

Отчёт по лабораторной работе 5

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

Аристова Арина Олеговна

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	8
3.1	5.2.1. Подготовка лабораторного стенда	8
3.2	5.3.1 Создание программы	8
3.3	5.3.2. Исследование Sticky-бита	15
4	Вывод	18
5	Список литературы. Библиография	19

Список иллюстраций

3.1	Установка gss	8
3.2	simpleid.c	9
3.3	3-5 пункты задания лабораторной работы	9
3.4	simpleid2	10
3.5	7 пункт задания лабораторной работы	10
3.6	8-12 пункты задания лабораторной работы	11
3.7	readfile.c	12
3.8	chmod	12
3.9	16-19 пункты Guest	13
3.10	суперпользователь	14
3.11	суперпользователь	15
3.12	1-3 пункты	15
3.13	4-12 пункты	17
3.14	Возвращение атрибута	17

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 Теоретическое введение

1. Дополнительные атрибуты файлов Linux

В Linux существует три основных вида прав — право на чтение (read), запись (write) и выполнение (execute), а также три категории пользователей, к которым они могут применяться — владелец файла (user), группа владельца (group) и все остальные (others). Но, кроме прав чтения, выполнения и записи, есть еще три дополнительных атрибута.

- **Sticky bit**

Используется в основном для каталогов, чтобы защитить в них файлы. В такой каталог может писать любой пользователь. Но, из такой директории пользователь может удалить только те файлы, владельцем которых он является. Примером может служить директория /tmp, в которой запись открыта для всех пользователей, но нежелательно удаление чужих файлов.

- **SUID (Set User ID)**

Атрибут исполняемого файла, позволяющий запустить его с правами владельца. В Linux приложение запускается с правами пользователя, запустившего указанное приложение. Это обеспечивает дополнительную безопасность т.к. процесс с правами пользователя не сможет получить доступ к важным системным файлам, которые принадлежат пользователю root.

- **SGID (Set Group ID)**

Аналогичен `suid`, но относиться к группе. Если установить `sgid` для каталога, то все файлы созданные в нем, при запуске будут принимать идентификатор группы каталога, а не группы владельца, который создал файл в этом каталоге.

- **Обозначение атрибутов `sticky`, `suid`, `sgid`**

Специальные права используются довольно редко, поэтому при выводе программы `ls -l` символ, обозначающий указанные атрибуты, закрывает символ стандартных прав доступа.

Пример:

```
rwsrwsrwt
```

где первая s — это `suid`, вторая s — это `sgid`, а последняя t — это `sticky bit`

В приведенном примере не понятно, `gwt` — это `rw-` или `gwx`? Определить это просто. Если `t` маленькое, значит `x` установлен. Если `T` большое, значит `x` не установлен. То же самое правило распространяется и на `s`.

В числовом эквиваленте данные атрибуты определяются первым символом при четырехзначном обозначении (который часто опускается при назначении прав), например в правах `1777` — символ `1` обозначает `sticky bit`. Остальные атрибуты имеют следующие числовое соответствие:

1 — установлен `sticky bit`

2 — установлен `sgid`

4 — установлен `suid`

2. Компилятор GCC

`GCC` - это свободно доступный оптимизирующий компилятор для языков `C`, `C++`. Собственно программа `gcc` это некоторая надстройка над группой компиляторов, которая способна анализировать имена файлов, передаваемые ей в качестве аргументов, и определять, какие действия необходимо выполнить. Файлы с расширением `.cc` или `.C` рассматриваются, как файлы на языке `C++`, файлы с расширением `.c` как программы на языке `C`, а файлы с расширением `.o` считаются объектными.

