Journal of The Korea Society of Computer and Information
Vol. 19, No. 1, January 2014
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2014.19.1.043

모바일 환경에서 SPDY 프로토콜의 성능분석에 관한 연구

김 희 정*, 이 규 선*, 이 성 원*

A Research on the Performance Analysis of SPDY Protocol in Mobile Networks

Hee-Jung Kim*, Gyu-Sun Yi *, Sung-Won Lee*

요 약

SPDY는 기존 HTTP(Hypertext Transfer Protocol)/1.1의 문제점을 보완하여 웹 속도를 향상시키기 위한 목적으로 구글에서 새롭게 제안한 프로토콜이다. 본 논문은 다양한 환경에서의 성능 측정을 실시함으로써 SPDY 프로토콜의 특징을 알아보고 기존 프로토콜과의 차이점을 비교 분석하였으며 나아가 SPDY의 Flow Control에 대해서 좀 더 자세히 살펴보았다. 측정 실험들을 통해 SPDY 프로토콜은 3G 환경에서는 성능 향상을 보이지만 고속의 무선랜 환경과 이동체 환경에서는 뚜렷한 성능 향상을 보이지 못하는 문제점이 있음을 파악하였으며 Flow Control 역시 제대로 동작하지 않고 있음을 확인하였다. 마지막으로 성능 측정 실험 결과와 이에 대한 분석을 바탕으로 하여 SPDY 프로토콜의 개선방향을 제시하였다.

▸ Keywords : SPDY, 3G, 무선랜, 와이브로, 페이지로드타임, SPDY Flow Control

Abstract

SPDY is the new protocol proposed by Google to complement problems of HTTP(Hypertext Transfer Protocol)/1.1 and to improve web speed. In this paper, we evaluated the performance of SPDY protocol in a variety environment. By this evaluation, we examined the characteristics of the SPDY protocol and compared the differences between the existing protocol and SPDY protocol. And we took a closer look at the Flow Control of SPDY. Through this, we analyzed the problem of SPDY. It improved performance at 3G network environment, but failed to improve performance at high speed WLAN and mobility environments. We also verified that Flow Control does not work

•제1저자 : 김희정 •교신저자 : 이성원
•투고일 : 2013. 10. 05, 심사일 : 2013. 10. 30, 게재확정일 : 2013. 11. 30.
* 경희대학교 컴퓨터공학과(Dept. of Computer Engneering, Kyung Hee University)
※본 연구는 미래창조과학부 및 정보통신산업진흥원의 대학 IT 연구센터 육성지원 사업의 연구결과로 수행되었음 (NIPA-2013-(H0301-13-4006))

well. And finally we proposed directions for improvement of this protocol.

▸ Keywords : SPDY, 3G, WLAN, WiBro, Page load time, SPDY Flow Control

I. 서 론

컴퓨터와 인터넷의 폭발적인 발전, 그리고 스마트폰 등 다양한 모바일 기기의 개발로 인하여 시간과 장소에 상관없이 인터넷 접속이 가능해지면서 인터넷은 우리 생활 속 깊숙한 곳에 자리 잡았다. 단순하고 규모가 작은 문서들로 이루어졌던 1세대 웹 페이지들과는 다르게 오늘날의 웹 페이지들은 더욱 화려해지고 그 속에 많은 정보들을 담고 있으며 이에 따라 페이지의 크기 역시 계속해서 증가하였다. 그러나 우리가 일반적으로 인터넷에 접속할 때 사용하는 HTTP/1.1은 1999년에 나온 버전으로 빠르게 규모가 커지는 웹 페이지의 변화에 유연하게 대처하지 못하였으며, 이는 인터넷 속도의 증가에도 불구하고 웹 속도를 저하시키는 하나의 원인이 되었다 [1].

인터넷 국제 표준화기구인 IETF(Internet Engineering Task Force)는 변화한 웹 페이지 환경에서의 적응을 위하여 HTTP/1.1의 등장 이후 약 15년 만에 새로운 버전인 HTTP/2.0의 표준화를 발표하였으며, SPDY 프로토콜을 기반으로 새로운 프로토콜을 정의한다고 밝혔다 [2]. SPDY는 기존 HTTP 프로토콜의 문제점을 보완하여 페이지 로드 타임을 줄이고자 하는 목적으로 구글이 새롭게 제안한 프로토콜이다. 마이크로소프트에서도 이를 발전시킨 형태의 HTTP Speed+ Mobility 를 제안하며 표준화에 힘쓰고 있는 등 SPDY는 새로운 통신 프로토콜로서 주목 받고 있다 [3]. 현재 구글 Chrome/Chromium, Firefox, Amazon Silk 브라우저 등에서 SPDY를 지원하고 있으며, Android/iPhone 용 모바일 Chrome과 Android 용 모바일 Firefox 브라우저에서도 SPDY를 지원 한다 [4,5].

문제는 SPDY 프로토콜의 구체적인 성능은 분석되지 않은 상황에서, 구글의 SPDY 오픈 소스 소프트웨어가 확산되고 있다는 것이다. 따라서 구글이 주장하는 장점들이 개념적으로만 받아들여질 뿐, 정성/정량적인 장단점은 규정되지 않고 있다.

본 논문에서는 여러 가지 모바일 환경에서 SPDY 프로토콜의 성능을 측정해봄으로써 기존의 HTTP 프로토콜과의 차이점을 비교 분석한다. 또한 SPDY의 특징 중 하나인 Flow Control에 대해서 좀 더 심도 있게 살펴본다. 나아가 측정 실험을 통하여 나타나는 SPDY 프로토콜의 특징과 문제점을 파악하고 프로토콜의 개선방향에 대해서도 이야기하도록 한다.

본 논문은 서론에 이어서 2장에서 SPDY 프로토콜이 무엇인지에 대하여 알아보고 3장에서는 기존에 진행 되었던 관련 연구들을 살펴본다. 이어 4장에서는 성능평가 방법 및 결과 값들을 분석하고 5장에서 SPDY의 Flow Control에 대하여 좀 더 자세히 알아본 후 마지막으로 6장에서 결론을 내리도록 한다.

II. SPDY 프로토콜의 개요

SPDY는 페이지 로드 타임을 줄이기 위한 목적으로 구글이 제안한 프로토콜이다. TCP/IP 모델의 SSL 상위 계층에 추가된 구조로 이루어졌으며, 주요 기능은 다음과 같다.

첫째, SPDY는 서버와 클라이언트 사이에 하나의 TCP 커넥션을 유지한다. 다수의 TCP 커넥션을 만들어 요청 메시지를 보내고, 이에 대한 응답 메시지를 받으면 커넥션을 해지하는 방식의 HTTP 프로토콜과는 달리 SPDY는 도메인 당 하나의 TCP 커넥션을 원칙으로 하며, 이를 계속해서 유지하려고 한다. SPDY는 연결된 하나의 커넥션 안에서 독립적인 스트림들을 생성하여 다수의 HTTP 요청을 동시에 수행하며, 이 때 스트림 간의 전송 순서는 지켜질 필요가 없다. 이를 위해 스트림 생성 시 스트림마다 고유한 ID를 부여한다.

둘째, 생성되는 스트림들 사이에 우선순위를 주고 이를 기반으로 하여 스트림을 전송한다. 이를 통하여 우선순위가 높은 스트림의 지연을 막을 수 있다.

