

Лабораторная работа №5

Информационная безопасность

Банникова Екатерина Алексеевна

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	7
3.1	Создание программы	7
3.2	Исследование Sticky-бита	10
4	Выводы	13
	Список литературы	14

Список иллюстраций

3.1	Текст программы simpleid.c	7
3.2	Компиляция и запуск simpleid	7
3.3	Текст программы simpleid2.c	8
3.4	Компиляция и запуск simpleid2	8
3.5	Смена владельца и установка SetUID	8
3.6	Запуск simpleid2	8
3.7	SetUID-бит	9
3.8	Текст программы readlife.c	9
3.9	Компиляция readlife.c	9
3.10	Компиляция readlife.c	9
3.11	Запуск readlife	10
3.12	Запуск readlife	10
3.13	Создание файла file01	10
3.14	Создание файла file01	11
3.15	Действия над file01 от лица guest2	11
3.16	Удаление Sticky-бита	11
3.17	Повтор действий	12
3.18	Возвращение Sticky-бита	12

Список таблиц

1 Цель работы

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

2 Теоретическое введение

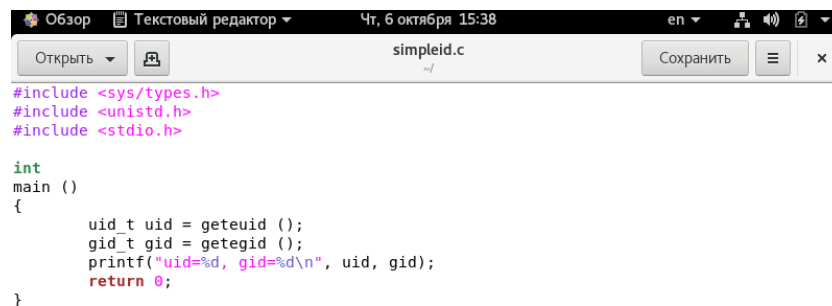
SetUID, SetGID и Sticky — это специальные типы разрешений, которые позволяют задавать расширенные права доступа на файлы и каталоги.

- SetUID — это бит разрешения, который позволяет пользователю запускать исполняемый файл с правами владельца этого файла. Другими словами, использование этого бита позволят поднять привилегии пользователя в случае, если это необходимо. Наличие SetUID бита выражается в том, что на месте классического бита x выставлен специальный бит s: -rwsr-xr-x
- SetGID — очень похож на SetUID с отличием, что файл будет запускаться от имени группы, который владеет файлом: -rwxr-sr-x
- Sticky — в случае, если этот бит установлен для папки, то файлы в этой папке могут быть удалены только их владельцем. Наличие этого бита показывается через букву t в конце всех прав: drwxrwxrwt

3 Выполнение лабораторной работы

3.1 Создание программы

Создадим программу simpleid.c

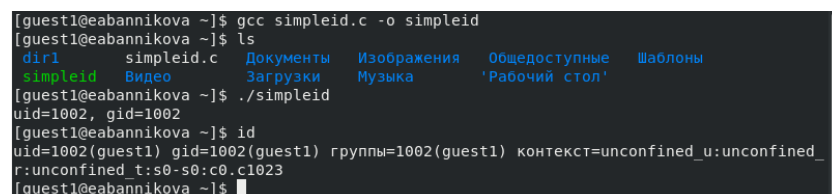


```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int
main ()
{
    uid_t uid = geteuid ();
    gid_t gid = getegid ();
    printf("uid=%d, gid=%d\n", uid, gid);
    return 0;
}
```

Рис. 3.1: Текст программы simpleid.c

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



```
[guest1@eabannikova ~]$ gcc simpleid.c -o simpleid
[guest1@eabannikova ~]$ ls
dir1  simpleid.c  Документы  Изображения  Общедоступные  Шаблоны
simpleid  Видео  Загрузки  Музыка  'Рабочий стол'
[guest1@eabannikova ~]$ ./simpleid
uid=1002, gid=1002
[guest1@eabannikova ~]$ id
uid=1002(guest1) gid=1002(guest1) группы=1002(guest1) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest1@eabannikova ~]$
```

Рис. 3.2: Компиляция и запуск simpleid

Усложним программу и назовём её 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. 3.3: Текст программы simpleid2.c

Скомпилируем и запустим файл simpleid2

```

[guest1@eabannikova ~]$ gcc simpleid2.c -o simpleid2
[guest1@eabannikova ~]$ ./simpleid2
e_uid=1002, e_gid=1002
real_uid=1002, real_gid=1002
[guest1@eabannikova ~]$

```

Рис. 3.4: Компиляция и запуск simpleid2

От имени суперпользователя сменим владельца файла simpleid2 на root и установим SetUID-бит. Далее через команду `ls -l` видим, что бит установлен корректно

```

[eabannikova@eabannikova ~]$ sudo chown root:guest1 /home/guest1/simpleid2
[sudo] пароль для eabannikova:
[eabannikova@eabannikova ~]$ sudo chmod u+s /home/guest1/simpleid2
[eabannikova@eabannikova ~]$ ls -l /home/guest1/simpleid2
ls: невозможно получить доступ к '/home/guest1/simpleid2': Отказано в доступе
[eabannikova@eabannikova ~]$ sudo ls -l /home/guest1/simpleid2
-rwsrwxr-x. 1 root guest1 11336 окт 6 15:43 /home/guest1/simpleid2
[eabannikova@eabannikova ~]$

```

Рис. 3.5: Смена владельца и установка SetUID

Запускаем программу simpleid2 и команду `id`. Теперь видим, что появились отличия в `uid` строках

```

[guest1@eabannikova ~]$ ./simpleid2
e_uid=0, e_gid=1002
real_uid=1002, real_gid=1002
[guest1@eabannikova ~]$ id
uid=1002(guest1) gid=1002(guest1) группы=1002(guest1) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest1@eabannikova ~]$

