인터넷 통합 서비스 지원을 위한 라우터 구현[+]

°최 영 수*, 조 유 제*, 노 병 희**
*경북대학교 전자전기공학부, **삼성전자
*guru0109@palgong.kyungpook.ac.kr, yzcho@ee.kyungpook.ac.kr

Implementation of a Router for Supporting Internet Integrated Service

°Choi Young Soo*, You-Ze Cho*, and Byung-Hee Roh**
*School of Electronic and Electrical Engineering, Kyungpook National University,
**Samsung Electronic Co.

요 약

인터넷 통합 서비스 모델(Internet Integrated Service Model)은 인터넷에서 화상, 음성, 데이터 등의 다양한 멀티미디어, 실시간 응용들에 대해서 서비스 품질을 제공하기 위해 제안된 새로운 인터넷 서비스 모델이다. 현재 인터넷을 통해 제공되는 서비스는 최선형 서비스(best effort service)에 속한다. 따라서 인터넷 서비스 제공자는 사용자가 원하는 서비스 품질을 보장할 수 없다. 인터넷 통합 서비스 모델에서는 자원 예약 프로토콜과 새로운 구성 요소들을 추가하여 각 플로우에 대해 예약된 서비스 품질을 제공한다. 즉, 인터넷 통합 서비스 모델에서는 신호 프로토콜, 패킷 분류자, 패킷 스케줄러, 수락 제어 등의 요소를 사용하여 각 플로우에 대해 예약된 서비스 품질을 제공한다. 본 논문에서는 FreeBSD 기반의 라우터에 SCFQ(Self-Clocked Fair Queueing) 스케줄링 방식, 버퍼 관리 방식 및 RSVP(ReSource Reservation Protocol)와 커널 사이의 트래픽 제어 인터페이스 부분을 구현하여 RSVP 를 이용한 자원 예약을 가능하게 함으로써 라우터가 인터넷 통합 서비스를 지원할 수 있도록 구현하였다. 그리고 테스트 베드 상에서 기존 라우터와 구현된 라우터와의 성능 평가를 수행하였다.

1. 서 론

인터넷의 급속한 성장과 함께 이를 사용하는 사용자의 요구도 더욱 확대되고 있다. 인터넷을 사용하는 사용자들은 음성, 화상이 지원되는 멀티미디어 응용과 원격 화상 회의와 같이 사용자간에 실시간으로 상호 작용을 할 수 있는 새로운 응용들이 지원될 수 있는 환경을 요구하고 있다. 하지만 기존의 인터넷 프로토콜을 이용할 경우 인터넷 사용자는 최선형 (best effort) 서비스 품질만을 보장받게 되어 인터넷 서비스 제공자는 사용자의 다양한 요구를 충족시킬 수 없게 된다. 최근 IETF (Internet Engineering Task Force) 에서는 인터넷과 같은 패킷망에서 실시간 서비스 제공을 위한 연구를 중점적으로 진행하였다. 현재까지 인터넷에서 서비스 품질 보장을 위한 연구를 위해 IETF 의 IntServ WG 와 DiffServ WG 에서는 각각 *Integrated Service Model*, RSVP 와 *Differentiated Service* 에 대한 기술을 연구하고 있다 [1], [2]. 본 논문에서는 인터넷 통합 서비스 모델의 에서 예약된 자원의 서비스 품질을 만족시키기 위한 라우터의 핵심 구현 요소인 스케줄링 방식, 버퍼 관리 방식과 신호 프로토콜인 RSVP 와의 트래픽 제어 인터페이스를 FreeBSD 3.2 PC 기반 라우터에서 구현하였다.

본 논문의 구성은 2 장에서 서비스 품질 보장을 위한 인터넷 통합 서비스 모델에 대하여 설명하고 3 장에서는 서비스 품질 보장을 위해 라우터에 구현된 패킷 분류자, 패킷 스케줄러, 트래픽 제어 인터페이스 및 버퍼 관리 방식에 대하여 설명하고 그 구현 구조를 설명한다. 4 장에서는 성능 평가를 위한 테스트 베드 구조와 실험 환경 및 결과 분석을 하고 5 장에서는 결론을 맺는다.

