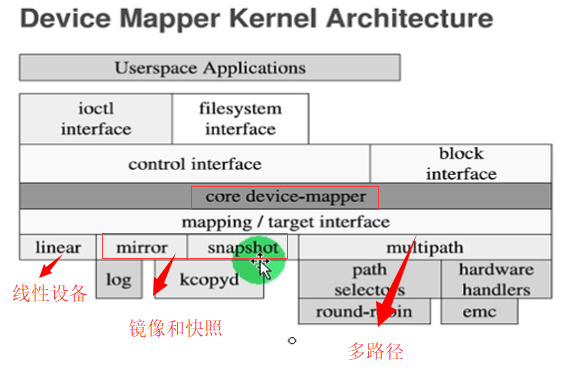
Linux基础



**linear** 线性,类似于jbod,连接多块磁盘,当作一个磁盘来使用

**mirro** 和 **snapshot** 是DM不具备的功能

**multipath** 在生成环境中或者对应可靠性要求高的环境当中使用

MD , DM

/dev/md#

meta device

DM device mapper 设备映射 ,将物理设备映射成逻辑设备

提供逻辑设备

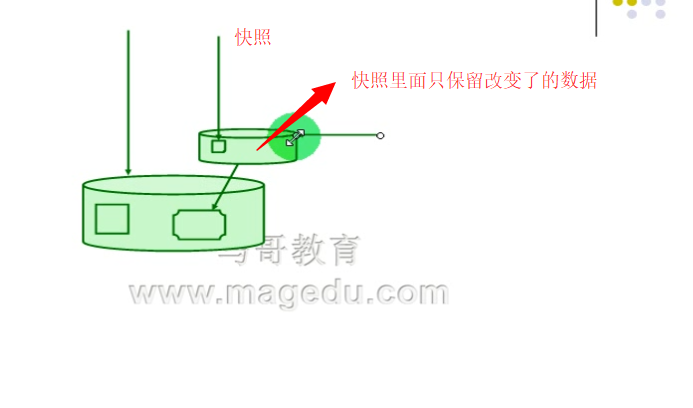
RAID,LVM2

DM : LVM2

快照 和软连接近似

多路径

快照的主要作用就是用来备份的,好处不言而喻

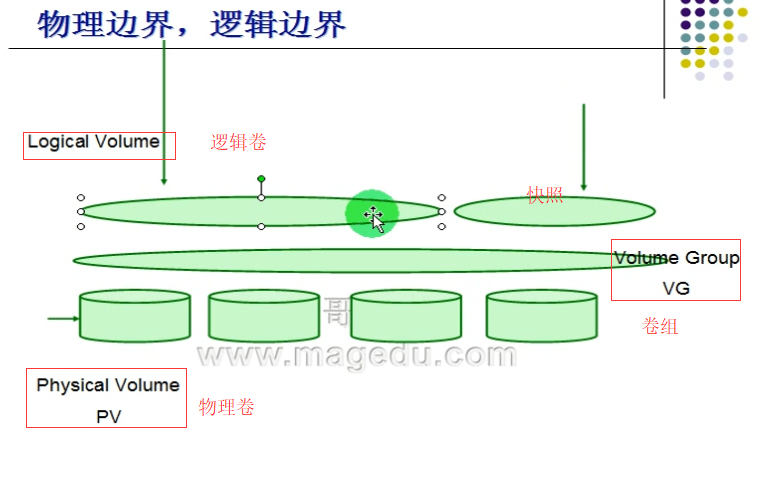


如果用cp dd 这样的命令来备份10T数据,不能下线备份,因此要提供服务又要备份,1天过去了,这样的速度很慢, 可能导致数据之间发生很变化. 快照就不会了,很块,保证数据之间一致性.

有的文件系统天生就有快照功能ZFS,

dm

逻辑设备 可以简单动态扩展缩减文件系统空间



**PV** Physical Volume 物理卷

底层物理设备

**VG** Volume Group 卷组

卷组可以包含多个物理卷,PV组合的大小就是VG的大小,VG不能够直接被上层使用,VG中存放一个一个的PE

Logical Volume 逻辑卷

只有逻辑卷才能被上层使用,一个VG可以划分多个逻辑卷来使用, 逻辑卷加起来的不能超过VG的大小,可以动态伸缩.扩展.

