Лабораторная работа №5

Информационная безопасность

Николаев Дмитрий Иванович

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

## 2.1 SetUID

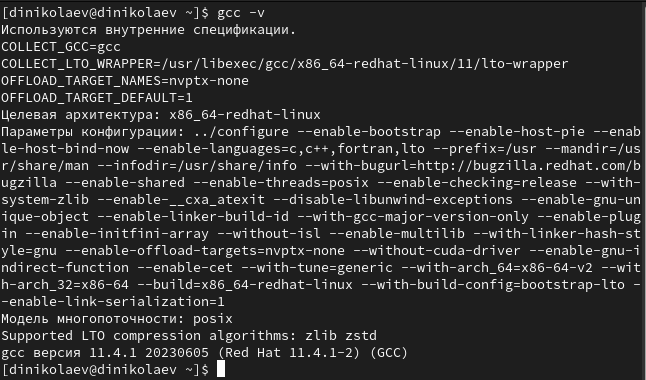
SetUID (сокращения от англ. set user ID upon execution — “установка ID пользователя во время выполнения”) разрешает пользователям запускать исполняемые файлы с правами владельца исполняемого файла. Иногда файлы требуют разрешения на выполнение для пользователей, которые не являются членами группы владельца, в этом случае потребуется предоставить специальные разрешения на выполнение. Когда SetUID установлен, пользователь может запускать любую программу, как её владелец.

## 2.2 Sticky-бит

В случае, если этот бит установлен для папки, то файлы в этой папке могут быть удалены только их владельцем. Пример использования этого бита в операционной системе это системная папка /tmp . Эта папка разрешена на запись любому пользователю, но удалять файлы в ней могут только пользователи, являющиеся владельцами этих файлов. Символ «t» указывает, что на папку установлен Sticky-бит.

# 3 Выполнение лабораторной работы

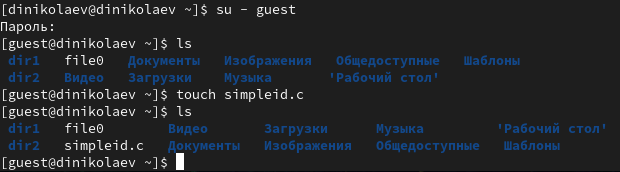
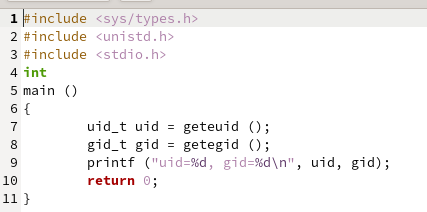
С помощью команды gcc -v убедимся, что у меня установлен компилятор gcc ([??]).



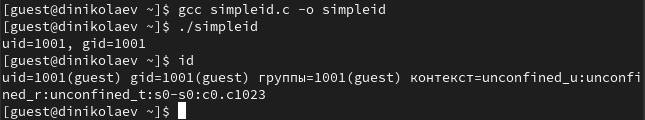
Проверка установленности gcc

## 3.1 Создание программы

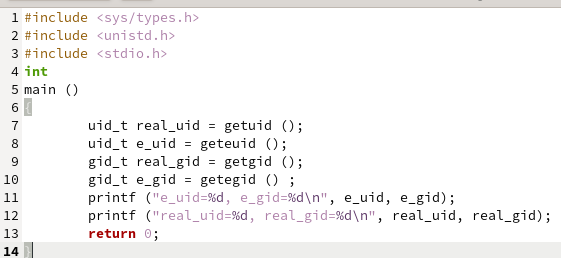
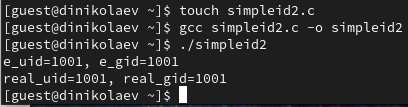
1. Действуя согласно [1], зайдём от имени пользователя guest и создадим файл simpleid.c ([??]–[??]).

