Assignment-5

Pallavi Kumari Jha AM.EN.P2CSN16014

1.

In first program, the code is being copied to buffer.

strcpy(buffer, str);

Thus code is on stack. And to run the code we need to make stack executable so execstack is required.

Whereas in second program, the code is in data segment.

Declaration:

```
const char code[] =
"\x31\xc0"
"\x50"
"\x68""//sh"
"\x68""/bin"
"\x89\xe3"
"\x50"
"\x53"
"\x89\xe1"
"\x99"
"\xb0\x0b"
"\xcd\x80";
```

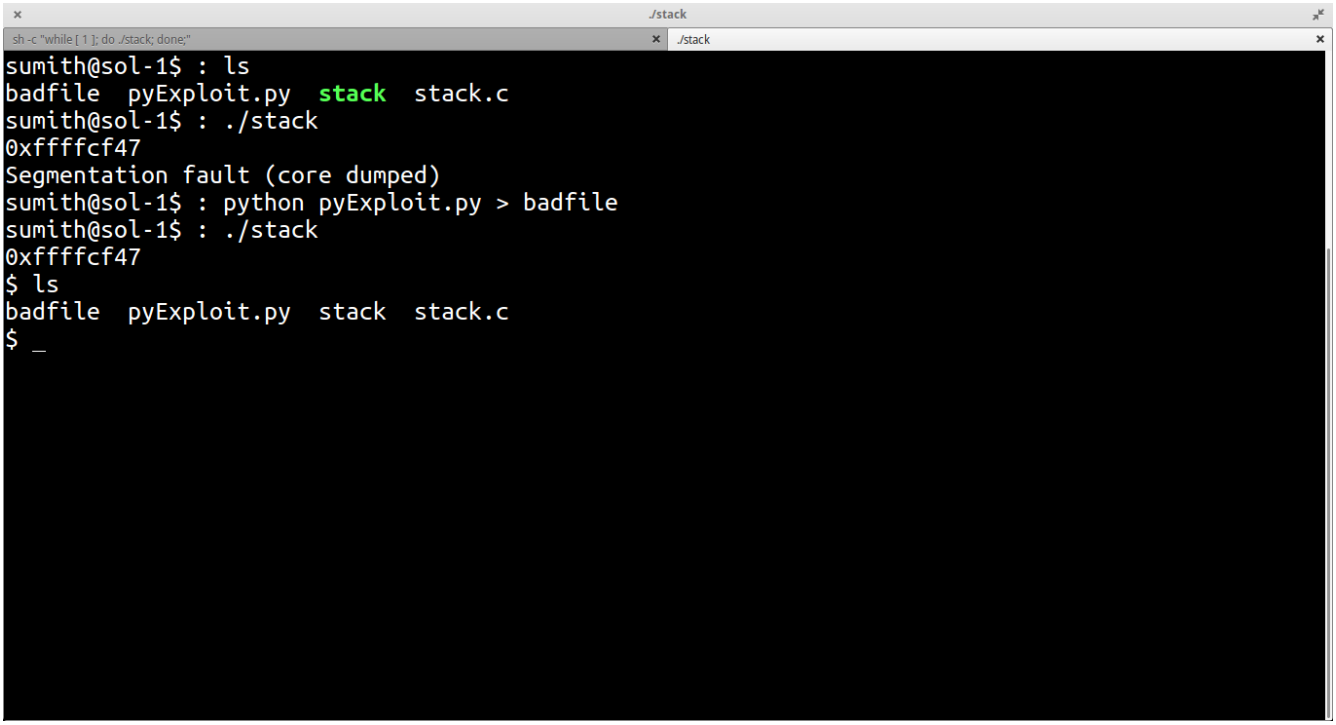
and the following line

```
int (*ret)() = (int(*)())code;
```

