

Отчёта по лабораторной работе №4

дисциплина: Информационная безопасность

Кашкин Иван Евгеньевич

Содержание

Цель работы.....	
Задание	
Теоретическое введение	
Выполнение лабораторной работы	
Выводы.....	
Список литературы.....	

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.

```
iekashkin@localhost:~/Lab5
6 | uid_t = geteuid();
   | ^
simpleid.c:7:7: error: expected identifier or '(' before '=' token
7 | gid_t = getegid();
   | ^
simpleid.c:8:28: error: 'uid' undeclared (first use in this function)
8 | printf("uid=%d, gid=%d\n", uid, gid);
   |                            ^
simpleid.c:8:28: note: each undeclared identifier is reported only once for each
function it appears in
simpleid.c:8:33: error: 'gid' undeclared (first use in this function); did you m
ean 'gid_t'?
8 | printf("uid=%d, gid=%d\n", uid, gid);
   |                                ^
   |                                gid_t

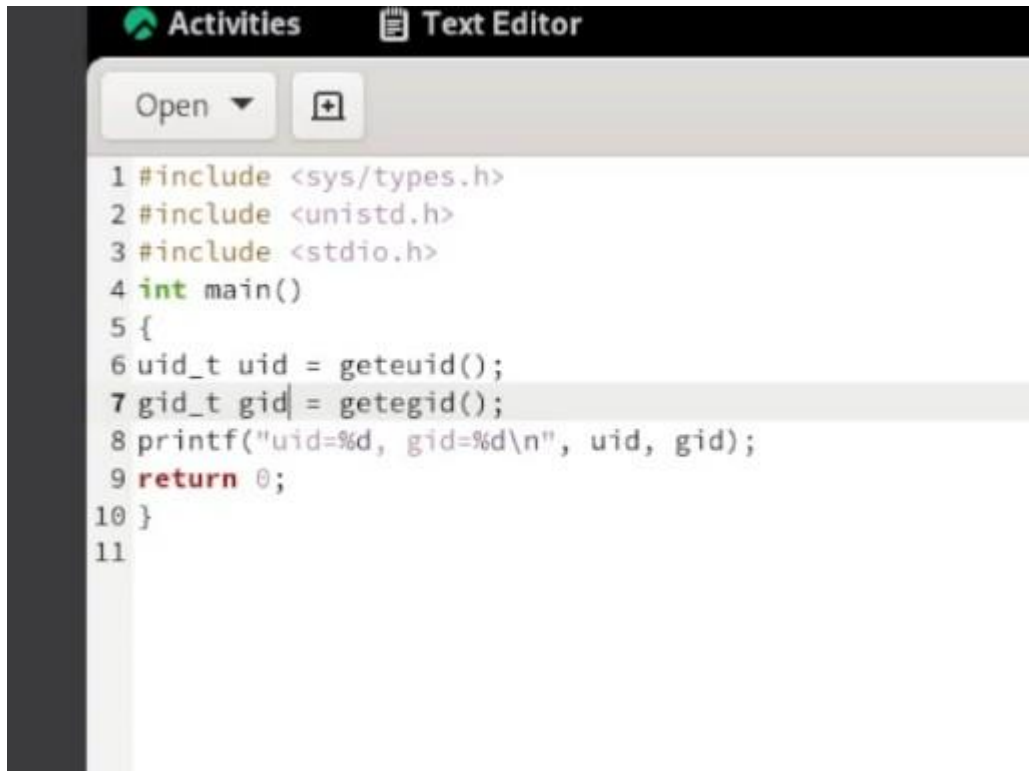
iekashkin@localhost Lab5]$ gedit simpleid.c
iekashkin@localhost Lab5]$ gcc simpleid.c
iekashkin@localhost Lab5]$ gcc simpleid.c -o simpleid
iekashkin@localhost Lab5]$ ./simpleid
uid=1000, gid=1000
iekashkin@localhost Lab5]$ id
uid=1000(iekashkin) gid=1000(iekashkin) groups=1000(iekashkin) context=unconfine
d_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
iekashkin@localhost Lab5]$
```

Выполнение лабораторной работы

Изучение механики SetUID

1. Вошли в систему от имени пользователя guest.
2. Написали программу simpleid.c.

