

# Лабораторная работа № 5

Дискреционное разграничение прав в Linux. Исследование влияния  
дополнительных атрибутов

Аксёнова Алина Владимировна

# Содержание

Цель работы	5
Задание	6
Теоретическое введение	7
Ход работы	9
Выводы	17
Библиографический список	18

# List of Figures

0.1	Листинг simpleid.c . . . . .	9
0.2	Запуск simpleid.c . . . . .	10
0.3	Сверка simpleid.c и id . . . . .	10
0.4	Листинг simpleid2.c . . . . .	10
0.5	Запуск simpleid2.c . . . . .	11
0.6	Смена владельца и добавление SetUID бита . . . . .	11
0.7	Запуск simpleid2.c . . . . .	11
0.8	Сверка simpleid2.c и id . . . . .	11
0.9	Добавление SetGID бита . . . . .	12
0.10	Запуск simpleid2.c . . . . .	12
0.11	Код программы . . . . .	12
0.12	Компиляция . . . . .	13
0.13	Смена владельца readfile.c . . . . .	13
0.14	Проверка на cat из-под guest'a . . . . .	13
0.15	Смена владельца readfile и SetUID . . . . .	13
0.16	Чтение readfile.c программой readfile . . . . .	14
0.17	Чтение /etc/shadow программой readfile . . . . .	14
0.18	Создание файла . . . . .	15
0.19	Правка прав файла . . . . .	15
0.20	Проверка прав . . . . .	15
0.21	Тестирование файла . . . . .	15
0.22	Попытка удаления файла . . . . .	15
0.23	Удаление Sticky-бита . . . . .	16
0.24	Повторное тестирование файла . . . . .	16
0.25	Возвращение Sticky-бита . . . . .	16

# List of Tables

## 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.

# Задание

Закрепить дискреционное разграничение прав в Linux с дополнительными атрибутами.

# Теоретическое введение

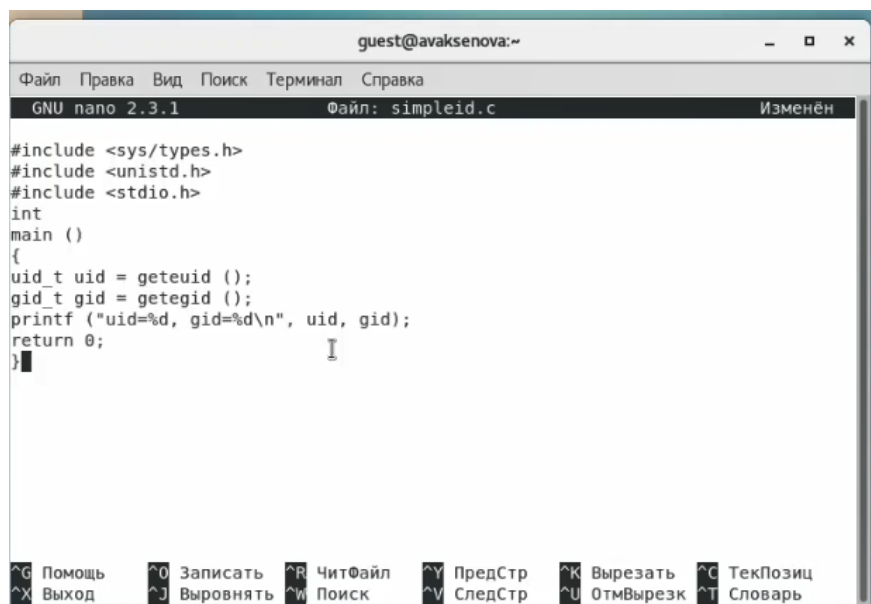
В Linux, как и в любой многопользовательской системе, абсолютно естественным образом возникает задача разграничения доступа субъектов — пользователей к объектам — файлам дерева каталогов. Один из подходов к разграничению доступа — так называемый дискреционный - предполагает назначение владельцев объектов, которые по собственному усмотрению определяют права доступа субъектов (других пользователей) к объектам (файлам), которыми владеют. Дискреционные механизмы разграничения доступа используются для разграничения прав доступа процессов как обычных пользователей, так и для ограничения прав системных программ в (например, служб операционной системы), которые работают от лица псевдопользовательских учетных записей. Чтобы получить доступ к файлам в Linux, используются разрешения. Эти разрешения назначаются трем объектам: файлу, группе и другому объекту. Для управления правами используется команда `chmod`. При использовании `chmod` в относительном режиме вы работаете с тремя индикаторами, чтобы указать, что вы хотите сделать. Сначала вы указываете, для кого вы хотите изменить разрешения. Для этого вы можете выбрать между пользователем (u), группой (g) и другими (o). Затем вы используете оператор для добавления или удаления разрешений из текущего режима или устанавливаете их абсолютно. В конце вы используете r(read), w(write) и x(execute), чтобы указать, какие разрешения вы хотите установить. При использовании `chmod` вы можете устанавливать разрешения для пользователя (user), группы (group) и других (other). Помимо основных разрешений, о которых вы только что прочитали, в Linux также есть набор расширенных разрешений. Это не те разрешения, которые вы устанавливаете по

