International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Analysis and Data Retrieval by Filtering Packets in High Speed Routers

Indumathi[1], K. Kumar[2]

[1]MCA (Final Year) Vel Tech, Veltech Technical University

[2]Assistant Professor, Velltech Technical University

Abstract: ***In this paper, we are going to decompose the operation of multimatch packet classification from the complicated multidimensional search to several single-dimensional searches, and present an asynchronous pipeline architecture based on a signature tree structure to combine the intermediate results returned from single-dimensional searches. By spreading edges of the signature tree across multiple hash tables at different stages, the pipeline can achieve a high throughput via the interstate parallel access to hash tables. Two edge-grouping algorithms are designed to evenly divide the edges associated with each stage into multiple work-conserving hash tables. The proposed pipeline architecture outperforms Hyper Cuts and B2PC schemes in classification speed by at least one order of magnitude, while having a similar storage requirement. Particularly, with different types of classifiers of 4K rules, the proposed pipeline architecture is able to achieve a throughput between 26.8 and 93.1 GB/s using perfect hash tables. Multiple string match is an important problem in many application areas of computer for instance there is an increasing demand for fast analysis and data retrieval although there are various kinds of comparison tools that provide aligning and approximate matching most of them are based on exact matching in order to speed up the process. Multiple string match is an important problem in many application areas of computer for instance there is an increasing demand for fast analysis and data retrieval although there are various kinds of comparison tools that provide aligning and approximate matching most of them are based on exact matching in order to speed up the process. Another important usage of multiple string matching algorithms appears in NIDS [network intrusion detection systems]. Snort is a light weight open source NIDS which can filter packets based on predefined rules. Another important usage of multiple string matching algorithms appears in NIDS [network intrusion detection systems]. Snort is a light weight open source NIDS which can filter packets based on predefined rules.***

Keywords: Multimatch packet, High-speed routers, Data retrieval, NIDS, TCOM

1. Introduction

As the Internet are growing to rapidly, packet classification has become a major bottleneck of high-speed Routers. Most traditional network applications require packet Classification to return the best or highest-priority matched rule[1]. However, with the emergence of new network applications like Network Intrusion Detection System (NIDS), packet-level accounting, and load balancing, packet classification is required to report all matched rules, not only the best matched rule. Packet classification with this capability is called multimatch packet classification to distinguish it from the conventional best-match packet classification[1][2].

Many schemes have been proposed in literature aiming at optimizing the performance of packet classification in terms of classification speed and storage cost. However, most of them focus on only the best-match packet classification. Although some of them could also be used for multimatch packet classification, they suffer from either a huge memory requirement or steep performance degradation under certain types of classifiers. Ternary content addressable memory is well known for its parallel search capability and constant processing speed, and it is widely used in IP route lookup and best-match packet classification. Due to the limitation of its native circuit structure, Ternary content addressable memory(TCAM)can only return the first matching entry, and therefore cannot be directly used in multimatch packet classification. To enable the multimatch packet classification on Ternary content addressable memory, some research works published recently propose to add redundant intersection rules in TCAM. However, the introduction of redundant intersection rules further increases the already high implementation cost of the Ternary content addressable memory system. The objective of this paper is to design a high-throughput and memory-efficient multimatch packet classification scheme without using TCAMs[1][3]. Given the fact that a single-dimensional search is much simpler and has already been well studied, we decompose the complex multimatch packet classification into two steps. In the first step, single-dimensional searches are performed in parallel to return matched fields in each dimension.

In this paper are summarized as follows.

1) We model the multimatch packet classification as a concatenated multistring matching problem[2], which can be solved by traversing a signature tree structure.
2) We propose an asynchronous pipeline architecture to accelerate the traversal of the signature tree. By distributing
3) Edges of the signature tree into hash tables at different stages, the proposed pipeline can achieve a very high throughput.

We propose two edge-grouping algorithms to partition the hash table at each stage of the pipeline into multiple work conserving hash tables, so that the intrastate parallelism can be exploited. By taking advantage of the properties of the signature tree, the proposed edge-grouping algorithms perform well in solving the location problem, overhead minimization problem, and balancing problem involved in the process of hash table partition.

4) We propose a hybrid perfect hash table construction scheme, which can build perfect hash tables for each stage

Licensed Under Creative Commons Attribution CC BY

of the pipeline structure, leading to an improved performance in both classification speed and storage complexity.

