CA Lab 5

Rahul Katinni

S20200010091

Phase 1 :

To diffuse the 1st phase of the bomb I used strings bomb command to view all the strings present in the bomb and searched for any abnormal string that could be the key to the 1st phase.

I found the key to my phase 1 to be

"`I turned the moon into something I call a Death Star.`"

```
rahul@ubuntu-lappy: ~/Desktop/Bomb_lab
E,$$
t)APRA
AWAVA
AUATL
[]A\A]A^A_
%s: Error: Couldn't open %s
Usage: %s [<input_file>]
That's number 2.  Keep going!
Halfway there!
Good work!  On to the next...
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?
So you got that one.  Try this one.
I turned the moon into something I call a Death Star.
Wow! You've defused the secret stage!
So you think you can stop the bomb with ctrl-c, do you?
Curses, you've found the secret phase!
But finding it and solving it are quite different...
Congratulations! You've defused the bomb!
Well...
OK. :-)
Invalid phase%s
BOOM!!!
The bomb has blown up.
%d %d %d %d %d %d
Error: Premature EOF on stdin
GRADE_BOMB
Error: Input line too long
%d %d %s
DrEvil
greatwhite.ics.cs.cmu.edu
angelshark.ics.cs.cmu.edu
makoshark.ics.cs.cmu.edu
whaleshark.ics.cs.cmu.edu
Program timed out after %d seconds
Error: HTTP request failed with error %d: %s
GET /%s/submitr.pl/?userid=%s&userpwd=%s&lab=%s&result=%s&submit=submit HTTP/1.0
Error: Unable to connect to server %s
%%%02X
%s %d %[a-zA-z ]
changeme.ics.cs.cmu.edu
AUTORESULT_STRING=%s
csapp
;*3$"
GCC: (Ubuntu 5.4.0-6ubuntu1~16.04.12) 5.4.0 20160609
<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...
/usr/include/x86_64-linux-gnu/bits
/usr/lib/gcc/x86_64-linux-gnu/5/include
/usr/include
```

```
rahul@ubuntu-lappy: ~/Desktop/Bomb_lab
rahul@ubuntu-lappy:~/Desktop/Bomb_lab$ 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_2
Breakpoint 1 at 0x400ea9
(gdb) b phase_3
Breakpoint 2 at 0x400f11
(gdb) b phase_4
Breakpoint 3 at 0x400ff3
(gdb) b phase_5
Breakpoint 4 at 0x401060
(gdb) b phase_6
Breakpoint 5 at 0x4010ec
(gdb) i b
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x0000000000400ea9 <phase_2>
2       breakpoint     keep y   0x0000000000400f11 <phase_3>
3       breakpoint     keep y   0x0000000000400ff3 <phase_4>
4       breakpoint     keep y   0x0000000000401060 <phase_5>
5       breakpoint     keep y   0x00000000004010ec <phase_6>
(gdb) r
Starting program: /home/rahul/Desktop/Bomb_lab/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
I turned the moon into something I call a Death Star.
Phase 1 defused. How about the next one?
```

Phase 2 :

For deciphering phase 2, I debugged the assembly code step by step and found that my 1st number should be positive and the

logic for the remaining five number to be as follows :

Let the numbers be a,b,c,d,e,f.

The logic is :

1st number a is any positive number.

