

# Лабораторная работа № 5

Дискреционное разграничение прав в Linux. Исследование влияния  
дополнительных атрибутов

Абакумов Егор Александрович

# Содержание

Цель работы	5
Теоретическое описание	6
Ход работы	8
Выводы	17
Список литературы	18

# List of Figures

0.1	Листинг simpleid.c . . . . .	8
0.2	Запуск и сверка simpled.c и id . . . . .	9
0.3	Листинг simpleid2.c . . . . .	9
0.4	Запуск simpleid2.c . . . . .	9
0.5	Команды под рутом . . . . .	10
0.6	Повторный запуск simpleid2.c . . . . .	10
0.7	Команды под рутом для SetGID . . . . .	10
0.8	Третий запуск simpleid2.c . . . . .	11
0.9	readfile.c (ч.1) . . . . .	11
0.10	readfile.c (ч.2) . . . . .	12
0.11	Смена владельца readfile.c . . . . .	12
0.12	Проверка на cat из-под guest'a . . . . .	13
0.13	Смена владельца readfile и SetUID . . . . .	13
0.14	Проверка readfile.c программой readfile . . . . .	14
0.15	Проверка /etc/shadow программой readfile . . . . .	14
0.16	Создание файла и правка прав . . . . .	15
0.17	Тестирование файла . . . . .	15
0.18	Удаление Sticky-бита . . . . .	16
0.19	Повторное тестирование файла . . . . .	16

# List of Tables

## Цель работы

Изучение механизмов изменения идентификаторов, применения SetUID-, SetGID- 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.

# Теоретическое описание

В операционной системе Linux есть много отличных функций безопасности, но она из самых важных - это система прав доступа к файлам. Linux, как последователь идеологии ядра Linux в отличие от Windows, изначально проектировался как многопользовательская система, поэтому права доступа к файлам в linux продуманы очень хорошо.

У каждого файла имеется определённый набор свойств в файловой системе. Например, это права доступа, владелец, имя, метки времени. В Linux каждый файл имеет довольно много свойств, например, права доступа устанавливаются трижды (для владельца, группы и всех прочих), метки времени также бывают трёх разных видов (время создание, доступа и изменения) [1].

Кроме того файлам и директориям могут быть установлены расширенные атрибуты доступа. Файловые атрибуты могут использовать администраторы и пользователи для защиты файлов от случайных удалений и изменений, а также их применяют злоумышленники, делая невозможным удаление вредоносного файла.

Для работы с этими атрибутами обычно используются утилиты `chattr` и `lsattr`, входящие в пакет `e2fsprogs` и предустановленные во всех современных дистрибутивах.

Для работы с правами используем команду `chmod`, синтаксис которой выглядит следующим образом:

```
chmod options permissions path_to_the_file
```

Сначала рассмотрим какими бывают права доступа linux и как они устанавливаются. Есть три основных вида прав:

r - чтение; w - запись; x - выполнение; s - выполнение от имени суперпользователя

(дополнительный);

Также есть три категории пользователей, для которых вы можете установить эти права на файл linux:

u - владелец файла; g - группа файла; o - все остальные пользователи; Синтаксис настройки прав такой:

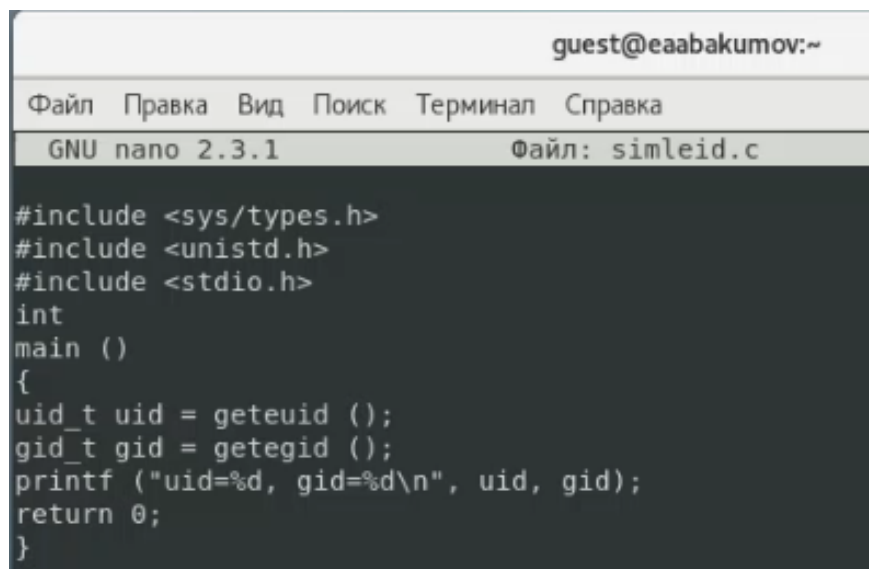
группа\_пользователей действие вид\_прав

