



(12) 发明专利

(10) 授权公告号 CN 101540720 B

(45) 授权公告日 2011.06.15

(21) 申请号 200810114690.8

US 6807172 B1, 2004.10.19, 全文.

(22) 申请日 2008.06.06

CN 1298241 A, 2001.06.06, 全文.

(73) 专利权人 曙光信息产业(北京)有限公司

审查员 杨盈霄

地址 100084 北京市海淀区水磨西街 64 号

(72) 发明人 曾宇 历军 聂华 刘朝辉

(74) 专利代理机构 北京安博达知识产权代理有限公司 11271

代理人 徐国文

(51) Int. Cl.

H04L 12/56(2006.01)

H04L 12/02(2006.01)

G06F 13/28(2006.01)

(56) 对比文件

US 6292492 B1, 2001.09.18, 全文.

CN 1479498 A, 2004.03.03, 全文.

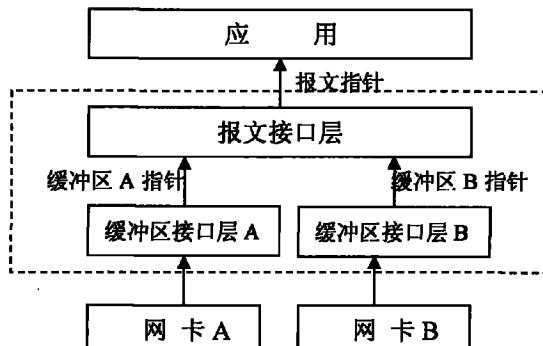
权利要求书 2 页 说明书 4 页 附图 1 页

(54) 发明名称

一种支持多类型网卡链路聚合的零拷贝方法

(57) 摘要

本发明涉及零拷贝捕包方法,具体地说是一种可支持多类型网卡、实现链路聚合功能的零拷贝方法,所述零拷贝方法包括A缓冲区接口层函数打开设备接口;B在用户空间缓冲区;C构造缓冲区队列;D接口函数通过内存映射得到缓冲区控制结构的信息;E报文接口层函数打开设备接口;F为应用新建报文指针队列,从缓冲区中提取出报文指针放入报文指针队列,应用程序报文指针队列中的报文数据进行处理。本发明采用分层结构,将现有的应用接口层划分为不关心具体设备缓冲区结构的缓冲区接口层和根据设备类型从缓冲区提取报文的报文接口层,实现了同一台机器上混合使用多种类型的网卡,并且方便了软件的移植和扩展。



1. 一种支持多类型网卡链路聚合的零拷贝方法,其特征在于,所述方法包括如下步骤:

- A) 应用程序调用接口库提供的缓冲区接口层函数打开设备接口,对各网卡指定一个用于区分不同类型网卡的类型号码,将各类型号码记录在设备类型列表中;在各网卡的内核驱动中都增加一个用于返回网卡类型号码的系统调用命令;
- B) 缓冲区接口层提供的打开设备接口函数在用户空间中申请一片连续的内存作为缓冲区,并将缓冲区首地址传给内核驱动;
- C) 内核驱动申请连续的页面来构造缓冲区控制结构,并把控制结构的物理地址传给打开设备接口函数;
- D) 打开设备接口函数根据缓冲区控制结构的物理地址,通过内存映射得到缓冲区控制结构中的信息;
- E) 应用程序调用接口库提供的报文接口层函数打开设备接口,通过各网卡设备的系统调用命令获得各网卡设备的类型号码,并将记录所有类型号码的设备类型列表记录在一个全局数据结构中,报文接口层根据类型号码对相应网卡设备的缓冲区进行操作;
- F) 在用户空间中为应用新建报文指针队列,将从缓冲区中提取出的报文指针放入报文指针队列,应用程序对报文指针队列中的报文数据进行处理。

2. 根据权利要求 1 所述的零拷贝方法,其特征在于,所述步骤 B 中缓冲区的大小为 2KB,为保证在用户空间申请的缓冲区位于连续的内存页面上,打开设备接口函数申请的内存为共享内存,因为在 linux 操作系统中,共享内存位于连续的物理页面上,具体操作为:

- B1) 打开设备接口函数在用户空间申请共享缓冲区,并连接到共享缓冲区;
- B2) 打开设备接口函数把共享缓冲区的首地址通过 ioctl 系统调用命令传给内核驱动;
- B3) 内核驱动计算缓冲区队列中每个缓冲区的物理地址,并将所述地址保存在一张物理地址表中以便 DMA 使用。

3. 根据权利要求 1 所述的零拷贝方法,其特征在于:所述步骤 C 中的缓冲区控制结构包括缓冲区队列、描述驱动工作模式的正常模式和直接拷贝模式以及用于进程同步及互斥的变量,所述缓冲区队列包括空闲队列和接收缓冲区队列。

4. 根据权利要求 1 所述的零拷贝方法,其特征在于,所述步骤 D 的具体步骤为:

- D1) 打开设备接口函数从系统调用返回的数据结构中获得内核驱动中构造的缓冲区控制结构的物理地址;
- D2) 打开设备接口函数打开内存设备文件;
- D3) 打开设备接口函数通过系统调用把内核中的缓冲区控制结构映射到自己的地址空间。

5. 根据权利要求 1 所述的零拷贝方法,其特征在于,所述步骤 F 的具体步骤为:

- F1) 报文接口层提供的打开设备接口函数在用户空间中新建报文指针队列;
- F2) 报文接口层的报文获取接口函数通过设备类型列表判断各网卡的缓冲区中的结构,从而提取出报文指针,放入报文指针队列中;
- F3) 报文获取接口函数从全局数据结构中得到报文指针队列;
- F4) 当获取报文接口被应用程序调用时,报文接口层函数从接收报文指针队列中取出

一报文，缓冲区接口层继续从填满报文的缓冲区中定位每个报文，以提取出报文首地址指针存入报文接口层的接收报文指针队列；

F5) 当报文被应用程序使用完后即释放报文接口被应用程序调用时，报文接口层函数向释放报文指针队列中插入一报文，缓冲区接口层通过驱动把释放了报文的缓冲区传给相应的网卡设备，等待网卡设备再次向缓冲区中写入报文。

6. 根据权利要求 1 或 5 所述的零拷贝方法，其特征在于：所述全局数据结构是在用户空间申请一块连续的内存所构建的，其成员包括接收报文指针队列和释放报文指针队列。

一种支持多类型网卡链路聚合的零拷贝方法

技术领域

[0001] 本发明涉及一种零拷贝捕包方法,具体地说是一种可支持多类型网卡、实现链路聚合功能的零拷贝方法。

背景技术

[0002] 在高速网络环境中,进行侦听,检测等工作的应用软件需要高效的报文捕获方式,也就是零拷贝捕包方式,在传统的零拷贝捕包软件中,一般采用内核驱动和应用接口库两层结构。内核驱动负责向报文缓冲区中写入报文,应用接口库负责把报文缓冲区提交给应用,当应用软件处理完报文后,会再把缓冲区通过应用接口库释放,以便内核驱动向缓冲区写入新的报文。

[0003] 在传统的零拷贝软件中,当一个应用在一块网卡上捕包时,应用通过零拷贝接口库和内核缓冲区共同操作一个缓冲区队列。但是随着CPU计算能力的增强,现在一台服务器已经有能力处理多块网卡上的流量,所以需要零拷贝软件提供链路聚合功能,把多个网卡的流量聚合成一个数据流提交给应用软件。

[0004] 实现链路聚合的一般方法是,在接口库中建立一系列缓冲队列,对应每个网卡设备有一个队列,当接口库向应用提交报文缓冲区时,接口库对每个网卡进行轮询,并寻找一个缓冲区给应用;当应用释放缓冲区时,接口库根据缓冲区的位置计算缓冲区属于哪个网卡,把缓冲区放入该网卡的空闲队列。

[0005] 随着应用需求的发展,现有技术方案出现了可扩展性不好的缺点,因为很多高速网上的报文处理应用需要特殊的定制化的网卡硬件,在这些定制网卡上,往往要对原始的网络报文做一些预处理,常见的是在原始报文结构前面增强一个额外的数据结构,或者把多个小报文拼合在一个缓冲区内。由于不同的捕包网卡上传的报文结构不同,现有的技术方案中,应用接口库把缓冲区和报文作为一致的数据结构处理,导致一套接口库只能在一种类型的网卡设备上使用,无法解决一个应用使用多种网卡的问题。

