

Отчёт

по лабораторной работе 5

Лекомцева Алёна

Содержание

Цель работы	1
Задание	1
Выполнение лабораторной работы	1
Выводы	7

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.

Задание

Лабораторная работа подразумевает создание программ и использование Sticky-бита.

Выполнение лабораторной работы

1. Войдем в систему от имени пользователя guest. (рис.1).
2. Создаем программу simpleid.c и компилируем ее. (рис.1).

```
[guest@aalekomceva ~]$ nano simpleid.c
[guest@aalekomceva ~]$ gcc simpleid.c -o simpleid

GNU nano 5.6.1 simpleid.c
#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.1. Программа simpleid.c.

3. Выполним программы simpleid и id. Сравним полученные нами результаты. Они совпадают. (рис.2).

```
[guest@aalekomceva ~]$ ./simpleid
uid=1001, gid=1001
[guest@aalekomceva ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi
ned_r:unconfined_t:s0-s0:c0.c1023
[guest@aalekomceva ~]$ nano simpleid.c
```

рис.2. Выполнение программы.

4. Усложним программу, добавив вывод действительных идентификаторов. Получившуюся программу назовем simpleid2.c. (рис.3).

```

[guest@aalekomceva ~]$ nano simpleid2.c
[guest@aalekomceva ~]$ gcc simpleid2.c -o simpleid2
GNU nano 5.6.1 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.3. Программа simpleid2.c.

5. Скомпилируем и запустим simpleid2.c. (рис.4).

```

[guest@aalekomceva ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@aalekomceva ~]$

```

рис.4. Выполнение программы simpleid2.c.

6. От имени суперпользователя выполним команды: `chown root:guest /home/guest/simpleid2` и `chmod u+s /home/guest/simpleid2`. Первая меняет владельца файла, а вторая устанавливает SetUID-бит. (рис.5).

```

[guest@aalekomceva ~]$ su -
Пароль:
[root@aalekomceva ~]# chown root:guest /home/guest/simpleid2
chown: невозможно получить доступ к 'guest': Нет такого файла или каталога
[root@aalekomceva ~]# chown root:guest /home/guest/simpleid2
[root@aalekomceva ~]# chmod u+s /home/guest/simpleid2
[root@aalekomceva ~]#

```

рис.5. Смена владельца и атрибутов.

7. Выполним проверку правильности установки новых атрибутов и смены владельца файла simpleid2 (рис.6).

```
[guest@aalekomceva ~]$ ls -l simpleid2
-rwsrwxr-x. 1 root guest 26048 фев  5 22:38 simpleid2
[guest@aalekomceva ~]$
```

рис.6. Проверка смены владельца и атрибутов.

8. Запустим simpleid2 и id и сравним результат. Результаты одинаковы. (рис.7).

```
[guest@aalekomceva ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@aalekomceva ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi
ned_r:unconfined_t:s0-s0:c0.c1023
[guest@aalekomceva ~]$
```

рис.7. Программы simpleid2 и id.

9. Прделаем тоже самое относительно SetGID-бита (рис.8).

```
[root@aalekomceva ~]# chmod g+s /home/guest/simpleid2
[guest@aalekomceva ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@aalekomceva ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi
ned_r:unconfined_t:s0-s0:c0.c1023
[guest@aalekomceva ~]$
```

рис.8. Действия относительно SetGID-бита.