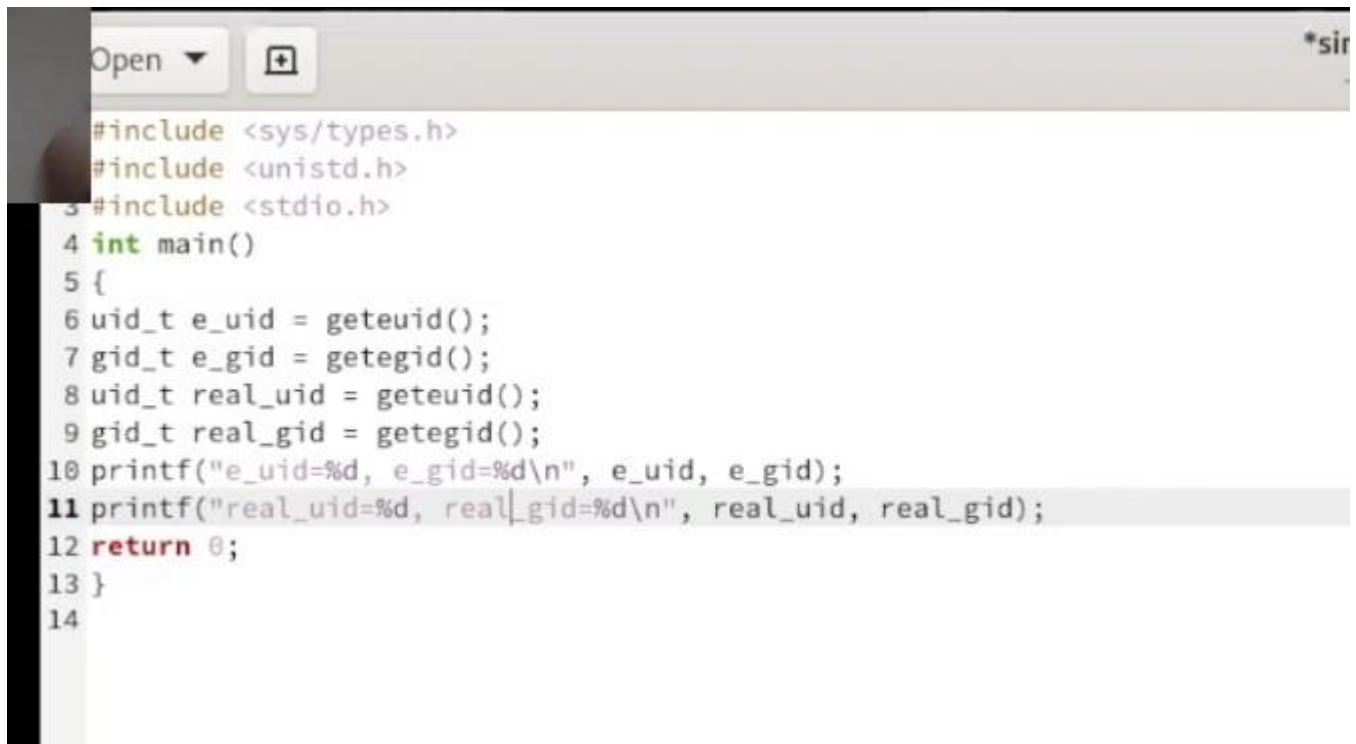
3. Скомпилировали программу и убедились, что файл программы создан: gcc simpleid.c
-o simpleid
4. Выполнили программу simpleid командой ./simpleid
5. Выполнили системную программу id с помощью команды id. uid и gid совпадает в обеих программах



```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4 int main()
5 {
6     uid_t uid = geteuid();
7     gid_t gid = getegid();
8     printf("uid=%d, gid=%d\n", uid, gid);
9     return 0;
10 }
11
```

