Лабораторная работа №5

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Латыпова Диана. НФИбд-02-21

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

**5.3.1. Создание программы**

1. Войдите в систему от имени пользователя guest.
2. Создайте программу simpleid.c
3. Скомплилируйте программу и убедитесь, что файл программы создан:

gcc simpleid.c -o simpleid

1. Выполните программу simpleid: ./simpleid
2. Выполните системную программу id: id и сравните полученный вами результат с данными предыдущего пункта задания.
3. Усложните программу, добавив вывод действительных идентификаторов. Получившуюся программу назовите simpleid2.c.
4. Скомпилируйте и запустите simpleid2.c:

gcc simpleid2.c -o simpleid2  
./simpleid2

1. От имени суперпользователя выполните команды

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

1. Используйте sudo или повысьте временно свои права с помощью su. Поясните, что делают эти команды.
2. Выполните проверку правильности установки новых атрибутов и смены владельца файла simpleid2: ls -l simpleid2
3. Запустите simpleid2 и id:

./simpleid2  
id

Сравните результаты. 12. Проделайте тоже самое относительно SetGID-бита. 13. Создайте программу readfile.c. 14. Откомпилируйте её.

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.
2. Проверьте, что пользователь guest не может прочитать файл readfile.c.
3. Смените у программы readfile владельца и установите SetU’D-бит.
4. Проверьте, может ли программа readfile прочитать файл readfile.c?
5. Проверьте, может ли программа readfile прочитать файл /etc/shadow? Отразите полученный результат и ваши объяснения в отчёте.

**5.3.2. Исследование Sticky-бита**

1. Выясните, установлен ли атрибут Sticky на директории /tmp, для чего выполните команду ls -l / | grep tmp
2. От имени пользователя guest создайте файл file01.txt в директории /tmp со словом test: echo "test" > /tmp/file01.txt
3. Просмотрите атрибуты у только что созданного файла и разрешите чтение и запись для категории пользователей «все остальные»:

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

1. От пользователя guest2 (не являющегося владельцем) попробуйте прочитать файл /tmp/file01.txt: cat /tmp/file01.txt
2. От пользователя guest2 попробуйте дозаписать в файл /tmp/file01.txt слово test2 командой echo "test2" > /tmp/file01.txt Удалось ли вам выполнить операцию?
3. Проверьте содержимое файла командой cat /tmp/file01.txt
4. От пользователя guest2 попробуйте 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?
5. Проверьте содержимое файла командой cat /tmp/file01.txt
6. От пользователя guest2 попробуйте удалить файл /tmp/file01.txt командой rm /tmp/fileOl.txt Удалось ли вам удалить файл?
7. Повысьте свои права до суперпользователя следующей командой su - и выполните после этого команду, снимающую атрибут t (Sticky-бит) с директории /tmp:

chmod -t /tmp

1. Покиньте режим суперпользователя командой exit
2. От пользователя guest2 проверьте, что атрибута t у директории /tmp нет: ls -l / | grep tmp
3. Повторите предыдущие шаги. Какие наблюдаются изменения?
4. Удалось ли вам удалить файл от имени пользователя, не являющегося его владельцем? Ваши наблюдения занесите в отчёт.
5. Повысьте свои права до суперпользователя и верните атрибут t на директорию /tmp:

su -  
chmod +t /tmp  
exit

# 3 Теоретическое введение

* **UID (User ID)** — идентификатор пользователя, который определяет, от имени какого пользователя выполняется программа[1].
* **GID (Group ID)** — идентификатор группы, определяющий группу, к которой принадлежит пользователь [1].

Программы могут выполняться с различными идентификаторами: действительный (реальный) и эффективный UID/GID. *Реальный UID* — это пользователь, который запустил процесс, а *эффективный UID* — это пользователь, от имени которого процесс выполняет операции.

* **SetUID** — специальный бит, который при установке на файл исполняемой программы позволяет запускать её с правами владельца файла, а не того пользователя, который её запускает [2].
* **SetGID** работает аналогично, но изменяет эффективный GID на GID владельца файла [2].

**Sticky-bit** — атрибут директории, который запрещает пользователям удалять файлы, если они не являются их владельцами, даже если у них есть права на запись в директорию[2].

# 4 Выполнение лабораторной работы

Для начала установили компилятор gcc (рис. 1):

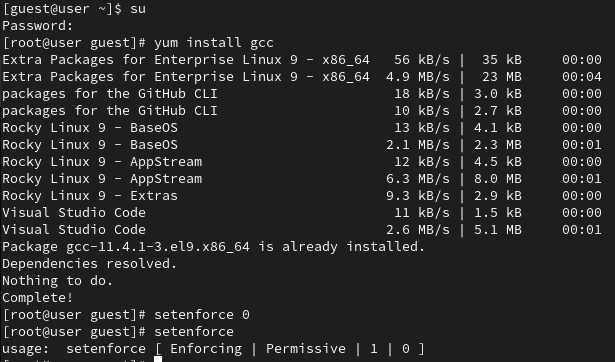


Рис. 1: Установка gcc

Вошла в систему от имени пользователя guest, создала программу simpleid.c:

#include <sys/types.h> // Для использования типов данных UID и GID  
#include <unistd.h> // Для функций getuid(), geteuid(), getgid(), getegid()  
#include <stdio.h> // Для функции printf()  
  
int main () {  
 // Получение эффективных идентификаторов пользователя и группы  
 uid\_t uid = geteuid(); // Получение эффективного UID  
 gid\_t gid = getegid(); // Получение эффективного GID  
   
 // Вывод эффективных идентификаторов  
 printf("uid=%d, gid=%d\n", uid, gid);  
   
 return 0;  
}

Скомплилировала программу, выполнила программу simpleid и системную программу id:

gcc simpleid.c -o simpleid  
./simpleid  
id

Видим, что UID и GID совпдают(рис. 2):

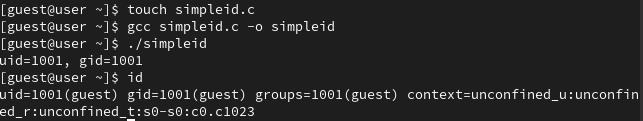


Рис. 2: Задание 1.1-5

Далее усложнила программу, добавив вывод действительных идентификаторов:

#include <sys/types.h> // Для использования типов данных UID и GID  
#include <unistd.h> // Для функций getuid(), geteuid(), getgid(), getegid()  
#include <stdio.h> // Для функции printf()  
  
int main () {  
 // Получение реальных и эффективных идентификаторов пользователя и группы  
 uid\_t real\_uid = getuid(); // Получение реального UID  
 uid\_t e\_uid = geteuid(); // Получение эффективного UID  
 gid\_t real\_gid = getgid(); // Получение реального GID  
 gid\_t e\_gid = getegid(); // Получение эффективного GID  
  
 // Вывод реальных и эффективных идентификаторов  
 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 назвала simpleid2.c, скомпилировала и запустила simpleid2.c (рис. 3):

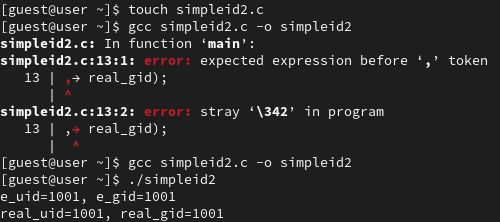


Рис. 3: Задание 1.6-7

От имени суперпользователя выполнила команды

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

Команда chown меняет владельца файла, а chmod u+s устанавливает SetUID бит, что позволяет выполнять программу с правами владельца (root). Далее выполнила проверку правильности установки новых атрибутов и смены владельца файла simpleid2. Ну и запустила simpleid2 и id. Проделала то же самое относительно SetGID-бита (рис. 4):

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

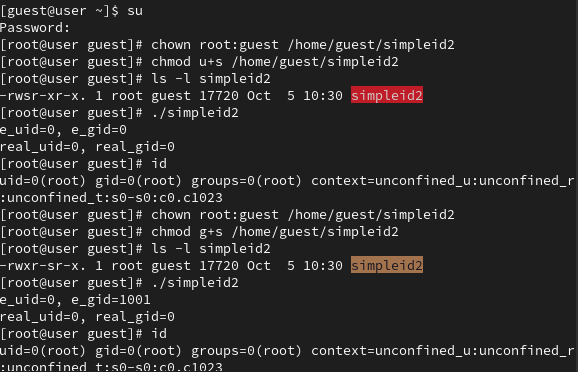


Рис. 4: Задание 1.8-12

Создала программу readfile.c:

#include <fcntl.h> // Для функции open()  
#include <stdio.h> // Для функции printf()  
#include <sys/stat.h> // Для работы с атрибутами файлов  
#include <sys/types.h> // Для использования типов данных  
#include <unistd.h> // Для функций read(), close()  
  
int main (int argc, char\* argv[]) {  
 unsigned char buffer[16]; // Буфер для хранения прочитанных данных  
 size\_t bytes\_read; // Количество прочитанных байт  
 int i;  
 int fd = open(argv[1], O\_RDONLY); // Открытие файла на чтение  
   
 do {  
 bytes\_read = read(fd, buffer, sizeof(buffer)); // Чтение данных из файла  
 for (i = 0; i < bytes\_read; ++i) {  
 printf("%c", buffer[i]); // Вывод прочитанных данных  
 }  
 } while (bytes\_read == sizeof(buffer)); // Чтение продолжается до конца файла  
   
 close(fd); // Закрытие файла  
 return 0;  
}

Откомпилировала её (рис. 5):

Задание 1.13-14

Рис. 5: Задание 1.13-14

Сменила владельца у файла readfile.c и изменила права так, чтобы только суперпользователь (root) мог прочитать его, a guest не мог (рис. 6):

Задание 1.13-14

Рис. 6: Задание 1.13-14

Проверила, что пользователь guest не может прочитать файл readfile.c,сменила у программы readfile владельца и установила SetU’D-бит. Проверила, может ли программа readfile прочитать файл readfile.c. Проверила, может ли программа readfile прочитать файл /etc/shadow? Ничего из выше перечисленного прочитать не смогли (рис. 7):