10. Создаем программу readfile.c и компилируем ее. (рис.9).

```
[guest@aalekomceva ~]$ nano readfile.c
[guest@aalekomceva ~]$ gcc readfile.c -o readfile

GNU nano 5.6.1 readfile.c Изменён
#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.9. Программа readfile.c.

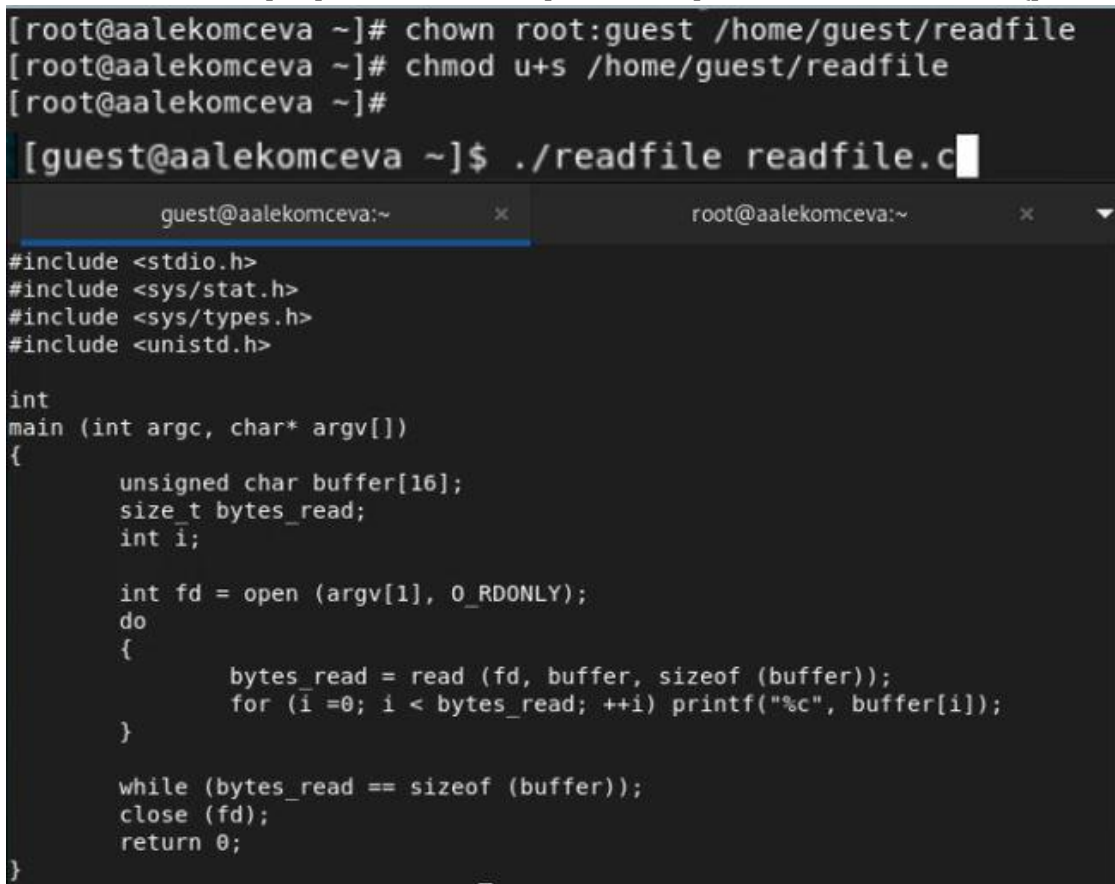
11. Сменим владельца у файла readfile.c и изменим права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог. Проверим, что пользователь guest не может прочитать файл readfile.c. (рис.10).

```
[root@aalekomceva ~]# chown root:guest /home/guest/readfile.c
[root@aalekomceva ~]# chmod 000 /home/guest/readfile.c
[root@aalekomceva ~]#
[guest@aalekomceva ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@aalekomceva ~]$
```

рис.10. Работа с программой readfile.c.

12. Сменим у программы readfile владельца и установим SetUID-бит. Проверим, может ли программа readfile прочитать файл readfile.c. Может. (рис.11).

```
[root@aalekomceva ~]# chown root:guest /home/guest/readfile
[root@aalekomceva ~]# chmod u+s /home/guest/readfile
[root@aalekomceva ~]#
[guest@aalekomceva ~]$ ./readfile readfile.c
```



```
#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.11. Программа readfile читает файл readfile.c.

