

Лабораторная работа №5

**Дискреционное разграничение прав в Linux. Исследование влияния
дополнительных атрибутов**

Парфенова Елизавета Евгеньевна

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	8
3.1	Подготовка лабораторного стенда	8
3.2	Создание программ и работа с SetUID- и SetGID-битами	8
3.3	Исследование Sticky-бита	14
4	Выводы	18
	Список литературы	19

Список иллюстраций

3.1	Подготовка лабораторного стенда	8
3.2	Создание файла simpleid.c	9
3.3	Программа simpleid.c	9
3.4	Вывод программы simpleid и команды id	9
3.5	Программа simpleid2.c	10
3.6	Вывод программы simpleid2	10
3.7	Смена владельца файла simpleid2 и установка SetUID-бит. Сравнение выводов программы ./simpleid2 и id	11
3.8	Манипуляции с установленным SetGID-битом	11
3.9	Программа readfile.c	12
3.10	Компиляция и чтение файла readfile.c	12
3.11	Смена владельца и прав файла readfile.c	13
3.12	Попытка чтения файла readfile.c от имени guest	13
3.13	Смена владельца у программы readfile и установка SetUID-бита	14
3.14	Чтение файла readfile.c программой readfile	14
3.15	Чтение файла /etc/shadow программой readfile	14
3.16	Права файла file01.txt	15
3.17	Манипуляции с файлом file01.txt	16
3.18	Манипуляции с файлом file01.txt без Sticky-бит на директории	16
3.19	Возвращение атрибута t на директорию tmp	17

Список таблиц

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 Теоретическое введение

Права доступа в операционной системе Linux представляют собой ключевой элемент безопасности, определяющий, какой доступ имеют пользователи и программы к файлам и каталогам [1].

Есть 3 вида разрешений. Они определяют права пользователя на 3 действия: чтение, запись и выполнение. В Linux эти действия обозначаются вот так:

- r — read (чтение) — право просматривать содержимое файла;
- w — write (запись) — право изменять содержимое файла;
- x — execute (выполнение) — право запускать файл, если это программа или скрипт.

У каждого файла есть 3 группы пользователей, для которых можно устанавливать права доступа.

- owner (владелец) — отдельный человек, который владеет файлом. Обычно это тот, кто создал файл, но владельцем можно сделать и кого-то другого.
- group (группа) — пользователи с общими заданными правами.
- others (другие) — все остальные пользователи, не относящиеся к группе и не являющиеся владельцами [2].

Но, кроме прав чтения, выполнения и записи, есть еще три дополнительных атрибута.

1. SetUID — это бит разрешения, который позволяет пользователю запускать исполняемый файл с правами владельца этого файла. Другими словами,

использование этого бита позволяет нам поднять привилегии пользователя в случае, если это необходимо. Например, права “-rwsr-xr-x”: на месте, где обычно установлен классический бит x (на исполнение), у нас выставлен специальный бит s. Командна, с помощью которой устанавливается этот доп.атрибут: *chmod u+s 'filename'*

2. SetGID - это бит разрешения, который позволяет пользователю запускать исполняемый файл от имени группы, которая владеет файлом. Например, права “-rwxr-sr-x”: на месте, где обычно установлен классический бит x (на исполнение группой), у нас выставлен специальный бит s. Командна, с помощью которой устанавливается этот доп.атрибут: *chmod g+s 'filename'*
3. Sticky Bit - специальный бит разрешения, который позволяет только владельцу удалять файлы в папке, на которой этот бит установлен. Пример использования этого бита в операционной системе это системная папка /tmp . Эта папка разрешена на запись любому пользователю, но удалять файлы в ней могут только пользователи, являющиеся владельцами этих файлов. [3]

