Comparison Study Of Packet Classification Algorithms In Wired Networks

Marwan Salim Mahmoud **Awos Khazal Ali**
Department of Computer Sciences / College of Education
University of Mosul

Received **Accepted**
01 / 03 / 2011 **28 / 06 / 2011**

الخلاصة

تلعب تقنية تصنيف البيانات دوراً مهما في العديد من خدمات الانترنيت المتقدمة التي تتطلب القدرة على تمييز البيانات في الشبكة وتقسيمها إلى عدة مسارات مختلفة ، مثل الراوترات وخدمات الحماية (كالجدار الناري)، ومرشحات البيانات . تقدم هذه الدراسة مقدمة في آلي ة تصنيف البيانات بشكل موجز ، بالإضافة إلى تصنيف الخوارزميات التي تصنف البيانات في الشبكات إلى مسارات مختلفة . وتقوم هذه الدراسة أيضا بإعطاء شرح موجز عن خوارزم يتان تنتميان إلى نفس الفئة، وهاتان الخوارزميتان (TSS) و(RFC)، وعلاوة عن ذلك تقدم هذه الدراسة مقارنة في الأداء لهاتين الخوارزميتين. تمت عملية التقييم لهاتين الخوارزميتين باستخدام نظام التشغيل Linux Red Hat 5.0 من اجل تحليل ومقارنة الأداء بين الخوارزميتين . والمقارنة تركز على تحقيق وقت تصنيف جيد في عملية التصنيف، ومعدل استهلاك ذاكرة قليل .

ABSTRACT

The packet classification technique play an important role for many internet advanced services that require the capability to distinguish traffic in different flows, such as routers and security services like firewalls and packet filters. This paper provide a brief introduction for packet classification process. Beside, the categorization of the algorithms that classify packets to different flow. Also, this paper give a brief description of two homogenous algorithms (algorithms that belongs to same category), these algorithms are Tuple Space Search (TSS) and Recursive Flow Classification (RFC). Furthermore, this paper provides a comparative

evaluation of these two algorithms (TSS & RFC). The evaluation of these techniques are done under Linux REDHAT 5.0 platform in order to analyze and compare their performance between each other. In particular, the comparison focuses on achieving a good classification time in classification process and, low memory consumption rate.

1. Introduction:

Packet classifiers are essential components of many network utilities, including routers and security services like firewalls and packet filters. A packet classifier inputs a list of rules, each specifying a class of packets matched by that rule. Most of packet classification processes happens in the internet [1]. However, internet content a number of routers interconnected between each other by links. Communication among nodes on the Internet (routers and hosts) takes place using the Internet Protocol, commonly known as IP. IP datagram's (packets) travel over links from one router to the next on their way towards to the final destination. Each router performs a forwarding decision on incoming packets to determine the packet next-hop router. The capability to forward packets is a requirement for every IP router. Additionally, an IP router may also choose to perform special processing on incoming packets as described earlier. Examples of special processing include filtering packets for security reasons, delivering packets to destination hosts, and treating high priority packets preferentially. Such special processing requires that the router classify incoming packets into one of several flows. all packets of a flow obey a predefined rule and are processed in a similar manner by the router. For example, all packets with the same source IP address may be defined to form a flow. A flow could also be defined by specific values of the destination IP address and by specific protocol values. Many algorithms have been proposed so far, and each of these algorithms have advantages and limitation; this study, will refer to categorization of packet classification algorithms that classify packets into flows and describe two algorithms in details [2].

To illustrate the variety of classifiers, this study will consider some examples of packet classification mechanism that can be used by an ISP (Internet Server Provider) to provide different services. Figure 1 shows ISP1 connected to three different sites: enterprise networks E1 and E2 and a Network Access Point1 (NAP), which is in turn connected to ISP2 and ISP3 [2].

