IPv6 流标签提供的服务质量支持*)

李 茹[1] 王春峰[1] 黄晓璐[1] 叶新铭[2]

(中国科学院计算技术研究所网络室 北京 100080)[1] (内蒙古大学计算机学院 呼和浩特 010020)[2]

摘 要 作为下一代互联网协议,IPv6 除了在地址空间上明显优于 IPv4 之外,还在数据报头中保留了 20 位流标签字段为实时流提供有别于尽力而为流的服务。流标签的研究尚处于试验阶段,目前还没有制定出正式的规范。本文在介绍了流和流标签的定义及属性之后,从流标签支持包分类、加速包转发、支持资源预留以及支持 QoS 的格式定义等多个方面探讨了流标签支持服务质量的能力,分析各种方式的利弊,给出可能的发展方向。

关键词 IPv6,流标签,服务质量

Aspects of IPv6 Flow Label in Supporting QoS

LI Ru[1] WANG Chun-Feng[1] HUANG Xiao-Lu[1] YE Xin-Min[2]

(Network Testing Lab, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080)[1]

(Academy of Computer Science and Technology, Inner Mongolia University, Hohhot 010020)[2]

Abstract As the next generation Internet Protocol, IPv6 has obvious advantages in extending address space. Another important fact is that IPv6 also defines 20-bit Flow Label to support QoS of real-time flows, and to provide service better than pure best effort delivery. Currently there is no standardized specification of flow label, and a lot of research works are still in experimenting stage. This paper first describes variant types of flows and their characteristics, and then introduces definition of Flow Label in IPv6 briefly. The paper mainly focuses on discussion about variant methods to use Flow Label to support quality of service, such as packet classification, packet forwarding, resource reservation and different QoS description formats in Flow Label. The paper finally analyzes the advantages and disadvantages of each method, and gives some potential research direction in future.

Keywords IPv6, Flow label, QoS

1 引言

Internet 的使用和基于 Internet 的应用的数量在过去几年里呈指数级增长,在传统数据应用的基础上,又出现了 IP 电话、视频会议等多媒体应用和服务。为了支持不同网络服务要求的语音、视频以及数据应用通信,IP 网络需要根据需求区分不同的通信,并为之提供服务。而目前 Internet 提供的尽力而为的服务不能区分成千上万种通信流,因此不能为任何应用通信提供优先级或者保证,因此也就无法运载对网络资源和服务有特定要求的通信。服务质量(QoS,Quality of Service)是网络传输流时需要满足的一系列服务要求,网络运营商据此为流分配网络资源,提供端到端的服务保证以及基于策略的性能控制。

流(flow)由从信源机器的应用程序到目标机器的应用程序的所有分组组成,IPv4 网络中通常使用数据报头的 5 个字段——源 IP 地址、目标 IP 地址、传输层协议标识符、源端口和目标端口区分流,属于同一个流的数据分组的 5 个字段的值相同。通过检查 5 个字段才能唯一地确定一个流从而确定流所提出的服务要求以提供相应的服务会引发所谓的层冲突问题。根据 OSI 协议分层的原则,对数据包的某种处理应该集中在 7 层模型中的某一层;然而由于这 5 个字段分散于网络层和传输层两个报头中,使得对流的处理需跨越两层才能完成,这就是层冲突;层冲突必然会降低流处理的性能。在 IPv6 中为了支持服务质量,在数据报头中保留了 20 位流标签(flow label)位。流标签与信源地址结合唯一地确定一个数据流,由此表示信源要求中间节点对该数据流进行非默认的处理。

虽然流标签是 IPv6 专门为支持服务质量而设置,但是关于流标签的用法和语义,在 IPv6 规范中并没有清楚的定义。关于如何利用流标签在 IPv6 网络环境中更好地支持服务质量目前尚处于试验阶段,因而成为限制 IPv6 广泛部署的一个重要方面。流标签对服务质量的支持基本上可以划分为两类,一类是在集成服务(Intserv, Integrated Services)体系结构中使用流标签帮助资源预留;主要的方法是先使用资源预留协议(如 RSVP)建立预留,然后在流标签中保存建立好的状态;其它的考虑也有直接利用流标签设计一种新的预留协议,省略高层预留协议(如 RSVP)的开销。另一类是区分服务(Differentiated Services)中使用流标签来划分流状态,通过设计 20 位中每一位以及位的组合代表的含义划分流所属的类型。还有一种综合的方案是考虑提供服务质量的各种可能方式,详细设计流标签的格式,尽可能完备地包括这些方面。

本文的目的是综合分析本领域内目前最新的研究状况,从流标签支持包分类、加速包转发、支持资源预留以及支持 QoS 的格式定义等方面讨论流标签支持服务质量的能力,分析各种方式的利弊,并给出未来可能的发展方向。