셋째, 반복되는 HTTP 헤더를 압축하여 그 크기를 감소시킨다. 대부분의 헤더 정보는 요청에 있어 반복적으로 사용된다. SPDY 프로토콜은 이러한 헤더를 압축시킴으로써 그 크기를 80% 이상 감소시킨다. 데이터 전송률이 낮은 네트워크 환경에서는 헤더의 압축만으로도 성능 향상을 기대할 수 있다.

넷째, 클라이언트의 요청이 없어도 서버가 요청 예측이 가능한 자원을 미리 클라이언트에 전송해주는 Server push 기

능을 제공한다.

이 외에도 서버가 응답메시지 전송 시 웹페이지의 특정 자원들에 대한 정보를 제공하는 Server hint 등의 기능을 제공하며 웹 속도의 향상을 지향 한다 [4,5].

III. 관련 연구

SPDY는 HTTP/2.0의 표준화에 기반이 되는 등 새로운 통신프로토콜로 주목받으며 그 중요성이 대두되고 있는 만큼, 최근에 SPDY에 대한 성능 평가 역시 다양한 방법으로 이루어졌다.

살펴볼 기존 연구들은 모두 웹 페이지에 접속할 때의 페이지 로드 타임을 측정함으로써 SPDY 프로토콜의 성능을 알아보았다. 그러나 기존 연구는 모바일 네트워크를 고려하지 않거나, 제한되고 단순한 유선 네트워크 환경을 고려하고 있다. 본 논문은 실제 모바일 네트워크 환경에서 성능평가를 진행했다는 점에서 기존의 연구들과의 차이점이 존재한다.

1. CableLabs의 SPDY 성능분석

미국 케이블 표준화 기구인 CableLabs는 인터넷을 더 빠르게 만들기 위한 목적으로 등장한 두 가지 프로토콜인 SPDY와 TCP tune-up에 대하여 조사하고 나아가 다양한 시뮬레이션환경에서 두 프로토콜에 대한 성능평가를 실시하였다. 테스트 네트워크는 ESX 가상서버 위에 7개의 가상 호스트로 구성 되었으며 가상 호스트들은 각각 두 개의 네트워크 인터페이스를 가진다. 네트워크 인터페이스는 ESX환경 내부의 가상 랜과 CableLabs의 네트워크망에 연결되었다. CableLabs는 이와 같은 환경에서 측정 웹 페이지의 종류, 데이터 전송률, RTT 등에 조금씩 변화를 주는 방식으로 총 288가지의 경우에 대해 측정을 진행하였으며 측정 결과 두 가지 프로토콜을 사용하는 경우가 기존 프로토콜을 사용하는 경우와 비교하여 평균 29% 정도의 페이지 로드 타임의 감소를 보였다. 그러나 적은 수의 큰 이미지들로 구성된 웹 페이지의 경우 SPDY 프로토콜을 사용하는 경우의 성능이 오히려 더 떨어짐을 확인할 수 있었다 [6].

2. 구글의 SPDY 성능분석

구글은 모바일 단말에서 SPDY 프로토콜의 성능을 알아보기 위하여 실제 안드로이드폰에서의 성능측정 실험을 진행하였다. 실험은 네트워크 에뮬레이션 툴을 사용하여 실제 3G 네트워크와 비슷한 환경을 조성한 후 모바일 단말의 Chrome 브라우저에서 31개 웹사이트의 77개 URL에서 접속하며 페이지 로드 타임을 측정하는 방식으로 진행되었다. 측정 결과 한 가지를 제외한 모든 경우에서 SPDY 프로토콜을 사용한 경우에 기존의 HTTP 프로토콜을 사용했을 때와 비교하여 페이지 로드 타임의 측면에서는 평균 23%, 속도 측면에서는 1.3배의 성능 향상을 보였다 [7].

IV. 모바일 환경에서의 SPDY 성능 분석

본 논문은 모바일 환경에서 SPDY 프로토콜을 사용할 때와 기존 프로토콜을 사용할 때의 성능 비교 및 SPDY 프로토콜의 특성 파악을 목표로 성능 측정을 진행 하였다. 실험 시 Ubuntu 11.04 운영체제를 탑재한 랩탑에서 Chromium 브라우저와 Firefox 브라우저를 사용하였으며, 성능측정의 기준은 페이지 로드 타임으로 정하고 각각의 경우에 대하여 이를 측정하였다. 이때 페이지 로드 타임은 웹 페이지의 모든 자원을 다운로드 하는데 걸린 총시간을 의미한다.

SPDY의 특성 및 성능을 좀 더 정확하게 파악하기 위하여 측정 실험은 크게 세 가지 경우로 나누었다.

1. 웹 페이지 크기에 따른 SPDY 성능 분석

SPDY 프로토콜의 사용유무와 더불어 웹 페이지의 크기, 무선통신의 종류와 품질에 변화를 주어 성능측정을 진행하였다. 무선통신의 종류는 3G와 무선 랜의 두 종류로 나누었으며 교내에서 무선통신의 신호세기가 좋은 장소와 좋지 않은 장소를 선정하여 이 두 장소에서 측정을 진행하였다. 따라서 측정환경은 3G 환경에서 신호세기가 좋을 때와 좋지 않을 때, 무선랜 환경에서 신호세기가 좋을 때와 좋지 않을 때 총 네 가지 경우로 나뉘었다. 또한, 웹 페이지의 크기가 미치는 영향을 살펴보기 위하여 Google play와 SPDY 프로토콜을 지원하는 국내의 한 커뮤니티 사이트에서 크기가 다른 두 개의 페이지를 선정하여 성능측정을 진행하였다.

표 1. 웹 페이지 크기 별 측정 사이트 정보
Table 1. Information of website in variety size

사이트	Object 수	Domain 수	페이지 크기(KB)
Google Play	52	17	1592.43
국내 사이트 (2MB)	88	4	2299.99
국내 사이트 (4MB)	86	6	3985.02

표 2. Google Play 사이트에서 측정한 페이지 로드 타임
Table 2. Page load time of Google play site (단위 : 초)

	Chromium 브라우저								Firefox 브라우저							
무선 통신	3G 0.90Mbps		3G 1.37Mbps		WLAN 17.04Mbps		WLAN 61.12Mbps		3G 0.88Mbps		3G 1.46Mbps		WLAN 12.84Mbps		WLAN 25.96Mbps	
SPDY	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON
최소	5.72	2.27	5.76	2.29	1.31	1.27	1.40	1.20	5.89	5.43	8.64	7.91	8.73	7.55	5.36	5.59
최대	17.10	18.80	12.30	13.80	2.67	4.02	3.53	5.67	21.40	21.86	25.24	30.75	37.64	55.67	19.14	18.14
평균	7.21	2.72	6.90	2.75	1.62	1.64	1.65	1.62	7.38	6.53	11.34	8.77	11.06	10.24	7.14	6.46

표 3. 국내 사이트(2MB)에서 측정한 페이지 로드 타임
Table 3. Page load time of 2MB site (단위 : 초)

