On the Delivered Performance of the Sun Crypto Accelerator 1000

Shih-Hao Hung and Pallab Bhattacharya

Performance and Availability Engineering

Sun Microsystems Inc.

901 San Antonio Road

Palo Alto, CA 94303

{hungsh, pallab}@eng.sun.com

Abstract

Today, computer and network security has received enormous amount of attention. In particular, security protocols and cryptographic operations are constantly employed to ensure authentic and private communications in the business world. However, the high cost associated with cryptographic operations has been a critical factor that has slowed down many transactions and has prevented security protocols from being widely enforced. For example, Web browsing using a secure protocol often results in slow responses as the Web servers frequently reach their capacities.

The *Sun Crypto Accelerator 1000* (Sun CA1000) is a hardware-based, high-performance cryptographic security solution that enables a Sun server to handle large client loads under security measures deployed using techniques such as the *Secure Sockets Layer* (SSL) protocol, which is commonly used in e-commerce environment. Many critical cryptographic functions employed by SSL can be off-loaded from a Web server to the Sun CA1000 board(s) and performed in high speed.

The power of the Sun CA1000 solution takes the performance of a Sun-based secure Web server solution to the next level. Our evaluation shows that it delivers remarkable improvement on the performance of the Sun Fire™ 6800 server. Each Sun CA1000 board can efficiently process 4400 RSA operations per second and 495 Mbps of Triple-DES data encryption, a performance that is 20 times and 8 times, respectively, faster than the performance of the host processor. With seamless integration, such optimization accelerates the Web server by more than 270% in our Web server benchmarks and allows the host processors to delegate themselves on other aspects of Web applications.

1. Introduction

Computer and network security is increasingly important to individuals and corporations in the highly-computerized society today. Many security measures rely on cryptographic functions to ensure the confidentiality, authentica-

The paper was originally presented in the Sun User Performance Group Conference, Honolulu, April 2002.

tion, and integrity of the data stored in computers and the messages transmitted over the network. There are, however, cost and overhead associated with the secure measures that have been significant concerns for the users.

The predominant security protocol for electronic commerce, on which we focus in this paper, is *Secure Sockets Layer (SSL)* [1][2]. SSL encrypts client-server communications and provides functions for authentication and message integrity. SSL was originally developed by Netscape. As it gains in popularity, open-source libraries such as Network Security Services (NSS) [3] and OpenSSL [4] were provided to the public and used to implement all sorts of security features for network applications. SSL is supported by the leading web servers, e.g. iPlanet™ Web Server (iWS) [5] and Apache [6] as well as Web browsers.

Support for SSL demands considerable performance from a Web server system due to the extra computations and messages involved in the protocol processing. The most computation-intensive cryptographic operations occur during SSL's session creation, where a cryptographic session is established between a client and a server. Additionally, SSL incurs computational load for encrypting data exchanged between the client and the server, which is also known as *bulk data encryption*.

The Sun Crypto Accelerator 1000 solution accelerates both session creation and bulk encryption computations for SSL that enhances the performance of SSL on Sun Servers. The Sun CA1000 is a solution that is designed to off-load cryptographic computations from the host processors. Many critical cryptographic functions, such as *RSA* [7] and *Triple-DES (3DES)* [8], can be off-loaded from a Web server to the Sun CA1000 and performed in parallel. Such an optimization significantly improves the throughput of a secure Web server and save the host processors for other work.

This paper discusses our observations on the delivered performance of the Sun CA1000. In Section 2, we overview the capability of the Sun CA1000 and evaluate its performance using in-house microbenchmark programs. We reveal the potential of the Sun CA1000 in terms of the acceleration of cryptographic functions (RSA and 3DES) and the savings of host processor time. In Section 3, as a case study, we examine the effectiveness of the Sun CA1000 in accelerating iPlanet Web server 6.0 (iWS6). We measure the performance of a Sun Fire 6800 server running iWS6, using a modified SPECweb99_SSL benchmark program that drives the server with secure Web requests (HTTPS) generated from remote clients. Section 4 discusses the performance considerations and tuning options that users should pay attention in running high-performance secure Web servers. Section 5 summarizes our findings and concludes the paper.

2. The Sun Crypto Accelerator 1000

The Sun Crypto Accelerator 1000 solution[1] consists of a single-chip cryptographic accelerator in a PCI package and software packages with SSL support. The solution is compatible with most PCI-based Sun servers running the Solaris™ 8 Operating Environment. Currently, SSL support is available for iPlanet 4.x/6.x Web Servers and OpenSSL/Apache 1.3.12.

1. In this paper, we use the term “solution” to emphasize the integration of hardware and software.

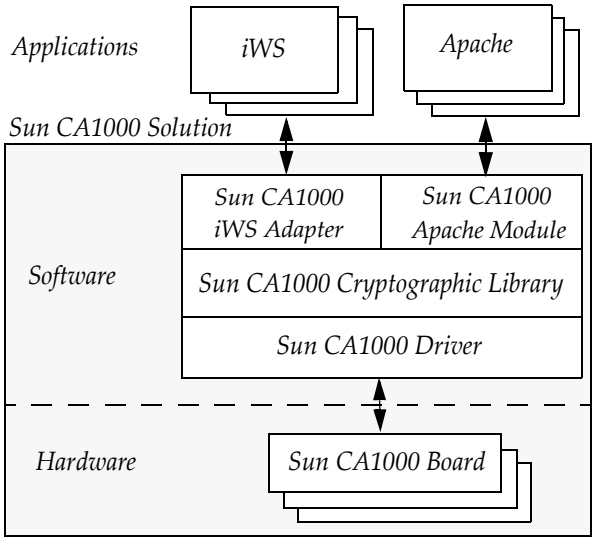


Figure 1: Web servers and the components of the Sun CA1000 solution.

Figure 1 illustrates how Web servers exercise the components of the Sun CA1000 software and hardware. Multiple servers can utilize the Sun CA1000 boards simultaneously via the iWS adapter or the Apache module that Sun provided. The Sun CA1000 Cryptographic library (*libcryptography*, in short) is responsible for scheduling cryptographic jobs on the Sun CA1000 boards. Also, libcryptography contains software algorithms that will be used to perform cryptographic functions in case all the Sun CA1000 boards in the system fail.

