

Отчет по лабораторной работе №5

Основы информационной безопасности

Чувакина Мария Владимировна

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	8
4	Выводы	13

Список иллюстраций

3.1	Подготовка к лабораторной работе	8
3.2	Вход от имени пользователя guest	8
3.3	Создание файла	8
3.4	Содержимое файла	9
3.5	Компиляция файла	9
3.6	Сравнение команд	9
3.7	Создание и компиляция файла	9
3.8	Содержимое файла	10
3.9	Смена владельца файла и прав доступа к файлу	10
3.10	Запуск файла	10
3.11	Создание и компиляция файла	10
3.12	Содержимое файла	11
3.13	Смена владельца файла и прав доступа к файлу	11
3.14	Попытка прочесть содержимое файла	11
3.15	Попытка прочесть содержимое файла программой	11
3.16	Попытка прочесть содержимое файла программой	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 Теоретическое введение

1. Дополнительные атрибуты файлов Linux

В Linux существует три основных вида прав — право на чтение (read), запись (write) и выполнение (execute), а также три категории пользователей, к которым они могут применяться — владелец файла (user), группа владельца (group) и все остальные (others). Но, кроме прав чтения, выполнения и записи, есть еще три дополнительных атрибута. [u?]

Sticky bit

Используется в основном для каталогов, чтобы защитить в них файлы. В такой каталог может писать любой пользователь. Но, из такой директории пользователь может удалить только те файлы, владельцем которых он является. Примером может служить директория /tmp, в которой запись открыта для всех пользователей, но нежелательно удаление чужих файлов.

SUID (Set User ID)

Атрибут исполняемого файла, позволяющий запустить его с правами владельца. В Linux приложение запускается с правами пользователя, запустившего указанное приложение. Это обеспечивает дополнительную безопасность т.к. процесс с правами пользователя не сможет получить доступ к важным системным файлам, которые принадлежат пользователю root.

SGID (Set Group ID)

Аналогичен suid, но относится к группе. Если установить sgid для каталога, то все файлы созданные в нем, при запуске будут принимать идентификатор группы каталога, а не группы владельца, который создал файл в этом каталоге.

Обозначение атрибутов **sticky**, **suid**, **sgid**

Специальные права используются довольно редко, поэтому при выводе программы `ls -l` символ, обозначающий указанные атрибуты, закрывает символ стандартных прав доступа.

Пример: `rwsrwsrwt`

где первая `s` — это `suid`, вторая `s` — это `sgid`, а последняя `t` — это `sticky bit`

В приведенном примере не понятно, `gwt` — это `gw-` или `gwx`? Определить это просто. Если `t` маленькое, значит `x` установлен. Если `T` большое, значит `x` не установлен. То же самое правило распространяется и на `s`.

В числовом эквиваленте данные атрибуты определяются первым символом при четырехзначном обозначении (который часто опускается при назначении прав), например в правах `1777` — символ `1` обозначает `sticky bit`. Остальные атрибуты имеют следующие числовое соответствие:

1 — установлен `sticky bit`

2 — установлен `sgid`

4 — установлен `suid`

2. Компилятор GCC

GCC - это свободно доступный оптимизирующий компилятор для языков C, C++. Собственно программа `gcc` это некоторая надстройка над группой компиляторов, которая способна анализировать имена файлов, передаваемые ей в качестве аргументов, и определять, какие действия необходимо выполнить. Файлы с расширением `.cc` или `.C` рассматриваются, как файлы на языке C++, файлы с расширением `.c` как программы на языке C, а файлы с расширением `.o` считаются объектными [**gcc?**].

3 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).

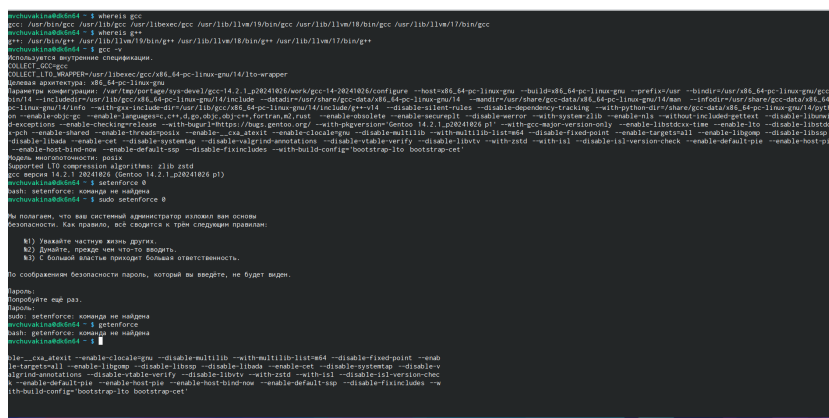


Рис. 3.1: Подготовка к лабораторной работе

Осуществляется вход от имени пользователя guest (рис. 2).

