

第8章 Linux相关

Linux存储管理基础：如何理解I/O？

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-07-04

Linux存储管理系统是如何处理I/O的？本文介绍了在操作系统的应用层和存储设备终端生成I/O的基本步骤和属性以及I/O类型。

通常定义的I/O（或Input/Output）是指在电脑和设备之间执行一次输入和/或输出操作的能力。输入设备可以是键盘或鼠标，基于输入设备，更新光标坐标的监视器可作为输出设备。在数据存储方面，I/O通常表示输入和输出磁盘设备的数据流（如：块设备）。

块设备是指能够处理文件系统和/或存储非易失性数据的设备(即：硬盘驱动，CD-ROM，软盘等)。在现代计算机系统里，块设备只能处理一次传输整个或多个块（或扇区）的I/O操作，通常为512字节长度（或2的指数倍）。块设备可以通过本地、网络或通过物理设备接口共享挂载的文件系统节点来访问。使用文件系统的优点是增加数据存储的组织结构特性使其可以同时被用户和操作系统（及其应用程序）访问。文件系统因众多的功能模块而日趋先进，用户能够管理冗余并在高性能环境下进行操作。没有文件系统，物理设备也只是一系列无意义的字节值。也将不会出现识别文件位置的源数据，包括文件相关信息如：文件大小、权限等。

回到I/O处理。有多种方式用于初始化存储设备的I/O处理。在操作系统的应用层和存储设备终端生成I/O的基本步骤包括：

- 打开设备或文件
- 设置读取或写入地址
- 执行读或写操作
- 重复执行步骤2和步骤3
- 关闭设备或文件

尽管这些步骤看起来较为简单，但仍需采用很多变量以定义I/O处理的行为。这些变量通常称为I/O属性。配置I/O属性的其中一部分参数包括：

- 传送大小 —— 需传送字节数或块数。
- 寻址方式 —— 顺序、随机或混合。
- 范围 —— 执行并维护I/O操作及其大小的区域。例如：磁盘设备的前100个块或创建长度小于1GB的文件。
- 进程数量 —— 产生同时运行在磁盘设备之上的I/O的进程总数。包括最大主机数及对中断存储设备进行I/O操作的进程数。
- 数据模式 —— 载入读写缓存并写入/读出磁盘设备的数据模式。
- 时序 —— I/O生成的速率包括读写操作的不同耗时差异。

以上列出了构成I/O属性的大部分参数，I/O属性也包括了主机总线适配器HBA的类型及配置，SCSI层队列深度，SCSI磁盘的超时限制，如果磁盘设备位于阵列中，用户还需考虑磁盘设备的stripe/chunk大小甚至RAID类型等其他因素。理解I/O属性的构成在研发，设计和故障定位中显得尤为重要。

当用户空间发起一个需要读出/写入磁盘设备时，需进入内核空间以恢复该进程。Linux系统I/O分为以下两类：

1.文件I/O

对于内核而言，所有打开的文件都通过文件描述符引用。文件描述符通常是一个小的非负整数，内核用它标识一个特定进程正在访问的文件。当内核打开一个已有文件或创建一个新文件时，它返回一个文件描述符。

可用的文件I/O函数——打开文件，读文件，写文件等。UNIX系统中的大多数文件I/O只需用到5个函数：open、read、write、lseek以及close。它们是不带缓冲的I/O，都使用文件描述符。在使用read和write函数时，选定不同大小的缓冲区（保存读和写的数据），效率是不同的。存在一个最佳效率的缓冲区大小，就是缓冲区大小等于文件系统的块长。

2.标准I/O

对于标准I/O库，它们的操作则是围绕流进行的。当用标准I/O库打开或创建一个文件时，使用一个流与一个文件相关联。当打开一个流时，标准I/O函数fopen返回一个指向FILE对象的指针。该对象通常是一个结构，它包含了标准I/O库为管理该流所需要的所有信息，包括：用于实际I/O的文件描述符，指向用于该缓冲区的指针、缓冲区的长度、当前在缓冲区中的字符数以及出错标志等。

预定义了三个标准I/O流，分别为三个文件指针stdin，stdout和stderr。

标准I/O库提供缓冲的目的是尽可能减少使用read和write调用的次数。它对每个I/O流自动地进行缓冲管理，从而避免了应用程序需要考虑这一点所带来的麻烦。

标准I/O提供三种类型的缓冲：全缓冲、行缓冲和不带缓冲。

- 1) 全缓冲：直到缓冲区填满，才调用系统I/O函数。
- 2) 行缓冲：直到遇到换行符\n才调用系统I/O函数。
- 3) 无缓冲：没有缓冲区，数据会立即读入或者输出到外存文件和设备上。

AIX主机逻辑卷管理器（LVM）概念详解：卷组、物理/逻辑卷、分区

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-07-05

本文主要介绍逻辑卷管理器成员的概念及其与应用程序与物理层之间的关联。

逻辑卷管理器（LVM）通过将数据在存储空间的逻辑视图与实际的物理磁盘之间进行映射来控制磁盘资源。实现方式是在传统的物理设备驱动层之上加载一层磁盘设备驱动代码。该磁盘存储逻辑视图供应用程序使用并独立于底层物理磁盘结构。

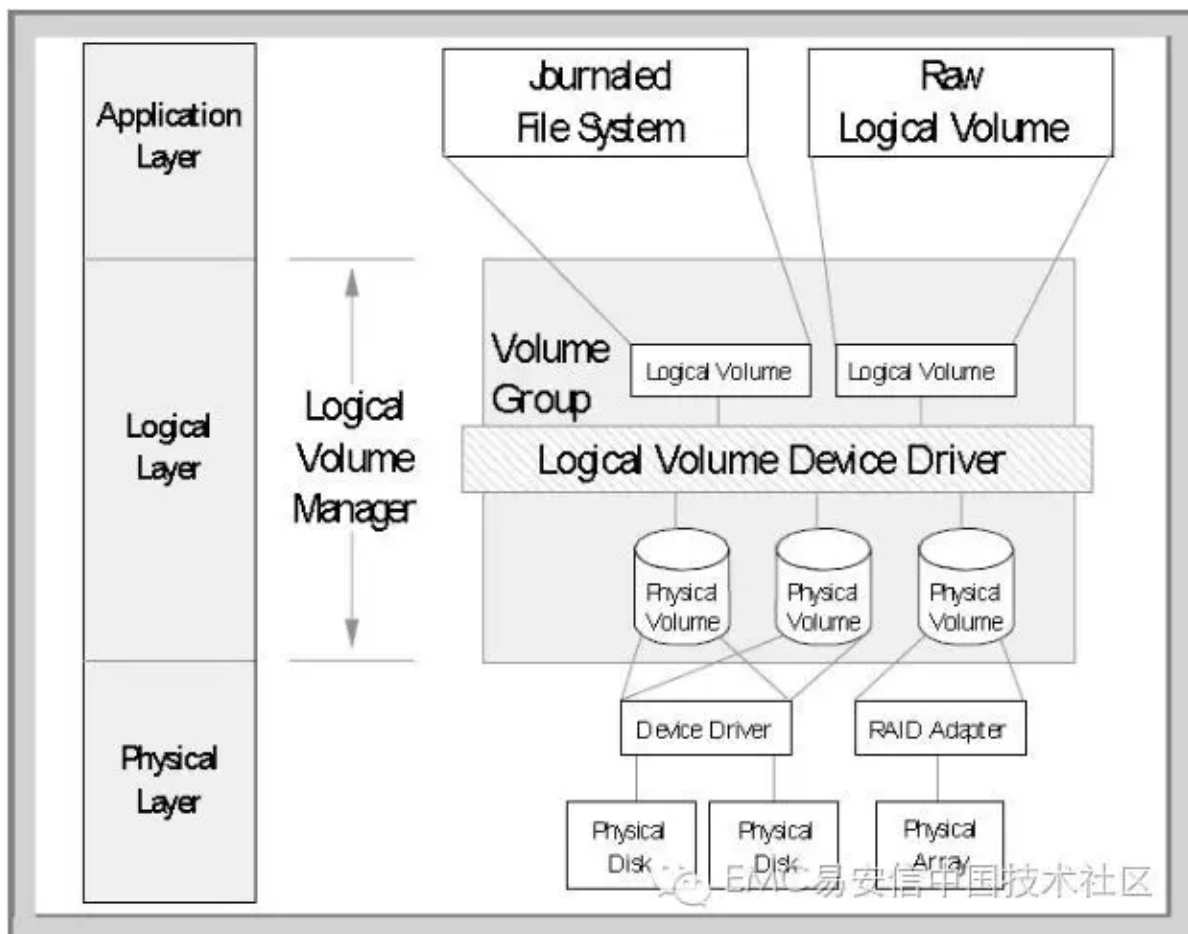


图1. 物理层和应用层之间的LVM适配图

如下结构层次图描述了固定磁盘存储的管理，各层级之间有明确定义的映射关系（包括卷组（datavg），逻辑卷（lv04和mirrLv），逻辑分区（LP1，...），物理卷（hdisk9），和物理分区（PP8））。

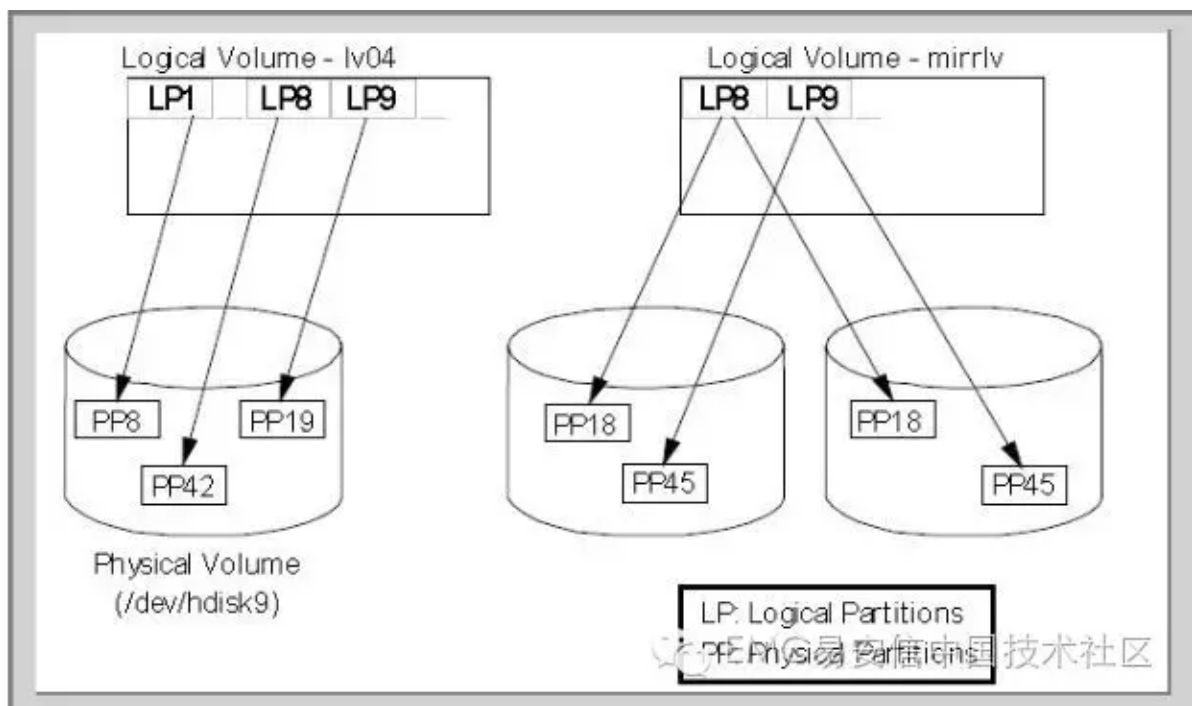


图2. LVM各成员之间的关系

每一个单独的磁盘设备被称为物理卷并赋予一个名称，通常是/dev/hdiskx（x是系统中唯一的整数值）。每一个正在使用的物理卷都属于一个卷组（VG），除非它是作为原始存储磁盘设备或现成备份盘（通常称为热备盘）。每个物理卷包含一定数量相互叠加的磁盘（或盘片），被分成固定大小的物理分区。出于空间分配的考虑，将每一个物理卷划分成五个区域：（外部边缘，外中间，中心，内中间，以及内部边缘），这些可看做通过磁盘盘片垂直切割的圆柱段（参见图3）。每一区域的物理分区数随着磁盘设备总容量而变化。

逻辑卷管理器（LVM）是操作系统命令、库子程序、其他允许用户建立和控制逻辑卷存储的集合。如前文所述，逻辑卷管理器（LVM）通过将数据在存储空间的逻辑视图与实际的物理磁盘之间进行映射来控制磁盘资源。实现方式是在传统的物理设备驱动层之上加载一层磁盘设备驱动代码。该磁盘存储逻辑视图供应用程序使用并独立于底层物理磁盘结构。

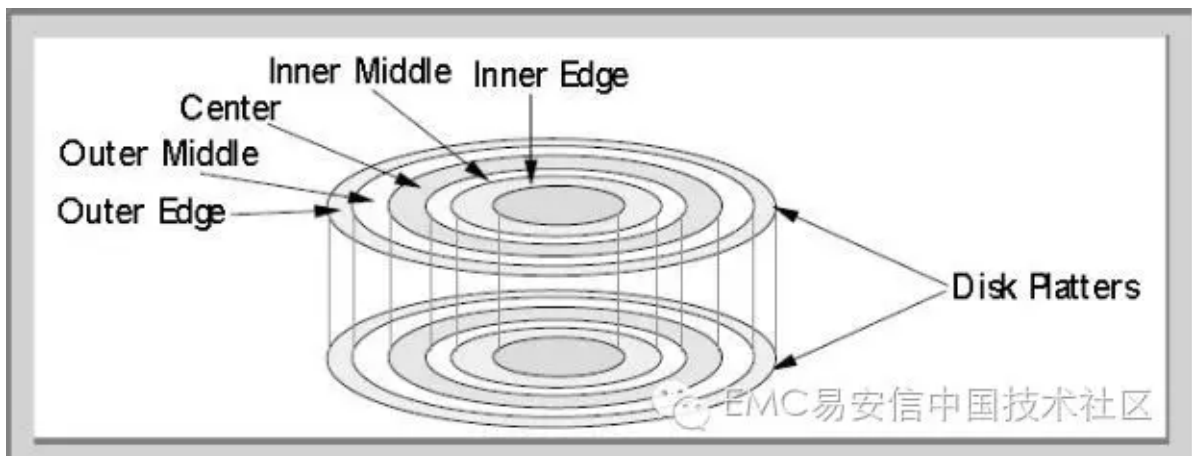


图3. 物理卷区域

逻辑卷管理器（LVM）通过同样的方式来管理RAID磁盘阵列。RAID阵列被当做单一磁盘来处理，即使在绝大多数情况下它的容量相当可观。

安装成功后，系统有一个卷组（root卷组称为rootvg），包括了启动系统所必须的逻辑卷以及安装脚本指定的其他逻辑卷。连接到系统的其他物理卷可被添加到卷组中（使用extendvg命令），或用于创建新卷组（使用mkvg命令）。

