

IT16152342

R.M.A.D.Rathnayaka.

4<sup>th</sup> year June batch

---

## Bigbang Theory

### Sheldon 1

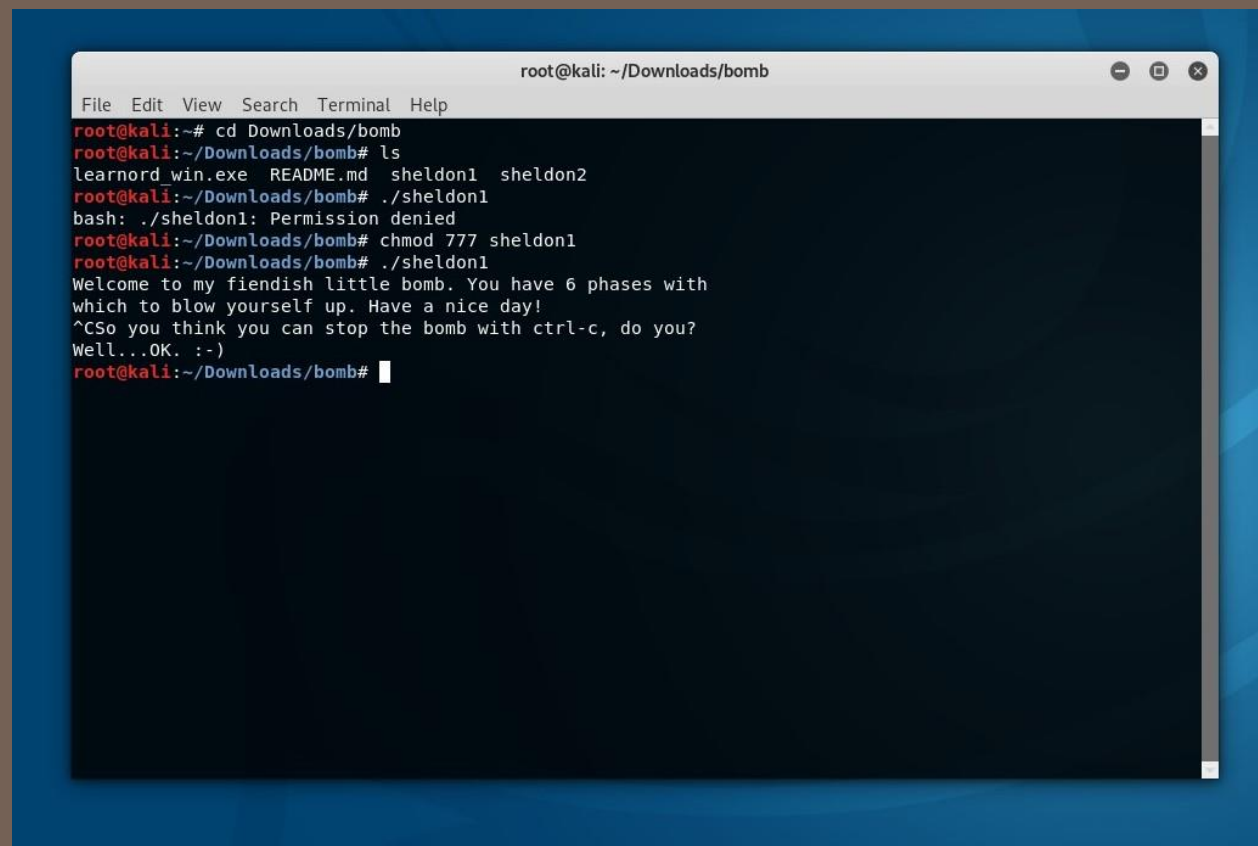
#### Phase 1

Defusing procedure must be performed in 6 phases.

First, we need to create a controlled bomb blasting environment to avoid damage coverage.

When we test our inputs using debugger.

