

Основы информационной безопасности

**Лабораторная работа № 4. Дискреционное разграничение прав в Linux.
Два пользователя**

Подлесный Иван Сергеевич

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Выводы	11

Список иллюстраций

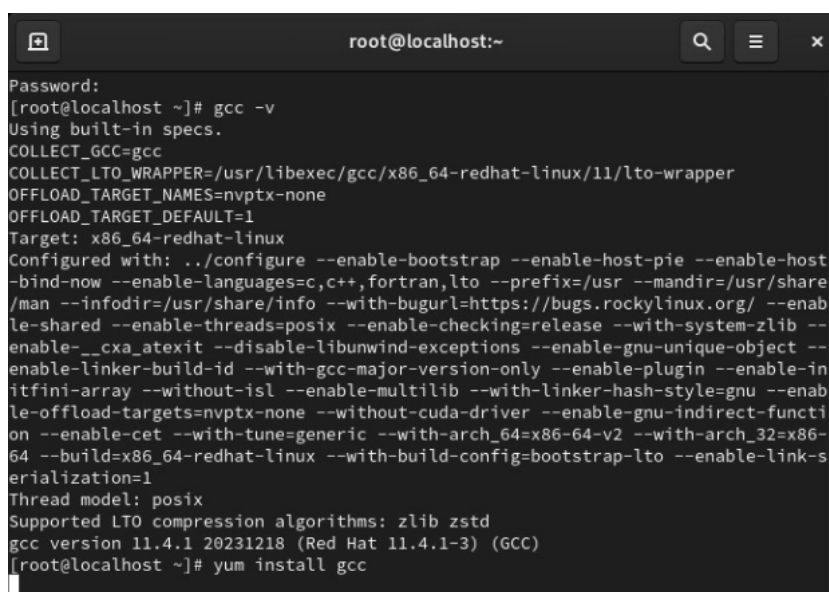
2.1	Подготовка лабораторного стенда	5
2.2	Текст программы simpleid.c	6
2.3	Запуск программы simpleid	6
2.4	Текст программы simpleid2.c	7
2.5	Запуск программы simpleid2	7
2.6	Изменение владельца и запуск программы simpleid2 с установленным SetUID-битом	8
2.7	Изменение владельца и запуск программы simpleid2 с установленным SetUID-битом	8
2.8	Текст программы readfile.c	8
2.9	Текст программы readfile.c	9
2.10	Изменение владельца и прав файла readfile.c	9
2.11	Установка SetUID-бита на исполняемый файл readfile и проверка прав	9
2.12	Работа с атрибутом Sticky	10
2.13	Работа с атрибутом Sticky	10

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 Выполнение лабораторной работы

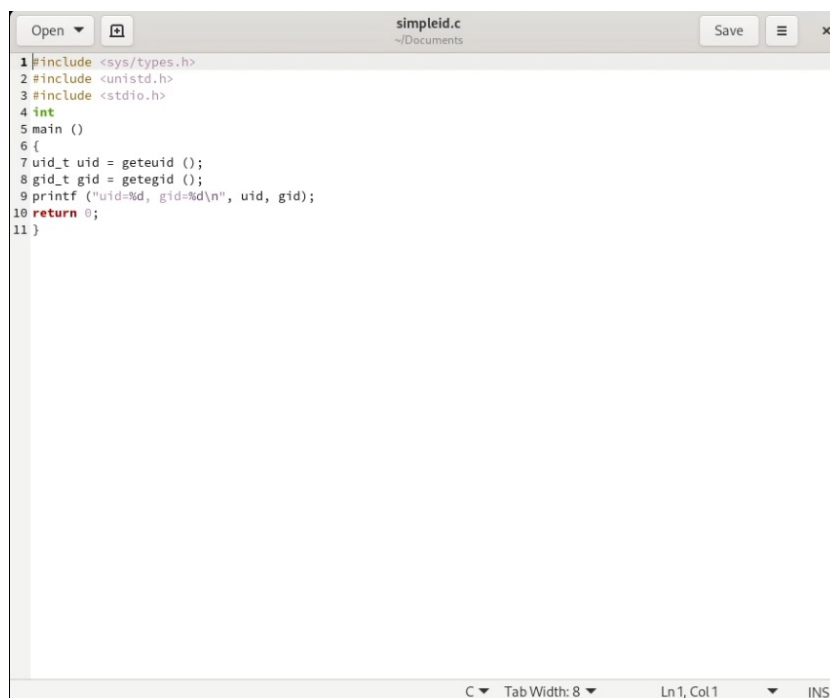
Проверим установлен ли компилятор gcc(обновим его), а также отключим SELinux(рис. fig. 2.1)

