目录：

[1、Linux系统磁盘分区知识指南 2](#_Toc418499242)

1.1磁盘存储逻辑结构简图 2

1.1.1主引导记录MBR 3

1.2磁盘分区的知识 4

1.2.1主分区primary 4

1.2.2扩展分区extended 5

1.2.3逻辑分区logical 5

1.2.4磁盘分区的注意事项 5

1.2.5有关磁盘分区的几个问题范例 5

1.3如何选择文件系统类型 7

1.3.1文件系统选型简单介绍 8

1.4分区命令（fdisk） 11

1.5分区命令parted 14

1.6格式化命令mkfs 15

2、Linux网络基础20

2.1用户管理回顾 11

2.2 /etc/skel 目录 13

2.3 /etc/default/useradd 文件 14

2.4 /etc/login.defs文件 15

2.5 useradd命令 15

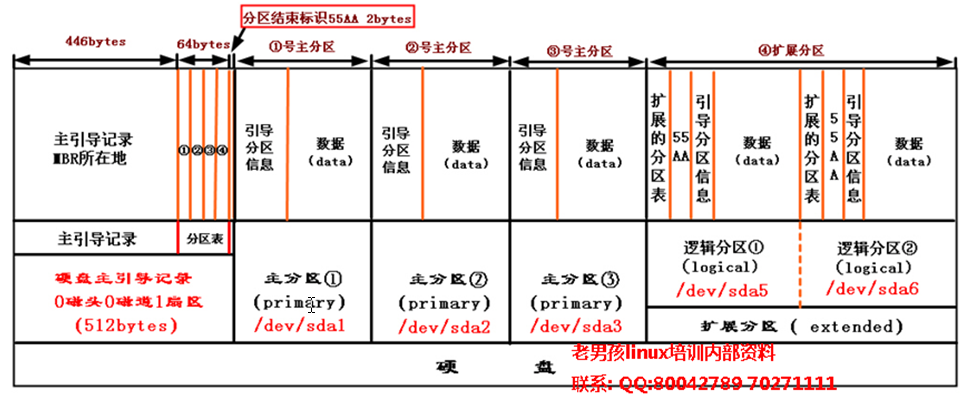
2.6用户查询相关命令id、finger、users、w、who、last、lastlog、groups 22

3、Rsync相关知识23

4、Rsync+inotify组合的起源39

1. **Linux系统磁盘分区知识指南**

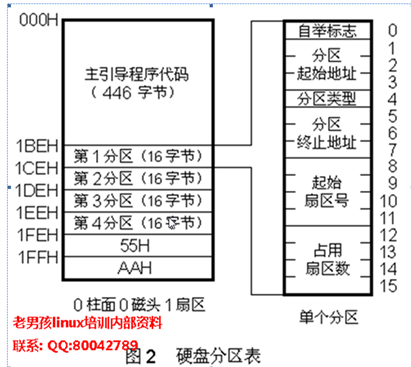
**1.1****磁盘存储逻辑结构简图**

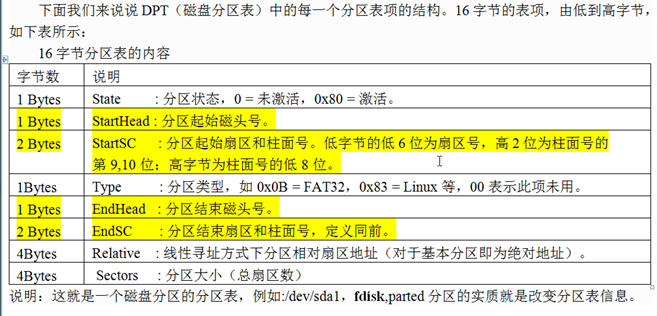


磁头读数据的原理是从0磁头0磁道1扇区开始，1扇区的大小为512字节，其中前446字节就是主引导记录（不光是Linux，Windows也是一样的），之后的64字节是分区表。一共4个主分区和扩展分区，每个分区为16字节，最后剩余两个字节是分区的结束标识，这个就是第一扇区里的东东。

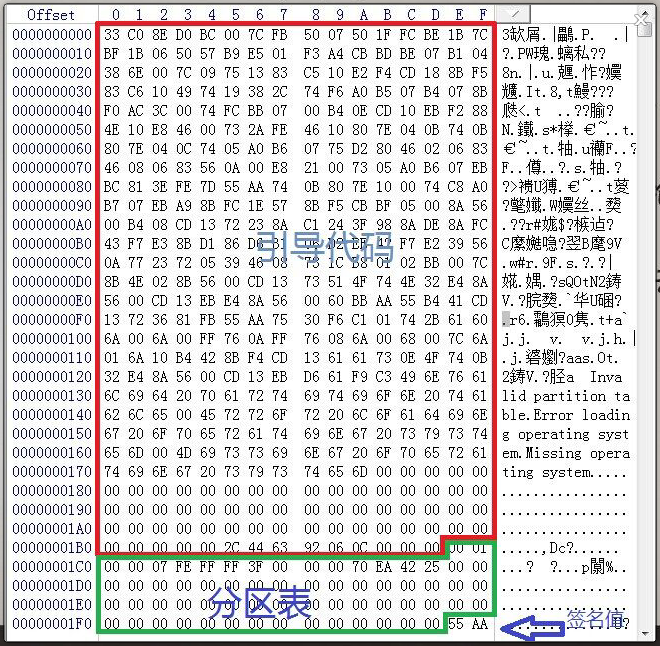
逻辑分区是在扩展分区中的，逻辑分区的分区表可有多个。扩展分区不能直接使用，逻辑分区的标号是从5开始的。

主引导扇区在整个硬盘里是唯一的，也就是一个硬盘只能有一个，下面是0磁道1扇区的放大图示：





* + 1. 主引导记录MBR



磁盘的0磁头0磁道1扇区，是硬盘上的一个非常重要的位置，在这个扇区中，存放有硬盘的主引导记录MBR（master boot record）和硬盘的分区表DPT（disk partition table）。

说明：可以用dd命令主引导记录的内容：

[root@moban ~]# dd if=/dev/sda of=mbr.bin bs=512 count=1

记录了1+0 的读入

记录了1+0 的写出

512字节(512 B)已复制，0.0619964 秒，8.3 kB/秒

[root@moban ~]# file mbr.bin

mbr.bin: x86 boot sector; GRand Unified Bootloader, stage1 version 0x3, boot drive 0x80, 1st sector stage2 0x4e68, GRUB version 0.94; partition 1: ID=0x83, active, starthead 32, startsector 2048, 409600 sectors; partition 2: ID=0x82, starthead 159, startsector 411648, 1572864 sectors; partition 3: ID=0x83, starthead 135, startsector 1984512, 14792704 sectors, code offset 0x48

[root@moban ~]# od -xa mbr.bin

0000000 48eb 1090 d08e 00bc b8b0 0000 d88e c08e

k H dle dle so P < nul 0 8 nul nul so X so @

0000760 0000 0000 0000 0000 0000 0000 0000 aa55

nul nul nul nul nul nul nul nul nul nul nul nul nul nul U \*

0001000

**1.2 磁盘分区的知识**

为什么分区？给磁盘分区和买房子要隔出卧室和客厅是一样的，可以在不同的分区存放不同的数据，一个分区满了不会对其他的分区大小造成影响。所以，一般情况下载工作中我们会根据具体的业务需求对硬盘进行分区，当然，部分区直接格式化使用一块硬盘存数据也是可以的。

**磁盘分区的重点：**

1、给磁盘分区的实质就是针对上述0磁头0磁道1扇区的前446字节后面接下来的64bytes的分区表进行设置，即主要是划分起始以及结束磁头号，及扇区号和柱面号。

2、给磁盘分区工具有fdisk（适合小于2T的磁盘分区），parted（擅长给大于2T的磁盘分区，可以对小于2T的磁盘分区），fdisk首选，只有大于2T时才去选parted。

补充：（一台服务器6块600G的磁盘，raid5后，总大小3T，此时无法装系统的，解决办法是做raid5后，不要重启系统，而是再raid界面，继续分1个小的虚拟磁盘vd 200G，用这个200G的虚拟磁盘装系统，装完系统后再把剩余的2.8T分区通过parted）。

3、一块磁盘的分区表仅有64bytes大小，每个分区表要占用16个字节，因此一块磁盘仅支持四个分区表信息，即主分区+扩展分区的总量不超过4个。

4、磁盘分区是按照柱面（cylinder）来划分的。（从磁盘读写原理角度可以理解）

5、扩展分区不能直接使用的，还需要扩展分区的基础上创建逻辑分区才行。

6、扩展分区有自己的分区表，因此扩展分区下面的逻辑分区可以有多个。

IDE磁盘逻辑分区打开编号可以是5-63，SATA编号5-15

**1.2.1 主分区primary**

一般来说，主分区是磁盘上必须存在的分区，一般为磁盘的第一个分区，可以在这个主分区上面安装操作系统，在一个磁盘上最多只能有四个主分区，因为每个主分区都会在主引导区中生成分区表DPT，每个分区表需要占用大小16bytes，当初硬盘设计时，只给分区表留下了64bytes的存储空间，所以，就只能有四个主分区！生产环境中，四个主分区有可能不够使用，那么怎么来扩展呢？要使用扩展分区来扩展！

一个磁盘想分成6个区，主分区和扩展分区的数量是多少：

1p+1e,2p+1e,3p+1e。

装操作系统：

/boot 主分区

Swap 主分区

/ 主分区

**1.2.2 扩展分区extended**

严格来说，扩展分区布恩那个算一个正常的分区，而是一个链接，起到一个指向的作用，可以再扩展分区内建立逻辑分区（logical），从上面逻辑结构图中可以看到，扩展分区就像一个虚拟出来的一个小硬盘一样，但是不同的是，没有MBR，而只有扩展分区表，而且这个扩展分区表示没有64bytes的闲置的，所以可以在扩展分区划分多个逻辑分区。

一块硬盘只能存在一个扩展分区，并且扩展分区不能直接存放数据，扩展分区受限于操作系统。

**1.2.3 逻辑分区logical**

不能再磁盘中单独直接划分逻辑分区（fdisk），逻辑分区必须存在于扩展分区内。在扩展分区内可以划分多个逻辑分区，逻辑分区的编号从数字5开始，在这个口占分区内可以划分多个逻辑分区（IDE磁盘大概编号可以是5-63）。SATA（编号5-15）。

实际应用：主分区和逻辑分区，都可以用。一般系统安装用主分区，存放数据都可以。

**1.2.4 磁盘分区的注意事项：**

一块硬盘的分区方式只能为如下组合之一：

1. 任意多个主分区，但要求1≤主分区数量≤4.

