

Лабораторная работа №5

Информационная безопасность

Волчок Кристина Александровна НПМбд-02-21

Содержание

1	Цель работы	4
2	Теоретическое введение	5
3	Выполнение лабораторной работы	6
4	Выводы	18
5	Список литературы	19

Список иллюстраций

3.1	Предварительная подготовка	6
3.2	Вход в систему и создание программы	7
3.3	Код программы simpleid.c	8
3.4	Компиляция и выполнение программы simpleid	9
3.5	Усложнение программы	10
3.6	Переименование программы в simpleid2.c	10
3.7	Переименование программы в simpleid2.c	11
3.8	Установка новых атрибутов (SetUID) и смена владельца файла . . .	12
3.9	Запуск simpleid2 после установки SetUID	12
3.10	Смена владельца и прав доступа у файла readfile.c	13
3.11	Запуск программы readfile	14
3.12	Создание файла file01.txt	15
3.13	Создание файла file01.txt	16
3.14	Удаление атрибута t (Sticky-бита) и повторение действий	17

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.

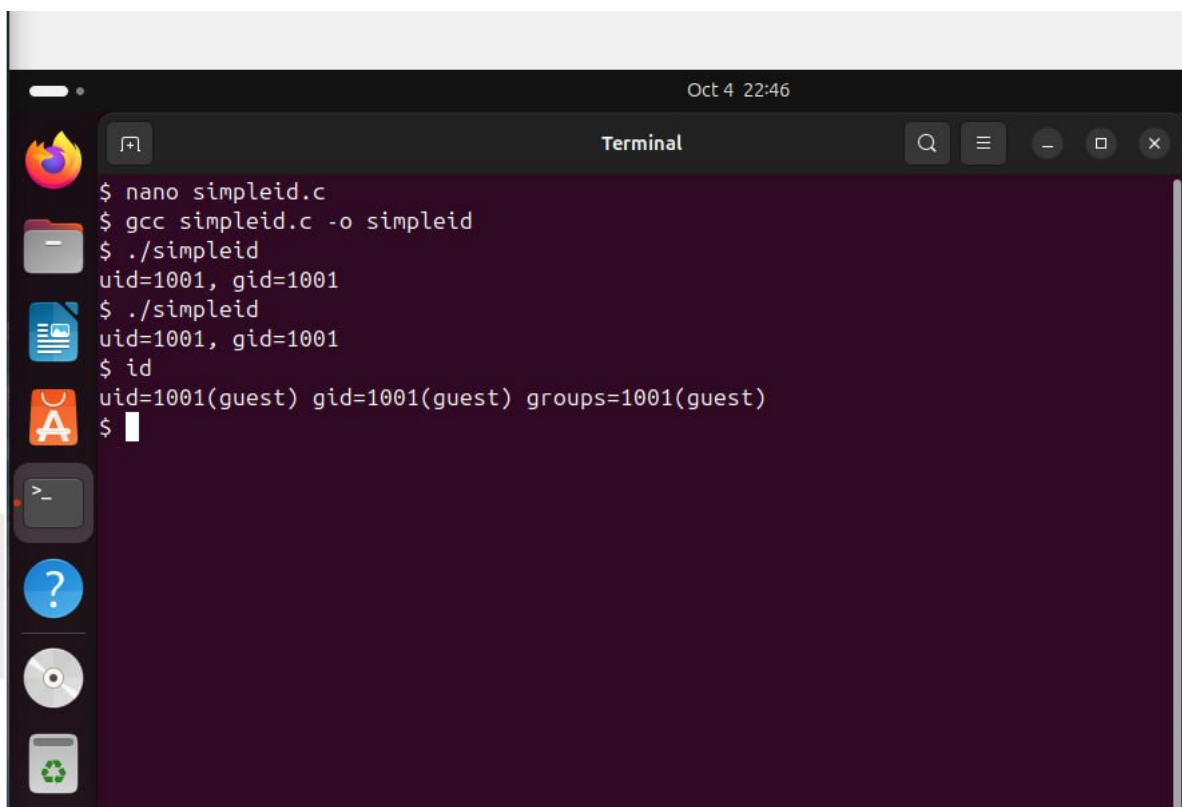
2 Теоретическое введение

SetUID, SetGID и Sticky - это специальные типы разрешений, которые 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

Создание программы:

Для начала я убедилась, что компилятор gcc установлен, используя команду `gcc -v`. Затем отключила систему запретов до очередной перезагрузки системы командой `sudo setenforce 0`, после чего команда `getenforce` вывела “Permissive”.

