



mkantoz1 / study_2024_2025_infosec



<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security



study_2024_2025_infosec / labs / lab05 / report / LAB5 report.md



mkantoz1 Rename report.md to LAB5 report.md

248cb88 · yesterday



232 lines (116 loc) · 13.6 KB

Preview

Code

Blame



Raw



Отчет по лабораторной работе №5

Основы информационной безопасности

Efe kantoz НКАбд-01-23

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

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. [1]

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 относиться 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

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 [2].

3 Выполнение лабораторной работы

```
whereis gcc
```

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).

```
gcc: /usr/bin/gcc /usr/lib/gcc /usr/libexec/gcc /usr/share/man/man1/gcc.1.gz /us
r/share/info/gcc.info.gz
evdvorkina@evdvorkina ~]$ whereis g++
g++: /usr/bin/g++ /usr/share/man/man1/g++.1.gz
evdvorkina@evdvorkina ~]$ gcc -v
Используются внутренние спецификации.
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
целевая архитектура: x86_64-redhat-linux
параметры конфигурации: ../configure --enable-bootstrap --enable-host-pie --enab
le-host-bind-now --enable-languages=c,c++,fortran,lto --prefix=/usr --mandir=/us
r/share/man --infodir=/usr/share/info --with-bugurl=https://bugs.rockylinux.org/
--enable-shared --enable-threads=posix --enable-checking=release --with-system-
lib --enable-__cxa_atexit --disable-libunwind-exceptions --enable-gnu-unique-ob
ject --enable-linker-build-id --with-gcc-major-version-only --enable-plugin --en
able-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
модель многопоточности: posix
supported LTO compression algorithms: zlib zstd
gcc version 11.4.1 20230605 (Red Hat 11.4.1-2) (GCC)
```

Подготовка к лабораторной работе

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

```
su guest
```

Вход от имени пользователя guest

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

```
~]$ touch simplified.c
~]$ 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)

```
GNU nano 5.6.1          simplified.c          Изменён
#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

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

```
gcc simplified.c -o simplified
```

```
[guest@evdvorkina ~]$ ls
dir1      test      Видео      Изображения  'Рабочий стол'
simplified test10     Документы  Музыка        Шаблоны
simplified.c test2     Загрузки   Общедоступные
```

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)

```
~]$ ./simplified
1
~]$ id
id=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi
t:s0-s0:c0.c1023
~]$
```

Сравнение команд

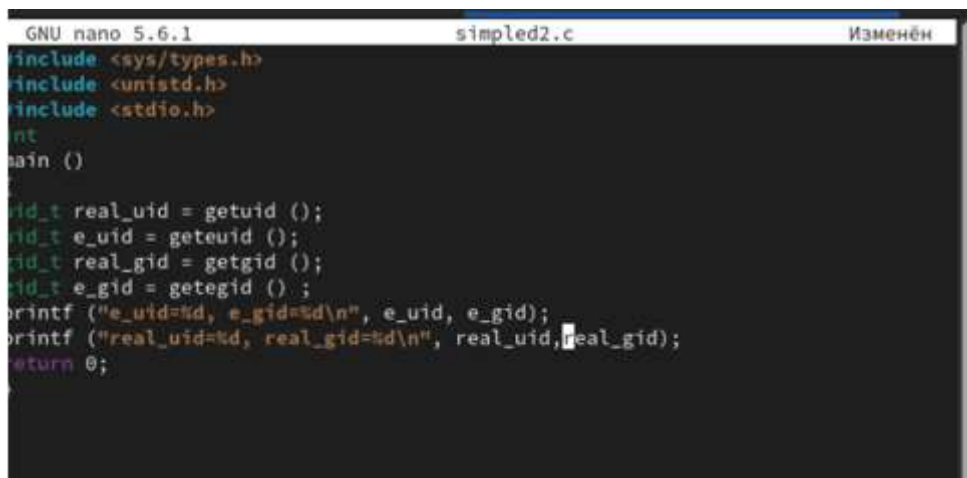
Создание, запись в файл и компиляция файла `simplified2.c`. Запуск программы (рис. 7)

```
~]$ touch simplified2.c
~]$ nano simplified2.c
~]$ gcc simplified2.c -o simplified2
~]$ ./simplified2
=1001
al_gid=1001
~]$
```

