Department of Computer Science

California State University, Channel Islands

COMP-524: Security

Lab Report

Lab Number: 2

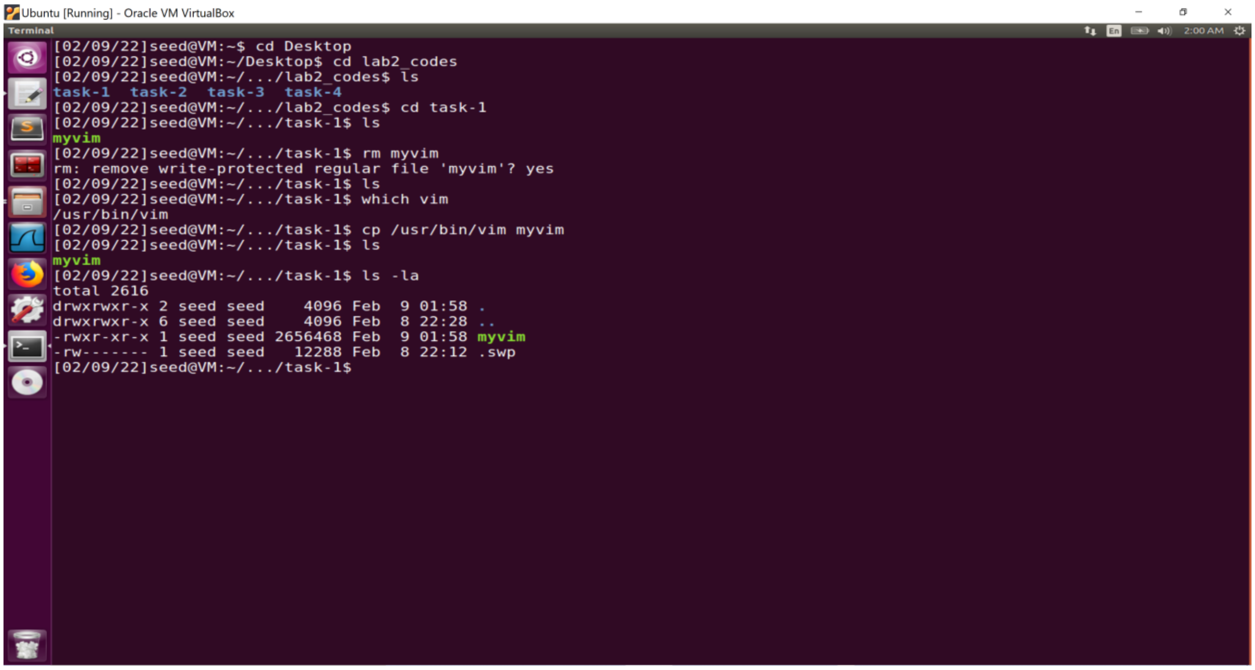
Lab Topic: SET-UID PROGRAMS

Task 1: Set-UID Programs

Set-UID is an important security mechanism in Unix operating systems. When a Set-UID program runs, it assumes the owner's privileges. For example, if the program's owner is root, then when anyone runs this program, the program gains the root's privileges during its execution. Set-UID allows us to do many interesting things, but it escalates the user's privilege when executed, making it quite risky. Although the behaviors of Set-UID programs are decided by their program logic, not by users, users can indeed affect the behaviors via environment variables.

In this task, create your own version of "vim" program and turn it to a root-owned set-UID program by following these steps:

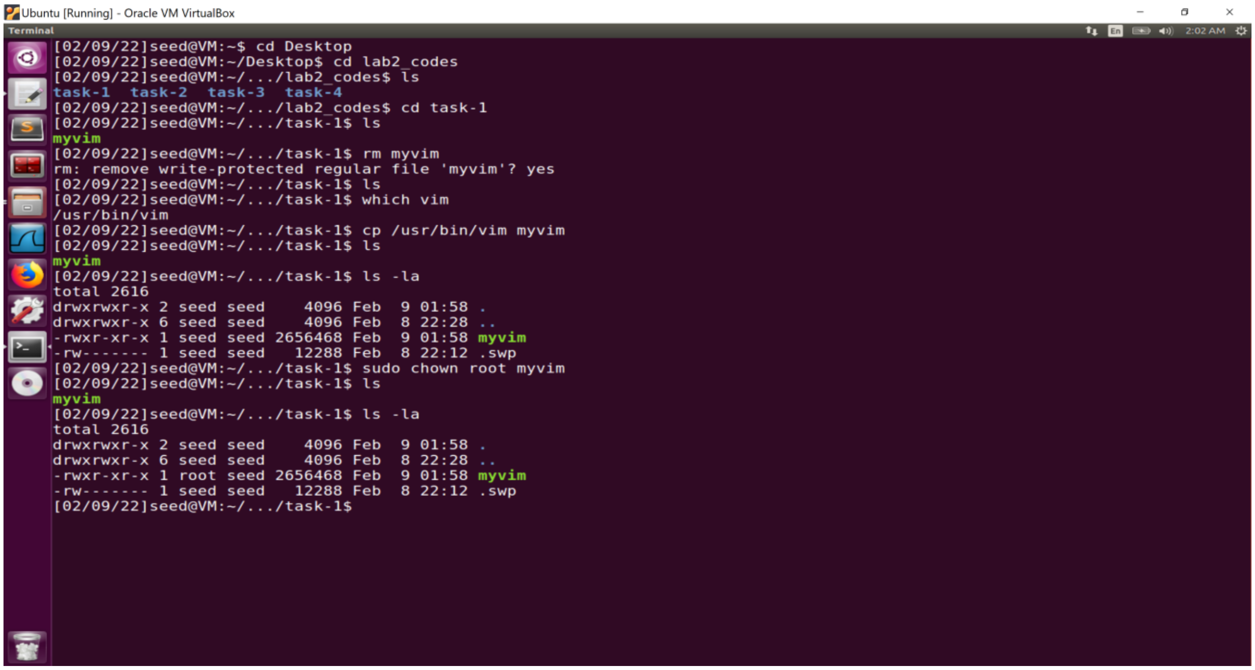
a) Copy the vim program (use "which vim" to know the location of the program in your system).



```
[02/09/22]seed@VM:~$ cd Desktop
[02/09/22]seed@VM:~/Desktop$ cd lab2_codes
[02/09/22]seed@VM:~/.../lab2_codes$ ls
task-1  task-2  task-3  task-4
[02/09/22]seed@VM:~/.../lab2_codes$ cd task-1
[02/09/22]seed@VM:~/.../task-1$ ls
myvim
[02/09/22]seed@VM:~/.../task-1$ rm myvim
rm: remove write-protected regular file 'myvim'? yes
[02/09/22]seed@VM:~/.../task-1$ ls
[02/09/22]seed@VM:~/.../task-1$ which vim
/usr/bin/vim
[02/09/22]seed@VM:~/.../task-1$ cp /usr/bin/vim myvim
[02/09/22]seed@VM:~/.../task-1$ ls
myvim
[02/09/22]seed@VM:~/.../task-1$ ls -la
total 2616
drwxrwxr-x 2 seed seed    4096 Feb  9 01:58 .
drwxrwxr-x 6 seed seed    4096 Feb  8 22:28 ..
-rwxr-xr-x 1 seed seed 2656468 Feb  9 01:58 myvim
-rw------- 1 seed seed   12288 Feb  8 22:12 .swp
[02/09/22]seed@VM:~/.../task-1$
```

Fig 2.1

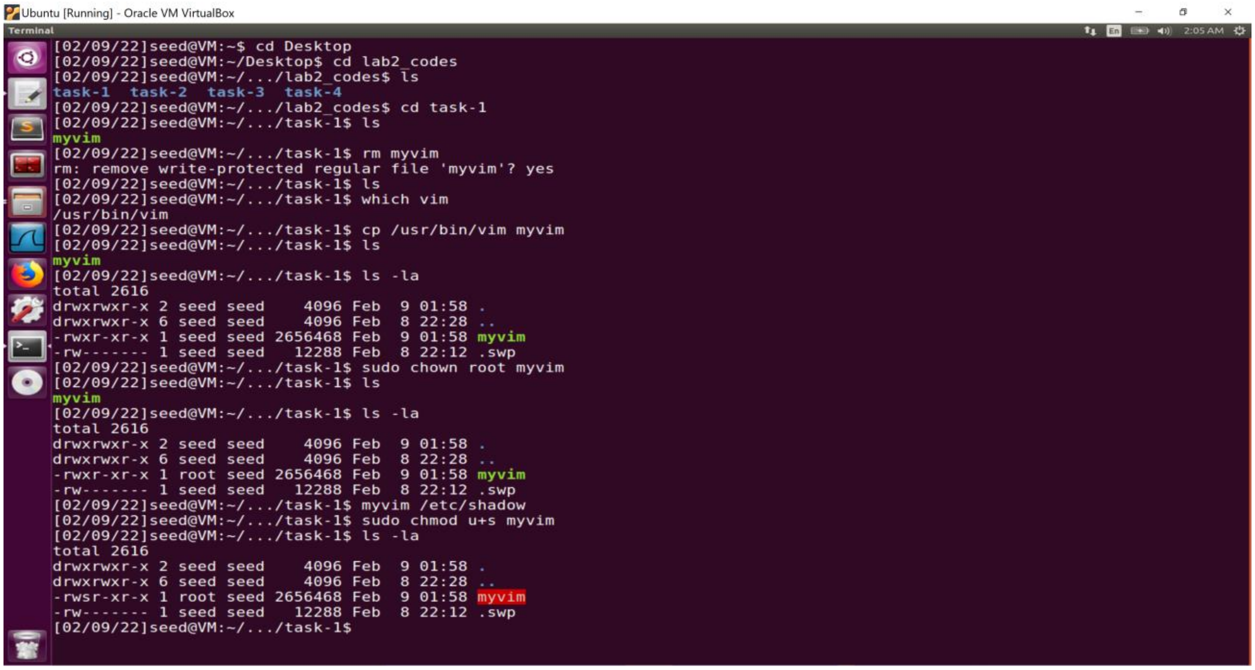
b) Change the ownership to "root".



```
[02/09/22]seed@VM:~$ cd Desktop
[02/09/22]seed@VM:~/Desktop$ cd lab2_codes
[02/09/22]seed@VM:~/.../lab2_codes$ ls
task-1  task-2  task-3  task-4
[02/09/22]seed@VM:~/.../lab2_codes$ cd task-1
[02/09/22]seed@VM:~/.../task-1$ ls
myvim
[02/09/22]seed@VM:~/.../task-1$ rm myvim
rm: remove write-protected regular file 'myvim'? yes
[02/09/22]seed@VM:~/.../task-1$ ls
[02/09/22]seed@VM:~/.../task-1$ which vim
/usr/bin/vim
[02/09/22]seed@VM:~/.../task-1$ cp /usr/bin/vim myvim
[02/09/22]seed@VM:~/.../task-1$ ls
myvim
[02/09/22]seed@VM:~/.../task-1$ ls -la
total 2616
drwxrwxr-x 2 seed seed    4096 Feb  9 01:58 .
drwxrwxr-x 6 seed seed    4096 Feb  8 22:28 ..
-rwxr-xr-x 1 seed seed 2656468 Feb  9 01:58 myvim
-rw------- 1 seed seed   12288 Feb  8 22:12 .swp
[02/09/22]seed@VM:~/.../task-1$ sudo chown root myvim
[02/09/22]seed@VM:~/.../task-1$ ls
myvim
[02/09/22]seed@VM:~/.../task-1$ ls -la
total 2616
drwxrwxr-x 2 seed seed    4096 Feb  9 01:58 .
drwxrwxr-x 6 seed seed    4096 Feb  8 22:28 ..
-rwxr-xr-x 1 root seed 2656468 Feb  9 01:58 myvim
-rw------- 1 seed seed   12288 Feb  8 22:12 .swp
[02/09/22]seed@VM:~/.../task-1$
```

Fig 2.2

c) Change the set-UID bit using "chmod" command.



