

Отчет по лабораторной работе №5

Информационная безопасность

Чекалова Лилия Руслановна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	15
	Список литературы	16

Список таблиц

Список иллюстраций

4.1	Создание файла	8
4.2	Программа 1	8
4.3	Компиляция программы 1	9
4.4	Запуск программы 1	9
4.5	Программа 2	9
4.6	Запуск модифицированной программы	10
4.7	Передача прав и перезапуск программы	10
4.8	Повтор действий относительно SetGID-бита	10
4.9	Программа 3	11
4.10	Смена владельца файла	11
4.11	Смена прав и проверка	11
4.12	Смена владельца программой и чтение readfile.c	12
4.13	Чтение etc/shadow	12
4.14	Проверка наличия Sticky-бита и создание файла	12
4.15	Попытка чтения и изменения	13
4.16	Попытка удаления	13
4.17	Снятие атрибута Sticky	13
4.18	Чтение, запись и удаление	14
4.19	Установка атрибута Sticky	14

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 Задание

- Написание программ
- Изменение владельца файлов и прав доступа на файлы
- Установка и снятие Sticky-бита и проверка доступных действий

3 Теоретическое введение

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. Классический пример использования этого бита в операционной системе это команда `sudo`.

На месте, где обычно установлен классический бит `x` (на исполнение), выставлен специальный бит `s`. Это позволяет обычному пользователю системы выполнять команды с повышенными привилегиями без необходимости входа в систему как `root`, зная пароль пользователя `root`.

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. Пример использования этого бита в операционной системе это системная папка `/tmp` . Эта папка разрешена на запись любому пользователю, но удалять файлы в ней могут только пользователи, являющиеся владельцами этих файлов.

Более подробно о см. в [1,2].

4 Выполнение лабораторной работы

В качестве первого шага лабораторной работы мы осуществили вход от лица пользователя guest и создали файл simpleid.c (рис. 4.1).

```
[lrchekalova@lrchekalova guest]$ su guest
Пароль:
[guest@lrchekalova ~]$ touch simpleid.c
[guest@lrchekalova ~]$
```

Рис. 4.1: Создание файла

Далее мы написали программу, которая выводит информацию об идентификаторах пользователя и группы (рис. 4.2). Скомпилировали эту программу (рис. 4.3).

```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4
5 int
6 main()
7 {
8     uid_t uid = geteuid();
9     gid_t gid = getegid();
10    printf("uid=%d, gid=%d\n", uid, gid);
11    return 0;
12 }
```

Рис. 4.2: Программа 1


```
[guest@lrchekalova ~]$ gcc simpleid.c -o simpleid
[guest@lrchekalova ~]$ ls
dir1  simpleid  simpleid.c
[guest@lrchekalova ~]$
```

Рис. 4.3: Компиляция программы 1

Выполнили эту программу и сравнили ее вывод с результатом работы системной команды `id` (рис. 4.4). Результаты сходятся, но команда `id` дает дополнительные сведения, например, название группы.

```
[guest@lrchekalova ~]$ ./simpleid
uid=1001, gid=1001
[guest@lrchekalova ~]$ id
uid=1001(guest) gid=1001(guest) rpnpy=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@lrchekalova ~]$
```

Рис. 4.4: Запуск программы 1

Модифицировали программу, чтобы она выводила также действительные идентификаторы (рис. 4.5).

```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4
5 int
6 main()
7 {
8     uid_t real_uid = getuid();
9     uid_t e_uid = geteuid();
10
11     gid_t real_gid = getgid();
12     gid_t e_gid = getegid();
13     printf("e_uid=%d, e_gid=%d\n", e_uid, e_gid);
14     printf("real_uid=%d, real_gid=%d\n", real_uid, real_gid);
15     return 0;
16 }
```

Рис. 4.5: Программа 2

Скомпилировали и запустили модифицированную программу (рис. 4.6).

```
[guest@lrchekalova ~]$ gcc simpleid2.c -o simpleid2
[guest@lrchekalova ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@lrchekalova ~]$
```

