Основы информационной безопасности

Лабораторная работа № 4. Дискреционное разграничение прав в Linux. Два пользователя

Подлесный Иван Сергеевич

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.

# Выполнение лабораторной работы

Проверим установлен ли компилятор gcc(обновим его), а также отключим SELinux(рис. fig. 1)

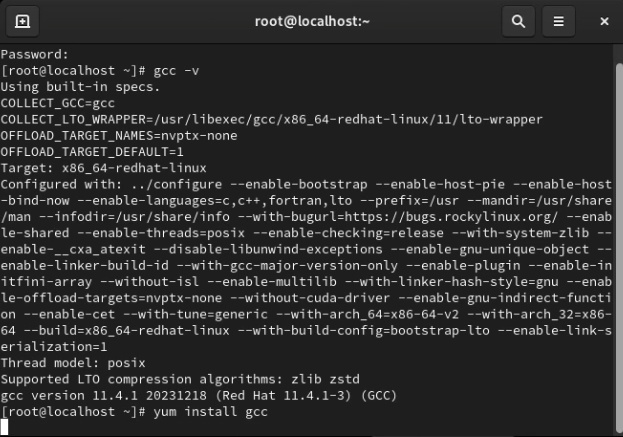


Рис. 1: Подготовка лабораторного стенда

Войдем в систему от имени пользователя guest и создадим программу simpleid.c, которая выводит идентификатор пользователя и группы(рис. fig. 2)

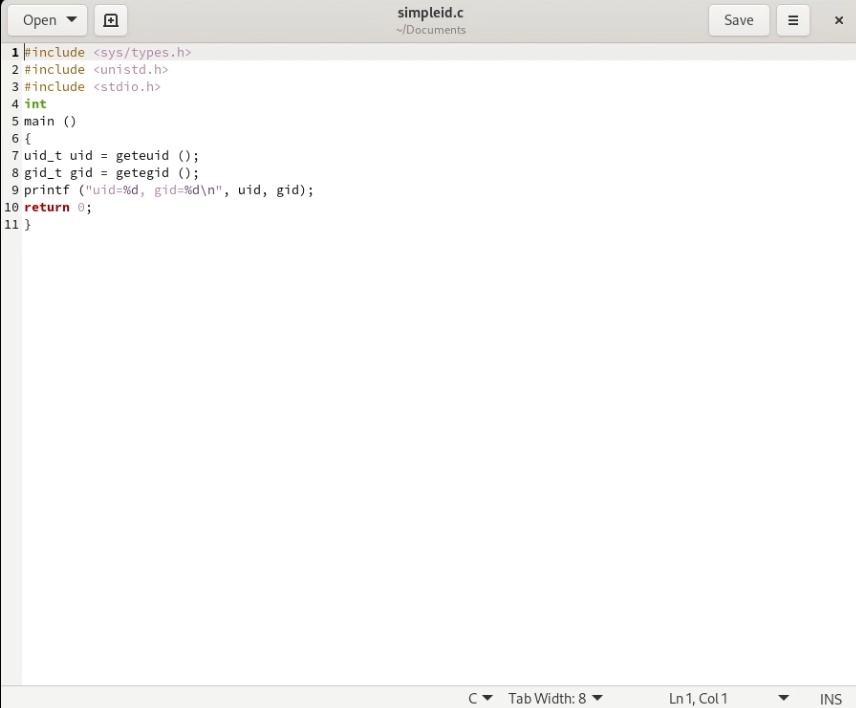


Рис. 2: Текст программы simpleid.c

Запустив её, увидим, что она выводит идентификаторы пользователя и группы 1001 и 1001 для guest, что совпадает с выводом команды id(рис. fig. 3)

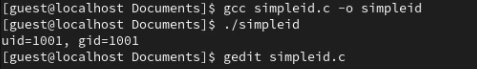


Рис. 3: Запуск программы simpleid

Изменим программу, добавив вывод действительных идентификаторов(рис. fig. 4).

