Отчёт по лабораторной работе 5

Дискреционное разграничение прав в Linux. Исследование влияниядополнительных атрибутов

Хамарнех Майя Ясер

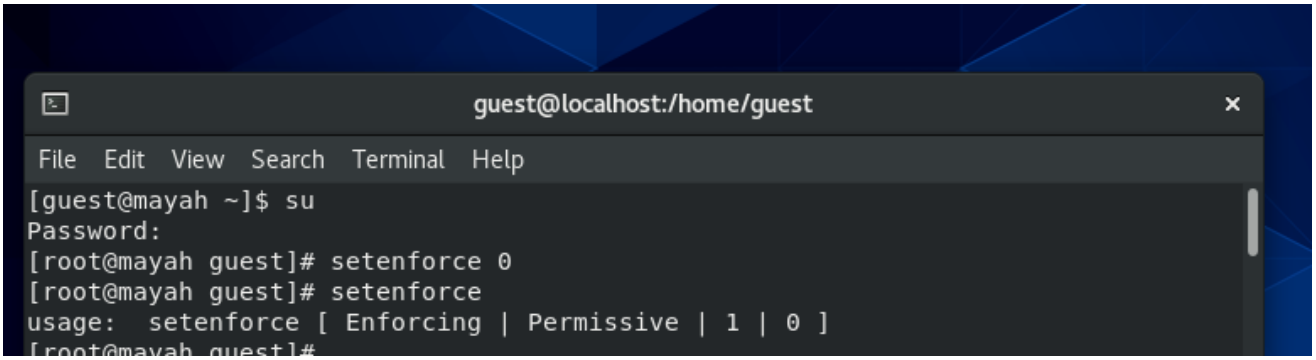
Содержание

1. Цель работы

Изучение механизмов изменения идентификаторов, применения SetUID- и Sticky-битов. Получение практических навыков работы в консоли с дополнительными атрибутами. Рассмотрение работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

