IE4012
Offensive Hacking Tactical and Strategic

4th Year, 1st Semester

BigBangTheory Sheldon1 Phase_1

Submitted to

Sri Lanka Institute of Information Technology

In partial fulfillment of the requirements for the

Bachelor of Science Special Honors Degree in Information Technology

08/03/2020

Declaration

I certify that this report does not incorporate without acknowledgement, any material previously submitted for a degree or diploma in any university, and to the best of my knowledge and belief it does not contain any material previously published or written by another person, except where due reference is made in text.

Registration Number : IT17037198

Name : J.A.Y.N Jayasinghe

List of Figures

BigBangTheory Sheldon1 Phase_1

When running the program “sheldon1”, it mentions that we have 6 phases to defuse a bomb. The program will continue only when we input a string. When entering “111111” as shown in Figure 1 below, the program terminates after displaying a message saying that the bomb has blown up.

```
root@kali:~/Documents/bigbangtheory-master# ./sheldon1
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
111111

BOOM!!!
The bomb has blown up.
root@kali:~/Documents/bigbangtheory-master#
```

Figure 1 : Running sheldon1 for the first time

To understand what we must do, let us first disassemble the main function using gdb.

```
0x08048a52 <+162>:   call   0x80491fc <read_line>
0x08048a57 <+167>:   add    esp,0xfffffff4
0x08048a5a <+170>:   push   eax
0x08048a5b <+171>:   call   0x8048b20 <phase_1>
0x08048a60 <+176>:   call   0x804952c <phase_defused>
0x08048a65 <+181>:   add    esp,0xfffffff4
0x08048a68 <+184>:   push   0x80496e0
0x08048a6d <+189>:   call   0x8048810 <printf@plt>
0x08048a72 <+194>:   add    esp,0x20
0x08048a75 <+197>:   call   0x80491fc <read_line>
0x08048a7a <+202>:   add    esp,0xfffffff4
0x08048a7d <+205>:   push   eax
```

Figure 2 : Disassembling the Main function

As shown in Figure 2 above, the first function called after the bomb is initialized is called “phase_1”. By disassembling phase_1 we can see what it contains as shown in Figure 3 below.

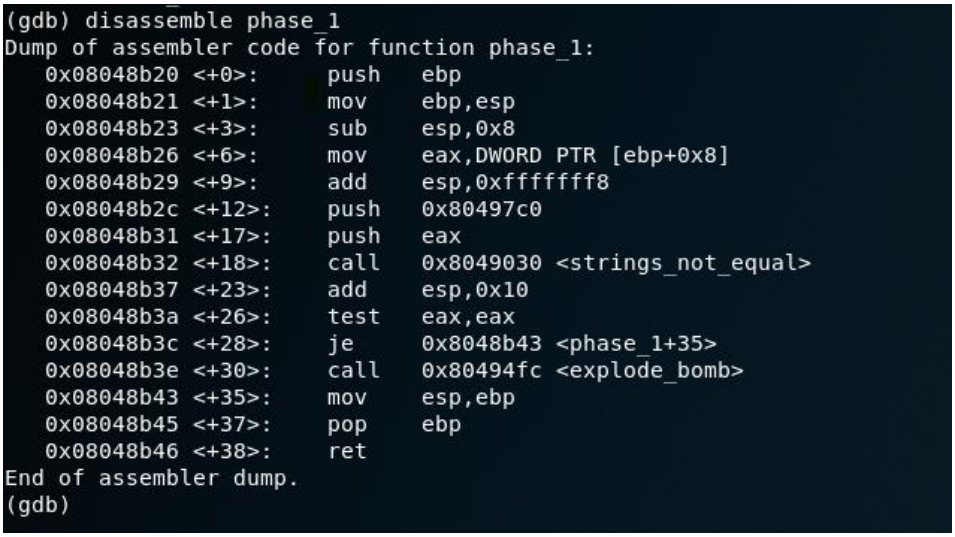

Figure 3 : Disassembling Phase_1

By analyzing the assembly code of phase_1 we can see that two things are being pushed into the stack: a memory location 0x80497c0 and register eax. Directly after that a function called “strings_not_equal” is called. We can assume that the strings in the memory location and the register are being compared in some way.

Let us set a breakpoint at phase_1 and run the program for further tests. This time, when entering a random string “abcdefg”, the program is halted by the breakpoint before the bomb explodes as shown in Figure 4.

