

Лабораторная работа № 5

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

Алибаева Данагуль НБибд-01-18

Содержание

1	Цель работы	6
2	Задание	7
3	Теоретическое введение	10
4	Выполнение лабораторной работы	12
5	Выводы	22
6	Список литературы	23

List of Tables

List of Figures

4.1	1.1. Проверка компилятора gcc	12
4.2	1.2. Отключение системы запретов	12
4.3	1.3. Ввод команды getenforce	13
4.4	1.4. Проверка компилятора языка C	13
4.5	1.5. Создание программы simpleid.c	13
4.6	1.6. Компилирование программы simpleid.c	14
4.7	1.7. Выполнение программы simpleid.c	14
4.8	1.8. Выполнение системной программы id	14
4.9	1.9. Создание программы simpleid2.c	14
4.10	1.10. Компилирование программы simpleid2.c	15
4.11	1.11. Выполнение команды chown	15
4.12	1.12. Повышение временно прав	15
4.13	1.13. Проверка правильности установки новых атрибутов	15
4.14	1.14. Запуск команд simpleid2 и id	16
4.15	1.15. Повторный запуск команд с SetGID-бита	16
4.16	1.16. Создание программы readfile.c	16
4.17	1.17. Компилирование программы readfile.c	16
4.18	1.18. Смена владельца у файла readfile.c	17
4.19	1.19. Проверка пользователя guest	17
4.20	1.20. Смена у программы readfile владельца	17
4.21	1.21. Проверка чтения файла readfile.c	17
4.22	1.22. Проверка чтения файла /etc/shadow	18
4.23	1.23. Установление атрибута Sticky	18
4.24	1.24. Создание файла file01.txt в директории /tmp	18
4.25	1.25. Разрешение чтения и записи	18
4.26	1.26. Попытка прочитать файл file01.txt	19
4.27	1.27. Попытка дозаписать в файл file01.txt	19
4.28	1.28. Проверка содержимого файла file01.txt	19
4.29	1.29. Попытка записать в файл file01.txt	19
4.30	1.30. Проверка содержимого файла file01.txt	20
4.31	1.31. Попытка удалить файл file01.txt	20
4.32	1.32. Повышение своих прав до суперпользователя	20
4.33	1.33. Ввод команды exit	20
4.34	1.34. Проверка атрибута t	20
4.35	1.35. Повтор предыдущих шагов	21
4.36	1.36. Удаление файла file01.txt	21

4.37 1.37. Возвращение атрибута t	21
---	----

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 Задание

1. Войдите в систему от имени пользователя `guest`.
2. Создайте программу `simpleid.c`
3. Скомпилируйте программу и убедитесь, что файл программы создан: `gcc simpleid.c -o simpleid`
4. Выполните программу `simpleid`: `./simpleid`
5. Выполните системную программу `id`: `id` и сравните полученный вами результат с данными предыдущего пункта задания.
6. Усложните программу, добавив вывод действительных идентификаторов
7. Скомпилируйте и запустите `simpleid2.c`: `gcc simpleid2.c -o simpleid2 ./simpleid2`
8. От имени суперпользователя выполните команды
9. Используйте `sudo` или повысьте временно свои права с помощью `su`. Поясните, что делают эти команды.
10. Выполните проверку правильности установки новых атрибутов и смены владельца файла `simpleid2`: `ls -l simpleid2`
11. Запустите `simpleid2` и `id`: `./simpleid2 id` Сравните результаты.
12. Прodelайте тоже самое относительно `SetGID`-бита.

13. Создайте программу readfile.c
14. Откомпилируйте её. gcc readfile.c -o readfile
15. Смените владельца у файла readfile.c (или любого другого текстового файла в системе) и измените права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог.
16. Проверьте, что пользователь guest не может прочитать файл readfile.c.
17. Смените у программы readfile владельца и установите SetU'D-бит.
18. Проверьте, может ли программа readfile прочитать файл readfile.c?
19. Проверьте, может ли программа readfile прочитать файл /etc/shadow? Отрадите полученный результат и ваши объяснения в отчёте.
20. Выясните, установлен ли атрибут Sticky на директории /tmp, для чего выполните команду `ls -l / | grep tmp`
21. От имени пользователя guest создайте файл file01.txt в директории /tmp со словом test: `echo "test" > /tmp/file01.txt`
22. Просмотрите атрибуты у только что созданного файла и разрешите чтение и запись для категории пользователей «все остальные»: `ls -l /tmp/file01.txt`
`chmod o+rw /tmp/file01.txt` `ls -l /tmp/file01.txt`
23. От пользователя guest2 (не являющегося владельцем) попробуйте прочитать файл /tmp/file01.txt: `cat /tmp/file01.txt`
24. От пользователя guest2 попробуйте дозаписать в файл /tmp/file01.txt слово test2 командой `echo "test2" > /tmp/file01.txt` Удалось ли вам выполнить операцию?
25. Проверьте содержимое файла командой `cat /tmp/file01.txt`

