МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ им. Н.Э. БАУМАНА
Факультет «Информатика и системы управления»
Кафедра «Автоматизированные системы обработки информации и управления»

Сёмкин П.С., Сёмкин А.П.

Методические материалы к лабораторным работам
по дисциплине
«Операционные системы»
(кафедра СГН3)

Лабораторная работа № 5

«ОС Ubuntu. Жесткие и символические ссылки»

Москва
2023 г.

ОГЛАВЛЕНИЕ

1 Цель работы

Целью работы является знакомство с физической реализацией каталогов и файлов ОС Ubuntu и создание жёстких и символических связей с данными.

2 Теоретическая часть

2.1 *Структуры данных файлов и каталогов*

Основными объектами файловой системы ОС Ubuntu являются **файлы и каталоги.**

Физическая реализация хранения файла заключается в хранении **атрибутов и данных файла.**

С каждым файлом связана структура данных, называемая **i-узлом (index node** – индексный узел), содержащей атрибуты файла и адреса блоков данных файла.

Каталоги - это **системные файлы**, обеспечивающие поддержку структуры файловой системы. Каждый каталог содержит информацию о файлах и других каталогах(подкаталогах), входящих в данный каталог.

В файловой системе ОС Ubuntu принята иерархическая организация данных (дерево каталогов). Каждому файлу соответствует запись в каталоге. Эта запись содержит **имя файла и номер i-узла**.

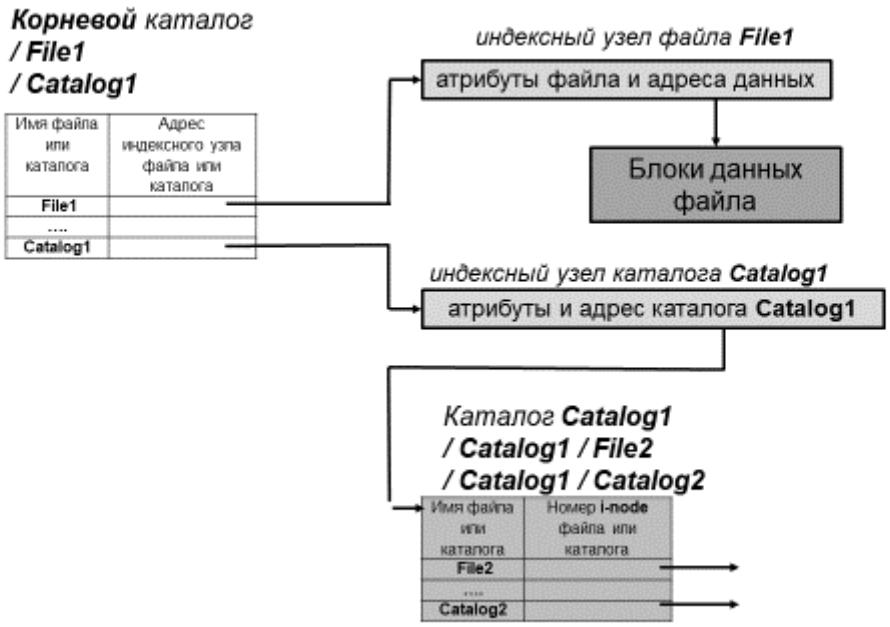

Рисунок 1. Структуры данных каталогов и файлов

2.2 *Совместное использование файлов*

Часто оказывается необходимым, чтобы один и тот же файл был одновременно доступен в различных каталогах, принадлежащих одному и тому же или различным пользователям.

Задача совместного использования файлов в файловых системах, использующих **i-узлы**, имеет два решения - создание **“жестких ссылок” (hard link) и “символических ссылок” (soft link) на файл.**

2.2.1 Жесткие ссылки (hard link).

Метод жёстких ссылок предполагает создание **ссылок из нескольких каталогов на один и тот же индексный узел**, описывающий какой-либо файл

Жесткие ссылки можно создавать только для файлов (а не для файлов и каталогов) и только в пределах одной файловой системы (внешнего накопителя).

Жесткие ссылки для операционной системе UNIX и реальное имя файла идентичны, после создания жесткой ссылки нельзя определить, какое имя первоначально являлось оригиналом (Поэтому любой файл всегда имеет как минимум одну жесткую ссылку – его имя, под которым он был создан.).

При удалении жесткой ссылки подсчитывается оставшееся количество ссылок, указывающих на индексный узел, и не освобождаются блоки данных

файла на физическом носителе до тех пор, пока не удалится его последняя ссылка.

