第 33 卷 第 7 期 计 算 机 工 程 2007 年 4 月
Vol.33 No.7 Computer Engineering April 2007
·网络与通信· 文章编号：1000—3428(2007)07—0112—02 文献标识码：A 中图分类号：TP393

基于网络处理器的 NAT 优化设计

杨志义，於志勇，张 凡，沈 键
(西北工业大学计算机学院，西安 710072)

摘 要：提出基于网络处理器 IXP2400 的网络地址转换(NAT)实现方案，并对 NAT 表的存储结构、表项的创建和删除、ME 与 XScale 之间的通信机制、MAC 地址修改与校验和计算等关键技术进行优化设计，使其既具有线速处理的能力，又能灵活地配置。
关键词：网络地址转换；网络处理器；优化；IXP2400

Optimized Design of NAT Based on Network Processor

YANG Zhiyi, YU Zhiyong, ZHANG Fan, SHEN Jian
(College of Computer Science, Northwestern Polytechnical University, Xi'an 710072)

【Abstract】 This paper proposes an implementation scheme of network address translation (NAT) based on network processor IXP2400, and describes the optimized design of key technologies, including storage structure of NAT table, creation and deletion of table items, communication between ME and XScale, modification of MAC address and calculation of checksum. The proposed approach brings not only the capability of wire-speed processing, but also the flexibility of configuration.
【Key words】 Network address translation; Network processor; Optimized; IXP2400

网络地址转换(Network Address Translation, NAT)指把IP地址从一个范围(realm)映射到另一个范围，为主机提供透明路由[1]。NAT主要用于缓解IPv4 地址枯竭、屏蔽内部网络的拓扑结构、IPv4 与IPv6 之间的转换等方面。NAT的实现方法有[2]：基于硬件实现独立的NAT 设备；集成在网络防火墙中，成为其中的一个功能模块；集成在操作系统中，通过用户配置来实现NAT 功能；在边缘路由器上实现。

NAT 运行的硬件平台可以分为通用处理器(General Purpose Processor, GPP)、专用集成电路(Application Specific Integrate Circuit, ASIC)和网络处理器 (Network Processor, NP)。随着网络带宽的迅速增长和应用的多样化，GPP 在处理速度上的劣势及 ASIC 在灵活性上的不足日益凸现，硬件厂商推出了专门为网络通信应用优化设计的处理器 NP，它既有 GPP 的可编程性，又有 ASIC 的高速处理能力。NAT 对数据的处理较少，主要考验设备的转发能力，而这恰恰是 NP 的特长，所以采用 NP 开发 NAT 模块具有良好前景。

目前，基于NP实现NAT的研究较少，主要集中在NAT协议的软件结构[2]、NAT表的存储结构、端口映射算法以及特殊报文处理等方面。本文综合考虑，提出针对IXP2400 系统结构而优化的集成在防火墙中的NAT模块实现方案。

1 NAT 工作原理

NAT 将内部网络中报文的地址转换为外部合法地址，向外部网络发送，而在收到外部报文后，再将其翻译成内部私有地址，向内部网络发送，其实质就是维护一张内部私有 IP 地址与外部合法 IP 地址的映射表。NAT 有静态地址转换、动态地址转换和网络地址端口转换(Network Address Port Translation, NAPT)3 种类型。其中 NAPT 是当前应用最广泛的一种，它把不同的内部 IP 地址映射到外部网络的一个合法 IP 地址的不同端口上。如图 1 所示，内部网络主机 Host1 与外部网络服务器 Server1 进行通信，Host1 数据包的源地址和端口为 10.0.0.1:3017，通过防火墙的 NAT 模块转换，把源地址和端口映射为全局的 138.76.28.4:9，同时修改数据包中相关字段，Server1 收到数据包并响应连接，发出以 138.76.28.4:9 为目的的数据包，经过 NAT 模块转换，把目的地址和端口修改为内部网络的 10.0.0.1:3017，同时修改数据包中相关字段，传给 Host1，即完成了一次通信。

