

# **Основы информационной безопасности**

**Лабораторная работа № 5. Дискреционное разграничение прав в Linux.  
Исследование влияния дополнительных атрибутов**

Нзита Диатезилуа Катенди

# Содержание

ПЦель работы	4
Теоретические сведения	5
Выполнение лабораторной работы	6
Выводы	11
Список литературы	12

# Список иллюстраций

1	Подготовка лабораторного стенда . . . . .	6
2	Текст прорааммы simpleid.c . . . . .	6
3	Запуск програмы simpleid.c . . . . .	7
4	Текст прорааммы simpleid2 . . . . .	7
5	Запуск програмы simpleid2.c . . . . .	8
6	Изменение владельца и запуск программы simpleid2 с установлен- ным SetUID-битом . . . . .	8
7	Текст программы readfile.c . . . . .	9
8	Изменение владельца и прав файла readfile.c . . . . .	9
9	Подключение образа диска дополнений . . . . .	10

## **ПЦель 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.

# Теоретические сведения

При работе с командой `chmod` важно понимать основные разрешения, назначаемые файлам или каталогам. В Linux[@scott\_linux\_2019] существует три основных типа разрешений:

- Чтение — обозначается буквой «r». Предоставляет возможность просмотра содержимого файла или каталога.
- Запись — обозначается буквой «w». Позволяет создавать, изменять и удалять файлы в каталоге, а также изменять содержимое файла.
- Выполнение — обозначается буквой «x». Предоставляет разрешение на выполнение файла или вход в каталог.

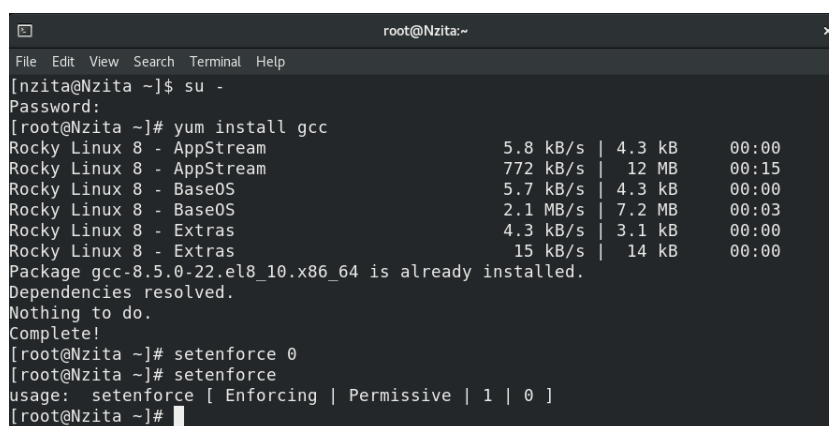
Каждый из вышеперечисленных типов разрешений может быть назначен трем группам пользователей:

- Владелец — пользователь, которому принадлежит файл или каталог.
- Группа — группа пользователей, к которой принадлежит файл или каталог.
- Другие — все остальные пользователи в системе.

Комбинация этих основных разрешений для каждой группы пользователей определяет полный набор разрешений для файла или каталога.

# Выполнение лабораторной работы

Проверим установлен ли компилятора gcc, а также отключим SELinux (рис. @fig:001)



```
root@Nzita:~  
File Edit View Search Terminal Help  
[nzita@Nzita ~]$ su -  
Password:  
[root@Nzita ~]# yum install gcc  
Rocky Linux 8 - AppStream 5.8 kB/s | 4.3 kB 00:00  
Rocky Linux 8 - AppStream 772 kB/s | 12 MB 00:15  
Rocky Linux 8 - BaseOS 5.7 kB/s | 4.3 kB 00:00  
Rocky Linux 8 - BaseOS 2.1 MB/s | 7.2 MB 00:03  
Rocky Linux 8 - Extras 4.3 kB/s | 3.1 kB 00:00  
Rocky Linux 8 - Extras 15 kB/s | 14 kB 00:00  
Package gcc-8.5.0-22.el8_10.x86_64 is already installed.  
Dependencies resolved.  
Nothing to do.  
Complete!  
[root@Nzita ~]# setenforce 0  
[root@Nzita ~]# setenforce  
usage: setenforce [ Enforcing | Permissive | 1 | 0 ]  
[root@Nzita ~]#
```

