



THE
POWER
TO KNOW.

SAS[®] 9.2 Web Applications Clustering



The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2009. *SAS® 9.2 Web Applications: Clustering*. Cary, NC: SAS Institute Inc.

SAS® 9.2 Web Applications: Clustering

Copyright © 2009, SAS Institute Inc., Cary, NC, USA

All rights reserved. Produced in the United States of America.

For a hard-copy book: No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, or otherwise, without the prior written permission of the publisher, SAS Institute Inc.

For a Web download or e-book: Your use of this publication shall be governed by the terms established by the vendor at the time you acquire this publication.

U.S. Government Restricted Rights Notice: Use, duplication, or disclosure of this software and related documentation by the U.S. government is subject to the Agreement with SAS Institute and the restrictions set forth in FAR 52.227-19, Commercial Computer Software-Restricted Rights (June 1987).

SAS Institute Inc., SAS Campus Drive, Cary, North Carolina 27513.

1st electronic book, October 2009

2nd electronic book, November 2009

SAS® Publishing provides a complete selection of books and electronic products to help customers use SAS software to its fullest potential. For more information about our e-books, e-learning products, CDs, and hard-copy books, visit the SAS Publishing Web site at support.sas.com/publishing or call 1-800-727-3228.

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 registered trademarks or trademarks of their respective companies.

Table of Contents

| | |
|--|-----------|
| Chapter 1 — Overview | 1 |
| Chapter 2 — SAS Middle Tier Limitations | 3 |
| SAS Web Report Studio Scheduling Limitation | 3 |
| SAS Content Server Limitation | 3 |
| SAS OnlineDoc for the Web Limitation..... | 3 |
| SAS BI Dashboard Event Generator | 3 |
| Deploy a Stand-alone Web Application Server | 3 |
| Hot and Cold Standby Strategy..... | 5 |
| Reconfigure SAS BI Dashboard Directories | 6 |
| Reconfigure SAS Content Server..... | 6 |
| High Availability Software Strategy | 7 |
| Java Messaging Service Servers | 7 |
| Chapter 3 — JBoss Clustering | 9 |
| Create Additional JBoss Server Instances | 10 |
| Configure a Reverse Proxy | 13 |
| Configure JBoss to Use mod_jk | 15 |
| Perform Common Configuration Tasks | 15 |
| Chapter 4 — Oracle WebLogic Server Clustering | 17 |
| Clone the Managed Server | 18 |
| Create a Cluster and Assign the Servers..... | 19 |
| Reconfigure Resources | 19 |
| Target the Mail Session to the Cluster | 19 |
| Reconfigure the JMS Resources | 19 |
| Target the JDBC Data Source | 20 |
| <i>Deploy the SAS Web Applications to the Cluster</i> | 20 |
| Configure a Reverse Proxy | 21 |
| Perform Common Configuration Tasks | 24 |
| Chapter 5 — IBM WebSphere Application Server Clustering | 25 |
| Create a Server Template from SASServer1 | 25 |
| Create a Stand-alone Server Instance | 25 |
| Create the Cluster and Add Servers | 26 |
| Configure JVM Arguments | 26 |
| Target SAS Web Applications to the Cluster | 26 |
| Configure a Reverse Proxy | 27 |
| Perform Common Configuration Tasks | 27 |
| Chapter 6 — Common Tasks | 29 |
| Load Balancing for SAS Web Applications That Are Not Proxied | 29 |
| Configure Logging for SAS Web Applications in a Cluster | 29 |
| Deploy SAS Themes to an HTTP Server..... | 30 |
| Change the Connections for the SAS Web Applications..... | 30 |

| | |
|---|-----------|
| Change the Connection for SAS Content Server | 31 |
| Change the WebDAV Repository URL..... | 31 |
| Recommended Reading..... | 33 |

Chapter 1 — Overview

The SAS 9.2 Web applications can be clustered to improve performance and to provide high availability. Performance is improved due to additional server instances and greater memory usage. Performance can also be improved by deploying servers on to additional hardware. Clustering can also provide high availability. If server instances are clustered on a single machine, then there is protection against software failure. If multiple machines are used for clustering servers, then the additional machines in the cluster provide protection from software failure and protection from hardware failure.

This document provides application-specific considerations for clustering and Web application server-specific details for clustering the SAS Web applications that are deployed with SAS Enterprise Business Intelligence Server.

Here are the high-level steps to follow:

1. **Plan for Deployment:** Using functional and non-functional requirements as well as a sizing and capacity plan, determine how many servers you need and the server capacity that you need.
2. **Choose Deployment Plan:** Based on the input from the previous step, choose a deployment plan that uses single or multiple servers for SAS 9.2 software.
3. **Install SAS Software with the SAS Deployment Wizard:** Use the wizard to install SAS software either interactively or non-interactively. The interactive method provides a series of prompts to enable you to modify parameters based on your environment. Alternatively, you can configure a response file with parameters for your environment and run the wizard in non-interactive mode. On the first machine in the middle tier, use the SAS Deployment Wizard to install and configure software. On the additional machines in the middle tier, do not use the SAS Deployment Wizard, but copy selected directories and configuration files from the first machine.
4. **Prepare for Clustering:** Manually configure the Web application server and the SAS Web applications to take advantage of clustering. You might need to configure resources such as JDBC data sources and Java Messaging Services. The work to perform during this step depends on the number of clusters, the number of servers within each cluster, and whether SAS Web applications are relocated to new Web application servers.
5. **Cluster:** Create the Web application server cluster. WebLogic Server and WebSphere Application Server provide administrative interfaces that assist with creating a cluster. For JBoss Application Server, the cluster is based on multiple stand-alone servers.
6. **Perform Post Configuration for the Cluster:** After creating the cluster, configure an HTTP Server as a reverse proxy so that user requests are load balanced and routed to the clustered servers.
7. **Validate the Cluster:** Perform a series of log on attempts to confirm that loads are distributed across the servers in the cluster. Afterward, perform more intensive tasks to confirm that the cluster meets your performance and high availability requirements.

Chapter 2 — SAS Middle Tier Limitations

SAS Web Report Studio Scheduling Limitation

SAS Web Report Studio jobs can be scheduled in two ways. The first way is with the Platform Process Manager that is part of Platform Suite for SAS. The second way is with in-process scheduling that is provided as part of SAS Web Report Studio. In a clustered environment, do not use the in-process scheduling because the jobs will fail. If you need to schedule SAS Web Report Studio jobs, use Platform Suite for SAS. Multiple instances of SAS Web Report Studio can be deployed in a cluster if scheduling is not used, or if scheduling is provided by Platform Suite for SAS.

SAS Content Server Limitation

The SAS Content Server (`sas.wip.scs9.2.ear`) cannot be clustered. The SAS Content Server provides WebDAV services for other SAS Web applications. Only a single instance of SAS Content Server can be running at any one time. To provide high availability, you can deploy the SAS Content Server application to a stand-alone server and then use a hot and cold standby strategy or high availability software. More information about these two strategies is provided in following sections.

SAS OnlineDoc for the Web Limitation

The SAS OnlineDoc for the Web application (`sas.onlinedocweb9.2.ear`) cannot be clustered. Your choices for providing high availability for this application are the same as for the SAS Content Server.

SAS BI Dashboard Event Generator

The SAS BI Dashboard Event Generator application is packaged in the SAS BI Dashboard EAR file. The SAS BI Dashboard Web application can be clustered, but the SAS BI Dashboard Event Generator cannot be clustered. The events generated by the event generation framework must be generated by a single instance of the event generation framework so that the events are generated for a single target. Like the previous SAS Web applications that cannot be clustered, you can provide high availability with a hot and cold standby strategy, or use high availability software. If SAS BI Dashboard receives a lot of use and multiple instances are needed, then there is another option. You can separate the event generation WAR file from the SAS BI Dashboard EAR file and then deploy a single instance of the WAR on a stand-alone server, but deploy SAS BI Dashboard to the cluster. This strategy is used through the rest of this document. More information is provided in the next section.

