2004 년도 대한전자공학회 하계종합학술대회 논문집 27 권 1 호

Parallel Multiple Hashing for Packet Classification

정여진, 김혜란, 임혜숙*

이화여자대학교, SoC 설계 연구실
전화 : 02-3277-2809

Parallel Multiple Hashing for Packet Classification

Yeojin Jung, Hyeran Kim, Hyesook Lim*

SoC design Lab
Information Electronics Engr., Ewha Womans University
*E-mail : hlim@ewha.ac.kr

Abstract

Packet classification is an essential architectural component in implementing the quality-of-service (QoS) in today' s Internet which provides a best-effort service to all of its applications. Multiple header fields of incoming packets are compared against a set of rules in packet classification, the highest priority rule among matched rules is selected, and the packet is treated according to the action of the rule. In this paper, we proposed a new packet classification scheme based on parallel multiple hashing on tuple spaces. Simulation results using real classifiers show that the proposed scheme provides very good performance on the required number of memory accesses and the memory size compared with previous works.

I. 서론

인터넷은 모든 패킷을 동등하게 처리하는 것을 기본으로 하며 이를 best-effort-service 라 한다. 그러나 인터넷을 통한 여러 가지 어플리케이션들이 증가하고 ISP 들이 사용자에 따른 차별화된 서비스를 필요로 함에 따라 라우터는 좀 더 복잡한 기능이 필요하게 되었다. 이러한 서비스를 위해서는 우선적으로 라우터에서 패킷을 중요도에 따라 구분하는 작업이 필요하다. 패킷을 구분하기 위한 각각의 기준을 rule 이라고 하고 각 rule 은 destination address 뿐 아니라 source address, destination port number, source port number, protocol type 등 여러 필드로 구성되어 있다. 그리고 priority 에 따른 rule 의 집합을 classifier 라 한다. Packet Classification 은 이 classifier 중에서 가장 priority 가 높은 rule 을 찾는 과정을 말한다.

Packet classification 알고리즘은 다음과 같은 사항을 고려하여 설계해야 한다. 가장 중요한 부분으로는 검색 시간과 memory 크기를 들 수 있다. 다음으로는 어떠한 크기의 classifier 에도 일정한 성능을 보장할 수 있어야 하고, 여러 개의 필드를 고려하는 것을 가능하게 하면서 동시에 여러 match 방식을 잘 적용할 수 있어야 한다. 또한 classifier 를 build 할 때 소요되는 시간과 rule 의 변화가 자주 일어나는 경우 update 할 때 걸리는 시간도 최소화 할 수 있는 알고리즘을 설계해야 한다. 본 논문에서는 빠른 검색 시간뿐 아니라, n 을 rule 의 개수라 할 때 memory 사용을 O(N)으로 할 수 있고, 여러 가지 필드의 검색이 가능한 효율적인 packet classification 구조를 제안한다.

본 논문의 구성은 다음과 같다. 2 장에서 packet

classification 을 위한 기존의 방법들을 살펴보고 3 장에서 본 논문에서 제안하는 구조를 설명하고 4 장에서 simulation 결과를 평가한 후 5 장에서 결론을 맺는다.

II. 기존의 Packet Classification 구조

기본적인 data 구조를 이용한 방법으로는 linear search 와 trie 구조가 있다. Linear search 는 priority 순서에 따라 순차적 검색을 하는 방식이고 trie 는 여러 field 에 따라 trie 를 구성하고 그것을 순차적으로 연결하여 search 하는 방식이다. 그러나 단순히 field 별로 trie 를 구성하여 연결하면 backtracking 이 발생하므로 [1]의 set pruning trie 에서는 rule 을 longest prefix 에 복사하여 저장하는 방식을 제안하고 [2]의 grid of trie 에서는 switching pointer 를 두어 backtracking 없이 가장 적합한 rule 을 찾는 구조를 제안하였다.

기하학적 구조를 기반으로 한 방법으로 [2]의 corss-producting 이 있다. 이것은 2 차원 평면에서 각 prefix 를 range 로 전환하여 두 field 에서 match 하는 range 를 미리 계산된 table 의 index 로 하여 rule 을 찾는 구조이다. 이 구조는 검색 시간은 빠르지만 메모리가 많이 필요한 단점을 갖고 있다. 경험적 특성을 이용한 구조로, [3]에서는 classifier 의 실제 rule 을 분석해 본 결과 tuple 의 종류가 다양하지 않다는 점에서 착안하여 해쉬 테이블을 구성함으로써 prefix match 문제를 exact match 문제로 전환하였다. Classifier 의 모든 tuple 쌍에 각 entry 를 저장하여 tuple 순서대로 해쉬를 이용해 한번에 검색한다. 경험적 특성을 이용한 구조는 classifier 의 특성에 따라 nondeterministic 하다는 단점을 갖고 있다.

