

Лабораторная работа № 5

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Сидоракин Кирилл Вячеславович НБибд-01-18

Содержание

Цель работы	1
Выполнение лабораторной работы	2
Вывод.....	5

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 лабораторной работы

Создаем программу simpleid.c:

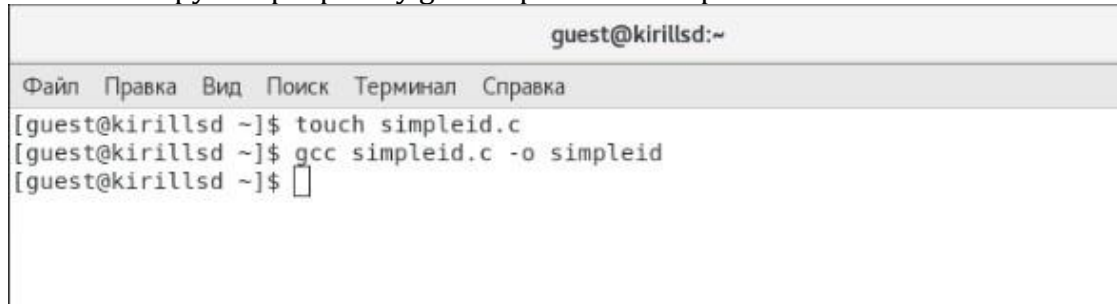


The screenshot shows a code editor window titled "guest@kirillsd:~". The menu bar includes "Файл", "Правка", "Вид", "Поиск", "Терминал", and "Справка". The terminal output shows the command `touch simpleid.c` being executed. Below the terminal, the file `simpleid.c` is open, showing the following C code:

```
#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;
}
```

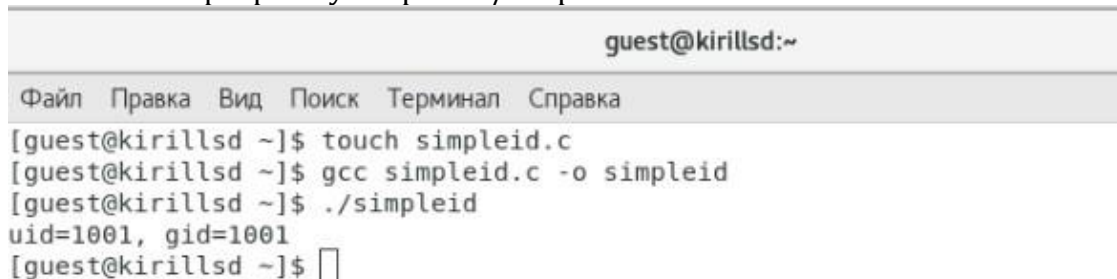
Скомпилируем программу `gcc simpleid.c -o simpleid`



The screenshot shows a terminal window titled "guest@kirillsd:~". The terminal output shows the following commands and their results:

```
[guest@kirillsd ~]$ touch simpleid.c
[guest@kirillsd ~]$ gcc simpleid.c -o simpleid
[guest@kirillsd ~]$
```

Выполняем программу simpleid: `./simpleid`



The screenshot shows a terminal window titled "guest@kirillsd:~". The terminal output shows the following commands and their results:

```
[guest@kirillsd ~]$ touch simpleid.c
[guest@kirillsd ~]$ gcc simpleid.c -o simpleid
[guest@kirillsd ~]$ ./simpleid
uid=1001, gid=1001
[guest@kirillsd ~]$
```

Выполняем системную программу id:

```
guest@kirillsd:~  
Файл Правка Вид Поиск Терминал Справка  
[guest@kirillsd ~]$ touch simpleid.c  
[guest@kirillsd ~]$ gcc simpleid.c -o simpleid  
[guest@kirillsd ~]$ ./simpleid  
uid=1001, gid=1001  
[guest@kirillsd ~]$ id  
uid=1001(guest) gid=1001(guest) группы=1001(guest) контекст=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023  
[guest@kirillsd ~]$
```

Усложняем программу, добавив вывод действительных идентификаторов:

```
[kirillsd@kirillsd ~]$ touch simpleid2.c  
[kirillsd@kirillsd ~]$  
simpleid.c simpleid2.c  
#include <sys/types.h>  
#include <unistd.h>  
#include <stdio.h>  
  
int  
main ()  
{  
    uid_t real_uid = getuid ();  
    uid_t e_uid = getegid ();  
  
    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;  
}
```

Компилируем и запускаем simpleid2.c:

```
[guest@kirillsd ~]$ gcc simpleid2.c -o simpleid2  
[guest@kirillsd ~]$ ./simpleid2  
e_uid=1001, e_gid=1001  
real_uid=1001, real_gid=1001
```

От имени суперпользователя выполняем команды:

```
[root@kirillsd guest]# chown root:guest /home/guest/simpleid2  
[root@kirillsd guest]# chmod u+s /home/guest/simpleid2
```

Используем sudo

```
[root@kirillsd guest]# sudo
usage: sudo -h | -K | -k | -V
usage: sudo -v [-AknS] [-g group] [-h host] [-p prompt] [-u user]
usage: sudo -l [-AknS] [-g group] [-h host] [-p prompt] [-U user]
usage: sudo [-AbEHknPS] [-r role] [-t type] [-C num] [-g group] [-T timeout] [-u user] [VAR=value] [-i|-s] [<command>]
usage: sudo -e [-AknS] [-r role] [-t type] [-C num] [-g group] [-
```

Выполняем проверку правильности установки новых атрибутов и смены владельца файла simpleid2

```
[root@kirillsd guest]# ls -l simpleid2
-rwsrwxr-x. 1 root guest 8616 ноя 12 00:56 simpleid2
[root@kirillsd guest]#
```

Запускаем simpleid2 и id

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

Создаем программу readfile.c

simpleid.c	simpleid2.c	*readfile.c
<pre>#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; }</pre>		

Откомпилируем её

```
[guest@kirillsd ~]$ touch readfile.c
[guest@kirillsd ~]$ gcc readfile.c -o readfile
```

Вывод

В результате выполнения лабораторной работы мы получили навыки изучения механизмов изменения идентификаторов, применения SetUID- и Sticky-битов. Получили практические навыки работы в консоли с дополнительными атрибутами. Рассмотрели работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.