第34卷 第17期 计 算 机 工 程 2008年9月
Vol.34 No.17 Computer Engineering September 2008
·安全技术· 文章编号：1000—3428(2008)17—0123—04 文献标识码：A 中图分类号：TP393

一种新型的复合式NAT防护系统的实现机制

司 靓[1,2]，李昀晖[1]，郜 帅[1]
(1. 北京交通大学电子信息工程学院，北京 100044；2. 中国人民解放军北京军区，北京 100041)

摘 要：提出一种基于可编程网络处理器IXP2400和GP-CPU的NAT/NAPT的实现方案，设计与实现了基于两片IXP2400和GP-CPU组成的具有安全防火墙功能的NAT防护系统。针对该NAT防护系统进行了性能分析，能够支持六十多万并发TCP/UDP的连接容量与全线速为2 Gb/s以太网连接速率，实现了网络地址复用，提高了NAT/NAPT的操作速度，克服了传统NAT实现方案中的性能瓶颈。
关键词：网络处理器；网络地址转换技术；网络地址与端口转换；防火墙；功能模块

Implementation Mechanism of Novel Compound NAT Firewall System

SI Liang[1,2], LI Yun-hui[1], GAO Shuai[1]
(1. School of Electronics and Information Engineering, Beijing Jiaotong University, Beijing 100044;
2. Beijing Military District of PLA, Beijing 100041)

【Abstract】This paper puts forward an implementation scheme, which is called Network Address Translation(NAT)/Network Address Port Translation(NAPT) based on programmable Network Processor(NP) IXP2400 and GP-CPU. Meanwhile, the NAT firewall system with firewall function, containing a pair of Intel IXP2400 and GP-CPU, is designed and implemented. And the performance analysis of the NAT Firewall system is made, which can support more than six hundred thousand of concurrent TCP/UDP sessions and sustain the full line rate on two Gigabit Ethernet links. In addition, the NAT Firewall system can successfully achieve the multiplexing of network address, effectively improve the performance of NAT/NAPT processing and overcome the bottleneck of performance in traditional implementation of NAT.
【Key words】Network Processor(NP); Network Address Translation(NAT) technology; Network Address Port Translation(NAPT); firewall; microblock

1 概述

由于现代通信网络对高性能和灵活性的要求，网络处理器作为一种新兴的下一代网络设备的核心，它吸收综合了通用处理器(GP-CPU)的完全可编程特性和专用处理器(ASIC)高速处理能力的优点[1]，并兼顾了网络设备功能的灵活性和性能的强大性，提高了网络服务的处理能力和高线速处理性能。IPv4在现代通信网络方面的应用较为广泛，大规模的突发式变革使得技术升级成本过大。因此，网络地址转换(Network Address Translation, NAT)技术是IETF提出的有效解决IPv4面临的网络地址枯竭问题的方案之一，是为提高网络地址利用率、缓解IPv4 地址枯竭压力而采取的一种有效 策略。

本文提出了一种基于IXP2400和GP-CPU的NAT防护系统的实现方案，该方案实现了网络安全防护与网络地址复用相结合的多处理功能，实现了基于有效的全局地址或用户配置的饱和度来执行NAT模式(包括静态NAT与动态NAT)与网络地址与端口转换(Network Address Port Translation, NAPT)模式的动态切换功能，同时提高了NAT/NAPT的地址转换能力，克服了传统的NAT网络应用方案中数据帧头处理时间过长，很难保持在NPs上一定的线速。

2 NAT防护系统的主要组件与核心技术

网络处理器IXP2400是用来执行数据处理和转发的高速可编程处理器。IXP2400 硬件体系结构的详细描述可参考文献[1-2]。

NAT具体可分为静态NAT、动态NAT、NAPT这3种类型，分别用来支持IP地址转换以及IP地址与TCP/UDP端口数据转换[3]。

防火墙分为2大类：包过滤防火墙和应用代理防火墙(应用层网关防火墙)。本文Ingress IXP2400采用的是包过滤型防火墙。

3 NAT防护系统的设计与实现机制

3.1 系统硬件架构的设计与实现

NAT防护系统设计的最终目标是要通过基于IXP2400和GP-CPU来实现具有安全防火墙功能的NAT/NAPT的应用方案。该系统的硬件体系结构如图1所示，其中，该系统中基于Egress IXP2400的NAT防护子系统，是实现整个NAT系统核心功能的重要组成部分，也是整个NAT系统取得高性能的关键，其架构如图2所示。在图2中，阴影部分表示各功能模块所对应的IXP2400的硬件。

