

基于上下文转移机制的移动 IPv6 服务质量方案¹

陈 阳 杨寿保 刘 沙 孙伟峰
(中国科技大学计算机科学技术系, 合肥, 230026)

摘要 针对移动 IPv6 节点切换时, 移动节点参与 QoS 信息重建的不足, 本文基于 HMIPv6 QoS 模型和上下文转移机制, 提出了一种新的改进切换时服务质量的方法。这种方法能够显著减少切换时 QoS 的延迟时间, 使得实时业务流能够获得更好的服务质量保证。

关键字: 服务质量, 上下文转移, 层次化移动 IPv6 服务质量模型

Context Transfer-based MIPv6 QoS Solution

CHEN Yang YANG Shoubao LIU Sha SUN Weifeng
(Department of Computer Science, University of Science and Technology of China, Hefei, 230026, China)

Abstract Because of the lacks of the Mobile Node participating QoS information re-establish in Mobile IPv6 Node's handover. Based on HMIPv6 QoS model and context transfer, this paper introduces a new method to improve the QoS when Mobile Node performs handover. The method can reduce the latency of mobile node in handover, and provide better QoS pledge to real-time flow.

key words: Quality-of-Service, Context Transfer, HMIPv6 QoS Model

¹本文得到国家 863 项目 (编号: 2001AA121041) 的资助。

作者简介: 陈 阳: 男, 硕士研究生, 研究方向: 移动 IPv6、计算机体系结构。杨寿保: 教授, 博士生导师, 研究方向: 计算机网络及应用。

引言

随着无线通信的飞速发展,移动网络中的实时业务与日俱增,数据传输对时延、带宽等的要求越来越高。这就对对移动网络条件下提供确定的服务质量提出了新的要求。当移动节点(Mobile Node 以下简称 MN)切换到新的子网时,传统上要求由移动节点重新建立 QoS 信息,如果 MN 发生频繁切换,这个 QoS 时延对于一些时延敏感的实时业务(如 VoIP、视频流)来说,无疑是不可忍受的。本文以 IETF MobileIP 小组提出的层次化移动 IPv6(HMIPv6)模型作为基础,结合 Context transfer(以下简称 CT)机制,实现移动节点在层次化移动 IPv6 QoS 模型中的平滑切换。

1. Context transfer 及其必要性

IETF 的 RFC3374 中明确提出了 context transfer 的定义,所谓 context 就是当前状态的服务信息,它需要在新的子网内重新建立,但不必要由移动节点重新开始参与整套的协议交换。所谓 context transfer 即上下文转移机制,指将上下文从一个路由器或其它网络实体转移到另一个上去,以作为在新的子网上重建某种服务的手段,这里所指的服务,可以是 QoS 信息、AAA 信息、头标压缩信息等等。

对于在移动节点上有 QoS 要求的实时业务而言,发生切换时不仅保证移动节点的移动对网络层以上透明,而且还将面临重建 QoS 信息的时延问题,如果每次切换到新的子网时,移动节点的 QoS 信息都必须由移动节点重新建立,这样的时间开销对于实时服务而言是不能忍受的,因为在 QoS 建立之前的这一段时间内,MN 的实时服务将得不到期望的 QoS 的需求,从而不得不忍受可能出现的延迟抖动和丢包。因此,有必要在现有移动 IPv6 QoS 模型的基础上加入 context transfer 的功能,以新旧网络之间传递移动节点 QoS 信息代替移动节点重新建立 QoS 信息,从而提高移动节点切换时 QoS 信息的建立速度。由于一个实时业务流在传输过程中有统一的 QoS 需求,即原先建立的 QoS 信息在切换过后仍然适用,因此以 QoS 信息作为一种上下文是可行的。

2. Intserv/RSVP 简介

Intserv/RSVP 是一种常用的保证实时业务服务质量的方法,它基于每流进行资源的预留,一个流指的是从特定的源端发往特定的目的端的一系列数据包。在 IPv6 中流可以用(源地址,目的地址,流标记)这个三元组来表示流,其中 COA 称为移动节点的转交地址,是移动节点在离开家乡子网后获得的地址,CN 是与 MN 进行通信的通信对端。流透明是移动环境下 QoS 的基本要求之一,流透明指的是当移动节点发生切换过后,流的标识不发生改变,RSVP 可以沿用之前公共路径上的预留,只要对路径改变的部分进行重新预留即可,如果流的标识发生改变,RSVP 将对整个路径上资源重新预留,造成资源的浪费和很大的时延。普通 MIPv6 模型与 Intserv/RSVP 的结合由于标识流的三元组中的 COA 切换时会发生改变,因此无法实现流透明。

