**Creating a merged web server plug-in configuration file for Liberty**

Contents

Table of Contents

[Lab 2 Creating a merged web server plug-in configuration file for Liberty 3](#_Toc477186242)

[1.1 Prerequisites 4](#_Toc477186243)

[1.2 Install and set up IHS and WebSphere Plug-in 5](#_Toc477186244)

[1.3 Configure a Liberty server to generate a configuration file for the lab web server. 6](#_Toc477186245)

[1.4 Copy the server 'blue' to create a cluster of servers 10](#_Toc477186246)

[1.5 Create a merged plug-in configuration file and configure the plug-in between IHS and the server cluster 12](#_Toc477186247)

[1.5.1 About plugin-cfg.xml 13](#_Toc477186248)

[1.6 Testing the plug-in 13](#_Toc477186249)

[1.7 Cleanup 14](#_Toc477186250)

[Notices 16](#_Toc477186251)

[Appendix A. Trademarks and copyrights 18](#_Toc477186252)

# Lab 2 Creating a merged web server plug-in configuration file for Liberty

The WebSphere Plug-in can be used with an Apache web server to route HTTP requests to applications running in Liberty servers. It is common to provide workload balancing and failover of application requests by running the same application in multiple application server processes, a pattern referred to as an application cluster. In such a topology, the web server plug-in needs to share the requests across all the application servers in the cluster, and it can be directed to do so by using a configuration that contains information about all of the application servers. This is known as a merged plug-in configuration because it involves merging routing information from multiple application servers.

There are three ways to provide the merged plug-in configuration to the web server:

1. Create the plug-in configuration for each application server, use a utility to merge these configurations into a single file, then copy it to the web server installation. This method can be used for Liberty servers from any edition of WebSphere that are not managed in a Liberty collective. This is the method that we will use in this lab.

2. Manage the application server in a Liberty collective, and use the dynamicRouting feature in the collective controller process to provide the information from each application server to the web server plug-in. For this method the web server plug-in configuration only needs to contain information about the collective controller process; the plug-in then contacts the controller to obtain information about all the servers in the collective, and will direct HTTP requests to all of the applications in all the application servers. This method requires a WebSphere Network Deployment (ND) license for the collective controller host, but the application servers running the applications can be from any WebSphere edition (Liberty Core, WAS Base or ND). An example of this method is shown in management lab 3\_DynamicRouting.

3. Organize the application servers into managed clusters within a Liberty collective (using the clusterMember feature in each server), and use the ClusterManagerMBean in the controller process to generate a merged plug-in configuration for all the servers in a given cluster. This method requires a WebSphere Network Deployment (ND) license for all the Liberty servers. An example of this method is shown in management lab 1\_Jython.

In this lab we will create a merged plug-in configuration file that will allow a web server to spray HTTP requests across three liberty server that are running the same sample application. We will make use of the ability of the Liberty server to automatically generate a plug-in configuration file (this was added in fix pack 16.0.0.3), and a utility to merge those files (also added in 16.0.0.3). We will also perform some updates to the application server configuration which result in generation of an updated plug-in configuration file, and see the effect of session affinity on routing in the application cluster. The steps will be as follows:

1. Install the IHS and WebSphere plug-in (you may have already done this for a previous lab).
2. Unzip and run a provided server configuration, including two simple applications that display application server information.
3. Examine the server configuration using the admin center, and adjust the web server plugin configuration to match your web server plugin installation
4. Clone the server to produce three servers with the same application but different names, and adjust the server HTTP port numbers to make them unique, so the servers can be run on the same host without port conflicts
5. Start all the servers to test the application and generate their plugin-cfg.xm files.
6. Examine the plug-in configuration files generated by the servers, then merge them into a single file with the pluginUtility
7. Configure the web server plug-in between httpd and Liberty using the merged plug-in configuration file, and test the application cluster.
8. Compare the behavior of the plug-in when sending requests to the two sample applications, and the effect of using an HTTP session in one of the applications.

