무선 인터넷 환경에서 HTTP over SCTP의 평균 응답 시간 추정

Mean Response Delay Estimation for HTTP over SCTP in Wireless Internet

이용진
한국교원대학교 기술교육과

Yong-Jin Lee(lyj@knue.ac.kr)

요약

현재 인터넷에서는 객체를 전송하기 위해 HTTP over TCP가 사용된다. SCTP는 TCP의 문제점을 보완하기 위해 제안된 프로토콜로 개별적인 스트림을 이용하여 데이터를 전송하므로 웹 객체의 평균 응답 시간을 감소시킬 수 있다. 본 논문에서는 HTTP over SCTP의 평균 응답 시간을 추정하는 해석적 모델을 개발함으로써 TCP에 대한 SCTP의 효과를 추정하고자 한다. TCP 지연을 구하기 위한 기존의 해석적 모델이 주로 유선 환경을 가정한데 비해, 본 연구에서 제안하는 모델은 다중 패킷 손실이 발생하고, 빠른 재전송이 가능하지 않은 무선 환경을 가정한다. 추정된 평균 응답 시간은 실제 웹 응용을 개발할 때, 종단 사용자의 서비스 품질을 만족시키기 위한 사전 벤치마크로 사용될 수 있다. 추정 모델의 타당성을 검증하기 위해 테스트베드 상에서 성능 실험을 수행한 결과, 모델과 실험결과의 차이가 평균 6 % 이내로 매우 작으며, 동시에 HTTP over SCTP의 평균 전송 시간이 HTTP over TCP의 평균 전송 시간보다 작은 것으로 확인되었다.

■ 중심어 : | 무선인터넷 | SCTP | 혼잡제어 | 응답시간 |

Abstract

Hyper text transfer protocol (HTTP) over transmission control protocol (TCP) is currently used to transfer objects in the Internet. Stream control transmission protocol (SCTP), an alternative to TCP, which allows for independent delivery among streams, and can thus reduce the mean response delay of web object. We present an analytical model to find the mean response delay for HTTP over SCTP, therefore, estimate the effectiveness of SCTP over TCP. Typical TCP delay models assume the wired environment. On the contrary, the proposed model in this paper assumes the multiple packet losses and wireless environment where fast retransmission is not possible due to small window. The estimated mean response time can be used the benchmark to meet quality of service (QoS) at end-user. We validate the accuracy of our model using experiments. It is shown that the differences between the results from model and those from experimental are very small below 6 % on average. We also find that the mean response delay for HTTP over SCTP is less than that for HTTP over TCP.

■ keyword : | Wireless Internet | SCTP | Congestion Control | Response Time |

* 본 논문은 2006년 정부(교육인적자원부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구입니다.

접수번호 : #080401-002
접수일자 : 2008년04월01일
심사완료일 : 2008년 04월 18일
교신저자 : 이용진, e-mail : lyj@knue.ac.kr

I. 서 론

HTTP는 월드 와이드 웹 시스템이 인터넷에서 객체를 전송하는 데 사용하는 프로토콜이다. HTTP는 연결 지향 프로토콜이므로 트랜스포트 계층에서 TCP를 사용한다. TCP 연결은 단일 스트림(single stream)을 사용하여 신뢰적인 순서 전송을 실현하지만 사용자가 느끼는 지연 시간을 증가시킨다. 이와 대조적으로 HTTP는 지속 연결(persistent connection)과 파이프라이닝(pipelining)을 사용하여 웹 객체의 전송 시간을 줄이는데 초점을 맞춘다. 이러한 두 개의 상반된 프로토콜의 입장- HTTP의 응답 지연 최소화에 대한 요구 사항과 TCP의 엄격한 순서 전달 요구사항-으로 인해 HTTP와 TCP는 최적으로 작동하지 않는다. 예를 들면, HTTP를 사용하여 여러 개의 내포된(embedded) 객체를 전송하는 경우, 각 객체는 목적지에 순서대로 도착하지 않아도 되지만 TCP는 순서대로 도착하도록 노력한다.

SCTP (stream control transmission protocol)[1]는 메시지 중심적으로 신뢰적인 전송을 제공하는 트랜스포트 계층의 프로토콜이다. TCP와 달리, SCTP는 멀티미디어 전송에 유용한 멀티스트리밍(multi-streaming) 기능을 가지고 있다. 이 기능은 목적지로 향하는 데이터를 여러 개의 스트림으로 분할하여 독립적으로 전송할 수 있도록 해준다. 따라서 어떤 특정한 스트림에서의 패킷 손실은 해당되는 스트림에만 영향을 주고, 다른 스트림에는 영향을 주지 않는다. 이에 비해, TCP는 단일 스트림을 사용하므로 특정 부분의 패킷 손실은 전체 데이터 전송에 영향을 미친다. 즉, TCP는 전달 순서를 엄격하게 유지하기 때문에 어떤 특정 객체 내에서 패킷이 손실되는 경우, 그 이후에 전송된 다른 객체들이 손실된 패킷이 재전송될 때까지 사용자에게 전달되지 못하고 기다려야만 한다. 이것을 head-of-line 블로킹이라 하는 데 사용자의 응답 시간(response time)을 증가시키는 원인이 된다. 이에 비해 SCTP는 하나의 어소시에션(association) 내에 웹 객체의 개수만큼 독립적인 스트림을 설정하여, 각 객체에 대응되는 스트림을 이용하여 개별적으로 전송한다. 따라서 SCTP를 사용하면 TCP에서 모든 객체가 겪는 head-of-line 블로킹이 특정 객체내로 한정됨으로써 사용자의 응답 시간을 줄일 수 있다[2].

일반적으로 응답 시간과 같은 통신 성능을 측정하기 위해서는 시뮬레이션, 실제 실험 그리고 해석적 모델링이 사용된다. 이와 관련하여 SCTP 위에서 연동되는 FTP(file transfer protocol)[8]와 SIP(session initiated protocol)[7]에 대한 성능 연구가 있으며 그 연구들은 실제 실험에 기반을 두고 있다. 그러나 SCTP 위에서 연동되는 HTTP의 응답 시간에 대한 해석적 모델은 제공된 바 없다.

응답 시간은 일반적으로 데이터 크기와 링크 전송률에 따른 전송 시간 뿐만 아니라, 혼잡 제어 메커니즘에 영향을 받는다. SCTP의 혼잡 제어 메커니즘은 TCP의 윈도우 기반의 혼잡 제어 메커니즘과 유사하며, 공통적인 기능은 슬로우-스타트(slow-start), 혼잡회피(congestion avoidance), 타임아웃(timeout) 그리고 빠른 재전송(fast retransmit) 등이다.

본 연구와 관련하여 기존의 TCP 위에서 연동되는 데이터 전송 지연에 대한 해석적 모델을 살펴보면 아래와 같다.

