

支持 ForCEs 的 IP 路由器技术与实现

高明, 王伟明

(浙江工商大学 信息与工程学院, 浙江 杭州 310035)

(gmyyq@hotmail.com)

摘要:支持 ForCEs 的 IP 路由器通过使 CE 和 FE 的分离, 从而实现网络功能的快速配置和重组。提出了一种在 Linux 下 FE Module 的实现机制, 在此基础上构造出一个支持 ForCEs 的 IP 路由器进行测试。结果表明, 通过这种机制实现的 ForCEs 路由器能很好地满足 ForCEs 的需求。

关键词:IP QoS 路由器; 转发和控制分离; 逻辑功能块; FE 模型; Linux 模块

中图分类号:TP393.05 **文献标识码:**A

Research and implementation of IP router that supports ForCEs

GAO Ming, WANG Wei-ming

(Department of Information and Electronic Engineering, Zhejiang Gongshang University, Hangzhou Zhejiang 310035, China)

Abstract: By separating CE from FE, ForCEs IP routers can easily configure out flexible and intelligent network. A mechanism for which FE Module can be implemented in Linux was brought forward. Based on the mechanism, A ForCEs IP router was constructed and corresponding result was given after testing. ForCEs IP routers implemented by the mechanism can meet all the requirements of ForCEs achitecture well.

Key words: IP QoS router; ForCEs; LFB, FE model; Linux module

0 引言

转发和控制分离 (Forwarding and Control Separation, ForCEs) 是 IETF 路由领域的一个工作组, 它专门研究开放编程的 IP 路由器的体系结构和协议问题, 是当前开放可编程网络研究最受关注的研究组织。ForCEs 基本思想是把 IP 路由器分成转发件 (Forwarding Elements, FE) 和控制件 (Control Elements, CE), 认为 IP 路由器可由多个 (可达几百个) FE、多个 CE 和连接他们的 ForCEs 协议构成。当前 IETF ForCEs 工作组已经完成了 ForCEs 需求 (ForCEs Requirements, RFC3654)^[2] 和基本完成了 ForCEs 框架 (ForCEs Framework)^[3] 工作, 当前的重点是 ForCEs 协议和 ForCEs FE 模型。

1 ForCEs 体系结构和模型

ForCEs 体系结构和模型主要在 ForCEs Requirements 和

ForCEs Framework 中定义。一个满足 ForCEs 标准的网络组件 (Network Element, NE) 的核心具有如图 1 所示结构。

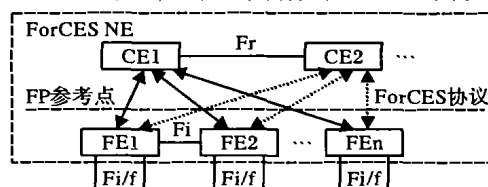


图1 ForCEs 基本结构

图1中, 一个 ForCEs NE 内有至少一个或多个冗余用的控制件 CE、有可多达几百个的转发件 FE, 它们间的联系通过 ForCEs 协议完成, 这个连接面称为 FP 参考点。Fi/f 为各个 FE 的网络接口参考点; Fr 为冗余 CE 连接参考点, 当前 ForCEs 暂不考虑该点的技术问题。

FE 内的体系结构主要由 FE 模型 (FE Model) 定义, FE 基本结构如图2所示。

其中 FE 内的资源被表示成各种不同功能的逻辑功能块

收稿日期: 2004-09-07; 修订日期: 2004-11-10 基金项目: 国家 863 计划资助项目 (2002AA121064); 国家自然科学基金资助项目 (60273061); 浙江省自然科学基金人才专项资助项目 (RC02063)

作者简介: 高明 (1979-), 男, 安徽淮南人, 硕士研究生, 主要研究方向: 网络通信; 王伟明 (1964-), 男, 浙江遂昌人, 教授, 博士, 主要研究方向: 网络通信。

参考文献:

- [1] ISO/IEC 9646-2. Information Processing Systems, Open System Interconnection, OSI Conformance Testing Methodology and Framework, Part 2: Abstract Test Suite Specification[S]. 1994.
- [2] TAHI IPv6 conformance test specification[EB/OL]. <http://www.tahi.org/conformance/doc/ct-document/>, 2000-07.
- [3] InterOperability Laboratory, Research Computing Center, University of New Hampshire. IPv6 Consortium Test Suite-Internet Protocol version 6, version 1.0[Z]. 2002.
- [4] 尹霞, 吴建平. 一种路由协议测试框架研究[J]. 清华大学学报 (自然科学版), 2001, 41(1): 81-84.
- [5] 杨晶, 屈玉贵, 赵保华. 路由协议一致性测试的系统设计[J]. 计算机应用, 2001, 21(5).
- [6] 杨晶, 赵保华, 屈玉贵. 基于层次结构的 OSPF 一致性测试[J]. 通信学报, 2002, 23(8).
- [7] OUSTERHOUT JK. Tcl and the Tk Toolkit[M]. Addison Wesley Professional, 1994.
- [8] COLTUN R, FERGUSON D, MOY J. OSPF for IPv6, RFC2740[S]. 1999.
- [9] 龚正虎. 计算机网络协议工程[M]. 长沙: 国防科技大学出版社, 1993.

