Отчёт по лабораторной работе №5. Информационная безопасность

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Выполнила: Данзанова Саяна Зоригтоевна, НПИбд-01-21, 1032217624

Содержание

# 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

# Теоретическое введение

**1. Дополнительные атрибуты файлов Linux**

В Linux существует три основных вида прав — право на чтение (read), запись (write) и выполнение (execute), а также три категории пользователей, к которым они могут применяться — владелец файла (user), группа владельца (group) и все остальные (others). Но, кроме прав чтения, выполнения и записи, есть еще три дополнительных атрибута. [1]

* **Sticky bit**

Используется в основном для каталогов, чтобы защитить в них файлы. В такой каталог может писать любой пользователь. Но, из такой директории пользователь может удалить только те файлы, владельцем которых он является. Примером может служить директория /tmp, в которой запись открыта для всех пользователей, но нежелательно удаление чужих файлов.

* **SUID (Set User ID)**

Атрибут исполняемого файла, позволяющий запустить его с правами владельца. В Linux приложение запускается с правами пользователя, запустившего указанное приложение. Это обеспечивает дополнительную безопасность т.к. процесс с правами пользователя не сможет получить доступ к важным системным файлам, которые принадлежат пользователю root.

* **SGID (Set Group ID)**

Аналогичен suid, но относиться к группе. Если установить sgid для каталога, то все файлы созданные в нем, при запуске будут принимать идентификатор группы каталога, а не группы владельца, который создал файл в этом каталоге.

* **Обозначение атрибутов sticky, suid, sgid**

Специальные права используются довольно редко, поэтому при выводе программы ls -l символ, обозначающий указанные атрибуты, закрывает символ стандартных прав доступа.

Пример:

rwsrwsrwt

*где первая s — это suid, вторая s — это sgid, а последняя t — это sticky bit*

В приведенном примере не понятно, rwt — это rw- или rwx? Определить это просто. Если t маленькое, значит x установлен. Если T большое, значит x не установлен. То же самое правило распространяется и на s.

В числовом эквиваленте данные атрибуты определяются первым символом при четырехзначном обозначении (который часто опускается при назначении прав), например в правах 1777 — символ 1 обозначает sticky bit. Остальные атрибуты имеют следующие числовое соответствие:

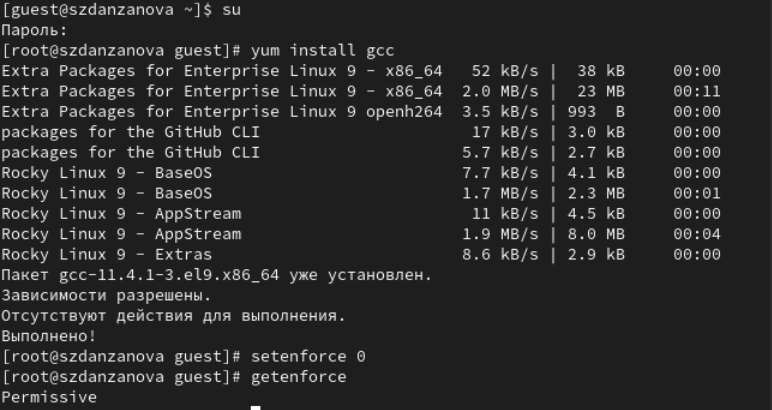
1 — установлен sticky bit  
2 — установлен sgid  
4 — установлен suid

**2. Компилятор GCC**

GСС - это свободно доступный оптимизирующий компилятор для языков C, C++. Собственно программа gcc это некоторая надстройка над группой компиляторов, которая способна анализировать имена файлов, передаваемые ей в качестве аргументов, и определять, какие действия необходимо выполнить. Файлы с расширением .cc или .C рассматриваются, как файлы на языке C++, файлы с расширением .c как программы на языке C, а файлы c расширением .o считаются объектными. [2]

