

## 一种基于索引指针的可扩展 IP 包分类算法

李金库<sup>1</sup>, 马建峰<sup>1</sup>, 张德运<sup>2</sup>

(1 西安电子科技大学 计算机学院, 陕西 西安 710071; 2 西安交通大学 电信学院, 陕西 西安 710049)

**摘 要:** 设计并实现了一种基于索引指针的可扩展 IP 包分类算法. 该算法通过分析源/目的端口号和协议类型字段在实际应用中的分布特性, 将这 3 个字段映射到一个 8 比特元组上, 压缩了分类维数; 算法依据压缩后的 8 比特元组将分类规则集划分为 256 个子集, 并为每个子集建立一个索引指针, 指向该子集的存储起始地址; 算法通过计算 IP 包中“源/目的 IP 地址联合字段”中各个比特的信息熵值, 找出最优的比特序列作为根和子节点, 为每个规则子集建立一棵 Tries 查找树, 既保证了存储空间和查找时间最小, 而且不存在回溯问题. 实验结果证明, 该算法分类效率高.

**关键词:** 包分类; 索引指针; 信息熵; Tries 查找树

**中图分类号:** TP393

**文献标识码:** A

**文章编号:** 1000-7180(2012)04-0032-04

## A Scalable IP Packet Classification Algorithm Using Indexed Pointers

LI Jin-ku<sup>1</sup>, MA Jian-feng<sup>1</sup>, ZHANG De-yun<sup>2</sup>

(1 School of Computer Science and Technology, Xidian University, Xi'an 710071, China;

2 School of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an 710049, China)

**Abstract:** In this paper, a scalable IP packet classification algorithm using indexed pointers has been proposed. According to the distribution of source port, destination port and protocol type fields in the real applications, the algorithm maps the three fields to an eight-bit value and divides the whole rule set into 256 subsets. It assigns each subset an indexed pointer that points to the starting address of its storage space. The algorithm finds the best bit sequence and uses them as root and child nodes by calculating each bit's information entropy value of the combined field of source IP address and destination IP address, then it establishes a Tries lookup tree for each rule subset. By doing so, it requires the least storage space and lookup time without retrospect. The experimental results indicate that the new algorithm is highly efficient.

**Key words:** packet classification; indexed pointer; entropy of information; Tries lookup tree

### 1 引言

Internet 技术的发展要求路由器, 尤其是边界路由器支持诸如策略路由、访问控制、VPN、网络地址转换、负载均衡、区分服务、QoS、认证计费等各种应用, 这些应用都是以数据包分类技术为基础的. 它们要求路由器能够根据包头中的某些字段将数据包划分为不同标识的流, 从而针对不同的流采取相应的操作. 数据包分类算法的好坏和效率的高低直接

影响了整个路由器的性能. 一个典型的数据包分类的例子是 IP 路由查找, 运用最长前缀匹配规则 LPM(Longest Prefix Match)查找路由表, 根据目的 IP 地址确定下一跳地址. 目前, 关于路由查找算法的研究已经逐步成熟, 而通用的多维数据包分类算法由于其复杂性和难度较高, 一直是近年来的研究热点<sup>[1-6]</sup>. 在现有算法中, AQT 算法<sup>[1]</sup>、Grid of Tries 算法<sup>[2]</sup>、FIS-tree 算法<sup>[3]</sup>都属于二维算法, 很难扩展到多维情形; 交叉乘积算法<sup>[2]</sup>、RFC 算法<sup>[4]</sup>、

HiCuts算法<sup>[5]</sup>对空间要求大,扩展性较差;位向量(BV)算法和聚合位向量(ABV)算法<sup>[6]</sup>的硬件实现速度快,但软件实现速度太慢;因此本文设计和实现了一种新的基于索引指针的可扩展 IP 包分类算法<sup>[7]</sup>。

## 2 算法设计

可供分类的字段分布在从数据链路层到应用层的各个层次中,但目前在实际应用中主要有两种分类组合,一种是源/目的 IP 地址的 2 维分类,另一种是源/目的 IP 地址、源/目的端口号、协议类型的 5 维分类。本文算法属于后者,又称为 5 元组分类。

### 2.1 主要设计思路

(1) 通过分析源/目的端口号和协议类型字段在实际应用中的分布特性,将这 3 个字段共 40 比特映射到一个 8 比特的元组上,从而将原 5 维分类压缩为 3 维分类;

(2) 依据(1)得到的 8 比特元组将原分类规则集划分为 256 个子集,并为每个子集建立一个索引指针,指向该规则子集的存贮起始地址;

