

# **Лабораторная работа №5**

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

**Ильинский Арсений Александрович**

# Содержание

<b>Цель работы</b>	<b>5</b>
<b>Задание</b>	<b>6</b>
<b>Теоретическое введение</b>	<b>7</b>
<b>Выполнение лабораторной работы</b>	<b>8</b>
Подготовка к работе . . . . .	8
Создание программы . . . . .	9
Работа с e SetUID-битом и SetGID-битом . . . . .	11
Исследование Sticky-бита . . . . .	15
<b>Выводы</b>	<b>19</b>
<b>Список литературы</b>	<b>20</b>

# Список иллюстраций

1	Компилятор gcc . . . . .	8
2	Отключение SELinux . . . . .	9
3	Отключение SELinux . . . . .	9
4	Компиляция и выполнение simpleid.c . . . . .	10
5	Результат команды id . . . . .	10
6	Компиляция и выполнение simpleid2 . . . . .	11
7	Изменение владельца и прав на файл simpleid2 . . . . .	11
8	Атрибуты и владелец файла simpleid2 . . . . .	11
9	Выполнение simpleid2 и id . . . . .	12
10	Программа readfile.c . . . . .	13
11	Изменение владельца и прав на файл readfile.c . . . . .	13
12	Установка UID бита для readfile.c . . . . .	13
13	Выполнение программы для файла readfile.c . . . . .	14
14	Выполнение программы для файла /etc/shadow . . . . .	14
15	Работа со Sticky битом . . . . .	16
16	Работа с файлом без Sticky бита (1/2) . . . . .	17
17	Работа с файлом без Sticky бита (2/2) . . . . .	17
18	Установление атрибута t . . . . .	18

## **Список таблиц**

## 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

Выполнить задания из лабораторной работы и проанализировать полученные результаты.

# Теоретическое введение

Для выполнения данной лабораторной нет специальной теории.

# Выполнение лабораторной работы

## Подготовка к работе

Проверю, установлен ли у меня компилятор gcc командой `gcc -v`:

```
[arilinskiy@arilinskiy ~]$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/x86_64-redhat-linux/11/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none
OFFLOAD_TARGET_DEFAULT=1
Target: x86_64-redhat-linux
Configured with: ../configure --enable-bootstrap --enable-host-pie --enable-host
-bind-now --enable-languages=c,c++,fortran,lto --prefix=/usr --mandir=/usr/share
/man --infodir=/usr/share/info --with-bugurl=https://bugs.rockylinux.org/ --enab
le-shared --enable-threads=posix --enable-checking=release --enable-multilib --w
ith-system-zlib --enable-__cxa_atexit --disable-libunwind-exceptions --enable-gn
u-unique-object --enable-linker-build-id --with-gcc-major-version-only --with-li
nker-hash-style=gnu --enable-plugin --enable-initfini-array --without-isl --enab
le-offload-targets=nvptx-none --without-cuda-driver --enable-gnu-indirect-functi
on --enable-cet --with-tune=generic --with-arch 64=x86-64-v2 --with-arch 32=x86-
64 --build=x86_64-redhat-linux --with-build-config=bootstrap-lto --enable-link-s
erialization=1
Thread model: posix
Supported LTO compression algorithms: zlib zstd
gcc version 11.2.1 20220127 (Red Hat 11.2.1-9) (GCC)
[arilinskiy@arilinskiy ~]$
```

Рис. 1: Компилятор gcc

У меня он уже установлен.