2 流和流标签

2.1 流标签

*)基金项目:国家自然科学基金项目资助(60273021)。李 茹 博士生。

对于流标签(Flow Label),IPv6 规范[5]中给出的定义是:源节点可以使用 IPv6 报头中 20 位的流标签标记信息包的顺序,源节点请求 IPv6 路由器对这些信息包进行特殊的处理,例如非默认的服务质量和"实时"业务。

IPv6 路由器在接受到一个信息包时,通过检查它的流标签,就可以判断它属于哪个流,然后就可以知道信息包的 QoS 需求。

IPv6 规范[5]中并没有完整地规定如何使用流标签域,定义之外就只有两点主要的规定:

1. 对不支持流标签的主机或路由器,要求在创建包时将该域设置为 0,在转发包是维持该域不做任何改变,在接收包时忽略该域的值。

2. 流的源节点为流分配流标签值;新的流标签值必须随机选择并且在 1 到 FFFFF(十六进制)之间均匀分布。

2.2 流

QoS 的两种体系结构——集成服务 Intserv 和区分服务 Diffserv 中都包含着流的概念。在 IPv4 中对流或微流的定义是一个应用程序到应用程序数据包序列的单个实例,数据包序列由源地址、源端口、目的地址、目的端口和协议标识符唯一标识。

集成服务体系结构[1]支持在每个流的基础上提供服务。集成服务模型使用资源预留协议作为标准的信令协议为网络中的应用程序流提供 QoS 支持。它提供三种服务类型:

1. 尽力而为服务(FCFS,意味着普通数据;默认方式);

2. 保障服务(意味着硬实时需求);已知延迟上界;遵照规格说明可靠地(无损失地)发送 IP 包;保障带宽支持;

3. 控制负载服务(意味着软实时需求)。

与集成服务提供"基于每个流"的 QoS 支持不同,区分服务提供"基于聚合流"的 QoS 支持。在区分服务体系结构中,根据服务级协商(Service Level Agreements—SLA)和流量整形协商(Traffic Conditioning Agreement—TCA)确定数据包所属的分类,从而决定分类域的值。

IPv6 规范[5]对流(Flow)的定义是:流是从特定源节点发往特定目的节点的包序列,而且源节点希望中间路由器能够对该包序列进行特殊的处理。

网络中的端节点与中间节点使用源地址和流标签来唯一地标识一个流,属于同一个流的所有包的处理方式都应该相同。根据流的目的是单个站还是一组站,可以分为单点传送流和多点传送流,但是流的概念始终是建立在无连接网络协议的基础上。

2.3 流标签的属性

流标签具有两种属性——端到端属性和逐跳属性。IPv6 规范给出的流标签定义只确定了流标签的逐跳(per hop)属性。规范中假设流标签具有能够帮助路由器操作系统处理包的功能,无论是在包的转发方面,还是在 QoS 处理方面。

关于流标签端到端属性目前存有争议。例如,有的观点认为流标签应该作为一种机制被目的端节点使用来识别一个流。但是这样的功能可以通过使用 IPv6 源地址和目的地址对以及主机到主机连接标识符来实现,所以流标签就只是上述机制的附加产品或替换品。然而如果路由器的包处理性能比端节点的包处理性能更为关键的话,强调流标签逐跳属性的重要性来使用流标签就具有更大的意义。

如果在路由器之间以及端节点之间存在某种协商机制,那么就没有必要讨论是使用流标签的端到端属性还是逐跳属性。流标签以及流标记协商(即通讯路径上经过的节点对应该赋给流什么样的标签达成一致)的技术已有类似的例子存在。例如 MPLS 中的标签分发协议(LDP),用来在邻居 MPLS 路由器之间交换标签;扩展的资源预留协议在邻居的标签交换路由器之间交换标签。但是对 IPv6 流标签来说这样的机制并不存在,因而可作为 IPv6 流标签未来发展的一个重要方向。

总之,就流标签的使用类型或它的重要性来说,流标签应该是一个二价属性:端到端和逐跳。端到端的意义不能取代逐跳的意义,反之亦然。如果一个节点在发送数据包时,在这些数据包的流标签上关联了某种端到端的含义,那么这种含义在到达最后目的地之前所经过的每一个路由器中都有意义。此外,流标签也可以被沿途经过的路由器修改,只要目的端节点能够明白流标签的值应该是什么,最后一跳的路由器可以自由选择是否恢复到修改之前的值。这样的行为应该是经过协商的,而且需要在中间路由器上存储状态,特别是要在最后一跳中保存。

关于中间路由器是否可修改流标签值的两种观点各有利弊。流标签不可变的最大优点是实现起来简单,而可变的最大优点则是其提供的灵活性,特别是当流标签具有逐跳意义的时候。但是要想使可变的流标签能够工作,必须有特定协议的支持,或者在邻居节点之间进行协商,或者对这些路由器进行特定的配置。这就需要在邻居路由器之间使用协商机制,或者是通过对路由器管理和配置完成安装过程,以确保数据包经过的路由器都明了流标签的值以及对流标签所做的变化。流标签的逐跳意义会因为这种可变属性而增强,但是 IPv6 规范[5]附录 A 中对流的如下要求必须放松或去掉:

1. 流由源地址和非零流标签的组合唯一标识。

2. 流的流标签值需由流的源节点分配。

3. 新的流标签值必须在十六进制的 1 到 FFFFF 之间随机和均匀地选择。随机分配的目的是使流标签域中任意位的组合都适合于被路由器拿来作为哈希值使用,以查找与流相关的状态。

4. 对以前使用过的流标签,在流处理状态的最大生命期内,源节点不能为新流重用该流标签。当一个节点停止并重新启动之后,节点必须小心不去使用曾经使用过的但生命期还没有结束的流标签的值。

显然在目前没有对流标签制定正式规范的情况下,各厂家分别实现自己的流标签可变方式,必然会引入同种网络产品之间的互操作性问题,而可能性最大的解决方式是定义并标准化 IPv6 流标签的协商机制。

3 流标签支持包分类

资源预留协议是实现 QoS 支持的主要方式,把这种方式称作流量控制。流量控制主要包括包分类、接入控制和包调度。接入控制用来保证网络节点有足够的可用资源满足允许进入网络的每个新请求的 QoS 需求;包分类器根据数据包所属的流确定其要求的 QoS;包调度器决定分组得到服务的顺序,由此提供允诺的 QoS。

就 RSVP 协议来说,接入控制和实际的资源预留的完成是在流的建立阶段,因而性能只会影响到对用户的响应时间,而不会直接影响到实时流数据的传输。包调度是通过复杂的排队算法来支持不同类型的服务,如公平排队(FQ)、加权公平排队(WFQ)、修正加权循环(MWRR)等,调度本身完全独立于网络层的协议。包分类则完全依赖于网络层或任何其它

上层协议。中间节点的路由器根据数据包所属的流的分类来确定其QoS类型。在IPv4中,包分类需要根据数据包的源地址、目的地址、传输层协议标识符、源端口号和目的端口号来确定,其中源地址和目的地址位于网络层,源端口号、目的端口号和协议标识符位于传输层,路由器对数据包进行分类时要在分类规则表中检查两个数据报头,不但处理困难而且不符合协议分层的要求。大多数的时候,这些报头是连续的,域的位置是已知的,因而处理可以流水或并行进行。但当一个或多个IPv4安全头存在时,就会打断报头的连续性;因为加密的数据包会把传输层报头加密,所以属于传输层的域就不能再作为数据包流分类规则的一部分。此外当数据包具有扩展头时,更增加了处理的难度。尤其是如果在IPv6中仍然使用传统的包分类方式,由于IPv6的扩展头规模更大,使用更频繁,会进一步增加流水或并行处理的困难而使效率再度降低。利用流标签作为包分类器,自然地解决了上述问题的同时还极大地提高了系统的效率。下面给出具体的性能分析。

网络路由器中包分类器的简单模型可以描述成一个进程,根据数据包的网络层和传输层报头的信息决定包所属的流;根据包所属的流查找流状态表,根据流状态表确定包应该进入的队列,调度算法由对不同队列的不同调度提供不同的服务。

传统RSVP对数据包的分类过程是:

第一步:在会话表中查找目的IP地址

第二步:比较传输层协议标识符

第三步:比较传输层目的端口号

第四步:比较数据包的源IP地址与过滤地址

第五步:比较传输层源端口号

前三步确定数据包所属的会话,后两步确定数据包是否与过滤规则相匹配。

为了使基于地址的表查找的处理成本最小,可以使用复杂的算法如哈希函数或Patricia树。如果在比较协议标识符之前,目的IP地址和传输层目的端口已经与会话相匹配,则数据包就极有可能属于该会话;这是因为当前使用RSVP的应用程序都主要以UDP作为传输协议。因而,先执行第三步在会话表项中比较数据包的传输端口,再执行第二步校验协议标识符,可以使包分类器的性能更好。

因为在IPv6数据报头中存在唯一的且是随机分布的流标签值,所以在支持流标签的IPv6包分类器中可以以一种更有效的方式执行分类处理。首先使用流标签作为关键字查找流状态表。尽管不同源节点的流使用相同流标签的概率非常小(20位键码空间中键值冲突的概率),还是需要使用流的源IP地址来解决可能存在的冲突。因而,第二步需要比较源IP地址。总之,基于IPv6流标签的数据包的分类过程是:

第一步:根据流标签查找流状态表

第二步:比较源IP地址

由于IPv4和IPv6地址结构与报头格式的不同,分类需要执行的各项操作的开销也不相同。如IPv6 128位地址的查找和比较要比对IPv4 32位地址操作的开销大,而IPv6数据包中大量使用扩展头,也使得获得传输层端口号的开销远大于IPv4。就IPv6本身而言,由于基于流标签的分类方式解决了分层冲突问题,而使得其性能要明显优于基于传统分类方式的包分类。文[8]给出了IPv4包分类、IPv6包分类和基于流标签的IPv6包分类三种方式的性能比较。通过计算每包处理成本PPC(per Packet Processing Cost),文[8]进行的理论分析结果表明基于IPv6流标签的分类方式要比IPv6传统的5元组分类方式性能提高3～6倍;即使IPv4的地址长度要小,基于IPv6流标签的分类方式比IPv4传统5元组的分类方式性能还会提高2～4倍。当查找表的空间足够大时,流标签冲突引发的问题只会对分类性能产生很小的影响。

由于从源到目的数据包经过的每一个路由器都需要执行包分类,因此包分类的性能直接影响到数据包端到端的传输属性,例如实时流中普遍关注的属性——传输延迟。由于流标签对包分类性能的巨大改善使得IPv6对实时应用程序的支持明显优于IPv4。

4 流标签支持资源预留

4.1 扩展RSVP支持流标签

关于如何扩展RSVP以支持流标签的讨论还很少,目前只存在一个Berson草案[11]。Berson草案中考虑了扩展的四个方面:第一,格式问题,如修改RSVP FILTER_SPEC格式,同时包含IPv6流标签和源端口;第二,分配问题,不存在一种IPv6 API可供RSVP进程获得流标签;第三,接口问题,由于数据包的路由可能依赖于流标签,所以RSVP与路由的接口需要访问流标签;最后,安全问题,如节点假冒流标签获得想要的服务或者是拒绝服务攻击。

文[4]中给出的FILTER_SPEC针对IPv6定义了两种格式:Class＝10, C-Type＝2的FILTER_SPEC对象包含IPv6源地址和源端口;Class＝10, C-Type＝3的FILTER_SPEC对象包含IPv6源地址和24位的流标签。之所以是24位的流标签是因为文[4]先于文[5]定义,此后的文档需要修改成20位。Berson草案为了同时兼容支持流标签与不支持流标签的路由器,定义Class＝10,C-Type＝6的FILTER_SPEC对象同时包含IPv6源地址、源端口和流标签值。

关于流标签的分配问题,IPv6套接字接口定义在文[12]和文[10]中都没有给出具体的方法。根据对流标签分配的两点主要要求:流的流标签值需由流的源节点分配和新的流标签值必须在十六进制的1到FFFFF之间随机和均匀地选择,Berson草案认为流标签应该由操作系统负责分配以保证其唯一性和随机性,而且由于RSVP要在其PATH消息中包含流标签的值,因此流标签值应能供RSVP访问。最简单的方式就是提供一个系统调用,但是这种方式使流标签的分配成为一种向用户提供的函数而不再是一种系统功能。第二种方式是通过套接字使用流标签,即要求bind调用使用流标签,这样一个流标签值就与绑定好的套接字联系在一起;为了把该流标签值提供给RSVP,还需要一个系统调用从套接字中提取该值。最后一种方式就是当观察到有类似流的行为时,内核自动开始使用流标签值。这时内核需要有一个RSVP上层调用向RSVP守护进程传递RSVP流描述符;而RSVP需要定位到正确的预留并加入流标签域。

文[4]中要求RSVP信令消息应沿着与数据消息相同的路径发送(PATH消息与数据包的路径相同,RESV消息使用数据包的反向路径)。Zappala草案[9]定义RSVP与路由的接口RSRR,允许RSVP访问路由表以确定信令消息的转发路径。但RSRR中只包含基本的源地址和目的地址信息,文[5]中对流标签的定义表明路由转发可能会基于流标签来实现,那么为了使RSVP信令消息仍然采用正确的路径,需要扩展RSRR以支持流标签。除了源地址和目的地址之外,路由决策还受到其它因素的影响。例如提供QoS支持的区分服务体系

结构中，数据包的转发是根据所属的流量类型完成；而流量类型可由 IPv4 数据报头的 TOS 域、IPv6 数据报头的 COS 域确定。如果采用源路由策略，数据包经过的路径由 IPv4 数据报头的选项字段、IPv6 数据报头的路由头字段明确指出。有关这些方面在 RSRR 中都没有考虑，RSVP 需向 RSRR 提供更多的信息以做出正确的路由决策。

根据文[5]，使用流标签和源地址可以唯一地确定一个流，沿途的路由器可据此提供非默认的服务质量。这里隐含着的安全隐患是攻击者冒用流标签值获得更好的服务，或者由于攻击者冒用流标签而使原有者无法获得服务。虽然可以使用根据源地址、目的地址和流标签对数据包进行过滤来解决这个问题，但是这种方法降低了通过使用流标签而提高的效率。

所以如果使 RSVP 能够支持流标签，首先，要使 RSVP 可从应用程序获得流标签的接口中获得流标签的值；其次，要扩展 RSVP 与路由模块的接口使用流标签值；第三，修改 RSVP 规范本身支持根据流标签的流过滤；第四，弥补使用流标签引入的安全隐患。