Padhye[3]는 TCP 위에서 안정 상태의 대량 데이터 전송을 고려하였다. 그러나 현재 인터넷에서 HTTP 데이터를 전송하는 대부분의 TCP 연결은 대량 데이터가 아닌 매우 작은 데이터 전송을 위한 짧은 연결(short connection)이다. 이 환경에서 웹의 성능을 지배하는 것은 연결 설정이나 슬로우-스타트 시간 등이다. 이러한 효과에 주목하여 Cardwell[4]은 앞의 안정 상태 모델을 확장하여 짧은 연결을 고려하였지만, 타임아웃 이후의 TCP 지연을 고려하지 않았다. Jiong[5]은 재전송 타임아웃 이후의 슬로우-스타트 시간을 고려함으로써 [4]의 모델을 개선하였다. 그러나 위 모델들은 유선 환경을 가정하였기 때문에 본 연구에서 가정하고 있는 멀티 홉(multi-hop) 모드를 갖는 무선 환경에 적용될 수 없다. 왜냐하면, 이러한 환경에서는 매우 작은 윈도우 크기 때문에 빠른 재전송이 가능하지 않은 것으로 알려져 있기 때문이다[6].

따라서 본 연구의 목적은 빠른 재전송이 가능하지 않

은 무선 환경에서 사용자가 HTTP over SCTP를 사용하여 웹 객체를 다운로드 받기까지의 평균 응답 시간을 추정하는 해석적 모델을 개발하는 데 있다.

본 연구에서 추정하는 평균 응답 시간은 종단 사용자가 요구하는 서비스 품질(quality of service)로 생각할 수 있기 때문에, 객체의 크기, 대역폭, 라운드 트립 타임(round trip time)등을 이용하여 사전에 응답 시간을 측정하기 위한 벤치마크로 사용할 수 있다.

본 연구에서 제안하는 HTTP over SCTP의 평균 응답 시간의 타당성을 입증하기 위해 간단한 테스트베드에서 실험을 통해 추정값과 실제 실험값을 비교하였고, 아울러, 기존의 HTTP over TCP의 평균 응답 시간과도 비교하였다. 이 연구의 초기 버전은[9]에 제시된 바 있다.

전체 4 장으로 구성되는 본 연구는 2 장에서 HTTP over SCTP의 평균 응답 시간 추정 모델과 알고리즘을, 3장에서는 성능평가와 분석을 제시하고 4장에서는 결론을 맺는다.

II. HTTP over SCTP의 평균 응답 시간 모델

1. 기본 개념

[표 1]은 SCTP 위에서 HTTP를 사용하여 웹 객체를 다운로드할 때 필요한 평균 응답 시간을 추정하기 위한 해석적 모델에 필요한 기호 정의를 나타낸다.

모델을 단순화하기 위해 웹 객체의 크기가 동일하고 수신된 패킷들은 윈도우 단위로 상위 계층으로 전달된다고 가정한다. 전송되는 객체의 크기가 θ 비트이고, 최대 전송 단위(maximum transfer unit)를 *mtu* 비트라고 하면 하나의 객체에 대해 전송되어야 할 패킷의 개수는 $\delta = \lceil \theta/mtu \rceil$이다.

패킷 손실 확률을 p 라고 할 때, 전체 패킷 손실 개수의 기댓값은 이항분포 (binomial distribution)에 의해 $\alpha = \lceil \delta p \rceil$가 된다. 이때, 특정 패킷 손실은 슬로우-스타트 또는 혼잡 회피 중에서 어느 한 시기에 일어난다.

[그림 1]은 SCTP의 혼잡 제어 메커니즘이다. $th(1)$, $th(2)$, $th(3)$은 슬로우-스타트 임계값(slow start threshold)으로 초기에 $th(1) = \infty$로 세팅된다. y 축은 SCTP의 cwnd (congestion window)를 표시하는 데, 초기값은 $2 \times mtu$ 이다. 따라서 SCTP는 2, 4, 8,.. 과 같이 지수적으로 cwnd를 증가시키는 슬로우-스타트 단계를 실행하다가 ①에서 타임아웃이 발생하면 패킷이 손실된 것을 탐지하게 된다.

표 1. 모델에 사용되는 기호 정의

Variable	Meaning
θ	전송되는 객체의 크기 (bits)
mtu	최대 전송 단위 (maximum transfer unit)
δ	객체 전송을 위한 패킷의 개수
p	패킷 손실 확률
α	δ 개의 패킷을 전송할 때 발생되는 패킷 손실 개수의 기댓값 ($\alpha = \lceil \delta p \rceil$)
$N(k)$	k 번째 패킷 손실 단계에서 전송을 위해 남아있는 패킷의 수 (k=1,2,...,α)
μ	클라이언트와 서버 사이의 평균 링크 전송률 (bps)
$E(T_{sctp})$	HTTP over SCTP의 평균 응답 시간
$E(T^k_{slow})$	k 번째 패킷 손실이 슬로우-스타트 구간에서 발생한 경우의 평균 응답 시간
$E(T^k_{cong})$	k 번째 패킷 손실이 혼잡 회피 구간에서 발생한 경우의 평균 응답 시간
β	패킷 손실이 일어난 구간을 나타내는 플래그; β = 0, 손실이 슬로우-스타트 구간에 발생 β = 1, 손실이 혼잡 회피 구간에 발생
$x(k)$	k 번째 패킷 손실 이전까지 전송된 패킷의 개수
$ST(X)$	k 번째 패킷 손실이 슬로우-스타트 구간에서 발생한 경우 X 개의 패킷을 전송하기 위해 걸린 슬로우-스타트 시간
ρ	min (Q, U-1)
Q	웹 서버가 정지한 시간
U	특정량의 데이터를 전송하기 위해 필요한 윈도우 개수
rtt	클라이언트와 서버 사이의 라운드 트립 타임 (round trip time)
$th(k)$	k 번째 패킷 손실 단계에서, 혼잡 회피가 시작되는 임계값(threshold)
$A_{th(k)}$	k 번째 패킷 손실 단계에서, $th(k)$ 이전에 전송된 패킷의 개수
$ST(A_{th(k)})$	k 번째 패킷 손실이 혼잡 회피 구간에서 발생한 경우 $th(k)$ 까지 전송하기 위해 필요한 슬로우-스타트 시간
$C(k)$	$x(k)$가 포함된 윈도우 번호 (1,2,..,$C(k)$)
$\gamma(k)$	$C(k)$ 윈도우에 포함된 패킷 개수
$b(k)$	k 번째 패킷 손실이 발생한 $C(k)$ 번째 윈도우에서 손실 이전에 전송된 패킷의 개수
R_{TO}	재전송 타임아웃
L	혼잡 회피 구간의 선형 증가 단계에서 전송된 패킷의 개수
R	a 개의 손실 이후에 남은 패킷($M(a+1)$)을 전송하기 위한 시간
$E(T_{sctp})$	제안된 모델에 의한 HTTP over SCTP의 평균 응답 시간
$E(T_{tcp})$	제안된 모델에 의한 HTTP over TCP 의 평균 응답 시간
T_{sctp}	실험에 의한 HTTP over SCTP의 평균 응답 시간
T_{tcp}	실험에 의한 HTTP over TCP의 평균 응답 시간

