

Лабораторная работа № 5

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

Абдуллаев Сайидазизхон Шухратович

Содержание

Цель работы	5
Задание	6
Теоретическое введение	7
Ход работы	9
Выводы	18

Список иллюстраций

1	Листинг simpleid.c	9
2	Запуск и сверка simpleid.c	10
3	Листинг simpleid2.c	10
4	Запуск simpleid2.c	11
5	Смена владельца и добавление SetUID бита, запуск и сверка simpleid2.c	11
6	Добавление SetGID бита	11
7	Запуск simpleid2.c	12
8	Листинг readfile.c	12
9	Компиляция	13
10	Смена владельца readfile.c	13
11	Проверка на cat из-под guest'a	13
12	Смена владельца readfile.c	14
13	Чтение readfile.c программой readfile	14
14	Чтение /etc/shadow программой readfile	15
15	Создание файла, правка прав файла	15
16	Проверка прав, тестирование, и попытка удаления	16
17	Удаление Sticky-бита и повторное тестирование	16
18	Возвращение Sticky-бита	17

Список таблиц

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

Закрепить дискреционное разграничение прав в Linux с дополнительными атрибутами.

Теоретическое введение

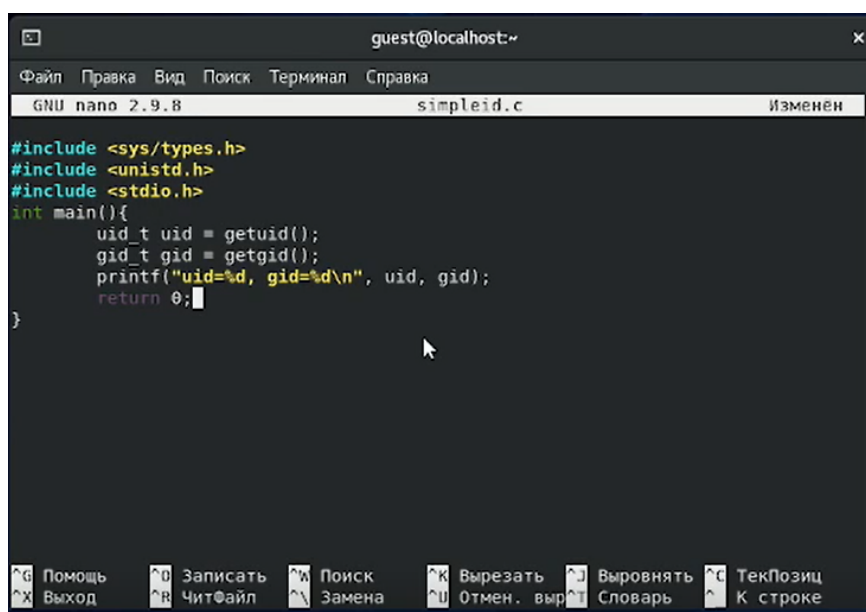
В Linux, как и в любой многопользовательской системе, абсолютно естественным образом возникает задача разграничения доступа субъектов — пользователей к объектам — файлам дерева каталогов. Один из подходов к разграничению доступа — так называемый дискреционный — предполагает назначение владельцев объектов, которые по собственному усмотрению определяют права доступа субъектов (других пользователей) к объектам (файлам), которыми владеют. Дискреционные механизмы разграничения доступа используются для разграничения прав доступа процессов как обычных пользователей, так и для ограничения прав системных программ в (например, служб операционной системы), которые работают от лица псевдопользовательских учетных записей. Чтобы получить доступ к файлам в Linux, используются разрешения. Эти разрешения назначаются трем объектам: файлу, группе и другому объекту. Для управления правами используется команда `chmod`. При использовании `chmod` в относительном режиме вы работаете с тремя индикаторами, чтобы указать, что вы хотите сделать. Сначала вы указываете, для кого вы хотите изменить разрешения. Для этого вы можете выбрать между пользователем (u), группой (g) и другими (o). Затем вы

используете оператор для добавления или удаления разрешений из текущего режима или устанавливаете их абсолютно. В конце вы используете `r(read)`, `w(write)` и `x(execute)`, чтобы указать, какие разрешения вы хотите установить. При использовании `chmod` вы можете устанавливать разрешения для пользователя (user), группы (group) и других (other). Помимо основных

разрешений, о которых вы только что прочитали, в Linux также есть набор расширенных разрешений. Это не те разрешения, которые вы устанавливаете по умолчанию, но иногда они предоставляют полезное дополнение.