2. Motivation

Due to the high power consumption of TCAM, some schemes are proposed to reduce it using one of the following two ideas: 1) reducing the TCAM entries required to represent a classifier by using range encoding or logic optimization or 2) selectively activating part of the TCAM blocks when performing a classification. Although these schemes reduce the power consumption of TCAM, they cannot be directly applied to multimatch packet classification. To enable the multimatch packet classification on TCAM, proposes several schemes that allow TCAM to return all matched entries by searching the TCAM multiple times after adding a discriminator field in TCAM. Consequently, the power consumption and processing time increase linearly when the number of entries matching a packet increases.

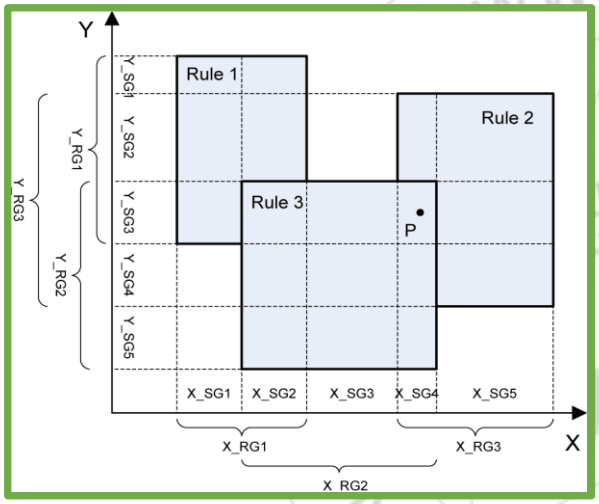

Figure 1: Segment encoding versus range encoding

In this paper, we focus on the two-stage schemes, in which the multidimensional search of packet classification is first decomposed into several single-dimensional searches, and then the intermediate results of single-dimensional searches are combined to get the final matched rule. To facilitate the combination operation, each field of rules in the two-stage schemes is usually encoded as either a range ID or several segment IDs. Consider the classifier shown in Fig. 1, which has three two-dimensional rules, each represented by a rectangle. Ranges are defined as the
Projections of the rectangles along a certain axis.

3. Problem Definition

A classifier is a set of rules, sorted in descending order of priorities. The priorities of rules are usually defined by their rule IDs, where a smaller rule ID means a higher priority. Each rule includes fields, each of which represents a range of a certain dimension. From a geometric point of view, each rule represents a hyper-rectangle in the -dimensional space. Since each packet header corresponds to a point in the –dimensional space, the problem of conventional best-match packet classification is equivalent to finding the highest-priority hyper-rectangle enclosing point ,while the problem multimatch packet classification is equivalent to finding all hyper-rectangles enclosing point . In order to perform the multimatch packet classification efficiently, given a classifier, we convert it to an encoded counterpart by assigning each distinct range a unique ID on each dimension. Given the classifier in Table I, its encoded counterpart is shown in Table II, in which is the ID of the unique range appearing on the dimension of the classifier.

In order to perform the multimatch packet classification efficiently, given a classifier, we convert it to an encoded counterpart by assigning each distinct range a unique ID on each dimension.

The main challenge of the concatenated multistring matching problem is to examine a large number of concatenated strings at an extremely high speed to meet the requirement of high-speed routers[6].

4. Literature Survey

If we can build an efficient filter for a single rule, does it mean we can build an equally efficient packet filtering system for multiple rules.

An ideal packet filtering system should have the following properties:

- All the matching rules can be identified in a single scan. It means that a packet can be processed without repeated tests on the same fields.
- The time required for rule matching is insensitive to the number of rules.

Basically, there are two approaches in building a packet filer for multiple rules. One is to simply put together all the filters for individual rules, then test these filters one after another. Clearly, this is not a good choice. For example, a packet will be tested against each rule separately, while every elementary comparison common to the multiple rules will be checked more than once. Like the expression "*p.e_type*", it is going to be compared with the same value by all the rules for the IP protocol in the system. More over, the number of matching rules largely affects the performance of a filtering system. In other words, a filtering system working fine for five rules may become unacceptably slow for one hundred rules[7].

Another approach in building a packet filtering system is through a DFA (Deterministic Finite Automaton) like automaton, which can rapidly select the matching-patterns in a single scan of input [8]. A typical scenario in fulfilling this approach is to preprocess all the patterns into a DFA-like automaton, then scan the packet fields in a left to right manner. In the paper written by R. C. Sekar, R. Ramesh, and I.V. Ramakrishnan [13], a new concept named "Adaptive Pattern Matching" is proposed. The basic idea is to adapt the traversal order to suit the input patterns. Simply put, instead of browsing the information from the input one by one, we can improve the system performance by skipping over those fields that are irrelevant for matching any pattern. A detailed discussion of this algorithm can be found in [13].

