轻松学习 Linux

第二回 Linux 文件管理

1）目录树

2）什么是路径？

3）绝对路径与相对路径

4）文件与目录相关操作

5）查看文件内容

6）用户和用户组的概念

7）Linux 文件属性

8）更改文件或目录的属性与权限

9）默认权限

10）特殊的权限

11）文件的特殊属性

12）Linux 文件搜索

13）Linux 文件种类与后缀名

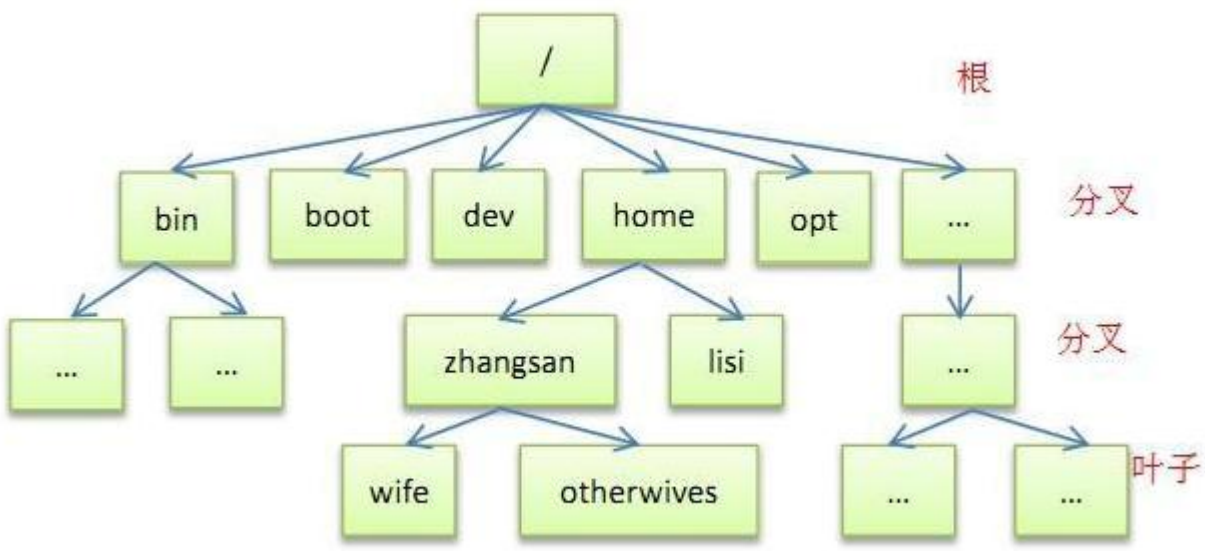
14）Linux 的链接文件

15）附录

本回导语：

Linux 下所有的文件其实都是亲戚关系，换句话讲所有文件都属于同一家族，因为它们在逻辑关系上是在同一棵树上。

1、目录树



2、什么是路径？

路径是系统中某个文件或目录（目录也是一种特殊的文件）在目录树中所在位置的表示形式，即在目录树中的位置！

路径有什么用？这就好比问你家的地址有什么用一样 naive！

3、绝对路径与相对路径

- 路径有绝对路径与相对路径之分：
- 绝对路径：
  - 绝对路径由 根目录“/”写起，例如　/usr/local/apache2/bin/apachectl

- 就好象现实生活中的完整地址：

  /宇宙/银河系/太阳系/地球/亚洲/中国/上海

- 相对路径：

  相对路径是相对于当前的所在的路径来写的

  比如当前在/usr/目录下 我们想进入到 local 目录下 那么路径可以直接写 local

- 命令

  1）cd

  作用：切换目录

  特殊目录：

  - . 当前目录
  - .. 上一层目录
  - - 前一个工作目录
  - ~ 当前登录用户的家目录
  - ~账户名 用户“账户名”的家目录

  注：所有目录下都有两个特殊的目录即“.”与“..”

  2）pwd

  作用：查看当前工作的目录

  3）basename

  作用：获取路径的文件名部分

  4）dirname

  作用：获取路径的目录部分

4、文件与目录相关操作

1）关于文件的时间概念

- mtime：modification time，文件内容修改时间，当文件内容被修改时，会更新这个时间
- ctime：change time，文件更改时间，文件的状态（比如文件的权限或属性）、内容 改变时，会更新这个时间

  即每当有关文件的任何更改（除访问时间外），这个时间都会更新
- atime：access time，文件的内容被读取时，会更新这个时间

2）查看文件与目录

ls [-adl...] [file]...