(3) 将规则集存贮在一段系统专门预留的 SRAM 空间上,同时通过预处理,计算出规则子集内所有规则“源/目的 IP 地址联合字段”各个比特的信息熵值,从中找出最优的比特序列,并以该最优比特序列作为根和子节点,为每个规则子集建立一棵 Tries 查找树。该查找树的叶子节点为一个数量不超过 cluster\_size 的规则 cluster,通过查找 Tries 树定位到相应的叶子节点后,数据包只需要跟不超过 cluster\_size 个规则依次进行比较,就可以确定数据包的类别。该 Tries 查找树结构既保证了存贮空间和查找时间最小,而且不存在回溯问题。

### 2.2 源/目的端口号及协议字段到 8 比特元组映射

通过对 Cisco、Juniper、华为等知名厂商路由器配置的研究和对西北高校(Cernet)网络设备的配置情况进行考察发现,协议字段虽然共有 8 位,但在实际配置中通常只会用到 TCP、UDP、ICMP 协议,其他比如 GRE、IGMP、IPNIP 等协议用到的非常少,所以可以将协议字段压缩为两个比特表示,对照关系如表 1 所示。

表 1 压缩协议字段对照表

| 协议类型     | TCP | UDP | ICMP | OTHER |
|----------|-----|-----|------|-------|
| 压缩后二进制编码 | 00  | 01  | 10   | 11    |

在客户端/服务器模式下,端口号被分为两种,一种是保留端口,范围为 1~1023,另一种为临时端口,范围为 1024~65535。临时端口通常由客户端临时动态分配,大多数由内核指定,除了标识一个连接的客户端点外,并没有其他意义。规则集中经常将所有临时端口作为一大类来处理,比较常见的表示方式为“port≥1024”。而在保留端口中,规则一般集中分布在 21 和 20 (FTP)、80 (HTTP)、23 (TELNET)、25 (SMTP)、110 (POP)、53 (DNS) 等几种情形,所以可以将 16 位的端口号字段压缩为 3 个比特表示,对照关系如表 2 所示。

表 2 压缩端口号字段对照表

| 端口号        | 对应协议   | 压缩后二进制编码 |
|------------|--------|----------|
| 21/20      | FTP    | 000      |
| 80         | HTTP   | 001      |
| 23         | TELNET | 010      |
| 25         | SMTP   | 011      |
| 110        | POP    | 100      |
| 53         | DNS    | 101      |
| 1024~65535 | 临时连接   | 110      |
| *          | OTHER  | 111      |

综合考虑源/目的端口号及协议类型字段,参照表 1 和表 2,可以将这三个字段映射到一个 8 比特的元组上,从而将原 5 维分类压缩为 3 维。映射关系如图 1 所示。

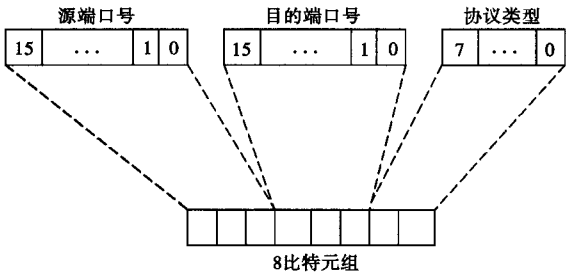


图 1 源/目的端口号协议字段到 8 比特元组映射

### 2.3 索引指针

传统的基于 CPU 缓存的数据包分类方法在实际应用环境中因为命中率较低,并且流的持续时间较短,工作效果并不是很好。本文算法采用了一种称为“索引指针”的机制。它为每个规则子集设立一个索引指针,该指针指向在 SRAM 空间上为规则子集建立的 Tries 查找树。索引指针在系统初始化时建立,系统运行过程中始终有效。系统进行数据包分类

时,首先根据图 1 将包头中源/目的端口号和协议类型字段映射到一个 8 比特元组,根据该元组的值取得对应的索引指针;然后转向该指针指向的 SRAM 空间,查找相应的 Tries 查找树,得到该数据包对应的规则。

#### 2.4 基于最优比特序列的 Tries 查找树

本文算法在系统预留的 SRAM 空间上为 256 个规则子集各建立一个“基于源/目的 IP 地址联合字段最优比特序列的 Tries 查找树”,建树的原理和过程如下。

源/目的 IP 地址是由多个比特组成的一个序列,每个比特可取值的范围为 $\{0,1,*\}$ ,其中 $*$ 代表任意值。可以考虑从中找出存贮空间和查找时间最小,也就是最优的比特序列将规则子集划分为多个 cluster,每个 cluster 包含的规则数目不超过某个可以配置的值,比如 2 或 4,记为 cluster\_size。通过比较数据包源/目的 IP 地址中与所选出的最优比特序列相对应的位,就可以迅速定位到相应的 cluster,然后再将数据包与 cluster 中的规则至多进行 cluster\_size 次比较,就可以确定数据包所对应的规则,完成数据包分类。那么,如何确定最优的比特序列呢?

