

# **SAS® 9.4 Web Application Performance: Monitoring, Tuning, Scaling, and Troubleshooting**

Rob Sioss, SAS Institute Inc., Cary, NC

## **ABSTRACT**

SAS® 9.4 introduces several new software products to better support SAS® web applications. These products include SAS® Web Server, SAS® Web Application Server (with the availability of out-of-the-box clustering), and SAS® Environment Manager. Even though these products have been tuned and tested for SAS 9.4 web applications, advanced users might want to know the tools and techniques that they can use to further monitor, manage, tune, and improve the performance of their environment. This paper discusses how customers can achieve that by exploring the following concepts, activities, techniques, and tools:

- using SAS Environment Manager to monitor run-time performance of middle-tier components
- using additional tools to monitor middle-tier components (Apache server-status, Java VisualVM, Java command-line tools, Java GC logging)
- identifying the potential bottlenecks and tuning suggestions
- identifying appropriate clustering strategy (single-server vs. multi-server for homogenous or heterogeneous clustering)
- suggesting the data to collect when analyzing performance (GC data, thread dumps, heapdumps, system resource utilization information, log files)
- discussing in-depth performance analysis tools (Thread Dump Analyzer, HPjmeter, Eclipse Memory Analyzer (MAT), IBM Support Assistant tools: GC and Memory Visualizer, Memory Analyzer, Thread, and Monitor Dump Analyzer)

## **INTRODUCTION**

There are many factors that can impact the overall performance of web applications and therefore dictate the resource requirements for various components of the environment. Some of the relevant factors include the number of active concurrent users, the usage patterns of the different applications in the deployment, common workflows within the applications, and dependencies on external resources. The out-of-the-box configuration of the SAS® web applications is intended to be a decent starting point, but will often require tuning to match the demands of the environment. While the opportunities to tune various components are seemingly endless, collecting the appropriate monitoring data and following a few general guidelines can greatly improve the performance of the applications. As the needs of the environment change over time (due to additional users, the adoption of different workflows and usage patterns, and so on), the systems can continually be monitored, tuned, and scaled in order to maintain optimal performance.

## **BEFORE GETTING STARTED**

Prior to configuring the SAS® 9.4 environment, there are a number of things to consider in order to determine an appropriate starting point for your deployment. Do not be alarmed if you have already configured your environment—the tools and guidelines presented in this paper are applicable to existing SAS 9.4 web application environments as well as newly configured environments. However, if you have not yet gone through the process of setting up your SAS 9.4 environment, here are some tips to keep in mind:

- The SAS Enterprise Excellence Center (EEC) is available to perform a sizing analysis in order to recommend a suitable topology based on the predicted needs for the environment.
- The *SAS 9.4 Web Applications: Tuning for Performance and Scalability* document provides generic tuning recommendations that provide an initial set of tuning suggestions. Combining the initial tuning recommendations with the techniques and tools outlined in this paper can help you determine any modifications that might be necessary for your environment.
- Historically, some customers have chosen to cluster applications in a heterogeneous fashion (a different number of cluster nodes for a particular application in contrast to the number of nodes for other applications). This can be useful when a particular application or set of applications are more heavily utilized than other applications.

Additional capacity can be added for the applications that require it. Due to the implementation of out-of-the-box clustering support for SAS 9.4, there are some key points to highlight regarding heterogeneous clustering:

1. Configuring a heterogeneous cluster requires that the web application server be configured with the multiple server option selected at configuration time. Some products cause this option to be selected by default, but for other products, the option must be selected manually by going through the custom path of prompts in the SAS Deployment Wizard.
2. When configuring clustering with the multiple server option selected, each server instance can be clustered independently. Each server instance is associated with a specific set of applications, and clustering in this approach enables the addition of nodes for that set of applications independently from applications designated for other server instances.

## MONITORING AND COLLECTING THE DATA

In order to identify potential improvements in the environment, the appropriate resources must be monitored and data must be collected about the utilization of these resources. While it is true that there are an overwhelming number of tunable properties, settings, and options, the good news is that there are a few key concepts that can be applied to solve the majority of the issues that impact the performance of the SAS web applications.

### WHAT TO MONITOR

When analyzing the performance of the SAS web applications, there are some key resources to track. These resources can frequently become the limiting factor to improved performance if not properly tuned for the needs of the environment. This list is not an exhaustive list of all factors that impact performance, but they do represent the items that most often require tuning in order to make improvements. In addition, the guidelines presented are not hard and fast rules. They are general recommendations that apply to the majority of scenarios, but the needs of a given deployment might dictate deviation from the suggestions. Steps for tuning these settings are discussed later in the section “Tuning and Scaling the Environment.”

#### Java Virtual Machine (JVM) Resources

- JVM heap space\*: recommend 500MB peak free heap space (space after garbage collection (GC))
- JVM permanent generation (PermGen)\* space (HotSpot only): recommend 200–300MB free PermGen space

\* For HotSpot JVMs, set initial and maximum sizes equally to avoid overhead of expansion, as discussed in *SAS 9.4 Web Applications: Tuning for Performance and Scalability*.

#### SAS® Web Application Server Resources

- tomcatThreadPool Executor thread pool: increase if number of active threads approaches 5–10% of maximum
- Java Database Connectivity (JDBC) resources:

There are JDBC resources configured for the data sources that the SAS web applications need to access. In some cases, the workload on the system may benefit from a larger pool of connections for a particular data source. The most commonly used data source is the SharedServices data source. The presence of the following error message in any of the SAS web application log files (found under `Web/Logs/<server instance>/`) normally indicates that the size of the connection pool should be increased:

```
com.atomikos.jdbc.AtomikosSQLException: Connection pool exhausted - try
increasing 'maxPoolSize' and/or 'borrowConnectionTimeout' on the DataSourceBean.
```

#### SAS® Web Infrastructure Platform Data Server Connections

- max\_connections:

Since there might be a number of JDBC resources configured across one or more SAS Web Application Server instances, it is possible that the SAS Web Infrastructure Platform Data Server might need to be configured to allow more total connections. The number of allowed connections on the data server should be greater than the sum of the maxPoolSize values of all JDBC resources in all of the SAS Web Application Server instances. Other processes (such as SAS® Environment Manager) will consume connections as well, so the limit needs to be high enough to accommodate all incoming connections. The following error message in the SAS web application log files normally indicates that the SAS Web Infrastructure Platform Data Server has run out of connections:

```
Could not get JDBC Connection; nested exception is
com.atomikos.jdbc.AtomikosSQLException: Failed to grow the connection pool
```

## SAS® Web Server Thread Utilization

- ThreadsPerChild (Windows) or MaxClients and ServerLimit (UNIX, Linux):

By default the SAS Web Server is configured to support the processing of 1,024 incoming requests at a given time. If a message similar to the example below (captured from a Windows system) is logged to the SAS Web Server's error log (`WebServer/logs/error.log`), the limit likely needs to be increased:

```
[warn] Server ran out of threads to serve requests. Consider raising the  
ThreadsPerChild setting
```

## Operating System (OS) Resources

The following OS resources should be monitored, and capacity increased if necessary—either by upgrading/adding hardware, or even adding additional systems and configuring additional middle-tier nodes (clustering) when needed:

- CPU resource utilization
- Memory resource utilization
- I/O resource utilization
- Network interface resource utilization

