SPDY Accelerator for Improving Web Access Speed

Gen Mineki, Satoshi Uemura, and Teruyuki Hasegawa

KDDI R&D Laboratories Inc., 2-1-15 Ohara, Fujimino-shi, Saitama, 356-8502 Japan

E-mail: {ge-mineki, uemura, teru}@kddilabs.jp

***Abstract*—It was decided that the next-generation protocol SPDY proposed by Google would be used as a basis of the technical specification of the HTTP/2.0 protocol, which has been studied in the IETF standardization process. SPDY is a protocol to realize high-speed Web access by using the SPDY session that has been established between the client and the Web server for transmitting and receiving page resources. Since a modern Web page usually consists of multiple page resources that are stored in multiple domains (multi-domain configuration), the client has to establish multiple SPDY sessions with multiple Web servers. In this case, SPDY is not able to realize high-speed Web access since it takes several seconds to establish multiple SPDY sessions in a mobile environment with high latency. In this paper, we propose a SPDY accelerator that can considerably accelerate Web access speed by combining the SPDY protocol and cache system even in a multi-domain configuration. We confirmed that the proposed system can reduce the page-loading time by one-third compared to the existing SPDY.**

***Keywords*—SPDY, Web contents delivery, cache, reverse proxy**

I. INTRODUCTION

Mobile Internet, referred to as Internet access with mobile terminals via cellular networks, has enabled us to enjoy various communication services through the Internet anytime and anywhere. In particular, in accordance the explosive growth of smartphone users, Web access based on HTTP using a Web browser has become an indispensable service, taking their usage trend into account [1]. Following extensive research, it is found that user satisfaction with her/his Web access heavily depends on the page-loading time, which is defined as the time interval from sending a request to displaying the corresponding Web page [2]. Although "8 seconds [3]" is a well-known threshold below which users will wait, a shorter page-loading time is now eagerly anticipated as cellular networks become broadband. For example, the literature [11] reported that an e-commerce site will lose 1.6 billion dollars if its page-loading time is increased by only 1 second. Since "making Web access faster in order to shorten the page-loading time" is a crucial issue for not only content providers but also mobile operators, there have been various research and development activities [8][9] to tackle this issue.

Google has proposed SPDY (pronounced speedy) [4] as a next-generation protocol for Web access. At the 84th IETF meeting, the HTTPbis Working Group officially decided that HTTP/2.0, the next version of the HTTP protocol, would be standardized based on SPDY. In SPDY, a client establishes a SPDY session with a Web server, which is identified by FQDN (fully qualified domain name; we call it simply "domain"), in order to exchange the content through the session. In other words, a SPDY session is established by the client with each domain. On the other hand, a modern Web page usually consists of multiple page resources that are stored in multiple domains, where, for example, a main HTML file and other files such as image and movie files (page resources) are stored separately in different domains. In this case, the client has to establish multiple SPDY sessions with multiple Web servers. Therefore, SPDY would not be able to realize high-speed Web access since it takes several seconds to establish multiple SPDY sessions in mobile Internet with high latency.

In this paper, we propose a novel method of realizing faster Web access with SPDY even in a multi-domain configuration by reducing the number of SPDY sessions utilizing cache systems. We newly introduce a SPDY accelerator as a proxy cache system between Web servers and clients. When a client requests a main HTML file on a Web server, the SPDY accelerator analyzes it online. If the URIs (unified resource identifiers) of page resources listed in the main HTML file belong to different domains, the SPDY accelerator rewrites them as if these page resources are stored in the same domain as the main HTML file. Since the client considers that all the page resources are stored in a single domain and can thus obtain them through a single SPDY session, faster Web access can be achieved even in such multi-domain configurations.

The rest of this paper is organized as follows. Section II summarizes some related works to improve page-loading time: extensions of existing HTTP, frontend optimization, and also SPDY to which our proposal applies. Section III explains the details of our proposal where the SPDY protocol is combined with a cache system. Section IV shows the results of our performance evaluation using our prototype implementation to verify the effectiveness of our proposal. Section V concludes this paper and envisions our future works.

II. RELATED WORK

There are several approaches to improving the page-loading time of a given Web page. Here, we summarize conventional approaches from the viewpoints of enhancing the protocol and frontend optimization.