Licensed Under Creative Commons Attribution CC BY

The adaptive pattern matching is a good fit for the packet filtering system. First, in a network packet, most of the critical information is stored in the various protocol headers, like IP header, TCP header or HTTP header, etc. Within a protocol header, we may only care about a small part of fields, e.g. source address in the IP header, SYN flag in the TCP header. Therefore, many fields in the protocol headers and almost the entire data portion of a packet are always useless for the pattern matching purpose, because most known intrusion patterns can be discovered through checking partial number of fields in a packet. So, by skipping most irrelevant data and examining only a limited number of fields of a packet, we can gain significant performance improvement over traditional packet filtering approach.

In the context of packet filtering for network intrusion detection, an intrusion pattern is described as a part of a matching rule in our ASL system. A direct observation is more than one rule can be matched simultaneously. For instance, a rule "*p.s_addr==xx.yy.zz.ww*" can be matched at the same time another rule "*(p.s_addr==xx.yy.zz.ww) && (p.protocol==IP_ICMP)*" is matched. This is the difference of our filter model from that of BPF or any other packet filter. In those filters, only one value is returned by the filter function to indicate whether the packet should be captured or discarded. By contrast, our filter needs to report all the rules matched by a packet.

5. Algorithm for Automaton Construction

Procedure Build (v) {

1. v is a node in automaton /* extra information are attached to each node: p is the offset to be inspected, m is the set of already matched rules and c is the set of candidate rules */
2. if (v.c is empty)
3. stop /* if no candidate rule, terminate the procedure */
4. v.p = select(v.c) /* select the next offset to inspect */
5. create all the possible branches of node v /* each branch has a edge to it from v, with corresponding value */
6. for each rule r in v.c
7. if r has test relevant to v.p
8. if test for equality
9. add r into matched rule set if r can be matched after this test, otherwise add r into candidate rule set of the branch with corresponding value
10. if test for inequality
11. add r into matched rule set if r can be matched after this test, otherwise add r into candidate rule set of all branches except the branch with corresponding value
12. else /* all the test in r are irrelevant to v.p */
13. add r into candidate rule set of all branches
14. for each branch v'
15. Build(v') /* recursively call Build for v' */

}

Architecture Diagram

SYSTEM ARCHITECTURE:

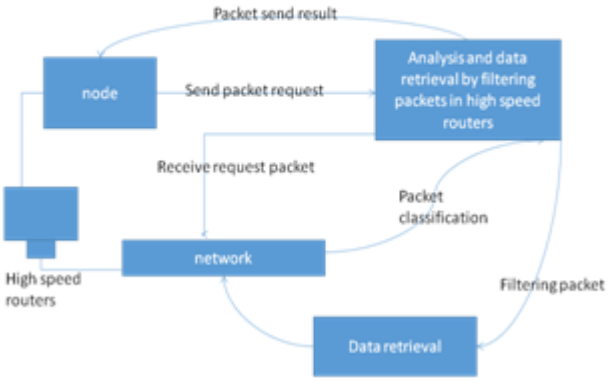

Figure 2: System architecture

The number of times that the high speed routers must be accessed to classify a packet is a good indicator for the processing speed of the signature tree-based packet classification. In the following sections, we divide the hash table into multiple partitions to exploit high speed routers access and thus improve the performance. Here, we introduce two properties about the universal character set and the signature tree, which will be used later.

Property1: *Characters in each universal character set can be encoded as any bit strings as long as there are no two characters with the same encoding.*

Property 2: *Nodes on the signature tree can be given any IDs, as long as there are no two nodes with the same IDs at the same level.*

6. Justifications of Results

In the proposed pipeline, when an AFIFO becomes full, the backpressure would prevent the upstream PM from processing new active node IDs; therefore, the size of an AFIFO might affect the throughput of the pipeline to a certain extent[9]. Fig. 3 shows the relationship between AFIFO size and the average number of time-slots needed for exporting one classification result when a conventional hash scheme is used. Curves in the figure show that the throughput of the pipeline is not sensitive to AFIFO size. When AFIFO size is larger than 8, the pipeline can achieve stable throughputs regardless of the classifier types or the value of. Further increasing AFIFO size cannot lead to significant throughput improvement. Therefore, in the remainder of our simulations, the sizes of AFIFOs are all set to eight entries.