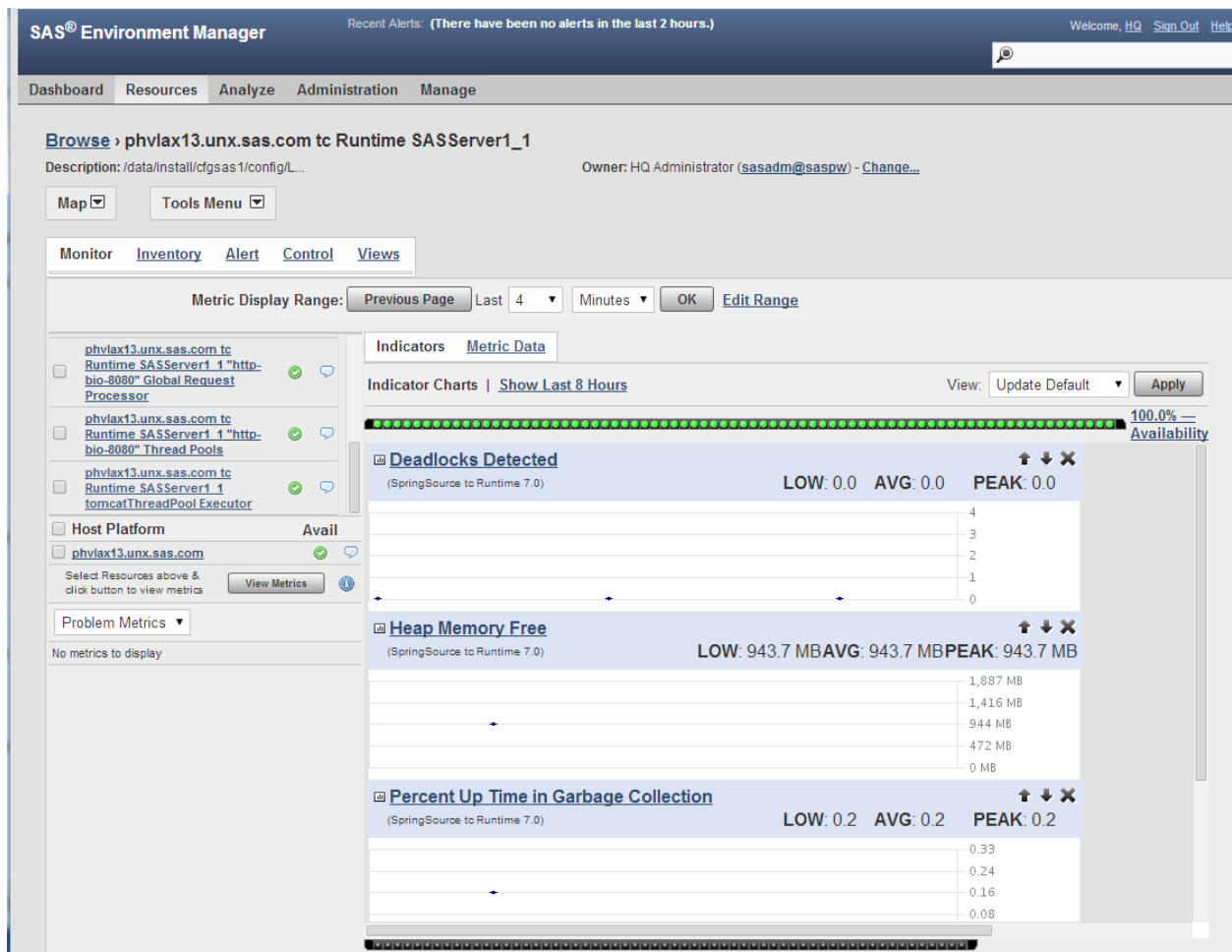
## MONITORING TOOLS

There are a number of different monitoring tools that can be used to collect the suggested data. As with the list of metrics to monitor, this is not an exhaustive list of monitoring tools, but it does include tools that can be used to collect the data needed to diagnose the majority of performance issues that can occur for the SAS web applications.

### SAS Environment Manager

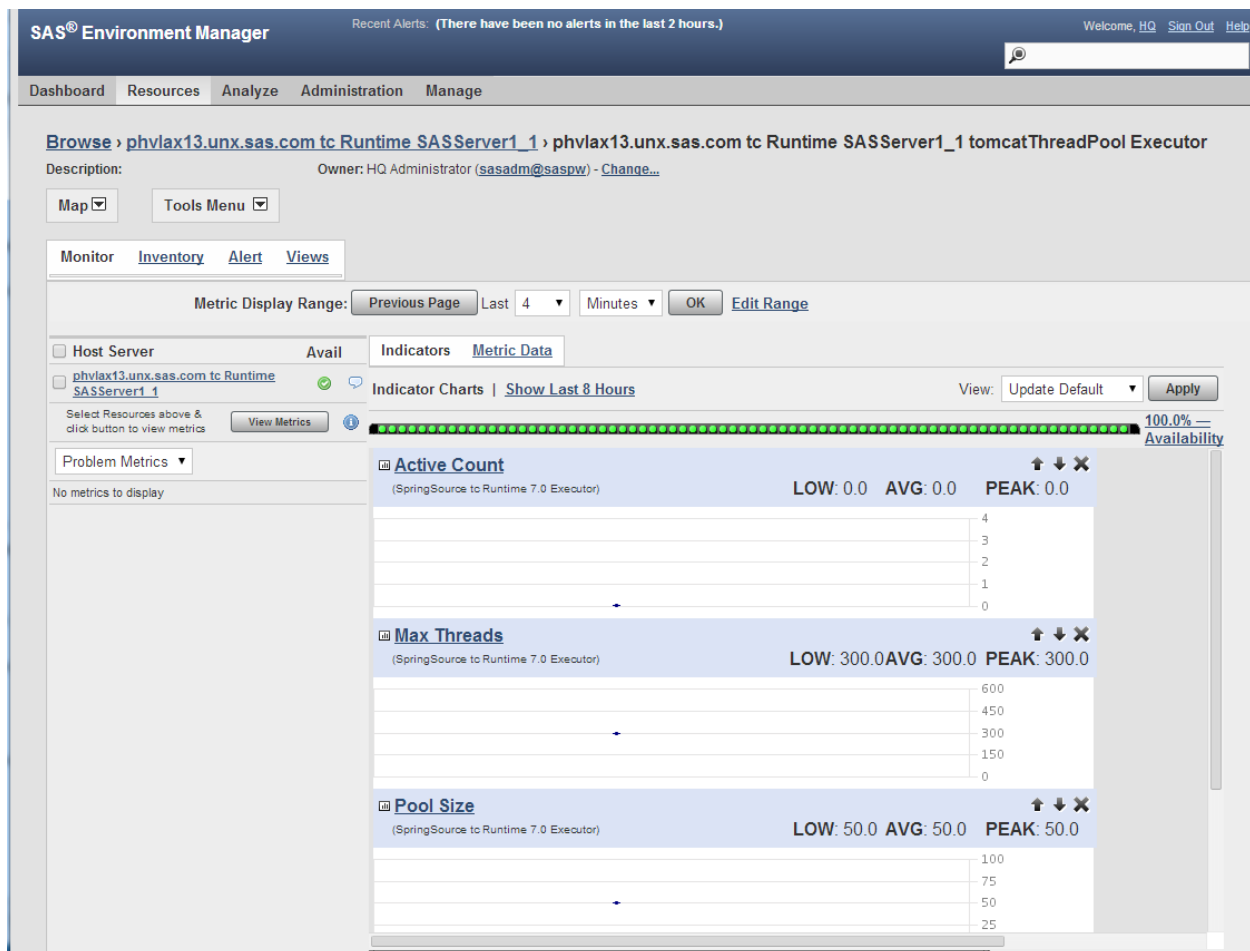
The SAS Environment Manager has been introduced with the SAS 9.4 release, and is an integrated monitoring solution for the entire SAS deployment. There are a few metrics listed above that are not available in SAS Environment Manager, such as PermGen space; however, the tool is included with most SAS deployments and is ready to monitor the environment immediately after deployment is complete. Among the metrics that can be tracked in SAS Environment Manager are OS resource metrics, JVM heap metrics, and SAS Web Application Server thread pool metrics. For more information about SAS Environment Manager, please refer to *SAS Environment Manager 2.1: User's Guide, Second Edition*.

To monitor SAS Web Application Server resources for a particular server instance, simply navigate to the server instance in the SAS Environment Manager inventory for the environment. Optionally, additional data can be viewed by clicking on **Metric Data** above the indicator charts.



