Лабораторная работа №5

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Ильин Андрей Владимирович

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 Задачи

1. Создать программу способную выводить gid, uid и провести ислледование SetUID-битов.
2. Исследовать Sticky-бит.

# 3 Теоретическое введение

## 3.1 Термины

* Терминал (или «Bash», сокращение от «Bourne-Again shell») — это программа, которая используется для взаимодействия с командной оболочкой. Терминал применяется для выполнения административных задач, например: установку пакетов, действия с файлами и управление пользователями. [1]
* Права доступа определяют, какие действия конкретный пользователь может или не может совершать с определенным файлами и каталогами. [2]
* Расширенные атрибуты файловых объектов (далее - расширенные атрибуты) - поддерживаемая некоторыми файловыми системами возможность ассоциировать с файловыми объектами произвольные метаданные. [3]
* Sticky Bit - в случае, если этот бит установлен для папки, то файлы в этой папке могут быть удалены только их владельцем. [4]

## 3.2 Окружение

* Rocky Linux - это корпоративная операционная система с открытым исходным кодом, разработанная таким образом, чтобы быть на 100% совместимой с Red Hat Enterprise Linux. Он находится в стадии интенсивной разработки сообществом. [5]
* Git - это распределенное программное обеспечение для контроля версиями. [6]
* VirtualBox - это кросс-платформенное ПО для виртуализации x86 и AMD64/Intel64 с открытым кодом для корпоративного и домашнего использования. [7]

# 4 Выполнение лабораторной работы

## 4.1 Создание программы

1. Создадим, скомплилируем и запустим программу simpleid.c. Сравним c выводом команды id. (рис. [1](#fig:001))

gcc simpleid.c -o simpleid  
./simpleid  
id

#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. Создадим, скомплилируем и запустим программу simpleid2.c (усложненная версия simpleid.c). (рис. [1](#fig:001))

gcc simpleid2.c -o simpleid2  
./simpleid2

#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;  
}

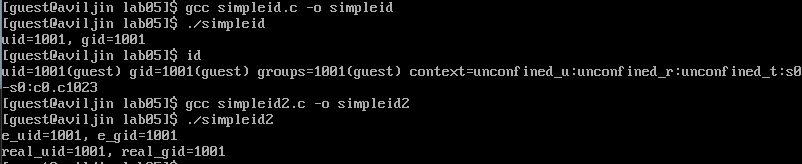


Figure 1: Запуск simpleid.c и simpleid2.c

1. От имени суперпользователя сменим пользователя и изменим аттрибуты на simpleid2. Выполним проверку правильности установки новых атрибутов и смены владельца файла simpleid2. После чего запустим simpleid2 (от имени guest) (рис. [2](#fig:002))

su  
chown root:guest /home/guest/simpleid2  
chmod u+s /home/guest/simpleid2  
exit  
ls -l simpleid2  
./simpleid2

1. От имени суперпользователя устаном SetGID-бит на simpleid2. Выполним проверку правильности установки новых атрибутов файла simpleid2. После чего запустим simpleid2 (от имени guest) (рис. [2](#fig:002))

su  
chmod u-s /home/guest/simpleid2  
chmod g+s /home/guest/simpleid2  
exit  
ls -l simpleid2  
./simpleid2



Figure 2: Запуск simpleid2.c с измененным владельцем и аттрибутами

1. Создим программу readfile.c. Откомпилируем ее. Сменим владельца у файла readfile.c и изменим права так, чтобы только суперпользователь (root) мог прочитать его. Проверим, что пользователь guest не может прочитать файл readfile.c. После этого сменим у программы readfile владельца и установим SetUID-бит. (рис. [3](#fig:003))

#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;  
}

su  
chown root:guest readfile.c  
chmod 700 readfile.c  
exit  
cat readfile.c  
su  
chown root:guest readfile  
chmod u+s readfile  
exit

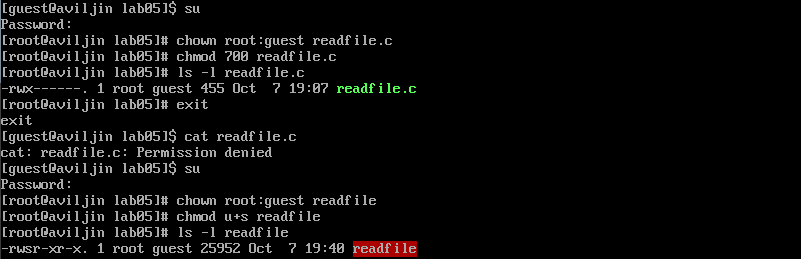


Figure 3: Подготовка к запуску readfile

1. Проверим, может ли программа readfile прочитать файлы readfile.c и /etc/shadow. (рис. [4](#fig:004), [5](#fig:005))

