

Отчет по лабораторной работе №5

Основы информационной безопасности

Кабанова Варвара, НПМбд02-21

Содержание

Цель работы	1
Теоретическое введение	1
Выполнение лабораторной работы	2
Выводы.....	10
Список литературы. Библиография	10

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.

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. [a]

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, rwt — это rw- или rwx? 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].

Выполнение лабораторной работы

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

```

[root@localhost ~]# where is gcc
-bash: where: command not found
[root@localhost ~]# whereis gcc
gcc: /usr/bin/gcc /usr/lib/gcc /usr/libexec/gcc /usr/share/man/man1/gcc.1.gz /usr/share/info/gcc.info.gz
[root@localhost ~]# whereis g++
g++:
[root@localhost ~]# gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/x86_64-redhat-linux/11/lto-wrapper
OFFLOAD_TARGET_NAMES=nvptx-none
OFFLOAD_TARGET_DEFAULT=1
Target: x86_64-redhat-linux
Configured with: ../configure --enable-bootstrap --enable-host-pie --enable-host-bind-now --enable-languages=c,c++,fortran,lto --prefix=/usr --mandir=/usr/share/man --infodir=/usr/share/info --with-bugurl=https://bugs.redhat.com/ --enable-shared --enable-threads=posix --enable-checking=release --with-system-zlib --enable-_cxa_atexit --disable-libunwind-exceptions --enable-gnu-unique-object --enable-linker-build-id --with-gcc-major-version-only --enable-plugin --enable-initfini-array --without-isl --enable-multilib --with-linker-hash-style=gnu --enable-offload-targets=nvptx-none --without-cuda-driver --enable-gnu-indirect-function --enable-cet --with-tune=generic --with-arch_64=x86-64-v2 --with-arch_32=x86-64 --build=x86_64-redhat-linux --with-build-config=bootstrap-lto --enable-link-serialization=1
Thread model: posix
Supported LTO compression algorithms: zlib zstd
gcc version 11.4.1 28231210 (Red Hat 11.4.1-3) (GCC)
[root@localhost ~]# setenforce 0
[root@localhost ~]# sudo setenforce 0
[root@localhost ~]# getenforce
Permissive
[root@localhost ~]#

```

Подготовка к лабораторной работе

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

```

[root@localhost ~]# su guest
[guest@localhost root]$ _

```

Вход от имени пользователя guest

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

```

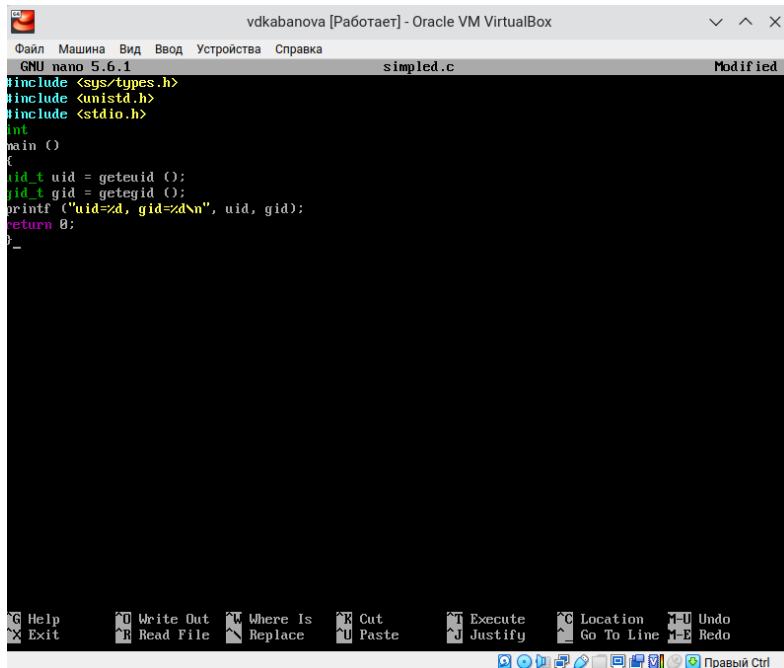
[root@localhost ~]# touch simplified.c
[root@localhost ~]# nano simplified.c_

```

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 следующти образом (рис. 4)



Содержимое файла

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

```
[root@localhost ~]# gcc simplified.c -o simplified
[root@localhost ~]# ls
anaconda-ks.cfg  anaconda-screenshots  simplified  simplified.c  'sudo fdisk -l'
[root@localhost ~]# _
```

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)

```
[root@localhost ~]# ./simplified
uid=0, gid=0
[root@localhost ~]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@localhost ~]# _
```