Then b = a + 1

c = b + 2

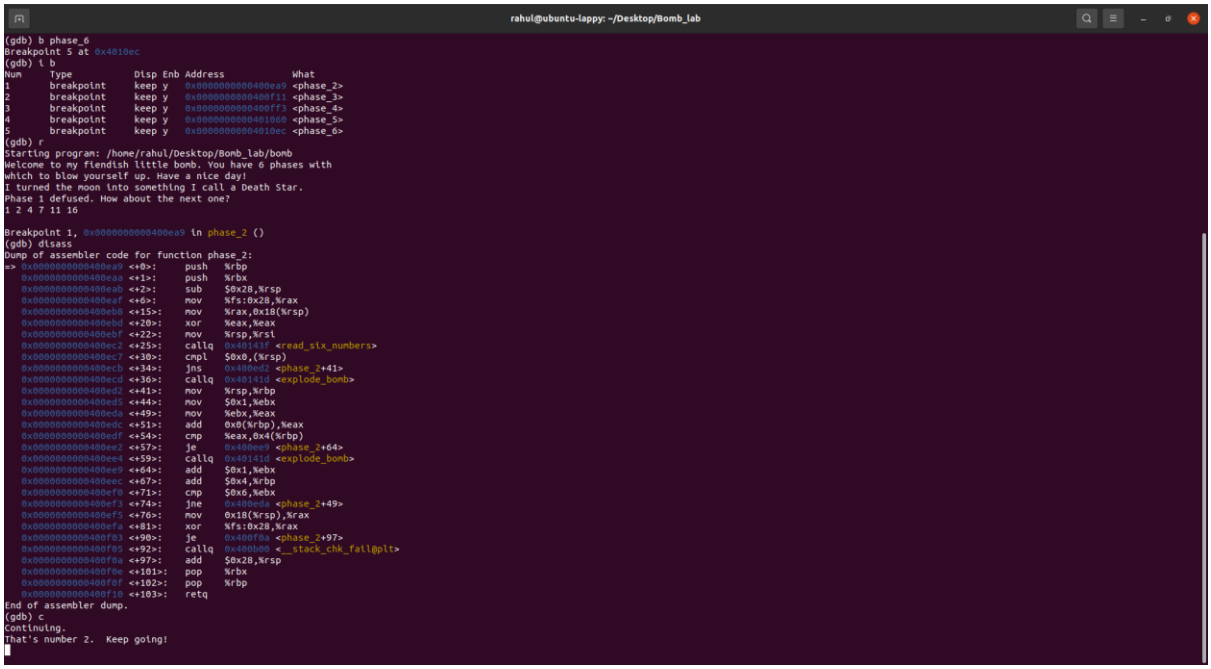
d = c + 3

e = d + 4

f = e + 5

Therefore taking 1st number as 1 the key to the 2nd phase is :

“ 1 , 2 , 4 , 7 , 11 , 16 ”