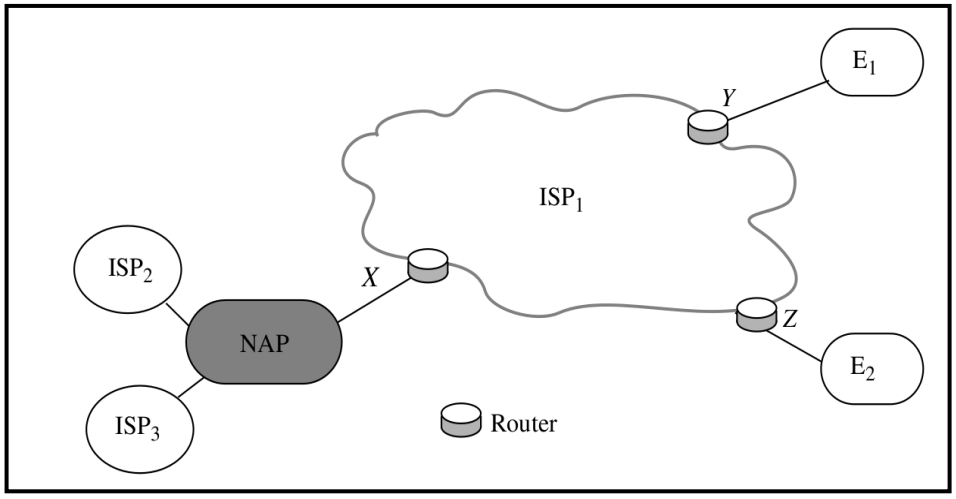


Figure 1: ISP1 connected to three different sites (E1 and E2 and (NAP) (ISP1 and ISP2))[2].

ISP1 provides a number of different services to its customers; Table 1 illustrates some of these services.

Table 1: different services that provide by ISP1

Service	Example
Packet Filtering	Deny all traffic from ISP3 (on interface X) destined to E2.
Policy Routing	Treat all video traffic to E1 (via interface Y) as highest priority and perform accounting for the traffic sent this way.
Traffic Rate Limiting	Ensure that ISP2 does not inject more than 10Mbps of email traffic and 50Mbps of total traffic on interface X.
Traffic Shaping	Ensure that no more than 50Mbps of web traffic is injected into ISP2 on interface X.

There are a number of properties that each algorithm have to desire it for efficient classification process, these proprieties are [2][3]:

- High speed.
- Low storage requirements.
- Flexibility in implementation.
- Ability to handle large real-life routing tables and classifiers.
- Low preprocessing time.
- Low update time.
- Scalability in the number of header fields (for classification algorithms only).
- Flexibility in specification (for classification algorithms only).

2. Existing Packet Classification Algorithms

Packet classification is performed by using a packet classifier and filter, flow classifier, or simply a classifier. A classifier is a collection of rules or policies or includes a set of filters to divide an incoming packet stream into multiple classes and predefine the next hops for packets matching the rules. However, each rule specifies a class that a packet may belong to based on some criterion on **F** fields of the packet header, and associates with each class an identifier, classID. Figure2 illustrate a Simple Example of classifier [3].