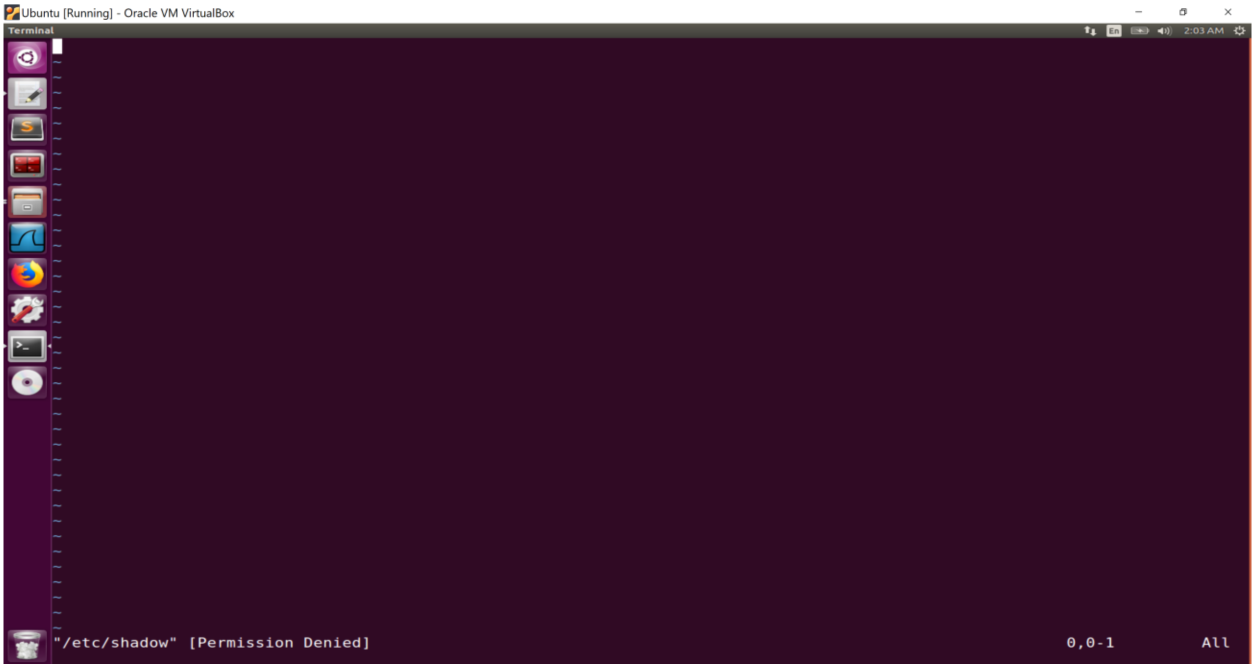
```
[02/09/22]seed@VM:~$ cd Desktop
[02/09/22]seed@VM:~/Desktop$ cd lab2_codes
[02/09/22]seed@VM:~/.../lab2_codes$ ls
task-1  task-2  task-3  task-4
[02/09/22]seed@VM:~/.../lab2_codes$ cd task-1
[02/09/22]seed@VM:~/.../task-1$ ls
myvim
[02/09/22]seed@VM:~/.../task-1$ rm myvim
rm: remove write-protected regular file 'myvim'? yes
[02/09/22]seed@VM:~/.../task-1$ ls
[02/09/22]seed@VM:~/.../task-1$ which vim
/usr/bin/vim
[02/09/22]seed@VM:~/.../task-1$ cp /usr/bin/vim myvim
[02/09/22]seed@VM:~/.../task-1$ ls
myvim
[02/09/22]seed@VM:~/.../task-1$ ls -la
total 2616
drwxrwxr-x 2 seed seed    4096 Feb  9 01:58 .
drwxrwxr-x 6 seed seed    4096 Feb  8 22:28 ..
-rwxr-xr-x 1 seed seed 2656468 Feb  9 01:58 myvim
-rw------- 1 seed seed   12288 Feb  8 22:12 .swp
[02/09/22]seed@VM:~/.../task-1$ sudo chown root myvim
[02/09/22]seed@VM:~/.../task-1$ ls
myvim
[02/09/22]seed@VM:~/.../task-1$ ls -la
total 2616
drwxrwxr-x 2 seed seed    4096 Feb  9 01:58 .
drwxrwxr-x 6 seed seed    4096 Feb  8 22:28 ..
-rwxr-xr-x 1 root seed 2656468 Feb  9 01:58 myvim
-rw------- 1 seed seed   12288 Feb  8 22:12 .swp
[02/09/22]seed@VM:~/.../task-1$ myvim /etc/shadow
[02/09/22]seed@VM:~/.../task-1$ sudo chmod u+s myvim
[02/09/22]seed@VM:~/.../task-1$ ls -la
total 2616
drwxrwxr-x 2 seed seed    4096 Feb  9 01:58 .
drwxrwxr-x 6 seed seed    4096 Feb  8 22:28 ..
-rwsr-xr-x 1 root seed 2656468 Feb  9 01:58 myvim
-rw------- 1 seed seed   12288 Feb  8 22:12 .swp
[02/09/22]seed@VM:~/.../task-1$
```

Fig 2.3

Please answer the following questions about this activity:

Q1) After completing step b, try to view the content of the "shadow" file and describe your observation. Can you view the content of the file?

Ans. No, I didn’t view the content of the file.

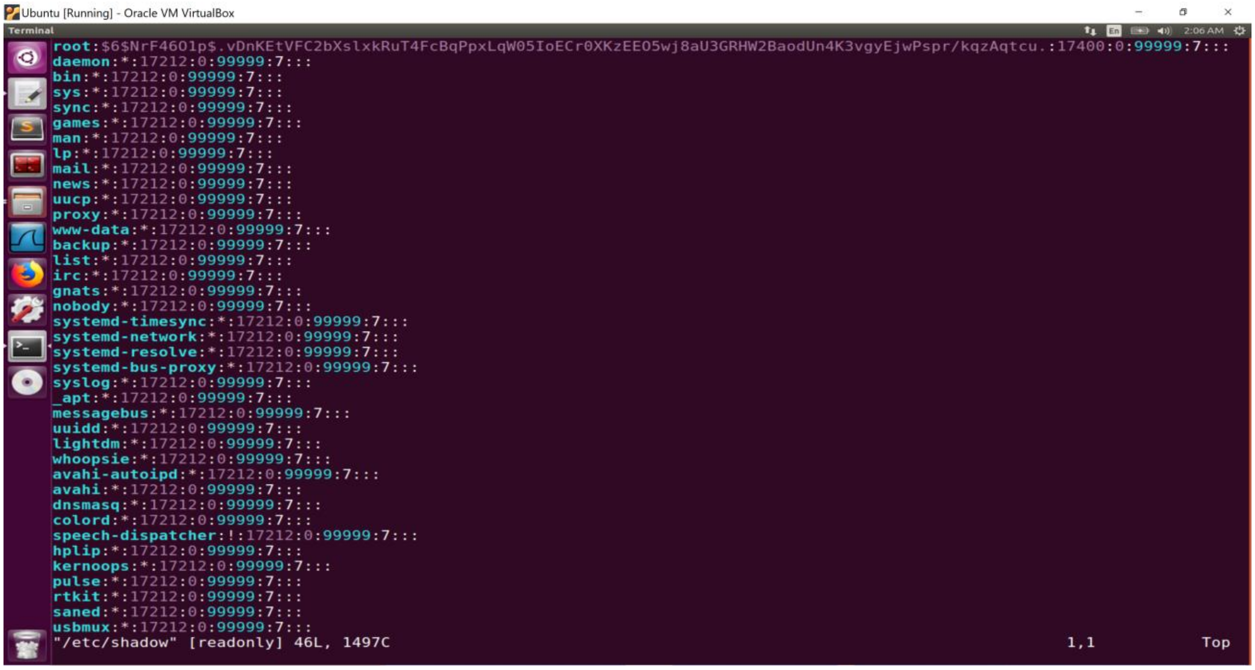


```
"/etc/shadow" [Permission Denied]                                    0,0-1         All
```

Fig 2.4

Q2) After completing step c, try to view the content of the "shadow" file and describe your observation.

Ans. As per Fig 2.5, I observed that it have different content such as root, bin, sys, lp, and mail, etc.



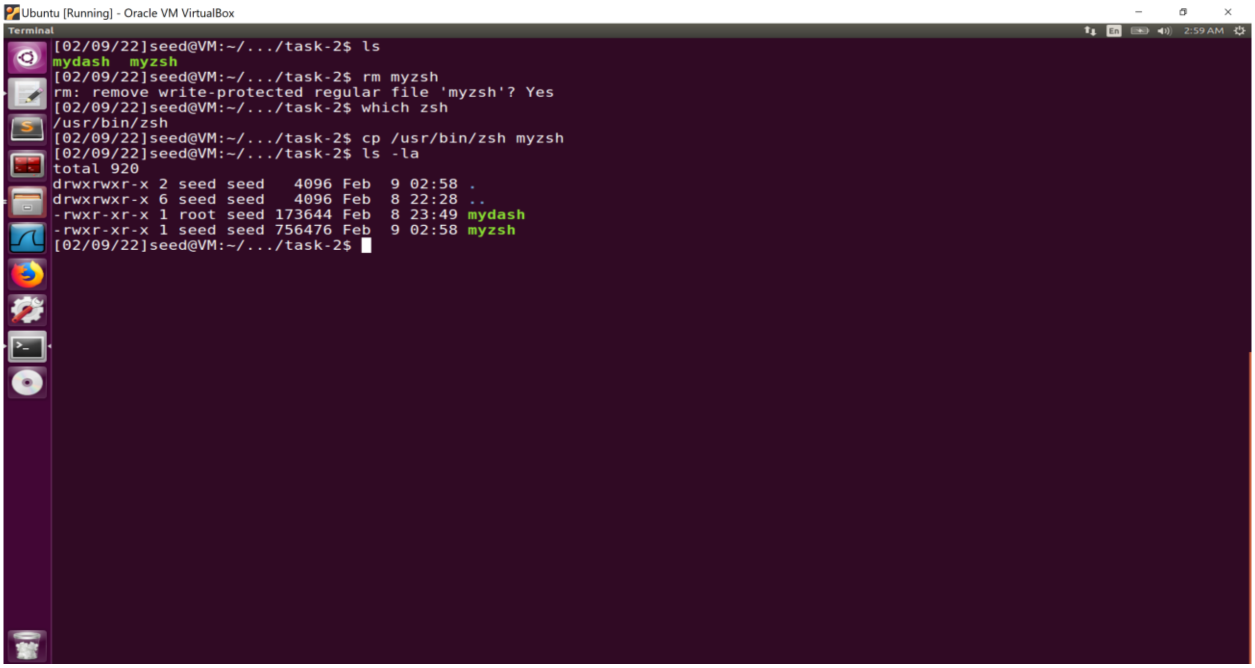
```
root:$6$NrF4601p$.vDnKEtVFC2bXslxkRuT4FcBqPpxLqW05IoECr0XKzEE05wj8aU3GRHW2BaodUn4K3vgyEjwPspr/kqzAqtcu.:17400:0:99999:7:::
daemon:*:17212:0:99999:7:::
bin:*:17212:0:99999:7:::
sys:*:17212:0:99999:7:::
sync:*:17212:0:99999:7:::
games:*:17212:0:99999:7:::
man:*:17212:0:99999:7:::
lp:*:17212:0:99999:7:::
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:::
systemd-bus-proxy:*:17212:0:99999:7:::
syslog:*:17212:0:99999:7:::
_apt:*:17212:0:99999:7:::
messagebus:*:17212:0:99999:7:::
uuidd:*: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:::
"/etc/shadow" [readonly] 46L, 1497C                                                                      1,1           Top
```

