

Отчёт по лабораторной работе №5

**Дискреционное разграничение прав в Linux. Исследование влияния
дополнительных атрибутов**

Савченко Елизавета НБИ-01-20

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
2.1	Подготовка	5
2.2	Изучение механики SetUID	6
2.3	Исследование Sticky-бита	11
3	Выводы	14
	Список литературы	15

List of Figures

2.1	установка компилятора	5
2.2	подготовка к работе	6
2.3	проверка установки компилятора	6
2.4	создание и редактирование файла simpleid.c	6
2.5	написание программы	7
2.6	результат программы simpleid	7
2.7	Создание и редактирование файла simpleid2.c	7
2.8	написание программы	8
2.9	результат программы simpleid2 от guest	8
2.10	результат программы simpleid2 от root	9
2.11	создание и редактирование файла readfile.c	9
2.12	программа readfile	10
2.13	смена прав на файле readfile	10
2.14	проверка файла на чтение	11
2.15	проверка прав на файле file01.txt	12
2.16	дозапись и перезапись слов в файл file01.txt	12
2.17	снятие атрибута	13

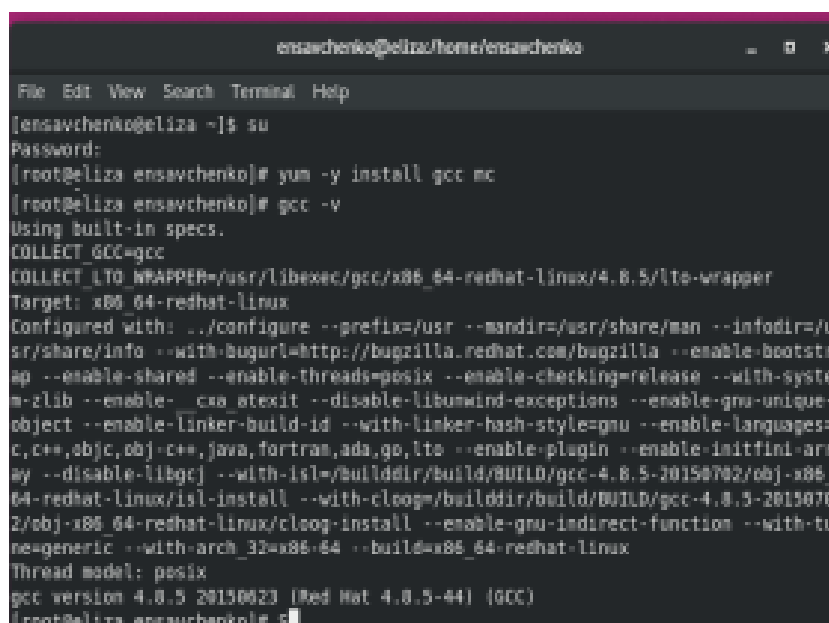
1 Цель работы

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 Выполнение лабораторной работы

2.1 Подготовка

1. Для выполнения части заданий требуются средства разработки приложений. Проверили наличие установленного компилятора gcc командой `gcc -v`: компилятор обнаружен.



```
ensavchenko@eliza:~/home/ensavchenko
File Edit View Search Terminal Help
[ensavchenko@eliza ~]$ su
Password:
[root@eliza ensavchenko]# yum -y install gcc nc
[root@eliza ensavchenko]# gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/x86_64-redhat-linux/4.8.5/lto-wrapper
Target: x86_64-redhat-linux
Configured with: ../configure --prefix=/usr --mandir=/usr/share/man --infodir=/u
sr/share/info --with-bugurl=http://bugzilla.redhat.com/bugzilla --enable-bootsr
ap --enable-shared --enable-threads=posix --enable-checking=release --with-syste
m-zlib --enable-__cxa_atexit --disable-libunwind-exceptions --enable-gnu-unique-
object --enable-linker-build-id --with-linker-hash-style=gnu --enable-languages=
c,c++,objc,obj-c++,java,fortran,ada,go,lto --enable-plugin --enable-initfini-arr
ey --disable-libgcj --with-isl=/build/builddir/build/BUILD/gcc-4.8.5-20150702/obj-x86_
64-redhat-linux/isl-install --with-cloog=/build/builddir/build/BUILD/gcc-4.8.5-2015070
2/obj-x86_64-redhat-linux/cloog-install --enable-gnu-indirect-function --with-tu
newgeneric --with-arch_32=x86_64 --build=x86_64-redhat-linux
Thread model: posix
gcc version 4.8.5 20150623 (Red Hat 4.8.5-44) (GCC)
[root@eliza ensavchenko]#
```