3. 用于 QoS 上下文转移时的 CT 信令

Context transfer 协议可以作为基于 UDP 的上层协议运行于接入路由器(AR)上,它的主要消息主要有:上下文切换触发请求包(CTAR),上下文切换请求(CT Request),上下文信息(CTD)等等。其中 CTD 是切换前路由器(PAR)向切换后路由器(NAR)提供 MN 的上下文信息,对于携带 QoS 的 CTD 来说,它的结构如图 3-1 所示。其中 MN 在 NAR 内的地址,MN 经过的链路层触发后,通过 PAR 可以获得 NAR 的地址前缀并生成 MN 在 NAR 内的地址,如果改地址经 DAD 顺利则设 C 位为 1。从安全方面考虑,为了确认 MN 的真实性,

我们还必须对 MN 进行认证,当 MN 正式接入 NAR 域内时 MN 必须发送与 PAR 发送给 NAR 相匹配的认证信息,经过认证之后, MN 才能够使用从 NAR 到 MAP 已预留好的信息。最后我们传输的 QoS 的上下文包括流标记和具体的 QoS 信息,流标记用以对应确定的实时业务。

1

32

消息类型	C	R	rsv	消息长度
时间(微秒)				
MN 在 PAR 内的地址				
MN 在 NAR 内的地址,If C=1				
认证类型	长度		认证算法	密钥长度
密钥				
流标记				
QoS 信息				

图 3-1 CTD 消息的结构

4. HMIPv6 QoS 模型以及 Context transfer 与 HMIPv6 QoS 模型的结合

本章将介绍我们为什么要采用 HMIPv6 模型以及 CT 与 HMIPv6 QoS 模型结合较与普通 MIPv6 QoS 模型结合的优势所在。最后介绍 CT 对 HMIPv6 模型中接入路由器 (AR) 的扩展。

4.1 HMIPv6 QoS 模型

HMIPv6 模型是在普通 MIPv6 模型上发展而来的模型,旨在提高对移动节点 QoS 的支持。HMIPv6 模型引入了移动停靠点 (MAP) 实体,增加了 MAP 域的概念。实验证明移动节点 69% 都是在一个 MAP 域内移动,在域内移动时,移动节点拥有 (RCOA, LOCA) 地址对,其中 RCOA 是 MN 域内移动保持不变的地址, LOCA 是 MN 在域内具体子网中的地址。MN 对外使用 RCOA 进行通信,当 MN 在域内子网间切换时, LOCA 将发生改变, MN 仅仅需要向 MAP 发送本地的绑定更新绑定新的 (RCOA, LOCA) 地址对。这样一来,在 HMIPv6 模型与 Intserv/RSVP 结合的 HMIPv6 QoS 模型中, MN 采用三元组 (RCOA, CN Addr, 流标记) 标识一个流移动节点在域内移动时就能够达到流透明。

4.2 Context transfer 与移动 IPv6 QoS 模型的结合

在基础的 MIPv6 QoS 模型中当 MN 由 PAR 子网切换到 NAR 子网时,在 MN 到 CN 的路径上可能发生多个路由器的改变,每次需要寻找一个所谓最近公共路由器这样就需要在每个新的路由器上传递 MN 的 QoS 信息,这就给 context transfer 造成很大的麻烦,如果采用层次化 MIPv6 模型,则 context transfer 所涉及到的发生变化的路由器最多到 MAP 就截止了,使得 context transfer 更为容易控制。当 MN 作为实时业务的发起端时,只需要在 PAR 与 NAR 之间进行 context transfer 而由 NAR 完成从 NAR 到 MAP 之间的资源预留。

