**服务器软硬磁盘阵列**

# 概述：

RAID（ Redundant Array of Inexpensive Disks）称为廉价磁盘冗余阵列。 RAID 的基本思想是把多个便宜的小磁盘组合到一起，组合为一个大磁盘组，使性能达到或超过一个容量巨大、价格昂贵、读写速度快的磁盘。

目前 RAID 技术主要分为两种：基于硬件的RAID技术和基于软件的RAID 技术。在 Linux 系统中通过自带的软件就能模拟实现 RAID 功能，这样可省去购买昂贵的硬件 RAID 控制器的费用，便可极大地增强磁盘的 IO 性能和可靠性。由于是用软件去模拟实现的 RAID 功能，所以它的配置灵活、管理方便。同时使用软件 RAID，还可以实现将几个物理磁盘合并成一个更大的虚拟设备，从而达到性能改进和数据冗余的目的。当然基于硬件的 RAID 解决方案比基于软件 RAID 技术在性能上会胜一筹，具体表现在检测和修复多位错误的能力、错误磁盘自动检测和阵列重建等方面。

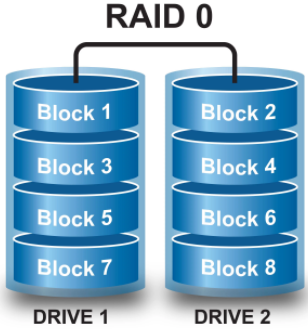
## RAID 级别介绍

随着 RAID 技术经过不断的发展，现已有 RAID 0 到 RAID 6 七种基本的 RAID 级别，同时还有 RAID 0+RAID 1 的组合形式，称为RAID10，还有 RAID 0+RAID 5 的组合形式，称为RAID50。当然级别并不能代表技术的高低，其中 RAID 2-RAID 4 基本上不再使用了。

目前这些常用的 RAID 级别 Linux 内核都能够支持，本节以 Linux 2.6 以上的内核为例，在 Linux 2.6 内核中的软 RAID 可支持以下级别： RAID 0、 RAID 1、RAID 4、 RAID 5 以及 RAID 6 等。 Linux 2.6 的内核除支持以上几种 RAID 级别外，还可支持 LINEAR（线性模式）的软 RAID，线性模式是将两个或更多的磁盘组合到一个物理设备中，磁盘不必具有相同的大小，在写入 RAID 设备时会首先填满磁盘 A，然后是磁盘 B，以此类推。

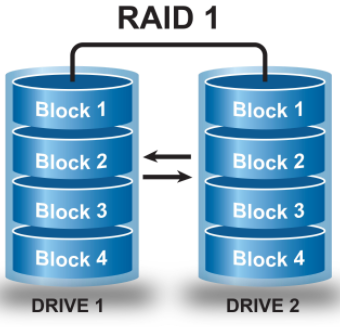
### RAID 0

也称为条带模式（striped），即把连续的数据分散到多个磁盘上存取。当系统有数据请求就可以被多个磁盘并行的执行，每个磁盘执行属于它自己的那部分数据请求。这种数据上的并行操作可以充分利用磁盘总线的带宽，显著提高磁盘整体存取性能。因为读取和写入是在设备上并行完成的，读取和写入性能将会增加，这通常是运行 RAID 0 的主要原因。但 RAID 0 没有数据冗余，如果其中一个硬盘出现故障，那么将无法恢复任何数据。



### RAID 1

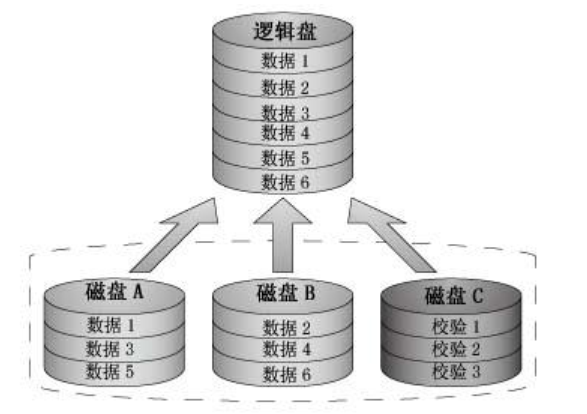
RAID 1 又称为镜像（ Mirroring），一个具有全冗余的模式。 RAID 1 可以用于两个或 2xN 个磁盘，并使用 0 块或更多的备用磁盘，每次写数据时会同时写入镜像盘。这种阵列可靠性很高，但其有效容量会减小到总容量的一半， 同时这些磁盘的大小应该相等，否则总容量只具有最小磁盘的大小。



### RAID 4

创建 RAID 4 需要三块或更多的磁盘，它在一个磁盘上保存校验信息，并以 RAID 0 方式将数据写入其它磁盘。因为一块磁盘是为校验信息保留的，所以阵列的空间大小是（ N－l）\*S，其中S 是阵列中最小磁盘的大小。就像在 RAID 1 那样，磁盘的大小应该相等。

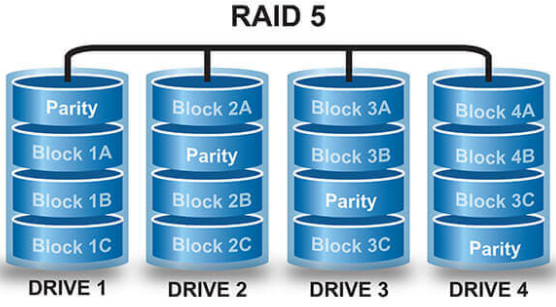
如果一个磁盘出现故障，那么可以使用校验信息及另一个磁盘来重建数据。如果两个磁盘出现故障，那么所有数据都将丢失。不经常使用这个级别的原因是校验信息存储在一个磁盘上。每次写入其它磁盘时，都必须更新这些信息。因此，在大量写入数据时很容易造成校验磁盘的瓶颈，所以目前这个级别的 RAID 很少使用了。