A terminal window titled 'root@localhost:~' with search, menu, and close icons. It shows the output of 'gcc -v' and the command 'yum install gcc'.

```
root@localhost:~  
Password:  
[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.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 11.4.1 20231218 (Red Hat 11.4.1-3) (GCC)  
[root@localhost ~]# yum install gcc
```

Рис. 2.1: Подготовка лабораторного стенда

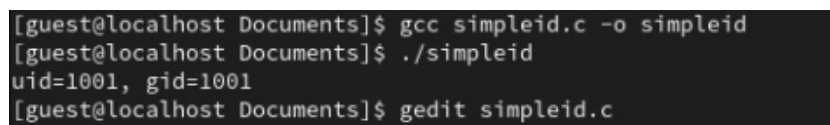
Войдем в систему от имени пользователя guest и создадим программу simpleid.c, которая выводит идентификатор пользователя и группы(рис. fig. 2.2)



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

Рис. 2.2: Текст программы simpleid.c

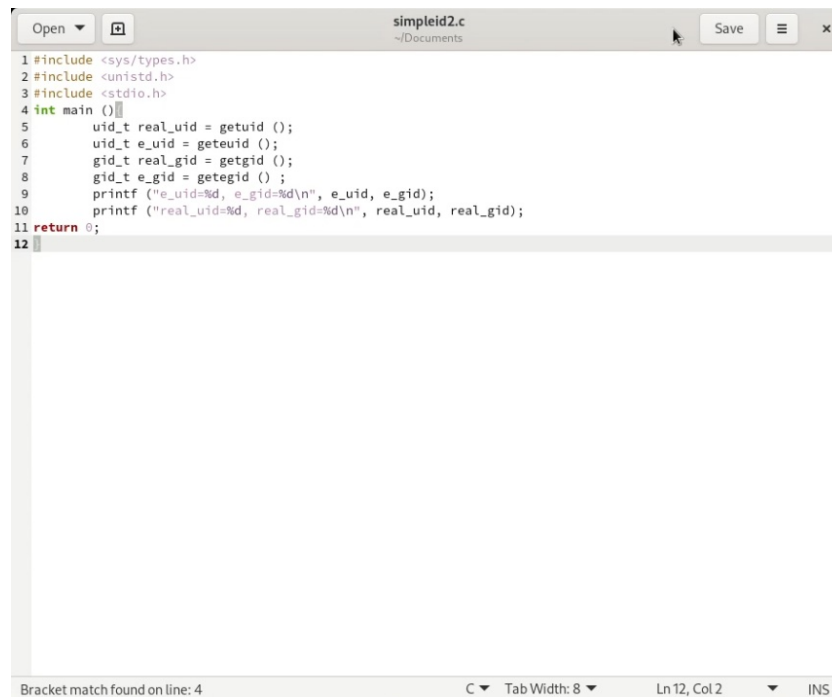
Запустив её, увидим, что она выводит идентификаторы пользователя и группы 1001 и 1001 для guest, что совпадает с выводом команды id(рис. fig. 2.3)



```
[guest@localhost Documents]$ gcc simpleid.c -o simpleid
[guest@localhost Documents]$ ./simpleid
uid=1001, gid=1001
[guest@localhost Documents]$ gedit simpleid.c
```

Рис. 2.3: Запуск программы simpleid

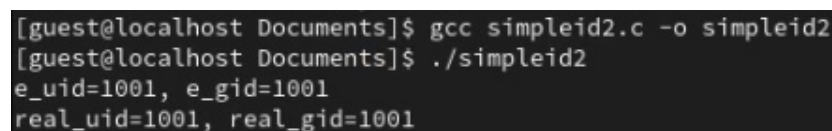
Изменим программу, добавив вывод действительных идентификаторов(рис. fig. 2.4).



```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4 int main ()
5 {
6     uid_t real_uid = getuid ();
7     uid_t e_uid = geteuid ();
8     gid_t real_gid = getgid ();
9     gid_t e_gid = getegid ();
10    printf ("e_uid=%d, e_gid=%d\n", e_uid, e_gid);
11    printf ("real_uid=%d, real_gid=%d\n", real_uid, real_gid);
12    return 0;
13 }
```

Рис. 2.4: Текст программы simpleid2.c

Компилируем программу с помощью gcc, затем, запустив её, увидим, что она выводит идентификаторы пользователя и группы 1001 и 1001 для guest, что совпадает с выводом команды id(рис. fig. 2.5).



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

Рис. 2.5: Запуск программы simpleid2

От имени суперпользователя изменим владельца файла homeguestsimpleid2 и установим SetUID-бит. Проверим корректность установленных прав и опять запустим simpleid2(рис. fig. 2.6).

```
root@localhost:~  
root@localhost:~  
guest@localhost:~/Documents  
[root@localhost ~]# chown root:guest /home/guest/Documents/simpleid2  
[root@localhost ~]# chmod u+s /home/guest/Documents/simpleid2  
chmod: cannot access '/home/guest/Documents/simpleid2': No such file or directory  
[root@localhost ~]# chmod u+s /home/guest/Documents/simpleid2  
[root@localhost ~]# ls -l /home/guest/Documents/simpleid2  
ls: cannot access '/home/guest/Documents/simpleid2': No such file or directory  
[root@localhost ~]# ls -l /home/guest/Documents/simpleid2  
-rwsr-xr-x. 1 root guest 24488 Oct  4 22:56 /home/guest/Documents/simpleid2  
[root@localhost ~]# ./home/guest/Documents/simpleid2  
-bash: ./home/guest/Documents/simpleid2: No such file or directory  
[root@localhost ~]# cd /home/guest/Documents/simpleid2
```

Рис. 2.6: Изменение владельца и запуск программы simpleid2 с установленным SetUID-битом

```
[guest@localhost Documents]$ ./simpleid2  
e_uid=0, e_gid=1001  
real_uid=1001, real_gid=1001  
[guest@localhost Documents]$ id  
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 2.7: Изменение владельца и запуск программы simpleid2 с установленным SetUID-битом

Прделаем аналогичные действия относительно SetGID-бита(рис. fig. ??):

```
[root@localhost ~]# chmod u-s /home/guest/Documents/simpleid2  
[root@localhost ~]# chmod g+s /home/guest/Documents/simpleid2  
[root@localhost ~]# ls -l /home/guest/Documents/simpleid2  
-rwxr-sr-x. 1 root guest 24488 Oct  4 22:56 /home/guest/Documents/simpleid2  
[guest@localhost Documents]  
e_uid=1001, e_gid=1001  
real_uid=1001, real_gid=1001
```

Создадим программу для чтения файлов readfile.c(рис. fig. 2.8):

```
Open readfile.c Save  
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 main (int argc, char* argv[])  
8 {  
9     unsigned char buffer[10];  
10    size_t bytes_read;  
11    int i;  
12    int fd = open (argv[1], O_RDONLY);  
13    do  
14    {  
15        bytes_read = read (fd, buffer, sizeof (buffer));  
16        for (i = 0; i < bytes_read; ++i) printf("%c", buffer[i]);  
17    } while (bytes_read == sizeof (buffer));  
18    close (fd);  
19    return 0;  
20
```

Рис. 2.8: Текст программы readfile.c


