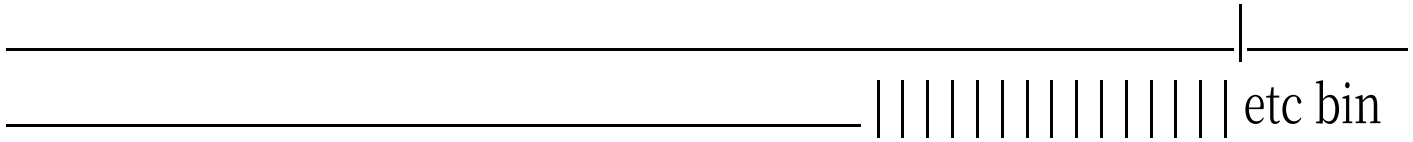
菜鸟必读：RHCE课堂学习笔记(2) PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/203/2021_2022__E8_8F_9C_E9_B8_9F_E5_BF_85_E8_c103_203839.htm 单元五：文件访问许可 所有文件都有一个拥有者(owned by a user)，并和一个组(group)相连。 因此一个用户是否有权限读写或者执行一个文件，是由这个文件是否被赋予了相应的权限所决定的。权限可以设定给文件拥有者，文件所在的组，或者其他任何人。 可由ls -l 命令来查看文件权限(permissions)： $ ls -l /bin/login -rwxr-xr-x 1 root root 19080 Apr 1 18:26 /bin/login 可以看到文件的访问权限由10个字符表示。 文件访问权限为三种用户种类使用。每个种类都有一个表示字符： u 文件的拥有者(owner) g 文件所在组的其他用户 o 任何用户(others) 每个种类的访问权限都彼此独立，互不相关。 三种标准文件访问类型： r 文件的读权限/罗列目录内容的权限(list a directorys contents) w 文件的写权限/在目录中建立或删除文件的权限 x 文件的执行权限/访问目录中文件的权限，例如cd 到该目录 此三种标准文件访问类型可以赋给上述的三种文件访问权限的用户种类，即u、g、o。 文件访问权限中，第2、3、4个字符表示了文件拥有者的权限；第5、6、7个字符表示了文件组的权限；第8，9，10个字符表示了其他用户的权限。例如： $ ls -l /bin/login -rwxr-xr-x 1 root root 19080 Apr 1 18:26 /bin/login 说明了该文件的拥有者可以读写并执行该文件，其他的用户（包括组内用户）可以读、执行该文件。又例： $ ls -l README -rw-rw-r-- 1 andersen visitor 2948 Oct 11 14:07 README 该文件可以由visitor组内的用户读写，但是并不能执行；能被其他用

户读，但是其他用户不能改写它或者执行它。 文件访问权限中的第一个字符"d"将目录和其他文件予以区分： $ ls -ld /bin drwxr-xr-x 2 root root 4096 Apr 20 18:13 /bin/ 更多例子： 用户fred 是组fred 和组staff 的成员 用户mary 是组mary 和组admin, staff 的成员 文件fileA 拥有者是fred，拥有者组是 fred 文件fileB 拥有者是mary，拥有者组是 root 文件fileC 拥有者是root，拥有者组是 staff 给出下表 ---------- user fileA fileB fileC ---------- fred u,g,o o, o,g mary o u,o o,g root u,g,o u,g,o u,g,o 如果fileA 有访问权限rwxr-xr--，那么它访问权限如下： read write excute ---------- fred yes yes yes mary yes no no root yes yes yes 改变文件权限用chmod 命令，chmod 后跟一个表达式，表达式可以为一串数字或者一段预置的代码组合：改变谁的(who)，操作符(operator)和权限(permission)。 改变谁的(who)可以有下面的选择： u 文件拥有者 g 在文件组内的用户 o 其他用户 a 所有用户 操作符(operator)可以有下面的选择： 增加权限 - 删除权限 = 将权限赋予… 权限(permission)可以有下面的选择： r 读 w 写 x 执行（对于目录来说是访问） s Set userID bit（第四位）或者set groupID bit（第七位） t Set sticky bit（对于目录来说便是防止其他非拥有者删除目录中的文件，位于第10位） 例如： $ chmod g w .bash_profile 该命令将写权限赋予了文件组内的用户。 $ chmod go-rw .bash_profile 该命令剥夺了非拥有者用户的读写权限。 chmod 中有一个有用的参数为-R （递归，注意大写），可以将整个目录中的文件和子目录的权限全部改写。 前面提到还有一种方法可以修