### RAID 5

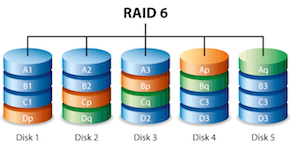
在希望结合大量物理磁盘并且仍然保留一些冗余时， RAID 5 可能是最有用的 RAID 模式。 RAID 5可以用在三块或更多的磁盘上，并使用0 块或更多的备用磁盘。就像 RAID 4 一样，RAID5 设备的大小是（N－1）\*S。

RAID5 与 RAID4 之间最大的区别就是校验信息均匀分布在各个驱动器上，这样就避免了 RAID 4 中出现的瓶颈问题。如果其中一块磁盘出现故障，那么由于有校验信息，所以所有数据仍然可以保持不变。如果可以使用备用磁盘，那么在设备出现故障之后，将立即开始往备用磁盘上同步数据。如果两块磁盘同时出现故障，那么所有数据都会丢失。



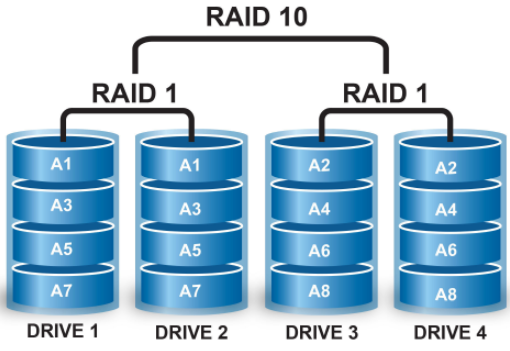
### RAID 6

RAID 6 是在 RAID 5 基础上扩展而来的。与 RAID 5 一样，数据和校验码都是被分成数据块然后分别存储到磁盘阵列的各个硬盘上。只是RAID 6 中多增加一块校验磁盘，用于备份分布在各个磁盘上的校验码，这样 RAID 6 磁盘阵列就允许两个磁盘同时出现故障，所以RAID 6 的磁盘阵列最少需要四块硬盘。



### RAID1+0：

N（偶数，N>=4）块盘两两镜像后，在组合成一个RAID 0。容量为N/2，N/2块盘同时写入，写速度一般，N快盘同时读取，读速度较快。性能高，可靠性高。



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | RAID0 | RAID1 | RAID5 | RAID6 | RAID10 | RAID50 |
| 特点 | 并行执行  读写速度快 | 数据完全冗余 | 读写速度快  数据有校验 | 读写速度快  数据有校验，冗余性高 | 读写速度更快  数据有校验高 | 读写速度更快  数据有校验高 |
| 缺点 | 数据无冗余 | 读写速度慢 |  | 磁盘数量太多 | 磁盘数量太多 | 磁盘数量太多 |
| 磁盘数 | 1 | 2 | 3 | 4 | 4 | 6 |
| 故障  磁盘数 | 0 | 1 | 1 | 2 | 2 | 2 |
| 空间 | 总和 | 1/2 | （N-1）\*S | （N-2）\*S | 1/2 | （N-2）\*S |
| 场景 | 大空间测试 | 系统 | 常规服务 |  | 数据库 |  |

## 创建软 RAID

在Linux 服务器中可通过 mdadm 工具来创建和维护软 RAID 的， mdadm 在创建和管理软 RAID 时非常方便，而且很灵活。 mdadm 常用的参数有如下：

* -C --create：创建一个软RAID，后面需要标识RAID设备的名称。例如，/dev/md0，/dev/md1
* -A --assemble：加载一个已存在的RAID，后面跟RAID以及设备的名称。
* -D --detail：输出指定 RAID 设备的详细信息。
* -S --stop：停止指定的 RAID 设备。
* -l --level：指定 RAID 配置级别，例如，设置“--level=5”则表示创建阵列的级别是 RAID 5。
* -n --raid-devices：指定RAID中活动磁盘的数目。
* -r --remove：删除RAID中的某个磁盘
* -a --add：向RAID中添加磁盘
* -x 指定备用磁盘数量
* -s --scan：扫描配置文件/proc/mdstat来搜索软 RAID 的配置信息，该参数不能单独使用，需要配合其它参数才能使用。

## 环境描述：

|  |  |  |  |
| --- | --- | --- | --- |
| 操作系统 | IP地址 | 主机名 | 软件包列表 |
| CentOS7.5-x86\_64 | **192.168.200.111** | **localhost** | **mdadm** |