Fig 2.5

Task 2: Vulnerability in Shell Programs

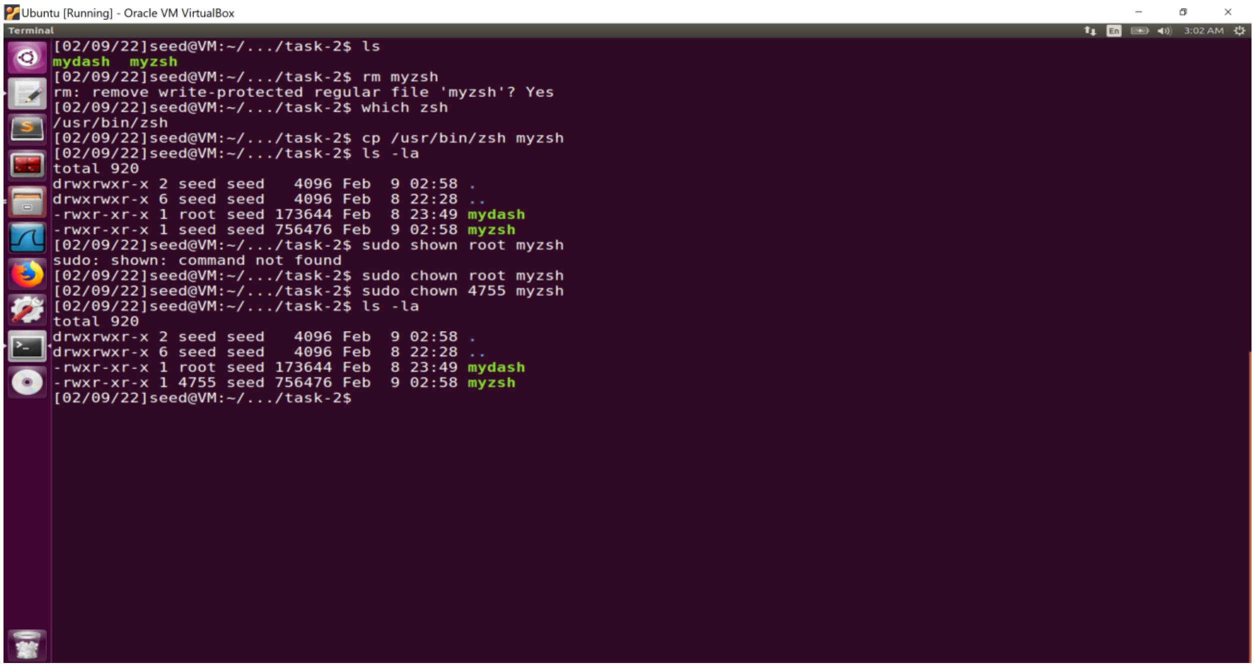
Show that "zsh" is not safe when a system uses set-UID programs to provide privilege to its users. Look at the Appendix A in set-UID lecture.

In Fig 2.6, you will see “zsh” is saved as “myzsh” .



```
[02/09/22]seed@VM:~/.../task-2$ ls
mydash  myzsh
[02/09/22]seed@VM:~/.../task-2$ rm myzsh
rm: remove write-protected regular file 'myzsh'? Yes
[02/09/22]seed@VM:~/.../task-2$ which zsh
/usr/bin/zsh
[02/09/22]seed@VM:~/.../task-2$ cp /usr/bin/zsh myzsh
[02/09/22]seed@VM:~/.../task-2$ ls -la
total 920
drwxrwxr-x 2 seed seed   4096 Feb  9 02:58 .
drwxrwxr-x 6 seed seed   4096 Feb  8 22:28 ..
-rwxr-xr-x 1 root seed 173644 Feb  8 23:49 mydash
-rwxr-xr-x 1 seed seed 756476 Feb  9 02:58 myzsh
[02/09/22]seed@VM:~/.../task-2$ 
```

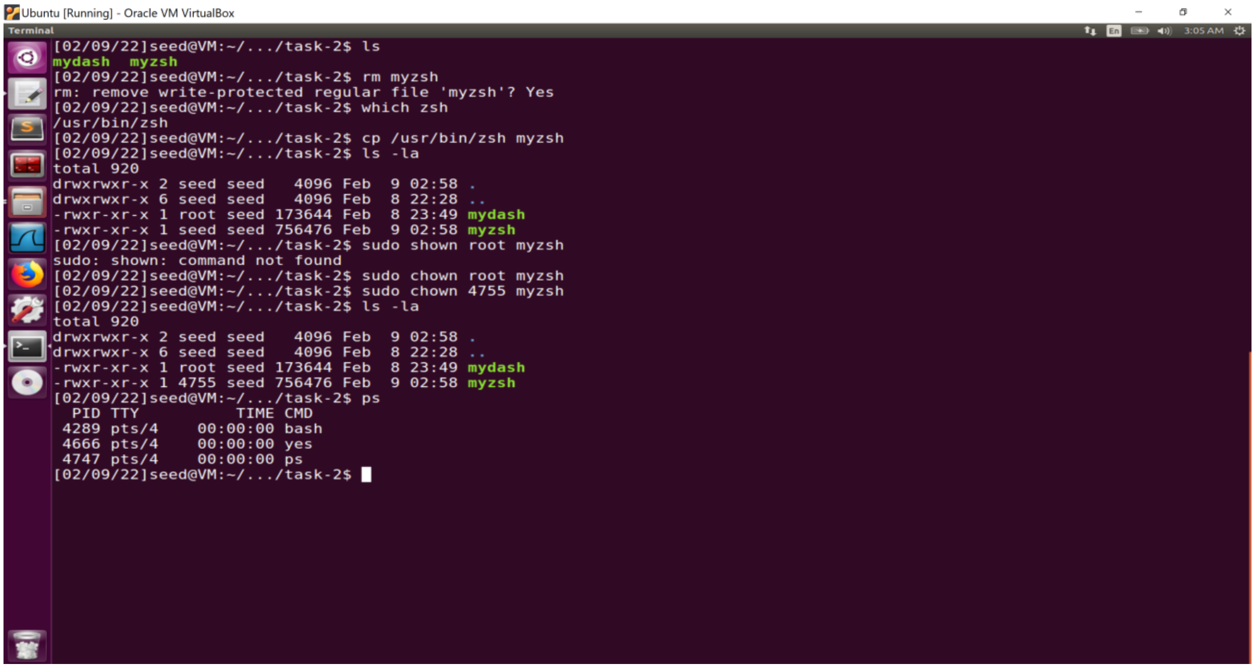
Fig 2.6



```
[02/09/22]seed@VM:~/.../task-2$ ls
mydash  myzsh
[02/09/22]seed@VM:~/.../task-2$ rm myzsh
rm: remove write-protected regular file 'myzsh'? Yes
[02/09/22]seed@VM:~/.../task-2$ which zsh
/usr/bin/zsh
[02/09/22]seed@VM:~/.../task-2$ cp /usr/bin/zsh myzsh
[02/09/22]seed@VM:~/.../task-2$ ls -la
total 920
drwxrwxr-x 2 seed seed   4096 Feb  9 02:58 .
drwxrwxr-x 6 seed seed   4096 Feb  8 22:28 ..
-rwxr-xr-x 1 root seed 173644 Feb  8 23:49 mydash
-rwxr-xr-x 1 seed seed 756476 Feb  9 02:58 myzsh
[02/09/22]seed@VM:~/.../task-2$ sudo shown root myzsh
sudo: shown: command not found
[02/09/22]seed@VM:~/.../task-2$ sudo chown root myzsh
[02/09/22]seed@VM:~/.../task-2$ sudo chown 4755 myzsh
[02/09/22]seed@VM:~/.../task-2$ ls -la
total 920
drwxrwxr-x 2 seed seed   4096 Feb  9 02:58 .
drwxrwxr-x 6 seed seed   4096 Feb  8 22:28 ..
-rwxr-xr-x 1 root seed 173644 Feb  8 23:49 mydash
-rwxr-xr-x 1 4755 seed 756476 Feb  9 02:58 myzsh
[02/09/22]seed@VM:~/.../task-2$
```

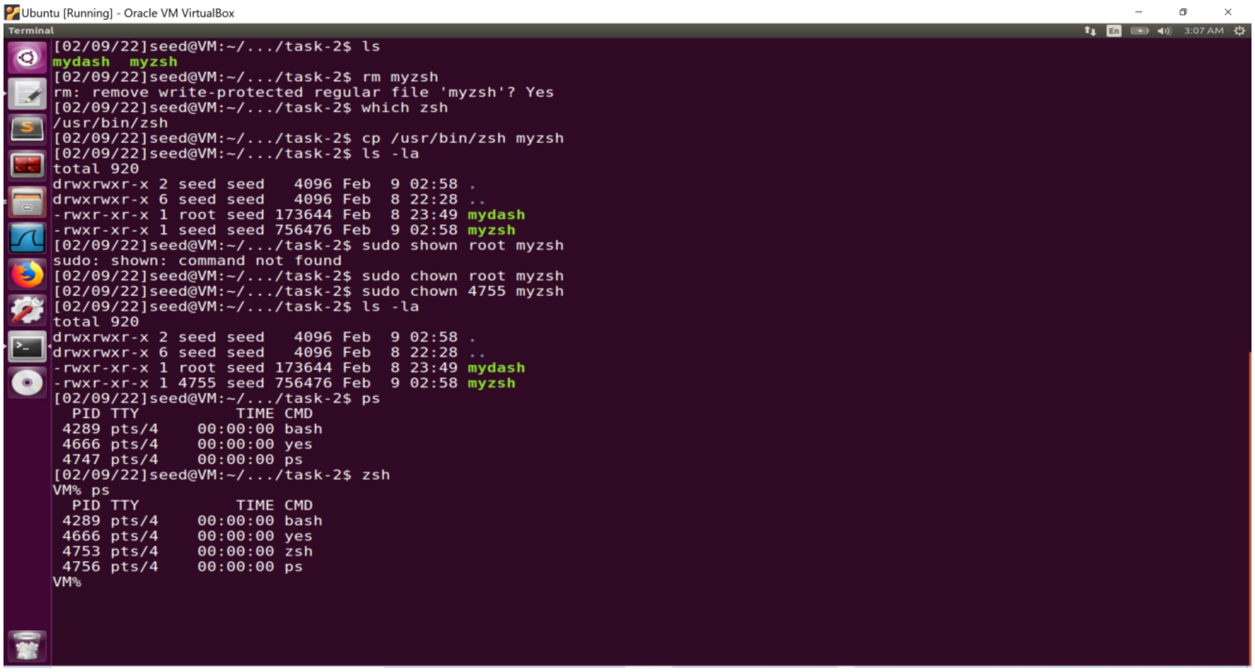
Fig 2.7

In Fig 2.7, you will find that I have changed the root as well as of access (that is Read, Write, and Execution) of "myzsh" file.