卷组和物理卷有如下关系：

- 单个系统上，一个或多个物理卷可组成一个卷组。
- 物理卷不能在卷组之间共享。
- 整个物理卷成为卷组的一部分。
- LVM独立于物理卷。因此不同类型的物理卷可组成一个卷组。

每个卷组内，定义了一个或多个逻辑卷。逻辑卷是用来存储数据的磁盘区域，对于应用来说是连续的，但是在实际物理卷上有可能非连续。逻辑卷可以扩展、重定位、跨越多个物理卷，并且其内容可以复制以提供更好地灵活性和可扩展性。

每一个逻辑卷包含一个或多个逻辑分区（LPs）。每一个逻辑分区对应至少一个物理分区。如果逻辑卷是镜像保护的，那么系统会分配额外的物理分区用以存储各逻辑分区的复制数据。为了保证可用性，这些数据通常位于不同的物理卷，但有时出于性能的考虑，也可能位于同一物理卷。

逻辑卷可通过命令或SMIT菜单形式创建或修改。

逻辑卷只能属于一个卷组。一个逻辑卷可以：

- 位于一个物理卷
- 跨越一个卷组的多个物理卷
- 多个镜像位于同一卷组的不同物理卷

对所有概念做一个总结：

- 物理卷：一个存储磁盘设备，可被分为多个物理分区
- 卷组：一个或多个物理卷的集合，独立于类型
- 逻辑卷：一组逻辑分区的集合，每一逻辑卷可映射到卷组内任一物理分区。如果使用镜像，逻辑分区映射到2个或3个物理分区。
- 逻辑卷管理器：通过逻辑卷磁盘驱动控制上述成员。它负责管理物理分区的复杂结构，包括镜像，对用户/应用呈现单一的逻辑分区。

磁盘分区对齐详解与配置 - Linux篇

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-06-15

在之前一篇[《磁盘分区对齐详解与配置 - Windows篇》](#)中，我介绍了磁盘分区对齐的作用和适用于MBR和GPT的两种磁盘类型的配置，以及Windows平台设置磁盘分区对齐的方法。

本文作为系列的第二篇，文章就分区对齐在存储系统缓存和Raid5下I/O分析，解释了为什么64KB作为offset错位的推荐配置。并且提供了使用Linux命令fdisk和parted创建磁盘分区对齐的方法。

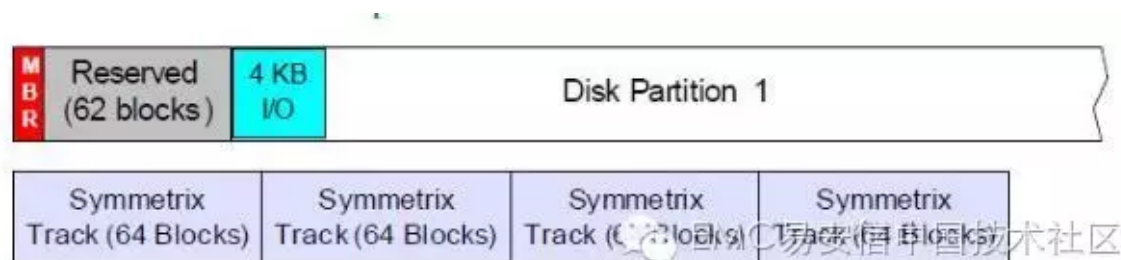
存储系统的磁道区域与Linux分区对齐问题:

出于最大化性能需求，任何到后端存储阵列的I/O需要通过配置适当的结构化，从而避免任何I/O操作跨界现象的发生。如果I/O操作跨越多个界限，会带来额外的资源开销从而造成性能下降。所以，为了避免因为分区不对齐所造成的性能影响。需要使用fdisk或者parted工具创建和对齐分区。

如何选择一个正确的磁盘分区offset值，基本上EMC和其他厂商建议的值都是“64KB”。为什么要选择64KB这个值呢，这里解释一下。首先讲一下一个叫做物理磁道区域的概念。因为对于不同类型的保护级别，也可以说叫Raid和磁盘类型，这个区域的范围有所不同的。我们举两个例子：

Symmetrix缓存中的磁道区域单位大小为一个Track，即64个Block、32KB大小。如图2所示，如果磁盘分区没有被对齐，则任何32KB或者更大的I/O就需要跨越两个Track来进行，50%的16KB的I/O可能会跨越两个Track，25%的8KB也会跨越两个Track，造成额外的存储系统性能开销。

图1



看了第一个例子，读者可能会想，那把起始位右移一个Block就好了啊，(MBR+Reserved=63Block)，就不会有跨Track的I/O发生了。接下来我们再看另外一个例子。一个3+1 Raid5的单个条带大小为四个Track，即256个Block、128KB大小。如果对于这类的磁盘，使用第64个block为起始位置，当linux I/O大小达到64KB的时候，如果I/O直接从缓存（单个track为32KB），则正好完成两次读取。但是如果，两个连续的64KB I/O，且需要牵涉到后端Raid5的物理磁盘读写，如下图所示，第二个64KB就会出现跨越两个条带的情况发生，从而倒是读或者写的开销加倍。

图2

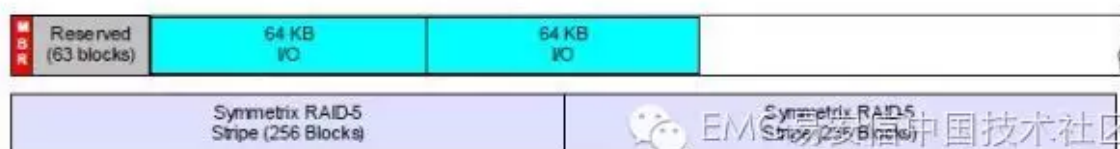
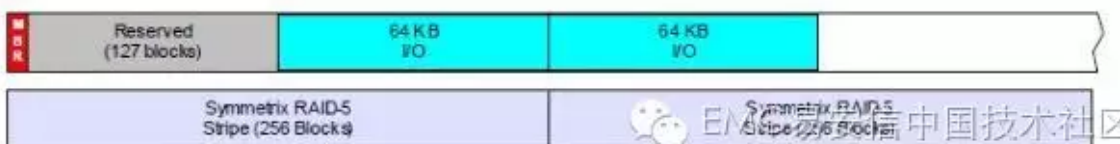


图3

在这种情况下，需要将起始位置调节成建议配置（64KB），这样一来，Linux最大I/O大小的情况也不会发生跨多个条带的情况发生了。（图3）



所以说，无论是从存储系统的缓存中读取数据，还是I/O在缓存中不存在的情况，需要从底层物理磁盘上读取数据。对于不同类型的磁盘，64KB的起始为一个建议配置。

Linux创建分区对齐方法：

描述了磁盘分区对齐的原理后，下面介绍如何使用fdisk创建对齐分区的例子。在Linux中，对齐分区操作需要清空数据的情况下进行，因为对齐分区操作会清空分区表并且该LUN上的数据会被删除。在这个例子中，我们对/dev/emcpowerfw设备，创建一个大小为51281 Cylinder（Cylinder是Symmetrix的计量单位，每个Cylinder大小为960KB，所以这个磁盘大小为50GB左右）、Offset错位大小为128个block的磁盘设备。方法和命令输出（图4）如下

方法1，使用fdisk创建分区对齐

Linux命令提示符下输入：

```
# fdisk /dev/emcpowerfw
```

输入n，创建一个分区：

输入p，创建分区为主分区：

输入起始Cylinder位置，默认为第一个：

输入最后Cylinder位置，默认为该磁盘设备的最后一个Cylinder：

输入x进入expert mode：

输入b，一定分区初始位置：

设定最初位置为128个block（128 block大小为64KB）：

再次输入p确认分区初始位置信息：

输入w保存退出：

图4

```
# fdisk /dev/emcpowerfw
Command (m for help): n [create a new partition]
Command action
   e   extended
   p   primary partition (1-4)
p [create a primary partition]
Partition number (1-4): 1
First cylinder (1-51281, default 1): [Press ENTER to use default]
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-51281, default 51281): [Press ENTER]
Using default value 51281

Command (m for help): x [go into Expert mode]

Expert command (m for help): b [move beginning of partition]
Partition number (1-4): 1
New beginning of data (63-823829264, default 63): 128 [128 blocks => 64KB offset]

Expert command (m for help): p [print new partition table]

Disk /dev/emcpowerfw: 255 heads, 63 sectors, 51281 cylinders

Nr AF Hd Sec Cyl Hd Sec Cyl      Start      Size ID
 1 00 1 1 0 254 62 1023      128 823829127 83
 2 00 0 0 0 0 0 0           0      0 00
 3 00 0 0 0 0 0 0           0      0 00
 4 00 0 0 0 0 0 0           0      0 00

Expert command (m for help): w [write partition table to disk and exit]
```

方法2：使用parted创建对齐分区。

Parted和fdisk相比，支持更多的类型（支持GPT）和更大的分区尺寸。下面一个例子给出一个给dev/sdb磁盘创建128blocc分区起始位的例子，方法和命令输出如下（图5）

Linux命令提示符下输入：

```
# parted /dev/sdb
```

将显示单位调整为Sector（大小512个字节）：

```
(parted) unit s
```

列出当前逻辑卷：

```
(parted) print
```

将原来Number1移除并且创建一个起始位为128 sector，小为976735934 sector的主分区。


```
(parted) rm 1
```

```
(parted) mkpart primary 128 976735934
```

```
(parted) print
```

图5


```
[root@localhost ~]# parted /dev/sdb
GNU Parted 1.8.1
Using /dev/sdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) unit s
(parted) print
Model: APPLE Xserve RAID (scsi)
Disk /dev/sdb: 976748543s
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Number Start End Size Type File system Flags
1 63s 976735934s 976735872s primary
(parted) rm 1
(parted) mkpart primary 128 976735934
(parted) print
Model: APPLE Xserve RAID (scsi)
Disk /dev/sdb: 976748543s
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Number Start End Size Type File system Flags
1 128s 976735934s 976735807s primary
(parted) quit
[root@localhost ~]#
```

 EMC易安信中国技术社区

Linux系统设备驱动入门

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-07-06

本文介绍了Linux版本，内核模块及启动过程，设备驱动基础，以及动态配置磁盘设备的步骤。

Linux版本概述:

Linux是与Unix类似的操作系统，与传统操作系统和花费较高的Unix系统相比，它为个人电脑用户提供了一个免费或价格相对低廉的操作系统。Linux以快速高效著称，与其他服务器操作系统一样，它具有多重应用。

Linux厂商发行版本有 Red Hat，Debian，Fedora，Gentoo，SuSE，TurboLinux 等，其中一部分可以下载。

FTP发行商如：Sunsinte.org.uk, Sunsinte.unc.edu, www.isoimages.org。

可通过以下URL找到Linux相关信息：

www.linux.com 和 www.linux.com

www.xfree86.com

Redhat企业版是一种常见的版本。它支持如Intel X86, Intel Itanium, AMD AMD64 and IBM z系列, POWER 系列, and S/390。Redhat使用最新版本稳定的2.5/2.6 Linux内核。

Redhat发展历程：

1991年，Linux内核发布。同年Bob Yong（加拿大人，多伦多大学毕业）在纽约UNIX用户组引入系统管理自由软件。

1993年，Bob Yang 建立了ACC公司，营销Linux和UNIX的支持软件和书籍杂志。

1994年，Marc Ewing（美国人，卡内基梅隆大学毕业）建立了自己的Linux分销业务，发布了Red Hat Linux 1.0。

1995年，Bob Yang 收购了Marc Ewing的业务，合并后的ACC公司成为新的Red Hat 软件公司，发布了Red Hat Linux 2.0。

1997年12月，Red Hat Linux 5.0发布，它支持Intel、alpha和Sparc平台和大多数的应用软件。极其简单易用的RPM模块化的安装、配置和卸载工具，使程序的安装可在15分钟内完成。软件升级也很方便，这对刚开始使用Linux的用户来说是一大福音。

2003年4月，Red Hat Linux 9.0发布。重点放在改善桌面应用方面，包括改进安装过程、更好的字体浏览、更好的打印服务等。统计表明，2003年，Red Hat的 Linux市场份额为86%。

2004年4月30日，Red Hat公司正式停止对Red Hat 9.0版本的支持，标志著Red Hat Linux的正式完结。原本的桌面版Red Hat Linux发行包则与来自民间的Fedora计划合并，成为Fedora Core发行版本。Red Hat公司不再开发桌面版的Linux发行包，而将全部力量集中在服务器版的开发上，也就是Red Hat EnterpriseLinux版。2005年10月RHEL4发布。

2007年3月，现行主流版本RHEL5发布（最新版本5.5）

2010年4月RHEL6 BETA测试版发布。

2011年04月12日 Oracle发布的Linux系统6.0（基于RedHat Enterprise Linux 6.0）

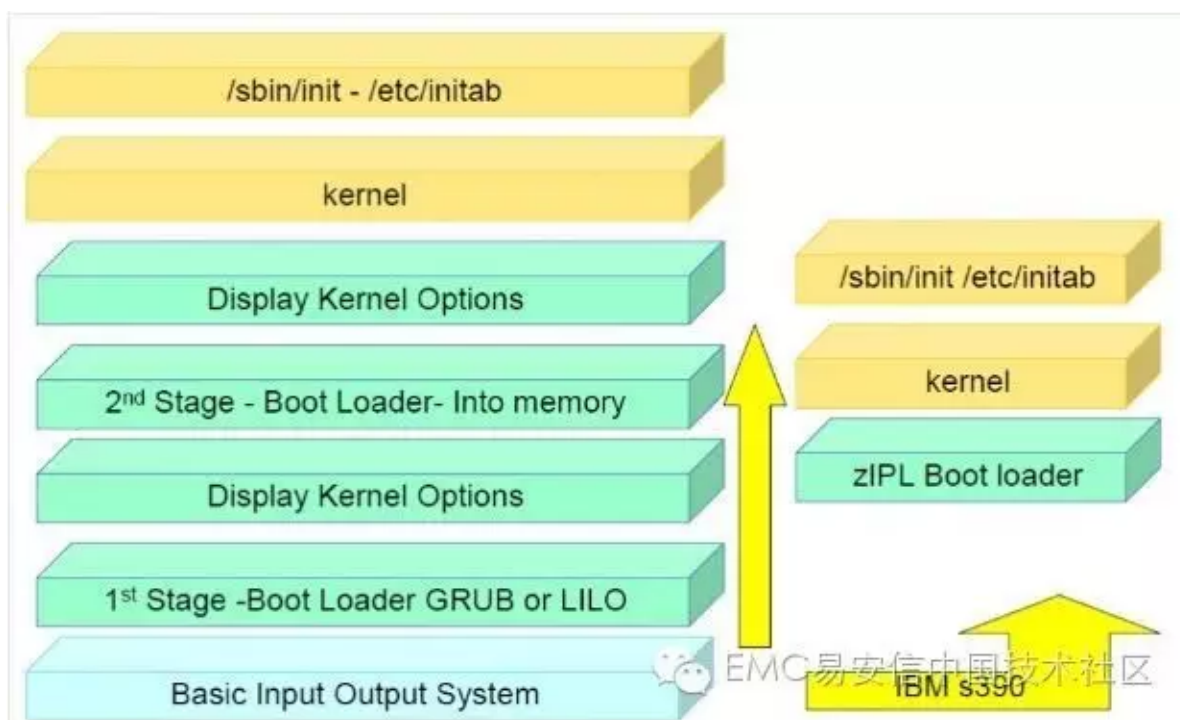
本文中，将采用Redhat企业版作为示例操作系统。

Linux内核模块及启动：

Linux内核采用模块化设计。系统启动时，只有最近贮存的内核会加载到内存中。此后，每当用户请求一个当前内核中不具备的功能，将动态加载一个内核模块（有时称为驱动）到内存中。

