

# **Лабораторная работа №5**

**Дисциплина: Основы информационной безопасности**

Аветисян Давид Артурович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Теоретическое введение</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
<b>4</b>	<b>Выводы</b>	<b>17</b>
<b>5</b>	<b>Список литературы</b>	<b>18</b>

# List of Figures

3.1	Предварительная подготовка . . . . .	7
3.2	Команда “whereis” . . . . .	8
3.3	Вход в систему и создание программы . . . . .	8
3.4	Код программы simpleid.c . . . . .	8
3.5	Компиляция и выполнение программы simpleid . . . . .	9
3.6	Усложнение программы . . . . .	9
3.7	Компиляция и выполнение программы simpleid2 . . . . .	9
3.8	Установка новых атрибутов (SetUID) и смена владельца файла . .	10
3.9	Запуск simpleid2 после установки SetUID . . . . .	10
3.10	Запуск simpleid2 после установки SetGID . . . . .	11
3.11	Код программы readfile.c . . . . .	12
3.12	Смена владельца и прав доступа у файла readfile.c . . . . .	13
3.13	Запуск программы readfile . . . . .	13
3.14	Создание файла file01.txt . . . . .	14
3.15	Попытка выполнить действия над файлом file01.txt от имени поль- зователя guest2 . . . . .	15
3.16	Удаление атрибута t (Sticky-бита) и повторение действий . . . . .	15
3.17	Возвращение атрибута t (Sticky-бита) . . . . .	16

# List of Tables

# 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 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. Более подробно см. в [1].

## 3 Выполнение лабораторной работы

### 1 часть: Создание программы

- 1) Для начала мы убеждаемся, что компилятор gcc установлен, используя команду “gcc -v”. Затем отключаем систему запретов до очередной перезагрузки системы командой “sudo setenforce 0”, после чего команда “getenforce” выводит “Permissive” (fig. 3.1).

```
[daavetisyan@daavetisyan ~]$ gcc -v
Используются внутренние спецификации.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/x86_64-redhat-linux/11/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none
OFFLOAD_TARGET_DEFAULT=1
Целевая архитектура: x86_64-redhat-linux
Параметры конфигурации: ../configure --enable-bootstrap --enable-host-p
g/ --enable-shared --enable-threads=posix --enable-checking=release --w
--enable-initfini-array --without-isl --enable-multilib --with-linker-h
--arch_32=x86-64 --build=x86_64-redhat-linux --with-build-config=bootstr
Модель многопоточности: posix
Supported LTO compression algorithms: zlib zstd
gcc версия 11.3.1 20221121 (Red Hat 11.3.1-4) (GCC)
[daavetisyan@daavetisyan ~]$ sudo setenforce 0

Мы полагаем, что ваш системный администратор изложил вам основы
безопасности. Как правило, всё сводится к трём следующим правилам:

    №1) Уважайте частную жизнь других.
    №2) Думайте, прежде что-то вводить.
    №3) С большой властью приходит большая ответственность.

[sudo] пароль для daavetisyan:
[daavetisyan@daavetisyan ~]$ getenforce
Permissive
[daavetisyan@daavetisyan ~]$
```

Figure 3.1: Предварительная подготовка

- 2) Проверяем успешное выполнение команд “whereis gcc” и “whereis g++” (их расположение)( fig. 3.2).

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

Figure 3.2: Команда “whereis”

- 3) Входим в систему от имени пользователя guest командой “su - guest”. Создаём программу simpleid.c командой “touch simpleid.c” и открываем её в редакторе командой “gedit /home/guest/lab05/simpleid.c” (fig. 3.3).