	Chromium 브라우저								Firefox 브라우저							
무선 통신	3G 0.90Mbps		3G 1.37Mbps		WLAN 17.04Mbps		WLAN 61.12Mbps		3G 0.88Mbps		3G 1.46Mbps		WLAN 12.84Mbps		WLAN 25.96Mbps	
SPDY	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON
최소	9.94	2.75	9.83	2.95	0.62	0.56	0.58	0.53	8.64	7.91	8.45	7.84	0.53	0.55	0.57	0.58
최대	21.3	25.4	19.7	16.3	3.55	4.85	1.39	7.3	25.24	30.75	20.50	23.50	4.48	4.88	1.84	2.35
평균	10.56	3.85	10.49	3.57	0.87	0.89	0.67	0.81	11.34	8.77	9.00	8.49	0.72	0.79	0.66	0.74

표 4. 국내 사이트(4MB)에서 측정한 페이지 로드 타임
Table 4. Page load time of 4MB site (단위 : 초)

	Chromium 브라우저								Firefox 브라우저							
무선 통신	3G 0.90Mbps		3G 1.37Mbps		WLAN 17.04Mbps		WLAN 61.12Mbps		3G 0.88Mbps		3G 1.46Mbps		WLAN 12.84Mbps		WLAN 25.96Mbps	
SPDY	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON
최소	6.81	3.12	9.83	2.95	9.52	2.61	0.62	0.56	8.73	7.55	5.36	5.59	8.45	7.84	8.01	7.53
최대	29.6	39.6	19.7	16.3	28	19.8	3.55	4.85	37.64	55.67	19.14	18.14	20.50	23.50	25.66	29.06
평균	10.36	4.50	10.49	3.57	10.33	3.63	0.87	0.89	11.06	10.24	7.14	6.46	9.00	8.49	8.97	8.37

각 사이트에 대한 자세한 정보는 표 1에 나타나있다.

성능 측정 시 보다 정확한 결과 값을 얻기 위하여 앞서 나뉜 모든 경우에 대해 페이지 로드 타임을 100번 이상씩 반복 측정하였으며, 새로운 경우에 대한 측정을 시작하기 전에는 인터넷 방문기록을 모두 삭제하였다.

측정값은 최솟값과 최댓값 그리고 평균값으로 나누어 나타내었다.

표 2는 Google Play 사이트에 대한 페이지 로드 타임 값이다. 먼저 Chromium 브라우저에서의 측정값들을 살펴보면, 최솟값의 경우 3G 환경에서는 SPDY 프로토콜을 사용하는 경우가 그렇지 않은 경우와 비교하여 60% 정도의 성능향상을 보였으나, 3G 환경보다 속도가 빠른 무선랜 환경에서는 SPDY 프로토콜이 뚜렷한 성능향상을 보이지 못했다. 최댓값은 대부분 웹페이지에 처음 접속하는 경우에 측정되었으며, SPDY 프로토콜을 사용하는 경우의 값이 그렇지 않은 경우에 비하여 10%에서 많게는 60%까지 증가하였다. 마지막으로

평균값은 3G 환경에서는 SPDY 프로토콜을 사용하는 경우가 기존 프로토콜을 사용하는 경우에 비하여 약 60%의 성능향상을 보였다. 그러나 최솟값의 경우와 마찬가지로 3G 환경보다 속도가 빠른 무선랜 환경에서는 SPDY 프로토콜이 기존 프로토콜과 비교하여 뚜렷한 성능향상을 보이지 못하였으며, 심지어 기존 프로토콜을 사용할 때의 성능이 더 좋은 경우도 있었다.

다음으로는 동일 사이트에 대한 Firefox 브라우저에서의 결과 값을 살펴보겠다. 최솟값의 경우 신호세기가 좋은 무선랜 환경을 제외하고는 Chromium 브라우저와 마찬가지로 SPDY 프로토콜을 사용할 때에 성능 향상을 보였다. 최댓값 역시 신호세기가 좋은 3G 환경을 제외하고는 Chromium 브라우저에서의 측정결과와 동일하게 SPDY 프로토콜을 사용했을 때의 페이지 로드 타임이 그렇지 않은 경우보다 1% 에서 6%까지 길게 측정되었다. 평균값의 경우에는 비교적 속도가 느린 3G 환경에서는 약 10% 정도의 성능향상을 보였으나 무선랜 환경에서는 성능 차이가 미미하거나 오히려 더 떨어지는 경우도 있었다.

표 3은 웹 페이지의 크기가 2MB인 국내 사이트에 성능 측정 결과 값이다. Chromium 브라우저에서 측정한 최솟값들을 보면, 3G환경에서는 SPDY 프로토콜을 사용한 경우가 그렇지 않은 경우와 비교하여 70% 이상의 뚜렷한 성능 향상이 있음을 알 수 있다. 그러나 무선랜 환경에서는 약 9%의 성능 향상을 보여 3G 환경과 비교했을 때 그 효과가 미미했다. 최댓값의 경우 신호세기가 좋은 3G 환경에서는 SPDY 프로토콜을 사용했을 때가 그렇지 않은 경우에 비하여 17% 정도의 성능 향상을 보였으나, 그 외의 환경에서는 SPDY 프로토콜을 사용할 때의 페이지 로드 타임이 사용하지 않을 때보다 적게는 20%에서 많게는 4배 이상까지 길게 측정되었다. 평균값은 3G 환경에서는 60% 이상의 성능 향상을 보였으나 무선랜 환경에서는 반대로 SPDY 프로토콜을 사용하지 않을 때의 성능이 더 좋게 측정되었다.

Firefox 브라우저에서의 성능 측정값을 살펴보면, 페이지 로드 타임 값 중 최솟값들은 정도의 차이는 존재하지만 Chromium 브라우저에서의 측정과 유사하게 3G 환경에서의 성능은 SPDY 프로토콜을 사용한 경우가 더 좋았다. 최댓값의 경우는 SPDY 프로토콜을 사용했을 때의 페이지 로드 타임이 그렇지 않을 때 보다 9%에서 많게는 28%까지 증가하였다. 평균값은 속도가 비교적 느린 3G 환경에서는 SPDY 프로토콜을 사용한 경우에 5%에서 23%의 성능 향상을 보였지만 무선랜 환경에서는 이와 반대로 SPDY 프로토콜을 사용한 경우에 12% 가량 성능이 나빠졌다.

표 4는 4MB 크기의 국내 사이트에 대한 페이지 로드 타임 값이다. Chromium 브라우저에서 측정한 결과 값의 경우, 최솟값은 3G 환경에서는 SPDY를 사용한 경우에 50%에서 70%정도의 성능 향상을 보였지만 무선랜 환경에서는 성능 차이가 5%정도로 미미하거나 같았다. 최댓값은 신호세기가 좋은 3G 환경을 제외하고는 SPDY 프로토콜을 사용하는 경우에 34%에서 많게는 다섯 배까지 더 길게 측정되었다. 평균값은 3G 환경에서는 SPDY 프로토콜을 사용한 경우에 60% 정도의 성능 향상을 보였으나 무선랜 환경에서는 정 반대의 경우가 관찰되었다. SPDY 프로토콜을 사용한 경우가 그렇지 않을 때 보다 64%에서 80%까지 성능 감소를 보였다.