在安装过程中，对系统硬件进行探测。基于本次探测及用户提供的信息，安装程序决定在启动时加载哪一种模块。安装程序建立了动态加载机制以实现透明操作。

如果安装结束后新添加了硬件，并且该硬件需要一个内核模块，则系统必须为新硬件配置并加载合适的内核模块。



上图列出了BIOS(Basic Input and Output System)的启动过程。

不同类型的服务器使用了不同的引导装载程序。以上图例中，如果服务器是基于Intel的则引导装载程序可能是GRUB或LILO。如果是mainframe大型机则有可能使用IBM s390引导加载程序。

BIOS发起第一阶段主引导加载程序。BIOS将位于引导介质第一扇区中的程序加载入内存，我们称之为引导记录（Master Boot Record或MBR）。主引导记录只有512字节大小，包含启动设备所需的机器代码，也称为引导加载程序（boot loader）。一旦BIOS将引导加载程序找到并将其加载入内存后，它就将启动程序的控制权交给二级引导加载程序。

在第一阶段，主引导加载程序通过BIOS从主引导记录被读入内存。它的主要工作是加载位于不同介质上任意位置的数据，通过这步操作来定位第二阶段引导加载程序。

第二阶段引导装载程序由第一阶段的引导装载程序发起，它包含有加载程序更需要磁盘空间的部分，如用户界面和内核引导程序。该阶段允许用户选择启动的操作系统或Linux内核。可用的两个启动加载程序是GRUB或LILO。GRUB是较新的启动加载程序，能够阅读ext2和ext3类型文件分区并在启动时加载它的配置文件——/boot/grub/grub.conf。LILO是较老的版本，它使用主引导记录中的信息以确定用户可用的启动选项。这意味着无论何时配置发生更改或内核被手动升级，必须通过命令来讲合适的信息写入MBR。命令为：/sbin/lilo -v -v。

GRUB 与 LILO 的比较

所有引导加载程序都以类似的方式工作，满足共同的目的。不过，LILO 和 GRUB 之间有很多不同之处：

- LILO 没有交互式命令界面，而GRUB 拥有。

- LILO 不支持网络引导，而GRUB 支持。
- LILO 将关于可以引导的操作系统位置的信息物理上存储在MBR 中。如果修改了LILO 配置文件，必须将LILO第一阶段引导加载程序重写到MBR。相对于 GRUB，这是一个更为危险的选择，因为错误配置的MBR 可能会让系统无法引导。使用GRUB，如果配置文件配置错误，则只是默认转到GRUB 命令行界面。

接下来，系统将操作系统加载入内存并将设备的控制权转交给该操作系统。操作系统将会处理/etc/inittab文件，并依据/etc/inittab设定的内容，依序启动相关进程。首先启动的程序为/etc/rc.sysinit。rc.sysinit设置环境变量，启动置换空间，检查文件系统，并执行所有系统初始化所需的其他步骤。例如，绝大多数系统使用时钟，因此rc.sysinit读取/etc/sysconfig/clock配置文件以初始化硬件时钟。另一个例子是如果要初始化串口，rc.sysinit将会执行/etc/rc.serial文件。

Linux磁盘设备驱动基础：

Linux内核中采用可加载的模块化设计，常见的驱动程序是作为内核模块动态加载的。

模块的相关命令：

- lsmod——列出当前系统加载的模块
- rmmod——将当前模块卸载
- insmod——加载当前模块
- mknod——创建相关模块

Linux将设备看作文件，每个设备对应一个文件名，内核中对应一个索引节点，对文件操作的系统调用大都适用于设备文件。对某个具体设备而言，文件操作和设备驱动是同一事物的不同层次。Linux将设备分为两大类，一类是像磁盘那样的以块或扇区为单位、成块进行输入/输出的设备，称为块设备；另一类是像键盘那样以字符（字节）为单位，逐个字符进行输入/输出的设备，称为字符设备；文件系统通常建立在块设备上。

本文将以SCSI磁盘为例来介绍磁盘设备驱动的基础知识。

SCSI（小型计算机系统接口）总线是一种高效的点对点数据总线，它最多可以支持8个设备，其中包括多个主设备。在总线上的两个设备间数据可以以同步或异步方式，在32位数据宽度下传输率为40M字节来交换数据。SCSI总线上可以在设备间同时传输数据与状态信息。

Linux SCSI子系统由两个基本部分组成，每个由一个数据结构来表示。

Host

一个SCSI host即一个硬件设备：SCSI控制权。在Linux系统中可以存在相同类型的多个SCSI控制权，每个由一个单独的SCSI host来表示。这意味着一个SCSI设备驱动可以控制多个控制权实例。SCSI host总是SCSI命令的initiator设备。

Device

虽然SCSI支持多种类型设备如磁带机、CD-ROM等等，但最常见的SCSI设备是SCSI磁盘。SCSI设备总是SCSI命令的target。这些设备必须区别对待，例如象CD-ROM或者磁带机这种可移动设备，Linux必须检测介质是否已经移动。不同的磁盘类型有不同的主设备号，这样Linux可以将块设备请求发送到正确的SCSI设备。

SCSI子系统的初始化非常复杂，它必须反映出SCSI总线及其设备的动态性。Linux在启动时初始化SCSI子系统。如果它找到一个SCSI控制器（即SCSI hosts）则会扫描此SCSI总线来找出总线上的所有设备。然后初始化这些设备并通过普通文件和buffer cache块设备操作使Linux内核的其它部分能使用这些设备。

一旦SCSI子系统初始化完成这些SCSI设备就可以使用了。每个活动的SCSI设备类型将其自身登记到内核以便Linux正确定向块设备请求。

如前所述，一个设备文件（即设备节点）可以通过mknod命令来创建，其中指定了主设备号和次设备号。主设备号表明某一类设备，一般对应着确定的驱动程序；次设备号一般是用于区分标明不同属性，例如不同的使用方法，不同的位置，不同的操作等，它标志着某个具体的物理设备。高字节为主设备号和底字节为次设备号。例如，在系统中的块设备SCSI 磁盘的主设备号是3，而多个SCSI 磁盘及其各个分区分别赋予次设备号1、2、3.....

SCSI系统中磁盘设备驱动层级如下：sd——直接访问磁盘，sg——SCSI通用接口，sr——Data CD-ROMs，st——磁带。sg驱动是基于字符的设备而其他三个驱动都是块设备驱动。sg驱动主要用于扫描仪，刻录机，以及打印机。从第一个SCSI控制器开始，sg设备文件动态映射到SCSI总线上的SCSI IDs/LUNs。

块设备的本地文件名具有以下格式：/dev/sdln，l表示物理设备而n表示该物理设备上的分区号。当主机总线适配器发现附加连接的存储后Linux会在设备文件/dev/sd[l][n]中定义这些设备。当主机总线适配器监测到随机附带存储后Linux将会定义设备文件/dev/sd[l][n]。

按照以上定义，以文件/dev/sda1为例，物理设备是“a”而分区是“1”。Linux内核为SCSI设备保留了16个主设备编号，各主设备编号可拥有0-255个从设备编号。这些从设备编号包括SCSI设备的分区。对于每个磁盘设备Linux支持0到15个分区。其中，1至4为主分区，分区5以上为逻辑分区或扩展分区；以上限制只适用于Intel平台。默认情况下，Linux并不使用slice这一概念。因此，16个主设备编号和16个从设备编号意味着256个SCSI磁盘设备，内核能够扫描范围从1至255的磁盘设备。Red Hat Linux和SuSE SLES 7最大支持128个SCSI设备，而SuSE SLES 8版本支持256个SCSI设备。

动态设备配置步骤：

★★

★★

在Linux内核中，与其他类型的UNIX系统（如：Sun，SGI，HP-UX，BSD）不同，设备名中并没有使用SCSI地址。如前文所述，块设备名格式为/dev/sdln，l是表示物理设备驱动的字符而数字n代表该物理设备驱动的分区号。设备名在启动时或设备加载时按发现顺序动态指定。

如果添加了硬件设备之后系统重新启动，设备编号将会更改从而造成主机的挂载列表不准确。为了保持设备编号的准确性并减少挂在列表出现偏差的可能性，应当把新的设备附加在当前设备列表中。例如，如果主机包含多个HBA，最好将新设备附加在最后一个HBA的磁盘设备列表的末端，这样就无需更改挂载列表中的现有记录。如果新设备添加到第一个HBA中，那么在系统重启之后，所有设备编号都会在原有数字加一同时挂在列表记录也需随着该设备发生偏移。如果只有一个HBA，则新设备可方便地添加到原有设备列表中并相应地改变挂载列表。

目前Linux系统缺乏植入到内核中的、如同drvconfig或ioscan这样能够动态配置SCSI通道的命令。

重新配置Linux主机的三种方式有：

- 重启系统
- 卸载并重新加载HBA驱动模块
- echo /proc文件系统

重启主机：

重启主机是检测新添加磁盘设备的可靠方式。在所有I/O停止之后方可重启主机，同时静态或以模块方式连接磁盘驱动。系统初始化时会扫描PCI总线，因此挂载其上的SCSI host adapter会被扫描到，并生成一个PCI device。之后扫描软件会为该PCI device加载相应的驱动程序。加载SCSI host驱动时，其探测函数会初始化SCSI host，注册中断处理函数，最后调用scsi_scan_host函数扫描scsi host adapter所管理的所有scsi总线。

卸载并重新加载HBA驱动模块：

通常情况下，HBA驱动在系统中以模块形式加载。从而允许模块被卸载并重新加载，在该过程中SCSI扫描函数得以调用。通常，在卸载HBA驱动之前，SCSI设备的所有I/O都应该停止，卸载文件系统，多路径服务应用也需停止。如果有代理或HBA应用帮助模块，也应当中止。

命令示例：

例如，rac节点上某台服务器执行fdisk -l命令看不到共享磁盘，可尝试执行如下命令：

```
# modprobe -r lpfc (卸载驱动)
```

```
# modprobe lpfc (加载驱动)
```

/proc下SCSI扫描：

2.4内核中，/proc文件系统提供了可用SCSI设备的列表。如果系统中SCSI设备重新配置，那么所有这些改变通过echo /proc接口反映到SCSI设备中。添加一个设备，主机，channel，target ID，以及磁盘设备的LUN编号会被添加到/proc/scsi/，需指定scsi编号。

命令示例：

```
# echo "scsi add-single-device 0 1 2 3" > /proc/scsi/scsi
```

0: 主机ID

1: channel ID

2: target ID

3: LUN编号

该命令会将新磁盘设备添加到/proc/scsi/scsi文件中。如果没有找到相应文件，需为/dev路径下新增磁盘设备创建设备文件名。

如果要删除一个磁盘设备，使用适当的主机，channel，target ID及LUN编号运行如下格式命令：

```
# echo "scsi remove-single-device 0 1 2 3" > /proc/scsi/scsi
```

0: 主机ID

1: channel ID

2: target ID

3: LUN编号

细数Linux发行版

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-07-03



与其他由单一公司控制、发行和支持的商业操作系统不同，Linux有许多不同的版本，而且Linux的内核的发行和使用是完全免费的。所以，很多的公司、组织，甚至是个人开发了他们自己Linux操作系统版本。当这些版本面向公众发布的时候。他们被称为“发行版”（Distribution）。

通常一个发行版的诞生是由于某种目的的驱使，并且特地为此而做过优化。一些Linux发行版被安装在承载繁重流量的服务器上，一些Linux发行版被部署到以安全级别为优先的网络环境中，还有一些Linux发行版用于安装到Windows和Mac OS这样的操作系统的上一层。

Linux也经常内嵌在许多硬件设备之中，例如：路由器、交换机、电话系统、智能手机、GPS系统，等等。更多的Linux发行版被用来安装到PC上作为原始操作系统使用。

那么，哪一种Linux发行版才是适合你的呢？答案是：视情况而定。在作出决定之前，不妨问问自己下面几个问题。

- 1) 代码库是恒定不变的吗？
- 2) 操作体验经常改变吗？
- 3) 更新软件包容易吗？
- 4) 升级系统版本容易吗？
- 5) 有很多开发者为之工作吗？

没有“最佳”Linux发行版。对于大部分用户来说，仅仅使用最基本的操作功能，大部分的Linux发行版都能满足，而且安装使用也很简便，就像Microsoft Windows系统。

比较受欢迎的Linux发行版：

- 1) Fedora
- 2) Ubuntu
- 3) Red Hat
- 4) Debian
- 5) SUSE

下面我们来了解下其中几个主要的发行版。



Fedora是一个基于Linux内核的操作系统，集成了一批软件，可以用来替换Microsoft Windows或者Mac OS，而且它是完全免费使用的。

Fedora Project是一个国际社区的名字，它由一群热爱、使用、希望推广免费软件的人组成的。Fedora由Red Hat发起主办的，全球最值得信赖的开源技术提供商。

是什么让fedora与众不同？我们相信是免费软件的价值。不仅仅是Fedora是一款免费软件，而且我们使用免费软件把它提供给所有人。

Fedora的核心价值是什么？



你可以自由地使用和发行免费软件。



你可以获得许多人对于Fedora的支持和帮助。



越多人加入，就会获得更多功能。



无需等待beta版，就能第一时间体验最新的软件。

了解更多，直接登入到：<http://fedoraproject.org>。



Ubuntu是一个古老的非洲单词，意思是“人与人”。他还有一层含义是“我就是我，人人都是我”。而Ubuntu操作系统很好地秉承了这个单词的内涵，将它带到了计算机的世界。

