IPv6 隧道代理机制研究

吴贤国[1,2], 刘 敏[1,2], 李忠诚[1]
(1. 中国科学院计算技术研究所, 北京 100080; 2. 中国科学院研究生院, 北京 100049)

摘 要: IPv6 隧道代理机制是一种重要的 IPv4/IPv6 过渡技术, 但存在不足和不够完善的地方. 首先, 它不支持 NAT(网络地址翻译)用户; 其次, 该机制没有为实现隧道服务器负载均衡提供一种具体的调度方案. 本文修改了代理服务器和隧道服务器的功能, 解决了其不支持 NAT 用户的问题; 并提出一种加权最少隧道调度算法, 仿真实验表明它比通用调度算法更有效地实现了隧道服务器负载均衡.

关键词: 隧道代理; IPv6; 过渡; 网络地址翻译; 负载均衡
中图分类号: TP393 文献标识码: A 文章编号: 0372-2112 (2007) 02-0354-04

Research on IPv6 Tunnel Broker

Wu Xiang-guo[1,2], Liu Min[1,2], Li Zhong-cheng[1]
(1. *Institute of Computing Technology, Chinese Academy of Sciences, Beijing* 100080, *China*;
2. *Graduate School, Chinese Academy of Sciences, Beijing* 100049, *China*)

Abstract: IPv6 Tunnel Broker is one of important mechanisms of transition from IPv4 to IPv6. However, This mechanism does not support NAT users, also not specific a way to balance the load of tunnel servers. The function of Broker Server and Tunnel Server is modified to support NAT users. Meanwhile, a new dispatching method is proposed for balancing the load between tunnel servers. Simulation result shows that the method has better performance than other general dispatching methods have.

Key words: tunnel broker; IPv6; transition; NAT; load balancing

1 引言

IPv6 隧道代理[1]是 IETF 制定的一种 IPv4/IPv6 过渡机制, 主要功能是为用户提供一种简化的隧道配置方法, 利用隧道实现用户和 IPv6 网络的互连.

由于隧道代理采用 IPv6-in-IPv4 封装方式, 而目前绝大部分 NAT[2]都不支持这类数据包(协议类型为 41)的转发, 因此隧道代理不支持 NAT 用户接入 IPv6 网络. 而目前唯一专门面向 NAT 用户设计的 Teredo[3]协议存在不支持对称类型的 NAT、不能为用户分配永久固定的 IPv6 地址以及安全性较差等不足. 所以, 有必要针对 NAT 问题修改隧道代理机制.

另外, 隧道代理采用客户端-服务器结构, 当用户增多时, 需要在网络中部署多个隧道服务器. 因此, 有必要引入负载均衡机制来为用户选择合适的服务器进行接入, 以尽量避免有些服务器负载过重、而另一些则资源过剩的情况, 从而提高系统稳定性. 然而, 在描述隧道代理机制的 RFC3053 文档中, 虽然提到采用负载均衡的想法, 但没有一个具体的调度方案.

本文对隧道代理机制做了改进, 解决了它不支持 NAT 用户的问题; 在此基础上提出一种加权最少隧道调度算法, 通过仿真比较了它和通用调度算法的性能, 仿真结果表明 WLT 更有效地实现了服务器负载均衡.

2 隧道代理机制的改进

2.1 系统描述

系统结构如图 1 所示, 客户端和服务器之间的隧道采用 IPv4 头部和 UDP 头部双重封装的方式来传输 IPv6 数据包. 由于所有的 NAT 设备都支持 UDP 报文的转发, 因此允许隧道主体上存在任何类型和任意数量的 NAT 设备.