3 Выполнение лабораторной работы

3.1 Подготовка лабораторного стенда

Начнем с подготовки лабораторного стенда. Проверим установку gcc в нашей ОС, а так же отключим систему запретов SELinux командой *setenforce 0*. После этого проверим, чтобы команда *getenforce* выводила Permissive. Это свидетельствует о том, что все получилось парвильно. (рис. 3.1).

```
[guest@eeparfenova ~]$ su - eeparfenova
Password:
[eeparfenova@eeparfenova ~]$ yum install gcc
Error: This command has to be run with superuser privileges (under the root user on most systems).
[eeparfenova@eeparfenova ~]$ su -
Password:
[root@eeparfenova ~]# yum install gcc
Rocky Linux 9 - BaseOS                2.5 MB/s | 2.3 MB   00:00
Rocky Linux 9 - AppStream              4.7 MB/s | 8.0 MB   00:01
Rocky Linux 9 - Extras                 28 kB/s | 15 kB    00:00
Package gcc-11.4.1-3.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@eeparfenova ~]# setenforce 0
[root@eeparfenova ~]# getenforce
Permissive
[root@eeparfenova ~]#
```

Рис. 3.1: Подготовка лабораторного стенда

3.2 Создание программ и работа с SetUID- и SetGID-битами

Далее войдем в систему от пользователя guest и создадим файл simpleid.c командой *touch*. (рис. 3.2)


```

[root@eeparfenova ~]# exit
logout
[eeparfenova@eeparfenova ~]$ su - guest
Password:
[guest@eeparfenova ~]$ cd Documents/
[guest@eeparfenova Documents]$ touch simpleid.c
[guest@eeparfenova Documents]$

```

Рис. 3.2: Создание файла simpleid.c

Запишем в файл программу, представленную в файле лабораторной работы, которая выводит на экран uid и gid пользователя (рис. 3.3)

```

1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4 int
5 main ()
6 {
7     uid_t uid = getuid ();
8     gid_t gid = getgid ();
9     printf ("uid=%d, gid=%d\n", uid, gid);
10    return 0;
11 }

```

Рис. 3.3: Программа simpleid.c

Далее скомпилируем программу командой *gcc simpleid.c -o simpleid* (проверяем, что программа появилась в каталоге) и выполним ее с помощью *./simpleid*. Видим, что выводятся uid и gid. Выполним команду *id* и сравним результаты. (рис. 3.4) Видим, что вывод обеих команд совпадает (uid и gid равны 1001)

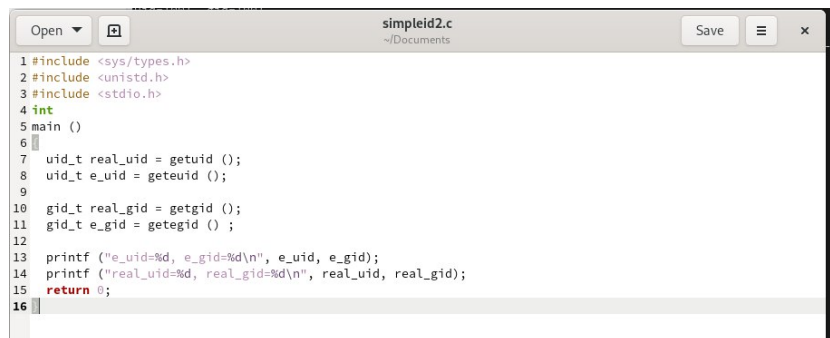
```

[guest@eeparfenova Documents]$ gcc simpleid.c -o simpleid
[guest@eeparfenova Documents]$ ./simpleid
uid=1001, gid=1001
[guest@eeparfenova Documents]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@eeparfenova Documents]$

```

Рис. 3.4: Вывод программы simpleid и команды id

Создадим файл simpleid2.c и запишем в него усложненную программу, в которой добавлен вывод действительных идентификаторов. (рис. 3.5)



```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4 int
5 main ()
6 {
7     uid_t real_uid = getuid ();
8     uid_t e_uid = geteuid ();
9
10    gid_t real_gid = getgid ();
11    gid_t e_gid = getegid ();
12
13    printf ("e_uid=%d, e_gid=%d\n", e_uid, e_gid);
14    printf ("real_uid=%d, real_gid=%d\n", real_uid, real_gid);
15    return 0;
16 }
```

