

Отчет по лабораторной работе №5

**Дискреционное разграничение прав в Linux. Исследование влияния
дополнительных атрибутов**

Татьяна Александровна Буллер

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
2.1	Компилирование программ	5
2.2	Исследование SUID-бита	7
2.3	Исследование Sticky-бита	11
3	Выводы	14

Список иллюстраций

2.1	simpleid.c	5
2.2	Скомпилированная программа	6
2.3	simpleid и id	6
2.4	simpleid2	6
2.5	Дополненный вывод	7
2.6	Переназначение владельца и добавление SUID-бита	7
2.7	Новые права программы	7
2.8	Вывод simpleid2 с SUID-битом	8
2.9	readfile	8
2.10	Изменение прав файла кода	9
2.11	Пользователь без прав	9
2.12	Передача суперпользователю	10
2.13	Добавление SUID-бита и новые права	10
2.14	Чтение файлов через readfile	11
2.15	Sticky-бит на директории tmp	11
2.16	Созданный в директории tmp файл	12
2.17	Работа с файлом от лица стороннего пользователя	12
2.18	Снятие Sticky-бита	13
2.19	Работа с файлом от лица стороннего пользователя	13

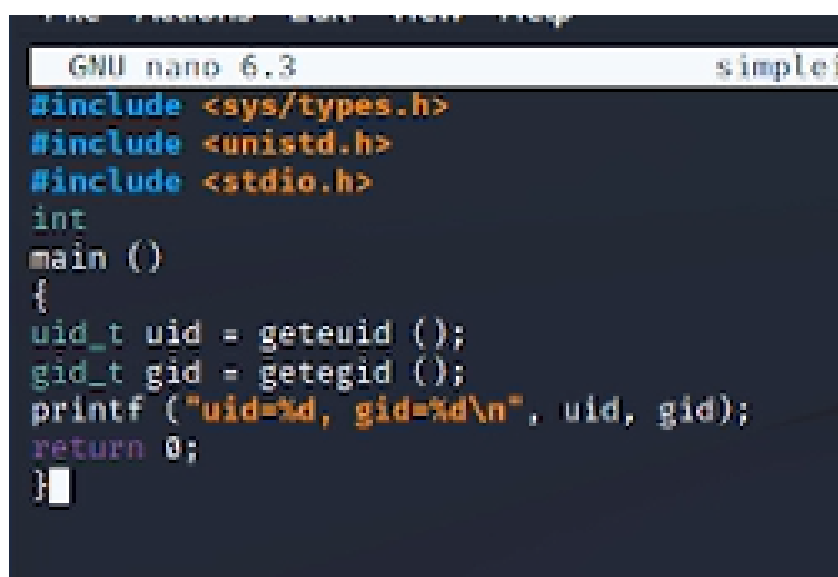
1 Цель работы

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 Выполнение лабораторной работы

2.1 Компилирование программ

Создадим программу simpleid.c. Эта программа с помощью функций `geteuid` и `getegid` получает `uid` и `gid` пользователя соответственно, после чего выводит их на экран.

A screenshot of a terminal window with a dark background. The title bar at the top reads "GNU nano 6.3" on the left and "simplei" on the right. The code is written in a color-coded font: blue for preprocessor directives, green for keywords, and white for identifiers and literals. The code is as follows:

```
#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. 2.1: simpleid.c

Скомпилировав программу, получим вывод, в значениях совпадающий с выводом команды `id`.

```
(guest@jordi)-[~]
$ gcc simpleid.c -o simpleid

(guest@jordi)-[~]
$ ./simpleid
uid=1002, gid=1002
```

Рис. 2.2: Скомпилированная программа

```
(guest@jordi)-[~]
$ ./simpleid
uid=1002, gid=1002

(guest@jordi)-[~]
$ id
uid=1002(guest) gid=1002(guest) groups=1002(guest)
```

Рис. 2.3: simpleid и id

После этого усложним программу, как показано на скриншоте:

```
GNU nano 6.3 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_uid, real_gid);
    return 0;
}
```

Рис. 2.4: simpleid2

Теперь вывод дополнен и все еще совпадает с выводом id.

```

(guest@jordi)-[~]
$ gcc simpleid2.c -o simpleid2

(guest@jordi)-[~]
$ ./simpleid2
e_uid=1002, e_gid=1002
real_uid=1002, real_gid=1002

```

Рис. 2.5: Дополненный вывод

2.2 Исследование SUID-бита

Следующим шагом от имени суперпользователя назначим владельцам файла программы суперпользователя и добавим ей SUID-бит.

```

(tabuller@jordi)-[/home/guest]
$ sudo chown tabuller:guest /home/guest/simpleid2
[sudo] password for tabuller:

(tabuller@jordi)-[/home/guest]
$ sudo chmod u+s /home/guest/simpleid2

```

Рис. 2.6: Переназначение владельца и добавление SUID-бита

```

(guest@jordi)-[~]
$ ls -l simpleid2
-rwsr-xr-x 1 tabuller guest 16168 Feb 14 08:05 simpleid2

```

Рис. 2.7: Новые права программы

Теперь при запуске этой программы видим, что она выводит `e_uid`: идентификатор пользователя, от имени которого она была запущена; `real_uid` - идентификатор пользователя, от имени которого она выполняется.

```
(tabuller@jordi)-[/home/guest]
$ sudo ./simpleid2
e_uid=1001, e_gid=0
real_uid=0, real_gid=0

(tabuller@jordi)-[/home/guest]
$ sudo id
uid=0(root) gid=0(root) groups=0(root)
```

Рис. 2.8: Вывод simpleid2 с SUID-битом

Создадим еще одну программу: аналог cat readfile, которая будет получать содержимое файла, название которого передано ей в аргументе, и выводить его на экран.