每一个逻辑卷都有两个边界,物理边界和逻辑边界.意味着每个逻辑卷就是一个文件系统,也可以对逻辑卷创建快照,必须要预留空间来放快照.快照也是一个访问入口

PE Physical Extend 物理盘区 逻辑存储单位

物理卷只有加入卷组才有PE,在创建VG的时候就要指定PE,PE大小就是VG中PE的大小

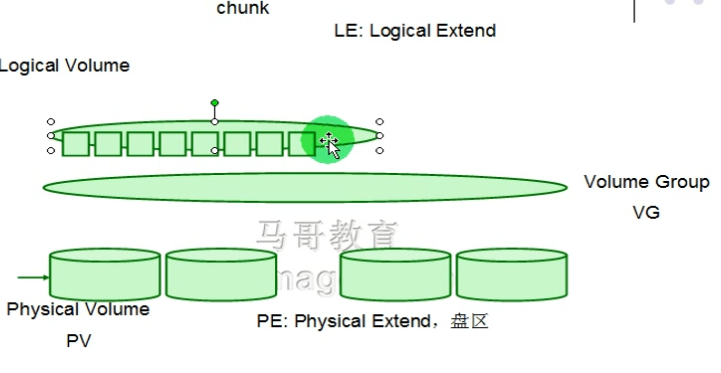
LE Logical Extend 逻辑盘区

站在逻辑盘区来说,就是LE, LE=PE

PE和LE只是不同角度上的说法

逻辑卷中的LE也可以做成镜像,但是不建议,一般底层物理RAID,在RAID上创建逻辑卷

逻辑卷底层可以是多块磁盘,也可以是分区,也可以是RAID



## pv 物理卷

**pvcreate** 创建PV

**pvremove** 移除PV中的元数据

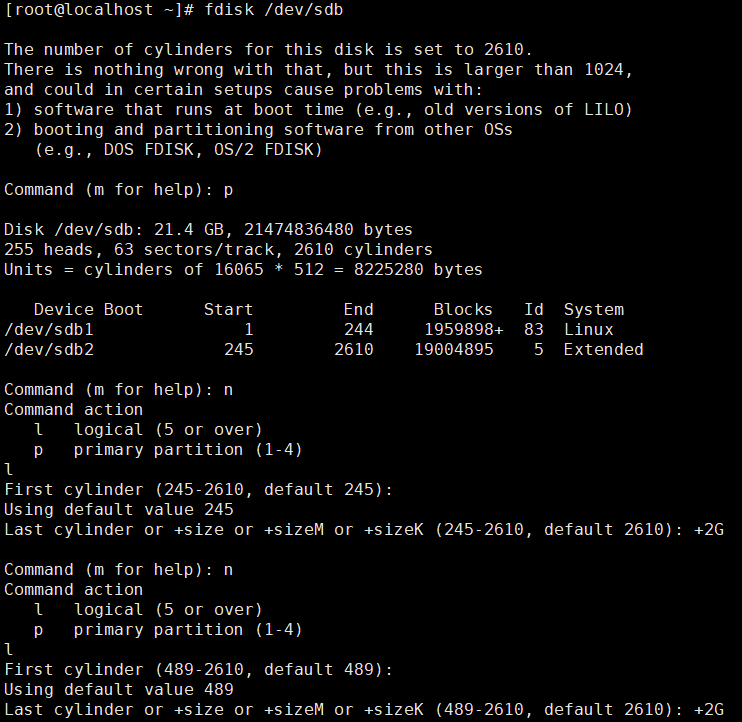
**pvscan** 扫描PV

**pvdisplay**  显示PV详细信息

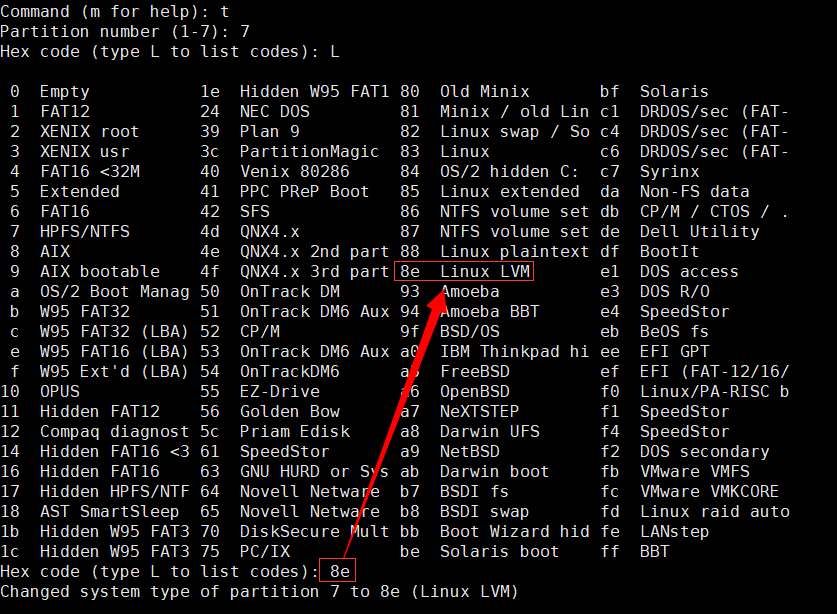
**pvmove** 移动PV

创建一个6G的逻辑卷

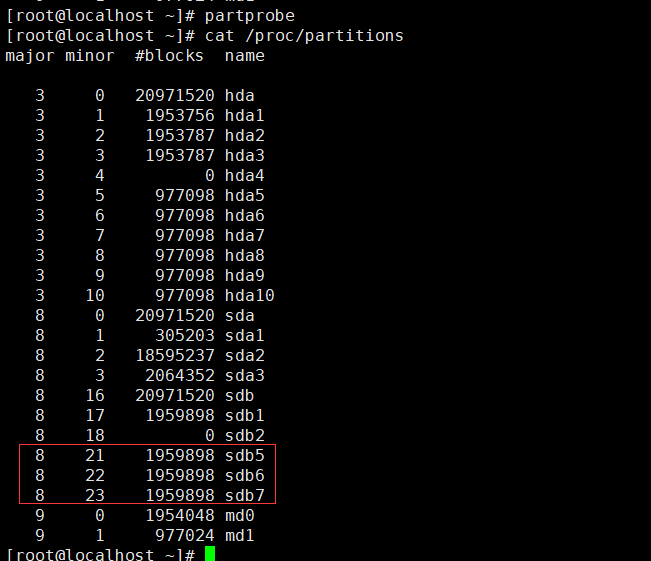
首先创建逻辑卷分区类型



调整分区类型 8e 逻辑卷类型



保存,内核读取识别