В качестве действий могут использоваться знаки “+” - включить или “-” - отключить [2].

Тем не менее, не стоит забывать, что вы не можете использовать `chattr` как меру безопасности так как атрибуты легко изменить. Один из способов решения этой проблемы - ограничить доступ к самой утилите `chattr`[3].

## Ход работы

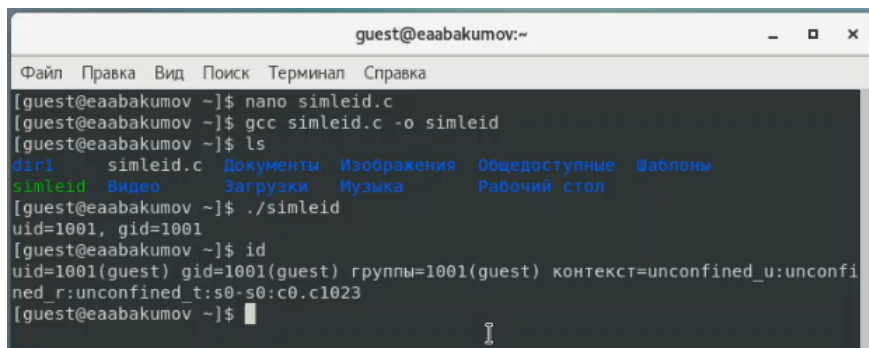
1. Готовим систему и входим из-под guest'а. Пишем программу simpleid.c (иллюстр. 0.1). Компилируем программу, запускаем, видим вывод uid и gid пользователя, сравниваем вывод с id, всё так же (иллюстр. 0.2).



```
guest@eaabakumov:~  
Файл Правка Вид Поиск Терминал Справка  
GNU nano 2.3.1 Файл: 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;  
}
```

Figure 0.1: Листинг simpleid.c

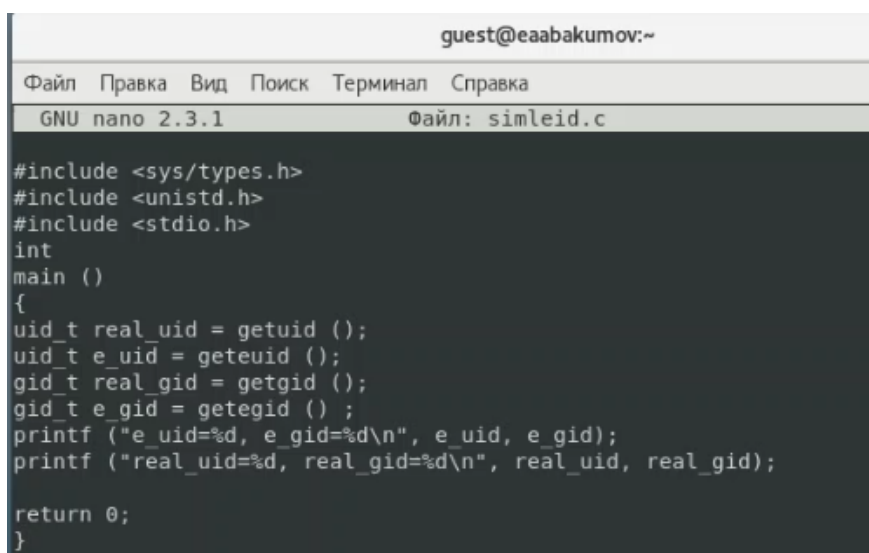




```
guest@eaabakumov:~  
Файл Правка Вид Поиск Терминал Справка  
[guest@eaabakumov ~]$ nano simpleid.c  
[guest@eaabakumov ~]$ gcc simpleid.c -o simpleid  
[guest@eaabakumov ~]$ ls  
dir1 simpleid.c Документы Изображения Общедоступные Шаблоны  
simpleid Видео Загрузки Музыка Рабочий стол  
[guest@eaabakumov ~]$ ./simpleid  
uid=1001, gid=1001  
[guest@eaabakumov ~]$ id  
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi  
ned_r:unconfined_t:s0-s0:c0.c1023  
[guest@eaabakumov ~]$
```