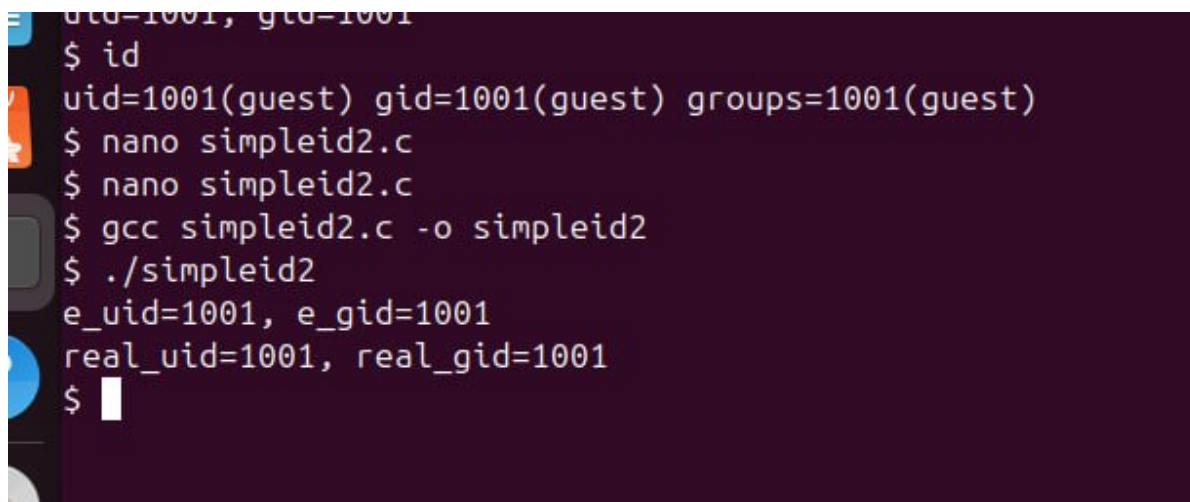


```
Oct 4 22:46
Terminal
$ nano simpleid.c
$ gcc simpleid.c -o simpleid
$ ./simpleid
uid=1001, gid=1001
$ ./simpleid
uid=1001, gid=1001
$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest)
$
```

Рис. 3.1: Предварительная подготовка

Проверила успешное выполнение команд “`whereis gcc`” и “`whereis g++`” (их расположение). Вошла в систему от имени пользователя `guest` командой “`su -`

