主題

고속 라우터 기술현황과 전망

서울대학교 컴퓨터공학과 하석재, 최양희

차 례

1. 서론
2. 일반적인 라우터의 구성요소
3. 연구동향
4. 결론 및 향후전망

1. 서 론

인터넷은 급격히 늘어나는 사용자와 WWW를 비롯한 멀티미디어 응용의 사용빈도가 커져 이들이 발생시키는 트래픽의 양이 크게 증가하고 있다. 더구나 인터넷 트래픽 패턴이 LAN중심에서 WAN 트래픽 중심으로 옮겨감에 따라 트래픽을 처리하는 백본 네트워크에 부하를 가중시키고 있다. 이러한 트래픽을 원활히 처리하기 위해서 백본 네트워크의 처리속도 및 전송용량을 고속화하기 위해 제품개발 및 연구가 활발히 이뤄지고 있다.

이러한 인터넷의 백본 네트워크는 직접 전송을 담당하는 전송선로와 트래픽을 패킷단위로 나눠 처리하는 라우터나 스위치로 구성되어 있다. 전송선로의 경우 국내와 미국의 인터넷 백본 네트워크는 T3(45Mbps)급으로 이루어져 있고 현재 광섬유(optical fiber)를 사용하는 155Mbps(OC-3)에서 10Gbps(OC-192) 규격의 시험망이 활발히 연구가 진행되고 있다. 여기에 현재의 전송용량을 크게 늘일 수 있는 WDM(Wavelength Division Multiplexing)기술을 적용할 경우 차세대 네트워크로 진화하는 데 전송선로의 경우에는 큰 문제가 없는 상황이다.

하지만 트래픽의 처리속도를 늘리기 위해서는 IP 패킷을 처리하는 라우터나 스위치를 고속으로 동작할 수 있게 만들어야 하지만 IP 프로토콜이 가지고 있는 문제점으로 인해 고속화에 어려움을 겪고 있다. 대표적인 문제점이라면 패킷길이의 가변성과 IP 주소참조의 어려움 등이 있다. 〈그림 1〉과 같이 패킷의 길이가 다양한 IP 트래픽을 대상으로 고속으로 패킷을 처리하는 패킷교환기술을 구현하기가 어렵고, 패킷의 다음 홉을 알아내기 위해서 라우팅테이블을 참조해야 하는 데 일정하지 않은 라우팅테이블 참조시간 등이 고속으로 동작하는 라우터나 스위치를 제작하기 위해서는 필수적으로 해결해야 하는 문제점이다.

또한 차세대 인터넷은 음성전송이나 비디오와 같은 멀티미디어 트래픽을 원활히 다룰 수 있도록

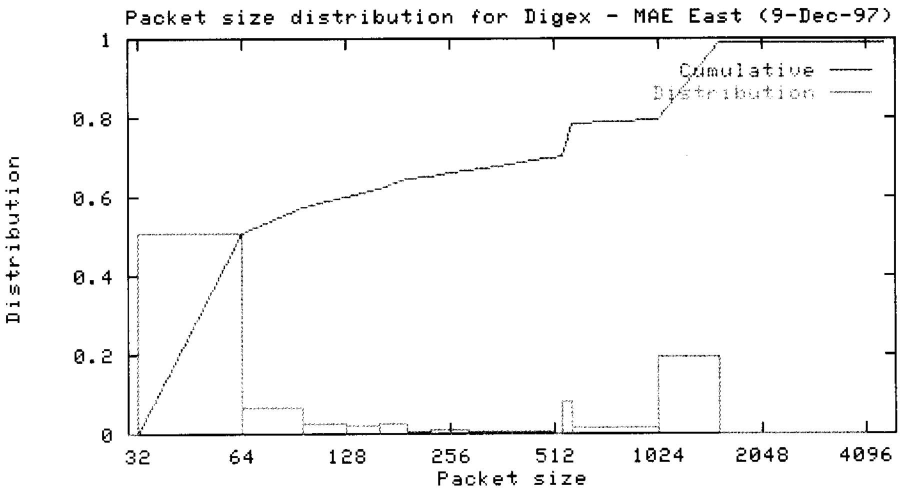


그림 1. IP 패킷의 길이분포[1]

QoS(Quality of Service:서비스품질)을 보장하는 것이 중요한 기술적인 요구사항이 되고 있다. 이와 같이 트래픽을 고속으로 처리하면서도 QoS를 만족시키는 고속 라우터를 설계 및 제작하기 위해서는 많은 노하우 및 기술적인 해결방안이 필요하다.

본 글에서는 고속 라우터를 제작하기 위해서 라우터의 각 요소별로 해결해야할 문제점에 대해 알아보고 이들의 해결하는 방안이나 기술동향에 대해 알아보도록 하겠다.

본 글의 구성은 다음과 같다. 2장에서는 고속라우터의 각 요소별로 필요한 기술적 요구사항에 대해서 알아보고 3장에서는 이에 대한 현재 제시되고 있는 해결책을, 4장에서는 결론 및 향후 전망에 대해 알아보도록 하겠다.

2. 일반적인 라우터의 구성요소

일반적인 라우터의 구조는 〈그림 2〉과 같다[1]. 입출력포트는 실제 네트워크인터페이스에 연결되어 IP 패킷을 읽어들이고 내보내는 역할을 한다. 입력포트부터 패킷을 읽어 IP주소 참조를 하여 스위칭연결망에 전달하는 부분을 포워딩엔진(Forwarding Engine:FE)이라고 하고 FE와 출력포트를 연결하는 라우터의 내부네트워크를 스위칭연결망이라고 한다. 이들을 연결하는 방식은 여러 가지가 있으며 성능이나 구현복잡도 면에서 서로 다른 특성을 가진다.

2.1 일반적인 라우터의 동작

라우터는 패킷이 입력포트로 들어오면 포워딩엔진에서 패킷의 전송에러 유무를 검사하고 난 뒤 라우팅테이블을 검색해서 목적지 IP주소에 해당하는 출력포트를 찾아낸 후 스위칭연결망을 통해 출력포트로 보내지게 된다. 이외에도 FE에서는 TTL (Time-to-live) 값을 감소시키고 체크섬을 재계산하고 패킷의 MTU(Maximum Transimission Unit) 값에 따라 패킷을 자르고 재결합(Fragmentation and Reassembly)하는 작업이 일어나게 된다〈그림 3〉.

FE와 출력포트를 연결하는 스위칭연결망의 경우 일반적으로 버스(bus)가 많이 사용된다.