在基础的 MIPv6+Intserv/RSVP QoS 模型中,当移动节点切换到新的子网时, COA 将发生改变,因此流的标识将发生改变,实现流透明比较困难。而 HMIPv6 QoS 模型的一大特点就是可以实现 MN 在 MAP 域内移动时的流透明,即 MN 在域内子网间切换时对流的标识对于 CN 来说不发生改变,切换前后仍然被认为是同一个流,从而大大的减少了重新预留的开销。当每次 MN 由 PAR 切换到 NAR 时, MAP 到 CN 这段主干路径上重用切换之前所作的预留,只要对 NAR 到 MAP 这段较短的路径上进行局部预留即可。这一点在 AR 加入 context transfer 功能之后,可以更好的减少建立局部预留的时间。当 PAR 将 MN 的上下文信息传递到 NAR 之后, NAR 可以作为 MN 的 RSVP 代理向 MAP 进行资源预留,由于 context transfer 发起于链路层的触发,很可能当 MN 正式接入到 NAR 子网内并向 MAP 发布绑定更新 (BU)

和接受绑定应答 (BA) 的同时, 由 NAR 到 MAP 的资源已经被预留好。

所以不难看出, CT 与 HMIPv6 QoS 模型可以更好的结合起来。

4.3 CT 对 HMIPv6 QoS 模型中 AR 的扩展

在支持 CT 的 AR 上,除了一般接入路由器的功能之外,还需要充当 MN 的 RSVP 代理。所谓 RSVP 代理是将 HMIPv6 中的 AR 为适应 CT 所做的简单扩展,当移动节点没有正式接入到 NAR 之前,使用 NAR 作为 MN 的 RSVP 代理,替 MN 向 MAP 发送 Path 和接收 Resv 消息,提前建立从 NAR 到 MAP 的预留信道。当 MN 正式接入 NAR 子网时,可以通过简单认证获得已被预留好的资源。如果发生 MN 切换的预测错误,即 MN 并没有真正的切换到 NAR 子网内,则 NAR 可以撤销已建立的预留。

5. 切换过程分析

CT 分为网络触发和 MN 触发,二者的不同在于网络触发由 NAR 或 PAR 产生链路层的触发(trigger)还是由 MN 产生链路层的触发。这里产生我们采用由 MN 触发的预言性 context transfer,设 MAP 域内有 PAR、NAR 为接入路由器的两个子网,MN 在 MAP 域内移动。当 MN 在 PAR 子网内时,设它与已 CN 建立好了相应的资源预留,并由 MN 发起到 CN 的一条实时业务流,如图 5-2a 所示。随后,当 MN 将要向新的子网切换时,MN 感知到新的链路层连接,产生一个上下文转移触发(CT trigger),继而 MN 将发送 CTAR 给 NAR,其中 CTAR 包含 MN 在 PAR 子网内的 COA 及 MN 的授权密钥(Authorization Token)等等,这个信息可以通过经 PAR 的隧道抵达 NAR,NAR 收到 CTAR 之后,知道 MN 可能要进行从 PAR 子网到 NAR 子网的切换。于是 NAR 将依据 CTAR 中提供的信息向 PAR 发出 CT Request,其中包含请求信息的类型,这里的类型是 QoS。PAR 收到请求后将向 NAR 提供 MN 的上下文信息(CTD)完成邻近路由器间上下文的转移。切换时的上下文转移消息时序如图 5-1 所示。当 NAR 收到来自 PAR 关于移动节点的 CTD 之后,从中可以获得 MN 在 NAR 上的转交地址(NCOA)以及要求的 QoS 信息,于是 NAR 将作为 MN 的 RSVP 代理向 MAP 发送 RSVP 请求,完成这一段局部的资源预留,在 NAR 向 MAP 发送资源预留请求的同时,MN 可能已经正式接入到 NAR 子网内部,于是 MN 将向 MAP 发出局部绑定更新(BU)以及接收绑定应答(BA)。这样,相当于 MN 的局部绑定与 NAR 的资源预留并行进行,而且 NAR 向 MAP 的 QoS 请求先于 MN 的 BU,如果 NAR 与 MAP 之间能够提供足够的资源预留,则 MN 在收到 BA 之后就能够进行有 QoS 保证的实时应用了,如图 5-2c。