Рис. 3.5: Программа simpleid2.c

Скомпилируем и запустим simpleid2.c командами `gcc simpleid2.c -o simpleid2` и `./simpleid2` соответственно. (рис. 3.6). Видим, что вывод этой программы, предыдущей и команды `id` полностью совпадают.



```
[guest@eeparfenova Documents]$ gcc simpleid2.c -o simpleid2
[guest@eeparfenova Documents]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@eeparfenova Documents]$
```

Рис. 3.6: Вывод программы simpleid2

От имени суперпользователя (временно повысив свои права с помощью `su`) выполним команды `chown root:guest /home/guest/simpleid2` и `chmod u+s /home/guest/simpleid2`, чтобы сменить владельца файла simpleid2 и установить дополнительный атрибут SetUID-бит соответственно. Выполним проверку правильности установки новых атрибутов и смены владельца файла simpleid2 командой `ls -l simpleid2`. Видим, что на месте x в правах владельца в необходимом месте появилас s. Запустим simpleid2 и id командами `./simpleid2` и `id`. Видим, что результаты вывода программы и команды одинаковы для суперпользователя (все параметры равны 0)(рис. 3.7)

```
[guest@eeparfenova Documents]$ su -
Password:
[root@eeparfenova ~]# chown root:guest /home/guest/Documents/simpleid2
[root@eeparfenova ~]# chmod u+s /home/guest/Documents/simpleid2
[root@eeparfenova ~]# ls -l simpleid2
ls: cannot access 'simpleid2': No such file or directory
[root@eeparfenova ~]# cd Doc
-bash: cd: Doc: No such file or directory
[root@eeparfenova ~]# ls -l /home/guest/Documents/simpleid2
-rwsr-xr-x. 1 root guest 24488 Oct 1 13:16 /home/guest/Documents/simpleid2
[root@eeparfenova ~]# ./simpleid2
-bash: ./simpleid2: No such file or directory
[root@eeparfenova ~]# ./home/guest/Documents/simpleid2
-bash: ./home/guest/Documents/simpleid2: No such file or directory
[root@eeparfenova ~]# /home/guest/Documents/simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@eeparfenova ~]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@eeparfenova ~]#
```

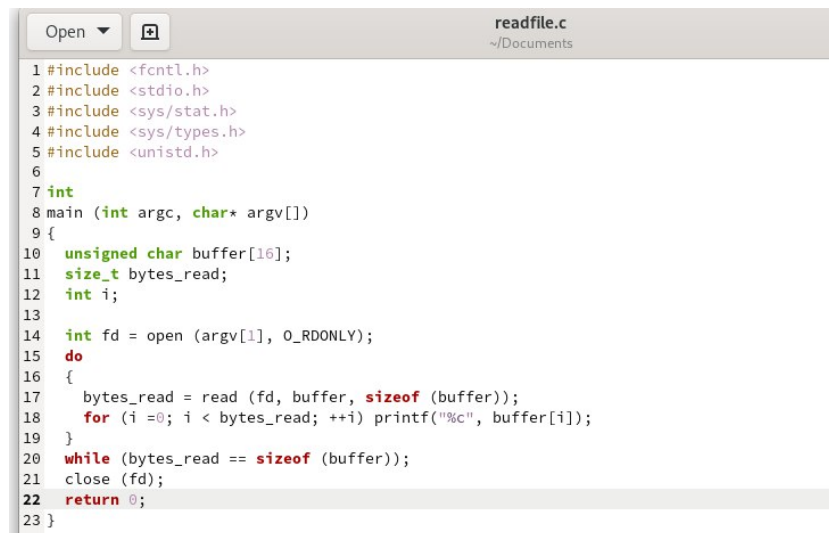
Рис. 3.7: Смена владельца файла simpleid2 и установка SetUID-бит. Сравнение выводов программы ./simpleid2 и id

