

# **Лабораторная работа № 5**

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

Алади Принц Чисом

# Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Исследование Sticky-бита.	12
4	ВАЖНОЕ ПРИМЕЧАНИЕ	15
5	Выводы	16
	Список литературы	17

## Список иллюстраций

2.1	Компиляция первой программы . . . . .	6
2.2	Запуск первой программы . . . . .	6
2.3	Вторая программа . . . . .	7
2.4	Запуск второй программы . . . . .	7
2.5	Изменение прав для root . . . . .	7
2.6	Проверка работы для root . . . . .	8
2.7	Установка SetUID-бита . . . . .	8
2.8	Установка SetUID-бита . . . . .	8
2.9	Компиляция readfile . . . . .	8
2.10	Проверка на root и guest пользователях . . . . .	9
2.11	Смена владельца . . . . .	9
2.12	Запуск с guest . . . . .	10
2.13	Запуск с root . . . . .	11
3.1	Проверка наличия атрибута . . . . .	12
3.2	Выдача прав для файла . . . . .	12
3.3	Выдача прав для файла . . . . .	12
3.4	Проверка от второго пользователя . . . . .	13
3.5	Проверка без атрибута . . . . .	13
3.6	Возвращение атрибута . . . . .	13
3.7	Возвращение атрибута . . . . .	14

## **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 Выполнение лабораторной работы

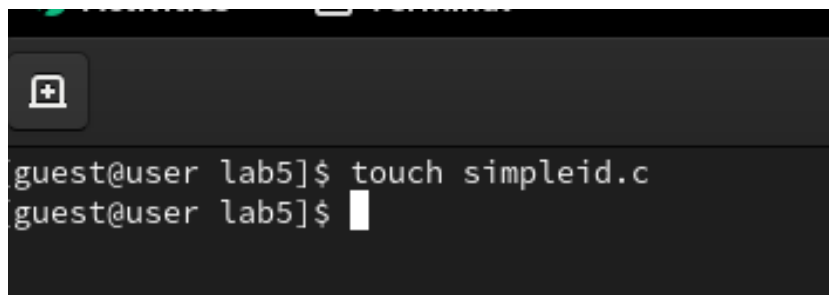
1) Я создал файл “simpleid.c” и внёс в него программу.



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

Первая программа

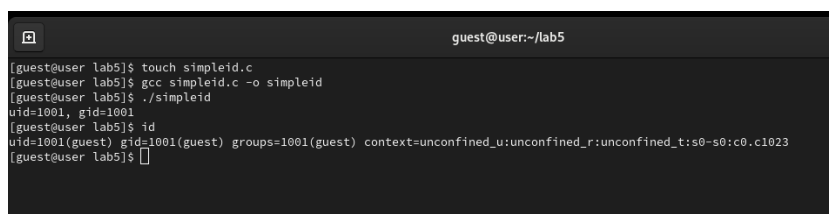
2) Скомпилировал программу и убедился, что файл создан правильно.



```
guest@user lab5]$ touch simpleid.c
guest@user lab5]$
```

Рис. 2.1: Компиляция первой программы

3) Запустил программу и посмотрел, как она работает. Затем прописал команду “id”, чтобы сравнить данные. Все данные сходятся.



```
guest@user:~/lab5
[guest@user lab5]$ touch simpleid.c
[guest@user lab5]$ gcc simpleid.c -o simpleid
[guest@user lab5]$ ./simpleid
uid=1001, gid=1001
[guest@user lab5]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@user lab5]$
```

Рис. 2.2: Запуск первой программы

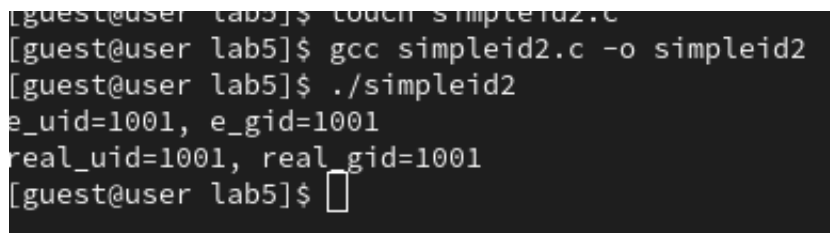
- 4) Создал второй файл и назвал его “simpleid2.c”. Усложнил первую программу и внёс ее в файл.



```
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     gid_t real_gid = getgid ();
10    gid_t e_gid = getegid ();
11    printf ("e_uid=%d, e_gid=%d\n", e_uid, e_gid);
12    printf ("real_uid=%d, real_gid=%d\n", real_uid,
13    , real_gid);
14    return 0;
15 }
```