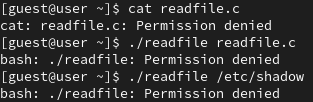


Рис. 7: Задание 1.16-19 (1)

Видим, что только суперпользователь может выполнить вышеперечисленное(рис. 8):

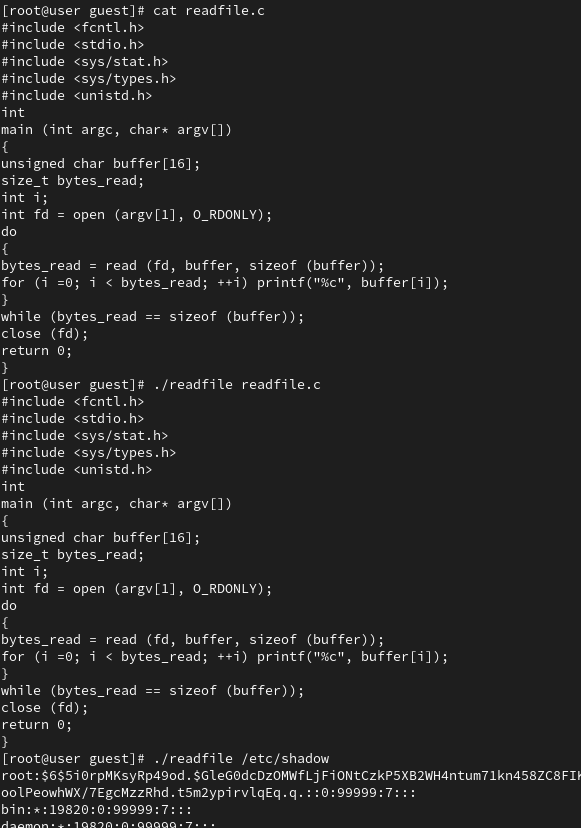


Рис. 8: Задание 1.16-19 (2)

Начали исследование sticky-бита.

Выяснила, установлен ли атрибут Sticky на директории /tmp, для чего выполнила команду:

ls -l / | grep tmp

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

echo "test" > /tmp/file01.txt

Просмотрела атрибуты у только что созданного файла и разрешила чтение и запись для категории пользователей «все остальные» (рис. 9):

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

От пользователя guest2 (не являющегося владельцем) попробовала прочитать файл /tmp/file01.txt:

cat /tmp/file01.txt



Рис. 9: Задание 2.1-4

От пользователя guest2 попробовала дозаписать в файл /tmp/file01.txt слово test2. Однако мне этого не удалось сделать. От пользователя guest2 попробовала записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию, этого тоже не удалось сделать. От пользователя guest2 попробовала удалить файл /tmp/file01.txt, этого тоже не удалось сделать. После чего повысила свои права до суперпользователя и выполнила после этого команду, снимающую атрибут t (Sticky-бит) с директории /tmp:

chmod -t /tmp

Покинула режим суперпользователя. Повторила предыдущие шаги. Но сколько бы ни старалась, изменений не видно. Удалить файл от имени пользователя, не являющегося его владельцем не получилось (рис. 10):

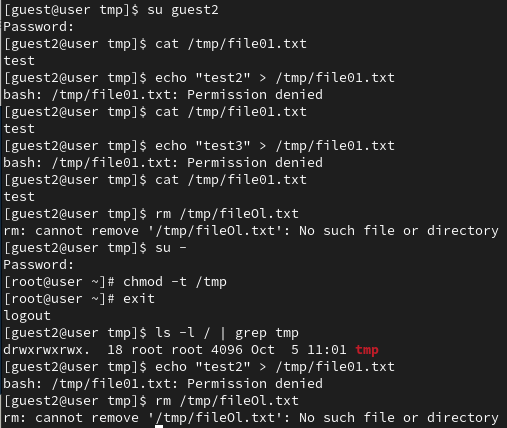


Рис. 10: Задание 2.5-14

После всего повысила свои права до суперпользователя и вернула атрибут t на директорию /tmp:

su -  
chmod +t /tmp  
exit

(рис. 11):

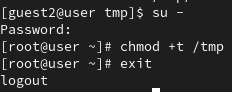


Рис. 11: Задание 2.15

# 5 Выводы

Я изучила механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получила практические навыки работы в консоли с дополнительными атрибутами. А также рассмотрела работу механизма смены идентификатора процессов пользователей, и влияние бита Sticky на запись и удаление файлов.

# Список литературы

1. Права в Linux: команды и группы [Электронный ресурс]. Tprogger, 2023. URL: <https://tproger.ru/articles/prava-v-linux-komandy-i-gruppy>.

2. Использование SETUID, SETGID и Sticky bit для расширенной настройки прав доступа в операционных системах Linux [Электронный ресурс]. RuVDS, 2024. URL: <https://ruvds.com/ru/helpcenter/suid-sgid-sticky-bit-linux/>.