

Отчёта по лабораторной работе № 5

Информационная безопасность

Адоле Фейт Эне

Содержание

0.1	Цель работы	4
0.2	Теоретическое введение	4
0.3	Выполнение лабораторной работы	5
0.4	5.1 Создание программы	5
0.5	3.2 Исследование Sticky-бита	13
0.6	Выводы	16

Список иллюстраций

1	Рис. 5.1: Предварительная подготовка	5
2	Рис. 5.2: Команда “whereis”	6
3	Рис. 5.3: Вход в систему и создание программы	6
4	Рис. 5.4: Код программы simpleid.c	7
5	Рис. 5.5: Компиляция и выполнение программы simpleid	7
6	Рис. 5.6: Усложнение программы	8
7	Рис. 5.7: Переименование программы в simpleid2.c	8
8	Рис. 5.8: Компиляция и выполнение программы simpleid2	9
9	Рис. 5.9: Установка новых атрибутов (SetUID) и смена владельца файла	9
10	Рис. 5.10: Запуск simpleid2 после установки SetUID	10
11	Рис. 5.10: Запуск simpleid2 после установки SetUID	10
12	Рис. 5.12: Код программы readfile.c	11
13	Рис. 5.13: Смена владельца и прав доступа у файла readfile.c	12
14	Рис. 5.14: Запуск программы readfile	13
15	Рис. 5.15: Создание файла file01.txt	14
16	Рис. 5.16: Попытка выполнить действия над файлом file01.txt от имени пользователя guest2	15
17	Рис. 5.17: Удаление атрибута t (Sticky-бита) и повторение действий	16
18	Рис. 5.18: Возвращение атрибута t (Sticky-бита)	16

Список таблиц

0.1 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.

