

Лабораторная работа № 5

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

Беличева Дарья Михайловна

Содержание

1	Цель работы	4
2	Теоретическое введение	5
3	Выполнение лабораторной работы	7
4	Выводы	14
	Список литературы	15

Список иллюстраций

3.1	Подготовка лабораторного стенда	7
3.2	Содержимое файла simpleid.c	7
3.3	Запуск программы simpleid	8
3.4	Содержимое файла simpleid2.c	8
3.5	Запуск программы simpleid2	8
3.6	Изменение владельца и запуск программы simpleid2 с установленным SetUID-битом	9
3.7	Запуск программы simpleid2 с установленным SetGID-битом	9
3.8	Содержимое файла readfile.c	10
3.9	Изменение владельца и прав файла readfile.c	11
3.10	Установка SetUID-бита на исполняемый файл readfile и проверка прав	12
3.11	Исследование Sticky-бита	13

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

В области разрешений Linux, SUID (установка идентификатора пользователя), SGID (установка идентификатора группы) и Sticky Bit являются критически важными атрибутами, которые выходят за рамки стандартных разрешений для файлов, обеспечивая мощные функциональные возможности, такие как предоставление временных привилегий, контроль группового доступа и сохранение целостности данных [1].

SUID, сокращение от Set User ID, представляет собой специальное разрешение, которое может быть назначено исполняемым файлам. Когда у исполняемого файла включено разрешение SUID, это позволяет пользователям, выполняющим файл, временно принимать привилегии владельца файла. Это означает, что даже если у пользователя нет необходимых разрешений для доступа или выполнения определенных действий, он может сделать это, запустив файл с разрешением SUID.

SGID, что означает Set Group ID, — это ещё одно специальное разрешение, которое можно применить к исполняемым файлам и каталогам. Если для исполняемого файла включено разрешение SGID, это позволяет пользователям, которые запускают файл, временно стать владельцами группы, к которой относится файл. Для каталогов с включённым разрешением SGID вновь созданные файлы и каталоги в этом каталоге наследуют принадлежность к группе родительского каталога, а не принадлежность к группе пользователя по умолчанию.

Sticky Bit – это специальное разрешение, которое можно установить только для каталогов. Когда залипание включено для каталога, оно ограничивает воз-

возможность удаления или переименования файлов в этом каталоге для владельца файла, владельца каталога и суперпользователя. Это гарантирует, что каждый пользователь может удалять или изменять только свои файлы, даже если у него есть права на запись в каталог.

3 Выполнение лабораторной работы

Установим компилятор gcc, а также отключим SELinux (рис. 3.1).

```
[dmbelicheva@dmbelicheva ~]$ su -
Password:
[root@dmbelicheva ~]# yum install gcc
Rocky Linux 9 - BaseOS                               4.4 kB/s | 4.1 kB      00:00
Rocky Linux 9 - BaseOS                               1.1 MB/s | 2.3 MB      00:02
Rocky Linux 9 - AppStream                             4.8 kB/s | 4.5 kB      00:00
Rocky Linux 9 - AppStream                             2.8 MB/s | 8.0 MB      00:02
Rocky Linux 9 - Extras                               3.0 kB/s | 2.9 kB      00:00
Package gcc-11.4.1-3.el9.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@dmbelicheva ~]# setenforce 0
[root@dmbelicheva ~]# getenforce
bash: getenforce: command not found...
[root@dmbelicheva ~]# getenforce
Permissive
[root@dmbelicheva ~]#
```

Рис. 3.1: Подготовка лабораторного стенда

Войдем в систему от имени пользователя guest и создадим программу simpleid.c (рис. 3.2).

```
Open simpleid.c ~/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 }
12
```

