# Отчёт по лабораторной работе

**Лабораторная работа № 5**

Живцова Анна

# Содержание

# Список иллюстраций

# Список таблиц

# 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

Каждый файл или каталог имеет права доступа, обозначаемые комбинацией букв латинского (обозначает разрешение) алфавита и знаков –(обозначает отсутствие разрешения). Для файла: r — разрешено чтение, w — разрешена запись, x — разрешено выполнение, для каталога: r — разрешён просмотр списка входящих файлов, w — разрешены создание и удаление файлов, x — разрешён доступ в каталог и есть возможность сделать его текущим, - — право доступа отсутствует. В сведениях о файле или каталоге указываются: – тип файла (символ (-) обозначает файл, а символ (d) — каталог);
– права для владельца файла;
– права для членов группы;
– права для всех остальных [1].

В качестве дополнительных атрибутов могут выступать SUID, GUID и sticky биты. Разберем их подробнее.

Бит SUID устанавливается на исполняемые файлы. После установки данного бита программа исполняется с правами доступа и привилегиями пользователя, который владеет соответствующим бинарным файлом. В том случае, если владельцем бинарного файла является пользователь root, причем для файла установлен бит SUID, любой исполняющий данный файл пользователь будет иметь такие же права доступа, как и пользователь root.

При установке бита GUID для исполняемых файлов будет достигаться такой же эффект, как и в случае установки бита SUID, за тем исключением, что вместо прав доступа пользователя, владеющего файлом, в процессе исполнения файла

устанавливаются права доступа группы пользователей, владеющей файлом. Бит GUID используется главным образом по отношению к директориям. В том случае, если для директории устанавливается бит GUID, создаваемые в директории файлы наследуют идентификатор группы пользователей, владеющей директорией. Данный механизм очень полезен в случае работы группы пользователей с файлами из одной и той же директории.

В директориях, для которых установлен бит sticky, активируется дополнительный слой защиты созданных файлов. В обычных условиях при доступе множества пользователей к содержимому директории каждый пользователь имеет возможность удаления файлов другого пользователя. (Это справедливо даже для того случая, когда пользователи не имеют прав на запись содержимого этих файлов!) В случае установки бита sticky для директории файлы могут удаляться лишь теми пользователями, которые владеют ими [2] .

# 3 Выполнение лабораторной работы

1. Создали простую программу на языке C для чтения uid и gid. Проверили ее работу, сравнив с системной утилитой id (см. рис. 3.1).