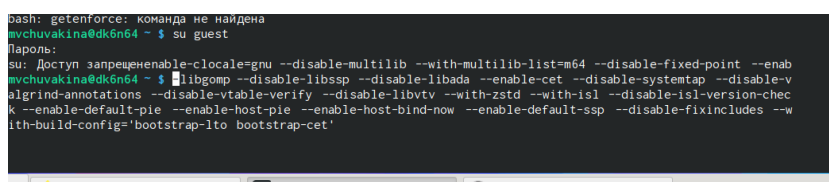


Рис. 3.2: Вход от имени пользователя guest

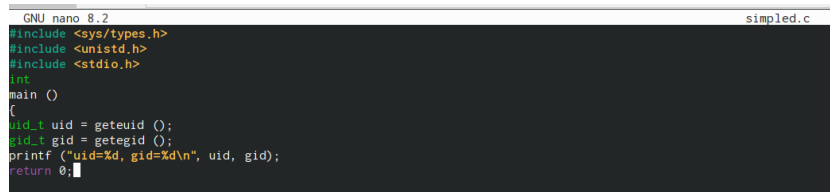
Создание файла `simplified.c` и запись в файл кода (рис. 3)



Рис. 3.3: 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. 4)



```
GNU nano 8.2 simplified.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. 3.4: Содержимое файла

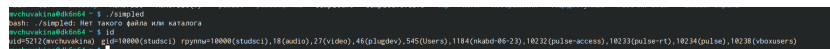
Компилирую файл, проверяю, что он скомпилировался (рис. 5)



```
hchuvakina@ddn4 ~$ gcc simplified.c -o simplified
/usr/lib/gcc/x86_64-linux-gnu/14/../../../../x86_64-linux-gnu/bin/ld: /usr/lib/gcc/x86_64-linux-gnu/14/../../../../lib64/Scrt1.o: в функции «_start»:
(.text+0): undefined reference to «main»
collect2: ошибка: выполнение ld завершилось с кодом возврата 1
hchuvakina@ddn4 ~$ ls
bin  dir1  GNUstep  Makefile  'public public.html'  simplified  simplified.c  save  tmp  work  Изображения  Общедоступные  'Рабочий стол'  'Список ядра'
hchuvakina@ddn4 ~$
```

Рис. 3.5: 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. 6)



```
hchuvakina@ddn4 ~$ ./simplified
uid=1000(gid=1000)
hchuvakina@ddn4 ~$
```

Рис. 3.6: Сравнение команд

Создание, запись в файл и компиляция файла simplified2.c. Запуск программы (рис. 7)



```
hchuvakina@ddn4 ~$ touch simplified2.c
hchuvakina@ddn4 ~$ gcc simplified2.c -o simplified2
/usr/lib/gcc/x86_64-linux-gnu/14/../../../../x86_64-linux-gnu/bin/ld: /usr/lib/gcc/x86_64-linux-gnu/14/../../../../lib64/Scrt1.o: в функции «_start»:
(.text+0): undefined reference to «main»
collect2: ошибка: выполнение ld завершилось с кодом возврата 1
hchuvakina@ddn4 ~$ ./simplified2
uid=1000(gid=1000)
hchuvakina@ddn4 ~$
```

Рис. 3.7: 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. 8)

```
GNU nano 2.2 simple2.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. 3.8: Содержимое файла

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

```
evchovak@dnk64:~$ sudo chown root:guest /home/guest/simple2
Пароль:
evchovak@dnk64:~$ sudo chmod u+s /home/guest/simple2
Пароль:
evchovak@dnk64:~$ sudo ls -l /home/guest/simple2
-rwsr-xr-x 1 root guest 1024 2014-08-23 10:23 /home/guest/simple2
evchovak@dnk64:~$
```