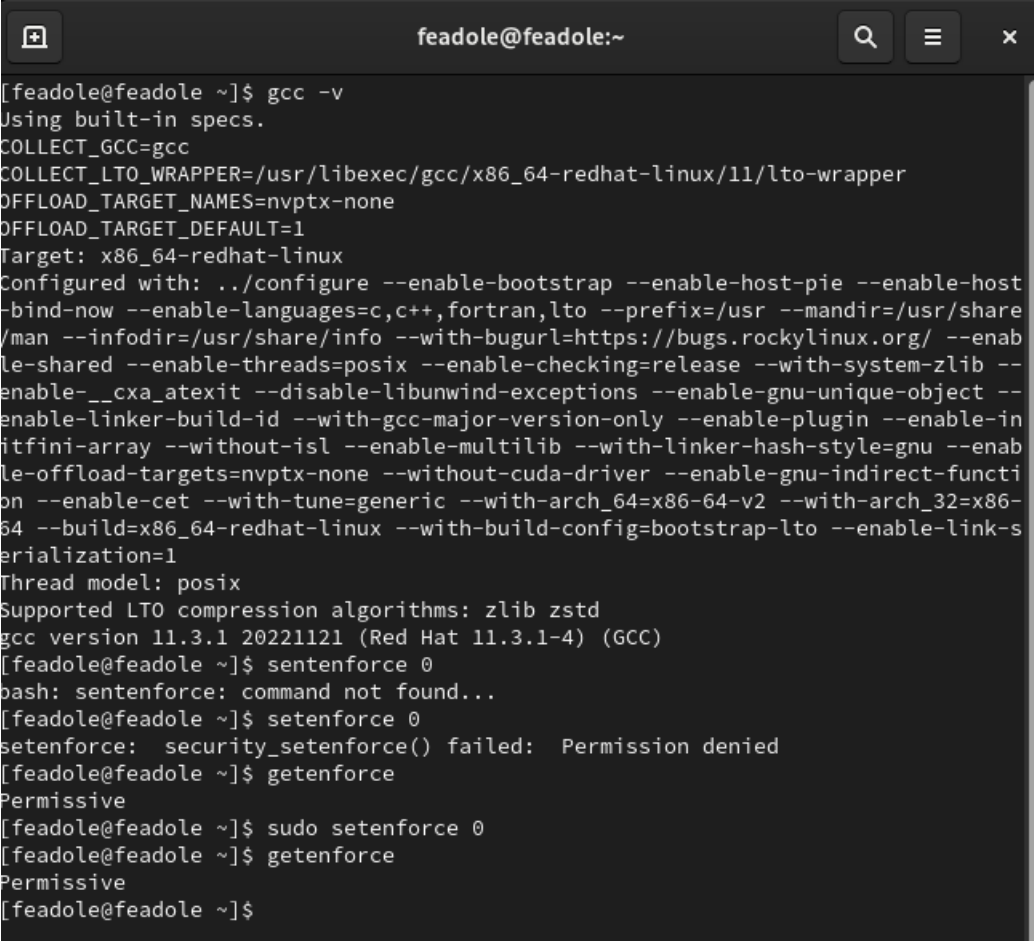
0.2 Теоретическое введение

SetUID, SetGID и Sticky - это специальные типы разрешений позволяют задавать расширенные права доступа на файлы или каталоги. • SetUID (set user ID upon execution — «установка ID пользователя во время выполнения) являются флагами прав доступа в Unix, которые разрешают пользователям запускать исполняемые файлы с правами владельца исполняемого файла. • SetGID (set group ID upon execution — «установка ID группы во время выполнения») являются флагами прав доступа в Unix, которые разрешают пользователям запускать исполняемые файлы с правами группы исполняемого файла. • Sticky bit в основном используется в общих каталогах, таких как /var или /tmp, поскольку пользователи могут создавать файлы, читать и выполнять их, принадлежащие другим пользователям, но не могут удалять файлы, принадлежащие другим пользователям.

0.3 Выполнение лабораторной работы

0.4 5.1 Создание программы

Для начала я убедилась, что компилятор gcc установлен, используя команду “gcc -v”. Затем отключила систему запретов до очередной перезагрузки системы командой “sudo setenforce 0”, после чего команда “getenforce” вывела “Permissive” (рис. 5.1).



```
feadole@feadole:~  
[feadole@feadole ~]$ 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.rockylinux.org/ --enab  
le-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-in  
itfini-array --without-isl --enable-multilib --with-linker-hash-style=gnu --enab  
le-offload-targets=nvptx-none --without-cuda-driver --enable-gnu-indirect-functi  
on --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-s  
erialization=1  
Thread model: posix  
Supported LTO compression algorithms: zlib zstd  
gcc version 11.3.1 20221121 (Red Hat 11.3.1-4) (GCC)  
[feadole@feadole ~]$ setenforce 0  
bash: setenforce: command not found...  
[feadole@feadole ~]$ setenforce 0  
setenforce: security_setenforce() failed: Permission denied  
[feadole@feadole ~]$ getenforce  
Permissive  
[feadole@feadole ~]$ sudo setenforce 0  
[feadole@feadole ~]$ getenforce  
Permissive  
[feadole@feadole ~]$
```

Рис. 1: Рис. 5.1: Предварительная подготовка

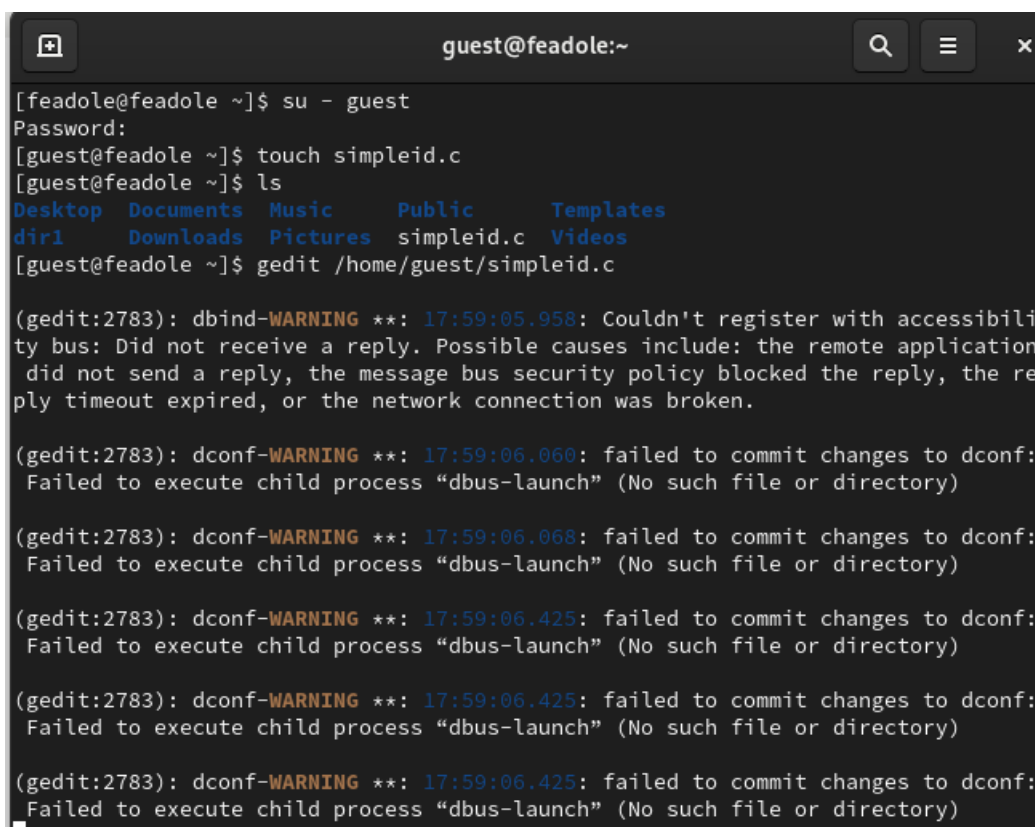
Проверила успешное выполнение команд “whereis gcc” и “whereis g++” (их расположение) (рис. 5.2).



```
feadole@feadole:~  
[feadole@feadole ~]$ 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  
[feadole@feadole ~]$ whereis g++  
g++: /usr/bin/g++ /usr/share/man/man1/g++.1.gz  
[feadole@feadole ~]$
```