26. 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` Удалось ли вам выполнить операцию?
27. Проверьте содержимое файла командой `cat /tmp/file01.txt`
28. От пользователя guest2 попробуйте удалить файл /tmp/file01.txt командой `rm /tmp/file01.txt` Удалось ли вам удалить файл?
29. Повысьте свои права до суперпользователя следующей командой `su -` и выполните после этого команду, снимающую атрибут t (Sticky-бит) с директории /tmp: `chmod -t /tmp`
30. Покиньте режим суперпользователя командой `exit`
31. От пользователя guest2 проверьте, что атрибута t у директории /tmp нет: `ls -l / | grep tmp`
32. Повторите предыдущие шаги. Какие наблюдаются изменения?
33. Удалось ли вам удалить файл от имени пользователя, не являющегося его владельцем? Ваши наблюдения занесите в отчёт.
34. Повысьте свои права до суперпользователя и верните атрибут t на директорию /tmp: `su - chmod +t /tmp exit`

3 Теоретическое введение

Setuid и Setgid (сокращения от англ. set user ID upon execution — «установка ID пользователя во время выполнения» и англ. set group ID upon execution — «установка ID группы во время выполнения») являются флагами прав доступа в Unix, которые разрешают пользователям запускать исполняемые файлы с правами владельца или группы исполняемого файла. [1]

В Unix-подобных системах приложение запускается с правами пользователя, вызвавшего указанное приложение. Это обеспечивает дополнительную безопасность, так как процесс с правами пользователя не сможет получить доступ к записи к важным системным файлам, например /etc/passwd, который принадлежит суперпользователю root. [1]

Если на исполняемый файл установлен бит suid, то при выполнении эта программа автоматически меняет «эффективный userID» на идентификатор того пользователя, который является владельцем этого файла. То есть, независимо от того — кто запускает эту программу, она при выполнении имеет права хозяина этого файла. [1]

Sticky bit — дополнительный атрибут файлов или каталогов в операционных системах семейства UNIX. [2]

Сегодня когда sticky bit используется в основном для каталогов, чтобы защитить в них файлы. Из такого каталога пользователь может удалить только те файлы, владельцем которых он является. Примером может служить каталог /tmp, в который запись открыта для всех пользователей, но нежелательно удаление чужих файлов. Установка атрибута производится утилитой chmod. [2]

В операционной системе Solaris для файлов, не являющихся программами, имеет строго противоположное действие — запрещает сохранение данных этого файла в системном кэше. [2]

4 Выполнение лабораторной работы

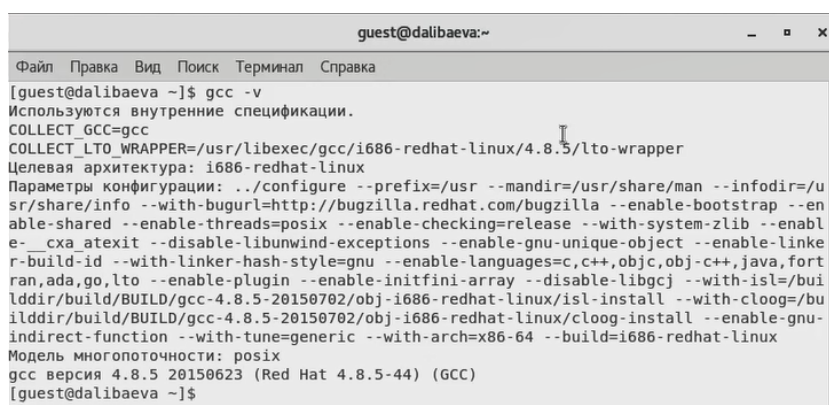
1. Лабораторная работа выполнялась дома со следующими характеристиками техники:

- Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz, 2304 МГц, ядер: 4, логических процессоров: 8

- ОС Майкрософт Windows 10 Pro

- VirtualBox верс. 6.1.26

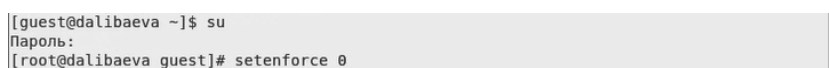
2. Убедилась, что в системе установлен компилятор gcc с помощью команды `gcc -v` (рис 1.1).



```
guest@dalibaeva:~$ gcc -v
Используются внутренние спецификации.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/i686-redhat-linux/4.8.5/lto-wrapper
Целевая архитектура: i686-redhat-linux
Параметры конфигурации: ../configure --prefix=/usr --mandir=/usr/share/man --infodir=/usr/share/info --with-bugurl=http://bugzilla.redhat.com/bugzilla --enable-bootstrap --enable-shared --enable-threads=posix --enable-checking=release --with-system-zlib --enable-cxx-exceptions --enable-gnu-unique-object --enable-linker-hash-style=gnu --enable-languages=c,c++,objc,obj-c++,java,fortran,ada,go,lto --enable-plugin --enable-initfini-array --disable-libgck --with-isl=/build/builddir/build/BUILD/gcc-4.8.5-20150702/obj-i686-redhat-linux/isl-install --with-cloog=/build/builddir/build/BUILD/gcc-4.8.5-20150702/obj-i686-redhat-linux/cloog-install --enable-gnu-indirect-function --with-tune=generic --with-arch=x86-64 --build=i686-redhat-linux
Модель многопоточности: posix
gcc версия 4.8.5 20150623 (Red Hat 4.8.5-44) (GCC)
guest@dalibaeva:~$
```

Figure 4.1: 1.1. Проверка компилятора gcc

3. Отключила систему запретов до очередной перезагрузки системы командой `setenforce 0` (рис 1.2).



```
[guest@dalibaeva ~]$ su
Пароль:
[root@dalibaeva guest]# setenforce 0
```

Figure 4.2: 1.2. Отключение системы запретов

4. Ввела команду `getenforce`, которая должна выводить `Permissive` (рис 1.3).

```
[guest@dalibaeva ~]$ su
Пароль:
[root@dalibaeva guest]# setenforce 0
[root@dalibaeva guest]# su guest
[guest@dalibaeva ~]$ getenforce
Permissive
```

Figure 4.3: 1.3. Ввод команды `getenforce`

5. Проверила компилятор языка C с помощью команд `whereis gcc` и `whereis g++` (рис 1.4).

```
[guest@dalibaeva ~]$ whereis gcc
gcc: /usr/bin/gcc /usr/lib/gcc /usr/libexec/gcc /usr/share/man/man1/gcc.1.gz
[guest@dalibaeva ~]$ whereis g++
g++: /usr/bin/g++ /usr/share/man/man1/g++.1.gz
```

Figure 4.4: 1.4. Проверка компилятора языка C

6. Создала программу `simpleid.c` (рис 1.5).