2004年，Linux就已经成为了企业服务器平台，但是他仍然不够亲民。所以一群来自Debian Linux项目的开发人员推出了一个易于使用的桌面系统：Ubuntu。

Ubuntu的愿景是一半面向公众，一半面向商用。作为免费软件，每个人都可以使用，而作为商用软件，Canonical公司为使用者提供一系列支持和服务。

每两年Ubuntu会发行至少四个更新版。Ubuntu与很多商用Linux软件的不同之处在于，它的公众版和商业版看起来是一模一样的，就如同一个的单一的高质量的版本而受到长期的持续的维护。重要的是，两个版本都是完全免费的。

2004年10月，第一个官方Ubuntu版本4.10面市了，成千上万的免费软件爱好者和专家加入到了Ubuntu社区。对于Ubuntu的管理却是由独立于Canonical公司之外的一群志愿者来承担的，所以我们可以把它看作是由Canonical公司、无数志愿者和其他第三方公司共同构建的免费软件的世界。

第一版的Ubuntu是基于GNOME的桌面系统。在添加了一个KDE版后变成了Kubuntu，一个服务器版。所有的Ubuntu版本共享同样的基础架构和软件，无论从消费电子到桌面系统，甚至是企业云计算都是一个统一的平台。

近年来，Ubuntu已经渗透到了许多新领域，诞生了Ubuntu Netbook版本和企业云版本，它在Amazon的EC2和Rackspace的云上非常受欢迎，它被预装到了Dell、Lenovo和其他全球厂商的计算机中。

了解更多，直接登入到：<http://ubuntu.com>。



早在1994年，Red Hat就诞生了，不过直到1999年8月11日，Red Hat才正式出现在公众的视野中。1999年11月15日，Red Hat公司获得了Cygnus提供的解决方案。Cygnus提供免费软件的财政支持并且维持GNU软件产品，例如：GNU Debugger和GNU Binutils。一位Cygnus的创建者还成为了Red Hat公司的CTO。

2000年2月，Infoworld授予Red Hat公司连续第4座“年度操作系统产品”大奖，正是因为Red Hat Linux 6.1。

2002年Red Hat公司推出了Red Hat Linux Advanced Server，后来又命名为Red Hat Enterprise Linux，简称RHEL。Dell、IBM、HP和Oracle公司宣布了对这一平台的支持。

2006年6月Red Hat公司收购了开源中间件提供商JBoss，9月Red Hat公司就发布了Red Hat Application Stack，这是第一个集成了开源中间件技术的版本。

2007年3月Red Hat发布了Red Hat Enterprise Linux 5。

目前，RHEL在商用领域的正被越来越多的用户使用。

了解更多，直接登入到：<http://redhat.com>。



Debian project是由一群希望创造免费操作系统的个人合力推动的。他们称这个操作系统为：Debian。

一个操作系统是一组基本的程序和功能的集合，他们能让电脑运转起来。在操作系统的中心就是kernel，它是最底层的程序，让你能启动其他的程序。

目前的Debian系统使用的是Linux kernel或者FreeBSD kernel。当然，还有其他一些project正在让Debian使用其他的kernel，例如GNU project的免费系统软件Hurd。来自GNU project的大量基础工具正在不断加入到这个些操作系统中，他们的名字也很有特色：GNU/Linux、GNU/FreeBSD、GNU/Hurd。

当然，人们最想要的还是应用软件，从文档编辑器到游戏。Debian上已经有超过29000个各式应用软件包了，而且全部免费。

这就像是一座塔。底座是kernel，上一层是基础工具集合，在上一层是应用程序。而塔顶就是容纳和整合他们的Debian。

了解更多，直接登入到：<http://debian.org>。



openSUSE是一个自由的基于Linux的服务于个人电脑、笔记本或者服务器的操作系统。openSUSE是一个全球项目，致力于让免费的Linux出现在我们工作和生活的每个角落。它的背后同样有一个免费和开源软件的社区。任何有意向的开发者和最终用户都能加入其中。

最新的版本是openSUSE 12.2。这个Linux 发行版通过Linux 3.4 kernel的快速存储层和glibc 2.15基础库中的加速函数带来了（尤其是64位版本）全线加速，并加入了全新的Btrfs文件系统和更新的GNOME3.4系统管理体验。

Linux 3.4 kernel允许跨越整个进程组对CPU使用率进行追踪。新版本的systemd提供一个看门狗功能和一组新的进程管理工具来监管所有的系统服务。这个全新的管理套件叫作Digital Forensics/Incident Response。

了解更多，直接登入到：<http://opensuse.org>。

你准备好使用免费软件了吗？

逻辑卷管理器（LVM）概念解析

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-07-07

在早期的计算机系统中，磁盘驱动器通常将一组连续的物理块分配给操作系统，导致操作系统或应用程序独享整个磁盘驱动器。这种模式的缺点是缺乏灵活性，当硬盘驱动器空间使用完后，无法扩展文件系统的容量。如果当磁盘驱动器存储容量增加时，把整个磁盘驱动器分配给文件系统通常会导致存储空间不能充分使用。为了使文件系统容量能够动态扩展以及高效的存储管理，逻辑卷管理器（LVM）开始发展。本文作为存储基础知识系列文章，将为您介绍逻辑卷管理器。

逻辑卷管理器（LVM）是一个运行在物理机器上管理逻辑或物理存储设备的软件，它在文件系统和物理磁盘之间起到桥梁作用。逻辑卷管理器（LVM）可以把一个大容量物理磁盘划分为若干小的虚拟磁盘，同时也可以把几个小的物理磁盘整合成一个大的虚拟磁盘，形成的这些虚拟磁盘可以给应用程序使用。

在介绍逻辑卷管理器（LVM）组件前，我们先介绍二个基本概念：“分区”和“串联”。

划分物理磁盘的过程叫磁盘分区（disk partitioning），它是为了改善磁盘驱动器的灵活性和使用率。在磁盘分区过程中，磁盘驱动器被划分成几个逻辑卷。通俗的说，就是将一个大的物理磁盘根据文件系统和应用程序的数据管理要求，划分成若干个小的逻辑卷。主机中磁盘进行初始化分区的过程，就是将一组连续的柱面分配给一个分区。当主机中有文件系统被访问是，完全不需要知道物理磁盘的结构和分区信息。

整合物理磁盘的过程叫串联（concatenation），简单说就是把若干小的物理磁盘整合成一个大的逻辑盘。

逻辑卷管理器（LVM）隐藏了物理磁盘的细节和数据在磁盘上的位置，优化了用户对存储的访问，简化了存储资源的管理，在更改存储配置时可以实现无应用程序中断。LVM主要由三部分组成：物理卷（physical volume）、卷组（volume group）和逻辑卷（logical volume）。

物理卷（physical volume）是指每一个连接到主机系统的物理磁盘。LVM将物理卷提供的物理存储空间转换成逻辑存储空间，方便应用程序或操作系统使用。卷组（Volume group）是由一个或多个物理卷组成，LVM在初始化时会为每个物理卷分配一个唯一的物理卷标识（PVID），在卷组使用中，支持物理卷的动态添加或删除，但是一个物理卷只能被一个卷组使用。在创建卷组时，每一个物理卷都会被划分为若干个大小相同的数据块，我们称这些数据块为物理区域（physical extent）。在指定的卷组中可以创建逻辑卷（logical volume），如果每个卷组可以被看成一个磁盘的话，那么每个逻辑卷都可以看成一个虚拟磁盘分区。一个卷组可以划分为多个逻辑卷，逻辑卷的大小取决于物理磁盘的数量和大小。

从操作系统的层面，逻辑卷就像一个“物理设备”，但是这个“物理设备”可以跨多个物理卷，并且由不连续的物理分区组成。在逻辑卷上创建文件系统后，就可以直接将逻辑卷分配给应用程序使用了。

Linux存储管理常见问题与解答

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-07-12

本文汇总了Linux存储管理的常见问题与解答，如：RHEL中如何确保PowerPath boot from SAN，如何在启动时自动加载Linux HBA驱动？供Linux系统管理员参考。

RHEL中如何确保PowerPath boot from SAN：

问题描述：安装PowerPath之后主机无法启动

环境：

EMC SW: PowerPath for Linux 5.6

OS: Red Hat Linux 6.2

原因：RHEL默认安装自带多路径软件，一旦安装之后卸载比较困难。

1. 确保只有一根光纤线缆接入主机。
2. 只在一个HBA上激活BIOS，其他都禁止。
3. 确保主机只能看见一个LUN（这一点非常重要）。
4. 在BIOS中选择HBA 1的LUN 0为启动设备。
5. 启动主机安装RHEL 6.2（不要添加选项，直接安装）。
6. 选择指定的存储设备（这里选SAN）。
7. 选择/dev/sda——需确保它是一块实际的磁盘，而不是LUNZ设备。
8. 安装过程常规进行。安装结束之后重启（与通常情况一样）。
9. 将 /boot/initramfs-xxxxx.img移动至/boot/initramfs-xxxx.orig.img，xxxxx是内核版本。
10. 安装PowerPath，license，然后执行"/etc/init.d/PowerPath start"，查看PowerPath配置。执行powermt save命令。
11. 编辑 /etc/fstab加载从/dev/emcpowera1启动。
12. Remountd /boot。
13. 将LVM filter改为 ["a/emcpower./", "r/sd./", "r/disk.*/"]。
14. 使用"dracut /boot/initramfs-PP-(*uname -r*).img(*uname -r*)"编译一个新的initramfs。
15. 在 /boot/grub/menu.lst添加initramfs-PP-xxxxxx.img。
16. 用阵列masking在配置添加剩余LUN，
17. 接入第二根FC线缆，扫描LUN，执行powermt config，确保所有路径alive。
18. 重启。

如果不是boot from SAN,最佳方法是加载OS而不接SAN线缆（因此绕开dm-multipath安装）。或业务中断，断开FC线缆，重启主机，禁用多路径（通过编译multipath.conf），安装PowerPath，之后重新连接线缆并重启主机。

如何在启动时自动加载Linux HBA驱动:

目的: 如何在系统启动时自动加载Linux HBA驱动

问题描述:

Linux服务器在重启之后没有自动加载HBA驱动

Linux在主机重启之后无法识别设备

Linux服务器在重启之后无法看见阵列

环境:

OS: Red Hat Linux Advanced Server 2.1

OS: Red Hat Linux Advanced Server 3.0

OS: SuSE Linux Enterprise Server (SLES) 8

原因: HBA驱动没有植入RAM磁盘镜像

解决: 在/etc/modules.conf文件中加入HBA驱动并通过mkinitrd重新编译RAM磁盘镜像

注释: 为了在启动时自动加载驱动, 应当在/etc/modules.conf文件中包含该文件:

```
vi /etc/modules.conf
```

为每一个安装的QLogic QLZ2200 HBA添加如下内容:

```
alias scsi_hostadapterN qla2200
```

为每一个安装的QLogic QLA23xx HBA添加如下一行:

```
alias scsi_hostadapterN qla2300
```

N表示系统中每个安装的QLogic HBA的序列值。从文件中最后一个主机适配器编号开始。(第一个主机适配器编号从0开始)

例如:

```
alias parport_lowlevel parport_pc
```

```
alias scsi_hostadapter sym53c8xx
```

```
alias scsi_hostadapter1 qla2300
```

```
alias scsi_hostadapter2 qla2300
```

```
alias eth0 tg3
```

```
alias eth1 tg3
```

```
options scsi_mod max_scsi_luns=255 scsi_allow_ghost_devices=1
```

创建一个新的ramdisk以包含上述改动:

```
mkinitrd v initrd-1.img1
```

\$1表示当前运行的v2.4.X内核版本.

例如: `mkinitrd v initrd-2.4.21-9.0.1.ELsmp.img 2.4.21-9.0.1.ELsmp`

重启系统。

Linux系统SCSI磁盘管理全攻略（一）

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-07-14

本系列文章系统、全面地介绍了Linux 的SCSI设备管理机制和整套操作命令。包括以下六大主题：

- Linux SCSI子系统概述
- Linux主机LUN识别
- 动态SAN网络重配
- LUN发现步骤
- Linux设备命名
- SCSI磁盘限制

本文作为系列第一篇，解答了用户的常见问题，如：怎样确认驱动模块是否被加载？怎样查看SCSI子系统已发现并识别的设备？如何验证内核已正确识别出LUN的编号，用户在从fabric中添加或删除磁盘时，有哪些方法让主机重新识别？

Linux SCSI子系统概述:

Linux SCSI子系统包括三层：

上层由特定的设备类型驱动所组成，如磁盘驱动、磁带驱动和CD-ROM驱动，最靠近用户空间。

下层包括诸如QLogic和Emulex HBA这类驱动，最靠近硬件。

中间层是SCSI核心代码，连接上层和下层。

按照内核版本的区别，驱动可编译进内核或以模块的形式加载到内核。sd是SCSI磁盘驱动，或块驱动，作为模块时命名为scsi_mod。

通常，在大多数版本中这些驱动都编译为模块，并在启动时作为初始化内存磁盘镜像文件（initrd image）的一部分被加载。如果当前没有在启动时加载，而启动过程中要求加载时，那么就需要重新编译一个包含该驱动的初始化内存磁盘镜像文件。在2.4版本内核中，需要修改/etc/modules.conf文件来完成；2.6版本内核中，不仅需要修改/etc/modules.conf文件，还要执行mkinitrd命令来对改动进行更新。

要确认驱动是否编译为模块及当前是否被加载，只需查看lsmod命令输出中的sd_mod 和 scsi_mod。以下为示例输出：

```
# lsmod | grep sd
```

```
sd_mod      13440 296
```

```
scsi_mod    104068 6 [qla2300 st sr_mod sg ncr53c8xx sd_mod]
```

注意：如果SCSI中间层驱动编译为模块，则scsi_mod必须先于上层和下层驱动加载。

Linux主机LUN识别：

Linux 2.4内核版本中，系统在加载主机适配层驱动时通过扫描SCSI总线时识别。SCSI子系统已发现并识别的设备在/proc/scsi/scsi文件中列出。比较好的方法是使用cat命令列出/proc/scsi/scsi的输出，来验证内核已正确识别出LUN ID。命令输出示例如下：

```
# cat /proc/scsi/scsi
```

```
# cat /proc/scsi/scsi | less
```

Attached devices:

Host: scsi1 Channel: 00 Id: 00 Lun: 00

Vendor: IBM-PSG Model: DDYS-T36950M M Rev: S96E

Type: Direct-Access ANSI SCSI revision: 03

Host: scsi1 Channel: 00 Id: 08 Lun: 00

Vendor: IBM Model: YGLv3 S2 Rev: 0

Type: Processor ANSI SCSI revision: 02

Host: scsi4 Channel: 00 Id: 00 Lun: 00

Vendor: IBM Model: 1750500 Rev: 5.17

Type: Direct-Access ANSI SCSI revision: 05

Host: scsi4 Channel: 00 Id: 00 Lun: 01

Vendor: IBM Model: 1750500 Rev: 5.17

Type: Direct-Access ANSI SCSI revision: 05

Host: scsi4 Channel: 00 Id: 00 Lun: 02

Vendor: IBM Model: 1750500 Rev: 5.17

Type: Direct-Access ANSI SCSI revision: 05

Host: scsi4 Channel: 00 Id: 00 Lun: 03

Vendor: IBM Model: 1750500 Rev: 5.17

Type: Direct-Access ANSI SCSI revision: 05

Host: scsi4 Channel: 00 Id: 00 Lun: 04

Vendor: IBM Model: 1750500 Rev: 5.17

Type: Direct-Access ANSI SCSI revision: 05

Host: scsi4 Channel: 00 Id: 00 Lun: 05

Vendor: IBM Model: 1750500 Rev: 5.17

Type: Direct-Access ANSI SCSI revision: 05

Host: scsi4 Channel: 00 Id: 00 Lun: 06

Vendor: IBM Model: 1750500 Rev: 5.17

Type: Direct-Access ANSI SCSI revision: 05

Host: scsi4 Channel: 00 Id: 00 Lun: 07

Vendor: IBM Model: 1750500 Rev: 5.17

Type: Direct-Access ANSI SCSI revision: 05

注意，`/proc/scsi/scsi` 列出的磁盘设备不是动态的，因此不受网络状态变更的影响。

从Linux2.6内核版本开始，`/proc`文件系统迁移至改进后的`/sys`文件系统。`/sys`文件系统中加入对动态变更的支持，如添加和删除LUN，而无需重新加载主机适配器驱动或重启主机。通常，可通过`/sys/class/scsi_host/hostN`目录下的内容来查看哪些SCSI设备已被主机识别。N是主机适配器ID号。用户空间有一个`lsscsi`工具，使用`/sys`中的信息来显示所有识别设备的汇总清单。（`lsscsi`命令仅在SLES中支持）

动态SAN网络重配：

用户在网络中添加或删除磁盘时，需要对其重新识别。可使用以下四种方法之一让Linux主机重新识别这些变更：

1. 重启主机
2. 卸载并重新加载主机适配器驱动
3. 通过`echo /sys`文件系统（仅适用于2.6内核版本）重新扫描总线
4. 通过`echo /proc`或`/sys`文件系统手动添加和删除SCSI磁盘

这一部分的更多命令示例可参见：[Linux系统SCSI磁盘扫描机制解析及命令实例](#)。

重启主机或卸载并加载主机适配器驱动

由于设备是通过扫描SCSI总线被发现的，通常重新扫描SCSI总线来发现SAN网络变更是最简便的方法。由重新加载主机适配器驱动或重启系统来触发总线扫描。

卸载主机适配器驱动或重启系统之前，用户必须：

1. 停止I/O
2. 卸载所有文件系统
3. 如果使用了SDD，在重新加载主机适配器驱动前，用`sdd stop`命令卸载SDD驱动。在主机适配器驱动重新加载之后，用`sdd start`命令重新加载SDD。

当主机适配器驱动作为模块植入时，可通过重新加载主机适配器驱动来解决。而无论主机适配器驱动编入内核还是作为模块，都可以重启系统来解决该问题。

通过echo /sys文件系统重新扫描总线（仅适用于Linux2.6内核版本）

对于Linux2.6内核版本，可通过/sys接口重新扫描总线，而无需重新加载主机适配器驱动或重启系统。以下命令可扫描所有通道、target、LUN和主机。

```
echo "- - -" > /sys/class/scsi_host/hostH/scan
```

手动添加和删除SCSI磁盘

用户可通过以下命令手动添加和删除SCSI磁盘。

注意：以下命令示例中，H, B, T, L代表设备的主机，总线，target，和LUN ID。

可通过以下命令删除或对SCSI磁盘取消配置：

```
echo "scsi remove-single-device H B T L" > /proc/scsi/scsi
```

如果驱动无法被卸载并重新加载，并且用户知道新设备的主机，总线，target和LUN ID，那么可以使用以下命令通过/proc/scsi/scsi 文件添加来实现：

```
echo "scsi add-single-device H B T L" > /proc/scsi/scsi
```

对于2.6内核版本，设备同样可通过/sys文件系统来添加和删除。使用以下命令删除磁盘并使系统重认此更新：

```
echo "1" > /sys/class/scsi_host/hostH/device/H:B:T:L/delete
```

或使用如下命令：

```
echo "1" > /sys/class/scsi_host/hostH/device/targetH:B:T/H:B:T:L/delete
```

使用以下命令在内核中重新注册磁盘：

```
echo "B T L" > /sys/class/scsi_host/hostH/scan
```

注意：Linux内核不在/dev目录中为网络设备指定固定名称。设备文件名在扫描总线时按照发现顺序指定，例如，一个LUN名为/dev/sda，在驱动重新加载后，这个LUN名称可能更改为/dev/sdce。网络重认可能导致主机，总线，target和LUN ID值的偏移，因此通过/proc/scsi/scsi文件添加特定磁盘是不可靠的。

Linux系统SCSI磁盘管理全攻略（二）

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-07-15

本系列文章系统、全面地介绍了Linux 的SCSI设备管理机制和整套操作命令。

包括以下六大主题：

1. Linux SCSI子系统概述
2. Linux主机LUN识别
3. 动态SAN网络重配
4. LUN发现步骤
5. Linux设备命名
6. SCSI磁盘限制

附：SCSI磁盘问题识别与解决方法

系列文章第一篇看这里：

[Linux系统SCSI磁盘管理全攻略（一）](#)

作为系列之二，本文主要介绍LUN发现步骤和Linux设备命名两大主题，解答用户常见问题乳：Linux驱动除了LUN 0以外，没有自动配置其他LUN怎么办？Linux磁盘的命名规则是什么？

LUN发现步骤：

如果用户的Linux驱动除了LUN 0以外，没有自动配置其他LUN，用户可以根据SCSI中间层驱动使用的参数和设置来手动配置。以下示例显示了Linux主机/proc/scsi/scsi文件中，各适配器端口仅配置了第一个LUN，即LUN 0的情况。

```
# cat /proc/scsi/scsi
...
Host: scsi0 Channel: 00 Id: 00 Lun: 00
Vendor: EMC Model: SYMMETRIX Rev: 5773
Type: Direct-Access ANSI SCSI revision: 04
Host: scsi0 Channel: 00 Id: 15 Lun: 00
Vendor: EMC Model: SYMMETRIX Rev: 5773
Type: Direct-Access ANSI SCSI revision: 04
Host: scsi2 Channel: 00 Id: 00 Lun: 00
Vendor: EMC Model: SYMMETRIX Rev: 5773
Type: Direct-Access ANSI SCSI revision: 04
```

Host: scsi3 Channel: 00 Id: 00 Lun: 00

Vendor: EMC Model: SYMMETRIX Rev: 5773

Type: Direct-Access ANSI SCSI revision: 04

Host: scsi3 Channel: 00 Id: 01 Lun: 00

Vendor: EMC Model: SYMMETRIX Rev: 5773

Type: Direct-Access ANSI SCSI revision: 04

解决这一问题有两种方式：

1. 创建脚本在/proc/scsi/scsi文件中手动添加磁盘设备
2. 通过修改initrd设置系统启动时自动发现LUN

创建脚本echo /proc文件系统

使用scsi add-single-device命令连续配置分配给主机的所有LUN。写一个脚本为各主机适配器重复scsi add-single-device命令，从而给每一个LUN配置ID。该脚本必须扫描所有主机适配器端口并识别分配给各端口的所有LUN。

脚本运行结束之后，用户可以在/proc/scsi/scsi文件中查看分配的所有LUN。

以下示例列出脚本配置所有LUN之后的/proc/scsi/scsi文件。

```
# cat /proc/scsi/scsi
```

...

Host: scsi3 Channel: 00 Id: 01 Lun: 00

Vendor: EMC Model: SYMMETRIX Rev: 5773

Type: Direct-Access ANSI SCSI revision: 04

Host: scsi3 Channel: 00 Id: 00 Lun: 01

Vendor: EMC Model: SYMMETRIX Rev: 5773

Type: Direct-Access ANSI SCSI revision: 04

Host: scsi3 Channel: 00 Id: 00 Lun: 02

Vendor: EMC Model: SYMMETRIX Rev: 5773

Type: Direct-Access ANSI SCSI revision: 04

Host: scsi3 Channel: 00 Id: 00 Lun: 03

Vendor: EMC Model: SYMMETRIX Rev: 5773

Type: Direct-Access ANSI SCSI revision: 04

Host: scsi3 Channel: 00 Id: 00 Lun: 04

Vendor: EMC Model: SYMMETRIX Rev: 5773

Type: Direct-Access ANSI SCSI revision: 04

...

在系统启动时自动发现LUN

第二种方法是通过设置SCSI中间层驱动的参数来完成，此参数控制在SCSI总线扫描过程中扫描LUN的数量。以下步骤同时适用于2.4和2.6内核版本，但假定SCSI中间层驱动编译为scsi_mod模块，并在系统启动时自动加载。对于Linux 2.4内核版本，为了发现所有卷，通常在系统中设置为磁盘设备的最大数量，用户需要设置SCSI中间层驱动的参数。例如，如果max_scsi_luns设置为1则SCSI总线只扫描到LUN 0。此参数应当设置为内核能够支持的最大磁盘数量，例如，128或256。在Linux 2.6内核中，可使用同样的步骤，除了参数名从max_scsi_lun更改为max_lun。

- 编辑/etc/modules.conf文件
- 添加以下内容：

```
options scsi_mod max_scsi_luns= (n表示探测的LUN总数)
```

- 保存文件
- 运行mkinitrd命令重新编译当前内核相应的ram-disk。以下命令示例中，为当前运行内核版本，可参见“uname -r”命令的输出，例如：2.4.21-292-smp。

SUSE中命令如下：

```
cd /boot
```

```
mkinitrd -k vmlinuz- -i initrd-
```

Red Hat中命令如下：

```
cd /boot
```

```
mkinitrd -v initrd-.img
```

- 重启主机
- 验证 /boot/grub/menu.lst文件中，已正确配置了新创建的initrd镜像。

Linux设备命名：

内核驱动可使用特定的设备文件来控制磁盘设备。映射到同一物理磁盘设备的设备文件可能不止一个。例如，在多路径环境下某一设备配置四条路径，则会有四个不同的设备文件映射到同一设备。

设备文件位于/dev目录下，通过major和minor编号访问。光纤通道连接的设备通过sd驱动作为SCSI磁盘设备管理。因此，每一个LUN在/dev目录下有一个对应的设备文件。

SCSI磁盘设备有一个以“sd”为前缀的特定设备文件，具有如下命名格式：

/dev/sd[a-z][1-15]

名称中不带数字表示整个磁盘，而名称中有数字则表示磁盘的一个分区。依据惯例，一个SCSI磁盘设备最多可以有16个minor编号。因此，对于一整块磁盘，每块磁盘最多有15个分区，使用一个minor编号来标示整块磁盘（例如/dev/sda），其他15个minor编号用来标示该磁盘的分区（例如/dev/sda1，/dev/sda2，等等）。以下示例显示整块磁盘/dev/sda的设备文件，该设备major编号为8，minor编号为0，有15个分区。

```
# ls -l /dev/sda*

brw-rw---- 1 root disk 8, 0 May 24 08:09 /dev/sda
brw-rw---- 1 root disk 8, 1 May 24 08:09 /dev/sda1
brw-rw---- 1 root disk 8, 10 May 24 08:09 /dev/sda10
brw-rw---- 1 root disk 8, 11 May 24 08:09 /dev/sda11
brw-rw---- 1 root disk 8, 12 May 24 08:09 /dev/sda12
brw-rw---- 1 root disk 8, 13 May 24 08:09 /dev/sda13
brw-rw---- 1 root disk 8, 14 May 24 08:09 /dev/sda14
brw-rw---- 1 root disk 8, 15 May 24 08:09 /dev/sda15
brw-rw---- 1 root disk 8, 2 May 24 08:09 /dev/sda2
brw-rw---- 1 root disk 8, 3 May 24 08:09 /dev/sda3
brw-rw---- 1 root disk 8, 4 May 24 08:09 /dev/sda4
brw-rw---- 1 root disk 8, 5 May 24 08:09 /dev/sda5
brw-rw---- 1 root disk 8, 6 May 24 08:09 /dev/sda6
brw-rw---- 1 root disk 8, 7 May 24 08:09 /dev/sda7
brw-rw---- 1 root disk 8, 8 May 24 08:09 /dev/sda8
brw-rw---- 1 root disk 8, 9 May 24 08:09 /dev/sda9
```

对于Red Hat版本，内核实际上为128个设备创建了设备文件。对于SUSE，只有前16块磁盘有设备文件。用户必须使用mknod命令为其他磁盘创建设备文件。在2.6内核版本中，只有在内核发现并识别出设备时才能创建设备文件。/proc/partitions文件列出所有SCSI磁盘驱动识别出的“sd”设备，包括sd名，major编号，minor编号，以及各磁盘设备的大小。

以下示例列出/proc/partitions文件的内容：

```
# cat /proc/partitions

major minor #blocks name
```

```
8 0 17774160 sda
```

8 1 1052226 sda1
8 2 208845 sda2
8 3 10490445 sda3
8 16 976576 sdb
8 32 976576 sdc
8 48 976576 sdd
8 64 976576 sde
8 80 976576 sdf
8 96 976576 sdg
8 112 976576 sdh
8 128 976576 sdi
8 144 976576 sdj
8 160 976576 sdk
8 176 976576 sdl
8 192 976576 sdm
8 208 976576 sdn
8 224 976576 sdo
8 240 976576 sdp
65 0 976576 sdq
65 16 1048576 sdr
65 32 1048576 sds
65 48 1048576 sdt
65 64 1048576 sdu
65 80 1048576 sdv
65 96 1048576 sdw
65 112 1048576 sdx
65 128 1048576 sdz
65 144 1048576 sdaa
65 160 1048576 sdab
65 176 1048576 sdac
65 192 1048576 sdad
65 208 1048576 sdae
65 224 1048576 sdaf

66 0 1048576 sdag
66 16 1048576 sdah
66 32 1048576 sdai
66 48 1048576 sdaj
66 64 1048576 sdak
66 80 1048576 sdal
66 96 1048576 sdam
66 112 1048576 sdan
66 128 1048576 sdao
66 144 1048576 sdap
66 160 1048576 sdaq
66 176 1048576 sdar
66 192 1048576 sdas
66 208 1048576 sdat
66 224 1048576 sdau
66 240 1048576 sdav

