Отчет по лабораторной работе №5

по дисциплине: Информационная безопасность

Логинов Егор Игоревич

Содержание

1 Цели работы **4**

2 Задание **5**

3 Теоретическое введение **6**

4 Выполнение лабораторной работы **7**

5 Выводы **15**

6 Список литературы **16**

Список иллюстраций

4.1 Использование команд ./simpleid и id 7
4.2 Запуск программы simpleid2 8
4.3 Установки новых атрибутов и смена владельца файла simpleid2 . 9
4.4 Использование команд ./simpleid2 9
4.5 Операции с SetGID-битом 9
4.6 Изменение владельца и прав файла readfile.c 10
4.7 Работа с параметрами readfile 10
4.8 Попытка прочитать файл readfile.c программой readfile 11
4.9 Попытка прочитать файл /etc/shadow программой readfile 12
4.10 Чтение атрибутов директории /tmp 12
4.11 Создание файла /tmp/file01.txt 13
4.12 Работа с файлом /tmp/file01.txt 13
4.13 Удаление атрибута t директории /tmp 14

1 Цели 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

1. Исследовать SetUID- и SetGID-биты.
2. Исследовать Sticky-бит.

3 Теоретическое введение

- Операционная система — это комплекс программ, предназначенных для управления ресурсами компьютера и организации взаимодействия с пользователем [1].
- Права доступа определяют, какие действия конкретный пользователь может или не может совершать с определенным файлами и каталогами. С помощью разрешений можно создать надежную среду — такую, в которой никто не может поменять содержимое ваших документов или повредить системные файлы. [2].

4 Выполнение лабораторной работы

1. От имени пользователя guest создадим программу simpleid.c, скомпилируем ее и убедимся, что файл создан. Выполним команды ./simpleid и id и убедимся, что полученные данные совпадают (4.1).

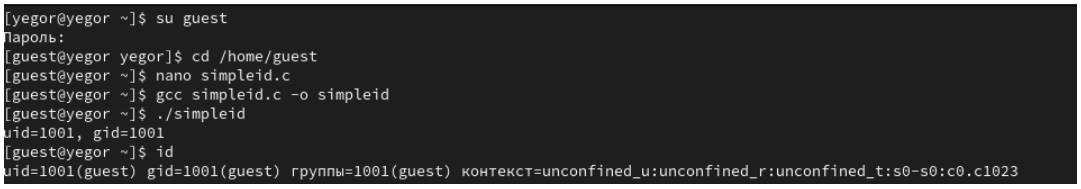

Рис. 4.1: Использование команд ./simpleid и id

Код программы 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;
}
```

2. Усложним программу и запишем ее в файл simpleid2.c. Запустим получившуюся программу и сверим результат с командой id (4.2).

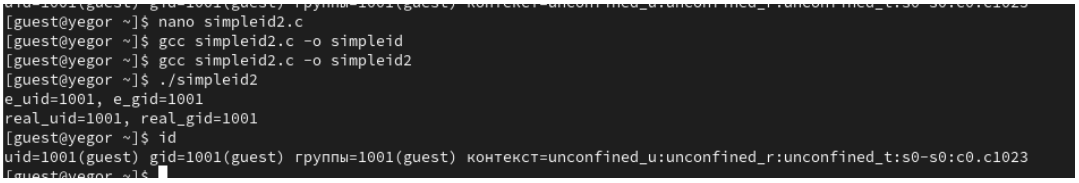

Рис. 4.2: Запуск программы simpleid2

Код программы 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;
}
```

3. От имени суперпользователя установим новые атрибуты и сменим владельца файла simpleid2 (4.3).

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

Рис. 4.3: Установки новых атрибутов и смена владельца файла simpleid2

4. Выполним команды ./simpleid2 и id (4.4).

```
[root@yegor guest]# ls -l simpleid2
-rwsr-xr-x. 1 root guest 26064 окт  7 20:24 simpleid2
[root@yegor guest]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@yegor guest]# id
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@yegor guest]#
```

Рис. 4.4: Использование команд ./simpleid2

5. Проделаем то же самое относительно SetGID-бита (4.5).

```
[root@yegor guest]# chmod g+s simpleid2
[root@yegor guest]# ls -l simpleid2
-rwsr-sr-x. 1 root guest 26064 окт  7 20:24 simpleid2
[root@yegor guest]# exit
выход
[guest@yegor ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@yegor ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@yegor ~]$
```

Рис. 4.5: Операции с SetGID-битом

6. Создадим и скомпилируем программу readfile.c. Сменим владельца у файла readfile.c и изменим права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог (4.6).

