Linux 文件系统概述

内容提要

1. 理解什么是文件系统
2. 熟悉 Linux 文件系统的布局
3. 熟悉主流的日志文件系统
4. 理解非日志文件系统和日志文件系统的不同
5. 掌握使用文件系统的一般方法

什么是文件系统

在Linux系统中，所有的程序、库、系统文件和用户文件都存放在文件系统之中，系统和用户所创建和保存的数据也都在文件系统当中，什么是文件系统呢？

文件系统是操作系统用于明确磁盘分区上的文件的方法和数据结构，即文件在磁盘上的组织方法。换句话说，文件系统规定了如何在存储设备上存储数据以及如何访问存储在设备上的数据。一个文件系统在逻辑上是独立的实体，他能单独地被操作系统管理和使用。

Linux的内核采用了称之为虚拟文件系统（Virtual File System，VFS）的技术，因此Linux可以支持多种不同的文件系统类型。每一种类型的文件系统都提供一个公共的软件接口给 VFS。Linux 文件系统的所有细节由软件进行转换，因而从 Linux 的内核以及在 Linux 中运行的程序来看，所有类型的文件系统都没有差别，Linux 的 VFS 允许用户同时不受干扰地安装和使用多种不同类型的文件系统。所以Linux核心的其它部分及系统中运行的程序将看到统一的文件系统。Linux的虚拟文件系统允许用户同时能透明地安装许多不同的文件系统。虚拟文件系统是为Linux用户提供快速且高效的文件访问服务而设计的。

随着 Linux 的不断发展，它所支持的文件格式系统也在迅速扩充。特别是 Linux 2.4 内核正式推出后，出现了大量新的文件系统，其中包括日志文件系统 ext3、ReiserFS、XFS、JFS和其它文件系统。Linux 系统核心可以支持十多种文件系统类型：JFS、 ReiserFS、ext、ext2、ext3、ISO9660、XFS、Minx、MSDOS、UMSDOS、VFAT、NTFS、HPFS、NFS、SMB、SysV 等。

文件系统布局

文件系统是Linux下的所有文件和目录的集合，这些文件和目录结构是以一个树状的结构来组织的，这个树状结构构成了Linux中的文件系统。也就是说当用户使用Linux文件系统时所面对的就是这个树状结构，在这个结构中包含大量的文件和目录。使用Linux，用户可以设置目录和文件的权限，以便允许或拒绝其他人对其进行访问。 用户可以浏览整个系统，可以进入任何一个已授权进入的目录，访问那里的文件。

Linux 的多级树状目录结构称为文件系统布局。Linux的目录结构遵从1994年制定的 Linux文件系统标准（Linux File System Standard，FSSTND）。在实现FSSTND的同时，又产生了文件系统层次结构标准（File System Hierarchy Standard，FHS）。现在大多数 Linux 遵从 Filesystem Hierarchy Standard [http://www.proxyserve.net/index.php?q=aHR0cDovL3d3dy5wYXRobmFtZS5jb20vZmhzLw%3D%3D] 来安排文件的系统布局。

系统布局包括如下目录分别用于存储不同类型的数据：

目录	说明
/bin	基本系统所需的命令。功能和/usr/bin类似，这个目录中的文件都是可执行的，普通用户都可以使用的命令。
/boot	内核和加载内核所需的文件。一般情况下，GRUB 或 LILO 系统引导管理器也位于这个目录。
/dev	设备文件存储目录，比如终端、磁盘等
/etc	所有的系统配置文件
/home	普通用户家目录的默认存放目录。
/lib	库文件和内核模块所存放的目录
/media	即插即用型存储设备的挂载点自动在这个目录下创建，比如 USB盘、CDROM/DVD自动挂载后，也会在这个目录中创建一个目录。
/mnt	临时文件系统的挂载点目录
/opt	第三方软件的存放目录。
/root	Linux超级权限用户root的家目录。
/sbin	基本的系统维护命令，只能由超级用户使用。
/srv	存放本系统提供的站点服务数据。
/tmp	临时文件目录。
/usr	存放用户使用系统命令和应用程序等信息，比如命令、帮助文件等。
/var	存放经常变动的数据。