4.2 基于流标签的资源预留协议

先使用资源预留协议建立流状态然后传输数据包的方法的主要缺陷有三：一是流的建立时间增加；二是复杂的资源预留协议导致开销增大；三是预留状态无法及时地反映网络状态的变化。好的资源预留可以保证服务提供商按照与用户事先签订好的合同准确无误地传输业务流，然而用户获得满足的代价是网络利用率的降低。如果用户可以容忍某种程度的包损失以及服务性能上的变化，那么可以在数据包传输的同时隐式预留。

Birkner[3]提出的 ImpRes(Implicit Reservation)协议使用 IPv6 流标签和逐跳选项头为数据流动态建立和维护预留。在 ImpRes 协议中，每个数据包都携带网络处理该数据流所需要的完整信息。如果网络节点上有足够的资源可用，则分配给该流；如果没有，则向流提供可用的最好服务。在流的生命周期内，这种分配可以动态地改变。这种方式无需建立时间，而且可以平滑地实现重新路由。逐跳选项头中包含建立和维护流状态所需的预留信息：由一组固定域和任意数量的附加域组成。固定域存在于具有非零流标签值的每个数据包中，主要包括时间戳字段，指明源生成数据包的时间；刷新周期字段，用于通知中间节点每隔多久需要读入 ImpRes 报头以更新流状态；平均带宽字段，指明在预定义的时间间隔内流想要的平均带宽。为流建立的状态是"软"状态，由属于相同流的数据包的到达事件维持激活；每个状态由〈流标签、源地址和目的地址〉唯一标示。

预留变化的时间粒度为刷新周期，每当刷新计时器超时时，发送者可以决定是否改变预留，或者网络可以决定资源是否需要重新分配。如果源节点改变资源需求，路由器要决定增加还是降低分配给流的资源；降低很容易处理，只需修改预留状态并释放多余资源。增加资源无异于重新接入流，需要运行接入控制判断是否有可用的资源。

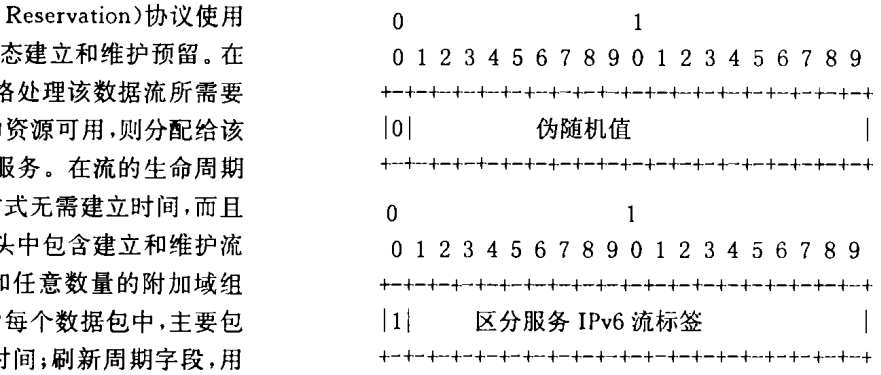
异常情况下的动态重路由导致从不同的输入线上接收到来自相同流的包，比较未预期输入线上数据包的时间戳和预期输入线下一个数据包的时间戳，转发最新生成的包，并对预留状态做相应的改变，此后旧的输入线上的数据包将被丢弃，以此来保证属于同一个流的数据包的按序发送。

状态的拆除可使用显式、隐式两种方式完成；显式的状态拆除需要路由器向下游转发控制消息；隐式的状态拆除在软状态的超时间隔过期后自动完成。

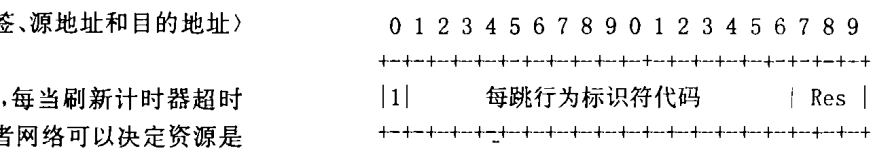
ImpRes 协议显然无法与 RSVP 协议相提并论，只实现了一个简单的框架，但是它鼓励后续者从不同的角度探讨流标签各种可能的使用方式。在 ImpRes 中尚有许多问题未做详细的讨论：第一，每个数据包都携带网络节点处理数据流所需的必要信息会急剧地增加网络开销；第二，文[5]中要求如果数据包存在逐跳选项头，则中间节点的路由器必须处理，那么不可能实现按照刷新周期定期地检测数据包的 ImpRes 报头；第三，动态重路由涉及到很多方面，例如，如何获得数据包的入口和出口信息，即上一跳和下一跳的节点地址；路由的变化如何反馈到 ImpRes 协议中；路径变化后能否仍然成功地预留等问题都未做考虑。第四，状态拆除的详细过程未做考虑，如显式状态拆除的控制消息存放在数据包的什么部分，格式如何等。总之，作为一个完整的协议需要考虑到每一个细节，才有可能在实践中应用，ImpRes 协议还有待进一步地完善。