# Выполнение лабораторной работы

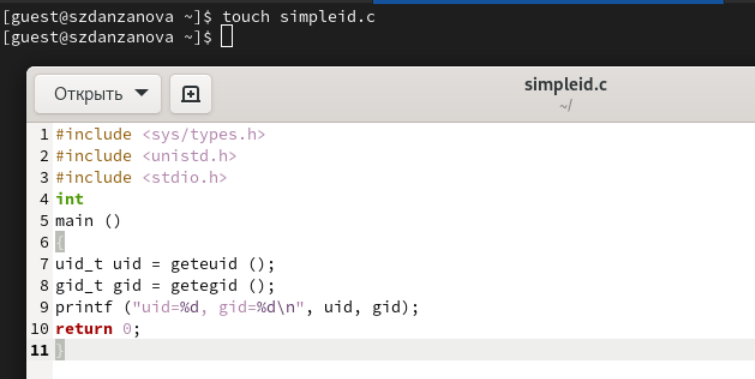
## 5.2.1. Подготовка лабораторного стенда



(рис. 1. Установка gss)

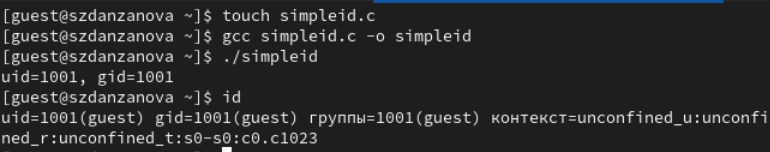
## 5.3.1 Создание программы

1. Войдите в систему от имени пользователя guest.
2. Создайте программу simpleid.c.



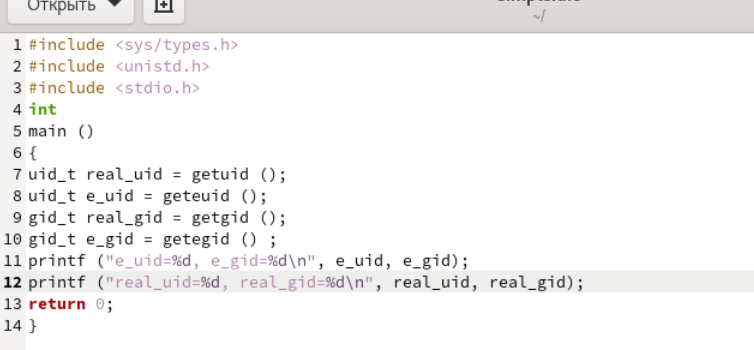
(рис. 2. simpleid.c)

1. Скомплилируйте программу и убедитесь, что файл программы создан: gcc simpleid.c -o simpleid
2. Выполните программу simpleid: ./simpleid
3. Выполните системную программу id: id и сравните полученный вами результат с данными предыдущего пункта задания.



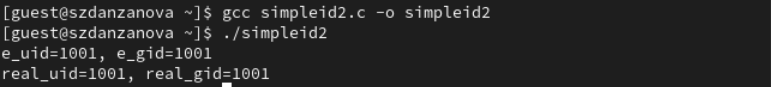
(рис. 3. 3-5 пункты задания лабораторной)

1. Усложните программу, добавив вывод действительных идентификаторов.



(рис. 4. simpleid2.c)

1. Скомпилируйте и запустите simpleid2.c: gcc simpleid2.c -o simpleid2 ./simpleid2

