리눅스 환경에서 SCTP와 TCP 프로토콜의 성능 비교

준회원 박 재 성*, 종신회원 고 석 주**°

Performance Comparison of SCTP and TCP over Linux Platform

Jae Sung Park* *Associate Member*, Seok Joo Koh**° *Lifelong Member*

요 약

본 논문에서는 다양한 망 환경에서 SCTP 및 TCP 프로토콜의 전송처리율(throughput) 성능을 비교 분석한다. 실험을 위해 리눅스 테스트베드를 구축하고 성능측정 변수로써 MSS(Maximum Segment Size), 전송지연, 패킷 손실률을 고려하였다. 또한, SCTP 세션의 스트림(stream) 수가 성능에 미치는 영향을 분석하였다. 실험 결과, 동일한 망 환경에서 SCTP는 TCP에 비해 20%~50% 정도의 높은 처리율을 제공하는데 이는 SCTP의 고유특성인 청크번들링(chunk bundling), 2 MTU로 시작하는 혼잡윈도우, SACK 기반 오류제어 등에서 기인한다. 한편, 패킷 손실이 존재하는 망에서 SCTP는 멀티스트리밍(multi-streaming) 전송을 통해 HoLB(Head-of-Line Blocking) 현상을 효과적으로 방지할 수 있음을 확인하였다.

Key Words : TCP, SCTP, Performance Comparison

ABSTRACT

This paper compares throughput performance of TCP and SCTP in a variety of network environments. For experiments, we construct a Linux-based testbed and consider a set of performance metrics such as MSS(Maximum Segment Size), transmission delay, and packet loss rate. In addition, we analyze the effect of SCTP multi-streaming on throughput. From the experimental results, we can see that SCTP provides throughput gain of approximately 20%~50% over TCP. This performance gain comes from the distinctive features of SCTP such as chunk bundling, initial congestion window of 2 MTU and SACK(Selective ACK) based error control. In the lossy networks, we can see that SCTP multi-streaming transmissions can effectively overcome the so-called HoLB(Head-of-Line Blocking) phenomenon of TCP.

Ⅰ. 서 론

SCTP(Stream Control Transmission Protocol)는 TCP 및 UDP에 이은 세 번째 수송계층 프로토콜로서, TCP와 유사한 제어 메커니즘을 제공하는 프로토콜이다[1]. 패킷 길이 관점에서 TCP 패킷의 기본 헤더는 20바이트인 반면에 SCTP의 기본 헤더는 그림 1처럼 28바이트로 구성되어, TCP에 비해 8바이트의 오버헤드를 가진다. SCTP는 TCP와 유사한 오류 제어, 흐름 제어 및 혼잡제어 방식을 사용한다. 다만, TCP의 경우 바이트 기반의 슬라이딩 윈도우 방식을 사용하는데 비하여, SCTP는 메시지 기반의 슬라이딩 윈도우 방식을 사용한다. 또한, TCP의 혼잡 제어 방식에서는 초기 혼잡 윈도우의 크기를 1 MTU로 설정하는데 반해 SCTP는 2 MTU로 설정한다. 게다가, SCTP는 SACK(Selective ACK)

※ 본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음. (IITA-2008-C1090-0804-0004)
* 경북대학교 전자전기컴퓨터학부 통신프로토콜연구실(knucsid@gmail.com)
** 경북대학교 전자전기컴퓨터학부 부교수(sjkoh@knu.ac.kr)(° : 교신저자)
논문번호 : KICS2007-09-405, 접수일자 : 2007년 9월 11일, 최종논문접수일자: 2008년 7월 30일

Source Port					Destination Port
Verification Tag					
Checksum					
Type	Reserved	U	B	E	Length
Transmission Sequence Number					
Stream Identifier					Stream Sequence Number
Payload Protocol Identifier					

그림 1. SCTP 헤더구조

기반의 오류제어 방식을 사용한다.

한편, SCTP는 TCP와는 달리 멀티스트리밍(multi-streaming) 및 멀티홈잉(multi-homing) 기능을 지원한다. 멀티스트리밍 기능은 하나의 연결에 여러 개의 스트림을 동시에 독립적으로 전송할 수 있는 기능이다. 이를 위해 SCTP는 각 스트림마다 Stream ID를 부여하고 다른 스트림과 독립적으로 SSN(Stream Sequence Number)를 할당한다. 멀티스트리밍 기능을 통해 한 스트림의 데이터 전송이 손실 되더라도 다른 스트림에서는 정상적으로 데이터 전송이 이루어지기 때문에 TCP의 HoLB(Head of Line Blocking) 문제를 방지할 수 있다. 멀티홈잉 기능은 여러 개의 네트워크 인터페이스가 있는 상황에서 여러 개의 IP 주소를 사용할 수 있다.

