#SHAREorg

SHARE
Technology · Connections · Results

Linux on System z Java Tips and Tricks: Running an Oracle Java Only Supported Portal on IBM Java and the OpenJDK Future

Brian Gormanly
Marist

Thursday, August 9th, 2012 9:30 AM PDT
11944

SHARE in Anaheim 2012

Trademarks

- z/VM® is a registered trademark of IBM Inc.
- Performance Toolkit for VM is a registered trademark of IBM Inc.
- System z9® is a registered trademark of IBM Inc.
- System z10® is a registered trademark of IBM Inc.
- System z196® is a registered trademark of IBM Inc.
- Ellucian, Luminis and Banner are registered trademarks of Ellucian Inc.
- Intel™ is a *registered trademark* of Intel Corporation
- Xeon™ is a trademark of Intel Corporation
- Oracle Database is a registered trademark of Oracle Corporation
- Oracle Enterprise Manager is a trademark of Oracle Corporation
- Spotlight for Oracle is a trademark of Quest Software
- Java is a registered trademark of Oracle Corporation.

Agenda

- Background
  - Java applications we currently or previously have had on System z
- Implementing our new enterprise portal on IBM Java
  - pingcp.jar (com.sun.net.ssl.HostnameVerifier)
  - algorithm="IbmX509”
  - JSVC
  - Performance
- Experimenting with Java on System z
  - J9 Engine and Java versions
  - Garbage collection / memory management
  - Monitoring
- Conclusions

Background

Java

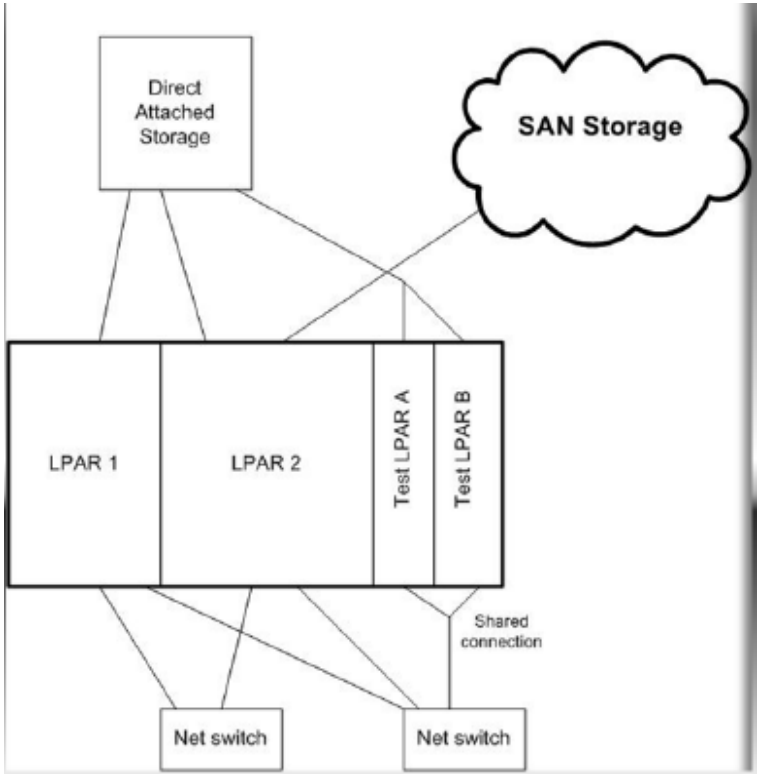
MARIST™

Our History on System z

- Marist and the IBM Mainframe Long history of innovative use of mainframe services to support academic experience
- First IBM mainframe at Marist: 1978
- First academic collaboration with IBM: 1981
  - Processor progression: S370 -> 4341 -> 4381 -> 3090 ->9121 -> 9672 ->z900 -> z9 -> z10 -> z114 (2)
  - Also z990 on loan from IBM for joint projects
  - Various models within each of the processors

Our History on System z (continued)

- Primary mainframe – z114 2 General Processors, 8 Integrated Facility for Linux (IFLs), 80G memory
  - 2 LPARs - z/VM runs both MARIST – contains main legacy systems (4 z/OS 1.9), numerous Linux and CMS based services
  - MARLINUX – Linux servers only (60 and growing)



Our History on System z (continued)

- Development mainframe – z990 Used for testing, development, and proof-of-concept projects.
  - Original part of OSDL project
  - Debian and Slackware have servers for development work
  - Hosts primary “git” server for Linux on System z (allows Open Source community to submit code for review and incorporation into Linux kernel)
  - Hosts Marist student academic servers (>600 currently)
  - Cognos
  - SPSS
  - Sakai Instances (Greystone Project)
  - Rational Servers