Ход работы

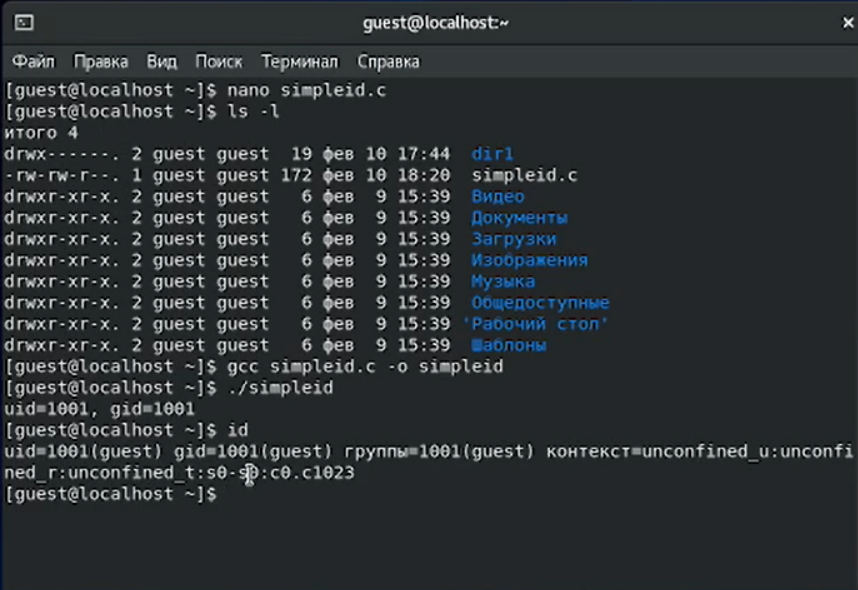
1. Готовим систему и входим из-под пользователя guest. Пишем программу simpleid.c. Компилируем программу, запускаем, видим вывод uid и gid пользователя, сравниваем вывод с id (все совпадает). (Рис. 1, 2).



```
guest@localhost:~  
Файл Правка Вид Поиск Терминал Справка  
GNU nano 2.9.8 simpleid.c изменен  
  
#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;  
}
```

^G Помощь ^O Записать ^W Поиск ^K Вырезать ^J Выводить ^C ТекПозиц
^X Выход ^R ЧитФайл ^\ Замена ^U Отмен. выр ^T Словарь ^_ К строке

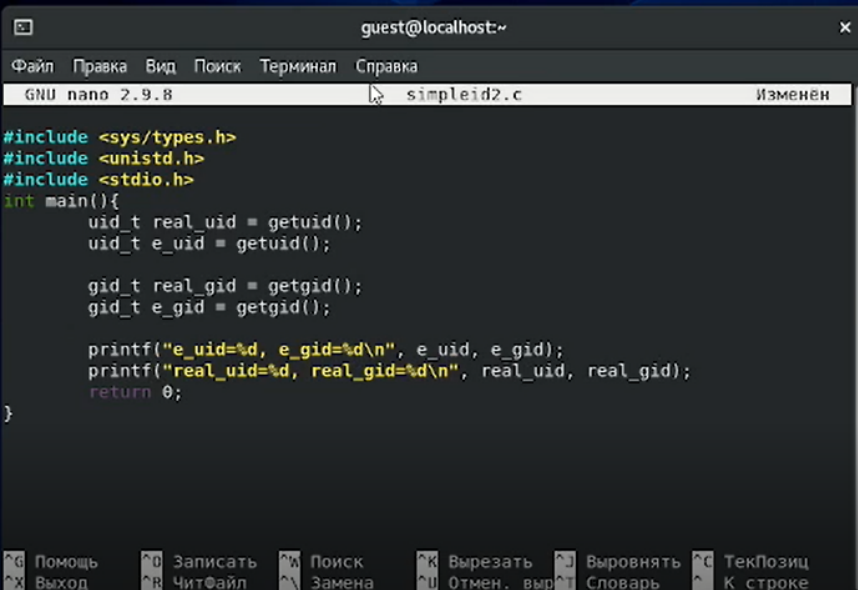
Рис. 1: Листинг simpleid.c



```
guest@localhost:~  
Файл Правка Вид Поиск Терминал Справка  
[guest@localhost ~]$ nano simpleid.c  
[guest@localhost ~]$ ls -l  
итого 4  
drwx----- 2 guest guest 19 фев 10 17:44 dir1  
-rw-rw-r-- 1 guest guest 172 фев 10 18:20 simpleid.c  
drwxr-xr-x 2 guest guest 6 фев 9 15:39 Видео  
drwxr-xr-x 2 guest guest 6 фев 9 15:39 Документы  
drwxr-xr-x 2 guest guest 6 фев 9 15:39 Загрузки  
drwxr-xr-x 2 guest guest 6 фев 9 15:39 Изображения  
drwxr-xr-x 2 guest guest 6 фев 9 15:39 Музыка  
drwxr-xr-x 2 guest guest 6 фев 9 15:39 Общедоступные  
drwxr-xr-x 2 guest guest 6 фев 9 15:39 'Рабочий стол'  
drwxr-xr-x 2 guest guest 6 фев 9 15:39 Шаблоны  
[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) контекст=unconfined_u:unconfi  
ned_r:unconfined_t:s0-s0:c0.c1023  
[guest@localhost ~]$
```