The following two of the most widely used, yet high-cost cryptographic operations can be accelerated[1]:

- *RSA: 4400 private key (CRT) operations/sec.*
- *3DES-CBC: 495 Mbps (16K packet size).*

1. These are the preliminary specifications for the pre-production Sun CA1000 boards that we received for testing. The final specifications have not been decided yet as of the submission date of this paper. The production release is expected to meet or exceed the specifications quoted in this paper. More cryptographic functions may be supported through software update.

The Sun CA1000 also supports other cryptographic functions, such as DSA and SHA1. In this paper, we chose to focus on the performance of RSA and 3DES as case studies for public-key and symmetric-key cryptography, respectively.

In this section, we briefly describe the Sun Fire 6800 system that we used to evaluate the Sun CA1000. Then we describe RSA and 3DES functions and validate the performance of the Sun CA1000 through a series of microbenchmark programs that run on a host machine. We measure and compare the throughput as well as the host processor utilization on a Sun Fire 6800 server with and without the Sun CA1000.

2.1 System under Test

Throughout this paper we evaluate the Sun CA1000 solution using the Sun Fire 6800 midframe server. The system that we use contains up to 24 UltraSPARC® III processors running at 900Mhz. The system can be dynamically configured to bring components on-line or take them off-line without disrupting system operation or requiring a system reboot.

The Sun Fire 6800 server can have up to four I/O assemblies, and each I/O assembly has eight 64-bit PCI slots. All PCI slots support 33Mhz bus speed. Two PCI slots additionally support 66Mhz operations. The I/O subsystem is designed to be scalable and upgradable so the system can handle I/O-intensive applications such as Web servers and database applications.

Sufficient amount of memory and network interfaces are installed so that the performance should not be impacted by virtual memory swapping and network congestion.

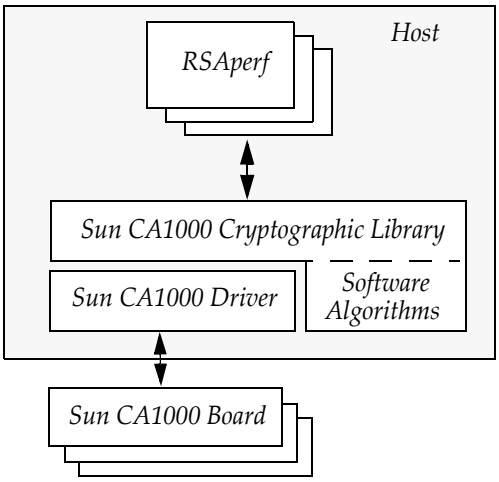


Figure 2: RSAperf, Sun CA1000 software and hardware.

The processors used for the test are currently Sun's top-of-the-line processors. Other experiments that we carried out indicate that slower systems benefit more from the use of the Sun CA1000 because software implementations run slower on these machines.

2.2 RSA Performance

RSA is a public key algorithm invented by Ron Rivest, Adi Shamir, and Len Adelman in 1977 [7]. It is the most popular public key algorithm used in establishing SSL sessions [1]. RSA operations are computationally expensive, and it is time consuming for a general-purpose processor to perform RSA operations. In this subsection, we examine the potential of the Sun CA1000, in terms of its capacity and efficiency in RSA processing.

2.2.1 RSAperf

We used a microbenchmark program developed in Sun, called *RSAperf* (**`rsaperf`**), to measure the performance of a system in processing RSA operations. As illustrated in Figure 2, multiple instances of RSAperf call the cryptographic library provided by the Sun CA1000 package to carry out RSA operations. The cryptographic library takes advantage of the Sun CA1000 automatically if the board is present, or performs the cryptographic functions in software using the host processor(s) if no Sun CA1000 board is on-line[1]. The software implementation is highly optimized for UltraSPARC® processors.

We executed RSAperf on our Sun Fire 6800 server. The results are listed in Table 1. We measure the throughput of *1024-bit* RSA operations, the most commonly used RSA algorithms with 1, 2, 4, and 8 on-line host processors. During the run, utilization of the host processors was monitored using the Solaris **`mpstat`** utility.

2.2.2 Software Performance

The software columns in Table 1 show the performance of the system when the operations were done completely by the host processor(s). The host processors were fully utilized to execute the RSA operations, as indicated by the 100% CPU utilization. A single processor system is capable of 245 RSA operations per second. The throughput scales almost linearly as the number of processors increases, as RSAperf can take advantage of multiple processors.

2.2.3 Accelerated Performance

With one Sun CA1000 board, the single-processor system is capable of 4287 RSA operations per second. For that performance, only 20% of host processor time

1. Note that this is a high-availability feature to ensure the cryptographic functions are done correctly even when no Sun CA1000 board is installed in the system.

Number of Host Processors	*Software*		*Sun CA1000*	
	RSA ops/sec	*CPU Utilization*	*RSA ops/sec*	*CPU Utilization*
1	*245*	*100%*	*4287*	*20%*
2	*484*	*100%*	*4337*	*18%*
4	*931*	*100%*	*4397*	*7%*
8	*1908*	*100%*	*4381*	*3%*

Table 1: RSA performance on Sun Fire 6800 Server.

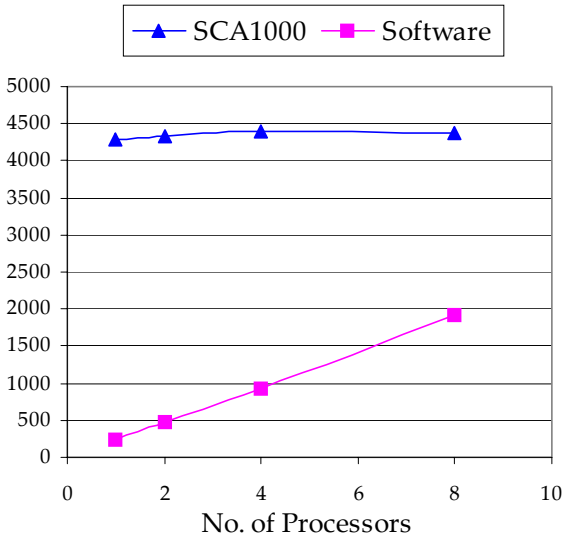


Figure 3: RSA throughput (ops/sec) on Sun Fire 6800 Server, varying the number of on-line host processors.

is needed for running the benchmark program and driving the Sun CA1000 board to its full potential.

