文章编号:1008-0570(2007)07-2-0048-02

Linux可加载内核模块机制的研究与应用

The Research and Application of Linux Loadable Kernel Module Mechanism

(1.东北大学;2.沈阳师范大学)刘天华[1,2]　陈　枭[2]　朱宏峰[1,2]　刘　骏[1,2]
LIU TIANHAU　CHEN XIAO　ZHU HONGFENG　LIU JUN

摘要: 为了缩短 Linux 开发和测试的时间，提高开发效率。详细分析了 Linux 可加载内核模块机制即 LKM (Loadable Kernel Module)的工作原理、实现细节和 LKM 机制中的关键技术，总结了在实际开发中针对最新内核稳定版本 2.6 编写内核模块的方法和需要注意的问题。在 Linux 操作系统环境下应用 LKM 对零拷贝原理进行了实现。在 Linux 开发和研究中应用 LKM 可以大大加快开发速度。

关键字:LKM; Linux; 零拷贝
中图分类号:TP393.04　**文献标识码**:A

Abstract:To shorten the time of Linux developing and testing, improve developing efficiency. This paper particularly analyses the work principle, implementing details and key technology of Linux loadable kernel module mechanism viz. LKM, summarizes the coding method and notice proceeding in allusion to the latest kernel stabilization edition 2.6 in developing. In the Linux environment implementing Zero-copy principle with LKM, the results indicate the application of LKM in Linux developing and researching can quicken the speed of developing greatly.

Key words: LKM, Linux, Zero-copy

技术创新

1 引言

Linux系统开放源代码、系统漏洞少，在面对病毒和黑客入侵时能提供更好的安全性和稳定性，基于以上这些优点，近年来对 Linux 操作系统及其相关技术的应用和研究越来越多。对 Linux 操作系统扩充或裁剪功能需要在重新编译内核上花费大量的时间。LKM 机制由于大大缩短了开发和测试的时间，在 Linux 开发、研究的过程中起到了举足轻重的作用。

LKM 主要包括内核模块在操作系统中的加载和卸载两部分功能，内核模块是一些在启动的操作系统内核需要时可以载入内核执行的代码块，不需要时由操作系统卸载。它们扩展了操作系统内核功能却不需要重新编译内核、启动系统。如果没有内核模块，就不得不反复编译生成操作系统的内核镜像来加入新功能，当附加的功能很多时，还会使内核变得臃肿。

2 LKM 的编写和编译

2.1 内核模块的基本结构

一个内核模块至少包含两个函数，模块被加载时执行的初始化函数 init_module()和模块被卸载时执行的结束函数 cleanup_module()。在最新内核稳定版本 2.6 中，两个函数可以起任意的名字，通过宏 module_init()和 module_exit()实现。唯一需要注意的地方是函数必须在宏的使用前定义。例如:

```
static int __init hello_init(void){}
static void __exit hello_exit(void ){}
module_init(hello_init);
module_exit(hello_exit);
```

这里声明函数为 static 的目的是使函数在文件以外不可见，__init 的作用是在完成初始化后收回该函数占用的内存，宏 __exit 用于模块被编译进内核时忽略结束函数。这两个宏只针对模块被编译进内核的情况，而对动态加载模块是无效的。这是因为编译进内核的模块是没有清理收尾工作的，而动态加载模块却需要自己完成这些工作。

2.2 内核模块的编译

编译时需要提供一个 makefile 来隐藏底层大量的复杂操作，使用户通过 make 命令就可以完成编译的任务。下面就是一个简单的编译 hello.c 的 makefile 文件:

```
obj-m += hello.ko
KDIR := /lib/modules/$(shell uname -r)/build
PWD := $(shell pwd)
default:
$(MAKE) -C $(KDIR) SUBDIRS=$(PWD) modules
```

编译后获得可加载的模块文件 hello.ko。

内核版本 2.6 中使用.ko 文件后缀代替了.o，这是为了与普通可执行文件相区别。

3 LKM 的主要功能

3.1 模块的加载

模块的加载有两种方法，第一种是使用 insmod 命令加载，另一种是当内核发现需要加载某个模块时，请求内核后台进程 kmod 加载适当的模块。当内核需要加载模块时，kmod 被唤醒并执行 modprobe，同时传递需加载模块的名字作为参数。modprobe 像 insmod 一样将模块加载进内核，不同的是在模块被加载时查看它是否涉及到当前没有定义在内核中的任何符号。如果有，在当前模块路径的其他模块中查找。如果找到，它们也会被加载到内核中。但在这种情况下使用 insmod，会以‘未解析符

