

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Камила Мухтарова НПИбд-01-20

28 сентября, 2023, 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@kamuhtarova ~]$  
[guest@kamuhtarova ~]$ mkdir lab5  
[guest@kamuhtarova ~]$ cd lab5/  
[guest@kamuhtarova lab5]$ touch simpleid.c  
[guest@kamuhtarova lab5]$  
[guest@kamuhtarova lab5]$ gcc simpleid.c  
[guest@kamuhtarova lab5]$ gcc simpleid.c -o simpleid  
[guest@kamuhtarova lab5]$ ./simpleid  
uid=1001, gid=1001  
[guest@kamuhtarova lab5]$ id  
uid=1001(guest) gid=1001(guest) rpynnw=1001(guest),10(wheel) контекст=unconfined  
_u:unconfined_r:unconfined_t:s0-s0:c0.c1023  
[guest@kamuhtarova lab5]$
```

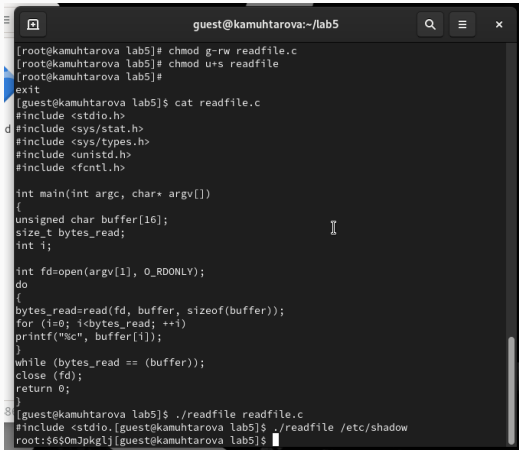
Figure 1: результат программы simpleid

Программа simpleid2

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

Figure 2: результат программы simpleid2

Программа readfile

A terminal window titled 'guest@kamuhtarova:~/lab5' with search, menu, and close icons. It shows the compilation and execution of a C program named 'readfile.c'. The program uses standard headers and opens a file specified as an argument in read-only mode. It reads the file in chunks of 16 bytes and prints each character. The execution output shows the contents of the file '/etc/shadow' being printed character by character.

```
guest@kamuhtarova:~/lab5
[root@kamuhtarova lab5]# chmod g-rw readfile.c
[root@kamuhtarova lab5]# chmod u+s readfile
[root@kamuhtarova lab5]#
exit
[guest@kamuhtarova 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@kamuhtarova lab5]$ ./readfile readfile.c
#include <stdio.h>[guest@kamuhtarova lab5]$ ./readfile /etc/shadow
root:$6$0mJpkg1j[guest@kamuhtarova lab5]$
```

Figure 3: результат программы readfile

Исследование Sticky-бита

```
[guest@kamuhtarova lab5]$  
[guest@kamuhtarova lab5]$ echo test >> /tmp/file01.txt  
[guest@kamuhtarova lab5]$ chmod o+rx /tmp/file01.txt  
[guest@kamuhtarova lab5]$ su guest2  
Пароль:  
[guest2@kamuhtarova lab5]$ cd /tmp/  
[guest2@kamuhtarova tmp]$ cat file01.txt  
test  
[guest2@kamuhtarova tmp]$ echo test2 >> /tmp/file01.txt  
bash: /tmp/file01.txt: Отказано в доступе  
[guest2@kamuhtarova tmp]$ cat file01.txt  
test  
[guest2@kamuhtarova tmp]$ rm file01.txt  
rm: удалить защищённый от записи обычный файл 'file01.txt'? y  
rm: невозможно удалить 'file01.txt': Операция не позволена  
[guest2@kamuhtarova tmp]$ su  
Пароль:  
[root@kamuhtarova tmp]# chmod -t /tmp/  
[root@kamuhtarova tmp]#  
exit  
[guest2@kamuhtarova tmp]$ echo test2 >> /tmp/file01.txt  
bash: /tmp/file01.txt: Отказано в доступе  
[guest2@kamuhtarova tmp]$ rm file01.txt  
rm: удалить защищённый от записи обычный файл 'file01.txt'? y  
[guest2@kamuhtarova 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.