Deploy a Stand-alone Web Application Server

As indicated in the previous sections, some SAS Web applications cannot be clustered. The packaging of SAS BI Dashboard Event Generator with the EAR file presents a challenge for sites that intend to cluster SAS BI Dashboard. The following steps describe the option of extracting the Web application WAR file and then modifying the SAS BI Dashboard EAR file. When these steps are complete, the SAS BI Dashboard Event Generator can be deployed to a stand-alone Web application server, and the SAS BI Dashboard EAR file can be deployed to the cluster of Web application servers. To ensure high

availability, consider using a hot and cold standby strategy or high availability software for the stand-alone server. Follow these steps:

1. Make a backup of the *SAS-config-dir\Levn\Web\Staging\sas.bidashboard4.2.ear* file.
2. In a temporary directory, extract the contents of the *sas.bidashboard4.2.ear* file:


```
jar xf sas.bidashboard4.2.ear
```
3. Move the *sas.eventsgenerationframework.war* file from the temporary directory to another temporary directory. (Modifications to this WAR file are done after the next steps that describe changes to the EAR file.)
4. Edit the *META-INF/application.xml* file and remove the following information:

```
<module>
  <web>
    <web-uri>sas.eventsgenerationframework.war</web-uri>
    <context-root>SASBIDashboardEventGen</context-root>
  </web>
</module>
```

5. Repackage *sas.bidashboard4.2.ear* file from the temporary directory:


```
jar cf ../sas.bidashboard4.2.ear .
```
6. In the second temporary directory that contains the *sas.eventsgenerationframework.war* file, extract the contents of the WAR file to a temporary directory:
 - For Oracle WebLogic Server, edit the *WEB-INF/weblogic.xml* file, and add a *context-root* element as shown in the following example:

```
<!DOCTYPE weblogic-web-app PUBLIC "-//BEA Systems, Inc.//DTD Web
Application 8.1
//EN" "http://www.bea.com/servers/wls810/dtd/weblogic810-web-
jar.dtd">
<weblogic-web-app>
  <context-root>/SASBIDashboardEventGen</context-root>

  <jsp-descriptor>
    <jsp-param>
    ...
```

- For JBoss Application Server, edit the *WEB-INF/jboss-web.xml* file, and change the *context-root* element as shown in the following example:

```
<jboss-web>
  <resource-ref>
    <res-ref-name>jms/AlertQueue</res-ref-name>
    <res-type>javax.jms.Queue</res-type>
    <jndi-name>sas/jms/AlertQueue</jndi-name>
  </resource-ref>
```

```

<resource-ref>
  <res-ref-name>jms/QueueConnectionFactory</res-ref-name>
  <res-type>javax.jms.QueueConnectionFactory</res-type>
  <jndi-name>sas/jms/QueueConnectionFactory</jndi-name>
</resource-ref>
<context-root>/SASBIDashboardEventGen</context-root>
</jboss-web>

```

- For IBM WebSphere Application Server, the context root is set when the WAR file is deployed.
7. Repackage the `sas.eventsgenerationframework.war` file from the temporary directory:


```
jar cf ../sas.eventsgenerationframework.war .
```
 8. Stop, undeploy, and remove the existing `sas.bidashboard4.2.ear` file on the Web application server.
 9. Deploy and start the repackaged `sas.bidashboard4.2.ear` and `sas.eventsgenerationframework.war` applications.

Hot and Cold Standby Strategy

This strategy can be used to overcome the challenge that three SAS Web applications cannot be clustered, yet meet high availability requirements. The strategy is to configure two machines, each with a stand-alone Web application server with the three SAS Web applications. One Web application server is active on the hot machine, and the other Web application server is inactive on the cold standby machine.

One consideration for implementing this configuration is that these applications use files that are external to the EAR file and the Web application server environment. Maintaining the integrity of these files, providing availability to these files, and coordinating that only one instance of an application is reading and writing to these files is critical. These files are located in the following directories:

- `SAS-config-dir\Levn\AppData\SASBIDashboard4.2\`
- `SAS-config-dir\Levn\AppData\SASBIDashboardEventGen4.2\`
- `SAS-config-dir\Levn\AppData\SASContentServer\`

The files in these directories must be accessible from both machines. One way to provide accessibility is to move the contents of these directories to network storage. The two directories that are related to SAS BI Dashboard are identified in metadata. The directory that is related to SAS Content Server is identified in a `web.xml` configuration file. Instructions for reconfiguring both applications are provided in the following sections.

Reconfigure SAS BI Dashboard Directories

Reconfigure the location of configuration and log files for SAS BI Dashboard and SAS BI Dashboard Event Generator. The default location is a directory path that is local to the machine. Modify the directory paths to identify a location on network storage. Follow these steps:

1. Start SAS Management Console.
2. Select **Application Management > Configuration Manager**.
3. Right-click **BI Dashboard 4.2** and select **Properties**.
4. Select the **Advanced** tab.
5. Modify the property values for `bid.DashboardConfig` and `bid.LogFile`. Set these values to a location on the shared network storage. Click **OK**.
6. Right-click **BI Dashboard 4.2 - Event Generator** and select **Properties**.
7. Select the **Advanced** tab.
8. Modify the property values for `bid.eventgen.EventGenConfig` and `bid.eventgen.LogFile`. Set these values to a location on the shared network storage. Click **OK**.
9. Restart the SAS BI Dashboard 4.2 Web application. If SAS BI Dashboard Event Generator WAR file was split from the SAS BI Dashboard EAR file, then also restart the SAS BI Dashboard Event Generator Web application.

Reconfigure SAS Content Server

SAS Content Server provides WebDAV services. The application uses a `web.xml` configuration file to identify the location of the WebDAV repository. After moving the contents of the `SAS-config-dir/Levn/AppData/SASContentServer` directory to network storage, modify the `web.xml` file to reference the new directory path. Follow these steps:

1. Make a backup of the `SAS-config-dir\Levn\Web\Staging\sas.wip.scs9.2.ear` file.
2. In a temporary directory, extract the contents of the `sas.wip.scs9.2.ear` file:

```
jar xf sas.wip.scs9.2.ear
```

3. In another temporary directory, extract the contents of the `sas.svcs.scs.war` file:

```
jar xf sas.svcs.scs.war
```

4. Edit the `web.xml` file that is located in the WEB-INF directory that was extracted from the `sas.svcs.scs.war` file:

5. Modify the parameter value for `repository-home` to reference the new directory location. The following example shows a directory path that begins with `\\nas`:

```
<init-param>
  <param-name>repository-home</param-name>
  <param-value>\\nas\AppData\SASContentServer\Repository</param-
value>
  <description>the repository home</description>
</init-param>
```

6. Rebuild the WAR and EAR files:


```
jar cvf sas.svcs.scs.war *
jar cvf sas.wip.scs9.2.ear *
```
7. Uninstall the `sas.wip.scs9.2.ear` file from the Web application server and install the modified `sas.wip.scs9.2.ear` file.

High Availability Software Strategy

Another alternative for ensuring the high availability of the three SAS Web applications that cannot be clustered is to use high availability software such as IBM PowerHA, Sun Cluster, and Veritas Cluster Server by Symantec. Most of these products provide virtual networking and replicated data storage. In this case, a virtual IP address is shared by two machines and the high availability software detects a failure and manages the transition of activity from the failed machine to the standby machine. The following list identifies the high-level steps that the high availability software takes:

1. Cleans up the process on the failed machine.
2. Releases disk resources.
3. Shuts down the network interface to which the virtual IP address is bound.
4. Mounts the disks on the standby machine.
5. Checks the disk.
6. Starts the network interface to which the virtual IP address is bound.
7. Starts the Web application server and the three SAS Web applications that are deployed on it.

Java Messaging Service Servers

In SAS 9.2, a Java Messaging Service (JMS) Server and the JMS resources are used by the SAS Shared Services Web application. SAS BI Dashboard has a Web module, SAS BI Dashboard Event Generator, that uses the JMS resources provided by the SAS Shared Services Web application. As a result, your choice of alternatives to deploy SAS BI Dashboard Event Generator affects your choices for how to configure JMS within a cluster. Because SAS BI Dashboard Event Generator is the application that sends messages and cannot be clustered, there is little benefit to providing JMS services in the cluster. SAS recommends using a dedicated, single JMS Server (not to be shared with other application JVMs) and configuring JNDI contexts to the Queue Connection Factory, Topic Connection Factory, Queues, and Topics.