5 流标签支持 QoS 的各种格式

虽然文[5]中要求流标签是一个随机分配的值，但是在区分服务体系结构中，随机分配的值无法具有事先定义好的意义。服务提供商和用户事先商定若干分类，以及为每种分类提供的服务，但是根据一个随机值无法确定流属于哪一种类型。因此，人们试图改变流标签是随机值的属性，定义流标签可能的各种格式，由此赋予流标签一些固定的意义。在 Conta 草案[14]中给出了流标签格式定义的各种可能方法。草案中使用 20 位中的第一位区分随机数和预定义值这两种情况：



区分服务 IPv6 流标签的值根据"每跳行为标识符代码(Per Hop Behavior Identification Code—PHB ID)"来构造。



Res. 表示预留。

PHB ID 或者是直接从标准的区分服务代码点派生，或者是"IANA 分配的值"(IANA，Internet Assigned Number Authority)。无论是哪一种情况，都可以实现区分处理数据包。正如第 3 节流标签支持包分类中所述，流标签通过替代传统 MF 分类器中源端口号、目的端口号和主机到主机协议标识符域解决了层冲突问题，由此可提高包分类的效率，改善实时流的性能参数，如传输延迟。虽然说 IPv6 数据报头中 8 位的流量类型域(COT)也可以用来支持区分服务，但是与流量类型域只能本地映射不同，流标签域可以具有端到端的含义，

关于这一点也已在第 2.3 节流标签的属性中进行过讨论。

在服务提供商网络中可以使用下列两种方式的任何一种来配置 MF 分类规则表：

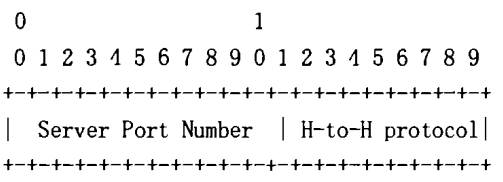
C=(SA, SAPrefix, DA, DAPrefix, Flow Label).

C'=(SA, SAPrefix, DA, DAPrefix, Flow Label min: Flow Label max).

把数据包的报头(SA, DA, Flow Label)与分类规则表项 C 或 C' 匹配，C' 方式中落入 Flow Label min 到 Flow Label max 范围内的流标签值都属于同一种流量类型。

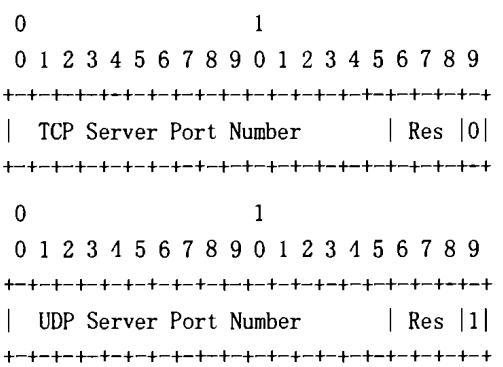
这里存在的一个问题是 IETF Diffserv 工作组定义的标准中 DSCP 值只使用 6 位，在上述方法中扩展到了 16 位，位数的扩展能够带来什么样的优势以及支持哪些扩充的功能还需进一步的探讨。

Conta 草案中的第二种方式是对传输层的协议类型和源端口号与目的端口号进行压缩或编码赋给流标签。下面的这种方式用于客户机/服务器程序，在流标签中映射服务器方的端口号和主机到主机协议。端口号使用 12 位，协议占用剩余的 8 位。



通过在流标签中标明应用程序的类型，也就表明该应用程序产生或接受的流量 QoS 需求。这种方式的缺点是无法区分运行在两个通讯端节点之间相同应用程序的多个实例，也就是说，当相同的应用程序却要求不同的服务质量时无法保证。而且端口号从 16 位缩减到 12 位限制了端口的值只能表示“IANA 周知端口”(1-1023)和部分“IANA 注册端口”(1024-4095)。优点则是与旧的 5 或 6 元组方式相兼容。

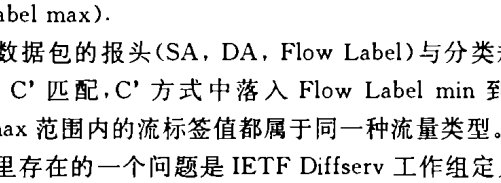
由于协议报头中包含端口号的两种主要的传输层协议只有 TCP 和 UDP，所以上述方式中主机到主机协议域可以减少到 1 位，这样端口号部分可以扩展到 16 位。



这种方式的优点是端口号扩展到 16 位，可以容纳所有的“IANA 周知端口”和“IANA 注册端口”。缺点则是流标签中只留 1 位作为协议域限制 TCP 和 UDP 之外的其它协议类型的使用。