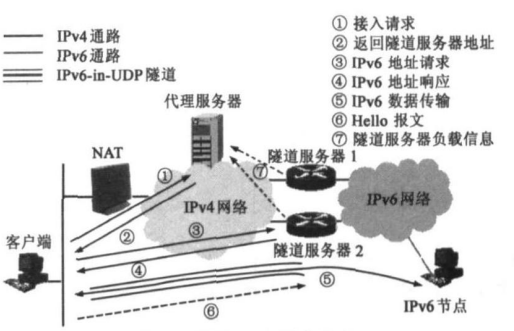


图 1 隧道代理系统结构

除了改变隧道的封装方式以外, 隧道代理的主要改进还体现在对代理服务器和隧道服务器功能的修改上. 修改后的

收稿日期: 2005-12-26 修回日期: 2006-07-20
基金项目: 国家自然科学基金(No. 60273021, No. 90604016)

代理服务器不再负责 IPv6 地址分配以及隧道的创建和管理, 它的主要功能是动态监测隧道服务器的运行状况, 然后根据调度算法为用户指定合适的隧道服务器. 地址分配和隧道的创建管理由隧道服务器负责. 进行功能移植的主要原因是为了降低接入用户增多时, 代理服务器因管理隧道而成为瓶颈的可能性.

客户端接入 IPv6 网络的过程如下: 首先向代理服务器发送接入请求; 代理服务器根据调度算法选择一个合适的隧道服务器并向客户端返回它的 IPv4 地址, 在执行调度算法的过程中需要用到隧道服务器的负载信息, 因此隧道服务器必须周期性的向代理服务器发送负载信息报文; 客户端得到隧道服务器的地址后向其发送 IPv6 地址请求报文; 隧道服务器从该报文中获得客户端的隧道参数, 然后向客户端返回 IPv6 地址响应报文; 客户端根据报文内容配置 IPv6 地址并建立隧道, 完成接入过程.

2.2　隧道参数的存储方式

通常有两种存储隧道参数的方式, 一是将它嵌入到客户端的 IPv6 地址中, 隧道服务器根据 IPv6 地址获得隧道参数, 从而实现自动封装, 6to4[4]、ISATAP[5] 和 Teredo 等都采用这种方式; 二是在隧道服务器上建立客户端 IPv6 地址和隧道参数之间的映射关系, 然后服务器根据 IPv6 地址查表得到隧道参数, 进而完成对数据包的封装.

如果采用第一种方式, 由于作为隧道参数的映射地址和映射端口不是固定的, 客户端每次和隧道服务器初始化通信时往往被 NAT 转换成不同的映射地址或映射端口, 所以通常情况下客户端总是获得和上次不同的 IPv6 地址. 在地址经常变化的情况下, 另一方很难主动向 NAT 用户发起连接, 使用户丧失了端到端的通信优势.

为了使 NAT 用户可以获得固定不变的 IPv6 地址, 本文采用第二种方式, 隧道服务器通过建立客户端 IPv6 地址和隧道参数之间的映射关系来实现对 IPv6 数据包的封装功能. 如图 2 所示, 即使客户端隧道参数发生变化, 隧道服务器只要重新设置映射关系即可, 客户端的 IPv6 地址可以保持不变. 在这种方式下, 隧道服务器可以灵活引入各种配址机制, 为用户分配永久固定的 IPv6 地址.

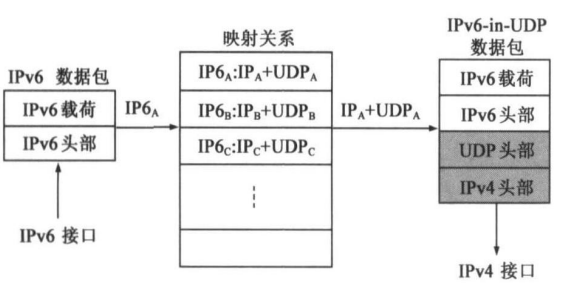


图 2　基于映射关系的封装

2.3　UDP 会话的维持和隧道管理

采用 IPv6-in-UDP 隧道会引起一个问题, 因为它是基于 UDP 会话的, 如果隧道长时间没有数据流, NAT 设备会删除 UDP 会话, 导致隧道失效. 要避免出现这种情况, 必须保证客户端和隧道服务器在一定时间内就有数据交互发生. 本文规定客户端接入 IPv6 网络后, 以一定的时间间隔不停地向隧道服务器发送 Hello 报文(没有 IPv6 载荷的 IPv6-in-UDP 报文).

