Accelerating Web Access through SPDY

N. Deepan Karthik[1]
[1]Department of Computer Science Engineering
[1]VIT University, Chennai

Abstract— The entire world is revolving around Internet and at this point of time people anticipate for faster browsing speed which is now being achieved by SPDY (pronounced speedy). 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. This paper describes the development in the transportation of web contents using fewer TCP connections.

Key words: SPDY, TCP, HTTP, SSL

I. INTRODUCTION

HTTP has a number of flaws that affect the efficiency of requests between a client and server. These flaws impose limitations on the performance of your websites. This issue led to the innovation of SPDY.

SPDY is a replacement for HTTP. SPDY is a packet (frame) oriented binary protocol, usually wrapped in TLS. SPDY manipulates HTTP traffic, with particular goals of reducing web page load latency and improving web security. SPDY achieves reduced latency through compression, multiplexing and prioritization although this depends on a combination of network and website deployment conditions. SPDY does not actually replace HTTP, it modifies the way HTTP requests and responses are sent. This means that all existing server-side applications can be used without modification if an SPDY-compatible translation layer is put in place.

The SPDY equivalent of an HTTP request / response is a stream. Each stream has an unique id, and handles a single request /response. A stream is split into frames. A control frame contains the HTTP headers. A data frame contains the data. In a response the data might be an image or gzipped HTML. If the request has data frames, they usually contain POST data. The response data can be split over several interleaved data frames.

II. FEATURES AND DESIGN

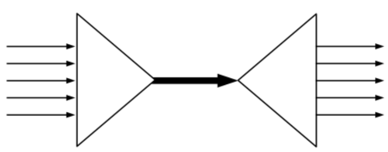

Fig. 1: Graphical representation of compressing, multiplexing and prioritizing.

A. BASIC FEATURES

1) Multiplexed streams

SPDY allows unlimited concurrent streams over a single TCP connection. Because requests are interleaved on a single channel, the efficiency of TCP is much higher: fewer network connections need to be made, but more densely packed, packets are issued.

2) Request prioritization

Although unlimited parallel streams solve the serialization problem, they introduce another one: if bandwidth on the channel is constrained, the client may block requests for fear of clogging the channel. To overcome this problem, SPDY implements request priorities: the client can request as many items as it wants from the server, and assign a priority to each request. This prevents the network channel from being congested with non-critical resources when a high priority request is pending.

3) HTTP HEADER compression

SPDY compresses request and response HTTP headers, resulting in fewer packets and fewer bytes transmitted.

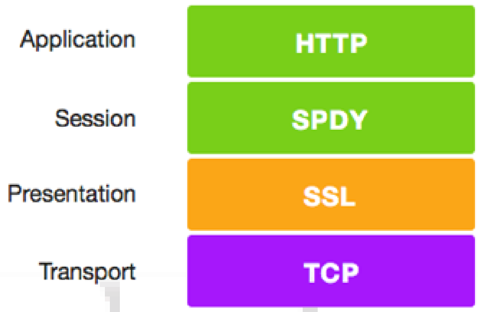

Fig. 2: Different Layers

B. ADVANCED FEATURES

In addition, SPDY provides an advanced feature, server-initiated streams. Server-initiated streams can be used to deliver content to the client without the client needing to ask for it. This option is configurable by the web developer in two ways:

1) Server push

SPDY experiments with an option for servers to push data to clients via the X-Associated-Content header. This header informs the client that the server is pushing a resource to the client before the client has asked for it. For initial-page downloads (example- the first time a user visits a site), this can vastly enhance the user experience.

2) Server hint

Rather than automatically pushing resources to the client, the server uses the X-Subresources header to suggest to the client that it should ask for specific resources, in cases where the server knows in advance of the client that those resources will be needed. However, the server will still wait for the client request before sending the content. Over slow links, this option can reduce the time it takes for a client to discover it needs a resource by hundreds of milliseconds, and may be better for non-initial page loads.

SPDY aims to achieve lower latency through basic (always enabled) and advanced (optionally enabled) features. The SPDY project defines and implements an application-layer protocol for the web, which greatly reduces latency.

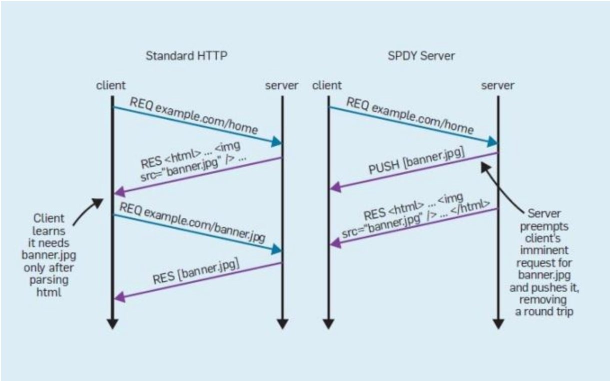

Fig. 3: HTTP vs. SPDY

III. Goals

The upcoming goals of SPDY are:

- To target a 50% reduction in page load time.
- To minimize deployment complexity. SPDY uses TCP as the underlying transport layer, so requires no changes to existing networking infrastructure.
- To avoid the need for any changes to content by website authors. The only changes required to support SPDY are in the client user agent and web server applications.
- To reduce the bandwidth currently used by HTTP by compressing headers and eliminating unnecessary headers.
- To define a protocol that is easy to implement and server-efficient. We hope to reduce the complexity of HTTP by cutting down on edge cases and defining easily parsed message formats.
- To enable the server to initiate communications with the client and push data to the client whenever possible.

	DSL 2 Mbps downlink, 375 kbps uplink	
	Average ms	**Speedup**
HTTP	3111.916	
SPDY basic multi-domain* connection / TCP	2242.756	27.93%
SPDY basic single-domain* connection / TCP	1695.72	45.51%
SPDY single-domain + server push / TCP	1671.28	46.29%
SPDY single-domain + server hint / TCP	1608.928	48.30%
SPDY basic single-domain / SSL	1899.744	38.95%
SPDY single-domain + client prefetch / SSL	1781.864	42.74%

Fig. 4: Average page load time

IV. Conclusion

The development in SPDY is still in progress. People can switch over to SPDY from HTTP/HTTPS (most of the browsers does it by its own). It does not only increase the web load time, it also improves the web security.

References

[1] http://www.chromium.org/spdy/spdy-whitepaper
[2] http://en.wikipedia.org/wiki/SPDY
[3] https://thethemefoundry.com/blog/why-we-dont-use-a-cdn-spdy-ssl/
[4] “High Performance Browser Networking” by IlyaGrigorik
[5] The SPDY book” by Chris Storm