Comparing the performance of the single-processor systems, as illustrated by Figure 3), the system with one Sun CA1000 board generates 4287/245=17.5 times as much throughput as the system without Sun CA1000 boards. Thus, the Sun CA1000 board would save 17 host processors for the user in delivering ~4000 RSA ops/s. Furthermore, considering that one Sun CA1000 board needs only 20% host processor power in realizing the performance, the Sun CA1000-enabled system is 87.5 times more efficient in running RSAperf.

2.2.4 Scalability of Accelerated Performance

The design of the Sun CA1000 emphasizes performance scalability. The Sun CA1000 board handles cryptographic operations concurrently in a pipelined fashion, and its architecture takes advantage of the abundant parallelism existing in the tasks that Web servers typically handle.

For optimizing throughput, one Sun CA1000 board can accept up to 24 concurrent RSA operations simultaneously without the host processor(s) waiting for prior operations to finish. In addition, multiple Sun CA1000 boards can be installed and utilized on one host machine to work for the same or different applications. The Sun CA1000 cryptographic library automatically schedules the cryptographic tasks to all the Sun CA1000 boards available in the system.

Figure 4 shows the results on the two-processor system, varying the number of processes that RSAperf forks to submit RSA jobs. With one RSAperf process, the Sun CA1000 board performs one RSA operation at a time, the system is capable of 539 ops/sec. From this test, we conclude that the latency for completing one

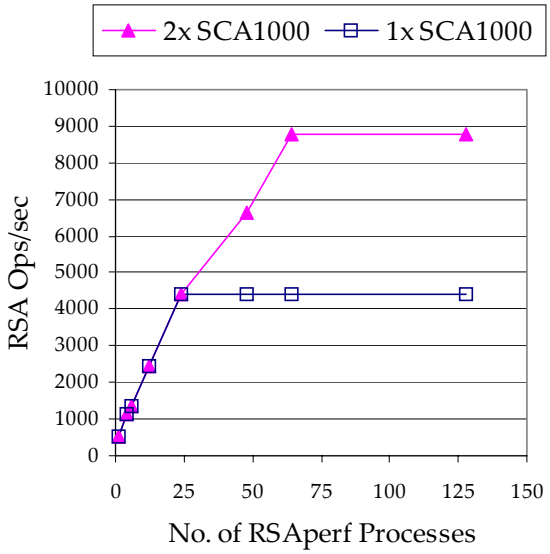


Figure 4: Sun CA1000 RSA throughput on Sun Fire 6800 Server with two on-line processors, varying the number of RSAperf processes used to generate the workload.

RSA operation is on average 1.85 milliseconds. In comparison, the host processor needs approximately 4 milliseconds to perform the same operation in software.

To fully exploit the potential of one Sun CA1000 board, RSAperf needs to submit at least 24 RSA jobs in parallel. Further adding processes does not increase the performance as single Sun CA1000 board cannot accept more than 24 concurrent RSA operations.

By adding the second Sun CA1000 board to the system, the peak performance can be doubled to 8795 RSA ops/sec. Our experiments show that it takes approximately 64 processes to keep two Sun CA1000 boards busy.

We also configured RSAperf to run in the thread mode, which uses threads to generate workload instead of processes. The two sets of results are very close, which means that applications can take advantage of the Sun CA1000 either with processes or threads.

2.3 Triple-DES Performance

Triple-DES (3DES) [8] is based on the Data Encryption Standard (DES), one of the most widely used symmetric cipher/bulk-data encryption algorithm. 3DES is computationally expensive as 3DES operation essentially requires three DES operations. The high cost associated with 3DES makes it one of the slower symmetric ciphers when implemented in software.

2.3.1 3DESperf

We used a microbenchmark program, called *3DESperf*, to evaluate the performance of 3DES executed in software and hardware. Similar to RSAperf (see Figure 2), 3DESperf calls the Sun CA1000 cryptographic library to carry out 3DES encryption of given messages as fast as the host machine can. The cryptographic library uses a Sun CA1000 board automatically when the board is present or uses host processors to perform the jobs when no Sun CA1000 board is on-line.

2.3.2 Size of Messages

The size of the encrypted messages is an important factor for the accelerated 3DES performance. For a Sun CA1000 board to encrypt a message using 3DES, the message has to be transferred between the host and the Sun CA1000 board via the PCI bus. The cost associated with job scheduling and data transfer discourages small-size payloads from utilizing hardware acceleration. There is a fixed software cost for setting up data structures, entering the kernel, and scheduling operations on the hardware,

Message Size	Software		Sun CA1000	
	Throughput (Mbps)	CPU Utilization	Throughput (Mbps)	CPU Utilization
16K	142	100%	501	20%
8K	140	100%	482	54%
4K	133	100%	419	93%
2K	128	100%	240	100%
1K	114	100%	136	100%

Table 2: 3DES performance on Sun Fire 6800 Server with 2 processors, varying message size.

which limits the effectiveness of the hardware for short messages.

For this reason, the Sun CA1000 cryptographic library handles messages smaller than 1K bytes in software rather than directing them to hardware. Our measurements, to be described in the next subsection, validate the efficacy of this policy.

2.3.3 Accelerated Performance

Table 2. shows the benchmark results measured on a system with dual 900Mhz UltraSPARC® III processors and one Sun CA1000 board. As the size of messages increases, the throughput from software increases because less function calls are involved to perform the encryption.

The overhead of function calls and job scheduling impacts the performance of the Sun CA1000 board significantly for short messages. Still, the data presented in Table 2 shows that the performance is better with the Sun CA1000 board for messages longer than 1KB.

