常用命令

查看帮助手册

使用 man(manual) 命令可以查看 Linux 内置的帮助手册。该手册分为多卷：第一卷是用来查看 shell 命令的；第二卷是用来查看系统调用相关信息的；第三卷是用来查看库函数信息的...

```
1    可执行程序或 shell 命令
2    系统调用(内核提供的函数)
3    库调用(程序库中的函数)
4    特殊文件(通常位于 /dev)
5    文件格式和规范，如 /etc/passwd
6    游戏
7    杂项(包括宏包和规范，如 man(7)，groff(7))
8    系统管理命令(通常只针对 root 用户)
9    内核例程 [非标准
```

man 命令的格式如下：

```
$ man [手册编号] cmd
```

比如，我们可以这样用：

```
$ man man
$ man 3 mkdir
```

进入帮助界面后，我们可以按下面按键浏览帮助信息：

```
d(down):     往下翻半页
u(up):       往上翻半页
f(forward): 往下翻一整页
b(backward):往上翻一整页
q(quit):     退出
```

Tips: Be a man!

关机命令

关闭主机之前，请务必先关闭虚拟机！否则，可能会损坏虚拟机文件，导致不能启动虚拟机。

```
$ man shutdown
    shutdown - 挂起，关机或重启计算机
常用选项：
    -H, --halt: 挂起
    -P, --poweroff: 关机(默认)
    -r, --reboot: 重启
    -c(cancel): 取消
```

比如，我们可以这样用：

```
$ sudo shutdown
    广播关机消息给所有用户，并于一分钟后关机。
$ sudo shutdown now
    立刻关机
```

1 用户子系统

Linux 用户可以分为：超级用户(root)和普通用户。超级用户又被称为：根用户和特权用户，它拥有至高无上的权利。普通用户又被划分为管理用户和其它用户。管理用户即我们俗称的 sudoers，他们可以临时提升权限(使用sudo命令)，安装Ubuntu过程中创建的用户默认就是 sudoers。

查看所有用户

Linux系统下的所有用户，在 passwd 配置文件中都有一条相关记录。

```
$ sudo cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
...
he:x:1000:1000:he,,,:/home/he:/bin/bash
```

每一行对应一条记录，每条记录有多个字段，字段之间以：分割。我们可以在5号手册中查看 passwd 文件的格式和规范：

```
$ man 5 passwd
/etc/passwd 为每个用户账户包含一行，包含使用冒号 (“:”) 分隔的七个字段，分别是：
    ·   登录名
    ·   可选的加密后的密码
    ·   数字用户 ID
    ·   数字组 ID
    ·   用户名和注释字段
    ·   用户主目录
    ·   可选的用户命令解释器
```

添加用户

我们可以使用 useradd 命令来添加用户，格式如下：

```
格式：
    useradd [选项] username
常用选项：
    -m, --create-home
        如果不存在，则创建用户主目录
    -s, --shell SHELL
        用户的登录 shell 名。
```

比如我们可以这样用：

```
$ sudo useradd test1
    这样创建的用户不会自动创建用户的家目录，默认的shell为sh
$ sudo useradd -m -s /bin/bash test2
    这样创建的用户会自动创建用户的家目录，并可以指定使用哪个shell
```

删除用户

我们可以使用 userdel 命令来删除用户，格式如下：

```
格式：
    userdel [选项] username
常用选项：
    -r, --remove
        用户主目录中的文件将随用户主目录和用户邮箱一起删除。
```

比如我们可以这样用：

```
$ sudo userdel test1
    这样删除用户，不会删除用户的家目录和用户的邮箱。
$ sudo userdel -r test2
    这样删除用户，会把用户的家目录和用户的邮箱一起删除。
```

设置密码

我们可以使用 passwd 命令来给用户设置密码：

```
$ man passwd
    passwd 命令用来更改用户账户的密码。普通用户通常只更改其自己账户的密码，而超级用户可以更改任何账户的密码。
格式：
    passwd [选项] [username]
```

