



THE
POWER
TO KNOW.

SAS[®] 9.2 Web Applications Clustering Second Edition



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

SAS® 9.2 Web Applications: Clustering, Second Edition

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

3rd electronic book, June 2010

4th electronic book, December 2010

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.

Table of Contents

What's New in SAS 9.2 Web Applications: Clustering	1
Overview	1
Enhanced Support for SAS Web Report Studio Scheduling	1
Enhanced Support for SAS Content Server Clustering.....	1
New Strategy to Proxy All SAS Web Applications	1
SAS BI Portlets 4.2 Limitation	1
Documentation Enhancements	1
Chapter 1 — Overview	3
Chapter 2 — SAS Middle Tier Limitations	5
SAS Information Delivery Portal and SAS BI Portlets	5
Java Messaging Service Servers	5
Strategies for Overcoming Limitations	5
Deploy a Stand-alone Web Application Server.....	5
Hot and Cold Standby Strategy	5
High Availability Software Strategy	6
Chapter 3 — Common Pre-Configuration Tasks	7
Purpose	7
SAS BI Dashboard Event Generator	7
SAS BI Dashboard	9
Access to Configuration Files for SAS Web Applications on Multiple Machines	9
Reconfigure the SAS Content Server Repository.....	9
Chapter 4 — JBoss Clustering	15
Create Additional JBoss Server Instances	16
Configure a Load Balancing HTTP Server	18
Configure JBoss to Use mod_jk	20
Perform Common Configuration Tasks	21
Chapter 5 — Oracle WebLogic Server 10.3 Clustering	23
Pack the Domain and Unpack on Additional Machines	24
Add Additional Machines	25
Clone the Managed Server	25
Create a Cluster and Assign the Servers.....	26
Reconfigure Resources	26
Target the Mail Session to the Cluster	26
Reconfigure the JMS Resources	27
Target the JDBC Data Source.....	27
<i>Deploy the SAS Web Applications to the Cluster</i>	28
Configure a Load Balancing HTTP Server	29
SAS Web Report Studio 4.2 Specific Update	32
Perform Common Configuration Tasks	32

Chapter 6 — IBM WebSphere Application Server Clustering	33
Add Additional Machines	34
Create a Server Template from SASServer1	34
Create a Stand-alone Server Instance	35
Create the Cluster and Add Servers	35
Configure JVM Arguments and the Cookie Name	35
Target SAS Web Applications to the Cluster	36
Add SASCluster as a SAS Messaging Bus Member	36
Configure a Load Balancing HTTP Server	37
Perform Common Configuration Tasks	37
Chapter 7 — Common Tasks	39
Configure Logging for SAS Web Applications in a Cluster	39
Deploy SAS Themes to an HTTP Server	39
Change the Connections for the SAS Web Applications	40
Change the Connection for SAS Content Server	40
Change the WebDAV Repository URL	41
Recommended Reading	42

What's New in SAS 9.2 Web Applications: Clustering

Overview

The SAS 9.2 Enterprise Business Intelligence Web applications have the following changes and enhancements:

- enhanced support for SAS Web Report Studio scheduling
- enhanced support for SAS Content Server clustering
- new strategy to proxy all SAS Web applications
- SAS BI Portlets 4.2 limitation for vertical clusters
- enhanced support for SAS OnlineDoc for the Web clustering
- documentation enhancements

Enhanced Support for SAS Web Report Studio Scheduling

In the third maintenance release for SAS 9.2, SAS Web Report Studio is enhanced to support scheduling jobs with in-process scheduling in a clustered environment. In the 9.2 release and previous maintenance releases, the in-process scheduling feature could not be used in a clustered environment. Additional scheduling software such as Platform Suite for SAS was needed.

Enhanced Support for SAS Content Server Clustering

In the third maintenance release for SAS 9.2, the SAS Content Server Web application can participate in Web application server clustering. The application must be reconfigured to use a database for repository storage and to migrate content from the file system repository. Details about how to perform these actions are included in "Reconfigure the SAS Content Server Repository" on page 9.

New Strategy to Proxy All SAS Web Applications

In previous releases of this document, an effort was made to proxy only the SAS Web applications that had an end-user interface, such as SAS Web Report Studio or SAS Information Delivery Portal. In this release, all SAS Web applications are made available through a load-balancing HTTP server. This strategy is no less secure, because all SAS Web applications are secured with the SAS Logon Manager.

SAS BI Portlets 4.2 Limitation

In the third maintenance release for SAS 9.2, the 4.2 release of SAS BI Portlets cannot be configured in a vertical cluster. Only one instance of SAS BI Portlets can be configured on each machine. As a result, SAS BI Portlets can be configured on each machine in a horizontal cluster. This restriction is removed with the 4.3 release of the SAS Enterprise Business Intelligence Web applications.

Enhanced Support for SAS OnlineDoc for the Web

In previous versions of this document, the SAS OnlineDoc for the Web application was identified as having a limitation that prevented deploying it to a Web application server cluster. There was not a

technical limitation to clustering the application, just the expectation that high availability and scalability of this Web application is not a business priority. In this release, the sample topology graphics continue to show SAS OnlineDoc for the Web deployed to a stand-alone server. If the resources are available and high availability and scalability are needed, then deploy the SAS OnlineDoc for the Web to the cluster.

Documentation Enhancements

The following documentation enhancements are made to *SAS 9.2 Web Applications: Clustering*:

- Chapter 2 - Middle Tier Limitations is revised to provide more detail about the limitations for clustering the SAS Web applications. Information about how to counter the limitations is moved from this chapter to Chapter 3 - Common Pre-Configuration Tasks.
- Chapter 3 - Common Pre-Configuration Tasks is a new chapter. This chapter provides detailed steps for suggested strategies to counter the limitations for clustering the SAS Web applications.

