

IPv6을 위한 효율적인 Address Lookup

나상준^o 장기현 이병호

한양대학교 정보통신대학원

{victory^o, nuky99, bhrhee}@scann.hanyang.ac.kr

Efficient Address Lookup for IPv6

Sang Jun Na^o, Kee Hyun Jang, Byung Ho Rhee

The Graduate School of Information & Commendations, Hanyang Univ.

요 약

현재 인터넷에서는 사용자의 급격한 증가로 인해 고성능의 라우터를 요구하고 있고 주소부족으로 IPv4에서 IPv6로 변화를 하고 있다. IPv4처럼 IPv6에서도 Address Lookup이 병목이 될 것이며 IPv4와는 달리 IPv6는 128bit의 주소 길이를 가지고 있어 이에 맞는 라우터 구조와 Address Lookup 알고리즘이 필요하다. 본 논문에서는 IPv6주소 128bit 중 외부에서 할당 받는 64bit를 3단계로 나누는 계층적 네트워크 구성과 각 단계에 적합한 라우팅 테이블 구조와 Address Lookup 알고리즘에 대해 연구하였고 펜티엄 III 866MHz의 프로세서에서 알고리즘의 검색 시간을 측정해 각 단계에 맞는 라우팅 테이블 구조를 제안하였다.

1. 서 론

인터넷의 급성장으로 호스트 수와 이들 사이의 트래픽량은 엄청난 속도로 증가해왔다. 하지만 적합한 경로를 선택하는 Address Lookup 작업이 병목 역할을 하고 있다 [1,2]. IPv6[3,4]는 128bit의 주소길이를 사용해 Address Lookup 문제를 더 심각하고 복잡하게 만들 것이며 네트워크 형태도 많은 변화가 있을 것이다. 그리고 IPv4에 적합한 Address Lookup 알고리즘이 IPv6에서도 적합한지 여부는 확신할 수 없으며 이에 대한 연구가 절실하다. 그러므로 본 논문에서는 IPv6환경에서 적합한 라우팅 구조와 라우팅 테이블을 제안한다.

2. Address Lookup 알고리즘

IPv4에서는 라우팅 테이블 접근 횟수를 줄이거나 라우팅 테이블의 크기를 줄여 속도를 증가 시켰다. 접근 횟수를 줄이려는 방식은 크기가 커져 캐쉬에 모두 저장할 수 없기 때문에 저속의 메모리를 사용해 속도가 느려지는 단점이 있고, 라우팅 테이블의 크기를 줄이려는 방식은 알고리즘이 복잡해져 접근 횟수가 많아지며 속도가 느려지는 단점이 있다.

그러므로 좋은 라우팅 테이블 구조는 테이블의 크기를 줄이고 접근 횟수를 최소화하는 방법이라 할 수 있다. 하지만 라우팅 테이블의 크기와 접근 횟수의 관계는 역에 가까워 두 조건을 동시에 만족하기는 어렵다.

라우팅 테이블의 크기가 캐쉬보다 작을 때는 [그림1]과 같은 계산식에 의해 테이블 접근 횟수를 줄이는 것이 좋다. 캐쉬 속도 a는 소프트웨어 알고리즘 측면에서 보면 상수이며 접근 횟수 n이 감소하면 평균시간은 감소한다. 라우팅 테이블의 크기가 캐쉬보다 크면 [그림2]을 따르며 x, c도

상수이지만 알고리즘 선택에 영향을 주며 평균시간은 캐쉬와 메모리 접근 속도의 차 x, 접근 횟수 n, 캐쉬 크기 c 그리고 라우팅 테이블의 크기 t의 상관관계에 의해 결정된다.

라우팅 테이블의 크기 : t

캐쉬 속도 : a

접근 횟수 : n

$$\text{Address lookup 평균시간} = (t \times a \times n) / t = a \times n$$

[그림1]

라우팅 테이블의 크기 : t

캐쉬 속도 : a

메모리 접근 속도 : xa

접근 횟수 : n

캐쉬 크기 : c

$$\text{Address lookup 평균시간} = ((c \times a \times n) + (t-c) \times xa \times n) / t = a \times t \times x \times n + (1-x) \times n \times c / t$$