基金项目：国家自然科学基金资助项目(60473001)；Intel“IXA大学合作计划”基金资助项目
作者简介：司 靓(1981－)，男，硕士研究生，主研方向：IP网络技术，IPv6的路由理论与技术；李昀晖，硕士研究生；郜 帅，讲师、在职博士研究生
收稿日期：2007-10-20 **E-mail：**fangaoren@163.com

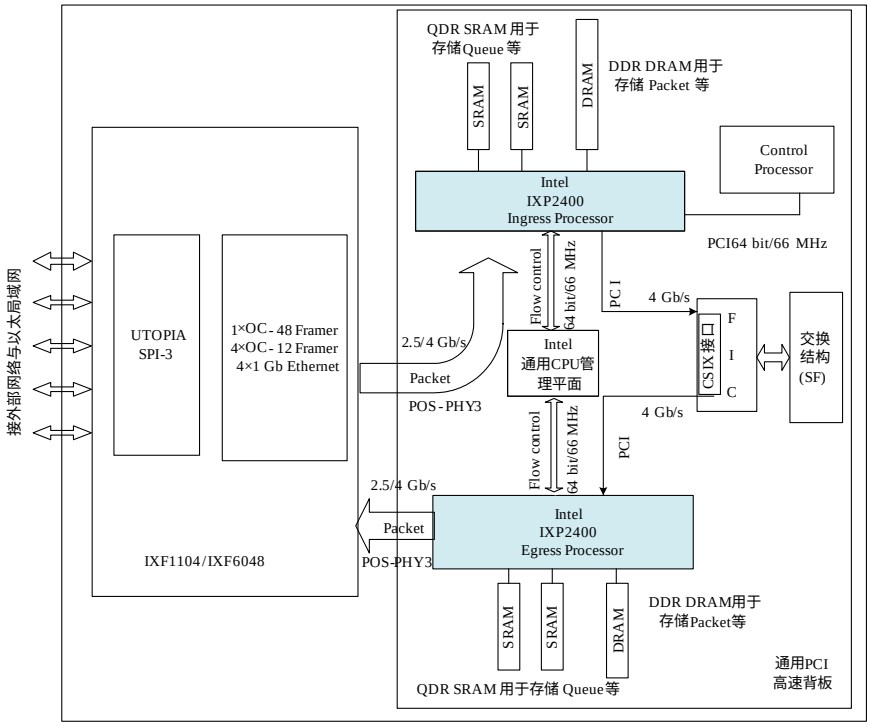

图 1　系统的硬件体系结构

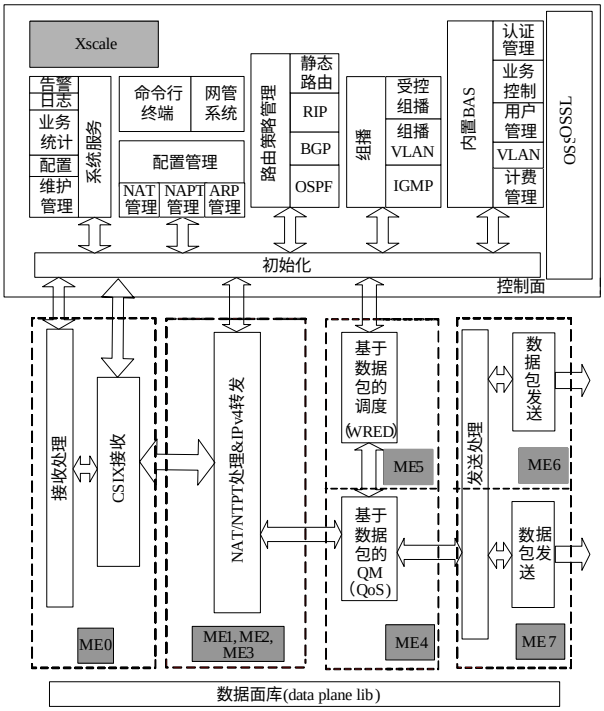

图 2　基于 Egress IXP2400 的 NAT 防护子系统框架

该系统的实现平台是由两片集成了时钟频率为 600 MHz 的IXP2400 板卡、通用PCI高速背板、一片集成有GP-CPU的主板(即系统的管理平面)、运行嵌入式Linux实时多任务操作系统、IXF专用网络设备接口卡以及Intel IXA SDK4.2[4] 及相关的构建框架(framework)组成。