LINUX服务器中添加1块100G空闲的硬盘，划分4个20G的分区用来练习创建一个 RAID 5及其后期维护操作。

### 配置过程：

**[root@localhost ~]# rpm -q mdadm**

**mdadm-4.0-13.el7.x86\_64**

### 创建4个20G的分区：

首先使用“fdisk”命令在硬盘上创建4个分区，操作如下：

**[root@localhost ~]# fdisk /dev/sdb**

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

**Building a new DOS disklabel with disk identifier 0x8f67b364.**

**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)**

**WARNING: DOS-compatible mode is deprecated. It's strongly recommended to**

**switch off the mode (command 'c') and change display units to**

**sectors (command 'u').**

**Command (m for help): n**

**Command action**

**e extended**

**p primary partition (1-4)**

**p**

**Partition number (1-4): 1**

**First cylinder (1-13054, default 1):**

**Using default value 1**

**Last cylinder, +cylinders or +size{K,M,G} (1-13054, default 13054): +20G**

**Command (m for help): n**

**Command action**

**e extended**

**p primary partition (1-4)**

**p**

**Partition number (1-4): 2**

**First cylinder (2613-13054, default 2613):**

**Using default value 2613**

**Last cylinder, +cylinders or +size{K,M,G} (2613-13054, default 13054): +20G**

**Command (m for help): n**

**Command action**

**e extended**

**p primary partition (1-4)**

**p**

**Partition number (1-4): 3**

**First cylinder (5225-13054, default 5225):**

**Using default value 5225**

**Last cylinder, +cylinders or +size{K,M,G} (5225-13054, default 13054): +20G**

**Command (m for help): n**

**Command action**

**e extended**

**p primary partition (1-4)**

**p**

**Selected partition 4**

**First cylinder (7837-13054, default 7837):**

**Using default value 7837**

**Last cylinder, +cylinders or +size{K,M,G} (7837-13054, default 13054): +20G**

**Command (m for help): p**

**Disk /dev/sdb: 107.4 GB, 107374182400 bytes**

**255 heads, 63 sectors/track, 13054 cylinders**

**Units = cylinders of 16065 \* 512 = 8225280 bytes**

**Sector size (logical/physical): 512 bytes / 512 bytes**

**I/O size (minimum/optimal): 512 bytes / 512 bytes**

**Disk identifier: 0x8f67b364**

**Device Boot Start End Blocks Id System**

**/dev/sdb1 1 2612 20980858+ 83 Linux**

**/dev/sdb2 2613 5224 20980890 83 Linux**

**/dev/sdb3 5225 7836 20980890 83 Linux**

**/dev/sdb4 7837 10448 20980890 83 Linux**

**Command (m for help): w**

**The partition table has been altered!**

**Calling ioctl() to re-read partition table.**

**Syncing disks.**

**[root@localhost ~]# partprobe /dev/sdb**

### 创建RAID5

创建完/dev/sdb1、 /dev/sdb2、 /dev/sdb3、 /dev/sdb4 四个分区后，下面就可以创建 RAID 5 了，其中设定/dev/sdb4 作为备用磁盘，其余为活动磁盘，备用磁盘的作用是一旦某一设磁盘损坏可以立即使用备用磁盘替换。操作命令如下：

**[root@localhost ~]# mdadm --create /dev/md5 --level=5 --raid-devices=3 --spare-devices=1 /dev/sdb[1-4]**

**mdadm: Defaulting to version 1.2 metadata**

**mdadm: array /dev/md5 started.**

其中“--spare-devices=1”表示当前阵列中备用设备只有一块，即作为备用设备的“/dev/sdb4”，若有多块备用设备，则将“--spare-devices”的值设置为相应的数目。成功创建完成 RAID 设备后，通过如下命令可以查看到 RAID 的详细信息：

**[root@localhost ~]# mdadm --detail /dev/md5**

**/dev/md5:**

**Version : 1.2**

**Creation Time : Tue Oct 18 22:59:32 2016**

**Raid Level : raid5**

**Array Size : 41928704 (39.99 GiB 42.93 GB)**

**Used Dev Size : 20964352 (19.99 GiB 21.47 GB)**

**Raid Devices : 3**

**Total Devices : 4**

**Persistence : Superblock is persistent**

**Update Time : Tue Oct 18 23:01:01 2016**

**State : clean, degraded, recovering**

**Active Devices : 2**

**Working Devices : 4**

**Failed Devices : 0**

**Spare Devices : 2**

**Layout : left-symmetric**

**Chunk Size : 512K**

**Rebuild Status : 14% complete**

**Name : localhost.localdomain:5 (local to host localhost.localdomain)**

**UUID : 307f1234:08c3d0ec:c31390d1:cbbecfc9**

**Events : 3**

**Number Major Minor RaidDevice State**

**0 8 17 0 active sync /dev/sdb1**

**1 8 18 1 active sync /dev/sdb2**

**4 8 19 2 spare rebuilding /dev/sdb3**

**3 8 20 - spare /dev/sdb4**

### 创建 RAID 的配置文件

RAID 的配置文件名为“mdadm.conf”，默认是不存在的，所以需要手工创建，该配置文件存在的主要作用是系统启动的时候能够自动加载软RAID，同时也方便日后管理。“mdadm.conf”文件内容包括：由 DEVICE 选项指定用于软 RAID 的所有设备，和 ARRAY 选项所指定阵列的设备名、 RAID 级别、阵列中活动设备的数目以及设备的 UUID 号。生成RAID 配置文件操做如下：

**[root@localhost ~]# mdadm --detail --scan > /etc/mdadm.conf**

但是当前生成“mdadm.conf”文件的内容并不符合所规定的格式，所以也是不生效的，这时需要手工修改该文件内容为如下格式：

**[root@localhost ~]# vim /etc/mdadm.conf**

**DEVICE /dev/sdb1 /dev/sdb2 /dev/sdb3 /dev/sdb4**

**ARRAY /dev/md5 metadata=1.2 spares=1 name=server01:5 UUID=307f1234:08c3d0ec:c31390d1:cbbecfc9 auto=yes**

如果没有创建 RAID 的配置文件，那么在每次系统启动后，需要手工加载软 RAID 才能使用，手工加载软 RAID 的命令是：

**[root@localhost ~]# mdadm --assemble /dev/md5 /dev/sdb1 /dev/sdb2 /dev/sdb3 /dev/sdb4**

### 创建文件系统

接下来就只需要在 RAID 设备上创建文件系统就可使用了，在 RAID 设备上创建文件系统和在分区或磁盘上创建文件系统的方法一样。在设备“/dev/md5”上创建 xfs 的文件系统命令如下：

