

Лабораторная работа №5

Дисциплина: Информационная безопасность

Боровикова Карина Владимировна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	19
	Список литературы	20

Список иллюстраций

4.1	Входим в систему от имени пользователя guest, проверяем содержимое файла simpleid.c	8
4.2	Содержимое файла simpleid.c	9
4.3	Содержимое файла simpleid.c и компиляция simpleid2	10
4.4	Меняем владельца файла и права доступа на него	11
4.5	Вывод команд simpleid2 и id	11
4.6	Делаем то же самое для SetGID-бита	12
4.7	Содержимое readfile.c	13
4.8	Компиляция, смена владельца и прав доступа	14
4.9	Смена прав доступа	14
4.10	Смена прав доступа и владельца	14
4.11	Программа readfile читает файл readfile.c	15
4.12	Программа readfile читает файл /etc/shadow	15
4.13	Записываем текст в файл	16
4.14	Записываем текст в файл	16
4.15	Выполняем действия с файлом file01.txt	17
4.16	Выполняем действия с файлом file01.txt после снятия Sticky-бита	18
4.17	Возвращаем на место	18

Список таблиц

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

- Произвести работу в консоли с учетных записей суперпользователя и гостя;
- Опытным путем проверить работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

3 Теоретическое введение

Для выполнения четвёртой части задания вам потребуются навыки программирования, а именно, умение компилировать простые программы, написанные на языке C (C++), используя интерфейс CLI. Само по себе создание программ не относится к теме, по которой выполняется работа, а является вспомогательной частью, позволяющей увидеть, как реализуются на практике те или иные механизмы дискреционного разграничения доступа. Если при написании (или исправлении существующих) скриптов на `bash`-е у большинства системных администраторов не возникает проблем, то процесс компилирования, как показывает практика, вызывает необоснованные затруднения

4 Выполнение лабораторной работы

1. Войдите в систему от имени пользователя guest. (рис. 4.1)
2. Создайте программу simpleid.c: (рис. 4.1)

```
#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;
}
```

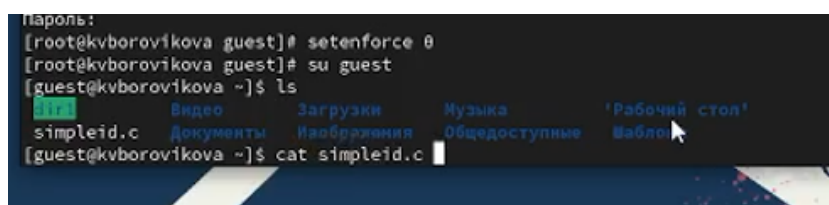
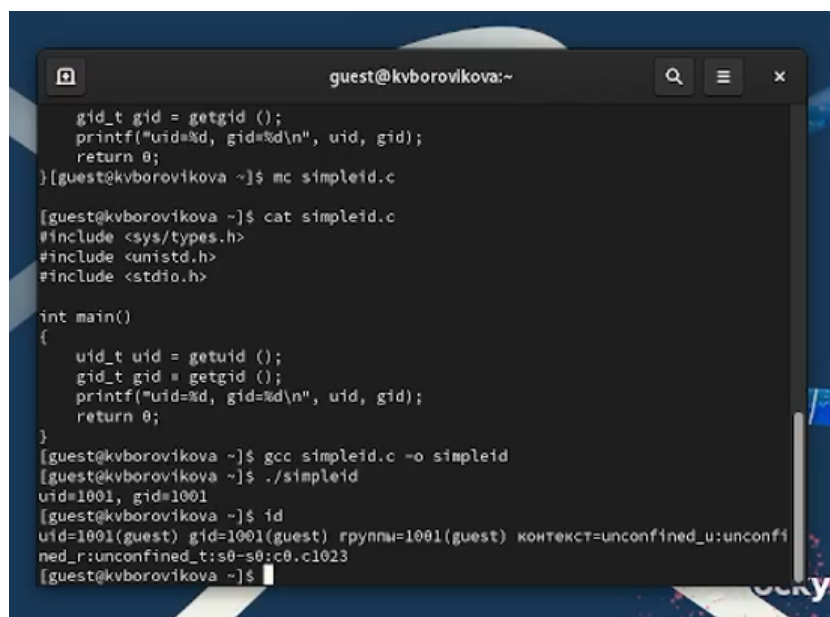


