

Отчёт по лабораторной работе 5

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

Радимов Игорь

Содержание

1	Цель работы	5
2	Задание	6
3	Теория	7
4	Выполнение лабораторной работы	8
4.1	Подготовка к выполнению	8
4.2	Выполнение основной части лабораторной работы	8
4.3	Исследование Sticky-бита	13
5	Библиография	17
6	Выводы	18

List of Tables

List of Figures

4.1	рис.1. Установка компилятора gcc.	8
4.2	рис.2. Программа simpleid.c	9
4.3	рис.3. Компиляция программы simpleid, её выполнение и сравнение с командой id.	9
4.4	рис.4. Программа simpleid2.c	10
4.5	рис.5. Компиляция программы simpleid2, её выполнение.	10
4.6	рис.6. Изменение владельца программы и установка SetUID-бита, проверка установки и изменения, запуск программы и команды id.	11
4.7	рис.7. Установка SetGID-бита, проверка установки, запуск программы и команды id.	11
4.8	рис.8. Программа readfile.c	12
4.9	рис.9. Компиляция readfile и другие действия в соответствии с 14-17 пунктами.	12
4.10	рис.10. Выполнение программы readfile с файлом readfile.c.	13
4.11	рис.11. Выполнение программы readfile с файлом /etc/shadow.	13
4.12	рис.12. Выполнение пунктов 1-4 исследования Sticky-бита	14
4.13	рис.13. Выполнение пунктов 5-9 исследования Sticky-бита	15
4.14	рис.14. Выполнение пунктов 10-13 исследования Sticky-бита	15
4.15	рис.15. Возвращение атрибута t на директорию.	16

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

Лабораторная работа подразумевает изучение влияния дополнительных атрибутов на файлы пользователя и изучение механизмов изменения идентификаторов.

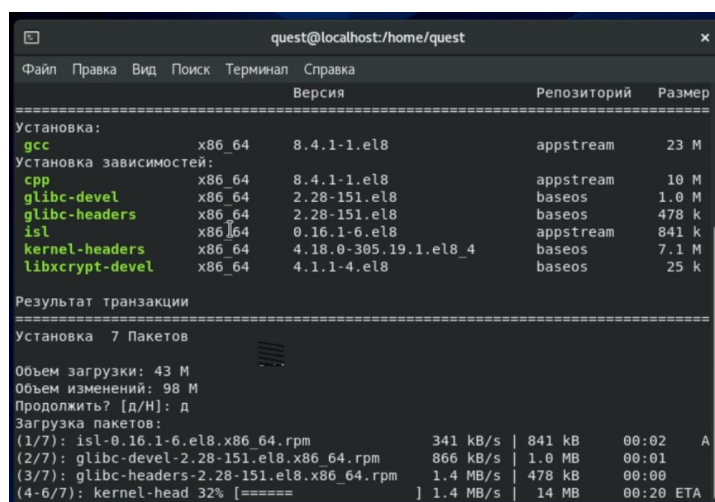
3 Теория

Биты SUID, SGID и Sticky Unix отслеживает не символьные имена владельцев и групп, а их идентификаторы (UID - для пользователей и GID для групп). Эти идентификаторы хранятся в файлах `/etc/passwd` и `/etc/group` соответственно. Установка битов SUID или SGID позволит пользователям запускать исполняемые файлы от имени владельца (или группы) запускаемого файла. Если мы установим SUID на исполняемый файл `/bin/chmod`, то обычный пользователь сможет использовать эту команду без использования `sudo`, так, что она будет выполняться от имени пользователя `root`. Каталог с установленным sticky-битом означает, что удалить файл из этого каталога может только владелец файла или суперпользователь. Другие пользователи лишаются права удалять файлы.

4 Выполнение лабораторной работы

4.1 Подготовка к выполнению

1. Установил компилятор gcc.