基于IXP2400 和GP-CPU的NAT防护系统能够提供线速 4 Gb/s的以太网实现方案[5]，并同时支持IPv4 的单播转发。笔者通过Intel IXA SDK4.2 编辑修改相关的微代码，使Ingress NP中的IPv4 单播转发功能模块转变成第 2 到第 4 层数据包过滤功能模块；同时编辑Egress NP中相关的微代码，并映射到其数据平面中创建一个NAT/NAPT功能模块，从而实现了具有安全防火墙功能的NAT防护系统的应用方案。

3.2　NAT 防护系统的软件模块设计与实现机制

3.2.1　Egress NP 软件模块设计与实现机制

基于Egress IXP2400 的NAT防护子系统数据面的数据处理流程如图 3 所示。笔者把Egress IXP2400 的 8 个微引擎分别对应地分成具体的功能模块[6]，它们包括：CSIX RX模块，NAT/NAPT&IPv4 转发处理功能模块，Queue Manger模块，Packet Scheduler模块，以及Packet TX模块。在设计方案中，接收、队列管理及调度这 3 个功能模块各占用一个微引擎，而数据包的发送模块要占用 2 个微引擎来发送数据包，其余 3 个微引擎用于NAT/NAPT的数据处理与IPv4 转发处理阶段。在NAT/NAPT&IPv4 数据转发处理阶段中，微引擎的分配使用是通过微引擎内部的指令执行周期与时延周期的数据分析来决　定的。

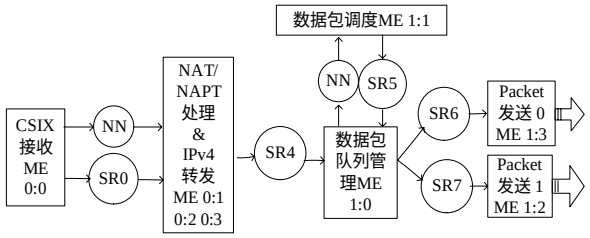

图 3　Egress NP 数据面的数据处理流程

在图 3 中，ME *x*:*y* , *x*=ME Cluster Number, *y*=ME Number, SR: Scratch Ring, NN: Next Neighbor Ring。

在Egress IXP2400 中NAT/NAPT的数据处理的详细模块如图 4 所示。在 Xscale Core(即控制面)与微引擎之间的通信过程中，有 2 个 Scratch Rings(SR1 和 SR2)按照优先排序的方式处理数据并控制数据包。作者所设计的 NAT 功能模块既可以通过手动配置运行静态 NAT 模式，同时也可以基于数据包运行动态 NAT 模式。

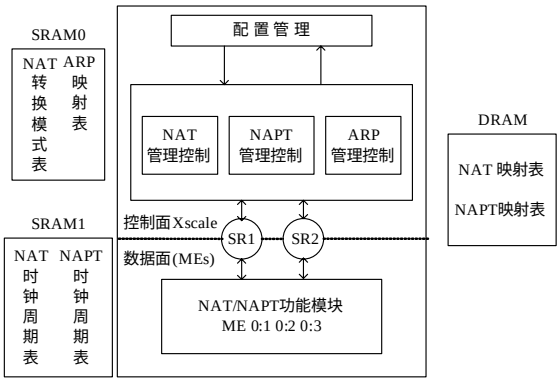

图 4　NAT/NAPT 的数据处理的详细模块

该 NAT 防护系统控制 NAT 处理模式与 NAPT 处理模式的转换，这种控制机理的实现是基于有效的全局地址或用户

配置的饱和度，确保内部网络用户对外部网络不间断的访问，并能够支持更多的网络服务。所以，Xscale Core 与 NAT/NAPT MEs(Fig.4)之间的通信是必不可少的，因为 Xscale Core 管理控制网络地址/端口映射表(NAT/NAPT Mapping Table, NMT)而 NAT/NAPT MEs 则是根据 Xscale Core 所管控的 NMT 中所提供的信息来对数据包进行处理的。在 NAT 防护子系统中，NAT/NAPT 模式的转换是根据 SRAM0 中存储的 NAT 转换模式表来实现的，见图 4。在 NAT 防护子系统中运用这种模式转换方案是把模式转换带来的数据传输中断的影响降到最低限。

NAT/NAPT 映射算法设计：在 NAT 中，利用 TCP 端口号区分、识别共享同一个外部 IP 地址的多台内部主机。因此，可以有多种映射算法对源 TCP 端口进行运算，以及对目的 TCP 端口进行逆运算。为了保证一一映射、信息不丢失，并且简单易行，遵循以下 2 个原则来选取映射关系：

