

Отчёт по лабораторной работе №5.

Информационная безопасность

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

Выполнила: Данзанова Саяна Зоригтеевна, НПИбд-01-21, 1032217624

Содержание

Цель работы	4
Теоретическое введение	5
Выполнение лабораторной работы	7
5.2.1. Подготовка лабораторного стенда	7
5.3.1 Создание программы	7
5.3.2. Исследование Sticky-бита	12
Вывод	15
Список литературы. Библиография	16

Список иллюстраций

1	(рис. 1. Установка gss)	7
2	(рис. 2. simpleid.c)	8
3	(рис. 3. 3-5 пункты задания лабораторной)	8
4	(рис. 4. simpleid2.c)	9
5	(рис. 5. 7 пункт задания лабораторной)	9
6	(рис. 6. 8-12 пункты задания лабораторной)	10
7	(рис. 7. readfile.c)	10
8	(рис. 8. chmod)	11
9	(рис. 9. 16-19 пункты Guest)	12
10	(рис. 12. 1-3 пункты)	13
11	(рис. 13. 4-12 пункты)	14
12	(рис. 15. Возвращение атрибута)	14

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

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. [1]

- **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 относится 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, `rwt` — это `rw-` или `rwX`? 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. [2]

Выполнение лабораторной работы

5.2.1. Подготовка лабораторного стенда

```
[guest@szdanzanova ~]$ su
Пароль:
[root@szdanzanova guest]# yum install gcc
Extra Packages for Enterprise Linux 9 - x86_64 52 kB/s | 38 kB 00:00
Extra Packages for Enterprise Linux 9 - x86_64 2.0 MB/s | 23 MB 00:11
Extra Packages for Enterprise Linux 9 openh264 3.5 kB/s | 993 B 00:00
packages for the GitHub CLI 17 kB/s | 3.0 kB 00:00
packages for the GitHub CLI 5.7 kB/s | 2.7 kB 00:00
Rocky Linux 9 - BaseOS 7.7 kB/s | 4.1 kB 00:00
Rocky Linux 9 - BaseOS 1.7 MB/s | 2.3 MB 00:01
Rocky Linux 9 - AppStream 11 kB/s | 4.5 kB 00:00
Rocky Linux 9 - AppStream 1.9 MB/s | 8.0 MB 00:04
Rocky Linux 9 - Extras 8.6 kB/s | 2.9 kB 00:00
Пакет gcc-11.4.1-3.el9.x86_64 уже установлен.
Зависимости разрешены.
Отсутствуют действия для выполнения.
Выполнено!
[root@szdanzanova guest]# setenforce 0
[root@szdanzanova guest]# getenforce
Permissive
```

Рис. 1: (рис. 1. Установка gss)

5.3.1 Создание программы

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

```
[guest@szdanzanova ~]$ touch simpleid.c
[guest@szdanzanova ~]$
```



```
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 }
```

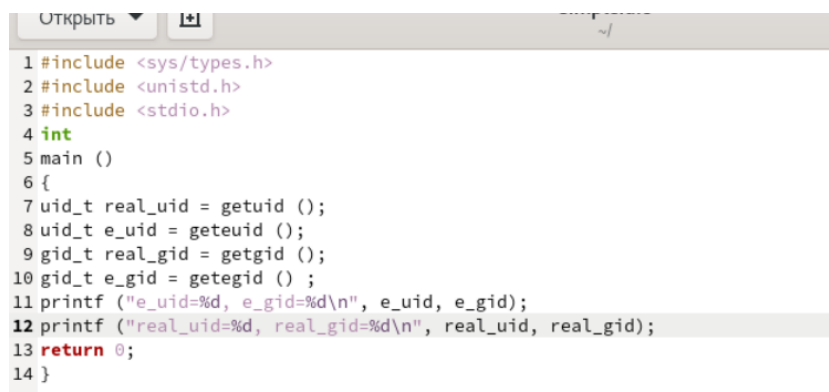
Рис. 2: (рис. 2. simpleid.c)

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@szdanzanova ~]$ touch simpleid.c
[guest@szdanzanova ~]$ gcc simpleid.c -o simpleid
[guest@szdanzanova ~]$ ./simpleid
uid=1001, gid=1001
[guest@szdanzanova ~]$ id
uid=1001(guest) gid=1001(guest) rгруппы=1001(guest) контекст=unconfined_u:unconfi
ned_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 3: (рис. 3. 3-5 пункты задания лабораторной)

6. Усложните программу, добавив вывод действительных идентификаторов.



```

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. 4: (рис. 4. simpleid2.c)

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



