패킷 필터링 시스템에서 범위 규칙의 효율적 TCAM 엔트리 변환 알고리즘 연구

A Study on the Efficient Algorithm for Converting Range Matching Rules into TCAM Entries in the Packet Filtering System

김용권*, 기장근*★, 조현묵*, 최진규**, 이규호***
YongKwon Kim*, JangGeun Ki*, HyunMook Cho*, JinKyu Choe**, KyouHo Lee***

요 약

패킷 분류란 규정된 규칙과 입력된 패킷의 헤더 필드를 검색하여 매칭 여부를 판단하는 것으로 하드웨어적인 패킷 필터링 시스템은 일반적으로 Ternary Content Addressable Memory를 사용하여 구현된다. 하지만 TCAM은 구조적인 특성으로 인해 범위 규칙을 효율적으로 분류할 수 없기 때문에 기존의 필터링 시스템에서는 주어진 범위를 대표할 수 있는 prefix 형태의 값으로 범위를 변환하고 변환된 값을 TCAM 엔트리에 저장하여 패킷 필터링을 수행하며, 이 경우 범위 규칙의 필드가 W비트일 때 최대 2W-2개의 엔트리가 필요하다. 범위 규칙이 일반적으로 패킷 헤더 필드 중 소스포트와 목적지포드 필드에 사용되는 것을 고려하면 하나의 규칙이 최대 900개의 엔트리를 점유하게 된다. 본 논문에서는 범위 규칙을 TCAM 엔트리로 변환시 점유 엔트리 수를 줄이기 위해 범위 규칙을 대칭성을 가지는 그레이 코드로 변환한 후 범위를 대표할 수 있는 TCAM 엔트리로 변환하는 알고리즘을 제시하였다. 제안된 알고리즘은 최대 2W-4개의 TCAM 엔트리로 변환되며, 모든 범위에 대해 기존의 방법 보다 항상 더 적은 수의 TCAM 엔트리를 생성한다. 또한 negation 범위에 대해서도 효율적으로 적용 할 수 있다. 시뮬레이션 결과 16비트의 범위 매칭에 대해 기존의 방법보다 제안된 알고리즘이 평균 7%의 TCAM 엔트리를 감소시킬 수 있으며, 패킷의 소스와 목적지 포트를 동시에 고려하는 경우 평균 14%를 절감할 수 있고, 실제 사용되고 있는 침입탐지 프로그램의 범위 규칙에 적용시킨 결과 10% 정도의 TCAM 엔트리를 절약할 수 있음을 보였다.

Abstract

Packet classification is defined as the action to match the packet with a set of predefined rules. One of classification is to use Ternary Content Addressable Memory hardware search engine that has faster than other algorithmic methods. However, TCAM has some limitations. One of them is that TCAM can not perform range matching efficiently. A range has to be expanded into prefixes to fit the boundary. In general, the number of expansion could be up to 2w-2, where w is the width of the field. For example, if two range fields with 16 bits are used, there could be up to 30 x 30 = 900 expansions for a single rule. In this paper, we describe the novel algorithm for converting range matching rules into TCAM entry efficiently. The number of maximum entry is 2w-4 when using the algorithm. Furthermore, it has also benefit about the negation range. In the result of experimentation, the new scheme practically reduces 14 percent in case that searched fields are source port and destination port number.

Keywords : TCAM, Packet filter, Packet classification, Range matching

Ⅰ. 서론

최근 인터넷 사용자의 급속한 증가와 함께 온라인 게임, 인터넷 뱅킹, VOD, VoIP, P2P 등의 새롭고 다양한 멀티미디어 서비스의 대중화에 따라 인터넷 트래픽 량과 속도가 급속히 증가하고 있다. 랜 카드의 경우 현재 1Gbps 랜카드가 시장에 나와 있으며, 백본 망의 속도 또한 1Gbps에서 10Gbps, 40Gbps로 계속적인 증가를 보이고 있다. 이러한 트래픽 속도의 증가는 스위칭 허브, 라우터, 방화벽 시스템, IDS(Intrusion Detection System) 등과 같은 네트워크 장비의 처리 속도 향상이 뒷받침되어야 한다. 그러므로 네트워크 장비의 핵심 요소 중 하나인 패킷 분류를 위한 필터링 시스템의 속도 향상이 필수적이다. 패킷 분류 처리 성능은 TCP/IP 패킷의 최소 길이가 40byte라고 가정할 때, 2.5Gbps에서 초당 7.81Mpps(packet per second)를 분류할 수 있어야하며, 10Gbps에서는 31.25Mpps, 40Gbps일 때는 125Mpps를 분류할 수 있어야 한다.

패킷 필터링 시스템에서 사용하는 패킷 분류방법은 크게 알고리즘적인 방법과 하드웨어적인 방법으로 분류할 수 있다.[1,2] 알고리즘적인 방법은 패킷 분류 시스템의 용도에 적합하도록 효율적인 데이터 구조를 만들어 하드웨어적으로 구현하기 보다는 소프트웨어적으로 구현되며, 하드웨어적인 방법은 일반적으로 무정의 조건을 포함할 수 있는 Ternary Content Addressable Memory(TCAM)와 규칙들이 저장되어 있는 RAM을 이용하여 구현된다. 현재 패킷 분류에 대한 연구는 트래픽 속도의 증가와 주소 필드가 128bits인 IPv6의 사용으로 인해 알고리즘적인 방법보다는 TCAM을 이용한 하드웨어적인 구현에 초점이 맞추어져 있다.

본 논문에서는 TCAM을 이용하여 패킷 필터링 시스템을 구현하는 경우 문제점으로 제기되는 범위 매칭에 대해 기존의 방법보다 고속이면서 효율적으로 범위 규칙을 TCAM 엔트리로 변환하는 알고리즘을 제안하고 있다. 논문의 구성은 2장에서 패킷 필터에 대한 기존 연구와 TCAM에 관해 설명하고 3장에서는 본 논문에서 제시한 알고리즘에 대해 기술하였으며, 4장에서 제안된 알고리즘의 타당성을 위한 실험 결과를 기술하고 5장에서 결론을 맺고 있다.