SCTP는 이에 대해 다음과 같이 반응한다[1].

$$th(2) = max(\frac{cwnd}{2}, 2 \times mtu) \qquad (1)$$
$$cwnd = 1 \times mtu$$

즉, 다음 과정의 임계값은 손실이 일어난 윈도우의 절반으로 줄이고, 혼잡 윈도우를 1로 하여 2, 4, 8,....순으로 증가시키면서 슬로우-스타트 단계를 반복한다. 만일 혼잡 윈도우가 임계값인 $th(2)$을 넘게되면, 혼잡 회피 단계를 실행한다. 이 단계에서는 1 개의 패킷마다 액노리지먼트(acknowledgement)를 필요로 하므로 선형 증가 단계라고도 한다. 이 단계에서 패킷이 손실되면 [그림 1]②, 타임 아웃 여부에 따라 두 가지의 대응이 가능하다. 먼저, 식 (1)을 이용하여 새로운 임계값($th(3)$)을 구한 후, 타임 아웃 이전에 3개의 중복된 액노리지먼트를 받으면 빠른 재전송 과정 [그림 1]③으로 들어가고, 그렇지 않으면 다시 슬로우-스타트 단계로 들어간다[그림 1]④. 본 연구에서는 타임아웃 동안에 3 개의 중복 액노리지먼트를 받을 수 없는 멀티-홉 무선환경을 가정하였으므로 슬로우-스타트 과정으로 들어간다.

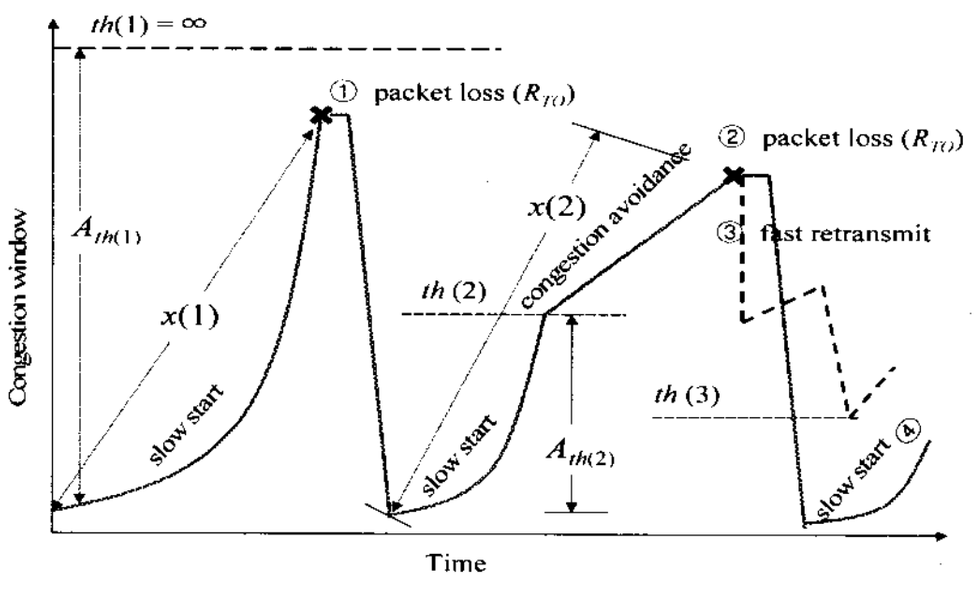

그림 1. SCTP의 혼잡제어

이상으로부터, k 번째 패킷 손실에 대해, 혼잡 회피가 시작되는 임계값 ($th(k)$, k=1,2,..,a) 까지 전송 가능한 패킷의 수 ($A_{th(k)}$)와 손실 이전에 전송된 패킷의 기대 개수 ($x(k)$: k=1,2,..,a)를 비교하면 손실 구간을 파악할 수 있다. 이 때, $x(k)$는 남아있는 패킷의 개수인 $N(k)$와 패킷 손실율(p)의 함수로 구해지며, 자세한 내용은 2절에서 다룬다.

결국, 임의의 k 번째 패킷 손실에 대해 $x(k) < A_{th(k)}$이면, 패킷 손실이 슬로우-스타트 동안에 일어난 것으로, $x(k) \geq A_{th(k)}$이면 혼잡 회피 동안에 일어난 것을 알 수 있다. 예를 들면, [그림 1]에서 첫 번째 패킷 손실 ①까지 전송된 패킷의 총 개수는 $x(1)$이고, $th(1)$까지 전송 가능한 패킷의 개수는 $A_{th(1)}$인 데, $x(1) < A_{th(1)}$ 이므로 손실이 슬로우-스타트 단계에서 일어난 것으로 알 수 있다. 동일하게, 손실 ② 이전까지 전송된 패킷의 개수, $x(2) > A_{th(2)}$ 이므로 손실이 혼잡 회피에서 일어났음을 알 수 있다.

HTTP over SCTP의 평균 응답 시간은 식(2)와 같이 주어진다.

$$E(T_{sctp}) = \sum_{k=1}^{a} [\beta E(T_{slow}^{k}) + (1-\beta)E(T_{cong}^{k})] + R \qquad (2)$$

식 (2)에서 SCTP의 첫 번째 패킷 손실(k=1)은 [그림 1]에서 보듯이, 항상 슬로우-스타트 구간에서 발생하므로 $E(T_{slow}^{1})$이 더해져야 한다. 두 번째 이후의 패킷 손실은 슬로우-스타트 또는 혼잡 회피 구간에서 발생한다. $E(T_{slow}^{k})$는 k 번째 패킷 손실(k=2,3,...a)이 슬로우-스타트 구간 에서 발생했을 때의 평균 응답 시간을, $E(T_{cong}^{k})$는 혼잡 회피 구간 동안에서 발생했을 때의 평균 응답 시간을 나타낸다. 임의의 패킷 손실이 슬로우-스타트 구간과 혼잡 회피 구간에서 동시에 발생할 수 없으므로 주어진 k 번째 손실에 대해, β는 0 또는 1이어야 한다. 즉, k 번째 패킷 손실이 슬로우-스타트 구간에서 발생하는 경우, β = 1로 놓으면 $E(T_{slow}^{k})$가 $E(T_{sctp})$에 더해져 누적된다. 동일한 방법으로, k 번째 패킷 손실이 혼잡 회피 구간에서 발생하는 경우, β=0으로 놓으면 $E(T_{cong}^{k})$가 $E(T_{sctp})$에 더해져 누적된다. 따라서 객체의 전체 평균 응답 시간을 구하려면, 패킷 손실 개수의 기댓값(a) 만큼 $E(T_{slow}^{k})$ 또는 $E(T_{cong}^{k})$ (k=1,2,...,a)를 더하면 된다. R은 마지막 패킷 손실이 일어난 후에 남은 데이터 ($N(a+1)$)를 송신하기 위해 필요

