

Отчёт по лабораторной работе

Лабораторная работа № 5

Акопян Изабелла Арменовна

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	7
4	Выводы	16
	Список литературы	17

Список таблиц

Список иллюстраций

3.1	Программа для чтения uid и gid	7
3.2	Результат работы	8
3.3	chown и chmod	9
3.4	readfile.c	10
3.5	смена владельца	11
3.6	чтение readfile.c	11
3.7	etc/shadow	12
3.8	Проверка функциональности Sticky-бита на примере файла, созданно- го в каталоге /tmp	12
3.9	атрибут t убран	14
3.10	атрибут t добавлен	15

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 Теоретическое введение

Речь пойдет о трех битах – Setuid, Setgid и Sticky Bit. Это специальные типы разрешений позволяют задавать расширенные права доступа на файлы или каталоги.

Setuid – это бит разрешения, который позволяет пользователю запускать исполняемый файл с правами владельца этого файла. Другими словами, использование этого бита позволяет нам поднять привилегии пользователя в случае, если это необходимо.

Принцип работы Setgid очень похож на setuid с отличием, что файл будет запускаться пользователем от имени группы, которая владеет файлом.

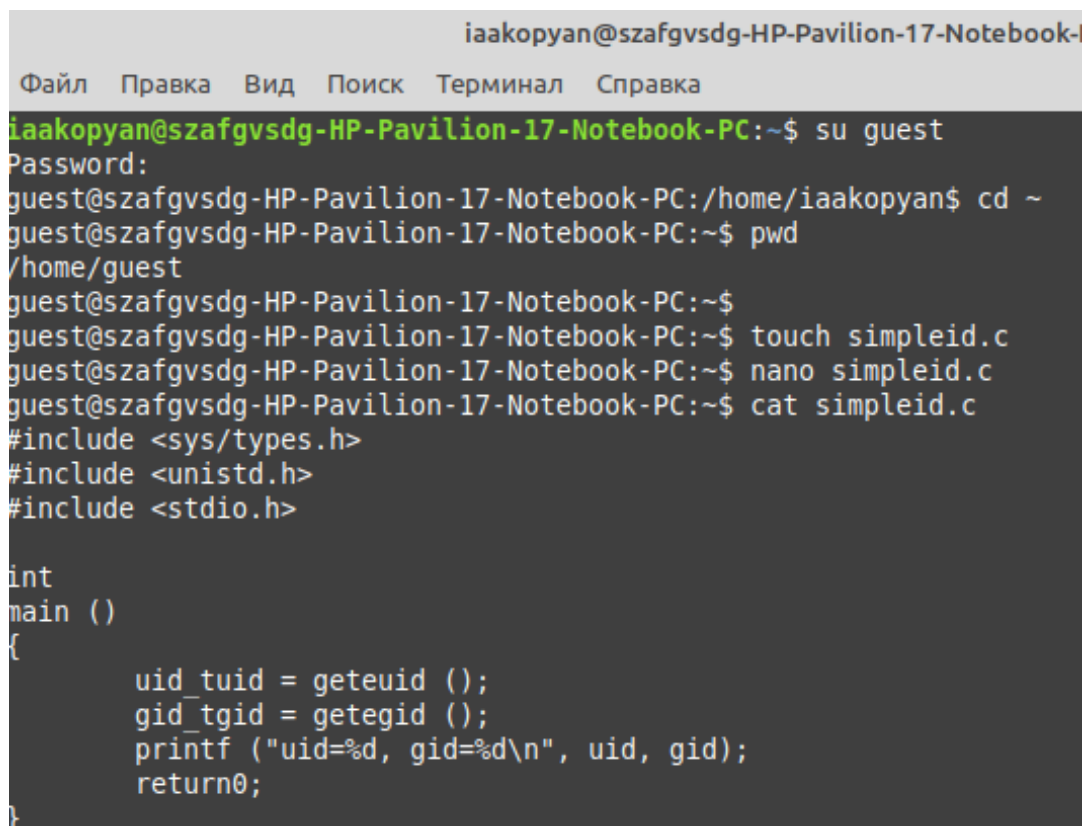
Последний специальный бит разрешения – это Sticky Bit . В случае, если этот бит установлен для папки, то файлы в этой папке могут быть удалены только их владельцем.

В отличие от установки sticky на каталог, на файл такой бит устанавливать уже не имеет смысла. Многие современные ядра попросту игнорируют sticky на файле. На файлах он использовался на старых системах с малой ОЗУ и был очень важен в те времена. Он запрещал выгрузку программ из памяти.

SUID работает только с файлами. Вы можете применять SGID к каталогам и файлам. Вы можете применять липкий бит только к каталогам. Если индикаторы «s», «g» или «t» отображаются в верхнем регистре, исполняемый бит (x) не установлен.