Chapter 3 — JBoss Clustering

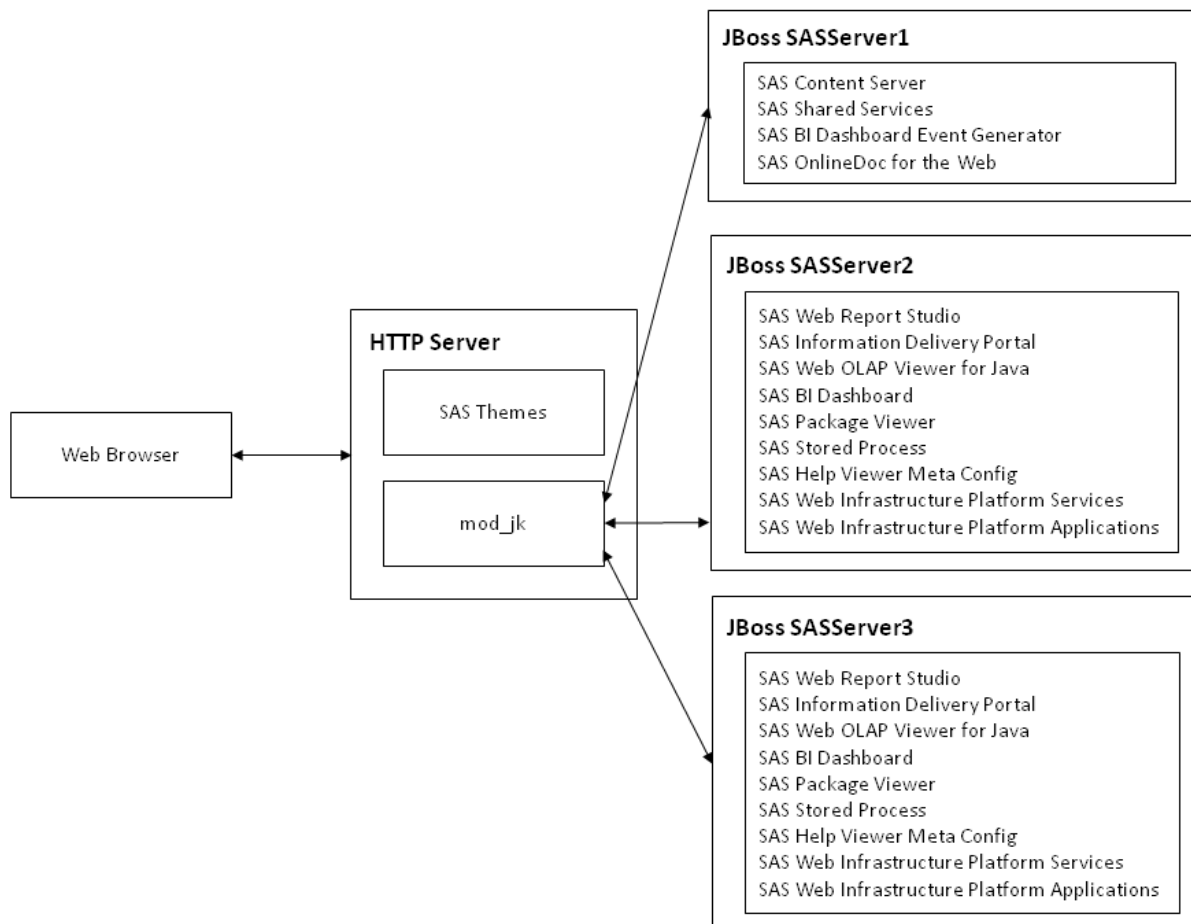
The following list identifies the necessary steps to perform clustering with JBoss Application Server 4.2:

Prerequisite: Use the SAS Deployment Wizard to configure JBoss Application Server and deploy the SAS Web applications. This step is not described in this document.

1. Use the server created by the wizard as a template for creating the additional server for the cluster and the stand-alone server. Modify the server instances to use unique ports.
2. Configure an HTTP server to act as a reverse proxy. This document describes using Apache HTTP Server with mod_jk.
3. Configure JBoss to work with the reverse proxy.
4. Perform the common configuration tasks.

The tasks in the following sections describe how to create a topology that is similar to the example in the following figure.

Figure 1 Sample Topology for JBoss Application Server



If you plan to configure a cluster on multiple machines, then copy the following directories (files and subdirectories) to each additional machine in the cluster:

- *SAS-config-dir/Levn/Web/Application*
- *SAS-config-dir/Levn/Web/Common*
- *SAS-config-dir/Levn/Web/Temp*
- *SAS-config-dir/Levn/Web/Utilities*
- *SAS-config-dir/Levn/AppData/SASBIDashboard4.2*

If the SAS Content Server Web application is moved to a different machine, then move the *SAS-config-dir/Levn/AppData/SASContentServer* directory to the new machine. If SAS BI Dashboard Event Generator is moved to a different machine, then move the *SAS-config-dir/Levn/AppData/SASBIDashboardEventGen4.2* directory to the different machine as well.

Create Additional JBoss Server Instances

Before beginning this task, use the SAS Deployment Wizard to perform an automated configuration of JBoss and an automatic deployment of the SAS Web applications. When the wizard completes the configuration, a server named *SASServer1* exists in *JBOSS_HOME\server\SASServer1*. Based on requirements and capacity planning, copy the *SASServer1* directory for each additional server and rename the directory *SASServer2*, *SASServer3*, and so on. After using the wizard, reconfigure each additional server to use unique ports. The following example shows the steps for modifying *SASServer2*:

1. Copy the contents of *SASServer1* to a new directory such as *SASServer2*. As shown in the previous figure, *SASServer1* is used as a stand-alone server, and the additional servers are used in the cluster.
2. Decide which applications you want to cluster. Leave those EAR files in the *SASServer2/deploy_sas* directory and remove the other EAR files. For information about which applications to deploy on each server, see the following table:

| Application | Target |
|--|---|
| <i>sas.bidashboard4.2.ear</i> | Target this EAR file to each server in the cluster if the <i>sas.eventsgenerationframework.war</i> file was removed from the EAR file. Otherwise, target this EAR file to the stand-alone server. |
| <i>sas.eventsgenerationframework.war</i> | If this WAR file is extracted from <i>sas.bidashboard4.2.ear</i> , then deploy this WAR file to the stand-alone server. |
| <i>sas.packageviewer.ear</i> | Target all servers in the cluster. |
| <i>sas.portal4.2.ear</i> | Target all servers in the cluster. |
| <i>sas.shared9.2.ear</i> | Target the stand-alone server. |
| <i>sas.storedprocess9.2.ear</i> | Target all servers in the cluster. |

| Application | Target |
|---|--|
| <code>sas.themes.ear</code> | Do not target this application to any servers. Later tasks in this document describe how to deploy SAS Themes Web application content to an HTTP server. |
| <code>sas.webdocmd9.2.ear</code> | Target all servers in the cluster. |
| <code>sas.webolapviewer4.2.ear</code> | Target all servers in the cluster. |
| <code>sas.webreportstudio4.2.ear</code> | Target all servers in the cluster. |
| <code>sas.wip.apps9.2.ear</code> | Target all servers in the cluster. |
| <code>sas.wip.scs9.2.ear</code> | Target the stand-alone server. |
| <code>sas.wip.services9.2.ear</code> | Target all servers in the cluster. |

3. Edit the `SASServer2\conf\jboss-service.xml` file. Change all instances of `SASServer1` to `SASServer2` as shown in this example:

```

<!-- ===== -->
<!-- Service Binding -->
<mbean code="org.jboss.services.binding.ServiceBindingManager"
  name="jboss.system:service=ServiceBindingManager">
  <attribute name="ServerName">SASServer2</attribute>
  <!-- NOTE: Put the StoreURL attribute all on one line, it is -->
  <!-- on more than one line here here due to space limitations. -->
  <attribute name="StoreURL">
    C:\jboss\server\SASServer2\jboss.port.bindings.xml
  </attribute>
  <attribute name="StoreFactoryClassName">
    org.jboss.services.binding.XMLServicesStoreFactory
  </attribute>
</mbean>