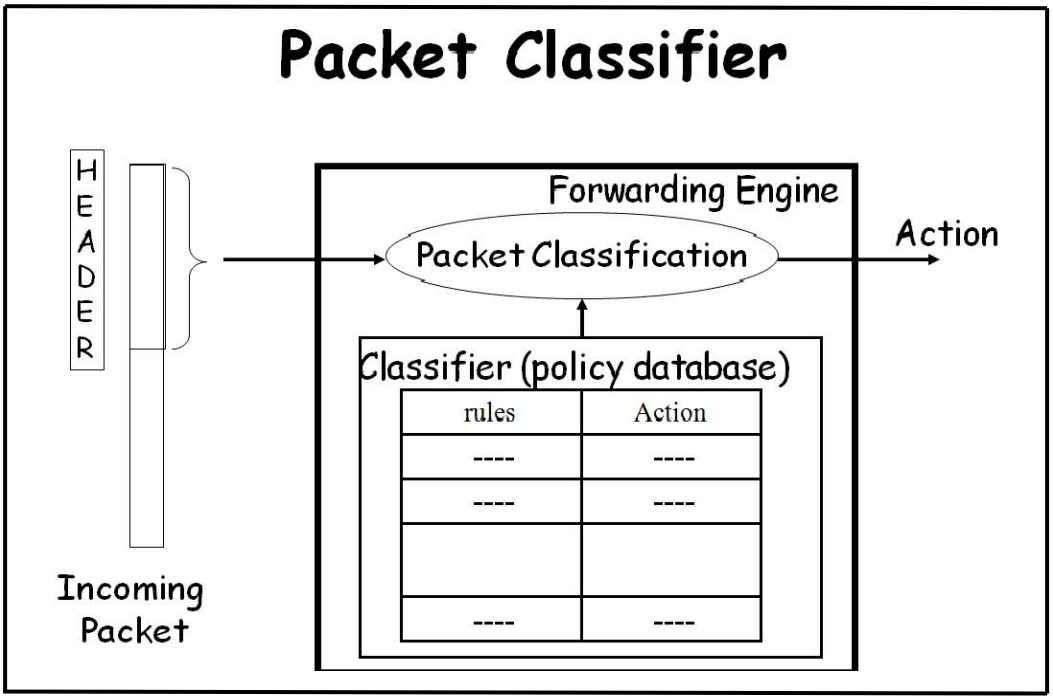


Figure 2: Simple Example Classifier[3].

Moreover, this section will describe the categorization of packet classification algorithms. Packet classifications algorithms can be classified into four kinds based on their mechanism of classify packets , some of these algorithms is software based and some of them is hardware based but these kinds of algorithms (hardware base) are expensive, but at the same time they has a high performance rate and low memory requirement rate [4][5].

On the other hand when number of rules is high, algorithms with hardware base becomes limited and have a lot of weak points. The following section explain the categorization of packet classification algorithms and some examples. Figure 3 illustrate categorization of packet classification algorithms [2].

- Decision-tree Based: such as Linear search, caching, hierarchical tries and set-pruning tries.
- Geometry-based: such as Grid-of-tries, AQT(Area-based quad tree) and FIS (Fat Inverted Segment).
- decomposition-based: such as RFC (Recursive Flow Classification), hierarchical cuttings and (TSS) Tuple-Space Search.
- Hardware only: such as Ternary CAM (Content Addressable Memory) and bitmap-intersection [2].

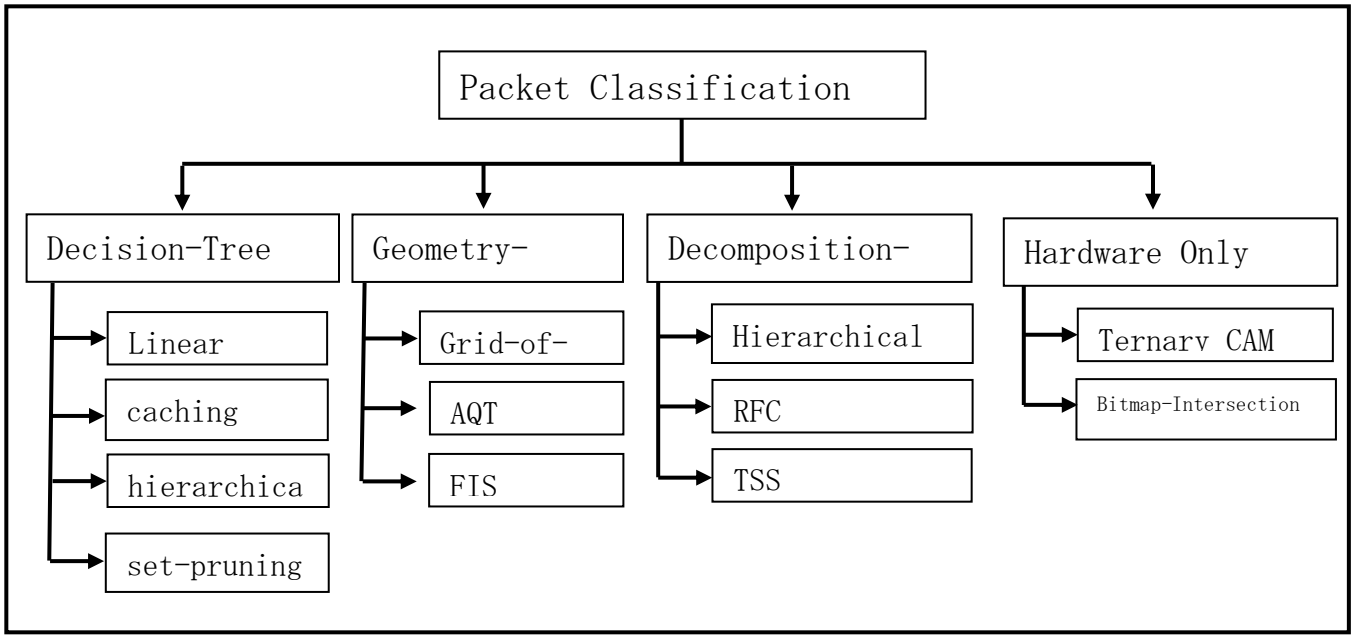


Figure 3: Categorization of packet classification algorithms[2].

