# Oracle® JRockit JVM

Release Notes

R27.6

June 2008

ORACLE

Oracle JRockit JVM Release Notes, R27.6

# Contents

## General Release Information



## R27 Release Information

# R26 Release Information

CHAPTER 1

# General Release Information

This document contains important release information for Oracle JRockit JDK. It contains information on the following subjects:

- Version Naming
- Java Support
- Platform Support
- Installation
- Documentation Accompanying the Oracle JRockit JDK

## Version Naming

The Oracle JRockit JDK version number consists of two parts: the Java SE version and the Oracle JRockit JVM version. For example, Oracle JRockit JDK 6 R27.6 supports Java Standard Edition 6 and contains the JRockit JVM R27.6.

Oracle licenses and bundles the Java class libraries from Sun Microsystems. For the exact version of the Sun class libraries shipped with this version of the JRockit JDK, see Java Support.

## Java Support

JRockit JDK Java certification depends upon the specific version of the product, as outlined here:

- Oracle JRockit JDK R27.6 is certified to be compatible with J2SE 1.4.2_17, J2SE 5.0 Update 15, and Java SE 6 Update 5.
- BEA JRockit JDK R27.5 is certified to be compatible with J2SE 1.4.2_16, J2SE 5.0 Update 14, and Java SE 6 Update 3.
- BEA JRockit JDK R27.4 is certified to be compatible with J2SE 1.4.2_16, J2SE 5.0 Update 14, and Java SE 6 Update 3.
- BEA JRockit JDK R27.4 is certified to be compatible with J2SE 1.4.2_15, J2SE 5.0 Update 12, and Java SE 6 Update 2.
- BEA JRockit JDK R27.3 is certified to be compatible with J2SE 1.4.2_14, J2SE 5.0 Update 11, and Java SE 6 Update 1.
- BEA JRockit JDK R27.2 is certified to be compatible with J2SE 1.4.2_13, J2SE 5.0 Update 10, and Java SE 6.
- BEA JRockit JDK R27.1 is certified to be compatible with J2SE 1.4.2_12 and J2SE 5.0 Update 8.
- BEA JRockit JDK R26.4 is certified to be compatible with J2SE 1.4.2_11 and J2SE 5.0 Update 6.
- BEA JRockit JDK R26.3 is certified to be compatible with J2SE 1.4.2_10 and J2SE 5.0 Update 6.
- BEA JRockit JDK R26.2 is certified to be compatible with J2SE 1.4.2_10.
- BEA JRockit JDK R26.1 is certified to be compatible with J2SE 5.0 Update 4.
- BEA JRockit JDK R26.0 is certified to be compatible with J2SE 5.0 Update 4.

# Platform Support

Oracle JRockit JDK is available for J2SE 1.4.2 and 5.0, and Java SE 6. Platform support varies with the J2SE version.

- Oracle JRockit JDK **6** releases are certified on the platforms listed on the Oracle JRockit JDK 6 Supported Configurations page.
- Oracle JRockit JDK **5.0** releases are certified on the platforms listed on the Oracle JRockit JDK 5.0 Supported Configurations page.

- Oracle JRockit JDK **1.4.2** releases are certified on the platforms listed on the Oracle JRockit JDK1.4.2 Supported Configurations page.

# Installation

Oracle JRockit JDK is included in several products, for example Oracle JRockit Mission Control, Oracle JRockit Real Time and Oracle WebLogic. For more information, see the installation guides for your specific Oracle product.

# Documentation Accompanying the Oracle JRockit JDK

The documentation that is connected to a specific version of the Oracle JRockit JDK is located here:

`http://edocs.bea.com/jrockit/releases/index.html`

CHAPTER 2

# R27 Release Information

This section contains important details for the Oracle JRockit JDK R27. It contains information on the following subjects:

# New Features and Enhancements in the Oracle JRockit JVM R27.6

The Oracle JRockit JVM R27.6 is a maintenance release and contains no new features.

- For information on resolved issues and any changes to existing functionality in this version, please see Changes in the Oracle JRockit JVM R27.6.
- For information on new features and resolved issues in the associated version of Oracle JRockit Mission Control (version 3.0.3), please refer to the *Release Notes* for that product at:

  `http://edocs.bea.com/jrockit/tools/relnotestools/relnotestools3.html`

# New Features and Enhancements in the BEA JRockit JVM R27.5

BEA JRockit R27.5 includes a number of new features and enhancements to existing features. These are described here.



## Support for Updated Java Versions

Java version updates: 1.4.2_16, J2SE 5.0 update 14, Java SE 6 update 3.

## Eclipse Integration of JRockit Mission Control

JRockit Mission Control is now available as an Eclipse plug-in edition. The plug-in version of Mission Control provides seamless integration of BEA JRockit's application profiling and monitoring toolset with the Eclipse development platform. By integrating Mission Control with Eclipse, you will have easy access to the powerful toolset that comprises Mission Control.

When Mission Control is run within the Eclipse IDE, you have access to IDE features that aren't otherwise available in the toolset when it is run as a standalone Rich Client Platform (RCP)

application. The most significant of these features is the ability to see specific code in the running application by opening it directly from Mission Control, a function called Jump-to-Source.

The other benefit of integrating Mission Control with the Eclipse IDE is that it allows you to profile and monitor an application during its development phase just as you would during its production phase. This allows you to spot potential runtime problems before you actually deploy your application to production; for example, you might, while monitoring an application during its development notice a memory leak. By catching the memory leak during development, you can correct it before you migrate your application to a production environment.

For more information, please see Integration with the Eclipse IDE or open Mission Control and launch the help system.

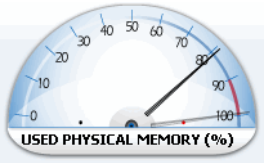
The location of the Eclipse update site will be published at `http://dev2dev.bea.com/jrockit/tools.html` when available.

# Other JRockit Mission Control Updates

The following updates have been made to JRockit Mission Control. These updates apply to both the RCP version and the Eclipse plug-in version.

- The JRockit Runtime Analyzer now shows the number of bytes of objects allocated by each Java thread.
- Three sample files that demonstrate the features of the Latency Analysis Tool have been added. The files are located at `JROCKIT_HOME/missioncontrol/samples/jrarecordings/`. The files are:
  - `pricing_server_logging_on.jra`
  - `pricing_server_logging_off.jra`
  - `java2d_demo.jra` (This file is a recording of the demo located at `JROCKIT_HOME/demo/jfc/Java2D`. The Java2D demo folder contains the source, allowing this recording to demonstrate Jump-to-Source (Jump-to-Source is only available when you are running Mission Control within Eclipse, as described in Eclipse Integration of JRockit Mission Control).
- Small adjacent Latency Analysis Tool (LAT) events of the same type are now clearly marked to make them easier to distinguish.
- Configurable velocimeters (Figure 2-1) have been added to the Console

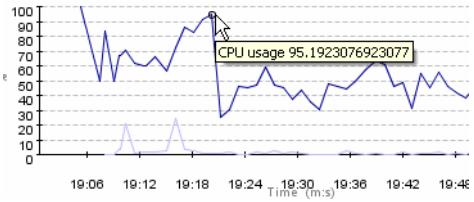
**Figure 2-1 Configurable Velocimeter**




- You can see the exact numerical value for a point in a graph in a tooltip by hovering your mouse pointer at the point.

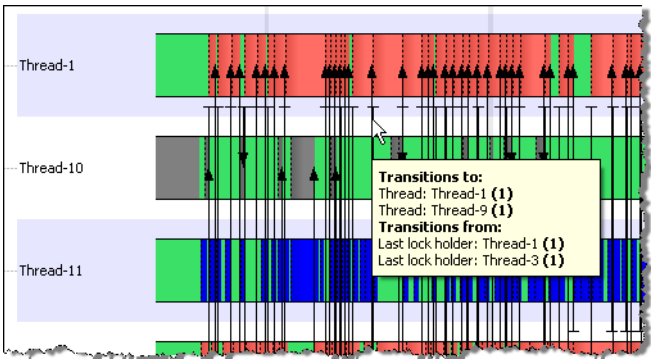
  **Note:** This feature is only available when the graph is frozen.

**Figure 2-2 Displaying the Value for a Point on a Graph**




- The time ranges of graphs shown on the same page can be synchronized.
- You can filter attributes by name in the attribute browser when you select attributes to add to a graph or similar.
- Thread transitions—a latency event in one thread that is associated with another thread—are now displayed as small black arrows on the Latency Graph, as shown in Figure 2-3. By hovering your pointer over a transition arrow, a tooltip will appear, describing the transition.

**Figure 2-3 Arrows Depict Thread Transitions in LAT; Tooltip Describes Selected Transition**




# Updated Command-line Options

This version of BEA JRockit includes updates to some of the command-line options used at startup.

## -Xverbose

A new verbose logging module, `-Xverbose:refobj` has been added. At `info` level this module provides low overhead information on `java.lang.ref.Reference` objects at each garbage collection. At the `debug` level, this module prints out information equivalent to the `info` level printouts from the old `-Xverbose:referents` module.

## -XpauseTarget

The `-XpauseTarget` value can now be set as low as 1 ms. The real minimum pause target still depends on the application size and behavior and the hardware.

## -XlargePages

By default the JVM will continue running without large pages if large pages cannot be acquired when `-XlargePages` is enabled. This option now can use the parameter `exitOnFailure=false` to override this behavior and force the JVM to exit if enough large pages can't be acquired; for example:

```
-XlargePages:exitOnFailure=false
```

# New Features and Enhancements in the BEA JRockit JVM R27.4

BEA JRockit R27.4 is a maintenance release and contains no new features. New features available in the associated version of JRockit Mission Control (JRockit Mission Control 3.0.1) are described in that product's Release Notes at:

`http://edocs.bea.com/jrockit/tools/relnotestools/index.html`

# New Features and Enhancements in the BEA JRockit JVM R27.3

This section describes new features and enhancements released in this version of BEA JRockit It includes information on the following subjects:

- Java Updates
- BEA JRockit Mission Control 3.0
- Localized Documentation
- Performance Improvements

## Java Updates

BEA JRockit R27.3 has been updated to use J2SE 1.4.2_14, J2SE 5.0 Update 11, and Java SE 6 Update 1.

## BEA JRockit Mission Control 3.0

An updated version of BEA JRockit Mission Control is bundled with BEA JRockit R27.3. For a full description on what the release contains, see the BEA JRockit Mission Control Release Notes.

## GUI Localizaton

The BEA JRockit Mission Control GUI is now also available in Japanese and simplified Chinese.

# Localized Documentation

Later in the summer of 2007, documentation also will be available in Japanese and simplified Chinese.

# Performance Improvements

Performance has improved in the following ares:

- Better Out of the Box Performance
- Improved Nursery
- Debugging Performance

## Better Out of the Box Performance

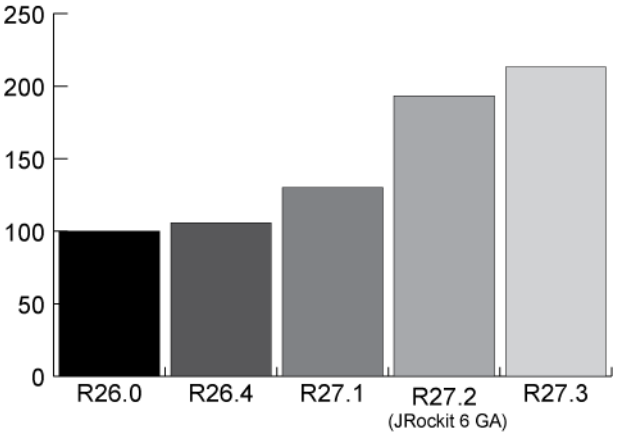
This version of BEA JRockit includes improvements to the following areas, resulting in better out of the box performance

- TLA size
- TLA handling
- Nursery resizing

**Note:** This means that your rarely need to tune `-XXtlaSize`, `-XXlargeObjectLimit`, and `-Xns` (nursery size).

Figure 2-4 demonstrates the release-to-release improvements to out of the box performance. The large increase between R27.1 and R27.2 coincides with the introduction of JRockit for Java SE 6. R27.3 contains further enhancements.

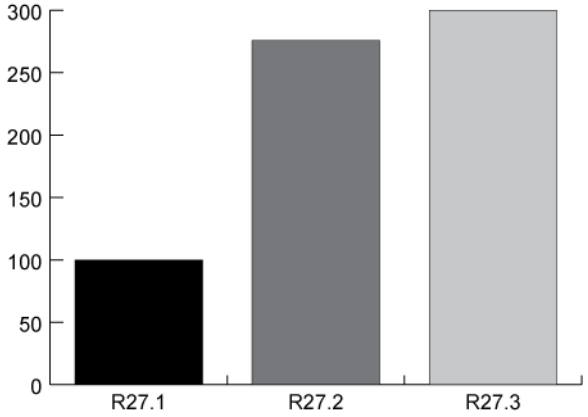
**Figure 2-4 SPECjbb2005 Out of the box improvements from R26.0 to R27.3**




## Improved Nursery

A new nursery implementation was introduced in R27.2, providing significant enhancements to application throughput as well as garbage collection pause times. This implementation has been refined in R27.3, leading to further improvements in performance (see Figure 2-5).

**Figure 2-5 SIP benchmark results—further improvements from R27.2**




## Debugging Performance

Single-stepping in debug mode on single-CPU machines is now significantly faster than in previous JRockit versions.

# New Features and Enhancements in the BEA JRockit JVM R27.2

This section describes new features and enhancements released in this version of BEA JRockit It includes information on the following subjects:



## Java SE 6 Support

BEA JRockit is now available for Java SE 6.

JRockit for Java SE 6 provides all current JRockit capabilities, including:

- Industry leading performance
- Advanced monitoring and diagnostics capabilities
- Full support for JRockit Mission Control 2.0

In addition, the Java SE 6 version of JRockit includes all generic Java SE 6 features, such as:

- XML and Web Services enhancements
  - Java Architecture for XML Binding (JAXB) 2.0
  - Java API for XML-Based Web Services (JAX-WS) 2.0
  - Streaming API for XML (StAX)
  - Web-Services Metadata

  - XML Digital-Signature APIs
- Annotations
  - Common Annotations
  - Pluggable Annotation-Processing API
- JDBC 4.0
- Scripting
- Java Compiler API

For more information about Java SE 6, please see the following web page:

http://java.sun.com/javase/6/docs/index.html

BEA JRockit for Java SE 6 is current available on x86 and 64-bit Xeon/AMD64 platforms.

# Java 1.4.2 and 5.0 Updates

BEA JRockit R27.2 has been updated to use J2SE 1.4.2_13 and J2SE 5.0 Update 10.

# New Platform Support

BEA JRockit is now supported on Windows Vista and Red Hat Enterprise Linux 5.0.

# Attach API Support

BEA JRockit now supports Sun Microsystem's Attach API, a Java extension that provides a way to attach tools written in Java to BEA JRockit JVM. For details, please refer to Attach API Support at:

`http://edocs.bea.com/jrockit/geninfo/diagnos/aboutjrockit.html#wp1083571`

# Improved System.nanoTime Resolution

The `System.nanoTime()` method has been improved to always use the best time resolution available on each platform. For more information about the `System.nanoTime()` method, see `Timing with nanoTime()` and `CurrentTimeMillis()` in the BEA JRockit Diagnostics Guide.

# Performance Improvements

This version of JRockit includes numerous performance enhancements, including an improved nursery implementation, software prefetching, and garbage collection heuristics. These enhancements will improve performance by an average of 10% over a broad range of applications, with the largest benefits expected for memory intensive applications and out-of-the-box configurations.

The following performance improvements can be noticed for this release:

- Improved Nursery Implementation
- Improved Software Prefetching
- Improved Garbage Collection Heuristics
- Examples of Performance Improvements

## Improved Nursery Implementation

The R27.2 release includes a new nursery implementation, which yields better application throughput and shorter nursery garbage collection pause times.

## Improved Software Prefetching

Software prefetching, previously enabled with the options `-XXallocPrefetch` and `-XXallocRedoPrefetch`, is now enabled by default. This can improve performance by up to 40%.

**Note:** To fully benefit from this feature on Intel Xeon servers, it is recommended that you disable hardware prefetching in the computer's BIOS.

## Improved Garbage Collection Heuristics

Nursery sizing heuristics have been improved for the default garbage collection algorithm (`-Xgcprio:throughput`), leading to better application throughput.