Прделаем тоже самое относительно SetGID-бита. Для этого командой *chmod u-s /home/guest/simpleid2* снимем предыдущий дополнительный атрибут и установим SetGID-бита с помощью *chmod g+s /home/guest/simpleid2*. Все манипуляции выполняем от имени суперпользователя. Затем выйдем из этого режима, и от имени пользователя guest запустим программу simpleid2.c и команду *id*. Все проходит успешно, и при сравнении результатов снова видим, что выходы одинаковы (рис. 3.8)

```
[root@eeparfenova ~]# chmod u-s /home/guest/Documents/simpleid2
[root@eeparfenova ~]# chmod g+s /home/guest/Documents/simpleid2
[root@eeparfenova ~]# exit
logout
[guest@eeparfenova Documents]$ ls -l simpleid2
-rwxr-sr-x. 1 root guest 24488 Oct 1 13:16 simpleid2
[guest@eeparfenova Documents]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@eeparfenova Documents]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@eeparfenova Documents]$
```

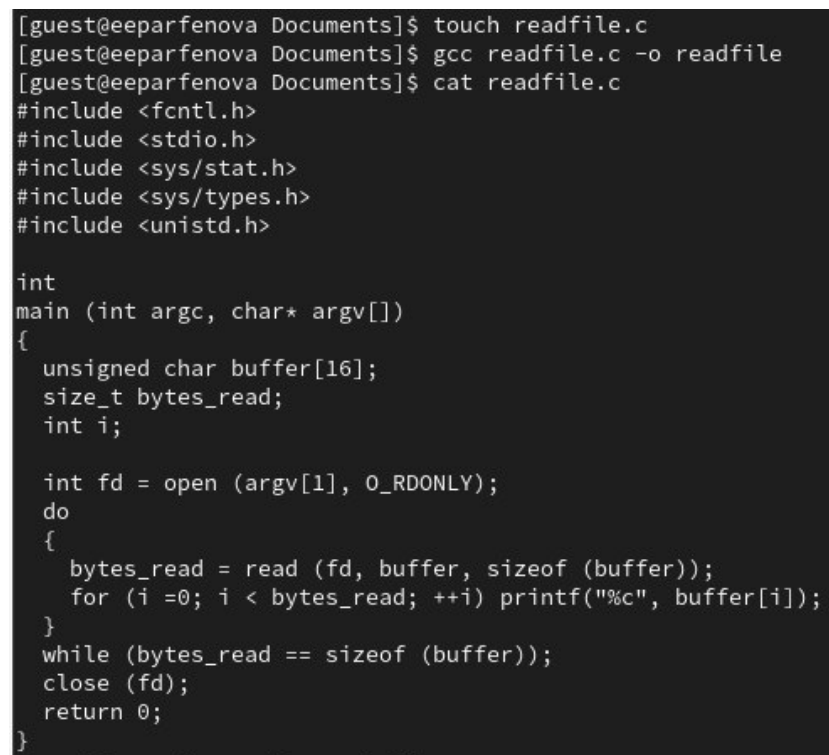
Рис. 3.8: Манипуляции с установленным SetGID-битом

