

Отчёт по лабораторной работе №5

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

Хусайнова Фароиз НКН6-01-18

Содержание

1	Цель работы	3
2	Выполнение лабораторной работы.....	4
2.1	Подготовка	4
2.2	Изучение механики SetUID.....	5
2.3	Исследование Sticky-бита	7
3	Выводы.....	10

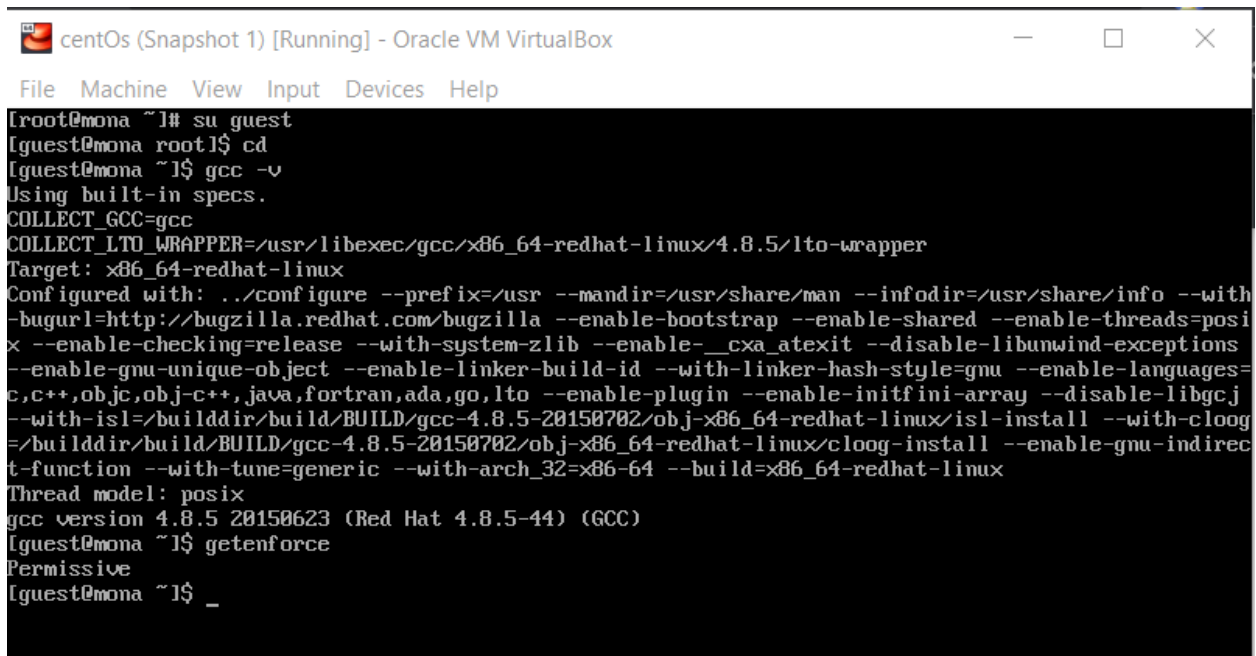
1 Цель работы

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.
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:

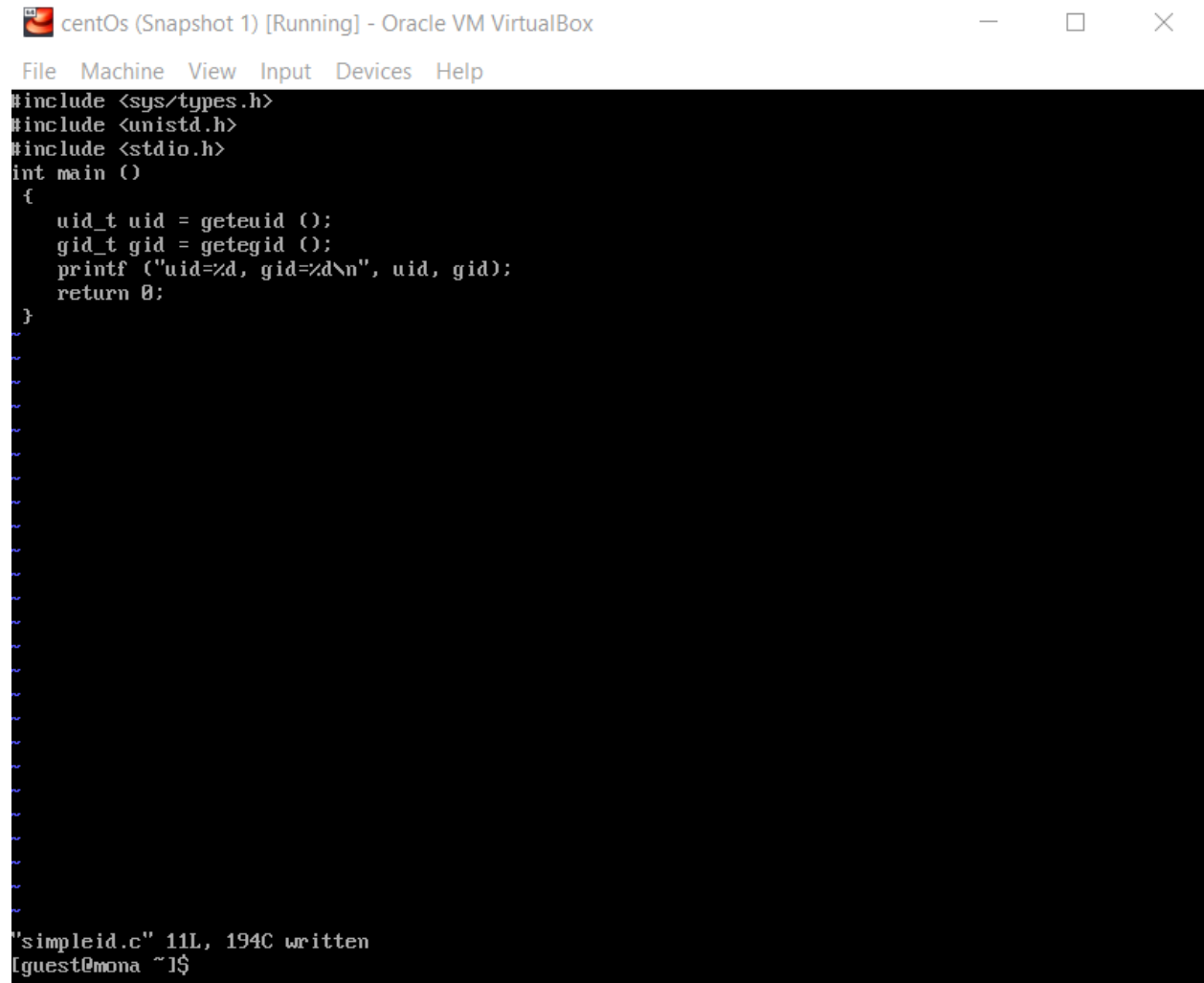


```
centOs (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
[root@mona ~]# su guest
[guest@mona root]$ cd
[guest@mona ~]$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/x86_64-redhat-linux/4.8.5/lto-wrapper
Target: x86_64-redhat-linux
Configured with: ../configure --prefix=/usr --mandir=/usr/share/man --infodir=/usr/share/info --with-
-bugurl=http://bugzilla.redhat.com/bugzilla --enable-bootstrap --enable-shared --enable-threads=posix
--enable-checking=release --with-system-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-array --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-4.8.5-20150702/obj-x86_64-redhat-linux/cloog-install --enable-gnu-indirec
t-function --with-tune=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)
[guest@mona ~]$ getenforce
Permissive
[guest@mona ~]$ _
```

Figure 1: подготовка к работе

2.2 Изучение механики SetUID

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



The screenshot shows a terminal window titled "centOs (Snapshot 1) [Running] - Oracle VM VirtualBox". The terminal displays the source code of a C program named simpleid.c. The code includes headers for sys/types.h, unistd.h, and stdio.h. The main function calls geteuid() and getegid() to retrieve the effective user and group IDs, prints them using printf, and then returns 0. The terminal also shows the command prompt [guest@mona ~]\$ and the output of the gcc compiler: "simpleid.c" 11L, 194C written.

```
centOs (Snapshot 1) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
#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;
}

[simpleid.c] 11L, 194C written
[guest@mona ~]$
```

