

Презентация к лабораторной работе 5

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Аристова А.О.

05 октября 2024

Российский университет дружбы народов имени Патриса Лумумбы, Москва, Россия

- Аристова Арина Олеговна
- студентка группы НФбд-01-21
- Российский университет дружбы народов имени Патриса Лумумбы
- 1032216433@rudn.ru
- <https://github.com/aoaristova>



- 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

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]

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]

Ход выполнения лабораторной работы

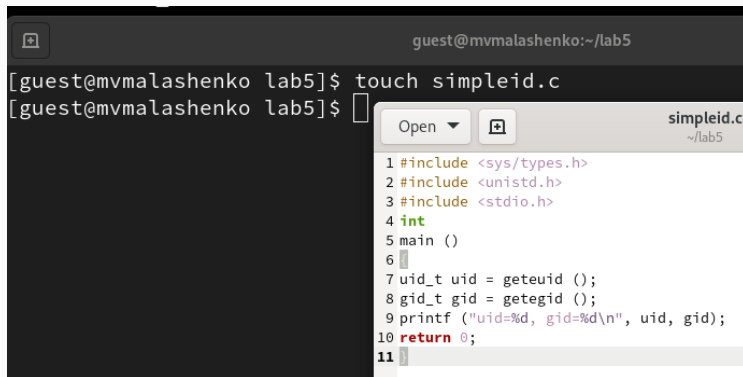
5.2.1. Подготовка лабораторного стенда

```
[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. 1: (рис. 1. Установка gss)

5.3.1 Создание программы

Создали программу simpleid.c



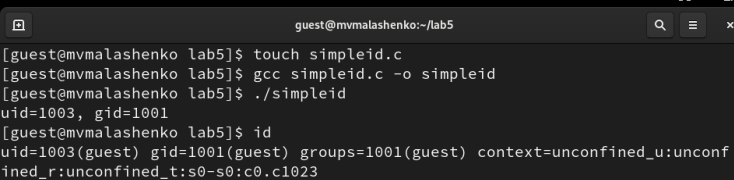
The image shows a terminal window and a code editor. The terminal window has a title bar 'guest@mvmalashenko:~/lab5' and shows the command 'touch simpleid.c' being executed. The code editor, titled 'simpleid.c' and located at '~/lab5', displays the following C code:

```
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)

5.3.1 Создание программы

Скомпилировали и выполнили программу `simpleid`. Затем выполнили системную программу `id` и сравнили полученные результаты

A terminal window with a dark background. The title bar shows 'guest@mvmalashenko:~/lab5'. The terminal contains the following text:

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

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

5.3.1 Создание программы

Усложнили 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. 4: (рис. 4. simpleid2.c)

5.3.1 Создание программы

Скомпилировали и выполнили программу 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. 5: (рис. 5. 7 пункт задания лабораторной)

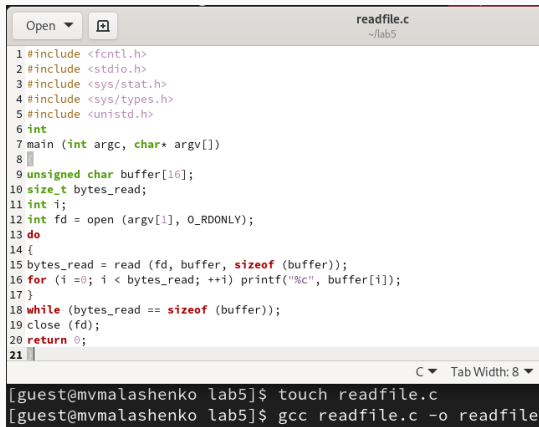
5.3.1 Создание программы

От имени суперпользователя выполнили команды и проверили правильность установки новых атрибутов и смены владельца файла. Запустили `simpleid2` и `id`. Сравнили результаты. Прodelали то же 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
```

5.3.1 Создание программы

Скомпилировали программу readfile.c