Создадим файл *readfile.c*, запишем в него программу для чтения файлов(рис. 3.9) и откомпилируем ее командой *gcc readfile.c -o readfile* (рис. 3.10) и попробуем прочитать командой *cat readfile.c*. Видим, что все проходит успешно.



```
1 #include <fcntl.h>
2 #include <stdio.h>
3 #include <sys/stat.h>
4 #include <sys/types.h>
5 #include <unistd.h>
6
7 int
8 main (int argc, char* argv[])
9 {
10     unsigned char buffer[16];
11     size_t bytes_read;
12     int i;
13
14     int fd = open (argv[1], O_RDONLY);
15     do
16     {
17         bytes_read = read (fd, buffer, sizeof (buffer));
18         for (i = 0; i < bytes_read; ++i) printf("%c", buffer[i]);
19     }
20     while (bytes_read == sizeof (buffer));
21     close (fd);
22     return 0;
23 }
```

Рис. 3.9: Программа readfile.c



```
[guest@eeparfenova Documents]$ touch readfile.c
[guest@eeparfenova Documents]$ gcc readfile.c -o readfile
[guest@eeparfenova Documents]$ cat 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;
}
```

Рис. 3.10: Компиляция и чтение файла readfile.c

Сменим владельца у файла readfile.c командой *chown root:guest /home/guest/readfile.c* (предварительно повысив свои права до суперпользователя) и изменим права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог. Сде-

лаем это с помощью команды `chmod 700 /home/guest/readfile.c`. Суперпользователь может прочесть файл (команда `cat`) (рис. 3.11)

```
[guest@eeparfenova Documents]$ su -
Password:
[root@eeparfenova ~]# chmod u+s /home/guest/Documents/readfile.c
[root@eeparfenova ~]# chmod u-s /home/guest/Documents/readfile.c
[root@eeparfenova ~]# chown root:guest /home/guest/Documents/readfile.c
[root@eeparfenova ~]# chmod 700 /home/guest/Documents/readfile.c
[root@eeparfenova ~]# cat readfile.c
cat: readfile.c: No such file or directory
[root@eeparfenova ~]# cat /home/guest/Documents/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;
}
```

Рис. 3.11: Смена владельца и прав файла `readfile.c`

Проверим, что пользователь `guest` не может прочитать файл `readfile.c`. Для этого выйдем из режима суперпользователя и примерим команду `cat readfile.c` от имени пользователя `guest`. В доступе нам отказано. (рис. 3.12)

```
[root@eeparfenova ~]# exit
logout
[guest@eeparfenova Documents]$ cat readfile.c
cat: readfile.c: Permission denied
[guest@eeparfenova Documents]$
```

Рис. 3.12: Попытка чтения файла `readfile.c` от имени `guest`

Сменим у программы `readfile` владельца и установите SetUID-бит. Для этого в режиме суперпользователя применим команды `chown root:guest /home/guest/readfile`

и `chmod u+s /home/guest/readfile` соответственно для каждой задачи. Проверим, что все прошло успешно командой `ls -l`. (рис. 3.13)

```
[guest@eeparfenova Documents]$ su -
Password:
[root@eeparfenova ~]# chown root:guest /home/guest/Documents/readfile
[root@eeparfenova ~]# chmod u+s /home/guest/Documents/readfile
[root@eeparfenova ~]# ls -l /home/guest/Documents/readfile
ls: cannot access '-': No such file or directory
ls: cannot access 'l': No such file or directory
/home/guest/Documents/readfile
[root@eeparfenova ~]# ls -l /home/guest/Documents/readfile
-rwsr-xr-x. 1 root guest 24432 Oct  1 13:27 /home/guest/Documents/readfile
[root@eeparfenova ~]#
```

Рис. 3.13: Смена владельца у программы `readfile` и установка SetUID-бита

Проверим, может ли программа `readfile` прочитать файл `readfile.c` командой `./readfile readfile.c`, предварительно выйдя из режим суперпользователя. Видим, что все получается. (рис. 3.14)

```
bash: ./readfile: No such file or directory
[root@eeparfenova ~]# exit
logout
[guest@eeparfenova Documents]$ ./readfile readfile.c
#include <fcntl.h>
```