例如：一个硬盘可以分为4个主分区、3个主分区、2个主分区、1个主分区。

1. 扩展分区可以和主分区组合，但要求2≤（主分区+扩展分区）数量≤4）

例如：3个主分区+1个扩展分区、2个主分区+1个扩展分区、1个主分区+1个扩展分区。当总分区的数量大于4个的时候，必须提前分一个扩展分区。

1. 如果要分成四个磁盘分区的话，那么最多就是可以：

P+P+P+P

P+P+P+E

其中的P为主分区，E为扩展分区。

提示：

由于MBR及磁盘分区表大小的限制，如果对磁盘分配了4P或3P+E后，即使硬盘还有剩余未分的空间，那么，这部分空间也无法分区使用了（因为无分区可分了，分区表空间已被沾满）。因此，在做分区规划时，如果业务需要，在分区之前就已经明确要超过4个分区，那么可选3P+1E的分区方式，并且在最后一个扩展分区要将所有的剩余空间都分配给这个扩展分区。当然，也可以采用1P+1E、2P+1E的分区方式。

**1.2.5 有关磁盘分区的几个问题范例：**

范例1：如果要将一块大硬盘这哪是分成四个分区，同时，还希望有其他的空间可以在未来需要的时候在进行分区，该如何分区？

3P+1E(1L)剩余空间保留

2P+1E(2L)剩余空间保留

1P+1E(3L)剩余空间保留

范例2：我可不可以仅分一个主分区和一个扩展分区呢？

可以，基本上逻辑分区可以有5-63编号，因此，可以仅分一个主分区和一个扩展分区，并且将所有其他的剩余空间都给扩展分区，然后在扩展分区上面利用逻辑分区来进行其他的更多划分规划。

范例3：假如有一块SAS硬盘，我想把磁盘分成6个可以使用的磁盘分区，那么每个磁盘分区在Linux系统下的数字编号时多少？

说明：  
由于硬盘在primary+extended最多可以有四个，因此，在Linux系统下，会默认将1-4的分区编号保留下来给主分区和扩展分区和扩展分区使用，逻辑分区不可以占用1-4分区编号。

因此，由于要求的分区总数为6个，因此，必须要有一个扩展分区，主分区可以有1-3个，但是无论分几个主分区，逻辑分区的编号都是从5开始计算。

如果是1P+1E，则分区编号为：

/dev/sda1，/dev/sda5，/dev/sda6，/dev/sda7，/dev/sda8，/dev/sda9

如果是2P+1E，则分区编号为：

/dev/sda1，/dev/sda2，/dev/sda5，/dev/sda6，/dev/sda7，/dev/sda8

如果是3P+1E，则分区编号为：

/dev/sda1，/dev/sda2，/dev/sda3，/dev/sda5，/dev/sda6，/dev/sda7

**企业生产场景Linux系统分区方案说明：**

**常规分区方案：**

方案1：集群架构的某个节点，数据有多份或者不重要。

/boot 100MB Linux引导程序

swap 物理内存的1.5倍，当内存大于等于8GB时，给8GB即可。

/ 剩余硬盘大小 （相当于Windows只分C盘）

方案2：数据库及存储，有大量重要的数据。

/boot 100MB

/ 50-200GB

swap物理内存的1.5倍，当内存大于等于8GB时，给8GB即可。

/data 剩余硬盘大小，放数据库及存储数据的。

方案3：门户级别或大网站：

/boot 100MB

swap物理内存的1.5倍，当内存大于等于8GB时，给8GB即可。

/ 50-200GB

剩余空间保留，不在进行分区，将来分给哪个部门，拿到服务器的部门，就自己根据需求再分。

生产场景集群架构不同节点分区方案建议：

<http://oldboy.blog.51cto.com/2561410/634725>

http://oldboy.blog.51cto.com/2561410/629558

**硬盘分区的设备名：**

在Linux系统中，是通过设备名来访问设备的，设备名存放在/dev目录中。

设备名称的定义规则如下，其他的分区可以以此内推：

系统的第一块IDE接口的硬盘称为/dev/hda.

系统的第二块IDE接口的硬盘称为/dev/hdb

系统的第一块SCSI接口的硬盘称为/dev/sda.

系统的第二块SCSI接口的硬盘称为/dev/sdb.

SATA。SAS都是sd开头。

每个分区则使用磁盘名称加相对应的数字编号表示，如：

一系统的第一块IDE接口硬盘的第1个分区称为/dev/hda1

一系统的第一块IDE接口硬盘的第5个分区称为/dev/hda5

一系统的第二块SCSI接口硬盘的第1个分区称为/dev/sdb1

一系统的第二块SCSI接口硬盘的第5个分区称为/dev/sdb5

注意：

1. 分区数字编号1-4留给主分区或者拓展分区使用，逻辑分区只能从5开始

2. IDE硬盘设备名均以/dev/hd开头，不同硬盘编号依次为/dev/had,dev/hdb.dev/hdc

3. SCSI/SAS/SATA/USB接口硬盘的设备名均以/dev/sda开头不同编号依次是/dev/sda，dev/sdb

4. 特别强调：SAS/SATA为当前2010-2012互联网生产环境主流硬盘设备名，SSD固态硬盘，由于是电子设备，性能很高，也逐渐被应用。

5. Linux环境下没有盘符的概念，要对磁盘设备进行操作，需要使用磁盘设备名；读取磁盘设备内容，需要把创建在分区或者逻辑卷上的文件系统挂载到指定目录上

6. 在Linux系统上划分了分区之后，还要在分区上创建件系统

Linux系统对分区的要求：

a） 最少要有/以及swap两个分区（其实swap可以没有）、

b） swap交换分区的作用：虚拟内存，swap分区的大小=1.5\*物理内存容量，如果内存大于5-16G那么8-16G作为swap即可

c） 建议设置独立的/boot分区

Linux引导分区，存放系统引导文件，如Linux内核等，所有文件大小一般只有几十M。因此该分区设置100-200M

安装即采用常规的生产线上一般服务器分区方式：/boo，swap，/三个分区

* 1. **如何选择文件系统类型**

文件系统是对一个存储设备上的数据和元数据（统计数据的数据）进行组织的一种机制。文件系统是在一个磁盘（包括光盘、软盘、闪盘及其他存储设备）或分区上组织文件的方式和方法，常见文件系统如：ext2、ext3、etx4、NTFS、FAT、fat32等。文件系统是组织存储文件或数据的方法，目的是易于查询和存取数据。因此，如果磁盘上没有文件系统也就无法存储数据了，因此，在磁盘分区后能够使用之前必须建立对应的文件系统才行（类似买了电脑不装系统无法用）

文件系统是基于存储设备的，比如硬盘或光盘，并且包含文件物理位置的维护；文件系统也可以说是虚拟数据或网络数据存储的方法，比如NFS，MFS，GFS。目前，Unix/Linux的文件系统有很多种实现，例如：UFS（基于BSD的Unix文件系统）、ext2、ext3、ext4、zfs、xfs、reiserfs等等。

**文件系统小结：**

1. 文件系统是对一个存储设备上的数据和元数据进行组织的一种机制。
2. 分区必须格式化创建文件系统才能存放数据。
3. 一个分区只能有一种文件系统。
4. Linux下常见文件系统ext2、ext3、ext4、zfs、xfs(centos7)、reiserfs（单独安装）。
   * 1. **文件系统选型简单介绍**
5. SAS/SATA硬盘文件系统选择：
6. reiserfs大量小文件业务首选reiserfs（100K以内）。效率高
7. xfs 有的门户的数据库MySQL业务。
8. etx4视频下载，流媒体，数据库，小文件业务也OK，可以用默认的。
9. ext2 没有日志，蓝汛的cache业务，CDN网站加速服务的。

常规应用:

对于centos5.x系列，常规应用就选默认的ext3文件系统即可。

对于centos6.x系列，常规应用就选默认的ext4文件系统即可。

对于centos7.x系列，常规应用就选默认的xfs文件系统即可。

注意：如果服务器数量少怎么选择都可以。如果海量服务器，考虑维护成本和性能，选一个折中。

**选择小结：**

1. centos默认的文件系统都是比较优秀的。
2. xfs，reiserfs需要单独安装，默认不支持的，海量服务器使用多种文件系统维护可能不方便。
3. 大并发并不是过多的做磁盘系统的优化，而是再架构前端增加缓存，存储数据库的前段增加缓存，这是架构师的能力。
4. SSD文件系统选择