```
readfile.c
~/lab5

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. 7: (рис. 7. readfile.c)

5.3.1 Создание программы

Сменили владельца у файла и изменили права так, чтобы только суперпользователь 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
[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. 8: (рис. 8. chmod)

5.3.1 Создание программы

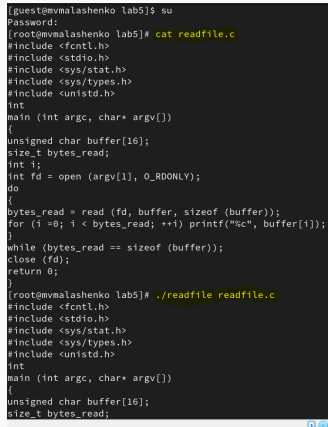
Проверили, что guest не может прочитать файл. Сменили у программы readfile владельца и установили SetU'D-бит. Проверили, может ли программа readfile прочитать файл readfile.c, файл /etc/shadow

```
[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. 9: (рис. 9. 16-19 пункты Guest)

5.3.1 Создание программы

От имени суперпользователя все предыдущие 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. 10: (рис. 10. 16-18 пункты суперпользователь)

5.3.2. Исследование Sticky-бита

Выяснили, установлен ли атрибут Sticky на директории /tmp, создали файл file01.txt со словом test. Просмотрели атрибуты у только что созданного файла и разрешили чтение и запись для категории пользователей «все остальные»

```
[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. 11: (рис. 12. 1-3 пункты)

5.3.2. Исследование Sticky-бита

От guest2 попробовали прочесть файл, дозаписать слово test2, затем записать слово test3, стерева при этом всю имеющуюся в файле информацию. Попробовали удалить файл. Этого сделать не удалось.

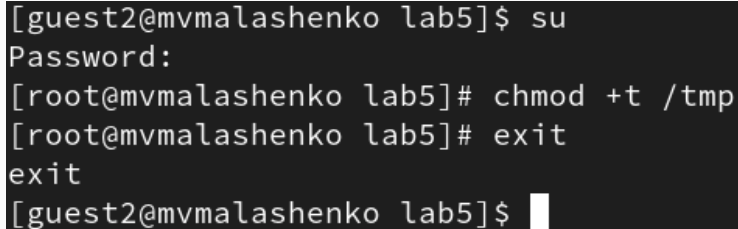
Повысили свои права до суперпользователя и сняли атрибут t с директории /tmp. От guest2 проверили, что атрибута t у директории /tmp нет

```
[guest@mvmalashenko lab5]$ guest2
bash: guest2: command not found...
[guest@mvmalashenko lab5]$ su guest2
Password:
[guest2@mvmalashenko lab5]$ cat /tmp/file01.txt
test
[guest2@mvmalashenko lab5]$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@mvmalashenko lab5]$ cat /tmp/file01.txt
test
[guest2@mvmalashenko lab5]$ echo "test3" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@mvmalashenko lab5]$ cat /tmp/file01.txt
test
[guest2@mvmalashenko lab5]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory
[guest2@mvmalashenko lab5]$ su
Password:
[root@mvmalashenko lab5]# chmod -t /tmp
```

5.3.2. Исследование Sticky-бита

Повторили предыдущие шаги. При повторении всё получилось. Удалось удалить файл от имени пользователя, не являющегося его владельцем.

Повысили свои права до суперпользователя и вернули атрибут `t` на директорию `/tmp`

A terminal window with a dark background and white text. The prompt is [guest2@mvmalashenko lab5]\$. The user enters 'su'. The prompt changes to [root@mvmalashenko lab5]#. The user enters 'chmod +t /tmp'. The prompt changes to [root@mvmalashenko lab5]#. The user enters 'exit'. The prompt returns to [guest2@mvmalashenko lab5]\$.

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

Рис. 13: (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>