首先这种方式单纯地为了使流标签替代位于传输层的三个域以解决分层冲突问题而设计使其具有很大的局限性，它限制了流标签在更广范围内的使用。而且，由于位数的限制，在这种方式中只能包含相对更为重要的服务器的端口号，而无法支持对等实体间的通讯。因此许多端到端的实时应用程序如 VoIP 将无法使用该功能。

Conta 草案中提议的第三种方式是在流标签中存储 IPv6 报头长度，指 IPv6 基本头和扩展头长度之和。域中头长度信息可以供区分服务 QoS 引擎分类器直接定位并获取源和目的端口号，结合源和目的地址以及流标签中主机到主机协议，去匹配区分服务 MF 分类规则表中源和目的地址、源和目的端口号以及协议标识符元素。



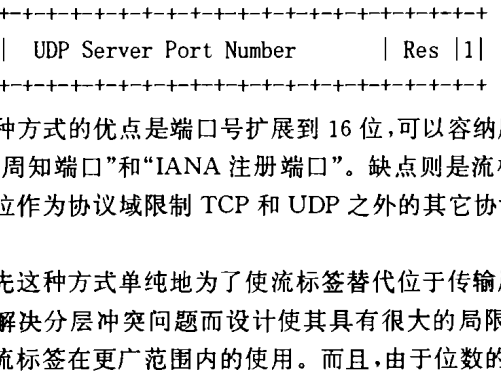
这种方式的优点是“IPv6 头长度”允许出于其它目的跳过 IPv6 头而直接去访问主机到主机头。这种格式在分类没有源和目的端口的非 TCP 和 UDP 的数据包时很有用。缺点则是 IPv6 报头中并没有包含“全部头长度域”。所以在流标签中引入这个新域需要额外的计算，因而会造成处理延迟；此外在流标签中包含“IPv6 头长度”当在 IP Security 中使用 ESP 时不会有任何作用。

Banerjee 草案[15]在上述方法的基础上提出了一种混合方案；流标签中的前 3 位用来定义使用的方法，后 17 位用来定义特定方法中的格式。前 3 位定义的 8 种方法类型是：

000	默认值
001	随机数值
010	使用 IPv6 逐跳扩展头的值代替流标签的值
011	PHB ID
100	端口号和协议
101	QoS 参数值
110	预留
111	预留

后 17 位的定义主要根据前 3 位的定义而变。默认值指明数据包无需特殊的 QoS 处理。随机数值的方法中 17 位随机数的范围从 1 到 1FFFF。逐跳扩展头方法用于集成服务模型，忽略 20 位的流标签值而使用逐跳扩展头。PHB ID 的方法同 Conta 草案，只是流标签的格式如下：

端口号和协议方法同 Conta 草案，流标签的格式修改如下：

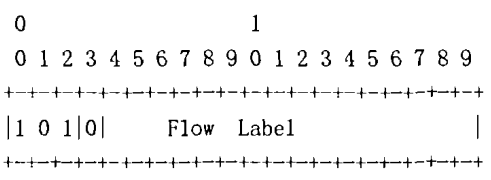


QoS 参数值方法是在流标签中存放各种 QoS 参数，主要考虑的参数类型有：1)带宽；2)延迟或滞后；3)抖动；4)包损失；5)缓冲区需求。

草案中认为任何应用程序都要求包损失和抖动最小，因

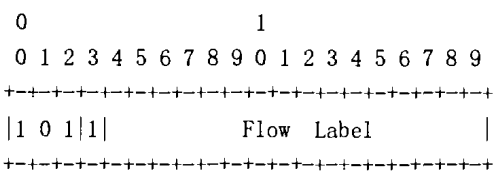
而可以在流标签域中省略，必要时可以在逐跳扩展头中指明。所以在流标签域中只需指明带宽、延迟和缓冲区需求。17 位中的第一位用来区分硬实时和软实时应用程序，1 表示硬实时，0 表示软实时。

软实时应用程序：



软实时应用程序的任意包具有一般的带宽需求和中度的端到端延迟。即使没有满足流标签中规定的值应用程序也可以容忍。

硬实时应用程序：



硬实时应用程序要求最小的延迟与抖动。

剩余的 16 位中带宽占用 6 位，中间 5 位表示缓冲区需求，最后 5 位表示延迟。这种方法实现的区分服务的分类器由〈带宽、缓冲区、延迟〉三元组构成。路由器检查到达数据包中此三个域中指明的值，据此提供服务。