由客户端发送 Hello 报文的另一个目的是解决隧道管理问题. 隧道服务器为每个隧道设置一个计时器, 收到 Hello 报文后查找对应的隧道, 然后将它的计时器重新置零, 一旦计时器超时, 就删除该隧道. 通过这种方式, 使隧道服务器上维护的都是处于活动状态的隧道, 避免服务器因维护无用的隧道而消耗系统资源.

2.4　和 Teredo 协议的比较

经修改后的隧道代理机制和另一支持 NAT 用户的 Teredo 协议相比, 主要有 3 个优点. 一是可以为用户分配永久固定的 IPv6 地址, 而 Teredo 为用户分配的地址是经常变化的; 二是支持所有类型的 NAT, 而 Teredo 不支持对称类型的 NAT; 三是隧道服务器是有状态的, 可以根据状态信息也就是映射关系对转发的数据包进行合法性检查, 因此具有比 Teredo 更好的安全性.

3　代理服务器调度方案

3.1　负载指示

前面提到, 代理服务器根据隧道服务器动态反馈的负载信息对客户端的接入请求进行调度. 负载反馈机制在早期的负载均衡方法中就得到了普遍的应用, 对于不同的系统, 负载的指示也不同, CPU 利用率、内存利用率、磁盘使用率、进程数等信息都可以用来指示负载.

隧道服务器的主要功能是转发数据, 突发性是数据流的一个重要特征, 导致 CPU 利用率或带宽利用率的波动十分剧烈, 使得某一时刻的采样值不能准确反映下一个采样周期内服务器的负载状况, 因此采用 CPU 利用率或带宽利用率来指示负载是不合适的. 本文采用隧道数目来指示负载, 主要依据有两点, 一是隧道服务器维护的都是处于活动状态的隧道, 管理隧道和通过隧道转发数据都需要消耗系统资源, 因此隧道数目能够反映系统的负载状况; 二是服务器上的隧道数目相对流量来讲是比较稳定的, 波动幅度较小, 更能体现采样前后一段时间内服务器的负载状况. 计算服务器的负载指示时, 借鉴文献[6]的算法对下一个采样周期内隧道数目的变化情况进行预测.

假设 T 表示服务器当前的隧道数目, $I(j)$ 表示服务器最近第 j 个采样周期内新增的隧道个数. 则服务器的负载指示 C 按下式计算:

$$C = T + 0.5 \cdot \sum_{j=1}^{K} (1/G) I(j) e^{-j/2} \qquad (1)$$

其中 G 是归一化系数, $G = \sum_{j=1}^{K} e^{-j/2}$.

3.2　加权最少隧道调度

加权最少隧道调度算法(以下简称 WLT) 考虑到各个服务器不同的处理能力, 把新的用户请求指派到单位(处理能力) 隧道数目最少的服务器上. 为阐述方便, 不妨设 N 个服务器为 $S = \{S_0, S_1, S_2, \cdots, S_{N-1}\}$, W_i 为服务器 S_i 的默认权值, 代表服务器的处理能力, C_i 指服务器 S_i 当前的负载指示, 根据

式(1)计算获得. 新的用户请求被指派到服务器 S_m, 当且仅当下式成立:

$$(C_m \setminus \sum_{j=0}^{N-1} C_j)/W_m = \min((C_i \setminus \sum_{j=0}^{N-1} C_j)/W_i),$$
$$i = 0, 1, 2, \cdots, N-1 \quad (2)$$

其中 W_i 不为 0, $\sum_{j=0}^{N-1} C_j$ 在一轮查找中是一个常数, 所以式(2)可简化为:

$$C_m/W_m = \min(C_i/W_i),\ i = 0, 1, 2, \cdots, N-1 \quad (3)$$

由于除法所需的时钟周期较多, 所以实现中把判断条件 $C_m/W_m > C_i/W_i$ 转化为 $C_m * W_i > C_i * W_m$, 同时保证服务器的权值为 0 时不被指派.

以隧道数目来指示服务器负载的一个缺点是不能反映实际过载的情况. 对此, 隧道服务器采用报警方式向代理服务器反映过载的情况.

