IPv6下DiffServ在Linux中的设计与实现

邱述威,张 霖
(安徽建筑工业学院 网络信息中心,安徽 合肥 230022)

摘 要:QoS的研究目标是有效地为用户提供端到端的服务质量控制或保证,而IPv6不仅有效地解决了网络地址危机的问题,而且为提升网络服务质量(QoS)提供了更好的支持。剖析区分服务模型的数据流分类和PHB实现预定义标识流,同时分析Linux内核中的流控制机制。在Linux平台上利用流量控制工具集TC,对物理网络设备绑定CBQ队列、在队列上建立分类、为每个分类建立路由的过滤器,从而实现不同类数据流的区分和带宽保证。并对现有的队列、分类、过滤器和路由进行监视。实现表明,测试平台运行稳定,可以为QoS研究提供一个开发式的测试环境。

关键词:IPv6;QoS;区分服务;流量控制
中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2008)05-0238-03

Design and Implementation of DiffServ on IPv6 Protocol under Linux

QIU Shu-wei, ZHANG Lin
(Network Information Center, Anhui Institute of Architecture & Industry, Hefei 230022, China)

Abstract: The goal of QoS is effective for users with end-to-end service quality control or assurances, and IPv6 is not only an effective solution to the network address of the crisis, but also for improving network quality of service (QoS) to provide better support. Analyzes the data flow classification of differentiated service model and identification PHB achieving predefined flow, and analysis of the Linux kernel in the flow control mechanisms. On the Linux platform using tool-set TC, the physical network equipment bundled CBQ queue, the queue in a classification, a classification for each routing filters, a different type of data flow and bandwidth distinction. And exist in the queue, classify, filter and routing surveillance. Achieve that stability testing platforms can provide QoS for the development of a test environment.

Key words: IPv6; QoS; DiffServ; TC

0 引 言

IPv6中的QoS提供区分服务(DiffServ),它根据Internet上通信的需求不同而区分服务类型。路由器把通过的数据流分成若干类,根据不同类型的数据流,采取不同的策略。IPv6协议中,它使用IPv6头部的Traffic Class(TC)字段,并把它作为DS模型的专用字段[1]。DS (Differentiated Service)字段的起始六位,被用作区分服务标记(DiffServ Code Point, DSCP)。路由节点根据DSCP选择提供特定质量的调度转发服务,这种特性称为逐跳行为(Per Hop Behavior, PHB)[2]。

由于Linux操作系统的普及,研究在Linux上的QoS保障变得十分有意义。目前,Linux操作系统的内核已经为QoS提供了良好的支持。较为常用的是一个流量控制器(TC, Traffic Control),TC以命令行的方式来设置QoS参数来实现QoS的保障,文中就此阐述了如何利用流量控制器在基于IPv6的情况下来实现DiffServ。

1 基于流标识的QoS控制

1.1 数据流分类技术

在DiffServ中,边缘路由器中的分类器对进入DS区域的数据流进行分类[3]。目前已经定义了两种分类器:行为聚集(Behavior Aggregate, BA)分类器和多域(Multi Field, MF)分类器。BA分类器使用IPv6基本头部中的流量类型TC字段来设置区分服务标DSCP,以此进行分类;MF分类器根据数据报头部中多个域内容的组合进行分类。例如:源地址目标地址、协议标识、源端口号、目标端口号,通常称它们为MF五元组分类器,五元组的一些信息需要从传输层或应用层协

收稿日期:2007-08-19
基金项目:安徽省自然科学资助项目(2005KJ077);2004年安徽建筑工业学院硕博资助项目(2004071)
作者简介:邱述威(1975-),男,实验师,硕士,研究方向为下一代互联网技术及QoS。

议得到。若把IPv6的流标识用于DiffServ MF分类器,只需要访问IPv6数据报的头部信息即可,不再需要访问传输层和应用层的信息,所以使用流标识有助于IPv6 QoS路由器取得更高的数据报处理效率。

但MF五元组分类器存在一定的局限性。根据数据报分段原理,IPv6数据报仅在第一个分段承载端口号,所以查看端口号的MF分类器很可能遗漏同一个数据报的后续段,从而会引起处理上的错误。使用流标识后,即使对IPv6数据报进行了分段,但是每个分段中都携带有同样的流标识,在传输过程中,这些分段会得到同样的QoS保障,从而避免了上述由分段引起的问题[4]。

另外使用流标识分类有利于端对端的IP级安全机制,因为使用流标识分类不再依赖于高层协议,可以使用IPSec对IPv6数据报进行加密处理。

1.2 预定义的标识流

在DiffServ中,流标识的值对于DS字段的入口路由器来说是已知的。所以要求IPv6流标识的值是一个非随机值,是预定义的或可预选的。流标识值的选择依赖于网络客户和网络服务提供者之间的协商,即依赖于服务等级协定SLAs、业务等级协定TCAs、服务等级说明SLSs和业务调整说明TCSs[5],流标识值要能够反映出这些协商结果。离开客户网络的数据报由源节点或第一跳路由器分配一个正确的值,携带流标识值的数据报到达提供者网络时,路由器会基于流标识对这些数据报进行处理。