패킷이 스위칭연결망을 통해 출력포트에 보내지

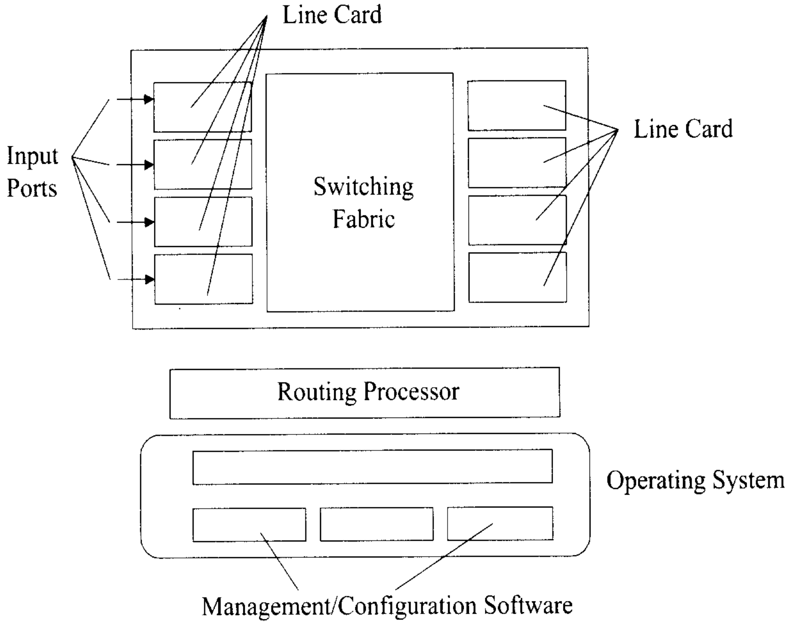


그림 2. 라우터의 아키텍쳐[2]

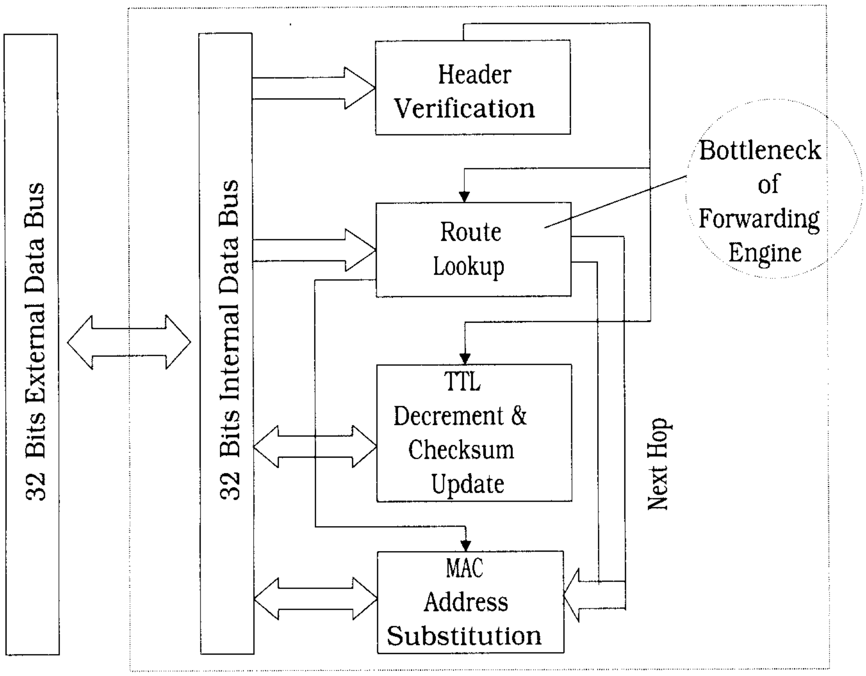


그림 3. 포워딩엔진 아키텍쳐[3]

면 큐에서 대기하게 있고 스케줄러가 큐에 들어있는 패킷을 대상으로 스케줄한 다음 다음 홉으로 포워딩하게 된다. 이 때 일반적으로 IP 패킷이 처리되는 순서는 FIFO방식이고 큐가 다 차게 되면 이후에 들어오는 패킷들은 버리게 된다(drop-tail).

라우터는 위와 같은 처리를 패킷마다 반복하게 되고 이 과정을 고속화하기 위해서 발생하는 문제점에 대해 각 구성요소별로 알아보도록 하겠다.

2.2 포워딩엔진(Forwarding Engine:FE)

2.2.1 포워딩엔진의 구조

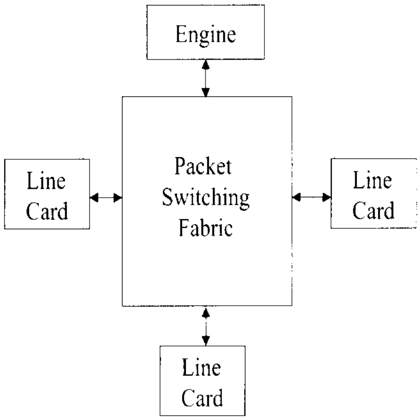


그림 4. 중앙집중식 아키텍쳐[4]

포워딩엔진의 중요한 역할은 주로 IP 패킷의 목적지주소로 라우팅테이블을 참조해 출력포트를 알아내 출력포트로 보내는 것이다. 포워딩엔진의 형태는 저속라우터의 경우는 여러 개의 입출력 포트와 버스구조로 물려서 동작된다. 이 경우를 중앙집중식구조(centralized architecture)라고 하고 여러 입출력포트의 처리를 하나의 라우팅엔진이 모두 처리하게 되어 고속화할 경우 심각한 병목현상을 발생하게 된다〈그림 4〉.

고속라우터를 구성하기 위해서 일반적으로 채택되고 있는 아키텍처는 분산형구조(distributed architecture)와 병렬형구조(parallel architecture)가 있다〈그림 5,6〉.

분산형구조는 각 카드마다 포워딩엔진을 장착하고 라우팅을 직접 처리하기 때문에 FE의 부담을 줄여서 중앙집중식 구조의 문제점이던 병목현상이 크게 줄어들게 된다. 하지만 이 구조의 단점으로는 각 카드에 달려있는 포워딩엔진에 의해 처리속도가 제한된다는 문제점이 있다. 즉 한 포트에 트래픽이 몰리면 어쩔 수 없이 성능의 감소가 발생한다는 것이다. 병렬형구조는 포워딩엔진이 여러 개 존재하고 각 카드에서 이를 클라이언트/서버 구조로 연결한 형태이다. 이 방식의 장점은 카드마다 포워딩엔진을 달지 않아도 되고 라우팅속도를 높이고 싶으면 포워