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 Web application servers 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 a load-balancing HTTP server 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 Information Delivery Portal and SAS BI Portlets

The SAS Information Delivery Portal uses the SAS BI Portlets to display content on portal pages. The SAS BI Portlets can be deployed to a horizontal cluster. However, the 4.2 release of the SAS BI Portlets cannot be deployed in a vertical cluster due to a technical limitation. This limitation is removed in the 4.3 release of SAS BI Portlets.

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 and the SAS BI Dashboard Event Generator Web application. Each Web application server vendor offers different strategies for providing JMS resources in a clustered deployment. As a result, these differences have an impact on how to deploy SAS Shared Services and SAS BI Dashboard Event Generator. Specifically, SAS BI Dashboard Event Generator is an application that uses JMS resources for monitoring SAS BI Dashboard indicators. The SAS BI Dashboard Event Generation Web application cannot be clustered. Therefore, 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.

WebSphere Application Server Network Deployment can provide JMS resources in a cluster and also make those resources available to the cell. As a result of the cell-level scope provided by WebSphere Application Server, SAS Shared Services is deployed to the cluster. Both JBoss and WebLogic Server can be configured to provide JMS services similarly. For more information, see the vendor documentation.

Strategies for Overcoming Limitations

Deploy a Stand-alone Web Application Server

As indicated in the previous section, 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 strategy that is described in this document is to deploy a stand-alone Web application server and to deploy the SAS BI Dashboard Event Generation Web application to it. This document also suggests deploying the SAS Shared Services Web application to the stand-alone server for JBoss and WebLogic Server in order to keep the JMS resources configuration and the deployment simple. For sites with more sophisticated needs and topology, the SAS Shared Services Web application can be deployed to clustered servers if JMS resources are available to the clustered application server instances.

Hot and Cold Standby Strategy

This strategy can be used to meet high availability requirements for the SAS Web applications that cannot be clustered. The strategy is to configure two machines, each with a stand-alone Web

application server and the stand-alone 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\

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. For information about reconfiguring the location of the SAS BI Dashboard directory, see "SAS BI Dashboard" on page 9.

High Availability Software Strategy

Another alternative for ensuring the high availability of the 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 SAS Web applications that are deployed on it.

Chapter 3 — Common Pre-Configuration Tasks

Purpose

This chapter describes steps that are common to all Web application servers, and that are required to perform the Web application server-specific configuration steps that are described later in this document.

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. You can keep the applications in the same EAR file and provide high availability with a hot and cold standby strategy, or use high availability software to run one active instance of both SAS BI Dashboard and the SAS BI Dashboard Event Generator.

However, 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 multiple instances of SAS BI Dashboard to the cluster. This strategy is used through the rest of this document.

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 the repackaged `sas.bidashboard4.2.ear` and `sas.eventsgenerationframework.war` applications.

If the event generation Web application is moved to another machine, the contents of the `SAS-config-dir/Levn/AppData/SASBIDashboardEventGen4.2` directory must be moved to the other machine as well. These files are used to store configuration and state information.

For deployments that use WebSphere Application Server, make sure that when the SAS Web applications are reinstalled, that the class loader order is set to **Classes loaded with the application class loader first** for each Web module. If the class loading order is incorrect, then SAS BI Dashboard can generate HTTP 500 errors when trying to access the `com.sas.collection.StaticDictionaryInterface` class.

SAS BI Dashboard

SAS BI Dashboard uses files to store configuration information about dashboards, indicators, and so on. The default location for the files is a directory path that is local to the machine. In a clustered environment, each instance of SAS BI Dashboard must have access to the configuration information. One solution is to reconfigure the default location, *SAS-config-dir/Levn/AppData/SASBIDashboard4.2*, to a location that is available on network storage. If this default location can be made visible to all the machines that will host SAS BI Dashboard, then the remaining information in this section about changing metadata does not apply.

An alternative solution is to move the configuration files to a location on network storage, and then change the metadata associated with SAS BI Dashboard to identify the directory path to the new location. 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 value for *bid.DashboardConfig*. Set the value to a location on the shared network storage. Click **OK**.
6. Stop the application server, or the SAS BI Dashboard Web application.
7. Copy the contents of directories from the original location, *SAS-config-dir/Levn/AppData/SASBIDashboard4.2*, to the new location on the shared network storage.

Access to Configuration Files for SAS Web Applications on Multiple Machines

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. Use the same directory paths on all the machines:

- *SAS-config-dir/Levn/Web/Applications*
- *SAS-config-dir/Levn/Web/Common*
- *SAS-config-dir/Levn/Web/Temp*
- *SAS-config-dir/Levn/Web/Utilities*

If you plan to deploy more than one instance of SAS BI Dashboard, then make sure that *SAS-config-dir/Levn/AppData/SASBIDashboard4.2* is on network storage and is available from each machine that will host the SAS BI Dashboard Web application, or that the files in that directory are moved to a location on network storage and *bid.DashboardConfig* property is changed to the same location. For more information, see "SAS BI Dashboard" on page 9.

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.

