STREAMS 机制及 IP 协议实现*

胡彦莉 鞠九滨

(吉林大学计算机系,长春 130023)

摘要 STREAMS 是UNIX 操作系统中一种新的字符I/O 机制,利用这种机制实现网络协议具有程序开发容易、可移植性好、易于集成的特点. 本文阐述了关于 STREAMS 的组成和原理的几个基本概念,给出了用STREAMS 实现IP 协议的总体设计,讨论了实现中的重要细节.

关键词 UNIX 操作系统,网络通信协议,网络软件.

AT&T 于1989年推出的 UNIX SVR4. 0把 STREAMS 机制作为所有字符输入/输出的机制,这是较 SVR3. 2的一个重大改进. 用传统的 I/O 机制开发设备驱动程序和网络协议,对于范围较广的各种设备和协议需要各种各样的、特定设计的实现,软件的模块性和可移植性差. 另外,对于现代网络协议的开发与集成,传统的 I/O 机制没有提供一个通用的框架和合适的开发工具,协议软件的可移植性、适用性和可复用性受到很大限制.

STREAMS 是一种灵活、通用的设施和工具,它支持从完整的网络协议包到单独的设备驱动程序的多种服务的实现. 它定义了用于内核中字符输入/输出的标准界面以及内核和 UNIX 系统其余部分之间的标准界面,与此相关的机制简单而且开放[1]. STREAMS 不局限于任何特殊的网络体系结构,为网络协议开发提供了统一的框架. 它的优点主要体现在两个方面:①STREAMS 简化了模块与任意相邻应用程序、模块和驱动程序的服务界面,使得模块之间的接口更加清晰,容易实现. ②STREAMS 具有在用户层灵活操纵模块的能力,它可以动态地连接模块,替换具有公共服务界面的模块.

1 STREAMS 机制

STREAMS 的组成和原理是围绕着几个基本概念展开的,下面简单介绍这几个概念.

1.1 流

"STREAMS"是用于创建、使用和拆除"流"的所有系统调用以及内核资源和内核实用例程的总称. "流"是内核空间中的驱动程序和用户空间的进程之间的一种全双工处理和数据传输的通路[1]. 在内核中,流由流首、驱动程序以及它们之间的零个或多个模块构成(见图1). 数据以消息的形式在流的各组成部分之间传递. 流首提供流与用户进程之间的界面,其

* 本文1994--06--17收到,1994--09--13定稿

本研究是国家"八五"科技攻关项目. 作者胡彦莉,1969年生,助教,主要研究领域为分布/并行计算系统,计算机网络. 鞠九滨,1935年生,教授,主要研究领域为分布/并行计算系统,计算机网络.

本文通讯联系人:胡彦莉,长春130023,吉林大学计算机科学系

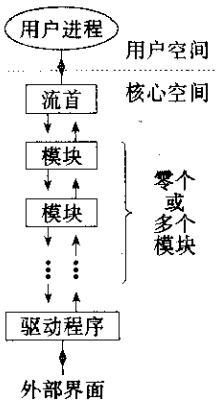

图1

主要功能是处理与 STREAMS 有关的系统调用;模块用于处理在流首和驱动程序之间传递的数据;STREAMS 驱动程序可以是一个外部 I/O 设备的驱动程序,也可以是一个内部的软件驱动程序.

利用 STREAMS 提供的系统调用,用户层程序可以动态地选择和连接 STREAMS 模块,得到任何合理的处理序列.

1.2 队列

图 1 所示的流的每个组成部分都是由一对队列构成的."队列"是一种数据结构,它总是成对分配,一个用于写侧,一个用于读侧.值得注意的是此处的队列不是先进先出意义上的队列,与下面要提到的消息队列也完全是两回事.

队列数据结构的内容可分为 3 部分:①指向若干例程的指针,这些例程是 open 例程、close 例程、put 例程和 service 例程.②指向消息队列的指针.③用于管理流的数据和状态.在核心层,STREAMS 模块或驱动程序由数据结构 Streamtab 定义,系统在分配队列对时,根据 Streamtab 的内容初始化队列的数据和指针.

1.3 消 息

"消息"是一组数据结构,是流内部的通信手段.STREAMS 的消息有不同类型,流的各组成部分对消息的处理是依消息的类型而定的.对消息的处理由 put 和 service 例程完成.put 例程用于在流的各部分之间传递消息,并完成对消息的即时性处理.service 例程可能有,也可能没有,它完成对消息的允许延迟的处理.

1.4 服务界面

利用 STREAMS 机制可以在流的任意两个相邻组成部分之间以及用户进程与流中顶层模块之间实现服务界面."服务界面"是一组原语和规则,它们定义了一种服务和允许的状态转换.服务界面的两边是服务用者和服务供者.原语在服务供者与用者之间双向传递,规则规定了原语传递的允许序列及引起的状态转换.原语由 M_PROTO 和 M_PCPROTO 类型的消息在流成员间传递.