#fig

6. Усложнили программу, добавив вывод действительных идентификаторов.



```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
int main()
{
    uid_t e_uid = geteuid();
    gid_t e_gid = getegid();
    uid_t real_uid = geteuid();
    gid_t real_gid = getegid();
    printf("e_uid=%d, e_gid=%d\n", e_uid, e_gid);
    11 printf("real_uid=%d, real_gid=%d\n", real_uid, real_gid);
    return 0;
}
```

#fig

7. Скомпилировали и запустили simpleid2.c:

```
gcc simpleid2.c -o simpleid2
./simpleid2
```

8. От имени суперпользователя выполнили команды:

```
chown root:guest /home/guest/simpleid2
chmod u+s /home/guest/simpleid2
```

9. Использовали su для повышения прав до суперпользователя
10. Выполнили проверку правильности установки новых атрибутов и смены владельца файла simpleid2:

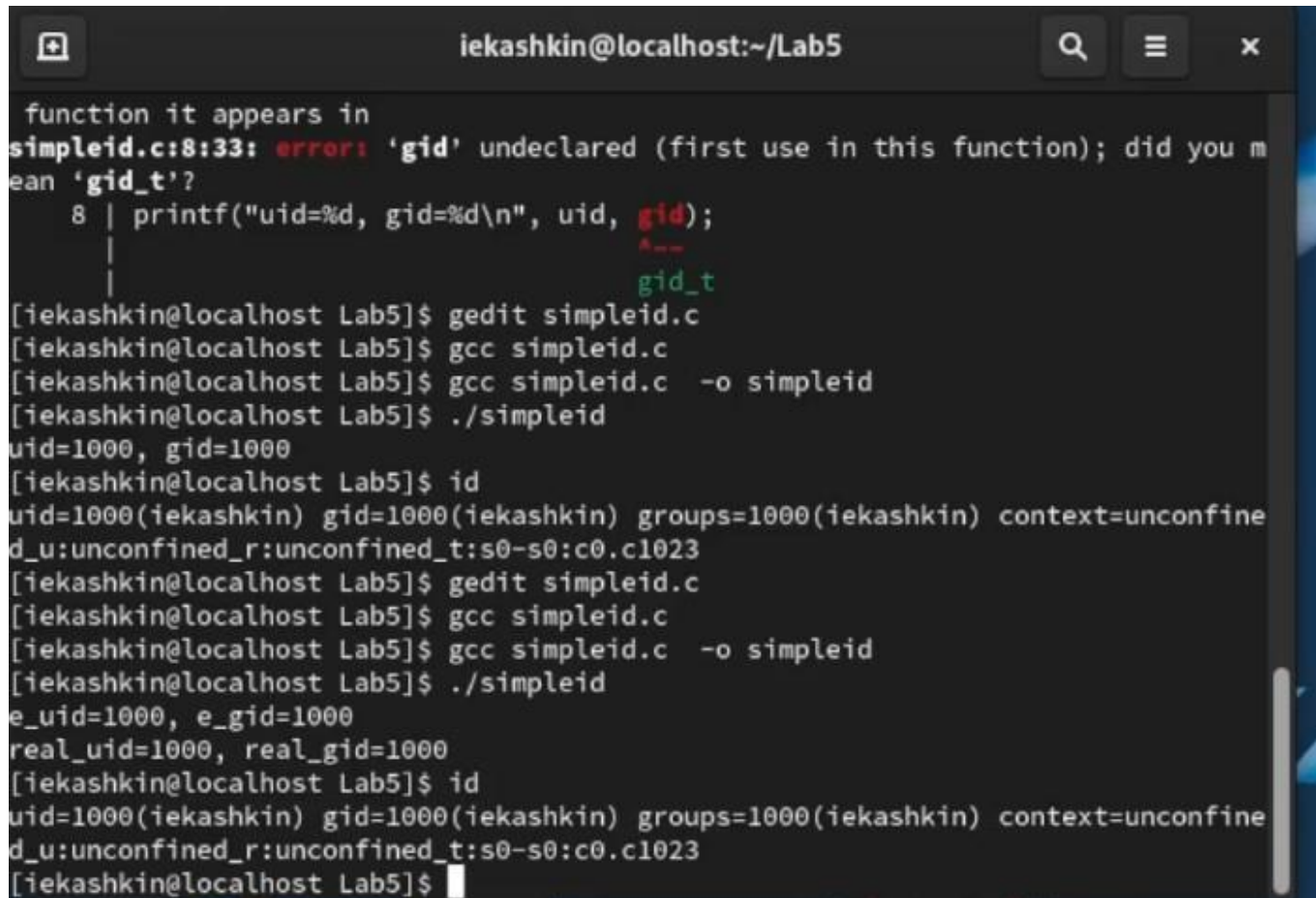
```
ls -l simpleid2
```

11. Запустили simpleid2 и id:

```
./simpleid2  
id
```

Результат выполнения программ теперь немного отличается


12. Проделали тоже самое относительно SetGID-бита.



```
function it appears in
simpleid.c:8:33: error: 'gid' undeclared (first use in this function); did you mean 'gid_t'?
      8 | printf("uid=%d, gid=%d\n", uid, gid);
        |                                ^
        |                                gid_t
[iekashkin@localhost Lab5]$ gedit simpleid.c
[iekashkin@localhost Lab5]$ gcc simpleid.c
[iekashkin@localhost Lab5]$ gcc simpleid.c -o simpleid
[iekashkin@localhost Lab5]$ ./simpleid
uid=1000, gid=1000
[iekashkin@localhost Lab5]$ id
uid=1000(iekashkin) gid=1000(iekashkin) groups=1000(iekashkin) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[iekashkin@localhost Lab5]$ gedit simpleid.c
[iekashkin@localhost Lab5]$ gcc simpleid.c
[iekashkin@localhost Lab5]$ gcc simpleid.c -o simpleid
[iekashkin@localhost Lab5]$ ./simpleid
e_uid=1000, e_gid=1000
real_uid=1000, real_gid=1000
[iekashkin@localhost Lab5]$ id
uid=1000(iekashkin) gid=1000(iekashkin) groups=1000(iekashkin) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[iekashkin@localhost Lab5]$