Ext4/reiserfs可以作为SSD文件系统，但未对SSD做优化，不能充分发挥SSD性能，并影响SSD使用时间。

Btrfs对SSD做了优化，mount通过参数启用，但btrfs仍处于试验阶段，生产环境谨慎使用。

JFFS2/Nilfs2/YAFFS是常用的flash file system，在嵌入式环境广泛应用，建议使用。性能目前还未做测试评估。

**在Linux中制作文件系统**

用类似如下命令在Linux中制作并优化reiserfs文件系统：

#/sbin/mkreiserfs/dev/sda2

用类似如下命令在Linux中制作xfs文件系统：

#mkfs –t xfs –f /dev/sda2

注意ext3其实简单的就是激活了记账的ext2fs。当创建该文件系统时，对mke2fs使用-j选项：

#/sbin/mke2fs –j /dev/sda2

[root@moban ~]# ls /lib/modules/2.6.32-573.el6.x86\_64/kernel/fs/

autofs4 dlm fat jffs2 nls

btrfs ecryptfs fscache lockd squashfs

cachefiles exportfs fuse mbcache.ko ubifs

cifs ext2 gfs2 nfs udf

configfs ext3 jbd nfs\_common xfs

cramfs ext4 jbd2 nfsd

<http://www.ibm.com/developerworks/cn/linux/filesystem/ReiserFS/intro/>

此篇博文有详细的讲解。

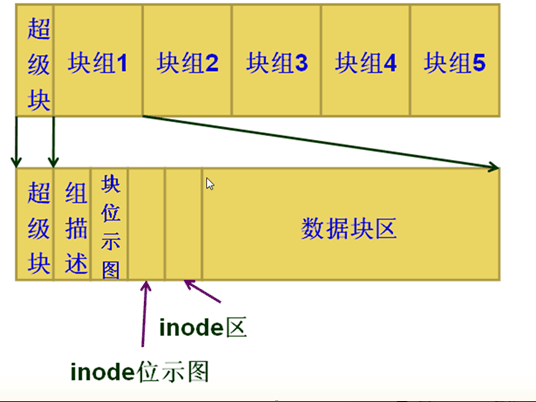
超级块存放格式化系统元数据的

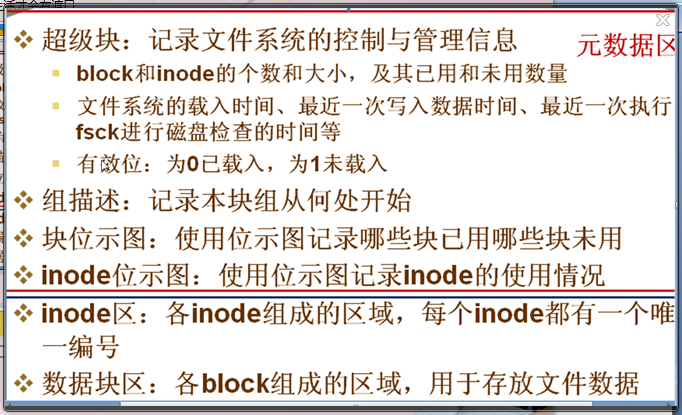
文件系统的布局

**ext4 相关的知识：**

ext4文件系统  
 Linux kernel 自 2.6.28 开始正式支持新的文件系统 Ext4。 Ext4 是 Ext3 的改进版，修改了 Ext3 中部分重要的数据结构，而不仅仅像 Ext3 对 Ext2 那样，只是增加了一个日志功能而已。Ext4 可以提供更佳的性能和可靠性，还有更为丰富的功能：  
与 Ext3 兼容。 执行若干条命令，就能从 Ext3 在线迁移到 Ext4，而无须重新格式化磁盘或重新安装系统。原有Ext3数据结构照样保留，Ext4 作用于新数据，当然，整个文件系统因此也就获得了Ext4所支持的更大容量。  
2. 更大的文件系统和更大的文件。 较之Ext3目前所支持的最大16TB文件系统和最大2TB文件，Ext4 分别支持1EB（1,048,576TB， 1EB=1024PB， 1PB=1024TB）的文件系统，以及16TB的文件。  
3. 无限数量的子目录。 Ext3 目前只支持 32,000 个子目录，而 Ext4 支持无限数量的子目录。  
4. Extents。 Ext3 采用间接块映射，当操作大文件时，效率低下。比如一个100MB 大小的文件，在 Ext3 中要建立 25,600 个数据块（每个数据块大小为4KB）的映射表。而 Ext4 引入了现代文件系统中流行的extents概念，每个extent 为一组连续的数据块，上述文件则表示为“该文件数据保存在接下来的25,600个数据块中”，提高了不少效率。  
5. 多块分配。 当写入数据到Ext3文件系统中时，Ext3的数据块分配器每次只能分配一个 4KB的块，写一个100MB 文件就要调用25,600次数据块分配器，而Ext4的多块分配器“multiblock allocator”（mballoc） 支持一次调用分配多个数据块。  
6. 延迟分配。 Ext3的数据块分配策略是尽快分配，而Ext4和其它现代文件操作系统的策略是尽可能地延迟分配，直到文件在cache中写完才开始分配数据块并写入磁盘，这样就能优化整个文件的数据块分配，与前两种特性搭配起来可以显著提升性能。  
7. 快速fsck。 以前执行fsck第一步就会很慢，因为它要检查所有的inode，现在Ext4给每个组的inode表中都添加了一份未使用inode的列表，今后fsck Ext4 文件系统就可以跳过它们而只去检查那些在用的inode了。  
8. 日志校验。 日志是最常用的部分，也极易导致磁盘硬件故障，而从损坏的日志中恢复数据会导致更多的数据损坏。Ext4 的日志校验功能可以很方便地判断日志数据是否损坏，而且它将Ext3的两阶段日志机制合并成一个阶段，在增加安全性的同时提高了性能。  
9. “无日志”（No Journaling）模式。 日志总归有一些开销，Ext4允许关闭日志，以便某些有特殊需求的用户可以借此提升性能。  
10. 在线碎片整理。 尽管延迟分配、多块分配和extents能有效减少文件系统碎片，但碎片还是不可避免会产生。Ext4支持在线碎片整理，并将提供e4defrag工具进行个别文件或整个文件系统的碎片整理。  
inode相关特性。 Ext4支持更大的inode，较之Ext3 默认的inode 大小 128 字节，Ext4 为了在inode中容纳更多的扩展属性（如纳秒时间戳或inode版本），默认inode大小为256字节。Ext4 还支持快速扩展属性（fast extended attributes）和inode保留（inodes reservation）。  
12. 持久预分配（Persistent preallocation）。 P2P 软件为了保证下载文件有足够的空间存放，常常会预先创建一个与所下载文件大小相同的空文件，以免未来的数小时或数天之内磁盘空间不足导致下载失败。 Ext4在文件系统层面实现了持久预分配并提供相应的API（libc 中的 posix\_fallocate()），比应用软件自己实现更有效率。  
13. 默认启用barrier。 磁盘上配有内部缓存，以便重新调整批量数据的写操作顺序，优化写入性能，因此文件系统必须在日志数据写入磁盘之后才能写commit记录，若commit记录写入在先，而日志有可能损坏，那么就会影响数据完整性。Ext4 默认启用barrier，只有当barrier之前的数据全部写入磁盘，才能写barrier之后的数据。（可通过 "mount -o barrier=0" 命令禁用该特性。）   
ReiserFS 文件系统  
ReiserFS是一个基于B状树的文件系统，拥有非常好的总体性能，特别是对于大量小文件。ReiserFS 拥有良好的伸缩性并具有日志功能。但该文件系统不再受到积极开发，不支持SELinux，基本上已被 Reiser4 取代。ReiserFS文件系统多年来一直用作一些发行版（包括SUSE）的默认文件系统，但现在用得少了。  
XFS文件系统  
XFS文件系统拥有日志功能，包含一些健壮的特性，并针对可伸缩性进行了优化。XFS在RAM中强制缓存中转数据，因此如果使用 XFS，建议采用不间断电源供应。淘宝的数据库在使用此文件系统。  
XFS 最初是由 Silicon Graphics，Inc. 于90年代初开发的。那时，SGI发现他们的现有文件系统（existing filesystem，EFS）正在迅速变得不适应当时激烈的计算竞争。为解决这个问题，SGI 决定设计一种全新的高性能64位文件系统，而不是试图调整EFS在先天设计上的某些缺陷。因此，XFS 诞生了，并于 1994 年随 IRIX 5.3 的发布而应用于计算。它至今仍作为 SGI 基于 IRIX 的产品（从工作站到超级计算机）的底层文件系统来使用。现在，XFS 也可以用于 Linux。XFS的Linux 版的到来是激动人心的，首先因为它为 Linux 社区提供了一种健壮的、优秀的以及功能丰富的文件系统，并且这种文件系统所具有的可伸缩性能够满足最苛刻的存储需求。

有关超级块的知识：





**1．4** **分区命令（fdisk）**

试验场景：首先添加一个磁盘（添加磁盘步骤），然后将磁盘分为6个区；使用1P+1E(5L)的方法，/dev/sdb1、/dev/sdb5、/dev/sdb6、/dev/sdb7、/dev/sdb8、/dev/sdb9.

1. 关闭虚拟机
2. 点击设置

使用fdisk -l 为列表（fdisk为分区）

分区是设置分区表，格式化是创建文件系统。

[root@moban ~]# fdisk -cu /dev/sdb

Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel

Building a new DOS disklabel with disk identifier 0xf230f8a5.

Changes will remain in memory only, until you decide to write them.

After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help): m ###查看帮助

Command action

a toggle a bootable flag 切换分区启动标记

b edit bsd disklabel 编辑sdb磁盘标签

c toggle the dos compatibility flag 切换dos兼容模式

d delete a partition 删除

l list known partition types 列分区的类型

m print this menu 打印这个菜单

n add a new partition 创建分区

o create a new empty DOS partition table 创建新的空白分区表

p print the partition table 打印分区表

q quit without saving changes 退出不保存

s create a new empty Sun disklabel 创建新的Sun磁盘标签

t change a partition's system id 更改分区的系统ID

u change display/entry units 修改容量单位，磁柱或扇区

v verify the partition table 检验分区表

w write table to disk and exit 保存，写入分区表。

x extra functionality (experts only) 拓展功能

#######开始实践：######

Command (m for help): p ---#此选项为列表

Disk /dev/sdb: 1073 MB, 1073741824 bytes

255 heads, 63 sectors/track, 130 cylinders, total 2097152 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

Disk identifier: 0xde415022

Device Boot Start End Blocks Id System

Command (m for help): n -------#添加分区

Command action

e extended 扩展分区

p primary partition (1-4) 主分区

p

Partition number (1-4): 1 选择编号

First sector (2048-2097151, default 2048): 扇区编号

Using default value 2048

Last sector, +sectors or +size{K,M,G} (2048-2097151, default 2097151): +100M

Command (m for help): p

Disk /dev/sdb: 1073 MB, 1073741824 bytes

255 heads, 63 sectors/track, 130 cylinders, total 2097152 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

Disk identifier: 0xde415022

Device Boot Start End Blocks Id System

/dev/sdb1 2048 206847 102400 83 Linux

Command (m for help): n

Command action

e extended

p primary partition (1-4)

e

Partition number (1-4): 2

First sector (206848-2097151, default 206848):

Using default value 206848

Last sector, +sectors or +size{K,M,G} (206848-2097151, default 2097151):

Using default value 2097151

Command (m for help): p

Disk /dev/sdb: 1073 MB, 1073741824 bytes

255 heads, 63 sectors/track, 130 cylinders, total 2097152 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

Disk identifier: 0xde415022

Device Boot Start End Blocks Id System

/dev/sdb1 2048 206847 102400 83 Linux

/dev/sdb2 206848 2097151 945152 5 Extended Command (m for help): w

The partition table has been altered!

Calling ioctl() to re-read partition table.

Syncing disks.

此次实践已完成，并没有全部放上来，此次实践需谨记几个参数；p 列表、n 添加分区、q 退出不保存、w 保存并退出、l 列分区的类型、d 删除分区。

[root@moban ~]# partprobe /dev/sdb 将分区表的修改结果反馈给内核。

#对新添加的磁盘快速分区方法：  
#echo -e "n\np\n1\n\n+10G\nn\np\n2\n\n+20G\nw" |fdisk /dev/sdb  
#partprobe

**1．5** **分区命令parted**

是一个磁盘分区管理工具，它比fdisk更加灵活，功能也更丰富，同时还支持GUID分区表（GUID partition table），这在IA64平台上管理磁盘时非常有用，它同时支持交互模式和非交互模式，它除了能够进行分区的添加，删除等常见操作外，还可以移动分区，制作文件系统，调整文件系统大小，复制文件系统。

GPT分区全名为（globally unique identifier partition table format），是指全局唯一标识磁盘分区表格式。

由于MBR分区表最大可寻址的存储空间只有2TB（232\*512字节）。因此，在大硬盘出现的现在，MBR分区方式逐渐被GUID分区表取代。

传统的MBR(Master Boot Record)分区方式，有一个局限：无法支持超过2TB的硬盘的分区（或单个分区超过2TB）。如果大于2T就用GPT(Globally Unique Identifier Partition Table Format)分区的概念，GPT分区表(http://baike.baidu.com/item/GPT/15413476)GPT的分区表很好了解决了传统MBR无法逾越2TB的限制。但是在Linux系统中，传统的fdisk命令无法支持gpt分区方式。（在红帽企业版Linux 4 Update 4之后的版本提供了对大于2 terabytes（TB）的磁盘设备的支持）parted的操作都是实时的，也就是说你执行了一个分区的命令，他就实实在在地分区了，而不是像fdisk那样，需要执行w命令写入所做的修改， 所以进行parted的测试千万注意不能在生产环境中！

[root@moban ~]# parted /dev/sdb

GNU Parted 2.1

使用 /dev/sdb

Welcome to GNU Parted! Type 'help' to view a list of

commands.