本文算法采用一种基于统计度量的信息熵(或香农熵)函数<sup>[8]</sup>来解决这个问题。假设选择源/目的 IP 地址联合字段的第  $j$  位比特作为依据划分规则子集,则整个规则子集被划分为左右两个 cluster,左 cluster 包含规则子集中所有第  $j$  位为 0 或 \* 的规则,右 cluster 包含规则子集中所有第  $j$  位为 1 或 \* 的规则。设  $N_j$  代表选择第  $j$  位比特时规则子集内规则总数, $N_0$  代表第  $j$  位为 0 的规则数目, $N_1$  代表第  $j$  位为 1 的规则数目, $N_*$  代表第  $j$  位为 \* 的规则数目,则

$$P(0) = (N_0 + N_*) / N_j \quad (1)$$

$$P(1) = (N_1 + N_*) / N_j \quad (2)$$

$P(0)$ 和 $P(1)$ 分别代表规则子集中第  $j$  位比特

为 0 和 1 的统计概率,且有  $\sum_{i=0}^1 P(i) = 1$ 。

假设  $S_j$  代表第  $j$  位比特的信息熵,则

$$S_j = - \sum_{i=0}^1 (P(i) * \log P(i)) \quad (3)$$

建立 Tries 查找树的过程为:首先判断规则子集的数目,如果大于 cluster\_size,则依次计算子集内规则“源/目的 IP 地址联合字段”所有比特位的信息熵,以熵值最大的位作为根,将规则子集划分为两

个 cluster;然后以 cluster 作为新的规则子集,重复上面的过程,直到每个 cluster 内规则的数目都不超过 cluster\_size 为止。图 2 中显示了一个划分后的 cluster\_size 为 2 的 3 层 Tries 查找树。

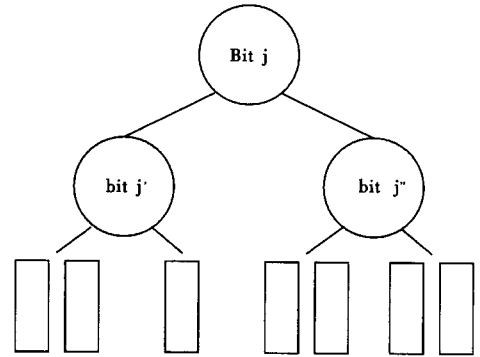


图 2 一个 clustersize 为 2 的 3 层 Tries 查找树

### 3 算法实现

本文算法包括预处理和包分类两部分。预处理部分在系统初始化时进行,它的处理过程如下:

(1) 初始化索引指针数组 indexed\_pointer [256] 为 0;

(2) 设定 Tries 查找树 cluster 大小 cluster\_size;

(3) 遍历规则集  $R$ ,取每个规则的源/目的端口号和协议字段,依据表 1、表 2 和图 1 将整个分类规则集划分为 256 个子集;

(4) 取其中一个子集,设它的子集号为  $n$  ( $n$  为它所对应的 8 位元组的值),如果非空,则在 SRAM 上为它分配一个 Tries 查找树初始地址  $P$ ,并将  $P$  赋给 indexed\_pointer [ $n$ ];

(5) 判断子集  $N$  内规则数目,如果大于 cluster\_size,依次取子集内规则的源/目的 IP 地址字段,根据式(1)~式(3)计算每一位的信息熵值;

(6) 以信息熵值最大的那一位比特为根,将规则子集划分为左右两个 cluster;

(7) 将左右两个 cluster 看作两个规则子集,重复(5)和(6),直到每个 cluster 内规则数目均不超过 cluster\_size;

(8) 重复(4)~(7),直到将所有的规则子集处理完成;

(9) 预处理结束。

数据包分类部分在系统运行过程中进行,处理过程如下:

(a) 分类器  $C$  收到一个数据包;

(b) 取数据包头中源/目的端口号和协议类型

字段,依据表1、表2和图1映射到一个8位元组 $m$ ;

(c) 判断  $\text{indexed\_pointer}[m]$  的值,如果为0,转(f);

(d) 查找  $\text{indexed\_pointer}[m]$  指向的 Tries 查找树,依次判定包头内源/目的IP地址字段与该 Tries 查找树最优比特序列相对应位的值,定位到对应的 cluster;

(e) 将数据包与该 cluster 内规则依次进行比较,找出优先权最大的规则;

(f) 查找结束,返回(a).

