

Лабораторная работа N° 5

Информационная безопасность

Леон Фернандо Хосе Фернандо | НПМбд02-20

Содержание

1 Цель работы	4
2 Теоретическое введение	4
3 Выполнение лабораторной работы	4
3.1 Создание программы	4
3.2 Исследование Sticky-бита	10
4 Выводы	12
5 Список Литературы	12

Список иллюстраций

Рисунок 1. Предварительная подготовка	5
Рисунок 2. Команда “whereis”	5
Рисунок 3. Вход в систему и создание программы	6
Рисунок 4. Код программы simpleid.c	6
Рисунок 5. Компиляция и выполнение программы simpleid	6
Рисунок 6. Усложнение программы	7
Рисунок 7. Переименование программы в simpleid2.c	7
Рисунок 8. Компиляция и выполнение программы simpleid2	7
Рисунок 9. Установка новых атрибутов (SetUID)	8
Рисунок 10. Запуск simpleid2 после установки SetUID	8
Рисунок 11. Запуск simpleid2 после установки SetGID	8
Рисунок 12. Код программы readfile.c	9
Рисунок 13. Смена владельца и прав доступа у файла readfile.c	9
Рисунок 14. Запуск программы readfile	10
Рисунок 15. Создание файла file01.txt	10
Рисунок 16. Попытка выполнить действия над файлом file01.txt от имени пользо	11
Рисунок 17. Удаление атрибута t (Sticky-бита)	11
Рисунок 18. Возвращение атрибута t (Sticky-бита)	12

1 Цель работы

Изучение механизмов модификации идентификатора, использования SetUID и Sticky bits. Получение практических навыков использования консольных команд с дополнительными атрибутами. Изучается работа механизма изменения идентификатора пользовательского процесса, а также влияние бита Sticky на запись и удаление файлов.

2 Теоретическое введение

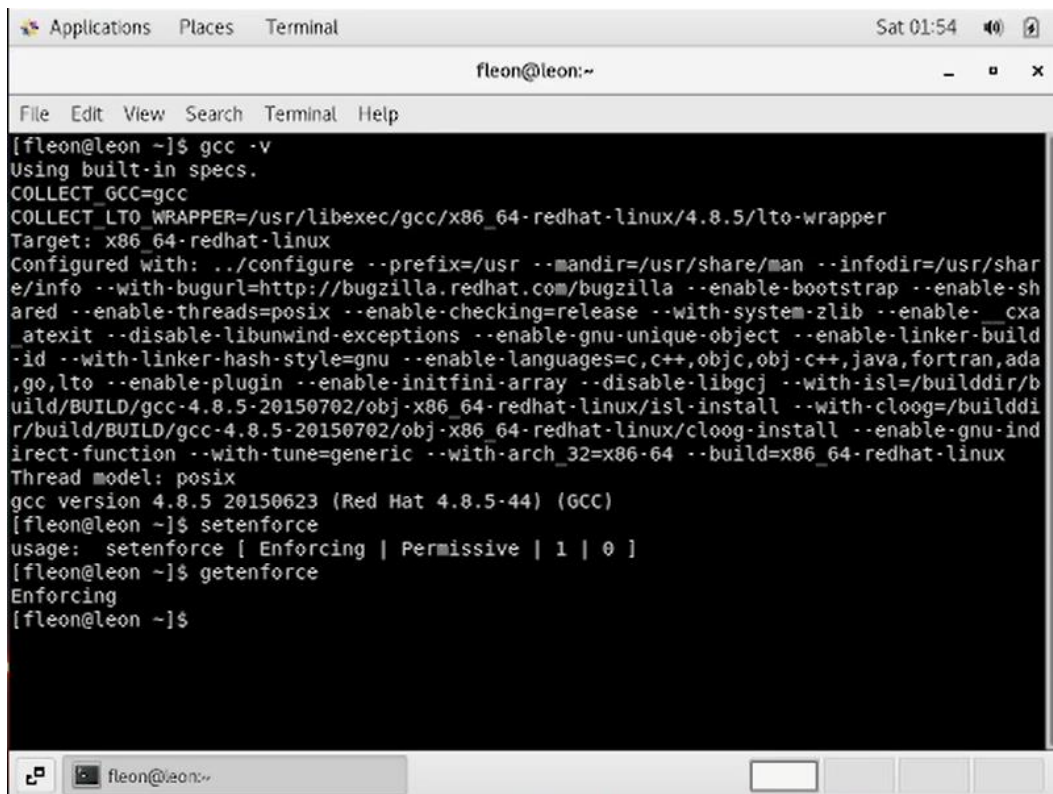
SetUID, SetGID и Sticky - это специальные типы разрешений позволяют задавать расширенные права доступа на файлы или каталоги.

- SetUID (set user ID upon execution — «установка ID пользователя во время выполнения) являются флагами прав доступа в Unix, которые разрешают пользователям запускать исполняемые файлы с правами владельца исполняемого файла.
- SetGID (set group ID upon execution — «установка ID группы во время выполнения») являются флагами прав доступа в Unix, которые разрешают пользователям запускать исполняемые файлы с правами группы исполняемого файла.
- Sticky bit в основном используется в общих каталогах, таких как /var или /tmp, поскольку пользователи могут создавать файлы, читать и выполнять их, принадлежащие другим пользователям, но не могут удалять файлы, принадлежащие другим пользователям.

