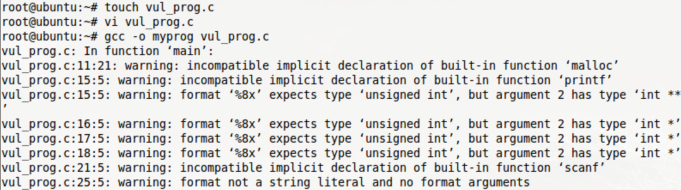
Lab 2 Format String Vulnerability Lab

Yukui Ye

SUID: 439644268

**Task 1: Exploit the vulnerability**

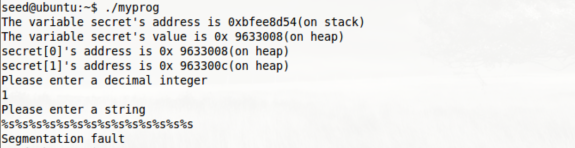
creat a file named vul\_prog.c, compile it, we can ignore the warning parts.



and make the program Set\_UID program



*a: Crash the program*

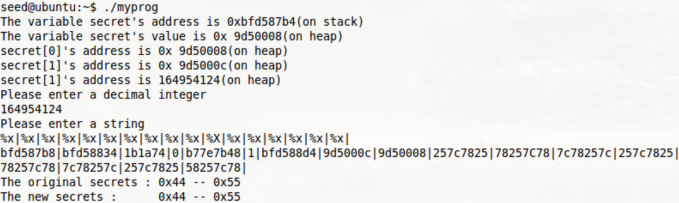
To crash the program , we need to make pointer points to an invalid place. Thus we input enough ”%s” till it reaches “\0”. 

*b: Print out the secret[1]value*

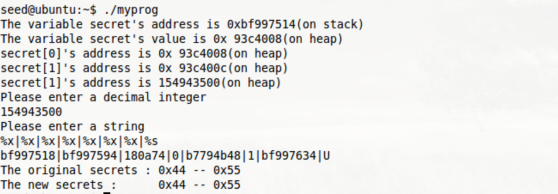
To read the secret[1]value, we need to put the address of secret[1] into stack since the original value of secret[1] is stored in heap. Therefore, we need to know the address of secret[1] in decimal form which can obtain by translating the hexadecimal into decimal integer. (To be more easier to solve this problem, I just add one line code in the original code file to print out the address of secret[1] which is in decimal formal so that we don’t need to calculate it every time we need that decimal integer).Then we use int\_input to store the address of secret[1] into the stack. After all the prepared work done, we need to use “%x” to move the pointer to that the place where store the address of secret[1]; and print it out.

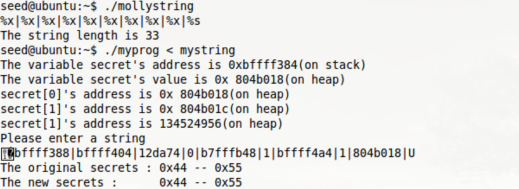
Add one line to get the address of secret[1] in decimal formal



put the address of secret[1] on the stack by using int\_input ; then calculate the secret[1]address is stored in the 8th %x place. 

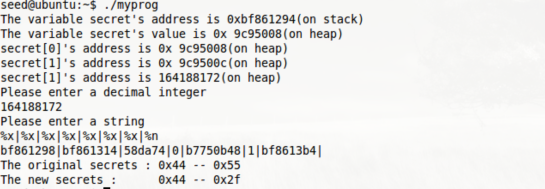
read the value of secret[1] by put %s after the 7th %x, The 8th block store the address of secret[1], by using %s, the pointer exactly point to the address of secret[1] to get the value.





*c: Modify the secret[1] value.*

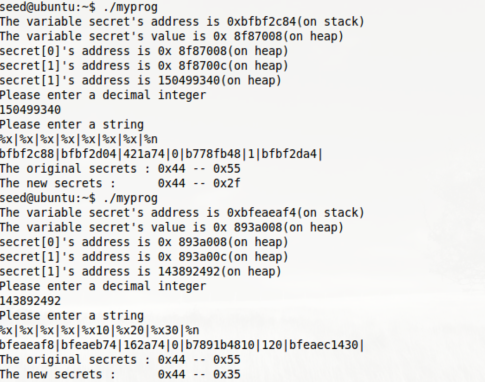
Depend on the previous task, we already known the 8th %x store the address of secret[1], all we need to do is change the “%s” to “%n”. “%n” will write the number of the string into a pointer. Then we could set the value of secret[1] as the number of the string that has been written.



we can easily tell that the value of secret[1] has been changed from 0x55 to 0x2f.