选项：

- -l：列出详细的信息
- -a：显示全部文件（包括隐藏的文件、Linux 下隐藏的文件是以.开头）
- -d：将目录名像其它文件一样列出，而不是列出它里面的内容
- -h：以适当的格式显示 sizes
- --full-time：列出完整的日期与时间，非常的精确
- --time=WORD
  - 如果配合-l 使用，将时间显示为 WORD 而不是默认 modification time

    WORD 可能为：
    - atime 或 access 或 use 同选项-u

- ctime 或 status 同选项-c
- 如果配合--sort = time 使用，也会使用这边指定的时间作为排序关键字

- --sort=WORD

  按 WORD 而不是名称排序

  WORD 可能为：

  - none (同选项-U)

    不要排序，按目录顺序列出条目

  - size (同选项-S)

    按文件大小排序

  - time (同选项-t)

    按时间排序

  - version (同选项-v)

    根据版本号（自然数）排序

    举例：

    touch 1 11 12 2 3 4 33 100

    touch 1.2.100.4 1.2.3.4 10.1.2.3 9.1.2.3

  - extension (同选项-X)

    按扩展名的字母顺序排序

    举例：

    touch sunshengli.php sifangku.com esc.xyz tefannao.com

- -S

  按文件大小排序

- -t

  按时间(默认按 modification time)排序，最新优先

- -c：

  - 搭配-l 使用时

    显示 ctime 并按名称排序

  - 搭配-lt 使用时

    显示 ctime 且按照 ctime 排序

  - -u

    - 搭配-l 使用时

      显示 access time 并按名称排序

    - 搭配-lt 使用时

      显示 access time 且按照 access time 排序

3）查看目录所占容量

du [-ahcs] 文件或目录名

选项：

-a：将文件的容量也显示出来，而不是仅仅列出目录

-h：以较易阅读的单位来显示容量（极常用）

-c：最后列出总量

-s：只列出总量（极常用）

注：这个命令在“磁盘管理”那一章也会讲，这边留个印象。

4）创建目录

mkdir [-mp] 目录名称...

选项：

-m：指定创建目录的权限

-p：递归创建

注：目录名称有空格的情况，请使用引号！（**强烈不建议 Linux 下文件或目录名称包含空格**）

5）删除空目录

rmdir [-p] 目录名称...

选项：

-p：连同上层空目录一起删除

6）复制

cp [-ripda] 源文件或目录 目标文件或目录

选项：

-r：递归复制（用于复制目录）

-i：遇到目标位置有同名文件时询问是否覆盖

RHEL/CentOS 中 cp 命令其实就是 cp -i 的别名，即默认就加上了-i 的功能，主要是为了安全！

-p：连同文件的属性一起复制

-d：如果源文件为连接文件，则复制链接文件属性而非复制文件本身

-a：等同-rpd

7）移动文件或目录、重命名

mv [-fiu] 源文件或目录... 目标文件或目录

选项：

-f：如果目标文件已经存在则不询问直接覆盖，强制的意思

-i：遇到目标位置有同名文件时询问是否覆盖

-u：若目标文件已经存在，源文件较新则更新

-t：指定目标目录

例如：mv -t 目标目录 源文件或目录...

情景分析：

- 目标文件是目录但对应目录不存在

  源目录会被重命名为目标名称

- 目标文件是目录且目录存在

  源文件或目录会移动到目标目录下

- 目标文件是文件且该文件不存在

  源文件重命名为目标文件名

- 目标文件是文件但该文件已存在

  询问是否覆盖

8）删除文件或目录

rm [-rfi] 文件或目录...

选项：

-r：递归删除（用于删除目录）

-f：强制删除、不询问是否删除、哪怕文件或目录不存在也不会报错

-i：询问是否删除（默认就是这样的）

注意：这个命令非常危险，千万别 rm -rf / 这下子你的整个系统就 over 了

类似于这种顺序也可以 rm /backup/etc/* -rf，其他命令也是同理

9）修改文件的时间戳记录或创建文件

touch [-acdmt] 文件...

默认修改 atime 和 mtime 为当前时间，当然 ctime 也会理所当然的被更新

选项：

-a：修改访问时间（**注：通过 touch -a 去修改访问时间也会导致 ctime 被更改**），说白了加上-a 则不会去修改 mtime

-m：仅修改 mtime

-d：设置修改成的时间等同--date='日期或时间'

例如：touch -d '2018-03-21 12:12:12' passwd

-t：设置修改成的时间，格式[[CC]YY]MMDDhhmm[.SS]

CC 为年份前两位

YY 为年份后两位

SS 为秒数

-c：仅修改文件的时间（若文件不存在不会创建新文件）

5、查看文件内容

1）一次性显示文件的内容

cat [-nA] 文件...

选项：

-n：显示行号

-A：显示特殊字符

2）倒着显示文件内容

tac...

3）head [-数字] 文件...

显示文件前 10 行，或者通过-数字 来显示指定的行数

4）tail [-数字] 文件...

