

Отчёт по лабораторной работе №5

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

Шемякин Алексей НФИбд-02-18

Содержание

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

List of Figures

2.1	подготовка к работе	5
2.2	программа simpleid	6
2.3	результат программы simpleid	7
2.4	программа simpleid2	7
2.5	результат программы simpleid2	8
2.6	программа readfile	9
2.7	результат программы readfile	10
2.8	исследование Sticky-бита	12

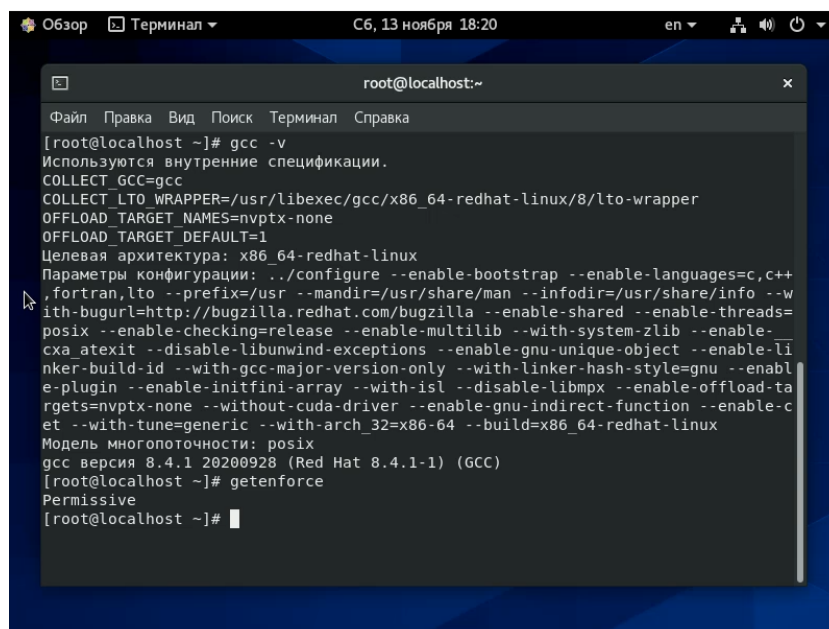
1 Цель работы

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`:

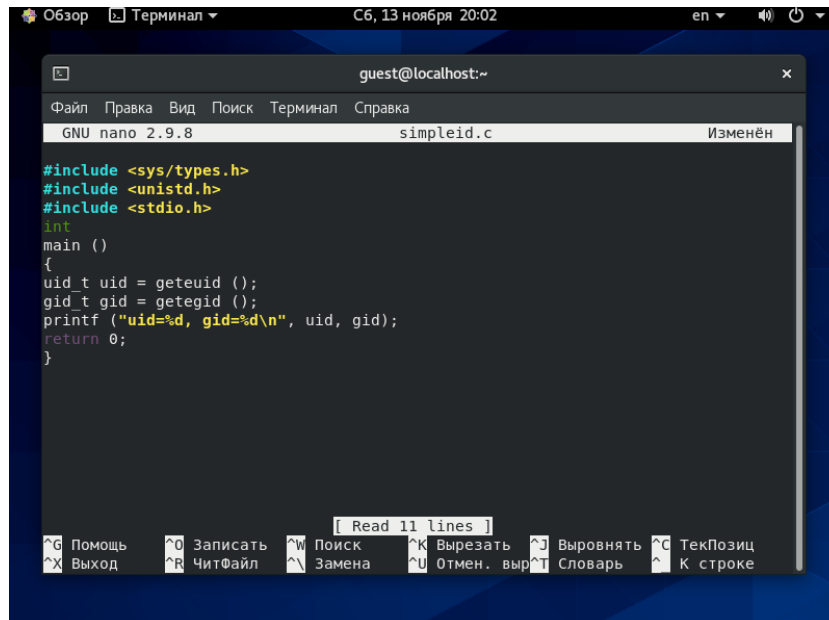


```
Обзор Терминал C6, 13 ноября 18:20 en
root@localhost:~
Файл Правка Вид Поиск Терминал Справка
[root@localhost ~]# gcc -v
Используются внутренние спецификации.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/x86_64-redhat-linux/8/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none
OFFLOAD_TARGET_DEFAULT=1
Целевая архитектура: x86_64-redhat-linux
Параметры конфигурации: ../configure --enable-bootstrap --enable-languages=c,c++,fortran,lto --prefix=/usr --mandir=/usr/share/man --infodir=/usr/share/info --with-bugurl=http://bugzilla.redhat.com/bugzilla --enable-shared --enable-threads=posix --enable-checking=release --enable-multilib --with-system-zlib --enable-cxx-atomic --disable-libunwind-exceptions --enable-gnu-unique-object --enable-linker-build-id --with-gcc-major-version-only --with-linker-hash-style=gnu --enable-plugin --enable-initfini-array --with-isl --disable-libmpx --enable-offload-targets=nvptx-none --without-cuda-driver --enable-gnu-indirect-function --enable-cet --with-tune=generic --with-arch_32=x86-64 --build=x86_64-redhat-linux
Модель многопоточности: posix
gcc версия 8.4.1 20200928 (Red Hat 8.4.1-1) (GCC)
[root@localhost ~]# getenforce
Permissive
[root@localhost ~]#
```