用户可以使用如下命令获得Linux文件层次结构的说明:

```
$ man hier
```

CentOS 5 默认的文件系统布局

```
$ tree -d -L 2 /
/
```

```
|-- bin
|-- boot
|   |-- grub
|   `-- lost+found
|-- dev
|   |-- VolGroup00
|   |-- bus
|   |-- disk
|   |-- fd -> /proc/self/fd
|   |-- input
|   |-- mapper
|   |-- net
|   |-- pts
|   |-- shm
|   `-- snd
|-- etc
|   |-- NetworkManager
|   |-- X11
|   |-- acpi
|   |-- alchemist
|   |-- alsa
|   |-- alternatives
|   |-- audit
|   |-- avahi
|   |-- blkid
|   |-- bluetooth
|   |-- bonobo-activation
|   |-- cron.d
|   |-- cron.daily
|   |-- cron.hourly
|   |-- cron.monthly
|   |-- cron.weekly
|   |-- cups
|   |-- dbus-1
|   |-- default
|   |-- depmod.d
|   |-- dev.d
|   |-- firmware
|   |-- fonts
|   |-- gconf
|   |-- gnome-vfs-2.0
|   |-- gtk-2.0
|   |-- hal
|   |-- httpd
|   |-- init.d -> rc.d/init.d
|   |-- iproute2
|   |-- isdn
|   |-- ld.so.conf.d
|   |-- logrotate.d
|   |-- logwatch
|   |-- lsb-release.d
|   |-- lvm
|   |-- mail
|   |-- makedev.d
|   |-- mgetty+sendfax
|   |-- modprobe.d
|   |-- netplug
|   |-- netplug.d
|   |-- news
|   |-- oddjob
|   |-- oddjobd.conf.d
|   |-- openldap
|   |-- opt
|   |-- pam.d
|   |-- pam_pkcs11
|   |-- pango
|   |-- pcmcia
|   |-- php.d
|   |-- pki
|   |-- pm
|   |-- ppp
|   |-- profile.d
|   |-- racoon
|   |-- rc.d
|   |-- rc0.d -> rc.d/rc0.d
|   |-- rc1.d -> rc.d/rc1.d
|   |-- rc2.d -> rc.d/rc2.d
|   |-- rc3.d -> rc.d/rc3.d
|   |-- rc4.d -> rc.d/rc4.d
|   |-- rc5.d -> rc.d/rc5.d
|   |-- rc6.d -> rc.d/rc6.d
```

```
|   |-- readahead.d
|   |-- reader.conf.d
|   |-- redhat-lsb
|   |-- rpm
|   |-- rwtab.d
|   |-- samba
|   |-- sasl2
|   |-- security
|   |-- selinux
|   |-- setuptool.d
|   |-- skel
|   |-- smrsh
|   |-- sound
|   |-- squid
|   |-- ssh
|   |-- stunnel
|   |-- sysconfig
|   |-- udev
|   |-- vsftpd
|   |-- wpa_supplicant
|   |-- xdg
|   |-- xinetd.d
|   |-- yum
|   `-- yum.repos.d
|-- home
|   |-- crq
|   |-- jason
|   |-- lost+found
|   `-- osmond
|-- lib
|   |-- bdevid
|   |-- dbus-1.0
|   |-- firmware
|   |-- i686
|   |-- iptables
|   |-- kbd
|   |-- lsb
|   |-- modules
|   |-- rtkaio
|   |-- security
|   |-- tls
|   `-- udev
|-- lost+found
|-- media
|-- misc
|-- mnt
|-- net
|-- opt
|-- proc
|   |-- 1
|   |-- 10
|   |-- 11
|   |-- 1193
|   |-- 1435
|   |-- 1436
|   |-- 1461
|   |-- 1463
|   |-- 15
|   |-- 151
|   |-- 152
|   |-- 153
|   |-- 154
|   |-- 155
|   |-- 16
|   |-- 17
|   |-- 1929
|   |-- 1931
|   |-- 1961
|   |-- 1964
|   |-- 1980
|   |-- 2
|   |-- 2015
|   |-- 2040
|   |-- 2086
|   |-- 2113
|   |-- 2128
|   |-- 2132
|   |-- 2159
|   |-- 2206
|   |-- 2230
|   |-- 2250
```

```
|   |-- 2273
|   |-- 2299
|   |-- 2316
|   |-- 2345
|   |-- 2368
|   |-- 2397
|   |-- 2412
|   |-- 2428
|   |-- 2429
|   |-- 2444
|   |-- 2445
|   |-- 2453
|   |-- 2459
|   |-- 2468
|   |-- 2500
|   |-- 2503
|   |-- 2504
|   |-- 2505
|   |-- 2512
|   |-- 2513
|   |-- 2514
|   |-- 2584
|   |-- 2586
|   |-- 2587
|   |-- 2639
|   |-- 2640
|   |-- 2664
|   |-- 3
|   |-- 309
|   |-- 351
|   |-- 355
|   |-- 356
|   |-- 357
|   |-- 373
|   |-- 376
|   |-- 4
|   |-- 403
|   |-- 437
|   |-- 5
|   |-- 6
|   |-- 7
|   |-- 78
|   |-- 79
|   |-- 8
|   |-- 82
|   |-- 84
|   |-- 9
|   |-- acpi
|   |-- asound
|   |-- bus
|   |-- driver
|   |-- fs
|   |-- ide
|   |-- irq
|   |-- mpt
|   |-- net
|   |-- scsi
|   |-- self -> 2664
|   |-- sys
|   |-- sysvipc
|   `-- tty
|-- root
|   `-- bin
|-- sbin
|-- selinux
|-- srv
|-- sys
|   |-- block
|   |-- bus
|   |-- class
|   |-- devices
|   |-- firmware
|   |-- fs
|   |-- kernel
|   |-- module
|   `-- power
|-- tmp
|-- usr
|   |-- bin
|   |-- etc
|   |-- games
```

```
|   |-- include
|   |-- kerberos
|   |-- lib
|   |-- libexec
|   |-- local
|   |-- sbin
|   |-- share
|   |-- src
|   `-- tmp -> ../var/tmp
`-- var
    |-- account
    |-- cache
    |-- db
    |-- empty
    |-- ftp
    |-- games
    |-- lib
    |-- local
    |-- lock
    |-- log
    |-- mail -> spool/mail
    |-- named
    |-- nis
    |-- opt
    |-- preserve
    |-- racoon
    |-- run
    |-- spool
    |-- tmp
    |-- tux
    |-- www
    `-- yp
275 directories
```

