

# **Отчёт по лабораторной работе №5**

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

Ким Реачна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>5</b>
2.1	Подготовка . . . . .	5
2.2	Изучение механики SetUID . . . . .	6
2.3	Исследование Sticky-бита . . . . .	10
<b>3</b>	<b>Выводы</b>	<b>13</b>
	<b>Список литературы</b>	<b>14</b>

## Список иллюстраций

2.1	Подготовка к работе . . . . .	5
2.2	Программа simpleid . . . . .	6
2.3	Результат программы simpleid . . . . .	6
2.4	Программа simpleid2 . . . . .	7
2.5	Результат программы simpleid2 . . . . .	8
2.6	Программа readfile . . . . .	8
2.7	Результат программы readfile . . . . .	9
2.8	Исследование Sticky-бита . . . . .	12

# 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

### 2.1 Подготовка

1. Для выполнения части заданий требуются средства разработки приложений. Проверили наличие установленного компилятора gcc командой `gcc -v`: компилятор обнаружен.
2. Чтобы система защиты SELinux не мешала выполнению заданий работы, отключили систему запретов до очередной перезагрузки системы командой `setenforce 0`:
3. Команда `getenforce` вывела `Permissive`:

```
[guest@kreachna ~]$ su
Password:
[root@kreachna guest]# 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-n
ow --enable-languages=c,c++,fortran,lto --prefix=/usr --mandir=/usr/share/man --infodir
=/usr/share/info --with-bugurl=https://bugs.rockylinux.org/ --enable-shared --enable-th
reads=posix --enable-checking=release --with-system-zlib --enable-__cxa_atexit --disabl
e-libunwind-exceptions --enable-gnu-unique-object --enable-linker-build-id --with-gcc-m
ajor-version-only --enable-plugin --enable-initfini-array --without-isl --enable-multil
ib --with-linker-hash-style=gnu --enable-offload-targets=nvptx-none --without-cuda-driv
er --enable-gnu-indirect-function --enable-cet --with-tune=generic --with-arch_64=x86-6
4-v2 --with-arch_32=x86-64 --build=x86_64-redhat-linux --with-build-config=bootstrap-lt
o --enable-link-serialization=1
Thread model: posix
Supported LTO compression algorithms: zlib zstd
gcc version 11.3.1 20221121 (Red Hat 11.3.1-4) (GCC)
[root@kreachna guest]# setenforce 0
[root@kreachna guest]# getenforce
Permissive
[root@kreachna guest]# exit
exit
[guest@kreachna ~]$
```

Рис. 2.1: Подготовка к работе

## 2.2 Изучение механики SetUID

1. Вошли в систему от имени пользователя guest.
2. Написали программу simpleid.c.



```
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 }
12
```

Рис. 2.2: Программа simpleid

3. Скомпилировали программу и убедились, что файл программы создан: `gcc simpleid.c -o simpleid`
4. Выполнили программу simpleid командой `./simpleid`
5. Выполнили системную программу `id` с помощью команды `id`. `uid` и `gid` совпадает в обеих программах



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

Рис. 2.3: Результат программы simpleid

6. Усложнили программу, добавив вывод действительных идентификаторов.



```
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, real_gid);
13    return 0;
14 }
```

Рис. 2.4: Программа simpleid2

7. Скомпилировали и запустили simpleid2.c:

```
gcc simpleid2.c -o simpleid2
./simpleid2
```

8. От имени суперпользователя выполнили команды:

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

9. Использовали su для повышения прав до суперпользователя

10. Выполнили проверку правильности установки новых атрибутов и смены владельца файла simpleid2:

```
ls -l simpleid2
```

11. Запустили simpleid2 и id:

./simpleid2

id

Результат выполнения программ теперь немного отличается

12. Проделали тоже самое относительно SetGID-бита.

```
[guest@kreachna lab5]$ touch simpleid2.c
[guest@kreachna lab5]$ gcc simpleid2.c -o simpleid2
[guest@kreachna lab5]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@kreachna lab5]$ su
Password:
[root@kreachna lab5]# chown root:guest /home/guest/simpleid2
chown: cannot access '/home/guest/simpleid2': No such file or directory
[root@kreachna lab5]# chown root:guest simpleid2
[root@kreachna lab5]# chmod u+s simpleid2
[root@kreachna lab5]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@kreachna lab5]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@kreachna lab5]# chmod g+s simpleid2
[root@kreachna lab5]# ./simpleid2
e_uid=0, e_gid=1001
real_uid=0, real_gid=0
[root@kreachna lab5]# exit
exit
```

Рис. 2.5: Результат программы simpleid2

13. Написали программу readfile.c



```
1 #include <fcntl.h>
2 #include <stdio.h>
3 #include <sys/stat.h>
4 #include <sys/types.h>
5 #include <unistd.h>
6 int
7 main (int argc, char* argv[])
8 {
9     unsigned char buffer[16];
10    size_t bytes_read;
11    int i;
12    int fd = open (argv[1], O_RDONLY);
13    do
14    {
15        bytes_read = read (fd, buffer, sizeof (buffer));
16        for (i = 0; i < bytes_read; ++i) printf("%c", buffer[i]);
17    }
18    while (bytes_read == sizeof (buffer));
19    close (fd);
20    return 0;
21 }
```

