Azul Inspector
Java Environment Checker

For All Java Applications

SYSTEM REQUIREMENTS

- JDK 1.6 or 1.7
- Memory 128MB
- Disk 1MB
- Tool location either on the server or remote

SUPPORTED TARGET SYSTEMS

- Any JDK
- RHEL, CentOS, SLES, Ubuntu
- Hypervisor: ESX, KVM, Xen
- Amazon Cloud

Azul Inspector is a Java program designed to collect information about a target Java application and its server environment. Azul Inspector is a free tool originally developed by Azul Systems to check that the local environment meets the system requirements for the Zing™ JVM. Zing is a fully Java-compatible JVM based on Oracle's HotSpot and the only JVM that can support very large heaps with pauseless operation.

Developers, IT and performance engineers can use Azul Inspector to quickly determine the JDK version in use, maximum heap size setting and the values of a variety of other setup variables.

MODES

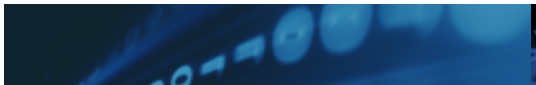
Azul Inspector has two main modes, Qualification and Monitoring. The tool can take just a snapshot of data or continue to collect it at a specific interval.

Qualification mode is a snapshot of data that can be used to determine if the server is compatible with the Zing JVM. This mode collects system level data about any Java application or JVM running on the server. The output contains the configuration settings to quickly set up the Zing JVM to run using the same parameters as the existing JVM.

Monitoring mode is data collected at specified intervals during testing to validate tuning settings and identify potential causes of poor scalability and performance or response time jitter. In Monitoring mode Azul Inspector collects and tags data for all Zing runtimes (and, as an option, all other JVMs) to identify trends and for graphing in Microsoft Excel®.

INFORMATION COLLECTED IN MONITORING MODE

- Runtime JVM metrics that describe the environment for the JVM
- Management beans for third-party software configuration and performance
- Linux resource utilization for the JVM, including CPU, network, file and memory metrics
  - If available, garbage collection metrics for the running JVM
- Hypervisor (ESX, KVM or Xen) metrics if authorized



OUTPUT AND USAGE

The output of Azul Inspector can be used to tune the Zing JVM, application and environment to reduce latency and remove the causes of inconsistent response times. In many ways the application and JVM are “victims” of their environment. Both can run only as fast as they can get access to resources or data from the Host System. These underlying performance issues often are hidden by long garbage collection pauses. Once Zing eliminates these pauses, other issues can be identified using Azul Inspector and addressed. Monitoring mode also has several parameters that can be set to achieve different purposes.

- **Latency mode** Runs Azul Inspector from a desktop. This reduces the Heisenberg effect for high performance or very low latency applications and collects Linux scheduling data for the JVM at high frequency.
- **Trend or TrendLocal mode** Collects metrics on CPU and JVM memory use to identify whether jitter is due to JVM garbage collection or some other event.
- **Watch** Watches existing management beans, for example the Average Processing Time mbeans often used by Tomcat or a caching system.

Azul Inspector’s post-processing provides a framework for tracking specific metric counts and correlating them to observed application events. Below are some example output graphs.

USING WITH jHICCUP

Azul Inspector is also complementary to the jHiccup open source performance measurement tool. jHiccup is used to find hiccups or variations in application response time. Azul Inspector gathers information about the JVM, Operating System and application environment to help identify factors that may have contributed to the hiccup. Together the two tools are very useful for lowering maximum latency and eliminating the sources of response time variation.

INSTALLING AND RUNNING

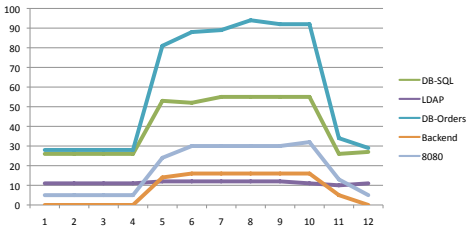
Simply download and unzip the file, then run ‘runMe.sh’ to operate in Qualification mode. For Monitoring mode, you need only replace one parameter in the script. See the ReadMe file for more details.

