



Java 7 Launch and Apache Lucene / Solr:
**Crashes and File Corruption due
to Hotspot Bugs**

<http://s.apache.org/Java7LaunchBugBlog>

Uwe Schindler

SD DataSolutions GmbH, uschindler@sd-datasolutions.de

Presented by

lucid
IMAGINATION



Issues found

- Jenkins reveals **SIGSEGV bug in Porter stemmer** (found when number of iterations were raised) [[LUCENE-3335](#)]
- New Lucene 3.4 faceting test sometimes **produces corrupt indexes** [[LUCENE-3346](#)]
- Small issue in **ICU** tests [[LUCENE-3344](#), ICU bug [#8734](#)]
- Test of **WordDelimiterFilter** fails [simple fix committed]
- Lot's of Solr tests suddenly fail [[SOLR-2673](#)]



WARNING !!!



- **Also Java 6 affected!**
(some time after the only stable version 1.6.0_18)
- Optimizations disabled by default, so:

Don't use `-XX:+AggressiveOpts`
if you want your loops behave correctly!

Chronology: Friday, July 29, 2011

29 July 2011, 12:58

[« previous](#) | [next »](#)

Java 7 paralyses Lucene and Solr

The hotspot compiler in the recently [released](#) Java 7 has a defective optimiser that can cause flawed loops, according to a [warning](#) published by the Apache Software Foundation. As a result, the Java Virtual Machine can crash, and calculations can produce incorrect results.



A number of Apache projects are affected, including every published version of [Lucene](#) and [Solr](#). The Apache developers say that the indexing of documents on Solr causes Java to crash. Loops in Lucene can also be incorrectly compiled, thereby corrupting the indexes. In particular, the trunk version of Lucene with the [pulsing codec](#) is affected.

The bugs were discovered only five days before Java 7 was published; Oracle says it will correct them in the second service release of Java 7 [at the latest](#); the first update to Java 7 was reserved solely for security fixes, but the issue may prompt Oracle to change that plan. Until then though, users of Lucene and Solr should refrain from using the new version of Java or at least use the JVM option `-XX:-UseLoopPredicate` to disable the optimisation and prevent the index from being damaged.

The Apache developers say that users of Java 6 could also be affected. However, the flaws only occur in Java 6 when the JVM is used with the options `-XX:+OptimizeStringConcat` or `-XX:+AggressiveOpts` which activate normally disabled Hotspot optimisations.

Oracle has registered the flaws under [7070134](#), [7044738](#) and [7068051](#). The first one causes JVM to crash when Martin Porter's [stemmer algorithm](#) is used, which traces English words back to their stems. This flaw currently is of "low priority" while the others are "medium".

(djwm)

2011


THE H
O P E N The H open source security In association with

Last 7 days News Archive Features Forums Newsletter RSS

29 July 2011, 12:58 << previous | next >>

Java 7 paralyses Lucene and Solr

The hotspot compiler in the recently [released](#) Java 7 has a defective optimiser that can cause flawed loops, according to a [warning](#) published by the Apache Software Foundation. As a result, the Java Virtual Machine can crash, and calculations can produce incorrect results.



A number of Apache projects are affected, including every published version of [Lucene](#) and [Solr](#). The Apache developers say that the indexing of documents on Solr causes Java to crash. Loops in Lucene can also be incorrectly compiled, thereby corrupting the indexes. In particular, the trunk version of Lucene with the [pulsing codec](#) is affected.

The bugs were discovered only five days before Java 7 was published; Oracle says it will correct them in the second service release of Java 7 [at the latest](#); the first update to Java 7 was reserved solely for security fixes, but the issue may prompt Oracle to change that plan. Until then though, users of Lucene and Solr should refrain from using the new version of Java or at least use the JVM option `-XX:-UseLoopPredicate` to disable the optimisation and prevent the index from being damaged.

The Apache developers say that users of Java 6 could also be affected. However, the flaws only occur in Java 6 when the JVM is used with the options `-XX:+OptimizeStringConcat` or `-XX:+AggressiveOpts` which activate normally disabled Hotspot optimisations.

Oracle has registered the flaws under [7070134](#), [7044738](#) and [7068051](#). The first one causes JVM to crash when Martin Porter's [stemmer algorithm](#) is used, which traces English words back to their stems. This flaw currently is of "low priority" while the others are "medium".

(djwm)

jaxenter

NEWS VIDEOS BOOKS EVENTS JAVA TECH JOURNAL

HOME NEWS JAVA EDITORS PICK ECLIPSE ANDROID ARCHITECTURE CLOUD

! Find out what's new in JBoss AS7 and OpenShift, in the new issue of Java Tech Journal!