Рис. 3.2: Содержимое файла simpleid.c

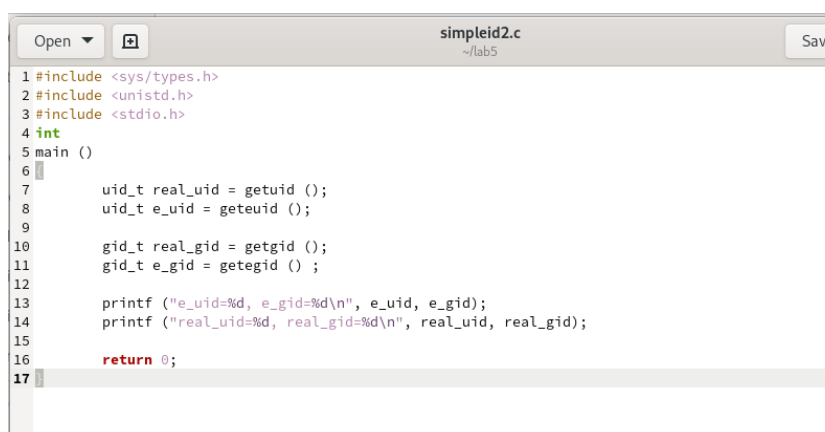
Скомпилируем программу и убедимся, что файл программы создан. Далее выполним программу simpleid, нам выведутся uid и gid. Выполним системную про-

грамму `id` и сравним полученный результат с данными предыдущего пункта задания (рис. 3.3). Увидим, что информация идентична.

```
simpleid.c
[guest@dmbelicheva lab5]$ gcc simpleid.c -o simpleid
[guest@dmbelicheva lab5]$ ./simpleid
uid=1001, gid=1001
[guest@dmbelicheva lab5]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfi
ned_r:unconfined_t:s0-s0:c0.c1023
[guest@dmbelicheva lab5]$
```

Рис. 3.3: Запуск программы `simpleid`

Усложните программу, добавив вывод действительных идентификаторов (рис. 3.4).



```
simpleid2.c
~/lab5
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
10    gid_t real_gid = getgid ();
11    gid_t e_gid = getegid ();
12
13    printf ("e_uid=%d, e_gid=%d\n", e_uid, e_gid);
14    printf ("real_uid=%d, real_gid=%d\n", real_uid, real_gid);
15
16    return 0;
17 }
```

Рис. 3.4: Содержимое файла `simpleid2.c`

Скомпилируем программу и убедимся, что файл программы создан. Далее выполним программу `simpleid`, нам выведутся `uid` и `gid`, а также их действительные идентификаторы (рис. 3.5).

```

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

Рис. 3.5: Запуск программы `simpleid2`

От имени суперпользователя изменим владельца файла `/home/guest/simpleid2`

и установим SetUID-бит. Проверим корректность установленных прав и опять запустим simpleid2 (рис. 3.6).