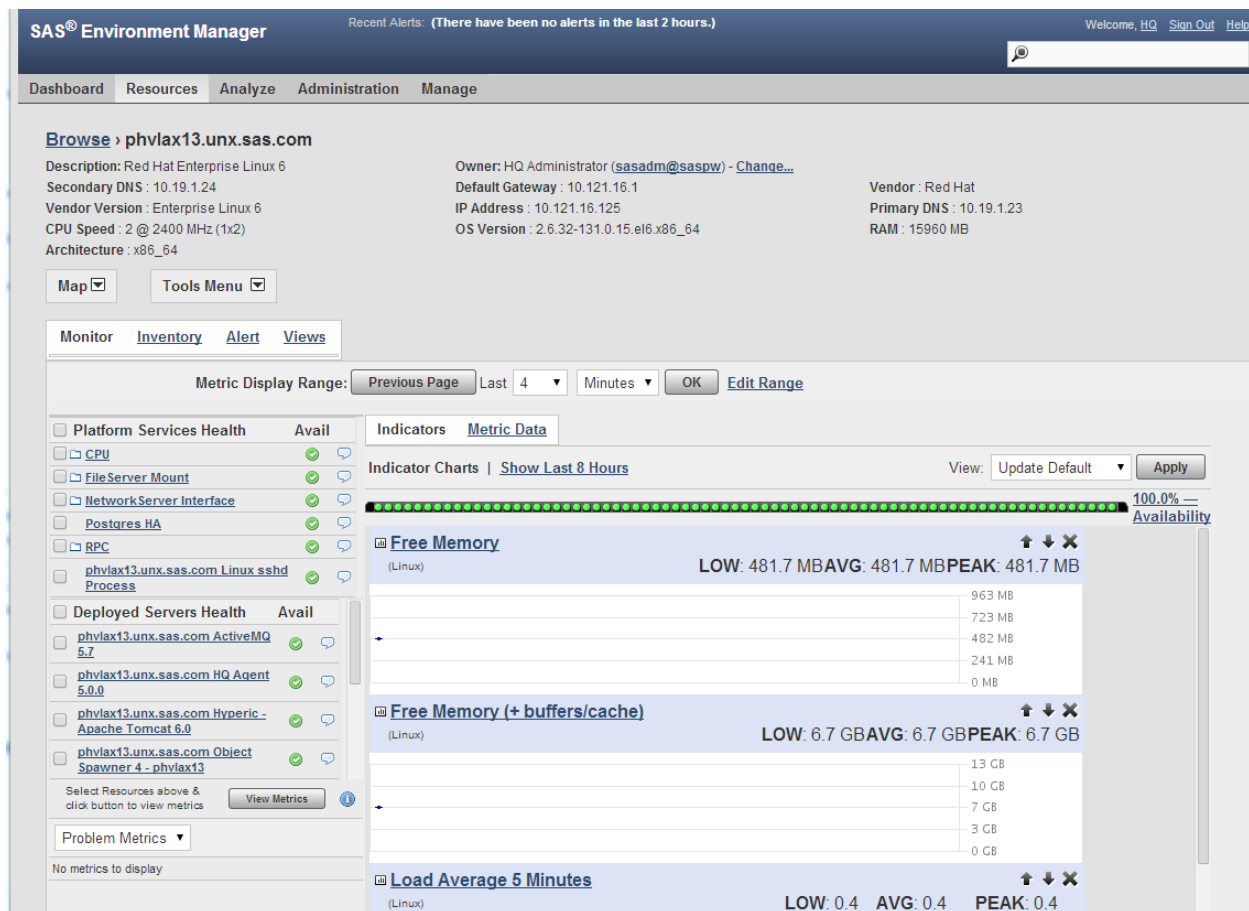
**Figure 1 – SAS Environment Manager Monitoring Data: SAS Web Application Server Resources, Including Heap**

The tomcatThreadPool Executor thread pool is considered a service of the SAS Web Application Server instance. Clicking the thread pool service in the list of services associated with the instance will display the metrics for the pool.



**Figure 2 – SAS Environment Manager Monitoring Data: SAS Web Application Server Executor Thread Pool Resources**

Lastly, each host platform system in the SAS Environment Manager inventory can be monitored for potential system resource utilization issues.



**Figure 3 – SAS Environment Manager Monitoring Data: Host Platform Operating System Resources**

### Oracle HotSpot Java Tools and Features (All Platforms except AIX)

The SAS 9.4 Web Application Server (along with other SAS products) use the SAS Private Java Runtime Environment (JRE) to run Java code. One limitation of the JRE is that it does not include some of the tools that are shipped with a full-fledged Java Development Kit (JDK). For some of the tools listed in this section, the full JDK must be downloaded from Oracle directly. For more information about these tools, please refer to the Oracle Java SE documentation. Links to some of the documentation are included in the "References" section.

### Java VisualVM (Requires JDK Download, Java 7 Update 15 or Higher)

The Java VisualVM utility from Oracle is another powerful monitoring tool. Using VisualVM, you can obtain statistics from a running Java process including heap utilization, PermGen utilization, overall thread count (and thread execution stacks), and much more. The tool also provides the capability to capture thread dumps and heap dumps for additional analysis. In addition to the base VisualVM product, there are a number of plugins that can be downloaded that extend the functionality. One such plugin is an MBean viewer, which provides a tree view of the various Java Management Extension (JMX) MBeans that are available for a Java process.

To connect to a SAS Web Application Server instance, you can create a JMX connection profile for the server. By default, the SAS Web Application Server processes are configured to only listen for incoming JMX connections on localhost (127.0.0.1), so it is simplest to run VisualVM from the same machine as the server instance being monitored. To allow monitoring from additional hosts, the JMX listener must be modified to listen on an address that is accessible from the client machine. Modify the following Listener element in the <server instance>/conf/server.xml file for each instance you wish to connect to remotely, modifying the bind attribute as needed (0.0.0.0 will listen on all addresses):

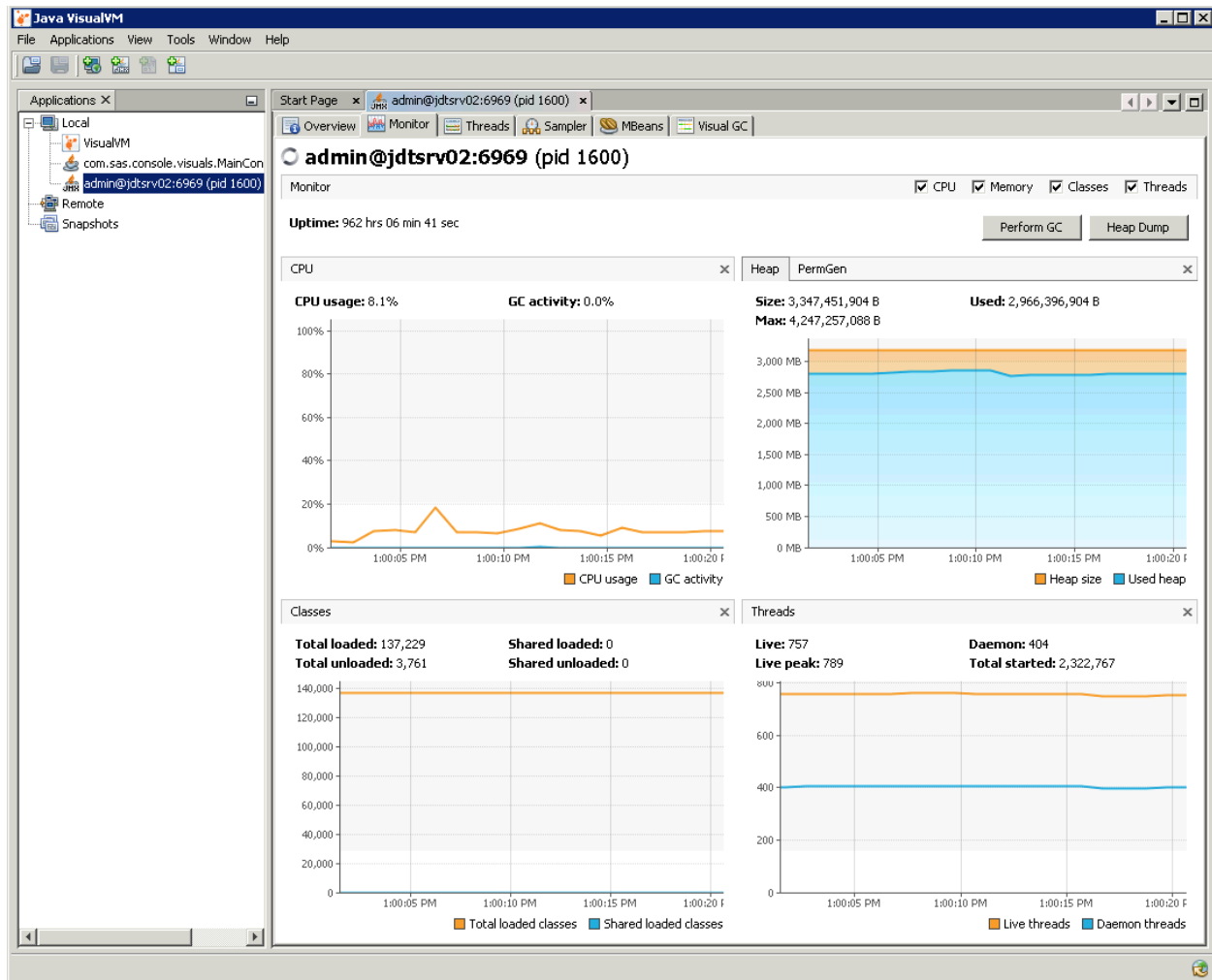
```
<Listener accessFile="${catalina.base}/conf/jmxremote.access" authenticate="true"
bind="0.0.0.0"
className="com.springsource.tcserver.serviceability.rmi.JmxSocketListener"
```

```
passwordFile="${catalina.base}/conf/jmxremote.password" port="${base.jmx.port}"
useSSL="false"/>
```