July 29, 2011 RELATED NEWS BOOKS VIDEOS EVENTS

Apache Code Affected by Java 7

Java 7 Could Cause Bugs in Some Apache Projects

f Share 4 t Tweet 5 +1 0 in Share 2 Comment Email Share

Uwe Schindler has posted that the just-released Java 7 contains hotspot compiler optimisations, which miscompile some loops, and this can **affect the code of "several" Apache projects**. This can potentially lead to JVM crashes, or the incorrect calculation of results, ultimately leading to bugs in applications. Currently, it is known that all versions of Lucene Core and Solr released today, are affected by these bugs. Java 6 users are also affected, if they use one of the JVM options that are not enabled by default:

```
-XX:+OptimizeStringConcat or
-XX:+AggressiveOpts
```

“These problems were detected only 5 days before the official Java 7 release, so Oracle had no time to fix those bugs,” states the announcement. “It is strongly recommended not to use any hotspot optimization switches in any Java version without extensive testing!”

Oracle have proposed to include fixes in service release u2, and eventually in service release u1.

Jessica Thornsby



THE H
O P E N The H open

Last 7 days News Archive Features

29 July 2011, 12:58

Java 7 paralyzes Lucene

The hotspot compiler in the recently defective optimiser that can cause a [warning](#) published by the Apache Solr result, the Java Virtual Machine can produce incorrect results.

A number of Apache projects are affected by [Lucene](#) and [Solr](#). The Apache developers say that Solr causes Java to crash. Loops in the code thereby corrupting the indexes. In particular, the [pulsing codec](#) is affected.

The bugs were discovered only five days before the official Java 7 release. Oracle says it will correct them in the second update to Java 7 was reserved. Oracle should refrain from using the new `-XX:-UseLoopPredicate` to disable the `-XX:+UseLoopPredicate` being damaged.

The Apache developers say that using `-XX:-UseLoopPredicate` However, the flaws only occur in Java 7. The `-XX:+OptimizeStringConcat` OR `-XX:+UseStringConcat` disabled Hotspot optimisations.

Oracle has registered the flaws and the first one causes JVM to crash when which traces English words back to "priority" while the others are "medium" (djwm)

cnet News | Home | Reviews | News | Downloads | Video | How To

Latest News | Apple | Crave | **Business Tech** | Green Tech | Wireless | Search

Home > News > Business Tech

Business Tech

Ad Info

Bis 12% und mehr



Endlich auch für Privatanleger: Teakholzinvestment ab 3.900€ bis zu 12% p.a. und mehr steuerfrei

Selbständig? Unter 55?



Private Krankenkasse ab nur 57,- Euro für Selbständige und Freiberufler unter 55 Jahren!

Gabelstapler



FILED UNDER: BUSINESS TECH

Oracle releases 'buggy' Java SE7

By: Ben Woods
JULY 29, 2011 10:25 AM PDT

Print | E-mail

Recommend 33 | Tweet 84 | +1 9 | Share | 7 comments

Oracle released its first full version of Java yesterday, but developers have reported bugs that can crash virtual machines, corrupt data, and cause errors in applications.

Java Standard Edition 7 (SE7) is the first milestone since **Oracle bought Java's creator, Sun**, which at the time prompted fears from some community members about the future of Java.

The release includes improved **support for dynamic languages**, multicore-compatible APIs, and additional networking and security features. Oracle said in a statement it is the culmination of "industry-wide development involving open review, weekly builds and extensive collaboration between Oracle engineers and members of the worldwide Java ecosystem."

However, the Apache Lucene search engine project management committee warned yesterday that Java SE7 contained bugs that could cause a Java Virtual Machine to crash or affect applications.

See also: [Scoop: Oracle scrubs site of embarrassing Java blog](#)

Read more of "Oracle releases 'buggy' Java SE7" at ZDNet UK.

inventor

JAVA TECH JOURNAL

CLIPSE | ANDROID | ARCHITECTURE | CLOUD

OpenShift, in the new issue of Java

NEWS | BOOKS | VIDEOS | EVENTS

Some Apache Projects

Share 2 | Comment | Email | Share

Java 7 contains hotspot compiler and this can **affect the code of "several"** JVM crashes, or the incorrect calculation of... Currently, it is known that all versions of... affected by these bugs. Java 6 users are also... are not enabled by default:

5 days before the official Java 7 release, so... "It is strongly... not optimization switches in any Java version

release u2, and eventually in service release

Jessica Thornsby





The First Word on Tech INFOWORLD TECH WATCH

JULY 29, 2011

Apache and Oracle warn of serious Java 7 compiler bugs

The newly released Java upgrade suffers hotspot-compiler problems that affect Lucene and Solr

By [Ted Samson](#) | InfoWorld

[Follow @tsamson_IW](#)

[Print](#) | [4 Comments](#)

[Gefällt mir](#)

It looks like a few bugs have crashed [Oracle's Java 7 release party](#) that can wreak havoc on Apache Project applications. The news likely will come as a disappointment to fans of Java, who've waited five long years for a major update to the language.