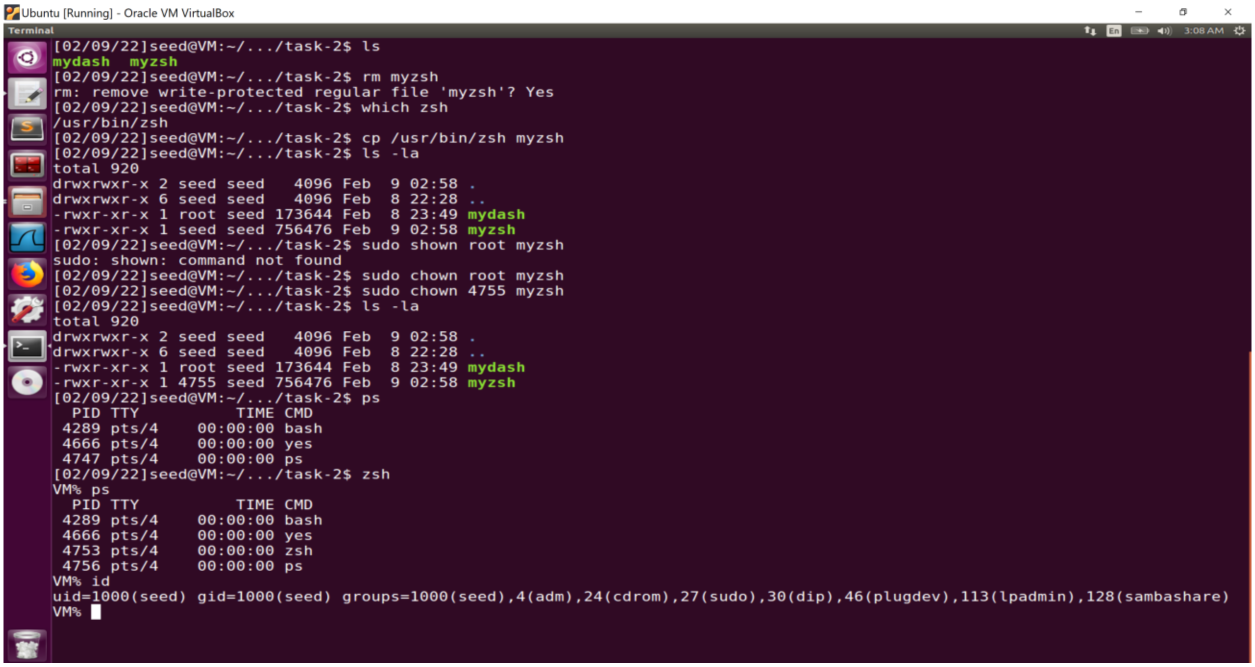
```
[02/09/22]seed@VM:~/.../task-2$ ls
mydash  myzsh
[02/09/22]seed@VM:~/.../task-2$ rm myzsh
rm: remove write-protected regular file 'myzsh'? Yes
[02/09/22]seed@VM:~/.../task-2$ which zsh
/usr/bin/zsh
[02/09/22]seed@VM:~/.../task-2$ cp /usr/bin/zsh myzsh
[02/09/22]seed@VM:~/.../task-2$ ls -la
total 920
drwxrwxr-x 2 seed seed   4096 Feb  9 02:58 .
drwxrwxr-x 6 seed seed   4096 Feb  8 22:28 ..
-rwxr-xr-x 1 root seed 173644 Feb  8 23:49 mydash
-rwxr-xr-x 1 seed seed 756476 Feb  9 02:58 myzsh
[02/09/22]seed@VM:~/.../task-2$ sudo shown root myzsh
sudo: shown: command not found
[02/09/22]seed@VM:~/.../task-2$ sudo chown root myzsh
[02/09/22]seed@VM:~/.../task-2$ sudo chown 4755 myzsh
[02/09/22]seed@VM:~/.../task-2$ ls -la
total 920
drwxrwxr-x 2 seed seed   4096 Feb  9 02:58 .
drwxrwxr-x 6 seed seed   4096 Feb  8 22:28 ..
-rwxr-xr-x 1 root seed 173644 Feb  8 23:49 mydash
-rwxr-xr-x 1 4755 seed 756476 Feb  9 02:58 myzsh
[02/09/22]seed@VM:~/.../task-2$ ps
  PID TTY          TIME CMD
 4289 pts/4    00:00:00 bash
 4666 pts/4    00:00:00 yes
 4747 pts/4    00:00:00 ps
[02/09/22]seed@VM:~/.../task-2$ █
```

Fig 2.8



```
[02/09/22]seed@VM:~/.../task-2$ ls
mydash  myzsh
[02/09/22]seed@VM:~/.../task-2$ rm myzsh
rm: remove write-protected regular file 'myzsh'? Yes
[02/09/22]seed@VM:~/.../task-2$ which zsh
/usr/bin/zsh
[02/09/22]seed@VM:~/.../task-2$ cp /usr/bin/zsh myzsh
[02/09/22]seed@VM:~/.../task-2$ ls -la
total 920
drwxrwxr-x 2 seed seed   4096 Feb  9 02:58 .
drwxrwxr-x 6 seed seed   4096 Feb  8 22:28 ..
-rwxr-xr-x 1 root seed 173644 Feb  8 23:49 mydash
-rwxr-xr-x 1 seed seed 756476 Feb  9 02:58 myzsh
[02/09/22]seed@VM:~/.../task-2$ sudo shown root myzsh
sudo: shown: command not found
[02/09/22]seed@VM:~/.../task-2$ sudo chown root myzsh
[02/09/22]seed@VM:~/.../task-2$ sudo chown 4755 myzsh
[02/09/22]seed@VM:~/.../task-2$ ls -la
total 920
drwxrwxr-x 2 seed seed   4096 Feb  9 02:58 .
drwxrwxr-x 6 seed seed   4096 Feb  8 22:28 ..
-rwxr-xr-x 1 root seed 173644 Feb  8 23:49 mydash
-rwxr-xr-x 1 4755 seed 756476 Feb  9 02:58 myzsh
[02/09/22]seed@VM:~/.../task-2$ ps
  PID TTY          TIME CMD
 4289 pts/4    00:00:00 bash
 4666 pts/4    00:00:00 yes
 4747 pts/4    00:00:00 ps
[02/09/22]seed@VM:~/.../task-2$ zsh
VM% ps
  PID TTY          TIME CMD
 4289 pts/4    00:00:00 bash
 4666 pts/4    00:00:00 yes
 4753 pts/4    00:00:00 zsh
 4756 pts/4    00:00:00 ps
VM%
```

Fig 2.9



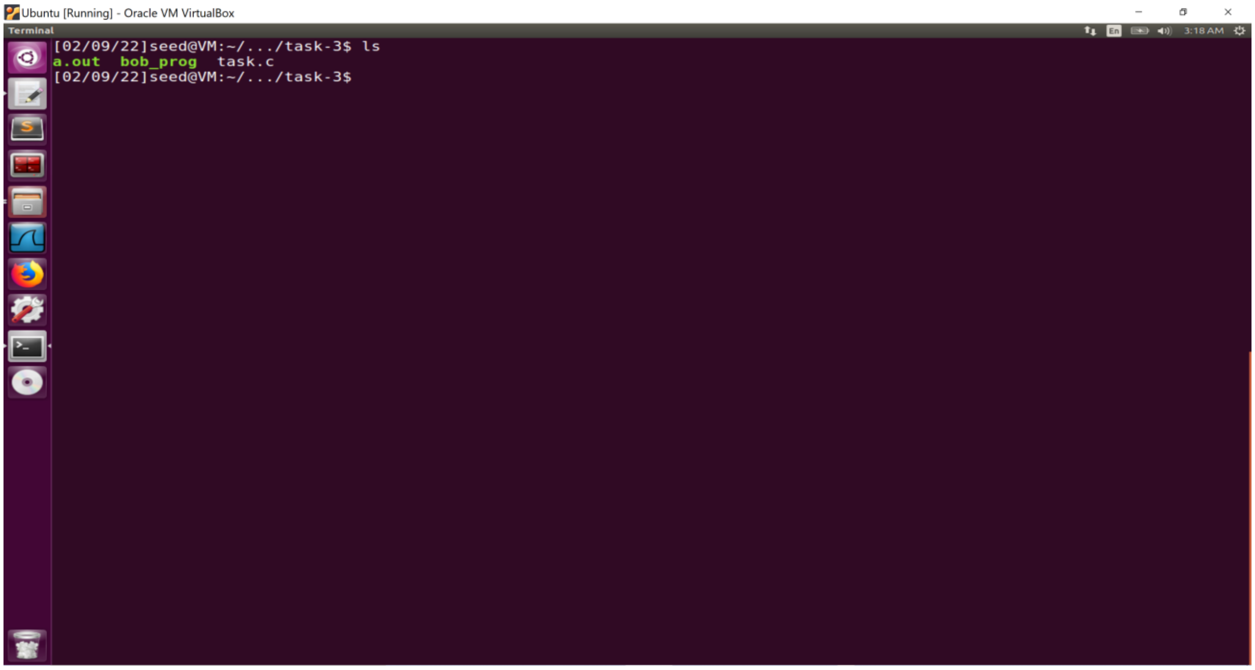
```
[02/09/22]seed@VM:~/.../task-2$ ls
mydash  myzsh
[02/09/22]seed@VM:~/.../task-2$ rm myzsh
rm: remove write-protected regular file 'myzsh'? Yes
[02/09/22]seed@VM:~/.../task-2$ which zsh
/usr/bin/zsh
[02/09/22]seed@VM:~/.../task-2$ cp /usr/bin/zsh myzsh
[02/09/22]seed@VM:~/.../task-2$ ls -la
total 920
drwxrwxr-x 2 seed seed   4096 Feb  9 02:58 .
drwxrwxr-x 6 seed seed   4096 Feb  8 22:28 ..
-rwxr-xr-x 1 root seed 173644 Feb  8 23:49 mydash
-rwxr-xr-x 1 seed seed 756476 Feb  9 02:58 myzsh
[02/09/22]seed@VM:~/.../task-2$ sudo shown root myzsh
sudo: shown: command not found
[02/09/22]seed@VM:~/.../task-2$ sudo chown root myzsh
[02/09/22]seed@VM:~/.../task-2$ sudo chown 4755 myzsh
[02/09/22]seed@VM:~/.../task-2$ ls -la
total 920
drwxrwxr-x 2 seed seed   4096 Feb  9 02:58 .
drwxrwxr-x 6 seed seed   4096 Feb  8 22:28 ..
-rwxr-xr-x 1 root seed 173644 Feb  8 23:49 mydash
-rwxr-xr-x 1 4755 seed 756476 Feb  9 02:58 myzsh
[02/09/22]seed@VM:~/.../task-2$ ps
  PID TTY          TIME CMD
 4289 pts/4    00:00:00 bash
 4666 pts/4    00:00:00 yes
 4747 pts/4    00:00:00 ps
[02/09/22]seed@VM:~/.../task-2$ zsh
VM% ps
  PID TTY          TIME CMD
 4289 pts/4    00:00:00 bash
 4666 pts/4    00:00:00 yes
 4753 pts/4    00:00:00 zsh
 4756 pts/4    00:00:00 ps
VM% id
uid=1000(seed) gid=1000(seed) groups=1000(seed),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),113(lpadmin),128(sambashare)
VM% 
```