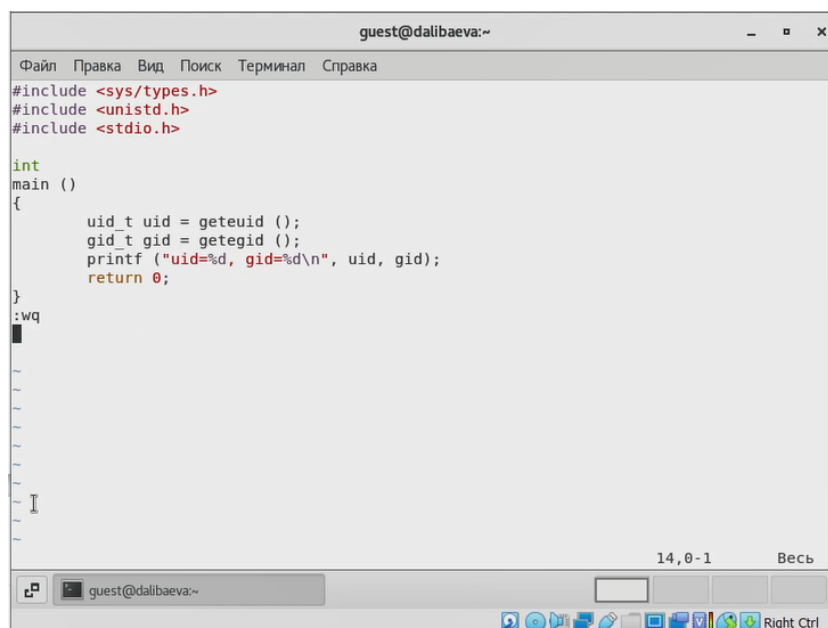


Figure 4.5: 1.5. Создание программы `simpleid.c`

7. Скомпилировала программу и убедилась, что файл программы создан: `gcc simpleid.c -o simpleid` (рис 1.6).

```
[guest@dalibaeva ~]$ whereis gcc
gcc: /usr/bin/gcc /usr/lib/gcc /usr/libexec/gcc /usr/share/man/man1/gcc.1.gz
[guest@dalibaeva ~]$ whereis g++
g++: /usr/bin/g++ /usr/share/man/man1/g++.1.gz
[guest@dalibaeva ~]$ vi simpleid.c
[guest@dalibaeva ~]$ gcc simpleid.c -o simpleid
```

Figure 4.6: 1.6. Компилирование программы simpleid.c

8. Выполнила программу simpleid: ./simpleid (рис 1.7).

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

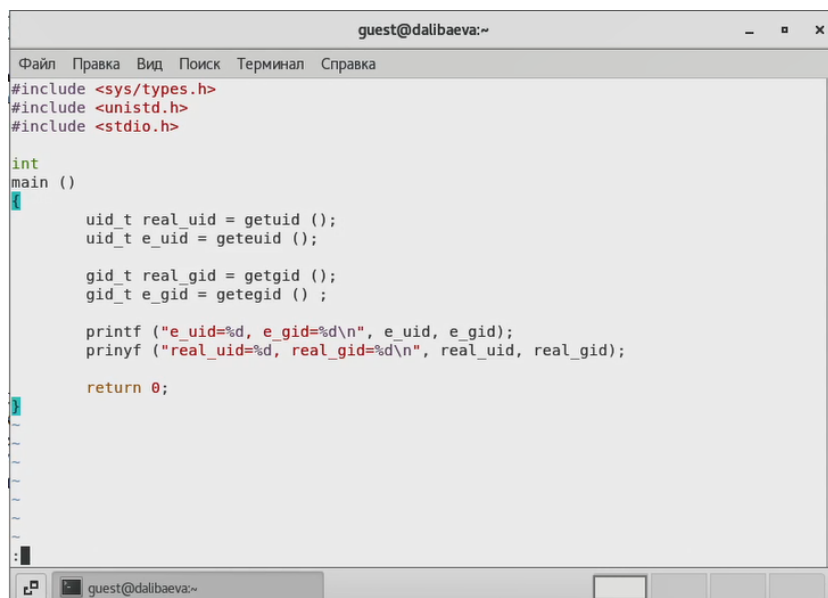
Figure 4.7: 1.7. Выполнение программы simpleid.c

9. Выполнила системную программу id: id и сравнила полученный результат с данными предыдущего пункта задания (рис 1.8). Uid и gid совпадают.

```
[guest@dalibaeva ~]$ ./simpleid
uid=1001, gid=1001
[guest@dalibaeva ~]$ id
uid=1001(guest) gid=1001(guest) rгруппы=1001(guest) контекст=unconfined_u:unconfined_r:u
nconfined_t:s0-s0:c0.c1023
```

Figure 4.8: 1.8. Выполнение системной программы id

10. Усложнила программу, добавив вывод действительных идентификаторов (рис 1.9). Получившуюся программу назвала simpleid2.c.



```
guest@dalibaeva:~
Файл Правка Вид Поиск Терминал Справка
#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);
    prnyf ("real_uid=%d, real_gid=%d\n", real_uid, real_gid);

    return 0;
}
```

Figure 4.9: 1.9. Создание программы simpleid2.c

11. Скомпилировала и запустила simpleid2.c: gcc simpleid2.c -o simpleid2 ./simpleid2 (рис 1.10).

```
[guest@dalibaeva ~]$ vi simpleid2.c
[guest@dalibaeva ~]$ gcc simpleid2.c -o simpleid2
/tmp/ccDxf6i7.o: In function 'main':
simpleid2.c:(.text+0x61): undefined reference to `prnfyf'
collect2: ошибка: выполнение ld завершилось с кодом возврата 1
[guest@dalibaeva ~]$ vi simpleid2.c
[guest@dalibaeva ~]$ gcc simpleid2.c -o simpleid2
[guest@dalibaeva ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
```

Figure 4.10: 1.10. Компилирование программы simpleid2.c

12. От имени суперпользователя выполнила команды chown root:guest /home/guest/simpleid2 chmod u+s /home/guest/simpleid2 (рис 1.11).

