

Лабораторная работа № 6

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

Шулуужук Айраана Вячеславовна НПИбд-02-22

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	12

Список иллюстраций

2.1	программа simpleid.c	6
2.2	программа simpleid.c	7
2.3	программа simpleid2.c.	7
2.4	запуск программы simpleid2.c.	8
2.5	запуск программы simpleid2.c.	8
2.6	программа readfile.c	9
2.7	файл file01.txt	10
2.8	запись и удаление файла file01.txt	10
2.9	удаление атрибута t	11
2.10	атрибут t	11

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

Войдем в систему от имени пользователя guest и создадим программу simpleid.c (рис. 2.1)

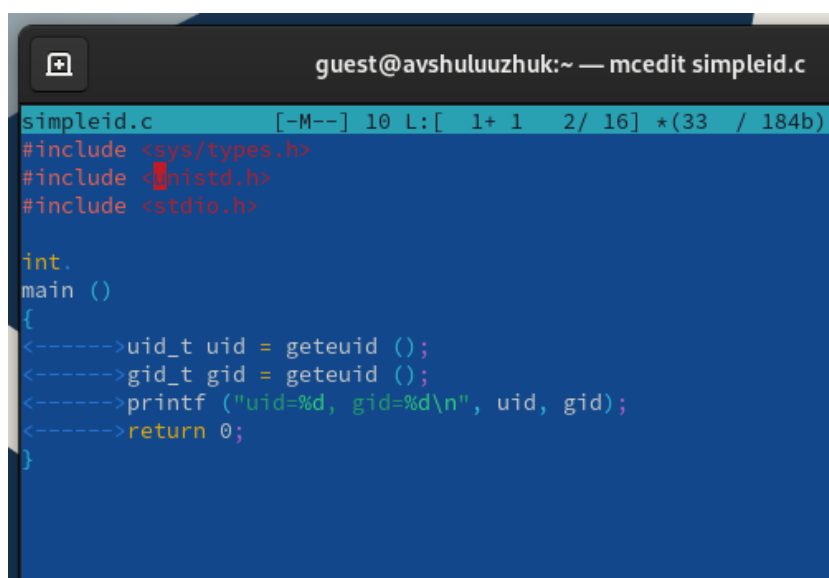
A screenshot of a terminal window with a dark background. The title bar at the top reads 'guest@avshuluuzhuk:~ — mcedit simpleid.c'. The terminal content shows the code for 'simpleid.c'. The first three lines are preprocessor directives: '#include <sys/types.h>', '#include <unistd.h>', and '#include <stdio.h>'. The next line is 'int.' followed by 'main ()' and an opening curly brace '{'. The body of the function contains four lines of code, each preceded by a comment '<----->': 'uid_t uid = geteuid ();', 'gid_t gid = geteuid ();', 'printf ("uid=%d, gid=%d\n", uid, gid);', and 'return 0;'. The function ends with a closing curly brace '}'.