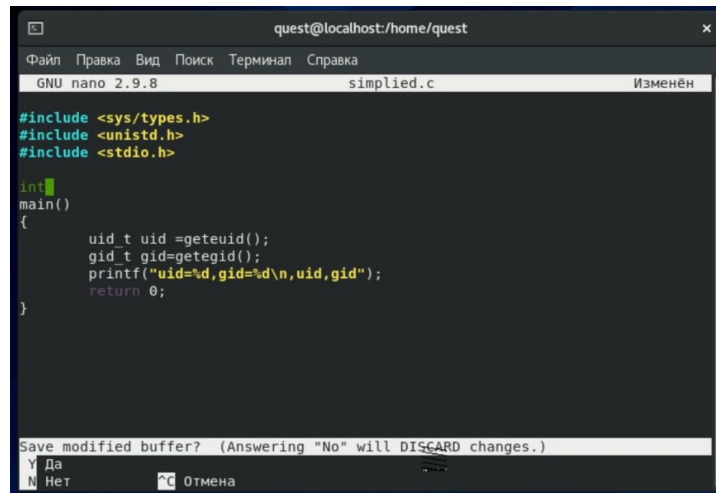
```
quest@localhost:/home/quest
Файл Правка Вид Поиск Терминал Справка
=====
Версия                                Репозиторий  Размер
=====
Установка:
gcc                                x86_64      8.4.1-1.el8    appstream    23 М
Установка зависимостей:
cpp                                x86_64      8.4.1-1.el8    appstream    10 М
glibc-devel                       x86_64      2.28-151.el8   baseos       1.0 М
glibc-headers                    x86_64      2.28-151.el8   baseos       478 k
isl                               x86_64      0.16.1-6.el8   appstream    841 k
kernel-headers                   x86_64      4.18.0-305.19.1.el8_4 baseos       7.1 М
libxcrypt-devel                  x86_64      4.1.1-4.el8    baseos       25 k
=====
Результат транзакции
=====
Установка 7 Пакетов

Объем загрузки: 43 М
Объем изменений: 98 М
Продолжить? [д/н]: д
Загрузка пакетов:
(1/7): isl-0.16.1-6.el8.x86_64.rpm      341 kB/s | 841 kB    00:02  A
(2/7): glibc-devel-2.28-151.el8.x86_64.rpm  866 kB/s | 1.0 MB   00:01
(3/7): glibc-headers-2.28-151.el8.x86_64.rpm 1.4 MB/s | 478 kB   00:00
(4-6/7): kernel-head 32% [=====] 1.4 MB/s | 14 MB    00:20 ETA
```

Figure 4.1: рис.1. Установка компилятора gcc.

4.2 Выполнение основной части лабораторной работы

1. Вошёл в систему от имени пользователя quest.
2. Создал программу simpleid.c (рис. 2).



```
quest@localhost:/home/quest
Файл  Правка  Вид  Поиск  Терминал  Справка
GNU nano 2.9.8      simplified.c      Изменён

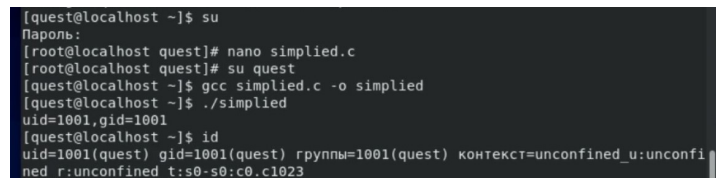
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int
main()
{
    uid_t uid =getuid();
    gid_t gid=getgid();
    printf("uid=%d,gid=%d\n,uid,gid");
    return 0;
}

Save modified buffer? (Answering "No" will DISCARD changes.)
Y Да
N Нет      Отмена
```