4 调度方案性能评估

4.1 仿真参数

本文通过仿真实验对 WLT 和几种通用调度算法如 RR(轮叫调度)、Random(随机调度)、WRR(加权轮叫调度)、LL(按最低 CPU 利用率或带宽利用率进行调度)的性能进行了比较.

表 1 仿真参数

类别	参数名称	取值(缺省)
服务器	数目	5
	异构程度	0, 20%, 35%, 50%, 65%(50%)
	平均利用率	0.52, 0.57, 0.62, 0.67, 0.72(0.67)
	系统总容量	10000
用户	数目	20000
	接入时间间隔 OFF	35 小时
	在线时间 ON	30 分钟
流量模式	OFF	Pareto 分布
	ON	Weibull 分布
	速率 V	Pareto 分布
调度算法	反馈周期	40, 80, 160, 240, 400 秒(40 秒)
	历史采样数 K	5
	报警阈值	0.96

系统仿真参数分为 4 类, 如表 1 所示. 整个网络模型由 5 个隧道服务器和 1 个代理服务器组成, 每个服务器的性能不同, 性能差异程度(以下简称异构程度)从 0 到 65% 不等, 如表 2, 括号内的数值表示每个服务器和最好服务器的性能比值. 系统总容量固定为 10000 个单位, 平均利用率从 0.52 到 0.72 不等. 用户行为参数即接入时间间隔和在线时间根据斯图加特大学校园网用户的流量统计结果设置[7], 考虑到互联网的发展, 下调接入时间间隔为 35 小时, 上调在线时间为 30 分钟. 用户数目为 20000 个. 采用 ON/OFF 模型生成用户流量[8], 其中 OFF 均值为用户接入时间间隔, ON 均值为用户在线时间. ON 时间长度服从 Weibull 分布, OFF 时间长度服从 Pareto 分布, 用户速率服从 Pareto 分布.

表 2 服务器异构程度

异构程度	服务器性能比值
20%	{1, 1, 0.8, 0.8, 0.8}
35%	{1, 0.8, 0.8, 0.65, 0.65}
50%	{1, 0.8, 0.65, 0.5, 0.5}
65%	{1, 0.8, 0.5, 0.35, 0.35}

基于离散事件模型在 Red Hat 8.0 上用 C 语言实现了仿真程序. 在每一次仿真中, 系统运行时间为 60 天. 仿真结果中给出的数据确信度为 0.95, 确信区间为均值的 ±3%.

4.2 评估方法

采用服务器最高利用率的累积频率作为性能指标. 设定一个阈值 ϑ, 服务器利用率超过 ϑ 表示负载过重. 通过观察一段时间内服务器最高利用率 U_{max} 小于 ϑ 的频率 F, 可以判断系统运行的稳定程度. $F = P/M$, 其中 M 为采样次数, P 为 U_{max} 小于 ϑ 的次数. F 值越大, 表明最高利用率不超过阈值的可能性越大, 系统出现过载的可能性越小, 稳定程度越高.

4.3 仿真结果及讨论

如图 3 所示, 横坐标表示阈值, 纵坐标表示最高利用率小于阈值的频率. 对于 Random[9]和 RR[10]算法, 阈值为 0.96 时, 累积频率分别为 0.353 和 0.346, 也就是说, 系统在约 65% 的时间里至少有一个服务器出现过载, 说明随机调度或轮叫调度不能有效地提高系统稳定性, 这主要是服务器性能存在差异的缘故. WLT 算法明显降低了系统出现过载的可能性, 并且在性能上优于 WRR[11]和 LL[12]算法, 阈值为 0.96 时, WLT 的累积频率为 0.846, WRR 和 LL 的累积频率分别为 0.745 和 0.772. 由于 WLT 采用动态反馈机制, 能够有效地将隧道按处理能力均匀分配到各个服务器上, 避免了像 WRR 算法那样因隧道的活动持续时间不一致引起服务器隧道数目不均匀的现象, 因此获得了比 WRR 更好的性能. 另外, 如上文所述, WLT 以隧道数目作为负载指示, 相比 LL 采用 CPU 利用率或带宽利用率更能准确反映服务器在一段时间内的负载状况, 并且避免了反馈值的剧烈波动, 更好的起到了动态反馈机制的调节作用.