This study consider RFC and TSS algorithms to evaluate their performance because both of them are under same category to find out their advantages and limitations. Furthermore, RFC (Recursive Flow Classification) and TSS (Tuple-Space Search) algorithms will describe briefly in way In the following sections.

2.1 Decomposition-based:

Decomposition-based algorithms perform independent search on each field and finally combine the search results from all fields. Such algorithms are use hardware implementation due to their parallel search on multiple fields. However, a huge storage is usually needed to merge the independent search results to obtain the final result. Thus decomposition-based algorithms have poor scalability, and work well only for small-scale rule sets[6].

2.1.1 Recursive Flow Classification (RFC)

RFC is a decomposition-based algorithm, which classifies packets at high speeds. The RFC algorithm has number of stages, each stage consists of a set of parallel memory lookups. Each lookup is a reduction in the sense that the value returned by the memory lookup is shorter (is expressed in fewer bits) than the index of the memory access. The RFC algorithm stages can be describe as follows:

Stage1: In the first stage, the fields of the packet header are split up into multiple parts that are used to index into multiple memories in parallel. The number of packet parts equals 8. Each of the parallel lookups obtain an output value that we will call equivalence class IDs (eqIDs is number of bits which can be encoded using two bits 00^b through 11^b). The contents of each packet part from memory are chosen, so that the result of the lookup is narrower than the index i.e. requires fewer bits to represent [3][8].

For example, in stage1 the packet header is split into 8 parts (see figures 4&5), the memory corresponding to part 6 is indexed using the 2^{16} (where 16 is no. of bits) different values of part 6 (see figure 5). In each location, eq ID placed for this Transport-layer Destination. For example, the value in the memory corresponding to "part 6" is 00. In this way, a 16-bit reduced to two-bit obtained for part 6 in stage0.

Stage2: in the following stages, the index into each packet part of memory is formed by combining the results of the lookups from first stage. For example, the results from the lookups may be such as a chain.

Stage3: In the last stage, one result have been remind from the lookup, because of the way the memory contents have been precomputed, this value represent the class ID of the packet. Figure 4 illustrate all stages of RFC algorithm [7].