3 Выполнение лабораторной работы

3.1 Создание программы

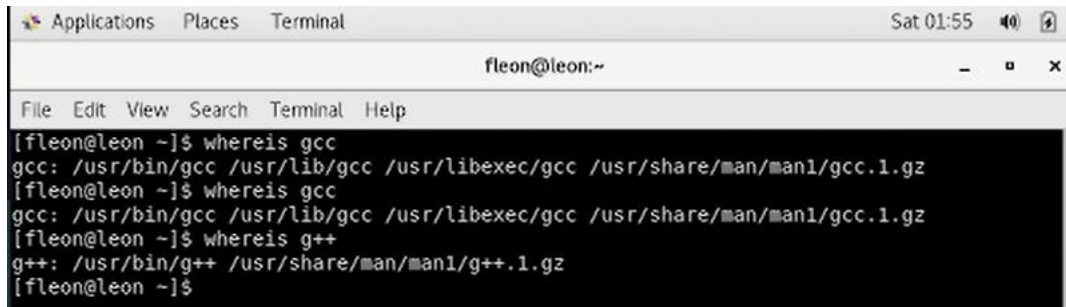
Для начала я проверил наличие компилятора gcc, используя команду 'gcc -v'. Затем я отключил системные ограничения до следующей перезагрузки системы с помощью команды "sudo setenforce 0", после чего команда "getenforce" отобразила "Разрешающий".



```
fleon@leon:~$ 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-libgck --with-isl=/build/gcc-4.8.5-20150702/obj-x86_64-redhat-linux/isl-install --with-cloog=/build/gcc-4.8.5-20150702/obj-x86_64-redhat-linux/cloog-install --enable-gnu-indirect-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)
[fleon@leon ~]$ setenforce
usage: setenforce [ Enforcing | Permissive | 1 | 0 ]
[fleon@leon ~]$ getenforce
Enforcing
[fleon@leon ~]$
```

Рисунок 1. Предварительная подготовка

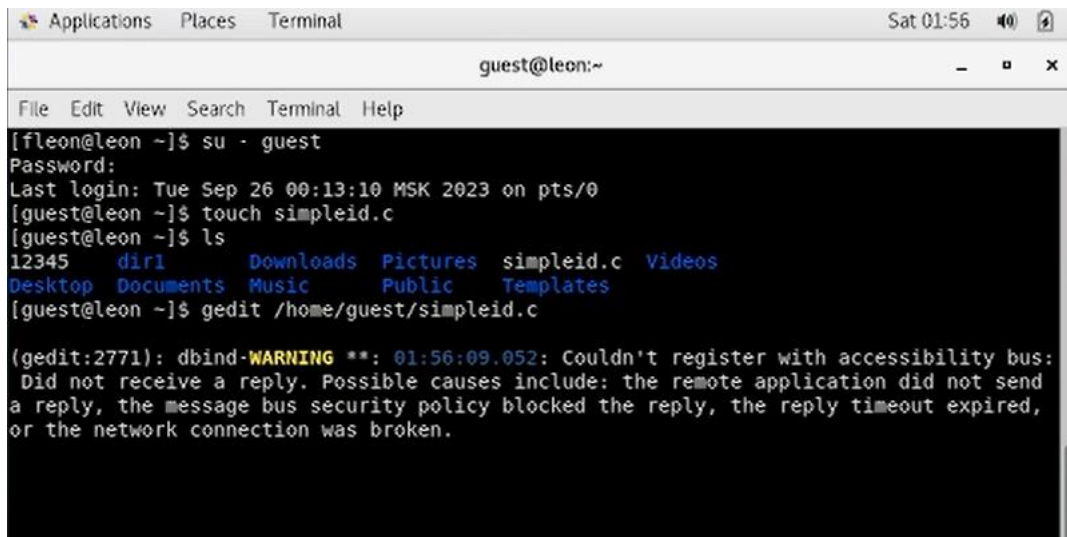
Проверить успешное выполнение команд “whereis gcc” и “whereis g++” (их расположение).



```
[fleon@leon ~]$ whereis gcc
gcc: /usr/bin/gcc /usr/lib/gcc /usr/libexec/gcc /usr/share/man/man1/gcc.1.gz
[fleon@leon ~]$ whereis gcc
gcc: /usr/bin/gcc /usr/lib/gcc /usr/libexec/gcc /usr/share/man/man1/gcc.1.gz
[fleon@leon ~]$ whereis g++
g++: /usr/bin/g++ /usr/share/man/man1/g++.1.gz
[fleon@leon ~]$
```

Рисунок 2. Команда “whereis”

Я вошел в систему как пользователь "гость", используя команду "su - guest". Я создал программу под названием "simplified.c" с помощью команды "touch simple id.c" и открыл ее в редакторе, используя команду "gedit /home/guest/simpleid.c".



