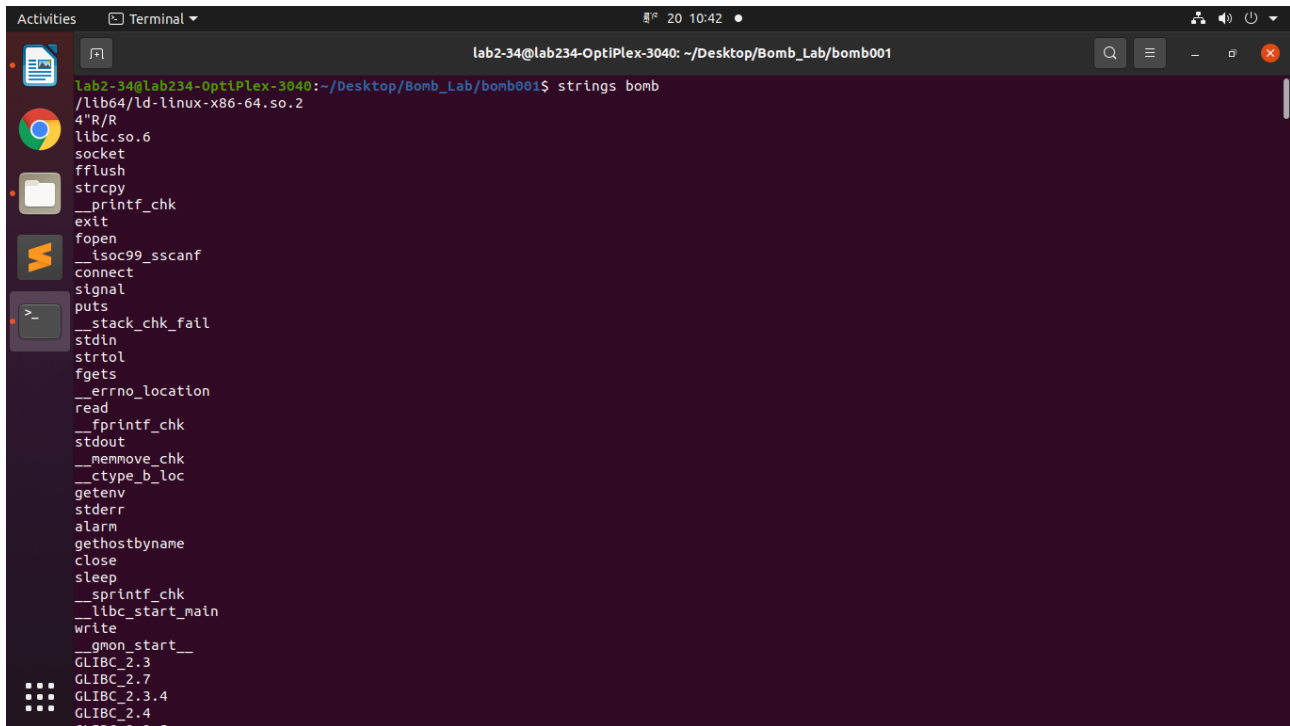


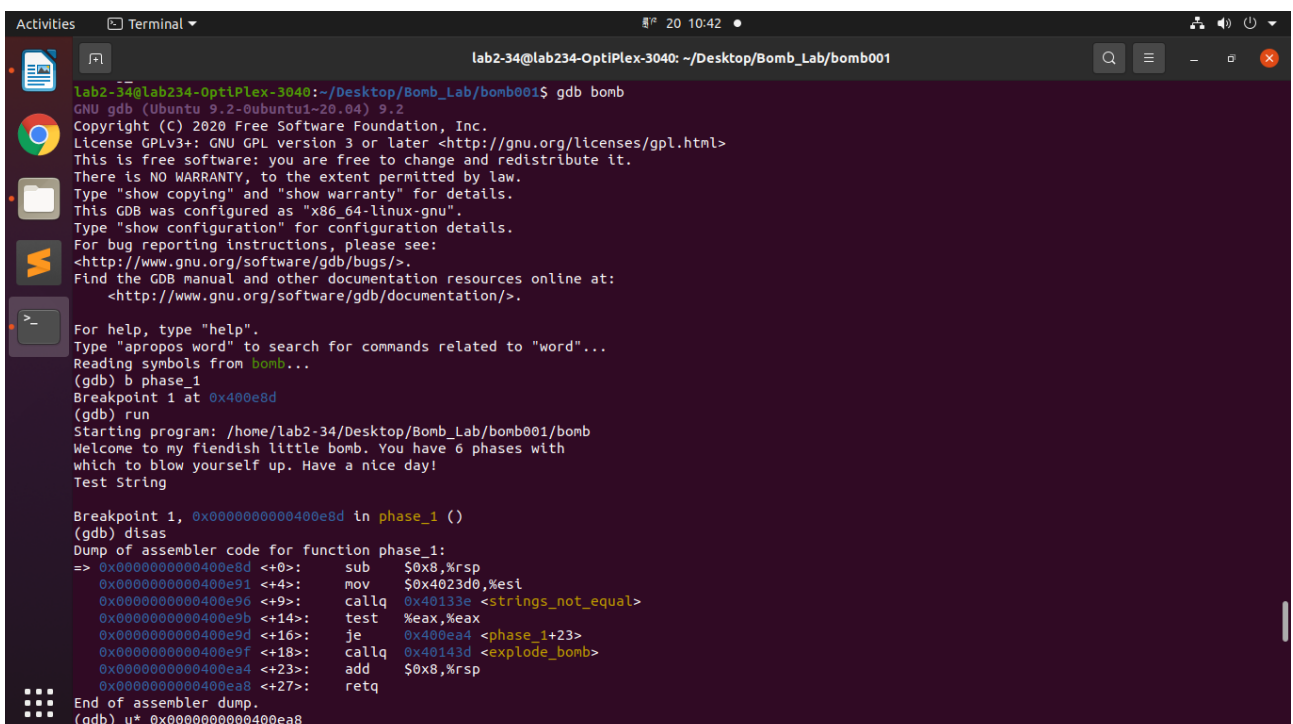
PHASE 01

Step 1) The strings bomb utility will display the printable strings in your bomb.



```
lab2-34@lab234-OptiPlex-3040: ~/Desktop/Bomb_Lab/bomb001
lab2-34@lab234-OptiPlex-3040:~/Desktop/Bomb_Lab/bomb001$ strings bomb
/lib64/ld-linux-x86-64.so.2
4"R/R
libc.so.6
socket
fflush
strcpy
__printf_chk
exit
fopen
__isoc99_sscanf
connect
signal
puts
__stack_chk_fail
stdin
strtol
fgets
__errno_location
read
__fprintf_chk
stdout
__memmove_chk
__ctype_b_loc
getenv
stderr
alarm
gethostbyname
close
sleep
__sprintf_chk
__libc_start_main
write
__gmon_start__
GLIBC_2.3
GLIBC_2.7
GLIBC_2.3.4
GLIBC_2.4
GLIBC_2.3.5
```

Step 2) The GNU debugger will enable the programmer to trace the program line by line, examine memory and registers, look at both the source code and assembly code.



```
lab2-34@lab234-OptiPlex-3040:~/Desktop/Bomb_Lab/bomb001$ gdb bomb
GNU gdb (Ubuntu 9.2-0ubuntu1~20.04) 9.2
Copyright (C) 2020 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 "x86_64-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 bomb...
(gdb) b phase_1
Breakpoint 1 at 0x400e8d
(gdb) run
Starting program: /home/lab2-34/Desktop/Bomb_Lab/bomb001/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Test String

Breakpoint 1, 0x00000000400e8d in phase_1 ()
(gdb) disas
Dump of assembler code for function phase_1:
=> 0x00000000400e8d <+0>: sub $0x8,%rsp
0x00000000400e91 <+4>: mov $0x4023d0,%esi
0x00000000400e96 <+9>: callq 0x40133e <strings_not_equal>
0x00000000400e9b <+14>: test %eax,%eax
0x00000000400e9d <+16>: je 0x400ea4 <phase_1+23>
0x00000000400e9f <+18>: callq 0x40143d <explode_bomb>
0x00000000400ea4 <+23>: add $0x8,%rsp
0x00000000400ea8 <+27>: retq
End of assembler dump.
(gdb) u* 0x00000000400ea8
```