./readfile readfile.c

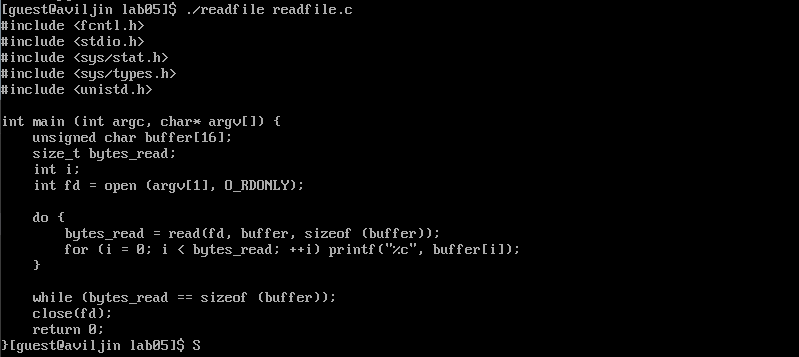


Figure 4: Запуск readfile (readfile.c)

./readfile /etc/shadow

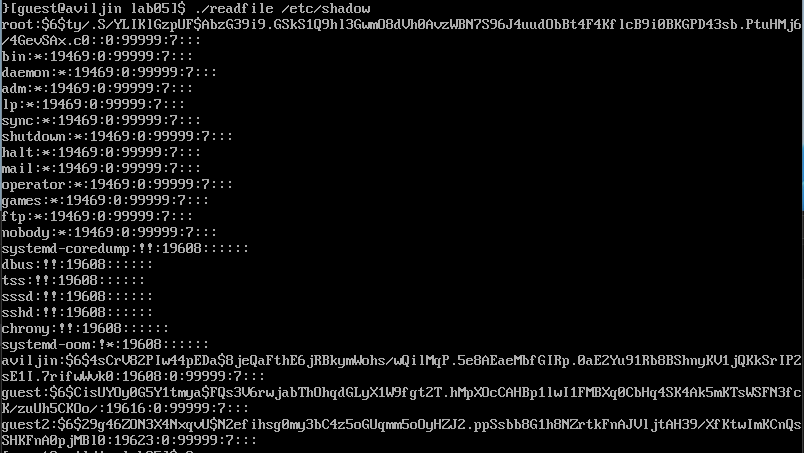


Figure 5: Запуск readfile (/etc/shadow)

## 4.2 Исследование Sticky-бита

1. Выясним, установлен ли атрибут Sticky на директории /tmp. От имени пользователя guest создадим файл file01.txt в директории /tmp со словом test. Просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные». (рис. [6](#fig:006))

ls -l / | grep tmp  
echo "test" > /tmp/file01.txt  
ls -l /tmp/file01.txt  
chmod o+rw /tmp/file01.txt  
ls -l /tmp/file01.txt

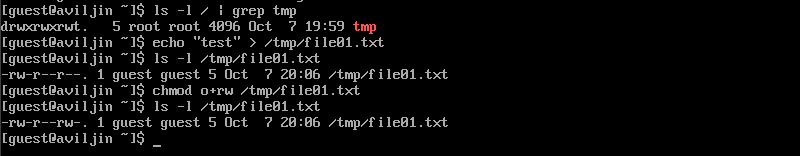


Figure 6: Атрибут Sticky

1. От имени пользователя guest2 проведем исследование атрибута Sticky. (рис. [7](#fig:007))

cat /tmp/file01.txt  
echo "test2" >> /tmp/file01.txt  
cat /tmp/file01.txt  
echo "test3" > /tmp/file01.txt  
cat /tmp/file01.txt  
rm /tmp/fileOl.txt

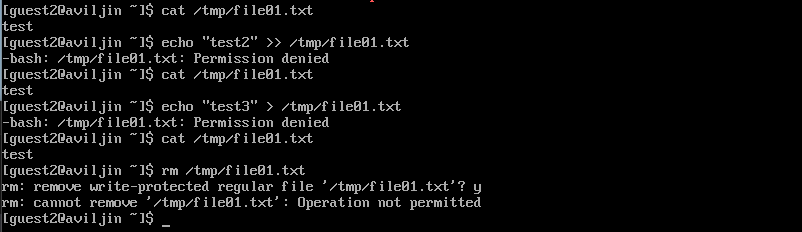


Figure 7: Исследование Sticky (1)

