



Taming Your Emu to Improve Application Performance

Richard McDougall, PAE

Sun BluePrints™ OnLine—February 2004



<http://www.sun.com/blueprints>

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95045 U.S.A.
650 960-1300

Part No. 817-5489-10
Revision 1.0, 1/29/04
Edition: February 2004

Copyright 2004 Sun Microsystems, Inc. 4150 Network Circle, Santa Clara, California 95045 U.S.A. All rights reserved.

Sun Microsystems, Inc. has intellectual property rights relating to technology described in this document. In particular, and without limitation, these intellectual property rights may include one or more patents or pending patent applications in the U.S. or other countries.

This product or document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this product or document may be reproduced in any form by any means without prior written authorization of Sun and its licensors, if any. Third-party software, including font technology, is copyrighted and licensed from Sun suppliers.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Sun BluePrints, Solaris, Sun Enterprise, SunONE, and Sun Fire are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the US and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

The OPEN LOOK and Sun™ Graphical User Interface was developed by Sun Microsystems, Inc. for its users and licensees. Sun acknowledges the pioneering efforts of Xerox in researching and developing the concept of visual or graphical user interfaces for the computer industry. Sun holds a non-exclusive license from Xerox to the Xerox Graphical User Interface, which license also covers Sun's licensees who implement OPEN LOOK GUIs and otherwise comply with Sun's written license agreements.

U.S. Government Rights—Commercial use. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the Far and its supplements.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2004 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, Californie 95045 Etats-Unis. Tous droits réservés.

Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a. Le logiciel détenu par des tiers, et qui comprend la technologie relative aux polices de caractères, est protégé par un copyright et licencié par des fournisseurs de Sun.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque enregistrée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company Ltd.

Sun, Sun Microsystems, the Sun logo, Sun BluePrints, Solaris, Sun Enterprise, SunONE, et Sun Fire sont des marques de fabrique ou des marques déposées, ou marques de service, de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

L'interface d'utilisation graphique OPEN LOOK et Sun™ a été développée par Sun Microsystems, Inc. pour ses utilisateurs et licenciés. Sun reconnaît les efforts de pionniers de Xerox pour la recherche et le développement du concept des interfaces d'utilisation visuelle ou graphique pour l'industrie de l'informatique. Sun détient une licence non exclusive de Xerox sur l'interface d'utilisation graphique Xerox, cette licence couvrant également les licenciés de Sun qui mettent en place l'interface d'utilisation graphique OPEN LOOK et qui en outre se conforment aux licences écrites de Sun.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REpondre A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.



Please
Recycle



Adobe PostScript

Taming Your Emu to Improve Application Performance

One of my favorite features of the Solaris™ 9 Operating System (Solaris OS) is multiple page size support (MPSS). Why? Because it's one of the easiest ways to achieve a significant performance gain for a large range of applications.

Memory intensive applications that have a large working set often perform suboptimally on the Solaris OS without a little tuning. This is because they make inefficient use of the microprocessor's translation lookup buffer (TLB) facility. MPSS allows you to exploit larger page sizes for the microprocessor's memory management unit (MMU, or M-Emu), which allows more efficient use of the TLB, ultimately resulting in improved application performance.

Applications most likely to benefit from MPSS typically have working sets greater than a few hundred megabytes, and are memory intensive. Because the TLB can only hold a few hundred translations at a time, these applications typically overflow the microprocessors TLB. The Solaris kernel services overflows from the UltraSPARC™ TLB, which can result in a significant amount of system-software time.

There is a catch, however. Regular performance tools like `mpstat`, `sar`, and `vmstat` do not report the time spent processing TLB overflows (we refer to them as TLB misses) as system time. Instead, they report an application's TLB misses as user time. This can be quite misleading because it can appear that the CPU is spending all of its time running an application when, in fact, it is really spending a large fraction of time in the kernel.

This Sun BluePrints™ OnLine article explains how to engage the MPSS feature on the Solaris OS and how to analyze its effect on performance. It briefly explains the hardware feature being exploited, how to measure the usage of this hardware feature with standard Solaris OS tools, and the ways by which users and programmers can invoke the feature. The article doesn't explain the underlying theory in great detail, but provides working examples and references to help you locate additional information on the subject.

Catching Your Emu With trapstat

To help you determine how frequently an application overflows the TLB, the Solaris 9 OS introduces a new tool called `trapstat`. This tool provides an easy way to measure the time spent in the kernel servicing TLB misses. Using the `-t` option, `trapstat` reports how many TLB misses occur and what percentage of the total CPU time is spent processing TLB misses.

