TAKING ADVANTAGE OF WIRE-SPEED CRYPTOGRAPHY

In Oracle WebLogic Server 10.3.x and Java™ Platform, Enterprise Edition 5 Application Environments

Ramesh Nagappan, ISV Engineering
Chad Prucha, Technical Marketing

Sun BluePrints™ Online

Part No 821-0801-10
Revision 1.0, 12/01/09

Table of Contents

The ubiquity of networked computing, combined with ever-growing demand for multimedia and rich Internet applications and Web services, has increased server workloads in the last two years. At the same time, the growing acceptance of virtualization technology has driven consolidation efforts and improved system utilization rates, straining many modern hardware platforms. Furthermore, security requirements to protect valuable information as it flows across the network are pushing IT organizations to implement data encryption capabilities. Unfortunately, these demands effectively cancel out the incremental clock speed gains and other computational power improvements provided by newer systems—gains that companies traditionally rely on for performance improvements.

IT organizations are searching for technologies to compensate for the additional overhead introduced by encryption. While useful, a network appliance approach introduces complexity and increases the power demands and costs of deployment projects. On the other hand, cryptography cards introduce a host of compatibility, configuration management, and software complexities. Foreseeing the need for purpose-designed hardware that can outpace workload demand, Sun introduced on-chip hardware cryptographic capabilities into its family of Sun servers with CoolThreads™ technology (Figure 1).

Provided in Sun servers with UltraSPARC® T1, T2, or T2 Plus processors, on-chip cryptographic acceleration eliminates the need for additional coprocessor cards, special licensing, network appliances, or power hungry add-on components. As a result, deploying Sun servers with CoolThreads technology in HTTP environments can help reduce system overhead, improve performance, and increase overall computing and network efficiency by improving responsiveness across the entire solution.

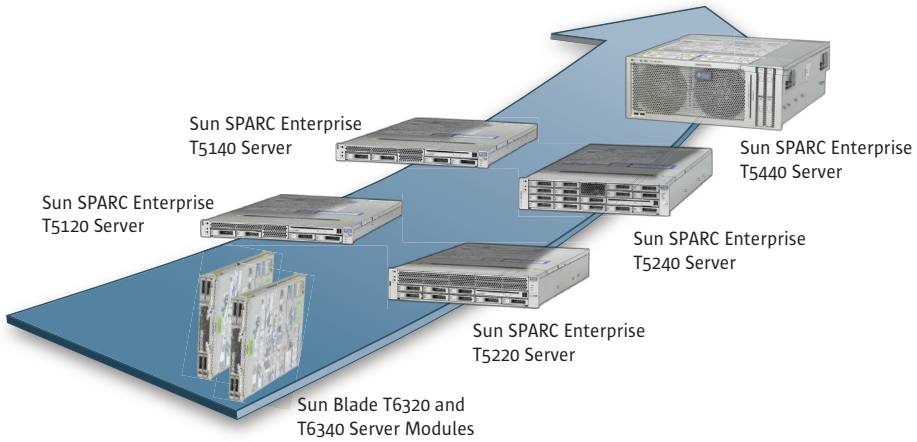

Figure 1. Sun servers with CoolThreads technology product family

This Sun BluePrints™ article provides an overview of how to off-load application security functions that include cryptographic operations in conjunction with Oracle® WebLogic Server, Java™ Platform, Enterprise Edition (Java EE platform)

application environments in order to accelerate performance while minimizing compromises. While this article includes many of the more arcane functional details of cryptography and related technologies, it emphasizes a simple approach to implementation that does not require the reader to become an expert in the subtleties of cryptographic techniques and Public Key Infrastructure (PKI).

Integrated Cryptographic Acceleration

Understanding the complex deployment scenarios that typically result from the need to keep information secure, Sun created the UltraSPARC T1, T2, and T2 Plus processors—processors that are targeted at throughput applications and are equipped with built-in hardware cryptographic units to simplify and accelerate cryptographic operations. The processors combine chip multiprocessing and hardware multithreading with an efficient instruction pipeline to enable chip multithreading (CMT). The resulting processor design provides multiple physical instruction execution pipelines and several active thread contexts per pipeline.

