Fast Packet Classification Algorithms

Mrudul Dixit, Anuja Kale, Madhavi Narote, Sneha Talwalkar, and B. V. Barbadekar

***Abstract*—A packet classifier possesses a set of rules for classifying packets based on header fields. To classify a packet belonging to a particular flow or set of flows, network nodes like routers or firewalls must perform a search over a set of filters using multiple header fields of packet as a search key. Routers classify packets to determine their respective flow and the services they should receive. The paper deals with fast packet classification algorithms, Recursive Flow Classification (RFC) and Hierarchical Space Mapping (HSM). Packet classification is based on header fields of packet. RFC and HSM deal with header fields namely source and destination IP addresses as well as source and destination port number. Using those header fields mapping tables are computed and finally a decision is made about packet classification of individual packet. The RFC and HSM algorithms are implemented and the analysis of space required and time taken for classification is done.**

***Index Terms*—Hierarchical space mapping, IP address, packet classification, port number, recursive flow classification.**

I. Introduction

The process of categorizing packets into "flows" in an Internet router is called packet classification. All packets belonging to the same flow obey a predefined rule and are processed in a similar manner by the router. Packet classification is an enabling function for a variety of internet applications including Quality of service (QoS), security, monitoring, multimedia Communications [1]. Growing and changing network traffic requirements invokes need of larger filter with more complex rules, which in turn gives rise to different fast packet classification algorithms. Packet classification is needed for non-best-effort services, such as firewalls and intrusion detection, routers, ISPs and usually in the most computation intensive task among others. Services such as bandwidth management, traffic provisioning, and utilization profiling also depend upon packet classification. Packet consists of header and information data and header consists of MAC address, IP address, port number etc.

Traditionally, the Internet provided only a "best-effort" service, treating all packets going to the same destination identically, servicing them in a first come-first-served manner. However, internet users and their demands for different quality services are increasing day by day. So, Internet Service Providers are seeking ways to provide differentiated services (on the same network infrastructure) to different users based on their different requirements and expectations of quality from the Internet. For this, routers need to have the capability to distinguish and isolate traffic belonging to different flows. The ability to classify each incoming packet to determine the flow it belongs to is called packet classification and could be based on an arbitrary number of fields in the packet header. Packet classification is a multi-dimensional form of IP lookup and finding longest prefix matching to provide next-hop in routers.[2]

Manuscript received August 16, 2012; revised October 7, 2012.

Mrudul Anand Dixit is with Cummins College of Engineering for Woma n, India (e-mail: dixitma@yahoo.com).

II. Related Work

In this section, we describe related research work on packet classification algorithms. There are various packet classification algorithms proposed so far. (Refer Fig. 1) [3], [4], [5].Algorithms for packet classification can be categorized on various bases such as

1) Hardware based: They use Ternary content addressable memories (TCAMs).
2) Software based: Trie base, Decision tree, Hash based etc.

Different algorithms for packet classification are as follows:

- GoT: Grid of Tries
- EGT: Extended Grid of Tries
- HiCuts: Hierarchical intelligent Cuts
- HSM: Hierarchical Space Mapping
- AFBV: Aggregated and Folded Bit Vector
- CP: Compression Path
- RFC: Recursive Flow Classification
- B-RFC: Bitmap aggregation Recursive Flow Classification
- H-Tries: Hierarchical tries
- SP-Tries: Set Pruning tries
- BV: Bit Vector
- ABV: Aggregated Bit Vector