比如我们可以这样用：

```
$ sudo passwd test1
$ sudo passwd root
```

切换用户

我们可以使用 su(switch user) 命令切换到另一个用户，格式如下：

```
格式：
    su [选项] [username]
```

比如我们可以这样用：

```
$ sudo su
    切换到root用户
$ sudo su test1
    切换到test1用户
```

退出切换

exit 命令可以用来退出用户切换

```
$ exit
```

使用 su 命令依次切换到多个用户时候，这些用户是使用**栈结构**来管理的。执行 su 命令相当于将用户压入栈顶，执行exit命令相当于将用户弹出栈顶。

注意：当用户在上述的栈结构中存在的时候，该用户是不能够被删除的。

2 文件子系统

2.1 目录相关命令

和 Windows 不一样，Linux是以树的结构来管理文件系统的。我们将树的根结点称为根目录，用 / 表示。

常见目录

在Linux中，我们是以分门别类的方式来管理文件的，也就是说，我们会将功能相似的文件放到同一个目录下进行管理。这和Windows按工程组织文件的方式不太一样。

常见目录的功能如下表所示：

目录名	功能
/bin(binary)	存放可执行程序或脚本文件
/sys(system)	存放和系统相关的文件
/dev(device)	存放设备文件
/etc	一般用来存放配置文件和启动脚本
/lib(library)	存放系统库文件
/var(variable)	存放变化很快的文件，比如日志文件
/proc(process)	存放进程相关的数据
/root	root用户的家目录
/home/{username}	普通用户的家目录

查看当前工作目录

pwd(print working directory) 命令可以查看当前工作目录

```
$ pwd
/home/he
```

改变当前工作目录

cd(change directory) 命令可以改变工作目录

```
$ man cd
    cd - change the working directory
格式:
    cd [选项] [directory]
```

常用方式：

```
$ cd            # 切换到用户家目录
$ cd /usr/lib   # 切换到/usr/lib目录
$ cd /          # 切换到 / 目录
$ cd ~          # 切换到用户家目录
$ cd .          # 切换到当前工作目录(不切换)
$ cd ..         # 切换到父目录
$ cd -          # 切换到上一次目录
```

> 注意：上一次目录保存在环境变量OLDPWD中(可以通过env命令查看)，它不是用栈结构管理的。

创建目录

我们可以用 mkdir 命令来创建目录

```
$ man mkdir
    mkdir - make directories
格式:
    mkdir [选项] directory...
常用选项:
    -p, --parents
        如果父目录不存在，则创建父目录
```

常用方式:

```
$ mkdir dir
$ mkdir dir1 dir2 dir3
$ mkdir -p a/b/c
```

删除空目录

rmdir 只能用来删除空目录

```
$ man rmdir
    rmdir - remove empty directories
格式:
    rmdir [选项] dirctory...
常用选项:
    -p, --parents
        递归删除空目录
```

常用方式:

```
$ rmdir dir
$ rmdir dir1 dir2 dir3
$ rmdir -p a/b/c
```

通配符

通配符(wildcard character)，可以用于匹配单个或多个字符。其中:

```
*: 匹配任意多个字符(包括0个)
?: 匹配任意一个字符
集合(类): [characters]匹配集合内任意一个字符。
          [!characters]匹配集合外任意一个字符。
          比如: [abc], [!abc], [0-9], [a-z], [0-9A-Za-z_]等
```

rmdir 命令可以和通配符一起使用（通配符可以提升命令的威力）：

```
$ rmdir dir?
$ rmdir dir*
$ rmdir [!abc]
```

查看目录内容

ls(list) 命令可以查看目录内容：

```
$ man ls
    ls - list directory contents
格式:
    ls [选项] [FILE]...
常用选项:
    -a, --all
        显示所有的内容，包括以.开头的文件和目录
    -i, --inode
        显示文件的inode编号(inode是物理文件的标识)。
    -l
        以长格式的形式显示目录中的内容
    -h, --human-readable
        和-l选项一起使用，以人类可读的方式显示文件的大小。
```