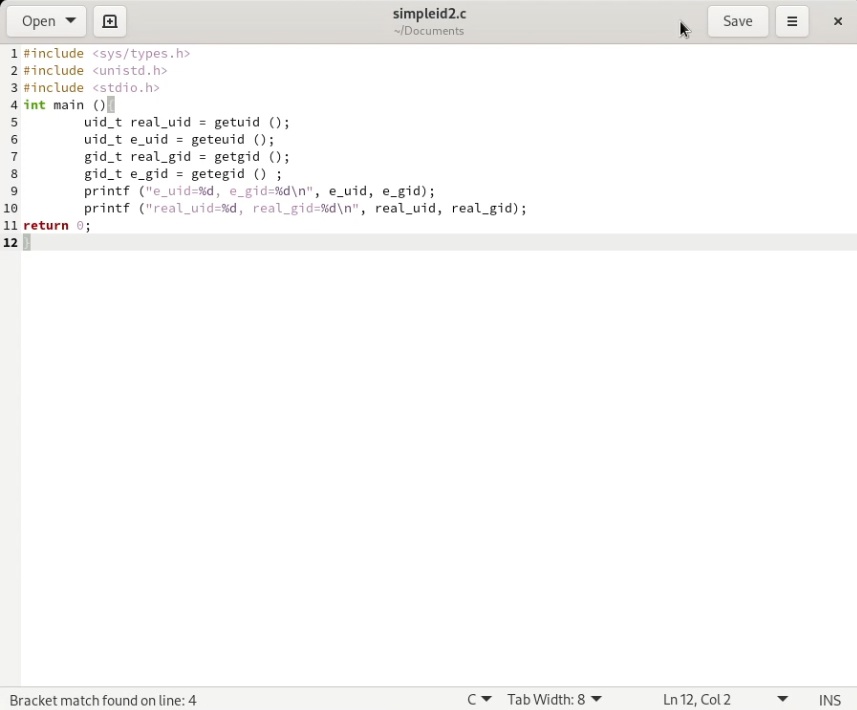


Рис. 4: Текст программы simpleid2.c

Компилириуем программу с помощью gcc, затем, запустив её, увидим, что она выводит идентификаторы пользователя и группы 1001 и 1001 для guest, что совпадает с выводом команды id(рис. fig. 5).

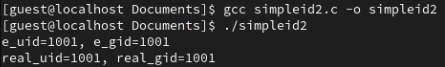


Рис. 5: Запуск программы simpleid2

От имени суперпользователя изменим владельца файла homeguestsimpleid2 и установим SetUID-бит. Проверим корректность установленных прав и опять запустим simpleid2(рис. fig. 6).

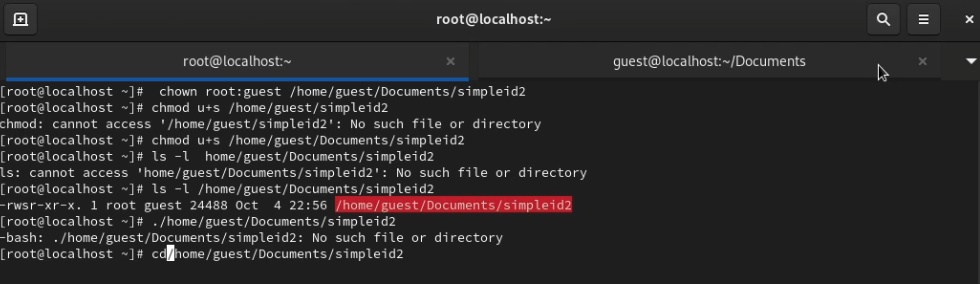
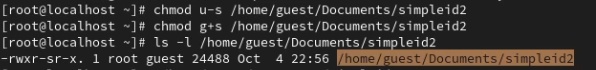


Рис. 6: Изменение владельца и запуск программы simpleid2 с установленным SetUID-битом



Рис. 7: Изменение владельца и запуск программы simpleid2 с установленным SetUID-битом

Проделаем аналогичные действия относительно SetGID-бита(рис. fig. **¿fig:007?**):

 Запуск программы simpleid2 с установленным SetGID-битом

Создадим программу для чтения файлов readfile.c(рис. fig. 8):