A. HTTP/1.0 and HTTP/1.1

The HTTP protocol is generally utilized when a client communicates with a Web server to obtain a main HTML file and its page resources composing the Web page such as image files, style sheets, and JavaScript. In HTTP/1.0 [5], the client has to establish a new TCP connection with the Web server whenever the client is about to obtain a page resource from the server. In the case where a number of page resources are included in the Web page, the client has to establish and terminate the TCP connection with the Web server iteratively. Since establishing/terminating the TCP connection wastes time, it is difficult for the client to obtain all the page resources efficiently in a short period of time. Since the client can establish N_{tcp} TCP connections with one Web server, if the number of page resources to be obtained from a certain Web server is fewer than N_{tcp}, the above problem hardly occurs. However, as Web access becomes a popular and indispensable communication service, a Web page tends to include various types and a number of page resources. The literature [6] reports that one modern Web page includes 44 page resources on average. Therefore, in the case where the client utilizes HTTP/1.0, it is difficult to achieve high-speed Web access, since there is an inherent bottleneck caused by iterative establishment/termination of TCP connection.

In HTTP/1.1 [7], on the other hand, since the client can use a *persistent connection* that enables the client to keep one TCP connection, through which the client can obtain several page resources from a certain Web server. Thanks to the persistent connection, the overhead caused by iterative establishments/terminations of the TCP connection can be considerably reduced. When the client obtains several page resources from a certain Web server using this persistent TCP connection, the client sends a GET request to the Web server and then receives a response to the request from the Web server. After that, the client sends a GET request for the next page resource. Although the client is also allowed to send multiple GET requests to the Web server continuously without receiving a response, i.e., *HTTP pipelining*, the client has to receive one response after another.

B. Frontend optimization

In order to improve the page-loading time of a given Web page, frontend optimization is effective [8]. Here, three typical techniques, domain sharding, use of a content delivery network (CDN), and CSS sprite, are summarized.

1) Domain sharding

As described in the previous subsection, the existing HTTP protocol allows the client to establish multiple TCP connections whose maximum number is N_{tcp} with one Web server. Thus, by splitting the page resources of the Web page across N_{dom} domains of Web servers, the client can obtain N_{tcp} x N_{dom} page resources concurrently by establishing N_{tcp} x N_{dom} TCP connections. Since the number of DNS lookups increase as the number of sharded domains increases, the page-loading time is not always reduced when the number of domain shardings increases. In the literature [8][9], although the optimum number of domain shardings depends on the number of page resources and their volume, it is empirically known that using two domains yields better results.

2) Use of CDN

The round-trip time (RTT) deeply affects the time needed to establish a TCP connection between the client and the Web server. The RTT also affects the downloading time of the page resources. Thus, it follows that the page-loading time can be reduced if the client communicates with the Web server in a shorter RTT. In the CDN, the duplicate of the content originally stored in the Web server is stored in the cache servers located near clients: note that the server that possesses original content is called the "origin server" and the server located near clients is called the "edge server." When a request is generated from a certain client, the CDN processes the request so that the client can communicate with the nearest edge server in the minimum RTT. Therefore, by using CDN, the page-loading time of the client can be reduced.

3) CSS sprite

CSS sprite [10] is a technique to show a part of an image by using the background position of CSS after consolidating multiple small image files such as icons into one large image file. Suppose a certain Web page includes 10 small image files. The client then has to send HTTP requests to the Web server 10 times. While a CSS sprite is used and the image files are consolidated into one, the client sends only one HTTP request to the Web server. By using a CSS sprite, the number of HTTP requests can be reduced. As a result, the page-loading time is greatly reduced.

C. SPDY

SPDY [4] is a protocol proposed by Google for the purpose of improving Web access speed. At the 84th IETF meeting held in Vancouver, the meeting decided to use SPDY as a basis of the technical specification of HTTP/2.0, which has been studied in the IETF standardization process. In SPDY, the client establishes one SPDY session per domain and communicates with the Web server using the SPDY session. The SPDY has the following features:

- Concurrent content receiving with request multiplexing,
- HTTP header compression,
- Content compression,
- Server push.

Thanks to the above features, it can be expected that SPDY will improve Web access speed compared with the existing HTTP protocol.