To meet the ever-increasing demand on cryptographic operations, the UltraSPARC T2 and T2 Plus processors use a unique System-on-a-Chip (SoC) design that incorporates additional cryptographic features as well as on-chip I/O and on-chip 10 Gigabit Ethernet networking capabilities to help improve performance (Figure 2).

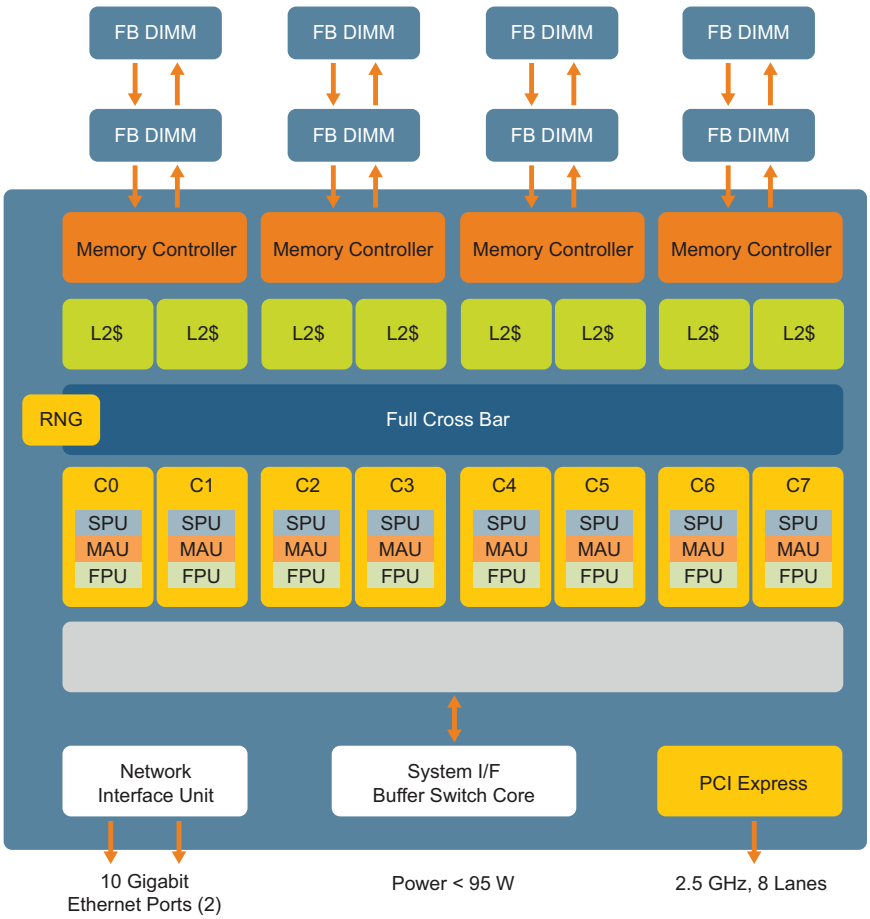

Figure 2. UltraSPARC T2 processor design

Rivest Shamir Adleman (RSA) operation is an important component of the Secure Sockets Layer (SSL) full handshake. Each core of the UltraSPARC T1, T2, and T2 Plus processors includes a Modular Arithmetic Unit (MAU) that supports RSA and Digital Signature Algorithm (DSA) operations. RSA operations utilize a compute-intensive algorithm that can be off-loaded to the MAU. Indeed, the MAU is capable of sustaining 14,000 RSA-1024 operations per second on a system with an UltraSPARC T1 processor, and more than 30,000 RSA-1024 operations per second on systems with an UltraSPARC T2 processor. Moving RSA operations to the MAU speeds SSL full handshake performance and frees the CPU to handle other computations.