#### 4 实验结果及分析

本文采用美国斯坦福大学的包分类算法测试仿真工具 PALAC(Packet Lookup And Classification simulator)<sup>[9]</sup>来测试本算法的性能.我们用 C++ 语言分别实现了本文算法和广泛使用的 HiCuts 算法<sup>[5]</sup>,测试和比较了两种算法的平均查询时间.

本文选用由美国华盛顿大学开发的 ClassBench<sup>[10]</sup>平台来生成实验规则集和相应的数据包集. ClassBench 以真实的规则库作种子,通过调控参数的取值来生成符合实际特征的规则库,从而能够更准确、真实的模拟数据包分类的运行环境.我们选用了最常用的规则类型 Access Control List (ACL),测试了包含 5 种不同规模规则集:2 000 条,4 000 条,6 000 条,8 000 条和 10 000 条情形下两种算法的平均查询时间,实验结果如图 3 所示.

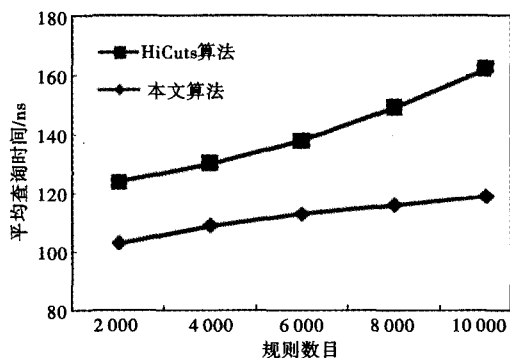


图3 本文算法和 HiCuts 算法平均查询时间

从图3可以看出,本文算法平均查询时间明显优于 HiCuts 算法(大约小 20~40 ns 左右).同时,从图中变化趋势分析,本文算法总体趋势较平缓,而 HiCuts 算法随着规则数目的增多平均查询时间增速加大.这与 HiCuts 算法在大规模集时扩展性较差有直接关系,而本文算法则不存在这个问题.

#### 5 结束语

本文设计并实现了一种基于索引指针的 IP 包分类算法.该算法通过分析源/目的端口号和协议类型字段在实际应用中的分布特性,将这 3 个元组共 40 比特映射到一个 8 比特的元组上,从而压缩了分类维数;算法依据压缩后的 8 比特元组将分类规则集划分为 256 个子集,并为每个子集建立一个索引指针,指向该子集的存储起始地址;算法通过计算 IP 包中“源/目的 IP 地址联合字段”各个比特的信息熵值,找出最优的比特序列作为根和子节点,为每个规则子集建立一棵 Tries 查找树;查找树的叶子节点为一个数量不超过 cluster\_size 的 cluster,通过查找 Tries 树定位到相应的叶子节点后,数据包只需跟不超过 cluster\_size 个规则进行比较,就可以确定数据包的类别;该 Tries 查找树结构保证了算法的存储空间和查找时间最小,而且不存在回溯问题.实验结果证明,该算法分类效率高.

#### 参考文献:

- [1] Buddhikot M, Suri S, Waldvogel M. Space decomposition techniques for fast layer4 switching[C]// Proceedings of Conference on Protocols for High Speed Networks. Salem, Massachusetts, USA, 1999: 25-41.
- [2] Srinivasan V, Varghese G, Suri S, et al. Fast and scalable layer four switching[C]// Proceedings of the ACM SIGCOMM'98. New York: ACM Press, 1998: 191-202.
- [3] Feldman A, Muthukrishnan S. Tradeoffs for packet classification[C]// Proceedings of the INFOCOM 2000 Conference on Computer Communications. Israel: IEEE, 2000: 1193-1202.
- [4] Gupta P, McKeown N. Packet classification on multiple fields[C]// Proceedings of the ACM SIGCOMM'99. New York: ACM Press, 1999: 147-160.
- [5] Gupta P, McKeown N. Packet classification using hierarchical intelligent cuttings[J]. IEEE Micro, 2000, 20(1): 34-41.
- [6] Baboescu F, Varghese G. Scalable packet classification[J]. IEEE/ACM Trans on Networking, 2005, 13(1): 2-14.
- [7] 柳斌,李之棠,涂浩.基于半监督学习的应用流分类方法[J].微电子学与计算机,2010,27(8):24-26.
- [8] 陈运,周亮,陈新.信息论与编码[M].北京:电子工业出版社,2002.