Рис. 2: Запуск и сверка simpleid.c

2. Усложняем программу и запускаем её. (Рис. 3, 4).



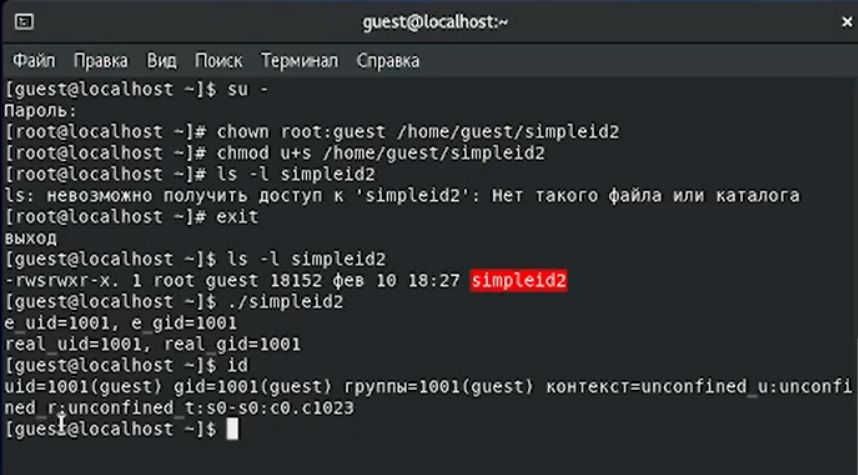
```
guest@localhost:~  
Файл Правка Вид Поиск Терминал Справка  
GNU nano 2.9.8 simpleid2.c Изменен  
#include <sys/types.h>  
#include <unistd.h>  
#include <stdio.h>  
int main(){  
    uid_t real_uid = getuid();  
    uid_t e_uid = getuid();  
  
    gid_t real_gid = getgid();  
    gid_t e_gid = getgid();  
  
    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. 3: Листинг 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 ~]$
```

Рис. 4: Запуск simpleid2.c

3. Из-под суперпользователя меняем владельца и добавляем SetUID бит на файл. Проверяем правильность и запускаем программу еще раз. `euid` возвращает `id` владельца, а `real_uid` возвращает `uid` запускающего пользователя. (Рис. 5).