* 公州大學校 情報通信工學部
(Division of Info. & Comm. Eng., Kongju Nat. Univ.)
** 韓南大學校 電子情報通信工學部 (School of Info. Tech. & Multimedia Eng., Hannam Univ.)
*** 仁濟大學校 電子情報通信工學部
(School of Electronics and Telecomm. Eng., Inje Univ.)

★ 교신 저자 (Correspondence author)
接受日:2005年 3月 16日, 修正完了日:2005年 7月 5日

Ⅱ. 패킷 분류 알고리즘과 TCAM

패킷 분류는 룰셑을 바탕으로 패킷이 입력되면 주소 필드와 같은 특정 필드를 읽어 규정된 룰셑과 비교하여 입력된 패킷이 규칙에 부합되는 지를 검사하는 것이다. 이때 주로 사용되는 패킷의 필드는 소스 주소, 목적지 주소, 소스 포트 번호, 목적지 포트 번호, 프로토콜 등 5개의 필드를 검사하게 된다. 그러므로 IPv4의 경우 104bit, IPv6의 경우 296bit를 검사하게 된다. 방화벽의 경우 5개 필드를 모두 검사하기도 하지만 라우터의 경우는 일반적으로 목적지 주소 필드만을 검색하기도 한다. 즉 패킷 필터 시스템의 용도에 따라 검색하는 필드의 종류와 비트 수가 달라지며, 규칙의 특성 또한 다르다. 입력된 패킷 필드와 규정된 규칙의 매칭 방법은 아래와 같이 3가지가 있다.

- exact 매칭 : 필드의 모든 비트를 정확히 검색하는 방법
- prefix 매칭 : 규칙이 무정의 조건을 포함하는 prefix 형태로 표현되어 있는 경우 검색하는 방법
- range 매칭 : 포트 번호와 같이 규칙이 십진수의 범위로 표현되는 경우 검색하는 방법

주소 검색을 위해서는 exact 매칭과 prefix 매칭 방법이 사용되며, 포트번호 검색에는 exact 매칭과 범위 매칭 방법이 사용된다.

패킷 분류 방법에는 Set-pruning Tries[3], Grid-of-Tries[4], EGT-PC[5], tuple space[6], bitmap-intersect[7], ABV[8], HiCut[9], HyperCut[10], P2C[11] 등과 같은 알고리즘적인 방법과 TCAM[12-14]을 이용한 하드웨어적인 방법이 있다. 표 1은 대표적인 패킷 분류 방법들을 검색 시간과 메모리 요구량으로 비교한 것이다[11,15]. 표에서 N은 규칙 수를 의미하며, d는 검색 필드 수, W는 검색 필드

본 논문은 공주대 자체학술연구비, 2005 두뇌한국21사업에 의하여 일부 지원되었음.

의 비트 수, W′는 필드 중 가장 긴 비트 수를 의미한다.

표에서 알 수 있듯이 알고리즘적인 방법은 검색시간이 검색 필드의 비트 수에 비례해 증가되므로 IPv6와 같이 주소 비트 수가 증가하게 되면 검색시간이 비례적으로 증가하게 된다. 그러므로 알고리즘적인 방법에서의 가장 큰 이슈는 검색시간을 줄이는 것이다. 이를 위해 Eatherton 등은 검색 시간을 줄이는 방법으로서 한번에 하나의 비트를 검색하는 것이 아니라 일정한 크기의 비트를 한번에 검색함으로서 시간을 단축할 수 있는 Tree Bitmap 방법을 제안[16]하고 있다.

표 1. 패킷분류 방법의 성능비교
Table 1. Performance comparison

패킷분류 방법	검색시간	메모리
set-pruning	dW	NddW
grid-of-tries*	dW	NdW
tuple space	Wd-1	NdW
bitmap-intersect	$W' + \frac{N}{memorywidth}$	dN2
HiCut	dW	Nd
P2C	W′/s+1	$dNklog(\frac{N}{k}+1)$
TCAM	1	NdW

* 2개의 필드만을 고려함

TCAM과 알고리즘적인 방법의 가장 큰 차이는 표에서 알 수 있듯이 검색시간에 있다. TCAM의 경우 검색시간이 검색 필드의 길이에 상관없이 “1”이다. 즉, 한 클록만에 검색을 수행할 수 있다는 것을 의미한다. 이러한 TCAM은 내부적으로 값과 마스크(value, bit mask)로 이루어진 엔트리들을 가지고 있으며, 규칙내의 비트 값이 ‘0’, ‘1’, ‘x’ 값 중 하나를 가질 수 있다. x는 무정의 조건(don′t care)을 의미하는 것으로 TCAM을 이용하여 prefix와 같이 무정의 조건을 포함하는 규칙을 검색할 수 있다.

TCAM은 검색시간이나 메모리 측면에서 다른 알고리즘적인 방법에 비해 우수하지만, 몇 가지의 단점을 가지고 있다. 기술적인 문제점으로서 표 2에 타나낸 것과 같이 일반 SRAM과 비교할 때 고가이며 소비전력이 많다. 하지만 TCAM은 처리속도 측면에서 타 알고리즘적인 방법보다 우수하기 때문에 수요가 증가하고 있으며, 이로 인해 제조 기술이 계속적으로 발전하고 있어 TCAM가격이나 소비 전력의 문제가 점차 줄어들고 있는 추세이다. 현재 18Mbit의 TCAM[17]이 판매되고 있으며 가격은 약 $250이다.