Please refer to the following table for file and resource location references on different operating systems. This lab runs only on the Windows and Linux platforms as the WebSphere plugin is not provided for Mac OSX.

|  |  |  |
| --- | --- | --- |
| Location Ref. | OS | Absolute Path |
| *{LAB\_HOME}* | Windows | C:/*WLP\_<version>* |
| Linux | ~/*WLP\_<version>* |
| Mac OSX | not applicable |

## Prerequisites

The following preparation must be completed prior to beginning this lab:

1. Complete the Setup lab to to install the JRE and Liberty runtime. You do not need to install the WDT developer tools (steps 0.5 and 0.6 in the Setup lab).

2. Note that this lab runs the IHS web server on port 9180. To check whether this port is available on your system, you can use the netstat command as follows:

Linux: **netstat -a | grep 9180**

Windows: **netstat -na | find "9180"**

If port 9180 is already in use on your host, select a different (available) port for this lab. Remember to use your chosen port numbers instead of port 9180 and 8443 for the rest of this lab

## Install and set up IHS and WebSphere Plug-in

In the following steps we will add a separate IHS server. If you have already installed IHS and the plug-in for another lab you do not need to repeat this section, but should go to section 1.3 to configure IHS for SSL.

This lab includes a copy of IHS and the WebSphere plug-in, which can be found under *{LAB\_HOME}*.

1. Unzip the {LAB\_HOME}\IHS-<version>-<platform>.zip for your system into the ${LAB\_HOME} directory. This will create a {LAB\_HOME}\IHS directory where IHS is now installed. This contains a copy of IBM HTTP Server, which is the Apache web server that is provided with WebSphere. The zip also has the WebSphere web server plug-in pre-installed in IHS.
2. Run the IHS postinstall script to perform some host-specific setup of IHS. Note this script creates the directory that we will use later for the plug-in log files: {LAB\_HOME}\IHS\plugin\logs\webserver1.

Linux: **{LAB\_HOME}/IHS/postinstall.sh**

Windows: **{LAB\_HOME}\IHS\postinstall.bat**

Note: If he following statement occurs, it is benign. **“postinst: Could not reliably determine the server’s fully qualified domain name, using 127.0.0.1 for ServerName”**

1. Windows only: install IHS as a service using this command:

**{LAB\_HOME}\IHS\bin\httpd.exe -k install -n "IBM HTTP Server V9.0"**

The output should be similar to this:

****

1. Change the default HTTP port number for IHS by editing the main configuration file, *{LAB\_HOME}\IHS\conf\httpd.conf* as follows

change the line

Listen 0.0.0.0:80

to use port 9180 (or an alternative port, if 9180 is not available on your system):

Listen 9180

1. Change the directory to *{LAB\_HOME}*\IHS\bin

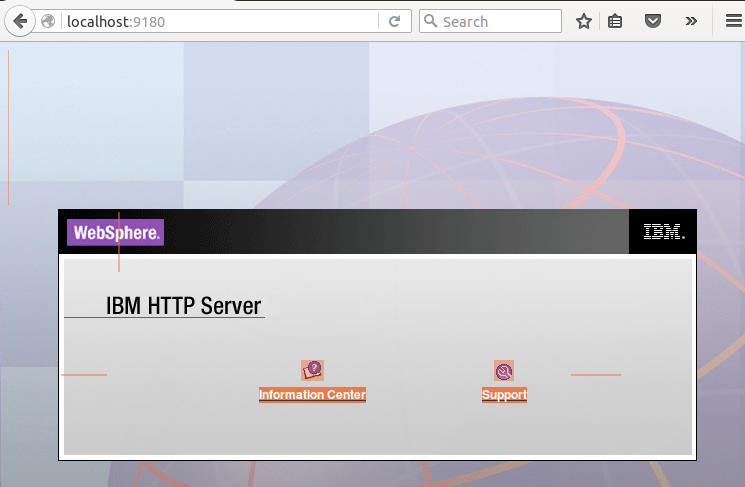
**cd *{LAB\_HOME}*\IHS\bin**

and start IHS with this command

Linux: **apachectl -k start -f conf/httpd.conf**

Windows: **httpd -k start -f {LAB\_HOME}\IHS\conf\httpd.conf**

1. Check IHS is running: point your browser to <http://localhost:9180>. It should show the screen below which proves that IHS server is up and running. This screen comes from *{LAB\_HOME}*\IHS\htdocs\index.html. (This htdocs directory is the “document root” for the Apache Httpd server.)