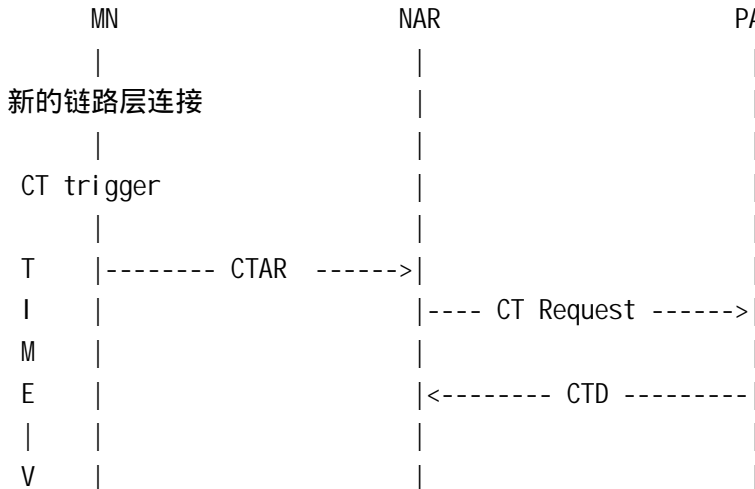


图 5-2 context transfer 时序图

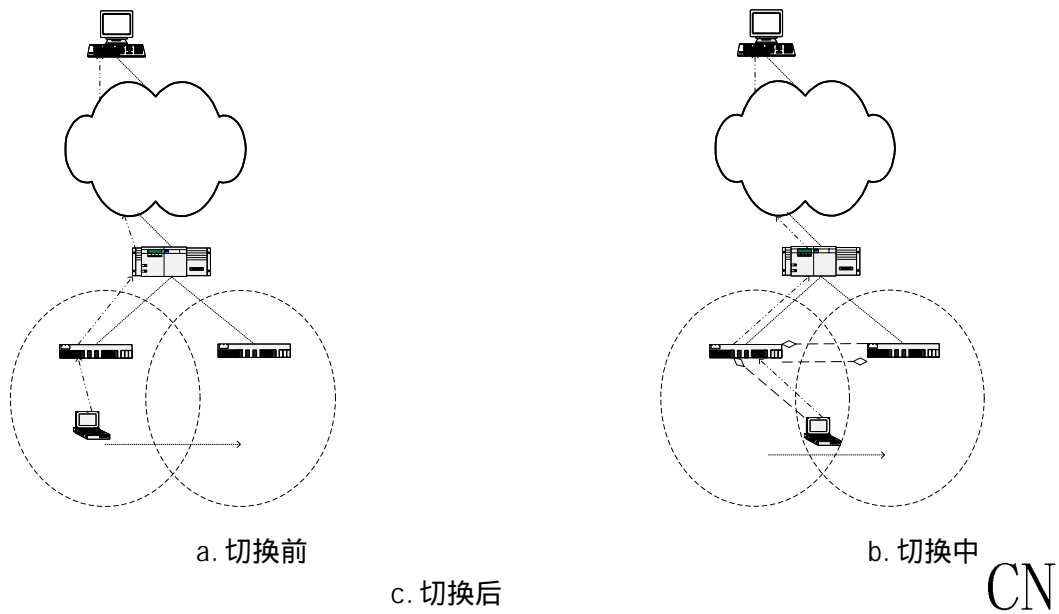


图 3-1 MN 由 PAR 子网切换到 NAR 子网时的消息传递

当 CN 作为实时业务流的发送端时,情况与 HMI Pv6 QoS 模型中的操作类似。设 MN 切换前, CN 与 MN 在有 QoS 保证的条件下传输实时业务流。在实时流传输过程中, MN 发生了切换的同时, PAR 将向 MAP 发送关于 MN 的 CT 信息,包括 MN 在 NAR 子网内的转交地址和流的 QoS 信息等等。MAP 将在 MAP 至 MN 的新转交地址这条路径上提前进行局部资源预留,同样的, NAR 将作为 MN 的 RSVP 代理。由于 HMI Pv6 QoS 模型流透明的特性,这条流在 CN 到 MAP 这段路径上仍然使用原先的预留,而 MN 切换至 NAR 子网内并在 MAP 上完成新的本地转交地址 (RCOA) 的绑定后,由于 MAP 到 NAR 的预留先于 MN 正式接入 NAR 子网向 MAP 进行绑定的时间,因此很可能当 MN 正式接入 NAR 子网后, CN 到 MN 已经能够继续有 QoS 保证的实时应用。

