

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

---

Соболев М. С.

6 октября 2022

Российский университет дружбы народов, Москва, Россия

# Информация

---

- Соболев Максим Сергеевич
- Студент 4 курса, 1032192035
- Направление: Бизнес-информатика
- Российский университет дружбы народов
- sobolek322lorek@gmail.com

# Вводная часть

---

- Продолжаем изучать дискреционное разграничение прав в Linux.

- Объектом исследования являются: исследование влияния дополнительных атрибутов
- Закрепление теоретических основ

- Получение практических навыков работы с дополнительными атрибутами

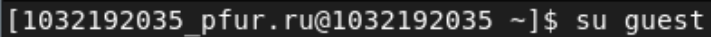
- Отчет по ранее выполненной работе



# **Выполнение лабораторной работы часть 1**

---

Входим в систему от имени пользователя guest.

A terminal window with a dark background. The prompt is [1032192035\_pfur.ru@1032192035 ~]\$ and the command being entered is su guest.

```
[1032192035_pfur.ru@1032192035 ~]$ su guest
```

Figure 1: 1

## Создаем программу



```
GNU nano 5.6.1 simpleid.c
#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;
}
```

Figure 2: 2

Компилируем программу и убеждаемся, что файл программы создан: gcc simpleid.c -o simpleid

```
[guest@1032192035 ~]$ gcc simpleid.c -o simpleid
```

Figure 3: 3

Выполняем программу simpleid: ./simpleid

```
[guest@1032192035 ~]$ ./simpleid
uid = 1001, gid = 1001
[guest@1032192035 ~]$
```

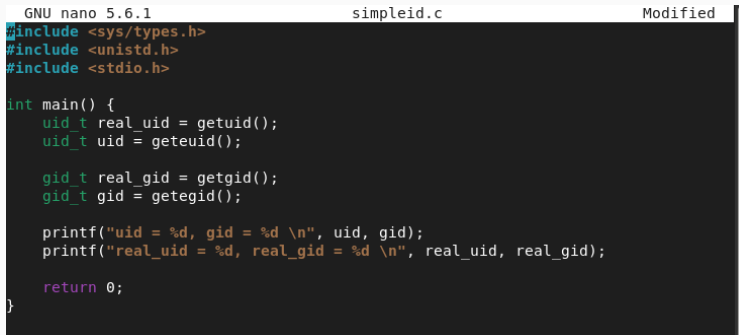
 ## 5

Выполняем системную программу id: id и сравниваем полученный результат с данными предыдущего пункта задания.

```
uid=1001, gid=1001
[guest@1032192035 ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Figure 4: 5

Усложняем программу, добавив вывод действительных идентификаторов



```
GNU nano 5.6.1                                simpleid.c                                Modified
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>

int main() {
    uid_t real_uid = getuid();
    uid_t uid = geteuid();

    gid_t real_gid = getgid();
    gid_t gid = getegid();

    printf("uid = %d, gid = %d \n", uid, gid);
    printf("real_uid = %d, real_gid = %d \n", real_uid, real_gid);

    return 0;
}
```

Figure 5: 6

Получившуюся программу называем simpleid2.c.

Компилируем и запускаем simpleid2.c: `gcc simpleid2.c -o simpleid2 ./simpleid2`

```
[root@1032192035 1032192035_pfur.ru]# chown root:guest /home/guest/simpleid2
[root@1032192035 1032192035_pfur.ru]# chmod u+s /home/guest/simpleid
simpleid    simpleid2    simpleid.c
[root@1032192035 1032192035_pfur.ru]# chmod u+s /home/guest/simpleid2
```

Figure 6: 7

От имени суперпользователя выполняем команды: `chown root:guest /home/guest/simpleid2 chmod u+s /home/guest/simpleid2`