умолчанию, но иногда они предоставляют полезное дополнение.



## Ход работы

1. Готовим систему и входим из-под пользователя guest. Пишем программу simpleid.c. Компилируем программу, запускаем, видим вывод uid и gid пользователя, сравниваем вывод с id (все совпадает). (Рис. 0.1, 0.2, 0.3).



The image shows a terminal window titled 'guest@avaksenova:~'. Inside, the GNU nano 2.3.1 editor is open with the file 'simpleid.c'. The code is as follows:

```
#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;
}
```

The bottom of the window displays a menu with various keyboard shortcuts and their corresponding actions in Russian, such as '^G Помощь' (Help), '^O Записать' (Write), '^R ЧитФайл' (Read File), '^Y ПредСтр' (Previous Page), '^K Вырезать' (Cut), '^C ТекПозиц' (Text Position), '^X Выход' (Exit), '^J Выворнять' (Align), '^W Поиск' (Search), '^V СледСтр' (Next Page), '^U ОтмВырезк' (Unmark Cut), and '^I Словарь' (Dictionary).

Figure 0.1: Листинг simpleid.c

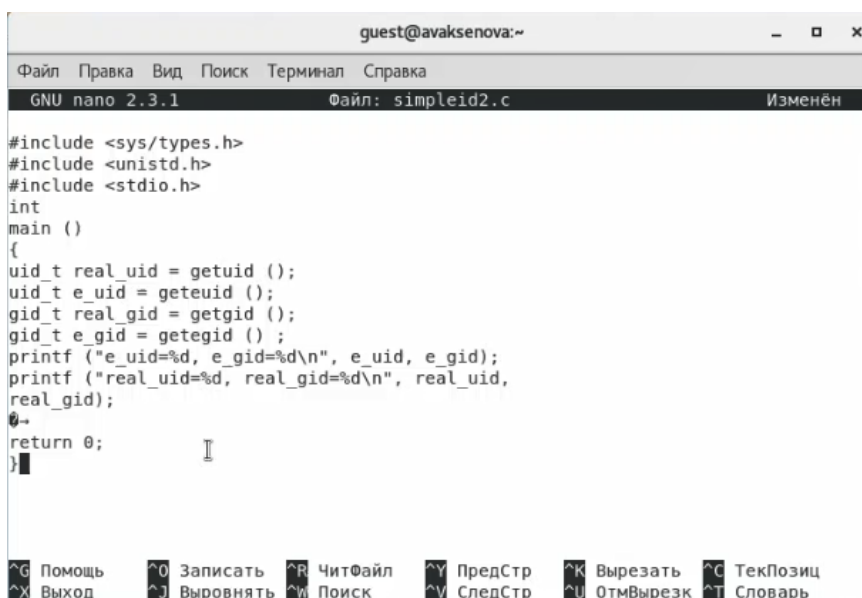
```
[guest@avaksenova ~]$ gcc simpleid.c -o simpleid
[guest@avaksenova ~]$ ls -l
итого 16
drwxrwxrwx. 2 guest guest 19 окт 27 22:21 dir1
-rwxrwxr-x. 1 guest guest 8472 ноя 12 15:49 simpleid
-rw-rw-r--. 1 guest guest 175 ноя 12 15:48 simpleid.c
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Видео
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Документы
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Загрузки
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Изображения
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Музыка
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Общедоступные
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Рабочий стол
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Шаблоны
[guest@avaksenova ~]$
```