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

Figure 4.11: 1.11. Выполнение команды chown

13. Повысила временно свои права с помощью su (рис 1.12).

```
[root@dalibaeva guest]# chown root:guest /home/guest/simpleid2
[root@dalibaeva guest]# chmod u+s /home/guest/simpleid2
[root@dalibaeva guest]# su guest
```

Figure 4.12: 1.12. Повышение временно прав

14. Выполнила проверку правильности установки новых атрибутов и смены владельца файла simpleid2: ls -l simpleid2 (рис 1.13).

```
[root@dalibaeva guest]# su guest
[guest@dalibaeva ~]$ ls -l simpleid2
-rwsrwxr-x. 1 root guest 7376 ноя 12 19:31 simpleid2
```

Figure 4.13: 1.13. Проверка правильности установки новых атрибутов

15. Запустила simpleid2 и id: ./simpleid2 id Сравнила результаты (рис 1.14). Программа запустилась от root с uid = 0, но реальный его id = 1002.

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

Figure 4.14: 1.14. Запуск команд simpleid2 и id

16. Проделала тоже самое относительно SetGID-бита (рис 1.15).

```
[guest@dalibaeva ~]$ su
Пароль:
[root@dalibaeva guest]# chown root:guest /home/guest/simpleid2
[root@dalibaeva guest]# chmod g+s /home/guest/simpleid2
[root@dalibaeva guest]# su guest
[guest@dalibaeva ~]$ ls -l simpleid2
-rwxrwsr-x. 1 root guest 7376 ноя 12 19:31 simpleid2
[guest@dalibaeva ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@dalibaeva ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Figure 4.15: 1.15. Повторный запуск команд с SetGID-бита

17. Создала программу readfile.c (рис 1.16).

```
guest@dalibaeva:~
Файл Правка Вид Поиск Терминал Справка
#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;
}
```

Figure 4.16: 1.16. Создание программы readfile.c

18. Откомпилировала её с помощью команды gcc readfile.c -o readfile (рис 1.17).

```
[guest@dalibaeva ~]$ vi readfile.c
[guest@dalibaeva ~]$ gcc readfile.c -o readfile
[guest@dalibaeva ~]$
```

Figure 4.17: 1.17. Компилирование программы readfile.c

19. Сменила владельца у файла readfile.c и изменила права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог (рис 1.18).

```
[guest@dalibaeva ~]$ su
Пароль:
[root@dalibaeva guest]# chown root:guest /home/guest/readfile.c
[root@dalibaeva guest]# chmod 700 /home/guest/readfile.c
[root@dalibaeva guest]# su guest
```

Figure 4.18: 1.18. Смена владельца у файла readfile.c

20. Проверила, что пользователь guest не может прочитать файл readfile.c (рис 1.19).

```
[root@dalibaeva guest]# su guest
[guest@dalibaeva ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
```

Figure 4.19: 1.19. Проверка пользователя guest

21. Сменила у программы readfile владельца и установила SetU'D-бит (рис 1.20).

```
[guest@dalibaeva ~]$ su
Пароль:
[root@dalibaeva guest]# chown root:guest /home/guest/readfile
[root@dalibaeva guest]# chmod u+s /home/guest/readfile
[root@dalibaeva guest]# chmod g+s /home/guest/readfile
[root@dalibaeva guest]# su guest
```

Figure 4.20: 1.20. Смена у программы readfile владельца

22. Проверила, может ли программа readfile прочитать файл readfile.c (рис 1.21).
Да, может.

```
[guest@dalibaeva ~]$ ./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]);
    }
}
```

Figure 4.21: 1.21. Проверка чтения файла readfile.c

23. Проверила, может ли программа readfile прочитать файл /etc/shadow (рис 1.22). Да, может.