显示文件后 10 行，或者通过-数字 来显示指定的行数

6）查看文件内容（可翻页）

more 文件...

q　　退出

输入/和关键字进行搜索、按 n 继续搜索

注：使用空格或 Ctrl+f 向下翻屏，翻到最后一个才能使用 Ctrl+D 向上翻屏，所以非常不便！

7）less

增强了 more，这个非常适合查看需要翻页的文档，操作方便！

注：搜索操作，和我们之间讲的 man 命令一样

6、用户和用户组的概念

Linux 是一个多用户、多任务的系统，经常会有很多人登录使用同一台主机，这么多用户如何有效的管理呢？

答案：必须得分组

注：Linux 下每个账户可以有多个用户组的支持哦，相当自由吧！

具体操作以后再学，这边知道这个概念即可。

7、Linux 文件属性

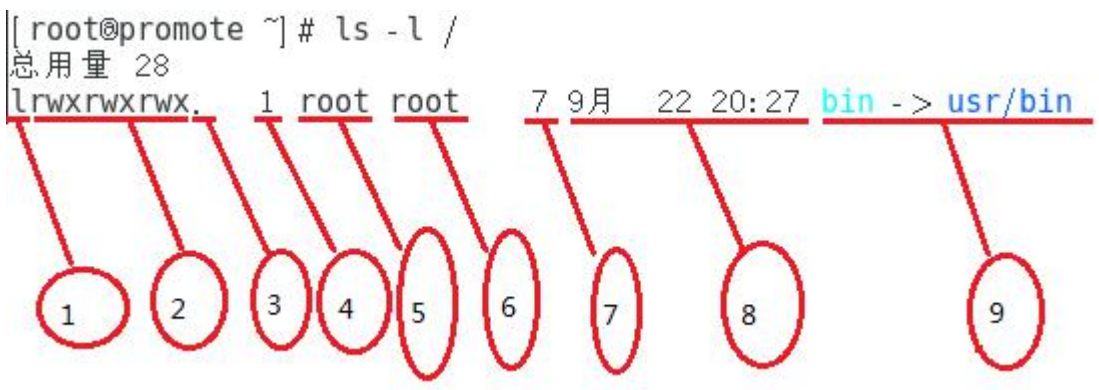
- 概念：
  - 在 Linux 里每个文件，都会有一个所有者和所属组。
  - 所有者：这个文件属于哪一个用户
  - 所属组：这个文件属于哪一个用户组
  - 其他人：除此之外的用户

这样设计的目的无非就是为了更灵活的保证文件的安全，比如一个文件或目录：

- 他的所有者有怎样的权限、
- 所属组有怎样的权限、
- 除此之外的其他人又有怎样的权限

这些都可以灵活的去设置，这样就可方便灵活的保证文件或目录的安全

- ls -l



1）文件类型

d：目录

-：普通文件

l：链接文件

b：块设备

c：串行端口设备文件（比如键盘、鼠标、打印机、tty 终端）

s：套接字文件，用于进程之间通信

扩展阅读：附加阅读 - Linux 一切皆文件该怎么理解？ 见附录 1

2）这边的 9 个字符意义重大

- 每 3 位一组，共 3 组
  - 第一组：文件所有者（user）对该文件的拥有权限
  - 第二组：该文件所属组（group）对该文件拥有的权限
  - 第三组：其他人（others）对该文件的权限

- 每一组是什么意思呢？关键要理解三个字母的意思：
  - r：可读
  - w：代表可写
  - x：可执行

  好那我们找个文件看下 文件所有者、所有者的所在组、其他人各自的权限是什么？

看似容易其实还是需要深究的：

- 权限对于目录文件与普通文件的不同意义：
  - 权限对于普通文件：
    - r：可读此文件内的内容
    - w：可写入或修改文件内的内容（注意不包括删除）
    - x：该文件可以被执行（Linux 下文件比如 shell 脚本是否可以被执行，和后缀名不管和这个权限有关）
  - 权限对于目录文件：

    思考下目录里的内容主要是什么？

    答：文件列表，所以呢？**针对目录的权限主要是对于目录里的文件列表的相关控制**
    - r：读取该目录里文件列表的权限比如通过 ls 显示该目录下的文件
    - w：修改目录内容的权限：
      - 目录内新建文件
      - 目录内删除文件
      - 目录内重命名文件
      - 调整目录内文件或目录的位置（比如剪切）
    - x：用户能够进入该目录成为工作目录

3）过去的 ls（命令其实就是程序，过去这个 ls 程序版本）是没有这个.的

.表示该文件或目录使用了 SElinux 的 context 属性

+表示该文件或目录设置了 ACL 权限

4）表示多少文件名连接到此索引节点（inode），如果是目录则与子目录的数量有关

这个概念一言难尽，后面再说吧！

5）该文件所有者

6）该文件所属组

7）该文件大小单位 B

8）该文件最后一次被修改的时间（mtime）

9）文件名

8、更改文件或目录的属性与权限

注：root 用户是超级管理员是具有特权的人，权限的限制对他是没啥用的哦！

1）改变文件所属组

chgrp [-R] 组名称 目录或文件...