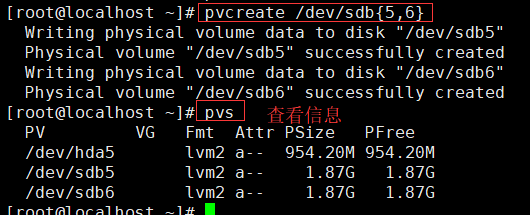


fdisk 命令组多支持到15个分区

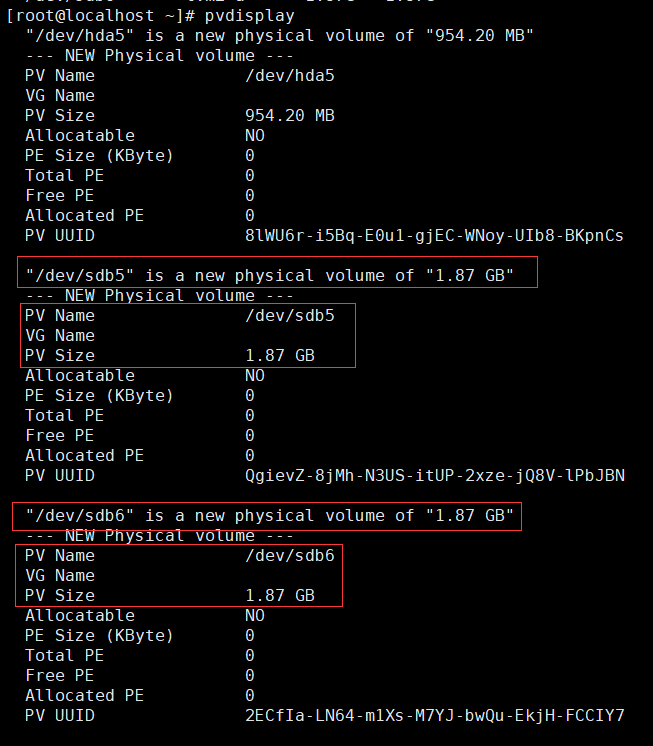
接下来创建PV

pvcreate /dev/sda{10,11} 创建sda10,sda11为物理卷

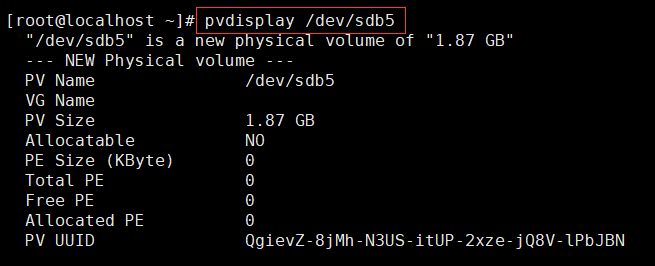
pvs 查看信息



pvdisplay 所有PV详细信息



pvdisplay /dev/sda10 只显示pv sda10的详细信息



## vg 卷组

**vgcreate**  创建VG

-s # : 指定PE大小,默认是4M

**vgrmove** 移除VG

**vgextend** 扩展VG

**vgreduce** 缩减VG

**vgs**  查看VG信息

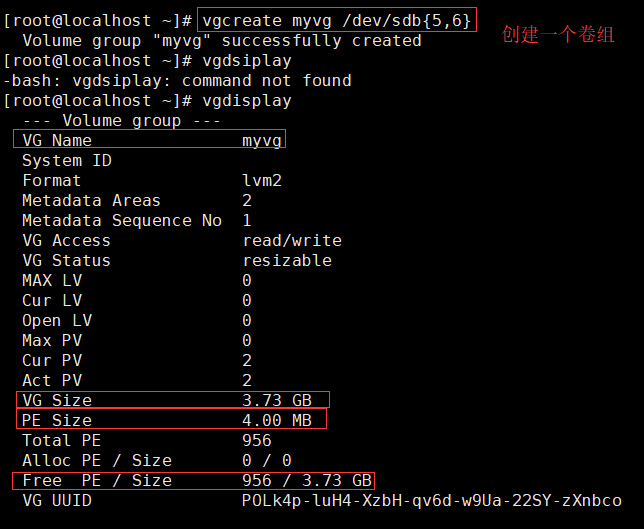
**vgdisplay** 显示VG详细信息

**vgscan** 扫面VG

vgcreate myvg /dev/sda{10,11}

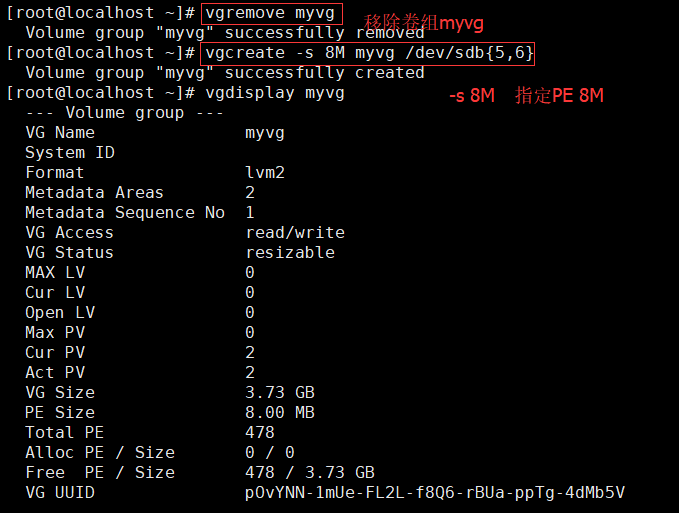
把sdb5,sdb6创建一个叫myvg的卷组

PE是4M



vrgremove myvg

移除myvg卷组



如果要换一个pv,该如何处理?

