Secure Web Server Performance Dramatically Improved by Caching SSL Session Keys

Arthur Goldberg, Robert Buff, Andrew Schmitt
[artg, buff, schm7136]@cs.nyu.edu
Computer Science Department
Courant Institute of Mathematical Science
New York University
715 Broadway, Room 711
New York, New York 10003

Abstract[1][2]

Performance measurements of secure production Web servers show that reusing cached SSL session keys can significantly reduce client response time. The time to download secure Web documents is reduced between 15% and 50% for 5 geographically diverse U.S. sites.

Introduction

The importance of electronic commerce is widely acknowledged. Surveys of Web users indicate that poor performance is a major cause of dissatisfaction. We have embarked on a major study of the performance of secure Web communications. Here we present results proving the importance of session key caching for improving end-to-end performance.

While the computational cost of public key encryption is widely understood [Kaufman95], and has led to the development of session key caching[3] across short-lived transactions as in the Web, there have been no detailed studies of the performance of key exchange in the Web.

We briefly review the operation of secure Web communications. Conducting secure communications typically involves the following steps:

- A client establishes a TCP session with a server, which involves one round-trip message exchange.
- On top of TCP, the client and server establish a secure SSL communication channel [Freier96]. The client and server exchange secret session keys that will be used to encrypt and decrypt application messages.
- On top of SSL, the client and server exchange one or more HTTP messages. (Multiple messages would be exchanged over keep-alive or persistent connections [Fielding98].)

When establishing an SSL channel the client and server may either create new session keys or reuse cached keys. Establishing an SSL channel first attempts to reuse a cached session key. Exchanging a cached session key takes one round-trip. If this fails, a new session key is created and encrypted with a public key for transmission. This takes two round trips. ([Bolyard97] nicely traces SSL session setup.)

We have measured the time to establish SSL connections for multiple Web sites at many times of the day over a period of several weeks. We find that 1) reusing a cached a session key significantly decreases the time to establish an SSL session, and that 2) in some situations the time to establish an SSL session is only slightly greater than the time to establish a TCP session when a cached session key is reused.

Measurement Techniques

We call our measurement apparatus WebPerf. WebPerf consists of a Web robot and a back-end database. The robot is written in C++ and compiled with Visual C++ version 5.0, with optimization. It communicates with Winsock 2.0. To minimize contention with itself the robot browser runs single-threaded on an otherwise idle machine. The machine runs Windows NT Workstation 4.0 with TCP/IP. The robot links to a widespread SSL implementation, written by Eric A. Young, SSLeay 0.8.1 that supports SSL versions 2.0 and 3.0. The robot does not

---

[1] This work has been supported by an IBM Partnership Award.

[2] Published in the “Workshop on Internet Server Performance”, held in conjunction with SIGMETRICS’98, June 23, 1998.

[3] Session key caching is also known as ‘session resume’ or ‘session restart’.

authenticate the server since this is a client side activity. The robot runs on a 100 MHz Pentium with 32 MB of RAM with a NE 2000 NIC connected to a 10 base T Ethernet at New York University. The NYU campus is T3 connected to be Internet via NYSERnet [Chapman97].

We set low upper bounds on the computational delay in our WebPerf robot client by measuring the performance of a secure Web server located at NYU. WebPerf can establish a TCP connection in 10 milliseconds, create an SSL connection with a new session key in 40 milliseconds, establish an SSL connection which reuses a cached session key in 10 milliseconds, and download an 1000 byte HTTP document in 20 milliseconds.[4] (These numbers can be seen in Figure 2, below.) Since WebPerf runs single-threaded, on a machine by itself, the local compute time for these activities should never exceed these values. Therefore, delays we measure for Web sites must occur in the network and/or on the remote server.

Measurements and Analysis

Raw data for wwwus.netscape.com are shown in Figure 1. We measured these delays at 10 minute intervals continuously over the time period indicated.

We can estimate which portion of each absolute delay occurs in the network and which portion is spent at the hosts. We observe the SSL connect time immediately after the TCP connect time, so network and server conditions vary little in between the two, on average. Therefore, we can be confident that, on average, the difference between the two durations occurs at the client and the server.

At any given time, we see that for wwwus.netscape.com it takes several times longer to perform an SSL connect using a cached key than it takes to connect TCP, and that it takes several times longer again to connect SSL with a new session key. As expected, the duration of all network and server activities increase during the congested afternoon of each day. Note that the minimum TCP connect duration is consistent with the cross-country signal travel time of about 50 milliseconds.