1. Stop IHS with the following command:

Linux: **apachectl -k stop**

Windows: **httpd -k stop**

## Configure a Liberty server to generate a configuration file for the lab web server.

For this lab we will set up a cluster of three servers. All three servers will contain the same two applications, and have the same configuration except for the HTTP port values (which need to be unique so they can all run on the same host). Starting from a server configuration provided for the lab, you will first use the Admin Center (admin GUI) to create some configuration elements that control most of the values that the server will put into the generated plugin-cfg.xml file. Using the Admin Center Config tool is a good way to see all the configuration elements and attributes available in the server.

You will then edit the server.xml directly to remove the Admin Center feature, as we don't want that application to be included in our deployed cluster.

1. Unzip server-lab6.zip into the **{LAB\_HOME}\wlp\usr\servers** directory, to create a server called 'blue'.
2. Start the server with the command:

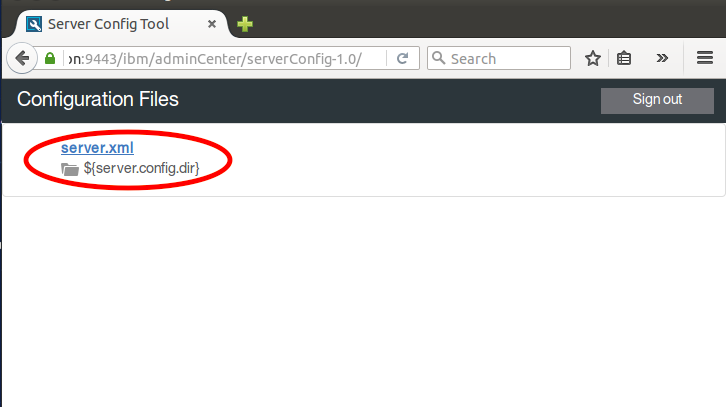
**{LAB\_HOME}\wlp\bin\server run blue**

1. Test that the application is running by entering this in your browser: <http://localhost:9080/WhereAmI>
2. Set the correct configuration properties for the liberty server to use when generating the web server plugin configuration file. Using the Admin Center Config tool: open the tool by entering this in your browser:

<http://localhost:9080/ibm/adminCenter/serverConfig-1.0/>

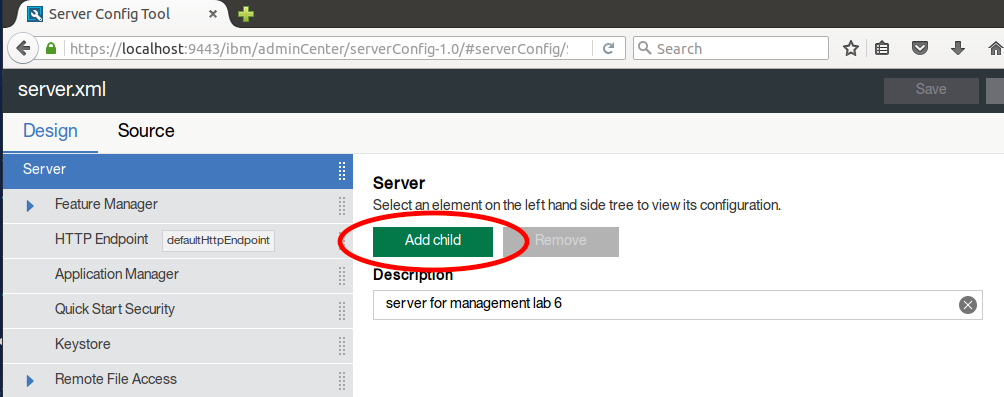
If the browser displays a warning about the connection being untrusted, add the security exception.