Сравнение команд

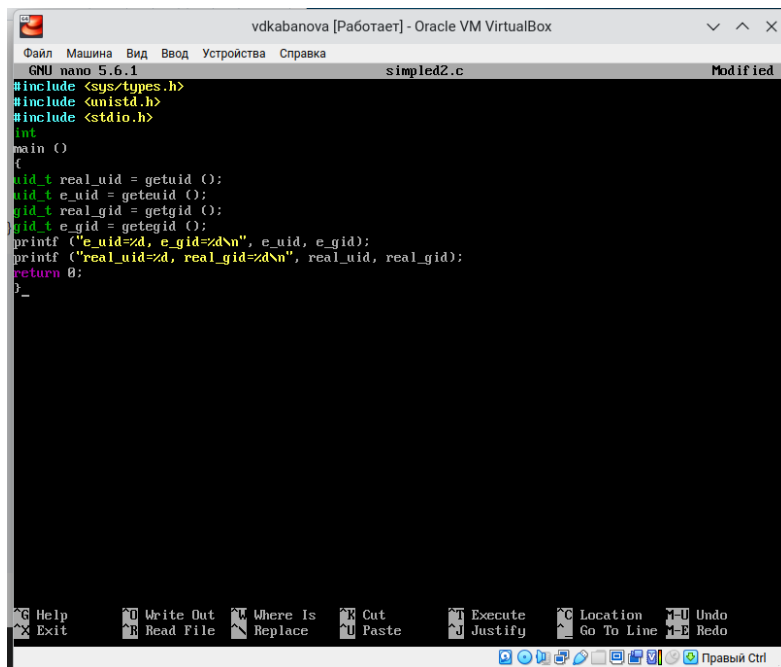
Создание, запись в файл и компиляция файла simplified2.c. Запуск программы (рис. 7)

```
[root@localhost ~]# gcc simplified2.c -o simplified2
[root@localhost ~]# ./simplified2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@localhost ~]# _
```

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)



Содержимое файла

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

```
sudo chown root:guest /home/guest/simplified2
sudo chmod u+s /home/guest/simplified2
sudo ls -l /home/guest/simplified2
6064 app 12 02:57 /home/guest/simplified2
```

Смена владельца файла и прав доступа к файлу

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

```
[root@localhost ~]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@localhost ~]# sudo id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@localhost ~]#
```

Запуск файла

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

```

[root@localhost ~]# gcc readfile.c -o readfile
[root@localhost ~]# ls
anaconda-ks.cfg      readfile      simplified    simplified2.c  'sudo fdisk -l'
anaconda-screenshots readfile.c    simplified2   simplified.c
[root@localhost ~]# _

```

Создание и компиляция файла

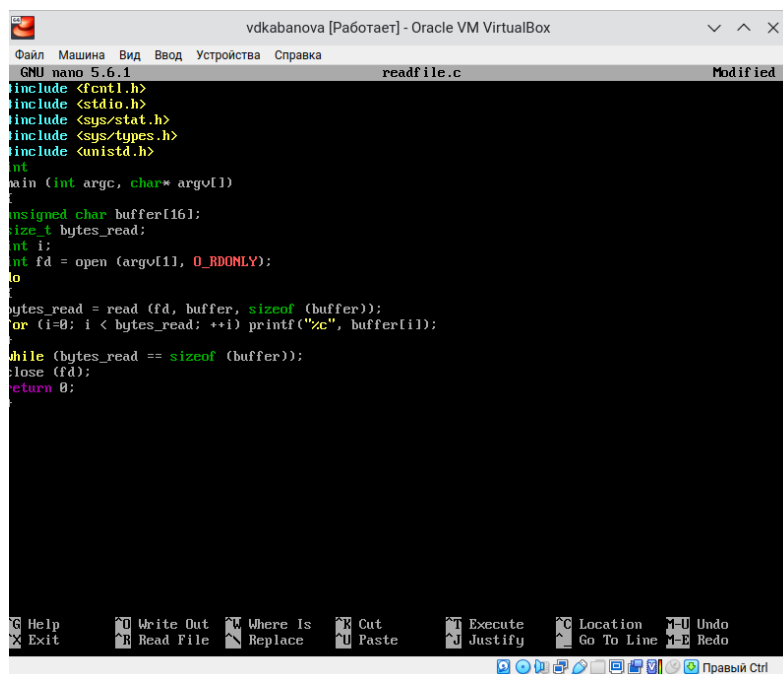
C++ Листинг 3

```

#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)



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)

```

~]$ sudo chown root:guest /home/guest/readfile.c
~]$ sudo chmod u+s /home/guest/readfile.c
~]$ sudo chmod 700 /home/guest/readfile.c
~]$ sudo chmod -r /home/guest/readfile.c
~]$ sudo chmod u+s /home/guest/readfile.c
~]$

```

Смена владельца файла и прав доступа к файлу

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

```
[guest@localhost root]$ cat readfile.c
cat: readfile.c: Permission denied
[guest@localhost root]$ _
```

Попытка прочесть содержимое файла

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

```
pg@localhost:~$ ./readfile readfile.c
cat: readfile.c: Permission denied
pg@localhost:~$
```

Попытка прочесть содержимое файла программой

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

