

Лабораторная работа №5

Царитова Нина НПМбд-01-19

Российский университет дружбы народов, Москва, Россия

Изучить особенности работы с дополнительными атрибутами SetUID, SetGID и Sticky битами и их влияние на работу с файлами при их наличии и отсутствии.

Выполнение лабораторной работы

Создадим программу simpleid.c и скомпилируем ее с помощью команды gcc и убеждаемся, что файл действительно создан. Далее запускаем исполняемый файл через ./ . Вывод написанной программы совпадает с выводом команды id

```
[n@n ~]$ su - guest
Пароль:
Последний вход в систему:Сб окт  1 15:27:55 MSK 2022на pts/3
[guest@n ~]$ touch simpleid.c
[guest@n ~]$ 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;
}
[guest@n ~]$ gcc simpleid.c -o simpleid
[guest@n ~]$ ./simpleid
-bash: ./simpleid: Нет такого файла или каталога
[guest@n ~]$ ./simpleid
uid=1001, gid=1001
[guest@n ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Усложним программу, скомпилируем и запустим (файл simpleid2)

```
[guest@n ~]$ touch simpleid2.c
[guest@n ~]$ cat 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_u
id, real_gid); return 0;
}
[guest@n ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:uncon
fined_t:s0-s0:c0.c1023
[guest@n ~]$ gcc simpleid2.c -o simpleid2
[guest@n ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
```

От имени суперпользователя сменим владельца файла simpleid2 на root и установим SetUID-бит.

```
[n@n ~]$ sudo chown root:guest /home/guest/simpleid2  
[n@n ~]$ sudo chmod u+s /home/guest/simpleid2
```

guest@n:~

Файл Правка Вид Поиск Терминал Справка

```
[guest@n ~]$ ls -l simpleid2  
-rwsrwxr-x. 1 root guest 8616 окт  1 15:30 simpleid2
```

Запускаем программу simpleid2 и команду id. Появились отличия в uid строках

```
[guest@n ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@n ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Продолжаем выше описанные действия для SetGID-бита. Появились отличия в gid строках

```
[n@n ~]$ sudo chmod g+s /home/guest/simpleid2
```

```
guest@n:~
```

Файл Правка Вид Поиск Терминал Справка

```
[guest@n ~]$ ./simpleid2
```

```
e_uid=0, e_gid=1001
```

```
real_uid=1001, real_gid=1001
```


Создадим программу `readfile.c` и откомпилируем эту программу командой `gcc`. Меняем владельца файла `readfile.c` и отнимаем у пользователя `guest` право на чтение. При попытке прочитать файл от имени пользователя `guest` возникает ошибка

```
[n@n ~]$ sudo chmod 700 /home/guest/readfile.c
[n@n ~]$ sudo chmod g+s /home/guest/simpleid2
[n@n ~]$ sudo chown root:guest /home/guest/readfile.c
```

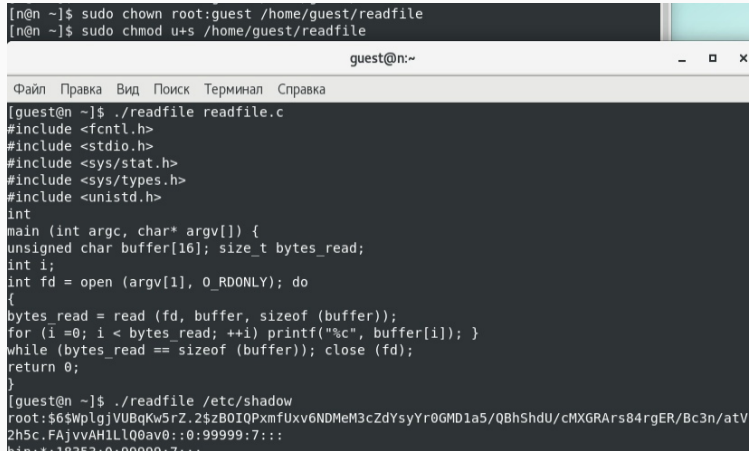
guest@n:~

Файл Правка Вид Поиск Терминал Справка

```
[guest@n ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
```

Меняем владельца файла readfile и устанавливаем на него SetUID-бит. Запускаем исполняемый файл и убеждаемся, что программа может прочитать файлы readfile.c и /etc/shadow

```
[n@n ~]$ sudo chown root:guest /home/guest/readfile
[n@n ~]$ sudo chmod u+s /home/guest/readfile
```