1. Login to the Admin Center using the user (**admin**) and password (**adminpwd**) that are specified in the quickStartSecurity configuration in the server.xml (the password is deliberately left in plain text so you can see it).
2. In the Configuration Files screen, select server.xml. This will open the server configuration editor. You can switch between the Design and Source views in the editor.

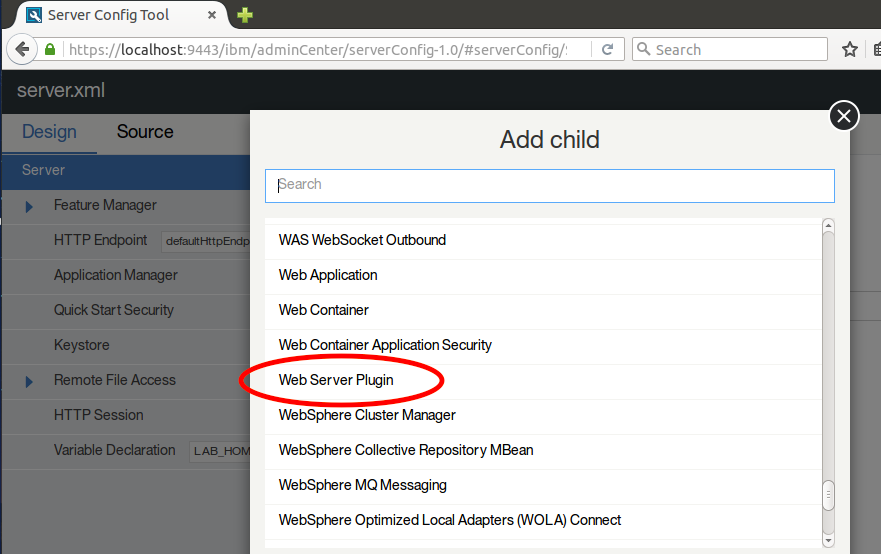


1. In the configuration editor **Design view**, click on the **Server** scope and then click on **Add child**.

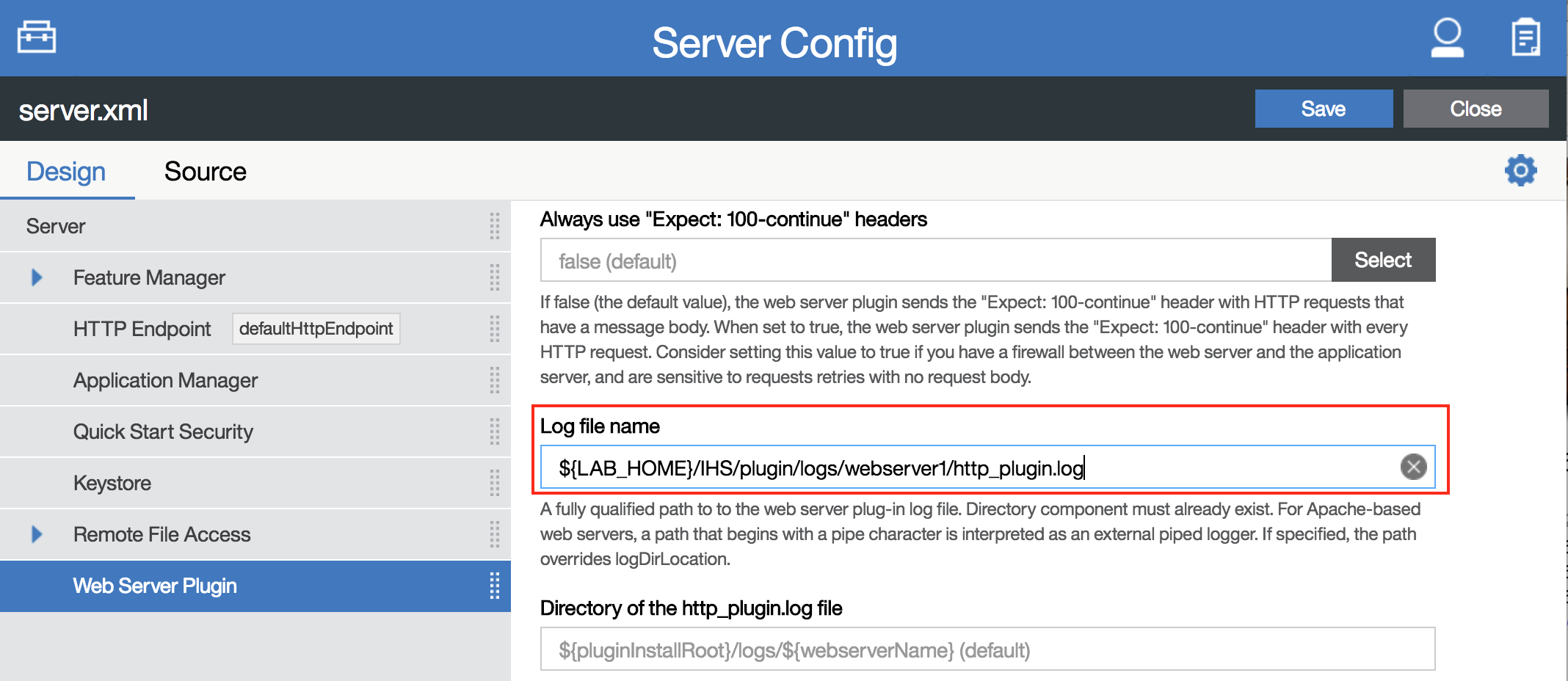
For this lab, we need to override some of the default configuration settings for the web server plugin so that the appropriate values are used for theplugin-cfg.xml file that is generated by the Liberty server.