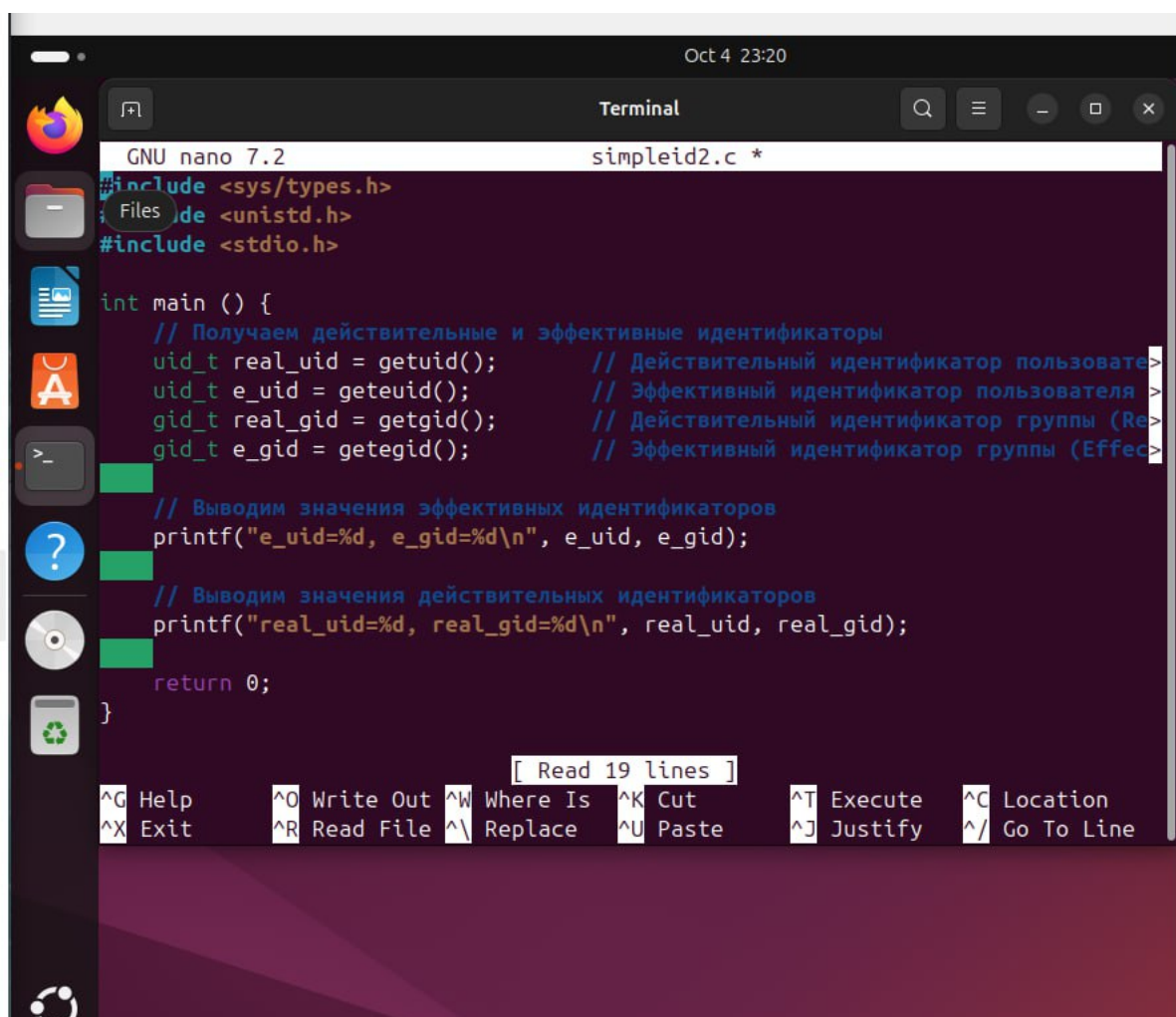
guest”. Создала программу simpleid.c командой “touch simpleid.c” и открыла её в редакторе командой “gedit /home/guest/simpleid.c”.

A terminal window with a dark purple background and light green text. The text shows the output of the 'id' command, followed by the creation and compilation of 'simpleid2.c' using 'nano' and 'gcc', and finally the execution of './simpleid2' which prints effective and real user/group IDs.

```
uid=1001, gid=1001
$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest)
$ nano simpleid2.c
$ nano simpleid2.c
$ gcc simpleid2.c -o simpleid2
$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
$
```

Рис. 3.2: Вход в систему и создание программы

Код программы выглядит следующим образом.