표 2. TCAM과 SRAM의 비교
Table 2. Comparison of TCAM and SRAM

종류	클럭	가격	소비전력	면적(mm)
SRAM(9Mbit)	250MHz	$20	0.75W	14x22
TCAM(9Mbit)	100MHz	$200	8.5W	40x40

TCAM을 사용하데 있어 문제점은 가격이나 소비전력 등의 기술적인 측면 외에 범위 매칭 규칙에 대한 문제가 있다. 표 1에서 TCAM의 메모리 요구량은 범위 매칭 규칙을 제외한 exact 매칭과 prefix 매칭 규칙인 경우만을 나타낸 것으로, 규칙 내에 범위로 규정된 규칙이 있는 경우 추가적인 TCAM 엔트리가 필요하다.

III. 범위 규칙을 TCAM 엔트리로 변환하기 위한 기존 방법

범위 매칭 규칙을 TCAM 엔트리로 변환하는 기존의 방법은 범위를 prefix 형태로 변환하여 사용하는 것으로, 예를 들어 필드가 4비트이고 범위 [3..12]인 규칙을 prefix로 나타내면 0011, 01**, 10**, 1100 등 4개의 항으로 나타낼 수 있다. prefix로 변환된 4개의 항을 TCAM 엔트리에 저장한 후 패킷을 수신하게 되면 검색에 필요한 패킷 헤더 필드를 TCAM에 입력하여 범위 매칭을 수행하게 된다. 범위를 prefix 형태의 TCAM 엔트리로 나타내면 필드의 길이가 W일 때 최대 2W-2개의 엔트리가 필요하게 된다. 예를 들어, 필드의 비트가 16 비트인 경우 규칙 하나가 최대 30개의 TCAM 엔트리를 점유하게 되며, 소스 포트와 목적지 포트 번호를 동시에 고려하는 경우에는 최대 900개의 엔트리가 필요하게 된다. 그러므로 TCAM을 이용하여 패킷 필터링 시스템을 구현하는 경우 범위 규칙을 TCAM 엔트리로 변환할 수 있는 효율적인 방안이 필요하다.

Ⅳ. 그레이 코드를 이용한 제안 알고리즘

TCAM을 이용한 패킷 필터 시스템 구현시 주어진 규칙이 범위로 규정되어 있는 경우 주어진 범위를 포함하는 여러 개의 prefix들로 변환하여 TCAM에 저장한다. 그러나 범위 규칙을 TCAM 엔트리로 변환할 때 TCAM의 특성상 무정의 조건이 임의의 비트 위치에 있어도 매칭 기능을 수행할 수 있다. 본 장에서는 이와 같은 TCAM의 동작 특성을 이용하고 또한 십진수 범위를 단순 이진코드가 아닌 그레이 코드로 표현함으로써 주어진 범위 규칙을 더 적은 수의 TCAM 엔트리로 변환하는 알고리즘을 제시하였다. 제안된 알고리즘은 필드의 길이가 W 비트일 때 최대 2W-4개의 TCAM 엔트리로 변환되며, prefix 표현으로 변환할 때보다 항상 더 적은 수의 TCAM 엔트리를 생성한다. 또한 negation 범위에 대해서도 효율적으로 적용시킬 수 있다.

4.1 그레이 코드

그레이 코드는 그림 1에 나타낸 것과 같이 인접한 값들이 단지 1비트만 다르며 대칭성을 가지고 있는 코드이다. 이진 코드를 대응되는 그레이 코드로 변환하는 방법은 그림 1과 같이 간단히 XOR 게이트를 사용하여 구현할 수 있기 때문에 하드웨어적으로 구현이 용이하다. 이진 코드와 그레이 코드를 이용한 TCAM 엔트리 변환 예를 그림 2에 나타내었다. 그림은 범위 [3..12]를 이진코드와 그레이 코드를 이용해 TCAM 엔트리로 표현한 것으로서, 이진 코드의 경우 주어진 범위가 4개의 prefix로 표현되지만, 그레이 코드를 이용하는 경우 대칭적인 특징으로 인해 2개의 TCAM 엔트리로 범위를 나타낼 수 있다.

4.2 제안 알고리즘

그레이 코드를 이용한 TCAM 엔트리 변환 알고리즘의 기본 원리는 주어진 범위가 2의 지수승개의 정수로 구성되고 범위의 시작 값이 범위에 속한 정수 개수의 1/2값의 공배수이면 주어진 범위는 하나의 항으로 표현될 수 있다는 것이다. 예를 들어 범위 [4..11]은 총 8개(=23)의 정수로 구성되고 8의 1/2 즉 4의 배수로 범위가 시작됨으로 범위 [4..11]은 하나의 항으로 표현 가능하다. 또한 그레이 코드는 표현되는 전체 수의 범위에서 중앙을 중심으로 대칭관계가 있음을 알 수 있다.

이와 같은 특성을 이용하여 본 논문에서 개발된 TCAM 엔트리 변환 알고리즘은 다음과 같다. 먼저 주어진 범위 [a..b]를 3개 영역 r1, r2, r3로 그림 3의 알고리즘에 따라 분할한다. 이와 같은 영역분할 알고리즘에 따라 분할된 영역 r1과 r2는 각각 m1과 m2를 중심으로 위아래 대칭 성질을 가지며, 따라서 쉽게 TCAM 엔트리로 변환할 수 있다. r3 영역을 TCAM 엔트리로 변환할 때는 비록 r3범위가 [a′+1..b′-1]로 주어지지만 최대 [m1..m2]범위까지 한 덩어리로 생각하면 쉽게 하나의 TCAM 엔트리로 변환할 수 있다.