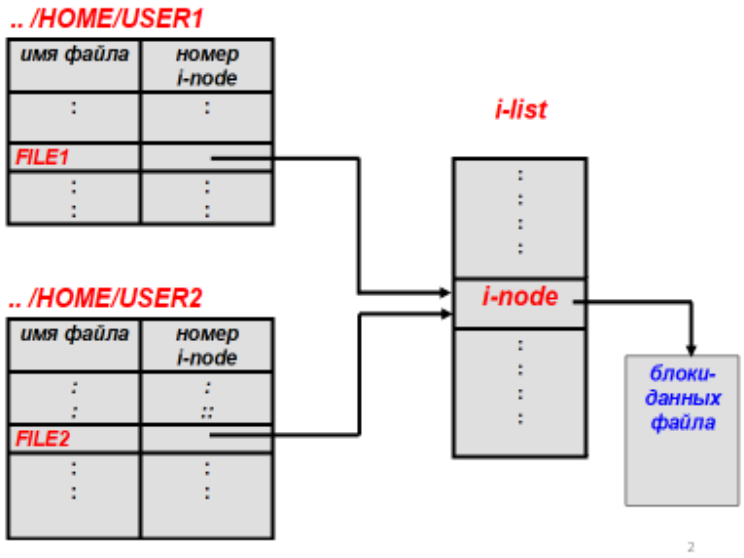

Рисунок 2. Жёсткие ссылки на атрибуты и данные файла

Таким образом, особенности жёстких ссылок следующие:

- ***работают только в пределах одной файловой системы;***
- ***нельзя ссылаться на каталоги;***
- ***имеют ту же информацию inode и набор разрешений что и у исходного файла;***
- ***разрешения на ссылку изменятся при изменении разрешений файла;***
- ***можно перемещать и переименовывать и даже удалять файл без вреда ссылке.***

2.2.2 Символическая ссылка (soft link, symbolic link) на файл

Символическая или косвенная ссылка обеспечивает возможность вместо имени файла (с путем) или каталога указывать имя ссылки; т.е. символическая ссылка представляет собой псевдоним (*текстовую подстановку*) для имени. Файл, на который указывает символическая ссылка, и сама ссылка представляют собой *разные* объекты файловой системы. Поэтому можно создавать ссылки на несуществующие файлы и каталоги, удалять оригинальные файлы, не удалив при этом ссылку. Можно создавать ссылки на ссылки и т.п.

Когда пользователь B устанавливает связь с одним из файлов пользователя C, система создает в каталоге пользователя B новый файл типа LINK (связь). Новый файл содержит просто имя пути к файлу, с которым он связан. Когда пользователь B читает данные из связанного файла, операционная система видит, что обращение производится к файлу типа LINK, поэтому она открывает файл, с которым связан этот файл, и читает данные из него. Такой метод называется символическим связыванием.

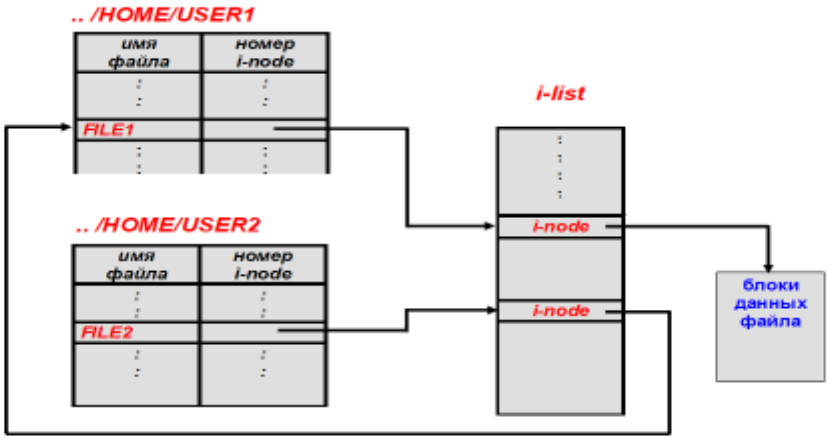

Рисунок 3.Символическая ссылка на атрибуты и данные файл

Основные особенности символических ссылок:

- ***могут ссылаться на файлы и каталоги;***
- ***после удаления, перемещения или переименования файла становятся недействительными;***
- ***права доступа и номер inode отличаются от исходного файла;***
- ***при изменении прав доступа для исходного файла, права на ссылку останутся неизменными;***
- ***можно ссылаться на другие разделы диска;***
- ***содержат только имя файла, а не его содержимое***

При символической ссылке не используются биты прав доступа (они всегда отображаются, как rwxrwxrwx). Вместо этого, права доступа к файлу, полученному символической ссылкой, определяются правами доступа к файлу, на который он ссылается

2.2.3 Недостатки жёстких и символических ссылок.