Figure 4.2: рис.2. Программа simplified.c

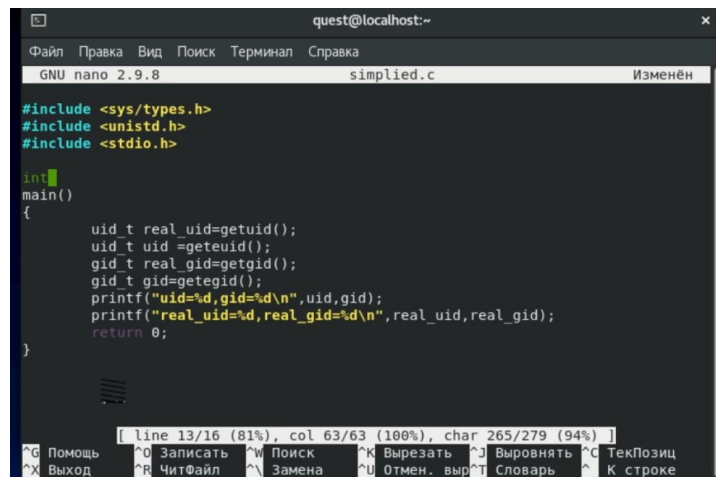
3. Скомпилировал программу и убедился, что файл программы создан командой `gcc simplified.c -o simpleid` (рис. 3).
4. Выполнял программу `simpleid` командой `./simpleid` (рис. 3).
5. Выполнял системную программу `id` и сравнил полученный результат с данными предыдущего пункта задания (рис. 3). Видим, что пользователи и группы совпадают. При этом команда `id` вывела действительные идентификаторы, а программа вывела эффективные, но при этом они совпадают и выводят 1001, то есть пользователя `quest`.



```
[quest@localhost ~]$ su
Пароль:
[root@localhost quest]# nano simplified.c
[root@localhost quest]# su quest
[quest@localhost ~]$ gcc simplified.c -o simpleid
[quest@localhost ~]$ ./simpleid
uid=1001,gid=1001
[quest@localhost ~]$ id
uid=1001(quest) gid=1001(quest) группы=1001(quest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Figure 4.3: рис.3. Компиляция программы `simpleid`, её выполнение и сравнение с командой `id`.

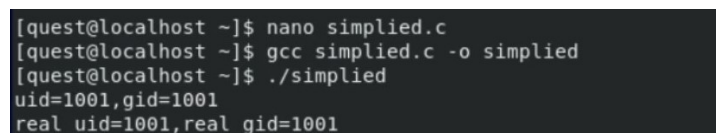
6. Усложнил программу, добавив вывод действительных идентификаторов, получившуюся программу назвал `simpleid2.c` (рис. 4).



```
quest@localhost:~  
Файл Правка Вид Поиск Терминал Справка  
GNU nano 2.9.8 simplified.c Изменён  
  
#include <sys/types.h>  
#include <unistd.h>  
#include <stdio.h>  
  
int  
main()  
{  
    uid_t real_uid=getuid();  
    uid_t uid =geteuid();  
    gid_t real_gid=getgid();  
    gid_t gid=getegid();  
    printf("uid=%d,gid=%d\n",uid,gid);  
    printf("real_uid=%d,real_gid=%d\n",real_uid,real_gid);  
    return 0;  
}
```

Figure 4.4: рис.4. Программа simpleid2.c

7. Скомпилировал и запустил simpleid2.c командами `gcc simpleid2.c -o simpleid2` и `./simpleid2` (рис. 5). Видим, что программа выводит эффективные и действительные идентификаторы пользователя и группы для файла. Видим, что везде это 1001, то есть пользователь quest.



```
[quest@localhost ~]$ nano simplified.c  
[quest@localhost ~]$ gcc simplified.c -o simplified  
[quest@localhost ~]$ ./simplified  
uid=1001,gid=1001  
real_uid=1001,real_gid=1001
```

Figure 4.5: рис.5. Компиляция программы simpleid2, её выполнение.

8-9. От имени суперпользователя выполнил команды: `chown root:quest /home/quest/simpleid2` и `chmod u+s /home/quest/simpleid2`. Временно поменяв свои права с помощью `su` (рис. 6).

С помощью этих команд файлу simpleid2 изменил владельца и группу на root и quest соответственно (`chown`), а также установил на файл SetUID-бит (`chmod`).

10. Выполнил проверку правильности установки новых атрибутов и смены владельца файла simpleid2 командой `ls -l simpleid2` (рис. 6).
11. Запустил simpleid2 и `id` командами `./simpleid2` и `id` (рис. 6). Сравнил результаты: действительные идентификаторы совпадают с выводом

команды `id` - везде 0, то есть рут-пользователь. Так же важно заметить, что эффективные идентификаторы совпадают с действительными.

