A Performance Analysis of Secure HTTP Protocol

Xubin He, *Member, IEEE*
Department of Electrical and Computer Engineering
Tennessee Technological University
Cookeville, TN 38505, U.S.A
hexb@tntech.edu

Abstract

Data security plays an essential role in today's web service. Secure HTTP (HTTPS) is one of the popular protocols to transfer sensitive data over the Internet. This paper shows how the security affects performance compared to HTTP. Based on our preliminary measurements, the average response time ranges from 0.1 to 5 seconds, and from 6 to 8 seconds, for HTTP and HTTPS, respectively. Our experiments also show that the HTTPS overhead mainly comes from clients other than servers.

1 Introduction

WITH the phenomenal growth of popularity of the Internet, security is more and more important to the E-commerce business[4]. For some sites such as finance and online payment system, it's more important than performance. Much research has been sparked to improve data security [3, 5, 8]. Some mechanisms are proposed to improve security of network attached storage systems. Freeman and Miller gave an architecture file system to guarantee user data security using end-to-end encryption[1]. A secure scheme to protect network-attached storage systems against different types of attacks using strong cryptography is proposed in [6]. HTTP [7, 9] is the most popular protocol to transfer document over the Internet and Secure HTTP (HTTPS) is a protocol to transfer sensitive HTTP data over SSL (Secure Socket Layer) [2].

To understand how and to what extent the security mechanism of HTTPS affects the performance, we have been performing extensive measurements on the web servers

environment. Our results show that HTTPS has different behaviors compared with HTTP and it costs more system resources on client side.

2 Preliminary Performance Analysis

We configured a standard http web server and a HTTPS server which is powered by a Verisign trial ID. Under HTTPS, all contents are encrypted through SSL (Secure Socket Layer). The system configurations are described in table 1, and all machines are connected through a 100Mbps switch to form an isolated LAN.

The benchmark tool we are using is *Microsoft Web Capacity Analysis Tool* (WCAT). WCAT runs simulated workloads on client-server configurations. WCAT measures how Internet Information Services and network configuration respond to a variety of different client requests for content, data, or html pages. The results of these tests can be used to determine the optimal server and network configuration. WCAT is specially designed to evaluate how Internet servers running Windows 2000 (or Windows NT) and Internet Information Services respond to various client workload simulations.

Table 1: Test Environments.

Machine	CPU	Memory	OS
Server	PIII 400	64MB	NT 4 Server, SP 5
Controller	PIII 800	256MB	NT4 Server, SP5
Clients (49)	P 166	64MB	NT4 Workstation

The server provides contents to the clients, and the controller collects the test data. The test suite we are using is Webstone, which is a standard benchmark and the size of the requested page ranges from 1k to 200k. Each client runs 2 threads simultaneously.

We started our experiments with measurement of system performance for HTTP and HTTPS under Webstone workload. Figure 1 show shows the measurement results in terms of throughput (Figure 1a) and response time (Figure 1b) against number of clients. It can be seen from these figures that throughput get saturated under HTTP much faster that HTTPS. For HTTP, the server gets saturated at throughput being 600 connections/second with 5 clients. While for HTTPS, the throughput is steadily

increasing and gets saturated at 400 connections/second with over 45 clients. That's around 33% performance reduction compared to HTTP. The response times for HTTP (less than 5 seconds) are also significantly lower than that of HTTPS which ranges from 6 to 8 seconds.