Reconfigure the SAS Content Server Repository

The SAS Content Server Web application is initially deployed to use the file system for persistence. Using a single instance of a repository on a file system prevents using SAS Content Server in a

clustered environment. The following steps describe how to reconfigure SAS Content Server to use the same database storage that is used by SAS Shared Services. By default, this is SAS Table Server, but it can be another database such as Oracle. Follow these steps to preserve the file system-based WebDAV repository and copy the repository contents to the database-based WebDAV repository:

1. Stop the application server if it is not already stopped.
2. Rename the directory for the existing repository. The following example shows UNIX syntax:

```
mv SAS-config-dir/Levn/AppData/SASContentServer/Repository SAS-config-dir/Levn/AppData/SASContentServer/RepositoryFS
```

3. For each application server instance in the cluster, create a directory in the SASContentServer directory. Name the directory after the application server instance name, such as "SASServer1." The name of the directory must match the value of the `-Dsas.appserver.instance.id` JVM property that is set later in this document:

```
mkdir SAS-config-dir/Levn/AppData/SASContentServer/SASServer1
```

Note: If the application server instances are deployed on separate machines, then create the directory and parent directories on the each machine, using the application server instance name that runs on the applicable machine.

4. Each of the created directories must have a repository.xml file to identify the database vendor to be used. The file must match the database vendor that is used for SAS Shared Services. Copy the database vendor-specific file from `SAS-install-dir/SASWebInfrastructurePlatform/9.2/Static/wars/sas.svcs.scs/WEB-INF/templates` to each of the directories created in the previous step. Rename the file to repository.xml. The following example uses UNIX syntax:

```
cp /opt/SAS_92/SAS/SASWebInfrastructurePlatform/9.2/Static/wars/sas.svcs.scs/WEB-INF/templates/repository.oracle.xml Repository/SASServer1/repository.xml
```

Note: The repository.tks.xml file is used for deployments that use SAS Table Server as the data store for SAS Shared Services.

5. Specifically for JBoss only, edit the repository.xml file so that the value for the URL parameter is prefixed with the `java:` namespace. The URL parameter is used six times in the configuration file, and the change to make is shown in the following example:

```
<param name="url" value="java:sas/jdbc/SharedServices"/>
```

6. Edit each of the copied repository.xml files and set the node ID by uncommenting the `Cluster` element, and setting the value of the `id` attribute to the application server instance name. As with step 2, the value must match the value for the `-Dsas.appserver.instance.id` JVM property that is set later:

```
<!-- Cluster configuration -->
<Cluster id="SASServer1">
  <Journal class="org.apache.jackrabbit.core.journal.DatabaseJournal">
    <param name="revision" value="{rep.home}/revision.log"/>
    <param name="driver" value="javax.naming.InitialContext"/>
    <param name="url" value="sas/jdbc/SharedServices"/>
    <param name="schemaObjectPrefix" value="SAS_SCS_"/>
    <param name="schema" value="mysql"/>
  </Journal>
</Cluster>
```



```

endlocal
if [%2] EQU [exit] exit %ERRORLEVEL%

UNIX

#!/bin/sh -p
#
# JCRCopyRepository.sh
#
. `dirname $0`/../../../../level_env.sh

LAUNCHERJAR=$SASVJR_HOME/eclipse/plugins/sas.launcher.jar

UTILITIESDIR=$LEVEL_ROOT/Web/Utilities
PICKLISTS=$SAS_HOME/SASWebInfrastructurePlatform/9.2/Picklists/wars/sas.
svcs.scs/picklist
DRIVER=path-to-jdbc-driver-JAR-file
CLASSPATH=$UTILITIESDIR:$LAUNCHERJAR

" $JAVA_JRE_COMMAND " \
-classpath "$CLASSPATH" \
-Djava.system.class.loader=com.sas.app.AppClassLoader \
-Dsas.app.launch.config="$PICKLISTS" \
-Dsas.app.repository.path="$SASVJR_REPOSITORYPATH" \
-Dsas.app.class.path="$UTILITIESDIR;$DRIVER" \
-Djava.security.auth.login.config=../Common/login.config \
-Xmx256m \
-Dscs.jndi.jndiName=sas/jdbc/SharedServices \
-Dscs.jndi.jdbcUrl=jdbc-url \
-Dscs.jndi.driver=jdbc-driver-class \
-Dscs.jndi.user=database-user \
-Dscs.jndi.pwd=password \
org.apache.jackrabbit.core.JCRCopyRepository $1 $2
exit 0

```

Note: Replace the values in italics with real values. You might need to edit the script to make sure that the backslash is the last character on each line in the script and that there are no trailing spaces. For deployments that use SAS Table Server, the JDBC URL must include a constring parameter such as the following example:

```
jdbc:sastkts://hostname:2171?constring=(DSN=SharedServices)
```

8. Run the JCRCopyRepository script. Make sure that Web application server is stopped, and run the script only once. In the syntax for the command, the first argument is the old repository directory, and the second argument is the new repository directory. The following example uses UNIX syntax:

```
./JCRCopyRepository.sh SAS-config-dir/Levn/AppData/SASContentServer/
RepositoryFS SAS-config-dir/Levn/AppData/
SASContentServer/SASServer1
```

Run the script only once even if there are multiple machines or more than one SASServern directory.

Do not start the application server to check SAS Content Server until the -Dsas.appserver.instance.id JVM parameter is added to the Web application server start-

up scripts (and the `wrapper.conf` file for Windows). That parameter is used by SAS Content Server to determine the location of the `repository.xml` file.