Using GDB debugger we can open sheldon1.



```
root@kali: ~/Downloads/bomb
File Edit View Search Terminal Help
root@kali:~# cd Downloads/bomb
root@kali:~/Downloads/bomb# ls
learnord_win.exe README.md sheldon1 sheldon2
root@kali:~/Downloads/bomb# ./sheldon1
bash: ./sheldon1: Permission denied
root@kali:~/Downloads/bomb# chmod 777 sheldon1
root@kali:~/Downloads/bomb# ./sheldon1
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
^CSo you think you can stop the bomb with ctrl-c, do you?
Well...OK. :-)
```

```
root@kali: ~/Downloads/bomb

File Edit View Search Terminal Help
All defined functions:

File bomb.c:
36:  int main(int, char **);

Non-debugging symbols:
0x080486e0  _init
0x08048720  __register_frame_info@plt
0x08048730  close@plt
0x08048740  fprintf@plt
0x08048750  tmpfile@plt
0x08048760  getenv@plt
0x08048770  signal
0x08048770  signal@plt
0x08048780  fflush
0x08048780  fflush@plt
0x08048790  bcopy
0x08048790  bcopy@plt
0x080487a0  rewind
0x080487a0  rewind@plt
0x080487b0  system
0x080487b0  system@plt
0x080487c0  __deregister_frame_info
0x080487c0  __deregister_frame_info@plt
0x080487d0  fgets
0x080487d0  fgets@plt
0x080487e0  sleep
0x080487e0  sleep@plt
0x080487f0  __strtol_internal
--Type <RET> for more, q to quit, c to continue without paging--
```

We need to check the available function. Then we need to check the main function.

```
root@kali: ~/Downloads/bomb

File Edit View Search Terminal Help
0x08048a23 <+115>: push    $0x8
0x08048a25 <+117>: call   0x08048850 <exit@plt>
0x08048a2a <+122>: lea    0x0(%esi,%esi)
0x08048a30 <+128>: call   0x08049160 <initialize_bomb>
0x08048a35 <+133>: add    $0xffffffff4,%esp
0x08048a38 <+136>: push   $0x8049660
0x08048a3d <+141>: call   0x08048910 <printf@plt>
0x08048a42 <+146>: add    $0xffffffff4,%esp
0x08048a45 <+149>: push   $0x80496a0
0x08048a4a <+154>: call   0x08048810 <printf@plt>
0x08048a4f <+159>: add    $0x20,%esp
0x08048a52 <+162>: call   0x080491fc <read_line>
0x08048a57 <+167>: add    $0xffffffff4,%esp
0x08048a5a <+170>: push   %eax
0x08048a5b <+171>: call   0x08048b20 <phase_1>
0x08048a60 <+176>: call   0x0804952c <phase_defused>
0x08048a65 <+181>: add    $0xffffffff4,%esp
0x08048a68 <+184>: push   $0x80496e0
0x08048a6d <+189>: call   0x08048810 <printf@plt>
0x08048a72 <+194>: add    $0x20,%esp
0x08048a75 <+197>: call   0x080491fc <read_line>
0x08048a7a <+202>: add    $0xffffffff4,%esp
0x08048a7d <+205>: push   %eax
0x08048a7e <+206>: call   0x08048b48 <phase_2>
0x08048a83 <+211>: call   0x0804952c <phase_defused>
0x08048a88 <+216>: add    $0xffffffff4,%esp
0x08048a8b <+219>: push   $0x8049720
0x08048a90 <+224>: call   0x08048810 <printf@plt>
0x08048a95 <+229>: add    $0x20,%esp
0x08048a98 <+232>: call   0x080491fc <read_line>
0x08048a9d <+237>: add    $0xffffffff4,%esp
0x08048aa0 <+240>: push   %eax
0x08048aa1 <+241>: call   0x08048b98 <phase_3>
0x08048aa6 <+246>: call   0x0804952c <phase_defused>
0x08048aab <+251>: add    $0xffffffff4,%esp
0x08048aae <+254>: push   $0x804973f
0x08048ab3 <+259>: call   0x08048810 <printf@plt>
0x08048ab6 <+264>: add    $0x20,%esp
```

Then we can get some ideas checking above details.

When the program runs we can enter the pass phrase and grant access to the next phase.

The program calls the relevant phase function upon the correct phase phrase.

Then check the phase\_1 function.

```
0x08048af2 <+321>: add    $0xffffffff,%esp
0x08048af4 <+324>: push   $0x80497a0
0x08048af9 <+329>: call   0x8048810 <printf@plt>
0x08048afe <+334>: add    $0x20,%esp
0x08048b01 <+337>: call   0x80491fc <read_line>
0x08048b06 <+342>: add    $0xffffffff4,%esp
0x08048b09 <+345>: push   %eax
0x08048b0a <+346>: call   0x8048d98 <phase_6>
0x08048b0f <+351>: call   0x804952c <phase_defused>
0x08048b14 <+356>: xor    %eax,%eax
0x08048b16 <+358>: mov    -0x18(%ebp),%ebx
0x08048b19 <+361>: mov    %ebp,%esp
0x08048b1b <+363>: pop    %ebp
0x08048b1c <+364>: ret
End of assembler dump.
(gdb) disass <phase_1>
A syntax error in expression, near `<phase_1>'.
(gdb) disass phase_1
Dump of assembler code for function phase_1:
0x08048b20 <+0>: push   %ebp
0x08048b21 <+1>: mov    %esp,%ebp
0x08048b23 <+3>: sub    $0x8,%esp
0x08048b26 <+6>: mov    0x8(%ebp),%eax
0x08048b29 <+9>: add    $0xffffffff8,%esp
0x08048b2c <+12>: push   $0x80497c0
0x08048b31 <+17>: push   %eax
0x08048b32 <+18>: call   0x8049030 <strings_not_equal>
0x08048b37 <+23>: add    $0x10,%esp
0x08048b3a <+26>: test   %eax,%eax
0x08048b3c <+28>: je     0x8048b43 <phase_1+35>
0x08048b3e <+30>: call   0x80494fc <explode_bomb>
0x08048b43 <+35>: mov    %ebp,%esp
0x08048b45 <+37>: pop    %ebp
0x08048b46 <+38>: ret
End of assembler dump.
(gdb)
```