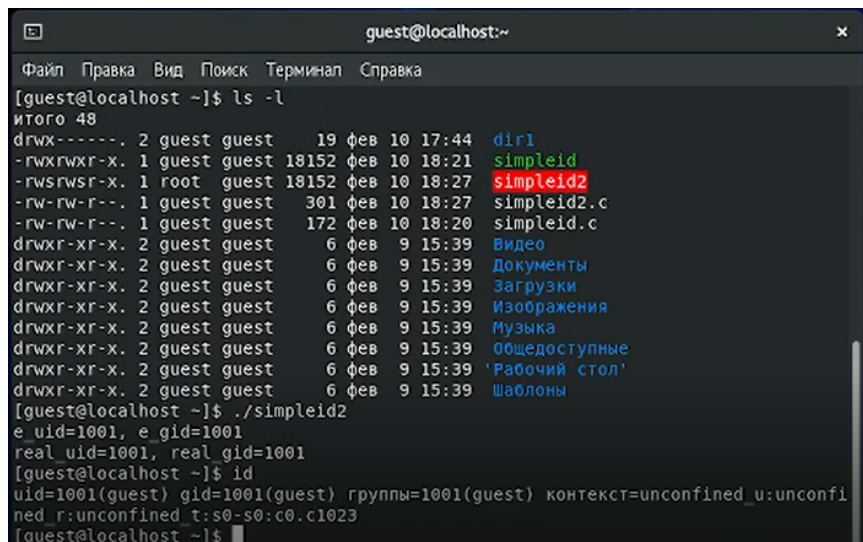
```
guest@localhost:~
Файл Правка Вид Поиск Терминал Справка
[guest@localhost ~]$ su -
Пароль:
[root@localhost ~]# chown root:guest /home/guest/simpleid2
[root@localhost ~]# chmod u+s /home/guest/simpleid2
[root@localhost ~]# ls -l simpleid2
ls: невозможно получить доступ к 'simpleid2': Нет такого файла или каталога
[root@localhost ~]# exit
выход
[guest@localhost ~]$ ls -l simpleid2
-rwsrwxr-x. 1 root guest 18152 фев 10 18:27 simpleid2
[guest@localhost ~]$ ./simpleid2
e uid=1001, e_gid=1001
real uid=1001, real gid=1001
[guest@localhost ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi
ned_r:unconfined_t:s0-s0:c0.c1023
[guest@localhost ~]$
```

Рис. 5: Смена владельца и добавление SetUID бита, запуск и сверка simpleid2.c

4. Теперь добавим на файл SetGID бит с проделаем все то же самое. (Рис. 6, 7).

```
[guest@localhost ~]$ su -
Пароль:
[root@localhost ~]# chmod g+s /home/guest/simpleid2
[root@localhost ~]# exit
выход
[guest@localhost ~]$
```

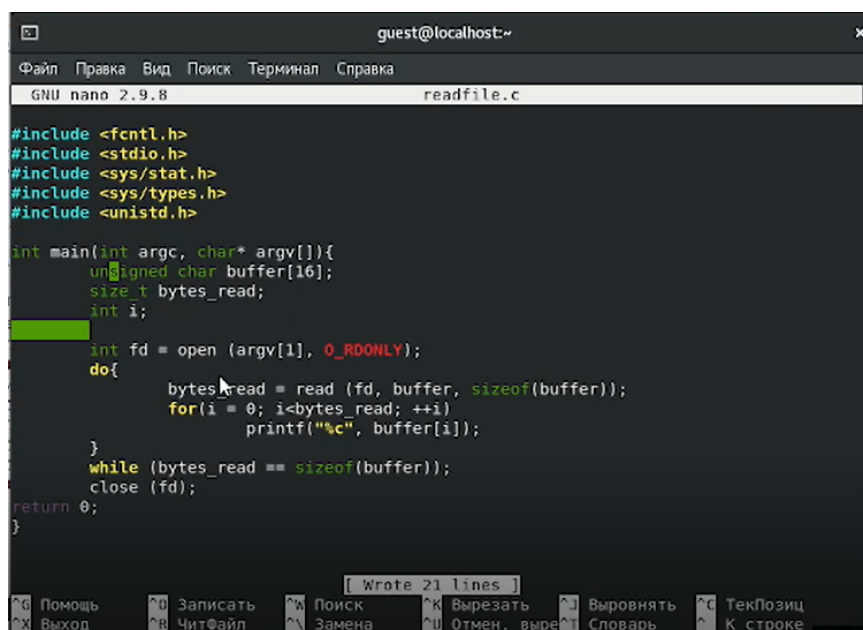
Рис. 6: Добавление SetGID бита



```
guest@localhost:~  
[guest@localhost ~]$ ls -l  
итого 48  
drwx----- 2 guest guest 19 фев 10 17:44 dir1  
-rwxrwxr-x. 1 guest guest 18152 фев 10 18:21 simpleid  
-rwsrwsr-x. 1 root guest 18152 фев 10 18:27 simpleid2  
-rw-rw-r-- 1 guest guest 301 фев 10 18:27 simpleid2.c  
-rw-rw-r-- 1 guest guest 172 фев 10 18:20 simpleid.c  
drwxr-xr-x. 2 guest guest 6 фев 9 15:39 Видео  
drwxr-xr-x. 2 guest guest 6 фев 9 15:39 Документы  
drwxr-xr-x. 2 guest guest 6 фев 9 15:39 Загрузки  
drwxr-xr-x. 2 guest guest 6 фев 9 15:39 Изображения  
drwxr-xr-x. 2 guest guest 6 фев 9 15:39 Музыка  
drwxr-xr-x. 2 guest guest 6 фев 9 15:39 Общедоступные  
drwxr-xr-x. 2 guest guest 6 фев 9 15:39 'Рабочий стол'  
drwxr-xr-x. 2 guest guest 6 фев 9 15:39 Шаблоны  
[guest@localhost ~]$ ./simpleid2  
e_uid=1001, e_gid=1001  
real_uid=1001, real_gid=1001  
[guest@localhost ~]$ id  
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi  
ned_r:unconfined_t:s0-s0:c0.c1023  
[guest@localhost ~]$
```

