

# **Лабораторная работа №5**

**Основы информационной безопасности**

Белов Никита Дмитриевич

# Содержание

|          |                                       |           |
|----------|---------------------------------------|-----------|
| <b>1</b> | <b>Цель работы</b>                    | <b>5</b>  |
| <b>2</b> | <b>Задание</b>                        | <b>6</b>  |
| <b>3</b> | <b>Выполнение лабораторной работы</b> | <b>7</b>  |
| <b>4</b> | <b>Выводы</b>                         | <b>12</b> |

## Список иллюстраций

|      |   |    |
|------|---|----|
| 3.1  | Установка gcc . . . . .                             | 7  |
| 3.2  | Код программы simpleid.c . . . . .                  | 7  |
| 3.3  | Сравнение результатов программы и команды . . . . . | 8  |
| 3.4  | Код программы simpleid2.c . . . . .                 | 8  |
| 3.5  | Компиляция и запуск simpleid2.c . . . . .           | 8  |
| 3.6  | Добавление SetUID . . . . .                         | 9  |
| 3.7  | Сверка результата программы и кода . . . . .        | 9  |
| 3.8  | Код программы readfile.c . . . . .                  | 9  |
| 3.9  | Редактирование прав файла . . . . .                 | 10 |
| 3.10 | Проверка чтения файла . . . . .                     | 10 |
| 3.11 | Проверка от guest2 . . . . .                        | 11 |
| 3.12 | Снятие и возвращение Sticky атрибута . . . . .      | 11 |

## Список таблиц

# 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 Задание**

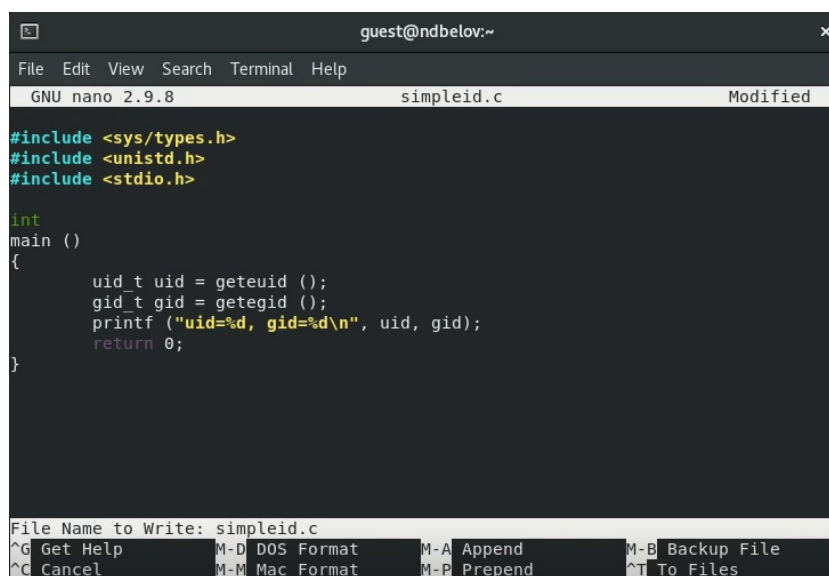
### 3 Выполнение лабораторной работы

Установил gcc с помощью команды `yum install gcc`.

```
gcc version 8.5.0 20210514 (Red Hat 8.5.0-10) (GCC)
[ndbelov@ndbelov ~]$ setenforce 0
setenforce: setenforce() failed
[ndbelov@ndbelov ~]$ sudo setenforce 0
[ndbelov@ndbelov ~]$ getenforce
Permissive
```

Рис. 3.1: Установка gcc

Отменил на текущую сессию SELinux командой `setenforce 0`. Вошёл в систему от имени пользователя `guest`, создал программу `simpleid.c`.



```
guest@ndbelov:~
File Edit View Search Terminal Help
GNU nano 2.9.8 simpleid.c Modified

#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;
}

File Name to Write: simpleid.c
^G Get Help      M-D DOS Format  M-A Append      M-B Backup File
^C Cancel        M-M Mac Format  M-P Prepend     ^T To Files
```

Рис. 3.2: Код программы `simpleid.c`

Скомпилировал программу и убедился, что файл программы создан: `gcc simpleid.c -o simpleid`. Выполнил программу `simpleid`: `./simpleid`. Выполнил

программу `id` и сравнил полученный результат с данными предыдущего пункта задания. Полученные значения `id` совпадают.

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

Рис. 3.3: Сравнение результатов программы и команды

Усложнил программу, добавив вывод действительных идентификаторов, получившуюся программу назвал `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;
}
```

File Name to Write: simpleid2.c

|          |                |             |                 |
|----------|----------------|-------------|-----------------|
| Get Help | M-D DOS Format | M-A Append  | M-B Backup File |
| Cancel   | M-M Mac Format | M-P Prepend | ^T To Files     |

Рис. 3.4: Код программы `simpleid2.c`

Скомпилировал и запустил `simpleid2.c` `gcc simpleid2.c -o simpleid2`, а затем `./simpleid2`.