```
[guest@dalibaeva ~]$ ./readfile /etc/shadow
root:$6$J/eJY5IfRVktE0G3$cZ0H56UoJ5ubtGxUNXau2G0s8BIBq9ID0tg0CQmGZFzsn57skNCNSAy1l7nxMH
8mY6eSZqsZJU2HALrL4PZ0i0::0:99999:7:::
bin:!:18353:0:99999:7:::
daemon:!:18353:0:99999:7:::
adm:!:18353:0:99999:7:::
lp:!:18353:0:99999:7:::
sync:!:18353:0:99999:7:::
shutdown:!:18353:0:99999:7:::
halt:!:18353:0:99999:7:::
mail:!:18353:0:99999:7:::
operator:!:18353:0:99999:7:::
games:!:18353:0:99999:7:::
ftp:!:18353:0:99999:7:::
nobody:!:18353:0:99999:7:::
systemd-network:!:18886:::
dbus:!:18886:::
polkitd:!:18886:::
libstoragemgmt:!:18886:::
colord:!:18886:::
abrt:!:18886:::
rpc:!:18886:0:99999:7:::
rtkit:!:18886:::
```

Figure 4.22: 1.22. Проверка чтения файла /etc/shadow

1. Выяснила, установлен ли атрибут Sticky на директории /tmp, с помощью команды `ls -l / | grep tmp` (рис 1.23).

```
[guest@dalibaeva ~]$ ls -l / | grep tmp
drwxrwxrwt. 17 root root 4096 ноя 12 20:21 tmp
```

Figure 4.23: 1.23. Установление атрибута Sticky

2. От имени пользователя guest создала файл file01.txt в директории /tmp со словом test: `echo "test" > /tmp/file01.txt` (рис 1.24).

```
[guest@dalibaeva ~]$ ls -l / | grep tmp
drwxrwxrwt. 17 root root 4096 ноя 12 20:21 tmp
[guest@dalibaeva ~]$ echo "test" > /tmp/file01.txt
```

Figure 4.24: 1.24. Создание файла file01.txt в директории /tmp

3. Просмотрела атрибуты у только что созданного файла и разрешила чтение и запись для категории пользователей «все остальные»: `ls -l /tmp/file01.txt` `chmod o+rw /tmp/file01.txt` `ls -l /tmp/file01.txt` (рис 1.25).

```
[guest@dalibaeva ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 ноя 12 20:22 /tmp/file01.txt
[guest@dalibaeva ~]$ chmod o+rw /tmp/file01.txt
[guest@dalibaeva ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 ноя 12 20:22 /tmp/file01.txt
```

Figure 4.25: 1.25. Разрешение чтения и записи

4. От пользователя guest2 (не являющегося владельцем) попробовала прочитать файл /tmp/file01.txt: cat /tmp/file01.txt (рис 1.26).

```
[guest2@dalibaeva ~]$ su guest2
Пароль:
su: Сбой при проверке подлинности
[guest2@dalibaeva ~]$ su guest2
Пароль:
[guest2@dalibaeva guest]$ cd /home/guest2
[guest2@dalibaeva ~]$ cat /tmp/file01.txt
test
```

Figure 4.26: 1.26. Попытка прочитать файл file01.txt

5. От пользователя guest2 попробовала дозаписать в файл /tmp/file01.txt слово test2 командой echo "test2" > /tmp/file01.txt (рис 1.27). Операцию выполнить удалось.

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

Figure 4.27: 1.27. Попытка дозаписать в файл file01.txt

6. Проверила содержимое файла командой cat /tmp/file01.txt (рис 1.28).

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

Figure 4.28: 1.28. Проверка содержимого файла file01.txt

7. От пользователя guest2 попробовала записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию командой echo "test3" > /tmp/file01.txt (рис 1.29). Операцию выполнить удалось.

```
[guest2@dalibaeva ~]$ cat /tmp/file01.txt
test2
[guest2@dalibaeva ~]$ echo "test3" > /tmp/file01.txt
```

Figure 4.29: 1.29. Попытка записать в файл file01.txt

8. Проверила содержимое файла командой cat /tmp/file01.txt (рис 1.30).