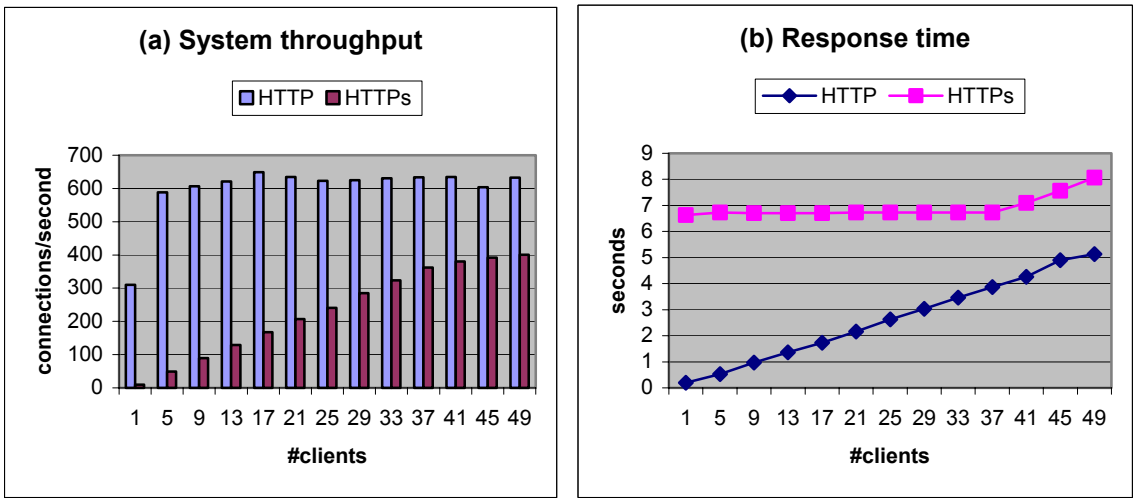

Figure 1: Webstone system throughput and response time

To understand why HTTPS reduces system performance so dramatically, we measured the server processor time and internal server behavior such as context switches and system calls, as shown in Figure 2 and Figure 3, respectively.

Figure 2 shows the CPU time. For HTTP, the server processor gets saturated very quickly, while for HTTPS, the server processor has much idle time before it is saturated with over 45 clients. This implies that some computation such as verification, SSL encryption is handled on the client side before a request is send to the server, which dramatically reduces the overall system performance. Figure 2b further shows that even the server is saturated, the total processor time spend on web server process are still less than 80% for both HTTP and HTTPS. Another 20% CPU time are used to operating system itself. Our further investigations on internal system counters such as context switch (Figure 3a) and system calls (Figure 3b) confirm the observation. For HTTP, with much more context switch and systems calls than HTTPS, the server provides much better system performance, that is, higher throughput and lower response time.