Рис. 7: Запуск simpleid2.c

5. Пишем программу readfile.c (Рис. 8).



```
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);  
    return 0;  
}
```

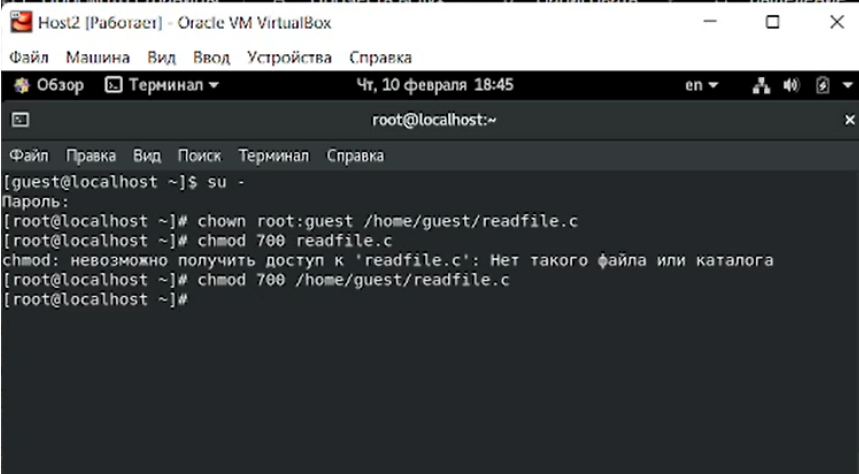
Рис. 8: Листинг readfile.c

6. Компилируем программу. (Рис. 9).

```
[guest@localhost ~]$ nano readfile.c
[guest@localhost ~]$ gcc readfile.c -o readfile
[guest@localhost ~]$
```

Рис. 9: Компиляция

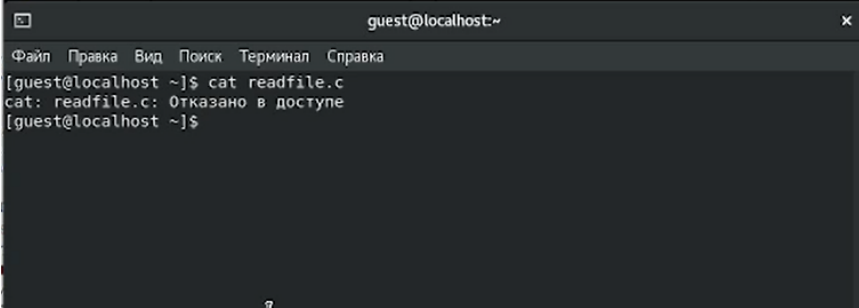
7. Меняем владельца у файла readfile.c и запрещаем чтение всем, кроме суперпользователя. Проверяем, что guest не может читать. Меняем владельца у программы readfile и добавляем SetUID бит на неё. (Рис. 10, 11, 12).



