

Лабораторная работа № 5

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

Тарусов Артём Сергеевич

Содержание

Цель работы	4
Задание	5
Теоретическое введение	6
Выполнение лабораторной работы	7
Выводы	14
Список литературы	15

Список иллюстраций

1	Создание файла simpleid.c	7
2	Использование команд ./simpleid и id	7
3	Создание и запуск программы simpleid2	7
4	Установки новых атрибутов и смена владельца файла simpleid2	8
5	Использование команд ./simpleid2 и id	8
6	Операции с SetGID-битом	8
7	Создание и компиляция программы readfile.c	8
8	Изменение владельца и прав файла readfile.c	9
9	Проверка, что пользователь guest не может прочитать файл readfile.c. . .	9
10	Работа с параметрами readfile	9
11	Попытка прочитать файл readfile.c программой readfile	10
12	Попытка прочитать файл /etc/shadow программой readfile	10
13	Чтение атрибутов директории /tmp	10
14	Чтение атрибутов директории /tmp	11
15	Попытка прочтения файла /tmp/file01.txt	11
16	Попытка дозаписи в файл /tmp/file01.txt	11
17	Попытка записи в файл /tmp/file01.txt	11
18	Попытка удаления файла /tmp/file01.txt	12
19	Удаление атрибута t директории /tmp	12
20	Повторение предыдущих шагов	12
21	Возвращение атрибута t директории /tmp	13

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

1. Исследовать SetUID- и SetGID-биты.
2. Исследовать Sticky-бит.

Теоретическое введение

- Операционная система — это комплекс программ, предназначенных для управления ресурсами компьютера и организации взаимодействия с пользователем [1].
- Права доступа определяют, какие действия конкретный пользователь может или не может совершать с определенными файлами и каталогами. С помощью разрешений можно создать надежную среду — такую, в которой никто не может менять содержимое ваших документов или повредить системные файлы. [2].

Выполнение лабораторной работы

1. От имени пользователя guest создадим программу simpleid.c, скомпилируем ее и убедимся, что файл создан (fig. 1).

```
[guest@user progs]$ touch simpleid.c
[guest@user progs]$ gcc simpleid.c -o simpleid
[guest@user progs]$ ls
simpleid  simpleid.c
```

Рис. 1: Создание файла simpleid.c

2. Выполним команды ./simpleid и id и убедимся, что полученные данные совпадают (fig. 2).

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

Рис. 2: Использование команд ./simpleid и id

3. Усложним программу и запишем ее в файл simpleid2.c. Запустим получившуюся программу (fig. 3).

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

Рис. 3: Создание и запуск программы simpleid2

4. От имени суперпользователя установим новые атрибуты и сменим владельца файла simpleid2 (fig. 4).

```
[guest@user progs]$ su
Password:
[root@user progs]# chown root:guest /home/guest/simpleid2
chown: cannot access '/home/guest/simpleid2': No such file or directory
[root@user progs]# chown root:guest simpleid2
[root@user progs]# chmod u+s simpleid2
[root@user progs]# ls -l simpleid2
-rwsr-xr-x. 1 root guest 26064 Oct  5 14:23 simpleid2
```

Рис. 4: Установки новых атрибутов и смена владельца файла simpleid2

5. Выполним команды ./simpleid2 и id и убедимся, что полученные данные совпадают (fig. 5).

```
[root@user progs]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@user progs]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 5: Использование команд ./simpleid2 и id

6. Прделаем то же самое относительно SetGID-бита (fig. 6).

```
[root@user progs]# chmod g+s simpleid2
[root@user progs]# ls -l simpleid2
-rwsr-sr-x. 1 root guest 26064 Oct  5 14:23 simpleid2
[root@user progs]# exit
exit
[guest@user progs]$ ^M
: command not found...
[guest@user progs]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@user progs]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 6: Операции с SetGID-битом

7. Создадим и скомпилируем программу readfile.c (fig. 7).

```
[guest@user progs]$ touch readfile.c
[guest@user progs]$ gcc readfile.c -o readfile
```

Рис. 7: Создание и компиляция программы readfile.c

8. Сменим владельца у файла readfile.c и изменим права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог (fig. 8).

```
[guest@user progs]$ su
Password:
[root@user progs]# chown root:guest readfile.c
[root@user progs]# chmod 700 readfile.c
```

Рис. 8: Изменение владельца и прав файла readfile.c

9. Проверим, что пользователь guest не может прочитать файл readfile.c. (fig. 9).

```
[guest@user progs]$ cat readfile.c
cat: readfile.c: Permission denied
```

Рис. 9: Проверка, что пользователь guest не может прочитать файл readfile.c.