6. 性能分析

这一章我们将比较移动节点发生切换时无 CT 支持 MIPv6 和无 CT 支持的 HMIPv6 和有 CT 支持的 HMIPv6 三种的情况下,实时业务流得到有保证服务的延迟状况。

切换时实时业务流得到有保证服务的延迟 T 可以分为转交地址绑定更新的延迟 T_{bu+ba} 以及资源预留延迟 $T_{path+resv}$ 两部分。

在普通无 CT 支持的 MIPv6 模型下, T_{bu+ba} 是 MN 向 CN 以及向 MN 的家乡代理 (HA) 发送绑定更新和接受应答时间,这个时间是由 CN 与 MN、HA 与 MN 的远近来决定的。第二项 $T_{path+resv}$ 是指 MN 向 CN 发送 path 消息和接收 resv 消息进行资源预留的时间,直到 MN 收到 resv 消息之后 MN 才能够确定从 MN 到 CN 的新路径上是否获得了资源预留。因此在普通无 CT 支持的 MIPv6 模型下切换时实时业务流得到有保证服务的延迟可以表示为:

$$T = T_{CN}(bu+ba) + T_{CN}(path+resv)$$

在无 CT 支持的 HMIPv6 模型下, MN 与域外通信时使用域内切换时不改变的域内转交地址 RCOA 作为 MN 的地址,因此在域内的移动只需要向 MAP 进行局部绑定,无需向 CN 及 HA 发送绑定更新。因为 MAP 与 MN 的距离较 CN 与 MN 的距离近很多,这样 T_{bu+ba} 的时间就大大的缩短了。对于第二项预留延迟,如前所述 HMIPv6 模型具有实现流透明的特性,因此我们只需要向 MAP 进行资源的局部更新就可以了,因此在无 CT 支持的 HMIPv6 模型下前述公式可以表示为:

$$T = T_{MAP}(bu+ba) + T_{MAP}(path+resv)$$

在有 CT 支持的 HMIPv6 模型下,除了有无 CT 支持的 HMIPv6 模型的特性之外,因为

切换后无需 MN 再作为 QoS 的发起者 ,重新进行 QoS 的预留 ,而是让 NAR 作为 MN 的 RSVP 代理向 MAP 作资源的局部预留 ,即 MN 的局部绑定更新和局部资源预留可以并行进行。这样前述公式可以进一步简化为 :

$$T = T_{MAP}(bu+ba) , T_{MAP}(path+resv)$$

综上所述 ,采用有 CT 支持的 HMIPv6 模型可以显著减少移动节点切换时实时业务流得到有保证服务的延迟 ,使得实时业务流尽快获得有保证的服务 ,降低延迟抖动和丢包的可能性 ,从而保证了移动节点上实时业务流更好的服务质量。

7. 总结与展望

本文探讨了在移动 IPv6 环境下利用 context transfer 如何更好的实现移动节点切换时实时业务的有保证服务。Context transfer 是 IETF seamoby 工作组的新颖议题 ,这里的 context ,除了 QoS 之外 ,AAA 认证、头标压缩等信息 ,都可以作为被转移的内容 ,目的旨在重用 MN 在之前的接入路由器上已建立好的信息 ,使 MN 在切换时无需自己参与繁复的协议交换 ,重新配置这些信息 ,减小了重新配置的开销。移动 IPv6 QoS 的 Context transfer 还可以与 Fast handover 等其它机制结合起来 ,以实现移动节点实时业务流的无缝切换。如何提供 AAA 的上下文转移机制以及 CT 与其它机制的结合将是下一阶段研究将要解决的问题。

参考文献

- [1] J. Kempf, RFC 3374 Problem Description: Reasons For Performing Context Transfers Between Nodes in an IP Access Network September 2002.
- [2] J. Loughney, M. Nakhjiri, Context Transfer Protocol INTERNET-DRAFT June 2003.
- [3] D. Johnson, C. Perkins, Mobility Support in IPv6 INTERNET-DRAFT, June 2003.
- [4] Hesham Soliman , Claude Castelluccia Hierarchical Mobile IPv6 mobility management ,INTERNET-DRAFT, October 2002 .
- [5] 5 . R. Braden , L. Zhang, RFC 2209 Resource ReSerVation Protocol (RSVP) -- Version 1 Message Processing Rules, September 1997