```
quest@localhost ~]$ su
Пароль:
[root@localhost quest]# chown root:quest /home/quest/simplified
[root@localhost quest]# chmod u+s /home/quest/simplified
[root@localhost quest]# ls -l simplified
-rwsrwxr-x. 1 root quest 17648 ноя  1 14:19 simplified
[root@localhost quest]# ./simplified
uid=0,gid=0
real uid=0,real gid=0
[root@localhost quest]# id
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Figure 4.6: рис.6. Изменение владельца программы и установка SetUID-бита, проверка установки и изменения, запуск программы и команды `id`.

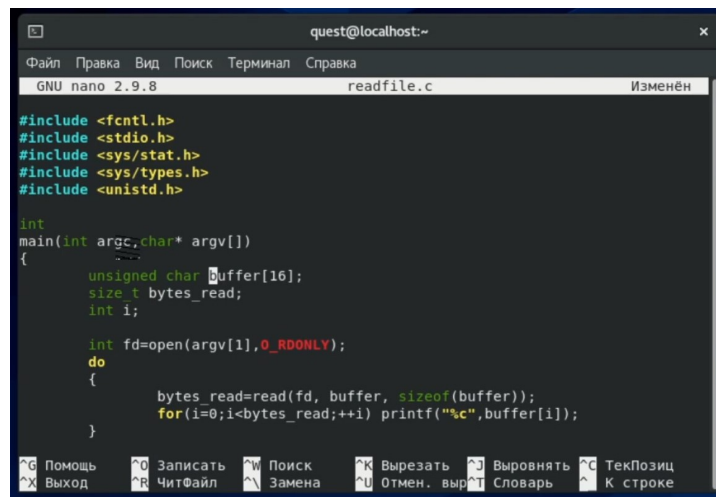
12. Прodelал тоже самое относительно SetGID-бита (рис. 7)

Установка SetGID-бита отражается к команде `ls`, а сравнение выполнения программы и команды `id` дало следующие результаты: действительные идентификаторы совпадают с выводом команды `id` - везде 0, то есть рут-пользователь. Но так же важно заметить, что эффективные идентификаторы отличны от действительных: пользователь - 0, группа - 1001.

```
quest@localhost ~]$ su
Пароль:
[root@localhost quest]# chmod g+s /home/quest/simplified
[root@localhost quest]# ls -l simplified
-rwsrwsr-x. 1 root quest 17648 ноя  1 14:19 simplified
[root@localhost quest]# ./simplified
uid=0,gid=1001
real uid=0,real gid=0
[root@localhost quest]# id
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@localhost quest]#
```

Figure 4.7: рис.7. Установка SetGID-бита, проверка установки, запуск программы и команды `id`.