Рис. 2: Рис. 5.2: Команда “whereis”

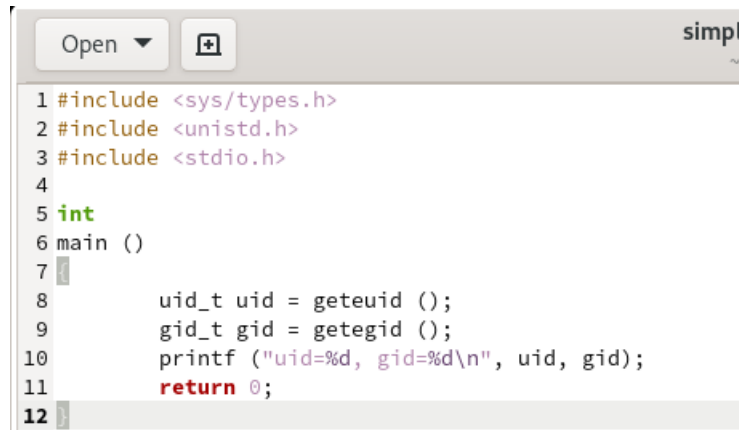
Вошла в систему от имени пользователя guest командой “su - guest”. Создала программу simpleid.c командой “touch simpleid.c” и открыла её в редакторе командой “gedit /home/guest/simpleid.c” (рис. 5.3).



```
guest@feadole:~  
[feadole@feadole ~]$ su - guest  
Password:  
[guest@feadole ~]$ touch simpleid.c  
[guest@feadole ~]$ ls  
Desktop Documents Music Public Templates  
dir1 Downloads Pictures simpleid.c Videos  
[guest@feadole ~]$ gedit /home/guest/simpleid.c  
  
(gedit:2783): dbind-WARNING **: 17:59:05.958: Couldn't register with accessibility bus: Did not receive a reply. Possible causes include: the remote application did not send a reply, the message bus security policy blocked the reply, the reply timeout expired, or the network connection was broken.  
  
(gedit:2783): dconf-WARNING **: 17:59:06.060: failed to commit changes to dconf: Failed to execute child process “dbus-launch” (No such file or directory)  
  
(gedit:2783): dconf-WARNING **: 17:59:06.068: failed to commit changes to dconf: Failed to execute child process “dbus-launch” (No such file or directory)  
  
(gedit:2783): dconf-WARNING **: 17:59:06.425: failed to commit changes to dconf: Failed to execute child process “dbus-launch” (No such file or directory)  
  
(gedit:2783): dconf-WARNING **: 17:59:06.425: failed to commit changes to dconf: Failed to execute child process “dbus-launch” (No such file or directory)  
  
(gedit:2783): dconf-WARNING **: 17:59:06.425: failed to commit changes to dconf: Failed to execute child process “dbus-launch” (No such file or directory)
```

Рис. 3: Рис. 5.3: Вход в систему и создание программы

Код программы выглядит следующим образом (рис. 5.4).



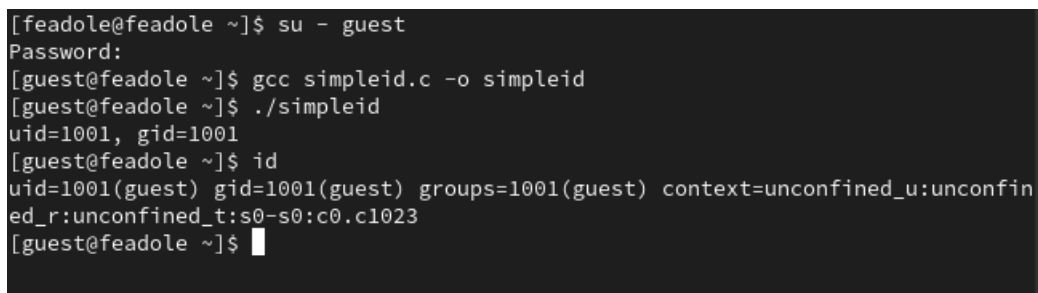
```

1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4
5 int
6 main ()
7 {
8     uid_t uid = geteuid ();
9     gid_t gid = getegid ();
10    printf ("uid=%d, gid=%d\n", uid, gid);
11    return 0;
12 }