Chapter 4 — JBoss Clustering

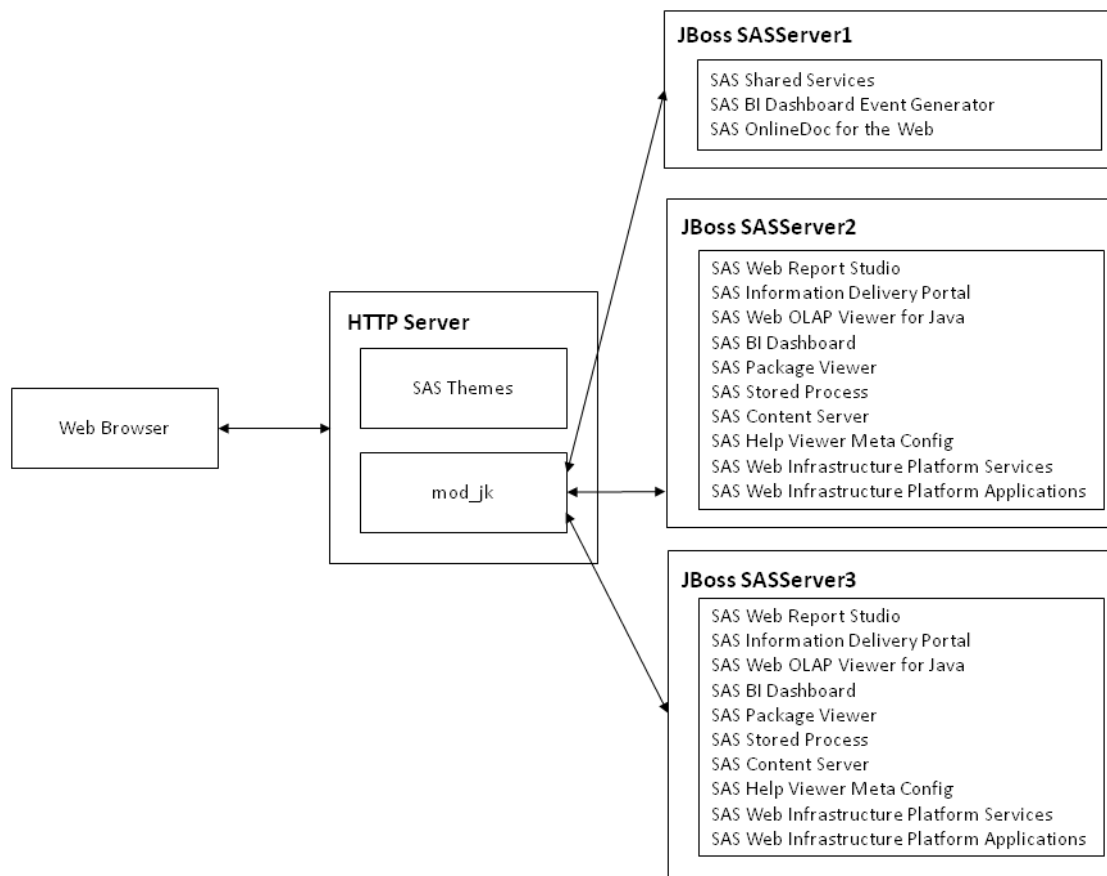
The following list identifies the necessary steps to perform clustering with JBoss Application Server 4.2:

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. Review the limitations in chapter 2 and perform the pre-configuration tasks in chapter 3.
2. Use the server created by the SAS Deployment 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.
3. Configure an HTTP server to act as a load-balancing HTTP server. This document describes using Apache HTTP Server with mod_jk.
4. Configure JBoss to work with the load-balancing HTTP server.
5. 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



Create Additional JBoss Server Instances

Based on requirements and capacity planning, copy the SAServer1 directory for each additional server and rename the directory SAServer2, SAServer3, and so on. Reconfigure each additional server to use unique ports. The following example shows the steps for modifying SAServer2:

1. Copy the contents of SAServer1 to a new directory such as SAServer2. As shown in the previous figure, SAServer1 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 SAServer2/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
<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.biportlets4.2.ear</code>	Target all servers in the cluster. For vertical cluster deployments, target 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. For vertical cluster deployments, target the stand-alone server.
<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 all servers in the cluster.
<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 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, prevent the clustered JBoss instances from creating JMS resources by removing the `SASServern\deploy\sas-jms-service-service.xml` file. Removing the file prevents the servers in the cluster from starting JMS services.
5. 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.

6. In the `JBOSS_HOME\bin` directory, copy `SASServer1.bat` to `SASServer2.bat`.
7. 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. Add `-Dsas.appserver.instance.id=SASServer2` to the `JAVA_OPTS` variable. Also add the `-Dsas.appserver.instance.id=SASServer1` parameter to the `SASServer1.bat` script. This parameter is needed by SAS Content Server and is used to configure logging later in this document.
 - d. 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 at this point 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 Load Balancing HTTP Server

HTTP requests must be distributed to the JBoss cluster members. This can be done with hardware or software. This document describes the steps that are used to configure Apache HTTP Server to load balance HTTP requests. 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:

1. Download the `mod_jk` binaries from <http://tomcat.apache.org/download-connectors.cgi>.
2. Rename the downloaded file to `mod_jk.so` and move the file to `APACHE_HOME/modules`.
3. 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>
```

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

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:

```

# 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

# stand-alone server applications
/SASSharedApps=sasserver1
/SASSharedApps/*=sasserver1
/SASSharedPortlets=sasserver1
/SASSharedPortlets/*=sasserver1
/SASSharedServices=sasserver1
/SASSharedServices/*=sasserver1
/SASWorkflowWebServices=sasserver1
/SASWorkflowWebServices/*=sasserver1
/SASWorkflowServices=sasserver1
/SASWorkflowServices/*=sasserver1

