块设备管理

毛迪林

dlmao@fudan.edu.cn

块设备管理概述

- 块设备对应的设备文件是什么？ lsblk -f查看
- 是使用整个磁盘还是划分为多个分区后使用？fdisk分区
- 对于整个磁盘或者分区创建文件系统 : mkfs/gparted
- 挂载文件系统到某个挂载点以及卸载 mount/umount

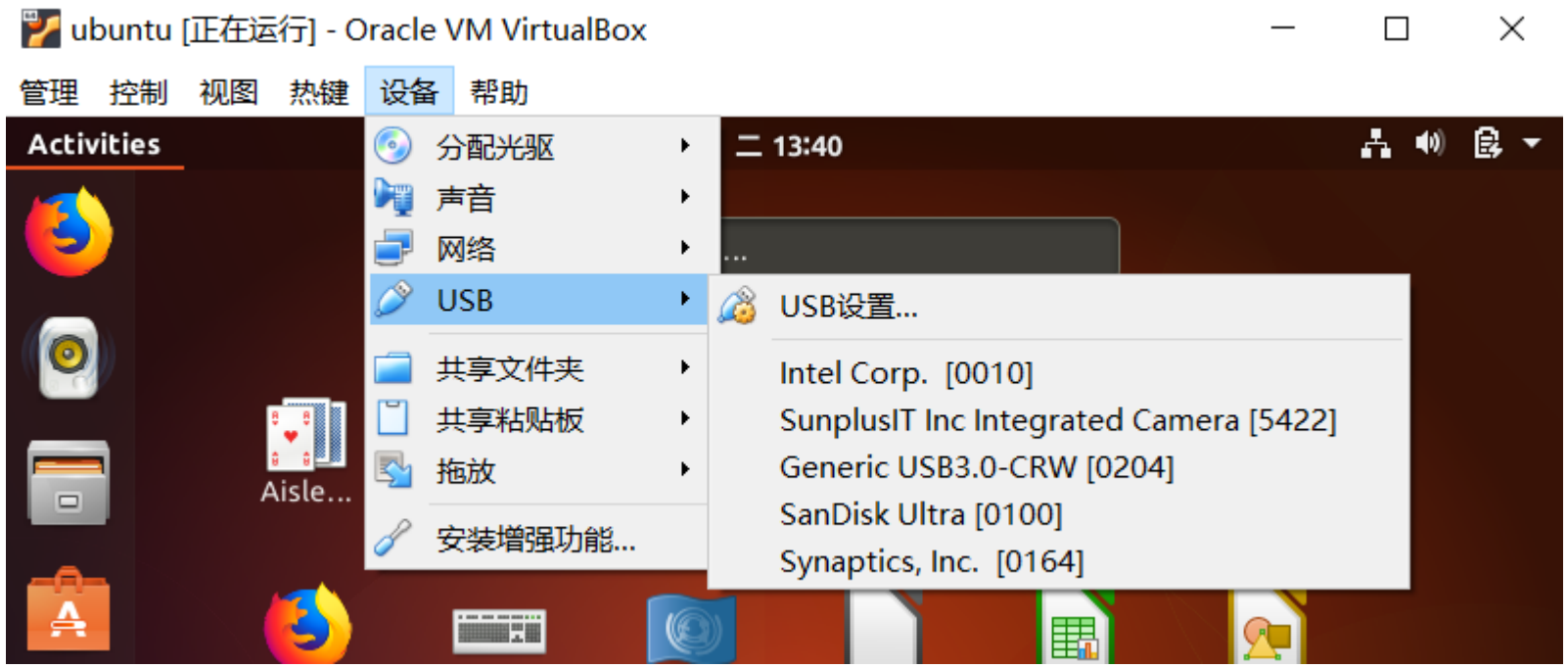
块设备新增和管理

- Virtualbox虚拟机设置→存储里面添加新的虚拟光盘和虚拟硬盘
- 重新启动，由于新加硬盘，如果新加的硬盘为第一个硬盘，则可能需要按F12选择原有的硬盘启动
- dmesg |grep -i attached 可以查看系统识别的存储设备

```
[    1.500733] sr 0:0:0:0: Attached scsi CD-ROM sr0
[    1.500907] sr 0:0:0:0: Attached scsi generic sg0 type 5
[    1.504048] sr 0:0:1:0: Attached scsi CD-ROM sr1
[    1.504203] sr 0:0:1:0: Attached scsi generic sg1 type 5
[    1.505031] sd 1:0:1:0: Attached scsi generic sg2 type 0
[    1.507416] sd 1:0:1:0: [sda] Attached SCSI disk
[    1.979671] sd 2:0:0:0: Attached scsi generic sg3 type 0
[    1.990112] sd 2:0:0:0: [sdb] Attached SCSI disk
```

块设备新增和管理

- 虚拟机要使用外部的USB 2.0/3.0设备，需要安装**VirtualBox Extension Pack**
- **在安装好后，应该在虚拟机的设置→USB驱动器中选择启动USB3.0或2.0控制器**
- VirtualBox中设备→USB→选择对应的U盘，VirtualBox捕获该U盘供当前虚拟机使用



lsblk [options] 查看块设备信息

- -f 查看文件系统相关(类型,卷标,UUID等)
- -o, --output <list> 自定义输出的字段列表，list前面如果为+，表示增加这些字段
- -b, --bytes 空间单位为字节而不是人可读(human readable)格式

```
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sda      8:0    0  1.1G  0 disk
sdb      8:16   0   20G  0 disk
 ├─sdb1   8:17   0   18G  0 part /
 ├─sdb2   8:18   0    1K  0 part
 └─sdb5   8:21   0    2G  0 part [SWAP]
sr0      11:0    1 56.5M  0 rom
sr1      11:1    1   16M  0 rom
```

lsblk -o +fstype,label,uuid,partuuid 表示在常规显示字段之后添加对应的字段

lsblk -o NAME,RM,FSTYPE,MOUNTPOINT,SIZE,OWNER,GROUP,TYPE,UUID

RM: 可移动块设备?

```
