

Отчёт

по лабораторной работе 5

Агеева Анастасия Борисовна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	14

List of Figures

3.1	рис.1. Программа simpleid.c.	7
3.2	рис.2. Компиляция и выполнение программы.	7
3.3	рис.3. Программа simpleid2.c.	8
3.4	рис.4. Работа с программой simpleid2.c.	8
3.5	рис.5. Действия относительно SetGID-бита.	9
3.6	рис.6. Программа readfile.c.	9
3.7	рис.7. Работа с программой readfile.c.	10
3.8	рис.8. Программа readfile читает файл readfile.c.	10
3.9	рис.9. Программа readfile читает файл /etc/shadow	11
3.10	рис.10. Атрибут Sticky.	11
3.11	рис.11. Работа от пользователя guest2.	12
3.12	рис.12. Возвращение атрибута t на дир /tmp.	13

List of Tables

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

Лабораторная работа подразумевает создание программы и использование Sticky-бита.

3 Выполнение лабораторной работы

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

```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int
main ()
{
    uid_t uid = geteuid ();
    gid_t gid = geteuid ();
    printf ("uid=%d, gid=%d\n", uid, gid);
    return 0;
}
```

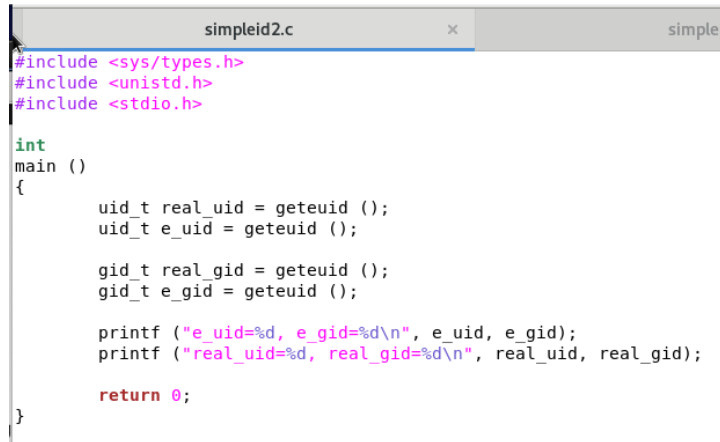
Figure 3.1: рис.1. Программа simpleid.c.

3. Скомпилируем программу и убедимся, что файл программы создан (рис.2).
4. Выполним программу simpleid (рис.2).
5. Выполним системную программу id и сравним полученный нами результат с данными предыдущего пункта задания (рис.2). Результаты совпадают

```
[guest@abageeval ~]$ gcc simpleid.c -o simpleid
[guest@abageeval ~]$ ./simpleid
uid=1001, gid=1001
[guest@abageeval ~]$ id
uid=1001(guest) gid=1001(guest) rгруппы=1001(guest),10(wheel) контекст=unconfined_u:unco
nfinеd_r:unconfined_t:s0-s0:c0.c1023
[guest@abageeval ~]$ █
```

Figure 3.2: рис.2. Компиляция и выполнение программы.

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



```
simpleid2.c x simplei
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int
main ()
{
    uid_t real_uid = geteuid ();
    uid_t e_uid = geteuid ();


    gid_t real_gid = getegid ();
    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;
}
```

Figure 3.3: рис.3. Программа simpleid2.c.