选项：

-R：递归更改（包括目录下面的后代）

2）改变文件所有者

chown [-R] 账号名[:组名] 文件或目录...

选项：

-R：递归更改（包括目录下面的后代）

注：因为这个命令也可以更改组所以 chgrp 用的较少

3）改变文件权限

- 方法一：

  chmod [-R] abc(abc 表示 3 个数字连写) 文件或目录...

  - 选项：

    -R：递归更改（包括目录下面的后代）

  - 参数：

    第一个数字 a：设置 所有者 对文件或目录 的权限

    第二个数字 b：设置 所属用户组 对文件或目录 的权限

    第三个数字 c：设置 其他人 对文件或目录 的权限

    数字怎么得来呢？

  - 三个不同的人（群）对文件或目录可以拥有我们之前讲的 3 个权限
    - r：4
    - w：2
    - x：1

    这个其实是 8 进制数

    - abc 这三个数字 每个数字 都是来自 4、2、1 这三个数字相加得来的（其实有更专业的内部操作，这边这样子说更人性化点）！
      - 比如：

        想设置成这样的权限怎么办？

        rwxr-x---

        7 5 0

- 方法二：
  - 不使用数字，直接用 r、w、x 来设置权限
  - 我们可以用 u、g、o、a 分别表示 user、group、others、all 所有的人

    chmod [-R] u=rwx 文件或目录...

    chmod [-R] u=rwx,g=rx,o=r 文件或目录... 说明：多个用,号分隔

    chmod [-R] a=rwx 文件或目录... 说明：将 all 所有的人 都设置成 rwx

    chmod [-R] ug=rwx,o=r 文件或目录... 说明：如果某类角色权限设置值相同可以直接连写
  - 不管原来的权限，只想更改部分权限

    chmod [-R] g-w 文件或目录... 说明：文件所属组去掉 w 权限

    chmod [-R] g+w 文件或目录... 说明：文件所属组加上 w 权限

    chmod [-R] a+x 文件或目录... 说明：所有的人对该文件都加上 x 权限

    chmod [-R] o+rw 文件或目录... 说明：其他人对该文件都加上 rw 权限

    chmod [-R] go-rw 文件或目录... 说明：所属组和其他人对该文件都去掉 rw 权限

    chmod [-R] g+x,o+rx 文件或目录... 说明：所属组加上 x 权限，其他人加上 rx 权限

9、默认权限

默认情况下 root 用户新建的普通文件权限为 644 ,新建的目录权限为 755 ,这个新建文件或目录的默认权限可以设置吗？

- umask：查询 umask 值 得到 0022，这数字是什么意思？
  - 首先死记硬背两条理论:

    ①系统预设的新建文件的权限为 666（Linux 预设不允许直接创建出具有执行权限的文件）

    ②系统预设的新建目录的权限为 777

  1）我来说一个比较专业的解释：

  这个数字是掩码，第一个数字（如果提供则）必须是 0， 因为这是一个 8 进制数，所以我们重点看后三位 022

  - 这三位数用于与 预设的 文件的权限（666）或 目录的权限（777）进行位运算，来屏蔽对应的位

    (~$remove) & $power

    022　　取反　　666　　**&后**结果 644

    0　000　111　　6　110　　110

    2　010　101　　6　110　　100

    2　010　101　　6　110　　100

    022　　取反　　777　　**&后**结果 755

    0　000　111　　7　111　　111

    2　010　101　　7　111　　101

    2　010　101　　7　111　　101

    这是比较专业的解释！

  2）上面这种解释比较专业，我再说一个更直观点的易于理解记忆的解释：

  - 说白了 022 就是指文件或目录的 所有者、所属组、其他人 在预设的权限的基础上要剔除的权限。

    022

    --- -w- -w-

    666

    rw- rw- rw-

    rw- r-- r--

    777

    rwx rwx rwx

    rwx r-x r-x

  - 注：这边的思维是剔除**对应权限**，而**不是**直接**减去**对应的数值然后再看数值对应的权限！

    比如：666-003 如果直接减，而不是剔除 3 对应的权限，则结果会和预期完全不符！

- umask -S：直观显示各角色拥有的权限（而不是上面这种用于剔除的 8 进制数）
- 设置 umask 值：

  umask 002

注：该设置方法仅仅是针对当前用户当前所在的终端，且重启会丢失，即是临时设置！

- 有没有办法为用户各自指定不同的 umask 值，且所有终端都有效，重启也不丢失呢？

  当然可以啦！既然你能想到这个需求，那么 Linux 是肯定有这个功能的，不过这个涉及到以后的知识，后面再和大家侃！

10、特殊的权限

文件的权限其实不只 r、w、x 这三种，之前为了考虑更容易被大家接受所以我们没有讲其他的特殊权限。

现在我们有些基础了，可以和大家讲了：

- 除了 r、w、x 之外 还有 s、t 这两个特殊权限，它们与系统的账户和系统的进程相关
  - s 这个标记可以出现在 **文件拥有者** 或者 **文件所属组** 的的 **x 权限**位置上

