

Отчёт по лабораторной работе №5

Дисциплина: Основы информационной безопасности

Набережных Дарина Денисовна, НПМбд-01-19

Содержание

| | | |
|---|--------------------------------|----|
| 1 | Цель работы | 5 |
| 2 | Теоретическое введение | 6 |
| 3 | Выполнение лабораторной работы | 7 |
| 4 | Выводы | 10 |
| | Список литературы | 11 |

Список иллюстраций

| | | |
|------|--|---|
| 3.1 | Создание файла simpleid | 7 |
| 3.2 | Работа программы simpleid.c | 7 |
| 3.3 | Файл simpleid2 | 8 |
| 3.4 | Компиляция simpleid2.c | 8 |
| 3.5 | Работа программы simpleid2.c | 8 |
| 3.6 | Установка бита | 8 |
| 3.7 | Запуск программы | 9 |
| 3.8 | Выполнение команды ls-l | 9 |
| 3.9 | Текст файла readfile | 9 |
| 3.10 | Компиляция readfile | 9 |

List of Tables

1 Цель работы

Изучение особенностей работы с дополнительными атрибутами и битами, а так же изучить их влияние на работу с файлами.

2 Теоретическое введение

StUID, SetGID, Sticku - это специальные типы разрешений, которые позволяют задавать расширенные права доступа на файлы и каталоги.

3 Выполнение лабораторной работы

Создадим файл `simpleid` и запишем в него следующую программу: (рис. 3.1).

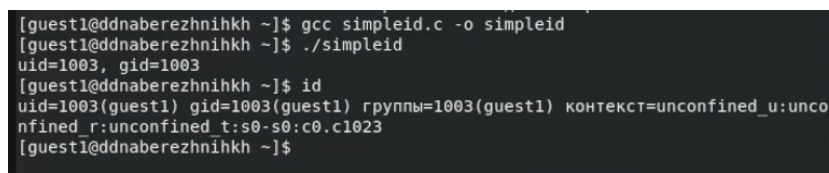


```
Open simpleid.c Save
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int
main ()
{
    uid_t uid = geteuid();
    gid_t gid = getegid();
    printf("uid=%d, gid=%d\n", uid, gid);
    return 0;
}
```

Рис. 3.1: Создание файла `simpleid`

Скомпилируем `simpleid.c` и запустим его через `./` (рис. 3.2).



```
[guest1@ddnaberezhnikh ~]$ gcc simpleid.c -o simpleid
[guest1@ddnaberezhnikh ~]$ ./simpleid
uid=1003, gid=1003
[guest1@ddnaberezhnikh ~]$ id
uid=1003(guest1) gid=1003(guest1) группы=1003(guest1) контекст=unconfined_u:unco
nfinied_r:unconfined_t:s0-s0:c0.c1023
[guest1@ddnaberezhnikh ~]$
```

Рис. 3.2: Работа программы `simpleid.c`

Теперь создадим программу `simpleid2` на основе `simpleid`, изменив часть текста и добавив новые строки (рис. 3.3).



```
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

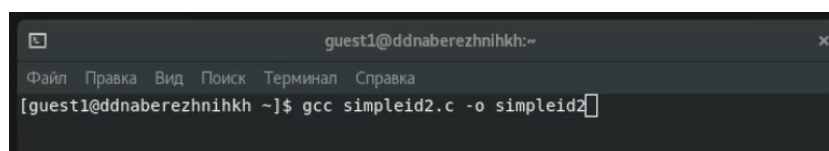
int
main ()
{
    uid_t real_uid = geteuid ();
    uid_t e_uid = getegid ();

    gid_t real_gid = getegid ();
    gid_t e_gid = geteuid ();

    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. 3.3: Файл simpleid2

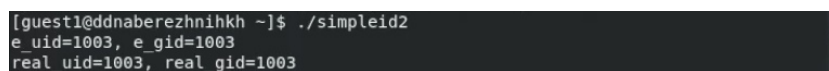
Скомпилируем программу simpleid2.c (рис. 3.4).



```
guest1@ddnabereznhikh:~$ gcc simpleid2.c -o simpleid2
```

Рис. 3.4: Компиляция simpleid2.c

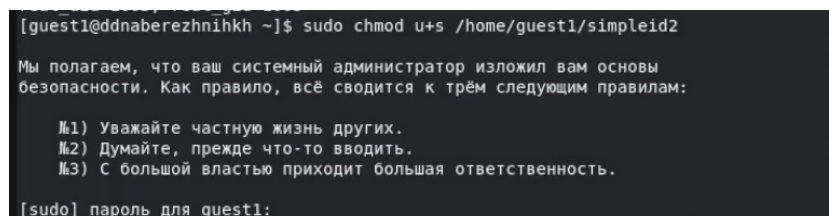
Теперь запустим программу simpleid2.c через ./ (рис. 3.5).



```
[guest1@ddnabereznhikh ~]$ ./simpleid2
e_uid=1003, e_gid=1003
real_uid=1003, real_gid=1003
```

Рис. 3.5: Работа программы simpleid2.c

Теперь установим SetGID-бит (рис. 3.6).



```
[guest1@ddnabereznhikh ~]$ sudo chmod u+s /home/guest1/simpleid2

Мы полагаем, что ваш системный администратор изложил вам основы
безопасности. Как правило, всё сводится к трём следующим правилам:

  №1) Уважайте частную жизнь других.
  №2) Думайте, прежде что-то вводить.
  №3) С большой властью приходит большая ответственность.

[sudo] пароль для guest1:
```

Рис. 3.6: Установка бита

Запускаем программу simpleid2 заново (рис. 3.7).


```
[guest1@ddnaberezhnikh ~]$ ./simpleid2
e_uid=1003, e_gid=1003
real_uid=1003, real_gid=1003
[guest1@ddnaberezhnikh ~]$
```

Рис. 3.7: Запуск программы

Выполним команду `ls-l` и получим следующие результаты: 3.8).

```
ddnaberezhnikh@ddnaberezhnikh:~
Файл Вид Поиск Терминал Справка
ddnaberezhnikh ~]$ su - ddnaberezhnikh

ezhnikh@ddnaberezhnikh ~]$ sudo ls -l /home/guest1/simpleid2
ароль для ddnaberezhnikh:
-x. 1 guest1 guest1 18152 сен 11 13:09 /home/guest1/simpleid2
ezhnikh@ddnaberezhnikh ~]$ sudo chmod g+s /home/guest1/simpleid2
ezhnikh@ddnaberezhnikh ~]$ sudo ls -l /home/guest1/simpleid2
-x. 1 guest1 guest1 18152 сен 11 13:09 /home/guest1/simpleid2
ezhnikh@ddnaberezhnikh ~]$
```

Рис. 3.8: Выполнение команды `ls-l`

Создадим `readfile` (рис. 3.9).

```
Open readfile.c Save
#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. 3.9: Текст файла `readfile`

Скомпилируем файл `readfile` (рис. 3.10).

```
[guest1@ddnaberezhnikh ~]$ gcc readfile.c -o readfile
[guest1@ddnaberezhnikh ~]$
```

Рис. 3.10: Компиляция `readfile`

4 Выводы

Я изучила механизм изменения идентификаторов и получила практические навыки по работе с битами

Список литературы