```

4. For all servers except the stand-alone server, remove the following lines from the `SASServern\conf\jboss-service.xml` file. Removing these lines prevents the servers in the cluster from starting JMS services and prevents JNDI naming conflicts with the stand-alone server:

```

<mbean code="org.jboss.naming.LinkRefPairService"
  name="jboss.jms:alias=SASTopicConnectionFactory">
  <attribute
    name="JndiName">sas/jms/TopicConnectionFactory</attribute>
  <attribute name="RemoteJndiName">ConnectionFactory</attribute>
  <attribute name="LocalJndiName">java:/JmsXA</attribute>
  <depends>jboss:service=Naming</depends>
</mbean>

```

```
<mbean code="org.jboss.naming.LinkRefPairService"
  name="jboss.jms:alias=SASQueueConnectionFactory">
  <attribute
    name="JndiName">sas/jms/QueueConnectionFactory</attribute>
  <attribute name="RemoteJndiName">ConnectionFactory</attribute>
  <attribute name="LocalJndiName">java:/JmsXA</attribute>
  <depends>jboss:service=Naming</depends>
</mbean>
```

```
<mbean code="org.jboss.mq.server.jmx.Queue"
  name="jboss.mq.destination:service=Queue,name=AlertQueue">
  <depends optional-attribute name="DestinationManager">
    jboss.mq:service=DestinationManager
  </depends>
  <attribute name="JNDIName">sas/jms/AlertQueue</attribute>
</mbean>
```

```
<mbean code="org.jboss.mq.server.jmx.Queue"
  name="jboss.mq.destination:service=Queue,name=WorkflowQueue">
  <depends optional-attribute name="DestinationManager">
    jboss.mq:service=DestinationManager</depends>
  <attribute name="JNDIName">sas/jms/WorkflowQueue</attribute>
</mbean>
```

5. From all servers, except the stand-alone server, delete the SASServern\deploy\SharedServices-ds.xml file.
6. Edit the SASServer2\jboss.port.bindings.xml file, and change the instances of ports-01 to SASServer2. This change sets SASServer2 to use a unique set of ports. All the ports are incremented by 100 over the ports used by SASServer1. This typically results in SASServer2 using port 8180 for HTTP. See this example:

```
<!-- ***** -->
<!-- *           ports-01 Change to SASServer2 * -->
<!-- ***** -->
<server name="SASServer2"> <!-- was "ports-01" -->
  <!-- EJB3 Remoting Connector ... -->
  <service-config name="jboss...
```

When you add SASServer3 and SASServer4, use ports-02 and ports-03. If you need more than four servers, then you need to modify jboss.port.bindings.xml to add a new and unique port stanza.

7. In the JBOSS_HOME\bin directory, copy SASServer1.bat to SASServer2.bat.
8. Edit SASServer2.bat and make the following changes:
 - a. Change all instances of SASServer1 to SASServer2.
 - b. Change the value for the -Dsas.auto.publish.port parameter to the HTTP port for SASServer2. This value is usually 8180.
 - c. Set the values for any other JVM parameters that are specific to your SAS middle-tier topology.

- On Windows operating systems, JBoss is configured to run as a service. The service configuration is set in `JBOSS_HOME\server\SASServer2\wrapper.conf`. If you plan to run JBoss as a Windows service, then edit the `wrapper.conf` file and make all the changes that you set in the previous step. For example, change all instances of `SASServer1` to `SASServer2`, and change the value for the `-Dsas.auto.publish.port` parameter to the new HTTP port. On all the machines except the machine that is running SAS Remote Services, remove the `wrapper.ntservice.dependency` entries. After the edits are complete, create the service entry as shown in the following example:

```
SASServer2.bat -install
```

The outcome for Stage 1 is a series of JBoss servers that are configured to run on unique ports. Some of the JBoss servers have identical deployments of the SAS Web applications. At least one JBoss server instance must be configured with the SAS Web applications that cannot be clustered.

Configure a Reverse Proxy

HTTP requests must be distributed to the JBoss cluster members. Two well known technologies for performing this function are `mod_jk` and `mod_cluster`. For JBoss Application Server 5.0.0 and later, `mod_cluster` is the native HTTP based load balancer. SAS 9.2 supports JBoss 4.2.0-GA as a minimum requirement, and works with `mod_jk`. This document describes using `mod_jk` with Apache HTTP Server, but `mod_jk` also works with Microsoft Internet Information Systems (IIS). Follow these steps:

- Download the `mod_jk` binaries from <http://tomcat.apache.org/download-connectors.cgi>.
- Rename the downloaded file to `mod_jk.so` and move the file to `APACHE_HOME/modules`.
- Edit the `APACHE_HOME/conf/httpd.conf` file, and add the `mod_jk` configuration information as shown in the following example:

```
LoadModule jk_module modules/mod_jk.so
```

```
JkWorkersFile conf/workers.properties
JkLogFile logs/mod_jk.log
JkLogLevel info
JkLogStampFormat "[%a %b %d %H:%M:%S %Y] "
# JKOptions indicates to send the SSL KEY SIZE
JkOptions +ForwardKeySize +ForwardURICompat -ForwardDirectories
JkRequestLogFormat "%w %V %T"
JkMount /application/* loadbalancer
JkMountFile conf/uriworkers.properties
JkShmFile logs/jk.shm
<Location /jkstatus/>
    JkMount status
    Order deny,allow
    Deny from all
    Allow from 127.0.0.1
</Location>
```

- Create an `APACHE_HOME/conf/workers.properties` file. The following example identifies `SASServer1` as the stand-alone server and two servers for a cluster:

```
worker.list=loadbalancer,status
worker.list=sasserver1
```

```
# the stand-alone server has no load balancing settings
worker.sasserver1.host=hostname1.example.com
```

```

worker.sasserver1.port=8009 # the original SASServer1 value
worker.sasserver1.type=ajp13

worker.sasserver2.host=hostname2.example.com
worker.sasserver2.port=8109 # the original ports-01 value
worker.sasserver2.type=ajp13
worker.sasserver2.lbfactor=1
worker.sasserver2.cachesize=10

worker.sasserver3.host=hostname2.example.com
worker.sasserver3.port=8209 # the original ports-02 value
worker.sasserver3.type=ajp13
worker.sasserver3.lbfactor=1
worker.sasserver3.cachesize=10

worker.loadbalancer.type=lb
worker.loadbalancer.balance_workers=sasserver2,sasserver3
worker.loadbalancer.sticky_session=True

worker.status.type=status

```

5. Create an `APACHE_HOME/conf/uriworkers.properties` file. In conjunction with the `workers.properties` file, this file maps the location of the SAS Web applications to the JBoss cluster and the stand-alone server. Add the following information to the file:

```

/SASContentServer=sasserver1
/SASContentServer/*=sasserver1
# uncomment the following two if the event generator WAR
# was separated from the EAR and deployed on the stand-alone server
#/SASBIDashboardEventGen=sasserver1
#/SASBIDashboardEventGen/*=sasserver1
# uncomment the following two if SAS OnlineDoc for the Web
# is installed on the stand-alone server
#/SASOnlineDoc=sasserver1
#/SASOnlineDoc/*=sasserver1
/SASSharedApps=sasserver1
/SASSharedApps/*=sasserver1
/SASSharedPortlets=sasserver1
/SASSharedPortlets/*=sasserver1

/SASAdmin=loadbalancer
/SASAdmin/*=loadbalancer
/SASBIDashboard=loadbalancer
/SASBIDashboard/*=loadbalancer
/SASBIWS=loadbalancer
/SASBIWS/*=loadbalancer
/SASPortal/*=loadbalancer
/SASPortal=loadbalancer
/SASLogon=loadbalancer
/SASLogon/*=loadbalancer
/SASPackageViewer=loadbalancer
/SASPackageViewer/*=loadbalancer
/SASPreferences=loadbalancer
/SASPreferences/*=loadbalancer
/SASStoredProcess=loadbalancer
/SASStoredProcess/*=loadbalancer