[23:16]demo@mars:~$ lsblk -o NAME,RM,FSTYPE,MOUNTPOINT,SIZE,OWNER,GROUP,TYPE,UUID
NAME    RM FSTYPE MOUNTPOINT          SIZE OWNER GROUP TYPE UUID
sda      0                             20G root  disk  disk
├─sda1   0 ext4   /                    18G root  disk  part 0ca1c737-726a-43db-b577-0bfefb7b35a5
├─sda2   0                              1K root  disk  part
└─sda5   0 swap   [SWAP]                2G root  disk  part f0c2c6fe-480b-47c2-a284-d1fc000665cd
sdb      0                            1.1G root  disk  disk
├─sdb1   0 ext4   /home2               800M root  disk  part f4568103-ce83-4709-9463-9d56e4d7c1c9
└─sdb2   0 vfat   /home/demo/msdos 310.2M root  disk  part 9490-58C9
sdc      0                              1G root  disk  disk
```

UUID唯一标识磁盘或者分区

查看当前加载文件系统：mount命令

- Linux系统的动态设备管理udev能够自动加载光盘
- mount命令用于加载文件系统，不带参数时表示查看当前加载的所有文件系统

```
demo@mars:~$ mount |grep media
/dev/sr1 on /media/demo/TinyCore type iso9660
(ro,nosuid,nodev,relatime,uid=1000,gid=1000,iocharset=utf8,mode=0400,dmode=0500,uhelper=udisks2)
/dev/sr0 on /media/demo/VBOXADDITIONS_5.1.6_110634 type iso9660
(ro,nosuid,nodev,relatime,uid=1000,gid=1000,iocharset=utf8,mode=0400,dmode=0500,uhelper=udisks2)
```

```
demo@mars:~$ mount |grep '^/dev'
mount |grep '^/dev'
/dev/sdb1 on / type ext4 (rw,relatime,errors=remount-ro,data=ordered)
/dev/sr1 on /media/demo/TinyCore type iso9660
(ro,nosuid,nodev,relatime,uid=1000,gid=1000,iocharset=utf8,mode=0400,dmode=0500,uhelper=udisks2)
/dev/sr0 on /media/demo/VBOXADDITIONS_5.1.6_110634 type iso9660
(ro,nosuid,nodev,relatime,uid=1000,gid=1000,iocharset=utf8,mode=0400,dmode=0500,uhelper=udisks2)
```