Distributions of these data for intranet.nyu.edu, www.coned.com and wwwus.netscape.com appear in Figures 2, 3 and 4 for measurements between February 21 and 28, 1998. The histograms show the percent of each measurement at a given duration for TCP connects, and SSL connects. For intranet.nyu.edu, the histogram also shows HTTP GETs of documents less than 5,000 bytes (or four 1500 byte IP packets).

Figures 1 to 4 show about 95% of the data; the remaining samples were classified as outliers. The following table shows the fraction of samples in percentage ignored for each figure.

	TCP	SSL Cached	SSL NOT Cached	HTTP GETs
intranet.nyu.edu	0.2	8.6	5.7	2.7
www.coned.com	4.7	1.0	1.9	NA
wwwus.netscape.com	3.5	4.0	4.7	NA

We use these histograms to compare the relative duration of TCP and SSL connects and HTTP GETs. They demonstrate the significant performance improvement achieved by reusing cached session keys. In figure 2, we also see that the median time to establish an SSL connection which creates new session keys takes about ¼ of the time of an HTTP GET, which demonstrates that it contributes significantly to the overall response time of a browser retrieving a Web object.

The Figures also show that at these sites queuing effects from contention only slightly increase the median delay of these operations. For example, at intranet.nyu.edu the minimum SSL connect without a cached session key takes 40 milliseconds—which certainly would have encountered almost no queuing delay since we took 1947 samples over the course of a week—and 99 percent of these connects take less then 200 milliseconds. We see that most of the distribution is close to the distribution minimum for all connects at both sites.

---

[4] From our data it appears that NT Workstation 4.0 quantizes time slices at 10 milliseconds and does not interrupt processes running at normal priority to report network message arrival. We therefore suspect our measurements are 5 milliseconds too large, on average. The delays we measure are large enough that this possible error does not alter our conclusions.

Host Name	HTTP Server Software	SSL Public Key – Encryption Key	Server Location
Intranet.nyu.edu	Stronghold/2.0 Apache/1.2b10	RSA – RC4 (128)	New York City (NYU)
Secure.webmaster.com	Microsoft-IIS/3.0	RSA (512) – RC4 (40)	California
www.coned.com	Microsoft-IIS/3.0	RSA (512) – RC4 (40)	New York City
www.farsight.com	Netscape-Enterprise/2.01	RSA – RC4 (128)	Boston
Wwwus.netscape.com	Netscape-Enterprise/3.5.1	RSA – RC4 (128)	California

Table 1. Sites and Server Software, Public Key – Encryption Key, and Location.

Host Name	Median TCP CONNECT	Median SSL CONNECT Duration		Median HTTP GET response time	Savings from SSL caching	Total Web response time		Savings from caching (%)
		Without caching	Cached			Without caching	Cached	
Formula	T	S_{nc}	S_c	H		$W = T+S_c+H$	$C = T+S_{nc}+H$	$100(W-C)/W$
Intranet.nyu.edu	10	40	10	150	30	200	170	**15**
Secure.webmaster.com	80	190	80	360	110	630	520	**17**
www.coned.com	30	110	40	210	70	350	280	**20**
www.farsight.com	100	930	360	520	570	1550	980	**36**
wwwus.netscape.com	80	940	320	410	620	1430	810	**43**

Table 2. Median performance of TCP and SSL connects demonstrate the advantage of caching SSL session keys. All times in milliseconds.

Finally, we summarize the performance of SSL connect for 5 sites in two tables. These sites were selected essentially randomly. We choose sites that provided multiple secure documents and were distributed at different distances from New York. Each row summarizes one secure site.

Table 1 lists each site's hostname, server software, public and sessions encryption keys, and location. The server was identified in the HTTP "server" header in responses. SSL negotiates and reports the keys which client and server agreed to exchange. An SSL key exchange is described by a pair, the public key (and its encryption algorithm) and the session key (and its encryption algorithm).

In Table 2, the median connect times are used because long connect durations (especially many seconds for TCP connects) significantly, and misleadingly, increase the average. This table shows that caching session keys improved performance for several different kinds of servers and several ciphers. The median connect HTTP duration is the time, in milliseconds, to retrieve documents less than 5000 bytes.

The last column of Table 2 shows the total response time savings for *complete* Web document retrievals achieved by caching SSL session keys. Let T = TCP connect time + SSL connect time + HTTP GET time. By T(c), we denote T(c) for a cached session key and by T(nc), we denote T for a non-cached session key. The last column in Table 2 is given by

$$(T(nc) - T(c)) / T(nc).$$

Conclusions

We have shown new techniques and measurements for evaluating the performance of SSL key exchange. Our results convincingly demonstrate that reuse of session keys for retrieving secure HTTP objects can reduce the time to securely access objects on the Web by as much as 50%.

