

# **Лабораторная работа №5**

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

Щербак Маргарита Романовна, НПИБд-02-21

2024

# Содержание

<b>Цель работы</b>	<b>4</b>
<b>Теоретическое введение</b>	<b>5</b>
<b>Выполнение лабораторной работы</b>	<b>6</b>
Создание программы . . . . .	6
Исследование Sticky-бита . . . . .	14
<b>Вывод</b>	<b>17</b>
<b>Библиография</b>	<b>18</b>

## Список иллюстраций

1	Содержимое программы simpleid.c . . . . .	6
2	Выполнение команд . . . . .	7
3	Усложненная программа . . . . .	8
4	Выполнение команд . . . . .	9
5	Выполнение команд . . . . .	10
6	Содержимое программы readfile.c . . . . .	11
7	Смена владельца . . . . .	12
8	Проверка . . . . .	12
9	Проверка . . . . .	13
10	Проверка . . . . .	14
11	Выполнение команд . . . . .	15
12	Снятие Sticky-бита . . . . .	16
13	Возвращение Sticky-бита . . . . .	16

## **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 [1].

# Теоретическое введение

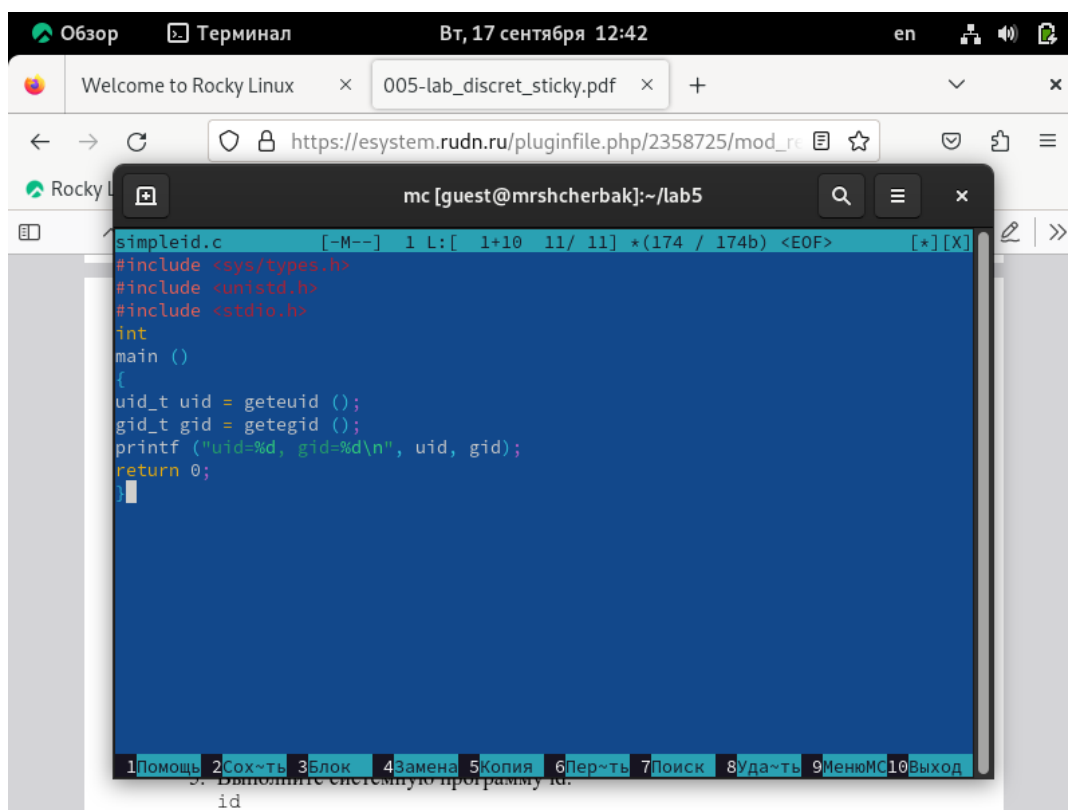
Информационная безопасность – это защищенность информации и поддерживающей инфраструктуры от случайных или преднамеренных воздействий естественного или искусственного характера, чреватых нанесением ущерба владельцам или пользователям информации и поддерживающей инфраструктуры.