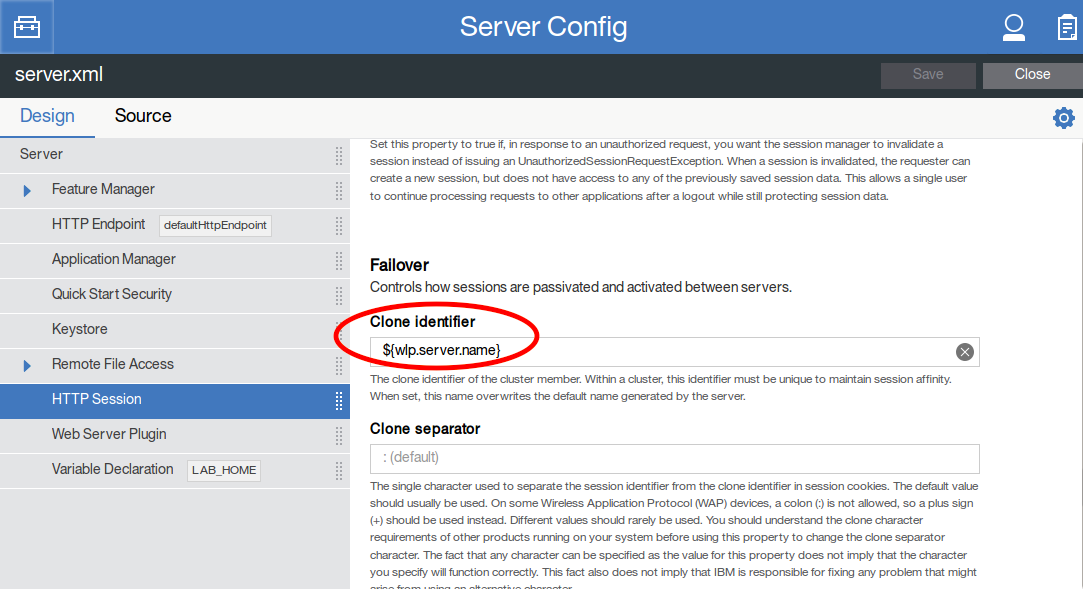
1. Double-click on **Web Server Plugin**.



