest-changes-in-c11-and-why-you-should-care/>
- [6] <https://stackoverflow.com/questions/4654636/how-to-determine-if-a-string-is-a-number-with-c>
- [7] <https://linux.die.net/man/7/tcp>
- [8] W. R. Stevens, Bill Fenner, and Andrew M. Rudoff. UNIX Network Programming – Networking APIs: Sockets and XTI (3rd ed.). Addison-Wesley, 2004. ISBN: 0-13-141155-1.

--- NOTHING FOLLOWS ---