维护分区表: fdisk

- Linux下对于SCSI、SATA设备等以sd命名，第一个磁盘为sda，之后的为sdb
- 磁盘进一步被分为多个分区，分区表维护了各个分区的信息，每个分区创建一个文件系统
  - 每个分区可以独立管理，使用不同的文件系统以及不同的文件系统参数
- GUI程序gnome-disks或者gparted
- fdisk命令管理分区表
  - o: 创建空的dos分区表
  - g: 创建空的GPT分区表
  - p: 显示当前分区表
  - n: 创建新分区
  - t: 改变分区类型
  - w: 分区表写入设备

```
demo@mars:~$ sudo fdisk /dev/sda
[sudo] password for demo:

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x7e40a4c9.
 Command (m for help): p
Disk /dev/sda: 1.1 GiB, 1165123584 bytes, 2275632 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x7e40a4c9
```

维护分区表

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1):
First sector (2048-2275631, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-2275631, default 2275631): +800M

Created a new partition 1 of type 'Linux' and of size 800 MiB.

Command (m for help): n
...
Created a new partition 2 of type 'Linux' and of size 310.2 MiB.

Command (m for help): t
Partition number (1,2, default 2): 2
Partition type (type L to list all types): L
```

```
Partition type (type L to list all types): 7

Changed type of partition 'Linux' to 'HPFS/NTFS/exFAT'
Command (m for help): p
Disk /dev/sda: 1.1 GiB, 1165123584 bytes, 2275632 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x8c915425

Device     Boot Start     End Sectors  Size Id Type
/dev/sda1        2048 1640447 1638400  800M 83 Linux
/dev/sda2     1640448 2275631  635184 310.2M  7
HPFS/NTFS/exFAT

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

创建文件系统:mkfs

e2label device label： 为device对应的文件系统添加逻辑卷标label

- lsblk –f 可以查看和文件系统相关的信息（包括uuid）
- man -k mkfs可以看到当前主机所支持的创建文件系统的命令
- mkfs –t type device: 通用的命令，在device对应的块设备上建立一个类型为type的文件系统。建议直接调用对应的特定文件系统命令
- mkfs.ext4 (ext2/ext3)创建Linux文件系统， e2label可以为文件系统设置逻辑卷标
- mkfs.vfat –F 16/32 device 创建Fat16或Fat32文件系统或不用-F选项时自动选择FAT16或FAT32
- mkfs.ntfs device创建NTFS文件系统
- dumpe2fs 命令列出ext文件系统的超级块和数据块组的信息
- tune2fs 调整参数，转化ext文件系统

```
mkfs.ext4 [options] device
-b 指定block的大小1024 2048 4096 默认是4096
-I 指定inode的大小(min 128/max 4096) 默认是256
-i 指定多少个字节分配一个inode
-L label 指定文件系统逻辑卷标为label
```

```
demo@mars:~$ sudo mkfs.ext4 /dev/sda1
mke2fs 1.42.13 (17-May-2015)
Creating filesystem with 204800 4k blocks and 51296 inodes
Filesystem UUID: f4568103-ce83-4709-9463-9d56e4d7c1c9
Superblock backups stored on blocks:
        32768, 98304, 163840
demo@mars:~$ sudo mkfs.vfat /dev/sda2
mkfs.fat 3.0.28 (2015-05-16)
```