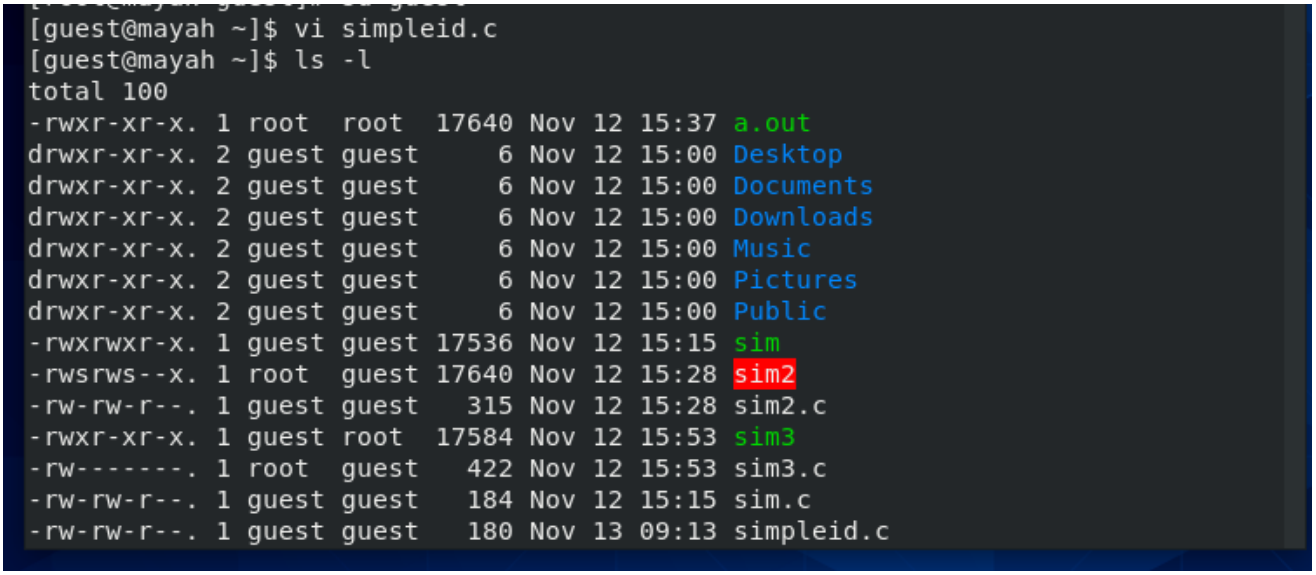
2. Выполнение лабораторной работы

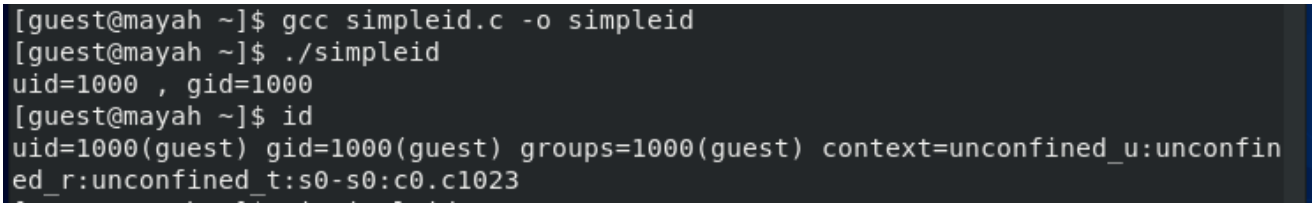
at first i disabled the SELinux system.



от пользователя-гостя я создала файл с с именем simpleid.c , а затем я написала файл и скомпилировала его

после этого я запустила программу, затем запустил команду "id" и сравнила результаты обоих, и оба результата были идентичны





Я создала другой файл с именем simpleid2, произвел дублирование программы (simpleid), добавив выходные данные действительных идентификаторов, затем я скомпилировала и запустила simpleid2.c.

```
[guest@mayah ~]$ vi simpleid2.c
[guest@mayah ~]$ gcc simpleid2.c -o simpleid2
[guest@mayah ~]$ ./simpleid2
e_uid=1000 , e_gid=1000
real_uid=1000 , real_gid=1000
```

Я переключился на пользователя root и изменила владельца файла simpleid2.c на root, пока группа все еще является guest, используя:

chown root:guest /home/guest/simpleid2

затем я изменил режим на SUID, чтобы дать пользователю временные разрешения на запуск simpleid2 с разрешениями владельца файла и root, а не пользователя, который его запускает

chmod u + s / домашний / гость / simpleid2

```
[root@mayah guest]# chown root:guest /home/guest/simpleid2
[root@mayah guest]# chmod u+s /home/guest/simpleid2
```

от гостя мы проверили с помощью команды ls новые атрибуты и владельца

затем мы скомпилировали и запустили файл, затем мы запустили команду id и сравнили оба результата, чтобы увидеть, что теперь он принадлежит root

```
[guest@mayah ~]$ ls -l simpleid2
-rwsrwxr-x. 1 root guest 17648 Nov 13 09:23 simpleid2
[guest@mayah ~]$ ./simpleid2
e_uid=0 , e_gid=1000
real_uid=1000 , real_gid=1000
[guest@mayah ~]$ id
uid=1000(guest) gid=1000(guest) groups=1000(guest) context=unconfined_u:unconfin
ed_r:unconfined_t:s0-s0:c0.c1023
```

Проделала тоже самое относительно SetGID-бита.

```
[root@mayah guest]# chmod g+s /home/guest simpleid2
[root@mayah guest]# su guest
[guest@mayah ~]$ ls -l simpleid2
-rwsrwsr-x. 1 root guest 17648 Nov 13 09:23 simpleid2
[guest@mayah ~]$ id
uid=1000(guest) gid=1000(guest) groups=1000(guest) context=unconfined_u:unconfin
ed_r:unconfined_t:s0-s0:c0.c1023
[guest@mayah ~]$ ./simpleid2
e_uid=0 , e_gid=1000
```

затем мы написали файл readfile.c для чтения файла и скомпилировали его

```
[guest@mayah ~]$ vi readfile.c
[guest@mayah ~]$ gcc readfile.c -o reafile
[guest@mayah ~]$ su
Password:
```

затем мы изменили владельца файла на root и дали права на чтение только root, а не гостю или другим пользователям:

```
[root@mayah guest]# chown root /home/guest/readfile.c
[root@mayah guest]# chmod u+x  /home/guest/readfile.c
[root@mayah guest]# chmod g-rw  /home/guest/readfile.c
[root@mayah guest]# chmod o-r  /home/guest/readfile.c
[root@mayah guest]# su guest
```

мы проверили, может ли пользователь guest прочитать его, и он не смог

```
[guest@mayah ~]$ cat readfile.c
cat: readfile.c: Permission denied
```