The default configuration of the `-XXgcThreads` option has been improved, resulting in better out-of-the-box behavior for latency sensitive applications. In most cases, there is no longer a need to tune this option manually.

## Examples of Performance Improvements

To demonstrate the performance improvements in JRockit R27.2, see the following benchmark results:

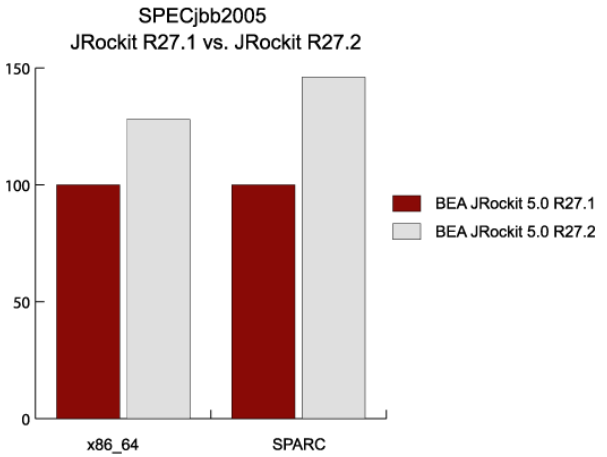
- SPECjbb2005
- WLSS Benchmark

### SPECjbb2005

The results on the SPECjbb2005 benchmark clearly shows the performance improvements that an upgrade from R27.1 to R27.2 can bring.

In Figure 2-6, a benchmark comparison between the BEA JRockit R27.1 and R27.2 releases is shown, where the JVMs have been tuned for optimal performance with various start-up options.

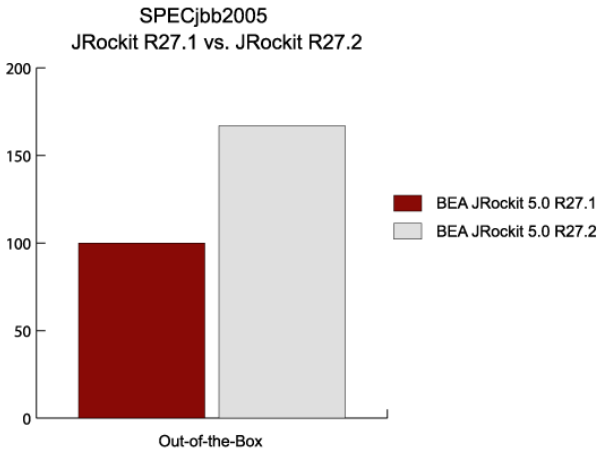
**Figure 2-6 SPECjbb2005 performance improvements using tuned JRockits**




As you can see, R27.2 shows a performance increase of more than 30% compared with the R27.1 release.

The most impressive SPECjbb2005 benchmark result was generated when R27.1 and R27.2 were compared completely out of the box, without any performance tuning. Figure 2-7 demonstrates the improvements in performance that the new and improved out-of-the-box behavior provides. The out of the box performance improvement is almost 70%.

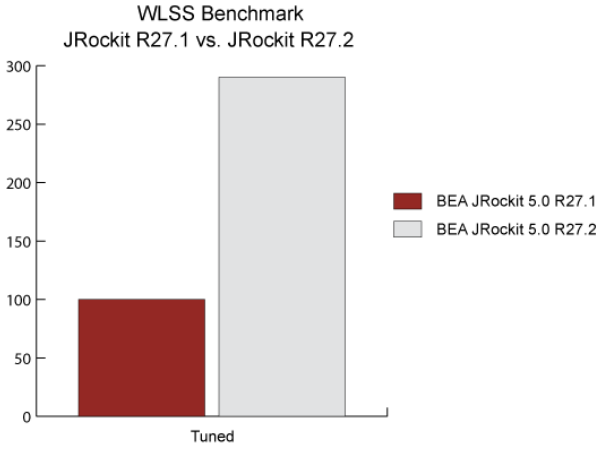
**Figure 2-7 Out of the box (OOTB) performance improvements**




## WLSS Benchmark

This is a benchmark that demonstrates the performance benefits of the new nursery implementations. The application is characterized by a large number of short lived sessions, leading to high memory allocation and short lived objects. Performance is measured in calls set up per second, under the boundary condition that 95% of the call setups should be done within 50 milliseconds. These requirements imply that the application benefits from an efficient nursery. Figure 2-8 visualizes the improvement.

**Figure 2-8 SIP benchmark results**



## Supportability Features

There is a new verbose module for referents. Use the start-up option `-Xverbose:referents` or the jrcmd parameter `verbosity set=referents=info` to make the JRockit JVM print verbose information on reference objects.

### JRA Improvements

In BEA JRockit Runtime Analyzer, you can now see how long JRockit had been running before the start of the JRA recording. In addition, a list of all processes running on the host is included in JRA recordings.

## New Command-line Options

The following command-line options have been added to BEA JRockit R27.2. For a description of any option, please refer to the Oracle JRockit JVM *Reference Manual*.

- `-XlargePages` (was `-XXlargePages`)
- `-XX:MaximumNurseryPercentage`

## Updated Command-line Options

The command-line options listed in Table 2-1 have been updated. For full descriptions on all command-line options, please refer to the BEA JRockit Reference Manual.

**Table 2-1 Updated Command-line Options**

| -X options | -XX options |
|---|---|
| `-Xgc` | `-XXlargeObjectLimit` |
| | `-XXminBlockSize` |
| | `-XXtlaSize` |

# New Features and Enhancements in the BEA JRockit JVM R27.1

This section describes new features and enhancements released in this version of BEA JRockit. It includes information on the following subjects:

# BEA JRockit Mission Control 2.0

A completely new version of BEA JRockit Mission Control is bundled with JRockit R27.1. This version contains a large set of usability enhancements, online documentation, and even more detailed diagnostics data, see the BEA JRockit Mission Control Release Notes.

# Improved Monitoring and Diagnostics

The verbose logging framework in JRockit has been completely reworked. It now provides fine granular control over a large number of JVM subsystems, such as memory and threads, and it allows you to specify the amount of log data from each subsystem, log destination, and a variety of *decorations*, such as configurable time stamps.

Verbose logging can be controlled in several different ways:

- `-Xverbose`, `-XverboseLog`, and `-XverboseDecorations` command-line options.
- jrcmd tool, see Using jrcmd in the JRockit Diagnostics Guide.
- JRockit Management API, see the JMAPI Javadocs.

The Java management (JMX) implementation in BEA JRockit 5.0 R27.1 has been changed to require security to be enabled by default. It also requires you to specify the IP port for the management server explicitly. To revert to the old behavior, use:

`-Xmanagement:ssl=false,authenticate=false,port=7091`. See `-Xmanagement` in the Reference Manual for details. This change does not affect JRockit 1.4.2.

## Improved Supportability

This version of JRockit also includes several supportability enhancements, including improved crash files, JVM self-checks, and other features. While these features are not intended for end-users, they will facilitate communication with BEA Support and speed up problem resolution.

## Connect On-Demand

JRockit 5.0 and later now supports the Attach API, see: http://java.sun.com/javase/6/docs/technotes/guides/attach/index.html. This means you can connect on-demand to:

- JMX, see Using the Management Console.
- JVMTI, which enables you to connect on-demand from any JVMTI client supporting this functionality.

## Improved Documentation

A completely new Diagnostics Guide has been added to the JRockit product documentation set. This guide will help you troubleshoot JRockit and your applications.

## IPv6 Support

IPv6 is now available on all platforms supported by JRockit.

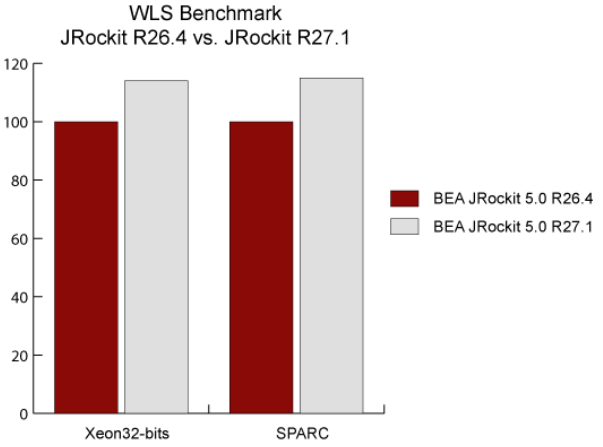
## Expanded Support for Solaris

JRockit is now available for both J2SE 1.4.2 and 5.0 on Solaris/SPARC.

## Performance Improvements

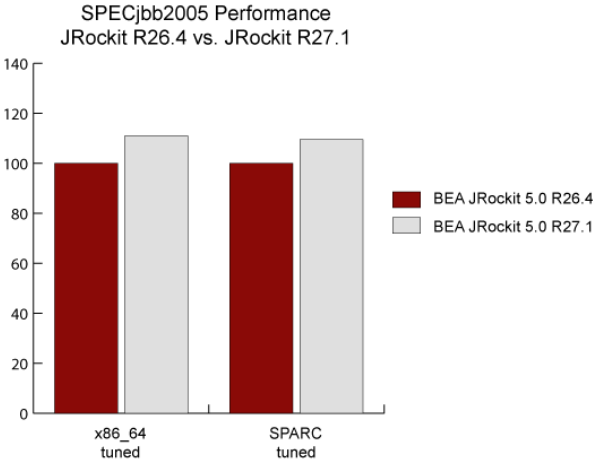
This version of JRockit includes numerous performance enhancements. One example is that performance has been improved for J2EE applications using a large number of JSP pages and servlets. These enhancements are expected to improve performance on many WLS applications by 10-15%, see Figure 2-9.

**Figure 2-9 WLS Benchmark—JRockit R26.4 vs. JRockit R27.1**




The release also includes a number of generic enhancements, as demonstrated by improved scores on the SPECjbb2005 benchmark. See Figure 2-10 for a comparison between the BEA JRockit R26.4 and BEA JRockit R27.1 releases.

**Figure 2-10 SPECjbb2005 performance improvements**



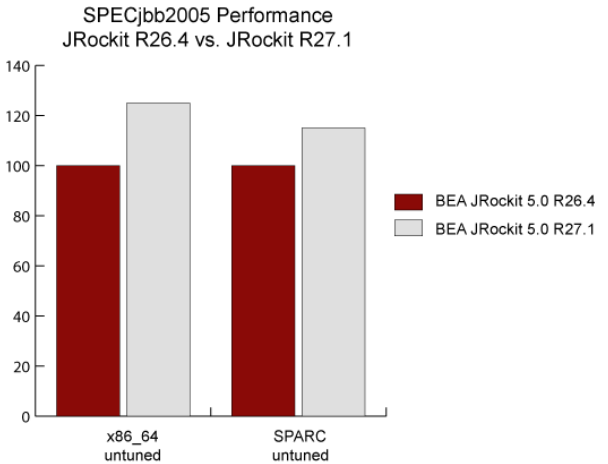

Out of the box performance has also been improved due to:

- Improved default configuration for memory allocation, see `-XXtlaSize` in the Reference Manual.

- Compressed references is now available on all 64-bit platforms (x86-64, Itanium, and SPARC) and is enabled by default.

Plus other enhancements, see Figure 2-11 for a comparison between the BEA JRockit R26.4 and BEA JRockit R27.1 releases.

**Figure 2-11 Out of the box performance**




# New Command-line Options

The following start-up commands have been added with BEA JRockit R27. For a description of any new option, please refer to the specific option in the BEA JRockit Reference Manual (by clicking on the option name in the following list).

- `-XXtsf`
- `-XverboseDecorations`

# Updated Command-line Options

The rules for how command-line parameters are parsed have been updated to avoid user confusion. Incompatible command-line combinations now cause BEA JRockit to print out an error message and terminate. Please refer to the specific option in the BEA JRockit Reference Manual (by clicking on the option name in Table 2-2) for a description of the new behavior.

**Table 2-2 Updated Command-line Options**

| -X options | -XX options |
|---|---|
| -Xgc | -XXallocClearChunks |
| -XgcPrio | -XXallocClearChunkSize |
| -Xmanagement | -XXcompressedRefs |
| -Xms | -XXdisableFatSpin |
| -Xmx | -XXdisableGcHeuristics |
| -Xns | -XXexternalCompactRatio |
| -XpauseTarget | -XXfullCompaction |
| | -XXfullSystemGC |
| | -XXinitialPointerVectorSize |
| | -XXinternalCompactRatio |
| | -XXlargeObjectLimit |
| | -XXmaxPooledPointerVectorSize |
| | -XXminBlockSize |
| | -XXnoCompaction |
| | -XXnoSystemGC |
| | -XXpointerMatrixLinearSeekDistance |
| | -XXsetGC |
| | -XXallocPrefetch |
| | -XXallocRedoPrefetch |
| | -XXcompactSetLimit |
| | -XXcompactSetLimitPerObject |
| | -XXcompactRatio |

**Table 2-2 Updated Command-line Options**

| -X options | -XX options |
|---|---|
| | -XXstaticCompaction |
| | -XXthroughputCompaction |
| | -XXtlaSize |
| | -XXusePointerMatrix |

# Changes in the Oracle JRockit JVM R27.6

Table 2-3 lists changes in this version of the Oracle JRockit JVM.

**Table 2-3 Changes in the Oracle JRockit JDK R27.6**

| Issue ID | Description |
|---|---|
| CR371396 | When a full compaction was issued, all references in the heap wanted to be stored in the compact set. If a limit was set on the compact set, it would likely skip the compaction due to too many pointers. This has been fixed. |
| CR366936 | In previous releases of the JRockit JVM, unloading class hierarchies that had many unloaded types sharing a common superclass or implementing the same interface could cause very long pause times. This has been improved in R27.6 . |
| CR366265 | Technical license checks have been removed. |
| CR366238 | The JRockit JVM running with WebLogic Event Server on Sparc systems was crashing in `cgStoreMetaInfo`, indicating a known issue in delay slot scheduling. The delay slot scheduling bug has been fixed and the system is no longer crashing. |
| CR364912 | In previous versions of the JRockit JVM, a thread suspended in a debugger in a call chain for a static initializes could cause a crash while reading local variable information on the receiving type for the method initializing the class. This has been fixed. |
| CR361911 | Calls to `java.lang.management.CompilationMXBean.getTotalCompilationTime()` previously returned 0. This has been fixed. |

**Table 2-3 Changes in the Oracle JRockit JDK R27.6**

| Issue ID | Description |
|---|---|
| CR361457 | Connecting the Memory Leak Detector to a running JRockit JVM built for Linux IA32 could cause the JVM to crash if the JVM process used more than about 1020 file descriptors at that time. This has now been fixed. |
| CR360910 | The JRockit JVM did not properly handle error codes returned from `Agent_OnAttach`. If an error was returned, the JVM was aborted. This has been fixed. |
| CR359309 | The ACLs (Access Control Lists) on the per-user `<TEMP>\hsperfdata_<user>` directory are set up such that users with administrator privileges have the rights to delete the directory. This only affects Windows versions of the JRockit JVM. For more information, please see: http://bugs.sun.com/bugdatabase/view_bug.do?bug_id=5073453. |
| CR358260 | When a large number of classes (~10,000) implementing the same interface were unloaded at the same time, unregistering them from the Interface could take a long time. This has been fixed |
| CR357526 | The maximum number of active Object monitors has been increased to 4,194,304. |
| CR357397 | The nursery size was slowly shrinking until it was almost zero. Eventually, the young collection stops reclaiming any space at all, although no old collection is triggered. Thus, the JRockit JVM just continues to do young collections. This has been fixed. |
| CR355985 | After upgrading from an earlier version of the JRockit JVM to R27.4, some users experienced a change in class loader behavior: a `NoClassDefFoundError` was thrown if the `$Inner` class was not present in the `CLASSPATH`. This has been fixed. |
| CR355465 | An operating system bug in some operating systems caused signals to get lost if they were sent at the exact time of a call to `pthread_create`. This could cause the JRockit JVM to hang. This has been solved by blocking those signals when calling `pthread_create`. |
| CR355117 | On previous versions of the JRockit JVM, setting back the system clock would cause calls to `Thread.sleep` to sleep longer than intended. This has now been fixed. |

**Table 2-3 Changes in the Oracle JRockit JDK R27.6**

| Issue ID | Description |
|---|---|
| CR330541 | The new `fontconfig.properties` file for CJK (Chinese, Japanese, Korean) locales compatible with RHEL 5 and Asianux 3.0 has been fixed by backporting the fix from Oracle JRockit JDK 6 to Oracle JRockit JDK 5.0. This is a backport of a fix for the problem reported by Sun in http://bugs.sun.com/bugdatabase/view_bug.do?bug_id=2149631. |
| CR328979 | In versions R27.1 to R27.5 of JRockit JVM, delay slot scheduling issues on SPARC resulted in crashes during garbage collection. This has been fixed. |
| CR236722 | The JRockit JVM is more robust when handling JNI threads which have terminated without calling `DetachCurrentThread`. The JVM will now try to detect these threads and ignore them. The JRockit JVM will print a warning message when this happens to alert developers to the problem. However, note that by not calling `DetachCurrentThread`, the JNI API is violated, and even though The JRockit JVM is now more robust, it can never guarantee the stability when the API is violated in this way. |

# Changes in the BEA JRockit JVM R27.5

Table 2-4 lists changes in this version of the BEA JRockit JVM.