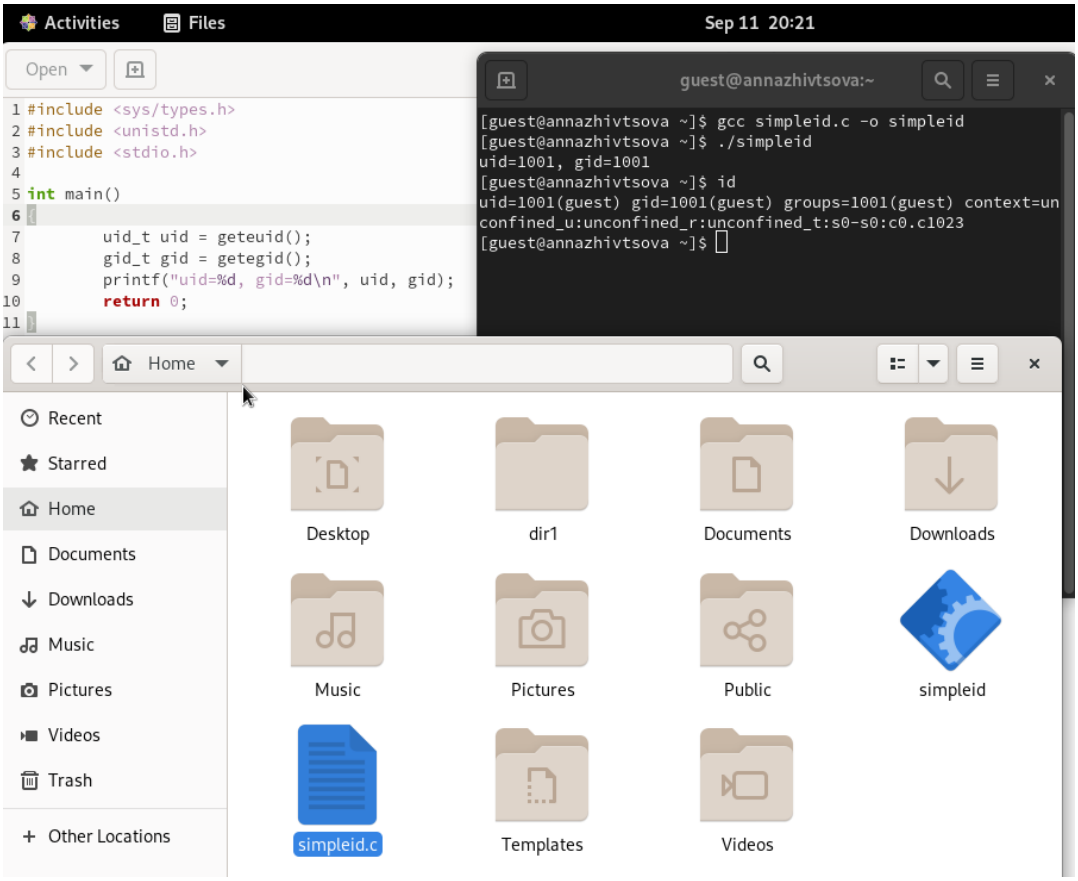


Рис. 3.1: Программа для чтения uid и gid

2. Усложнили программу, добавив вывод действительных идентификаторов. Командой `chown root:guest /home/guest/simpleid2` передали

права обладания файлов суперпользователю. Командами `chmod u+s /home/guest/simpleid2` и `chmod g+s /home/guest/simpleid2` присвоили файлу SUID и GUID биты. Проверили функционирование. И правда, при выполнении с правами суперпользователя uid указывается суперпользователя (см. рис. 3.2). Так как мы не меняли группу файла, изменений в программе при выставлении GUID не наблюдается.

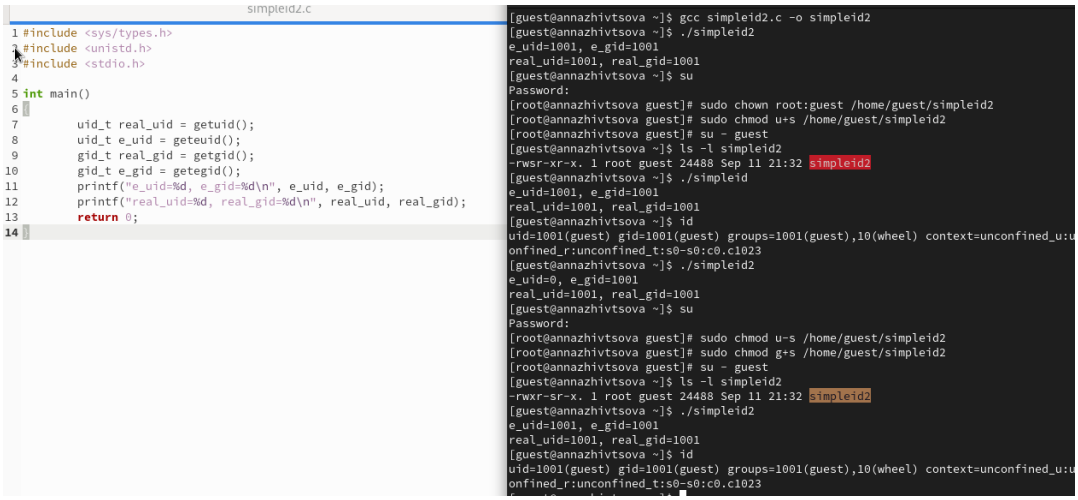


Рис. 3.2: Проверка действительных идентификаторов при выставленных SUID и GUID битах

3. Создали программу для чтения файлов. Установили на текст программы права доступа 000. Установили владельцем исполняемого файла суперпользователя. Поставили SUID бит. Запустили исполняемый файл от обычного пользователя. Исполняемый файл смог прочесть текст программы, у которого отсутствует разрешение для чтения (см. рис. 3.3).

```
[guest@annazhivtsova ~]$ gcc readfile.c -o readfile
[guest@annazhivtsova ~]$ chmod 000 readfile.c
[guest@annazhivtsova ~]$ su
Password:
[root@annazhivtsova guest]# sudo cat readfile.c
#include <sys/types.h>
#include <unistd.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdio.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;
}
[root@annazhivtsova guest]# su - guest
[guest@annazhivtsova ~]$ cat readfile.c
cat: readfile.c: Permission denied
[guest@annazhivtsova ~]$ su
Password:
[root@annazhivtsova guest]# sudo chown root:guest /home/guest/readfile
[root@annazhivtsova guest]# sudo chmod u+s /home/guest/readfile
[root@annazhivtsova guest]# su - guest
[guest@annazhivtsova ~]$ ./readfile readfile.c
#include <sys/types.h>
#include <unistd.h>
#include <sys/stat.h>
#include <fcntl.h>
```

Рис. 3.3: Проверка функциональности SUID бита на примере программы для чтения файлов и запрещенного к просмотру текстового документа

4. Протестировали действие Sticky-бита, установленного в каталоге /tmp. Убедились, что удаление файла доступно только создателю (см. рис. 3.4).

