Отчёт по лабораторной работе 5

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Гебриал Ибрам Есам Зекри НПИ-01-18

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.

# Теоретические сведения

Linux — многопользовательская система, поэтому она должна предоставлять систему разрешений, чтобы контролировать авторизованные операции с файлами и каталогами, к которым относятся все системные ресурсы и устройства (в Unix-системах любое устройство представляется в виде файла или каталога). Этот принцип является общим для всех Unix-систем, но напомнить об этом ещё раз будет не лишним, тем более что существуют некоторые интересные и сравнительно малоизвестные способы применения.

У каждого файла и каталога имеются специальные разрешения для трёх категорий пользователей:

* его владельца (обозначается u, от «user»);
* его группы-владельца (обозначается g, от «group»), представленная всеми членами группы;
* остальных (обозначается o, от «other»).

Три типа прав могут использоваться совместно:

* чтение (обозначается r, от «read»);
* запись (или изменение, обозначается w, от «write»);
* исполнение (обозначается x, от «eXecute»).

Два своеобразных права имеют смысл для исполняемых файлов: setuid и setgid (обозначаются буквой «s»). Обратите внимание, что они часто называются «битами», поскольку каждое из этих логических значений может быть представлено как 0 или 1. Эти два права позволяют любому пользователю выполнять программу на правах её владельца или группы соответственно. Данный механизм предоставляет доступ к функциям, требующим разрешений более высокого уровня, чем обычно есть у пользователя.

Поскольку setuid-программа, принадлежащая root, систематически запускается с правами суперпользователя, крайне важно убедиться в её безопасности и надёжности. Действительно, пользователь, которому удастся заставить её вызвать другую произвольную программу, сможет представиться как root и получить все права в системе. [1]

Чтобы применить соответствующие разрешения, первое, что нужно учитывать, это владение. Для этого есть команда chown.

Для управления правами используется команда chmod. При использовании chmod мы можем устанавливать разрешения для пользователя (user), группы (group) и других (other). [2]

# Выполнение лабораторной работы

1. Отключил систему SELinux. (рис. 1)

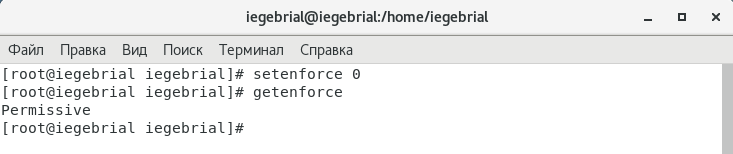


Figure 1: Отключение системы SELINUX

1. Вошёл в систему от имени пользователя guest и создал программу simpleid.c. (рис. 2)

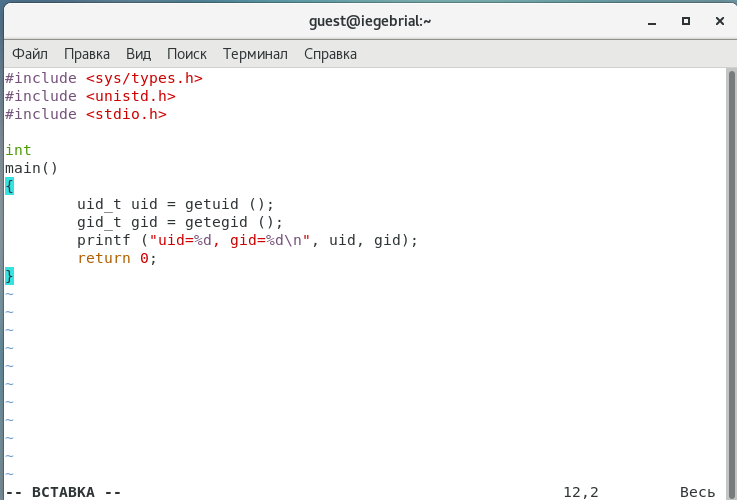


Figure 2: Создание программы simpleid.c

1. Скомплилировал программу. (рис. 3)

С помощью команды:

gcc simpleid.c -o simpleid

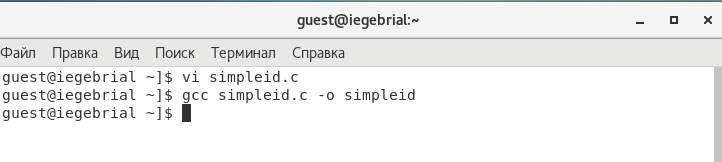


Figure 3: Компиляция программы

1. Выполнил программу simpleid и выполнил системную программу id. (рис. 4)

При сравнении результата id и simpleid получил одинаковые данны.

