

# Отчёт по лабораторной работе №5

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

Жижченко Валерия Викторовна

### Содержание

#### Цель работы

Изучение механизмов изменения идентификаторов, применения *SetUID-битов* 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.

#### Выполнение лабораторной работы

##### Создание программы

1. Входим в систему от имени пользователя *guest*.
2. Создаем 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. Скомпилируем программу и убедимся, что файл программы создан:  
\$ gcc simpleid.c -o simpleid
4. Выполняем программу *simpleid*:  
\$ ./simpleid
5. Выполняем системную программу *id* и сравниваем полученный результат с данными предыдущего пункта задания:

```
$ id
```

6. Усложняем программу, добавив вывод действительных идентификаторов.  
Получившуюся программу назовем *simpleid2.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;
}
```

7. Скомпилируем и запустим *simpleid2.c*:

```
$ gcc simpleid2.c -o simpleid2
$ ./simpleid2
```

8. От имени суперпользователя выполним следующие команды:

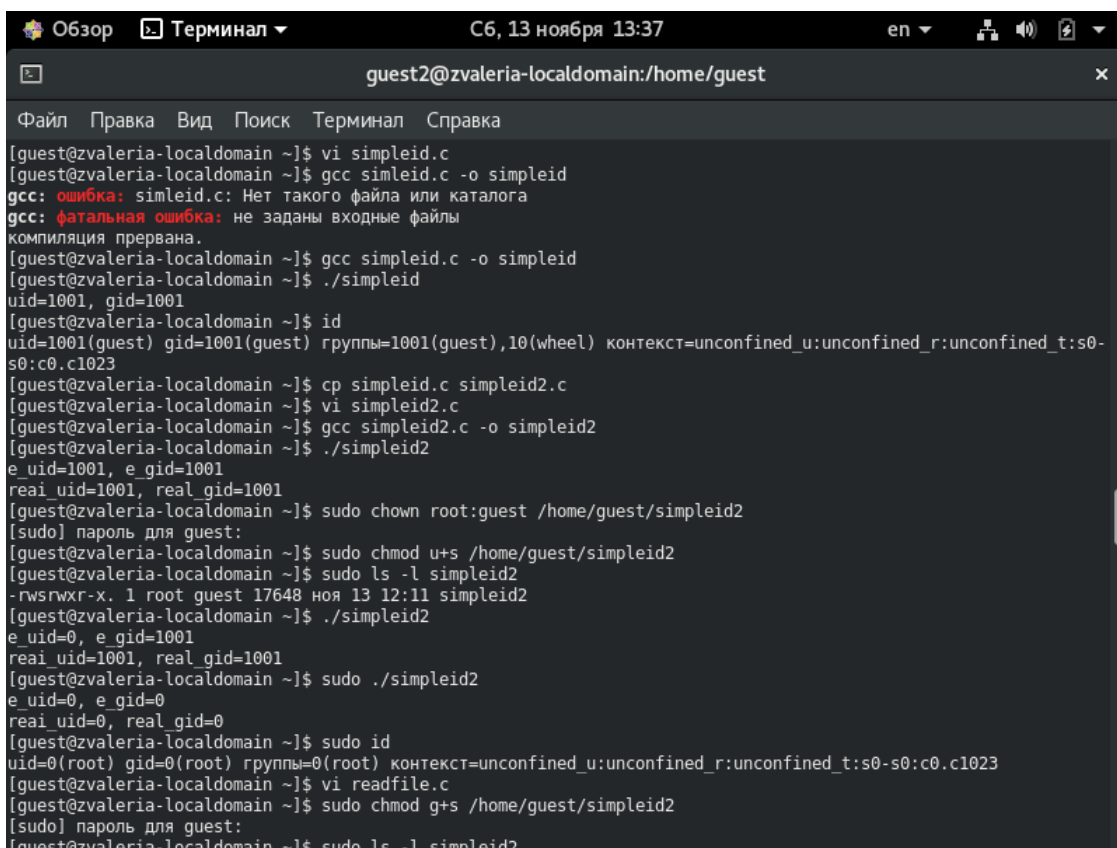
```
$ sudo chown root:guest /home/guest/simpleid2
$ sudo chmod u+s /home/guest/simpleid2
```

9. Выполним проверку правильности установки новых атрибутов и смены владельца файла *simpleid2*:

```
$ ls -l simpleid2
```

10. Запустим *simpleid2* и *id*:

```
$ ./simpleid2
$ id
```



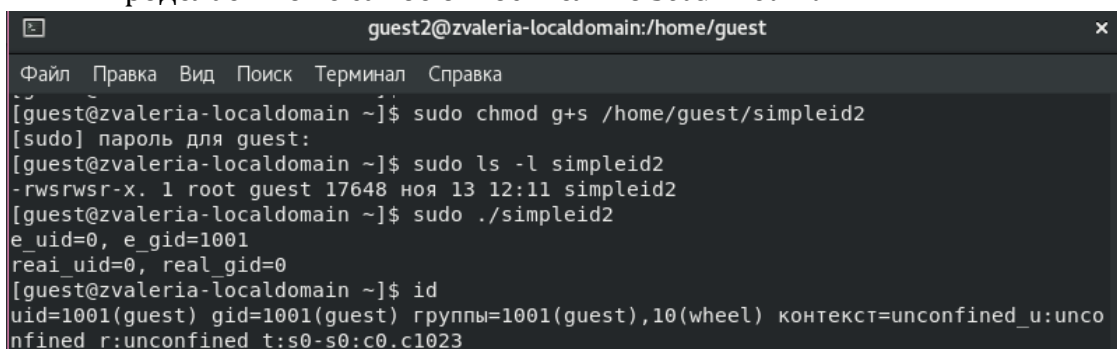
```
Обзор Терминал C6, 13 ноября 13:37 en
guest2@zvaleria-localdomain:/home/guest

Файл Правка Вид Поиск Терминал Справка

[guest@zvaleria-localdomain ~]$ vi simpleid.c
[guest@zvaleria-localdomain ~]$ gcc simpleid.c -o simpleid
gcc: ошибка: simpleid.c: Нет такого файла или каталога
gcc: фатальная ошибка: не заданы входные файлы
компиляция прервана.
[guest@zvaleria-localdomain ~]$ gcc simpleid.c -o simpleid
[guest@zvaleria-localdomain ~]$ ./simpleid
uid=1001, gid=1001
[guest@zvaleria-localdomain ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest),10(wheel) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@zvaleria-localdomain ~]$ cp simpleid.c simpleid2.c
[guest@zvaleria-localdomain ~]$ vi simpleid2.c
[guest@zvaleria-localdomain ~]$ gcc simpleid2.c -o simpleid2
[guest@zvaleria-localdomain ~]$ ./simpleid2
e_uid=1001, e_gid=1001
reai uid=1001, real_gid=1001
[guest@zvaleria-localdomain ~]$ sudo chown root:guest /home/guest/simpleid2
[sudo] пароль для guest:
[guest@zvaleria-localdomain ~]$ sudo chmod u+s /home/guest/simpleid2
[guest@zvaleria-localdomain ~]$ sudo ls -l simpleid2
-rwsrwxr-x. 1 root guest 17648 ноя 13 12:11 simpleid2
[guest@zvaleria-localdomain ~]$ ./simpleid2
e_uid=0, e_gid=1001
reai uid=1001, real_gid=1001
[guest@zvaleria-localdomain ~]$ sudo ./simpleid2
e_uid=0, e_gid=0
reai uid=0, real_gid=0
[guest@zvaleria-localdomain ~]$ sudo id
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@zvaleria-localdomain ~]$ vi readfile.c
[guest@zvaleria-localdomain ~]$ sudo chmod g+s /home/guest/simpleid2
[sudo] пароль для guest:
[guest@zvaleria-localdomain ~]$ sudo ls -l simpleid2
```

Figure 1: Операции с SetUID-битом

11. Проделаем тоже самое относительно SetGID-бита.



```
guest2@zvaleria-localdomain:/home/guest

Файл Правка Вид Поиск Терминал Справка

[guest@zvaleria-localdomain ~]$ sudo chmod g+s /home/guest/simpleid2
[sudo] пароль для guest:
[guest@zvaleria-localdomain ~]$ sudo ls -l simpleid2
-rwsrwsr-x. 1 root guest 17648 ноя 13 12:11 simpleid2
[guest@zvaleria-localdomain ~]$ sudo ./simpleid2
e_uid=0, e_gid=1001
reai uid=0, real_gid=0
[guest@zvaleria-localdomain ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest),10(wheel) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Figure 2: Операции с SetGID-битом