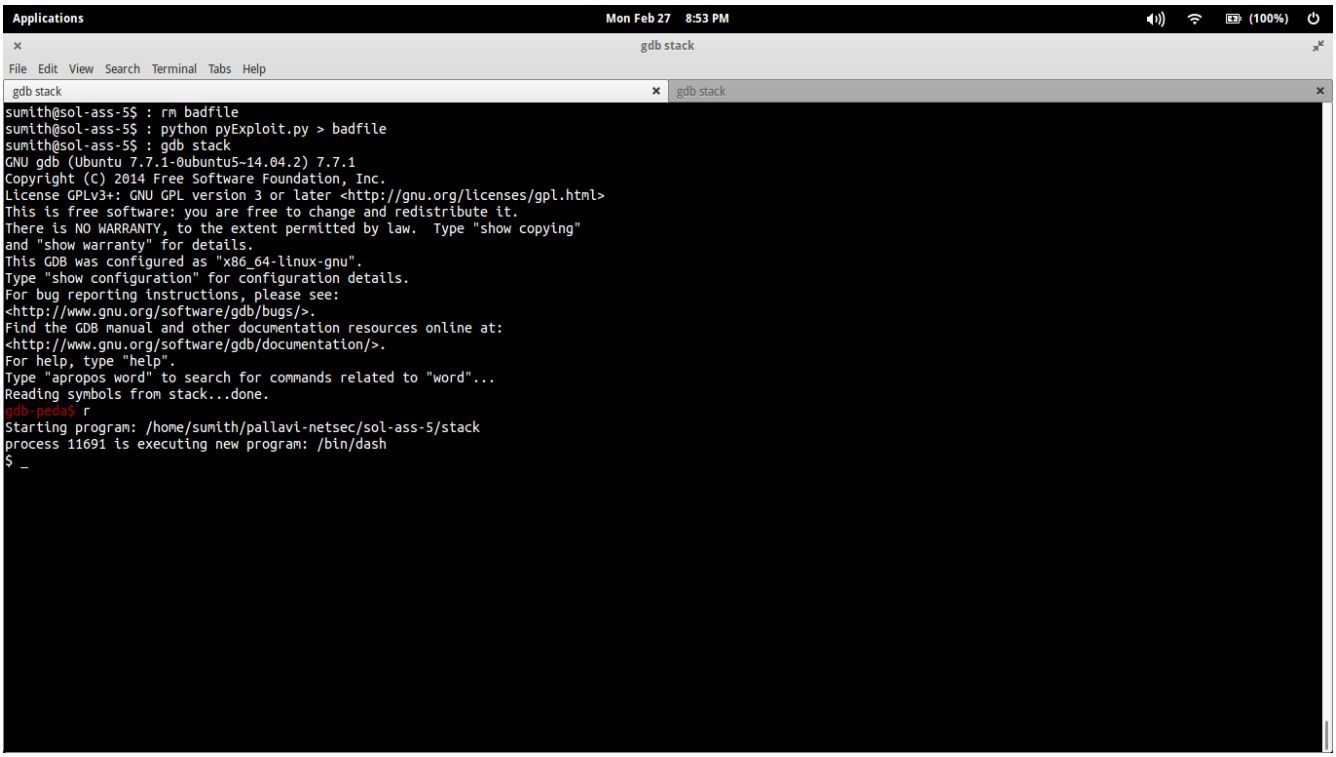
The data segment is read only, it is executable and thus flag execstack is not required.

2.