Fig 2.10

Task 3: Invoking External Programs Using system() versus execve()

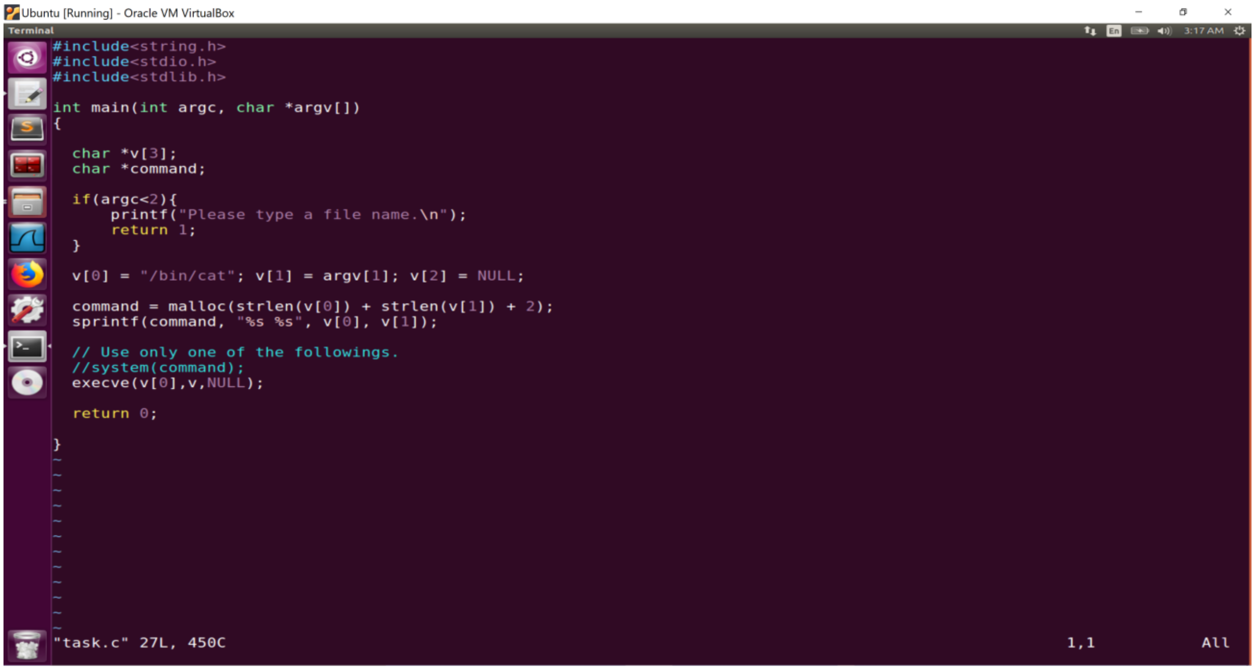
Although system() and execve() can both be used to run new programs, system() is quite dangerous if used in a privileged program, such as Set-UID programs. We have seen how the PATH environment variable affect the behavior of system(), because the variable affects how the shell works. execve() does not have the problem, because it does not invoke shell. Invoking shell has another dangerous conse-quence, and this time, it has nothing to do with environment variables. Let us look at the following scenario. Bob works for an auditing agency, and he needs to investigate a company for a suspected fraud. For the investigation purpose, Bob needs to be able to read all the files in the company's Unix system; on the other hand, to protect the integrity of the system, Bob should not be able to modify any file. To achieve this goal, Vince, the superuser of the system, wrote a special set-root-uid program (see below), and then gave the executable permission to Bob. This program requires Bob to type a file name at the command line, and then it will run /bin/cat to display the specified file. Since the program is running as a root, it can display any file Bob specifies. However, since the program has no write operations, Vince is very sure that Bob cannot use this special program to modify any file.

Step 1: Compile the above program, make it a root-owned Set-UID program. The program will use system () to invoke the command. If you were Bob, can you compromise the integrity of the system? For example, can you remove a file that is not writable to you?



```
[02/09/22]seed@VM:~/.../task-3$ ls
a.out  bob_prog  task.c
[02/09/22]seed@VM:~/.../task-3$
```

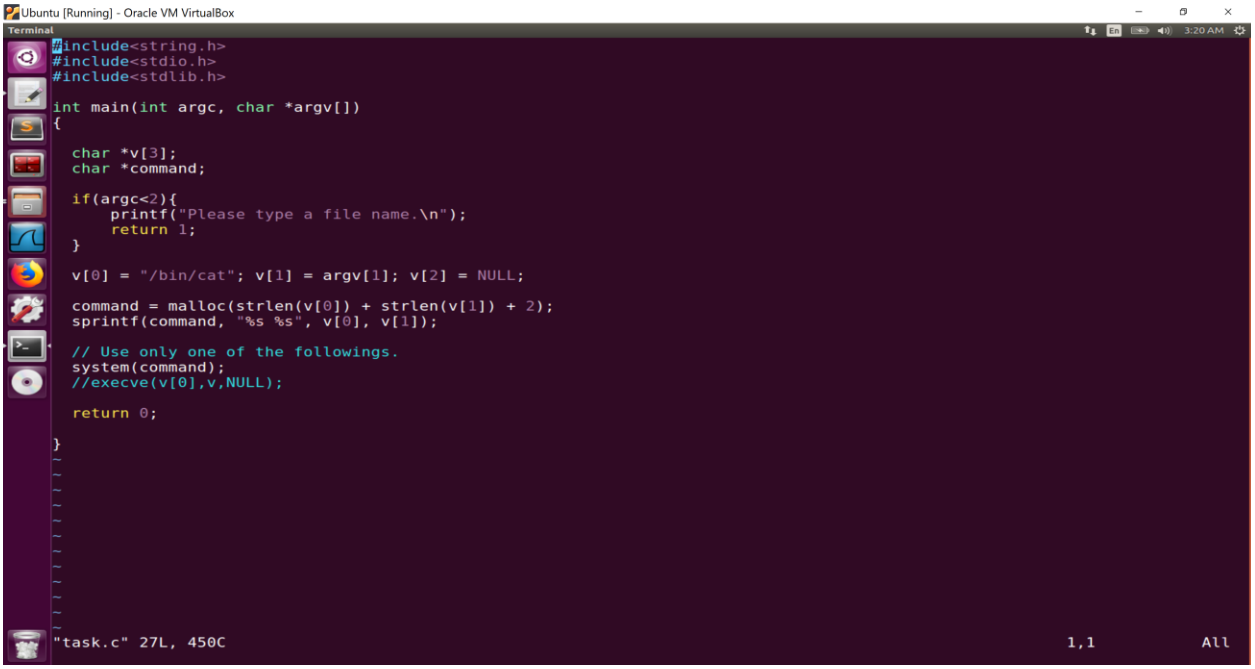
Fig 2.11



```
#include<string.h>
#include<stdio.h>
#include<stdlib.h>

int main(int argc, char *argv[])
{

  char *v[3];
  char *command;

  if(argc<2){
      printf("Please type a file name.\n");
      return 1;
  }

  v[0] = "/bin/cat"; v[1] = argv[1]; v[2] = NULL;

  command = malloc(strlen(v[0]) + strlen(v[1]) + 2);
  sprintf(command, "%s %s", v[0], v[1]);

  // Use only one of the followings.
  //system(command);
  execve(v[0],v,NULL);

  return 0;

}
~
~
~
~
~
~
~
~
~
~
~
~
"task.c" 27L, 450C                                                  1,1           All
```

Fig 2.12

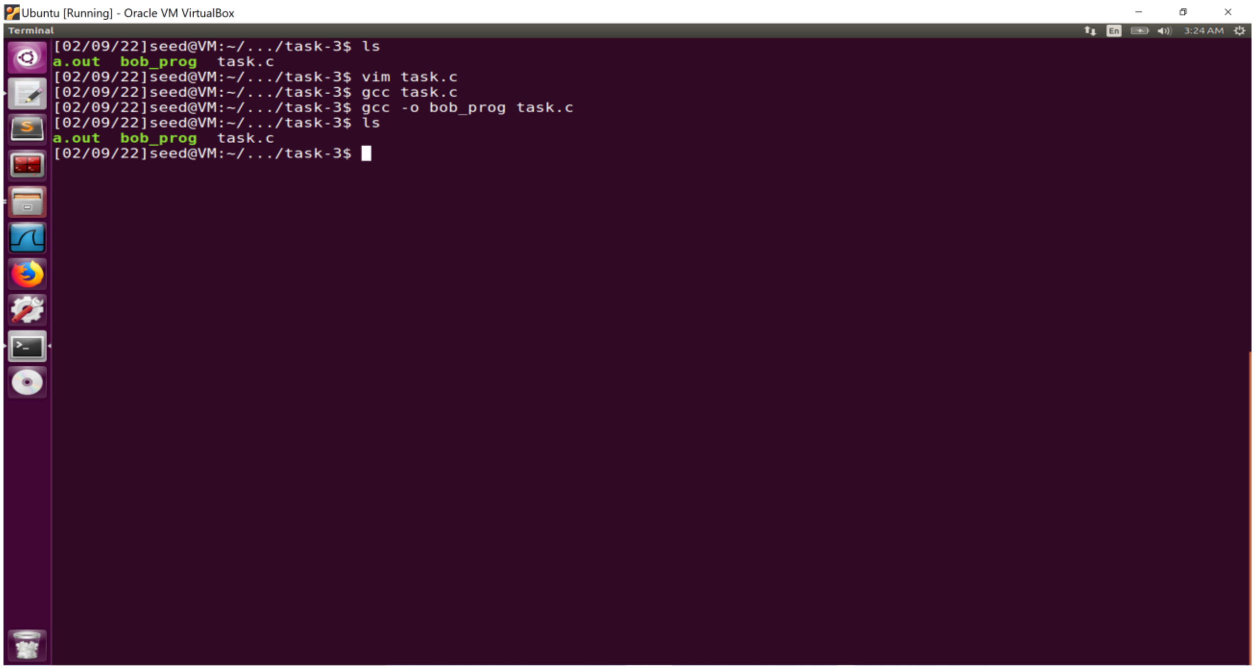
In Fig 2.12, you will find that “execve(v[0], v, NULL)” is not commented.



```
#include<string.h>
#include<stdio.h>
#include<stdlib.h>

int main(int argc, char *argv[])
{

  char *v[3];
  char *command;

  if(argc<2){
      printf("Please type a file name.\n");
      return 1;
  }

  v[0] = "/bin/cat"; v[1] = argv[1]; v[2] = NULL;

  command = malloc(strlen(v[0]) + strlen(v[1]) + 2);
  sprintf(command, "%s %s", v[0], v[1]);

  // Use only one of the followings.
  system(command);
  //execve(v[0],v,NULL);

  return 0;

}
~
~
~
~
~
~
~
~
~
~
~
~
"task.c" 27L, 450C                                                         1,1           All
```