pv上有数据,那么数据先移动到其他PV上,避免数据丢失

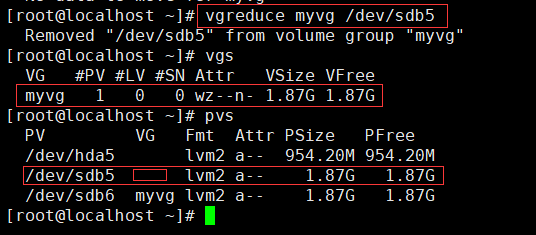
pvmove /dev/sda11

把sdb6上的数据移动到其他逻辑卷上sdb5

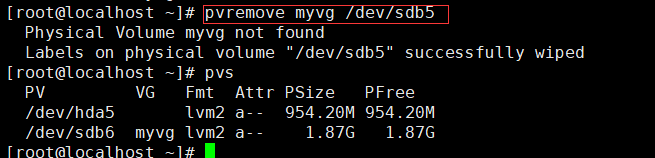


vgreduce myvg /dev/sda11

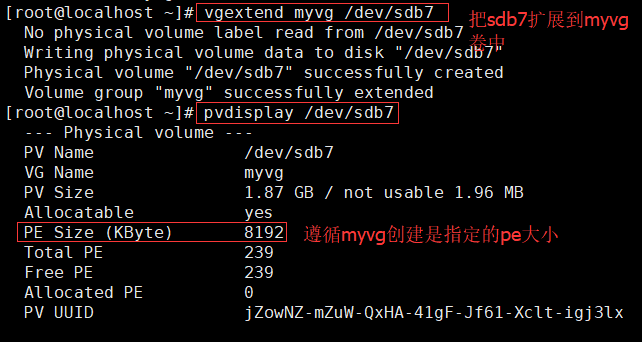
从myvg这个卷组上移除 sda11就不属于myvg这个卷组了



接下来就移除pv sda11



把sdb7扩展到myvg卷组



## lv 逻辑卷

**lvcreate**

-n : 指定名字

-L #/k/M/G/T..: 指定大小

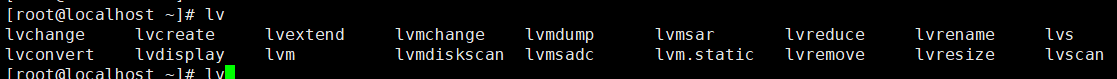
**lvremove**

**lvextend**

**lvreduce**

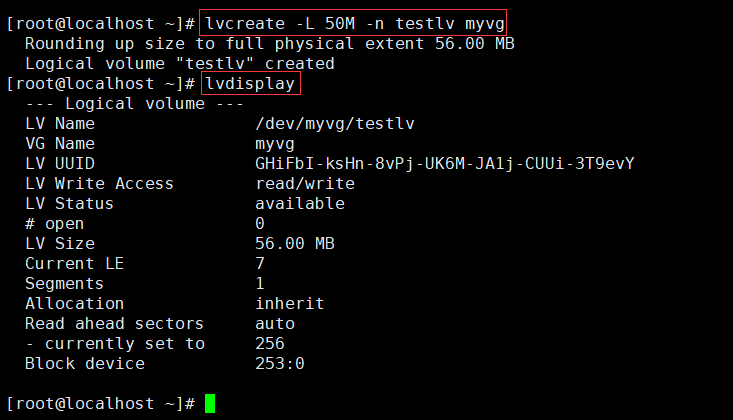
**lvs**

**lvdisplay**

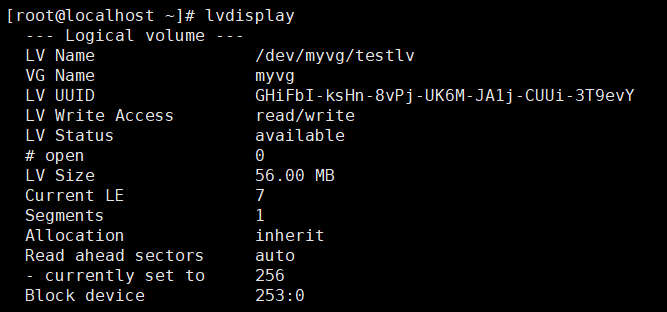


lvcreate -L 50M -n testlv myvg

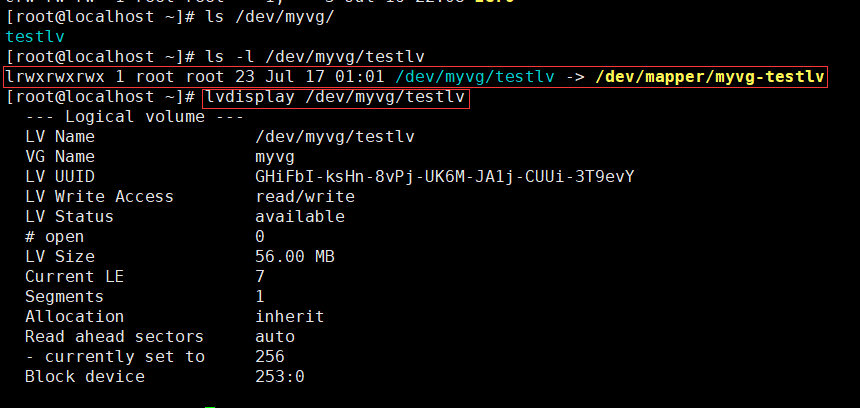
大小50M 名称为testlv 管辖myvg卷组



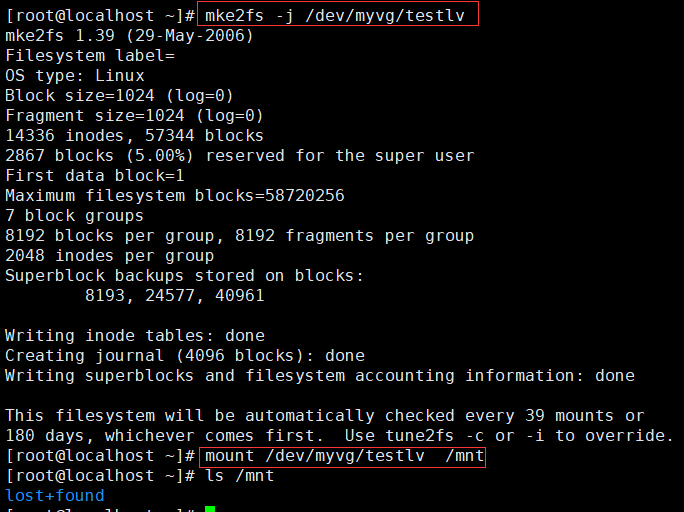
查看所有的lv详细信息



只查看一个逻辑卷的详细信息

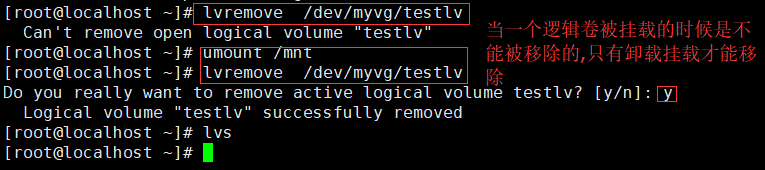


格式化并挂载使用





可以看出创建逻辑卷名称不能含有-

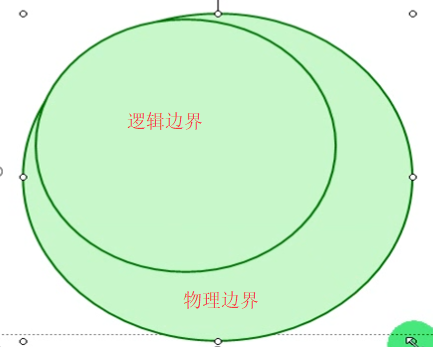


练习:创建一个由两个物理卷组成大小为20G的卷组muvg,要求PE大小16M,而后在此卷组中创建一个大小为5G的逻辑卷lv1,此逻辑卷要能在开机的时候自动挂载至/users目录,且支持ACL功能.