다음으로 Firefox 브라우저에서의 측정값을 보면, 최솟값은3G 환경의 경우 SPDY 프로토콜을 사용한 경우에 성능 향상을 보이지만 무선랜 환경에서는 성능 차이가 없거나 오히려 더 나빠짐을 확인할 수 있었다. 최댓값은 앞의 측정결과들과 유사하게 SPDY 프로토콜을 사용하는 경우의 페이지 로드 타임이 13%에서 48%까지 길게 측정되었다. 마지막으로 평균값의 경우에는 3G 환경에서는 SPDY 프로토콜을 사용한 경우에 약 7%의 성능 향상을 보였지만 무선랜 환경에서는 그 차이가 같거나 오히려 SPDY 프로토콜을 사용하지 않은 경우의 성능이 더 좋게 측정되었다.

2. 초기 접속시간 성능 분석

앞의 측정결과로 우리는 대부분의 경우에서 페이지 로드 타임의 최댓값은 SPDY 프로토콜을 사용한 경우가 SPDY 프로토콜을 사용하지 않은 경우보다 길게 측정됨을 알 수 있었으며 페이지 로드 타임의 최댓값은 거의 모든 경우에 처음 웹 페이지에 접속 할 때 측정됨을 확인할 수 있었다. 일반적으로 인터넷 사용자는 하나의 웹 페이지에 반복적으로 접속하기보다는 여러 개의 웹 페이지를 이동하며 인터넷을 이용한다. 따라서 다음으로는 실제 사용자가 인터넷을 이용하는 패턴과 유사하게 각 웹사이트들에 한 번씩 접속하며 이때 웹 페이지를 불러오는데에 소요되는 시간을 측정하는 방법으로 초기 접속시간에 대한 성능평가를 진행하였다. 성능 측정을 위하여 SPDY를 지원하는 20개 사이트를 선별하였으며 그 목록은 표 5와 같다.

표 5에 명시된 20개의 SPDY 지원 사이트 중 웹 페이지의 크기가 작은 상위 10개 사이트에 대해서는 신호세기가 좋지 않은 3G 환경, 웹 페이지의 크기가 큰 하위 10개 사이트에 대해서는 신호세기가 좋은 무선랜 환경에서 각각 성능 측정을 진행하였다. 측정은 10개의 웹 사이트에 차례대로 한 번씩 접속하는 방식으로 이루어졌다. 이 때 SPDY를 켠 상태와

SPDY를 끈 상태에 대하여 Chromium 브라우저와 Firefox 브라우저에서 번갈아 가며 측정을 진행하였으며 보다 정확한 측정을 위하여 새로운 측정이 진행 될 때마다 인터넷 사용기록을 삭제하고 랩탑 또한 재부팅 하였다.

표 6은 페이지의 크기가 작은 상위 10개 사이트의 페이지 로드 타임을 신호세기가 좋지 않은 3G 환경에서 측정한 결과 값이다. Chromium 브라우저에서의 측정값을 살펴보면 첫 번째로 측정된 하나의 사이트를 제외하고는 SPDY 프로토콜을 사용할 때의 초기 접속 시간이 SPDY 프로토콜을 사용하지 않을 때 보다 많게는 두 배 이상 증가했다. Firefox 브라우저에서의 측정값에서도 증가량의 차이는 조금 있으나 비슷한 결과를 볼 수 있다. 두 번째와 여섯 번째로 측정한 두 개의 사이트를 제외하고는 역시 SPDY 프로토콜을 사용할 때의 초기접속시간이 그렇지 않을 때에 비해 증가하였다.

표 7은 페이지의 크기가 큰 하위 10개 사이트의 페이지 로드 타임을 신호세기가 좋은 무선랜 환경에서 측정한 결과 값이다. Chromium 브라우저를 통해 측정한 값은 두 번째와 세 번째 그리고 다섯 번째로 측정된 세 개의 사이트를 제외하고는 SPDY 프로토콜을 사용한 경우의 초기접속시간이 SPDY 프로토콜을 사용하지 않을 때와 비교하여 적게는 1%에서 많게는 49%까지 증가하였다. Firefox 브라우저를 이용하여 성능 측정을 진행한 결과 값에서도 Chromium 브라우저에서의 측정 결과와 비교하여 시간차이의 폭이 크지는 않았지만 역시 두 번째와 세 번째 여덟 번째로 측정된 세 개의 사이트를 제외하고는 SPDY 프로토콜을 사용한 경우의 초기접속시간이 SPDY 프로토콜을 사용하지 않을 때보다 증가하였다.

표 5. SPDY 지원 웹 사이트 목록
Table 5. Information of website supporting SPDY protocol

사이트	Object 수	Domain 수	페이지 크기 (KB)
google plus	7	4	41.84
gmail	11	4	51.46
Igvita.com	12	6	136.46
webtide	18	2	177.54
Erianna	19	3	217
google shopping	15	5	264.04
WordPress	17	11	268.25
Twitter	12	3	288.23
CloudFlare	32	4	445.68
Lilp Blog	25	4	648.77
YouTube	60	13	735.65
Google news	53	6	740.46
Google maps	53	5	836.92
Is the spdy protocol?	31	12	1002.91
Google play	52	17	1592.43
Humble bundle	99	24	1630.22
Tech Crunch	174	55	1880.33
Anizon	90	4	2353.93
VIP word press	57	22	3802.77
gigaom	113	39	5499.83

표 6. 3G 환경에서 측정한 10개 사이트의 초기 접속 시간
Table 6. Initial connection time of 10 sites in 3G network (단위 : 초)

	SITE 1		SITE 2		SITE 3		SITE 4		SITE 5		SITE 6		SITE 7		SITE 8		SITE 9		SITE 10	
SPDY	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON
Chromium	3.86	3.97	2.37	1.87	6.76	9.38	9.82	11.33	8.45	8.95	11.70	11.83	5.08	5.83	10.64	12.10	10.71	24.00	13.40	17.25
Firefox	5.13	6.00	3.74	3.58	6.51	6.65	8.47	8.68	7.49	8.46	13.14	10.98	5.51	5.81	8.70	8.87	15.48	19.66	11.58	11.58

표 7. 무선랜 환경에서 측정한 10개 사이트의 초기 접속 시간
Table 7. Initial connection time of 10 sites in WLAN network (단위 : 초)

	SITE 1		SITE 2		SITE 3		SITE 4		SITE 5		SITE 6		SITE 7		SITE 8		SITE 9		SITE 10	
SPDY	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON
Chromium	2.46	3.67	1.71	1.53	3.39	2.15	3.40	4.22	2.73	1.72	8.29	9.65	6.89	7.47	3.36	3.38	7.63	7.77	9.44	12.55
Firefox	4.45	4.70	2.20	2.10	4.01	3.76	4.71	4.74	3.26	3.69	6.60	6.98	8.14	8.54	2.24	2.04	5.77	5.99	7.95	8.01

3. WiBro 환경에서의 측정

마지막으로 이동 시에 SPDY프로토콜을 사용할 때와 사용하지 않을 때의 성능 비교를 위하여 버스를 타고 이동하며 성능측정을 진행하였다. 그림 1의 지도에서 위쪽에 표시된 경로를 이동하며 SPDY 프로토콜을 켠 경우의 성능 측정을 진행하였으며 아래쪽에 표시된 경로로 되돌아오며 SPDY 프로토콜을 껐을 때의 성능측정을 진행하였다. 이 때 일정한 시간간격으로 성능 측정을 진행하였으나 전철역으로 이동하는 경로와 학교로 되돌아오는 경로상의 거리 차이가 있어 SPDY 프로토콜을 켠 상태는 63회 SPDY 프로토콜을 끈 상태에서는 93회 반복 측정으로 동일한 횟수의 측정을 진행하지는 못하였다.