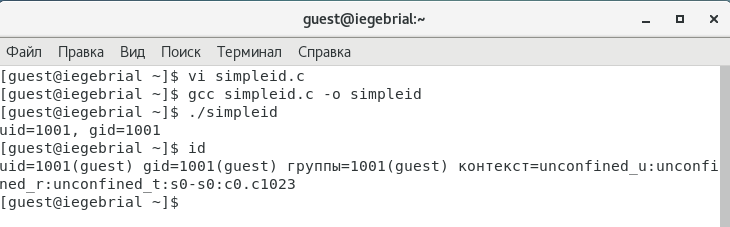


Figure 4: Выполнение программы

1. Усложнил программу, добавив вывод действительных идентификаторов. (рис. 5)



Figure 5: Создание программы simpleid2.c

1. Скомпилировал и запустил simpleid2.c. (рис. 6)

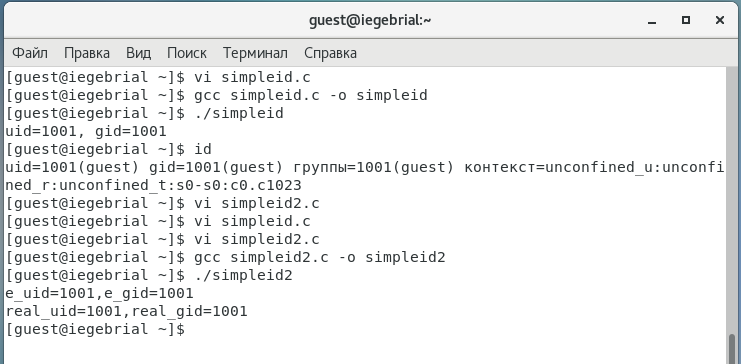


Figure 6: Компиляция и выполнение программы

1. От имени суперпользователя выполнил команды. (рис. 7)

chown root:guest /home/guest/simpleid2 - поменял пользователь на root и группа на guest.

chmod u+s /home/guest/simpleid2 - разрешение для владельца на выполнение с разрешением суперпользователя.

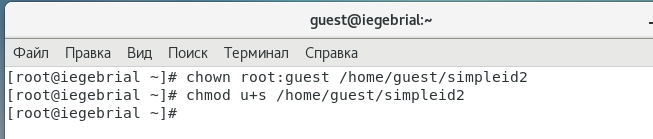


Figure 7: Выполнение команды

1. Выполнил проверку правильности установки новых атрибутов и смены владельца файла simpleid2. (рис. 8)

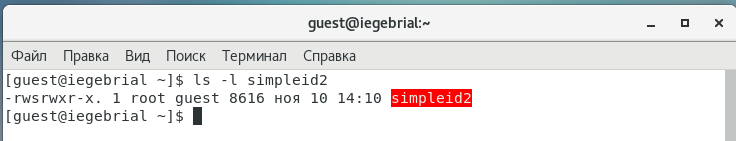


Figure 8: Проверка правильности установки новых атрибутов и смены владельца файла simpleid2

1. Запустил simpleid2 и id. (рис. 9)

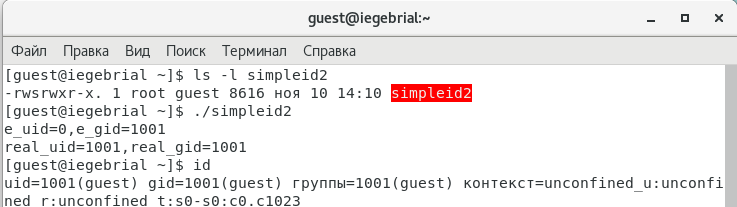


Figure 9: Выполнение программы

1. Проделал тоже самое относительно SetGID-бита. (рис. 10)