The image shows a terminal window titled "Terminal" with a timestamp of "Oct 4 23:20". The editor is "GNU nano 7.2" editing "simpleid2.c *". The code is as follows:

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

int main () {
    // Получаем действительные и эффективные идентификаторы
    uid_t real_uid = getuid();      // Действительный идентификатор пользователя
    uid_t e_uid = geteuid();        // Эффективный идентификатор пользователя
    gid_t real_gid = getgid();      // Действительный идентификатор группы (Real)
    gid_t e_gid = getegid();       // Эффективный идентификатор группы (Effective)

    // Выводим значения эффективных идентификаторов
    printf("e_uid=%d, e_gid=%d\n", e_uid, e_gid);

    // Выводим значения действительных идентификаторов
    printf("real_uid=%d, real_gid=%d\n", real_uid, real_gid);

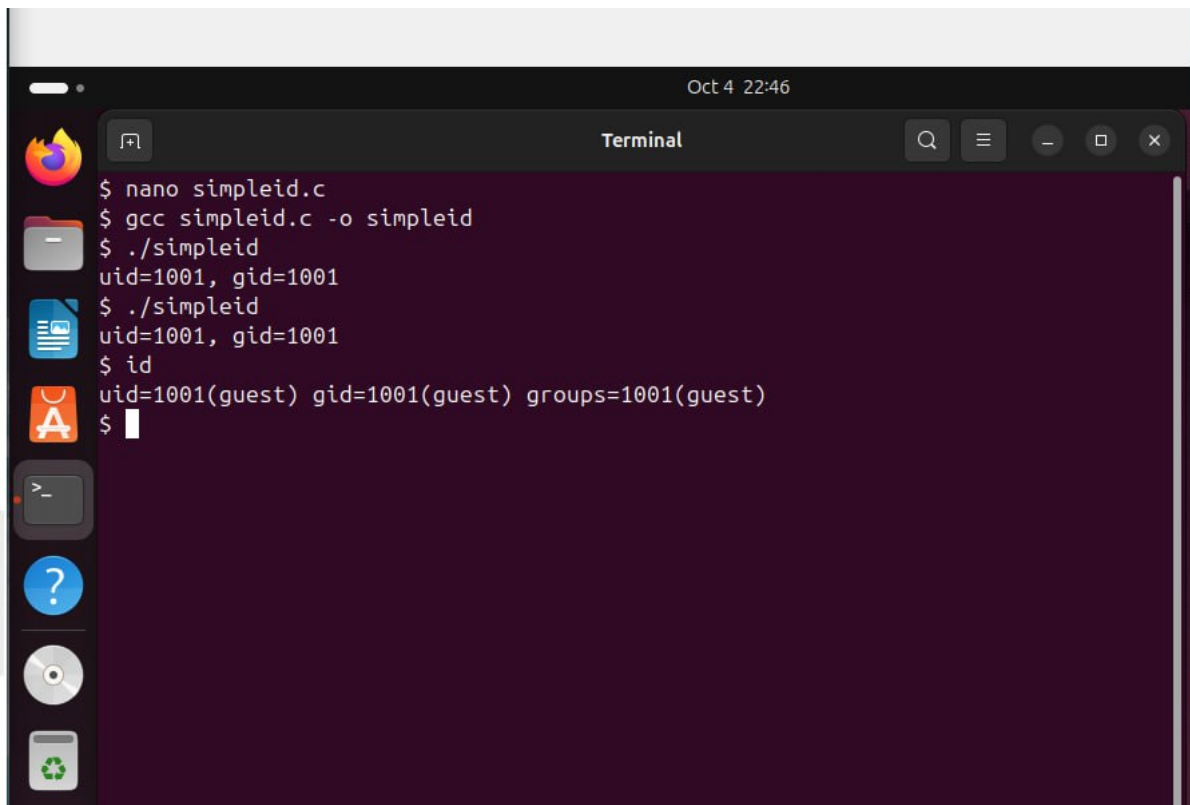
    return 0;
}
```

At the bottom of the terminal, there is a status bar with the text "[Read 19 lines]" and a table of keyboard shortcuts:

^G Help	^O Write Out	^W Where Is	^K Cut	^T Execute	^C Location
^X Exit	^R Read File	^_ Replace	^U Paste	^J Justify	^_ Go To Line

Рис. 3.3: Код программы simpleid.c

Скомпилировала программу и убедилась, что файл программы был создан командой “gcc simpleid.c -o simpleid”. Выполнила программу simpleid командой “./simpleid”, а затем выполнила системную программу id командой “id”. Результаты, полученные в результате выполнения обеих команд, совпадают (uid=1001 и gid=1001) .

A screenshot of a macOS Terminal window. The title bar shows the date and time as 'Oct 4 22:46'. The terminal has a dark purple background with white text. On the left side, there is a vertical dock with icons for Firefox, a folder, a document, an application, a terminal, a help icon, a CD-ROM, and a trash can. The terminal content shows the following commands and output:

```
$ nano simpleid.c
$ gcc simpleid.c -o simpleid
$ ./simpleid
uid=1001, gid=1001
$ ./simpleid
uid=1001, gid=1001
$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest)
$
```

Рис. 3.4: Компиляция и выполнение программы simpleid

Усложнила программу, добавив вывод действительных идентификаторов .

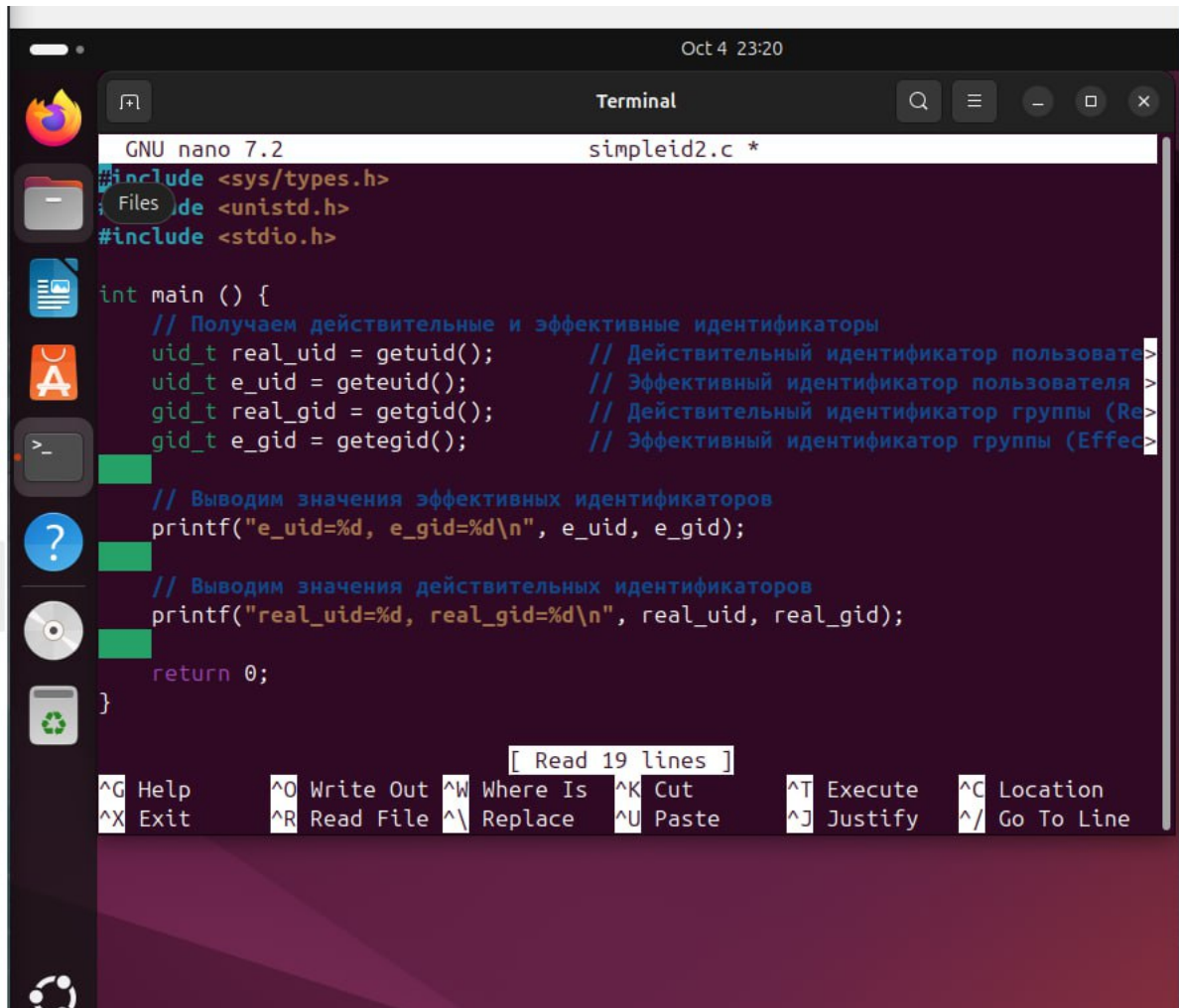


Рис. 3.5: Усложнение программы

Получившуюся программу назвала simpleid2.c.

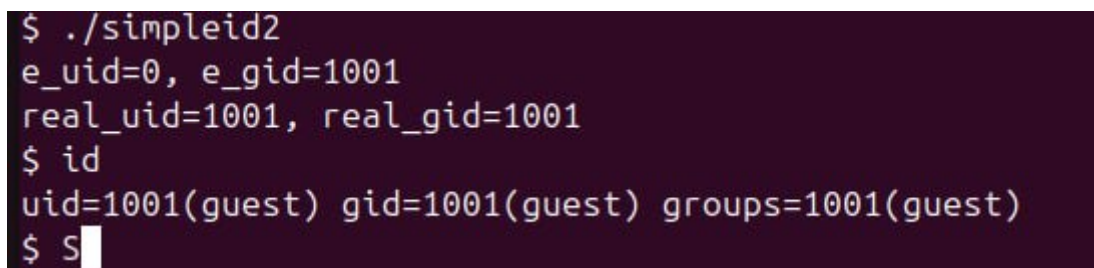


Рис. 3.6: Переименование программы в simpleid2.c

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