그림 1. WiBro 환경에서의 성능 측정 시 이동 경로
Fig. 1. Path of performance evaluation in WiBro network environment

측정 웹 페이지로는 앞의 두 실험에서도 언급된 Google Play를 선정하였으며 이번 실험 역시 새로운 경우에 대한 측정을 시작하기 전에는 인터넷 방문기록을 모두 삭제하였다.

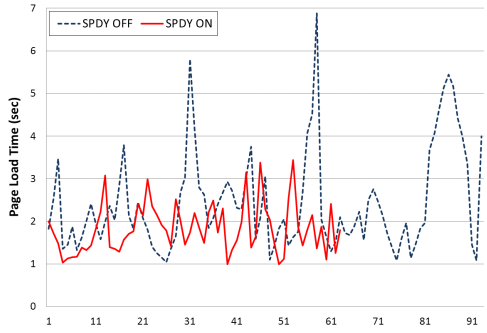

그림 2. WiBro 환경에서의 성능 측정 그래프
Fig. 2.Evaluation result in WiBro network environment

이에 대한 결과 그래프는 그림 2와 같다. 고정된 장소에서 진행한 앞의 두 성능 측정 실험에 비교하여 마지막으로 진행한 이동체에서의 성능 측정 에서는 네트워크 환경이 안정적이지 못했으며 결과값 역시 이런 환경의 영향을 많이 받아 측정값이 크게 증가 하는 경우가 종종 나타났다.

그림 2의 그래프에서 중간 중간 접속이 원활하지 못하여 값이 크게 나온 지점을 제외하고는 SPDY 프로토콜을 사용할 때가 SPDY 프로토콜을 사용하지 않을 때와 비교하여 뚜렷한 성능향상을 보이지는 않음을 확인할 수 있다.

V. SPDY Flow Control 의 성능 분석

다음으로는 기존 프로토콜과의 차이점이며 SPDY 프로토콜의 특징 중 하나인 Flow Control에 대한 보다 심층적인 성능 분석을 진행하였다. 이는 앞서의 분석을 통해서 SPDY의 동작이 경우에 따라 HTTP보다 좋을 수도 있고 나쁠 수도 있음을 확인하였으며 이 때 성능에 영향을 미치는 중요한 요인 중의 하나이자 4계층의 TCP와 독립적으로 동작하는 SPDY의 Flow Control의 영향을 파악하기 위함이다.

1. SPDY Flow Control 동작 분석

SPDY는 TCP 단의 Flow Control과 별개로 독자적인 Flow Control을 제공하여 다수의 스트림을 동시에 생성함으로써 발생할 수 있는 문제들을 미연에 방지한다. SPDY가 제공하는 Flow Control의 동작방식은 구글의 SPDY 규격 문서와 오픈 소스 소프트웨어 소스 코드의 분석을 통하여 그림 3~6처럼 도출하였다.

그림 3과 4는 Apache용 SPDY 서버인 mod_spdy와 구글의 Chromium 브라우저에서 Flow Control이 수행될 때 발생하는 메시지에 대한 시퀀스 차트이다. 현재 mod_spdy와 Chromium 모두 오픈 소스로 제공되고 있어 소스코드를 중심으로 분석하여 차트를 그렸으며 코드의 변수 명을 차트에 그대로 사용하였다. 또한 표준문서에도 언급된 변수의 경우에는 표준에서 사용된 명칭을 괄호 안에 명시해주었다.

그 중 그림 3은 mod_spdy서버에서 Chromium 브라우저로 파일전송이 발생하는 경우의 시퀀스 차트이다. 박스 안의 kSpdyStreamInitialWindowSize는 송신 윈도우 사이즈를 나타내주는 변수로 Chromium과 mod_spdy 모두 이를 64KB로 초기화 하고 있다. 그러나 Chromium의 수신 윈도우 사이즈는 10MB로 초기화되어 표준문서에서 언급된 크기인 64KB와는 상이함을 확인할 수 있었다. 처음 SPDY 세션

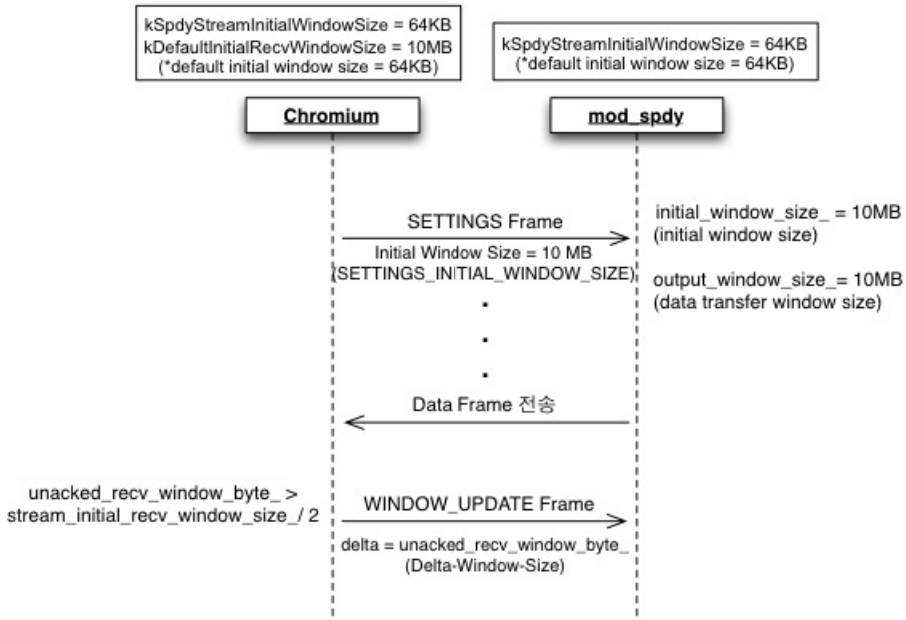

그림 3. mod_spdy에서 Chromium으로 파일 전송 시 브라우저와 서버 간 Flow Control 시퀀스 차트
Fig. 3. Flow Control Sequence Chart when sending file from mod_spdy to Chromium