본 논문에서는 리눅스 환경에서 SCTP와 TCP의 전송성능을 비교 분석하고자 한다. 특히, 일반적으로 SCTP가 TCP에 비해 8바이트의 오버헤드를 가지는데, 이로 인한 전송성능 효과를 분석한다. 또한, SCTP 멀티스트리밍 기능에서 스트림 개수가 성능에 미치는 영향을 알아볼 것이다.

본 논문은 다음과 같이 구성된다. Ⅱ절에서는 SCTP 성능분석에 대한 기존 관련 연구에 대해서 기술하고, Ⅲ절에서는 성능 분석을 위한 실험 환경 및 시나리오를 제시한다. Ⅳ절에서는 실험결과를 토대로 두 프로토콜의 성능을 비교 분석한다. 마지막으로 Ⅴ절에서 결론을 맺는다.

Ⅱ. 관련 연구

SCTP 프로토콜이 표준화 이후로 SCTP에 대한 성능비교 분석 연구가 많이 수행되었다. 동일한 환경에서의 SCTP와 TCP의 성능비교에 대한 연구와 함께, SCTP의 고유 특성인 멀티스트리밍과 멀티홈잉을 이용하여 TCP보다 성능을 향상시키는 연구도 많이 이루어졌다.

먼저 [2] 연구에서는 동일한 대역폭을 갖는 망 환경에서 TCP와 SCTP의 성능을 비교하였다. 해당 연구에 의하면, 제한된 대역폭에서 사용자 데이터 크기가 상대적으로 큰 경우에는 SCTP가 높은 처리율을 보인다. 손실이 많이 발생하는 링크에서 SCTP의 성능을 향상시키는 연구에서는[3], SCTP 프로토콜에 혼잡 상황을 명시적으로 알리는 ECN(Explicit Congestion Notification) 기법의 적용을 제안하였다. 해당 연구에서는 혼잡 윈도우의 최적화를 위해 혼잡 상황에서의 손실과 비혼잡 상황에서의 손실의 차별화를 통해 성능을 개선하는 기법을 제안하였다.

또한, [4] 연구에서는 리눅스 플랫폼에서 사용자 데이터의 크기에 따른 성능 비교와 TCP와 SCTP의 트래픽경쟁 비교 및 멀티홈잉 특성에 의한 성능 비교분석을 수행하였다. 연구결과, 사용자의 입력 데이터의 크기가 2048 바이트 이하일 경우에는 TCP의 처리율이 SCTP에 비해 높게 측정되고 사용자의 입력 데이터의 크기가 8196 바이트 이상일 경우에는SCTP의 처리율이 TCP에 비해 높게 측정된다. SCTP와 TCP의 트래픽이 동일 링크에 공존하는 경우에는 거의 공정하게 경쟁하는 것을 알 수 있다.
SCTP의 멀티스트리밍 특성을 이용한 멀티미디어 품질 향상을 제안하는 연구도 이루어졌다[5]. 기존의 멀티미디어 경우 음성, 영상, 텍스트를 각각 다른 채널로 전송한다. 이 논문에서는 SCTP의 멀티스트리밍 특성을 이용하여 음성, 영상, 텍스트 별로 다른 스트림을 사용하는 것이 바람직하다고 제안하고 있다. MANET 환경에서 SCTP와 TCP의 성능을 비교 분석한 연구도 이루어졌다[6]. 이 연구에서 SCTP와 TCP는 MANET 환경에서 유사한 동작을 수행하지만, 대부분의 시나리오에서 TCP가 SCTP보다 성능이 높게 측정되었다.

한편, 멀티홈잉 특성을 가지는 SCTP와 TCP의 성능을 비교한 연구도 이루어졌다[7]. 해당 연구에서는 멀티홈잉을 지원하는 SCTP와 TCP Reno, TCP SACK 방식의 성능을 시뮬레이션을 통하여 비교하였다. 실험결과, 멀티홈잉 SCTP의 재전송 정책이 우선경로(primary path)와 대체경로(alternate path) 간의 특성에 따라 성능이 달라짐을 보여주었다. 또한, [8] 연구에서는 SCTP의 멀티홈잉 기법을 이용하여 TCP보다 향상된 성능을 제공하고, 다중 손실이 존재하는 네트워크에서 다양한 SCTP 혼잡 제어 방식을 제안하였다.