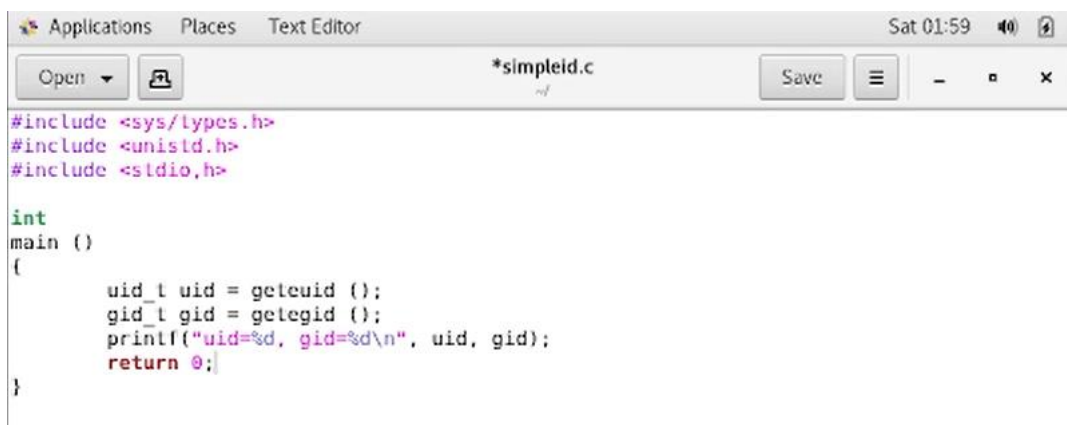
```
Applications  Places  Terminal  Sat 01:56
guest@leon:~

File Edit View Search Terminal Help
[flleon@leon ~]$ su - guest
Password:
Last login: Tue Sep 26 00:13:10 MSK 2023 on pts/0
[guest@leon ~]$ touch simpleid.c
[guest@leon ~]$ ls
12345  dirl  Downloads  Pictures  simpleid.c  Videos
Desktop  Documents  Music  Public  Templates
[guest@leon ~]$ gedit /home/guest/simpleid.c

(gedit:2771): dbind-WARNING **: 01:56:09.052: Couldn't register with accessibility bus:
Did not receive a reply. Possible causes include: the remote application did not send
a reply, the message bus security policy blocked the reply, the reply timeout expired,
or the network connection was broken.
```

Рисунок 3. Вход в систему и создание программы

Код программы выглядит следующим образом.



```
Applications  Places  Text Editor  Sat 01:59
*simpleid.c

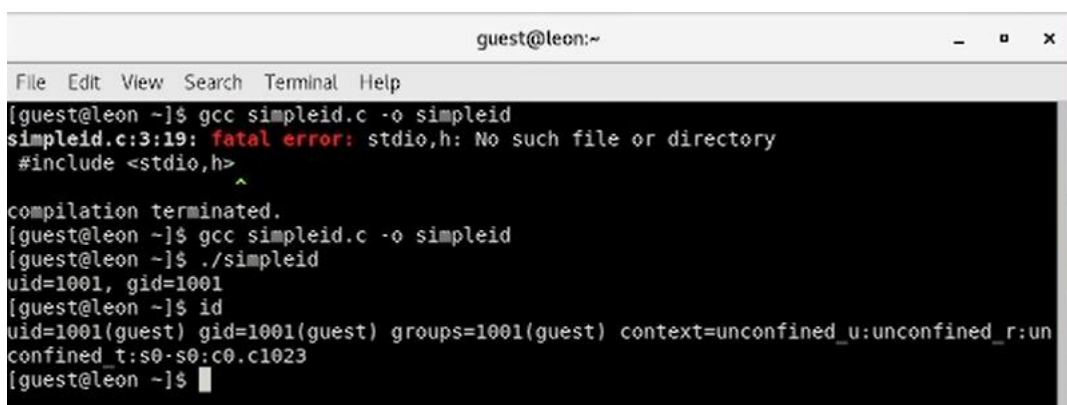
Open  Save  -  +  x

#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;
}
```

Рисунок 4. Код программы simpleid.c

Я скомпилировал программу и убедился, что программный файл был создан с помощью команды 'gcc simpleid.c -o simpleid'. Я выполнил программу 'simpleid' с помощью команды './simpleid', а затем запустил системную программу 'id' с помощью команды 'id'. Результаты, полученные в результате выполнения обеих команд, совпали (uid=1001 и gid=1001).



```
guest@leon:~

File Edit View Search Terminal Help
[guest@leon ~]$ gcc simpleid.c -o simpleid
simpleid.c:3:19: fatal error: stdio.h: No such file or directory
#include <stdio.h>
^
compilation terminated.
[guest@leon ~]$ gcc simpleid.c -o simpleid
[guest@leon ~]$ ./simpleid
uid=1001, gid=1001
[guest@leon ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@leon ~]$
```

Рисунок 5. Компиляция и выполнение программы simpleid

Усложнила программу, добавив вывод действительных идентификаторов.



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

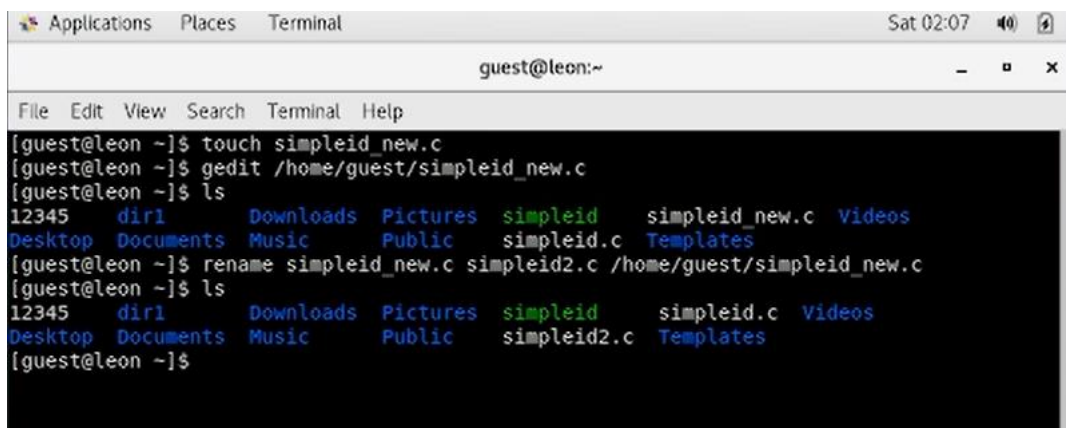
int
main ()
{
    uid_t real_uid = geteuid ();
    uid_t e_uid = geteuid ();