Licensed Under Creative Commons Attribution CC BY

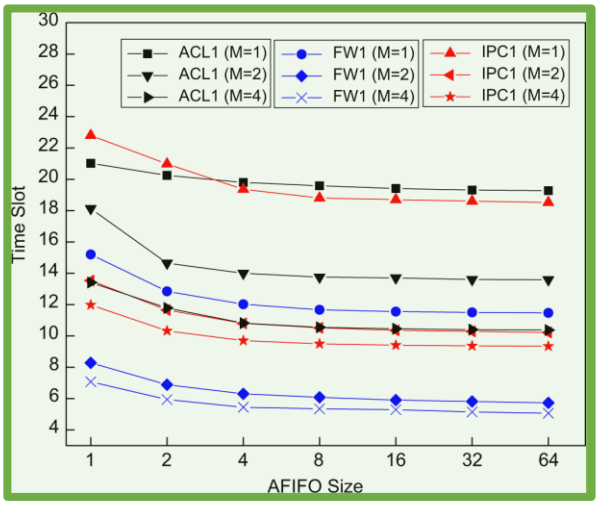

Figure 3: Time-slots for generating one result versus AFIFO size, when conventional hash scheme is used.

The proposed pipeline architecture has very strong robustness. It significantly outperforms Hyper Cuts and B2PC schemes for all tested classifiers[11]. Although part of the performance improvement is gained from the parallelism of the pipeline (in fact, the B2PC scheme also employs many parallel bloom filters to accelerate its classification speed), the use of parallelism does not increase the overall storage cost thanks to the high partition efficiency provided by the NCB_EG scheme. For ACL2, FW1, and IPC1, the Hyper Cuts scheme requires more than 200 time-slots on average to perform each packet classification.

7. Related Works

Many schemes have been proposed in literature to address the best-match packet classification problem, such as Tries-based schemes decision tree-based schemes TCAM-based schemes and two-stage schemes.

Due to the high power consumption of TCAM, some schemes are proposed to reduce it using one of the following two ideas: 1) reducing the TCAM entries required to represent a classifier by using range encoding or logic optimization or 2) selectively activating part of the TCAM blocks when performing a classification. Although these schemes reduce the power consumption of TCAM, they cannot be directly applied to multimatch packet classification. To enable the multimatch packet classification on TCAM, [1] proposes several schemes that allow TCAM to return all matched entries by searching the TCAM multiple times after adding a discriminator field in TCAM. Consequently, the power consumption and processing time increase linearly when the number of entries matching a packet increases.

8. Conclusion and Future Works

In this paper, we model the multimatch packet classification as a concatenated multistring matching problem, which can be solved by traversing a flat signature tree. To speed up the traversal of the signature tree, the edges of the signature tree are divided into different hash tables in both vertical and horizontal directions. These hash tables are then connected together by a they work in parallel when packet classification operations are performed[14]. A perfect hash table construction is also presented, which guarantees that each hash table lookup can be finished in exactly one memory access. Because of the large degree of parallelism and elaborately designed edge partition scheme, the proposed pipeline architecture is able to achieve an ultra-high packet classification speed with a very low storage requirement. Simulation results show that the proposed pipeline architecture outperforms Hyper Cuts and B2PC schemes in classification speed by at least one order of magnitude with a storage requirement similar to that of the Hyper Cuts and B2PC schemes.

References

[1] Hyessook lim, lee, geumdam jin, jungwon Lee, Youngju choi, and changhoon yim, "Boundary cutting for packet classification", vol. 22, no. 2, April 2014.
[2] M. Faezipour and M. Nourani, "Wire-speed TCAM-based architectures for multimatch packet classification," *IEEE Trans. Comput.*, vol. 58, no. 1, pp. 5–17, Jan. 2009.
[3] Snort, "A free lightweight network intrusion detection system for UNIX and Windows," 2013 [Online]. Available: http://www.snort.org
[4] S. Kumar, S. Dharmapurikar, F. Yu, P. Crowley, and J. Turner, "Algorithms to accelerate multiple regular expressions matching for deep packet inspection," in *Proc. ACM SIGCOMM*, New York, NY, USA, 2006, pp. 339–350.
[5] H. J. Chao and B. Liu, *High Performance Switches and Routers*. Hoboken, NJ, USA: Wiley-IEEE Press, 2007.
[6] V. Srinivasan, G. Varghese, S. Suri, and M. Waldvogel, "Fast and scalable layer four switching," *Comput. Commun. Rev.*, vol. 28, pp. 191–202, Oct. 2007.
[7] B.Vamanan,G.Voskuilen, and T.N.Vijaykumar, "Efficuts: optimizing packet classification for memory and throughput," in *Proc. ACM SIGCOMM*, New York, NY, USA, 2010, pp. 207–218.

Licensed Under Creative Commons Attribution CC BY