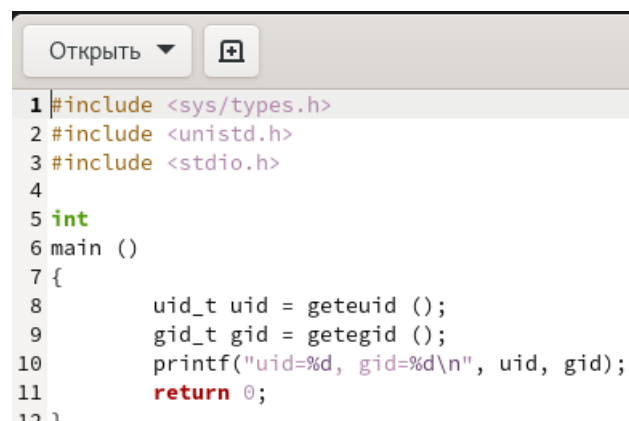
```
[daavetisyan@daavetisyan ~]$ su - guest
Пароль:
[guest@daavetisyan ~]$ ls
dir1 Видео Документы Загрузки Изображения Музыка Общедоступные 'Рабочий стол' Шаблоны
[guest@daavetisyan ~]$ mkdir lab05
[guest@daavetisyan ~]$ cd lab05/
[guest@daavetisyan lab05]$ ls
[guest@daavetisyan lab05]$ touch simpleid.c
[guest@daavetisyan lab05]$ gedit simpleid.c

(gedit:4723): dbind-WARNING **: 18:55:26.788: Couldn't register with accessibility bus: Did not receive a reply. Possible causes include: the remote application did not
reply timeout expired, or the network connection was broken.

(gedit:4723): dconf-WARNING **: 18:55:26.981: failed to commit changes to dconf: Не удалось выполнить процесс-потомок «dbus-launch» (Нет такого файла или каталога)
(gedit:4723): dconf-WARNING **: 18:55:26.911: failed to commit changes to dconf: Не удалось выполнить процесс-потомок «dbus-launch» (Нет такого файла или каталога)
(gedit:4723): dconf-WARNING **: 18:55:27.269: failed to commit changes to dconf: Не удалось выполнить процесс-потомок «dbus-launch» (Нет такого файла или каталога)
(gedit:4723): dconf-WARNING **: 18:55:27.279: failed to commit changes to dconf: Не удалось выполнить процесс-потомок «dbus-launch» (Нет такого файла или каталога)
(gedit:4723): dconf-WARNING **: 18:55:27.279: failed to commit changes to dconf: Не удалось выполнить процесс-потомок «dbus-launch» (Нет такого файла или каталога)
```

Figure 3.3: Вход в систему и создание программы

- 4) Код программы выглядит следующим образом (fig. 3.4).



```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4
5 int
6 main ()
7 {
8     uid_t uid = geteuid ();
9     gid_t gid = getegid ();
10    printf("uid=%d, gid=%d\n", uid, gid);
11    return 0;
12 }
```

Figure 3.4: Код программы simpleid.c

- 5) Скомпилируем программу и убедимся, что файл программы был создан командой “gcc simpleid.c -o simpleid”. Выполняем программу simpleid командой “./simpleid”, а затем системную программу id командой “id”. Результаты,



полученные в результате выполнения обеих команд, совпадают (uid=1001 и gid=1001) (fig. 3.5).

```
[guest@daavetisyan lab05]$ gcc simpleid.c -o simpleid
[guest@daavetisyan lab05]$ ./simpleid
uid=1001, gid=1001
[guest@daavetisyan lab05]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@daavetisyan lab05]$
```

Figure 3.5: Компиляция и выполнение программы simpleid

- 6) Усложняем программу, добавив вывод действительных идентификаторов, новый файл назовём simpleid2.c (fig. 3.6).

```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4
5 int
6 main ()
7 {
8     uid_t e_uid = geteuid ();
9     gid_t e_gid = getegid ();
10
11     uid_t real_uid = getuid ();
12     gid_t real_gid = getgid ();
13
14     printf("uid=%d, gid=%d\n", uid, gid);
15     printf("real_uid=%d, real_gid=%d\n", real_uid, real_gid);
16     return 0;
17 }
```

Figure 3.6: Усложнение программы

- 7) Скомпилируем и запустим simpleid2.c командами “gcc simpleid2.c -o simpleid2” и “./simpleid2” (fig. 3.7).

```
[guest@daavetisyan lab05]$ gcc simpleid2.c -o simpleid2
[guest@daavetisyan lab05]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@daavetisyan lab05]$
```

