Отчет по лабораторной работе 5

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Шалыгин Георгий Эдуардович

Содержание

# 1 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.

# 2 Теоретическое введение

В Linux, как и в любой многопользовательской системе, абсолютно естественным образом возникает задача разграничения доступа субъектов — пользователей к объектам — файлам дерева каталогов.

Один из подходов к разграничению доступа — так называемый дискреционный (от англ, discretion — чье-либо усмотрение) — предполагает назначение владельцев объектов, которые по собственному усмотрению определяют права доступа субъектов (других пользователей) к объектам (файлам), которыми владеют.

Дискреционные механизмы разграничения доступа используются для разграничения прав доступа процессов как обычных пользователей, так и для ограничения прав системных программ в (например, служб операционной системы), которые работают от лица псевдопользовательских учетных записей.

В Linux у каждого файла и каждого каталога есть два владельца: пользователь и группа.

Эти владельцы устанавливаются при создании файла или каталога. Пользователь, который создаёт файл становится владельцем этого файла, а первичная группа, в которую входит этот же пользователь, так же становится владельцем этого файла. Чтобы определить, есть ли у вас как у пользователя права доступа к файлу или каталогу, оболочка проверяет владение ими.

**Это происходит в следующем порядке:**

1. Оболочка проверяет, являетесь ли вы владельцем файла, к которому вы хотите получить доступ. Если вы являетесь этим владельцем, вы получаете разрешения и оболочка прекращает проверку.
2. Если вы не являетесь владельцем файла, оболочка проверит, являетесь ли вы участником группы, у которой есть разрешения на этот файл. Если вы являетесь участником этой группы, вы получаете доступ к файлу с разрешениями, которые для группы установлены, и оболочка прекратит проверку.
3. Если вы не являетесь ни пользователем, ни владельцем группы, вы получаете права других пользователей (Other).

Чтобы увидеть текущие назначения владельца, вы можете использовать команду **ls -l**. Эта команда показывает пользователя и группу-владельца.

Подробнее в [1].

### 2.0.1 Изменение владельца

Чтобы применить соответствующие разрешения, первое, что нужно учитывать, это владение. Для этого есть команда **chown**. Синтаксис этой команды несложен для понимания:

chown кто что

Например, следующая команда меняет владельца каталога /home/account на пользователя linda:

chown linda /home/account

## 2.1 Использование chmod

Для управления правами используется команда **chmod**. При использовании **chmod** вы можете устанавливать разрешения для пользователя (user), группы (group) и других (other). Вы можете использовать эту команду в двух режимах: относительный режим и абсолютный режим. В абсолютном режиме три цифры используются для установки основных разрешений.

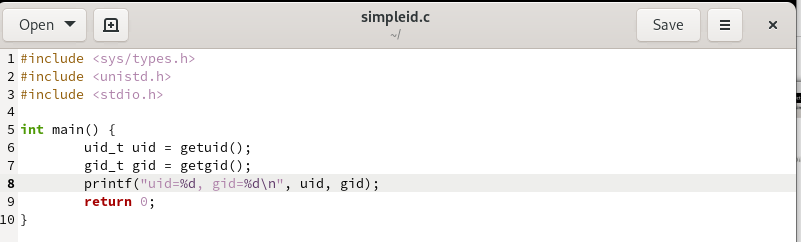
При настройке разрешений рассчитайте необходимое вам значение. Если вы хотите установить чтение, запись и выполнение для пользователя, чтение и выполнение для группы, а также чтение и выполнение для других в файле /somefile, то вы используете следующую команду **chmod**:

chmod 755 /somefile

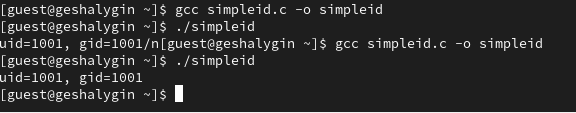
Подробнее в [2].

# 3 Выполнение лабораторной работы

1. От имени пользователя guest2 создадим программу simpleid.c: (fig. 1).

* 
* Рис. 1: simpleid

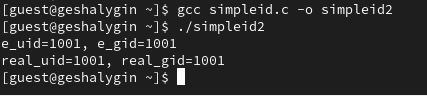
1. Скомпилируем программу, что файл программы создан: gcc simpleid.c -o simpleid (fig:002).

* Выполним программу simpleid и id, в результате получаем правильные значения (fig:002).
* 
* Рис. 2: run simpleid

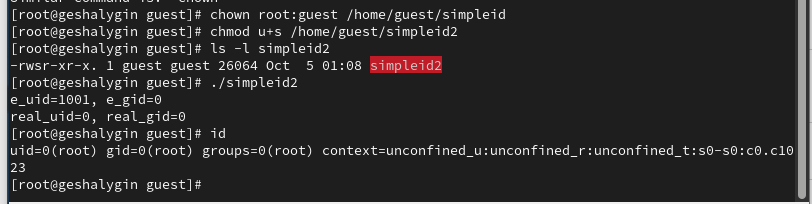
1. Сравнение вывода с id (fig. 3).