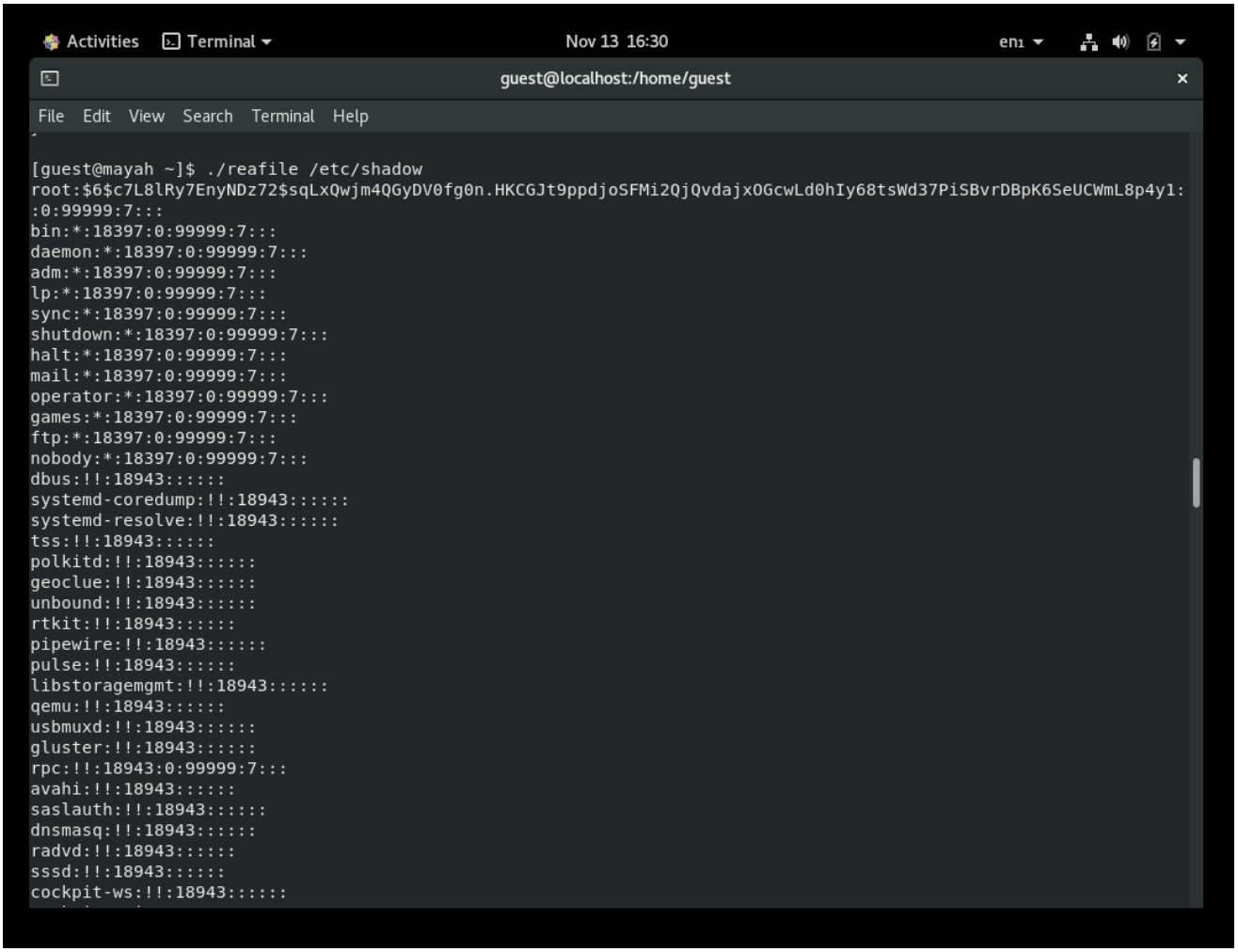
мы изменили владельца прочитанного файла и установили бит SetUiD.

```
[root@mayah guest]# chown root /home/guest/reafile
[root@mayah guest]# chmod u+s /home/guest/reafile
```

мы проверили, может ли файл reafile прочитать readfile.c, и он сработал

```
[root@mayah guest]# chmod u+s /home/guest/reafile
[root@mayah guest]# su guest
[guest@mayah ~]$ ./reafile readfile.c
#include <fcntl.h>
#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>


int main(int argc , char* argv[]){

        unsigned char buffer[16];
        size_t bytes_read;
        int i;

        int fd = open(argv[1] , O_RDONLY);
        do
        {
                bytes_read = read(fd , buffer , sizeof(buffer));

                for (i=0 ; i < bytes_read ; ++i ) printf("%c" , buffer[i]);
        }
```

мы снова проверили, может ли прочитанный файл прочитать файл / etc / shadow, и он сработал