Figure 3.7: Компиляция и выполнение программы simpleid2

- 8) От имени суперпользователя выполняем команды “sudo chown root:guest /home/guest/lab05/simpleid2” и “sudo chmod u+s /home/guest/lab05/simpleid2”, затем выполняем проверку правильности установки новых атрибутов и смены владельца файла simpleid2 командой “sudo ls -l /home/guest/lab05/simpleid2” (fig. 3.8). Этими командами была произведена смена пользователя файла на root и установлен SetUID-бит.

```
[daavetisyan@daavetisyan ~]$ sudo chown root: /home/guest/lab05/simpleid2
[sudo] пароль для daavetisyan:
[daavetisyan@daavetisyan ~]$ sudo chmod u+s /home/guest/lab05/simpleid2
[daavetisyan@daavetisyan ~]$ sudo ls -l /home/guest/lab05/simpleid2
-rwsr-xr-x. 1 root root 26064 окт  9 19:08 /home/guest/lab05/simpleid2
[daavetisyan@daavetisyan ~]$
```

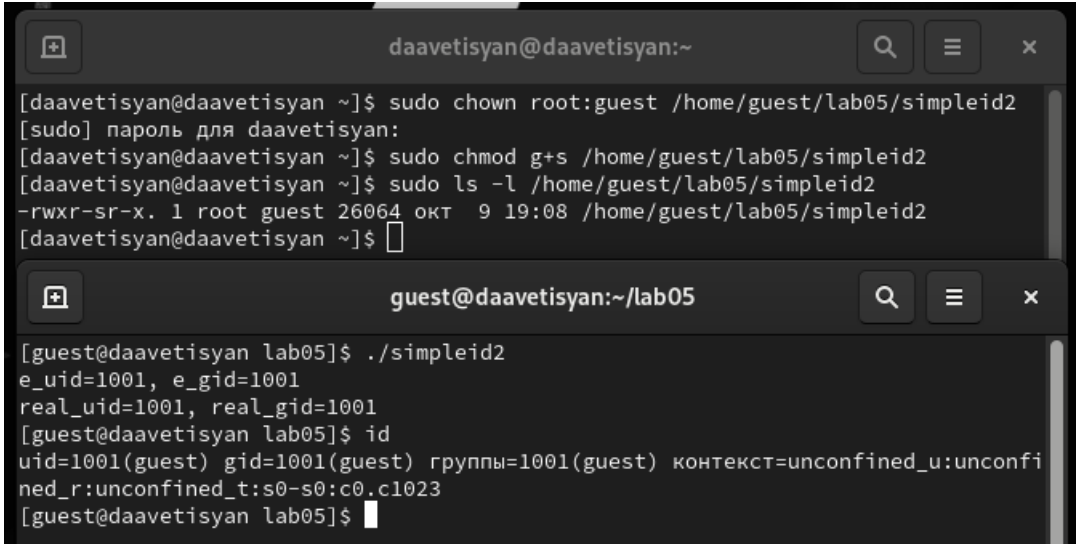
Figure 3.8: Установка новых атрибутов (SetUID) и смена владельца файла

- 9) Запускаем программы simpleid2 и id. Теперь появились различия в uid (fig. 3.9).

```
[guest@daavetisyan lab05]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@daavetisyan lab05]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@daavetisyan lab05]$
```

Figure 3.9: Запуск simpleid2 после установки SetUID

- 10) Проделаем тоже самое относительно SetGID-бита. Также можем заметить различия с предыдущим пунктом (fig. 3.10).



The image shows two terminal windows. The top window is titled 'daavetisyan@daavetisyan:~' and shows the following commands and output:

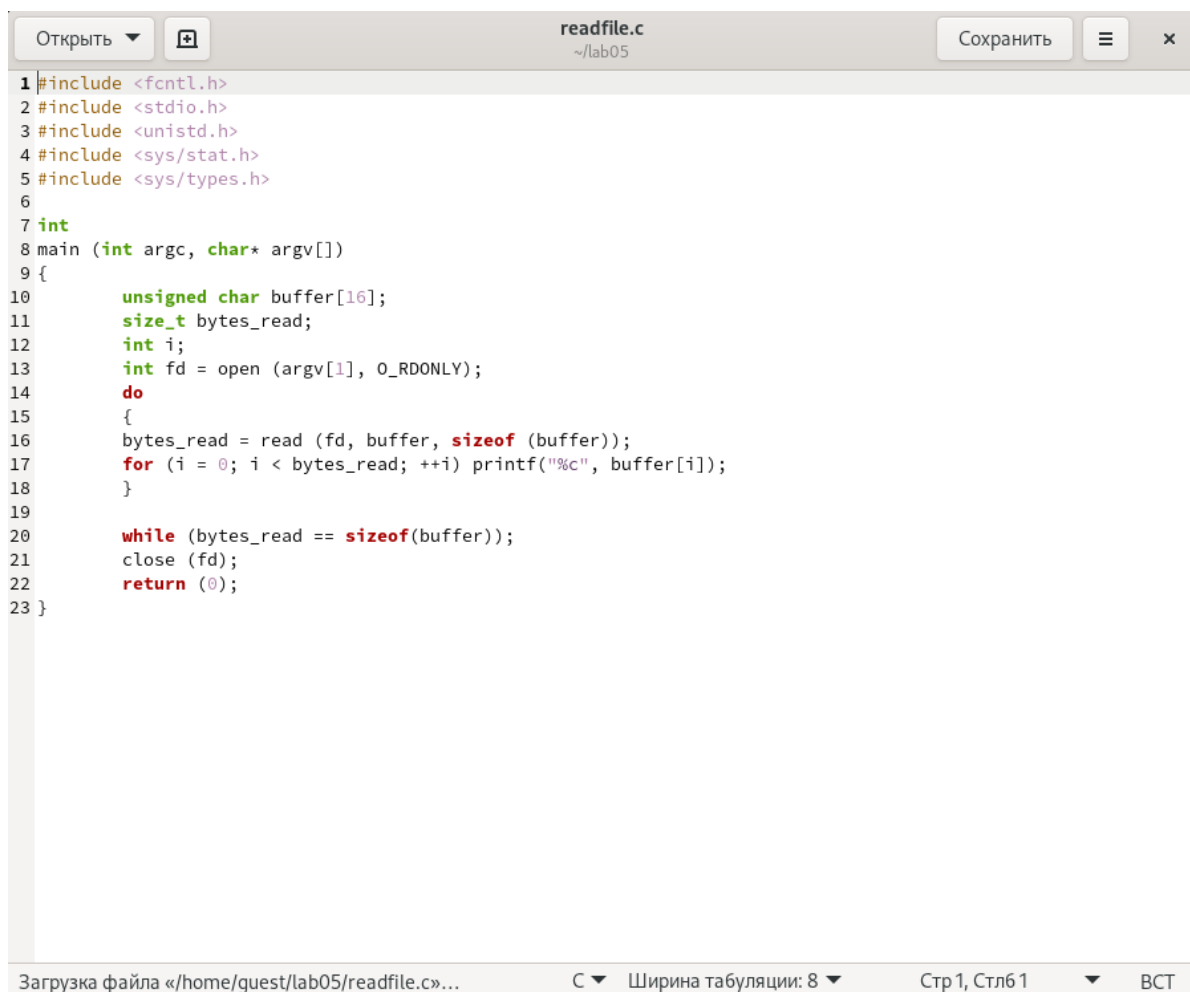
```
[daavetisyan@daavetisyan ~]$ sudo chown root:guest /home/guest/lab05/simpleid2
[sudo] пароль для daavetisyan:
[daavetisyan@daavetisyan ~]$ sudo chmod g+s /home/guest/lab05/simpleid2
[daavetisyan@daavetisyan ~]$ sudo ls -l /home/guest/lab05/simpleid2
-rwxr-sr-x. 1 root guest 26064 окт  9 19:08 /home/guest/lab05/simpleid2
[daavetisyan@daavetisyan ~]$
```

The bottom window is titled 'guest@daavetisyan:~/lab05' and shows the following commands and output:

```
[guest@daavetisyan lab05]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@daavetisyan lab05]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi
ned_r:unconfined_t:s0-s0:c0.c1023
[guest@daavetisyan lab05]$
```

Figure 3.10: Запуск simpleid2 после установки SetGID

11) Создаем программу readfile.c (fig. 3.11).