改文件权限，就是数字方式。以三个数字的方式确定文件的访问权限。第一个数字代表拥有者权限、第二个代表文件所在组内的用户权限、第三个代表其他用户权限。权限表达式由以下数字相加而得：4（读权限）2（写权限）1（执行权限） 例：设置文件file 为所有人均为只读权限：$ chmod 444 file 例：设置文件file 为拥有者拥有读写和执行权限，组内用户有读和执行权限、其他用户无任何权限：$ chmod 750 file 还要提一下默认的文件权限。默认的文件权限，是由umask 来决定的。非特权用户的umask 为002，即文件默认权限为664；而root 用户的umask 为022，即文件默认权限为644。 如果没有umask 的话，所有文件都会默认为666权限，意味着所有人都可以读写新建的文件。注意所有新建的文件都没有执行权限，即使umask 也无能为力。所有的执行文件都要显式的给出执行权限才能运行。但是对于目录来说，无论umask 的值是多少，新建目录时就默认赋予了执行权限（可以访问目录）。 要改变umask，只需打：$ umask 022 这样就使得原本默认的002 umask值改为022。但是当下次再登录的时候，umask 又会改回到原来的值，这就需要你在bash 的初始化脚本(initialization script)中加入特定的umask 值。 除了三种标准访问权限外，还有前面提到过的三种特殊权限。它们是setuid、setgid 和sticky 位。（下面我的理解不知道正确与否，希望高手指点！） setuid 位将进程的用户ID (user ID)设置为文件的用户ID，对目录无效。 setuid 是一个非常强大也很危险的工具。比如，如果一个程序设置了setuid 位而且它的拥有者是root，那么当该程序执行时它就拥有了root 用户执行这个程序的特权。有些程序必须使用它来使程序正确运行。比

如ping 程序必须设置setuid 为root，因为它要在网络上传输ICMP包的裸数据。任何setuid 的程序必须小心编写并排除安全漏洞。 setgid 位将进程的组ID (group ID)设置为文件的组ID，对目录来说，它迫使所有在该目录中创建的文件都拥有与该目录相同的组，而无论文件的创建者是谁。 setgid 和setuid 一样，也很强大，应该小心使用。因为它允许无意识的对文件和资源的访问。举例来说，minicom 终端模拟器程序设置setgid 为uucp 组，它提供了向计算机的串行口访问的权限（组uucp 拥有该种权限）。 sticky 位作用在目录上，防止用户删除它们并不拥有的文件(files they do not own)。sticky 位典型的应用便是在/tmp 目录中，防止用户删除彼此的文件。sticky 位在文件上没有效果。 单元六：Linux文件系统 文件和目录被组织在一个单根(single-rooted)反向树状结构中。包括独立的物理设备卷，比如软盘、CDROM或多个硬盘。反向树状结构的基点（最高点）为根目录或者叫"/"。 文件名是大小写敏感的，"/"符号为界定符，分开路径名的各个元素。例如 /usr/bin/X11/X 每个shell 和系统进程都有一个“ 当前目录 ”。".."表示当前目录的上一级目录，"."表示当前目录。文件或者目录以"."开头的是所谓的隐藏文件，它们在默认的罗列中不出现。 一个用户的路径(path)是一串目录，用以搜索执行命令的程序所在。 /(root)



etc bin sbin lib root usr mnt var boot tmp opt dev home proc lost found bin ： 用来储存用户命令。/usr/bin 目录也存放用户命令。 sbin ：系统命令例如shutdown 之类的所在。/usr/sbin 目录也存放许

多系统命令。 root： 超级用户的home 目录。 mnt： 通常在系统启动以后含有文件系统装载的装载点(mount point)。 boot ： 含有在系统启动时所用到的内核和其他文件。 lost found： fsck 使用用来存储找到的文件碎片(无文件名的文件)。 lib： 含有许多在/bin 和/sbin 下的程序所用到的库文件。/usr/lib 目录下含有更多的库文件。 dev： 存储设备文件。 etc： 含有许多配置文件和目录。 var： 存储像系统日志和打印池等“ 变量 ”文件。 proc： 一个含有系统进程信息的虚拟文件系统（不存储在磁盘上） tmp： 一个用户和程序的“ 草稿本 ”，/tmp 对所有人和所有进程开放读写权限。 opt： 类似StarOffice 这种第三方包的安装目录。 以下是关于ext2/ext3 文件系统的细节。 当ext2/ext3文件系统建立时，系统的元数据(metadata)就存储在超级块中(superblock)。因为这些数据对于操作文件系统来说至关重要，所以同时也建立了一份文件系统超级块的拷贝。（比方在小的文件系统上每8192个块建立一份拷贝）。dumpe2fs 命令可以查看超级块上的数据。超级块(superblock)含有文件系统的元数据(metadata)：卷名，UID，inode数，块数，保留块数，块组的位置等等。 inode 就像数据在块上的描述。inode 不存储文件的真实数据，而是存储文件的信息。stat 命令可以帮助我们查看文件的inode 从而得知是如何存储在文件系统中的。 $ stat passwd File: "passwd" Size: 1129 Blocks: 8 IO Block: 4096 Regular File Device: 306h/774d Inode: 214057 Links: 1 Access: (0644/-rw-r--r--) Uid: (0/root) Gid: (0/root) Access: Sat Sep 29 13:34:57 2001 Modify: Sun Sep 23 17:12:41 2001 Change: Sun Sep 23 17:12:41 2001 下面是硬连接(hard link)的概念。 硬连接是一个在文件系统中“ 物理存在

”的文件，每个link 指向文件的inode。它使得一个文件拥有两个或以上名字成为可能。注意，用户可以link 他们不拥有的文件，但是还是会被inode 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com