常用方式：

```
$ ls                    # 查看当前工作目录
$ ls dir                # 查看dir
$ ls dir1 dir2 dir3     # 查看dir1,dir2,dir3
$ ls -a dir             # 查看dir中的所有内容，包括以.开头的文件和目录
$ ls -ilh dir           # 显示inode编号，显示详细信息，并以人类可读的方式显示文件的大小
```

接下来，我们一起来看一下目录项的详细信息：

```
$ ls -l
总用量 8560
-rw-rw-r--  1 he he 4349666 3月   7 11:12 a.txt
drwxrwxr-x  2 he he    4096 3月   2 21:18 c
...
```

- 第1列的第一个字母用来表示文件的类型。

```
-: 普通文件
d(directory): 目录
c(character): 字符设备文件(鼠标，键盘，显示器...)
b(block): 块设备文件(磁盘)
l(symbolic link): 符号链接
s(socket): 本地套接字
p(named pipe): 有名管道
```

- 第1列后面九个字符(分为3组)表示权限。
  依次代表拥有者、拥有组和其他用户的读、写、执行权限。可读则显示r，可写则显示w，可执行则显示x， 没有相关权限则显示-。
- 第2列表示硬链接数。
- 第3列表示拥有者。
- 第4列表示拥有组。
- 第5列表示文件所占空间的大小。
- 第6列表示最近修改时间。
- 第7列为文件名。

以树状结构显示目录内容

tree 命令不是Linux系统自带的命令，使用之前，我们必须先安装 tree 命令：

```
$ sudo apt install tree
```

tree 命令可以以树状结构显示目录内容：

```
$ man tree
    tree - list contents of directories in a tree-like format.
格式:
    tree [选项] [directory]...
```

常用方式：

```
$ tree                  # 以树状结构显示当前工作目录的内容
$ tree dir              # 以树状结构显示目录dir中的内容
$ tree dir1 dir2 dir3   # 以树状结构显示目录dir1, dir2, dir3中的内容
```

复制文件或目录

cp(copy) 命令可以用来复制文件和目录。

```
$ man cp
    cp - copy files and directories
格式:
    cp [选项] SOURCE DEST
    cp [选项] SOURCE... DEST
常用选项:
    -n, --no-clobber
        如果文件已存在，则不覆盖(默认会覆盖已有文件)。
    -i, --interactive
        如果文件存在，则给用户提示信息(由用户决定是否覆盖?)
    -R, -r, --recursive
        递归复制(用于copy目录)
```

常用方式:

```
$ cp text1 text2                    # 将text1复制到text2中；如果text2存在，则覆盖。
$ cp text1 text2 text3 dir          # 将text1,text2,text3复制到目录dir中；如果文件已存在，则覆盖。
$ cp -n text1 text2                 # 将text1复制到text2中；如果text2存在，不覆盖。
$ cp -i text1 text2 text3 dir       # 将text1,text2,text3复制到目录dir中；如果文件已存在，则提示用户是否覆盖。
$ cp -r dir1 dir2                   # 递归地将目录dir1复制到目录dir2
```

移动文件和目录

mv(move) 命令可以用来移动文件和目录，我们也可以用 mv 命令对文件和目录重命名。

```
$ man mv
    mv - move (rename) files
格式:
    mv [选项] SOURCE DEST
    mv [选项] SOURCE... DEST
    Rename SOURCE to DEST, or move SOURCE(s) to DIRECTORY.
常用选项:
    -n, --no-clobber
        如果文件已存在，则不覆盖(默认会覆盖已有文件)。
    -i, --interactive
        如果文件存在，则给用户提示信息(由用户决定是否覆盖?)
```

Q: 为什么移动目录时，不需要指定 -r 选项?

