

PHASE_1

Phase_1 : finding the hidden string.

Assembly code for phase_1

```
dorji@dorji-TravelMate-P214-53:~/Downloads/Assignment 1/bomb003$ gdb bomb
GNU gdb (Ubuntu 10.1-2ubuntu2) 10.1.90.20210411-git
Copyright (C) 2021 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:
<https://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) disas phase_1
Dump of assembler code for function phase_1:
   0x0000000000400e8d <+0>:      sub    $0x8,%rsp
   0x0000000000400e91 <+4>:      mov    $0x4023b0,%esi
   0x0000000000400e96 <+9>:      call   0x40132b <strings_not_equal>
   0x0000000000400e9b <+14>:     test   %eax,%eax
   0x0000000000400e9d <+16>:     je     0x400ea4 <phase_1+23>
   0x0000000000400e9f <+18>:     call   0x40142a <explode_bomb>
   0x0000000000400ea4 <+23>:     add    $0x8,%rsp
   0x0000000000400ea8 <+27>:     ret
```

Set break point to avoid the bombs from getting explode and run to test the string if the string doesn't match the bomb will not explode because breaks point are set before hand in order to prevent the explode of bomb.

```
(gdb) break phase_1
Breakpoint 1 at 0x400e8d
(gdb) break explode_bomb
Breakpoint 2 at 0x40142a
(gdb) run
Starting program: /home/dorji/Downloads/Assignment 1/bomb003/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
string text

Breakpoint 1, 0x0000000000400e8d in phase_1 ()
(gdb) disas
```

Given the string “String test ” the code compare the input string with the system string and if the two string are equal phase 1 gets defused, if the string are not equal then the bomb explodes.

```
(gdb) disas
Dump of assembler code for function phase_1:
=> 0x0000000000400e8d <+0>:      sub    $0x8,%rsp
    0x0000000000400e91 <+4>:      mov    $0x4023b0,%esi
    0x0000000000400e96 <+9>:      call   0x40132b <strings_not_equal>
    0x0000000000400e9b <+14>:     test   %eax,%eax
    0x0000000000400e9d <+16>:     je     0x400ea4 <phase_1+23>
    0x0000000000400e9f <+18>:     call   0x40142a <explode_bomb>
    0x0000000000400ea4 <+23>:     add    $0x8,%rsp
    0x0000000000400ea8 <+27>:     ret
End of assembler dump.
(gdb) x/s 0x4023b0
0x4023b0: "Border relations with Canada have never been better."
(gdb) delete
Delete all breakpoints? (y or n) y
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/dorji/Downloads/Assignment 1/bomb003/bomb
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Border relations with Canada have never been better.
Phase 1 defused. How about the next one?
```

To get the hidden string, code must be executed line by line and we have to check what else are their in each registers. After the execution we found out that phase_1’s hidden string is inside the memory address “0x4023b0” . To get string inside the address use “x/s” command.

Hidden String for phase_1 is “Border relations with Canada have never been better.”

After getting the hidden string remove all the break point and run it.