Рис. 4.1: Входим в систему от имени пользователя guest, проверяем содержимое файла simpleid.c



```
guest@kvborovikova:~  
gid_t gid = getgid ();  
printf("uid=%d, gid=%d\n", uid, gid);  
return 0;  
}[guest@kvborovikova ~]$ mc simpleid.c  
[guest@kvborovikova ~]$ cat simpleid.c  
#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;  
}  
[guest@kvborovikova ~]$ gcc simpleid.c -o simpleid  
[guest@kvborovikova ~]$ ./simpleid  
uid=1001, gid=1001  
[guest@kvborovikova ~]$ id  
uid=1001(guest) gid=1001(guest) rpyнны=1001(guest) контекст=unconfined_u:unconfi  
ned_r:unconfined_t:s0-s0:c0.c1023  
[guest@kvborovikova ~]$
```

Рис. 4.2: Содержимое файла simpleid.c

3. Скомпилируйте программу и убедитесь, что файл программы создан: (рис. 4.2) `gcc simpleid.c -o simpleid`
4. Выполните программу simpleid: (рис. 4.2) `./simpleid`
5. Выполните системную программу id: (рис. 4.2) id видим, что выведенные данные совпадают
6. Усложните программу, добавив вывод действительных идентификаторов: (рис. 4.3)

```
#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;
}

```

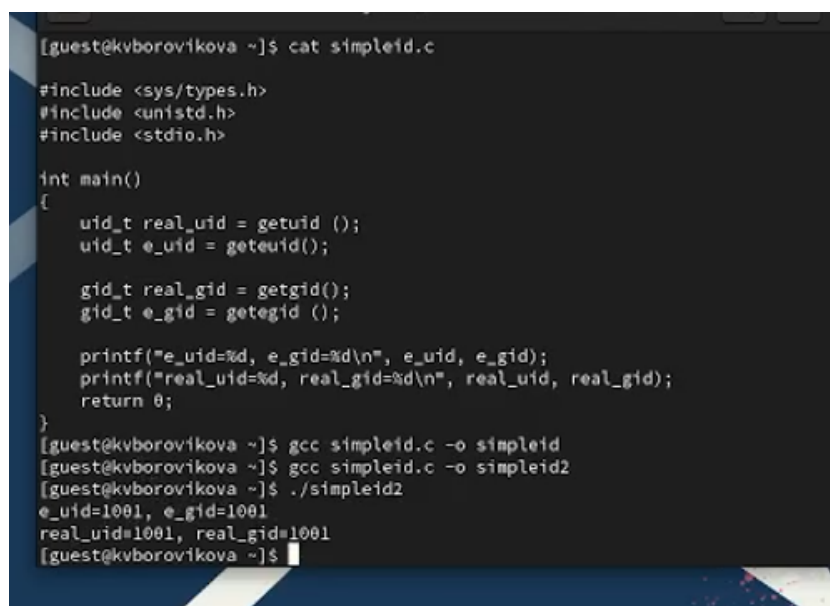
Получившуюся программу назовите simpleid2.c.

7. Скомпилируйте и запустите simpleid2.c: (рис. 4.3)

```

gcc simpleid2.c -o simpleid2
./simpleid2

```



```

[guest@kvborovikova ~]$ cat simpleid.c
#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;
}
[guest@kvborovikova ~]$ gcc simpleid.c -o simpleid
[guest@kvborovikova ~]$ gcc simpleid.c -o simpleid2
[guest@kvborovikova ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@kvborovikova ~]$

```

Рис. 4.3: Содержимое файла simpleid.c и компиляция simpleid2

8. От имени суперпользователя выполните команды: (рис. 4.4)

```

chown root:guest /home/guest/simpleid2
chmod u+s /home/guest/simpleid2

```

```
[guest@kvborovikova ~]$ su
Пароль:
[root@kvborovikova guest]# chown root:guest /home/guest/simpleid2
[root@kvborovikova guest]# chmod u+s /home/guest/simpleid2
[root@kvborovikova guest]# ls -l
итого 60
drwxrwxrwx. 2 guest guest 19 сен 30 15:02 .
-rwxr-xr-x. 1 guest guest 25960 окт 7 14:09 simpleid
-rwsr-xr-x. 1 root guest 26064 окт 7 14:07 simpleid2
-rw-r--r--. 1 guest guest 193 окт 7 14:09 simpleid.c
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Видео
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Документы
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Загрузки
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Изображения
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Музыка
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Общедоступные
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 'Рабочий стол'
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Шаблоны
[root@kvborovikova guest]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@kvborovikova guest]#
```