References

[Bolyard97] Nelson Bolyard, "Export Client SSL Connection Details", 1997, http://home.netscape.com/eng/ssl3/traces/trc-clnt-ex.html

[Chapman97] Gary Chapman, "NYU-NET: Report on a Work in Progress", Connect, Fall 1997. http://www.nyu.edu/acf/pubs/connect/fall97/NetsNYU-NETFall97.html

[Freier96] Freier, Alan O., Philip Karlton, Paul C. Kocher, "The SSL Protocol Version 3.0" Internet Draft, November 18, 1996. http://home.netscape.com/eng/ssl3/draft302.txt

[Fielding98] R. Fielding, J. Gettys, J. C. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, March 13, 1997, "Hypertext Transfer Protocol -- HTTP/1.1", http://www.w3.org/Protocols/History.html

[Hudson] Hudson, Tim J., and Eric A. Young. "SSLeay Programmer Reference", circa 1997, http://psych.psy.uq.oz.au/~ftp/Crypto/ssl.html

[Kaufman95] Kaufman, Charlie, Radia Perlman, Mike Speciner, "Network Security: Private Communication in a Public World", Englewood Cliffs, NJ Prentice Hall, 1995.

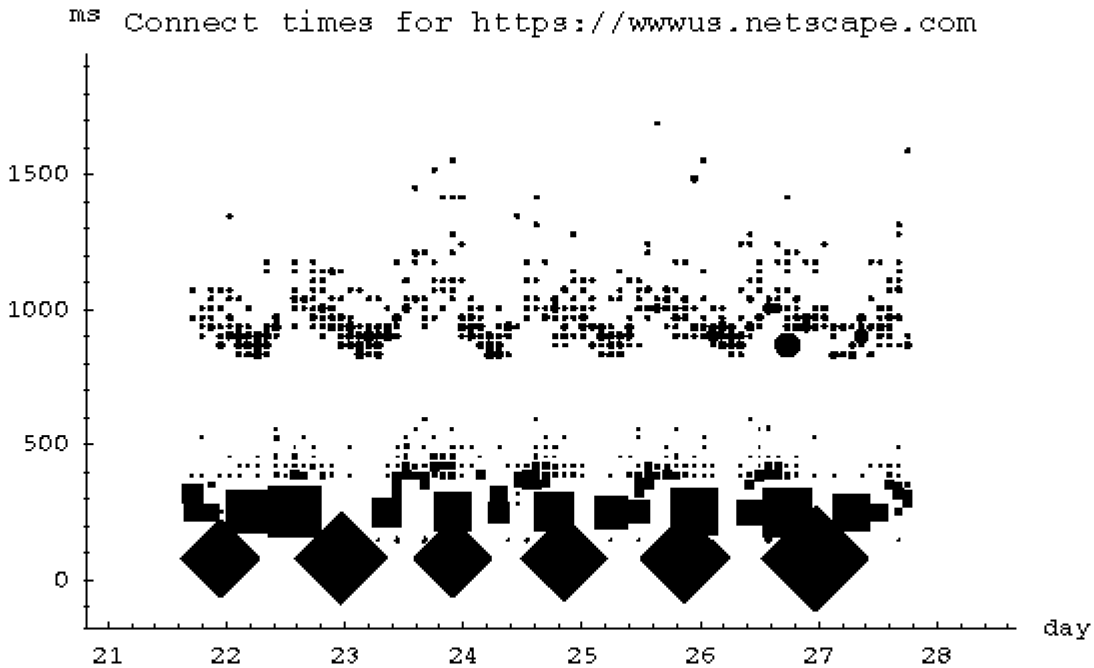

Figure 1. Duration of TCP and SSL connect times between New York University and Netscape Corp. in February 1998, showing the benefits of caching SSL session keys. The day number represents that start of the day, midnight EST. The circles in the upper portion of the graph represent 659 SSL connects that create a new session key; the boxes represent 5975 SSL connects that use a cached session key; the diamonds represent 6674 TCP connects. Each graphic symbol represents many points. Its area is proportional to the number of data points. The center of each symbol is placed at the centroid of the points it represents.