    1）出现在 **文件拥有者** 的 x 权限位置上 称为 Set UID，简称 SUID 比如：rwsrwxrwx

    SUID 权限具有这样的功能（注：SUID 仅对二进制程序有效）：

    ①执行者若能执行该程序则在且仅在该程序的执行过程中将具有该程序所有者的权限

    设置方法：

    u+s

  2）出现在 **文件所属组** 的 x 权限位置上 称为 Set GID，简称 SGID 比如：rwxrwsrwx

  SGID 权限具有这样的功能（注：SGID 对二进制程序或目录有效）：

  ①对于二进制程序：

  执行者在且仅在该程序的执行过程中将具有该程序所属用户组的权限

  ②对于目录：

  用户在若能进入此目录下则在此目录之后有效用户组将变成该目录的所属用户组

  用户名若能在此目录下创建新的文件则文件的用户组将与此目录的用户所属组相同

  设置方法：

  g+s

- t 这个标记可以出现在 其他人 的 x 权限位置上比如：rwxrwxrwt，这时我们称为 Sticky Bit 简称 SBIT，仅对目录有效

  作用：

  用户若能对此目录拥有 w 和 x 权限

  则 用户在此目录下创建的文件或目录仅自己或 root 才能够有权利删除这个新建的文件或目录

  设置方法：

  o+t

PS：有时候 s 或者 t 会是大写的！这是为啥呢？

如果未给予可执行权限，即使设置了特殊属性，也将会是空的，会用大写的 S 和 T 表示

PS：也可以用数字来说设置 SUID、SGID、SBIT 权限，分别对应数字 4、2、1

在设置时将数字写在 chmod 设置权限时 3 个数字的前面即 4 个数字连写即可!

比如你想设置 SUID 权限：

chmod 4xyz 文件

扩展：其实 Linux 为了防止一些特殊的权限配置需求，还提供了 ACL 权限控制机制，他可以提供更加细致的权限设置，比如可以针对个文件单独设置某个用户（不是文件所有者）具有怎样的权限！ACL 设置是非常简单的，不过这个

