```
brow1770@ecegrid-thin7 ~/404/hw10
$ ./client 127.0.0.1
Say something: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA\x18\x0e\x40\x00
You Said: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA@
```

The string above is the string that caused the buffer to overflow. I obtained this string by first using the disas secretFunction command to obtain the address I needed to replace the return of clientComm with, with the last 4 bytes of that address shown above. Then, I set a breakpoint at the end of clientComm at the leaveq instruction in order to use a test case. Then, I connected to the server, sent in a test string, and let the program continue until the breakpoint, where I used the info frame command and found the difference in the rip – the start of my test string, which is how I figured out how many A's to put before the last 4 bytes of my return address, which was 40.

Code Modification -> Because str was declared at the top of the clientComm function with MAX_DATA_SIZE as its size, and before numBytes of recvBuff was known, there is a vulnerability because it will allow strings of any size to be strcpyd into recvBuff. In order to fix this, I commented out the original declaration of str[MAX_DATA_SIZE], and declared it right above the strcpy function using "char *str = malloc(sizeof(char) * (numBytes + 1));", which only allows strings of size numBytes + 1 to be copied, thus fixing the problem.