几个重要的文件系统

/etc 文件系统

目录	说明
/etc/X11	X Windows 的设置目录
/etc/alternatives	存储具有相同功能程序的二/多选一链接的目录
/etc/apm	高级电源管理的配置目录
/etc/init.d	守护进程启动脚本目录
/etc/logrotate.d	日志滚动脚本的配置目录
/etc/lvm	LVM2 的配置目录
/etc/opt	/opt 应用程序的配置目录
/etc/pam.d	PAM 配置目录
/etc/pcmcia	pcmcia 设备配置目录
/etc/ppp	ppp 设备配置目录
/etc/rc?.d	启动、或改变运行级时运行的 scripts 目录
/etc/skel	普通用户初始环境目录
/etc/ssh	ssh 的配置目录
/etc/vim	vim 的配置目录
/etc/w3m	w3m 的配置目录
/etc/yum	yum 的配置目录
/etc/yum.repos.d	yum 源的配置目录

/usr 文件系统

目录	说明
/usr/X11R6	存放 X Window 的目录
/usr/bin	存放了许多用户命令
/usr/games	存放游戏和教育类软件
/usr/include	存放Linux下开发和编译应用程序所需要的头文件
/usr/lib	放一些常用的动态链接共享库和静态档案库
/usr/local	供给本地用户的/usr目录，在这里安装本地的应用软件
/usr/sbin	存放root超级用户使用的管理程序
/usr/share	系统共用的东西存放地，如：手册、文档、字体等

