# Day 17 - [Reverse Engineering] ReverseELFneering

**Tool Used:** Kali Linux, firefox, Nmap, radare2

## Solution/walkthrough:

## Q1

3. Register me this, register me that...

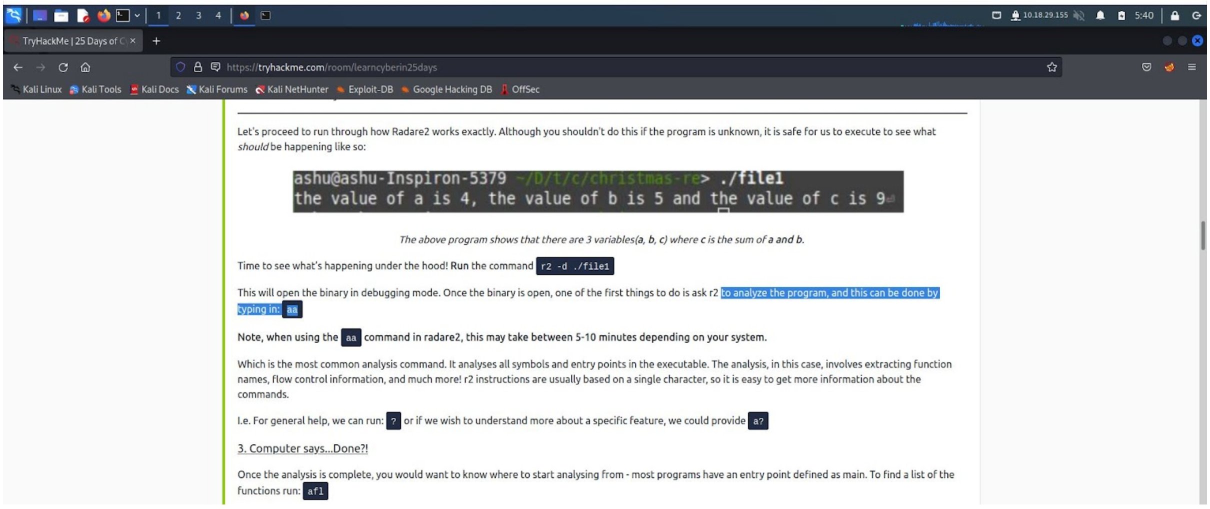
The core of assembly language involves using registers to do the following:

- Transfer data between memory and register, and vice versa
- Perform arithmetic operations on registers and data
- Transfer control to other parts of the program Since the architecture is x86-64, the registers are 64 bit and Intel has a list of 16 registers:

| Initial Data Type | Suffix | Size (bytes) |
|---|---|---|
| Byte | b | 1 |
| Word | w | 2 |
| Double Word | l | 4 |
| Quad | q | 8 |
| Single Precision | s | 4 |
| Double Precision | l | 8 |

study from Try Hack Me.

## Q2



Let's proceed to run through how Radare2 works exactly. Although you shouldn't do this if the program is unknown, it is safe for us to execute to see what *should* be happening like so:

```
ashu@ashu-Inspiron-5379 ~/D/t/c/christmas-re> ./file1
the value of a is 4, the value of b is 5 and the value of c is 9
```

*The above program shows that there are 3 variables(a, b, c) where c is the sum of* ***a and b.***

Time to see what's happening under the hood! **Run the command** `r2 -d ./file1`

This will open the binary in debugging mode. Once the binary is open, one of the first things to do is ask r2 to analyze the program, and this can be done by typing in: `aa`

**Note, when using the `aa` command in radare2, this may take between 5-10 minutes depending on your system.**

Which is the most common analysis command. It analyses all symbols and entry points in the executable. The analysis, in this case, involves extracting function names, flow control information, and much more! r2 instructions are usually based on a single character, so it is easy to get more information about the commands.

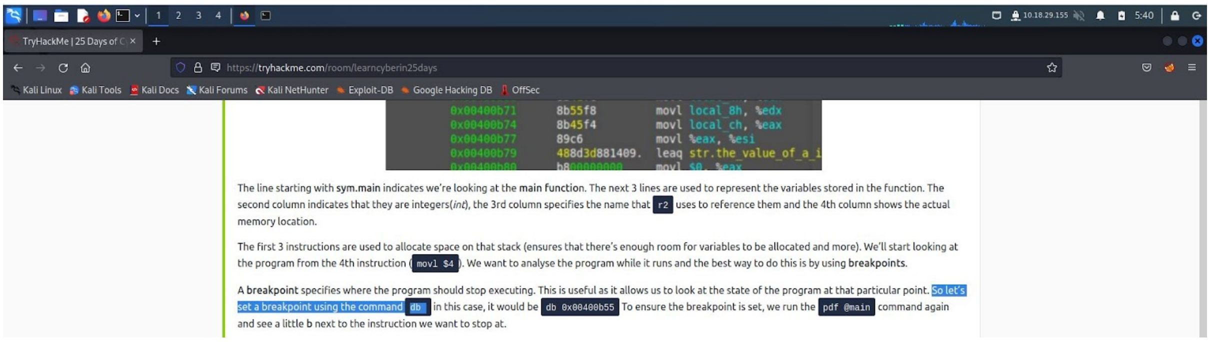
I.e. For general help, we can run: `?` or if we wish to understand more about a specific feature, we could provide `a?`