* 
* Создание файла simpleid.c от имени guest
* 
* Код файла simpleid.c

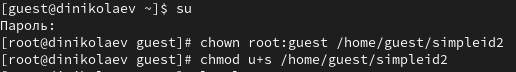
1. Скомпилируем программу и убедимся, что исполняемый файл программы создан: gcc simpleid.c -o simpleid. Выполним программу simpleid: ./simpleid и сравним с выводом системной программы id. Как можно заметить, выводы идентичны ([??]).

* 
* Компиляция и запуск программы simpleid

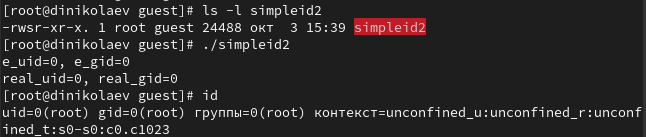
1. Усложним программу, добавив вывод действительных идентификаторов. Получившуюся программу назовём simpleid2.c ([??]). После скомпилируем и запустим получившийся исполняемый файл ([??]).

* 
* Код программы simpleid2.c
* 
* Компиляция и исполнение файла simpleid2

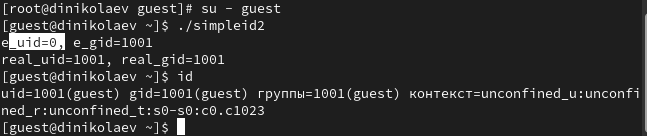
1. От имени суперпользователя выполним команды ([??]):
   * chown root:guest /home/guest/simpleid2
   * chmod u+s /home/guest/simpleid2

* 
* Изменение владельца и прав доступа к файлу simpleid2
* Команда chown root:guest /home/guest/simpleid2 меняет владельца файла (с guest на root). Команда chmod u+s /home/guest/simpleid2 меняет права доступа к файлу (добавляет атрибут s (вместо x у владельца файла)).

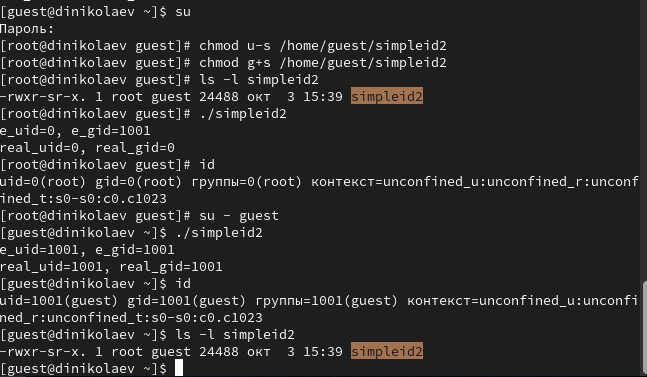
1. Выполним проверку правильности установки новых атрибутов и смены владельца файла simpleid2: ls -l simpleid2 и запустим simpleid2 и id. Для пользователя root все выводы равняются 0 ([??]).

* 
* Проверка атрибутов файла simpleid2 и вывод для root пользователя

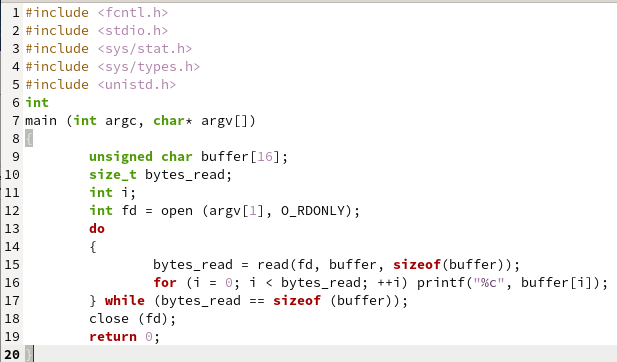
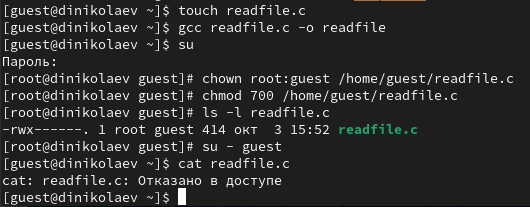
1. Проверим вывод для пользователя guest ([??]).