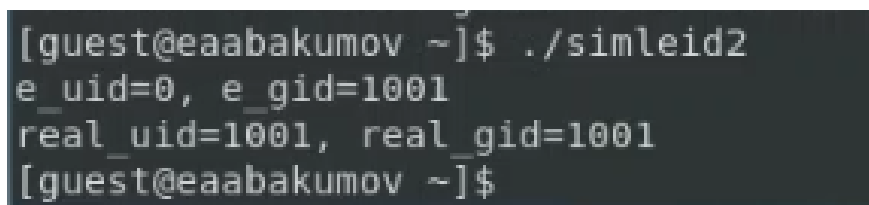
Figure 0.2: Запуск и сверка simpleid.c и id

2. Усложняем программу и запускаем её (иллюстр. 0.3, 0.4). Из-под рута меняем владельца и добавляем SetUID бит на файл (иллюстр. 0.5). Проверяем правильность и запускаем программу еще раз. `euid` возвращает `id` владельца, а `real_uid` возвращает `uid` запускающего пользователя (иллюстр. 0.6).



```
guest@eaabakumov:~  
Файл Правка Вид Поиск Терминал Справка  
GNU nano 2.3.1 Файл: simpleid.c  
#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;  
}
```

Figure 0.3: Листинг simpleid2.c



```
[guest@eaabakumov ~]$ ./simpleid2  
e_uid=0, e_gid=1001  
real_uid=1001, real_gid=1001  
[guest@eaabakumov ~]$
```

Figure 0.4: Запуск simpleid2.c

```
root@eaabakumov:~  
Файл Правка Вид Поиск Терминал Справка  
[guest@eaabakumov ~]$ su -  
Пароль:  
Последний вход в систему: Чт ноя 11 17:27:55 MSK 2021 на pts/0  
[root@eaabakumov ~]# chown root:guest /home/guest/simpleid2  
[root@eaabakumov ~]# chmod u+s /home/guest/simpleid2  
[root@eaabakumov ~]#
```

Figure 0.5: Команды под рутом

```
[guest@eaabakumov ~]$ ls -l  
итого 16  
drwxrwxr-x. 2 guest guest 59 ноя 11 17:26 dir1  
-rwsrwxr-x. 1 root guest 8576 ноя 11 17:48 simpleid2  
-rw-rw-r--. 1 guest guest 304 ноя 11 17:47 simpleid2.c  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Видео  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Документы  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Загрузки  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Изображения  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Музыка  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Общедоступные  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Рабочий стол  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Шаблоны  
[guest@eaabakumov ~]$ ls -l simpleid2  
-rwsrwxr-x. 1 root guest 8576 ноя 11 17:48 simpleid2  
[guest@eaabakumov ~]$ ./simpleid2  
e_uid=0, e_gid=1001  
real_uid=1001, real_gid=1001  
[guest@eaabakumov ~]$ id  
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi  
ned_r:unconfined_t:s0-s0:c0.c1023  
[guest@eaabakumov ~]$
```

Figure 0.6: Повторный запуск simpleid2.c

3. Теперь добавим на файл SetGID бит с проделаем все то же самое (иллюстр. 0.7, 0.8).

```
[root@eaabakumov ~]# id  
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfi  
ned_t:s0-s0:c0.c1023  
[root@eaabakumov ~]# chmod g+s /home/guest/simpleid2
```

Figure 0.7: Команды под рутом для SetGID

```
[guest@eaabakumov ~]$ ls -l simleid2
-rwsrwxr-x. 1 root guest 8576 ноя 11 17:48 simleid2
[guest@eaabakumov ~]$ ls -l simleid2
-rwsrwxr-x. 1 root guest 8576 ноя 11 17:48 simleid2
[guest@eaabakumov ~]$ ./simleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@eaabakumov ~]$
```