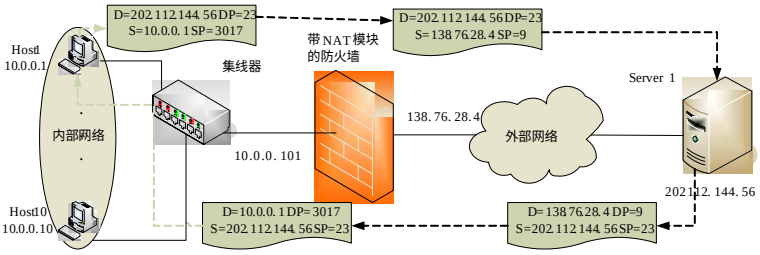

图 1 NAPT 工作过程示意图

2 IXP2400 平台简介

本文选用的IXP2400 平台[3]主要面向速率为 2.5Gbps的网络应用，具有编程环境灵活、代码可重用、处理机制性能好、易于实现快速应用等特点。IXP2400 主要包括 1 个 600MHz 的XScale核心处理器和 8 个可编程微引擎 (Microengine, ME)，每个ME含一个CRC Unit并支持 8 个硬件实现的线程。ME不仅能够访问其内部的多种存储器，还可以通过SRAM

基金项目：国家“863”计划基金资助项目(2003AA1z2100)；西北工业大学青年科技创新基金资助项目(M016213)；西北工业大学研究生创业种子基金资助项目(Z2000643)
作者简介：杨志义(1952－)，男，教授，主研方向：分布实时计算，网络化嵌入式计算；於志勇，硕士生；张 凡，讲师、在职博士生；沈 键，硕士生
收稿日期：2006-04-06 **E-mail：**yzyhb@126.com

Controller和DRAM Controller对片外的SRAM、DRAM进行访问。此外，IXP2400 包含接收和发送数据包的媒体交换接口(Media and Switch Fabric, MSF)，接收到封装有IP报文的数据链路层帧后，自动按固定大小切割成微包，然后缓存在DRAM中。Hash Unit提供 48 位、64 位、128 位的硬件散列算法。

3 关键技术的分析与设计

根据 IXP2400 系统结构的特点，让 XScale 进行初始化、参数配置、管理 NAT 表、并处理 ME 传递上来的特殊数据包(如 ICMP)等工作，让 ME 查询 NAT 表、修改数据包中相关字段、接收和发送数据包等。关键的处理流程如图 2 所示。

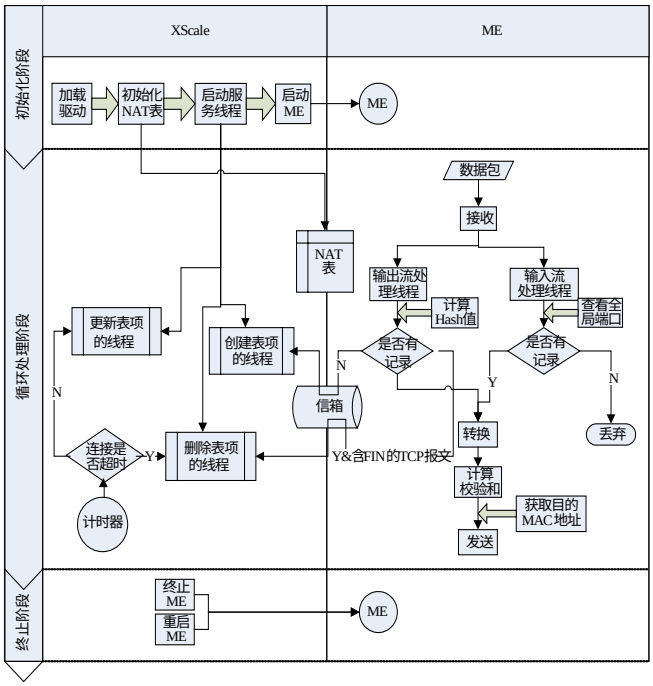

图 2 NAT 模块处理流程

3.1 NAT 表的存储结构

NAPT 的实质就是动态地维护一张记录内部 IP 和端口与全局 IP 和端口之间对应关系的映射表，即 NAT 表。NAT 表项可以动态地创建和删除，设计原则是尽量小且各字段是整字节，格式如图 3 所示。表项的主要内容是二元组(inner_ip，inner_port)。标志 flag(1 位)，1 表示有效，0 表示无效，初值为 0。timeout(7 位)是定时器，用于判断连接是否活跃。预留 RESERVED(8 位)。inner_port(16 位)是内部网络端口。inner_ip(32 位)是内部网络 IP 地址。

flag	timeout	RESERVED	inner_port
inner_ip			

图 3 NAT 表的格式

NAT 表存放于 SRAM 中，基址由可配置参数 NAT_TABLE_BASE 定义，每项长度 8 字节。所有内部网络的 IP 地址都要转换成外部网络有效的全局 IP 地址，即

global_ip = GLOBAL_IP (1)