* 
* Вывод simpleid2 для пользователя guest

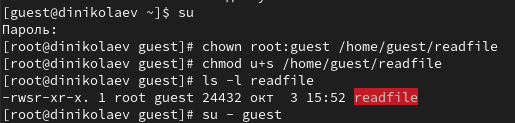
1. Проделаем тоже самое относительно SetGID-бита, предварительно сняв UID-бит ([??]).

* 
* Установка SetGID-бита и проверка вывода файла simpleid2

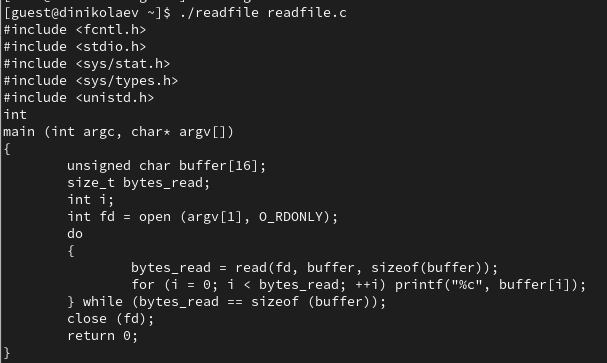
1. Создадим программу readfile.c ([??]) и скомпилируем её: gcc readfile.c -o readfile. Сменим владельца у файла readfile.c и изменим права так, чтобы только суперпользователь (root) мог прочитать его, a guest не мог, совершив проверку выполненного ([??]).

* 
* Код программы readfile.c
* 
* Смена владельца и прав на файл readfile.c с проверкой

1. Сменим у программы readfile владельца и установим SetUID-бит ([??]).

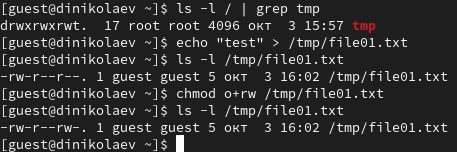
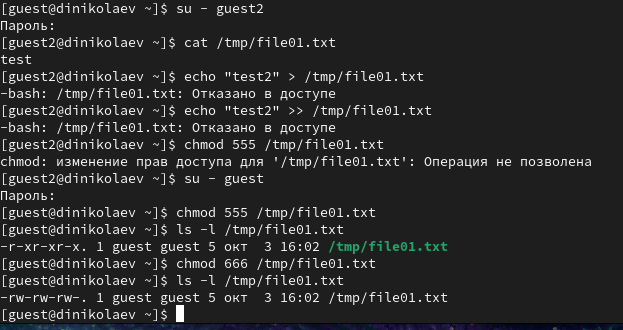
* 
* Смена владельца и установление SetUID-бита на файл readfile

1. Проверим, может ли программа readfile прочитать файл readfile.c ([??]) и /etc/shadow ([??]). Так как мы установили SetUID-бит программе readfile, то ей временно предоставляются права владельца файла.

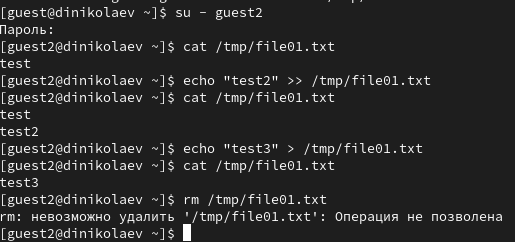
* 
* Чтение программы readfile.c с помощью readfile
* 
* Чтение программы/etc/shadow с помощью readfile