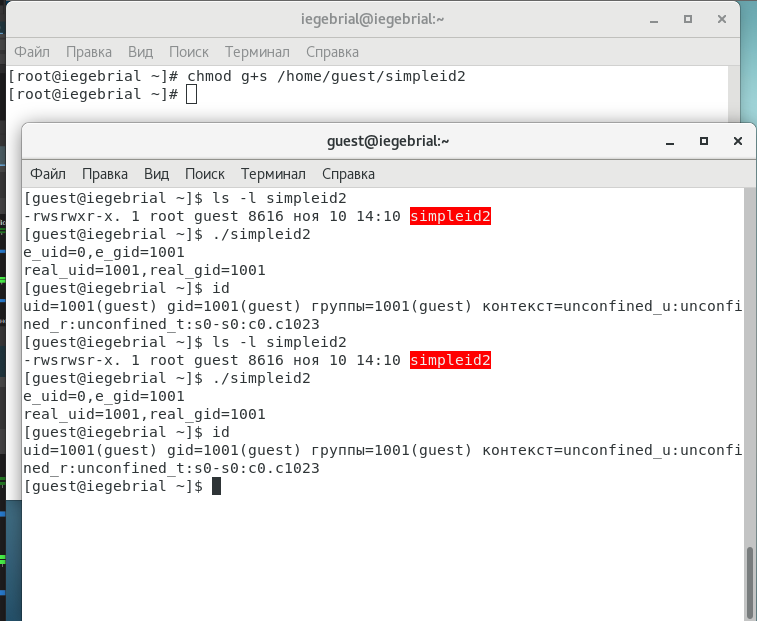


Figure 10: Выполнение программы

1. Создал программу readfile.c. (рис. 11)

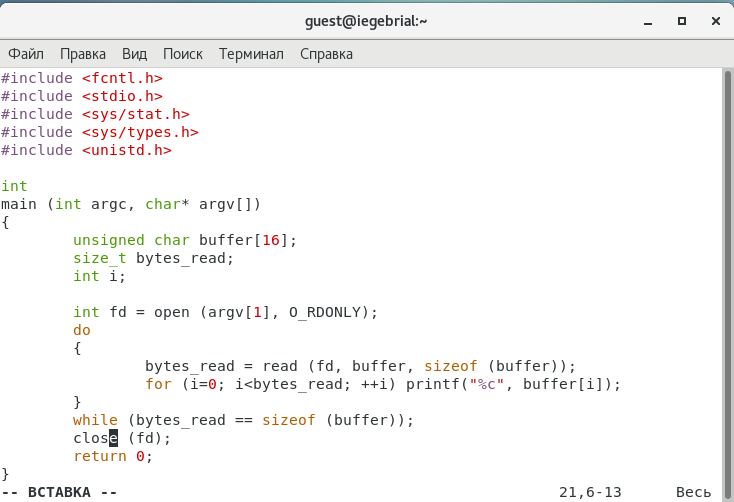


Figure 11: Создание программы readfile.c

1. Откомпилировал её. (рис. 12)

с помощью команды:

gcc readfile.c -o readfile

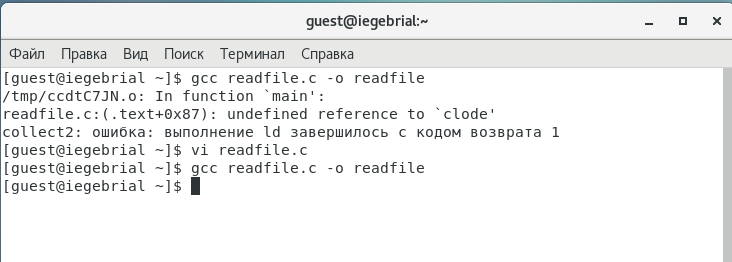


Figure 12: Компиляция программы

1. Сменил владельца у файла readfile.c (или любого другого текстового файла в системе) и изменил права так, чтобы только суперпользователь (root) мог прочитать его, a guest не мог.. (рис. 13)