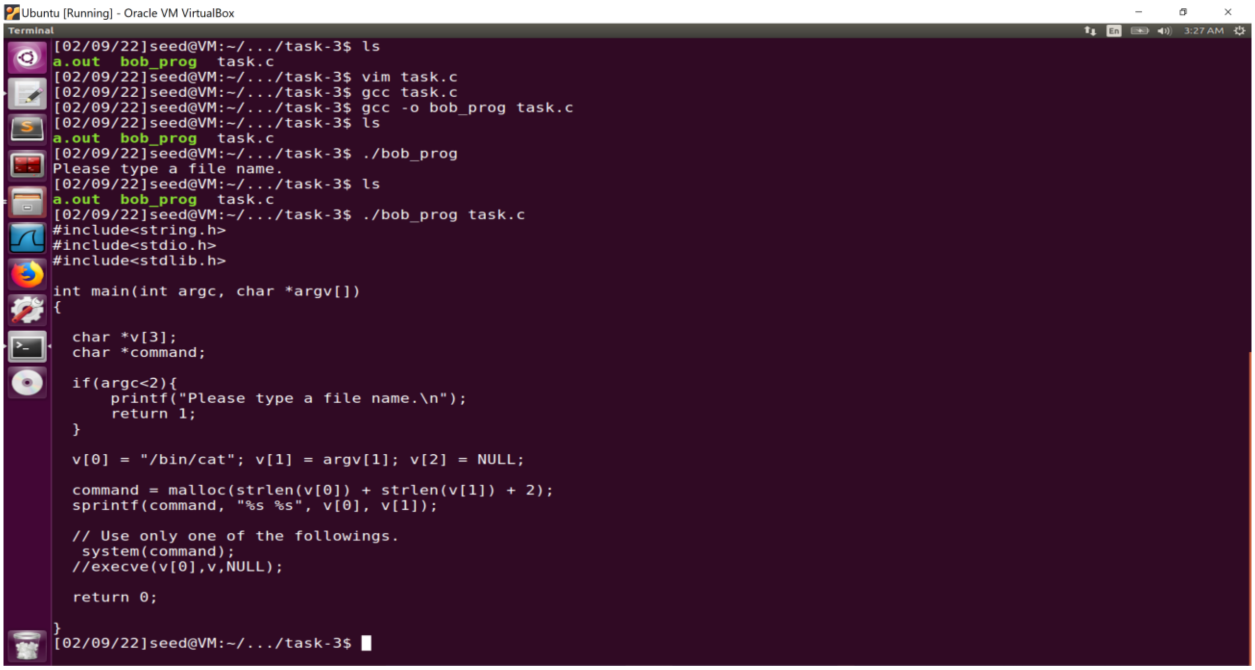
Fig 2.13

In Fig 2.13, you will see that I have change “execve(v[0], v, NULL)” to comment and change system(command) uncomment.



```
[02/09/22]seed@VM:~/.../task-3$ ls
a.out  bob_prog  task.c
[02/09/22]seed@VM:~/.../task-3$ vim task.c
[02/09/22]seed@VM:~/.../task-3$ gcc task.c
[02/09/22]seed@VM:~/.../task-3$ gcc -o bob_prog task.c
[02/09/22]seed@VM:~/.../task-3$ ls
a.out  bob_prog  task.c
[02/09/22]seed@VM:~/.../task-3$ 
```

Fig 2.14

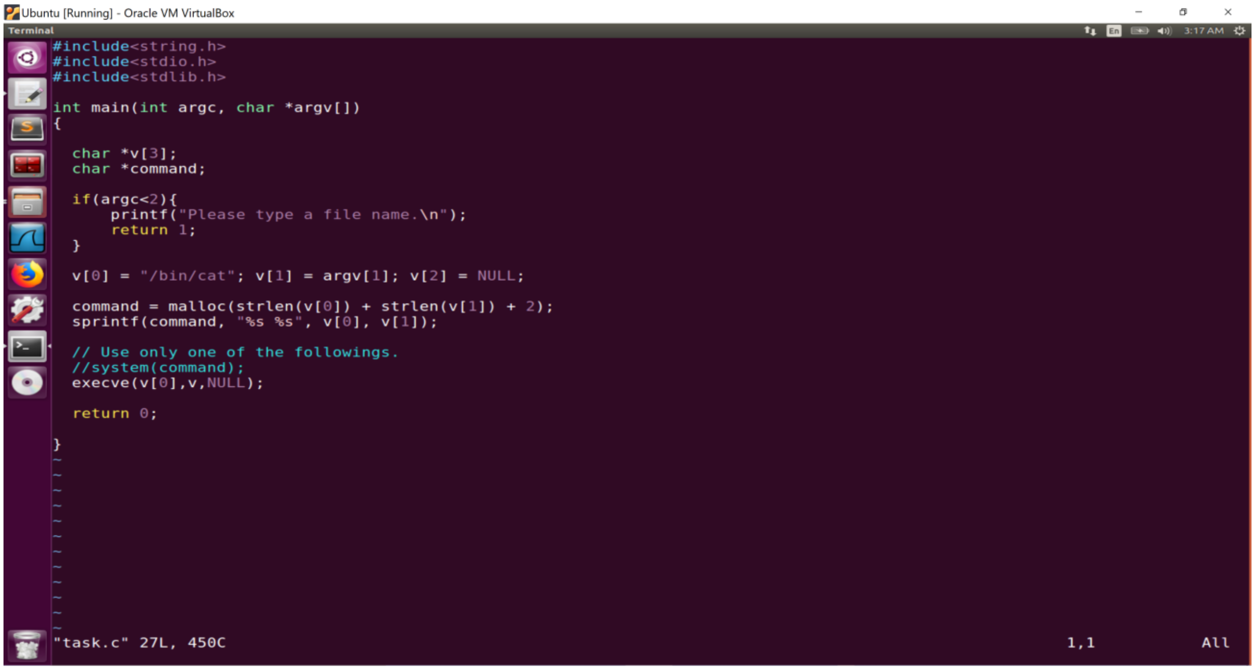


```
[02/09/22]seed@VM:~/.../task-3$ ls
a.out  bob_prog  task.c
[02/09/22]seed@VM:~/.../task-3$ vim task.c
[02/09/22]seed@VM:~/.../task-3$ gcc task.c
[02/09/22]seed@VM:~/.../task-3$ gcc -o bob_prog task.c
[02/09/22]seed@VM:~/.../task-3$ ls
a.out  bob_prog  task.c
[02/09/22]seed@VM:~/.../task-3$ ./bob_prog
Please type a file name.
[02/09/22]seed@VM:~/.../task-3$ ls
a.out  bob_prog  task.c
[02/09/22]seed@VM:~/.../task-3$ ./bob_prog task.c
#include<string.h>
#include<stdio.h>
#include<stdlib.h>

int main(int argc, char *argv[])
{

  char *v[3];
  char *command;

  if(argc<2){
      printf("Please type a file name.\n");
      return 1;
  }

  v[0] = "/bin/cat"; v[1] = argv[1]; v[2] = NULL;

  command = malloc(strlen(v[0]) + strlen(v[1]) + 2);
  sprintf(command, "%s %s", v[0], v[1]);

  // Use only one of the followings.
   system(command);
  //execve(v[0],v,NULL);

  return 0;

}
[02/09/22]seed@VM:~/.../task-3$ 
```

Fig 2.15

In Fig 2.15, you will see that when command "./bob_prog task.c" runs, the output will be the program.

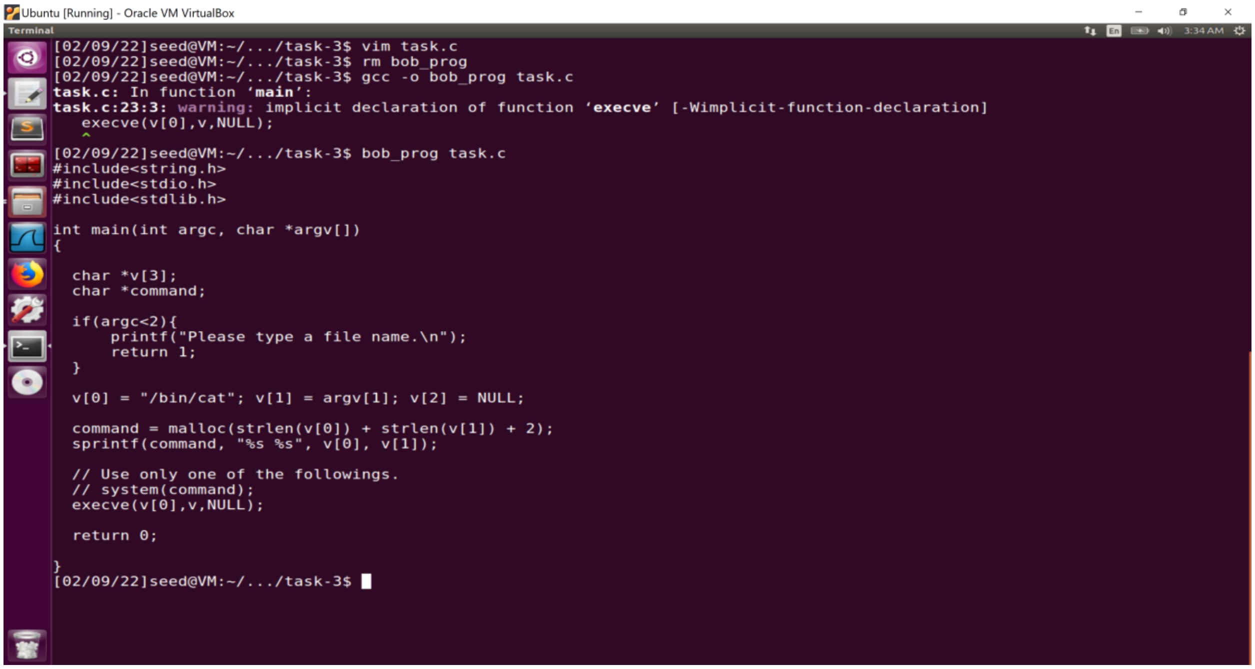
Step 2: Comment out the system(command) statement, and uncomment the execve() statement; the program will use execve() to invoke the command. Compile the program and make it a root-owned Set - UID. Do your attacks in Step 1 still work? Please describe and explain your observations.



```
#include<string.h>
#include<stdio.h>
#include<stdlib.h>

int main(int argc, char *argv[])
{

  char *v[3];
  char *command;

  if(argc<2){
      printf("Please type a file name.\n");
      return 1;
  }

  v[0] = "/bin/cat"; v[1] = argv[1]; v[2] = NULL;

  command = malloc(strlen(v[0]) + strlen(v[1]) + 2);
  sprintf(command, "%s %s", v[0], v[1]);

  // Use only one of the followings.
  //system(command);
  execve(v[0],v,NULL);

  return 0;

}
~
~
~
~
~
~
~
~
~
~
~
~
"task.c" 27L, 450C                                                    1,1           All
```

Fig 2.16

In Fig 2.16, you will see that I change the “system(command)” to comment and “execve(v[0], v, NULL)” to uncomment.