The cryptographic capabilities of the UltraSPARC T1, T2, and T2 Plus processors can be accessed via the Solaris™ Cryptographic Framework (SCF). SCF provides cryptographic services for kernel-level and user-level consumers, as well as several software encryption modules. SCF continues to include Kernel SSL proxy (KSSL), which off-loads SSL processing from user applications and enables them to transparently take advantage of hardware accelerators, such as those available in UltraSPARC T1, T2, and T2 Plus processors.

Sun Servers with CoolThreads™ Technology

Sun servers with chip multithreading technology provide on-chip cryptographic acceleration support through a dedicated cryptographic accelerator, called the Niagara Crypto Provider (NCP), implemented on each processor core (Figure 3). The introductory UltraSPARC T1 processor included a NCP implementation that introduced public-key cryptographic mechanisms, including RSA and DSA algorithms. The latest UltraSPARC T2 and T2 Plus processors extend algorithm support by introducing symmetric key-based encryption and decryption mechanisms, such as Data Encryption Standard (DES), Triple DES (3DES), Advanced Encryption Standards (AES-128, AES-192, and AES-256), Ron's Code 4 (RC4), as well as hashing operations such as Message Digest 5 (MD5) algorithm, SHA1, SHA256, and Elliptic Curve Cryptography (ECC) mechanisms, such as the ECCp-160 and ECCb-163 algorithms.

In addition, UltraSPARC T2 processors provide an on-chip Random Number Generator (N2RNG) to support random number generation operations intended for cryptographic applications. In practice, the NCP uses the Solaris Cryptographic Framework (SCF) to allow user-level applications to off-load cryptographic operations and take advantage of NCP-based on-chip cryptographic acceleration.

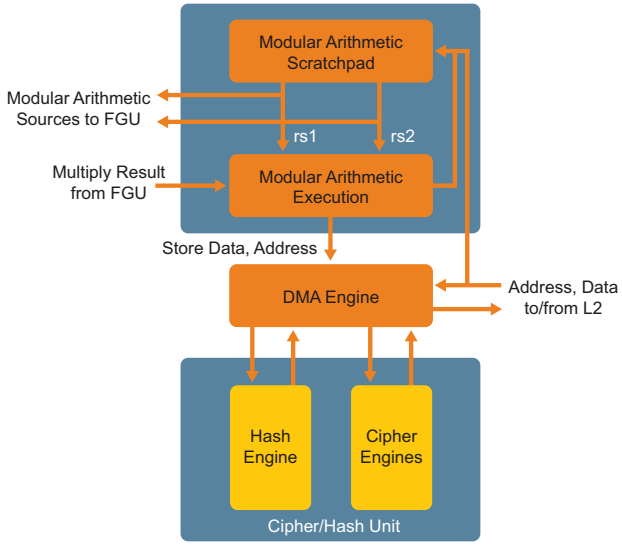

Figure 3. Sun servers with chip multithreading technology provide an on-chip cryptographic accelerator on each processor core

Understanding the Solaris™ Cryptographic Framework Library

The Solaris Cryptographic Framework library provides a set of cryptographic services for kernel-level and user-level consumers. Based on the PKCS#11 public key cryptography standard created by RSA Security, Inc., the framework provides a mechanism and API whereby both kernel- and user-based cryptographic functions can transparently use hardware accelerators configured on the system (Figure 4).