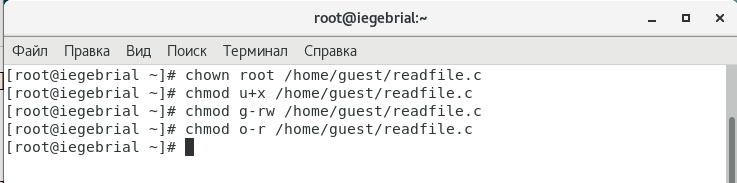


Figure 13: Изменение владельца прав у файла readfile.c

1. Проверил, что пользователь guest не может прочитать файл readfile.c. (рис. 14)

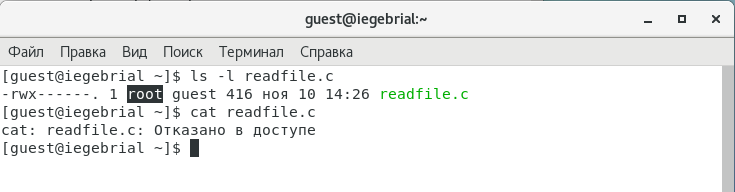


Figure 14: Проверка, что пользователь guest не может прочитать файл readfile.c

1. Сменил у программы readfile владельца и установил SetU’D-бит. (рис. 15)

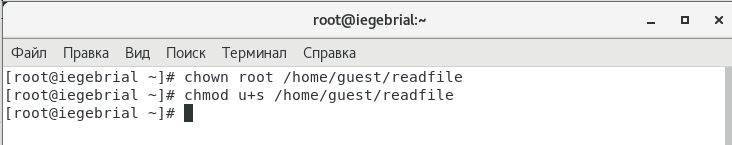


Figure 15: Изменение прав для программы readfile

1. Проверил, что программа readfile может прочитать файл readfile.c. (рис. 16)

Да получилось для программы readfile читать файл readfile.c.

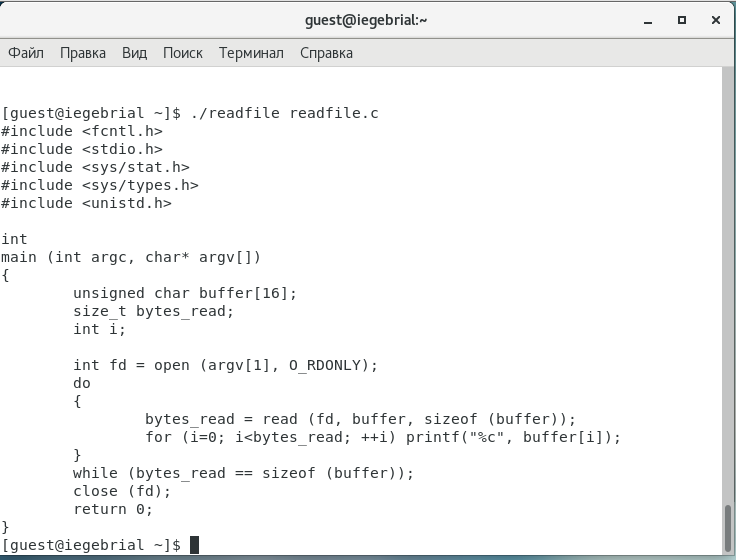


Figure 16: Проверка читения файла readfile.c с помощью readfile

1. Проверил, что программа readfile может прочитать файл /etc/shadow. (рис. 17)

Да получилось для программы readfile читать файл /etc/shadow.

