# Large Page Performance

ESX Server 3.5 and ESX Server 3i v3.5

VMware® ESX Server supports the use of large pages inside virtual machines. The large-page support enables server applications to establish large-page memory regions. Memory address translations use translation lookaside buffers (TLB) inside the CPU. The use of large pages can potentially increase TLB access efficiency and thus improve program performance. This study examines the performance of this feature.

Most modern processors support memory pages of large sizes, called "large pages" or "super pages." For example, the x86-64 architecture supports the large page size of 2MB in addition to small 4KB pages. X86-64 and x86 PAE support large page size of 2MB; x86 non-PAE supports 4MB. The large translation lookaside buffer (TLB) covers a larger memory range than the small TLB, thus the use of large pages can possibly reduce TLB misses. As a result, many applications can benefit from large pages. For example, an Oracle database uses many large shared memory regions and can potentially benefit from large pages.

VMware ESX Server 3.5 and VMware ESX Server 3i v3.5 introduce 2MB large page support to the virtualized environment. In earlier versions of ESX Server, guest operating system large pages were emulated using small pages. This meant that, even if the guest operating system was using large pages, it did not get the performance benefit of reducing TLB misses. The enhanced large page support in ESX Server 3.5 and ESX Server 3i v3.5 enables 32-bit virtual machines in PAE mode and 64-bit virtual machines to make use of large pages. If an application can benefit from large pages on a native machine, it can potentially achieve similar performance improvement in a virtual machine running on an ESX Server host with either of these two products as the hypervisor. It is worth configuring your guest operating system and the application to use large pages.

Large pages improve the performance of many applications, but misuse of large pages may also hurt performance in some situations. The potential for performance degradation is a result of the fact that the number of large TLB entries is usually smaller than the number of small TLB entries. If the working set of an application is scattered over a wide range of address space, the application is likely to experience thrashing of a relatively small number of large TLB entries. This thrashing may result in worse overall performance with higher TLB miss rates.

This performance study examines the performance of large page support in VMware ESX Server, using SPECjbb2005. For more information about the benchmark, see http://www.spec.org/jbb2005/. We conducted the measurements on three different guest operating systems.

The study covers the following topics:

- "Performance Evaluation" on page 2
- "Conclusion" on page 3
- "Appendix A: Enabling Large Page Support in Guest Operating Systems" on page 3
- "Appendix B: Enabling Large Page Support in ESX Server" on page 4
- "Appendix C: Enabling Large Page Support in Applications" on page 5

# Performance Evaluation

This study shows the performance of large page support in ESX Server 3.5, using SPECjbb2005. We conducted our tests in Linux, Windows, and Solaris guest operating systems.

## Server Configuration

- Base hardware: HP DL585
- Processors: 4 dual-core 2.6GHz revE Opteron 885 processors
- DTLB on one core: 32 L1 small TLB entries, 512 L2 small TLB entries and 8 L1 large TLB entries
- Memory: 32GB

## Virtual Machine Configuration

- Virtual hardware: 4 processors and 16GB memory
- Guest operating systems:
  - 64-bit Red Hat Enterprise Linux 4 Advanced Server
  - 64-bit Windows Server 2003
  - 64-bit Solaris 10 U3
- JVM for SPECjbb2005: BEA JRockit R26.3.0 - JRE 1.5.0_06

**Figure 1.** SPECjbb2005 performance (Red Hat Enterprise Linux 4 AS)

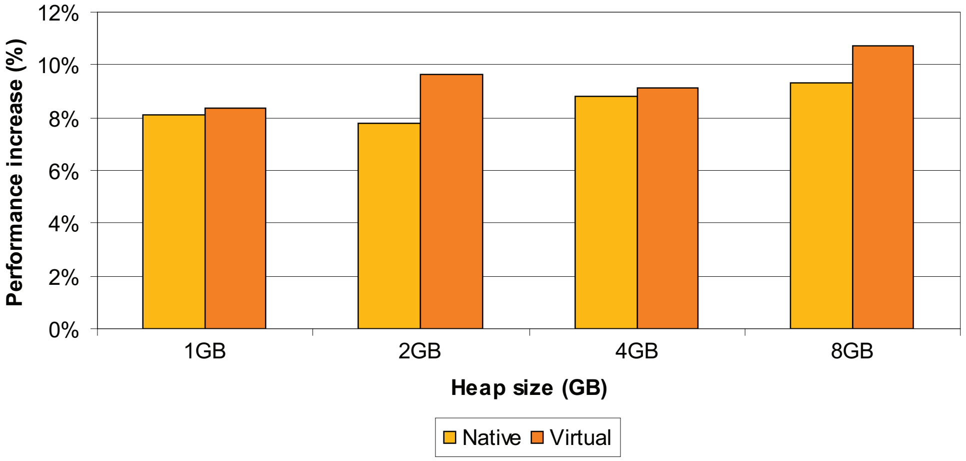
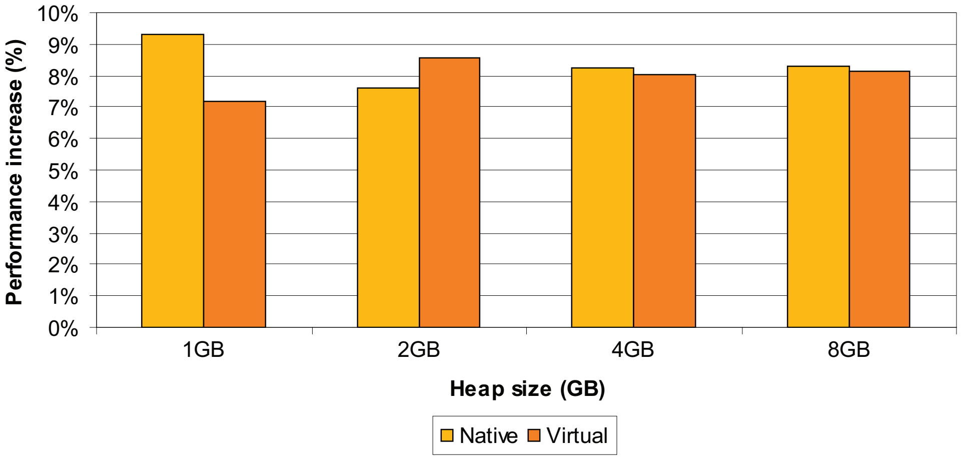