刘天华: 教授　博士
基金项目:本项目受辽宁省自然基金资助(20042042);
辽宁省教育科学基金项目资助(05L420)

号”信息结束。

关于模块加载，可以用图 3.1 来简要描述：

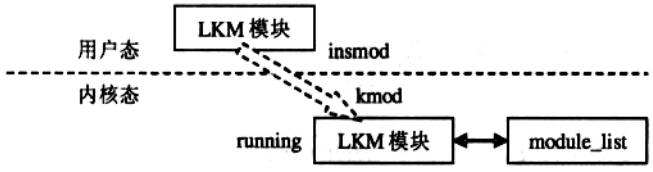

图 3.1 LKM 模块的装入

insmod 程序必须找到要求加载的内核模块，这些内核模块是已链接的目标文件，与其他文件不同的是，它们被链接成可重定位映象即映象没有被链接到特定地址上。insmod 将执行一个特权级系统调用来查找内核的输出符号，这些符号都以符号名和数值形式如地址值成对保存。内核输出符号表被保存在内核维护的模块链表的第一个 module 结构中。只有特殊符号才被添加，它们在内核编译与链接时确定。insmod 将模块读入虚拟内存并通过使用内核输出符号来修改其未解析的内核函数和资源的引用地址。这些工作采取由 insmod 程序直接将符号的地址写入模块中相应地址来进行。

当 insmod 修改完模块对内核输出符号的引用后，它将再次使用特权级系统调用申请足够的空间容纳新模块。内核将为其分配一个新的 module 结构以及足够的内核内存来保存新模块，并将其插入到内核模块链表的尾部，最后将新模块标志为 UNINITIALIZED。insmod 将模块拷贝到已分配空间中，如果为它分配的内核内存已用完，将再次申请，但模块被多次加载必然处于不同的地址。另外此重定位工作包括使用适当地址来修改模块映象。如果新模块也希望将其符号输出到系统中，insmod 将为其构造输出符号映象表。每个内核模块必须包含模块初始化和结束函数，所以为了避免冲突它们的符号被设计成不输出，但是 insmod 必须知道这些地址，这样可以将它们传递给内核。在所有这些工作完成以后，insmod 将调用初始化代码并执行一个特权级系统调用将模块的初始化和结束函数地址传递给内核。当将一个新模块加载到内核中时，内核必须更新其符号表并修改那些被新模块使用的老模块。那些依赖于其他模块的模块必须在其符号表尾部维护一个引用链表并在其 module 数据结构中指向它。内核调用模块的初始化函数，如果成功将安装此模块。模块的结束函数地址被存储在其 module 结构中，将在模块卸载时由内核调用，模块的状态最后被设置成 RUNNING。

3.2 模块的卸载

模块可以使用 rmmod 命令删除，但是请求加载模块在其使用计数为 0 时，自动被系统删除。kmod 在其每次 idle 定时器到期时都执行一个系统调用，将系统中所有不再使用的请求加载模块删除。

关于模块卸载，可以用图 3.2 来描述：

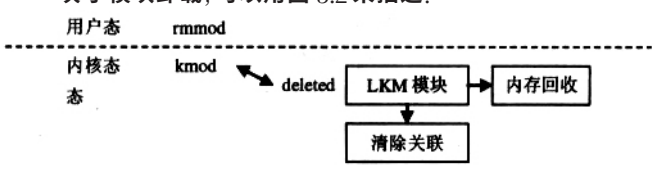

图 3.2 LKM 模块的卸载

内核中其他部分还在使用的模块不能被卸载。例如系统中安装了多个 VFAT 文件系统则不能卸载 VFAT 模块。执行 lsmod 将看到每个模块的引用计数。模块的引用计数被保存在其映象的第一个常字中，这个字还包含 autoclean 和 visited 标志。如果模块被标记成 autoclean，则内核知道此模块可以自动卸载。visited 标志表示此模块正被一个或多个文件系统部分使用，只要有其他部分使用此模块则这个标志被置位。每次系统要将没有被使用的请求加载模块删除时，内核将在所有模块中扫描，但是一般只查看那些被标志为 autoclean 并处于 running 状态的模块。如果某模块的 visited 标记被清除则它将被删除。其他依赖于它的模块将修改各自的引用域，表示它们间的依赖关系不复存在。此模块占有的内核内存将被回收。