(parted) p

Model: VMware, VMware Virtual S (scsi)

Disk /dev/sdb: 107MB

Sector size (logical/physical): 512B/512B

Partition Table: msdos ######-----此时的分区表格式为MBR，对于2T以上的磁盘无法分区。

Number Start End Size File system Name 标志

(parted) mklabel gpt #####-----使用此条命令修改分区表的格式

警告: The existing disk label on /dev/sdb will be destroyed

and all data on this disk will be lost. Do you want to

continue?

是/Yes/否/No? y

parted) p

Model: VMware, VMware Virtual S (scsi)

Disk /dev/sdb: 107MB

Sector size (logical/physical): 512B/512B

Partition Table: gpt

Number Start End Size File system Name 标志

(parted)

(parted) mkpart primary 0 10 #####-----此条命令为添加一个主分区，大小为10M

警告: The resulting partition is not properly aligned for

best performance.

忽略/Ignore/放弃/Cancel? Ignore

(parted) p

Model: VMware, VMware Virtual S (scsi)

Disk /dev/sdb: 107MB

Sector size (logical/physical): 512B/512B

Partition Table: gpt

Number Start End Size File system Name 标志

1 17.4kB 10.0MB 9983kB primary

(parted)

parted命令可以在命令行模式下进行，此命令谨需注意的参数：rm 删除、help 帮助。 Quit 退出并保存、p打印列表分区。

非交互分区忽略提示实例：

parted /dev/sdb mklabel gpt yes

parted /dev/sdb mkpart primary ext4 0 10 ignore

parted /dev/sdb mkpart primary linux-swap 11 20 ignore

parted /dev/sdb mkpart logical ext4 21 30 ignore

parted /dev/sdb p

parted /dev/sdb quit

[root@moban ~]# ll /dev/sd\* #####--parted命令的q参数为退出并保存。。

brw-rw----. 1 root disk 8, 0 11月 3 00:08 /dev/sda

brw-rw----. 1 root disk 8, 1 11月 3 00:08 /dev/sda1

brw-rw----. 1 root disk 8, 2 11月 3 00:08 /dev/sda2

brw-rw----. 1 root disk 8, 3 11月 3 00:08 /dev/sda3

brw-rw----. 1 root disk 8, 16 11月 3 00:50 /dev/sdb

brw-rw----. 1 root disk 8, 17 11月 3 00:30 /dev/sdb1

**1.6 格式化：**

[root@moban ~]# fdisk -cu /dev/sdb

WARNING: GPT (GUID Partition Table) detected on '/dev/sdb'! The util fdisk doesn't support GPT. Use GNU Parted.

Command (m for help): p

Disk /dev/sdb: 106 MB, 106954752 bytes

255 heads, 63 sectors/track, 13 cylinders, total 208896 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

Disk identifier: 0x00000000

Device Boot Start End Blocks Id System

/dev/sdb1 2048 43007 20480 83 Linux

/dev/sdb2 43008 208895 82944 5 Extended

Command (m for help): w

The partition table has been altered!

Calling ioctl() to re-read partition table.

Syncing disks.

[root@moban ~]# partprobe /dev/sdb

Warning: /dev/sdb contains GPT signatures, indicating that it has a GPT table. However, it does not have a valid fake msdos partition table, as it should. Perhaps it was corrupted -- possibly by a program that doesn't understand GPT partition tables. Or perhaps you deleted the GPT table, and are now using an msdos partition table. Is this a GPT partition table?

这里会出现一个报错提示，这个是因为刚刚使用parted命令将分区表格式改成了GPT了，若想修改应使用

[root@moban ~]# parted /dev/sdb mklabel msdos yes 进行修改

[root@moban ~]# partprobe /dev/sdb 此时在执行命令将无报错。

[root@moban ~]# ll /dev/sdb\*

brw-rw----. 1 root disk 8, 16 11月 3 01:10 /dev/sdb

brw-rw----. 1 root disk 8, 17 11月 3 01:10 /dev/sdb1

brw-rw----. 1 root disk 8, 18 11月 3 01:10 /dev/sdb2

开始进行这一关的主题，#####格式化######。

[root@moban ~]# mkfs -t ext4 -b 4096 /dev/sdb1

mke2fs 1.41.12 (17-May-2010)

文件系统标签=

操作系统:Linux

块大小=4096 (log=2)

分块大小=4096 (log=2)

Stride=0 blocks, Stripe width=0 blocks

2560 inodes, 2560 blocks

128 blocks (5.00%) reserved for the super user

第一个数据块=0

1 block group

32768 blocks per group, 32768 fragments per group

2560 inodes per group

正在写入inode表: 完成

Creating journal (1024 blocks): 完成

Writing superblocks and filesystem accounting information: 完成

This filesystem will be automatically checked every 33 mounts or

180 days, whichever comes first. Use tune2fs -c or -i to override.

[root@moban ~]# tune2fs -c -1 /dev/sdb1 ####--使用此命令可以不进行检查。

tune2fs 1.41.12 (17-May-2010)

[root@moban ~]# mount /dev/sdb1 /mnt ####--进行挂载

[root@moban mnt]# mount –a ####--加载fstab进行挂载测试。

[root@moban ~]# df –h ####--查看挂载信息

Filesystem Size Used Avail Use% Mounted on

/dev/sda3 6.9G 1.5G 5.1G 22% /

tmpfs 118M 0 118M 0% /dev/shm

/dev/sda1 190M 36M 145M 20% /boot

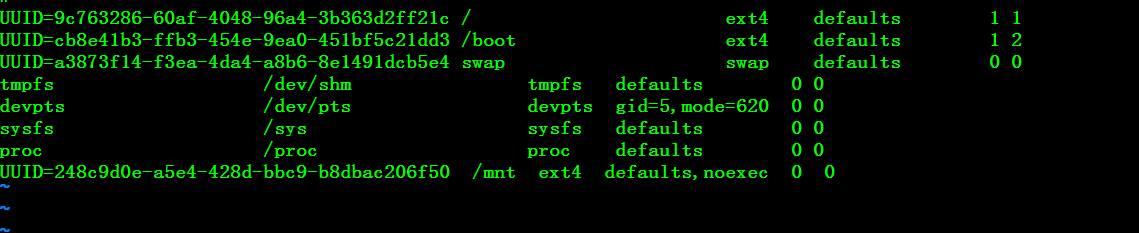
/dev/sdb1 5.7M 24K 5.2M 1% /mnt

如果想在开机就可挂载的话，放入/etc/fstab中。可以使用设备名称，也可使用UUID，下面使用UUID来进行配置。

[root@moban mnt]# blkid

/dev/sdb1: UUID="248c9d0e-a5e4-428d-bbc9-b8dbac206f50" TYPE="ext4"

[root@moban mnt]# vi /etc/fstab



第一个是需要挂载的名称 第二个是要挂载到哪里 第三个是文件类型 第四个是默认属性 第五个是是否备份 第六个是是否开机自检

[root@moban mnt]# tail -1 /etc/fstab

UUID=248c9d0e-a5e4-428d-bbc9-b8dbac206f50 /mnt ext4 defaults,noexec 0 0

[root@moban ~]# reboot

Broadcast message from root@moban

(/dev/pts/0) at 1:43 ...

The system is going down for reboot NOW!

[root@moban ~]# df –h 重启之后发现已是开机自动挂载了。

Filesystem Size Used Avail Use% Mounted on

/dev/sda3 6.9G 1.5G 5.1G 22% /

tmpfs 118M 0 118M 0% /dev/shm

/dev/sda1 190M 36M 145M 20% /boot

/dev/sdb1 5.7M 28K 5.2M 1% /mnt

强制卸载

[root@oldboy mnt]# umount -lf /mnt

**小结：磁盘管理相关命令**

**分区：** fdisk（小于2T），parted ，partprobe

实质：设置修改分区表

**格式化：**mkfs –t ext4 /dev/sdb1与tune2fs –c -1 /dev/sdb1 (,resize2fs) (dumpe2fs)

实质：创建文件系统（生成超级块，生成各种块组，组中有inode和block）

mkfs –t(指定类型) ext4 （-b与-I不需要选择）直接接设备 /dev/sdb1

tune2fs –c -1 /dev/sdb1 （不让处理）

**挂载：**

实质：为文件系统指定访问入口

mount –t ext4 /dev/sdb1 /mnt

**开机自动挂载：** /etc/fstab

/dev/sdb1 /mnt ext4 defaults 0 0

Mount –a 加载fstab进行挂载测试

Mount查看挂载

**检查挂载：**

df –h ,df –i ,fsck ,e2fsck

cat /proc/mounts或mount

**raid层面：**

使用MegaCli命令查看raid信息

ipmitool 检查硬件信息



**使用文件构建swap ：**

[root@oldboy ~]# df -h  
Filesystem      Size  Used Avail Use% Mounted on  
/dev/sda3       6.9G  1.5G  5.1G  22% /  
tmpfs           937M     0  937M   0% /dev/shm  
/dev/sda1       190M   40M  141M  23% /boot  
/dev/sdb1       8.3M   91K  7.7M   2% /mnt  
[root@oldboy ~]# free -m  
             total       used       free     shared    buffers     cached  
Mem:          1873        179       1694          0         10         58  
-/+ buffers/cache:        110       1762   
Swap:          767          0        767   
[root@oldboy ~]#   
[root@oldboy ~]#   
[root@oldboy ~]#   
[root@oldboy ~]#   
[root@oldboy ~]#   
  
[root@oldboy ~]#  dd if=/dev/zero  of=/tmp/swap bs=1M count=128  
128+0 records in  
128+0 records out  
134217728 bytes (134 MB) copied, 1.29967 s, 103 MB/s  
[root@oldboy ~]# ll /tmp/swap   
-rw-r--r--. 1 root root 134217728 Nov  3 13:42 /tmp/swap  
[root@oldboy ~]# mkswap  /tmp/swap  
mkswap: /tmp/swap: warning: don't erase bootbits sectors  
        on whole disk. Use -f to force.  
Setting up swapspace version 1, size = 131068 KiB  
no label, UUID=1e8eed1f-684c-49d1-a352-18798d7240b7  
[root@oldboy ~]# mkswap  -f /tmp/swap  
Setting up swapspace version 1, size = 131068 KiB  
no label, UUID=04d8b4ff-a770-4f3e-9c2b-606b11abf17f  
[root@oldboy ~]# swapon  /tmp/swap  
[root@oldboy ~]# free -m  
             total       used       free     shared    buffers     cached  
Mem:          1873        310       1562          0         10        186  
-/+ buffers/cache:        113       1759   
Swap:          895          0        895

1. **Linux网络基础**

**思想：**

**从系统的角度去学习网络，了解网络知识。**

**从系统的角度去学习英语，了解英语的知识。**

**2.1网线**

网线的两个标准：568A与568B（常用的）

**2.2 交换机**

品牌：dlink、H3C、Cisco

交换机（switch，意为“开关”）是一种用于电信号转发的网络设备。它可以为接入交换机的任意两个网络节点提供独享的电信号通路。最常见的交换机是以太网交换机。其他常见的还有电话语音交换机、光纤交换机等。

