|  |  |  |  |
| --- | --- | --- | --- |
| **《操作系统原理》实验报告** | | | |
| 实验名称 | 文件系统 | 实验序号 | 5 |
| 实验日期 | 2023/4/25 | 实验人 |  |
| 一、实验题目  下面的练习检查了 UNIX 或 Linux 系统上文件和 inode 之间的关系。在这些系统上，文件用 inode 表示。也就是说，一个 inode 是一个文件（反之亦然）。您可以在随本文提供的 Linux 虚拟机上完成此练习。您也可以在任何 Linux、UN IX 或 Mac OS X 系统上完成练习，但需要创建两个名为 file1.txt 和 file3.txt 的简单文本文件，其内容是唯一的句子。 | | | |
| 二、相关原理与知识   1. inode   是UNIX或类UNIX操作系统中的一种数据结构，其本质是结构体，用于存储文件或目录的元数据，如文件的所有者、创建时间、修改时间、访问权限、大小、块的位置等等。每个文件和目录都有一个唯一的inode节点，它们被文件系统用来却分文件和目录，并管理它们的元数据  当一个文件被创建时，操作系统会在文件系统中为其分配一个inode节点，并为该文件指定一个唯一的inode号。inode节点包含了文件的所有元数据信息，但不包括文件的实际内容。实际的文件内容被存储在磁盘块中，并由inode节点指向这些块  每个inode节点都有唯一的inode号，该号码在整个文件系统中是唯一的。因此，文件系统可以使用inode号码来识别一个文件或目录，并跟踪文件和目录的元数据。当文件名被更改时，inode号码保持不变，因此文件系统可以通过inode号码来找到文件  由于文件系统使用inode号码来管理文件和目录，因此文件名并不是文件本身的一部分。这意味着，如果文件名被更改或删除，文件本身仍然存在，并且可以通过inode号码来找到它，这也是硬链接允许多个文件名指向同一inode节点的原因。   1. 硬链接   硬链接时将一个文件链接到文件系统上的一个位置，这个链接的文件和原始文件共享相同的inode节点和硬盘空间，即它们是同一个文件的两个不同名字，因此硬链接不会创建新的inode信息，也不会更改inode的总数。这意味着，硬链接的文件和原始文件在文件系统中的位置和内容是相同的，如果修改其中任何一个文件，另一个文件也会发生相应的更改。删除其中任何一个文件并不会对另一个文件造成影响。硬链接智能链接到同一个文件系统中的文件，因为文件系统是通过inode节点来查找文件的，而不是通过路径名。   1. 软链接   软链接是一个指向另一个文件或目录的特殊文件，它创建一个新的文件或目录，并在其中添加一个指向原始文件或目录的链接。软链接是一个指向文件或目录的路径名，它们与原始文件或目录不共享inode节点，会新建自己的inode信息和block，只是在block中不存储实际的文件数据，而存储的是源文件的文件名及inode号。软链接文件本身只包含指向原始文件或目录的路径，而不是实际的数据。如果原始文件或目录被删除或移动，软链接会失效。软链接可以链接到不同文件系统上的文件或目录。   1. ln命令   ln是在UNIX和类UNIX操作系统中用于创建链接的命令。它允许用户创建硬链接和软链接。  其常见用法如下：  1. 创建硬链接   |  | | --- | | **ln** source\_file hard\_link |   创建一个名为hard\_link的硬链接，指向source\_file文件  2. 创建软链接   |  | | --- | | **ln** -s source\_file symbolic\_link |   创建一个名为symbolic\_link的软链接，指向source\_file文件  3. 创建目录的软链接   |  | | --- | | **ln** -s **/**path**/**to**/**source\_directory symbolic\_link |   创建一个名为symbolic\_link的软链接，该链接指向source\_directory目录 | | | |
| 三、实验过程及实验结果   1. **创建两个文件 file1.txt和file3.txt。每个文件中各包含一句话。**   创建结果如下：    这两个文件的初始内容分别为：  file1.txt：This is the first file.  file3.txt：This is the second file.   1. **使用命令获取file1.txt的inode编号**   针对查询inode编号，使用命令 ls -li filename，即可查看对应“filename”文档的inode编号，实验结果如下：     1. **使用ln命令在源文件和目标文件之间创建硬链接，用命令获取file2.txt的inode编号，与file1.txt相比较，比较两个文件的内容**   硬链接的创建如下：    对于file2.txt的inode编号：    可以看到，file1.txt的inode编号和file2.txt的inode编号相同，这是体现了硬链接的特征，即硬链接的文件和原始文件共享相同的inode节点，它既不会建立新的inode信息，也不会更改inode总数。  