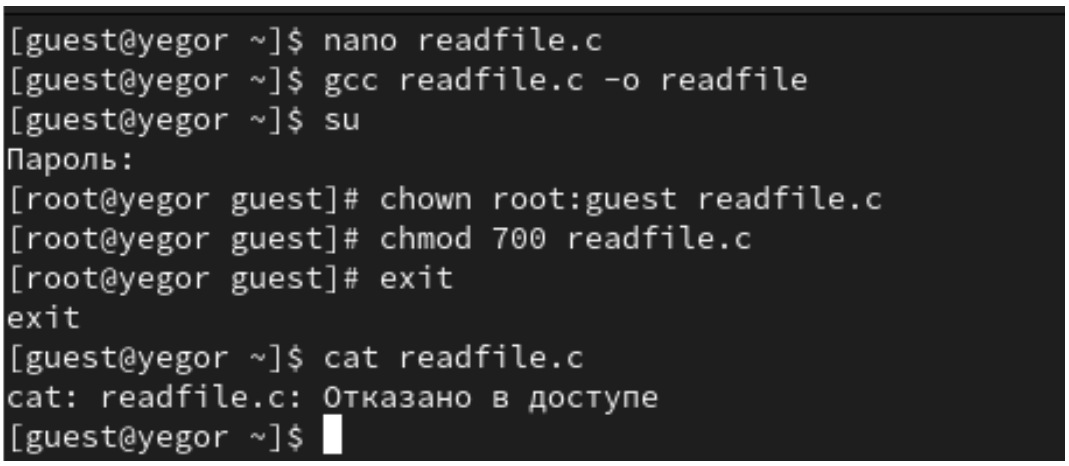

Рис. 4.6: Изменение владельца и прав файла readfile.c

Пользователь guest не может прочитать файл readfile.c

7. Сменим у программы readfile владельца и установим SetUID-бит (4.7).

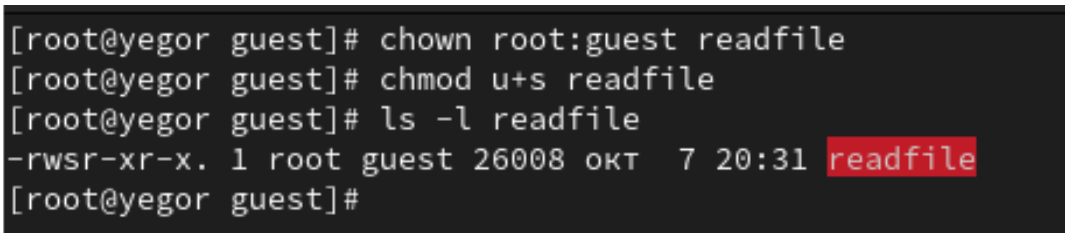

Рис. 4.7: Работа с параметрами readfile

8. Проверим, может ли программа readfile прочитать файл readfile.c (4.8).

```
[guest@yegor ~]$ ./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;
}
```

Рис. 4.8: Попытка прочитать файл readfile.c программой readfile

9. Проверим, может ли программа readfile прочитать файл /etc/shadow (4.9).