## 边界

分区: 物理边界

文件系统 : 逻辑边界



物理边界 > 逻辑边界

**扩展**: **先扩展物理边界在扩展逻辑边界**

**缩减**: **先缩减逻辑边界在缩减物理边界**

## 扩展逻辑卷

**1.扩展逻辑卷的物理边界**

**lvextend**

-L [+] # /dev/xxx

原来2G如果要扩展至5G有两种写法

lvextend -L +3G / lvextend -L 5G

**2.扩展逻辑卷的逻辑边界(文件系统边界)**

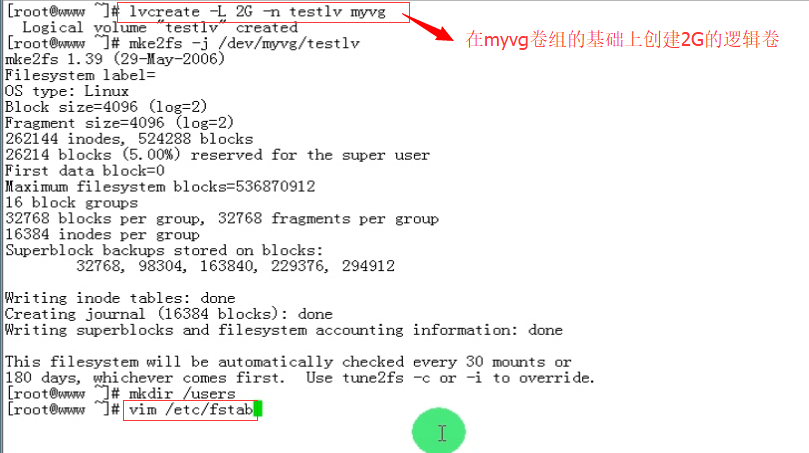
**resize2fs**

resize2fs /dev/xxx 5G

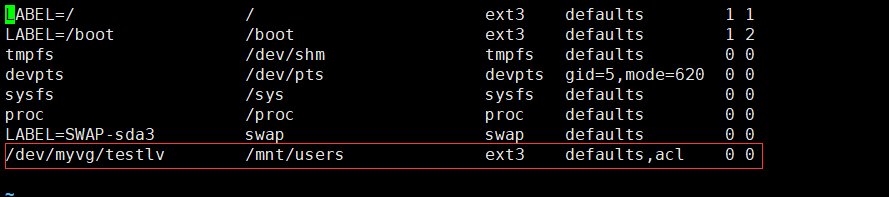
扩展的逻辑边界不能超过物理边界的大小,超过会导致文件系统崩溃

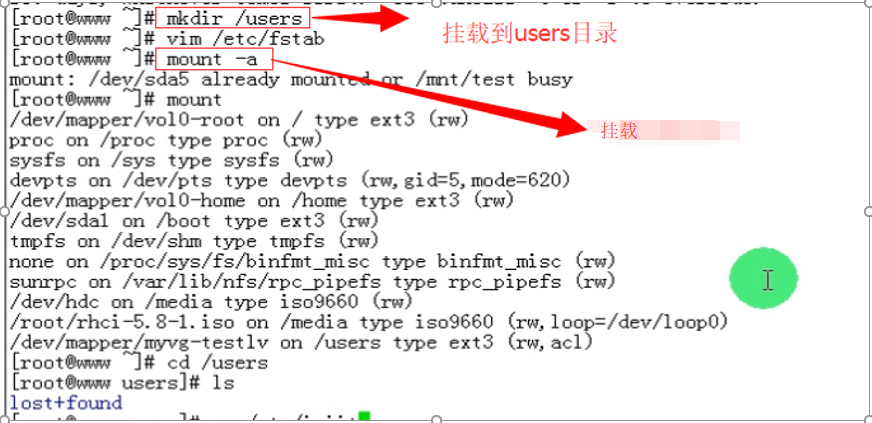
resize2fs -p /dev/xxx

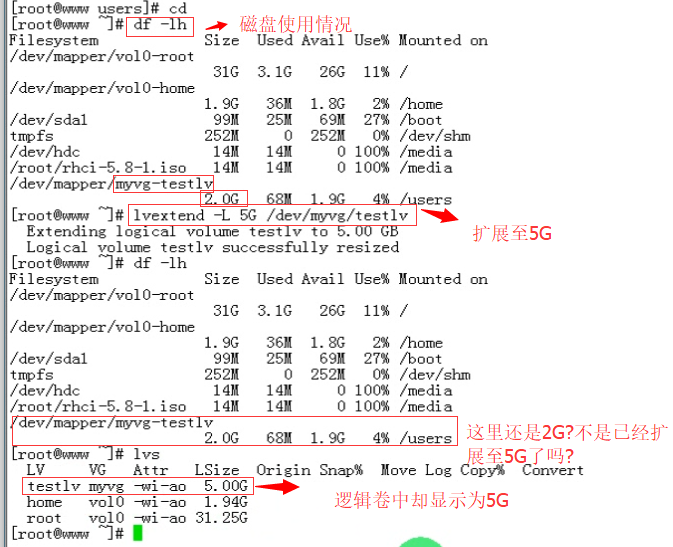
-p 就不用指定大小了,创建逻辑边界和物理边界一样大



配置到fstab,启动就挂载

