

# **Отчет по лабораторной работе №5**

**Информационная безопасность**

Байрамгельдыев Довлетмурат

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
<b>5</b>	<b>Выводы</b>	<b>13</b>
	<b>Список литературы</b>	<b>14</b>

## Список иллюстраций

4.1	Создание файла . . . . .	8
4.2	Программа 1 . . . . .	8
4.3	Компиляция программы 1 . . . . .	9
4.4	Программа 2 . . . . .	9
4.5	Запуск модифицированной программы . . . . .	9
4.6	Программа 3 . . . . .	10
4.7	Чтение, запись и удаление . . . . .	12

## **Список таблиц**

# 1 Цель работы

- Изучение механизмов изменения идентификаторов, применения SetUID- и Sticky-битов
- Получение практических навыков работы в консоли с дополнительными атрибутами

## 2 Задание

- Написание программ
- Изменение владельца файлов и прав доступа на файлы
- Установка и снятие Sticky-бита и проверка доступных действий

## 3 Теоретическое введение

Setuid — это бит разрешения, который позволяет пользователю запускать исполняемый файл с правами владельца этого файла. Другими словами, использование этого бита позволяет нам поднять привилегии пользователя в случае, если это необходимо. Классический пример использования этого бита в операционной системе это команда `sudo`.

На месте, где обычно установлен классический бит `x` (на исполнение), выставлен специальный бит `s`. Это позволяет обычному пользователю системы выполнять команды с повышенными привилегиями без необходимости входа в систему как `root`, зная пароль пользователя `root`.

Принцип работы Setgid очень похож на setuid с отличием, что файл будет запускаться пользователем от имени группы, которая владеет файлом.

Последний специальный бит разрешения – это Sticky Bit . В случае, если этот бит установлен для папки, то файлы в этой папке могут быть удалены только их владельцем. Пример использования этого бита в операционной системе это системная папка `/tmp` . Эта папка разрешена на запись любому пользователю, но удалять файлы в ней могут только пользователи, являющиеся владельцами этих файлов.

Более подробно о см. в [lab-theory?,ruvds?].

## 4 Выполнение лабораторной работы

В качестве первого шага лабораторной работы осуществил вход от лица пользователя guest и создал файл simpleid.c (рис. 4.1).

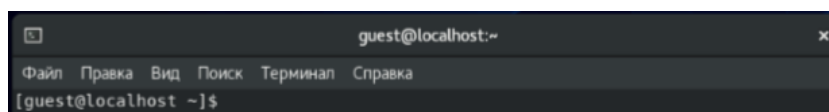


Рис. 4.1: Создание файла

Далее написал программу, которая выводит информацию об идентификаторах пользователя и группы (рис. 4.2). Скомпилировал эту программу (рис. 4.3).



Рис. 4.2: Программа 1



```
[guest@localhost ~]$ gcc simpleid.c -o simpleid
[guest@localhost ~]$ ls
dir1          simpleid      Документы    Музыка        Шаблоны
script1.sh    simpleid.c    Загрузки     Общедоступные
script.sh      Видео         Изображения  'Рабочий стол'
```

Рис. 4.3: Компиляция программы 1

Выполнил эту программу и сравнил ее вывод с результатом работы системной команды id (рис. ??)(рис. ??).

```
[guest@localhost ~]$ ./simpleid
uid=1001, gid=1001

[guest@localhost ~]$ id
uid=1001(guest) gid=1001(guest) rpyнны=1001(guest)
ned_r:unconfined t:s0-s0:c0.c1023
```

Модифицировал программу, чтобы она выводила также действительные идентификаторы (рис. 4.4).

```
simpleid.c  ✕  simpleid2.c

#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. 4.4: Программа 2

Скомпилировал и запустил модифицированную программу (рис. 4.5).

```
[guest@localhost ~]$ gcc simpleid2.c -o simpleid2
[guest@localhost ~]$ ls
dir1          simpleid      simpleid.c    Загрузки     Общедоступные
script1.sh    simpleid2     Видео         Изображения  'Рабочий стол'
script.sh     simpleid2.c   Документы     Музыка        Шаблоны
[guest@localhost ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@localhost ~]$
```

Рис. 4.5: Запуск модифицированной программы

Передал право владения программой суперпользователю и наделил его правом на исполнение программы. Запустили программу и сверили результат с выводом команды `id` (рис. ??) (рис. ??). Запустил `simpleid2` и `id`. Ничего не из-

менилось (рис. ??)

```
[guest@localhost ~]$ su
Пароль:

[guest@localhost ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@localhost ~]$ id
uid=1001(guest) gid=1001(guest) rpyнны=1001(guest) ко
ned r:unconfined t:s0-s0:c0.c1023

[root@localhost guest]# chown root:guest /home/guest/simpleid2
[root@localhost guest]# chmod u+s /home/guest/simpleid2
```

Создал новую программу `readfile`, позволяющую прочесть содержимое файла (рис. 4.6).