For messages larger than 8KB, the Sun CA1000 board outperforms the software mechanism by 3.4 times, utilizing only 54% of the host processors. In terms of host processor utilization per byte encrypted, the system with one Sun CA1000 board is 6.4 time more efficient than the system without it, for encrypting 8KB messages.

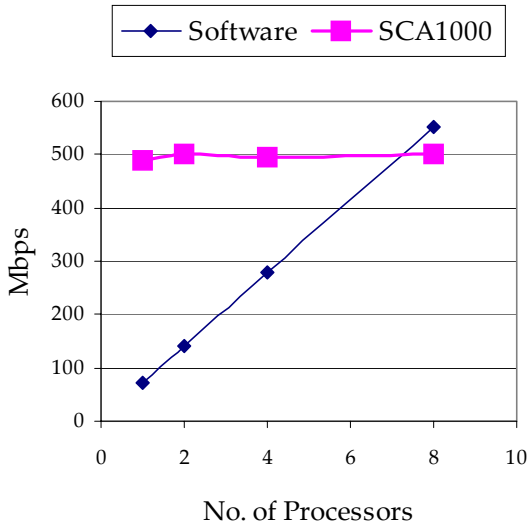


Figure 5: 3DES throughput on Sun Fire 6800 Server, varying the number of on-line host processors (Message Size=16KB).

Figure 5 shows that the performance of software 3DES scales well when more host processors are on-line. More 3DES throughput can be generated by adding host processors. It requires, however, eight host processors to achieve the same level of performance that one Sun CA1000 board provides. Notice that only 20% of processor utilization is needed to perform 500Mbps 3DES with one Sun CA1000 board. Essentially, the Sun

CA1000 board saves almost eight processors in this case.

2.4 Performance with Various PCI Bus Speeds

The Sun CA1000 board is compatible with both 32-bit and 64-bit PCI bus running at either 33Mhz or 66Mhz. The data collected on the Sun Fire 6800 server are based on a 64-bit PCI bus. While many Sun platforms are equipped with 64-bit PCI slot(s), we evaluated the performance of Sun CA1000 boards on a small server with a 32-bit PCI slot, and the results showed that the performance of a Sun CA1000 board could be sub-optimal when connected to a 32-bit PCI slot.

RSA operations require very low bus traffic besides job scheduling, thus the bus speed does not pose a limitation. For 3DES, a 64-bit 33Mhz PCI bus still provides sufficient bandwidth for one Sun CA1000 board to encrypt at 495 Mbps, but a 32-bit 33Mhz PCI bus limits the 3DES throughput to approximately 295Mbps for this UltraSPARC II-based server.

Thus, the users should be aware that the performance of a Sun CA1000 board can be affected by the PCI bus, especially when multiple high-speed I/O devices, such as Gigabit network cards, disk controllers, or another Sun CA1000 board, contend for the same PCI bus.

3. Accelerating Secure Web Server Performance

The World Wide Web is used by millions of people everyday. Numerous applications and services are provided to end users through Web servers and HyperText Transfer Protocol (HTTP). SSL is a protocol that provides a secure channel between two machines by protecting data in transit and authenticating the machines that are exchanging information. Web traffic can be protected by running HTTP over SSL (HTTPS), which is supported by nearly all modern commercial Web servers.

HTTPS puts a much higher load on the server machine than does pure HTTP, which is why HTTPS is typically used only when the situation demands a secure channel, such as when sending credit card information. Even so, World Wide Web users should not be surprised when they receive slow responses from the use of HTTPS.

We focus on the performance of secure Web servers in this section, as an example of how a high-performance crypto accelerator can improve the performance of an important application. Besides Web servers, one should notice that there are a variety of network applications that also use SSL to facilitate secure communications. Popular network protocols, such as NNTP, SMTP, POP, IMAP, can take advantage of SSL to secure news and mail access on the Internet [1]. Like Web servers, the high-cost of SSL has prevented many Internet protocols from using SSL on a regular basis. With the advent of SSL accelerators like the Sun CA1000, we envision that Internet will become a more secure place in the near future.

3.1 SSL Handshakes: Session Creation and Resumption

Here we briefly introduce the notions of SSL session. Details can be found in [1].

An SSL *connection* represents one specific communication channel. It can be divided into two phases, the *handshake* and *data transfer* phases. The handshake phase authenticates the server and estab-

lishes the private cryptographic keys to be used to encrypt the data to be transmitted during the data transfer phase.

When an SSL connection is created between the client and server for the first time, a *full handshake* is needed for the client and server to negotiate algorithms and the *master secret* that are to be used to authenticate and encrypt bulk data in the current and perhaps future SSL connections. An SSL *session* is a virtual construct representing the negotiated algorithms and the master secret, as the result of the first SSL connection. A *full SSL handshake*, also referred to as an SSL *session creation*, requires a computationally expensive public key cryptographic operation using algorithms such as RSA to establish the master secret.

An opportunity to reduce the overhead of session creation is to use the *session resumption* mechanism provided by SSL, which is also known as *session reuse* or *session caching*. It is possible for an SSL connection to resume to a previous session by reusing the same master secret established in a previous SSL connection. This avoids the computationally expensive public key cryptographic operation required for creating a new SSL session.