Figure 2.1: подготовка к работе

2.2 Изучение механики SetUID

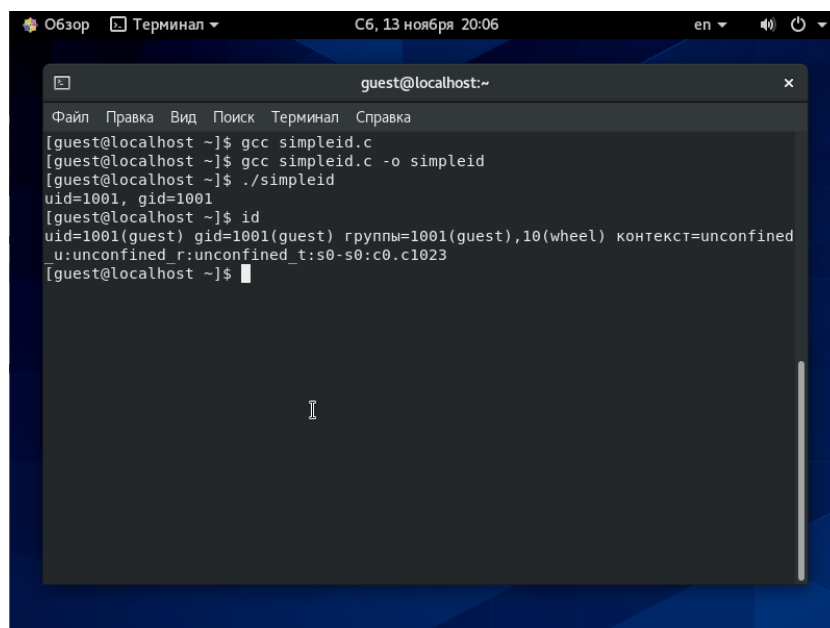
1. Вошли в систему от имени пользователя guest.
2. Написали программу simpleid.c.



```
guest@localhost:~  
Файл Правка Вид Поиск Терминал Справка  
GNU nano 2.9.8 simpleid.c Изменён  
  
#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;  
}  
  
[ Read 11 lines ]  
^G Помощь ^O Записать ^W Поиск ^K Вырезать ^J Вывернуть ^C ТекПозиц  
^X Выход ^R ЧитФайл ^\ Замена ^U Отмен. выр ^T Словарь ^_ К строке
```