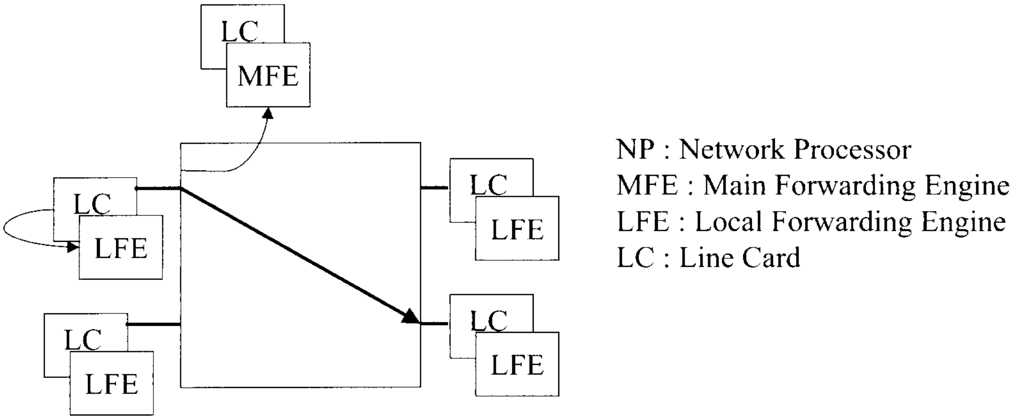


그림 5. 분산형 아키텍쳐[4]

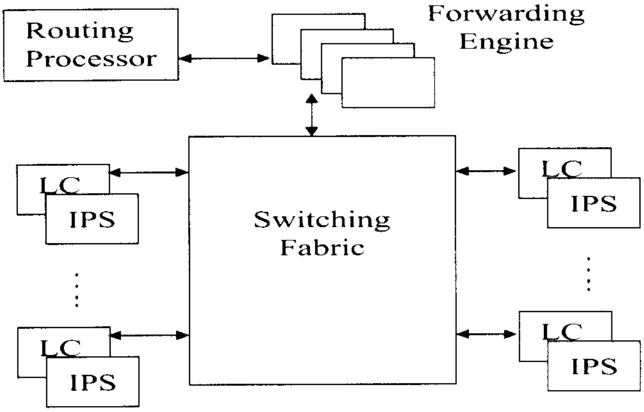


그림 6. 병렬형 구조[4]

딩엔진을 더 장착하면 되는 장점이 있다. 이 경우에는 각 입출력카드와 포워딩엔진 사이에는 비블로킹 성능을 보장하는 고속 스위칭연결망(switching fabric)이 필수적이고 이들 사이에 적절히 부하(load)를 분배할 수 있는 효율좋은 알고리즘이 필수적으로 요구된다.

분산형과 병렬형구조는 고속라우터를 설계할 때 고려해야 할 중요한 선택사항이며 일반적으로 병렬형구조가 비용이 많이 들지만 라우터의 확장성(scalability)측면에서는 장점을 가지고 있는 것으로 알려져 있다.

2.2.2 IP 주소참조(IP Address Lookup)

IP 패킷의 목적지 주소를 보고 다음 경로를 알아내기 위해서 라우팅테이블에서 해당되는 주소의 엔트리를 검색하게 되는데 이 경우 IP주소의 체계의 특징으로 인해 Longest prefix matching (LPM) 문제에 부딪치게 된다. LPM문제란 예를 들면 라우팅 테이블에 147.46/16/5, 147.46.114/24/5(IP주소/prefix길이/출력포트)와 같은 엔트리가 있고 IP패킷의 목적지 IP주소가 147.46.114.112일 경우 위의 두 가지 엔트리가 모두 매칭이 되고 이중에서 해당되는 prefix중 가장 긴 147.46.114/24 의 엔트리의 출력포트 값이 5번 포트로 패킷을 내보내게 된다[5]. 이러한 CIDR문제는 기존의 검색방식인 정합(exact matching) 방식에서 많이 사용되는 바이너리 검색, 해쉬기법들을 직접적으로 적용하기 어렵게 한다. 또한 해당되는 엔트리 전체를 검색해야 하기 때문에 하나만 찾으면 검색이 끝나는 기존의 검색방식과는 많은 차이가 있다. LPM문제는 CIDR (Classless Inter-Domain Routing) [3]라고 불리는 IP 주소할당방식이 널리 사용되면서 발생한 문제이다. CIDR방식의 주소할당방식은 기존의 클래스별로 할당해 C 클래스 주소공간의 부족 및 A,B 클래스의 주소공간의 낭비를 가져온 기존의 IP 주소할당방식의 낭비를 줄여 IP 주소할당방식의 효율적인 운영이 가능하도록 했지만 라우터가 유지해야 하는 라우팅테이블의 엔트리가 크게 늘어나게 되어 결과적으로 평균IP주소참조시간이 느려지게 되었다. 특히 CIDR의 사용으로 인해 발생한 LPM문제는 대부분의 트래픽을 처리하는 백본 네트워크 라우터의 병목현상을 야기시켰다.

기존의 라우터에서는 일반적으로 바이너리 trie의 변형인 Patricia trie로 LPM 문제를 해결하고 있다〈그림 7〉. Patricia trie의 경우에는 즉, trie의 길이가 긴 엔트리일수록 메모리참조 및 비교가 늘어나기 때문에 매칭되는 엔트리를 찾는 일정하지

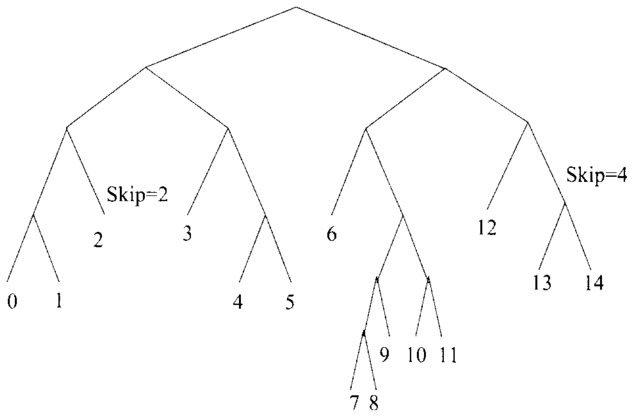


그림 7. Patricia Trie

않다. 이런 이유 때문에 Patricia trie는 IP 주소 참조를 고속화하기에는 적합하지 않은 자료구조이다.

위에서 본 것과 같이 LPM문제는 결국 주어진 IP 주소에 대해 같은 공통 prefix를 갖는 모든 prefix를 찾아내고 이 중 가장 길이가 긴 prefix를 매칭시키는 문제이다. LPM 문제는 고속라우터를 제작하는 데 중요한 성능상의 문제를 야기하고 이를 해결하기 위해 활발한 연구가 진행되고 있다.