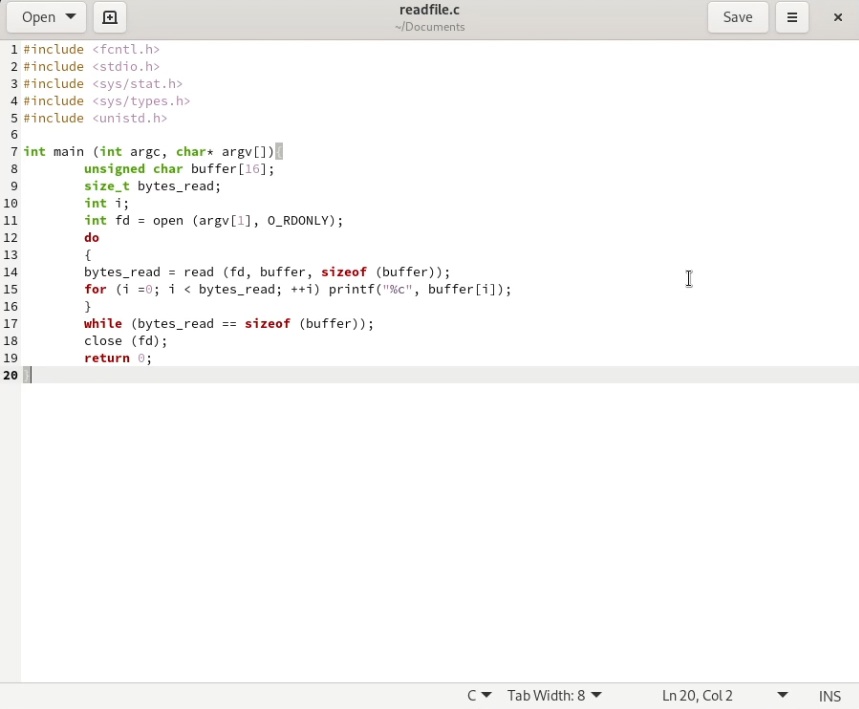


Рис. 8: Текст программы readfile.c

Текст программы readfile.c

Рис. 9: Текст программы readfile.c

Скомпилируем её и сменим владельца у файла с текстом программы, затем изменим права так, чтобы только суперпользователь (root) мог прочитать его, и проверим корректность настроек(рис. fig. 10):

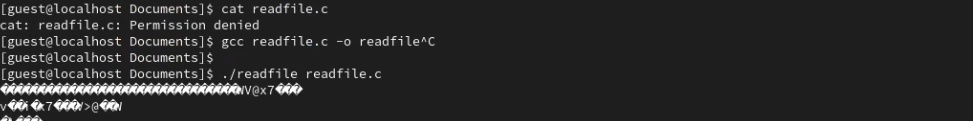


Рис. 10: Изменение владельца и прав файла readfile.c

Сменим у программы readfile владельца и установим SetUID-бит. Теперь эта программа может прочитать файл readfile.c , также она может прочитать файл etcshadow, владельцем которого guest также не является, так как программа readfile теперь имеет все права пользователя root(рис. fig. 11):



Рис. 11: 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. fig. 12):

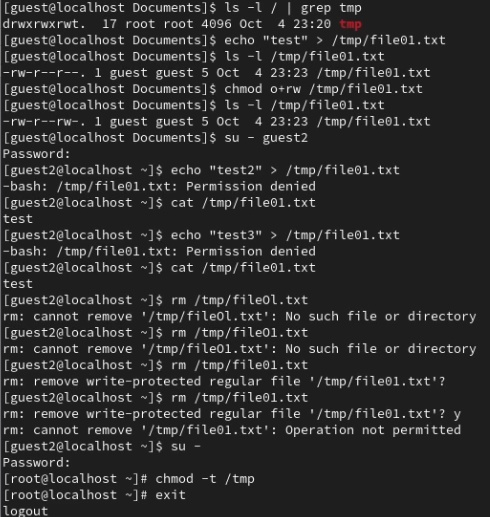


Рис. 12: Работа с атрибутом Sticky

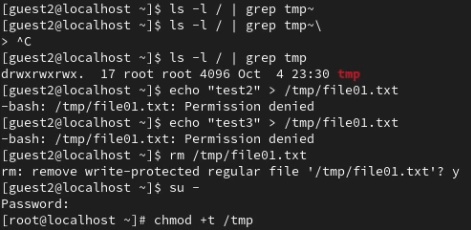


Рис. 13: Работа с атрибутом Sticky

# Выводы

В результате выполнения работы были визучены механизмы изменения идентификаторов, применения SetUID- и Sticky-битов.

Были получены практических навыков работы в консоли с дополнительными атрибутами.

Были рассмотрены работы механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.