

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №5

дисциплина: Основы информационной безопасности

Студент: Исаев Булат Абубакарович

Студ. Билет: 1132227131

Группа: НПИбд-01-22

МОСКВА

2024 г.

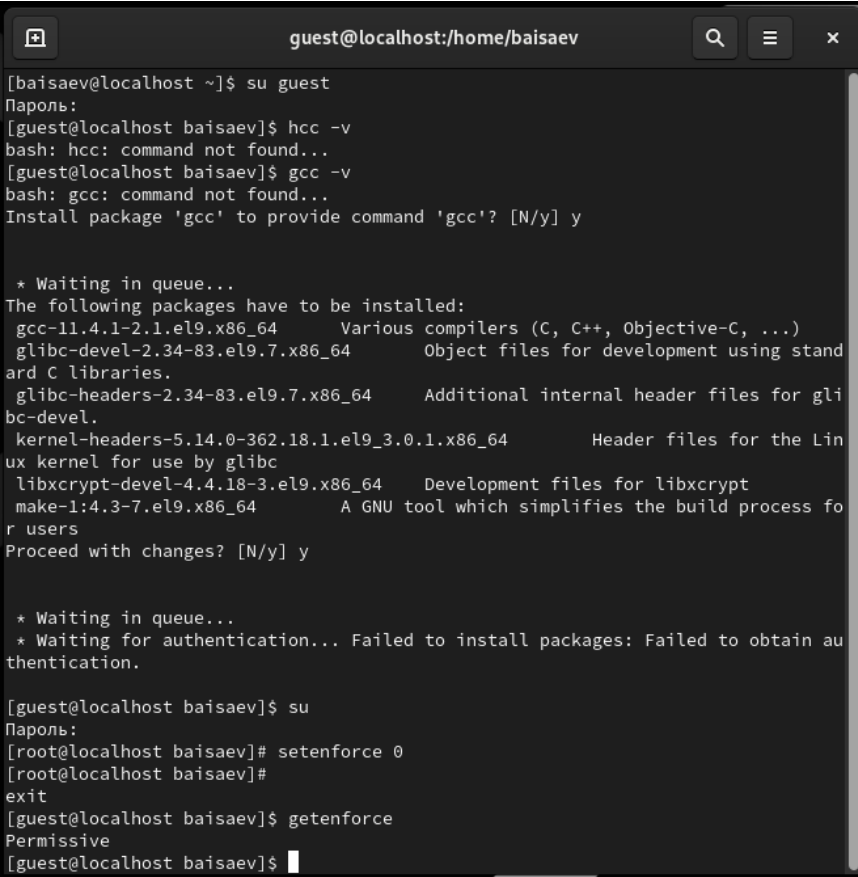
Цель работы:

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.

Выполнение работы:

1. Для выполнения части заданий требуются средства разработки приложений.
Проверяем наличие установленного компилятора gcc (gcc -v)
2. Чтобы система защиты SELinux не мешала выполнению заданий работы, отключили систему запретов до очередной перезагрузки системы. (setenforce 0)
3. Команда getenforce вывела Permissive (getenforce)

Вышеописанные команды показаны в (рис. 1)



```
guest@localhost:/home/baisaev
[baisaev@localhost ~]$ su guest
Пароль:
[guest@localhost baisaev]$ hcc -v
bash: hcc: command not found...
[guest@localhost baisaev]$ gcc -v
bash: gcc: command not found...
Install package 'gcc' to provide command 'gcc'? [N/y] y

* Waiting in queue...
The following packages have to be installed:
gcc-11.4.1-2.1.el9.x86_64      Various compilers (C, C++, Objective-C, ...)
glibc-devel-2.34-83.el9.7.x86_64  Object files for development using standard C libraries.
glibc-headers-2.34-83.el9.7.x86_64  Additional internal header files for glibc-devel.
kernel-headers-5.14.0-362.18.1.el9_3.0.1.x86_64  Header files for the Linux kernel for use by glibc
libxcrypt-devel-4.4.18-3.el9.x86_64  Development files for libxcrypt
make-1:4.3-7.el9.x86_64      A GNU tool which simplifies the build process for users
Proceed with changes? [N/y] y

* Waiting in queue...
* Waiting for authentication... Failed to install packages: Failed to obtain authentication.

[guest@localhost baisaev]$ su
Пароль:
[root@localhost baisaev]# setenforce 0
[root@localhost baisaev]#
exit
[guest@localhost baisaev]$ getenforce
Permissive
[guest@localhost baisaev]$
```

Рис. 1 – Подготовка к работе

2.2 Изучение механики SetUID

1. Входим в систему от имени пользователя guest. (su guest)

2. Пишем программу simpleid.c.

