

Лабораторная работа №5

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Ван И

7 октября 2023

Российский университет дружбы народов, Москва, Россия

Информация

- Ван И
- студент НФИбд-02-20
- Российский университет дружбы народов
- 1032198069@pfur.ru
- <https://github.com/WangYi157>

Логические объекты файловой системы (файлы) являются носителями своеобразных меток, которые привычно называют правами доступа. Некоторые метки действительно означают право выполнения определенного действия пользователя над этим объектом. Важно изучить их для дальнейшего применения на практике.

- Атрибуты файлов
- Дистрибутив Rocky
- Дискреционное разграничение доступа

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.

Выполнение работы

От имени пользователя guest создадим программу simpleid.c, скомпилируем ее и убедимся, что файл создан. Выполним команды ./simpleid и id и убедимся, что полученные данные совпадают

```
[yi@yi ~]$ su guest
Пароль:
[guest@yi yi]$ cd /home/guest
[guest@yi ~]$ nano simpleid.c
[guest@yi ~]$ gcc simpleid.c -o simpleid
[guest@yi ~]$ ./simpleid
uid=1001, gid=1001
[guest@yi ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@yi ~]$
```

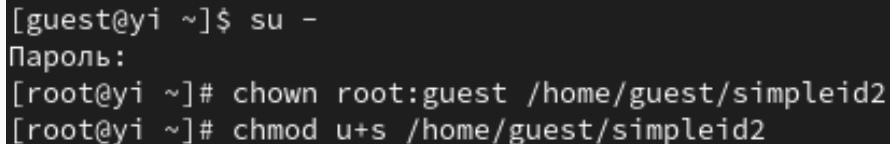
Рис. 1: Использование команд ./simpleid и id

Усложним программу и запишем ее в файл simpleid2.c. Запустим получившуюся программу

```
[guest@yi ~]$ nano simpleid2.c
[guest@yi ~]$ gcc simpleid2.c -o simpleid2
[guest@yi ~]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@yi ~]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@yi ~]$
```

Рис. 2: Запуск программы simpleid2

От имени суперпользователя установим новые атрибуты и сменим владельца файла simpleid2



```
[guest@yi ~]$ su -  
Пароль:  
[root@yi ~]# chown root:guest /home/guest/simpleid2  
[root@yi ~]# chmod u+s /home/guest/simpleid2
```

Рис. 3: Установки новых атрибутов и смена владельца файла simpleid2

Выполним команду `./simpleid2`

```
[root@yi ~]# cd /home/guest
[root@yi guest]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@yi guest]# id
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
```

Рис. 4: Использование команд `./simpleid2`

Проделаем то же самое относительно SetGID-бита

```
[root@yi guest]# chmod g+s simpleid2
[root@yi guest]# ls -l simpleid2
-rwsr-sr-x. 1 root guest 26064 окт  7 19:41 simpleid2
[root@yi guest]# exit
выход
[guest@yi ~]$ ./simpleid2
e_uid=0, e_gid=1001
real_uid=1001, real_gid=1001
[guest@yi ~]$ id
uid=1001(guest) gid=1001(guest) rpyнны=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[guest@yi ~]$
```

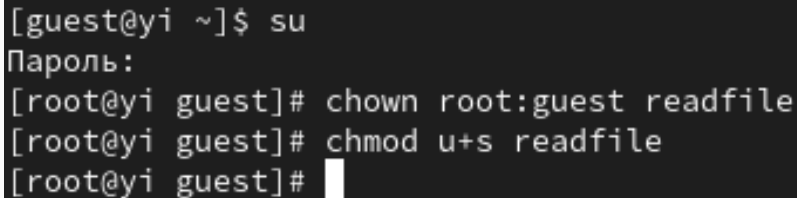
Рис. 5: Операции с SetGID-битом

Создание программы

Создадим и скомпилируем программу readfile.c. Сменим владельца у файла readfile.c и изменим права так, чтобы только суперпользователь (root) мог прочитать его, а guest не мог

```
[guest@yi ~]$ nano readfile.c
[guest@yi ~]$ gcc readfile.c -o readfile
[guest@yi ~]$ su
Пароль:
[root@yi guest]# chown root:guest readfile.c
[root@yi guest]# chmod 700 readfile.c
[root@yi guest]# exit
exit
[guest@yi ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@yi ~]$
```

Сменим у программы readfile владельца и установим SetUID-бит



```
[guest@yi ~]$ su
Пароль:
[root@yi guest]# chown root:guest readfile
[root@yi guest]# chmod u+s readfile
[root@yi guest]#
```

Рис. 7: Работа с параметрами readfile

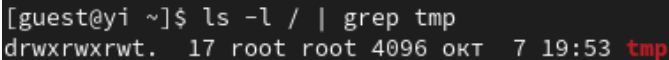
Проверим, может ли программа readfile прочитать файл readfile.c

```
[guest@yi ~]$ ./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);
}
```