Released just today, Java 7 includes hotspot-compiler optimizations that miscompile certain loops, potentially affecting projects such as Apache Lucene Core, Apache Solr, and possibly others, [according to a warning](#) from the Apache Project. At best, the bugs only cause JVMs to crash; in other cases, they result in miscalculations that can lead to application bugginess.



aventer

Downloads Video How To

Business Tech Green Tech Wireless Se

Ad Info

selbständig? Unter 55? Private Krankenkasse ab nur 57,- Euro für Selbständige und Freiberufler unter 55 Jahren!

Gabelst

Share 2 Comment Email Share

Print E-mail

Share 7 comments

rs have reported bugs that can crash virtual

bought Java's creator, Sun, which at the time of Java.

multicore-compatible APIs, and additional the culmination of "industry-wide development between Oracle engineers and members of the

committee warned yesterday that Java SE7 or affect applications.

va blog



OpenShift, in the new issue of Java

NEWS BOOKS VIDEOS EVENTS

Some Apache Projects

Share 2 Comment Email Share

Java 7 contains hotspot compiler and this can **affect the code of "several"** JVM crashes, or the incorrect calculation of ns. Currently, it is known that all versions of d by these bugs. Java 6 users are also are not enabled by default:

by 5 days before the official Java 7 release, so gs," states the announcement. "It is strongly ot optimization switches in any Java version

release u2, and eventually in service release

Jessica Thornsby



Categories

- apache
- ApacheCon
- Books
- BoostingTermQuery
- Droids
- ecommerce
- Enterprise Search
- Events
- functions
- Hadoop
- Libraries
- Lucene
- Lucene Connector Framework
- Lucid Imagination
- Lucid Imagination Solutions

- LucidGaze
- LucidWorks
- Lucy
- Mahout
- ManifoldCF
- NoSQL
- nutch
- Open Relevance Project
- Open Source
- opennlp
- Payloads
- PyLucene
- Relevancy

Don't Use Java 7, For Anything

July 28, 2011

Posted by *hossman*

Java 7 GA was released today, but as noted by Uwe Schindler, there are some very frightening bugs in HotSpot Loop optimizations that are enabled by default. In the best case scenario, these bugs cause the JVM to crash. In the worst case scenario, they cause incorrect execution of loops.

Bottom Line: **Don't use Java 7** for anything (unless maybe you know you don't have any loops in your java code)

From: Uwe Schindler

Date: Thu, 28 Jul 2011 23:13:36 +0200

Subject: [WARNING] Index corruption and crashes in Apache Lucene Core / Apache Solr with Java 7

*Hello Apache Lucene & Apache Solr users,
Hello users of other Java-based Apache projects,*

Oracle released Java 7 today. Unfortunately it contains hotspot compiler optimizations, which miscompile some loops. This can affect code of several Apache projects. Sometimes JVMs only crash, but in several cases, results calculated can be incorrect, leading to bugs in applications (see Hotspot bugs 7070134 [1], 7044738 [2], 7068051 [3]).

Apache Lucene Core and Apache Solr are two Apache projects, which are affected by these bugs, namely all versions released until today. Solr users with the default configuration will have Java crashing with SIGSEGV as soon as they start to index documents, as one affected part is the well-known Porter stemmer (see LUCENE-3335 [4]). Other loops in Lucene may be miscompiled, too, leading to index corruption (especially on Lucene trunk with pulsing codec; other loops may be affected, too - LUCENE-3346 [5]).

Recent Posts

- Multivalued geolocation fields in Solr
- Monitoring Apache Solr and LucidWorks with Zabbix
- Lucene in Barcelona, in Action
- SF Bay Lucene/Solr Meetup Attracts 100 Attendees (and a special appearance by Doug Cutting!)
- Announcing LucidWorks 2.0, the search platform for Apache Solr/Lucene
- Some more European Search in Action
- Lucene goes from Enterprise Search to search platform
- SF Bay Area Lucene/Solr Meetup: 9/22 6:30PM (<http://bit.ly/r19aZx>)
- Happy Anniversary, Lucene! 10 years at the ASF
- Stump The Chump? Win A Prize!