    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;
}
```

Рисунок 6. Усложнение программы


Получившуюся программу назвала simpleid2.c



```
guest@leon:~
File Edit View Search Terminal Help
[guest@leon ~]$ touch simpleid_new.c
[guest@leon ~]$ gedit /home/guest/simpleid_new.c
[guest@leon ~]$ ls
12345  dir1  Downloads  Pictures  simpleid  simpleid_new.c  Videos
Desktop Documents Music  Public  simpleid.c  Templates
[guest@leon ~]$ rename simpleid_new.c simpleid2.c /home/guest/simpleid_new.c
[guest@leon ~]$ ls
12345  dir1  Downloads  Pictures  simpleid  simpleid.c  Videos
Desktop Documents Music  Public  simpleid2.c  Templates
[guest@leon ~]$
```

Рисунок 7. Переименование программы в simpleid2.c

Скомпилировала и запустила simpleid2.c командами "gcc simpleid2.c -o sipleid2" и "./simpleid2".

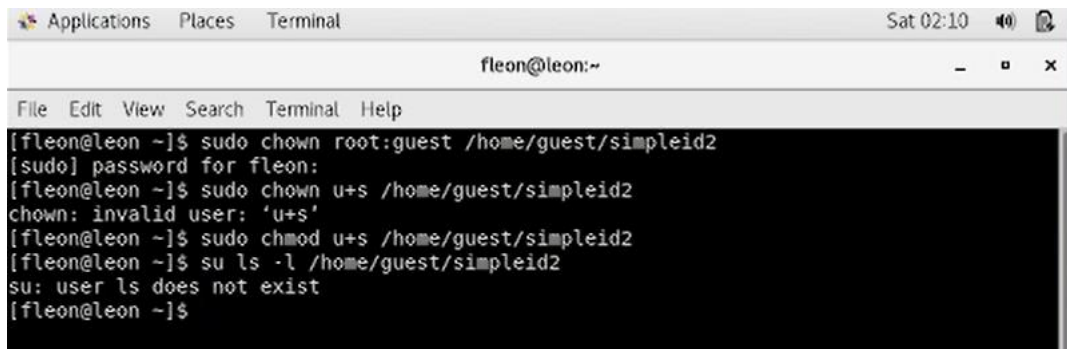


```
guest@leon:~
File Edit View Search Terminal Help
[guest@leon ~]$ gcc simpleid2.c -o simpleid2
[guest@leon ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@leon ~]$
```

Рисунок 8. Компиляция и выполнение программы simpleid2

От имени суперпользователя я выполнил команды "sudo chown root:guest /home/guest/simpleid2" и "sudo chmod u+s /home/guest/simpleid2". После этого я проверил правильность установки новых атрибутов и смену владельца файла 'simpleid2', используя

команду 'sudo ls -l /home/guest/simpleid2'. Эти команды привели к изменению пользователя файла на "root" и установке бита SetUID.



```
Applications  Places  Terminal  Sat 02:10
fleon@leon:~
File Edit View Search Terminal Help
[fleon@leon ~]$ sudo chown root:guest /home/guest/simpleid2
[sudo] password for fleon:
[fleon@leon ~]$ sudo chown u+s /home/guest/simpleid2
chown: invalid user: 'u+s'
[fleon@leon ~]$ sudo chmod u+s /home/guest/simpleid2
[fleon@leon ~]$ su ls -l /home/guest/simpleid2
su: user ls does not exist
[fleon@leon ~]$
```

Рисунок 9. Установка новых атрибутов (SetUID)

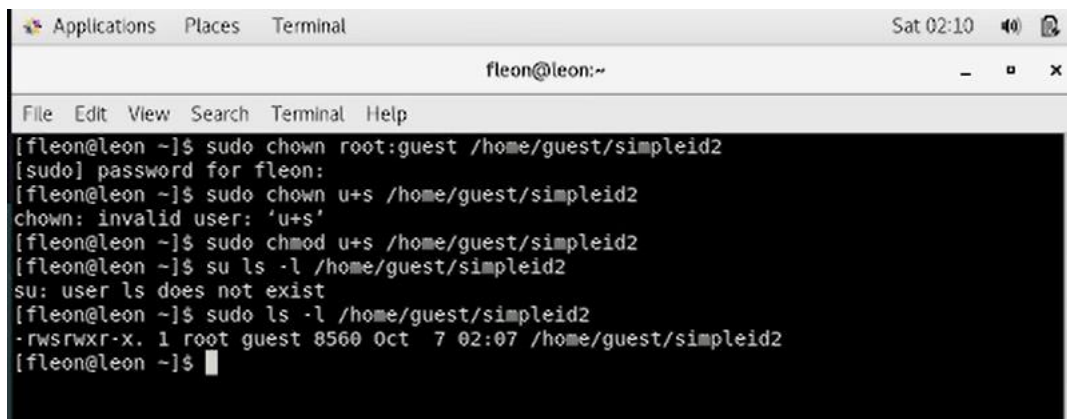
Запустил программы simpleid2 и id. Теперь появились различия в uid