В методе жёстких ссылок, когда пользователь B устанавливает связь с совместно используемым файлом, то в i-узле владельцем файла числится пользователь A, создавший файл. Создание связи не изменяет владельца файла, а только увеличивает счетчик i-узла, что позволяет системе отслеживать количество записей в каталогах, ссылающихся на этот файл. Если впоследствии пользователь A попытается удалить этот файл, операционная система столкнется с проблемой. Если она удалит файл и очистит i-узел, то у пользователя B окажется запись в каталоге, указывающая на неверный i-узел. Если этот i-узел впоследствии будет назначен другому файлу, связь в каталоге пользователя B будет ссылаться на неверный файл. По значению счетчика i-узла система сможет определить, что этот файл используется кем-то еще, но у нее нет способа найти все записи в каталогах, указывающие на этот файл, чтобы удалить их. Указатели на каталоги не могут храниться в i-узлах, так как число таких каталогов ничем не ограничивается.

Единственное, что может сделать операционная система - это удалить запись в каталоге пользователя A, но оставить на месте i-узел, уменьшив значение счетчика на 1. Таким образом, мы теперь получили ситуацию, в которой пользователь B является единственным пользователем, имеющим ссылку на файл, владельцем которого считается пользователь A. Если система ведет учет или выделяет ограниченные квоты на использование дискового пространства, пользователь A будет продолжать получать счета за этот файл до тех пор, пока пользователь B не решит, наконец, удалить его. В этом случае счетчик i-узла уменьшится до нуля и система удалит файл.

При символическом связывании такой проблемы не возникает, так как указатель на i-узел есть только у владельца файла. У остальных пользователей, установивших связь с этим файлом, есть только пути к файлу, а не указатели на i-

узел. Когда владелец удаляет файл, файл уничтожается. Последующие попытки использовать этот файл при помощи символической связи не будут иметь успеха, так как системе не сможет найти файл. Удаление символьной связи никоим образом не повлияет на файл.

Недостатком символической связи является необходимость накладных расходов. Чтобы получить доступ к i-узлу, следует сначала прочитать файл, содержащий путь. Затем следует пройти по этому пути, открывая каталог за каталогом, пока, наконец, не будет получен нужный i-узел. Все эти действия могут потребовать существенного количества обращений к диску.

Кроме того**,** для каждой символической связи требуется дополнительный i-узел, а также дополнительный блок диска для хранения пути к файлу, хотя, если имя пути короткое, то в качестве оптимизации система может хранить его в самом i-узле. Преимущество символических связей состоит в том, что они могут использоваться для ссылок на файлы, расположенные на удаленных компьютерах. Для этого, кроме обычного пути файла, нужно всего лишь указать сетевой адрес машины, на которой файл располагается.

Использование связей создает еще одну проблему. При использовании связей у одного файла оказывается несколько путей. Программы, сканирующие каталоги со всеми подкаталогами, могут наткнуться на один и тот же файл несколько раз. Если такая программа, например, записывает все файлы на магнитную ленту, она запишет на ленту несколько копий одного и того же файла. Более того, если запись с этой ленты будет потом прочитана на другом компьютере, этот файл окажется скопированным на диск несколько раз, если только считывающая ленту программа не догадается, что это один и тот же файл.

2.3 *Создание жёстких и символических ссылок в командной строке*

Для создания жёстких и символических ссылок используется утилита командной строки **ln**

$ ln параметры <file1> <file2>

file1 - существующая жёсткая или символическая ссылка
file2 - создаваемая жёсткая или символическая ссылка

Параметры утилиты:

- **-d** – разрешить создавать жесткие ссылки для директорий суперпользователю;
- **-f** – удалять существующие ссылки;
- **-i** – спрашивать нужно ли удалять существующие ссылки;
- **-P** – создать жесткую ссылку;
- **-r** – создать символическую ссылку с относительным путем к файлу;
- **-s** – создать символическую ссылку.

