[September, 2010]

Top 10 64-bit IBM® WebSphere® Application Server FAQ

Document version [1.0]

WebSphere software

Christopher J. Blythe
WebSphere Application Server Performance

Introduction

As more organizations move to 64-bit IBM® WebSphere® Application Server, a few common questions always come up in the discussion. The purpose of this article is to quickly address these common questions in an FAQ style format that is easy to consume. For more detailed explanations, we may reference material documented in our previous 64-bit related whitepapers located at the following URLs.

- IBM WebSphere Application Server and 64-bit platforms 64-bit Performance – version 2.2
  ftp://public.dhe.ibm.com/software/webserver/appserv/was/64bitPerf.pdf
- IBM WebSphere Application Server V7 64-bit Performance and Scalability
  ftp://public.dhe.ibm.com/software/webserver/appserv/was/WAS_V7_64-bit_performance.pdf
- WebSphere for z/OS V6.1 – 64-bit Addressing Support
  http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP100920
- Match 32-bit WebSphere Application Server performance with new features in 64-bit Java™ on System z
  http://www.ibm.com/partnerworld/wps/servlet/ContentHandler/whitepaper/systemz/java_websphere/performance

The first of these papers is based on WebSphere Application Server V6.1 and discusses in detail the performance implications (both good and bad) associated with 64-bit hardware platforms and Java. The second paper, on the other hand, focuses on the improved performance and Java heap usage characteristics associated with the Compressed Reference technology available within IBM JDK 6 and WebSphere Application Server V7. We strongly advise that everyone read these papers before making the jump to 64-bit application server installations and environments.

This article primarily focuses on 64-bit WebSphere Application Server in distributed environments like AIX®, Windows®, and Linux® on Power™, AMD®/Intel® x86-64, and System z® platforms. On the System z platform, however, there are subtle differences which will be highlighted under the answer to each question by the "*System z Specific"* delimiter.

Frequently Asked Questions

1) ***Can I continue to run 32-bit WebSphere Application Server instances on a 64-bit hardware platform and operating system?***

   Yes. However, you should check the list of WebSphere Application Server supported platforms to verify that the environment is supported.
   http://www-01.ibm.com/software/webservers/appserv/was/requirements/

   System z Specific: On z/OS® and Linux on System z, WebSphere Application Server supports both 31-bit and 64-bit modes. In WebSphere Application Server for z/OS v6.1, 31-bit mode was the default. However, starting with v7, 31-bit mode has been deprecated and the default is now 64-bit mode.

2) ***Can 32-bit and 64-bit installations of WebSphere Application Server co-exist on the same system?***

   Yes, but keep in mind that the 32-bit and 64-bit versions of WebSphere Application Server are completely separate installations. You cannot simply install the 32-bit version of the application server and replace the JDK for a 64-bit version. There are native C libraries within the application server that are specific to 32-bit and 64-bit platforms.

   System z Specific: Unlike distributed platforms, there is a single installation image that supports both 31-bit and 64-bit modes on z/OS. The choice between 31-bit and 64-bit modes can be modified either through the Administration Console or through wsadmin scripting.

3) ***Can 32-bit and 64-bit WebSphere Application server instances co-exist within the same Cell?***

   Yes. 32-bit and 64-bit instances can be federated to and managed within the same Network Deployment cell very

similar to how multiple WebSphere Application Server versions (i.e. v6.1.X and V7.X) can co-exist in the same cell.

This is outlined in the following IBM support document: http://www-01.ibm.com/support/docview.wss?uid=swg27007163&loc=en_US

4) ***Do I need to re-compile my application code for deployment on 64-bit instances?***

This is one of the most asked questions that we field and ties back to the answer from question number two. Java classes are portable and do not have to be re-compiled when moving from a 32-bit JDK to a 64-bit JDK. However, native code is platform dependent and must be re-compiled. Therefore, if your application uses any native libraries that are access through Java Native Interface (JNI), etc., these must be re-compiled for 64-bit platforms.

5) ***Can I develop my application on a 32-bit environment and deploy to a 64-bit environment?***

This is basically the same question as number four and again the answer is the same. Java code is portable and does not require re-compilation. However, native code and libraries are platform dependent and must be re-compiled.

6) ***How does 64-bit impact memory usage?***

This is probably one of the most important topics to understand when it comes to 64-bit. There are two aspects to consider here, objects in the Java heap and native memory.

Java Heap – Prior to the Compressed Reference technology (see question #9)in WebSphere Application Server v7 and IBM JDK 6, all memory references to Java objects double in size when moving from 32-bit to 64-bit Java. So, what exactly does this mean? To sum it up quickly, since the memory references

are larger, the Java heap will fill up faster requiring more frequent garbage collections. Furthermore, larger memory references also lead to a higher percentage of processor cache misses. These two factors have obvious performance implications.

The naive approach to solving this “problem” is to simply increase your heap size. However, increasing the heap size requires more native memory on your system, generally leads to increased garbage collection pause times since more memory must be scanned during a GC cycle, and can also lead even more processor cache misses. For further details, please consult the first whitepaper referenced at the beginning of this article.

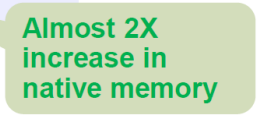
Native Memory – Another direct implication of 64-bit Java is an increase in native memory usage for the running Java Virtual Machine (JVM) instance. This increase in native memory usage is common for both WebSphere Application Server v6.1 and v7 (with Compressed References enabled). This was alluded to in the first whitepaper, but needs further clarification because this is another very important point to understand from a migration perspective.