12. Создаем 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;
}

```

13. Откомпилируем её:

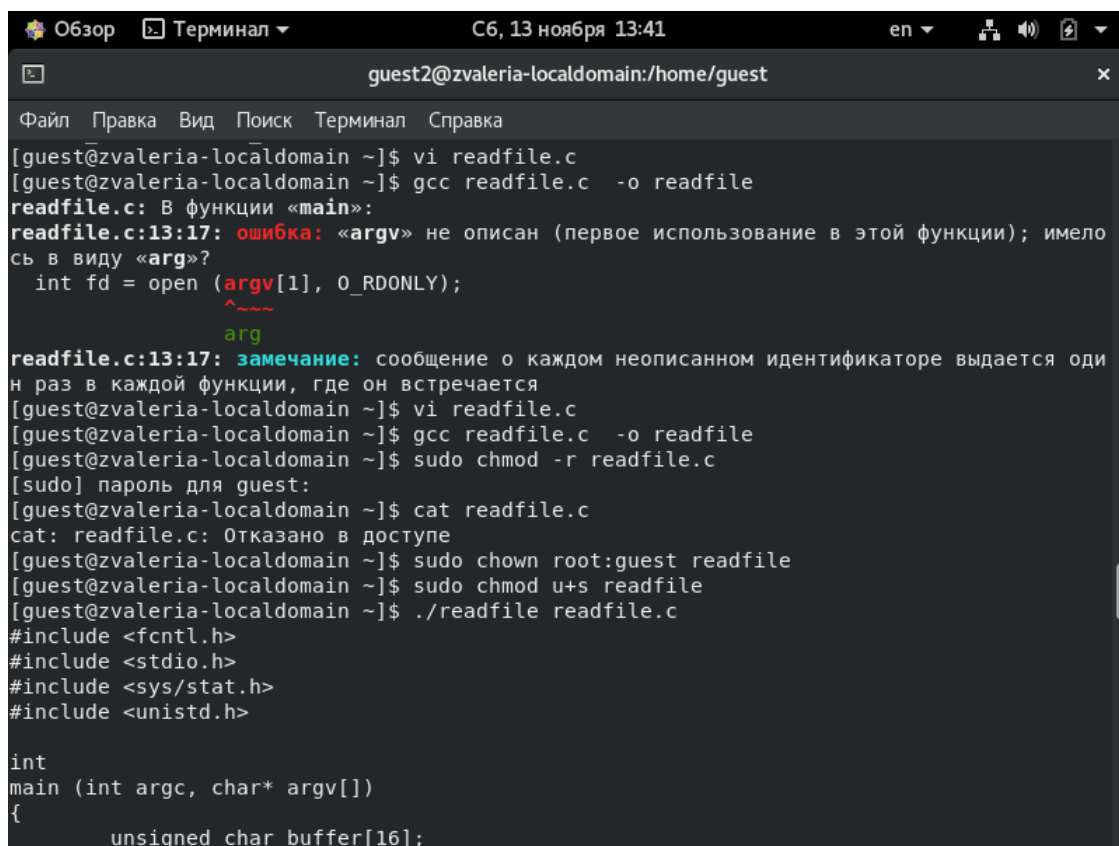
```
$ gcc readfile.c -o readfile
```

14. Сменим владельца у файла *readfile.c* и изменим права так, чтобы только суперпользователь мог прочитать его.

15. Проверяем, что пользователь *guest* не может прочитать файл *readfile.c*.

16. Сменим у программы *readfile* владельца и установим *SetUID-bit*.

17. Проверим, может ли программа *readfile* прочитать файл *readfile.c*:



```

Обзор Терминал C6, 13 ноября 13:41 en
guest2@zvaleria-localdomain:/home/guest

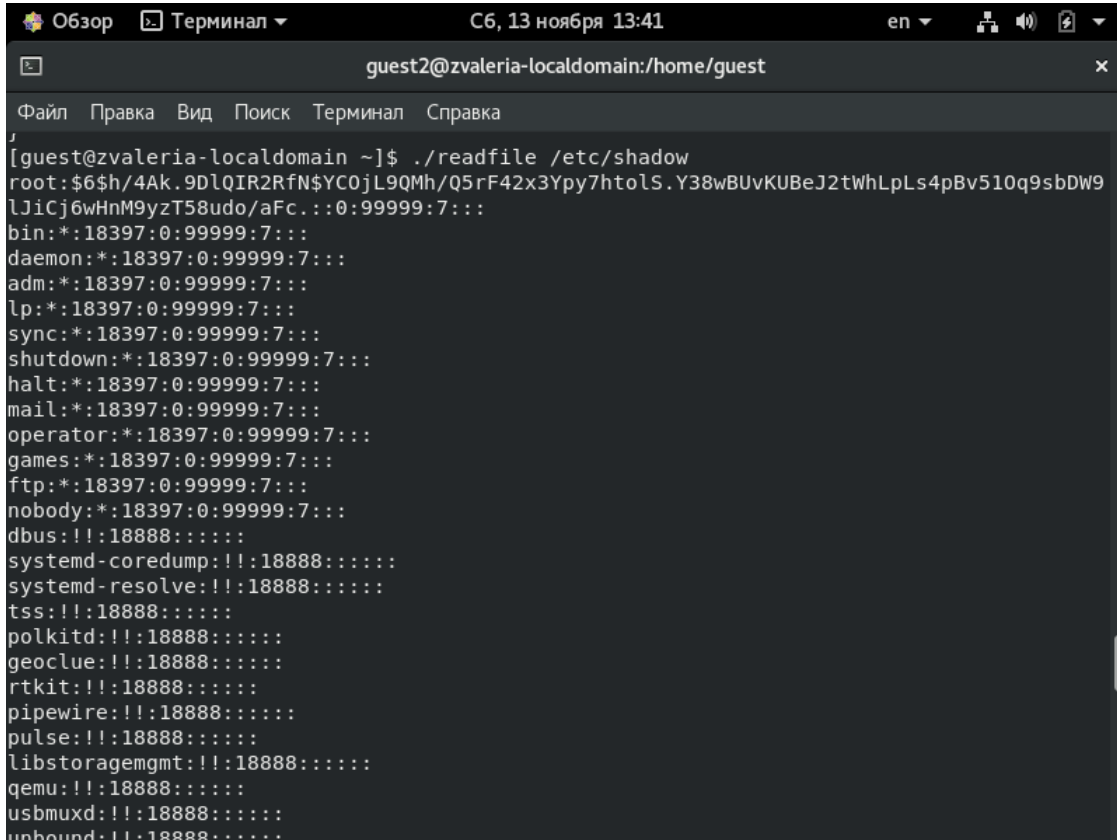
Файл Правка Вид Поиск Терминал Справка
[guest@zvaleria-localdomain ~]$ vi readfile.c
[guest@zvaleria-localdomain ~]$ gcc readfile.c -o readfile
readfile.c: В функции «main»:
readfile.c:13:17: ошибка: «argv» не описан (первое использование в этой функции); имело
сь в виду «arg»?
    int fd = open (argv[1], O_RDONLY);
                   ^~~~
                   arg
readfile.c:13:17: замечание: сообщение о каждом неопisanном идентификаторе выдается оди
н раз в каждой функции, где он встречается
[guest@zvaleria-localdomain ~]$ vi readfile.c
[guest@zvaleria-localdomain ~]$ gcc readfile.c -o readfile
[guest@zvaleria-localdomain ~]$ sudo chmod -r readfile.c
[sudo] пароль для guest:
[guest@zvaleria-localdomain ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@zvaleria-localdomain ~]$ sudo chown root:guest readfile
[guest@zvaleria-localdomain ~]$ sudo chmod u+s readfile
[guest@zvaleria-localdomain ~]$ ./readfile readfile.c
#include <fcntl.h>
#include <stdio.h>
#include <sys/stat.h>
#include <unistd.h>

int
main (int argc, char* argv[])
{
    unsigned char buffer[16];

```

Figure 3: Выполнение пунктов 12-17

18. Проверим, может ли программа *readfile* прочитать файл */etc/shadow*:



The screenshot shows a terminal window titled "guest2@zvaleria-localdomain:/home/guest". The terminal output shows the command `./readfile /etc/shadow` being executed. The output is a long string of characters, including letters, numbers, and symbols, which appears to be a hash or a representation of the shadow file's contents. The terminal window has a menu bar with "Файл", "Правка", "Вид", "Поиск", "Терминал", and "Справка". The top status bar shows "С6, 13 ноября 13:41" and "en".