```

```

/SASWebOLAPViewer=loadbalancer
/SASWebOLAPViewer/*=loadbalancer
/SASWebReportStudio=loadbalancer
/SASWebReportStudio/*=loadbalancer
/SASWebDoc=loadbalancer
/SASWebDoc/*=loadbalancer

```

Configure JBoss to Use mod_jk

At this stage of the reconfiguration, Apache HTTP Server is prepared as a reverse proxy. There is a stand-alone JBoss server that does not participate in clustering and at least two JBoss servers that are prepared to run in a cluster. This stage describes the steps that you must perform to configure each managed server to interface with the reverse proxy. Follow these steps for each JBoss server:

1. Edit the `JBOSS_HOME/server/SASServern/deploy/jboss-web.deployer/server.xml` file and make the following changes:
 - a. Confirm that the AJP Connector on port 8009 is enabled:


```

<Connector address="{ jboss.bind.address}"
emptySessionPath="true"
enableLookups="false" port="8009" protocol="AJP/1.3"
redirectPort="8443"/>

```
 - b. Add a `jvmRoute` attribute to the Engine element:


```

<Engine defaultHost="localhost" name="jboss.web"
jvmRoute="sasserver1">

```

Use "sasserver2" when you edit the file for SAServer2. These `jvmRoute` values match the names used in the `worker.properties` and `uriworkers.properties` files.
2. (Optional) Enable logging of access requests. You can disable this after confirming that your configuration is stable:


```

<!-- Access logger -->
<Valve className="org.apache.catalina.valves.AccessLogValve"
prefix="localhost_access_log." suffix=".log"
pattern="common" directory="{ jboss.server.home.dir}/log"
resolveHosts="false" />

```
3. Edit the `JBOSS_HOME/server/SASServern/deploy/jboss-web.deployer/META-INF/jboss-service.xml` file and enable the `UseJK` attribute:


```

<attribute name="UseJK" replace="true" trim="true">true</attribute>

```
4. Configure the JAAS Login Module for Application Logins – PFS and SCS.

Perform Common Configuration Tasks

To complete JBoss clustering, perform the procedures in "Chapter 6 – Common Tasks."

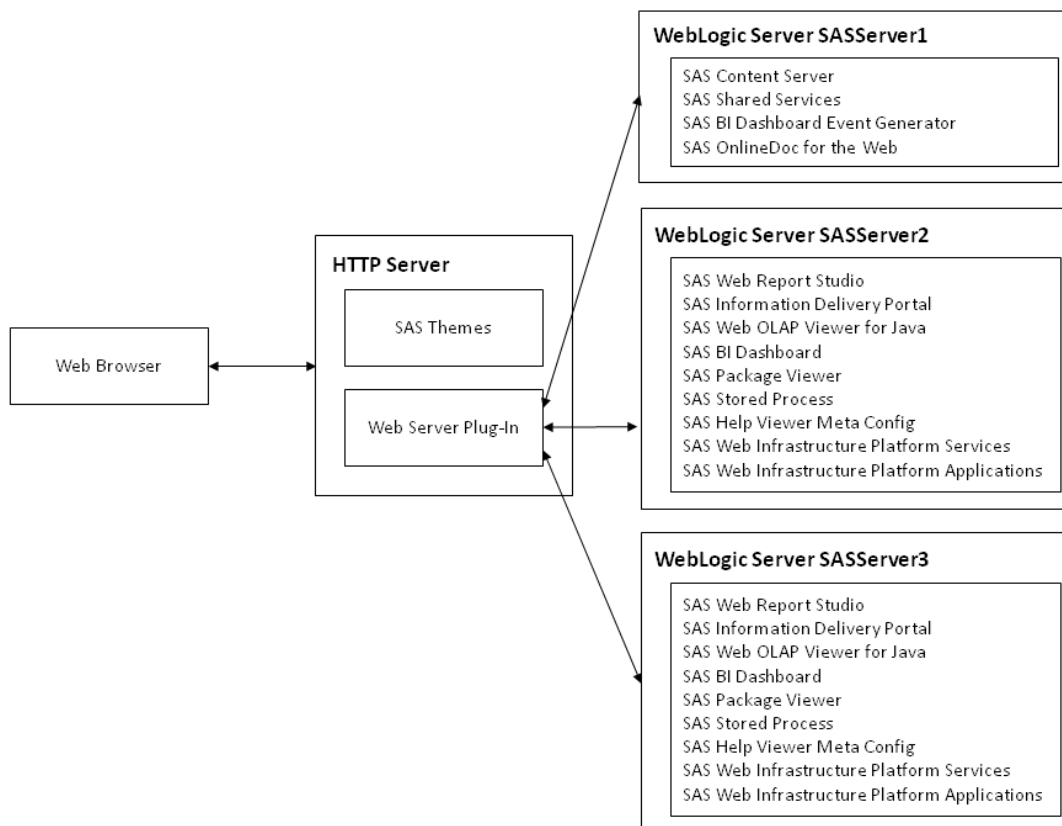
Chapter 4 — Oracle WebLogic Server Clustering

The starting point for this configuration is an Oracle WebLogic Server domain, SASDomain, that was created by the SAS Deployment Wizard and is not in a clustered configuration. This section describes how to clone the managed server that was configured by the wizard, configure the resources needed by the SAS Web applications, and then create a cluster. The tasks in this section include steps that use the WebLogic Administration Console, but the same tasks can be completed with the WebLogic Scripting Tool (WLST). The following list identifies the high-level configuration steps:

1. Clone the existing server or configure a new server, and leave one server as a stand-alone server for the SAS Web applications that are not clustered.
2. Create the cluster and assign servers.
3. Reconfigure resources such as mail, JDBC, and JMS.
4. Deploy the SAS Web applications to the cluster and some applications to the stand-alone server.
5. Configure a reverse proxy.
6. Perform the common configuration tasks.

The tasks in the following sections describe how to create a topology that is similar to the example in the following figure.

Figure 2 Sample Topology for Oracle WebLogic Server



If you plan to configure a cluster on multiple machines, then copy the following directories (files and subdirectories) to each additional machine in the cluster:

- `SAS-config-dir/Levn/Web/Applications`
- `SAS-config-dir/Levn/Web/Common`
- `SAS-config-dir/Levn/Web/SASDomain` (Do not include the servers subdirectory)
- `SAS-config-dir/Levn/Web/Temp`
- `SAS-config-dir/Levn/Web/Utilities`
- `SAS-config-dir/Levn/AppData/SASBIDashboard4.2`

If the SAS Content Server Web application is moved to a different machine, then move the `SAS-config-dir/Levn/AppData/SASContentServer` directory to the new machine. If SAS BI Dashboard Event Generator is moved to a different machine, then move the `SAS-config-dir/Levn/AppData/SASBIDashboardEventGen4.2` directory to the different machine as well.

Clone the Managed Server

The SAS Deployment Wizard can configure WebLogic Server and deploy the SAS Web applications to WebLogic Server. The wizard creates a server named `SASServer1`. The following steps describe how to clone `SASServer1` and then cluster it with other server instances. To clone `SASServer1`, follow these steps:

1. Use the Administration Console to shut down the managed server, `SASServer1`.
2. Select **Lock & Edit**.
3. In the Domain Structure panel, select **Environment > Servers**.
4. In the **Servers** table, select the check box for `SASServer1` and click **Clone**.
5. On the Clone a Server page, provide a unique name (`SASServer2`) and port number (7101) for the new server. Increment the port number by 100 for each additional server to avoid port conflicts. Do not use port 7501 because that port is used by the administrative server. Click **OK**.
6. For each additional server, click the server name (such as `SASServer2`) and then click **Server Start**. (This is the right-most option on the **Configuration** tab.)
7. In the **Arguments** text field, change the values for the following parameters:
 - `Dsas.server.name=SASServer2`
 - `Dsas.auto.publish.port=7101`

Note: Do not add these parameters. You must change the existing values for these parameters to the values you specified in step 5.