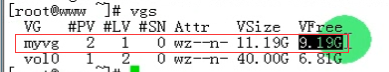






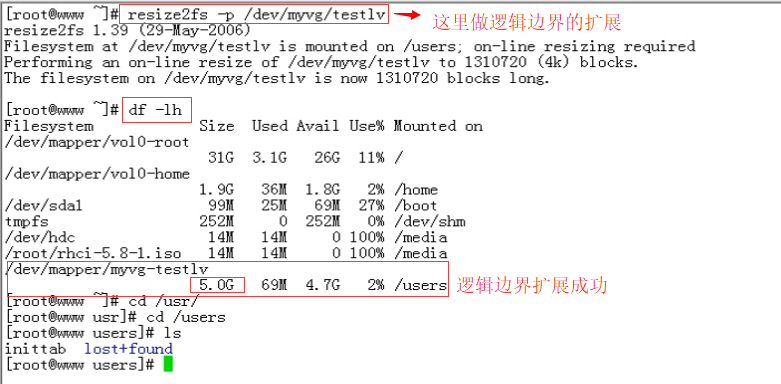
这里2G展至5G是扩展的物理边界,而不是逻辑边界

在扩展之前要查看 myvg 的空闲空间是否满足扩展的大小,这里需要3G



可以看出 myvg的空闲空间还有9.19G,满足逻辑卷扩展条件

做逻辑边界的扩展



好处: 扩展一个逻辑卷,并不影响原有的文件数据,也很安全,风险小,也不用卸载挂载

## 缩减逻辑卷

**注意:**

1.不能在线缩减,缩减前得先卸载挂载

2.确保缩减后的空间大小依然能够存储原有的文件数据

3.为了足够可靠,在缩减之前应该先强行检查文件系统,以确保文件系统处于一致性状态

resize2fs

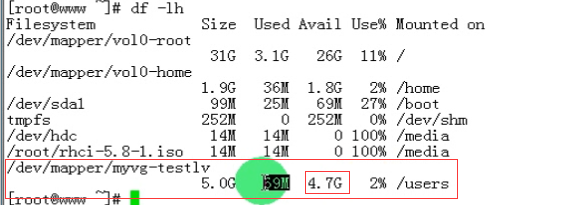
/dev/xxx

lvreduce

-L [-]# /dev/xxx

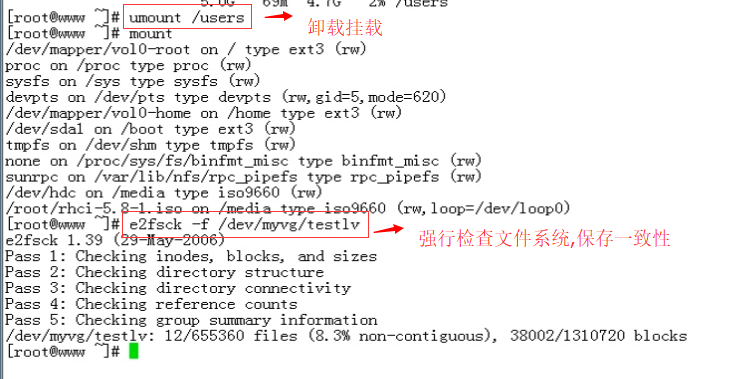
重新挂载

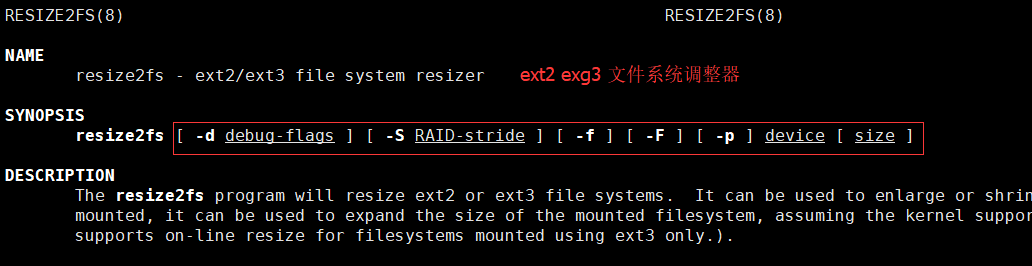
1.