1. Change the first attribute, the **Plugin Install Location**, to match the plugin directory in your IHS install, which should be **${LAB\_HOME}\IHS\plugin**. You can either declare LAB\_HOME as a variable in the server configuration (add a server child element type Variable Declaration) or enter the absolute path for ${LAB\_HOME} in the **Plugin install location**. Save your changes by clicking the Save button.
2. Go back to the **Web Server Plugin** configuration element, and scroll down to the **Log file name** property. This will be used by the plug-in to create a log file with the specified name, but the specified directory must already exist. The postinstall script that you ran in step 1.2.2 above created the directory ${LAB\_HOME}/IHS/plugin/logs/webserver1 so we will specify the log file name as **${LAB\_HOME}/IHS/plugin/logs/webserver1/http\_plugin.log**.



1. This lab will demonstrate HTTP session affinity, which uses the CloneID of each server to determine the server for which a request has affinity. These identifiers need to be unique for each server in a given cluster. By default, a Liberty server will generate a unique identifier to use as its CloneID, and will persist that in its workarea. The workarea is in the local file system, by default under the server configuration/output directory. If the workarea is deleted, or the server is started with the --clean option, the CloneId is lost and will be replaced with a new value as the server is restarted, requiring regeneration or editing of the plugin-cfg.xml information for that server in order to re-establish affinity for requests to that server. Setting a value for the CloneID in the server's configuration avoids the risk of losing a generated CloneID so is a good practice if you can determine values that will be unique for each server within the cluster. In this case we are going to create servers with unique names, so we will use the server name as the CloneID value. Click on the top-level **Server** element, and **Add child** of type **HTTP Session**. Set the **Clone identifier** value in the HTTP Session configuration to the variable that liberty uses for the server name: **${wlp.server.name}**.



1. Click the **Save** button to save the configuration.
2. You have now finished using the Admin Center and can remove it before cloning your server to create the cluster. Close your browser window and open the **{LAB\_HOME}\wlp\usr\servers\blue\server.xml** file in an editor.
3. Delete (or comment out) the following line:

<feature>adminCenter-1.0</feature>

1. Look at the plugin-cfg.xml file that was automatically generated by the server *blue*. It will be in the **{server.output.dir}\logs\state** directory. ${server.output.dir}/ in this case is **{LAB\_HOME}\wlp\usr\servers\blue**. If you used the $(LAB\_HOME} variable in the **Log file name** property, make sure it has been correctly expanded to the fully qualified path. See that the **CloneID** value has been resolved to the actual server name of *blue*. Each time you make changes to the web server plugin configuration or the HTTP session configuration, the server will regenerate the plugin-cfg.xml for the server in the ${server.config.dir}\logs\state directory. After generating the file, the server writes this message to the messages.log file:

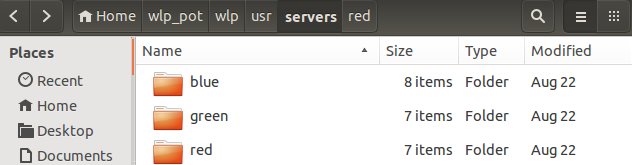
SRVE9103I: A configuration file for a web server plugin was automatically generated for this server at /home/liberty/wlp\_pot/wlp/usr/servers/blue/logs/state/plugin-cfg.xml.

## Copy the server 'blue' to create a cluster of servers

Now that the configuration is correct for the first server, we can copy it to create more servers with the same applications.

1. Stop the server with the command **{LAB\_HOME}\wlp\bin\server stop blue**
2. Copy the **{LAB\_HOME}\wlp\usr\servers\blue** directory and rename the copied directory to **green**.
3. Copy the **{LAB\_HOME}\wlp\usr\servers\blue** directory again and rename the copied directory to **red**.

You should now have three server directories, named blue, green and red



1. Edit the server.xml files for the **green** and **red** servers to change the HTTP ports to unique values, so that the port numbers won't conflict when they are all run on the same host. You can do this by editing the server.xml files in a file editor.