After launching VisualVM, create the connection by doing the following:

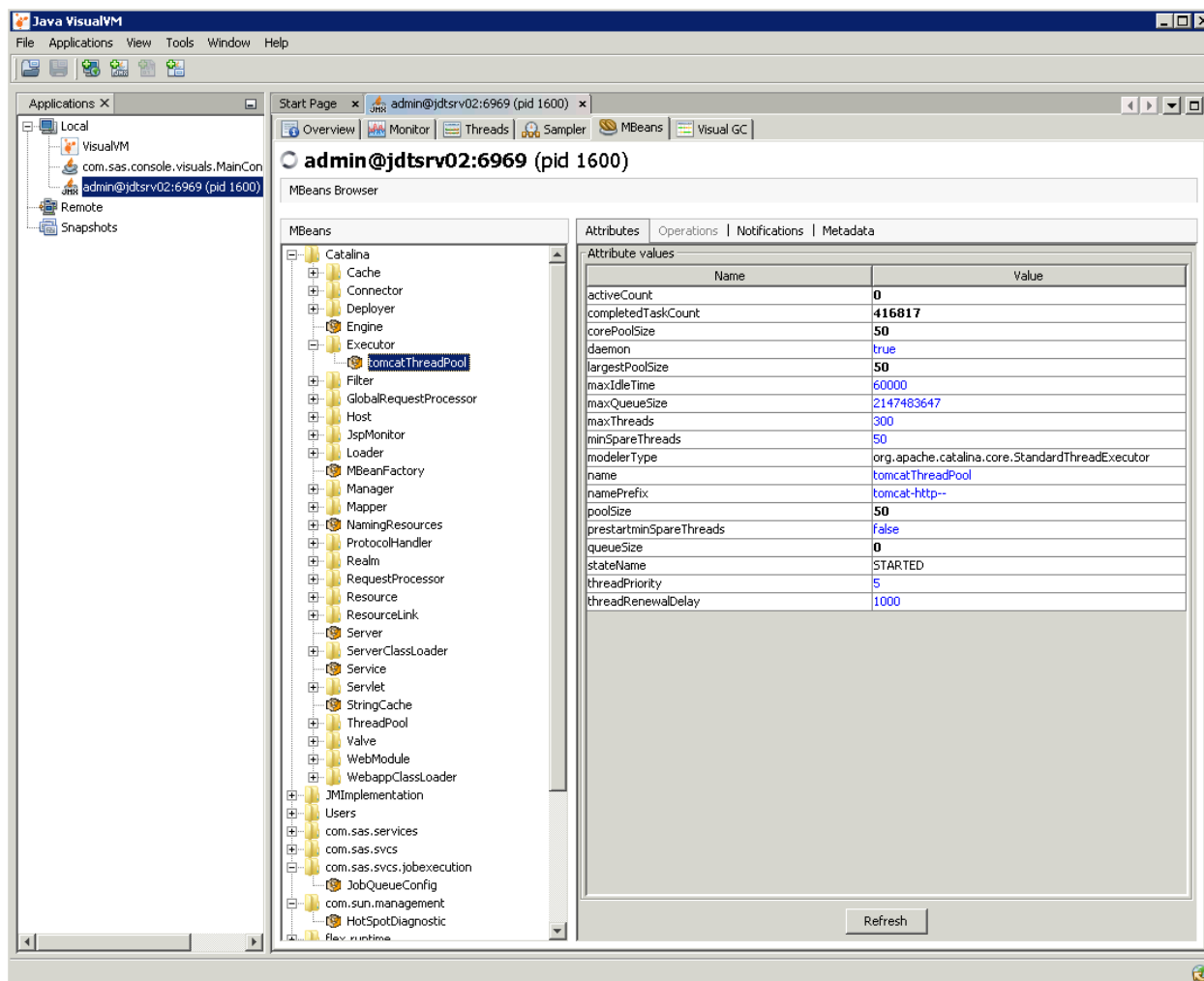
1. Select **File -> Add JMX Connection**.
2. Enter the host and port (each server instance's port is specified via the `base.jmx.port` property in the `<server instance>/conf/catalina.properties` file).
3. Select the **Use security credentials** option.
4. Enter the JMX user name and password (found in the `<server instance>/conf/jmxremote.password` file).

Once connected to the running process, the Monitor tab within VisualVM will display a graphical overview of several key resources. The Heap graph can also be toggled to display data about PermGen as well.



**Figure 4 – Java VisualVM Monitoring Data: JVM Resources, Including CPU, Heap, Threads, and Classes**

If desired, additional plugins can be downloaded and added to VisualVM to extend the functionality by going into the Tools -> Plugins menu. The VisualVM-MBeans plugin provides a tree view of the available JMX MBeans.



**Figure 5 – Java VisualVM Monitoring Data: MBean Viewer Plugin Showing tomcatThreadPool MBean Data**

### Verbose GC Logging

Another mechanism for monitoring the Java heap is to enable GC logging for the server instances. To do so, add the following arguments to the JVM options for each server instance to be monitored (substituting a unique, valid path for the log for each instance):

```
-verbose:gc -XX:+PrintGCDetails -XX:+PrintGCTimeStamps -XX:+PrintHeapAtGC -Xloggc:/path/to/filename.log
```

The process for modifying the JVM options is detailed in *SAS 9.4 Web Applications: Tuning for Performance and Scalability*, Chapter 4. The GC log data can be analyzed and displayed graphically using a tool such as HPjmeter, as discussed below in the section "Advanced Troubleshooting and In-Depth Analysis."

### Java Command Line Utilities

There are additional Java command line utilities that are shipped with the JDK that can be used to collect other types of data from the running process. These utilities require the ability to attach to the process, so they are most useful in UNIX or Linux environments and Windows environments when the server is executed from a script. When a server is deployed on Windows as a service, these utilities will not work.

The jmap utility (requires JDK download, Java 7 Update 15 only) can be used to capture statistics about running Java processes, including heap configuration and heap usage. To obtain this data, run the following command:

```
jmap -heap <pid>
```

In addition, jmap can be used to capture a heap dump, which is a snapshot of all Java objects in the heap of the



running process. This is useful for analyzing a suspected memory leak. To capture the heap dump, run the following command (substituting a valid file path with the .hprof extension):

```
jmap -dump:format=b,file=/path/to/file.hprof <pid>
```

The jstack utility (requires JDK download, Java 7 Update 15 or higher) can be used to capture a thread dump, which is a snapshot of the execution stack of all threads in the running process. This is useful for troubleshooting a variety of performance issues, such as slowdowns, hangs, or unresponsiveness of a process. To capture the thread dump, run the following (substituting a valid path to redirect the output to):

```
jstack -F <pid> > /path/to/threaddump.txt
```

Alternatively, a thread dump can be captured on UNIX or Linux by running the following command (which does not require a JDK), and the output will be sent to the <server instance>/logs/server.log file:

```
kill -QUIT <pid>
```

The heap dumps and thread dumps can be analyzed using tools such as Eclipse Memory Analyzer (MAT) and Thread Dump Analyzer (TDA), as discussed later in the section "Advanced Troubleshooting and In-Depth Analysis."