(Logical Functional Block, LFB)。LFB 及它的属性都是可以由 CE 通过 ForCEs 协议进行控制的,各个 LFB 之间通过数据通道相互连接,该连接关系也是由 CE 经过 ForCEs 协议定义,以形成不同的 LFB 拓扑结构,进而实现动态资源配置以完成各种不同的 IP 类型服务。典型的 LFB 如分类器、调度器、IPv4 或 IPv6 转发器等。

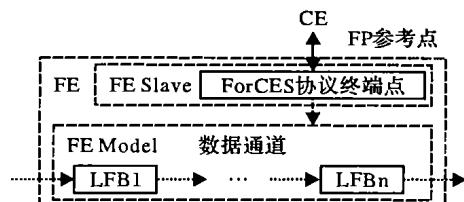


图2 ForCEs FE 基本结构

2 Linux 操作系统中 FE Model 实现机制分析

2.1 选择 Linux 作为实验开发平台

Linux 操作系统具有开放性,把它作为实验开发平台是一个不错的选择,因此选择 Linux 操作系统来实现我们的 FE。

FE Slave 部分主要负责 ForCEs 协议的解析,并执行用以控制 FE Model 和伺服策略的建立。由于本文重点在于 FE Model,所以有关 FE Slave 的内容,就不再多做介绍。

在 ForCEs 中,FE Model 被表示成各种不同功能的 LFB,每个 LFB 都有自己的拓扑结构和属性,并且这些 LFB 都是动态可加载、删除或修改的。根据 ForCEs Requirements^[7] 和 ForCEs Framework^[8] 中的定义,我们知道一个支持 ForCEs 的 IP Qos 路由器的 FE 应该有分类 LFB、调度 LFB、转发 LFB 等组成。因为我们选择 Linux 操作系统作为实现 FE Model 的平台(Linux 内核版本号为 2.4.20),所以在 Linux 操作系统中选择什么方式来实现 FE Model,就变得格外重要。通过对 Linux 操作系统研究发现,Linux 系统有一种模块机制能很好地满足我们的要求。

2.2 Linux 的模块机制

Linux 操作系统采用的是微内核体系,它的结构特点是,在操作系统的核心部分有一个很小的内核,实现一些操作系统最基本的服务,如创建和删除、内存管理、中断管理等,而其他的如文件系统、网络协议等部分则都在微内核的外部空间运行。但是,在以往这样一个微内核的操作系统,会有一个问题:那就是当系统程序员希望给 Linux 内核增加新功能时,他们是必须编写新代码然后静态地链接到内核中,这就需要重新编译内核,相当麻烦。

于是,Linux 的模块机制应运而生。在内核中实现一些基本的功能,如从模块到内核的接口,内核管理所有模块的方式等,而系统的可扩展性就留给了模块来完成。进行模块编程的同时就是进行内核编程,因为它的运行环境是内核环境,它的程序运行函数库都在内核空间定义,而不是在用户函数库空间。使用模块能让操作系统内核更加紧凑。当需要这个模块的功能的时候,将这个模块加载;不需要的时候就把它从内核中卸载掉。

显然,如果把 FE Model 中的各个 LFB 做成 Linux 模块的话,借助 Linux 提供的模块机制就很容易实现 FE Model。

2.2 Linux 模块以及 FE Model 的实现

Linux 模块在文件系统中是以 ELF 对象文件存储的。Linux 内核只会考虑那些由/sbin/insmod 程序装载到 RAM 中的模块,对于每个模块,系统都分配一个包含以下数据的内存

区:一个 module 对象;表示模块名的一个以 NULL 结束的字符串;实现模块功能的代码。

下面以一个最简单的模块为例来说明 Linux 模块的实现。

```
#define MODULE
#define __KERNEL__
#include <linux/module.h>
void function( )
{ printk( "hello, world\n"); }
int init_module( void)
{ return 0; }
void cleanup_module( void)
{ printk( "goodbye, world\n"); }
```