3 Выполнение лабораторной работы

Создали простую программу на языке C для чтения uid и gid. Проверили ее работу, сравнив с системной утилитой id (см. рис. @fig:000@fig:001).



```
iaakopyan@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ su guest
Password:
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:/home/iaakopyan$ cd ~
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ pwd
/home/guest
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ touch simpleid.c
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ nano simpleid.c
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ cat 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;
}
```

Рис. 3.1: Программа для чтения uid и gid

```

guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ nano simpleid.c
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ gcc simpleid.c -o simpleid
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ ./simpleid
uid=1003, gid=1003
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ id
uid=1003(guest) gid=1003(guest) groups=1003(guest)
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ nano simpleid.c

Use "fg" to return to nano.

[1]+  Stopped                  nano simpleid.c
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ nano simpleid.c
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ gcc simpleid.c -o simpleid2
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ ./simpleid2
e_uid=1003, e_gid=1003
real_uid=1003, real_gid=1003
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ ls -l simpleid
-rwxrwxr-x 1 guest guest 16792 окт  7 14:05 simpleid
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ ls -l simpleid1
ls: cannot access 'simpleid1': No such file or directory
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ ls -l simpleid2
-rwsrwsr-x 1 root guest 16880 окт  7 14:09 simpleid2
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ ./simpleid2
e_uid=0, e_gid=1003
real_uid=1003, real_gid=1003
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ id
uid=1003(guest) gid=1003(guest) groups=1003(guest)

```

Рис. 3.2: Результат работы

Усложнили программу, добавив вывод действительных идентификаторов. Командой `chown root:guest /home/guest/simpleid2` передала права обладания файлов суперпользователю. Присвоила файлу SUID и GUID биты (см. рис. @fig:002). Проверили функционирование. И правда, при выполнении с правами суперпользователя `uid` указывается суперпользователя. Так как мы не меняли группу файла, изменений в программе при выставлении GUID не наблюдается.


```
root@szafgvsdg-HP-Pavilion-17-Notebook-PC: /
Файл Правка Вид Поиск Терминал Справка
iaakopyan@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ su -
Password:
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:~# chown root:guest /home/guest/simpleid2
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:~# chown u+s /home/guest/simpleid2
chown: invalid user: 'u+s'
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:~# chmod u+s /home/guest/simpleid2
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:~# su
```

Рис. 3.3: chown и chmod

Создали программу для чтения файлов. Установили на текст программы права доступа 000. Установили владельцем исполняемого файла суперпользователя. Поставили SUID бит. Запустили исполняемый файл от обычного пользователя. Исполняемый файл смог прочесть текст программы, у которого отсутствует разрешение для чтения (см. рис. @fig:003@fig:006).

```

guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ touch readfile.c
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ nano readfile.c

Use "fg" to return to nano.

[2]+  Stopped                  nano readfile.c
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ nano readfile.c

Use "fg" to return to nano.

[3]+  Stopped                  nano readfile.c
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ fg
nano readfile.c
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ gcc readfile.c -o readfile
readfile.c:7:15: error: expected ')' before 'char'
    7 | main (int argc, char* argv[])
      |                  ^
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ nano readfile.c
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ gcc readfile.c -o readfile
guest@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ cat 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;
}

```

Рис. 3.4: readfile.c


```

0: pcx=01,35: .mov=01,35: .segmentation fault (core dumped)
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:/home/guest# ./readfile /etc/shadow
root:$6$skRDqf6tyHuG7PWZ$c.DQ8co6hIIFq2xcD2P1pyrK046QdhRGFj8Mv3xnA7aBoPk5kE.npZGtqzgR6i3kzQDn8/SKo
gGQJgk8QbAPN0:19607:0:99999:7:::
daemon*:18630:0:99999:7:::
bin*:18630:0:99999:7:::
sys*:18630:0:99999:7:::
sync*:18630:0:99999:7:::
games*:18630:0:99999:7:::
man*:18630:0:99999:7:::
lp*:18630:0:99999:7:::
mail*:18630:0:99999:7:::
news*:18630:0:99999:7:::
uucp*:18630:0:99999:7:::
proxy*:18630:0:99999:7:::
www-data*:18630:0:99999:7:::
backup*:18630:0:99999:7:::
list*:18630:0:99999:7:::
irc*:18630:0:99999:7:::
gnats*:18630:0:99999:7:::
nobody*:18630:0:99999:7:::
systemd-network*:18630:0:99999:7:::
systemd-resolve*:18630:0:99999:7:::
systemd-timesync*:18630:0:99999:7:::
messagebus*:18630:0:99999:7:::
syslog*:18630:0:99999:7:::
_apt*:18630:0:99999:7:::
_apt*:18630:0:99999:7:::
_apt*:18630:0:99999:7:::
rtkit*:18630:0:99999:7:::
systemd-coredump*:18630:0:99999:7:::
kernoops*:18630:0:99999:7:::
uuidd*:18630:0:99999:7:::
cups-pk-helper*:18630:0:99999:7:::
tcpdump*:18630:0:99999:7:::
geoclue*:18630:0:99999:7:::
avahi-autoipd*:18630:0:99999:7:::
usbmux*:18630:0:99999:7:::
dnsmasq*:18630:0:99999:7:::
_flatpak*:18630:0:99999:7:::
_apt*:18630:0:99999:7:::
_apt*:18630:0:99999:7:::
_apt*:18630:0:99999:7:::