```
Applications  Places  Terminal  Sat 02:13
guest@leon:~
File Edit View Search Terminal Help
[guest@leon ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@leon ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@leon ~]$
```

Рисунок 10. Запуск simpleid2 после установки SetUID

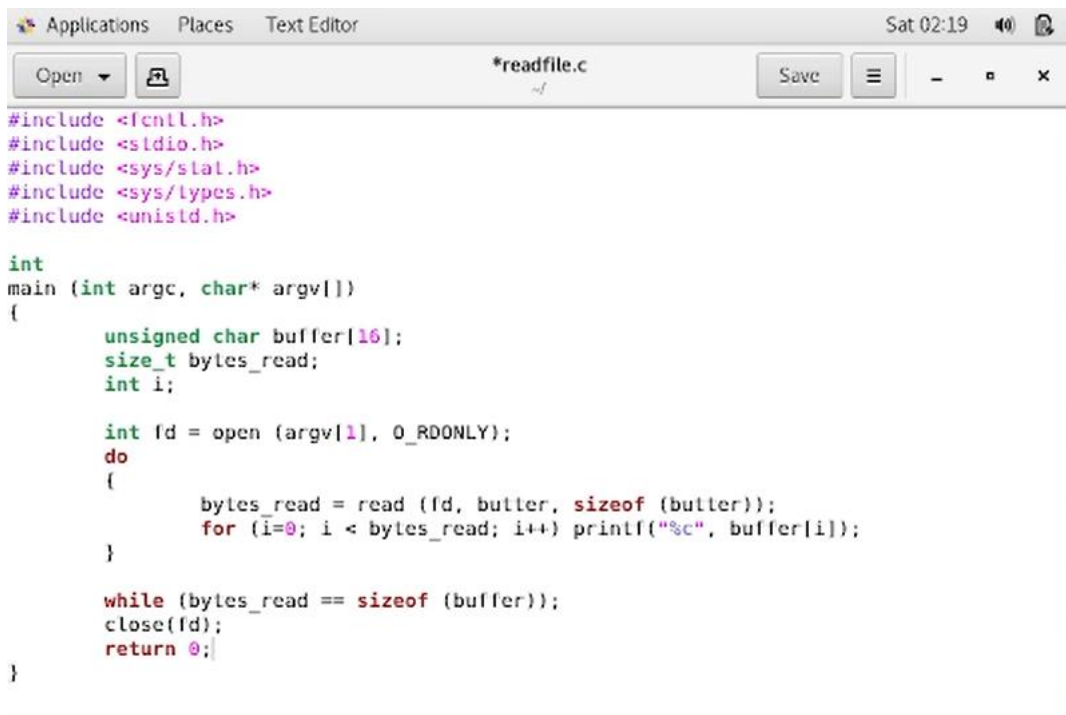
Проделал тоже самое относительно SetGID-бита. Также можем заметить различия с предыдущим пунктом.



```
Applications  Places  Terminal  Sat 02:10
fleon@leon:~
File Edit View Search Terminal Help
[fleon@leon ~]$ sudo chown root:guest /home/guest/simpleid2
[sudo] password for fleon:
[fleon@leon ~]$ sudo chown u+s /home/guest/simpleid2
chown: invalid user: 'u+s'
[fleon@leon ~]$ sudo chmod u+s /home/guest/simpleid2
[fleon@leon ~]$ su ls -l /home/guest/simpleid2
su: user ls does not exist
[fleon@leon ~]$ sudo ls -l /home/guest/simpleid2
-rwsrwxr-x. 1 root guest 8560 Oct  7 02:07 /home/guest/simpleid2
[fleon@leon ~]$
```

Рисунок 11. Запуск simpleid2 после установки SetGID

Создаем программы readfile.c



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

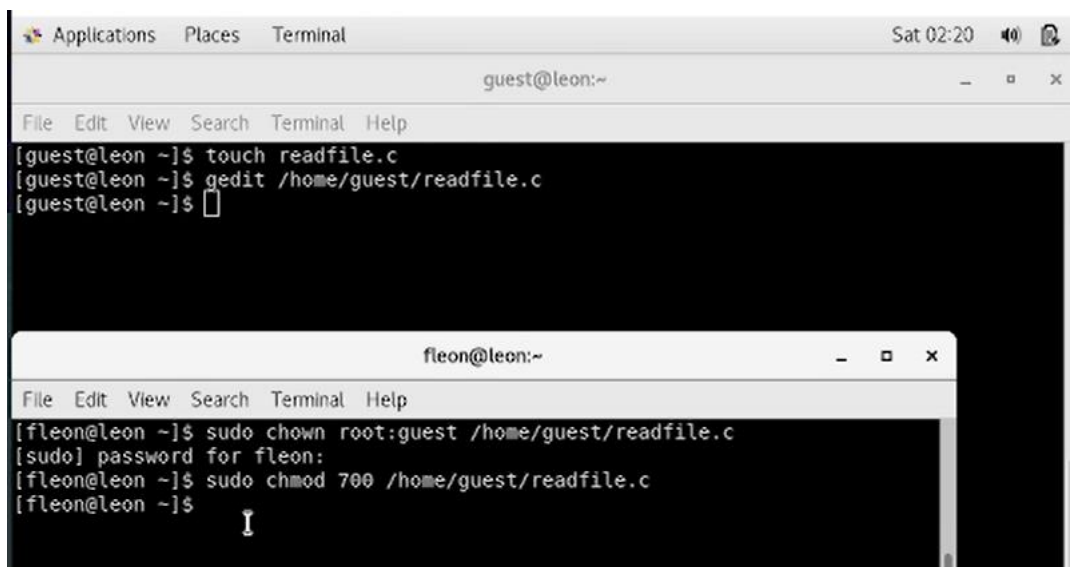
int
main (int argc, char* argv[])
{
    unsigned char buffer[16];
    size_t bytes_read;
    int i;

    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;
}
```