**Table 2-4 Changes in the BEA JRockit JVM R27.5**

| Issue ID | Description |
|---|---|
| CR359828 | In earlier versions of BEA JRockit, the value for **Heap Usage Before** for a garbage collection in a JRA recording was incorrect, as it actually showed the **Heap Usage After** for the proceeding collection instead. This is now fixed. |
| CR358662 | BEA JRockit would occasionally crash during optimization of methods containing two or more different chains of String object concatenations. This has been fixed. |
| CR356984 | A bug was causing code optimization to crash in `function _irTypesIsExactClass`. This has been fixed. |
| CR354539 | BEA JRockit was occasionally leaving sockets in a `CLOSE_WAIT` state on Windows when using `java.nio` channel selectors. This has been fixed. |

**Table 2-4 Changes in the BEA JRockit JVM R27.5**

| Issue ID | Description |
|---|---|
| CR354463 | Two threads calling `getThreadInfo` or `getStackTrace` on each other was causing BEA JRockit to deadlock. This has now been fixed. |
| CR352232 | The command line option `-XlargePages` now can use the parameter `exitonfailure=false` to override its default behavior and force the JVM to exit if enough large pages can't be acquired; for example:<br>`-XlargePages:exitOnFailure=true` |
| CR350569 | On some occasions, calling inflate on a closed Inflater would make BEA JRockit crash, creating a core file. Now, JRockit will instead throw a NullPointerException. |
| CR350009 | The verbose module `referents` (with the alias verboserefs) has been replaced by the new `refobj` module. Using `-Xverbose:refobj` will provide simpler but, performance-wise, cheaper statistics about reference objects. An output similar to the detailed output from the old referents module is provided by `-Xverbose:refobj=debug`. Please note that this output, like the old referents module, is performance-wise very costly.<br>If you use the old referents module, it will be converted into `refobj=debug`. |
| CR349794 | The crash dump summary now includes more detailed information on the memory system. |
| CR348007 | A mistake in the heuristics for heap expansion that caused BEA JRockit to throw an Out of Memory error when it should have expanded the heap has been fixed. |
| CR347294 | A direct link has been added from crash files to the BEA JRockit documentation. Now, if BEA JRockit crashes, you can click this link to open the troubleshooting documents. |
| CR346988 | A bug in the stack trace printing code caused BEA JRockit to crash in some circumstances when printing stack traces. This happened regularly when starting WebLogic Server with `-Xverbose:exceptions=debug`. This has been fixed. |
| CR345875 | Calling RetransformClasses via JVMTI would sometimes fail with error code `JVMTI_ERROR_INVALID_CLASS_FORMAT`. This condition has been fixed. |
| CR345588 | The JMAPI call `com.bea.jvm.GarbageCollector.hasCompaction()` returned incorrect values. This has been fixed. |

**Table 2-4 Changes in the BEA JRockit JVM R27.5**

| Issue ID | Description |
|---|---|
| CR345574 | If BEA JRockit was started with `-Xcheck:jni` and a thread reference passed into a JVMTI callback function was used in a JNI call, the JVM would exit with an error saying `[ERROR][native ] Invalid reference`. This has been fixed. |
| CR344773 | In some reports generated by the `-Xgcreport` flag, garbage collection pauses would sometimes appear to be longer than the garbage collection itself. This has been resolved. |
| CR342358 | The minimum pause target limit has been lowered to less that 5 ms. The actual minimum target depends upon the application you are running. |
| CR340660 | Calling the methods `isLoaded()`, `load()` or `force()` in `java.nio.MappedByteBuffer` on an empty buffer would throw an IOException. This has been fixed. |
| CR340016 | In R27.5 the generation of exception stacktraces has been changed to always include the full stacktrace, regardless of whether the exceptions were generated asynchronously (NullPointerException, StackOverflowError, etc) or lazy stacktraces is on.<br>As a consequence of this, the verbose module stackoverflow has been deprecated and is now a no-op. This module will be fully removed in the next major release. |
| CR339531 | For some customers using BEA JRockit R27.3.1, `AssertionError: unused` was thrown. This happened because the ClassLoader was fed class/package names in the wrong format. Instead of receiving `com.bea.MyClass` it received `com/bea/MyClass`. This has been fixed. |
| CR339281 | BEA JRockit 5.0 Update 14 R27.5.0 adds support for Sun PKCS#11 provider (`http://java.sun.com/j2se/1.5.0/docs/guide/security/p11guide.html`) for Linux on Itanium in addition to Linux on 64-bit Xeon/AMD64 (Sun Bug ID 6467921). |
| CR338872 | In R27.1, the class bytes preprocessing facility was changed to allow for recursive preprocessing. This meant that a classpreprocessor instance that was currently doing class preprocessing and through this caused a new class to be loaded would be recursively called with the new classbytes. This caused failures in some existing preprocessor implementations that relied on the old pre-27.1 behavior. In R27.5, this has been reverted. A thread doing class preprocessing will now silently refuse to preprocess any types created by executing the preprocessor itself. |

**Table 2-4 Changes in the BEA JRockit JVM R27.5**

| Issue ID | Description |
|---|---|
| CR335834 | Because of the much larger amount of suspensions that lazy unlocking introduces, BEA JRockit users running on Windows IA64 were often bumping into the GetThreadContext bug. Lazy unlocking is now enabled by default in the Java 6 version of BEA JRockit R27.5 on all platforms except IA64 and for all garbage collection strategies except the deterministic garbage collector. In older releases you can enable lazy unlocking with the command line option `-XXlazyUnlocking`. |
| CR335688 | The garbage collection strategy is now correctly reported when a static concurrent garbage collector uses a parallel mark or sweep phase as an emergency action when the heap has become full before or during garbage collection.<br>`-XXdisableGcHeuristics` now disables all strategy changes, including emergency parallel mark or sweep in the static concurrent garbage collectors. |
| CR333688 | In R27, a bug was introduced that would cause memory leaks whenever a JVMTI agent with the `can_retransform_classes` capability replaced byte code of a class being loaded. This would also impact byte code preprocessing done through the JRockit Management API (JMapi). This bug has been fixed in R27.5. |
| CR329800 | JRockit Mission Control 3.0 did not properly detect license errors from JRockit R27.3.0 when starting a JRA/LAT recording or opening Memleak. This has been fixed. |
| CR328964 | `-XXcompactSetLimit` is now always respected. Note however, that this only applies to references outside the compaction area. The number of references inside the compaction area is not limited by this flag. |
| CR325082 | A rare occurrence in the register allocation code was causing BEA JRockit to crash. This has been fixed. |
| CR306735 | BEA JRockit will no longer accept memory sizes that are larger than the address space on the current platform. In practice, this means that on a 32-bit system, the value given to `-Xmx`, `-Xms` and `-Xns` cannot be larger than 4 GB, or BEA JRockit won't start. |

# R27.3.1 Umbrella Patch for WLS 10.0 MP1 Now Available

An umbrella patch of BEA JRockit JVM R27.3.1 has been built for distribution with WLS 10.0 MP1. You can download this patch as a `zip` (Windows) or `tar.gz` (Linux) file from commerce.bea.com.

The patch include fixes for the CRs listed in Table 2-5:

**Table 2-5 Changes in BEA JRockit JVM R27.3.1 CP**

| Issue ID | Description |
|---|---|
| CR344232 | On some rare occasions, BEA JRockit would crash when allocating memory. This only happened when a call to `mmAllocObjectOrArray` tried to allocate `largeObjectlimit` bytes that were exactly the same size as the TLA fetched. |
| CR339531 | Disabling assertions did not work for ClassLoader-managed assertions. This could result in Assertion Errors when starting WLS in debug mode. The reason for this was that the ClassLoader was fed wrong class/package names. This has been fixed.<br><br>**Note:** This issue has been resolved in BEA JRockit R27.3.1 CP **but not in** BEA JRockit **R27.4**. It will be fixed in BEA JRockit R27.5, scheduled for release in early 2008. |
| CR336511 | A patch has been built and backported for all publicly known security issues in BEA JRockit R27.3.1. This fix corresponds to the security issues in BEA Security Advisory BEA07-177.00 and BEA07-178.00. |
| CR331724 | When running AquaLogic Service Bus SFTP tests, BEA JRockit was creating regular core dumps. This issue occurred when two mutually exclusive code paths doing an arraycopy to the same to-array were both subject to an erroneous optimization. |
| CR328154 | In some Solaris environments, BEA JRockit was unable to detect the number of sockets, cores and hardware threads. This caused the JVM to abort during start up and display error message:<br>`[ERROR] Fatal error in JRockit during memory setup phase.`<br>This situation would occur in a local zone associated with a subset of all available processors. This issue has been resolved. |
| CR326728 | A customer experienced a system crash in `mmListAddLast`. This has been fixed. |

# Changes in the BEA JRockit JVM R27.4

Table 2-6 lists changes in this version of the BEA JRockit JVM.

**Table 2-6 Changes in the BEA JRockit JVM R27.4**

| Issue ID | Description |
|---|---|
| CR345588 | The JMAPI call `com.bea.jvm.GarbageCollector.hasCompaction()` was returning incorrect values. This has been fixed. |
| CR345574 | If the JVM was started with `-Xcheck:jni` and a thread reference passed into a JVMTI callback function was used in a JNI call, the JVM would exit with the error `[ERROR][native ] Invalid reference`. This has been fixed. |
| CR336996 | In earlier versions of BEA JRockit, calling the JVMTI function `IterateOverHeap` with `JVMTI_HEAP_OBJECT_TAGGED` and then `JVMTI_HEAP_OBJECT_UNTAGGED` would sometimes report more objects than doing `IterateOverHeap` with `JVMTI_HEAP_OBJECT_EITHER`. This has been fixed. |
| CR336790 | Modifying a next pointer through reflection was causing a memory leak when processing phantom reference objects. This has been fixed in this version of BEA JRockit. |
| CR336285 | Sometimes the OS would fail to suspend a thread, which lead to BEA JRockit crashing and throwing an `EXCEPTION_ACCESS_VIOLATION` error. This is now fixed. |
| CR334151 | In earlier versions of BEA JRockit, the JVM could hang while doing a Latency Analysis Recording and recording a park event. This has been fixed. |
| CR332182 | In earlier versions of BEA JRockit, the `java.lang.management.ThreadInfo` API and the JVMTI functions `GetOwnedMonitorStackDepthInfo` and `GetOwnedMonitorInfo` did not return monitors that had been entered by calling the JNI function `MonitorEnter`. This has been fixed. |
| CR332016 | In some cases when allocation of a large array failed, the JVMTI ResourceExhausted event was not sent. This has been fixed. |
| CR332012 | In earlier versions of BEA JRockit, the `jvmtiHeapReferenceCallback` callback function was sometimes called with the wrong `class_tag` parameter. This has been fixed. |

**Table 2-6 Changes in the BEA JRockit JVM R27.4**

| Issue ID | Description |
|---|---|
| CR332002 | In earlier versions of BEA JRockit, the JVMTI functions `FollowReferences` and `IterateThroughHeap` did not respect the `klass` parameter. This has been fixed. |
| CR331991 | In earlier versions of BEA JRockit, the JVMTI function `GetClassVersionNumbers` did not return `JVMTI_ERROR_ABSENT_INFORMATION` for primitive and array classes. This has been fixed. |
| CR331724 | When running AquaLogic Service Bus SFTP tests, BEA JRockit was creating regular core dumps. This issue occurred when two mutually exclusive code paths doing an arraycopy to the same to-array were both subject to an erroneous optimization. |
| CR328924 | BEA JRockit no longer fails the Java Language Specification requirement on unique references for boxed integers in the -128 to 127 interval. |
| CR328368 | Allocation prefetch has been enabled by default on AMD's Opteron architectures. |
| CR328154 | In some Solaris environments, BEA JRockit was unable to detect the number of sockets, cores and hardware threads. This caused the JVM to abort during start up and display error message:<br>`[ERROR] Fatal error in JRockit during memory setup phase.`<br>This situation would occur in a local zone associated with a subset of all available processors. This issue has been resolved. |
| CR327167 | The JVMTI ClassPrepare event was previously dependant on class initialization order and thus subject to user class hierarchy design. In R27.4 this has changed so that ClassPrepare is always sent according to specification. |

**Table 2-6 Changes in the BEA JRockit JVM R27.4**

| Issue ID | Description |
|---|---|
| CR321557 | The experimental and unsupported API GCControl, containing the methods<br>• `jrockit.ext.GCControl.forceGCToExit(boolean enabled)`<br>and<br>• `jrockit.ext.GCControl.fullGC()`<br>is now removed. |
| CR321348 | Call profiling is an optimization feature known to provide significant benefit to many workloads, including the SPECjbb2005 benchmark. Up until BEA JRockit R27.1, you could enable this feature by using the `-XXaggressive` command-line option, but it was removed from the flag in R27.1. As of BEA JRockit R27.4, you can enable call profiling by using the `-XXcallProfiling` command-line option. |
| CR318629 | Due to a bug in the attach framework (Sun bug #6559427), Mission Control was leaking several handles per locally-running JVM (JVM running on the same machine as Mission Control is) every time a Mission Control polls for locally running JVMs. This has been fixed in R27.4. |
| CR311802 | Some customers experienced linked lists breaking, which would result in leaking objects caused by a modification of next pointers through reflection. This is now fixed. |
| CR304741 | This version of BEA JRockit has two new Long perf variables:<br>• `jrockit.gc.oc.compactionInternalCount`<br>• `jrockit.gc.oc.compactionExternalCount.`<br>These variables count the number of internal and external compactions, respectively. They both sum up to the previously existing `jrockit.gc.oc.compaction.no` value. |
| CR104868 | A rewrite of an internal code generation framework for R27.4 has eliminated known bugs that were causing BEA JRockit to crash. |

# Changes in the BEA JRockit JVM R27.3

This section lists changes and known issues in the BEA JRockit R27.3

## Known Issues in the BEA JRockit JVM R27.3.1

Table 2-3 lists known issues in this version of the BEA JRockit JVM.

**Table 2-7 Known Issues in the BEA JRockit JVM R27.3.1**

| Issue ID | Description |
|---|---|
| CR336813 | BEA JRockit might sometimes incorrectly optimize loops, assigning the same negative value to all elements of an array. |
| CR331774 | If you are running on Linux or Solaris and press Ctrl-C to properly shut down your application, it will actually terminate immediately and you risk losing any runtime data that hasn't been saved to disk or a database. This happens because BEA JRockit fails to register the SIGINT signal handler used for the shut down hooks.<br>**Workaround:**<br>If you encounter this problem, please download the updated version of the product, R27.3.1 from the BEA Downloads page at:<br>`http://commerce.bea.com/products/weblogicjrockit/jrockit_prod_fam.jsp`<br>This issue does not apply to applications running on Windows. |

## Changes in the BEA JRockit JVM R27.3

Table 2-8 lists changes made in this version of the BEA JRockit JVM.