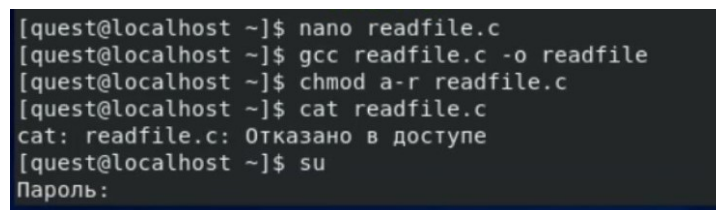
13. Создал программу `readfile.c` (рис. 8).



```
quest@localhost:~  
Файл Правка Вид Поиск Терминал Справка  
GNU nano 2.9.8 readfile.c Изменён  
  
#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 4.8: рис.8. Программа readfile.c

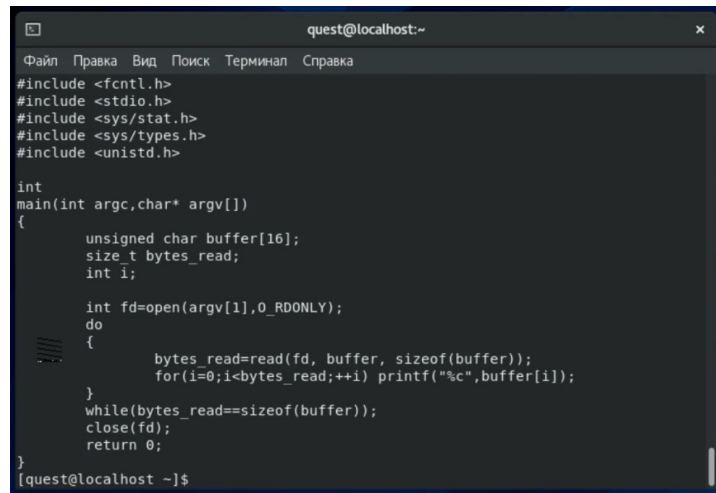
14. Откомпилировал её командой `gcc readfile.c -o readfile` (рис. 9).
15. Сменил владельца у файла `readfile.c` (`chown`) и изменил права так, чтобы только суперпользователь (`root`) мог прочитать его, а `quest` не мог. Использовал `chmod a-r` (рис. 9).
16. Проверил, что пользователь `quest` не может прочитать файл `readfile.c` командой `cat` (рис. 9).
17. Сменил у программы `readfile` владельца и установил SetUID-бит (рис. 9).



```
[quest@localhost ~]$ nano readfile.c  
[quest@localhost ~]$ gcc readfile.c -o readfile  
[quest@localhost ~]$ chmod a-r readfile.c  
[quest@localhost ~]$ cat readfile.c  
cat: readfile.c: Отказано в доступе  
[quest@localhost ~]$ su  
Пароль:
```

Figure 4.9: рис.9. Компиляция readfile и другие действия в соответствии с 14-17 пунктами.

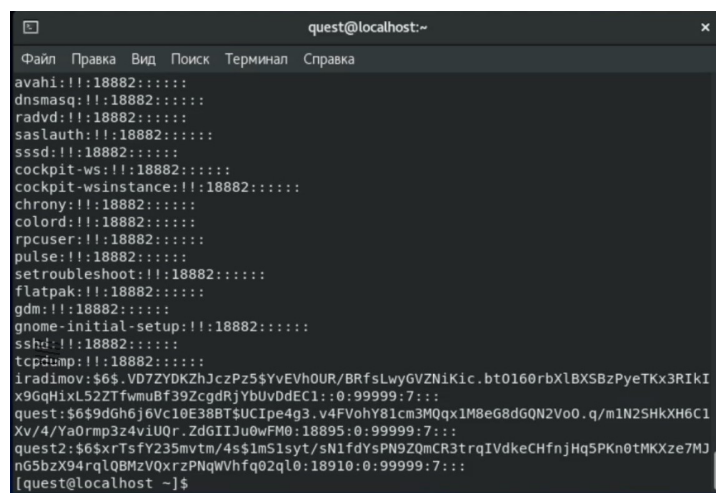
18. Проверил, может ли программа `readfile` прочитать файл `readfile.c`. Да, может (рис. 10).



```
quest@localhost:~  
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;  
}  
[quest@localhost ~]$
```

Figure 4.10: рис.10. Выполнение программы readfile с файлом readfile.c.

19. Проверил, может ли программа readfile прочитать файл /etc/shadow. Её выполнению возможно в том числе, так как владельцем файла является root-пользователь (рис. 11).



