Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 29 (2006) 3037–3050
computer communications
www.elsevier.com/locate/comcom

A scalable IPv6 route lookup scheme via dynamic variable-stride bitmap compression and path compression ☆

Kai Zheng *, Zhen Liu, Bin Liu

Department of Computer Science, Tsinghua University, Beijing 100084, PR China

Available online 10 January 2006

Abstract

The significantly increased address length of IPv6 (128-bit) provides an endless pool of address space. However, it also poses a great challenge on wire-speed route lookup for high-end routing devices, because of the explosive growth of both lookup latency and storage requirement. As a result, even today's most efficient IPv4 route lookup schemes can hardly be competent for IPv6. In this paper, we develop a novel IPv6 lookup scheme based on a thorough study of the distributions of real-world route prefixes and associative RFC documents. The proposed scheme combines the bitmap compression with path compression, and employs a variable-stride mechanism to maximize the compress ratio and minimize average memory reference. A possible implementation using mixed CAM devices is also suggested to further reduce the memory consumption and lookup steps.

The experimental results show that for an IPv6 route table containing over 130K prefixes, our scheme can perform 22 million lookups per second even in the worst case with only 440 Kbytes SRAM and no more than 3 Kbytes TCAM. This means that it can support 10 Gbps wire-speed forwarding for back-to-back 40-byte packets using on-chip memories or caches. What's more, incremental updates and high scalability is also achieved.

Keywords: CAM; Compression; IPv6; Route lookup

1. Introduction

Internet Protocol Version 6 (IPv6) is one of the key supporting technologies of the *Next Generation Network* (NGN). The most distinctive feature of IPv6 is its 128-bit address, which is significantly increased from IPv4 and provides an extremely large pool of address space for the Internet. However, this innovation also poses a great challenge on the data path functions of IP packet forwarding, such as IP address lookup. This pressure is caused mainly by the following two factors: (1) the performances of most existing IP lookup algorithms, including lookup throughput and storage requirement, are sensitive to the key length and will decrease dramatically when migrated to IPv6. (2) The rapid advances in fiber-optic technology push the link speed of backbone from 2.5 to 10 Gbps or even 40 Gbps, making wire-speed forwarding much more difficult to guarantee. Therefore, an effective *Longest Prefix Matching* (LPM) algorithm for 128-bit IP address is very essential for routers/switches deployed at future networks.

Currently, packet forwarding based on CIDR IPv4 address lookup is well understood with both trie-based algorithms and TCAM-based mechanisms in the literature. trie-based algorithms are usually said to be time consuming, and commonly a lot of memory accesses are needed for a single address lookup. Moreover, most of them [1–4] can hardly be scaled to support IPv6, because their lookup time grows linearly with the search key (Destination IP Address) length. Some algorithms have constant search

☆ This work was supported by NSFC (Nos. 60173009 and 60373007), China 863 High-tech Plan (Nos. 2002AA103011-1 and 2003AA115110), China/Ireland Science and Technology Collaboration Research Fund (CI-2003-02), and the Specialized Research Fund for the Doctoral Program of Higher Education of China (No. 20040003048).
* Corresponding author. Tel.: +8610 6277 3441; fax: +8610 6277 3616.
E-mail addresses: Zk01@mails.tsinghua.edu.cn (K. Zheng), Liuzhen02@mails.tsinghua.edu.cn (Z. Liu), Liub@tsinghua.edu.cn (B. Liu).

0140-3664/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2005.11.003

time for IPv4, such as DIR-21-3-8 [5]. However, their high storage requirements only allow them to be implemented with mass-but-slow SDRAMs. What's more, the exponentially increased memory requirements with the search key length make them not suitable for IPv6 implementation.

