

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Любимова Таисия НБИ-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.

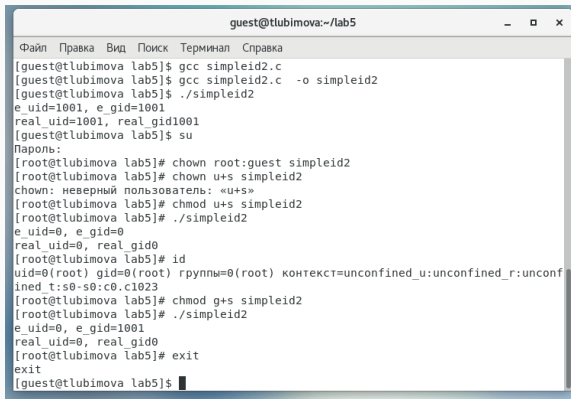
Выполнение лабораторной работы

Программа simpleid

```
permissive
[guest@tlubimova ~]$
[guest@tlubimova ~]$ cd
[guest@tlubimova ~]$ mkdir lab5
[guest@tlubimova ~]$ cd lab5/
[guest@tlubimova lab5]$ touch simpleid.c
[guest@tlubimova lab5]$ touch simpleid2.c
[guest@tlubimova lab5]$ touch readfile.c
[guest@tlubimova lab5]$ gedit simpleid.c
[guest@tlubimova lab5]$
[guest@tlubimova lab5]$ gcc simpleid.c
[guest@tlubimova lab5]$ gcc simpleid.c -o simpleid
[guest@tlubimova lab5]$ ./simpleid
uid=1001, gid=1001
[guest@tlubimova lab5]$ id
uid=1001(guest) gid=1001(guest) rгруппы=1001(guest) контекст=unconfined_u:unconfi
ned_r:unconfined_t:s0-s0:c0.c1023
[guest@tlubimova lab5]$
```

Figure 1: результат программы simpleid

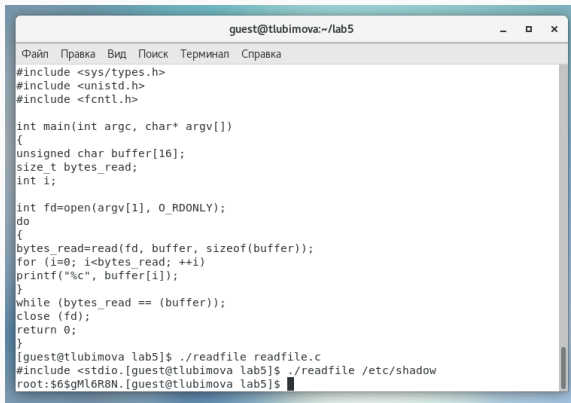
Программа simpleid2



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

Figure 2: результат программы simpleid2

Программа readfile



The screenshot shows a terminal window with the title bar "guest@tlubimova:~/lab5". The menu bar includes "Файл", "Правка", "Вид", "Поиск", "Терминал", and "Справка". The terminal content displays the source code of the "readfile" program, which includes headers for `<sys/types.h>`, `<unistd.h>`, and `<fcntl.h>`. The `main` function opens a file specified by `argv[1]` in read-only mode, reads its contents into a 16-byte buffer, and prints each byte. The execution sequence is as follows:

```
[guest@tlubimova lab5]$ ./readfile readfile.c
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>

int main(int argc, char* argv[])
{
    unsigned char buffer[16];
    size_t bytes_read;
    int i;

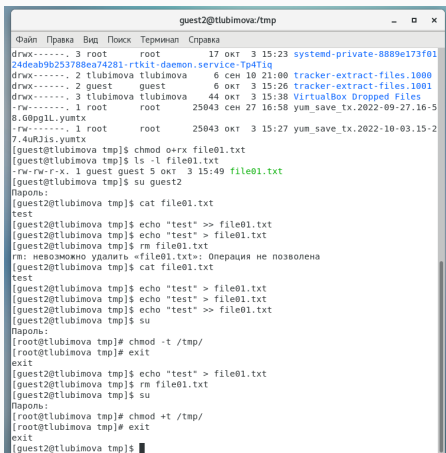
    int fd=open(argv[1], O_RDONLY);
    do
    {
        bytes_read=read(fd, buffer, sizeof(buffer));
        for (i=0; i<bytes_read; ++i)
            printf("%c", buffer[i]);
    }
    while (bytes_read == (buffer));
    close (fd);
    return 0;
}
[guest@tlubimova lab5]$ ./readfile readfile.c
#include <stdio.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@tlubimova lab5]$ ./readfile /etc/shadow
root:$6$gMl6R8N.[guest@tlubimova lab5]$
```

Figure 3: результат программы readfile

Исследование Sticky-бита



```
guest2@tlubimova:/tmp
Файл  Правка  Вид  Поиск  Терминал  Справка
drwx-----, 3 root      root      17 окт  3 15:23  systemd-private-8889e173f01
24deab9b253788ea74281-rtkit-daemon.service-Tp4Tiq
drwx-----, 2 tlubimova tlubimova  6 сен 10 21:00  tracker-extract-files.1000
drwx-----, 2 guest     guest      6 окт  3 15:26  tracker-extract-files.1001
drwx-----, 3 tlubimova tlubimova  44 окт  3 15:38  VirtualBox Dropped Files
-rw-----, 1 root      root      25043 сен 27 16:58  yum_save_tx.2022-09-27.16-5
8.00pg1L.yumtx
-rw-----, 1 root      root      25043 окт  3 15:27  yum_save_tx.2022-10-03.15-2
7.4uRjis.yumtx
[guest2@tlubimova tmp]$ chmod o+rx file01.txt
[guest2@tlubimova tmp]$ ls -l file01.txt
-rw-rw-r-x. 1 guest guest 5 окт  3 15:49  file01.txt
[guest2@tlubimova tmp]$ su guest2
Пароль:
[guest2@tlubimova tmp]$ cat file01.txt
test
[guest2@tlubimova tmp]$ echo "test" >> file01.txt
[guest2@tlubimova tmp]$ echo "test" > file01.txt
[guest2@tlubimova tmp]$ rm file01.txt
rm: невозможно удалить «file01.txt»: Операция не позволена
[guest2@tlubimova tmp]$ cat file01.txt
test
[guest2@tlubimova tmp]$ echo "test" > file01.txt
[guest2@tlubimova tmp]$ echo "test" > file01.txt
[guest2@tlubimova tmp]$ echo "test" >> file01.txt
[guest2@tlubimova tmp]$ su
Пароль:
[root@tlubimova tmp]# chmod -t /tmp/
[root@tlubimova tmp]# exit
exit
[guest2@tlubimova tmp]$ echo "test" > file01.txt
[guest2@tlubimova tmp]$ rm file01.txt
[guest2@tlubimova tmp]$ su
Пароль:
[root@tlubimova tmp]# chmod +t /tmp/
[root@tlubimova tmp]# exit
exit
[guest2@tlubimova tmp]$
```

Figure 4: исследование Sticky-бита

Выводы

Результаты выполнения лабораторной работы

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.