(touch simpleid.c)

(gedit simpleid.c)

Вышеописанные команды показаны в (рис. 2)

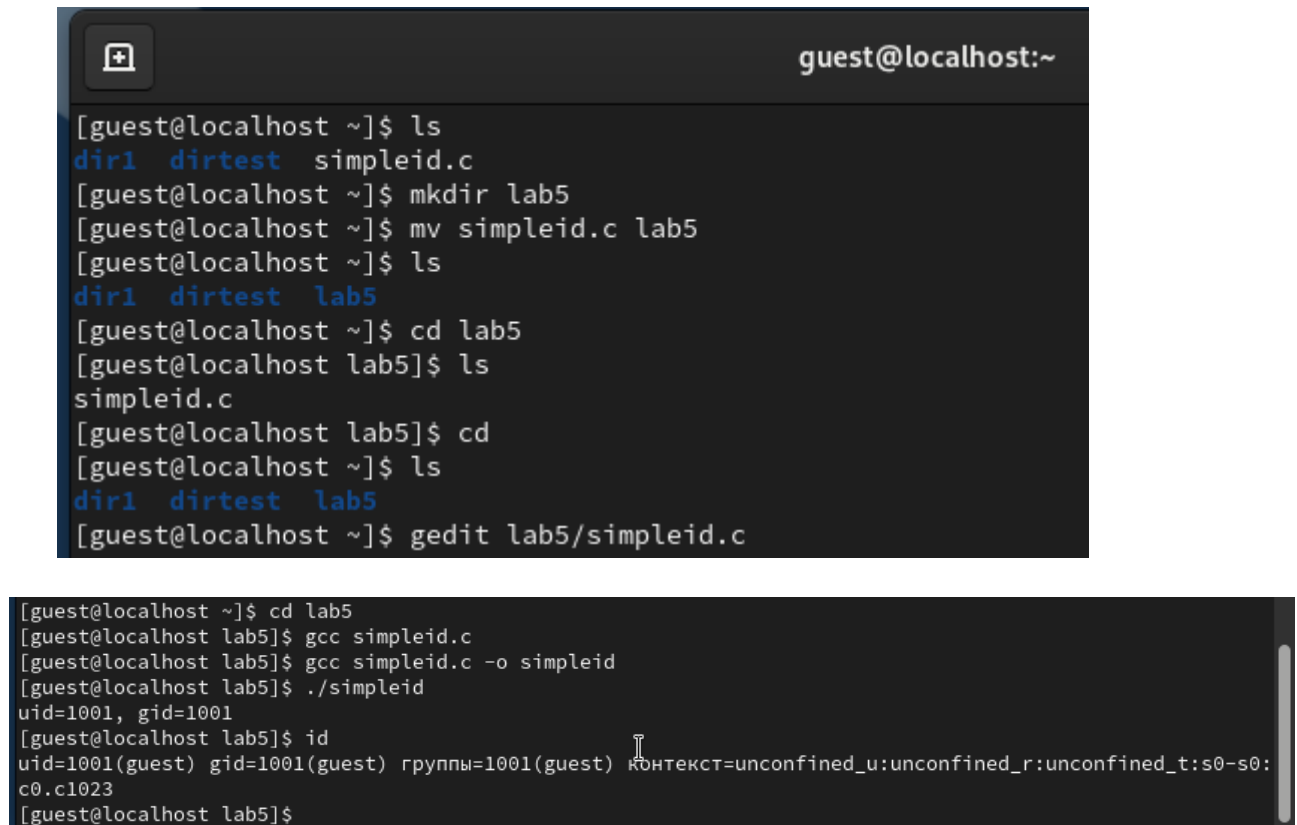


```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4 int main()
5 {
6 uid_t uid = geteuid();
7 gid_t gid = geteuid();
8 printf("uid=%d, gid=%d\\n", uid, gid);
9 return 0;
10 }
```

Рис. 2 – Программа simpleid

3. Скомпилируем программу (gcc simpleid.c -o simpleid)
 4. Выполняем программу simpleid (./simpleid)
 5. Выполняем системную программу id (id).
- uid и gid совпадает в обеих программах