Archives

Jessica Thornsby

JULY 29, 2011

Apache compiler problems

By Ted Sams



Java 7 release

Apache Project

come as a dis

waited five lon

language.

Released just

compiler optim

loops, potentially affecting projects such as

Apache Lucene Core, Apache Solr, and possibly

others, according to a warning from the Apache Project. At best, the bugs only cause JVMs to

crash; in other cases, they result in miscalculations that can lead to application bugginess.

Further analysis the week after

Further an

The screenshot shows a web browser window with a search bar and navigation links (Share, Report Abuse, Next Blog). The main heading is "Enterprise Software Development with Java". The introductory text states: "This is a blog about software development for the enterprise. It focuses on Java Enterprise Edition (J2EE/Java Beside this, I blog about Oracle WebLogic and GlassFish Server and other technologies that hit my road." The date is "FRIDAY, JULY 29, 2011". The article title is "Don't Use Java 7? Are you kidding me?". The main text discusses the release of Java 7 and the community's reaction, mentioning a "general warning" issued by Uwe Schindler. Social media buttons for "Tweet" (40) and "+1" (10) are visible. A sidebar on the right includes an "ABOUT ME" section with a profile picture of "myfear", a "Subscribe in a reader" button, and a "MORE EISELE.NET" section with links to "home", "work", "blog", "personal", and "disclaimer". A "LABELS" section lists "oracle", "server", "weblogic", "glassfish", "java EE", and "article". An image of a keyboard with a "Java" key and a "Download" key is shown at the bottom right of the article content.

Share Report Abuse Next Blog»

Enterprise Software Development with Java

This is a blog about software development for the enterprise. It focuses on Java Enterprise Edition (J2EE/Java Beside this, I blog about Oracle WebLogic and GlassFish Server and other technologies that hit my road.

FRIDAY, JULY 29, 2011

Don't Use Java 7? Are you kidding me?

Java 7 was released yesterday and some guys from the Apache Lucene & Apache Solr community quickly came up with a couple of issues which lead them to the point where they are actively rejecting Java 7 and advice anybody else to to likewise. Even a [general warning](#) was issued by Apache Lucene PMC Member Uwe Schindler. But what exactly is wrong with Java 7 and why shouldn't you use it after waiting nearly five years for it? Let's look at this.

[Tweet](#) 40
[+1](#) 10

It's not about Java 7 but about the JVM
First of all, it's not about Java 7 in general but about the HotSpot JVM. The GA release contains three bugs (7070134, 7044738 and 7068051) which could affect the users with either JVM crashes or wrong calculations.

Hotspot crashes with sigsegv from PorterStemmer
The first one is about a wrong compiler optimization that changed the loop optimizations. The problem is, that this JVM feature is on by



oracle weblogic
server glassfish
java EE article javaOne

Get Involved

- About Java.net
- Create a Project
- Java.net Enhancements

Get Informed

- Articles
- Blogs
- Site Wiki
- Events
- Oracle University



Cay Horstmann

In Praise of Language Specs | Main | Inner Classes in Scala and Java



Java 7 Unsafe at Any Speed?

Posted by [cayhorstmann](#) on July 29, 2011 at 7:32 PM EDT



Some people are nervous about everything—killer bees, poison oak, martian invaders, socialized medicine, you know the type. I try not to be like that. When JDK 7 went final yesterday, I boldly went into my .bashrc and changed JAVA_HOME to point to jdk1.7.0. Then I read this.

So, apparently, under some conditions, Hotspot messes up. It might crash, but that doesn't bother me so much—I'd notice that. But it might also silently produce the wrong result. I try not to be a scaredy squirrel about these things, but I must say that "rarely happening" bugs in a widely used platform bother me. When Toyota cars had random brake problems, the NHSTA ultimately concluded that floor mats, sticky pedals, and "pedal misapplication" were the culprits. But what if the electronics had a bug that only happens in a confluence of rare circumstances? They said no, but how can they really know?

It all reminds me of the Pentium bug from 1994. Intel had just released the first Pentium chip, and I immediately went and bought one at considerable expense to the management. Later I learned that a mathematics professor, Dr. Thomas Nicely of Lynchburg College, had run into a curious issue. On a small set of inputs, the multiplication was buggy. For example, 4195835- 4195835 / 3145727 × 3145727 yielded 256 instead of the expected 0. I tried it out on my new computer. Sure enough, I got 256. I tried it out on an older 486. I got 0.