```

kavolchok@kavolchok-VirtualBox:~$ sudo chown root:guest/home/guest/readfile.c
[sudo] password for kavolchok:
chown: missing operand after 'root:guest/home/guest/readfile.c'
Try 'chown --help' for more information.
kavolchok@kavolchok-VirtualBox:~$ sudo chown root:guest/home/guest/readfile.c
chown: missing operand after 'root:guest/home/guest/readfile.c'
Try 'chown --help' for more information.
kavolchok@kavolchok-VirtualBox:~$ sudo chown root:guest /home/guest/readfile.c
kavolchok@kavolchok-VirtualBox:~$ "sudo chmod 700/home/guest/readfile.c
Command '"sudo' not found, did you mean:
  command 'sudo' from deb sudo (1.9.14p2-1ubuntu1)
  command 'sudo' from deb sudo-ldap (1.9.14p2-1ubuntu1)
Try: sudo apt install <deb name>
kavolchok@kavolchok-VirtualBox:~$ sudo chmod 700/home/guest/readfile.c
chmod: missing operand after '700/home/guest/readfile.c'
Try 'chmod --help' for more information.
kavolchok@kavolchok-VirtualBox:~$ sudo chmod 700 /home/guest/readfile.c
kavolchok@kavolchok-VirtualBox:~$ S

```

Рис. 3.7: Переименование программы в simpleid2.c

От имени суперпользователя выполнила команды “sudo chown root:guest /home/guest/simpleid2” и “sudo chmod u+s /home/guest/simpleid2”, затем выполнила проверку правильности установки новых атрибутов и смены владельца файла simpleid2 командой “sudo ls -l /home/guest/simpleid2”. Этими командами была произведена смена пользователя файла на root и установлен SetUID-бит.