```
[root@1032192035 1032192035_pfur.ru]# chown root:guest /home/guest/simpleid2
[root@1032192035 1032192035_pfur.ru]# chmod u+s /home/guest/simpleid
simpleid    simpleid2  simpleid.c
[root@1032192035 1032192035_pfur.ru]# chmod u+s /home/guest/simpleid2
```

Figure 7: 8



Повысьте временно свои права с помощью `su`. `sudo` позволяет выполнить команду от имени суперпользователя. `su` позволяет войти от имени другого пользователя, в том числе `root`.

Выполняем проверку правильности установки новых атрибутов и смены владельца файла simpleid2: `ls -l simpleid2`

```
[guest@1032192035 ~]$ ls -l simpleid2
-rwsrwxr-x. 1 root guest 26008 Oct  4 17:01 simpleid2
```

**Figure 8: 10**

Запускаем simpleid2 и id:./simpleid2 id и сравниваем результаты

```
[guest@1032192035 ~]$ ./simpleid2
uid = 0, gid = 1001
real_uid = 1001, real_gid = 1001
[guest@1032192035 ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Figure 9: 11

Продельываем тоже самое относительно SetGID-бита

```
[root@1032192035 1032192035_pfur.ru]# chmod g+s /home/guest/simpleid2
```

Figure 10: 12.1

```
[guest@1032192035 ~]$ ./simpleid2
uid = 0, gid = 1001
real_uid = 1001, real_gid = 1001
[guest@1032192035 ~]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Figure 11: 12.2

## Создаем программу readfile.c:



```
GNU nano 3.0.1 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;
}
```

Figure 12: 13

Компилируем её. `gcc readfile.c -o readfile`

```
[guest@1032192035 ~]$ gcc readfile.c -o readfile
```

**Figure 13: 14**

Сменяем владельца у файла readfile.c и изменяем права так, чтобы только суперпользователь мог прочитать его, а guest не мог

```
[guest@1032192035 ~]$ chmod 700 readfile.c  
[guest@1032192035 ~]$ chown root:root readfile.c
```

Figure 14: 15

Проверяем, что пользователь guest не может прочитать файл readfile.c.

```
[guest@1032192035 ~]$ ls -l readfile.c
-rwx-----. 1 root root 464 Oct  4 17:34 readfile.c
[guest@1032192035 ~]$ cat readfile.c
cat: readfile.c: Permission denied
```

Figure 15: 16

Получившуюся программу называем simpleid2.c.



Меняем у программы readfile владельца и устанавливаем SetU'D-бит.

```
[root@1032192035 guest]# chmod u+s readfile  
[root@1032192035 guest]# chown root:root readfile  
[root@1032192035 guest]# ls -l readfile  
-rwxrwxr-x. 1 root root 25952 Oct  4 17:34 readfile
```

Figure 16: 17

Проверяем, может ли программа readfile прочитать файл readfile.c?

```
[guest@1032192035 ~]$ ./readfile readfile.c
@n0000000000000000P000@v@0h000G000<00h000V@000>@G00V"01G0000\01h000v0R0q00000000
0>@h0000000 000p@`0000@x0000#0000#000$000T$000h$000$000$000$000$000$000
0$000%000!%000;%000I%000b%000x%0000%0000%0000%0000%0000+0000+000,000),000,0
000,0000,0000,0000,0000,000
-000-000-0001-000P-000i-000.0000.0000.000/000-/0000/0
P00/000!p@030000d@8
0
00900000/000I0000v0R0q00BY00
00x86_64./readfilereadfile.cSHELL=/bin/bashSESSION_MANAGER=local/unix:@/tmp/.ICE
-unix/1457,unix/unix:/tmp/.ICE-unix/1457COLORTERM=truecolorHISTCONTROL=ignoredup
sXDG_MENU_PREFIX=gnome-HOSTNAME=1032192035.localHISTSIZE=100SSH_AUTH_SOCK=/run/
user/1000/keyring/sshXMODIFIERS=@im=ibusDESKTOP_SESSION=gnomePWD=/home/guestXDG_
SESSION_DESKTOP=gnomeLOGNAME=guestXDG_SESSION_TYPE=waylandSYSTEMD_EXEC_PID=1482X
AUTHORITY=/home/guest/.xauth4UdukBGDM_LANG=en_US.UTF-8HOME=/home/guestUSERNAME=1
032192035_pfur.ruLANG=en_US.UTF-8LS_COLORS=rs=0:di=01;34:ln=01;36:mh=00:pi=40;33
:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or=40;31;01:mi=01;37;41:su=37;41:sg=3
0;43:ca=30;41:tw=30;42:ow=34;42:st=37;44:ex=01;32:*.tar=01;31:*.tgz=01;31:*.arc
=01;31:*.arj=01;31:*.taz=01;31:*.lha=01;31:*.lz4=01;31:*.lzh=01;31:*.lzma=01;31:
*.tlz=01;31:*.txz=01;31:*.tzo=01;31:*.t7z=01;31:*.zip=01;31:*.z=01;31:*.dz=01;31:
```

Figure 17: 18

Чтение файла невозможно, программа выдаёт segfault при попытке чтения

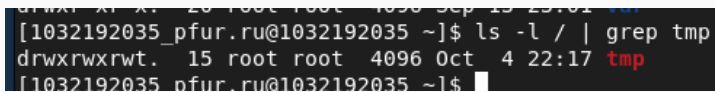
Проверяем, может ли программа readfile прочитать файл /etc/shadow?