십진수	이진 코드	그레이 코드
0	000	000
1	001	001
2	010	011
3	011	010
4	100	110
5	101	111
6	110	101
7	111	100

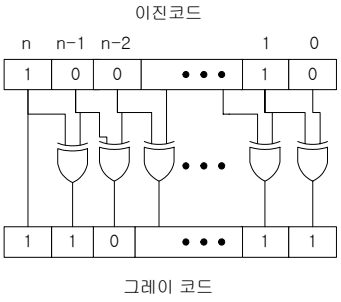
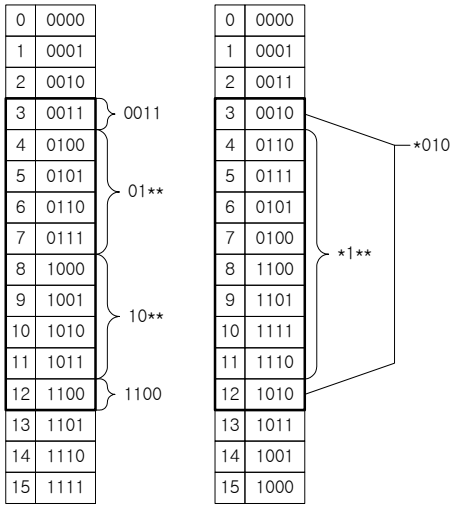


그림 1. 그레이 코드 변환
Fig. 1. Conversion into gray code



(a) 이진코드 경우 (b) 그레이코드 경우

그림 2. 범위 [3..12]의 TCAM 엔트리 표현

Fig. 2. TCAM entry value for range of [3..12]

```
// 범위 [a..b]를 3개 영역으로 분할하는 알고리즘
m1 = get_midval(a, b, bits);
if (m1 == (a+b+1)/2)
{   set range r1 = [a..b];
    return;
} else if (m1 < (a+b+1)/2)
{   a′ = m1 + (m1-a-1);
    set range r1 = [a..a′]
    m2 = get_midval(a′+1, b, bits);
    b′ = m2 - (b-m2+1);
    set range r2 = [b′..b];
    if (a′+1 < b′) set range r3 = [a′+1..b′-1];
    return;
} else
{   b′ = m1 - (b-m1+1);
    set range r1 = [b′..b]
    m2 = get_midval(a, b′-1, bits);
    a′ = m2 + (m2-a-1);
    set range r2 = [a..a′];
    if (a′ +1< b′) set range r3 = [a′+1..b′-1];
    return;
}
```

(a) 범위를 영역 r1,r2,r3로 분할하는 알고리즘

```
int get_midval(int from, int to, int bits)
{
    if (from == to)
        return from;
    else
    {   int pval = pow(2, bits-1);
        int mval = pval;
        for(int i=bits; i>0; i--)
        {   if (mval > to)
            {   // range [from..to] exists at left
                pval /= 2; mval -= pval;
            } else if (mval <= from)
            {   // range [from..to] exists at right
                pval /= 2; mval += pval;
            } else
                return mval;
        }
    }
}
```

(b) get_midval() 함수

그림 3. 범위 [a..b]의 영역분할 알고리즘

Fig. 3. Range division algorithm

영역 분할 알고리즘을 5비트로 표현된 범위 [a..b]=[5..20]을 예로 계산해보면 get_midval(5,20,5) 함수의 결과로 m1=16이 되고, m1=16>(a+b+1)/2= 13 임으로 b′=m1-(b-m1+1)=16-(20-16+1)=11이 된다. 따라서 영역 r1 = [b′..b] = [11..20] 으로 설정된다. 다음에 m2=get_midval(5,10,5)=8이 되고 a′=m2+(m2-a-1)=8+(8-5-1)=10이 되어 영역 r2= [a..a′]=[5..10]으로 설정된다. 마지막으로 a′+1=11이고 b′=11임으로 r3 영역은 존재하지 않게 된다. 범위 [5..20]에 대한 영역분할과 이를 이용한 TCAM 엔트리 변환 값을 그림 4에 나타내었다.

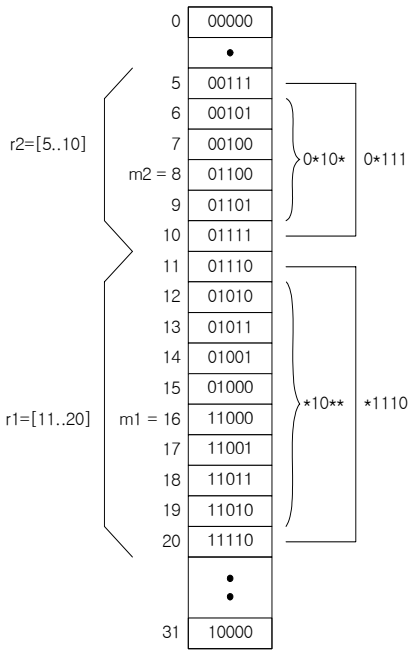


그림 4. 영역분할 알고리즘 적용 예
Fig. 4. Example of the range division

제안된 알고리즘에서 중심값 m을 중심으로 위아래 대칭 성질을 갖는 영역 r1 또는 r2를 TCAM 엔트리로 변환하는 방법은 다음과 같다. 먼저 영역 r1=[b′..b]이고 중심값이 m1이라고 하면, 아래 식에 따라 영역의 길이 d=b-b′+1보다 작은 최대 2의 지수승 값 2^{n_i}를 차례로 구하면 이 값들이 중심값을 중심으로 묶일 수 있는 개수가 된다. 예를 들어 r1=[11..20]이고, m1=16인 경우 n_0=3이 되고 따라서 2^{n_0}=8개가 중심값 16을 중심으로, 즉 범위 [12..19]의 값들이 하나의 항으로 묶일 수 있다. 다음에 n_1을 구하면 2가 되고 이는 11과 20이 서로 묶여 하나의 항으로 표현될 수 있음을 의미한다. n_i값을 구하는 프로그램은 그림 5에 나타낸 것과 같이 b-b′+1 값을 정수변수에 넣은 후 각 비트 값을 검사하면 쉽게 구현할 수 있다.

