Лабораторная работа № 5

Дискреционное разграничение прав в Linux. Исследование влияния дополнительных атрибутов

Беличева Дарья Михайловна

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 области разрешений Linux, SUID (установка идентификатора пользователя), SGID (установка идентификатора группы) и Sticky Bit являются критическими атрибутами, которые выходят за рамки стандартных разрешений для файлов, обеспечивая мощные функциональные возможности, такие как предоставление временных привилегий, контроль группового доступа и сохранение целостности данных [1].

SUID, сокращение от Set User ID, представляет собой специальное разрешение, которое может быть назначено исполняемым файлам. Когда у исполняемого файла включено разрешение SUID, это позволяет пользователям, выполняющим файл, временно принимать привилегии владельца файла. Это означает, что даже если у пользователя нет необходимых разрешений для доступа или выполнения определенных действий, он может сделать это, запустив файл с разрешением SUID.

SGID, что означает Set Group ID, — это ещё одно специальное разрешение, которое можно применить к исполняемым файлам и каталогам. Если для исполняемого файла включено разрешение SGID, это позволяет пользователям, которые запускают файл, временно стать владельцами группы, к которой относится файл. Для каталогов с включённым разрешением SGID вновь созданные файлы и каталоги в этом каталоге наследуют принадлежность к группе родительского каталога, а не принадлежность к группе пользователя по умолчанию.

Sticky Bit – это специальное разрешение, которое можно установить только для каталогов. Когда залипание включено для каталога, оно ограничивает возможность удаления или переименования файлов в этом каталоге для владельца файла, владельца каталога и суперпользователя. Это гарантирует, что каждый пользователь может удалять или изменять только свои файлы, даже если у него есть права на запись в каталог.

# 3 Выполнение лабораторной работы

Установим компилятор gcc, а также отключим SELinux (рис. 1).

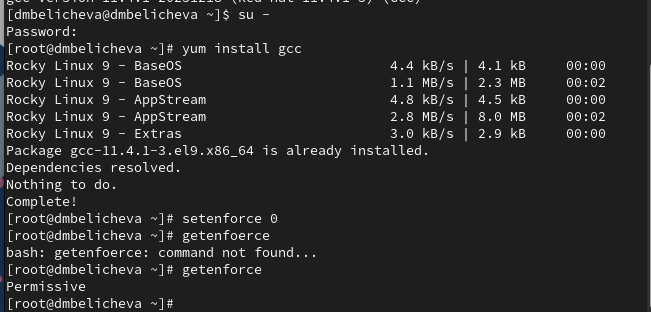


Рис. 1: Подготовка лабораторного стенда

Войдем в систему от имени пользователя guest и создадим программу simpleid.c (рис. 2).

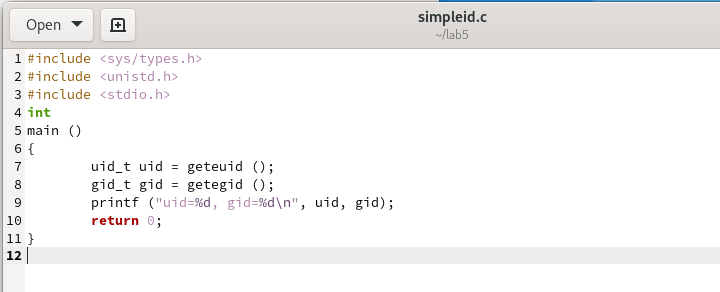


Рис. 2: Содержимое файла simpleid.c

Скомпилируем программу и убедимся, что файл программы создан. Далее выполним программу simpleid, нам выведутся uid и gid. Выполним системную программу id и сравним полученный результат с данными предыдущего пункта задания (рис. 3). Увидим, что информация идентична.

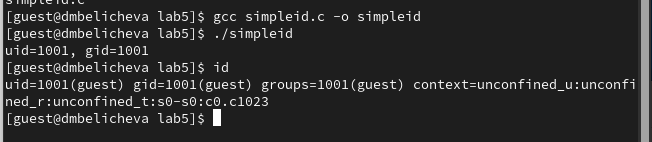


Рис. 3: Запуск программы simpleid

Усложните программу, добавив вывод действительных идентификаторов (рис. 4).