상기와 같이 기존 연구들은 대부분 SCTP의 고유 특성인 멀티스트리밍과 멀티홈잉 관점에서 SCTP의 전송성능을 분석하는 데에 중점을 두고 있다. 본 논

문에서는 TCP와 비교하여 8 바이트의 패킷 오버헤드를 가지는 SCTP의 성능이 다양한 네트워크 및 프로토콜 환경변수에 따라 어떻게 영향을 받는지 분석한다. 또한, 멀티스트리밍 관점에서 스트림 개수와 전송 성능과의 상관관계를 분석하고자 한다.

Ⅲ. 실험환경 및 성능분석 모델

3.1 실험환경

본 논문에서는 데이터 전송 시에 SCTP와 TCP의 처리율 성능을 비교하기 위해서 리눅스 기반의 테스트베드를 구축하였다. SCTP의 실험을 위해 커널 버전이2.6인 리눅스에서 LK-SCTP[9]를 설치하였고, 전송지연 및 패킷 손실 등의 다양한 네트워크의 성능 요인을 실험하기 위해 NISTNET 네트워크 에뮬레이터를 사용하였다[10]. 또한, 데이터 전송에 대한 처리율 성능을 측정하기 위해 패킷 분석기인 Ethereal[11]을 이용하였다.

그림 2는 실험에 사용한 테스트베드 구성도이다. PC 라우터의 역할을 수행하는 컴퓨터에 NISTNET을 설치하여 다양한 네트워크 환경변수를 조절한다. 성능 비교를 위해 조절되는 성능 요인을 제외한 다른 실험조건(예: 커널버퍼 크기, 소켓옵션)들은 모두 동일하게 설정하였다. 모든 실험에서 링크 계층의 MTU(Maximum Transmission Unit)의 크기는 1500 바이트로 고정된다.

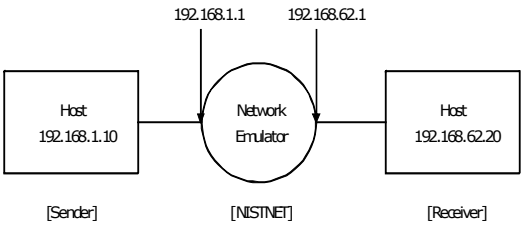


그림 2. 테스트베드 구성도

3.2 성능분석 시나리오

SCTP와 TCP의 전송 성능에 영향을 주는 성능요인(factor) 변수는 크게 호스트에서 설정하는 '호스트변수'와 네트워크 환경에 의해 결정되는 '네트워크 변수'로 구분할 수 있다. 본 실험에서는 호스트 변수로서, MSS와 SCTP의 스트림 개수를 고려하였다. 또한, 네트워크 변수로는 '패킷 손실률(packet loss rate)'과 '전송지연(transmission delay)'을 고려하였다.

성능분석 실험을 위해 다음 4가지의 시나리오를 적용한다. 각 시나리오에 대해 5번의 실험을 수행하고, 실험에서 추출된 평균값으로 성능을 비교하였다.

3.2.1 MSS에 따른 성능비교

일반적으로 패킷의 수가 많을수록 SCTP의 패킷 오버헤드는 증가할 것이다. 실제로 패킷 오버헤드가 성능에 미치는 영향을 분석하기 위해 MSS 값을 변화시키며 실험을 수행한다. 실험에 적용되는 MSS를 256 바이트, 512 바이트 및 1024 바이트로 변경하면서 각각에 대한 성능을 측정한다. 해당 실험에서 전송에 사용되는 파일의 크기는 10MB이고 전송지연은 100ms, 패킷 손실률은 0%로 고정된다.

3.2.2 전송지연 시간에 따른 성능비교

패킷의 오버헤드가 발생하게 되면 전송해야 하는 패킷량이 많아지게 된다. 패킷량이 많아지는 경우 전송지연 시간이 성능에 어떤 영향을 주는지 알아보기 위해, 전송지연 시간을 100ms, 200ms, 500ms로 변경하여 실험을 수행하였다. 해당 실험에서 전송에 사용되는 파일의 크기는 10MB이고 MSS는 1024바이트, 패킷 손실률은 0%로 고정된다.

3.2.3 패킷 손실률에 따른 성능 비교

패킷이 손실되면 재전송 패킷이 자주 발생하게 되고, 이 경우 패킷 오버헤드가 있는 SCTP의 성능이 더욱 저하될 것이다. 이에 대한 성능 변화를 분석하기 위해 패킷 손실률을 1%, 2%, 3%, 4%, 5%로 변화시키며 전송성능을 측정한다. 전송에 사용되는 파일의 크기는 100MB이고 MSS는 1024바이트, 전송 지연 시간은 각 0ms로 고정된다.