```
test2
[guest2@dalibaeva ~]$ echo "test3" > /tmp/file01.txt
[guest2@dalibaeva ~]$ cat /tmp/file01.txt
test3
```

Figure 4.30: 1.30. Проверка содержимого файла file01.txt

9. От пользователя guest2 попробовала удалить файл /tmp/file01.txt командой `rm /tmp/file01.txt` (рис 1.31). Команду выполнить не удалось.

```
[guest2@dalibaeva ~]$ cat /tmp/file01.txt
test3
[guest2@dalibaeva ~]$ rm /tmp/file01.txt
rm: невозможно удалить «/tmp/file01.txt»: Операция не позволена
```

Figure 4.31: 1.31. Попытка удалить файл file01.txt

10. Повысила свои права до суперпользователя следующей командой `su -` и выполнила после этого команду, снимающую атрибут `t` (Sticky-бит) с директории `/tmp`: `chmod -t /tmp` (рис 1.32).

```
[guest2@dalibaeva ~]$ su -
Пароль:
Последний вход в систему: Пт ноя 12 20:07:19 MSK 2021 на pts/0
[root@dalibaeva ~]# chmod -t /tmp
```

Figure 4.32: 1.32. Повышение своих прав до суперпользователя

11. Покинула режим суперпользователя командой `exit` (рис 1.33).

```
[guest2@dalibaeva ~]$ su -
Пароль:
Последний вход в систему: Пт ноя 12 20:07:19 MSK 2021 на pts/0
[root@dalibaeva ~]# chmod -t /tmp
[root@dalibaeva ~]# exit
logout
```

Figure 4.33: 1.33. Ввод команды exit

12. От пользователя guest2 проверила, что атрибута `t` у директории `/tmp` нет: `ls -l / | grep tmp` (рис 1.34).

```
[root@dalibaeva ~]# exit
logout
[guest2@dalibaeva ~]$ ls -l / | grep tmp
drwxrwxrwx. 16 root root 4096 ноя 12 20:26 tmp
```

Figure 4.34: 1.34. Проверка атрибута t

13. Повторила предыдущие шаги (рис 1.35).

```
[guest2@dalibaeva ~]$ ls -l / | grep tmp
drwxrwxrwx. 16 root root 4096 ноя 12 20:26 tmp
[guest2@dalibaeva ~]$ cat /tmp/file01.txt
test3
[guest2@dalibaeva ~]$ echo "test2" > /tmp/file01.txt
[guest2@dalibaeva ~]$ cat /tmp/file01.txt
test2
[guest2@dalibaeva ~]$ echo "test3" > /tmp/file01.txt
[guest2@dalibaeva ~]$ cat /tmp/file01.txt
test3
```

Figure 4.35: 1.35. Повтор предыдущих шагов

14. Файл удалить удалось (рис 1.36). Получается, что если есть атрибут Sticky t, то удалять файл может только владелец файла. Но если атрибута нет, то удалить этот файл может любой пользователь.

```
[guest2@dalibaeva ~]$ echo "test3" > /tmp/file01.txt
[guest2@dalibaeva ~]$ cat /tmp/file01.txt
test3
[guest2@dalibaeva ~]$ rm /tmp/file01.txt
```

Figure 4.36: 1.36. Удаление файла file01.txt

15. Повысила свои права до суперпользователя и вернула атрибут t на директорию /tmp: su - chmod +t /tmp exit (рис 1.37).

```
[guest2@dalibaeva ~]$ su -
Пароль:
Последний вход в систему: Пт ноя 12 20:26:22 MSK 2021 на pts/0
[root@dalibaeva ~]# chmod +t /tmp
[root@dalibaeva ~]# exit
logout
[guest2@dalibaeva ~]$ █
```

Figure 4.37: 1.37. Возвращение атрибута t

5 Выводы

В результате выполнения работы я изучила механизмы изменения идентификаторов, применение SetUID- и Sticky-битов. Получила практические навыки работы в консоли с дополнительными атрибутами. Рассмотрела работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

6 Список литературы

- SetUID // URL: [https://ru.wikipedia.org/wiki/Suid#:~:text=setuid%20%D0%B8%20setgid%20%D0%A6%D0%BC](https://ru.wikipedia.org/wiki/Suid#:~:text=setuid%20%D0%B8%20setgid%20D0%A6%D0%BC) (дата обращения: 12.11.2021).
- Sticky bit // URL: https://ru.wikipedia.org/wiki/Sticky_bit (дата обращения: 12.11.2021).