Лабораторная работа №5

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Панченко Денис Дмитриевич

Содержание

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 Задачи

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

3 Выполнение лабораторной работы

3.1 Создание программы

1) Войдем в систему от имени пользователя guest (рис. 3.1).

```
[ddpanchenko@derenchik ~]$ su guest
Password:
[guest@derenchik ddpanchenko]$ cd
```

Рис. 3.1: Вход

2) Создадим программу simpleid.c (рис. 3.2 - 3.3).

```
[guest@derenchik ~]$ touch simpleid.c
[guest@derenchik ~]$ nano simpleid.c
```

Рис. 3.2: Программа

```
  GNU nano 5.6.1
#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. 3.3: Программа

3) Скомпилируем программу (рис. 3.4).

```
[guest@derenchik ~]$ gcc simpleid.c -o simpleid
```

Рис. 3.4: Компиляция

4) Выполним команду ./simpleid. После выполним команду id и сравним их (рис. 3.5).

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

Рис. 3.5: Команда

5) Создадим программу simpleid2.c (рис. 3.6 - 3.7).

```
[guest@derenchik ~]$ touch simpleid2.c
[guest@derenchik ~]$ nano simpleid2.c
```

Рис. 3.6: Программа

```
 GNU nano 5.6.1                                  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. 3.7: Программа

6) Скомпилируем программу (рис. 3.8).

```
[guest@derenchik ~]$ gcc simpleid2.c -o simpleid2
```

Рис. 3.8: Компиляция

7) Запустим программу (рис. 3.9).

```
[guest@derenchik ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
```

Рис. 3.9: Программа

8) От имени суперпользователя выполним следующие команды (рис. 3.10).

```
[guest@derenchik ~]$ su -
Password:
Last login: Thu Apr 11 18:18:01 MSK 2024 on pts/0
[root@derenchik ~]# chown root:guest /home/guest/simpleid2
[root@derenchik ~]# chmod u+s /home/guest/simpleid2
[root@derenchik ~]# exit
logout
```

Рис. 3.10: Команды

9) Выполним проверку правильности установки новых атрибутов (рис. 3.11).

```
[guest@derenchik ~]$ ls -l simpleid2
-rwsr-xr-x. 1 root guest 80768 Apr 11 18:24 simpleid2
```

Рис. 3.11: Проверка

10) Запустим simpleid2 и id, сравним результаты (рис. 3.12).

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

Рис. 3.12: Команды

11) Создадим программу readfile.c (рис. 3.13 - 3.14).

```
[guest@derenchik ~]$ touch readfile.c
[guest@derenchik ~]$ nano readfile.c
```

Рис. 3.13: Программа

```
 GNU nano 5.6.1                                  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. 3.14: Программа

12) Откомпилируем её (рис. 3.15).

```
[guest@derenchik ~]$ gcc readfile.c -o readfile
```

Рис. 3.15: Компиляция

13) Выполним команду (рис. 3.16).

```
[guest@derenchik ~]$ ./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;
}
```

Рис. 3.16: Команда

3.2 Исследование Sticky-бита

1) Выясним, установлен ли атрибут Sticky на директории /tmp (рис. 3.17).

```
[guest@derenchik ~]$ ls -l / | grep tmp
drwxrwxrwt.  19 root root 4096 Apr 11 18:28 tmp
```

Рис. 3.17: Sticky

2) От имени пользователя guest создадим файл file01.txt в директории /tmp со словом test (рис. 3.18).

```
[guest@derenchik ~]$ echo "test" > /tmp/file01.txt
```

Рис. 3.18: Файл

3) Просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные» (рис. 3.19).

```
[guest@derenchik ~]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 Apr 11 18:29 /tmp/file01.txt
[guest@derenchik ~]$ chmod o+rw /tmp/file01.txt
[guest@derenchik ~]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 Apr 11 18:29 /tmp/file01.txt
```

Рис. 3.19: Изменение атрибутов

4) От пользователя guest2 попробуем прочитать файл (рис. 3.20).

```
[guest@derenchik ~]$ su guest2
Password:
[guest2@derenchik guest]$ cat /tmp/file01.txt
test
```

Рис. 3.20: Чтение файла

5) От пользователя guest2 попробуем дозаписать в файл слово test2 (рис. 3.21). Операцию выполнить не удалось.

```
[guest2@derenchik guest]$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@derenchik guest]$ cat /tmp/file01.txt
test
```

Рис. 3.21: Дозапись

6) От пользователя guest2 попробуем удалить файл (рис. 3.22). Операцию выполнить не удалось.

```
[guest2@derenchik 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. 3.22: Удаление файла

7) Повысим свои права до суперпользователя и выполним после этого команду, снимающую атрибут t с директории /tmp 3.23.

```
[guest2@derenchik guest]$ su -
Password:
Last login: Thu Apr 11 18:24:53 MSK 2024 on pts/0
[root@derenchik ~]# chmod -t /tmp
[root@derenchik ~]# exit
logout
```

Рис. 3.23: Снятие атрибута

8) Снова попробуем дозаписать в файл слово test2 (рис. 3.24). Снова операцию выполнить не удалось.

```
[guest2@derenchik guest]$ echo "test2" > /tmp/file01.txt
bash: /tmp/file01.txt: Permission denied
[guest2@derenchik guest]$ cat /tmp/file01.txt
test
```

Рис. 3.24: Дозапись

9) Снова попробуем удалить файл (рис. 3.25). Теперь операцию выполнить удалось.

```
[guest2@derenchik guest]$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'? y
```

Рис. 3.25: Удаление файла

10) Вернем атрибут t на директорию /tmp (рис. 3.26).

```
[guest2@derenchik guest]$ su -
Password:
Last login: Thu Apr 11 18:31:30 MSK 2024 on pts/0
[root@derenchik ~]# chmod +t /tmp
[root@derenchik ~]# exit
logout
```

Рис. 3.26: Возвращение атрибута

4 Вывод

В результате выполнения работы я изучил механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получил практические навыков работы в консоли с дополнительными атрибутами. Рассмотрел работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.