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 5.6.1 simplified2.c Изменён
#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;
}
```

Содержимое файла

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



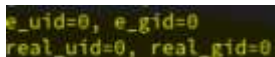
```
~]$ sudo chown root:guest /home/guest/simplified2
~]$ sudo chmod u+s /home/guest/simplified2
~]$ sudo ls -l /home/guest/simplified2
-rwsr-xr-x 1 root:guest 13 03:57 /home/guest/simplified2
~]$
```

Смена владельца файла и прав доступа к файлу



```
~]$ sudo /home/guest/simplified2
```

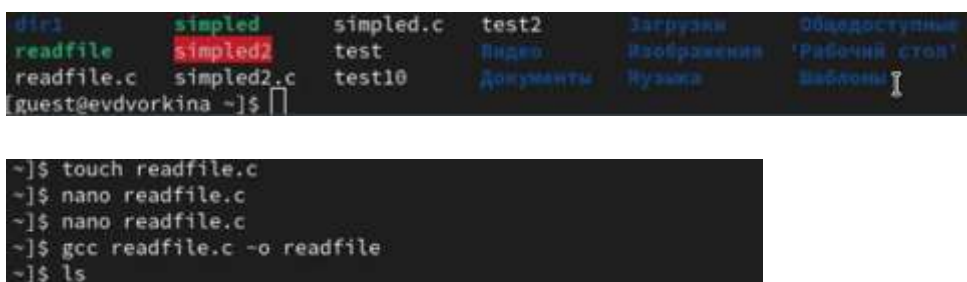
Сравнение вывода программы и команды id, наша команда снова вывела только ограниченное количество информации(рис. 10)



```
e_uid=0, e_gid=0
real_uid=0, real_gid=0
```

Запуск файла

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

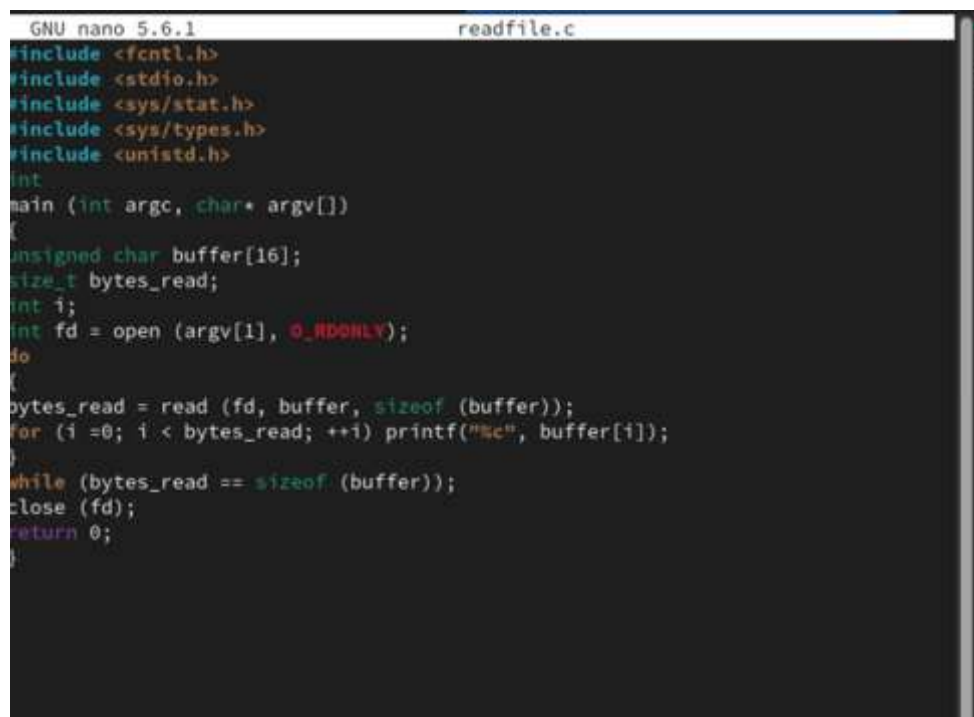