功能不常用，所以后面挑个时间再和大家说吧！

11、文件的特殊属性

除了上面所讲各种文件属性之外还有特殊的属性我们要说下：

- chattr [-R] [+-=] [Aaci] 文件或目录

  -R：递归设置

  +-=：

  +：增加后面指定的属性

  -：去掉后面指定的属性

  =：直接设置成后面指定的属性

  a：增加该属性后文件只能追加数据，（甚至 root 都）不能删除也不能修改，且只有 root 可设置该属性

  i：增加该属性后文件（甚至 root 都）不能被删除、重命名、设定连接、写入、新增数据，且只有 root 可设置该属性

  注：目录也可以设置这个属性，试想目录的内容不能被删除，写入代表什么呢？和前面的权限对于目录同理！

  A：增加该属性后文件或目录的 atime 将不可修改（节省计算机资源）

  c：增加该属性后自动压缩该文件，读取时会自动解压

  等其他不常用参数

  注：并不是所有的文件系统都支持这个命令的所有参数，所以使用的时候如果提示不支持也不要诧异！

- 查看文件或目录的特殊权限

  lsattr [-adR] 文件或目录

  -a：类似于 ls 的-a 参数

  -d：类似于 ls 的-d 参数

  -R：连同子目录的数据一同列出

12、Linux 文件搜索

1）which

作用：查找并显示指定命令的绝对路径

- 其实命令就是可执行程序，那么这些命令放在系统中的什么位置呢？
- 为什么我们直接敲命令名字而不用写全路径就能执行这个命令程序呢？

  这和 **PATH 环境变量**有关

- 环境变量 PATH

  为啥：直接敲 rm 命令可以执行下面这个程序，而不用写完整的路径呢？比如：

  /usr/bin/mkdir

  这个就是环境变量 PATH 的作用了

  echo $PATH

  具体的我们在后面在做介绍，这边知道这么多就 OK 了

  注：环境变量 PATH 中保存了查找命令时需要遍历的目录，即 which 只能用来查找 PATH 环境变量中出现的路径下的可执行文件！

2）whereis [-bms] 文件名称

作用：查找命令的二进制程序、源代码文件和 man 手册页等相关文件的路径

查找位置：尝试在标准 Linux 位置以及$ PATH 和$ MANPATH 指定的位置找到所需的程序，这个我们不用关心

-b：只查找二进制文件

-m：只查找帮助文件

-s：只查找源代码文件

3）locate [-ir] 关键字

作用：按名称查找文件（在预先生成的文件列表数据库/var/lib/mlocate/mlocate.db 中查找，速度很快）

-i：忽略大小写

-r：后面可接正则表达式

通过 updatedb 命令生成或更新文件列表数据库

注意：

- 系统刚装好之后，就是用 locate 进行搜索可能会提示/var/lib/mlocate/mlocate.db 不存在，这时只要运行 updatedb 生成这个文件列表数据库即可
- 数据库文件并不是实时更新（通常每天或每周更新一次），因此在用 locate 时，有时会找到已经被删除的文件，或者刚刚建立文件，却无法查找到，原因就是因为数据库文件没有被更新
- /etc/updatedb.conf 文件配置了文件列表数据库生成或更新的规则，比如哪天你希望更新文件列表数据库时忽略某个目录，就可以在这个文件里配置即可，具体配置非常简单，需要更改的时候自己通过大部分的搜索引擎搜索就能知道这个配置文件里各个配置项的作用以及配置方法。

4）find

find 是一个极其强大且常用的文件搜索命令，请大家一定要掌握它的常用方式！

- find [-P] [-L] [path...] [expression]
  - **常用选项：**
    - -P

      这是默认行为。当查找或打印信息文件时，文件是符号链接时，使用的信息应取自符号链接本身的属性。
    - -L

      当查找检查或打印文件信息时，所使用的信息应取自链接指向的文件的属性，而不是来自链接本身（除非它是一个断开的符号链接或查找无法检查文件链接点）。
  - path...

    在指定的路径搜索，可以指定多个用空格隔开！

    注：默认指定路径的后代目录也都会搜索哦！
  - **expression**

    表达式（这边的表达式你可以理解为表达形式，即这个地方可以放各种表现形式的参数来实现对应的功能）

    这边的 expression 由 选项、tests、actions 组成具体形式为：

    expression 格式：[选项...] [tests...] [actions...]
    - 常用选项

      -maxdepth levels

      设置最大目录层级

      例如：

      -maxdepth 1

      则只在指定目录下的子文件进行查询，而非所有后代

      -mindepth levels

      设置最小目录层级

例如：

-mindepth 1

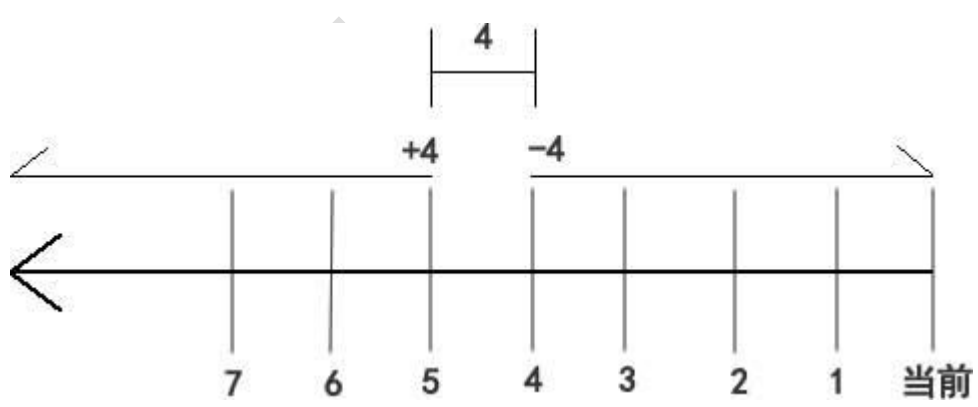
则在指定目录下的第 1 代（假设指定的目录为第零代，包括第一代）以下的目录进行查找

- tests
  - -name pattern
    - 文件名的基础部分（删除了前导目录的路径）与 pattern 匹配。
    - pattern 之所以叫 pattern 而不是叫普通的字符串是因为它支持通配符写法
      - 通配符我们会在后面花专门的时间介绍的！
      - 先简单使用个*试试，*在通配符里代表 0 到无穷个任意字符
      - 注：通配符和正则表达式有点类似，都是能代表其他某类字符串，但是具体作用和正则表达式是有很大差别的，不是同一样东西哦！
      - 大括号｛ 在这边并不被认为是特殊的，尽管包括 Bash 在内的一些 shell 中具有特殊的含义（以后讲到 shell 那边你就知道{有什么特殊性了）。
      - **最好将 pattern 用引号引起来，防止被 shell 理解错误而导致歧义执行**
  - -atime [+-]n
    - -atime n