Рисунок 12. Код программы `readfile.c`

Я скомпилировал созданную программу, используя команду `'gcc readfile.c -o readfile'`. Я изменил владельца файла `'readfile.c'` с помощью команды `'sudo chown root:guest /home/гость / readfile.c'` и настроил разрешения так, чтобы только суперпользователь мог его прочитать, в то время как `'гость'` не мог, используя команду `'sudo chmod 700 /home/гость/readfile.c'`. Теперь я убедился, что пользователь "гость" не может прочитать файл `"read file.c"` с помощью команды `"cat readfile.c"` и получил сообщение об ошибке `"отказано в доступе"`.

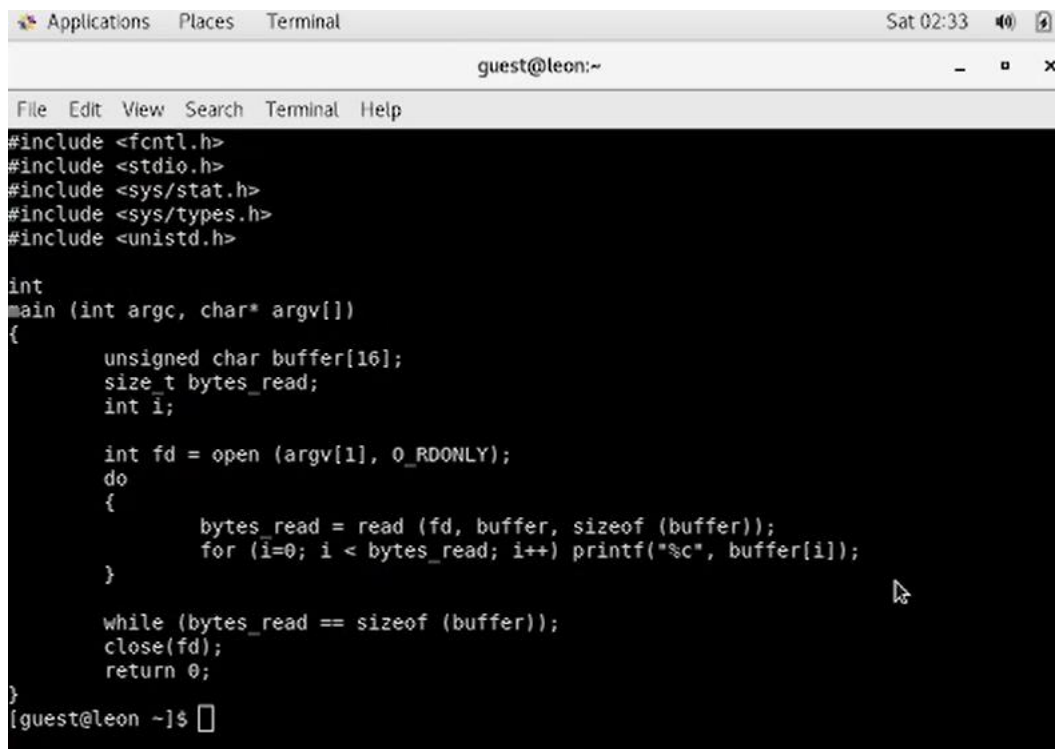


```
guest@leon:~
[guest@leon ~]$ touch readfile.c
[guest@leon ~]$ gedit /home/guest/readfile.c
[guest@leon ~]$

fleon@leon:~
[fleon@leon ~]$ sudo chown root:guest /home/guest/readfile.c
[sudo] password for fleon:
[fleon@leon ~]$ sudo chmod 700 /home/guest/readfile.c
[fleon@leon ~]$
```

Рисунок 13. Смена владельца и прав доступа у файла `readfile.c`

Я сменил владельца программы `"readfile"` и установил SetUID. Я проверил, может ли программа `"readfile"` прочитать файл `"read file.c"`, используя команду `"/.readfile readfile.c"`. Он смог это прочитать. Аналогично, я проверил, возможно ли прочитать файл `'/etc/shadow'`, и это также прошло успешно.



```
#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;

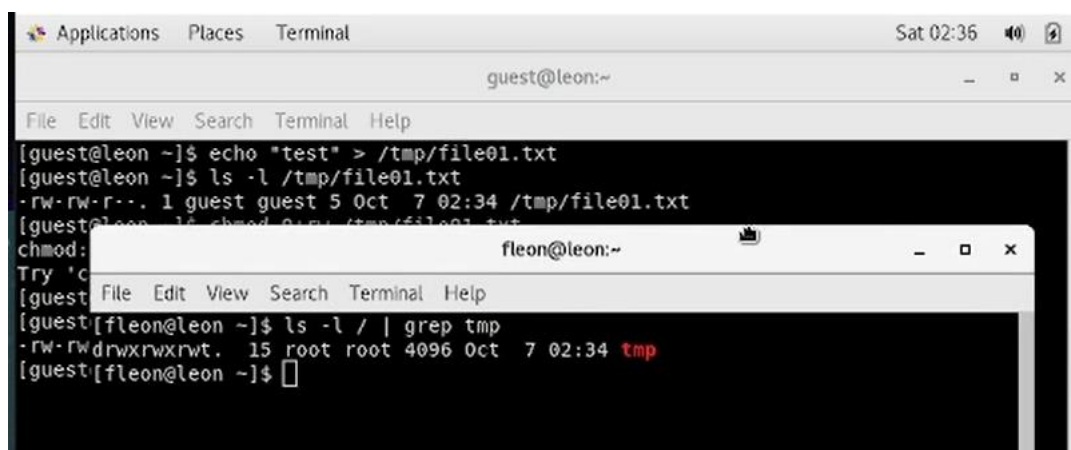
    int fd = open (argv[1], O_RDONLY);
    do
    {
        bytes_read = read (fd, buffer, sizeof (buffer));
        for (i=0; i < bytes_read; i++) printf("%c", buffer[i]);
    }

    while (bytes_read == sizeof (buffer));
    close(fd);
    return 0;
}
[guest@leon ~]$
```

