Linux操作系统自如的装卸内核模块 PDF转换可能丢失图片或格式，建议阅读原文
https://www.100test.com/kao_ti2020/145/2021_2022_Linux_E6_93_8D_E4_BD_c103_145247.htm Linux是单内核结构，也就是说，它是一个大程序，其中任一函数都可以访问公共数据结构和其它函数调用。(作为操作系统)另外一种可能的结构是多核式的，各功能块自成一体，相互之间由严格的通信机制相连。单核结构在添加新模块时，一种方法是重新调整设置，所以非常费时。比如，你想在内核中加一个NCR 810 SCSI的驱动程序，你必须重新设置，重建内核。这也有另外一个办法,Linux 允许动态装载和卸掉模块. Linux 模块是一段可以在机器起动后任意时间被动态连接的代码. 在不需要时, 它们可以被从内核中卸掉. 大多数Linux 模块是设备驱动程序或伪设备驱动程序, 如网络驱动程序, 文件系统等。你可以使用 insmod 和 rmmod 命令来装载和卸掉 Linux 模块, 内核自己也可以调用内核驻留程序(Kerneld) 来按需要装载和卸掉模块。按需动态装载模块可以使内核保持最小, 并更具灵活性. 我现在的 Intel 内核由于大量使用动态装载模块, 只有 406 K 字节. 例如，我很少用到 VFAT 文件系统, 所以我让 Linux 内核只在我装载 VFAT 分区时, 才自动上载 VFAT 文件系统. 当我卸掉 VFAT 分区时, 内核会检测到, 并自动卸掉 VFAT 文件系统. 当测试新程序时, 你如果不想每次都重建内核, 动态装载模块是非常有用的. 但是, 运用模块会多消耗一些内存, 并对速度有一定影响. 并且模块装载程序是一段代码, 它的数据将占用一部份内存. 这样还会造成不能直接访问内核资源, 效率不高的问题. 一旦 Linux 模块被装载后, 它就和一般内核代码一样, 对其

它内核代码, 享受同样的访问权限。换句话说， Linux 内核模块可以像其它内核代码, 或驱动程序一样使系统崩溃。模块可以使用内核资源，但首先它需知道怎样调用. 例如, 一个模块要调用 Kmalloc() (内核内存分配程序). 但在模块建立时, 它并不知道到哪儿去找 Kmalloc(), 所以在它被装载时, 内核必须先设定模块中所有 Kmalloc() 调用的函数指针. 内核有一张所有资源调用的列表, 在模块被装载时, 内核重设所有资源调用的函数指针. Linux 允许栈式模块, 即一个模块调用另一个模块的函数. 例如, 由于 VFAT 文件系统可以看成是 FAT 文件系统的超集, 所以 VFAT 文件系统模块需要调用 FAT 文件系统提供的服务。一个模块调用另一模块的资源与调用内核资源很相似。唯一的不同是被调用的模块需被先载入。一个模块被载入后，内核将修改它的内核符号表(KERAEL SYMOBOL TABLE)，加入新载入模块提 供的所有资源和符号。所以另一个模块被载入时， 它就可以调用所有已载入模块提供的服务。 当卸掉一模块时，内核先确定该模块不会再被调用，然后通过某种方式通知它。在该模块被内核卸掉以前，该模块须释放所有占用的系统资源。例如，内存或中断，当模块被卸掉后，内核从内核符号表中删除所有该模块提供的资源。 如果模块代码不严谨，它将使整个操作系统崩溃。另一个问题，如果你载入的是为其它版本服务的模块，那怎么办?例如，一个模块凋用一个内函数，但提供了错误的输入参数，这将导致运行错误。但内核可以在模块被载入时选择性地通过严格版本检查来杜绝这种现象。 载入模块有两种方法。第一种是通过INSTALL 命令来载入.另一种更聪明的方法是在模块被调用时自动载入，这叫所需载入(DEMAND LOADING)。例如，

当用户在装一个不在内核中的文件系统，内核会自动调用内核驻 留程序(KERNELD)来载入对应的处理模块。 内核驻留程序是一个具有超级用户极限的普通用户程序。当它被启动时(通常在系统启动时)，它将打开一个和内核之间的进程间通信管道(IPC CHANNEL)。 内核将利用这条管道来通知进程驻留程序去完成各种任务。内核驻留程序的主要任务是载入和卸掉模块，它也能完成其她一些任务。如按需打开和关掉一条通过串口的 DDD LINK。KERNELD 自己并不完成这些任务。它将调用如INSMOD 这样的命令来完成，KERNELD 只是一个内核的代理，协调完成各项任务。 载入模块 载入模块时，INSMODE 命令必须先找到要被载入的模块。可所需载入的模块通常被放在/LIB/MODULES/KERNEL-VERSION下，这些模块与一般系统程序都是已连接好的目标代码，不同处在于模块是可重定位的映像文件。 也就是说，模块并不是从一个固定的地址开始执行的。模块可以是 a.out， 也可以是ELF 格式的目标代码。INSMODE 通过一个有系统权限的调用来找到内核中可被调用的资源。 系统(资源)符号由名和值俩部份组成。内核用MODULE_LIST 指针指向其管理的所有模块所串成的链表。内核的输出符号表在第一个MODULE 数据结构中，并不是内核所有的符号都能被模块调用，可调用符号必须被加入输出符号表中，而输出符号表是与内核一起编译连接的。例如，当一驱动程序想控制某一系统中断时，她需调用 ” REQUEST_IRQ ” 这样一个系统函数，在我机器的内核中，它现在的值是0x0010cd30，你可以看/PROC/KSYMS文件或用KSYMS 来查询。KSYMS 命令可以显示所有内核输出符号的值，也可以显示载入模块的输出符号的值。当INSMOD 载