挂载文件系统mount

```
demo@mars:~$ sudo mkdir /home2
demo@mars:~$ sudo mount /dev/sda1 /home2
demo@mars:/home2$ df –h |grep sd
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb1        18G  6.8G   10G  41% /
/dev/sda1       772M  808K  715M   1% /home2
```

除非在/etc/fstab里面特别设置，挂载和卸载都要超级用户权限

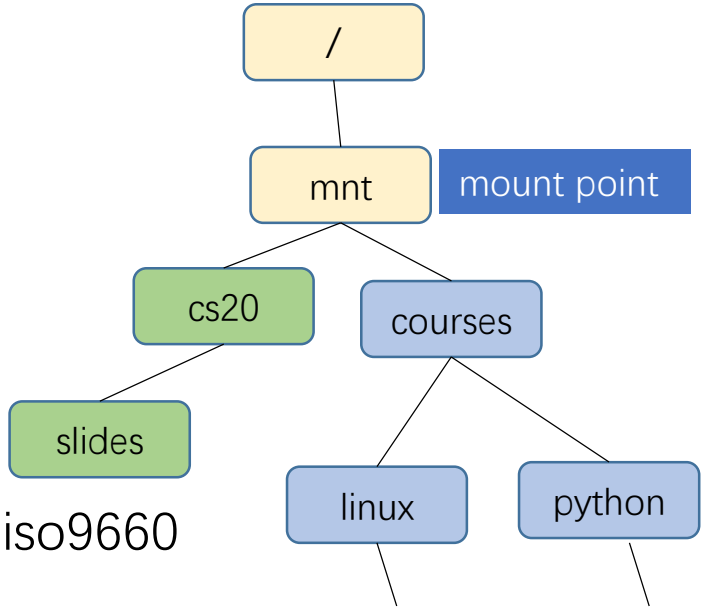
mount [-t fstype] [-o options] device dir

mount –a 自动挂载/etc/fstab给出的标记为auto的分区

mount device|dir 挂载/etc/fstab中设备或者目录所对应的文件系统

- mount命令把某个设备上的分区device挂载到某个目录dir
  - 该分区的文件系统的内容可以通过挂载点访问
  - 原挂载点下的内容暂时屏蔽不可访问，等待umount后可以继续访问
- -t 选项可以指定挂载分区的文件系统类型(ext2, ext3, ext4, msdos,vfat,ntfs, iso9660等)，一般会自动探测文件系统类型
  - cat /proc/filesystems查看支持的文件系统类型
- -o options：给出了挂载文件系统时使用的选项

umount [-l] directory|device 卸载已经挂载的文件系统，-l选项(lazy)表示在设备忙时也卸载，等待在那些导致设备忙的进程退出时释放相应的资源



在挂载前/mnt目录下的cs20分支可以访问，但是挂载文件系统到/mnt之后，cs20分支暂时不可访问

挂载文件系统mount

类型	含义
devpts	提供访问伪终端的设备文件接口， /dev/pts
procfs	提供访问内核状态的接口
sysfs	提供访问系统数据(设备和驱动等)的接口
tmpfs	提供访问临时文件的接口

- -o options：给出了挂载文件系统时使用的选项, defaults 表示使用所有选项的默认值（ rw, suid, dev, exec, auto, nouser, async. ）

选项	含义	选项	含义
auto/noauto	是否可用-a选项挂载	rw	读写方式挂载
dev/nodev	是否对文件系统上的设备文件进行解释	loop	挂载loop设备
exec/noexec	是否允许运行系统上的二进制文件	user/nouser	是否允许普通用户挂载(如果挂载成功卸载只有该用户或超级用户允许)
suid/nosuid	是否允许suid/sgid起作用	owner/noowner	允许设备文件的拥有者mount
async/sync	是否同步方式	group/nogroup	允许设备文件的用户组成员mount
remount	重新挂载已挂载的文件系统	users	允许所有用户挂载和卸载
ro	只读方式挂载		

sudu mount –o remount,rw /dev/sda1 #重新挂载，只是以读写方式挂载
sudo mount –a # 自动挂载在/etc/fstab中配置好的块设备