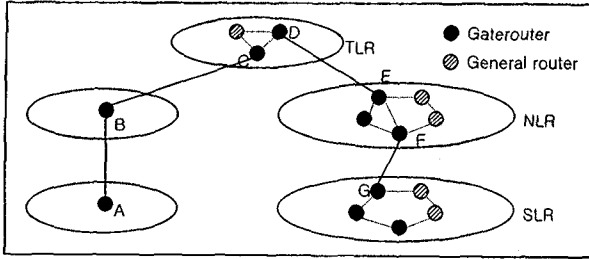
[그림2]

3. 계층적 라우팅 구조

IPv6에서 주소 수는 약 3.4×10^{38} 개가 가능해 현재 사용하는 평평한 네트워크 구성은 불가능하며 다른 형태가 필요하다. 본 논문에서는 IPv6에 적합한 네트워크 구성은 계

층적 라우팅 구조[5]가 가장 적합하다고 판단한다.

TLR, NLR, SLR로 나누어지며 TLR은 TLA, NLR은 NLA, SLR은 SLA를 대상으로 한다.



[그림3]

[그림3]과 같은 계층적 라우팅 구조는 128bit의 주소 중 상위 64bit를 대상으로 한 라우팅 구조이며 하위 64bit는 본 논문의 주제에 벗어나므로 제외한다.

동작하는 방법은 [그림3]에서 A라우터에 접속된 호스트가 G라우터에 접속된 호스트로 패킷을 보내려고 한다면 A->B->C->D->E->F->G의 순서로 라우터를 거친다. 다른 TLA와 NLA주소공간을 가질 경우는 최소 5번의 라우터를 지나며 NLA와 SLA를 더 세분화하면 5개 이상의 라우터를 지나게 된다.

계층적 라우팅 구조는 많은 라우터를 지나야 하므로 효율은 떨어지지만 IPv6주소에 대한 라우팅이 가능하고 네트워크 구성이 간단해서 사용하기가 적합하다.

4. 라우팅 테이블 구조

IPv6패킷에 대한 분석이 필요하지만 현재 구축된 네트워크는 작은 규모이거나 터널링으로 구축된 것이다. 여기서 패턴분석은 IPv6 패킷의 전체적인 특징을 대표한다고 볼 수 없어 현재 잘 구성된 IPv4에 IPv6의 특징을 부여해 IPv6의 패턴을 예상해 보고 앞서 기술한 특징들을 조합해 본다. IPv6에서는 CIDR[6]을 사용하지 않아 LPM이 아닌 Exactly Match방식이 사용가능하며 class가 존재하지 않아 상위 64bit에서 최소 3단계로 나눌 수 있다.

4.1 Offset Key

예로 설명하면 한 엔트리가 1byte이고 4bit의 주소체계를 가지는 라우팅 테이블이 있다면 테이블은 16(최대 주소 수)×1byte=16byte 공간이 필요하며 공간을 미리 확보하고 나면 엔트리는 목적지 주소로 찾을 수 있다.

라우팅 테이블의 크기가 캐쉬 크기보다 작을 때는 접근 횟수를 줄여야 하며 Offset Key는 간단하면서도 단 한번의 라우팅 테이블 접근으로 원하는 엔트리를 구할 수 있다.

계층적 라우팅 구조는 상위 64bit 주소를 세 개의 단계로 나누기 때문에 각 단계에서 최대 엔트리 수는 IPv4에 비해 작아 Offset Key방식을 사용할 수 있는 크기이다.

4.2 압축된 라우팅 테이블

라우팅 테이블을 압축하는 방식은 라우팅 테이블 크기를 줄여서 캐쉬 공간에 저장하는 방식이다. 테이블을 압축하

는 방식은 인접한 두 엔트리가 같은 Output Address를 가지고 있을 때 두 엔트리를 한 엔트리에 보관하는 방식으로 두 엔트리는 반드시 인접해 있어야 한다. 압축률을 알아보기 위해 Exactly Match를 사용하여 Merit Network의 IPv4 라우팅 테이블[7]을 압축해 본 결과 [표1]을 얻을 수 있었다.

[표1]

| Backbone router | IPv4 entry | Compressed entry | Compress rate |
|-----------------|------------|------------------|---------------|
| AADS | 27834 | 7959 | 0.286 |
| MAE | 33645 | 10161 | 0.302 |

IPv4에 IPv6의 특징을 부여해 압축하였기 때문에 순수 IPv6에서도 비슷한 양상을 가질 것으로 예상하며 IPv6의 압축률은 0.3으로 가정한다.