7. Скомпилируем и запустим simpleid2.c (рис.4).
8. От имени суперпользователя выполним команды: `chown root:guest /home/guest/simpleid2` и `chmod u+s /home/guest/simpleid2` (рис.4).
9. Повысим временно свои права с помощью `su`. (рис.4).
10. Выполним проверку правильности установки новых атрибутов и смены владельца файла simpleid2 (рис.4).
11. Запустим simpleid2 и id и сравним результат. Результаты одинаковы. (рис.4).



```
[guest@abageeval ~]$ nano simpleid2.c
[guest@abageeval ~]$ gcc simpleid2.c -o simpleid2
[guest@abageeval ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@abageeval ~]$ id
uid=1001(guest) gid=1001(guest) rгруппы=1001(guest),10(wheel) контекст=unconfined_u:unco
nfinеd_r:unconfined_t:s0-s0:c0.c1023
[guest@abageeval ~]$ su
Пароль:
[root@abageeval guest]# chown root:guest /home/guest/simpleid2
[root@abageeval guest]# chmod u+s /home/guest/simpleid2
[root@abageeval guest]# ls -l simpleid2
-rwsrwxr-x, 1 root guest 8656 ноя 11 23:30 simpleid2
[root@abageeval guest]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@abageeval guest]# id
uid=0(root) gid=0(root) rгруппы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:
s0-s0:c0.c1023
[root@abageeval guest]#
```

Figure 3.4: рис.4. Работа с программой simpleid2.c.

12. Проделаем тоже самое относительно SetGID-бита (рис.5).

```
[root@abageeval guest]# chmod g+s /home/guest/simpleid2
[root@abageeval guest]# ls -l simpleid2
-rwsrwsr-x. 1 root guest 8656 ноя 11 23:30 simpleid2
[root@abageeval guest]# ./simpleid2
e_uid=0, e_gid=1001
real_uid=0, real_gid=0
[root@abageeval guest]# id I
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:
s0-s0:c0.c1023
[root@abageeval guest]#
```

Figure 3.5: рис.5. Действия относительно SetGID-бита.

13. Создаем программу readfile.c (рис.6).

```
#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) print("%c", buffer[i]);
    }

    while (bytes_read == sizeof (buffer));
    close (fd);
    return 0;
}
```

Figure 3.6: рис.6. Программа readfile.c.

14. Откомпилируем её (рис.7).

15. Сменим владельца у файла readfile.c (или любого другого текстового файла в системе) и изменим права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог (рис.7).

16. Проверим, что пользователь guest не может прочитать файл readfile.c (рис.7).

17. Сменим у программы readfile владельца и установим SetU'D-бит (рис.7).

```

[root@abageeval guest]# gcc readfile.c -o readfile
[root@abageeval guest]# chmod a-r readfile.c
[root@abageeval guest]# exit
exit
[guest@abageeval ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@abageeval ~]$ su
Пароль:
[root@abageeval guest]# chown root:guest readfile
[root@abageeval guest]# chmod u+s readfile
[root@abageeval guest]# exit
exit
[guest@abageeval ~]$ █

```

Figure 3.7: рис.7. Работа с программой readfile.c.

18. Проверим, может ли программа readfile прочитать файл readfile.c (рис.8).

```

[guest@abageeval ~]$ ./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;
    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@abageeval ~]$ █

```

Figure 3.8: рис.8. Программа readfile читает файл readfile.c.

19. Проверим, может ли программа readfile прочитать файл /etc/shadow (рис.9).

```
[guest@abageeval ~]$ ./readfile /etc/shadow
root:$6$zzkG4rnjRBsC.dE$krIs2Eh4/jZ6lqe.u0b6/ngdXdsETCeMwsQerJU5LByPFm0QY7uGdUjYYK0mP
A6sveVqrCSUyyrrEWOAIJ05.:0:99999:7:::
bin:*:17834:0:99999:7:::
daemon:*:17834:0:99999:7:::
adm:*:17834:0:99999:7:::
lp:*:17834:0:99999:7:::
sync:*:17834:0:99999:7:::
shutdown:*:17834:0:99999:7:::
halt:*:17834:0:99999:7:::
mail:*:17834:0:99999:7:::
operator:*:17834:0:99999:7:::
games:*:17834:0:99999:7:::
ftp:*:17834:0:99999:7:::
nobody:*:17834:0:99999:7:::
systemd-networkd:!:18900:0:0:
dbus:!:18900:0:0:
polkitd:!:18900:0:0:
libstoragemgmt:!:18900:0:0:
colord:!:18900:0:0:
rpc:!:18900:0:99999:7:::
saslauth:!:18900:0:0:
abrt:!:18900:0:0:
rtkit:!:18900:0:0:
pulse:!:18900:0:0:
radvd:!:18900:0:0:
```