常用方式：

```
$ mv text1 text2                    # 将text1重命名为text2；如果text2存在，则覆盖。
$ mv dir1 dir2                      # 将dir1重命名为dir2
$ mv -n text1 text2                 # 将text1重命名为text2；如果text2存在，不覆盖。
$ mv text1 text2 text3 dir          # 将text1,text2,text3移动到目录dir中；如果文件已存在，则覆盖。
$ mv -i text1 text2 text3 dir       # 将text1,text2,text3移动到目录dir中；如果文件已存在，则提示用户是否覆盖。
```

删除文件和目录

rm(remove)命令可以删除文件和目录。

```
$ man rm
    rm - remove files or directories
格式：
    rm [选项] FILE...
常用选项：
    -f, --force
        忽略不存在的文件，永远不提示
    -i
        在每次删除前，都提示用户是否删除
    -r, -R, --recursive
        递归删除
```

常用方式：

```
$ rm text1                  # 删除文件text1
$ rm text1 text2 text3      # 删除文件text1, text2, text3
$ rm -i *.txt               # 删除当前目录下所有以.txt结尾的文件，并提示用户是否删除
$ rm -rf dir                # 递归删除目录dir，不给出任何提示
```

注意：使用rm命令删除文件和目录后，就很难恢复了（Linux系统没有所谓的回收站）。所以除非必要，不要以 root 身份或者使用 sudo 权限来执行 rm 命令！

2.2 文件相关命令

创建文件

在 Linux 下创建文件的方式有多种，最常用的是以下三种方式：

```
 $ echo "Hello Linus Torvalds" > text    # 创建文件text，并且文件中包含内容"Hello Linus Torvalds"
 $ touch text1 text2 text3               # 创建空文件text1，text2，text3(要求text1,text2,text3不存在)
 $ vim hello.c                           # 编辑hello.c，按:wq退出。
```

后面我们会着重讲解vim编辑器~

查找文件

1. 我们可以使用 which 命令来查找可执行程序的路径：

```
$ man which
    which - locate a command
格式:
    which [-a] cmd...
选项:
    -a
        显示所有匹配的路径
```

常用方式：

```
$ which bash          # 查看bash的路径
$ which ls tree       # 查看命令ls和tree的路径
$ which -a vim        # 查看vim的所有路径（我们可能装了多个版本的vim）
```

which 是根据 PATH 环境变量中的路径依次去查找的，然后显示第一个匹配项，或者显示所有匹配项。

我们可以使用 env 命令查看环境变量：

```
$ env
...
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
...
```

2. find 命令可以在一个目录中递归地中查找符合指定条件的文件，它的功能非常强大：

```
$ man find
    find - search for files in a directory hierarchy
格式:
    find [start-point...] 查找条件       # 省略start-point，默认起始点为当前工作目录
```

常用选项：

```
    -name pattern
        查找文件名符合pattern的文件
    -type c
        查找类型为c的文件：
            b(block)：块设备文件
            c(character)：字符设备文件
            d(directory)：目录
            p(named pipe)：有名管道
            f(file)：普通文件
            l(symbolic link)：符号链接
            s(socket)：套接字
    -size n[cwbkMG]
        File uses n units of space, rounding up.
            b: for 512-byte blocks (this is the default if no suffix is
used)
            c: for bytes
            w: for two-byte words
            k: for Kibibytes (KiB, units of 1024 bytes)
            M: for Mebibytes (MiB, units of 1024 * 1024 = 1048576 bytes)
            G: for Gibibytes (GiB, units of 1024 * 1024 * 1024 =
1073741824 bytes)
        可以在n前面添加'+'和'-'，表示大于和小于。
    -empty
        查找空的文件或空的文件夹
    -user username, -uid uid
        根据用户名和用户id查找
    -group groupname, -gid gid
        根据组名和组id查找
    -perm mode
        根据权限查找
    根据时间查找：
        -amin n, -atime n, -cmin n, -ctime n, -mmin n, -mtime n
            a(access)：文件访问的时间
            c(change)：文件属性发生改变的时间
            m(modify)：文件内容发生改变的时间
            min：以分钟为单位
            time：以天为单位
        可以在n前面添加'+'和'-'，表示大于和小于。
    组合查找：
        -a(and), -o(or), !(not)：分别表示与、或、非
```