**Figure 2.** SPECjbb2005 performance (Windows Server 2003



Figures 1 and 2 show the SPECjbb2005 results for Red Hat Enterprise Linux 4 Advanced Server and Windows Server 2003, respectively. The Java virtual machine heap size varies from 1GB to 8GB, in order to represent different working set sizes.

On both operating systems, using large pages improves overall SPECjbb2005 performance by 8–10 percent. The performance improvements are almost the same on a virtual machine as on a native machine. Further analysis shows that the performance improvement comes from a significant reduction in L1 DTLB misses.

SPECjbb2005 on a Solaris 9 guest operating system exhibits a similar 8.1 percent performance gain with 4GB JVM heap size. Although the performance gain varies depending on the system setup and workload, these tests indicate the potential performance improvements of using large pages. We expect database workloads to exhibit similar performance improvements using large pages.

# Conclusion

The enhanced large page support in VMware ESX Server 3.5 and ESX Server 3i v3.5 enables 32-bit virtual machines in PAE mode and 64-bit virtual machines to make better use of large pages than they could when running on earlier versions of ESX Server. Our study, using SPECjbb2005, shows that using large pages can significantly improve the performance of this workload, compared to running the workload using small pages. The results demonstrate that if an application can benefit from large pages on a native machine, it can potentially achieve similar performance improvement in a virtual machine running on ESX Server 3.5 and ESX Server 3i v3.5.

# Appendix A: Enabling Large Page Support in Guest Operating Systems

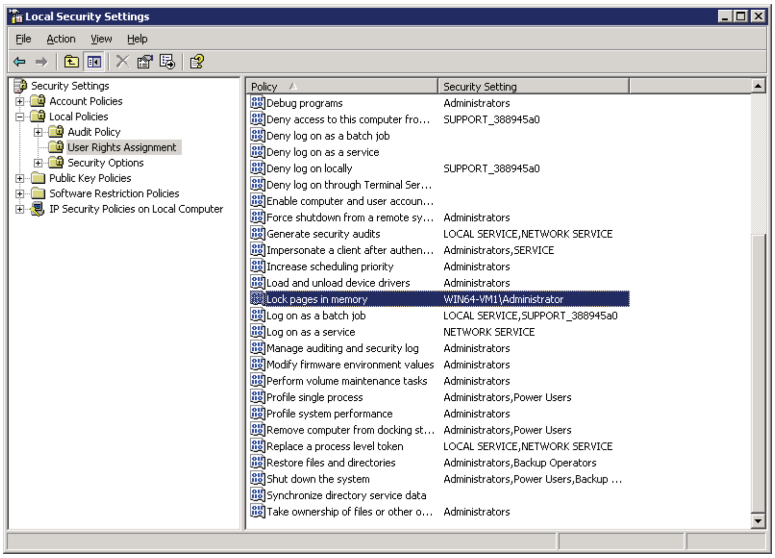
Consult the documentation for your operating system for details on how to configure large page support. This section illustrates how to enable large page support in Windows Server 2003 and Red Hat Enterprise Linux.

## Enabling Large Page Support in Windows Server 2003

To enable large page support in Windows Server 2003, the system administrator must grant appropriate users the privilege to “Lock pages in memory.” This privilege is not enabled by default when Windows is installed. To grant this privilege, take the following steps:

1. Choose **Start** > **Control Panel** > **Administrative Tools** > **Local Security Policy**.
2. In the left pane of the Local Security Settings window, expand **Local Policies** and choose **User Rights Assignment**.
3. In the right pane of the Local Security Settings window, choose **Lock pages in memory** and choose **Action** > **Properties**. The Local Security Setting dialog box opens.
4. In the **Local Security Setting** dialog box, click **Add User or Group**.
5. Enter the appropriate user name, then click **OK** to close the Select Users or Groups dialog box.