(1)由映射得到的替换值中应包含内部主机的特征值；

(2)该映射最好可以通过简单运算来实现。

为此，本文采用如下映射算法：

$$Vm=\mathrm{mod\_64}(Vo)+M\times(low8(S)-1) \quad (1)$$

$$Write(Vm,\ Vo) \quad (2)$$

$$Vq=index(Vm) \quad (3)$$

$$Vr=\mathrm{mod\_64}(Vm) \quad (4)$$

$$Vo=read\{BA+LEN\times Vq+Vr\} \quad (5)$$

$$S=\mathrm{COMM\_IP}+Vq+1 \quad (6)$$

其中，*Vm* 表示 TCP 端口经映射后得到的替换值；*Vo* 表示 TCP 端口的原始值；*S* 表示内部主机的 IP 地址；*M* 表示内部主机特征值的单位，是一个常量值，取值为 64；*Vq*, *Vr* 为中间变量；*BA* 为存储区基地址，是一个常量值，取值为 0；*LEN* 为给每一台内部主机分配的存储区的长度，是一个常量值，取值为 32；COMM_IP 表示所有内部 IP 地址的共同部分；mod_64(*x*)表示对 *x* 进行模 64 运算，可通过将 *x* 对应的二进制数的高位屏蔽，保留低 6 位来实现；*low*8(*x*)表示取 IP 地址 *x* 的低 8 位；*write*(*y*, *x*)表示将变量 *x* 写入地址为 *y* 的存储单元中；*index*(*x*)表示获取 *x* 的特征值，当特征值单位 *M* 选取为 64 时，将 *x* 对应的二进制数简单地右移 6 位便可得到相应的特征值，如 *index*(156)=2；*read*{*x*}表示读取地址为 *x* 的存储单元的值。

3.2.2 Ingress NP 软件模块设计与实现机制

基于Ingress IXP2400 的NAT防护子系统数据面的数据处理流程如图 5 所示。为了保持数据处理达到 2 Gb/s的全线速，防火墙功能模块与流处理模块基于每个微引擎在处理数据包时，都要求在连续数据包的间隙内进行处理[7]。笔者将防火墙模块的功能映射到微引擎上来执行，是通过估算有效指令执行周期与I/O时延周期来实现的。Ingress IXP2400 的 8 个微引擎分别对应地分成具体的功能模块，主要包括：Packet_RX 模块，以太网解封装及分类处理模块，防火墙功能模块，流处理模块，Queue Manger模块，Packet Scheduler模块，以及 CSIX_TX模块。在以太网接口层，最小的链路层数据帧的大小为 84 Byte(最小的以太网数据帧是 64 Byte，再加上帧间的数据帧头是 20 Byte)。当线速为 2 Gb/s时，如果数据包的大小是 84 Byte，数据包的吞吐量将达到每秒钟处理 2.976 百万个数据包。其中，连续 2 个 84 Byte的数据包的间隙为 336 ns。

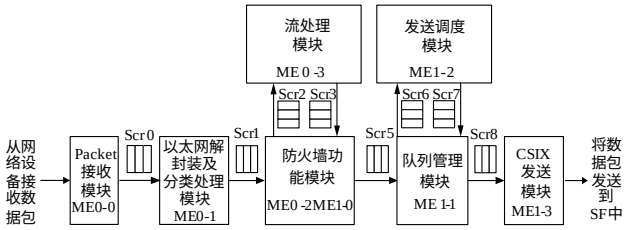

图 5 Ingress NP 数据面的数据处理流程

帧间间隙可以被设置成子系统中处理器的时钟。由于 NAT 防护子系统是基于 IXP2400 来实现的，而 IXP2400 的时钟频率为 600 MHz，因此该处理器的时钟应为 1.67 ns。

整个数据处理流程(见图 3)中的任何一个处理进程都必须能够在特定周期(符合这种周期算法)内完成所要求的所有接收到的数据包的处理，而且，每个处理进程都应该能够达到每 202 个周期处理一个新的数据包的处理速度，以支持 2.0 Gb/s 的全线速。