发明内容

[0006] 本发明的目的在于解决现有技术方案的缺点,从而提供一种通过将现有技术中的捕包应用接口层划分为缓冲区接口层和报文接口层的分层结构实现对多类型网卡的报文的聚合、对应用提供分流、过滤等附加功能的零拷贝方法。

[0007] 为实现上述发明目的,本发明通过如下技术方案实现:一种支持多类型网卡链路聚合的零拷贝方法,包括如下步骤:

[0008] A) 应用程序调用接口库提供的缓冲区接口层函数打开设备接口,对各网卡指定一个用于区分不同类型网卡的类型号码,将各类型号码记录在设备类型列表中;在各网卡的内核驱动中都增加一个用于返回网卡类型号码的系统调用命令;

[0009] B) 缓冲区接口层提供的打开设备接口函数在用户空间中申请一片连续的内存作为缓冲区,并将缓冲区首地址传给内核驱动;

- [0010] C) 内核驱动申请连续的页面来构造缓冲区队列的控制结构，并把控制结构的物理地址传给缓冲区接口层函数；
- [0011] D) 接口函数根据缓冲区控制结构的地址，通过内存映射得到缓冲区控制结构中的信息；
- [0012] E) 应用程序调用接口库提供的报文接口层函数打开设备接口，通过各网卡设备的系统调用命令获得各网卡设备的类型号码，并将记录所有号码的设备类型列表记录在一个全局数据结构中，报文接口层根据类型号码对相应网卡设备的缓冲区进行操作；
- [0013] F) 在用户空间中为应用新建报文指针队列，将从缓冲区中提取出的报文指针放入报文指针队列，应用程序对报文指针队列中的报文数据进行处理。
- [0014] 所述步骤 B 中缓冲区的大小为 2KB，为保证在用户空间申请的缓冲区位于连续的内存页面上，接口库函数申请的内存为共享内存，因为在 linux 操作系统中，共享内存位于连续的物理页面上，具体操作为：
- [0015] B1) 接口库函数在用户空间中请共享缓冲区，并连接到共享缓冲区；
- [0016] B2) 接口库函数把共享缓冲区的首地址通过 ioctl 系统调用命令传给内核驱动。
- [0017] B3) 内核驱动计算缓冲区队列中每个缓冲区的物理地址，并将所述地址保存在一张物理地址表中以便 DMA 使用；
- [0018] 所述步骤 C 中的缓冲区控制结构包括缓冲区队列，此队列包括空闲队列、接收缓冲区队列、描述驱动工作模式的正常模式和直接拷贝模式，用于进程同步及互斥的变量等。
- [0019] 所述步骤 D 的具体步骤为：
- [0020] D1) 用户接口库函数从系统调用返回的数据结构中获得内核驱动中构造的缓冲区队列控制结构的物理地址；
- [0021] D2) 用户接口库函数打开内存设备文件；
- [0022] D3) 用户接口库函数通过系统调用把内核中的缓冲区控制结构映射到自己的地址空间。
- [0023] 所述步骤 F 的具体步骤为：
- [0024] F1) 报文接口层提供的打开设备接口函数在用户空间中新建报文指针队列；
- [0025] F2) 报文接口库的报文提取函数通过设备类型列表判断各网卡的缓冲区中的结构，从而提取出报文指针，放入报文指针队列中。
- [0026] F3) 报文获取接口函数从全局数据结构中得到报文指针队列；
- [0027] F4) 当获取报文接口被应用程序调用时，报文接口层函数从接收报文指针队列中取出一报文，缓冲区接口层继续从填满报文的缓冲区中定位每个报文，以提取出报文首地址指针存入报文接口层的接收报文指针队列。
- [0028] F5) 当报文被应用程序使用完后即释放报文接口被应用程序调用时，报文接口层函数向释放报文指针队列中插入一报文，缓冲区接口层通过驱动把释放了报文的缓冲区传给相应的网卡设备，等待网卡设备再次向缓冲区中写入报文。
- [0029] 所述全局数据结构是在用户空间申请一块连续的内存所构建的，其成员包括接收报文指针队列和释放报文指针队列。
- [0030] 本发明所带来的有益效果为：
- [0031] 1、本发明通过分层结构，将现有的应用接口层划分为缓冲区接口层和报文接口