It turned out that Intel had known about the bug but decided to ship the processor anyway. Intel claimed that under normal use, a typical consumer would only notice the problem once every 27,000 years. Unfortunately for Intel, Dr. Nicely had not been a normal user. Intel stonewalled for a while, but eventually they sent out replacement chips for everyone.

n (J2EE/Java road.

myfear
German author and software architect
plete profile

in a reader

.NET
| blog | disclaimer

e is frozen
ppenauer)

weblogic
glassfish
ido JavaOne

Get Involved

- [About Java.net](#)
- [Create a Project](#)
- [Java.net Enhancements](#)

Get Informed

- [Articles](#)
- [Blogs](#)
- [Site Wiki](#)
- [Events](#)
- [Oracle University](#)



Fatal Exception NEIL MCALLISTER

AUGUST 04, 2011

Oracle: Java's worst enemy

The buggy Java SE 7 release is only the latest misstep in a mounting litany of bad behavior

By [Neil McAllister](#) | [InfoWorld](#) [Follow @infoworld](#) [Print](#) | [27 Comments](#) [Gefällt mir](#)

Oracle shipped Java SE 7 with a [serious, showstopping bug](#), and who was the first to [alert the Java community](#)? The Apache Foundation. Oh, the irony.

This is the same Apache Foundation that [resigned from the Java Community Process \(JCP\) executive committee](#) in protest after Oracle repeatedly refused to give it access to the Java Technology Compatibility Kit (TCK).

[Neil McAllister reveals [the most dangerous programming mistakes](#). | Get software development news and insights from [InfoWorld's Developer World newsletter](#). | And sharpen your Java skills with the [JavaWorld Enterprise Java newsletter](#).]

It's the same Apache Foundation that developed [Harmony](#), an open source implementation of the Java platform. Google used Harmony to build its Android mobile OS, which is now the subject of a [multi-billion-dollar lawsuit](#) from Oracle alleging intellectual property violations. Oracle has [subpoenaed documents](#) from the Apache Foundation to help make its case. Nobody is sure what this means for other Harmony users.



n (J2EE/Java road.

[myfear](#)

German author and software architect

[plete profile](#)[in a reader](#)

.NET

[| blog |](#)
[sclaimer](#)

e is frozen

ppenhauer)

[weblogic](#)
[glassfish](#)
[ido, JavaOne](#)

Get Involved

- About Java.net
- Create a Project
- Java.net Enhancements

Get Informed

- Articles
- Blogs
- Site Wiki
- Events
- Oracle University

Sign in

InfoWorld Home / Application Development /



Fatal Exception
NEIL MCALLISTER

AUGUST 04, 2011

Oracle: Java's worst enemy

The buggy Java SE 7 release is causing a mounting litany of bad behavior

By Neil McAllister | InfoWorld

Print 27 Comments

Oracle shipped Java SE 7 with a serious, show-stopping bug. What's the deal with the Java community? The Apache Foundation. Oh, yes.

This is the same Apache Foundation that resigned its executive committee in protest after Oracle released its Java Technology Compatibility Kit (TCK).

[Neil McAllister reveals the most dangerous pieces of development news and insights from InfoWorld. You can improve your Java skills with the JavaWorld Enterprise Edition.

It's the same Apache Foundation that developed Harmony, an open source implementation of the Java platform. Google used Harmony to build its Android mobile OS, which is now the subject of a multi-billion-dollar lawsuit from Oracle alleging intellectual property violations. Oracle has subpoenaed documents from the Apache Foundation to help make its case. Nobody is sure what this means for other Harmony users.



Your Maps
Your World

CHANNELS

- Architecture & Design
- C/C++
- Database
- Development Tools
- Embedded Systems
- High Performance Computing
- Java
- Jolt Awards
- Mobility
- Open Source
- Security
- Web Development
- Windows/.NET
- Visual Studio 2010

JAVA

Tweet 89 Gefällt mir Share

Sloppy Work at Oracle

By Andrew Binstock, August 01, 2011

5 Comments

Poor testing and bad decision-making mar an important release

Within a few days of my editorial suggesting that Java 7 be adopted quickly, news began to leak out that there were showstopper bugs in the Java 7 HotSpot compiler. I'll get into the defects shortly, but what really turned up the heat was Oracle's decision to ship the compiler aware that the known defects would cause one of two types of errors: hang the program or silently generate incorrect results. Given that Java 7 took five years to see light, it seems to me and many others that Oracle could have waited a bit longer to fix the bug before releasing the software. To a large extent, there is a feeling in the Java community that Oracle does not understand Java (despite the company's earlier acquisition of BEA). That may or may not be, but I would have expected it to understand enterprise software enough not to ship a compiler with defects that hang a valid program.