2.3 스위칭연결망(Switching Fabric)

스위칭연결망은 FE에서 처리된 IP 패킷을 출력 포트로 보내기 위해서 거치는 라우터 내부네트워크다. 보통 중저속의 라우터에서는 버스가 많이 쓰이지만 한 번의 하나의 패킷만 처리할 수 있다는 단점 때문에 버스를 사용하여 스위칭연결망을 구성할 경우에는 라우터나 스위치 내부의 스위칭속도가 각 포트가 지원하는 속도와 포트의 수를 곱한 정도의 속도가 필요하게 된다. 이러한 버스방식의 라우터를 설계하기 위해서는 지원하는 포트가 늘어날수록, 포트의 속도가 빨라질수록 만들기가 어렵게 된다.

이를 위해서 보통 고속의 라우터나 스위치에서는 크로스바(crossbar)나 반얀(banyan) 네트워크와 같은 비블로킹(non-blocking) 스위칭연결망이 사용된다. 비블로킹스위칭 스위칭연결망은 포트의 동작속도를 그대로 유지하면서 전체 포트가 동작가능하기 때문에 라우터나 스위치의 고속 스위칭연결망으로 사용되는 것이 일반적이다. 하지만 비블로킹 스위칭망을 라우터의 스위칭연결망으로 사용할 경우 IP 패킷의 가변길이 때문에 스위칭연결망이 고속으로 동작하려면 성능저하가 발생하게 된다. 즉, 동시에 처리되는 패킷중 가장 길이가 긴 패킷이 처리될 때까지 다음 패킷의 처리가 늦어지게 되는 문제점이 있다. 이 문제를 해결하기 위해서 라우터는 패킷이 스위칭연결망을 지나가기 전에 전처리(preprocessing)을 하게 된다.

보통 고속라우터를 제작하는 데 많이 고려되는 스위칭연결망으로 크로스바가 있다. 크로스바는 여러 개의 패킷들이 서로 다른 포트로 스위칭될 경우 동시에 스위칭이 가능한 장점이 있으나 지원하는 포트의 수가 늘어나면 포트의 수의 제곱에 해당하는 스위치가 필요하기 때문에 확장성(scalability)에 문제점이 있다. 이 문제를 해결하기 위해 ATM에서는 반얀 스위칭연결망이나 오메가 네트워크와 같은 다단(multi-stage) 스위칭연결망을 사용하고 있다. 하지만 현재 고속라우터의 경우에는 보통 지원하는 포트의 수가 16-64개 정도로 비교적 필요한 포트

수가 작기 때문에 크로스바 스위치를 채택하는 경우가 많다.

2.4 스케줄 기법과 큐 관리기법

2.1에서 알아본 바와 같이 라우터의 각 출력포트에서는 큐를 가지고 패킷을 큐에 저장하고 순서대로(First-In-First-Out:FIFO) 스케줄해 다음 홉으로 포워딩하게 된다. 이 때 IP 패킷의 가변길이 특성 때문에 가변길이 큐를 관리해야 하기 때문에 패킷당 처리시간이 일정하지 않고 패킷의 평균대기시간이 가변적인 문제점이 발생한다. 또한 출력포트에는 단일 큐가 있고 큐가 다차는 경우 새롭게 도착하는 패킷은 버린다(trop-tail). 이 경우 TCP 수준에서는 트래픽혼잡(traffic congestion) 때문에 전송에러가 발생한 것으로 판단하고 윈도우 크기를 줄여 전송속도를 감소시키게 된다.

문제는 이러한 현상이 라우터를 경유하는 모든 연결(connection)에 대해 다르게 나타난다는 데 있다. 이는 상황에 따라 각 연결별로 받는 서비스 수준이 다르다는 것을 의미하기 때문에 서비스의 공정성(fairness)측면에서 문제가 많다. 이러한 큐 관리기법은 라우터가 QoS를 보장하게 하려는 차세대 네트워크의 조건을 만족시키기 위해서는 많은 변화가 필요한 부분이다. 즉, QoS를 보장하기 위해서는 새로운 큐 관리 기법 및 스케줄기법이 필요하다.

3. 연구동향

3.1 고속 IP 주소참조(High Speed IP Address Lookup)

IP 라우터의 주소참조문제는 2.2.2에서 알아본 바와 같이 고속라우터를 제작하기 위해 개선해야 하는 상당한 성능상의 병목이다. 이 문제를 해결하기 위해서 접근하는 방식으로는 크게 두 가지가 있다. IP 주소참조속도를 고속화시키기 위해 새로운 기법을 제안하거나 비용이 많이 드는 IP 주소참조를 가능한 한 줄이려는 접근방법이 있다. 전자의 경우는 순수 IP에 기반해 IP주소참조과정을 고속화시키는 기술이고 후자의 경우는 MPLS(Multiprotocol Label Switching) 기술이나 IP 스위치 기술과 같은 3계층 스위치(Layer 3 switch)기술이다.

두 가지 접근방식은 동시에 적용이 가능하고 이러한 기술을 같이 적용한 제품이 제작되는 경향이 많다. 예를 들면, 분산구조의 고속라우터의 경우에 각 라인카드별로 IP스위칭 모듈을 두어 대부분의 트래픽을 처리하고 라우팅을 테이블을 참조해야할 경우에 주 FE에는 순수 IP 기반의 고속주소참조기법이 적용가능하다.

3.1.1 순수 IP에 기반한 기술

2.2.2에서 알아본 LPM문제를 해결하면서도 고속으로 IP주소참조가 가능한 기법이 최근 활발한 연구가 진행되고 있고 일부 결과가 발표되어 고속라우터의 포워딩엔진에 탑재되고 있다. 고속 IP주소참조기술는 일반적으로 고속으로 동작해야 하는 인터넷 백본 라우터를 적용대상으로 하며 백본 라우터의 주요 성능상의 병목을 해결하게 된다.

인터넷 트래픽이 급격히 증가하게 되고 WAN 트래픽이 증가하면서 SOHO나 사무실에서 사용하는 액세스라우터나 기업에서 사용하는 엔터프라이즈라우터에는 대부분의 패킷을 기본경로(default path)로 라우팅하고 실제 경로배정은 대부분 백본 네트워크의 라우터에서 담당하게 된다. 이로 인해서 라우팅테이블의 크기가 커지게 되고 CIDR의 적용빈도가 늘어남에 따라 하나의 IP 주소에 대해 매칭되는 엔트리가 늘어나게 된다. 현재 가장 커다란 인터넷 백본 라우터(MAE-EAST)의 경우에는 라우팅테이블의 엔트리 수가 약 55,000개이고 라우팅테이블의 크기는 점점 증가하고 있다.

