Лабораторная работа №5

Дисциплина: Информационная безопасность

Боровикова Карина Владимировна

Содержание

# 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 Задание

* Произвести работу в консоли с учетных записей суперпользователя и гостя;
* Опытным путем проверить работу механизма смены идентификатора процессов пльзователей, а также влияние бита Sticky на запись и удаление файлов.

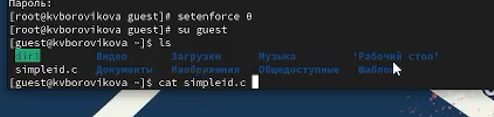
# 3 Теоретическое введение

Для выполнения четвёртой части задания вам потребуются навыки программирования, а именно, умение компилировать простые программы, написанные на языке С (С++), используя интерфейс CLI. Само по себе создание программ не относится к теме, по которой выполняется работа, а является вспомогательной частью, позволяющей увидеть, как реализуются на практике те или иные механизмы дискреционного разграничения доступа. Если при написании (или исправлении существующих) скриптов на bash-e у большинства системных администраторов не возникает проблем, то процесс компилирования, как показывает практика, вызывает необоснованные затруднения

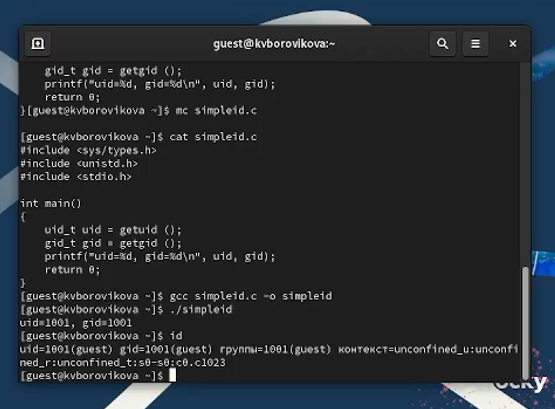
# 4 Выполнение лабораторной работы

1. Войдите в систему от имени пользователя guest. (рис. ??)
2. Создайте программу simpleid.c: (рис. ??)

#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;  
}



Входим в систему от имени пользователя guest, проверяем содержимое файла simpleid.c



Cодержимое файла simpleid.c

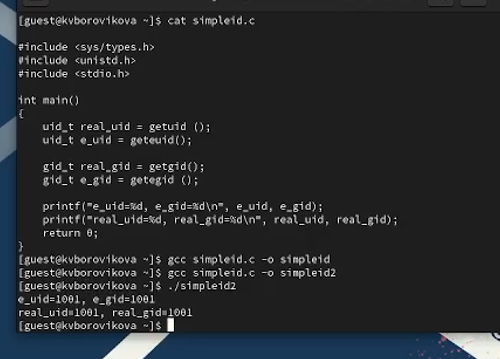
1. Скомплилируйте программу и убедитесь, что файл программы создан: (рис. ??) gcc simpleid.c -o simpleid
2. Выполните программу simpleid: (рис. ??) ./simpleid
3. Выполните системную программу id: (рис. ??) ìd видим, что выведенные данные совпадают
4. Усложните программу, добавив вывод действительных идентификаторов: (рис. ??)

#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;  
}

Получившуюся программу назовите simpleid2.c.

1. Скомпилируйте и запустите simpleid2.c: (рис. ??)

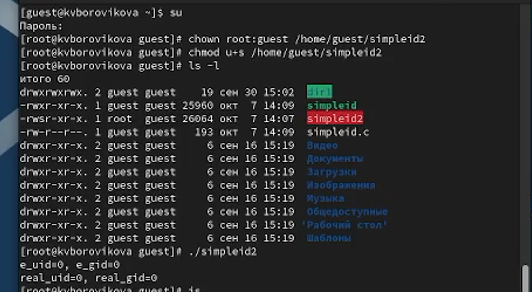
gcc simpleid2.c -o simpleid2  
./simpleid2



Cодержимое файла simpleid.c и компиляция simpleid2

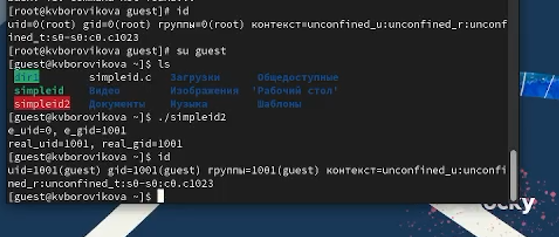
1. От имени суперпользователя выполните команды: (рис. ??)