$$n_0 = floor(log_2(b-b'+1))$$

$$n_i = floor(log_2(b-b'+1-\sum_{j=0}^{i-1}2^{n_j}))\ \ (i=1,2,...)$$

```
int get_n(int from, int to, int bits, int *n)
{  int d = to - from + 1;
   unsigned int mask = pow(2,bits-1);
   int i = 0;
   for(int k=bits-1; k>=0; k--, mask>>=1)
        if (d & mask)
             n[i++] = mask;
   return i;
}
```

그림 5. 대칭적인 범위내의 항수 계산 알고리즘
Fig. 5. Algorithm to calculate terms in the symmetrical range

그레이 코드는 특성상 주어진 범위가 2의 지수승 개의 정수로 구성되고, 범위의 시작 값이 범위 길이를 2로 나눈 값의 공배수이면 주어진 범위는 하나의 항으로 표현되며, 이 경우 TCAM 엔트리 표현식을 구하는 알고리즘은 그림 6과 같다. 예를 들어 범위 [12..19]는 범위 길이가 19-12+1=8이고 범위 시작값 12가 범위길이의 반, 즉 8/2=4의 공배수임으로 하나의 항으로 표현할 수 있다. 항에 대한 표현식을 얻기 위해 먼저 범위의 시작값에 대한 그레이 코드의 각 비트를 g_i (i=m-1, m-2,..., 0)라고 하고, 2^n개의 항이 묶인다고 하면, 먼저 다음 식과 같은 XOR 연산을 이용해 비트열 h_i를 계산한다.

$$h_{m-1} = g_{m-1}$$
$$h_i = h_{i+1} \oplus g_i\ (i = m-2, \ldots, 0)$$

구해진 비트열 h_i를 왼쪽 방향(i=0부터 i=m-1 방향)으로 차례로 검사하면서 비트값이 0일 경우 g_i 비트를 '*'로 바꾸어주는 작업을 n번 수행해 주면 항에 대한 표현식을 얻을 수 있다. 그림 6에 6비트로 표현되는 범위 [12..19]를 하나의 항으로 표현하는 과정을 예로 나타내었다.

제안된 알고리즘은 negation 범위에 대해서도 효율적으로 적용시킬 수 있다. prefix 표현의 경우 negation 형태로 범위가 주어지면 2개의 범위로 영역을 나누어 각 영역에 대해 prefix로 표현하기 때문에 항수가 늘어나는 경향이 있으나, 제안 알고리즘은 나누어진 2개의 범위가 연속성이 있는 대칭형태를 가짐으로 하나의 범위처럼 취급할 수 있다. negation 예로서 매칭 필드가 8bit일 때 ![100..139]의 negation 매칭

을 수행하는 경우, prefix는 범위를 [0..99]와 [140..255]로 나누어 변환 시키지만 그레이 코드는 그림 7에서 알 수 있듯이 대칭적인 특징을 가지기 때문에 범위를 나누어 고려하지 않아도 된다. 표 3은 prefix와 그레이 코드에 의한 TCAM 엔트리 값을 나타낸 것이다. prefix는 7개의 TCAM 엔트리가 필요하지만 제안 알고리즘은 5개의 엔트리만 있으면 negation 범위를 나타낼 수 있다.

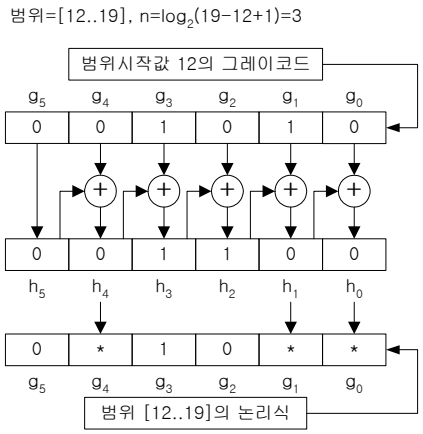


그림 6. 범위 [12..19]의 최종 엔트리 값
Fig. 6. TCAM entry value of range [12..19]

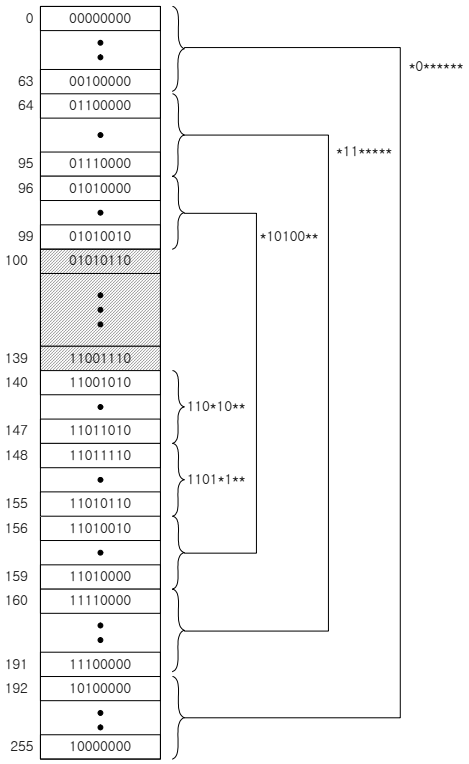


그림 7. 제안 알고리즘을 이용한 negation 변환
Fig. 7. Example of converting a negation range rule

표 3. negation 범위 ![100..139]의 변환
Table 3. Conversion of negation rage ![100..139]

negation 범위	Prefix	제안알고리즘
![100..139]	00****** 010***** 011000** 100011** 1001**** 101***** 11******	*0****** *11***** *10100** 110*10** 1101*1**

Ⅴ. 시뮬레이션 결과

제안된 알고리즘의 성능을 평가하기 위해 1비트부터 16비트까지 가능한 모든 범위에 대해 시뮬레이션을 수

행했으며, 침입탐지 프로그램인 SNORT[18]의 규칙 중에서 범위로 규정된 규칙에 대해 기존의 방법과 제안된 알고리즘을 적용시켜 성능을 분석하였다.