```

#fig

13. Написали программу readfile.c

```
Open ▾ 
#include <stdio.h>
#include <sys/stat.h>
3 #include <sys/stat.h>
4 #include <unistd.h>
5 #include <fcntl.h>
6
7 int main(int argc, char* argv[])
8 {
9     unsigned char buffer[16];
10    size_t bytes_read;
11    int i;
12
13    int fd=open(argv[1], O_RDONLY);
14    do
15    {
16    bytes_read=read((fd, buffer, sizeof(buffer));
17    for (i=0; i<bytes_read; ++i)
18    printf("%c", buffer[i]);
19    }
20    while (bytes_read == (buffer));
21    close (fd);
22    return 0;
23 }
```

#fig

14. Откомпилировали её.

```
gcc readfile.c -o readfile
```

15. Сменили владельца у файла readfile.c и изменили права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог.

```
chown root:guest /home/guest/readfile.c
chmod 700 /home/guest/readfile.c
```

16. Проверили, что пользователь guest не может прочитать файл readfile.c.

17. Сменили у программы readfile владельца и установили SetU'D-бит.

18. Проверили, может ли программа readfile прочитать файл readfile.c
19. Проверили, может ли программа readfile прочитать файл /etc/shadow


```
iekashkin@localhost:~/Lab5
|
[iekashkin@localhost Lab5]$ su
Password:
[root@localhost Lab5]# chown root:root readfile
chown: cannot access 'readfile': No such file or directory
[root@localhost Lab5]# xit
bash: xit: command not found...
^[[A^[[D[root@localhost Lab5]# exit
exit
[iekashkin@localhost Lab5]$ gcc readfile.c -o readfile
readfile.c: In function 'main':
readfile.c:20:19: warning: comparison between pointer and integer
 20 | while (bytes_read == (buffer));
    |                   ^~
[iekashkin@localhost Lab5]$ su
Password:
[root@localhost Lab5]# chown root:root readfile
[root@localhost Lab5]# chmod -rwx readfile.c
[root@localhost Lab5]# chmod u+s readfile
[root@localhost Lab5]# exit
exit
[iekashkin@localhost Lab5]$ cat readfile.c
cat: readfile.c: Permission denied
[iekashkin@localhost Lab5]$ cat readfile
```

