

Лабораторная работа №5

Разанацуа Сара Естэлл

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.

Выполнение лабораторной работы

Для лабораторной работы необходимо проверить, установлен ли компилятор gcc, команда `gcc -v` позволяет это сделать. Также осуществляется отключение системы запретов с помощью `setenforce 0` (рис. 1).

Mis à niveau:

```
cpp-11.5.0-5.el9_5.x86_64
glibc-2.34-125.el9_5.3.x86_64
glibc-all-langpacks-2.34-125.el9_5.3.x86_64
glibc-common-2.34-125.el9_5.3.x86_64
glibc-gconv-extra-2.34-125.el9_5.3.x86_64
glibc-langpack-fr-2.34-125.el9_5.3.x86_64
libgcc-11.5.0-5.el9_5.x86_64
libgomp-11.5.0-5.el9_5.x86_64
```

Installé:

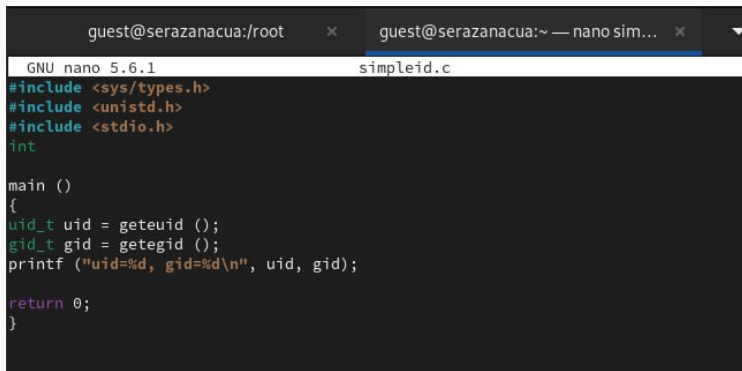
```
annobin-12.65-1.el9.x86_64
gcc-11.5.0-5.el9_5.x86_64
gcc-plugin-annobin-11.5.0-5.el9_5.x86_64
glibc-devel-2.34-125.el9_5.3.x86_64
glibc-headers-2.34-125.el9_5.3.x86_64
kernel-headers-5.14.0-503.28.1.el9_5.x86_64
```

Создание файла simpleid.c и запись в файл кода (рис. 2)

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

Рис. 2: Создание файла

Содержимое файла выглядит следующти образом (рис. 3)



```
guest@serazanacua:/root x guest@serazanacua:~ — nano sim... x
GNU nano 5.6.1 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: Содержимое файла

Компилирую файл, проверяю, что он скомпилировался (рис. 4)

```
[guest@serazanacua ~]$ gcc simpleid.c -o simpleid
[guest@serazanacua ~]$ ls
Bureau  Documents  Images  Musique  simpleid  Téléchargements
dir1    file1      Modèles Public   simpleid.c  Vidéos
[guest@serazanacua ~]$
```

Рис. 4: Компиляция файла

Запускаю исполняемый файл. В выводе файла выписаны номера пользователя и групп, от вывода при вводе `if`, они отличаются только тем, что информации меньше (рис. 5)

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

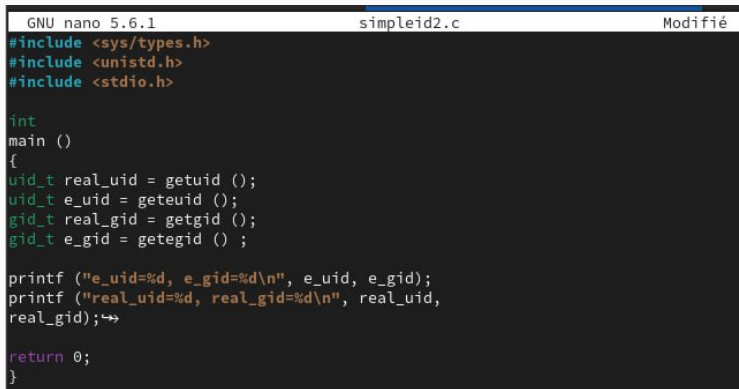
Рис. 5: Сравнение команд

Создание, запись в файл и компиляция файла simpleid2.c. Запуск программы (рис. 6)

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

Рис. 6: Создание и компиляция файла

Запуск программы



```
GNU nano 5.6.1                                simpleid2.c                                Modifié
#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. 7: Содержимое файла

С помощью `chown` изменяю владельца файла на суперпользователя, с помощью `chmod` изменяю права доступа (рис. 9)

```
[guest@serazanacua ~]$ chmod u+s /home/guest/simpleid2
[guest@serazanacua ~]$ ls -lchmod u+s /home/guest/simpleid2
ls: impossible d'accéder à 'u+s': Aucun fichier ou dossier de ce type
-rwsr-xr-x. 1 guest 18K 17 avril 19:39 /home/guest/simpleid2
[guest@serazanacua ~]$
```

Рис. 8: Смена владельца файла и прав доступа к файлу

Создание и компиляция файла readfile.c (рис. 10)

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

Рис. 9: Создание и компиляция файла

(рис. 12)

```
GNU nano 5.6.1                                readfile.c                                Modifié
#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;
}
```

Снова от имени суперпользователя меняю владельца файла readfile. Далее меняю права доступа так, чтобы пользователь guest не смог прочесть содержимое файла (рис. 13)

```
[guest@serazanacua ~]$ chmod u+s /home/guest/readfile
[guest@serazanacua ~]$ ls
Bureau      file1      Musique    readfile.c  simpleid2.c  Vidéos
dir1        Images    Public     simpleid     simpleid.c
Documents  Modèles   readfile   simpleid2    Téléchargements
[guest@serazanacua ~]$
```

Рис. 11: Смена владельца файла и прав доступа к файлу

Проверка прочесть файл от имени пользователя guest. Прочесть файл не удастся (рис. 14)

```
[guest@serazanacua ~]$ cat 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;
}
```

Выполнение лабораторной работы

Попытка прочесть тот же файл с помощью программы readfile, в ответ получаем “отказано в доступе” (рис. 15)

```
[guest@serazanacua ~]$ ./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]);
    }
```

Попытка прочесть файл `\etc\shadow` с помощью программы, все еще получаем отказ в доступе (рис. 16)

[illegible]

Проверяем папку tmp на наличие атрибута Sticky, т.к. в выводе есть буква t, то атрибут установлен (рис. 17)

```
[guest@serazanacua ~]$ ls -l / | grep tmp  
drwxrwxrwt. 18 root root 4096 17 avril 19:59 tmp  
[guest@serazanacua ~]$
```

Рис. 15: Проверка атрибутов директории tmp

Выводы

Изучила механизм изменения идентификаторов, применила SetUID- и Sticky-биты. Получила практические навыки работы в консоли с дополнительными атрибутами. Рассмотрела работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

Спасибо за внимание