Рис. 4.4: Меняем владельца файла и права доступа на него

9. Используйте `sudo` или повысьте временно свои права с помощью `su`. Данные команды меняют владельца файла и права доступа на него.
10. Выполните проверку правильности установки новых атрибутов и смены владельца файла `simpleid2`: (рис. 4.4) `ls -l simpleid2`
11. Запустите `simpleid2` и `id`: (рис. 4.4 - 4.5) `./simpleid2 id` Видим, что команды выдали различные данные, `e_uid = 0`.

```
[root@kvborovikova guest]# id
uid=0(root) gid=0(root) rгруппы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@kvborovikova guest]# su guest
[guest@kvborovikova ~]$ ls
simpleid  simpleid.c  Загрузки  Общедоступные
simpleid  Видео        Изображения  'Рабочий стол'
simpleid2  Документы    Музыка      Шаблоны
[guest@kvborovikova ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@kvborovikova ~]$ id
uid=1001(guest) gid=1001(guest) rгруппы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@kvborovikova ~]$
```

Рис. 4.5: Вывод команд `simpleid2` и `id`

12. Прodelайте тоже самое относительно SetGID-бита. (рис. 4.6)

```
guest@kvborovikova:~  
[root@kvborovikova guest]# chown root:root /home/guest/simpleid2  
[root@kvborovikova guest]# chmod g+s /home/guest/simpleid2  
[root@kvborovikova guest]# su guest  
[guest@kvborovikova ~]$ ls -l  
итого 60  
drwxrwxrwx. 2 guest guest 19 сен 30 15:02 .  
-rwxr-xr-x. 1 root root 25960 окт 7 14:09 simpleid  
-rwxr-sr-x. 1 root root 26064 окт 7 14:07 simpleid2  
-rw-r--r--. 1 guest guest 193 окт 7 14:09 simpleid.c  
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Видео  
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Документы  
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Загрузки  
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Изображения  
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Музыка  
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Общедоступные  
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 'Рабочий стол'  
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 Шаблоны  
[guest@kvborovikova ~]$ ./simpleid2  
e_uid=1001, e_gid=0  
real_uid=1001, real_gid=1001  
[guest@kvborovikova ~]$ id  
uid=1001(guest) gid=1001(guest) rpyrny=1001(guest) контекст=unconfined_u:unconfi  
ned_r:unconfined_t:s0-s0:c0.c1023  
[guest@kvborovikova ~]$
```

Рис. 4.6: Делаем то же самое для SetGID-бита

Видим, что команды выдали различные данные, `e_gid = 0`.

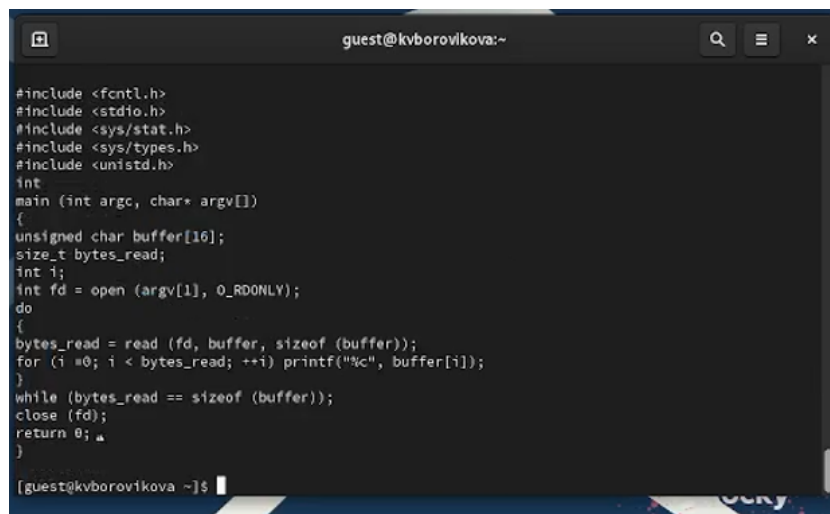
13. Создайте программу `readfile.c`: (рис. 4.7)

```
#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@kvborovikova:~
#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@kvborovikova ~]$

```

Рис. 4.7: Содержимое readfile.c

14. Откомпилируйте её. (рис. 4.8)

```
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.