3.2.4 SCTP 스트림 수에 따른 성능 비교

SCTP의 멀티스트리밍 특성에 따라 스트림수가 전송성능에 주는 영향을 분석하기 위하여, SCTP 스트림의 수를 1개에서 5개까지 변화시키며 성능을 측정한다. 전송에 사용되는 파일의 크기는 100MB이고 패킷 손실률이 0%, 5%, 10%일 때 각각 성능을 측정하고, 또한 전송 지연 시간이 0ms, 100ms, 200ms일 때 성능을 각각 측정한다.

MSS와 전송지연에 따른 성능 비교의 경우 10MB의 파일을 전송하였으나, 패킷 손실률과 SCTP 스트림 수에 따른 성능 비교의 경우 좀 더 오랜 기간 관측하기 위해 100MB의 파일 전송을 수행하였다.

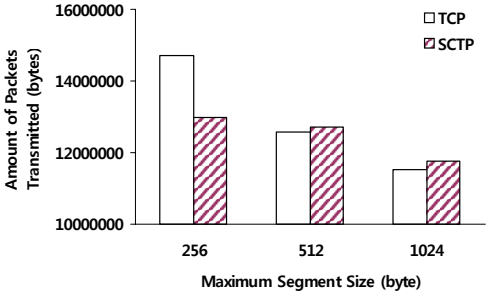
Ⅳ. 실험 결과 분석

4.1 MSS에 따른 성능비교

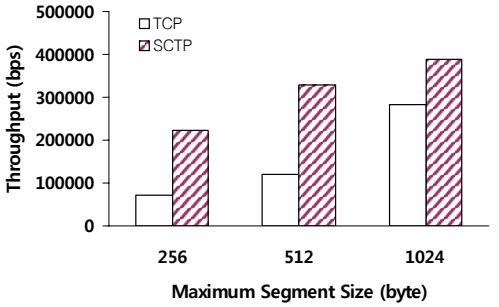
그림 3은 다양한 MSS 크기에 따른 TCP와 SCTP의 성능실험 결과를 보여준다. 일반적으로 MSS가 커질수록 패킷에서 사용자 데이터 페이로드(payload)가 차지하는 비율이 커지므로, 패킷 헤더 크기(TCP: 20 바이트, SCTP: 28 바이트)로 인한 오버헤드 영향은 감소된다. 따라서, MSS가 커질수록 전송된 패킷의 전체 개수가감소하고 이에 비례하여 패킷량(byte)도 감소한다.

그림3(a)의 전송된 패킷량을 비교해 보면, TCP의 경우 MSS가 증가함에 따라서 패킷량이 급속히 감소하는 반면에, SCTP의 경우 비교적 완만하게 감소한다. 이는 TCP의 경우, MSS가 TCP 패킷 크기를 결정하는 데 비하여, SCTP의 경우 데이터 청크의 크기만을 제한하고, SCTP 패킷 전체 크기에는 영향을 주지 않기 때문이다. 즉, SCTP의 경우 하나의 패킷에 소규모 MSS 크기의 데이터 청크를 2개 이상 번들링(bundling)하여 전송할 수 있다. 예를 들어 MTU가 1000바이트이고 MSS가 250 바이트라고 하면 TCP는 4개의 패킷을 보내게 되고 SCTP는 4개의 데이터 청크를 하나의 패킷으로 전송하게 된다. 이 경우, TCP 헤더의 크기는 20*4=80 바이트가 되고 SCTP 헤더의 크기는 공용헤더 16 바이트에 메시지 청크 헤더 12*4 바이트를 합하여 64 바이트가 된다. 따라서, 그림 3(a)에서처럼 MSS가 256 바이트인 경우에는 SCTP의 패킷헤더 오버헤드 효과가 사라지고, 오히려 TCP 보다 더 적은 패킷 개수 및 패킷량을 보이게 된다. 실험결과에서, MSS가 커질수록 이러한 SCTP 번들링 효과가 감소되면서, MSS가 512 바이트인 경우에는 SCTP의 패킷량이 TCP에 비해 13K 바이트 정도 많고 MSS가 1024 바이트인 경우에는 SCTP의 패킷량이 22K 바이트 정도 많은 것을 확인할 수 있다.