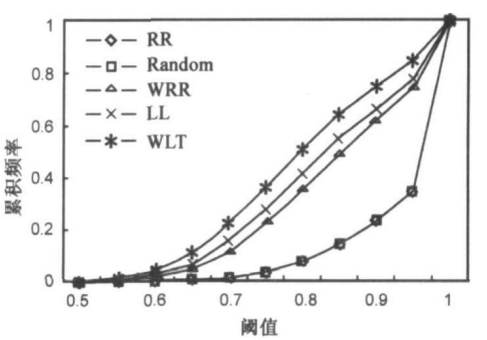


图 3 缺省参数下不同调度算法的性能比较

本文还评估了服务器平均利用率、异构程度以及反馈周期对算法性能的影响. 仿真结果表明, 在不同的情况下, WLT 相比其他算法都具有更好的性能. 限于篇幅, 不再列出数据.

5 结论

本文的主要贡献有两点, 如下:

(1) 对 IPv6 隧道代理机制进行了修改, 由隧道服务器负责 IPv6 地址的分配以及隧道的创建和管理. 采用 IPv6-in-UDP 隧道, 基于映射关系实现隧道服务器对 IPv6 数据包的封装, 并采用客户端发送 Hello 包的方式来维护和管理隧道, 解决了隧道代理机制不支持 NAT 用户的问题.

(2) 设计代理服务器调度方案,采用加权最少隧道调度算法WLT对用户的接入请求进行调度. 仿真结果表明, WLT 比通用调度算法更有效地提高了系统稳定性.

参考文献:

[1] A Durand, P Fasano, I Guardini, D Lento. IPv6 Tunnel Broker [S]. RFC 3053, 2001.

[2] K Egevang, P Francis. The IP Network Address Translator (NAT) [S]. RFC 1631, 1994.

[3] C Huitema. Teredo: Tunneling IPv6 over UDP through NATs [Z]. draft-huitema-v6ops-teredo-04. txt, 2005.

[4] B Carpenter, K Moore. Connection of IPv6 domain via IPv4 clouds[S]. RFC 3056, 2001.

[5] F Templin, T Gleeson, M Talwar, D Thaler. intra-site automatic tunnel addressing protocol (ISATAP) [Z]. draft-ietf-ngtrans-isatap-24. txt, 2005.

[6] T Lin, K Wang. An efficient load balancing strategy for scalable WAP gateways[A]. 9th International Conference on Parallel and Distributed Systems[C]. 2002. 625- 630.

[7] J Farber, S bodamer, J Charzinski. Measurement and modeling of internet traffic at access networks[A]. Proceedings of the EUNICE'98[C]. 1998. 196- 203.

[8] X Yang. Designing Traffic profiles for bursty Internet traffic [J]. IEEE Global Internet, 2002, 3(3): 2149- 2154.

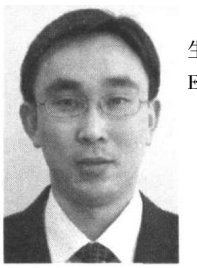
[9] D M Dias. A scalable and highly available Web server[A]. Proceedings of the IEEE International Computer Conference [C]. 1996. 85- 92.

[10] L Zhang. The Performance of Clustering Techniques for Scalable Web Servers[D]. Thesis of Master Degree, Simon Fraser University, 2002.

[11] G D H Hunt, et al. Network Dispatch: A connection router for scalable Internet services[J]. Computer Networks and ISDN Systems, 1998, 30: 347- 357.

[12] M Colajanni, et al. Analysis of task assignment policies in scalable distributed Web-server systems[J]. IEEE Transactions on Parallel and Distributed Systems, 1998, 9(6): 585- 600.

作者简介:



吴贤国　男, 1978 年生于浙江, 博士研究生, 研究方向为下一代互联网与无线网络.
E-mail: xgwu@ict.ac.cn

刘　敏　女, 1976 年生于河南, 博士研究生, 副研究员, 研究方向为移动切换与网络测量.

李忠诚　男, 1962 年生于山东, 博士, 研究员, 博士生导师, 研究方向为计算机网络和测试等.