3.2 Benchmarking Methodology

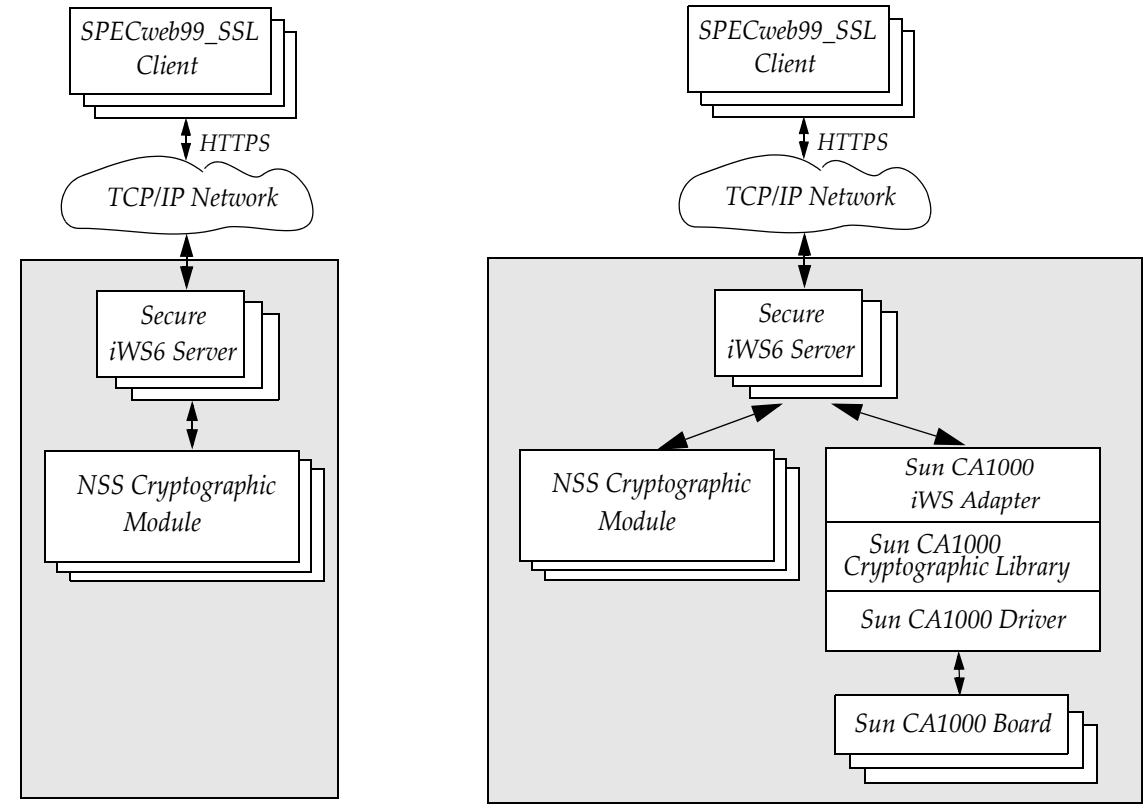
Popular benchmark programs, for example, *HTTP_LOAD* [10], *WebStone* [11], and *SPECweb* [12], exist for measuring the performance of non-secure Web servers. One can also find secure versions of these benchmark programs, which may or may not be officially supported, to experiment with secure Web servers. In our earlier experiments, we used a secure version of HTTP_LOAD, WebStone, and SPECweb96, and they all generates similar results against small Web servers.

A secure version of SPECweb99, called *SPECweb99_SSL*, was in its final development stage when we experimented with the Sun CA1000. By April 2002, the official version should be available to the public. SPECweb99_SSL is being promoted by major computer manufacturers as the standard benchmark program for performance testing of secure Web servers. Sun has been actively involved in the development of SPECweb99_SSL, and we adapted a pre-release version of SPECweb99_SSL. For testing the Sun CA1000, we found the benchmark program to be very usable and stable.

As illustrated in Figure 6, we use SPECweb99_SSL to generate the workload from a set of client machines to emulate real-world scenarios where thousands or more clients could be simultaneously accessing to a secure Web server via HTTPS. We customized the SPECweb99_SSL workload generation program to provide the tests we needed to evaluate the impact that the Sun CA1000 brings to the performance of Web servers. These customized tests were designed to measure the performance of session creation and session resumption on the server[1].

In our session creation test, the client program generates HTTPS requests to fetch small (102 bytes) files from the server, one file per SSL session, with no session resumption allowed. Small files are intentionally chosen to isolate the factors of bulk data encryption and file

1. We cannot disclose our SPECweb99_SSL results as the benchmark was not finalized when we performed the test. Our customized tests are highly deviated from the standard SPECweb99_SSL, and the results presented in this paper should not be used to compare to any standard SPECweb99_SSL results.



(a) Software-only Server (b) Accelerated Server

Figure 6: A functional diagram for the Web server benchmark environments.

transfer (disk I/O) in the test, so the focus can be on the session creation/SSL handshake phase. SSL sessions are established using 1024-bit RSA.

Our session resumption test generates similar workloads, except that the requests generated by the client program allow the Web server to speedup the SSL handshake phase with session resumption. In our experiments, Web servers took full advantage of session resumption after the first HTTPS operations from the clients, thus the ratio of session resumption[1] was nearly 100%.

Figure 6 shows two basic server configurations: (a) *software-only* and (b) *accelerated*. On the software-only server, the NSS cryptographic modules handle the cryptographic operations for the Web server using its own (internal) implementation. On the accelerated server, most cryptographic functions are handled by the NSS internal modules, except the RSA operations are accelerated using the Sun CA1000 board(s).

Our session creation test was designed to measure how well the Sun CA1000 solution can accelerate session creations with its high-performance RSA functional capability. As opposed to the session creation test, the session resumption test does not involve heavy cryptographic operations. Hence the results of the session resumption test should not be affected by the presence of the Sun

1. Session resumption ratio is defined as (number of session resumption operations)/(number of HTTPS operations).

No. of Host Processors	*NSS Module*		*Sun CA1000*		*Acceleration* (Sun CA1000/Software)
	Throughput HTTPS/sec	*Normalized Performance*	*Throughput HTTPS/sec*	*Normalized Performance*	
1	*124.7*	*1*	*337*	*2.70*	*2.7*
2	*236.3*	*1.89*	*668*	*5.36*	*2.83*
4	*462.4*	*3.71*	*1288*	*10.33*	*2.79*
8	*845.5*	*6.78*	*2457*	*19.70*	*2.91*
12	*1239*	*9.93*	*3636*	*29.16*	*2.93*
16	*1611*	*12.92*	*4385*	*35.16*	*2.72*

Table 3: iWS6 SSL session creation performance on Sun Fire 6800 Server.

CA1000 software/hardware. Because session creation basically includes the operations involved in session resumption, session resumption should be faster than session creation, and we consider session resumption performance an upper bound that limits how well the same system can do in the session creation test.