### IBM Java Tools and Features (AIX Platform)

Additional JVM arguments can be added to the JVM options for the IBM JVM on AIX in order to capture the data discussed in this section. The process for modifying the JVM options is detailed in *SAS 9.4 Web Applications: Tuning for Performance and Scalability*, Chapter 4.

As with the Oracle JVM, enabling GC logging is another mechanism for monitoring the Java heap of the server instances. To do so, add the following arguments to the JVM options (substituting a unique, valid path for the log for each instance):

```
-verbose:gc -Xverbosegclog:/path/to/filename.log
```

The GC log data can be analyzed and displayed graphically using a tool such as the IBM Garbage Collection and Memory Visualizer (GCMV), as discussed later in the section "Advanced Troubleshooting and In-Depth Analysis."

The IBM JVM also has the ability of generating heap dumps and javacore files (thread dumps) from a running process. To enable the generation of heap dumps on demand, add the following JVM option:

```
-Xdump:heap:file=/path/to/heapdump.phd:events=user
```

To enable the generation of javacores on demand, add the following JVM option:

```
-Xdump:java:file=/path/to/javacore.txt:events=user
```

If desired, these arguments can both be added together in order to produce both types of artifacts on demand. After adding these arguments, run the following command to trigger the generation of the snapshots:

```
kill -QUIT <pid>
```

For additional information about these tools, please refer to the *AIX User Guide for IBM SDK, Java Technology Edition, Version 7 Release 1*. A link to the documentation has been included in the "References" section below.

### Apache mod\_status Module

The Apache Status module, mod\_status, can be used to monitor various aspects of the state of the SAS Web Server including the number of idle worker threads, requests that are being processed, the status of the connections to the SAS Web Application Server instances, the number of requests sent to a given server instance, and much more. By default, the status information is accessible only from a client running on the same system as the SAS Web Server (localhost). To enable other clients to view the status information, you can modify the `WebServer/conf/extra/httpd-info.conf` file to add additional domains, subdomains, hosts, and IP addresses to access the /server-status page:

```
<Location /server-status>
    SetHandler server-status
    Order deny,allow
    Deny from all
```

```
Allow from 127.0.0.1 subdomain.domain.com host.subdomain.domain.com
</Location>
```

To view the mod\_status data, simply point a browser running from a system that is allowed access to the /server-status location to the host and port of the SAS Web Server with a request path of /server-status: `http://hostname.domain.com:port/server-status`. Details about the status of the different workers that point to SAS Web Application Server instances can be found near the bottom of the server-status page as shown below in Figure 6. For additional information about the Apache mod\_status data, please refer to the Apache mod\_status module documentation. A link to the documentation can be found in the "References" section below.

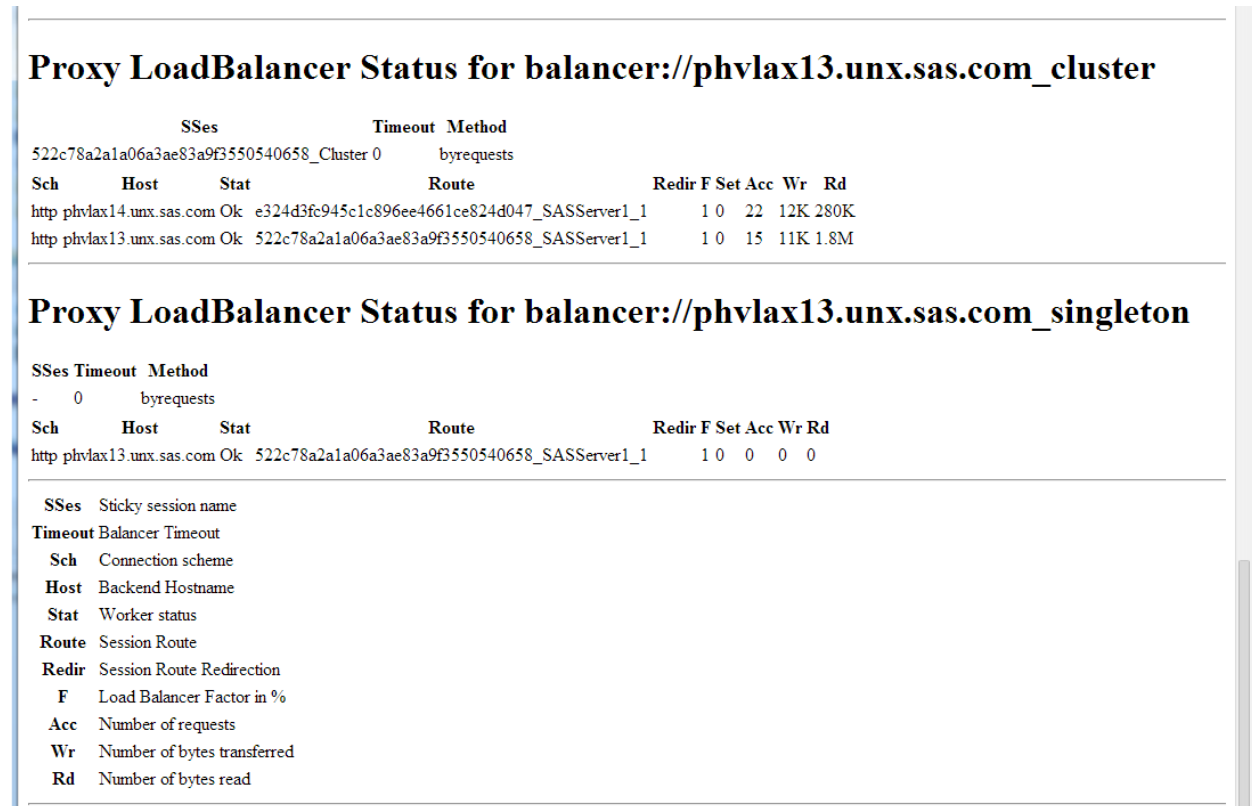


Figure 6 – Apache mod\_status Module Monitoring Data

## TUNING AND SCALING THE ENVIRONMENT

After monitoring, collecting data, and analyzing resource utilization trends, it is likely that the knowledge gained will suggest potential tuning or scaling changes that can be made to improve performance. Some general tuning guidelines were presented above in the section "What to Monitor," but once again those are just suggested starting points as the needs of each environment will vary. Use these guidelines in conjunction with the data that has been collected in order to determine what values would be appropriate for the needs of the environment being tuned. The process for modifying many of the tunable parameters is detailed in the *SAS 9.4 Web Applications: Tuning for Performance and Scalability* document. To tune the following items, please refer to the instructions in that guide:

- tomcatThreadPool Executor thread pool (Chapter 3)
- JDBC data source maxPoolSize (Chapter 3)
- JVM Heap space (Chapter 4)
- JVM PermGen (Chapter 4)
- SAS Web Infrastructure Platform Data Server (Chapter 6)

If the data collected suggests that the SAS Web Server has run out of threads (or that the number of active requests + idle threads is approaching the maximum number of possible requests for the configuration), the limit likely needs to be increased. To accomplish this, modify the appropriate section of the `WebServer/conf/extra/httpd-mpm.conf` file, choosing sufficient values for the italicized parameters:

```
# WINDOWS SYSTEMS
<IfModule mpm_winnt_module>
    ThreadsPerChild    1280
    MaxRequestsPerChild 0
</IfModule>

# ALL OTHER SYSTEMS
<IfModule mpm_worker_module>
    ServerLimit        40
    StartServers        2
    MaxClients          1280
    MinSpareThreads     25
    MaxSpareThreads     75
    ThreadsPerChild     32
    MaxRequestsPerChild 0
</IfModule>

#NOTE: for mpm_worker_module MaxClients must be less than or equal to ServerLimit *
ThreadsPerChild
```