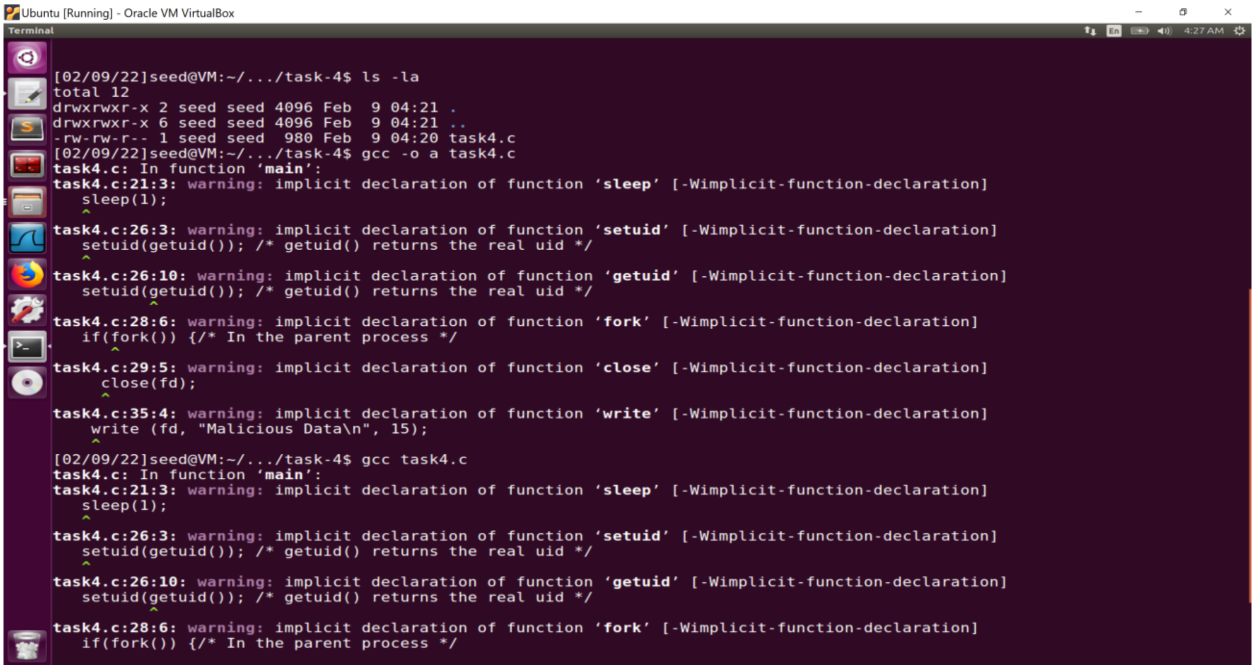
```
[02/09/22]seed@VM:~/.../task-3$ vim task.c
[02/09/22]seed@VM:~/.../task-3$ rm bob_prog
[02/09/22]seed@VM:~/.../task-3$ gcc -o bob_prog task.c
task.c: In function ‘main’:
task.c:23:3: warning: implicit declaration of function ‘execve’ [-Wimplicit-function-declaration]
   execve(v[0],v,NULL);
   ^
[02/09/22]seed@VM:~/.../task-3$ bob_prog task.c
#include<string.h>
#include<stdio.h>
#include<stdlib.h>

int main(int argc, char *argv[])
{

  char *v[3];
  char *command;

  if(argc<2){
      printf("Please type a file name.\n");
      return 1;
  }

  v[0] = "/bin/cat"; v[1] = argv[1]; v[2] = NULL;

  command = malloc(strlen(v[0]) + strlen(v[1]) + 2);
  sprintf(command, "%s %s", v[0], v[1]);

  // Use only one of the followings.
  // system(command);
  execve(v[0],v,NULL);

  return 0;

}
[02/09/22]seed@VM:~/.../task-3$ 
```

Fig 2.17

In Fig 2.17, you will see that when I run the command “gcc -o bob_prog task.c” the output is warning.

Task 4: Capability Leaking

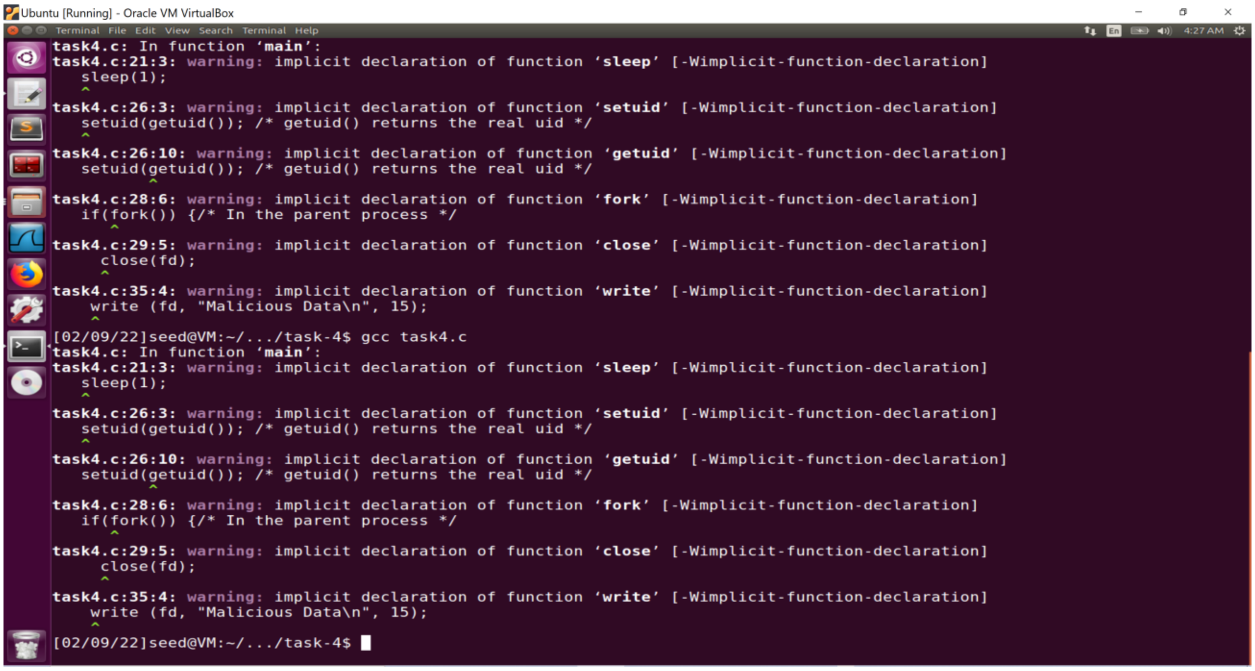
To follow the Principle of Least Privilege, Set-UID programs often permanently relinquish their root privileges if such privileges are not needed anymore. Moreover, sometimes, the program needs to hand over its control to the user; in this case, root privileges must be revoked. The setuid() system call can be used to revoke the privileges. According to the manual, “setuid() sets the effective user ID of the calling process. If the effective UID of the caller is root, the real UID and saved set-user-ID are also set”. Therefore, if a Set-UID program with effective UID 0 calls setuid(n), the process will become a normal process, with all its UIDs being set to n. When revoking the privilege, one of the common mistakes is capability leaking. The process may have gained some privileged capabilities when it was still privileged; when the privilege is downgraded, if the program does not clean up those capabilities, they may still be accessible by the non-privileged process. In other words, although the effective user ID of the process becomes non-privileged, the process is still privileged because it possesses privileged capabilities. Compile the following program, change its owner to root, and make it a Set-UID program. Run the program as a normal user and describe what you have observed. Will the file /etc/zzz be modified? Please explain your observation.



```
[02/09/22]seed@VM:~/.../task-4$ ls -la
total 12
drwxrwxr-x 2 seed seed 4096 Feb  9 04:21 .
drwxrwxr-x 6 seed seed 4096 Feb  9 04:21 ..
-rw-rw-r-- 1 seed seed  980 Feb  9 04:20 task4.c
[02/09/22]seed@VM:~/.../task-4$ gcc -o a task4.c
task4.c: In function ‘main’:
task4.c:21:3: warning: implicit declaration of function ‘sleep’ [-Wimplicit-function-declaration]
   sleep(1);
   ^
task4.c:26:3: warning: implicit declaration of function ‘setuid’ [-Wimplicit-function-declaration]
   setuid(getuid()); /* getuid() returns the real uid */
   ^
task4.c:26:10: warning: implicit declaration of function ‘getuid’ [-Wimplicit-function-declaration]
   setuid(getuid()); /* getuid() returns the real uid */
          ^
task4.c:28:6: warning: implicit declaration of function ‘fork’ [-Wimplicit-function-declaration]
   if(fork()) {/* In the parent process */
      ^
task4.c:29:5: warning: implicit declaration of function ‘close’ [-Wimplicit-function-declaration]
     close(fd);
     ^
task4.c:35:4: warning: implicit declaration of function ‘write’ [-Wimplicit-function-declaration]
    write (fd, "Malicious Data\n", 15);
    ^
[02/09/22]seed@VM:~/.../task-4$ gcc task4.c
task4.c: In function ‘main’:
task4.c:21:3: warning: implicit declaration of function ‘sleep’ [-Wimplicit-function-declaration]
   sleep(1);
   ^
task4.c:26:3: warning: implicit declaration of function ‘setuid’ [-Wimplicit-function-declaration]
   setuid(getuid()); /* getuid() returns the real uid */
   ^
task4.c:26:10: warning: implicit declaration of function ‘getuid’ [-Wimplicit-function-declaration]
   setuid(getuid()); /* getuid() returns the real uid */
          ^
task4.c:28:6: warning: implicit declaration of function ‘fork’ [-Wimplicit-function-declaration]
   if(fork()) {/* In the parent process */
```