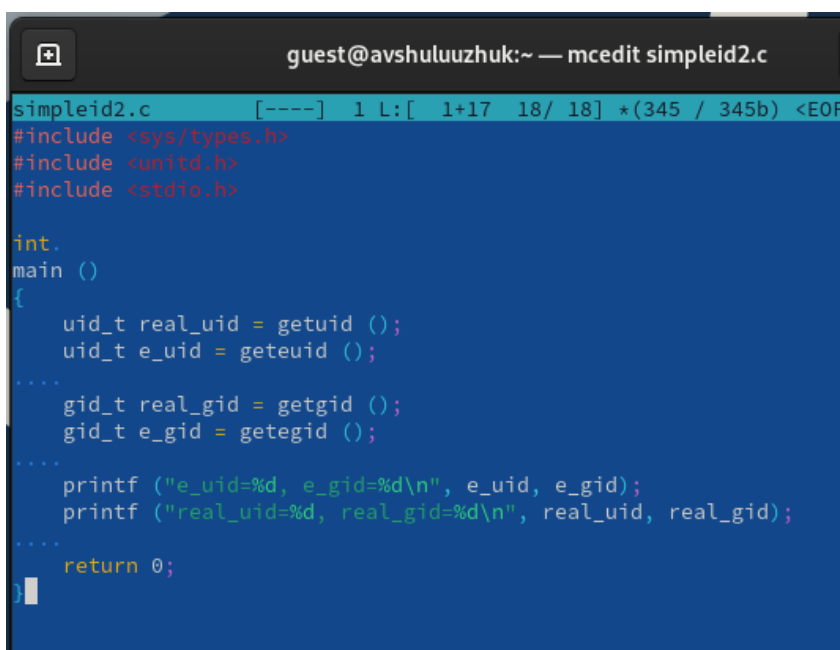
Рис. 2.1: программа simpleid.c

Скомпилируем программу и убедимся, что файл программы создан: `gcc simpleid.c -o simpleid`. Выполним программу simpleid: `./simpleid`. Выполним системную программу `id`: `id` и сравним их вывод (рис. 2.2)

```
[guest@avshuluuzhuk ~]$ mcedit simpleid.c
[guest@avshuluuzhuk ~]$ gcc simpleid.c -o simpleid
[guest@avshuluuzhuk ~]$ ./simpleid
uid=1004, gid=1004
[guest@avshuluuzhuk ~]$ id
uid=1004(guest) gid=100(users) groups=100(users),1005(guest) context=unconfined_
u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@avshuluuzhuk ~]$
```

Рис. 2.2: программа simpleid.c

Усложним программу, добавив вывод действительных идентификаторов, и сохраним ее simpleid2.c. (рис. 2.3)



```
simpleid2.c  [----]  1 L: [ 1+17 18/ 18] *(345 / 345b) <EOF
#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. 2.3: программа simpleid2.c.

Скомпилируем и запустим simpleid2.c: gcc simpleid2.c -o simpleid2
./simpleid2 (рис. 2.4)

```
[guest@avshuluuzhuk ~]$ mcedit simpleid2.c

[guest@avshuluuzhuk ~]$ gcc simpleid2.c -o simpleid2
[guest@avshuluuzhuk ~]$ ./simpleid2
e_uid=1004, e_gid=100
real_uid=1004, real_gid=100
[guest@avshuluuzhuk ~]$
```

Рис. 2.4: запуск программы simpleid2.c.

От имени суперпользователя выполните команды:

```
chown root:guest /home/guest/simpleid2
```

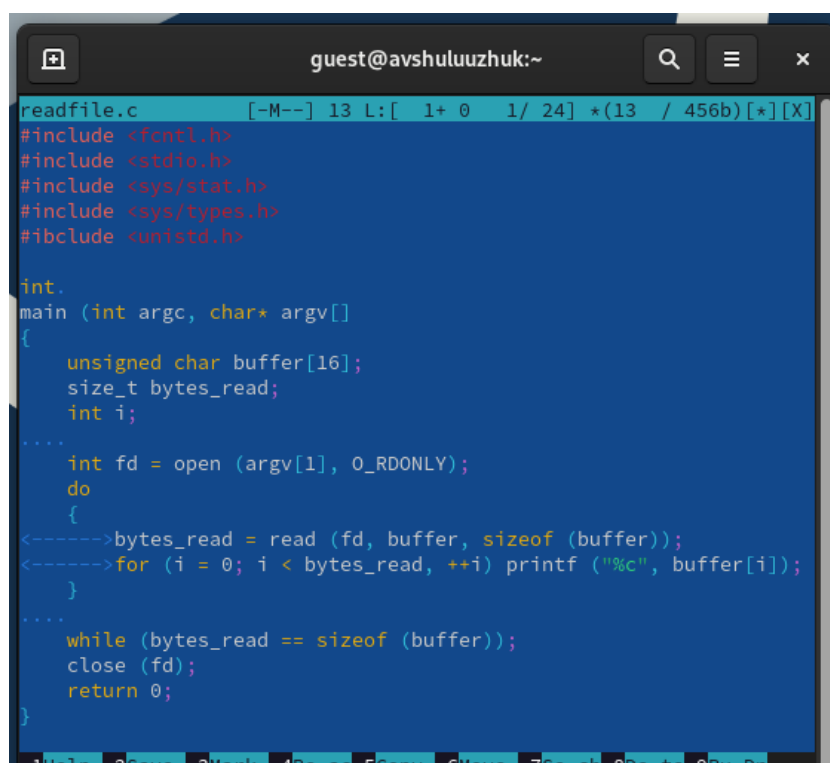
```
chmod u+s /home/guest/simpleid2
```

Выполним проверку правильности установки новых атрибутов и смены владельца файла simpleid2: `ls -l simpleid2`. Запустим simpleid2 и `id` и сравним результаты (рис. 2.5)

```
[guest@avshuluuzhuk ~]$ su -
Password:
[root@avshuluuzhuk ~]# chomn root:guest /home/guest/simpleid2
bash: chomn: command not found...
[root@avshuluuzhuk ~]# chown root:guest /home/guest/simpleid2
[root@avshuluuzhuk ~]# chmod u+s /home/guest/simpleid2
[root@avshuluuzhuk ~]# ls -l simpleid2
ls: cannot access 'simpleid2': No such file or directory
[root@avshuluuzhuk ~]# su guest
[guest@avshuluuzhuk root]$ cd ~
[guest@avshuluuzhuk ~]$ ls -l simpleid2
-rwsr-xr-x. 1 root guest 26048 Apr 10 12:41 simpleid2
[guest@avshuluuzhuk ~]$
```

Рис. 2.5: запуск программы simpleid2.c.

Создадим программу readfile.c и откомпилируем ее (рис. 2.6)