/usr/src	是内核源码存放的目录

/var 文件系统

目录	说明
/var/cache	应用程序缓存的数据目录
/var/lib	存储系统或各个应用程序运行时的状态信息数据
/var/lock	存储程序运行时的锁定文件的目录。许多程序遵循在/var/lock 中产生一个锁定文件的约定，以支持他们正在使用某个特定的设备或文件。其他程序注意到这个锁定文件，将不试图使用这个设备或文件
/var/log	系统日志存放，分析日志要看这个目录的东西
/var/mail	用户 mailbox 文件存储目录
/var/opt	存储 /opt 目录下应用程序的经常变化的数据
/var/run	存储到下次引导前有效的关于系统的信息文件
/var/spool	打印机、邮件、代理服务器等假脱机目录
/var/tmp	存放临时文件

几个特殊的文件系统

文件系统	挂装点	说明
root	/	Linux 系统运行的基点，根文件系统不能被卸载
proc	/proc	以文件系统的方式为访问系统内核数据的操作提供接口，适用于2.4和2.6内核
sysfs	/sys	以文件系统的方式为访问系统内核数据的操作提供接口，2.6内核使用
tmpfs	/dev /var/run /var/lock	程序访问共享内存资源时使用的文件系统
usbfs	/proc/bus/usb	访问 USB 设备时使用的文件系统
devpts	/dev/pts	内核用来与伪终端进行交互的文件系统
swap	内核使用的特殊文件系统，无挂装点	用来创建虚拟内存

/proc

下面说明 /proc 的一些最重要的文件和目录。

文件	说明
/proc/n	n为PID，每个进程在 /proc 下有一个名为其进程号的目录。
/proc/cpuinfo	处理器信息，如类型、制造商、型号和性能。
/proc/devices	当前运行的核心配置的设备驱动的列表。
/proc/dma	显示当前使用的DMA通道。
/proc/filesystems	核心配置的文件系统。
/proc/interrupts	显示使用的中断。
/proc/ioports	当前使用的I/O端口。
/proc/kcore	系统物理内存映象。与物理内存大小完全一样，但不实际占用内存。
/proc/kmsg	核心输出的消息。也被送到syslog 。
/proc/ksyms	核心符号表。
/proc/loadavg	系统"平均负载"：3个指示器指出系统当前的工作量。
/proc/meminfo	存储器使用信息，包括物理内存和虚拟内存。
/proc/modules	当前加载了哪些核心模块。
/proc/net	网络协议状态信息。
/proc/self	到查看/proc 的程序的进程目录的符号连接。当多个进程查看 /proc 时，是不同的连接。
/proc/stat	系统状态。
/proc/uptime	系统启动的时间长度。
/proc/version	内核版本信息。

主流的日志文件系统

ext3

ext3 文件系统是直接从 ext2 文件系统发展而来，目前 ext3 文件系统已经非常稳定可靠。它完全兼容 ext2 文件系统。用户可以平滑地过渡到一个日志功能健全的文件系统中来。这实际上了也是 ext3 日志文件系统初始设计的初衷。ext3日志文件系统具有如下特点:

- 高可用性：系统使用了ext3文件系统后，即使在非正常关机后，系统也不需要检查文件系统。宕机发生后，恢复ext3文件系统的时间只要数十秒钟。
- 数据的完整性：ext3文件系统能够极大地提高文件系统的完整性，避免了意外宕机对文件系统的破坏。在保证数据完整性方面，ext3文件系统有2种模式可供选择。其中之一就是"同时保持文件系统及数据的一致性"模式。采用这种方式，你永远不再会看到由于非正常关机而存储在磁盘上的垃圾文件。
- 文件系统的速度：尽管使用ext3文件系统时，有时在存储数据时可能要多次写数据，但是，从总体上看来，ext3比ext2的性能还要好一些。这是因为ext3的日志功能对磁盘的驱动器读写头进行了优化。所以，文件系统的读写性能较之ext2文件系统并来说，性能并没有降低。
- 数据转换：由ext2文件系统转换成ext3文件系统非常容易，只要简单地键入两条命令即可完成整个转换过程，用户不用花时间备份、恢复、格式化分区等。用

