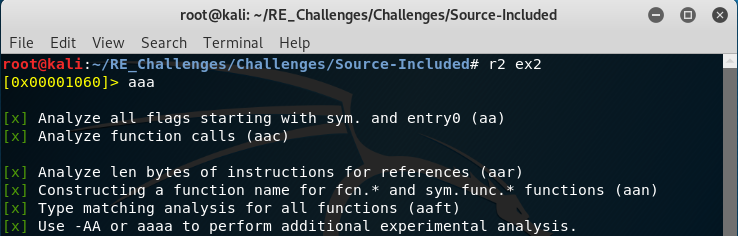
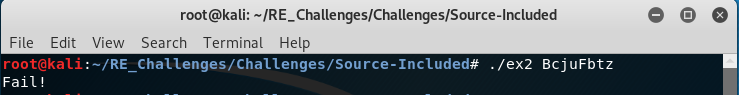
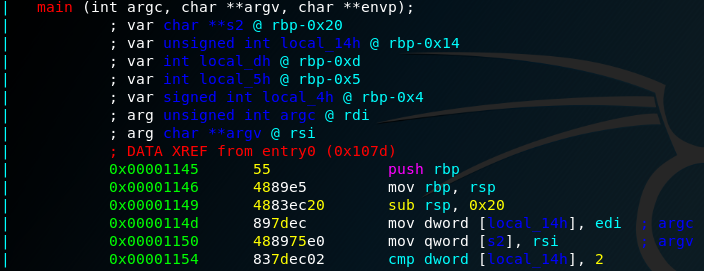
***Ex2***

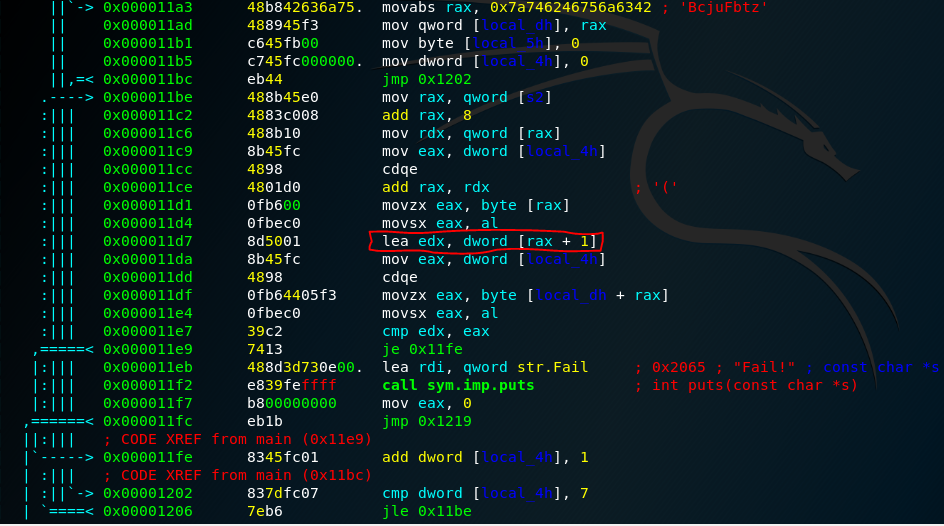
In this challenge we must find the hard-coded string and decrypt it using a caesar cipher. A caesar cipher is the first cipher you will learn about when reading or being taught about cryptography. Julius Caesar used a simple ciphering technique to communicate with his military to prevent people from being able to intercept and know his commands. The cipher is very simple, you take a message “Hello” and a shift “1” and shift the characters by that amount. The cipher for the message “Hi” with a shift of “1” is “If” and to get the original message back we perform a shift of “-1”. If you look up the ascii table on your terminal with the command “man ascii” you can see the hex, decimal, octal and so on information regarding ascii text. If you were to look at the ascii hexadecimal values for the numbers 0-9 you will see they are 0x30-0x39, just as these values are in numerical order starting at 0x30 all alphabetical characters including lower case and capital start at one hexadecimal value and progress in order, alphabetical order. From a programmer’s perspective we can easily just add 1 to a hexadecimal representation of a character to get the next one in sequence just as we want to do with the caesar ciphers shift. With this priming knowledge we are ready to begin this exercise.

Just as before we run radare2 and start analysis (get use to running “a” associated analysis even if it may not be needed because it is better you have it done and not need it that get deep into reversing and possibly have needed it, that is unless you are looking at very large files)

Upon performing a “[pdf@main](mailto:pdf@main)” you will find this line of disassembly. Lets try this sequence of ascii text and see what happens.

As you would probably expect this is not the correct input but the ciphered text that is being compared to our input string before it has had the caesar cipher shift to it.

You will need to make sure you are aware of the stack variables and where our user input string is to understand what is what as you follow the flow of the code.

If you followed the code from the beginning you will eventually get to this segment of code, this deals with a loop that iterates through the user input string and the hard coded ciphered string comparing the current character in the hard coded string to the shifted character of our user input string, “lea edx, dword [rax + 1]”. That instruction is moving the contents of the rax register plus 1, rax at this point in execution is holding a character from our user input string so we can deduce that the cipher shift is 1.

With this knowledge we can take the hard-coded string we found earlier and build the correct input by stepping each character back one, “B” would become “A” and “c” would become “b”.