Figure 3.9: рис.9. Программа readfile читает файл /etc/shadow

20. Выясним, установлен ли атрибут Sticky на директории /tmp (рис.10).
21. От имени пользователя guest создадим файл file01.txt в директории /tmp со словом test (рис.10).
22. Просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные» (рис.10).

```
[guest@abageeval ~]$ ls -l / | grep tmp
drwxrwxrwt. 16 root root 4096 ноя 11 23:49 tmp
[guest@abageeval ~]$ echo "test" > /tmp/file01.txt
[guest@abageeval ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 ноя 11 23:52 /tmp/file01.txt
[guest@abageeval ~]$ chmod o+rw /tmp/file01.txt
[guest@abageeval ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 ноя 11 23:52 /tmp/file01.txt
[guest@abageeval ~]$ █
```

Figure 3.10: рис.10. Атрибут Sticky.

23. От пользователя guest2 (не являющегося владельцем) попробуем прочитать файл /tmp/file01.txt (рис.11).
24. От пользователя guest2 попробуем дозаписать в файл /tmp/file01.txt слово test2 (рис.11).
25. Проверим содержимое файла (рис.11).

26. От пользователя guest2 попробуем записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию (рис.11).
27. Проверим содержимое файла командой (рис.11).
28. От пользователя guest2 попробуем удалить файл /tmp/file01.txt (рис.11).
29. Повысим свои права до суперпользователя командой su - и выполним после этого команду, снимающую атрибут t (Sticky-бит) с директории /tmp (рис.11).
30. Покинем режим суперпользователя (рис.11).
31. От пользователя guest2 проверим, что атрибута t у директории /tmp нет (рис.11).
32. Повторим предыдущие шаги (рис.11).
33. Нам удалось удалить файл от имени пользователя, не являющегося его владельцем (рис.10-11).

```
[guest@abageeval ~]$ su guest2
Пароль:
[guest2@abageeval guest]$ cat /tmp/file01.txt
test
[guest2@abageeval guest]$ echo "test2" > /tmp/file01.txt
[guest2@abageeval guest]$ cat /tmp/file01.txt
test2
[guest2@abageeval guest]$ echo "test3" > /tmp/file01.txt
[guest2@abageeval guest]$ cat /tmp/file01.txt
test3
[guest2@abageeval guest]$ rm /tmp/file01.txt
rm: невозможно удалить «/tmp/file01.txt»: Операция не позволена
[guest2@abageeval guest]$ su -
Пароль:
Последний вход в систему: 4т ноя 11 23:48:35 MSK 2021 на pts/1
[root@abageeval ~]# chmod -t /tmp
[root@abageeval ~]# exit
logout
[guest2@abageeval guest]$ ls -l / | grep tmp
drwxrwxrwx. 16 root root 4096 ноя 11 23:59 tmp
[guest2@abageeval guest]$ cat /tmp/file01.txt
test3
[guest2@abageeval guest]$ echo "test2" > /tmp/file01.txt
[guest2@abageeval guest]$ cat /tmp/file01.txt
test2
[guest2@abageeval guest]$ rm /tmp/file01.txt
```

Figure 3.11: рис.11. Работа от пользователя guest2.

34. Повысим свои права до суперпользователя и вернем атрибут `t` на директорию `/tmp` (рис.12).

```
[guest2@abageeval guest]$ su -  
Пароль:  
Последний вход в систему: Чт ноя 11 23:58:45 MSK 2021 на pts/1  
[root@abageeval ~]# chmod +t /tmp  
[root@abageeval ~]# exit  
logout  
[guest2@abageeval guest]$ █
```

Figure 3.12: рис.12. Возвращение атрибута `t` на дир `/tmp`.

4 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.