

Лабораторная работа № 5.
Дискреционное разграничение прав в
Linux. Исследование влияния
дополнительных атрибутов

Ильин Никита Евгеньевич, НФИбд-01-19

Содержание

1	Цель работы	5
2	Последовательность выполнения работы	6
2.1	Создание программы	6
2.2	Исследование Sticky-бита	11
3	Выводы	16
4	Библиография	17

List of Figures

2.1	компиляция и запуск	7
2.2	проверка	7
2.3	Права	8
2.4	Установка атрибутов	9
2.5	Код программы	9
2.6	смена владельца	10
2.7	ошибка доступа	10
2.8	смена владельца	11
2.9	проверка возможности чтения файла	11
2.10	попытка прочитать файл	12
2.11	проверка содержимого	13
2.12	проверка содержимого	13
2.13	удаление файла	14
2.14	возвращение атрибута	15

List of Tables

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 Последовательность выполнения работы

2.1 Создание программы

1. Войдите в систему от имени пользователя guest.
2. Создайте программу simpleid.c:

```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int
main ()
{
    uid_t uid = geteuid ();
    gid_t gid = getegid ();
    printf ("uid=%d, gid=%d\n", uid, gid);
    return 0;
}
```

3. Скомпилируйте программу и убедитесь, что файл программы создан:

```
gcc simpleid.c -o simpleid
```

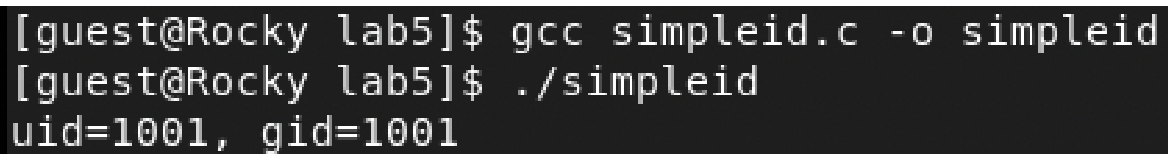
4. Выполните программу simpleid:

```
./simpleid
```

5. Выполните системную программу id:

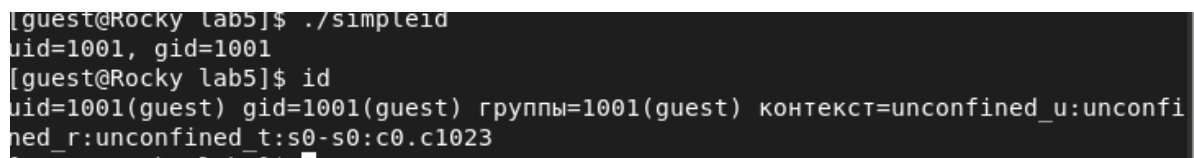
```
id
```

и сравните полученный вами результат с данными предыдущего пункта задания.



```
[guest@Rocky lab5]$ gcc simpleid.c -o simpleid
[guest@Rocky lab5]$ ./simpleid
uid=1001, gid=1001
```

Figure 2.1: компиляция и запуск



```
[guest@Rocky lab5]$ ./simpleid
uid=1001, gid=1001
[guest@Rocky lab5]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi
ned_r:unconfined_t:s0-s0:c0.c1023
```

Figure 2.2: проверка

6. Усложните программу, добавив вывод действительных идентификаторов:

```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int
main ()
{
    uid_t real_uid = getuid ();
```

```

uid_t e_uid = geteuid ();
gid_t real_gid = getgid ();
gid_t e_gid = getegid () ;
printf ("e_uid=%d, e_gid=%d\n", e_uid, e_gid);
printf ("real_uid=%d, real_gid=%d\n", real_uid,
, real_gid);
return 0;
}

```

Получившуюся программу назовите simpleid2.c.