对于两个文件的内容，分别如下：    可以看到，file1和file2的内容相同，这是由于硬链接共享硬盘空间，因此这两个文档的文本内容是相同的。   1. **编辑file2.txt并更改其内容。完成以后检查file1.txt的内容。比较file1.txt和file2.txt的内容**   此处此处我更改file2文档的内容，在文档最后加上了“（modified）”的文本标识，其实验结果如下：    可以看到，当我针对file2进行修改后，file1的内容也被修改。  上面提到过，硬链接共享硬盘空间，这意味着，硬链接的文件和原始文件在文件系统中的位置和内容是相同的，如果修改其中任何一个文件，另一个文件也会发生相应的更改，正如实验中展示的一样，当我修改file2时，file1也被修改；如果修改file1，那么file2也会被进行相同的修改。   1. **接下来，输入命令删除file1.txt，再查看file2.txt是否存在**   删除file1时，我们使用rm命令，之后再查看file2是否存在。实验结果如下：      可以看到，当删除了file1以后，file2仍然可以正常打开并显示其中的内容，这也是硬链接的特性之一，即删除其中任何一个文件并不会对另一个文件造成影响，在这里也得到了验证。   1. **输入命令删除file2.txt文件**   题目中要求使用strace跟踪删除file2指令的执行情况，使用命令 strace rm file2.txt运行即可。实验结果如下：    可以发现，在删除file2时，系统执行了openat、close、read、mmap、fstat、unlinkat等系统调用，其中删除硬链接主要为unlinkat命令。  以下是大致解释：  openat：该系统调试试图打开“file2.txt”以读取它的内容。由于使用的是相对路径，此处open退化成openat。  fstat：该系统调用获取文件描述符为3的文件的元数据，例如文件类型、访问权限等。  mmap：将文件映射到内存中。映射文件到内存中可以提高文件的访问速度，因为内存中的数据访问比磁盘中的数据访问速度更快。在实验中，mmap系统调用映射了file2.txt文件，以便进程可以读取它的内容。  read：读取文件内容。由openat可知“read”的系统调用号为3，并且该系统调用试图以只读模式打开“file2.txt”文件以便读取它的内容。  unlinkat：系统尝试删除“file2.txt”硬链接。由于“file2.txt”是“file1.txt”的硬链接，因此并不会实际删除“file1.txt”，只会减少硬链接的数量。  close：关闭文件流。   1. **输入命令创建file3.txt文件的软链接，使用命令获取file3.txt和file4.txt的inode编号**   软链接建立：    之后使用命令查询file3和file4的inode号：    可以看到，file3和file4两个文档的inode值并不相同，这符合软链接的特性。即，软链接是一个指向文件或目录的路径名，它们与原始文件或目录不共享inode节点，会新建自己的inode信息和block，只是在block中不存储实际的文件数据，而存储的是源文件的文件名及inode号。软链接文件本身只包含指向原始文件或目录的路径，而不是实际的数据。   1. **编辑file4.txt的内容，查看file3.txt的内容是否被修改**   更改file4.txt的内容，此处我在file4.txt的最前面加上了(modified)标识，修改完成以后对file3.txt的内容进行观察。实验如下：    可以发现，当修改软链接中的某一文件时，原文件的内容也会被修改，这也符合软链接的特点。当创建一个软链接时，实际上创建的是一个新的文件，这个新文件指向了另一个文件，而不是复制原文件。当使用一个软链接来访问另一个文件时，实际上访问的是原文件，因此对原文件的修改会影响软链接，对软链接的修改也会影响原文件。   1. **删除file3.txt，查看file4.txt是否仍存在**   此处直接使用rm指令删除软链接的原文件file3，对file4进行观察。实验结果如下：    可以看到，当删除软链接中的源文件时，软链接产生的文件也直接被删除无法访问。上面提到过，软链接创建的新的文档实际上是指向了原文档，如果原始文件或目录被删除或移动，软链接会失效。如实验所示，当删除原文档file3后，软链接即失效，file4也被删除。 | | | |
| 四、问题总结  本次实验并未遇到特别的问题，仅熟悉了linux系统中的文件系统与操作。 | | | |
| 五、源代码  本次实验未涉及编程内容，所有操作指令均已在实验内容中呈现 | | | |