```
[guest@kvborovikova ~]$ gcc readfile.c -o readfile
[guest@kvborovikova ~]$ su
Пароль:
[root@kvborovikova guest]# chown root:root /home/guest/readfile.c
[root@kvborovikova guest]# chmod u+s /home/guest/readfile.c
chmod: невозможно получить доступ к 'g+s': Нет такого файла или каталога
[root@kvborovikova guest]# chmod u+s /home/guest/readfile.c
[root@kvborovikova guest]# chmod g+s /home/guest/readfile.c
[root@kvborovikova guest]#
```

Рис. 4.8: Компиляция, смена владельца и прав доступа

```
по команде «chmod --help» можно получить дополнительную информацию.
[root@kvborovikova guest]# chmod o-r readfile.c
[root@kvborovikova guest]# ls -l
итого 92
drwxrwxrwx. 2 guest guest 19 сен 30 15:02 .
-rwxr-x--x. 1 guest guest 26008 окт 7 15:30 readfile
-rws--S---. 1 root root 404 окт 7 15:30 readfile.c
-rwxr-xr-x. 1 root root 25960 окт 7 14:09 simpleid
-rwxr-sr-x. 1 root root 26064 окт 7 14:07 simpleid2
-rw-r--r--. 1 guest guest 193 окт 7 14:00 simpleid.c
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 видео
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 документы
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 загрузки
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 изображения
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 музыка
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 общедоступные
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 'Рабочий стол'
drwxr-xr-x. 2 guest guest 6 сен 16 15:19 шаблоны
[root@kvborovikova guest]# su guest
[guest@kvborovikova ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@kvborovikova ~]$
```

Рис. 4.9: Смена прав доступа

16. Проверьте, что пользователь guest не может прочитать файл readfile.c. (рис. 4.9)
17. Смените у программы readfile владельца и установите SetU'D-бит. (рис. 4.10)

```
[guest@kvborovikova ~]$ su
Пароль:
[root@kvborovikova guest]# chown root:root readfile
[root@kvborovikova guest]# chmod u+s readfile
[root@kvborovikova guest]# su guest
```

Рис. 4.10: Смена прав доступа и владельца

18. Проверьте, может ли программа readfile прочитать файл readfile.c? (рис. 4.11)

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

Рис. 4.11: Программа readfile читает файл readfile.c

19. Проверьте, может ли программа readfile прочитать файл /etc/shadow? (рис. 4.12)

```
[root@kvborovikova guest]# ./readfile /etc/shadow
root:$6$cITa9kjb94Dl2gwox$8mV3WDYYLHi6VvSm17AYevyqy8W80q7pH420NwE3GhkyKjREpzjTb5UmEG3Q3.fv3/Hy05E
KwXC0vEuZ89/Jyl::0:99999:7:::
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::19616:1:::
```

Рис. 4.12: Программа readfile читает файл /etc/shadow

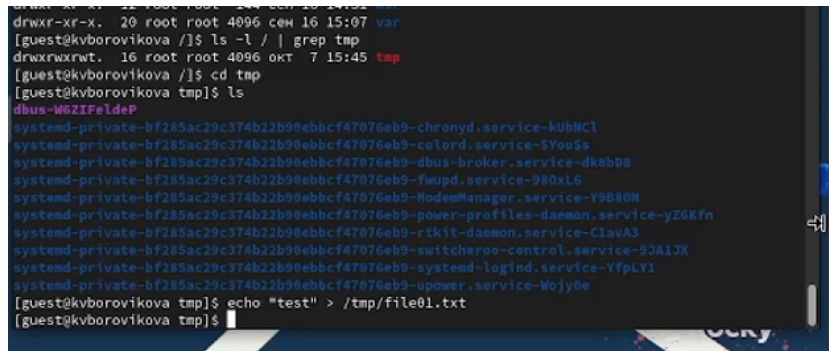
Видим, что программа читает файл /etc/shadow

1. Выясните, установлен ли атрибут Sticky на директории /tmp, для чего выполните команду (рис. 4.13) `ls -l / | grep tmp`

Видим, что установлен

2. От имени пользователя guest создайте файл file01.txt в директории /tmp со словом test:

```
echo "test" > /tmp/file01.txt
```

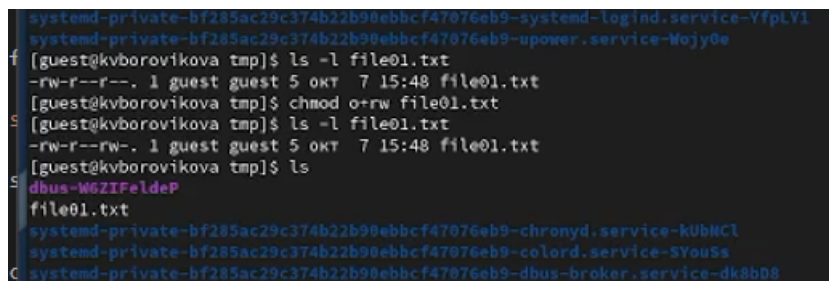


```
drwxr-xr-x. 20 root root 4096 сеп 16 15:07 var
[guest@kvborovikova /]$ ls -l / | grep tmp
drwxrwxrwt. 16 root root 4096 окт 7 15:45 tmp
[guest@kvborovikova /]$ cd tmp
[guest@kvborovikova tmp]$ ls
dbus-W6ZIFeldeP
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-chronyd.service-kUbMCl
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-colord.service-SYouSs
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-dbus-broker.service-dk8bD8
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-fwupd.service-980xLG
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-ModemManager.service-Y9880M
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-power-profiles-daemon.service-yZGKfn
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-rtkit-daemon.service-ClauA3
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-switcheroo-control.service-9JA1JX
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-systemd-logind.service-YfplY1
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-upower.service-Wojoy0e
[guest@kvborovikova tmp]$ echo "test" > /tmp/file01.txt
[guest@kvborovikova tmp]$
```

Рис. 4.13: Записываем текст в файл

3. Просмотрите атрибуты у только что созданного файла и разрешите чтение и запись для категории пользователей «все остальные»: (рис. 4.14)

```
ls -l /tmp/file01.txt
chmod o+rw /tmp/file01.txt
ls -l /tmp/file01.txt
```



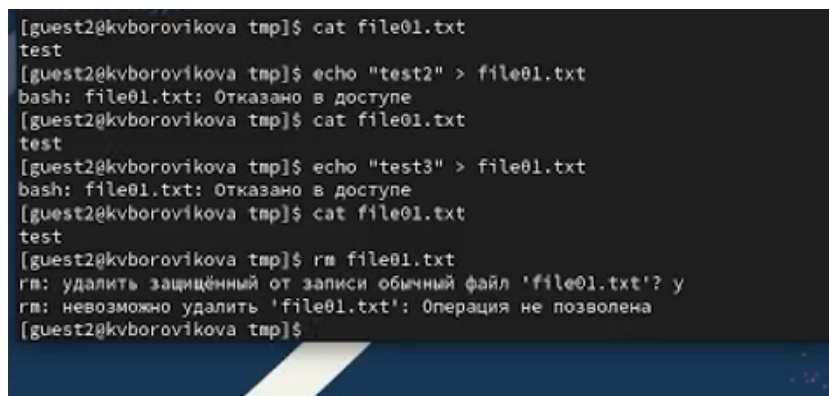
```
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-systemd-logind.service-YfplY1
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-upower.service-Wojoy0e
[guest@kvborovikova tmp]$ ls -l file01.txt
-rw-r--r--. 1 guest guest 5 окт 7 15:48 file01.txt
[guest@kvborovikova tmp]$ chmod o+rw file01.txt
[guest@kvborovikova tmp]$ ls -l file01.txt
-rw-r--rw-. 1 guest guest 5 окт 7 15:48 file01.txt
[guest@kvborovikova tmp]$ ls
dbus-W6ZIFeldeP
file01.txt
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-chronyd.service-kUbMCl
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-colord.service-SYouSs
systemd-private-bf285ac29c374b22b98ebbcf47076eb9-dbus-broker.service-dk8bD8
```

Рис. 4.14: Записываем текст в файл

4. От пользователя guest2 (не являющегося владельцем) попробуйте прочитать файл /tmp/file01.txt: (рис. 4.15)