```
[guest@1032192035 ~]$ ./readfile /etc/shadow
@@00G00000000000000000PN0G0@@v@00
000x0J0
0V@0گ>@000l0000000^>000
00000000N0G0
```

Figure 18: 19

Чтение файла невозможно, программа выдаёт segfault при попытке чтения файлов. При этом попытка чтения файла без атрибута +s, принадлежащего пользователю guest происходит без проблем \*\*\* # Выполнение лабораторной работы часть 2

Выясняем, установлен ли атрибут Sticky на директории /tmp, для чего выполняем команду `ls -l / | grep tmp`



```
[1032192035_pfur.ru@1032192035 ~]$ ls -l / | grep tmp
drwxrwxrwt. 15 root root 4096 Oct 4 22:17 tmp
[1032192035_pfur.ru@1032192035 ~]$
```

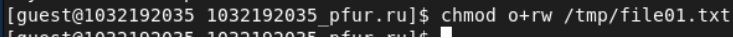
Figure 19: 2.1

От имени пользователя guest создаем файл file01.txt в директории /tmp со словом test: `echo "test" > /tmp/file01.txt`

```
[guest@1032192035 1032192035_pfur.ru]$ echo "test" > /tmp/file01.txt
```

**Figure 20: 2.2**

Просматриваем атрибуты у только что созданного файла и разрешаем чтение и запись для категории пользователей «все остальные»: `ls -l /tmp/file01.txt`  
`chmod o+rw /tmp/file01.txt` `ls -l /tmp/file01.txt`



```
[guest@1032192035 1032192035_pfur.ru]$ chmod o+rw /tmp/file01.txt  
[guest@1032192035 1032192035_pfur.ru]$
```

**Figure 21: 2.3**

От пользователя guest2 попробуем прочитать файл /tmp/file01.txt: cat /tmp/file01.txt

```
[guest2@1032192035 1032192035_pfur.ru]$ cat /tmp/file01.txt  
test
```

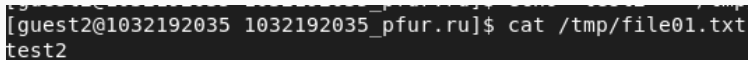
## 5

От пользователя guest2 попробуем дозаписать в файл /tmp/file01.txt слово test2 командой echo "test2" > /tmp/file01.txt

```
[guest2@1032192035 1032192035_pfur.ru]$ echo "test2" > /tmp/file01.txt
```

**Figure 22: 2.5**

Проверьте содержимое файла командой `cat /tmp/file01.txt`