3 Выполнение лабораторной работы

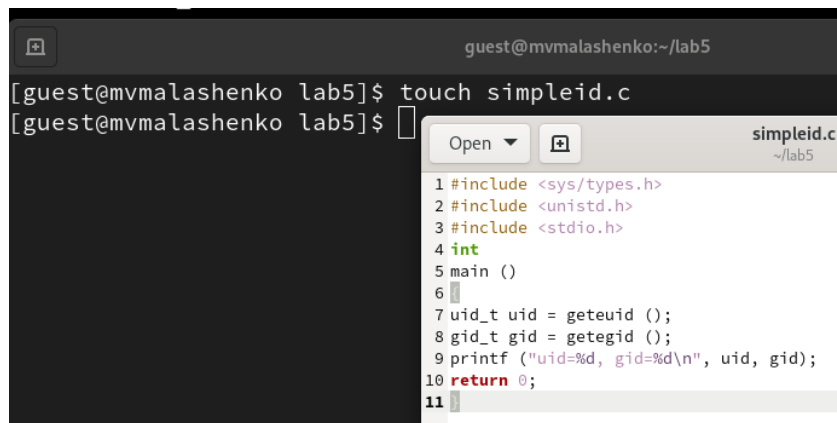
3.1 5.2.1. Подготовка лабораторного стенда

```
[root@mvmalashenko guest]# yum install gcc
Extra Packages for Enterprise Linux 9 - x86_64 33 kB/s | 29 kB 00:00
Extra Packages for Enterprise Linux 9 - x86_64 4.0 MB/s | 19 MB 00:04
Extra Packages for Enterprise Linux 9 openh26 3.6 kB/s | 993 B 00:00
packages for the GitHub CLI 7.0 kB/s | 3.0 kB 00:00
packages for the GitHub CLI 4.0 kB/s | 2.6 kB 00:00
Rocky Linux 9 - BaseOS 1.4 kB/s | 4.1 kB 00:02
Rocky Linux 9 - BaseOS 1.2 MB/s | 1.9 MB 00:01
Rocky Linux 9 - AppStream 5.1 kB/s | 4.5 kB 00:00
Rocky Linux 9 - AppStream 4.3 MB/s | 7.1 MB 00:01
Rocky Linux 9 - Extras 3.7 kB/s | 2.9 kB 00:00
Rocky Linux 9 - Extras 1.0 kB/s | 11 kB 00:10
Package gcc-11.3.1-4.3.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@mvmalashenko guest]# setenforce 0
[root@mvmalashenko guest]# getenforce
Permissive
```

Рис. 3.1: Установка gss

3.2 5.3.1 Создание программы

1. Войдите в систему от имени пользователя guest.
2. Создайте программу simpleid.c.



```
guest@mvmlashenko:~/lab5
[guest@mvmlashenko lab5]$ touch simpleid.c
[guest@mvmlashenko lab5]$
```

```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4 int
5 main ()
6 {
7     uid_t uid = geteuid ();
8     gid_t gid = getegid ();
9     printf ("uid=%d, gid=%d\n", uid, gid);
10    return 0;
11 }
```

Рис. 3.2: 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.



```
guest@mvmlashenko:~/lab5
[guest@mvmlashenko lab5]$ touch simpleid.c
[guest@mvmlashenko lab5]$ gcc simpleid.c -o simpleid
[guest@mvmlashenko lab5]$ ./simpleid
uid=1003, gid=1001
[guest@mvmlashenko lab5]$ id
uid=1003(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 3.3: 3-5 пункты задания лабораторной работы

6. Усложните программу, добавив вывод действительных идентификаторов.

```
[guest@mvmalashenko lab5]$ touch simpleid2.c
```

```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4 int
5 main ()
6 {
7 uid_t real_uid = getuid ();
8 uid_t e_uid = geteuid ();
9 gid_t real_gid = getgid ();
10 gid_t e_gid = getegid ();
11 printf ("e_uid=%d, e_gid=%d\n", e_uid, e_gid);
12 printf ("real_uid=%d, real_gid=%d\n", real_uid, real_gid);
13 return 0;
14 }
```

Рис. 3.4: simpleid2

7. Скомпилируйте и запустите simpleid2.c: `gcc simpleid2.c -o simpleid2`
`./simpleid2`

```
[guest@mvmalashenko lab5]$ gcc simpleid2.c -o simpleid2
[guest@mvmalashenko lab5]$ ./simpleid2
e_uid=1003, e_gid=1001
real_uid=1003, real_gid=1001
```

Рис. 3.5: 7 пункт задания лабораторной работы

8. От имени суперпользователя выполните команды: `chown root:guest`
`/home/guest/simpleid2` `chmod u+s /home/guest/simpleid2`
9. Используйте `sudo` или повысьте временно свои права с помощью `su`. Поясните, что делают эти команды.

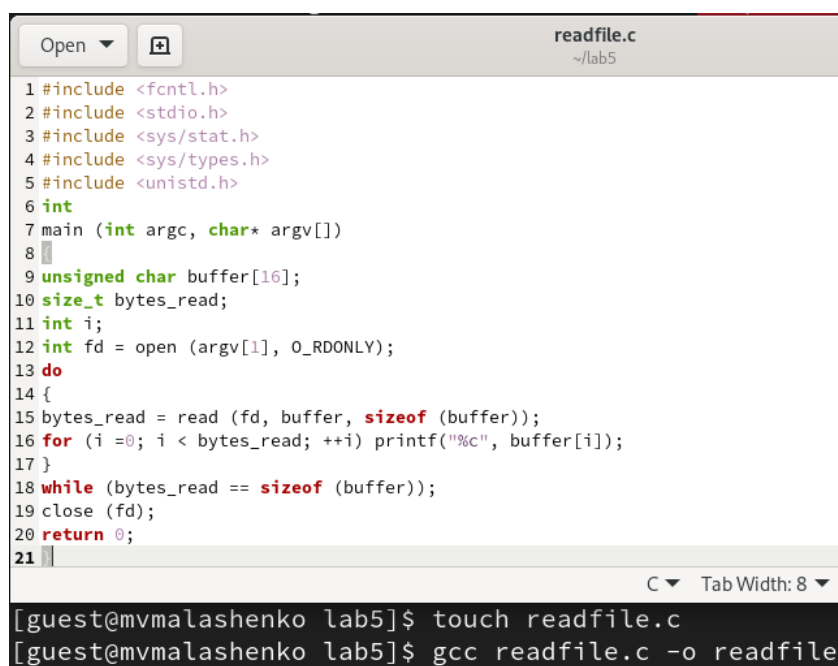
От имени суперпользователя выполнила команды “`sudo chown root:guest /home/guest/simpleid2`” и “`sudo chmod u+s /home/guest/simpleid2`”, затем выполнила проверку правильности установки новых атрибутов и смены владельца файла `simpleid2` командой “`sudo ls -l /home/guest/simpleid2`” (рис. 3.9). Этими командами была произведена смена пользователя файла на `root` и установлен SetUID-бит.

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.