한 시간으로, 패킷 손실은 이미 기댓값인 a 개가 일어난 상태이므로 추가 패킷 손실을 고려할 필요가 없이 계산 가능하다. 즉, $N(a+1)$가 마지막 임계값($th(a+1)$)까지 전송 가능한 데이터의 량보다 작은 경우에는 슬로우-스타트 구간에서 전송이 완료된 것이므로 그 때 까지의 슬로우-스타트 시간($ST(N(a+1))$과 전송시간($N(a+1)\times mtu/\mu$)을 더하면 되고, 그렇지 않은 경우에는 혼잡회피 구간에서 전송이 완료된 것이므로 임계값까지의 슬로우-스타트 시간($ST(A_{th(a+1)})$과 전송시간 ($N(a+1)\times mtu/\mu$) 그리고 혼잡회피 구간에서의 추가시간 ($N(a+1)-A_{th}((a+1))\times rtt$)을 더하면 된다. 따라서 R은 식(2)와 같이 된다.

$$\begin{aligned}
&\text{if } N(\alpha+1) < A_{th(\alpha+1)},\\
&\quad R = ST(N(\alpha+1)) + N(\alpha+1)\frac{mtu}{\mu}\\
&\text{if } \ N(\alpha+1) \geq A_{th(\alpha+1)},\\
&\quad R = ST(A_{th(\alpha+1)}) + N(\alpha+1)\frac{mtu}{\mu}\\
&\qquad + [N(\alpha+1) - A_{th(\alpha+1)}] \times rtt
\end{aligned} \tag{3}$$

식(3)에서 사용한 기호 등의 구체적인 의미는 앞으로 전개되는 2절과 3절에서 보다 명확해 질 것이다.

2. 슬로우-스타트 구간에서의 패킷 손실

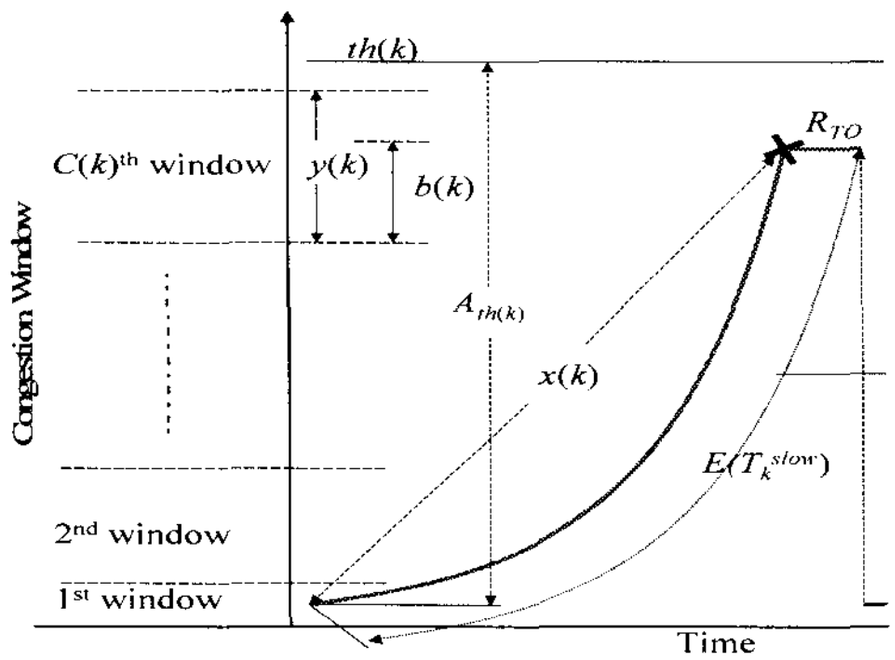

그림 2. 슬로우-스타트 구간에서의 K 번째 패킷 손실

[그림 2]는 HTTP over SCTP의 슬로우-스타트 단계에서 k 번째 패킷 손실(X)이 일어난 경우를 나타낸 것으로, 첫 번째 패킷 손실(k=1)에서는 첫 번째 윈도우에서 2 개, 두 번째 윈도우에서 4 개 등으로, 두 번째 이후의 슬로우-스타트 구간(k ≥2)에서는 1,2,4...등으로 지수적으로 증가하다가 임의의 C 번째 윈도우에서 $y(k)$ 개를 전송하는 도중에 손실이 발생한 것을 보여준다.

먼저, 손실 이전에 전송된 패킷의 기대 개수(손실 패킷은 제외)는 식(4)로 주어진다. $N(k)$는 k 번째 손실 단계에서 전송을 위해 남아있는 패킷의 개수로 재전송해야 할 패킷을 포함한 수자이다. [그림 2]에서 $y(k)$ 개의 윈도우를 전송하고 그 중에 $b(k)$ 개 이후의 패킷이 손실되는 경우, (k+1) 번째 단계에서 재전송해야 할 개수는 손실된 패킷을 포함하여 $y(k)-b(k)+1$이다.

$$x(k) = \frac{1-(1-p)^{N(k)}}{p} + (1-p)^{N(k)} \tag{4}$$

따라서 (k+1)번째 손실 단계에서 전송되어야 할 패킷의 수는

$$\begin{aligned}
&\text{for } k = 0,\\
&\quad N(k+1) = \delta = \lceil \theta/mtu \rceil\\
&\text{for } k = 1,2,3,...,\alpha\\
&\quad N(k+1) = N(k) - x(k) + y(k) - b(k) + 1
\end{aligned} \tag{5}$$

이다. 예를 들어, 식(4)에서 k=1 인 경우, $x(1)$은 $N(1)=\delta=\lceil\theta/mtu\rceil$가 대입되어 구해지고, k=2 인 경우, $x(2)$는 $N(2)=N(1)-x(1)+y(1)-b(1)+1 = \delta - x(1)+y(1)-b(1)+1$이 대입되어 얻어진다.