**Table 2-8 Changes in BEA JRockit JVM R27.3**

| Change Request ID | Description |
|---|---|
| CR327343 | The documentation for jrcmd at:<br>`http://edocs.bea.com/jrockit/geninfo/diagnos/ctrlbreakhndlr.html#wp1000350`<br>now includes information on the known limitations of jrcmd. |
| CR324513 | The default preferred TLA size (`-XXtlaSize`) has changed, see the BEA JRockit Command-Line Reference Manual for details. |
| CR323910 | JRockit could crash due to stack overflow while optimizing very large methods. This has now been fixed. |
| CR323086 | JRockit gave unexpected errors with the `-Xcheck:jni` command-line option. JRockit would detect a false positive when using JNI to do a downcast array assignment (assign an array of subclass to Object as element to an array of Object arrays, i.e `set [byte -> [[Object)`. |
| CR322633 | JRockit sometimes could not load very large jsp classes, which resulted in the error:<br>`java.lang.ClassFormatError: <class> : illegal attribute length (SourceDebugExtension:91802)`<br>This error was caused by that JRockit used a limit of 65536 on the `SourceDebugExtension` attribute. This limit has now been removed. |
| CR322146 | The garbage collector mode `-XgcPrio:pausetime` now uses a fixed nursery of the same size as `-Xgc:gencon`, which is 10MB times the number of hardware threads. |
| CR321899 | When using the parallel garbage collector and if an evacuation was aborted because the time limit was reached, the evacuation area size was doubled. This bug could cause unnecessary long pause times in the parallel garbage collector. This has now been fixed. |
| CR321325 | The JVMTI function `GetAllStackTraces` previously returned `JVMTI_ERROR_ILLEGAL_ARGUMENT` if the `max_frame_count` parameter was `0` (zero). This has now been fixed. |
| CR319804 | The JVMTI function `GetObjectMonitorUsage` could return `JVMTI_ERROR_THREAD_NOT_ALIVE` if the thread holding the object terminated during the call. To comply with the specification, this has now been changed to return `JVMTI_ERROR_NONE` and set `info_ptr->owner` to `NULL` instead. |

**Table 2-8 Changes in BEA JRockit JVM R27.3**

| Change Request ID | Description |
|---|---|
| CR319764 | JRockit now handles names that are as long they are allowed by the ZIP standard. The previous limitation of maximum 512 bytes long entry names in zip files no longer exists. |
| CR319239 | JRockit did not always find all free memory in the fourth GB of the virtual memory space. This bug manifested on 64 bit Linux platforms and could lead to `OutOfMemoryErrors` when using a maximum java heap size between approximately 3 and 4 GB. This has now been fixed |
| CR319234 | The JMAPI `getProcessAffinity/suggestProcessAffinity` now works correctly when running on a Linux system with a GLIBC version older than 2.3.4. |
| CR317171 | Pause time measurements between R27.3 and earlier JRockit releases (except for JRockit R27.2) are now comparable. |
| CR317113 | JRockit now reports the correct amount of physical RAM in 32 bit machines with PAE extension and more than 4GB of RAM installed. |
| CR316942 | JRockit no longer hangs if you specify a nursery size (`-Xns`) that is less than 18 times the thread-local area size (`-XXtlaSize`). |
| CR315905 | The Mercury Profiler tool has been omitted as of this release. |
| CR315761 | It is now possible to use the `EPollSelectorProvider` in `java.nio` on Linux ia64.<br>**Note:** The `EPollSelectorProvider` is only used if the system property `java.nio.channels.spi.SelectorProvider` has been set to `sun.nio.ch.EPollSelectorProvider`. |
| CR314598, | In JRockit R27.1 and R27.2, when trying to run some MBean servers, some classes could not be found, even though they were on the classpath. This problem has now been fixed. |
| CR314527 | Previously, on rare occasions, external compaction caused very long pause times (if the heap was fragmented) when trying to move a large object from the highest heap parts. This has now been fixed. |
| CR314031 | JRockit no longer calculates the wrong `serialVersionUID` for some classes backported from JDK 5 to JDK 1.4 where Enums is used. Enums are now correctly ignored (they are not part of the 1.4 specifications) when calculating the `serialVersionUID` for 1.4. |

**Table 2-8 Changes in BEA JRockit JVM R27.3**

| Change Request ID | Description |
|---|---|
| CR312360 | The `-Xmanagement` flag has been updated.<br>You start a local in-memory connector if start without any arguments.<br>You start a remote connector if:<br>• Any of the options `authenticate`, `ssl`, `port`, or `autodiscovery` is set.<br>• Any of the above options is set through system or management properties, (`com.sun.management.jmxremote.port`, `jrockit.management.port`, etc.).<br>• The password file, pointed to by the management property `com.sun.management.jmxremote.password.file` (default `jmxremote.password`), exists.<br>The remote connector requires that you have username and password defined in the above password file unless the option `authenticate` is set to `false`.<br>The remote connector uses SSL by default, unless the option `ssl` is set to `false`. |
| CR312194 | To see the version of the time zone data (tzdata) of a JRE you can now run:<br>`<jdk>\bin\tzinfo`<br>The output shows Java version, JRE version, and Time Zone data (tzdata) version. |
| CR311518 | The default TLA size now grows more aggressively and it has been increased to 2k at a 16Mb heap up to 256k at a 2GB heap. |
| CR311336 | This release offers a new heuristic for updating the nursery size (for the static garbage collector) during runtime. |
| CR311327 | The value for `TotalGarbageCollectionTime` is now showed in milliseconds on Windows XP. |
| CR293617 | Singlestepping with JDI on a single-CPU computer is now faster and easier to use. |
| CR262438 | JRockit no longer fails to detect whether `HyperThreading` (HT) has been enabled or not, which means that it will no longer start a non-optimal number of garbage collection threads. |
| CR206755 | The initial heap size will now be at least twice the size of the nursery if `-Xns` (the nursery size) is set and `-Xms` (the initial heap size) is not, unless this leads to that the initial heap size becomes larger than the maximum heap size. |

# Changes in the BEA JRockit JVM R27.2

Table 2-9 lists changes made in this version of the BEA JRockit JVM.

**Table 2-9 Changes in the BEA JRockit JVM R27.2**

| Change Request ID | Description |
|---|---|
| CR315538 | Problems were occurring because the JRA was unable to handle class unloading. This situation has been corrected. |
| CR311708 | BEA JRockit no longer detects false positive Java deadlocks. |
| CR311186 | A regression in BEA JRockit R27.1 caused `-Xverbose` output to be buffered and therefore delayed. The output is no longer delayed. |
| CR310238 | The `javaw` launcher in JDK 6 on Windows supports class-path wildcards. This is a backport of a fix for the problem reported by Sun in http://bugs.sun.com/bugdatabase/view_bug.do?bug_id=6510337 |
| CR309555 | The JNI method `ToReflectedMethod` crashed if the class parameter was NULL. This has now been fixed. |
| CR308967 | Sometimes BEA JRockit crashed when the method `java.util.concurrent.atomic.AtomicReferenceArray.compareAndSet` was called. This has now been fixed. |
| CR308312 | The byte code verification in BEA JRockit has been relaxed in cases where JRockit's strict byte code verification would otherwise cause `ClassFormatErrors` to be thrown. |
| CR307903 | If a thread was interrupted for garbage collection while copying an array, the garbage collection could result in long pauses. This has now been fixed. |
| CR307114 | When reflection was used to set volatile static variables in JRockit on Windows, JRockit crashed. This has now been fixed. |
| CR306848 | The `jstat` tool (in the `/bin` directory) used counters that triggered error messages about Unresolved Symbols. This has now been fixed and `jstat` no longer uses the obsolete counters. |
| CR306729 | The heuristics used by `-Xgcprio:throughput` to set nursery size and select garbage collector strategy have been improved. |
| CR306048 | The `referrer_index` argument to the `jvmtiObjectReferenceCallback` function was not always set to `-1` when it should have been. This has now been fixed. |

**Table 2-9 Changes in the BEA JRockit JVM R27.2**

| Change Request ID | Description |
|---|---|
| CR305091 | The `jrcmd` utility on ia64 had problems with user names longer than 9 characters. This has now been fixed. |
| CR304733 | BEA JRockit's method profiler timing counters are no longer available on Fujitsu's SPARC implementation SPARC64, since JRockit sometimes gave the wrong timing data (negative numbers) on that platform. |
| CR304335 | The method `getNurserySize()` in the `GarbageCollector` class now works as documented, that is, it throws a `NotAvailableException` when the JVM is running a single-spaced garbage collector (without a nursery). Previously, the method returned `0 (zero)`.<br>The method `setNurserySize()` now throws a `NotAvailableException` when the JVM is running a single-spaced garbage collector. |
| CR303790 | The new command line option `-XX:MaximumNurseryPercentage` limits the maximum size of the nursery to a percentage of the free heap space available after the latest old collection. The default value is 95%. |
| CR302924 | A ctrl-break handler can now be sent to stop a JRA recording even if JRA has not actually started recording yet (but is in a start up state). If JRA has just been started, then there may be a short delay before the recording is actually stopped.<br>Previously, if you sent a ctrl-break handler to stop JRA before it had actually started recording, you would have generated the following error message:<br>`Error: No JRA recording running.` |
| CR301964 | Issues with thread names not being available in Linux environments are fixed. |
| CR299662 | The parameters `genpar` and `singlepar` have been added to the `-Xgc` option. Using `-Xgc` with these parameters are equivalent to using the `-XXsetgc` option with the parameters `genparpar` and `singleparpar`. |
| CR299651 | The option `-XlargePages` has been added. This option is intended to replace `-XXlargePages` but the old command-line option is retained for backward compatibility purposes. |

**Table 2-9 Changes in the BEA JRockit JVM R27.2**

| Change Request ID | Description |
|---|---|
| CR298847 | At all times, the following relations are now automatically maintained between minimum and preferred TLA size, large object limit, and minimum block size:<br>`-XXlargeObjectLimit` <= `-XXtlaSize:min` <= `-XXminBlockSize`<br>`-XXtlaSize:min` <= `-XXtlaSize:preferred`<br>If you set two or more of the options, then you must make sure that the values you use fulfil these criteria. If you only set one of the options, the others will adapt if necessary so that the values are valid.<br>It is recommended that you primarily set the TLA size parameters for memory management tuning purposes, while letting BEA JRockit automatically adjust the large object limit and minimum block size. By default, the large object limit will be set to whichever is the lower value of (1) the minimum TLA size and (2) the preferred TLA size divided by 2. The default minimum block size is 2k. |
| CR297814 | The limit of capturing only the first 20 frames of allocation stack traces in the Memory Leak Detector has been removed. |
| CR278996 | `-Xverbose:referents` (alternatively, `-Xverbose:verboserefs`) gives you printouts of all reference objects for each old generation garbage collection as well as the referents to which they point. In previous BEA JRockit versions, this could be achieved with the option `-Djrockit.verboserefs`. |

# Changes in the BEA JRockit JVM R27.1

Table 2-10 lists changes made in this version of the BEA JRockit JVM.

**Table 2-10 Changes in the BEA JRockit JVM R27.1**

| Change Request ID | Description |
|---|---|
| CR266204 | Optimized method calls on null objects now throws `NullPointerException` as is the expected behavior. |
| CR264251 | The JRockit installer on Windows previously only had an “Install Public JRE” option on 32-bit Windows. This option is now also available with the 64-bit JRockit versions for Windows (x86-64 and Itanium). |
| CR274190 | The command-line option `-Xverbose:gcpause` has been improved, see `-Xverbose` in the BEA JRockit Reference Manual for more information. |

**Table 2-10 Changes in the BEA JRockit JVM R27.1**

| Change Request ID | Description |
|---|---|
| CR276950 | JRockit R27.1 and later no longer supports Linux versions using the old `LinuxThreads` library, which includes Red Hat AS 2.1, SuSE ES 8.0, and other distributions based on the standard 2.4 kernel. See the Supported Configurations documentation for information on supported releases. |
| CR277922 | It is now possible to install both the 32-bit and the 64-bit BEA JRockit JDK/JRE on Windows x86_64, side by side. |
| CR280059 | JRockit no longer generates faulty machine code for Java code that stores the `>this<` pointer into its own object. This caused Java variables to be set to zero in rare situations. |
| CR281323 | Internal JVM threads in the "system" ThreadGroup did not have the correct value for `isDaemon()` and `getPriority()`. This has now been fixed. |
| CR281936, CR283237 | Previously, if the memory was very fragmented when JRockit started, JRockit crashed during startup without creating any dump file. This problem is now fixed. |
| CR283236 | The behavior of `java.lang.StackTraceElement` has been changed to conform to that of the Sun reference implementation. In previous releases the method signature was included in the resulting String, while the new version behaves as described in:<br>`http://java.sun.com/j2se/1.5.0/docs/api/java/lang/StackTraceElement.html#toString()`<br>This also affects the behavior of `Throwable.printStackTrace()`, in that the result will not list any method signatures. |
| CR283454, CR283915 | Previously long thread sleeps issued by `Thread.sleep(...)` and `Object.wait(...)` could end too early if the sleeps were longer than `0x3FFFFFFF` milliseconds (approximately 12.4 days). This problem has now been fixed. |
| CR284604 | The JVMTI API version of the Java SE 5.0 version of JRockit R27.1 is 1.1. This can be seen with the `GetVersionNumber` API function. Some capabilities of JVMTI 1.1 are not available in the Java SE 5.0 version of JRockit. |
| CR286267 | Previously an uncommon internal deadlock could occur. In the stack traces of a thread dump, the deadlock situation may have had calls that had to do with class loading. |
| CR286625 | The BEA standalone installer filenames for Windows platforms have changed names. Now "windows" is used instead of "win" to name the target operating system in the platform part. |

**Table 2-10 Changes in the BEA JRockit JVM R27.1**

| Change Request ID | Description |
|---|---|
| CR286926 | Command line option verification has become stricter and you will receive more comprehensive error messages and warnings when using command line options that are incorrect. See the BEA JRockit Reference Manual for details. |
| CR291898 | JRockit could in rare circumstances crash when compiling methods calling `String.indexOf()`. This has been fixed. |
| CR291969 | JRockit no longer crashes due to a previous bug in the JRockit code optimizer that was triggered by applications throwing a large amount of exceptions. |
| CR294608 | By default, the output from `-Xverbose` now includes the logging level as well as the module. Use `-XverboseDecorations` to change the default settings. |
| CR296429 | The JMAPI function `JVM.getProcessAffinity()` now works properly on Linux. |
| CR296668 | A bug that caused JRockit to crash in the internal function `handlersMatchExceptForUnlockHandler` has been fixed. |
| CR297036 | The Security Vulnerability in RSA Signature Verification issue, Sun Alert ID 102686, fixed in Sun JDK 1.4.2_13 has been back ported to JRockit 1.4.2_12 R27.1.0. |
| CR297037 | The Security Vulnerability in RSA Signature Verification issue, Sun Alert ID 102686, fixed in Sun JDK 5.0 update 9, has been back ported to JRockit 5.0 update 8 R27.1.0. |
| CR297675 | Now JRockit shows the correct value for `TotalGCTime`.<br>The incorrect value could previously be observed through the JRockit Management API, and through the WLS Console. |
| CR298049 | JRockit no longer crashes when compiling bytecode where normal control flow and exception control flow do not follow javac conventions. As far as BEA is aware, such bytecode is only produced by a small set of bytecode obfuscators. |
| CR298365 | The command-line option `-Xmanagement` has changed behavior in this release. |
| CR301758 | As part of the rewrite of JRCMD in R27, the `<jre>/bin/jrcmd` launcher has been removed (the `<jdk>/bin/jrcmd` still remains), i.e. the JRCMD tool is now only available if installing the full JRockit JDK, not only the JRockit JRE. |
| CR302559 | The “Security Vulnerability in Serialization” issue, Sun Alert ID 102731, fixed in Sun JDK 1.4.2_13, has been back ported to JRockit 1.4.2_12 R27.1. |

# Known Issues

Table 2-11 lists issues that, in some circumstances, might affect the performance of the JRockit JDK.

