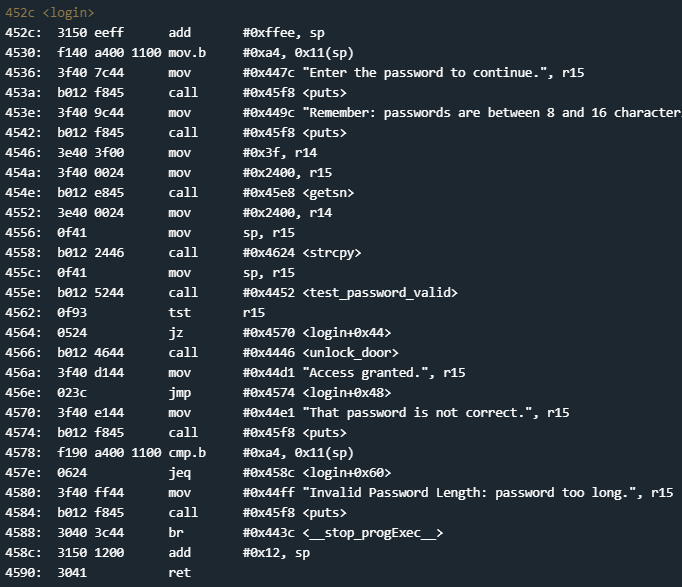
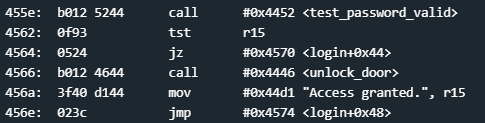
Skipping over <main> and going right to <login>:



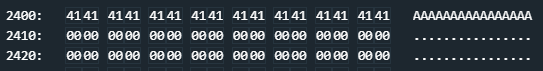
Looking for any cmp we find one at line 4578:



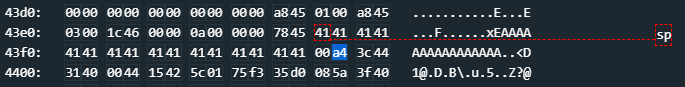
0x11(sp)? Probably position 11 from wherever sp is located. We at least know to look for a4 in memory. That being said, this is pretty far in the code, let’s look around where the door unlocks:



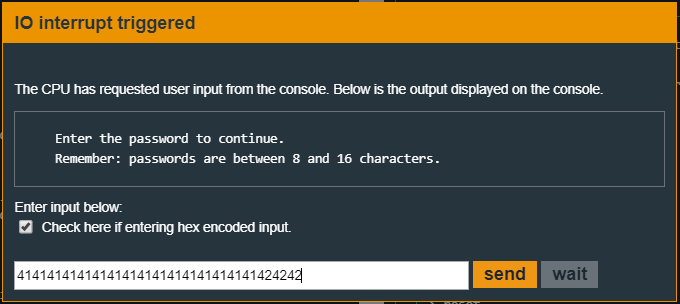
Looks very similar to Cusco. If that is the case, then I’m going to guess that we don’t actually need to go into <test\_password\_valid> to exploit this lock. Let’s throw in 16 As worth of test input and see what memory looks like:



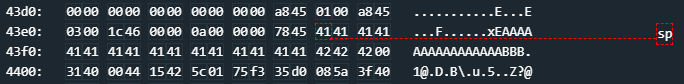
They got smart and didn’t store it near the program code. This may cause a problem. Let’s look near the compare, setting a breakpoint right before it so we can see some values before the compare:



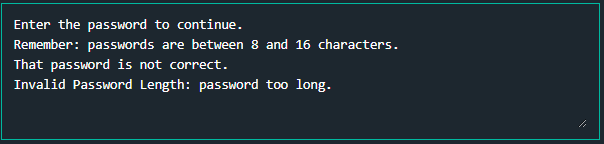
Wait, now our input is copied up near our sp AND we see the hex value a4, which is what is getting compared. I bet that it is checking that a4 is in position 11 to check that there hasn’t been an overflow. Let’s test that theory:



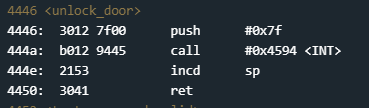
Passing in 16 As and 4 Bs gives us:



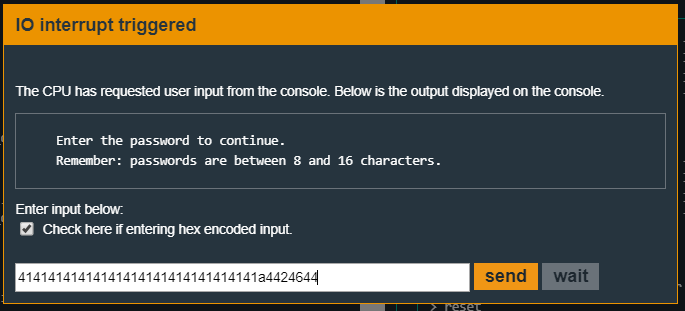
Notice that a4 is overwritten by 42 at position 11 from sp. This should cause us some problems:



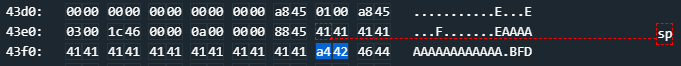
As expected it kicks us out because we fail the comparison. Here’s what happens: We are checking to make sure that no more than 16 characters have been entered. If a 17th character is detected it kicks us out. The reason we check against position 11 instead of position 10 is because we are little endian here. The 17th character will be at position 11, not 10. My guess is that ebp is just past position 11, so if we pass a string with a4 at position 11 and then put the value we want to jump to, we should be able to hijack program execution. Where is that unlock function:



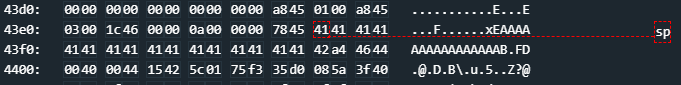
Alright, time to test our theory and, hopefully, jump to 4446:



Notice that we have a442, this should show up as 42a4 in memory:



Oh? Yeah! Whenever it places it into memory it reads in exactly but when it pops memory and places it into a register it is little endian. Let’s try this again:



Much better and, as expected, we unlocked the door:

