

Отчёт по лабораторной работе №5

**Дискреционное разграничение прав в Linux. Исследование влияния
дополнительных атрибутов**

Артамонов Тимофей Евгеньевич

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	7
4	Выводы	12
	Список литературы	13

Список иллюстраций

3.1	После запуска получили uid и gid нашего пользователя	7
3.2	Теперь выводятся и real uid и gid, все совпадает с результатами предыдущих шагов	8
3.3	chown изменяет владельца файла, а chmod u+s позволяет запускать файл с правами владельца. Теперь при запуске файла от имени guest получаем e_uid root	8
3.4	Вывод такой же	8
3.5	Меняем владельца на root и забираем все права у всех кроме владельца	9
3.6	guest не может прочесть readfile.c	9
3.7	Успешно	9
3.8	Успешно	10
3.9	Можем только читать файл, все что связано с изменением запрещено	10
3.10	Теперь изменение не для владельца открыто	11

Список таблиц

1 Цель работы

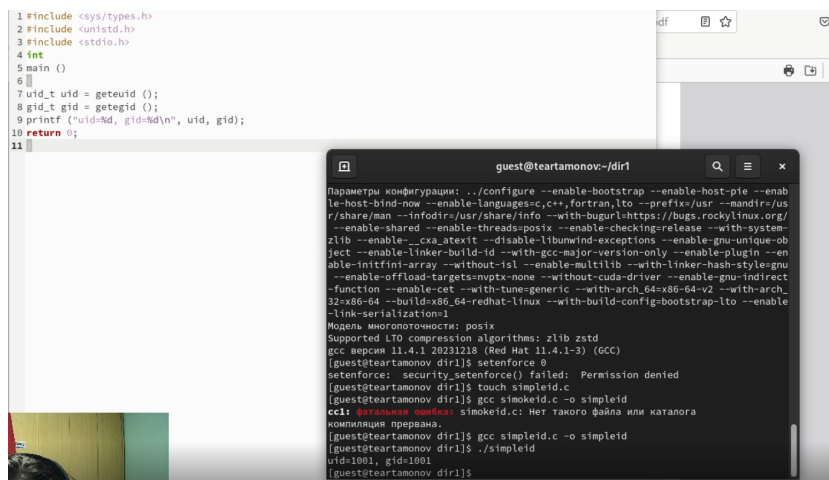
Изучение механизмов изменения идентификаторов, применения SetUID и Sticky-битов. Получение практических навыков работы в консоли с дополнительными атрибутами. Рассмотрение работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

2 Теоретическое введение

В настоящее время sticky bit используется в основном для каталогов, чтобы защитить в них файлы. Из такого каталога пользователь может удалить только те файлы, владельцем которых он является. Примером может служить каталог /tmp, в который запись открыта для всех пользователей, но нежелательно удаление чужих файлов. Установка атрибута производится утилитой `chmod`. [1]

3 Выполнение лабораторной работы

Создали файл simple.id и записали в него код из лабораторной. (рис. [3.1])



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

```
guest@teartamonov:~/dir1
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-
zlib --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
Модель микронормочности: posix
Supported LTO compression algorithms: zlib zstd
gcc версия 11.4.1 20231218 (Red Hat 11.4.1-3) (GCC)
[guest@teartamonov dir1]$ setenforce 0
setenforce: security.setenforce() failed: Permission denied
[guest@teartamonov dir1]$ touch simpleid.c
[guest@teartamonov dir1]$ gcc simpleid.c -o simpleid
cc1: параметр оптимизации: simpleid.c: Нет такого файла или каталога
компиляция прервана.
[guest@teartamonov dir1]$ gcc simpleid.c -o simpleid
[guest@teartamonov dir1]$ ./simpleid
uid=1001, gid=1001
[guest@teartamonov dir1]$
```

Рис. 3.1: После запуска получили uid и gid нашего пользователя

Усложним скрипт, добавив вывод real uid и gid. (рис. [3.2])

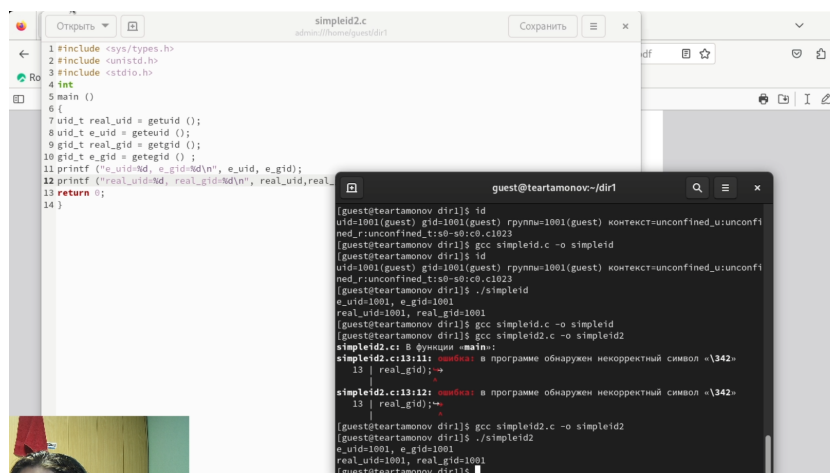


Рис. 3.2: Теперь выводятся и real uid и gid, все совпадает с результатами предыдущих шагов

Пропишем chown и chmod. (рис. [3.3])

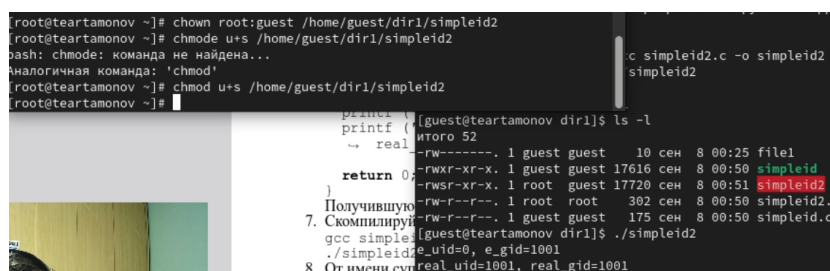


Рис. 3.3: chown изменяет владельца файла, а chmod u+s позволяет запускать файл с правами владельца. Теперь при запуске файла от имени guest получаем e_uid root

Проделаем то же самое с SetGID-битом. (рис. [3.4])

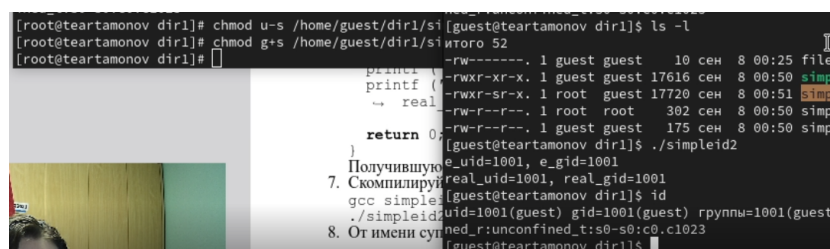


Рис. 3.4: Вывод такой же

Создадим файл readfile.c как в лабораторной и скомпилируем его. (рис. [3.5])

```
[root@teartamonov dir1]# chown root:guest /home/guest/dir1/readfile.c
[root@teartamonov dir1]# chmod 700 /home/guest/dir1/simpleid2
```

Рис. 3.5: Меняем владельца на root и забираем все права у всех кроме владельца

Проверяем. (рис. [3.6])

```
[guest@teartamonov dir1]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@teartamonov dir1]$
```

Рис. 3.6: guest не может прочесть readfile.c

Попробуем прочитать readfile.c с помощью readfile. (рис. [3.7])

```
[guest@teartamonov dir1]$ ./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;
}
[guest@teartamonov dir1]$ ./readfile /etc/shadow
```

Рис. 3.7: Успешно

Попробуем прочитать /etc/shadow с помощью readfile. (рис. [3.8])