**[root@localhost ~]# mkfs.xfs /dev/md5**

**meta-data=/dev/md5 isize=512 agcount=8, agsize=130944 blks**

**= sectsz=512 attr=2, projid32bit=1**

**= crc=1 finobt=0, sparse=0**

**data = bsize=4096 blocks=1047040, imaxpct=25**

**= sunit=128 swidth=256 blks**

**naming =version 2 bsize=4096 ascii-ci=0 ftype=1**

**log =internal log bsize=4096 blocks=2560, version=2**

**= sectsz=512 sunit=8 blks, lazy-count=1**

**realtime =none extsz=4096 blocks=0, rtextents=0**

**[root@localhost ~]# mkdir /data**

**[root@localhost ~]# mount /dev/md5 /data**

**[root@localhost ~]# df -Th /data**

**文件系统 类型 容量 已用 可用 已用% 挂载点**

**/dev/md5 xfs 4.0G 33M 4.0G 1% /data**

创建完文件系统后，将该设备挂载上就可正常的使用了。如果要创建其它级别的 RAID，其步骤和创建 RAID 5 基本都一样，区别在于指定“--level”值的时候，需要将该值设置为相应的级别。

设置开机自动挂载，在/etc/fstab中加入下面一行：

**/dev/md5 /data xfs defaults 0 0**

挂载测试

**[root@localhost ~]# umount /dev/md5**

**[root@localhost ~]# mount /dev/md5**

## 维护软RAID

软 RAID 虽然很大程度上能保证数据的可靠性，但是在日常的工作中，有时可能需要对 RAID 进行调整以及不排除 RAID 设备物理介质损坏的可能等相关问题，当遇到这些情况时，那么同样可以通过“mdadm”命令来完成这些操作。下面也将通过一个实例来介绍更换 RAID 故障磁盘的完整过程。

1、模拟故障磁盘

假设其中的“/dev/sdb2”设备出现故障时，更换一个新的磁盘，整个过程的详细说明如下：

在实际中，当软 RAID 检测到某个磁盘有故障时，会自动标记该磁盘为故障磁盘，并停止对故障磁盘的读写操作，所以这里需要将/dev/sdb2标记为出现故障的磁盘，命令如下：

**[root@localhost ~]# mdadm /dev/md5 --fail /dev/sdb2**

**mdadm: set /dev/sdb2 faulty in /dev/md5**

由于RAID 5 设置了一个备用设备，所以当有标记为故障磁盘的时候，备用磁盘会自动顶替故障磁盘工作，阵列也能够在短时间内实现重建。通过“/proc/mdstat”文件可查看到当前阵列的状态，如下：

**[root@localhost ~]# cat /proc/mdstat**

**Personalities : [raid6] [raid5] [raid4]**

**md5 : active raid5 sdb3[4] sdb4[3] sdb2[1](F) sdb1[0]**

**41928704 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/2] [U\_U]**

**[==>..................] recovery = 11.9% (2503664/20964352) finish=10.0min speed=30455K/sec**

**unused devices: <none>**

以上信息表明阵列正在重建， 当一个设备出现故障或被标记故障时，相应设备的方括号后将被标以(F)，如“sdb2[1](F)”，其中“[3/2]”的第一位数表示阵列所包含的设备数，第二位数表示活动的设备数，因为目前有一个故障设备，所以第二位数为 2；这时的阵列以降级模式运行，虽然该阵列仍然可用，但是不具有数据冗余；而“[U\_U]”表示当前阵列可以正常使用的设备是/dev/sdb1和/dev/sdb3，如果是设备“/dev/sdb1”出现故障时，则将变成[\_UU]。

重建完数据后，再次查看阵列状态时，就会发现当前的 RAID 设备又恢复了正常，如下：

**[root@localhost ~]# cat /proc/mdstat**

**Personalities : [raid6] [raid5] [raid4]**

**md5 : active raid5 sdb3[4] sdb4[3] sdb2[1](F) sdb1[0]**

**41928704 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/3] [UUU]**

**unused devices: <none>**

### 移除故障磁盘

既然“/dev/sdb2”出现了故障，当然要移除该设备，移除故障磁盘的操作如下：

**[root@localhost ~]# mdadm /dev/md5 --remove /dev/sdb2**

**mdadm: hot removed /dev/sdb2 from /dev/md5**

其中“--remove”表示移除指定 RAID 设备中的某个磁盘，也可用“-r”来代替该参数。

### 添加新硬盘

在添加新的硬盘前，同样需要对新硬盘进行创建分区的操作，例如，添加新硬盘的设备名为“/dev/sdb2”，则具体操作如下：

**[root@localhost ~]# mdadm /dev/md5 --add /dev/sdb2**

**mdadm: added /dev/sdb2**

其中“--add”与前面的“--remove”其义刚好相反，用于将某个磁盘添加到指定的设备中，也可用“-a”代替该参数。

由于RAID 5 设置了一个备用设备，所以不需要做任何操作 RAID 5 也能正常运行，但是如果这时某块磁盘再出现故障的话，会导致 RAID 5 没有数据冗余功能，这对于存放重要的数据的设备来说显得太不安全了。那么这时增加到 RAID 5 中的“/dev/sdb2”则作为备用设备出现在阵列中，如下：