Примеры использования утилиты **ln** приведены в приложении

3 Выполнение работы

3.1 *Задание.*

1. Создать каталоги и файлы

2.1 2.В каталоге **КАФЕДРА** создать каталог **СТЕНА**

2.2 В каталоге **СТЕНА** создать файл **лента.txt** с полным доступом всех членов группы **stud3k** и пользователя **admin_kaf**

2.3 В каталоге **СТЕНА** создать каталог **РАСПИСАНИЕ**

2.4 В каталоге **РАСПИСАНИЕ** создать файлы расписаний с полным доступом пользователя **admin_kaf** и возможностью чтения пользователям **stud_51 – stud_54**

Расписание_первого_курса.txt

Расписание_второго_курса.txt

Расписание_третьего_курса.txt

Расписание_четвёртого_курса.txt

3 Создать жёсткие и символические ссылки на файлы и каталоги

3.1 В домашних каталогах пользователей **stud_51 – stud_54** создать жёсткие ссылки на файл **лента.txt**

3.2 В домашних каталогах пользователей **stud_51 – stud54** создать символические ссылки на каталог **РАСПИСАНИЕ** и файл **Расписание_третьего_курса.txt**

4 Проверить правильность доступа по соответствующим жестким и символическим ссылкам

3.2 *Порядок выполнения работы.*

1. Войти в систему под учётной записью **user2**.
2. Запустить программу виртуализации **Oracle VM VirtualBox**
3. Запустить виртуальную машину **Uduntu_XX**
4. Войти в систему под учётной записью **admin_kaf.**
5. В каталоге **КАФЕДРА** создать каталог **СТЕНА**
6. Установить разрешения для каталога **СТЕНА**
7. В каталоге **СТЕНА** создать каталог **РАСПИСАНИЕ**
8. Создать текстовые файлы в каталогах **СТЕНА** и **РАСПИСАНИЕ**
9. В домашних каталогах пользователей **stud_51 – stud_54** создать жёсткие ссылки на файл **лента.txt**
10. В домашних каталогах пользователей **stud_51 – stud54** создать символические ссылки:
    - на каталог **РАСПИСАНИЕ**
    - на файл **Расписание_третьего_курса.txt**
11. Проверить правильность создания жёстких и символических ссылок

4 Контрольные вопросы

1. Что такое индексный узел?
2. Какая информация о файле находится в каталоге и какая – в индексном узле?
3. В чём разница между жёсткими и символическими ссылками?

5 ЛИТЕРАТУРА

1. Сёмкин П.С., Аксёнов А.Н. Файловые системы. Логическая организация и физическая реализация. Сборник учебно-методических работ кафедры «Системы обработки информации и управления» (бакалавры). Учебное пособие. Вып. 1./Под ред: В.М. Черненького. –М: «АртКом», 2013. – стр. 95-120
2. Сёмкин П.С., Семкин А.П. Файловые системы операционных систем Windows и Unix. Сборник учебно-методических работ кафедры «Системы обработки информации и управления» (бакалавры). Учебное пособие. Вып. 2./Под ред. В.М. Чёрненького. –М: «АртКом», 2014. – стр. 160-189
3. Негус К. Ubuntu и Debian Linux для продвинутых. 2-е изд. – СПб.: Питер,2014. -384 с.: ил.

6 Приложение. Основные команды для работы с файлами и каталогами

6.1 *Создание файлов и каталогов в командной строке*

$ pwd - вывод на экран пути к текущему каталогу

$ ls - вывод списка файлов и каталогов текущего каталога

$ ls -l - вывод списка файлов и каталогов в форматированном виде

$ ls -la - вывод списка файлов и каталогов в форматированном виде, в том числе начинающихся с точки

$ cd - переход в домашний каталог пользователя

$ cd /home - переход в каталог **/home**

$ cd .. - переход в родительский каталог данного каталога

$ cd / - переход в корневой каталог файловой системы

$ sudo mkdir имя каталога - создание каталога

$ sudo chown Владелец: Имя каталога или файла - изменение владельца файла каталога или файла

$ sudo chgrp Владелец: Имя каталога или файла - изменение группы каталога или файла

$ touch <путь> <имя файла> - создание пустого файла

$ touch /tmp/newfile1.txt - создание пустого файла)

$ ls –li <имя файла> - вывод информации о файле(включая информацию о индексном дескрипторе

cp – копирование файла

cat - вывести на экран файл (**cat <имя файла>**) или ввод с консоли (**cat >> <имя файла>**). (остановить ввод - **Ctrl+Z**)

6.2 *Создание жёстких и символических ссылок*

$ ln file1 file1-hard - создание жёсткой ссылки **file1-hard** на файл **file1**

$ ln –s /home/fire/file1 /home/file1-link - создание символической ссылки **file1-link** на файл **file1**