Рис. 2.3: Вторая программа

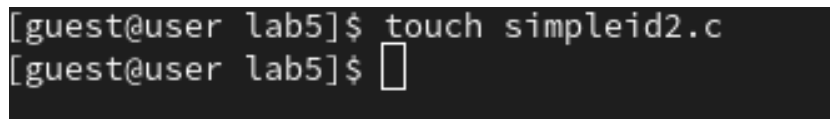
- 5) Скомпилировал и посмотрел вторую программу. Проверил как она работает.



```
[guest@user lab5]$ touch simpleid2.c
[guest@user lab5]$ gcc simpleid2.c -o simpleid2
[guest@user lab5]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@user lab5]$
```

Рис. 2.4: Запуск второй программы

- 6) От имени суперпользователя я выполнил команды и временно повысил свои права. Команды сменили пользователя файла на root и установили SetUID-бит. Я запустил файл от имени root-пользователя и проверил сходство с командой “id”.



```
[guest@user lab5]$ touch simpleid2.c
[guest@user lab5]$
```

Рис. 2.5: Изменение прав для root

```

root@user lab5]# chown root:guest /home/guest/lab5/simpleid2
root@user lab5]# chmod u+s /home/guest/lab5/simpleid2
root@user lab5]#

```

Рис. 2.6: Проверка работы для root

```

exit
[guest@user lab5]$ ls -l simpleid2
-rwsr-xr-x. 1 root guest 26064 Apr 13 01:06 simpleid2
[guest@user lab5]$

```

Рис. 2.7: Установка SetUID-бита

```

-rwsr-xr-x. 1 root guest 26064 Apr 13 01:06 simpleid2
[guest@user lab5]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@user lab5]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@user lab5]$

```

Рис. 2.8: Установка SetUID-бита

7) Я создал файл “readfile.c”. Внёс туда программу.

Программа readfile

8) Скомпилировал программу readfile.

```

[guest@user lab5]$ su
Password:
[root@user lab5]# chmod u+s /home/guest/lab5/readfile
[root@user lab5]# chmod 700 readfile
[root@user lab5]# chown root:guest readfile
[root@user lab5]# chown -r readfile.c
chown: invalid option -- 'r'
Try 'chown --help' for more information.
[root@user lab5]# chmod -r readfile.c
[root@user lab5]# chmod u+s readfile
[root@user lab5]# exit
exit

```

Рис. 2.9: Компиляция readfile

9) Я выдал программе “readfile” права так, чтобы root пользователь мог прочитать файл, а простой пользователь нет.



```
[guest@user lab5]$ cat readfile.c
cat: readfile.c: Permission denied
[guest@user lab5]$ ./readfile readfile.c
bash: ./readfile: Permission denied
[guest@user lab5]$
```

Рис. 2.10: Проверка на root и guest пользователях

10) Я сменил владельца программы “readfile” на root-пользователя.

```
[guest@user lab5]$ ./readfile readfile.c
bash: ./readfile: Permission denied
[guest@user lab5]$ su
Password:
[root@user lab5]# 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. 2.11: Смена владельца

11) Попытался запустить программу и прочитав два файла с простого пользователя, но программа выдала ошибку. А если запускать с аккаунта root, то программа запускается нормально и работает. Связано это с тем, что владельцем программы является root-пользователь, а у других пользователей нет доступа и прав на использование программы.

```

[root@user lab5]# ./readfile /etc/shadow
root:$6$ru5Er4001MIsCD1$GLeJ13Hv4CGFDKHH5LnZyzUngetY8MXV/hOFT2h5sYqTDROo89QPyrTo8AwhYQdFKmFEXgP20tg9nsInL./fj...:0:99999:7:::
bin::19469:0:99999:7:::
daemon::19469:0:99999:7:::
adm::19469:0:99999:7:::
lp::19469:0:99999:7:::
sync::19469:0:99999:7:::
shutdown::19469:0:99999:7:::
halt::19469:0:99999:7:::
mail::19469:0:99999:7:::
operator::19469:0:99999:7:::
games::19469:0:99999:7:::
ftp::19469:0:99999:7:::
nobody::19469:0:99999:7:::
systemd-coredump::!19770:::
dbus::!19770:::
polkitd::!19770:::
avahi::!19770:::
rtkit::!19770:::
pipewire::!19770:::
sssd::!19770:::
libstoragemgmt::!19770:::
systemd-oom::!19770:::
tas::!19770:::
geoclue::!19770:::
cockpit-ws::!19770:::
cockpit-wsinstance::!19770:::
flatpak::!19770:::
colord::!19770:::
cups::!19770:::
setroubleshoot::!19770:::
gdm::!19770:::
pesign::!19770:::
gnome-initial-setup::!19770:::
sshd::!19770:::
chrony::!19770:::
dnsmasq::!19770:::
tcpdump::!19770:::
aladipcc::!6$5j2deIoafQI0e81Zn$RvmujyXVZSZGo7Rlp.KSLgn9SAFjz44BY6MFubg7yOurwyecTeo4JnHWIBLqTnuHb/YMcflZTuRDQ.JFpMTER0::0:99999:7:::
vboxadd::!19770:::
guest:$6$ovMP0o54N4Mcml1$37L3G8S1lVEKpRgxOY7Pk5tp1C/PpuXpMF0DNBfsEbgvhyby/. /U7qOcw3677r7mU4CL6gsRWmrRog9crLzDh1:19784:0:99999:7:::
guest2:$6$1Ww1uWc39Bj7EpjP$1vmvzkWN37cmCaTV3po7Dn508MSBOTPLB1AkbbatLxp67mSD2e85tVzF9dgVtGgs8Sw3m4y4nyIAb3hyr6UVs.:19797:0:99999:7:::

```

Рис. 2.12: Запуск с guest



### 3 Исследование Sticky-бита.

- 1) Я выяснил, установлен ли атрибут Sticky (t) на директории “/tmp”. Атрибут установлен.