```

Рис. 4: Рис. 5.4: Код программы simpleid.c

Скомпилировала программу и убедилась, что файл программы был создан командой “gcc simpleid.c -o simpleid”. Выполнила программу simpleid командой “./simpleid”, а затем выполнила системную программу id командой “id”. Результаты, полученные в результате выполнения обеих команд, совпадают (uid=1001 и gid=1001) (рис. 5.5).



```

[feadole@feadole ~]$ su - guest
Password:
[guest@feadole ~]$ gcc simpleid.c -o simpleid
[guest@feadole ~]$ ./simpleid
uid=1001, gid=1001
[guest@feadole ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@feadole ~]$

```

Рис. 5: Рис. 5.5: Компиляция и выполнение программы simpleid

Усложнила программу, добавив вывод действительных идентификаторов (рис. 5.6).

Рис. 6: Рис. 5.6: Усложнение программы

Получившуюся программу назвала simpleid2.c (рис. 3.7).

Рис. 7: Рис. 5.7: Переименование программы в simpleid2.c

Скомпилировала и запустила simpleid2.c командами “gcc simpleid2.c -o sipleid2”

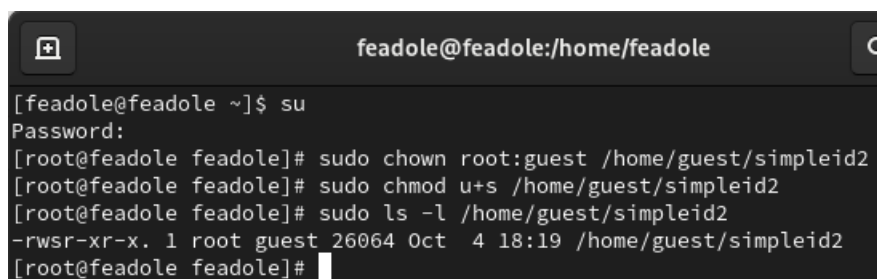
и “./simpleid2” (рис. 3.8).



```
[guest@feadole ~]$ gcc simpleid2.c -o simpleid2
[guest@feadole ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@feadole ~]$
```

Рис. 8: Рис. 5.8: Компиляция и выполнение программы simpleid2

От имени суперпользователя выполнила команды “sudo chown root:guest /home/guest/simpleid2” и “sudo chmod u+s /home/guest/simpleid2”, затем выполнила проверку правильности установки новых атрибутов и смены владельца файла simpleid2 командой “sudo ls -l /home/guest/simpleid2” (рис. 3.9). Этими командами была произведена смена пользователя файла на root и установлен SetUID-бит.



```
feadole@feadole:/home/feadole
[feadole@feadole ~]$ su
Password:
[root@feadole feadole]# sudo chown root:guest /home/guest/simpleid2
[root@feadole feadole]# sudo chmod u+s /home/guest/simpleid2
[root@feadole feadole]# sudo ls -l /home/guest/simpleid2
-rwsr-xr-x. 1 root guest 26064 Oct  4 18:19 /home/guest/simpleid2
[root@feadole feadole]#
```

Рис. 9: Рис. 5.9: Установка новых атрибутов (SetUID) и смена владельца файла

Запустила программы simpleid2 и id. Теперь появились различия в uid (рис. 5.10).

```
feadole@feadole:/home/feadole x guest@feadole:~ x ▼
[feadole@feadole ~]$ su - guest
Password:
[guest@feadole ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@feadole ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@feadole ~]$
```