Linux系统SCSI磁盘管理全攻略（三）

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-07-18

本系列文章系统、全面地介绍了Linux 的SCSI设备管理机制和整套操作命令。包括以下六大主题：

1. Linux SCSI子系统概述
2. Linux主机LUN识别
3. 动态SAN网络重配
4. LUN发现步骤
5. Linux设备命名
6. SCSI磁盘限制

附：SCSI磁盘问题识别与解决方法

系列文章第一篇看[这里](#)：

作为系列之三，本文主要回答Linux磁盘设备数量与major和minor的关系，以及SCSI磁盘问题识别与解决方法。

Linux磁盘限制:定义磁盘设备数量

Linux内核使用静态的major和minor编号来进行SCSI寻址。major可看作设备驱动程序，同一设备驱动程序管理的设备由相同的major编号。而minor编号代表被访问的具体设备。内核为SCSI磁盘设备保留了一定数量的major编号。因此，SCSI磁盘设备的数量受到可用的major编号的限制。这些可用的major编号数量又随内核版本不同而有所区别。

对于Linux 2.4内核版本，Linux主机系统上最大可配置设备数是128、256或2304。保留8位bit位用作minor编号，因此对于每一个major有256个可用的minor编号。按照惯例，每块SCSI磁盘要保留16个minor，因此每块磁盘最多可有15个分区（另1个minor代表整块磁盘）。这样，通过16个minor编号来定义一块磁盘和它的分区，而最多可以有 $2^8=256$ 个minor，则每个major编号可支持 $256 \div 16=16$ 块SCSI磁盘。

比较旧的内核版本为SCSI磁盘设备保留8个major编号。分别是8,65,66,67,68,79,70和71。REHL2.1和REHL3中，就是使用以上保留的major。有了这8个保留的major，和每个major有的16个minor编号，内核一共可支持 $8 \times 16=128$ 个磁盘设备。早期的SuSE SLES8内核由于保留16个major，共可以支持256个磁盘设备。额外的major编号是128, 129, 130, 131, 132, 133, 134, 135。

新的内核版本使用动态配置的方法，在头16个保留major编号用完之后，会使用任何可用的major编号。因此这些内核可支持2304个磁盘设备。

通常，可使用以下公式来计算Linux主机系统可支持的最大设备数：

最大磁盘数=major编号数×minor编号数÷分区数

例如：磁盘数= $8 \times 256 \div 16=128$ 或 磁盘数= $144 \times 256 \div 16=2304$

对于Linux2.6内核版本major数增长至12比特位而minor增长至20比特位，因此Linux 2.6内核支持上千块磁盘，每块磁盘最多15个分区的限制没有改变。由于major并不仅仅用于磁盘（/proc/devices），所以不能以 2^{12} 个major来计算磁盘个数。一个major可用的minor数就达到 $2^{20}=65536$ ，而每个SCSI磁盘16个minor的限制仍然不变，每个major可支持 $2^{20} \div 16=4096$ 个。

其他限制磁盘数量的因素

如果主机适配器驱动作为模块加载在Linux系统中，则内核对可配置磁盘总数有一定的限制。有可能小于内核支持的最大值（通常128或256）。系统加载的第一个模块发现磁盘数可配置为内核支持的最大磁盘数，之后的驱动则达不到这个值。所有这些驱动共享一个pool中的设备结构，这些设备结构是在第一个主机适配器驱动加载后静态分配的。

SCSI磁盘问题识别与解决方法：

- SAN发生故障之后，内核可能永久禁用LUN并对磁盘设备记录一条消息：“device set offline”。如果这发生在2.4内核版本，则无法将LUN重新设为online，除非卸载底层设备驱动并重新加载，或重启系统。
- 在2.6内核版本上，设备可通过以下方法之一重新设置为online：
Redhat: `echo "running" >/sys/class/scsi_host/hostH/device/targetH:C:T/H:C:T:L/state`
SLES: `echo "1" > /sys/class/scsi_host/hostH/device/H:C:T:L /online`
- 系统ps命令可能周期性显示进程处于D-state，表示不可中断的状态，这是由于进程一直在内核中等待。在错误的情形下，进程可能永久停留在这个状态，需要系统重启才能回复。

Linux主机HBA常用操作指南

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-07-20

本文阐述了Linux系统下HBA卡的常用基本操作，包括如何通过命令或日志查找HBA，如何查找WWN以及设置永久绑定，以及HBA卡安装完成之后如何识别存储设备。

主机外接存储的准备工作：

HBA卡与操作系统的安装顺序决定了主机外接存储的操作步骤。如果HBA安装在操作系统之前，那么连接磁盘还是比较简单的。安装程序检测到发现的硬件并准备所需模块。如果适配器安装于操作系统之后，或在操作系统安装之后有变更，则用户需要手动安装。本文以Emulex 1000作为示例HBA。

HBA安装于操作系统之前：安装程序发现硬件，准备模块

HBA安装或变更于操作系统之后：用户手动安装

查看HBA:

lspci (Linux command)

```
[root@sandboxlinux etc]# lspci
```

```
00:00.0 Host bridge: VIA Technologies, Inc. VT82C693A/694x [Apollo PRO133x] (rev c4)
00:01.0 PCI bridge: VIA Technologies, Inc. VT82C598/694x [Apollo MVP3/Pro133x AGP]
00:04.0 ISA bridge: VIA Technologies, Inc. VT82C686 [Apollo Super South] (rev 40)
00:04.1 IDE interface: VIA Technologies, Inc.
VT82C586A/B/VT82C686/A/B/VT8233/A/C/VT8235 PIPC Bus Master IDE (rev 06)
00:04.2 USB Controller: VIA Technologies, Inc. USB (rev 16)
00:04.3 USB Controller: VIA Technologies, Inc. USB (rev 16)
00:04.4 Host bridge: VIA Technologies, Inc. VT82C686 [Apollo Super ACPI] (rev 40)
00:09.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL-8139/8139C/8139C+(rev 10)
00:0a.0 Ethernet controller: 3Com Corporation 3c905 100BaseTX [Boomerang]
00:0c.0 Fibre Channel: Emulex Corporation: Unknown device fa00 (rev 01)
00:0c.1 Fibre Channel: Emulex Corporation: Unknown device fa00 (rev 01)
01:00.0 VGA compatible controller: ATI Technologies Inc Rage XL AGP 2X (rev 27)
```

如果已经安装了HBA，使用上述命令确认HBA芯片集的类型。此命令列出了适配器的芯片集。本例显示，Fibre Channel Emulex card, device fa00 , rev 01。

在消息日志中查找HBA:

另一个在服务器中查找HBA类型的方法是查看消息日志文件最后一次启动过程中保存的信息。本例中，在/var/log/目录下执行“more messages |grep HBA”我们看见两个Emulex LP10000 HBA。

```
[#/var/log/more messages |grep HBA
```

```
Oct 18 14:55:21 sandboxlinux kernel: scsi0 : HBA: Emulex LightPulse
```

```
LP10000 on PCI bus 00 device 60 irq 11
```

```
Oct 18 14:55:21 sandboxlinux kernel: scsi1 : HBA: Emulex LightPulse
```

```
LP10000 on PCI bus 00 device 61 irq 9
```

Emulex HBA驱动:

Emulex驱动位于/lib/modules/2.4.21-4.EL/kernel/drivers/scsi 。SCSI驱动文件是lpfcdd.o。


```
/lib/modules/2.4.21-4.EL/kernel/drivers/scsi
```

```
3w-xxxx.o ata_piix.o ide-scsi.o lpfcdd.orig scsi_mod.o st.o
```

```
aacraid BusLogic.o ips.o megaraid2.o sd_mod.o sym53c8xx_2
```

```
aic7xxx dpt_i2o.o libata.o megaraid.o sg.o sym53c8xx.o
```

```
aic7xxx_old.o gdth.o lpfcdd.o
```

lsmod命令验证模块已被加载。本例显示驱动已被加载，名为lpfcdd。

```
/sbin/lsmod
```

```
Module Size Used by Tainted: P
```

```
ide-cd 34016 0 (autoclean)
```

```
cdrom 32576 0 (autoclean) [ide-cd]
```

```
mousedev 5624 1 (autoclean)
```

```
input 6144 0 (autoclean) [mousedev]
```

```
iptable_filter 2412 1 (autoclean)
```

```
ext3 89960 2
```

```
jbd 55060 2 [ext3]
```

```
lpfcdd 295016 18
```

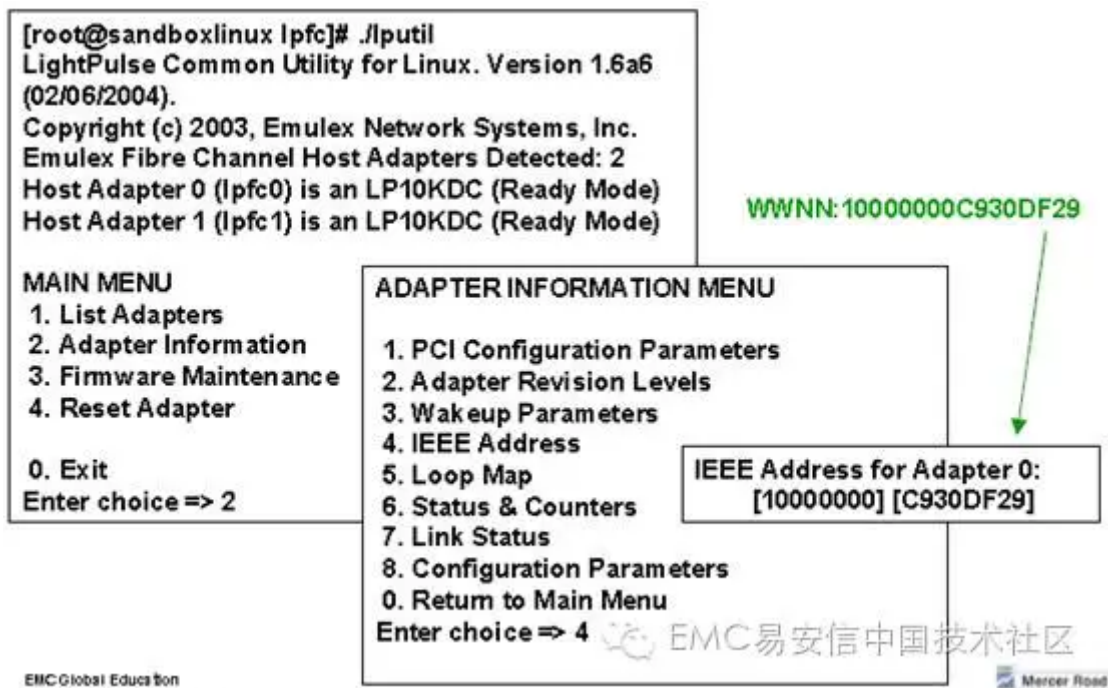
```
aic7xxx 162064 4
```

```
sd_mod 13360 25 [emcp]
```

```
scsi_mod 112680 5 [emcnpmpap emcnpmp]
```

如何使用Emulex lputil工具查找WWN：

通过执行lputil，可通过Emulex “lputil”工具查找HBA卡的WWN名。选择2“Adapter Information”，之后选择4“IEEE Address”。此地址将会被显示成为2个8位16进制字符，即为WWN。



Emulex永久绑定:

在光纤网络环境下，EMC推荐永久绑定驱动，但不强制要求，目的是限制网络存储只能通过预定义的端口进行通讯。如不绑定，则无法保证服务器端与EMC存储阵列通过光纤网络通讯逻辑路由的一致性。如果交换机的物理配置发生变更（如调换线缆或主机重启），逻辑路由将无法保持一致，这将会导致数据损坏。注意Emulex驱动力的永久绑定是基于target的，而非LUN。可通过两种方式来实现永久绑定，使用Emulex配置文件/etc/lpfc.conf，或在/etc/modules.conf文件中添加选项。

lpfc.conf文件包含所有控制驱动初始化的参数。此文件一经修改，需要经过重新编译和加载才能使改动生效。也就是说，如果lpfc.conf文件中添加了永久绑定则必须重新加载。

配置lpfc.conf文件的步骤:

在lpfc.conf文件中配置永久绑定，首先在lpfc.conf文件中查找lpfc_bind_entries参数并将此参数设置成该主机实际的绑定数目。例如：int lpfc_bind_entries =4; 定义了4个绑定。

第二，选择一种永久绑定的方式。Emulex有三种永久绑定的方法：

lpfc_fcp_bind_WWNN将指定的World Wide Node Name绑定至target ID

lpfc_fcp_bind_WWPN将指定的World Wide Port Name绑定至target ID

lpfc_fcp_bind_DID将指定的port ID绑定至target ID

EMC推荐通过WWPN永久绑定。只能使用一种永久绑定方式。当指定target ID时，Target 0不能被使用，因为target 0是为控制器保留的。

第三步，查找lpfc_automap参数并将其设置为0。

第四步，查找lpfc_scandown参数并将其设置为0。

第五步，保存lpfc.conf文件。

第六步，使用make build命令编译驱动。

第七步，使用make install命令在当前运行的内核安装驱动。make install参数将驱动复制到/lib/modules/*myversion*/kernel/drivers/scsi（*myversion*是内核版本）。

我们按照target device WWPN建立FCP绑定。lpfc_fcp_bind_WWPN将指定WWPN绑定至target ID。该绑定确保分配的target在重启之后仍被保存。绑定内容格式为：“NNNNNNNNNNNNNNNNNN:lpfcXtY”，NNNNNNNNNNNNNNNNNN是16bit数代表着target WWPN。X是物理硬件适配器编号，Y是分配的target。多条语句之间以逗号相间隔，以分号为结束。目前无法将适配器target指定给特定的sd设备，/dev/sdX。

例如：char *lpfc_fcp_bind_WWPN={"21000020370cf8263:lpfc1t0"};

举例1：

```
char *lpfc_fcp_bind_WWPN={.NNNNNNNNNNNNNNNNNN:lpfcXtY.};
```

```
vi /etc/lpfc.conf
```

```
int lpfc_bind_entries =4;
```

```
int lpfc_automap parameter=0;
```

```
int lpfc_scandown=0;
```

```
char *lpfc_fcp_bind_WWPN={"50060160006000ed:lpfc0t1",
```

```
"50060168006000ed:lpfc0t2","50060160006000ed:lpfc1t1",
```

```
"50060168006000ed:lpfc1t2"};
```

本例中我们修改了lpfc.conf文件，将参数lpfc_bind_entries，lpfc_automap和lpfc_scandown设置为0，并通过lpfc_fcp_bind_WWPN绑定WWPN。

配置module.conf文件的步骤：

在module.conf文件中配置永久绑定，首先为系统中每一个adapter指定WWNN/WWPN以及target信息，之后通过在文件中添加永久绑定参数修改module.conf文件。