**2.3 路由器**

品牌：Cisco

路由器（router），是连接因特网中各局域网、广域网的设备，它会根据信道的情况自动选择和设定路由，以最佳路径，按前后顺序发送信号。

**2.4 OSI**

7层网络模型包括：  
物理层、数据链路层、网络层、传输层、会话层、表示层、应用层。

需要了解的是OSI层模型，以及这些层上常见的都对应哪些协议。

**2.5 协议**

协议是通信双方为了实现通信而设计的约定或通话规则。

http协议，tcp/ip协议。

必须掌握的：

1. tcp/ip协议的三次握手和四次断开过程。
2. http协议的工作原理。

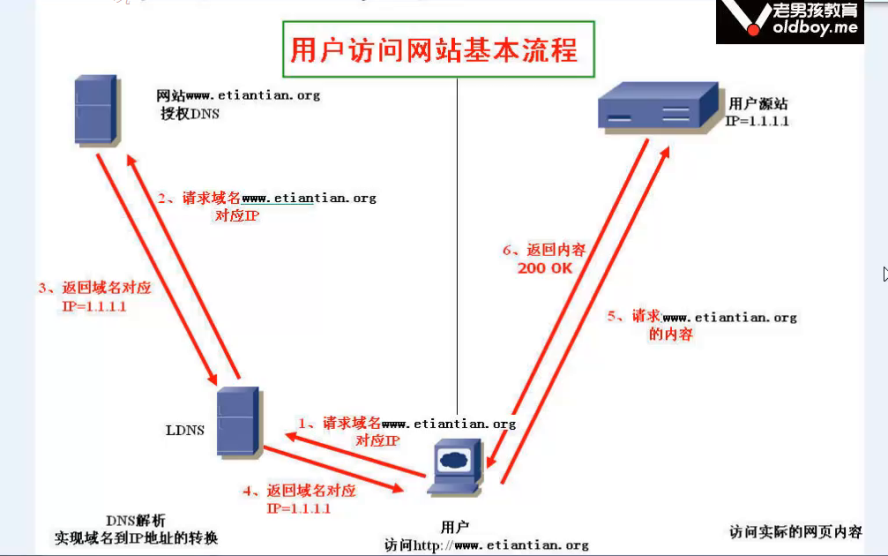
OSI七层模型及协议-包封装街封装详解

<http://www.tudou.com/programs/view/sP9JY KranA/>

TCP三次握手四次断开原理过程详解

http://www.tudou.com/programs/view/XjHCDedZQa8/

**2.6 用户访问网站的基本流程（图解）**



私网地址：

下面的网络地址段就是分配给专用网络地址使用的。

10.0.0.0/8 （10.0.0.0到10.255.255.255）

172.16.0.0/12 （172.16.0.0到172.31.255.255）

192.168.0.0/16（192.168.0.0到192.168.255.255）

169.254.0.0/16（169.254.0.0到169.254.255.255）

**2.7 DNS**

DNS是什么？

DNS(Nomain name system)

DNS能干什么？

A 记录 [www.oldboy.me](http://www.oldboy.me) --🡪 1.1.1.1 （A记录就是域名变IP的过程）

CNAME 常被CDN服务商应用

MX解析记录49000448@qq.com 搭建邮件服务

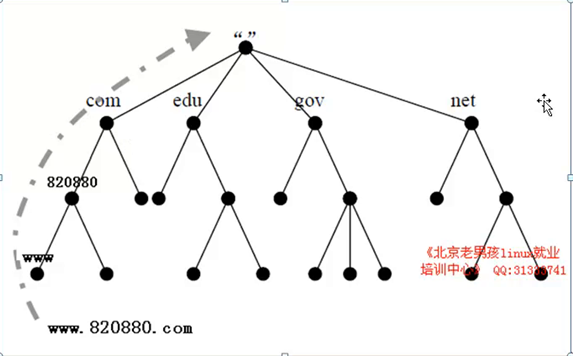
A记录解析过程：

NDS A记录

浏览器===》www.etiantian.org===》网站服务器的IP地址===》IP地址对应的网络服务器

NDS解析流程原理（必会）

NDS本身是倒挂的树状结构（与Linux目录结构类似），它的顶点也是根，是点“.”



=====》根服务器，全有有13台根服务器。

顶级域名：org、com、net、me、cc

一级域名：oldboy、etiantian、baidu、sina、google

**2.8 DNS域名解析查询：**

a.dig www.etiantian.org

b.nslookup

c.host

d.ping

[root@moban ~]# nslookup

> www.baidu.com

Server: 10.0.0.2

Address: 10.0.0.2#53

Non-authoritative answer:

www.baidu.com canonical name = www.a.shifen.com.

Name: www.a.shifen.com

Address: 119.75.217.109

Name: www.a.shifen.com

Address: 119.75.218.70

dig命令实例：

[root@moban ~]# dig @10.0.0.2 www.baidu.com +trace

; <<>> DiG 9.8.2rc1-RedHat-9.8.2-0.37.rc1.el6 <<>> @10.0.0.2 www.baidu.com +trace

; (1 server found)

;; global options: +cmd

. 5 IN NS l.root-servers.net.

. 5 IN NS b.root-servers.net.

. 5 IN NS j.root-servers.net.

. 5 IN NS a.root-servers.net.

. 5 IN NS g.root-servers.net.

. 5 IN NS m.root-servers.net.

. 5 IN NS h.root-servers.net.

. 5 IN NS i.root-servers.net.

. 5 IN NS f.root-servers.net.

. 5 IN NS e.root-servers.net.

. 5 IN NS k.root-servers.net.

. 5 IN NS d.root-servers.net.

. 5 IN NS c.root-servers.net.

;; Received 496 bytes from 10.0.0.2#53(10.0.0.2) in 50290 ms

com. 172800 IN NS b.gtld-servers.net.

com. 172800 IN NS i.gtld-servers.net.

com. 172800 IN NS k.gtld-servers.net.

com. 172800 IN NS e.gtld-servers.net.

com. 172800 IN NS f.gtld-servers.net.

com. 172800 IN NS c.gtld-servers.net.

com. 172800 IN NS h.gtld-servers.net.

com. 172800 IN NS g.gtld-servers.net.

com. 172800 IN NS j.gtld-servers.net.

com. 172800 IN NS l.gtld-servers.net.

com. 172800 IN NS m.gtld-servers.net.

com. 172800 IN NS a.gtld-servers.net.

com. 172800 IN NS d.gtld-servers.net.

;; Received 491 bytes from 192.112.36.4#53(192.112.36.4) in 26783 ms

baidu.com. 172800 IN NS dns.baidu.com.

baidu.com. 172800 IN NS ns2.baidu.com.

baidu.com. 172800 IN NS ns3.baidu.com.

baidu.com. 172800 IN NS ns4.baidu.com.

baidu.com. 172800 IN NS ns7.baidu.com.

;; Received 201 bytes from 192.52.178.30#53(192.52.178.30) in 10463 ms

www.baidu.com. 1200 IN CNAME www.a.shifen.com.

a.shifen.com. 1200 IN NS ns1.a.shifen.com.

a.shifen.com. 1200 IN NS ns2.a.shifen.com.

a.shifen.com. 1200 IN NS ns5.a.shifen.com.

a.shifen.com. 1200 IN NS ns4.a.shifen.com.

a.shifen.com. 1200 IN NS ns3.a.shifen.com.

;; Received 228 bytes from 202.108.22.220#53(202.108.22.220) in 37 ms

**2.9 网卡配置**

网卡配置文件 /etc/sysconfig/network-scripts/ifcfg-eth0

修改网卡：编辑配置文件 /etc/sysconfig/network-scripts/ifcfg-eth0

重启服务/etc/init.d/network restart 《==针对所有网卡的

ifup eth0启动eth0网卡

ifdown eth0 停掉eth0 网卡

注意：

1. 网卡如果配置DNS，会优先于/etc/resolve.conf的配置，并且重启网卡，会把/etc/resolv.conf里的覆盖。
2. 网络如果没有配置NDS，那么在/etc/resolv.conf里配置会生效，如果有多块网卡（DHCP获取方式）的时候，可能会覆盖/et/resolv.conf里已有的配置。

**2.10 推荐修改主机名的步骤：**

1. hostname oldboylinux
2. vi /etc/sysconfig/network
3. vi /etc/hosts

如果用setup—DNS设置修改主机名，永久生效（命令行不生效）

影响/etc/sysconfig/network和/etc/hosts配置文件。

**2.11 配置默认网关**

第一生效文件

[root@moban oldboy]# grep -i gate /etc/sysconfig/network-scripts/ifcfg-eth0

GATEWAY=10.0.0.2

第二生效文件

[root@moban oldboy]# grep -i gate /etc/sysconfig/network

GATEWAY=10.0.0.2

第三生效文件：命令行优先，且临时生效

[root@moban oldboy]# route –n 查看默认网关

Kernel IP routing table

Destination Gateway Genmask Flags Metric Ref Use Iface

10.0.0.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0

169.254.0.0 0.0.0.0 255.255.0.0 U 1002 0 0 eth0

0.0.0.0 10.0.0.2 0.0.0.0 UG 0 0 0 eth0

[root@moban oldboy]# netstat -rn 查看默认网关

Kernel IP routing table

Destination Gateway Genmask Flags MSS Window irtt Iface

10.0.0.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0

169.254.0.0 0.0.0.0 255.255.0.0 U 0 0 0 eth0

0.0.0.0 10.0.0.2 0.0.0.0 UG 0 0 0 eth0

[root@moban oldboy]# route del default gw 10.0.0.2 删除默认网关

[root@moban oldboy]# route -n

Kernel IP routing table

Destination Gateway Genmask Flags Metric Ref Use Iface

10.0.0.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0

169.254.0.0 0.0.0.0 255.255.0.0 U 1002 0 0 eth0

[root@moban oldboy]# ping [www.baidu.com](http://www.baidu.com) 删除默认网关后，系统无法ping通百度

connect: Network is unreachable

[root@moban oldboy]# route add default gw 10.0.0.2 添加默认网关

[root@moban oldboy]# route -n

Kernel IP routing table

Destination Gateway Genmask Flags Metric Ref Use Iface

10.0.0.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0

169.254.0.0 0.0.0.0 255.255.0.0 U 1002 0 0 eth0

0.0.0.0 10.0.0.2 0.0.0.0 UG 0 0 0 eth0

[root@moban oldboy]# ping www.baidu.com

PING www.a.shifen.com (220.181.111.188) 56(84) bytes of data.