**Table 2-11 Known Issues in the JRockit JDK**

| Issue ID | Description |
|---|---|
| CR372377 | Sometimes, when debugging the throwing of an uncaught `java.lang.ClassCastException` with Eclipse, the debugged JVM might crash. |
| CR371381 | The JRockit JVM is incompatible with Sun HotSpot when serializing `java.math.BigDecimal` objects over IIOP. |
| CR364607 | Supplying a file name (including path) the length of which exceeds 256 characters to the `java.util.zip.ZipFile` constructor on Windows will fail and throw a `java.io.FileNotFoundException` even if the file exists. |
| CR363637 | The JRockit JVM might occasionally crash in the `pkcs11_softtoken.so` native library.when executing AES cryptography code. This is due to a bug in Solaris versions 10.4 and earlier. The bug has been fixed in Solaris 10.5. |
| CR363487 | Due to an issue with 64-bit Windows Vista/2008, if you try to install the 64-bit JRockit JVM in the standard folder, you'd be told the installer cannot write there; however if you chose to install to another folder (one that the installer *can* write to), the files are copied and then the installer tries to write the uninstaller information and fails. The result is a broken installation without a working uninstaller.<br>**Workaround:**<br>Right-click on the file and select **Run as administrator** when installing the JRockit JDK. |

**Table 2-11 Known Issues in the JRockit JDK**

| Issue ID | Description |
|---|---|
| CR361457 | Connecting the Memory Leak Detector to a running JRockit JVM (version R26.3 through R27.5) built for Linux IA-32 might cause the JVM to crash if the JVM process uses more than about 1020 file descriptors at the time. This might only happen if the file descriptor limit has been set higher than 1024 (typically by using the `ulimit` command).<br>**Workaround**:<br>Currently you can start the Memory Leak Detector Server at JVM startup, when few file descriptors are in use. To do this, add `-Djrockit.memleak.port=12345` early in the JVM command line.<br>Now, using JRockit Mission Control, create a custom connection in the JRockit Browser with a Custom JMX service URL of `service:jmx:mlp://localhost:12345`. (Replace *`localhost`* and the port *`12345`* as needed). Using this connection, you can connect the Memory Leak Detector in JRockit Mission Control to this JVM once (without restarting the JVM).<br>Note that using many file descriptors might be an indication of a resource leak in the Java application. Make sure to always close opened files and sockets. You should not rely on the Garbage Collection and the object finalization to free a non-Java resource such as a file descriptor. See Too Many Open Files at:<br>`https://support.bea.com/application_content/product_portlets/support_patterns/wls/TooManyOpenFilesPattern.html`<br>for more information. |
| CR361032 | When debugging Java floating point applications on SSE2-enabled platforms (supported on Pentium 4 or newer and AMD Opteron, Athlon 64 from 2003 or later processors) using a 32-bit version of the JRockit JVM on Windows Vista x64, local floating point variable values might be bogus. This is due to a bug in Windows Vista x64. It has been fixed in Windows Vista x64 SP1. |
| CR359954 | In JRockit Mission Control released with BEA JRockit JVM R27.4, the value shown in **Heap Usage Before** for a garbage collection in a JRA recording is incorrect. The value shown is actually the value of **Heap Usage After** for the proceeding collection. This will be fixed in BEA JRockit JVM R27.5. |

**Table 2-11 Known Issues in the JRockit JDK**

| Issue ID | Description |
|---|---|
| CR359328 | The ACL on the per-user `hsperfdata_<user>` directory might cause administration problems on some systems. One example of the problem is that the files under the directory can be removed by a user with Administrator privileges but the directory itself cannot.<br>**Workaround:**<br>Logged in as Administrator, modify the ACL with the `cacls` command before attempting to remove it. |
| CR359311 | Under rare and very specific circumstances, the JRockit JVM might prematurely garbage collect a member variable. This will cause spurious behavior, including hangs or crashes. |
| CR357402 | If a Java program uses too many (currently 2097151) active monitors (that is, by doing wait/notify or contended synchronization on too many objects) the JVM's internal monitor index can overflow. This is more likely to happen when using large heaps and few garbage collections occur. If this happens, the JVM will crash with an error saying "The number of active Object monitors has overflowed." |
| CR349882 | JRockit JDK installation and the use of applications on compositing window managers under Linux may fail with the following error message: `xcb_xlib.c:50: xcb_xlib_unlock: Assertion 'c->xlib.lock' failed.` This is due to a known problem in the native Sun JDK libraries, see Sun Bug 6532373.<br>**Workaround:**<br>To work around this issue, please refer to Workarounds for CR349882. |
| CR348820 | Debugging a Java program when running the JRockit JVM with hardware performance counters on IA64 (`-XXhpm`) can result in exceptions. This is because the implementation of the hardware performance counters uses signals to communication with the JVM and the debugger agent implementation (JDWP) does not correctly handle this.<br>**Workaround:**<br>Disable hardware performance counters when debugging. |
| CR341568 | The JRockit JDK exposes Mbeans under the `bea.jrockit.management` domain, accessible through the Management Console. These MBeans are proprietary and subject to change at any time. While you can access these MBeans directly using other JMX based tools, such direct access is not supported by Oracle. |

**Table 2-11 Known Issues in the JRockit JDK**

| Issue ID | Description |
|---|---|
| CR340838 | When starting the JRockit JVM with a maximum heap size larger than the maximum address space (4 GB) on Linux ia32, the JVM might crash, instead of terminating nicely and generating an error message, as it should. |
| CR340660 | Calling the methods `isLoaded()`, `load()` or `force()` in `java.nio.MappedByteBuffer` on an empty buffer throws an IOException instead of silently returning. This is know to affect the IntelliJ IDEA IDE. |
| CR339469 | Copying event information from the Thread Latency Log table to the clipboard does not work properly. Only the header information will be copied. This issue will be fixed in the Mission Control included in JRockit JVM R27.5.0. |
| CR338731 | Some events in the JRA latency recordings have their thread ID's set to 0. In particular, this applies to `JVM Event Wait->Signalling thread`, `Java Synchronization->Last holder thread` and `Java Synchronization->Holder thread`. |
| CR338678 | If you are running JRockit JDK 1.4.2, you might receive an incorrect error message when using the command-line option `-Xmanagement` with the parameter `class`, followed by additional parameters; for example:<br>`java -Xmanagement:class=foo,ssl=false Hello`<br>results in this error message:<br>`Unknown parameter class`<br>`Could not create the Java virtual machine`<br>You cannot specify any parameters after the `class` parameter, so the correct error message should be:<br>`Unknown parameter ssl`<br>`Could not create the Java virtual machine`<br>For more information, please refer to the `-Xmanagement` documentation. |
| CR337697 | Compiling a program that uses the JRockit Management API (JMAPI) with `javac` from a Java SE 6 version of the JRockit JDK will give an error saying that "`package com.bea.jvm` does not exist".<br>**Workaround:**<br>Do one of the following:<br>• Delete (or rename) the file `<jrockit_home>\lib\ct.sym` and then recompile.<br>• Use `javac` from a 5.0 version of the JRockit JDK instead. |

**Table 2-11 Known Issues in the JRockit JDK**

| Issue ID | Description |
|---|---|
| CR337475 | In a JRA recording, the number of allocated TLA (Thread Local Areas) is recorded, as well as the preferred size of a TLA (in bytes). The JRA GUI will multiply these values to get the number of bytes allocated in TLAs during the entire recording; however, the size of the TLAs actually used can sometimes be a bit smaller than the reported size (the preferred size is only a preferred size; fragmentation can cause the TLAs to become smaller) and the value printed in the GUI can be overestimated. |
| CR328975 | Latency data in a JRA recording will be erased from the disk if comments on the Notes tab in Mission Control are saved. Non-latency data will still be available, but the message "`Warning! Error(s) when reading JRA-recording`" will appear.<br>**Workaround:**<br>Don't use the Notes tab in Mission Control when working with recordings that contain latency data. |
| CR328964 | The `-XXcompactSetLimit` flag does not always limit the compaction set. In some circumstances compactions can exceed the given limit, typically in the initial compactions before the whole heap has been processed for external compaction. |
| CR328729 | When starting a JRA-recording by using Mission Control, the recording might not start and the error message, "Could not delete file" will appear. This happens when the recording has the exact same filename as a previously-started recording.<br>**Workaround:**<br>In the JRA-recording wizard, give each recording a unique name or close Mission Control and restart it. |
| CR326746 | The `set_filename` handler will not update the output for the running command batch.<br>**Workaround**:<br>1. Issue a `set_filename` command.<br>2. Issue the commands that you wish to send to the output. |
| CR322908 | A known issue in Red Hat Enterprise Linux 5.0 on x86_64 with the `dladdr()` call in glibc might cause irregular behavior or a crash when running graphical applications; see also Red Hat Errata RHBA-2007:0619-3 at `http://rhn.redhat.com/errata/RHBA-2007-0619.html`. The issue is fixed in Red Hat Enterprise Linux 5.1. |

**Table 2-11 Known Issues in the JRockit JDK**

| Issue ID | Description |
|---|---|
| CR317171 | A regression has been introduced in R27.2 in how pause times are measured. Pause times are visible in the JRockit Runtime Analyzer Tool and the verbose logs in the `Mark:Final:StopThreads` pause part, where they appear to be much longer than in previous JRockit JVM versions. This means that pause time measurements are not comparable between R27.2 and earlier JRockit JVM versions. The actual pause times have not changed.<br><br>**Note:** This issue has been fixed in BEA JRockit R27.3. |
| CR316942 | If you specify a nursery size (`-Xns`) that is less than 18 times the thread-local area size (`-XXtlaSize`), the JRockit JVM will hang without printing an error.<br><br>**Workaround**:<br><br>Increase the specified nursery size (using `-Xns`) or lower the minimum TLA size.<br><br>As of JRockit R27.1 the format for how to specify TLA size changed to specify both minimum and preferred TLA size. The old way (`-XXtlaSize:<size>`) sets both minimum and preferred. Use `-XXtlaSize:preferred` to set the preferred TLA size, for example: `-XXtlaSize:preferred=64k`.<br><br>**Note:** This issue has been fixed in BEA JRockit R27.3. |
| CR315939 | If you are using a 32-bit JVM and set the maximum heap size to a value above 4 GB, the JRockit JVM will allocate as large a heap as possible, but not exceeding 4 GB. This can result in the JVM throwing an internal out of memory error because the heap has taken all the address space.<br><br>**Workaround:**<br><br>When you encounter this situation, reduce the heap to a value less than 4 GB. |
| CR315761 | It is not possible to use the `EPollSelectorProvider` in `java.nio` on Linux ia64 with JRockit 5.0 R27.2. Note that the `EPollSelectorProvider` is only used if the system property `java.nio.channels.spi.SelectorProvider` has been set to `sun.nio.ch.EPollSelectorProvider`.<br><br>**Note:** This issue has been fixed in BEA JRockit R27.3. |

**Table 2-11 Known Issues in the JRockit JDK**

| Issue ID | Description |
|---|---|
| CR312235 | The code garbage collection is disabled during JRA recordings, so you might in special (rare) circumstances see an increased use of native memory during recordings. This can happen if you load a lot of classes when you do either a very long recording (several hours or even days) or shorter recordings back-to-back.<br>**Workaround:**<br>The workaround is to do several recordings, but leave some time (a few minutes should suffice) between JRA runs, so the JVM can run the code garbage collection. |
| CR311188 | On Solaris 10, a bug that makes `getrusage` return bogus values in turn causes all printouts of page faults to present bogus values. You may get these printouts when you, for example, use `-Xverbose:memory`.<br>The Solaris bug is identified as "6288308, Uninitialized struct causes getrusage(3C) to return bogus data". According to Sun, the first kernel patch with a fix for bug 6288308 is 118833-24. See http://sunsolve.sun.com/search/document.do?assetkey=urn:cds:docid:1-21-118833-24-1 |
| CR310666 | A known issue in Sun's J2SE 5.0 update 11 might cause BEA JRockit to dump when `PrinterJob.printDialog()` is called from a sub-thread. BEA has only identified the bug using Windows Vista on the IA32 architecture. This known issue will be fixed in 5.0 update 11 and included in JRockit R27.3.<br>More information can be found in the Sun bug database: http://bugs.sun.com/bugdatabase/view_bug.do?bug_id=6358747 |
| CR310230 | The new launcher `java-rmi.exe` that is included in JDK 6 on Windows does not work as expected. This is also reported by Sun in the original bug report at http://bugs.sun.com/bugdatabase/view_bug.do?bug_id=6512052 |
| CR307903 | If a thread is interrupted for garbage collection while it is in the process of copying an array, then the garbage collection may result in very long pauses. If you get occasional long pause times, this may be the problem. Note that this issue has been fixed in BEA JRockit R27.2. |
| CR307902<br>*For Linux users only.* | If you explicitly request to use the Motif AWT instead of the default X11 AWT on Linux/IA64 and run a Linux version with a GLIBC version older than `glibc-2.3.4`, this operation might fail with an `UnsatisfiedLinkError` since the file `<jre>/lib/ia64/motif21/libmawt.so` requires linkage to `GLIBC >= 2.3.4`.<br>See the see the Supported Configurations document for supported Linux versions. |

**Table 2-11 Known Issues in the JRockit JDK**

| Issue ID | Description |
|---|---|
| CR305844 | If you are using the JRockit JDK on the Itanium version of Windows Server 2003 and the Java application unexpectedly hangs during heavy system load, then it might mean that the JVM has triggered an operating system bug. At the OS level, this is manifested as the JRockit JVM blocking on a call to the Windows `GetThreadContext` function. Microsoft has posted a knowledge base article that also includes instructions for obtaining a hotfix for the problem. It is available at `http://support.microsoft.com/kb/947504`. |
| CR305091 | The `jrcmd` utility can have problems on ia64 if your username is longer than 9 characters. This is not a problem on other platforms. The issue has been fixed in BEA JRockit R27.2. |
| CR304556 | If the JRockit JVM is started with a minimum TLA size (`-XXtlaSize:min=X`) that is larger than the maximum specified heap size (`-Xmx`), JRockit will deadlock at startup and never start running.<br>**Workaround:**<br>Set a minimum TLA size that is less than the maximum heap. Typically, the TLA size should be much smaller than the heap size. |
| CR304335 | In R27.1, the JMAPI method `getNurserySize()` in the `GarbageCollector` class doesn't work as documented. If the garbage collector that the JVM is running isn't using a nursery, the method should throw a `NotAvailableException`. Instead it returns `0`. This has been fixed in R27.2.<br>**Workaround:**<br>If you depend on the exception being thrown, e.g. checking if you use a nursery or not, you can work around the problem by both catching the `NotAvailableException`, as well as checking the return value and see if it returns `0`. If it throws an exception, or returns `0`, a nursery is not being used. |
| CR302141 | Files containing JRA recordings can be dragged and dropped into JRockit Mission Control. However, when dropping multiple files, some open file tabs may be labeled "JRA Editor" instead of the actual file name.<br>**Workaround:**<br>Select a tab for the file, then the file is actually read and the label is set to the correct file name. |

**Table 2-11 Known Issues in the JRockit JDK**

| Issue ID | Description |
|---|---|
| CR300393 | If the nursery is too small, JRockit may get stuck in triggering young collections, "back to back", without promoting anything. This can be seen in the `-Xverbose:memdbg` outputs as repeated young collections where the number of promoted objects is zero. It can also be seen as very short times between the young collections (close to 0 ms).<br>**Workaround:**<br>Increase the nursery size. If nursery size has been set automatically by `-Xgcprio:throughput`, it can be overridden by manually setting `-Xns` to a higher value. |
| CR300097 | A known issue in Red Flag 5.0 with the `wait()` call might cause irregular behavior or crashes for Red Flag customers using this OS version. This has been fixed in AsianUX 2.0 SP1 (of which Red Flag 5.0 is a part) and we strongly recommend that our users upgrade their OS to resolve this issue. |
| CR295457 | If an application is configured with a heap size close to 4 GB and includes a lot of classes, an out of memory situation might occur if some JVM internal structures and the Java heap both try to share the low 4 GB memory space of the process. If this happens, try to increase or decrease the Java heap size by using either the `-Xmx` option or disable compressed references by using the `-XXcompressedRefs=0` option. |
| CR284519 | To use the `-XXmme` option on Red Hat Enterprise Linux 4.0 you need to have Red Hat Enterprise Linux 4.0 QU4 or a later release installed; otherwise, you might encounter sporadic crashes. |
| CR283776 | In 5.0 Update 7, Sun changed the `serialVersionUID` for the `javax.xml.namespace.QName()` class due to a historical defect. For the original bug report, see CR6267224 in Sun's bug database at:<br>`http://bugs.sun.com/bugdatabase/view_bug.do?bug_id=6267224`<br>To use the old compatibility value, set the following system property:<br>`com.sun.xml.namespace.QName.useCompatibleSerialVersionUID=1.0` |

# Workarounds for CR349882

This section instructions for working around the known issue described in CR349882.

## Workaround #1

Apply the following patch (Listing 2-1), `patch_java.sh`, to the unpacked JDK or JRE.