#fig

Исследование Sticky-бита

1. Выяснили, установлен ли атрибут Sticky на директории /tmp:

```
ls -l / | grep tmp
```

2. От имени пользователя guest создали файл file01.txt в директории /tmp со словом test:

```
echo "test" > /tmp/file01.txt
```

3. Просмотрели атрибуты у только что созданного файла и разрешили чтение и запись для категории пользователей «все остальные»:

```
ls -l /tmp/file01.txt  
chmod o+rw /tmp/file01.txt
```

```
ls -l /tmp/file01.txt
```

Первоначально все группы имели право на чтение, а запись могли осуществлять все, кроме «остальных пользователей».

4. От пользователя (не являющегося владельцем) попробовали прочитать файл /file01.txt:

```
cat /file01.txt
```

5. От пользователя попробовали дозаписать в файл /file01.txt слово test3 командой:

```
echo "test2" >> /file01.txt
```

6. Проверили содержимое файла командой:

```
cat /file01.txt
```

В файле теперь записано:

```
Test  
Test2
```

7. От пользователя попробовали записать в файл /tmp/file01.txt слово test4, стерев при этом всю имеющуюся в файле информацию командой. Для этого воспользовалась командой `echo "test3" > /tmp/file01.txt`
8. Проверили содержимое файла командой

```
cat /tmp/file01.txt
```

9. От пользователя попробовали удалить файл /tmp/file01.txt командой `rm /tmp/file01.txt`, однако получила отказ.
10. От суперпользователя командой выполнили команду, снимающую атрибут `t` (Sticky- бит) с директории /tmp:

```
chmod -t /tmp
```

Покинули режим суперпользователя командой `exit`.

11. От пользователя проверили, что атрибута `t` у директории `/tmp` нет:

```
ls -l / | grep tmp
```