Рис. 3.14: Чтение файла `readfile.c` программой `readfile`

Теперь проверим, может ли программа `readfile` прочитать файл `/etc/shadow`, используя `./readfile /etc/shadow`. (рис. 3.15) Видим, что файл также успешно читается, несмотря на то, что `guest` не является его владельцем. Это происходит потому, что программа `readfile` теперь имеет все права пользователя `root`.

```
[guest@eeparfenova Documents]$ ./readfile /etc/shadow
root:$6$n7606yPrBNTbrvC4$t08gv8uTEXdq6sPcf/xyGwMRMzdKNE4zRAQqIlsoZW5hC9Idst6LkejF6Zo/4yJ/GEU50heg94ASnU1V0jg0::
0:99999:7:::
bin:!:19820:0:99999:7:::
daemon:!:19820:0:99999:7:::
adm:!:19820:0:99999:7:::
```

Рис. 3.15: Чтение файла `/etc/shadow` программой `readfile`

3.3 Исследование Sticky-бита

Выясним, установлен ли атрибут Sticky на директории `/tmp`, выполнив команду `ls -l | grep tmp`. Видим, что в конце есть атрибут `t`, что свидетельствует об атрибуте

Sticky. Далее от имени пользователя `guest` создадим файл `file01.txt` в директории `/tmp` со словом `test` следующей командой `echo "test" > /tmp/file01.txt`. Посмотрим атрибуты созданного файла с помощью `ls -l /tmp/file01.txt`. Видим, что в данный момент категории “все остальные” доступно только чтение, поэтому командой `chmod o+rw /tmp/file01.txt` разрешим им чтение и запись. Проверим, что все получилось корректно. (рис. 3.16)

```
[guest@eeparfenova Documents]$ cd ..
[guest@eeparfenova ~]$ ls -l / | grep tmp
drwxrwxrwt. 16 root root 4096 Oct 1 13:36 tmp
[guest@eeparfenova ~]$ echo "test" > /tmp/file01.txt
[guest@eeparfenova ~]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 Oct 1 13:39 /tmp/file01.txt
[guest@eeparfenova ~]$ chmod o+rw /tmp/file01.txt
[guest@eeparfenova ~]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 Oct 1 13:39 /tmp/file01.txt
```

Рис. 3.16: Права файла `file01.txt`

От имени пользователя `guest2` (не являющегося владельцем) попробуем прочитать файл `/tmp/file01.txt` командой `cat /tmp/file01.txt`. Видим, что в нем записано слово “test”. Далее попробуем дозаписать в файл слово “test2” командой `echo "test2" > /tmp/file01.txt`, но в операции нам отказано. Проверим содержимое файла, видим, что ничего не изменилось. Затем попробуем записать в файл `/tmp/file01.txt` слово “test3”, стерев при этом всю имеющуюся в файле информацию командой `echo "test3" > /tmp/file01.txt`, однако операция также не выполняется. Проверим содержимое файла, чтобы в этом убедиться. В конце концов, попробуем удалить файл командой `rm /tmp/file01.txt`, но и в этом нам отказано. (рис. 3.17)


```

[guest@eeparfenova ~]$ su - guest2
Password:
[guest2@eeparfenova ~]$ cat /tmp/file01.txt
test
[guest2@eeparfenova ~]$ echo "test2" > /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@eeparfenova ~]$ cat /tmp/file01.txt
test
[guest2@eeparfenova ~]$ echo "test3" > /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@eeparfenova ~]$ cat /tmp/file01.txt
test
[guest2@eeparfenova ~]$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'? y
rm: cannot remove '/tmp/file01.txt': Operation not permitted