```
[root@mvmalashenko lab5]# chown root:guest simpleid2
[root@mvmalashenko lab5]# chmod u+s simpleid2
[root@mvmalashenko lab5]# ls -l simpleid2
-rwsr-xr-x. 1 root guest 26064 Oct  6 01:56 simpleid2
[root@mvmalashenko lab5]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@mvmalashenko lab5]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@mvmalashenko lab5]# chown root:guest simpleid2
[root@mvmalashenko lab5]# chmod g+s simpleid2
[root@mvmalashenko lab5]# ls -l simpleid2
-rwxr-sr-x. 1 root guest 26064 Oct  6 01:56 simpleid2
[root@mvmalashenko lab5]# ./simpleid2
e_uid=0, e_gid=1001
real_uid=0, real_gid=0
[root@mvmalashenko lab5]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 3.6: 8-12 пункты задания лабораторной работы

13. Создайте программу readfile.c
14. Откомпилируйте её. `gcc readfile.c -o readfile`

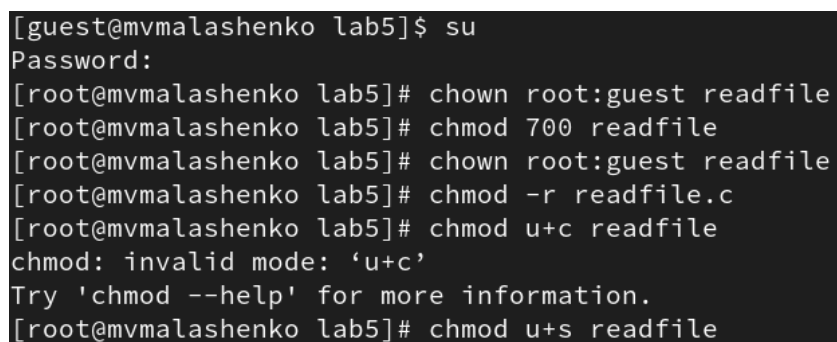


```
1 #include <fcntl.h>
2 #include <stdio.h>
3 #include <sys/stat.h>
4 #include <sys/types.h>
5 #include <unistd.h>
6 int
7 main (int argc, char* argv[])
8 {
9     unsigned char buffer[16];
10    size_t bytes_read;
11    int i;
12    int fd = open (argv[1], O_RDONLY);
13    do
14    {
15        bytes_read = read (fd, buffer, sizeof (buffer));
16        for (i = 0; i < bytes_read; ++i) printf("%c", buffer[i]);
17    }
18    while (bytes_read == sizeof (buffer));
19    close (fd);
20    return 0;
21 }
```

```
[guest@mvmalashenko lab5]$ touch readfile.c
[guest@mvmalashenko lab5]$ gcc readfile.c -o readfile
```

Рис. 3.7: readfile.c

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@mvmalashenko lab5]$ su
Password:
[root@mvmalashenko lab5]# chown root:guest readfile
[root@mvmalashenko lab5]# chmod 700 readfile
[root@mvmalashenko lab5]# chown root:guest readfile.c
[root@mvmalashenko lab5]# chmod -r readfile.c
[root@mvmalashenko lab5]# chmod u+c readfile
chmod: invalid mode: 'u+c'
Try 'chmod --help' for more information.
[root@mvmalashenko lab5]# chmod u+s readfile
```

Рис. 3.8: chmod

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? Отразите полученный результат и ваши объяснения в отчёте.