3.3 iPlanet Web Server Performance

We have used the Sun CA1000 solution to accelerate iPlanet Web Server 6.0 and Apache 1.3.12. In our experience, both Web servers, running on small systems with no more than 8 processors, gave comparable performance. On large systems, iWS6 scaled well beyond 8 processors, while the performance of Apache 1.3.12 suffered from its architecture and its lack of performance tuning options[1]. The abundant performance tuning options provided by iWS6, on the other hand, allowed us to properly set up the Web server to give good performance for each machine configuration. In this paper, we choose to study the performance of iWS6 only.

3.3.1 Session Creation Performance

Table 3 and Figure 7 show the SSL session creation performance of iWS6 running on a Sun Fire 6800 server with a variety of configurations[2], measured from the modified SPECweb99_SSL benchmark program mentioned in the previous subsection. The Sun Fire 6800 server can have up to 24 UltraSPARC® III processors on-line, and we selectively disabled some of the processors to measure the scalability of the Web server.

As the number of processors increased, the performance of the Web server increased, but not linearly, because of increased contention in system resources. Overall, the "software curve" in Figure 7 shows the Sun Fire 6800 server scaled well, as a 16-processor system produced approximately 13 times throughput of a one-processor system, an efficiency[3] of 81%, without the Sun CA1000.

1. Apache version 2 claims to perform better with its new thread model and tuning options. The Sun CA1000 support for Apache 2 is planned.

2. Note that two Sun CA1000 boards were used for the 24-processor configuration because the capacity of one Sun CA1000 is reached in the 16-processor case.

3. Efficiency is defined as (normalized performance)/(no. of processors), which characterizes how effective the processors are utilized to enhance application performance in a multiprocessor system.

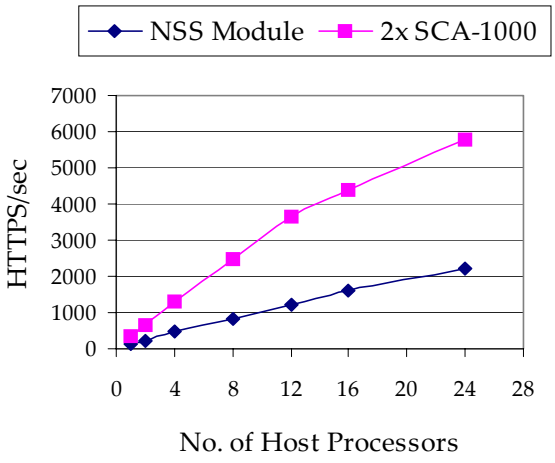


Figure 7: iWS6 SSL Session Creation Performance on Sun Fire 6800 Server.

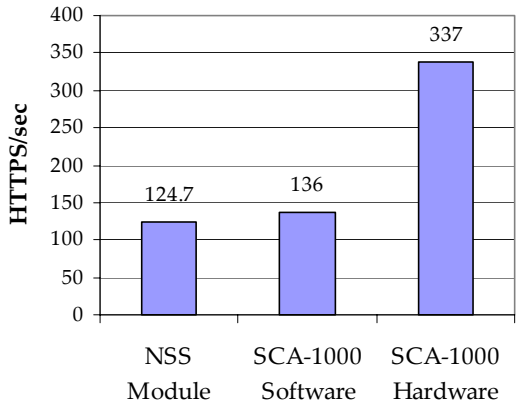


Figure 8: Comparison of iWS6 SSL Session Creation Performance on one-processor Sun Fire 6800 Server.

The Web server also scaled well when it is accelerated by the Sun CA1000, until the capacity of the Sun CA1000 board is reached for processing the RSA operations. With one Sun CA1000 board, the performance of the Web server scaled to 4385 HTTPS/sec (session creations per second). With two Sun CA1000 boards, 5777 HTTPS/sec was produced on the 24-processor system.

3.3.2 Session Creation Acceleration

Table 3 also lists the acceleration ratios that one Sun CA1000 board provided to each test configuration. The acceleration ratios are between 2.7 and 2.93 across the table.

In a preliminary study, we found that, with one processor, the NSS internal cryptographic module is capable of processing 208 RSA ops/sec, compared to the 245 RSA ops/sec that we measured with the Sun CA1000 library when the hardware is disabled (Section 2.2). Figure 8 compares the SSL session creation performance with different implementations of the 1024-bit RSA algorithms.

The one-processor system served at an average rate of 124.7 HTTPS/sec using the NSS crypto module. Thus, the system spent approximately 124.7/208=60% of the host processing power in the RSA computations and the remaining 40% processing power in other parts of HTTPS.

If the 60% of software RSA processing time is completely off-loaded from the host processor to the Sun CA1000 board, we should see that the Sun CA1000 solution to accelerate SSL session creation by a factor of 2.5, which is very close to the 2.7 acceleration ratio that we observed[1].

3.3.3 Peak Performance of Single Sun CA1000 Systems

The potential of a single Sun CA1000 board was fully exploited by the 16-processor system, giving a performance of 4385 HTTPS/sec. Recall that each HTTPS request from the client invoked a new SSL session that required one 1024-bit RSA operation from the server. Thus the

1. The discrepancy could be due to measurement errors or factors which are not obvious to us at this moment.

performance is bound by the Sun CA1000 board's capacity in handling RSA operations. This is confirmed as 13% of CPU idle time was observed in the 16-processor case. When the RSA bandwidth of the Sun CA1000 board was exhausted, some host processors had to wait to submit RSA jobs to Sun CA1000.

The 4385 HTTPS/sec performance was reported by the benchmark program in the end of a five-minute run to reflect the average throughput during the runtime. We validated the test by monitoring the Sun CA1000's performance counters, which showed the same average completion rate of RSA jobs.

3.3.4 Scalability - Multiple Sun CA1000 Boards

While a single Sun CA1000 board can effectively accelerate the Web server to an impressive 4385 HTTPS/sec performance, the scalability of the Sun CA1000 solution reached the next level as an average rate of 5778 HTTPS/sec was enabled by two Sun CA1000 boards in the case of 24 processors. Note that 15% idle time observed on the host processors was limited by workload generation capacity of the clients.