```
[guest2@1032192035 1032192035_pfur.ru]$ cat /tmp/file01.txt  
test2
```

**Figure 23: 2.6**

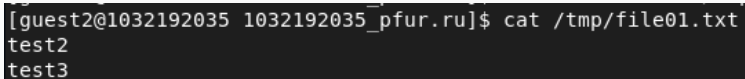


От пользователя `guest2` попробуем записать в файл `/tmp/file01.txt` слово `test3`, стерев при этом всю имеющуюся в файле информацию командой `echo "test3" > /tmp/file01.txt`

```
[guest2@1032192035 1032192035_pfur.ru]$ echo "test3" >> /tmp/file01.txt
```

**Figure 24: 2.7**

Проверяем содержимое файла командой `cat /tmp/file01.txt`



```
[guest2@1032192035 1032192035_pfur.ru]$ cat /tmp/file01.txt  
test2  
test3
```

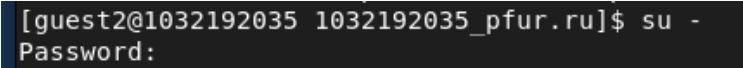
**Figure 25: 2.8**

От пользователя guest2 попробуем удалить файл /tmp/file01.txt командой rm /tmp/file01.txt

```
test3  
[guest2@1032192035 1032192035_pfur.ru]$ rm /tmp/file01.txt  
rm: cannot remove '/tmp/file01.txt': Operation not permitted  
[guest2@1032192035 1032192035_pfur.ru]$
```

**Figure 26: 2.9**

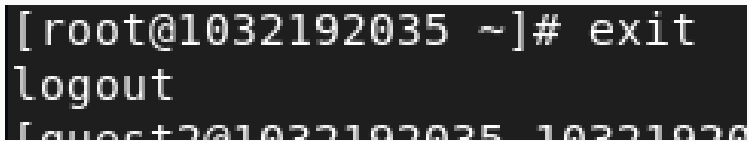
Повышаем свои права до суперпользователя следующей командой `su -` и выполняем после этого команду, снимающую атрибут `t` (Sticky-бит) с директории `/tmp`: `chmod -t /tmp`

A terminal window with a dark background. The prompt is `[guest2@1032192035 1032192035_pfur.ru]$`. The user has entered `su -`. The next line shows `Password:` with a cursor.

```
[guest2@1032192035 1032192035_pfur.ru]$ su -  
Password:
```

**Figure 27: 2.10**

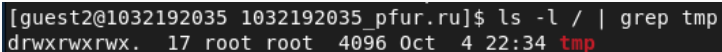
Покидаем режим суперпользователя командой exit



```
[root@1032192035 ~]# exit
logout
[quest2@1032192035 ~]#
```

Figure 28: 2.11

От пользователя guest2 проверяем, что атрибуты t у директории /tmp нет: `ls -l / | grep tmp`



```
[guest2@1032192035 1032192035_pfur.ru]$ ls -l / | grep tmp
drwxrwxrwx. 17 root root 4096 Oct 4 22:34 tmp
```

Figure 29: 2.12

Запись и дозапись работают без изменений, удаление файла стало доступно

```
[guest2@1032192035 1032192035_pfur.ru]$ echo "test2" > /tmp/file01.txt  
[guest2@1032192035 1032192035_pfur.ru]$ cat /tmp/file01.txt  
test2  
[guest2@1032192035 1032192035_pfur.ru]$ echo "test3" >> /tmp/file01.txt  
[guest2@1032192035 1032192035_pfur.ru]$ cat /tmp/file01.txt  
test2  
test3  
[guest2@1032192035 1032192035_pfur.ru]$ rm /tmp/file01.txt
```

Figure 30: 2.13

Да удалось

2.14



Повышаем свои права до суперпользователя и возвращаем атрибут `t` на директорию `/tmp`: `su - chmod +t /tmp exit`

```
[guest2@1032192035 1032192035_pfur.ru]$ su
Password:
[root@1032192035 1032192035_pfur.ru]# chmod +t /tmp
[root@1032192035 1032192035_pfur.ru]# exit
exit
[guest2@1032192035 1032192035_pfur.ru]#
```

**Figure 31: 2.15**

## Выводы

---

Мы изучили механизмы изменения идентификаторов, применение 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.