8. Click **Activate Changes**.

Create a Cluster and Assign the Servers

Create a cluster from the newly cloned servers. Leave SASServer1 as the stand-alone server. Follow these steps:

1. Select **Lock & Edit**.
2. In the Domain Structure panel, select **Environment > Clusters** and then click **New**.
3. On the Create a New Cluster page, provide a name for the cluster such as SASCluster1. The other values can remain unchanged. Click **OK**.
4. Click **SASCluster1**.
5. Click **Servers** and then click **Add**.
6. On the Add a Server to Cluster page, select a cloned server from the list and then click **Next**.
7. Add additional servers to the cluster in the same way.
8. Click **Activate Changes**.

Reconfigure Resources

Target the Mail Session to the Cluster

The SAS Mail Session must be targeted to all servers in the cluster. To target the SAS Mail Session, follow these steps:

1. Select **Lock & Edit**.
2. In the Domain Structure panel, select **Services > Mail Sessions**.
3. Click **SASMailSession**.
4. Click the **Targets** tab. Select the check box for each server in the cluster and the stand-alone server. Click **Save**.
5. Click **Activate Changes**.

Reconfigure the JMS Resources

SAS BI Dashboard Event Generator uses the JMS resources that the SAS Shared Services Web application makes available for the SAS Web applications. The SAS Deployment Wizard configured the JMS resources for the first managed server, SASServer1. SAS recommends leaving SASServer1 as the stand-alone server. If you choose to use a different server as the stand-alone server, then reconfigure the JMS resources to the stand-alone server by following these steps:

1. Select **Lock & Edit**.
2. In the Domain Structure panel, select **Services > Messaging > JMS Modules**.
3. Click **sasJmsSystemResource**. Click **Targets**.
4. Select the check box for the stand-alone server. Do not clear the check box for **SASServer1**. The resource is removed from SASServer1 later in this task.
5. Click **Activate Changes**.

6. Select **Lock & Edit**.
7. In the Domain Structure panel, select **Services > Messaging > JMS Servers**.
8. Click the **SASJMSServer**. Click **Targets**.
9. Change **Target** to the server that you selected in step 4.
10. Click **Activate Changes**.
11. Select **Lock & Edit**.
12. In the Domain Structure panel, select **Services > Messaging > JMS Modules** again.
13. Click **sasJmsSystemResource**. Click **Targets**.
14. Clear the **SASServer1** check box and click **Save**.
15. Click **Activate Changes**.

Target the JDBC Data Source

The SAS Shared Services Web application uses JDBC to provide the SharedServices data source. The SharedServices data source must be configured for the stand-alone server that will run the SAS Shared Service Web application. If the stand-alone server was changed from SASServer1, then follow these steps:

1. Select **Lock & Edit**.
2. In the Domain Structure panel, select **Services > JDBC > Data Sources**.
3. Click **SharedServices**.
4. Click the **Targets** tab, select the check box for the stand-alone server that will run the SAS Shared Services Web application, and then click **Save**.
5. Click **Activate Changes**.

Deploy the SAS Web Applications to the Cluster

All the SAS Web applications were deployed to SASServer1. To have the applications available from the cluster and some applications available from the stand-alone server, the applications must be targeted to the correct servers. Follow these steps:

1. Select **Lock & Edit**.
2. In the Domain Structure panel, click **Deployments**.
3. For each SAS Web application, perform the following steps:
 - a. Click the **Targets** tab.
 - b. Select the check box for the top-level EAR file and click **Change Targets**.
 - c. Select the check box for the target server. For applications that can be clustered, select the check box for each server in the cluster. Click **Save**.

4. Repeat the previous steps for each EAR file and then click **Activate Changes**. For information about which applications to deploy on each server, see the following table:

| Application | Target |
|--|---|
| <code>sas.bidashboard4.2.ear</code> | Target this EAR file to each server in the cluster if the <code>sas.eventsgenerationframework.war</code> file was removed from the EAR file. Otherwise, target this EAR file to the stand-alone server. |
| <code>sas.eventsgenerationframework.war</code> | If this WAR file is extracted from <code>sas.bidashboard4.2.ear</code> , then deploy this WAR file to the stand-alone server. |
| <code>sas.packageviewer.ear</code> | Target all servers in the cluster. |
| <code>sas.portal4.2.ear</code> | Target all servers in the cluster. |
| <code>sas.shared9.2.ear</code> | Target the stand-alone server. |
| <code>sas.storedprocess9.2.ear</code> | Target all servers in the cluster. |
| <code>sas.themes.ear</code> | Do not target this application to any servers. Later tasks in this document describe how to deploy SAS Themes Web application content to an HTTP server. |
| <code>sas.webdocmd9.2.ear</code> | Target all servers in the cluster. |
| <code>sas.webolapviewer4.2.ear</code> | Target all servers in the cluster. |
| <code>sas.webreportstudio4.2.ear</code> | Target all servers in the cluster. |
| <code>sas.wip.apps9.2.ear</code> | Target all servers in the cluster. |
| <code>sas.wip.scs9.2.ear</code> | Target the stand-alone server. |
| <code>sas.wip.services9.2.ear</code> | Target all servers in the cluster. |

Configure a Reverse Proxy

One of the purposes of a reverse proxy is to perform load balancing of HTTP requests to the servers in the WebLogic Server cluster. Load balancing can be performed through hardware or software. From a software perspective, WebLogic Server supports a wide range of plug-ins for HTTP Servers to perform load balancing. The information in this section supplements the WebLogic Server

documentation that is provided in [Using Web Server Plug-Ins with WebLogic Server](#). To configure Apache HTTP Server with the WebLogic Server plug-in, follow these steps:

1. Use the following table to locate the correct plug-in for your operating system and architecture:

| Operating System | Location |
|------------------|--|
| Solaris | WL_HOME/weblogic92/server/plugin/solaris/sparc WL_HOME/weblogic92/server/plugin/solaris/sparc/largefile WL_HOME/weblogic92/server/plugin/solaris/x86 WL_HOME/weblogic92/server/plugin/solaris/x86/largefile |
| Windows 32-bit | WL_HOME/weblogic92/server/plugin/win/32 |
| HP-UX 11i | WL_HOME/weblogic92/server/plugin/hpux11/IPF64 |

After locating the correct directory, choose the correct file:

| Apache HTTP Server Version | Regular Encryption | 128-bit Encryption |
|-------------------------------|--------------------|--------------------|
| Standard Apache Version 2.0.x | mod_wl_20.so | mod_wl128_20.so |
| Standard Apache Version 2.2.x | mod_wl_22.so | mod_wl128_22.so |

2. Copy the plug-in to the `APACHE_HOME/modules` directory.
3. Edit the `APACHE_HOME/conf/httpd.conf` file so that Apache HTTP Server:
 - loads the WebLogic Server plug-in
 - configures the plug-in with information about the WebLogic Server
 - uses the plug-in for the SAS Web applications

Add the LoadModule, IfModule, and Include directives to the `httpd.conf` file as shown in the following example:

```
LoadModule weblogic_module      modules/mod_wl_22.so

<IfModule mod_weblogic.c>
    Include conf/weblogic.conf
</IfModule>
```