Рис. 4.6: Запуск модифицированной программы

Передали право владения программой суперпользователю и наделили его правом на исполнение программы. Запустили программу и сверили результат с выводом команды `id` (рис. 4.7). Теперь в качестве идентификаторов указана информация root-пользователя.

```
[guest@lrchekalova ~]$ su -
Пароль:
[root@lrchekalova ~]# chown root:guest /home/guest/simpleid2
[root@lrchekalova ~]# chmod u+s /home/guest/simpleid2
[root@lrchekalova ~]# ls -l simpleid2
ls: невозможно получить доступ к 'simpleid2': Нет такого файла или каталога
[root@lrchekalova ~]# ls -l /home/guest/simpleid2
-rwsr-xr-x. 1 root guest 26064 окт  5 11:43 /home/guest/simpleid2
[root@lrchekalova ~]# ./simpleid2
-bash: ./simpleid2: Нет такого файла или каталога
[root@lrchekalova ~]# cd /home/guest
[root@lrchekalova guest]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@lrchekalova guest]# id
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@lrchekalova guest]#
```

Рис. 4.7: Передача прав и перезапуск программы

Проделали то же самое относительно SetGID-бита (рис. 4.8).

```
[root@lrchekalova guest]# chown root:root /home/guest/simpleid2
[root@lrchekalova guest]# chmod g+s /home/guest/simpleid2
[root@lrchekalova guest]# ls -l simpleid2
-rwxr-sr-x. 1 root root 26064 окт  5 11:43 simpleid2
```

Рис. 4.8: Повтор действий относительно SetGID-бита

Создали новую программу `readfile`, позволяющую прочесть содержимое файла (рис. 4.9).

```

1 #include <fcntl.h>
2 #include <stdio.h>
3 #include <sys/stat.h>
4 #include <sys/types.h>
5 #include <unistd.h>
6
7 int
8 main (int argc, char* argv[])
9 {
10     unsigned char buffer[16];
11     size_t bytes_read;
12     int i;
13
14     int fd = open(argv[1], O_RDONLY);
15     do
16     {
17         bytes_read = read(fd, buffer, sizeof(buffer));
18         for(i=0; i<bytes_read; ++i) printf("%c", buffer[i]);
19     }
20     while (bytes_read == sizeof(buffer));
21     close (fd);
22     return 0;
23 }

```

Рис. 4.9: Программа 3

Скомпилировали ее, а затем поменяли владельца файла readfile.c (рис. 4.10) и изменили права так, чтобы читать файл мог только суперпользователь. Проверили, может ли пользователь guest прочесть файл (рис. 4.11). Ему было отказано в доступе.

```

[root@lrchekalova guest]# chown root:root readfile.c
[root@lrchekalova guest]# chmod u+s readfile.c
[root@lrchekalova guest]# ls -l readfile.c
-rwSr--r--. 1 root root 456 окт  5 12:05 readfile.c

```

Рис. 4.10: Смена владельца файла

```

[root@lrchekalova ~]# chmod o-r /home/guest/readfile.c
[root@lrchekalova ~]# exit
выход
[guest@lrchekalova ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@lrchekalova ~]$

```

Рис. 4.11: Смена прав и проверка

Передали право владения программой суперпользователю и проверили, может ли программа прочитать readfile.c (рис. 4.12) и etc/shadow (рис. 4.13). Действия были выполнены успешно.

```
[guest@lrchekalova ~]$ su -
Пароль:
[root@lrchekalova ~]# chown root:root /home/guest/readfile
[root@lrchekalova ~]# chmod u+s /home/guest/readfile
[root@lrchekalova ~]# ./readfile readfile.c
-bash: ./readfile: Нет такого файла или каталога
[root@lrchekalova ~]# cd /home/guest
[root@lrchekalova guest]# ./readfile readfile.c
#include <fcntl.h>
#include <stdio.h>
```

Рис. 4.12: Смена владельца программой и чтение readfile.c

```
[root@lrchekalova guest]# ./readfile /etc/shadow
root:$6$HiPuTZPpU0s5jFtY$xjdDaoqwUV7zpkYu/Vqbtjq3y
bin:!:19469:0:99999:7:::
daemon:!:19469:0:99999:7:::
```

Рис. 4.13: Чтение etc/shadow

Проверили, установлен ли Sticky-бит на директории tmp, создали file01.txt и передали категории пользователей “все остальные” право на чтение и запись (рис. 4.14).

```
[guest@lrchekalova ~]$ ls -l / | grep tmp
drwxrwxrwt. 17 root root 4096 окт  5 12:29 tmp
[guest@lrchekalova ~]$ echo "test" > /tmp/file01.txt
[guest@lrchekalova ~]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 окт  5 12:30 /tmp/file01.txt
[guest@lrchekalova ~]$ chmod o+rw /tmp/file01.txt
[guest@lrchekalova ~]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 окт  5 12:30 /tmp/file01.txt
[guest@lrchekalova ~]$
```