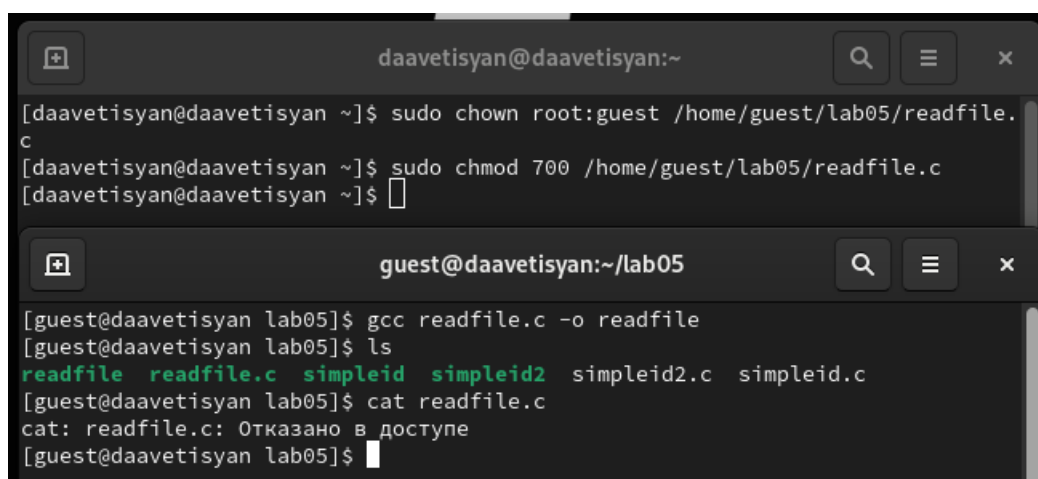


```
1 #include <fcntl.h>
2 #include <stdio.h>
3 #include <unistd.h>
4 #include <sys/stat.h>
5 #include <sys/types.h>
6
7 int
8 main (int argc, char* argv[])
9 {
10     unsigned char buffer[16];
11     size_t bytes_read;
12     int i;
13     int fd = open (argv[1], O_RDONLY);
14     do
15     {
16         bytes_read = read (fd, buffer, sizeof (buffer));
17         for (i = 0; i < bytes_read; ++i) printf("%c", buffer[i]);
18     }
19
20     while (bytes_read == sizeof(buffer));
21     close (fd);
22     return (0);
23 }
```

Загрузка файла «/home/guest/lab05/readfile.c»... С    Ширина табуляции: 8    Стр 1, Стлб 1    ВСТ

Figure 3.11: Код программы readfile.c

- 12) Скомпилируем созданную программу командой “gcc readfile.c -o readfile”.  
Сменим владельца у файла readfile.c командой “sudo chown root:guest /home/guest/readfile.c” и поменяем права так, чтобы только суперпользователь мог прочитать его, а guest не мог, с помощью команды “sudo chmod 700 /home/guest/readfile.c”. Убеждаемся, что пользователь guest не может прочитать файл readfile.c командой “cat readfile.c”, получив отказ в доступе (fig. 3.12).

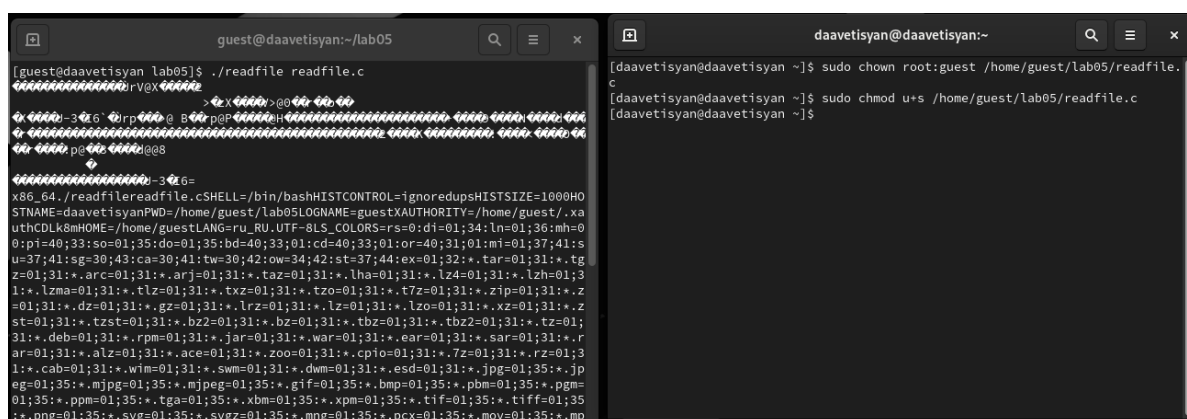


```
daavetisyan@daavetisyan:~$ sudo chown root:guest /home/guest/lab05/readfile.c
[daavetisyan@daavetisyan ~]$ sudo chmod 700 /home/guest/lab05/readfile.c
[daavetisyan@daavetisyan ~]$

guest@daavetisyan:~/lab05$ gcc readfile.c -o readfile
guest@daavetisyan lab05$ ls
readfile readfile.c simpleid simpleid2 simpleid2.c simpleid.c
guest@daavetisyan lab05$ cat readfile.c
cat: readfile.c: Отказано в доступе
guest@daavetisyan lab05$
```