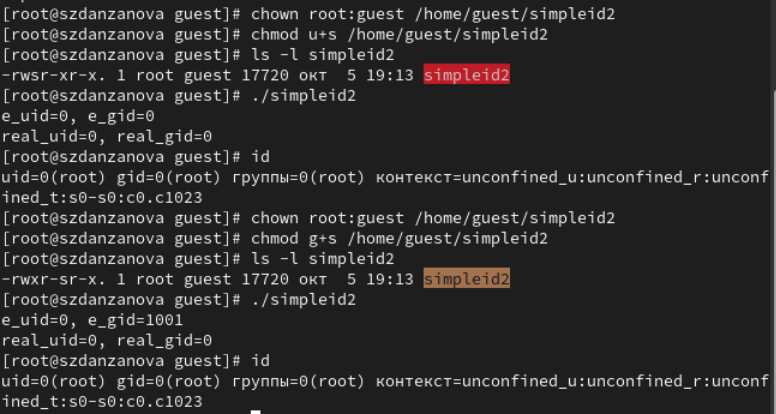


(рис. 5. 7 пункт задания лабораторной)

1. От имени суперпользователя выполните команды: chown root:guest /home/guest/simpleid2 chmod u+s /home/guest/simpleid2
2. Используйте sudo или повысьте временно свои права с помощью su. Поясните, что делают эти команды.

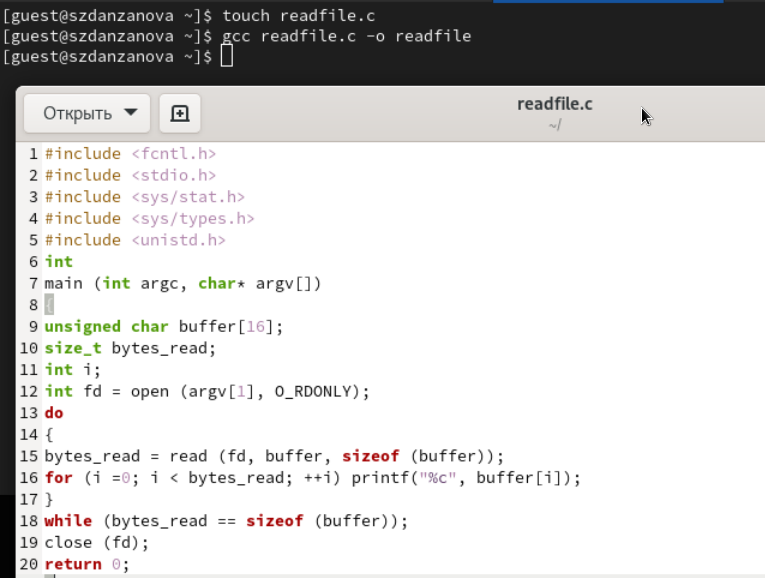
От имени суперпользователя выполнила команды “sudo chown root:guest /home/guest/simpleid2” и “sudo chmod u+s /home/guest/simpleid2”, затем выполнила проверку правильности установки новых атрибутов и смены владельца файла simpleid2 командой “sudo ls -l /home/guest/simpleid2” (рис. 3.9). Этими командами была произведена смена пользователя файла на root и установлен SetUID-бит.

1. Выполните проверку правильности установки новых атрибутов и смены владельца файла simpleid2: ls -l simpleid2
2. Запустите simpleid2 и id: ./simpleid2 id Сравните результаты.
3. Проделайте тоже самое относительно SetGID-бита.



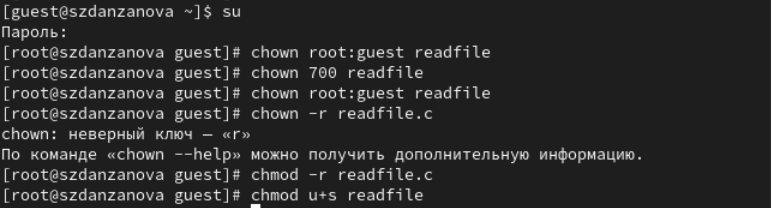
(рис. 6. 8-12 пункты задания лабораторной)

1. Создайте программу readfile.c
2. Откомпилируйте её. gcc readfile.c -o readfile



(рис. 7. readfile.c)

1. Смените владельца у файла readfile.c (или любого другого текстового файла в системе) и измените права так, чтобы только суперпользователь (root) мог прочитать его, a guest не мог.



