

**Лабораторная работа № 5.  
Дискреционное  
разграничение прав в Linux.  
Исследование влияния  
дополнительных атрибутов**

Радикорский Павел Михайлович,  
НФИбд-03-18

13.11.2021

## **Содержание**

<b>Цели и задачи</b>	<b>4</b>
<b>Выполнение</b>	<b>5</b>
<b>Выводы</b>	<b>9</b>

## Список иллюстраций

1.	simpleid.c . . . . .	5
2.	simpleid . . . . .	5
3.	simpleid.c . . . . .	6
4.	simpleid . . . . .	6
5.	superuser . . . . .	6
6.	ls -l . . . . .	7
7.	simpleid . . . . .	7
8.	readfile.c . . . . .	7
9.	readfile, superuser . . . . .	8
10.	guest2 . . . . .	8
11.	guest2 . . . . .	8

## Цели и задачи

**Цель:** 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

## Выполнение

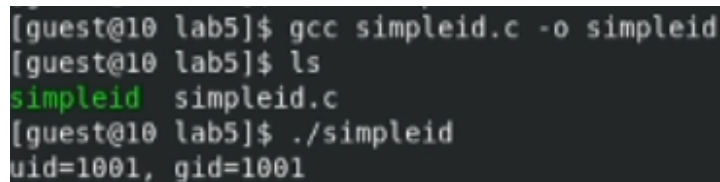
Создали программу simpleid.c со следующим кодом

```
#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: simpleid.c

Скомпилировали и выполнили программу. Результат совпал с командой id



```
[guest@10 lab5]$ gcc simpleid.c -o simpleid
[guest@10 lab5]$ ls
simpleid  simpleid.c
[guest@10 lab5]$ ./simpleid
uid=1001, gid=1001
```

Рис. 2: simpleid

Усложнили программу, добавив вывод действительных идентификаторов

```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int
main ()
{
    uid_t real_uid = getuid ();
    uid_t e_uid = geteuid ();

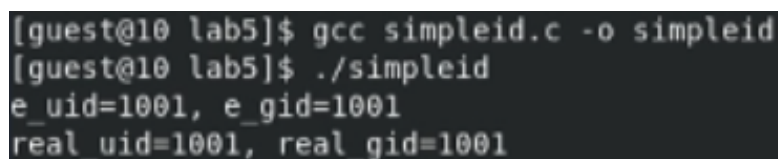
    gid_t real_gid = getgid ();
    gid_t e_gid = getegid ();

    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;
}
```

**Рис. 3:** simpleid.c

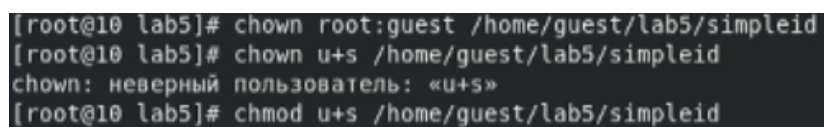
Скомпилировали и выполнили программу



```
[guest@10 lab5]$ gcc simpleid.c -o simpleid
[guest@10 lab5]$ ./simpleid
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
```

**Рис. 4:** simpleid

От имени суперпользователя выполнили команды



```
[root@10 lab5]# chown root:guest /home/guest/lab5/simpleid
[root@10 lab5]# chown u+s /home/guest/lab5/simpleid
chown: неверный пользователь: «u+s»
[root@10 lab5]# chmod u+s /home/guest/lab5/simpleid
```

**Рис. 5:** superuser

Выполнили проверку правильности установки новых атрибутов и смены владельца файла simpleid

```
[root@10 lab5]# ls -l
итого 24
-rwsrwxr-x. 1 root  guest 17648 ноя 13 22:38 simpleid
-rw-rw-r--. 1 guest guest  308 ноя 13 22:38 simpleid.c
```

Рис. 6: ls -l

Запустили simpleid, результат e\_uid изменился. При смене SetGod-бита получили аналогичный результат

```
[root@10 lab5]# ./simpleid
e_uid=0, e_gid=0
real uid=0, real gid=0
[root@10 lab5]# su
[root@10 lab5]# exit
[root@10 lab5]# exit
[guest@10 lab5]$ ./simpleid
e_uid=0, e_gid=1001
real uid=1001, real gid=1001
[guest@10 lab5]$ id
uid=1001(guest) gid=1001(guest) rгруппы=1001(guest) контекст=unconfined_u:unconfi
ned r:unconfined t:s0-s0:c0.c1023
```

Рис. 7: simpleid

Создайте программу readfile.c со следующим кодом

```
#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. 8: readfile.c

Сменили владельца у файла readfile.c и изменили права так, чтобы только супер-пользователь (root) мог прочитать его, а guest не мог

```
guest@10 lab5]$ gcc readfile.c -o readfile
guest@10 lab5]$ chmod 000 readfile.c
guest@10 lab5]$ chown root:root readfile.c
chown: изменение владельца 'readfile.c': Операция не позволена
guest@10 lab5]$ sudo chown root:root readfile.c
[sudo] пароль для guest:
guest is not in the sudoers file. This incident will be reported.
guest@10 lab5]$ su
пароль:
root@10 lab5)# chown root:root readfile.c
```

**Рис. 9:** readfile, superuser

Попробовали прочитать, записать и удалить файл от имени guest2, удалить не получилось

```
[guest2@10 tmp]$ cat file01.txt
test
[guest2@10 tmp]$ echo "test2" > file01.txt
[guest2@10 tmp]$ cat file01.txt
test2
[guest2@10 tmp]$ echo "test3" > file01.txt
[guest2@10 tmp]$ cat file01.txt
test3
[guest2@10 tmp]$ rm file01.txt
rm: невозможно удалить 'file01.txt': Операция не позволена
```

**Рис. 10:** guest2

Сняли атрибут t, выполнили те же команды, в этот раз удаление произошло успешно

```
[guest2@10 tmp]$ echo "test3" > file01.txt
[guest2@10 tmp]$ cat file01.txt
test3
[guest2@10 tmp]$ rm file01.txt
```

**Рис. 11:** guest2



## **Выводы**

В результате выполнения работы я изучил механизмы изменения идентификаторов, применения SetUID- и Sticky-битов