

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Маммедгулыев Максат НФИбд-01-19

3 октября, 2022, Москва, Россия

Российский Университет Дружбы Народов

Цели и задачи

- SUID - разрешение на установку идентификатора пользователя. Это бит разрешения, который позволяет пользователю запускать исполняемый файл с правами владельца этого файла.
- SGID - разрешение на установку идентификатора группы. Принцип работы очень похож на SUID с отличием, что файл будет запускаться пользователем от имени группы, которая владеет файлом.

Цель лабораторной 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.

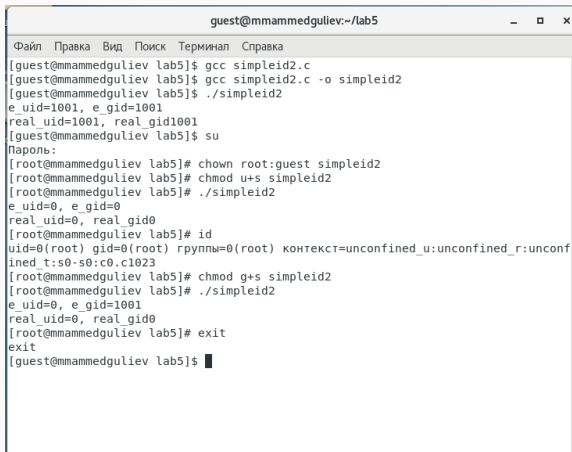
Выполнение лабораторной работы

Программа simpleid

```
[guest@mmammedguliev mmammedguliev]$ getenforce
Permissive
[guest@mmammedguliev mmammedguliev]$ cd
[guest@mmammedguliev ~]$ cd lab5
[guest@mmammedguliev lab5]$ gcc simpleid.c
[guest@mmammedguliev lab5]$ gcc simpleid.c -o simpleid
[guest@mmammedguliev lab5]$ ./simpleid
uid=1001, gid=1001
[guest@mmammedguliev lab5]$ id
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfi
ned_r:unconfined_t:s0-s0:c0.c1023
[guest@mmammedguliev lab5]$
```

Figure 1: результат программы simpleid

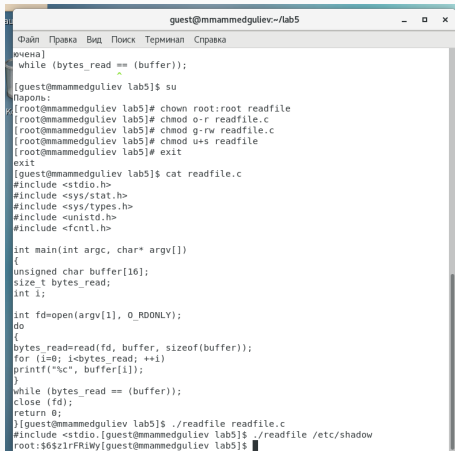
Программа simpleid2



```
guest@mmammedguliev:~/lab5
Файл Правка Вид Поиск Терминал Справка
[guest@mmammedguliev lab5]$ gcc simpleid2.c
[guest@mmammedguliev lab5]$ gcc simpleid2.c -o simpleid2
[guest@mmammedguliev lab5]$ ./simpleid2
e_uid=1001, e_gid=1001
real_uid=1001, real_gid=1001
[guest@mmammedguliev lab5]$ su
Пароль:
[root@mmammedguliev lab5]# chown root:guest simpleid2
[root@mmammedguliev lab5]# chmod u+s simpleid2
[root@mmammedguliev lab5]# ./simpleid2
e_uid=0, e_gid=0
real_uid=0, real_gid=0
[root@mmammedguliev lab5]# id
uid=0(root) gid=0(root) группы=0(root) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@mmammedguliev lab5]# chmod g+s simpleid2
[root@mmammedguliev lab5]# ./simpleid2
e_uid=0, e_gid=1001
real_uid=0, real_gid=0
[root@mmammedguliev lab5]# exit
exit
[guest@mmammedguliev lab5]$
```