层，缓冲区接口层不关心具体设备的缓冲区结构，报文接口层根据设备类型从缓冲区提取报文，这样不同设备的报文数据流可以在报文接口层聚合为一个数据流，实现同一台机器上混合使用多种类型的网卡。

[0032] 2、本发明通过采用分层结构，实现了接口库针对不同设备的代码，只有报文接口层不同，可以方便软件的移植和扩展。

附图说明

[0033] 图 1 是应用接口层的结构示意图，图中虚线框中为现有技术的应用接口层，本发明中将其拆分为缓冲区接口层和报文接口层；

[0034] 图 2 是实现本发明零拷贝方法的流程示意图。

具体实施方式

[0035] 下面结合附图，以两种类型网卡（网卡 A 和网卡 B）为例对本发明作进一步说明。

[0036] 如图 1 所示，本发明通过分层结构，将现有零拷贝技术的捕包应用接口层划分为缓冲区接口层和报文接口层，在缓冲区接口层中只有缓冲区的概念，只针对一个网络设备和一个缓冲区队列进行操作，不再关心缓冲区内的数据结构，缓冲区接口层 A 面向网卡 A，缓冲区接口层 B 面向网卡 B；报文接口层面向应用，使用设备类型列表，对每个设备的缓冲区进行获取和释放操作，把报文结构从缓冲区提取出来，并进行多网卡的报文的聚合。

[0037] 图 2 为实现本发明零拷贝方法的总体流程图。

[0038] A) 应用程序调用接口库提供的缓冲区接口层函数打开设备接口，对各网卡指定一个用于区分不同类型网卡的类型号码，设备类型列表是一个全局数组 dev-type[]，每类设备有一个全局唯一的类型号码，设定网卡 A 的类型号码为 1，网卡 B 的类型号是 2，那么就在设备类型列表中记录 dev_type[A] = 1, dev_type[B] = 2。在网卡 A 和网卡 B 的驱动中均提供 ioctl 接口，ioctl 接口中提供 GET_DEV_TYPE 命令，该命令返回设备的类型号码，用户空间的函数可以通过使用 GET_DEV_TYPE 命令调用网卡 A 和网卡 B 的 ioctl 接口，获得相应网卡设备的类型号码。

[0039] B) 缓冲区接口层提供的打开设备接口函数在用户空间中申请一片连续的内存作为报文缓冲区，并将缓冲区首地址传给内核驱动。在本步骤中，检查系统初始设置的最大共享内存值，若请求内存值大于系统默认共享内存值，则对系统默认内存值进行修改，根据所需的内存大小来分配共享内存，成功返回内存的起始地址后，就可以对内存进行保存数据或锁定操作。在使用完内存后，将此内存归还给系统。

[0040] 缓冲区队列是由一系列指向缓冲区的指针组成的循环队列，内核驱动分别从缓冲区队列 A 和缓冲区队列 B 中取出空闲缓冲区，将网卡 A 的报文传送到缓冲区队列 A 中的空闲缓冲区中，将网卡 B 的报文传送到缓冲区队列 B 中的空闲缓冲区。

[0041] C) 首先内核驱动根据整个缓冲区队列的大小计算需要的缓冲区控制结构的大小，其次内核驱动申请连续的页面来构造缓冲区队列的控制结构，再次内核驱动计算缓冲区队列中每个缓冲区的物理地址，保存在一张物理地址表中以便 DMA 使用，最后内核驱动计算缓冲区队列控制结构的物理地址传给缓冲区接口层函数。

[0042] 缓冲区控制结构的成员包括：缓冲区队列，此队列的结构包括空闲队列、接收缓冲

区队列、描述驱动工作模式的正常模式和直接拷贝模式,用于进程同步及互斥的变量等。控制句柄进行初始化时,将每一块缓冲区的首地址转换成内核空间的物理地址,即 DMA 操作所需的地址,存储在物理地址表中。

[0043] D) 首先,缓冲区接口库接口函数从系统调用返回数据结构中获得内核驱动中构造的缓冲区队列控制结构的物理地址;其次,打开内存设备文件 /dev/mem;最后,通过系统调用 mmap 把内核中的缓冲区控制结构映射到自己的地址空间。