**Listing 2-1 patch_java.sh Sample**

```
$ cat patch_java.sh
#!/bin/sh
jh=$1
sed -i 's/XINERAMA/FAKEEXTN/g' $jh/lib/*/xawt/libmawt.so
sed -i 's/XINERAMA/FAKEEXTN/g' $jh/lib/*/motif21/libmawt.so

$ patch_java.sh <path-to-jrockit-jdk>/jre
$ patch_java.sh <path-to-jrockit-jre>
```

## Workaround #2

In case the JRockit JDK installer `*.bin` fails immediately with the same error message as above, you can work around it by unpacking the installer program manually, applying the following patch (Listing 2-2) to the internal JRE, then starting the installer yourself in GUI mode using the patched internal JRE.

**Listing 2-2 patch_and_run_installer.sh Sample**

```
$ cat patch_and_run_installer.sh
#!/bin/sh
jrinstaller=$1
rm -rf mytmp
unzip -d mytmp $jrinstaller
cd mytmp
GUI=`cat autorun.inf |grep GUI= | cut -d= -f2`
UNZIP=`cat autorun.inf |grep UNZIP= | cut -d= -f2`
UNZIPTO=`cat autorun.inf |grep UNZIPTO= | cut -d= -f2`
unzip -d $UNZIPTO $UNZIP
jh=$UNZIPTO
sed -i 's/XINERAMA/FAKEEXTN/g' $jh/lib/*/xawt/libmawt.so
sed -i 's/XINERAMA/FAKEEXTN/g' $jh/lib/*/motif21/libmawt.so
```

```
$GUI
$ patch_and_run_installer.sh <jrockit-installer>.bin
```

---

After successful installation using workaround #2 you might also have to apply workaround #1, using the path to the installed JRockit.

CHAPTER 3

# R26 Release Information

This document contains important details for BEA JRockit R26. It contains information on the following subjects:



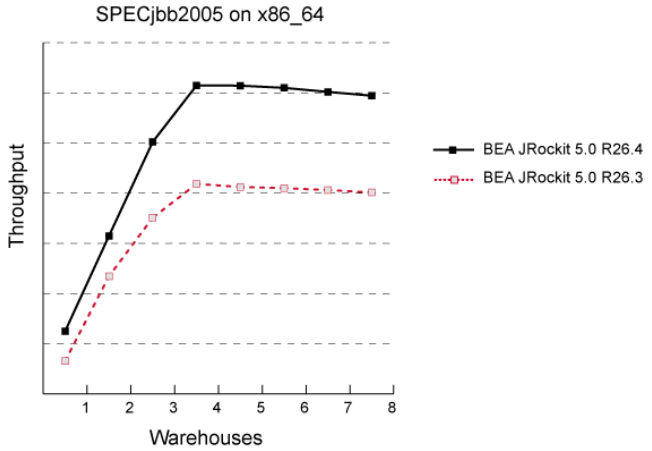
## New Features and Enhancements in BEA JRockit R26.4

BEA JRockit R26.4 provides, apart from full support for J2SE 5.0 on all supported platforms, the following areas of improvement:

# Performance Improvements on 64-bit Platforms

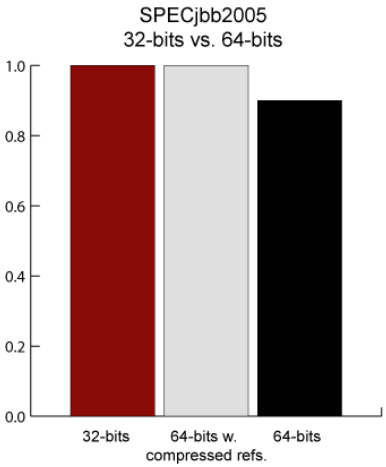
- General improvements on x86_64 platforms. See Figure 3-1 for a comparison between the BEA JRockit R26.3 and R26.4 releases.

**Figure 3-1 Comparison between BEA JRockit R26.3 and R26.4**




- Compressed references has greatly increased the performance on 64-bit platforms. See Figure 3-2 for a comparison of a 64-bit platform with and without compressed references compared to a 32-bit platform. For more information on compressed references, see `XXcompressedRefs` in the BEA JRockit Reference Manual.

**Figure 3-2 Comparison between 64-bit platform with and without compressed references and a 32-bit platform.**



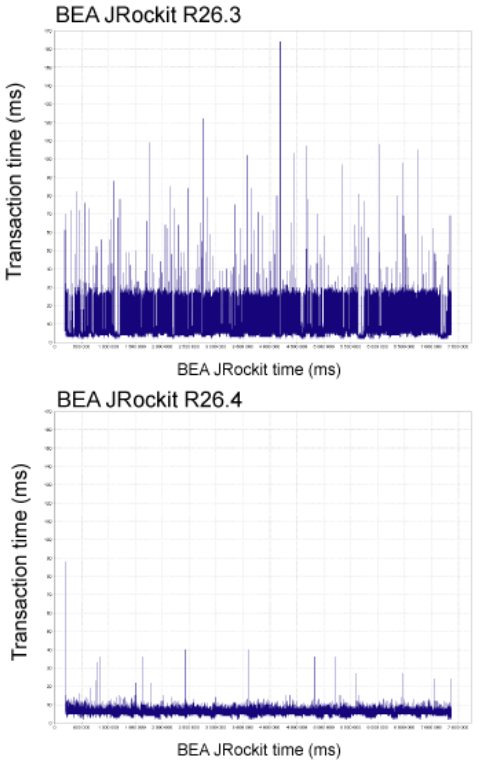

# Performance Improvements for Low Latency Applications

The performance of BEA JRockit has gone through major improvements for low latency applications by the following additions:

- Lock tuning on all supported platforms—Intel Itanium, Intel Xeon, AMD Opteron, Sun SPARC (including T1), and Fujitsu SPARC 64 V.
- Incremental handling of weak native handles have been moved to the concurrent phase, which lowers the pause times.

See Figure 3-3 for an illustration of how the performance have been improved with the above improvements.

**Figure 3-3 Performance improvements compared between BEA JRockit R26.3 and R26.4**




# Performance Improvements on SPARC

SPARC performance running BEA WebLogic Server has been improved by approximately 15% compared to BEA JRockit R26.3.

# Additional Tuning Possibilities

The following tuning options have been added in this release:

- `XXgcThreads`
- `XXoptThreads`
- `XXcompressedRefs`
- `XXlazyUnlocking`
- `XXthroughputCompaction`.

For complete information on how to use all tuning options in BEA JRockit, please see the BEA JRockit Reference Manual.

# New Features and Enhancements in BEA JRockit R26.3

- There have been performance enhancements to the Deterministic garbage collector:
  - More efficient compaction.
  - Improved handling of reference objects.

# New Features and Enhancements in BEA JRockit R26.2

- This is the first BEA JRockit JDK 1.4.2 release containing full support for the Memory Leak tool and the JRA.

# New Features and Enhancements in BEA JRockit R26.1

- BEA JRockit is now available for Solaris on Sparc. Please see BEA JRockit Supported Configurations for exact configurations.

  **Note:** The Sparc version of BEA JRockit is a 64-bit JVM only.

# New Features and Enhancements in BEA JRockit R26.0

- Significant improvements in code generation speed, especially for large methods, e.g. jsp:s.
- Deterministic garbage collector (requires a separate license). For more information, see the Memory Management Guide.

# Most Recent Changes

This has happened with BEA JRockit:

- Changes in the BEA JRockit R26.4 Release
- Changes in the BEA JRockit R26.3 Release
- Changes in the BEA JRockit R26.2 Release
- Changes in the BEA JRockit R26.0 Release

# Changes in the BEA JRockit R26.4 Release

The following CRs have been corrected for the BEA JRockit R26.4 release.

| Change Request ID | Description |
|---|---|
| CR279188 | When JRockit 1.4.2_10 R26.3 calculates the SUID for a class to be serialized it includes the synthetic bits in the calculation, which generates a SUID that differs from the Sun 1.4.2_10 and BEA JRockit 1.4.2_08 R24. This problem causes serialization to fail.<br>Synthetic attributes are included in a class when the class references a class pointer `ATestClass.class` or uses assertions.<br>This means that RMI communication between 1.4.2_10 R26.3 and Sun 1.4.2_10 can fail. RMI communication between BEA JRockit 1.4.2_10 R26.3 and BEA JRockit 1.4.2_08 R24.5 can also fail in the same way. Serialized classes, stored in a database, might also fail to be loaded.<br>**Note:** This has been fixed in R26.4. |
| CR258122, CR258159, CR264680 | The following new options have been added:<br>• `-XXgcTreads`<br>• `-XXlazyUnlocking`<br>• `-XXthroughputCompaction`<br>• `-XXinternalCompactionMultiplier`<br>• `-XXexternalCompactionMultiplier` |
| CR258934 | New arguments have been added to enable control of large pages (`-XXlargePages`) on Solaris. |
| CR267987 | Using `java.nio.channels.SelectionKey.OP_CONNECT` will make BEA JRockit block forever. This has now been fixed. |
| CR268133 | Previously `java.lang.reflect.Method.getParameterAnnotations()` returned the wrong result for methods that did not have annotations. This has now been fixed. |
| CR268439 | Previously when calling JVMTI functions from a non-attached thread caused BEA JRockit to crash. This has now been fixed. |
| CR269375 | For each old collection, the reason for a garbage collection is printed in the `verbose:memdbg`. |

| Change Request ID | Description |
|---|---|
| CR271551 | Previously buffers that had been allocated through `ByteBuffer.allocateDirect()` would not be released, even when discarded by the application, which caused C heap memory leaks. This has now been fixed. |
| CR272364 | The JVMTI function `GetThreadInfo` returned an error if passed NULL as the thread. It now treats NULL as the current thread in accordance with the specification. |
| CR274636 | Previously BEA JRockit 1.4.2 R26.2 and R26.3 threw a `java.lang.NullPointerException` when passing a null argument to `java.lang.String.getBytes(String charsetName)`. This was not an issue with BEA JRockit 1.4.2 R24.<br>This has now been fixed. |
| CR275108 | Previously, when using the `lookfor` parameter to the Ctrl-Break handler `print_object_summary` could crash BEA JRockit if certain kinds of references were found.<br>This has now been fixed. In addition, the output includes the kind of each reference. |
| CR275725 | The implementation of BEA JRockit's `System.nanoTime()` on Linux provided the time since BEA JRockit started. Now BEA JRockit uses `gettimeofday` instead, which makes it easier to use the time to compare different JVM instances. |
| CR276308 | The verbose framework in BEA JRockit is now stricter and possible ambiguous shortcuts have been removed, for example, the option `-Xverbose:compact` no longer works, instead you need to use `-Xverbose:compaction`. |
| CR276311 | BEA JRockit R26.3.0 could, under rare circumstances, crash when using `-Xgc:gencon`. All platforms were affected. This has now been fixed. |
| CR276460 | BEA JRockit no longer fails on RHEL4 when using `FileChannel.transferTo`. |
| CR276987 | In BEA JRockit R26 releases prior to R26.4, a breakpoint could be garbage collected if the garbage collection went off before the method was generated. This has now been fixed. |
| CR277702 | The `JraRecordingStarter.jar` is no longer shipped with BEA JRockit. Use `jrcmd` instead to start a JRA recording. |
| CR278411 | Previously, BEA JRockit incorrectly threw a `BufferUnderflowException` in `java.nio.DirectByteBuffer.put()` instead of `BufferOverflowException` when the buffer overflowed. This has now been fixed. |

| Change Request ID | Description |
|---|---|
| CR279712 | Previously, the Ctrl-Break handler `print_threads` could result in a memory leak. This was due to handles that were not released during the printing of the thread status.<br>This problem has been noted on Linux and Solaris platforms only and it has now been fixed. |
| CR280443 | Previously when you expanded types in the Type Graph in the BEA JRockit Memory Leak Tool, BEA JRockit hung and became unresponsive while consuming CPU resources. This issue has now been fixed. |

# Changes in the BEA JRockit R26.3 Release

The following CRs have been corrected for the BEA JRockit R26.3 release.

| Change Request ID | Description |
|---|---|
| CR265225 | The BEA JRockit JRE installer in Silent mode will now be correctly installed if the `USER_INSTALL_DIR` in the XML-file has been set. |
| CR264064 | The default stack size for Solaris on Sparc has doubled to 512 kB. |
| CR263376 | RHEL 4.0 QU1 on Itanium contains a critical kernel bug that was corrected in QU2. Therefore, BEA require that you run BEA JRockit with RHEL 4.0 QU2 (or later) on Itanium systems.<br>For IA32 and x64, there are no known issues that require an update to QU2, and you can stay on QU1. |
| CR262962 | Hyperthreading detection has changed slightly, which makes BEA JRockit better at detecting wether Hyperthreading is available and turned on or off. This has also caused the BEA JRockit property `jrockit.cpu.ia32.ht` to default to the value "`os`" (OS detection) instead of "`hw`" (Hardware detection). |
| CR262571 | Previously the `sizeof` parameter to the `Ctrl-Break` handler `print_object_summary` did not work as expected. This has been fixed. |
| CR262540 | Previously when running HP OpenView Java Diagnostics Profiling Agent could cause crashes with BEA JRockit R26.0.0. This issue has now been fixed. |
| CR262527 | JRockit no longer fails to handle bytecode where the control flow enters an exception handler without a thrown exception. |

| Change Request ID | Description |
|---|---|
| CR262448 | Multiple issues with `-Xdebug` in R24.5.0 have been fixed in this release. |
| CR260490 | The new version of `rtmon` (`librtm.so`) is now compatible with Montecito systems, even if the kernel is not patched with the Montecito perfmon-2 patch. This means that there are no more failing on calls to `RTMonRegisterThread()` or `RTMonStart()`, which earlier caused BEA JRockit create a core dump. |
| CR260241 | BEA JRockit sometimes crashed while stepping into an `ArrayIndexOfOutBounds-exception` in the debugger on x86. This has now been fixed |
| CR259231 | Previously the JVMTI function `GetObjectsWithTags` was broken. This has now been fixed. |
| CR258894 | When running BEA JRockit with `gcprio:pausetime` or `gcprio:deterministic`, it is now possible to set the pausetime target using JMAPI or JLMEXT. See the Javadocs for further information. |
| CR258395 | BEA JRockit never called the JVMTI function `Agent_OnUnload` when the JVM shut down. This has now been fixed. |
| CR258248 | You can now use the environment variable `JROCKIT_DUMP_PATH` to tell BEA JRockit to save crash dump information in a different location than the current working directory. The path specified must exist and be writable. |
| CR258200 | In BEA JRockit R26 releases prior to R26.3, BEA JRockit running could livelock due to a thread priority issue on Windows platforms.<br>This has now been fixed. |
| CR257905 | (Linux only) When BEA JRockit was started from a process that blocks signal (such as Sun's HotSpot JVM) it was not possible to send SIGQUIT to or use the *jrcmd* tool against BEA JRockit. This has now been fixed by unblocking signals that BEA JRockit listens to. |
| CR257799 | JRockit now allows more than one JVMTI agent to get the `can_tag_objects` capability. |
| CR257687 | In BEA JRockit R26 releases prior to R26.3, BEA JRockit could crash due to a bug in the implementation of `Class.getMethod()`. The "`this`" reference was not treated properly by the garbage collector if threads were stopped in bad locations.<br>This has now been fixed. |

| Change Request ID | Description |
|---|---|
| CR257039 | The new option `-XXdisableGCHeuristics` disables the dynamic garbage collector selection heuristics when `-Xgcprio` is used. The JMAPI for changing garbage collectors in runtime can still be used.<br><br>**Note:** If you disable the dynamic garbage collector selection heuristics, it will affect the behavior of the dynamic garbage collector and may lead to lower throughput or longer garbage collection pauses. |
| CR256867 | The initialization of the initial (main) thread on Linux now respects the `-Xss:<size>` option, and commits an area corresponding to the largest of this and the current system stack rlimit (see `man rlimit`). |
| CR255959 | A fix for Sun bug 5103041 has been added. |
| CR255271 | • The compaction for the deterministic garbage collector has been improved.<br>• The following options have been added:<br>– `-XXinitialPointerVectorSize:<nn>`<br>– `-XXmaxPooledPointerVectorSize:<nn>`<br>– `-XXpointerMatrixLinearSeekDistance:<nn>`<br>• The option `-XXcompactSetLimit` now also works with `-Xgcprio:deterministic`. |
| CR254297 | A race condition existed in BEA JRockit when the Memory Leak Server was stopped immediately after a client had connected, which caused sockets to be left in `CLOSE_WAIT` state. On Linux, this could eventually cause the BEA JRockit process to run out of file descriptors. This problem has now been fixed. |
| CR252610 | The problem regarding the handling of TrueType fonts for certain font files has now been fixed. |
| CR251838 | Previously a problem regarding the handling of TrueType fonts for certain font files caused a call to `java.awt.Font.getXXX()` methods that resulted in an `IllegalArgumentException` being thrown. This issue has now been fixed. (See, CR252610 for more information). |
| CR248132 | Explicit font support for Asianux, part of Red Flag Linux, has been added for BEA JRockit 1.4.2, keeping the support for Red Flag 4.1 font patch CR200703 for `LANG=zh_CN.GB2312` locale. The font support is based on the Red Hat Linux font support. |
| CR247393 | Linux and Solaris man pages have been removed. |

| Change Request ID | Description |
|---|---|
| CR247026 | In earlier BEA JRockit R26 versions, on Windows operating systems, BEA JRockit could sometimes expose a problem in the OS related to multimedia timers that caused the system time to be adjusted backwards.<br>This could cause the system time to jump back by about 1 minute. If this happened, you could turn off the use of multimedia timers with `-Djrockit.periodictask.usemmtimers=false`.<br>This problem has now been fixed. |
| CR189181 | Explicit font support for Asianux, part of Red Flag Linux, has been added for BEA JRockit 5.0. The font support is based on the Red Hat Linux font support. |
| CR179421 | BEA JRockit no longer misses to report some contended monitors to JVMPI. |