7. Скомпилируйте и запустите simpleid2.c:

```

gcc simpleid2.c -o simpleid2
./simpleid2

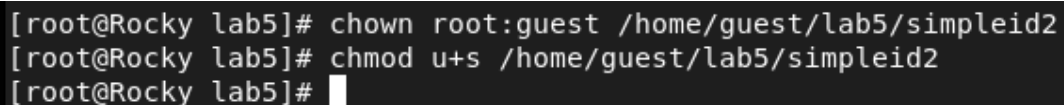
```

8. От имени суперпользователя выполните команды:

```

chown root:guest /home/guest/simpleid2
chmod u+s /home/guest/simpleid2

```



```

[root@Rocky lab5]# chown root:guest /home/guest/lab5/simpleid2
[root@Rocky lab5]# chmod u+s /home/guest/lab5/simpleid2
[root@Rocky lab5]# █

```

Figure 2.3: Права

9. Используйте `sudo` или повысьте временно свои права с помощью `su`. Поясните, что делают эти команды.
10. Выполните проверку правильности установки новых атрибутов и смены владельца файла `simpleid2`:


```
ls -l simpleid2
```



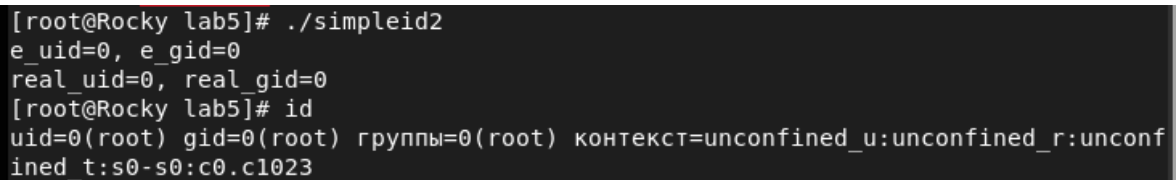
```
[root@Rocky lab5]# ls -l simpleid2
33575076 simpleid2
```

Figure 2.4: Установка атрибутов

11. Запустите simpleid2 и id:

```
./simpleid2
id
```

Сравните результаты.



```
[root@Rocky lab5]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@Rocky lab5]# id
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Figure 2.5: Код программы

12. Прodelайте тоже самое относительно SetGID-бита.

13. Создайте программу 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]);
}
while (bytes_read == sizeof (buffer));
close (fd);
return 0;
}

```

14. Откомпилируйте её.

```
gcc readfile.c -o readfile
```

15. Смените владельца у файла readfile.c (или любого другого текстового файла в системе) и измените права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог.



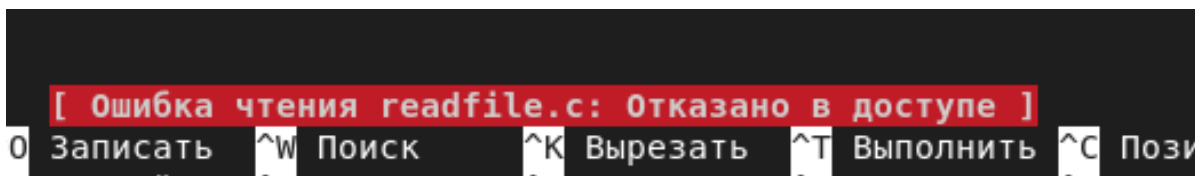
```

[root@Rocky lab5]# chmod 600 readfile.c
[root@Rocky lab5]#

```

Figure 2.6: смена владельца

16. Проверьте, что пользователь guest не может прочитать файл readfile.c.



```

[ Ошибка чтения readfile.c: Отказано в доступе ]
0 Записать ^W Поиск ^K Вырезать ^T Выполнить ^C Пози

```

Figure 2.7: ошибка доступа

17. Смените у программы readfile владельца и установите SetU'D-бит.

```
[root@Rocky lab5]# chmod 600 readfile.c
[root@Rocky lab5]# chown root:guest readfile
[root@Rocky lab5]# chmod u+s readfile
[root@Rocky lab5]# ls -l
итого 84
-rwsr-xr-x. 1 root  guest 80504 окт  3 18:04 readfile
-rw----- 1 root  guest  402 окт  3 18:04 readfile.c
-rwxrwxr-x. 1 guest guest 80488 окт  3 17:57 simpleid
-rwsrwxr-x. 1 root  guest 80584 окт  3 18:00 simpleid2
-rw-rw-r-- 1 guest guest  303 окт  3 18:00 simpleid2.c
-rw-rw-r-- 1 guest guest  175 окт  3 17:57 simpleid.c
[root@Rocky lab5]#
```

Figure 2.8: смена владельца

18. Проверьте, может ли программа readfile прочитать файл readfile.c?

