отчёта по лабораторной работе 5

Отчёт по лабораторной работе: Идентификаторы пользователя и группы, биты SetUID, SetGID и Sticky

Чилеше Лупупа

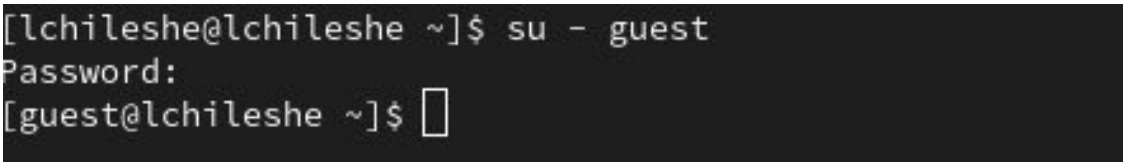
Содержание

1 Цель работы

Изучить механизм управления правами доступа в операционной системе Linux на уровне идентификаторов пользователя (UID) и группы (GID), а также на практике освоить работу с бітами SetUID, SetGID и Sticky, их влияние на выполнение программ и управление файлами. Научиться использовать системные вызовы для получения реальных и эффективных UID/GID, изменять права доступа и анализировать безопасность при совместной работе пользователей.

2 Создание программы

1. Вход в систему от имени guest



2. Создание и компиляция программы simpleid.c

#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", uid, gid); return 0; }

Компиляция: gcc simpleid.c -o simpleid

```
[guest@lchileshe ~]$ gcc simpleid.c -o simpleid
[guest@lchileshe ~]$ ./simpleid
```

4. Запуск simpleid

./simpleid

5. Сравнение с системной программой id id

6. Усложнённая программа simpleid2.c

#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;
```

}

```
 GNU nano 5.6.1                    simpleid2.c
#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;
}
```

7. Компиляция и запуск:

gcc simpleid2.c -o simpleid2 ./simpleid2

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

8. Изменение владельца и установка SetUID:

sudo chown root:guest simpleid2 sudo chmod u+s simpleid2

9. Пояснение:
   - chown root:guest — устанавливает владельцем root, а группой — guest.
   - chmod u+s — устанавливает SetUID-бит, при этом программа будет выполняться с правами владельца (то есть root).
10. Проверка установки прав:

ls -l simpleid2

```
[guest@lchileshe ~]$ ls -l simpleid2
-rwsr-xr-x. 1 root guest 17704 Apr 19 14:12 simpleid2
```

11. Запуск программы и id:

./simpleid2 id

12. Повтор для SetGID:

sudo chown guest:root simpleid2 sudo chmod g+s simpleid2

2.1 Программа readfile.c

13. Код:

#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; }

14. Компиляция:

gcc readfile.c -o readfile

15. Изменение владельца и прав:

sudo chown root:root readfile.c sudo chmod 600 readfile.c

16. Проверка как guest:

cat readfile.c

17. Установка SetUID для readfile:

sudo chown root:guest readfile sudo chmod u+s readfile

18. Проверка доступа к readfile.c через программу:

./readfile readfile.c

19. Проверка доступа к /etc/shadow:

./readfile /etc/shadow

Результат: Вывод невозможен — файл защищён дополнительными механизмами безопасности, включая SELinux или AppArmor.

2.2 Исследование Sticky-бита

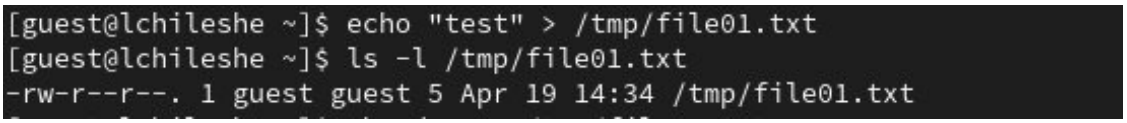
1. Проверка Sticky-бита у /tmp:

ls -ld /tmp

t на конце прав означает установленный Sticky-бит.

2. Создание файла в /tmp:

echo “test” > /tmp/file01.txt



3. Изменение прав:

chmod o+rw /tmp/file01.txt ls -l /tmp/file01.txt

- 4–7. Проверки от имени guest2:

  Чтение файла: успешно

  Добавление через >>: успешно

  Перезапись через >: успешно

8. Проверка содержимого:

cat /tmp/file01.txt

9. Удаление файла от guest2:

rm /tmp/file01.txt

Удаление невозможно — включён Sticky-бит.

10. Снятие Sticky-бита:

su - chmod -t /tmp exit

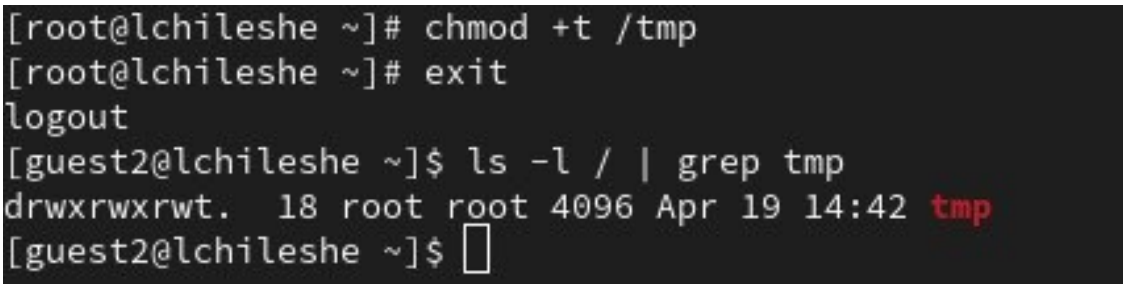
- 11–12. Проверка отсутствия Sticky:

ls -ld /tmp

Результат: отсутствует t

13. Повтор удаления от guest2: Удаление файла теперь возможно, несмотря на то, что пользователь — не владелец.
14. Наблюдение: Sticky-бит предотвращает удаление файлов пользователями, не являющимися владельцами, даже при наличии прав.
15. Восстановление Sticky:

su - chmod +t /tmp exit



2.3 Выводы:

- Реальные и эффективные UID/GID показывают, от имени какого пользователя и группы исполняется процесс.
- SetUID позволяет выполнять программу от имени владельца, SetGID — от имени группы, что может быть полезно для доступа к защищённым ресурсам.
- Sticky-бит обеспечивает защиту от удаления чужих файлов в общем каталоге, например, /tmp.
- Использование этих битов должно быть контролируемо, так как они могут быть источником уязвимостей при неправильной настройке.