```
[guest@zvaleria-localdomain ~]$ ./readfile /etc/shadow
root:$6$h/4Ak.9DlQIR2RfN$YCOjL9QMh/Q5rF42x3Ypy7htoLS.Y38wBUvKUBeJ2tWhLpLs4pBv510q9sbDW9
lJiCj6wHnM9yzT58udo/aFc.:0:99999:7:::
bin:!:18397:0:99999:7:::
daemon:!:18397:0:99999:7:::
adm:!:18397:0:99999:7:::
lp:!:18397:0:99999:7:::
sync:!:18397:0:99999:7:::
shutdown:!:18397:0:99999:7:::
halt:!:18397:0:99999:7:::
mail:!:18397:0:99999:7:::
operator:!:18397:0:99999:7:::
games:!:18397:0:99999:7:::
ftp:!:18397:0:99999:7:::
nobody:!:18397:0:99999:7:::
dbus:!!:18888::::::
systemd-coredump:!!:18888::::::
systemd-resolve:!!:18888::::::
tss:!!:18888::::::
polkitd:!!:18888::::::
geoclue:!!:18888::::::
rtkit:!!:18888::::::
pipewire:!!:18888::::::
pulse:!!:18888::::::
libstoragemgmt:!!:18888::::::
qemu:!!:18888::::::
usbmuxd:!!:18888::::::
unbound:!!:18888::::::
```

Figure 4: Выполнение пункта 18

## Исследование *Sticky-бита*

1. Выясним, установлен ли атрибут *Sticky* на директории */tmp*:

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

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

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

3. Просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей "все остальные":

```
guest$ ls -l /tmp/file01.txt
guest$ chmod o+rw /tmp/file01.txt
guest$ ls -l /tmp/file01.txt
```

4. От пользователя *guest2* попробуем прочитать файл */tmp/file01.txt*:

```
guest2$ cat /tmp/file01.txt
```

5. От пользователя *guest2* попробуем дозаписать в файл */tmp/file01.txt* слово *test2*:

```
guest2$ echo "test2" > /tmp/file01.txt
```

6. Проверяем содержимое файла:

```
guest2$ cat /tmp/file01.txt
```

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

```
guest2$ echo "test3" > /tmp/file01.txt
```

8. Проверяем содержимое файла:

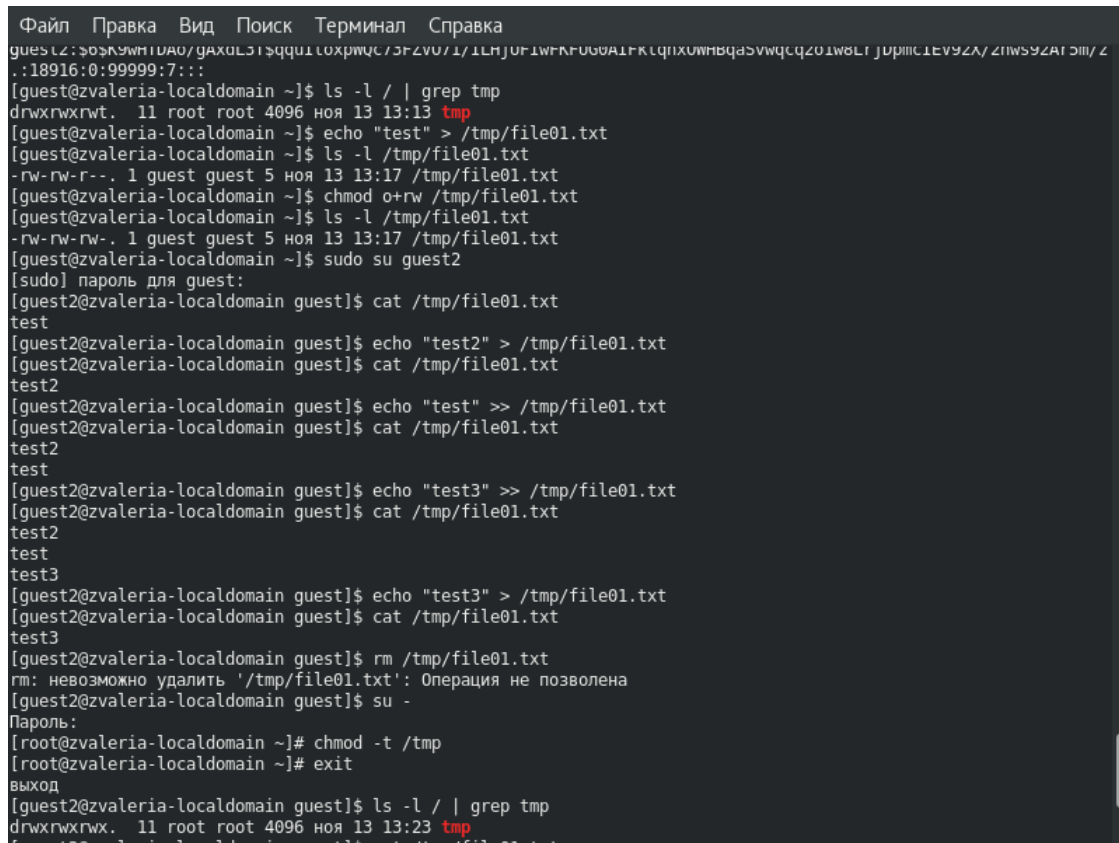
```
guest2$ cat /tmp/file01.txt
```

9. От пользователя *guest2* попробуем удалить файл */tmp/file01.txt*:

```
guest2$ rm /tmp/file01.txt
```

10. От имени суперпользователя выполним команду, снимающую атрибут *t* с директории */tmp*:

```
guest2$ sudo chmod -t /tmp
```



```
Файл Правка Вид Поиск Терминал Справка
guest2$ echo "test3" > /tmp/file01.txt
[guest2@zvaleria-localdomain ~]$ ls -l / | grep tmp
drwxrwxrwt. 11 root root 4096 ноя 13 13:13 tmp
[guest2@zvaleria-localdomain ~]$ echo "test" > /tmp/file01.txt
[guest2@zvaleria-localdomain ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 ноя 13 13:17 /tmp/file01.txt
[guest2@zvaleria-localdomain ~]$ chmod o+rw /tmp/file01.txt
[guest2@zvaleria-localdomain ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 ноя 13 13:17 /tmp/file01.txt
[guest2@zvaleria-localdomain ~]$ sudo su guest2
[sudo] пароль для guest:
[guest2@zvaleria-localdomain guest]$ cat /tmp/file01.txt
test
[guest2@zvaleria-localdomain guest]$ echo "test2" > /tmp/file01.txt
[guest2@zvaleria-localdomain guest]$ cat /tmp/file01.txt
test2
[guest2@zvaleria-localdomain guest]$ echo "test" >> /tmp/file01.txt
[guest2@zvaleria-localdomain guest]$ cat /tmp/file01.txt
test2
test
[guest2@zvaleria-localdomain guest]$ echo "test3" >> /tmp/file01.txt
[guest2@zvaleria-localdomain guest]$ cat /tmp/file01.txt
test2
test
test3
[guest2@zvaleria-localdomain guest]$ echo "test3" > /tmp/file01.txt
[guest2@zvaleria-localdomain guest]$ cat /tmp/file01.txt
test3
[guest2@zvaleria-localdomain guest]$ rm /tmp/file01.txt
rm: невозможно удалить '/tmp/file01.txt': Операция не позволена
[guest2@zvaleria-localdomain guest]$ su -
Пароль:
[root@zvaleria-localdomain ~]# chmod -t /tmp
[root@zvaleria-localdomain ~]# exit
выход
[guest2@zvaleria-localdomain guest]$ ls -l / | grep tmp
drwxrwxrwt. 11 root root 4096 ноя 13 13:23 tmp
```

Figure 5: Выполнение с установленным Sticky-битом

12. От пользователя *guest2* проверяем, что атрибута *t* у директории */tmp* нет:

```
guest2$ ls -l / | grep tmp
```

13. Повторяем предыдущие шаги.

14. Повысим свои права до суперпользователя и вернем атрибут *t* на директорию */tmp*:

```
guest2$ sudo chmod +t /tmp
```

```
[guest2@zvaleria-localdomain guest]$ cat /tmp/file01.txt
test3
[guest2@zvaleria-localdomain guest]$ echo "test" >> /tmp/file01.txt
[guest2@zvaleria-localdomain guest]$ cat /tmp/file01.txt
test3
test
[guest2@zvaleria-localdomain guest]$ echo "test3" > /tmp/file01.txt
[guest2@zvaleria-localdomain guest]$ cat /tmp/file01.txt
test3
[guest2@zvaleria-localdomain guest]$ rm /tmp/file01.txt
[guest2@zvaleria-localdomain guest]$ su -
Пароль:
[root@zvaleria-localdomain ~]# chmod +t /tmp
[root@zvaleria-localdomain ~]# exit
выход
[guest2@zvaleria-localdomain guest]$
```

Figure 6: Выполнение со неустановленным Sticky-битом

## Вывод

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