Bryan Heim – Project 2 – unlocking bph11_1

Correct Passphrase: eWNU1DqVAgYHMYzmVTPYinZVTC

Path to solution: The first thing I decided to do was run strings on bph11_1 to see if the password could be easily recognized. The output was well over 100 lines but after skimming through I ran into this interesting string.

```
[^_]
ffffff.
eWMNUlDqVAgYHMYzmVTPYinZVTC
Sorry! Not correct!
Congratulations!
Unlocked with passphrase %s
FATAL: kernel too old
```

So I decided to investigate further and run the program in gdb. I set a breakpoint at main, which it found, and then ran the program and disassembled the code in main. Inside the code I saw two functions and a comparison. One was to the function call fgets and the other to a function named chopped. So I decided to set a breakpoint after each of these statements to see what happens to the input.

```
(gdb) b *0x080482fa
Breakpoint 2 at 0x80482fa
(gdb) b *0x08048302
Breakpoint 3 at 0x8048302
(gdb) c
Continuing.
attempt
```

At breakpoint two I decided to check $edi to make sure fgets put our input “attempt” into the register.

```
(gdb) x/s $edi
0xffffd15c:      "attempt\n"
```

And then at breakpoint 3, I decided to see what the chomp function did to my input

```
(gdb) x/s $edi
0xffffd15c:      "attempt"
```

So that I know the program took my input and deleted the \n captured from fgets, I went to the comparison statement to check to see what was in the register that it was comparing my input to.

```
   0x0804830c <+61>:    repz cmpsb %es:(%edi),%ds:(%esi)
```

After setting the breakpoint I went to see the contents of that register.

```
Breakpoint 4, 0x0804830c in main ()
(gdb) x/s $edi
0xffffd15c:      "attempt"
(gdb) x/s $esi
0x80b388c <__dso_handle+4>:      "eWMNUlDqVAgYHMYzmVTPYinZVTC"
```

Sure enough, it was the same string found by the program strings, and was what my input was being compared too. I then quit gdb and used the found contents of $esi as the passphrase

```
(7) thot $ ./bph11_1
eWMNUlDqVAgYHMYzmVTPYinZVTC
Congratulations!
Unlocked with passphrase eWMNUlDqVAgYHMYzmVTPYinZVTC
```

After unlocking the program, I realized that this string was hardcoded into the c source code and was what the program was comparing my input too.