微引擎对数据包的处理包括对内存储器和外存储器，如：Scratch Memory, SRAM, DRAM等的访问。总的来说，访问存储器的时延与相邻数据包之间的间隙是近似的，例如：SRAM 存储器的访问时延大约是 150 Cycles，而DRAM存储器的访问时延大约是 250 Cycles~300 Cycles[8]。每个微引擎内部有 8 个硬件线程(context)，采用多线程交换(multi-threading)技术时，由相应的硬件结构提供支持，由软件指令来控制，可以将存储器的访问时延影藏在指令执行周期的后面，因此，提供一个I/O时延周期相当于数据包传输速率的 8 倍，也就是说，一个I/O时延周期总计为 8 乘以 202，即 1616 Cycles(即 8 倍的以太网传输模式上的IPv4 数据包最短到达时间)，充分发挥了微引擎的MIPS性能，提高了系统的并行处理能力。

4 NAT 防护系统的性能分析

这项仿真测试运用了 IXP2400 Developer Workbench 专用软件开发平台，通过编译微代码来实现数据的仿真。在测试中，Workbench 编译器使用的参数设置都是默认值。

4.1 并发连接容量与最大连接速率/最大会话速率

该性能测试为系统的 TCP 会话处理性能，峰值速率和并发连接性能的测试集合。在性能测试之前，ARP 的映射条目与 NAT/NAPT 的转换条目采用动态配置的形式，其测试的结果如下：(1)关于并发连接容量的测试。该测试所建立的总的用户连接数是从 10 万~60 万之间，在所有 5 种不同情况所产生的变化可以忽略不计的情况下，该系统可以达到 100%的连接容量，其建立该连接容量所需的平均时间为 172.5 ms。根据测试结果，其结论是该 NAT 防护系统完全能够支持 60 多万并发 TCP/UDP 的连接容量；(2)关于最大连接速率与最大会话速率的测试。该测试是通过固定的数据帧长度(64 Byte)与 50 个客户机和一个服务器来完成的。测试结果表明了所设计的 NAT 防护系统完全能够以全线速处理数据包。

4.2 平均转发速率与时延分析

图 6 展示了当所有数据包的 NMT 查找采用一次地址冲突量(N-depth)匹配的方式时相应的平均转发速率的变化曲线。在这项仿真测试过程中，数据包以最小长度 64 Byte 以太网帧生成于每个网络地址，而且 NMTs 的查找模式以 N-depth 匹配方式被静态设置，所有的数据包都汇成一股数据流，执行 NAT/NAPT 处理的微引擎中的每个线程都会同时去访问 NMT，这样就会引起 DRAM 存储器访问的冲突。当哈希冲突的次数不断增加时，DRAM 存储器的负担就会增大而

且 DRAM 存储器的访问时延也会大大增加，再加上等待数据读入线程的开销，执行 NAT/NAPT 处理的微引擎中线程将会产生过度的时延。图中曲线的走势说明了随着 NMT 查找匹配的一次地址冲突量的增加，将会导致平均转发速率逐渐地降低。笔者将所设计的 NAT 防护系统的时延与不具有 NAT/NAPT 功能仅只提供简单的数据包转发功能的系统的时延进行了对比，如图 7 所示。以上测试结果表明，基于 NP 的 NAT/NAPT 功能模块需要对数据包头进行深度地处理。

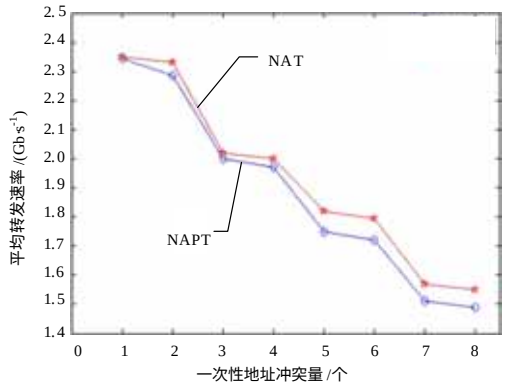

图 6　平均转发速率曲线图

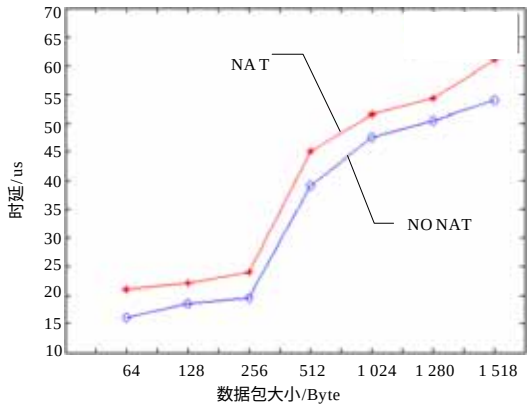

图 7　时延曲线图

5　结束语

本文提出了一种基于 IXP2400 和 GP-CPU 的 NAT 防护系统实现方案，运用 IXP2400 网络处理器来构架安全防火墙功能的 NAT 防护系统的硬件平台，可以充分发挥 IXP2400 网络处理器强大的数据处理性能和灵活的处理功能，使得 NAT 防护系统能够同时满足网络宽带化和综合化的要求，成功地实现了地址复用，并且微引擎的多线程交换技术与并行处理能力有效地提高了系统的 NAT/NAPT 模块数据处理的性能，解决了传统 NAT 实现方案中的性能瓶颈。