한편, 그림3(b)의 처리율비교에서 보면, 모든 경우에 대하여 SCTP가 더 나은 성능을 보인다. 이는 기본적으로 SCTP 혼잡제어서는 초기 혼잡윈도우(congestion window) 크기가 2 MTU에서 시작하는 반면에, TCP에서는 1 MTU에서 시작하기 때문이다. 즉, 초기 혼잡윈도우가 더 크게 설정된 SCTP에서는 패킷 손실이 없는 상황에서 TCP에 비해 혼잡윈도우가 더욱 급격히(exponentially) 증가하여 좋은 처리율성능을 보이는 것으로 볼 수 있다. 특히, MSS가 작은 경우에는 이러한 혼잡윈도우 효과에다



(a) 패킷량 비교



(b) 처리율 비교

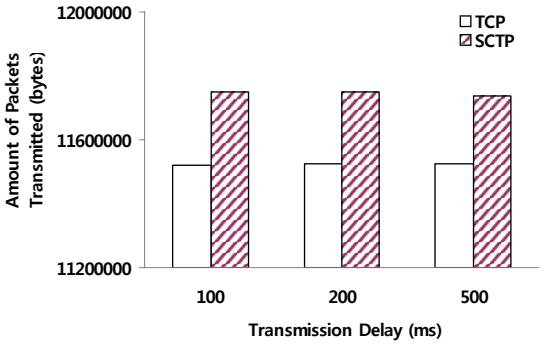
그림 3. MSS에 따른 성능비교

가, SCTP 번들링 효과가 함께 적용되어 TCP와의 성능 차이가 더 커지는 것으로 분석된다. 실험결과에서, MSS가 작은 경우에 SCTP는 TCP에 비하여 약 20~30% 정도의 처리율 성능을 개선하고 있음을 확인할 수 있다.

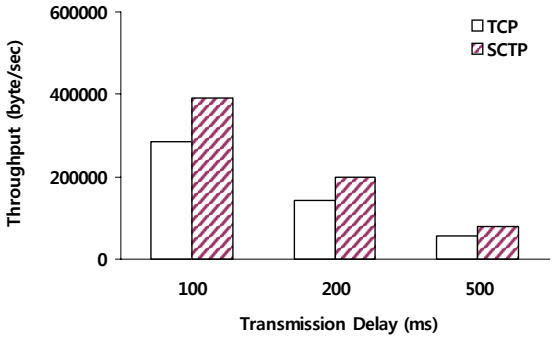
4.2 전송 지연에 따른 성능비교

그림 4는 다양한 전송지연 시간에 따른 실험 결과이다. 관련 실험에서 MSS는 모두 1024 바이트를 사용하였다. 먼저 그림4(a)의 패킷량 비교에 따르면, SCTP가 TCP보다 지연 시간과는 관계없이 200KB 정도 많은 패킷량을 생성한다. 이러한 결과는 그림 3(a)에서처럼 SCTP 패킷헤더에 대한 오버헤드 때문이다. 반면에 그림4(b)의 처리율 성능비교를 보면 SCTP가 TCP에 비해 향상된 성능을 보이고 있다. 이는 이전에 기술한 것처럼, SCTP의 초기 혼잡윈도우가 더 크게 설정되고, 그 만큼 더 많은 패킷을 한꺼번에 전송할 수 있기 때문이다.

한편, 그림 4(b)에서 전송지연 시간이 커질수록 성능 차이는 점진적으로 줄어들고 있음을 확인할 수 있다. 이러한 현상은 전송지연 시간이 일정 수준 이상으로 커지게 되면, SCTP의 '재전송 타이머'가 만료되어 패킷 재전송이 이루어지고, 이와 함께 SCTP

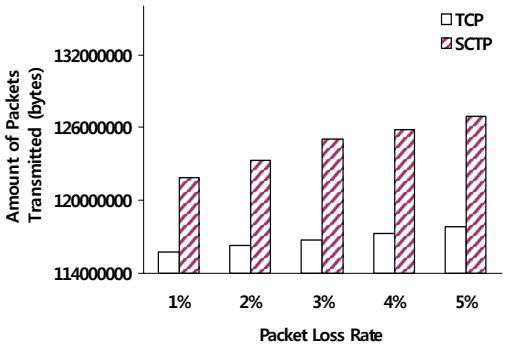


(a) 패킷량 비교

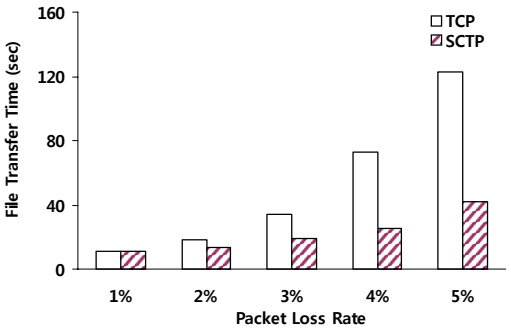


(b) 처리율 비교

그림 4. 전송지연에 따른 성능비교



(a) 패킷량 비교



(b) 전송시간 비교

그림 5. 패킷 손실률에 따른 성능비교

혼잡윈도우가 1 MTU로 감소하여, 이후부터는 더이상 혼잡제어 방식으로 인한 성능이득을 얻을 수 없기 때문이다. 실험결과, SCTP는 TCP에 비하여 최대 30%까지 처리율 성능을 개선하는 것으로 확인되었다.