Figure 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

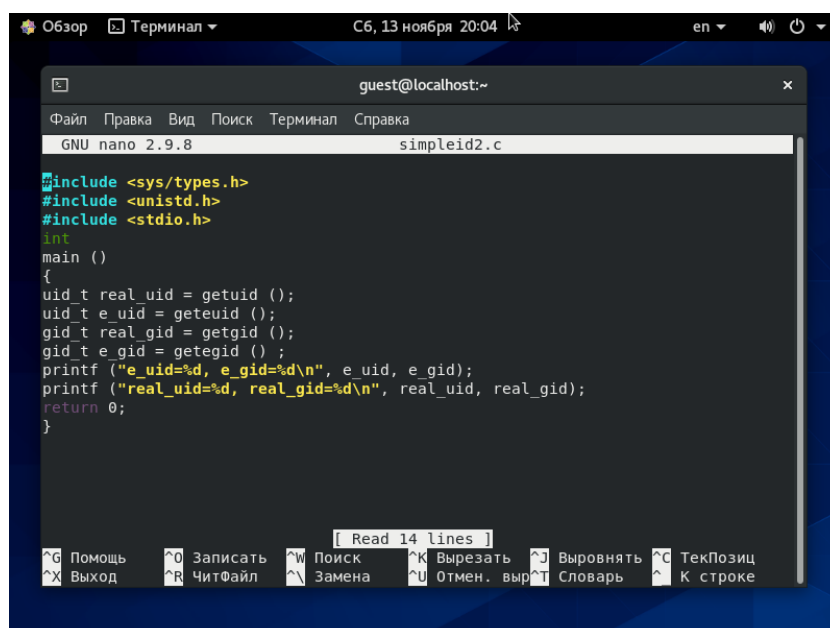


A terminal window titled 'guest@localhost:~' with a menu bar (Файл, Правка, Вид, Поиск, Терминал, Справка). The terminal shows the following commands and output:

```
[guest@localhost ~]$ gcc simpleid.c
[guest@localhost ~]$ gcc simpleid.c -o simpleid
[guest@localhost ~]$ ./simpleid
uid=1001, gid=1001
[guest@localhost ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest),10(wheel) контекст=unconfined
u:unconfined r:unconfined_t:s0-s0:c0.c1023
[guest@localhost ~]$
```

Figure 2.3: результат программы simpleid

6. Усложнили программу, добавив вывод действительных идентификаторов.



A terminal window titled 'guest@localhost:~' showing the GNU nano 2.9.8 editor editing 'simpleid2.c'. The code is as follows:

```
#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;
}
```

The bottom of the window shows a status bar with 'Read 14 lines' and a menu bar with various keyboard shortcuts like ^G Помощь, ^O Записать, ^W Поиск, etc.

Figure 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:

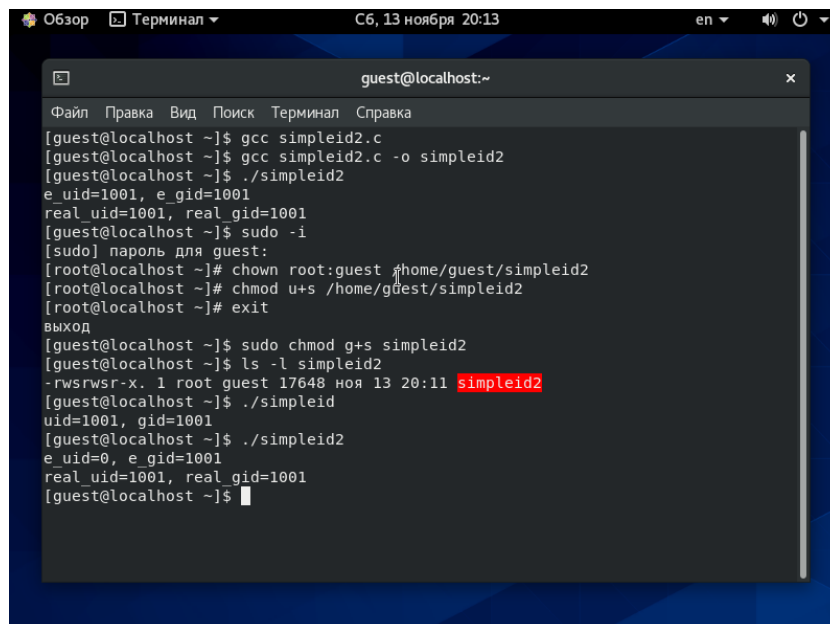
```
ls -l simpleid2
```

11. Запустили simpleid2 и id:

```
./simpleid2
id
```

Результат выполнения программ теперь немного отличается

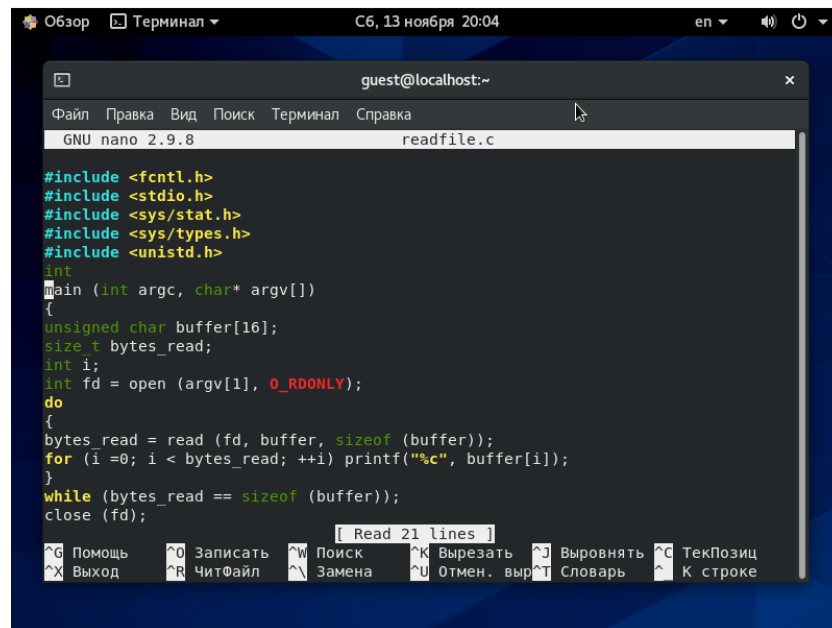
12. Проделали тоже самое относительно SetGID-бита.



```
Обзор Терминал C6, 13 ноября 20:13 en
guest@localhost:~
Файл Правка Вид Поиск Терминал Справка
[guest@localhost ~]$ gcc simpleid2.c
[guest@localhost ~]$ gcc simpleid2.c -o simpleid2
[guest@localhost ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real uid=1001, real_gid=1001
[guest@localhost ~]$ sudo -i
[sudo] пароль для guest:
[root@localhost ~]# chown root:guest /home/guest/simpleid2
[root@localhost ~]# chmod u+s /home/guest/simpleid2
[root@localhost ~]# exit
выход
[guest@localhost ~]$ sudo chmod g+s simpleid2
[guest@localhost ~]$ ls -l simpleid2
-rwsrwsr-x. 1 root guest 17648 ноя 13 20:11 simpleid2
[guest@localhost ~]$ ./simpleid2
uid=1001, gid=1001
[guest@localhost ~]$ ./simpleid2
e_uid=0, e_gid=1001
real uid=1001, real_gid=1001
[guest@localhost ~]$
```

Figure 2.5: результат программы simpleid2

13. Написали программу readfile.c



```
guest@localhost:~  
Файл Правка Вид Поиск Терминал Справка  
GNU nano 2.9.8 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);  
}
```

Figure 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
```

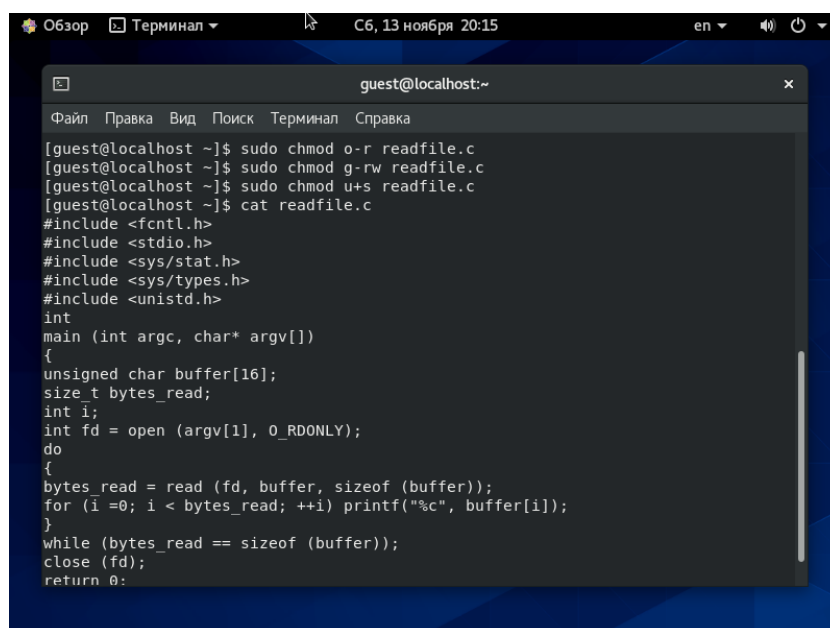
```
chmod 700 /home/guest/readfile.c
```

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