Figure 3.12: Смена владельца и прав доступа у файла readfile.c

13) Поменяем владельца у программы readfile и установим SetUID. Проверим, может ли программа readfile прочитать файл readfile.c командой “./readfile readfile.c”. Прочитать удалось. Аналогично проверяем, можно ли прочитать файл /etc/shadow. Прочитать удалось (fig. 3.13).



```
guest@daavetisyan:~/lab05$ ./readfile readfile.c
x86_64:./readfilereadfile.cSHELL=/bin/bashHISTCONTROL=ignoredupsHISTSIZE=100H0
STNAME=daavetisyanPWD=/home/guest/lab05LOGNAME=guestXAUTHORITY=/home/guest/.x
uthCDLk8mHOME=/home/guestLANG=ru_RU.UTF-8LS_COLORS=rs=0:d1=01;34:ln=01;36:mh=0
0:p1=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=01;37;41:s
u=37;41:sg=30;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tg
z=01;31:*.arc=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;3
1:*.lzma=01;31:*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z
=01;31:*.dz=01;31:*.gz=01;31:*.lrz=01;31:*.lzo=01;31:*.xz=01;31:*.z
st=01;31:*.tzt=01;31:*.b2z=01;31:*.bz2=01;31:*.tbz=01;31:*.tbz2=01;31:*.tz=01;
31:*.deb=01;31:*.rpm=01;31:*.jar=01;31:*.war=01;31:*.ear=01;31:*.rar=01;31:*.r
ar=01;31:*.alz=01;31:*.aces=01;31:*.zoo=01;31:*.cpio=01;31:*.7z=01;31:*.rz=01;3
1:*.cab=01;31:*.wim=01;31:*.swm=01;31:*.dwm=01;31:*.esd=01;31:*.jpg=01;35:*.jp
eg=01;35:*.mjpg=01;35:*.mjpeg=01;35:*.gif=01;35:*.bmp=01;35:*.pbm=01;35:*.pgm
=01;35:*.ppm=01;35:*.tga=01;35:*.xbm=01;35:*.xpm=01;35:*.tif=01;35:*.tiff=01;35
:*.png=01;35:*.svg=01;35:*.svgz=01;35:*.mng=01;35:*.pcx=01;35:*.mov=01;35:*.mp

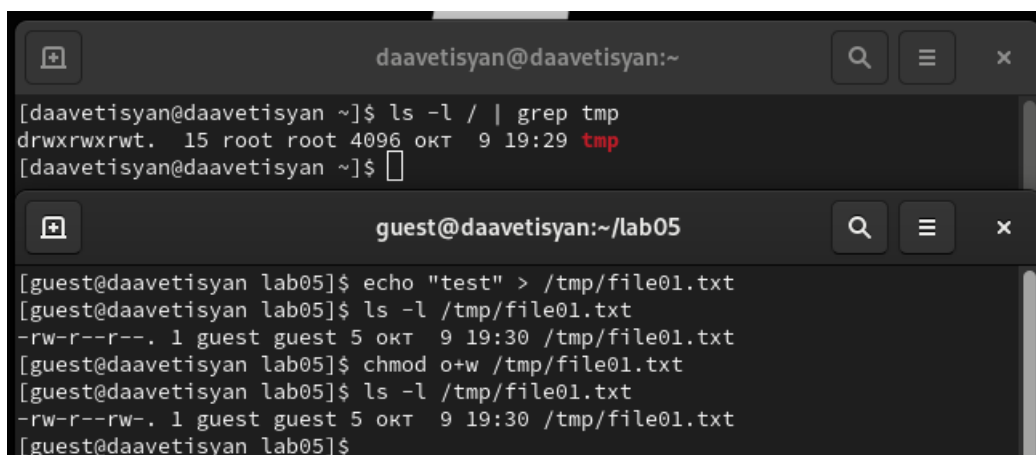
daavetisyan@daavetisyan:~$ sudo chown root:guest /home/guest/lab05/readfile.c
[daavetisyan@daavetisyan ~]$ sudo chmod u+s /home/guest/lab05/readfile.c
[daavetisyan@daavetisyan ~]$
```