The problem, from what is known so far, derives from a command-line optimization switch on the Java compiler. This switch incorrectly optimized loops, resulting in the various reported errors. In Java 7, this switch is on by default, while it was off by default in previous versions.

INFO-LINK

- Threading? GPU load in Windows? Sound important?

Discussions

News: Lucene should just shut up about Java 7

SHARE

GET THREAD FEED

Lucene should just shut up about Java 7 (24 messages)

POSTED BY: Richard Mayhew POSTED ON: August 05 2011 12:00 EDT

When Java 7 was released last week, Lucene and Solr both issued a warning saying that you can't use Java 7 with Lucene or Solr. These are popular text search libraries (and Solr is an app) so this can be pretty severe, and the way the community was informed at first was pretty emotional. See [Java 7 paralyses Lucene and Solr](#) for an example of the kind of headline generated.

The actual warnings on the Apache sites are pretty mild and state the problem although not the history of the problem well, and give an expectation of when you'll be able to use Lucene with Java 7.

Some people have written some pretty good responses to the issue, see [Don't Use Java 7? Are you kidding me?](#) for one example.

These do a good job of hiding the emotions from the problem; in mailing lists and other forums, the word is that Java 7 is buggy, that Oracle's ignoring the problem (they're not, they are working on the bug and have a fix scheduled) and that Oracle didn't test enough.

I say that's crap. Sure, it'd be nice if Oracle never released a JVM with bugs, but it's silly to blame them for this.

There's enough blame for everyone, for Lucene and Oracle.

Oracle honestly did the right thing, though. They had some optimizations in Java 6 (-XX:+OptimizeStringConcat and -XX:+AggressiveOpts) that could break some code, like all things can. With Java 7, they made the optimizations on by default, starting a month before Java 7's release.

Java platform. Google used Harmony to build its Android mobile OS, which is now the subject of a [multi-billion-dollar lawsuit](#) from Oracle alleging intellectual property violations. Oracle has [subpoenaed documents](#) from the Apache Foundation to help make its case. Nobody is sure what this means for other Harmony users.



Your Maps
Your World

AVA

Tweet 89 Gefällt mir Share

Slippy Work at Oracle

Andrew Binstock, August 01, 2011

Comments

for testing and bad decision-making mar an important release

Within a few days of my editorial suggesting that Java 7 be adopted quickly, news began to leak out that there were showstopper bugs in the Java 7 HotSpot compiler. I'll get into the facts shortly, but what really turned up the heat was Oracle's decision to ship the compiler aware that the known defects would use one of two types of errors: hang the program or silently generate incorrect results. Given that Java 7 took five years to see light, it seems to me and many others that Oracle could have waited a bit longer to fix the bug before releasing the software. To a large extent, there is a feeling in the Java community that Oracle does not understand Java (despite the company's earlier acquisition of BEA). That may or may not be, but I would have expected it to understand enterprise software enough not to ship a compiler with defects that hang a valid program.

The problem, from what is known so far, derives from a command-line optimization switch on the Java compiler. This switch correctly optimized loops, resulting in the various reported errors. In Java 7, this switch is on by default, while it was off by default in previous versions.



AGILE ZONE

Software Methodologies for Development Managers

Home Microzones Zones Library Refcardz Links ITQuestions Snippets



No **<BS>** APIs
So easy to integrate into your
existing system, it's almost criminal!

Home



Of communities, companies, and bugs (Or, "Dr Dobbs Journal is a slut!")

Submitted by Ted Neward on Sat, 2011/08/06 - 11:25am

Tags: Java Oracle Agile

Andrew Binstock (Editor-in-Chief at DDJ) [has taken a shot at Oracle's Java7 release](#), and I found myself feeling a need to respond.

In his article, Andrew notes that

... what really turned up the heat was Oracle's decision to ship the compiler aware that the known defects would cause one of two types of errors: hang the program or silently generate incorrect results. Given that Java 7 took five years to see light, it seems to me and many others that Oracle could have waited a bit longer to fix the bug before releasing the software. To a large extent, there is a feeling in the Java community that Oracle does not understand Java (despite the company's earlier acquisition of BEA). That may or may not be, but I would have expected it to understand enterprise software enough not to ship a compiler with defects that hang a valid program.