Figure 0.2: Запуск simpleid.c

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

Figure 0.3: Сверка simpleid.c и id

2. Усложняем программу и запускаем её. (Рис. 0.4, 0.5).



```

guest@avaksenova:~
Файл Правка Вид Поиск Терминал Справка
GNU nano 2.3.1 Файл: simpleid2.c Изменён

#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;
}

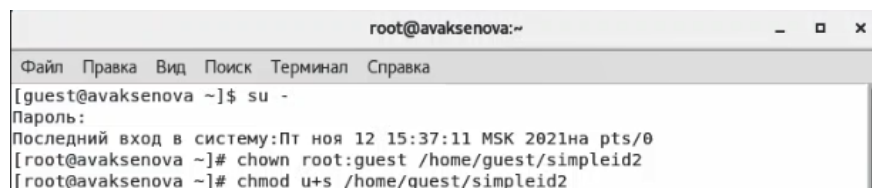
```

Figure 0.4: Листинг simpleid2.c

```
[guest@avaksenova ~]$ gcc simpleid2.c -o simpleid2
[guest@avaksenova ~]$ ls -l
итого 32
drwxrwxrwx. 2 guest guest 19 окт 27 22:21 dir1
-rwxrwxr-x. 1 guest guest 8472 ноя 12 15:49 simpleid
-rwxrwxr-x. 1 guest guest 8576 ноя 12 15:53 simpleid2
-rw-rw-r--. 1 guest guest 313 ноя 12 15:53 simpleid2.c
-rw-rw-r--. 1 guest guest 175 ноя 12 15:48 simpleid.c
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Видео
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Документы
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Загрузки
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Изображения
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Музыка
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Общедоступные
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Рабочий стол
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Шаблоны
[guest@avaksenova ~]$
```

Figure 0.5: Запуск simpleid2.c

3. Из-под суперпользователя меняем владельца и добавляем SetUID бит на файл. Проверяем правильность и запускаем программу еще раз. `euid` возвращает `id` владельца, а `real_uid` возвращает `uid` запускающего пользователя. (Рис. 0.6, 0.7, 0.8).



```
root@avaksenova:~
Файл Правка Вид Поиск Терминал Справка
[guest@avaksenova ~]$ su -
Пароль:
Последний вход в систему: Пт ноя 12 15:37:11 MSK 2021 на pts/0
[root@avaksenova ~]# chown root:guest /home/guest/simpleid2
[root@avaksenova ~]# chmod u+s /home/guest/simpleid2
```

Figure 0.6: Смена владельца и добавление SetUID бита

```
[guest@avaksenova ~]$ ls -l simpleid2
-rwsrwxr-x. 1 root guest 8576 ноя 12 15:53 simpleid2
[guest@avaksenova ~]$
```

Figure 0.7: Запуск simpleid2.c

```
[guest@avaksenova ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@avaksenova ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi
ned_r:unconfined_t:s0-s0:c0.c1023
[guest@avaksenova ~]$
```

Figure 0.8: Сверка simpleid2.c и `id`

4. Теперь добавим на файл SetGID бит с проделаем все то же самое. (Рис. 0.9, 0.10).

```
[root@avaksenova ~]# chmod g+s /home/guest/simpleid2
```