(下转第40页)

## 5 结束语

针对浮点 FFT 处理器实现中资源开销较大的问题,提出基于蝶形共享策略的  $R2/2^2$  SDF 流水结构,在保证高速流水处理的前提下,提高浮点计算单元的利用率并降低面积开销。

通过比较分析发现基 4 算法是实现浮点 FFT 流水结构的较好算法,在复数乘法次数和复数乘法器数目方面优于使用较多的基 8 算法。在使用单口存储器构建 FIFO 读写时序的前提下,提出一种蝶形共享的可配置流水结构,不但有效提高了蝶算单元的使用效率,而且减少了面积开销。

本文提出的方法有效解决了浮点设计中资源开销瓶颈,可以应用于其他高精度可配置的 FFT/IFFT 处理器设计中。

## 参考文献:

- [1] Yuan chen, Lin Yuwei, Lee Chenyi. A block scaling FFT-IFFT processor FOR WiMax applications[C]//IEEE Asian Solid-State Circuits Conference. Hangzhou: IEEE, 2006: 203-206.
- [2] 云霄. 可配置 FFT/IFFT 处理器的设计及其 FPGA 构造[D]. 西安: 西安电子科技大学, 2009.
- [3] LogiCORE IP fast fourier transform v8.0[EB/OL]. (2010-09-21). [2011-06-20]. [http://china.xilinx.com/support/document-ation/ip\\_documentation/ds808\\_xfft.pdf](http://china.xilinx.com/support/document-ation/ip_documentation/ds808_xfft.pdf).
- [4] 吴斌, 姜鑫, 周玉梅. 基于 802.11a 的 FFT/IFFT 处理器设计[J]. 微电子学与计算机, 2011, 28(4): 61-64.
- [5] Xiaojin Li, Zongsheng Lai, Jianmin Cui. A low power and small area FFT processor for OFDM demodulator[J]. IEEE Transactions on Consumer Electronics, 2007, 53(2): 274-277.
- [6] 何星, 张铁军, 侯朝焕. 流水线结构 FFT/IFFT 处理器的设计与实现[J]. 微电子学与计算机, 2007, 24(4): 141-147.
- [7] 姜鑫, 唐广, 吴斌. 可配置高精度 FFT/IFFT 的设计[J]. 微电子学与计算机, 2010, 27(7): 44-48.
- [8] Hao Xiao, An Pan, Yun Chen, et al. Low-cost reconfigurable VLSI architecture for fast fourier Transform[J]. IEEE Transactions on Consumer Electronics, 2008, 54(4): 1617-1622.
- [9] Yang Yaoxian, Li Jinfu, Liu Hsiangning. Design of cost-efficient memory-based FFT processors using single-port memories[C]//IEEE International Conference on SOC. Hsinchu: IEEE, 2007: 29-32.
- [10] Ambarish Mukund Sule. Design of pipeline fast fourier transform processors using 3 dimensional integrated circuit technology[D]. Raleigh: North Carolina State University, 2007.

## 作者简介:

桑红石 女, (1970—), 副教授. 研究方向为图像处理和集成电路设计等。

高伟 男, (1987—), 硕士研究生. 研究方向为数字信号处理的 VLSI 设计。

(上接第 35 页)

- [9] Gupta P, Balkman J. PALAC: PAccket lookup and classification simulator[EB/OL]. [2011-06-10]. [http://klamath.stanford.edu/tools/PALAC/SRC/DOCUMENT/palac\\_manual5.pdf](http://klamath.stanford.edu/tools/PALAC/SRC/DOCUMENT/palac_manual5.pdf).
- [10] Taylor D, Turner J. ClassBench: A packet Classification Benchmark[J]. IEEE/ACM Trans on Networking, 2007, 15(3): 499-511.

## 作者简介:

李金库 男, (1976—), 博士, 副教授. 研究方向为网络体系架构\计算机系统及网络安全。

马建峰 男, (1963—), 博士, 教授, 博士生导师. 研究方向为无线/移动网络体系及安全。

张德运 男, (1941—), 教授, 博士生导师. 研究方向为网络系统结构、网络安全。