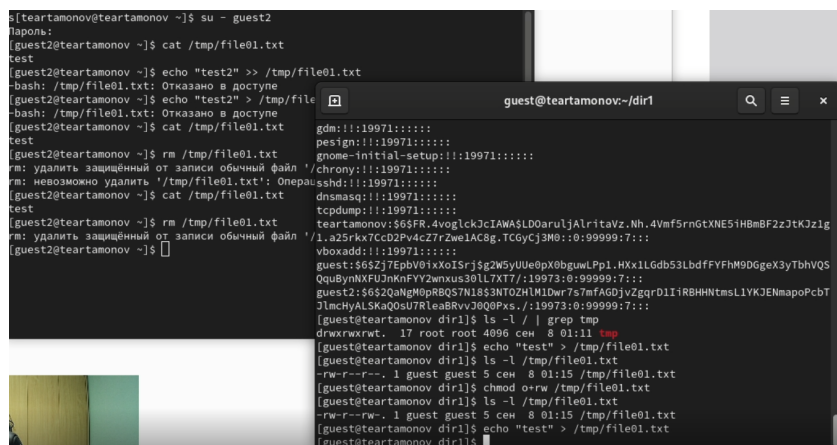
```

cockpit-ws:!!:19971:!!!!:
cockpit-wsinstance:!!:19971:!!!!:
rtkit:!!:19971:!!!!:
pipewire:!!:19971:!!!!:
libstoragemgmt:!*:19971:!!!!:
flatpak:!!:19971:!!!!:
colord:!!:19971:!!!!:
clevis:!!:19971:!!!!:
setroubleshoot:!!:19971:!!!!:
gdm:!!:19971:!!!!:
pesign:!!:19971:!!!!:
gnome-initial-setup:!!:19971:!!!!:
chrony:!!:19971:!!!!:
sshd:!!:19971:!!!!:
dnsmasq:!!:19971:!!!!:
tcpdump:!!:19971:!!!!:
teartamonov:$6$FR.4voglckJcIAWA$LD0aruljAlritaVz.Nh.4Vmf5rNgTxE5iHBmBF2zJtKJzlg
1.a25rkx7CcD2Pv4cZ7rZwe1AC8g.TCGyCj3M0::0:99999:7:::
vboxadd:!!:19971:!!!!:
guest:$6$Zj7EpbV0ixXoISrj$g2W5yUue0pX0bgwLpP1.HXx1LGdb53LbdfFYFhM9DGeX3yTbhVQS
QquBynNXFUJnKnFY2wnxus30lL7XT7/:19973:0:99999:7:::
guest2:$6$2QaNgM0pRBQS7N18$3NTOZHlM1Dwr7s7mfAGDjvZgqrD1iRBHHNtmsL1YKJENmapoPcbT
JlmcHyALSKaQ0sU7RleaBRvvJ0Q0Pxs./:19973:0:99999:7:::
[guest@teartamonov dir1]$

```

Рис. 3.8: Успешно

Найдем директорию tmp, создадим там файл от имени guest и от имени guest2
попробуем выполнить с ним разные действия. (рис. [3.9])



```

[teartamonov@teartamonov ~]$ su - guest2
[guest2@teartamonov ~]$ cat /tmp/file01.txt
test
[guest2@teartamonov ~]$ echo "test2" >> /tmp/file01.txt
-bash: /tmp/file01.txt: Отказано в доступе
[guest2@teartamonov ~]$ echo "test2" > /tmp/file01.txt
-bash: /tmp/file01.txt: Отказано в доступе
[guest2@teartamonov ~]$ cat /tmp/file01.txt
test
[guest2@teartamonov ~]$ rm /tmp/file01.txt
rm: удалить защищенный от записи обычный файл
rm: невозможно удалить '/tmp/file01.txt': Ошибка доступа
[guest2@teartamonov ~]$ cat /tmp/file01.txt
test
[guest2@teartamonov ~]$ rm /tmp/file01.txt
rm: удалить защищенный от записи обычный файл
[guest2@teartamonov ~]$

```

```

[guest@teartamonov dir1]$ ls -l | grep tmp
drwxrwxrwt. 17 root root 4096 сен  8 01:11 tmp
[guest@teartamonov dir1]$ echo "test" > /tmp/file01.txt
[guest@teartamonov dir1]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 сен  8 01:15 /tmp/file01.txt
[guest@teartamonov dir1]$ chmod o+rw /tmp/file01.txt
[guest@teartamonov dir1]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 сен  8 01:15 /tmp/file01.txt
[guest@teartamonov dir1]$ echo "test" > /tmp/file01.txt
[guest@teartamonov dir1]$

```

Рис. 3.9: Можем только читать файл, все что связано с изменением запрещено

Уберем параметр -t и попробуем еще раз. (рис. [3.10])

```
[root@teartamonov dir1]# chmod -t /tmp
[root@teartamonov dir1]#
[guest2@teartamonov ~]$ rm /tmp/file01.txt
rm: удалить защищённый от записи обычный файл '/tmp/file01.txt'? y
[guest2@teartamonov ~]$ [guest@teartamonov dir1]
```

Рис. 3.10: Теперь изменение не для владельца открыто

4 Выводы

Изучили механизмы изменения идентификаторов, применения SetUID и Sticky-битов. Получили практические навыки работы в консоли с дополнительными атрибутами. Рассмотрели работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

Список литературы

1. Sticky bit [Электронный ресурс]. Wikimedia Foundation, 2024. URL: https://ru.wikipedia.org/wiki/Sticky_bit.