The screenshot shows a terminal window titled "Host2 [Работает] - Oracle VM VirtualBox". The terminal is running as root@localhost. The user guest@localhost has entered the command "su -" and provided a password. The root user then executes the following commands: "chown root:guest /home/guest/readfile.c", "chmod 700 readfile.c", and "chmod 700 /home/guest/readfile.c". The output of the second command shows an error: "chmod: невозможно получить доступ к 'readfile.c': Нет такого файла или каталога".

```
root@localhost:~# chown root:guest /home/guest/readfile.c
root@localhost:~# chmod 700 readfile.c
chmod: невозможно получить доступ к 'readfile.c': Нет такого файла или каталога
root@localhost:~# chmod 700 /home/guest/readfile.c
root@localhost:~#
```

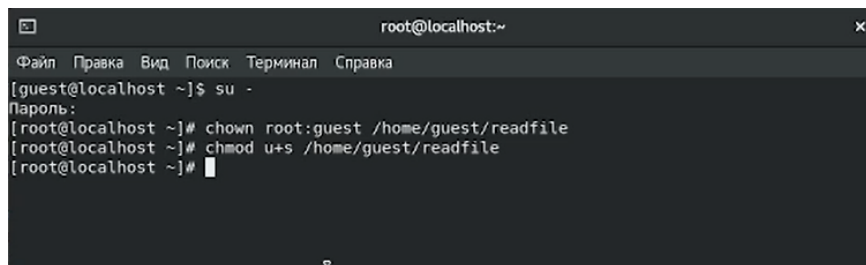
Рис. 10: Смена владельца readfile.c



The screenshot shows a terminal window titled "guest@localhost:~". The user guest@localhost has entered the command "cat readfile.c". The output of the command is "cat: readfile.c: Отказано в доступе", indicating that the guest user does not have permission to read the file.

```
guest@localhost:~$ cat readfile.c
cat: readfile.c: Отказано в доступе
guest@localhost:~$
```

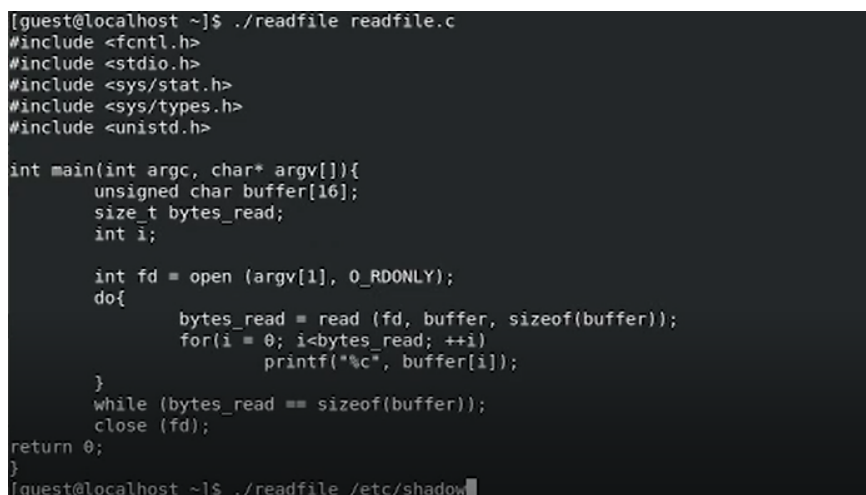
Рис. 11: Проверка на cat из-под guest'a



```
root@localhost:~  
Файл Правка Вид Поиск Терминал Справка  
[guest@localhost ~]$ su -  
Пароль:  
[root@localhost ~]# chown root:guest /home/guest/readfile  
[root@localhost ~]# chmod u+s /home/guest/readfile  
[root@localhost ~]#
```

Рис. 12: Смена владельца readfile.c

8. Проверяем, может ли программа readfile прочитать файл readfile.c и файл /etc/shadow. Да, может. Хотя сам пользователь вручную не мог. Всё дело в том, что при вызове программы права пользователя повышаются SetUID битом до прав владельца, который может читать файлы (суперпользователь в нашем случае). (Рис. 13, 14).



```
[guest@localhost ~]$ ./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]);  
    }  
    while (bytes_read == sizeof(buffer));  
    close (fd);  
    return 0;  
}  
[guest@localhost ~]$ ./readfile /etc/shadow
```