일반적으로 기가비트급 처리속도를 가지고 고속 라우터를 제작하려고 하면 FE의 주소처리속도가 평균 IP 패킷길이를 기준으로 보통 2백만 개에서 4백만 개 정도의 패킷을 처리하는 수준은 되어야 한다. 즉, 55,000개의 라우팅테이블을 검색하여 매칭되는 엔트리를 전부 찾아 가장 긴 prefix를 찾는 작업을 1/2,000,000~1/4,000,000 초 내에서 마쳐야 하기 때문에 이러한 요구조건을 만족하시키는 포워딩 엔진을 제작하려면 현재의 DRAM의 latency를 60-70ns로 잡았을 때 IP주소참조가 2~3 번의 참조로 끝나야만 한다. 이러한 성능은 2.2.2에서 본 patricia trie의 경우 많은 메모리참조 때문에 도저히 맞출 수가 없다.

최근에 이러한 요구조건을 만족시키는 고속 IP 주소참조기법이 여러 가지가 제안되었는데 크게 하드웨어에 기반 한 방법, Multibit trie를 이용한 방법, 직접 참조 기법, 해쉬을 사용한 기법, LC(Level Compressed) trie를 사용한 기법 그리고 다른 다양한 방법들이 제안되고 있다.

하드웨어에 기반 한 방식은 CAM(Content-Addressable Memory)를 사용하는 방법이 있다. 하지만 이 방식은 당연히 빠른 속도를 보장하지만 많은 수의 prefix를 저장하지 못하고 가격이 비싸다는 단점이 있다. 그 외에는 캐쉬에 기반한 방식이 있다.

직접 참조 기법은 IP 주소공간(2^{32})에 해당하는 메모리에 라우팅테이블을 저장하는 방식이다〈그림 9〉. 이 경우 prefix expansion 기법을 사용하게 되는데 예를 들면 147.46으로 시작하는 prefix의 경우 147.46.0.0에서 147.46.255.255에 해당하는 엔트리에 모두 같은 출력포트를 기록하는 방식이다. 이와 같이 IP 주소참조가 한 번의 메모리참조로 가능하게 되지만 대용량의 메모리가 필요하고 prefix의 출력포트의 값이 바뀌는 경우 해당되는 prefix의 범위 전체를 업데이트해야 하기 때문에 엄청난 오버헤드가 존재하게 된다.

최근에 테이블의 압축과 약간의 계산을 통해 직접 참조 기법의 많은 메모리요구량을 줄이면서 빠른 참조가 가능한 기법이 제안되었다[9,10].

직접 참조 기법과 바이너리 trie를 적절히 조합한 방식이 Multibit trie이다. 〈그림 8〉에서와 같이 제안된 stride 단위로 확장시키고(expand) 이들 테이블을 trie로 연결하는 방식이다. 이 경우 직접 참조 기법과 같이 전체를 다 확장시키지 않기 때문

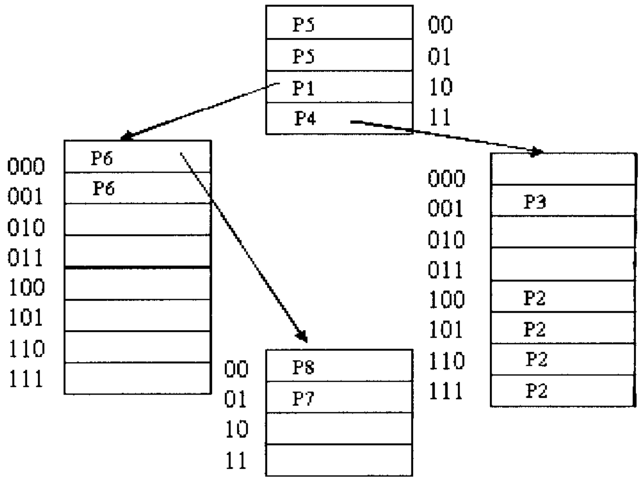


그림 8. Multibit Trie

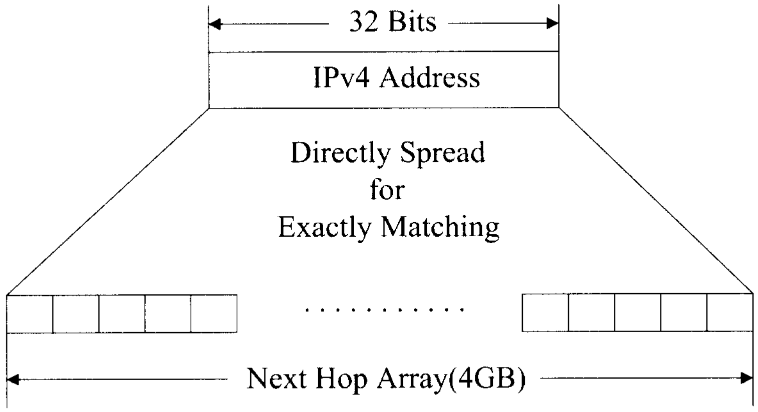


그림 9. 직접 참조 기법

에 사용메모리와 속도를 적절히 조절할 수가 있다. 이 방식은 stride의 길이와 레벨을 조정하는 것이 성능에 중요한 영향을 미친다. 일반적으로는 8-8-8 stride가 쓰인다.

해쉬기법을 적용한 기법은 prefix길이마다 별도의 해쉬테이블을 두고 목적지 IP주소의 longest prefix를 빠르게 찾을 수 있도록 binary trie를 적용한 기법도 있다[12].

LC trie기법과 같이 patricia trie를 변형해 trie의 레벨을 가능한 줄여 메모리 참조회수를 줄이고 라우팅테이블사이즈를 줄여 SRAM에 들어갈 수 있도록 하여 빠른 주소참조가 가능한 기법도 제시되어 있다.

보통 고속 IP 주소참조기법은 테이블생성시간이나 업데이트시간을 희생하더라도 빠른 주소참조시간을 보장하도록 고안하는 것이 일반적이다. 이는 인터넷 백본 라우터의 경우에 보통 30-60초에 한번 테이블 업데이트가 일어난다는 관찰결과에 기반한 것이다.

그리고 고속라우터는 라우팅테이블 생성이나 계산은 라우팅프로세서가 하고 각 포워딩엔진에는 이를 일정시간마다 다운로드하는 방식으로 포워딩엔진의 부담을 줄이는 경우가 많다.

3.1.2 IP스위칭 및 MPLS(Multiprotocol Label Switching)

순수 IP라우터에서는 IP 주소참조를 고속으로 하기 위해 여러 가지 기법을 제안한 반면에 IP 스위칭이나 MPLS에서는 IP 주소참조를 가능한 한 줄이기 위해 다른 기술을 사용하고 있다. ATM에서는 VCI/VPI 참조시 고정길이를 사용하고 전체 대상 엔트리의 수가 작기 때문에 메모리속도의 고속참조가 가능하다. 각 스위치간 사이의 포워딩은 VCI/VPI 매핑테이블을 보고 이뤄지게 된다. 이런 이유로 ATM 에서는 상대적으로 IP 라우터에 비해 고속으로 패킷(셀)의 처리가 가능하다. 이런 개념을 IP라우터에 적용한 것이 바로 IP 스위치 기술이다. IP 스위치 기술은 Ipsilon사에서 처음 제품화시킨 개념으로서 처리하는 패킷의 일정한 흐름(flow)을 찾아내 별도의 VC 경로를 할당해서 흐름에 해당하는 패킷의 경우에는 주소참조를 하지 않고 고속 스위칭이 가능하도록 만든 기술이다.