This experiment has proven that Web server performance can scale with multiple Sun CA1000 boards. The performance scalability ensures that Sun CA1000 will be an attractive, competitive, upgradable solution for the years to come.

3.3.5 Session Resumption Performance

Under the session resumption test, we allow the client and server to reuse an SSL session for unlimited times once the session is created. Since SSL session resumption does not require expensive public key cryptographic operations, we did not find Sun CA1000 to provide any acceleration for this test, nor did we find that Sun CA1000 impaired the session resumption performance.

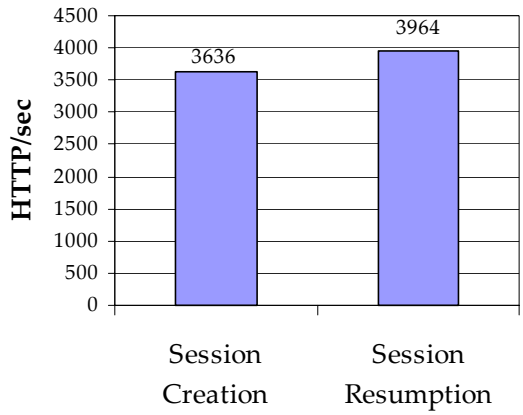


Figure 9: Comparison of iWS6 performance on 12-processor Sun Fire 6800 Server.

The performance gap between session resumption and session creation with the Sun CA1000 solution is relatively small, as illustrated by Figure 9. For the 12-processor system, session resumption performance is approximately 10% better than session creation performance with the Sun CA1000 solution. In fact, this summarizes the role of the Sun CA1000 solution in the SSL handshake phase: the Sun CA1000 effectively reduces the cost of SSL session creation to nearly the same as the cost of session resumption for our test configurations.

4. Tuning for Secure iWS

Before the advent of products like the Sun Crypto Accelerator 1000, it was difficult for users or even developers to envision a high-performance secure Web server that can carry out 4400 HTTPS/sec and 495 Mbps of 3DES encryption simultaneously. Running at this level of per-

formance, a Web server has to be efficient in handling not only the SSL protocol, but the HTTP, TCP/IP, and disk I/O operations as well. Performance tuning is a common practice for a high-performance Web server, as Web servers and operating systems often need to be optimized to handle high Web traffic [13]. This section shares our tuning experience.

4.1 Networking

Today, 100 Mbps networks are widely deployed, and gigabit networks are beginning to gain momentums. While network bandwidth traditionally has not been a very critical issue for secure Web servers, to take full advantage of the hardware encryption capability of the Sun Crypto Accelerator 1000, e.g. 495Mbps 3DES, the host machine must be equipped with proper network interface(s) to support the network bandwidth.

HTTPS incurs extra handshake messages for establishing each SSL connection. During the handshake phase, a full SSL handshake would require the server to receive 4 messages from the client and reply with 5 messages. While these handshake messages are relatively short, they put pressure on the network interface and the operating system. For example, to serve 4,000 HTTPS in one second, the server has to handle 16,000 incoming handshake messages and deliver 20,000 outgoing messages to the clients in one second. In addition, these messages incur TCP ACK packets. Even with high-speed network interfaces such as Sun GigabitEthernet/P 2.0 adapter, network interface can be the point of contention simply serving for SSL handshakes.

In our session creation tests, we found that a 100 Mbps network interface (Sun HME) can support up to 2900 HTTPS/sec of short file requests with no tuning applied. At the same time, the network interface generates 13,000 interrupts to the processor every second. Switching to a Gigabit network interface may not solve the problem unless the network interface is much more efficient in packet processing.

The short SSL handshake messages can be handled more efficiently through adjusting operating system (TCP/IP stack) tuning parameters that control *Nagle algorithm* and *deferred ack* [1][14]. However, optimizing for short messages can hurt the performance of bulk data transfer if it is not done properly. The users may need to profile the network traffic on their Web server since different Web applications may have different behaviors.

4.2 iWS6 Tuning Options

Performance tuning is a common practice for a high-performance Web server. Because the secure version of iPlanet Web Server is based on the non-secure version, most tuning parameters that affect HTTP protocol processing would also affect the performance of a secure Web server. The readers are referred to [13] for tuning options for non-secure iWS. In this subsection, we provide some general guidelines for tuning a secure Web server running iWS6. Notice that many tuning options require intimate knowledge of the server platform and the Web applications that run on the server.

4.2.1 Listen Sockets and Virtual Servers

By default, an iWS6 server instance listens on port 443 to handle requests from all IP addresses. Performance can be affected if contention occurs on the listen socket.

iWS6 supports the notion of virtual servers that share the same configuration information, except that each single virtual server can be associated with one listen socket. We found that virtual servers can be employed (via the Web server's **`server.xml`** file) to create multiple listen sockets for a single Web server instance to reduce socket contention.

4.2.2 Keep-Alive/Persistent Connection Requests

The use of persistent requests, also known as *keep-alive* requests, generally reduces the overhead of creating TCP connections for multiple HTTPS requests. Unlike regular requests, the response for a keep-alive request does not cause a close of the TCP connection.

Keep-alive may improve the performance of the Web server because reusing TCP connections reduces the overhead of establishing TCP connections.

4.2.3 RqThrottle

By default, iWS6 runs with multiple worker threads to process HTTP/HTTPS requests. The number of worker threads can be adjusted by setting a parameter called *RqThrottle* in the Web server's **`magnus.conf`** file. The more worker threads the Web server employs, the more concurrent requests the Web server can handle, but the number of worker threads needs to match the processing power of the system and the application to achieve optimal performance.

4.2.4 MaxProcs

By default, iWS6 runs with one process, within which multiple worker threads are used to process requests. This thread model is favored because threads typically have less overhead in context switching and migration than processes.

Under certain workload, running iWS with multiple processes can benefit performance as processes offer these advantages over threads: (1) processes could be bound to processors using the Solaris *pbind* facility to reduce migration, and (2) objects are often replicated in each process, reducing contentions for the locks that protect them.

