Информационная безопасность. Лабораторная работа #5.

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Хохлачева Яна, учебная группа: НКНбд-01-18

Содержание

## 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.

# Создание программы

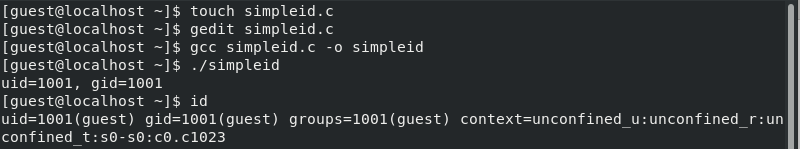
1. Подготовила стенд лабораторной в соответствии с указаниями, а в частности:

* Проверила наличие компилятора **GCC** командой: **gcc -v**
* Отключила систему запретов до очередной перезагрузки системы командой **setenforce 0** и проверила вывод комады **getenforce**
* Ознакомилась с информацией о компиляции программ при помощи **GCC**.

1. Вошла в систему от имени пользователя **guest**.
2. Создала программу **simpleid.c** со следущим содержанием:

#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. Скомплилировала программу при помощи команды **gcc simpleid.c -o simpleid** и убедилась, что файл программы создан.
2. Выполнила программу **simpleid** командой **./simpleid**.
3. Выполнила системную команду **id**, результаты выполнения программы и команды - идентичные.

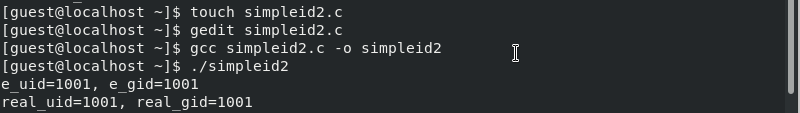


Выполнение simpleid.c и команды id

1. Усложнила программу, добавив вывод действительных идентификаторов, назвала ее **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. Скомпилировала и запустила **simpleid2.c** командами: **gcc simpleid2.c -o simpleid2**, \*\* ./simpleid2\*\*.



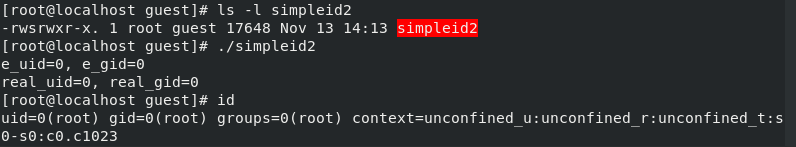
Запуск simpleid2.c

1. От имени суперпользователя выполнила команды: **chown root:guest /simpleid2**, \*\* chmod u+s /simpleid2\*\* .



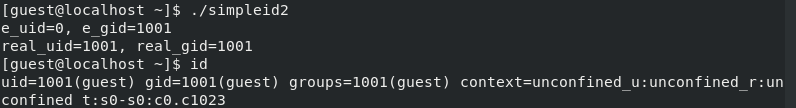
Установка новых атрибутов и смены владельца файла

1. Выполнила проверку правильности установки новых атрибутов и смены владельца файла **simpleid2** командой: **ls -l simpleid2**



Проверка правильности

1. Запустила **simpleid2** и **id**.



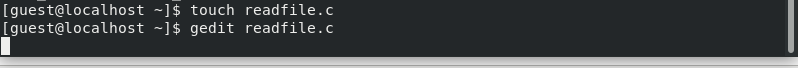
simpleid2 и id

Результаты отличаются

1. Проделала тоже самое относительно SetGID-бита.



1. Создала программу **readfile.c**:

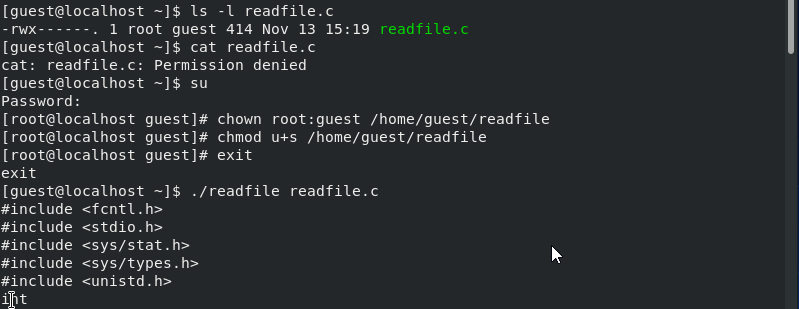


#include <fcntl.h>  
#include <stdio.h>  
#include <sys/stat.h>  
#include <sys/types.h>  
#include <unistd.h>  
int  
main (int argc, char\* argv[])  
{  
unsigned char buffer[16];  
size\_t bytes\_read;  
int i;  
int fd = open (argv[1], O\_RDONLY);  
do  
{  
bytes\_read = read (fd, buffer, sizeof (buffer));  
for (i =0; i < bytes\_read; ++i) printf("%c", buffer[i]);  
}  
while (bytes\_read == sizeof (buffer));  
close (fd);  
return 0;  
}

1. Откомпилировала её командой **gcc readfile.c -o readfile** и сменила владельца у файла **readfile.c** и изменила права так, чтобы только суперпользователь(root) мог прочитать его, a guest не мог, также проверила, что пользователь guest не может прочитать файл **readfile.c**.



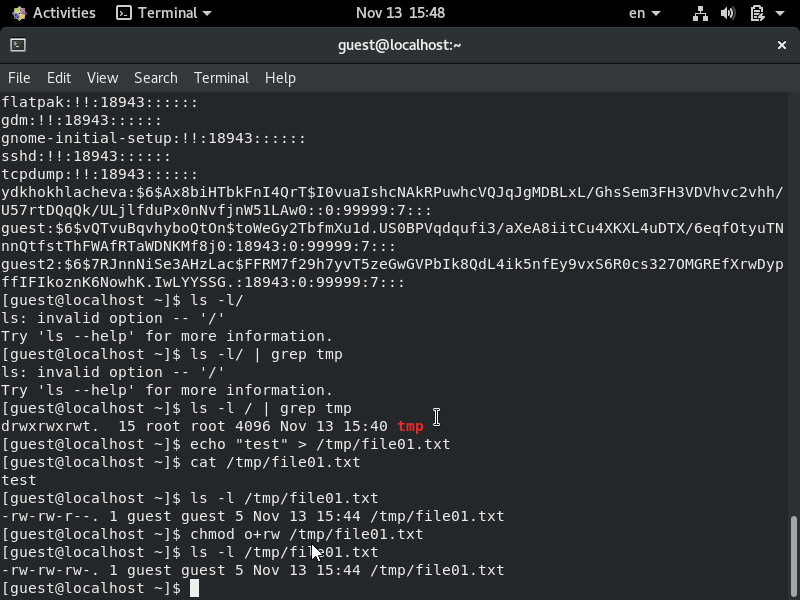
1. Сменила у программы **readfile** владельца и установила SetU’D-бит.Проверила, может ли программа **readfile** прочитать файл **readfile.с**. Проверила, что программа **readfile** прочитать файл **/etc/shadow**.



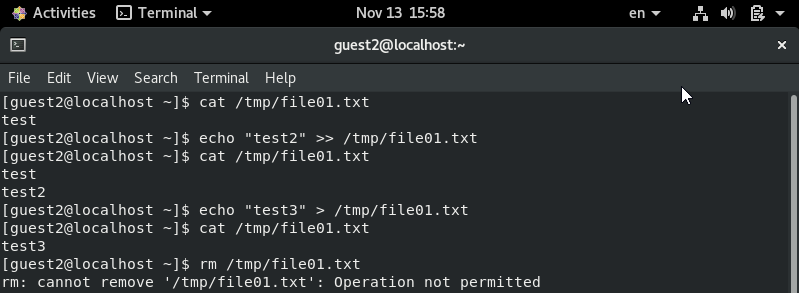
# Исследование Sticky-бита

1. Выяснила, установлен ли атрибут Sticky на директории **/tmp**, для чего выполнил команду: **ls -l / | grep tmp**.От имени пользователя guest создала файл **file01.txt** в директории **/tmp** со словом test командой: **echo “test” > /tmp/file01.txt**. Просмотрела атрибуты у только что созданного файла и разрешил чтение и запись для категории пользователей «все остальные» командами:

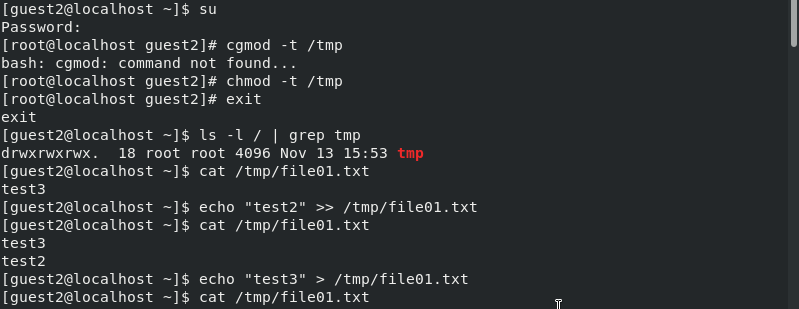
**ls -l /tmp/file01.txt**, **chmod o+rw /tmp/file01.txt**, **ls -l /tmp/file01.txt**.



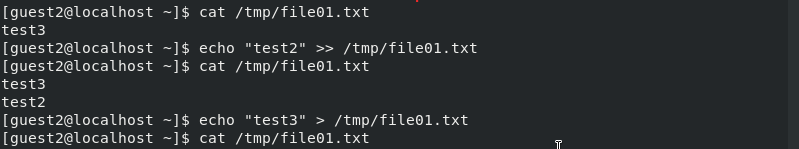
1. От пользователя guest2 попробовала прочитать файл **/tmp/file01.txt** командой: **cat /tmp/file01.txt**. От пользователя guest2 попробовала дозаписать в файл **/tmp/file01.txt** слово test2 командой **echo “test2” > /tmp/file01.txt**. Удалось ли вам выполнить операцию? (Да).Проверила содержимое файла командой **cat /tmp/file01.txt**. От пользователя guest2 попробовала записать в файл **/tmp/file01.txt** слово test3, стерев при этом всю имеющуюся в файле информацию командой **echo “test3” > /tmp/file01.txt**.Удалось ли вам выполнить операцию?(Да). Проверила содержимое файла командой **cat /tmp/file01.txt**. От пользователя guest2 попробовал удалить файл **/tmp/file01.txt** командой **rm /tmp/fileOl.txt**.Удалось ли вам удалить файл?(Нет)



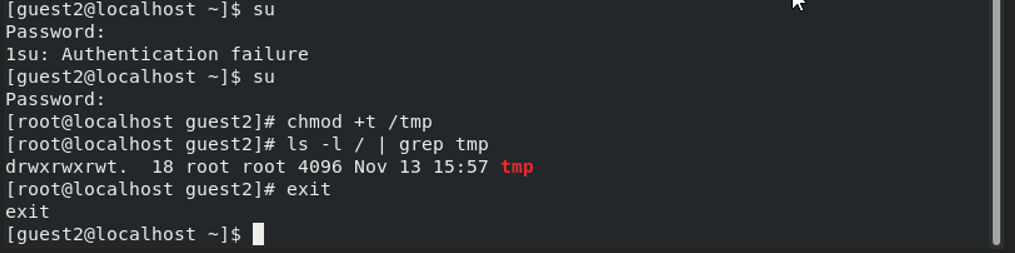
1. Повысила свои права до суперпользователя следующей командой: **su -** И выполнила после этого команду, снимающую атрибут t (Sticky-бит) с директории **/tmp**: **chmod -t /tmp**. Покинула режим суперпользователя командой **exit**.



1. От пользователя guest2 проверил, что атрибута t у директории **/tmp** нет командой: **ls -l / | grep tmp**.Поверила предыдущие шаги.Удалось удалить файл от имени пользователя, не являющегося его владельцем



1. Повысила свои права до суперпользователя и вернула атрибут t на директорию **/tmp**: **su -**, **chmod +t /tmp**, **exit**.



## Вывод

* Изучила механизмы изменения идентификаторов, научилас применять SetUID- и Sticky-биты. 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.