# load-balanced applications that are clustered
/SASAdmin=loadbalancer
/SASAdmin/*=loadbalancer
/SASBIDashboard=loadbalancer
/SASBIDashboard/*=loadbalancer
/SASBIPortlets=loadbalancer      # use the stand-alone server
/SASBIPortlets/*=loadbalancer    # for vertical deployments
/SASBIWS=loadbalancer
/SASBIWS/*=loadbalancer
/SASContentServer=loadbalancer
/SASContentServer/*=loadbalancer

```

```

/SASJSR168RemotePortlet=loadbalancer
/SASJSR168RemotePortlet/*=loadbalancer
/SASLogon=loadbalancer
/SASLogon/*=loadbalancer
/SASPackageViewer=loadbalancer
/SASPackageViewer/*=loadbalancer
/SASPortal/*=loadbalancer           # use the stand-alone server
/SASPortal=loadbalancer             # for vertical deployments
/SASPreferences=loadbalancer
/SASPreferences/*=loadbalancer
/SASStoredProcess=loadbalancer
/SASStoredProcess/*=loadbalancer
/SASWIPClientAccess=loadbalancer
/SASWIPClientAccess/*=loadbalancer
/SASWIPServices=loadbalancer
/SASWIPServices/*=loadbalancer
/SASWIPSoapServices=loadbalancer
/SASWIPSoapServices/*=loadbalancer
/SASWebOLAPViewer=loadbalancer
/SASWebOLAPViewer/*=loadbalancer
/SASWebDoc=loadbalancer
/SASWebDoc/*=loadbalancer
/SASWebReportStudio=loadbalancer
/SASWebReportStudio/*=loadbalancer

```

Configure JBoss to Use mod_jk

At this stage of the reconfiguration, Apache HTTP Server is prepared to load balance HTTP requests. 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 load-balancing HTTP server. 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 SASServer2. These `jvmRoute` values match the names and case that is used in the `worker.properties` and `uriworkers.properties` files. If the case does not match, then session affinity does not work.

- (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" />
```

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

Perform Common Configuration Tasks

To complete JBoss clustering, perform the tasks in "Chapter 7 – Common Tasks."

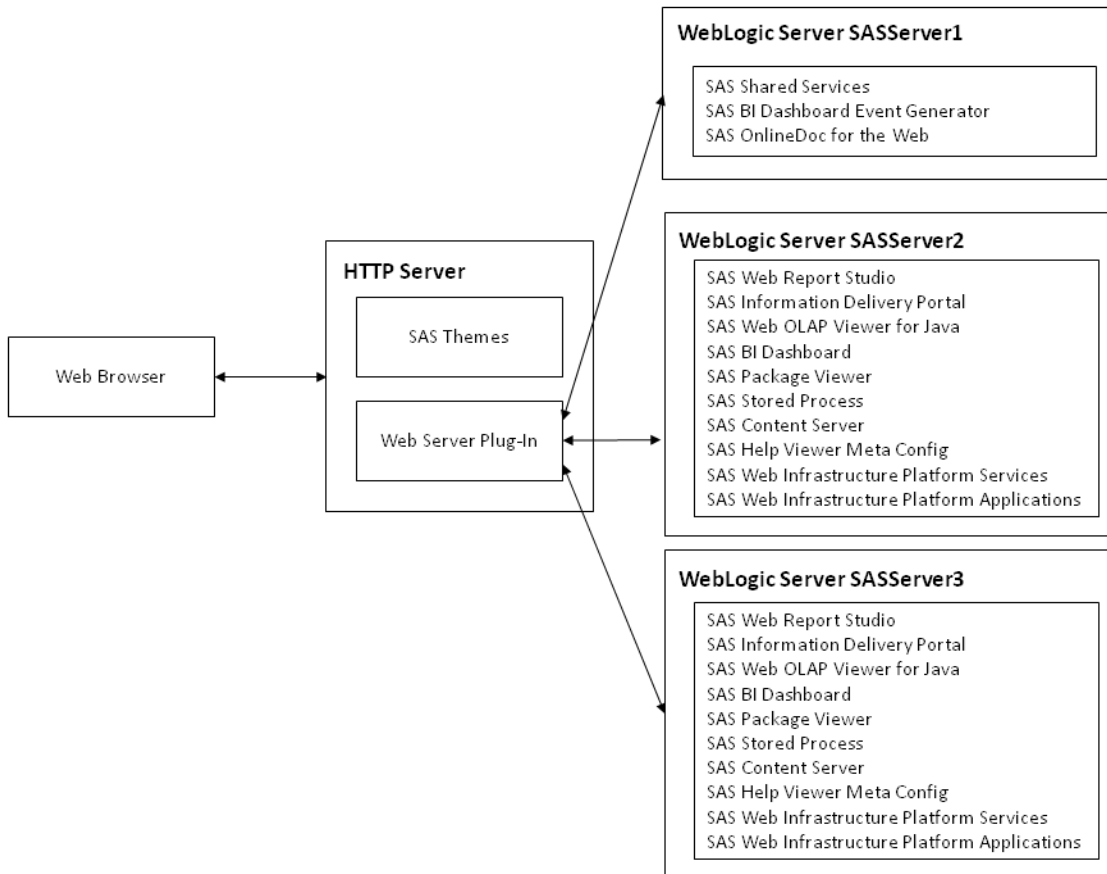
Chapter 5 — Oracle WebLogic Server 10.3 Clustering

The starting point for this configuration is an Oracle WebLogic Server 10.3 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. Review the limitations in chapter 2, and perform the pre-configuration tasks in chapter 3.
2. 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.
3. Create the cluster and assign servers.
4. Reconfigure resources such as mail, JDBC, and JMS.
5. Deploy the SAS Web applications to the cluster and some applications to the stand-alone server.
6. Configure a load-balancing HTTP server.
7. 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



Pack the Domain and Unpack on Additional Machines

For deployments that use multiple machines in the cluster, the domain used for the SAS Web applications must be copied to each additional machine in the cluster. This topology is often called a horizontal cluster. The information in this section does not apply to deployments that want a vertical cluster topology—where multiple server instances are deployed on a single machine.