다음으로, $A_{th(k)}$ (k=1,2,...,a)은 $th(k)$까지 전송되는 패킷의 개수로, k=1에 대해서는 2, 4, 8... 식으로, k ≥2에 대해서는 1,2,4,8 식으로 2의 배수로 전송된다. 따라서

$$\begin{aligned}
&\text{for } k = 1,\\
&\quad A_{th(k)} = \begin{cases} 2^{\lceil \log_2^{th(k)+1} \rceil} - 1 & \text{if } th(k) = 2^j \\ 2^{\lceil \log_2^{th(k)+1} \rceil} - 1 + th(k) & \text{if } th(k) \neq 2^j \end{cases}\\
&\text{for } k = 2,3,...,\alpha+1,\\
&\quad A_{th(k)} = \begin{cases} 2^{\lceil \log_2^{th(k)-1} \rceil} - 2 & \text{if } th(k) = 2^j \\ 2^{\lceil \log_2^{th(k)-1} \rceil} - 2 + th(k) & \text{if } th(k) \neq 2^j \end{cases}
\end{aligned} \tag{6}$$

이 된다.

슬로우-스타트 기간에 k 번째 패킷 손실이 발생하는 경우의 평균 응답 시간($E(T_{slow}^{k})$)은 패킷 손실을 포함하고 있는 윈도우를 전송할 때까지의 슬로우-스타트 시간과 전송시간, 그리고 재전송 타임아웃의 합이다.

먼저, k 번째 패킷 손실이 발생한 윈도우 번호를 구하면 다음과 같다.

$$C(k)=\begin{cases} min\{j: 2^1+2^2+\cdots+2^j \geq x(k) \\ \quad = \lceil \log_2^{x(k)} \rceil & k=1 \\ min\{j: 2^0+2^1+\cdots+2^{j-1} \geq x(k) \\ \quad = \lceil \log_2^{x(k)+1} \rceil & k \geq 2 \end{cases} \quad (7)$$

따라서 $C(k)$ 번째 윈도우의 크기인 $y(k)$는

$$y(k)=\begin{cases} 2^{C(k)-1}-(2^{C(k)}-N(k))+1 & \text{if } N(k)<2^{C(k)} \\ 2^{C(k)-1} & otherwise \end{cases} \quad (8)$$

이 된다.

$y(k)$는 임계값인, $th(k)$보다 작아야 하므로 만일 계산된 $y(k)$가 $th(k)$보다 큰 경우에는 $y(k)$ = $th(k)$로 놓는다.

$C(k)$ 번째 윈도우에서 패킷 손실 이전까지 전송된 패킷의 개수는

$$b(k)=\begin{cases} x(k)+1-(2^1+2^2+\cdots+2^{C(k)-1}) \\ \quad = x(k)-2^{C(k)} & k=1 \\ x(k)+1-(2^0+2^1+\cdots+2^{C(k)-2}) \\ \quad = x(k)+1-2^{C(k)-1} & k \geq 2 \end{cases} \quad (9)$$

가 된다. 따라서 재전송을 요하는 패킷의 개수는 손실이 일어난 패킷을 포함하여 $y(k)$-$b(k)$+1이 됨을 알 수 있다. 이 패킷들은 아직 전송되지 못한 패킷과 합쳐져서 다음 (k+1) 번째 손실 단계에서 전송될 것이다.

그 개수는 식(5)의 $N(k+1)$로 주어진다. 따라서 손실된 패킷을 포함한 전송 패킷의 개수는 $x(k)$+$y(k)$-$b(k)$+1이 된다.

이제, 전송되는 임의의 패킷 개수에 대한 슬로우-스타트 시간을 구하기 위해 먼저 일반식을 구해 보자. SCTP는 초기에 혼잡 윈도우(congestion window)를 2개의 mtu로 하여 시작하므로 임의의 X 개의 패킷을 전송하는 경우의 슬로우-스타트 시간은 [10]을 이용하여 구할 수 있다.

$$ST(X)=\rho(rtt+\frac{2mtu}{\mu})-(2^{\rho+1}-2)\frac{mtu}{\mu}$$
$$where \qquad \text{for } k=1 \quad (10)$$
$$\rho=min(Q, U-1)$$
$$=\min\left\{\left\lfloor log_2(2+\frac{rtt}{mtu/\mu})\right\rfloor, \lceil log_2^{X+1}-1\rceil\right\}$$

식 (10)에서 rtt는 라운드-트립-시간, Q는 웹 서버의 정지 횟수 그리고 U는 X 개의 패킷을 전송하는 데 필요한 윈도우의 개수이다.

SCTP는 첫 번째 패킷 손실이후에 혼잡 윈도우를 1로 줄이므로 임의의 X 개의 패킷을 전송하는 경우, $k \geq 2$ 번째 패킷 손실까지의 슬로우-스타트 시간은 식(11)처럼 된다.

$$ST(X)=\rho(rtt+\frac{mtu}{\mu})-(2^{\rho}-1)\frac{mtu}{\mu}$$
$$where \qquad \text{for } k \geq 2 \quad (11)$$
$$\rho=min(Q, U-1)$$
$$=\min\left\{\left\lfloor log_2(1+\frac{rtt}{mtu/\mu})\right\rfloor+1, \lceil \log_2^{X+1}\rceil-1\right\}$$

따라서 $x(k)$+$y(k)$-$b(k)$+1의 슬로우-스타트 시간은 k=1 인 경우 식(10)에, $k \geq 2$ 인 경우는 식(11)에 X 대신에 $x(k)$+$y(k)$-$b(k)$+1을 대입하면 구할 수 있다.

다음으로 $x(k)$+$y(k)$-$b(k)$+1 개의 패킷을 전송하기 위한 시간은 최대 전송 단위가 mtu이고 링크의 평균 전송률이 μ이므로 간단히 [$x(k)$+$y(k)$-$b(k)$ +1]×mtu/μ가 된다.

여기서, $y(k)$-$b(k)$+1 만큼의 패킷은 다음 (k+1)번째 손실 단계에서 전송되므로 수신측 웹 사용자는 이미 수신된 $b(k)$ 개만큼의 패킷을 응용계층으로 전달하지 못하므로 $y(k)$-$b(k)$+1 개만큼의 재전송 시간과 슬로우-스타트 시간이 head-of-line 블로킹 시간이 됨을 알 수 있다. 이상으로부터 k 번째 패킷이 슬로우-스타트 기간에 손실되는 경우, 그에 따른 평균 응답 시간은 식(12)로 유도된다.

$$E(T_{slow}^{k}) = ST(x(k)+y(k)-b(k)+1)+ [x(k)+y(k)-b(k)+1)]\times\frac{mtu}{\mu} + R_{TO} \quad (12)$$