5. 시뮬레이션

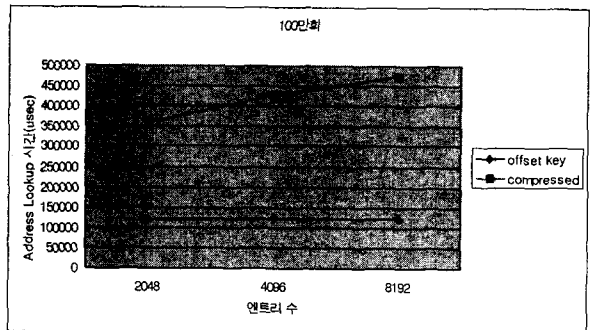
시뮬레이션은 펜티엄 III 866MHz, 256Mb시스템에서 시행하였다. Address Lookup 횟수 100, 200, 300만회와 3가지 엔트리 수를 조합한 9번의 경우에 대해 30번씩 반복 수행을 하였으나 지면 관계상 100만회만 기술하였다.

5.1 TLR

Offset Key는 메모리 순서 자체가 목적지 주소를 나타내므로 목적지 주소를 위한 공간이 필요 없어 한 엔트리에 필요한 메모리 크기는 matrix 1byte, Output Address 2byte가 된다. 엔트리 수에 상관없이 전체 공간이 필요하므로 라우팅 테이블의 크기는 8,192×3byte=24Kb가 필요하고 라우팅 테이블 접근 횟수의 경우 1회로 고정될 수 있다.

압축된 라우팅 테이블에서 한 엔트리에 필요한 메모리 크기는 목적지 주소 2byte, Output Address 2byte, matrix 3byte가 필요해 8,192×7byte×0.3=17Kb 크기가 필요하며 접근 횟수는 경우에 따라 달라진다.

압축된 라우팅 테이블의 크기는 Offset Key에 비해 작지만 둘 다 캐쉬 크기보다 작으므로 [그림1]에서 알 수 있듯이 접근 횟수가 작은 Offset Key가 더 유리하다.



[그림4]

[그림4]는 100만회 Lookup을 수행 했을 때 Address Lookup시간을 조회한 것으로 라우팅 테이블 조회 횟수가 작은 Offset Key의 Address Lookup시간이 더 작은 값을

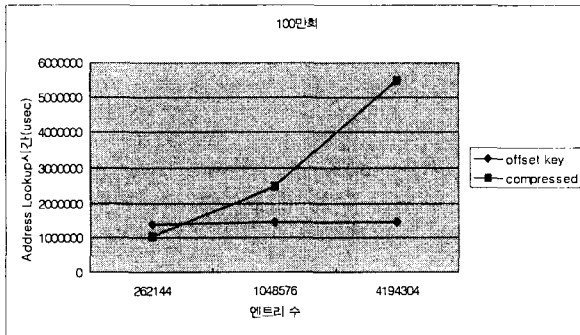
가진다. 압축된 방법의 경우 엔트리 수가 증가 할수록 시간이 증가하지만 Offset Key의 경우는 거의 변화가 없다. 이는 Offset Key는 엔트리 수에 상관없이 최대 엔트리 수만큼 메모리를 확보하기 때문이다.

5.2 NLR

Offset Key에 필요한 메모리 크기는 Output Address 3byte, matrix 1byte가 필요해 $16,777,216 \times 4 = 64\text{Mb}$ 크기가 필요하다.

압축 방법은 한 엔트리에 목적지 주소 3byte, matrix 4byte, Output Address 3byte가 필요해 10byte 크기를 가지며 0.3의 주소 압축률을 사용해 $16,777,216 \times 10 \times 0.3 = 48\text{Mb}$ 의 크기가 필요하다.

모두 일반적인 캐쉬보다 큰 라우팅 테이블을 가지며 [그림2]에는 따라 여러 요소의 상관관계에 의해 알고리즘이 결정되므로 NLR에 적합한 방법은 경우에 따라 달라진다.



[그림5]

[그림5]에서 엔트리 수가 작을 때는 압축된 테이블 방식이 우수하지만 엔트리 수가 늘어날수록 Offset Key방식이 우수해진다. 이는 Offset Key방식은 항상 최대 엔트리 수의 메모리를 요구하지만 압축된 라우팅 테이블은 엔트리 수에 따라 다른 메모리 크기를 요구하기 때문이다.