挂载FAT和NTFS文件系统

- 微软使用的文件系统实现与Linux下的文件系统不一致，特别是权限控制，因此挂载时模拟Linux的权限控制的相关字段，主要是uid和gid以及umask。缺省情况下挂载这些文件系统时uid和gid设为执行挂载操作的进程的uid和gid，一般为root，而缺省的umask有可能为进程的umask，在读写该文件系统上的文件时umask优先，可能出现只有root可读，但是不可写，即便拥有超级用户权限

选项	含义	选项	含义
uid=value	设置文件系统中所有文件的拥有者	codepage=value	设置转换FAT上的短文件名时使用的代码页。简体中文为936
gid=value	设置文件系统中所有文件的拥有者	iocharset=value	设置转换FAT上的长(16位Unicode)文件名使用的字符集， iocharset=utf8
umask=value	设置文件系统中所有文件和目录所不允许的权限	utf8	用于FAT，采用utf8编码，建议采用，而不是iocharset=utf8,
fmask=value	设置文件系统中所有文件所不允许的权限	nls=name	NTFS中使用，转换文件名使用的字符集，比如nls=utf8
dmask=value	设置文件系统中所有目录所不允许的权限		

sudo mount -o utf8,umask=000 /dev/sda2 /home/demo/msdos #任何人都可以读写

sudo mount -o utf8,uid=$(id –u),gid=$(id –g) /dev/sda2 /home/demo/msdos # 拥有者和用户组都是当前用户

自动挂载文件系统

如果采用GPT分区表，则还可通过PARTLABEL或者PARTUUID来标识相应的文件系统

- /etc/mtab以及/proc/partitions 保存了当前已经挂载的文件系统信息
- /etc/fstab保存了可以挂载的文件系统的信息
  - 第一个字段描述块设备文件，可以是设备文件名，也可以是LABEL=<label>或者UUID=<uuid> 如果其中有空格，代替以 \040
    - LABEL为文件系统的逻辑卷标：命令e2label/dosfslabel/ntfslabel查看或修改卷标
    - UUID为唯一标识该文件系统的字符串（小写）：blkid device或者lsblk –f device可查看
    - 由于设备增删时设备文件名可能会变动，建议采用LABEL或者UUID
  - dump字段： 备份命令dump时使用，为0表示不备份，1表示备份
  - pass字段： 启动时使用fsck命令检查文件系统时的顺序，0表示不检验，1表示最早检验(根文件系统) ,2表示最晚检验(一般的文件系统)

```
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
UUID=0ca1c737-726a-43db-b577-0bfefb7b35a5 /                 ext4    errors=remount-ro 0       1
UUID=f4568103-ce83-4709-9463-9d56e4d7c1c9 /home2            ext4    errors=remount-ro 0       2
UUID=f0c2c6fe-480b-47c2-a284-d1fc000665cd none              swap    sw                0
# 任何用户可调用mount /dev/sdb4或/home/demo/dlmao16-netac来加载(uid=..,gid=…)，该用户也可umount
/dev/sdb4      /home/demo/dlmao16-netac   vfat  noauto,rw,user,utf8   0   0
```

命令行加载： sudo mount -o rw,utf8,uid=$(id -u),gid=$(id -g) /dev/sdb4 ~/dlmao16-netac

映像文件作为文件系统

- 回环设备文件/dev/loop是一种伪设备，使得文件可以如同块设备一样被访问
- mount命令的loop选项会自动创建(自动调用losetup)对应着映像文件的loop设备，并且挂载该映像文件中的文件系统