III. SPDY ACCELERATOR

In this section, we describe the method of high-speed Web access that combines the SPDY protocol and cache system. As mentioned in Section 2.3, the client has to establish multiple

SPDY sessions with multiple Web servers in a multi-domain configuration. In this case, SPDY is not able to realize high-speed Web access since it takes several seconds to establish multiple SPDY sessions in a mobile environment with high latency. In this paper, we propose the SPDY accelerator, which can considerably accelerate Web access speed by reducing the number of SPDY sessions in a multi-domain configuration. Figure 1 shows an overview of the proposed system. The proposed system is composed of a client, Web servers, DNS servers, and a SPDY accelerator. The SPDY accelerator plays the role of a reverse proxy cache system.

Figure 2 shows the sequence flow of the proposed system, where the client receives a main HTML file from Web server A and subsequently its page resources from Web server B.

1. The client performs DNS lookup of Web server A to DNS1. DNS1 responds with the IP address of the SPDY accelerator in response to the client's query.
2. The client sends the request of the maim HTML file on Web server A to the SPDY accelerator according to the response from DNS1.
3. On receiving the client's request, the SPDY accelerator checks whether the main HTML file has been already cached or not. If not, the SPDY accelerator accesses Web server A to obtain the main HTML file.
4. The SPDY accelerator sends the main HTML file to the client. At that time, if its page resources belong to domains different from the domain to which the main HTML file belongs, the SPDY accelerator rewrites the file itself so that its page resources (site-b.com) belong to the same domain as the main HTML file (site-a.com).
5. The client parses the main HTML file and then understands that all resources included in the Web page are in a single domain. Therefore, the client can send and receive all page resources through the single SPDY session established between the client and the SPDY accelerator.

Consequently, we were able to realize high-speed Web access without the unnecessary TCP connection established between the Web server and the client.

Figure 3 shows the URL rewriting rules in the proposed system. As shown in Figure 3, the SPDY accelerator adds the FQDN of the main HTML file as a prefix to the URLs of the page resources if the FQDNs of the page resources are different from that of the main HTML file. Thereby, the SPDY accelerator can easily identify the original URLs by only removing the FQDN part of the main HTML file.