```
readfile.c      [-M--] 13 L:[ 1+ 0 1/ 24] *(13 / 456b) [*] [X]
#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. 2.6: программа readfile.c

Выясним, установлен ли атрибут Sticky на директории /tmp, для чего выполним команду `ls -l / | grep tmp`. От имени пользователя guest создадим файл file01.txt в директории /tmp со словом test: `echo "test" > /tmp/file01.txt`. Просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные» (рис. 2.7) :

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

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

```
ls -l /tmp/file01.txt (рис. 2.7)
```

```
[guest@avshuluuzhuk ~]$ ls -l / | grep tmp
drwxrwxrwt. 17 root root 4096 Apr 10 13:11 tmp
[guest@avshuluuzhuk ~]$ echo "test" > /tmp/file01.txt
[guest@avshuluuzhuk ~]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest users 5 Apr 10 13:14 /tmp/file01.txt
[guest@avshuluuzhuk ~]$ chmod o+rw /tmp/file01.txt
[guest@avshuluuzhuk ~]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest users 5 Apr 10 13:14 /tmp/file01.txt
[guest@avshuluuzhuk ~]$
[guest@avshuluuzhuk ~]$
```

Рис. 2.7: файл file01.txt

От пользователя guest2 (не являющегося владельцем) попробуем прочитать файл /tmp/file01.txt: cat /tmp/file01.txt. От пользователя guest2 попробуем дозаписать в файл /tmp/file01.txt слово test2 командой echo "test2" > /tmp/file01.txt. В итоге нам не удалось это сделать. Проверим содержимое файла командой cat /tmp/file01.txt. От пользователя guest2 попробуйте записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию командой echo "test3" > /tmp/file01.txt. Снова нам не удалось выполнить операцию. Проверьте содержимое файла командой cat /tmp/file01.txt. От пользователя guest2 попробуйте удалить файл /tmp/file01.txt командой rm /tmp/file01.txt. Удалить файл не получилось (рис. 2.8)

```
[guest@avshuluuzhuk ~]$
[guest@avshuluuzhuk ~]$ su guest2
Password:
[guest2@avshuluuzhuk guest]$ cat /tmp/file01.txt
test
[guest2@avshuluuzhuk guest]$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@avshuluuzhuk guest]$ cat /tmp/file01.txt
test
[guest2@avshuluuzhuk guest]$ echo "test3" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@avshuluuzhuk guest]$ cat /tmp/file01.txt
test
[guest2@avshuluuzhuk guest]$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'? y
rm: cannot remove '/tmp/file01.txt': Operation not permitted
[guest2@avshuluuzhuk guest]$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'? yes
rm: cannot remove '/tmp/file01.txt': Operation not permitted
[guest2@avshuluuzhuk guest]$ su -
```

Рис. 2.8: запись и удаление файла file01.txt

Повысим свои права до суперпользователя следующей командой `su -` и выполним после этого команду, снимающую атрибут `t` (Sticky-бит) с директории `/tmp`: `chmod -t /tmp`. Покинем режим суперпользователя командой `exit`. От пользователя `guest2` проверим, что атрибута `t` у директории `/tmp` нет: `ls -l / | grep tmp`. После удаления атрибута у нас получилось удалить файл от имени пользователя `guest2` (рис. 2.9)

```
[guest2@avshuluuzhuk ~]$ su -
Password:
[root@avshuluuzhuk ~]# chmod -t /tmp
[root@avshuluuzhuk ~]# exit
logout
[guest2@avshuluuzhuk ~]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory
[guest2@avshuluuzhuk ~]$
```

Рис. 2.9: удаление атрибута `t`

Повысим свои права до суперпользователя и верните атрибут `t` на директорию `/tmp`: `chmod +t /tmp` (рис. 2.10)

```
[guest2@avshuluuzhuk ~]$ su -
Password:
[root@avshuluuzhuk ~]# chmod +t /tmp
[root@avshuluuzhuk ~]# exit
logout
[guest2@avshuluuzhuk ~]$
```

Рис. 2.10: атрибут `t`

3 Выводы

В ходе выполнения лабораторной работы мы изучили механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получили практические навыки работы в консоли с дополнительными атрибутами. Рассмотрели работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.