Figure 0.9: Добавление SetGID бита

```
[guest@avaksenova ~]$ ls -l
итого 32
drwxrwxrwx. 2 guest guest 19 окт 27 22:21 dir1
-rwxrwxr-x. 1 guest guest 8472 ноя 12 15:49 simpleid
-rwsrwsr-x. 1 root guest 8576 ноя 12 15:53 simpleid2
-rw-rw-r--. 1 guest guest 313 ноя 12 15:53 simpleid2.c
-rw-rw-r--. 1 guest guest 175 ноя 12 15:48 simpleid.c
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Видео
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Документы
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Загрузки
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Изображения
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Музыка
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Общедоступные
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Рабочий стол
drwxr-xr-x. 2 guest guest 6 окт 1 16:23 Шаблоны
[guest@avaksenova ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@avaksenova ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi
ned r:unconfined t:s0-s0:c0.c1023
```

Figure 0.10: Запуск simpleid2.c

5. Пишем программу readfile.c. (Рис. 0.11).

```
guest@avaksenova:~
Файл Правка Вид Поиск Терминал Справка
GNU nano 2.3.1 Файл: readfile.c Изменён

#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 0.11: Код программы

6. Компилируем программу. (Рис. 0.12).

```
[guest@avaksenova ~]$ nano readfile.c
[guest@avaksenova ~]$ gcc readfile.c -o readfile
[guest@avaksenova ~]$
```

Figure 0.12: Компиляция

7. Меняем владельца у файла readfile.c и запрещаем чтение всем, кроме суперпользователя. Проверяем, что guest не может читать. Меняем владельца у программы readfile и добавляем SetUID бит на неё. (Рис. 0.13, 0.14, 0.15).

```
[root@avaksenova guest]# chown root:guest /home/guest/readfile.c
[root@avaksenova guest]# chmod 700 readfile.c
```

Figure 0.13: Смена владельца readfile.c

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

Figure 0.14: Проверка на cat из-под guest'a

```
[root@avaksenova guest]# chown root:guest /home/guest/readfile
[root@avaksenova guest]# chmod u+s /home/guest/readfile
[root@avaksenova guest]#
```

Figure 0.15: Смена владельца readfile и SetUID

8. Проверяем, может ли программа readfile прочитать файл readfile.c и файл /etc/shadow. Да, может. 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). (Рис. 0.16, 0.17).

```

cat: readfile.c: Отказано в доступе
[guest@avaksenova ~]$ ./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;
}
[guest@avaksenova ~]$ ./readfile /etc/shadow

```

Figure 0.16: Чтение readfile.c программой readfile

```

chrony:!!:18884:::::
unbound:!!:18884:::::
qemu:!!:18884:::::
tss:!!:18884:::::
usbmuxd:!!:18884:::::
geoclue:!!:18884:::::
gluster:!!:18884:::::
gdm:!!:18884:::::
rpcuser:!!:18884:::::
nfsnobody:!!:18884:::::
gnome-initial-setup:!!:18884:::::
sshd:!!:18884:::::
avahi:!!:18884:::::
postfix:!!:18884:::::
ntp:!!:18884:::::
tcpdump:!!:18884:::::
avaksenova:$6$0Ef4s.fzn7vrjuof$Q1Tpo/r9cru0PLfpSoilAxDxZoqV5mLiFH6Gce3x/wqeTWuWR
yM0u/rUdbB9gTq0ryZKgY./goIexehbCTwo2/:0:99999:7:::
vboxadd:!!:18901:::::
guest:$6$5AvQt6Bw$JrKnsKRzARNm9IEAaU4Z0U6t.sqBkf/H/n0geZP9toTH5yk9329gtWLEjCTdjr
nasv3PnmTgqkrYYZ8LfSlNv1:18901:0:99999:7:::
guest2:$6$Jcjy12PA$f7g0FRUsMD0kPYzMBCqRrv/t62EV4z/z/jR90rMKHoh/FD2vBJC0z9TtX5UoK
NeeC2sXWad8lYVoI8Z8MbyE70:18943:0:99999:7:::
[guest@avaksenova ~]$

```

Figure 0.17: Чтение /etc/shadow программой readfile

9. Проверяем Sticky бит. Для этого создаем файл, которому даем rw права для others и пишем туда слово test. Теперь пробуем выполнить дозапись в файл, перезапись файла и его удаление. Всё, кроме удаления, прошло успешно. (Рис. 0.18, 0.19, 0.20, 0.21, 0.22).