Установил `setenforce 0` и проверил, что данная команда выполнилась:



```
[guest@arilinskiy ~]$ su
Password:
[root@arilinskiy guest]# setenforce 0
[root@arilinskiy guest]# getenforce 0
Permissive
[root@arilinskiy guest]#
```

Рис. 2: Отключение SELinux

## Создание программы

Вошел в систему от имени пользователя guest и создал программу simpleid.c

```
[guest@arilinskiy ~]$ touch simpleid.c
[guest@arilinskiy ~]$ ls
Desktop  Documents  Music      Public      Templates
dir1     Downloads  Pictures   simpleid.c  Videos
[guest@arilinskiy ~]$ vim simpleid.c
```

Рис. 3: Отключение SELinux

```
#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;
}
```

Скомпилирую программу командой `gcc simpleid.c -o simpleid` и запустил ее

```
[guest@arilinskiy ~]$ gcc simpleid.c -o simpleid
[guest@arilinskiy ~]$ ./simpleid
uid=1001, gid=1001
```

Рис. 4: Компиляция и выполнение simpleid.c

Выполню системную программу `id` командой `id`. Как видно результат совпадает:

```
[guest@arilinskiy ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 5: Результат команды `id`

Усложню 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;
}
```

Скомпилирую в новый файл `simpleid2`

```
[guest@arilinskiy ~]$ gcc simpleid.c -o simpleid2
[guest@arilinskiy ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
```

Рис. 6: Компиляция и выполнение simpleid2

## Работа с e SetUID-битом и SetGID-битом

От имени суперпользователя выполню команды:

1. `chown root:guest /home/guest/simpleid2`
2. `chmod u+s /home/guest/simpleid2`

```
[guest@arilinskiy ~]$ su
Password:
[root@arilinskiy guest]# chown root:guest /home/guest/simpleid2
[root@arilinskiy guest]# chmod u+s /home/guest/simpleid2
```

Рис. 7: Изменение владельца и прав на файл simpleid2

Пояснение:

- Команда `chown root:guest /home/guest/simpleid2` меняет владельца файла.
- Команда `chmod u+s /home/guest/simpleid2` меняет права доступа к файлу.

Проверю правильность установки новых атрибутов и смены владельца файла simpleid2 командой: `ls -l simpleid2`

```
[guest@arilinskiy ~]$ ls -l simpleid2
-rwsrwxr-x. 1 root guest 26008 Oct  6 14:02 simpleid2
```

Рис. 8: Атрибуты и владелец файла simpleid2

Запущу simpleid2 и id, командами соответственно: `./simpleid2` и `id`

```
[guest@arilinskiy ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@arilinskiy ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 9: Выполнение simpleid2 и id

Создам программы 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@arilinskiy ~]$ touch readfile.c
[guest@arilinskiy ~]$ vim readfile.c
```

Рис. 10: Программа readfile.c

Скомпилирую её командой: `gcc readfile.c -o readfile`, а затем сменю владельца у файла `readfile.c` и изменю права так, чтобы только суперпользователь (root) мог прочитать его, а `guest` не мог:

```
[guest@arilinskiy ~]$ gcc readfile.c -o readfile
[guest@arilinskiy ~]$ su
Password:
[root@arilinskiy guest]# chown root:guest /home/guest/readfile.c
[root@arilinskiy guest]# chmod 700 /home/guest/readfile.c
[root@arilinskiy guest]#
exit
[guest@arilinskiy ~]$ ls -l readfile.c
-rwx-----. 1 root guest 415 Oct  6 14:14 readfile.c
[guest@arilinskiy ~]$ cat readfile.c
cat: readfile.c: Permission denied
```

Рис. 11: Изменение владельца и прав на файл `readfile.c`

Как видим, пользователь `guest` не может прочитать файл `readfile.c`. Сменю у программы `readfile` владельца и установлю SetUID-бит:

```
[guest@arilinskiy ~]$ su
Password:
[root@arilinskiy guest]# chown root:guest /home/guest/readfile
[root@arilinskiy guest]# chmod u+s /home/guest/readfile
```

Рис. 12: Установка UID бита для `readfile.c`

Проверю, может ли программа `readfile` прочитать файл `readfile.c`:

```
[guest@arilinskiy ~]$ ./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;
}
```