2. 인터넷 통합 서비스 모델

그림 1에서는 인터넷 통합 서비스를 지원하기 위한 라우터의 일반적인 구성 요소를 나타내었다. 수신자는 자원 예약 프로토콜인 RSVP 를 사용하여 자원 예약 요구를 송신원으로 알리게 된다. 이를 위해 경로 상의 라

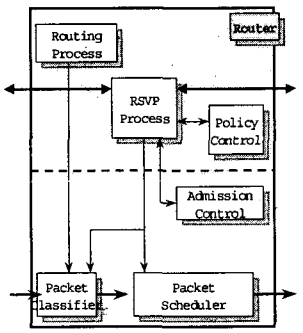


그림 1. 인터넷 통합 서비스 지원 라우터 모델

우터에서는 수신원에서 송신원 방향으로 자원 예약 요청을 송신원까지 RSVP 를 통하여 전달하게 된다. 각 라우터는 정책 제어를 통해서 자원 예약 요청의 적격 여부를 검사하고 수락 제어를 통하여 현재 자원 예약 상태를 확인하여 자원 예약 요청을 받아들일 여분의 자원이 있는지를 검사한다. 자원 예약 요청이 수락, 정책 제어 검사를 통과하면 RSVP 는 자원 예약 요청 정보를 트래픽 제어 인터페이스를 통해 패킷 분류자와 패킷 스케줄러에게 전달하여 사용자가 요구한 서비스 품질 보장이 가능하도록 한다.

RSVP 와 다른 구성 요소 사이의 인터페이스에는 RAPI (RSVP API), 트래픽 제어 인터페이스 (Traffic Control Interface), RSRR (Routing Support Resource Reservation) 이 있다. 이 중 RAPI 는 호스트에서 응용 프로그램과 RSVP 사이의 인터페이스로 라우터 구현에서는 고려되지 않는다. RSVP 자체는 라우팅 프로토콜이 아니기 때문에 RSVP 는 RSRR 을 통해

[+] 본 논문은 정보통신부의 '99 정보통신부의 대학기초과제와 삼성전자의 지원에 의해 수행되었음.

서 라우팅 데몬과 통신하게 된다. 예를 들어 RSVP는 RSRR을 통해 메시지 전달을 위한 라우팅 정보를 획득하게 된다. 그리고 트래픽 제어 인터페이스는 패킷 분류자, 패킷 스케줄러, 수락 제어로 구성된 트래픽 제어부와 RSVP 사이의 인터페이스로 자원 예약이 설립, 제거등의 시기에 이 인터페이스를 통하여 RSVP와 트래픽 제어부 사이의 통신이 이루어 진다. 본 논문에서는 예약된 서비스 품질을 실질적으로 보장하기 위한 라우터의 구현 요소인 트래픽 제어부의 패킷 분류자와 패킷 스케줄러 및 트래픽 제어 인터페이스를 구현하였다.

3. 트래픽 제어부의 구현 및 구현 구조

본 논문에서는 PC를 기반으로 라우터 기능을 FreeBSD 3.2 환경에서 구현하였다. 트래픽 제어부 구현을 위해 altq-1.2를 사용하였고 RSVP는 ISI의 rel 4.2a4 버전을 사용하였다 [6], [10]. 현재 rel 4.2a4 버전에서는 RSVP와 RAPI, RSRR 등이 구현되어 있고 트래픽 제어 인터페이스는 실질적인 트래픽 제어부와의 통신 없이 틀만 구현되어 있는 상황이다. 따라서 본 논문에서는 패킷 분류자와 패킷 스케줄러를 구현하고 RSVP와 트래픽 제어부와의 통신을 위해 트래픽 제어 인터페이스를 구현하여 라우터가 인터넷 통합 서비스를 지원하도록 구현 하였다.