在下一步的工作中，笔者计划对目前系统防火墙功能与 NAT 功能相结合的多处理功能模式进行扩展，增加 IPv6 数据处理、IPv4-IPv6 之间相互转换的功能(即 NAT-PT 技术)；要创建一个新的哈希函数来减少哈希冲突，重新设计哈希表的数据结构来提高存储器的利用率，最小化存储器的访问时延。

参考文献

[1] Intel. Intel IXP2400 Network Processor Hardware Reference Manual [M/CD]. [S. 1.]: Intel Press, 2004.

[2] 张宏科, 苏　伟, 武　勇. 网络处理器原理与技术[M]. 北京: 北京邮电大学出版社, 2004: 18-138.

[3] Stevens W R. TCP/IP Illustrated, Volume 2: The Implementation[M]. Beijing: China Machine Press, 2000: 63-560.

[4] Intel. IXP2400 and IXP2800 Network Processor: Programmer's Reference Manual[M/CD]. [S. 1.]: Intel Press, 2005.

[5] Intel. Internet Exchange Architecture Software Building Blocks Reference Manual[M/CD]. [S. 1.]: Intel Press, 2003.

[6] Intel. Internet Exchange Architecture Software Building Blocks Developer's Manual[M/CD]. [S. 1.]: Intel Press, 2005.

[7] Intel. Building Blocks Apps Design Guide[M/CD]. [S. 1.]: Intel Press, 2003.

[8] Lakshmanamurthy S, Liu K, Pun Y, et al. Network Processor Performance Analysis Methodology[J]. Intel Technology Journal, 2002, 6(3): 19-28.

（上接第 92 页）

[2] Kamerman A, Monteban L. WaveLAN II: A High-performance Wireless LAN for the Unlicensed Band[J]. Bell Labs Technical Journal, 1997, 2(3): 118-133.

[3] Holland G, Vaidya N, Bahl P. A Rate-adaptive MAC Protocol for Multi-hop Wireless Networks[C]//Proceedings of ACM Mobicom. [S. l.]: ACM Press, 2001.

[4] Sadeghi B, Kanodia V, Sabharwal A. OAR: An Opportunistic Auto-rate Media Access Protocol for Ad Hoc Networks[J]. Wireless Networks, 2005, 11(1/2): 39-53.

[5] Bianchi G. Performance Analysis of the IEEE 802.11 Distributed Coordination Function[J]. IEEE Journal on Selected Areas in Communications, 2000, 18(3): 535-547.

[6] 段中兴, 张德运. 多速率无线局域网的速率自适应算法[J]. 计算机工程, 2007, 33(8): 33-35.

（上接第 101 页）

一定程度上解决上述的问题。而且方案的伸缩性较好，适用范围很广，无需增加其他的设备，仅在原有架构的基础上增加链路便可达到很好的优化。虽说系统复杂性相对于 HMIPv6 来说增加了一点，但在丢包率、绑定更新时延、切换速度等方面都会有较好的提高，可以明显提高 HMIPv6 系统的性能以及 MAP 选择的最优化。

参考文献

[1] Johnson D, Perkins C E, Arkk J. Mobility Support in IPv6[S]. RFC 3775, 2004-06.

[2] Soliman H, Catelluccia C, Malki K E, et al. Hierarchical MIPv6 Mobility Management (HMIPv6)[Z]. draft-ietf-mobileip-fast-mipv6-06.txt, 2003.

[3] 於时才, 朱宏涛, 宋　健, 等. HMIPv6 中的 MAP 发现协议的研究与改进[J]. 计算机应用, 2006, 26(1): 5-7.

[4] 方　波, 宋俊德. 一种多连接度的多层移动 IPv6 网络架构[J]. 计算机工程, 2005, 31(21): 90-92.

[5] 刘缵武. 应用图论[M]. 长沙: 国防科技大学出版社, 2006.