이러한 IP 스위칭 관련기술을 표준화한 MPLS[4]는 기존 IP에서 고속스위칭을 하기 위해 특정 패킷의 주소나 포트와 같이 일정한 규칙을 설정하고 이에 해당되는 패킷에 대해서는 레이블(label)을 부여해 MPLS망에서는 IP 주소가 아닌 레이블을 기반으로 고속스위칭을 하는 기술이다

(label swapping).

레이블을 할당하는 것은 MPLS망의 진입점인 LER(Label Edge Router)에서 하고 망 내부의 라우터인 LSR(Lable Switched Router)는 레이블이 할당되어 있으면 정의된 레이블 변환 테이블을 보고 스위칭하게 된다. 이러한 MPLS 에서는 LER은 상대적으로 IP라우터보다 많은 작업을 하지만 중간의 LSR에서는 IP 패킷의 주소를 보지 않고 할당된 레이블만 보고 스위칭하게 되어 망 전체적으로 빠른 패킷처리가 가능하게 된다.

레이블을 할당하는 방식은 control-driven과 traffic-driven 방식으로 나뉘는 데 control-driven 방식은 특정 흐름에 대한 레이블값을 미리 할당하고 들어오는 패킷을 검사해 할당된 레이블이 있으면 이를 할당해주는 방식이다. traffic-driven 방식은 LER이 처리하는 패킷을 계속 모니터링하고 있다가 흐름이 일정한 threshold가 넘으면 레이블을 하나 할당해 주는 방식이다. LER, LSR사이의 레이블 할당은 LDP(Label Distribution Protocol)가 담당하고 하나의 흐름은 일정한 LSP(Label Switched Path)를 통해 스위칭이 된다.

MPLS의 중요한 기술적인 이슈로는 LER의 트래픽분류기(traffic classfier)에서 관리하는 레이블의 개수를 적정한 수준으로 유지하면서도 QoS와 같은 서비스수준을 잘 보장해 주는 것이다. 만약 정의가능한 모든 흐름의 조합(Source IP, Source Port, Destination IP, Destnation port, protocol)에 대해 레이블을 할당하게 되면 많은 개수의 레이블이 필요하게 된다. 특히 현재 MPLS를 ATM스위치 위에 구현하려고 하는 측에서는 MPLS의 레이블을 ATM의 VCI/VPI 조합으로 대응시키게 되는 데 ATM에서 지원할 수 있는 VCI/VPI의 개수가 한계가 있기 때문에 레이블의 수를 적절하게 유지시키는 것이 중요하다. 이를 위해서 흐름을 aggregation하거나 여러 개의 조합을 적절히 조정해서 생성되는 레이블의 개수를 조정하는 것이 연구되고 있다.

3.2 스위칭연결망

보통 고속라우터에는 크로스바 그리고 배처-반얀 네트워크와 같은 비블로킹 성질을 갖는 스위칭연결망이 사용된다. 또한 IP 패킷은 길이가 가변이라 스위칭성능을 높이기 어려운 문제를 가지고 있다. 이를 내부적으로 일정한 길이의 패킷으로 자른 뒤 스위칭하고 출력단에서 다시 재결합함으로써 고속스위칭이 가능하도록 해결했다. 보통 고속 IP 라우터는 ATM의 스위칭연결망을 코어로 하고 IP 패킷의 가변길이를 지원하기 위해서 세그멘테이션과 재결합(segmentation & reassembly) 모듈을 결합하고 있는 구조를 가지고 있다.

현재의 고속 IP 라우터에서는 ATM 스위치에 비해 상대적으로 작은 포트(16~64)를 제공하기 때문에 상대적으로 복잡도가 높지 않은 크로스바를 많이 사용하며 또한 크로스바의 경우 멀티캐스팅 문제를 쉽게 처리가능 하다는 장점도 있다.

3.3 QoS 요구조건을 만족시키는 스케줄 방식과 큐 관리기법

3.3.1 스케줄 방식

라우터에서 각 연결마다 QoS수준을 만족시키려면 기존의 FIFO 방식의 스케줄러와 trop tail 방식의 큐 관리기법으로는 한계가 있다. 그래서 QoS 기반의 스케줄 기법으로는 연결이나 흐름(flow)별로 큐를 두고 스케줄러가 QoS요구조건을 참조해서 스케줄하는 WFQ(Weighted Fair Queueing) 기법이 가장 적합하다고 알려져 있다. 하지만 이러한 WFQ는 구현하기가 어렵고 다루는 큐의 개수나 포트의 수가 늘어나면 복잡도가 증가하게 된다. 이를 위해서 요즘에는 복잡도를 어느정도 줄이면서 스

케줄가능한 SCFQ, CBQ등과 같은 기법이 제안되고 있다.

3.3.2 큐 관리기법

보통 큐의 위치는 출력포트에 있는 것이 일반적이다. 즉, QoS수준을 맞추며 스케줄가능하게 하려면 출력단에서 하는 것이 효율적이라는 이야기이다. 하지만 고속라우터에서는 이 경우 가장 문제가 되는 것이 Head-of-line(HOL) 블로킹문제와 출력포트 speedup이다.

HOL 블로킹문제란 입력단에 두 개 이상의 패킷들이 동일한 출력포트로 가려고 경우 출력단에서는 한 번에 하나의 패킷만 처리할 수 있기 때문에 나머지 패킷들은 기다려야 한다. 이 때 큐에서 기다리는 패킷의 뒤에서 다른 출력포트로 가는 패킷의 경우에는 불필요하게 블로킹되는 문제가 발생하게 된다. HOL 블로킹 문제는 라우터의 성능을 심각하게 제한하게 되는 데 이론상 하드웨어가 낼 수 있는 속도의 58.6%의 속도 밖에 내지 못한다[6]. 이는 출력포트만 큐잉하는 경우에 발생하는 심각한 성능상의 제한사항이다.

HOL 블로킹을 없애기 위해서는 출력포트가 입력포트의 수만큼 빠르게 동작하면 된다(n-speedup). 즉, HOL 블로킹 문제는 동일한 출력포트로 가는 패킷이 발생할 경우에 발생하므로 이를 없애기 위해서는 출력포트가 입력포트보다 빠르게 동작하도록 만들면 HOL 블로킹문제는 발생하지 않게 된다. 하지만 문제는 출력포트의 동작속도를 빠르게 만드는 데는 한계가 있다는 데 있다. 이를 해결하기 위해 입력포트에도 큐를 두는 VOQ (Virtual Output Queueing)이라는 기법이 제안되었다<그림 10>[6].