```

kavolchok@kavolchok-VirtualBox:~$ sudo chown root:guest/home/guest/readfile.c
[sudo] password for kavolchok:
chown: missing operand after 'root:guest/home/guest/readfile.c'
Try 'chown --help' for more information.
kavolchok@kavolchok-VirtualBox:~$ sudo chown root:guest/home/guest/readfile.c
chown: missing operand after 'root:guest/home/guest/readfile.c'
Try 'chown --help' for more information.
kavolchok@kavolchok-VirtualBox:~$ sudo chown root:guest /home/guest/readfile.c
kavolchok@kavolchok-VirtualBox:~$ "sudo chmod 700/home/guest/readfile.c
Command "'sudo' not found, did you mean:
  command 'sudo' from deb sudo (1.9.14p2-1ubuntu1)
  command 'sudo' from deb sudo-ldap (1.9.14p2-1ubuntu1)
Try: sudo apt install <deb name>
kavolchok@kavolchok-VirtualBox:~$ sudo chmod 700/home/guest/readfile.c
chmod: missing operand after '700/home/guest/readfile.c'
Try 'chmod --help' for more information.
kavolchok@kavolchok-VirtualBox:~$ sudo chmod 700 /home/guest/readfile.c
kavolchok@kavolchok-VirtualBox:~$ S

```

Рис. 3.8: Установка новых атрибутов (SetUID) и смена владельца файла

Запустила программы simpleid2 и id. Теперь появились различия в uid.

```

uid=1001, gid=1001
$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest)
$ nano simpleid2.c
$ nano simpleid2.c
$ gcc simpleid2.c -o simpleid2
$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
$

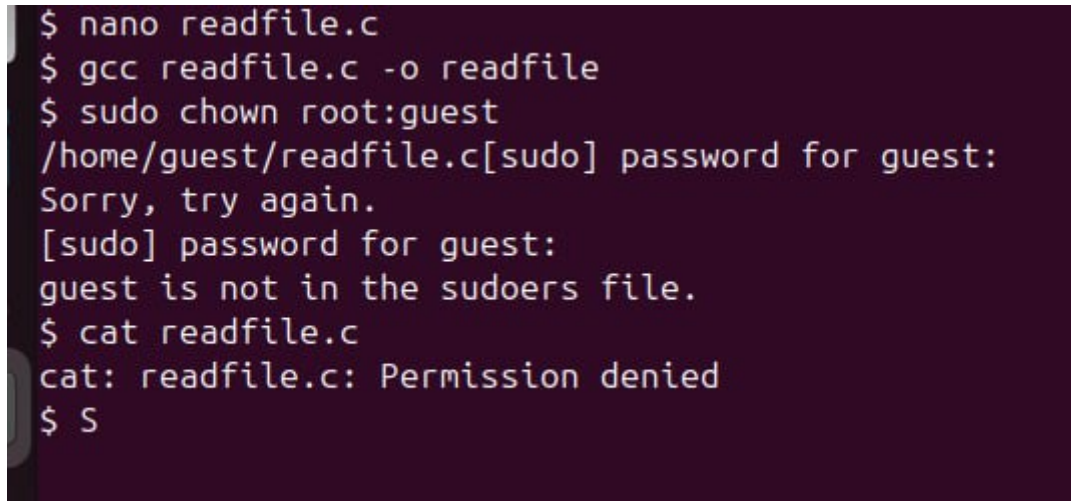
```

Рис. 3.9: Запуск simpleid2 после установки SetUID

Прodelала тоже самое относительно SetGID-бита. Также можем заметить различия с предыдущим пунктом. Далее создаем программу readfile.c.

Скомпилировала созданную программу командой "gcc readfile.c -o readfile". Сменила владельца у файла readfile.c командой "sudo chown root:guest/home/guest/readfile.c"

и поменяла права так, чтобы только суперпользователь мог прочитать его, а guest не мог, спомощью команды “sudo chmod 700 home/guest/readfile.c”. Теперь убедилась, что пользователь guest не может прочитать файл readfile.c командой “cat readfile.c”, получив отказ в доступе

A terminal window with a dark purple background and light green text. The commands and output are as follows:

```
$ nano readfile.c
$ gcc readfile.c -o readfile
$ sudo chown root:guest
/home/guest/readfile.c[sudo] password for guest:
Sorry, try again.
[sudo] password for guest:
guest is not in the sudoers file.
$ cat readfile.c
cat: readfile.c: Permission denied
$ S
```

Рис. 3.10: Смена владельца и прав доступа у файла readfile.c

Поменяла владельца у программы readfile и установила SetUID. Проверила, может ли программа readfile прочитать файл readfile.c командой “./readfile readfile.c”. Прочитать удалось. Аналогично проверила, можно ли прочитать файл /etc/shadow. Прочитать удалось.