5.1 제안된 알고리즘의 성능 평가

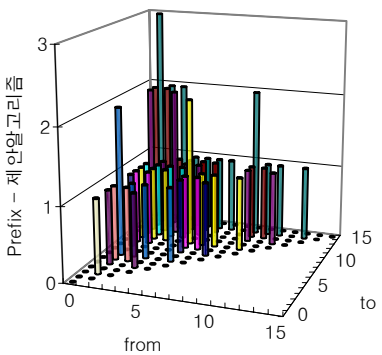
그림 8(a)에는 범위 필드가 4비트이면서 이진코드를 사용해 prefix로 표현하는 경우 필요한 prefix 개수를 나타내었으며, 그림 8(b)에는 제안된 알고리즘에 의한 TCAM 엔트리 수를 나타내었다. 예를 들어 범위 [3..12]의 경우 이진코드를 사용하고 범위를 prefix로 나타내면 0011, 01**, 10**, 1100의 4개 prefix가 필요하며, 따라서 그림 8(a)의 3번 컬럼 12번 칸에 4가 기록되어 있다. 같은 범위 [3..12]를 제안 알고리즘을 이용하여 나타내면 *010, *1**의 2개의 TCAM 엔트리로 표현되고, 이 개수를 그림 8(b)에 기록하였다. 그림 8(c)는 prefix 표현의 경우와 제안된 알고리즘에 의한 TCAM 엔트리 수의 차이를 그래프로 나타낸 것이다. 이 그림으로부터 이진코드의 prefix 표현보다 제안된 알고리즘에 의한 TCAM 엔트리 수가 항상 같거나 더 적음을 확인 할 수 있다. 따라서 본 연구에서 개발된 알고리즘이 매우 효율적으로 최소 개수의 TCAM 엔트리 수를 계산해냄을 알 수 있다.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1															
1	1	1														
2	2	2	1													
3	1	2	1	1												
4	2	3	2	2	1											
5	2	3	2	2	1	1										
6	3	4	3	3	2	2	1									
7	1	3	2	2	1	2	1	1								
8	2	4	3	3	2	3	2	2	1							
9	2	4	3	3	2	3	2	2	1	1						
10	3	5	4	4	3	4	3	3	2	2	1					
11	2	4	3	3	2	3	2	2	1	2	1	1				
12	3	5	4	4	3	4	3	3	2	3	2	2	1			
13	3	5	4	4	3	4	3	3	2	3	2	2	1	1		
14	4	6	5	5	4	5	4	4	3	4	3	3	2	2	1	
15	1	4	3	3	2	3	2	2	1	3	2	2	1	2	1	1

(a) 이진코드 prefix 표현

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1															
1	1	1														
2	2	1	1													
3	1	2	1	1												
4	2	2	2	1	1											
5	2	2	1	2	1	1										
6	3	2	2	2	2	1	1									
7	1	3	2	2	1	2	1	1								
8	2	3	3	3	2	2	2	1	1							
9	2	3	2	3	2	2	1	2	1	1						
10	3	4	3	3	3	2	2	2	2	1	1					
11	2	3	2	2	1	3	2	2	1	2	1	1				
12	3	3	3	2	2	3	3	3	2	2	2	1	1			
13	3	3	2	3	2	3	2	3	2	2	1	2	1	1		
14	4	3	3	3	3	4	3	3	3	2	2	2	2	1	1	
15	1	4	3	3	2	3	2	2	1	3	2	2	1	2	1	1

(b) 제안 알고리즘의 TCAM 엔트리 표현



(c) Prefix - 제안 알고리즘 차이 그래프

그림 8. 4비트일 경우 TCAM 엔트리수의 비교
Fig. 8. Comparison of the number of TCAM entries for 4bits

그림 9는 8비트를 가정하여 0-255 사이에서 가능한 모든 범위를 적용한 것으로 prefix 표현에 의한 TCAM 엔트리 수와 제안된 알고리즘에 의한 TCAM 엔트리 수의 차이를 나타낸 것이다. 엔트리 수의 최대 차이는 7개이며, 제안된 알고리즘을 이용하는 경우 평균 엔트리 수는 5.14개이고 prefix로 표현하는 경우에는 6.04개로서 제안된 알고리즘이 평균 약 1개 정도 엔트리 수가 적다.

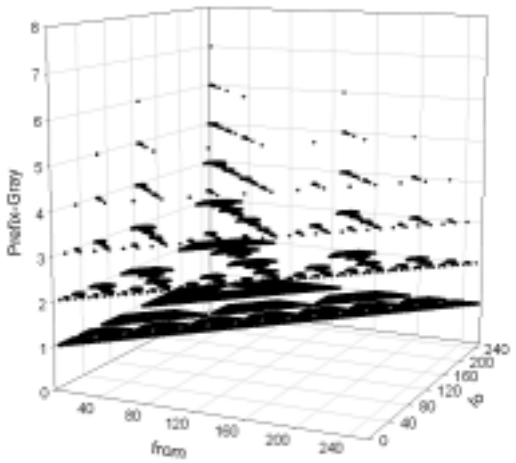


그림 9. prefix 표현과 제안된 알고리즘에 의한 TCAM 엔트리 수 차이(8비트)
Fig. 9. Difference of the number of the TCAM entries for 8bits

그림 10은 제안된 알고리즘을 이용해 비트별로 평균 TCAM 엔트리 수를 계산하여 나타낸 것이다. 그림에서 알 수 있듯이 이진코드를 사용해서 prefix로 표현하는 것에 비해 제안된 알고리즘을 이용할 경우 평균 TCAM 엔트리 수가 더 적음을 알 수 있다. 필드가 16비트인 경우 최대 엔트리 개수의 차이는 15개이고, 평균 TCAM 엔트리 수는 약 7% 정도 감소되며, 패킷 필터링 시스템에서 범위 매칭이 요구되는 필드는 각각 16비트 길이인 소스 포트 번호와 목적지 포트 번호임을 감안하면 평균 14% 정도 TCAM 엔트리 수를 줄일 수 있다.