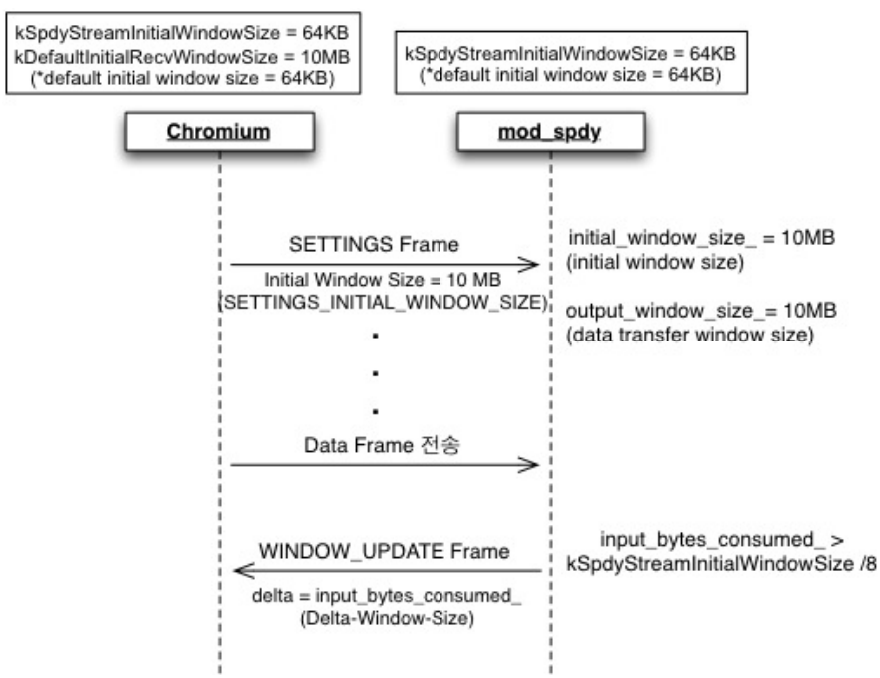

그림 4 Chromium에서 mod_spdy로의 파일 전송 시 브라우저와 서버 간 Flow Control 시퀀스 차트
Fig. 4. Flow Control Sequence Chart when sending file from Chromium to mod_spdy

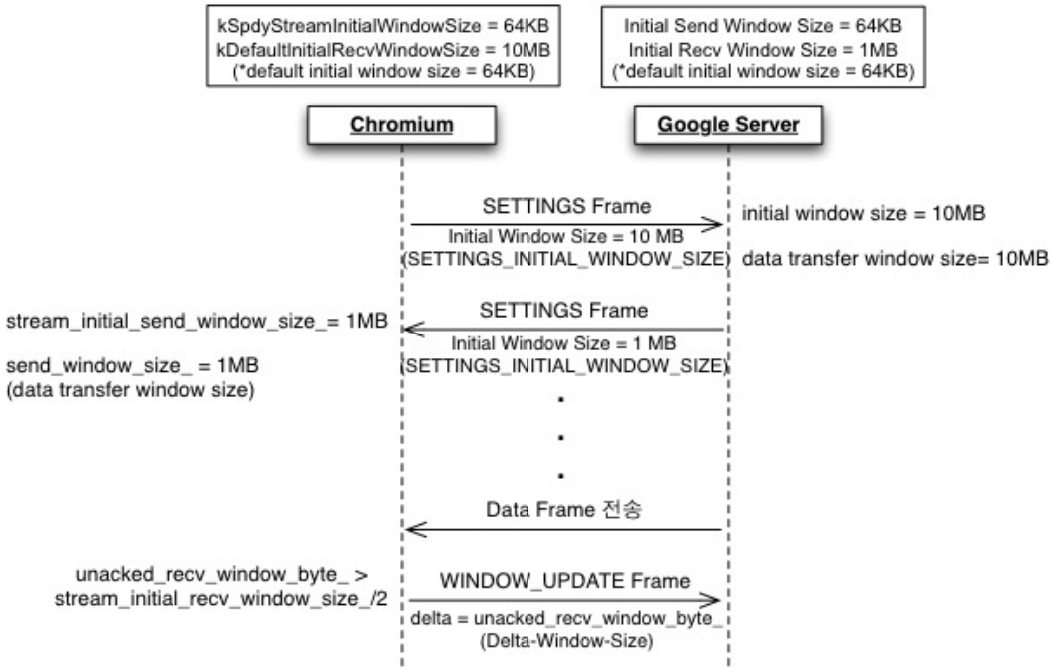

그림 5 구글 서버에서 Chromium으로 파일 전송 시 브라우저와 서버 간 Flow Control 시퀀스 차트
Fig. 5. Flow Control Sequence Chart when sending file from Google Server to Chromium

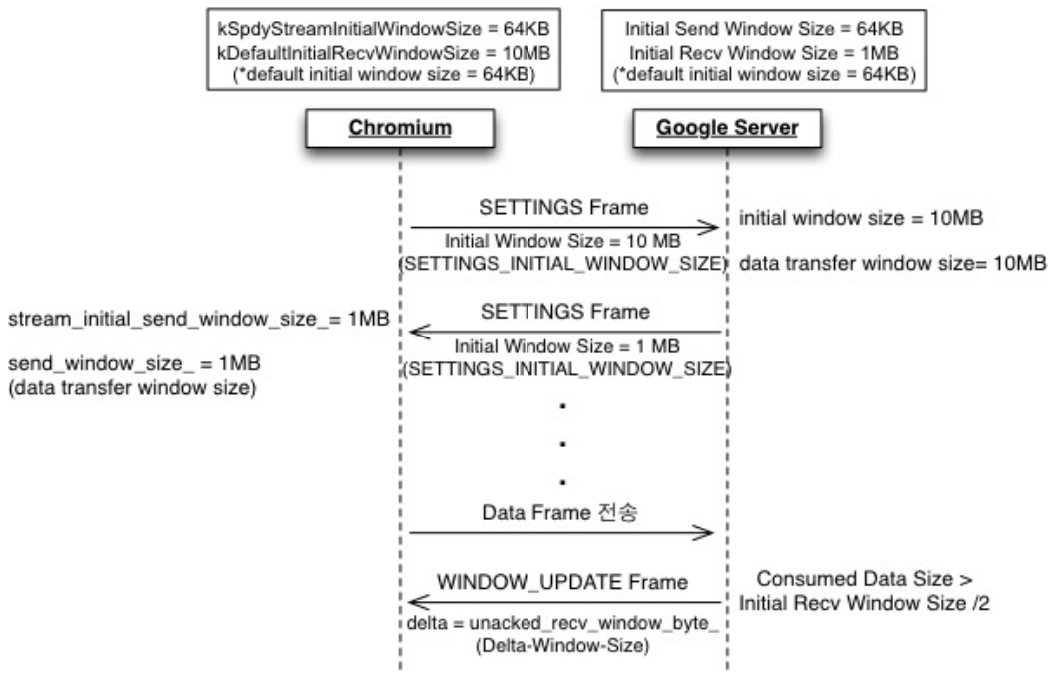

그림 6 Chromium에서 구글 서버로 파일 전송 시 브라우저와 서버 간 Flow Control 시퀀스 차트
Fig. 6. Flow Control Sequence Chart when sending file from Chromium to Google Server

이 확립되면 Chromium은 서버로 SETTINGS 프레임을 전송하며 자신의 수신 윈도우 사이즈를 알린다. 프레임을 수신한 서버는 자신의 초기 송신 윈도우 사이즈와 프레임 수신 이전에 생성된 스트림들의 송신윈도우 사이즈를 모두 클라이언트의 수신 윈도우 사이즈에 맞게 변경한 후에 스트림 생성 및 데이터의 전송을 진행한다. Chromium은 수신 받은 데이터의 크기가 자신의 수신 윈도우사이즈의 1/2보다 커지면 WINDOW UPDATE 프레임을 발생시켜 delta값만큼 윈도우의 크기를 확장시키는 방법으로 Flow Control을 진행하며 이때 delta값으로는 수신 받은 데이터의 크기가 사용된다. 그림 4는 Chromium에서 mod_spdy로 파일전송이 발생하는 경우이다. 전체적인 흐름은 앞의 (a)와 동일하며 mod_spdy는 수신된 데이터의 크기가 자신의 수신 윈도우 사이즈의 1/8 이상일 때 WINDOW UPDATE 프레임을 발생시키는 방식으로 Flow Control을 진행한다는 차이점이 존재한다.