4. Create the `APACHE_HOME/conf/weblogic.conf` file as shown in the following example:

```
WLLogFile "/opt/apache2/logs/wlproxy.log"
KeepAliveEnabled ON
KeepAliveSecs 120
Idempotent OFF
WLIOTimeoutSecs 600
Debug ALL
DebugConfigInfo ON

# sasserver1 is a stand-alone server
# sasserver2 and sasserver3 are cluster members

MatchExpression /SASContentServer
WebLogicHost=sasserver1.example.com|WebLogicPort=7001|Debug=ON
```

```

MatchExpression /SASBIDashboard
WebLogicHost=sasserver1.example.com|WebLogicPort=7001|Debug=ON

# If sas.eventsgenerationframework.war is extracted from the
# sas.bidashboard4.2.ear and deployed separately, then comment
# the previous entry that uses the stand-alone server, and
# uncomment the following entry that uses the cluster. Take
# note of the cookie name that uses SASLogon.sessionID.
#MatchExpression /SASBIDashboard
#WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|WLCookieName=SASLogon.sessionID|SetHandler=weblogic-handler

MatchExpression /SASPortal
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|WLCookieName=SASPortal.sessionID|SetHandler=weblogic-handler

MatchExpression /SASSharedApps
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|WLCookieName=SASSharedApps.sessionID|SetHandler=weblogic-handler

MatchExpression /SASWebReportStudio
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|WLCookieName=SASWebReportStudio.sessionID|SetHandler=weblogic-handler

MatchExpression /SASWebOLAPViewer
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|WLCookieName=SASWebOLAPViewer.sessionid|SetHandler=weblogic-handler

MatchExpression /SASStoredProcess
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|WLCookieName=SASStoredProcess.sessionID|SetHandler=weblogic-handler

MatchExpression /SASWebDoc
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|WLCookieName=SASWebDoc.sessionID|SetHandler=weblogic-handler

MatchExpression /SASPreferences
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|WLCookieName=SASPreferences.sessionID|SetHandler=weblogic-handler

MatchExpression /SASLogon
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|WLCookieName=SASLogon.sessionID|SetHandler=weblogic-handler

MatchExpression /SASSharedPortlets
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|SetHandler=weblogic-handler

MatchExpression /SASAdmin
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|WLCookieName=SASAdmin.sessionID|SetHandler=weblogic-handler

MatchExpression /SASWIPServices
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201
|SetHandler=weblogic-handler

# If the SAS Themes Web application is not served statically as
# described later in this document, then uncomment the following entry.
#MatchExpression /SASTheme_default

```

```
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201  
|SetHandler=weblogic-handler
```

Perform Common Configuration Tasks

To complete the WebLogic Server clustering, perform the tasks in "Chapter 6 – Common Tasks."

Chapter 5 — IBM WebSphere Application Server Clustering

IBM WebSphere Application Server can be clustered by using the administrative console or by scripting with the wsadmin tool. The starting point for this configuration is a WebSphere Application Server that has been configured by the SAS Deployment Wizard. The wizard configures WebSphere Application Server to provide the resources needed by the SAS Web applications such as mail, JDBC, and JMS. The wizard can also deploy the SAS Web applications to a server instance. The following list identifies the high-level steps for deploying the SAS Web applications on a WebSphere Application Server cluster:

1. Create a server template from the application server that was configured by the wizard.
2. Create a stand-alone server instance for the SAS Web applications that cannot be clustered.
3. Create a cluster and populate it with servers created with the server template.
4. Configure JVM options for the clustered servers and the stand-alone server.
5. Deploy the SAS Web applications to the cluster.
6. Configure a reverse proxy to perform load balancing.
7. Perform the common configuration tasks.

When configuring a cluster of WebSphere Application Servers, do not enable HTTP memory-to-memory replication. The SAS Web applications can access large data structures such as data sets. Memory-to-memory replication would reduce performance.

Create a Server Template from SAServer1

When the SAS Deployment Wizard is used to configure WebSphere Application Server, an application server, SAServer1, is configured with all the resources that are needed by the SAS Web applications. This task describes how to use SAServer1 as a server template to reduce the effort that is required to configure additional servers for the SAS Web applications and the cluster. Follow these steps:

1. Select **Servers > Application Servers > Templates** and then click **New**.
2. Select the radio button for SAServer1 and click **OK**.
3. Provide a name, such as SAServerTemplate, and a description. Click **OK**.

Create a Stand-alone Server Instance

A Web application server instance is needed for the SAS Web applications that cannot be clustered. To create the stand-alone WebSphere Application Server instance that is based on the SAServerTemplate, follow these steps:

1. Select **Application servers** and click **New**.
2. Select the node and provide a name, such as SAServer2. Click **Next**.
3. Select the Server Template for the SAServerTemplate and click **Next**.

4. Select **Generate Unique Ports** and then click **Next**.
5. Click **Finish** and then save the changes to the master configuration.

Create the Cluster and Add Servers

To create a cluster and add servers to it, follow these steps:

1. Select **Clusters** and click **New**.
2. Provide a name, such as SASCluster1, and make sure that the **Prefer local** and the **HTTP session memory-to-memory replication** check boxes are cleared. Click **Next**.

Note: SAS Web applications do not use Enterprise Java Beans, and do not support session level failover.

3. On the **Create first cluster member** page, select **Create the member by converting an existing application server**, and then select the server instance. Click **Next**.

Note: You can also create a server instance as you create the cluster.

4. You can add server instances to the cluster. To add a server, provide the member name, and make sure that **Generic Unique HTTP Ports** is selected. Click **Add Member**.

Configure JVM Arguments

Each server instance inherits the JVM configuration from the SASServerTemplate. Some JVM arguments must be modified for each server instance. The **sas.auto.publish.port** JVM argument must be modified to match the HTTP port for the server instance. Follow these steps:

1. Select **Application servers**, and then click the link for the server instance.
2. Click **Ports** and make a note of the value for **WC_defaulthost**.
3. Select **Java and Process Management > Process Definition**.
4. Select **Java Virtual Machine** from the right panel of the Web page.
5. Change the value for `-Dsas.auto.publish.port` to match the value of **WC_defaulthost**.

Target SAS Web Applications to the Cluster

Because the starting configuration for WebSphere Application Server was a single server, SASServer1, all the SAS Web applications are targeted to that server. Now that the cluster is configured, some of the SAS Web applications need to be reconfigured to use the stand-alone server and the others to use the cluster.

If you have separated the SAS BI Dashboard Event Generator Web application from the SAS BI Dashboard EAR file, then use the administrative console to install the repacked WAR and EAR files and also remove the SAS BI Dashboard Web application.

To target each SAS Web application, follow these steps:

1. Select **Enterprise Applications**, and then click the respective SAS 9.2 Web application.
2. Select **Manage Modules**, and then select each check box for all the modules.

3. From the **Clusters and Servers** field, select the cluster or server instance, and then click **Apply**. The Server field for each module is updated with your choices.
4. Click **OK** to target the application to the cluster or server.
5. Repeat these steps for each SAS Web application to target the cluster or stand-alone server.

Configure a Reverse Proxy

One option for load balancing HTTP requests to the cluster is to configure IBM HTTP Server as a reverse proxy. For more information about performing this configuration, see [Configuring IBM HTTP Server Reverse Server 9.2 Web Applications Deployed on IBM WebSphere Application Server](#) or IBM documentation.

For sites that proxy the SAS Content Server application, some WebSphere Application Server settings must be changed so that SAS Content Server is available from IBM HTTP Server, and so that the IBM HTTP Server accepts content, such as attachments, with HTTP requests. Follow these steps in the WebSphere administrative console:

1. Change the virtual host for the SAS Content Server application to the default host.
2. Select **Servers > Web servers > webserver1 > Plug-in properties > Request and Response**.
3. Select the **Accept content for all requests** check box.

After you configure the reverse proxy, regenerate the Web server plug-in configuration file and propagate it.

Perform Common Configuration Tasks

To complete WebSphere Application Server clustering, perform the tasks in "Chapter 6 — Common Tasks."

Chapter 6 — Common Tasks

Load Balancing for SAS Web Applications That Are Not Proxied

Each section in this document that provides steps for configuring your Web application server also includes steps for configuring a reverse proxy. The reverse proxy also performs load balancing of requests for the SAS Web applications. However, the SAS Web applications identified in the following table are deployed on the cluster, but they are not proxied. Therefore, the applications are not load balanced.

SAS recommends deploying an additional reverse proxy or some other mechanism or device to perform load balancing of requests for the remaining SAS Web applications. The applications and URLs are listed in the following table:

| SAS Web Application | URL |
|---|---------------------|
| SAS Web Infrastructure Platform Services 9.2 | /SASWIPServices |
| SAS Web Infrastructure Platform Client Access 9.2 | /SASWIPClientAccess |
| SAS Web Infrastructure Platform SOAP Services 9.2 | /SASWIPSoapServices |

In the SAS Enterprise Business Intelligence Server deployment that this document describes, the applications in the previous table are accessed exclusively by other SAS Web applications. Because these applications offer no end-user interface, there is no advantage to exposing the URL access points from the public interface of the reverse proxy, and preventing access from the public interface of the reverse proxy increases security. However, some deployments, such as SAS solutions or custom Web applications, might require providing access to these applications from the public reverse proxy.

