 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。 