4.3 패킷 손실률에 따른 성능비교

그림 5는 다양한 패킷 손실률에 따른 성능 비교 실험 결과이다. 실험에서 MSS는 1024 바이트로 고정되고, 그림 5(b)에서 처리율대신에 데이터 파일전송이 완료되기까지 소요된 시간을 측정하였다. 먼저 그림 5(a)의 패킷량 비교에서, 패킷 오버헤드로 인해 SCTP의 패킷량이 TCP에 비해 더 많음을 알 수 있다. 또한 패킷 손실률이 증가할수록 재전송 패킷의 증가로 인해 패킷량이 7%~10%까지 증가한다. 하지만, 그림 5(b)의 전송시간 성능을 보면, 전송된 패킷량이 SCTP가 TCP에 비해 많음에도 불구하고 SCTP가 더 빠른 시간에 파일 전송을 완료하고 있다. 즉, 전송 처리율이 높아진다. 특히 패킷 손실률이 증가할수록 두 프로토콜의 성능차이가 벌어짐을 볼 수 있다. 실험결과, 패킷 손실이 존재하는 경우에는 SCTP는 TCP에 비하여 최대 50%까지 처리율 성능이 개선됨을 알 수 있다.

패킷 손실이 존재하는 경우에 SCTP가 TCP에 비하여 성능이득을 보이는 이유는, SCTP의 SACK (Selective ACK) 메커니즘에 의한 것으로 분석된다. 즉, SCTP의 경우 SACK을 통하여 'Gap Block' 형태로 손실된 패킷과 수신된 패킷 정보를 송신자에게 통보할 수 있으며, 이로 인해 불필요한 패킷 재전송을 방지하고 손실된 패킷만을 재전송하여 보다 빠른 시간에 파일전송을 완료함으로써 전송 처리율을 개선시킬 수 있다. SACK 매커니즘으로 인한 TCP와의 성능차이는 패킷 손실률이 증가할수록 더욱 커진다.

4.4 SCTP 스트림 수에 따른 성능비교

그림 6은 패킷 손실에 존재하는 경우, SCTP의 스트림 개수에 따른 성능 비교 결과이다. 상기 실험은 SCTP 멀티스트리밍에서 스트림 개수가 전송 성능에 주는 영향을 분석하기 위한 것이다. 먼저 그림 6(a)에서 알 수 있듯이, 패킷 손실이 존재하는 경우에도 스트림 수는 패킷량에 영향을 주지 않는다.

한편, 그림 6(b), 6(c) 및 6(d)는 패킷 손실률에 따른 스트림수와 전송 성능과의 관계를 보여준다. 먼저 그림 6(b)에서, 손실이 없는 경우에는 스트림 수가 성능에 영향을 주지 않는다. 반면에 그림 6(c)와 6(d)에서 알 수 있듯이, 손실이 존재하는 경우에는 스트림 개수가 성능에 영향을 준다. 본 실험 환경에서는 스트림 개수가 2개일 때에 가장 좋은 성능을 보인다. 이러한 결과는 실험환경이 바뀌는 경우 다른 결과를 얻을 수도 있을 것이나, 주목할 만한 점은 패킷 손실이 존재하는 망 환경에서 전송 성능을 최대화하는 최적의 스트림 개수가 존재한다는 점이다. 멀티스트림 전송이 단일 스트림 전송보다 전송 성능이 좋은 이유는, HoLB 현상을 피할 수 있기 때문이다. 즉, 패킷 손실로 인해 특정 스트림 데이터의 전송이 지연되더라고, 다른 스트림의 데이터들은 이와 무관하게 전송될 수 있기 때문이다. 이로 인해 다중 스트림의 전송 성능이 좋아질 수 있다. 하지만, 일정 수준 이상으로 스트림 개수 증가하면, 더 이상 성능은 증가하지 않는다. 그 이유는 스트림 개수가 증가할수록 HoLB 현상을 피하여 전송을 하더라도 패킷 순서 재배치에 의한 오버헤드가 발생하기 때문이다.

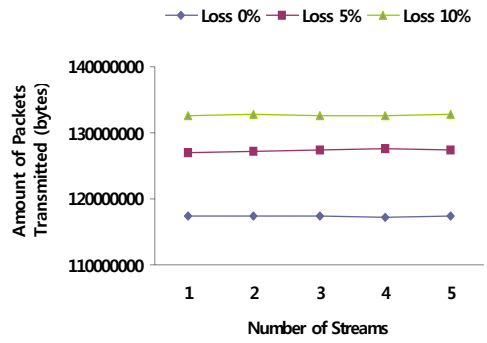
한편, 그림 7은 네트워크 전송지연 시간의 변화에 따른 멀티스트림 성능 효과를 보여준다. 그림에서 멀티스트림 전송의 패킷량이나 처리율 성능은 전송지연 시간과는 관계가 없음을 알 수 있다. 즉, HoLB 현상은 패킷 손실에는 영향을 받지만, 전송지연에는 별로 영향을 받지 않음을 보여준다.

