Лабораторная работа №5

Дисциплина: Основы информационной безопасности

Георгес Гедеон

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.

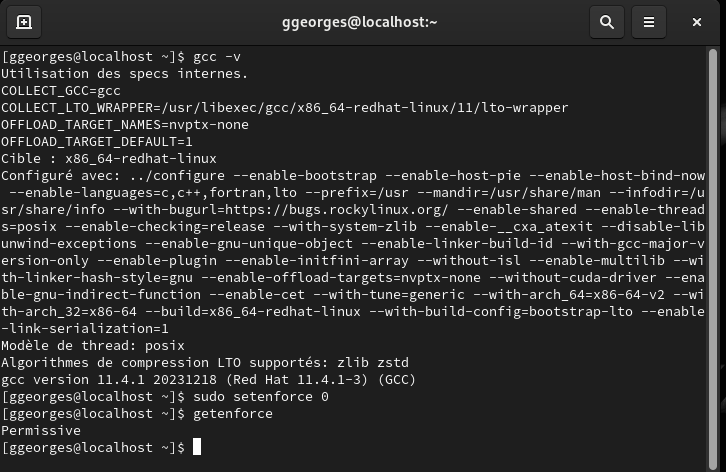
# Теоретическое введение

SetUID, SetGID и Sticky - это специальные типы разрешений позволяют задавать расширенные права доступа на файлы или каталоги. • SetUID (set user ID upon execution — «установка ID пользователя во время выполнения) являются флагами прав доступа в Unix, которые разрешают пользователям запускать исполняемые файлы с правами владельца исполняемого файла. • SetGID (set group ID upon execution — «установка ID группы во время выполнения») являются флагами прав доступа в Unix, которые разрешают пользователям запускать исполняемые файлы с правами группы исполняемого файла. • Sticky bit в основном используется в общих каталогах, таких как /var или /tmp, поскольку пользователи могут создавать файлы, читать и выполнять их, принадлежащие другим пользователям, но не могут удалять файлы, принадлежащие другим пользователям. Более подробно см. в [1].

# Выполнение лабораторной работы

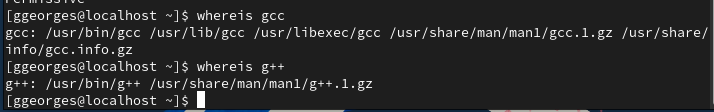
1 часть: Создание программы

1)Для начала мы убеждаемся, что компилятор gcc установлен, исолпьзуя команду “gcc -v”. Затем отключаем систему запретов до очередной перезагрузки системы командой “sudo setenforce 0”, после чего команда “getenforce” выводит “Permissive”(Рисунок 3.1).



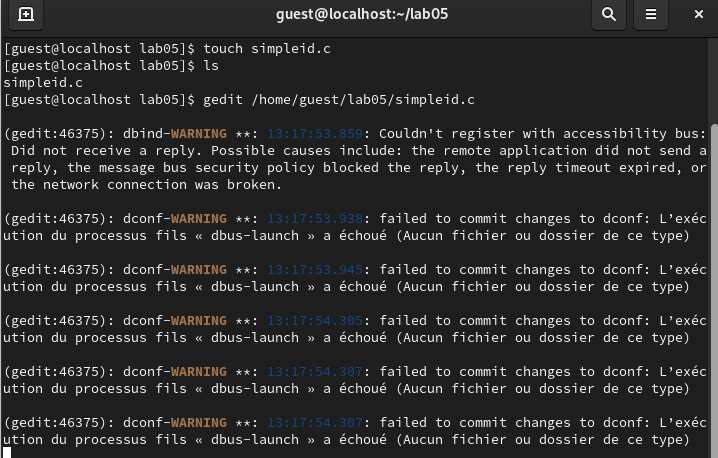
Предварительная подготовка

2)Проверяем успешное выполнение команд “whereis gcc” и “whereis g++”(их расположение)(Рисунок 3.2).



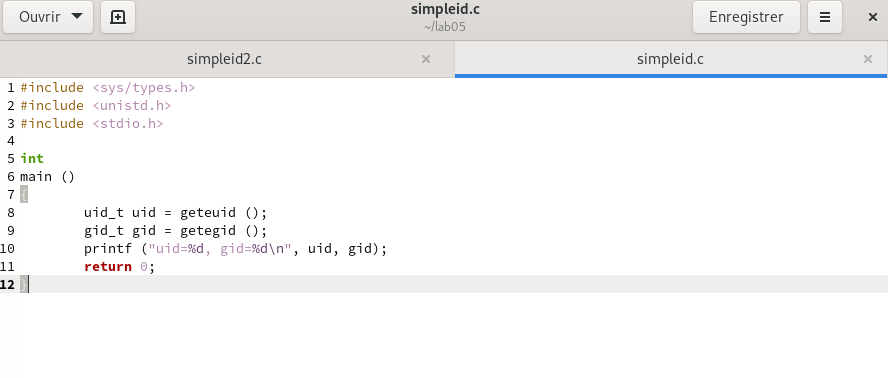
Команда “whereis”