```
quest@localhost:~  
Файл Правка Вид Поиск Терминал Справка  
avahi:!:18882:~::~:  
dnsmasq:!:18882:~::~:  
radvd:!:18882:~::~:  
saslauth:!:18882:~::~:  
sssd:!:18882:~::~:  
cockpit-ws:!:18882:~::~:  
cockpit-wsinstance:!:18882:~::~:  
chrony:!:18882:~::~:  
colord:!:18882:~::~:  
rpcuser:!:18882:~::~:  
pulse:!:18882:~::~:  
setroubleshoot:!:18882:~::~:  
flatpak:!:18882:~::~:  
gdm:!:18882:~::~:  
gnome-initial-setup:!:18882:~::~:  
sshd:!:18882:~::~:  
tcpdump:!:18882:~::~:  
lradiomov:$6$,VD7ZYDKZhJczPz5$YvEVh0UR/BRfsLwyGVZniKic.bt0160rbXlBXSbZPyeTkx3RIKI  
x9GqHixL52ZTfwmUBf39ZcgdRjYbUvDdEC1:!:0:99999:7:~::~:  
quest:$6$9dGh6j6Vc10E38BT$UCIpe4g3.v4FVohY81cm3MQqx1M8eG8dGQN2Vo0.q/m1N25HkXH6C1  
Xv/4/Ya0rmp3z4viUQr.ZdGIIJu0wFM0:18895:0:99999:7:~::~:  
quest2:$6$xrTsFY235mvtm/4s$1m51syT/sN1fdYsPN9ZQmCR3trqIVdkeCHfnjHq5PKn0tMKXze7MJ  
nG5bzX94rqlQBMzVQxrzPNqWVhfq02ql0:18910:0:99999:7:~::~:  
[quest@localhost ~]$
```

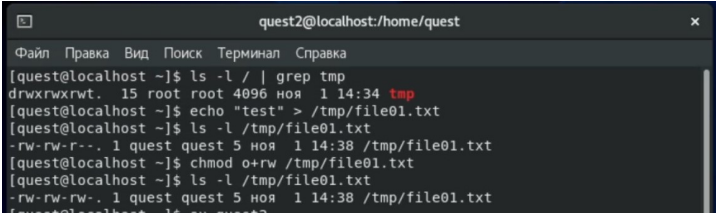
Figure 4.11: рис.11. Выполнение программы readfile с файлом /etc/shadow.

4.3 Исследование Sticky-бита

1. Выяснил, установлен ли атрибут Sticky на директории /tmp, для чего выполнил команду `ls -l / | grep tmp` (рис. 12). Видим, что установлен, так

как есть буква t.

2. От имени пользователя quest создал файл file01.txt в директории /tmp со словом test командой `echo "test" > /tmp/file01.txt` (рис. 12).
3. Просмотрел атрибуты у только что созданного файла и разрешил чтение и запись для категории пользователей «все остальные». Выполнил команды `ls -l /tmp/file01.txt` и `chmod o+rw /tmp/file01.txt` и `ls -l /tmp/file01.txt` (рис. 12).
4. От пользователя quest2 (не являющегося владельцем) попробовал прочитать файл /tmp/file01.txt командой `cat /tmp/file01.txt` (рис. 12).



```
quest2@localhost/home/quest
Файл Правка Вид Поиск Терминал Справка
[quest@localhost ~]$ ls -l / | grep tmp
drwxrwxrwt. 15 root root 4096 ноя  1 14:34 tmp
[quest@localhost ~]$ echo "test" > /tmp/file01.txt
[quest@localhost ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 quest quest 5 ноя  1 14:38 /tmp/file01.txt
[quest@localhost ~]$ chmod o+rw /tmp/file01.txt
[quest@localhost ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 quest quest 5 ноя  1 14:38 /tmp/file01.txt
```

Figure 4.12: рис.12. Выполнение пунктов 1-4 исследования Sticky-бита .

5. От пользователя quest2 попробовал дозаписать в файл /tmp/file01.txt слово test2 командой `echo "test2" >> /tmp/file01.txt`. Мне удалось выполнить операцию (рис. 13).
6. Проверил содержимое файла командой `cat /tmp/file01.txt` (рис. 13).
7. От пользователя quest2 попробовал записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию командой `echo "test3" > /tmp/file01.txt`. Мне удалось выполнить операцию (рис. 13).
8. Проверил содержимое файла командой `cat /tmp/file01.txt` (рис. 13).
9. От пользователя quest2 попробовал удалить файл /tmp/file01.txt командой `rm /tmp/file01.txt`. Мне не удалось удалить файл (рис. 13).