[0044] E) 应用程序调用接口库提供的报文接口层函数打开设备接口,通过各网卡设备的 ioctl 系统调用命令获得各网卡设备的类型号码,并将记录所有号码的设备类型列表记录在一个全局数据结构中,报文接口层根据类型号码对相应网卡设备的缓冲区进行操作。

[0045] 在用户空间申请一块连续的内存作为全局数据结构,其成员包括接收报文指针队列和释放报文指针队列。

[0046] 由报文接口层的函数到网卡设备列表中找到各网卡设备,并调用这些网卡设备的缓冲区接口层的函数打开设备接口,通过各网卡设备的 ioctl 系统调用命令获得各网卡设备的类型号码,将记录所有号码的设备类型列表记录在全局数据结构中。

[0047] F) 首先,报文接口层提供的打开设备接口函数在用户空间的数据结构中新建报文指针队列,所述报文指针队列中的队列元素是指向每个报文的指针,这些报文是从多个不同设备获得的。报文接口库的报文提取函数通过设备类型列表判断网卡 A 和网卡 B 的缓冲区中的结构,从而提取出报文指针,放入报文指针队列中;

[0048] 报文获取接口函数从全局数据结构中获取报文指针队列,当获取报文接口被应用程序调用时,报文接口层函数从接收报文指针队列中取出一报文,应用程序便对此报文进行相应处理,此时缓冲区接口层继续从填满报文的缓冲区中定位每个报文,以提取出报文首地址指针存入报文接口层的接收报文指针队列;当报文被应用程序使用完后即释放报文接口被应用程序调用时,报文接口层函数向释放报文指针队列中插入一报文,此时缓冲区接口层通过驱动把释放了报文的缓冲区传给相应的网卡设备,等待网卡设备再次向缓冲区中写入报文。

[0049] 应用程序会不断从报文指针队列中获取报文指针,此时被提取的报文数据便会展成一个数据流,以实现数据的聚合。

[0050] 实施例 2

[0051] 本实施例的结构和操作步骤基本同与实施例 1,唯有不同之处在于,所述步骤 B 中每个缓冲区的大小为 2KB,为保证在用户空间申请的缓冲区位于连续的内存页面上,接口库函数申请的内存为共享内存,因为在 linux 操作系统中,共享内存位于连续的物理页面上。具体操作为:

[0052] B1) 在接口库函数申请共享缓冲区,并连接到共享缓冲区。

[0053] B2) 接口库函数把共享缓冲区的首地址通过 ioctl 系统调用命令传给内核驱动。

[0054] 最后应当说明的是:以上实施例仅用以说明本发明的技术方案而非对其限制,尽管参照上述实施例对本发明进行了详细的说明,所属领域的普通技术人员应当理解:依然可以对本发明的具体实施方式进行修改或者等同替换,而未脱离本发明精神和范围的任何修改或者等同替换,其均应涵盖在本发明的权利要求范围当中。

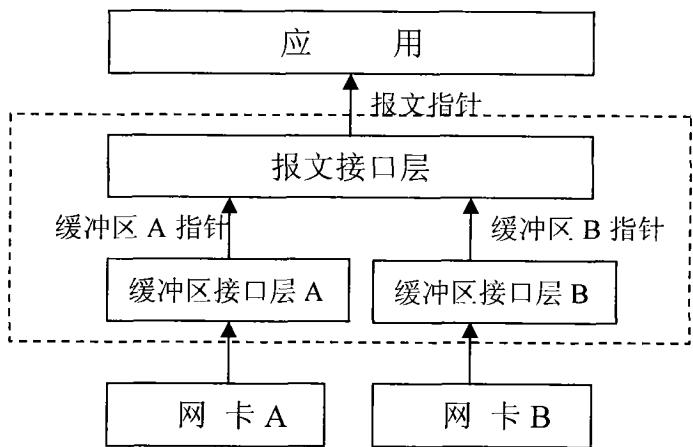


图 1

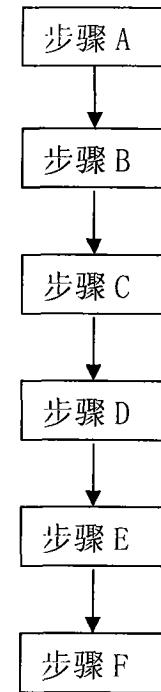


图 2