```
Обзор Терминал Сб, 13 ноября 20:15 en
guest@localhost:~
Файл Правка Вид Поиск Терминал Справка
[guest@localhost ~]$ sudo chmod o-r readfile.c
[guest@localhost ~]$ sudo chmod g-rw readfile.c
[guest@localhost ~]$ sudo chmod u+s readfile.c
[guest@localhost ~]$ cat 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;
}
```

Figure 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

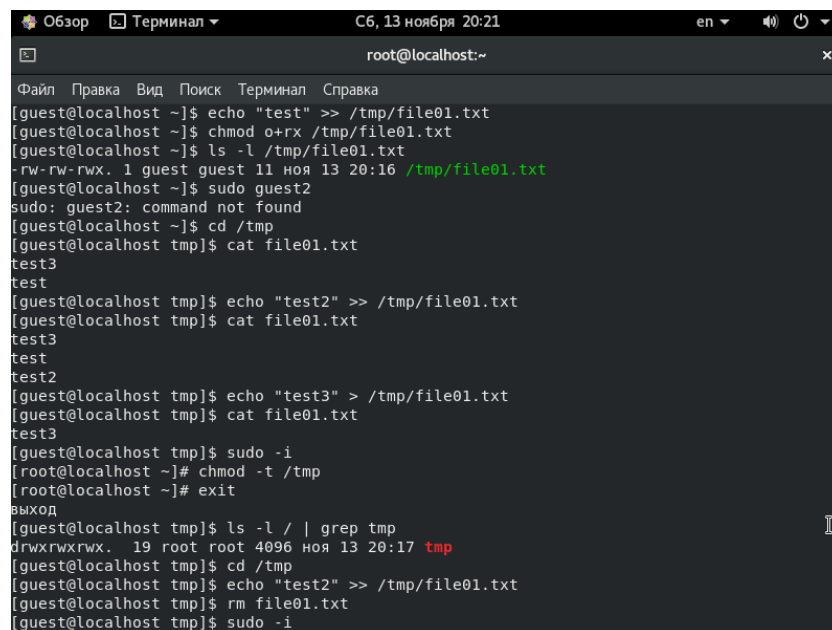
13. Удалось удалить файл от имени пользователя, не являющегося его владельцем.

14. Повысили свои права до суперпользователя и вернули атрибут `t` на директорию `/tmp` :

```
su
```

```
chmod +t /tmp
```

```
exit
```



```
Обзор Терминал C6, 13 ноября 20:21 en
root@localhost:~
Файл Правка Вид Поиск Терминал Справка
[guest@localhost ~]$ echo "test" >> /tmp/file01.txt
[guest@localhost ~]$ chmod o+rx /tmp/file01.txt
[guest@localhost ~]$ ls -l /tmp/file01.txt
-rw-rw-rwx. 1 guest guest 11 ноя 13 20:16 /tmp/file01.txt
[guest@localhost ~]$ sudo guest2
sudo: guest2: command not found
[guest@localhost ~]$ cd /tmp
[guest@localhost tmp]$ cat file01.txt
test3
test
[guest@localhost tmp]$ echo "test2" >> /tmp/file01.txt
[guest@localhost tmp]$ cat file01.txt
test3
test
test2
[guest@localhost tmp]$ echo "test3" > /tmp/file01.txt
[guest@localhost tmp]$ cat file01.txt
test3
[guest@localhost tmp]$ sudo -i
[root@localhost ~]# chmod -t /tmp
[root@localhost ~]# exit
выход
[guest@localhost tmp]$ ls -l / | grep tmp
drwxrwxrwx. 19 root root 4096 ноя 13 20:17 tmp
[guest@localhost tmp]$ cd /tmp
[guest@localhost tmp]$ echo "test2" >> /tmp/file01.txt
[guest@localhost tmp]$ rm file01.txt
[guest@localhost tmp]$ sudo -i
```

Figure 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