Neat Marist and System z Facts

- Marist produced the first Linux on System z distribution, known as Marist Linux.
- Leadership position with the Enterprise Computing Community funded by National Science Foundation (NSF)
- Marist hosts the online Franklin D. Roosevelt (FDR) Library on one of our mainframes.
- git://git390.marist.edu – Linux for z kernel repository

Java on our System z (Past and Present)

- Sakai (our learning management system)
- Jobs (jobs.marist.edu)
- Liferay (enterprise open source portal)
  - Multiple installations
- IBM Cognos
- IBM OmniFind
- Workplicity
- SPSS (statistics software)
- CAS (our SSO software)
- Luminis (our new enterprise portal, the subject of the next section of this presentation)

Implementing our New Portal

503 Service Unavailable: http://httpstatusdogs.com/503-service-unavailable

Software Details

- Luminis is a enterprise portal produced by Ellucian (formally Sungard Higher Education)
- New version of Luminis a complete overhaul of the existing version which was based on uPortal.
- New Luminis version based on Liferay Portal
  - We are already successfully running Liferay on System z
  - Currently based on over 3 year old Liferay version! (5.2.3)
- Multiple tier architecture
- Initially planned to use MySQL database

Luminis Tiers and Services

- Administrative portal tier running on Tomcat 5.5
  - Complete portal tier used for administrative management
- CAS for SSO with other Ellucian services running in it's own Tomcat.
  - We are not using the delivered CAS software with the Luminis software in favor of our existing Enterprise CAS
- OpenDS LDAP stand alone Java
  - Used to pass off messages from glassfish query broker for JMS
- Java Message Queue running in glassfish container

Luminis Tiers and Services (continued)

- MySQL Database: The software supports using either MySQL or Oracle DB.
- 1 or more portal tiers running on Tomcat 5.5
  - 1 Portal tier is currently sufficient for us, but we setup 2 in development for testing.
- Load balancer / proxy (Apache)

Source: http://devhumory.com

Implementation

- We were very hopeful this would be an easy conversion to run on System z since the software that Ellucian based Luminis off of, Liferay, ran on System z out of the box, with no modifications.
- We would not get off that easy of course…
- We encountered 3 separate problems attempting to run the portal on a zLinux guest on our System z114.

pingcp.jar

- Ellucian install and startup scripts failed on executing a class in this jar.
- The class was essentially just a rapper for doing a ping on the Luminis LDAP server to make sure the service had successfully started.
- We were receiving the following in the stack trace:

```
Exception in thread "main" java.lang.NoClassDefFoundError:
com.sun.net.ssl.HostnameVerifier
        at java.lang.J9VMInternals.verifyImpl(Native Method)
        at java.lang.J9VMInternals.verify(J9VMInternals.java:72)
```

pingcp.jar (continued…)

- The authors of the class were incorrectly directly importing from sun's own SSL library, dependencies to sun libraries created a Oracle Java dependent application.
- We fiddled about trying to make jsse.jar (the missing jar) available in the classpath.
- Ultimately we decided that the need to ping the LDAP server in the installation and startup scripts was an avoidable requirement. We removed the calls to the pingcp code.
- If needed we would always write our own ping shell script.

algorithm="IbmX509"

- After removing the pingcp references we ran the install script again with great anticipation…
- Fail.

java.io.IOException: Error initializing socket factory SSL context:

SunX509 TrustManagerFactory not available

at sun.security.jca.GetInstance.getInstance(GetInstance.java:230)

at javax.net.ssl.TrustManagerFactory.getInstance (TrustManagerFactory.java:11)

- By default the KeyManagerFactory setup was the Oracle/ Sun algorithm.

algorithm="IbmX509" (continued…)

- After a bit of searching around we realized we just had to make a configuration change to use the IBM algorithm
- Normally this is found in the server.xml file depending on the java portlet container. We are using Tomcat for all the tier affected.
  - In the <Connector> tag there is a keystoreFile attribute that needs to have algorithm="IbmX509" added to look like:
    - keystoreFile="/opt/luminis/.keystore" algorithm="IbmX509"
    - Default is algorithm="SunX509" even if it is not explicitly defined.
- The IBM JVM does not support the Sun algorithm.

