

# **Отчет по лабораторной работе №5**

**Основы информационной безопасности**

Разанацуа Сара Естэлл НКАбд-02-23

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
<b>3</b>	<b>Выводы</b>	<b>13</b>
	<b>Список литературы</b>	<b>14</b>

## Список иллюстраций

2.1	Подготовка к лабораторной работе . . . . .	6
2.2	Создание файла . . . . .	6
2.3	Содержимое файла . . . . .	7
2.4	Компиляция файла . . . . .	7
2.5	Сравнение команд . . . . .	7
2.6	Создание и компиляция файла . . . . .	8
2.7	Содержимое файла . . . . .	8
2.8	Смена владельца файла и прав доступа к файлу . . . . .	8
2.9	Создание и компиляция файла . . . . .	9
2.10	Содержимое файла . . . . .	9
2.11	Смена владельца файла и прав доступа к файлу . . . . .	10
2.12	Попытка прочесть содержимое файла . . . . .	10
2.13	Попытка прочесть содержимое файла программой . . . . .	11
2.14	Попытка прочесть содержимое файла программой . . . . .	11
2.15	Проверка атрибутов директории tmp . . . . .	12

## **Список таблиц**

# 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 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-5.14.0-503.38.1.el9_5.x86_64
  libxcrypt-devel-4.4.18-3.el9.x86_64
  make-1:4.3-8.el9.x86_64

Terminé !
```

Рис. 2.1: Подготовка к лабораторной работе

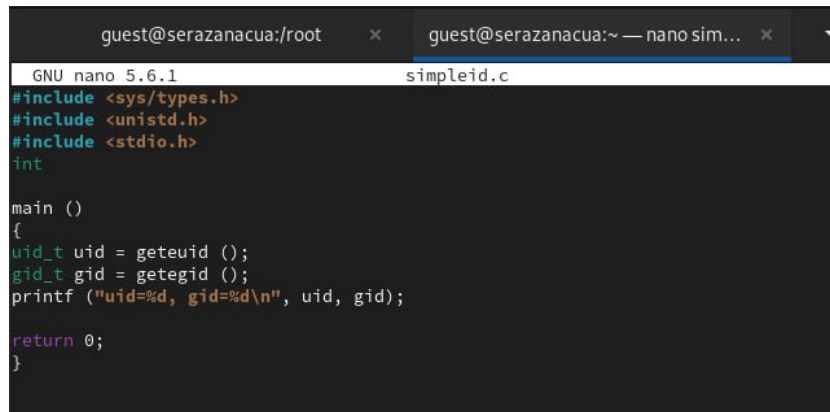
- Создание файла `simplified.c` и запись в файл кода (рис. 2)

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

Рис. 2.2: Создание файла

C++ Листинг 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; }`

- Содержимое файла выглядит следующи образом (рис. 3)



```
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. 2.3: 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. 2.4: 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. 2.5: Сравнение команд

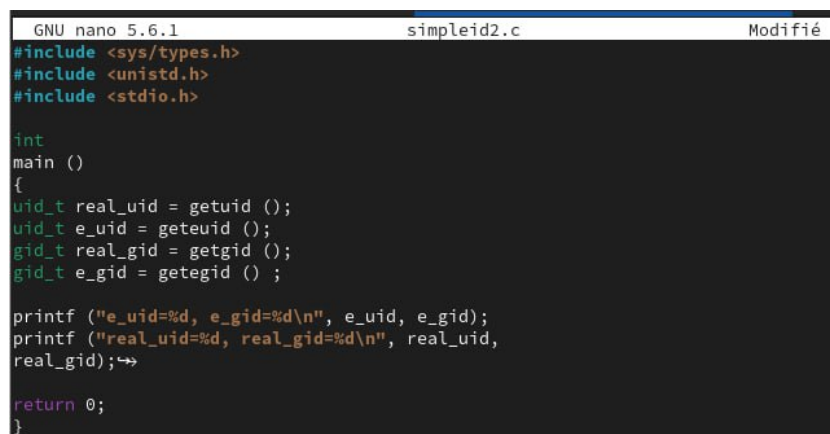
- Создание, запись в файл и компиляция файла simpleid2.c. Запуск программы (рис. 6)

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

Рис. 2.6: Создание и компиляция файла

C++ Листинг 2 `#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)



```
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. 2.7: 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. 2.8: Смена владельца файла и прав доступа к файлу



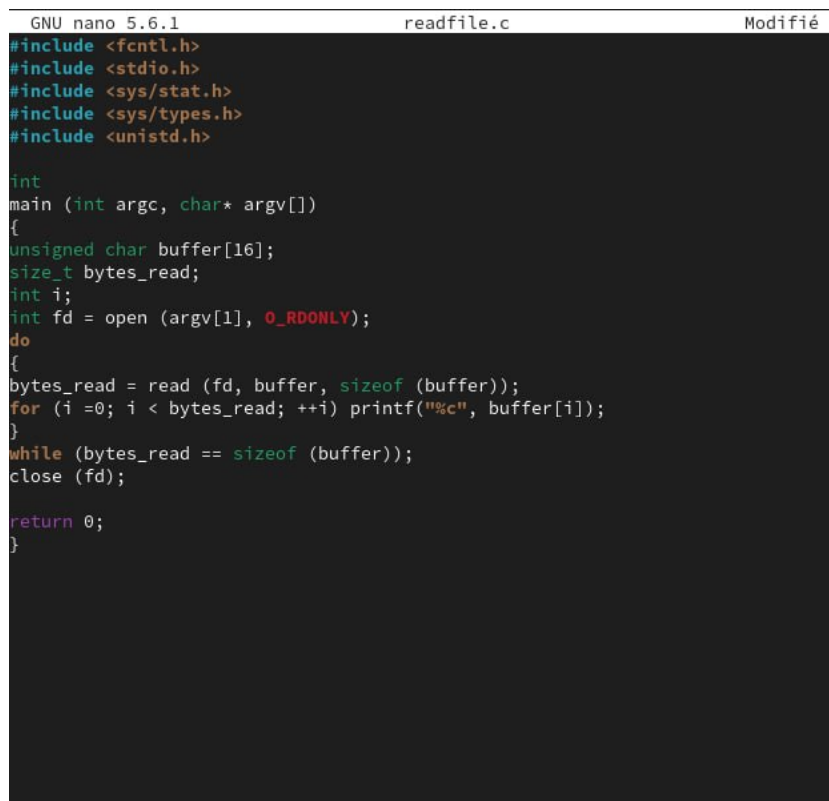
- Создание и компиляция файла readfile.c (рис. 10)

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

Рис. 2.9: Создание и компиляция файла

```
C++ Листинг 3 #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. 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;
}
```

Рис. 2.10: Содержимое файла

- 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. 2.11: 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;
}
[guest@serazanacua ~]$
```

Рис. 2.12: Попытка прочесть содержимое файла

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



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

Рис. 2.15: Проверка атрибутов директории tmp

## 3 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.

## **Список литературы**