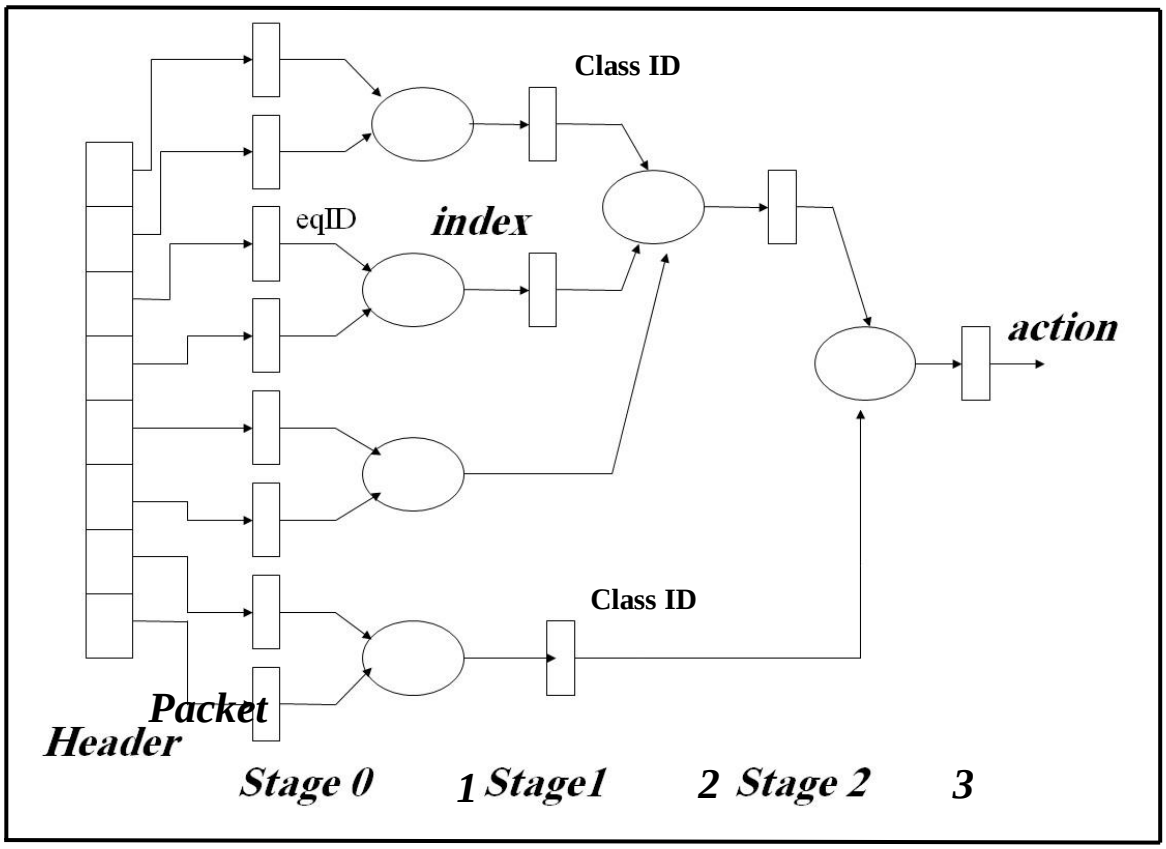


Figure 4: All stages of RFC algorithm[7].

Figure 5: Example of splitting the packet header into parts for the first RFC stage. L3 refers to Network-layer and L4 refers to Transport-layer fields[7].

2.1.2 Tuple-Space Search (TSS)

The tuple Space Search algorithm [8] is type of a decomposition base algorithm. A tuple is defined as an array, the size of array represent the number of fields in a filter. Then the filters can be divided into filter set to the different tuple groups. However, the filters that have the same tuple group having the same tuple specification. Then find the best matched filter through passing the packet classification across all the tuples. If more than

one tuple groups matches, it can be classify the packet by comparing their priorities [9].

For example a filter with 3 fields, the tuple [3, 1, 2] present the tuple specification, the tuple specification can be divided into different tuple groups. So the filters that have the same tuple group have the same tuple specification [2][10].

Since the filters in a same tuple group have the same tuple specification, they are share exclusive and none of them overlaps with others in this tuple group. Now it can be perform the packet classification across all the tuples to find the best matched filter. If multiple tuple groups report matches, the packet classify by the best matched filter by comparing their priorities. However, the filters in a tuple can be easily organized into a hash table, where using the tuple specification to extract the proper number of bits from each field as the hash key. Assume there is no hash overlap in the hash tables. One memory access can determine if there is a matched filter in a hash table so the lookup performance is only determined by the number of tuple specifications. If the hash tables are properly implemented, this algorithm gives an excellent storage performance [11][12], which is linear to the filter set size [13][14].