```

Рис. 3.17: Манипуляции с файлом file01.txt

Попробуем снять атрибут `t` с директории `/tmp` командой `chmod -t/tmp`, предварительно новысив свои права до суперпользователя командой `su -`. Покинем режим суперпользователя и проверим правильность выполнения предыдущей команды с помощью `ls -l / | grep tmp`. Видим, что атрибут `t` в конце пропал. Прделаем все манипуляции с `file01.txt`, описанные выше, без атрибута `t` на директории `/tmp`, и увидим, что мы все также можем прочитать файл, дозапись и перезапись опять недоступны. Однако удаление файла из этой директории стало возможным. Это объясняется снятием атрибута Sticky-бит, так как он запрещал не владельцу директории удалять файлы из нее, а вот на создание и чтение запретов не накладывал. (рис. 3.18)

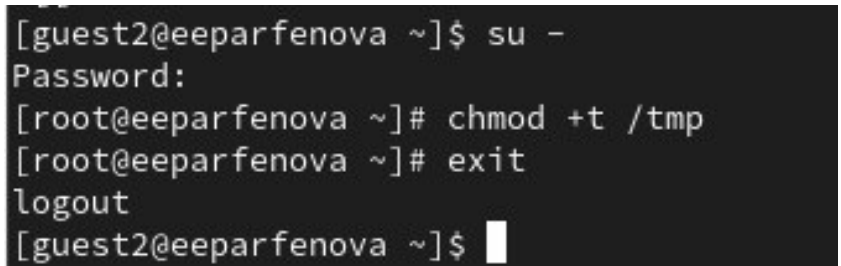
```

[guest2@eeparfenova ~]$ su -
Password:
[root@eeparfenova ~]# chmod -t /tmp
[root@eeparfenova ~]# exit
logout
[guest2@eeparfenova ~]$ ls -l / | grep tmp
drwxrwxrwx. 17 root root 4096 Oct  1 13:43 tmp
[guest2@eeparfenova ~]$ cat /tmp/file01.txt
test
[guest2@eeparfenova ~]$ echo "test2" > /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@eeparfenova ~]$ echo "test2" >> /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@eeparfenova ~]$ echo "test3" > /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@eeparfenova ~]$ cat /tmp/file01.txt
test
[guest2@eeparfenova ~]$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'? y

```

Рис. 3.18: Манипуляции с файлом file01.txt без Sticky-бит на директории

Теперь снова повысим свои права до суперпользователя и вернем атрибут `t` на директорию `/tmp` для будущей ее безопасности. (рис. 3.19)

A terminal window with a black background and white text. The prompt is [guest2@eeparfenova ~]\$ and the user enters 'su -'. The prompt changes to [root@eeparfenova ~]# and the user enters 'chmod +t /tmp'. The prompt changes back to [root@eeparfenova ~]# and the user enters 'exit'. The prompt changes to [guest2@eeparfenova ~]\$ and the user enters 'logout'. The prompt changes back to [guest2@eeparfenova ~]\$ and the user enters a space character.

```
[guest2@eeparfenova ~]$ su -  
Password:  
[root@eeparfenova ~]# chmod +t /tmp  
[root@eeparfenova ~]# exit  
logout  
[guest2@eeparfenova ~]$
```

Рис. 3.19: Возвращение атрибута `t` на директорию `tmp`

4 Выводы

Мы изучили механизм изменения идентификаторов, применения SetUID-, SetGID- и Sticky-битов. Также получили практические навыки работы в консоли с дополнительными атрибутами и рассмотрели работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

Список литературы

1. Как дать права пользователю Linux: инструкция [Электронный ресурс]. ООО «ТАЙМВЭБ.КЛАУД», 2024. URL: <https://timeweb.cloud/tutorials/linux/kak-dat-prava-polzovatelyu-linux>.
2. Права доступа в Linux [Электронный ресурс]. CodeChick.io, 2024. URL: <https://codechick.io/tutorials/unix-linux/unix-linux-permissions>.
3. Использование SETUID, SETGID и Sticky bit для расширенной настройки прав доступа в операционных системах Linux [Электронный ресурс]. RuVDS, 2021. URL: <https://ruvds.com/ru/helpcenter/suid-sgid-sticky-bit-linux/>.