      查询出 atime 在 n 天之前的 1 天内 的文件
    - -atime +n

      查询出 atime 是在 n 天以前（不含 n 天本身） 的文件，说白了即 n+1 天之前的文件
    - -atime -n

      查询出 atime 是在 n 天以内（含 n 天本身） 的文件
    - **如图：假设 n 为 4**



注：该图的绘制参考了《鸟哥的 Linux 私房菜》，这本书在本课程最后也会推荐给大家！

  - -ctime [+-]n
    - -ctime n
    - -ctime +n
    - -ctime -n
  - -mtime [+-]n
    - -mtime n
    - -mtime +n
    - -mtime -n
  - -type 文件类型

    通过文件类型查询文件

    文件类型：

    f

普通文件

d

目录文件

l

符号链接文件

b

块设备文件

c

字符设备文件

s

socket 文件

p

named pipe (FIFO)文件，命名管道 是 Unix 和类 Unix 系统上传统管道概念的延伸，也是进程间通信（IPC）的一种方法

- -size [+-]size[ckMG]

  根据文件大小搜索

  +：比 size 大

  -：比 size 小

  文件使用 n 个单位的空间。 常见的单位：

  c

  字节

  k

  KB

  M

  MB

  G

  GB

- -perm [+-/]mode
  - -perm -mode

    搜索出完全包含了 mode 权限的文件，说的简单的就是找出权限大于等于 mode 的那些文件

    mode 可以为：

    可以为权限数字如 744 或者 0744

    或者为符号模式如 u=rwx,g=rx,o=r 也可以值指定其中一个如 u=rwx 或 u=x

    注：-perm -000 将匹配任何文件
  - -perm /mode

    搜索出只要包含 mode 中任一权限的文件

    如果模式中没有设置权限位-perm /000，则匹配任何文件(同-perm -000)。
  - -perm mode

    文件的权限位准确模式（八进制或符号）。由于必须完全匹配，所以如果您使用符号模式，你可能

必须指定一个相当复杂的模式字符串。 例如-perm g=w 只会匹配 0020 模式的文件（而非那些包含了 g 有 w 权限的文件，但是也有其他权限的文件）。

- -perm +mode

  不推荐使用模式，此模式已经或即将被废弃，使用/代替。

- -user uname

  用户 uname(可以使用用户 ID)拥有的文件

- -group gname

  所属组为 gname(可以使用组 ID)的文件

- -nouser

  查找无有效 所属者 的文件

  即文件的 所属者 在/etc/passwd 中不存在

- -nogroup

  查找无有效 所属组 的文件

  即文件的 所属组 在/etc/groups 中不存在

- -newer file

  查找比 file 还要新的文件

  -newer file1! file2

  查找比 file1 还要新 但比文件 file2 旧的文件

- actions

  -exec 命令

  将 find 搜索出来的结果交给后面的命令处理

  命令格式为：

  -exec command {} \;

  - find 的搜索结果会被放到{}处
  - 命令直到\;表示结束（;在这边表示结束，但是它在 bash 中特殊含义所以需要加\进行转义）

  举例：

  find / -name sun -exec ls -dlh {} \;

  -print0

  如果找到文件则在标准输出打印文件全名，然后接一个 null 字符，而不是换行

  PS：空字符确实存在，只是它是非打印字符你看不见

  什么用呢？后面讲到 xargs 再说！

注：动作有很多，但是我们后面会讲 xargs 命令，所以这边的动作了解即可！

注：这些各种选项或参数是可以按需多个配合起来一起使用!

find 命令功能强大，使用上其实并不简单，大家可以先把上面这些学会，再有特殊需求时可自行 man 查询！

有的时候用 find 搜索到/proc 目录里的一些文件时会出现类似于以下提示：

```
find: ‘/proc/23617/task/23617/fd/6’: No such file or directory
find: ‘/proc/23617/task/23617/fdinfo/6’: No such file or directory
find: ‘/proc/23617/fd/6’: No such file or directory
find: ‘/proc/23617/fdinfo/6’: No such file or directory
```

这是因为：/proc 是内存的映射，下面的文件/proc 实际上并不存在于你的硬盘上，它们里面包含当前正在运

行的进程信息（文件形式），出现这些错误的原因是某些进程在 find 运行时已经退出了，所以对应的文件就不存在了，所以报这个提示完全正常。

13、Linux 文件种类与后缀名

- Linux 中一切皆文件，都是文件也总归有不同的地方吧？

  当然有，有些文件是表示设备的，有些文件就是普通的用来存放文本的等等！

- **我们来看下 Linux 中常见文件种类**

  1）普通文件

  ls -l 第一个字符是“-”的文件就是普通文件

  普通文件分为纯文本文件和二进制文件

  2）目录

  目录其实也是特殊的文件哦

  3）链接文件

  ls -l 第一个属性为 l 的文件就是链接文件，类似于 windows 下快捷方式

  4）设备

  系统的设备文件都集中在/dev 目录下，ls -l 第一个属性为 b 或者 c 的

  为 b 的一般是可供存储的接口设备

  为 c 的一般是串行端口的接口设备比如键盘、鼠标等

  - 查看文件类型：file

- **Linux 文件后缀**

  在 Linux 系统中，文件的后缀名是没有特殊意义的，有没有都无所谓，这可不像 windows 中比如.exe 的就是可执行文件！而 Linux 下文件是否可执行，主要看这个文件是否具有执行权限，Linux 中我们给文件加上后缀名主要是为了利于我们肉眼上区分文件功能或内容，他的后缀名对于系统来说是没有其他任何特殊功能。