Figure 0.8: Третий запуск simpleid2.c

4. Пишем программу readfile.c (иллюстр. 0.9, 0.10).

```

guest@eaabakumov:~
Файл  Правка  Вид  Поиск  Терминал  Справка
GNU nano 2.3.1      Файл: 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);

```

Figure 0.9: readfile.c (ч.1)

```
guest@eaabakumov:~  
Файл Правка Вид Поиск Терминал Справка  
GNU nano 2.3.1 Файл: readfile.c  
  
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;  
}
```

Figure 0.10: readfile.c (ч.2)

5. Меняем владельца у файла readfile.c и запрещаем чтение всем, кроме супер-пользователя (иллюстр. 0.11). Проверяем, что guest не может читать (иллюстр. 0.12). Меняем владельца у программы readfile и добавляем SetUID бит на неё (иллюстр. 0.13).

```
root@eaabakumov:/home/guest  
Файл Правка Вид Поиск Терминал Справка  
[root@eaabakumov guest]# chown root:guest /home/guest/readfile.c  
[root@eaabakumov guest]# ls -l  
итого 32  
drwxrwxr-x. 2 guest guest 59 ноя 11 17:26 dir1  
-rwxrwxr-x. 1 guest guest 8512 ноя 11 17:59 readfile  
----- 1 root guest 402 ноя 11 17:59 readfile.c  
-rwsrwsr-x. 1 root guest 8576 ноя 11 17:48 simleid2  
-rw-rw-r--. 1 guest guest 304 ноя 11 17:47 simleid2.c  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Видео  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Документы  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Загрузки  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Изображения  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Музыка  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Общедоступные  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Рабочий стол  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Шаблоны  
[root@eaabakumov guest]# chmod 700 readfile.c  
[root@eaabakumov guest]#
```

Figure 0.11: Смена владельца readfile.c

```
[guest@eaabakumov ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@eaabakumov ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@eaabakumov ~]$ █
```

Figure 0.12: Проверка на cat из-под guest'a

```
[root@eaabakumov guest]# chown root:guest readfile
[root@eaabakumov guest]# ls -l
итого 32
drwxrwxr-x. 2 guest guest 59 ноя 11 17:26 dir1
-rwxrwxr-x. 1 root guest 8512 ноя 11 17:59 readfile
-rwx----- 1 root guest 402 ноя 11 17:59 readfile.c
-rwsrwsr-x. 1 root guest 8576 ноя 11 17:48 simleid2
-rw-rw-r-- 1 guest guest 304 ноя 11 17:47 simleid2.c
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Видео
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Документы
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Загрузки
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Изображения
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Музыка
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Общедоступные
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Рабочий стол
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Шаблоны
[root@eaabakumov guest]# chmod u+s readfile
[root@eaabakumov guest]# █
```

Figure 0.13: Смена владельца readfile и SetUID

6. Проверяем, может ли программа readfile прочитать файл readfile.c и файл /etc/shadow. Да, может (иллюстр. 0.14, 0.15). Хотя сам пользователь вручную не мог. Всё дело в том, что при вызове программы права пользователя повышаются SetUID битом до прав владельца, который может читать файлы (суперпользователь в нашем случае).

```
guest@eaabakumov:~  
Файл Правка Вид Поиск Терминал Справка  
drwxr-xr-x. 2 guest guest 6 сен 28 20:17 Шаблоны  
[guest@eaabakumov ~]$ ./readfile 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;  
}
```

Figure 0.14: Проверка readfile.c программой readfile

```
[guest@eaabakumov ~]$ ./readfile /etc/shadow  
root:$6$2xVCv/JiCEGpdjJJ$GJ68BRa7p0fKFrlrMDpa8A3qIBe2qr05qDDYDxoMh9wbn0zbIRvc2cH  
QMeP07bldhjzgVYLVQXXZmVWy4fDrD1::0:99999:7:::  
bin:!:18353:0:99999:7:::  
daemon:!:18353:0:99999:7:::  
adm:!:18353:0:99999:7:::  
lp:!:18353:0:99999:7:::  
sync:!:18353:0:99999:7:::  
shutdown:!:18353:0:99999:7:::  
halt:!:18353:0:99999:7:::  
mail:!:18353:0:99999:7:::  
operator:!:18353:0:99999:7:::  
games:!:18353:0:99999:7:::  
ftp:!:18353:0:99999:7:::  
nobody:!:18353:0:99999:7:::  
systemd-network:!!:18884:::~
```