Вышеописанные команды показаны в (рис. 3)



```
guest@localhost:~  
[guest@localhost ~]$ ls  
dir1  dirtest  simpleid.c  
[guest@localhost ~]$ mkdir lab5  
[guest@localhost ~]$ mv simpleid.c lab5  
[guest@localhost ~]$ ls  
dir1  dirtest  lab5  
[guest@localhost ~]$ cd lab5  
[guest@localhost lab5]$ ls  
simpleid.c  
[guest@localhost lab5]$ cd  
[guest@localhost ~]$ ls  
dir1  dirtest  lab5  
[guest@localhost ~]$ gedit lab5/simpleid.c  
[guest@localhost ~]$ cd lab5  
[guest@localhost lab5]$ gcc simpleid.c  
[guest@localhost lab5]$ gcc simpleid.c -o simpleid  
[guest@localhost lab5]$ ./simpleid  
uid=1001, gid=1001  
[guest@localhost lab5]$ id  
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023  
[guest@localhost lab5]$
```

Рис. 3 – Результат программы simpleid

6. Усложняем программу, добавив вывод действительных идентификаторов.



```
1 #include <fcntl.h>
2 #include <stdio.h>
3 #include <sys/stat.h>
4 #include <sys/types.h>
5 #include <unistd.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) printf("%c", buffer[i]);
18    }
19
20    while(bytes_read == sizeof(buffer));
21    close(fd);
22    return 0;
23 }
```

Рис. 4 – Программа simpleid2

7. Скомпилируем и запустили simpleid2.c

(gcc simpleid2.c -o simpleid2)

(./simpleid2)

8. От имени суперпользователя выполняем команды:

(chown root:guest /home/guest/simpleid2)

(chmod u+s /home/guest/simpleid2)

9. Повышаем права до суперпользователя (su)

10. Выполняем проверку правильности установки новых атрибутов и смены владельца файла simpleid2 (ls -l simpleid2)

11. Запускаем simpleid2 и id (./simpleid2 id)

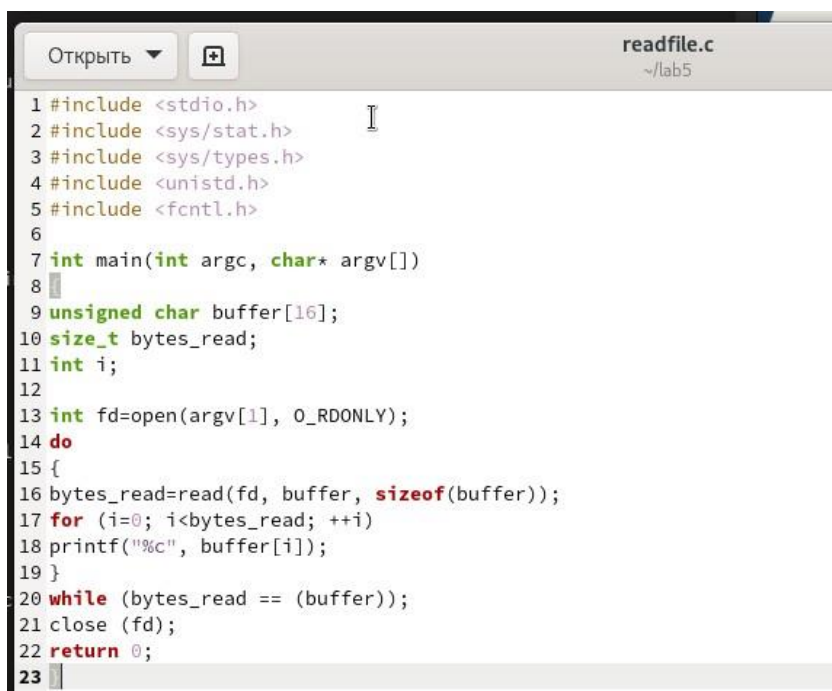
Результат выполнения программ теперь немного отличается