Figure 2.1: установка компилятора

2. Чтобы система защиты SELinux не мешала выполнению заданий работы, отключили систему запретов до очередной перезагрузки системы командой `setenforce 0`:

3. Команда `getenforce` вывела `Permissive`:

```
gcc version 4.8.5 20150623 (Red Hat 4.8.5-44) (GCC)
[root@eliza ensavchenko]# setenforce 0
[root@eliza ensavchenko]# getenforce
Permissive
```

Figure 2.2: подготовка к работе

4. Теперь проверим корректность установки компилятора, как показано на скриншоте номер 3.

```
[root@eliza ensavchenko]# gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/x86_64-redhat-linux/4.8.5/lto-wrapper
Target: x86_64-redhat-linux
Configured with: ../configure --prefix=/usr --mandir=/usr/share/man --infodir=/u
sr/share/info --with-bugurl=http://bugzilla.redhat.com/bugzilla --enable-bootsr
ap --enable-shared --enable-threads=posix --enable-checking=release --with-syste
m-zlib --enable-_cxa_atexit --disable-libunwind-exceptions --enable-gnu-unique-
object --enable-linker-build-id --with-linker-hash-style=gnu --enable-languages=
c,c++,objc,obj-c++,java,fortran,ada,go,lto --enable-plugin --enable-initfini-arr
ay --disable-libgcj --with-isl=/builddir/build/BUILD/gcc-4.8.5-20150702/obj-x86_
64-redhat-linux/isl-install --with-cloog=/builddir/build/BUILD/gcc-4.8.5-2015070
2/obj-x86_64-redhat-linux/cloog-install --enable-gnu-indirect-function --with-tu
ne=generic --with-arch_32=x86_64 --build=x86_64-redhat-linux
Thread model: posix
gcc version 4.8.5 20150623 (Red Hat 4.8.5-44) (GCC)
[root@eliza ensavchenko]# getenforce
Permissive
```

Figure 2.3: проверка установки компилятора

2.2 Изучение механики SetUID

1. Вошли в систему от имени пользователя `guest`.

```
[guest@eliza ~]$ mkdir laba5
mkdir: cannot create directory 'laba5': File exists
[guest@eliza ~]$ touch simpleid.c
[guest@eliza ~]$ cat simpleid.c
```

Figure 2.4: создание и редактирование файла `simpleid.c`

2. Написали программу `simpleid.c`.