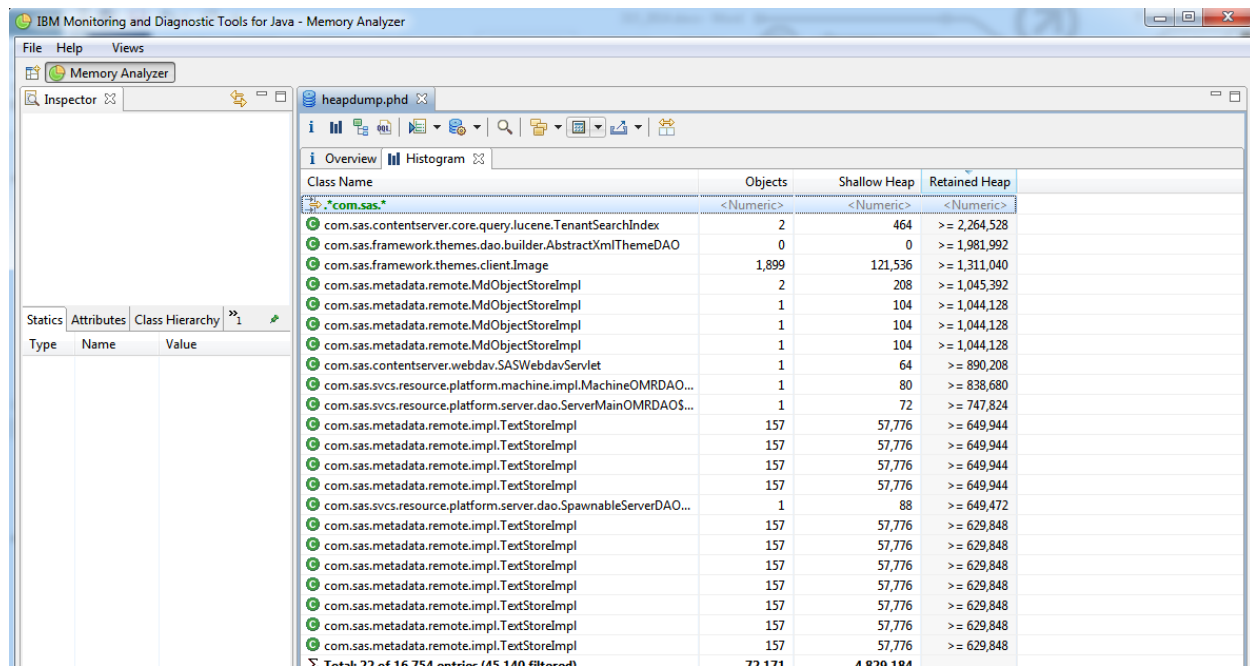
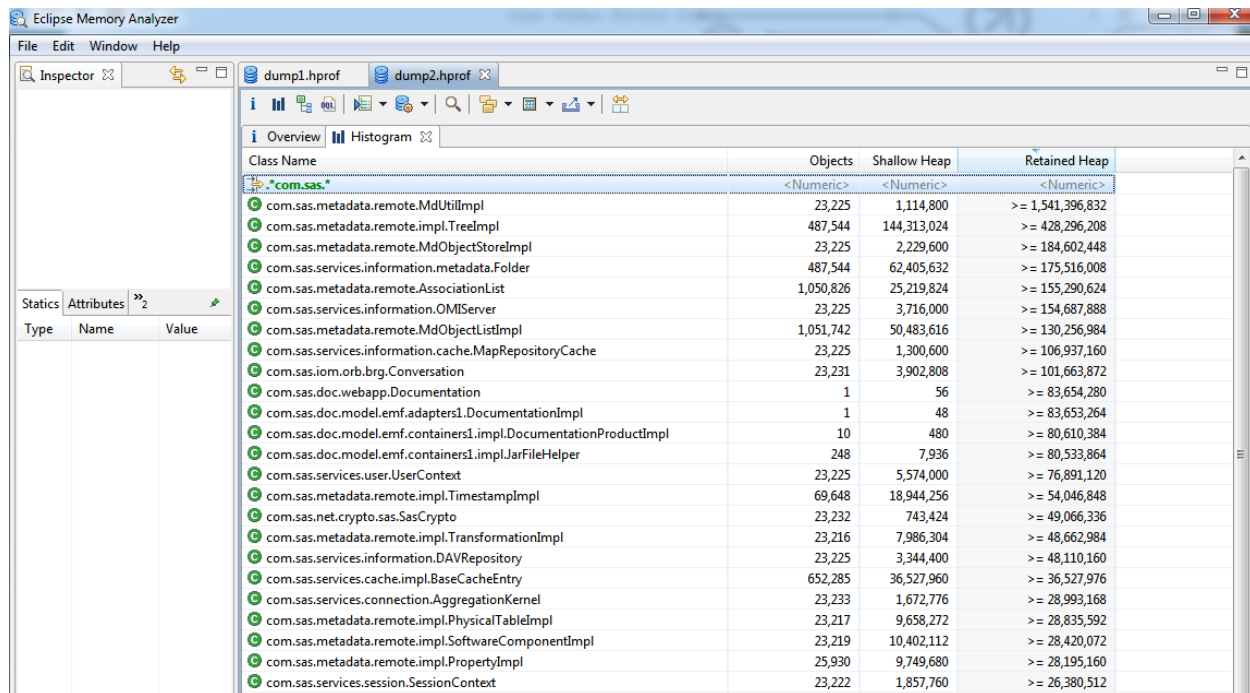
If system resource utilization for the OS resources indicates that a resource is being exhausted, the consideration must be made whether to upgrade the physical hardware to add more capacity for the strained resource, or to add cluster nodes on additional hardware. The process of adding cluster nodes is documented in the *SAS 9.4 Intelligence Platform: Middle-Tier Administration Guide*.

## ADVANCED TROUBLESHOOTING AND IN-DEPTH ANALYSIS

The SAS web applications are subjected to thorough testing before they are released, and the goal is to catch any potential performance problems with our internal testing processes. However, in some rare circumstances, a unique use case or environmental factor can uncover a previously unknown issue. In the event that a situation like this occurs, SAS Technical Support should be engaged, and in turn the appropriate SAS Research and Delivery personnel will work to diagnose the underlying cause. As a part of this process, these are some of the tools that are available to examine the data that has been collected. Note that the tools are not covered in detail in this document, but rather are being shown for informational purposes as SAS experts are available to perform the analysis.

### ANALYZING JAVA HEAP DUMPS

When monitoring data suggests a potential memory leak (such as a diminishing amount of Java free heap space over the life of a server process), heap dumps are typically required in order to pinpoint the source of the leak. Once heap dumps have been collected (for example, via jmap or VisualVM for HotSpot JVMs / -Xdump and kill -QUIT for IBM JVMs), they must be parsed and analyzed. Two commonly used tools for this activity are Eclipse Memory Analyzer for HotSpot heap dumps, and IBM Memory Analyzer (an IBM Support Assistant Tool) for IBM heap dumps. The interfaces and capabilities provided by the tools are very similar. It is often useful to examine the object histogram for a given heap dump—comparing object counts, shallow heap (memory consumed by an object or set of objects of a particular type), and retained heap (the sum of the shallow heap of all objects in the reference chain for an object or set of objects of a particular type). Multiple heap dumps can also be compared against one another to analyze object growth. A sample screenshot of Eclipse Memory Analyzer is shown below in Figure 7, and a sample screenshot of IBM Memory Analyzer is shown in Figure 8.



## ANALYZING JAVA THREAD DUMPS

Thread dumps are often useful for analyzing a variety of general performance issues, such as slowdowns, hangs, or unresponsiveness of a process or application. Thread dumps capture the execution stack and lock/monitor data of all threads within the Java process. Just like with heap dumps, once thread dumps have been collected (for example, via jstack or VisualVM for HotSpot JVMs / -Xdump and kill -QUIT for IBM JVMs), they must be parsed and analyzed. Two commonly used tools for this activity are Thread Dump Analyzer for HotSpot heap dumps, and IBM Thread and

Monitor Dump Analyzer (an IBM Support Assistant Tool) for IBM heap dumps. A sample screenshot of Thread Dump Analyzer is shown below in Figure 9. A similar screenshot of IBM Thread Dump and Monitor Analyzer is shown in Figure 10.