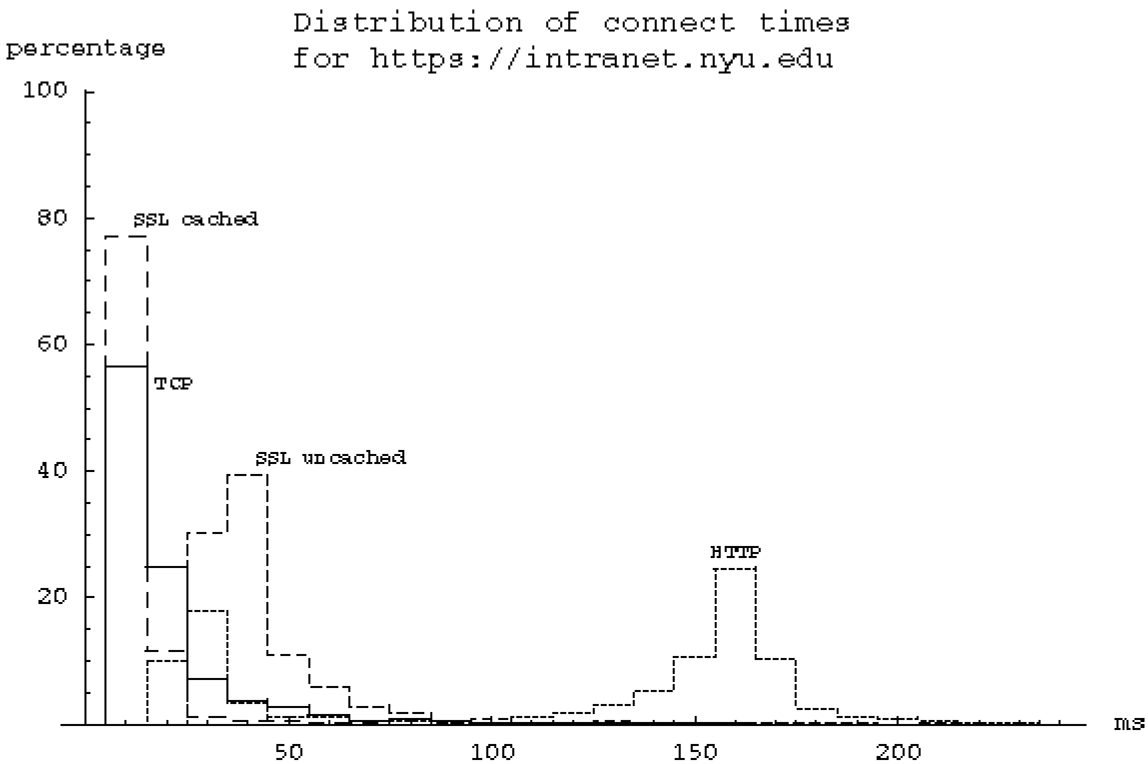

Figure 2. Distribution in 10 millisecond bins of connect times for TCP, SSL reusing a cached session key, SSL creating a new session key, and HTTP GETs, for 1947 pairs of connections in the last week of February, 1998 for intranet.nyu.edu.