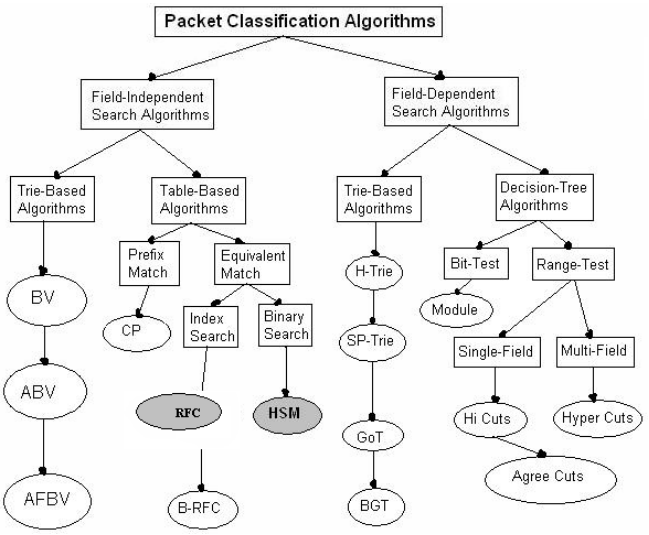

Fig. 1. Packet classification algorithms.

A. Hardware Based Packet Classification

A high degree of parallelism can be implemented in hardware to increase the speed of classification. This can be

achieved by using Ternary content addressable memories(TCAMs).But TCAMs cannot be used where flexible filter specifications are required as well as they have high power consumption and low scalability.[6], [7]

B. Software Based Packet Classification

Tri based algorithms has memory requirement of O(NW) and requires 2W-1 memory access per lookup, where N is number of filters and W is length of IP address [8]. In [9], area based quad tree (AQT) was proposed for two field filters.AQT supports efficient update time.The performance of trie-based algorithms are studied in [10].

Schemes using decision tree to categarise filters into multiple sets is presented in papers [6] and [4]. The number of filters in each set is limited by predefined values and linear search is used to traverse the filter set.

In [8], [11] the mechanism called cross producting, involving BMP lookups on individual fields and use of precomputed table to combine results of individual prefix lookups is presented. But in this scheme the memory requirements increase with the number of fields, O (N^k) where k is number of classified fields.

The hash based idea [12] has given rise to 2-D filters. The filters with specific prefix length are grouped into a tuple, each tuple is then concatenated to create a hash key which is used for the tuple lookup. The matched filter can be found by probing each tuple alternately while tracking the least cost filter.

C. Software Based Packet Classification

Tri based algorithms has memory requirement of O(NW) and requires 2W-1 memory access per lookup, where N is number of filters and W is length of IP address [8]. In [9], area based quad tree (AQT) was proposed for two field filters.AQT supports efficient update time.The performance of trie-based algorithms are studied in [10].

Schemes using decision tree to categarise filters into multiple sets is presented in papers [6] and [4]. The number of filters in each set is limited by predefined values and linear search is used to traverse the filter set.

In [8], [11] the mechanism called cross producting, involving BMP lookups on individual fields and use of precomputed table to combine results of individual prefix lookups is presented. But in this scheme the memory requirements increase with the number of fields, O (N ^{k}) where k is number of classified fields.

The hash based idea [12] has given rise to 2-D filters. The filters with specific prefix length are grouped into a tuple, each tuple is then concatenated to create a hash key which is used for the tuple lookup. The matched filter can be found by probing each tuple alternately while tracking the least cost filter.

III. Implementation of RFC Algorithm

RFC is decomposition based algorithm i.e.it starts computing multiple fields and ends with single field result.RFC uses different fields from header of the packet for classification as that of the HSM**.** Eight different header fields can be considered while classifying a packet in a router. Those 8 fields are source IP address (SP), Destination IP address (DA), source port number (SP), destination port number (DP), type of service (TOS), type of protocol used, Protocol field, Protocol flag etc.We have implemented RFC using four fields out of eight mentioned above .Those four fields are Source IP address (SP), Destination IP address (DA), Source Port number (SP), Destination Port Number (DP). In RFC standard rule set named Policy Table is predefined (Refer table IV).

Each incoming packet in a network at router is compared with standard rule set and it is checked that whether it lies in the rule set or not. If the packet lies in the rule set, then appropriate action whether to accept or deny the packet is taken. For computing rule set four header fields which are mentioned above are taken into consideration. For each policy or rule, different ranges of source IP address (SP), Destination IP address (DA), source port number (SP) ,destination port number (DP) are decided and particular action is permitted for each policy.