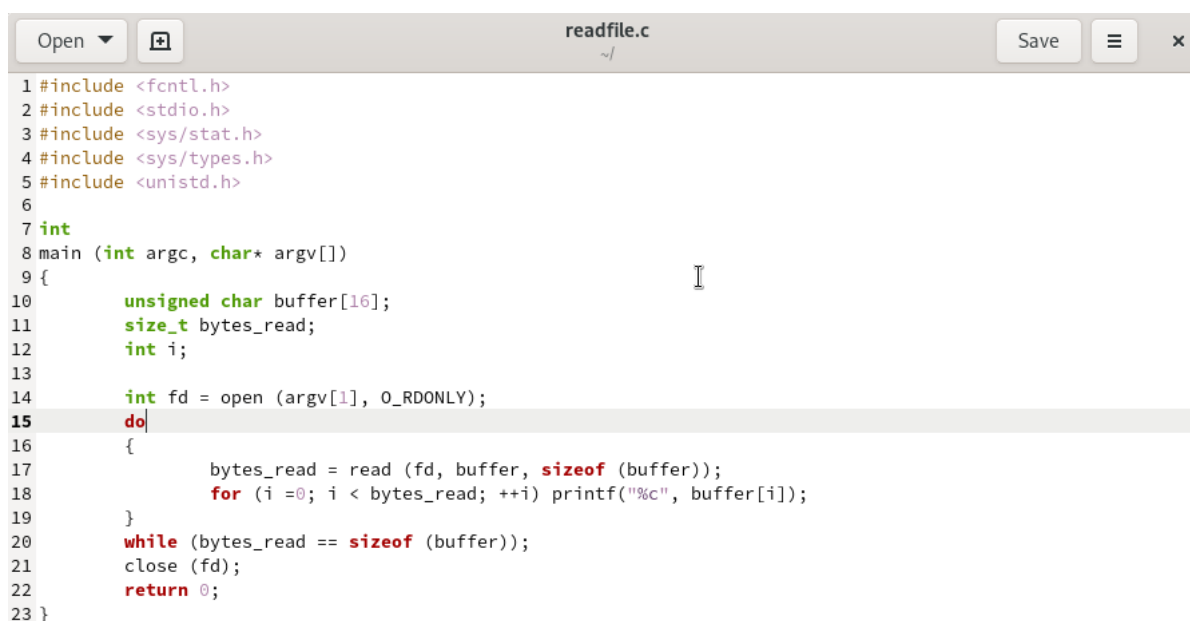
Рис. 10: Рис. 5.10: Запуск simpleid2 после установки SetUID

Проделала тоже самое относительно SetGID-бита. Также можем заметить различия с предыдущим пунктом (рис. 5.11).

```
feadole@feadole:/home/feadole x guest@feadole:~ x ▼
[feadole@feadole ~]$ su - guest
Password:
[guest@feadole ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@feadole ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@feadole ~]$
```