4 LKM 的应用

零拷贝基本思想是：数据分组从网络设备到用户程序空间传递的过程中，减少数据拷贝次数，减少系统调用，实现 CPU 的零参与，彻底消除 CPU 在这方面的负载。零拷贝的实现分为实现 DMA 数据传输和地址映射两个部分。其中 DMA 数据传输与本文关系不大，就不详细叙述了，这里主要介绍应用 LKM 机制实现的地址映射。

地址映射的基本原理是在内核空间申请内存，通过 proc 文件系统和 mmap 函数将其映射到用户空间来允许应用程序访问，这样就消除了内核空间到应用程序空间的数据拷贝。地址映射部分的实现主要分为以下三步：

第一，建立 LKM 的基本结构，包括编写初始化和结束函数等。

第二，声明完成映射功能所需要的函数，主要有分配和初始化内核内存函数 init_mem()，释放内核内存函数 del_mem()，向内核内存输入内容的函数 put_mem()等。

第三，在初始化函数中应用第二步建立的函数分配一块内存空间、输入内容、建立 proc 文件系统入口。在结束函数中释放已分配的内核内存，删除 proc 文件系统入口。

编写应用程序测试该 LKM，发现已经达到了映射内核内存到应用程序空间的目的。在实现零拷贝的过程中采用 LKM 机制不但便于调试而且大大减少了开发时间。

5 LKM 与普通应用程序的比较

LKM 与普通应用程序之间的区别主要体现在四个方面。

第一，也是最重要的区别，普通应用程序运行在用户空间，而 LKM 运行在内核空间。通过区分不同的运行空间，操作系统能够安全地保护操作系统中一些重要数据结构的内容不被普通应用程序所修改，达到保证操作系统正常运转的目的。

第二，普通应用程序的目标很明确，它们从头至尾都是为了完成某一项特定任务。而 LKM 是在内核中注册并为后续应用程序的请求提供服务的。

第三，普通应用程序可以调用并没有在其中定义的函数，但一个 LKM 是链接到内核上的，它所能调用的函数只有内核导出来的那些函数。

第四，普通应用程序和 LKM 处理错误的方式不同。当应用程序中出现错误时并不会给系统造成很大的伤害。LKM 则不然，在其中出现的错误对子系统来说通常是致命的，至少对于当前正在运行的进程而言。LKM 中的一个错误常常会导致整个系统崩溃。

6 编写 LKM 需要注意的问题

LKM 运行在内核空间，它们拥有对整个系统所有资源的访问权限，因此，编写 LKM 首先要注意就是安全问题，而且还应该避免将可能导致出现安全问题的代码带到 LKM 中。

(下转第 127 页)

技术创新

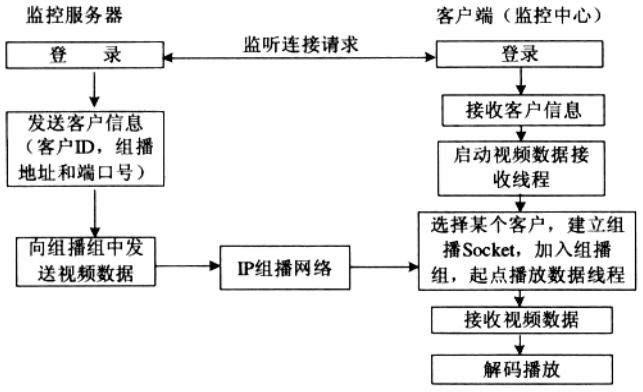

图 4 视频网络传输流程

5 结束语

嵌入式视频监控系统不仅具有抗干扰能力强，适合远距离传输，能够加密，可用计算机对图像信息进行压缩、分析、存储和显示，充分利用现有网络资源等诸多优点，而且具有体积小、功耗低、易于安装、使用方便和便于维护等优点。该系统以TCP/IP网络为传输媒介，采用组播技术实现视频信号在网上的传输，在网络的任意位置都可实现对整个监控系统的指挥、调度、存贮和授权控制。

创新观点：数字视频监控系统取代了模拟视频监控系统后，得到了广泛应用。随着嵌入式技术和网络技术的发展，出现了基于嵌入式和Internet的视频监控系统。该系统的核心是以S3C45B0为核心的视频服务器，以TCP/IP网络为传输媒介，采用组播技术实现视频信号在网上的传输，在网络的任意位置都可实现对整个监控系统的指挥、调度、存贮、授权控制。