常用方式：

```
$ find /usr/include -name "stdio.h"     # 在/usr/include目录下查找stdio.h文件
$ find . -name "*.c"                    # 在当前工作目录下查找所有以.c结尾的文件
$ find /dev -type b                     # 在/dev目录下查找所有的块设备文件
$ find . -size 5M                       # 在当前工作目录下查找所有大小为5M的文件
$ find . -size +5M                      # 在当前工作目录下查找所有大于5M的文件
$ find dir1 dir2 dir3 -empty            # 在dir1,dir2,dir3目录下查找所有空的文件和空的文件夹
$ find . -user he                       # 在当前工作目录下查找he用户拥有的文件
$ find . -gid 0                         # 在当前工作目录下查找root组(gid=0)拥有的文件
$ find . -perm 664                      # 在当前工作目录下查找权限为664(rw-rw-r--)的文件
$ find . -mtime 1                       # 在当前工作目录下查找在[1, 2)天前内容发生修改的文件（find会省略小数部分）
$ find . -mtime +2                      # 在当前工作目录下查找在[3, )天前内容发生修改的文件
$ find /dev -type c -a -name "tty*"     # 在/dev目录下查找以tty开头的字符设备文件
$ find /dev -type b -o -name "tty*"     # 在/dev目录下查找块设备文件或者是以tty开头的文件
$ find /dev -type c -a ! -name "tty*"   # 在/dev目录下查找不以tty开头的字符设备文件
```

查看文件内容

1. 我们可以用 cat 命令查看文件

```
$ man cat
    cat - concatenate files and print on the standard output
格式:
    cat [选项] [file]...
常见选项:
    -n, --number
        对每一行进行编号
```

常见用法：

```
$ cat /etc/passwd
$ cat -n /etc/passwd
$ sudo cat /etc/passwd /etc/shadow
```

2. 使用 head 命令查看文件的前几行

```
$ man head
    head - output the first part of files
格式:
    head [选项] [file]...
常见选项:
    -n, --lines=[-]NUM
        显示前NUM行；若在NUM前面添加'-'号，则显示除了后NUM行的所有行
```

常见用法:

```
$ head text1                # 显示text1的前10行
$ head text1 text2 text3    # 显示text1,text2,text3的前10行
$ head -n 5 text1           # 显示text1的前5行
$ head -n -5 text1          # 显示除了最后5行的所有行
```

3. 使用 tail 命令查看文件的后几行

```
$ man tail
    tail - output the last part of files
格式:
    tail [选项] [file]...
常见选项:
    -n, --lines=[+]NUM
        显示后NUM行；若在NUM前面添加'+'号，则从第NUM开始显示
    -F
        显示后面追加的数据。一般用于查看日志文件。
```

常见用法:

```
$ tail text1                # 显示text1的后10行
$ tail text1 text2 text3    # 显示text1,text2,text3的后10行
$ tail -n 5 text1           # 显示text1的后5行
$ tail -n +5 text1          # 从第5行开始显示
$ tail -F server.log        # 显示后10行，并且会显示后面追加的数据
```

4. 使用 less (more) 单页浏览文件

```
$ man more
    more - file perusal filter for crt viewing (说人话就是单页浏览)
格式:
    more [选项] file...
```

NOTE：less 是 more 的高级版本，若系统中没有 less 命令，才考虑使用 more。

常见用法：

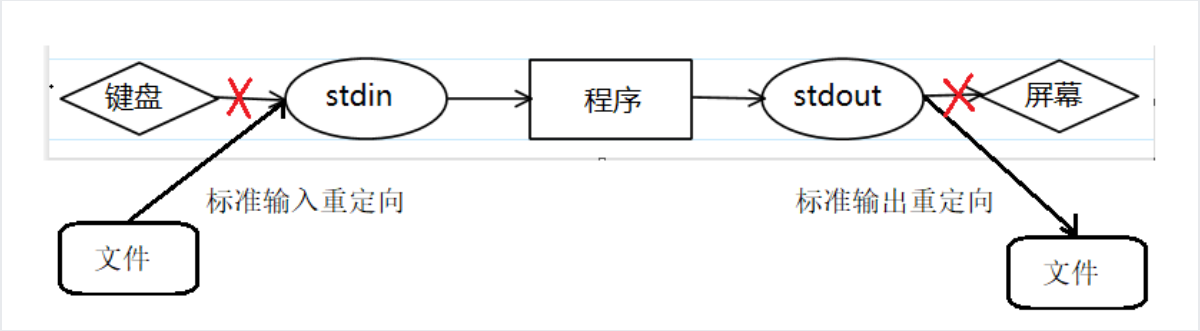
```
$ less text1              # 浏览text1
$ less text1 text2        # 浏览text1, text2
```