10. Сменим у программы readfile владельца и установим SetUID-бит (fig. 10).

```
[guest@user progs]$ su
Password:
[root@user progs]# chown root:guest readfile
[root@user progs]# chmod u+s readfile
```

Рис. 10: Работа с параметрами readfile

11. Проверим, может ли программа readfile прочитать файл readfile.c (fig. 11).

```
[guest@user progs]$ ./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[])
{
```

Рис. 11: Попытка прочитать файл readfile.c программой readfile

12. Проверим, может ли программа readfile прочитать файл /etc/shadow (fig. 12).

```
[guest@user progs]$ ./readfile /etc/shadow
root:$6$uzGy1v1w8gk3IDI8$r0qN7B6vNyXUlgVDry0j906qXjdd2LsgS70tU.IBTWvfKD/IQ0f4G5v0GzUKnIPS030pVV7s/
999:7:::
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:!:19608::::::
```

Рис. 12: Попытка прочитать файл /etc/shadow программой readfile

13. Выясним, установлен ли атрибут Sticky на директории /tmp (fig. 13).

```
[guest@user progs]$ ls -l / | grep tmp
drwxrwxrwt. 16 root root 4096 Oct  5 14:40 tmp
```

Рис. 13: Чтение атрибутов директории /tmp

14. От имени пользователя guest создадим файл file01.txt в директории /tmp со словом test. Просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные» (fig. 14).

```
[guest@user progs]$ echo "test" > /tmp/file01.txt
[guest@user progs]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 Oct  5 14:44 /tmp/file01.txt
[guest@user progs]$ chmod o+rw /tmp/file01.txt
[guest@user progs]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 Oct  5 14:44 /tmp/file01.txt
```

Рис. 14: Чтение атрибутов директории /tmp

15. От пользователя guest2 попробуем прочитать файл /tmp/file01.txt (fig. 15).

```
[guest@user progs]$ su guest2
Password:
[guest2@user progs]$ cat /tmp/file01.txt
test
```

Рис. 15: Попытка прочтения файла /tmp/file01.txt

16. От пользователя guest2 попробуем дозаписать в файл /tmp/file01.txt слово test2 (fig. 16).

```
[guest2@user progs]$ echo "test2" >> /tmp/file01.txt
[guest2@user progs]$ cat /tmp/file01.txt
test2
test2
```

Рис. 16: Попытка дозаписи в файл /tmp/file01.txt

17. От пользователя guest2 попробуем записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию (fig. 17).

```
[guest2@user progs]$ echo "test3" > /tmp/file01.txt
[guest2@user progs]$ cat /tmp/file01.txt
test3
```

Рис. 17: Попытка записи в файл /tmp/file01.txt

18. От пользователя guest2 попробуем удалить файл /tmp/file01.txt (fig. 18).

```
[guest2@user progs]$ rm /tmp/file01.txt  
rm: cannot remove '/tmp/file01.txt': Operation not permitted
```

Рис. 18: Попытка удаления файла /tmp/file01.txt

19. От имени суперпользователя снимем атрибут t с директории /tmp. От пользователя guest2 проверим, что атрибута t у директории /tmp нет (fig. 19).

```
[guest2@user progs]$ su  
Password:  
[root@user progs]# chmod -t /tmp  
[root@user progs]# exit  
exit  
[guest2@user progs]$ ls -l / | grep tmp  
drwxrwxrwx. 17 root root 4096 Oct  5 15:00 tmp
```

Рис. 19: Удаление атрибута t директории /tmp

20. Повторим предыдущие шаги. Теперь мы можем удалить файл (fig. 20).

```
[guest2@user progs]$ echo "test2" >> /tmp/file01.txt  
[guest2@user progs]$ cat /tmp/file01.txt  
test3  
test2  
[guest2@user progs]$ echo "test3" > /tmp/file01.txt  
[guest2@user progs]$ cat /tmp/file01.txt  
test3  
[guest2@user progs]$ rm /tmp/file01.txt  
rm: cannot remove '/tmp/file01.txt': No such file or directory  
[guest2@user progs]$ rm /tmp/file01.txt
```

Рис. 20: Повторение предыдущих шагов

21. Повысим свои права до суперпользователя и вернем атрибут t на директорию /tmp (fig. 20).

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

Рис. 21: Возвращение атрибута t директории /tmp

Выводы

В рамках данной лабораторной работы были изучены механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получены практические навыки работы в консоли с дополнительными атрибутами. Рассмотрены принципы работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

Список литературы

[1] <https://blog.skillfactory.ru/glossary/operaczionnaya-sistema/>

[2] <https://codechick.io/tutorials/unix-linux/unix-linux-permissions>