V. 결 론

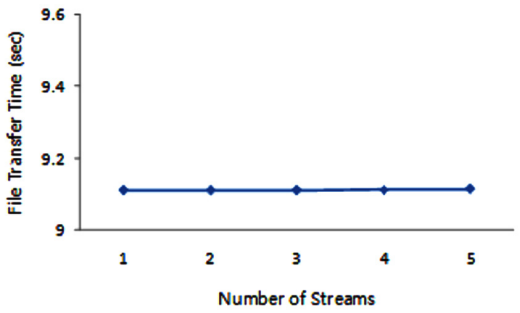
본 논문에서는 패킷량 및 전송처리율 관점에서 SCTP와 TCP의 성능을 비교 분석하였다. 데이터 패킷의 구조상 SCTP는 TCP에 비하여 8바이트의 오버헤드를 갖는다. 이러한 패킷 오버헤드가 데이터 전송 성능에 어떠한 영향을 주는지를 분석하기 위하여 MSS, 전송지연, 패킷 손실률 등의 다양한 성능 요인에 따라 두 프로토콜을 비교 분석하였다.

실험 결과, MSS 크기가 작은 경우에는 SCTP 데이터 청크의 번들링 효과로 인하여, 전송되는 패킷량이 SCTP가 20% 정도 적게 측정되었다. MSS가 큰 경우에는 SCTP 패킷 전송량이 TCP에 비하여 많지만, SCTP의 초기 혼잡윈도우가 2 MTU로 시작하는 특성으로 인해 TCP보다SCTP가 20%~30% 정도 좋은 처리율성능을 보인다.

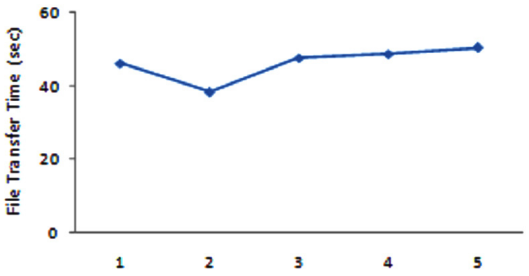
다양한 전송지연 시간에 대한 실험에서도 SCTP가 패킷량은 많지만 혼잡제어 방식의 특성으로 인해 좋은 처리율 성능을 보였다. 또한, 패킷 손실이



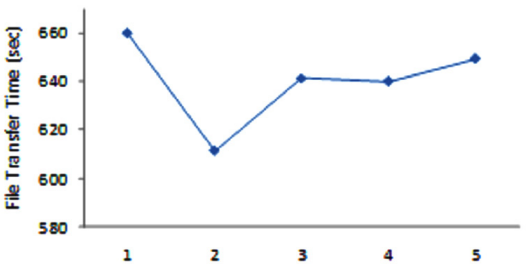
(a) 패킷량



(b) 전송시간 (패킷 손실률 = 0%)

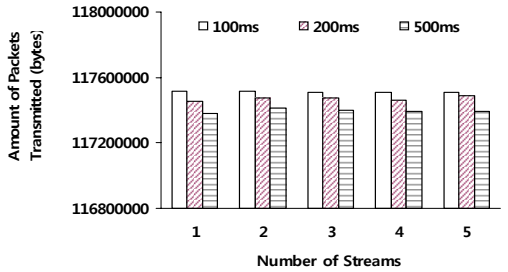


(c) 전송시간 (패킷 손실률 = 5%)

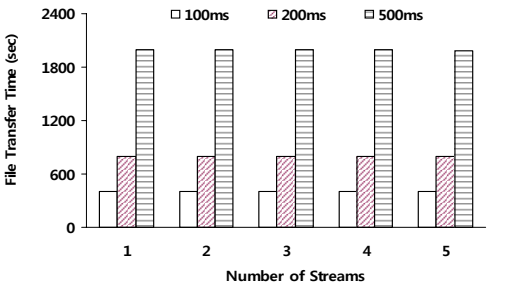


(d) 전송시간 (패킷 손실률 = 10%)

그림 6. 패킷손실과 멀티스트리밍 성능비교



(a) 패킷량 비교



(b) 전송 시간 비교

그림 7. 전송지연과 멀티스트리밍 성능 비교

존재하는 네트워크 환경에서는 패킷 손실 및 재전송으로 인해 SCTP 초기 혼잡윈도우 크기로 인한 장점은 사라지지만, Gap Block을 사용하는 SCTP SACK 특성으로 인해 재전송회수를 줄임으로써 SCTP가 TCP에 비하여 최대 50%까지 처리율 성능을 개선할 수 있다. 이러한 성능차이는 패킷 손실률이 커질수록 증가하는 경향을 보였다.