Рис. 1: Подготовка лабораторного стенда

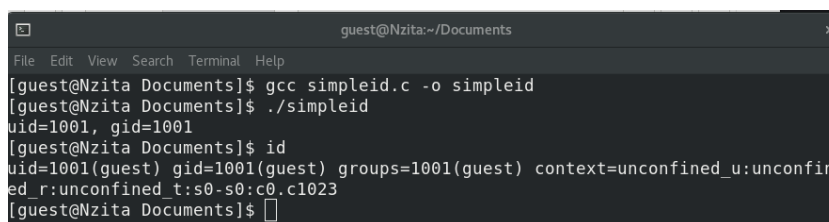
Войдем в систему от имен пользователя guest и создадим программу simpleid.c, которая выводит идентификатор пользователя и группы(рис. @fig:002)



```
*simpleid.c  
~/Documents  
#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. 2: Текст прорааммы simpleid.c

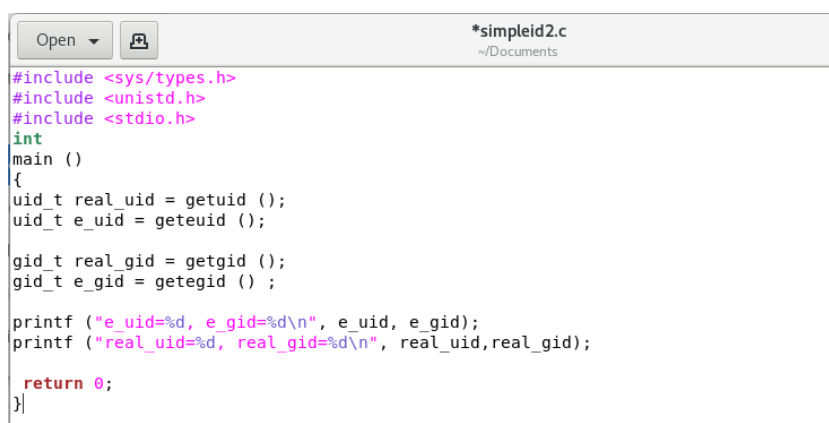
Теперь скомпилируем программу с помощью gcc, затем, запустив её, увидим, что она выводит идентификаторы пользователя и группы 1001 и 1001 для guest, что совпадает с выводом команды id(рис. @fig:003)



```
guest@Nzita:~/Documents
File Edit View Search Terminal Help
[guest@Nzita Documents]$ gcc simpleid.c -o simpleid
[guest@Nzita Documents]$ ./simpleid
uid=1001, gid=1001
[guest@Nzita Documents]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@Nzita Documents]$
```

Рис. 3: Запуск программы simpleid.c

Усложним программу, добавив вывод действительных идентификаторов(рис. @fig:004).



```
*simpleid2.c
~/Documents

#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;
}
```

Рис. 4: Текст программы simpleid2

Теперь скомпилируем программу с помощью gcc, затем, запустив её, увидим, что она выводит идентификаторы пользователя и группы 1001 и 1001 для guest, что совпадает с выводом команды id(рис. @fig:005).

```
guest@Nzita:~/Documents
File Edit View Search Terminal Help
[guest@Nzita Documents]$ gcc simpleid2.c -o simpleid2
[guest@Nzita Documents]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@Nzita Documents]$
```

Рис. 5: Запуск программы simpleid2.c

От имени суперпользователя изменим владельца файла /home/guest/simpleid2 и установим SetUID-бит. Проверим корректность установленных прав и опять запустим simpleid2(рис. @fig:006).

```
[root@Nzita ~]# chown root:guest /home/guest/Documents/simpleid2
[root@Nzita ~]# chown root:guest /home/guest/Documents/simpleid2
[root@Nzita ~]# ls -l /home/guest/Documents/simpleid2
-rwxrwxr-x. 1 root guest 18312 Oct  2 13:34 /home/guest/Documents/simpleid2
[root@Nzita ~]# exit
logout
[nzita@Nzita ~]$ su - guest
Password:
[guest@Nzita ~]$ cd Documents/
[guest@Nzita Documents]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@Nzita Documents]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@Nzita Documents]$ su - Nzita
su: user Nzita does not exist
[guest@Nzita Documents]$ su - nzita
Password:
[nzita@Nzita ~]$ su -
Password:
[root@Nzita ~]# /home/guest/Documents/simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@Nzita ~]#
```

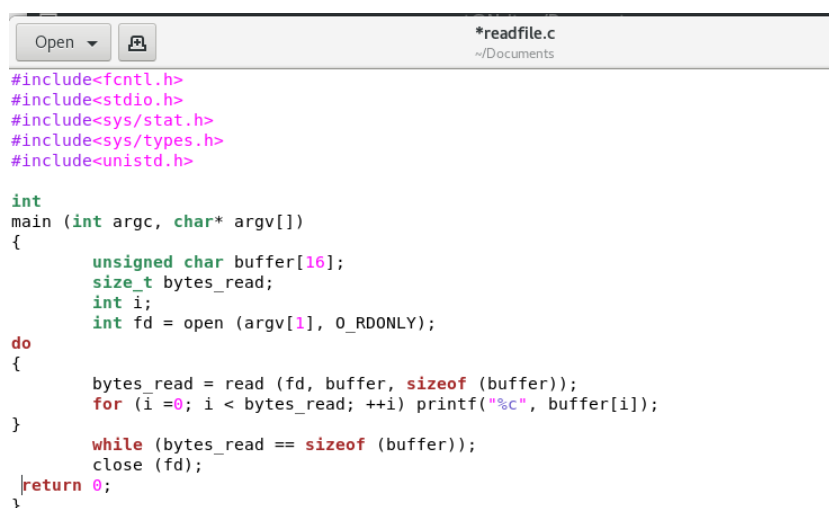
Рис. 6: Изменение владельца и запуск программы simpleid2 с установленным SetUID-битом

Прделаем аналогичные действия относительно SetGID-бита(рис. @fig:007):

!Запуск программы simpleid2 с установленным SetGID-битом

Создадим программу для чтения файлов readfile.c(рис. @fig:008):