一个 ext3文件系统提供的小工具tune2fs，它可以将ext2文件系统轻松转换为ext3日志文件系统。另外，ext3文件系统可以不经任何更改，而直接加载成为ext2文件系统。

- 多种日志模式：ext3有多种日志模式，一种工作模式是对所有的文件数据及metadata（定义文件系统中数据的数据,即数据的数据）进行日志记录（data=journal模式）；另一种工作模式则是只对 metadata 记录日志，而不对数据进行日志记录，也即所谓 data=ordered 或者 data=writeback 模式。系统管理人员可以根据系统的实际工作要求，在系统的工作速度与文件数据的一致性之间作出选择。

XFS

XfS 文件系统是 SGI 开发的高级日志文件系统，XFS 极具伸缩性，非常健壮。所幸的是 SGI 将其移植到了 Linux系统中。XFS 的主要特性如下：

- 数据完全性：采用XFS文件系统，当意想不到的宕机发生后，首先，由于文件系统开启了日志功能，所以你磁盘上的文件不再会意外宕机而遭到破坏了。不论目前文件系统上存储的文件与数据有多少，文件系统都可以根据所记录的日志在很短的时间内迅速恢复磁盘文件内容。
- 传输特性：XFS文件系统采用优化算法，日志记录对整体文件操作影响非常小。XFS查询与分配存储空间非常快。xfs文件系统能连续提供快速的反应时间。笔者曾经对XFS、JFS、Ext3、ReiserFS文件系统进行过测试，XFS文件文件系统的性能表现相当出众。
- 可扩展性：XFS 是一个全64-bit的文件系统，它可以支持上百万T字节的存储空间。对特大文件及小尺寸文件的支持都表现出众，支持特大数量的目录。最大可支持的文件大小为263 = 9 x 1018 = 9 exabytes，最大文件系统尺寸为18 exabytes。 XFS使用高的表结构(B+树)，保证了文件系统可以快速搜索与快速空间分配。XFS能够持续提供高速操作，文件系统的性能不受目录中目录及文件数量的限制。
- 传输带宽：XFS 能以接近裸设备I/O的性能存储数据。在单个文件系统的测试中，其吞吐量最高可达7GB每秒，对单个文件的读写操作，其吞吐量可达4GB每秒。

JFS

JFS 是 IBM 公司为 Linux 系统开发的一个日志文件系统。从 IBM 的实力及它对 Linux 的态度来看，JFS 应该是未来日志文件系统中最具实力的一个文件系统。JFS 提供了基于日志的字节级文件系统，该文件系统是为面向事务的高性能系统而开发的。JFS 能够在几秒或几分钟内就把文件系统恢复到一致状态。JFS 能够保证数据在任何意外宕机的情况下，不会造成磁盘数据的丢失与损坏。JFS文件系统的特点如下：

- 存储空间更大：JFS 支持的最小文件系统是 16M 字节。最大文件系统的大小为 512 万亿字节（TB）。JFS 是真正意义上的 64 位的文件系统。所有 JFS 文件系统结构化字段都是 64 位大小。
- 动态磁盘 **inode** 分配：JFS 按需为磁盘 inode 动态地分配空间，释放不再需要的空间。这种方式避开了在文件系统创建期间，为磁盘 inode 保留固定数量空间的传统方法。用户不需要考虑文件系统包含的文件和目录最大数目。
- 基于盘区的寻址结构：JFS 使用基于盘区的寻址结构，JFS 分配尝试通过分配最小数量的盘区策略，而使每个盘区尽可能大。这有利于大的 I/O 传送，磁盘读写性能所有提高。
- 块尺寸可变：JFS 支持 512、1024、2048 和 4096 字节的块尺寸，允许用户根据应用环境优化空间利用率。较小的块尺寸减少有利于内部存储碎片的数量，提高空间利用率。系统缺省块尺寸为 4096 字节。

ReiserFS

ReiserFS 是一个非常优秀的文件系统，也是最早用于 Linux 的日志文件系统之一。ReiserFS 的开发者非常有魄力，整个文件系统完全是从头设计的。ReiserFS可轻松管理上百G的文件系统，这在企业级应用中非常重要。ReiserFS的特点如下：