Права доступа в системах управляют тем, какие операции может выполнять пользователь с определенными файлами и папками. Правильная настройка прав доступа помогает создать безопасную среду, где никто не сможет изменять ваши данные или нарушать работу важных системных файлов. Помимо групп root и users, в системе существует множество других, которые созданы для управления доступом программ к общим ресурсам. Участники каждой группы получают права на чтение или изменение конкретных файлов и каталогов, что регулирует их доступ и действия. Эти же права передаются процессам, которые запускает пользователь [2].

# Выполнение лабораторной работы

## Создание программы

1. Я подготовила лабораторный стенд. У меня был установлен gcc. Я вошла в систему от имени пользователя guest. Создала программу simpleid.c (рис.1).

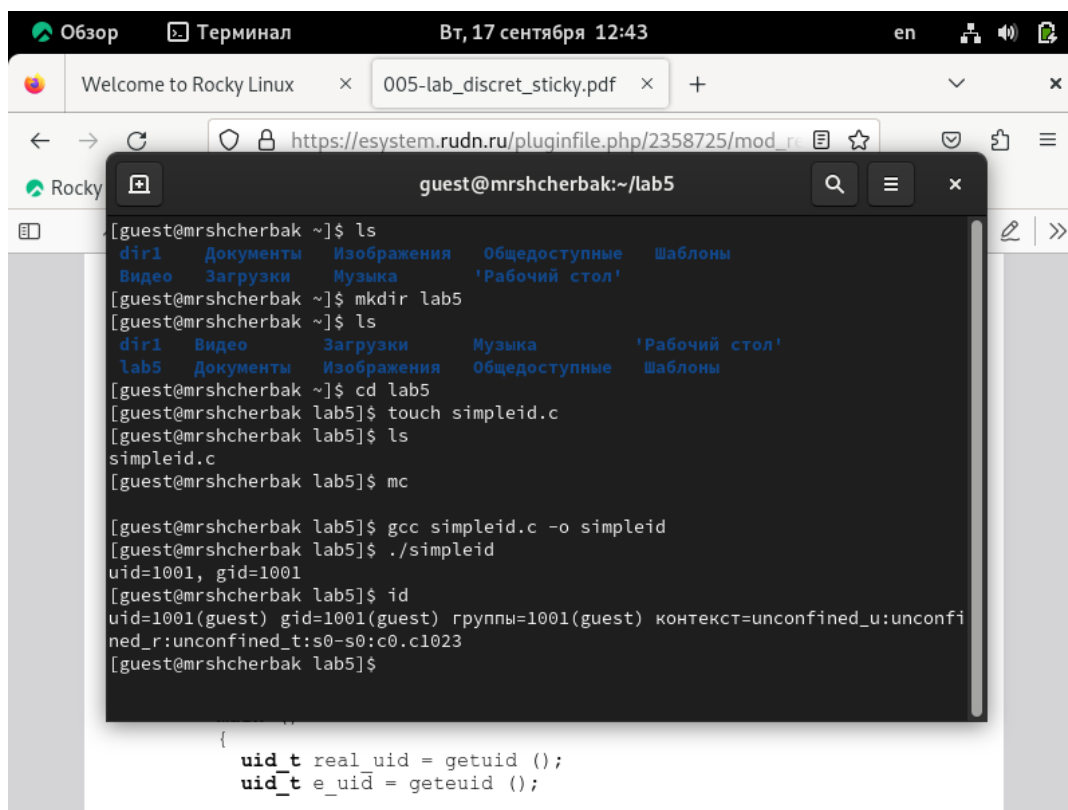


```
simpleid.c  [-M--]  1  L: [  1+10  11/ 11]  *(174 / 174b)  <EOF>  [*][X]
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
int
main ()
{
    uid_t uid = geteuid ();
    gid_t gid = getegid ();
    printf ("uid=%d, gid=%d\n", uid, gid);
    return 0;
}
```