위 식에서 R_{TO}는 재전송 타임아웃으로 3/2×rtt로 주어진다. 서론에서도 기술한 바와 같이 멀티홉 무선 환경에서는 타임아웃 동안에 3개의 중복 ACK를 수신하기가 어렵기 때문에 빠른 재전송이 가능하지 않다. (k+1)번째 패킷 손실에 대비하여 임계값($th(k+1)$)은 현재의 윈도우의 크기가 $x(k)$+$y(k)$−$b(k)$+1 이므로 식(13)이 된다.

$$\text{for } k=0,\quad th(k+1)=\infty$$

$$\text{for } k=1,2,3,\ldots,\alpha\quad th(k+1)=max\left(\left\lceil\frac{x(k)+y(k)-b(k)+1}{2}\right\rceil, 2mtu\right) \quad (13)$$

3. 혼잡회피 구간에서의 패킷 손실

이제, k 번째 패킷 손실이 혼잡회피 구간에서 발생하는 경우의 평균 응답시간, $E(T_{cong}^{k})$를 계산해 보자. [그림 3]은 HTTP over SCTP의 혼잡 회피 구간에서 k 번째 패킷이 손실되는 상황을 보여준다. 이 경우는 슬로우-스타트 구간에서 패킷 전송이 첫 번째 윈도우에서 1 개, 두 번째 윈도우에서 2 개 등으로 지수적으로 증가하다가 임계값($th(k)$)을 초과하여 혼잡 회피 구간으로 진입한 경우에는 선형적으로 증가하며, 결국, 특정 패킷이 손실되는 과정을 보여준다.

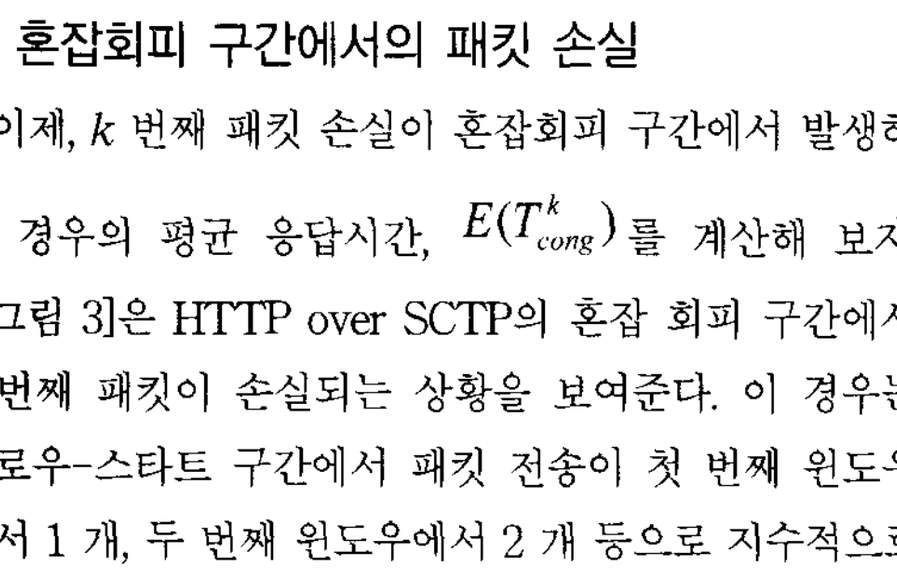

그림 3. 혼잡회피 구간에서의 K 번째 패킷 손실

패킷 손실이 혼잡회피 구간에서 발생하는 지를 확인하는 방법은, 앞에서 설명한 바와 같이, 손실 이전 까지 전송된 패킷의 개수인 $x(k)$가 혼잡제어 임계값까지 전송 가능한 패킷 개수($A_{th(k)}$)보다 큰 지를 확인하면 된다.

임의의 k 번째 패킷이 혼잡회피 구간에서 손실되는 경우의 평균 응답 시간($E(T_{cong}^{k})$)은 [그림 3]에서 보듯이 임계값($th(k)$)까지 전송되는 슬로우-스타트 시간, 손실 이전까지 전송된 $x(k)$의 전송 시간, 혼잡 회피 구간에서 선형적으로 증가된 패킷들(L)의 라운드-트립-시간, 그리고 재전송 타임아웃의 합이다.

먼저, 임계값($th(k)$)까지의 슬로우-스타트 시간을 구해보면, 식(11)의 일반식에서 X 대신에 $A_{th(k)}$를 대입하면 되므로,

$$ST(A_{th(k)}) = \rho(rtt+\frac{mtu}{\mu})-(2^{\rho}-1)\frac{mtu}{\mu}$$
$$where$$
$$\rho = min(Q, U-1)$$
$$= \min\left\{\left\lfloor log_2(1+\frac{rtt}{mtu/\mu})\right\rfloor+1, \left\lceil\log_2^{(A_{th(k)}+1)}\right\rceil-1\right\} \quad (14)$$

손실 이전에 전송된 패킷의 기대 개수(손실 패킷은 제외) $x(k)$는 식(4)에 의해 구해지고, 전송된 개수는 손실 패킷을 포함하여 $x(k)$+1 개이다.

다음으로, 혼잡 회피 구간에서 전송된 패킷의 개수(L)는 $x(k) = A_{th(k)} + L$로부터

$$L = x(k) - A_{th(k)} \quad (15)$$

이다. 이 L 개의 패킷 각각에 대해서는 확인응답(acknowledgement)이 필요하므로, 이를 위한 추가 시간은 L×rtt이다. 재전송 타임아웃은 R_{TO} = 3/2 × rtt가 된다.

이상을 정리하면, k 번째 패킷이 혼잡회피 구간에서 손실되는 경우의 평균 응답 시간은

$$E(T_{cong}^{k}) = ST_{A_{th(k)}} + (x(k)+1)\frac{mtu}{\mu} + L\times rtt + R_{TO}$$
$$= ST_{A_{th(k)}} + (x(k)+1)\frac{mtu}{\mu} + (L+3/2)\times rtt \quad (16)$$

가 된다. 본 연구에서는 빠른 재전송이 가능하지 않은 무선 환경을 가정하였으므로, (k+1)번째의 혼잡제어 메커니즘은 다시 슬로우-스타트 단계부터 반복될 것이다. 한편, (k+1)번째 패킷 손실에 대비하여 임계값 (th(k+1))은 아래 식에 의해 변경된다.

$$th(k+1)=\begin{cases}\infty & k=0\\ max\left(\left\lceil\frac{x(k)+1}{2}\right\rceil, 2\times mtu\right) & k=1..,\alpha\end{cases} \quad (17)$$

다음 단계에서 전송을 위해 남아있는 패킷의 개수는

$$N(k+1)=\begin{cases}\delta=\lceil\theta/mtu\rceil & k=0\\ N(k)-x(k) & k=1,2,...,\alpha\end{cases} \quad (18)$$

이 된다.