To pack the domain on the original machine and unpack the domain on an additional machine, follow these steps:

1. Navigate to the WL_HOME bin directory, and use the pack command to pack SASDomain. The following example uses UNIX syntax:

```
cd /opt/WLS/wlserver_10.3/common/bin
```

```
./pack.sh -managed=true -domain=/opt/SAS/Config/Lev1/Web/SASDomain  
-template=/export/home/sas/midtier_template.jar -template_name="Managed  
Server SASDomain"
```

- The template file includes the SAS Web applications and is typically larger than 2GB. If the unpack command fails to work with the generated template file, then SAS recommends extracting the contents of the template to a temporary location, removing the applications, and then recreating the template file:

```
mkdir /tmp/work
cd /tmp/work
jar xf ~/midtier_template.jar
rm -rf _apps_
jar cf ~/midtier_template.jar .
```

- For each additional machine in the cluster, copy the midtier_template.jar file to the additional machine, and unpack the domain template. The following example uses UNIX syntax:

```
cd /opt/WLS/wlserver_10.3/common/bin

./unpack.sh -domain=/opt/SAS/Config/Lev1/Web/SASDomain
-template=temporary-location/midtier_template.jar
```

- Start the node manager on each additional machine from the SASDomain/bin directory with the startNodeManager.sh command.

For information about the pack and unpack commands, see the Oracle WebLogic Server documentation.

Add Additional Machines

If the SAS Web applications will be deployed to application server instances on additional machines, follow these steps to add a new machine in the administrative console:

- Select **SASDomain > Environment > Machines** and then click **New**.
- Enter the fully qualified domain name for the **Name** field, use the Machine OS menu to select the operating system type, and click **OK**.
- Click the link for the newly added machine and then select **Node Manager**.
- Enter the host name in the **Listen Address** field. Do not use the fully qualified domain name. Click **Save**.
- Select the machine name and then click **Monitoring**. Confirm that the **Status** field indicates **Reachable**. Do not proceed unless the additional machines are reachable.

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. Later steps describe how to cluster it with other server instances. To clone SASServer1, follow these steps:

- Use the Administration Console to shut down the managed server, SASServer1.
- In the Domain Structure panel, select **Environment > Servers**.
- In the **Servers** table, select the check box for **SASServer1** and click **Clone**.

4. On the Clone a Server page, provide a unique name (SASServer2) and port number (7101) for the new server. One way to avoid port conflicts is to increment the port number by 100 for each additional server. However, do not use port 7501 because that port is used by the administrative server, and any port management strategy is acceptable so long as it avoids port conflicts. If the server will be deployed on another machine, enter the host name in the Server Listen Address field. Click **OK**.
5. For each additional server, select the server name (such as **SASServer2**). If the server will be deployed on another machine, use the Machine menu to select the machine.
6. Click **Server Start**. (This is the right-most option on the **Configuration** tab.)
7. In the **Arguments** text field, perform the following steps:
 - a. 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 4.
 - b. Add the following parameter:

```
-Dsas.appserver.instance.id=SASServer2
```

Also add this parameter to each server instance, using the appropriate server instance ID value. The SAS Content Server Web application relies on this property, and it is also used in logging.
8. Click **Save**.

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

Reconfigure Resources

Target the Mail Session to the Cluster

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

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

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. In the Domain Structure panel, select **Services > Messaging > JMS Modules**.
2. Select sasJmsSystemResource. Click **Targets**.
3. 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.
4. Click **Save**.
5. In the Domain Structure panel, select **Services > Messaging > JMS Servers**.
6. Select SASJMSServer. Click **Targets**.
7. Change **Target** to the server that you selected in step 3.
8. Click **Save**.
9. In the Domain Structure panel, select **Services > Messaging > JMS Modules** again.
10. Select sasJmsSystemResource. Click **Targets**.
11. Clear the check box for any server in the cluster (**SASServer1**) and click **Save**.

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 all the servers in the cluster and the stand-alone server. Follow these steps:

1. In the Domain Structure panel, select **Services > JDBC > Data Sources**.
2. Select SharedServices.
3. Click **Targets** and then select the check box for the stand-alone server and each server in the cluster. Click **Save**.

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. In the Domain Structure panel, click **Deployments**.
2. For each SAS Web application, perform the following steps:
 - a. Click **Targets**.
 - 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**.

Repeat the previous steps for each EAR file and then click **Save**. 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.biportlets4.2.ear</code>	Target all servers in the cluster. For vertical cluster deployments, target 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. For vertical cluster deployments, target the stand-alone server.
<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.

Application	Target
sas.wip.scs9.2.ear	Target all servers in the cluster.
sas.wip.services9.2.ear	Target all servers in the cluster.