3)Входим в систему от имени пользователя guest командой “su - guest”. Создаём программу simpleid.c командой “touch simpleid.c” и открываем её в редакторе командой “gedit /home/guest/lab05/simpleid.c” (Рисунок 3.3).



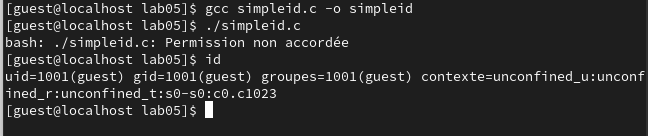
Вход в систему и создание программы

4)Код программы выглядит следующим образом (Рисунок 3.4).



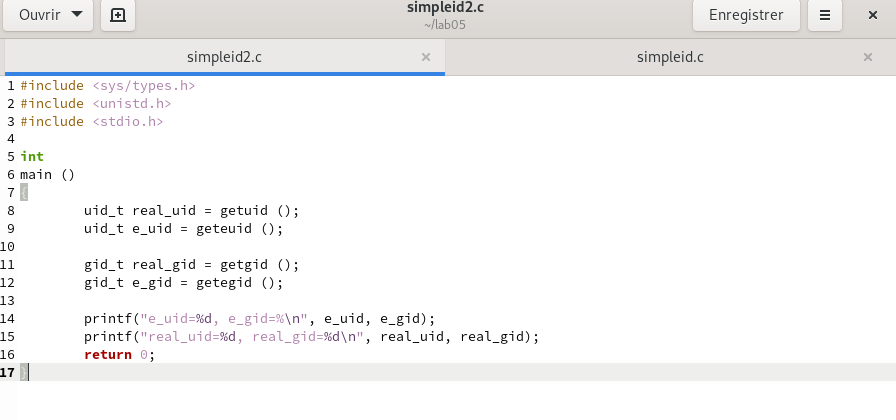
Код программы simpleid.c

5)Скомпилируем программу и убедимся, что файл программы был создан командой “gcc simpleid.c -o simpleid”. Выполняем программу simpleid командой “./simpleid”, а затем системную программу id командой “id”. Результаты, полученные в результате выполнения обеих команд, совпадают(uid=1001 и gid=1001) (Рисунок 3.5).



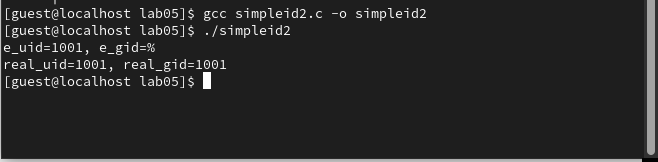
Компиляция и выполнение программы simpleid

6)Усложняем программу, добавив вывод действительных идентификаторов, новый файл назовём simpleid.c(Рисунок 3.6).



Усложнение программы

7)Скомпилируем и запустим simpleid2.c командами “gcc simpleid2.c -o simpleid2” и “./simpleid2” (Рисунок 3.7).



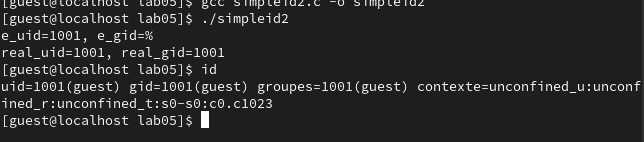
Компиляция и выполнение программы simpleid2

8)От имени суперпользователя выполняем команды “sudo chown root:guest /home/guest/lab05/simpleid2” и “sudo chmod u+s /home/guest/lab05/simpleid2”, затем выполняем проверку правильности установки новых атрибутов и смены владельца файла simpleid2 командой “sudo ls -l /home/guest/lab05/simpleid2”(Рисунок 3.8). Этими командами была произведена смена пользователя файла на root и установлен SetUID-бит.