(1) 패킷 분류자와 패킷 스케줄러의 구현

패킷 분류자를 위해 본 논문에서는 목적지 IP 주소, 프로토콜 ID, 목적지 포트 번호를 입력으로하는 해시 함수를 써서 적절한 플로우로의 매핑이 가능하도록 구현하였다.

인터넷 통합 서비스 모델의 패킷 스케줄러를 위해 WFQ (Weighted Fair Queueing) 을 기본으로 다양한 스케줄링 방식이 연구되었다 [3]-[5]. 하지만 패킷의 서비스 순서를 결정해야 하는 시간이 매우 짧은 고속망에서 WFQ와 같은 복잡한 스케줄링 방식의 적용은 어려운 점이 있다.

따라서 본 논문에서는 WFQ 보다 공평성은 저하되지만 그 복잡성을 줄이고 비교적 구현이 간단한 SCFQ (Self-Clocked Fair Queueing) 방식을 스케줄링 방식으로 구현하였다 [5]. 스케줄링 방식 구현은 ALTQ를 기반으로 구현하였으며 사용자는 기존 FIFO 스케줄링과 구현된 SCFQ 방식을 선택하여 사용할 수 있도록 구현하였다. 스케줄러의 구현 구조는 그림 2와 같다.

상위 계층에서 내려온 패킷은 각 세션에 대해 할당된 큐에 IP 주소와 포트 번호, 프로토콜 ID 정보를 바탕으로 패킷 분류자에 의해 분류되어 적절한 큐에 큐잉되게 된다. SCFQ 스케줄러는 큐잉된 패킷의 가상 종료 시간 값을 비교하여 가장 작은 가상 종료 시간 값을 가지는 패킷부터 서비스한다. SCFQ의 정의에 따르자면 입력되는 모든 패킷은 큐잉 되기 전에 해당 패킷의 가상 종료 시간이 마킹되어야 한다. 하지만 기존 네트웍 구조에서는 패킷의 가상 종료 시간을 붙일 수 있는 부분이 없을 뿐더러 붙일 수 있게 패킷의 형식을 수정할 수 도 없다. 그러므로 본 논문에서는 각각의 패킷이 아닌 각 세션에 대해 할당된 큐마다 가상 종료 시간을 관리하도록 구현하였다. 이것은 SCFQ의 정의에 따라 모든 패킷에 가상 종료 시간을 붙여 패킷의 서비스 순서를 결정하는 스케줄링 하는 방식과 각각의 큐의 가장 앞 패킷만의 가상 종료 시간 값 만을 관리하여 스케줄링 하는 방식이 결과적으로 동일한 패킷의 서비스 순서로 스케줄링 한다는 것