Рис. 13: Чтение readfile.c программой readfile

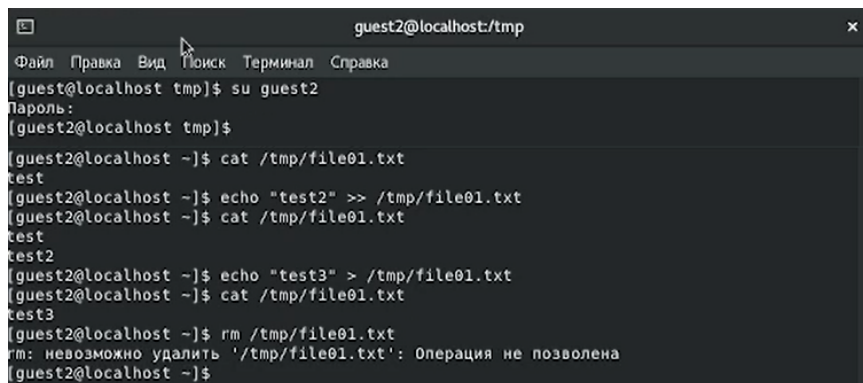
```
guest@localhost:~  
Файл Правка Вид Поиск Терминал Справка  
cluster:!!:19031:.....  
rpc:!!:19031:0:99999:7:..  
chrony:!!:19031:.....  
avahi:!!:19031:.....  
saslauth:!!:19031:.....  
libstoragemgmt:!!:19031:.....  
dnsmasq:!!:19031:.....  
radvd:!!:19031:.....  
sssd:!!:19031:.....  
cockpit-ws:!!:19031:.....  
cockpit-wsinstance:!!:19031:.....  
colord:!!:19031:.....  
rpcuser:!!:19031:.....  
setroubleshoot:!!:19031:.....  
flatpak:!!:19031:.....  
gdm:!!:19031:.....  
clevi:!!:19031:.....  
gnome-initial-setup:!!:19031:.....  
design:!!:19031:.....  
tcpdump:!!:19031:.....  
sshd:!!:19031:.....  
ssabdullaev:$6$w2huP9pLTnpEeJG4$NXFbzN9sniGr9F755PKghe058l0q0qxYI34htYDJrylcw06cHFXr4  
m0q/Ljo9qZF3jH00h.xHPRUYHtzQr.:.:0:99999:7:..  
guest:$6$b2hqXKmkT5/6VnxNSW5iwH/5dhz7dZxXUMuMd0FdRREcZa3mkS.x7kANZniW09FqbV7pkg6sL6TF61  
YyZ44LSSraUeNbWjg0Jid7o/:19032:0:99999:7:..  
guest2:$6$UqULlg7dy8qHF0y$Sac0W0AtDn0qW59jEmi8y0Up4vwzKS2C7K086JBf5ic5myEXuo/0jvwyu/Cl  
wR8S8Yu8HShvjUvml/cpEeEF1:19033:0:99999:7:..  
guest@localhost ~$
```

Рис. 14: Чтение /etc/shadow программой readfile

9. Проверяем Sticky бит. Для этого создаем файл, которому даем gw права для others и пишем туда слово test. Теперь пробуем выполнить дозапись в файл, перезапись файла и его удаление. Всё, кроме удаления, прошло успешно. (Рис. 15, 16).

```
guest@localhost:/tmp  
Файл Правка Вид Поиск Терминал Справка  
[guest@localhost ~]$ ls -l | grep tmp  
[guest@localhost ~]$ ls -l / | grep tmp  
drwxrwxrwt. 18 root root 4096 фев 10 18:47 tmp  
[guest@localhost ~]$ cd /tmp  
[guest@localhost tmp]$ touch file01.txt  
[guest@localhost tmp]$ echo "test" > /tmp/file01.txt  
[guest@localhost tmp]$ ls -l file01.txt  
-rw-rw-r--. 1 guest guest 5 фев 10 18:52 file01.txt  
[guest@localhost tmp]$ chmod o+rw file01.txt  
[guest@localhost tmp]$ ls -l file01.txt  
-rw-rw-rw-. 1 guest guest 5 фев 10 18:52 file01.txt  
[guest@localhost tmp]$
```