Configure a Load Balancing HTTP Server

There are several ways to use hardware or software for distributing HTTP requests to the servers in the cluster. This section describes how to use Apache HTTP Server. 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. In previous releases of WebLogic Server, the plug-ins were distributed with the WebLogic Server installation. For installations that use WebLogic Server 10.3, the Oracle WebLogic Server Web Server Plugins 1.1 must be downloaded from the [Oracle WebLogic Server Downloads](#) Web site.
2. After extracting the plug-ins from the ZIP file, choose the correct plug-in 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

3. Copy the plug-in to the `APACHE_HOME/modules` directory.
4. 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>
```

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

```
WLogFile "/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 /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 that SASBIDashboard uses a cookie name of SASLogon.sessionID.
#
#MatchExpression /SASBIDashboard
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201|W
LCookieName=SASLogon.sessionID|SetHandler=weblogic-handler

# stand-alone server applications
MatchExpression /SASBIDashboardEventGen
WebLogicHost=sasserver1.example.com|WebLogicPort=7001|Debug=ON

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

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

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

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

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

# load-balanced applications that are clustered
MatchExpression /SASAdmin
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201|W
LCookieName=SASBIPortlets.sessionID|SetHandler=weblogic-handler

# for vertical cluster deployments, use the stand-alone server
MatchExpression /SASBIPortlets
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201|W
LCookieName=SASBIPortlets.sessionID|SetHandler=weblogic-handler

MatchExpression /SASBIWS
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201|W
LCookieName=SASBIPortlets.sessionID|SetHandler=weblogic-handler

MatchExpression /SASContentServer
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201|W
LCookieName=SASContentServer.sessionID|SetHandler=weblogic-handler

MatchExpression /SASJSR168RemotePortlet
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201|W
LCookieName=SASJSR168RemotePortlet.sessionID|SetHandler=weblogic-handler

```

```

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

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

# for vertical cluster deployments, use the stand-alone server
MatchExpression /SASPortal
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201|
WLCookieName=SASPortal.sessionID|SetHandler=weblogic-handler

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

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

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

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

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

MatchExpression /SASWebDoc
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201|
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