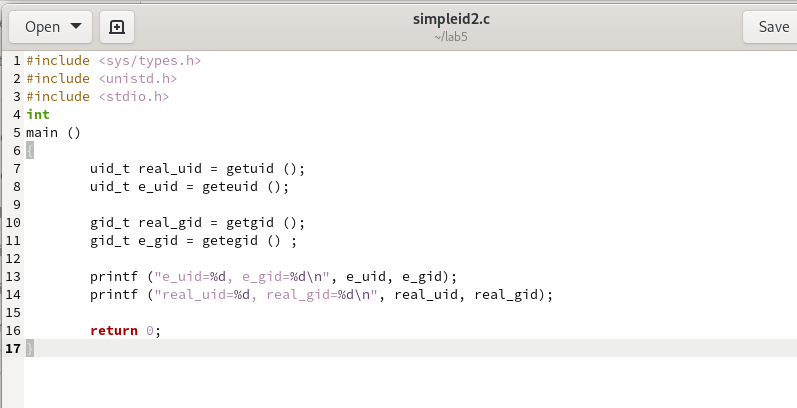


Рис. 4: Содержимое файла simpleid2.c

Скомпилируем программу и убедимся, что файл программы создан. Далее выполним программу simpleid, нам выведутся uid и gid, а также их действительные идентификаторы (рис. 5).



Рис. 5: Запуск программы simpleid2

От имени суперпользователя изменим владельца файла /home/guest/simpleid2 и установим SetUID-бит. Проверим корректность установленных прав и опять запустим simpleid2 (рис. 6).

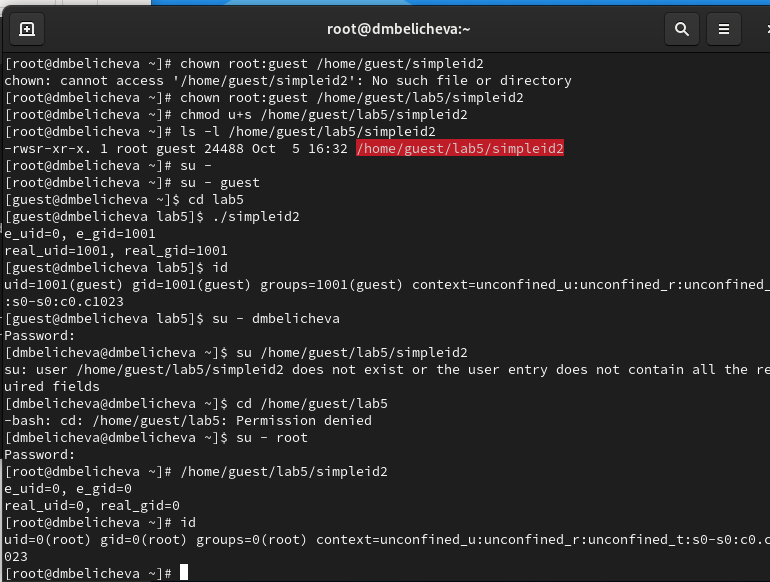


Рис. 6: Изменение владельца и запуск программы simpleid2 с установленным SetUID-битом

Проделаем тоже самое относительно SetGID-бита (рис. 7).

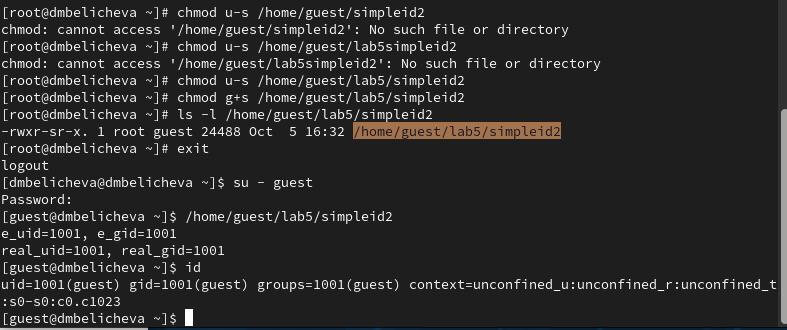


Рис. 7: Запуск программы simpleid2 с установленным SetGID-битом

Создадим программу readfile.c (рис. 8).

