

Лабораторная работа №5

Павлова Варвара Юрьевна

Содержание

Цель работы	1
Ход работы.....	1
Вывод.....	8

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.

Ход работы

- 1. Войдите в систему от имени пользователя guest.
- 2. Создайте программу simpleid.c:: (рис. [-@fig:001]) (рис. [-@fig:002])

```
[guest@localhost ~]$ touch simpleid.c
```

touch

```

[guest@localhost ~]$ nano simpleid.c
[guest@localhost ~]$ cat simpleid.c
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int
main ()
{
    uid_t uid = getuid ();
    gid_t gid = getgid ();
    printf ("uid=%d, gid=%d\n", uid, gid);
    return 0;
}
[guest@localhost ~]$

```

simpleid.c

3. Скомпилируйте программу и убедитесь, что файл программы создан: (рис. [-@fig:003])

```

[guest@localhost ~]$ gcc simpleid.c -o simpleid
[guest@localhost ~]$ ls
dir1      simpleid.c  Документы  Изображения  Общедоступные  Шаблоны
simpleid  Видео      Загрузки  Музыка      'Рабочий стол'

```

компиляция

4. Выполните программу simpleid: (рис. [-@fig:004])

```

[guest@localhost ~]$ ./simpleid
uid=1001, gid=1001
[guest@localhost ~]$

```

выполнение

5. Выполните системную программу id: (рис. [-@fig:005])

```

[guest@localhost ~]$ id
uid=1001, gid=1001
[guest@localhost ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023

```

id

6. Усложните программу, добавив вывод действительных идентификаторов: (рис. [-@fig:006])

```

[guest@localhost ~]$ nano simpleid.c
[guest@localhost ~]$ cat simpleid.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;
}

```

simpleid2

7. Скомпилируйте и запустите simpleid2.c: (рис. [-@fig:007])

```

[guest@localhost ~]$ gcc simpleid2.c -o simpleid2
[guest@localhost ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001

```

компиляция

8. От имени суперпользователя выполните команды: (рис. [-@fig:008])

```

[guest@localhost ~]$ su -
Пароль:
[root@localhost ~]# chown root:guest /home/guest/simpleid2
[root@localhost ~]# chmod u+s /home/guest/simpleid2
[root@localhost ~]#

```

команды

9. Выполните проверку правильности установки новых атрибутов и смены владельца файла simpleid2: (рис. [-@fig:009])

```

[guest@localhost ~]$ ls -l simpleid2
-rwsr-xr-x. 1 root guest 24488 окт  5 13:43 simpleid2

```

ls -l

10. Запустите simpleid2 и id: (рис. [-@fig:010])

```
-rwsr-xr-x. 1 root guest 24488 0K 5 13:43 simpleid2
[guest@localhost ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@localhost ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfine
ned_r:unconfined_t:s0-s0:c0.c1023
[guest@localhost ~]$
```

запуск

11. Создайте программу readfile.c:(рис. [-@fig:011])

```
[guest@localhost ~]$ cat 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;
}
```

readfile.c

12. Откомпилируйте её. (рис. [-@fig:012])

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

компиляция

13. Смените владельца у файла readfile.c (или любого другого текстового файла в системе) и измените права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог. (рис. [-@fig:013])


```
Ошибка сегментирования (стек памяти сброшен на диск)
[guest@localhost ~]$ ls -l / | grep tmp
drwxrwxrwt. 16 root root 4096 окт  5 14:02 tmp
```

ls

19. От имени пользователя `guest` создайте файл `file01.txt` в директории `/tmp` со словом `test`: (рис. [-@fig:019])

```
[guest@localhost tmp]$ echo "test" > file01.txt
[guest@localhost tmp]$ ls
dbus-KCbEl2r705
dbus-M4C6AWjmqQ
file01.txt
```

echo

20. Просмотрите атрибуты у только что созданного файла и разрешите чтение и запись для категории пользователей «все остальные»: (рис. [-@fig:020])

```
[guest@localhost tmp]$ ls -l file01.txt
-rw-r--r--. 1 guest guest 5 окт  5 14:04 file01.txt
[guest@localhost tmp]$ chmod o+rw file01.txt
[guest@localhost tmp]$ ls -l file01.txt
-rw-r--rw-. 1 guest guest 5 окт  5 14:04 file01.txt
```

chmod

21. От пользователя `guest2` (не являющегося владельцем) попробуйте прочитать файл `/tmp/file01.txt`: (рис. [-@fig:021])

```
systemd-private-ecaf1712840b408eb441c2e
[guest2@localhost tmp]$ cat file01.txt
test
```

cat

22. От пользователя `guest2` попробуйте дозаписать в файл `/tmp/file01.txt` слово `test2` (рис. [-@fig:022])

```
test
[guest2@localhost tmp]$ echo "test2" > file01.txt
```

echo

23. Проверьте содержимое файла командой (рис. [-@fig:023])

```
[guest2@localhost tmp]$ echo "test2" >
[guest2@localhost tmp]$ cat file01.txt
test2
```

cat

24. От пользователя guest2 попробуйте записать в файл /tmp/file01.txt слово test3(рис. [-@fig:024])

```
test3  
[guest2@localhost tmp]$ echo "test3" > file01.txt
```

chmod

25. Проверьте содержимое файла командой(рис. [-@fig:025])

```
[guest2@localhost tmp]$ cat file01.txt  
test3
```

cat

26. От пользователя guest2 попробуйте удалить файл /tmp/file01.txt командой (рис. [-@fig:026])

```
test3  
[guest2@localhost tmp]$ rm file01.txt  
rm: невозможно удалить 'file01.txt': Операция не позволена  
[guest2@localhost tmp]$
```

rm

27. Повысьте свои права до суперпользователя и выполните после этого команду, снимающую атрибут t (Sticky-бит) с директории /tmp:(рис. [-@fig:027])

```
[guest@localhost tmp]$ su -  
Пароль:  
[root@localhost ~]# chmod -t /tmp  
[guest@localhost tmp]$
```

chmod

28. От пользователя guest2 проверьте, что атрибута t у директории /tmp нет: (рис. [-@fig:028])

```
rm: невозможно удалить 'file01.txt': Операция не позволена  
[guest2@localhost tmp]$ ls -l / | grep tmp  
drwxrwxrwx. 16 root root 4096 окт  5 14:11 tmp
```

ls

29. Удалось ли вам удалить файл от имени пользователя, не являющегося его владельцем?(рис. [-@fig:029])

```
drwxrwxrwx. 16 root root 4096 окт  5 14:11 tmp  
[guest2@localhost tmp]$ rm file01.txt  
[guest2@localhost tmp]$
```

rm

30. Повысьте свои права до суперпользователя и верните атрибут `t` на директорию `/tmp`: (рис. [-@fig:030])

```
bash: 1234: команда не найдена...
[guest2@localhost tmp]$ su -
Пароль:
[root@localhost ~]# chmod +t /tmp
[root@localhost ~]# exit
выход
[guest2@localhost tmp]$
```

chmod

Вывод

В ходе выполнения данной лабораторной работы я изучила механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получила практические навыки работы в консоли с дополнительными атрибутами. Рассмотрела работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.