Nicholas Garrett

Professor Zibran

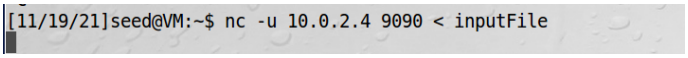
CS 4416

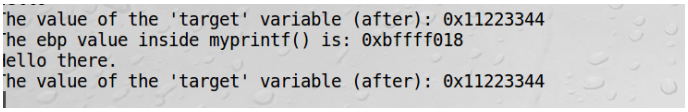
11/20/2021

Homework 2: Format String Exercise

1.
Part 1 involves setting up the server and sending date to the server from the client side. As shown in the screenshots, I have successfully sent a data string to the server from the client.

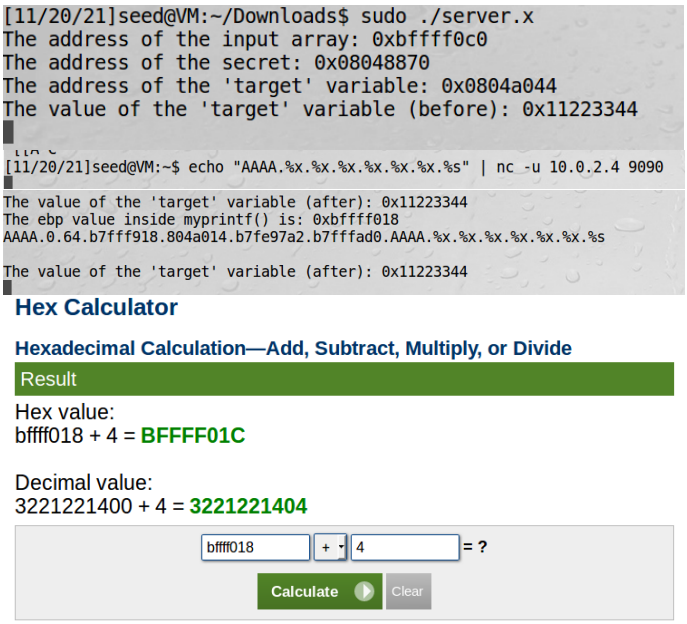
.





2

The figure shows the address of the buffer, followed by the client-side input and server-side output to find the address of the format string. The fourth image is of the Hex calculator I used to find the return address.



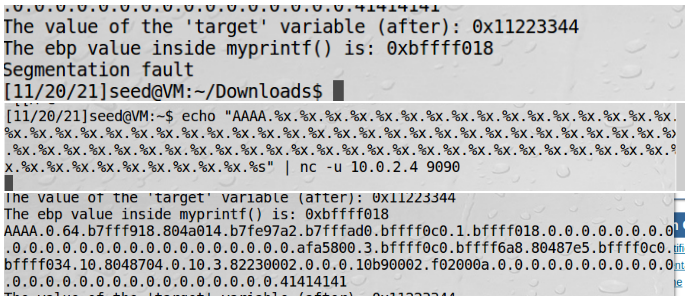
1. Format String: by following the screenshot of the server displaying 414141 found in problem 3, the address of the format string is 80 * 4 bytes above the buffer = bffff200.
2. Return address: EBP + 4 bytes = bffff01c
3. Buffer address: 0xbffff0c0

What is the distance between the locations marked by 1 and 3?
The distance between the two locations is 4 bytes * 80, or 320 bytes, shown in the middle two figures displayed above.

3.
String to make the server crash:

```
echo"AAAA.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.
%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.
%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.
%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%x.%s" | nc -u 10.0.2.4 9090
```

This is shown by how the server responds (top) to the client sending that string (middle), and the server sending the string 41414141 in the third figure, which indicates the format string address is found. This occurs when the last client sends x instead of s for its last character.



.

4.
4.a.
As shown in the third figure accompanying problem 2, by printing

```
echo "AAAA.%x.%x.%x.%x.%x.%x.%s" | nc -u 10.0.2.4 9090
```

, I was able to have the server print my input using 8 format specifiers

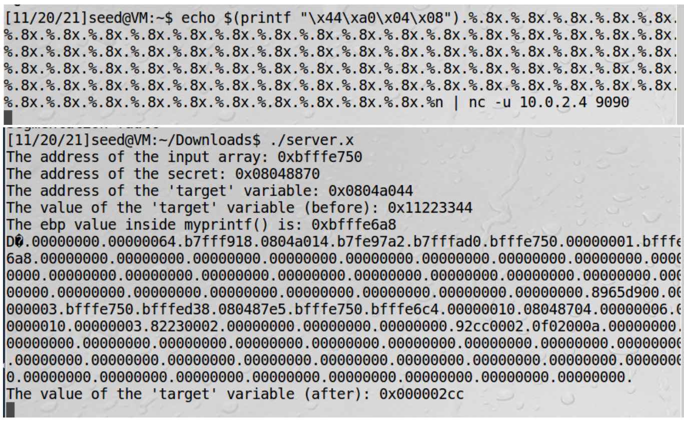
4.b.
By running the string (top) on the client, the server responded by printing its secret message, shown on the bottom. .