这一种方式首先存在的问题是 010 类型只表明 IPv6 扩展报头中的逐跳扩展头将具有指明 QoS 含义的作用，流标签中剩余的 17 位将不代表任何含义，由此造成很大的浪费。IPv6 规范[5]中规定如果 IPv6 数据包中包含逐跳扩展头，那么沿途经过的任何结点都必须对该报头做出处理。因此，这样的功能可通过定义逐跳扩展头的类型来实现，无需单独定义一种类型的流标签。其次，010 逐跳扩展头类型只是针对 101QoS 参数值类型的扩充，同时存在两种相似功能的类型占据宝贵的流标签类型空间的意义在此并不明显。第三，在 101QoS 参数值类型中，带宽、缓冲区和延迟并不能代表现在 Internet 上广泛使用的实时应用程序的所有需求，例如视频点播更关心的是延迟抖动参数，由此削弱了这种类型的广泛适用性。而且，即使是用这种方式来表示应用程序要求的 QoS 参数，每种参数所能表示的范围也很有限。以带宽为例，假设这里采用的单位是 kbps，即每秒千比特，6 位所能表示的范围是 0 到 64kbps；如果单位是 Gbps，则最小的需求就已达到 1G，这样的范围是无法满足现在实时流的多样需求的，也就是说 6 位是远远不够的，缓冲区和延迟两种类型也具有相似的问题。流标签由于其空间有限，并不适合实现这种功能，更好的方式还是在逐跳扩展头中携带应用程序 QoS 参数需求，位数更多，灵活性更好。

综上所述的各种方式都主要是从两条线来考虑，第一，在 IPv6 流标签中存放 DSCP 值以支持区分服务，但是这种方式取代现有使用 IPv6 COS 字段存放 DSCP 值的优点尚未讨论清楚。第二，在集成服务中使用流标签作为包分类器解决分层冲突问题，这种方式的优点已在第 3 节中作了充分的讨论。但是集成服务是针对每个流来作处理，在目前 Internet 上有成千上万个流同时存在的情况下，集成服务体系结构已被普遍认为不具有可扩展性。而且流标签只凭在这一点上的显著优势并不足以说服业界更改已普遍实现的 RSVP 五元组的分类方式。关于流标签支持 QoS 的更好方式还需要人们做更深入的研究。

结论 在 IPv6 转发设备中，流标签可在查找下一跳、服务质量、包过滤引擎三个方面对包进行更有效的处理。本文的讨论是从服务质量方面来讨论流标签的用法和语义。IPv6 规范[5]定义流标签为 20 位的随机整数，并说明流标签本身以及流标签中任意位的组合可以作为哈希函数值使用来加速数据包的转发，也就是说使用流标签查找下一跳的功能。但是这三种功能其实是密不可分的，查找下一跳功能的实现需要流信令机制的支持，而流信令机制需要根据在服务质量中定义的流标签的语义在数据包的转发路径上建立流状态表，来确定数据包的输出端口。服务质量需要包过滤引擎定义的包分类方式。包过滤引擎对数据包的分类不仅仅只局限在支持 QoS 方面，还有安全、性能、策略等多方面的考虑。然而无论从哪个方面而言，对流标签的探索尚都处于初期阶段，流标签标准的制定在推动 IPv6 的广泛部署中具有举足轻重的地位。

参 考 文 献

1 Braden R, Clark D, Shenker S. Integrated Services in the Internet Architecture: an Overview. RFC 1633, June 1994

2 Partridge C. Using the Flow Label Field in IPv6. RFC 1809, June 1995

3 Birkner M. ImpRes: Supporting Services in IPv6 using the Flow Label and Hop-by-Hop Option Fields. M. S. Thesis, Auburn University, Oct. 1996

4 Braden R, et al. Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification. RFC 2205, Sep. 1997

5 Deering S, Hinden R. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460, Dec. 1998

6 Blake S, Black D, Carlson M, et al. An Architecture for Differentiated Services. RFC 2475, Dec. 1998

7 Nichols K, Blake S, Baker F, Black D. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers. RFC 2474, Dec. 1998

8 Schmid S, Scott A, Hutchison D, Froitzheim K. QoS based Real Time Audio Streaming on IPv6 Networks. SPIE, Nov. 1998, 3259:102～113

9 Zappala D, Kann J. RSRR: A Routing Interface For RSVP. Internet Draft, June 1998

10 Stevens W, Thomas M. Advanced Sockets API for IPv6. RFC 2292, Feb. 1998

11 Berson S. RSVP and Integrated Services with IPv6 Flow Labels. Internet Draft, June 1999

12 Gilligan R, Thomson S, Bound J, Stevens W. Basic Socket Interface Extensions for IPv6. RFC 2553, March 1999

13 Metzler J, Hauth S. An end-to-end usage of the IPv6 flow label. Internet Draft, Nov. 2000

14 Conta A, Carpenter B. A proposal for the IPv6 Flow Label Specification. Internet Draft, July 2001

15 Banerjee R, Mahaveer M. A Modified Specification for use of the IPv6 Flow Label for Providing An efficient Quality of Service using a hybrid approach. Internet Draft, March 2002

16 Rajahalme J, Conta A, Carpenter B, Deering S. IPv6 Flow Label Specification. Internet Draft, March 2002