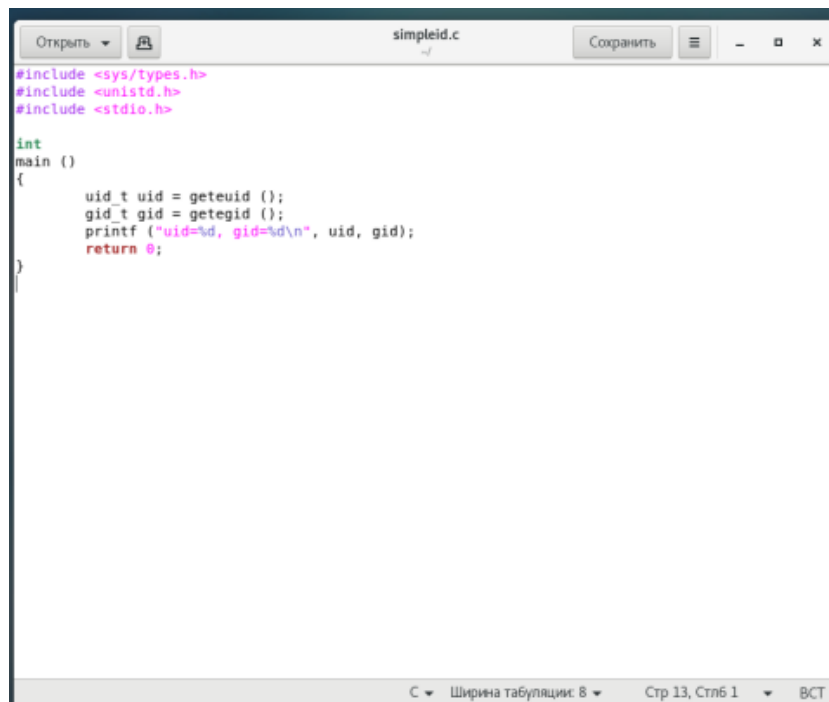


Figure 2.5: написание программы

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

```

[guest@eliza ~]$ gcc simpleid.c -o simpleid
[guest@eliza ~]$ ./simpleid
uid=1001, gid=1001
[guest@eliza ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023

```

Figure 2.6: результат программы simpleid

6. Усложнили программу, добавив вывод действительных идентификаторов.

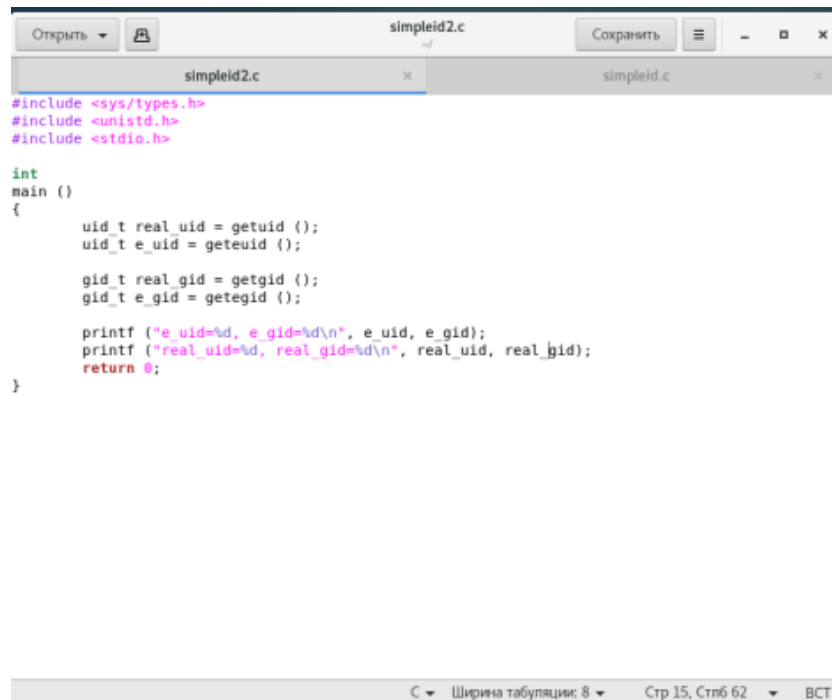
```

[guest@eliza ~]$ touch simpleid2.c
[guest@eliza ~]$ gedit simpleid2.c

```

Figure 2.7: Создание и редактирование файла simpleid2.c

7. Написали следующую программу в файле simpleid2.c



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

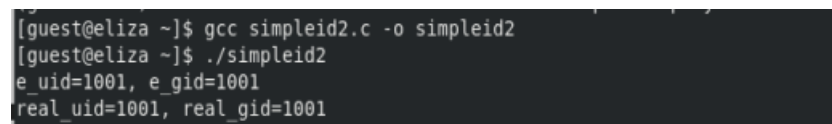
int
main ()
{
    uid_t real_uid = getuid ();
    uid_t e_uid = geteuid ();