```

Рис. 3.7: etc/shadow

Далее поработали с Sticky-битом

Создала файл, добавила права остальным пользователям.

```

root@szafgvsdg-HP-Pavilion-17-Notebook-PC:/# ls
bin  cdrom  etc  lab02  lib32  libx32  media  opt  root  sbin  swapfile  tmp  var
boot  dev  home  lib  lib64  lost+found  mnt  proc  run  srv  sys  usr
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:/# ls -l / | grep tmp
drwxrwxrwt  21 root root    4096 окт  7 16:10 tmp
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:/# echo "test" > /tmp/file01.txt
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:/# ls -l /tmp/file01.txt
-rw-r--r--  1 root root  5 окт  7 16:14 /tmp/file01.txt
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:/# chmod o+rw /tmp/file01.txt
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:/# ls -l /tmp/file01.txt
-rw-r--rw-  1 root root  5 окт  7 16:14 /tmp/file01.txt
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:/#

```

Рис. 3.8: Проверка функциональности Sticky-бита на примере файла, созданного в каталоге /tmp

От второго гостя попробовала записать в файл текст, удалить файл. Потом убрала атрибут `t` и сделала все то же самое. (см. рис. @fig:008)

```

iaakopyan@szafgvsdg-HP-Pavilion-17-Notebook-PC:~$ su guest2
Password:
$ echo "test2" > tmp/file01.txt
sh: 1: cannot create tmp/file01.txt: Directory nonexistent
$ cat tmp/file01.txt
cat: tmp/file01.txt: No such file or directory
$ echo "test3" > /tmp/file01.txt
$ cat tmp/file01.txt
cat: tmp/file01.txt: No such file or directory
$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': Operation not permitted
$ su -
Password:
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:~# chmod -t /tmp
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:~# exit
logout
$ ls -l / | grep tmp
drwxrwxrwx  21 root root      4096 окт  7 16:14 tmp
$ echo "test2" > /tmp/file01.txt
sh: 8: echotest2: not found
$ echo "test2" > /tmp/file01.txt
$ cat tmp/file01.txt
cat: tmp/file01.txt: No such file or directory
$ rm /tmp/file01.txt
$ ls
13                '2023-09-30 15-13-40.mkv'
1314.zip          '2023-09-30 18-35-15.mkv'
13.zip            '2023-09-30 18-38-25.mkv'
14                '2023-10-07 14-01-18.mkv'
14.zip            '2023-10-07 15-34-09.mkv'
'2023-09-05 13-56-42.mkv' 4
'2023-09-05 14-29-45.mkv' 4.sage
'2023-09-07 16-03-30.mkv' 5
'2023-09-07 16-05-23.mkv' 6
'2023-09-07 16-12-25.mkv' 6.zip
'2023-09-07 18-01-22.mkv' 7
'2023-09-08 15-02-40.mkv' 8
'2023-09-08 17-00-38.mkv' cat.jpeg
'2023-09-08 17-17-17.mkv' euler1.sage
'2023-09-09 18-23-56.mkv' euler2.sage
'2023-09-09 18-28-06.mkv' euler4.sage
'2023-09-13 12-07-20.mkv' euler.sage
'2023-09-16 12-48-09.mkv' file.txt
'2023-09-16 14-06-49.mkv' filw

```

Рис. 3.9: атрибут t убран

Вернула все на место (см. рис. @fig:009)

```
$ su -  
Password:  
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:~# chmod +t /tmp  
root@szafgvsdg-HP-Pavilion-17-Notebook-PC:~# exit  
logout  
$ ls -l / | grep tmp  
drwxrwxrwt 21 root root      4096 окт  7 16:22 tmp  
#
```

Рис. 3.10: атрибут t добавлен

4 Выводы

Изучили теорию механизмов изменения идентификаторов, применения SetUID, SetGID и Sticky-битов. Рассмотрела работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

Список литературы

Лабораторная работа № 5. Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов. [Электронный ресурс]. https://esystem.rudn.ru/pluginfile.php/2090417/mod_resource/content/2/005-lab_discret_sticky.pdf

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