进入浏览界面后：

```
f(forward)      往后翻一页
b(backward)     往前翻一页
:n(next)        查看下一个文件
:p(previous)    查看上一个文件
q(quit)         退出浏览界面
```

重定向

cat，head，tail等命令经常和重定向一起使用，比如：tail -n 20 text1 > a.txt。重定向的示意图如下：



在讲重定向之前，我们需要引入文件描述符的概念(后面我们会详细讲解这个概念，现在我们只需要知道它是一个非负整数即可)。

标准流	文件描述符	重定向符号
stdin	0	<
stdout	1	> 和 >>
stderr	2	2> 或 2>>

常见用法：

```
$ who > users
$ echo "liu yi fei" >> users
$ wc -l < users
$ cat users file1 > text     # file1不存在
$ cat users file1 2> text    # file1不存在
$ cat users file1 >& text    # file1不存在
```

NOTE：输入重定向用得比较少，输出重定向用得比较多

搜索文件内容

grep(globally search for a regular expression)命令可以用于搜索文件内容，它的功能非常强大！grep 命令会按正则表达式去搜索文件，如果文件中某一行匹配指定的正则表达式，grep命令则会显示这一行。

```
$ man grep
    grep - print lines matching a pattern
格式:
    grep [选项] pattern [file]...
常见选项:
    -E, --extended-regexp
        使用扩展的正则表达式
    -i, --ignore-case
        忽略大小写
    -v, --invert-match
        显示不匹配正则表达式的行
    -n, --line-number
        显示行号
    -c, --count
        不显示匹配的行，显示匹配行的个数
```

常见用法:

```
$ grep -nE "firmament" The_Holy_Bible.txt       # 显示The_Holy_Bible.txt中包含"firmament"的所有行，并显示行号
$ grep -cvE "firmament" The_Holy_Bible.txt      # 统计The_Holy_Bible.txt中不包含"firmament"的行数
```

正则表达式

正则表达式语法有点复杂，而且不同的工具使用的语法还不尽相同。下面我们介绍一些常用的正则表达式语法规则:

```
1. 基本单位
    基本单位主要包含：字符、转义字符、.(匹配任意一个字符)、集合(比如，[abc], [^abc])、(expr)
2. 基本操作
    操作的对象是基本单位，主要包含两个基本操作：连接和重复
    a. 连接："ab", "[abc]x", ".txt", "\.txt"
    b. 重复
        +：重复至少一次(>=1)，比如："abc+", "[abc]+"
        ?：重复零次或一次(0|1)，比如："abc?", "[abc]?"
        *：重复任意次数(>=0)，比如："abc*", "[abc]*", ".*"
        {m}：重复m次
```

```
        {m,n}：重复m到n次([m,n])
        {n, }：至少重复n次([>=n])
3．指定基本单位出现的位置
    ^：行首，比如："^abc"
    $：行尾，比如："xyz$"
    \<：词首
    \>：词尾
```

Q：以f开头，以t结尾的单词的正则表达式如何写？（单词之间以空格分隔）

命令的组合

grep 命令之所以强大，主要在于它可以和其它命令组合使用。命令的组合主要有以下三种方式：

```
1．cmd1 ; cmd2
    先执行cmd1，再执行cmd2。如：mkdir dir; cd dir
2．cmd1 | cmd2 (管道)
    将上一个命令的输出作为下一个命令的输入。如：history | tail -n 20
3．cmd1 | xargs cmd2
    xargs(extended arguments) 可以将标准输入里面的每一行转换成命令的参数
    xargs往往和管道一起使用，如：cmd1 | xargs cmd2，它把cmd1输出的每一行作为cmd2的参数
    find . -name "*.c" | xargs grep -nE "\<main\("
```