JSVC

- By default the Luminis portal uses JSVC for Tomcat.
- JSVC allows Tomcat to perform some privileged operations as root (e.g. bind to a port < 1024), and then switch identity to a non-privileged user.
- JSVC does did not work on IBM java.
- JSVC is not the only way to deal with the problem.
- We decided on using apache with mod_ajp to forward all traffic to to tomcat on port 8080.
- We actually do not look as this as a work around at all, but a better way to deal with the problem of the privileged port, so we did expend any energy on JSVC on J9.

Up and Running…

- After addressing these 3 issues we were able to successfully install and run the portal.
- Everything else worked
- All these tiers use an incredible amount of memory. If you thought 1 java application was a memory hog, just image what a java portal tier, java admin tier, java base LDAP, java messaging, database server and java CAS server can do!!!!

Performance

- We initially ran into a lot of performance issues.
- Being that we were the first ones attempting to run the software on System z and we have become accustom to the vendors first response pointing the finger at the platform, we took the initiative and installed on System X as well.

Performance (continued)

- Performance issues appeared on System X as well.
- Symptoms were the same on both platforms
  - Portlets were timing out before rendering
  - Entire UI held up while waiting for responses.
- We search support requests and talked with other schools and discovered we were not the only ones running into this issue.
- After pounding our heads for weeks in frustration and many calls with the vendor we decided to try using Oracle for the database instead of MySQL.

Performance (continued)

- Even though both MySQL and Oracle are supported databases when using Oracle the timeout issues all but disappeared.
- Turned out that the developers at Ellucian only used Oracle during development. Most likely reason is that their ORM layer is highly tuned for Oracle and not MySQL.
- Timeout issues still happen rarely under certain circumstances.
- Timeout issues seemed to surround waiting on Ellucian delivered portlets.

Web 2.0 What??????????

- Next we created a custom portlet to do a directory lookup against or enterprise LDAP (making Asynchronous calls) the portlet would not deploy in Luminis. But it worked fine in Liferay, the underlying portal open source software.
- Turns out that Ellucian had decided to completely turn of the ability for portlets to make asynchronous requests.
- We were able to have a dialog with some of the developers at Ellucian on the issue
  - Portals by there very nature are supposed to pull information from many different systems (some times with high latency)
  - Users expect the overall UI to render within a second or two.

Back to Basics

- As a result of our dialog Ellucian admitted the need to revisit the decision to disable asynchronous portlet communication.
- They have provided us with the means to remove the overall timeout set at 10 seconds, but we are still working on getting asynchronous portlets working again.

What to do with Luminis???

- So with Oracle as the datastore things are running better but we still had a platform decision to make for the software since now it was running on both System z and X.
- For now the massive memory footprint that this software requires (total of 24 GB) has had us leave the software on System X for now.
- But this exorcise made us take a deeper look at Java in general on System Z…

Experimenting with Java on System z

Source: http://ilike.nitty-witty.com

The Java Question

- We realized our exorcise trying to run the Luminis portal on System z essentially was a question of getting it to run on the IBM JVM.
- Recently the amount of java based software we are deploying and supporting seems to have been growing.
- The emergence of the OpenJDK as the standard reference implementation as of Java 7 caused us to question how this would affect our JVM choices in the future.

The Pieces…

- To best understand the problem, one has to understand the components.
- A Java full implementation usually consists of 3 principle components
  - Virtual Machine
  - Java Class Library
  - Compiler
- Java support on a platform centers on the virtual machine.
- There are many implementations of Java and much has changed in the last 2 years.

Some Implementations of Java

- Proprietary
  - Jrocket – Oracle
  - J9 – IBM
  - HP Java – HP
  - Apogee
  - Mac OS runtime
  - PERC
  - Mac OS runtime
  - SAPJVM
  - JamaicaVM
  - …
- Open Source
  - Hotspot – Oracle
  - Apache Harmony
  - Dalvik – Google / Android
  - IcedTea
  - TinyVM
  - Jikes (formerly IBM Jalapeno)
  - GCJ – compiles java to native machine code.
  - …

The Current State of Java

- IBM, SAP and Oracle have backed the OpenJDK as the reference implementation.
- Apple contributed it's own virtual machine, libraries, etc. to the OpenJDK port, based on the existing BSD port.
- OpenJDK
  - OpenJDK 8 is on the way soon. It has been released as a preview on July 5th, 2012 (currently build 46)
  - OpenJDK 7 was released on July 22nd, 2011 (first OpenJDK reference implementation)
  - OpenJDK 6 was released on February 12th, 2008