(рис. 8. chmod)

1. Проверьте, что пользователь guest не может прочитать файл readfile.c.
2. Смените у программы readfile владельца и установите SetU’D-бит.
3. Проверьте, может ли программа readfile прочитать файл readfile.c?
4. Проверьте, может ли программа readfile прочитать файл /etc/shadow? Отразите полученный результат и ваши объяснения в отчёте.



(рис. 9. 16-19 пункты Guest)

## 5.3.2. Исследование Sticky-бита

1. Выясните, установлен ли атрибут Sticky на директории /tmp, для чего выполните команду ls -l / | grep tmp
2. От имени пользователя guest создайте файл file01.txt в директории /tmp со словом test: echo “test” > /tmp/file01.txt
3. Просмотрите атрибуты у только что созданного файла и разрешите чтение и запись для категории пользователей «все остальные»: ls -l /tmp/file01.txt chmod o+rw /tmp/file01.txt ls -l /tmp/file01.txt



(рис. 12. 1-3 пункты)

1. От пользователя guest2 (не являющегося владельцем) попробуйте прочитать файл /tmp/file01.txt: cat /tmp/file01.txt
2. От пользователя guest2 попробуйте дозаписать в файл /tmp/file01.txt слово test2 командой echo “test2” > /tmp/file01.txt

Удалось ли вам выполнить операцию? Нет.

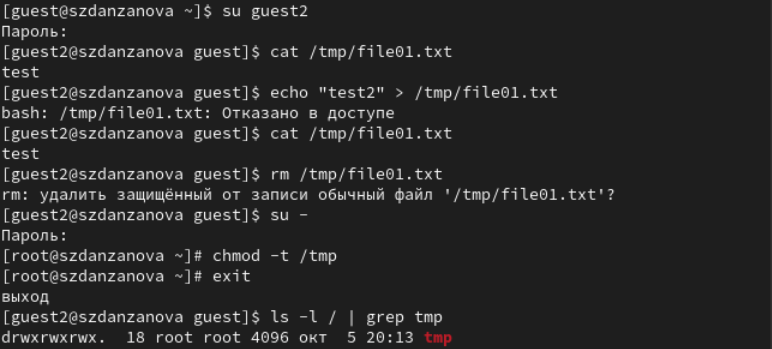
1. Проверьте содержимое файла командой cat /tmp/file01.txt
2. От пользователя guest2 попробуйте записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию командой echo “test3” > /tmp/file01.txt

Удалось ли вам выполнить операцию? Нет.

1. Проверьте содержимое файла командой cat /tmp/file01.txt
2. От пользователя guest2 попробуйте удалить файл /tmp/file01.txt командой rm /tmp/fileOl.txt

Удалось ли вам удалить файл? Нет.

1. Повысьте свои права до суперпользователя следующей командой su и выполните после этого команду, снимающую атрибут t (Sticky-бит) с директории /tmp: chmod -t /tmp
2. Покиньте режим суперпользователя командой exit
3. От пользователя guest2 проверьте, что атрибута t у директории /tmp нет: ls -l / | grep tmp



(рис. 13. 4-12 пункты)

1. Повторите предыдущие шаги. Какие наблюдаются изменения?

При повторении всё получилось.

1. Удалось ли вам удалить файл от имени пользователя, не являющегося его владельцем? Удалось.
2. Повысьте свои права до суперпользователя и верните атрибут t на директорию /tmp: su chmod +t /tmp exit



(рис. 15. Возвращение атрибута)

# Вывод

Были изучены механизмы изменения идентификаторов и применения SetUID- и Sticky-битов. Получены практические навыки работы в консоли с дополнительными атрибутами. Были рассмотрены работа механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов

# Список литературы. Библиография

[0] Методические материалы курса

[1] Дополнительные атрибуты: https://tokmakov.msk.ru/blog/item/141

[2] Компилятор GSS: http://parallel.imm.uran.ru/freesoft/make/instrum.html