For the **green** server use these values:

**<httpEndpoint host="\*" httpPort="9081" httpsPort="9444" id="defaultHttpEndpoint"/>**

and for the red server use these:

**<httpEndpoint host="\*" httpPort="9082" httpsPort="9445" id="defaultHttpEndpoint"/>**

1. Add a virtual host definition to each server.xml. This entry controls which ports a client can use to send requests to the server. In each case the virtual host entry should reference the **defaultHttpEndpoint**, which is being used by the sample applications, and have child host alias elements that specify the HTTP port of the liberty server (so that requests can still be made directly to the server) and the port of the IHS server, which we set earlier to be **9180**. For convenience we will add the complete set of ports, for all server, to the server.xml of each server:

Add this to the server.xml file for the blue, green and red servers:

**<virtualHost id="default\_host" allowFromEndpointRef="defaultHttpEndpoint">**

**<hostAlias>\*:9080</hostAlias>**

**<hostAlias>\*:9081</hostAlias>**

**<hostAlias>\*:9082</hostAlias>**

**<hostAlias>\*:9180</hostAlias>**

**</virtualHost>**

5. Start all the servers with the --clean option to clear out the copied workarea:

**{LAB\_HOME}/wlp/bin/server run blue --clean**

**{LAB\_HOME}/wlp/bin/server run green --clean**

**{LAB\_HOME}/wlp/bin/server run red --clean**

6. Make sure you can access the demo applications in all the servers by invoking them directly. Use these URLs in your browser:

<http://localhost:9080/WhereAmI>

<http://localhost:9080/WhereAmIWithSession>

<http://localhost:9081/WhereAmI>

<http://localhost:9081/WhereAmIWithSession>

<http://localhost:9082/WhereAmI>

<http://localhost:9082/WhereAmIWithSession>

## Create a merged plug-in configuration file and configure the plug-in between IHS and the server cluster

Each server has generated its plugin-cfg.xml file in its **{server.output.dir}\logs\state** directory. There is a pluginUtility command in the **{LAB\_HOME}\wlp\bin** directory that can merge these files into a single plugin-cfg.xml file. There are two ways to specify the input files for the utility: as a comma-separated list of fully qualified file names, or as a directory that contains all the files to be merged (in which case you would need to rename those files so they can all go in the same directory). For this lab we will use the first method.

1. From the {LAB\_HOME}\wlp\usr\servers\blue directory, run the merge utility with this command to see the options:

**{LAB\_HOME}\wlp\bin\pluginUtility help**

2. To see the help for the merge action, enter

**{LAB\_HOME}\wlp\bin\pluginUtility help merge**

3. Merge the files for the three servers:

**{LAB\_HOME}\wlp\bin\pluginUtility merge --sourcePath=logs\state\plugin-cfg.xml,..\green\logs\state\plugin-cfg.xml,..\red\logs\state\plugin-cfg.xml --targetPath=plugin-cfg.xml**

4. Examine the output file **{LAB\_HOME}\wlp\usr\servers\blue\plugin-cfg.xml**.

### About plugin-cfg.xml

Notice the **<Route>** directive, which ties the **VirtualHostGroup**, **UriGroup**, and **ServerCluster** elements together. The **<Route>** directive instructs the plug-in to forward requests for URLs that match the **UriGroup** (meaning, URLs that match the application context routes in the servers, like **"/WhereAmI/\*")** and requests that are sent to hosts in the **VirtualHostGroup** (meaning, any requests that arrive on port 9180); the requests should be forwarded to a WebSphere server in **ServerCluster** (meaning, the servers that listen on ports 9080, 9081 and 9082 – our Liberty servers). In other words, the plug-in will cause IHS to forward any requests that belong to our application to Liberty, and it will handle all other requests itself.

The plugin-cfg.xml file is directly tied to your Liberty topology. If you make changes you to the topology, you have to update plugin-cfg.xml. If you were to add more applications to your Liberty server, you would have to copy the regenerated plug-in configuration file to the IHS install in order to access the new applications through IHS. The new plugin-cfg.xml file would include your new applications in **UriGroup**, so the plug-in would forward those requests to Liberty as well as requests for the SimpleSSL application. In the case of a Liberty cluster, you would also have to re-generate a new plug-in configuration file whenever you add or remove cluster members, to update the **ServerCluster** directives. If you manage your Liberty servers in a Collective you can use the Dynamic Routing feature to make those routing changes automatically; that is shown in Lab 3\_Dymanic Routing.