The `-t` option provides first-level summary statistics. Time spent servicing TLB misses is summarized in the lower right corner of the report. As shown in the following example, 46.2 percent of the total execution time is spent servicing TLB misses. The TLB miss detail is broken down to show TLB misses incurred in the data portion of the address space (dTLB) and for the instruction portion of the address space (iTLB). Data are also provided for user misses (u) and kernel-mode misses (k). We are primarily interested in the user-mode misses because our application likely runs in user mode.

```
sol9# trapstat -t 1 111
```

cpu	m	itlb-miss	%tim	itsb-miss	%tim	dtlb-miss	%tim	dtsb-miss	%tim	%tim
0	u	1	0.0	0	0.0	2171237	45.7	0	0.0	45.7
0	k	2	0.0	0	0.0	3751	0.1	7	0.0	0.1
=====										
	ttl	3	0.0	0	0.0	2192238	46.2	7	0.0	46.2

For additional detail, you can use the `-T` option to provide a per-page-size breakdown. In our example, `trapstat -T`, shows that almost all of the misses occurred on 8-kilobyte pages.

```
sol9# trapstat -T 1
```

cpu	m	size	itlb-miss	%tim	itsb-miss	%tim	dtlb-miss	%tim	dtsb-miss	%tim	%tim
0	u	8k	30	0.0	0	0.0	2170236	46.1	0	0.0	46.1
0	u	64k	0	0.0	0	0.0	0	0.0	0	0.0	0.0
0	u	512k	0	0.0	0	0.0	0	0.0	0	0.0	0.0
0	u	4m	0	0.0	0	0.0	0	0.0	0	0.0	0.0

0	k	8k	1	0.0	0	0.0	4174	0.1	10	0.0	0.1
0	k	64k	0	0.0	0	0.0	0	0.0	0	0.0	0.0
0	k	512k	0	0.0	0	0.0	0	0.0	0	0.0	0.0
0	k	4m	0	0.0	0	0.0	0	0.0	0	0.0	0.0
=====											
	ttl		31	0.0	0	0.0	2174410	46.2	10	0.0	46.2

We can conclude from this output that the application could potentially run almost twice as fast if we could eliminate the majority of the TLB misses. Our objective in using the mechanisms discussed in the following paragraphs is to minimize the user-mode data TLB misses (dTLB) by instructing the application to use larger pages for its data segments. Typically, data misses are incurred in the program's heap or stack segments. We can use the Solaris 9 OS MPSS commands to direct the application to use 4-megabyte pages for its heap, stack, or anonymous memory mappings.

Programming Your Emu

Prior to the Solaris 9 OS, databases were typically the only users of larger page sizes through the intimate shared memory (ISM) facility provided by the `SHM_SHARE_MMU` option of `shmat(2)`. The Solaris 9 OS provides three methods for advising preferred page sizes for applications:

- Wrapper program, `ppgsz(1)`
- Preload library, `libmpss.so.1`
- Programmatic interface, `memcntl(2)`

UltraSPARC supports four page sizes; 8 kilobyte (default), 64 kilobyte, 512 kilobyte, and 4 megabyte.

Enabling large pages for an application's heap is quite straightforward. Simply wrap the target binary with the `ppgsz` command and the appropriate options, as follows:

```
sol9# ppgsz -o heap=4M ./testprog &
```

After starting the target program, you can check to see how many large pages were allocated with the `pmap -sx` command. In the following example, the majority of the heap has been allocated with 4-megabyte pages:

```

sol19# pmap -sx `pgrep testprog`
2953:  ./testprog
  Address Kbytes  RSS      Anon  Locked Pgsz Mode  Mapped File
00010000      8      8      -    -    8K r-x-- dev:277,83 ino:114875
00020000      8      8      8    -    8K rwx-- dev:277,83 ino:114875
00022000   3960   3960   3960    -    8K rwx-- [ heap ]
00400000  131072  131072  131072    -    4M rwx-- [ heap ]
FF280000    120    120     -    -    8K r-x-- libc.so.1
FF29E000    136    128     -    -    - r-x-- libc.so.1
FF2C0000     72     72     -    -    8K r-x-- libc.so.1
FF2D2000    192    192     -    -    - r-x-- libc.so.1
FF302000    112    112     -    -    8K r-x-- libc.so.1
FF31E000     48     32     -    -    - r-x-- libc.so.1
FF33A000     24     24     24    -    8K rwx-- libc.so.1
FF340000     8      8      8    -    8K rwx-- libc.so.1
FF390000     8      8      -    -    8K r-x-- libc_psr.so.1
FF3A0000     8      8      -    -    8K r-x-- libdl.so.1
FF3B0000     8      8      8    -    8K rwx-- [ anon ]
FF3C0000    152    152     -    -    8K r-x-- ld.so.1
FF3F6000     8      8      8    -    8K rwx-- ld.so.1
FFBFA000     24     24     24    -    8K rwx-- [ stack ]
-----
total Kb  135968  135944  135112    -

```

The `ppgsz` command is the simplest to use, but the specified page size preferences will not be inherited across `exec(2)` calls. If your program execs another and you want the page size preferences, you should use the `mpss.so.1` preload library to make this happen.

The `mpss.so.1` shared object in `/usr/lib` provides a means by which the preferred stack or heap page size can be automatically enforced for launched processes and their descendants. The library has an the advantage over the wrapper in that page sizes are inherited across `exec(2)`. To enable `mpss.so`, set `LD_PRELOAD` in the environment [see `ld.so.1(1)`], along with one or more MPSS environment variables. For example, you might use either of the following variables to specify the preferred page sizes for the heap and stack, respectively. The specified page size(s) are applied to all created processes.