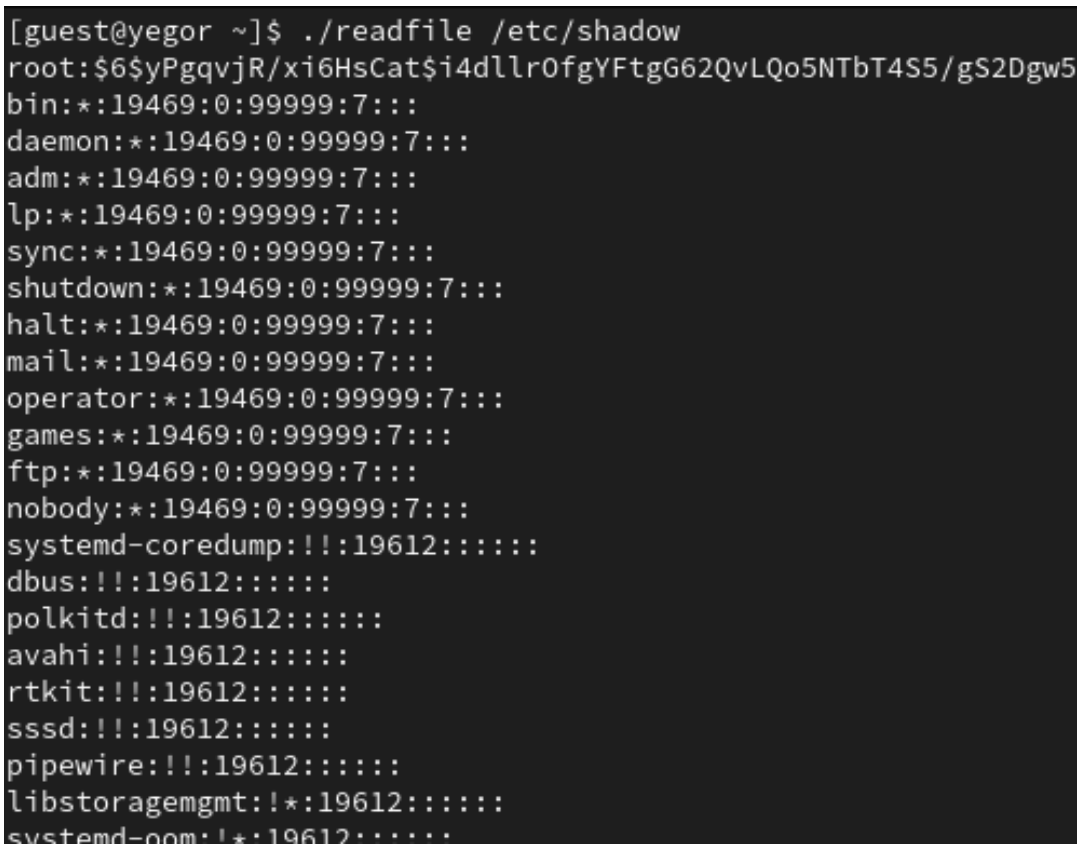

Рис. 4.9: Попытка прочитать файл /etc/shadow программой readfile

10. Выясним, установлен ли атрибут Sticky на директории /tmp (4.10).

Рис. 4.10: Чтение атрибутов директории /tmp

11. 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» (4.11).

```
[guest@yegor ~]$ echo "test" > /tmp/file01.txt
[guest@yegor ~]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 окт  7 20:36 /tmp/file01.txt
[guest@yegor ~]$ chmod o+rw /tmp/file01.txt
[guest@yegor ~]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 окт  7 20:36 /tmp/file01.txt
[guest@yegor ~]$
```

Рис. 4.11: Создание файла /tmp/file01.txt

12. От пользователя guest2 попробуем прочитать файл /tmp/file01.txt. Далее попробуем дозаписать в файл /tmp/file01.txt слово test2, записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию. После этого попробуем удалить данных файл (4.12).

```
[guest@yegor ~]$ su guest2
Пароль:
[guest2@yegor guest]$ echo "test2" >> /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@yegor guest]$ cat /tmp/file01.txt
test
[guest2@yegor guest]$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@yegor guest]$ cat /tmp/file01.txt
test
[guest2@yegor guest]$ rm /tmp/file01.txt
rm: удалить защищённый от записи обычный файл '/tmp/file01.txt'? y
rm: невозможно удалить '/tmp/file01.txt': Операция не позволена
[guest2@yegor guest]$
```

Рис. 4.12: Работа с файлом /tmp/file01.txt

Пользователь guest2 принадлежит группе guest, поэтому у него нет доступа к вышеописанным действиям, так как у группы нет права доступа на запись для данного файла.

13. От имени суперпользователя снимем атрибут t с директории /tmp. От пользователя guest2 проверим, что атрибута t у директории /tmp нет. Повторим предыдущие шаги. Теперь мы можем удалить файл. (4.13).

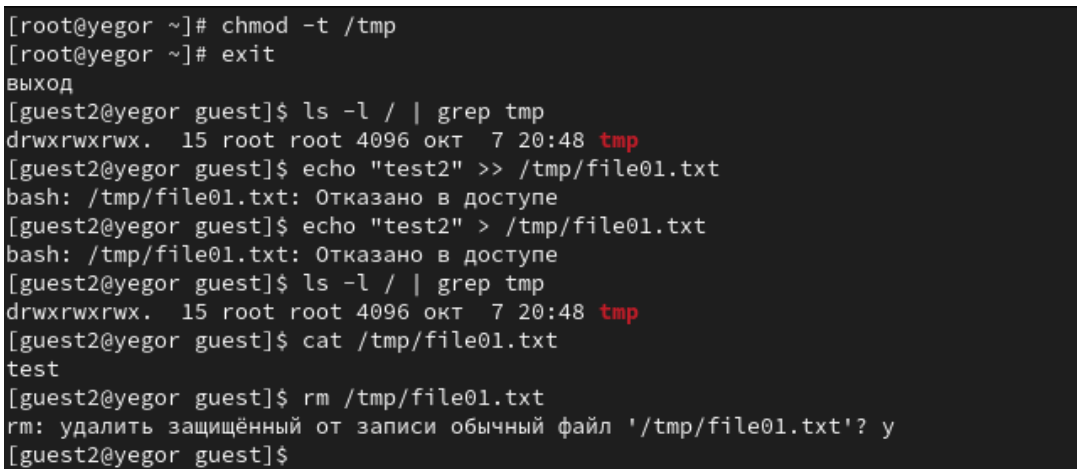

Рис. 4.13: Удаление атрибута t директории /tmp

5 Выводы

В рамках данной лабораторной работы были изучены механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получены практические навыков работы в консоли с дополнительными атрибутами. Рассмотрены принципы работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

6 Список литературы

1. Операционные системы [Электронный ресурс]. URL: https://softline.tm/solutions/programmnoe-obespechenie/operating-system.

2. Права доступа [Электронный ресурс]. URL: https://w.wiki/7UBB.