Рис. 11: Рис. 5.10: Запуск simpleid2 после установки SetUID

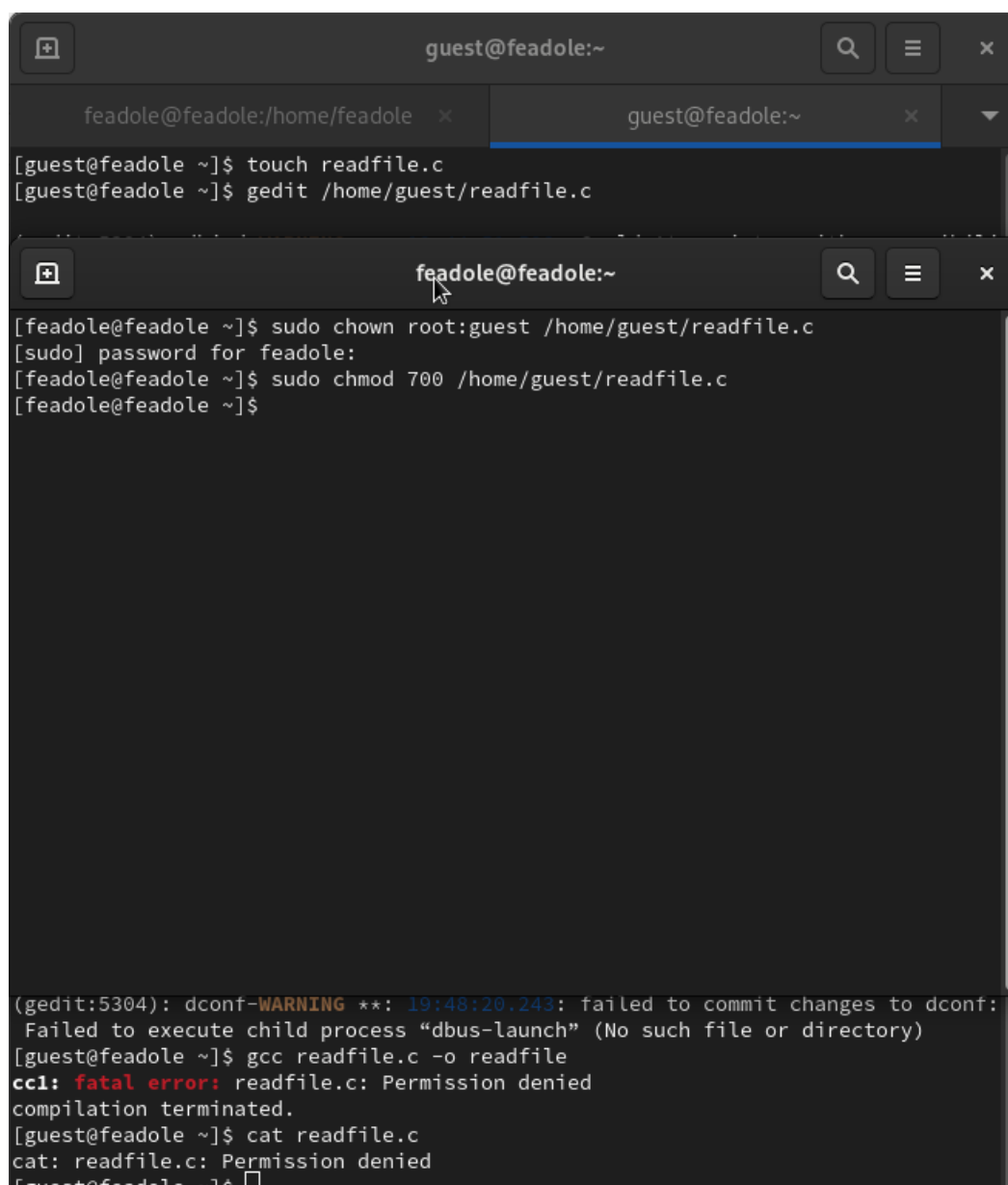
Создаем программу readfile.c (рис. 5.12).



```
1 #include <fcntl.h>
2 #include <stdio.h>
3 #include <sys/stat.h>
4 #include <sys/types.h>
5 #include <unistd.h>
6
7 int
8 main (int argc, char* argv[])
9 {
10     unsigned char buffer[16];
11     size_t bytes_read;
12     int i;
13
14     int fd = open (argv[1], O_RDONLY);
15     do
16     {
17         bytes_read = read (fd, buffer, sizeof (buffer));
18         for (i = 0; i < bytes_read; ++i) printf("%c", buffer[i]);
19     }
20     while (bytes_read == sizeof (buffer));
21     close (fd);
22     return 0;
23 }
```

Рис. 12: Рис. 5.12: Код программы readfile.c

Скомпилировала созданную программу командой “gcc readfile.c -o readfile”. Сменила владельца у файла readfile.c командой “sudo chown root:guest /home/guest/readfile.c” и поменяла права так, чтобы только суперпользователь мог прочитать его, а guest не мог, с помощью команды “sudo chmod 700 /home/guest/readfile.c”. Теперь убедилась, что пользователь guest не может прочитать файл readfile.c командой “cat readfile.c”, получив отказ в доступе (рис. 5.13).



The image shows two terminal windows. The top window is titled 'guest@feadole:~' and contains the following commands and output:

```
[guest@feadole ~]$ touch readfile.c
[guest@feadole ~]$ gedit /home/guest/readfile.c
```

The bottom window is titled 'feadole@feadole:~' and contains the following commands and output:

```
[feadole@feadole ~]$ sudo chown root:guest /home/guest/readfile.c
[sudo] password for feadole:
[feadole@feadole ~]$ sudo chmod 700 /home/guest/readfile.c
[feadole@feadole ~]$
```

Below these commands, there is a warning message from gedit:

```
(gedit:5304): dconf-WARNING **: 19:48:20.243: failed to commit changes to dconf:
Failed to execute child process "dbus-launch" (No such file or directory)
```

Then, the following commands and output are shown in the bottom window:

```
[guest@feadole ~]$ gcc readfile.c -o readfile
cc1: fatal error: readfile.c: Permission denied
compilation terminated.
[guest@feadole ~]$ cat readfile.c
cat: readfile.c: Permission denied
[guest@feadole ~]$
```