3. Computer says...Done?!

Once the analysis is complete, you would want to know where to start analysing from - most programs have an entry point defined as main. To find a list of the functions run: `afl`

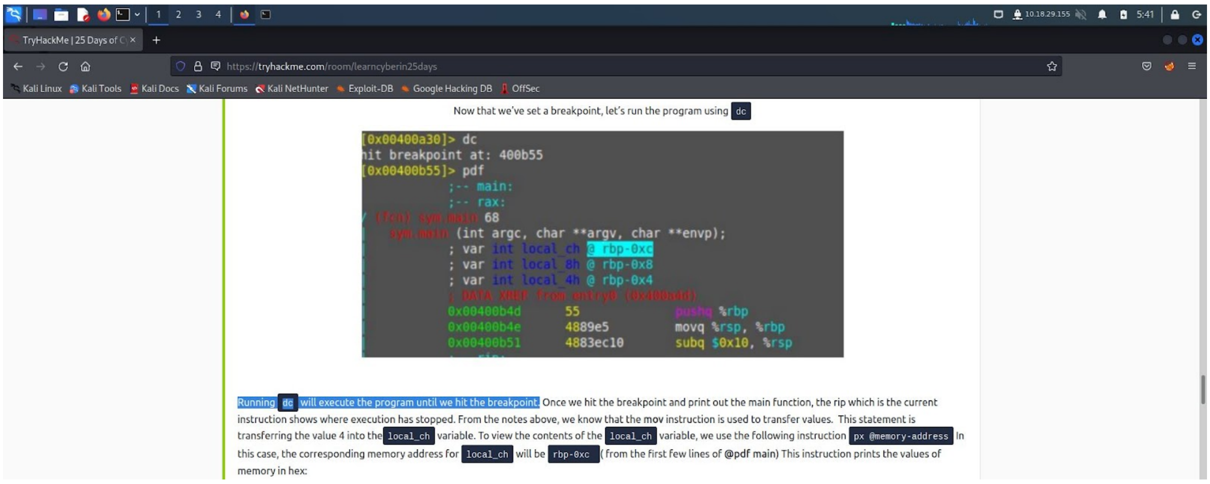
study from Try Hack Me.

## Q3



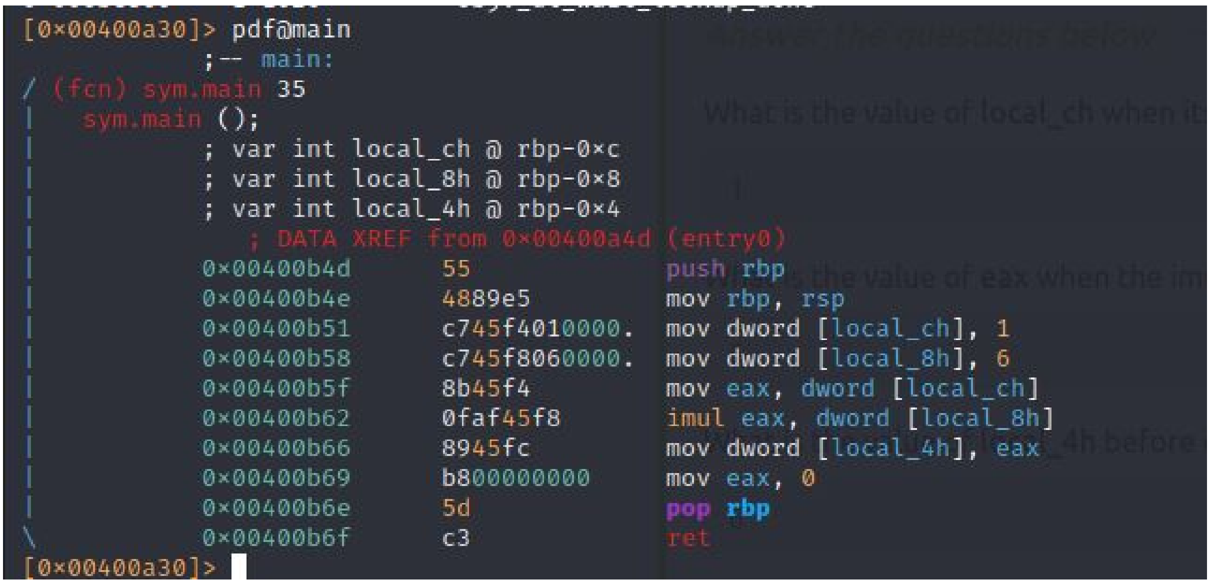

study from Try Hack Me.

## Q4




Study from Try Hack Me.

## Q5




it copies value 1 to [local_ch]. therefore, the value of [local_8h] should be 1.

## Q6

when mov dword [local_8h] is called, the value of [local_8h] becomes 6. when mov eax, dword [local_ch] is called, the value of eax becomes 1. when imul is called, it multiplies the value of [local_8h]to eax, which is 6*1. Therefore, the value of eax should be 6.

## Q7

it copies the value of eax to [local_4h]. Therefore, the value should be 6.