```
root@dmbelicheva:~  
[root@dmbelicheva ~]# chown root:guest /home/guest/simpleid2  
chown: cannot access '/home/guest/simpleid2': No such file or directory  
[root@dmbelicheva ~]# chown root:guest /home/guest/lab5/simpleid2  
[root@dmbelicheva ~]# chmod u+s /home/guest/lab5/simpleid2  
[root@dmbelicheva ~]# ls -l /home/guest/lab5/simpleid2  
-rwsr-xr-x. 1 root guest 24488 Oct  5 16:32 /home/guest/lab5/simpleid2  
[root@dmbelicheva ~]# su -  
[root@dmbelicheva ~]# su - guest  
[guest@dmbelicheva ~]# cd lab5  
[guest@dmbelicheva lab5]# ./simpleid2  
e_uid=0, e_gid=1001  
real_uid=1001, real_gid=1001  
[guest@dmbelicheva lab5]# id  
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023  
[guest@dmbelicheva lab5]# su - dmbelicheva  
Password:  
[dmbelicheva@dmbelicheva ~]# su /home/guest/lab5/simpleid2  
su: user /home/guest/lab5/simpleid2 does not exist or the user entry does not contain all the required fields  
[dmbelicheva@dmbelicheva ~]# cd /home/guest/lab5  
-bash: cd: /home/guest/lab5: Permission denied  
[dmbelicheva@dmbelicheva ~]# su - root  
Password:  
[root@dmbelicheva ~]# /home/guest/lab5/simpleid2  
e_uid=0, e_gid=0  
real_uid=0, real_gid=0  
[root@dmbelicheva ~]# id  
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023  
[root@dmbelicheva ~]#
```

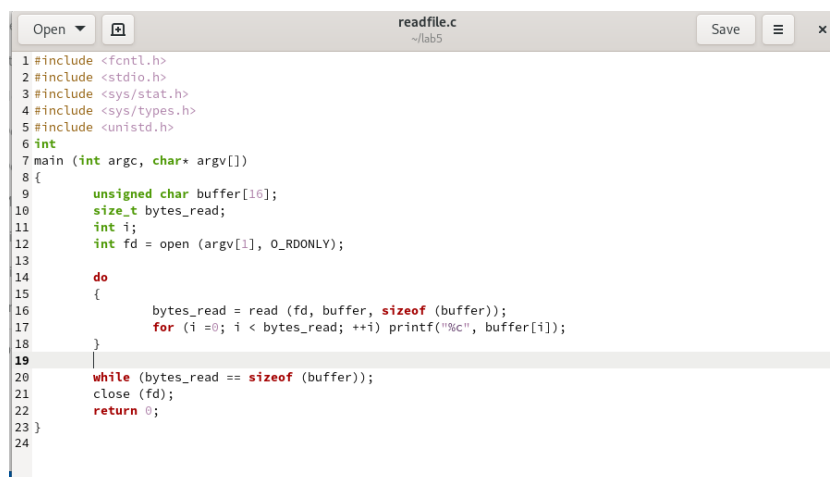
Рис. 3.6: Изменение владельца и запуск программы simpleid2 с установленным SetUID-битом

Продедаем тоже самое относительно SetGID-бита (рис. 3.7).

```
[root@dmbelicheva ~]# chmod u-s /home/guest/simpleid2  
chmod: cannot access '/home/guest/simpleid2': No such file or directory  
[root@dmbelicheva ~]# chmod u-s /home/guest/lab5/simpleid2  
chmod: cannot access '/home/guest/lab5/simpleid2': No such file or directory  
[root@dmbelicheva ~]# chmod u-s /home/guest/lab5/simpleid2  
[root@dmbelicheva ~]# chmod g+s /home/guest/lab5/simpleid2  
[root@dmbelicheva ~]# ls -l /home/guest/lab5/simpleid2  
-rwxr-sr-x. 1 root guest 24488 Oct  5 16:32 /home/guest/lab5/simpleid2  
[root@dmbelicheva ~]# exit  
logout  
[dmbelicheva@dmbelicheva ~]# su - guest  
Password:  
[guest@dmbelicheva ~]# /home/guest/lab5/simpleid2  
e_uid=1001, e_gid=1001  
real_uid=1001, real_gid=1001  
[guest@dmbelicheva ~]# id  
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023  
[guest@dmbelicheva ~]#
```

Рис. 3.7: Запуск программы simpleid2 с установленным SetGID-битом

Создадим программу readfile.c (рис. 3.8).



```

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
14    do
15    {
16        bytes_read = read (fd, buffer, sizeof (buffer));
17        for (i = 0; i < bytes_read; ++i) printf("%c", buffer[i]);
18    }
19    while (bytes_read == sizeof (buffer));
20    close (fd);
21    return 0;
22 }
23
24

```

Рис. 3.8: Содержимое файла readfile.c

Откомпилируем её. Сменим владельца у файла readfile.c на root и изменим права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог. Для этого изменим права файла на 700, то есть только пользователю (root) разрешены все действия. Проверим, что пользователь guest не может прочитать файл readfile.c (рис. 3.9).

```

[guest@dmbelicheva lab5]$ gcc readfile.c -o readfile
[guest@dmbelicheva lab5]$ cat /home/guest/lab5/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@dmbelicheva lab5]$ su -
Password:
[root@dmbelicheva ~]# chown root:guest /home/guest/lab5/readfile.c
[root@dmbelicheva ~]# chmod 700 /home/guest/lab5/readfile.c
[root@dmbelicheva ~]# cat /home/guest/lab5/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@dmbelicheva ~]# exit
logout
[guest@dmbelicheva lab5]$ cat /home/guest/lab5/readfile.c
cat: /home/guest/lab5/readfile.c: Permission denied
[guest@dmbelicheva lab5]$

```

Рис. 3.9: Изменение владельца и прав файла readfile.c

Сменим у программы readfile владельца и установим SetUID-бит. Убедимся, что программа readfile может прочитать файл readfile.c. Также проверим, что программа readfile может прочитать файл /etc/shadow (рис. 3.10).