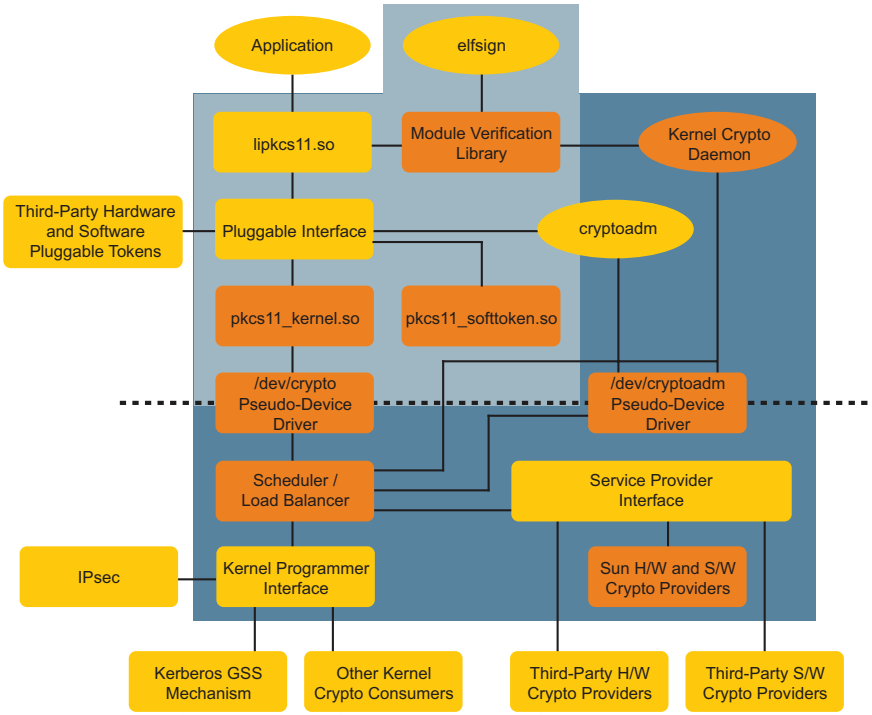

Figure 4. The Solaris Cryptographic Framework provides access to the hardware accelerators on Sun chip multithreading servers

Applications integrate with cryptographic service and token providers through a set of interfaces. Three types of cryptographic token providers are supported: a user-level provider (a PKCS#11 shared library), a kernel software provider, and a kernel hardware provider (a cryptographic hardware device such as NCP or N2CP). Applications looking to take advantage of hardware cryptography acceleration must go through the operating system kernel hardware provider. A user-level provider, `pkcall_kernel.so`, gives applications access to hardware devices through the kernel. To administer the cryptographic token provider and its supported mechanisms, the Solaris OS provides the `cryptoadm` utility that helps to identify and manage cryptographic providers with the following administrative tasks:

- Installing and uninstalling cryptographic providers
- Configuring the mechanism policy for each provider
- Displaying information about the framework

The `cryptoadm` utility provides a set of command-line options that are available for installing and uninstalling a cryptographic provider, and enabling and disabling the metaslot's features and mechanisms of the cryptographic token provider. The following commands can be used to explore the cryptographic capabilities of Sun chip multithreading servers with cryptographic accelerators.

- Display the list of installed software- and hardware-based cryptographic providers.

```
# cryptoadm list
```

- View the list of kernel hardware providers. On Sun servers with UltraSPARC T2 processors, the displayed list of kernel hardware providers is similar to the following list.

```
Kernel hardware providers:
ncp/0
n2rng/0
n2cp/0
```

- Use the `cryptoadm list -m` command to display the mechanisms provided by the available cryptographic providers. Use the following command to display the list of mechanisms provided by NCP on Sun servers with UltraSPARC T2 processors.

```
# cryptoadm list -m provider=ncp/0
ncp/0:
CKM_DSA, CKM_RSA_X_509, CKM_RSA_PKCS,
CKM_RSA_PKCS_KEY_PAIR_GEN, CKM_DH_PKCS_KEY_PAIR_GEN,
CKM_DH_PKCS_DERIVE, CKM_EC_KEY_PAIR_GEN,
CKM_ECDH1_DERIVE, CKM_ECDSA
```

- Disable or enable individual mechanisms, as needed. The following example uses the `enable` or `disable` sub-command to demonstrate the difference in encryption speed of hardware and software solutions.