The Java heap is only one component of the total native memory used by a running JVM instance. Additional memory is required by the JVM to store internal JVM components like the Just-In-Time (JIT) compiler, garbage collector, core virtual machine, Java classes, etc. Furthermore, any memory allocations performed by JNI code are also handled within this memory space. So, if you specify a 1 GB heap, the actually native memory used by the JVM will be (and could be considerably) higher than 1 GB. Much like Java object references in the heap, memory references in the native components of the JVM also double in size when moving from 32-bit to 64-bit. This is an extremely import point to remember when sizing your hardware.

To demonstrate this point, we configured a 32-bit instance and a 64-bit instance of WebSphere Application Server v7 each with a 1 GB heap running the Apache Geronimo DayTrader

Benchmark Sample application and measured the native memory usage of each. The results are provided below:

	32-bit JVM	64-bit JVM
Total Process Size	~1,390 MB	~1,765 MB
Native Memory Usage	~366 MB	~741 MB



This will vary by application and platform, but clearly demonstrates the increase in memory used by the native components of the JVM.

7) ***What are the performance implications of 64-bit Java?***

This too is covered in detail in the first whitepaper referenced, but in general, you should expect a slight degradation in performance when moving from a 32-bit version of WebSphere Application Server to a 64-bit version. This degradation can be directly attributed to the increased size of memory references within the Java and native heap and the subsequent pressure this places on the garbage collector and processor cache hierarchy. Just to provide an example, with WebSphere Application Server V6.1 and IBM Java 5 JDK, we measured approximately a 15% reduction in peak throughput when moving from 32-bit to 64-bit and maintaining the same heap size in an Intel/Linux platform. This degradation will vary based on the underlying platform and, more specifically, the cache architecture associated with the processor. Fortunately, Compressed Reference technology (see question #9) in WebSphere Application Server v7 and IBM JDK 6 reduces this degradation, but does not fully reach the performance levels of a 32-bit instance. However, there are some scenarios where 64-bit actually provides a performance benefit.

8) ***What types of applications will benefit from 64-bit?***

The primary purpose of 64-bit Java is to provide the ability to create Java heaps larger than a 32-bit address space will allow (typically around 2 GB depending on the platform and configuration). Therefore, the obvious answer is applications that need a heap larger than 2 GB. This typically comes into play for applications that either frequently create large objects (multiple MB in size) and/or applications that maintain large caches of objects. However, the complete answer goes a bit deeper than this. 64-bit also benefits applications that perform an extensive amount of complex computational processing typically associated with double-precision mathematical calculations and security algorithms. The degree of benefit is highly dependent upon the underlying hardware platform. For instance, Intel and AMD x86-64 platforms provide 8 additional processor registers that are available when running in 64-bit mode. Again for additional details, please consult the first whitepaper referenced in this article.

9) ***What is the Compressed Reference technology?***

Compressed Reference technology is new to WebSphere Application Server V7 and the IBM Java 6 JDK. This technology allows 64-bit Java memory references to be compressed and treated like 32-bit memory references, thus taking up less space in the Java heap and reducing the performance degradation associated with 64-bit Java. For instance, with a 1 GB heap running our DayTrader application, the peak throughput degradation we reported for v6.1 was reduced from approximately 15% to 5%. Compressed References are enabled by default in WebSphere Application Server v7 64-bit up to a heap size of 25GB, but can be manually disabled via the Generic JVM arguments. Also, as you continue to increase the heap size, the difference in performance between the 64-bit JVM with and without Compressed References will decrease in a stair step fashion additional bit shifting is performed to handle the memory address translation.

One final note regarding Compresses References is that this only impacts objects within the Java heap. The native memory used by the JVM for JVM internals is not affected and will continue to be approximately double the size used by a 32-bit

JVM.

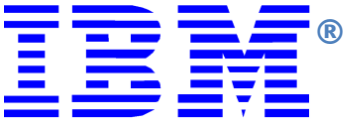
Compressed references are covered in detail in the second 64-bit whitepaper referenced at the beginning of this article.

10) ***Should I move to 64-bit WebSphere Application Server?***

So, last but not least, the ultimate question… should I move to 64-bit? The general rule-of-thumb is…

If you have not noted a measureable improvement in performance associated with the mathematical processing capabilities associated with some 64-bit platforms with your application, do not require the ability to support/utilize large heaps (above ~2 GB) for caching, or do not specifically need to consolidate to a full 64-bit platform stack, stick with 31-bit or 32-bit WebSphere Applciation Server instances.

System z Specific: As previously mentioned, 31-bit mode on z/OS was deprecated in WebSphere Application Server v7. Consequently, you should consider plans for converting to 64-bit on z/OS as support for 31-bit mode may not exist in future releases of the application server.



IBM, the IBM (logo), AIX, POWER, z/OS and WebSphere are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

Solaris, Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Intel is a trademark of Intel Corporation in the U.S. and/or other countries.

Microsoft, Windows, and the Windows logo are trademarks, or registered trademarks of Microsoft Corporation in the United States and/or other countries.

Other company, product, or service names may be trademarks or service marks of others.

References in this publication to IBM products or services do not imply that IBM intends to make them available in all countries in which IBM operates. The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATIONPROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTYOF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUTNOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR APARTICULAR PURPOSE.

Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

The information in this publication is provided AS IS without warranty. Such information was obtained from publicly available sources, is current as of

January 2009, and is subject to change. Any performance data included in the paper was obtained in the specific operating environment and is provided as an illustration. Performance in other operating environments may vary. More specific information about the capabilities of products described should be obtained from the suppliers of those products.