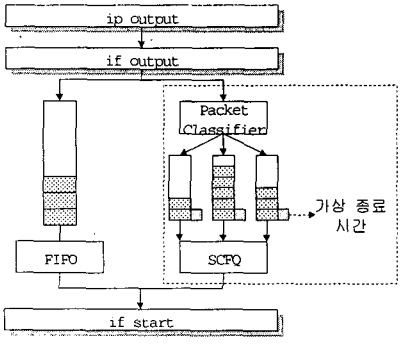


그림 2. 라우터의 스케줄러 구현 구조

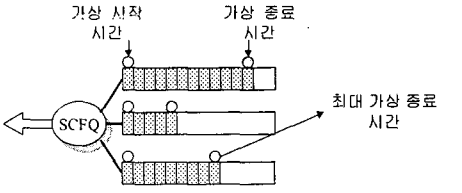


그림 3. 버퍼 폐기 방식

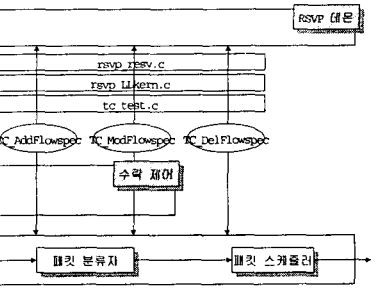


그림 4. 트래픽 제어 인터페이스를 통한 자원 예약

에 인한다. 따라서 구현된 SCFQ 스케줄러는 각 큐 혹은 각 세션을 단위로 가상 종료 시간 값을 관리하여 최소 가상 종료 시간 값을 가지는 큐의 맨 앞에 있는 패킷을 서비스 하도록 구현되었다. 이러한 구현 방법은 향후 다양한 버퍼 폐기 방식을 사용할 경우 보다 유연한 가상 시간 관리가 가능하다는 장점이 있다.

(2) 버퍼 폐기 방식의 구현

현재까지 구현된 혹은 연구된 패킷의 폐기 방식은 대부분 큐 길이를 기반으로 하는 폐기 방식이다. 즉 가장 큐 길이가 긴 큐 혹은 가중치에 따라서 가장 긴 큐 길이를 검색하여 그 큐의 패킷을 폐기하는 방식이었다. 하지만 WFQ와 같은 *rate* 기반의 스케줄링 방식을 사용할 경우 버퍼 할당에 있어서 max-min fair share를 이룩하려면 패킷을 폐기해야 할 때 최대 가상 종료 시간을 가지는 패킷을 폐기하여야 한다 [7]. 그리고 SCFQ 스케줄링 방식을 구현한 경우 세션에 할당된 큐 각각은 FIFO와 마찬가지로 동작한다. 즉, max-min fair share를 위해 버퍼를 폐기해야 할 상황에서 각 큐의 가장 마지막에 위치하는 패킷의 가상 종료 시간 값을 검색하여 가장 큰 가상 종료 시간 값을 가지는 큐의 패킷을 폐기하면 된다. 또한 큐 하나만을 생각하였을 경우 큐의 가장 마지막에 위치한 패킷의 가상 종료 시간은 다음 식과 같이 큐의 길이 및 큐의 가장 앞에 위치한 패킷의 가상 시작 시간 값 및 세션에 해당하는 가중치로 구할 수 있다. 따라서 본 논문에서는 패킷을 폐기할 큐를 선택하기 위해 그림 3과 같이 스케줄러가 각 큐의 맨 앞에 위치하는 패킷의 가상 종료 시간 뿐 아니라 가상 시작 시간 값을 관리하도록 구현하였다. 따라서 본 논문에서는 패킷의 가장 마지막 패킷의 가상 종료 시간을 검색하여 선택된 큐의 가장 앞에 위치하는 패킷을 폐기하도록 구현하였다.

$$F_k^{last} = S_k^{first} + \frac{L_k}{r_k}$$

F_k^{last} : 세션 k의 가장 마지막 패킷의 가상 종료 시간

S_k^{first} : 세션 k의 가장 앞에 있는 패킷의 가상 시작 시간

L_k : 세션 k의 큐 길이

r_k : 세션 k에 할당된 가중치

(3) 트래픽 제어 인터페이스의 구현

자원 예약은 하부 링크 계층 기술에 의존적이기 때문에 현재 RSVP의 트래픽 제어 인터페이스 부분은 링크 계층에 의존적이지 않은 핵심 계층

과 의존적인 링크 적용 계층 두 가지로 나누어 구현되었다. 한편 ISI의 RSVP는 실제로는 트래픽 제어 인터페이스를 통해서 패킷 분류자와 스케줄러와 통신이 가능하도록 구현되어 있지 않고 현재는 틀만이 구현 되어 있다. 따라서 본 논문에서는 RSVP를 사용하여 자원 예약이 가능하도록 SCFQ와 RSVP 사이의 트래픽 제어 인터페이스를 구현하였다. 즉, 그림 4와 같이 RSVP가 TC_AddFlowspec, TC_ModFlowspec, TC_DelFlowspec 등의 함수를 호출을 통해 커널과 통신하여 자원 예약의 설립, 변경 및 삭제가 가능하도록 구현하였다. 현재 자원 예약 방식으로는 Fixed-Filter 방식만을 구현하였으며, TC_AddFlowspec, TC_ModFlowspec 에서 호출되는 수락 제어 부분은 구현되어 있지 않다. 즉, 자원 예약 요청을 수락 제어 없이 모두 받아들이도록 구현되어 있다. 그림 4는 자원 예약 요청, 변경, 삭제를 위한 RSVP와 트래픽 제어부 사이의 트래픽 제어 인터페이스의 구현 구조를 나타내었다.