Step 3) Set the breakpoints on <phase 1> function so that to enable the programmer to diffuse the bomb set at phase 1. This will ensure that the bomb doesn't blow up when we run the program with the compiler. Then run the program with run command.

```
Terminal
lab2-34@lab234-OptiPlex-3040: ~/Desktop/Bomb_Lab/bomb001

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from bomb...
(gdb) b phase_1
Breakpoint 1 at 0x400e8d
(gdb) run
```

Step 4) After the run command, the programmer has to enter string and here the “ Test String ” is entered.

The disas command is shorthand for disassemble. This command will compare two strings, one the strings input by user and another is the default string of computer system.

```
Starting program: /home/lab2-34/Desktop/Bomb_Lab/bomb001/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Test String

Breakpoint 1, 0x00000000400e8d in phase_1 ()
(gdb) disas
Dump of assembler code for function phase_1:
=> 0x00000000400e8d <+0>:      sub    $0x8,%rsp
0x00000000400e91 <+4>:      mov     $0x4023d0,%esi
0x00000000400e96 <+9>:      callq  0x40133e <strings_not_equal>
0x00000000400e9b <+14>:     test   %eax,%eax
0x00000000400e9d <+16>:     je      0x400ea4 <phase_1+23>
0x00000000400e9f <+18>:     callq  0x40143d <explode_bomb>
0x00000000400ea4 <+23>:     add     $0x8,%rsp
0x00000000400ea8 <+27>:     retq
End of assembler dump.
```

Step 5) If the string input by user and the value stored at register %esi doesn't match then the bomb set at phase 1 will explode during execution.

```
(gdb) u* 0x00000000400ea8
BOOM!!!
The bomb has blown up.
[Inferior 1 (process 38003) exited with code 010]
(gdb)
```

Step 6) To find out the real value stored at %esi, the programmer has to type the command x/s followed by its address \$0x4023d0.

```
BOOM!!!
The bomb has blown up.
[Inferior 1 (process 38003) exited with code 010]
(gdb) run
Starting program: /home/user/Downloads/Assignment 1_2/Assignment 1/bomb001/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Test String

Breakpoint 1, 0x0000000000400e8d in phase_1 ()
(gdb) disas
Dump of assembler code for function phase_1:
=> 0x0000000000400e8d <+0>:      sub     $0x8,%rsp
0x0000000000400e91 <+4>:      mov     $0x4023d0,%esi
0x0000000000400e96 <+9>:      callq  0x40133e <strings_not_equal>
0x0000000000400e9b <+14>:     test    %eax,%eax
0x0000000000400e9d <+16>:     je      0x400ea4 <phase_1+23>
0x0000000000400e9f <+18>:     callq  0x40143d <explode_bomb>
0x0000000000400ea4 <+23>:     add     $0x8,%rsp
0x0000000000400ea8 <+27>:     retq

End of assembler dump.
(gdb) x/s 0x4023d0
0x4023d0:      "The moon unit will be divided into two divisions."
```

Step 7) After that run the program again and input the string that we got earlier.

```
user@user-OptiPlex-3040: ~/Downloads/Assignment 1_2/As...
0x4023d0: "The moon unit will be divided into two divisions."
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/user/Downloads/Assignment 1_2/Assignment 1/bomb001/bomb

Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
The moon unit will be divided into two divisions.

Breakpoint 1, 0x000000000400e8d in phase_1 ()
```

Step 8) Execute the program until the end and check weather the bomb explodes or not.

Since we tracked the stored string at \$esi register and input the similar string the bomb will be diffused.

```
(gdb) u* 0x0000000000400ea8
main (argc=<optimized out>, argv=<optimized out>) at bomb.c:75
75         phase_defused();                /* Drat!  They figured it out!
(gdb) □
```

Step 9) Now the bomb set at phase 1 is finally diffused.

```
user@user-OptiPlex-3040:~/Downloads/Assignment 1_2/Assignment 1/bomb001$ ./bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
The moon unit will be divided into two divisions.
Phase 1 defused. How about the next one?
█
```