Рис. 1: Содержимое программы simpleid.c

2. Скомпилировала программу и убедилась, что файл программы создан. Выполнила

программу simpleid. Выполнила системную программу id и сравнила полученный результат с данными предыдущего пункта задания. Результаты совпадают (рис.2).



```
[guest@mrshcherbak ~]$ ls
dir1  Документы  Изображения  Общедоступные  Шаблоны
Видео  Загрузки  Музыка  'Рабочий стол'
[guest@mrshcherbak ~]$ mkdir lab5
[guest@mrshcherbak ~]$ ls
dir1  Видео  Загрузки  Музыка  'Рабочий стол'
lab5  Документы  Изображения  Общедоступные  Шаблоны
[guest@mrshcherbak ~]$ cd lab5
[guest@mrshcherbak lab5]$ touch simpleid.c
[guest@mrshcherbak lab5]$ ls
simpleid.c
[guest@mrshcherbak lab5]$ mc

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

Рис. 2: Выполнение команд

3. Усложнила программу, добавив вывод действительных идентификаторов. Получившуюся программу назвала simpleid2.c (рис.3).

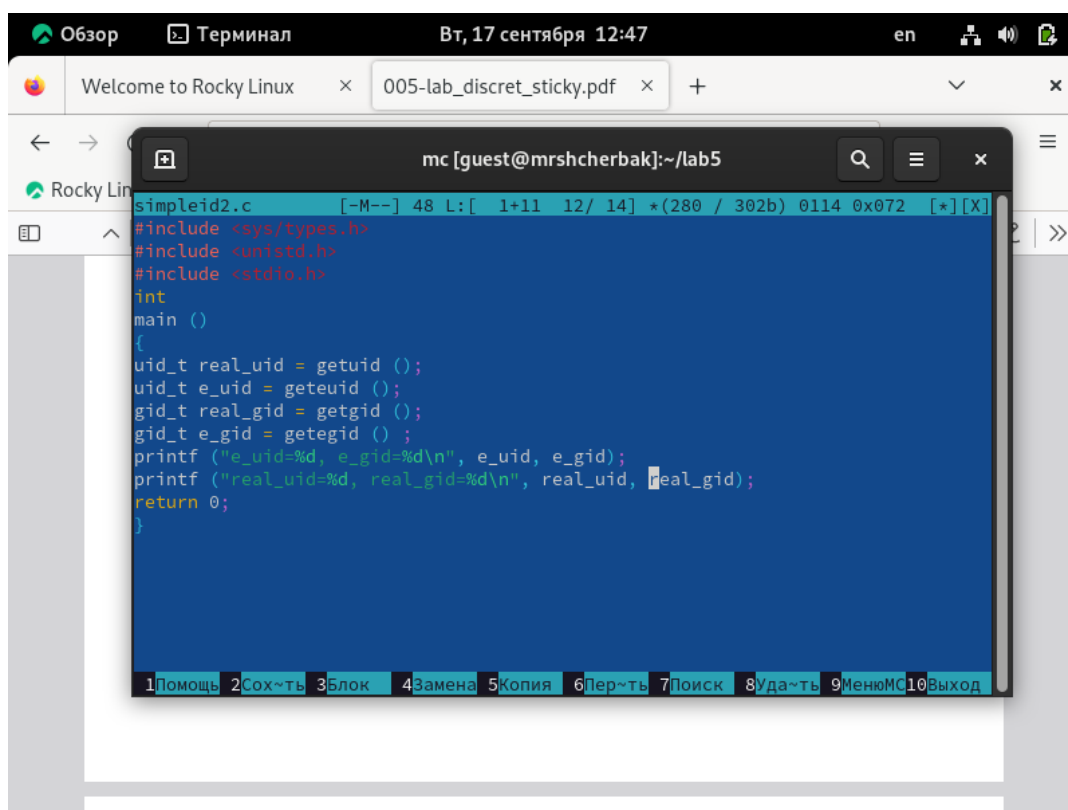


Рис. 3: Усложненная программа

4. Скомпилировала и запустила simpleid2.c. От имени суперпользователя выполнила команды: `chown root:guest /home/guest/simpleid2` и `chmod u+s /home/guest/simpleid2` (рис.4).



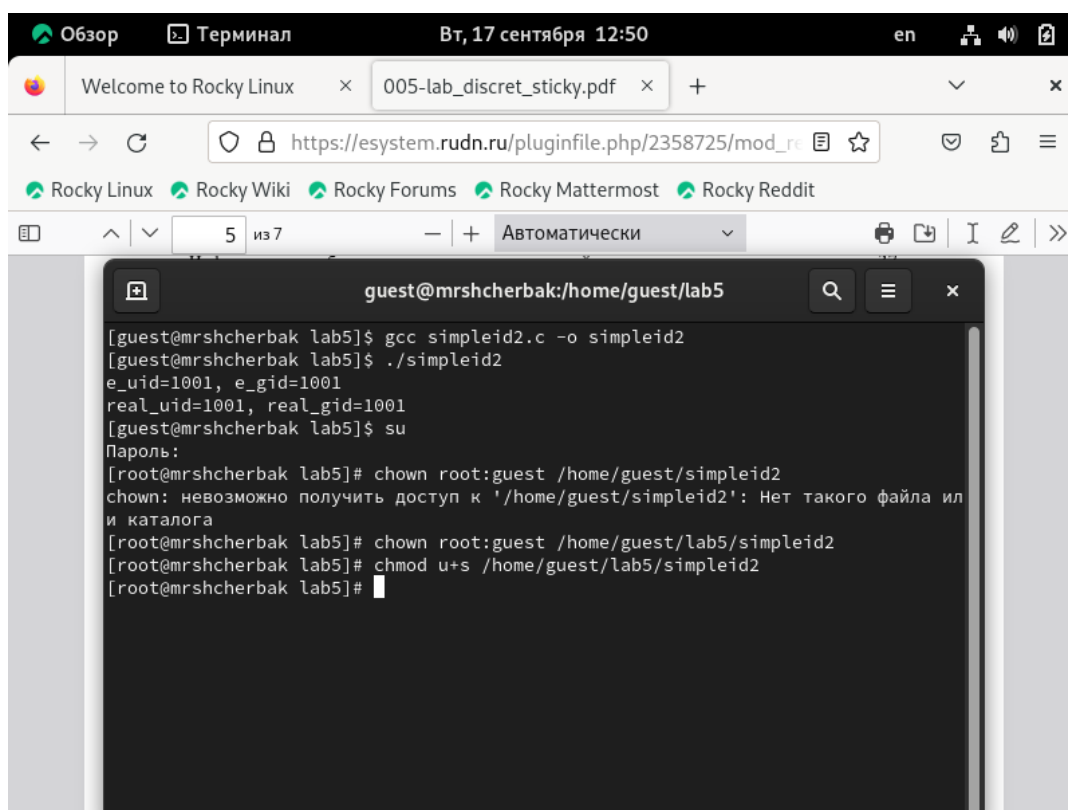


Рис. 4: Выполнение команд

- `sudo`: выполняет одну команду с правами суперпользователя (`root`) без полного переключения на него. Требуется пароль.
  - `su`: полностью переключает на другого пользователя (обычно `root`) с его окружением, требуя пароль этого пользователя.
5. Выполнила проверку правильности установки новых атрибутов и смены владельца файла `simpleid2`. Запустила `simpleid2` и `id`. Результаты совпадают (рис.5).