```
[guest@ndbelov ~]$ gcc simpleid2.c -o simpleid2
[guest@ndbelov ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real uid=1001, real gid=1001
```

Рис. 3.5: Компиляция и запуск `simpleid2.c`

От имени суперпользователя выполнил команды: `chown root:guest /home/guest/simpleid2`, а затем `chmod u+s /home/guest/simpleid2`. Первая команда изменяет права на файл с `guest` на `root`. А затем устанавливает атрибут



SetUID, который запускает программу не с правами пользователя, а с правами владельца файла. Затем выполнил проверку изменений с помощью команды `ls -l simpleid2`. (рис. -fig. 3.6)

```
[guest@ndbelov ~]$ su ndbelov
Password:
[ndbelov@ndbelov guest]$ sudo chown root:guest /home/guest/simpleid2
[sudo] password for ndbelov:
[ndbelov@ndbelov guest]$ sudo chmod u+s /home/guest/simpleid2
```

Рис. 3.6: Добавление SetUID

Запустил `simpleid2` и `id: ./simpleid2, id`. При данном запуске выводы совпадают.

```
[guest@ndbelov ~]$ ls -l simpleid2
-rwsrwxr-x. 1 root guest 18256 Oct  2 17:51 simpleid2
[guest@ndbelov ~]$ ./simpleid2
e uid=0, e gid=1001
real uid=1001, real gid=1001
[guest@ndbelov ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 3.7: Сверка результата программы и кода

Проделал то же самое с атрибутом SetGID (установление прав для владеющей группы). Запустил файл. Теперь выводы для группы различны.

Создал программу `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]);
    }
}
```

Рис. 3.8: Код программы `readfile.c`

Откомпилировал программу: `gcc readfile.c -o readfile`. Сменил владельца у файла `readfile.c` и изменил права так, чтобы только суперпользователь (root)

мог прочитать его, а guest не мог. Проверил, что пользователь guest не может прочитать файл readfile.c

```
[ndbelov@ndbelov guest]$ sudo chown root:guest /home/guest/readfile.c
[sudo] password for ndbelov:
[ndbelov@ndbelov guest]$ sudo chmod 700 /home/guest/readfile.c
[ndbelov@ndbelov guest]$ su guest
Password:
```

Рис. 3.9: Редактирование прав файла

```
[ Error reading readfile.c: Permission denied ]
```

Рис. 3.10: Проверка чтения файла

Сменил у программы readfile владельца и установил SetU'D-бит. Программа readfile может прочитать файл readfile.c. Программа readfile может прочитать файл /etc/shadow. Исследование Sticky-бита. Узнал, установлен ли атрибут Sticky на директории /tmp, для чего выполнил команду `ls -l / | grep tmp`

```
[guest@ndbelov ~]$ ls -l / | grep tmp
drwxrwxrwt. 13 root root 4096 Oct  2 18:09 tmp
[guest@ndbelov ~]$ echo "test" > /tmp/file01.txt
[guest@ndbelov ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 Oct  2 18:10 /tmp/file01.txt
[guest@ndbelov ~]$ chmod o+rw /tmp/file01.txt
[guest@ndbelov ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 Oct  2 18:10 /tmp/file01.txt
```

От имени пользова-

теля guest создал файл file01.txt в директории /tmp со словом test `echo "test" > /tmp/file01.txt`. Просмотрел атрибуты у только что созданного файла и разрешил чтение и запись для категории пользователей «все остальные»: `ls -l /tmp/file01.txt`, `chmod o+rw /tmp/file01.txt`, `ls -l /tmp/file01.txt`. От пользователя guest2 (не являющегося владельцем) попробовал прочитать файл /tmp/file01.txt: `cat /tmp/file01.txt`, записать в файл /tmp/file01.txt текст test3, стерев при этом всю имеющуюся в файле информацию командой `echo "test3" > /tmp/file01.txt`. Проверил содержимое файла командой `cat /tmp/file01.txt`, попробовал дозаписать в файл /tmp/file01.txt слово test2 командой `echo "test2" >> /tmp/file01.txt`, удалить файл /tmp/file01.txt командой `rm /tmp/file01.txt`. Файл удалить не удалось.

```
[guest@ndbelov ~]$ su guest2
Password:
[guest2@ndbelov guest]$ cat /tmp/file01.txt
test
[guest2@ndbelov guest]$ echo "test" > /tmp/file01.txt
[guest2@ndbelov guest]$ echo "test2" > /tmp/file01.txt
[guest2@ndbelov guest]$ cat /tmp/file01.txt
test2
[guest2@ndbelov guest]$ echo "test3" > /tmp/file01.txt
[guest2@ndbelov guest]$ cat /tmp/file01.txt
test3
[guest2@ndbelov guest]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': Operation not permitted
```

Рис. 3.11: Проверка от guest2

Повысил свои права до суперпользователя следующей командой `su -` и выполнил после этого команду, снимающую атрибут `t` (Sticky-бит) с директории `/tmp`: `chmod -t /tmp`. Затем попробовал выполнить все вышеперечисленные операции. Все удалось.

Повысил свои права до суперпользователя и вернул атрибут `t` на директорию `/tmp`: `su -`, `chmod +t /tmp`, `exit`.

```
[guest2@ndbelov guest]$ su -
Password:
[root@ndbelov ~]# chmod -t /tmp
[root@ndbelov ~]# exit
logout
[guest2@ndbelov guest]$ ls -l / | grep tmp
drwxrwxrwx. 13 root root 4096 Oct 2 18:14 tmp
[guest2@ndbelov guest]$ su -
Password:
[root@ndbelov ~]# chmod +t /tmp
[root@ndbelov ~]# exit
logout
[guest2@ndbelov guest]$ ls -l / | grep tmp
drwxrwxrwt. 14 root root 4096 Oct 2 18:15 tmp
```

Рис. 3.12: Снятие и возвращение Sticky атрибута

## 4 Выводы

Изучил механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получил практические навыки работы в консоли с дополнительными атрибутами. Рассмотрел работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.