# If the SAS Themes Web application is served statically as
# described later in this document, then leave the following entry
# commented. If it is not served statically, then uncomment the
# following entry.
#MatchExpression /SASTheme_default
WebLogicCluster=sasserver2.example.com:7101,sasserver3.example.com:7201|
SetHandler=weblogic-handler

```

SAS Web Report Studio 4.2 Specific Update

By default, SAS Web Report Studio 4.2 uses a special redirection filter. When used with an HTTP server, this filter must be disabled. Start SAS Management Console and then perform the following steps:

1. Select **Application Management > Configuration Manager**.
2. Right-click **Web Report Studio 4.2**, and select **Properties**.
3. Select **Advanced**, and then click **Add**.
4. Enter a property name of `App.RedirectionFilterDisabled` and a value of `true`.

Perform Common Configuration Tasks

To complete the WebLogic Server clustering, perform the tasks in "Chapter 7 — Common Tasks."

Chapter 6 — IBM WebSphere Application Server Clustering

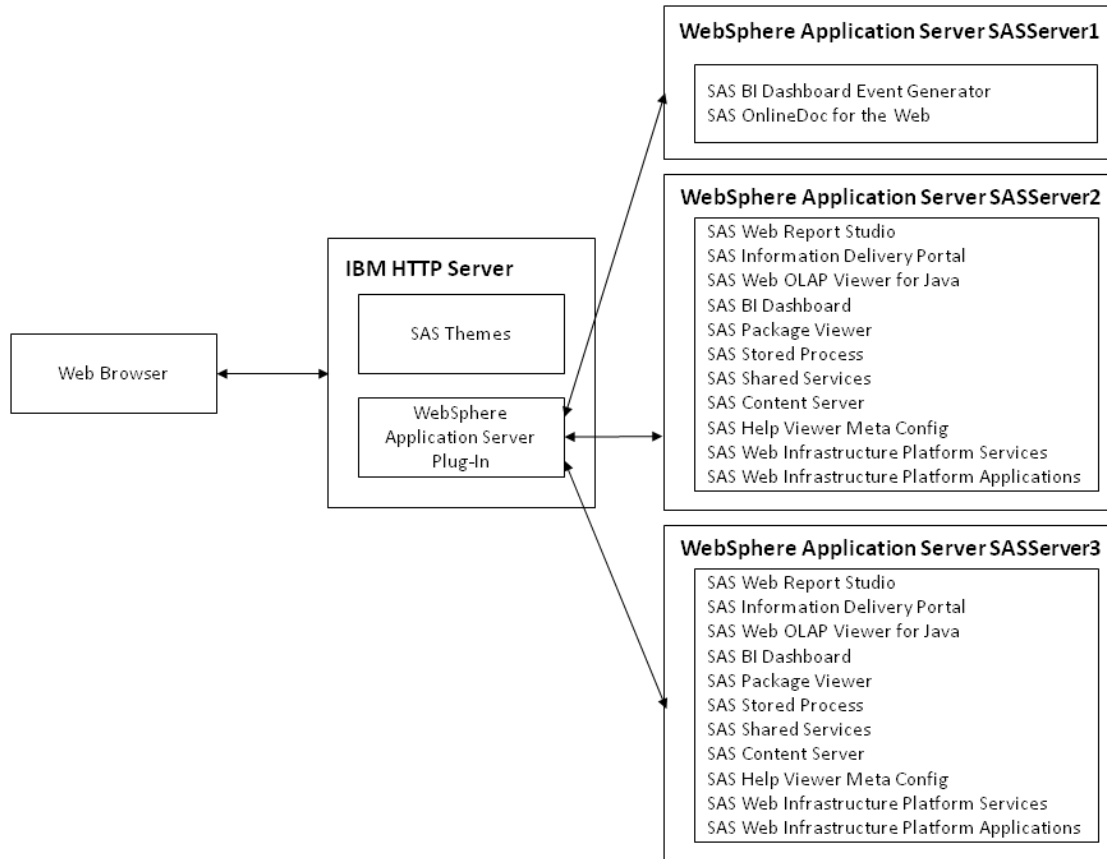
IBM WebSphere Application Server can be clustered by using the administrative console or by scripting with the wsadmin tool. The starting point for this configuration is a WebSphere Application Server that has been configured by the SAS Deployment Wizard. The wizard configures WebSphere Application Server 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. Review the limitations in chapter 2, and perform the pre-configuration tasks in chapter 3.
2. Create a server template from the application server that was configured by the wizard.
3. Create a stand-alone server instance for the SAS Web applications that cannot be clustered.
4. Create a cluster and populate it with servers created with the server template.
5. Configure JVM options for the clustered servers and the stand-alone server.
6. Deploy the SAS Web applications to the cluster.
7. Configure a load-balancing HTTP server.
8. 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.

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

Figure 3 Sample Topology for IBM WebSphere Application Server



Add Additional Machines

If the SAS Web applications will be deployed to application server instances on additional machines, then see WebSphere Application Server documentation for information about adding nodes to the cell.

Create a Server Template from SASServer1

When the SAS Deployment Wizard is used to configure WebSphere Application Server, an application server, SASServer1, is configured with all the resources that are needed by the SAS Web applications. This task describes how to use SASServer1 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 SASServer1 and click **OK**.
3. Provide a name, such as SASServerTemplate, and a description. Click **OK**.

Create a Stand-alone Server Instance

A Web application server instance is needed for the SAS BI Dashboard Event Generation Web application. To create the stand-alone WebSphere Application Server instance that is based on the SASServerTemplate, follow these steps:

1. Select **Application servers** and click **New**.
2. Select the node and provide a name, such as SASServer2. Click **Next**.
3. Select the Server Template for the SASServerTemplate 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 and the Cookie Name

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. Also, the default cookie name for Web applications is `JSessionID`. This value must be changed to a different value. SAS recommends changing the value to `SASCluster`. Follow these steps for each application server instance:

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`.
6. Add the `-Dsas.appserver.instance.id=SASServern` property for each application server instance. This value is used by the SAS Content Server Web application and is used to configure logging later in this document.

7. Select **Application servers** again, and then select **Web Container Settings > Web container**.
8. Select **Session Management** from the right panel of the Web page.
9. Make sure the check box for **Enable cookies** is selected, and then click the **Enable cookies** link.
10. Change the value for **Cookie name** to SASCluster1.

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. Target SAS BI Dashboard Event Generator to the stand-alone server. For vertical cluster deployments, target SAS BI Portlets and SAS Information Delivery Portal to the stand-alone server as well.

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.

Add SASCluster as a SAS Messaging Bus Member

SAS Shared Services needs access to JMS resources. Adding the cluster as a member of the SAS Messaging Bus can provide high availability. The following steps describe how to add the cluster as a bus member, with a single messaging engine.

One of the requirements for high availability is that the message store that is used by the messaging engine must be accessible by all the servers in the cluster. A JDBC data source can be used as a message store, or a file system location on network storage can be used as the message store. Before you begin this task, make sure that either a JDBC data source has been configured in the administration console, or know the network storage location that will be used.

To reconfigure the SAS Messaging Bus to remove SASServer1 as a bus member, and add SASCluster1 as a member, follow these steps:

1. Select **Service integration > Buses > SAS Messaging Bus**, and then select **Bus members**.
2. Click **Add**, select **Cluster**, and select the cluster name from the menu. Click **Next**.
3. Choose either **File store**, or **Data store** and click **Next**.
4. For deployments that use a network storage location for a file store, provide the path to the directory that will be used for the message store. For deployments that use a JDBC data source as the data store, provide the JNDI name of the data source. Click **Next**. For more information about configuring a message store, see the WebSphere Application Server help.

5. Select **Resources > JMS > Queues**. For each of the queues that are listed, follow these steps to reconnect the queue to the SAS Messaging Bus:
 - a. Click the queue name.
 - b. In the Connection area, copy the queue name, such as AlertQueue, to your clipboard.
 - c. Select **Create SIB destination** from the Queue name menu.
 - d. Paste the queue name in the Identifier text field and click **Next**.
 - e. Select the cluster name from the Bus member menu and click **Next**. Click **Finish**.

Configure a Load Balancing HTTP Server

Load balancing the HTTP requests to the application server instances can be accomplished in many ways, using hardware or software. One option for load balancing HTTP requests is to configure IBM HTTP Server for load balancing. For more information about performing this configuration, see [Configuring IBM HTTP Server as a Reverse Proxy Server 9.2 Web Applications Deployed on IBM WebSphere Application Server](#) or IBM documentation.

For sites that cluster 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 HTTP server, 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 7 – Common Tasks."

Chapter 7 — Common Tasks

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.

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 how 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 load-balancing 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 load-balancing HTTP server and then click **OK**.

Change the Connection for SAS Content Server

Reconfigure SAS metadata for the SAS Content Server connection to the load-balancing HTTP server host name and port. This change is similar to the change required for each of the SAS Web applications.

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 load-balancing HTTP 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 load-balancing HTTP server host name and port.
8. Click **OK**. Confirm the Change Content Mapping dialog box.

Change the WebDAV Repository URL

Just as in the previous step, because SAS Content Server is accessed through the load-balancing HTTP server, 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 WebDAV repository provided by 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 load-balancing HTTP server.
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 servers that are hosting the SAS Content Server application.

Recommended Reading

The following URLs are current as of May 2010:

Oracle Corporation. 2008. "Using Web Server Plug-Ins with WebLogic Server." Available at http://download.oracle.com/docs/cd/E12840_01/wls/docs103/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/v92m3/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



sas

THE
POWER
TO KNOW®