```
guest@mrshcherbak:~/home/guest/lab5
[guest@mrshcherbak lab5]$ gcc simpleid2.c -o simpleid2
[guest@mrshcherbak lab5]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@mrshcherbak lab5]$ su
Пароль:
[root@mrshcherbak lab5]# chown root:guest /home/guest/simpleid2
chown: невозможно получить доступ к '/home/guest/simpleid2': Нет такого файла или каталога
[root@mrshcherbak lab5]# chown root:guest /home/guest/lab5/simpleid2
[root@mrshcherbak lab5]# chmod u+s /home/guest/lab5/simpleid2
[root@mrshcherbak lab5]# ls -l simpleid2
-rwsr-xr-x. 1 root guest 17720 сен 17 12:48 simpleid2
[root@mrshcherbak lab5]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@mrshcherbak lab5]# id
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@mrshcherbak lab5]#

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

Рис. 5: Выполнение команд

6. Проделала то же самое относительно SetGID-бита. Создала программу readfile.c (рис.6).

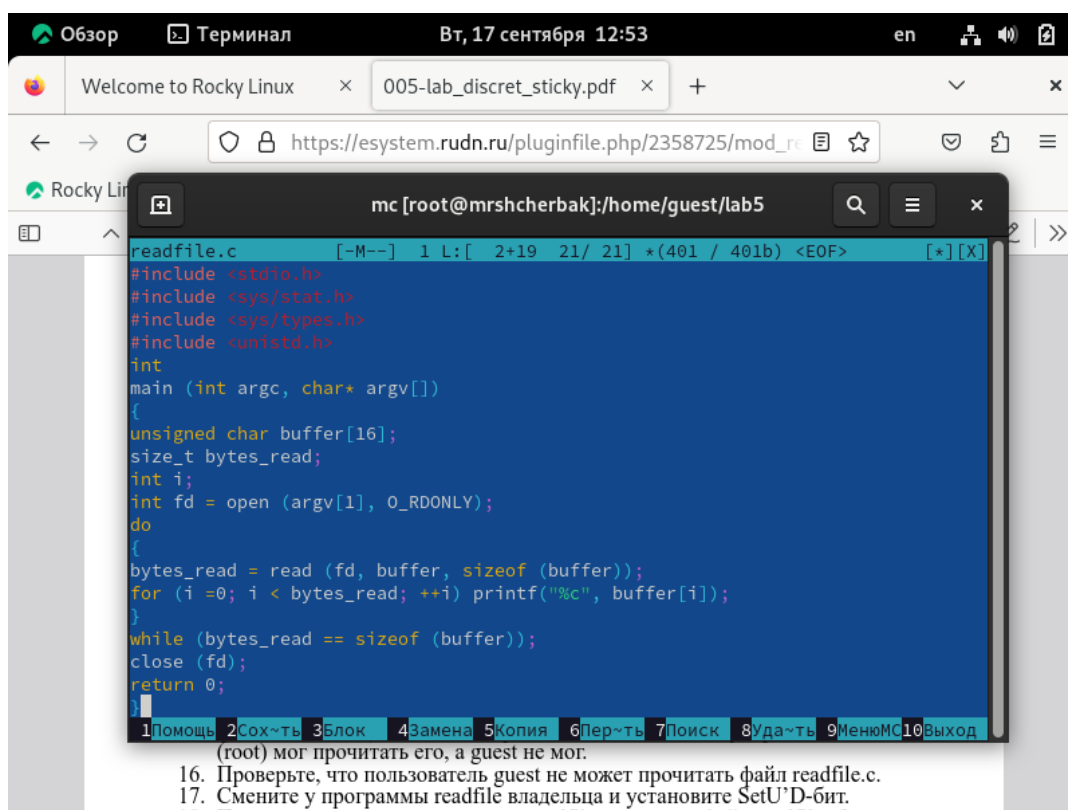


Рис. 6: Содержимое программы readfile.c

7. Откомпилировала её. Сменила владельца у файла readfile.c и изменила права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог (рис.7).