19. Проверьте, может ли программа readfile прочитать файл /etc/etc/shadow?

```
[root@Rocky lab5]# ./readfile /etc/shadow
root:$6$ZrcK5nRqYyTgG90r$9q8koDp5M8QEHY.vecNYvyA97hnh1rh0tAQ9TSw2B8dgPKrYkL6bASo
FHTz9520zZrNiChCI3v64S2boCwIxj/::0:99999:7:::
bin:!:19123:0:99999:7:::
daemon:!:19123:0:99999:7:::
adm:!:19123:0:99999:7:::
lp:!:19123:0:99999:7:::
sync:!:19123:0:99999:7:::
shutdown:!:19123:0:99999:7:::
halt:!:19123:0:99999:7:::
mail:!:19123:0:99999:7:::
operator:!:19123:0:99999:7:::
```

Figure 2.9: проверка возможности чтения файла

Отразите полученный результат и ваши объяснения в отчёте.

2.2 Исследование Sticky-бита

1. Выясните, установлен ли атрибут Sticky на директории /tmp, для чего выполните команду

```
ls -l / | grep tmp
```

2. От имени пользователя guest создайте файл file01.txt в директории /tmp со словом test:

```
echo "test" > /tmp/file01.txt
```

3. Просмотрите атрибуты у только что созданного файла и разрешите чтение и запись для категории пользователей «все остальные»:


```
ls -l /tmp/file01.txt
```

```
chmod o+rw /tmp/file01.txt
```

```
ls -l /tmp/file01.txt
```

4. От пользователя guest2 (не являющегося владельцем) попробуйте прочитать файл /tmp/file01.txt:

```
cat /tmp/file01.txt
```



```
[guest@Rocky ~]$ su guest2
Пароль:
[guest2@Rocky guest]$ cat /tmp/file01.txt
test
[guest2@Rocky guest]$ echo "test2" > /tmp/file01.txt
[guest2@Rocky guest]$ cat /tmp/file01.txt
test2
[guest2@Rocky guest]$
```

Figure 2.10: попытка прочитать файл

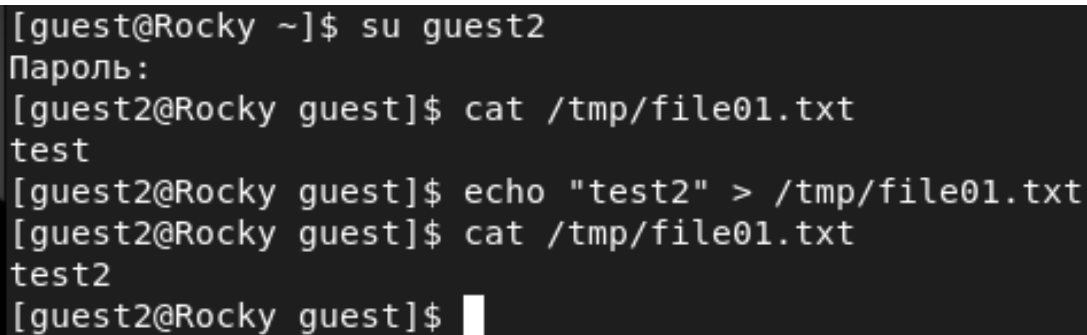
5. От пользователя guest2 попробуйте дозаписать в файл /tmp/file01.txt слово test2 командой

```
echo "test2" > /tmp/file01.txt
```

Удалось ли вам выполнить операцию?

6. Проверьте содержимое файла командой

```
cat /tmp/file01.txt
```



```
[guest@Rocky ~]$ su guest2
Пароль:
[guest2@Rocky guest]$ cat /tmp/file01.txt
test
[guest2@Rocky guest]$ echo "test2" > /tmp/file01.txt
[guest2@Rocky guest]$ cat /tmp/file01.txt
test2
[guest2@Rocky guest]$
```

Figure 2.11: проверка содержимого

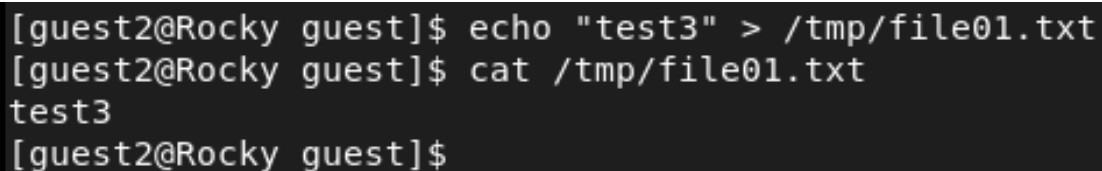
7. От пользователя guest2 попробуйте записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию командой

```
echo "test3" > /tmp/file01.txt
```

Удалось ли вам выполнить операцию?