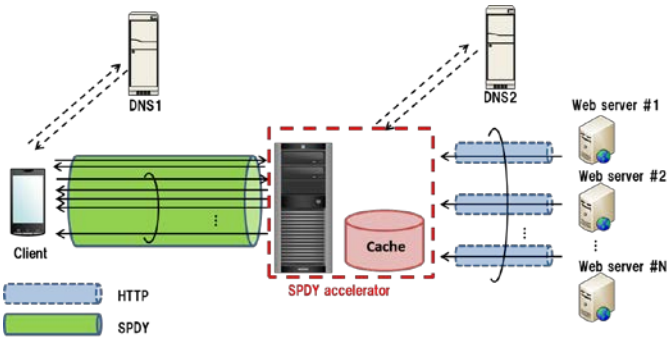

Figure 1 Overview of the proposed system

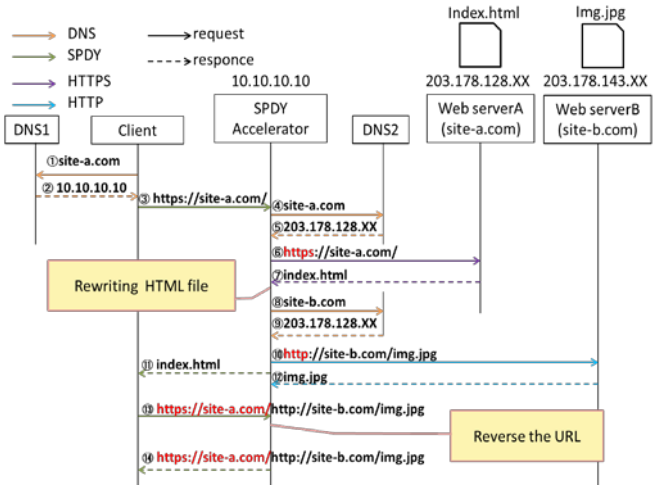

Figure 2 Sequence flow

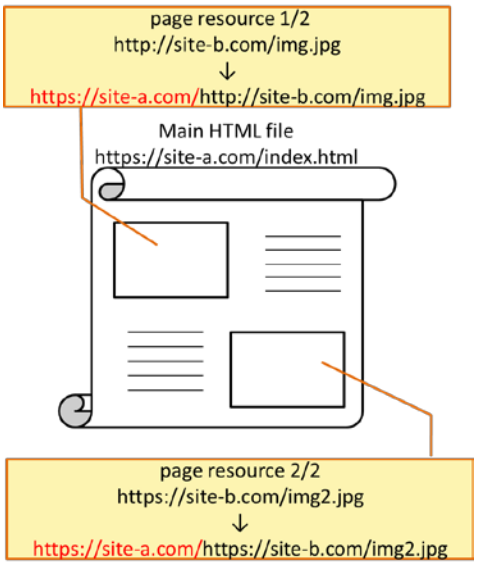

Figure 3 URL rewrite rules

IV. PERFORMANCE EVALUATION

In order to verify the efficiency of the proposed system, we conducted a performance evaluation by using the prototype system. Table 1 shows the specifications of the server that implemented the prototype system.

Table 1 Specifications of the prototype system

CPU	Intel Xeon E5506 (2.13 GHz)
Memory	DDR3 16 GB
HDD	SATA 2.5 inches 500 GB
NIC	1000BASE
OS	CentOS5.8 64 bits Linux 2.6.18
Application	Apache2.2 (mod_spdy,mod_proxy,mod_ssl)

Figure 4 shows the system configuration of the performance evaluation. In this evaluation, we arranged a client, a SPDY accelerator, two DNS servers, a physical server operated by 7 VM Web servers, and a dummynet server. In the literature [6], the average Web page has been configured with seven DNS lookups and 44 resources whose total size is 320 Kbytes. According to the literature [6], we originally created the Web page shown in Figure 5. The Web page includes 42 image files, which are stored in seven different domains, a CSS file, and an HTML file; the total size of this Web page is 320 Kbytes. In this evaluation, we measured the page-loading time with the SPDY accelerator. In addition, we also measured the page-loading time of HTTP, HTTPS, and SPDY. In the case of using HTTP, HTTPS, and SPDY, the client sends requests to the origin server directly. Here, a dummynet was used to emulate the mobile environment. The parameters of the dummynet were set as follows: the upload bandwidth is set to 1024 kpbs; the download bandwidth is set to 128 kbps, 256 kbps, 512 kbps, 1024 kbps, 2048 kbps, and 4096 kbps; and the RTT is set to 150 ms. Figure 6 shows the results of measuring the page-loading time for each condition. As we can see from Figure 6, the proposed system is significantly faster than SPDY. In this evaluation, we confirmed that the page-loading time can be shortened by about a third. Consequently, we confirmed that by using the proposed system, the page-loading time can be significantly reduced even when the Web page consists of a multi-domain configuration.