**[root@localhost ~]# mdadm --detail /dev/md5**

**/dev/md5:**

**Version : 1.2**

**Creation Time : Tue Oct 18 22:59:32 2016**

**Raid Level : raid5**

**Array Size : 41928704 (39.99 GiB 42.93 GB)**

**Used Dev Size : 20964352 (19.99 GiB 21.47 GB)**

**Raid Devices : 3**

**Total Devices : 4**

**Persistence : Superblock is persistent**

**Update Time : Tue Oct 18 23:45:36 2016**

**State : clean**

**Active Devices : 3**

**Working Devices : 4**

**Failed Devices : 0**

**Spare Devices : 1**

**Layout : left-symmetric**

**Chunk Size : 512K**

**Name : localhost.localdomain:5 (local to host localhost.localdomain)**

**UUID : 307f1234:08c3d0ec:c31390d1:cbbecfc9**

**Events : 39**

**Number Major Minor RaidDevice State**

**0 8 17 0 active sync /dev/sdb1**

**3 8 20 1 active sync /dev/sdb4**

**4 8 19 2 active sync /dev/sdb3**

**5 8 18 - spare /dev/sdb2**

### 测试：

**[root@localhost ~]# yum -y install hdparm**

**[root@localhost ~]# hdparm -t /dev/md5**

**/dev/md5:**

**Timing buffered disk reads: 1314 MB in 3.00 seconds = 437.64 MB/sec**

**[root@localhost ~]# hdparm -t /dev/sda**

**/dev/sda:**

**Timing buffered disk reads: 58 MB in 3.00 seconds = 19.32 MB/sec**

## 阵列卡介绍（raid控制器）

阵列卡就是用来实现RAID功能的板卡，通常是由I/O处理器、硬盘控制器、硬盘连接器和缓存等一系列组件构成的。

不同的RAID卡支持的RAID功能不同，例如支持RAID 0、RAID 1、RAID 5、RAID 6、RAID 10等，RAID卡的接口类型：IDE接口、SCSI接口、SATA接口和SAS接口。

缓存（Cache）是RAID卡与外部总线交换数据的场所，RAID卡先将数据传送到缓存，再由缓存和外边数据总线交换数据。它是RAID卡电路板上的一块存储芯片，与硬盘盘片相比，具有极快的存取速度。

缓存的大小与速度是直接关系到RAID卡的实际传输速度的重要因素，大缓存能够大幅度地提高数据命中率从而提高RAID卡整体性能。不同的RAID卡出厂时配置的内存容量不同，一般为几兆到数百兆容量不等。

基于不同的架构，RAID 又可以分为:

     ● 软件RAID

     ● 硬件RAID

     ● 外置RAID (External RAID)

     软件RAID很多情况下已经包含在系统之中，并成为其中一个功能，如Windows、Netware及Linux。软件RAID中的所有操作皆由中央处理器负责，所以系统资源的利用率会很高，从而使系统性能降低。软件RAID是不需要另外添加任何硬件设备，因为它是靠你的系统，主要是中央处理器的功能提供所有现成的资源。

     硬件RAID通常是一张PCI卡，你会看到在这卡上会有处理器及内存。因为这卡上的处理器已经可以提供一切RAID所需要的资源，所以不会占用系统资源，从而令系统的表现可以大大提升。硬件RAID可以连接内置硬盘、热插拔背板或外置存储设备。无论连接何种硬盘，控制权都是在RAID卡上，亦即是由系统所操控。在系统里，硬件RAID PCI卡通常都需要安驱动程序，否则系统会拒绝支持。

        外置式RAID也是属于硬件RAID的一种，区别在于RAID卡不会安装在服务器内，而是安装在外置的存储设备内。而这个外置的储存设备则会连接到系统的SCSI卡上。系统没有任何的RAID功能，因为它只有一张SCSI卡；所有的RAID功能将会移到这个外置存储里。好处是外置的存储往往可以连接更多的硬盘，不会受系统机箱的大小所影响。而一些高级的技术，如双机容错，是需要多个服务器外连到一个外置储存上，以提供容错能力。