```
[guest@user lab5]$ ls -l / | grep tmp
drwxrwxrwt. 17 root root 4096 Apr 13 01:57 tmp
```

Рис. 3.1: Проверка наличия атрибута

- 2) От пользователя “guest” я создал файл “file01.txt” в директории “/tmp”. Вписал в файл слово “test”. И дал права на чтение и запись для категории “все остальные (o)”.

```
[guest@user lab5]$ echo "test" > /tmp/file01.txt
```

Рис. 3.2: Выдача прав для файла

```
[guest@user lab5]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 Apr 13 01:59 /tmp/file01.txt
[guest@user lab5]$ chmod o+rw /tmp/file01.txt
[guest@user lab5]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 Apr 13 01:59 /tmp/file01.txt
```

Рис. 3.3: Выдача прав для файла

- 3) От пользователя “guest2”, который не является владельцем, я попробовал прочитать файл. Я могу прочитать файл. Но не могу дописывать содержимое, вписывать новое или удалять этот файл.

```

bash: guest: command not found...
[guest@user lab5]$ su guest2
Password:
[guest2@user lab5]$ cat /tmp/file01.txt
test

```

Рис. 3.4: Проверка от второго пользователя

- 4) я отключил атрибут “t” у директории “/tmp”. Попробовал повторить все предыдущие действия. Я так же не смог вписать в файл данные или дописать их. Но смог прочитать файл и удалить его.

```

test
[guest2@user lab5]$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@user lab5]$ cat /tmp/file01.txt
test
[guest2@user lab5]$ █

```

Рис. 3.5: Проверка без атрибута

- 5) Чтобы в дальнейшем у меня не было проблем в работе с директорией “/tmp” я вернул атрибут на директорию, используя суперпользователя.

```

[root@user lab5]# exit
exit
[guest2@user lab5]$ ls -l / | grep tmp
drwxrwxrwx. 17 root root 4096 Apr 13 02:11 tmp
[guest2@user lab5]$ cat /tmp/file01.txt
test
[guest2@user lab5]$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@user lab5]$ █

```

Рис. 3.6: Возвращение атрибута

```
[guest2@user lab5]$ su
Password:
[root@user lab5]# chmod +t /tmp
[root@user lab5]# exit
exit
[guest2@user lab5]$
```

Рис. 3.7: Возвращение атрибута

## 4 ВАЖНОЕ ПРИМЕЧАНИЕ

По итогам лабораторной работы я понял, что Sticky-бит создан для защиты файла от удаления. Даже не смотря на то, что я дал права на запись и чтение файлов для категории “все остальные”, я не смог вписать в файл данные с пользователя “guest2”. А не смог я это сделать, так как этот аккаунт у меня находится в группе с “guest”. То есть я не дал права на вписывание для категории “группа”, но дал права для категории “все остальные”. Из-за этого Sticky-бит не влиял на возможность записи, а влиял только на возможность удаления. А изменять файл я не мог, так как мой аккаунт находился не в той группе. Если бы я использовал другой аккаунт, который не находится в группе, результаты бы были другие.

## 5 Выводы

Я изучил механизмы изменения идентификатора, применил SetUID-бит и Sticky-бит. Получил практические навыки работы в консоли с дополнительными атрибутами. Рассмотрел работы механизма смены идентификатора процессов пользователя, а так же влияние бита Sticky на запись и удаление файлов.



## **Список литературы**