为了保持与IPv6中定义的流标识值是一个随机数相兼容,并且兼顾DiffServ对流标识的要求,一种可能的流标识的格式定义如图1所示。DiffServ中的IPv6流标识是一个基于16位的每跳行为(Per Hop Behavior)标识码PHB ID。PHB ID可以直接从一个标准区分服务代码点DSCP获得,也可以由因特网号码指派管理局IANA(Internet Assigned Numbers Authority)分配一个值。

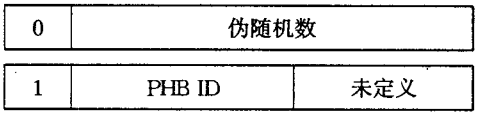

图1　IPv6流标识格式的定义

2 Linux的流量控制

Linux内核很早就支持IPv6,从2.2版本以后加入了支持QoS的流量控制框架,实现了多种缓冲管理机制,为实现区分服务提供了很大的方便。

在Linux内核中,流量的控制主要由排队策略、包类别和包过滤器三个部分组成[6]。每个网络设备对应着一个排队的策略,用来指定该网络设备中的数据包的处理方式。包的类别和过滤器与排队策略紧密相关。过滤器根据队列中取出的数据包的属性将其分类,放入相应的队列中,每个类别都有一个队列与其对应,并包含相应的排队策略[7]。这三者之间的构成如图2所示。

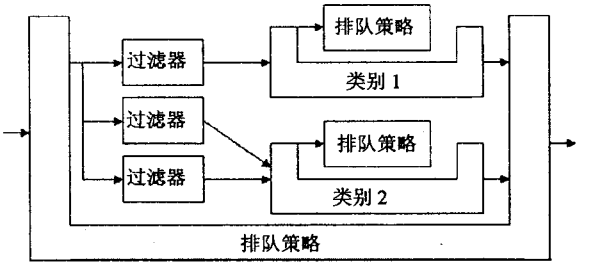

图2　流量控制的三个组成部分的构成

2.1 排队策略

每个网络设备都对应一个排队的策略,用来指定该网络设备队列中的数据包的处理方式。目前Linux的内核支持的排队策略有:

(1)基于类的队列(CBQ - Class Based Queuing):优先队列的变体,为每个输出队列指定优先级和允许的最大流量,通过不同优先队列的时间片轮转来输出数据。它的优点在于为不同类型的数据提供了一定的公平性。文中后面的实现部分就是基于CBQ实现的。

(2)令牌桶过滤(TBF - Token Bucket Filter):用令牌的数量限制队列中的数据所能获得的服务。它能够限制高优先级的业务量的速率,避免低优先级的饥饿,还具有对突发业务量的整形作用。

(3)随机早期检测(RED - Random Early Detection):RED算法监视队列的长度,一旦队列将要满了,它就随机地选择TCP流丢弃数据包,从而使发送方减慢发送的速度,来避免一些不合时宜的拥塞。

2.2 包的类别

每个类别都有一个属于它的队列,并包含相应的排队策略。当一个数据包到达一个队列时,排队策略首先调用包过滤器来确认包的类别,然后将其放入与之相应的队列。

2.3 过滤器

过滤器又叫分类器,可以根据数据包的属性对其进行分类。TC可以使用的分类器有:fwmark分类器、U32分类器、基于路由的分类器和RSVP分类器[8](分别用于IPv6和IPv4)等。

3 基于IPv6的DiffServ在Linux中的实现

首先是对Linux的内核进行配置和编译,使其支持IPv6和QoS,在Linux系统中,内核存放在/usr/src

文件夹中,进入到系统内核中,再输入 make menuconfig 命令进入系统内核的配置界面,进入内核配置界面后将支持 IPv6 和 DiffServ 的模块设为 Y(编入内核),然后保存即配置完成。内核配置完成后,就要对新内核进行编译。

内核编译完成后,在 Linux 终端中执行命令 modprobe ipv6 既加载了 IPv6 的模块,此时 Linux 服务器和与其相连的两台主机同时分配了处在同一网段内的 IPv6 地址,通过配置和使用流量控制器 TC 在服务器上实现区分服务了。分为以下方面:建立队列、建立分类、建立过滤器和建立路由,另外还需要对现有的队列、分类、过滤器和路由进行监视。其基本步骤为:

1)针对物理网络设备绑定一个 CBQ 队列;

2)在该队列上建立分类;

3)为每一个分类建立一个基于路由的过滤器;