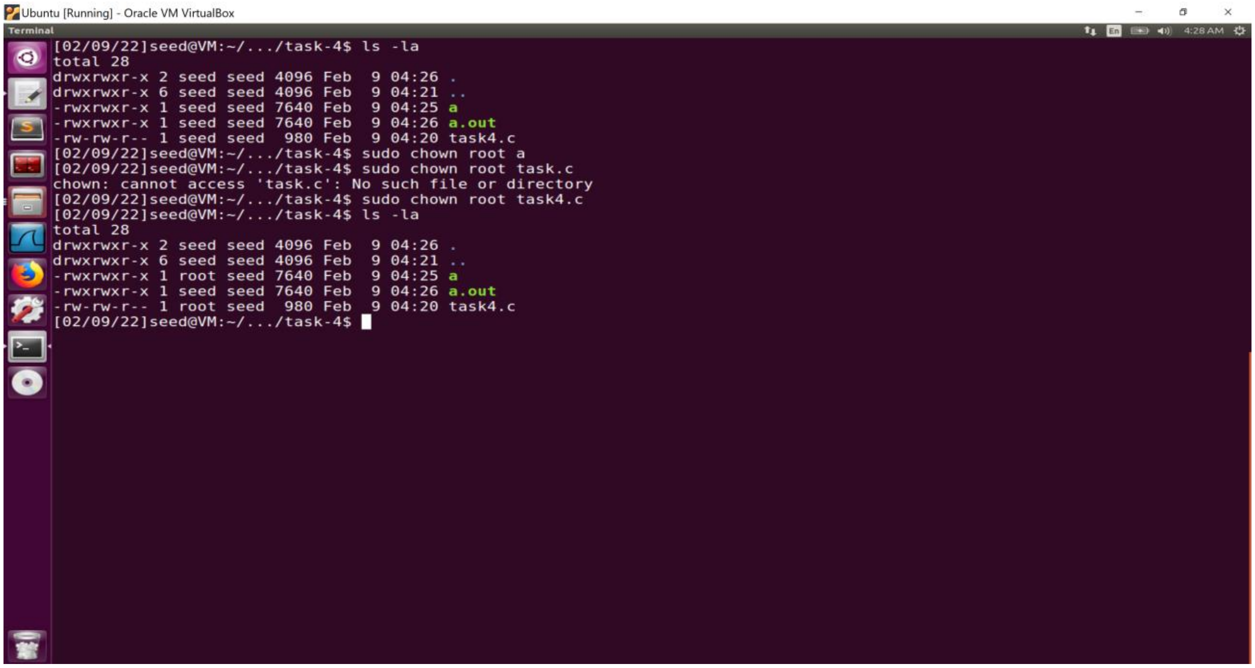
Fig 2.18

In Fig 2.18, you will see the task4.c file. I compile the task4.c as a.


Ubuntu [Running] - Oracle VM VirtualBox
Terminal File Edit View Search Terminal Help
4:27 AM

```
task4.c: In function ‘main’:
task4.c:21:3: warning: implicit declaration of function ‘sleep’ [-Wimplicit-function-declaration]
   sleep(1);
   ^
task4.c:26:3: warning: implicit declaration of function ‘setuid’ [-Wimplicit-function-declaration]
   setuid(getuid()); /* getuid() returns the real uid */
   ^
task4.c:26:10: warning: implicit declaration of function ‘getuid’ [-Wimplicit-function-declaration]
   setuid(getuid()); /* getuid() returns the real uid */
          ^
task4.c:28:6: warning: implicit declaration of function ‘fork’ [-Wimplicit-function-declaration]
   if(fork()) {/* In the parent process */
      ^
task4.c:29:5: warning: implicit declaration of function ‘close’ [-Wimplicit-function-declaration]
     close(fd);
     ^
task4.c:35:4: warning: implicit declaration of function ‘write’ [-Wimplicit-function-declaration]
    write (fd, "Malicious Data\n", 15);
    ^
[02/09/22]seed@VM:~/.../task-4$ gcc task4.c
task4.c: In function ‘main’:
task4.c:21:3: warning: implicit declaration of function ‘sleep’ [-Wimplicit-function-declaration]
   sleep(1);
   ^
task4.c:26:3: warning: implicit declaration of function ‘setuid’ [-Wimplicit-function-declaration]
   setuid(getuid()); /* getuid() returns the real uid */
   ^
task4.c:26:10: warning: implicit declaration of function ‘getuid’ [-Wimplicit-function-declaration]
   setuid(getuid()); /* getuid() returns the real uid */
          ^
task4.c:28:6: warning: implicit declaration of function ‘fork’ [-Wimplicit-function-declaration]
   if(fork()) {/* In the parent process */
      ^
task4.c:29:5: warning: implicit declaration of function ‘close’ [-Wimplicit-function-declaration]
     close(fd);
     ^
task4.c:35:4: warning: implicit declaration of function ‘write’ [-Wimplicit-function-declaration]
    write (fd, "Malicious Data\n", 15);
    ^
[02/09/22]seed@VM:~/.../task-4$ 
```

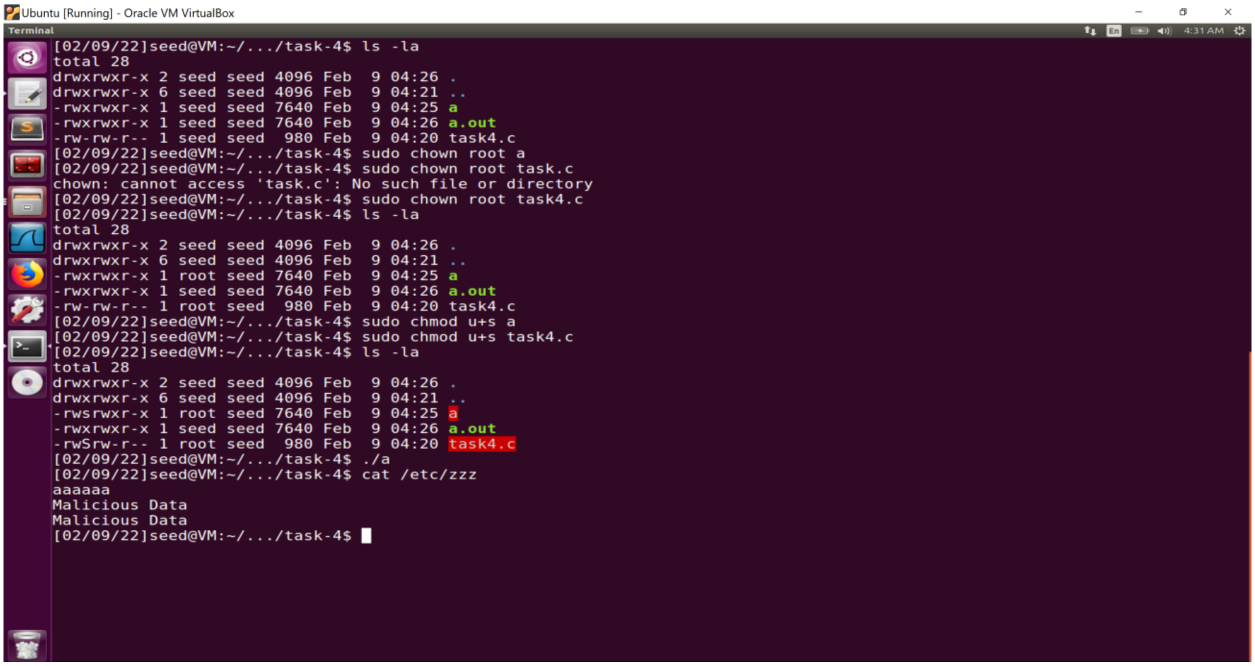
Fig 2.19



```
[02/09/22]seed@VM:~/.../task-4$ ls -la
total 28
drwxrwxr-x 2 seed seed 4096 Feb  9 04:26 .
drwxrwxr-x 6 seed seed 4096 Feb  9 04:21 ..
-rwxrwxr-x 1 seed seed 7640 Feb  9 04:25 a
-rwxrwxr-x 1 seed seed 7640 Feb  9 04:26 a.out
-rw-rw-r-- 1 seed seed  980 Feb  9 04:20 task4.c
[02/09/22]seed@VM:~/.../task-4$ sudo chown root a
[02/09/22]seed@VM:~/.../task-4$ sudo chown root task.c
chown: cannot access 'task.c': No such file or directory
[02/09/22]seed@VM:~/.../task-4$ sudo chown root task4.c
[02/09/22]seed@VM:~/.../task-4$ ls -la
total 28
drwxrwxr-x 2 seed seed 4096 Feb  9 04:26 .
drwxrwxr-x 6 seed seed 4096 Feb  9 04:21 ..
-rwxrwxr-x 1 root seed 7640 Feb  9 04:25 a
-rwxrwxr-x 1 seed seed 7640 Feb  9 04:26 a.out
-rw-rw-r-- 1 root seed  980 Feb  9 04:20 task4.c
[02/09/22]seed@VM:~/.../task-4$ 
```

Fig 2.20

In Fig 2.20, I change the owner of compilation file that is “a” as well as “task4.c”. Also, I change the permission of both the files.



```
[02/09/22]seed@VM:~/.../task-4$ ls -la
total 28
drwxrwxr-x 2 seed seed 4096 Feb  9 04:26 .
drwxrwxr-x 6 seed seed 4096 Feb  9 04:21 ..
-rwxrwxr-x 1 seed seed 7640 Feb  9 04:25 a
-rwxrwxr-x 1 seed seed 7640 Feb  9 04:26 a.out
-rw-rw-r-- 1 seed seed  980 Feb  9 04:20 task4.c
[02/09/22]seed@VM:~/.../task-4$ sudo chown root a
[02/09/22]seed@VM:~/.../task-4$ sudo chown root task.c
chown: cannot access 'task.c': No such file or directory
[02/09/22]seed@VM:~/.../task-4$ sudo chown root task4.c
[02/09/22]seed@VM:~/.../task-4$ ls -la
total 28
drwxrwxr-x 2 seed seed 4096 Feb  9 04:26 .
drwxrwxr-x 6 seed seed 4096 Feb  9 04:21 ..
-rwxrwxr-x 1 root seed 7640 Feb  9 04:25 a
-rwxrwxr-x 1 seed seed 7640 Feb  9 04:26 a.out
-rw-rw-r-- 1 root seed  980 Feb  9 04:20 task4.c
[02/09/22]seed@VM:~/.../task-4$ sudo chmod u+s a
[02/09/22]seed@VM:~/.../task-4$ sudo chmod u+s task4.c
[02/09/22]seed@VM:~/.../task-4$ ls -la
total 28
drwxrwxr-x 2 seed seed 4096 Feb  9 04:26 .
drwxrwxr-x 6 seed seed 4096 Feb  9 04:21 ..
-rwsrwxr-x 1 root seed 7640 Feb  9 04:25 a
-rwxrwxr-x 1 seed seed 7640 Feb  9 04:26 a.out
-rwSrw-r-- 1 root seed  980 Feb  9 04:20 task4.c
[02/09/22]seed@VM:~/.../task-4$ ./a
[02/09/22]seed@VM:~/.../task-4$ cat /etc/zzz
aaaaaa
Malicious Data
Malicious Data
[02/09/22]seed@VM:~/.../task-4$ 
```

Fig 2.21

In Fig 2.21, you will find that after running command “cat /etc/zzz” the output is:

“aaaaaa”

“Malicious Data”

“Malicious Data”