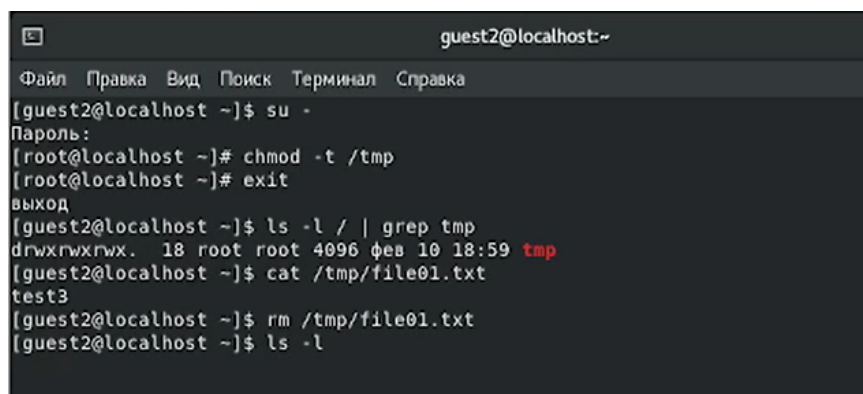
Рис. 15: Создание файла, правка прав файла



```
guest2@localhost:/tmp
Файл Правка Вид Поиск Терминал Справка
[guest2@localhost tmp]$ su guest2
Пароль:
[guest2@localhost tmp]$
[guest2@localhost ~]$ cat /tmp/file01.txt
test
[guest2@localhost ~]$ echo "test2" >> /tmp/file01.txt
[guest2@localhost ~]$ cat /tmp/file01.txt
test
test2
[guest2@localhost ~]$ echo "test3" > /tmp/file01.txt
[guest2@localhost ~]$ cat /tmp/file01.txt
test3
[guest2@localhost ~]$ rm /tmp/file01.txt
rm: невозможно удалить '/tmp/file01.txt': Операция не позволена
[guest2@localhost ~]$
```

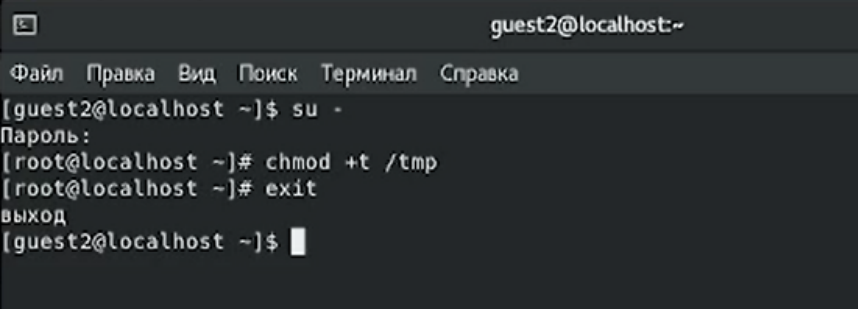
Рис. 16: Проверка прав, тестирование, и попытка удаления

10. Повышаем права до суперпользователя и удаляем Sticky-бит с папки /tmp. Повторяем наши тесты. Теперь прошли все команды, включая удаление файла. Таким образом, пользователь, не являющийся владельцем файла, смог его удалить, так как Sticky-бит не был настроен. Возвращаем Sticky-бит на папку /tmp. (Рис. 17, 18).



```
guest2@localhost:~
Файл Правка Вид Поиск Терминал Справка
[guest2@localhost ~]$ su -
Пароль:
[root@localhost ~]# chmod -t /tmp
[root@localhost ~]# exit
выход
[guest2@localhost ~]$ ls -l / | grep tmp
drwxrwxrwx. 18 root root 4096 фев 10 18:59 tmp
[guest2@localhost ~]$ cat /tmp/file01.txt
test3
[guest2@localhost ~]$ rm /tmp/file01.txt
[guest2@localhost ~]$ ls -l
```

Рис. 17: Удаление Sticky-бита и повторное тестирование

A terminal window titled "guest2@localhost:~" with a menu bar containing "Файл", "Правка", "Вид", "Поиск", "Терминал", and "Справка". The terminal shows the following commands and output:

```
[guest2@localhost ~]$ su -  
Пароль:  
[root@localhost ~]# chmod +t /tmp  
[root@localhost ~]# exit  
выход  
[guest2@localhost ~]$
```

Рис. 18: Возвращение Sticky-бита

Выводы

В результате выполнения данной работы были практические навыки работы в консоли с расширенными атрибутами файлов.