*d: Modify the secret[1] value to a pre-determined value*

We need to add more characters in the printf in order to change the secret[1]’s value to a pre-determined value.



Here, we add 6 characters in the printf, then the result turn from 0x2f to 0x35 which is exactly 6 characters different between them by checking the table of ASCII.

**Task 2: Memory Randomization**

From previous experience, we found that we got different address each time we run the program. To do this task, we need to shut down the random mechanism by typing this line code in root account.

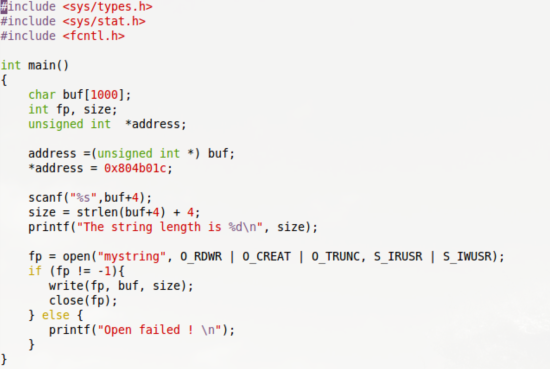


Then we need to modify the original “vul\_prog.c”

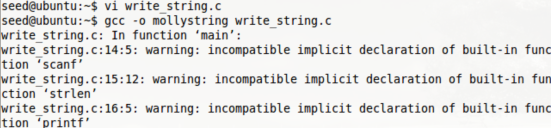
file since we removed the scanf() for int\_input.



Then creat the file named “write\_string.c”

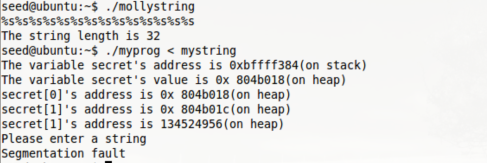


Compile it to file named “mollystring”, ignore the warning.



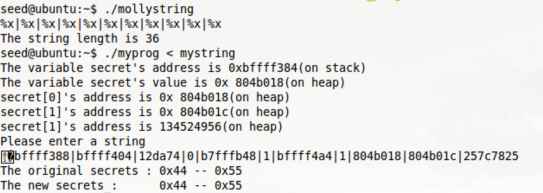
*a: Crash the program*

Just do the way like task1.

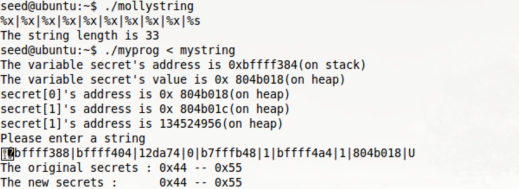


*b: Print out the value of secret[1]*

Test it in order to find out which %x is the place for address secret[1].



As showed from above screenshot, we found that the address of secret[1] lied in the 10th %x. then we just need to print it out with %s at the 10th %x.

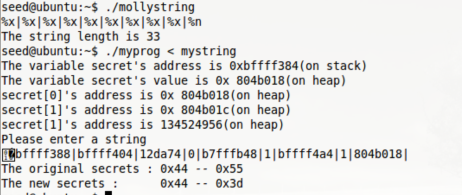


The result is just like the task 1. U is equal to 0x55 in ASCII.

*c: Modify the secret[1] value.*

Depend on the previous task, we already known the 10th %x store the address of secret[1], all we need to do is change the “%s” to “%n”. “%n” will write the number of the string into a pointer. Then we could set the value of secret[1] as the number of the string that has been written.

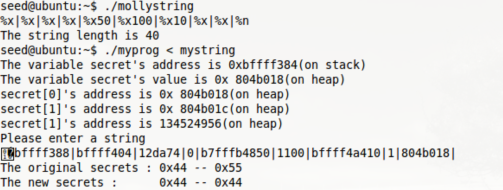
we can easily tell that the value of secret[1] has been changed from 0x55 to 0x2f.



By doing above thing, we changed the value of secret[1] from 0x55 to 0x3d.

*d: Modify the secret[1] value to a pre-determined value*

We need to add more characters in the printf in order to change the secret[1]’s value to a pre-determined value.



By adding 7number in the user input , we changed the value of secret[1] from previous 0x3d to 0x44.