- Disable a selected mechanism policy (**AES_CBC_PAD**) on an installed cryptographic provider

```
# cryptoadm disable provider=/usr/lib/security \
  \$ISA/pkcs11_softtoken.so mechanism=CKM_AES_CBC_PAD
```

- Enabled a selected mechanism policy on an installed cryptographic provider

```
# cryptoadm enable provider=n2cp/0 mechanism=CKM_AES_CBC_PAD
```

Using NCP with the Java™ Development Kit

With the Java™ Development Kit 5.0 (JDK™ software) and later releases, Java technology-based applications can access hardware cryptographic tokens—such as Secure Sockets Layer (SSL) accelerators, Smartcards, and HSMs—using the SunPKCS11 cryptographic provider. Bundled with the JDK software, the SunPKCS11 provider is a Java PKCS#11 implementation that integrates with underlying PKCS#11 implementations supplied by cryptographic providers and the Solaris Cryptographic Framework. It does not provide its own cryptographic algorithms.

In a typical installation, the Java runtime environment is pre-configured to make use of the SunPKCS#11 provider. Look at the *$JAVA_HOME/jre/lib/security/java.security* properties file and make sure it identifies SunPKCS#11 as the default provider.

```
security.provider.1=sun.security.pkcs11.SunPKCS11  ${java.home}/lib/security/sunpkcs11-solaris.cfg
```

The *$JAVA_HOME/jre/lib/security/sunpkcs11-solaris.cfg* file contains the configuration information used by the SunPKCS11 provider for accessing the Solaris Cryptographic Framework. To help ensure Java technology-based applications benefit from cryptographic acceleration, the *sunpkcs11-solaris.cfg* file allows users to disable the soft-token (user-level provider) mechanisms that can be delegated to the underlying hardware-based cryptographic token provider and the supported mechanisms exposed by its metaslot. This enables Java technology-based applications and Java EE server hosted Web applications and XML Web services to automatically delegate their cryptographic-intensive operations to the underlying cryptographic provider via the Solaris Cryptographic Framework. As a result, Java EE application servers can take advantage of NCP for performing cryptographic operations, particularly in the following applied scenarios:

- Transport-layer security of Web applications (SSL)
- Java Remote Method Invocation (Java RMI) over Internet Inter-Orb Protocol (IIOP) with SSL

Techniques for Accessing Hardware Acceleration

The hardware cryptography features available on Sun chip multithreading servers can be accessed in a variety of ways, depending on circumstances and technical requirements.

The Solaris OS tightly integrates with the specialized features of Sun's chip multithreading servers. To access the hardware accelerators, the operating system provides mechanisms that use Solaris Cryptographic Framework kernel modules, such as Kernel SSL (KSSL) and IPSec. PKCS#11 interfaces are available for user applications. As a result, service-oriented architectures and Java EE applications can take advantage of the PKCS#11 providers in the Java Cryptographic Extension (JCE) framework to off-load specialized cryptographic workloads to the hardware.

Using KSSL as an SSL Proxy

KSSL is a Solaris OS kernel module that acts as a server-side SSL protocol for off-loading operations such as SSL/TLS-based communication, SSL/TLS termination, and reverse proxies for end-user applications. KSSL takes advantage of the Solaris Cryptographic Framework to acts as an SSL proxy server, performing complete SSL handshake processing in the Solaris OS kernel. It uses the underlying hardware cryptographic providers—SSL accelerators, PKCS#11 keystores, and HSMs—to enable SSL acceleration and support secure key storage. Key aspects of KSSL include:

- Helps to introduce—non-intrusively—an SSL proxy server for Web servers, Java EE application servers, and applications that do not implement SSL.
- Listens to secured requests on the designated SSL port (http://:443) and renders cleartext traffic via a reverse proxy port (http://:8080) for underlying Web or application servers. All SSL operations, including the SSL handshake and session state, are performed asynchronously in the Solaris OS kernel and without the knowledge of the target application server.
- Automatically uses the Solaris Cryptographic Framework for off-loading operations to the underlying hardware cryptographic providers. No extra effort is required.
- Manages all SSL certificates independently and supports most standard formats, including PKCS12 and PEM. Key artifacts can be stored in a flat file or a PKCS#11 conforming keystore to help ensure the protection of private keys.
- Supports the use of Solaris Zones. Each IP-identified zone can be configured with a KSSL proxy.
- Delivers 25% to 35% faster SSL performance compared to traditional SSL configurations used in popular Web servers and Java application servers, such as Oracle WebLogic Server, Sun GlassFish™ Enterprise Server, and IBM WebSphere Application Server.