한편, SCTP 멀티스트리밍 기능이 성능에 주는 영향을 분석한 실험에서는, 스트림 개수가 2개일 경우 전송 성능이 가장 좋게 측정되었다. 이를 통해 패킷 손실이 존재하는 네트워크에서 전송 성능을 극대화하는 SCTP 스트림 개수가 존재함을 확인하였고, 정확한 스트림 개수는 망 환경에 따라 다를 수 있다. 이러한 성능 이득은 SCTP 멀티스트리밍 기능이 HoLB 현상을 효과적으로 예방하기 때문이다. 반면에, 전송지연은 멀티스트리밍 성능에 큰 영향을 주지 않는다.

결론적으로, SCTP는 TCP에 비하여 8바이트의 패킷 오버헤드를 가지지만, SCTP의 데이터 청크 번들링, 2 MTU로 시작하는 초기 혼잡윈도우, 그리로 SACK을 사용하는 오류제어 방법, 또한 HoLB 현상을 방지하는 멀티스트리밍 등의 특성으로 인하여 TCP보다 더 좋은 전송 성능을 제공할 수 있음을 알 수 있다. 참고로, 본 논문의 실험결과는 기본적인 TCP 프로토콜에 대한 성능 실험 결과이며, TCP-SACK, TCP-New Reno 등의 확장기법은 적용하지 않았다. TCP 확장 기법과 SCTP와의 성능 비교는 향후 연구과제이다.

참 고 문 헌

〔1〕 Stewart R., *et al.*, Stream Control Transmission Protocol, IETF RFC 2960, October, 2000.

〔2〕 A. Jungmajer, *et al,* “Performance Evaluation for the Stream Control Transmission Protocol”, *Proceeding of the Joint ATM Workshop 2000*, pp.141-148, June, 2000.

〔3〕 Guanhua Ye, *et al.*, “Improving Stream Control Transmission Protocol Performance Over Lossy Links”, *IEEE Journal on Selected Areas in Communications (JSAC)*, Vol.22, Issue 4, pp.727-736, May, 2004.

〔4〕 Jong-Shik Ha, *et al.,* “Performance Comparison of SCTP and TCP over Linux Platform”, *LNCS 3645*, pp.396-404, August, 2005.

〔5〕 민경주 외, “SCTP를 이용한 멀티미디어 품질 향상 방법”, *한국정보과학회 추계학술대회발표논문집* 제30권 제2호, pp.280-282, 2003년 10월.

〔6〕 A. Kumar, L. Jacob, A. L. Ananda, “SCTP vs TCP: Performance Comparison in MANETs”, *Proceeding of the 29th IEEE LCN’04*, pp.431-432, November, 2004.

〔7〕 송정화 외, “SCTP의 멀티홈잉 특성에 대한 성능 평가”, *한국정보처리학회논문지*, 제11-C권 제2호, pp.245-252, 2004년 4월.

〔8〕 A. Caro, *et al.*, “SCTP and TCP Variants: Congestion Control Under Multiple Losses”, Technical Report TR2003-04, *Dept of Computer and Information Science*, University of Delaware, February, 2003.

〔9〕 Linux Kernel SCTP, Available from http://lksctp.sourceforge.net.

〔10〕 NISTNET Network Emulator, Available from http://www-x.antd.nist.gov/nistnet/

〔11〕 Ethereal Packet Analyzer, Available from http://www.ethereal.com.

박 재 성 (Jae Sung Park) 준회원

2006년 2월 경북대학교 컴퓨터과학과 이학사

2006년 3월~현재 경북대학교 컴퓨터과학과 석사과정

<관심분야> 차세대인터넷, NGN 모바일 멀티캐스트, SCTP

고 석 주 (Seok Joo Koh) 종신회원

1992년 2월 KAIST 경영과학과 공학사

1994년 2월 KAIST 경영과학과 공학석사

1998년 8월 KAIST 산업공학과 공학박사

1998년 8월~2004년 2월 ETRI 표준연구센터 선임연구원

2004년 3월~현재 경북대학교 전자전기컴퓨터학부 부교수

2000년~현재 ITU-T SG13, SG17, SG19 및 JTC1/SC6 Editor

<관심분야> 이동성 제어, 미래인터넷, NGN, SCTP