4)最后与过滤器相配合,建立特定的路由表。

网络结构图如图 3 所示。

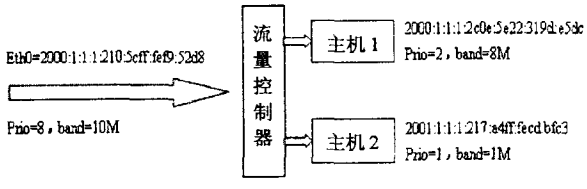

图 3 网络结构图

在服务器和与其相连的两台主机组成的网络环境中,流量控制器上的以太网卡(eth0)的 IPv6 地址为 2000:1:1:1:210:5cff:fef9:52d8,在其上建立一个 CBQ 队列,设包的大小为 1000 字节,包间隔发送单元的大小为 8 字节,可接受冲突的发送最长包数目为 20 字节。

有以下两种流量需要控制:

(i)是发往主机 1 的,其 IPv6 地址为 2000:1:1:1:2c0e:5e22:319d:e5dc,其流量控制在 8Mbit,优先级为 2;

(ii)是发往主机 2 的,其 IPv6 地址为 2000:1:1:1:217:a4ff:fecd:bfc3,其流量控制在 1Mbit,优先级为 1。

一般情况下,针对一个网卡只需建立一个队列。将一个 cbq 队列绑定到网络物理设备 eth0 上,其编号为 1:0;网络物理设备 eth0 的实际带宽为 10Mbit,包的平均大小为 1000 字节;包间隔发送单元的大小为 8 字节,最小传输包大小为 64 字节。

tc qdisc add dev eth0 root handle 1:0 cbq bandwidth 10Mbit avpkt 1000 cell 8 mpu 64

(1)创建根分类 1:1 分配带宽为 10Mbit,优先级别为 8。

tc class add dev eth0 parent 1:0 classid 1:1 cbq bandwidth 10Mbit rate 10Mbit maxburst 20 allot 1514 prio 8 avpkt 1000 cell 8 weight 1Mbit

该队列的最大可用带宽为 10Mbit,实际分配的带宽为 10Mbit,可接收冲突的发送最长包数目为 20 字节;最大传输单元加 MAC 头的大小为 1514 字节,优先级别为 8,包的平均大小为 1000 字节,包间隔发送单元的大小为 8 字节,相应于实际带宽的加权速率为 1Mbit。

(2)创建分类 1:2,其父分类为 1:1,分配带宽为 8Mbit,优先级别为 2。

tc class add dev eth0 parent 1:1 classid 1:2 cbq bandwidth 10Mbit rate 8Mbit maxburst 20 allot 1514 prio 2 avpkt 1000 cell 8 weight 800kbit split 1:0 bounded

该队列的最大可用带宽为 10Mbit,实际分配的带宽为 8Mbit,可接收冲突的发送最长包数目为 20 字节;最大传输单元加 MAC 头的大小为 1514 字节,优先级别为 2,包的平均大小为 1000 字节,包间隔发送单元的大小为 8 字节,相应于实际带宽的加权速率为 800kbit,分类的分离点为 1:0,且不可借用未使用带宽。

(3)创建分类 1:3,其父分类为 1:1,分配带宽为 1Mbit,优先级别为 1。

tc class add dev eth0 parent 1:1 classid 1:3 cbq bandwidth 10Mbit rate 1Mbit maxburst 20 allot 1514 prio 1 avpkt 1000 cell 8 weight 100kbit split 1:0

(4)建立过滤器。过滤器主要服务于分类。一般只需针对根分类提供一个过滤器,然后为每个子分类提供路由映射。

应用路由分类器到 cbq 队列的根,父分类编号为 1:0,过滤协议为 ip,优先级别 100,过滤器为基于路由表。

tc filter add dev eth0 parent 1:0 protocol ip prio 100 route

建立路由映射分类 1:2、1:3。

tc filter add dev eth0 parent 1:0 protocol ip prio 100 route to 2 flowid 1:2

tc filter add dev eth0 parent 1:0 protocol ip prio 100 route to 3 flowid 1:3

(5)建立路由,该路由是与前面所建立的路由映射一一对应。

发往主机 2000:1:1:1:2c0e:5e22:319d:e5dc 的数据报通过分类 2 转发(分类 2 的速 8Mbit)

ip route add 2000:1:1:1:2c0e:5e22:319d:e5dc dev eth0 via 2000:1:1:1:210:5cff:fef9:52d8 realm 2

(下转第 244 页)

```
    INT8U stops;
    INT8U parity;
    INT8U mode;
    }COMM_SETUP, * PCOMM_SETUP;
typedef struct{
    INT8U management;
    BOOLEAN asc;
    BOOLEAN bin;
    BOOLEAN esc;
    BOOLEAN crc16;
    }Cmd_Option, * pCmd_Option;
```