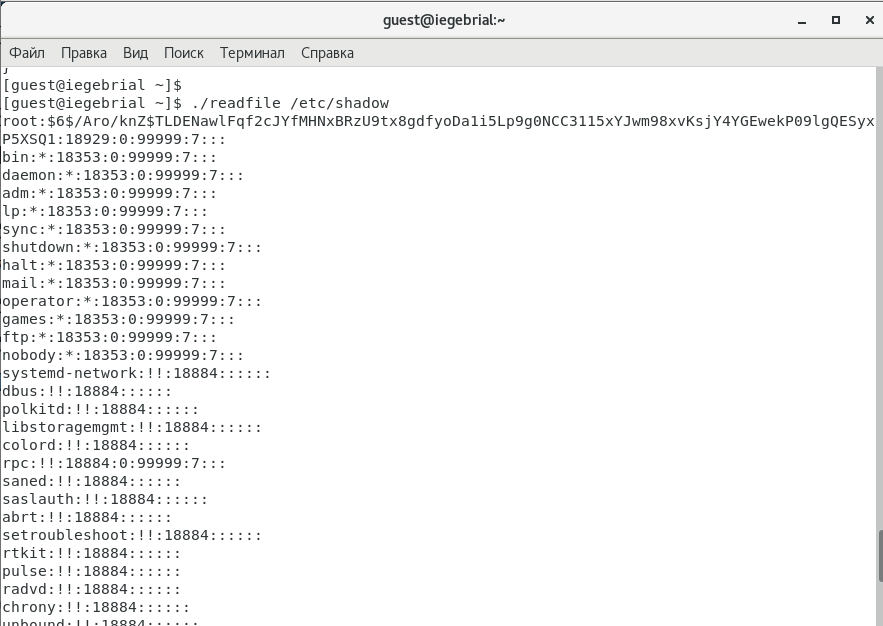


Figure 17: Проверка читения файла /etc/shadow с помощью readfile

1. Выяснил, установлен ли атрибут Sticky на директории /tmp (рис. 18)

С помощью команды

ls -l / | grep tmp

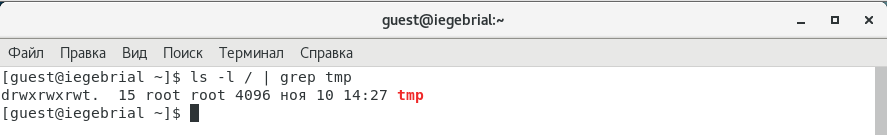


Figure 18: выяснение, установлен ли атрибут Sticky на директории /tmp

1. От имени пользователя guest создал файл file01.txt в директории /tmp со словом test. (рис. 19)



Figure 19: Создание файла file01.txt в директории /tmp со словом test

1. Просмотрел атрибуты у только что созданного файла и разрешил чтение и запись для категории пользователей «все остальные». (рис. 20)

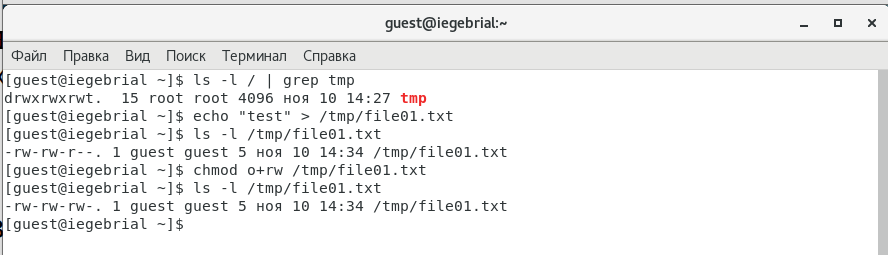


Figure 20: Просмотр атрибуты и разрешил чтение и запись для категории пользователей «все остальные».

1. Просмотрел атрибуты у только что созданного файла и разрешил чтение и запись для категории пользователей «все остальные». (рис. 21)

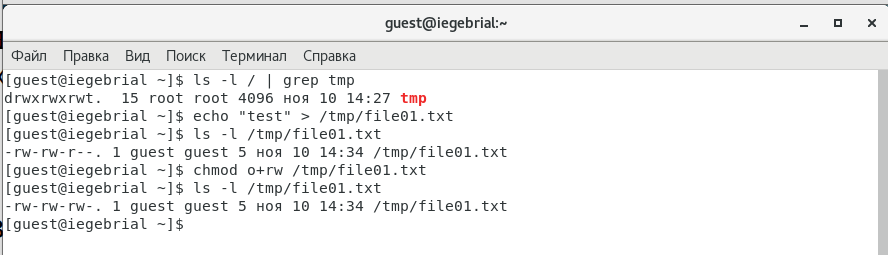


Figure 21: Просмотр атрибуты и разрешил чтение и запись для категории пользователей «все остальные».