```
GNU nano 6.3      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. 2.9: readfile

Скомпилируем ее, передадим файл кода во владение суперпользователю и добавим SUID-бит, после чего заберем у всех остальных пользователей все права на него. Видим, что теперь пользователь guest не может прочитать содержимое.


```

(guest@jordi)-[~]
$ gcc readfile.c -o readfile

(guest@jordi)-[~]
$ su tabuller
Password:
(tabuller@jordi)-[/home/guest]
$ sudo chown root:guest /home/guest/readfile.c

(tabuller@jordi)-[/home/guest]
$ sudo chmod u+s /home/guest/readfile.c

```

Рис. 2.10: Изменение прав файла кода

```

(tabuller@jordi)-[/home/guest]
$ sudo chmod ga-rwx /home/guest/readfile.c

(tabuller@jordi)-[/home/guest]
$ exit
exit

(guest@jordi)-[~]
$ cat readfile.c
cat: readfile.c: Permission denied

```

Рис. 2.11: Пользователь без прав

Следующим шагом изменим права на программу, которая была скомпилирована по коду `readfile.c`. Добавим тот же SUID-бит и передадим во владение суперпользователю. Вилем, что теперь название программы подсвечено красным.

```
(tabuller@jordi)-[/home/guest]
$ sudo chown root:guest readfile
[sudo] password for tabuller:

(tabuller@jordi)-[/home/guest]
$ ls -l readfile
ls: cannot access 'readfile': Permission denied

(tabuller@jordi)-[/home/guest]
$ exit
exit

(guest@jordi)-[~]
$ ls -l readfile
-rwxr-xr-x 1 root guest 16104 Feb 14 08:11 readfile
```

Рис. 2.12: Передача суперпользователю

```
(tabuller@jordi)-[/home/guest]
$ sudo chmod u+s readfile

(tabuller@jordi)-[/home/guest]
$ exit
exit

(guest@jordi)-[~]
$ ls -l readfile
-rwsr-xr-x 1 root guest 16104 Feb 14 08:11 readfile
```

Рис. 2.13: Добавление SUID-бита и новые права

Теперь при попытке прочитать readfile.c с помощью скомпилированной им программы мы получаем содержимое файла. Это происходит потому, что программа выполняется от имени суперпользователя и наделена всеми теми же правами, что и root. Она в том числе может прочитать и системный файл /etc/shadow, в котором по умолчанию хранится список пользователей и их хэшированные пароли.

```

(guest@jordi)-[~]
$ ./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@jordi)-[~]
$ ./readfile /etc/shadow
root!:20120:0:99999:7:::
daemon*:20120:0:99999:7:::

```

Рис. 2.14: Чтение файлов через readfile

2.3 Исследование Sticky-бита

Просмотрев в корневой директории системы права поддиректорий, видим, что на директории tmp установлен бит t. Полезно отметить также, что писать и читать эту директорию может кто угодно.

```

(guest@jordi)-[~]
$ ls -l / |grep tmp
drwxrwxrwt 13 root root 300 Feb 14 08:11 tmp

```

Рис. 2.15: Sticky-бит на директории tmp

Далее создадим в этой директории файл file01. Изначально права на него позволяют пользователям кроме владельца только читать его, владельцу - записывать и читать. Добавим для остальных пользователей права на запись.

```

(guest@ jordi)-[~]
$ echo "test" > /tmp/file01.txt

(guest@ jordi)-[~]
$ ls -l /tmp/file01.txt
-rw-r--r-- 1 guest guest 5 Feb 14 08:18 /tmp/file01.txt

(guest@ jordi)-[~]
$ chmod o+rw /tmp/file01.txt

(guest@ jordi)-[~]
$ ls -l /tmp/file01.txt
-rw-r--rw- 1 guest guest 5 Feb 14 08:18 /tmp/file01.txt

```

Рис. 2.16: Созданный в директории tmp файл

Переключившись на пользователя guest2, однако, мы все еще не можем сделать с этим файлом ничего, кроме прочтения, потому что пользователь guest2 входит в группу пользователя guest, а группе не были добавлены права на запись. Файл, кроме прочего, “защищен” Sticky-битом, установленным на директории.

```

(guest2@ jordi)-[/home/guest]
$ echo "test2" >> /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied

(guest2@ jordi)-[/home/guest]
$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied

(guest2@ jordi)-[/home/guest]
$ cat /tmp/file01.txt
test

(guest2@ jordi)-[/home/guest]
$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory

(guest2@ jordi)-[/home/guest]
$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'? y
rm: cannot remove '/tmp/file01.txt': Operation not permitted

```

Рис. 2.17: Работа с файлом от лица стороннего пользователя

От имени суперпользователя снимем Sticky-бит и снова проверим права на директорию tmp: символ t пропал.

```

(root@jordi)-[/home/guest]
# chmod -t /tmp

(root@jordi)-[/home/guest]
# exit

(tabuller@jordi)-[/home/guest]
$ exit
exit

(guest2@jordi)-[/home/guest]
$ ls -l / | grep tmp
drwxrwxrwx 13 root root 320 Feb 14 08:18 tmp

```

Рис. 2.18: Снятие Sticky-бита

Теперь уже, все еще не являясь владельцем файла, мы можем его переименовать и удалять.

```

(guest2@jordi)-[/home/guest]
$ echo "test2" >> /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied

(guest2@jordi)-[/home/guest]
$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied

(guest2@jordi)-[/home/guest]
$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'? y

```

Рис. 2.19: Работа с файлом от лица стороннего пользователя

3 Выводы

Изучены механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получены практические навыки работы в консоли с дополнительными атрибутами. Рассмотрена работа механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.