以上是一个 C 程序代码文件(hello.c),在程序的开头通过#define MODULE 和#define __KERNEL__来声明这是一个运行在内核空间的模块程序。函数 printk 在 Linux 内核中定义,其功能和标准 C 库中的函数 printf 类似。

```
root# gcc -c hello.c
root# insmod hello.o
hello, world
root# rmmod hello
goodbye, world
root#
```

首先把 C 文件编译成以 .o 结尾的目标文件,即为模块,然后用 insmod 装入模块,这样模块就被链接到内核,并执行函数 function(),从而打印字符串“hello, world”,当用 rmmod 卸载模块时,就会打印字符串“goodbye, world”。

要实现分类 LFB、调度 LFB、转发 LFB,只要把上述程序中函数 function()换成相应的 LFB 处理函数即可。

下而列出了一些 LFB 对应的 Linux 模块:1)分类 LFB→Classifier;2)调度 LFB→Scheduler;3)转发 LFB→Forward。

上而对模块的添加、删除都是在终端(root#)中完成的,需要用户实时地敲入指令。然而根据 ForCEs 的要求,FE Model 的管理实际上是由 CE 通过 ForCEs 协议与 FE 中的 FE Slave 通信,然后 FE Slave 根据 CE 传送过来的消息管理 FE Model,这也就是 FE Model 的管理需要在 FE Slave 中编程实现的,而不是通过实时地敲入指令的方式。换个角度来说,就是如何在程序中调用 insmod、rmmod 等指令。经过研究发现,Linux 内核提供了一个函数 call_usermodehelper(),使用方法如下:

```
argv[0] = "/home/~classifier.o"; //表示当前模块所在的位置
envp = "PATH=/sbin/insmod"; //表示当前指令的所在的目录
call_usermodehelper( "insmod", argv[0], envp);
//执行装载模块操作
call_usermodehelper( "rmmod", argv[0], envp);
//执行卸载模块操作
```

由于 FE Model 是由各种 LFB 组成的,所以首先编写好分类 LFB、调度 LFB、转发 LFB 等对应的 C 程序文件,然后编译成 Linux 模块,存放在系统的某个位置。这样当 CE 向 FE 发送 FE Model 控制消息时,FE Slave 会根据消息管理这些 LFB,从而形成了 FE Model。例如:CE 向 FE 发送消息,要求添加分类 LFB 时,FE Slave 就会通过函数 call_usermodehelper()来装载分类 LFB 对应的 Linux 模块。其他的情况,依次类推。

3 FE Model 的测试

3.1 测试目的

目前测试 FE Model 的目的主要是:1)检验以上介绍的

FE Model 实现机制的可行性;2)构造一个支持 ForCEs 的 IP Qos 路由器的基本框架并测试其工作情况。因此设计了以下的实验场景。

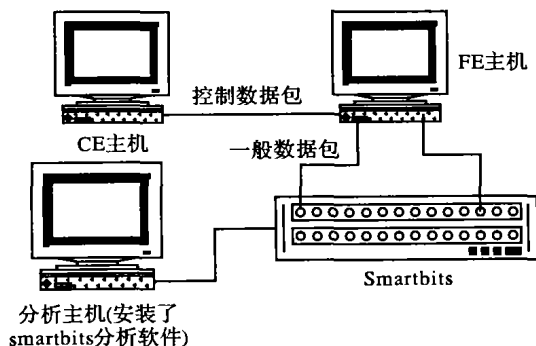


图3 硬件布置图

3.2 硬件布置图

当前设置了一个 FE 主机和一个 CE 主机。在 CE 主机上的 CE 模块起到控制 FE 主机上 FE 模块的作用。根据 ForCEs 协议 CE 和 FE 之间的通信以 TCP socket 的方式实现, CE 模块在 Windows 2000 上用 VC++ 实现, CE 模块运行时, 用户可以在线设置各类发送消息及其相关参数到各个 FE。FE 负责和 CE 进行通信, 并根据 CE 发送过来的消息构造 FE Model, 从而形成一个支持 ForCEs 的 IP Qos 路由器。Smartbits 网络测试仪被用来产生各种速率各种类型的数据包, 并且对接收到的数据包进行统计分析。

3.3 测试情景描述

在 CE 端按照图 4 对 FE 中的 FE Model 进行配置。测试中, Smartbits 向 FE 发送两类数据包: 一类目的地址为 192.168.2.5; 一类目的地址为 192.168.7.5。经过分类器时, 这两类将被分开, 第一类包将被放入流标识号 (packet flow ID, pkfID) 为 1 的队列中, 第二类包放入 pkfID 为 2 的队列中。分类后的两个数据包流进入调度器, 调度器根据调度算法从 pkfID 为 1 和 2 的队列中取数据, 然后重新排队, 并把重新排队的数据包放入 pkfID 为 3 的队列中。从调度器出来的数据包进入转发器, 转发器从 pkfID 为 3 的队列取数据发送出去。