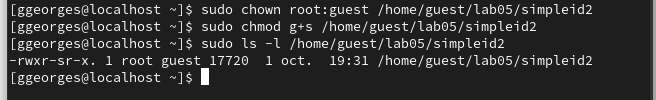
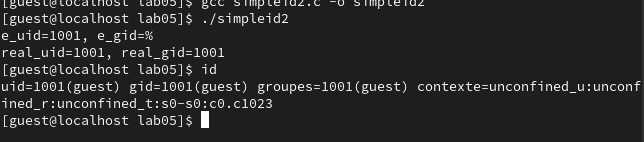
Установка новых атрибутов (SetUID) и смена владельца файла

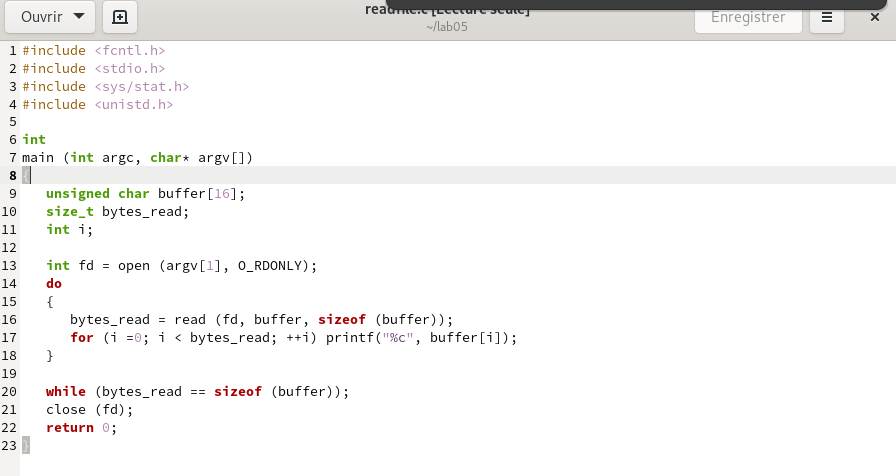
9)Запускаем программы simpleid2 и id. Теперь появились различия в uid (Рисунок 3.9).



Запуск simpleid2 после установки SetUID

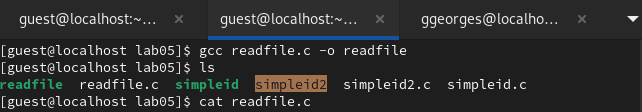
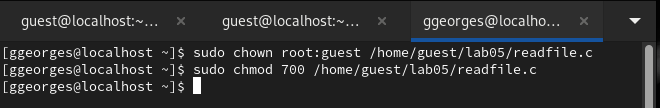
10)Проделаем тоже самое относительно SetGID-бита. Также можем заметить различия с предыдущим пунктом (Рисунок 3.10).

  11)Создаем программу readfile.c(Рисунок 3.11).

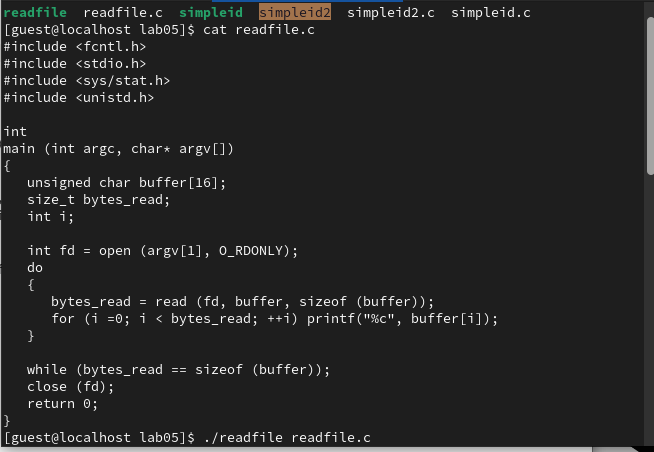


Код программы readfile.c

12)Скомпилируем созданную программу командой “gcc readfile.c -o readfile”. Сменим владельца у файла readfile.c командой “sudo chown root:guest /home/guest/readfile.c” и поменяем права так, чтобы только суперпользователь мог прочитать его, а guest не мог, с помощью команды “sudo chmod 700 /home/guest/readfile.c”. Убеждаемся, что пользователь guest не может прочитать файл readfile.c командой “cat readfile.c”, получив отказ в доступе (Рисунок 3.12).