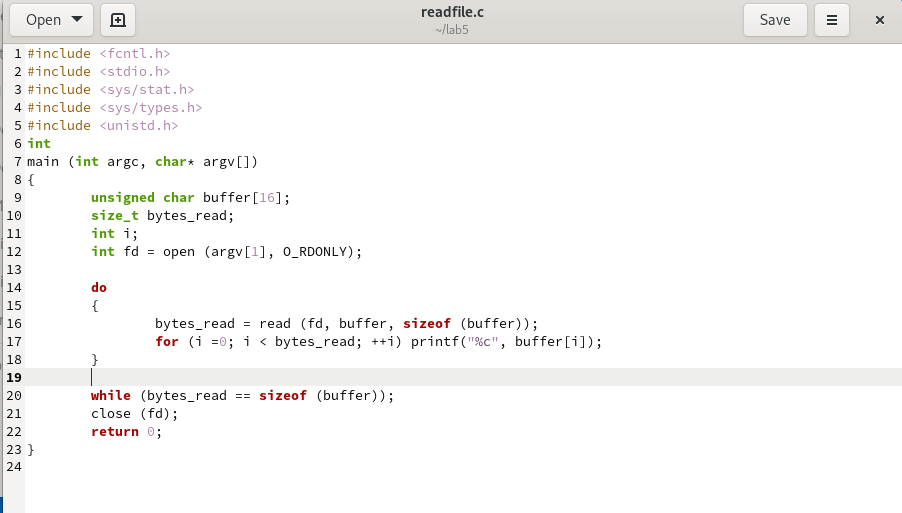


Рис. 8: Содержимое файла readfile.c

Откомпилируем её. Сменим владельца у файла readfile.c на root и изменим права так, чтобы только суперпользователь (root) мог прочитать его, a guest не мог. Для этого изменим права файла на 700, то есть только пользователю (root) разрешены все действия. Проверим, что пользователь guest не может прочитать файл readfile.c (рис. 9).

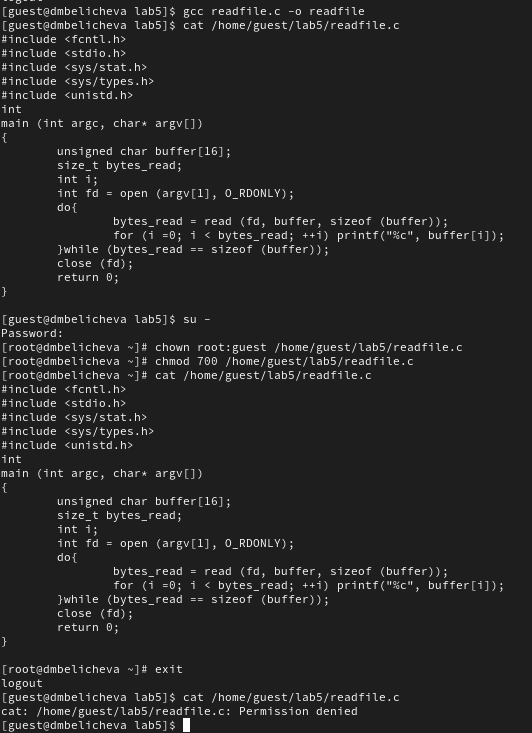


Рис. 9: Изменение владельца и прав файла readfile.c

Сменим у программы readfile владельца и установим SetUID-бит. Убедимся, что программа readfile может прочитать файл readfile.c. Также проверим, что программа readfile может прочитать файл /etc/shadow (рис. 10).