```

[guest@szdanzanova ~]$ gcc simpleid2.c -o simpleid2
[guest@szdanzanova ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001

```

Рис. 5: (рис. 5. 7 пункт задания лабораторной)

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.

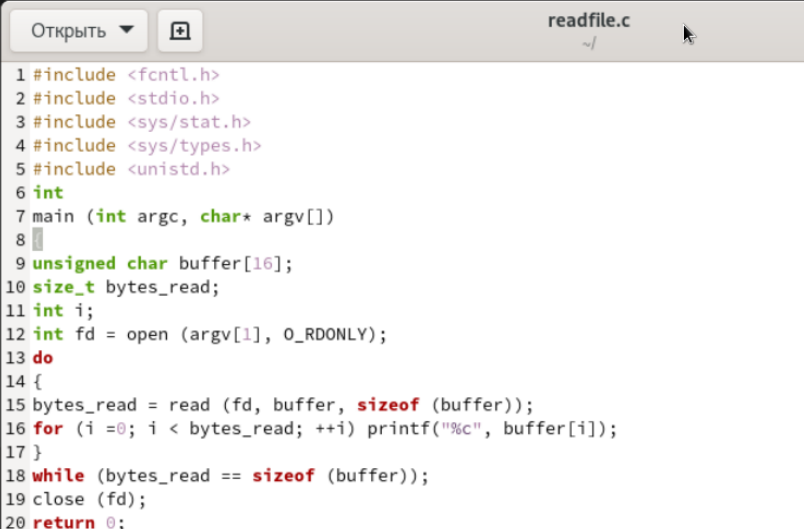
```
[root@szdanzanova guest]# chown root:guest /home/guest/simpleid2
[root@szdanzanova guest]# chmod u+s /home/guest/simpleid2
[root@szdanzanova guest]# ls -l simpleid2
-rwsr-xr-x. 1 root guest 17720 окт 5 19:13 simpleid2
[root@szdanzanova guest]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@szdanzanova guest]# id
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@szdanzanova guest]# chown root:guest /home/guest/simpleid2
[root@szdanzanova guest]# chmod g+s /home/guest/simpleid2
[root@szdanzanova guest]# ls -l simpleid2
-rwxr-sr-x. 1 root guest 17720 окт 5 19:13 simpleid2
[root@szdanzanova guest]# ./simpleid2
e_uid=0, e_gid=1001
real_uid=0, real_gid=0
[root@szdanzanova guest]# id
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 6: (рис. 6. 8-12 пункты задания лабораторной)

13. Создайте программу readfile.c

14. Откомпилируйте её. gcc readfile.c -o readfile

```
[guest@szdanzanova ~]$ touch readfile.c
[guest@szdanzanova ~]$ gcc readfile.c -o readfile
[guest@szdanzanova ~]$
```



```
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;
```

Рис. 7: (рис. 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@szdanzanova ~]$ su
Пароль:
[root@szdanzanova guest]# chown root:guest readfile
[root@szdanzanova guest]# chown 700 readfile
[root@szdanzanova guest]# chown root:guest readfile
[root@szdanzanova guest]# chown -r readfile.c
chown: неверный ключ - «r»
По команде «chown --help» можно получить дополнительную информацию.
[root@szdanzanova guest]# chmod -r readfile.c
[root@szdanzanova guest]# chmod u+s readfile
```

Рис. 8: (рис. 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? Отрадите полученный результат и ваши объяснения в отчёте.

```
[guest@szdanzanova ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@szdanzanova ~]$ ./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@szdanzanova ~]$ ./readfile /etc/shadow
root:$6$0S.X8qKi.B7YyXaY$sieLpkqyQVlckEygcblyv6tu00BEa3GkaCrLIoJ80ud0hB99gLHb2Nz
piqrD6eYEF6FOY5cQAGVTWEGTk8cq20::0:99999:7:::
bin:!:19820:0:99999:7:::
daemon:!:19820:0:99999:7:::
adm:!:19820:0:99999:7:::
lp:!:19820:0:99999:7:::
sync:!:19820:0:99999:7:::
shutdown:!:19820:0:99999:7:::
halt:!:19820:0:99999:7:::
mail:!:19820:0:99999:7:::
operator:!:19820:0:99999:7:::
games:!:19820:0:99999:7:::
ftp:!:19820:0:99999:7:::
nobody:!:19820:0:99999:7:::
```

Рис. 9: (рис. 9. 16-19 пункты Guest)

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@szdanzanova ~]$ ls -l / | grep tmp
drwxrwxrwt. 18 root root 4096 окт  5 20:02 tmp
[guest@szdanzanova ~]$ echo "test" > /tmp/file01.txt
[guest@szdanzanova ~]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 окт  5 20:08 /tmp/file01.txt
[guest@szdanzanova ~]$ chmod o+rw /tmp/file01.txt
[guest@szdanzanova ~]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 окт  5 20:08 /tmp/file01.txt
```

Рис. 10: (рис. 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@szdanzanova ~]$ su guest2
Пароль:
[guest2@szdanzanova guest]$ cat /tmp/file01.txt
test
[guest2@szdanzanova guest]$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@szdanzanova guest]$ cat /tmp/file01.txt
test
[guest2@szdanzanova guest]$ rm /tmp/file01.txt
rm: удалить защищённый от записи обычный файл '/tmp/file01.txt'?
[guest2@szdanzanova guest]$ su -
Пароль:
[root@szdanzanova ~]# chmod -t /tmp
[root@szdanzanova ~]# exit
выход
[guest2@szdanzanova guest]$ ls -l / | grep tmp
drwxrwxrwx. 18 root root 4096 окт  5 20:13 tmp
```

Рис. 11: (рис. 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@szdanzanova guest]$ su -
Пароль:
[root@szdanzanova ~]# chmod +t /tmp
[root@szdanzanova ~]# exit
выход
[guest2@szdanzanova guest]$ ls -l / | grep tmp
drwxrwxrwt. 20 root root 4096 окт  5 20:14 tmp
```

Рис. 12: (рис. 15. Возвращение атрибута)

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

Список литературы. Библиография

[0] Методические материалы курса

[1] Дополнительные атрибуты: <https://tokmakov.msk.ru/blog/item/141>

[2] Компилятор GSS: <http://parallel.imm.uran.ru/freesoft/make/instrum.html>