```
~]$ touch readfile.c
~]$ nano readfile.c
~]$ gcc readfile.c -o readfile
~]$ ls
readfile
```

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
#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;
}
```

Содержимое файла

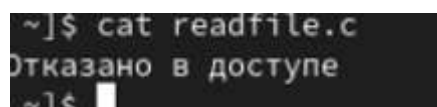
Снова от имени суперпользователя меняю владельца файла readfile. Далее меняю права доступа так, чтобы пользователь guest не смог прочесть содержимое файла (рис. 13)



```
~]$ 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)



```
~]$ cat readfile.c
Отказано в доступе
~]$
```

Попытка прочесть содержимое файла

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



Попытка прочесть содержимое файла программой

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



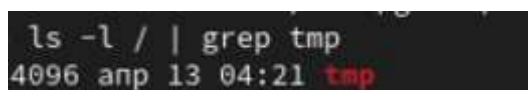
Попытка прочесть содержимое файла программой

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



Чтение файла от имени суперпользователя

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



Проверка атрибутов директории tmp

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

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)



Попытка чтения файла

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

```
bash: /tmp/file01.txt: Отказано в доступе
```

Попытка записи в файл

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

```
$ rm /tmp/file01.txt
с обычный файл '/tmp/file01.txt'? y
file01.txt': Операция не позволена
```

Попытка удалить файл

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

```
# chmod -t /tmp
# exit
```

Смена атрибутов файла

Проверяем, что атрибут действительно снят (рис. 24)

```
drwxrwxrwx. 18 root root 4096 anp 13 04:32 tmp
```

```
ls -l / | grep tmp
```

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)

```
итого 108
drwx-----, 3 guest guest 38 map 3 01:55 dir1
-rwxr-xr-x, 1 guest guest 26008 anp 13 04:19 readfile
-rw-r--r--, 1 guest guest 402 anp 13 04:19 readfile1.c
--ws-----, 1 root guest 402 anp 13 04:08 readfile.c
-rwxr-xr-x, 1 guest guest 25960 anp 13 03:53 simplified
-rwsr-xr-x, 1 root guest 26064 anp 13 03:57 simplified2
-rw-r--r--, 1 guest guest 302 anp 13 03:56 simplified2.c
-rw-r--r--, 1 guest guest 175 anp 13 03:53 simplified.c
-rw-r--r--, 1 guest guest 5 фев 18 20:39 test
-----, 1 guest guest 5 фев 18 20:27 test10
-----, 1 guest guest 0 фев 18 21:05 test2
drwxr-xr-x, 2 guest guest 6 фев 18 18:49 Видео
drwxr-xr-x, 2 guest guest 6 фев 18 18:49 Документы
drwxr-xr-x, 2 guest guest 6 фев 18 18:49 Загрузки
drwxr-xr-x, 2 guest guest 6 фев 18 18:49 Изображения
drwxr-xr-x, 2 guest guest 6 фев 18 18:49 Музыка
drwxr-xr-x, 2 guest guest 6 фев 18 18:49 Общедоступные
drwxr-xr-x, 2 guest guest 6 фев 18 18:49 Рабочий стол
drwxr-xr-x, 2 guest guest 6 фев 18 18:49 Шаблоны
```


Повтор предыдущих действий

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

```
~]# chmod +t /tmp  
~]# exit
```

Изменение атрибутов

4 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

1. Дополнительные атрибуты файлов: sticky bit, suid, sgid [Электронный ресурс]. 2018. URL: .
2. Инструментарий программиста в Linux: Компилятор GCC [Электронный ресурс]. URL: .