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



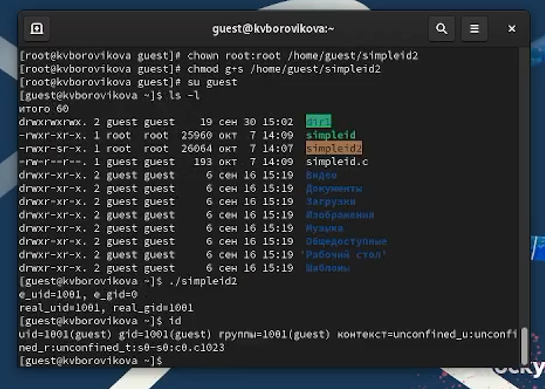
Меняем владельца файла и права доступа на него

1. Используйте sudo или повысьте временно свои права с помощью su. Данные команды меняют владельца файла и права доступа на него.
2. Выполните проверку правильности установки новых атрибутов и смены владельца файла simpleid2: (рис. ??) ls -l simpleid2
3. Запустите simpleid2 и id: (рис. ?? - ??) ./simpleid2 id Видим, что команды выдали различные данные, e\_uid = 0.



Вывод команд simpleid2 и id

1. Проделайте тоже самое относительно SetGID-бита. (рис. ??)



Делаем то же самое для SetGID-бита

Видим, что команды выдали различные данные, e\_gid = 0.

1. Создайте программу readfile.c: (рис. ??)

#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;  
}

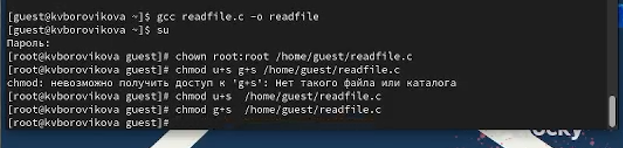


Содержимое readfile.c

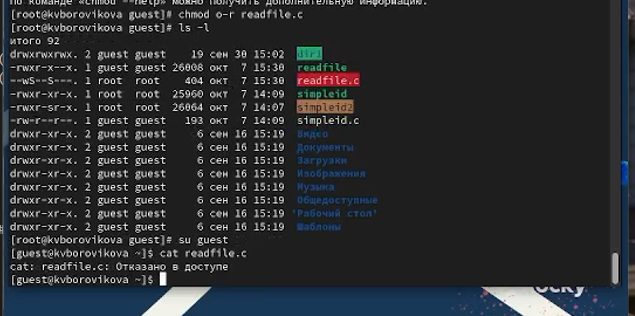
1. Откомпилируйте её. (рис. ??)

gcc readfile.c -o readfile

1. Смените владельца у файла readfile.c (или любого другого текстового файла в системе) и измените права так, чтобы только суперпользователь (root) мог прочитать его, a guest не мог.



Компиляция, смена владельца и прав доступа



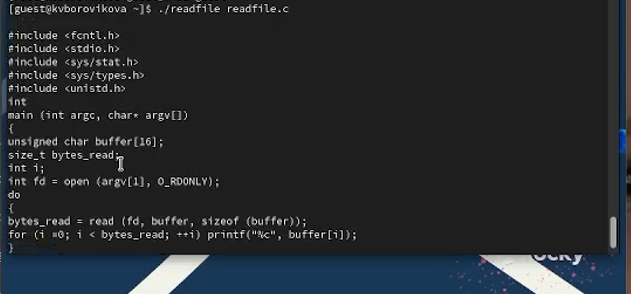
Смена прав доступа

1. Проверьте, что пользователь guest не может прочитать файл readfile.c. (рис. ??)
2. Смените у программы readfile владельца и установите SetU’D-бит. (рис. ??)

Смена прав доступа и владельца

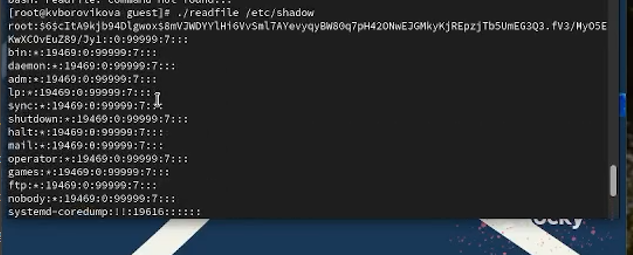
Смена прав доступа и владельца

1. Проверьте, может ли программа readfile прочитать файл readfile.c? (рис. ??)



Программа readfile читает файл readfile.c

1. Проверьте, может ли программа readfile прочитать файл /etc/shadow? (рис. ??)