`MPSSHEAP=size`

`MPSSSTACK=size`

For example, using `sh` or `ksh`:

```
sol9# LD_PRELOAD=$LD_PRELOAD:mpss.so.1 MPSSHEAP=4M
sol9# ./myprog
```

Or, using `csh` or `tcsh`:

```
sol9# setenv LD_PRELOAD $LD_PRELOAD:mpss.so.1
sol9# setenv MPSSHEAP 4M
sol9# ./myprog
```

To confirm that the application is now running more efficiently, run `trapstat` again. Ideally, the percentage of time spent in the kernel will be lower. In this example, the percentage of time has dropped from 46.2 percent to 0.2 percent!

```
sol9# trapstat -T 1
```

cpu	m	size	itlb-miss	%tim	itsb-miss	%tim	dtlb-miss	%tim	dtsb-miss	%tim	%tim
0	u	8k	30	0.0	0	0.0	221	0.1	0	0.0	0.1
0	u	64k	0	0.0	0	0.0	0	0.0	0	0.0	0.0
0	u	512k	0	0.0	0	0.0	0	0.0	0	0.0	0.0
0	u	4m	0	0.0	0	0.0	0	0.0	0	0.0	0.0
0	k	8k	1	0.0	0	0.0	4271	0.1	10	0.0	0.1
0	k	64k	0	0.0	0	0.0	0	0.0	0	0.0	0.0
0	k	512k	0	0.0	0	0.0	0	0.0	0	0.0	0.0
0	k	4m	0	0.0	0	0.0	0	0.0	0	0.0	0.0
=====											
ttl			31	0.0	0	0.0	4491	0.2	10	0.0	0.2

Things to Watch Out For

TLB sizes vary between UltraSPARC versions. The UltraSPARC I and II microprocessors (143 megahertz–480 megahertz) data TLB has 64 entries that supports all four page sizes. User applications can use any of the four page sizes available.

However, the 750-megahertz UltraSPARC III microprocessor includes several small TLBs and one large 512-entry TLB that supports only 8-kilobyte entries. Thus, use of large pages is typically not beneficial on 750 megahertz UltraSPARC III systems.

The 900-megahertz UltraSPARC III onwards has two large 512 entry TLBs, one of which is configured automatically based on the most common page sizes in a process address space. A process using one large page size in addition to the base page size (8 kilobytes) will have one of its large TLBs automatically programmed to enable the large page size when there are eight or more pages using the larger page size within the process. Thus, large pages can be used very effectively on UltraSPARC III. However, since the large TLBs can be configured only for one page size per process at a time, only two page sizes should be used concurrently (typically 8 kilobytes plus one other size).

Another point of interest is large page fragmentation. Large pages require contiguous physical memory, and if smaller pages fragment physical memory, a request for larger pages might not be fulfilled. When the system boots, a sizeable pool of large pages is available, but if a significant number of smaller page sizes are allocated and locked down, an application requesting larger page sizes might not have its request fulfilled with the larger page size. The application's request for memory is still satisfied, it just might not be allocated with its preferred larger page size. In this case, it will still run, but its mappings will be backed by smaller page sizes. For this reason, it is advisable to use `pmap -xs` to ensure that larger page sizes are indeed allocated.

There is, however, a way to minimize the amount of fragmentation. The Solaris 9 kernel will attempt to relocate pages in order to create the required contiguous memory; this works for all but locked pages. By enabling the kernel cage, you can vastly improve this situation because the kernel will be allocated from a small contiguous range of memory, thus minimizing the fragmentation of other pages within the system.

The kernel cage is enabled on Sun Enterprise™ 10K and Sun Fire™ 15K systems, but not on other systems. You can enable it by setting the `kernel_cage_enable` in `/etc/system`, as follows:

```
set kernel_cage_enable=1
```

Commands and API Quick Reference

The following table summarizes the commands and tools described in this article.

<code>ppgsz</code>	An administrative wrapper program for advising page size preferences. The <code>ppgsz</code> command is not inherited across <code>exec()</code> by a new program.
<code>pmap -sx</code>	A utility to print the MMU page size for each mapping in the program.
<code>mpss.so.1</code>	A preload library enabled by setting <code>LD_PRELOAD=mpss.so.1</code> for advising page size preferences for existing applications. Advice is held across <code>exec()</code> of a new program.
<code>trapstat -t</code> <code>trapstat -T</code>	A tool for measuring the amount of time spent servicing TLB misses.
<code>cc</code>	New options are included in the Sun™ ONE Studio 8 compiler <code>-xpagesize_heap</code> and <code>-xpagesize_stack</code>

About the Authors

Richard has over 15 years of UNIX® experience including application design, kernel development, and performance analysis. Richard specializes in operating system tools and architecture.

Ordering Sun Documents

The SunDocsSM 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`