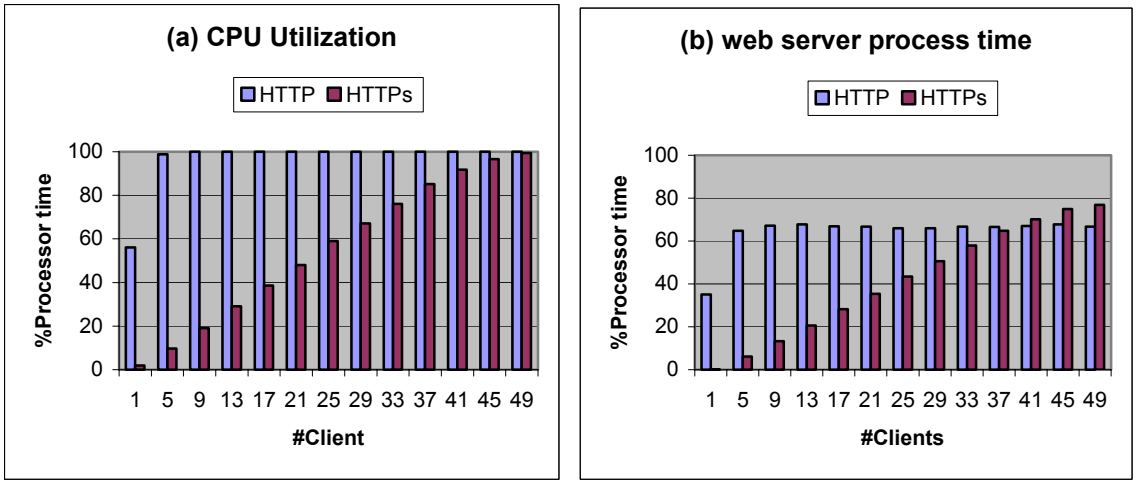

Figure 2: CPU time

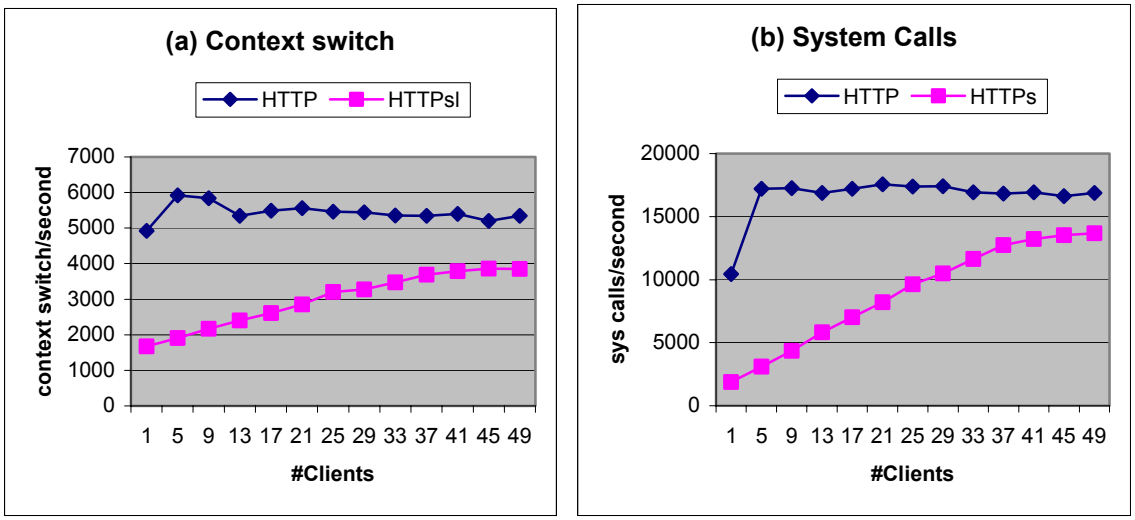

Figure 3: System behavior

3 Conclusions

In this paper, we have studied the security cost of HTTPS under Webstone workloads. Compared to standard HTTP, HTTPS costs more system resources on clients. Some computation such as verification, SSL encryption is handled on the client side before a request is send to the server, so much more clients are needed to saturate the server than that of HTTP. Once the server is saturated, the system performance of HTTPS achieves around 67% of HTTP in terms of throughput.

References

[1] W. Freeman and E. Miller, "Design for a Decentralized Security System for Network Attached Storage," *Proc of 17th IEEE Symposium on Mass Storage Systems*, 2000.

[2] A. O. Freier, P. Karlton, and P. C. Kocher, "The SSL Protocol Version 3.0," *http://home.netscape.com/eng/ssl3/draft302.txt*,1996.

[3] A. Goldberg, R. Buff, and A. Schmitt, "Secure Web Server Performance Dramatically Improved by Caching SSL Session Keys," *Proc of Workshop on Internet Server Performance*, 1998.

[4] B. K. Haddon, "Security in Storage Management: The Standards Question," *Proc of 18th IEEE Symposium on Mass Storage Systems*, 2001.

[5] J. Hughes, "Storage Security," *Proc of 19th IEEE Symposium on Mass Storage Systems*, 2002.

[6] E. L. Miller, et al., "Strong Security for Network-Attached Storage," *Proc of Conference on File and Storage Technologies (FAST 02)*, 2002.

[7] J. C. Mogul, "The case for persistent-connection HTTP," *Proc of SIGCOMM Symposium on Communications Architectures and Protocols*, Cambridge, MA, 1995.

[8] E. Riedel, M. Kallahalla, and R. Swaminathan, "A framework for evaluating storage system security," *Proc of Conference on File and Storage Technologies (FAST 02)*, 2002.

[9] A. Rousskov and V. Soloviev, "A Performance Study of the Squid Proxy on HTTP/1.0," *World Wide Web*, vol. 2, pp. 47-67, 1999.