The screenshot shows a terminal window titled 'guest@n:~'. The window has a menu bar with 'Файл', 'Правка', 'Вид', 'Поиск', 'Терминал', and 'Справка'. The terminal content shows the user running a C program named 'readfile.c'. The program includes headers for `<fcntl.h>`, `<stdio.h>`, `<sys/stat.h>`, `<sys/types.h>`, and `<unistd.h>`. It defines an integer `i` and a `main` function that takes `argc` and `argv`. Inside `main`, it declares a `buffer` of size 16 and a `bytes_read` variable. It opens the file specified in `argv[1]` in read-only mode and enters a loop that reads the file into the buffer and prints each character. The program is then run with `./readfile /etc/shadow`. The output shows the root user's password hash, indicating successful access due to the SetUID bit.

```
[guest@n ~]$ ./readfile 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;
}
[guest@n ~]$ ./readfile /etc/shadow
root:$6$WplgjVUBqKw5rZ.2$zB0IQPxmFUXv6NDMeM3cZdYsyYr0GMD1a5/QBhShdU/cMXGRars84rgER/Bc3n/atV
2h5c.FAjvVAH1LlQ0av0::0:99999:7:::
bin:*.12352:0:00000:7:::
```

Исследование Sticky-бита

Выполняя команду `ls -l` выявняем, что на каталоге `/tmp` установлен Sticky-бит. Это видно, т.к. в конце написана `t`. Далее от имени пользователя `guest` создаём файл `/tmp/file01.txt`. Потом просматриваем атрибуты только что созданного файла и даём всем пользователям право на чтение и запись

```
[n@n ~]$ ls -l / | grep tmp
drwxrwxrwt. 51 root root 8192 окт  1 15:41 tmp
```

guest@n:~

Файл Правка Вид Поиск Терминал Справка

```
[n@n ~]$ su - guest
Пароль:
Последний вход в систему:Сб окт  1 15:28:54 MSK 2022на pts/2
[guest@n ~]$ echo "test" > /tmp/file01.txt
[guest@n ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 окт  1 15:46 /tmp/file01.txt
[guest@n ~]$ chmod o+rw /tmp/file01.txt
[guest@n ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 окт  1 15:46 /tmp/file01.txt
```

От имени пользователя guest2 читаем файл file01.txt командой cat. Повторяем предыдущие шаги. При попытке удалить файл возникла ошибка.

```
[n@n ~]$ su - guest2
Пароль:
Последний вход в систему:Вт сен 27 00:27:54 MSK 2022на pts/1
[guest2@n ~]$ cat /tmp/file01.txt
test
[guest2@n ~]$ echo "test2" > /tmp/file01.txt
[guest2@n ~]$ cat /tmp/file01.txt
test2
[guest2@n ~]$ echo "test3" > /tmp/file01.txt
[guest2@n ~]$ cat /tmp/file01.txt
test3
[guest2@n ~]$ rm /tmp/file01.txt
rm: невозможно удалить «/tmp/file01.txt»: Нет такого файла или каталога
```

Повышаем права до суперпользователя и снимаем с директории /tmp Sticky-бит. Покидаем режим суперпользователя командой exit

```
[n@n ~]$ su -  
Пароль:  
Последний вход в систему:Сб окт  1 15:49:00 MSK 2022на pts/3  
[root@n ~]# chmod +t /tmp  
[root@n ~]# exit  
logout  
[n@n ~]$ sudo chmod g+s /home/guest/simpleid2  
[n@n ~]$ ls -l / | grep tmp  
drwxrwxrwt. 49 root root 8192 окт  1 16:10 tmp
```

Исследование Sticky-бита

Убеждаемся через команду `ls -l`, что Sticky-бит действительно отсутствует. Далее повторяем действия от имени пользователя `guest2`, описанные выше. В этот раз удалось удалить файл `file01.txt` даже при условии, что `guest2` не является его владельцем

```
[guest2@n ~]$ ls -l / | grep tmp
drwxrwxrwx. 51 root root 8192 окт  1 15:49 tmp
[guest2@n ~]$ cat /tmp/file01.txt
test3
[guest2@n ~]$ echo "test2" >> /tmp/file01.txt
[guest2@n ~]$ cat /tmp/file01.txt
test3
test2
[guest2@n ~]$ echo "test3" > /tmp/file01.txt
[guest2@n ~]$ cat /tmp/file01.txt
test3
[guest2@n ~]$ rm /tmp/file01.txt
rm: невозможно удалить «/tmp/file01.txt»: Нет такого файла или каталога
[guest2@n ~]$ rm /tmp/file01.txt
rm: невозможно удалить «/tmp/file01.txt»: Нет такого файла или каталога
[guest2@n ~]$ ls /tmp | grep *.tmp
[guest2@n ~]$ ls /tmp | grep file01.tmp
```

Повышаем права до суперпользователя и возвращает Sticky-бит на каталог /tmp

```
[n@n ~]$ su -  
Пароль:  
Последний вход в систему:Сб окт  1 15:49:00 MSK 2022на pts/3  
[root@n ~]# chmod +t /tmp  
[root@n ~]# exit  
logout  
[n@n ~]$ sudo chmod g+s /home/guest/simpleid2  
[n@n ~]$ ls -l / | grep tmp  
drwxrwxrwt. 49 root root 8192 окт  1 16:10 tmp
```

Выводы

Изучила механизмы изменения идентификаторов и получила практические навыки по работе с SetUID, SetGID и Sticky битами и узнала об их особенностях и влиянии на файлы и директории.