다음 하드웨어를 이용한 방식으로 TCAM 을 들수 있다. 이것은 longest prefix match 를 위한 IP address 검색에서 매우 널리 쓰이는 메모리의 형태로 packet classification 을 위해 TCAM 의 각 entry 에 classifier 의 모든 field 의 prefix 를 concatenating 한 형태로 priority 순서대로 저장한다. TCAM 은 검색속도가 매우 빠르다는 장점을 가진다. 그러나 한 개의 칩에 72bit 의 128k entry 나 144bit 의 64k entry 정도 밖에 저장하지 못하고 range 검색에 매우 불리한 단점을 갖고 있다.

III. 제안하는 Packet Classification 구조

본 논문에서는 [4]에서 이용한 parallel multiple hashing 을 packet classification 에 적용하여 destination IP address 와 source IP address 에 대해 첫번째 검색을 하고 걸러진 rule 들을 나머지 field 에 관해 rule table 에서 검색하는 PMH_PC 방식을 제안한다.

3.1 Parallel Multiple Hashing

Parallel multiple hashing 은 [4]에서 address lookup 을 위해 이용한 방법이다. Multiple hashing 은 한 prefix 길이에 대해 하나의 hash index 가 나오는 것이 아니라 2 개의 hash index 를 뽑아서 2 개의 table 중에서 load 가 적게 걸린 table 에 저장하는 방식이다. 이렇게 하므로써 같은 hash 값을 갖는 entry 들을 분산시켜 저장하므로 collision 을 줄일 수 있게 된다. 또한 parallel 은 hashing 을 위해 prefix 길이별로 만든 table 을 parallel 하게 검색하는 것을 말한다.

그림 1 은 parallel multiple hashing 의 구조를 나타낸다. Destination address 가 들어오면 CRC hash 함수를 이용해 각 prefix 길이 별로 2 개의 hash index 를 만들고 각 index 가 가리키는 table 로 가서 match 하는지를 찾는다. 본 논문에서는 이 방식을 main memory 검색에 이용한다.

3.2 Semi-Tuple Space

[4]의 address lookup 에서는 prefix 길이에 대해 8~32 까지 25 개의 tuple 로 나누었지만 packet classification 에 동일한 방식을 적용하게 되면 destination address 와 source address 를 모두

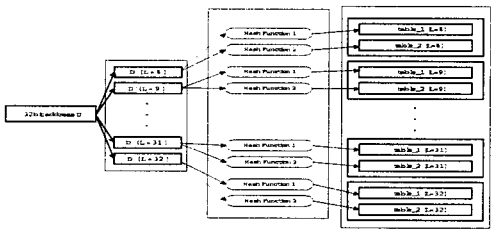

그림 1. Parallel multiple hashing 구조

고려해야 하기 때문에 wildcard 를 제외하고도 25*25=625 개의 tuple 을 가져야 한다. [3]에서는 classifier 를 분석해 본 결과 실제 tuple 의 수는 많지 않다는 가정 하에 classifier 가 같는 tuple 에 대해서만 table 을 나누었다. 그러나 그렇게 하면 classifier 의 특성에 따라 구조나 성능의 차이가 발생하게 된다. 따라서 본 논문에서는 wild card 를 포함해 0, 8, 16, 24, 32 의 5 개의 prefix 길이를 고려하여 25 개의 tuple 로 grouping 을 하였다. 이 경우 각 tuple 에 포함되는 prefix 는 0~7, 8~15, 16~23, 24~31, 32 가 된다.

3.3 Table 의 구성

본 논문에서 제안하는 구조는 main table 과 rule table 의 두가지 table 을 가진다. Hash index 를 통해 첫번째로 접근하게 되는 main table 의 entry 에는 각 tuple 길이의 destination prefix, source prefix 그리고 match 했을 때 검색해야 할 rule table 의 index 를 저장하고 있다. 이 구조는 multiple hashing 을 사용하므로 두개의 main table 로 구성되고 각 main table 은 3 개의 entry 를 갖는다.

Rule table 은 tuple 길이 보다 긴 prefix 의 나머지 부분과 다른 field 들, 그리고 rule number 가 저장되어 있다. 같은 main table entry 에 해당하는 rule 들이 여러 개가 있을 수 있는데 이 rule 들은 효율적인 검색을 위해 priority 순서대로 linked list 로 연결되어 있다.

3.4 Search Procedure

이 구조의 검색 방법은 main table 을 검색하는 첫 번째 단계와 rule table 을 검색하는 두 번째 단계로 나누어진다. 그림 2 에서와 같이 첫 번째 단계에서 들어온 destination address 와 source address 를 tuple 에 맞게 조합하여 XOR 하고 그것을 CRC hashing 함수의 입력으로 준다. 그 다음 hash index 를 이용하여 main table 에 일치하는 prefix 가 있는지 검색한다. 첫번째 단계에서 일치하는 prefix 가 있는 tuple 은 일치하는 entry 가 가리키는 rule table 의 위치로 가서 rule table 을 linked list 를 따라가면서 검색한다. linked list 는 priority 가 높은 것부터 낮은 순으로 정렬 되어 있으므로 검색 중에 match 하는 rule 이 있으면 검색은 중단된다. 또한 다른 tuple 에서 match 하여 나온 rule 이 다른 rule table 에서 검색 중인 rule 보다 priority 가 높은 rule 이면 검색을 끝낸다. 각 tuple 에서 나온 rule 들 중에서 priority 가 높은 rule 이 최종적으로 선택된다.

