

Лабораторная работа №5:
Дискреционное разграничение прав в
Linux. Исследование влияния
дополнительных атрибутов

дисциплина: Информационная безопасность

Швец Сергей Сергеевич

2021, 13 November

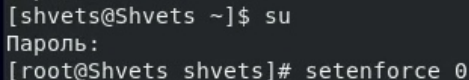
Цель работы

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 работы

отключение системы запретов

Так как программы с установленным битом SetUID могут представлять большую брешь в системе безопасности, в современных системах используются дополнительные механизмы защиты. Чтобы система защиты SELinux не мешала выполнению заданий работы, она была отключена до следующей перезагрузки.

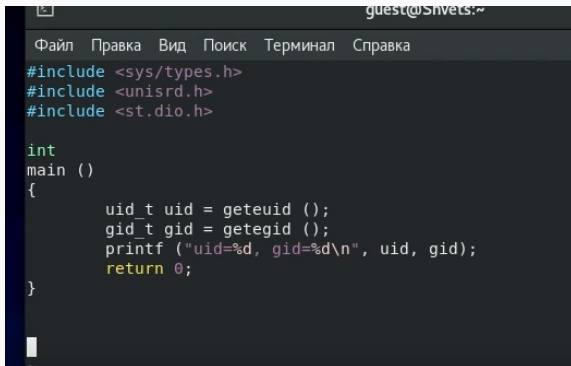
A terminal window with a dark background. The first line shows the user 'shvets' at host 'Shvets' in the home directory, typing 'su'. The second line shows the password prompt 'Пароль:'. The third line shows the user 'root' at host 'Shvets' as 'shvets', typing 'setenforce 0'.

```
[shvets@Shvets ~]$ su
Пароль:
[root@Shvets shvets]# setenforce 0
```

Figure 1: отключение системы запретов

Создание программы simpleid

Создание программы simpleid.

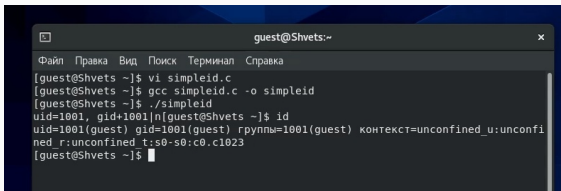


```
guest@Shvets:~  
Файл Правка Вид Поиск Терминал Справка  
#include <sys/types.h>  
#include <unistd.h>  
#include <stdio.h>  
  
int  
main ()  
{  
    uid_t uid = geteuid ();  
    gid_t gid = getegid ();  
    printf ("uid=%d, gid=%d\n", uid, gid);  
    return 0;  
}
```

Figure 2: Guest

компиляция, запуск программы, а также команды id

Далее программа была скомпилирована и запущена, а также было проведено сравнение результатов работы программы с командой id

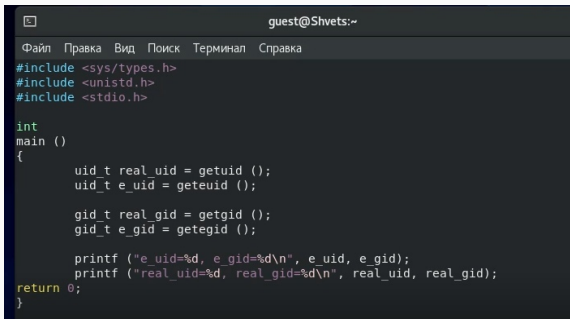


```
guest@Shvets:~  
Файл  Правка  Вид  Поиск  Терминал  Справка  
[guest@Shvets ~]$ vi simpleid.c  
[guest@Shvets ~]$ gcc simpleid.c -o simpleid  
[guest@Shvets ~]$ ./simpleid  
uid=1001, gid=1001|n|guest@Shvets ~|$ id  
uid=1001(guest) gid=1001(guest) rгруппы=1001(guest) контекст=unconfined_u:unconfi  
ned_r:unconfined_t:s0-s0:c0.c1023  
[guest@Shvets ~]$
```