```

$ su - guest2
Password:
$ cat /tmp/file01.txt
test
$
ec
$
Terminal
$ echo "test2" > /tmp/file01.txt$
-sh: 5: cannot create /tmp/file01.txt: Permission denied
$ echo "test2" > /tmp/file01.txt
-sh: 6: cannot create /tmp/file01.txt: Permission denied
$ ls -ld /tmp
drwxrwxrwt 18 root root 4096 Oct  5 01:18 /tmp
$ sudo echo "test2" > /tmp/file01.txt
-sh: 8: cannot create /tmp/file01.txt: Permission denied
$ echo "test2" | sudo tee /tmp/file01.txt
[sudo] password for guest2:
guest2 is not in the sudoers file.
$ cat /tmp/file01.txt
test
$ echo "test3" > /tmp/file01.txt
-sh: 11: cannot create /tmp/file01.txt: Permission denied
$ sudo rm /tmp/file01.txt
[sudo] password for guest2:
guest2 is not in the sudoers file.
$ sudo whoami
[sudo] password for guest2:

```

Рис. 3.12: Создание файла file01.txt

От имени пользователя guest2 попробовала прочитать файл командой “cat/tmp/file01.txt” - это удалось. Далее попыталась дозаписать в файл слово test2, проверить содержимое файла и записать в файл слово test3, стерев при этом всю имеющуюся в файле информацию - эти операции удалось выполнить только в случае, если еще дополнительно разрешить чтение и запись для группы пользователей командой “chmod g+rw /tmp/file01.txt”. От имени пользователя guest2 попробовала удалить файл - это не удастся ни в каком из случаев, возникает ошибка.

```
-sh: 14: cannot create /tmp/file01.txt: Permission denied
$ ls -l /tmp/file01.txt
-rw-rw-rw- 1 guest guest 5 Oct  5 01:18 /tmp/file01.txt
$ echo "test2" > /tmp/file01.txt
-sh: 16: cannot create /tmp/file01.txt: Permission denied
$ echo "test2" > /tmp/file01.txt
-sh: 17: cannot create /tmp/file01.txt: Permission denied
$ sudo chattr -i /tmp/file01.txt
[sudo] password for guest2:
guest2 is not in the sudoers file.
$ echo "test2" > /tmp/file01.txt
-sh: 19: cannot create /tmp/file01.txt: Permission denied
$ rm /tmp/file01.txt
echo "test2" > /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': Operation not permitted
$ -sh: 21: cannot create /tmp/file01.txt: Permission denied
$ su- guest
-sh: 22: su-: not found
$ su - guest
Password:
$ echo "test3" > /tmp/file01.txt
$ cat /tmp/file01.txt
test3
$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory
$
```

Рис. 3.13: Создание файла file01.txt

Повысила права до суперпользователя командой “su -” и выполнила команду, снимающую атрибут t с директории /tmp “chmod -t /tmp”. После чего покинула режим суперпользователя командой “exit”. Повторила предыдущие шаги. Теперь мне удалось удалить файл file01.txt от имени пользователя, не являющегося его владельцем.


```
kavolchok@kavolchok-VirtualBox:~$ sudo usermod -aG sudo guest2
[sudo] password for kavolchok:
kavolchok@kavolchok-VirtualBox:~$ sudo usermod -aG sudo guest2
kavolchok@kavolchok-VirtualBox:~$ chmod -t /tmp
chmod: changing permissions of '/tmp': Operation not permitted
kavolchok@kavolchok-VirtualBox:~$ /tmp "chmod -t /t
```

Рис. 3.14: Удаление атрибута t (Sticky-бита) и повторение действий

И далее овысила свои права до суперпользователя и вернула атрибут t на директорию /tmp

4 Выводы

В ходе выполнения данной лабораторной работы я изучила механизмы изменения идентификаторов, применение 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.

5 Список литературы

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