14、Linux 的链接文件

Linux 链接文件分为 硬链接（hard link）和软链接（symbolic link）两种，既然分为两种那他们肯定是有区别的咯。

1）软链接

软连接你把它看成是 windows 下的快捷方式即可！

ln -s 源文件 目的文件

2）硬链接

ln 源文件 目的文件

硬链接是啥？我们首先得了解下 inode 这个概念

- 文件在存储时会将“文件的实际内容”与“各种文件属性”这两部分数据分别存放在不同块中

  一个文件的“实际内容”被放在 data block 中，一个 block 不够存放时会占用多个 block。“这个文件的各种属性”被放在 inode 中，一个文件占用一个 inode，inode 中记录着文件数据所存放的 block 号。

- 硬链接相当于是 不同的文件名 引用了（连接、指向） 相同的 inode

  通俗点的理解办法是，硬链接相当于给一个文件多建立一个别名文件，如果给一个文件建立了硬链接,那个除非把两个文件都删了否则文件依然存在哦，只删除一个相当于删除的就是一个引用。

- 注意：硬链接不能跨分区或者说跨一个文件系统且不能链接目录

- 这下就可以理解 ls 那边第 4 列属性的含义了

15、附录

1）Linux 一切皆文件该怎么理解？

注：以下这段话如果当前实在不能理解，就先放一放，等随着对 Linux 的不断使用以及体会，就能够明白了，但是不管你理解不理解，重要的是你在没理解之前一样到经常把这段话拿出来思考思考！

Linux 系统中的各种资源（不管是普通文件还是目录、磁盘分区、设备等等等）在 Linux 中都被一种“逻辑关系”相关联着，这个逻辑关系就是目录树，目录树只有一个根，即一个祖先，这样它们在系统中就可以用类似于这种方式来表示/sun/sheng/li，所有的资源都可以用/开头的这种形式来表示，这是一种非常科学的标识资源的方式，逻辑清晰且方便（因为可以很容易的去表示它们且很容易的就知道它们在目录树中的关系），所以目录树更像是一个逻辑关系图，这个图正好像一棵树。这棵树上的所有东西我们都叫它”文件 “，不管它实际代表的是普通文件（存放文字等信息），还是非普通文件（比如目录、磁盘分区等等）。

所以：目录树是 Linux 设计者为了方便在操作系统里表示系统的每个资源而设计的一种逻辑关系，至于它们之间的实际关系那是系统内部的处理行为了，和这棵树就关系不大了，比如我们第一回介绍的 磁盘分区 与目录树的关系，磁盘分区是系统的资源，它在系统的目录树这个逻辑关系上，这样我们就可以使用类似于“/dev/sda1”这种方式来表示它，至于这个分区实际上是挂载在哪个目录上就和这棵目录上没有直接关系了，换句话讲就是这个分区实际挂载在哪个目录上被用于存储数据，光从这棵逻辑关系树上就看不出来了，因为它挂载在哪个目录上和这个逻辑关系树无关，Linux 设计者让它在这个目录关系树上使得我们使用者可以通过/dev/sda1 这种方式去表示它，这样 Linux 系统上的各种资源就都有自己的表示方式且逻辑关系非常明确、毫不混乱，使用者可以有条不紊的去标识它们。Linux 设计者也可以利用这个逻辑关系去合理灵活的分配资源（比如磁盘挂载到目录上就是个很好的例子，哪些目录用这个磁盘来存储数据透过目录树就能有逻辑的去分配了）。

所以：Linux 设计者不管是为了自己在设计逻辑上方便处理系统的所有资源，还是为了让使用者方便表示它们，让它们都在逻辑上建立了目录树关系。因为目录树这个逻辑关系是我们使用者在面对 Linux 操作系统时肉眼就会看到的东西，这个关系网中的每个节点在 Linux 里表现出来的样子就是每个文件。

总结：站在我们使用者的角度所有的文件在 Linux 中就是一种科学的表示方式，我们可以轻松的通过/开头的表示方式来表示它，我们只要学会透过这个目录树关系去看待它就行了而不是纠结它在操作系统内部实际上是被操作系统怎么看待以及处理的实现细节。