```
pg@localhost:~$ ./readfile /etc/shadow
cat: /etc/shadow: Permission denied
pg@localhost:~$
```

Попытка прочесть содержимое файла программой

Пробуем прочесть эти же файлы от имени суперпользователя и чтение файлов проходит успешно (рис. 17)

```
[root@localhost ~]# ./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;
}
```

Чтение файла от имени суперпользователя

Проверяем папку tmp на наличие атрибута Sticky, т.к. в выводе есть буква t, то атрибут установлен (рис. 18)

```
[root@localhost ~]# ls -l / | grep tmp
drwxrwxrwt.  5 root root 4096 Sep 23 14:20 tmp
[root@localhost ~]#
```

Проверка атрибутов директории tmp

От имени пользователя guest создаю файл с текстом, добавляю права на чтение и запись для других пользователей (рис. 19)

```
[root@localhost ~]# echo "test" > /tmp/file01.txt
[root@localhost ~]# ls -l /tmp/file01.txt
-rw-r--r--. 1 root root 5 Sep 23 14:24 /tmp/file01.txt
[root@localhost ~]# chmod o+rw /tmp/file01.txt
[root@localhost ~]# ls -l /tmp/file01.txt
-rw-r--rw-. 1 root root 5 Sep 23 14:24 /tmp/file01.txt
[root@localhost ~]# _
```

Создание файла, изменение прав доступа

Вхожу в систему от имени пользователя guest2, от его имени могу прочитать файл file01.txt, но перезаписать информацию в нем не могу (рис. 20)

```
[root@localhost ~]# su guest2
[guest2@localhost root]# cat /tmp/file01.txt
test
[guest2@localhost root]# echo 'test2' >> /tmp/file01.txt
[guest2@localhost root]# cat /tmp/file01.txt
test
test2
```

Попытка чтения файла

Также невозможно добавить в файл file01.txt новую информацию от имени пользователя guest2 (рис. 21)

```
[guest@localhost root]# echo 'test3' >> /tmp/file01.txt
[guest@localhost root]# cat /tmp/file01.txt
test
test2
test3
```

Попытка записи в файл

Далее пробуем удалить файл, снова получаем отказ (рис. 22)

```
rm: удалить защищённый от записи обычный файл '/tmp/file01.txt'? y
rm: невозможно удалить '/tmp/file01.txt': Операция не позволена
```

Попытка удалить файл

От имени суперпользователя снимаем с директории атрибут Sticky (рис. 23)


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

Смена атрибутов файла

Проверяем, что атрибут действительно снят (рис. 24)

```
[guest2@localhost root]$ ls -l / | grep tmp
drwxrwxrwx.  5 root root 4096 Sep 23 14:31 tmp
[guest2@localhost root]$
```

Проверка атрибутов директории

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 (рис. 25)

```
[root@localhost ~]# cat /tmp/file01.txt
test
test2
test3
[root@localhost ~]# echo 'test4' >> /tmp/file01.txt
[root@localhost ~]# rm /tmp/file01.txt
rm: remove regular file '/tmp/file01.txt'? y
[root@localhost ~]# ls -l / | grep tmp
drwxrwxrwx.  5 root root 4096 Sep 23 14:34 tmp
[root@localhost ~]# ls -l
total 124
-rw-----. 1 root root  921 Sep  4 11:21 anaconda-ks.cfg
drwxr-x---. 2 root root 4096 Sep  4 11:21 anaconda-screenshots
-rwxr-xr-x. 1 root root 17664 Sep 23 14:16 readfile
-rw-r--r--. 1 root root  401 Sep 23 14:16 readfile.c
-rwxr-xr-x. 1 root root 17616 Sep 23 13:50 simplified
-rwxr-xr-x. 1 root root 17720 Sep 23 14:01 simplified2
-rw-r--r--. 1 root root  302 Sep 23 13:59 simplified2.c
-rw-r--r--. 1 root root  175 Sep 23 13:50 simplified.c
-rw-r--r--. 1 root root 41539 Sep  4 11:45 'sudo fdisk -l'
[root@localhost ~]# ls -l /home/guest
total 4
d-----.. 2 guest guest 19 Sep  9 16:45 dirl
d-----.. 2 guest guest 19 Sep  9 13:59 dirl
-rw-r--r--. 1 guest guest  5 Sep  9 13:51 test
[root@localhost ~]#
```

Повтор предыдущих действий

Возвращение директории tmp атрибута t от имени суперпользователя (рис. 26)

```
[root@localhost ~]# chmod +t /tmp
[root@localhost ~]# exit
exit
[guest2@localhost root]$
```

Изменение атрибутов

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. Библиография

[0] Методические материалы курса

[1] Права доступа: <https://codechick.io/tutorials/unix-linux/unix-linux-permissions>

[2] Расширенные атрибуты: <https://ru.manpages.org/xattr/7>

[3] Операции с расширенными атрибутами: <https://p-n-z-8-8.livejournal.com/64493.html>