mkfs /tmp/disk.img 在映像文件上创建一个ext文件系统，也可通过-t fstype创建其他类型文件系统

mount /tmp/disk.img /mnt # 会自动采用loop选项，挂载映像文件到/mnt

mount -t vfat -o loop /tmp/disk.img /mnt # 自动选择可用loop设备，然后mount映像文件

mount -t vfat -o loop=/dev/loop3 /tmp/disk.img /mnt 也可以指定一个loop设备

losetup可以更精细地设置和控制回环设备，不介绍

- 列出第一个可用(find)的回环设备： losetup –f
- 设置回环设备，使用第一个可用的回环设备: losetup -f --show imgfile
- 查看回环设备信息: losetup loopdev 或者 losetup –l 或者losetup –j imgfile
- 断开(detach)回环设备： losetup –d loopdev 或losetup –D

```
root@mars:/home/demo# losetup -f --show vbox.iso
/dev/loop0
root@mars:/home/demo# losetup /dev/loop0
/dev/loop0: [2065]:293509 (/home/demo/vbox.iso)
root@mars:/home/demo# losetup -l
NAME       SIZELIMIT OFFSET AUTOCLEAR RO BACK-FILE
/dev/loop0         0      0         0  0 /home/demo/vbox.iso
root@mars:/home/demo# losetup -j vbox.iso
/dev/loop0: [2065]:293509 (/home/demo/vbox.iso)
root@mars:/home/demo# losetup -d /dev/loop0
```

dd:数据块读写

- dd [option] 命令从标准输入或者文件中读取数据，然后进行相应的转换后，输出到标准输出或者文件中
  - if=IFILE : 从文件中读，如果IFILE为块设备时以“原始形式"读
  - of=OFILE：写入到文件中，OFILE为块设备时以”原始形式“写
  - bs=BYTES: 块大小为BYTES，缺省512字节，即读写时一次读写BYTES个字节
  - ibs=BYTES obs=BYTES: 在读的时候的块大小以及写的时候的块大小
  - count=N： 总共读写N块，实际读写的数据长 N*BYTES
  - seek=N： 跳过OFILE最前面N个obs大小的block再写
  - skip=N： 跳过IFILE最前面N个ibs大小的block再读
  - conv=CONVS: 说明如何转换(比如lcase,ucase进行大小写转换)，缺省不作转换

BYTES 后面还可以添加单位，可以是c(字节), w(2字节), b(512字节), kB(1000字节), K(1024字节)等

dd:数据块读写

dd if=/dev/sdx of=/dev/sdy	将整个硬盘sdx备份到另外一个硬盘sdy，潜在磁盘杀手，当心！！！
dd if=/dev/sda2 of=/path/to/image	将分区sda2备份到映像文件
dd if=/path/to/image of=/dev/sda2	将备份映像恢复到分区sda2
dd if=/dev/sdb of=mbr count=1 bs=512	备份磁盘前面512字节(即主引导纪录MBR)到指定文件
dd if=/dev/sr0 of=cd.iso	光盘中的内容制作成ISO映像
dd if=/dev/urandom of=/dev/sdc1	利用随机数据填充硬盘，销毁数据
dd if=FILE1 skip=2 ibs=1M count=1 of=FILE2 seek=1 obs=512	将FILE1文件中从第2M到第3M间的内容附加到FILE2的512字节开始的位置

如何创建映像文件?

- 利用dd命令直接从已有文件系统创建映像文件
  - dd if=/dev/sr0 of=cd.iso
- 利用genisoimage命令从目录创建一个ISO9660映像文件

  genisoimage -J -o /tmp/cd.iso cd_dir 使用Joliet格式的目录与文件名称
- 利用dd命令创建一个指定大小的文件，然后绑定到回环设备后创建文件系统

  dd if=/dev/zero of=file.img bs=1k count=10240 # 创建一个10M字节的文件

  mkfs file.img #等同于调用mke2fs来创建一个文件系统

  sudo mount file.img /mnt # 挂载文件系统

位桶(bit-bucket):

- /dev/zero 读时产生一系列的NULL字符，写入时数据丢弃
- /dev/null 读时产生EOF，写入时丢弃
- /dev/urandom: 读时立刻返回随机数
- /dev/random: 读时可能要等待噪声足够的情况下才返回随机数，比urandom更加安全