## Testing the plug-in

1. Copy the merged plugin-cfg.xml file to the **{LAB\_HOME}\IHS\plugin\config\webserver1** directory.

2. Start the web server with this command:

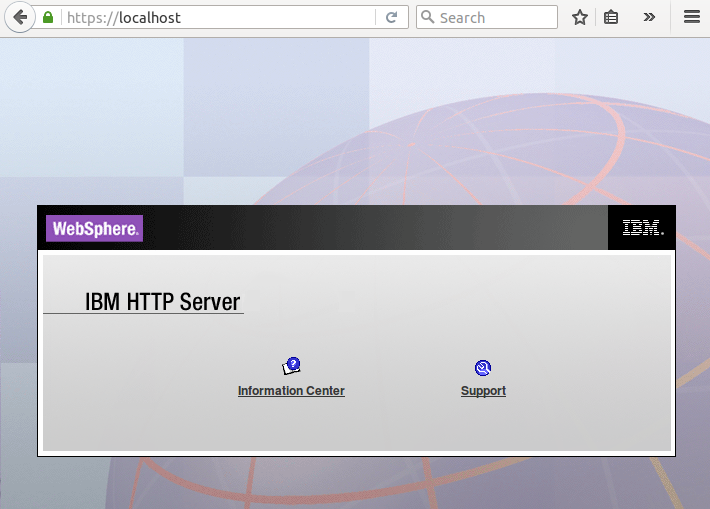
Linux: **apachectl -k start -f conf/httpd.conf**

Windows: **httpd -k start -f {LAB\_HOME}\IHS\conf\httpd.conf**

3. Check that IHS is running by entering this into your browser:

<http://localhost:9180>

You should see the IHS welcome screen displayed in your browser:



4. Check that the web server plug-in has started by looking for an http\_plugin.log file created in the logdir:

**{LAB\_HOME}\IHS\plugin\logs\webserver1\http\_plugin.log**

5. Access the first application through the web server using this URL:

[http://localhost:9180/WhereAmI](http://localhost/WhereAmI)

Refresh the browser several times and see the plug-in performing the round robin load balancing around the servers in the cluster.

6. Access the second application, which creates an HTTP session, using this URL:

<http://localhost:9180/WhereAmIWithSession>

Refresh the browser and note that all the requests go to the same server; this is the session affinity function. Try stopping that server and retrying the request. What happens? You should see that the session affinity causes all the requests to go to that server, unless you stop it and then the request 'fails over' to a different server and establishes affinity there.

Note: If still going in round robin, the browser may need to be restarted to clean out the session already created.

## Cleanup

1. Stop the liberty servers using the following command for each server name: **{LAB\_HOME}/wlp/bin/server stop <server\_name>**

2. Stop the web server using the following command:

3. Delete any parts of the lab that you don't want to keep.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation  
Licensing  
2-31 Roppongi 3-chome, Minato-ku  
Tokyo 106-0032, Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:** INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental. All references to fictitious companies or individuals are used for illustration purposes only.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

1. Trademarks and copyrights

The following terms are trademarks of International Business Machines Corporation in the United States, other countries, or both:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| IBM | AIX | CICS | ClearCase | ClearQuest | Cloudscape |  |
| Cube Views | DB2 | developerWorks | DRDA | IMS | IMS/ESA |  |
| Informix | Lotus | Lotus Workflow | MQSeries | OmniFind |  |  |
| Rational | Redbooks | Red Brick | RequisitePro | System i |  |  |
| *System z* | *Tivoli* | *WebSphere* | *Workplace* | *System p* |  |  |

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

ITIL is a registered trademark, and a registered community trademark of The Minister for the Cabinet Office, and is registered in the U.S. Patent and Trademark Office.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.



© Copyright IBM Corporation 2018.

The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. This information is based on current IBM product plans and strategy, which are subject to change by IBM without notice. Product release dates and/or capabilities referenced in these materials may change at any time at IBM’s sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way.

IBM, the IBM logo and ibm.com are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml.