4. 성능 평가

본 논문에서는 기존 FIFO 기반의 라우터와 패킷 분류자, 스케줄러, 버퍼 관리 방식 및 트래픽 제어 인터페이스가 구현된 라우터의 성능 평가를 위해 그림 5와 같은 테스트 베드를 구성하여 구현 오버헤드 및 서비스 품질 보장에 대한 성능 평가를 수행한다. 각 링크는 10Mbps 이며 모든 PC에는 rel 4.2a4 버전의 RSVP를 사용하고 라우터에는 altq-1.2 버전을 사용하였다. 송신원은 이더넷 상에서의 충돌을 방지하기 위해서 라우터와 점대점 연결로 구성되어 있다.

트래픽 제어부를 구현함으로써 생기는 오버헤드를 측정하기 위해 네트워크 성능 평가 프로그램 중 하나인 netperf[8]를 사용하여 99%의 신뢰도, ±10%의 신뢰구간에서 성능 평가를 수행하였다. netperf를 사용함으로써 테스트 베드 상의 snd1 과 rcv1 사이에서 snd1 은 rcv1 에게 udp 패킷을 요청하고 이에 대한 응답을 snd1 으로부터 받게 된다. 요청하는 패킷과 응답 패킷의 크기에 따른 snd1 에서의 응답 요청 패킷의 송신 수와 스케줄링 적용으로 인한 각 패킷의 추가적인 지연 오버헤드는 표 1과 같다. 표에서 보는 바와 같이 패킷 분류자, 패킷 스케줄러를 구현함으로 패킷 당 12μ초 정도의 추가적인 지연이 발생한다.

RSVP를 사용한 대역 할당 성능 평가를 위해 트래픽 발생 프로그램중 하나인 MGEN[9]을 이용하였다. 구현된 스케줄링 기법의 성능을 평가하기 위해서 흐름제어가 사용되지 않는 UDP를 사용한다. 표 2에서는 각 플로우의 트래픽 전송 속도와 플로우 전송 시작 시간 및 할당된 대역폭을 나타내었다. 라우터와 수신원 사이에서 실효 대역폭은 8.5Mbps로 측정되었다. 송신원은 0초를 시작으로 10, 20, 30, 40초 마다 새로운 플로우가 생기게 되고 40초 뒤에 망의 폭주 상태가 일어나도록 각 플로우의 트래픽 발생 속도를 조절하였다. 그림 6은 위와 같은 상황에서 각 플로우의링크 사용 대역을 나타내었다. 그림에서 볼 수 있듯이 망의 폭주가 일어나면 자신의 대역보다 많은 트래픽을 전송하는 플로우는 스케줄러에 의해 전송이 할당된 대역만큼 트래픽 전송율이 제한 되는 것을 알 수 있다. 즉, 망의 폭주 시 기존의 FIFO 방식과는 달리 RSVP를 이용하여 각 플로우에 대한 대역이 할당된 SCFQ 방식은 스케줄링 알고리즘에 의한 트래픽의 보호가 이루어져 할당된 대역보다 전송 속도가 높은 플로우 1의 동작과는 상관 없이 각 플로우는 자신의 할당된 대역을 보장 받음을 알 수 있다. 또한 망이 폭주를 경험하지 않을 때는 FIFO와 마찬가지로 여분의 대역은 각 플로우가 공유하여 사용함을 알 수 있다.