64 bytes from 220.181.111.188: icmp\_seq=1 ttl=128 time=6.50 ms

64 bytes from 220.181.111.188: icmp\_seq=2 ttl=128 time=11.5 ms

64 bytes from 220.181.111.188: icmp\_seq=3 ttl=128 time=422 ms

^C

--- www.a.shifen.com ping statistics ---

3 packets transmitted, 3 received, 0% packet loss, time 3824ms

rtt min/avg/max/mdev = 6.509/146.981/422.933/195.138 ms

route功能很多，不仅仅配置默认网关，网络（静态）路由。

参考资料：

<http://oldboy.blog.51cto.com/2561410/1119453>

<http://oldboy.blog.51cto.com/2561410/974194>

**2.12 配置与查看网络信息**

ifconfig 查看配置别名ip等。

别名：IP：VIP 虚拟IP

使用ifconfig命令加临时IP：

**法1：**

[root@moban oldboy]# ifconfig eth0:0 10.0.0.10 netmask 255.255.255.0 up

[root@moban oldboy]# ifconfig

eth0 Link encap:Ethernet HWaddr 00:0C:29:5C:4A:FC

inet addr:10.0.0.8 Bcast:10.0.0.255 Mask:255.255.255.0

inet6 addr: fe80::20c:29ff:fe5c:4afc/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:10484 errors:0 dropped:0 overruns:0 frame:0

TX packets:3894 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:826936 (807.5 KiB) TX bytes:513286 (501.2 KiB)

eth0:0 Link encap:Ethernet HWaddr 00:0C:29:5C:4A:FC

inet addr:10.0.0.10 Bcast:10.0.0.255 Mask:255.255.255.0

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

**法2：**

[root@moban oldboy]# ifconfig eth0:1 10.0.0.11/24 up

[root@moban oldboy]# ifconfig

eth0 Link encap:Ethernet HWaddr 00:0C:29:5C:4A:FC

inet addr:10.0.0.8 Bcast:10.0.0.255 Mask:255.255.255.0

inet6 addr: fe80::20c:29ff:fe5c:4afc/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:10635 errors:0 dropped:0 overruns:0 frame:0

TX packets:3975 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:840052 (820.3 KiB) TX bytes:523026 (510.7 KiB)

eth0:0 Link encap:Ethernet HWaddr 00:0C:29:5C:4A:FC

inet addr:10.0.0.10 Bcast:10.0.0.255 Mask:255.255.255.0

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

eth0:1 Link encap:Ethernet HWaddr 00:0C:29:5C:4A:FC

inet addr:10.0.0.11 Bcast:10.0.0.255 Mask:255.255.255.0

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

**关闭临时IP的命令：**

[root@moban oldboy]# ifconfig eth0:0 down

[root@moban oldboy]# ifconfig eth0:1 down

**法3：ip的方式配多个IP，辅助IP**

[root@moban oldboy]# ip addr add 10.0.0.13/24 dev eth0:3

[root@moban oldboy]# ip add

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

inet 127.0.0.1/8 scope host lo

inet6 ::1/128 scope host

valid\_lft forever preferred\_lft forever

2: eth0: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP qlen 1000

link/ether 00:0c:29:5c:4a:fc brd ff:ff:ff:ff:ff:ff

inet 10.0.0.8/24 brd 10.0.0.255 scope global eth0

inet 10.0.0.13/24 scope global secondary eth0

inet6 fe80::20c:29ff:fe5c:4afc/64 scope link

valid\_lft forever preferred\_lft forever

使用IP命令配置辅助IP若重启网卡的话，会失效，想要永久生效的话，将配置文件写成（/etc/sysconfig/network-scripts/ifcfg-eth0:1）

**2.13 查看网络状态**

ifconfig

iP

route –n

hostname(uname -n)

netstat(ESTABLISHED建立连接的状态，数量表示网络连接数)

lsof

面试题：已知一个端口为333，如何查看端口的对应的是什么服务名？

方法1：lsof –i ：333

方法2：netstat –lntup|grep 333

**2.14 网络及服务故障的排查**

例：检查老男孩教育的网站是否有问题。

1. ping [www.etiantian.org](http://www.etiantian.org) icmp协议，（高速公路有没有修通），Linux禁止icmp。

Ping –c3 –i2 –s512 [www.baidu.com](http://www.baidu.com)

1. traceroute [www.baidu.com](http://www.baidu.com) 基础检查，各个高速节点有没有修通。
2. Telnet [www.baidu.com](http://www.baidu.com) 检查服务器WEB有没有开启，服务开没开。

以及防火墙有没有挡住，（让不让这个服务通过，让不让你跑骑车）。

Nmap 10.0.0.25 –p 80

Nc

不通：

1、80服务没开或端口不存在。

2、fw防火墙阻挡了。

3、服务监听的端口不在链接的IP上（127.0.0.1:25）

4、ISP运营商默认不开端口，申请开端口。

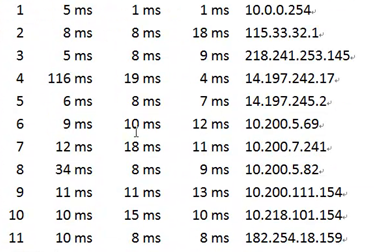
d. win：跟踪路由

tracert –d [www.baidu.com](http://www.baidu.com)

C:\Users\oldboy>tracert –d [www.qq.com](http://www.qq.com)

通过最多30个跃点跟踪到

[www.qq.com](http://www.qq.com) [182.254.18.159]的路由：



跟踪完成。

**2.15 抓包工具：**

tcpdump 命令

tcpdump –n icmp –i eth0 （如果在W环境下ping虚拟机的IP，则虚拟机会有反应）

nmap

windows

wireshark,sniffer，科来网络分析几个软件。

linux如何禁ping

解决方法：

[root@oldboy ~]# echo 1 >/proc/sys/net/ipv4/icmp\_echo\_ignore\_all。

linux如何允许ping

[root@oldboy~]# echo 0 >/proc/sys/net/ipv4/icmp\_echo\_ignore\_all

解决方法：操作完即可查看效果，无需重启。

还要修改配置文件

/etc/sysctl

最后重启内核 sysctl -p

1. **Rsync 数据同步工具应用指南**

**3．1 Rsync 介绍**

Rsync是一款开源的、快速的、多功能的、可实现全量及增强的本地或远程数据同步备份的优秀工具。Rsync软件适用于Unix/linux/windows等多种操作系统平台。

官网网站：

http://www.samba.org/ftp/rsync/rsync.html

Rsync英文全称为Remote synchronization，从软件的名称就可以看出来，Rsync具有可是本地和远程两台主机之间的数据快速复制同步镜像、远程备份的功能，这个功能类似ssh带的scp命令，但又优先于scp命令的功能，scp每次都是全量拷贝，而rsync可以增量拷贝。当然，Rsync还可以在本地主机的不同分区或目录之间全量及增量的复制数据，这又类似cp命令，但同样也优先于cp命令，cp每次都是全量拷贝，而rsync可以增量拷贝。利用rsync还可以实现删除文件和目录的功能，这又相当于rm命令。

一个rsync相当于scp（远程拷贝），cp（本地拷贝），rm（本地删除），但是还优先于这些命令。Rsync还可以是一个网络服务（端口port、socket服务）

在同步备份数据时，默认情况下，rsync通过其独特的“quick check”算法，它仅同步大小或者最后修改时间发生变化的文件或目录，当然也可根据权限，属主等属性的变化同步，但需要制定相应的参数，甚至可以实现只同步一个文件里有变化的内容部分，所以，可以实现快速的同步备份数据。

提示：传统的cp，scp工具拷贝每次均为完整的拷贝，而rsync除了可以完整拷贝外，还具备增量拷贝的功能，因此，从同步数据的性能及效率上，rsync工具更胜一筹。

**Rsync的特性**

* 支持拷贝特殊文件（链接文件），设备等。
* 可以有排除指定文件或目录同步的功能，相当于打包命令tar的排除功能。
* 可以做到保持原文件或目录的权限、时间、软硬链接、属主、组等所有属性均不改变 –p。
* 可以实现蹭量同步，即只同步发生变化的数据，因此数据传输效率很高，tar –N。
* 可以使用rcp,rsh,ssh等方式来配合传输文件（rsync本身不对数据加密）。
* 可以通过socket（进程方式）传输文件和数据（服务端和客户端）。\*\*\*\*\*
* 支持匿名的或认证（无需系统用户）的进程模式传输，可实现方便安全的进行数据备份及镜像。

**Rsync的企业工作场景说明**

1、两台服务器之间数据同步cron+rsync

2、把所有客户服务器数据同步到备份服务器（定时备份）。

期中架构实时同步方案：

实时备份；inotify(sersync)；drbd,双写，

生产场景集群架构服务器备份方案：cron+rsync

全网服务器数据备份解决方案提出及负责实施 200x.03 – 200x.09

1. 针对公司重要数据备份混乱状况和领导提出备份全网数据的解决方案。
2. 通过本地打包备份，然后rsync结合inotify应用把全网数据统一备份到一个固定存储服务器，然后在存储服务器上通过脚本检查并报警管理员备份结果。
3. 定期将IDC机房的数据备份公司的内部服务器，防止机房地震及火灾问题导致数据丢失。

Rsync的工作方式

一般来讲 ，rsync大致使用三种主要的传输数据的方式。分别为：

* 单个主机本地之间的数据传输（此时类似于cp命令的功能）。
* 借助rcp，ssh等通道来传输数据（此时类似于scp命令的功能）。
* 以守护进程（socket）的方式传输数据（这个是rsync自身的重要的功能）。

以上的几种rsync的工作方式，可以通过man rsync帮助或查看官方的手册获得：

常用的生产参数：

avz相当于vzrtopgDl

生产参数：-avz或者用-vzrtopg

更多参数：

1.<http://www.samba.org/ftp/rsync/rsync.html>

2.man rsync

**第一种工作模式：Rsync命令是客户端命令（本地可执行）：**

rsync -avz /etc/hosts /tmp/ ====cp

rsync -avz --delete /lifen/ /tmp/ ====rm

[root@moban ~]# ll /tmp/

总用量 0

[root@moban ~]# rsync -avz /etc/hosts /tmp/ 相当于复制----cp命令

sending incremental file list

hosts

sent 124 bytes received 31 bytes 310.00 bytes/sec

total size is 158 speedup is 1.02

[root@moban ~]# ll /tmp/

总用量 4

-rw-r--r-- 1 root root 158 1月 12 2010 hosts

[root@moban ~]# cp -a /etc/hosts /tmp/

cp：是否覆盖"/tmp/hosts"？ y

[root@moban ~]# mkdir /lifen

[root@moban ~]# rsync -avz --delete /lifen/ /tmp/ 相当于----rm命令

sending incremental file list

./

deleting hosts

sent 29 bytes received 15 bytes 88.00 bytes/sec

total size is 0 speedup is 0.00

[root@moban ~]# ll /tmp/

总用量 0

**第二种工作模式：通过远程（remote）的shell**

push：推

rsync –avzP –e ‘ssh –p 52113’ /tmp/ [root@10.0.0.11:/tmp/](mailto:root@10.0.0.11:/tmp/)

pull：拉

rsync –avzP –e ‘ssh –p 52113’ [root@10.0.0.11:/tmp/](mailto:root@10.0.0.11:/tmp/) /tmp/