3. Experiment Results:

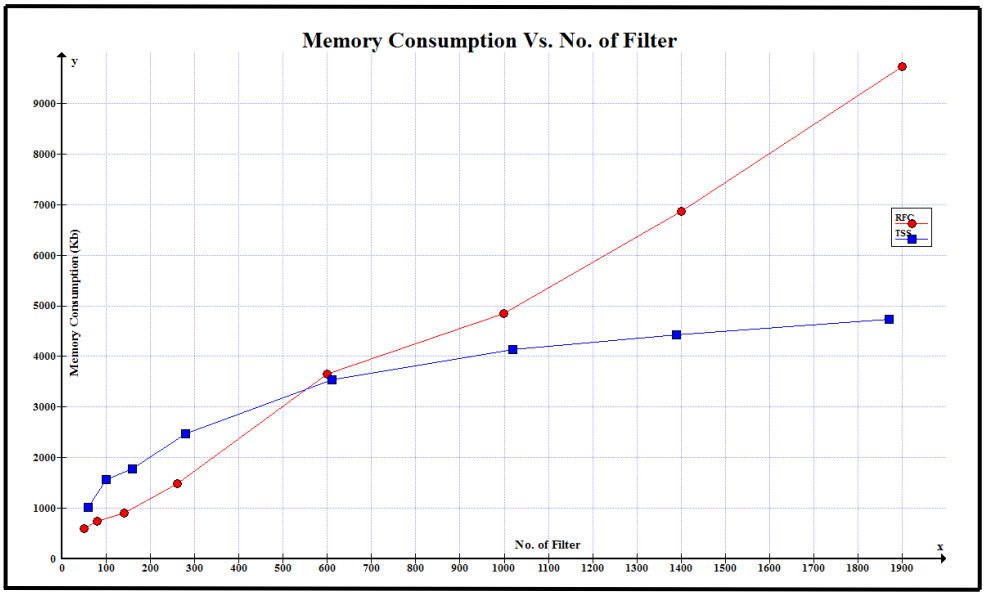
In this section, the experiment has been done by capturing data from the internet for one hour using Winshark software. RFC and TSS algorithms classified the data that has been captured in order to evaluate the performance for both algorithms. However, this experiment aims to giving a homogeneous comparison between two algorithms that categorized under same category, to determine the advantages and limitations for each of the algorithms; and this evaluation considering two kinds of metric that will describe later. RFC and TSS are written in C++ code and running at a PC with 2.4 GHz CPU and 2 GB memory, the algoritms tested under LINUX REDHAT 5.0 environment. Consequently, this study considers the following metrics to evaluate the performance for both algorithms (RFC and TSS).

- **Classification Time**: Faster links require faster classification. For example, links running at 10Mbps can bring 30517 packets per second (assuming minimum sized 40 byte TCP/IP packets) [2]. So, classification time can be defined as the amount of time needed to classify a packet. Therefore each packet must match at least one filter from the filter set, hence searching time to reached the matched filter called classification time .

- **Memory consumption**: Small storage requirements enable the use of fast Memory technologies like SRAM (Static Random Access Memory). Beside, Memory consumption is an excellent indicator of the compression capability of the algorithm measured as the number of rules and number of fields).

The following section discusses the results that had been extracted from the experiment in details.

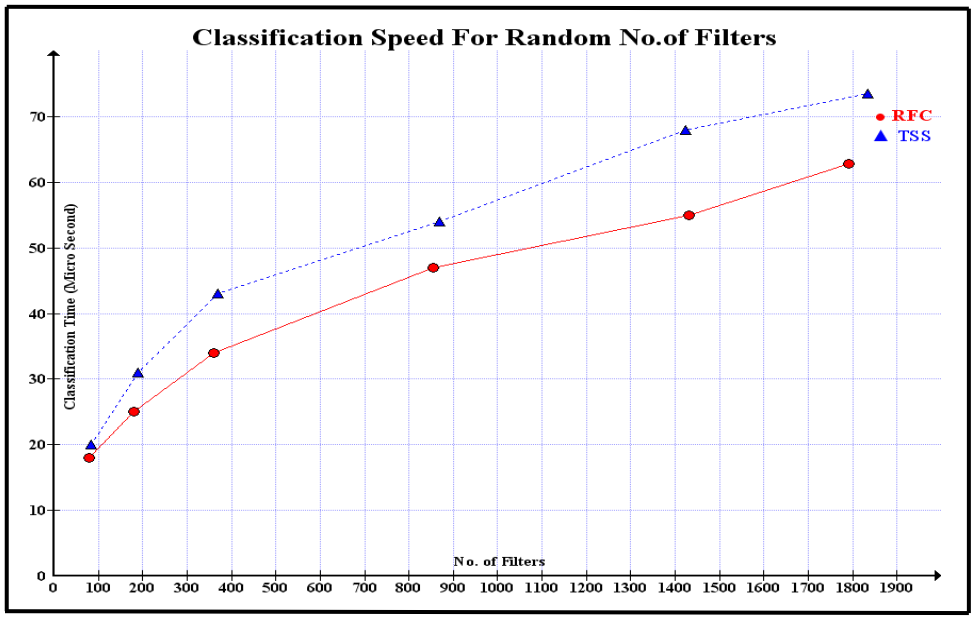
Figure 1 illustrates the memory consumption for both algorithms (RFC & TSS) versus number of filters (filter set =2000) that used in the experiments. Moreover this study consider filter set = 2000 because this number of filters consume more time during classification to clarification the performance of algorithms in term of classification time.