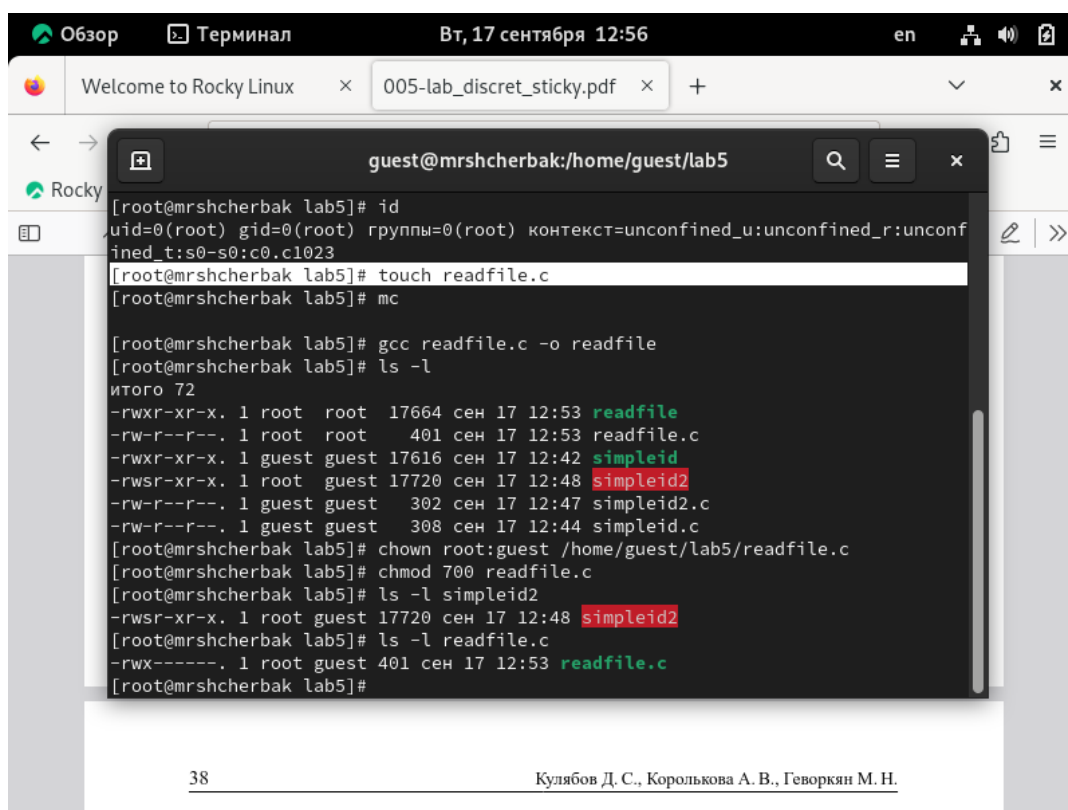


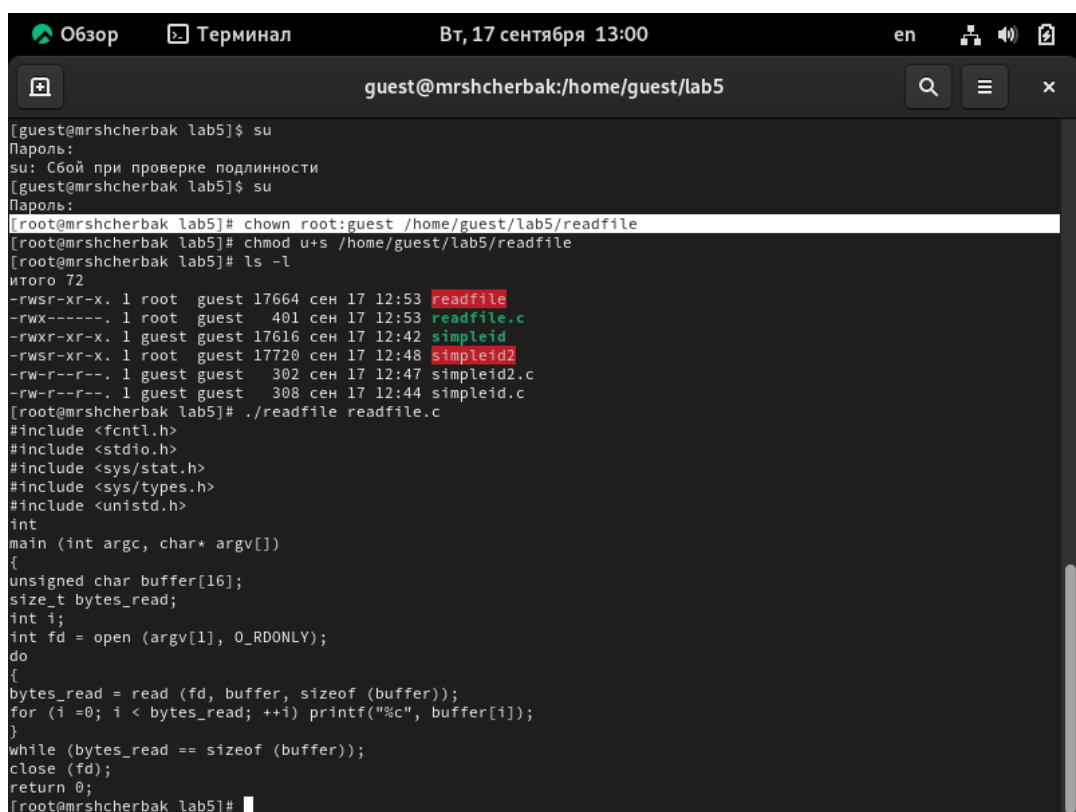
Рис. 7: Смена владельца

8. Проверила, что пользователь guest не может прочитать файл readfile.c (рис.8).