그림 5와 6은 실제 구글의 서버와 Chromium 브라우저 간의 Flow Control 시퀀스 차트이다. 구글 서버는 소스코드가 공개되어있지 않으므로 실제 주고받는 프레임 분석을 중심으로 하여 시퀀스차트를 작성하였다. 공개된 mod_spdy 코드와 달리 구글 서버는 별도로 서버의 송신윈도우 사이즈는 64KB, 수신 윈도우 사이즈는 1MB로 정의하고 있었다. Chromium으로부터 SETTINGS 프레임을 수신하면 서버 역시 SETTINGS 프레임을 전송하며 자신의 수신윈도우 사이즈를 알린다. 그림 5는 구글 서버에서 파일을 전송하는 경우, 그림 6은 Chromium 에서 파일을 전송하는 경우의 시퀀스 차트이며 이 후에 진행되는 과정들은 그림 3과 4와 동일하다.

2. Flow Control 성능 측정 방법

다음으로는 SPDY의 Flow Control이 효과적으로 동작하는지 알아보기 위해 자체적으로 구성한 테스트베드에서 성능 측정 실험을 진행하였다. 측정에 사용된 테스트베드는 그림 7에 나타나있다.

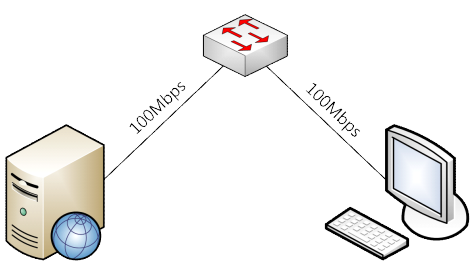

그림 7 Flow Control 측정 테스트베드
Fig. 7. Testbed for evaluating Flow Control

서버는 x86 아키텍처 기반의 PC에 Ubuntu 12.04를 탑재한 후 Apache2.2과 SPDY의 서버 버전인 mod_spdy를 설치하여 구성하였으며 이 때 설치된 SPDY의 버전은 3이다. 클라이언트로는 소스코드를 직접 빌드 한 Chromium 브라우저를 사용하였으며 SPDY의 윈도우 사이즈를 조절하기 위하여 필요에 따라 코드를 수정 후 다시 빌드하였다. 서버와 클라이언트는 100Mbps의 이더넷 스위치로 연결해주었다. 테스트베드 위에는 표 8과 같이 크기가 다양한 네 개의 웹사이트를 올린 후 이를 반복 접속하면서 페이지 로드 타임 값을 측정하였으며 정확한 결과를 얻기 위하여 새로운 경우에 대한 측정을 시작하기 전에는 인터넷 방문기록을 모두 삭제하였으며 서버 또한 다시 시작하였다.

표 8. Flow Control 관련 측정사이트 목록
Table 8. Information of website for evaluating Flow Control

사이트	Object 수	페이지 크기 (KB)
SITE1	20	127.3
SITE2	1	870
SITE3	1	18,506.47
SITE4	3	42,891.83

성능 실험 시 SPDY의 윈도우 사이즈는 10MB와 100MB의 두 가지로 나누었으며 각각의 경우에 대해 TCP의 윈도우 사이즈 역시 세분화하여 측정을 진행하였다. 링크의 속도 또한 10Mbps와 100Mbps두 가지로 변화를 주었다.

3. Flow Control 성능 측정 결과

표 9는 링크 속도 10Mbps, Ping delay 0.5ms이하에서의 성능 측정 결과이다. 왼쪽부터 차례대로 Site 1, 2, 3, 4에 대한 측정 결과이며 이를 쉽게 볼 수 있도록 SPDY를 사용했을 때 성능이 더 나빠진 결과 값은 빗금으로 표시하였다. SPDY 프로토콜을 사용했을 때 성능 개선을 보인 경우는 드물었으며 향상 폭 또한 최대 7%에 그치는 등 미미한 수준이었다. 특히 Site1을 제외하고는 거의 모든 경우에서 SPDY 프로토콜을 사용할 때의 성능이 기존 프로토콜을 사용할 때에 비하여 더 나빠졌으며 약 12%까지 성능 감소가 발생함을 확인할 수 있었다.

표 10은 링크 속도 100Mbps, Ping delay 1ms 이하에서의 성능 측정 결과이다. 앞서 살펴 본 표 9와 마찬가지로 이번 측정에서도 역시 Site1에서의 결과 값들을 제외하고는 대부분 SPDY 프로토콜을 사용하는 경우에 성능 저하가 발생

표 9. 전송 속도 10Mbps 환경에서의 성능 측정 결과
Table 9. Evaluation result in 10Mbps link speed (단위 : 초)

TCP Window Size	SITE 1			SITE 2			SITE 3			SITE 4		
	SPDY Initial Window Size		Non SPDY	SPDY Initial Window Size		Non SPDY	SPDY Initial Window Size		Non SPDY	SPDY Initial Window Size		Non SPDY
	64MB	10MB	OFF	64MB	10MB	OFF	64MB	10MB	OFF	64MB	10MB	OFF
64KB	1.36	1.41	1.46	1.94	1.98	1.93	17.65	17.60	17.45	39.20	40.05	38.60
128KB	1.39	1.35	1.40	1.94	1.97	1.92	17.65	18.10	17.40	39.30	39.55	38.70
256KB	1.36	1.50	1.40	1.94	1.94	1.92	17.75	17.65	17.35	39.10	39.45	38.70
512KB	1.35	1.37	1.40	1.94	1.99	1.93	17.60	17.65	17.40	39.10	39.25	38.70
1MB	1.44	1.60	1.44	1.95	1.94	1.93	17.65	17.50	17.45	39.05	39.20	38.65
2MB	1.41	1.37	1.41	1.94	1.95	1.93	17.65	17.50	17.50	39.10	39.15	38.70
4MB	1.42	1.41	1.44	1.94	2.00	1.92	17.60	17.55	17.65	39.25	39.10	38.75
8MB	1.36	1.37	1.41	1.94	1.96	1.93	17.75	17.75	17.50	39.10	39.10	38.65
16MB	1.36	1.36	1.44	1.95	2.04	1.93	17.65	17.55	17.45	39.15	39.10	38.70
32MB	1.38	1.36	1.46	1.95	1.94	1.93	17.60	17.55	17.35	39.35	39.10	39.30

표 10. 전송 속도 100Mbps 환경에서의 성능 측정 결과
Table 10. Evaluation result in 100Mbps link speed (단위 : 초)

TCP Window Size	SITE 1			SITE 2			SITE 3			SITE 4		
	SPDY Initial Window Size		Non SPDY	SPDY Initial Window Size		Non SPDY	SPDY Initial Window Size		Non SPDY	SPDY Initial Window Size		Non SPDY
	64MB	10MB	OFF	64MB	10MB	OFF	64MB	10MB	OFF	64MB	10MB	OFF
64KB	1.40	1.40	1.56	1.46	1.46	1.34	8.84	8.84	8.68	9.75	10.02	9.40
128KB	1.42	1.45	1.43	1.63	1.66	1.33	8.57	8.54	8.82	9.73	9.41	9.05
256KB	1.39	1.46	1.53	1.50	1.53	1.34	8.72	8.70	8.66	9.79	8.99	9.01
512KB	1.41	1.41	1.48	1.46	1.40	1.36	8.88	8.92	8.72	9.68	8.81	8.90
1MB	1.41	1.46	1.42	1.60	1.60	1.32	8.83	8.45	8.66	9.79	9.13	9.05
2MB	1.41	1.41	1.48	1.65	1.59	1.32	8.87	8.83	8.76	9.73	9.02	8.93
4MB	1.40	1.48	1.54	1.58	1.64	1.33	8.80	8.74	8.65	9.72	9.03	8.92
8MB	1.42	1.42	1.45	1.57	1.64	1.33	8.88	8.91	8.73	9.73	9.05	9.00
16MB	1.43	1.41	1.44	1.52	1.58	1.37	8.87	8.85	8.81	9.73	8.85	8.94
32MB	1.57	1.43	1.49	1.59	1.53	1.33	8.88	8.90	8.73	9.86	9.08	8.96

했으며 성능 차이 또한 약 25%까지 크게 발생하는 등 10Mbps에서의 실험 결과보다도 더 큰 성능 감소를 보였다.