A parameter in the Web server's magnus.conf file, called *MaxProcs*, can be set to specify the number of processes that the Web server runs with. Notice that RqThrottle adjusts the number of worker threads per process. Thus MaxProcs and RqThrottle jointly adjusts the total number of threads employed by the Web server.

4.2.5 Multiple Server Instances

Multiple independent Web server instances can simultaneously take advantage of the Sun CA1000 solution. Occasionally, as a way to minimize interferences between different Web applications and to allow performance to scale on a large system, it is better to partition the system into several processor groups and run separate Web server instances in different processor groups.

4.2.6 Thread Libraries

Lately, Solaris 8 Operating Environment (Solaris OE) offers two different flavors of thread libraries for serving multithreaded applications and handling

multithreading activities. Since iWS6 is a highly-parallel multithreaded application, its performance can depend significantly on the thread library. In our experience, we found that a secure iWS6 runs better with the alternate thread library (also known as *lwp*) in Solaris 8 OE. The alternate thread library will be the default thread library in the forthcoming Solaris 9 OE.

4.2.7 Memory Allocation

The performance of Memory Allocation Subsystem (MAS) in the iWS6 can affect the performance of a secure Web server. iWS6 comes with support for default MAS provided by Solaris OE or can be tuned to take advantage of other MAS, such as SmartHeap™ that is bundled with some versions of iWS6. In order for the web server uses SmartHeap, the Web server **`start`** script should be modified. A few lines in the **`start`** scripts should be un-commented to enable the use of SmartHeap.

Solaris 8 OE has a MAS available via a dynamically loadable library **`/usr/lib/libmtmalloc.so`** that has proven to be the fastest MAS available on Solaris OE. If SmartHeap is not enabled, then setting the environment variable **`LD_PRELOAD`** to **`/usr/lib/libmtmalloc.so`** immediately before the Web server is started, would allow the Web server to use this MAS.

4.2.8 SSL3SessionTimeout

This tunable in iPlanet Web Server's **`magnus.conf`**, is represented in seconds. The default value for this parameter is 24 hours. It controls the maximum time that an SSL version 3 (SSL3) session will be cached by the Web server. Since SSL Session caching can potentially consume memory (see the next tunable), this value should be tuned down for a site that experiences heavy new SSL session creation traffic. Tuning this value too low can potentially increase the load on the SSL session creation subsystem in the Web server.

4.2.9 SSLCacheEntries

This is a tunable in iPlanet Web Server's **`magnus.conf`**. It sets the number of SSL sessions that can be cached. The Web server may decide to remove a session entry from its session cache when the session cache is full, even if the **`SSL3SessionTimeout`** has not expired. If a Web site does not want to support resumable SSL sessions, then this parameter should be tuned to its minimum, i.e. 1. Note that it will use the default cache size of 10000 if it is set to 0.

5. Summary

In this paper, we presented various aspects of performance acceleration delivered by Sun Crypto Accelerator 1000. We evaluated the potential of the Sun CA1000 solution by measuring its performance with low-level RSA and 3DES benchmark programs. We have also shown that, the Sun CA1000 solution successfully accelerated SSL session creation for secure Web servers in our HTTPS benchmarks.

Performance tuning is often necessary for a Web server to achieve high performance. We have shared some tuning guidelines in this paper. We work with engineers of the Sun CA1000 Team, iPlanet and NSS to deliver the best performance of the integrated Sun/Sun CA1000/iWS solution. Based on our experience, we believe that the Sun CA1000 solution will be appreciated by

users as a cost-effective solution for running their applications securely without sacrificing performance.

6. Acknowledgement

This paper is a result of a project started jointly by the Cryptography group and the Performance and Availability Engineering (PAE) in Sun. Experts of iWS and NSS from iPlanet later joined and contributed to this project, as the co-optimization of the Web server and Sun CA1000 became a key focus. In PAE, we would like to thank the Sun CA1000 Engineering Team for providing software and hardware support to this performance study. We also would like to acknowledge iPlanet and AOL/NSS for investigating with us on issues related to iWS and SSL.

References

[1] E. Rescorla. *SSL and TLS: Design and Building Secure Systems*. Addison Wesley, 2001.

[2] W. Stallings. *Cryptography and Network Security - Principles and Practice*, 2nd Ed. Prentice Hall, 1998.

[3] The Network Security Services (NSS) Project Homepage. *Overview of NSS Open Source Crypto Libraries*. http://www.mozilla.org/projects/security/pki/nss/overview.html.

[4] The OpenSSL Project Homepage. http://www.openssl.org/.

[5] iPlanet Web Server Homepage. http://www.iplanet.com/products/iplanet_web_enterprise/home_web_server.html.

[6] The Apache Software Foundation. http://www.apache.org/.

[7] R. L. Rivest, A. Shamir, and L. M. Adelman. *On Digital Signature and Public Key Cryptosystems*, Technical Report, MIT/LCS/TR-212, MIT Laboratory for Computer Science, January 1979.

[8] ANSI. *American National Standard for FInancial Institution Key Management*. ANSI X9.17, 1985

[9] RSA Laboratories. *PKCS #11: Cryptographic Token Interface Standard*. Rev. 1. November 2001.

[10] ACME Labs. *http_load - multiprocessing http test client*. http://www.acme.com/software/http_load/.

[11] Mindcraft. *WebStone - The Benchmark for Web Servers*. http://www.mindcraft.com/webstone/.

[12] Standard Performance Evaluation Corporation. *SPECweb99 Design Document*. http://www.spec.org/osg/web99/docs/whitepaper.html, July 2000.

[13] N. Sun, A. Guzovskiy, P. Bhattacharya, and K.-T. Ko. *Web Server Performance under SPECweb99 Workload*. Sun Users Performance Group Conference (SUPerG), Oct. 2001, Amsterdam, Netherlands.

[14] J. Huang, S.-H Hung, G.-P. Musumeci, M. S. Klivansky, and K.-T. Ko. *Sun Fire Gigabit Performance Characterization*. Sun Users Performance Group Conference, Oct. 2001, Amsterdam, Netherlands.

Legal Notice