그림 11은 negation 범위에 대해 prefix에 의한 TCAM 엔트리 수와 제안 알고리즘에 의한 TCAM 엔트리수를 비교한 것이다. 그림에서 알 수 있듯이 negation 범위에 대해서도 제안 알고리즘이 우수함을 알 수 있다.

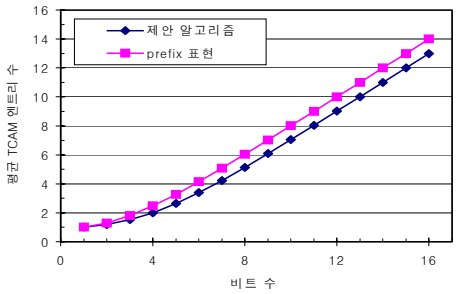


그림 10. 기존 방법과 제안 알고리즘에 의한 TCAM 엔트리 수 차이
Fig. 10. Difference of the number of the average TCAM entries

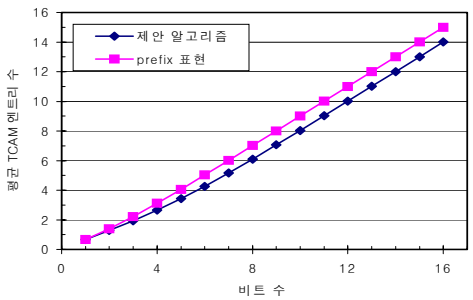


그림 11. Negation 범위에 대한 TCAM 엔트리 수 차이
Fig. 11. Difference of the number of the average TCAM entries for the negation range

5.2 침입탐지 프로그램의 규칙에 적용한 결과

제안된 알고리즘을 실제 사용되고 있는 SNORT[18] 프로그램의 규칙들에 적용한 결과를 표 4에 나타내었으며, 실험에 사용된 규칙은 SNORT 규칙 중 범위를 규정하고 있는 규칙만을 대상으로 하였다. SNORT 2.3의 경우 규칙들 중 범위 매칭이 규정된 규칙은 62개로서 두 가지 방법에 의해 변환된 TCAM 엔트리 수는 표에서 알 수 있듯이 prefix를 사용하는 경우 총 351개의 엔트리가 필요하며, 제안된 알고리즘을 이용하는 경우 318개의 엔트리가 필요하다. 결과적으로 제안된 알고리즘을 이용하여 범위를 TCAM 엔트리로 변환하면 prefix로 표현하는 경우 보다 약 10%(33개)의 TCAM 엔트리를 절약할 수 있다. SNORT 2.2는

SNORT 2.3과 동일한 범위 규칙을 가지며, SNORT 2.1은 범위 규칙 수가 59개이며 절감 비율은 약 10%로서 제안된 알고리즘이 기존의 prefix 표현방법보다 효율적으로 범위 규칙을 TCAM 엔트리로 변환 할 수 있음을 보여주고 있다.

표 4. SNORT 범위 규칙의 변환 결과 비교
Table 4. Results of converting SNORT's range rules

프로그램	포트	규칙수	prefix 표현	제안 알고리즘	절감비율 (%)
snort_2.3	소스	9	34	29	14.7
	목적지	53	317	289	8.8
	합계	62	351	318	9.4
snort_2.1	소스	8	28	23	17.8
	목적지	51	313	285	`8.9
	합계	59	341	308	9.8

Ⅵ. 결론

본 논문에서는 TCAM을 이용해 패킷 필터링 시스템을 구현하는 경우 범위 규칙을 TCAM 엔트리로 변환하는 효율적인 알고리즘을 제시하였다. 범위 규칙을 TCAM 엔트리로 변환하는 기존의 방법은 이진 코드를 이용하여 주어진 범위를 대표하는 prefix 형태의 값으로 변환하는 것으로 검색 필드가 W일 때 최대 2W-2개의 엔트리가 필요하다. 제안 알고리즘은 범위 규칙을 TCAM 엔트리로 변환시 엔트리 수를 감소시키기 위해 대칭성을 가지는 그레이 코드를 이용하여 주어진 범위를 TCAM 엔트리로 변환하는 것이다. 기존의 방법과 비교할 때 제안 알고리즘은 동일 범위에 대해 더 적은 수의 엔트리로 범위를 나타낼 수 있기 때문에 기존의 방법보다 TCAM 엔트리 개수를 줄일 수 있고, 이는 동일한 TCAM 용량에서 보다 많은 규칙을 검색할 수 있음을 의미한다. 하드웨어적인 측면에서 보면 기존의 방법은 수신한 패킷 데이터를 그대로 TCAM에 입력하지만 제안 방법은 TCAM에 입력하기 전에 수신 데이터를 그레이코드로 변환하는 과정이 필요하다. 하지만 그레이 코드는 검색하고자 하는 패킷 헤더 필드의 비트 수 만큼의 XOR 게이트만 있으면 쉽게 구현할 수 있기 때문에 전체 패킷 필터링 시스템에서 미치는 영향이 극히 미미하며 지연시간 또한 게이트 하나를 통과하는 시간만 추가적으로 소요되기 때문에 시스템의 성능에 영향을 주지 않는다. 결론적으로 제안 알고리즘은 기존의 방법과 비교해서 검색 속도를 그대로 유지하면서도 패킷 필터링 시스템이 검색할 수 있는 규칙 수를 증대시킨다. 시뮬레이션 결과 16비트의 소스 포트와 목적지포트를 동시에 고려하는 경우 평균 14%의 엔트리를 절감할 수 있으며, 또한 실제 사용되고 있는 침입탐지 프로그램의 범위 규칙에 적용시킨 결과 기존의 방법보다 약 10%의 엔트리를 절약할 수 있음을 확인하였다.