确定兼容的服务界面,是流成员连接、移植、复用的前提.

1.5 多路复用

流的多路复用是指一个流的组成部分在多个流之间切换消息的传递.实现多路复用功能的软件驱动程序被称为多路复用驱动程序.

在多路复用的情况下,对应每一个上层流,STREAMS 分配一个上层读/写队列对,它们具有上层读/写的 put 和 service 过程;对应每一个下层流,STREAMS 分配一个下层读/写队列对,它们具有下层读/写的 put 和 service 过程.STREAMS 并不识别在多路转接器处交汇的多个流之间的关系,消息在不同流间的切换和流量控制都必须由多路复用驱动程序完成.多路复用驱动程序根据消息的类型及其内容来分路消息.

2 用 STREAMS 实现 IP

我们利用 STREAMS 在 UNIX SVR4.0 上实现了 IP 协议.用我们实现的 IP 协议替代原系统中的 IP 协议,与系统的其它协议模块一起运行,可完成系统关于网络的全部功能.

2.1 IP实现的总体设计

IP协议模块向下与多个网络的链路层驱动程序相连,向上与UDP、TCP等多种上层协议相连,因此它被实现为STREAMS多路复用驱动程序,对应每一个上层协议,有一对上层队列,对应每一个下层的链路层驱动程序,有一对下层队列.

所有上层队列的写put过程和写service过程,所有下层队列的读put过程和读service过程是用STREAMS实现IP的最重要的4个过程,它们的功能如下:

上层写put过程处理和传递从上层传来的多种消息,其中最重要的消息处理包括:

①对于M_IOCTL消息:该类消息是控制命令.对于设置IP地址、标志位或要求读取网络配置信息的命令,设置网络接口数据结构的相应字段,或读取相应的接口信息;对于链接或拆除流的命令,分别向链路层协议发汇集请求原语或释放汇集请求原语.在对M_IOCTL消息处理之后,向上回送M_IOCACK或M_IOCNAK消息,带回操作的结果.

②对于M_PCPROTO和M_PROTO消息:这类消息包括IP与上层协议之间的服务界面原语.对于从上层传递来的各类网络层服务界面请求原语,给予相应的回答原语.对于要求IP输出数据的原语消息,将其放入消息队列,等待上层写service过程对其进行处理.

上层写service过程被STREAMS调度程序调度执行,从上层队列的消息队列中取出由上层写put过程放入的消息,完成IP协议规定的对输出数据报的处理,包括填写IP报头,进行路由选择,进行分块,最后写service过程把处理完的IP数据报传到与之相邻的下一队列的put过程[2].

下层读put过程处理和传递从下层传来的多种消息,其中最重要的消息处理包括:

①对于M_IOCACK和M_IOCNAK类型消息:这两类消息是流成员对M_IOCTL消息处理后的回答消息,前者是正确处理后的肯定回答,后者是出错后的否定回答.对于网络信息设置命令的肯定回答,需根据该回答的内容进一步设置网络信息,对于其它的肯定回答消息和所有的否定回答消息,都将其向上传递.

②对于M_PCPROTO和M_PROTO消息:这类消息包括IP与链路层驱动程序之间的服务界面原语.下层读put过程根据链路层服务界面的要求接收并处理链路层传上来的多种原语,并把其中表明接收到IP数据报的原语消息放入消息队列,等待下层读service过程处理.

下层读service过程被STREAMS调度程序调度执行,它从下层读队列的消息队列中取出由下层读put过程放入的消息,完成IP协议规定的对输入数据报的处理,包括对IP报头的检查和解释,重装多个IP报片,最后下层读service过程把处理后的IP数据报送往其指定的上层协议队列[2].

2.2 消息分路

消息在多个流之间分路是多路复用驱动程序必须解决的问题.IP协议对于收到的IP数据报,根据报头中的协议号一项把数据送到相应的上层协议,对于输出的IP数据报,IP根据其目的地址进行路由选择,确定应传向哪一个网络接口.具体的作法是IP驱动程序定义两个数组:

```
struct  ip_pcb  ip_pcb  〔IPCNT〕
struct  ip_provider  provider  〔IPPROVCNT〕
```

每一对上层读/写队列以一个 ip_pcb 数组成员作为私用数据结构，ip_pcb 结构中包括该上层队列对所对应的高层协议的协议号和指向该上层队列对中上层读队列的指针，IP 把输入数据报的协议号与 ip_pcb 所有成员的协议号匹配，由此找到数据应传向的上层协议.

每一对下层读/写队列以一个 provider 数组成员作为私用数据结构，ip_provider 结构包括指向该下层队列对写队列的指针. IP 的路由选择表的每一项都包括指向一个 ip_provider结构的指针. 在路由选择时，IP 根据这一指针找到输出数据应传向的网络接口.