```

Рис. 3.6: Запуск simpleid2

Продолываем выше описанные действия для SetGID-бита. Теперь после запуска simpleid2 можем увидеть отличие и в `gid` строках


```
[eabannikova@eabannikova ~]$ sudo chmod g+s /home/guest1/simpleid2
[eabannikova@eabannikova ~]$ su - guest1
Пароль:
[guest1@eabannikova ~]$ ./simpleid2
e uid=0, e gid=1002
real_uid=1002, real_gid=1002
[guest1@eabannikova ~]$ id
uid=1002(guest1) gid=1002(guest1) группы=1002(guest1) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest1@eabannikova ~]$
```

Рис. 3.7: SetUID-бит

Создадим программу 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. 3.8: Текст программы readlife.c

Откомпилируем эту программу командой gcc. Далее меняем владельца файла readfile.c и отнимаем у пользователя guest право на чтение. При попытке прочитав файл от имени пользователя guest возникает ошибка

```
[eabannikova@eabannikova ~]$ sudo chown root:guest1 /home/guest1/readfile.c
[sudo] пароль для eabannikova:
[eabannikova@eabannikova ~]$ sudo chmod 700 /home/guest1/readfile.c
[eabannikova@eabannikova ~]$
```

Рис. 3.9: Компиляция readlife.c

```
[guest1@eabannikova ~]$ gcc readfile.c -o readfile
[guest1@eabannikova ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest1@eabannikova ~]$
```

Рис. 3.10: Компиляция readlife.c

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

```
[eabannikova@eabannikova ~]$ sudo chown root:guest1 /home/guest1/readfile
[eabannikova@eabannikova ~]$ sudo chmod u+s /home/guest1/readfile
[eabannikova@eabannikova ~]$
```

Рис. 3.11: Запуск readlife

```
[guest1@eabannikova ~]$ ./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;
}
[eabannikova@eabannikova ~]$ ./readfile /etc/shadow
root:$6$bKjXi0r8GrFcTPpt$qHRwtuIhfVvXwHOX/TB52QX4Sig2eTiwrZGUBku2eMZVxhaWztJmk80dYdNKRn
TNAODiGi/ZHW0zqYIRVqPsF/::0:99999:7:::
bin:*.18078:0:99999:7:::
```

Рис. 3.12: Запуск readlife

3.2 Исследование Sticky-бита

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

```
[eabannikova@eabannikova ~]$ ls -l / | grep tmp
drwxrwxrwt. 23 root root 4096 окт 6 16:15 tmp
[eabannikova@eabannikova ~]$
```

Рис. 3.13: Создание файла file01

```
[guest1@eabannikova ~]$ echo "test" > /tmp/file01.txt
[guest1@eabannikova ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest1 guest1 6 окт  6 16:16 /tmp/file01.txt
[guest1@eabannikova ~]$ chmod o+rw /tmp/file01.txt
[guest1@eabannikova ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest1 guest1 6 окт  6 16:16 /tmp/file01.txt
[guest1@eabannikova ~]$
```

Рис. 3.14: Создание файла file01

От имени пользователя guest2 читаем файл file01.txt командой cat. Далее успешно дозаписываем в конец файла строку “test2”, а затем успешно перезаписываем содержимое, меняя его на строку “test3”. Однако при попытке удалить файл возникла ошибка

```
[guest2@eabannikova ~]$ cat /tmp/file01.txt
test
[guest2@eabannikova ~]$ echo "test2" > /tmp/file01.txt
[guest2@eabannikova ~]$ cat /tmp/file01.txt
test2
[guest2@eabannikova ~]$ echo "test3" > /tmp/file01.txt
[guest2@eabannikova ~]$ cat /tmp/file01.txt
test3
[guest2@eabannikova ~]$ rm /tmp/file01.txt
rm: невозможно удалить '/tmp/file01.txt': Нет такого файла или каталога
[guest2@eabannikova ~]$
```

Рис. 3.15: Действия над file01 от лица guest2

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

```
[eabannikova@eabannikova ~]$ su -
Пароль:
[root@eabannikova ~]# chmod -t /tmp
[root@eabannikova ~]# exit
выход
[eabannikova@eabannikova ~]$
```

Рис. 3.16: Удаление 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@eabannikova ~]$ ls -l / | grep tmp
drwxrwxrwx. 23 root root  4096 окт 6 16:21 tmp
[guest2@eabannikova ~]$ cat /tmp/file01.txt
test3
[guest2@eabannikova ~]$ echo "test2" > /tmp/file01.txt
[guest2@eabannikova ~]$ cat /tmp/file01.txt
test2
[guest2@eabannikova ~]$ echo "test3" > /tmp/file01.txt
[guest2@eabannikova ~]$ cat /tmp/file01.txt
test3
[guest2@eabannikova ~]$ rm /tmp/file01.txt
[guest2@eabannikova ~]$ ls /tmp | grep *.txt
[guest2@eabannikova ~]$ ls /tmp | grep file01.txt
[guest2@eabannikova ~]$

```

Рис. 3.17: Повтор действий

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

```

[eabannikova@eabannikova ~]$ su -
Пароль:
[root@eabannikova ~]# chmod +t /tmp
[root@eabannikova ~]# exit
выход
[eabannikova@eabannikova ~]$ ls -l / | grep tmp
drwxrwxrwt. 23 root root  4096 окт 6 16:24 tmp
[eabannikova@eabannikova ~]$

```

Рис. 3.18: Возвращение Sticky-бита

4 Выводы

Я 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.

Список литературы