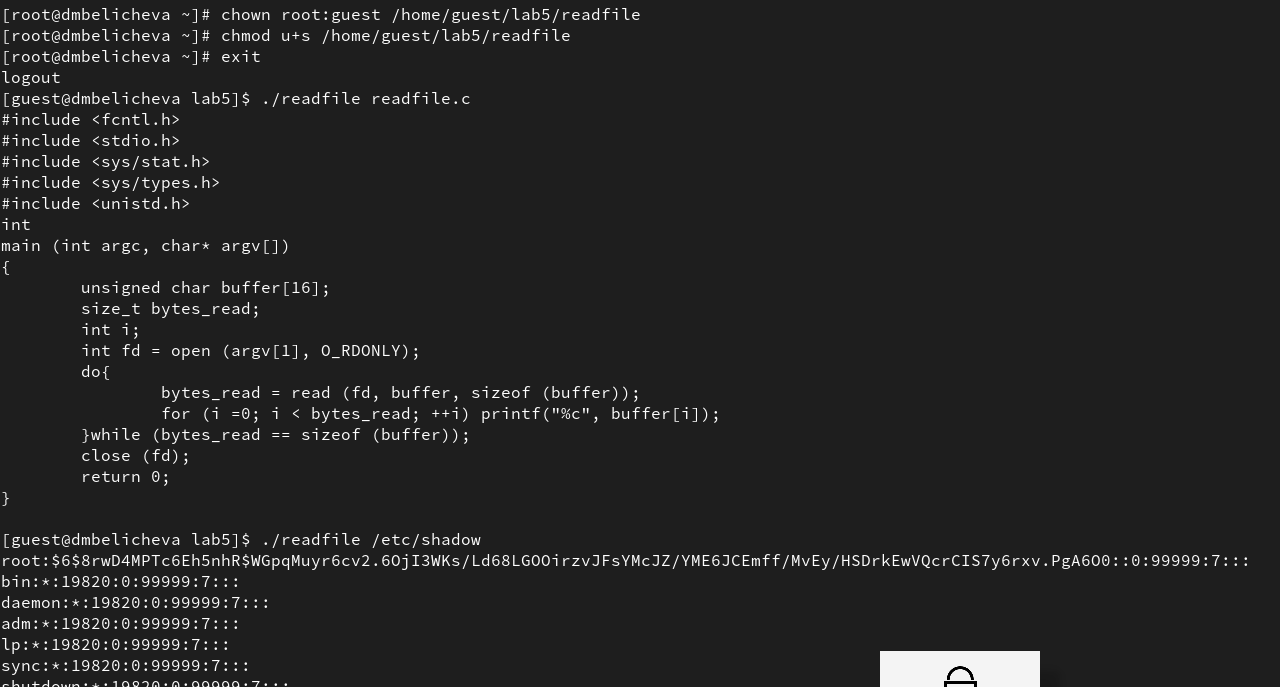


Рис. 10: Установка SetUID-бита на исполняемый файл readfile и проверка прав

Проверим, что установлен атрибут Sticky на директории /tmp (в конце стоит t). Затем от имени пользователя guest создадим файл file01.txt в директории /tmp со словом test, затем просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные». После этого от пользователя guest2 попробуем дозаписать в этот файл новое слово, однако получим отказ, также нам отказано в перезаписи и удалении этого файла. Если же убрать Sticky бит, то нам будет разрешено удаление этого файла (рис. 11).

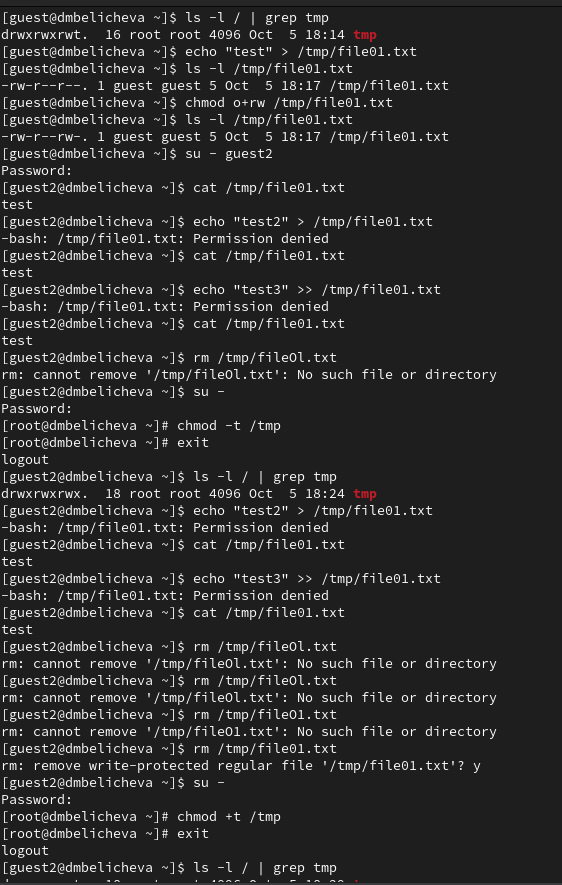


Рис. 11: Исследование Sticky-бита

# 4 Выводы

В процессе выполнения данной лабораторной работы я изучила механизмы изменения идентификаторов, применения SetUID- и Sticky-битов. Получила практические навыки работы в консоли с дополнительными атрибутами. Рассмотрела работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

# Список литературы

1. What is SUID, SGID, and Sticky Bit? [Электронный ресурс]. 2024. URL: <https://www.scaler.com/topics/special-permissions-in-linux/>.