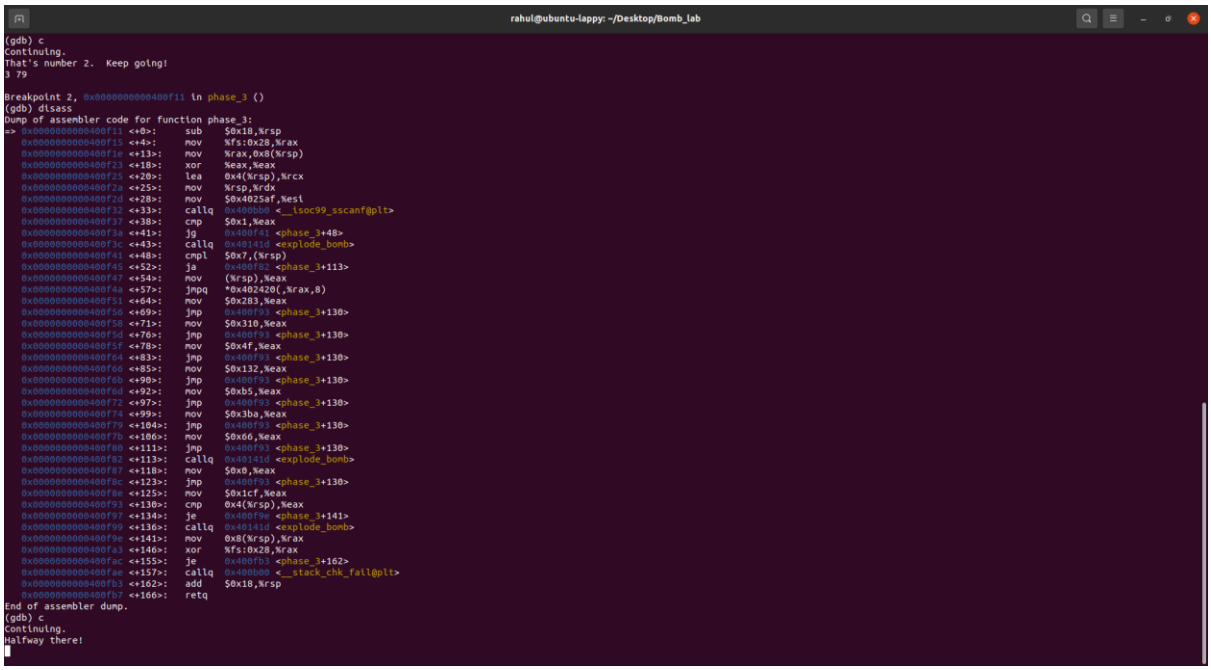


Phase 3 :

For phase 3 after looking at the assembly code I found that the 1st number should be less than 7 and to find the 2nd number I took a random input and ran it through the code step by step and at the final comparison I extracted the correct input from the registers.

The key I got for phase 3 is :

“ 3 , 79 ”

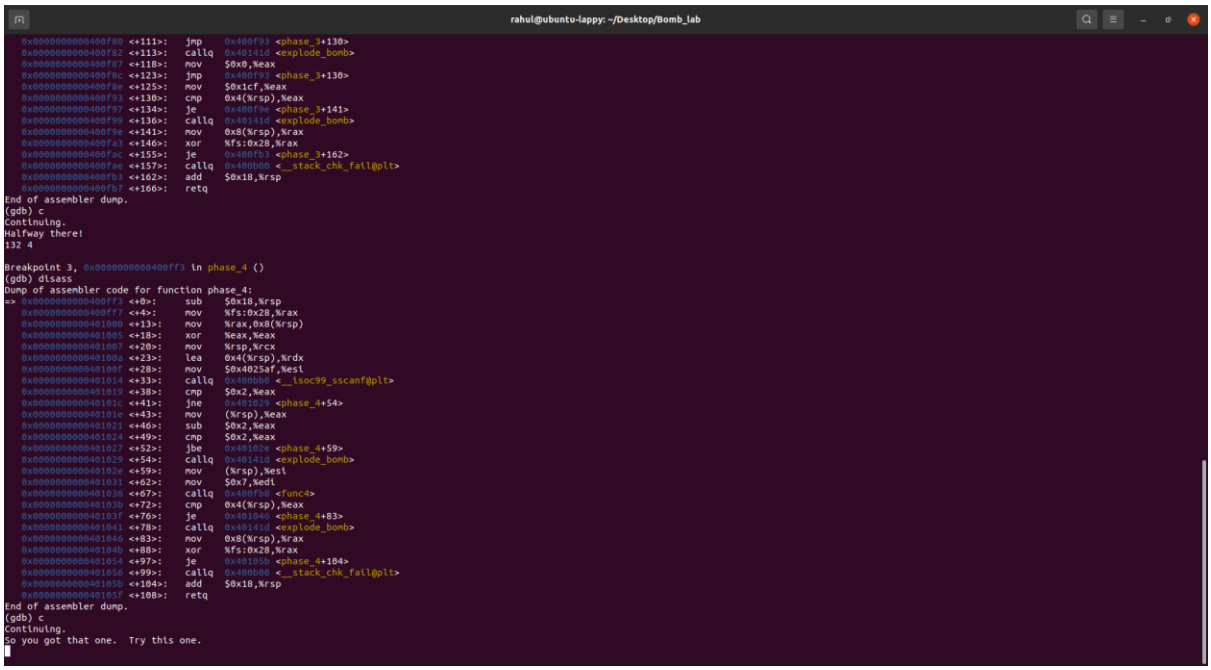


Phase 4 :

I have done phase 4 also in a similar way as phase 3 I took 2 random number as input and ran them through the code and found the correct replacement for one of them in the registers in the final comparison step after the fun4 function in phase 4 was executed.