Configure Logging for SAS Web Applications in a Cluster

The logging location for each SAS Web application is stored in metadata. Metadata identifies a single pathname for each SAS Web application. As a result, when multiple instances of each SAS Web application open a log file, each instance attempts to open an identical pathname. This causes resource contention for vertical clusters. The following task describes how to configure the JVM for each Web application server with a unique server identity and then to configure the logging location that is stored in metadata to use the unique server identity. This resolves the resource contention for vertical clusters, but is also useful for horizontal clusters if the log files are written to shared storage, or if they are aggregated to a central location. Follow these steps:

1. Modify the JVM options for each instance of the Web application server to include the following options:

```
-Dsas.appserver.instance.id=SASServern
-Dsas.wrs.keyUserActionLog.filename=
SASWebReportStudio4.2_KeyActions_SASServern.log
```

Do not restart each Web application server instance until this task is complete.

2. In SAS Management Console, select **Environment Management > Foundation Services Manager**.

3. For each of the applications other than Remote Services, follow these steps:
 - a. Select the application, and then select **Core > Logging Service**.
 - b. Right-click **Logging Service** and select **Properties**.
 - c. Select the **Service Configuration**, tab and then click **Configuration**.
 - d. Select the **Outputs** tab.
 - e. Select SAS_LS_FILE, and then click **Edit**.
 - f. Change the File parameter to use the unique identifier. The following example orders the logs by the Web application server instance:


```
/opt/SAS/Config/Lev1/Web/Logs/${sas.appserver.instance.id}/
SASBIDashboard4.2.log
```
 - g. Click **OK** on each of the three dialog boxes to return to SAS Management Console.
4. Restart each Web application server instance.

Deploy SAS Themes to an HTTP Server

In all the topologies described in this document, SAS recommends deploying the SAS Themes content to an HTTP server. By serving the SAS Themes content from an HTTP server, you shift the processing load of serving static HTML files from the Web application server to the HTTP server. This task describes how to perform this deployment on Apache HTTP Server. Performing this reconfiguration on other HTTP servers is similar. To configure Apache HTTP Server to serve the static HTML files for SAS Themes, follow these steps:

1. In a temporary directory, extract the contents of the *SAS-config-dir/Levn/Web/Staging* /sas.themes.ear file:


```
jar xf /opt/SAS/Config/Lev1/Web/Staging/sas.themes.ear
```

The sas.theme.default.war file is extracted.
2. Create a new directory that is named *APACHE_HOME/htdocs/SASTheme_default*.
3. Extract the sas.theme.default.war file into the *APACHE_HOME/htdocs/SASTheme_default* directory.

Change the Connections for the SAS Web Applications

After SAS Themes is deployed to the HTTP server and the SAS Web applications are distributed to different servers, information about to access the applications such as host and port must be updated in SAS metadata. Change the connection information to a URL that includes the Web application server host name and port number for the Web application server that is hosting the SAS Web application. Information about the SAS Themes application must be changed to the HTTP server host name and port.

To change the connection access point, follow these steps in SAS Management Console:

1. Select **Application Management > Configuration Manager**.
2. Right-click on the SAS Web application you want to reconfigure, and select **Properties**.

3. Click the **Connection** tab, set **Host Name** and **Port Number** to the host name and port number of the Web application server that is hosting the SAS Web application (use the HTTP server host name and port for SASTheme_default), and then click **OK**.

Change the Connection for SAS Content Server

If the SAS Content Server application (`sas.wip.scs9.2.ear`) is not deployed on the original Web application server and port, or the connection is proxied by a reverse proxy, then you must reconfigure SAS metadata with the connection information. This change is similar to the change required for each of the SAS Web applications. SAS metadata must be changed to identify the host name and port of the Web application server or reverse proxy that is hosting the SAS Content Server application. To reconfigure the host name and port of the SAS Content Server in SAS metadata, follow these steps in SAS Management Console:

1. Select **Server Manager > SAS Content Server**.
2. Right-click the **Connection: SAS Content Server** icon in the right panel and select **Properties**.
3. Click the **Options** tab and set the **Host name** and **Port number** fields to the host name and port number of the Web application server or reverse proxy that is hosting the `sas.wip.scs9.2.ear` EAR file for SAS Content Server.
4. Click **OK**.
5. Select the **Folders** tab.
6. Right-click the **SAS Folders** icon at the root of the folder tree in the left pane and select **Properties**.
7. Select the **Content Mapping** tab and use the Server menu to select **SAS Content Server**. The URL field then shows the reverse proxy host name and port.
8. Click **OK**.

Change the WebDAV Repository URL

Just as in the previous step, if the SAS Content Server application (`sas.wip.scs9.2.ear`) is not deployed on the original Web application server and port or uses a reverse proxy, then you must reconfigure SAS metadata with the connection information for the WebDAV repository. The following applications use SAS metadata to identify the connection information for the SAS Content Server:

- Remote Services
- SASPackageViewer4.2 Local Services
- SASPortal4.2 Local Services
- SASStoredProcess9.2 Local Services
- SASWebReportStudio4.2 Local Services

To reconfigure the WebDAV URL for the applications, follow these steps in SAS Management Console:

1. Select **Environment Management > Foundation Services Manager**.
2. Select the application, and then select **Core > Information Service**.
3. Right-click **Information Service** and select **Properties**.
4. On the **Information Service Properties** dialog box, click the **Service Configuration** tab, and then click **Configuration**.
5. On the **Information Service Configuration** dialog box, click the **Repositories** tab.
6. Select **WebDAV**, and then click **Edit**.
7. Change the **Host** and **Port** values to the host name and port of the Web application server that is hosting the `sas.wip.scs9.2.ear` EAR file.
8. Click **OK** to close the **Information Service Configuration** dialog box.
9. Click **OK** to close the **Information Service Properties** dialog box.
10. Restart SAS Remote Services and the Web application server that is hosting the SAS Content Server application.

Recommended Reading

The following URLs are current as of October 2009:

Oracle Corporation. 2007. "Using Web Server Plug-Ins with WebLogic Server." Available at http://download.oracle.com/docs/cd/E13222_01/wls/docs92/pdf/plugins.pdf.

SAS Institute Inc. 2009. "Configuring IBM HTTP Server as a Reverse Proxy Server for SAS®9.2 Web Applications Deployed on IBM WebSphere Application Server." Cary, NC: SAS Institute Inc. Available at <http://support.sas.com/resources/thirdpartysupport/v92/appservers/webspheredoc.html>

Your Turn

We welcome your feedback.

- If you have comments about this book, please send them to **yourturn@sas.com**. Include the full title and page numbers (if applicable).
- If you have comments about the software, please send them to **suggest@sas.com**.

SAS® Publishing Delivers!

Whether you are new to the work force or an experienced professional, you need to distinguish yourself in this rapidly changing and competitive job market. SAS® Publishing provides you with a wide range of resources to help you set yourself apart. Visit us online at support.sas.com/bookstore.

SAS® Press

Need to learn the basics? Struggling with a programming problem? You'll find the expert answers that you need in example-rich books from SAS Press. Written by experienced SAS professionals from around the world, SAS Press books deliver real-world insights on a broad range of topics for all skill levels.

support.sas.com/saspress

SAS® Documentation

To successfully implement applications using SAS software, companies in every industry and on every continent all turn to the one source for accurate, timely, and reliable information: SAS documentation. We currently produce the following types of reference documentation to improve your work experience:

- Online help that is built into the software.
- Tutorials that are integrated into the product.
- Reference documentation delivered in HTML and PDF – free on the Web.
- Hard-copy books.

support.sas.com/publishing

SAS® Publishing News

Subscribe to SAS Publishing News to receive up-to-date information about all new SAS titles, author podcasts, and new Web site features via e-mail. Complete instructions on how to subscribe, as well as access to past issues, are available at our Web site.

support.sas.com/spn



THE
POWER
TO KNOW®