GET AZUL INSPECTOR

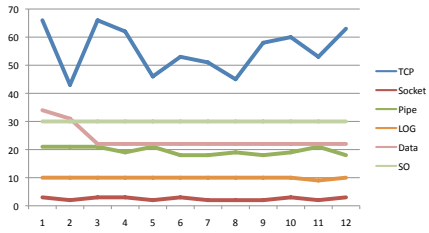
Free download on the web:
www.azulsystems.com/azul_inspector

Email us and request a copy:
info@azulsystems.com

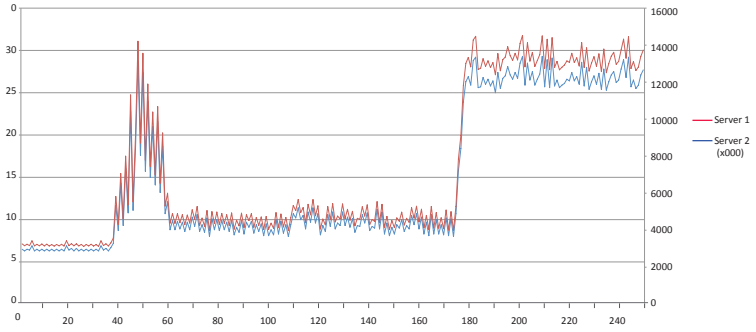
Or call: +1 650.230.6500



Tracking TCP port usage to the database during a load test



File handles by type, used to check for handle leaks and that file reuse is consistent with the design



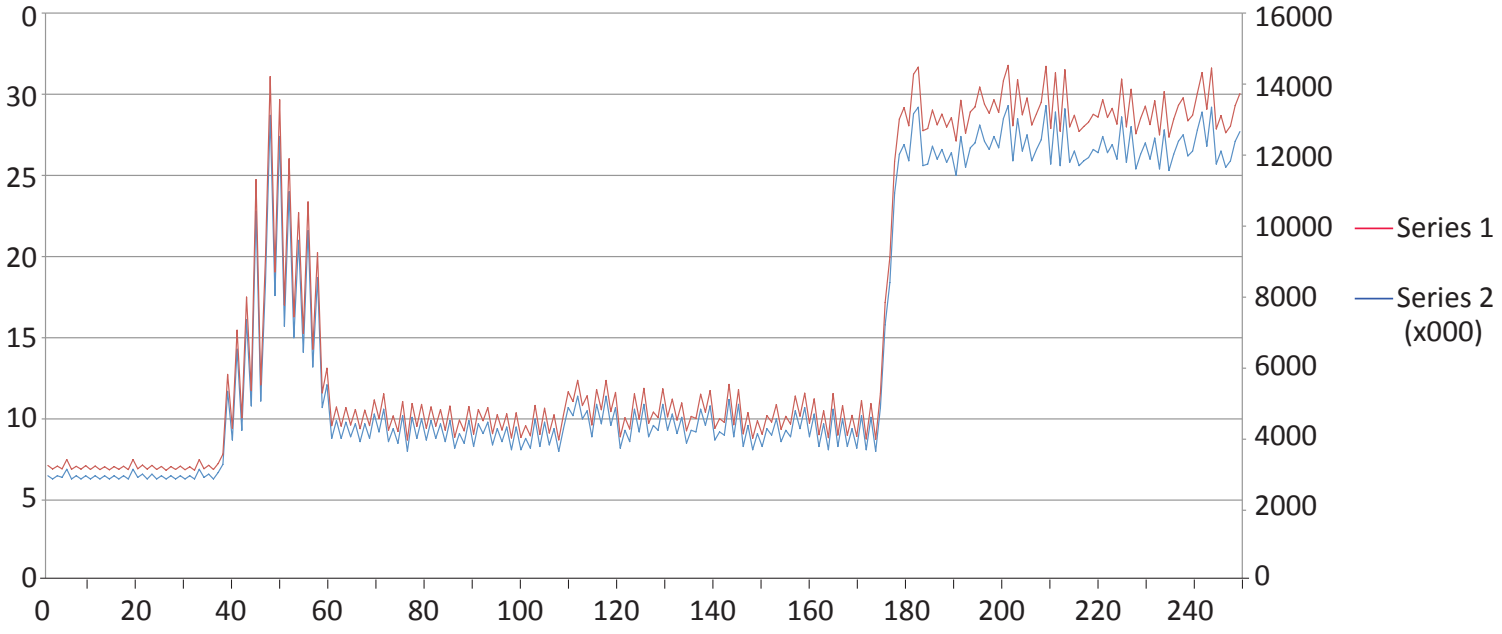
Correlating Linux memory growth to establish root cause

Contact Azul Systems:

1173 Borregas Avenue
Sunnyvale, CA
94089 USA

T + 1.650.230.6500
F + 1.650.230.6600

www.azulsystems.com/azul_inspector


0
30
25
20
15
10
5
0
16000
14000
12000
10000
8000
6000
4000
2000
0
0
20
40
60
80
100
120
140
160
180
200
220
240
Series 1
Series 2
(x000)