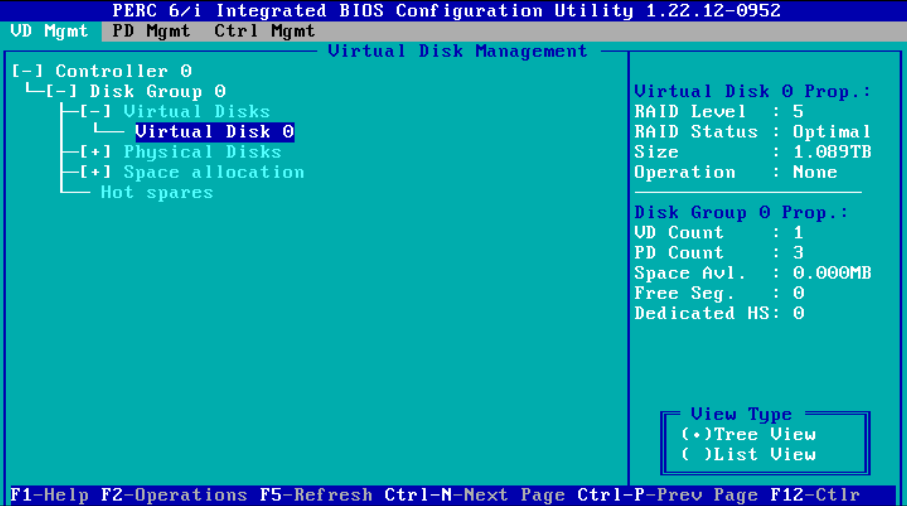
### 硬RAID创建步骤

#### DELL R710服务器案例：

1、开机等待出现以下界面后看屏幕提示，准备按组合键（DELL ：Crtl+R IBM：Crtl+H）进入 raid 配置界面



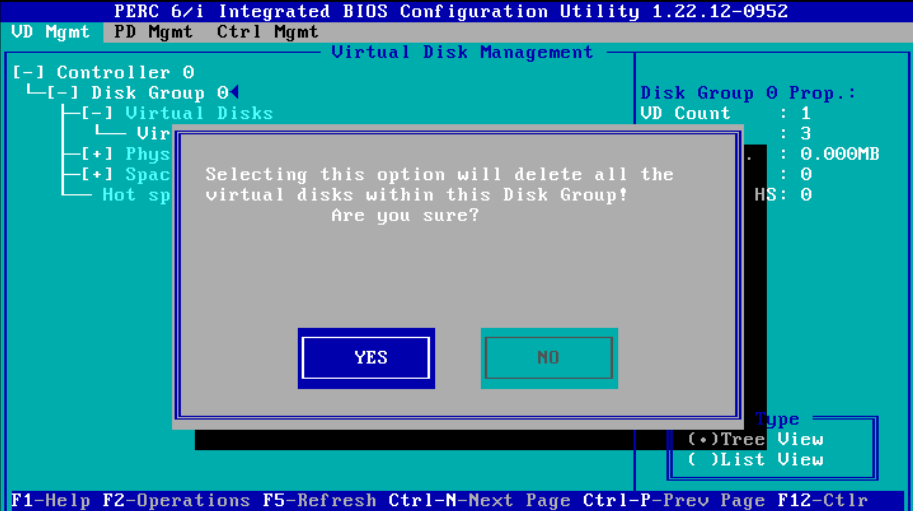
2、当 dell 界面过去后屏幕会有相关组合键提示，同时按下 ctrl + r 进入 raid 配置界面，根据经验建议多按几次，进入到如下的配置界面，在图的右侧可以看到默认为 raid 5，磁盘大小等信息。



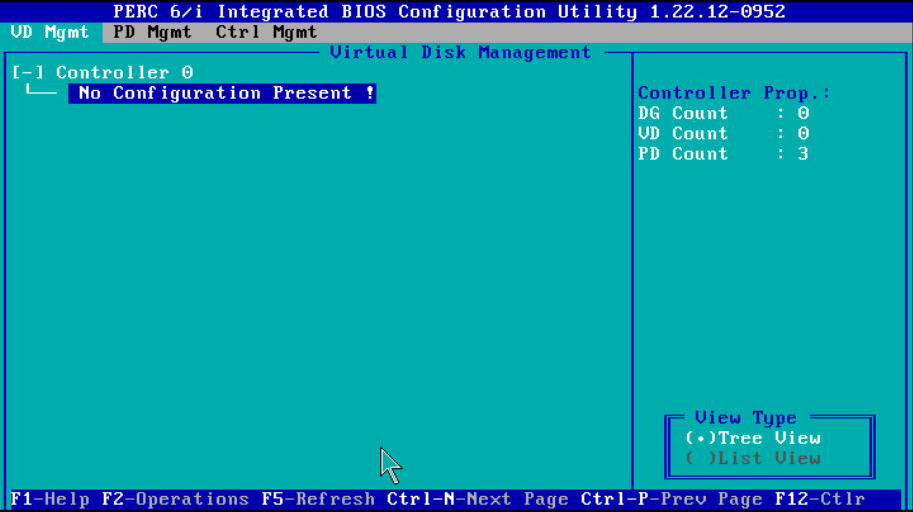
3、使用上下键移动到第二行的位置按下 F2,出现以下窗口，选择第二项（ Delete Disk Group）然后回车，删除 raid5