```
[guest@avaksenova ~]$ ls -l / | grep tmp
drwxrwxrwt. 15 root root 4096 ноя 12 16:24 tmp
[guest@avaksenova ~]$ cd /tmp
[guest@avaksenova tmp]$ touch file01.txt
[guest@avaksenova tmp]$ echo "test" > /tmp/file01.txt
[guest@avaksenova tmp]$
```

Figure 0.18: Создание файла

```
[guest@avaksenova tmp]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 ноя 12 16:26 /tmp/file01.txt
[guest@avaksenova tmp]$ chmod o+rw /tmp/file01.txt
[guest@avaksenova tmp]$
```

Figure 0.19: Правка прав файла

```
[guest@avaksenova tmp]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 ноя 12 16:26 /tmp/file01.txt
[guest@avaksenova tmp]$
```

Figure 0.20: Проверка прав

```
[guest@avaksenova ~]$ su guest2
Пароль:
[guest2@avaksenova guest]$ cat /tmp/file01.tx
cat: /tmp/file01.tx: Нет такого файла или каталога
[guest2@avaksenova guest]$ cat /tmp/file01.txt
test
[guest2@avaksenova guest]$ echo "test2" >> /tmp/file01.txt
[guest2@avaksenova guest]$ cat /tmp/file01.txt
test
test2
[guest2@avaksenova guest]$ echo "test3" > /tmp/file01.txt
[guest2@avaksenova guest]$ cat /tmp/file01.txt
test3
[guest2@avaksenova guest]$
```

Figure 0.21: Тестирование файла

```
[guest2@avaksenova tmp]$ rm file01.txt
rm: невозможно удалить «file01.txt»: Операция не позволена
[guest2@avaksenova tmp]$
```

Figure 0.22: Попытка удаления файла

10. Повышаем права до суперпользователя и удаляем Sticky-бит с папки /tmp. Повторяем наши тесты. Теперь прошли все команды, включая удаление файла. Таким образом, пользователь, не являющийся владельцем файла, смог его

удалить, так как Sticky-бит не был настроен. Возвращаем Sticky-бит на папку /tmp. (Рис. 0.23, 0.24, 0.25).

```
[root@avaksenova guest]# chmod -t /tmp
[root@avaksenova guest]#
```

Figure 0.23: Удаление Sticky-бита

```
[guest2@avaksenova home]$ ls -l / | grep tmp
drwxrwxrwx. 15 root root 4096 ноя 12 16:28 tmp
[guest2@avaksenova home]$ cat /tmp/file01.txt
test3
[guest2@avaksenova home]$ rm /tmp/file01.txt
rm: невозможно удалить «/tmp/file01.txt»: Нет такого файла или каталога
[guest2@avaksenova home]$ rm /tnp/file01.txt
rm: невозможно удалить «/tnp/file01.txt»: Нет такого файла или каталога
[guest2@avaksenova home]$ rm /tmp/file01.txt
[guest2@avaksenova home]$ ls -l
итого 8
drwx-----. 15 avaksenova avaksenova 4096 окт  1 15:54 avaksenova
drwxrwx---. 16 guest      guest      4096 ноя 12 16:18 guest
drwx-----.  5 guest2     guest2     148 ноя 12 16:28 guest2
[guest2@avaksenova home]$
```

Figure 0.24: Повторное тестирование файла

```
[root@avaksenova guest]# chmod +t /tmp
```

Figure 0.25: Возвращение Sticky-бита



## Выводы

В результате выполнения данной работы были изучены механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Кроме того, получены практические навыки работы в консоли с дополнительными атрибутами, рассмотрена работа механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

## Библиографический список

- [illegible]