自由研究專題II

經濟二 407510046 黃郁雯

問題一

怎麼編譯成gdb可以debug的形式？

先在terminal輸入`vim test.c`，開啟檔案後將題目敘述的程式輸入後按`:wq`儲存並退出。再在terminal輸入`gcc -g -o test test.c`(gcc：程式名/指令名; -g：參數 ，可讓gdb被開啟 ;-o：指定檔名用的參數 ;test：-o需要的檔名; test.c：檔案)就可以debug了。

問題二

上述程式哪個地方發生錯誤？你怎麼用gdb找出來的？知道原因並修復的？

1. 程式的第六行：`scanf("%d",input);`發生錯誤。
2. 在terminal輸入`gcc -g -o test test.c`後，按enter鍵。
3. 按下enter鍵後，便會出現以下敘述。 因此便會知道第六行的第十一個發生錯誤。

```
test.c: In function ‘get_num’:
test.c:6:11: warning: format ‘%d’ expects argument of type ‘int *’, but argument
2 has type ‘int’ [-Wformat=]
   scanf("%d",input);
          ~^
```

4.所以發生錯誤的地方是：`scanf("%d",input);`
修正之後為：`scanf("%d",&input);`

問題三

那些情況可能會觸發segmentation fault，是一定會觸發還是不一定，為何？

1. segmentation fault 簡而言之就是，當程式嘗試訪問不允許訪問的存儲位置或試圖以不允許的方式訪問存儲位置（例如，嘗試寫入只讀位置時，或覆蓋部分操作系統），就會發生分段錯誤。
2. 不一定，有時棧溢不一定會觸發segmentation fault。當編譯程式時看到“ warning: function returns address of local variable ”，GCC 已經在警告棧溢的可能了。實際運行結果一切正常。原因是操作系統通常以“頁”的粒度來管理內存，Linux 中典型的頁大小為4K ，內核為segmentation fault進程棧分配內存也是以4K 為粒度的。故當棧溢的幅度小於頁的大小時，不會產生segmentation fault 。但是不一定溢出超過4K就會產生segmentation fault，也有可能是因為無效的棧內存（即棧指針範圍外未被回收的棧內存），這個是由操作系統在需要時回收的，這是無法預測的，也就無法預測何時訪問非法的棧內容會引發segmentation fault。
   通常發生segmentation fault的原因：

   01.訪問已釋放的地址
   02.修改字符串文字
   03.訪問超出數組的索引範圍
   04.訪問不存在的內容
   05.不正確地使用scanf (此題目會觸發segmentation fault的原因)

問題四

怎麼使用gdb來找出發生問題的點？請給出範例以及debug的過程。

在以下的範例中，用註解的方式來解釋debug的過程。

```
ubuntu1804@ubuntu1804-VirtualBox:~$ gdb ./test //啟動GDB
GNU gdb (Ubuntu 8.1-0ubuntu3) 8.1.0.20180409-git
Copyright (C) 2018 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 ./test...done.
(gdb) l //等同於list，並從第一行列出程式碼
1       #include <stdio.h>
2       #include <string.h>
3
4       int get_num(){
5         int input;
6         scanf("%d",input);
7         return input;
8       }
9
10      int main(){
(gdb) b main //等同於breakpoints，設斷點在程式main函式後處
Breakpoint 1 at 0x6d5: file test.c, line 11.
(gdb) b get_num //等同於breakpoints，設斷點在程式 get_num函式處
Breakpoint 2 at 0x6b2: file test.c, line 6.
(gdb) info break //印出關於斷點的資訊
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x00000000000006d5 in main
                                                   at test.c:11
2       breakpoint     keep y   0x00000000000006b2 in get_num
                                                   at test.c:6
(gdb) r //等同於run，執行程式
Starting program: /home/ubuntu1804/test

Breakpoint 1, main () at test.c:11
11        int input = get_num();
(gdb) n//等同於next，執行下一行

Breakpoint 2, get_num () at test.c:6
6         scanf("%d",input);
(gdb) n
20//輸入任何數字，觸發Segmentation fault

Program received signal SIGSEGV, Segmentation fault.//觸發題目所要求的Segmentation
fault，代表第六行程式有誤
```

```
0x00007f14e85a58c2 in _IO_vfscanf_internal ( s=<optimized out>,format=<optimized
out>, argptr=argptr@entry=0x7ffc32402110, errp=errp@entry=0x0) at vfscanf.c:1898
   vfscanf.c: No such file or directory.
(gdb) c//等同於continue，繼續執行程式
Continuing.

Program terminated with signal SIGSEGV, Segmentation fault.
The program no longer exists.
(gdb) q//等同於quit，結束執行程式
```

資料來源

https://henrybear327.gitbooks.io/gitbook_tutorial/content/Linux/GDB/index.html

https://www.itread01.com/content/1550660597.html

https://codertw.com/%E7%A8%8B%E5%BC%8F%E8%AA%9E%E8%A8%80/560380/

https://silencewt.github.io/2015/05/11/Segmentation-Fault%E9%94%99%E8%AF%AF%E5%8E%9F%E5%9B%A0%E6%80%BB%E7%BB%93/

https://www.cnblogs.com/hello--the-world/archive/2012/05/31/2528326.html

http://www.wtango.com/%E6%AE%B5%E9%94%99%E8%AF%AFsegmentation-fault%E4%BA%A7%E7%94%9F%E7%9A%84%E5%8E%9F%E5%9B%A0%E4%BB%A5%E5%8F%8A%E8%B0%83%E8%AF%95%E6%96%B9%E6%B3%95/

https://www.cnblogs.com/no7dw/archive/2013/02/20/2918372.html