入模块时，它先将模块载入虚存，根据内核输出符号，重设所有内核资源函数调用的指针。即在模块的函数调用处写入对应符号的物理地址。 当INSMOD 重设完内核输出符号的地址后，它将调用一个系统函数，要 求内核分配足够的空间。内存就会分配一个新的MODULE 数据结构和足够的内存来装载这个新模块，并把这个MODULE 数据结构放在模块链表的最后， 置成未初使化(UNINITALIZED)。 显示的是内核载入FAT 和VFAT 两模块后的模块链表。链表的第一模块并没有显示出来，那是一个伪模块，只是用来记录内核的输出符号表。你可以用ISMOD命令来列出所有载入模块及它们之间的关系。ISMOD只是格式化的输出记录内核链表的/PROC/MODULES文件。INSMOD 可以访问内核分配给新载入模块的内存，它先将模块写入这块内存，然后对它进行重定位处理，使模块可以从这个地址开始执行。由于每次模块被载入时，无论在不在同一台机器上，都不大可能分配到相同的内存地址，所以重定位(即重设它的函数指针)是必须的。 新载入模块也可以输出符号，INSMOD 会为这些符号建一个表。另外，每一个模块必须有自己的初始和清理(即析构)函数。这两个函数不能被输出，但它的地址将在初使化时由INSMOD 传给内核。 当一个新模块被载入内核时，它要更新系统符号表及被它调用的模块。内核中被调用模块都需在其符号表的最后保留一列指向调用模块的指针。显示VFAT文件系统依赖于FAT文件系统，所以，在FAT 模块中有一个指向VFAT 的指针，这个指针是在VFAT被载入时加入的。内核将调用模块的初使化函数，如果成功，它将继续完成安装新模块的任务。模块的清理函数的地址将被存在它的MODULE

数据结构中。当模块被卸掉时，它将被调用。到这里模块的状态被置为“ 运行“ (RUNNING)。 卸掉模块 用RMMOD命令可以卸掉一个指定模块，但按需载入模块没用时，它会被内核自动卸掉，KERNELD每次被激活时，它会调用一个系统函数将所有没用的模块从内核中卸掉。例如，如果你装了一个ISO9660的CDROM，并且它的文件系统是一个按需载入模块，那么当你卸掉CDROM 后不久，ISO9660文件系统也会被从内核中卸掉。你可以在起动KERNELD 时，设置其被激活的时间间隔，我的KERNELD 每180秒被激活一次。 当还在被其他模块调用时，模块是不能被卸掉的。例如，当你还在用VFAT文件系统时，VFAT模块不会被卸掉。当你看ISMOD命令的输出时，你会发现每个模块都带有一个计数器。这个计 数器记录依赖于该模块的模块数。在上面的例子中，VFAT 和MSDOS 都依赖于FAT模块，所以FAT 模块的计数器为2，VFAT 和MODOS的都为1，表示只有文件系统依赖于它们。如果我再装入一个VFAT文件系统，VFAT模块的计数器将变成2。模块的计数器是它映像的第一个长字(LONGWORD)。 这个长字同时也记录了AUTOCLEAN 和VISITED 两个标志，只有按需载入模块才用到这两个标志。AUTOCLEAN用来使系统识别哪一个模块需被自动卸掉。VISITED 标志表示该模块是否还在被其它模块调用。每次KERNELD 试图卸掉已没用的按需载入模块时，系统检查所有模块。它只注意标为AUTOCLEAN并正在运行的模块，如果这个模块没有设置VISTIED 标志，它将被卸掉。否则，系统就清掉VISTIED标志，并继续检查下一模块。 当一个模块可以被卸掉时，系统会调用它的清理函数来释放它所占用

的所有系统资源。 该模块的MODULE数据结构将被标为DELEDTED，并从模块链表中去除，所有它依赖的模块会修改它们的指针，表示该模块已不再依赖它们了。所有该模块占用的内存将被释放掉。 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com