已经使用59Ｍ，缩减的时候要保证缩减后的空间大于59M

2.

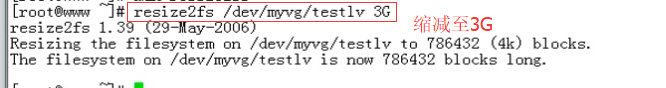




3.

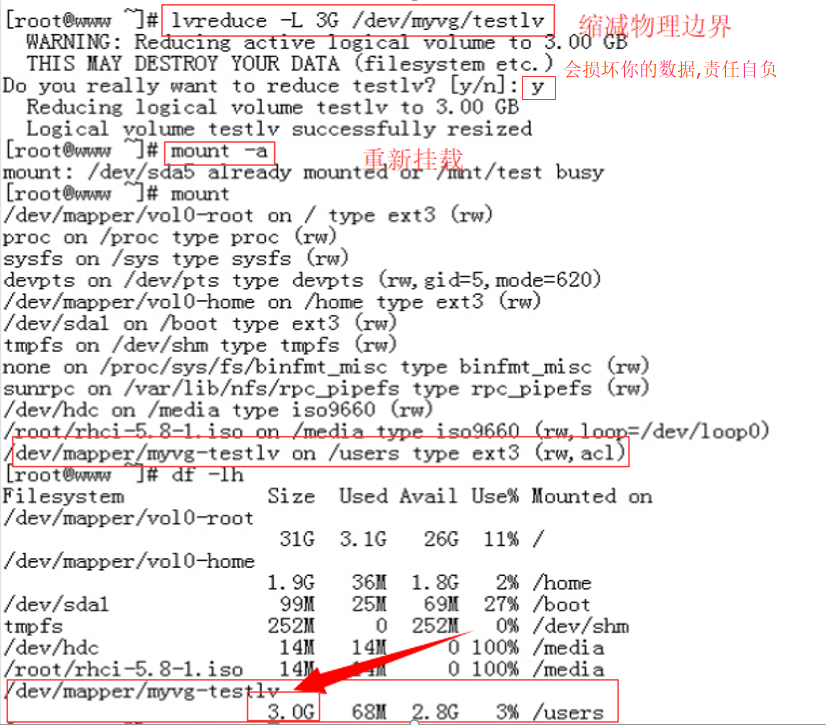
resize2fs /dev/myvg/testlv 3G

这里先缩减逻辑边界



4. 缩减物理边界

**缩减有风险,操作需谨慎**



## 快照卷

**注意:**

**快照卷主要目的是备份**

1.生命周期为整个数据时长,在这段时长内,数据的增长量(变化量)不能超出快照卷大小最好快照卷和原卷一样大小,

2.快照卷应该是只读的

3.跟原卷在同一个卷组内,vg

lvcreate

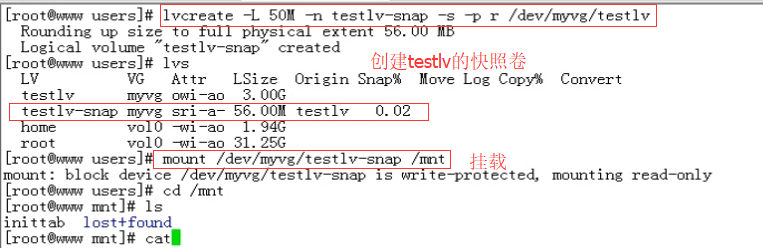
-s 创建快照卷

-p r|w 一般创建为只读,快照不允许修改

lvcreate -L # -n kzname -p r /dev/xxx(逻辑卷)

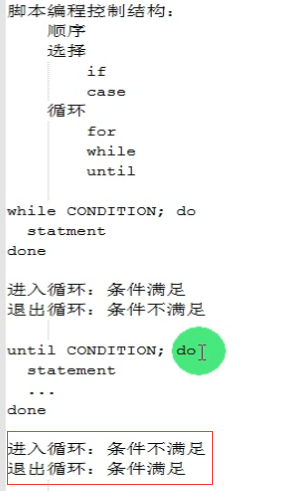
lvcreate -L 50M -n testlv-snap -s -p r /dev/myvg/testlv

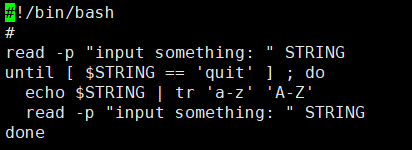
为逻辑卷testlv 创建一个名字为testlv-snap 并且大小50M的只读快照



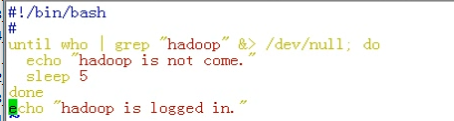
一个进程inittab进行修改,另一个进程通过快照的入口访问,另一个进程访问的数据,是创建快照那一刻的数据,因此,这属于完全备份,还原的时候,数据只是快照那一刻的数据,而不能还原到数据出错那一刻的状态,如果要还原到出错前那一刻,需要结合**增量备份,差异备份**