关键语法说明：

1）-avz相当于-vzrtopgD1，表示同步时文件和目录属性不变。

2）-P显示同步的过程，可以用—progress替换。

3）-e ‘ssh –p 52113’，表示通过ssh的通道传输数据，-p 52113可省略。

4）[oldboy@10.0.0.11:/tmp](mailto:oldboy@10.0.0.11:/tmp) 远程的主机系统用户，地址，路径。

5）/tmp本地的路径。

推的实例：

[root@moban tmp]# rsync -avzP -e 'ssh -p 52113' /tmp/ root@10.0.0.11:/tmp/

root@10.0.0.11's password:

sending incremental file list

./

a

0 100% 0.00kB/s 0:00:00 (xfer#1, to-check=4/6)

b

0 100% 0.00kB/s 0:00:00 (xfer#2, to-check=3/6)

c

0 100% 0.00kB/s 0:00:00 (xfer#3, to-check=2/6)

d

0 100% 0.00kB/s 0:00:00 (xfer#4, to-check=1/6)

e

0 100% 0.00kB/s 0:00:00 (xfer#5, to-check=0/6)

sent 258 bytes received 110 bytes 105.14 bytes/sec

total size is 0 speedup is 0.00

拉的实例：

[root@moban tmp]# rm -fr \*

[root@moban tmp]# ll

总用量 0

[root@moban tmp]# rsync -avzP -e 'ssh -p 52113' root@10.0.0.11:/tmp/ /tmp/

root@10.0.0.11's password:

receiving incremental file list

./

a

0 100% 0.00kB/s 0:00:00 (xfer#1, to-check=4/6)

b

0 100% 0.00kB/s 0:00:00 (xfer#2, to-check=3/6)

c

0 100% 0.00kB/s 0:00:00 (xfer#3, to-check=2/6)

d

0 100% 0.00kB/s 0:00:00 (xfer#4, to-check=1/6)

e

0 100% 0.00kB/s 0:00:00 (xfer#5, to-check=0/6)

sent 109 bytes received 252 bytes 80.22 bytes/sec

total size is 0 speedup is 0.00

[root@moban tmp]# ll

总用量 0

-rw-rw-r-- 1 test test 0 11月 5 09:56 a

-rw-rw-r-- 1 test test 0 11月 5 09:56 b

-rw-rw-r-- 1 test test 0 11月 5 09:56 c

-rw-rw-r-- 1 test test 0 11月 5 09:56 d

-rw-rw-r-- 1 test test 0 11月 5 09:56 e

总结：在推文件的时候，被推方的用户要使用root，不然文件的属主与属组会改变，且在推送过程中会提示报错，但可以正常推送。

**第三种工作模式：daemon（服务）访问**

[root@backup ~]# cat /etc/redhat-release   
CentOS release 6.7 (Final)  
[root@backup ~]#   
[root@backup ~]# uname -m  
x86\_64  
[root@backup ~]# uname -r  
2.6.32-573.el6.x86\_64

#####切记：rsync也就是备份服务器端#####

[root@moban tmp]# rsync –version ###查看rsync版本。

rsync version 3.0.6 protocol version 30

[root@moban tmp]# rpm -qa rsync ###查询是否有安装包。

rsync-3.0.6-12.el6.x86\_64

rsync服务的配置文件是默认不存在的，需要添加

[root@moban etc]# touch /etc/rsyncd.conf

[root@moban etc]# vi rsyncd.conf ###添加rsync的配置信息

[root@moban etc]# cat rsyncd.conf

#Rsync server

#created by lifen 11:32 2015-11-05

##rsyncd.conf start##

uid = rsync ###客户端连过来具备什么样的权限（具备rsync的权限）

gid = rsync ###客户端连过来具备什么样的权限（具备rsync的权限）

use chroot = no ###程序出现bug的时候开启这个服务会有好处

max connections = 2000 ###客户端有多少个可以同时连接进来

timeout = 600 ###超时

pid file = /var/run/rsyncd.pid ###进程号的文件

lock file = /var/run/rsync.lock ###锁文件

log file = /var/log/rsyncd.log ###日志文件（出现问题可来此看）

ignore errors ###忽略报错

read only = false ###可读写（变成ture就只能拉不鞥呢推，没有了写的权限）

list = false ###列表等于假，不允许列表

hosts allow = 10.0.0.0/24 ###允许IP段

hosts deny = 0.0.0.0/32 ###不限制IP

auth users = rsync\_backup ###启用rsync\_backup虚拟用户（远程连接的用户）

secrets file = /etc/rsync.password ###指定密码文件连接（存放用户和密码的文件）

#######################################

[backup] ###共享一个目录（模块）

comment = www by lifen 11:38 2015-11-05 ###注释

path = /backup ###要共享的目录

下一步开始：

[root@moban ~]# rsync –daemon ###启动rsync服务

[root@moban ~]# ps -ef|grep rsync|grep -v grep ###查询rsync端口号

root 6589 1 0 14:06 ? 00:00:00 rsync --daemon

[root@moban ~]# netstat -lntup|grep rsync ###查询rsync端口号

tcp 0 0 0.0.0.0:873 0.0.0.0:\* LISTEN 6589/rsync

tcp 0 0 :::873 :::\* LISTEN 6589/rsync

[root@moban ~]# ss -lntup|grep rsync ###查询rsync端口号

tcp LISTEN 0 5 :::873 :::\* users:(("rsync",6589,5))

tcp LISTEN 0 5 \*:873 \*:\* users:(("rsync",6589,3))

[root@moban ~]# lsof -i :873 ###查询该端口号对应的服务

COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME

rsync 6589 root 3u IPv4 24513 0t0 TCP \*:rsync (LISTEN)

rsync 6589 root 5u IPv6 24514 0t0 TCP \*:rsync (LISTEN)

[root@qizhog1 etc]# useradd rsync -s /sbin/nologin -M

正在创建信箱文件: 文件已存在

[root@qizhog1 etc]# id rsync

uid=504(rsync) gid=504(rsync) 组=504(rsync)

[root@qizhog1 etc]# mkdir /backup

[root@qizhog1 etc]# chown -R rsync /backup/

[root@qizhog1 etc]# ls -ld /backup/

drwxr-xr-x 2 rsync root 4096 11月 5 14:20 /backup/

[root@qizhog1 etc]# echo "rsync\_backup:oldboy" > /etc/rsync.password

[root@qizhog1 etc]# cat /etc/rsync.password

文件中的账号是和配置文件auth users = rsync\_backup一样的，冒号后面的是密码。

rsync\_backup:oldboy

[root@qizhog1 etc]# ll /etc/rsync.password

-rw-r--r-- 1 root root 20 11月 5 14:23 /etc/rsync.password

[root@qizhog1 etc]# chmod 600 /etc/rsync.password #将这个文件的权限修改为600

[root@qizhog1 etc]# ll /etc/rsync.password

-rw------- 1 root root 20 11月 5 14:23 /etc/rsync.password

[root@qizhog1 ~]# echo "rsync --daemon" >> /etc/rc.local

[root@qizhog1 ~]# cat /etc/rc.local

#!/bin/sh

#

# This script will be executed \*after\* all the other init scripts.

# You can put your own initialization stuff in here if you don't

# want to do the full Sys V style init stuff.

touch /var/lock/subsys/local

rsync --daemon

**Rsync server：配置服务端步骤**

1、vi /etc/rsyncd.conf 加入配置

2、创建rsync用户，及共享目录/backup

useradd rsync –s /sbin/nologin –M

id rsync

mkdir /backup

chown –R rsync /backup

3、创建密码文件

[root@qizhog1 etc]# echo "rsync\_backup:oldboy" > /etc/rsync.password

chmod 600 /etc/rsync.password

4.、rsync –daemon （启动服务）

netstat –lntup|grep rsync

ps –ef|grep rsync|grep –v grep

5、加入开机自启动

echo “/usr/bin/rsync --daemon” >> /etc/rc.local

cat /etc/rc.local

**Rsync server：配置客户端步骤**

1、创建密码文件

echo "oldboy" > /etc/rsync.password

chmod 600 /etc/rsync.password

ls –l /etc/rsync.password

cat /etc/rsync.password

2、rsync

push:

rsync –avz /tmp/ [rsync\_backup@10.0.0.11::backup](mailto:rsync_backup@10.0.0.11::backup) --password-file=/etc/rsync.password

pull:

[root@qizhong2 tmp]# rsync -avz rsync\_backup@10.0.0.11::backup /tmp/ --password-file=/etc/rsync.password

[root@qizhong2 ~]# echo "oldboy" > /etc/rsync.password

[root@qizhong2 ~]# chmod 600 /etc/rsync.password

[root@qizhong2 ~]# ls -l /etc/rsync.password

-rw------- 1 root root 7 11月 5 14:57 /etc/rsync.password

[root@qizhong2 ~]# cat /etc/rsync.password

Oldboy

第二种方法：

第二种方式不常用：  
(推)  
rsync -avz /tmp/ rsync://rsync\_backup@172.16.1.41/backup/ --password-file=/etc/rsync.password   
(拉)  
rsync -avz rsync://rsync\_backup@10.0.0.11/backup/  /tmp/ --password-file=/etc/rsync.password

**重启rsync的组合命令**

pkill rsync #---关闭rsync服务

killall rsync

kill –HUP `cat /var/run/rsyncd.pid`

kill –USR2 `cat /var/run/rsyncd.pid`

kill -9 很危险，禁止使用。

rsync –daemon 启动rsync服务

检查启动的进程

ps –ef|grep rsync 检查rsync服务。

**Rsync服务问题：**

1、不管是推还是拉都是客户端操作

2、注意斜线问题。

path = /backup/

客户端命令：

rsync –avz /tmp/ [rsync\_backup@10.0.0.11::backup](mailto:rsync_backup@10.0.0.11::backup) --password-file=/etc/rsync.password

3、同步安全优化

Pkill rsync

losf –i :873

rsync –daemon –address=10.0.0.12（本地IP）

netstat –lntup|grep rsync

**rsync排错思路**

**服务端排错思路：**

1、查看rsync服务配置文件路径是否正确，正确的默认路径为：/etc/rsyncd.conf

2、查看配置文件里host allow，host deny，允许的ip网段是否是允许客户端访问的ip网段。

3、查看配置文件中path参数里的路径是否存在，权限是否正确（正常应为配置文件中的UID参数对应的属主和组）

4、查看rsync服务是否启动，查看命令：ps –ef|grep rsync，端口是否存在netstat –lntup|grep 873.

