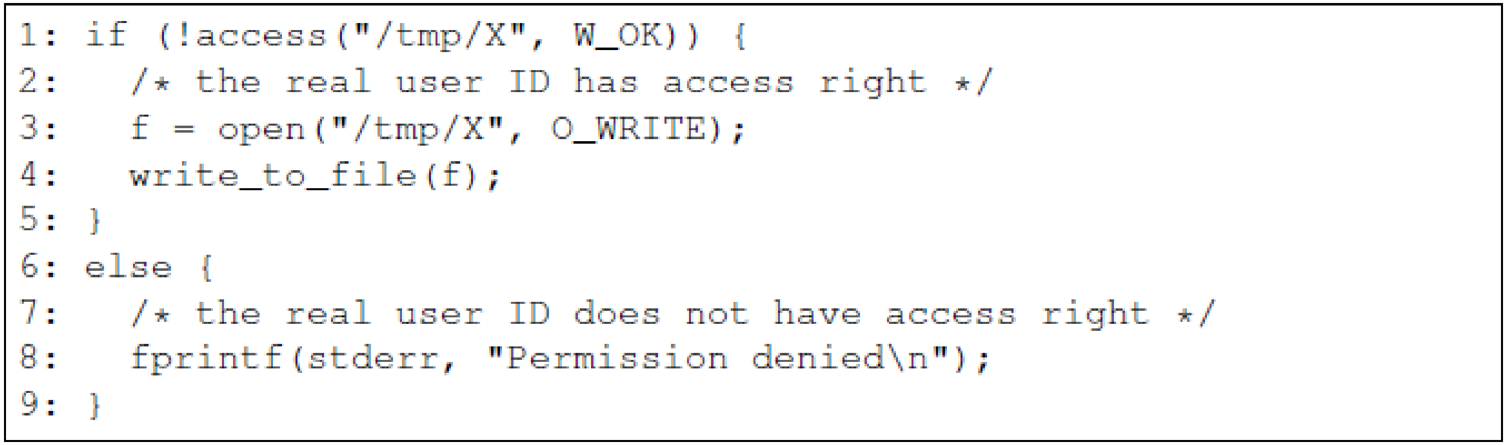
Race Condition Vulnerability Lab

Kailiang

Race Condition Vulnerability



```
if (!access("/tmp/X", W_OK)) {
  /* the real user ID has access right */
  f = open("/tmp/X", O_WRITE);
  write_to_file(f);
}
else {
  /* the real user ID does not have access right */
  fprintf(stderr, "Permission denied\n");
}
```

Step 1: Before access(/tmp/X,W_OK), the file /tmp/X is indeed /tmp/X
Step 2: After access(/tmp/X,W_OK), change /tmp/X to /etc/passwd

Reason: The window between the checking and using: Time-of-Check, Time-of-Use(TOCTOU)