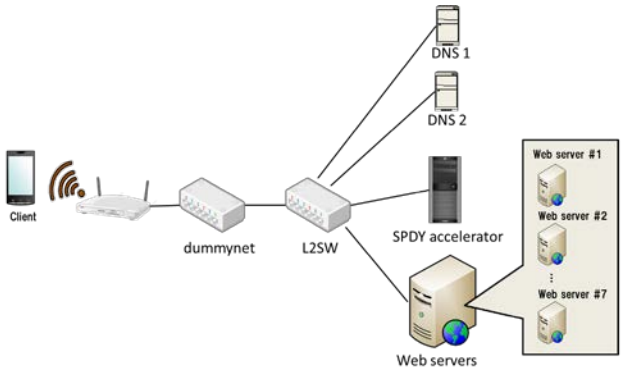

Figure 4 Evaluation environment

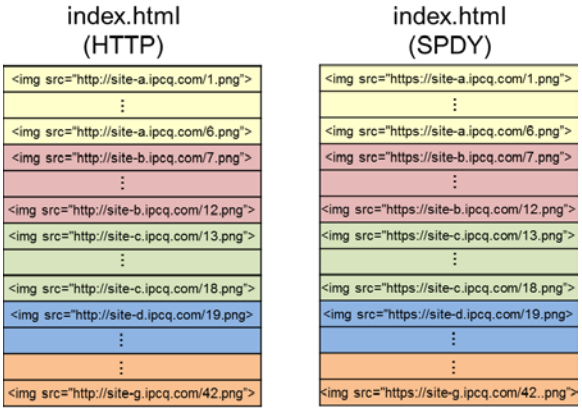

Figure 5 HTML file used in this evaluation

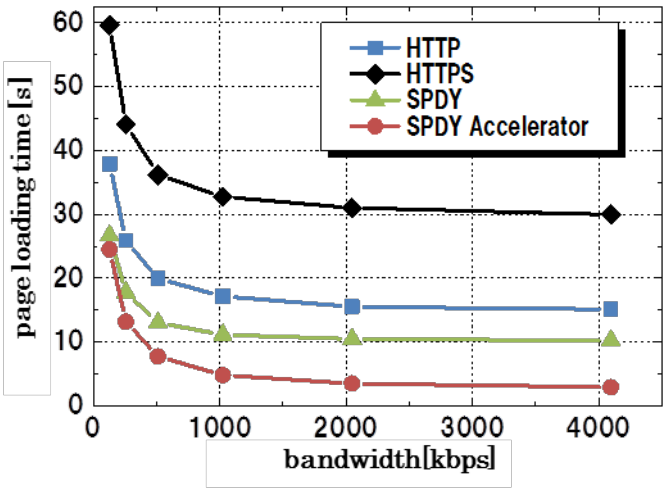

Figure 6 Page-loading time

V. CONCLUSIONS

This paper presented the SPDY accelerator, which is a Web acceleration system by combining the SPDY protocol and a cache system. With the current SPDY protocol, the client has

to establish multiple SPDY sessions with multiple Web servers, when the client is about to load a Web page that includes several page resources stored in multiple domains (multi-domain configuration). Since it takes several seconds to establish multiple SPDY sessions in a mobile environment with high latency, SPDY is not able to realize high-speed Web access. In the proposed system, in order to reduce unnecessary SPDY sessions, the FQDN of the page resource is rewritten so as to correspond with the FQDN of the Web page. To verify the effectiveness of the proposed system, we developed a prototype system and evaluated the page-loading time by using the prototype system. We confirmed that by using the proposed system, the page-loading time can be significantly reduced even when the Web page consists of a multi-domain configuration.

ACKNOWLEDGMENTS

If needed, one or more (all) author(s) may thank or acknowledge other people's or organization's helps or support.

REFERENCES

[1] Ministry of Internal Affairs and Communications, 2012 White Paper Information and Communications in Japan, 2012.

[2] A. Bouch, M.A. Sasse, and H. DeMeer, “Of Packets and People: A User-centered Approach to Quality of Service,” proc. IEEE International Workshop on Quality of Service (IWQoS 2000), pp. 189-197, June 2000.

[3] Zona Research, “The Need for Speed II,” Zona Market Bulletin, no. 5, April 2001.

[4] R. Peon and M. Belshe, Spdy draft-mbelshe-httpbis- spdy-00, IETF Internet Draft, Feb 2012.

[5] T. Berners-Lee, R. Fielding, and H. Frystyk, “Hypertext Transfer Protocol—HTTP/1.0” IETF RFC1945, May 1996.

[6] S. Ramachandran, “Web Metrics: Size and Number of Resources,” https://developers.google.com/speed/articles/web-metrics?hl=ja.

[7] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, “Hypertext Transfer Protocol—HTTP/1.1,” IETF RFC2616, June 1999.

[8] S. Souders, “High Performance Web Sites,” O'Reilly Media, 2007.

[9] S. Souders, “Even Faster Web Sites,” O'Reilly Media, 2009.

[10] Shea, Dave: CSS Sprites: Image Slicing's Kiss of Death, March 2004, http://www.alistapart.com/articles/sprites, visited on 2010-06-25.

[11] OnlineGraduatePrograms, Instant America, March 2012 http://www.onlinegraduateprograms.com/instant-america/.

[12] Tenni Theurer, “Performance Research, Part 4: Maximizing Parallel Downloads in the Carpool Lane,” April 2007, http://yuiblog.com/blog/2007/04/11/performance-research-part-4/.