The key I got for phase 4 is :

“132 , 4”



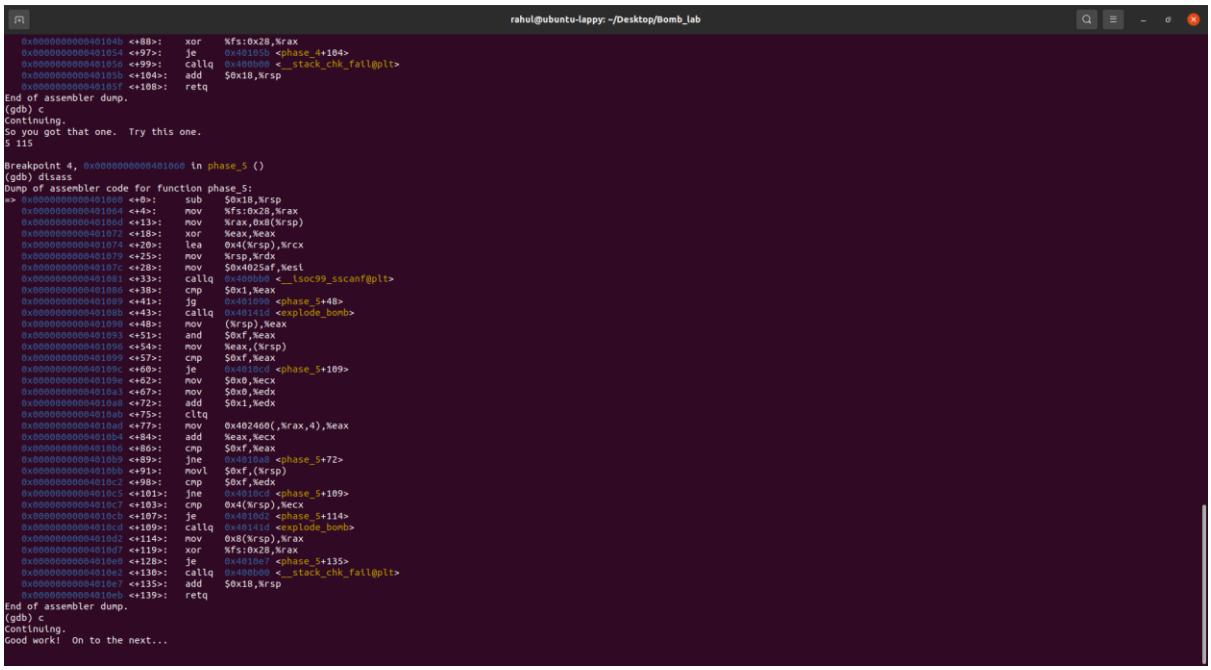
Phase 5 :

In phase 5 there is an hardcoded array which we traverse it in such a way that the element in the current index is taken to be

the next index to be searched. The 1st number of the key is the index at which we start searching for 15 such that it is found on the 15th iteration and the 2nd number is the sum of all the elements of the array which we visited in the process.

The key I found for phase 5 is :

“ 5 , 115 ”