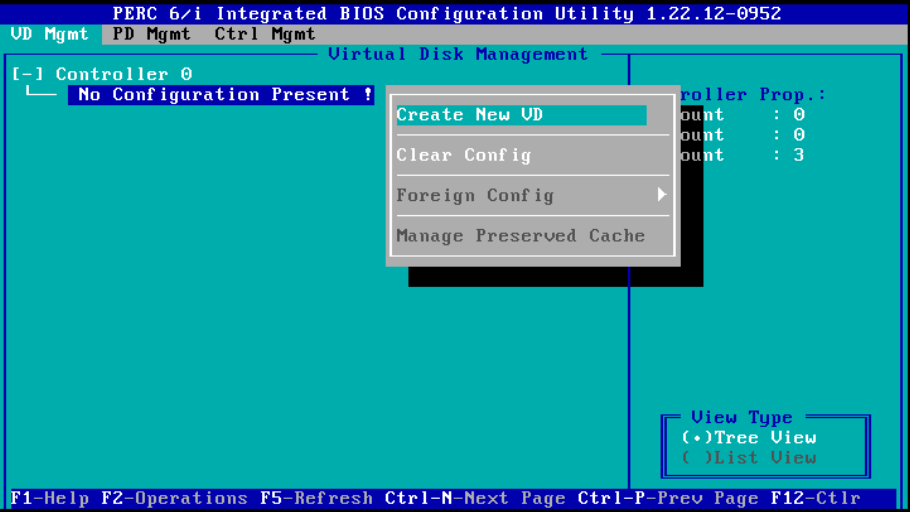
出现确认窗口，选择 yes确认。



4、删除 raid 5 后的状态：



5、然后按 F2 选择第一项（Create New VD）创建新的RAID



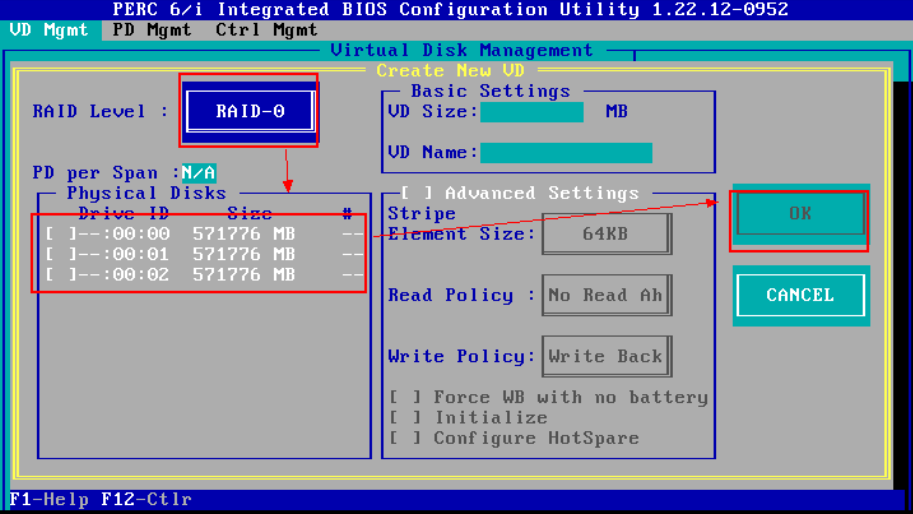
6、进入到下面的界面，蓝色体的为选中状态，此时光标移动到 RAID level 那一栏，继续回车



7、此时出现三个选项，分别是 RAID-0 RAID-1 RAID-5 说明只能做 raid 0 和 raid1 和 raid5 ， raid10 需要至少 4 块磁盘，所以这里没有 raid10 的选项。



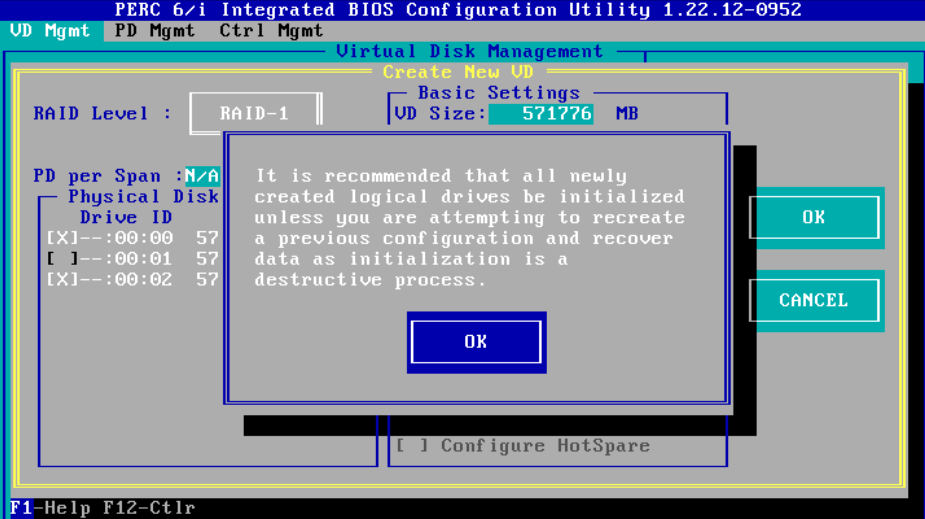
8、选择需要做的选项比如 RAID-0 ，然后在三块硬盘中选择需要做的硬盘， tab 键移动到该位置选择用空格键就可以，最后选择 ok



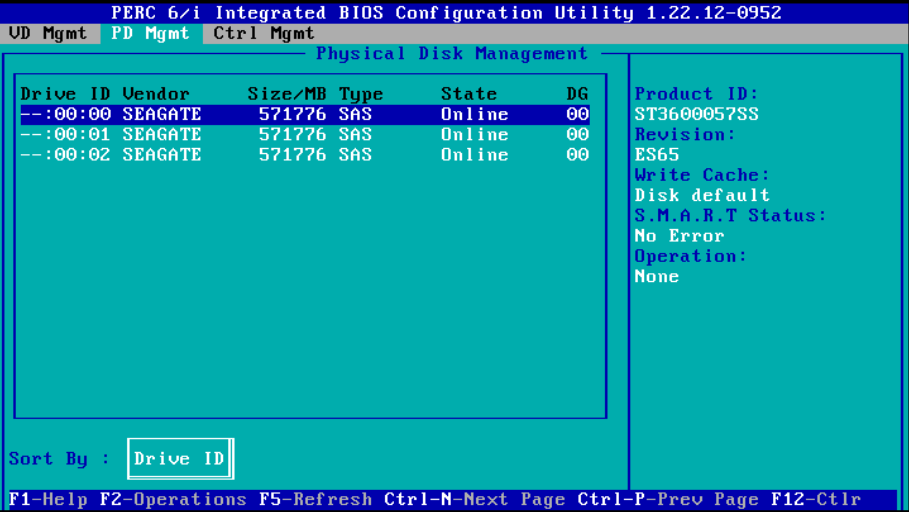
RAID-1 只能选择两块硬盘，如下图 （ raid-0 和 raid-5 都可以选择 3 块）