Figure 3.13: Запуск программы readfile

## 2 часть: Исследование Sticky-бита

1) Командой “ls -l / | grep tmp” убеждаемся, что атрибут Sticky на директории /tmp установлен. От имени пользователя guest создаём файл file01.txt в директории /tmp со словом test командой “echo test” > /tmp/file01.txt”. Просматриваем атрибуты у только что созданного файла и разрешаем чтение

и запись для категории пользователей “все остальные” командами “ls -l /tmp/file01.txt” и “chmod o+rw /tmp/file01.txt” (fig. 3.14).



```
daavetisyan@daavetisyan:~  
[daavetisyan@daavetisyan ~]$ ls -l / | grep tmp  
drwxrwxrwt. 15 root root 4096 окт  9 19:29 tmp  
[daavetisyan@daavetisyan ~]$  
  
guest@daavetisyan:~/lab05  
[guest@daavetisyan lab05]$ echo "test" > /tmp/file01.txt  
[guest@daavetisyan lab05]$ ls -l /tmp/file01.txt  
-rw-r--r--. 1 guest guest 5 окт  9 19:30 /tmp/file01.txt  
[guest@daavetisyan lab05]$ chmod o+w /tmp/file01.txt  
[guest@daavetisyan lab05]$ ls -l /tmp/file01.txt  
-rw-r--rw-. 1 guest guest 5 окт  9 19:30 /tmp/file01.txt  
[guest@daavetisyan lab05]$
```

Figure 3.14: Создание файла file01.txt

- 2) От имени пользователя guest2 пробуем прочитать файл командой “cat /tmp/file01.txt” - это удалось. Далее пытаемся дозаписать в файл слово test2, проверить содержимое файла и записать в файл слово test3, стерева 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 пробуем удалить файл - это не удастся ни в каком из случаев, возникает ошибка (fig. 3.15).

```
guest2@daavetisyan:~$ cat /tmp/file01.txt
test
[guest2@daavetisyan ~]$ echo "test2" > /tmp/file01.txt
-bash: /tmp/file01.txt: Отказано в доступе
[guest2@daavetisyan ~]$ cat /tmp/file01.txt
test
[guest2@daavetisyan ~]$ echo "test3" >> /tmp/file01.txt
-bash: /tmp/file01.txt: Отказано в доступе
[guest2@daavetisyan ~]$ cat /tmp/file01.txt
test
[guest2@daavetisyan ~]$ rm -R /tmp/file01.txt
rm: удалить защищенный от записи обычный файл '/tmp/file01.txt'? n
[guest2@daavetisyan ~]$ rm /tmp/file01.txt
rm: удалить защищенный от записи обычный файл '/tmp/file01.txt'? y
rm: невозможно удалить '/tmp/file01.txt': Операция не позволена
[guest2@daavetisyan ~]$ echo "test3" > /tmp/file01.txt
[guest2@daavetisyan ~]$ cat /tmp/file01.txt
test3
[guest2@daavetisyan ~]$
```

```
guest@daavetisyan:~$ chmod g+rw /tmp/file01.txt
[guest@daavetisyan ~]$
```

Figure 3.15: Попытка выполнить действия над файлом file01.txt от имени пользователя guest2