首先，将lpfc_bind_entries参数设置成主机实际绑定的数量：比如：lpfc_bind_entries=4。

第二步，将lpfc_scandown参数设置成“0”。

第三步，将lpfc_automap参数设置成“0”。

第四步，选择lpfc_fcp_bind_WWPN, lpfc_fcp_bind_WWNN 或lpfc_fcp_bind_DID，target ID设置绑定类型。

EMC推荐通过WWPN实现永久绑定。

第五步，保存对/etc/modules.conf文件的更改。

第六步，通过执行/sbin/#./modprobe lpfcdd更新lpfcdd模块。

举例2：

```

#/etc/more modules.conf

alias parport_lowlevel parport_pc

alias scsi_hostadapter sym53c8xx

alias scsi_hostadapter1 lpfcdd

alias scsi_hostadapter2 lpfcdd

alias eth0 tlant

options scsi_mod max_scsi_luns=256 scsi_allow_ghost_devices=1

options lpfcdd lpfc0_topology=0x02 lpfc1_topology=0x02

lpfc_network_on=0 lpfc_use_data_direction=0 lpfc_automap=0

lpfc_scandown=0 lpfc_bind_entries=4

lpfc_fcp_bind_WWPN={.50060160006000ed:lpfc0t1,50060168006000ed:lpfc0t2,50
060160006000ed:lpfc1t1,50060168006000ed:lpfc1t2.};

```

本例中我们修改了/etc/modules.conf文件，设置lpfc_automap=0, lpfc_scandown=0, lpfc_bind_entries=4，并通过lpfc_fcp_bind_WWPN={"50060160006000ed:lpfc0t1,50060168006000ed:lpfc0t2,50060160006000ed:lpfc1t1,50060168006000ed:lpfc1t2"}绑定WWPN。

识别存储设备：

HBA安装结束并重启系统之后，用户可以通过执行“more messages |grep SCSI”识别HBA发现的磁盘设备。

本例中我们看到的设备从sda到sdf。

举例3：

```

[root@sandboxlinux log]# more messages |grep SCSI
Oct 18 14:55:21 sandboxlinux kernel: SCSI subsystem driver Revision: 1.00
Oct 18 14:55:21 sandboxlinux kernel: Emulex LightPulse FC SCSI/IP: 2.02e Osgt: 1.08
Oct 18 14:55:21 sandboxlinux kernel: Type: Direct-Access ANSI SCSI revision: 02
Oct 18 14:55:21 sandboxlinux kernel: Type: Direct-Access ANSI SCSI revision: 02
Oct 18 14:55:22 sandboxlinux kernel: Type: Direct-Access ANSI SCSI revision: 02
Oct 18 14:55:22 sandboxlinux kernel: Type: Direct-Access ANSI SCSI revision: 02
Oct 18 14:55:22 sandboxlinux kernel: Type: Direct-Access ANSI SCSI revision: 02
Oct 18 14:55:22 sandboxlinux kernel: Type: Direct-Access ANSI SCSI revision: 02
Oct 18 14:55:22 sandboxlinux kernel: SCSI device sda: 499200000 512-byte hdwr sectors (255590 MB)
Oct 18 14:55:22 sandboxlinux kernel: SCSI device sdb: 499200000 512-byte hdwr sectors (255590 MB)
Oct 18 14:55:22 sandboxlinux kernel: SCSI device sdc: 5760 512-byte hdwr sectors (3 MB)
Oct 18 14:55:22 sandboxlinux kernel: SCSI device sdd: 5760 512-byte hdwr sectors (3 MB)
Oct 18 14:55:22 sandboxlinux kernel: SCSI device sde: 5760 512-byte hdwr sectors (3 MB)
Oct 18 14:55:22 sandboxlinux kernel: SCSI device sdf: 5760 512-byte hdwr sectors (3 MB)

```

Linux/AIX系统实用监控命令详解

原创 EMC中文技术社区 [戴尔易安信技术支持](#) 2016-07-21

Linux/UNIX系统提供了一些有用的监控命令如：iostat, vmstat, ps, sar, 通过它们系统管理员可以方便地监测系统资源是否平衡并解决性能问题。本文阐述了这些命令的使用方法，并以AIX系统为例附加应用实例。为Linux/AIX系统管理员提供参考。

iostat

iostat命令主要通过物理磁盘的活跃时间及它们的平均传输速度，监控系统输入/输出设备负载。根据iostat命令产生的报告，用户可确定一个系统配置是否平衡，并据此在物理磁盘与适配器之间更好的平衡输入/输出负载，从而在用户注意到服务器运行缓慢之前提早发现输入/输出缓慢的问题。

iostat工具的主要目的是通过监控磁盘的利用率（tm_act字段），检测系统的I/O瓶颈。此外，还可用于确定CPU问题，辅助容量规划。vmstat和iostat联合使用，可获得与CPU，内存和I/O子系统有关的性能问题的必需数据。

下图是AIX系统iostat命令输出：

```
blondie# iostat

System configuration: lcpu=4 drives=55 paths=2 vdisks=0

tty:          tin          tout      avg-cpu: % user % sys % idle % iowait
            0.0            0.5              0.2   0.3   99.4   0.0

Disks:          % tm_act      Kbps      tps      Kb_read  Kb_wrtn
hdisk0           0.5         7.0         1.1      209961   3554724
hdisk1           0.0         0.0         0.0         0         0
hdisk74          0.0         0.0         0.0         0         0
hdisk70          0.0         0.0         0.0         0         0
hdisk78          0.0         0.0         0.0         0         0
hdisk81          0.1         0.0         0.0         0         0
hdisk75          0.0         0.0         0.0         0         0
hdisk79          0.0         0.0         0.0         0         0
hdisk87          0.0         0.0         0.0         0         0
hdisk88          0.0         0.0         0.0         0         0
hdisk90          0.0         0.0         0.0         0         0
hdisk76          0.0         0.0         0.0         0         0
hdisk80          0.0         0.0         0.0         0         0
hdisk84          0.0         0.0         0.0         0         0
```

iostat命令可产生如下四种类型的报告：

- tty和CPU利用情况
- 磁盘利用情况
- 系统吞吐率
- 适配器吞吐率

% tm_act: 物理磁盘活动的时间百分比

Kbps: 某块磁盘传输数据的总量(读或写)

tps: 某块物理磁盘每秒钟 IO 传输的数量

Kb_read: 从磁盘上读取数据的总量

Kb_wrtn: 写入磁盘的数据总量

vmstat

★★

★★

vmstat命令报告关于核心线程，虚拟内存，自陷（trap），磁盘以及CPU行为的统计。而且每种行为报告都被更细致地用百分比分别表示用户态、核态、空闲以及等待磁盘I/O等情况。内核维持了对核心线程，换页以及中断行为的统计数据，而vmstat命令则通过使用knlist子程序和/dev/kmem伪设备驱动器访问这些数据。磁盘的输入/输出统计是通过设备驱动器维持的。对于磁盘，平均传输速度是通过使用活跃时间核传输信息数目决定的。而活跃时间百分比则是从报告期间驱动器忙的时间量计算出来的。

下图是AIX系统vmstat命令输出:

```
blondie# vmstat
System Configuration: lcpu=4 mem=7775MB

kthr      memory                                  page          faults        cpu
-----
r  b   avm   fre  re  pi  po  fr   sr   in   sy   cs  us  sy  id  wa
1  1 336941 1578665  0  0  0  0   0   0   9  712 198  0  0 99  0
```

vmstat命令产生五种类型的报告：

- 虚存行为报告
- fork子进程情况报告
- 每个设备产生的中断情况报告
- 汇总报告
- 输入/输出行为报告

page: 页面调入调出的数量

wa: 等待I/O的时间

avm: 活动虚拟页面,在进程运行中分配到工作段的页面空间数，单位为4K

fre: 空闲列表的数量.一般不少于120,当fre少于120时,系统开始自动的kill进程去释放free list

PS

★★

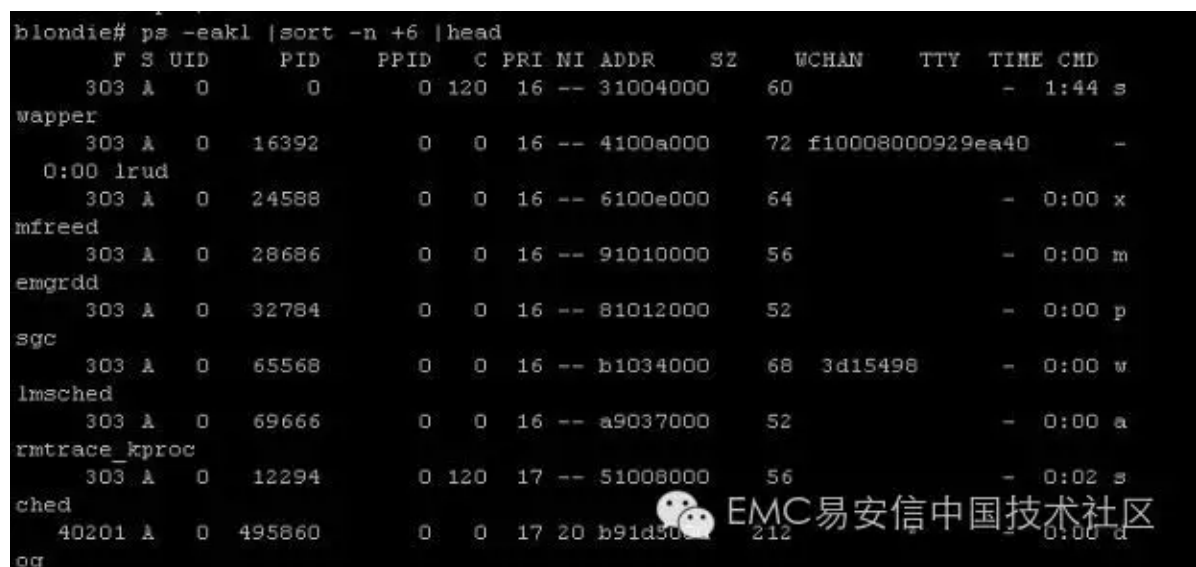
★★

ps命令是UNIX系统中最常见的命令，它主要显示系统中关于进程的统计和状态信息，如进程ID，I/O行为以及CPU利用率等。利用ps命令提供的信息，可决定一个进程运行了多长时间，进程使用了多少CPU时间，以及进程是否受系统的惩罚。还可用ps命令确定进程使用了多少内存，完成多少I/O，进程的优先级以及是谁创建了进程。下面这几个命令组合对于管理AIX系统有帮助：

- (1) 显示10个消耗CPU最多的进程： # ps aux | head -1 ;ps aux | sort -rn +2 | head -10
- (2) 显示10个消耗存储空间最多的进程： # ps aux | head -1 ;ps aux | sort -rn +3 | head -10
- (3) 按顺序显示系统中受罚的进程： #ps -eakl | head -1 ;ps -eakl | sort -rn +5
- (4) 按优先级顺序显示系统中的进程： #ps -eakl | sort -n +6 | head
- (5) 按处理时间为顺序显示系统中的前十个进程： #ps vx | head -1 ;ps vx | grep -v PID | sort -rn +3 | head -10
- (6) 按实际内存使用的多少顺序显示系统中的前十个进程： #ps vx | head -1 ;ps vx | grep -v PID | sort -rn +6 | head -10
- (7) 按换入页面的多少顺序显示系统中的前10个进程： #ps vx | head -1 ;ps vx | grep -v PID | sort -rn +4 | head -10

下图是AIX系统ps命令输出:

```
blondie# ps -eakl | sort -n +6 | head
  F S UID      PID      PPID    C PRI NI ADDR      SZ      WCHAN      TTY  TIME CMD
 303 A   0        0          0    0 120  16 -- 31004000    60              -  1:44 s
wapper
 303 A   0    16392          0    0  16 -- 4100a000    72 f10008000929ea40 -
0:00 lrud
 303 A   0    24588          0    0  16 -- 6100e000    64              -  0:00 x
mfreed
 303 A   0    28686          0    0  16 -- 91010000    56              -  0:00 m
emgrdd
 303 A   0    32784          0    0  16 -- 81012000    52              -  0:00 p
sgc
 303 A   0    65568          0    0  16 -- b1034000    68 3d15498      -  0:00 w
lmsched
 303 A   0    69666          0    0  16 -- a9037000    52              -  0:00 a
rmtrace_kproc
 303 A   0    12294          0 120  17 -- 51008000    56              -  0:02 s
ched
40201 A   0   495860          0    0  17 20 b91d3000   212              -  0:00 d
og
```



sar

sar命令报告CPU的使用情况，I/O以及其它系统行为。sar命令可以收集，报告以及保存系统行为信息。如果没有指定输入文件，则sar调用sarc命令访问系统数据。用户可用让cron命令运行两个shell脚本 (/usr/lib/sa/sa1和/usr/lib/sa2) 以提供日统计和报表。在crontab文件/var/spool/cron/crontabs/adm中包括了一些样本节，用于示范cron要在何时运行这些shell脚本。以这种方式收集到的数据对于确定系统的时间周期特征和决定峰值使用时间是有帮助的。但要注意的是，sar命令自己运行时会产生相当数量的读写。因此最好在没有任何工作量的情况下运行sar统计，看看sar对总的统计数字有多大的影响。


```
ibm-550-1-lpar1# sar -u 60 5

AIX ibm-550-1-lpar1 3 5 00C12EFE4C00      10/09/12

System configuration: lcpu=4 ent=1.00 mode=Capped

23:05:15      %usr      %sys      %wio      %idle      physc      %entc
23:06:15          51          37          0          12          1.00      100.0
23:07:15          50          37          0          12          1.00      100.0
23:08:15          51          37          0          12          1.00      100.0
23:09:15          51          37          0          12          1.00      100.0
23:10:15          51          37          0          12          1.00      100.0
Average          51          37          0          12          1.00      100.0
```



EMC易安信中国技术社区

显示内容包括：

%usr：CPU处在用户模式下的时间百分比

%sys：CPU处在系统模式下的时间百分比

%wio：CPU等待输入输出完成时间的百分比

%idle：CPU空闲时间百分比

physc：消耗物理处理器的数目，只在共享处理器或启用同时多线程的分区上报告

在所有的显示中，我们应主要注意%wio和%idle，%wio的值过高，表示硬盘存在I/O瓶颈，%idle值高，表示CPU较空闲，如果%idle值高但系统响应慢时，有可能是CPU等待分配内存，此时应加大内存容量。%idle值如果持续低于10，那么系统的CPU处理能力相对较低，表明系统中最需要解决的资源是CPU。

浅谈主机FC-HBA卡更换

原创 EMC中国现场支持 [戴尔易安信技术支持](#) 2016-07-22

有时候我们会接到客户的来电询问，“我主机的HBA卡坏了，怎么更换？存储需要更改什么设置啊？对于主机有没有影响啊？”等等之类的问题。

下面我简单的谈一下更换主机HBA卡的方法和步骤，将围绕EMC目前主流的存储Symmetrix，VNX，Clariion进行介绍。

在进行更换之前，我们需要做以下工作：

1. 确认主机状态, 是否有磁盘单链路情况 (EMC Powerpath 命令为powermt display dev=all, 链路上是否存在 dead链路, 其他多路径软件请咨询相关厂商)
2. 确认主机HBA槽位, 并且确认主机连接的光纤线标签是否清楚, 我们需要非常清楚的了解主机连接的交换机端口和存储端口
3. 是否连接交换机
4. 如果连接交换机, 确认交换机上的端口
5. 接着根据端口情况, 确认交换机是哪种类型的zone 设置 (如果是WWN Zone, 请注意需要重新定义新的 zone因为新的HBA会有不同的WWN, 如果是port zone则不需要任何操作)

以下例子是在线的更换HBA的例子 (基于IBM AIX系统及 EMC Powerpath)

\1. Powermt display 确认故障HBA的序列号

```
# powermt display
Symmetrix logical device count=214
CLARiON logical device count=0
Hitachi logical device count=0
HP xp logical device count=0
Ess logical device count=0
Invista logical device count=0
HP HSx logical device count=0

----- Host Bus Adapters -----
### HW Path                               Summary    Total    Dead    Io/Sec  Q-Ios  Errors
----- I/O Paths -----
0 fscsi0                                optimal    214      0      0      0      0
1 fscsi1                                optimal    214      0      0      0      0
```

\2. Powermt remove hba=# 删除故障HBA上磁盘链路 (此步可以拔出光纤)

\3. rmdev 命令删除相关HBA卡

\4. 更换HBA卡, 插上光纤 (是否需要停机由主机厂商决定)

\5. 在交换机上找到相对应端口的HBA WWN(记录下)

\6. 如果是WWN Zone, 需要使用新的HBA WWN更新ZONE

\7. 下一步开始我们需要针对不同的存储分开讨论

我们先从 Symmetrix 开始介绍

首先从存储管理员那里了解故障主机连接的Symmetrix 磁盘是否使用了device masking(VCM).