6 Click **OK** to close the Local Security Policy Setting dialog box. The Local Security Settings window shows the user assigned to the policy.



| Policy | Security Setting |
|---|---|
| Debug programs | Administrators |
| Deny access to this computer fro... | SUPPORT_388945a0 |
| Deny log on as a batch job | |
| Deny log on as a service | |
| Deny log on locally | SUPPORT_388945a0 |
| Deny log on through Terminal Ser... | |
| Enable computer and user accoun... | |
| Force shutdown from a remote sy... | Administrators |
| Generate security audits | LOCAL SERVICE,NETWORK SERVICE |
| Impersonate a client after authen... | Administrators,SERVICE |
| Increase scheduling priority | Administrators |
| Load and unload device drivers | Administrators |
| Lock pages in memory | WIN64-VM1\Administrator |
| Log on as a batch job | LOCAL SERVICE,SUPPORT_388945a0 |
| Log on as a service | NETWORK SERVICE |
| Manage auditing and security log | Administrators |
| Modify firmware environment values | Administrators |
| Perform volume maintenance tasks | Administrators |
| Profile single process | Administrators,Power Users |
| Profile system performance | Administrators |
| Remove computer from docking st... | Administrators,Power Users |
| Replace a process level token | LOCAL SERVICE,NETWORK SERVICE |
| Restore files and directories | Administrators,Backup Operators |
| Shut down the system | Administrators,Power Users,Backup ... |
| Synchronize directory service data | |
| Take ownership of files or other o... | Administrators |

You need to log out, then log back in or reboot to free memory. Now, applications running as this user can access large pages.

## Enabling Large Page Support in Red Hat Enterprise Linux

To enable large page support in Red Hat Enterprise Linux, the root user must preallocate the large pages from the command line.

1 If you need 2GB large pages, enter the following command to preallocate 1024 2MB large pages:

```
echo 1024 > /proc/sys/vm/nr_hugepages
```

2 The system may not allocate all the pages you specified. To see how many large pages were actually allocated, enter the following command:

```
cat /proc/sys/vm/nr_hugepages
```

3 Some applications, such as the BEA Java Virtual Machine, use `/mnt/hugepages` to allocate large pages. If you are using such an application, you must create the mount point for mapping the large pages. To do so, enter the following commands:

```
mkdir /mnt/hugepages
mount –t hugetlbfs nodev /mnt/hugepages
chmod 777 /mnt/hugepage
```

# Appendix B: Enabling Large Page Support in ESX Server

ESX Server 3.5 and ESX Server 3i v3.5 enable large page support by default. When a virtual machine requests a large page, the ESX Server kernel tries to find a free machine large page. Both the virtual machine monitor and the guest operating system can request large pages. The virtual machine monitor requests large pages to improve its own performance. The guest operating system or an application running on the guest operating system requests large pages just as it would when running on a native machine. The ESX Server kernel supplies large pages to the virtual machine opportunistically. When there is no free large page available, ESX Server emulates a guest operating system large page using small machine pages.

In ESX Server 3.5 and ESX Server 3i v3.5, large pages cannot be shared as copy-on-write pages. This means, the ESX Server page sharing technique might share less memory when large pages are used instead of small pages. In order to recover from nonsharable large pages, ESX Server uses a "share-before-swap" technique. When free machine memory is low and before swapping happens, the ESX Server kernel attempts to share identical small pages even if they are parts of large pages. As a result, the candidate large pages on the host machine are broken into small pages. In rare cases, you might experience performance issues with large pages. If this happens, you can disable large page support for the entire ESX Server host or for the individual virtual machine.

To disable the large page support for the entire ESX Server host, take the following steps using the VMware Infrastructure Client:

1 In the left pane of the VI Client, choose the ESX Server host.

2 In the right pane of the VI Client, click the **Configuration** tab.

3 Choose **Software** > **Advanced Settings**. The Advanced Settings dialog box opens.

4 In the left pane of the Advanced Settings dialog box, choose **Mem**.

5 In the right pane of the Advanced Settings dialog box, set **Mem.AllocGuestLargePage** to `0`.

6 Click **OK** to close the Advanced Settings dialog box.

To disable guest operating system large page support for a virtual machine, add the following line in the virtual machine's configuration file:

```
monitor_control.disable_mmu_largepages = TRUE
```

## Appendix C: Enabling Large Page Support in Applications

Consult the documentation for your application for details on how to configure large page support. For example, the *Oracle Database Platform Guide* has details on how to enable large page support for an Oracle database

**VMware, Inc. 3401 Hillview Ave., Palo Alto, CA 94304 www.vmware.com**

Revision 20080205 Item: PS-050-PRD-01-01