Рис. 4.14: Проверка наличия Sticky-бита и создание файла

От лица пользователя guest2 (не являющегося владельцем файла) попробовали прочесть файл и записать туда новые данные (рис. 4.15), а также удалить файл (рис. 4.16). Мы

смогли прочесть файл, но не смогли изменить его и удалить, так как guest2 входит в группу guest, а не в группу “все остальные”.

```
[guest@lrchekalova ~]$ su guest2
Пароль:
[guest2@lrchekalova guest]$ cat /tmp/file01.txt
test
[guest2@lrchekalova guest]$ echo "test3" > /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@lrchekalova guest]$
```

Рис. 4.15: Попытка чтения и изменения

```
[guest2@lrchekalova guest]$ rm /tmp/file01.txt
rm: удалить защищённый от записи обычный файл '/tmp/file01.txt'? y
rm: невозможно удалить '/tmp/file01.txt': Операция не позволена
[guest2@lrchekalova guest]$
```

Рис. 4.16: Попытка удаления

Сняли с папки tmp атрибут Sticky (рис. 4.17).

```
[guest2@lrchekalova guest]$ su -
Пароль:
[root@lrchekalova ~]# chmod -t /tmp
[root@lrchekalova ~]# exit
выход
[guest2@lrchekalova guest]$ ls -l / | grep tmp
drwxrwxrwx. 17 root root 4096 окт  5 12:34 tmp
[guest2@lrchekalova guest]$
```

Рис. 4.17: Снятие атрибута Sticky

От лица пользователя guest2 попробовали прочесть файл, произвести в него запись и удалить (рис. 4.18). Удаление файла прошло успешно, так как с папки был снят атрибут, запрещавший удаление файлов всеми, кроме владельца.

```
[guest2@lrchekalova guest]$ cat /tmp/file01.txt
test
[guest2@lrchekalova guest]$ echo "test3" > /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@lrchekalova guest]$ rm /tmp/file01.txt
rm: удалить защищённый от записи обычный файл '/tmp/file01.txt'? y
[guest2@lrchekalova guest]$ ls /tmp
systemd-private-b33f02659fc546bdbbc78fc7c1990a8d8-chronyd.service-eBqpWs
systemd-private-b33f02659fc546bdbbc78fc7c1990a8d8-colord.service-te702B
systemd-private-b33f02659fc546bdbbc78fc7c1990a8d8-dbus-broker.service-TNybZb
systemd-private-b33f02659fc546bdbbc78fc7c1990a8d8-fwupd.service-DS9osC
systemd-private-b33f02659fc546bdbbc78fc7c1990a8d8-ModemManager.service-cHwCwQ
systemd-private-b33f02659fc546bdbbc78fc7c1990a8d8-power-profiles-daemon.service-njCTdm
systemd-private-b33f02659fc546bdbbc78fc7c1990a8d8-rtkit-daemon.service-2HYTg0
systemd-private-b33f02659fc546bdbbc78fc7c1990a8d8-switcheroo-control.service-TaUiYt
systemd-private-b33f02659fc546bdbbc78fc7c1990a8d8-systemd-logind.service-2zSk8l
systemd-private-b33f02659fc546bdbbc78fc7c1990a8d8-upower.service-vpDYbo
tracker-extract-3-files.1000
vboxguest-Module.symvers
```

Рис. 4.18: Чтение, запись и удаление

Установили атрибут Sticky обратно на папку tmp (рис. 4.19).

```
[guest2@lrchekalova guest]$ su -
Пароль:
[root@lrchekalova ~]# chmod +t /tmp
[root@lrchekalova ~]# exit
выход
[guest2@lrchekalova guest]$
```

Рис. 4.19: Установка атрибута Sticky

5 Выводы

В результате лабораторной работы я повысила навыки использования командой строки и изучила SetUID-, SetGID- и Sticky-биты. Также я рассмотрела работу механизма смены идентификатора процессов пользователей.

Список литературы

1. Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов [Электронный ресурс]. URL: <https://esystem.rudn.ru/mod/resource/view.php?id=1031377>.
2. Использование SETUID, SETGID и Sticky Bit для расширенной настройки прав доступа в операционных системах Linux [Электронный ресурс]. URL: <https://ruvds.com/ru/helpcenter/suid-sgid-sticky-bit-linux/>.