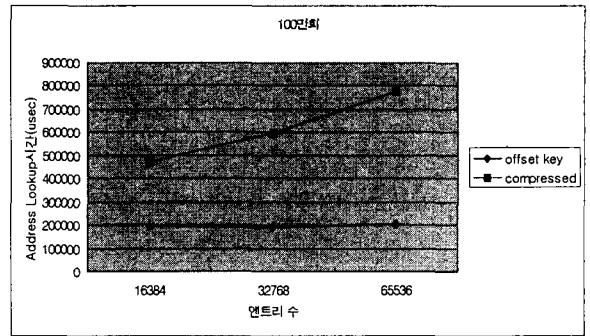
결국 적합한 알고리즘은 엔트리 수에 따라 달라지며 엔트리 수가 작을 때는 압축된 라우팅 테이블 방식을 사용하고 엔트리 수가 많을 경우에는 Offset Key를 사용하는 것이 더 효율적이다.

5.3 SLR

Offset Key는 한 엔트리에 matrix 1byte와 Output Address 2byte의 메모리가 필요하다. 엔트리 수에 상관없이 전체 공간을 위한 $65,536 \times 3\text{byte} = 192\text{Kb}$ 가 필요하고 라우팅 테이블 접근 횟수는 1회로 고정된다.

압축된 라우팅 테이블은 한 엔트리에 matrix 3byte, 목적지 주소 2byte, Output Address 2byte로 $65,536 \times 7\text{byte} \times 0.3 = 134\text{Kb}$ 가 필요하며 접근 횟수는 경우에 따라 다르다.

라우팅 테이블의 크기가 일반적인 캐쉬크기와 비슷해 라우팅 테이블 접근 횟수를 최소화하는 Offset Key방법이 적합한 방법이다. 만일 시스템이 제공하는 캐쉬의 크기가 192Kb보다 현저하게 작다면 NLR와 마찬가지로 경우에 따라 달라질 것이다.



[그림6]

본 시뮬레이션에서 제공해 주는 캐쉬 크기는 256Kb이므로 [그림6]의 결과는 TLR의 결과와 동일해 메모리 조희 횟수가 작은 Offset Key가 우수함을 알 수 있다

6. 결론과 향후 과제

TLR와 SLR의 경우는 Offset Key방식이 우수하였고 NLR은 엔트리 수가 작을 때 압축된 방식이 우수하고 엔트리 수가 많을 때는 Offset Key가 우수하였다. 특히 NLR의 경우에는 평균 Address Lookup시간이 다른 단계에 비해 현저히 느림을 알 수 있다.

본 논문이 좀 더 정확성을 지니기 위해서는 순수 IPv6 패킷분석이 필요하며 라우팅 속도 개선을 위해 3단계 계층적 라우팅 구조가 아닌 그 이상의 계층적 라우팅 구조도 비교, 분석하는 것이 필요하다.

7.참고 문헌

- [1] S. Nilsson and G. Karlsson, "Fast Address Look-Up for Internet Routers," In proceeding of IEEE Broadband Communications' 98, April 1998
- [2] T. Kijkanjanarat and H. J. Chao, "Fast IP Lookups using a Two-Trie Data Structure," In Proceedings of Globecom '99, 1999
- [3] Y. Rekhter, P. Lothberg, R. M. Hinden, S. E. Deering and J. Postel, "An IPv6 Provider-Based Unicast Address Format," RFC 2073, Jan. 1997
- [4] W. Stallings, "IPv6: The New Internet Protocol," IEEE Comm. Mag., pp. 96-108, Jul. 1996.
- [5] F. Amer and Y. Lien, "A Survey of Hierarchical Routing Algorithms and A New Hierarchical Hybrid Adaptive Routing Algorithm for Large Scale Computer Communication Networks," IEEE ICC '88, Vol. 2, pp.999-1003, June 1988
- [6] V. Fuller, T.Li, J. Yu and K. Varadhan, "Classless Inter-Domain Routing (CIDR): and Address Assignment and Aggregation Strategy," RFC-1519, September 1993
- [7] Merit Network, Inc. See <http://www.merit.edu>