참고 문헌

[1] Pankaj Gupta and Nick McKeown, "Algorithms for Packet Classification," IEEE Network Special Issue, vol. 15, no. 2, pp 24-32, March/April 2001

[2] Shubhash Wasti, "Hardware Assisted Packet Filtering Firewall," http://bistrica.usask.ca/madmuc/Pubs/shw320.pdf, Saskatchewan university, 2001

[3] Paul Francis Tsuchiya, "A Search Algorithm for Table Entries with Non-contiguous Wildcarding," http://citeseer.ist.psu.edu/ tsuchiya91search.html, 1991

[4] V. Srinivasan, George Varghese, Subhash Suri, Marcel Waldvogel, "Fast and Scalable Layer Four Switching," Proceedings of ACM SIGCOMM'98, pp.203-214, Sept. 1998

[5] Florin Baboescu, Sumeet Singh, George Varghese, "Packet Classification for Core Routers: Is there an alternative to CAMs?," INFOCOM 2003

[6] V. Srinivasan, S. Suri and G. Varghese, "Packet Classification using Tuple Space Search," Proc. ACM Sigcomm, pp.135-146 ,Sept. 1999

[7] T. V. Lakshman and D. Stiliadis, "High-Speed Policy-based Packet Forwarding Using Efficient Multi- dimensional Range Matching," Proc. ACM Sigcomm, pp.191-202, Sept. 1998.

[8] Florin Baboescu, George Varghese, "Scalable Packet Classification," ACM Sigcomm, August 2001

[9] P. Gupta and N. McKeown, "Packet Classification Using Hierarchical Intelligent Cuttings," Proc. Hot Interconnects VII, Aug. 1999; also available in IEEE Micro, vol.20, no.1, pp.34-41, Jan./Feg. 2000

[10] Sumeet Singh, Florin Baboescu, George Varghese, and Jia Wang, "Packet Classification Using Multidimensional Cutting," Proc. ACM Sigcomm, Aug. 2003
[11] Jan van Lunteren, Ton Engbersen, "Fast and scalable packet classification," IEEE Journal on Selected Areas in Communications, vol.21, no.4, pp. 560-571, May 2003
[12] Ed Spitznagel, David E. Taylor, Jonathan S. Turner, "Packet Classification Using Extended TCAMs," ICNP 2003, pp. 120-131
[13] Nen-Fu Huang, Whai-En Chen, Jiau-Yu Luo, Jun-Min Chen, "Design of multi-field IPv6 packet classifiers using ternary CAMs," GLOBECOM2001,no.1, pp.1877-1881,Nov 2001
[14] Girija J. Narlikar, Anindya Basu, Francis Zane, "CoolCAMs: Power-Efficient TCAMs for Forwarding Engines," INFOCOM 2003
[15] David E. Taylor, "Survey & Taxonomy of Packet Classification Techniques," Washington University in Saint Louis WUCSE-2004-24, May 2004
[16] Will Eatherton, George Varghese, Zubin Dittia, "Tree bitmap: hardware/software IP lookups with incremental updates," ACM SIGCOMM Computer Communication Review, Vol.34, Issue2, pp.97-122, April 2004
[17] SiberCore, http://www.sibercore.com
[18] SNORT, http://www.snort.org

저 자 소 개

金 容 權 (學生會員)

1999년 : 공주대학교 전자공학과졸업 (공학사)
2001년 : 공주대학교 대학원 전기전자정보공학과 졸업(공학석사)
2001년~현재 공주대학교 대학원 전기전자정보공학과 박사과정

<주관심분야> 차세대 인터넷 기술, 인터넷 망 관리, 패킷 필터링 시스템

奇 長 根 (正會員)

1986년 : 고려대학교 전자공학과 졸업(공학사)
1988년 : 고려대학교 대학원 전자공학과 졸업(공학석사)
1992년 : 고려대학교 대학원 전자공학과 졸업(공학박사)
1992년~현재 공주대학교 정보통신공학부 교수
2002년~2003년 미국 Univ. of Arizona 방문교수

<주관심분야> 컴퓨터 네트워크, 차세대 네트워크, 멀티미디어 통신

趙 鉉 默 (正會員)

1989년 : 고려대학교 전자공학과 졸업(공학사)
1991년 : 고려대학교 대학원 전자공학과 졸업(공학석사)
1995년 : 고려대학교 대학원 전자공학과 졸업(공학박사)
1995년~ 현재 공주대학교 정보통신공학부 교수

<주관심분야> SoC 설계, 멀티미디어 시스템 설계 등

崔 眞 圭 (正會員)

1980년 : 고려대학교 전자공학과 졸업(공학사)
1982년 : 고려대학교 대학원 전자공학과 졸업(공학석사)
1987년 : 고려대학교 대학원 전자공학과 졸업(공학박사)
1987년 9월~1990년 8월 대전공업대학 조교수
1999년~2000년 미국 Oregon State Univ. 방문교수
1990년 8월~현재 한남대학교 교수

<주관심분야> 통신망 성능평가, 디지털시스템설계

李 圭 晧 (正會員)

1980년 : 경북대 전자공학과(공학사)
1982년 : 경북대 대학원 전자공학과(공학석사)
1998년 : The University of Gent 컴퓨터공학(Ph.D)
1986년~1988년 : 미국 AIT Inc, 연구원
1983년~2005년 : 한국전자통신연구원(ETRI) 책임연구원
2005년~현재 : 인제대학교 정보통신공학과 교수

<주관심분야> 인터넷고속화 기술, IP기반광대역유무선통합네트워크(BcN), IP기반 실시간응용기술, Network Processor 및응용기술, 고속 패킷처리, IP 스위치 및 라우터 시스템, 고성능네트워크시스템의내장형 (embedded) 제어기술