Current State of Java for zLinux

- Hopspot is only supported on i386, x86-64 and SPARC
- J9 is IBM's JVM that runs on PPC-32, PPC-64, x86-32, x86-64, s390, s390x for linux.
- The J9 (currently at version R26) is the java virtual machine that is available to run on s390 and s390x.
- IBM's backing of the OpenJDK does not mean that the Oracle Hotspot implementation will run on System z.
  - Hotspot does not know how to talk to the System z hardware
  - The J9 Implementation can talk to the hardware.
- Currently both java 7 and 6 are available for the J926 version. 6R26 and 7R26

Advantages to J9

- Platform specific tuning
  - System z196 Exploitation
  - OOO Pipeline
- IBM Monitoring and Diagnostic Tool
  - Improve application performance
  - Optimize garbage collection
  - Can be used to find potential problems in the application
- Concurrency libraries
  - Spreading out he threads

64 Bit Memory Optimization

- 64 to 32 bit memory optimization (-Xcompressedrefs)
  - Causes object references to be stored as 32 bit representations
  - As the 64-bit objects with compressed references are smaller than default 64-bit objects, they occupy a smaller memory footprint in the Java heap and improves data locality. This results in better memory utilization and improved performance.
  - When your Java applications does not need more than a 25 GB Java heap.
  - When your application uses a lot of native memory and needs the JVM to run in a small footprint.

Garbage Collection

- Default mode in R26 -Xgcpolicy:gencon
  - Tries to optimize both throughput and pause times
  - divides the heap into 2 spaces - nursery and the tenure space: All new objects go into the nursery. Because new objects tend to die off the nursery can be collected separately.
  - Works very well if you are in a environment where objects really do die young it must be collected quick and frequent.
  - If an object survives a set number or collections then it moves to the tenure space. Once in the tenure space the object is not looked at again until the tenure space is CG.
    - This only happens when the tenure space is full.

Garbage Collection (continued…)

- -Xgcpolicy:optthruput
  - Optimized for throughput (used to be the default garbage collection policy)
  - work happens until the garbage collector hits a threshold and then it pauses to clean up
- -Xgcpolicy:optavgpause
  - Reduce pause times lets let the garbage collector do some work as the application is running. at some point in time the application will need to stop but the pause will be shorter.
  - If you are having problems with erratic application response times that are caused by normal garbage collections, you can correct those problems, at the cost of some throughput

Garbage Collection (continued…)

- -Xgcpolicy:balanced
  - Added in Java 6.0.1/7
  - Works well for large heaps
  - Greater then 4GB
- -Xgcpolicy:subpools
  - Deprecated!
  - optimized for multi processor systems (recommended for 16 or more processors)

Conclusions

- J9 is still the (only) way to go for zLinux.
- Generally implementing java software on the IBM J9 based JVM is painless.
  - Because the IBM implementation more strictly follows standards
  - If the software vendor did not comply by say, referencing a sun library directly, all may not be lost.
- Once understood the tuning abilities of the J9 memory management and garbage collection offer substantial application performance gains.
- Tools such as the Monitoring and Diagnostic Tool can help with discovery of potential problems.

References

- **IBM Java 6.0.1 and 7.0 in zEnterprise and Technology Update** - Ken Irwin, Marcel Mitran, Theresa Tai – Share Atlanta 2012
- **IBM Monitoring and Diagnostic Tools for Java -** http://www.ibm.com/developerworks/java/library/j-ibmtools2/
- **IBM User Guides for Java v6 on Linux**: More effective heap usage using compressed references - http://publib.boulder.ibm.com/infocenter/javasdk/v6r0/index.jsp?topic=%2Fcom.ibm.java.doc.user.lnx.60%2Fuser%2Fgarbage_compressed_refs.html
- **JVM Performance Tuning with respect to Garbage Collection(GC) policies for WebSphere Application Server V6.1 - Part 1** - Giribabu Paramkusham, Ajay Bhalodia http://www-01.ibm.com/support/docview.wss?uid=swg27013400

Links

- General JVM
  - http://www.s390java.com/
  - http://en.wikipedia.org/wiki/List_of_Java_virtual_machines
  - http://www.ibm.com/developerworks/java/library/j-ibmtools2/
- Open JDK
  - https://blogs.oracle.com/henrik/entry/moving_to_openjdk_as_the
  - http://en.wikipedia.org/wiki/OpenJDK
- J9
  - http://www-03.ibm.com/systems/z/os/zos/tools/java/
  - http://wiki.eclipse.org/J9
- Compiler for java
  - http://gcc.gnu.org/java/
  - http://en.wikipedia.org/wiki/GNU_Compiler_for_Java

Questions?

Thank You!