Рис. 13: Рис. 5.13: Смена владельца и прав доступа у файла readfile.c

Поменяла владельца у программы readfile и установила SetUID. Проверила, может ли программа readfile прочитать файл readfile.c командой “./readfile readfile.c”. Прочитать удалось. Аналогично проверила, можно ли прочитать файл /etc/shadow. Прочитать удалось (рис. 5.14).

```
guest@feadole:~  
[guest@feadole ~]$ ./readfile1 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@feadole ~]$ ./readfile1 /etc/shadow  
root:$6$5i/2qizgyvzaBNM$9JTHfQceCFK7Qnf0xkXPQc2JjIQhvpJbJB9f3yZBkVesqIF6HntwdHU0qggMgzIWkcRdXvw/  
0M7SA/3VEPhHL1::0:99999:7:::  
bin::19295:0:99999:7:::  
daemon::19295:0:99999:7:::  
adm::19295:0:99999:7:::  
lp::19295:0:99999:7:::  
sync::19295:0:99999:7:::  
shutdown::19295:0:99999:7:::  
halt::19295:0:99999:7:::  
mail::19295:0:99999:7:::  
operator::19295:0:99999:7:::  
games::19295:0:99999:7:::  
ftp::19295:0:99999:7:::  
nobody::19295:0:99999:7:::  
systemd-coredump::19607:1:::  
  
[feadole@feadole ~]$ sudo chown root:guest /home/guest/readfile1  
[feadole@feadole ~]$ sudo chmod u+s /home/guest/readfile1  
[feadole@feadole ~]$
```

Рис. 14: Рис. 5.14: Запуск программы readfile

0.5 3.2 Исследование Sticky-бита

Командой “ls -l / | grep tmp” убедилась, что атрибут Sticky на директории /tmp установлен. От имени пользователя guest создала файл file01.txt в директории /tmp со словом test командой “echo”test” > /tmp/file01.txt”. Просмотрела атрибуты у только что созданного файла и разрешаем чтение и запись для категории пользователей “все остальные” командами “ls -l /tmp/file01.txt” и “chmod o+rw /tmp/file01.txt” (рис. 5.15).

The screenshot shows a terminal window with two tabs. The left tab, titled 'guest@feadole:~', shows the following commands and output:

```
[guest@feadole ~]$ echo "test" > /tmp/file01.txt
[guest@feadole ~]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 Oct  5 01:05 /tmp/file01.txt
[guest@feadole ~]$ chmod o+rw /tmp/file01.txt
[guest@feadole ~]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 Oct  5 01:05 /tmp/file01.txt
[guest@feadole ~]$
```

The right tab, titled 'feadole@feadole:~', shows the following commands and output:

```
[feadole@feadole ~]$ ls -l / | grep tmp
drwxrwxrwt. 17 root root 4096 Oct  5 01:05 tmp
[feadole@feadole ~]$
```

Рис. 15: Рис. 5.15: Создание файла file01.txt

От имени пользователя guest2 попробовала прочитать файл командой “cat /tmp/file01.txt” - это удалось. Далее попыталась дозаписать в файл слово test2, проверить содержимое файла и записать в файл слово test3, стерев при этом всю имеющуюся в файле информацию - эти операции удалось выполнить только в случае, если еще дополнительно разрешить чтение и запись для группы пользователей командой “chmod g+rw /tmp/file01.txt”. От имени пользователя guest2 попробовала удалить файл - это не удастся ни в каком из случаев, возникает ошибка (рис. 5.16).