Рисунок 14. Запуск программы readfile

3.2 Исследование Sticky-бита

Я подтвердил, что атрибут Sticky был установлен в каталоге '/tmp', используя команду 'ls -l / | grep tmp'. От имени пользователя "гость" я создал файл с именем 'file01.txt' в каталоге '/tmp' со словом 'test' с помощью команды 'echo "test" > /tmp/file01.txt'. Я проверил атрибуты вновь созданного файла и предоставил разрешения на чтение и запись для категории пользователей "все остальные", используя команды 'ls -l /tmp/file01.txt' и 'chmod o+rw /tmp/file01.txt'.

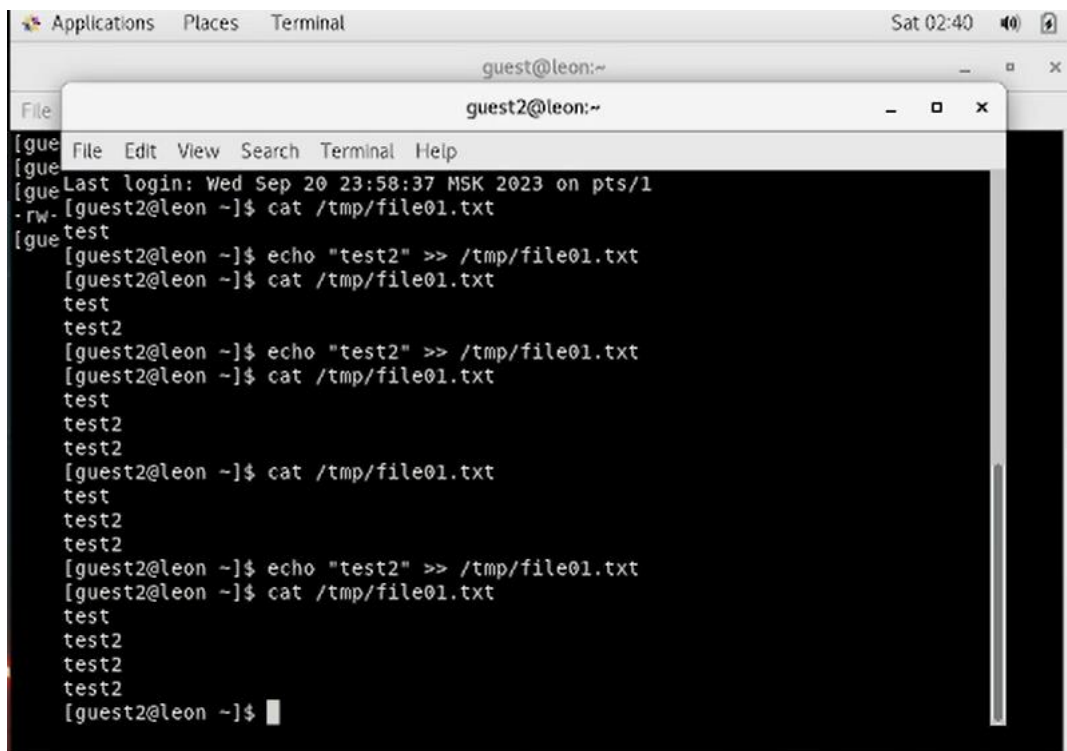


```
[guest@leon ~]$ echo "test" > /tmp/file01.txt
[guest@leon ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 Oct 7 02:34 /tmp/file01.txt
[guest@leon ~]$ chmod o+rw /tmp/file01.txt
chmod:
Try 'c
[guest@fleon@leon ~]$ ls -l / | grep tmp
-rw-rw-rw-rwt. 15 root root 4096 Oct 7 02:34 tmp
[guest@fleon@leon ~]$
```

Рисунок 15. Создание файла file01.txt

От имени пользователя "guest2" я попытался прочитать файл, используя команду "cat /tmp/file01.txt", и это было успешно. Затем я попытался добавить слово "test2" к файлу, проверить его содержимое и записать в файл "test3", удалив при этом всю существующую информацию. Эти операции были успешными только тогда, когда я дополнительно предоставил разрешения на чтение и запись для "группы" пользователей, используя