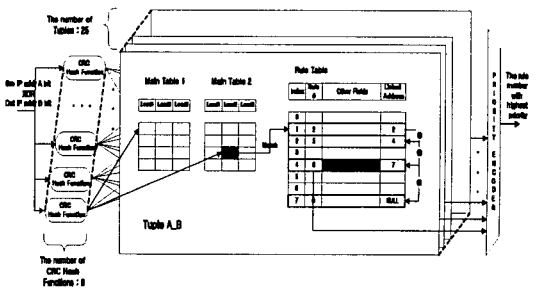

그림 2. Two Step Process

3.5 Building Procedure

Building 하는 과정은 search 과정과 비슷하다. Rule 의 prefix 길이에 따라 hash 함수를 통과시켜 해당하는 main table 로 가서 match 하는 entry 가 없으면 두개의 table 중에서 load 가 적게 걸린 쪽으로 rule 은 저장하고 나머지 field 는 next rule table pointer 가 가리키는 곳으로 가서 저장한다. 그리고 match 하는 entry 가 있으면 rule table 의 address 를 따라 검색하여 적합한 priority 위치에 나머지 랴딩를 저장하고 linked list pointer 를 update 한다. Update 도 build 와 같은 순서로 수행된다.

IV. 시뮬레이션 결과 및 비교

본 논문에는 제안한 구조의 시뮬레이션을 위해 실제 라우터에서 사용되는 4 개의 classifier 를 사용하였다. 검색을 위해서 rule 과 동일한 입력을 주고 wildcard 인 rule 을 실험하기 위해 random 하게 발생시킨 입력을 또한 가하였다.

표 1 은 simulation 결과로 rule 에 따라 최대 메모리 접근 횟수가 11~49, 평균 메모리 접근 횟수가 4~30 임을 알 수 있다. 또한 이 구조에서 필요한 memory 크기는 78k~183k 으로 rule 을 한번씩만 저장하면 되므로 O(N)의 memory 를 필요로 한다.

표 1 Simulation Results

Rule	Number of Rules	Max. Acc.	Avg. Acc.	Mem. size
acl00	2250	11	9	104k
acl67	2375	38	4	117k
acl01	2783	43	9	78k
acl61	4710	49	30	183k

그림 3 은 [5]에서 발췌한 데이터와 본 논문에서 수행된 시뮬레이션 결과를 바탕으로 각 구조들의 성능을 비교한 것이다. 다른 구조들이 메모리 크기와 접근 횟수가 거의 반비례 관계에 있는 것에 비해 본 논문에서 제안한 PMH_PC 구조는 메모리 크기와

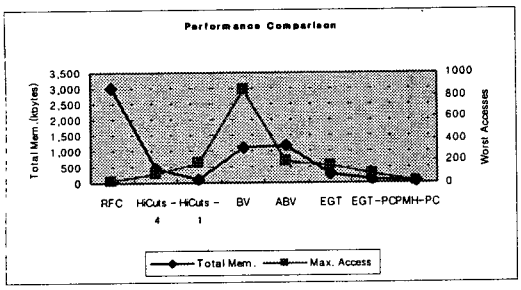

그림 3. Performance Comparison

메모리 접근 횟수 모두 우수한 성능을 보이는 것을 알 수 있다.

V. 결론

본 논문에서는 IP address lookup 을 위한 구조인 parallel multiple hashing 을 패킷 분류에 확장하여 packet classification 에 적용한 구조를 제안하였다. Packet classification 을 위한 기존의 방법들은 다양한 연산을 요구하는 다차원 검색을 지원하지 못하고 프리픽스 연산에만 치중한 경우가 많았고, 검색 속도와 소요되는 메모리 크기 사이에서 절충을 취해야 하는 제한점을 가졌다. 본 논문에서 제안한 구조는 이러한 문제들을 해결하면서도 N 개의 룰을 위하여 O(N)의 메모리를 사용하면서도 빠른 검색을 수행하는 성능을 보였다.

References

[1] P. Tsuchiya, "A search algorithm for table entries with non-contiguous wildcarding," Bellcore, internet document.

[2] V. Srinivasan, G. Varghese, S. Suri, and M. Waldvogel, " Fast and scalable layer four switching," *in Proc. ACM SIGCOMM' 98*, Aug. 1998, pp. 191-202

[3] V. Srinivasan, S. Suri, and G. Varghese, " Packet classification using tuple space search," in *Proc. ACM SIGCOMM' 99*, Aug. 1999, pp.135-146

[4] Hyesook Lim and Yeojin Jung, "A Parallel Multiple Hashing Structure for IP Address Lookup,"

[5] Florin Baboescu, Sumeet Singh, and George Varghese, " Packet Classification for Core Router:Is there an alternative CAMs?," *IEEE Infocom 2003*, vol.1, pp. 53 - 63, Apr., 2003