2.3 IP 与上层和下层协议的服务界面

为与原系统的协议接口，我们必须弄清 IP 协议与上下层协议的服务界面. IP 驱动程序与上层协议之间的网络层服务界面的原语及其意义如下：

由网络层服务用者发出的原语：

N_INFO_REQ	数据连接层协议参数
N_BIND_REQ	汇集请求，向服务供者提供自己的协议号
N_UNBIND_REQ	释放汇集请求
N_UNITDATA_REQ	发送数据请求，对应数据结构中含目的地址

由网络层服务供者发出的原语：

N_INFO_ACK	协议信息应答，是对 N_INFO_REQ 的回答
N_BIND_ACK	协议汇集认可，是对 N_BIND_REQ 的肯定答复
N_ERROR_ACK	否定应答，指明前一个原语有错
N_OK_ACK	对前一个原语的肯定应答
N_UNITDATA_IND	接收数据指示
N_UDERROR_IND	接收错误指示

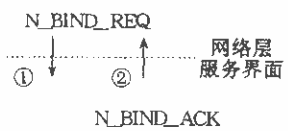

图2

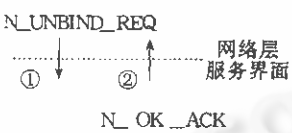

图3

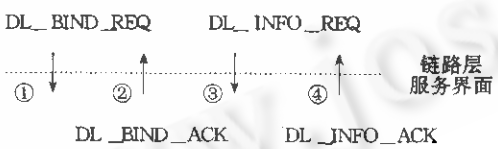

图4

图5

在网络流被连接时，原语以图 2 的顺序通过 IP 驱动程序和上层协议之间的网络层服务界面，在网络流被拆除时，原语以图 3 的顺序通过网络层服务界面. IP 驱动程序接到汇集请求后，把上层协议的协议号赋给 ip_pcb 结构的协议号字段. 在 IP 应答了汇集请求之后和上层协议发出释放汇集请求之前，上层协议可利用 N_UNITDATA_REQ 原语传送要发出的数据，IP 可用 N_UNITDATA_IND 原语向上传送接收到的数据.

IP 驱动程序与网络链路层驱动程序之间的链路层服务界面也有与网络层界面类似的原语，这些原语分别用 DL_代替前述原语名称中的 N_来标识. 在连接网络协议流时，原语以图 4 的顺序通过链路层服务界面，在拆除网络协议流时，原语以图 5 的顺序通过链路层服务界面. IP 驱动程序通过 DL_INFO_ACK 原语得到网络链路层所允许的最大、最小包尺寸

等信息,存入 ip_provider 结构. 在此之后和拆除网络协议流之前,IP 通过发 DL_UNITDATA_REQ 原语把要发出的数据传给链路层驱动程序,链路层驱动程序通过 DL_UNITDATA_IND 把收到的数据传给 IP 驱动程序.

2.4 替换与程序调试

为了检验我们实现的 IP 多路复用驱动程序,我们把它链入系统核心,代替原有的 IP 运行,具体步骤是:

①把/etc/conf/cf.d/mdevice 中 ip 一行改为如下形式:(mux 是我们的驱动程序前缀)

ip　—　Scio　mux　0　26　1　1　—1

②把我们的驱动程序的 space.c 和 Driver.o 文件拷到/etc/conf/pack.d/ip 目录下,覆盖原有的 space.c 和 Driver.o

③用/etc/conf/bin/idbuild 命令重建核心

④用 init 6 重新启动系统

这样我们实现的 IP 就代替原有 IP 在系统中运行了.

核心程序要求可靠性好、效率高,调试核心程序需要反复重建核心,而且缺乏核心调试工具,我们在调试过程中多采用核心打印的办法进行程序跟踪. 另外,在流的各部分之间插入模块,打印流经消息的类型和内容也是跟踪消息流动的好办法,特别是在弄清相邻模块或驱动程序的关系时十分有效.

3 结束语

STREAMS 机制的引入使 UNIX 的国际化和当地化更为容易,掌握 STREAMS 技术将对 UNIX 在我国的应用与发展起促进作用.

参考文献

1　程序员指南:STREAMS. UNIX 系统 V,第 4 版. 北京:电子工业出版社,1992.

2　Douglas E Comer. Internetworking with TCP/IP, 2nd ed. Englewood Cliffs: Prentice Hall, 1991.

STREAMS MECHANISM AND IP IMPLEMENTATION

Hu Yanli　Ju Jiubin

(*Department of Computer Science, Jilin University, Changchun* 130023)

Abstract　STREAMS is a new character I/O mechanism in UNIX systems. Using STREAMS to develop network protocols enables easy development, more portability and easy integration. This paper pays particular attention to some basic concepts concerning the components and the principles of STREAMS. It also provides the overall design of our IP implementation by STREAMS and discusses some important details of this implementation.

Key words　UNIX operating system, network communication protocol, network software.