- 先进的日志机制：ReiserFS 有先进的日志(Journaling/logging)功能机制。日志机制保证了在每个实际数据修改之前，相应的日志已经写入硬盘。文件与数据的安全性有了很大提高。
- 高效的磁盘空间利用：ReiserFS 对一些小文件不分配inode。而是将这些文件打包，存放在同一个磁盘分块中。而其它文件系统则为每个小文件分别放置到一个磁盘分块中。这意味着：如果有10000个小文件，就要占用10000个分块。想想看这多浪费磁盘空间。
- 独特的搜寻方式：ReiserFS 基于快速平衡树(balanced tree)搜索，平衡树在性能上非常卓越，这是一种非常高效的算法。ReiserFS搜索大量文件时，搜索速度要比ext2快得多。ReiserFS文件系统使用B*Tree存储文件，而其它文件系统使用B+Tree树。B*Tree查询速度比B+Tree要快很多。ReiserFS在文件定位上速度非常快。在实际运用中，ReiserFS 在处理小于 1k 的文件时，比ext2 快 8 到 15 倍！ReiserFS 几乎在各个方面都优于 ext2，具体数据请参见笔者的测试篇。
- 支持海量磁盘：ReiserFS是一个非常优秀的文件系统，可轻松管理上百G的文件系统，ReiserFS文件系统最大支持的文件系统尺寸为16TB。这非常适合企业级应用中。
- 优异的性能：由于它的高效存储和快速小文件I/O特点，使用ReiserFS文件系统的PC，在启动X窗口系统时，所花的时间要比在同一台机器上使用ext2文件系统少1/3。另外，ReiserFS文件系统支持单个文件尺寸为4G的文件，这为大型数据库系统在 Linux 上的应用提供了更好的选择。

参考

- Filesystems (ext3, reiser, xfs, jfs) comparison on Debian Etch [http://www.proxyserve.net/index.php?q=aHR0cDovL3d3dy5kZWJpYW4tYWRtaW5pc3RyYXRpb24ub3JnL2FydGljbGVzLzM4OA%3D%3D]
- http://blog.csdn.net/ak47mig/archive/2006/07/08/891397.aspx [http://www.proxyserve.net/index.php?q=aHR0cDovL2Jsb2cuY3Nkbi5uZXQvYWs0N21pZy9hcmNoaXZlLzIwMDYvMDcvMDgvODkxMzk3LmFzcHg%3D]
- http://hi.baidu.com/xuzhi1977/blog/item/c5869758dfafbade9d82040a.html [http://www.proxyserve.net/index.php?q=aHR0cDovL2hpLmJhaWR1LmNvbS94dXpoaTE5NzcvYmxvZy9pdGVtL2M1ODY5NzU4ZGZhZmJhZGU5ZDgyMDQwYS5odG1s]

其他类型的文件系统

ext2

ext2 文件系统（二次扩展文件系统）是Linux中自带的文件系统类型，该文件系统是Linux中原来使用的 ext 文件系统的后续版本，在 Linux 早期发行版本中

一直使用 ext2 作为操作系统默认使用的文件系统。

swap

swap文件系统在Linux中作为交换分区使用，交换分区用于操作系统管理内存的交换空间。在安装Linux操作系统时，交换分区是必须建立的，并且其类型一定是swap。交换分区由操作系统自动管理，用户不需要对其进行过多的操作。

vfat

在Linux中把DOS下的所有FAT文件系统统称为vfat，其中包括FAT12、FAT16和FAT32。Red Hat Linux 9中既可以使用同系统中已存在的FAT分区，也可以建立新的FAT分区。

NFS

NFS即网络文件系统，用于在UNIX系统间通过网络进行文件共享，用户可以把网络中NFS服务器提供的共享目录挂装到本地文件目录中，可以像对本地文件系统一样操作NFS文件系统中的内容。