其中:management = 0 不覆盖文件;management = 1 无条件覆盖文件;management = 2 追加在目标文件后;management = 4 源文件新则覆盖目标文件。

(3) BOOLEAN ReceiveFiles(RS232 com, Cmd_Options op, INT16U * err)

功能:使用 ZMODEM 协议接收文件。

参数:com 串口参数;op 命令行参数;err 错误参数。

返回值:TRUE 成功,ALSE 失败。

3 结 语

实现了在 μC/OS-Ⅱ环境下,两台 PC 机之间基于 ZMODEM 协议的文件传输,其传输速度快、可靠性高、断点能够续传。从而说明 ZMODEM 协议在文件传输时是一个不错的选择。采用模块化设计方法,使 ZMODEM 协议化繁为简,更便于理解与掌握。采用分层实现方法,使代码简洁、清晰,且可以只修改驱动程序而使此系统用于不同场合的文件传输。而在本系统中与操作系统与硬件相关的代码大部分放在 PC.C 文件中,便于此系统以后的移植。

参考文献:

[1] Forsberg C. The ZMODEM Inter Application File Transfer Protocol[M]. [s.l.]:Omen Technology Inc,1988.

[2] 岳晓庆,张其善. 串口扩口技术在嵌入式系统中的实现[J]. 电子测量技术,2006,29(2):45-46.

[3] 许春冬,陈良军. 嵌入式数字视频监控系统中串口通信的设计与实现[J]. 电子科技,2005(11):61-63.

[4] 刘燕军,蒋存波,陈占海. 嵌入式系统与 IPC 的一种串口通讯协议及其实现[J]. 桂林工学院学报,2006,26(4):579-582.

[5] 赵世湖,周 辉. 数字成像嵌入式 DSP 系统与 PC 间的串行通信[J]. 影像技术,2007(2):26-28.

[6] 贾瑞玉,赖大荣. 车流量测量仪串口通信的设计与实现[J]. 计算机技术与发展,2006,16(10):199-201.

[7] Nelson M. 串行通信开发指南[M]. 潇湘工作室译. 北京:中国水利水电出版社,2001.

[8] 王荣良. 微机原理与接口技术[M]. 北京:高等教育出版社,2005.

(上接第 240 页)

发往主机 2000:1:1:1:217:a4ff:fecd:bfc3 的数据报通过分类 3 转发(分类 3 的速 1Mbit)

```
# ip route add 2000:1:1:1:217:a4ff:fecd:bfc3
dev eth0 via 2000:1:1:1:210:5cff:fef9:52d8 realm 3
```

经过以上步骤,就能够对通过网卡 eth0 的流量进行控制和管理,同时还可以用命令对包括现有队列、分类和过滤器的状况进行监视,看看控制是否达到预期的效果,可用的命令有:

* tc qdisc ls dev:显示队列的状况
* tc class ls dev:显示分类的状况
* tc - s filter ls dev:显示过滤器的状况

4 结 语

文章针对区分服务流分类和如何利用 Linux 的流量控制策略,根据队列、分类、过滤器和路由来设计并实现对不同数据流的区分和带宽保证,提高了传统网络的服务质量。为了让 DiffServ 机制更完善,还有很多工作需继续进行,如在流量巨大的网络中,如何寻找到一种最合理的排队机制、最有效的分配资源,是需要进一步努力的,同时也需要更大量的实践和仿真。

参考文献:

[1] Deering S, Hinden R. Internet Protocol, Version 6(IPv6) Specification[S/OL]. RFC 2460. 1998-12. http://www.faqs.org/rfcs/rfc2460.html.

[2] Jacobson V, Nichols K, Poduri K. An Expedited Forwarding PHB[S/OL]. IETF RFC 2598. 1999-06. http://www.ietf.org/rfc/rfc2598.txt.

[3] Blake S, Black D, Carlson M, et al. An Architecture for Differentiated Services[S/OL]. RFC 2475. 1998. http://www.ietf.org/rfc/rfc2475.txt.

[4] 吴建平,盛立杰,林 闯,等. 区分服务及其几个热点问题的研究[J]. 计算机学报,2004(4):419-433.

[5] Conta A, Carpenter B. The IPv6 flow label and use of IPv6 flow labels with DiffServ[R/OL]. 2001. http://playground.sun.com/pub/ipng/html/presentations/aug2001/IPv6-flow-label-07.PDF.

[6] 李蔚泽. Red Hat Linux 9 系统管理[M]. 北京:清华大学出版社, 2004.

[7] Almesberge W. Linux Traffic Control [EB/OL]. 2001. http://feela.network.cz/ltc.html.

[8] Hubert B, Netherlabs B V. Linux Advanced Routing & Traffic Control[EB/OL]. 2003. http://lartc.org/lartc.pdf.