    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;
}
```

Figure 2.8: написание программы

8. Скомпилировали и запустили simpleid2.c: gcc



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

Figure 2.9: результат программы simpleid2 от guest

9. На скриншоте 10 видно следующее: теперь от имени суперпользователя выполнили команды, которые меняют владельца файла. Использовали su для повышения прав до суперпользователя. Выполнили проверку правильности установки новых атрибутов и смены владельца файла simpleid2. Также запустили simpleid2 и id.


```
[guest@eliza ~]$ su
Password:
[root@eliza guest]# chown root:guest /home/guest/simpleid2
[root@eliza guest]# chmod u+s /home/guest/simpleid2
[root@eliza guest]# ls -l simpleid2
-rwsrwxr-x. 1 root guest 8616 Oct  7 22:39 simpleid2
[root@eliza guest]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@eliza guest]#
```

Figure 2.10: результат программы simpleid2 от root

Приходим к выводу, что результат выполнения программ при заходе от пользователя guest и через root отличается. Теперь создадим файл readfile.c, применяя те же команды, что и в предыдущий раз.

10. Создание и редактирование файла

```
[guest@eliza ~]$ touch readfile.c
[guest@eliza ~]$ gedit readfile.c
```

Figure 2.11: создание и редактирование файла readfile.c

11. Написали программу readfile.c

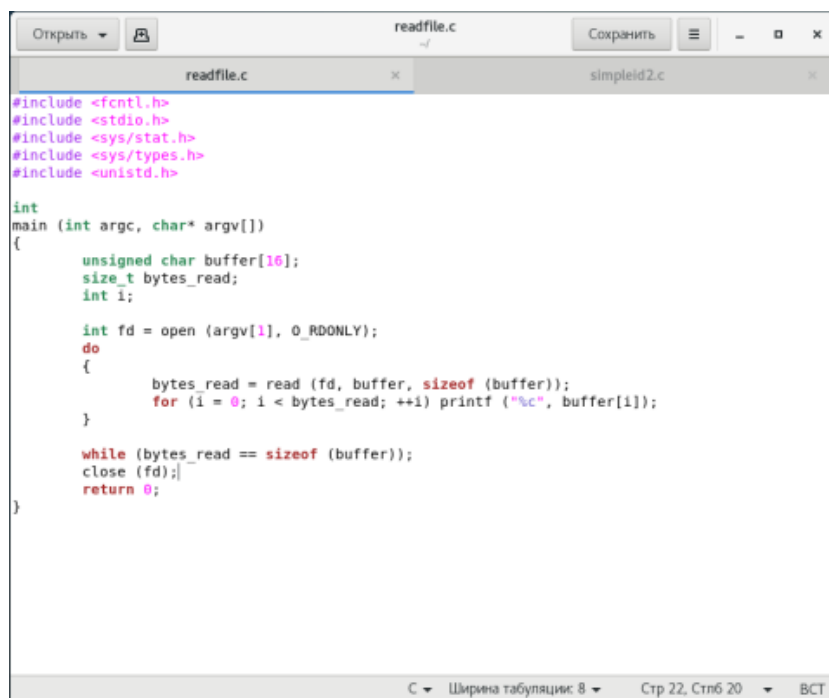


Figure 2.12: программа readfile

12. Откомпилировали программу. Затем сменили владельца у файла readfile.c и изменили права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог.

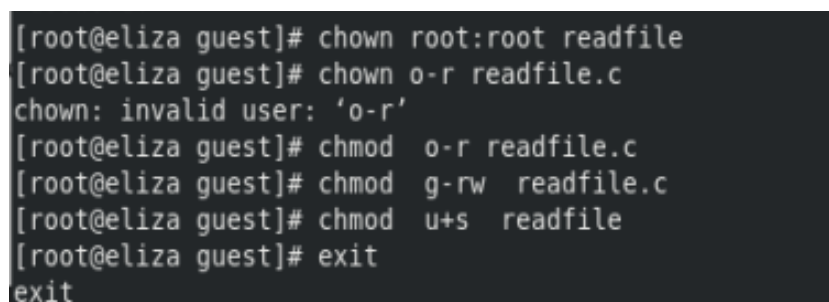


Figure 2.13: смена прав на файле readfile

13. Проверили, что пользователь guest не может прочитать файл readfile.c. Но увидели, что файл читается.