Рис. 2.6: Программа readfile

14. Откомпилировали её.



```
gcc readfile.c -o readfile
```

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

```
chown root:guest /home/guest/readfile.c
```

```
chmod 700 /home/guest/readfile.c
```

16. Проверили, что пользователь guest не может прочитать файл readfile.c.
17. Сменили у программы readfile владельца и установили SetU'D-бит.
18. Проверили, может ли программа readfile прочитать файл readfile.c
19. Проверили, может ли программа readfile прочитать файл /etc/shadow

```
[guest@kreachna lab5]$ touch readfile.c
[guest@kreachna lab5]$ gcc readfile.c
[guest@kreachna lab5]$ gcc readfile.c -o readfile
[guest@kreachna lab5]$ su
Password:
[root@kreachna lab5]# chown root:root readfile
[root@kreachna lab5]# chmod -rwx readfile.c
[root@kreachna lab5]# chmod u+s readfile
[root@kreachna lab5]# exit
exit
[guest@kreachna lab5]$ cat readfile.c
cat: readfile.c: Permission denied
[guest@kreachna 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;
}
[guest@kreachna lab5]$ ./readfile /etc/shadow
root:$6$QuGLvjQbEW.9s2px$Lbt00B0Yx6xUpFz1zp3uCqIYG6cUW/CN1cfZex.PI/FzEHLFrIAobt0Jh7ZkWKi0BZq0j5I3l5KzCQ15.X1::0:999
99:7:::
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:::
```

Рис. 2.7: Результат программы readfile

## 2.3 Исследование Sticky-бита

1. Выяснили, установлен ли атрибут Sticky на директории /tmp:

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

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

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

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

```
ls -l /tmp/file01.txt  
chmod o+rw /tmp/file01.txt  
ls -l /tmp/file01.txt
```

Первоначально все группы имели право на чтение, а запись могли осуществлять все, кроме «остальных пользователей».

4. От пользователя (не являющегося владельцем) попробовали прочитать файл /file01.txt:

```
cat /file01.txt
```

5. От пользователя попробовали дозаписать в файл /file01.txt слово test3 командой:

```
echo "test2" >> /file01.txt
```

6. Проверили содержимое файла командой:

```
cat /file01.txt
```

В файле теперь записано:

Test

Test2

7. От пользователя попробовали записать в файл /tmp/file01.txt слово test4, стерев при этом всю имеющуюся в файле информацию командой. Для этого воспользовалась командой `echo "test3" > /tmp/file01.txt`

8. Проверили содержимое файла командой

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

9. От пользователя попробовали удалить файл /tmp/file01.txt командой `rm /tmp/file01.txt`, однако получила отказ.
10. От суперпользователя командой выполнили команду, снимающую атрибут `t` (Sticky-бит) с директории /tmp:

```
chmod -t /tmp
```

Покинули режим суперпользователя командой `exit`.

11. От пользователя проверили, что атрибута `t` у директории /tmp нет:

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

12. Повторили предыдущие шаги. Получилось удалить файл
13. Удалось удалить файл от имени пользователя, не являющегося его владельцем.
14. Повысили свои права до суперпользователя и вернули атрибут `t` на директорию /tmp :

```
su
```

```
chmod +t /tmp
```

```
exit
```

```

[guest@kreachna lab5]$ cd /tmp
[guest@kreachna tmp]$ ls -l | grep tmp
drwx-----, 2 root root 6 Oct 5 11:42 snap-private-tmp
[guest@kreachna tmp]$ echo "test" >> file01.txt
[guest@kreachna tmp]$ ls -l /file01.txt
ls: cannot access '/file01.txt': No such file or directory
[guest@kreachna tmp]$ ls -l /tmp/file01.txt
-rw-r--r--, 1 guest guest 5 Oct 5 12:51 /tmp/file01.txt
[guest@kreachna tmp]$ chmod o+rw /tmp/file01.txt
[guest@kreachna tmp]$ ls -l /tmp/file01.txt
-rw-r--rw-, 1 guest guest 5 Oct 5 12:51 /tmp/file01.txt
[guest@kreachna tmp]$ su guest2
Password:
[guest2@kreachna tmp]$ cat /tmp/file01.txt
test
[guest2@kreachna tmp]$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@kreachna tmp]$ cat /tmp/file01.txt
test
[guest2@kreachna tmp]$ echo "test3" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@kreachna tmp]$ echo "test3" >> /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@kreachna tmp]$ cat /tmp/file01.txt
test
[guest2@kreachna tmp]$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'?
[guest2@kreachna tmp]$ su -
Password:
[root@kreachna ~]# chmod -t /tmp
[root@kreachna ~]# exit
logout
[guest2@kreachna tmp]$ ls -l | grep tmp
drwx-----, 2 root root 6 Oct 5 11:42 snap-private-tmp
[guest2@kreachna tmp]$ rm file01.txt
rm: remove write-protected regular file 'file01.txt'?
[guest2@kreachna tmp]$ su -
Password:
[root@kreachna ~]# chmod +t /tmp
[root@kreachna ~]# exit
logout

```

Рис. 2.8: Исследование Sticky-бита

## 3 Выводы

Изучили механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получили практические навыки работы в консоли с дополнительными атрибутами. Также мы рассмотрели работу механизма смены идентификатора процессов пользователей и влияние бита Sticky на запись и удаление файлов.

# Список литературы

1. КОМАНДА CHATTR В LINUX
2. chattr