

# Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов.

---

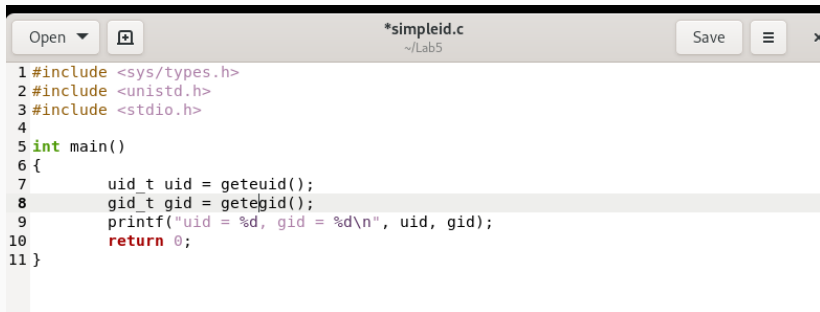
Критский Сергей Димитриевич

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.

## Ход выполнения работы

---

# Программа 1



```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4
5 int main()
6 {
7     uid_t uid = geteuid();
8     gid_t gid = getegid();
9     printf("uid = %d, gid = %d\n", uid, gid);
10    return 0;
11 }
```

Рис. 1: Код программы

```
function it appears in
guest@SDKritskiy Lab5]$ gcc simpleid.c -o simpleid
guest@SDKritskiy Lab5]$ ./simpleid
id = 1001, gid = 1001
guest@SDKritskiy Lab5]$ id
id=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
guest@SDKritskiy Lab5]$
```

Рис. 2: Результат выполнения

## Программа 2



The image shows a code editor window with the title 'simpleid.c' and a path '~\Lab5'. The editor contains C code that uses `getuid()`, `geteuid()`, `getgid()`, and `getegid()` to retrieve and print user and group IDs. The code is as follows:

```
1 #include <sys/types.h>
2 #include <unistd.h>
3 #include <stdio.h>
4
5 int main()
6 {
7     uid_t real_uid = getuid();
8     uid_t e_uid = geteuid();
9
10    gid_t real_gid = getgid();
11    gid_t e_gid = getegid();
12
13    printf("e_uid = %d, e_gid = %d\n", e_uid, e_gid);
14    printf("real_uid = %d, real_gid = %d\n", real_uid, real_gid);
15    return 0;
16 }
```

Рис. 3: Код программы

```
[guest@SDKritskiy Lab5]$ id
uid=1001(guest) gid=1001(guest) groups=1001(guest) context=unconfined_u:unco
ed_r:unconfined_t:s0-s0:c0.c1023
[guest@SDKritskiy Lab5]$ gcc simpleid.c -o simpleid2
[guest@SDKritskiy Lab5]$ ./simpleid2
e_uid = 1001, e_gid = 1001
real_uid = 1001, real_gid = 1001
[guest@SDKritskiy Lab5]$
```

Рис. 4: Пользователь guest

```
[root@SDKritskiy Lab5]# ./simpleid2
e_uid = 0, e_gid = 0
real_uid = 0, real_gid = 0
[root@SDKritskiy Lab5]# id
uid=0(root) gid=0(root) groups=0(root) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[root@SDKritskiy Lab5]#
```

Рис. 5: Суперпользователь





The image shows a code editor window with two tabs: 'simpleid.c' and 'readfile.c'. The 'readfile.c' tab is active and displays the following C code:

```
1 #include <fcntl.h>
2 #include <stdio.h>
3 #include <sys/stat.h>
4 #include <sys/types.h>
5 #include <unistd.h>
6
7 int main(int argc, char* argv[])
8 {
9     unsigned char buffer[16];
10    size_t bytes_read;
11    int i;
12
13    int fd = open (argv[1], O_RDONLY);
14    do
15    {
16        bytes_read = read(fd, buffer, sizeof(buffer));
17        for(i = 0; i < bytes_read; ++i) printf("%c", buffer[i]);
18    }
19
20    while(bytes_read == sizeof(buffer));
21    close(fd);
22    return 0;
23 }
```

Рис. 6: Код программы

```
[root@SDKritskiy Lab5]# ./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);
    return 0;
}
```

Рис. 7: Результат выполнения на примере readfile.c

```
[root@SDKritskiy Lab5]# su guest2
[guest2@SDKritskiy Lab5]$ cat /tmp/file01.txt
test
[guest2@SDKritskiy Lab5]$ echo "test2" > /tmp/file01.txt
[guest2@SDKritskiy Lab5]$ cat /tmp/file01.txt
test2
[guest2@SDKritskiy Lab5]$ echo "test3" > /tmp/file01.txt
[guest2@SDKritskiy Lab5]$ cat /tmp/file01.txt
test3
[guest2@SDKritskiy Lab5]$ rm /tmp/file01.txt
rm: cannot remove '/tmp/file01.txt': Operation not permitted
```

Рис. 8: Взаимодействие с файлом в директории со Sticky-битом

```
[guest2@SDKritskiy Lab5]$ su
Password:
[root@SDKritskiy Lab5]# chmod -t /tmp
[root@SDKritskiy Lab5]# exit
exit
[guest2@SDKritskiy Lab5]$ ls -l / | grep tmp
drwxrwxrwx. 16 root root 4096 Oct  8 11:43 tmp
[guest2@SDKritskiy Lab5]$ echo "test3" > /tmp/file01.txt
[guest2@SDKritskiy Lab5]$ echo "test4" > /tmp/file01.txt
[guest2@SDKritskiy Lab5]$ cat /tmp/file01.txt
test4
[guest2@SDKritskiy Lab5]$ rm /tmp/file01.txt
[guest2@SDKritskiy Lab5]$ su
Password:
[root@SDKritskiy Lab5]# chmod +t /tmp
[root@SDKritskiy Lab5]#
```

Рис. 9: Взаимодействие с файлом в директории без Sticky-бита

Я изучил механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получил практические навыки работы в консоли с дополнительными атрибутами. Рассмотрел работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.