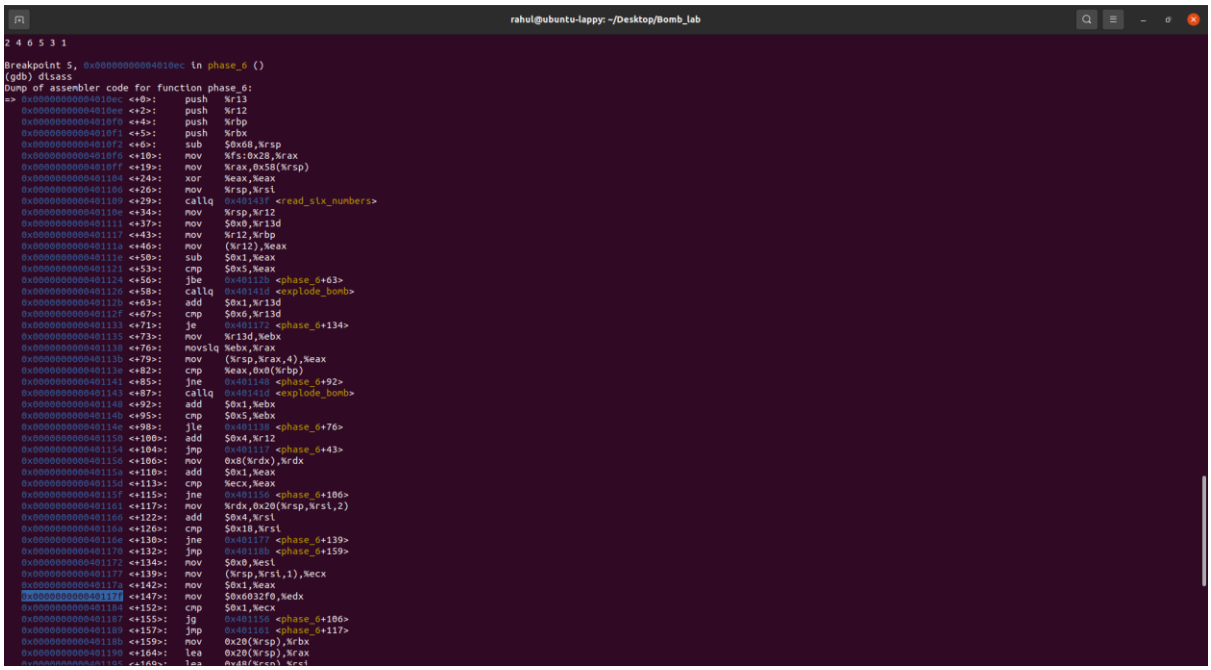
Phase 6 :

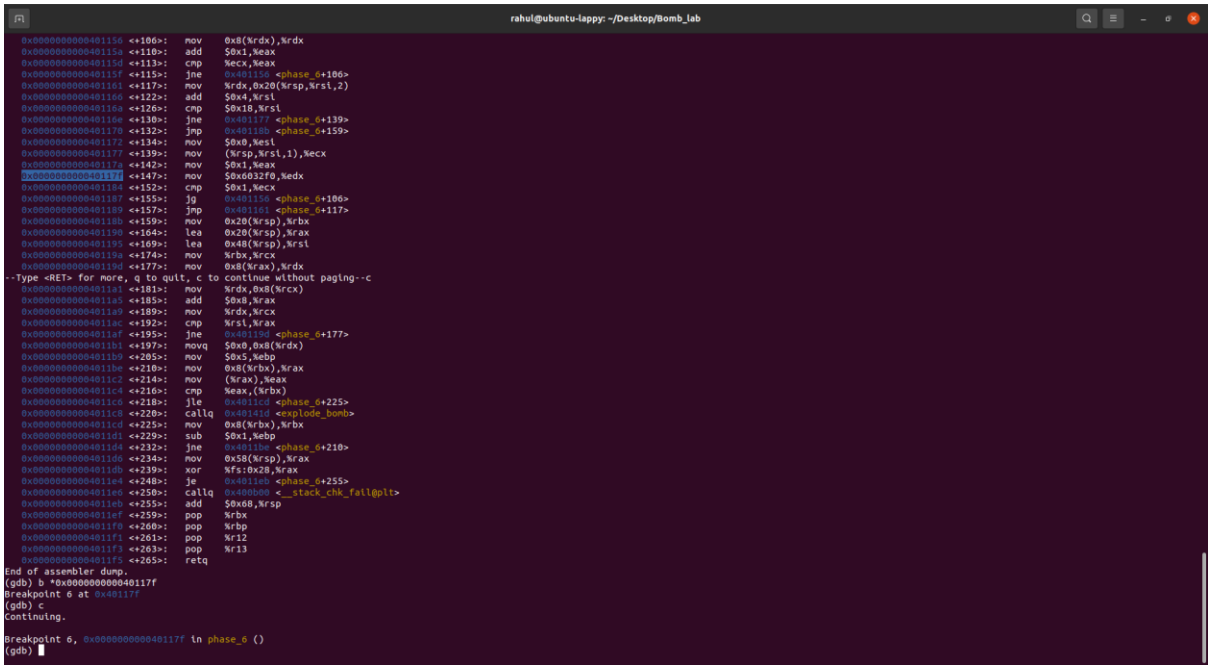
In phase 6 we enter the node numbers of a linked list in ascending order which is hardcoded, the bomb 1st checks that all the