Figure 0.15: Проверка /etc/shadow программой readfile

7. Проверяем Sticky бит. Для этого создаем файл, которому даем rw права для others и пишем туда слово test (иллюстр. 0.16). Теперь пробуем выполнить дозапись в файл, перезапись файла и его удаление. Всё, кроме удаления, прошло успешно (иллюстр. 0.17).

```
guest@eaabakumov:~  
Файл Правка Вид Поиск Терминал Справка  
[guest@eaabakumov ~]$ ls -l / | grep tmp  
drwxrwxrwt. 16 root root 4096 ноя 11 18:08 tmp  
[guest@eaabakumov ~]$ echo "test" > /tmp/file01.txt  
[guest@eaabakumov ~]$ ls -l /tmp/file01.txt  
-rw-rw-r--. 1 guest guest 5 ноя 11 18:10 /tmp/file01.txt  
[guest@eaabakumov ~]$ chmod o+rw /tmp/file01.txt  
[guest@eaabakumov ~]$ ls -l /tmp/file01.txt  
-rw-rw-rw-. 1 guest guest 5 ноя 11 18:10 /tmp/file01.txt  
[guest@eaabakumov ~]$
```

Figure 0.16: Создание файла и правка прав

```
guest2@eaabakumov:/home/guest  
Файл Правка Вид Поиск Терминал Справка  
[guest2@eaabakumov guest]$ cat /tmp/file01.txt  
test  
[guest2@eaabakumov guest]$ echo "test2" > /tmp/file01.txt  
[guest2@eaabakumov guest]$ cat /tmp/file01.txt  
test2  
[guest2@eaabakumov guest]$ echo "test" > /tmp/file01.txt  
[guest2@eaabakumov guest]$ cat /tmp/file01.txt  
test  
[guest2@eaabakumov guest]$ echo "test2" >> /tmp/file01.txt  
[guest2@eaabakumov guest]$ echo "test3" > /tmp/file01.txt  
[guest2@eaabakumov guest]$ rm /tmp/file01.txt  
rm: невозможно удалить «/tmp/file01.txt»: Нет такого файла или каталога  
[guest2@eaabakumov guest]$ rm /tmp/file01.txt  
rm: невозможно удалить «/tmp/file01.txt»: Нет такого файла или каталога  
[guest2@eaabakumov guest]$ rm /tmp/file01.txt  
rm: невозможно удалить «/tmp/file01.txt»: Операция не позволена  
[guest2@eaabakumov guest]$  
  
-rw-rw-rw-. 1 guest guest 5 ноя 11 18:10 /tmp/file01.txt  
[guest@eaabakumov ~]$ ls -l /tmp/file01.txt  
-rw-rw-rw-. 1 guest guest 5 ноя 11 18:12 /tmp/file01.txt  
[guest@eaabakumov ~]$ cat /tmp/file01.txt  
test  
test2  
[guest@eaabakumov ~]$ cat /tmp/file01.txt  
test3  
[guest@eaabakumov ~]$
```