Рис. 13: Выполнение программы для файла readfile.c

```
[guest@arilinskiy ~]$ ./readfile /etc/shadow
root:$6$gVF5yHxGxqk9g0oX$iVHgsVLtBcp/ewkZGke73j4AmCU5/X05S0VimcXIz9sgfGJ347wqXdU
a2jwUqru6Tiiq0PK0zHZFaijTmC7ET.:0:99999:7:::
bin:!:19123:0:99999:7:::
daemon:!:19123:0:99999:7:::
adm:!:19123:0:99999:7:::
lp:!:19123:0:99999:7:::
sync:!:19123:0:99999:7:::
shutdown:!:19123:0:99999:7:::
halt:!:19123:0:99999:7:::
mail:!:19123:0:99999:7:::
operator:!:19123:0:99999:7:::
games:!:19123:0:99999:7:::
ftp:!:19123:0:99999:7:::
nobody:!:19123:0:99999:7:::
systemd-coredump:!!:19244:::::::
dbus:!!:19244:::::::
polkitd:!!:19244:::::::
rtkit:!!:19244:::::::
```

Рис. 14: Выполнение программы для файла /etc/shadow

Поскольку у программы установлен SetUID-бит, то ей временно предоставляются права владельца файла (суперпользователя). Поэтому программа может прочитать файл с правами доступа только для владельца суперпользователя

# Исследование Sticky-бита

Прделаю ряд действий:

1. Выясню, установлен ли атрибут Sticky на директории /tmp, для чего выполню команду:

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

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

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

3. Просмотрю атрибуты у только что созданного файла и разрешу чтение и запись для категории пользователей «все остальные»:

```
ls -l /tmp/file01.txt
```

```
chmod o+rw /tmp/file01.txt
```

```
ls -l /tmp/file01.txt
```

4. От пользователя guest2 (не являющегося владельцем) попробую прочитат файл /tmp/file01.txt:

```
*cat /tmp/file01.**txt*
```

5. От пользователя guest2 попробую дозаписать в файл /tmp/file01.txt слово test2 командой:

```
echo "test2" » /tmp/file01.txt.
```

Мне удалось выполнить операцию.

6. Проверю содержимое файла командой:

```
cat /tmp/file01.txt
```

7. От пользователя guest2 попробую записать в файл /tmp/file01.txt слово test3, стеревав при этом всю имеющуюся в файле информацию командой:

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

Мне удалось выполнить операцию.

8. Проверю содержимое файла командой:

```
cat /tmp/file01.txt
```

9. От пользователя guest2 попробую удалить файл /tmp/file01.txt командой:

```
rm /tmp/file01.txt
```

Мне не удалось удалить файл

```
[guest@arilinskiy ~]$ ls -l / | grep tmp
drwxrwxrwt. 16 root root 4096 Oct  6 14:20 tmp
[guest@arilinskiy ~]$ echo "test" > /tmp/file01.txt
[guest@arilinskiy ~]$ cat /tmp/file01.txt
test
[guest@arilinskiy ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 Oct  6 14:24 /tmp/file01.txt
[guest@arilinskiy ~]$ chmod o+rw /tmp/file01.txt
[guest@arilinskiy ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 Oct  6 14:24 /tmp/file01.txt
[guest@arilinskiy ~]$ su guest2
Password:
[guest2@arilinskiy guest]$ cat /tmp/file01.txt
test
[guest2@arilinskiy guest]$ echo "test2" >> /tmp/file01.txt
[guest2@arilinskiy guest]$ cat /tmp/file01.txt
test
test2
[guest2@arilinskiy guest]$ echo "test3" > /tmp/file01.txt
[guest2@arilinskiy guest]$ cat /tmp/file01.txt
test3
[guest2@arilinskiy guest]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': Operation not permitted
[guest2@arilinskiy guest]$
```

Рис. 15: Работа со Sticky битом

**Повышу свои права до суперпользователя и сниму Sticky-бит, после чего повторю сделанные ранее шаги:**



1. Повышу свои права до суперпользователя следующей командой *su* и выполню после этого команду, снимающую атрибут *t* (Sticky-бит) с директории */tmp*:

```
chmod -t /tmp
```

2. От пользователя *guest2* проверил, что атрибута *t* у директории */tmp* нет:

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

3. Повторю предыдущие шаги.

```
[guest2@arilinskiy guest]$ su
Password:
[root@arilinskiy guest]# chmod -t /tmp
```

Рис. 16: Работа с файлом без Sticky бита (1/2)

```
[guest2@arilinskiy guest]$ ls -l / | grep tmp
drwxrwxrwx. 16 root root 4096 Oct  6 14:29 tmp
[guest2@arilinskiy guest]$ cat /tmp/file01.txt
test3
[guest2@arilinskiy guest]$ echo "test2" >> /tmp/file01.txt
[guest2@arilinskiy guest]$ cat /tmp/file01.txt
test3
test2
[guest2@arilinskiy guest]$ echo "test3" > /tmp/file01.txt
[guest2@arilinskiy guest]$ cat /tmp/file01.txt
test3
[guest2@arilinskiy guest]$ rm /tmp/file01.txt
[guest2@arilinskiy guest]$ ls /tmp
dbus-NLvZQkBZqo
dbus-QFSLYBR8sf
dbus-rbwIPQWEme
systemd-private-017df871978e423a82ee6ca025d81135-chrond.service-aAgCqB
systemd-private-017df871978e423a82ee6ca025d81135-colord.service-ggetFK
systemd-private-017df871978e423a82ee6ca025d81135-dbus-broker.service-oqrQ9M
systemd-private-017df871978e423a82ee6ca025d81135-fwupd.service-0w4xdc
systemd-private-017df871978e423a82ee6ca025d81135-ModemManager.service-plAcsf
systemd-private-017df871978e423a82ee6ca025d81135-power-profiles-daemon.service-8
UrQWX
systemd-private-017df871978e423a82ee6ca025d81135-rtkit-daemon.service-MhmQI1
systemd-private-017df871978e423a82ee6ca025d81135-switcheroo-control.service-Gz3Y
Uy
systemd-private-017df871978e423a82ee6ca025d81135-systemd-logind.service-BZIHnI
systemd-private-017df871978e423a82ee6ca025d81135-upower.service-iuE72k
```

Рис. 17: Работа с файлом без Sticky бита (2/2)

Мне удалось удалить файл от имени пользователя, не являющегося его владельцем. Это связано с тем, что Sticky-bit позволяет защищать файлы от случайного удаления, когда несколько пользователей имеют права на запись в один и тот же каталог. Если у файла атрибут `t` стоит, значит пользователь может удалить файл, только если он является пользователем-владельцем файла или каталога, в котором содержится файл. Если же этот атрибут не установлен, то удалить файл могут все пользователи, которым позволено удалять файлы из каталога.

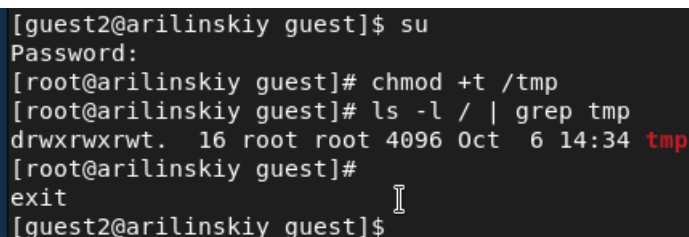
**Повышу свои права до суперпользователя и верну Sticky-бит:**

1. Верну атрибут `t` на директорию `/tmp`:

*su*

*chmod +t /tmp*

*exit*



```
[guest2@arilinskiy guest]$ su
Password:
[root@arilinskiy guest]# chmod +t /tmp
[root@arilinskiy guest]# ls -l / | grep tmp
drwxrwxrwt. 16 root root 4096 Oct  6 14:34 tmp
[root@arilinskiy guest]#
exit
[guest2@arilinskiy guest]$
```

Рис. 18: Установление атрибута `t`

## Выводы

Благодаря данной лабораторной работе я изучил механизмы изменения идентификаторов, применения SetUID-, SetGID- и Sticky-битов. Рассмотрел работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

## Список литературы

- Кулябов Д.С., Королькова А.В., Геворкян М.Н *Лабораторная работа №5*