Программа readfile читает файл /etc/shadow

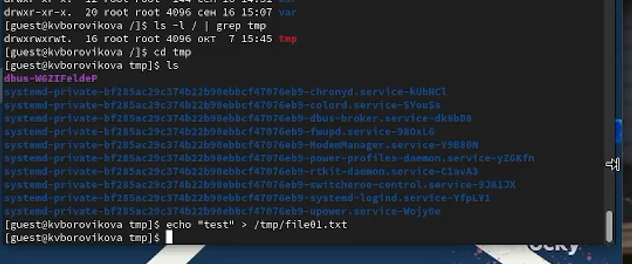
Видим, что программа читает файл /etc/shadow

1. Выясните, установлен ли атрибут Sticky на директории /tmp, для чего выполните команду (рис. ??) ls -l / | grep tmp

Видим, что установлен

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

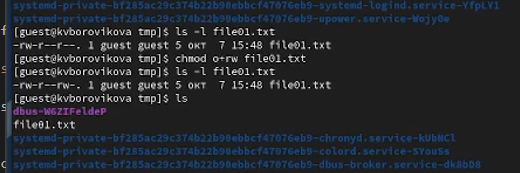
echo "test" > /tmp/file01.txt



Записываем текст в файл

1. Просмотрите атрибуты у только что созданного файла и разрешите чтение и запись для категории пользователей «все остальные»: (рис. ??)

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

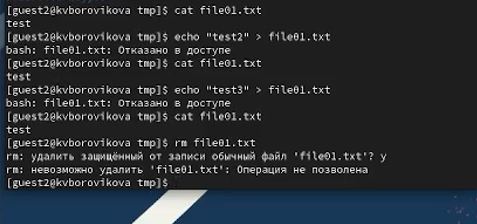


Записываем текст в файл

1. От пользователя guest2 (не являющегося владельцем) попробуйте прочитать файл /tmp/file01.txt: (рис. ??)

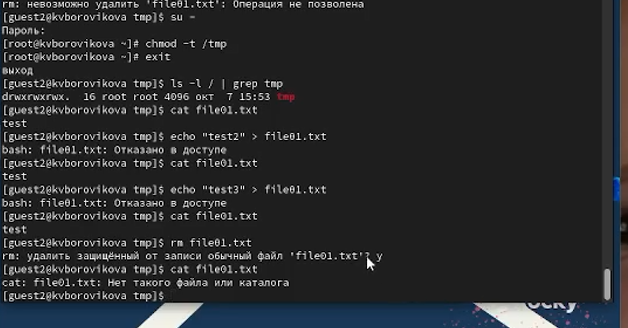
cat /tmp/file01.txt

1. От пользователя guest2 попробуйте дозаписать в файл /tmp/file01.txt слово test2 командой (рис. ??) echo "test2" > /tmp/file01.txt Удалось ли вам выполнить операцию? нет
2. Проверьте содержимое файла командой (рис. ??) cat /tmp/file01.txt
3. От пользователя guest2 попробуйте записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию командой (рис. ??) echo "test3" > /tmp/file01.txt Удалось ли вам выполнить операцию? нет
4. Проверьте содержимое файла командой (рис. ??) cat /tmp/file01.txt
5. От пользователя guest2 попробуйте удалить файл /tmp/file01.txt командой (рис. ??) rm /tmp/fileOl.txt Удалось ли вам удалить файл? нет



Выполняем действия с файлом file01.txt

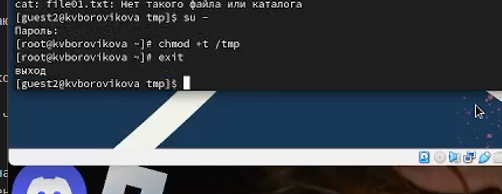
1. Повысьте свои права до суперпользователя следующей командой su - и выполните после этого команду, снимающую атрибут t (Sticky-бит) с директории /tmp: (рис. ??) chmod -t /tmp
2. Покиньте режим суперпользователя командой (рис. ??) exit
3. От пользователя guest2 проверьте, что атрибута t у директории /tmp нет: (рис. ??) ls -l / | grep tmp
4. Повторите предыдущие шаги. Какие наблюдаются изменения? Видим, что нам удалось удалить файл. (рис. ??)
5. Удалось ли вам удалить файл от имени пользователя, не являющегося его владельцем? да. (рис. ??)



Выполняем действия с файлом file01.txt после снятия Sticky-бита

1. Повысьте свои права до суперпользователя и верните атрибут t на директорию /tmp: (рис. ??)

su -  
chmod +t /tmp  
exit



Возвращаем на место

# 5 Выводы

В ходе выполнения лабораторной работы нам удалось изучить механизмы изменения идентификаторов, применения SetUID- и Sticky-битов, получить практические навыки работы в консоли с дополнительными атрибутамиб рассмотреть работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

# Список литературы