通过 ME 调用 Hash Unit，对内部网络发送到外部网络数据包中的 inner_port、inner_IP 共 48 位进行散列计算，得到散列值 hash_value，取其 0～15 位作为 key。利用 key 可以索引 65 536 个表项，而端口也是 16 位，可以直接用 key 作为全局端口，即

global_port = key (2)

把 65 536 个表项从 NAT 基址开始按顺序存放在 SRAM 中，就可以用下面的式子把每一项的起始地址与 key 关联：

address(key)=NAT_TABLE_BASE+16*key (3)

若不同的二元组通过散列计算得到相同的 key，则产生散列冲突。利用“闭散列法”可解决冲突，即取 hash_value 的 1～16 位作为新的 key。若再次冲突，则再右移一位取值。NAT 表共占 512KB，SRAM 足以容纳。

3.2 NAT 表项的创建和删除

数据包分为两类：从内部网络发往外部网络的包称为输出流；从外部网络发往内部网络的包称为输入流。只有输出流才有可能建立新的表项，步骤如下：

(1)一个内部包被 ME 接收后，取出二元组经散列计算得到 key，到(2)。

(2)读 NAT 表中 key 关联的地址。若 flag 为 0，表示一个新的连接，需创建表项，到(4)；若 flag 为 1，则检查包的源 IP 地址和端口是否与当前表项的二元组一致，到(3)。

(3)如果一致，进行转换操作，到(5)；若不一致，则解决冲突得到新的 key，到(2)。

(4)用中断通知 XScale，把二元组和 key 放入信箱(见 3.3 节)供 XScale 使用，到(6)。ME 可以立即转换该包，到(5)。

(5)将数据包中的 inner_port、inner_ip 分别用 global_port、global_ip 取代，传递给下一处理线程，同时将刚使用过的表项 timeout 值重置为初值。对含 FIN 的 TCP 报文需用中断通知 XScale，并把 key 放入信箱，XScale 负责删除该 key 对应的表项。

(6)中断处理程序唤醒服务程序，从信箱中读出二元组和 key，由式(3)算出 key 关联地址，为新表项的起始地址，按图 4 写 SRAM，其中 flag 为 1，timeout 为初始值。

输入流的处理相对简单，只需读出 NAT 表中该包目的端口所对应的表项。若该表项有效，则用二元组覆盖数据包中的目的 IP 地址和端口，传递给下一个处理线程，同时更新该表项的 timeout 值为初值，对含 FIN 的 TCP 报文作相应处理；若表项无效，则丢弃数据包。

触发删除表项的事件有收到 FIN 置位的 TCP 报文或该表项的 timeout 值为 0。对前者，XScale 收到中断后，调用服务程序取出信箱中的 key，将 key 对应的表项 flag 置 0。timeout 初值为可配置参数 TIMEOUT_INIT，默认值 60，单位为可配置参数 TIMEOUT_UNIT，默认值 5s。具体取多长时间合适，是一个值得研究的问题。XScale 上的定时器负责每单位时间提醒一次服务程序，查看 NAT 表的每个有效表项的 timeout 值，若大于 0，则将 timeout 值减 1 后重写；若为 0，则将 flag 置 0，即删除该 NAT 表项。

消息类型	消息发起者	消息接收者
属性 1 的长度	属性 1 的值	
属性 2 的长度	属性 2 的值	
…		
属性 n 的长度	属性 n 的值	

图 4 信箱的消息格式

3.3 ME 与 XScale 之间的通信机制

ME 与 XScale 之间的通信机制是否快速有效是 NP 应用设计的关键。前面提到 ME 设置中断，把有关信息放在信箱中，而 XScale 被中断后读取信箱中的信息作下一步操作，这就是我们设计的信箱通信机制。信箱的消息格式如图 4 所示。

消息的主要内容为若干条属性值。信箱基址由可配置参数 MAILBOX_BASE 定义。消息类型 8 位，类型决定了属性有多少项以及每项的意义。如类型 1 表示请求创建表项，有 3 个属性，分别是 inner_ip、inner_port、key；类型 2 表示请求删除表项，有 1 个属性 key。属性的值为 1 字节～256 字节不等。

（下转第 116 页）