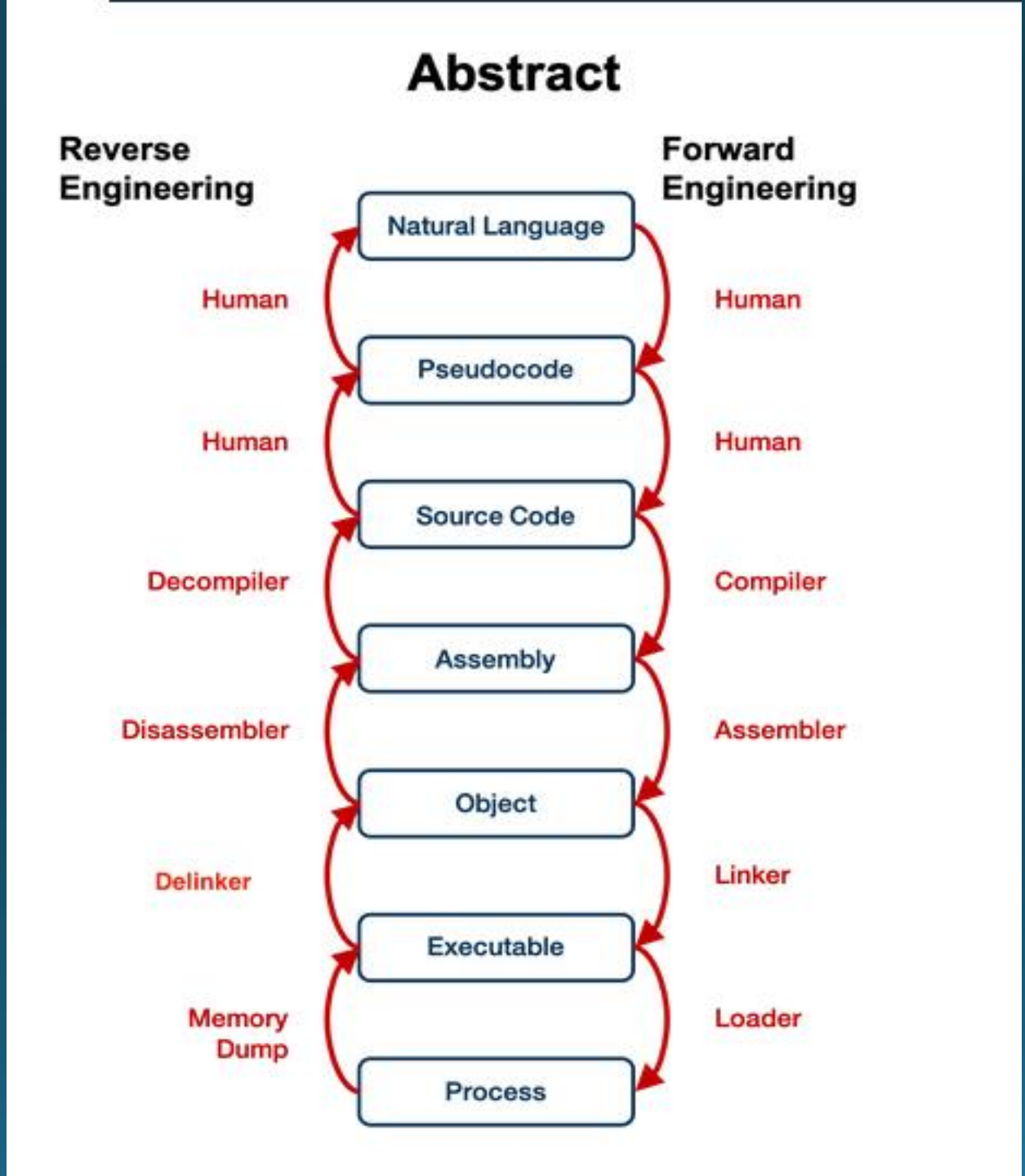
for fun and flags

Intro to Reverse Engineering

The Life of a Binary



ex01.c

- On your Kali VM, open a terminal
- `cd ~/IntroToRe`
- `code .`
- Open and read ex01.c
- `cd ./ex01`
- `gcc ./ex01.c -o ex01.exe`
- `file ./ex01.exe`
- `strings ./ex01.exe`
- `./ex01.exe`

Ghidra!

- `ghidra`
- File -> New Project -> Non-Shared Project -> Next
- Name it whatever you like -> Finish
- File -> Import File -> IntroToRe/ex01/ex01.exe -> Select File To Import -> OK -> OK
- Double click on ex01.exe -> Yes to analyze
- Close the window when done

ex02.c

- In VS Code, open and read ex02.c
- `cd ../ex02`
- `gcc ./ex02.c -o ex02.exe`
- `./ex02.exe`
- Now let's open it in Ghidra!
- File -> Import -> etc as before
- Now here we've got some work to do

GDB

- GDB is a debugger
- `gdb ./ex02.exe`
- GDB will choose a base address to load the process into
- Ghidra will likely have chosen a different base address
- Let's sync the two so our addresses line up!
- `info proc mappings`
- In Ghidra, click on the circuit board Icon (memory map)
- Click on the house in the upper right corner
- Input the start address given by gdb

Syncing GDB and Ghidra

- Start the target binary in GDB
- Go to the Ghidra projects window
- Rightclick on the program -> Open with -> Debugger
- Debugger targets window -> Click on the green/yellow Icon
- Select gdb from the dropdown and check “Use existing session via new-ui”
- Hit Connect
- Enter the command from the popup window in your active gdb session

Linux 64bit Calling convention

- The first 6 arguments to a function call are passed in order, in the following register:
  - RDI, RSI, RDX, RCX, R8, R9
  - Any subsequent arguments are passed on the stack

strcmp in GDB

- strcmp(string_1, string_2)
- string_1 is stored in RDI
- string_2 is stored in RSI
- First we set a breakpoint on the address where strcmp is called
- `b *0x<address of strcmp>`
- Run the program
- `r`
- The program runs until we hit our breakpoint
- Then we examine (x) the strings (s) stored in the registers ($rdi, $rsi)
- `x/s $rdi`
- `x/s $rsi`
- To continue execution after a breakpoint, hit `c`

ex03.exe

- `cd ../ex03`
- ex03.exe is the classic CMU Binary Bomb
- There are 6 phases you have to diffuse, or it blows up
- You can enter the passwords one at a time, or you can create a text file with the passwords for each phase on different lines
- `./ex03.exe ./text_file_with_passwds.txt`

ex03.exe

- At this point, we will walk through a few stages together
- Feel free to work ahead with what you've learned so far