We Recommend These Resources

- Enterprise Integration Patterns: Past, Present and Future
- Open Source as a Catalyst for Innovation and Cultural Change featuring Yahoo!
- The Future of Camel
- Migrating to FUSE Mediation Router
- Apache Camel - How to go from EIPs to production

There's so many things in this paragraph alone I want to respond to, I feel it necessary to deconstruct it and respond individually:

this means for other Harmony users.



Search: Site Source Code

Logs Source Code Dobbs on DVD Dobbs TV Webinars



Your Maps
Your World



weet 89 Gefällt mir Share +1 SU FB Email Print Permalink

Happy Work at Oracle

Andrew Binstock, August 01, 2011

Comments

Testing and bad decision-making mar an important release

In a few days of my editorial suggesting that Java 7 be tested quickly, news began to leak out that there were stopper bugs in the Java 7 HotSpot compiler. I'll get into the details shortly, but what really turned up the heat was Oracle's decision to ship the compiler aware that the known defects would be one of two types of errors: hang the program or silently generate incorrect results. Given that Java 7 took five years to see light, it seems to me and many others that Oracle could have waited a bit longer to fix the bug before releasing the software. To a large extent, there is a feeling in the Java community that Oracle does not understand Java (despite the company's earlier acquisition of BEA). That may or may not be, but I would have expected it to understand enterprise software enough not to ship a compiler with defects that hang a valid program.

The problem, from what is known so far, derives from a command-line optimization switch on the Java compiler. This switch controls directly optimized loops, resulting in the various reported errors. In Java 7, this switch is on by default, while it was off by default in previous versions.

... subject of a ... as ... sure what



AGILE ZONE

Software Methodologies for Development Managers

Home Microzones Zones Library Refcardz Links ITQues



No <BS>
So easy to integrate
existing system, it

Home



Of communities, companies, and “Dr Dobbs Journal is a slut!”

Submitted by Ted Neward on Sat, 2011/08/06 - 11:25am

Tags: Java Oracle Agile

Andrew Binstock (Editor-in-Chief at DDJ) [has taken a shot at Oracle's Java7 release](#), and I found myself feeling a need to respond.

In his article, Andrew notes that

... what really turned up the heat was Oracle's decision to ship the compiler aware that the known defects would cause one of two types of errors: hang the program or silently generate incorrect results. Given that Java 7 took five years to see light, it seems to me and many others that Oracle could have waited a bit longer to fix the bug before releasing the software. To a large extent, there is a feeling in the Java community that Oracle does not understand Java (despite the company's earlier acquisition of BEA). That may or may not be, but I would have expected it to understand enterprise software enough not to ship a compiler with defects that hang a valid program.

There's so many things in this paragraph alone I want to respond to, I feel it necessary to respond to them individually:

this means for other Harmony users.

Jaxenter

NEWS VIDEOS BOOKS EVENTS JAVA TECH JOURNAL
HOME NEWS EDITORS PICK JAVA ECLIPSE ANDROID ARCHITECTURE CLOUD

Find out what's new in JBoss AS7 and OpenShift, in the new issue of Java Tech Journal!

August 10, 2011

RELATED NEWS BOOKS VIDEOS EVENTS

Java 7 Debate Rages On

Java 7 Bugs: Should the Release Have Been Delayed?

Share Tweet 6 +1 0 Share Comment Email Share

Java 7 may have brought with it some useful (and long-awaited) updates for the Java community, but it has also sparked controversy as the release shipped with some **bugs in the Java 7 HotSpot compiler**. These bugs affect all currently released versions of Apache Lucene Core and Apache Solr, but the problem could also affect Java 6 users, if they use one of the JVM options that are not enabled by default:

-XX:+OptimizeStringConcat or
-XX:+AggressiveOpts

These bugs were discovered five days before Java 7 was published, which has caused some to question whether the release should have been delayed. At his blog, Uwe Schindler, who first posted about the bug, has drawn attention to the fact that the final release of Java 7, is the same as the preview release, and has questioned the point of the preview release. "It was for sure not intended for public review and bug hunting!" he says. Others, such as Markus Eisele, have defended Oracle, stressing that: "these problems were detected only 5 days before the official Java 7 release, so Oracle had no time to fix those bugs."

Andrew Binstock, Executive Editor of Dr. Dobb's, has [posted his thoughts on the controversy](#), referring to the HotSpot compiler problems as "showstopper bugs" and stating that Oracle should have delayed the release. He goes on to claim that there is a "feeling in the



The Porter Stemmer SIGSEGV Bug

Java 7 Crashes Eclipse...