.

5.

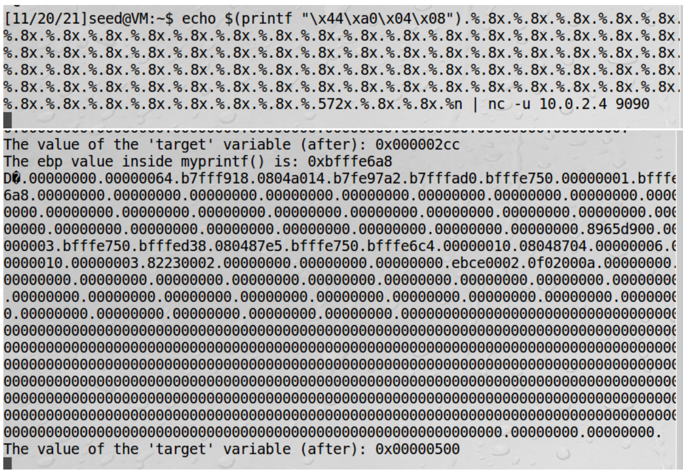
5.a

By inputting the value shown in the top figure, the server displays the output shown in the bottom figure. As you can see, the value of the "target" variable is 0x000002cc, which is not 11223344, so the variable value is changed. .

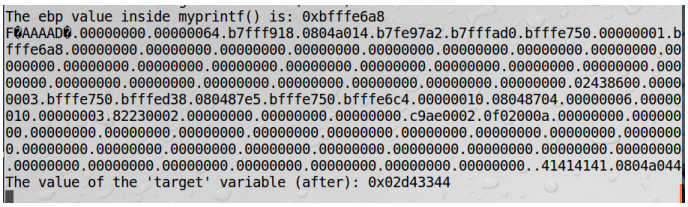


5.b

By running the below script, I was able to reset the value of the variable to 0x00000500 (shown in the bottom figure), which is 0x500–the value we were trying to reset it to. .



5.c In this figure, we can see the server's return for an initial input string. Therefore, I was able to calculate the upper address using the expected target = offset - current value: ff99 = 65433 = 64717

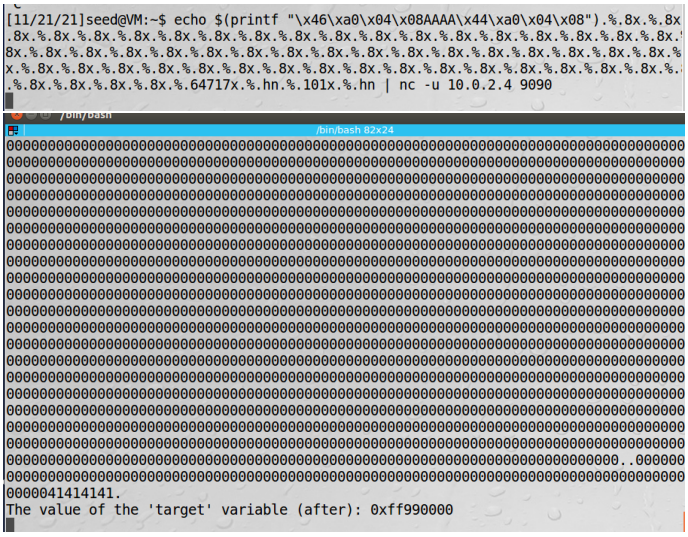


- 724 (the current value)

Similarly, the lower string can be calculated by performing an integer overflow.

ffff = 0-1 = 65535 = 65536 = 65433 + (offset + 1) So, 0 = 65433 + (offset + 2)

Finally, as a culmination, by inputting the string shown in the top figure, the server prints the desired result, shown in the bottom figure.


[11/21/21]seed@VM:~$ echo $(printf "\x46\xa0\x04\x08AAAA\x44\xa0\x04\x08").%.8x.%.8x
.8x.%.8x.%.8x.%