12. Проделаем тоже самое относительно SetGID-бита.

```
[guest@localhost lab5]$  
[guest@localhost lab5]$ touch simpleid2.c  
[guest@localhost lab5]$ gedit simpleid2.c  
[guest@localhost lab5]$ gcc simpleid2.c  
[guest@localhost lab5]$ gcc simpleid2.c -o simpleid2  
[guest@localhost lab5]$ ./simpleid2  
[guest@localhost lab5]  
[guest@localhost lab5]=1001  
[guest@localhost lab5]$ su  
Пароль:  
[root@localhost lab5]#  
[root@localhost lab5]# chown root:guest simpleid2  
[root@localhost lab5]# chmod u+s simpleid2  
[root@localhost lab5]# ./simpleid2  
e_uid=0, e_gid=0  
real_uid=0, real_gid=0  
[root@localhost lab5]# id  
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfine  
[root@localhost lab5]# chmod g+s simpleid2  
[root@localhost lab5]# ./simpleid2  
e_uid=0, e_gid=1001  
real_uid=0, real_gid=0  
[root@localhost lab5]#  
exit  
[guest@localhost lab5]$ ./simpleid2  
e_uid=0, e_gid=1001  
real_uid=1001, real_gid=1001  
[guest@localhost lab5]$
```

Рис. 5 – Результат программы simpleid2

13. Напишем программу readfile.c



```
1 #include <stdio.h>
2 #include <sys/stat.h>
3 #include <sys/types.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 }
```

Рис. 6 – Программа readfile

14. Откомпилируем её. (gcc readfile.c -o readfile)

15. Сменим владельца у файла readfile.c и изменим 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. Проверим, что пользователь guest не может прочитать файл readfile.c.

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

18. Проверим, может ли программа readfile прочитать файл readfile.c

19. Проверим, может ли программа readfile прочитать файл /etc/shadow

```
[guest@localhost lab5]$ touch readfile.c
[guest@localhost lab5]$ gedit readfile.c
[guest@localhost lab5]$ gcc readfile.c
readfile.c: В функции «main»:
readfile.c:20:19: предупреждение: сравнение указателя и целого
   20 | while (bytes_read == (buffer));
       |                   ^~
[guest@localhost lab5]$ gcc readfile.c -o readfile
readfile.c: В функции «main»:
readfile.c:20:19: предупреждение: сравнение указателя и целого
   20 | while (bytes_read == (buffer));
       |                   ^~
[guest@localhost lab5]$ su
Пароль:
[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
[guest@localhost lab5]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@localhost lab5]$ ./readfile readfile.c
#include <stdio.[guest@aburlakova lab5]$
[guest@localhost lab5]$ ./readfile /etc/shadow
root:$6$0mJpkglj[guest@localhost lab5]$
```

Рис.7 - Результат программы readfile

2.3 Исследование Sticky-бита

1. Выясним, установлен ли атрибут Sticky на директории /tmp: (ls -l / | grep tmp)
2. От имени пользователя guest создаём файл file01.txt в директории /tmp со словом test:

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

3. Просмотрим 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) попробуем прочитать файл /file01.txt:

```
(cat /file01.txt)
```

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

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

6. Проверим содержимое файла командой: (cat /file01.txt)

В файле теперь записано:

```
Test
```

```
Test2
```

7. От пользователя попробуем 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 попробуем удалить файл /tmp/file01.txt командой rm /tmp/file01.txt, однако получила отказ.

10. От суперпользователя командой выполним команду, снимающую атрибут t (Sticky-бит) с директории /tmp:

```
chmod -t /tmp
```

Покинули режим суперпользователя командой exit.

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

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

12. Повторим предыдущие шаги. Получилось удалить файл

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

14. Повысим свои права до суперпользователя и вернули атрибут t на директорию /tmp :

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

```
[guest@aburlakova lab5]$  
[guest@aburlakova lab5]$ cd /tmp  
[guest@aburlakova tmp]$ echo test >> file01.txt  
[guest@aburlakova tmp]$ chmog g+rwX file01.txt  
bash: chmog: command not found...  
[guest@aburlakova tmp]$ chmod g+rwX file01.txt  
[guest@aburlakova tmp]$ su guest2  
Пароль:  
[guest2@aburlakova tmp]$ echo test >> file01.txt  
[guest2@aburlakova tmp]$ cat file01.txt  
test  
test  
[guest2@aburlakova tmp]$ echo 1123 > file01.txt  
[guest2@aburlakova tmp]$ rm file01.txt  
rm: невозможно удалить 'file01.txt': Операция не позволена  
[guest2@aburlakova tmp]$ su  
Пароль:  
[root@aburlakova tmp]# chmod -t /tmp  
[root@aburlakova tmp]#  
exit  
[guest2@aburlakova tmp]$ rm file01.txt  
[guest2@aburlakova tmp]$
```

Figure 2.8: исследование Sticky-бита

Выводы

Изучили механизмы изменения идентификаторов, применения SetUID- и Stickyбитов.

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

1. КОМАНДА CHATTR В LINUX
2. chattr