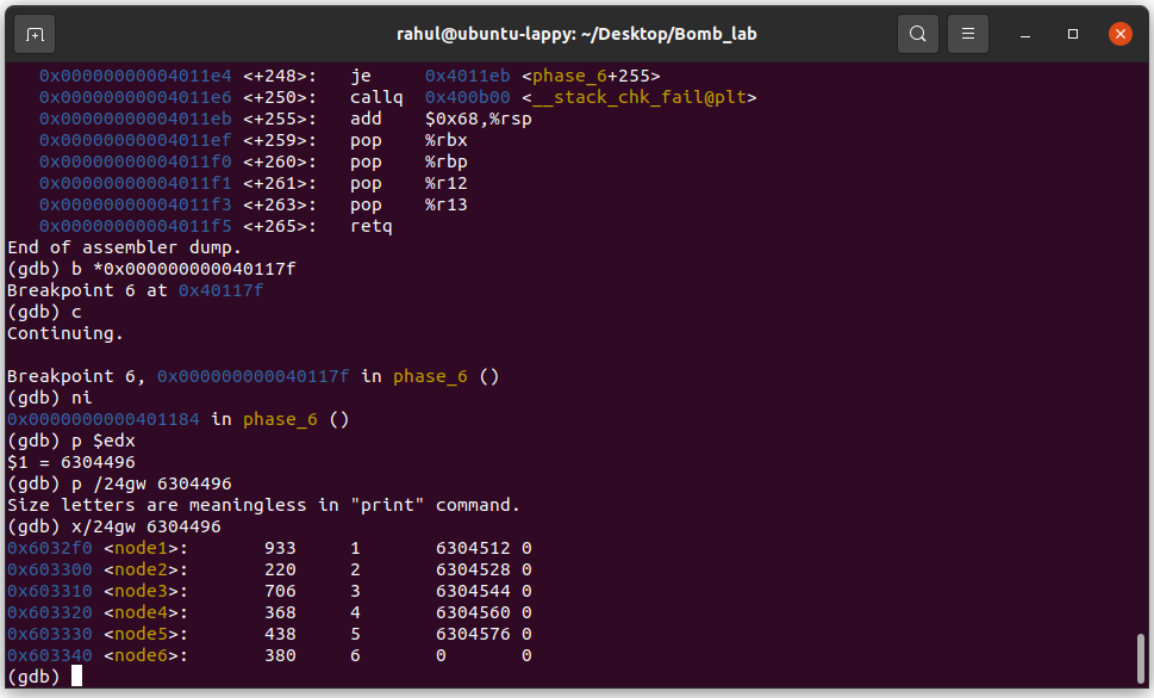
number are between 1 and 6 and that all are different using nested loops. Then the bomb sorts the linked list to be in ascending order.

The key I found for phase 6 Is :

“ 2 , 4 , 6 , 5 , 3 , 1 ”







After finding the keys of all the phases I defused the bomb successfully.

```
rahul@ubuntu-lappy: ~/Desktop/Bomb_lab
rahul@ubuntu-lappy:~/Desktop/Bomb_lab$ ./bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
I turned the moon into something I call a Death Star.
Phase 1 defused. How about the next one?
1 2 4 7 11 16
That's number 2.  Keep going!
3 79
Halfway there!
132 4
So you got that one.  Try this one.
5 115
Good work!  On to the next...
2 4 6 5 3 1
Congratulations! You've defused the bomb!
rahul@ubuntu-lappy:~/Desktop/Bomb_lab$
```