```

[root@mbelicheva ~]# chown root:guest /home/guest/lab5/readfile
[root@mbelicheva ~]# chmod u+s /home/guest/lab5/readfile
[root@mbelicheva ~]# exit
logout
[guest@mbelicheva 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;
    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@mbelicheva lab5]$ ./readfile /etc/shadow
root:$6$8rwD4MPTc6Eh5nhR$WgppMuYr6cv2.60jI3Wks/Ld68LG00irzvJFsYMcJZ/YME6JCEmff/MvEy/HSDrkEwVQcrCIS7y6rxv.PgA600:0:99999:7:::
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:::

```

Рис. 3.10: Установка SetUID-бита на исполняемый файл readfile и проверка прав

Проверим, что установлен атрибут Sticky на директории /tmp (в конце стоит t). Затем от имени пользователя guest создадим файл file01.txt в директории /tmp со словом test, затем посмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные». После этого от пользователя guest2 попробуем дозаписать в этот файл новое слово, однако получим отказ, также нам отказано в перезаписи и удалении этого файла. Если же убрать Sticky бит, то нам будет разрешено удаление этого файла (рис. 3.11).

```

[guest@dmbelicheva ~]$ ls -l / | grep tmp
drwxrwxrwt. 16 root root 4096 Oct  5 18:14 tmp
[guest@dmbelicheva ~]$ echo "test" > /tmp/file01.txt
[guest@dmbelicheva ~]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 Oct  5 18:17 /tmp/file01.txt
[guest@dmbelicheva ~]$ chmod o+rw /tmp/file01.txt
[guest@dmbelicheva ~]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 Oct  5 18:17 /tmp/file01.txt
[guest@dmbelicheva ~]$ su - guest2
Password:
[guest2@dmbelicheva ~]$ cat /tmp/file01.txt
test
[guest2@dmbelicheva ~]$ echo "test2" > /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@dmbelicheva ~]$ cat /tmp/file01.txt
test
[guest2@dmbelicheva ~]$ echo "test3" >> /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@dmbelicheva ~]$ cat /tmp/file01.txt
test
[guest2@dmbelicheva ~]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory
[guest2@dmbelicheva ~]$ su -
Password:
[root@dmbelicheva ~]# chmod -t /tmp
[root@dmbelicheva ~]# exit
logout
[guest2@dmbelicheva ~]$ ls -l / | grep tmp
drwxrwxrwx. 18 root root 4096 Oct  5 18:24 tmp
[guest2@dmbelicheva ~]$ echo "test2" > /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@dmbelicheva ~]$ cat /tmp/file01.txt
test
[guest2@dmbelicheva ~]$ echo "test3" >> /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@dmbelicheva ~]$ cat /tmp/file01.txt
test
[guest2@dmbelicheva ~]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory
[guest2@dmbelicheva ~]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory
[guest2@dmbelicheva ~]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory
[guest2@dmbelicheva ~]$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'? y
[guest2@dmbelicheva ~]$ su -
Password:
[root@dmbelicheva ~]# chmod +t /tmp
[root@dmbelicheva ~]# exit
logout
[guest2@dmbelicheva ~]$ ls -l / | grep tmp

```

Рис. 3.11: Исследование Sticky-бита

4 Выводы

В процессе выполнения данной лабораторной работы я изучила механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получила практические навыки работы в консоли с дополнительными атрибутами. Рассмотрела работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

Список литературы

1. What is SUID, SGID, and Sticky Bit? [Электронный ресурс]. 2024. URL: <https://www.scaler.com/topics/special-permissions-in-linux/>.