- 3) Повышаем права до суперпользователя командой “su -” и выполняем команду, снимающую атрибут t с директории /tmp “chmod -t /tmp”. После чего покидаем режим суперпользователя командой “exit”. Повторяем предыдущие шаги. Теперь нам удаётся удалить файл file01.txt от имени пользователя, не являющегося его владельцем (fig. 3.16).

```
[guest2@daavetisyan ~]$ su -
Пароль:
[root@daavetisyan ~]# chmod -t /tmp/
[root@daavetisyan ~]# exit
выход
[guest2@daavetisyan ~]$ ls -l / | grep tmp
drwxrwxrwx. 17 root root 4096 окт 9 19:37 tmp
[guest2@daavetisyan ~]$ cat /tmp/file01.txt
test3
[guest2@daavetisyan ~]$ echo "test2" >> /tmp/file01.txt
[guest2@daavetisyan ~]$ cat /tmp/file01.txt
test3
test2
[guest2@daavetisyan ~]$ echo "test3" > /tmp/file01.txt
[guest2@daavetisyan ~]$ cat /tmp/file01.txt
test3
[guest2@daavetisyan ~]$ rm /tmp/file01.txt
[guest2@daavetisyan ~]$ ls -l /tmp/
итого 0
drwx-----, 3 root root 17 окт 9 18:47 systemd-private-79bc9477a8874b5bbc1cdd2112a9c82a-chronyd.service-SfiQzU
drwx-----, 3 root root 17 окт 9 18:47 systemd-private-79bc9477a8874b5bbc1cdd2112a9c82a-colord.service-v222m3
drwx-----, 3 root root 17 окт 9 18:47 systemd-private-79bc9477a8874b5bbc1cdd2112a9c82a-dbus-broker.service-gruNJF
drwx-----, 3 root root 17 окт 9 18:48 systemd-private-79bc9477a8874b5bbc1cdd2112a9c82a-fwupd.service-HVUTWA
drwx-----, 3 root root 17 окт 9 18:52 systemd-private-79bc9477a8874b5bbc1cdd2112a9c82a-geoclue.service-MwUGnk
drwx-----, 3 root root 17 окт 9 18:47 systemd-private-79bc9477a8874b5bbc1cdd2112a9c82a-ModemManager.service-qZH5dV
drwx-----, 3 root root 17 окт 9 18:47 systemd-private-79bc9477a8874b5bbc1cdd2112a9c82a-power-profiles-daemon.service-IB0R5z
drwx-----, 3 root root 17 окт 9 18:47 systemd-private-79bc9477a8874b5bbc1cdd2112a9c82a-rtkit-daemon.service-D0hzzC
drwx-----, 3 root root 17 окт 9 18:47 systemd-private-79bc9477a8874b5bbc1cdd2112a9c82a-switcheroo-control.service-Gl4s3d
drwx-----, 3 root root 17 окт 9 18:47 systemd-private-79bc9477a8874b5bbc1cdd2112a9c82a-systemd-logind.service-Cu3kjg
drwx-----, 3 root root 17 окт 9 18:47 systemd-private-79bc9477a8874b5bbc1cdd2112a9c82a-upower.service-GPtbrt
[guest2@daavetisyan ~]$
```

Figure 3.16: Удаление атрибута t (Sticky-бита) и повторение действий

- 4) Повышаем свои права до суперпользователя и возвращаем атрибут t на директорию /tmp (fig. 3.17).

```
[guest2@daavetisyan ~]$ su -  
Пароль:  
[root@daavetisyan ~]# chmod +t /tmp/  
[root@daavetisyan ~]# exit  
выход  
[guest2@daavetisyan ~]$ ls -l / | grep tmp  
drwxrwxrwt. 17 root root 4096 окт  9 19:39 tmp  
[guest2@daavetisyan ~]$
```

Figure 3.17: Возвращение атрибута t (Sticky-бита)



## 4 Выводы

- В ходе выполнения данной лабораторной работы я изучил механизмы изменения идентификаторов, применение SetUID- и Sticky-битов. Получил практические навыки работы в консоли с дополнительными атрибутами. Рассмотрел 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/>.