1. Удалим атрибут Sticky на директории /tmp и повторим действия из предыдущего пункта. После вернем атрибут. (рис. [8](#fig:008))

su -  
chmod -t /tmp  
cat /tmp/file01.txt  
exit  
ls -l / | grep tmp  
  
echo "test2" >> /tmp/file01.txt  
cat /tmp/file01.txt  
echo "test3" > /tmp/file01.txt  
cat /tmp/file01.txt  
rm /tmp/fileOl.txt  
  
su -  
chmod +t /tmp  
cat /tmp/file01.txt  
exit  
ls -l / | grep tmp

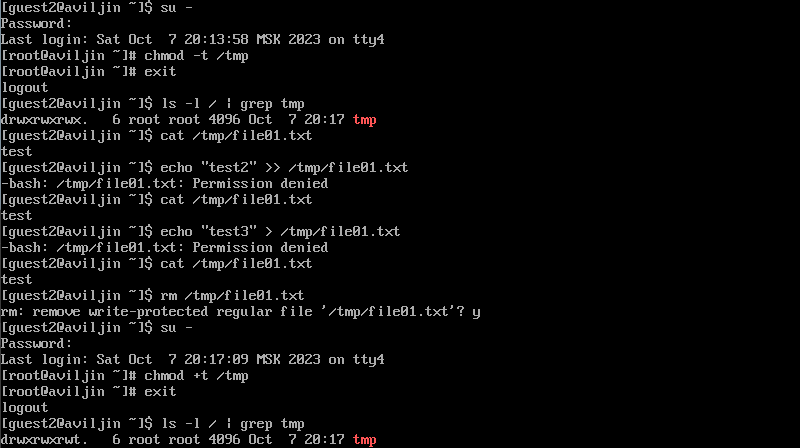


Figure 8: Исследование Sticky (2)

# 5 Анализ результатов

Работа выполненна без непредвиденных проблем в соответствии с руководством. Ошибок и сбоев не произошло.

# 6 Выводы

Изучены идентификаторы SetUID-биты и Sticky-биты. Опробовали их действие на практике. Изучили влияние бита Sticky. Повысили свои навыки использования интерфейса командой строки (CLI).

# Список литературы

1. Терминал Linux [Электронный ресурс]. URL: <{https://www.reg.ru/blog/linux-shpargalka-komandy-terminala-dlya-novichkov/#:~:text=%D0%A2%D0%B5%D1%80%D0%BC%D0%B8%D0%BD%D0%B0%D0%BB%20(%D0%B8%D0%BB%D0%B8%20%C2%ABBash%C2%BB%2C,%D1%81%20%D1%84%D0%B0%D0%B9%D0%BB%D0%B0%D0%BC%D0%B8%20%D0%B8%20%D1%83%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B5%20%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8F%D0%BC%D0%B8}>.

2. Права доступа [Электронный ресурс]. URL: <https://codechick.io/tutorials/unix-linux/unix-linux-permissions>.

3. Расширенные атрибуты [Электронный ресурс]. URL: <{https://wiki.astralinux.ru/pages/viewpage.action?pageId=149063848#:~:text=Common%20Edition%202.12-,%D0%A7%D1%82%D0%BE%20%D1%82%D0%B0%D0%BA%D0%BE%D0%B5%20%D1%80%D0%B0%D1%81%D1%88%D0%B8%D1%80%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5%20%D0%B0%D1%82%D1%80%D0%B8%D0%B1%D1%83%D1%82%D1%8B,%D1%81%20%D1%84%D0%B0%D0%B9%D0%BB%D0%BE%D0%B2%D1%8B%D0%BC%D0%B8%20%D0%BE%D0%B1%D1%8A%D0%B5%D0%BA%D1%82%D0%B0%D0%BC%D0%B8%20%D0%BF%D1%80%D0%BE%D0%B8%D0%B7%D0%B2%D0%BE%D0%BB%D1%8C%D0%BD%D1%8B%D0%B5%20%D0%BC%D0%B5%D1%82%D0%B0%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D0%B5}>.

4. Использование SETUID, SETGID и Sticky bit для расширенной настройки прав доступа в операционных системах Linux [Электронный ресурс]. 2021. URL: <https://ruvds.com/ru/helpcenter/suid-sgid-sticky-bit-linux/>.

5. Документация Rocky Linux [Электронный ресурс]. URL: <https://docs.rockylinux.org/>.

6. Git-Guides [Электронный ресурс]. URL: <https://github.com/git-guides>.

7. VirtualBox [Электронный ресурс]. URL: <https://www.virtualbox.org/>.