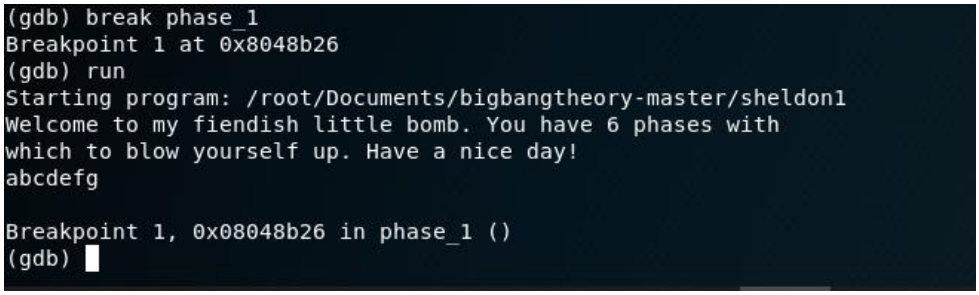

Figure 4 : Inserting a breakpoint at phase_1 and running the program

Now that we have halted the program, we can see what are the strings that are stored at the pushed memory location and register. We can use the command “x /20c $eax” to display the first 20 characters after the memory address of register $eax (Figure 5).

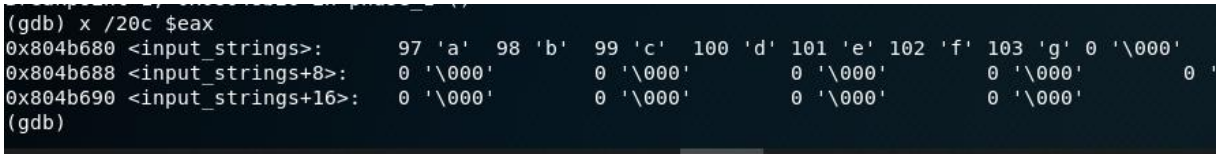

Figure 5 : Displaying the string stored in register eax

As shown in the above figure, we can see that register eax contains the string “abcdefg” that we entered. With this we can conclude that this phase must want us to enter the correct string, which must be stored in the other pushed memory location 0x80497c0. Let us see what the first 20 characters after it contains using “x /20c 0x80497c0”.

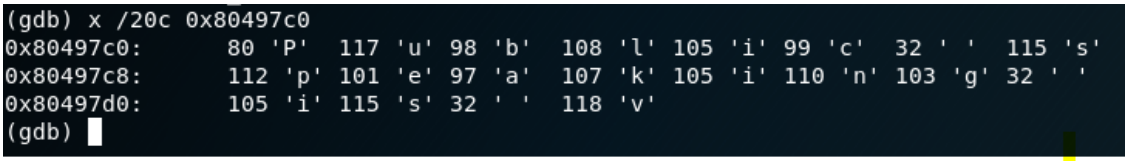

Figure 6 : Displaying the string stored in the fixed memory location

As shown in figure 6 above, we can see that the first 20 characters of this memory location contains “Public speaking is v”. This means there are more characters to this string, we can use the same command but with more characters to display the entire string, but we can extract this string directly using the strings command combined with grep to search for the string using the keywords “Public speaking”.

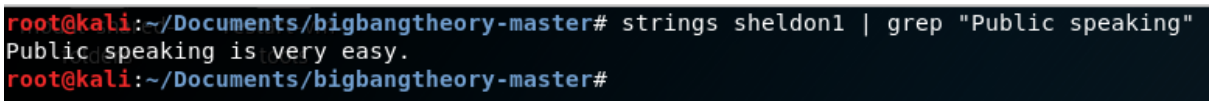

Figure 7 : Using the Strings command to display the string

As shown in Figure 7 above, sheldon1 contains only one such string, which is “Public speaking is very easy.”. This must be the string we must enter to pass phase_1. Running the program and entering this string will successfully pass us through phase1 as shown in Figure 8 below.

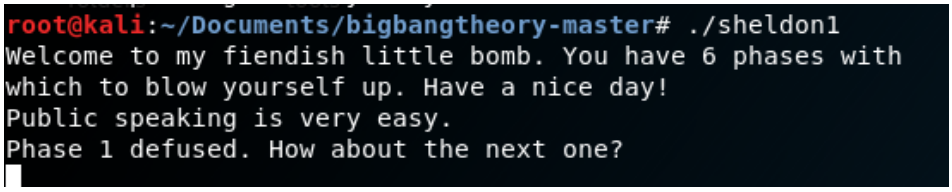

Figure 8 : Passing the first phase