SP hw3 report

b07902048 資工二 李宥霆

a

stack(high address->low address)	rbp	rsp
dummy	0x7fffffffe640	0x7fffffff49e0
funct_1	0x7fffffff49d0	0x7fffffff49b0
dummy	0x7fffffff49a0	0x7fffffead40
funct_2	0x7fffffead30	0x7fffffead10
dummy	0x7fffffead00	0x7ffffffe10a0
funct_3	0x7ffffffe1090	0x7ffffffe1070
dummy	0x7ffffffe1060	0x7fffffd7400
funct_4	0x7fffffd73f0	0x7fffffd73d0

b

不一定，要看變數被存在哪裡，如果變數被存在memory裡，下次回到function時這些變數的內容仍然會被保留，如果變數被存在register裡，下次回function就會被restore

c

dummy是為了funct_1~5內部要做function call而留的位置，因為在funct_1~5裡面會call到許多function(e.g: setjmp, sigpending, signal handler…)，而在call這些function時也會push出一個stack，如果沒有用dummy function把funct_1~5隔開的話，signal_handler就會覆蓋到原本位置上的funct而造成錯誤。

d

會出現錯誤，使用gdb後會發現在funct_4 return回dummy，在dummy內部出現錯誤，錯誤如下：

```
(gdb)
funct_5 (name=3) at hw3test.c:322
322     }
(gdb)
*** stack smashing detected ***: <unknown> terminated

Program received signal SIGABRT, Aborted.
0x00007ffff7de0f25 in raise () from /usr/lib/libc.so.6
(gdb)
```

這是因為dummy當初設計是要來暫存funct_1~4所呼叫的function call，因此在建完stack之後，dummy就失去了他的功能而被那些function call覆蓋掉，因此若用return的方式回dummy就會發生錯誤。

e

首先在做第一個task時，可以初步了解題目想要建出來的stack frame長什麼樣子，並用setjmp和longjmp把它建起來，第二個task只要搞懂mutex的使用方法就可以解決，最複雜的是第三個task，需要考慮的小細節有很多，像是判斷mutex的時機必須在每次進大迴圈之前，如果mutex不屬於自己或是0就必須longjmp，並在下次進入function時要重新判斷，還有sigpromask的參數設定需要徹底了解，不然容易出現沒有block到或沒有重新block到的問題