VOQ 기법은 HOL 블로킹 문제를 없애기 위해 각 입력포트마다 출력포트의 수만큼의 큐를 두어 HOL 블로킹문제를 발생하지 않도록 했다. 여기다 i-SLIP이라는 스케줄링 기법을 제시하고 이를 동시에 적용할 경우에 출력포트의 speedup을 2배로만 해도 이론상 성능의 99%까지 낼 수 있다는 실험결과가 제시되었다.

3.4 서비스수준(QoS)보장

3.4.1 QoS 제공능력

IP 망의 중요한 단점 중 하나는 QoS(서비스품질:Quality of Service)를 제공하지 못하는 것이

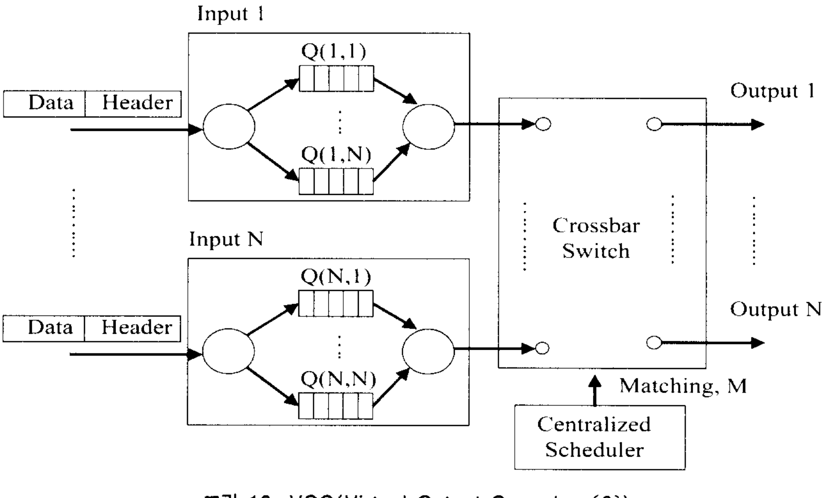


그림 10. VOQ(Virtual Output Queueing [6])

다. 즉, 기존의 IP에서는 패킷이 들어온 순서대로 처리되고 각 패킷이 서비스 받는 수준인 QoS가 상황에 따라 일정하지 않았다. 하지만 최근에 이를 해결하기 위해 여러 가지 연구가 활발히 진행하고 있다. MPOA(Multiprotocol over ATM)와 같이 ATM위에 IP서비스를 제공하는 경우에는 별도의 VC를 설정하고 QoS를 제공하는 것이 가능하다. MPLS에서도 QoS를 제공하기 위해 활발히 논의되고 있다.

순수 IP 네트워크 상에서 QoS를 제공하기 위해서 IETF에서는 IntServ(Integrated Service)를 제안했다. IntServ에서는 RSVP(Resource Reservation Protocol)를 제안하고 이를 기반으로 종단간(end-to-end) QoS를 보장하는 구조이다. 하지만 프로토콜이 복잡하고 확장성 문제에 부딪쳐 크게 관심을 끌지 못하고 있다.

인터넷에서 QoS를 보장하기 위해 최근 주목받고 있는 기술로서 DiffServ(differentiated service)가 있다. DiffServ[8]는 인터넷상에서 종단간 QoS이 불가능하다는 것을 인정하고 라우터간(hop by hop)의 제한된 QoS만을 보장하자는 움직임으로 최근에 커다란 관심을 끌고 있다. DiffServ에서는 해당 TOS값에 따른 우선순위 기반의 차별화된 서비스가 가능하다.

인터넷을 개선하기 위해 제안된 ATM이나 IntServ, IPv6등 여러 가지 기술들이 인터넷의 구조를 수정하거나 대체해야 하는데 비해 DiffServ는 이러한 수정을 최소화하고 있다. 즉, IP 헤더 중 거의 사용되지 않았던 TOS(type of service)바이트를 이용해 제한된 몇 가지 레벨의 서비스 수준인 CoS(Class of Service)만을 정하고 패킷에 원하는 서비스클래스인 TOS값을 설정해 보내면 해당 라우터에서 이를 기반으로 스케줄링을 하는 것을 말한다.

하지만 ATM과 같이 종단간 QoS를 보장하는 것이 아니고 홉단위로 서비스 우선순위만을 제공하는 것이기 때문에 완벽한 QoS를 제공하는 것이라고는 볼 수 없다. 또한 해당 CoS에 대한 처리는 라우터마다 다르게 설정할 수 있기 때문에 여러 가지 다른 정책들의 적용이 가능하다(policy-driven).

또한 DiffServ의 경우에는 MPLS나 ATM과 같이 망 전체가 MPLS기능을 제공할 필요 없이 DiffServ를 지원하지 않는 라우터나 장비의 경우에도 별다른 문제없이 연동가능 하다는 장점이 있다.

3.4.2 흐름 식별(Flow Identification)[2]

3.3에서 본 것과 같이 차세대 인터넷이 가져야 하는 중요한 요구사항 중 하나가 QoS를 제공하는 것이다. 이에 차세대 라우터관련기술들은 공통적으로 기존의 IP가 제공하지 못했던 QoS(Quality of Service)와 같은 새로운 기능과 고속 IP 처리능력을 제공하고 있다.

이를 위해서 기존 라우터구조에서 포워딩엔진을 개선함으로써 개선된 성능을 얻으려고 하는 노력을 하고 있다. MPLS나 DiffServ와 같은 기술의 성패는 흐름(flow)을 효율적이고 빠르게 식별하는 것이 가장 중요하다고 할 수 있다.

흐름이란 일정한 시간동안 같은 source/destination을 가지는 연속된 패킷으로 정의되는데 좀 더 엄밀하게 정의하면 흐름은 보통 〈source IP(32), source port(16), destination IP(32), destination port(16), protocol(8)〉(총 104비트)의 조합으로 표시되며 이를 식별할 수 있는 규칙을 포워딩엔진에 설정해두고 해당되는 패킷이 들어올 경우에는 별도로 정의된 경로로 스위칭하게 된다. 또한 얼마간의 시간간격으로 패킷이 들어와야만 하나의 흐름으로 보느냐에 따라 MPLS와 같은 스위칭에서는 관리해야하는 흐름의 개수가 차이가 난다. 하지만 위에서 말한 다섯 가지 인자를 가지고 있는 테이블과 매 패킷의 헤더를 비교하게 되면 라우터 입장에서는 커다란 부하가 걸리게 된다. 보통 IP 라우터가 32비트의 목적지 IP주소만 보는 데 비해

흐름을 식별하기 위해서는 최대 104비트의 내용을 모두 비교해야 하기 때문이다.