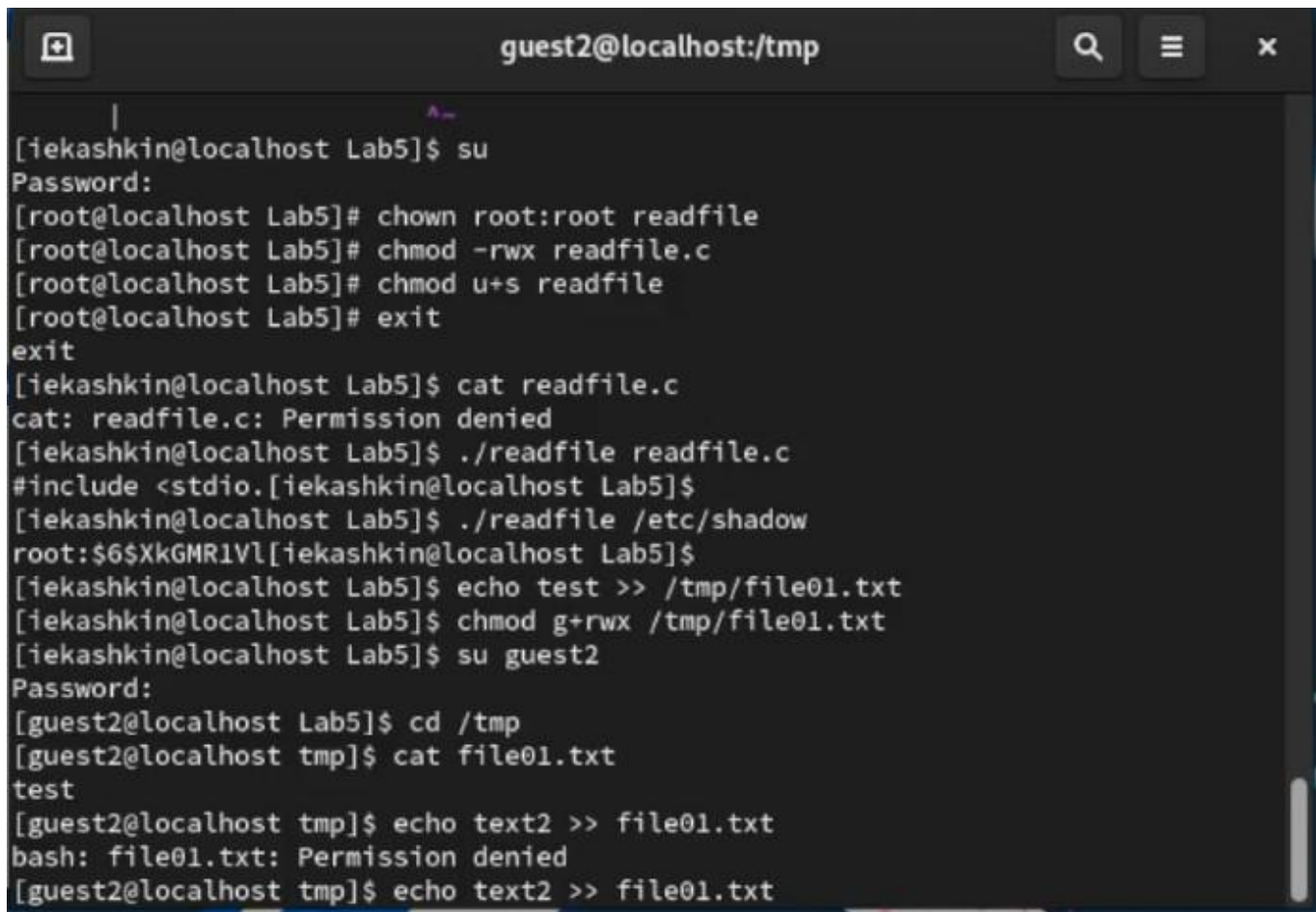
12. Повторили предыдущие шаги. Получилось удалить файл

13. Удалось удалить файл от имени пользователя, не являющегося его владельцем.

14. Повысили свои права до суперпользователя и вернули атрибут `t` на директорию `/tmp`

:

```
su
chmod +t /tmp
exit
```



```
guest2@localhost:/tmp
|
[iekashkin@localhost Lab5]$ su
Password:
[root@localhost Lab5]# chown root:root readfile
[root@localhost Lab5]# chmod -rwx readfile.c
[root@localhost Lab5]# chmod u+s readfile
[root@localhost Lab5]# exit
exit
[iekashkin@localhost Lab5]$ cat readfile.c
cat: readfile.c: Permission denied
[iekashkin@localhost Lab5]$ ./readfile readfile.c
#include <stdio.h>
[iekashkin@localhost Lab5]$ ./readfile /etc/shadow
root:$6$XkGMR1Vl
[iekashkin@localhost Lab5]$ echo test >> /tmp/file01.txt
[iekashkin@localhost Lab5]$ chmod g+rw /tmp/file01.txt
[iekashkin@localhost Lab5]$ su guest2
Password:
[guest2@localhost Lab5]$ cd /tmp
[guest2@localhost tmp]$ cat file01.txt
test
[guest2@localhost tmp]$ echo text2 >> file01.txt
bash: file01.txt: Permission denied
[guest2@localhost tmp]$ echo text2 >> file01.txt
```

#fig

Выводы

Изучили механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получили практические навыки работы в консоли с дополнительными атрибутами. Также

мы рассмотрели работу механизма смены идентификатора процессов пользователей и влияние бита Sticky на запись и удаление файлов.

Список литературы{.unnumbered}

1. [КОМАНДА CHATTR В LINUX](#)
2. [chattr](#)