

# Отчёт по лабораторной работе №5

---

Дарижапов Тимур Андреевич

30 Сентября 2023

РУДН, Москва, Россия

## Отчет по лабораторной работе №5

---

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.

## 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.

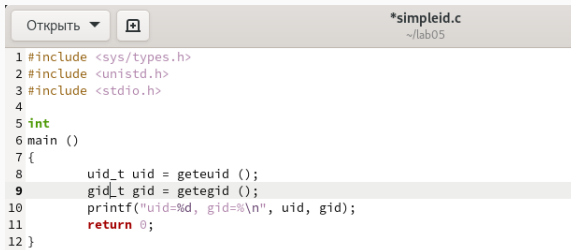
## 1 часть: Создание программы

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”.

```
[tadarizhapov@tadarizhapov ~]$ 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-pie --enable-host-bind-now
--enable-languages=c,c++,fortran,lto --prefix=/usr --mandir=/usr/share/man --infodir=/usr/share
/info --with-bugurl=https://bugs.rockylinux.org/ --enable-shared --enable-threads=posix --enable
-checking=release --with-system-zlib --enable-_cxa_atexit --disable-libunwind-exceptions --enab
le-gnu-unique-object --enable-linker-build-id --with-gcc-major-version-only --enable-plugin --en
able-initfini-array --without-isl --enable-multilib --with-linker-hash-style=gnu --enable-offloa
d-targets=nvptx-none --without-cuda-driver --enable-gnu-indirect-function --enable-cet --with-tu
ne-generic --with-arch_64=x86-64-v2 --with-arch_32=x86-64 --build=x86_64-redhat-linux --with-bui
ld-config-bootstrap-lto --enable-link-serialization=1
Модель многопоточности: posix
Supported LTO compression algorithms: zlib zstd
gcc версия 11.3.1 20221121 (Red Hat 11.3.1-4) (GCC)
[tadarizhapov@tadarizhapov ~]$ sudo setenforce 0
[sudo] пароль для tadarizhapov:
[tadarizhapov@tadarizhapov ~]$ getenforce
Permissive
[tadarizhapov@tadarizhapov ~]$
```

Рис. 1: Рисунок 1

Код программы выглядит следующим образом.



```
*simpleid.c
~/lab05

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 }
```

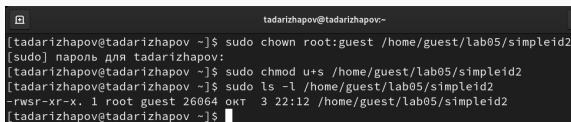
Рис. 2: Рисунок 2

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).

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

Рис. 3: Рисунок 3

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”(Рисунок 3.8). Этими командами была произведена смена пользователя файла на root и установлен SetUID-бит.



```
tadarizhapov@tadarizhapov:~  
[tadarizhapov@tadarizhapov ~]$ sudo chown root:guest /home/guest/lab05/simpleid2  
[sudo] пароль для tadarizhapov:  
[tadarizhapov@tadarizhapov ~]$ sudo chmod u+s /home/guest/lab05/simpleid2  
[tadarizhapov@tadarizhapov ~]$ sudo ls -l /home/guest/lab05/simpleid2  
-rwsr-xr-x. 1 root guest 26064 окт  3 22:12 /home/guest/lab05/simpleid2  
[tadarizhapov@tadarizhapov ~]$
```

Рис. 4: Рисунок 4

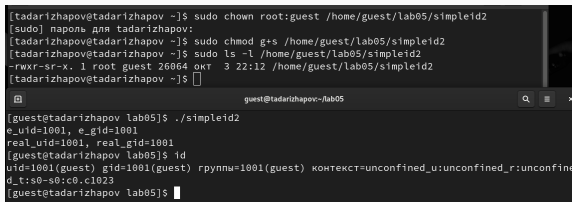


Запускаем программы `simpleid2` и `id`. Теперь появились различия в `uid`.

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

Рис. 5: Рисунок 5

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



```
[tadarizhapov@tadarizhapov ~]$ sudo chown root:guest /home/guest/lab05/simpleid2
[sudo] пароль для tadarizhapov:
[tadarizhapov@tadarizhapov ~]$ sudo chmod g+s /home/guest/lab05/simpleid2
[tadarizhapov@tadarizhapov ~]$ sudo ls -l /home/guest/lab05/simpleid2
-rwxr-sr-x. 1 root guest 26064 окт  3 22:12 /home/guest/lab05/simpleid2
[tadarizhapov@tadarizhapov ~]$
```

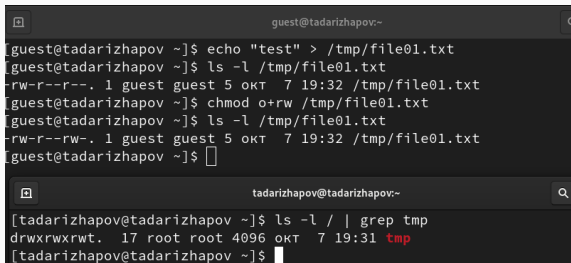
  

```
guest@tadarizhapov:~/lab05
[guest@tadarizhapov lab05]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@tadarizhapov lab05]$ id
uid=1001(guest) gid=1001(guest) rpnny=1001(guest) контекст=unconfined_u:unconfined_r:unconfine
d_t:s0-s0:c0.c1023
[guest@tadarizhapov lab05]$
```

Рис. 6: Рисунок 6

## 2 часть: Исследование Sticky-бита

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”.



```
guest@tadarizhapov:~  
[guest@tadarizhapov ~]$ echo "test" > /tmp/file01.txt  
[guest@tadarizhapov ~]$ ls -l /tmp/file01.txt  
-rw-r--r--. 1 guest guest 5 окт  7 19:32 /tmp/file01.txt  
[guest@tadarizhapov ~]$ chmod o+rw /tmp/file01.txt  
[guest@tadarizhapov ~]$ ls -l /tmp/file01.txt  
-rw-r--rw-. 1 guest guest 5 окт  7 19:32 /tmp/file01.txt  
[guest@tadarizhapov ~]$  
  
tadarizhapov@tadarizhapov:~  
[tadarizhapov@tadarizhapov ~]$ ls -l / | grep tmp  
drwxrwxrwt. 17 root root 4096 окт  7 19:31 tmp  
[tadarizhapov@tadarizhapov ~]$
```

Рис. 7: Рисунок 7

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, стеревав 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.

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

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.

```
[guest2@tadarizhapov ~]$ su -
Пароль:
[root@tadarizhapov ~]# chmod -t /tmp
[root@tadarizhapov ~]# exit
выход
[guest2@tadarizhapov ~]$ ls -l / | grep tmp
drwxrwxrwx. 17 root root 4096 окт 7 19:40 tmp
[guest2@tadarizhapov ~]$ cat /tmp/file01.txt
test3
[guest2@tadarizhapov ~]$ echo "test2" >> /tmp/file01.txt
[guest2@tadarizhapov ~]$ cat /tmp/file01.txt
test3
test2
[guest2@tadarizhapov ~]$ echo "test3" > /tmp/file01.txt
[guest2@tadarizhapov ~]$ cat /tmp/file01.txt
test3
[guest2@tadarizhapov ~]$ rm /tmp/file01.txt
[guest2@tadarizhapov ~]$ ls -l /tmp
итого 0
srwxrwxrwx. 1 gdm          gdm          0 окт 3 22:53 dbus-N00DK7CLCV
srwxrwxrwx. 1 gdm          gdm          0 сен 30 19:05 dbus-yuH1BbivKT
```

Рис. 9: Рисунок 9

- 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.