The screenshot shows the Thread Dump Analyzer (TDA) application window. The title bar reads "TDA - Thread Dumps of C:\Users\vrosio\Desktop\threaddump.txt". The interface includes a menu bar (File, View, Tools, Help) and a toolbar. On the left, a tree view shows the dump structure: "Dump No. 1 at line 2 around 2014-02-14 08:19:46", "Threads (119 Threads overall)", "Threads waiting for Monitors (2 Threads waiting)", "Threads sleeping on Monitors (90 Threads sleeping)", "Threads locking Monitors (22 Threads locking)", "Monitors (122 Monitors)", and "Monitors without locking thread (63 Monitors)". The main pane displays a table of threads:

Name	Thread-ID	State
Thread-58	125	BLOCKED
localhost-startStop-2	33	BLOCKED

Below the table, the details for "Thread-58" are shown, including the thread name "Thread t8125", the state "BLOCKED", and a detailed stack trace starting with "java.lang.Thread.State: BLOCKED".

Figure 9 – Thread Dump Analyzer Viewing Thread and Monitor Details from a HotSpot JVM

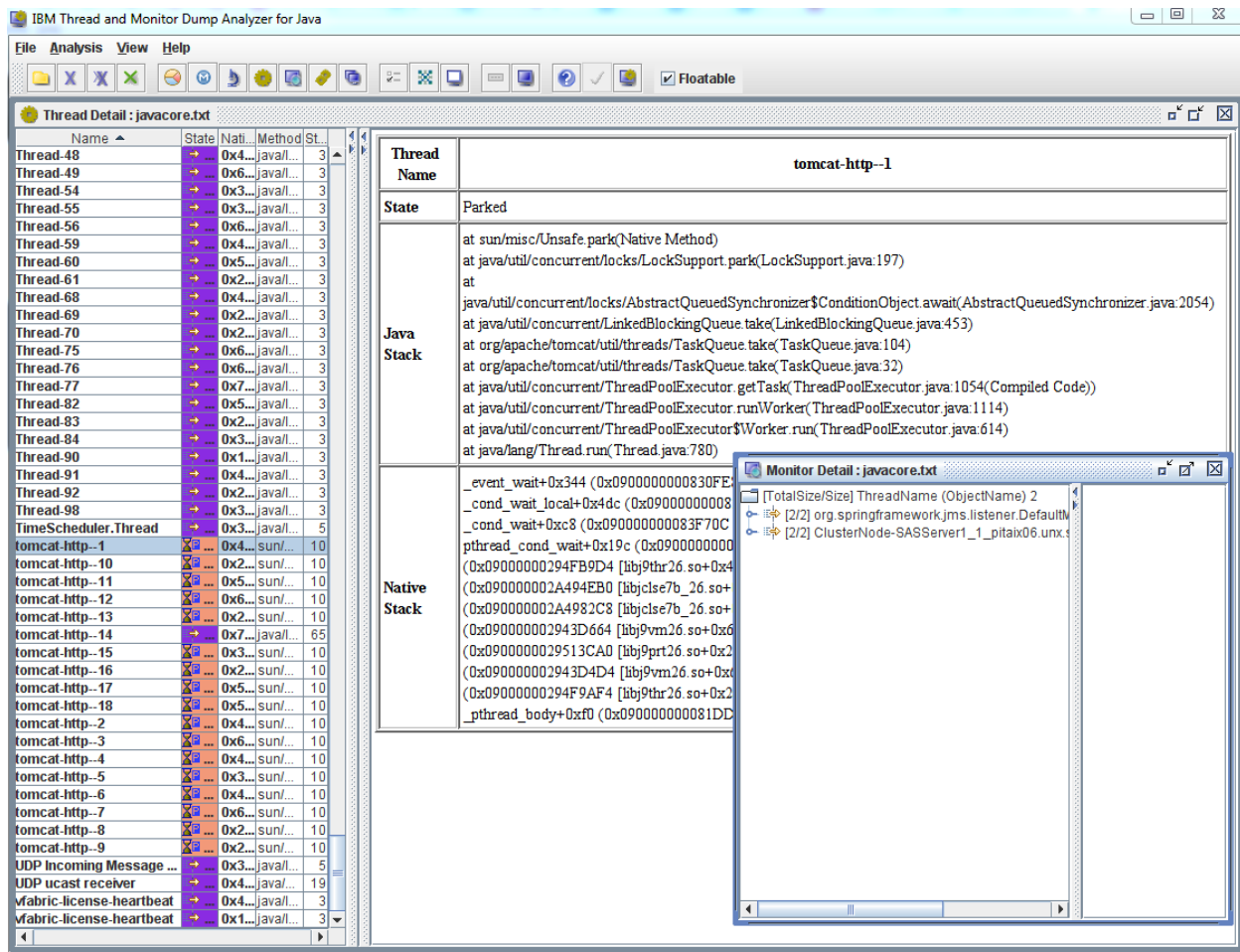


Figure 10 – IBM Thread and Monitor Dump Analyzer Viewing Thread and Monitor Details from an IBM JVM

## ANALYZING GC LOGS

As previously mentioned, one technique for monitoring the health of the Java heap over time is to enable GC logging for the Java processes. Thankfully there are tools available to that can easily parse and display the data from these logs graphically for analysis. Among the tools that provide this functionality are HPjmeter (for HotSpot JVMs) and IBM GCMV (for IBM JVMs, but also includes support for HotSpot JVMs). Sample screenshots of HPjmeter (Figure 11) and IBM GCMV (Figure 12) are shown below.