# Changes in the BEA JRockit R26.2 Release

The following CRs have been corrected for the BEA JRockit R26.2 release.

| Change Request ID | Description |
|---|---|
| CR256719 | The JVM experiences a slowdown with large number of threads doing reflective invocation of the same method concurrently. |
| CR239984 | In earlier releases of the BEA JRockit JDK 1.4.2, it was necessary to specify `-Xmanagement:class=com.JRockit.management.rmp.RmpSocketListener` on the command line to start up the Management Server. This can now be done by simply specifying `-Xmanagement` or by setting the management server port with `jrockit.managementserver.port`. |
| CR249272 | The property `jrockit.managementserver.usejmx` has been added for BEA JRockit JDK 1.5. Setting this property to false will make BEA JRockit use the RMP-protocol instead of the default management protocol (JMX) on a BEA JRockit JDK 1.5. |
| CR250218 | When trying to print heap references, non-fatal JVMPI error messages were displayed. This has now been fixed. |
| CR250712 | A race could cause BEA JRockit to crash during a JRA recording if a thread completed at the wrong moment. This race has been fixed. |

| Change Request ID | Description |
|---|---|
| CR252315 | The compaction heuristics now ignore exceptional compactions when adjusting the compact ratio and pointerset limit. |
| CR252348 | The option `-Xverbose:cpuinfo` is now available on IA64. |
| CR252567 | The default stack size on X64 platforms has doubled from previous releases. |
| CR253588 | The amount of free space is now calculated correctly when BEA JRockit calculates the maximum allowed nursery size during automatic nursery resizing. |
| CR253952 | Several JVMPI problems have been fixed. |
| CR254354 | For some garbage collectors, the minimum block size value, set by `-XXminBlockSize`, was also (incorrectly) used as thread-local area size. With this fix, increasing the `-XXminBlockSize` value will no longer affect the thread-local area size.<br>If you have been using `-XXminBlockSize` to adjust the thread-local area size, you now must also set `-XXlargeObjectLimit` and `-XXtlaSize` to the same value as you set `-XXminBlockSize`, as described in the BEA JRockit Reference Manual. |
| CR257184, CR257379 | The option `-Xpausetarget` didn't always work when running `-Xgcprio:deterministic`. This has now been fixed. |
| CR257540 | The JNI function `GetDirectBufferAddress` has now been changed to work with all direct `java.nio.Buffers`. Previously it only worked for direct `java.nio.ByteBuffers`. |
| CR257840 | To make it easier to diagnose JRockit crashes, the option `-XXdumpfullstate` has been made default. This means that if BEA JRockit crashes a lot more information is saved to disk than was the case previously. To get the old behavior use `-XXdumpsize:normal`. |
| CR258002 | Loading extra data from zip-file entries now work. |
| N/A | Java Web Start and the Java Plug-in were included in the previous 1.4.2 BEA JRockit version, i.e. BEA JRockit 1.4.2_08 R24.5.0. These features are dropped for BEA JRockit R26 on 1.4.2. |

# Changes in the BEA JRockit R26.0 Release

The following CRs have been corrected for the BEA JRockit R26.0 release.

| Change Request ID | Description |
|---|---|
| CR211951 | In previous versions of BEA JRockit, the JVM Process Load was capped to 100/number of CPUs on multi CPU Windows machines. This has now been fixed. |
| CR213687, CR213685 | The non-supported option `-XXprintStackOverflow` has been added. This option produces a full stackdump when the `StackOverFlowError` is thrown. |
| CR218035, CR230226 | In the previous release, BEA JRockit was known to hang or crash on 2.6 kernels on Itanium, due to a bug in the Linux 2.6.11 (and previous) kernel. The bug has now been fixed in kernel 2.6.12 by the Linux vendors.<br>**Note:** To run this release of BEA JRockit, you need to use SLES 9 SP2 or RHEL4 U1 (or later). |
| CR219610 | The default freelist cache size is 10% of the current heap size (with a minimum of 3 MB). |
| CR225145 | Changes to java.vendor* system properties. The correct values are:<br>• `java.vendor = "BEA Systems, Inc."`<br>• `java.vendor.url = "http://www.bea.com/"`<br>• `java.vendor.url.bug = "http://support.bea.com"` |
| CR226460 | The experimental code cache feature has been removed due to stability issues. |
| CR228592 | Using the Memory Leak Tool could in some instances make JRockit freeze or crash. This has now been fixed. |
| CR228822 | When running BEA JRockit on a single CPU machine, the code optimizer was in some cases too intrusive (true for BEA JRockit 5.0 R25.2.0). The problem was sometimes noticed at the first start of the WebLogic console. This problem has now been fixed.<br>In the previous release of BEA JRockit, this problem was worked around by setting the flag `-Djrockit.codegen.optpriority=1`; if you are using this flag, remove it when updating to this release. |
| CR229981 | Improved behavior of internal locks that can lead to better performance during heavy loads. |
| CR230236 | The `-Xmanagement` option resulted in an overhead even though the BEA JRockit Management Console was not connected. This has now been fixed. |
| CR232847 | Improved floating point performance. |

| Change Request ID | Description |
|---|---|
| CR235100 | The `java.vm.version` for the previous release was:<br>`dra-45238-20050523-2021-win-ia32`<br>The `java.vm.version` for this release is:<br>`R26.0.0-188-52875-1.5.0_04-20051110-0917-win-ia32`<br>The `java.vm.info` for the previous release was:<br>`R25.2.0-28`<br>The `java.vm.info` for this release is:<br>`<empty>` |
| CR235101 | Previously, BEA JRockit calculated `MemoryMXBean.getNonHeapMemoryUsage().used` as the process virtual bytes minus the heap size. Now `MemoryMXBean.getNonHeapMemoryUsage().used` is calculated as the process rw memory minus the heap size. |
| CR235105 | The heap occupancy trigger heuristics have been corrected. |
| CR235107, CR236922 | When running JRockit with a concurrent garbage collector, the garbage collector starts before the heap is completely full to be able to finish the garbage collection before running out of heap memory.<br>tries to determine when a garbage collection needs to be triggered through heuristics, but in certain situations it might be beneficial to set this trigger by hand and to a fixed value. Use the following argument `-XXgcTrigger=<int>`, where `int` is an integer that takes values between 0 and 100. The value specifies the amount of free heap, measured in percentage, that should be available for the argument to trigger. |
| CR235682 | Previously selecting a generational, concurrent mark, concurrent sweep strategy resulted in a non generational, parallel mark, parallel sweep strategy being chosen. This has now been fixed. |
| CR236723 | A warning appears at start-up if you are running with a “suspicious” thread system. |
| CR237093 | The time for the reference update phase was measured incorrectly when running the parallel garbage collector. This made the statistics that the (compaction) heuristics are based on incorrect.<br>The pause target for compaction increased too fast when running static garbage collections. In this release, it can be increased with at most 50% for each garbage collection. The initial value has been set to 100 ms (this may be further tuned). |
| CR238220 | CPU load and CPU description is now returned as CompositeData in `JRockitConsoleMXBean`. |

| Change Request ID | Description |
|---|---|
| CR239499 | The Memory Leak Detector did not work properly when the management server was started by the Ctrl-Break handler (using `ctrlhandler.act`) as opposed to using the `-Xmanagement` startup option. This has now been fixed and the Memory Leak Detector works as expected, regardless of how the management server has been started. |
| CR239968 | In previous versions, the maximum stack trace depth value, in JRA recordings, was always 16 frames. In this version it is possible to set this value by adding the "tracedepth" option to `-XXjra` and the "jrarecording" ctrlbreak handler.<br>The default value is still 16 frames. |
| CR240355 | In this release it is possible to change verbosity for the "memory," "memdbg," and "gcpause" subsystems using the ctrlbreak handler. |
| CR240359 | The syntax for the "verbosity" ctrlbreak handler has changed. The previous argument "args" has been changed to "set."<br>Run the ctrlbreak handler "help verbosity" for more details. |
| CR240510 | • The information from `jrockit.verboserefs` has been improved and now includes more details regarding garbage collection.<br>• Support for verbose information in `ctrlbreakhandler/jrcmd` has been added. |
| CR241377 | The default stack size on Solaris/Sparc is 256k. |
| CR241546 | This release of BEA JRockit does not ship with Java Web Start or Java Plug-in. Some earlier releases did and the installers and uninstallers of those versions do not behave properly when this release is installed. To avoid problems with the installation, do one of the following before installing:<br>1. Uninstall all earlier BEA JRockit JRE releases **before** installing this release. Java Web Start and Java Plug-in will not be available after this process.<br>2. Install all earlier BEA JRockit JRE releases that are needed **before** installing this release. Java Web Start and Java Plug-in will be available if included in any of the earlier releases.<br>**Note:** Do **not** install or uninstall an earlier release of BEA JRockit JRE while this release is installed. Doing so may corrupt the state of Java Web Start and Java Plug-in. |

| Change Request ID | Description |
|---|---|
| CR241638 | The following compaction tuning flags have been added:<br>`-XXinternalCompactRatio`<br>Sets the number of heap parts to compact during internal compaction. Default is dynamic or 8 when running `-Xgcprio:throughput`.<br>`-XXexternalCompactRatio`<br>Sets the number of heap parts to compact during external compaction (aka "evacuation"). Default is dynamic or 8 when running `-Xgcprio:throughput`.<br>`-XXheapParts`<br>Sets the number of heap parts. Default is 128.<br>Furthermore, the system property `jrockit.gc.usematrix` has been turned into an `-XX` option.<br>`-XXusePointerMatrix`<br>Indicates that the pointer matrix should be used instead of the pointerset. The pointer matrix is default when running `-Xgcprio:deterministic` or `-Xgcprio:pausetime`. |
| CR241665 | In the management API, the functions `getMAC` and `getMTU` are supported on Windows. On Unix systems these functions return an empty string or zero. |
| CR242307 | To get fixes for potential security vulnerabilities, this release has upgraded zlib from zlib-1.2.1 to zlib-1.2.3. |
| CR242944 | The command `jrockit.oomdiagnostics.filename` specifies where to write out of memory diagnostics (if this is enabled through `jrockit.oomdiagnostics`). If diagnostics are enabled and no file is specified, the output ends up where the `-Xverbose` information ends up (typically `stderr`). |
| CR244403, CR238634 | Traversing superclasses of interfaces in `find_method` have been removed. These classes returned methods that were not declared in the interface or its super interfaces, for example, `Object.*`. The supermost class of an interface is always `Object`. |
| CR245707 | JVMDI is not supported in the BEA JRockit R26.0 release (nor in the 25.0 or 1.4.2 builds); however, JDWP and JDI are supported. This means that remote debugging tools will still work as in previous releases. |

| Change Request ID | Description |
|---|---|
| CR245732 | Previously, retrieving JMAPI stack traces could deadlock in certain cases for traces that included overridden hashCode methods that had been taking locks. This has now been fixed. |
| CR255294 | Previously, when calling `java.io.File.getCanonicalFile()` on a path where `java.io.FilePermission` was not granted, the call did not fail as expected. This has now been fixed and the appropriate exception is thrown. |

# Known Issues

The following issues are known in BEA JRockit R26:

| Issue | Description |
|---|---|
| BEA JRockit is crashing due to a signal handling conflict.<br>*For Linux users only*. | If you are using BEA JRockit in conjunction with a native library that relies on OS signals you may experience crashes due to a signal handling conflict between BEA JRockit and the native library.<br>**Workaround**:<br>Set the environment variable `LD_PRELOAD` as follows:<br>`export LD_PRELOAD=$JROCKIT_HOME/jre/lib/i386/libjsig.so`<br>BEA Engineering found this conflict using IBM's MQSeries native drivers, and it may be present in other libraries that rely on native code.<br>For more information, see:<br>http://java.sun.com/j2se/1.5.0/docs/guide/vm/signal-chaining.html |
| CR361457 | Connecting the Memory Leak Detector to a running JRockit JVM (version R26.3 through R27.5) built for Linux IA-32 might cause the JVM to crash if the JVM process uses more than about 1020 file descriptors at the time. This might only happen if the file descriptor limit has been set higher than 1024 (typically by using the `ulimit` command).<br>**Workaround**:<br>Currently you can start the Memory Leak Detector Server at JVM startup, when few file descriptors are in use. To do this, add `-Djrockit.memleak.port=12345` early in the JVM command line.<br>Now, using JRockit Mission Control, create a custom connection in the JRockit Browser with a Custom JMX service URL of `service:jmx:mlp://`*`localhost`*`:`*`12345`*. (Replace *`localhost`* and the port *`12345`* as needed). Using this connection, you can connect the Memory Leak Detector in JRockit Mission Control to this JVM once (without restarting the JVM).<br>Note that using many file descriptors might be an indication of a resource leak in the Java application. Make sure to always close opened files and sockets. You should not rely on the Garbage Collection and the object finalization to free a non-Java resource such as a file descriptor. See Too Many Open Files at:<br>`https://support.bea.com/application_content/product_portlets/support_patterns/wls/TooManyOpenFilesPattern.html`<br>for more information. |

| Issue | Description |
|---|---|
| CR311515 | The new explicit font support for Asianux, part of Red Flag Linux, available since BEA JRockit R26.3.0 depends on the existence and contents of the file /etc/asianux-release to correctly identify a Asianux compatible Linux distribution. If this file is not present the JRockit JDK or JRE will revert to look for `/etc/redhat-release` and instead identify a Red Hat Linux compatible distribution with Red Hat compatible font support.<br><br>The font support for Asianux is based on the font support for Red Hat Linux, with an additional patch for Chinese LANG=zh_CN.GB2312 locale for Red Flag Advanced Server 4.1 (and higher) in JRockit 1.4.2. Thus, if you are using Chinese zh_CN locale with JRockit 1.4.2 R26.3.0 and later on Red Flag Advanced Server 4.1 (and higher), that does not contain the file `/etc/asianux-release`, then BEA JRockit will load the Red Hat-specific `font.properties.zh_CN.Redhat` file instead of the expected Asianux-specific `font.properties.zh_CN.Asianux` file. This might cause the JVM to fail and result in unexpected behavior when it tries to load specific fonts with incorrect paths.<br><br>On Red Flag Advanced Server 4.1 and Red Flag Advanced Server 4.1 (SP1) the file `/etc/asianux-release` file might not be installed by default and if that is the case you can apply either one of the following workarounds.<br><br>- Install the optional RPM package asianux-release, if available. This is the preferred solution. On Red Flag Advanced Server 4.1 (SP1) the optional package asianux-release may conflict with the possibly already installed optional package redflag-release, you may choose to either uninstall the redflag-release package before installing the asianux-release package, or to force install the asianux-release package in parallel with the redflag-release package.<br><br>- Manually replace the contents of `<jdk>/jre/lib/font.properties.zh_CN.Redhat` with the contents of `<jdk>/jre/lib/font.properties.zh_CN.Asianux`. |
| CR307903 | If a thread is interrupted for garbage collection while it is in the process of copying an array, then the garbage collection may result in very long pauses. If you get occasional long pause times, this may be the problem. Note that this issue has been fixed in BEA JRockit R27.2. |
| CR307902<br><br>*For Linux users only.* | If you explicitly request to use the Motif AWT instead of the default X11 AWT on Linux/IA64 and run a Linux version with a GLIBC version older than `glibc-2.3.4`, this operation might fail with an `UnsatisfiedLinkError` since the file `<jre>/lib/ia64/motif21/libmawt.so` requires linkage to `GLIBC >= 2.3.4`.<br><br>See the see the Supported Configurations document for supported Linux versions. |
| CR300097 | There is a known bug in RedFlag 5.0 with the `wait()` call that might cause undetermined behavior for RedFlag customers using this OS version |