What's wrong with these methods?

```
private final void step4() {
    switch (b[k]) {
        case 'e': if (ends("icate")) { r("ic"); break; }
                  if (ends("ative")) { r(""); break; }
                  if (ends("alize")) { r("al"); break; }
                  break;
        case 'i': if (ends("iciti")) { r("ic"); break; }
                  break;
        case 'l': if (ends("ical")) { r("ic"); break; }
                  if (ends("ful")) { r(""); break; }
                  break;
        case 's': if (ends("ness")) { r(""); break; }
                  break;
    }
}
```

```
private final boolean ends(String s) {
    int l = s.length();
    int o = k-l+1;
    if (o < 0) return false;
    for (int i = 0; i < l; i++) {
        if (b[o+i] != s.charAt(i)) return false;
    }
    j = k-1;
    return true;
}
```

Conclusion: Porter Stemmer Bug

- Less serious bug as your virtual machine simply crashes. You won't use it!
- Oracle marked bug report “serious”, as this affects their software reproducible to everyone.
- Can be prevented by JVM option:
-XX:-UseLoopPredicate



The Vint bug

Loop Unwinding

What's wrong with this method?

```
/** Reads an int stored in variable-length format. Reads between one and
 * five bytes. Smaller values take fewer bytes. Negative numbers are not
 * supported.
 * @see IndexOutput#writeVInt(int)
 */
public int readVInt() throws IOException {
    byte b = readByte();
    int i = b & 0x7F;
    for (int shift = 7; (b & 0x80) != 0; shift += 7) {
        b = readByte();
        i |= (b & 0x7F) << shift;
    }
    return i;
}
```

What's wrong with this method?

```
/** Reads an int stored in variable-length format. Reads between one and
 * five bytes. Smaller values take fewer bytes. Negative numbers are not
 * supported.
 * @see DataOutput#writeVInt(int)
 */
public int readVInt() throws IOException {
    byte b = readByte();
    int i = b & 0x7F;
    if ((b & 0x80) == 0) return i;
    b = readByte();
    i |= (b & 0x7F) << 7;
    if ((b & 0x80) == 0) return i;
    b = readByte();
    i |= (b & 0x7F) << 14;
    if ((b & 0x80) == 0) return i;
    b = readByte();
    i |= (b & 0x7F) << 21;
    if ((b & 0x80) == 0) return i;
    b = readByte();
    assert (b & 0x80) == 0;
    return i | ((b & 0x7F) << 28);
}
```

Conclusion: Vint Bug

- **Serious data corruption:** Some methods using loops silently return wrong results!
- Bug already existed in Java 6
 - appeared some time after 1.6.0_18, enabled by default
 - is prevented since Lucene 3.1 by manual loop unwinding (*helps only in Java 6*)
- Cannot easily be reproduced, Oracle assigned “medium” bug priority – was never fixed in Java 6.
- **Problems got worse with Java 7**, only safe way to prevent is to disable loop unwinding completely, but that makes Lucene very slow.

How to debug hotspot problems

Hands-On

First...



- Fetch some beer!
- Tell your girlfriend that you will not come to bed!
- Forget about Eclipse & Co! We need a command line and our source code...

Hardcore: Debugging *without* Debugger

- Open `hs_err` file and watch for stack trace (*if your JVM crashed like in Porter stemmer*)
- *Otherwise:* disable Hotspot to verify that it's not a logic error! (`-Xint / -Xbatch`)
- Start to dig around by adding `System.out.println`, assertions, ...
Please note: You cannot use a debugger!!!



Digging...

- If you found a method that works incorrectly, disable Hotspot optimizations for only that one:
`-XX:CompileCommand=exclude,your/package/Class,method`
 - **If program works now, you found a workaround!**
 - **But this may not be the root cause – or does not help at all!**
- Step down the call hierarchy and replace exclusion by methods called from this one.
- **Open a bug report at Oracle!**
- Inform **hotspot-compiler-dev@openjdk.java.net** mailing list.

Java 7 Update 1?

- Brand new: **Java 7u1** released yesterday!
- Bugs *seem* to be fixed, but nothing in release notes!

Don't use Java 7 until an official statement from Oracle!

See my blog post: <http://s.apache.org/Java7u1Lucene>

Contact

Uwe Schindler

uschindler@apache.org

<http://www.thetaphi.de>

 @thetaph1

*Many thanks to **Robert Muir** for help during debugging these bugs!*



SD DataSolutions GmbH

Wätjenstr. 49

28213 Bremen, Germany

+49 421 40889785-0

<http://www.sd-datasolutions.de>