Рис. 3.9: Смена владельца файла и прав доступа к файлу

Сравнение вывода программы и команды `id`, наша команда снова вывела только ограниченное количество информации(рис. 10)

```
evchovak@dnk64:~$ sudo /home/guest/simple2
Пароль:
evchovak@dnk64:~$ id
uid=512(evchovak) gid=1000(studsci) группы=1000(studsci),18(audio),27(video),46(pulse),545(User),1184(nkud-06-23),1823(pulse-access),1823(pulse-r),1823(pulse),1823(vboxusers)
evchovak@dnk64:~$ sudo id
uid=0(root) gid=0(root) группы=0(root)
evchovak@dnk64:~$
```

Рис. 3.10: Запуск файла

Создание и компиляция файла `readfile.c` (рис. 11)

```
evchovak@dnk64:~$ touch readfile.c
evchovak@dnk64:~$ nano readfile.c
evchovak@dnk64:~$ gcc readfile.c -o readfile
/usr/lib/gcc/x86_64-linux-gnu/4.7/../../../../x86_64-linux-gnu/bin/ld: /usr/lib/gcc/x86_64-linux-gnu/4.7/../../../../lib64/Scrt1.o: в функции «_start»:
(.text+0x1b): undefined reference to `main'
collect2: ошибка: невозможно загрузить ссылаемые файлы
evchovak@dnk64:~$ ls
bin  dir  objdump  readfile  public  public_html  readfile.c  simple2.c  simple2  simple2.c.save  tmp  work
evchovak@dnk64:~$
```

Рис. 3.11: 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 2.2 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. 3.12: Содержимое файла

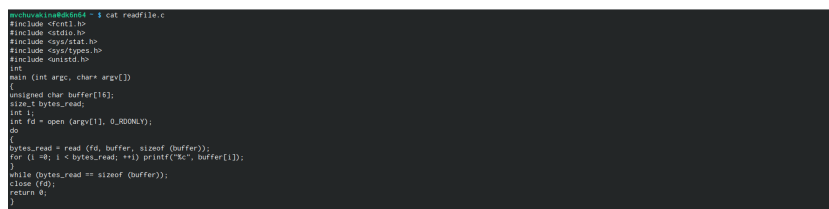
Снова от имени суперпользователя меняю владельца файла readfile. Далее меняю права доступа так, чтобы пользователь guest не смог прочесть содержимое файла (рис. 13)



```
rchovak@ns000004:~$ sudo chown root:guest /home/guest/readfile.c
Пароль:
Пользователь rchovak@ns000004 не разрешено выполнять «/usr/bin/chown root:guest /home/guest/readfile.c» как root на dn004.dk.sci.pfu.edu.ru.
rchovak@ns000004:~$ sudo chmod 000 /home/guest/readfile.c
Пароль:
Пользователь rchovak@ns000004 не разрешено выполнять «/usr/bin/chmod 000 /home/guest/readfile.c» как root на dn004.dk.sci.pfu.edu.ru.
rchovak@ns000004:~$ sudo ls -l /home/guest/readfile.c
```

Рис. 3.13: Смена владельца файла и прав доступа к файлу

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



```
rchovak@ns000004:~$ 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;
}
```

Рис. 3.14: Попытка прочесть содержимое файла

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



```
rchovak@ns000004:~$ ./readfile readfile.c
./readfile: Нет такого файла или каталога
rchovak@ns000004:~$
```

Рис. 3.15: Попытка прочесть содержимое файла программой

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

Рис. 3.16: Попытка прочесть содержимое файла программой

Пробуем прочесть эти же файлы от имени суперпользователя и чтение файлов проходит успешно

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

От имени пользователя guest создаю файл с текстом, добавляю права на чтение и запись для других пользователей

Вхожу в систему от имени пользователя guest2, от его имени могу прочитать файл file01.txt, но перезаписать информацию в нем не могу

Также невозможно добавить в файл file01.txt новую информацию от имени пользователя guest2

Далее пробуем удалить файл, снова получаем отказ

От имени суперпользователя снимаем с директории атрибут Sticky

Проверяем, что атрибут действительно снят

Далее был выполнен повтор предыдущих действий. По результатам без Sticky-бита запись в файл и дозапись в файл осталась невозможной, зато удаление файла прошло успешно

Возвращение директории tmp атрибута t от имени суперпользователя

4 Выводы

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