We have used a rule set table with three different policies. The ranges of SA, DA, SP, DP can be either overlapping or non-overlapping. Overlapping ranges in rule set will reduce the 'don't care' (packet does not belong to any rule set condition) condition. The common factors between HSM and RFC are rule set, Header fields considered for classification and number of phases**.** The difference between RFC and HSM lies in the method of computation. In HSM, first bitmaps for different header fields are computed and later their ANDing is done to get final policy look up table with ultimate result of classification. While in RFC, after calculating class bitmap, index values are computed.

The advantage of RFC is that, formulae for computing index values can be changed or modified at Internet Service provider (ISP) end which means that any change in the rule set can be successfully reflected in the packet classification.RFC classifies packet in multiple phases. (Refer Fig. 2) [13]. In classifying packet RFC forms class bitmaps, eqIDs and index tables.

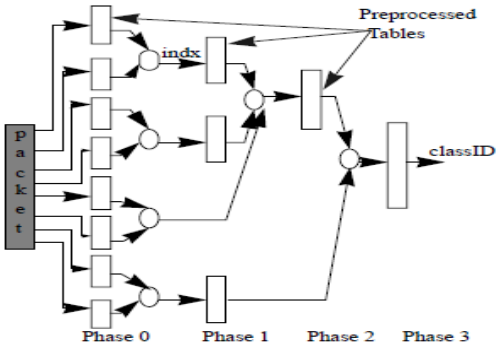

Fig. 2. Packet flow in rfc.

A. Procedure for Step By Step Generation of Policy Lookup Table for RFC

1) IP addresses fragmentations

IP address decomposition is done for both source address space and destination address space respectively .The fragmentation is done in the same manner for both source and destination IP addresses. For each address range (including address or subnet) appeared in the policy table, its two

boundaries IP addresses are marked down in the corresponding source address or destination address IP space (Refer Fig. 2) [1].

After completion of construction of policies in the policy table, for each segment that is following at least one policy falls in it, an Equivalence class ID (eqID) number is assigned in the ascending order along the direction ofincreasing IP address, starting from 0.

There are many ways to map a given IP address (i.e. the source or destination IP address of a received packet) to a segment. In RFC, this is achieved by taking any number of chunks that are convenient. We have kept the number of segments same at phase0 for all fields as that in HSM so that preprocessing time is not taken into consideration during analysis.

2) Port number fragmentations

The principle of port number fragmentation to get port sequence number is similar to IP address fragmentation. For each segment that is following at least one policy from the policy table, an address sequence number is assigned in the ascending order along the direction of increasing port address, starting from 0. For the port number mapping, a direct look-up table (216 or 65536 in size) is sufficient and usually more efficient when there is enough memory to be allocated.

3) Generation of phase1 tables

Each eqID is assigned a class bit map number(CBM).The index into each memory is formed by combining the results of the lookup tables from earlier phases, that is similar to the ANDing of BM in HSM, ANDing of CBM is carried out for respective eqID. But index value for chunks in phase1 is computed using eqIDs of the chunks from the previous phase using the following formulae:

$$Index1=C00\times1+C01\times4\ \ldots C00=Chunk\ 00$$

Formula derivation: Since at phase0 chunk 01 we have 4 eqIDs and at chunk 02 we have 3eqIDs, taking into account all the possibilities for eqIDs we have derived the above formula. It can be seen that the formula is thus completely heuristic. Following is the formula for computing index values of chunk 02 of phase1:

$$Index2=C02\times1+C03\times4$$

Every entry corresponding to an index in phase1 is the eqID in that phase and it is computed using the eqIDs of the previous phase, hence the name 'Recursive Flow'.

4) Generation of phase2 tables

$$Index3=C00\times7+C01\times1$$

This formula is used to find the index values in phase2. Every entry to an index is the policy to which the packet belongs. Using these index values finally an action is taken on a packet.

