Tuning Your GlassFish – Performance Tips

Deep Singh
Enterprise Java Performance Team
Sun Microsystems, Inc.

Presentation Goal

Learn tips and techniques on how to improve performance of GlassFish Application Server

Presentation Agenda

- GlassFish Out-of-Box Performance
- How to Tune GlassFish
- Performance Tuning GlassFish
- Performance Best Practices
- Performance results
- GlassFish Performance Tuning References

GlassFish Out-of-box performance

- Most GlassFish components are tuned to perform well out-of-box
  - Some components may not be tuned enough for performance tests or production environment
- Depending on your platform and application, you can tune further
  - Tune only what you need
  - Tuning can be a repetitive process
  - Tune judiciously

How to tune GlassFish

- Two methods to apply tunings
- Use Admin Console
  - > Done through a browser
  - > Default admin port is 4848
    - > For example - http://localhost:4848
- Use 'asadmin' command
    - > 'asadmin' binary is in GlassFish bin directory
    - > Execute 'asadmin set' command
- Use GlassFish monitoring to help you with tuning
  - > You can monitor using Admin Console or 'asadmin get' command

Presentation Agenda

- GlassFish Out-of-Box Performance
- How to Tune GlassFish
- Performance Tuning GlassFish
- Performance Best Practices
- Performance results
- GlassFish Performance Tuning References

Basic JVM Tuning

- JVM can run in client or server mode
  - > Different modes are targeted for different class of machines
  - > Right mode can produce optimized performance
- Client mode
  - > Java option '-client': for developer profile
  - > It is GlassFish default
  - > Used mostly for application development
- Server mode
  - > Java option '-server': for cluster profile
  - > Recommended for performance testing

Basic JVM Tuning

- Java heap size affects performance
  - All objects are created and maintained in Java heap
  - A larger heap can have more objects - but can also lead to longer garbage collection times
- Minimum Java heap size
  - Set using Java option '-Xms'
  - Glassfish default is 512 MB
- Maximum VM heap size
  - Set using Java option '-Xmx'
  - Recommended to set value based on available physical memory
- Recommended to keep same values for -Xms and -Xmx to avoid heap re-sizing during performance tests

Web Container Tuning

- Tune HTTP and Keep-Alive connections
- HTTP service provides a pool of threads for processing HTTP requests
  - Adjust number of request processing threads based on load
    - Default thread count = 5
    - For peformance testing, recommended 32 or higher
    - Use GlassFish monitoring to find right value
- Keep-Alive subsystem keeps HTTP connections alive until client disconnects or times out
  - Adjust max connections
    - Default is 250 connections
    - For peformance testing, recommended 10000 or higher
    - Use GlassFish monitoring to find right value

EJB Container Tuning

- Tune EJB Container pool and cache
- Stateless Session Beans – Adjust Pool size
  - Default Minimum Pool Size = 8
  - Defaul Maximum Pool Size = 32
  - Default Pool Idle Timeout = 600 secs
  - Use GlassFish monitoring to find right values
- Stateful Session Beans – Adjust Cache size
  - Default Max Cache Size = 512
  - Default Removal Timeout = 60 mins
  - Default Cache Idle Timeout = 600 secs
  - Use GlassFish monitoring to find right values

High Availability Tuning

- GlassFish has in-built high availability feature
  - In-memory replication keeps copy of user session data in all GlassFish instances
  - Needs a cluster of 2 or more instances
- Tuning In-Memory Replication
  - Choice of Persistence Frequency
    - web-method – persist on a session activity
    - time-based – persist at regular interval
  - Choice of Persistence scope
    - 'modified attribute' - persists only attributes which are modified
    - 'modified session' - persists all session data but only when session is modifed
    - 'session' - persists all of session data for any session activity