Рис. 8: Проверка

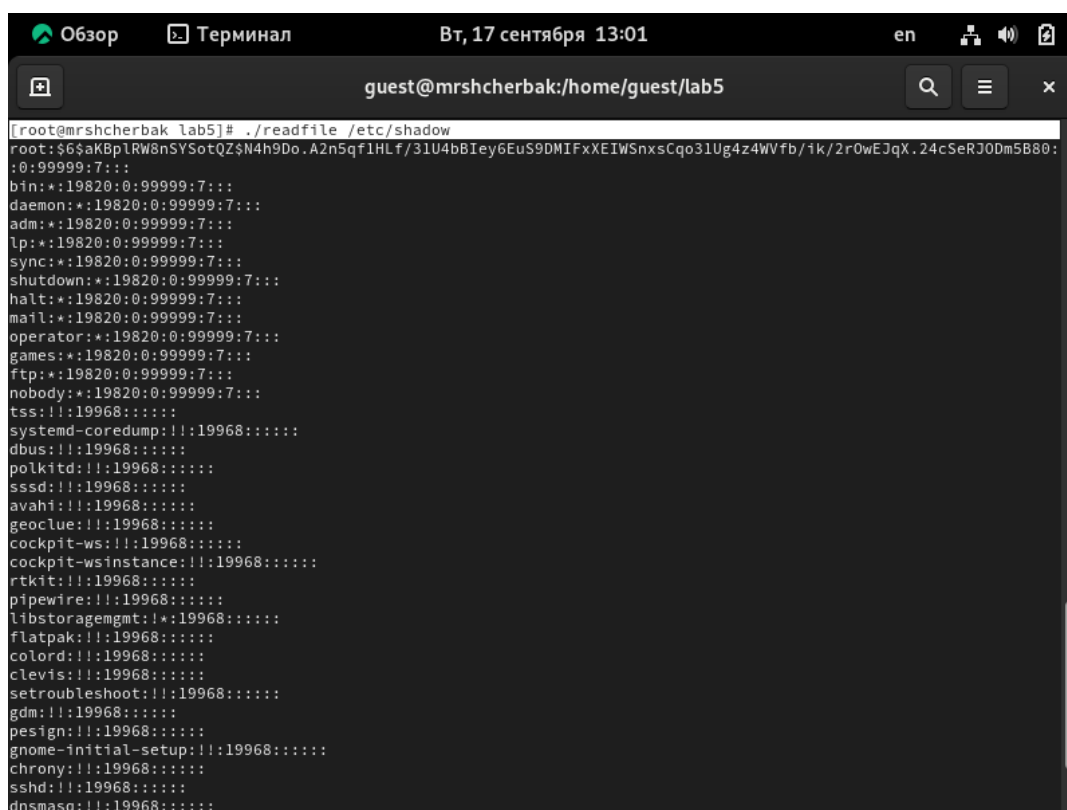
9. Сменила у программы readfile владельца и установила SetU'D-бит Проверила, может ли программа readfile прочитать файл readfile.c (рис.9).



```
[guest@mrshcherbak lab5]$ su
Пароль:
su: Сбой при проверке подлинности
[guest@mrshcherbak lab5]$ su
Пароль:
[root@mrshcherbak lab5]# chown root:guest /home/guest/lab5/readfile
[root@mrshcherbak lab5]# chmod u+s /home/guest/lab5/readfile
[root@mrshcherbak lab5]# ls -l
итого 72
-rwsr-xr-x. 1 root  guest 17664 сен 17 12:53 readfile
-rwx----- 1 root  guest   401 сен 17 12:53 readfile.c
-rwxr-xr-x. 1 guest  guest 17616 сен 17 12:42 simpleid
-rwsr-xr-x. 1 root  guest 17720 сен 17 12:48 simpleid2
-rw-r--r-- 1 guest  guest   302 сен 17 12:47 simpleid2.c
-rw-r--r-- 1 guest  guest   308 сен 17 12:44 simpleid.c
[root@mrshcherbak lab5]# ./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;
}
[root@mrshcherbak lab5]#
```

Рис. 9: Проверка

10. Проверила, может ли программа readfile прочитать файл /etc/shadow (рис.10).



The screenshot shows a terminal window with a dark background. At the top, there is a header bar with icons for 'Обзор' (Overview), 'Терминал' (Terminal), and a clock showing 'Вт, 17 сентября 13:01'. Below the header, the terminal title bar reads 'guest@mrshcherbak:/home/guest/lab5'. The terminal content shows a root prompt followed by the command `./readfile /etc/shadow`. The output is a long string of text representing the contents of the `/etc/shadow` file, starting with `root:$6$aK8pLRW8nSYsotQZ$N4h9Do.A2n5qf1HLf/31U4b8Iey6EuS9DMIFxXEIWSnxsCqo31Ug4z4wVfb/ik/2r0wEJqX.24cSeR3J0dm5B80:` and listing various system users like `bin`, `daemon`, `adm`, `lp`, `sync`, `shutdown`, `halt`, `mail`, `operator`, `games`, `ftp`, `nobody`, `tss`, `systemd-coredump`, `dbus`, `polkitd`, `sssd`, `avahi`, `geoclue`, `cockpit-ws`, `cockpit-wsinstance`, `rtkit`, `pipewire`, `libstoragemgmt`, `flatpak`, `colord`, `clevis`, `setroubleshoot`, `gdm`, `pesign`, `gnome-initial-setup`, `chrony`, `sshd`, and `dnsmasq`.