Figure 2: результат программы simpleid2

Программа readfile

A screenshot of a terminal window titled 'guest@mmammedguliev:~/lab5'. The window shows a series of commands and their outputs. The user starts in the 'guest' shell, runs 'su' to become root, then uses 'chown' and 'chmod' to set permissions on 'readfile.c'. After exiting root, the user runs 'cat readfile.c' to display the source code of the program. The code includes standard headers, defines a buffer, opens a file, reads it, and prints the contents. Finally, the user runs './readfile readfile.c' to execute the program, which outputs the contents of the shadow file.

```
guest@mmammedguliev:~/lab5
Файл  Правка  Вид  Поиск  Терминал  Справка
ючена)
while (bytes_read == (buffer));

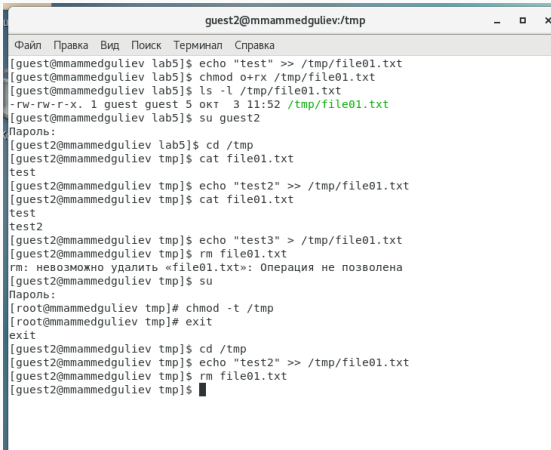
[guest@mmammedguliev lab5]$ su
Пароль:
[root@mmammedguliev lab5]# chown root:root readfile
[root@mmammedguliev lab5]# chmod o-r readfile.c
[root@mmammedguliev lab5]# chmod g-rw readfile.c
[root@mmammedguliev lab5]# chmod u+s readfile
[root@mmammedguliev lab5]# exit
exit
[guest@mmammedguliev lab5]$ cat readfile.c
#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>
#include <fcntl.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 == (buffer));
    close (fd);
    return 0;
}[guest@mmammedguliev lab5]$ ./readfile readfile.c
#include <stdio.h>[guest@mmammedguliev lab5]$ ./readfile /etc/shadow
root:$6$z1rFRiWy[guest@mmammedguliev lab5]$
```

Figure 3: результат программы readfile

Исследование Sticky-бита



```
guest2@mmammedguliev:tmp
Файл  Правка  Вид  Поиск  Терминал  Справка
[guest@mmammedguliev lab5]$ echo "test" >> /tmp/file01.txt
[guest@mmammedguliev lab5]$ chmod o+rx /tmp/file01.txt
[guest@mmammedguliev lab5]$ ls -l /tmp/file01.txt
-rw-rw-r-x. 1 guest guest 5 окт  3 11:52 /tmp/file01.txt
[guest@mmammedguliev lab5]$ su guest2
Пароль:
[guest2@mmammedguliev lab5]$ cd /tmp
[guest2@mmammedguliev tmp]$ cat file01.txt
test
[guest2@mmammedguliev tmp]$ echo "test2" >> /tmp/file01.txt
[guest2@mmammedguliev tmp]$ cat file01.txt
test
test2
[guest2@mmammedguliev tmp]$ echo "test3" > /tmp/file01.txt
[guest2@mmammedguliev tmp]$ rm file01.txt
rm: невозможно удалить «file01.txt»: Операция не позволена
[guest2@mmammedguliev tmp]$ su
Пароль:
[root@mmammedguliev tmp]# chmod -t /tmp
[root@mmammedguliev tmp]# exit
exit
[guest2@mmammedguliev tmp]$ cd /tmp
[guest2@mmammedguliev tmp]$ echo "test2" >> /tmp/file01.txt
[guest2@mmammedguliev tmp]$ rm file01.txt
[guest2@mmammedguliev tmp]$
```

Figure 4: исследование Sticky-бита

Выводы

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