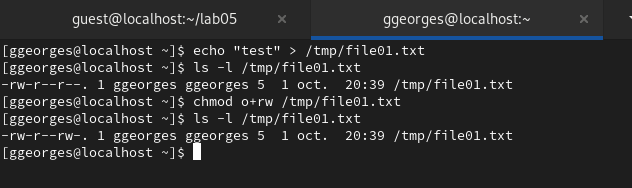
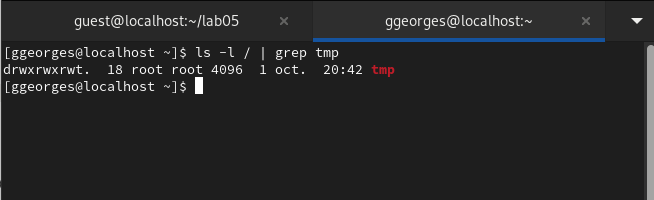
  { width=70% }

13)Поменяем владельца у программы readfile и установим SetUID. Проверим, может ли программа readfile прочитать файл readfile.c командой “./readfile readfile.c”. Прочитать удалось. Аналогично проверяем, можно ли прочитать файл /etc/shadow. Прочитать удалось (Рисунок 3.13).

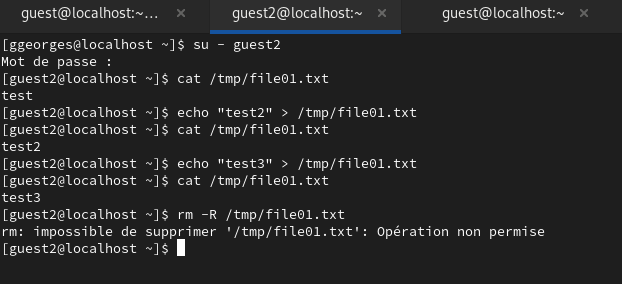
  { width=70% }

2 часть: Исследование Sticky-бита

1)Командой “ls -l / | grep tmp” убеждаемся, что атрибут Sticky на директории /tmp установлен. От имени пользователя guest создаём файл file01.txt в директории /tmp со словом test командой “echo”test” > /tmp/file01.txt”. Просматриваем атрибуты у только что созданного файла и разрешаем чтение и запись для категории пользователей “все остальные” командами “ls -l /tmp/file01.txt” и “chmod o+rw /tmp/file01.txt” (Рисунок 3.14).

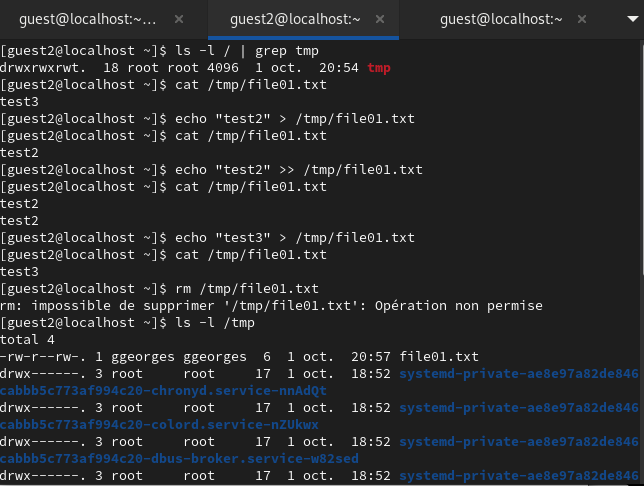
  { width=70% }

2)От имени пользователя guest2 пробуем прочитать файл командой “cat /tmp/file01.txt” - это удалось. Далее пытаемся дозаписать в файл слово test2, проверить содержимое файла и записать в файл слово test3, стерев при этом всю имеющуюся в файле информацию - эти операции удалось выполнить только в случае, если еще дополнительно разрешить чтение и запись для группы пользователей командой “chmod g+rw /tmp/file01.txt”. От имени пользователя guest2 пробуем удалить файл - это не удается ни в каком из случаев, возникает ошибка (Рисунок 3.15).



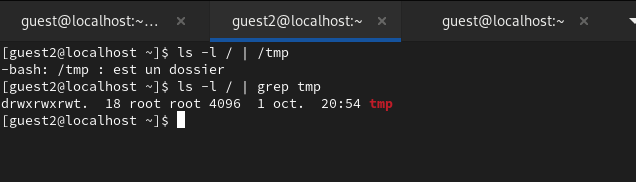
Попытка выполнить действия над файлом file01.txt от имени пользователя guest2

3)Повышаем права до суперпользователя командой “su -” и выполняем команду, снимающую атрибут t с директории /tmp “chmod -t /tmp”. После чего покидаем режим суперпользователя командой “exit”. Повторяем предыдущие шаги. Теперь нам удаётся удалить файл file01.txt от имени пользователя, не являющегося его владельцем (Рисунок 3.16).



Удаление атрибута t (Sticky-бита) и повторение действий

4)Повышаем свои права до суперпользователя и возвращаем атрибут t на директорию /tmp (Рисунок 3.17).



Возвращение атрибута t (Sticky-бита)

# Выводы

* В ходе выполнения данной лабораторной работы я изучил механизмы изменения идентификаторов, применение 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.