| Issue | Description |
|---|---|
| CR286338 | In BEA JRockit 26.4, JRA Recordings always report that the heap usage is zero after a garbage collection with a single-spaced garbage collector. The `-Xverbose:memory` and `-Xverbose:memdbg` printouts report the correct value for the heap usage after garbage collection. |
| CR284602 | The jrcmd tool might fail to find all BEA JRockit processes on a machine if there are any Java processes by other JVM vendors running on the same machine. Contact BEA JRockit Support for a patch that solves this problem. |
| CR283915 | Long thread sleeps issued by `Thread.sleep(...)` and `Object.wait(...)` can end too early if the sleeps are longer than `0x3FFFFFFF` milliseconds (approximately 12.4 days).<br>All platforms are affected. |
| CR283787 | Upgrading from BEA WebLogic Platform 8.1 SP5 to 8.1 SP6 and running it on BEA JRockit R26 SP3 can result in a performance regression of up to 10%.<br>**Workaround:**<br>To avoid this regression, you can improve the performance of memory intensive applications by setting the command-line options `-XXtlaSize:`<default 2kB> and `-XXlargeObjectLimit:`<default 2kB>. |
| CR280443 | When you expand types in the Type Graph in the BEA JRockit Memory Leak Tool, BEA JRockit can hang and become unresponsive while consuming CPU resources. This issue was introduced as a regression in BEA JRockit R26.2.<br>**Note:** This has been fixed in R26.4. |
| CR279998 | Objects that are allocated with reflection, for example, with `java.lang.Class.newInstance()`, do not show up in the allocation stacktraces in the Memory Leak Detector Tool. |
| CR279584 | The synchronization code in `java.util.Random.next()` has been optimized for the case where there is no contention, i.e. the `java.util.Random` object is used by only one thread. A drawback of this optimization is worse performance if the object is used heavily in several concurrent threads. This typically happens if the convenience method `java.lang.Math.random()` is used.<br>To avoid this, create a new `java.util.Random` object instead of calling `java.lang.Math.random()`. |

| Issue | Description |
|---|---|
| CR278796 | Null-pointer exception bypasses first catch block and is caught in the next.<br>A catch block immediately following "`return x.y;`" where the variable `x` points to null, will not catch the null-pointer exception. The exception will be caught in the next catch block.<br>`public class Test {`<br>`  int y = 0;`<br>`  public static int foo() {`<br>`  try {`<br>`  Test x = null;`<br>`  return x.y;`<br>`  } catch (Exception e) {`<br>`  System.out.println("It works!");`<br>`  }`<br>`  return 0;`<br>`  }`<br>`  public static void main(String[] args) {`<br>`  try {`<br>`  foo();`<br>`  } catch(Exception ex) {`<br>`  System.out.println("Failure!");`<br>`  }`<br>`  }`<br>`}` |
| CR276311 | BEA JRockit R26.3.0 can, under rare circumstances, crash when using `-Xgc:gencon`. All platforms are affected.<br>**Workaround:**<br>Use another garbage collector or upgrade to BEA JRockit R26.4.0. |
| CR275524 | On some Linux versions, the library functions `exit()` and `_exit()` do not always terminate the calling process. This causes BEA JRockit to hang during shut down on SLES 8.<br>**Workaround**:<br>Use a later SLES version. |

| Issue | Description |
|---|---|
| CR274636 | BEA JRockit 1.4.2 R26.2 and R26.3 throw a `java.lang.NullPointerException` when passing a null argument to `java.lang.String.getBytes(String charsetName)`, while BEA JRockit 1.4.2 R24 did not. The behavior is unspecified but the change might cause problems when running TIBCO with XML messaging on BEA JRockit 1.4.2 R26.2 or R26.3.<br><br>**Note:** This has been fixed in R26.4. |
| CR272699 | Occasionally, when using a JVMTI Java debugger, breakpoints are not hit. This issue can arise with any R26 version of the product.<br><br>**Workaround:**<br><br>Start BEA JRockit with the option `-XXnoCodeGC`.<br><br>**Note:** `-XXnoCodeGC` is only intended for troubleshooting and is not recommended nor supported for production use. |
| CR271551 | Buffers that have been allocated through `ByteBuffer.allocateDirect()` would not be released, even when discarded by the application, which causes C heap memory leaks.<br><br>**Note:** This has been fixed in R26.4. |
| CR269115 | BEA JRockit crashes when optimizing method. This issue can be identified by the following stack trace:<br><br>`at renameVar+36()@...`<br>`at irCompactVars+240()@...`<br>`at ssaConvertTo+2356()@...`<br>`...`<br><br>This crash has only been noted on the Solaris on Sparc platform but might be a problem on other platforms as well.<br><br>**Workaround**:<br><br>Use `optfile` and remove the particular method that causes the crash. |
| CR268746 | On older Linux distributions that run LinuxThreads instead of NPTL, BEA JRockit can sometimes hang when it is shutting down. |
| CR268439 | Calling JVMTI functions from a non-attached thread will cause BEA JRockit to crash.<br><br>**Note:** This has been fixed in R26.4. |
| CR268423 | BEA JRockit releases previous to R26.3.0 *do not* contain the fix for the Australian Daylight Savings Time change for 2006. Please contact **BEA Support** for a patch |

| Issue | Description |
|---|---|
| CR267987 | Using `java.nio.channels.SelectionKey.OP_CONNECT` will make BEA JRockit block forever.<br><br>**Note:** This has been fixed in BEA JRockit R26.4. |
| CR266871 | BEA JRockit R26.0.0 on Linux IA32 can experience problems setting up memory for object allocation. It will manifest itself through this printout (and then exit BEA JRockit):<br>`[JRockit] ERROR: Fatal error in JRockit during memory setup phase. Try to reduce the heap size using -Xmx:<size>m, i.e. "-Xmx:16m". Could not create the Java virtual machine.`<br>**Workaround**:<br>Try different `-Xmx` values to find a heap size that is setup correct.<br><br>**Note:** This known issue is valid for R26.0.0. The problem is fixed in releases R26.1.0 and later. |
| CR266870 | IA64 RedFlag 4.1 creates broken core files when programs crash. This makes it impossible for BEA JRockit engineers to resolve customer issues on RF41/IA64. |
| CR266667 | Slow startup because of a hang in `java.net.PlainSocketImpl.initProto()`, which typically is called when creating the first Socket or ServerSocket.<br>In BEA JRockit 5.0 R26 the network stack is configured so that IPv6 is used in preference to IPv4 when it is present.<br>During initialization of the network stack, the network code connects a socket to its own loopback interface to set up some data structures. Blocking this connection (e.g. with a firewall) will cause the initialization code to wait for a socket timeout, after which the system falls back on using IPv4.<br>**Workaround**:<br>Either set `-Djava.net.preferIPv4Stack=true`, which forces Java to use IPv4 instead, or you disable IPv6 entirely in the system. The proper fix is to allow IPv6 traffic from localhost to localhost.<br>For more information, see the Sun documentation:<br>http://java.sun.com/j2se/1.4.2/docs/guide/net/ipv6_guide/#ipv6-networking |
| CR265793 | With JRockit 1.4.2 R26, `java.lang.reflect.Array.set(Object array, int index, Object value)` always throws `NullPointerException` when value is null without checking if array is of primitive type.<br>A patch addressing this issue is available through BEA support. |

| Issue | Description |
|---|---|
| CR265227 | The BEA JRockit in Silent mode will not be correctly installed if the `USER_INSTALL_DIR` in the XML-file has been set to anything other than the default installation path.<br>During the installation, the registry settings will not be set correctly, causing the `.jar` file association to fail. The files `Java.exe` and `Javaw.exe` will not be copied to `%SystemRoot%\System32` either.<br>**Note:** This problem has been fixed in R26.3. |
| CR264913, CR244553 | A bug in the Linux operating system on x64 will cause BEA JRockit to crash if it is invoked from the `pthread_once` system call.<br>**Workaround**:<br>Install RHEL 4.0 QU3 or SUSE 9.0 SP3. |
| CR262540 | An issue running HP OpenView Java Diagnostics Profiling Agent may cause crashes with BEA JRockit R26.0.0.<br>**Note:** This issue has been fixed in R26.3.0. |
| CR262157 | In rare cases BEA JRockit can print incorrect information about locks in the stack dump, given by the Ctrl-Break handler or by the Management Console/MAPI.<br>When a lock is taken by a method that has been optimized in a certain way (inlining), this lock can be printed as being taken, not only on the correct frame, but also as being taken on one or several nearby frames. This does not affect how BEA JRockit treats the lock when executing, only the stack dump itself. |
| CR256312 | On Windows x64 and Itanium, when using the BEA JRockit JRE console mode installer to remove a previously installed BEA JRockit, the uninstall will be interrupted and the following message is displayed:<br>`A fatal error has occurred. This application will terminate.`<br>The uninstall information is now removed, but most of the files and registry settings are still left on the machine.<br>**Workaround to completely uninstall the BEA JRockit JRE**:<br>Run the installer in graphical mode and click **Remove previous and reinstall** when prompted; the BEA JRockit JRE is once again installed. To completely uninstall the BEA JRockit JRE (including its files and registry settings), use the graphical uninstall procedure. |

| Issue | Description |
|---|---|
| CR252610 | A problem regarding the handling of TrueType fonts for certain font files may cause a call to `java.awt.Font.getXXX()` methods that results in an `IllegalArgumentException` being thrown.<br><br>This problem has been reported to Sun as a problem found in Sun JDK 5.0 Update 4 in bug #6349101 (http://bugs.sun.com/bugdatabase/view_bug.do?bug_id=6349101). This problem can theoretically occur on all platforms but has been observed on RedFlag Linux Advanced Server 5.0 when the RPM package `ttfonts-zh_TW-5.0-2AX.noarch.rpm` is installed and the user is requesting a font with Chinese locale.<br><br>**Workaround**:<br><br>Uninstall the package `ttfonts-zh_TW-5.0-2AX`. This solves the problem but will at the same time remove those fonts from the system, which may cause other problems when trying to display Chinese text. If you try this, you should try to replace the use of the removed font with some other available font on the system.<br><br>**Note:** The problem has been fixed in release R26.3.0, but since that release does not include an Itanium version, it is still a known issue for the R26.0.0 Itanium version. |
| CR251457 | If `ulimit -v` is used on Linux to limit the virtual memory usage, BEA JRockit may crash if the limit is set too low. The x64 version of BEA JRockit requires a much larger setting because it reserves addresses for compiled code at startup. The BEA recommendation is to not use the `ulimit -v` setting at all. |
| CR251452 | BEA JRockit needs enough virtual memory (address space) to be able to run properly. Note that a high value on allowed virtual memory does not imply high memory consumption. On Linux, the amount of available virtual memory can be changed, typically by the command `ulimit -v <amount>`.<br><br>The default is an unlimited address space, which is the recommended.Do not change this default, since it risks having BEA JRockit running out of address space, which in turn causes BEA JRockit to terminate immediately.<br><br>BEA JRockit requires at least the following amount of virtual memory to even start:<br><br>• On IA64: 88 MB<br>• On IA32: 61 MB<br>• On x64: 1100 MB<br><br>**Note:** If you start BEA JRockit with less virtual memory than twice the above values, you'll be given a warning. This is not recommended, and is likely to cause problems when running an actual Java application. |

| Issue | Description |
|---|---|
| CR249667 | Some applications may experience problems with the automatic nursery sizing heuristics when running in `-Xgcprio:throughput` (default) and `-Xgcprio:pausetime` mode. This causes too frequently triggered garbage collections.<br>**Workaround**:<br>Set the nursery size manually using `-Xns:<size>` or to select a static garbage collector.<br>The automatic heap resizing heuristics are not optimal for all applications. If your application has problems due to frequent garbage collections and the heap hasn't been expanded to the maximum heap size, you can increase the initial heap size (`-Xms:<size>`) to improve the performance. |
| CR248565 | When running BEA JRockit as an embedded service on Windows (for example, Jakarta Tomcat service wrapper), the directory `<path-to-jdk>/jre/bin` must be added to the PATH variable. |
| CR248551 | When installing the 32-bit JRE on a Windows x64, the `java.exe` and `javaw.exe` files will not be copied to the system32 folder; they will be placed in the syswow64 folder instead.<br>This is expected behavior for 32-bit applications running on Windows x64. |
| CR247613 | The "jrcmd" utility shipped in the Windows ia32 BEA JRockit package does not work on Windows x64. Instead, use the "jrcmd" utility shipped with the Windows x64 package of BEA JRockit. |
| CR246634 | Thread priorities are supported on the Windows platforms only. |
| CR246224, CR260004 | When doing a JRA recording on Windows XP x64, the JRA recording incorrectly displays that it has been done on Windows 2003 Server. The current implementation of BEA JRockit cannot separate between the two different operating system's version information.<br>**Note:** This known issue is valid for R26.0 and R26.1. |

| Issue | Description |
|---|---|
| CR245914 | Java applications (such as Eclipse) may hang on Linux distributions which use the "gamin" File Alteration Monitor implementation, for example RHEL4. This is due to a bug in gamin's handling of signals.<br>**Workaround**:<br>Use "signal chaining" by loading the libjsig.so library. Do this by executing<br>`> export LD_PRELOAD=$JDK_HOME/jre/lib/i386/libjsig.so`<br>before starting BEA JRockit.<br>For more information on signal chaining see:<br>http://java.sun.com/j2se/1.5.0/docs/guide/vm/signal-chaining.html |
| CR244773<br>CR250025 | BEA JRockit's nursery pool can be invalidated when the garbage collection strategy changes. According to the Java 2 Platform SE API documentation, a pool that has been invalidated may return null. Programmers must therefore assume that `MemoryPoolMXBean#getUsage()` and `MemoryPoolMXBean#getPeakUsage()` may return null at any time.<br>The MemoryMonitor demo and the VerboseGC demo can throw `NullPointerExceptions`, since these attributes are not checked for nulls. |
| CR243996 | The VerboseGC demo, located at `\demo\management\VerboseGC\VerboseGC.jar`, can throw a `NullPointerExcecption` when used with a BEA JRockit that has a nursery.<br>According to the API specification for the Java 2 Platform Standard Edition 5.0 the method `getUsage()` of the `MemoryPoolMxBean` can return null if a memory pool is not valid, which can be the case when you run BEA JRockit with a nursery.<br>This validity check is missing in the demo and is the cause of the `NullPointerException`. |

| Issue | Description |
|---|---|
| CR242655 | In Windows, faulting code can be caught by Structured Exception Handling (SEH). The Microsoft C compiler allows a special construct, see below:<br>`__try {`<br>`  // do something that can fail`<br>`} __except (filterException()) {`<br>`  // handle the fault`<br>`}`<br>This sets up an SEH handler, which would get called if the code in the `__try` block fails (for example, a read/write to an illegal address).<br>However, on 64-bit Windows (IA64 and x64), BEA JRockit uses a new exception handling feature known as vectored exception handlers. The vectored exception handlers will be called before any SEH handler gets called. If the BEA JRockit vectored exception handler detects a fault in native code, it will make BEA JRockit produce a crash dump.<br>The effect of this is that you cannot use SEH on 64-bit Windows in native code that gets called by BEA JRockit. Either install a vectored exception handler yourself and add it first in the chain, or test the memory before trying to read/write to it with `IsBadReadPtr()`. |
| CR232872 | On RHEL3u6 and earlier, as well as RHEL4u2 and earlier, `fork()ing` new processes can sometimes fail. This can make in turn make `Runtime.exec()` fail. This has been fixed in RHEL3u7 and RHEL4u3.<br>The Red Hat Issue Tracker case number for this is 77560. This isn't available in Red Hat's public bugzilla. |
| CR210743 | If you are running SLES 8.0, RFAS 4.1, RHEL 3.0 QU4 or older, you might run into serious IO problems.<br>**Workaround (for RHEL)**:<br>Install version QU5 or later. |
| CR128962 | IPv6 support for Windows is included as an unsupported feature in this release. |