4. HTTP over SCTP 응답시간 추정 알고리즘

지금까지 논의한 모델에 기초하여 전체 순서를 알고리즘으로 구성하면 [그림 4]와 같다. 객체의 패킷 개수가 δ 일 때, 알고리즘의 시간 복잡도는 O(δ)이다.

Procedure object response delay

Begin

1. Compute the total number of packets in object ($\delta=\lceil\theta/mtu\rceil$) and the expected number of packet loss, $\alpha=\lceil\delta p\rceil$
2. Set $N(1)=\delta$ and $th(1)=\infty$
3. Set $E(T_{sctp})=0$

for all k such that k=1,2$\cdots$,α do

① Set $E(T^k_{slow})=0$ and $E(T^k_{cong})=0$

② Compute the expected number of packets sent before the loss,

$$x(k)=\frac{1-(1-p)^{N(k)}}{p}+(1-p)^{N(k)}$$

③ Compute the total number of packets sent until the threshold($A_{th(k)}$),

for $k=1$,

$$A_{th(k)}=\begin{cases}2^{\lceil\log_2^{th(k)+1}\rceil}-1 & \text{if } th(k)=2^j\\ 2^{\lceil\log_2^{th(k)-1}\rceil}-1+th(k) & \text{if } th(k)\neq 2^j\end{cases}$$

for $k=2,3,...,\alpha+1$,

$$A_{th(k)}=\begin{cases}2^{\lceil\log_2^{th(k)+1}\rceil}-2 & \text{if } th(k)=2^j\\ 2^{\lceil\log_2^{th(k)-1}\rceil}-2+th(k) & \text{if } th(k)\neq 2^j\end{cases}$$

④ If $x(k)<A_{th(k)}$

{Set β=1 and perform Slow_start_rtn (k);}

else

{Set β=0 and perform Congestion_avoid_rtn (k);}

⑤ Accumulate the mean response delay,

$$E(T_{sctp})=E(T_{sctp})+\beta E(T^k_{slow})+(1-\beta)E(T^k_{cong})$$

end for

4. Find mean response time,

$$E(T_{sctp})=E(T_{sctp})+R$$

where,

① Find $ST(N(\alpha+1))$ or $ST(A_{th(\alpha+1)})$ using eqn. (11).

② Compute R.

if $N(\alpha+1)<A_{th(\alpha+1)}$,

$$R=ST(N(\alpha+1))+N(\alpha+1)\frac{mtu}{\mu}$$

if $N(\alpha+1)\geq A_{th(\alpha+1)}$,

$$R=ST(A_{th(\alpha+1)})+N(\alpha+1)\frac{mtu}{\mu}+[N(\alpha+1)-A_{th(\alpha+1)}]\times rtt$$

End

Slow_start_rtn (k)

1. Compute the slow-start time in the slow start phase using eqn. (10) for k=1, or using eqn. (11) for k=2,3..,a when sending packets of a web object.
2. Compute the window number($C(k)$) in which loss is occurred using eqn. (7).
3. Compute the window size ($y(k)$) which covers

x(k) packets when packet loss occurs using eqn. (8).

4. Compute the number of packets ($b(k)$) sent before the loss in the $C(k)^{th}$ window using eqn. (9).
5. Find the object response delay during slow-start ($E(T_{slow}^{k})$) using eqn, (12).
6. Set $N(k+1)$ using eqn. (5).
7. Set $th(k+1)$ using eqn. (13).

Congestion_avoid_rtn (k)

1. Compute the slow-start time ($ST(A_{th(k)})$ until congestion avoidance threshold ($A_{th(k)}$) using eqn. (11)
2. Compute the number of packets before the loss (L) during congestion avoidance using eqn. (15).
3. Find the object response delay during congestion avoidance($E(T_{cong}^{k})$) using eqn. (16).
4. Set $th(k+1)$ using eqn. (17).
5. Set $N(k+1)$ using eqn. (18).

그림 4. 평균 객체 응답 시간 추정 알고리즘

Ⅲ. 성능평가

이 장에서는 2 장에서 제시한 해석적 모델에 대한 적합도를 검증한다. 이를 위해, TCP와 SCTP를 시뮬레이션하는 웹 서버와 HTTP를 에뮬레이션하는 환경을 고려한다. 즉, TCP와 SCTP를 공평하게 비교하기 위해, TCP에 기반을 둔 HTTP를 사용하지 않을 것이다. 왜냐하면, SCTP 기반의 웹 서버가 아직 미비하고, 구현되었다하더라도, TCP에 비해 성능 튜닝이 되지 않았기 때문이다. 본 연구에서 제시한 모델의 기본적인 목적함수가 평균 응답 시간이므로, 시뮬레이션 환경역시 간단히 웹 객체를 요구하고 전송받는 간단한 환경을 가정하였다. 그렇지만, 해석적 모델을 검증하는 데는 문제가 없다.

[표 2]는 실험에 사용된 테스트베드 환경을 보여준다. 데스크 탑 컴퓨터는 데이터 전송을 위한 클라이언트-서버로 사용되었다. 멀티 홉 특성을 갖는 무선 환경을 시뮬레이션하기 위해 클라이언트와 서버 사이에 NIST 에뮬레이터 [11]를 장착한 컴퓨터를 사용하여 여러 가지 네트워크 조건- 패킷 손실, 대역폭 그리고 RTT 등을 조절하였다.

실험을 위해 HTTP over SCTP와 HTTP over TCP를 흉내내는 2 개의 Linux C 서버 프로그램을 작성하였다. 또한, 파이프라이닝(TCP/SCTP)과 멀티스트리밍(SCTP)를 시뮬레이션하는 2 개의 클라이언트 프로그램을 작성하였다. 웹 객체의 응답 시간은 Ethereal[12]을 이용하여 캡쳐된 패킷 수준에서 측정되었다.

표 2. 실험환경

노드	하드웨어	소프트웨어	운영체제
웹 서버	Dell Dimension Desktop.	File Sender program.	Redhat Linux 9 kernel 2.6.6
클라이언트	Dell Dimension Desktop.	File Receiver program.	Redhat Linux 9 kernel 2.6.6
NIST 에뮬레이터	Dell Inspiron-100 Laptop.	Nist-Emulator Software.	Redhat Linux 9 kernel 2.4.20

[표 3]은 본 연구에서 제안한 모델의 평균 응답시간과 실험값을 보여준다. $E(T_{sctp})$는 제안된 모델의 HTTP over SCTP의 평균 응답 시간을, $E(T_{tcp})$는 제안된 모델의 HTTP over TCP의 평균 응답 시간을 각각 나타낸다. 여기서 HTTP over TCP의 모델은 초기 윈도우의 개수가 HTTP over SCTP의 그것과 다른 점을 제외하고는 기본적으로 동일하다. 즉, 2 장에서 제시된 [그림 4]의 알고리즘에서 첫 번째 패킷 손실이 슬로우-스타트 구간에서 발생하는 경우의 평균 응답 시간 ($E(T_{slow}^{1})$)이 다르게 얻어지는 것을 제외하고는 그 처리 과정이 동일하다.