```
[guest@eliza ~]$ 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;
}
[guest@eliza ~]$
```

Figure 2.14: проверка файла на чтение

14. Ищем ошибку. Оказалось, что была допущена ошибка в установке прав. Переделали команду с установкой прав и повторили проверку, на этот раз в доступе отказано, как и должно быть. А команда ./readfile сработала потому что она выполняет чтение от имени суперпользователя, а не guest.

2.3 Исследование Sticky-бита

1. От имени пользователя guest создали файл file01.txt в директории /tmp со словом test. Просмотрели атрибуты у только что созданного файла и разрешили чтение и запись для категории пользователей «все остальные». Первоначально все группы имели право на чтение, а запись могли осуществлять все, кроме «остальных пользователей».

```
[guest@eliza ~]$ ls -l /tmp/file01.txt
ls: cannot access /tmp/file01.txt: No such file or directory
[guest@eliza ~]$ echo "test" > /tmp/file01.txt
[guest@eliza ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 Oct  7 22:48 /tmp/file01.txt
[guest@eliza ~]$ chmod o+rw /tmp/file01.txt
[guest@eliza ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 Oct  7 22:48 /tmp/file01.txt
[guest@eliza ~]$
```

Figure 2.15: проверка прав на файле file01.txt

2. От пользователя (не являющегося владельцем) попробовали прочитать файл /file01.txt. Мы видим, что в файле записано одно слово. далее к нему мы должны были поочередно дозаписать слова test2 и test3, но я забыла добавить по одной галочке и вместо дозаписи перезаписала слова. От пользователя попробовали удалить файл /tmp/file01.txt командой rm /tmp/file01.txt, однако получила отказ.

```
[root@eliza guest]# su guest2
[guest2@eliza guest]$ cd /tmp
[guest2@eliza tmp]$ cat file01.txt
test
[guest2@eliza tmp]$ echo "test2" > /tmp/file01.txt
[guest2@eliza tmp]$ cat file01.txt
test2
[guest2@eliza tmp]$
[guest2@eliza tmp]$ echo "test3" > /tmp/file01.txt
[guest2@eliza tmp]$ cat file01.txt
test3
[guest2@eliza tmp]$ rm file01.txt
rm: cannot remove 'file01.txt': Operation not permitted
[guest2@eliza tmp]$
```

Figure 2.16: дозапись и перезапись слов в файл file01.txt

3. От суперпользователя командой выполнили команду, снимающую атрибут t (Sticky-бит) с директории /tmp. Покинули режим суперпользователя командой exit. От пользователя проверили, что атрибута t у директории /tmp нет. Повторили предыдущие шаги. Получилось удалить файл.

```
[quest2@eliza tmp]$ su
Password:
[root@eliza tmp]# chmod -t /tmp
[root@eliza tmp]# exit
exit
[quest2@eliza tmp]$ ls -l / | grep tmp
drwxrwxrwx. 39 root root 4096 Oct  7 22:53 tmp
[root@eliza tmp]# echo "test2" >> /tmp/file01.txt
[root@eliza tmp]# cat file01.txt
test3
test2
[root@eliza tmp]# rm file01.txt
rm: remove regular file 'file01.txt'?
[root@eliza tmp]#
```

Figure 2.17: снятие атрибута

3 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

1. КОМАНДА CHATTR В LINUX
2. chattr