Tuning Web Services & XML

- Recommended to use Web Container tunings
  - Good for most applications
- Woodstox parser – streaming parser that can outperform bundled SJSXP parser
  - -Djavax.xml.stream.XMLInputFactory=com.ctc.wstx.stax.WstxInputFactory
  - -Djavax.xml.stream.XMLOutputFactory=com.ctc.wstx.stax.WstxOutputFactory
- Fast Infoset – binary encoding for faster serialization and parsing
  - -Dcom.sun.xml.ws.client.ContentNegotiaton=optimistic

Presentation Agenda

- GlassFish Out-of-Box Performance
- How to Tune GlassFish
- Performance Tuning GlassFish
- Performance Best Practices
- Performance results
- GlassFish Performance Tuning References

Web Container Best Practices

- Consider low value for user session timeout
  - Default is 30 minutes
- Keep session reap interval small
- Disable Dynamic JSP Loading
- Disable Access Logging

EJB Container Best Practices

- JPA Best Practices
  - Lock Mode [Optimistic vs Pessimistic locking]
    - For Data integrity
  - Flush Mode
  - FetchType
  - NamedQuery

High Availability Best Practices

- Replication is memory intensive – size JVM properly (Java heap, garbage collection strategy, etc)
- Tune User Sessions
  - > Keep the session size as small as possible – write only what is needed
  - > Control frequency – store data in session just when needed
  - > Don't keep stale data – examine session expiration strategy

Web Services Best Practices

- Try to keep message size small
- Complex XML schema reduces performance
  - Check your XML data types - some data types are higher performing than others

General Tuning Tips

- Unused features could have a negative impact on the performance and should be disabled
  - Auto-deployment of applications
  - JMX Monitoring
  - JMS
  - Dynamic JSP reloading
  - JDBC Connection validation
- Security Manager could be turned off if the applications are all trusted internal applications

Presentation Agenda

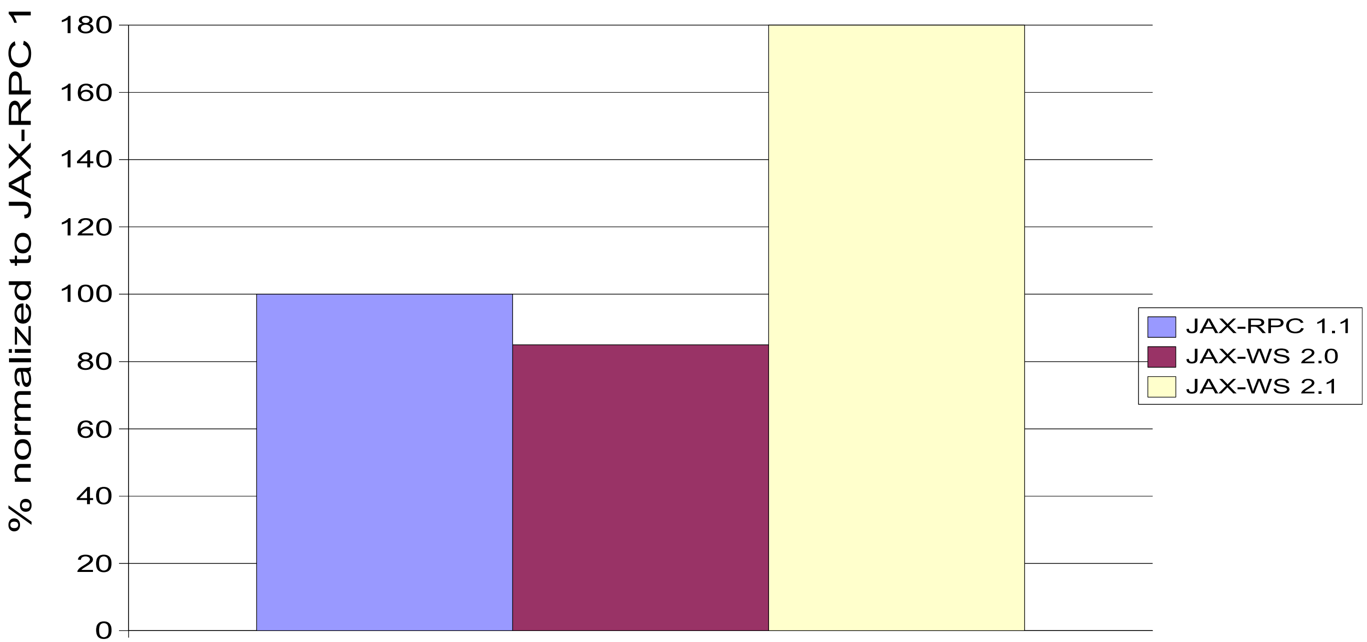
- GlassFish Out-of-Box Performance
- How to Tune GlassFish
- Performance Tuning GlassFish
- Performance Best Practices
- Performance results
- GlassFish Performance Tuning References

