Отчёт по лабораторной работе №5

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Жижченко Валерия Викторовна

Содержание

# Цель работы

Изучение механизмов изменения идентификаторов, применения *SetUID-битов* 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.

# Выполнение лабораторной работы

## Создание программы

1. Входим в систему от имени пользователя *guest*.
2. Создаем программу *simpleid.c*:

#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;  
}

1. Скомплилируем программу и убедимся, что файл программы создан:

$ gcc simpleid.c -o simpleid

1. Выполняем программу *simpleid*:

$ ./simpleid

1. Выполняем системную программу *id* и сравниваем полученный результат с данными предыдущего пункта задания:

$ id

1. Усложняем программу, добавив вывод действительных идентификаторов. Получившуюся программу назовем *simpleid2.c*:

#include <sys/types.h>  
#include <unistd.h>  
#include <stdio.h>  
  
int main () {  
 uid\_t real\_uid = getuid();  
 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;  
}

1. Скомпилируем и запустим *simpleid2.c*:

$ gcc simpleid2.c -o simpleid2  
$ ./simpleid2

1. От имени суперпользователя выполним следующие команды:

$ sudo chown root:guest /home/guest/simpleid2  
$ sudo chmod u+s /home/guest/simpleid2

1. Выполним проверку правильности установки новых атрибутов и смены владельца файла *simpleid2*:

$ ls -l simpleid2

1. Запустим *simpleid2* и *id*:

$ ./simpleid2  
$ id

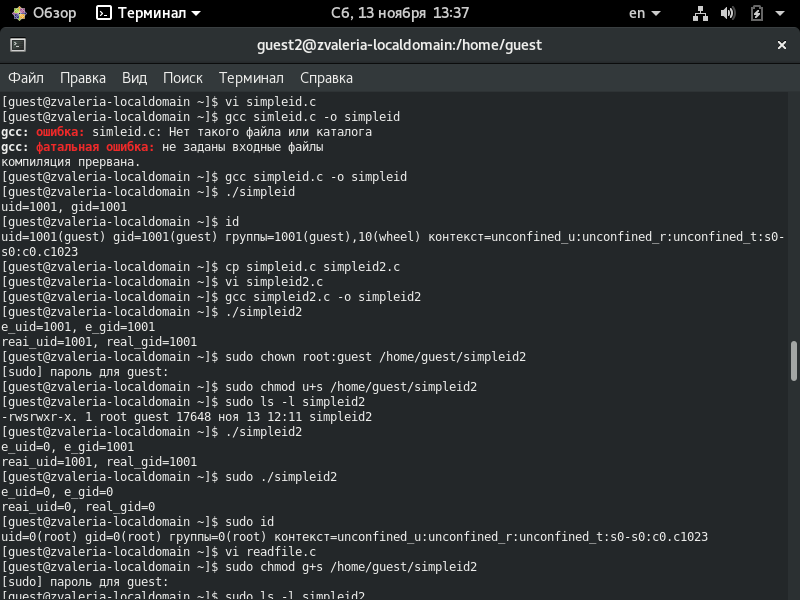


Figure 1: Операции с *SetUID-битом*

1. Проделаем тоже самое относительно *SetGID-бита*.

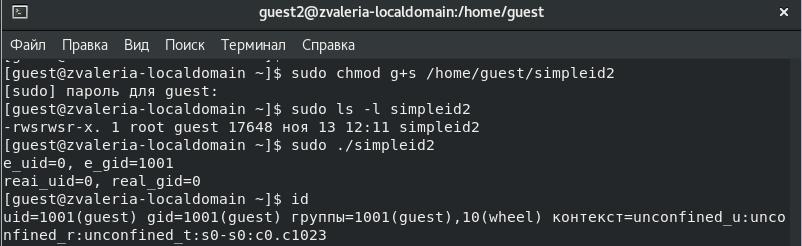


Figure 2: Операции с *SetGID-битом*

1. Создаем программу *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;  
}

1. Откомпилируем её:

$ gcc readfile.c -o readfile

1. Сменим владельца у файла *readfile.c* и изменим права так, чтобы только суперпользователь мог прочитать его.
2. Проверяем, что пользователь *guest* не может прочитать файл *readfile.c*.
3. Сменим у программы *readfile* владельца и установим *SetUID-бит*.
4. Проверим, может ли программа *readfile* прочитать файл *readfile.c*:

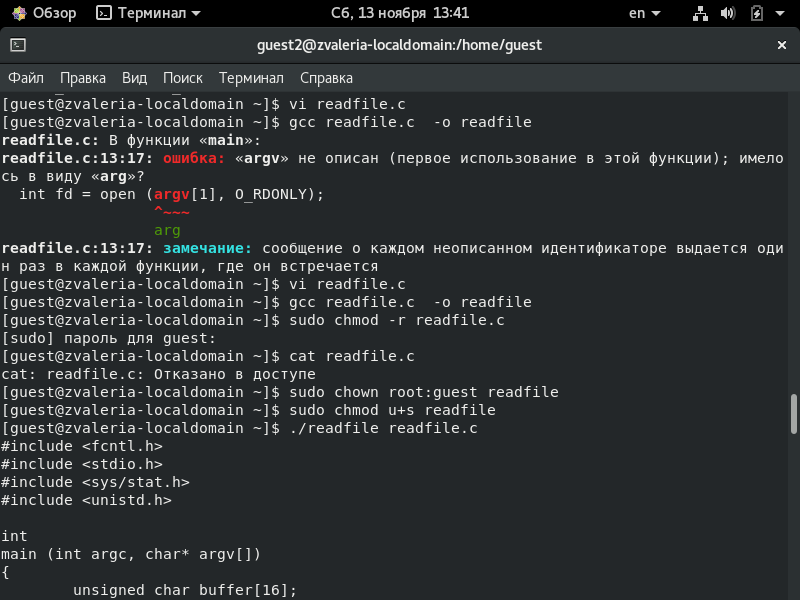


Figure 3: Выполнение пунктов 12-17

1. Проверим, может ли программа *readfile* прочитать файл */etc/shadow*:

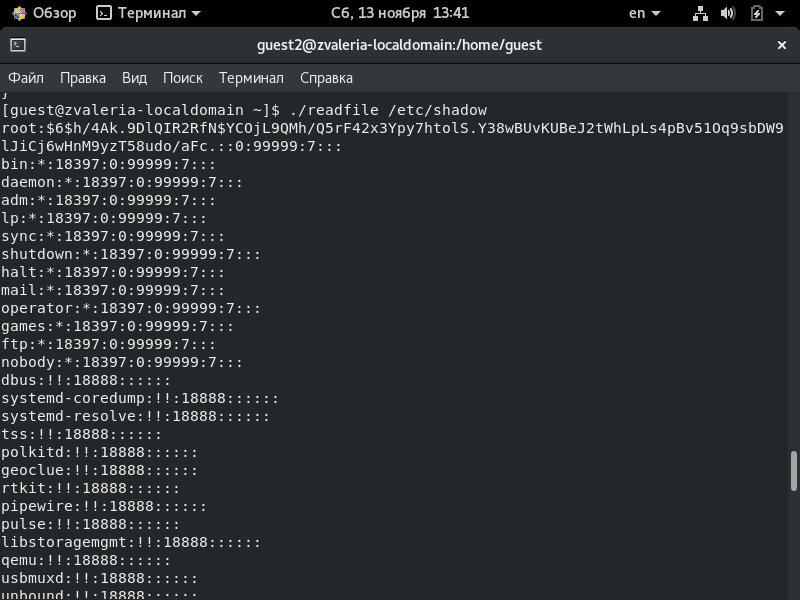


Figure 4: Выполнение пункта 18

## Исследование *Sticky-бита*

1. Выясним, установлен ли атрибут *Sticky* на директории */tmp*:

guest$ ls -l / | grep tmp

1. От имени пользователя *guest* создаем файл *file01.txt* в директории */tmp* со словом *test*:

guest$ echo "test" > /tmp/file01.txt

1. Просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей “все остальные”:

guest$ ls -l /tmp/file01.txt  
guest$ chmod o+rw /tmp/file01.txt  
guest$ ls -l /tmp/file01.txt