The image shows two terminal windows side-by-side. The left window is titled 'guest2@feadole:~' and shows a user switching from 'feadole' to 'guest2' using 'su - guest2'. The user then attempts to create and modify a file '/tmp/file01.txt' but receives 'Permission denied' errors. The right window is titled 'guest@feadole:~' and shows the user switching back to 'feadole' using 'su - guest'. The user then successfully creates and modifies the file '/tmp/file01.txt' using 'echo', 'cat', and 'chmod' commands. The terminal output in the right window shows the file permissions changing from 'rw-r--rw-. 1 guest guest 5 Oct 5 01:16 /tmp/file01.txt' to '-----rw-. 1 guest guest 5 Oct 5 01:16 /tmp/file01.txt' and finally to '----rw-rw-. 1 guest guest 5 Oct 5 01:16 /tmp/file01.txt'.

```
guest2@feadole:~  
[guest@feadole ~]$ su - guest2  
Password:  
[guest2@feadole ~]$ cat /tmp/file01.txt  
test  
[guest2@feadole ~]$ echo "test2" >> /tmp/file01.txt  
-bash: /tmp/file01.txt: Permission denied  
[guest2@feadole ~]$ cat /tmp/file01.txt  
test  
[guest2@feadole ~]$ ls -l /tmp/file01.txt  
-rw-r--rw-. 1 guest guest 5 Oct 5 01:09 /tmp/file01.txt  
[guest2@feadole ~]$ echo "test2" >> /tmp/file01.txt  
-bash: /tmp/file01.txt: Permission denied  
[guest2@feadole ~]$ cat /tmp/file01.txt  
test  
[guest2@feadole ~]$ echo "test2" > /tmp/file01.txt  
-bash: /tmp/file01.txt: Permission denied  
[guest2@feadole ~]$ echo "test2" >> /tmp/file01.txt  
-bash: /tmp/file01.txt: Permission denied  
[guest2@feadole ~]$ echo "test2" >> /tmp/file01.txt  
[guest2@feadole ~]$ cat /tmp/file01.txt  
test  
test2  
[guest2@feadole ~]$ echo "test3" >> /tmp/file01.txt  
[guest2@feadole ~]$ cat /tmp/file01.txt  
test  
test2  
test3  
[guest2@feadole ~]$ echo "test3" > /tmp/file01.txt  
[guest2@feadole ~]$ cat /tmp/file01.txt  
test3  
[guest2@feadole ~]$  
[guest2@feadole ~]$  
[feadole@feadole ~]$ su - guest  
Password:  
[guest@feadole ~]$ echo "test" > /tmp/file01.txt  
[guest@feadole ~]$ ls -l /tmp/file01.txt  
-rw-r--rw-. 1 guest guest 5 Oct 5 01:16 /tmp/file01.txt  
[guest@feadole ~]$ chmod 0 +rw /tmp/file01.txt  
chmod: cannot access '+rw': No such file or directory  
[guest@feadole ~]$ chmod 0+rw /tmp/file01.txt  
chmod: invalid mode: '0+rw'  
Try 'chmod --help' for more information.  
[guest@feadole ~]$ chmod o+rw /tmp/file01.txt  
[guest@feadole ~]$ ls -l /tmp/file01.txt  
-----rw-. 1 guest guest 5 Oct 5 01:16 /tmp/file01.txt  
[guest@feadole ~]$ chmod o+rw /tmp/file01.txt  
[guest@feadole ~]$ ls -l /tmp/file01.txt  
-----rw-. 1 guest guest 5 Oct 5 01:16 /tmp/file01.txt  
[guest@feadole ~]$ chmod g+rw /tmp/file01.txt  
[guest@feadole ~]$ ls -l /tmp/file01.txt  
----rw-rw-. 1 guest guest 5 Oct 5 01:16 /tmp/file01.txt  
[guest@feadole ~]$
```

Рис. 16: Рис. 5.16: Попытка выполнить действия над файлом file01.txt от имени пользователя guest2

Повысила права до суперпользователя командой “su -” и выполнила команду, снимающую атрибут t с директории /tmp “chmod -t /tmp”. После чего покинула режим суперпользователя командой “exit”. Повторила предыдущие шаги. Теперь мне удалось удалить файл file01.txt от имени пользователя, не являющегося его владельцем (рис. 5.17).

The image shows two terminal windows side-by-side. The left window, titled 'guest2@feadole:~', shows a series of commands and outputs:
1. `ls -l / | grep tmp` outputs `drwxrwxrwx. 17 root root 4096 Oct 5 01:22 tmp`.
2. `cat /tmp/file01.txt` outputs `test3`.
3. `echo "test2" >> /tmp/file01.txt` is executed.
4. `cat /tmp/file01.txt` outputs `test3` followed by `test2`.
5. `echo "test3" > /tmp/file01.txt` is executed.
6. `cat /tmp/file01.txt` outputs `test3`.
7. `rm /tmp/file01.txt` is executed.
The right window, titled 'guest@feadole:~', shows:
1. `su -` is executed, prompting for a password.
2. As root, `chmod -t /tmp` is executed.
3. `exit` is executed, logging out the user.
4. The prompt returns to `guest@feadole ~]$`.

Рис. 17: Рис. 5.17: Удаление атрибута t (Sticky-бита) и повторение действий

Повысила свои права до суперпользователя и вернула атрибут t на директорию /tmp (рис. 5.18).

The image shows two terminal windows side-by-side. The left window, titled 'guest2@feadole:~', shows:
1. `la -l / | grep tmp` is executed, resulting in a `bash: la: command not found...` error.
2. `ls -l / | grep tmp` is executed, outputting `drwxrwxrwx. 17 root root 4096 Oct 5 01:23 tmp`.
The right window, titled 'guest@feadole:~', shows:
1. `su -` is executed, prompting for a password.
2. As root, `chmod -t /tmp` is executed.
3. `exit` is executed, logging out the user.
4. The prompt returns to `guest@feadole ~]$`.

Рис. 18: Рис. 5.18: Возвращение атрибута t (Sticky-бита)

0.6 Выводы

В ходе выполнения данной лабораторной работы я изучила механизмы изменения идентификаторов, применение SetUID- и Sticky-битов. Получила практические навыки работы в консоли с дополнительными атрибутами. Рассмотрела работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.