Figure 3: директория и пользователь

модификация программы

Далее программа была модифицирована несколькими атрибутами.



```
guest@Shvets:~  
Файл Правка Вид Поиск Терминал Справка  
#include <sys/types.h>  
#include <unistd.h>  
#include <stdio.h>  
  
int  
main ()  
{  
    uid_t real_uid = getuid ();  
    uid_t e_uid = geteuid ();  
  
    gid_t real_gid = getgid ();  
    gid_t e_gid = getegid ();  
  
    printf ("e_uid=%d, e_gid=%d\n", e_uid, e_gid);  
    printf ("real_uid=%d, real_gid=%d\n", real_uid, real_gid);  
    return 0;  
}
```

Figure 4: модификация программы

Выполнение команд из пункта 8.

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

Figure 5: 8 пункт

Написание программы readfile.

```
#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 6: readfile

Проверка доступа.

```
[guest@Shvets ~]$ vi readfile.c
[guest@Shvets ~]$ gcc readfile.c -o readfile
[guest@Shvets ~]$ ls -l readfile.c
-rwx-----. 1 root guest 454 ноя 13 22:35 readfile.c
[guest@Shvets ~]$ cat readfile.c
cat: readfile.c: Отказано в доступе
[guest@Shvets ~]$ ./readfile /etc/shadow
```

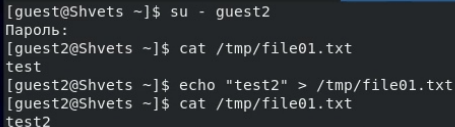
Figure 7: Расширенные атрибуты

Создание файла file01.

```
[guest@Shvets ~]$ echo "test" > /tmp/file01.txt
[guest@Shvets ~]$ ls -l /tmpfile01.txt
ls: невозможно получить доступ к '/tmpfile01.txt': Нет такого файла или каталога
[guest@Shvets ~]$ ls -l /tmp/file01.txt
-rw-rw-r--. 1 guest guest 5 ноя 13 22:45 /tmp/file01.txt
[guest@Shvets ~]$ echo "test" > /tmp/file01.txt
[guest@Shvets ~]$ chmod o+rw /tmp/file01.txt
[guest@Shvets ~]$ ls -l /tmp/file01.txt
-rw-rw-rw-. 1 guest guest 5 ноя 13 22:46 /tmp/file01.txt
[guest@Shvets ~]$
```

Figure 8: file01

попытка прочитать и изменить файле file01 от имени пользователя, не являющегося владельцем файла



```
[guest@Shvets ~]$ su - guest2
Пароль:
[guest2@Shvets ~]$ cat /tmp/file01.txt
test
[guest2@Shvets ~]$ echo "test2" > /tmp/file01.txt
[guest2@Shvets ~]$ cat /tmp/file01.txt
test2
```

Figure 9: guest2

Модификация атрибутов

модификация атрибутов директории от имени пользователя guest2

```
[guest2@Shvets ~]$ su
Пароль:
[root@Shvets guest2]# rm /tmp/file01.txt
rm: удалить обычный файл '/tmp/file01.txt'? n
[root@Shvets guest2]# chmod -t /tmp
[root@Shvets guest2]# exit
exit
[guest2@Shvets ~]$ ls -l / | grep tmp
drwxrwxrwx. 15 root root 4096 ноя 13 22:51 tmp
[guest2@Shvets ~]$ su
Пароль:
[root@Shvets guest2]# chmod +t /tmp
[root@Shvets guest2]# ex
```

Figure 10: атрибуты

Выводы

Мной были изучены механизмы изменения идентификаторов, применение SetUID- и Stickyбиты. Получены практические навыки работы в консоли с дополнительными атрибутами. Рассмотрена работа механизмов смены идентификаторов процесса пользователей, а также влияние бита Sticky на запись и удаление файлов.