이와 같은 문제는 네트워크의 보안을 담당하는 방화벽(firewall)에서도 발생하게 된다. 패킷필터링을 통해 패킷의 입출력규칙을 정하고 이에 해당되는 패킷만 선별적으로 통과시키는 방화벽도 기본적으로 위에서 말한 다섯 가지의 값을 비교해 패킷을 처리하게 된다.

방화벽과 고속라우터의 패킷분류기에서는 흐름처리나 패킷필터링 작업을 고속으로 처리해기 때문에 고속화하기 위해서는 3.1.1에서 제시한 고속 IP 주소참조기법보다 어려운 점이 많아 아직 연구성과가 미진한 편이다. 최근에 기가비트 속도의 방화벽을 제작할 수 있는 연구결과가 제시되었다[9].

3.4.3 트래픽 engineering

MPLS의 중요한 활용분야가 트래픽 engineering 측면의 적용이다. 즉, 기존의 라우터는 이 패킷 단위로 처리되었기 때문에 라우팅테이블 값이 변하면 동일한 흐름에 해당하는 패킷이 서로 다른 경로를 통해 전달가능하기 때문에 QoS 보장이나 트래픽의 흐름을 제어하는 트래픽 engineering 측면에서는 어려움이 있었다. 하지만 MPLS에서는 하나의 레이블 값을 갖는 흐름들은 설정LSP를 따라 스위칭되기 때문에 네트워크의 부하(load)에 따라 적절히 밸런싱해 네트워크의 효율을 높여주는 것이 가능하다.

4. 결론 및 향후전망

본 글에서는 고속라우터를 제작하기 위해 필요한 문제점 및 이들을 해결하기 위한 기술에 대해 알아보았다. 고속 IP주소 참조기술에 관련해서는 직접 참조 기법과 Multibit trie와 같은 순수 IP 기반의 기법을 , IP 스위칭이나 MPLS기술과 같이 주소참조를 가능한 한 줄이는 기법에 대해서 알아보았다. 그리고 고속 라우터에서 사용되는 스위칭연결망, 그리고 QoS를 제공하기 위한 스케줄 기법과 큐 관리 기법을, 흐름을 효율적으로 인식하고 망의 효율성을 높이는 트래픽 engineering 기법에 대해 알아보았다.

세계적으로 차세대 인터넷을 구성하는 핵심기술인 고속라우터에 대한 핵심기술을 확보하고 시장을 선점하기 위해 연구 및 기술개발경쟁을 하고 있다. 국내에서도 98년 말부터 라우터 관련 핵심기술의 중요함을 인식하고 관련기술에 대한 투자 및 연구가 시작되어 연구가 진행되고 있다.

국내에서는 MPLS라우터 및 MPOA기반 라우터를 한국전자통신연구원(ETRI)과 일부 대학 및 산업체의 공동연구로 개발하고 있으며 순수 IP기반 고속라우터도 개발할 예정으로 있다. 또한 1998년부터 국내외의 라우터 관련기술에 대해서 산학연의 전문가들이 모여 인터넷 라우터 연구회(http://mmlab.snu.ac.kr/router)를 결성하고 최신기술에 대한 정보,자료 및 의견을 교환하고 있다.

라우터분야는 시스코를 비롯한 미국의 대형 네트워크 회사들이 세계시장의 대부분을 장악하고 있는 분야이다. 인터넷의 성장에 발맞춰 이들 대형 네트워크 회사들의 영향력이 점점 증가하고 있는 상황에서 국내에서도 자체 라우터 관련 기술개발에 노력해 기술을 축적하여 세계수준의 경쟁력있는 네트워크 관련기술을 확보해야 할 것이다.

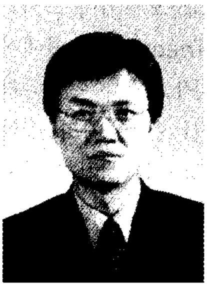
※참고문헌

[1] CAIDA, http://www.caida.org.
[2] S. Keshav and R. Sharma, "Issues and Trends in Router Design", IEEE Communications Magazine, Vol. 36

No. 5, pp.144-151, May 1998.
[3] Nen-Fu Huang and Shi-ming Zhao, "A Novel IP-Routing Looking Scheme and Hardware Architecture for Multigigabit Switching Routers", IEEE Journal of Selected Areas in Communications, Vol. 17 No.6, pp.1093-1104, June 1999.
[4] Henry C.B.Chan, Hussein M.Alnuweiri and Victor C.M Leung, "A Framework for Optimizing the Cost and Performance of Next-Generation IP Routers", IEEE Journal of Selected Areas in Communications, Vol. 17 No.6, pp.1013-1029, June 1999.
[5] V. Fuller, T. Li, J. Yu and K. Varadhan, "Classless Inter-Domain Routing (CIDR): an Address Assignment and Aggregation Strategy", RFC 1519, September 1998.
[6] Nick McKeown, "Fast Switched Backplane for a Gigabit Switched Router", Business Communications Review, vol. 27, no. 12, Dec. 1997.
[7] R. Callon, N. Feldman, A. Fredette, G. Swallow and A. Viswanathan, "A Framework for Multiprotocol Label Switching", Internet Draft, http://www.ietf.org/internet-drafts/draft-ietf-mpls-framework-03.txt, June 1999.
[8] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, December 1998.
[9] Jun Xu, and Mukesh Singhal, "Design and Evaluation of a High-Performanc ATM Firewall Switch and Its Applications", IEEE Journal of Selected Areas in Communications, Vol. 17 No.6, pp.1190-1200, June 1999.

하 석 재

1992년 3월~1996년 2월 서울대학교 컴퓨터공학과 졸업
1996년 3월~1998년 2월 서울대학교 컴퓨터공학과 석사
1997년 3월~현재 서울대학교 컴퓨터공학과 박사과정
※관심분야 : 고속라우터기술, 액티브네트웍, 무선 Ad-Hoc네트웍, 자바, 웹기술



최 양 희

1971년~1975년 서울대학교 전자공학과 학사
1975년~1977년 한국과학기술원 전기공학과 석사
1980년~1984년 프랑스 ENST 컴퓨터공학 박사
1981년 9월~1984년 6월 프랑스 CNET 연구소
1988년 7월~1989년 6월 미국 IBM 왓슨연구소, 한국전자통신연구소 책임연구원
1995년 10월~현재 서울대학교 컴퓨터공학과 재직
※관심분야 : 고속통신망, 멀티미디어통신, 인터넷