改变文件权限

chmod 命令可以改变文件和目录的权限。在讲 chmod 之前，我们有必要给大家讲讲目录相关知识。

目录：本质就是一个文件。 这个文件的内容是一个一个目录项(directory entry)，在逻辑上，目录项之间是用链表的方式链接起来的。

权限对于普通文件和目录文件而言，其含义是不太一样的：

```
普通文件
    r：可读
    w：可写
    x：可执行（可执行程序和脚本文件需要x权限才能直接运行）
目录
    r：可读（ls）
    w：可写（在目录下添加和删除文件）
    x：可执行（cd，读写权限依赖于x权限，所以目录一般都设置了x权限）
```

chmod 目录的用法如下：

```
1. 文字设定法（较少使用）
    chmod [ugoa][+=-][rwx] file/dir
        u: user
        g: group
        o: other
        a: all
        +: 添加权限
        -: 删除权限
        =: 将权限设置为
2. 数字设定法（常用）
    chmod mode file/dir
        mode：三位八进制数字
```

常见用法：

```
$ chmod o+w text1
$ chmod u=rwx,g=rx,o=r text1
$ chmod 664 text1
```

文件创建掩码

文件和目录在创建的时候都有一个默认的权限，该权限是由文件创建掩码 umask 决定的。

我们可以用 umask 命令查看当前的文件创建掩码：

```
$ umask
  0002      # 当前的文件创建掩码为2
```

接下来，我们创建文件 text 和目录 dir，查看它们的权限：

```
$ ls -l
drwxrwxr-x  2 he he 4096 3月  12 20:55 dir/
-rw-rw-r--  1 he he    0 3月  13 20:34 text
```

可见当 umask = 0002 时，目录的默认权限为 775，文件的默认权限为 664

将文件创建掩码 umask 设置为 0023

```
$ umask 0023
```

再创建文件 text1 和目录 dir1，查看它们的权限

```
$ ls -l
drwxr-xr-- 2 he he 4096 3月  13 20:40 dir1
-rw-r--r-- 1 he he    0 3月  13 20:40 text1
```

可见当 umask = 0023 时，目录的默认权限为 754，文件的默认权限为 644（643-023）

So，文件创建掩码是如何工作的？请验证你的猜想(*^_^*)

```
目录：777 & (~umask)
文件：666 & (~umask)
```

链接

我们可以用 ln 命令创建硬链接和符号连接(软链接)。

创建硬链接：

```
$ ln text1 h_link
$ ls -li
268648 -rw-r--r-- 2 he he    0 3月  13 20:40 h_link
268648 -rw-r--r-- 2 he he    0 3月  13 20:40 text1
```

可见 text1 和 h_link 是指向同一个文件的，并且硬链接数也由原来的 1 变成了 2。删除 text1 或 h_link 只会减少硬链接数，当硬链接数为 0 的时候，才会真正删除磁盘上的文件。

```
$ rm text1
$ ls -li
268648 -rw-r--r-- 1 he he    0 3月  13 20:40 h_link          # 硬链接数变为了 1
$ rm h_link
```