5. 결론

본 논문에서는 인터넷 통합 서비스를 제공하기 위한 핵심 구현 요소인 패킷 분류자, 패킷 스케줄러 및 트래픽 제어 인터페이스를 라우터에 구현하여 RSVP를 이용한 자원 예약이 가능한 라우터를 구현하였다. 그리고 버퍼 관리 방식으로는 max-min fair share를 위해 최대 가상 종료 시간 패킷 폐기 방식을 라우터에 구현하였다. 또한 테스트 베드상에서 구현된 라우터의 성능 평가를 수행하여 인터넷 통합 서비스를 지원하는 라우터가 적은 오버헤드로 구현 가능하고 RSVP를 이용한 서비스 품질 보장이 가능함을 보였다.

참 고 문 헌

[1] R. Braden, D. Clark, S. Shenker, "Integrated Services in the Internet Architecture: An Overview," RFC 1633, June 1994.

[2] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin, "Resource ReSerVation Protocol (RSVP) - Verseion 1 Functional Specification," RFC 2205, September 1997.

[3] Abhay K. Parekh, Robert G. Gallager, "A Generalized Processor Sharing Approach to Flow Control in Integrated Service Networks : The Single-Node Case," IEEE/ACM transactions on networking, vol. 1, June 1993.

[4] Pawan Goyal, Harrick M. Vin, "Start-time Fair Queueing:A Scheduling Algorithm for Integrated Services Packet Switching Networks," IEEE/ACM Transactions on Networking,1997

[5] S. Jamaloddin Golestani, "A Self-Clocked Fair Queueing Scheme for Broadband Applications," In Proceeding of IEEE INFOCOM'94, Toronto, CA, June 1994.

[6] Kenjiro Cho, "A Framework for Alternate Queueing: Towards Traffic Management by PC-UNIX Based Routers," In Proceedings of *USENIX 1998* Annual Technical Conference, New Orleans LA, June 1998.

[7] S. Keshav,"An Engineering Approach to Computer Networking," Addison-Wesley, 1997.

[8] http://netperf.org/netperf/NetperfPage.html

[9] http://manimac.itd.nrl.navy.mil/MGEN

[10] http://www.isi.edu/rsvp

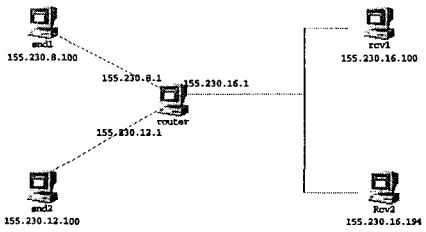


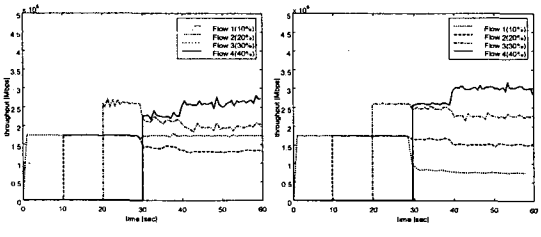
그림 5. 테스트 베드

표 1. SCFQ의 구현상 오버헤드

요청 패킷 크기 [byte]	응답패킷크기 [byte]	초당 송신 개수		Overhead [μsec]
		FIFO	SCFQ	
1	1	1639.15	1610.12	10.99
64	64	1076.31	1062.96	11.66
1024	64	307.98	306.87	11.74
8192	1024	83.18	83.08	14.47

표 2. 플로우별 전송 속도 및 할당 대역폭

flow	트래픽 전송 속도 [Mbps]	트래픽 발생 시간 [sec]	할당된 대역
1	1.7Mbps	0	10%
2	1.7 Mbps	10	20%
3	2.55	20	30%
4	2.55→3.4	30	40%



(a) FIFO (b) SCFQ

그림 6. 스케줄링 방식에 따른 각 플로우 별 전송율