Figure 2: программа simpleid

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

Figure 3: результат программы *simpleid*

6. Усложнили программу, добавив вывод действительных идентификаторов.

```
#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.c" 14L, 338C written
[guest@mona ~]$_
```

Figure 4: программа simpleid2

7. Скомпилювали 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. Прodeлали тоже самое относительно SetGID-бита.

```
[guest@mona ~]$ gcc simpleid2.c
[guest@mona ~]$ gcc simpleid2.c -o simpleid2
[guest@mona ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@mona ~]$ su
Password:
[root@mona guest]# chown root:guest simpleid2
[root@mona guest]# chmod u+s simpleid2
[root@mona guest]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@mona guest]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@mona guest]# chmod g+s simpleid2
[root@mona guest]# ls -l simpleid2
-rwsrwsr-x. 1 root guest 8576 Nov 13 17:42 simpleid2
[root@mona guest]# ./simpleid2
e_uid=0, e_gid=1001
real_uid=0, real_gid=0
[root@mona guest]# exit
exit
[guest@mona ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@mona ~]$ _
```

Figure 5: результат программы simpleid2

13. Написали программу readfile.c

```

#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" 22L, 444C written
[guest@mona ~]$_

```

Figure 6: программа readfile

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

2.3 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
```

```

[guest@mona ~]$ ls -l / | grep tmp
drwxrwxrwt.  7 root root 111 Nov 13 17:42 tmp
[guest@mona ~]$ echo "test" > /tmp/file01.txt
[guest@mona ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 Nov 13 18:09 /tmp/file01.txt
[guest@mona ~]$ chmod o+rw /tmp/file01.txt
[guest@mona ~]$ ls -l /tmpfile01.txt
ls: cannot access /tmpfile01.txt: No such file or directory
[guest@mona ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 Nov 13 18:09 /tmp/file01.txt
[guest@mona ~]$

```

```

[guest2@mona guest]$ cat /tmp/file01.txt
test2
[guest2@mona guest]$ echo "test3" > /tmp/file01.txt
[guest2@mona guest]$ cat /tmp/file01.txt
test3
[guest2@mona guest]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': Operation not permitted
[guest2@mona guest]$ su
Password:
[root@mona guest]# chmod -t /tmp
[root@mona guest]# exit
exit
[guest2@mona guest]$ ls -l / grep tmp
ls: cannot access grep: No such file or directory
ls: cannot access tmp: No such file or directory
/:
total 16
drwxrwxrwx.  1 root root   7 Sep 18 20:37 bin -> usr/bin
dr-xr-xr-x.  5 root root 4096 Sep 18 21:21 boot
drwxr-xr-x. 20 root root 3080 Nov 13 15:34 dev
drwxr-xr-x. 74 root root 8192 Nov 13 15:34 etc
drwxr-xr-x.  6 root root  64 Oct 30 16:07 home
drwxrwxrwx.  1 root root   7 Sep 18 20:37 lib -> usr/lib
drwxrwxrwx.  1 root root   9 Sep 18 20:37 lib64 -> usr/lib64
drwxr-xr-x.  2 root root   6 Apr 11 2018 media
drwxr-xr-x.  2 root root   6 Apr 11 2018 mnt
drwxr-xr-x.  2 root root   6 Apr 11 2018 opt
dr-xr-xr-x. 113 root root   8 Nov 13 15:33 proc
dr-xr-xr-x.  2 root root 135 Nov 13 16:44 root
drwxr-xr-x. 23 root root 700 Nov 13 15:34 run
drwxrwxrwx.  1 root root   8 Sep 18 20:37 sbin -> usr/sbin
drwxr-xr-x.  2 root root   6 Apr 11 2018 srv
dr-xr-xr-x. 13 root root   8 Nov 13 15:33 sys
drwxrwxrwx.  7 root root 111 Nov 13 17:16 tmp
drwxr-xr-x. 13 root root 155 Sep 18 20:37 usr
drwxr-xr-x. 19 root root 267 Sep 18 21:25 var
[guest2@mona guest]$ _

```

```

[guest2@mona guest]$ su -
Password:
Last login: Sat Nov 13 17:24:46 MSK 2021 on tty1
[root@mona ~]# chmod +t /tmp
[root@mona ~]# exit
logout

```

Figure 8&9&10 : исследование Sticky-бита

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.