IV. Implementation of HSM Algorithm

HSM classifies packets using four parameter based policy lookup table. The four parameters considered in the policy lookup table are destination IP address (DA), source IP address (SA), destination port number (DP), and source port (SP) number. The basic idea of the HSM algorithm is to reduce the searching fields by mapping the lookup domains two-to-one, step by step and hierarchically. It is shown in Fig 3. [1], [7]

A. Steps for Packet Clasisfication Using HSM

1) In HSM the 2 IP address fields (DA, SA) and the 2 port numbers (DP, SP) are mapped into non-overlapped segments properly. For this the network address ranges and port number ranges are used as per the rule set (refer Table IV). It divides the original four-dimension space into a two dimensional space by looking up the following two tables:
   a) AMT — source/destination IP address mapping table
   b) PMT — source/destination port number mapping table
2) The two-dimension table resulted from the previous step is transformed to the one-dimension policy table. This is done by looking up the third table i.e. PLT - policy lookup table.

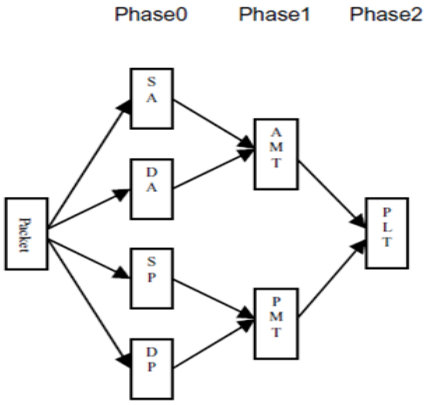

Fig. 3. Packet flow in hsm.

B. Procedure for Step By Step Generation of Policy Lookup Tables for Hierarchical Space Mapping

1) IP addresses fragmentations

IP address fragmentation for HSM is same as that of RFC. After completion of construction of policies in the policy table, for each segment that is following at least one policy falls in it, an Address Sequence Number (ASN)is assigned in the ascending order along the direction of increasing IP address, starting from 0.

2) Port Number fragmentations

The principle of port number fragmentation to get Port Sequence Number (PSN) is similar to IP address fragmentation. For the port number mapping, a direct look-up table is more efficient when there is enough memory to be allocated.

3) Generation of address mapping table (AMT)

For IP address segmentation, a bitmap is assigned for each address sequence number. The bit map has one bit for each policy in the policy table. Each entry of address mapping port is given an address group number (AGN) according to the order of its appearance, along with a bit map tagged to it. The bit map is formed by an AND operation of the two bit maps of source address and destination address respectively.

TABLE I: AMT STRUCTURE AND SETUP.

AMT	SA#0	SA#1	SA#2	SA#3
DA#0		1,2	1,2	1
DA#1	0	0,1,2	1,2	1
DA#2		2	2	

4) Generation of port mapping table (PMT)

The generation of port mapping table is identical to address mapping table that is the bit map has one bit for each policy in the policy table. Each entry of port mapping table is given a port group number (PGN) according to the order of its appearance, along with a BM tagged to it. The bit map is formed by an AND operation of the two bit maps of source address and destination address. The Bit Maps are not physically stored in lookup table; they are only used in the setup of lookup tables. The bit maps will be released after establishing look-up table.

5) Generation of policy lookup table (PLT)

Each entry of Policy Lookup Table (PLT) is filled with a policy number. The Bit Maps of address group number and port group numbers are combined and then thepolicynumber of the highest priority is picked out. Sample policytable is shown below.

TABLE II: PMT STRUCTURE AND SETUP.

PMT	SP#0	SP#1	SP#2	SP#3
DP#0		1	1	
DP#1	0	0,1	1,2	0,2
DP#3		1	1,2	2

TABLE III: PLT STRUCTURE AND SETUP.

PLT	AGN#0	AGN#1	AGN#2	AGN#3	AGN#4
PGN#0	1	1		1	
PGN#1			0	0	
PGN#2	1	1	0	0(1)	
PGN#3	1(2)	1	0	0(1,2)	2
PGN#4	2		0	0(2)	2
PGN#5	1(2)			1(2)	2
PGN#6	2	1		2	2

TABLE IV: RULE TABLE USED FOR IMPLEMENTATION OF RFC AND HSM ALGORITHMS.