```
[guest@mvmalashenko lab5]$ cat readfile.c
cat: readfile.c: Permission denied
[guest@mvmalashenko lab5]$ ./readfile readfile.c
bash: ./readfile: Permission denied
[guest@mvmalashenko lab5]$ ./readfile /etc/shadow
bash: ./readfile: Permission denied
```

Рис. 3.9: 16-19 пункты Guest

От имени суперпользователя все команды удастся выполнить.

```
[guest@mvmalashenko lab5]$ su
Password:
[root@mvmalashenko lab5]# 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;
}
[root@mvmalashenko lab5]# ./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;
```

Рис. 3.10: суперпользователь

```
[root@mvmalashenko lab5]# ./readfile /etc/shadow
root:$6$Yq..H.XQpiieRkIh$pJoDaebJmfXvkr6Bo09eyd1f.TYP70S0UqNzx09b90I3D8nSKPwtY
dN/9lc8yeyKrGmmhzwAx4M9aPWF7HKlN/::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:!!:19608:::::
dbus:!!:19608:::::
polkitd:!!:19608:::::
avahi:!!:19608:::::
rtkit:!!:19608:::::
sssd:!!:19608:::::
pipewire:!!:19608:::::
libstoragemgmt:!:19608:::::
```

Рис. 3.11: суперпользователь

3.3 5.3.2. Исследование Sticky-бита

1. Выясните, установлен ли атрибут Sticky на директории /tmp, для чего выполните команду `ls -l / | grep tmp`
2. От имени пользователя guest создайте файл file01.txt в директории /tmp со словом test: `echo "test" > /tmp/file01.txt`
3. 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`

```
[guest@mvmalashenko lab5]$ ls -l / | grep tmp
drwxrwxrwt. 15 root root 4096 Oct  6 02:13 tmp
[guest@mvmalashenko lab5]$ echo "test" > /tmp/file01.txt
[guest@mvmalashenko lab5]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 Oct  6 02:13 /tmp/file01.txt
[guest@mvmalashenko lab5]$ chmod o+rw /tmp/file01.txt
[guest@mvmalashenko lab5]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 Oct  6 02:13 /tmp/file01.txt
```

Рис. 3.12: 1-3 пункты

4. От пользователя `guest2` (не являющегося владельцем) попробуйте прочитать файл `/tmp/file01.txt`: `cat /tmp/file01.txt`
5. От пользователя `guest2` попробуйте дозаписать в файл `/tmp/file01.txt` слово `test2` командой `echo "test2" > /tmp/file01.txt`

Удалось ли вам выполнить операцию? Нет.

6. Проверьте содержимое файла командой `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 `echo "test3" > /tmp/file01.txt`

Удалось ли вам выполнить операцию? Нет.

8. Проверьте содержимое файла командой `cat /tmp/file01.txt`
9. От пользователя `guest2` попробуйте удалить файл `/tmp/file01.txt` командой `rm /tmp/file01.txt`

Удалось ли вам удалить файл? Нет.

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


```
[guest@mvmlashenko lab5]$ guest2
bash: guest2: command not found...
[guest@mvmlashenko lab5]$ su guest2
Password:
[guest2@mvmlashenko lab5]$ cat /tmp/file01.txt
test
[guest2@mvmlashenko lab5]$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@mvmlashenko lab5]$ cat /tmp/file01.txt
test
[guest2@mvmlashenko lab5]$ echo "test3" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@mvmlashenko lab5]$ cat /tmp/file01.txt
test
[guest2@mvmlashenko lab5]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory
[guest2@mvmlashenko lab5]$ su
Password:
[root@mvmlashenko lab5]# chmod -t /tmp
[root@mvmlashenko lab5]# exit
exit
[guest2@mvmlashenko lab5]$ ls -l / | grep tmp
drwxrwxrwx. 16 root root 4096 Oct 6 02:17 tmp
```

Рис. 3.13: 4-12 пункты

13. Повторите предыдущие шаги. Какие наблюдаются изменения?

При повторении всё получилось.

14. Удалось ли вам удалить файл от имени пользователя, не являющегося его владельцем? Удалось.

15. Повысьте свои права до суперпользователя и верните атрибут t на директорию /tmp: su chmod +t /tmp exit

```
[guest2@mvmlashenko lab5]$ su
Password:
[root@mvmlashenko lab5]# chmod +t /tmp
[root@mvmlashenko lab5]# exit
exit
[guest2@mvmlashenko lab5]$
```

Рис. 3.14: Возвращение атрибута

4 Вывод

Были изучены механизмы изменения идентификаторов и применения SetUID- и Sticky-битов. Получены практические навыки работы в консоли с дополнительными атрибутами. Были рассмотрены работа механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов

5 Список литературы. Библиография

- Описание лабораторной работы