```

[quest@localhost ~]$ su quest2
Пароль:
[quest2@localhost quest]$ cat /tmp/file01.txt
test
[quest2@localhost quest]$ echo "test2" > /tmp/file01.txt
[quest2@localhost quest]$ cat /tmp/file01.txt
test2
[quest2@localhost quest]$ echo "test3" > /tmp/file01.txt
[quest2@localhost quest]$ cat /tmp/file01.txt
test3
[quest2@localhost quest]$ rm /tmp/file01.txt
rm: невозможно удалить '/tmp/file01.txt': Операция не позволена
[quest2@localhost quest]$ su
Пароль:

```

Figure 4.13: рис.13. Выполнение пунктов 5-9 исследования Sticky-бита .

Можем сделать вывод, что разрешена дозапись в файл, запись в файл, но мы не можем удалять файл из директории, на которую установлен атрибут Sticky.

10. Повысил свои права до суперпользователя следующей командой `su -` и выполнил после этого команду, снимающую атрибут `t` (Sticky-бит) с директории `/tmp`: `chmod -t /tmp` (рис. 14).
11. Покинул режим суперпользователя командой `exit` (рис. 14).
12. От пользователя `quest2` проверил, что атрибута `t` у директории `/tmp` нет: `ls -l / | grep tmp` (рис. 14).
13. Повторил предыдущие шаги (рис. 14). Видим, что дозапись и запись так же разрешены, но при этом удалось и удалить файл.

```

quest2@localhost/home/quest
Файл  Правка  Вид  Поиск  Терминал  Справка
test3
[quest2@localhost quest]$ rm /tmp/file01.txt
rm: невозможно удалить '/tmp/file01.txt': Операция не позволена
[quest2@localhost quest]$ su
Пароль:
[root@localhost quest]# chmod -t /tmp
[root@localhost quest]# exit
exit
[quest2@localhost quest]$ ls -l / | grep tmp
drwxrwxrwx. 15 root root 4096 ноя 1 14:42 tmp
[quest2@localhost quest]$ cat /tmp/file01.txt
test3
[quest2@localhost quest]$ echo "test2" > /tmp/file01.txt
[quest2@localhost quest]$ cat /tmp/file01.txt
test2
[quest2@localhost quest]$ cat /tmp/file01.txt
test2
[quest2@localhost quest]$ rm /tmp/file01.txt
[quest2@localhost quest]$ su
Пароль:
[root@localhost quest]# chmod +t /tmp
[root@localhost quest]# exit
exit
[quest2@localhost quest]$

```

Figure 4.14: рис.14. Выполнение пунктов 10-13 исследования Sticky-бита .

14. Мне удалось удалить файл от имени пользователя, не являющегося его владельцем.
15. Повысил свои права до суперпользователя и верните атрибут `t` на директорию `/tmp`: `su -`, `chmod +t /tmp`, `exit` (рис. 15).

```
[root@localhost quest]# chmod +t /tmp
[root@localhost quest]# exit
exit
[quest2@localhost quest]$
```

Figure 4.15: рис.15. Возвращение атрибута `t` на директорию.

5 Библиография

1. ТУИС РУДН
2. https://help.ubuntu.ru/wiki/стандартные_права_unix

6 Выводы

Я изучил механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получил практические навыки работы в консоли с дополнительными атрибутами. Рассмотрел работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.