* id
* Рис. 3: id

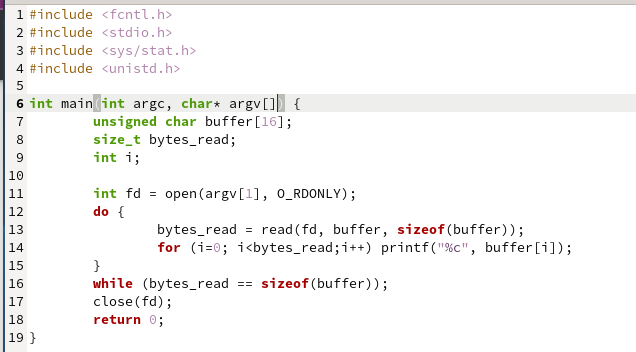
1. Усложним программу, добавив вывод действительных идентификаторов. Скомпилируем и запустим simpleid2.c (fig. 4).

* 
* Рис. 4: simpleid2

1. От имени суперпользователя выполним команды (fig. 5). Изменим владельца и добавим uid бит.

* Выполним проверку правильности установки новых атрибутов и смены владельца файла simpleid2: ls -l simpleid2 11.
* Запустите simpleid2 и id: ./simpleid2 id
* Результаты совпадают.
* 
* Рис. 5: simpleid2 with u+s

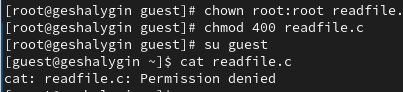
1. Создадим программу readfile.c: (fig. 6).

* 
* Рис. 6: readfile.c

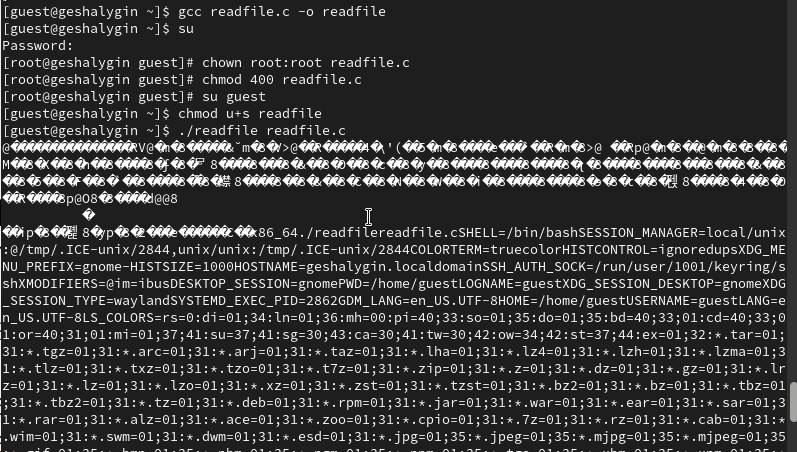
1. Скомпилируем (fig. 7).

* Неудача
* Рис. 7: Неудача

1. Сменим владельца у файла readfile.c и права так, чтобы только суперпользователь (root) мог прочитать его, a guest не мог. (fig. 8). Проверим что guest не имеет доступ к файлу.

* 
* Рис. 8: Проверка прав доступа для новых атрибутов

1. Сменим у программы readfile владельца и установите SetU’D-бит. Проверим, может ли программа readfile прочитать файл readfile.c? Как видим, может(fig. 9).

* 
* Рис. 9: run readfile

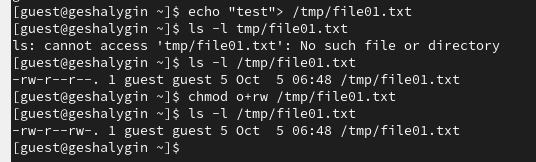
1. Проверим, может ли программа readfile прочитать файл /etc/shadow (fig. 10).

* 
* Рис. 10: etc/shadow

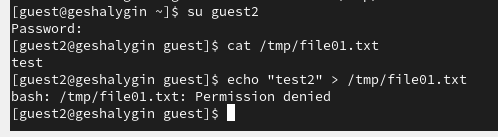
1. Выясним, установлен ли атрибут Sticky на директории /tmp (fig. 11).

* просмотр атрибутов
* Рис. 11: просмотр атрибутов

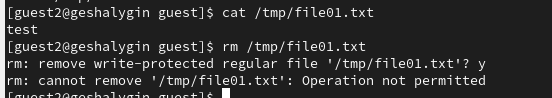
1. От имени пользователя guest создадим файл file01.txt в директории /tmp со словом test. Просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные» (fig. 12).

* 
* Рис. 12: просмотр атрибутов

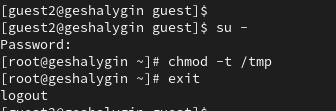
1. От пользователя guest2 (не являющегося владельцем) попробуем прочитать файл /tmp/file01.txt. Доступ открыт. Дозаписать в файл уже нельзя (fig. 13).

* 
* Рис. 13: проверка доступа

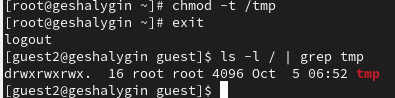
1. Проверьте содержимое файла. Дозапись и удаление невозможны(fig. 14).

* 
* Рис. 14: проверка доступа

1. Выполним команду, снимающую sticky-бит (fig. 15).

* 
* Рис. 15: удаление стики-бита

1. Выведем атрибуты (fig. 16).

* 
* Рис. 16: атрибуты

1. Теперь дозапись недоступна, удаление доступно(fig. 17).

* 
* Рис. 17: проверка доступа

# 4 Выводы

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

# Список литературы

1. Кетов Д.В. Внутреннее устройство Linux. BHV, 2017. 124 с.

2. Л. М. Ухлинов. Управление доступом в ОС GNU /Linux . ОКБ САПР», Москва, Россия, 2010.