## 3.2 Исследование Sitcky-бита

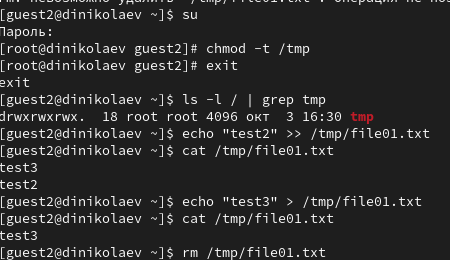
1. Выясним, установлен ли атрибут Sticky на директории /tmp, для чего выполним команду ls -l / | grep tmp. От имени пользователя guest создадим файл file01.txt в директории /tmp со словом test: echo “test” > /tmp/file01.txt. Просмотрим атрибуты у только что созданного файла и разрешим чтение и запись для категории пользователей «все остальные» ([??]):
   * ls -l /tmp/file01.txt
   * chmod o+rw /tmp/file01.txt
   * ls -l /tmp/file01.txt

* 
* Проверка Sticky-бита и создание файла file01.txt с правами на чтение и запись
* Установим права на запись и чтение для всех категорий пользователей ([??]).
* 
* Установка прав на запись и чтение file01.txt

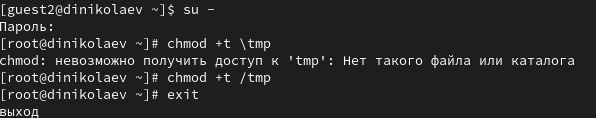
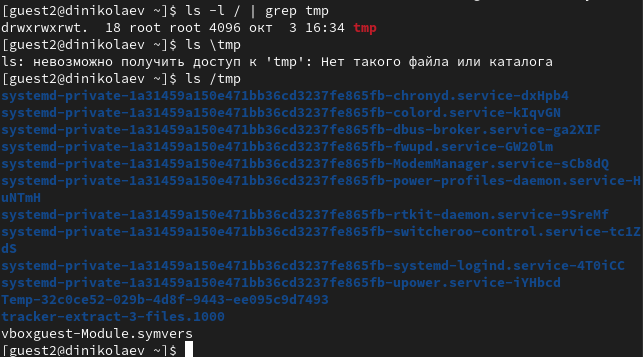
1. От имени пользователя guest2 (не являющегося владельцем) попробуем прочитать файл /tmp/file01.txt: cat /tmp/file01.txt, дозаписать текст в него командой echo “test2” >> /tmp/file01.txt, перезаписать файл командой echo “test3” > /tmp/file01.txt и попробуем удалить файл. Все действия кроме удаления выполнить удалось ([??]).

* 
* Проверка некоторых действий с файлом file01.txt от имени пользователя guest2

1. Снимем атрибут t (Sticky-бит) с файла file01.txt от имени суперпользователя и повторим действия из предыдущего шага ([??]). Помимо уже успешно выполнимых действий с файлом, мы теперь смогли удалить файл от имени пользователя, не являющегося его владельцем. Таким образом, Sticky-bit позволяет защищать файлы от случайного удаления, когда несколько пользователей имеют права на запись в один и тот же файл. Если у файла атрибут t стоит, значит пользователь может удалить файл, только если он является пользователем-владельцем файла или каталога, в котором содержится файл. Если же этот атрибут не установлен, то удалить файл могут все пользователи, которым позволено удалять файлы из каталога.

* 
* Проверка некоторых действий с файлом file01.txt без Sticky-бита от имени пользователя guest2

1. Повысим свои права до суперпользователя и вернём атрибут t на директорию /tmp ([??]) и проверим возвращение атрибута ([??]).

* 
* Возвращение Sticky-бита на директории /tmp
* 
* Проверка атрибутов на директории /tmp

# 4 Выводы

В ходе выполнения лабораторной работы я изучил механизмы изменения идентификаторов, применения SetUID и Sticky-битов. Я получила практические навыки работы в консоли с дополнительными атрибутами. Я рассмотрела работу механизма смены идентификатора процессов пользователей, а также влияние бита Sticky на запись и удаление файлов.

# Список литературы

1. Кулябов Д. С., Королькова А. В., Геворкян М. Н Лабораторная работа №5 [Электронный ресурс]. RUDN, 2023. URL: <https://esystem.rudn.ru/pluginfile.php/2090208/mod_resource/content/2/005-lab_discret_sticky.pdf>.