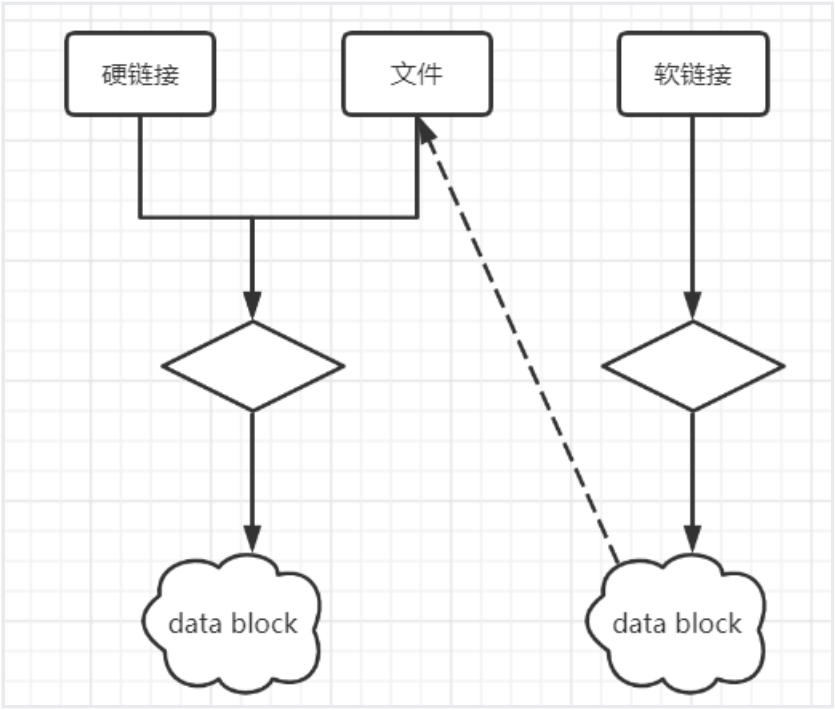
创建符号链接

```
$ ln -s text s_link
$ ls -li
268648 lrwxrwxrwx 1 he he 4 3月  13 21:24 s_link -> text     # s_link文件的内容为text，因此大小为4
264859 -rw-rw-r-- 1 he he 0 3月  13 20:34 text
```

可见 text 和 s_link 是指向不同文件的，text 的硬链接数不会发生改变。删除 text 会导致符号链接 s_link 悬空：

```
$ rm text
$ ls -li
268648 lrwxrwxrwx 1 he he 4 3月  13 21:24 s_link -> text      # 颜色会发生改变
```

硬链接和符号链接的原理如下图所示：



我们可以将符号链接和 Windows 系统上的快捷方式，或者 C 语言中的指针做类比。

对于大多数命令(rm除外)，如果参数是符号链接，其实操作的是符号链接指向的文件(类似指针的解引用操作)：

```
$ echo "I love xixi\n" > text
$ cat s_link
I love xixi
$ rm s_link      # 删除符号链接
```

3 其它命令

远程复制

scp 命令可以用于上传和下载文件：

```
上传：将本地的文件复制到远程
下载：将远程的文件复制到本地
```

scp 的用法和 cp 非常类似：

```
$ man scp
    scp -- secure copy (remote file copy program)
格式:
    scp [选项] SRC... DEST
    本地路径: 可以用绝对路径, 也可以用相对路径
    远程路径: 用户名@IP:路径
常用选项:
    -r
        递归复制整个目录
```

常见用法:

```
$ scp he@IP:~/text1 .    # 将he用户家目录下的text1文件下载到当前工作目录
$ scp ./file he@IP:~     # 将当前工作目录下的file文件上传到he用户的家目录下
```

打包压缩

我们可以用 tar 命令打包和压缩文件。tar 是一个非常有历史的工具，这里我们只介绍它的传统(经典)用法:

```
格式:
    tar [主选项+辅选项] 包名 [文件或目录]...
    主选项(有且只能选择其中一个):
        c: 创建
        r: 追加
        x: 释放
    辅选项:
        f: 指定包文件的名称
        v: 显示详细信息
        z: 使用gzip算法压缩或解压缩包文件
```

常见用法:

```
$ tar cvf package.tar text*
$ tar rvf package.tar The_Holy_Bible.txt
$ tar xvf package.tar
$ tar czvf package.tar.gz *
$ tar xzvf package.tar.gz
```

别名

有些命令比较长，自然我们就想给这些命令起一个别名，方便以后使用，alias 命令就是用来做这事情的。

```
$ man alias
    alias - define or display aliases
格式:
    alias [命令=别名]...
```

常见用法:

```
$ alias                 # 查看别名
alias g++11='g++ -std=c++11'
alias ll='ls -alF'
alias ls='ls --color=auto'
...
$ alias h='history'     # 设置别名
```