GlassFish SpecjAppServer2004 Performance Results

- Only Open Source Application Server to publish SpecjAppServer2004 numbers
- Fastest open source results
  - > 813.73 JOPS using PostgreSQL database
  - > 883.66 JOPS using DB2 database
- More info: http://www.spec.org/jAppServer2004/results/

GlassFish Web Services Performance Results

- Web Services results
  - Results using open source WSTest micro-benchmark
    - https://wstest.dev.java.net/
  - Major performance improvements in JAX-WS 2.1



GlassFish Performance Tuning References

- GlassFish Performance Tuning Guide
  http://wiki.glassfish.java.net/Wiki.jsp?page=PerformanceTuningGuide
- Blogs
  - Scott Oaks on overall Glassfish Performance
    http://weblogs.java.net/blog/sdo
  - Dave Dagastine on Java SE performance
    http://blogs.sun.com/dagastine
  - Arun Gupta on Web Services and Web 2.0
    http://blogs.sun.com/arungupta/
  - Java EE Blog http://blogs.sun.com/theaquarium
  - Many other blogs on http://blogs.sun.com provide performance tips for various Sun technologies

Q&A

- Further questions
  - Post your queries to forums on
    - http://glassfish.dev.java.net
    - http://performance.dev.java.net
  - Send them to me: deep_singh@dev.java.net

Tuning Your GlassFish – Performance Tips

Deep Singh
Enterprise Java Performance Team
Sun Microsystems, Inc.

Back-up slides

Basic JVM Tuning

- Garbage Collection
  - Serial collector for single processor machines and small heap
    - It is default garbage collector
  - Parallel collector for medium to large heaps and run on multiprocessor machines.
    - -XX:+UseParallelGC
  - CMS collector for short GC pauses, when response time is more critical
    - -XX:+UseConcMarkSweepGC
- Upgrade to latest JVM for better results

Tuning System Resources

- Monitor resource usage before tuning
  - > Unix based systems: mpstat, vmstat, netstat, iostat
- Operating System Tuning
  - > File Descriptors
  - > Shared Memory
- Network Performance Tuning
  - > TCP/IP tuning
  - > Network bandwidth
- Tune Disk IO

EJB Container Tuning

- Optimistic Concurrency allows simultaneous access to an ejb
  - If transactions do not modify the ejb, they all succeed
  - If one transaction changes the ejb, other transactions will fail and need to be retried
  - Good for EJBs that are rarely modified
- Request Partitioning allows to assign request priority to an EJB
  - Prioritized EJB requests execute in a separate thread pool

Web 2.0

- Use Web Container Tunings
- Resource Consumption Management (RCM)
  - reserve a specific percentage of request processing capability for a specific URL/service
  - Grizzly's Application Resources Allocation (ARA) extension:
    - Implementation of a RCM system
    - Enables virtualization of system resources per web application, similar to Solaris 10 zone or the outcome of the upcoming JSR 284.
    - Supported in Glassfish v3

General Tuning Tips

- Glassfish out of the box settings intended for development use
  - Must be tuned for production environments
- Proper JVM tuning greatly improves performance across the board
- Monitor Glassfish components through Admin Console or command line to get an idea of what needs to be tuned
- Use profilers such as NetBeans profiler to identify bottlenecks in your application