如果回答是没有, 说明新的HBA卡可以直接访问该磁盘。

如果有, 则需要通过以下命令把新的HBA卡更新到VCM

Symmashdb list -database

```
# symmaskdb -sid 1112 list database
```

Symmetrix ID : 000290301112

Database Type : Type6

Last updated at : 05:28:52 PM on Thu Oct 25,2012

Director Identification : FA-2C

Director Port : 0

Identifier	Type	User-generated Node Name	Port Name	Devices
10000000c96fc4fa	Fibre	linux2_new	10000000c96fc4fa	0023:0091 0351:0352 0355 0359
10000000c966b95b	Fibre	aix2_new	10000000c966b95b	0092:0100
210000e08b1cb342	Fibre	w2k3-2-qla0	210000e08b1cb342	0031 0101:016F
210000e08b1c3543	Fibre	w2k3-2-qla1	210000e08b1c3543	0101:016F
10000000c92edca1	Fibre	sun2_c3_new	10000000c92edca1	0170:01DE
10000000c92aa277	Fibre	sun2_c4	10000000c92aa277	None

Director Identification : FA-16C

Director Port : 0

Identifier	Type	User-generated Node Name	Port Name	Devices
10000000c92aa277	Fibre	sun2_c4	10000000c92aa277	0170:01DF
10000000c966b95b	Fibre	aix2_new	10000000c966b95b	None
10000000c96fc4fa	Fibre	linux2_new	10000000c96fc4fa	None

Director Identification : FA-2D

Director Port : 1

Identifier	Type	User-generated Node Name	Port Name	Devices
50060b00002d8bf6	Fibre	HP2_fcd1	50060b00002d8bf6	04E8:0556

上图我们看到相关的端口访问的主机磁盘。

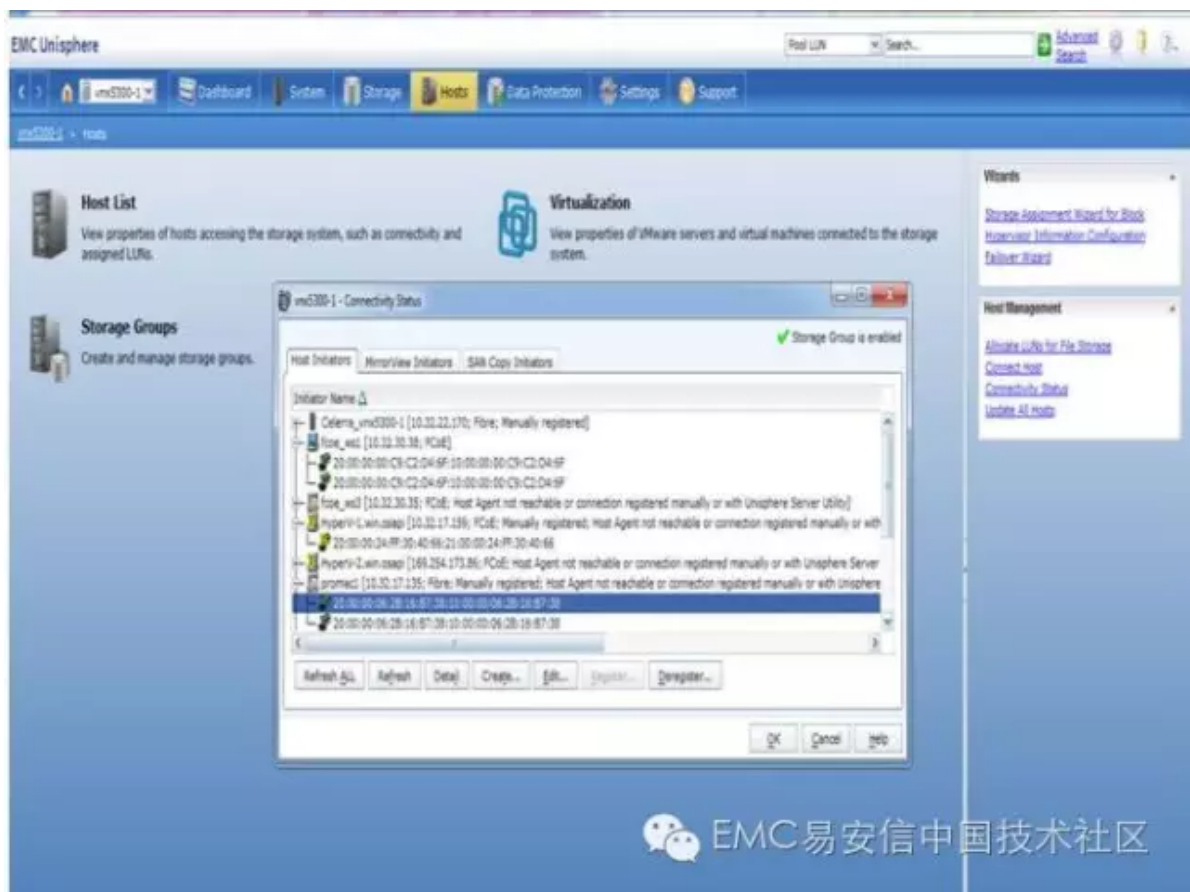
如果我们定位连接在16C port 0上的HBA卡 10000000C92aa277为故障卡需要更换的话

只需键入symmask -sid 1112 -wwn 10000000C92aa277 -replace new WWN

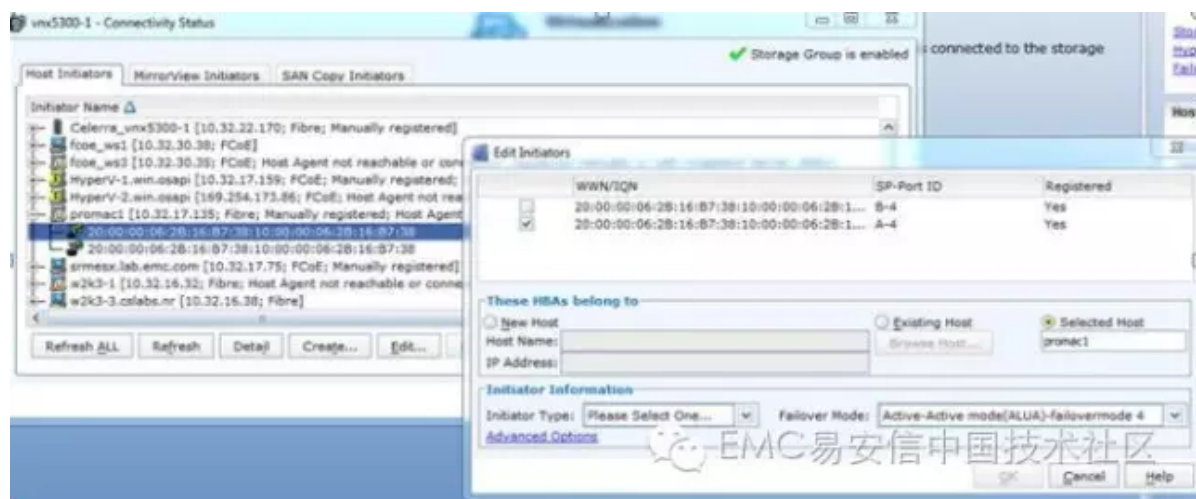
Symmask -sid 1112 refresh

这样对于symmetrix 上的主机操作就完成了

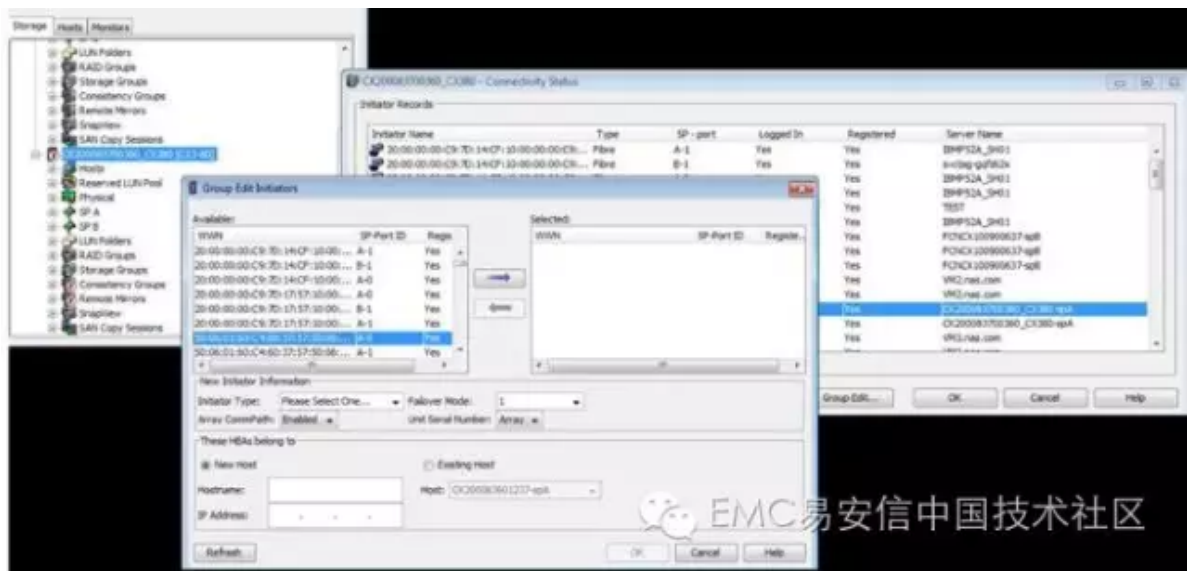
然后介绍VNX Unisphere上的相关设置



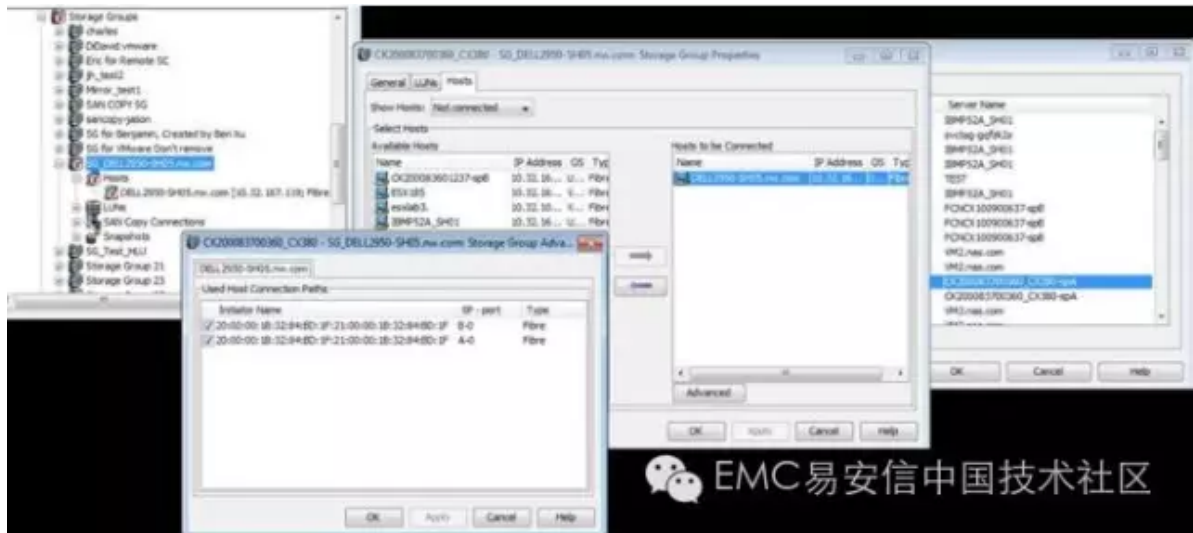
进入unisphere,,, 选择HOST TAB , 然后选择右侧的 Connectivity Status找出故障主机的HBA卡, 选择 Deregister (需要Engineering mode) .. 然后选择新的HBA, 点击register, 注意编辑initiator type 和 failover mode, 选择的参数应与该台主机另外块HBA相同.



Storage group确认新的HBA卡正常添加在正确的SG里, 选择主机, 然后点advance 选项, 查看是否两块HBA 卡前已经打钩。



Storage group确认新的HBA卡正常添加在正确的SG里，选择主机，然后点advance 选项，查看是否两块HBA 卡前已经打钩。



以上列出了不同存储需要的具体设置，待完成以后，我们需要再次扫盘。

```
/usr/lpp/Symmetrix/bin/emc_cfgmgr //在主机重新扫盘（针对AIX主机，其他主机请查阅相关文档）
powermt config //powerpath恢复磁盘路径
powermt save //powerpath 保存磁盘路径信息
powermt display //确认磁盘链路状况
```

最后期待所有磁盘路径都alive吧。。。希望以上的一些步骤能帮到大家！