Figure 0.17: Тестирование файла

8. Повышаем права до суперпользователя и удаляем Sticky-бит с папки /tmp (иллюстр. 0.18). Повторяем наши тесты. Теперь прошли все команды, включая удаление файла (иллюстр. 0.19). Таким образом, пользователь, не являющийся владельцем файла, смог его удалить, так как Sticky-бит не был настроен. Возвращаем Sticky-бит на папку /tmp.



```
root@eaabakumov:~  
Файл Правка Вид Поиск Терминал Справка  
Последний вход в систему: Чт ноя 11 17:49:25 MSK 2021 на pts/1  
[root@eaabakumov ~]# chmod -t /tmp/  
[root@eaabakumov ~]# ls -l /  
итого 20  
lrwxrwxrwx. 1 root root 7 сен 14 14:12 bin -> usr/bin  
dr-xr-xr-x. 5 root root 4096 сен 14 14:58 boot  
drwxr-xr-x. 20 root root 3200 ноя 11 17:24 dev  
drwxr-xr-x. 140 root root 8192 ноя 11 17:24 etc  
drwxr-xr-x. 5 root root 51 окт 11 17:53 home  
lrwxrwxrwx. 1 root root 7 сен 14 14:12 lib -> usr/lib  
lrwxrwxrwx. 1 root root 9 сен 14 14:12 lib64 -> usr/lib64  
drwxr-xr-x. 2 root root 6 апр 11 2018 media  
drwxr-xr-x. 2 root root 6 апр 11 2018 mnt  
drwxr-xr-x. 4 root root 49 сен 14 14:36 opt  
dr-xr-xr-x. 230 root root 0 ноя 11 17:24 proc  
dr-xr-xr-x. 5 root root 245 ноя 11 18:14 root  
drwxr-xr-x. 42 root root 1320 ноя 11 17:28 run  
lrwxrwxrwx. 1 root root 8 сен 14 14:12 sbin -> usr/sbin  
drwxr-xr-x. 2 root root 6 апр 11 2018 srv  
dr-xr-xr-x. 13 root root 0 ноя 11 17:24 sys  
drwxrwxrwx. 17 root root 4096 ноя 11 18:14 tmp  
drwxr-xr-x. 13 root root 155 сен 14 14:12 usr  
drwxr-xr-x. 20 root root 282 сен 14 14:33 var  
[root@eaabakumov ~]#
```

Figure 0.18: Удаление Sticky-бита

```
guest2@eaabakumov:/home/guest  
Файл Правка Вид Поиск Терминал Справка  
[guest2@eaabakumov guest]$ cat /tmp/file01.txt  
test  
[guest2@eaabakumov guest]$ echo "test2" >> /tmp/file01.txt  
[guest2@eaabakumov guest]$ cat /tmp/file01.txt  
test  
test2  
[guest2@eaabakumov guest]$ echo "test3" > /tmp/file01.txt  
[guest2@eaabakumov guest]$ cat /tmp/file01.txt  
test3  
[guest2@eaabakumov guest]$ rm /tmp/file01.txt  
[guest2@eaabakumov guest]$ ls /tmp/  
ssh-m0b9nzuCRAYV  
systemd-private-5c6f325816eb405485c6720e7ddc4633-bolt.service-FXclJ5  
systemd-private-5c6f325816eb405485c6720e7ddc4633-chronyd.service-nuhM61  
systemd-private-5c6f325816eb405485c6720e7ddc4633-colord.service-DQo8IS  
systemd-private-5c6f325816eb405485c6720e7ddc4633-cups.service-LXEI2C  
systemd-private-5c6f325816eb405485c6720e7ddc4633-fwupd.service-r6sEGj  
systemd-private-5c6f325816eb405485c6720e7ddc4633-rtkit-daemon.service-5YYDq8  
tracker-extract-files.1001  
yum_save_tx.2021-11-11.17-26.bmpag5.yumtx  
[guest2@eaabakumov guest]$
```

Figure 0.19: Повторное тестирование файла



## Выводы

В ходе работы мы успешно изучили механизмы изменения идентификаторов, применения SetUID-, SetGID- и Sticky-битов, получили практические навыки работы в консоли с дополнительными атрибутами, рассмотрели принципы работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

## Список литературы

1. Атрибуты файлов в Linux. // zalinux.ru. 2021. URL: <https://zalinux.ru/?p=6440> (дата обращения 11.11.2021).
2. Команда chmod в Linux. // Losst. 2020. URL: <https://losst.ru/komanda-chmod-linux> (дата обращения 11.11.2021).
3. КОМАНДА CHATTR В LINUX. // Losst. 2020. URL: <https://losst.ru/neizmenyaemye-fajly-v-linux> (дата обращения 11.11.2021).
4. Д. С. Кулябов, А. В. Королькова, М. Н. Геворкян. Информационная безопасность компьютерных сетей: лабораторные работы. // Факультет физико-математических и естественных наук. М.: РУДН, 2015. 64 с..