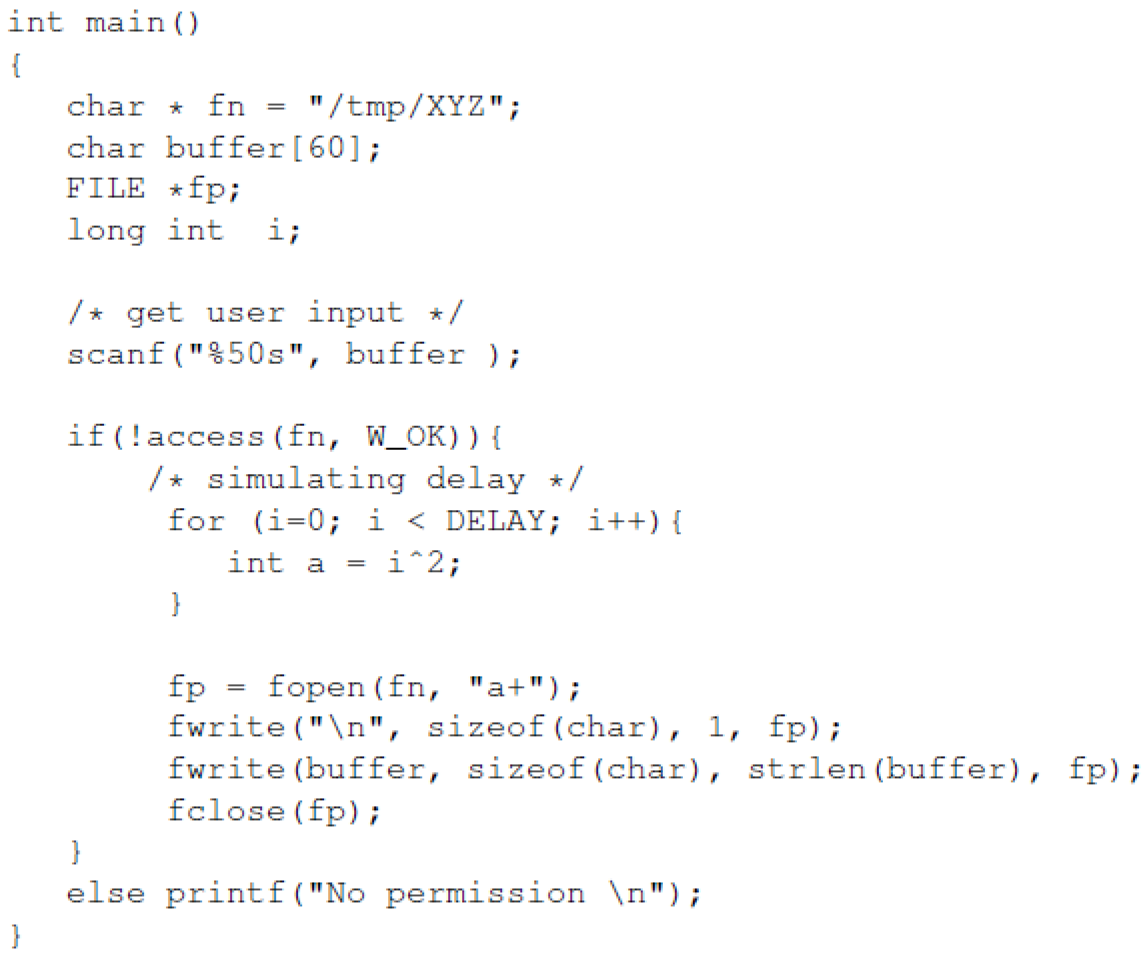
Lab Tasks

2.1 Initial Setup

1> Ubuntu 12.04 comes with an built-in protection against race condition attacks.
2> This scheme works by restricting who can follow a symlink.
3> Disable this protection using the following command:

```
$ sudo sysctl w kernel.yama.protected_sticky_symlinks=0
```

2.2 A Vulnerable Program (Vulp.c)



```
int main()
{
   char * fn = "/tmp/XYZ";
   char buffer[60];
   FILE *fp;
   long int  i;

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

   if(!access(fn, W_OK)){
       /* simulating delay */
        for (i=0; i < DELAY; i++){
           int a = i^2;
        }

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

Task 1: Exploit the Race Condition Vulnerabilities

1. Overwrite any file that belongs to Root. (/etc/passwd)
   Point: 20

2. Gain root privileges. (/etc/shadow)
   Point: 10

3. Professor competition question Point: 10

Task 1: Notes

1> To add a new user to the PC, add a new entry to /etc/passwd and /etc/shadow.

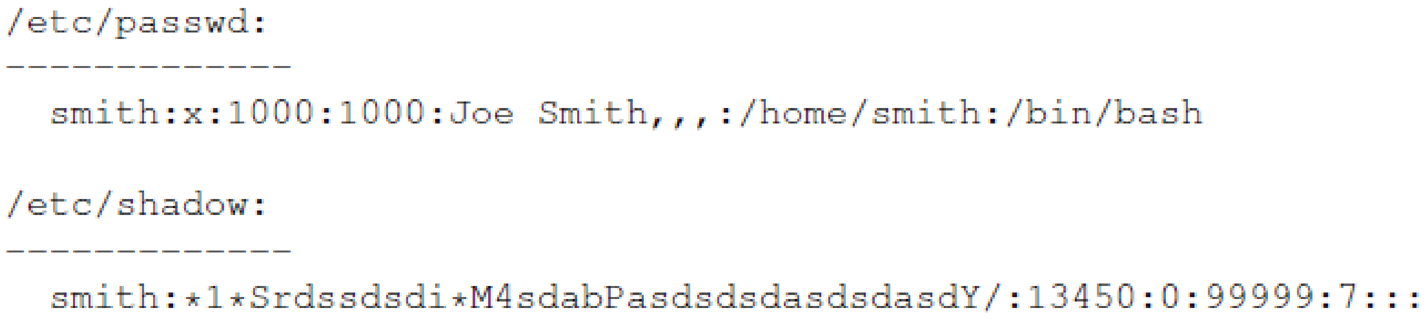
2> Add a new user attacker. Pay close attention to the user id and group id fields.

3> Remember to save a copy of /etc/passwd and /etc/shadow to other directory.

4> Before you reboot, make sure that /etc/passwd and /etc/shadow are correct.

Task 1: Steps

1> Look at /etc/passwd and /etc/shadow. Understand the format.