The Lulea [6] algorithm, for the first time, develops the concept of bitmap compression to improve storage efficiency, achieving both small memory requirement and almost constant lookup time. Hence much faster SRAM chips can be employed to replace SDRAMs. However, its specially designed memory organization is un-scalable for IPv6 or large route tables. Furthermore, it is known to be very hard to update since it needs leaf-pushing. The Eatherton's Tree Bitmap algorithm [7] improves upon Lulea by creating a data structure (with two kinds of bitmaps, Internal Bitmap and External Bitmap) which does not require leaf-pushing and therefore supports fast incremental updates. An implementation of the Tree Bitmap algorithm, referred to as Fast IP Lookup (FIPL) [8], shows that a storage requirement of about 6.3 bytes per prefix and performance of over one million lookups per FIPL engine can be achieved. However, FIPL uses a fixed lookup stride of four at each level, hence for each IP address lookup it may require more than $32/4 = 8$ memory accesses in the case of IPv4, and $128/4 = 32$[1] in the case of IPv6, which make it also infeasible for IPv6 migration.

Ternary Content Addressable Memory (TCAM) is a fully associative memory that allows a "don't care" state to be stored in each memory cell in addition to 0s and 1s. Since the contents of a TCAM can be searched in parallel and the first matching result can be returned within only a single memory access, TCAM-based scheme is very promising in terms of building a high-speed LMP lookup engine [9,10]. Moreover, TCAM-based tables are typically much easier to manage and update than trie-based ones. However, on the other hand, the high cost to density ratio and high power consumption of TCAM prevent it from being widely adopted in building route lookup engines. This situation will get even worse when migrated to IPv6, since the storage requirement for TCAM-based tables also grows linearly with the address length.

A good IP (v6) address lookup scheme should have the following features. (1) High enough lookup throughput to support high-speed interface even in the worst case (i.e., minimum-sized packets coming in back-to-back); (2) small memory requirement, making it practical to be implemented with small but fast memory chips or on-chip caches; (3) scalability with the key length, maintaining the lookup time, and memory requirement at a feasible low level when migrated to IPv6; (4) Low update cost, to cope with the instability of the BGP protocol.

In this paper, we develop a novel lookup scheme which can achieve all the features mentioned above. The main contributions of the paper are threefold: first, we introduce a series of trie-associated concepts and terms, presenting a new and explicit view of the IP lookup problem and the related techniques/methods. Second, by analyzing real-world IPv4 route tables and several IPv6 "initial allocation" route tables, we not only provide a number of informative observations on distributions of route prefixes and several useful heuristics for lookup scheme designing, but also build an IPv6 prefix generator according to these observations and the recommendations from associative RFC and RIPE documents. This is very essential for current IPv6 lookup scheme performance evaluations, since the available "future-like" IPv6 route tables are not sufficient. Finally, based on those observations, we develop a novel *Dynamic Variable-Stride Bitmap Compressing lookup scheme with Path Compression* (DVSBC-PC) and a further optimized scheme with mixed *CAM device* (DVSBC-PC-CAM).

The rest of the paper is organized as follows. Section 2 presents several trie-associated concepts and terms. In Section 3, we list a number of informative observations on distributions of route prefixes, present several useful heuristics for lookup scheme designing, and develop an IPv6 prefix generator for performance evaluation. Section 4 describes the dynamic variable-stride bitmap compressing IP lookup scheme with path compression (DVSBC-PC) and a further optimized scheme with mixed CAM mechanism (DVSBC-PC-CAM). Section 5 presents the experimental results and performance evaluation of the proposed scheme, as well as the comparison with other schemes. Finally, a conclusion is drawn in Section 6.

2. Definitions and terms for IP address lookup and the trie system

In our opinion, the prefix trie system is more than a tool to introduce trie-based algorithms. In addition to an important data structure in both route table organization and route lookup implementation, it is also very helpful in acquiring a better understanding of the LPM problem. In this section, for the sake of providing intuitions for IP address lookup scheme design and presenting an explicit view of the trie system, we introduce/develop a set of definitions and terms, which will be used throughout the paper.

2.1. Definition: IP address

The bits in a specific IPv6 address are ordered as shown in Fig. 1, where the 1st bit (MSB) lies in the leftmost position and the 128th bit (LSB) lies in the rightmost position. And we let IP[$i - j$] denotes the set of the ith to jth bits of an IP address.

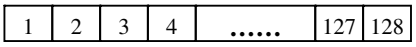

Fig. 1. IP address format.

[1] Actually, as reported by the authors, three additional memory accesses are needed.