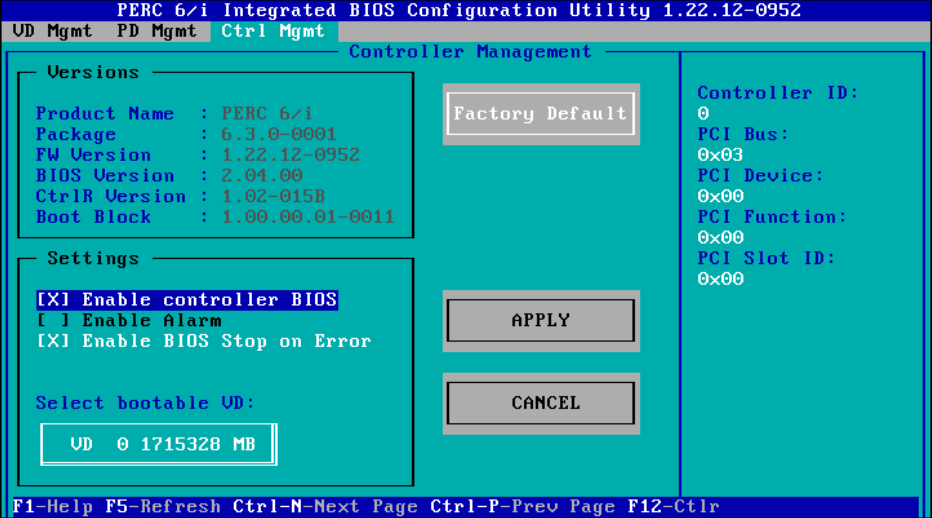


然后再选择 ok

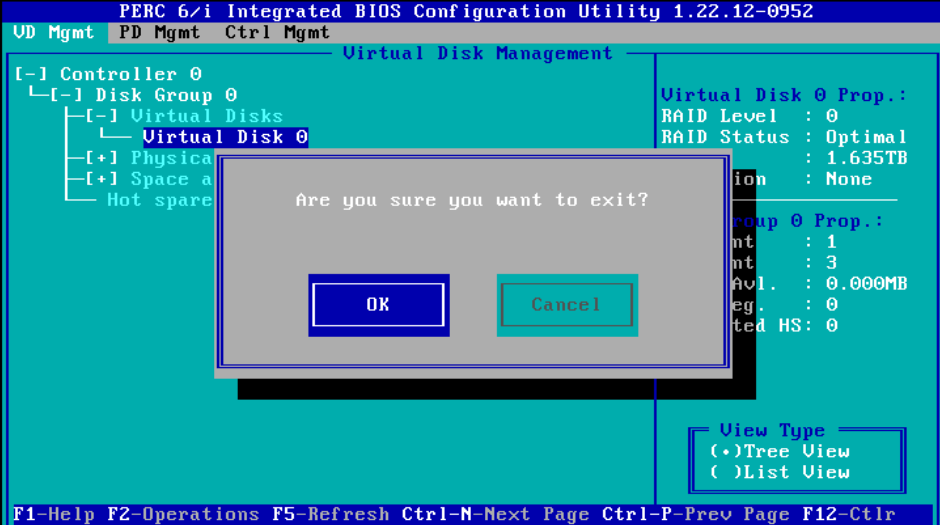


然后按 ctrl + n 可以翻屏查看其他信息

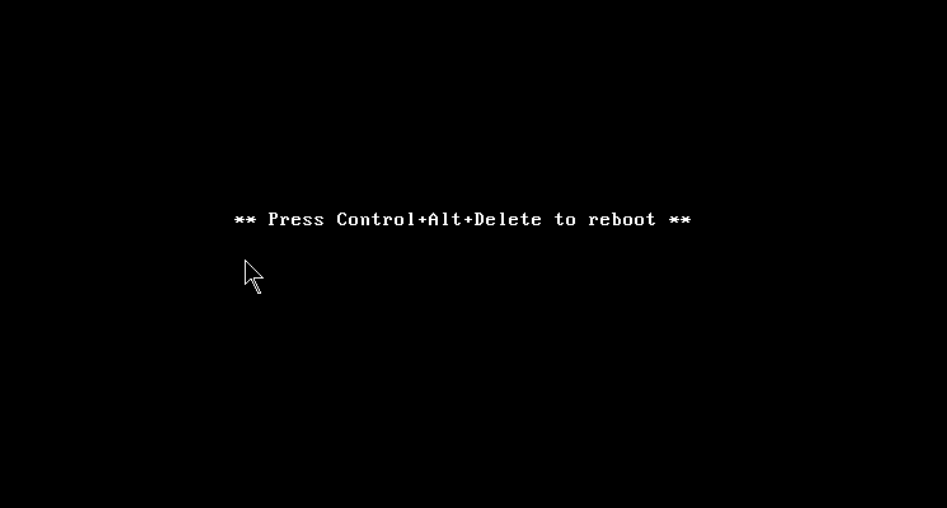




最后按 esc 准备保存退出，弹出下图窗口选择 ok



出现以下界面后，根据提示操作就可以了



在实际生产环境中系统、数据库、应用建议放到不同的分区中，这样有利于系统的维护和对数据应用的调用，磁盘阵列在安装系统之前，系统会视之为一个大型硬盘，而它具有容错及冗余的功能。磁盘阵列可以支持容量扩展，方法也很简单，只需要加入一个新的硬盘并执行一些简单的指令，系统便可以实时利用这新加的容量。