```
Open [icon] *readfile.c
~/Documents

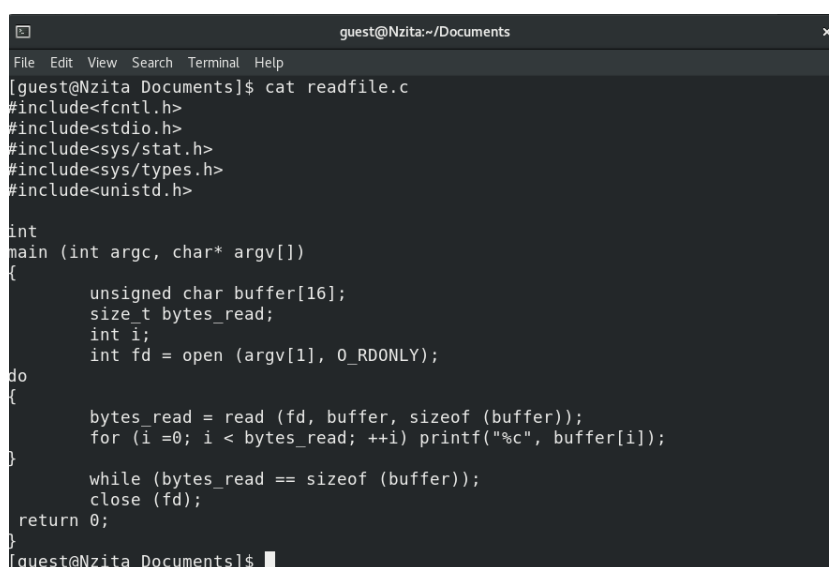
#include<fcntl.h>
#include<stdio.h>
#include<sys/stat.h>
#include<sys/types.h>
#include<unistd.h>

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;
}
```

Рис. 7: Текст программы readfile.c

Скомпилируем её и сменим владельца у файла с текстом программы, затем изменим права так, чтобы только суперпользователь (root) мог прочитать его, и проверим корректность настроек(рис. @fig:009):



```
guest@Nzita:~/Documents
File Edit View Search Terminal Help
[guest@Nzita Documents]$ 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;
}
[guest@Nzita Documents]$
```

Рис. 8: Изменение владельца и прав файла readfile.c

После завершения установки операционной системы корректно перезапустим виртуальную машину и при запросе примем условия лицензии.

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. @fig:010):

```
[guest@Nzita Documents]$ ls -l / | grep tmp
drwxrwxrwt. 14 root root 4096 Oct  2 14:00 tmp
[guest@Nzita Documents]$ echo "test" > /tmp/file01.txt
[guest@Nzita Documents]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 Oct  2 14:04 /tmp/file01.txt
[guest@Nzita Documents]$ chmod o+rw /tmp/file01.txt
[guest@Nzita Documents]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 Oct  2 14:04 /tmp/file01.txt
[guest@Nzita Documents]$ su - guest2
Password:
[guest2@Nzita ~]$ cat /tmp/file01.txt
test
[guest2@Nzita ~]$ echo "test2" > /tmp/file01.txt
[guest2@Nzita ~]$ cat /tmp/file01.txt
test2
[guest2@Nzita ~]$ echo "test3" > /tmp/file01.txt
[guest2@Nzita ~]$ cat /tmp/file01.txt
test3
[guest2@Nzita ~]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory
[guest2@Nzita ~]$ su -
Password:
[root@Nzita ~]# chmod -t /tmp
[root@Nzita ~]# exit
logout
[guest2@Nzita ~]$ ls -l / | grep tmp
drwxrwxrwx. 14 root root 4096 Oct  2 14:10 tmp
[guest2@Nzita ~]$ cat /tmp/file01.txt
cat: /tmp/file01.txt: No such file or directory
[guest2@Nzita ~]$ cat /tmp/file01.txt
test3
[guest2@Nzita ~]$ echo "test3" > /tmp/file01.txt
[guest2@Nzita ~]$ cat /tmp/file01.txt
test3
[guest2@Nzita ~]$ rm /tmp/file01.txt
[guest2@Nzita ~]$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
```

Рис. 9: Подключение образа диска дополнений

# Выводы

В результате выполнения работы были выполнены:

- 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.

## **Список литературы**