Configuring KSSL for Off-loading Oracle WebLogic Server SSL

Using the KSSL kernel module as an SSL proxy requires obtaining and installing a certificate from a Certificate Authority. However, a self-signed certificate also can be used.

1. Create a self-signed certificate.

```
# mkdir /etc/pki
# cd /etc/pki
# /usr/sfw/bin/openssl req -x509 -nodes -days 365 -subj \
  "/C=US/ST=State/L=City/CN=serverhostname" \
  -newkey rsa:1024 -keyout /etc/certkeys/key00.pem -out \
  /etc/certkeys/cert00.pem
```

2. Place all certificate artifacts in a single file.

```
# cat cert00.pem key00.pem > mySSLCerts.pem
```

3. Move to the */etc/certkeys* directory and execute the following command.

```
# chown 600 /etc/mySSLCerts.pem
```

4. Configure the KSSL proxy and its redirect HTTP cleartext port. The WebLogic Server listen port, or cleartext port, is port 7001.

```
# ksslcfg create -f pem -i /etc/pki/mySSLCerts.pem \
  -x 7001 -p /etc/pki/passwordfile serverhostname 443
```

5. Use the Service Management Facility (SMF) to verify that KSSL is in maintenance mode.

```
# svcs -a | grep "kssl"
```

6. Use a Web browser to check that the application server listens to the KSSL secured port. Go to *https://myhostname.com*

If a PKCS#11 keystore, NSS soft token keystore, or third-party HSM is used to ensure the security of private-key, KSSL must be configured with the following options.

```
# ksslconfig create -f pkcs11 -d token_directory \
  -T token_label -C certificate_subject -x proxy_port \
  target_server_name port
```

Configuring Oracle WebLogic Server for SSL Acceleration

The following steps explain how to configure Oracle WebLogic Server for SSL acceleration using the on-chip capabilities of Sun servers with chip multithreading capabilities.

1. Configure Oracle WebLogic Server to listen for SSL. Be sure to install SSL certificates appropriately. Follow the SSL configuration guidelines specified in the *Oracle Fusion Middleware - Securing WebLogic Web Services for Oracle WebLogic Server 11g Release 1 (10.3.1)* guide.

2. Confirm that Oracle WebLogic Server is listening and responds over the secured port.
3. Enable hardware acceleration by editing the Java Runtime Environment (JRE™ software) SunPKCS#11 provider configuration file located at *weblogic-java-home/jre/lib/security/sunpkcs11-solaris.cfg*. This file contains vital attributes for allowing Oracle WebLogic Server to access the hardware acceleration facilities. Mechanisms and attributes supported by the underlying hardware accelerator can be enabled or disabled in the SunPKCS11 configuration file.
4. Include the RSA mechanisms in the `disabledMechanisms` list of the SunPKCS11 softoken. Doing so forces the RSA mechanisms to be performed by the NCP.

```
name = Solaris
description = SunPKCS11 accessing Solaris Cryptographic
Framework
library = /usr/lib/$ISA/libpkcs11.so
handleStartupErrors = ignoreAll
attributes = compatibility
disabledMechanisms = {
  CKM_MD2
  CKM_MD5
  CKM_SHA_1
  CKM_SHA256
  CKM_SHA384
  CKM_SHA512
  CKM_DSA_KEY_PAIR_GEN
  CKM_SHA1_RSA_PKCS
  CKM_MD5_RSA_PKCS
  CKM_SHA256_RSA_PKCS
  CKM_SHA384_RSA_PKCS
  CKM_SHA512_RSA_PKCS
  CKM_TLS_KEY_AND_MAC_DERIVE
  CKM_RSA_PKCS_KEY_PAIR_GEN
  CKM_SSL3_KEY_AND_MAC_DERIVE
}
```