13. Проверим, может ли программа readfile прочитать файл /etc/shadow. Может. (рис.12).

```
[guest@aalekomceva ~]$ ./readfile /etc/shadow
gnome-initial-setup:!!:19028::::::
sshd:!!:19028::::::
chrony:!!:19028::::::
dnsmasq:!!:19028::::::
tcpdump:!!:19028::::::
systemd-oom:!:*:19028::::::
systemd-resolve:!:*:19028::::::
aalekomceva:$6$Is9abtItNl1fQ490$MLCfNtHZiKtwB9pSudCCGGIHh8Nknf0xTZChuX3f.PH9eSZx
eyb/0EfeZfQL0eVLnB3yImKCZ7J72G27bbPam/:19028:0:99999:7:::
guest:$6$cbQ1SZLf9YR/maan$7LDMSYLekWQN/HhJ6ZfC0Qfn.3tbm4VbRRju3zin4JtvZYptHKXczn
Ys/fr5i6cdI4i.p69FgWYAXlbdDxdC51:19028:0:99999:7:::
guest2:$6$CNtsvTk1oEL/zLzH$fKLJ2Yz.jsk89r9kYtXhK.uKSDC3ewlnnhd5Xpa8RTW9YM0AXiUPj
UGVFLX22gMqfAVHNCUthGvZV05IMLCgb0:19028:0:99999:7:::
[guest@aalekomceva ~]$
```

рис.12. Программа readfile читает файл /etc/shadow.

14. Выясним, установлен ли атрибут Sticky на директории /tmp/. Установлен. От имени пользователя guest создадим файл file01.txt в директории /tmp со словом test. Просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные». Проверим правильность выполнения команд. (рис.13).

```
[guest@aalekomceva ~]$ ls -l / | grep tmp
drwxrwxrwt. 16 root root 4096 фев  5 23:47 tmp
[guest@aalekomceva ~]$ echo "test" > /tmp/file01.txt
[guest@aalekomceva ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 фев  5 23:52 /tmp/file01.txt
[guest@aalekomceva ~]$ chmod o+rw /tmp/file01.txt
[guest@aalekomceva ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 фев  5 23:52 /tmp/file01.txt
[guest@aalekomceva ~]$
```

рис.13. Атрибут Sticky.

15. От пользователя guest2 (не являющегося владельцем) попробуем прочитать, дозаписать, перезаписать и удалить файл /tmp/file01.txt. Удалось выполнить все команды, кроме удаления файла. (рис.14).


```
[guest@aalekomceva ~]$ su guest2
Пароль:
[guest2@aalekomceva guest]$ cat /tmp/file01.txt
test
[guest2@aalekomceva guest]$ echo "test2" >> /tmp/file01.txt
[guest2@aalekomceva guest]$ cat /tmp/file01.txt
test
test2
[guest2@aalekomceva guest]$ echo "test3" > /tmp/file01.txt
[guest2@aalekomceva guest]$ cat /tmp/file01.txt
test3
[guest2@aalekomceva guest]$ rm /tmp/file01.txt
rm: невозможно удалить '/tmp/file01.txt': Операция не позволена
[guest2@aalekomceva guest]$
```

рис.14. Атрибут Sticky.

16. Повысим свои права до суперпользователя командой su - и выполним после этого команду, снимающую атрибут t (Sticky-бит) с директории /tmp. Повторим предыдущие шаги от пользователя guest2. Нам удалось удалить файл от имени пользователя, не являющегося его владельцем. (рис.15).

```
[guest2@aalekomceva guest]$ cat /tmp/file01.txt
test3
[guest2@aalekomceva guest]$ echo "test2" >> /tmp/file01.txt
[guest2@aalekomceva guest]$ cat /tmp/file01.txt
test3
test2
[guest2@aalekomceva guest]$ echo "test" > /tmp/file01.txt
[guest2@aalekomceva guest]$ cat /tmp/file01.txt
test
[guest2@aalekomceva guest]$ rm /tmp/file01.txt
[guest2@aalekomceva guest]$ su -
Пароль:
[root@aalekomceva ~]# chmod +t tmp
chmod: невозможно получить доступ к 'tmp': Нет такого файла или каталога
[root@aalekomceva ~]# chmod +t /tmp
[root@aalekomceva ~]# exit
выход
[guest2@aalekomceva guest]$
```

рис.15. Выполнение команд без атрибута Sticky.

Выводы

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.