T_{sctp}는 실험에 의한 HTTP over SCTP의 평균 응답 시간을, T_{tcp}는 실험에 의한 HTTP over TCP의 평균 응답 시간을 각각 나타낸다. 객체의 평균 크기(θ)는

13.5 KB이고 최대 전송단위(*mtu*)는 536 B이다. HTML 파일에 내포된 웹 객체의 개수는 5 개이다.

먼저, *rtt*를 256 ms, 링크 전송률(μ)을 40 Kbps로 고정시키고, 패킷 손실율(p)를 변화시켰을 때의 값을 살펴보면, p가 감소함에 따라, 식 (8),(9)에 의한 재전송 개수가 0으로 근접함으로써, 슬로우-스타트 시간과 재전송 시간이 0에 근접하게 된다. 그 이유는, 패킷 손실의 경우에만 재전송 패킷에 대한 슬로우-스타트 시간이 필요하기 때문이다.

표 3. HTTP over SCTP와 HTTP over TCP에 대한 모델과 실험값의 비교

파라미터 \ 평균값		$E(T_{tcp})$	$E(T_{sctp})$	T_{tcp}	T_{sctp}
p (%)	0.4	20.76	20.25	18.19	18.11
	1	20.33	20.30	18.22	18.13
	2	21.26	20.75	18.49	18.34
μ (Kbps)	40	18.27	18.13	21.33	20.82
	400	4.91	4.41	5.09	4.58
	3,000	3.69	3.28	3.99	3.48
rtt (ms)	55	14.78	14.72	13.71	13.60
	80	14.86	14.76	14.55	14.47
	256	18.22	18.13	21.33	20.82

두 번째로, p = 1 %, *rtt* = 0.256 초로 고정하고 링크 전송률(μ)를 증가시키면, HTTP over TCP와 HTTP over SCTP의 평균 응답 시간은 거의 같아짐을 알 수 있다. 이것은 μ가 증가함에 따라, mtu/μ의 값이 재전송 시간을 감소시키기 때문이다.

세 번째로, p = 1 %, μ = 40 Kbps로 고정시킨 후에 *rtt*를 변화시켰다. [표 2]는 *rtt*가 증가함에 따라 평균 응답시간이 급격하게 증가함을 보여주고 있다. 이것은 HTTP over SCTP의 평균 응답 시간이 *rtt*에 대해 가장 민감한 것을 보여주는 것이다.

[표 3]에서 모델과 실험값 사이의 평균 표준 편차는 6 %인데, 이 작은 오차는 NIST 에뮬레이터의 부정확도에 기인한다. [표 3]에서 추가적으로 발견할 수 있는 것은 HTTP over SCTP의 평균 응답 시간이 HTTP over TCP에 비해 모델과 실험 모두에서 작게 나타나고 있다는 사실이다.

Ⅳ. 결 론

본 연구는 멀티-홉 무선 환경에서 HTTP over SCTP에 대한 평균 응답 시간을 추정하는 해석적 모델을 다루었다. 평균 응답 시간은 웹 사용자에게 제공되는 서비스 품질의 한 요소로 웹 성능을 평가하기 위한 중요한 파라미터 중 하나이다. 기존의 HTTP over TCP에 대한 모델이 단일 패킷 손실이나 유선 환경을 가정한 데 비해, 본 연구는 다중 패킷 손실이 발생하고, 빠른 재전송이 가능하지 않은 멀티-홉 무선 환경을 가정하였다. 간단한 테스트베드에서의 시뮬레이션 결과, 해석적 모델과 실제 실험값 사이의 표준 오차가 6 % 이내의 작은 값을 가짐으로써 해석적 모델이 비교적 잘 맞는 것으로 나타났다. 아울러, HTTP over SCTP의 평균 응답 시간이 모델과 실험 모두에서 HTTP over TCP보다 작음이 확인되었다. 앞으로의 연구에서는 보다 정교하면서도 유무선 환경에 모두 적용될 수 있는 모델의 개발이 기대된다.

참 고 문 헌

[1] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson, "Stream Control Transmission Protocol," RFC 2960, 2000.

[2] S. Fu, M. Atiquzzaman, L. Ma, and Y. Lee, "Signaling Cost and Performance of SIGMA," Wireless Communication and Mobile Computing, Vol.5, No.7, pp.825-845, 2005.

[3] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Reno Performance: A Simple Model and its Empirical Validation," ACM Transactions on Networking, Vol.8, No.2, pp.133-145, 2000.

[4] N. Cardwell, S. Savage, and Y. Anderson, "Modeling TCP Latency," IEEE Infocom, Tel Aviv, Vol.3, pp.1742-1751, 2000.

[5] Z. Jiong, Z. Shu-jing, and Qi-gang, "An Adapted Full Model for TCP Latency," Proceedings of IEEE TENCON '02, Beijing, Vol.2, pp.801-804, 2002.

[6] D. Oliveria and R. Braun, "A Dynamic Adative Acknowledgement Strategy for TCP over Multihop Wireless Networks," Proceedings of IEEE INFOCOM '05, pp.1863-1874, 2005.

[7] R. Camarllo, R. Kantola, and R. Schulzrinne, "Evaluation of Transport Protocols for the Session Initiation Protocol," IEEE Network, Vol.17, No.5, pp.40-46, 2003.

[8] S. Ladha and P. Amer, "Improving Multiple File Transfer Using SCTP Multistreaming," University of Delaware, TR, 2003.

[9] Y. Lee, M. Atiquzzaman, and S. Sivagurunathan, "Mean Response Time Estimation for HTTP over SCTP in Wireless Environment," Proceedings of IEEE International conference on communications, pp.164-169, 2006.

[10] J. Kurose and K. Ross, *Computer Networking*, Addison Wesley, 2006.

[11] http://snad.ncsl.nist.gov/itg/nistnet

[12] http://www.etherreal.com

저 자 소 개

이 용 진(Yong-Jin Lee) 정회원

- 1995년 2월 : 고려대학교 전산과학과(이학박사)
- 1995년 3월 ~ 2005년 8월 : 우송대학교 컴퓨터정보학과 부교수
- 2003년 8월 ~ 2004년 8월 : 미국 Oklahoma 주립대학 컴퓨터학과 방문연구원
- 2005년 9월 ~ 현재 : 한국교원대학교 기술교육과 부교수

<관심분야> : 무선 및 이동 네트워크, 인터넷 QoS, 인터넷 토폴로지