5. Restart Oracle WebLogic Server.
6. In addition, the `cryptoadm` administration utility provided in the Solaris OS can be used to verify and ensure the off-loading of cryptographic operations, install or uninstall software token providers, and enable or disable hardware token providers and mechanisms.
   - Display the mechanisms supported by the installed providers

```
# cryptoadm list -m
```

   - Uninstall the soft token implementation

```
# cryptoadm uninstall provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
```

   - Install the soft token implementation

```
# cryptoadm install provider=/usr/lib/security/\$ISA/pkcs11_softtoken.so
```

   - Disable the `CKM_MD5_RSA_PKCS` and `CKM_SHA1_RSA_PKCS` mechanisms from the soft token implementation

```
# cryptoadm disable provider=/usr/lib/security/\$ISA/pkcs11
_softtoken.so mechanism=CKM_MD5_RSA_PKCS, CKM_SHA1_RSA_PKCS
```

- Enable the `CKM_MD5_RSA_PKCS` and `CKM_SHA1_RSA_PKCS` mechanisms from the soft token implementation

```
# cryptoadm enable provider=/usr/lib/security/\$ISA/pkcs11_
softtoken.so mechanism=CKM_MD5_RSA_PKCS, CKM_SHA1_RSA_PKCS
```

Performance Characteristics

In order to demonstrate the performance improvements made possible by on-chip cryptographic acceleration, Sun engineers ran a series of tests that used Faban to drive a workload against the application. The Faban harness is infrastructure for hosting workloads and automating execution, and was installed on all systems in the configuration. Designed to drive the application and to simulate real world interactions by a number of concurrent users, Faban gathers data from a test run and presents it to users in graphical form through a Web interface. It can manage the systems used in a workload test, including the system under test, load generation systems, and even database and cache servers commonly used in complex workload testing.

The testing effort was designed to demonstrate that any workload—even the lightest workload with only 10 users ramping up usage over 10 minutes and sustained activity for 10 minutes running a simple application—can benefit greatly from investing a few minutes in configuring the system to off-load cryptographic operations to the MAUs in the server. During the test, 10 virtual users access a simple set of Java technology-based applications repeatedly. These applications identified the Internet Protocol (IP) address of the client, dynamically rendered a “Hello World” message, and served a simple 16 KB JPEG file. The test started with one user. One user was added at each minute of the test, with each user sustaining activity for 10 minutes by accessing a basic “hello world” Web application served by Oracle WebLogic Server.

Figures 5 shows the results obtained. These results show that throughput and transaction rates improved application responsiveness to near original unencrypted speeds when hardware cryptography was enabled.