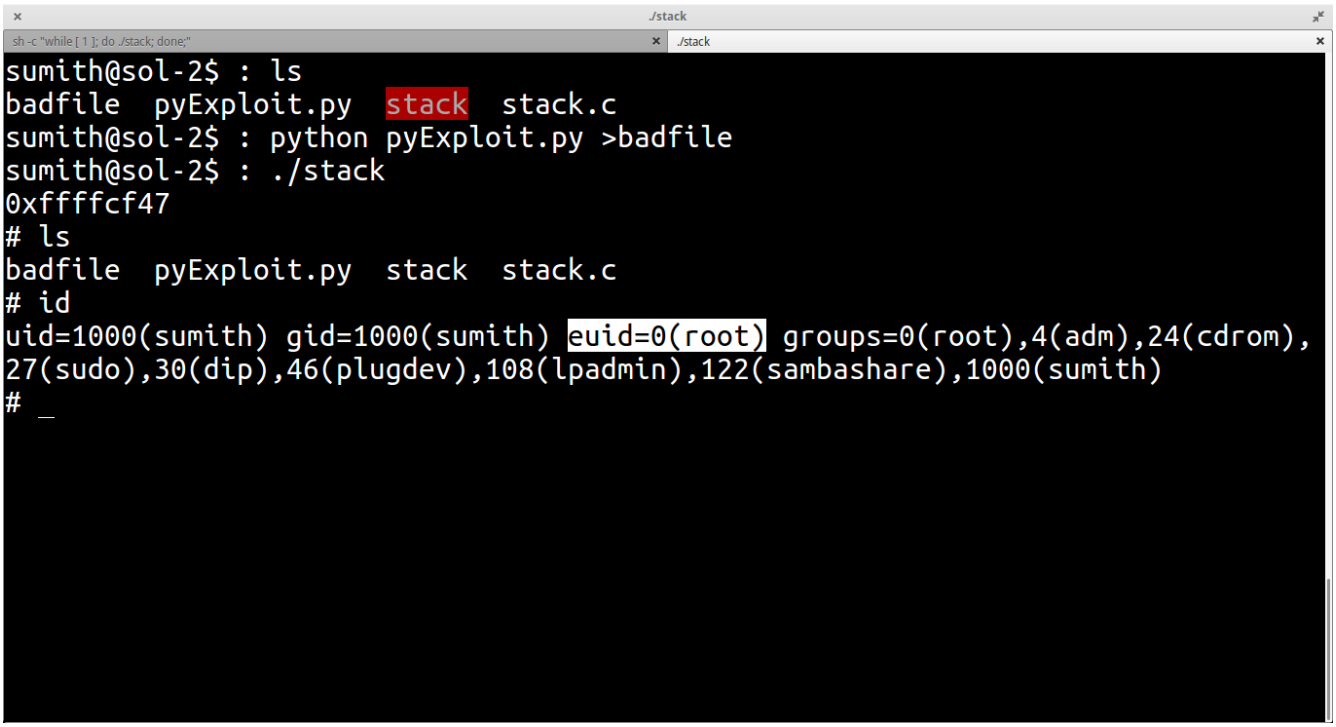
Shell in terminal



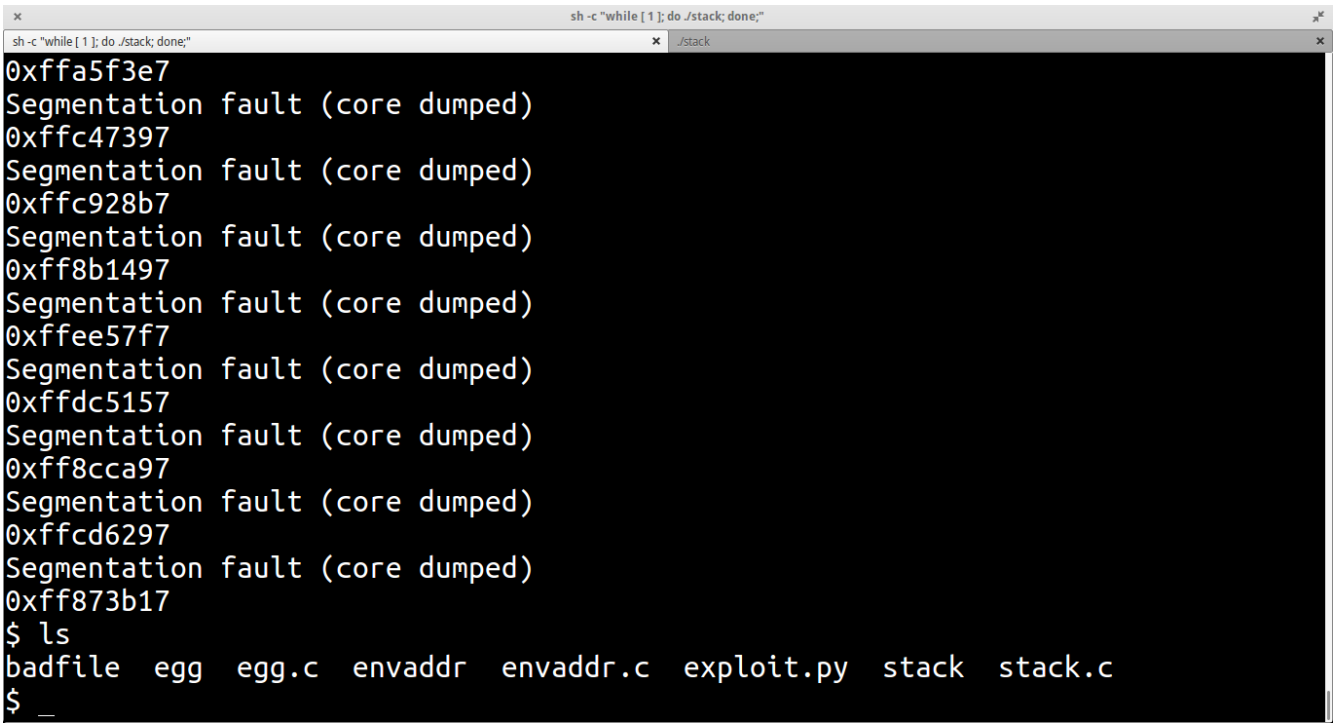
Shell in GDB



3.
Root shell in terminal



4.
Shell with Address Randomization on



ASLR (Address space layout randomization) purpose is to randomization the address of the items in memory. Buffer overflow done in question 1 and 2 depends on the prior knowledge of the memory location of the item (which was overflown). If ASLR is used then the items are given new memory location every time the code gets executed making it difficult for the attacker to make valid memory references (referencing to malicious self contained codes).

Approach: Modified exploit code given in class lab hour a bit to get shell with ASLR enabled. Increases the number of NOPs to 5000 and changed the return address in python exploit file accordingly.