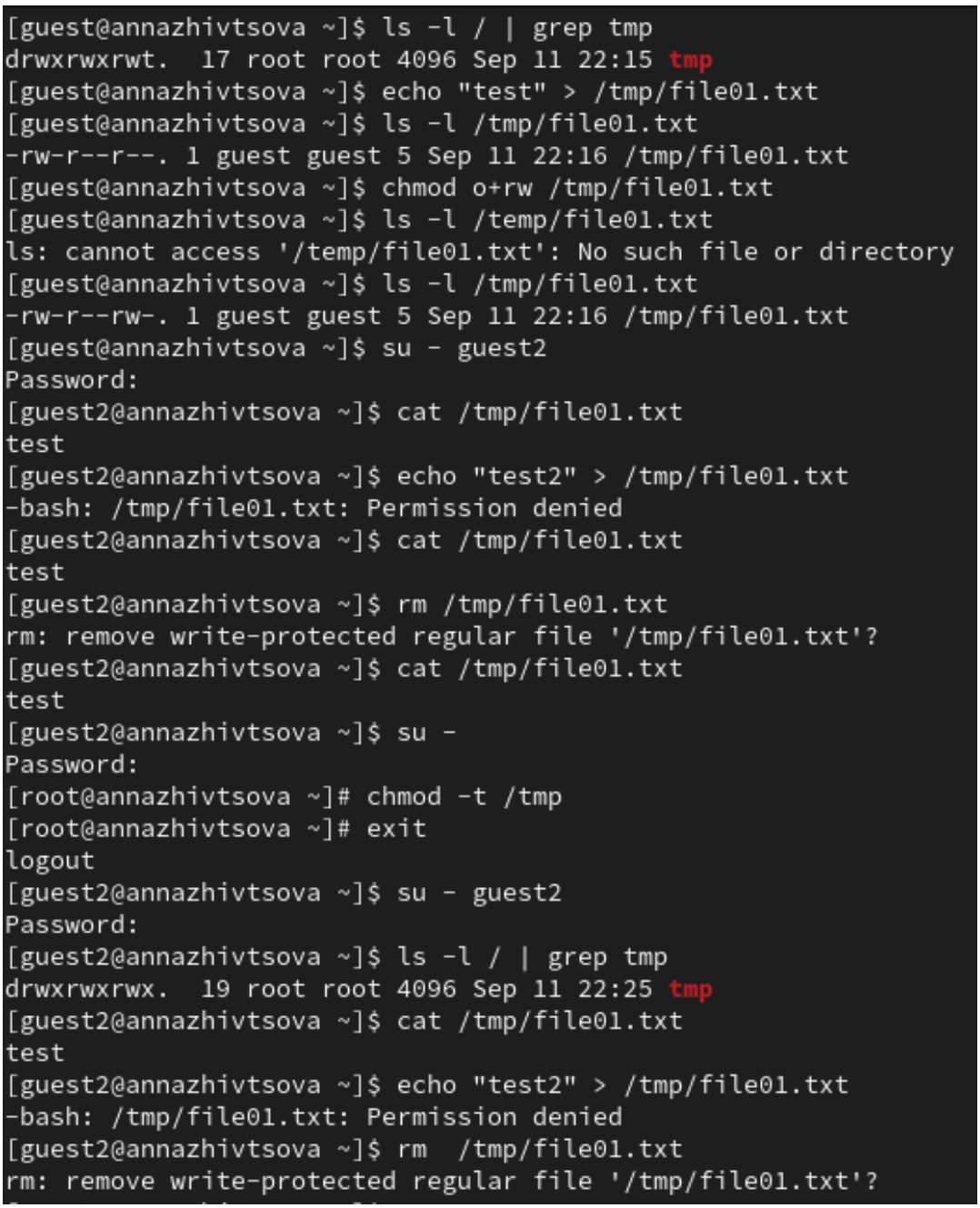


Рис. 3.4: Проверка функциональности Sticky-бита на примере файла, созданного в каталоге /tmp

# 4 Выводы

Изучили теорию механизмов изменения идентификаторов, применения SetUID, SetGID и Sticky-битов. Отработали на практике с помощью консоли влияние дополнительных атрибутов. Рассмотрели механизм смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

# Список литературы

1. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. 2-е изд. БХВ-Петербург, 2010. 656 с.
2. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.