Лабораторная работа № 5

Дисциплина: Информационная безопасность

Покрас Илья Михайлович

Содержание

Список иллюстраций

1 Цель работы

Целью данной лабораторной работы является 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 Создание программы

1. Я вошёл в систему от имени пользователя guest(2.1).

```
[guest@localhost dir1]$ touch simpleid.c
[guest@localhost dir1]$ emacs simpleid.c
```

Рис. 2.1: Создание файла и и последующее редактирование в emacs

И создал программу simpleid.c(2.2).

```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. 2.2: Код программы в редакторе Emacs

2. Я скомпилировал и выполнил программу simpleid. Далее я выполнил системную программу id(2.3).

```
[guest@localhost dir1]$ gcc simpleid.c -o simpleid
[guest@localhost dir1]$ ./simpleid
uid=1001, gid=1001
[guest@localhost dir1]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 2.3: Успешная компиляция и запуск simpleid. Запуск системной программы id

Данные, выведенные simpleid и id совпадают.

3. Я скопировал файл simpleid.c для последующего редактирования(**??**).
```
[guest@localhost dir1]$ cp simpleid.c simpleid2.c
[guest@localhost dir1]$ emacs simpleid2.c
```

И усложнил программу, добавив вывод действительных идентификаторов(2.4).

```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: Код программы в редакторе Emacs

4. Я скомпилировал и запустил simpleid2.c(2.5).

```
[guest@localhost dir1]$ gcc simpleid2.c -o simpleid2
[guest@localhost dir1]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
```

Рис. 2.5: Успешная компиляция

5. От имени суперпользователя я выполнил команды:

```
chown root:guest /home/guest/simpleid2
chmod u+s /home/guest/simpleid2
```

Далее я выполнил проверку правильности установки новых атрибутов и смены владельца файла simpleid2, после чего запустил simpleid2 и id. И проделал то же самое относительно SetGID-бита(2.6).

```
[guest@localhost dir1]$ su
Password:
[root@localhost dir1]# chown root:guest /home/guest/dir1/simpleid2
[root@localhost dir1]# chmod u+s /home/guest/dir1/simpleid2
[root@localhost dir1]# ls -l simpleid2
-rwsrwxr-x. 1 root guest 8616 Oct  7 02:08 simpleid2
[root@localhost dir1]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@localhost dir1]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@localhost dir1]# chmod u-s /home/guest/dir1/simpleid2
[root@localhost dir1]# chmod g+s /home/guest/dir1/simpleid2
[root@localhost dir1]# ls -l simpleid2
-rwxrwsr-x. 1 root guest 8616 Oct  7 02:08 simpleid2
[root@localhost dir1]# ./simpleid2
e_uid=0, e_gid=1001
real_uid=0, real_gid=0
[root@localhost dir1]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 2.6: Успешная компиляция с новым владельцем файла

6. Я создал программу readfile.c(2.7).

```
[guest@localhost dir1]$ touch readfile.c
[guest@localhost dir1]$ emacs readfile.c
```

Рис. 2.7: Создание файла и и последующее редактирование в emacs

```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. 2.8: Код программы в редакторе Emacs

И успешно скомпилировал её (2.9).

```
[root@localhost dir1]# gcc readfile.c -o readfile
[root@localhost dir1]#
```

Рис. 2.9: Успешная компиляция

7. Я сменил владельца у файла readfile.c и изменил права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог, после чего проверил, что пользователь guest не может прочитать файл readfile.c(2.10).

```
[root@localhost dir1]# chown root /home/guest/dir1/readfile.c
[root@localhost dir1]# chmod 700 /home/guest/dir1/readfile.c
[root@localhost dir1]# su guest
[guest@localhost dir1]$ cat readfile.c
cat: readfile.c: Permission denied
```

Рис. 2.10: Смена прав и владельца и проверка чтения файла

8. Я сменил у программы readfile владельца и установил SetU'D-бит, после чего проверил, может ли программа readfile прочитать файлы readfile.c и /etc/shadow(2.11).