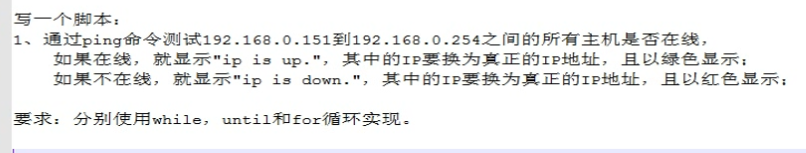
## until循环











ping

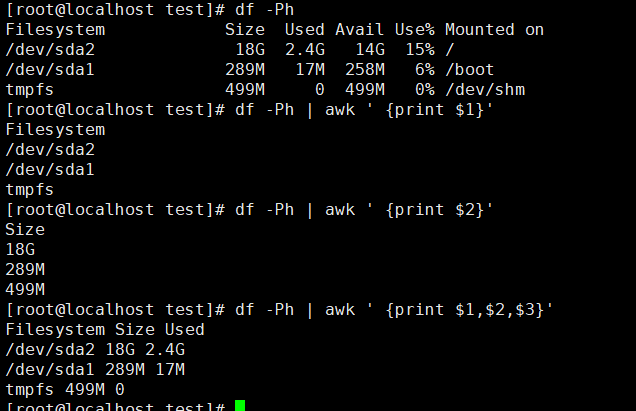
-c 只定ping几次

-W 限定时长

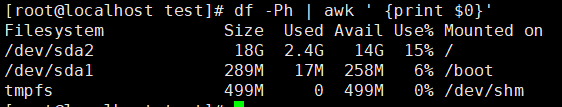
## awk

awk 一次读一行,按照指定的模式进行分割,被分割下来的变量,用$1,$2,$3…来保存,如果要做处理,使用变量就可以了

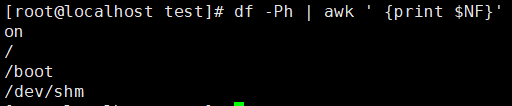
awk ‘patter{action}’ file



$0 表示所有字段

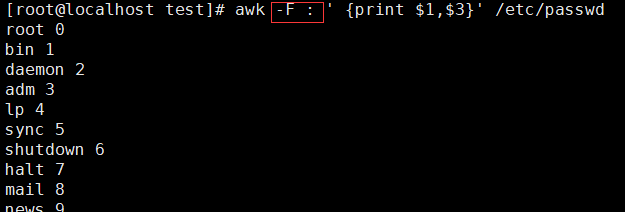


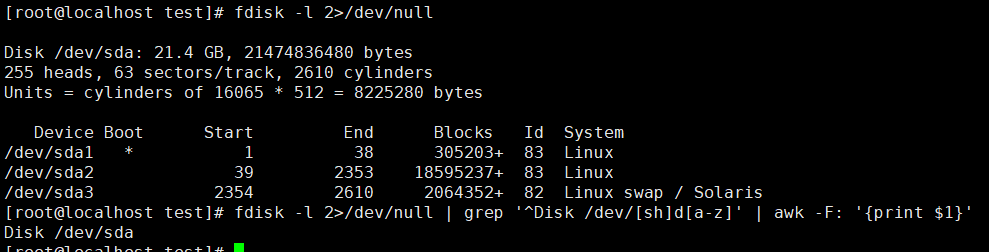
$NF (number of filed) 表示字段个数 $NF = $4 最后一个

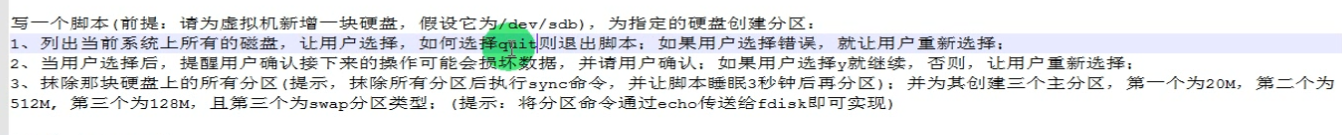


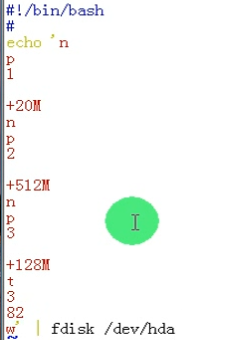
awk -F : ‘{print $1,$2}’

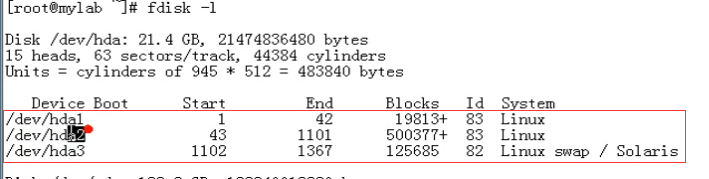
-F : 指定分隔符

















#!/bin/bash

#

echo "initial a disk..."

echo -e "\033[31mwaring:\033[0m"

fdisk -l 2>/dev/null | grep -o "^Disk /dev/[sh]d[a-z]"

read -p "your choice:" PARTDISK

if [ $PARTDISK == 'quit' ];then

echo 'quit'

exit 7

fi

until fdisk -l 2>/dev/null | grep -o "^Disk /dev/[sh]d[a-z]" | grep "^Disk $PARTDISK$" &>/dev/null ;do

read -p "Wrong option, your choice aging: " PARTDISK

done

read -p "will destroy all data,continue: " CHOICE

until [ $CHOICE == 'y' -o $CHOICE == 'n' ];do

read -p "will destroy all data,continue: " CHOICE

done

if [ $CHOICE == 'n' ]; then

echo "quit"

exit 9

else

dd if=/dev/zero of=$PARTDISK bs=512 count=1 &>/dev/null

sync

sleep 3

echo 'n

p

1

+20M

n

p

2

+512M

n

p

3

+128M

t

3

82

w' | fdisk $PARTDISK &> /dev/null

partprobe $PARTDISK

sync

sleep 2

mke2fs -j ${PARTDISK}1 &>/dev/null

mke2fs -j ${PARTDISK}1 &>/dev/null

mkswap ${PARTDISK}3 &>/dev/null

fi