We have a fixed address pushed to the memory and then we got a TEST instruction. \$0x80497c0

In that memory address we can print out the first 30 characters.

```

File Edit View Search Terminal Help
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 sheldon1...
(gdb) disass phase_1

```

```

Dump of assembler code for function phase_1:
0x08048b20 <+0>:    push    %ebp
0x08048b21 <+1>:    mov     %esp,%ebp
0x08048b23 <+3>:    sub     $0x8,%esp
0x08048b26 <+6>:    mov     0x8(%ebp),%eax
0x08048b29 <+9>:    add     $0xffffffff,%esp
0x08048b2c <+12>:   push    $0x80497c0
0x08048b31 <+17>:   push    %eax
0x08048b32 <+18>:   call    0x8049030 <strings_not_equal>
0x08048b37 <+23>:   add     $0x10,%esp
0x08048b3a <+26>:   test    %eax,%eax
0x08048b3c <+28>:   je      0x8048b43 <phase_1+35>
0x08048b3e <+30>:   call    0x80494fc <explode_bomb>
0x08048b43 <+35>:   mov     %ebp,%esp
0x08048b45 <+37>:   pop     %ebp
0x08048b46 <+38>:   ret

```

```

End of assembler dump.
(gdb) x/30c 0x80497c0
0x80497c0:  80 'P' 117 'u' 98 'b' 108 'l' 105 'i' 99 'c' 32 ' ' 115 's'
0x80497c8:  112 'p' 101 'e' 97 'a' 107 'k' 105 'i' 110 'n' 103 'g' 32 ' '
0x80497d0:  105 'i' 115 's' 32 ' ' 118 'v' 101 'e' 114 'r' 121 'y' 32 ' '
0x80497d8:  101 'e' 97 'a' 115 's' 121 'y' 46 '.' 0 '\000'
(gdb)

```

We have a phrase. But we still not sure this is the right thing to defuse the bomb,

Then we can use gdb to run the program to get the string and therefore we can test this is the right phrase.

```
root@kali: ~/Downloads/bomb
File Edit View Search Terminal Help
cuserid
fprintf
__deregister_frame_info
stdin
signal
bcopy
sscanf
gethostbyname
sprintf
fclose
rewind
exit
fopen
_IO_stdin_used
__libc_start_main
__register_frame_info
close
GLIBC_2.1
GLIBC_2.0
PTRh
LWVS
\WVS
nobo
%s: Error: Couldn't open %s
Usage: %s [<input_file>]
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
That's number 2. Keep going!
Halfway there!
So you got that one. Try this one.
Good work! On to the next...
Public speaking is very easy.
%d %c %d
giants
Wow! You've defused the secret stage!
whitefish.cmcl.cs.cmu.edu
warmouth.cmcl.cs.cmu.edu
```

```

root@kali: ~/Downloads/bomb
File Edit View Search Terminal Help
connect@@GLIBC_2.0
stdin@@GLIBC_2.0
fopen@@GLIBC_2.1
dup@@GLIBC_2.0
_IO_stdin_used
sprintf@@GLIBC_2.0
_data_start
socket@@GLIBC_2.0
phase_1
skip
node2
cuserid@@GLIBC_2.0
_gmon_start
strcpy@@GLIBC_2.0
root@kali:~/Downloads/bomb# gdb sheldon1
GNU gdb (Debian 8.3.1-1) 8.3.1
Copyright (C) 2019 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 sheldon1...
(gdb) r
Starting program: /root/Downloads/bomb/sheldon1
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Public speaking is very easy.
Phase 1 defused. How about the next one?

```

Finally, we defused the first bomb!