参考文献

[1]景慧燕.一种嵌入式移动视频监控系统的设计[J]. 电视技术. 2005，11.91-93

[2]何小敏，张小花.智能化远程图像监控系统的研究及其应用[J]. 组合机床与自动化加工技术.2004(9).13-15

[3]景绍学，李正明，宋永献等.S3C4510B在远程网络视频监控中的应用[J]微计算机信息 2006，8-2:14-16

[4]徐兵.基于Web的远程视频监控系统在自动化中的设计与应用[J]微计算机信息.2006，10-1:286-287

[5]黄贤英，田淑宁.包装车间数字化无线视频监控系统的研究与实现[J].包装工程.2006，4:207-208

[6]胡勇华，谢宝昌，李军.基于ARM的无线视频监控系统的设计[J].电工技术.2006,3:48-50

作者简介：刘恒洋(1977-)，男，江西人，硕士，工程师，主要研究方向：嵌入式计算机技术；王森(1978-)，男，河南人，硕士，软件设计师，主要研究方向：软件工程。

Biography: Liu Hengyang, Male, born in 1977, the HAN nationality, master degree, Engineer, Research Area: application and research of embedded system; Wang Sen, Male, born in 1978, the HAN nationality, master degree, software engineer, Research Area: software engineering.

(400050 重庆 重庆工学院计算机系)刘恒洋 王 森

(Chongqing institute of technology, Chongqing, 400050, China) Liu Hengyang Wang Sen

通讯地址：(400050 重庆 重庆工学院计算机系)刘恒洋

(收稿日期：2007.5.23)(修稿日期：2007.6.25)

(上接第 49 页)

LKM加载后是作为操作系统内核的一部分运行的，因此，在设计、编写操作系统内核过程中应该注意的问题在LKM中也应该引起足够的重视。在这里，主要指的是并发问题和指针引用问题。并发是指在同一时间有多个进程在操作系统内核中同时运行。并发结合共享资源最终会导致竞态条件，在这种情况下应该对各个并发进程访问共享资源进行严格的控制。如果在LKM中出现指针引用错误，内核将没有办法将内存的虚拟地址映射到物理地址，从而导致出现内核中的意外，如内存访问冲突、除0以及非法操作等。

7 LKM的不足之处

LKM虽然在设备驱动程序的编写和扩充内核功能中扮演着非常重要的角色，但它仍有许多不足的地方。

第一，LKM对于内核版本的依赖性过强，每一个LKM都是靠内核提供的函数和数据结构组织起来的。当这些内核函数和数据结构因为内核版本变化而发生变动时，原先的LKM不经过修改就可能不能正常运行。

第二，虽然现在有针对内核编程调试的工具kgdb，但是在LKM编写过程中调试仍非常麻烦，而且在调试过程中，系统所能提供的出错信息极为晦涩。

本文作者创新点：针对Linux内核，利用LKM，在实现了数据的零拷贝(Zero-copy)的过程中，将LKM与普通应用程序进行比较，提出了LKM的优势和不足。

参考文献

[1]任家东，梁哲，赵黎.网络协议的构件化方法研究与实现，[J]微计算机信息，2006，22-17，85-87.

[2]Peter Jay Salzman, Michael Burian, Ori Pomerantz. The Linux Kernel Module Program-ming Guide [M], 2001: 5-43.

[3]Henderson B. Linux Loadable Kernel Module HOWTO, http://www6.uniovi.es/linux/H-OWTO/Module-HOWTO/, 2002.

[4]徐伟，贾春福.扩充Linux系统功能的LKM技术[J]，2003，第四期:100-102.

[5]毛德操，胡希明.Linux内核源代码情景分析[M]，浙江：浙江大学出版社，2001年9月，277-280.

[6]Alessandro Rubini, Jonathan Corbet, Greg Kroah-Hartman. Linux Device Drivers Third Edition [M], O'Reilly

作者简介：刘天华(1966-)，男(汉族)，辽宁省沈阳市人，沈阳师范大学教授，博士，主要研究领域为：计算机网络、计算机体系结构、信息安全；

Biography: Liu Tian-hua(1966-), male(Han), Shenyang Liaoning, Dept of Software, Shenyang Normal University, professor, doctor, mostly researching domain: computer network, computer architectonic, information security;

(110004 沈阳 东北大学信息科学与工程学院)刘天华 朱宏峰 刘骏

(110034 沈阳 沈阳师范大学科信软件学院)刘天华 陈枭 朱宏峰 刘骏

通讯地址：(110004 沈阳 东北大学信息科学与工程学院)刘天华

(收稿日期：2007.4.23)(修稿日期：2007.5.25)

技术创新