ISO9660

ISO9660是光盘所使用的标准文件系统，在Linux中对光盘有非常好的支持，不仅可以读取光盘中的文件，还可以进行光盘的刻录。

非日志文件系统和日志文件系统

非日志文件系统是如何工作的？

文件系统通过为文件分配文件块的方式把数据存储在磁盘上。每个文件在磁盘上都会占用一个以上的磁盘扇区，文件系统的工作就是维护文件在磁盘上的存放，记录文件占用了哪几个扇区。另外扇区的使用情况也要记录在磁盘上。文件系统在读写文件时，首先找到文件使用的扇区号，然后从中读出文件内容。如果要写文件，文件系统首先找到可用扇区，进行数据追加。随后或同时更新文件扇区使用信息。

非日志文件系统的缺陷

ext2的设计者主要考虑的是文件系统的效率和性能方面的问题。ext2在写入文件内容的同时并没有同时写入文件的meta-data（和文件有关的信息，例如：权限、所有者以及创建和访问时间）。换句话说，Linux先写入文件的内容，然后等到有空的时候才写入文件的meta-data。如果在写入文件内容之后写入文件的meta-data之前，突然断电了，文件系统就会处于不一致的状态。在一个需要大量文件操作的系统中（例如，像Hotmail这样的免费的Web e-mail），出现这种情况会导致很严重的后果。日志文件系统可以帮助解决这个问题。

日志文件系统是如何工作的？

在日志文件系统中，所有的文件系统的变化、添加和改变都被记录到“日志”(即记录文件metadata信息的数据)中。每隔一定时间，文件系统会将更新后的文件metadata及文件内容写入磁盘，之后删除这部分日志。重新开始新日志记录。 在对元数据做任何改变以前，文件系统驱动程序会向日志中写入一个条目，这个条目描述了它将要做些什么。然后，它继续并修改元数据。

通过这种方法，日志文件系统就拥有了近期元数据被修改的历史记录，当检查到没有彻底卸载的文件系统的一致性问题时，只要根据数据的修改历史进行相应的检查即可了。也即日志文件系统除了存储数据和元数据（metadata）以外，它们还保存有一个日志，我们可以称之为元元数据（关于元数据的元数据）。

日志文件系统的系统开销

日志文件系统使得数据、文件变安全了，但是系统开销加大了。每一次更新和大多数的日志操作都需要写同步，这需要更多的磁盘I/O操作。

从日志文件的原理出发，将那些需要经常写操作的分区上使用日志文件系统是一个好的主意。 Linux系统中可以混合使用日志文件系统或非日志文件系统。日志增加了文件操作的时间，但是，从文件安全性角度出发，磁盘文件的安全性得到了重大的提高。

使用日志文件系统的好处

文件的安全提高了，文件被破坏的机率降低了，对磁盘的扫描时间缩短了，扫描次数减少了。当系统意外宕机后，不会再有文件内容的丢失，至少文件应该保持上一个版本的内容；采用日志文件系统，通常系统每重新启动20－30次后，才会对磁盘进行一次整体扫描，扫描次数减少了。

使用文件系统的一般方法

在Linux中使用的文件系统通常是在我们安装Linux时建立的，但是对于实际的应用系统中经常会需要对现有的分区进行调整或建立新的分区和 LVM 的情况。

要使用文件系统，一般要遵循如下的过程：

1. 在硬盘上创建分区：可以使用fdisk命令进行。
2. 在分区或逻辑卷上创建文件系统：类似于在Windows下进行格式化操作。
3. 挂装文件系统到系统中：在分区中创建好文件系统后就可以将该分区挂装到系统中的相应目录以便使用。
   - 挂装文件系统可以使用 mount 命令。

- 若需要系统每次启动时都自动挂装该文件系统则需要在文件“/etc/fstab”中添加相应的设置行。

4. 卸装文件系统：对于可移动介质上的文件系统，当使用完毕需要使用 umount 命令实施卸装操作。

参考

- http://www.xenotime.net/linux/linux-fs.html [http://www.proxyserve.net/index.php?q=aHR0cDovL3d3dy54ZW5vdGltZS5uZXQvbGludXgvbGludXgtZnMuaHRtbA%3D%3D]
- 显示源文件
- 登录