1. От пользователя *guest2* попробуем прочитать файл */tmp/file01.txt*:

guest2$ cat /tmp/file01.txt

1. От пользователя *guest2* попробуем дозаписать в файл */tmp/file01.txt* слово *test2*:

guest2$ echo "test2" > /tmp/file01.txt

1. Проверяем содержимое файла:

guest2$ cat /tmp/file01.txt

1. От пользователя *guest2* попробуем записать в файл */tmp/file01.txt* слово *test3*, стерев при этом всю имеющуюся в файле информацию:

guest2$ echo "test3" > /tmp/file01.txt

1. Проверяем содержимое файла:

guest2$ cat /tmp/file01.txt

1. От пользователя *guest2* попробуем удалить файл */tmp/file01.txt*:

guest2$ rm /tmp/fileOl.txt

1. От имени суперпользователя выполним команду, снимающую атрибут *t* с директории */tmp*:

guest2$ sudo chmod -t /tmp

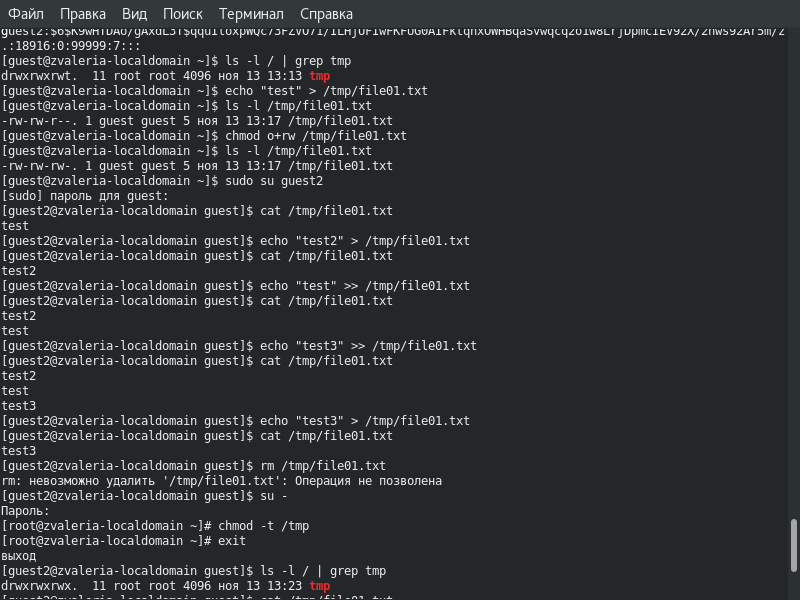


Figure 5: Выполнение с установленным *Sticky-битом*

1. От пользователя *guest2* проверяем, что атрибута *t* у директории */tmp* нет:

guest2$ ls -l / | grep tmp

1. Повторяем предыдущие шаги.
2. Повысим свои права до суперпользователя и вернем атрибут *t* на директорию */tmp*:

guest2$ sudo chmod +t /tmp

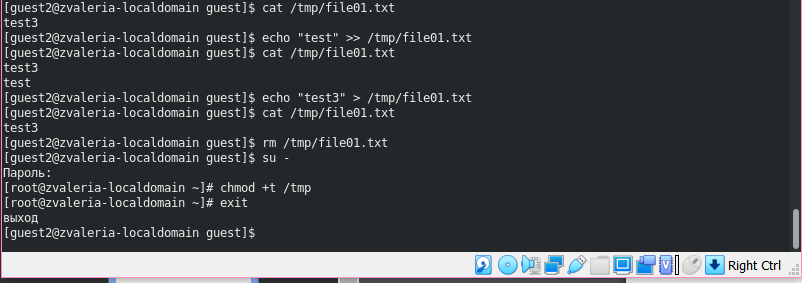


Figure 6: Выполнение со неустановленным *Sticky-битом*

# Вывод

Изучили механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получили практические навыки работы в консоли с дополнительными атрибутами. Рассмотрели работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.