Rule	SA Range	DA Range	SP Range	DP Range	Action
0	0.0.0.0~64.0.0.0	32.0.0.0~64.0.0.0	0~65535	128~256	Deny
1	32.0.0.0~255.255.255.255	0.0.0.0~64.0.0.0	64~256	0~65535	Permit
2	32.0.0.0~128.0.0.0	0.0.0.0~255.255.255.255	128~65535	128~65535	Deny

V. RESULTS AND ANALYSIS

Both the algorithms, RFC and HSM are implemented. We analyzed them on the basis of memory requirement and processing time requirement for a predefined rule set table. We found following results.

1) Memory requirement

HSM - 4.26Kilobytes RFC - 9.89kilobytes

Hierarchical Space Mapping (HSM) required less memory as compared to Recursive Flow Classification (RFC).

2) Processing time required

Time required for RFC: 0.1497002330 seconds

Time required for HSM: 0.4096179104 seconds

The processing time required for Recursive Flow Classification algorithm is less than that of Hierarchical Space Mapping algorithm

VI. LIMITATIONS

If the number of policies in the rule set increase then more memory will be required to store the rule set.

VII. CONCLUSION

- Due to ordered and overlapping policies, packet classification on multiple fields cannot be done by policy sorting prior to policy lookup. To achieve high performance policy lookup, special hardware such as TCAMs can be applied but they introduce additional cost.
- HSM and RFC Algorithms provide a generic solution that can be implemented either in software or hardware, with balanced time and space computational complexity.
- The HSM and RFC can be applied to general cases of multiple field classification problems where sorting and caching do not help.
- HSM and RFC leverages on reduction of structure redundancy.
- HSM consumes much less memory space while keeping the average lookup time on the same order as RFC.
- Both the algorithms can work well with the IPV6.
- The processing time required for RFC is less than that of HSM

REFERENCES

[1] B. Xu, D. Jiang, and J. Li, "HSM: A fast packet classification algorithm," *19th International Conference on Advanced Information Networking and Applications (AINA'05)*, vol. 1, pp. 987-992, 2005.

[2] T. V. Lakshman and D. Stiliadis, "High-speed policy-based packet forwarding using efficient multidimensional range matching," in *ACM SIGCOMM'98*, September 1998.

[3] A. Feldmann and S. Muthukrishnan, "Tradeoffs for packet classification," *AT & T Labs Research Florham Park, NJ*.

[4] T. Y. C. Woo, "A modular approch to packet classification: Algorithms and results," *Bell Laboratories, Lucent Technologies*.

[5] H. Kim, K. C. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee, "Internet traffic classification demystified: Myths, caveats, and the best practicies," *ACM Conext*, 2008.

[6] P. Gupta and N. McKeown, "Packet classification using hierarchical intelligent cuttings," in *Hot Interconnects VII*, August 1999.

[7] P. Gupta and N. McKeown, *Algorithms for Packet Classification, IEEE Network*, 2001.
[8] V. Srinivasan, G. Varghese, S. Suri, and M. Waldvogel, " Fast scalable level four switching," in *Proc. ACM SIGCOMM*, Sep. 1998, pp. 191-202
[9] M. Buddhikot, S. Suri, and M. Waldvogel, "Space decomposition techniques for fast layer-4 switching," in *Proc. IFIP 6th Int. Workshop on Protocols for High Speed Networks*, Aug. 1999, pp. 25-41.
[10] V. Sahasranaman and M. Buddhikot, "Comparative evaluation of software implementation of layer 4 packet classifcation schemes," in *Proc. IEEE ICNP*, Nov. 2001, pp. 220-228.
[11] D. E. Taylor, "Survey and taxonomy of packet classification techniques," *WUCSE* – 2004.
[12] V. Srinivasan, G. Varghese, and S. Suri, "Packet classifcation using tuple space search," in *Proc. ACM SIGCOMM*, Sep. 1999, pp. 135-146.
[13] P. Gupta and N. McKeown, "Packet classification on multiple fields," in *ACM SIGCOMM*, Sept. 1999, pp. 147-160.