Ⅵ. 결론

SPDY는 웹 페이지의 로드 타임을 줄이기 위한 목적으로 구글에서 새롭게 제안한 프로토콜이다. 그러나 SPDY 프로토콜은 구체적인 성능이 분석되지 않은 상황에서 오픈 소스 소프트웨어가 확산되고 있으며 따라서 정성/정량적인 장단점은 규정되지 않고 있다. 본 논문은 다양한 환경에서 성능측정 실험을 실시하여 SPDY 프로토콜과 기존 프로토콜의 차이점을 비교 분석하고 SPDY 프로토콜의 특징과 문제점을 파악하였다.

본 논문에서 진행된 성능 측정 실험을 통하여, 첫째로 우리는 데이터 전송률이 비교적 낮은 3G 환경의 고정 단말의 통신에서 SPDY 프로토콜을 이용하여 인터넷을 사용하는 경우에 페이지 로드 타임은 기존의 프로토콜을 사용할 때와 비교하여 10%에서 60% 정도 감소함을 확인하였으며 이는 구글의 초기 SPDY 프로토콜 개발 목표에 부합한다.

그러나 둘째로 기존 HTTP 프로토콜에 비해 길고 복잡해진 초기 연결 설정은 웹 페이지에 처음 접속할 때 걸리는 시간의 증가를 가져왔으며 이는 웹 페이지에 대한 반복접속보다 초기 접속이 많은 인터넷 사용패턴을 고려할 때 반드시 해결되어야 할 문제점이라 할 수 있다.

셋째로 데이터 전송률이 높은 무선랜 환경에서는 SPDY 프로토콜이 뚜렷한 성능향상을 보이지 못했으며 성능이 나빠지는 경우도 종종 있었는데 LTE, 5G 등 고속 데이터 통신의 시대가 도래한 만큼 빠른 데이터 전송 환경 에서도 성능향상을 보일 수 있는 개선책이 필요하다.

넷째로 이동체에서 아직까지 뚜렷한 성능 향상을 보이지 못하는 점 역시 앞으로 SPDY 프로토콜이 보완해 나가야 할 문제점이라 하겠다.

여섯째로 TCP 레벨의 Flow Control과는 별개로 SPDY에서 독자적으로 진행되는 Flow Control은 일반적인 경우 성능향상 가져오기 보다는 오히려 이로 인한 오버헤드를 증가시켜 성능의 감소를 야기하는 것으로 보이며 이 또한 앞으로 개선되어야 할 부분이다. 특히 성능 분석 결과 SPDY의 Flow Control은 사실상 기능을 비활성화 시켜야 SPDY의 성능 개선이 나타나는 상황이다. 따라서 앞으로 Flow Control에 대한 큰 기술개선이 필요해 보인다.

일곱째로 SPDY의 규격문서와 실제 구글 오픈 소스 코드는 부합되지 않는 면들이 있다. 구글의 경우 규격과 다른 흐름제어 버퍼 크기들을 사용하는 등 표준 규격과는 상이한 동작을 구현하고 있기에 이에 대해 보다 엄밀한 규격 인증도 필요하다.

반면 사용자 입장이 아닌 서버의 측면에서 살펴보면 다음과 같은 이점을 확인할 수 있다. SPDY는 클라이언트와 서버의 TCP 연결 개수를 여러 개에서 하나로 줄여 유지함으로써 서버의 프로세싱 및 메모리 부하를 감소시킨다. 또한 서버의 SPDY 수신버퍼 사이즈를 조절하여 클라이언트로부터의 트래픽 전송 속도를 조절 하고 서버에서 클라이언트로의 Server Push를 가능하게 한다. 앞의 결과들을 종합해 볼 때 SPDY는 아직까지 사용자 측면에서의 전송 속도 향상 보다는 서버 관리자 관점에서의 성능 개선에서 더 큰 이점을 얻고 있는 것으로 보인다.

본 논문에서 나타난 프로토콜의 문제점과 언급된 개선방향들에 대한 해결책이 제시된다면 좀 더 나아진 SPDY 프로토콜을 기대해볼 수 있으리라 생각된다.

참고문헌

[1] Strangeloop, “State of the Union:Ecommerce Page Speed&Web Performance,” pp.7-11, 2012.

[2] Belshe, M., Thomson, M., Melnikov, A., & Peon, R., “Hypertext Transfer Protocol version 2.0”, draft-ietf-httpbis-http2-05, August 2013.

[3] Trace, R., Foresti, A. Singhal, S., Mazahir, O., Nielsen, H., Montenegro, G.,"HTTP Speed+ Mobility", draft-montenegro-httpbis-speed-mobility-01, Mar 2012.

[4] SPDY: An experimental protocol for a faster web, http://www.chromium.org/spdy/spdy-whitepaper

[5] Belshe, M. and R. Peon, "SPDY Protocol", draft-mbelshe-httpbis-spdy-00, February 2012.

[6] White, G., "Analysis of Google SPDY and TCP initcwnd", May 2012, http://www.cablelabs.com/downloads/pubs/Analysis_of_Google_SPDY_TCP.pdf

[7] SPDY Performance on Mobile Networks, https://developers.google.com/speed/articles/spdy-for-mmobil?hl=ko

저 자 소 개

김 희 정

2012: 경희대학교
컴퓨터공학과 공학사.
2012~현재: 경희대학교
컴퓨터공학과 석사과정.
관심분야: 컴퓨터네트워크,
네트워크 성능분석,
통신 프로토콜
Email : heejung0310@gmail.com

이 규 선

2012: 경희대학교
컴퓨터공학과 공학사.
2012~현재: 경희대학교
컴퓨터공학과 석사과정.
관심분야: 이동 단말기 솔루션,
이동통신 서비스,
이종망 네트워크,
소프트웨어 정의 네트워크
(SDN), 플로우 제어
Email : lysters@gmail.com

이 성 원

1998: 경희대학교
컴퓨터공학과 공학박사.
1999~2008: 삼성전자 정보통신총괄
네트워크사업부/통신연구소
2008~현재: 경희대학교
컴퓨터공학과 교수.
관심분야: 무선광대역통신,
컴퓨터 네트워크,
미래인터넷,
M2M/MTC 통신시스템
Email : drsungwon@khu.ac.kr