1. От пользователя guest2 попробовал дозаписать в файл /tmp/file01.txt слово test2 командой и проверял.Попробовал записать в файл /tmp/file01.txt слово test3, стерев при этом всю имеющуюся в файле информацию командой. Прроверил содержимое файла командой и потом постарлся удалить файл но не получилось удалить файл (рис. 22)

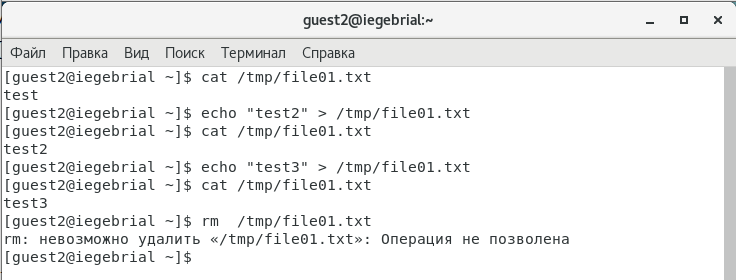


Figure 22: попытка дозаписи в файла, записи, удаления файла и проверка.

1. Повысил свои права до суперпользователя и выполнил после этого команду, снимающую атрибут t (Sticky-бит) с директории /tmp покинил режим суперпользователя командой. (рис. 23)

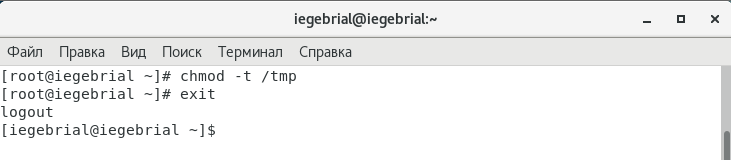


Figure 23: Снятие атрибута “t” с директории /tmp.

1. Повторил предыдущие шаги. Получилось удалить файл . (рис. 24)

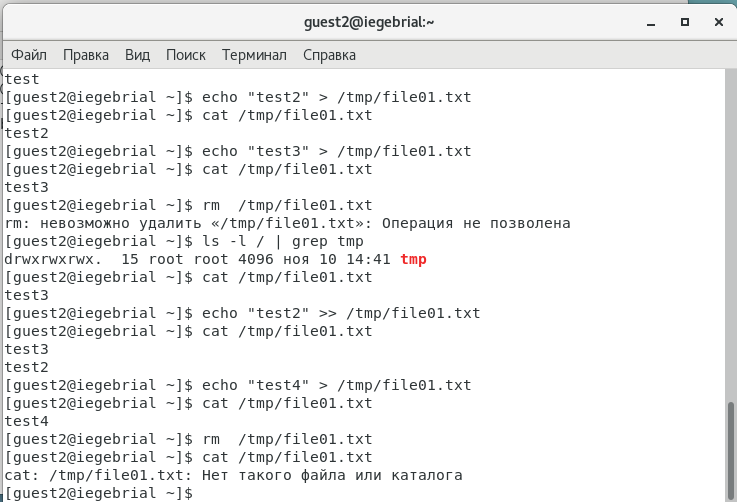


Figure 24: Повторение предыдущих шагов.

1. Повысил свои права до суперпользователя и вернил атрибут t на директорию /tmp . (рис. 25)

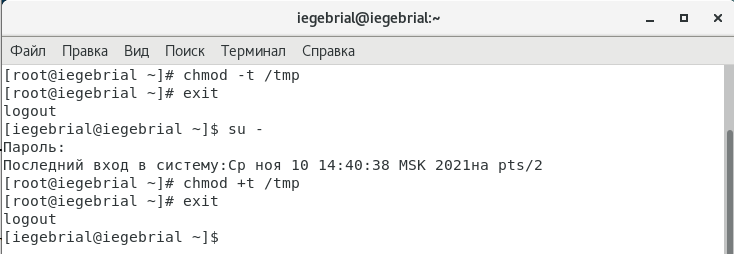


Figure 25: Добавление атрибута “t” на директорию /tmp

# Выводы

Изучал механизм 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.

# Список литературы

1. Управление правами. https://debian-handbook.info/browse/ru-RU/stable/sect.rights-management.html.
2. Права в Linux. //Losst.2020. URL:https://losst.ru/neizmenyaemye-fajly-v-linux (дата обращения 1.10.2019).