мы проверили, установлен ли атрибут Sticky в каталоге / tmp, и он установил.
Используя ls -l / | grep tmp
затем от имени гостя мы создали файл file01.txt в каталоге / tmp и написали слово "test" внутри этого файла.
затем мы проверили атрибуты вновь созданного файла и разрешили чтение и запись для остальных

```
[guest@mayah root]$ ls -l / | grep tmp
drwxrwxrwt.  15 root root 4096 Nov 13 13:52 tmp
[guest@mayah root]$ nkf
bash: nkf: command not found...
[guest@mayah root]$ ls -l / | grep tmp
drwxrwxrwt.  15 root root 4096 Nov 13 13:52 tmp
[guest@mayah root]$ echo "test" > /tmp/file1.txt
[guest@mayah root]$ ls -l /tmp/file1.txt
-rw-rw-r--. 1 guest guest 5 Nov 13 14:08 /tmp/file1.txt
[guest@mayah root]$ su
Password:
[root@mayah ~]# chmod o+rw /tmp/file.txt
chmod: cannot access '/tmp/file.txt': No such file or directory
[root@mayah ~]# chmod o+rw /tmp/file1.txt
```

```
[guest@mayah ~]$ ls -l /tmp/file1.txt
-rw-rw-rw-. 1 guest guest 5 Nov 13 14:08 /tmp/file1.txt
```

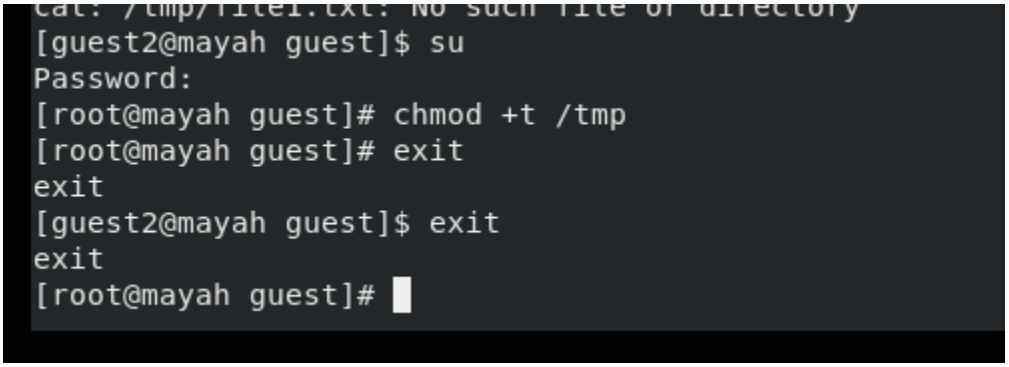
мы добавили еще одного пользователя по имени guest2 и оттуда мы попытались добавить несколько слов в файл file1.txt в tmp, каждый раз, когда мы писали слово, он показывал нам это слово, стирая все предыдущие слова
но когда мы пытались удалить файл, он не давал нам на это разрешения

```
[root@mayah guest]# useradd guest2
[root@mayah guest]# su guest2
[guest2@mayah guest]$ cat /tmp/file1.txt
test
[guest2@mayah guest]$ echo "test2" > /tmp/file1.txt
[guest2@mayah guest]$ cat /tmp/fil1.txt
cat: /tmp/fil1.txt: No such file or directory
[guest2@mayah guest]$ cat /tmp/file1.txt
test2
[guest2@mayah guest]$ rm /tmp/file1.txt
rm: cannot remove '/tmp/file1.txt': Operation not permitted
```

затем мы удалили липкий бит, чтобы у него были разрешения, и мы повторили некоторые шаги и попытались удалить файл впоследствии, и это сработало, файл был удален

```
Password:
[root@mayah guest]# chmod -t /tmp
[root@mayah guest]# exit
exit
[guest2@mayah guest]$ ls -l / | grep tmp
drwxrwxrwx.  15 root root 4096 Nov 13 14:16 tmp
[guest2@mayah guest]$ echo "test" > /tmp/file1.txt
[guest2@mayah guest]$ ls -l / | grep tmp
drwxrwxrwx.  15 root root 4096 Nov 13 14:16 tmp
[guest2@mayah guest]$ cat /tmp/file1.txt
test
[guest2@mayah guest]$ echo "test4" >> /tmp/file1.txt
[guest2@mayah guest]$ cat /tmp/file1.txt
test
test4
[guest2@mayah guest]$ rm /tmp/file1.txt
[guest2@mayah guest]$ cat /tmp/file1.txt
cat: /tmp/file1.txt: No such file or directory
[guest2@mayah guest]$ su
Password:
```

используя корень суперпользователя, мы вернули липкий бит chmod + t и вышли



3. Вывод

мы узнали все о SUID и липких битах, а также о том, как их использовать для изменения разрешений в системе и между пользователями.
также как тщательно проверять эти атрибуты и разрешения и перемещаться между пользователями