8. Проверьте содержимое файла командой

```
cat /tmp/file01.txt
```



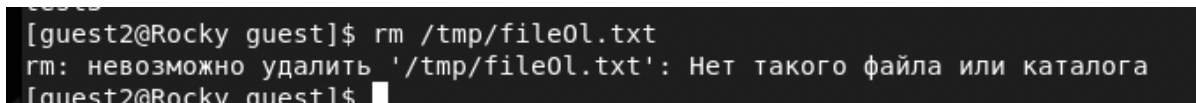
```
[guest2@Rocky guest]$ echo "test3" > /tmp/file01.txt
[guest2@Rocky guest]$ cat /tmp/file01.txt
test3
[guest2@Rocky guest]$
```

Figure 2.12: проверка содержимого

9. От пользователя `guest2` попробуйте удалить файл `/tmp/file01.txt` командой

```
rm /tmp/file01.txt
```

Удалось ли вам удалить файл?



```
[guest2@Rocky guest]$ rm /tmp/file01.txt
rm: невозможно удалить '/tmp/file01.txt': Нет такого файла или каталога
[guest2@Rocky guest]$
```

Figure 2.13: удаление файла

10. Повысьте свои права до суперпользователя следующей командой

```
su -
```

и выполните после этого команду, снимающую атрибут `t` (Sticky-бит) с директории `/tmp`:

```
chmod -t /tmp
```

11. Покиньте режим суперпользователя командой `exit`

12. От пользователя `guest2` проверьте, что атрибута `t` у директории `/tmp` нет:

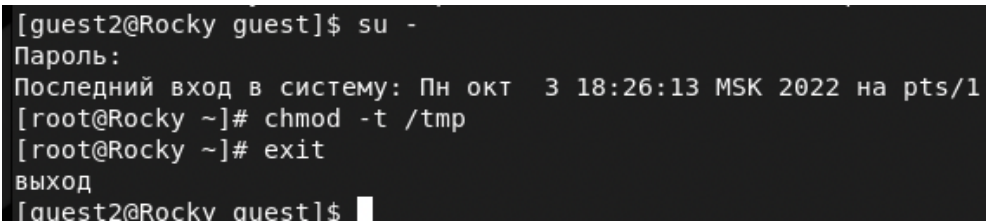
```
ls -l / | grep tmp
```

13. Повторите предыдущие шаги. Какие наблюдаются изменения?

14. Удалось ли вам удалить файл от имени пользователя, не являющегося его владельцем? Ваши наблюдения занесите в отчёт.

15. Повысьте свои права до суперпользователя и верните атрибут `t` на директорию `/tmp`:

```
su -  
chmod +t /tmp  
exit
```

A terminal window with a black background and white text. The user is in a shell as 'guest2' on a 'Rocky' machine. They enter 'su -' to become root. The prompt changes to '[root@Rocky ~]#'. They enter 'chmod -t /tmp' to remove the sticky bit. The prompt changes back to '[root@Rocky ~]#'. They enter 'exit' to return to the guest user. The prompt changes back to '[guest2@Rocky guest]\$'. There is a small white cursor at the end of the last line.

```
[guest2@Rocky guest]$ su -  
Пароль:  
Последний вход в систему: Пн окт  3 18:26:13 MSK 2022 на pts/1  
[root@Rocky ~]# chmod -t /tmp  
[root@Rocky ~]# exit  
выход  
[guest2@Rocky guest]$
```

Figure 2.14: возвращение атрибута

3 Выводы

Изучены механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получены практические навыки работы в консоли с дополнительными атрибутами. Рассмотрены работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

4 Библиография

1. Методические материалы курса