Graph 1: Memory Consumption for (RFC & TSS) Vs. No. of filters.

The above figure shows RFC algorithm consume memory more than TSS algorithm when number of filters are huge but when number of filters are small, RFC achieves less than TSS memory consumption rate in classification process because RFC classify the packets by reducing structure of packets, and hence needs a few memory accesses. Besides, TSS algorithm gives an excellent storage performance when number of filters is huge, which is linear to the filter set size; Thus, TSS is still superior to RFC, using 50% ~ 60% less memories than RFC for larger number of filters.

On the other hand, figure 2 illustrates classification speed for both algorithms (RFC & TSS) versus number of filters (filter set =2000) that used in the experiments.



Graph 2: Classification speed for (RFC & TSS) Vs. No. of filters.

figure 2, shows the performance evaluation result in terms of the classification speed, RFC algorithm achieves outstanding performance compared with TSS algorithm in term of classification speed, that's means RFC algorithm faster than TSS algorithm about (20% ~ 30%) in packet classification process; because RFC required a few memory access in comparison with TSS, Since search for the matching rule (classification speed) is highly dependent on the number of memory accesses; eventually, this few memory access save time in classification process.

4. Conclusion

in this paper the categorization of packet classification algorithms are presented, beside, two algorithms that belong to the same category (Tuple Space Search & Recursive Flow Classification) are describe brief way. Moreover, these two technique are evaluated under Linux REDHAT ver. 5.0 platform in order to analyze and compare there performance between each other. However, the results shows that the RFC algorithm consume memory more than TSS algorithm when number of filters are huge but when number of filters are small, RFC achieves less than TSS memory consumption. On the other hand, RFC algorithm perform faster than TSS algorithm about (20% ~ 30%) in the classification process. The results that extracted by this study did not compare with other results from different studies because the experiment of this study had different parameters and metrics. Furthermore, in future work can be add more algorithms to evaluate their performance and compare it with each other.

REFERENCES

1) FANG Y., et al, *"A Multidimensional Packet Classification Algorithm Based on Network Processors*", Dept. of Electrical & Computer Engineering Florida International University 10555 W. Flagler Street, Miami, Florida 33174, 2009.

2) Gupta P. and McKeown N., *"Algorithms for Packet Classification" Computer Systems Laboratory,* Stanford University Stanford, CA 94305-9030 (2002).

3) Gupta P., Ph.D., Thesis, *"ALGORITHMS FOR ROUTING LOOKUPS AND PACKET CLASSIFICATION",* Department Of Computer Science And The Committee On Graduate Studies, University Of Stanford, (2000).

4) Lee C. L., Chan C. T., and Chang H. Y., *"Performance Improvement of Two-Dimensional Packet Classification by Filter Rephrasing",* ieee/acm transactions on networking, vol. 15, no. 4, august 2007.

5) Antoš D., Ph D., Thesis, *"Hardware-constrained Packet Classification",* College Of Informatics, University of Masaryk (2006).

6) Jiang W. and Prasanna V. K., *"Large-Scale Wire-Speed Packet Classification on FPGAs"* .

7) Gupta P. and McKeown N., *"Packet Classification on Multiple Fields*", Computer Systems Laboratory, Stanford University Stanford, (2000) CA 94305-9030.

8) V. Srinivasan, et al. *"Packet Classification using Tuple Space Search",* ACM SIGCOMM, Sept. 1999.

9) Sahasranaman V. and Buddhikot M. M., *"Comparative Evaluation of Software Implementation of Layer-4 Packet Classification Schemes"*, (2006).

10) Lee F., and Shieh S., *"Packet classification using diagonal-based tuple space search"* (2003).

11) Sum X., Sahni S. K. and Zhao Y. Q., *"Packet Classification Consuming Small Amount of Memory",* (2008).

12) Kennedy A., Bermingham D., and etc., *"Power Analysis Of Packet Classification On Programmable Network Processors"* IEEE International Conference on Signal Processing and Communications (2007).

13) Srinivasan V., Suri S. and Varghese G., *"Packet Classification using Tuple Space Search "*, (2001).

14) varenni G., Stirano F, and etc., *"Comparative Evaluation of Packet Classification Algorithms for Implementation on Resource Constrained Systems",* (2009).