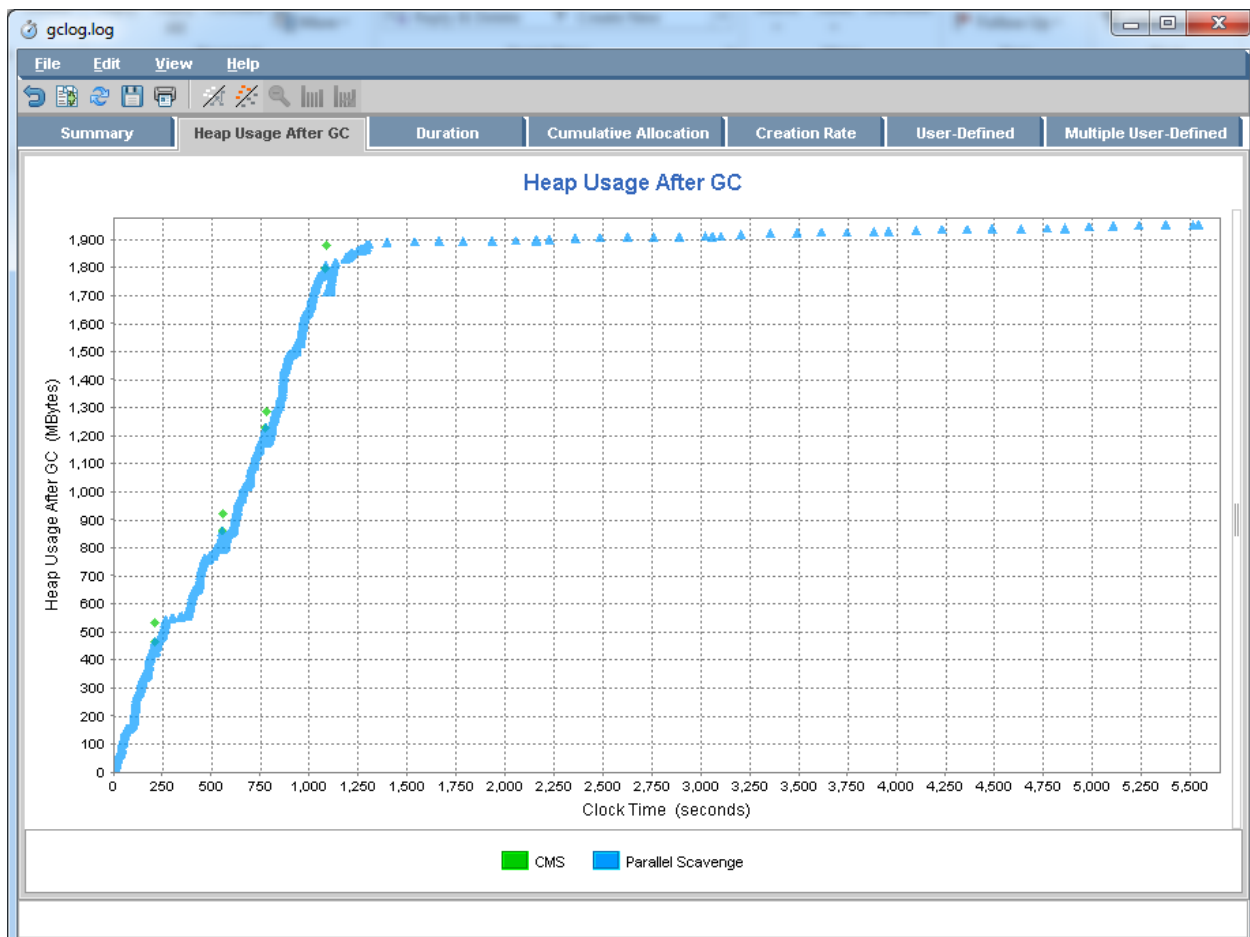


Figure 11 – Sample GC Data Displayed in HPjmeter

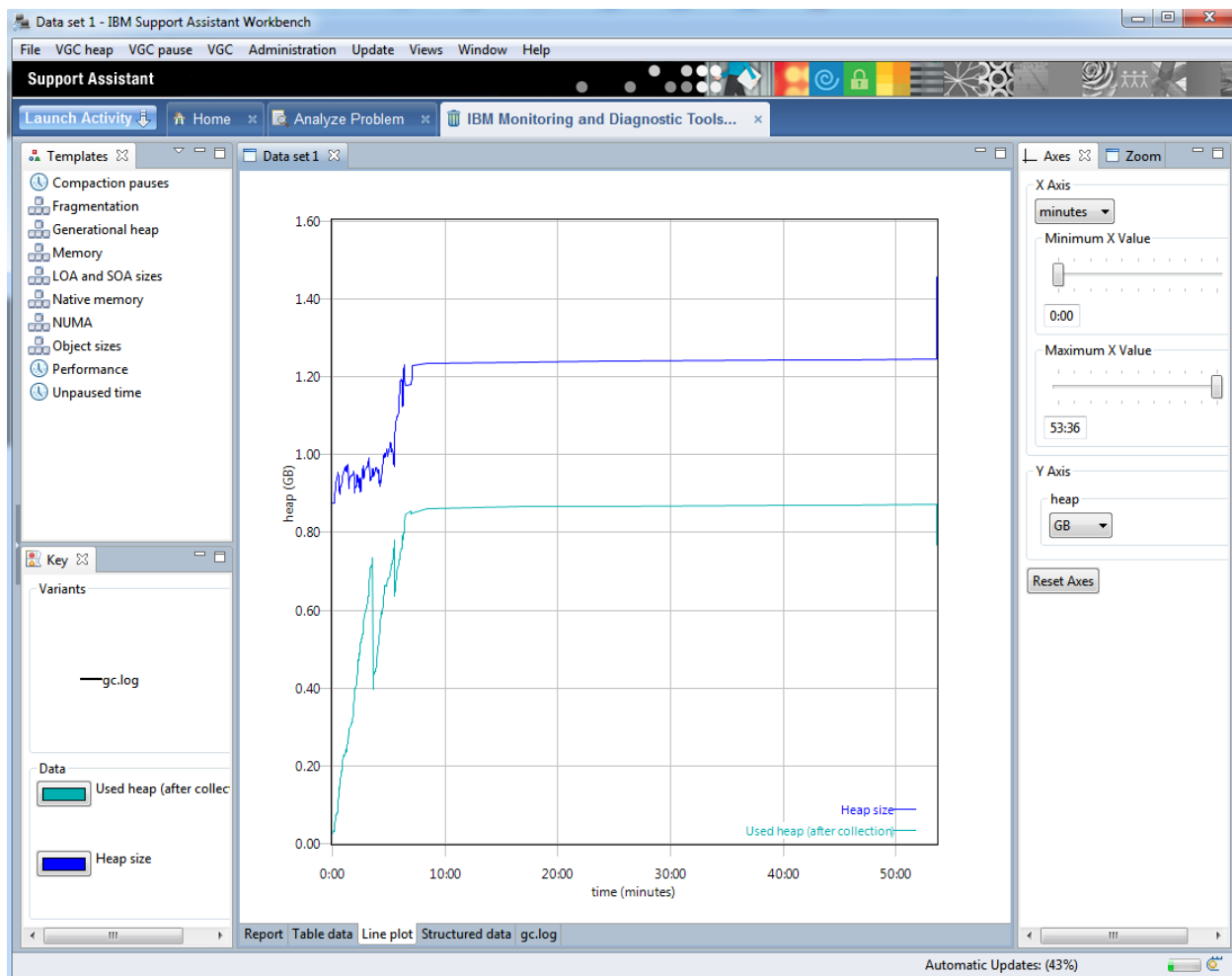


Figure 12 – Sample GC Data Displayed in IBM GCMV

## CONCLUSION

The SAS web application environment is a complex environment comprised of many different components, each of which contains a number of resource dependencies that can be tuned or scaled up to improve performance. As a result, the process of determining how to analyze the needs of the environment and pinpoint the potential areas of improvement might seem overwhelming. However, by applying the information presented in this article, key resources can be tracked in a variety of ways in order to identify potential areas of concern. Monitoring, tuning, and scaling is an ongoing process, and by applying these concepts on a consistent basis administrators can keep their SAS web applications running smoothly for their users.

## REFERENCES

The Apache Software Foundation. Apache Module mod\_status. Available at [http://httpd.apache.org/docs/2.2/mod/mod\\_status.html](http://httpd.apache.org/docs/2.2/mod/mod_status.html).

The Eclipse Foundation. Eclipse Memory Analyzer. Available at <https://www.eclipse.org/mat/>.

Hewlett-Packard Development Company. HPjmeter Downloads and Documentation. Available at <http://www.hp.com/go/hpjmeter>.

IBM Corporation. "AIX problem determination" In AIX User Guide for IBM SDK, Java Technology Edition, Version 7 Release 1. Available at [http://pic.dhe.ibm.com/infocenter/java7sdk/v7r0/index.jsp?topic=%2Fcom.ibm.java.aix.71.doc%2Fdiag%2Fproblem\\_determination%2Faix.html](http://pic.dhe.ibm.com/infocenter/java7sdk/v7r0/index.jsp?topic=%2Fcom.ibm.java.aix.71.doc%2Fdiag%2Fproblem_determination%2Faix.html).



IBM Corporation. IBM Monitoring and Diagnostic Tools for Java. Available at <https://www.ibm.com/developerworks/java/jdk/tools/#tools>.

IBM Corporation. IBM Support Assistant Workbench. Available at <http://www-01.ibm.com/software/support/isa/workbench.html>.

IBM Corporation. IBM Thread and Monitor Dump Analyzer for Java. Available at <https://www.ibm.com/developerworks/community/alphaworks/tech/jca>.

Oracle. Java VisualVM. Available at <http://docs.oracle.com/javase/7/docs/technotes/guides/visualvm/>.

Oracle. JDK Tools and Utilities. Available at <http://docs.oracle.com/javase/7/docs/technotes/tools/>.

Oracle. Thread Dump Analyzer. Available at <https://java.net/projects/tda>.

## RECOMMENDED READING

SAS Institute Inc. 2013. *SAS Environment Manager 2.1: User's Guide*. 2d edition. Cary, NC: SAS Institute Inc.

SAS Institute Inc. 2013. *SAS 9.4 Intelligence Platform: Middle-Tier Administration Guide*. 2d edition. Cary, NC: SAS Institute Inc.

SAS Institute Inc. 2013. *SAS 9.4 Intelligence Platform: Overview*. Cary, NC: SAS Institute Inc.

SAS Institute Inc. 2013. *SAS 9.4 Intelligence Platform: System Administration Guide*. 2d edition. Cary, NC: SAS Institute Inc.

SAS Institute Inc. 2014. *SAS 9.4 Web Applications: Tuning for Performance and Scalability*. Cary, NC: SAS Institute Inc.

## CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Rob Sioss  
SAS Institute  
SAS Campus Drive  
Cary, NC 27513  
(919) 531-1297  
Fax: (919) 677-4444  
rob.sioss@sas.com  
<http://www.sas.com>

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.