Smartbits 上发送和接收的数据包率可以显示在和 smartbits 相连的分析主机上。Smartbits 发出的数据包, 经过 FE 的处理又回到了 Smartbits, 这时 Smartbits 分析软件将对收到数据包进行统计分析, 从而得到测试结果。

需要说明的是, 由于本文重点在 FE Model 实现机制分析, 所以有关调度算法这里没有多做介绍。下面路由器的测试数据是采用最常用的先到先服务 (first come first serve, FCFS) 调度算法得到的结果。



图4 测试 FE Model 的拓扑结构

3.4 测试结果

1) 上述 FE Model 实现机制可行性

在 CE 端运行时, 用户在线对 FE 中的 FE Model 进行配置。然后我们在 Linux 终端中, 敲入如下指令:

```
root# lsmod //查看当前运行的 Linux 模块
屏幕上有如下的显示:
Module      size  used by    Not tainted
Classifier  4094    0          (autoclean)
```

```
Scheduler  8096    0          (autoclean)
Forward    10906   0          (autoclean)
```

这说明 FE Model 中的各个 LFB 已经以 Linux 模块的形式存在于系统中了。

2) 路由器的测试数据

通过前面的操作, 构造了一个路由器。下面就通过数据来看这样一个路由器的工作情况。图 5 给出了测试结果。

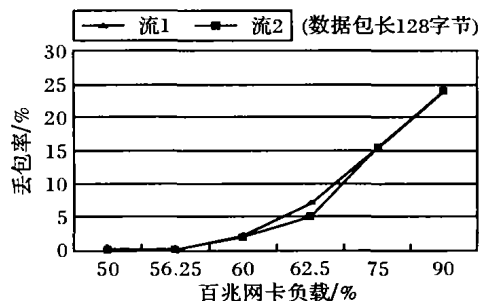


图5 采用 FCFS 调度算法时丢包率

测试中发送数据源是一个百兆的网卡, 它发出速率相同的两个流: 流 1 和流 2。流 1 就是我们前面所说的目的地址为 192.168.2.5 的流, 流 2 就是目的地址为 192.168.7.5 的流。由于采用的是 FCFS 调度算法, 所以流 1 和流 2 应该被同等对待。从图 5 中我们发现, 路由器在百兆网卡负载为 56.25% 左右是一个临界状态, 一旦超出了这个值就会丢包, 超出越多, 丢包率越大。从流 1 和流 2 的各项统计数据来看, 两者基本相同, 充分说明了流 1 和流 2 被同等对待。

4 结语

本文所介绍的 FE Model 实现机制一定的针对性。由于 Linux 操作系统本身的开放性, 对于那些从事路由器研究的人员来说, 在研究实验阶段选择它作为开发平台, 能够带来极大的方便并且成本低廉。

参考文献:

- [1] BISWAS J, LAZAR AA, HUARD JF, et al. The IEEE P1520 Standards Initiative for Programmable Network Interfaces[J]. IEEE Communications Magazine, 1998, 36(10): 64-72.
- [2] KHOSRAVI H, ANDERSON T. Requirements for Separation of IP Control and Forwarding [EB/OL]. <http://www.ietf.org/rfc/rfc3654.txt?number=3654>, 2003-11.
- [3] YANG L, DANTU R, ANDERSON T, et al. Forwarding and Control Element Separation (ForCEs) Framework [EB/OL]. <http://www.ietf.org/internet-drafts/draft-ietf-forces-framework-13.txt>, 2003-12.
- [4] Internet-Draft, General Router Management Protocol (GRMP) Version 1 [EB/OL]. <http://www.ietf.org/internet-drafts/draft-wang-forces-grmp-01.txt>, 2003-11.
- [5] PUTZOLU D. ForCEs Protocol Evaluation Draft [EB/OL]. <http://www.ietf.org/internet-drafts/draft-ietf-forces-evaluation-00.txt>, 2003-12.
- [6] YANG L. ForCEs Forwarding Element Model [EB/OL]. <http://www.ietf.org/internet-drafts/draft-ietf-forces-model-01.txt>, 2003-10.
- [7] China Linux Forum [EB/OL]. <http://www.linuxforum.net/forum/postlist.php?Cat=&Board=linuxK>, 2004.
- [8] 李善平. Linux 内核 2.4 源代码分析大全 [M]. 北京: 机械工业出版社, 2002.
- [9] 陈莉君, 译. 深入 Linux 内核 [M]. 北京: 中国电力出版社, 2001.