```
cat /tmp/file01.txt
```


5. От пользователя guest2 попробуйте дозаписать в файл /tmp/file01.txt слово test2 командой (рис. 4.15) `echo "test2" > /tmp/file01.txt` Удалось ли вам выполнить операцию? нет
6. Проверьте содержимое файла командой (рис. 4.15) `cat /tmp/file01.txt`
7. От пользователя guest2 попробуйте записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию командой (рис. 4.15) `echo "test3" > /tmp/file01.txt` Удалось ли вам выполнить операцию? нет
8. Проверьте содержимое файла командой (рис. 4.15) `cat /tmp/file01.txt`
9. От пользователя guest2 попробуйте удалить файл /tmp/file01.txt командой (рис. 4.15) `rm /tmp/file01.txt` Удалось ли вам удалить файл? нет



```
[guest2@kvborovikova tmp]$ cat file01.txt
test
[guest2@kvborovikova tmp]$ echo "test2" > file01.txt
bash: file01.txt: Отказано в доступе
[guest2@kvborovikova tmp]$ cat file01.txt
test
[guest2@kvborovikova tmp]$ echo "test3" > file01.txt
bash: file01.txt: Отказано в доступе
[guest2@kvborovikova tmp]$ cat file01.txt
test
[guest2@kvborovikova tmp]$ rm file01.txt
rm: удалить защищенный от записи обычный файл 'file01.txt'? y
rm: невозможно удалить 'file01.txt': операция не позволена
[guest2@kvborovikova tmp]$
```

Рис. 4.15: Выполняем действия с файлом file01.txt

10. Повысьте свои права до суперпользователя следующей командой `su -` и выполните после этого команду, снимающую атрибут t (Sticky-бит) с директории /tmp: (рис. 4.16) `chmod -t /tmp`
11. Покиньте режим суперпользователя командой (рис. 4.16) `exit`
12. От пользователя guest2 проверьте, что атрибута t у директории /tmp нет: (рис. 4.16) `ls -l / | grep tmp`

13. Повторите предыдущие шаги. Какие наблюдаются изменения? Видим, что нам удалось удалить файл. (рис. 4.16)
14. Удалось ли вам удалить файл от имени пользователя, не являющегося его владельцем? да. (рис. 4.16)

```
rm: невозможно удалить 'file01.txt': операция не позволена
[guest2@kvborovikova tmp]$ su -
Пароль:
[root@kvborovikova ~]# chmod -t /tmp
[root@kvborovikova ~]# exit
выход
[guest2@kvborovikova tmp]$ ls -l / | grep tmp
drwxrwxrwx. 16 root root 4096 окт 7 15:53 tmp
[guest2@kvborovikova tmp]$ cat file01.txt
test
[guest2@kvborovikova tmp]$ echo "test2" > file01.txt
bash: file01.txt: Отказано в доступе
[guest2@kvborovikova tmp]$ cat file01.txt
test
[guest2@kvborovikova tmp]$ echo "test3" > file01.txt
bash: file01.txt: Отказано в доступе
[guest2@kvborovikova tmp]$ cat file01.txt
test
[guest2@kvborovikova tmp]$ rm file01.txt
rm: удалить защищенный от записи обычный файл 'file01.txt'? y
[guest2@kvborovikova tmp]$ cat file01.txt
cat: file01.txt: Нет такого файла или каталога
[guest2@kvborovikova tmp]$
```

Рис. 4.16: Выполняем действия с файлом file01.txt после снятия Sticky-бита

15. Повысьте свои права до суперпользователя и верните атрибут t на директорию /tmp: (рис. 4.17)

```
su -
chmod +t /tmp
exit
```

```
cat: file01.txt: Нет такого файла или каталога
[guest2@kvborovikova tmp]$ su -
Пароль:
[root@kvborovikova ~]# chmod +t /tmp
[root@kvborovikova ~]# exit
выход
[guest2@kvborovikova tmp]$
```

Рис. 4.17: Возвращаем на место

5 Выводы

В ходе выполнения лабораторной работы нам удалось изучить механизмы изменения идентификаторов, применения SetUID- и Sticky-битов, получить практические навыки работы в консоли с дополнительными атрибутами, рассмотреть работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

Список литературы