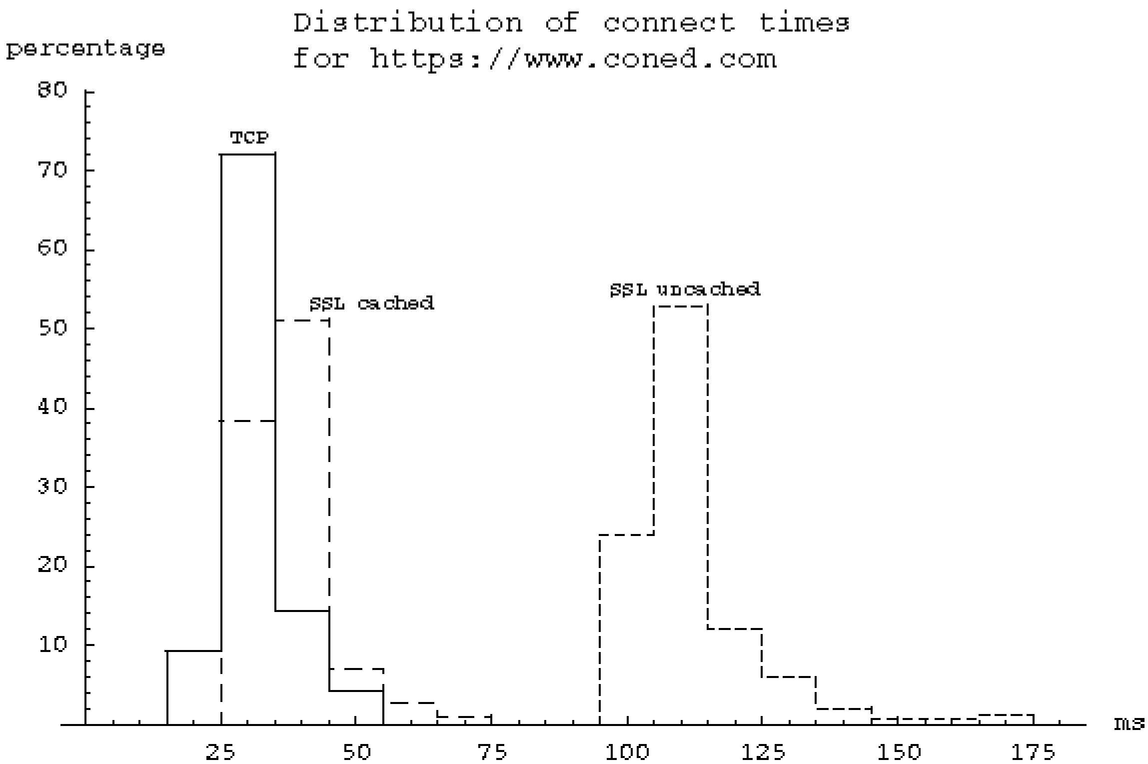

Figure 3. Distribution in 10 millisecond bins of connect times for TCP, SSL reusing a cached session key, and SSL creating a new session key, for 8003 samples in the last week of February, 1998 for www.coned.com.

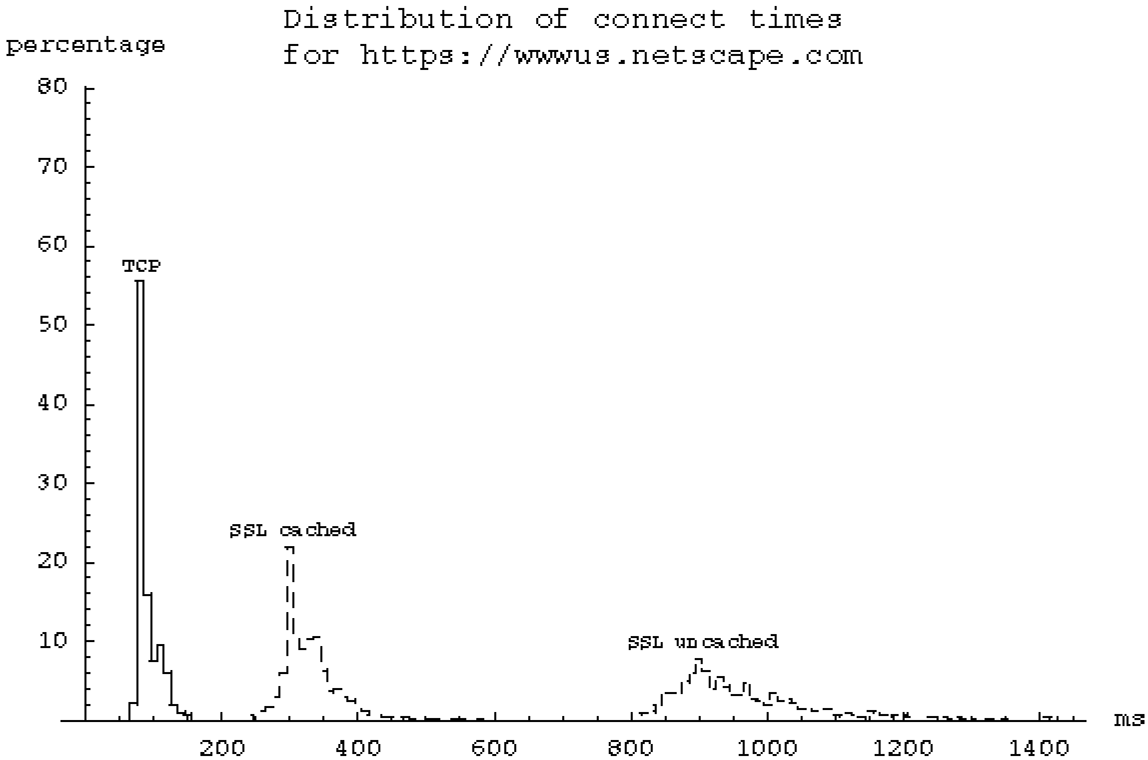

Figure 4. Distribution in 10 millisecond bins of connect times for TCP (6674 samples), SSL reusing a cached session key (5975 samples), and SSL creating a new session key (659 samples), in the last week of February, 1998 for wwwus.netscape.com.