5、查看iptables防火墙和selinux是否开启允许rsync服务通过，也可考虑关闭。

6、查看服务端rsync配置的密码文件是否为600的权限，密码文件格式是否正确，正确格式用户名：密码，文件路径和配置文件里的secrect files 参数对应。

7、如果是推送数据，要查看下，配置rsyncd.conf文件中用户是否对模块下目录有可读写的权限。

**Rsync客户端排错思路**

1、 查看客户端rsync配置的密码文件是否为600的权限，密码文件格式是否正确，注意：文件文件内需要有密码，并且和服务端的密码一致。

2、 用Telnet连接rsync服务器ip地址873端口，查看服务是否启动（可测试服务端防火墙是否阻挡）。Telnet 10.0.0.12 873

3、客户端执行命令时rsync –avzP [rsync\_backup@10.0.0.11::oldboy/test/](mailto:rsync_backup@10.0.0.11::oldboy/test/) /test/ --password-file=/etc/rsync.password

此命令的细节要记清楚，尤其10.0.0.11::oldboy/test/处的双冒号及随其后的oldboy模块名称。

**自我模拟排错实践**

正确的路走通，然后变通，偏偏不按要求走，看有什么错误。

**Rsync优点：**

1. 蹭量备份同步，支持socket（daemon模式），集中备份（支持推拉，都是以客户端为参照物）。
2. 远程shell通道模式还可以加密（SSH）传输。socket（daemon模式）需要加密传输，可以利用vpn服务或ipsec服务。

**Rsync缺点：\*\*\*\*\***

1、大量小文件同步的时候，比对时间比较长，有的时候rsync进程会停止。（或是网络中断导致）

解决方案：如果是大量小文件的话最好先打包，然后同步；drbd（文件系统同步复制block）；写文件的时候同时写两份。

2、同步大文件，10G这样的大文件有时也会有问题，中断，且在未完整同步前，是隐藏文件，同步完成改为正常文件。

**无差异同步： --delete（工作中慎用，若使用前记得备份）**

推送：

本地有的远端就会有，本地没有的，就会删除远端独有的。

Rsync –avz –delete /tmp/ rsync://rsync\_backup@10.0.0.11/backup –password-file=/etc/rsync.password

[root@qizhong2 tmp]# rsync -avz --delete /tmp/ rsync://rsync\_backup@10.0.0.11/backup --password-file=/etc/rsync.password

sending incremental file list

./

deleting e

deleting d

deleting c

deleting b

deleting a

1

2

拉取：

远端有的本地就有，远端没有的，就会删除本地独有的。

Rsync –avz –delete rsync://rsync\_backup@10.0.0.11/backup /tmp/ –password-file=/etc/rsync.password

[root@qizhong2 tmp]# rsync -avz --delete rsync://rsync\_backup@10.0.0.11/backup /tmp/ --password-file=/etc/rsync.password

receiving incremental file list

deleting e

deleting d

deleting c

deleting b

deleting a

./

**客户端rsync命令排除：**

排除单个文件：

[root@qizhong2 tmp]# rsync -avz --exclude=a /tmp/ rsync\_backup@10.0.0.11::backup --password-file=/etc/rsync.password

**排除多个文件：**

法1：

[root@qizhong2 tmp]# rsync -avz --exclude={1,3,a,b} /tmp/ rsync\_backup@10.0.0.11::backup --password-file=/etc/rsync.password

法2：

[root@qizhong2 tmp]# rsync -avz --exclude=1 --exclude=e /tmp/ rsync\_backup@10.0.0.11::backup --password-file=/etc/rsync.password

法3：

[root@qizhong2 tmp]# rsync -avz --exclude={1..5} /tmp/ rsync\_backup@10.0.0.11::backup --password-file=/etc/rsync.password

老师发的文档：

排除同步：  
排除单个文件：  
rsync -avz --exclude=a /backup/ rsync\_backup@172.16.1.41::backup --password-file=/etc/rsync.password  
  
排除多个文件：  
rsync -avz --exclude={a,b} /backup/ rsync\_backup@172.16.1.41::backup --password-file=/etc/rsync.password  
  
rsync -avz --exclude=a  --exclude=b /backup/ rsync\_backup@172.16.1.41::backup --password-file=/etc/rsync.password   
  
rsync -avz --exclude={a..g} /backup/ rsync\_backup@172.16.1.41::backup --password-file=/etc/rsync.password  
  
rsync -avz --exclude-from=paichu.log /backup/ rsync\_backup@172.16.1.41::backup --password-file=/etc/rsync.password   
  
拉取和推送都可以。  
也可以服务端排除，配置文件里参数。  
exclude=a c b d

**使用定时任务+rsync**

[root@qizhong2 ~]# mkdir /backup/`ifconfig eth1|awk -F '[ :]+' 'NR==2 {print $4}'`\_$(date +%F) -p

[root@qizhong2 ~]# ll /backup/

总用量 4

drwxr-xr-x 2 root root 4096 11月 5 21:38 10.0.0.12\_2015-11-05

[root@qizhong2 ~]# cp /var/spool/cron/root /backup/10.0.0.12\_2015-11-05/cron\_root\_$(date +%F)

[root@qizhong2 ~]# cp /etc/rc.local /backup/10.0.0.12\_2015-11-05/rc.local\_$(date +%F)

[root@qizhong2 ~]# ls /backup/10.0.0.12\_2015-11-05/

[root@backup backup]# ls -l /proc/sys/fs/inotify

总用量 0

-rw-r--r-- 1 root root 0 11月 6 19:20 max\_queued\_events

-rw-r--r-- 1 root root 0 11月 6 19:20 max\_user\_instances

-rw-r--r-- 1 root root 0 11月 6 19:20 max\_user\_watches

[root@backup backup]#

[root@backup backup]# ls -l /proc/sys/fs/inotify/max\_queued\_events

-rw-r--r-- 1 root root 0 11月 6 19:20 /proc/sys/fs/inotify/max\_queued\_events

[root@backup backup]# cat /proc/sys/fs/inotify/max\_queued\_events

16384

[root@backup backup]# cat /proc/sys/fs/inotify/max\_user\_instances

128

[root@backup backup]# cat /proc/sys/fs/inotify/max\_user\_watches

8192

**4、Rsync+inotify组合的起源**

Rsync远程同步工具，通过rsync可以实现对远程服务器数据的增量备份同步，但rsync自身也有瓶颈，同步数据时，rsync采用核心算法对远程服务器的目录文件进行比对，只进行差异同步。如果服务器的文件数量达到百万甚至千万量级，那么文件比对将是非常耗时的，而且发生变化的往往是期中很少的一部分，这是非常低效的方式。inotify的出现，可以缓解rsync不足之处，取长补短。

Inotify是一种强大的、细粒度的、异步的文件系统时间监控机制，Linux内核从2.6.13起，加入了inotify支持，通过inotify可以监控文件系统中添加、删除、修改、移动等各种事件，利用这个内核接口，第三方软件就可以监控文件系统下文件的各种变化情况，而inotify-tools正式实施这样监控的软件，国人周洋在金山公司开发的sersync。

Inotify实际是一种事件驱动机制，它为应用程序监控文件系统时间提供了实时相应事件的机制，而无须通过诸如cron等的轮询机制来获取时间。Cron等机制步进无法做到实时性，而且消耗大量系统资源。相比之下inotify基于事件驱动，可以做到对事件处理的实时响应，也没有轮询造成的系统资源消耗，是非常自然的事件通知接口，也与自然世界的事件机制相符合。

特别说明：下面的inotify配置是建立在rsync服务基础上的配置过程。

实施环境：

配置rsync服务，并且能在该客户端推送和拉取。

关键参数说明：

在/proc/sys/fs/inotify目录下有三个文件，对inotify机制有一定的限制

Max\_user\_watches:

老师视频讲解中安装inotify整个过程：

ls –l /proc/sys/fs/inotify/

tar zxf inotify-tools-3.14.tar.gz

cd inotify-tools-3.14

./configure –perfix=/usr/local/inotify-tools-3.14

echo $?

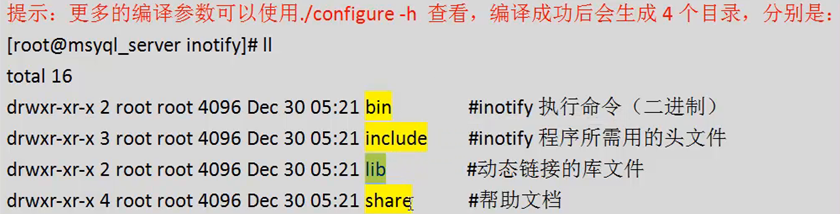
make &&make install

echo $?

cd ../

ln –s /usr/local/inotify-tools-3.14/ /usr/local/inotify-tools

ls –l /usr/local/inotify-tools/



[root@backup backup]# ls -l /proc/sys/fs/inotify

总用量 0

max\_queued\_events 设置inotifywait或inotifywatch命令可以监视的文件数量（单进程）

max\_user\_instances 设置每个用户可以运行的inotifywait或inotifywatch命令的进程数

max\_user\_watches 设置inotify实例事件（event）队列可容纳的事件数量。

[root@backup backup]#

[root@backup backup]# ls -l /proc/sys/fs/inotify/max\_queued\_events

-rw-r--r-- 1 root root 0 11月 6 19:20 /proc/sys/fs/inotify/max\_queued\_events

[root@backup backup]# cat /proc/sys/fs/inotify/max\_queued\_events

16384

[root@backup backup]# cat /proc/sys/fs/inotify/max\_user\_instances

128

[root@backup backup]# cat /proc/sys/fs/inotify/max\_user\_watches

8192

[root@backup backup]# rpm -qa inotify-tools

inotify-tools-3.14-1.el6.x86\_64

添加

inotifywait -mrq --timefmt '%d/%m/%y %H:%M' --format '%T %w%f' -e create /backup

删除

inotifywait -mrq --timefmt '%d/%m/%y %H:%M' --format '%T %w%f' -e delete /backup

添加+删除

inotifywait -mrq --format '%w%f' -e create,close\_write,delete /backup

inotify另外的脚本：精准处理  
[root@web scripts]# cat inotify1.sh   
#!/bin/sh  
cmd="/usr/local/inotify-tools/bin/inotifywait"  
$cmd -mrq  --format '%w%f' -e create,close\_write,delete /backup|\  
while read line  
do  
  [ ! -e  "$line" ] && cd /backup &&\  
 rsync -az --delete ./ rsync\_backup@10.0.0.8::backup --password-file=/etc/rsync.password && continue  
  rsync -az --delete $line rsync\_backup@10.0.0.8::backup --password-file=/etc/rsync.password  
done

**Inotify优点：**

1. 监控文件系统事件变化，通过同步工具实现实时数据同步。

**Inotify缺点：**

1. 并发如果大于200个文件（10-100K），同步就会有延迟。
2. 前面写的脚本，每次都是全部推送一次，但确实是增量。也可以只同步变化的文件，不变化的不理。
3. 监控到事件后，调用rsync同步时单进程的（加&并发），sersync多进程同步。

既然有了inotify-tools，为什么还要开发sersync？

Sersync功能多：

1. 支持配置文件
2. 真正的守护进程socket
3. 可以对失败文件定时重传。
4. 第三方的HTTP接口
5. 默认多线程同步。

高并发数据实时同步方案小结：

1. inotify+rsync文件级别
2. drbd文件系统级别，基于block
3. 第三方软件的同步功能：MySQL同步，oracle，mongodb
4. 程序双写，直接写两台服务器。
5. 业务逻辑解决。（读写分离，备读不到，读主）