Рис. 10: Проверка

Программа может прочитать оба файла.

## Исследование Sticky-бита

1. Выяснила, установлен ли атрибут Sticky на директории `/tmp`, для чего выполните команду `ls -l | grep tmp`. Атрибут “t” установлен. От имени пользователя `guest` создала файл `file01.txt` в директории `/tmp` со словом `test`. Просмотрела атрибуты у только что созданного файла и разрешила чтение и запись для категории пользователей «все остальные». От пользователя `guest2` (не являющегося владельцем) попробовала прочитать файл `/tmp/file01.txt`. От пользователя `guest2` попробовала дозаписать в файл `/tmp/file01.txt` слово `test2`. От пользователя `guest2` попробовала удалить файл `/tmp/file01.txt` командой `rm /tmp/file01.txt` (рис.11). Операции выполнить не удалось.

```
[guest@mrshcherbak lab5]$ ls -l / | grep tmp
drwxrwxrwt. 17 root root 4096 сен 17 12:58 tmp
[guest@mrshcherbak lab5]$ echo "test" > /tmp/file01.txt
[guest@mrshcherbak lab5]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 сен 17 13:02 /tmp/file01.txt
[guest@mrshcherbak lab5]$ chmod o+rw /tmp/file01.txt
[guest@mrshcherbak lab5]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 сен 17 13:02 /tmp/file01.txt
[guest@mrshcherbak lab5]$ su guest2
Пароль:
[guest2@mrshcherbak lab5]$ cat /tmp/file01.txt
test
[guest2@mrshcherbak lab5]$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@mrshcherbak lab5]$ echo "test2" >> /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@mrshcherbak lab5]$ echo "test3" > /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@mrshcherbak lab5]$ rm /tmp/file01.txt
rm: невозможно удалить '/tmp/file01.txt': Нет такого файла или каталога
[guest2@mrshcherbak lab5]$
```

Рис. 11: Выполнение команд

2. Повысила свои права до суперпользователя следующей командой `su -` и выполнила после этого команду, снимающую атрибут `t` (Sticky-бит) с директории `/tmp`. Покинула режим суперпользователя командой `exit`. От пользователя `guest2` проверила, что атрибута `t` у директории `/tmp` нет (рис.12).

```

[guest2@mrshcherbak lab5]$ su -
Пароль:
[root@mrshcherbak ~]# chmod -t /tmp
[root@mrshcherbak ~]# exit
выход
[guest2@mrshcherbak lab5]$ ls -l / | grep tmp
drwxrwxrwx. 18 root root 4096 сен 17 13:07 tmp
[guest2@mrshcherbak lab5]$

```

Рис. 12: Снятие Sticky-бита

3. Повторила предыдущие шаги. Повысила свои права до суперпользователя и вернула атрибут t на директорию /tmp (рис.13).

```

[guest2@mrshcherbak guest]$ cat /tmp/file01.txt
test
[guest2@mrshcherbak guest]$ echo "test2" >>/tmp/file01.txt
[guest2@mrshcherbak guest]$ cat /tmp/file01.txt
test
test2
[guest2@mrshcherbak guest]$ echo "test3" >/tmp/file01.txt
[guest2@mrshcherbak guest]$ cat /tmp/file01.txt
test3
[guest2@mrshcherbak guest]$ rm /tmp/file01.txt
[guest2@mrshcherbak guest]$ su -
Пароль:
[root@mrshcherbak ~]# chmod +t /tmp
[root@mrshcherbak ~]# exit
выход
[guest2@mrshcherbak guest]$

```

Рис. 13: Возвращение Sticky-бита



## **Вывод**

Таким образом, в ходе ЛР№5 я изучила механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получил практические навыки работы в консоли с дополнительными атрибутами. Рассмотрел работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

# Библиография

1. Методические материалы курса.
2. Chmod. [Электронный ресурс]. М. URL: Файловая система (Дата обращения: 17.09.2024).