Проверим, может ли программа readfile прочитать файл /etc/shadow

```
[guest@yi ~]$ ./readfile /etc/shadow
root:$6$H5khZBmzzDa28eDh$QbTX8spVK7Ha8UZ540qYI4e9Jfqx2bNb8AKEq5asAn15LxRT.MARt86ovDI9eCgBG3UKcuv3ufbPYvrkBeuPr.:0:99999:7:::
bin:!:19469:0:99999:7:::
daemon:!:19469:0:99999:7:::
adm:!:19469:0:99999:7:::
lp:!:19469:0:99999:7:::
sync:!:19469:0:99999:7:::
shutdown:!:19469:0:99999:7:::
halt:!:19469:0:99999:7:::
mail:!:19469:0:99999:7:::
operator:!:19469:0:99999:7:::
games:!:19469:0:99999:7:::
ftp:!:19469:0:99999:7:::
nobody:!:19469:0:99999:7:::
systemd-coredump:!!:19616::::::
dbus:!!:19616::::::
polkitd:!!:19616::::::
avahi:!!:19616::::::
rtkit:!!:19616::::::
sssd:!!:19616::::::
pipewire:!!:19616::::::
libstoragemgmt:!:19616::::::
systemd-oom:!:19616::::::
tss:!!:19616::::::
geoclue:!!:19616::::::
cockpit-ws:!!:19616::::::
cockpit-wsinstance:!!:19616::::::
flatpak:!!:19616::::::
colord:!!:19616::::::
clevi:!!:19616::::::
setroubleshoot:!!:19616::::::
gdm:!!:19616::::::
pesign:!!:19616::::::
```

Выясним, установлен ли атрибут Sticky на директории /tmp



```
[guest@yi ~]$ ls -l / | grep tmp
drwxrwxrwt. 17 root root 4096 окт  7 19:53 tmp
```

Рис. 10: Чтение атрибутов директории /tmp

От имени пользователя guest создадим файл file01.txt в директории /tmp со словом test. Просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные»

```
[guest@yi ~]$ ls -l / | grep tmp
drwxrwxrwt. 17 root root 4096 окт  7 19:53 tmp
[guest@yi ~]$ echo "test" > /tmp/file01.txt
[guest@yi ~]$ ls -l /tmp/file01.txt
-rw-r--r--. 1 guest guest 5 окт  7 19:55 /tmp/file01.txt
[guest@yi ~]$ chmod o+rw /tmp/file01.txt
[guest@yi ~]$ ls -l /tmp/file01.txt
-rw-r--rw-. 1 guest guest 5 окт  7 19:55 /tmp/file01.txt
[guest@yi ~]$
```

Рис. 11: Создание файла /tmp/file01.txt

Исследование Sticky-бита

От пользователя guest2 попробуем прочитать файл /tmp/file01.txt. Далее попробуем дозаписать в файл /tmp/file01.txt слово test2, записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию. После этого попробуем удалить данный файл

```
[guest@yi ~]$ su guest2
Пароль:
[guest2@yi guest]$ echo "test2" >> /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@yi guest]$ echo "test3" > /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@yi guest]$ cat /tmp/file01.txt
test
[guest2@yi guest]$ rm /tmp/file01.txt
rm: удалить защищённый от записи обычный файл '/tmp/file01.txt'? y
rm: невозможно удалить '/tmp/file01.txt': Операция не позволена
[guest2@yi guest]$
```

Исследование Sticky-бита

От имени суперпользователя снимем атрибут `t` с директории `/tmp`. От пользователя `guest2` проверим, что атрибута `t` у директории `/tmp` нет. Повторим предыдущие шаги. Теперь мы можем удалить файл

```
[root@yi ~]# chmod -t /tmp
[root@yi ~]# exit
выход
[guest2@yi guest]$ ls -l / | grep tmp
drwxrwxrwx. 18 root root 4096 окт  7 20:00 tmp
[guest2@yi guest]$ echo "test2" >> /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@yi guest]$ echo "test3" > /tmp/file01.txt
bash: /tmp/file01.txt: Отказано в доступе
[guest2@yi guest]$ cat /tmp/file01.txt
test
[guest2@yi guest]$ rm /tmp/file01.txt
rm: удалить защищённый от записи обычный файл '/tmp/file01.txt'? y
[guest2@yi guest]$
```

В ходе лабораторной работы мне удалось:

- Изучить механизмы изменения идентификаторов, применения SetUID- и Sticky-битов.
- Получить практических навыков работы в консоли с дополнительными атрибутами.
- Рассмотреть работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.