```
[root@localhost Documents]# gcc readfile.c -o readfile
```

Рис. 2.9: Текст программы readfile.c

Скомпилируем её и сменим владельца у файла с текстом программы, затем изменим права так, чтобы только суперпользователь (root) мог прочитать его, и проверим корректность настроек(рис. fig. 2.10):

```
[guest@localhost Documents]$ cat readfile.c
cat: readfile.c: Permission denied
[guest@localhost Documents]$ gcc readfile.c -o readfile
[guest@localhost Documents]$ ./readfile readfile.c
V@x7
```

Рис. 2.10: Изменение владельца и прав файла readfile.c

Сменим у программы readfile владельца и установим SetUID-бит. Теперь эта программа может прочитать файл readfile.c, также она может прочитать файл etcshadow, владельцем которого guest также не является, так как программа readfile теперь имеет все права пользователя root(рис. fig. 2.11):

```
[guest@localhost Documents]$ ./readfile etc/shadow
V@x7
```

Рис. 2.11: Установка SetUID-бита на исполняемый файл readfile и проверка прав

Проверим, что установлен атрибут Sticky на директории tmp(в конце стоит t). Затем от имени пользователя guest создадим файл file01.txt в директории tmp со словом test, затем просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные». После этого от пользователя guest2 попробуем дозаписать в этот файл новое слово, однако получим отказ, также нам отказано в перезаписи и удалении этого файла. Если же убрать Sticky бит, то нам будет разрешено удаление этого файла(рис. fig. 2.12):

```

[guest@localhost Documents]$ ls -l / | grep tmp
drwxrwxrwt. 17 root root 4096 Oct  4 23:20 tmp
[guest@localhost Documents]$ echo "test" > /tmp/file01.txt
[guest@localhost Documents]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 Oct  4 23:23 /tmp/file01.txt
[guest@localhost Documents]$ chmod o+rw /tmp/file01.txt
[guest@localhost Documents]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 Oct  4 23:23 /tmp/file01.txt
[guest@localhost Documents]$ su - guest2
Password:
[guest2@localhost ~]$ echo "test2" > /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@localhost ~]$ cat /tmp/file01.txt
test
[guest2@localhost ~]$ echo "test3" > /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@localhost ~]$ cat /tmp/file01.txt
test
[guest2@localhost ~]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory
[guest2@localhost ~]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': No such file or directory
[guest2@localhost ~]$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'?
[guest2@localhost ~]$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'? y
rm: cannot remove '/tmp/file01.txt': Operation not permitted
[guest2@localhost ~]$ su -
Password:
[root@localhost ~]# chmod -t /tmp
[root@localhost ~]# exit
logout

```

Рис. 2.12: Работа с атрибутом Sticky

```

[guest2@localhost ~]$ ls -l / | grep tmp~
[guest2@localhost ~]$ ls -l / | grep tmp~\
> ^C
[guest2@localhost ~]$ ls -l / | grep tmp
drwxrwxrwx. 17 root root 4096 Oct  4 23:30 tmp
[guest2@localhost ~]$ echo "test2" > /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@localhost ~]$ echo "test3" > /tmp/file01.txt
-bash: /tmp/file01.txt: Permission denied
[guest2@localhost ~]$ rm /tmp/file01.txt
rm: remove write-protected regular file '/tmp/file01.txt'? y
[guest2@localhost ~]$ su -
Password:
[root@localhost ~]# chmod +t /tmp

```

Рис. 2.13: Работа с атрибутом Sticky

3 Выводы

В результате выполнения работы были изучены механизмы изменения идентификаторов, применения SetUID- и Sticky-битов.

Были получены практических навыков работы в консоли с дополнительными атрибутами.

Были рассмотрены работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.