```
/etc/passwd:
--------------
  smith:x:1000:1000:Joe Smith,,,:/home/smith:/bin/bash

/etc/shadow:
--------------
  smith:*1*Srdssdsdi*M4sdabPasdsdsdasdsdasdY/:13450:0:99999:7:::
```

In the /etc/shadow file, for the encrypted password, use **U6aMy0wojraho** as the encrypted password. (This is the encrypted format for a blank password)

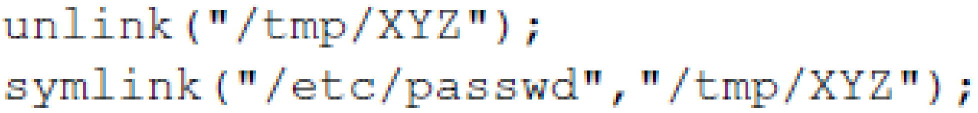
Task 1: Steps

2> Modify /etc/passwd file and /etc/shadow file using set-root-uid program vulp.c (Use input redirection. Create a file with the new attacker user details. Run the input redirection command to vulp in a loop. Use a shell script for that).

```
./vulp < input_file
```

Task 1: Steps

4> Create symbolic links

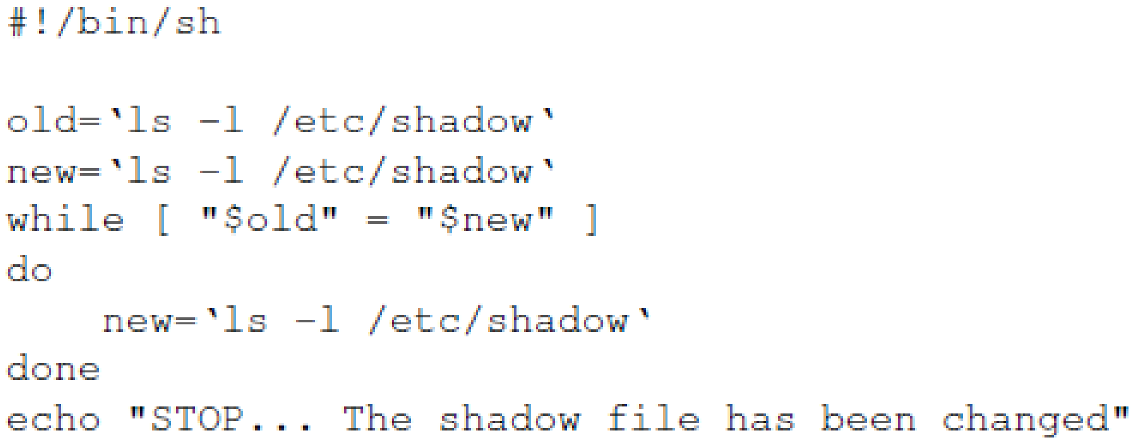


```
unlink("/tmp/XYZ");
symlink("/etc/passwd","/tmp/XYZ");
```

- Unlink
- Link to a normal file
- sleep
- Unlink
- Link to the target file (/etc/passwd or /etc/shadow)

Task 1: Steps

5> Use check.sh from the lab description website.



```
#!/bin/sh

old=`ls -l /etc/shadow`
new=`ls -l /etc/shadow`
while [ "$old" = "$new" ]
do
    new=`ls -l /etc/shadow`
done
echo "STOP... The shadow file has been changed"
```

Task 1: Summary

- Terminal: totally 2
    - Terminal 1: attack.sh (execute vulp & check modify)
    - Terminal 2: link.sh (switch links)
- Write down professor competition question in your lab report eg:

Task 2: Protection Mechanism A: Repeating (Point: 20)

1> Add new access() and open() checks to program. Also add i-node checks.

lstat(): returns the status of the link
fstat(): return status of the file link point to

Algorithms:

```
    if(!access (fn, W_OK)) {
        f1 = fopen (fn, “a”);
    }
     if(!access (fn, W_OK)) {
        f2 = fopen (fn, “a”);
    }
    if (whether two fds point to same inode)
        write to file
```

Task 2: fstat() lstat() example

lstat:

```
struct stat fstat;
 FILE *f = fopen(“/tmp/XYZ”,”a+”);
lstat(“/tmp/XYZ”, &fstat);
printf(fstat.st_ino); //uniquely identify the file within the system
```

fstat:

```
struct stat fstat
FILE *f = fopen(“/tmp/XYZ”, ”a+”);
fstat(fileno(f), &fstat);
printf(fstat.st_ino); //uniquely identify the file within the system
```

Task 3: Protection Mechanism B: Principle of Least Privilege (Point: 15)

1> Use seteuid() to change the user’s effective user id from root to a lower privilege level

2> Report if attack was successful

Task 4: Protection Mechanism C: Ubuntu’s Built-in Scheme (Point: 25)

1> Reactivate protection scheme.

2> Answer the questions asked in the lab description.

3> Hint: in what situation attack can success and why?