```
simpleid.c  ×  simpleid2.c  ×  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;
}
```

Рис. 4.6: Программа 3

Скомпилировал ее, а затем поменял владельца файла `readfile.c` (рис. ??) и изменил права так, чтобы читать файл мог только суперпользователь. Проверил, может ли пользователь `guest` прочитать файл (рис. ??). Ему было отказано в доступе.

```
[root@localhost guest]# chown root:guest /home/guest/readfile.c
[root@localhost guest]# chmod 700 /home/guest/readfile.c

[guest@localhost ~]$ cat /home/guest/readfile.c
cat: readfile.c: Отказа
```

Передал право владения программой суперпользователю и проверили, может ли программа прочитать `readfile.c` (рис. ??) (рис. ??) и `etc/shadow` (рис. ??). Действия были

выполнены успешно.

```
[root@localhost guest]# chown root:guest /home/guest/readfile
[root@localhost guest]# chmod u+s /home/guest/readfile

[guest@localhost ~]$ ./readfile /etc/shadow
root:$6$0C3WkySzs7.eemZD$bktPyMRhX3z7SHE04ngRFdRsDpJUGlfiLs4P8PJ3tVwWATuRTWBuX3B
9Z0Uw25fVDJw7zzupnftXRco5Q09j.:0:99999:7:::
bin:*.18358:0:99999:7:::
daemon:*.18358:0:99999:7:::
adm:*.18358:0:99999:7:::
lp:*.18358:0:99999:7:::
sync:*.18358:0:99999:7:::
shutdown:*.18358:0:99999:7:::
halt:*.18358:0:99999:7:::
mail:*.18358:0:99999:7:::
operator:*.18358:0:99999:7:::
games:*.18358:0:99999:7:::
```

```
[guest@localhost ~]$ ./read
#include <fcntl.h>
#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>
```

Проверил, установлен ли Sticky-бит на директории tmp, создал file01.txt и передал категории пользователей “все остальные” право на чтение и запись

```
[guest@localhost ~]$ echo "test" > /tmp/file01.txt
[guest@localhost ~]$ cat /tmp/file01.txt
test
```

(рис. ??) (рис. ??).

```
[guest2@localhost guest]$ cat /tmp/file01.txt
test
```

От лица пользователя guest2 (не являющегося владельцем файла) попробовал прочесть файл и записать туда новые данные (рис. ??) (рис. ??), а также удалить файл (рис. ??).

```
[guest2@localhost guest]$ cat /tmp/file01.txt
test
test2

[guest2@localhost guest]$ rm /tmp/file01.txt
rm: невозможно удалить '/tmp/file01.txt': Операция не позволена
```

```
[guest2@localhost guest]$ echo "test"
[guest2@localhost guest]$
```

Снял с папки tmp атрибут Sticky (рис. ??) (рис. ??).

```
[guest2@localhost guest]$ su
Пароль:
[root@localhost guest]# chmod -t /tmp
```

```
[root@localhost guest]# exit
```

От лица пользователя guest2 попробовал прочесть файл, произвести в него запись и удалить (рис. 4.7). Удаление файла прошло успешно.

```

[guest2@localhost guest]$ cat /tmp/file01.txt
test3
[guest2@localhost guest]$ echo "test2" >> /tmp/file01.txt
[guest2@localhost guest]$ cat /tmp/file01.txt
test3
test2
[guest2@localhost guest]$ echo "test3" > /tmp/file01.txt
[guest2@localhost guest]$ cat /tmp/file01.txt
test3
[guest2@localhost guest]$ rm /tmp/file01.txt
[guest2@localhost guest]$ ls /tmp
systemd-private-8553bbdafd5a4298b486bab82e5b76a9-bolt.service-rjjeGm
systemd-private-8553bbdafd5a4298b486bab82e5b76a9-colord.service-JRG69N
systemd-private-8553bbdafd5a4298b486bab82e5b76a9-fwupd.service-MNRVCT
systemd-private-8553bbdafd5a4298b486bab82e5b76a9-geoclue.service-TbVhfc
systemd-private-8553bbdafd5a4298b486bab82e5b76a9-ModemManager.service-NJ6cCp
systemd-private-8553bbdafd5a4298b486bab82e5b76a9-rtkit-daemon.service-ZVzhLm
tracker-extract-files.1001

```

Рис. 4.7: Чтение, запись и удаление

Установил атрибут Sticky обратно на папку tmp (рис. ??) (рис. ??).

```

[guest2@localhost guest]$ su
Пароль:
[root@localhost guest]# chmod +t /tmp
[root@localhost guest]# ls -l / | grep tmp

```

```

[root@localhost guest]# exit
[guest2@localhost guest]#

```

## **5 Выводы**

В результате лабораторной работы усовершенствовал навыки использования командой строки и изучил SetUID-, SetGID- и Sticky-биты.

## Список литературы

1. Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов [Электронный ресурс]. URL: <https://esystem.rudn.ru/mod/resource/view.php?id=1031377>.
2. Использование SETUID, SETGID и Sticky Bit для расширенной настройки прав доступа в операционных системах Linux [Электронный ресурс]. URL: <https://ruvds.com/ru/helpcenter/suid-sgid-sticky-bit-linux/>.