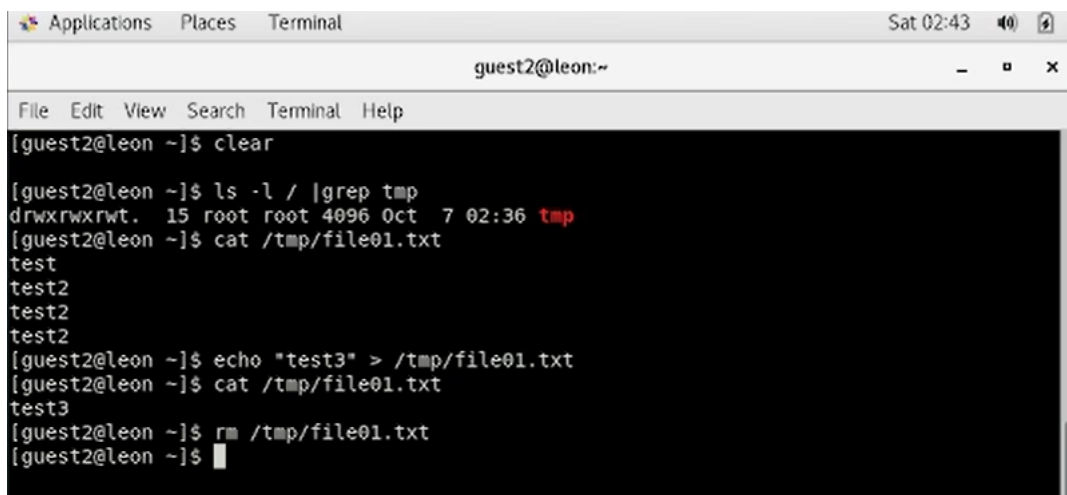
команду "chmod g+rw /tmp/file01.txt ". Однако, когда я попытался удалить файл от имени пользователя 'guest2', это не было возможно ни в одном из случаев, и произошла ошибка.



```
Applications  Places  Terminal  Sat 02:40
guest@leon:~
File Edit View Search Terminal Help
[guest@leon ~]$ cat /tmp/file01.txt
test
[guest@leon ~]$ echo "test2" >> /tmp/file01.txt
[guest@leon ~]$ cat /tmp/file01.txt
test
test2
[guest@leon ~]$ echo "test2" >> /tmp/file01.txt
[guest@leon ~]$ cat /tmp/file01.txt
test
test2
test2
[guest@leon ~]$ cat /tmp/file01.txt
test
test2
test2
[guest@leon ~]$ echo "test2" >> /tmp/file01.txt
[guest@leon ~]$ cat /tmp/file01.txt
test
test2
test2
test2
[guest@leon ~]$
```

Рисунок 16. Попытка выполнить действия над файлом file01.txt от имени пользо.

Я повысил разрешения до статуса суперпользователя с помощью команды 'su -' и выполнил команду для удаления атрибута 't' из каталога '/tmp' с помощью 'chmod -t /tmp'. После этого я вышел из режима суперпользователя с помощью команды "exit". Я повторил предыдущие шаги. Теперь я смог удалить 'file01.txt " файл от имени пользователя, который не был его владельцем.

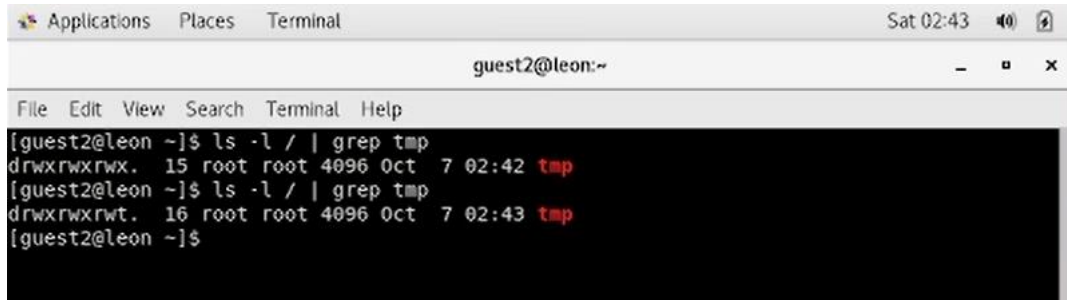


```
Applications  Places  Terminal  Sat 02:43
guest2@leon:~
File Edit View Search Terminal Help
[guest2@leon ~]$ clear
[guest2@leon ~]$ ls -l / |grep tmp
drwxrwxrwt. 15 root root 4096 Oct 7 02:36 tmp
[guest2@leon ~]$ cat /tmp/file01.txt
test
test2
test2
test2
[guest2@leon ~]$ echo "test3" > /tmp/file01.txt
[guest2@leon ~]$ cat /tmp/file01.txt
test3
[guest2@leon ~]$ rm /tmp/file01.txt
[guest2@leon ~]$
```

Рисунок 17. Удаление атрибута t (Sticky-бита)

Повысила свои права до суперпользователя и вернула атрибут t на директорию

/tmp



```
Applications Places Terminal Sat 02:43
guest2@leon:~
File Edit View Search Terminal Help
[guest2@leon ~]$ ls -l / | grep tmp
drwxrwxrwx. 15 root root 4096 Oct 7 02:42 tmp
[guest2@leon ~]$ ls -l / | grep tmp
drwxrwxrwt. 16 root root 4096 Oct 7 02:43 tmp
[guest2@leon ~]$
```

Рисунок 18. Возвращение атрибута t (Sticky-бита)

4 Выводы

Во время выполнения этого лабораторного задания я изучил механизмы модификации идентификаторов, применение SetUID и Sticky bits, а также приобрел практические навыки использования консоли с дополнительными атрибутами. Я изучил работу механизма изменения идентификатора пользовательского процесса и влияние Sticky-бита на запись и удаление файлов.

5 Список Литературы

1. Стандартные права SetUID, SetGID, Sticky в Linux [Электронный ресурс]. URL: <https://linux-notes.org/standartny-e-prava-unix-suid-sgid-sticky-bity/>.