```
[root@localhost dir1]# chown root:guest /home/guest/dir1/readfile
[root@localhost dir1]# chmod u+s readfile
[root@localhost dir1]# su guest
[guest@localhost dir1]$ ./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@localhost dir1]$ ./readfile /etc/shadow
root:$6$S4BuB7BHERxJEgWS$bgSCze0Ns/mvBMK2wn3lAFWssz2YIhiLvy3YYKs68
bin:*:18353:0:99999:7:::
daemon:*:18353:0:99999:7:::
```

Рис. 2.11: Смена прав и владельца и проверка функционала файла

2.2 Исследование Sticky-бита

1. Я выяснил, установлен ли атрибут Sticky на директории /tmp(2.12).

```
[guest@localhost dir1]$ ls -l / | grep tmp
drwxrwxrwt.  17 root root 4096 Oct  7 06:18 tmp
```

Рис. 2.12: Смена прав и владельца и проверка функционала программы

2. От имени пользователя guest я создал файл file01.txt в директории /tmp со словом test, далее просмотрел атрибуты у только что созданного файла и разрешил чтение и запись для категории пользователей «все остальные»(2.13).

```
[root@localhost dir1]# su guest
[guest@localhost dir1]$ echo "test" >> /tmp/file01.txt
[guest@localhost dir1]$ chmod o+rw  /tmp/file01.txt
[guest@localhost dir1]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 Oct  7 02:57 /tmp/file01.txt
```

Рис. 2.13: Создание файла и проверка его атрибутов

3. От пользователя guest2 я прочитал файл /tmp/file01.txt и дозаписал в файл слово test. Далее я записал в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию и проверил содержимое файла командой. Я попробовал удалить файл file01.txt(2.14).

```
[guest@localhost dir1]$ su guest2
Password:
[guest2@localhost dir1]$ cat /tmp/file01.txt
test
[guest2@localhost dir1]$ echo "test" >> /temp/file01.txt
bash: /temp/file01.txt: No such file or directory
[guest2@localhost dir1]$ echo "test" >> /tmp/file01.txt
[guest2@localhost dir1]$ cat /tmp/file01.txt
test
test
[guest2@localhost dir1]$ echo "test3" > /tmp/file01.txt
[guest2@localhost dir1]$ cat /tmp/file01.txt
test3
[guest2@localhost dir1]$ rm /tmp/file01.txt
rm: cannot remove ‘/tmp/file01.txt’: Operation not permitted
```

Рис. 2.14: Редактирование файла

Однако в ответ я получил отказ от выполнения операции.

4. Я повысил свои права до суперпользователя, затем снял атрибут t с директории /tmp, после чего покинул режим суперпользователя командой exit(2.15).

```
[guest2@localhost dir1]$ su
Password:
[root@localhost dir1]# chmod -t /tmp
[root@localhost dir1]# exit
exit
```

Рис. 2.15: Удаление атрибута /tmp в режиме суперпользователя

5. Я от имени пользователя guest2 проверил, что атрибута t у директории /tmp нет, после чего повторил те же действия. Теперь я могу удалить файл file01.txt(2.16).

```
[guest2@localhost dir1]$ ls -l / | grep tmp
drwxrwxrwx.  17 root root 4096 Oct  7 03:01 tmp
[guest2@localhost dir1]$ cat /tmp/file01.txt
test3
[guest2@localhost dir1]$ echo "test2" >> /tmp/file01.txt
[guest2@localhost dir1]$ cat /tmp/file01.txt
test3
test2
[guest2@localhost dir1]$ echo "test3" > /tmp/file01.txt
[guest2@localhost dir1]$ cat /tmp/file01.txt
test3
[guest2@localhost dir1]$ rm /tmp/file01.txt
```

Рис. 2.16: Редактирование файла в режиме суперпользователя

6. Я повысил свои права до суперпользователя и вернул атрибут t на директорию /tmp(2.17).

```
[guest2@localhost dir1]$
[guest2@localhost dir1]$ su
Password:
[root@localhost dir1]# chmod +t /tmp
[root@localhost dir1]# exit
exit
[guest2@localhost dir1]$ ls -l / | grep tmp
drwxrwxrwt.  18 root root 4096 Oct  7 03:02 tmp
```

Рис. 2.17: Добавление атрибута /tmp в режиме суперпользователя

3 Вывод

В ходе проделанной работы я изучил механизмы изменения идентификаторов, применения SetUID- и Sticky-битов, получил практических навыков работы в консоли с дополнительными атрибутами, а также рассмотрел работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.