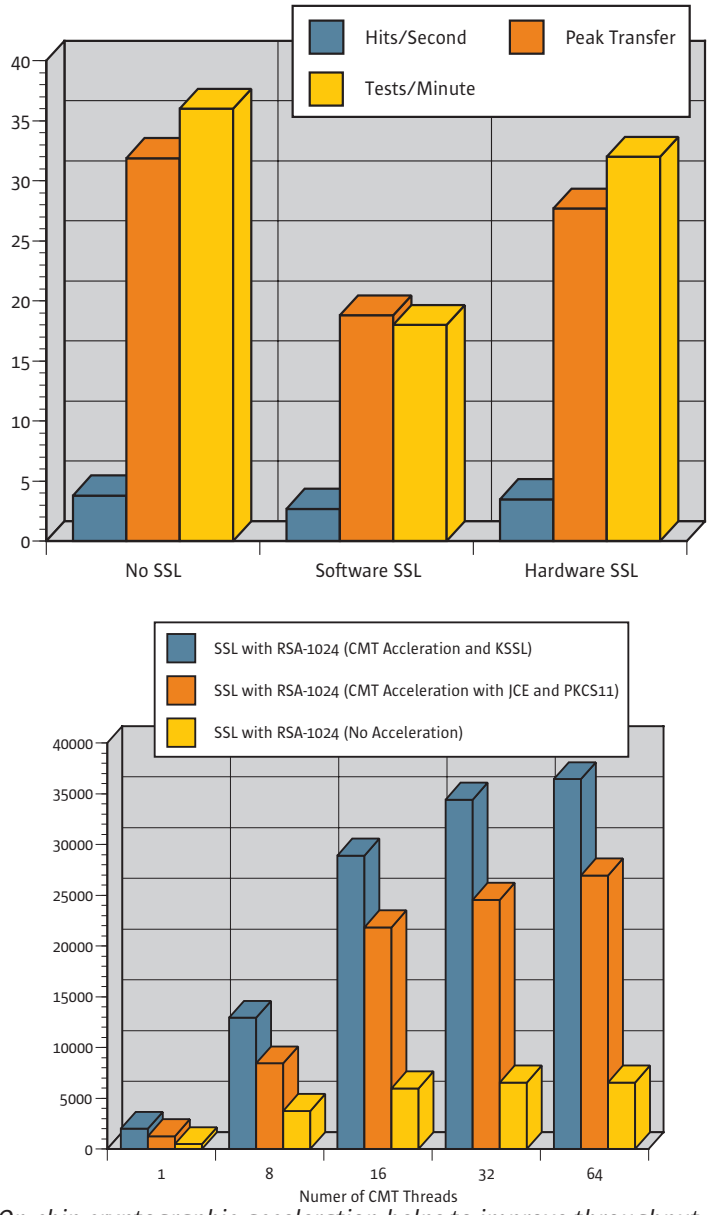

Figure 5. On-chip cryptographic acceleration helps to improve throughput, transaction rates, and application responsiveness

About the Authors

A Security Architect at Sun, Ramesh Nagappan has extensive experience with, and remains focused on, securing applications, XML Web services, and identity management technologies. He is a coauthor of *Core Security Patterns* and four other books on topics related to the Java EE platform, EAI, and XML Web services. Ramesh speaks frequently at industry conferences and contributes to industry standards and open-source initiatives on Java technology, XML, and security. Currently, Ramesh works on security-sensitive and citizen-scale applications, identity

assurance, and physical and logical access control solutions using PKI, smartcards, and biometrics for projects aligned with government, intelligence, law enforcement, and national security related organizations. Ramesh can be reached at *http://www.coresecuritypatterns.com/blogs*

Chad Prucha has worked with cryptographic acceleration gear since 2001, and has extensive operational experience in security, virtualization, datacenter design, and telepresence disciplines. He has worked as a professional services practice lead and senior architect for several years, and has headed consolidation and optimization projects for public utilities, large-scale manufacturing, health and heavy shipping concerns. Chad makes an effort to train and certify in several competing products in order to better determine the ideal technology for solving a given problem. Relatively new to Sun, Chad works to prove and communicate the power and flexibility of Sun servers with CoolThreads technology.

References

Table 1. References for more information.

Web Sites	
Solaris Operating System	sun.com/solaris
Solaris 10 OS Reference Manual Collection	http://docs.sun.com/app/docs/coll/40.10
Sun Servers with CoolThreads Technology	sun.com/servers/coolthreads
Sun BluePrints Articles	
Using the Cryptographic Accelerators in the UltraSPARC T1 and T2 Processors	sun.com/blueprints/0306/819-5782.pdf

Ordering Sun Documents

The SunDocs^{SM} program provides more than 250 manuals from Sun Microsystems, Inc. If you live in the United States, Canada, Europe, or Japan, you can purchase documentation sets or individual manuals through this program.

Accessing Sun Documentation Online

The docs.sun.com Web site enables you to access Sun technical documentation online. You can browse the docs.sun.com archive or search for a specific book title or subject. The URL is http://docs.sun.com

To reference Sun BluePrints Online articles, visit the Sun BluePrints Online Web site at: http://www.sun.com/blueprints/online.html

Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, CA 95054 USA Phone 1-650-960-1300 or 1-800-555-9SUN (9786) Web sun.com

Printed in USA 12/09